FreeBSD kernel kern code Reference Manual

Generated by Doxygen 1.4.7

Sat Feb 24 14:36:39 2007

Contents

Chapter 1

FreeBSD kernel kern code Main Page

IMPORTANT: This API documentation may contain both functions which are public and functions that are for internal use only. Since we have not reviewed every part of the documentation yet, *some internal functions are not marked as such*. Until we finish reviewing the API documentation and add appropriate comments to functions which are only for internal use, you should take this into account. In case you want to use a function of this kernel subsystem in another kernel subsystem you should search for precedence of use outside this subsystem. If the function is not used outside this subsystem you should ask on the mailinglists about it, else you risk breaking something.

Chapter 2

FreeBSD kernel kern code Module Index

2.1 FreeBSD kernel kern code Modules

Here is a list of all modules:

Chapter 3

FreeBSD kernel kern code Directory Hierarchy

3.1 FreeBSD kernel kern code Directories

This directory hierarchy is sorted roughly, but not completely, alphabetically:

Chapter 4

FreeBSD kernel kern code Data Structure Index

4.1 FreeBSD kernel kern code Data Structures

Here are the data structures with brief descriptions:

Chapter 5

FreeBSD kernel kern code File Index

5.1 FreeBSD kernel kern code File List

Here is a list of all files with brief descriptions:

Chapter 6

FreeBSD kernel kern code Module Documentation

6.1 bus - KObj methods for drivers of devices with children

A set of methods required device drivers that support child devices.

Variables

- INTERFACE bus

6.1.1 Detailed Description

A set of methods required device drivers that support child devices.

6.1.2 Variable Documentation

6.1.2.1 INTERFACE bus

Definition at line 37 of file bus_if.m.

6.2 device - KObj methods for all device drivers

A basic set of methods required for all device drivers.

Variables

- INTERFACE device

6.2.1 Detailed Description

A basic set of methods required for all device drivers.

The device interface is used to match devices to drivers during autoconfiguration and provides methods to allow drivers to handle system-wide events such as suspend, resume or shutdown.

6.2.2 Variable Documentation

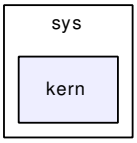
6.2.2.1 INTERFACE device

Definition at line 40 of file device_if.m.

Chapter 7

FreeBSD kernel kern code Directory Documentation

7.1 /usr/src/sys/kern/ Directory Reference



Files

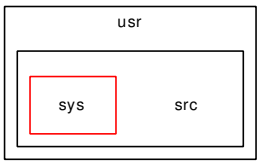
- file bus_if.m
- file clock_if.m
- file cpufreq_if.m
- file device_if.m
- file imgact_aout.c
- file imgact_elf.c
- file imgact_elf32.c
- file imgact_elf64.c
- file imgact_gzip.c
- file imgact_shell.c
- file inflate.c
- file init_main.c
- file init_sysent.c
- file kern_acct.c
- file kern_acl.c
- file kern_alq.c
- file kern_clock.c
- file kern_condvar.c
- file kern_conf.c
- file kern_context.c

- file kern_cpu.c
- file kern_descrip.c
- file kern_environment.c
- file kern_event.c
- file kern_exec.c
- file kern_exit.c
- file kern_fork.c
- file kern_idle.c
- file kern_intr.c
- file kern_jail.c
- file kern_kse.c
- file kern_kthread.c
- file kern_ktr.c
- file kern_ktrace.c
- file kern_linker.c
- file kern_lock.c
- file kern_lockf.c
- file kern_malloc.c
- file kern_mbuf.c
- file kern_mib.c
- file kern_module.c
- file kern_mtxpool.c
- file kern_mutex.c
- file kern_ntptime.c
- file kern_physio.c
- file kern_pmc.c
- file kern_poll.c
- file kern_priv.c
- file kern_proc.c
- file kern_prot.c
- file kern_resource.c
- file kern_rwlock.c
- file kern_sema.c
- file kern_shutdown.c
- file kern_sig.c
- file kern_subr.c
- file kern_switch.c
- file kern_sx.c
- file kern_synch.c
- file kern_syscalls.c
- file kern_sysctl.c
- file kern_tc.c
- file kern_thr.c
- file kern_thread.c
- file kern_time.c
- file kern_timeout.c
- file kern_umtx.c
- file kern_uuid.c
- file kern_xxx.c
- file ksched.c

- file link_elf.c
- file link_elf_obj.c
- file linker_if.m
- file Make.tags.inc
- file md4c.c
- file md5c.c
- file p1003_1b.c
- file posix4_mib.c
- file sched_4bsd.c
- file sched_core.c
- file sched_ule.c
- file serdev_if.m
- file subr_acl_posix1e.c
- file subr_autoconf.c
- file subr_blist.c
- file subr_bus.c
- file subr_clock.c
- file subr_devstat.c
- file subr_disk.c
- file subr_eventhandler.c
- file subr_fattime.c
- file subr_firmware.c
- file subr_hints.c
- file subr_kdb.c
- file subr_kobj.c
- file subr_lock.c
- file subr_log.c
- file subr_mbpool.c
- file subr_mchain.c
- file subr_module.c
- file subr_msgbuf.c
- file subr_param.c
- file subr_pcpu.c
- file subr_power.c
- file subr_prf.c
- file subr_prof.c
- file subr_rman.c
- file subr_rtc.c
- file subr_sbuf.c
- file subr_scanf.c
- file subr_sleepqueue.c
- file subr_smp.c
- file subr_stack.c
- file subr_taskqueue.c
- file subr_trap.c
- file subr_turnstile.c
- file subr_unit.c
- file subr_witness.c
- file sys_generic.c
- file sys_pipe.c

- file sys_process.c
- file sys_socket.c
- file syscalls.c
- file systrace_args.c
- file sysv_ipc.c
- file sysv_msg.c
- file sysv_sem.c
- file sysv_shm.c
- file tty.c
- file tty_compat.c
- file tty_conf.c
- file tty_cons.c
- file tty_pts.c
- file tty_pty.c
- file tty_subr.c
- file tty_tty.c
- file uipc_accf.c
- file uipc_cow.c
- file uipc_debug.c
- file uipc_domain.c
- file uipc_mbuf.c
- file uipc_mbuf2.c
- file uipc_mqueue.c
- file uipc_sem.c
- file uipc_sockbuf.c
- file uipc_socket.c
- file uipc_socket2.c
- file uipc_syscalls.c
- file uipc_usrreq.c
- file vfs_aio.c
- file vfs_bio.c
- file vfs_cache.c
- file vfs_cluster.c
- file vfs_default.c
- file vfs_export.c
- file vfs_extattr.c
- file vfs_hash.c
- file vfs_init.c
- file vfs_lookup.c
- file vfs_mount.c
- file vfs_subr.c
- file vfs_syscalls.c
- file vfs_vnops.c

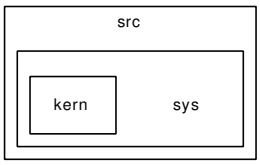
7.2 /usr/src/ Directory Reference



Directories

- directory sys

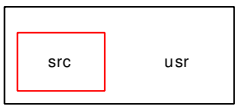
7.3 /usr/src/sys/ Directory Reference



Directories

- directory kern

7.4 /usr/ Directory Reference



Directories

- directory src

Chapter 8

FreeBSD kernel kern code Data Structure Documentation

8.1 __getcwd_args Struct Reference

Data Fields

- u_char * buf
- u_int buflen

8.1.1 Detailed Description

Definition at line 680 of file vfs_cache.c.

8.1.2 Field Documentation

8.1.2.1 u_char* __getcwd_args::buf

Definition at line 681 of file vfs_cache.c.

Referenced by __getcwd(), and systrace_args().

8.1.2.2 u_int __getcwd_args::buflen

Definition at line 682 of file vfs_cache.c.

Referenced by __getcwd(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_cache.c

8.2 __getrlimit_args Struct Reference

Data Fields

- u_int which
- rlimit * rlp

8.2.1 Detailed Description

Definition at line 766 of file kern_resource.c.

8.2.2 Field Documentation

8.2.2.1 struct rlimit* __getrlimit_args::rlp

Definition at line 768 of file kern_resource.c.

Referenced by systrace_args().

8.2.2.2 u_int __getrlimit_args::which

Definition at line 767 of file kern_resource.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_resource.c

8.3 __mac_execve_args Struct Reference

Data Fields

- char * fname
- char ** argv
- char ** envv
- mac * mac_p

8.3.1 Detailed Description

Definition at line 194 of file kern_exec.c.

8.3.2 Field Documentation

8.3.2.1 char** __mac_execve_args::argv

Definition at line 196 of file kern_exec.c.

Referenced by __mac_execve(), and systrace_args().

8.3.2.2 char** __mac_execve_args::envv

Definition at line 197 of file kern_exec.c.

Referenced by __mac_execve(), and systrace_args().

8.3.2.3 char* __mac_execve_args::fname

Definition at line 195 of file kern_exec.c.

Referenced by __mac_execve(), and systrace_args().

8.3.2.4 struct mac* __mac_execve_args::mac_p

Definition at line 198 of file kern_exec.c.

Referenced by __mac_execve(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_exec.c

8.4 __semctl_args Struct Reference

Data Fields

- int semid
- int semnum
- int cmd
- semun * arg

8.4.1 Detailed Description

Definition at line 539 of file sysv_sem.c.

8.4.2 Field Documentation

8.4.2.1 union semun* __semctl_args::arg

Definition at line 543 of file sysv_sem.c.

Referenced by __semctl(), and systrace_args().

8.4.2.2 int __semctl_args::cmd

Definition at line 542 of file sysv_sem.c.

Referenced by __semctl(), and systrace_args().

8.4.2.3 int __semctl_args::semid

Definition at line 540 of file sysv_sem.c.

Referenced by __semctl(), and systrace_args().

8.4.2.4 int __semctl_args::semnum

Definition at line 541 of file sysv_sem.c.

Referenced by __semctl(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sysv_sem.c

8.5 __setrlimit_args Struct Reference

Data Fields

- u_int which
- rlimit * rlp

8.5.1 Detailed Description

Definition at line 619 of file kern_resource.c.

8.5.2 Field Documentation

8.5.2.1 struct rlimit* __setrlimit_args::rlp

Definition at line 621 of file kern_resource.c.

Referenced by systrace_args().

8.5.2.2 u_int __setrlimit_args::which

Definition at line 620 of file kern_resource.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_resource.c

8.6 abort2_args Struct Reference

Data Fields

- char * why
- int nargs
- void ** args

8.6.1 Detailed Description

Definition at line 549 of file kern_exit.c.

8.6.2 Field Documentation

8.6.2.1 void** abort2_args::args

Definition at line 552 of file kern_exit.c.

Referenced by abort2(), and systrace_args().

8.6.2.2 int abort2_args::nargs

Definition at line 551 of file kern_exit.c.

Referenced by abort2(), and systrace_args().

8.6.2.3 char* abort2_args::why

Definition at line 550 of file kern_exit.c.

Referenced by abort2(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_exit.c

8.7 access_args Struct Reference

Data Fields

- char * path
- int flags

8.7.1 Detailed Description

Definition at line 1864 of file vfs_syscalls.c.

8.7.2 Field Documentation

8.7.2.1 int access_args::flags

Definition at line 1866 of file vfs_syscalls.c.

Referenced by systrace_args().

8.7.2.2 char* access_args::path

Definition at line 1865 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.8 adjtime_args Struct Reference

Data Fields

- timeval * delta
- timeval * olddelta

8.8.1 Detailed Description

Definition at line 925 of file kern_ntptime.c.

8.8.2 Field Documentation

8.8.2.1 struct timeval* adjtime_args::delta

Definition at line 926 of file kern_ntptime.c.

Referenced by adjtime(), and systrace_args().

8.8.2.2 struct timeval* adjtime_args::olddelta

Definition at line 927 of file kern_ntptime.c.

Referenced by adjtime(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_ntptime.c

8.9 aiocblist Struct Reference

8.9.1 Detailed Description

Definition at line 219 of file vfs_aio.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_aio.c

8.10 aioliojob Struct Reference

Data Fields

- int lioj_flags
- int lioj_count
- int lioj_finished_count
- sigevent lioj_signal

8.10.1 Detailed Description

Definition at line 263 of file vfs_aio.c.

8.10.2 Field Documentation

8.10.2.1 int aioliojob::lioj_count

Definition at line 265 of file vfs_aio.c.

Referenced by aio_aqueue(), aio_bio_done_notify(), aio_free_entry(), aio_proc_rundown(), aio_qphysio(), and do_lio_listio().

8.10.2.2 int aioliojob::lioj_finished_count

Definition at line 266 of file vfs_aio.c.

Referenced by aio_bio_done_notify(), aio_free_entry(), aio_proc_rundown(), and do_lio_listio().

8.10.2.3 int aioliojob::lioj_flags

Definition at line 264 of file vfs_aio.c.

Referenced by aio_bio_done_notify(), do_lio_listio(), and filt_lio().

8.10.2.4 struct sigevent aioliojob::lioj_signal

Definition at line 267 of file vfs_aio.c.

Referenced by aio_bio_done_notify(), and do_lio_listio().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_aio.c

8.11 aiothreadlist Struct Reference

Data Fields

- int aiothreadflags

8.11.1 Detailed Description

Definition at line 254 of file vfs_aio.c.

8.11.2 Field Documentation

8.11.2.1 int aiothreadlist::aiothreadflags

Definition at line 255 of file vfs_aio.c.

Referenced by aio_daemon(), aio_kick(), and aio_kick_nowait().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_aio.c

8.12 alq Struct Reference

Data Fields

- int aq_entmax
- int aq_entlen
- char * aq_entbuf
- int aq_flags
- mtx aq_mtx
- vnode * aq_vp
- ucred * aq_cred
- ale * aq_first
- ale * aq_entfree
- ale * aq_entvalid

8.12.1 Detailed Description

Definition at line 51 of file kern_alq.c.

8.12.2 Field Documentation

8.12.2.1 struct ucred* alq::aq_cred

Definition at line 58 of file kern_alq.c.

Referenced by alq_doio(), and alq_shutdown().

8.12.2.2 char* alq::aq_entbuf

Definition at line 54 of file kern_alq.c.

Referenced by alq_close().

8.12.2.3 struct ale* alq::aq_entfree

Definition at line 60 of file kern_alq.c.

Referenced by alq_doio(), and alq_get().

8.12.2.4 int alq::aq_entlen

Definition at line 53 of file kern_alq.c.

Referenced by alq_doio(), and alq_write().

8.12.2.5 int alq::aq_entmax

Definition at line 52 of file kern_alq.c.

8.12.2.6 struct ale* alq::aq_entvalid

Definition at line 61 of file kern_alq.c.

Referenced by alq_doio(), and alq_post().

8.12.2.7 struct ale* alq::aq_first

Definition at line 59 of file kern_alq.c.

Referenced by alq_close().

8.12.2.8 int alq::aq_flags

Definition at line 55 of file kern_alq.c.

Referenced by ald_deactivate(), alq_doio(), alq_flush(), alq_get(), alq_post(), and alq_shutdown().

8.12.2.9 struct mtx alq::aq_mtx

Definition at line 56 of file kern_alq.c.

Referenced by alq_close().

8.12.2.10 struct vnode* alq::aq_vp

Definition at line 57 of file kern_alq.c.

Referenced by alq_doio(), and alq_shutdown().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_alq.c

8.13 cf_saved_freq Struct Reference

Data Fields

- cf_level level
- int priority

8.13.1 Detailed Description

Definition at line 60 of file kern_cpu.c.

8.13.2 Field Documentation

8.13.2.1 struct cf_level cf_saved_freq::level

Definition at line 61 of file kern_cpu.c.

Referenced by cf_set_method().

8.13.2.2 int cf_saved_freq::priority

Definition at line 62 of file kern_cpu.c.

Referenced by cf_set_method().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_cpu.c

8.14 cf_setting_array Struct Reference

Data Fields

- cf_setting sets [MAX_SETTINGS]
- int count

8.14.1 Detailed Description

Definition at line 78 of file kern_cpu.c.

8.14.2 Field Documentation

8.14.2.1 int cf_setting_array::count

Definition at line 80 of file kern_cpu.c.

Referenced by cpufreq_expand_set().

8.14.2.2 struct cf_setting cf_setting_array::sets[MAX_SETTINGS]

Definition at line 79 of file kern_cpu.c.

Referenced by cpufreq_expand_set().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_cpu.c

8.15 chdir_args Struct Reference

Data Fields

- char * path

8.15.1 Detailed Description

Definition at line 734 of file vfs_syscalls.c.

8.15.2 Field Documentation

8.15.2.1 char* chdir_args::path

Definition at line 735 of file vfs_syscalls.c.

Referenced by chdir(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.16 chflags_args Struct Reference

Data Fields

- char * path
- int flags

8.16.1 Detailed Description

Definition at line 2392 of file vfs_syscalls.c.

8.16.2 Field Documentation

8.16.2.1 int chflags_args::flags

Definition at line 2394 of file vfs_syscalls.c.

Referenced by systrace_args().

8.16.2.2 char* chflags_args::path

Definition at line 2393 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.17 chmod_args Struct Reference

Data Fields

- char * path
- int mode

8.17.1 Detailed Description

Definition at line 2520 of file vfs_syscalls.c.

8.17.2 Field Documentation

8.17.2.1 int chmod_args::mode

Definition at line 2522 of file vfs_syscalls.c.

Referenced by systrace_args().

8.17.2.2 char* chmod_args::path

Definition at line 2521 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.18 chown_args Struct Reference

Data Fields

- char * path
- int uid
- int gid

8.18.1 Detailed Description

Definition at line 2663 of file vfs_syscalls.c.

8.18.2 Field Documentation

8.18.2.1 int chown_args::gid

Definition at line 2666 of file vfs_syscalls.c.

Referenced by systrace_args().

8.18.2.2 char* chown_args::path

Definition at line 2664 of file vfs_syscalls.c.

Referenced by systrace_args().

8.18.2.3 int chown_args::uid

Definition at line 2665 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.19 chroot_args Struct Reference

Data Fields

- char * path

8.19.1 Detailed Description

Definition at line 825 of file vfs_syscalls.c.

8.19.2 Field Documentation

8.19.2.1 char* chroot_args::path

Definition at line 826 of file vfs_syscalls.c.

Referenced by chroot(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.20 clock_getres_args Struct Reference

Data Fields

- clockid_t clock_id
- timespec * tp

8.20.1 Detailed Description

Definition at line 301 of file kern_time.c.

8.20.2 Field Documentation

8.20.2.1 clockid_t clock_getres_args::clock_id

Definition at line 302 of file kern_time.c.

Referenced by clock_getres(), and systrace_args().

8.20.2.2 struct timespec* clock_getres_args::tp

Definition at line 303 of file kern_time.c.

Referenced by clock_getres(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_time.c

8.21 clock_gettime_args Struct Reference

Data Fields

- clockid_t clock_id
- timespec * tp

8.21.1 Detailed Description

Definition at line 184 of file kern_time.c.

8.21.2 Field Documentation

8.21.2.1 clockid_t clock_gettime_args::clock_id

Definition at line 185 of file kern_time.c.

Referenced by clock_gettime(), and systrace_args().

8.21.2.2 struct timespec* clock_gettime_args::tp

Definition at line 186 of file kern_time.c.

Referenced by clock_gettime(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_time.c

8.22 clock_settime_args Struct Reference

Data Fields

- clockid_t clock_id
- timespec * tp

8.22.1 Detailed Description

Definition at line 256 of file kern_time.c.

8.22.2 Field Documentation

8.22.2.1 clockid_t clock_settime_args::clock_id

Definition at line 257 of file kern_time.c.

Referenced by clock_settime(), and systrace_args().

8.22.2.2 struct timespec* clock_settime_args::tp

Definition at line 258 of file kern_time.c.

Referenced by clock_settime(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_time.c

8.23 clonedevs Struct Reference

8.23.1 Detailed Description

Definition at line 793 of file kern_conf.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_conf.c

8.24 close_args Struct Reference

Data Fields

- int fd

8.24.1 Detailed Description

Definition at line 964 of file kern_descrip.c.

8.24.2 Field Documentation

8.24.2.1 int close_args::fd

Definition at line 965 of file kern_descrip.c.

Referenced by close(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_descrip.c

8.25 cn_device Struct Reference

8.25.1 Detailed Description

Definition at line 88 of file tty_cons.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/tty_cons.c

8.26 cpufreq_softc Struct Reference

Data Fields

- sx lock
- cf_level curr_level
- int curr_priority

8.26.1 Detailed Description

Definition at line 66 of file kern_cpu.c.

8.26.2 Field Documentation

8.26.2.1 struct cf_level cpufreq_softc::curr_level

Definition at line 68 of file kern_cpu.c.

Referenced by cf_get_method(), cf_set_method(), and cpufreq_attach().

8.26.2.2 int cpufreq_softc::curr_priority

Definition at line 69 of file kern_cpu.c.

Referenced by cf_set_method().

8.26.2.3 struct sx cpufreq_softc::lock

Definition at line 67 of file kern_cpu.c.

Referenced by cf_get_method(), cf_levels_method(), cf_set_method(), cpufreq_attach(), cpufreq_dup_set(), cpufreq_expand_set(), and cpufreq_insert_abs().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_cpu.c

8.27 dev_event_info Struct Reference

Data Fields

- char * dei_data

8.27.1 Detailed Description

Definition at line 362 of file subr_bus.c.

8.27.2 Field Documentation

8.27.2.1 char* dev_event_info::dei_data

Definition at line 364 of file subr_bus.c.

Referenced by devread(), and sysctl_devctl_disable().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_bus.c

8.28 dev_softc Struct Reference

Data Fields

- int inuse
- int nonblock
- mtx mtx
- cv cv
- selinfo sel
- devq devq
- proc * async_proc

8.28.1 Detailed Description

Definition at line 370 of file subr_bus.c.

8.28.2 Field Documentation

8.28.2.1 struct proc* dev_softc::async_proc

Definition at line 378 of file subr_bus.c.

Referenced by devctl_queue_data(), devioctl(), and devopen().

8.28.2.2 struct cv dev_softc::cv

Definition at line 375 of file subr_bus.c.

Referenced by devclose(), devctl_queue_data(), devinit(), and devread().

8.28.2.3 struct devq dev_softc::devq

Definition at line 377 of file subr_bus.c.

Referenced by devctl_queue_data(), devinit(), devpoll(), devread(), and sysctl_devctl_disable().

8.28.2.4 int dev_softc::inuse

Definition at line 372 of file subr_bus.c.

Referenced by devclose(), and devopen().

8.28.2.5 struct mtx dev_softc::mtx

Definition at line 374 of file subr_bus.c.

Referenced by devclose(), devctl_queue_data(), devinit(), devpoll(), devread(), and sysctl_devctl_disable().

8.28.2.6 int dev_softc::nonblock

Definition at line 373 of file subr_bus.c.

Referenced by devioctl(), devopen(), and devread().

8.28.2.7 struct selinfo dev_softc::sel

Definition at line 376 of file subr_bus.c.

Referenced by devctl_queue_data(), and devpoll().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_bus.c

8.29 device Struct Reference

Implementation of device.

Data Fields

- KOBJ_FIELDS

8.29.1 Detailed Description

Implementation of device.

Definition at line 93 of file subr_bus.c.

8.29.2 Field Documentation

8.29.2.1 device::KOBJ_FIELDS

Definition at line 98 of file subr_bus.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_bus.c

8.30 driverlink Struct Reference

Data Fields

- kobj_class_t driver

8.30.1 Detailed Description

Definition at line 66 of file subr_bus.c.

8.30.2 Field Documentation

8.30.2.1 kobj_class_t driverlink::driver

Definition at line 67 of file subr_bus.c.

Referenced by bus_generic_probe(), devclass_delete_driver(), devclass_find_driver(), devclass_find_driver_internal(), devclass_get_drivers(), devclass_quiesce_driver(), device_probe_child(), and next_matching_driver().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_bus.c

8.31 dup2_args Struct Reference

Data Fields

- u_int from
- u_int to

8.31.1 Detailed Description

Definition at line 273 of file kern_descrip.c.

8.31.2 Field Documentation

8.31.2.1 u_int dup2_args::from

Definition at line 274 of file kern_descrip.c.

Referenced by dup2(), and systrace_args().

8.31.2.2 u_int dup2_args::to

Definition at line 275 of file kern_descrip.c.

Referenced by dup2(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_descrip.c

8.32 dup_args Struct Reference

Data Fields

- u_int fd

8.32.1 Detailed Description

Definition at line 294 of file kern_descrip.c.

8.32.2 Field Documentation

8.32.2.1 u_int dup_args::fd

Definition at line 295 of file kern_descrip.c.

Referenced by dup(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_descrip.c

8.33 eaccess_args Struct Reference

Data Fields

- char * path
- int flags

8.33.1 Detailed Description

Definition at line 1921 of file vfs_syscalls.c.

8.33.2 Field Documentation

8.33.2.1 int eaccess_args::flags

Definition at line 1923 of file vfs_syscalls.c.

Referenced by systrace_args().

8.33.2.2 char* eaccess_args::path

Definition at line 1922 of file vfs_syscalls.c.

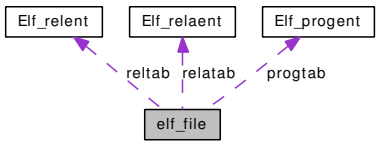
Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.34 elf_file Struct Reference

Collaboration diagram for elf_file:



Data Fields

- linker_file lf
- int preloaded
- caddr_t address
- Elf_Dyn * dynamic
- Elf_Hashelt nbuckets
- Elf_Hashelt nchains
- const Elf_Hashelt * buckets
- const Elf_Hashelt * chains
- caddr_t hash
- caddr_t strtab
- int strsz
- const Elf_Sym * symtab
- Elf_Addr * got
- const Elf_Rel * pltrel
- int pltrelsize
- const Elf_Rela * pltrela
- int pltrelasize
- const Elf_Rel * rel
- int relsize
- const Elf_Rela * rela
- int relasize
- caddr_t modptr
- const Elf_Sym * ddbsymtab
- long ddbsymcnt
- caddr_t ddbstrtab
- long ddbstrcnt
- caddr_t symbase
- caddr_t strbase
- linker_file lf
- vm_object_t object
- Elf_Shdr * e_shdr
- Elf_progent * progtab
- int nprogtab
- Elf_relaent * relatab
- int nrela
- Elf_relent * reltab
- int nrel
- Elf_Sym * ddbsymtab
- caddr_t shstrtab
- long shstrcnt

8.34.1 Detailed Description

Definition at line 69 of file link_elf.c.

8.34.2 Field Documentation

8.34.2.1 caddr_t elf_file::address

Definition at line 72 of file link_elf.c.

Referenced by elf_lookup(), link_elf_init(), link_elf_link_common_finish(), link_elf_link_preload(), link_elf_load_file(), link_elf_reloc_local(), link_elf_search_symbol(), link_elf_symbol_values(), link_elf_unload_file(), parse_dynamic(), and relocate_file().

8.34.2.2 const Elf_Hashelt* elf_file::buckets

Definition at line 79 of file link_elf.c.

Referenced by link_elf_lookup_symbol(), and parse_dynamic().

8.34.2.3 const Elf_Hashelt* elf_file::chains

Definition at line 80 of file link_elf.c.

Referenced by link_elf_lookup_symbol(), and parse_dynamic().

8.34.2.4 long elf_file::ddbstrcnt

Definition at line 98 of file link_elf.c.

Referenced by link_elf_load_file(), link_elf_preload_parse_symbols(), and parse_dynamic().

8.34.2.5 caddr_t elf_file::ddbstrtab

Definition at line 97 of file link_elf.c.

Referenced by elf_obj_lookup(), link_elf_each_function_name(), link_elf_fix_link_set(), link_elf_load_file(), link_elf_lookup_symbol(), link_elf_preload_parse_symbols(), link_elf_symbol_values(), parse_dynamic(), and symbol_name().

8.34.2.6 long elf_file::ddbsymcnt

Definition at line 96 of file link_elf.c.

Referenced by elf_obj_lookup(), link_elf_fix_link_set(), link_elf_load_file(), link_elf_preload_parse_symbols(), link_elf_reloc_local(), link_elf_symbol_values(), parse_dynamic(), and relocate_file().

8.34.2.7 Elf_Sym* elf_file::ddbsymtab

Definition at line 101 of file link_elf_obj.c.

8.34.2.8 const Elf_Sym* elf_file::ddbsymtab

Definition at line 95 of file link_elf.c.

Referenced by elf_obj_lookup(), link_elf_each_function_name(), link_elf_fix_link_set(), link_elf_load_file(), link_elf_lookup_symbol(), link_elf_preload_parse_symbols(), link_elf_reloc_local(), link_elf_search_symbol(), link_elf_symbol_values(), parse_dynamic(), relocate_file(), and symbol_name().

8.34.2.9 Elf_Dyn* elf_file::dynamic

Definition at line 76 of file link_elf.c.

Referenced by link_elf_init(), link_elf_link_common_finish(), link_elf_link_preload(), link_elf_link_preload_finish(), link_elf_load_file(), and parse_dynamic().

8.34.2.10 Elf_Shdr* elf_file::e_shdr

Definition at line 90 of file link_elf_obj.c.

Referenced by link_elf_link_preload(), and link_elf_load_file().

8.34.2.11 Elf_Addr* elf_file::got

Definition at line 85 of file link_elf.c.

Referenced by parse_dynamic().

8.34.2.12 caddr_t elf_file::hash

Definition at line 81 of file link_elf.c.

8.34.2.13 struct linker_file elf_file::lf

Definition at line 85 of file link_elf_obj.c.

8.34.2.14 struct linker_file elf_file::lf

Definition at line 70 of file link_elf.c.

Referenced by link_elf_link_preload(), and relocate_file().

8.34.2.15 caddr_t elf_file::modptr

Definition at line 94 of file link_elf.c.

Referenced by link_elf_link_preload(), and link_elf_preload_parse_symbols().

8.34.2.16 Elf_Hashelt elf_file::nbuckets

Definition at line 77 of file link_elf.c.

Referenced by link_elf_lookup_symbol(), and parse_dynamic().

8.34.2.17 Elf_Hashelt elf_file::nchains

Definition at line 78 of file link_elf.c.

Referenced by elf_get_sym(), elf_get_symname(), elf_lookup(), link_elf_lookup_symbol(), link_elf_symbol_values(), and parse_dynamic().

8.34.2.18 int elf_file::nprogtab

Definition at line 93 of file link_elf_obj.c.

Referenced by findbase(), link_elf_fix_link_set(), link_elf_link_preload(), link_elf_load_file(), and link_elf_lookup_set().

8.34.2.19 int elf_file::nrel

Definition at line 99 of file link_elf_obj.c.

Referenced by link_elf_link_preload(), link_elf_load_file(), link_elf_reloc_local(), link_elf_unload_file(), and relocate_file().

8.34.2.20 int elf_file::nrela

Definition at line 96 of file link_elf_obj.c.

Referenced by link_elf_link_preload(), link_elf_load_file(), link_elf_reloc_local(), and link_elf_unload_file().

8.34.2.21 vm_object_t elf_file::object

Definition at line 89 of file link_elf_obj.c.

Referenced by link_elf_init(), link_elf_link_preload(), link_elf_load_file(), and link_elf_unload_file().

8.34.2.22 const Elf_Rel* elf_file::pltrel

Definition at line 86 of file link_elf.c.

Referenced by parse_dynamic(), and relocate_file().

8.34.2.23 const Elf_Rela* elf_file::pltrela

Definition at line 88 of file link_elf.c.

Referenced by parse_dynamic(), and relocate_file().

8.34.2.24 int elf_file::pltrelasize

Definition at line 89 of file link_elf.c.

Referenced by parse_dynamic(), and relocate_file().

8.34.2.25 int elf_file::pltrelsize

Definition at line 87 of file link_elf.c.

Referenced by parse_dynamic(), and relocate_file().

8.34.2.26 int elf_file::preloaded

Definition at line 71 of file link_elf.c.

Referenced by link_elf_init(), link_elf_link_preload(), and link_elf_unload_file().

8.34.2.27 Elf_progent* elf_file::progtab

Definition at line 92 of file link_elf_obj.c.

Referenced by findbase(), link_elf_fix_link_set(), link_elf_lookup_set(), and link_elf_unload_file().

8.34.2.28 const Elf_Rel* elf_file::rel

Definition at line 90 of file link_elf.c.

Referenced by link_elf_reloc_local(), parse_dynamic(), and relocate_file().

8.34.2.29 const Elf_Rela* elf_file::rela

Definition at line 92 of file link_elf.c.

Referenced by link_elf_reloc_local(), parse_dynamic(), and relocate_file().

8.34.2.30 int elf_file::relasize

Definition at line 93 of file link_elf.c.

Referenced by link_elf_reloc_local(), parse_dynamic(), and relocate_file().

8.34.2.31 Elf_relaent* elf_file::relatab

Definition at line 95 of file link_elf_obj.c.

Referenced by link_elf_reloc_local(), and link_elf_unload_file().

8.34.2.32 int elf_file::relsize

Definition at line 91 of file link_elf.c.

Referenced by link_elf_reloc_local(), parse_dynamic(), and relocate_file().

8.34.2.33 Elf_relent* elf_file::reltab

Definition at line 98 of file link_elf_obj.c.

Referenced by link_elf_reloc_local(), link_elf_unload_file(), and relocate_file().

8.34.2.34 long elf_file::shstrcnt

Definition at line 107 of file link_elf_obj.c.

8.34.2.35 caddr_t elf_file::shstrtab

Definition at line 106 of file link_elf_obj.c.

8.34.2.36 caddr_t elf_file::strbase

Definition at line 100 of file link_elf.c.

Referenced by link_elf_load_file(), and link_elf_unload_file().

8.34.2.37 int elf_file::strsz

Definition at line 83 of file link_elf.c.

Referenced by parse_dynamic().

8.34.2.38 caddr_t elf_file::strtab

Definition at line 82 of file link_elf.c.

Referenced by elf_get_symname(), elf_lookup(), link_elf_link_preload_finish(), link_elf_load_file(), link_elf_lookup_symbol(), link_elf_symbol_values(), parse_dynamic(), and symbol_name().

8.34.2.39 caddr_t elf_file::symbase

Definition at line 99 of file link_elf.c.

Referenced by link_elf_load_file(), and link_elf_unload_file().

8.34.2.40 const Elf_Sym* elf_file::symtab

Definition at line 84 of file link_elf.c.

Referenced by elf_get_sym(), elf_get_symname(), elf_lookup(), link_elf_lookup_symbol(), link_elf_symbol_values(), parse_dynamic(), and symbol_name().

The documentation for this struct was generated from the following files:

- /usr/src/sys/kern/link_elf.c
- /usr/src/sys/kern/link_elf_obj.c

8.35 Elf_progent Struct Reference

Data Fields

- void * addr
- Elf_Off size
- int flags
- int sec
- char * name

8.35.1 Detailed Description

Definition at line 63 of file link_elf_obj.c.

8.35.2 Field Documentation

8.35.2.1 void* Elf_progent::addr

Definition at line 64 of file link_elf_obj.c.

Referenced by findbase(), link_elf_fix_link_set(), and link_elf_lookup_set().

8.35.2.2 int Elf_progent::flags

Definition at line 66 of file link_elf_obj.c.

8.35.2.3 char* Elf_progent::name

Definition at line 68 of file link_elf_obj.c.

Referenced by link_elf_fix_link_set(), and link_elf_lookup_set().

8.35.2.4 int Elf_progent::sec

Definition at line 67 of file link_elf_obj.c.

Referenced by findbase().

8.35.2.5 Elf_Off Elf_progent::size

Definition at line 65 of file link_elf_obj.c.

Referenced by link_elf_fix_link_set(), and link_elf_lookup_set().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/link_elf_obj.c

8.36 Elf_relaent Struct Reference

Data Fields

- Elf_Rela * rela
- int nrela
- int sec

8.36.1 Detailed Description

Definition at line 77 of file link_elf_obj.c.

8.36.2 Field Documentation

8.36.2.1 int Elf_relaent::nrela

Definition at line 79 of file link_elf_obj.c.

Referenced by link_elf_reloc_local().

8.36.2.2 Elf_Rela* Elf_relaent::rela

Definition at line 78 of file link_elf_obj.c.

Referenced by link_elf_reloc_local(), and link_elf_unload_file().

8.36.2.3 int Elf_relaent::sec

Definition at line 80 of file link_elf_obj.c.

Referenced by link_elf_reloc_local().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/link_elf_obj.c

8.37 Elf_relent Struct Reference

Data Fields

- Elf_Rel * rel
- int nrel
- int sec

8.37.1 Detailed Description

Definition at line 71 of file link_elf_obj.c.

8.37.2 Field Documentation

8.37.2.1 int Elf_relent::nrel

Definition at line 73 of file link_elf_obj.c.

Referenced by link_elf_reloc_local(), and relocate_file().

8.37.2.2 Elf_Rel* Elf_relent::rel

Definition at line 72 of file link_elf_obj.c.

Referenced by link_elf_reloc_local(), link_elf_unload_file(), and relocate_file().

8.37.2.3 int Elf_relent::sec

Definition at line 74 of file link_elf_obj.c.

Referenced by link_elf_reloc_local(), and relocate_file().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/link_elf_obj.c

8.38 execve_args Struct Reference

Data Fields

- char * fname
- char ** argv
- char ** envv

8.38.1 Detailed Description

Definition at line 164 of file kern_exec.c.

8.38.2 Field Documentation

8.38.2.1 char** execve_args::argv

Definition at line 166 of file kern_exec.c.

Referenced by execve(), start_init(), and systrace_args().

8.38.2.2 char** execve_args::envv

Definition at line 167 of file kern_exec.c.

Referenced by execve(), start_init(), and systrace_args().

8.38.2.3 char* execve_args::fname

Definition at line 165 of file kern_exec.c.

Referenced by execve(), start_init(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_exec.c

8.39 fchdir_args Struct Reference

Data Fields

- int fd

8.39.1 Detailed Description

Definition at line 669 of file vfs_syscalls.c.

8.39.2 Field Documentation

8.39.2.1 int fchdir_args::fd

Definition at line 670 of file vfs_syscalls.c.

Referenced by fchdir(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.40 fchflags_args Struct Reference

Data Fields

- int fd
- int flags

8.40.1 Detailed Description

Definition at line 2454 of file vfs_syscalls.c.

8.40.2 Field Documentation

8.40.2.1 int fchflags_args::fd

Definition at line 2455 of file vfs_syscalls.c.

Referenced by systrace_args().

8.40.2.2 int fchflags_args::flags

Definition at line 2456 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.41 fchmod_args Struct Reference

Data Fields

- int fd
- int mode

8.41.1 Detailed Description

Definition at line 2594 of file vfs_syscalls.c.

8.41.2 Field Documentation

8.41.2.1 int fchmod_args::fd

Definition at line 2595 of file vfs_syscalls.c.

Referenced by systrace_args().

8.41.2.2 int fchmod_args::mode

Definition at line 2596 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.42 fchown_args Struct Reference

Data Fields

- int fd
- int uid
- int gid

8.42.1 Detailed Description

Definition at line 2749 of file vfs_syscalls.c.

8.42.2 Field Documentation

8.42.2.1 int fchown_args::fd

Definition at line 2750 of file vfs_syscalls.c.

Referenced by systrace_args().

8.42.2.2 int fchown_args::gid

Definition at line 2752 of file vfs_syscalls.c.

Referenced by systrace_args().

8.42.2.3 int fchown_args::uid

Definition at line 2751 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.43 fcntl_args Struct Reference

Data Fields

- int fd
- int cmd
- long arg

8.43.1 Detailed Description

Definition at line 313 of file kern_descrip.c.

8.43.2 Field Documentation

8.43.2.1 long fcntl_args::arg

Definition at line 316 of file kern_descrip.c.

Referenced by fcntl(), and systrace_args().

8.43.2.2 int fcntl_args::cmd

Definition at line 315 of file kern_descrip.c.

Referenced by fcntl(), and systrace_args().

8.43.2.3 int fcntl_args::fd

Definition at line 314 of file kern_descrip.c.

Referenced by fcntl(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_descrip.c

8.44 fhopen_args Struct Reference

Data Fields

- fhandle * u_fhp
- int flags

8.44.1 Detailed Description

Definition at line 4008 of file vfs_syscalls.c.

8.44.2 Field Documentation

8.44.2.1 int fhopen_args::flags

Definition at line 4010 of file vfs_syscalls.c.

Referenced by fhopen(), and systrace_args().

8.44.2.2 struct fhandle* fhopen_args::u_fhp

Definition at line 4009 of file vfs_syscalls.c.

Referenced by fhopen(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.45 fhstat_args Struct Reference

Data Fields

- fhandle * u_fhp
- stat * sb

8.45.1 Detailed Description

Definition at line 4199 of file vfs_syscalls.c.

8.45.2 Field Documentation

8.45.2.1 struct stat* fhstat_args::sb

Definition at line 4201 of file vfs_syscalls.c.

Referenced by systrace_args().

8.45.2.2 struct fhandle* fhstat_args::u_fhp

Definition at line 4200 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.46 fhstatfs_args Struct Reference

Data Fields

- fhandle * u_fhp
- statfs * buf

8.46.1 Detailed Description

Definition at line 4249 of file vfs_syscalls.c.

8.46.2 Field Documentation

8.46.2.1 struct statfs* fhstatfs_args::buf

Definition at line 4251 of file vfs_syscalls.c.

Referenced by fhstatfs(), and systrace_args().

8.46.2.2 struct fhandle* fhstatfs_args::u_fhp

Definition at line 4250 of file vfs_syscalls.c.

Referenced by fhstatfs(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.47 filedesc0 Struct Reference

Data Fields

- filedesc fd_fd
- file * fd_dfiles [NDFILE]
- char fd_dfileflags [NDFILE]
- NDSLOTTYPE fd_dmap [NDSLOTS(NDFILE)]

8.47.1 Detailed Description

Definition at line 123 of file kern_descrip.c.

8.47.2 Field Documentation

8.47.2.1 char filedesc0::fd_dfileflags[NDFILE]

Definition at line 130 of file kern_descrip.c.

8.47.2.2 struct file* filedesc0::fd_dfiles[NDFILE]

Definition at line 129 of file kern_descrip.c.

8.47.2.3 NDSLOTTYPE filedesc0::fd_dmap[NDSLOTS(NDFILE)]

Definition at line 131 of file kern_descrip.c.

8.47.2.4 struct filedesc filedesc0::fd_fd

Definition at line 124 of file kern_descrip.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_descrip.c

8.48 flock_args Struct Reference

Data Fields

- int fd
- int how

8.48.1 Detailed Description

Definition at line 2221 of file kern_descrip.c.

8.48.2 Field Documentation

8.48.2.1 int flock_args::fd

Definition at line 2222 of file kern_descrip.c.

Referenced by flock(), and systrace_args().

8.48.2.2 int flock_args::how

Definition at line 2223 of file kern_descrip.c.

Referenced by flock(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_descrip.c

8.49 fork_args Struct Reference

Data Fields

- int dummy

8.49.1 Detailed Description

Definition at line 80 of file kern_fork.c.

8.49.2 Field Documentation

8.49.2.1 int fork_args::dummy

Definition at line 81 of file kern_fork.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_fork.c

8.50 fpathconf_args Struct Reference

Data Fields

- int fd
- int name

8.50.1 Detailed Description

Definition at line 1146 of file kern_descrip.c.

8.50.2 Field Documentation

8.50.2.1 int fpathconf_args::fd

Definition at line 1147 of file kern_descrip.c.

Referenced by fpathconf(), and systrace_args().

8.50.2.2 int fpathconf_args::name

Definition at line 1148 of file kern_descrip.c.

Referenced by fpathconf(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_descrip.c

8.51 fstat_args Struct Reference

Data Fields

- int fd
- stat * sb

8.51.1 Detailed Description

Definition at line 1075 of file kern_descrip.c.

8.51.2 Field Documentation

8.51.2.1 int fstat_args::fd

Definition at line 1076 of file kern_descrip.c.

Referenced by fstat(), and systrace_args().

8.51.2.2 struct stat* fstat_args::sb

Definition at line 1077 of file kern_descrip.c.

Referenced by fstat(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_descrip.c

8.52 fstatfs_args Struct Reference

Data Fields

- int fd
- statfs * buf

8.52.1 Detailed Description

Definition at line 289 of file vfs_syscalls.c.

8.52.2 Field Documentation

8.52.2.1 struct statfs* fstatfs_args::buf

Definition at line 291 of file vfs_syscalls.c.

Referenced by systrace_args().

8.52.2.2 int fstatfs_args::fd

Definition at line 290 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.53 fsync_args Struct Reference

Data Fields

- int fd

8.53.1 Detailed Description

Definition at line 3187 of file vfs_syscalls.c.

8.53.2 Field Documentation

8.53.2.1 int fsync_args::fd

Definition at line 3188 of file vfs_syscalls.c.

Referenced by fsync(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.54 ftruncate_args Struct Reference

Data Fields

- int fd
- int pad
- off_t length

8.54.1 Detailed Description

Definition at line 3067 of file vfs_syscalls.c.

8.54.2 Field Documentation

8.54.2.1 int ftruncate_args::fd

Definition at line 3068 of file vfs_syscalls.c.

Referenced by systrace_args().

8.54.2.2 off_t ftruncate_args::length

Definition at line 3070 of file vfs_syscalls.c.

Referenced by systrace_args().

8.54.2.3 int ftruncate_args::pad

Definition at line 3069 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.55 futimes_args Struct Reference

Data Fields

- int fd
- timeval * tptr

8.55.1 Detailed Description

Definition at line 2954 of file vfs_syscalls.c.

8.55.2 Field Documentation

8.55.2.1 int futimes_args::fd

Definition at line 2955 of file vfs_syscalls.c.

Referenced by systrace_args().

8.55.2.2 struct timeval* futimes_args::tptr

Definition at line 2956 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.56 getcontext_args Struct Reference

Data Fields

- __ucontext * ucp

8.56.1 Detailed Description

Definition at line 49 of file kern_context.c.

8.56.2 Field Documentation

8.56.2.1 struct __ucontext* getcontext_args::ucp

Definition at line 50 of file kern_context.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_context.c

8.57 getdents_args Struct Reference

Data Fields

- int fd
- char * buf
- size_t count

8.57.1 Detailed Description

Definition at line 3778 of file vfs_syscalls.c.

8.57.2 Field Documentation

8.57.2.1 char* getdents_args::buf

Definition at line 3780 of file vfs_syscalls.c.

Referenced by systrace_args().

8.57.2.2 size_t getdents_args::count

Definition at line 3781 of file vfs_syscalls.c.

Referenced by systrace_args().

8.57.2.3 int getdents_args::fd

Definition at line 3779 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.58 getdirentries_args Struct Reference

Data Fields

- int fd
- char * buf
- u_int count
- long * basep

8.58.1 Detailed Description

Definition at line 3690 of file vfs_syscalls.c.

8.58.2 Field Documentation

8.58.2.1 long* getdirentries_args::basep

Definition at line 3694 of file vfs_syscalls.c.

Referenced by getdents(), and systrace_args().

8.58.2.2 char* getdirentries_args::buf

Definition at line 3692 of file vfs_syscalls.c.

Referenced by getdents(), and systrace_args().

8.58.2.3 u_int getdirentries_args::count

Definition at line 3693 of file vfs_syscalls.c.

Referenced by getdents(), and systrace_args().

8.58.2.4 int getdirentries_args::fd

Definition at line 3691 of file vfs_syscalls.c.

Referenced by getdents(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.59 getdomainname_args Struct Reference

Data Fields

- char * domainname
- int len

8.59.1 Detailed Description

Definition at line 255 of file kern_xxx.c.

8.59.2 Field Documentation

8.59.2.1 char* getdomainname_args::domainname

Definition at line 256 of file kern_xxx.c.

Referenced by getdomainname(), and systrace_args().

8.59.2.2 int getdomainname_args::len

Definition at line 257 of file kern_xxx.c.

Referenced by getdomainname(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_xxx.c

8.60 getdtablesize_args Struct Reference

Data Fields

- int dummy

8.60.1 Detailed Description

Definition at line 246 of file kern_descrip.c.

8.60.2 Field Documentation

8.60.2.1 int getdtablesize_args::dummy

Definition at line 247 of file kern_descrip.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_descrip.c

8.61 getegid_args Struct Reference

Data Fields

- int dummy

8.61.1 Detailed Description

Definition at line 274 of file kern_prot.c.

8.61.2 Field Documentation

8.61.2.1 int getegid_args::dummy

Definition at line 275 of file kern_prot.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.62 geteuid_args Struct Reference

Data Fields

- int dummy

8.62.1 Detailed Description

Definition at line 232 of file kern_prot.c.

8.62.2 Field Documentation

8.62.2.1 int geteuid_args::dummy

Definition at line 233 of file kern_prot.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.63 getfh_args Struct Reference

Data Fields

- char * fname
- fhandle_t * fhp

8.63.1 Detailed Description

Definition at line 3960 of file vfs_syscalls.c.

8.63.2 Field Documentation

8.63.2.1 fhandle_t* getfh_args::fhp

Definition at line 3962 of file vfs_syscalls.c.

Referenced by systrace_args().

8.63.2.2 char* getfh_args::fname

Definition at line 3961 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.64 getfsstat_args Struct Reference

Data Fields

- statfs * buf
- long bufsize
- int flags

8.64.1 Detailed Description

Definition at line 373 of file vfs_syscalls.c.

8.64.2 Field Documentation

8.64.2.1 struct statfs* getfsstat_args::buf

Definition at line 374 of file vfs_syscalls.c.

Referenced by systrace_args().

8.64.2.2 long getfsstat_args::bufsize

Definition at line 375 of file vfs_syscalls.c.

Referenced by systrace_args().

8.64.2.3 int getfsstat_args::flags

Definition at line 376 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.65 getgid_args Struct Reference

Data Fields

- int dummy

8.65.1 Detailed Description

Definition at line 249 of file kern_prot.c.

8.65.2 Field Documentation

8.65.2.1 int getgid_args::dummy

Definition at line 250 of file kern_prot.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.66 getgroups_args Struct Reference

Data Fields

- u_int gidsetsize
- gid_t * gidset

8.66.1 Detailed Description

Definition at line 291 of file kern_prot.c.

8.66.2 Field Documentation

8.66.2.1 gid_t* getgroups_args::gidset

Definition at line 293 of file kern_prot.c.

Referenced by systrace_args().

8.66.2.2 u_int getgroups_args::gidsetsize

Definition at line 292 of file kern_prot.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.67 getitimer_args Struct Reference

Data Fields

- u_int which
- itimerval * itv

8.67.1 Detailed Description

Definition at line 544 of file kern_time.c.

8.67.2 Field Documentation

8.67.2.1 struct itimerval* getitimer_args::itv

Definition at line 546 of file kern_time.c.

Referenced by getitimer(), and systrace_args().

8.67.2.2 u_int getitimer_args::which

Definition at line 545 of file kern_time.c.

Referenced by getitimer(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_time.c

8.68 getlogin_args Struct Reference

Data Fields

- char * namebuf
- u_int namelen

8.68.1 Detailed Description

Definition at line 1984 of file kern_prot.c.

8.68.2 Field Documentation

8.68.2.1 char* getlogin_args::namebuf

Definition at line 1985 of file kern_prot.c.

Referenced by getlogin(), and systrace_args().

8.68.2.2 u_int getlogin_args::namelen

Definition at line 1986 of file kern_prot.c.

Referenced by getlogin(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.69 getpgid_args Struct Reference

Data Fields

- pid_t pid

8.69.1 Detailed Description

Definition at line 145 of file kern_prot.c.

8.69.2 Field Documentation

8.69.2.1 pid_t getpgid_args::pid

Definition at line 146 of file kern_prot.c.

Referenced by getpgid(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.70 getpgrp_args Struct Reference

Data Fields

- int dummy

8.70.1 Detailed Description

Definition at line 125 of file kern_prot.c.

8.70.2 Field Documentation

8.70.2.1 int getpgrp_args::dummy

Definition at line 126 of file kern_prot.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.71 getpid_args Struct Reference

Data Fields

- int dummy

8.71.1 Detailed Description

Definition at line 79 of file kern_prot.c.

8.71.2 Field Documentation

8.71.2.1 int getpid_args::dummy

Definition at line 80 of file kern_prot.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.72 getppid_args Struct Reference

Data Fields

- int dummy

8.72.1 Detailed Description

Definition at line 102 of file kern_prot.c.

8.72.2 Field Documentation

8.72.2.1 int getppid_args::dummy

Definition at line 103 of file kern_prot.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.73 getresgid_args Struct Reference

Data Fields

- gid_t * rgid
- gid_t * egid
- gid_t * sgid

8.73.1 Detailed Description

Definition at line 1235 of file kern_prot.c.

8.73.2 Field Documentation

8.73.2.1 gid_t* getresgid_args::egid

Definition at line 1237 of file kern_prot.c.

Referenced by getresgid(), and systrace_args().

8.73.2.2 gid_t* getresgid_args::rgid

Definition at line 1236 of file kern_prot.c.

Referenced by getresgid(), and systrace_args().

8.73.2.3 gid_t* getresgid_args::sgid

Definition at line 1238 of file kern_prot.c.

Referenced by getresgid(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.74 getresuid_args Struct Reference

Data Fields

- uid_t * ruid
- uid_t * euid
- uid_t * suid

8.74.1 Detailed Description

Definition at line 1205 of file kern_prot.c.

8.74.2 Field Documentation

8.74.2.1 uid_t* getresuid_args::euid

Definition at line 1207 of file kern_prot.c.

Referenced by getresuid(), and systrace_args().

8.74.2.2 uid_t* getresuid_args::ruid

Definition at line 1206 of file kern_prot.c.

Referenced by getresuid(), and systrace_args().

8.74.2.3 uid_t* getresuid_args::suid

Definition at line 1208 of file kern_prot.c.

Referenced by getresuid(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.75 getrusage_args Struct Reference

Data Fields

- int who
- rusage * rusage

8.75.1 Detailed Description

Definition at line 929 of file kern_resource.c.

8.75.2 Field Documentation

8.75.2.1 struct rusage* getrusage_args::rusage

Definition at line 931 of file kern_resource.c.

Referenced by systrace_args().

8.75.2.2 int getrusage_args::who

Definition at line 930 of file kern_resource.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_resource.c

8.76 getsid_args Struct Reference

Data Fields

- pid_t pid

8.76.1 Detailed Description

Definition at line 180 of file kern_prot.c.

8.76.2 Field Documentation

8.76.2.1 pid_t getsid_args::pid

Definition at line 181 of file kern_prot.c.

Referenced by getsid(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.77 gettimeofday_args Struct Reference

Data Fields

- timeval * tp
- timezone * tzp

8.77.1 Detailed Description

Definition at line 434 of file kern_time.c.

8.77.2 Field Documentation

8.77.2.1 struct timeval* gettimeofday_args::tp

Definition at line 435 of file kern_time.c.

Referenced by gettimeofday(), and systrace_args().

8.77.2.2 struct timezone* gettimeofday_args::tzp

Definition at line 436 of file kern_time.c.

Referenced by gettimeofday(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_time.c

8.78 getuid_args Struct Reference

Data Fields

- int dummy

8.78.1 Detailed Description

Definition at line 212 of file kern_prot.c.

8.78.2 Field Documentation

8.78.2.1 int getuid_args::dummy

Definition at line 213 of file kern_prot.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.79 huft Struct Reference

Collaboration diagram for huft:

Data Fields

- uch e
- uch b
- union {
  ush n
  huft * t
  } v

8.79.1 Detailed Description

Definition at line 264 of file inflate.c.

8.79.2 Field Documentation

8.79.2.1 uch huft::b

Definition at line 266 of file inflate.c.

Referenced by huft_build(), inflate_block(), inflate_codes(), inflate_dynamic(), and inflate_stored().

8.79.2.2 uch huft::e

Definition at line 265 of file inflate.c.

Referenced by huft_build(), inflate_codes(), and xinflate().

8.79.2.3 ush huft::n

Definition at line 268 of file inflate.c.

Referenced by huft_build(), inflate_codes(), inflate_dynamic(), and inflate_stored().

8.79.2.4 struct huft* huft::t

Definition at line 270 of file inflate.c.

Referenced by huft_build(), huft_free(), inflate_block(), and inflate_codes().

8.79.2.5 union { ... } huft::v

Referenced by huft_build(), huft_free(), and inflate_codes().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/inflate.c

8.80 imgact_gzip Struct Reference

Data Fields

- image_params * ip
- exec a_out
- int error
- int gotheader
- int where
- u_char * inbuf
- u_long offset
- u_long output
- u_long len
- int idx
- u_long virtual_offset
- u_long file_offset
- u_long file_end
- u_long bss_size

8.80.1 Detailed Description

Definition at line 49 of file imgact_gzip.c.

8.80.2 Field Documentation

8.80.2.1 struct exec imgact_gzip::a_out

Definition at line 51 of file imgact_gzip.c.

Referenced by do_aout_hdr(), and Flush().

8.80.2.2 u_long imgact_gzip::bss_size

Definition at line 60 of file imgact_gzip.c.

Referenced by do_aout_hdr().

8.80.2.3 int imgact_gzip::error

Definition at line 52 of file imgact_gzip.c.

Referenced by Flush(), and NextByte().

8.80.2.4 u_long imgact_gzip::file_end

Definition at line 60 of file imgact_gzip.c.

Referenced by do_aout_hdr(), and Flush().

8.80.2.5 u_long imgact_gzip::file_offset

Definition at line 60 of file imgact_gzip.c.

Referenced by do_aout_hdr(), and Flush().

8.80.2.6 int imgact_gzip::gotheader

Definition at line 53 of file imgact_gzip.c.

Referenced by Flush().

8.80.2.7 int imgact_gzip::idx

Definition at line 59 of file imgact_gzip.c.

Referenced by NextByte().

8.80.2.8 u_char* imgact_gzip::inbuf

Definition at line 55 of file imgact_gzip.c.

Referenced by NextByte().

8.80.2.9 struct image_params* imgact_gzip::ip

Definition at line 50 of file imgact_gzip.c.

Referenced by do_aout_hdr(), and NextByte().

8.80.2.10 u_long imgact_gzip::len

Definition at line 58 of file imgact_gzip.c.

Referenced by NextByte().

8.80.2.11 u_long imgact_gzip::offset

Definition at line 56 of file imgact_gzip.c.

Referenced by NextByte().

8.80.2.12 u_long imgact_gzip::output

Definition at line 57 of file imgact_gzip.c.

Referenced by Flush().

8.80.2.13 u_long imgact_gzip::virtual_offset

Definition at line 60 of file imgact_gzip.c.

Referenced by do_aout_hdr(), and Flush().

8.80.2.14 int imgact_gzip::where

Definition at line 54 of file imgact_gzip.c.

Referenced by do_aout_hdr(), Flush(), and NextByte().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/imgact_gzip.c

8.81 intr_entropy Struct Reference

Data Fields

- thread ∗ td
- uintptr_t event

8.81.1 Detailed Description

Definition at line 74 of file kern_intr.c.

8.81.2 Field Documentation

8.81.2.1 uintptr_t intr_entropy::event

Definition at line 76 of file kern_intr.c.

Referenced by intr_event_schedule_thread().

8.81.2.2 struct thread∗ intr_entropy::td

Definition at line 75 of file kern_intr.c.

Referenced by intr_event_schedule_thread().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_intr.c

8.82 intr_thread Struct Reference

Data Fields

- intr_event * it_event
- thread * it_thread
- int it_flags
- int it_need

8.82.1 Detailed Description

Definition at line 64 of file kern_intr.c.

8.82.2 Field Documentation

8.82.2.1 struct intr_event* intr_thread::it_event

Definition at line 65 of file kern_intr.c.

Referenced by intr_event_add_handler(), ithread_destroy(), ithread_loop(), and ithread_update().

8.82.2.2 int intr_thread::it_flags

Definition at line 67 of file kern_intr.c.

Referenced by ithread_destroy(), and ithread_loop().

8.82.2.3 int intr_thread::it_need

Definition at line 68 of file kern_intr.c.

Referenced by intr_event_schedule_thread(), and ithread_loop().

8.82.2.4 struct thread* intr_thread::it_thread

Definition at line 66 of file kern_intr.c.

Referenced by intr_event_schedule_thread(), ithread_create(), ithread_destroy(), ithread_loop(), and ithread_update().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_intr.c

8.83 ioctl_args Struct Reference

Data Fields

- int fd
- u_long com
- caddr_t data

8.83.1 Detailed Description

Definition at line 513 of file sys_generic.c.

8.83.2 Field Documentation

8.83.2.1 u_long ioctl_args::com

Definition at line 515 of file sys_generic.c.

Referenced by ioctl(), and systrace_args().

8.83.2.2 caddr_t ioctl_args::data

Definition at line 516 of file sys_generic.c.

Referenced by ioctl(), and systrace_args().

8.83.2.3 int ioctl_args::fd

Definition at line 514 of file sys_generic.c.

Referenced by ioctl(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sys_generic.c

8.84 issetugid_args Struct Reference

Data Fields

- int dummy

8.84.1 Detailed Description

Definition at line 1265 of file kern_prot.c.

8.84.2 Field Documentation

8.84.2.1 int issetugid_args::dummy

Definition at line 1266 of file kern_prot.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.85 kaioinfo Struct Reference

Data Fields

- mtx kaio_mtx
- int kaio_flags
- int kaio_maxactive_count
- int kaio_active_count
- int kaio_qallowed_count
- int kaio_count
- int kaio_ballowed_count
- int kaio_buffer_count

8.85.1 Detailed Description

Definition at line 280 of file vfs_aio.c.

8.85.2 Field Documentation

8.85.2.1 int kaioinfo::kaio_active_count

Definition at line 284 of file vfs_aio.c.

Referenced by aio_daemon(), aio_init_aioinfo(), aio_kick(), aio_kick_nowait(), and aio_selectjob().

8.85.2.2 int kaioinfo::kaio_ballowed_count

Definition at line 287 of file vfs_aio.c.

Referenced by aio_init_aioinfo(), and aio_qphysio().

8.85.2.3 int kaioinfo::kaio_buffer_count

Definition at line 288 of file vfs_aio.c.

Referenced by aio_init_aioinfo(), aio_qphysio(), and biohelper().

8.85.2.4 int kaioinfo::kaio_count

Definition at line 286 of file vfs_aio.c.

Referenced by aio_aqueue(), aio_free_entry(), aio_init_aioinfo(), and aio_qphysio().

8.85.2.5 int kaioinfo::kaio_flags

Definition at line 282 of file vfs_aio.c.

Referenced by aio_bio_done_notify(), aio_init_aioinfo(), aio_proc_rundown(), aio_waitcomplete(), and do_lio_listio().

8.85.2.6 int kaioinfo::kaio_maxactive_count

Definition at line 283 of file vfs_aio.c.

Referenced by aio_init_aioinfo(), aio_kick(), aio_kick_nowait(), and aio_selectjob().

8.85.2.7 struct mtx kaioinfo::kaio_mtx

Definition at line 281 of file vfs_aio.c.

Referenced by aio_init_aioinfo().

8.85.2.8 int kaioinfo::kaio_qallowed_count

Definition at line 285 of file vfs_aio.c.

Referenced by aio_aqueue(), and aio_init_aioinfo().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_aio.c

8.86 kevent_args Struct Reference

Data Fields

- int fd
- kevent * changelist
- int nchanges
- kevent * eventlist
- int nevents
- timespec * timeout

8.86.1 Detailed Description

Definition at line 551 of file kern_event.c.

8.86.2 Field Documentation

8.86.2.1 struct kevent* kevent_args::changelist

Definition at line 553 of file kern_event.c.

Referenced by kevent(), kevent_copyin(), and systrace_args().

8.86.2.2 struct kevent* kevent_args::eventlist

Definition at line 555 of file kern_event.c.

Referenced by kevent(), kevent_copyout(), and systrace_args().

8.86.2.3 int kevent_args::fd

Definition at line 552 of file kern_event.c.

Referenced by kevent(), and systrace_args().

8.86.2.4 int kevent_args::nchanges

Definition at line 554 of file kern_event.c.

Referenced by kevent(), and systrace_args().

8.86.2.5 int kevent_args::nevents

Definition at line 556 of file kern_event.c.

Referenced by kevent(), and systrace_args().

8.86.2.6 struct timespec* kevent_args::timeout

Definition at line 557 of file kern_event.c.

Referenced by kevent(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_event.c

8.87 kill_args Struct Reference

Data Fields

- int pid
- int signum

8.87.1 Detailed Description

Definition at line 1731 of file kern_sig.c.

8.87.2 Field Documentation

8.87.2.1 int kill_args::pid

Definition at line 1732 of file kern_sig.c.

Referenced by systrace_args().

8.87.2.2 int kill_args::signum

Definition at line 1733 of file kern_sig.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_sig.c

8.88 krqbits Struct Reference

Data Fields

- kqb_word_t rqb_bits [KQB_LEN]

8.88.1 Detailed Description

Definition at line 165 of file sched_core.c.

8.88.2 Field Documentation

8.88.2.1 kqb_word_t krqbits::rqb_bits[KQB_LEN]

Definition at line 166 of file sched_core.c.

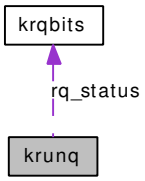
Referenced by krunq_check(), krunq_clrbit(), krunq_findbit(), and krunq_setbit().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sched_core.c

8.89 krunq Struct Reference

Collaboration diagram for krunq:



Data Fields

- krqbits rq_status
- krqhead rq_queues [KQ_NQS]

8.89.1 Detailed Description

Definition at line 173 of file sched_core.c.

8.89.2 Field Documentation

8.89.2.1 struct krqhead krunq::rq_queues[KQ_NQS]

Definition at line 175 of file sched_core.c.

Referenced by krunq_add(), krunq_choose(), and krunq_remove().

8.89.2.2 struct krqbits krunq::rq_status

Definition at line 174 of file sched_core.c.

Referenced by krunq_check(), krunq_clrbit(), krunq_findbit(), and krunq_setbit().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sched_core.c

8.90 ksched Struct Reference

Data Fields

- timespec rr_interval

8.90.1 Detailed Description

Definition at line 53 of file ksched.c.

8.90.2 Field Documentation

8.90.2.1 struct timespec ksched::rr_interval

Definition at line 54 of file ksched.c.

Referenced by ksched_rr_get_interval().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/ksched.c

8.91 kse_switchin_args Struct Reference

Data Fields

- kse_thr_mailbox * tmbx
- int flags

8.91.1 Detailed Description

Definition at line 121 of file kern_kse.c.

8.91.2 Field Documentation

8.91.2.1 int kse_switchin_args::flags

Definition at line 123 of file kern_kse.c.

Referenced by kse_switchin(), and systrace_args().

8.91.2.2 struct kse_thr_mailbox* kse_switchin_args::tmbx

Definition at line 122 of file kern_kse.c.

Referenced by kse_switchin(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_kse.c

8.92 ksem_close_args Struct Reference

Data Fields

- semid_t id

8.92.1 Detailed Description

Definition at line 564 of file uipc_sem.c.

8.92.2 Field Documentation

8.92.2.1 semid_t ksem_close_args::id

Definition at line 565 of file uipc_sem.c.

Referenced by ksem_close(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/uipc_sem.c

8.93 ksem_destroy_args Struct Reference

Data Fields

- semid_t id

8.93.1 Detailed Description

Definition at line 791 of file uipc_sem.c.

8.93.2 Field Documentation

8.93.2.1 semid_t ksem_destroy_args::id

Definition at line 792 of file uipc_sem.c.

Referenced by ksem_destroy(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/uipc_sem.c

8.94 ksem_getvalue_args Struct Reference

Data Fields

- semid_t id
- int * val

8.94.1 Detailed Description

Definition at line 759 of file uipc_sem.c.

8.94.2 Field Documentation

8.94.2.1 semid_t ksem_getvalue_args::id

Definition at line 760 of file uipc_sem.c.

Referenced by ksem_getvalue(), and systrace_args().

8.94.2.2 int* ksem_getvalue_args::val

Definition at line 761 of file uipc_sem.c.

Referenced by ksem_getvalue(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/uipc_sem.c

8.95 ksem_init_args Struct Reference

Data Fields

- unsigned int value
- semid_t * idp

8.95.1 Detailed Description

Definition at line 238 of file uipc_sem.c.

8.95.2 Field Documentation

8.95.2.1 semid_t* ksem_init_args::idp

Definition at line 240 of file uipc_sem.c.

Referenced by ksem_init(), and systrace_args().

8.95.2.2 unsigned int ksem_init_args::value

Definition at line 239 of file uipc_sem.c.

Referenced by ksem_init(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/uipc_sem.c

8.96 ksem_open_args Struct Reference

Data Fields

- char * name
- int oflag
- mode_t mode
- unsigned int value
- semid_t * idp

8.96.1 Detailed Description

Definition at line 283 of file uipc_sem.c.

8.96.2 Field Documentation

8.96.2.1 semid_t* ksem_open_args::idp

Definition at line 288 of file uipc_sem.c.

Referenced by ksem_open(), and systrace_args().

8.96.2.2 mode_t ksem_open_args::mode

Definition at line 286 of file uipc_sem.c.

Referenced by ksem_open(), and systrace_args().

8.96.2.3 char* ksem_open_args::name

Definition at line 284 of file uipc_sem.c.

Referenced by ksem_open(), and systrace_args().

8.96.2.4 int ksem_open_args::oflag

Definition at line 285 of file uipc_sem.c.

Referenced by ksem_open(), and systrace_args().

8.96.2.5 unsigned int ksem_open_args::value

Definition at line 287 of file uipc_sem.c.

Referenced by ksem_open(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/uipc_sem.c

8.97 ksem_post_args Struct Reference

Data Fields

- semid_t id

8.97.1 Detailed Description

Definition at line 594 of file uipc_sem.c.

8.97.2 Field Documentation

8.97.2.1 semid_t ksem_post_args::id

Definition at line 595 of file uipc_sem.c.

Referenced by ksem_post(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/uipc_sem.c

8.98 ksem_timedwait_args Struct Reference

Data Fields

- semid_t id
- timespec * abstime

8.98.1 Detailed Description

Definition at line 651 of file uipc_sem.c.

8.98.2 Field Documentation

8.98.2.1 struct timespec* ksem_timedwait_args::abstime

Definition at line 653 of file uipc_sem.c.

Referenced by ksem_timedwait(), and systrace_args().

8.98.2.2 semid_t ksem_timedwait_args::id

Definition at line 652 of file uipc_sem.c.

Referenced by ksem_timedwait(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/uipc_sem.c

8.99 ksem_trywait_args Struct Reference

Data Fields

- semid_t id

8.99.1 Detailed Description

Definition at line 679 of file uipc_sem.c.

8.99.2 Field Documentation

8.99.2.1 semid_t ksem_trywait_args::id

Definition at line 680 of file uipc_sem.c.

Referenced by ksem_trywait(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/uipc_sem.c

8.100 ksem_unlink_args Struct Reference

Data Fields

- char * name

8.100.1 Detailed Description

Definition at line 516 of file uipc_sem.c.

8.100.2 Field Documentation

8.100.2.1 char* ksem_unlink_args::name

Definition at line 517 of file uipc_sem.c.

Referenced by ksem_unlink(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/uipc_sem.c

8.101 ksem_wait_args Struct Reference

Data Fields

- semid_t id

8.101.1 Detailed Description

Definition at line 637 of file uipc_sem.c.

8.101.2 Field Documentation

8.101.2.1 semid_t ksem_wait_args::id

Definition at line 638 of file uipc_sem.c.

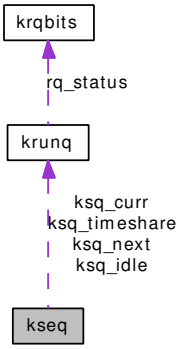
Referenced by ksem_wait(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/uipc_sem.c

8.102 kseq Struct Reference

Collaboration diagram for kseq:



Data Fields

- krunq * ksq_curr
- krunq * ksq_next
- krunq ksq_timeshare [2]
- krunq ksq_idle
- int ksq_load
- uint64_t ksq_last_timestamp
- unsigned ksq_expired_tick
- signed char ksq_expired_nice

8.102.1 Detailed Description

Definition at line 238 of file sched_core.c.

8.102.2 Field Documentation

8.102.2.1 struct krunq* kseq::ksq_curr

Definition at line 239 of file sched_core.c.

Referenced by kseq_choose(), kseq_runnable(), kseq_setup(), and sched_add().

8.102.2.2 signed char kseq::ksq_expired_nice

Definition at line 246 of file sched_core.c.

Referenced by kseq_choose(), kseq_setup(), sched_starving(), and sched_tick().

8.102.2.3 unsigned kseq::ksq_expired_tick

Definition at line 245 of file sched_core.c.

Referenced by kseq_choose(), kseq_setup(), sched_starving(), and sched_tick().

8.102.2.4 struct krunq kseq::ksq_idle

Definition at line 242 of file sched_core.c.

Referenced by kseq_choose(), kseq_runnable(), kseq_setup(), and sched_add().

8.102.2.5 uint64_t kseq::ksq_last_timestamp

Definition at line 244 of file sched_core.c.

Referenced by sched_tick(), and sched_wakeup().

8.102.2.6 int kseq::ksq_load

Definition at line 243 of file sched_core.c.

Referenced by kseq_load_add(), kseq_load_rem(), and sched_starving().

8.102.2.7 struct krunq* kseq::ksq_next

Definition at line 240 of file sched_core.c.

Referenced by kseq_choose(), kseq_runnable(), kseq_setup(), and sched_add().

8.102.2.8 struct krunq kseq::ksq_timeshare[2]

Definition at line 241 of file sched_core.c.

Referenced by kseq_setup().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sched_core.c

8.103 ktimer_create_args Struct Reference

Data Fields

- clockid_t clock_id
- sigevent * evp
- int * timerid

8.103.1 Detailed Description

Definition at line 950 of file kern_time.c.

8.103.2 Field Documentation

8.103.2.1 clockid_t ktimer_create_args::clock_id

Definition at line 951 of file kern_time.c.

Referenced by ktimer_create(), and systrace_args().

8.103.2.2 struct sigevent* ktimer_create_args::evp

Definition at line 952 of file kern_time.c.

Referenced by ktimer_create(), and systrace_args().

8.103.2.3 int* ktimer_create_args::timerid

Definition at line 953 of file kern_time.c.

Referenced by ktimer_create(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_time.c

8.104 ktimer_delete_args Struct Reference

Data Fields

- int timerid

8.104.1 Detailed Description

Definition at line 1092 of file kern_time.c.

8.104.2 Field Documentation

8.104.2.1 int ktimer_delete_args::timerid

Definition at line 1093 of file kern_time.c.

Referenced by ktimer_delete(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_time.c

8.105 ktimer_gettime_args Struct Reference

Data Fields

- int timerid
- itimerspec * value

8.105.1 Detailed Description

Definition at line 1198 of file kern_time.c.

8.105.2 Field Documentation

8.105.2.1 int ktimer_gettime_args::timerid

Definition at line 1199 of file kern_time.c.

Referenced by ktimer_gettime(), and systrace_args().

8.105.2.2 struct itimerspec* ktimer_gettime_args::value

Definition at line 1200 of file kern_time.c.

Referenced by ktimer_gettime(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_time.c

8.106 ktimer_settime_args Struct Reference

Data Fields

- int timerid
- int flags
- itimerspec * value
- itimerspec * ovalue

8.106.1 Detailed Description

Definition at line 1154 of file kern_time.c.

8.106.2 Field Documentation

8.106.2.1 int ktimer_settime_args::flags

Definition at line 1156 of file kern_time.c.

Referenced by ktimer_settime(), and systrace_args().

8.106.2.2 struct itimerspec* ktimer_settime_args::ovalue

Definition at line 1158 of file kern_time.c.

Referenced by ktimer_settime(), and systrace_args().

8.106.2.3 int ktimer_settime_args::timerid

Definition at line 1155 of file kern_time.c.

Referenced by ktimer_settime(), and systrace_args().

8.106.2.4 struct itimerspec* ktimer_settime_args::value

Definition at line 1157 of file kern_time.c.

Referenced by ktimer_settime(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_time.c

8.107 ktrace_args Struct Reference

Data Fields

- char * fname
- int ops
- int facs
- int pid

8.107.1 Detailed Description

Definition at line 571 of file kern_ktrace.c.

8.107.2 Field Documentation

8.107.2.1 int ktrace_args::facs

Definition at line 574 of file kern_ktrace.c.

Referenced by systrace_args().

8.107.2.2 char* ktrace_args::fname

Definition at line 572 of file kern_ktrace.c.

Referenced by systrace_args().

8.107.2.3 int ktrace_args::ops

Definition at line 573 of file kern_ktrace.c.

Referenced by systrace_args().

8.107.2.4 int ktrace_args::pid

Definition at line 575 of file kern_ktrace.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_ktrace.c

8.108 lchmod_args Struct Reference

Data Fields

- char * path
- int mode

8.108.1 Detailed Description

Definition at line 2560 of file vfs_syscalls.c.

8.108.2 Field Documentation

8.108.2.1 int lchmod_args::mode

Definition at line 2562 of file vfs_syscalls.c.

Referenced by systrace_args().

8.108.2.2 char* lchmod_args::path

Definition at line 2561 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.109 lchown_args Struct Reference

Data Fields

- char * path
- int uid
- int gid

8.109.1 Detailed Description

Definition at line 2706 of file vfs_syscalls.c.

8.109.2 Field Documentation

8.109.2.1 int lchown_args::gid

Definition at line 2709 of file vfs_syscalls.c.

Referenced by systrace_args().

8.109.2.2 char* lchown_args::path

Definition at line 2707 of file vfs_syscalls.c.

Referenced by systrace_args().

8.109.2.3 int lchown_args::uid

Definition at line 2708 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.110 lgetfh_args Struct Reference

Data Fields

- char * fname
- fhandle_t * fhp

8.110.1 Detailed Description

Definition at line 3921 of file vfs_syscalls.c.

8.110.2 Field Documentation

8.110.2.1 fhandle_t* lgetfh_args::fhp

Definition at line 3923 of file vfs_syscalls.c.

Referenced by systrace_args().

8.110.2.2 char* lgetfh_args::fname

Definition at line 3922 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.111 link_args Struct Reference

Data Fields

- char * path
- char * link

8.111.1 Detailed Description

Definition at line 1358 of file vfs_syscalls.c.

8.111.2 Field Documentation

8.111.2.1 char* link_args::link

Definition at line 1360 of file vfs_syscalls.c.

Referenced by systrace_args().

8.111.2.2 char* link_args::path

Definition at line 1359 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.112 logsoftc Struct Reference

Data Fields

- int sc_state
- selinfo sc_selp
- sigio * sc_sigio
- callout sc_callout

8.112.1 Detailed Description

Definition at line 77 of file subr_log.c.

8.112.2 Field Documentation

8.112.2.1 struct callout logsoftc::sc_callout

Definition at line 81 of file subr_log.c.

8.112.2.2 struct selinfo logsoftc::sc_selp

Definition at line 79 of file subr_log.c.

8.112.2.3 struct sigio* logsoftc::sc_sigio

Definition at line 80 of file subr_log.c.

8.112.2.4 int logsoftc::sc_state

Definition at line 78 of file subr_log.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_log.c

8.113 lseek_args Struct Reference

Data Fields

- int fd
- int pad
- off_t offset
- int whence

8.113.1 Detailed Description

Definition at line 1715 of file vfs_syscalls.c.

8.113.2 Field Documentation

8.113.2.1 int lseek_args::fd

Definition at line 1716 of file vfs_syscalls.c.

Referenced by systrace_args().

8.113.2.2 off_t lseek_args::offset

Definition at line 1718 of file vfs_syscalls.c.

Referenced by systrace_args().

8.113.2.3 int lseek_args::pad

Definition at line 1717 of file vfs_syscalls.c.

Referenced by systrace_args().

8.113.2.4 int lseek_args::whence

Definition at line 1719 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.114 lstat_args Struct Reference

Data Fields

- char * path
- stat * ub
- char * path
- stat * ub

8.114.1 Detailed Description

Definition at line 2103 of file vfs_syscalls.c.

8.114.2 Field Documentation

8.114.2.1 char* lstat_args::path

Definition at line 2208 of file vfs_syscalls.c.

8.114.2.2 char* lstat_args::path

Definition at line 2104 of file vfs_syscalls.c.

Referenced by systrace_args().

8.114.2.3 struct stat* lstat_args::ub

Definition at line 2209 of file vfs_syscalls.c.

8.114.2.4 struct stat* lstat_args::ub

Definition at line 2105 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.115 lutimes_args Struct Reference

Data Fields

- char * path
- timeval * tptr

8.115.1 Detailed Description

Definition at line 2910 of file vfs_syscalls.c.

8.115.2 Field Documentation

8.115.2.1 char* lutimes_args::path

Definition at line 2911 of file vfs_syscalls.c.

Referenced by systrace_args().

8.115.2.2 struct timeval* lutimes_args::tptr

Definition at line 2912 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.116 mbfree Struct Reference

8.116.1 Detailed Description

Definition at line 69 of file subr_mbpool.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_mbpool.c

8.117 mbpage Struct Reference

Data Fields

- bus_dmamap_t map
- bus_addr_t phy
- void * va

8.117.1 Detailed Description

Definition at line 73 of file subr_mbpool.c.

8.117.2 Field Documentation

8.117.2.1 bus_dmamap_t mbpage::map

Definition at line 74 of file subr_mbpool.c.

Referenced by mbp_alloc_page(), and mbp_destroy().

8.117.2.2 bus_addr_t mbpage::phy

Definition at line 75 of file subr_mbpool.c.

Referenced by mbp_alloc_page().

8.117.2.3 void* mbpage::va

Definition at line 76 of file subr_mbpool.c.

Referenced by mbp_alloc_page(), and mbp_destroy().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_mbpool.c

8.118 mbpool Struct Reference

Data Fields

- const char * name
- bus_dma_tag_t dmat
- u_int max_pages
- size_t page_size
- size_t chunk_size
- mtx free_lock

8.118.1 Detailed Description

Definition at line 79 of file subr_mbpool.c.

8.118.2 Field Documentation

8.118.2.1 size_t mbpool::chunk_size

Definition at line 84 of file subr_mbpool.c.

Referenced by mbp_alloc(), and mbp_sync().

8.118.2.2 bus_dma_tag_t mbpool::dmat

Definition at line 81 of file subr_mbpool.c.

Referenced by mbp_alloc_page(), mbp_destroy(), and mbp_sync().

8.118.2.3 struct mtx mbpool::free_lock

Definition at line 86 of file subr_mbpool.c.

Referenced by mbp_alloc(), mbp_card_free(), mbp_count(), mbp_destroy(), and mbp_free().

8.118.2.4 u_int mbpool::max_pages

Definition at line 82 of file subr_mbpool.c.

Referenced by mbp_alloc_page().

8.118.2.5 const char* mbpool::name

Definition at line 80 of file subr_mbpool.c.

Referenced by mbp_alloc_page(), mbp_destroy(), mbp_get(), and mbp_get_keep().

8.118.2.6 size_t mbpool::page_size

Definition at line 83 of file subr_mbpool.c.

Referenced by mbp_alloc_page().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_mbpool.c

8.119 mbtrail Struct Reference

Data Fields

- uint16_t chunk
- uint16_t page

8.119.1 Detailed Description

Definition at line 60 of file subr_mbpool.c.

8.119.2 Field Documentation

8.119.2.1 uint16_t mbtrail::chunk

Definition at line 61 of file subr_mbpool.c.

Referenced by mbp_alloc(), and mbp_alloc_page().

8.119.2.2 uint16_t mbtrail::page

Definition at line 62 of file subr_mbpool.c.

Referenced by mbp_alloc(), mbp_alloc_page(), mbp_card_free(), mbp_count(), mbp_destroy(), mbp_free(), mbp_get(), and mbp_get_keep().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_mbpool.c

8.120 mkdir_args Struct Reference

Data Fields

- char * path
- int mode

8.120.1 Detailed Description

Definition at line 3368 of file vfs_syscalls.c.

8.120.2 Field Documentation

8.120.2.1 int mkdir_args::mode

Definition at line 3370 of file vfs_syscalls.c.

Referenced by systrace_args().

8.120.2.2 char* mkdir_args::path

Definition at line 3369 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.121 mkfifo_args Struct Reference

Data Fields

- char * path
- int mode

8.121.1 Detailed Description

Definition at line 1277 of file vfs_syscalls.c.

8.121.2 Field Documentation

8.121.2.1 int mkfifo_args::mode

Definition at line 1279 of file vfs_syscalls.c.

Referenced by systrace_args().

8.121.2.2 char* mkfifo_args::path

Definition at line 1278 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.122 mknod_args Struct Reference

Data Fields

- char * path
- int mode
- int dev

8.122.1 Detailed Description

Definition at line 1149 of file vfs_syscalls.c.

8.122.2 Field Documentation

8.122.2.1 int mknod_args::dev

Definition at line 1152 of file vfs_syscalls.c.

Referenced by systrace_args().

8.122.2.2 int mknod_args::mode

Definition at line 1151 of file vfs_syscalls.c.

Referenced by systrace_args().

8.122.2.3 char* mknod_args::path

Definition at line 1150 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.123 mntaarg Struct Reference

8.123.1 Detailed Description

Definition at line 2002 of file vfs_mount.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_mount.c

8.124 mntarg Struct Reference

Data Fields

- iovec * v
- int len
- int error

8.124.1 Detailed Description

Definition at line 2007 of file vfs_mount.c.

8.124.2 Field Documentation

8.124.2.1 int mntarg::error

Definition at line 2010 of file vfs_mount.c.

Referenced by kernel_mount(), kernel_vmount(), mount(), mount_arg(), and mount_argf().

8.124.2.2 int mntarg::len

Definition at line 2009 of file vfs_mount.c.

Referenced by kernel_mount(), mount_arg(), and mount_argf().

8.124.2.3 struct iovec* mntarg::v

Definition at line 2008 of file vfs_mount.c.

Referenced by free_mntarg(), kernel_mount(), mount_arg(), and mount_argf().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_mount.c

8.125 module_stat_v1 Struct Reference

Data Fields

- int version
- char name [MAXMODNAME]
- int refs
- int id

8.125.1 Detailed Description

Definition at line 343 of file kern_module.c.

8.125.2 Field Documentation

8.125.2.1 int module_stat_v1::id

Definition at line 347 of file kern_module.c.

8.125.2.2 char module_stat_v1::name[MAXMODNAME]

Definition at line 345 of file kern_module.c.

8.125.2.3 int module_stat_v1::refs

Definition at line 346 of file kern_module.c.

8.125.2.4 int module_stat_v1::version

Definition at line 344 of file kern_module.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_module.c

8.126 mount_args Struct Reference

Data Fields

- char * type
- char * path
- int flags
- caddr_t data

8.126.1 Detailed Description

Definition at line 731 of file vfs_mount.c.

8.126.2 Field Documentation

8.126.2.1 caddr_t mount_args::data

Definition at line 735 of file vfs_mount.c.

Referenced by mount(), and systrace_args().

8.126.2.2 int mount_args::flags

Definition at line 734 of file vfs_mount.c.

Referenced by mount(), and systrace_args().

8.126.2.3 char* mount_args::path

Definition at line 733 of file vfs_mount.c.

Referenced by mount(), and systrace_args().

8.126.2.4 char* mount_args::type

Definition at line 732 of file vfs_mount.c.

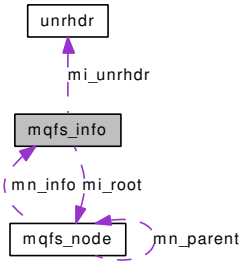
Referenced by mount(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_mount.c

8.127 mqfs_info Struct Reference

Collaboration diagram for mqfs_info:



Data Fields

- sx mi_lock
- mqfs_node ∗ mi_root
- unrhdr ∗ mi_unrhdr

8.127.1 Detailed Description

Definition at line 106 of file uipc_mqueue.c.

8.127.2 Field Documentation

8.127.2.1 struct sx mqfs_info::mi_lock

Definition at line 107 of file uipc_mqueue.c.

Referenced by do_unlink(), kmq_open(), kmq_unlink(), mqf_close(), mqfs_create(), mqfs_init(), mqfs_-lookup(), mqfs_readdir(), mqfs_reclaim(), mqfs_remove(), mqfs_root(), and mqfs_uninit().

8.127.2.2 struct mqfs_node∗ mqfs_info::mi_root

Definition at line 108 of file uipc_mqueue.c.

Referenced by kmq_open(), kmq_unlink(), mqfs_fileno_alloc(), mqfs_init(), mqfs_root(), and mqfs_-uninit().

8.127.2.3 struct unrhdr∗ mqfs_info::mi_unrhdr

Definition at line 109 of file uipc_mqueue.c.

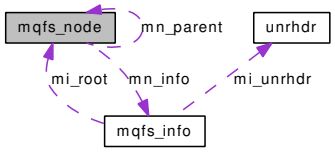
Referenced by mqfs_fileno_alloc(), mqfs_fileno_free(), mqfs_fileno_init(), and mqfs_fileno_uninit().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/uipc_mqueue.c

8.128 mqfs_node Struct Reference

Collaboration diagram for mqfs_node:



Data Fields

- char mn_name [MQFS_NAMELEN+1]
- mqfs_info * mn_info
- mqfs_node * mn_parent

8.128.1 Detailed Description

Definition at line 122 of file uipc_mqueue.c.

8.128.2 Field Documentation

8.128.2.1 struct mqfs_info* mqfs_node::mn_info

Definition at line 124 of file uipc_mqueue.c.

Referenced by do_unlink(), mqfs_add_node(), mqfs_destroy(), and mqfs_init().

8.128.2.2 char mqfs_node::mn_name[MQFS_NAMELEN+1]

Definition at line 123 of file uipc_mqueue.c.

Referenced by mqfs_allocv(), mqfs_create_node(), mqfs_fixup_dir(), mqfs_readdir(), and mqfs_search().

8.128.2.3 struct mqfs_node* mqfs_node::mn_parent

Definition at line 125 of file uipc_mqueue.c.

Referenced by do_unlink(), mqfs_add_node(), mqfs_destroy(), mqfs_fileno_alloc(), and mqfs_lookupx().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/uipc_mqueue.c

8.129 mqfs_vdata Struct Reference

8.129.1 Detailed Description

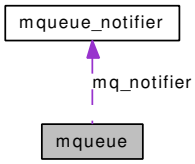
Definition at line 112 of file uipc_mqueue.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/uipc_mqueue.c

8.130 mqueue Struct Reference

Collaboration diagram for mqueue:



Data Fields

- mtx mq_mutex
- int mq_flags
- long mq_maxmsg
- long mq_msgsize
- long mq_curmsgs
- long mq_totalbytes
- msgq mq_msgq
- int mq_receivers
- int mq_senders
- selinfo mq_rsel
- selinfo mq_wsel
- mqueue_notifier * mq_notifier

8.130.1 Detailed Description

Definition at line 160 of file uipc_mqueue.c.

8.130.2 Field Documentation

8.130.2.1 long mqueue::mq_curmsgs

Definition at line 165 of file uipc_mqueue.c.

Referenced by _mqueue_recv(), _mqueue_send(), filt_mqread(), filt_mqwrite(), kmq_setattr(), mqf_poll(), and mqfs_read().

8.130.2.2 int mqueue::mq_flags

Definition at line 162 of file uipc_mqueue.c.

Referenced by _mqueue_recv(), mqf_poll(), and mqueue_fdclose().

8.130.2.3 long mqueue::mq_maxmsg

Definition at line 163 of file uipc_mqueue.c.

Referenced by _mqueue_send(), filt_mqwrite(), kmq_setattr(), mqf_poll(), mqfs_read(), and mqueue_alloc().

8.130.2.4 struct msgq mqueue::mq_msgq

Definition at line 167 of file uipc_mqueue.c.

Referenced by _mqueue_recv(), _mqueue_send(), kmq_notify(), mqueue_alloc(), and mqueue_free().

8.130.2.5 long mqueue::mq_msgsize

Definition at line 164 of file uipc_mqueue.c.

Referenced by kmq_setattr(), mqfs_read(), mqueue_alloc(), and mqueue_send().

8.130.2.6 struct mtx mqueue::mq_mutex

Definition at line 161 of file uipc_mqueue.c.

Referenced by _mqueue_recv(), _mqueue_send(), filt_mqread(), filt_mqwrite(), kmq_notify(), mq_proc_exit(), mqf_poll(), mqueue_alloc(), mqueue_fdclose(), mqueue_free(), mqueue_send_notification(), and notifier_remove().

8.130.2.7 struct mqueue_notifier* mqueue::mq_notifier

Definition at line 172 of file uipc_mqueue.c.

Referenced by _mqueue_recv(), _mqueue_send(), kmq_notify(), mqueue_send_notification(), and notifier_remove().

8.130.2.8 int mqueue::mq_receivers

Definition at line 168 of file uipc_mqueue.c.

Referenced by _mqueue_recv(), _mqueue_send(), and kmq_notify().

8.130.2.9 struct selinfo mqueue::mq_rsel

Definition at line 170 of file uipc_mqueue.c.

Referenced by filt_mqdetach(), mqf_kqfilter(), mqf_poll(), mqueue_alloc(), mqueue_fdclose(), and mqueue_free().

8.130.2.10 int mqueue::mq_senders

Definition at line 169 of file uipc_mqueue.c.

Referenced by _mqueue_recv(), and _mqueue_send().

8.130.2.11 long mqueue::mq_totalbytes

Definition at line 166 of file uipc_mqueue.c.

Referenced by _mqueue_recv(), _mqueue_send(), and mqfs_read().

8.130.2.12 struct selinfo mqueue::mq_wsel

Definition at line 171 of file uipc_mqueue.c.

Referenced by _mqueue_recv(), filt_mqdetach(), mqf_kqfilter(), mqf_poll(), mqueue_alloc(), mqueue_-fdclose(), and mqueue_free().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/uipc_mqueue.c

8.131 mqueue_msg Struct Reference

8.131.1 Detailed Description

Definition at line 178 of file uipc_mqueue.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/uipc_mqueue.c

8.132 mqueue_notifier Struct Reference

8.132.1 Detailed Description

Definition at line 153 of file uipc_mqueue.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/uipc_mqueue.c

8.133 msgctl_args Struct Reference

Data Fields

- int msqid
- int cmd
- msqid_ds * buf

8.133.1 Detailed Description

Definition at line 379 of file sysv_msg.c.

8.133.2 Field Documentation

8.133.2.1 struct msqid_ds* msgctl_args::buf

Definition at line 382 of file sysv_msg.c.

Referenced by systrace_args().

8.133.2.2 int msgctl_args::cmd

Definition at line 381 of file sysv_msg.c.

Referenced by systrace_args().

8.133.2.3 int msgctl_args::msqid

Definition at line 380 of file sysv_msg.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sysv_msg.c

8.134 msgget_args Struct Reference

Data Fields

- key_t key
- int msgflg

8.134.1 Detailed Description

Definition at line 556 of file sysv_msg.c.

8.134.2 Field Documentation

8.134.2.1 key_t msgget_args::key

Definition at line 557 of file sysv_msg.c.

Referenced by systrace_args().

8.134.2.2 int msgget_args::msgflg

Definition at line 558 of file sysv_msg.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sysv_msg.c

8.135 msgmap Struct Reference

Data Fields

- short next

8.135.1 Detailed Description

Definition at line 144 of file sysv_msg.c.

8.135.2 Field Documentation

8.135.2.1 short msgmap::next

Definition at line 145 of file sysv_msg.c.

Referenced by kern_msgrcv(), msg_freehdr(), and msginit().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sysv_msg.c

8.136 msgrcv_args Struct Reference

Data Fields

- int msqid
- void * msgp
- size_t msgsz
- long msgtyp
- int msgflg

8.136.1 Detailed Description

Definition at line 1006 of file sysv_msg.c.

8.136.2 Field Documentation

8.136.2.1 int msgrcv_args::msgflg

Definition at line 1011 of file sysv_msg.c.

Referenced by systrace_args().

8.136.2.2 void* msgrcv_args::msgp

Definition at line 1008 of file sysv_msg.c.

Referenced by systrace_args().

8.136.2.3 size_t msgrcv_args::msgsz

Definition at line 1009 of file sysv_msg.c.

Referenced by systrace_args().

8.136.2.4 long msgrcv_args::msgtyp

Definition at line 1010 of file sysv_msg.c.

Referenced by systrace_args().

8.136.2.5 int msgrcv_args::msqid

Definition at line 1007 of file sysv_msg.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sysv_msg.c

8.137 msgsnd_args Struct Reference

Data Fields

- int msqid
- const void * msgp
- size_t msgsz
- int msgflg

8.137.1 Detailed Description

Definition at line 667 of file sysv_msg.c.

8.137.2 Field Documentation

8.137.2.1 int msgsnd_args::msgflg

Definition at line 671 of file sysv_msg.c.

Referenced by systrace_args().

8.137.2.2 const void* msgsnd_args::msgp

Definition at line 669 of file sysv_msg.c.

Referenced by systrace_args().

8.137.2.3 size_t msgsnd_args::msgsz

Definition at line 670 of file sysv_msg.c.

Referenced by systrace_args().

8.137.2.4 int msgsnd_args::msqid

Definition at line 668 of file sysv_msg.c.

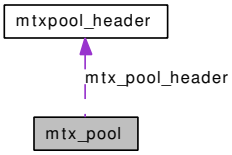
Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sysv_msg.c

8.138 mtx_pool Struct Reference

Collaboration diagram for mtx_pool:



Data Fields

- mtxpool_header mtx_pool_header
- mtx mtx_pool_ary [1]

8.138.1 Detailed Description

Definition at line 76 of file kern_mtxpool.c.

8.138.2 Field Documentation

8.138.2.1 struct mtx mtx_pool::mtx_pool_ary[1]

Definition at line 78 of file kern_mtxpool.c.

Referenced by mtx_pool_alloc(), mtx_pool_destroy(), mtx_pool_find(), and mtx_pool_initialize().

8.138.2.2 struct mtxpool_header mtx_pool::mtx_pool_header

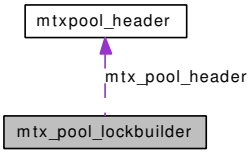
Definition at line 77 of file kern_mtxpool.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_mtxpool.c

8.139 mtx_pool_lockbuilder Struct Reference

Collaboration diagram for mtx_pool_lockbuilder:



Data Fields

- mtxpool_header mtx_pool_header
- mtx mtx_pool_ary [MTX_POOL_LOCKBUILDER_SIZE]

8.139.1 Detailed Description

Definition at line 81 of file kern_mtxpool.c.

8.139.2 Field Documentation

8.139.2.1 struct mtx mtx_pool_lockbuilder::mtx_pool_ary[MTX_POOL_LOCKBUILDER_SIZE]

Definition at line 83 of file kern_mtxpool.c.

8.139.2.2 struct mtxpool_header mtx_pool_lockbuilder::mtx_pool_header

Definition at line 82 of file kern_mtxpool.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_mtxpool.c

8.140 mtxpool_header Struct Reference

Data Fields

- int mtxpool_size
- int mtxpool_mask
- int mtxpool_shift
- int mtxpool_next

8.140.1 Detailed Description

Definition at line 69 of file kern_mtxpool.c.

8.140.2 Field Documentation

8.140.2.1 int mtxpool_header::mtxpool_mask

Definition at line 71 of file kern_mtxpool.c.

8.140.2.2 int mtxpool_header::mtxpool_next

Definition at line 73 of file kern_mtxpool.c.

8.140.2.3 int mtxpool_header::mtxpool_shift

Definition at line 72 of file kern_mtxpool.c.

8.140.2.4 int mtxpool_header::mtxpool_size

Definition at line 70 of file kern_mtxpool.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_mtxpool.c

8.141 namecache Struct Reference

8.141.1 Detailed Description

Definition at line 61 of file vfs_cache.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_cache.c

8.142 nanosleep_args Struct Reference

Data Fields

- timespec * rqtp
- timespec * rmtp

8.142.1 Detailed Description

Definition at line 399 of file kern_time.c.

8.142.2 Field Documentation

8.142.2.1 struct timespec* nanosleep_args::rmtp

Definition at line 401 of file kern_time.c.

Referenced by nanosleep(), and systrace_args().

8.142.2.2 struct timespec* nanosleep_args::rqtp

Definition at line 400 of file kern_time.c.

Referenced by nanosleep(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_time.c

8.143 netcred Struct Reference

Data Fields

- radix_node netc_rnodes [2]
- int netc_exflags
- ucred netc_anon

8.143.1 Detailed Description

Definition at line 67 of file vfs_export.c.

8.143.2 Field Documentation

8.143.2.1 struct ucred netcred::netc_anon

Definition at line 70 of file vfs_export.c.

Referenced by vfs_hang_addrlist(), and vfs_stdcheckexp().

8.143.2.2 int netcred::netc_exflags

Definition at line 69 of file vfs_export.c.

Referenced by vfs_hang_addrlist(), and vfs_stdcheckexp().

8.143.2.3 struct radix_node netcred::netc_rnodes[2]

Definition at line 68 of file vfs_export.c.

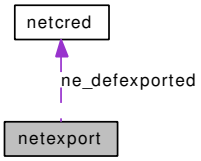
Referenced by vfs_export_lookup().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_export.c

8.144 netexport Struct Reference

Collaboration diagram for netexport:



Data Fields

- netcred ne_defexported
- radix_node_head ∗ ne_rtable [AF_MAX+1]

8.144.1 Detailed Description

Definition at line 76 of file vfs_export.c.

8.144.2 Field Documentation

8.144.2.1 struct netcred netexport::ne_defexported

Definition at line 77 of file vfs_export.c.

Referenced by vfs_export_lookup(), and vfs_hang_addrlist().

8.144.2.2 struct radix_node_head∗ netexport::ne_rtable[AF_MAX+1]

Definition at line 78 of file vfs_export.c.

Referenced by vfs_export_lookup(), vfs_free_addrlist(), and vfs_hang_addrlist().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_export.c

8.145 netsend_cow_stats Struct Reference

Data Fields

- int attempted
- int fail_not_mapped
- int fail_sf_buf
- int success
- int iodone

8.145.1 Detailed Description

Definition at line 61 of file uipc_cow.c.

8.145.2 Field Documentation

8.145.2.1 int netsend_cow_stats::attempted

Definition at line 62 of file uipc_cow.c.

Referenced by socow_setup().

8.145.2.2 int netsend_cow_stats::fail_not_mapped

Definition at line 63 of file uipc_cow.c.

Referenced by socow_setup().

8.145.2.3 int netsend_cow_stats::fail_sf_buf

Definition at line 64 of file uipc_cow.c.

Referenced by socow_setup().

8.145.2.4 int netsend_cow_stats::iodone

Definition at line 66 of file uipc_cow.c.

Referenced by socow_iodone().

8.145.2.5 int netsend_cow_stats::success

Definition at line 65 of file uipc_cow.c.

Referenced by socow_setup().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/uipc_cow.c

8.146 nfstat_args Struct Reference

Data Fields

- int fd
- nstat * sb

8.146.1 Detailed Description

Definition at line 1118 of file kern_descrip.c.

8.146.2 Field Documentation

8.146.2.1 int nfstat_args::fd

Definition at line 1119 of file kern_descrip.c.

Referenced by nfstat(), and systrace_args().

8.146.2.2 struct nstat* nfstat_args::sb

Definition at line 1120 of file kern_descrip.c.

Referenced by nfstat(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_descrip.c

8.147 nosys_args Struct Reference

Data Fields

- int dummy

8.147.1 Detailed Description

Definition at line 3221 of file kern_sig.c.

8.147.2 Field Documentation

8.147.2.1 int nosys_args::dummy

Definition at line 3222 of file kern_sig.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_sig.c

8.148 nstat_args Struct Reference

Data Fields

- char * path
- nstat * ub

8.148.1 Detailed Description

Definition at line 2178 of file vfs_syscalls.c.

8.148.2 Field Documentation

8.148.2.1 char* nstat_args::path

Definition at line 2179 of file vfs_syscalls.c.

Referenced by systrace_args().

8.148.2.2 struct nstat* nstat_args::ub

Definition at line 2180 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.149 ntp_adjtime_args Struct Reference

Data Fields

- timex * tp

8.149.1 Detailed Description

Definition at line 305 of file kern_ntptime.c.

8.149.2 Field Documentation

8.149.2.1 struct timex* ntp_adjtime_args::tp

Definition at line 306 of file kern_ntptime.c.

Referenced by ntp_adjtime(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_ntptime.c

8.150 ntp_gettime_args Struct Reference

Data Fields

- ntptimeval * ntvp

8.150.1 Detailed Description

Definition at line 257 of file kern_ntptime.c.

8.150.2 Field Documentation

8.150.2.1 struct ntptimeval* ntp_gettime_args::ntvp

Definition at line 258 of file kern_ntptime.c.

Referenced by ntp_gettime(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_ntptime.c

8.151 oaiocb Struct Reference

Data Fields

- int aio_fildes
- off_t aio_offset
- volatile void * aio_buf
- size_t aio_nbytes
- osigevent aio_sigevent
- int aio_lio_opcode
- int aio_reqprio
- __aiocb_private _aiocb_private

8.151.1 Detailed Description

Definition at line 184 of file vfs_aio.c.

8.151.2 Field Documentation

8.151.2.1 struct __aiocb_private oaiocb::_aiocb_private

Definition at line 192 of file vfs_aio.c.

8.151.2.2 volatile void* oaiocb::aio_buf

Definition at line 187 of file vfs_aio.c.

8.151.2.3 int oaiocb::aio_fildes

Definition at line 185 of file vfs_aio.c.

8.151.2.4 int oaiocb::aio_lio_opcode

Definition at line 190 of file vfs_aio.c.

8.151.2.5 size_t oaiocb::aio_nbytes

Definition at line 188 of file vfs_aio.c.

8.151.2.6 off_t oaiocb::aio_offset

Definition at line 186 of file vfs_aio.c.

8.151.2.7 int oaiocb::aio_reqprio

Definition at line 191 of file vfs_aio.c.

8.151.2.8 struct osigevent oaiocb::aio_sigevent

Definition at line 189 of file vfs_aio.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_aio.c

8.152 open_args Struct Reference

Data Fields

- char * path
- int flags
- int mode

8.152.1 Detailed Description

Definition at line 940 of file vfs_syscalls.c.

8.152.2 Field Documentation

8.152.2.1 int open_args::flags

Definition at line 942 of file vfs_syscalls.c.

Referenced by systrace_args().

8.152.2.2 int open_args::mode

Definition at line 943 of file vfs_syscalls.c.

Referenced by systrace_args().

8.152.2.3 char* open_args::path

Definition at line 941 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.153 openbsd_poll_args Struct Reference

Data Fields

- pollfd * fds
- u_int nfds
- int timeout

8.153.1 Detailed Description

Definition at line 1058 of file sys_generic.c.

8.153.2 Field Documentation

8.153.2.1 struct pollfd* openbsd_poll_args::fds

Definition at line 1059 of file sys_generic.c.

Referenced by systrace_args().

8.153.2.2 u_int openbsd_poll_args::nfds

Definition at line 1060 of file sys_generic.c.

Referenced by systrace_args().

8.153.2.3 int openbsd_poll_args::timeout

Definition at line 1061 of file sys_generic.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sys_generic.c

8.154 pathconf_args Struct Reference

Data Fields

- char * path
- int name

8.154.1 Detailed Description

Definition at line 2236 of file vfs_syscalls.c.

8.154.2 Field Documentation

8.154.2.1 int pathconf_args::name

Definition at line 2238 of file vfs_syscalls.c.

Referenced by systrace_args().

8.154.2.2 char* pathconf_args::path

Definition at line 2237 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.155 phdr_closure Struct Reference

Data Fields

- Elf_Phdr * phdr
- Elf_Off offset

8.155.1 Detailed Description

Definition at line 875 of file imgact_elf.c.

8.155.2 Field Documentation

8.155.2.1 Elf_Off phdr_closure::offset

Definition at line 877 of file imgact_elf.c.

Referenced by cb_put_phdr().

8.155.2.2 Elf_Phdr* phdr_closure::phdr

Definition at line 876 of file imgact_elf.c.

Referenced by cb_put_phdr().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/imgact_elf.c

8.156 poll_args Struct Reference

Data Fields

- pollfd * fds
- u_int nfds
- int timeout

8.156.1 Detailed Description

Definition at line 886 of file sys_generic.c.

8.156.2 Field Documentation

8.156.2.1 struct pollfd* poll_args::fds

Definition at line 887 of file sys_generic.c.

Referenced by poll(), and systrace_args().

8.156.2.2 u_int poll_args::nfds

Definition at line 888 of file sys_generic.c.

Referenced by poll(), and systrace_args().

8.156.2.3 int poll_args::timeout

Definition at line 889 of file sys_generic.c.

Referenced by poll(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sys_generic.c

8.157 pollrec Struct Reference

Data Fields

- poll_handler_t * handler
- ifnet * ifp

8.157.1 Detailed Description

Definition at line 253 of file kern_poll.c.

8.157.2 Field Documentation

8.157.2.1 poll_handler_t* pollrec::handler

Definition at line 254 of file kern_poll.c.

Referenced by ether_poll_register().

8.157.2.2 struct ifnet* pollrec::ifp

Definition at line 255 of file kern_poll.c.

Referenced by ether_poll_deregister(), and ether_poll_register().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_poll.c

8.158 pread_args Struct Reference

Data Fields

- int fd
- void * buf
- size_t nbyte
- int pad
- off_t offset

8.158.1 Detailed Description

Definition at line 124 of file sys_generic.c.

8.158.2 Field Documentation

8.158.2.1 void* pread_args::buf

Definition at line 126 of file sys_generic.c.

Referenced by pread(), and systrace_args().

8.158.2.2 int pread_args::fd

Definition at line 125 of file sys_generic.c.

Referenced by pread(), and systrace_args().

8.158.2.3 size_t pread_args::nbyte

Definition at line 127 of file sys_generic.c.

Referenced by pread(), and systrace_args().

8.158.2.4 off_t pread_args::offset

Definition at line 129 of file sys_generic.c.

Referenced by pread(), and systrace_args().

8.158.2.5 int pread_args::pad

Definition at line 128 of file sys_generic.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sys_generic.c

8.159 preadv_args Struct Reference

Data Fields

- int fd
- iovec * iovp
- u_int iovcnt
- off_t offset

8.159.1 Detailed Description

Definition at line 201 of file sys_generic.c.

8.159.2 Field Documentation

8.159.2.1 int preadv_args::fd

Definition at line 202 of file sys_generic.c.

Referenced by preadv(), and systrace_args().

8.159.2.2 u_int preadv_args::iovcnt

Definition at line 204 of file sys_generic.c.

Referenced by preadv(), and systrace_args().

8.159.2.3 struct iovec* preadv_args::iovp

Definition at line 203 of file sys_generic.c.

Referenced by preadv(), and systrace_args().

8.159.2.4 off_t preadv_args::offset

Definition at line 205 of file sys_generic.c.

Referenced by preadv(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sys_generic.c

8.160 priv_fw Struct Reference

Collaboration diagram for priv_fw:

Data Fields

- int refcnt
- priv_fw * parent
- int flags
- linker_file_t file
- firmware fw

8.160.1 Detailed Description

Definition at line 77 of file subr_firmware.c.

8.160.2 Field Documentation

8.160.2.1 linker_file_t priv_fw::file

Definition at line 95 of file subr_firmware.c.

Referenced by firmware_get(), firmware_put(), and firmware_unregister().

8.160.2.2 int priv_fw::flags

Definition at line 86 of file subr_firmware.c.

Referenced by firmware_modevent(), and firmware_put().

8.160.2.3 struct firmware priv_fw::fw

Definition at line 104 of file subr_firmware.c.

Referenced by firmware_get(), firmware_modevent(), and lookup().

8.160.2.4 struct priv_fw* priv_fw::parent

Definition at line 84 of file subr_firmware.c.

Referenced by firmware_unregister().

8.160.2.5 int priv_fw::refcnt

Definition at line 78 of file subr_firmware.c.

Referenced by firmware_get(), firmware_modevent(), firmware_put(), and firmware_unregister().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_firmware.c

8.161 profil_args Struct Reference

Data Fields

- caddr_t samples
- size_t size
- size_t offset
- u_int scale

8.161.1 Detailed Description

Definition at line 398 of file subr_prof.c.

8.161.2 Field Documentation

8.161.2.1 size_t profil_args::offset

Definition at line 401 of file subr_prof.c.

Referenced by systrace_args().

8.161.2.2 caddr_t profil_args::samples

Definition at line 399 of file subr_prof.c.

Referenced by systrace_args().

8.161.2.3 u_int profil_args::scale

Definition at line 402 of file subr_prof.c.

Referenced by systrace_args().

8.161.2.4 size_t profil_args::size

Definition at line 400 of file subr_prof.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_prof.c

8.162 pt_desc Struct Reference

Data Fields

- int pt_num

8.162.1 Detailed Description

Definition at line 137 of file tty_pts.c.

8.162.2 Field Documentation

8.162.2.1 int pt_desc::pt_num

Definition at line 138 of file tty_pts.c.

Referenced by ptcioctl(), ptcopen(), pty_clone(), pty_maybecleanup(), and pty_release().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/tty_pts.c

8.163 ptrace_args Struct Reference

Data Fields

- int req
- pid_t pid
- caddr_t addr
- int data

8.163.1 Detailed Description

Definition at line 342 of file sys_process.c.

8.163.2 Field Documentation

8.163.2.1 caddr_t ptrace_args::addr

Definition at line 345 of file sys_process.c.

Referenced by ptrace(), and systrace_args().

8.163.2.2 int ptrace_args::data

Definition at line 346 of file sys_process.c.

Referenced by ptrace(), and systrace_args().

8.163.2.3 pid_t ptrace_args::pid

Definition at line 344 of file sys_process.c.

Referenced by ptrace(), and systrace_args().

8.163.2.4 int ptrace_args::req

Definition at line 343 of file sys_process.c.

Referenced by ptrace(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sys_process.c

8.164 ptsc Struct Reference

Data Fields

- int pt_flags
- selinfo pt_selr pt_selw
- u_char pt_send
- u_char pt_ucntl
- tty * pt_tty
- cdev * devs
- cdev * devc
- int pt_devs_open
- int pt_devc_open
- prison * pt_prison

8.164.1 Detailed Description

Definition at line 104 of file tty_pty.c.

8.164.2 Field Documentation

8.164.2.1 struct cdev * ptsc::devc

Definition at line 110 of file tty_pty.c.

8.164.2.2 struct cdev* ptsc::devs

Definition at line 110 of file tty_pty.c.

Referenced by ptcopen(), ptsclose(), pty_create_slave(), and pty_destroy_slave().

8.164.2.3 int ptsc::pt_devc_open

Definition at line 111 of file tty_pty.c.

Referenced by ptcclose(), ptcopen(), and pty_maybe_destroy_slave().

8.164.2.4 int ptsc::pt_devs_open

Definition at line 111 of file tty_pty.c.

Referenced by ptsclose(), ptsopen(), and pty_maybe_destroy_slave().

8.164.2.5 int ptsc::pt_flags

Definition at line 105 of file tty_pty.c.

Referenced by ptcioctl(), ptcopen(), ptcpoll(), ptcread(), ptsioctl(), ptsstart(), and ptsstop().

8.164.2.6 struct prison* ptsc::pt_prison

Definition at line 112 of file tty_pty.c.

Referenced by ptcopen(), and ptsopen().

8.164.2.7 struct selinfo pt_selr ptsc::pt_selw

Definition at line 106 of file tty_pty.c.

Referenced by ptcpoll(), and ptcwakeup().

8.164.2.8 u_char ptsc::pt_send

Definition at line 107 of file tty_pty.c.

Referenced by ptcopen(), ptcpoll(), ptcread(), ptsioctl(), ptsstart(), and ptsstop().

8.164.2.9 struct tty* ptsc::pt_tty

Definition at line 109 of file tty_pty.c.

Referenced by ptcopen(), pty_create_slave(), and pty_destroy_slave().

8.164.2.10 u_char ptsc::pt_ucntl

Definition at line 108 of file tty_pty.c.

Referenced by ptcopen(), ptcpoll(), and ptsioctl().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/tty_pty.c

8.165 putchar_arg Struct Reference

Data Fields

- int flags
- int pri
- tty * tty
- char * p_bufr
- size_t n_bufr
- char * p_next
- size_t remain

8.165.1 Detailed Description

Definition at line 79 of file subr_prf.c.

8.165.2 Field Documentation

8.165.2.1 int putchar_arg::flags

Definition at line 80 of file subr_prf.c.

Referenced by log(), printf(), putchar(), tprintf(), ttyprintf(), uprintf(), and vprintf().

8.165.2.2 size_t putchar_arg::n_bufr

Definition at line 84 of file subr_prf.c.

Referenced by printf(), putcons(), and vprintf().

8.165.2.3 char* putchar_arg::p_bufr

Definition at line 83 of file subr_prf.c.

Referenced by log(), printf(), putcons(), and vprintf().

8.165.2.4 char* putchar_arg::p_next

Definition at line 85 of file subr_prf.c.

Referenced by printf(), putcons(), and vprintf().

8.165.2.5 int putchar_arg::pri

Definition at line 81 of file subr_prf.c.

Referenced by log(), log_console(), printf(), putchar(), tprintf(), and vprintf().

8.165.2.6 size_t putchar_arg::remain

Definition at line 86 of file subr_prf.c.

Referenced by printf(), putcons(), and vprintf().

8.165.2.7 struct tty* putchar_arg::tty

Definition at line 82 of file subr_prf.c.

Referenced by log(), printf(), putchar(), tprintf(), ttyprintf(), uprintf(), and vprintf().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_prf.c

8.166 pwrite_args Struct Reference

Data Fields

- int fd
- const void * buf
- size_t nbyte
- int pad
- off_t offset

8.166.1 Detailed Description

Definition at line 334 of file sys_generic.c.

8.166.2 Field Documentation

8.166.2.1 const void* pwrite_args::buf

Definition at line 336 of file sys_generic.c.

Referenced by pwrite(), and systrace_args().

8.166.2.2 int pwrite_args::fd

Definition at line 335 of file sys_generic.c.

Referenced by pwrite(), and systrace_args().

8.166.2.3 size_t pwrite_args::nbyte

Definition at line 337 of file sys_generic.c.

Referenced by pwrite(), and systrace_args().

8.166.2.4 off_t pwrite_args::offset

Definition at line 339 of file sys_generic.c.

Referenced by pwrite(), and systrace_args().

8.166.2.5 int pwrite_args::pad

Definition at line 338 of file sys_generic.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sys_generic.c

8.167 pwritev_args Struct Reference

Data Fields

- int fd
- iovec * iovp
- u_int iovcnt
- off_t offset

8.167.1 Detailed Description

Definition at line 411 of file sys_generic.c.

8.167.2 Field Documentation

8.167.2.1 int pwritev_args::fd

Definition at line 412 of file sys_generic.c.

Referenced by pwritev(), and systrace_args().

8.167.2.2 u_int pwritev_args::iovcnt

Definition at line 414 of file sys_generic.c.

Referenced by pwritev(), and systrace_args().

8.167.2.3 struct iovec* pwritev_args::iovp

Definition at line 413 of file sys_generic.c.

Referenced by pwritev(), and systrace_args().

8.167.2.4 off_t pwritev_args::offset

Definition at line 415 of file sys_generic.c.

Referenced by pwritev(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sys_generic.c

8.168 quotactl_args Struct Reference

Data Fields

- char * path
- int cmd
- int uid
- caddr_t arg

8.168.1 Detailed Description

Definition at line 165 of file vfs_syscalls.c.

8.168.2 Field Documentation

8.168.2.1 caddr_t quotactl_args::arg

Definition at line 169 of file vfs_syscalls.c.

Referenced by systrace_args().

8.168.2.2 int quotactl_args::cmd

Definition at line 167 of file vfs_syscalls.c.

Referenced by systrace_args().

8.168.2.3 char* quotactl_args::path

Definition at line 166 of file vfs_syscalls.c.

Referenced by systrace_args().

8.168.2.4 int quotactl_args::uid

Definition at line 168 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.169 read_args Struct Reference

Data Fields

- int fd
- void * buf
- size_t nbyte

8.169.1 Detailed Description

Definition at line 90 of file sys_generic.c.

8.169.2 Field Documentation

8.169.2.1 void* read_args::buf

Definition at line 92 of file sys_generic.c.

Referenced by read(), and systrace_args().

8.169.2.2 int read_args::fd

Definition at line 91 of file sys_generic.c.

Referenced by read(), and systrace_args().

8.169.2.3 size_t read_args::nbyte

Definition at line 93 of file sys_generic.c.

Referenced by read(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sys_generic.c

8.170 readlink_args Struct Reference

Data Fields

- char * path
- char * buf
- int count

8.170.1 Detailed Description

Definition at line 2280 of file vfs_syscalls.c.

8.170.2 Field Documentation

8.170.2.1 char* readlink_args::buf

Definition at line 2282 of file vfs_syscalls.c.

Referenced by systrace_args().

8.170.2.2 int readlink_args::count

Definition at line 2283 of file vfs_syscalls.c.

Referenced by systrace_args().

8.170.2.3 char* readlink_args::path

Definition at line 2281 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.171 readv_args Struct Reference

Data Fields

- int fd
- iovec * iovp
- u_int iovcnt

8.171.1 Detailed Description

Definition at line 160 of file sys_generic.c.

8.171.2 Field Documentation

8.171.2.1 int readv_args::fd

Definition at line 161 of file sys_generic.c.

Referenced by readv(), and systrace_args().

8.171.2.2 u_int readv_args::iovcnt

Definition at line 163 of file sys_generic.c.

Referenced by readv(), and systrace_args().

8.171.2.3 struct iovec* readv_args::iovp

Definition at line 162 of file sys_generic.c.

Referenced by readv(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sys_generic.c

8.172 rename_args Struct Reference

Data Fields

- char * from
- char * to

8.172.1 Detailed Description

Definition at line 3233 of file vfs_syscalls.c.

8.172.2 Field Documentation

8.172.2.1 char* rename_args::from

Definition at line 3234 of file vfs_syscalls.c.

Referenced by systrace_args().

8.172.2.2 char* rename_args::to

Definition at line 3235 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.173 resource_i Struct Reference

Data Fields

- resource r_r

8.173.1 Detailed Description

Definition at line 82 of file subr_rman.c.

8.173.2 Field Documentation

8.173.2.1 struct resource resource_i::r_r

Definition at line 83 of file subr_rman.c.

Referenced by rman_reserve_resource_bound().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_rman.c

8.174 revoke_args Struct Reference

Data Fields

- char * path

8.174.1 Detailed Description

Definition at line 3833 of file vfs_syscalls.c.

8.174.2 Field Documentation

8.174.2.1 char* revoke_args::path

Definition at line 3834 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.175 rmdir_args Struct Reference

Data Fields

- char * path

8.175.1 Detailed Description

Definition at line 3457 of file vfs_syscalls.c.

8.175.2 Field Documentation

8.175.2.1 char* rmdir_args::path

Definition at line 3458 of file vfs_syscalls.c.

Referenced by rmdir(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.176 root_hold_token Struct Reference

Data Fields

- const char * who

8.176.1 Detailed Description

Definition at line 1312 of file vfs_mount.c.

8.176.2 Field Documentation

8.176.2.1 const char* root_hold_token::who

Definition at line 1313 of file vfs_mount.c.

Referenced by root_mount_wait().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_mount.c

8.177 rtprio_args Struct Reference

Data Fields

- int function
- pid_t pid
- rtprio * rtp

8.177.1 Detailed Description

Definition at line 377 of file kern_resource.c.

8.177.2 Field Documentation

8.177.2.1 int rtprio_args::function

Definition at line 378 of file kern_resource.c.

Referenced by systrace_args().

8.177.2.2 pid_t rtprio_args::pid

Definition at line 379 of file kern_resource.c.

Referenced by systrace_args().

8.177.2.3 struct rtprio* rtprio_args::rtp

Definition at line 380 of file kern_resource.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_resource.c

8.178 rtprio_thread_args Struct Reference

Data Fields

- int function
- lwpid_t lwpid
- rtprio * rtp

8.178.1 Detailed Description

Definition at line 282 of file kern_resource.c.

8.178.2 Field Documentation

8.178.2.1 int rtprio_thread_args::function

Definition at line 283 of file kern_resource.c.

Referenced by rtprio_thread(), and systrace_args().

8.178.2.2 lwpid_t rtprio_thread_args::lwpid

Definition at line 284 of file kern_resource.c.

Referenced by rtprio_thread(), and systrace_args().

8.178.2.3 struct rtprio* rtprio_thread_args::rtp

Definition at line 285 of file kern_resource.c.

Referenced by rtprio_thread(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_resource.c

8.179 select_args Struct Reference

Data Fields

- int nd
- fd_set * in
- fd_set * ou
- fd_set * ex
- timeval * tv

8.179.1 Detailed Description

Definition at line 654 of file sys_generic.c.

8.179.2 Field Documentation

8.179.2.1 fd_set * select_args::ex

Definition at line 656 of file sys_generic.c.

Referenced by systrace_args().

8.179.2.2 fd_set* select_args::in

Definition at line 656 of file sys_generic.c.

Referenced by systrace_args().

8.179.2.3 int select_args::nd

Definition at line 655 of file sys_generic.c.

Referenced by systrace_args().

8.179.2.4 fd_set * select_args::ou

Definition at line 656 of file sys_generic.c.

Referenced by systrace_args().

8.179.2.5 struct timeval* select_args::tv

Definition at line 657 of file sys_generic.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sys_generic.c

8.180 sem_undo Struct Reference

Public Member Functions

- SLIST_ENTRY (sem_undo)

Data Fields

- un_ent [1]

8.180.1 Detailed Description

Definition at line 124 of file sysv_sem.c.

8.180.2 Member Function Documentation

8.180.2.1 sem_undo::SLIST_ENTRY (sem_undo) [inline]

Definition at line 125 of file sysv_sem.c.

8.180.3 Field Documentation

8.180.3.1 sem_undo::un_ent[1]

Definition at line 132 of file sysv_sem.c.

Referenced by semexit_myhook(), semundo_adjust(), and semundo_clear().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sysv_sem.c

8.181 semget_args Struct Reference

Data Fields

- key_t key
- int nsems
- int semflg

8.181.1 Detailed Description

Definition at line 870 of file sysv_sem.c.

8.181.2 Field Documentation

8.181.2.1 key_t semget_args::key

Definition at line 871 of file sysv_sem.c.

Referenced by semget(), and systrace_args().

8.181.2.2 int semget_args::nsems

Definition at line 872 of file sysv_sem.c.

Referenced by semget(), and systrace_args().

8.181.2.3 int semget_args::semflg

Definition at line 873 of file sysv_sem.c.

Referenced by semget(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sysv_sem.c

8.182 semop_args Struct Reference

Data Fields

- int semid
- sembuf * sops
- size_t nsops

8.182.1 Detailed Description

Definition at line 986 of file sysv_sem.c.

8.182.2 Field Documentation

8.182.2.1 size_t semop_args::nsops

Definition at line 989 of file sysv_sem.c.

Referenced by semop(), and systrace_args().

8.182.2.2 int semop_args::semid

Definition at line 987 of file sysv_sem.c.

Referenced by semop(), and systrace_args().

8.182.2.3 struct sembuf* semop_args::sops

Definition at line 988 of file sysv_sem.c.

Referenced by semop(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sysv_sem.c

8.183 setdomainname_args Struct Reference

Data Fields

- char * domainname
- int len

8.183.1 Detailed Description

Definition at line 283 of file kern_xxx.c.

8.183.2 Field Documentation

8.183.2.1 char* setdomainname_args::domainname

Definition at line 284 of file kern_xxx.c.

Referenced by setdomainname(), and systrace_args().

8.183.2.2 int setdomainname_args::len

Definition at line 285 of file kern_xxx.c.

Referenced by setdomainname(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_xxx.c

8.184 setegid_args Struct Reference

Data Fields

- gid_t egid

8.184.1 Detailed Description

Definition at line 777 of file kern_prot.c.

8.184.2 Field Documentation

8.184.2.1 gid_t setegid_args::egid

Definition at line 778 of file kern_prot.c.

Referenced by setegid(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.185 seteuid_args Struct Reference

Data Fields

- uid_t euid

8.185.1 Detailed Description

Definition at line 615 of file kern_prot.c.

8.185.2 Field Documentation

8.185.2.1 uid_t seteuid_args::euid

Definition at line 616 of file kern_prot.c.

Referenced by seteuid(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.186 setgid_args Struct Reference

Data Fields

- gid_t gid

8.186.1 Detailed Description

Definition at line 674 of file kern_prot.c.

8.186.2 Field Documentation

8.186.2.1 gid_t setgid_args::gid

Definition at line 675 of file kern_prot.c.

Referenced by setgid(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.187 setgroups_args Struct Reference

Data Fields

- u_int gidsetsize
- gid_t * gidset

8.187.1 Detailed Description

Definition at line 828 of file kern_prot.c.

8.187.2 Field Documentation

8.187.2.1 gid_t* setgroups_args::gidset

Definition at line 830 of file kern_prot.c.

Referenced by setgroups(), and systrace_args().

8.187.2.2 u_int setgroups_args::gidsetsize

Definition at line 829 of file kern_prot.c.

Referenced by setgroups(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.188 setitimer_args Struct Reference

Data Fields

- u_int which
- itimerval * itv
- itimerval * oitv

8.188.1 Detailed Description

Definition at line 599 of file kern_time.c.

8.188.2 Field Documentation

8.188.2.1 struct itimerval* setitimer_args::itv

Definition at line 601 of file kern_time.c.

Referenced by setitimer(), and systrace_args().

8.188.2.2 struct itimerval * setitimer_args::oitv

Definition at line 601 of file kern_time.c.

Referenced by setitimer(), and systrace_args().

8.188.2.3 u_int setitimer_args::which

Definition at line 600 of file kern_time.c.

Referenced by setitimer(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_time.c

8.189 setlogin_args Struct Reference

Data Fields

- char * namebuf

8.189.1 Detailed Description

Definition at line 2015 of file kern_prot.c.

8.189.2 Field Documentation

8.189.2.1 char* setlogin_args::namebuf

Definition at line 2016 of file kern_prot.c.

Referenced by setlogin(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.190 setpgid_args Struct Reference

Data Fields

- int pid
- int pgid

8.190.1 Detailed Description

Definition at line 395 of file kern_prot.c.

8.190.2 Field Documentation

8.190.2.1 int setpgid_args::pgid

Definition at line 397 of file kern_prot.c.

Referenced by systrace_args().

8.190.2.2 int setpgid_args::pid

Definition at line 396 of file kern_prot.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.191 setpriority_args Struct Reference

Data Fields

- int which
- int who
- int prio

8.191.1 Detailed Description

Definition at line 168 of file kern_resource.c.

8.191.2 Field Documentation

8.191.2.1 int setpriority_args::prio

Definition at line 171 of file kern_resource.c.

Referenced by setpriority(), and systrace_args().

8.191.2.2 int setpriority_args::which

Definition at line 169 of file kern_resource.c.

Referenced by setpriority(), and systrace_args().

8.191.2.3 int setpriority_args::who

Definition at line 170 of file kern_resource.c.

Referenced by setpriority(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_resource.c

8.192 setregid_args Struct Reference

Data Fields

- gid_t rgid
- gid_t egid

8.192.1 Detailed Description

Definition at line 978 of file kern_prot.c.

8.192.2 Field Documentation

8.192.2.1 gid_t setregid_args::egid

Definition at line 980 of file kern_prot.c.

Referenced by setregid(), and systrace_args().

8.192.2.2 gid_t setregid_args::rgid

Definition at line 979 of file kern_prot.c.

Referenced by setregid(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.193 setresgid_args Struct Reference

Data Fields

- gid_t rgid
- gid_t egid
- gid_t sgid

8.193.1 Detailed Description

Definition at line 1133 of file kern_prot.c.

8.193.2 Field Documentation

8.193.2.1 gid_t setresgid_args::egid

Definition at line 1135 of file kern_prot.c.

Referenced by setresgid(), and systrace_args().

8.193.2.2 gid_t setresgid_args::rgid

Definition at line 1134 of file kern_prot.c.

Referenced by setresgid(), and systrace_args().

8.193.2.3 gid_t setresgid_args::sgid

Definition at line 1136 of file kern_prot.c.

Referenced by setresgid(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.194 setresuid_args Struct Reference

Data Fields

- uid_t ruid
- uid_t euid
- uid_t suid

8.194.1 Detailed Description

Definition at line 1048 of file kern_prot.c.

8.194.2 Field Documentation

8.194.2.1 uid_t setresuid_args::euid

Definition at line 1050 of file kern_prot.c.

Referenced by setresuid(), and systrace_args().

8.194.2.2 uid_t setresuid_args::ruid

Definition at line 1049 of file kern_prot.c.

Referenced by setresuid(), and systrace_args().

8.194.2.3 uid_t setresuid_args::suid

Definition at line 1051 of file kern_prot.c.

Referenced by setresuid(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.195 setreuid_args Struct Reference

Data Fields

- uid_t ruid
- uid_t euid

8.195.1 Detailed Description

Definition at line 906 of file kern_prot.c.

8.195.2 Field Documentation

8.195.2.1 uid_t setreuid_args::euid

Definition at line 908 of file kern_prot.c.

Referenced by setreuid(), and systrace_args().

8.195.2.2 uid_t setreuid_args::ruid

Definition at line 907 of file kern_prot.c.

Referenced by setreuid(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.196 setsid_args Struct Reference

Data Fields

- int dummy

8.196.1 Detailed Description

Definition at line 335 of file kern_prot.c.

8.196.2 Field Documentation

8.196.2.1 int setsid_args::dummy

Definition at line 336 of file kern_prot.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.197 settimeofday_args Struct Reference

Data Fields

- timeval * tv
- timezone * tzp

8.197.1 Detailed Description

Definition at line 463 of file kern_time.c.

8.197.2 Field Documentation

8.197.2.1 struct timeval* settimeofday_args::tv

Definition at line 464 of file kern_time.c.

Referenced by settimeofday(), and systrace_args().

8.197.2.2 struct timezone* settimeofday_args::tzp

Definition at line 465 of file kern_time.c.

Referenced by settimeofday(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_time.c

8.198 setuid_args Struct Reference

Data Fields

- uid_t uid

8.198.1 Detailed Description

Definition at line 499 of file kern_prot.c.

8.198.2 Field Documentation

8.198.2.1 uid_t setuid_args::uid

Definition at line 500 of file kern_prot.c.

Referenced by setuid(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_prot.c

8.199 shmat_args Struct Reference

Data Fields

- int shmid
- const void * shmaddr
- int shmflg

8.199.1 Detailed Description

Definition at line 340 of file sysv_shm.c.

8.199.2 Field Documentation

8.199.2.1 const void* shmat_args::shmaddr

Definition at line 342 of file sysv_shm.c.

Referenced by shmat(), and systrace_args().

8.199.2.2 int shmat_args::shmflg

Definition at line 343 of file sysv_shm.c.

Referenced by shmat(), and systrace_args().

8.199.2.3 int shmat_args::shmid

Definition at line 341 of file sysv_shm.c.

Referenced by shmat(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sysv_shm.c

8.200 shmctl_args Struct Reference

Data Fields

- int shmid
- int cmd
- shmid_ds * buf

8.200.1 Detailed Description

Definition at line 538 of file sysv_shm.c.

8.200.2 Field Documentation

8.200.2.1 struct shmid_ds* shmctl_args::buf

Definition at line 541 of file sysv_shm.c.

Referenced by shmctl(), and systrace_args().

8.200.2.2 int shmctl_args::cmd

Definition at line 540 of file sysv_shm.c.

Referenced by shmctl(), and systrace_args().

8.200.2.3 int shmctl_args::shmid

Definition at line 539 of file sysv_shm.c.

Referenced by shmctl(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sysv_shm.c

8.201 shmdt_args Struct Reference

Data Fields

- const void * shmaddr

8.201.1 Detailed Description

Definition at line 288 of file sysv_shm.c.

8.201.2 Field Documentation

8.201.2.1 const void* shmdt_args::shmaddr

Definition at line 289 of file sysv_shm.c.

Referenced by shmdt(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sysv_shm.c

8.202 shmget_args Struct Reference

Data Fields

- key_t key
- size_t size
- int shmflg

8.202.1 Detailed Description

Definition at line 688 of file sysv_shm.c.

8.202.2 Field Documentation

8.202.2.1 key_t shmget_args::key

Definition at line 689 of file sysv_shm.c.

Referenced by shmget(), shmget_allocate_segment(), and systrace_args().

8.202.2.2 int shmget_args::shmflg

Definition at line 691 of file sysv_shm.c.

Referenced by shmget(), shmget_existing(), and systrace_args().

8.202.2.3 size_t shmget_args::size

Definition at line 690 of file sysv_shm.c.

Referenced by shmget_allocate_segment(), shmget_existing(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sysv_shm.c

8.203 shmmap_state Struct Reference

Data Fields

- vm_offset_t va
- int shmid

8.203.1 Detailed Description

Definition at line 127 of file sysv_shm.c.

8.203.2 Field Documentation

8.203.2.1 int shmmap_state::shmid

Definition at line 129 of file sysv_shm.c.

Referenced by kern_shmat(), shm_delete_mapping(), shmdt(), shmexit_myhook(), and shmfork_myhook().

8.203.2.2 vm_offset_t shmmap_state::va

Definition at line 128 of file sysv_shm.c.

Referenced by shm_delete_mapping(), and shmdt().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sysv_shm.c

8.204 sigaction_args Struct Reference

Data Fields

- int sig
- sigaction * act
- sigaction * oact

8.204.1 Detailed Description

Definition at line 801 of file kern_sig.c.

8.204.2 Field Documentation

8.204.2.1 struct sigaction* sigaction_args::act

Definition at line 803 of file kern_sig.c.

Referenced by systrace_args().

8.204.2.2 struct sigaction* sigaction_args::oact

Definition at line 804 of file kern_sig.c.

Referenced by systrace_args().

8.204.2.3 int sigaction_args::sig

Definition at line 802 of file kern_sig.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_sig.c

8.205 sigaltstack_args Struct Reference

Data Fields

- stack_t * ss
- stack_t * oss

8.205.1 Detailed Description

Definition at line 1599 of file kern_sig.c.

8.205.2 Field Documentation

8.205.2.1 stack_t* sigaltstack_args::oss

Definition at line 1601 of file kern_sig.c.

Referenced by systrace_args().

8.205.2.2 stack_t* sigaltstack_args::ss

Definition at line 1600 of file kern_sig.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_sig.c

8.206 sigpending_args Struct Reference

Data Fields

- sigset_t * set

8.206.1 Detailed Description

Definition at line 1320 of file kern_sig.c.

8.206.2 Field Documentation

8.206.2.1 sigset_t* sigpending_args::set

Definition at line 1321 of file kern_sig.c.

Referenced by sigpending(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_sig.c

8.207 sigprocmask_args Struct Reference

Data Fields

- int how
- const sigset_t * set
- sigset_t * oset

8.207.1 Detailed Description

Definition at line 1043 of file kern_sig.c.

8.207.2 Field Documentation

8.207.2.1 int sigprocmask_args::how

Definition at line 1044 of file kern_sig.c.

Referenced by sigprocmask(), and systrace_args().

8.207.2.2 sigset_t* sigprocmask_args::oset

Definition at line 1046 of file kern_sig.c.

Referenced by sigprocmask(), and systrace_args().

8.207.2.3 const sigset_t* sigprocmask_args::set

Definition at line 1045 of file kern_sig.c.

Referenced by sigprocmask(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_sig.c

8.208 sigqueue_args Struct Reference

Data Fields

- pid_t pid
- int signum
- void * value

8.208.1 Detailed Description

Definition at line 1804 of file kern_sig.c.

8.208.2 Field Documentation

8.208.2.1 pid_t sigqueue_args::pid

Definition at line 1805 of file kern_sig.c.

Referenced by sigqueue(), and systrace_args().

8.208.2.2 int sigqueue_args::signum

Definition at line 1806 of file kern_sig.c.

Referenced by sigqueue(), and systrace_args().

8.208.2.3 void* sigqueue_args::value

Definition at line 1807 of file kern_sig.c.

Referenced by sigqueue(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_sig.c

8.209 sigsuspend_args Struct Reference

Data Fields

- const sigset_t * sigmask

8.209.1 Detailed Description

Definition at line 1475 of file kern_sig.c.

8.209.2 Field Documentation

8.209.2.1 const sigset_t* sigsuspend_args::sigmask

Definition at line 1476 of file kern_sig.c.

Referenced by sigsuspend(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_sig.c

8.210 sleepqueue Struct Reference

8.210.1 Detailed Description

Definition at line 116 of file subr_sleepqueue.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_sleepqueue.c

8.211 sleepqueue_chain Struct Reference

8.211.1 Detailed Description

Definition at line 127 of file subr_sleepqueue.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_sleepqueue.c

8.212 snprintf_arg Struct Reference

Data Fields

- char * str
- size_t remain

8.212.1 Detailed Description

Definition at line 89 of file subr_prf.c.

8.212.2 Field Documentation

8.212.2.1 size_t snprintf_arg::remain

Definition at line 91 of file subr_prf.c.

Referenced by snprintf_func(), vsnprintf(), and vsnrprintf().

8.212.2.2 char* snprintf_arg::str

Definition at line 90 of file subr_prf.c.

Referenced by snprintf_func(), vsnprintf(), and vsnrprintf().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_prf.c

8.213 sseg_closure Struct Reference

Data Fields

- int count
- size_t size

8.213.1 Detailed Description

Definition at line 881 of file imgact_elf.c.

8.213.2 Field Documentation

8.213.2.1 int sseg_closure::count

Definition at line 882 of file imgact_elf.c.

Referenced by cb_size_segment(), and coredump().

8.213.2.2 size_t sseg_closure::size

Definition at line 883 of file imgact_elf.c.

Referenced by cb_size_segment(), and coredump().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/imgact_elf.c

8.214 stat_args Struct Reference

Data Fields

- char * path
- stat * ub

8.214.1 Detailed Description

Definition at line 2052 of file vfs_syscalls.c.

8.214.2 Field Documentation

8.214.2.1 char* stat_args::path

Definition at line 2053 of file vfs_syscalls.c.

Referenced by systrace_args().

8.214.2.2 struct stat* stat_args::ub

Definition at line 2054 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.215 statfs_args Struct Reference

Data Fields

- char * path
- statfs * buf

8.215.1 Detailed Description

Definition at line 213 of file vfs_syscalls.c.

8.215.2 Field Documentation

8.215.2.1 struct statfs* statfs_args::buf

Definition at line 215 of file vfs_syscalls.c.

Referenced by systrace_args().

8.215.2.2 char* statfs_args::path

Definition at line 214 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.216 statspage Struct Reference

8.216.1 Detailed Description

Definition at line 442 of file subr_devstat.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_devstat.c

8.217 swapcontext_args Struct Reference

Data Fields

- __ucontext * oucp
- __ucontext_t * ucp

8.217.1 Detailed Description

Definition at line 55 of file kern_context.c.

8.217.2 Field Documentation

8.217.2.1 struct __ucontext* swapcontext_args::oucp

Definition at line 56 of file kern_context.c.

Referenced by swapcontext(), and systrace_args().

8.217.2.2 struct __ucontext_t* swapcontext_args::ucp

Definition at line 57 of file kern_context.c.

Referenced by swapcontext(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_context.c

8.218 symlink_args Struct Reference

Data Fields

- char * path
- char * link

8.218.1 Detailed Description

Definition at line 1484 of file vfs_syscalls.c.

8.218.2 Field Documentation

8.218.2.1 char* symlink_args::link

Definition at line 1486 of file vfs_syscalls.c.

Referenced by systrace_args().

8.218.2.2 char* symlink_args::path

Definition at line 1485 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.219 sync_args Struct Reference

Data Fields

- int dummy

8.219.1 Detailed Description

Definition at line 102 of file vfs_syscalls.c.

8.219.2 Field Documentation

8.219.2.1 int sync_args::dummy

Definition at line 103 of file vfs_syscalls.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.220 sysctl_args Struct Reference

Data Fields

- int * name
- u_int namelen
- void * old
- size_t * oldlenp
- void * new
- size_t newlen

8.220.1 Detailed Description

Definition at line 1288 of file kern_sysctl.c.

8.220.2 Field Documentation

8.220.2.1 int* sysctl_args::name

Definition at line 1289 of file kern_sysctl.c.

Referenced by __sysctl(), and systrace_args().

8.220.2.2 u_int sysctl_args::namelen

Definition at line 1290 of file kern_sysctl.c.

Referenced by __sysctl(), and systrace_args().

8.220.2.3 void* sysctl_args::new

Definition at line 1293 of file kern_sysctl.c.

Referenced by __sysctl(), and systrace_args().

8.220.2.4 size_t sysctl_args::newlen

Definition at line 1294 of file kern_sysctl.c.

Referenced by __sysctl(), and systrace_args().

8.220.2.5 void* sysctl_args::old

Definition at line 1291 of file kern_sysctl.c.

Referenced by __sysctl(), and systrace_args().

8.220.2.6 size_t* sysctl_args::oldlenp

Definition at line 1292 of file kern_sysctl.c.

Referenced by __sysctl(), and systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_sysctl.c

8.221 td_sched Struct Reference

Data Fields

- thread ∗ ts_thread

8.221.1 Detailed Description

Definition at line 81 of file sched_4bsd.c.

8.221.2 Field Documentation

8.221.2.1 struct thread∗ td_sched::ts_thread

Definition at line 187 of file sched_core.c.

Referenced by krunq_add(), sched_calc_pri(), sched_choose(), sched_recalc_pri(), sched_update_-runtime(), schedinit(), tdq_choose(), tdq_runq_add(), and tdq_runq_rem().

The documentation for this struct was generated from the following files:

- /usr/src/sys/kern/sched_4bsd.c
- /usr/src/sys/kern/sched_core.c

8.222 tdq Struct Reference

Data Fields

- runq tdq_idle
- runq tdq_timeshare
- runq tdq_realtime
- u_char tdq_idx
- u_char tdq_ridx
- short tdq_flags
- int tdq_load
- int tdq_sysload

8.222.1 Detailed Description

Definition at line 177 of file sched_ule.c.

8.222.2 Field Documentation

8.222.2.1 short tdq::tdq_flags

Definition at line 183 of file sched_ule.c.

Referenced by tdq_runq_add(), and tdq_runq_rem().

8.222.2.2 struct runq tdq::tdq_idle

Definition at line 178 of file sched_ule.c.

Referenced by sched_add(), tdq_choose(), tdq_print(), and tdq_setup().

8.222.2.3 u_char tdq::tdq_idx

Definition at line 181 of file sched_ule.c.

Referenced by sched_clock(), tdq_print(), tdq_runq_add(), and tdq_runq_rem().

8.222.2.4 int tdq::tdq_load

Definition at line 184 of file sched_ule.c.

Referenced by sched_runnable(), tdq_load_add(), tdq_load_rem(), tdq_print(), and tdq_setup().

8.222.2.5 struct runq tdq::tdq_realtime

Definition at line 180 of file sched_ule.c.

Referenced by sched_add(), tdq_choose(), tdq_print(), and tdq_setup().

8.222.2.6 u_char tdq::tdq_ridx

Definition at line 182 of file sched_ule.c.

Referenced by sched_clock(), tdq_choose(), tdq_print(), tdq_runq_add(), and tdq_runq_rem().

8.222.2.7 int tdq::tdq_sysload

Definition at line 190 of file sched_ule.c.

Referenced by tdq_load_add(), and tdq_load_rem().

8.222.2.8 struct runq tdq::tdq_timeshare

Definition at line 179 of file sched_ule.c.

Referenced by sched_add(), sched_clock(), tdq_choose(), tdq_print(), tdq_runq_add(), tdq_runq_rem(), and tdq_setup().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sched_ule.c

8.223 timehands Struct Reference

Collaboration diagram for timehands:

Data Fields

- timecounter * th_counter
- int64_t th_adjustment
- u_int64_t th_scale
- u_int th_offset_count
- bintime th_offset
- timeval th_microtime
- timespec th_nanotime
- volatile u_int th_generation
- timehands * th_next

8.223.1 Detailed Description

Definition at line 50 of file kern_tc.c.

8.223.2 Field Documentation

8.223.2.1 int64_t timehands::th_adjustment

Definition at line 53 of file kern_tc.c.

Referenced by tc_windup().

8.223.2.2 struct timecounter* timehands::th_counter

Definition at line 52 of file kern_tc.c.

Referenced by pps_capture(), tc_cpu_ticks(), tc_delta(), tc_getfrequency(), and tc_windup().

8.223.2.3 volatile u_int timehands::th_generation

Definition at line 60 of file kern_tc.c.

Referenced by binuptime(), getbintime(), getbinuptime(), getmicrotime(), getmicrouptime(), getnanotime(), getnanouptime(), pps_capture(), and tc_windup().

8.223.2.4 struct timeval timehands::th_microtime

Definition at line 57 of file kern_tc.c.

Referenced by getmicrotime(), and tc_windup().

8.223.2.5 struct timespec timehands::th_nanotime

Definition at line 58 of file kern_tc.c.

Referenced by getnanotime(), and tc_windup().

8.223.2.6 struct timehands* timehands::th_next

Definition at line 61 of file kern_tc.c.

Referenced by tc_windup().

8.223.2.7 struct bintime timehands::th_offset

Definition at line 56 of file kern_tc.c.

Referenced by binuptime(), getbintime(), getbinuptime(), getmicrouptime(), getnanouptime(), and tc_windup().

8.223.2.8 u_int timehands::th_offset_count

Definition at line 55 of file kern_tc.c.

Referenced by tc_delta(), and tc_windup().

8.223.2.9 u_int64_t timehands::th_scale

Definition at line 54 of file kern_tc.c.

Referenced by binuptime(), and tc_windup().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_tc.c

8.224 timer_getoverrun_args Struct Reference

Data Fields

- int timerid

8.224.1 Detailed Description

Definition at line 1231 of file kern_time.c.

8.224.2 Field Documentation

8.224.2.1 int timer_getoverrun_args::timerid

Definition at line 1232 of file kern_time.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_time.c

8.225 truncate_args Struct Reference

Data Fields

- char * path
- int pad
- off_t length

8.225.1 Detailed Description

Definition at line 3001 of file vfs_syscalls.c.

8.225.2 Field Documentation

8.225.2.1 off_t truncate_args::length

Definition at line 3004 of file vfs_syscalls.c.

Referenced by systrace_args().

8.225.2.2 int truncate_args::pad

Definition at line 3003 of file vfs_syscalls.c.

Referenced by systrace_args().

8.225.2.3 char* truncate_args::path

Definition at line 3002 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.226 turnstile Struct Reference

Data Fields

- threadqueue ts_blocked [2]
- threadqueue ts_pending

8.226.1 Detailed Description

Definition at line 117 of file subr_turnstile.c.

8.226.2 Field Documentation

8.226.2.1 struct threadqueue turnstile::ts_blocked[2]

Definition at line 118 of file subr_turnstile.c.

Referenced by turnstile_adjust(), turnstile_adjust_thread(), turnstile_alloc(), turnstile_broadcast(), turnstile_disown(), turnstile_empty(), turnstile_first_waiter(), turnstile_free(), turnstile_head(), turnstile_-signal(), turnstile_unpend(), and turnstile_wait().

8.226.2.2 struct threadqueue turnstile::ts_pending

Definition at line 119 of file subr_turnstile.c.

Referenced by turnstile_alloc(), turnstile_broadcast(), turnstile_disown(), turnstile_free(), turnstile_-signal(), turnstile_unpend(), and turnstile_wait().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_turnstile.c

8.227 turnstile_chain Struct Reference

8.227.1 Detailed Description

Definition at line 127 of file subr_turnstile.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_turnstile.c

8.228 umask_args Struct Reference

Data Fields

- int newmask

8.228.1 Detailed Description

Definition at line 3807 of file vfs_syscalls.c.

8.228.2 Field Documentation

8.228.2.1 int umask_args::newmask

Definition at line 3808 of file vfs_syscalls.c.

Referenced by systrace_args(), and umask().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.229 umtx_key Struct Reference

Data Fields

- int hash
- int type
- int shared
- union {
  struct {
  vm_object_t object
  uintptr_t offset
  } shared
  struct {
  vmspace * vs
  uintptr_t addr
  } private
  struct {
  void * a
  uintptr_t b
  } both
  } info

8.229.1 Detailed Description

Definition at line 69 of file kern_umtx.c.

8.229.2 Field Documentation

8.229.2.1 void* umtx_key::a

Definition at line 83 of file kern_umtx.c.

8.229.2.2 uintptr_t umtx_key::addr

Definition at line 80 of file kern_umtx.c.

8.229.2.3 uintptr_t umtx_key::b

Definition at line 84 of file kern_umtx.c.

8.229.2.4 struct { ... } umtx_key::both

Referenced by umtx_key_match(), and umtxq_hash().

8.229.2.5 int umtx_key::hash

Definition at line 70 of file kern_umtx.c.

Referenced by umtxq_getchain(), and umtxq_hash().

8.229.2.6 union { ... } umtx_key::info

Referenced by umtx_key_get(), umtx_key_match(), umtx_key_release(), and umtxq_hash().

8.229.2.7 vm_object_t umtx_key::object

Definition at line 75 of file kern_umtx.c.

8.229.2.8 uintptr_t umtx_key::offset

Definition at line 76 of file kern_umtx.c.

8.229.2.9 struct { ... } umtx_key::private

Referenced by umtx_key_get().

8.229.2.10 struct { ... } umtx_key::shared

8.229.2.11 int umtx_key::shared

Definition at line 72 of file kern_umtx.c.

Referenced by umtx_key_get(), and umtx_key_release().

8.229.2.12 int umtx_key::type

Definition at line 71 of file kern_umtx.c.

Referenced by umtx_key_get(), and umtx_key_match().

8.229.2.13 struct vmspace* umtx_key::vs

Definition at line 79 of file kern_umtx.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_umtx.c

8.230 umtx_pi Struct Reference

Data Fields

- thread ∗ pi_owner
- int pi_refcount

8.230.1 Detailed Description

Definition at line 90 of file kern_umtx.c.

8.230.2 Field Documentation

8.230.2.1 struct thread∗ umtx_pi::pi_owner

Definition at line 92 of file kern_umtx.c.

Referenced by do_unlock_pi(), umtx_pi_claim(), umtx_pi_setowner(), umtx_pi_unref(), umtx_propagate_priority(), umtx_thread_cleanup(), umtx_unpropagate_priority(), and umtxq_sleep_pi().

8.230.2.2 int umtx_pi::pi_refcount

Definition at line 95 of file kern_umtx.c.

Referenced by umtx_pi_ref(), and umtx_pi_unref().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_umtx.c

8.231 umtx_q Struct Reference

8.231.1 Detailed Description

Definition at line 111 of file kern_umtx.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_umtx.c

8.232 umtxq_chain Struct Reference

Data Fields

- mtx uc_lock
- umtxq_head uc_queue
- char uc_busy
- int uc_waiters

8.232.1 Detailed Description

Definition at line 145 of file kern_umtx.c.

8.232.2 Field Documentation

8.232.2.1 char umtxq_chain::uc_busy

Definition at line 153 of file kern_umtx.c.

Referenced by umtxq_sysinit().

8.232.2.2 struct mtx umtxq_chain::uc_lock

Definition at line 147 of file kern_umtx.c.

8.232.2.3 struct umtxq_head umtxq_chain::uc_queue

Definition at line 150 of file kern_umtx.c.

8.232.2.4 int umtxq_chain::uc_waiters

Definition at line 156 of file kern_umtx.c.

Referenced by umtxq_sysinit().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_umtx.c

8.233 uname_args Struct Reference

Data Fields

- utsname * name

8.233.1 Detailed Description

Definition at line 173 of file kern_xxx.c.

8.233.2 Field Documentation

8.233.2.1 struct utsname* uname_args::name

Definition at line 174 of file kern_xxx.c.

Referenced by systrace_args(), and uname().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_xxx.c

8.234 unlink_args Struct Reference

Data Fields

- char * path

8.234.1 Detailed Description

Definition at line 1629 of file vfs_syscalls.c.

8.234.2 Field Documentation

8.234.2.1 char* unlink_args::path

Definition at line 1630 of file vfs_syscalls.c.

Referenced by systrace_args(), and unlink().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.235 unmount_args Struct Reference

Data Fields

- char * path
- int flags

8.235.1 Detailed Description

Definition at line 1082 of file vfs_mount.c.

8.235.2 Field Documentation

8.235.2.1 int unmount_args::flags

Definition at line 1084 of file vfs_mount.c.

Referenced by systrace_args().

8.235.2.2 char* unmount_args::path

Definition at line 1083 of file vfs_mount.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_mount.c

8.236 unr Struct Reference

8.236.1 Detailed Description

Definition at line 173 of file subr_unit.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_unit.c

8.237 unrb Struct Reference

Data Fields

- u_char busy
- bitstr_t map [sizeof(struct unr)-1]

8.237.1 Detailed Description

Definition at line 179 of file subr_unit.c.

8.237.2 Field Documentation

8.237.2.1 u_char unrb::busy

Definition at line 180 of file subr_unit.c.

Referenced by alloc_unrl(), collapse_unr(), free_unrl(), and optimize_unr().

8.237.2.2 bitstr_t unrb::map[sizeof(struct unr)-1]

Definition at line 181 of file subr_unit.c.

Referenced by alloc_unrl(), free_unrl(), and optimize_unr().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_unit.c

8.238 unrhdr Struct Reference

8.238.1 Detailed Description

Definition at line 191 of file subr_unit.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_unit.c

8.239 utimes_args Struct Reference

Data Fields

- char * path
- timeval * tptr

8.239.1 Detailed Description

Definition at line 2866 of file vfs_syscalls.c.

8.239.2 Field Documentation

8.239.2.1 char* utimes_args::path

Definition at line 2867 of file vfs_syscalls.c.

Referenced by systrace_args().

8.239.2.2 struct timeval* utimes_args::tptr

Definition at line 2868 of file vfs_syscalls.c.

Referenced by systrace_args().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_syscalls.c

8.240 uuid_private Struct Reference

Data Fields

- union {
  uint64_t ll
  struct {
  uint32_t low
  uint16_t mid
  uint16_t hi
  } x
  } time

- uint16_t seq
- uint16_t node [UUID_NODE_LEN >>1]

8.240.1 Detailed Description

Definition at line 57 of file kern_uuid.c.

8.240.2 Field Documentation

8.240.2.1 uint16_t uuid_private::hi

Definition at line 63 of file kern_uuid.c.

8.240.2.2 uint64_t uuid_private::ll

Definition at line 59 of file kern_uuid.c.

Referenced by kern_uuidgen().

8.240.2.3 uint32_t uuid_private::low

Definition at line 61 of file kern_uuid.c.

8.240.2.4 uint16_t uuid_private::mid

Definition at line 62 of file kern_uuid.c.

8.240.2.5 uint16_t uuid_private::node[UUID_NODE_LEN >>1]

Definition at line 67 of file kern_uuid.c.

Referenced by kern_uuidgen(), and snprintf_uuid().

8.240.2.6 uint16_t uuid_private::seq

Definition at line 66 of file kern_uuid.c.

Referenced by kern_uuidgen(), and snprintf_uuid().

8.240.2.7 union { ... } uuid_private::time

Referenced by kern_uuidgen(), and snprintf_uuid().

8.240.2.8 struct { ... } uuid_private::x

Referenced by kern_uuidgen(), and snprintf_uuid().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_uuid.c

8.241 uuidgen_args Struct Reference

Data Fields

- uuid * store
- int count

8.241.1 Detailed Description

Definition at line 176 of file kern_uuid.c.

8.241.2 Field Documentation

8.241.2.1 int uuidgen_args::count

Definition at line 178 of file kern_uuid.c.

Referenced by systrace_args(), and uuidgen().

8.241.2.2 struct uuid* uuidgen_args::store

Definition at line 177 of file kern_uuid.c.

Referenced by systrace_args(), and uuidgen().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/kern_uuid.c

8.242 vfsopt Struct Reference

8.242.1 Detailed Description

Definition at line 109 of file vfs_mount.c.

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/vfs_mount.c

8.243 witness Struct Reference

Data Fields

- const char * w_name
- lock_class * w_class

8.243.1 Detailed Description

Definition at line 134 of file subr_witness.c.

8.243.2 Field Documentation

8.243.2.1 struct lock_class* witness::w_class

Definition at line 136 of file subr_witness.c.

Referenced by depart(), enroll(), and itismychild().

8.243.2.2 const char* witness::w_name

Definition at line 135 of file subr_witness.c.

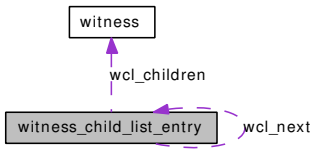
Referenced by enroll().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_witness.c

8.244 witness_child_list_entry Struct Reference

Collaboration diagram for witness_child_list_entry:



Data Fields

- witness_child_list_entry * wcl_next
- witness * wcl_children [WITNESS_NCHILDREN]
- u_int wcl_count

8.244.1 Detailed Description

Definition at line 150 of file subr_witness.c.

8.244.2 Field Documentation

8.244.2.1 struct witness* witness_child_list_entry::wcl_children[WITNESS_NCHILDREN]

Definition at line 152 of file subr_witness.c.

Referenced by isitmychild(), and isitmydescendant().

8.244.2.2 u_int witness_child_list_entry::wcl_count

Definition at line 153 of file subr_witness.c.

Referenced by insertchild(), isitmychild(), isitmydescendant(), and removechild().

8.244.2.3 struct witness_child_list_entry* witness_child_list_entry::wcl_next

Definition at line 151 of file subr_witness.c.

Referenced by depart(), insertchild(), isitmychild(), isitmydescendant(), removechild(), witness_child_-free(), and witness_child_get().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_witness.c

8.245 witness_order_list_entry Struct Reference

Data Fields

- const char * w_name
- lock_class * w_class

8.245.1 Detailed Description

Definition at line 165 of file subr_witness.c.

8.245.2 Field Documentation

8.245.2.1 struct lock_class* witness_order_list_entry::w_class

Definition at line 167 of file subr_witness.c.

Referenced by STAILQ_HEAD().

8.245.2.2 const char* witness_order_list_entry::w_name

Definition at line 166 of file subr_witness.c.

Referenced by STAILQ_HEAD().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/subr_witness.c

8.246 write_args Struct Reference

Data Fields

- int fd
- const void * buf
- size_t nbyte

8.246.1 Detailed Description

Definition at line 300 of file sys_generic.c.

8.246.2 Field Documentation

8.246.2.1 const void* write_args::buf

Definition at line 302 of file sys_generic.c.

Referenced by systrace_args(), and write().

8.246.2.2 int write_args::fd

Definition at line 301 of file sys_generic.c.

Referenced by systrace_args(), and write().

8.246.2.3 size_t write_args::nbyte

Definition at line 303 of file sys_generic.c.

Referenced by systrace_args(), and write().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sys_generic.c

8.247 writev_args Struct Reference

Data Fields

- int fd
- iovec * iovp
- u_int iovcnt

8.247.1 Detailed Description

Definition at line 370 of file sys_generic.c.

8.247.2 Field Documentation

8.247.2.1 int writev_args::fd

Definition at line 371 of file sys_generic.c.

Referenced by systrace_args(), and writev().

8.247.2.2 u_int writev_args::iovcnt

Definition at line 373 of file sys_generic.c.

Referenced by systrace_args(), and writev().

8.247.2.3 struct iovec* writev_args::iovp

Definition at line 372 of file sys_generic.c.

Referenced by systrace_args(), and writev().

The documentation for this struct was generated from the following file:

- /usr/src/sys/kern/sys_generic.c

Chapter 9

FreeBSD kernel kern code File Documentation

9.1 notreviewed.dox File Reference

9.2 /usr/src/sys/kern/bus_if.m File Reference

```
#include <sys/bus.h>
```

Include dependency graph for bus_if.m:

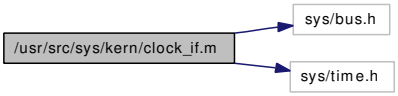
Variables

- INTERFACE bus

9.3 /usr/src/sys/kern/clock_if.m File Reference

```
#include <sys/bus.h>
#include <sys/time.h>
```

Include dependency graph for clock_if.m:



Variables

- INTERFACE clock

9.3.1 Variable Documentation

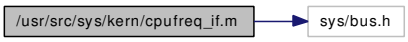
9.3.1.1 INTERFACE clock

Definition at line 30 of file clock_if.m.

9.4 /usr/src/sys/kern/cpufreq_if.m File Reference

```
#include <sys/bus.h>
```

Include dependency graph for cpufreq_if.m:



Variables

- INTERFACE cpufreq

9.4.1 Variable Documentation

9.4.1.1 INTERFACE cpufreq

Definition at line 31 of file cpufreq_if.m.

9.5 /usr/src/sys/kern/device_if.m File Reference

```
#include <sys/bus.h>
```

Include dependency graph for device_if.m:

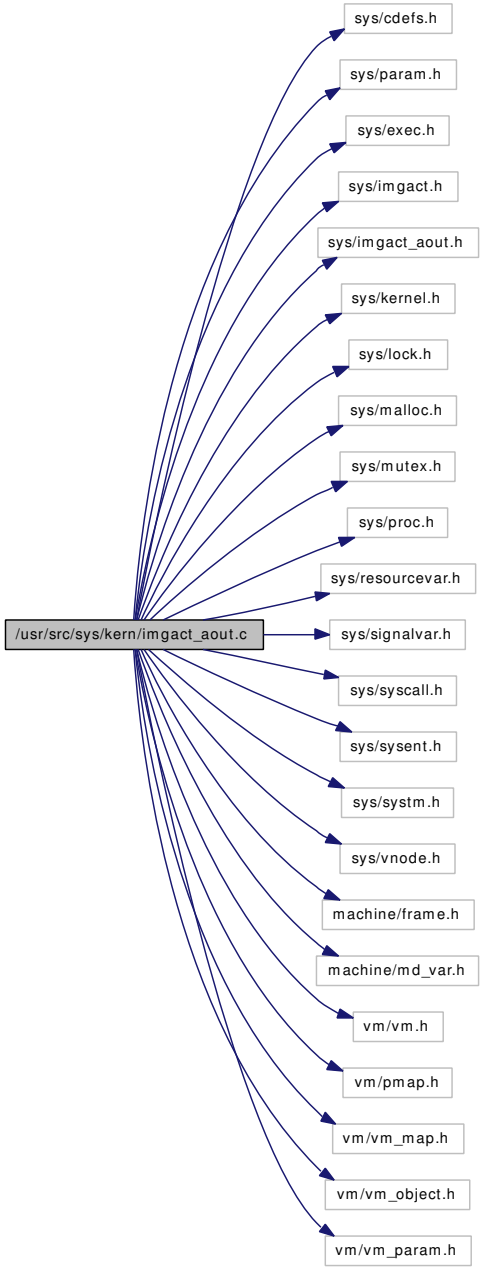
Variables

- INTERFACE device

9.6 /usr/src/sys/kern/imgact_aout.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/exec.h>
#include <sys/imgact.h>
#include <sys/imgact_aout.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/resourcevar.h>
#include <sys/signalvar.h>
#include <sys/syscall.h>
#include <sys/sysent.h>
#include <sys/systm.h>
#include <sys/vnode.h>
#include <machine/frame.h>
#include <machine/md_var.h>
#include <vm/vm.h>
#include <vm/pmap.h>
#include <vm/vm_map.h>
#include <vm/vm_object.h>
#include <vm/vm_param.h>
```

Include dependency graph for imgact_aout.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/imgact_aout.c,v 1.101 2006/03/16 08:51:59 alc Exp $")
- static int exec_aout_imgact (struct image_params *imgp)
- static int aout_fixup (register_t **stack_base, struct image_params *imgp)
- EXEC_SET (aout, aout_execsw)

Variables

- sysentvec aout_sysvec

- static struct execsw aout_execsw = { exec_aout_imgact, "a.out" }

9.6.1 Function Documentation

9.6.1.1 __FBSDID ("$FreeBSD: src/sys/kern/imgact_aout. *c*, v 1.101 2006/03/16 08:51:59 alc Exp $")

9.6.1.2 static int aout_fixup (register_t ** *stack_base*, struct image_params * *imgp*) [static]

Definition at line 88 of file imgact_aout.c.

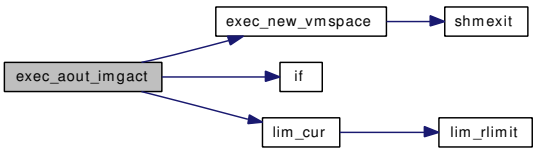
References suword.

9.6.1.3 static int exec_aout_imgact (struct image_params * *imgp*) [static]

Definition at line 97 of file imgact_aout.c.

References aout_sysvec, exec_new_vmspace(), if(), lim_cur(), maxtsiz, and td.

Here is the call graph for this function:



9.6.1.4 EXEC_SET (aout, aout_execsw)

9.6.2 Variable Documentation

9.6.2.1 struct execsw aout_execsw = { exec_aout_imgact, "a.out" } [static]

Definition at line 271 of file imgact_aout.c.

9.6.2.2 struct sysentvec aout_sysvec

Initial value:

```
{
        SYS_MAXSYSCALL,
        sysent,
        0,
        0,
        NULL,
        0,
        NULL,
        NULL,
        aout_fixup,
        sendsig,
        sigcode,
        &szsigcode,
        NULL,
```

```
        "FreeBSD a.out",
        NULL,
        NULL,
        MINSIGSTKSZ,
        PAGE_SIZE,
        VM_MIN_ADDRESS,
        VM_MAXUSER_ADDRESS,
        USRSTACK,
        PS_STRINGS,
        VM_PROT_ALL,
        exec_copyout_strings,
        exec_setregs,
        NULL
}
```

Definition at line 58 of file imgact_aout.c.

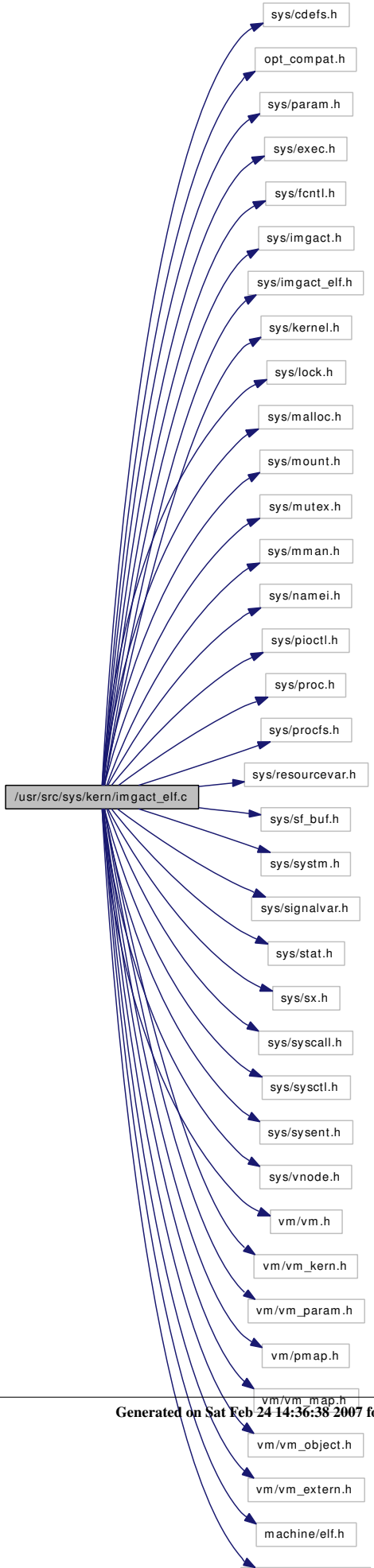
Referenced by do_aout_hdr(), and exec_aout_imgact().

9.7 /usr/src/sys/kern/imgact_elf.c File Reference

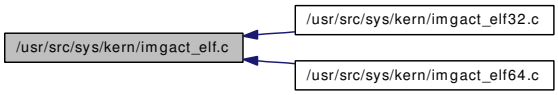
```
#include <sys/cdefs.h>
#include "opt_compat.h"
#include <sys/param.h>
#include <sys/exec.h>
#include <sys/fcntl.h>
#include <sys/imgact.h>
#include <sys/imgact_elf.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mount.h>
#include <sys/mutex.h>
#include <sys/mman.h>
#include <sys/namei.h>
#include <sys/pioctl.h>
#include <sys/proc.h>
#include <sys/procfs.h>
#include <sys/resourcevar.h>
#include <sys/sf_buf.h>
#include <sys/systm.h>
#include <sys/signalvar.h>
#include <sys/stat.h>
#include <sys/sx.h>
#include <sys/syscall.h>
#include <sys/sysctl.h>
#include <sys/sysent.h>
#include <sys/vnode.h>
#include <vm/vm.h>
#include <vm/vm_kern.h>
#include <vm/vm_param.h>
#include <vm/pmap.h>
#include <vm/vm_map.h>
#include <vm/vm_object.h>
#include <vm/vm_extern.h>
#include <machine/elf.h>
```

```
#include <machine/md_var.h>
```

Include dependency graph for imgact_elf.c:


/usr/src/sys/kern/imgact_elf.c
sys/cdefs.h
opt_compat.h
sys/param.h
sys/exec.h
sys/fcntl.h
sys/imgact.h
sys/imgact_elf.h
sys/kernel.h
sys/lock.h
sys/malloc.h
sys/mount.h
sys/mutex.h
sys/mman.h
sys/namei.h
sys/pioctl.h
sys/proc.h
sys/procfs.h
sys/resourcevar.h
sys/sf_buf.h
sys/systm.h
sys/signalvar.h
sys/stat.h
sys/sx.h
sys/syscall.h
sys/sysctl.h
sys/sysent.h
sys/vnode.h
vm/vm.h
vm/vm_kern.h
vm/vm_param.h
vm/pmap.h
vm/vm_map.h
vm/vm_object.h
vm/vm_extern.h
machine/elf.h

This graph shows which files directly or indirectly include this file:



Data Structures

- struct phdr_closure
- struct sseg_closure

Defines

- #define OLD_EI_BRAND 8
- #define trunc_page_ps(va, ps) ((va) & ~(ps - 1))
- #define round_page_ps(va, ps) (((va) + (ps - 1)) & ~(ps - 1))
- #define suword __CONCAT(suword, __ELF_WORD_SIZE)

Typedefs

- typedef void(*) segment_callback (vm_map_entry_t, void *)
- typedef prstatus_t elf_prstatus_t
- typedef prpsinfo_t elf_prpsinfo_t
- typedef prfpregset_t elf_prfpregset_t
- typedef prfpregset_t elf_fpregset_t
- typedef gregset_t elf_gregset_t

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/imgact_elf.c,v 1.177 2007/01/17 14:58:53 delphij Exp $")
- static int __elfN() check_header (const Elf_Ehdr *hdr)
- static Elf_Brandinfo *__elfN() get_brandinfo (const Elf_Ehdr *hdr, const char *interp)
- static int __elfN() load_file (struct proc *p, const char *file, u_long *addr, u_long *entry, size_t pagesize)
- static int __elfN() load_section (struct vmspace *vmspace, vm_object_t object, vm_offset_t offset, caddr_t vmaddr, size_t memsz, size_t filsz, vm_prot_t prot, size_t pagesize)
- static int __CONCAT (exec_, __elfN(imgact))
- int __elfN() remove_brand_entry (Elf_Brandinfo *entry)
- int __elfN() brand_inuse (Elf_Brandinfo *entry)
- static int __elfN() map_partial (vm_map_t map, vm_object_t object, vm_ooffset_t offset, vm_offset_t start, vm_offset_t end, vm_prot_t prot)
- static int __elfN() map_insert (vm_map_t map, vm_object_t object, vm_ooffset_t offset, vm_offset_t start, vm_offset_t end, vm_prot_t prot, int cow)
- int __elfN() freebsd_fixup (register_t **stack_base, struct image_params *imgp)
- static void cb_put_phdr (vm_map_entry_t, void *)
- static void cb_size_segment (vm_map_entry_t, void *)
- static void each_writable_segment (struct thread *, segment_callback, void *)
- static int __elfN() corehdr (struct thread *, struct vnode *, struct ucred *, int, void *, size_t)

- static void __elfN() puthdr (struct thread *, void *, size_t *, int)
- static void __elfN() putnote (void *, size_t *, const char *, int, const void *, size_t)
- int __elfN() coredump (struct thread *td, struct vnode *vp, off_t limit)
- static struct execsw __elfN (execsw)
- EXEC_SET (__CONCAT(elf, __ELF_WORD_SIZE), __elfN(execsw))

Variables

- int osreldate

9.7.1 Define Documentation

9.7.1.1 #define OLD_EI_BRAND 8

Definition at line 78 of file imgact_elf.c.

9.7.1.2 #define round_page_ps(va, ps) (((va) + (ps - 1)) & ~(ps - 1))

9.7.1.3 #define suword __CONCAT(suword, __ELF_WORD_SIZE)

Definition at line 832 of file imgact_elf.c.

Referenced by aio_aqueue(), aio_return(), aio_waitcomplete(), aout_fixup(), exec_copyout_strings(), freebsd_fixup(), kse_switchin(), and start_init().

9.7.1.4 #define trunc_page_ps(va, ps) ((va) & ~(ps - 1))

9.7.2 Typedef Documentation

9.7.2.1 typedef prfpregset_t elf_fpregset_t

Definition at line 1109 of file imgact_elf.c.

9.7.2.2 typedef gregset_t elf_gregset_t

Definition at line 1110 of file imgact_elf.c.

9.7.2.3 typedef prfpregset_t elf_prfpregset_t

Definition at line 1108 of file imgact_elf.c.

9.7.2.4 typedef prpsinfo_t elf_prpsinfo_t

Definition at line 1107 of file imgact_elf.c.

9.7.2.5 typedef prstatus_t elf_prstatus_t

Definition at line 1106 of file imgact_elf.c.

9.7.2.6 typedef void(∗) segment_callback(vm_map_entry_t, void ∗)

Definition at line 872 of file imgact_elf.c.

9.7.3 Function Documentation

9.7.3.1 static int __CONCAT (exec_, __elfN(imgact)) [static]

Definition at line 88 of file imgact_elf.c.

9.7.3.2 static struct execsw __elfN (execsw) [static]

Referenced by coredump(), and map_insert().

9.7.3.3 __FBSDID ("$FreeBSD: src/sys/kern/imgact_elf. *c*, v 1.177 2007/01/17 14:58:53 delphij Exp $")

9.7.3.4 int __elfN() brand_inuse (Elf_Brandinfo ∗ *entry*)

Definition at line 142 of file imgact_elf.c.

References allproc_lock.

9.7.3.5 static void cb_put_phdr (vm_map_entry_t, void ∗) [static]

Definition at line 965 of file imgact_elf.c.

References phdr_closure::offset, and phdr_closure::phdr.

9.7.3.6 static void cb_size_segment (vm_map_entry_t, void ∗) [static]

Definition at line 997 of file imgact_elf.c.

References sseg_closure::count, and sseg_closure::size.

Referenced by coredump().

9.7.3.7 static int __elfN() check_header (const Elf_Ehdr ∗ *hdr*) [static]

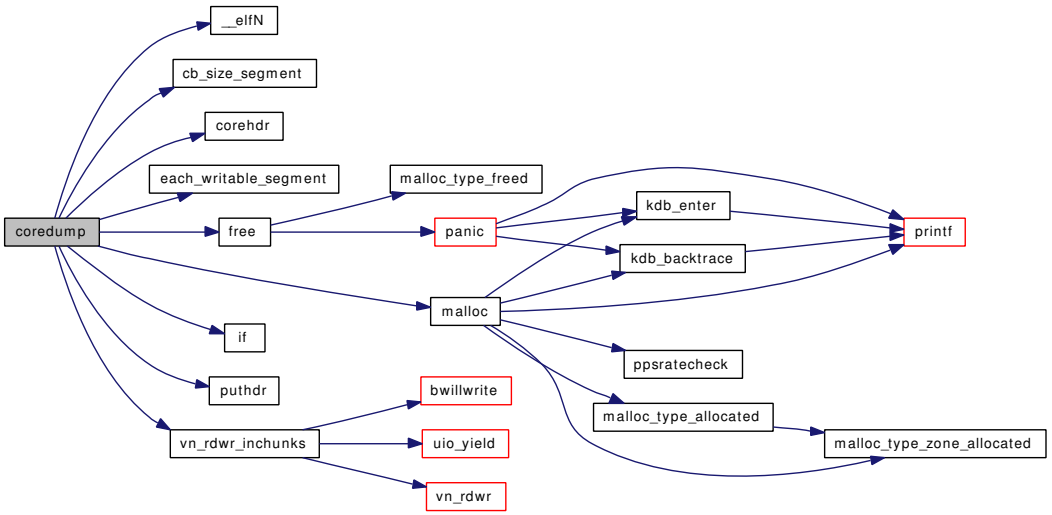
9.7.3.8 int __elfN() coredump (struct thread ∗ *td*, struct vnode ∗ *vp*, off_t *limit*)

Definition at line 898 of file imgact_elf.c.

References __elfN(), cb_size_segment(), corehdr(), sseg_closure::count, each_writable_segment(), free(), if(), malloc(), puthdr(), sseg_closure::size, and vn_rdwr_inchunks().

Referenced by sigexit().

Here is the call graph for this function:



9.7.3.9 static int __elfN() corehdr (struct thread *, struct vnode *, struct ucred *, int, void *, size_t) [static]

Referenced by coredump().

9.7.3.10 static void each_writable_segment (struct thread *, segment_callback, void *) [static]

Definition at line 1013 of file imgact_elf.c.

Referenced by coredump().

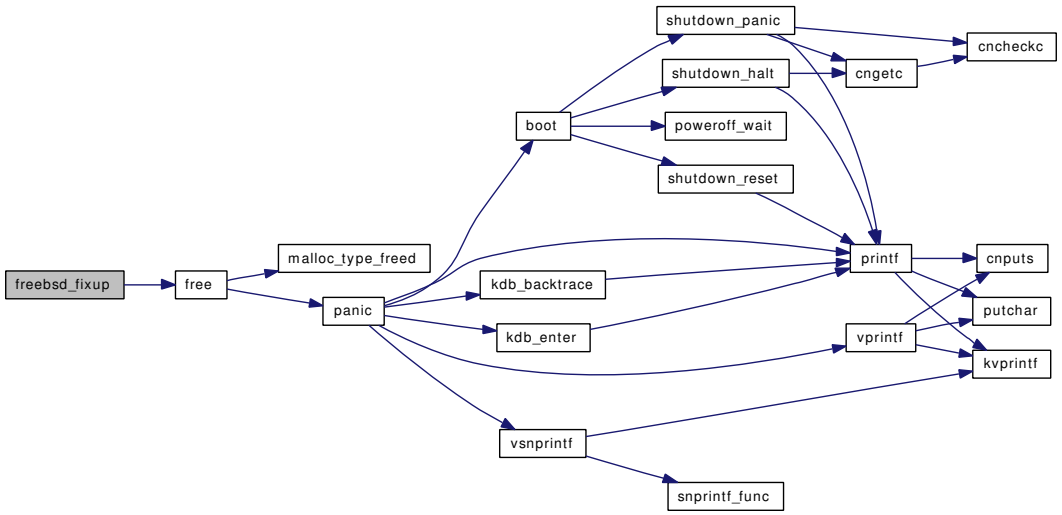
9.7.3.11 EXEC_SET (__CONCAT(elf, __ELF_WORD_SIZE), __elfN(execsw))

9.7.3.12 int __elfN() freebsd_fixup (register_t ** *stack_base*, struct image_params * *imgp*)

Definition at line 835 of file imgact_elf.c.

References free(), and suword.

Here is the call graph for this function:



9.7.3.13 static Elf_Brandinfo* __elfN() get_brandinfo (const Elf_Ehdr * *hdr*, const char * *interp*) [static]

9.7.3.14 static int __elfN() load_file (struct proc * *p*, const char * *file*, u_long * *addr*, u_long * *entry*, size_t *pagesize*) [static]

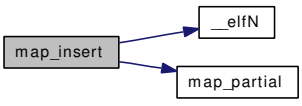
9.7.3.15 static int __elfN() load_section (struct vmspace * *vmspace*, vm_object_t *object*, vm_offset_t *offset*, caddr_t *vmaddr*, size_t *memsz*, size_t *filsz*, vm_prot_t *prot*, size_t *pagesize*) [static]

9.7.3.16 static int __elfN() map_insert (vm_map_t *map*, vm_object_t *object*, vm_ooffset_t *offset*, vm_offset_t *start*, vm_offset_t *end*, vm_prot_t *prot*, int *cow*) [static]

Definition at line 268 of file imgact_elf.c.

References __elfN(), and map_partial().

Here is the call graph for this function:



9.7.3.17 static int __elfN() map_partial (vm_map_t *map*, vm_object_t *object*, vm_ooffset_t *offset*, vm_offset_t *start*, vm_offset_t *end*, vm_prot_t *prot*) [static]

Definition at line 233 of file imgact_elf.c.

Referenced by map_insert().

9.7.3.18 static void __elfN() puthdr (struct thread *, void *, size_t *, int) [static]

Referenced by coredump().

9.7.3.19 static void __elfN() putnote (void *, size_t *, const char *, int, const void *, size_t) [static]

9.7.3.20 int __elfN() remove_brand_entry (Elf_Brandinfo * *entry*)

Definition at line 126 of file imgact_elf.c.

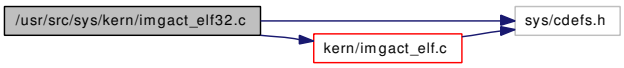
9.7.4 Variable Documentation

9.7.4.1 int osreldate

9.8 /usr/src/sys/kern/imgact_elf32.c File Reference

```
#include <sys/cdefs.h>
#include <kern/imgact_elf.c>
```

Include dependency graph for imgact_elf32.c:



Defines

- #define __ELF_WORD_SIZE 32

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/imgact_elf32.c,v 1.3 2003/06/11 00:56:54 obrien Exp $")

9.8.1 Define Documentation

9.8.1.1 #define __ELF_WORD_SIZE 32

Definition at line 30 of file imgact_elf32.c.

Referenced by link_elf_init(), and link_elf_link_preload().

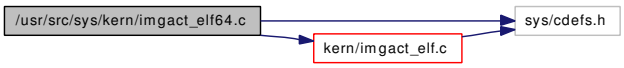
9.8.2 Function Documentation

9.8.2.1 __FBSDID ("$FreeBSD: src/sys/kern/imgact_elf32. *c*, v 1.3 2003/06/11 00:56:54 obrien Exp $")

9.9 /usr/src/sys/kern/imgact_elf64.c File Reference

```
#include <sys/cdefs.h>
#include <kern/imgact_elf.c>
```

Include dependency graph for imgact_elf64.c:



Defines

- #define __ELF_WORD_SIZE 64

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/imgact_elf64.c,v 1.3 2003/06/11 00:56:54 obrien Exp $")

9.9.1 Define Documentation

9.9.1.1 #define __ELF_WORD_SIZE 64

Definition at line 30 of file imgact_elf64.c.

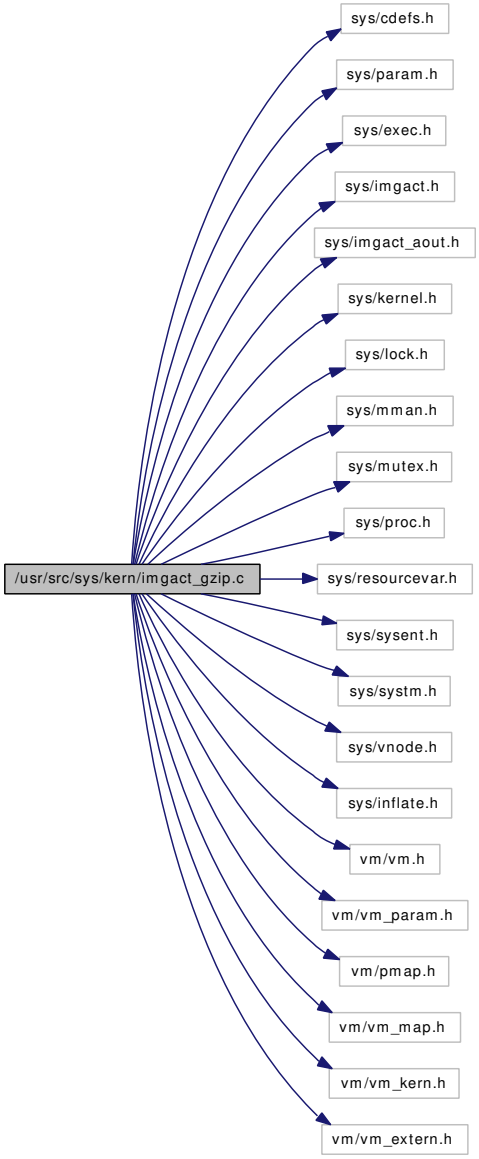
9.9.2 Function Documentation

9.9.2.1 __FBSDID ("$FreeBSD: src/sys/kern/imgact_elf64. *c*, v 1.3 2003/06/11 00:56:54 obrien Exp $")

9.10 /usr/src/sys/kern/imgact_gzip.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/exec.h>
#include <sys/imgact.h>
#include <sys/imgact_aout.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/mman.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/resourcevar.h>
#include <sys/sysent.h>
#include <sys/systm.h>
#include <sys/vnode.h>
#include <sys/inflate.h>
#include <vm/vm.h>
#include <vm/vm_param.h>
#include <vm/pmap.h>
#include <vm/vm_map.h>
#include <vm/vm_kern.h>
#include <vm/vm_extern.h>
```

Include dependency graph for imgact_gzip.c:



Data Structures

- struct imgact_gzip

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/imgact_gzip.c,v 1.55 2005/12/24 04:57:50 alc Exp $")
- static int exec_gzip_imgact (struct image_params *imgp)
- static int NextByte (void *vp)
- static int do_aout_hdr (struct imgact_gzip *)
- static int Flush (void *vp, u_char *, u_long siz)
- EXEC_SET (execgzip, gzip_execsw)

Variables

- static struct execsw gzip_execsw = {exec_gzip_imgact, "gzip"}

9.10.1 Function Documentation

9.10.1.1 __FBSDID ("$FreeBSD: src/sys/kern/imgact_gzip. *c*, v 1.55 2005/12/24 04:57:50 alc Exp $")

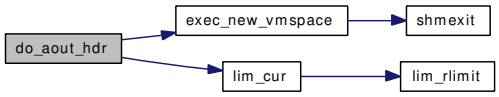
9.10.1.2 static int do_aout_hdr (struct imgact_gzip *) [static]

Definition at line 158 of file imgact_gzip.c.

References imgact_gzip::a_out, aout_sysvec, imgact_gzip::bss_size, exec_new_vmspace(), imgact_gzip::file_end, imgact_gzip::file_offset, imgact_gzip::ip, lim_cur(), maxtsiz, td, imgact_gzip::virtual_offset, and imgact_gzip::where.

Referenced by Flush().

Here is the call graph for this function:

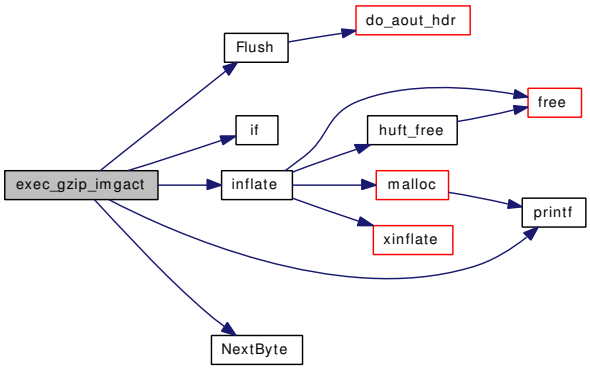


9.10.1.3 static int exec_gzip_imgact (struct image_params * *imgp*) [static]

Definition at line 69 of file imgact_gzip.c.

References Flush(), if(), inflate(), NextByte(), and printf().

Here is the call graph for this function:



9.10.1.4 EXEC_SET (execgzip, gzip_execsw)

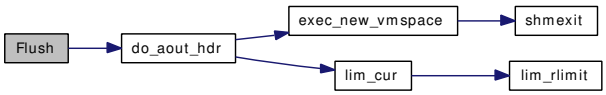
9.10.1.5 static int Flush (void * *vp*, u_char *, u_long *siz*) [static]

Definition at line 340 of file imgact_gzip.c.

References imgact_gzip::a_out, do_aout_hdr(), imgact_gzip::error, imgact_gzip::file_end, imgact_gzip::file_offset, imgact_gzip::gotheader, imgact_gzip::output, imgact_gzip::virtual_offset, and imgact_gzip::where.

Referenced by exec_gzip_imgact().

Here is the call graph for this function:



9.10.1.6 static int NextByte (void * *vp*) [static]

Definition at line 299 of file imgact_gzip.c.

References imgact_gzip::error, imgact_gzip::idx, imgact_gzip::inbuf, imgact_gzip::ip, imgact_gzip::len, imgact_gzip::offset, and imgact_gzip::where.

Referenced by exec_gzip_imgact().

9.10.2 Variable Documentation

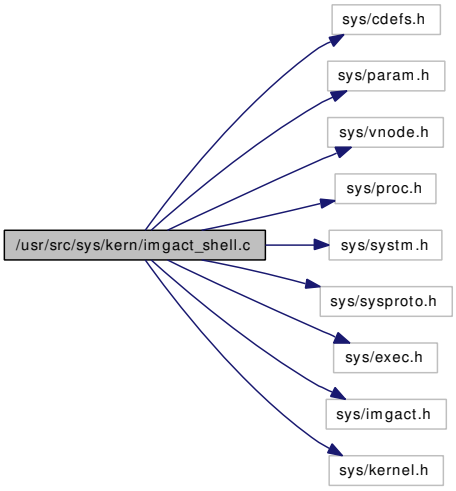
9.10.2.1 struct execsw gzip_execsw = {exec_gzip_imgact, "gzip"} [static]

Definition at line 398 of file imgact_gzip.c.

9.11 /usr/src/sys/kern/imgact_shell.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/vnode.h>
#include <sys/proc.h>
#include <sys/systm.h>
#include <sys/sysproto.h>
#include <sys/exec.h>
#include <sys/imgact.h>
#include <sys/kernel.h>
```

Include dependency graph for imgact_shell.c:



Defines

- #define SHELLMAGIC 0x2123

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/imgact_shell.c,v 1.35 2005/06/19 02:21:03 gad Exp $")
- int exec_shell_imgact (struct image_params *imgp)
- EXEC_SET (shell, shell_execsw)

Variables

- static struct execsw shell_execsw = { exec_shell_imgact, "#!" }

9.11.1 Define Documentation

9.11.1.1 #define SHELLMAGIC 0x2123

Definition at line 40 of file imgact_shell.c.

Referenced by exec_shell_imgact().

9.11.2 Function Documentation

9.11.2.1 __FBSDID ("$FreeBSD: src/sys/kern/imgact_shell. *c*, v 1.35 2005/06/19 02:21:03 gad Exp $")

9.11.2.2 EXEC_SET (shell, shell_execsw)

9.11.2.3 int exec_shell_imgact (struct image_params * *imgp*)

Shell interpreter image activator. An interpreter name beginning at imgp->args->begin_argv is the minimal successful exit requirement.

If the given file is a shell-script, then the first line will start with the two characters '#!' (aka SHELLMAGIC), followed by the name of the shell-interpreter to run, followed by zero or more tokens.

The interpreter is then started up such that it will see: arg[0] -> The name of interpreter as specified after '#!' in the first line of the script. The interpreter name must not be longer than MAXSHELLCMDLEN bytes. arg[1] -> *If* there are any additional tokens on the first line, then we add a new arg[1], which is a copy of the rest of that line. The copy starts at the first token after the interpreter name. We leave it to the interpreter to parse the tokens in that value. arg[x] -> the full pathname of the script. This will either be arg[2] or arg[1], depending on whether or not tokens were found after the interpreter name. arg[x+1] -> all the arguments that were specified on the original command line.

This processing is described in the execve(2) man page.

Definition at line 94 of file imgact_shell.c.

References SHELLMAGIC.

9.11.3 Variable Documentation

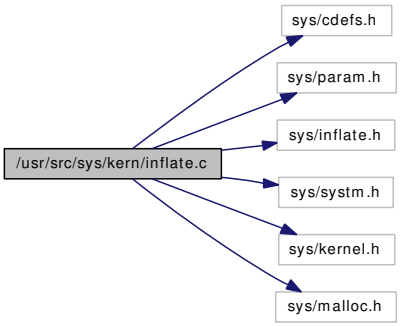
9.11.3.1 struct execsw shell_execsw = { exec_shell_imgact, "#!" } [static]

Definition at line 237 of file imgact_shell.c.

9.12 /usr/src/sys/kern/inflate.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/inflate.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/malloc.h>
```

Include dependency graph for inflate.c:



Data Structures

- struct huft

Defines

- #define uch u_char
- #define ush u_short
- #define ulg u_long
- #define memzero(dest, len) bzero(dest,len)
- #define NOMEMCPY
- #define FPRINTF printf
- #define FLUSH(x, y)
- #define PKZIP_BUG_WORKAROUND
- #define INFMOD
- #define NEEDBITS(glbl, n)
- #define DUMPBITS(n) {b>>=(n);k-=(n);}
- #define BMAX 16
- #define N_MAX 288

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/inflate.c,v 1.20 2005/10/31 15:41:25 rwatson Exp $")
- static MALLOC_DEFINE (M_GZIP,"gzip_trees","Gzip trees")

- static int huft_build (struct inflate *, unsigned *, unsigned, unsigned, const ush *, const ush *, struct huft **, int *)
- static int huft_free (struct inflate *, struct huft *)
- static int inflate_codes (struct inflate *, struct huft *, struct huft *, int, int)
- static int inflate_stored (struct inflate *)
- static int xinflate (struct inflate *)
- static int inflate_fixed (struct inflate *)
- static int inflate_dynamic (struct inflate *)
- static int inflate_block (struct inflate *, int *)
- int inflate (struct inflate *glbl)

Variables

- static const int qflag = 0
- static const unsigned border [ ]
- static const ush cplens [ ]
- static const ush cplext [ ]
- static const ush cpdist [ ]
- static const ush cpdext [ ]
- static const ush mask [ ]
- static const int lbits = 9
- static const int dbits = 6

9.12.1 Define Documentation

9.12.1.1 #define BMAX 16

Definition at line 402 of file inflate.c.

Referenced by huft_build().

9.12.1.2 #define DUMPBITS(n) {b>>=(n);k-=(n);}

Definition at line 363 of file inflate.c.

Referenced by inflate_block(), inflate_codes(), inflate_dynamic(), and inflate_stored().

9.12.1.3 #define FLUSH(x, y)

Value:

```
{                                                   \
        int foo = (*x->gz_output)(x->gz_private,x->gz_slide,y); \
        if (foo)                                                \
                return foo;                                     \
        }
```

Definition at line 43 of file inflate.c.

Referenced by inflate_codes(), inflate_stored(), and xinflate().

9.12.1.4 #define FPRINTF printf

Definition at line 37 of file inflate.c.

Referenced by inflate_dynamic().

9.12.1.5 #define INFMOD

Definition at line 254 of file inflate.c.

9.12.1.6 #define memzero(dest, len) bzero(dest,len)

Definition at line 33 of file inflate.c.

Referenced by huft_build().

9.12.1.7 #define N_MAX 288

Definition at line 404 of file inflate.c.

Referenced by huft_build().

9.12.1.8 #define NEEDBITS(glbl, n)

Value:

```
{                                               \
                while(k<(n)) {                                          \
                        int c=(*glbl->gz_input)(glbl->gz_private);      \
                        if(c==GZ_EOF)                                   \
                                return 1;                               \
                        b|=((ulg)c)<<k;                                 \
                        k+=8;                                           \
                }                                                       \
        }
```

Definition at line 353 of file inflate.c.

Referenced by inflate_block(), inflate_codes(), inflate_dynamic(), and inflate_stored().

9.12.1.9 #define NOMEMCPY

Definition at line 35 of file inflate.c.

9.12.1.10 #define PKZIP_BUG_WORKAROUND

Definition at line 234 of file inflate.c.

9.12.1.11 #define uch u_char

Definition at line 27 of file inflate.c.

Referenced by huft_build(), inflate_codes(), and inflate_stored().

9.12.1.12 #define ulg u_long

Definition at line 29 of file inflate.c.

Referenced by inflate_block(), inflate_codes(), inflate_dynamic(), and inflate_stored().

9.12.1.13 #define ush u_short

Definition at line 28 of file inflate.c.

9.12.2 Function Documentation

9.12.2.1 __FBSDID ("$FreeBSD: src/sys/kern/inflate. *c*, v 1.20 2005/10/31 15:41:25 rwatson Exp $")

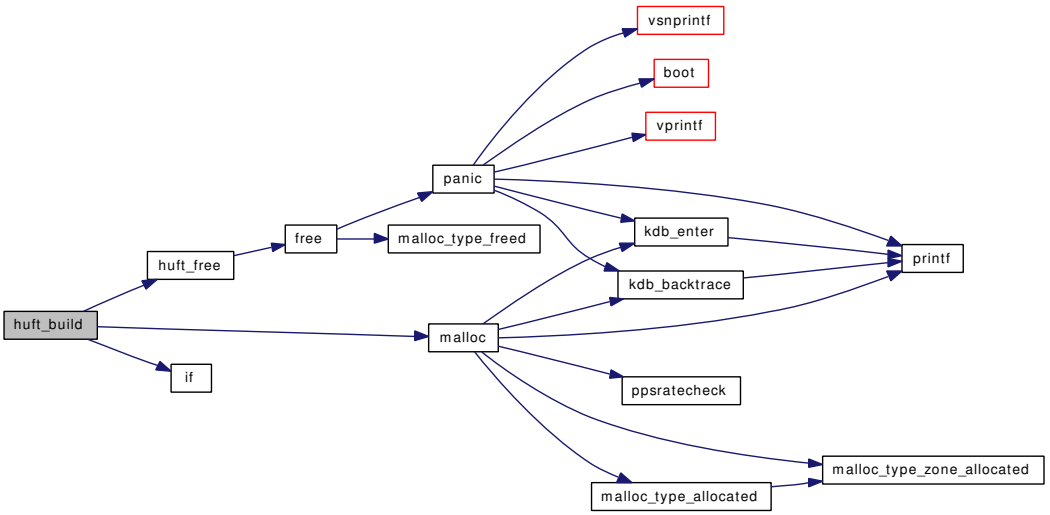
9.12.2.2 static int huft_build (struct inflate *, unsigned *, unsigned, unsigned, const ush *, const ush *, struct huft **, int *) [static]

Definition at line 415 of file inflate.c.

References huft::b, BMAX, huft::e, huft_free(), if(), malloc(), memzero, huft::n, N_MAX, huft::t, uch, and huft::v.

Referenced by inflate_dynamic(), and inflate_fixed().

Here is the call graph for this function:



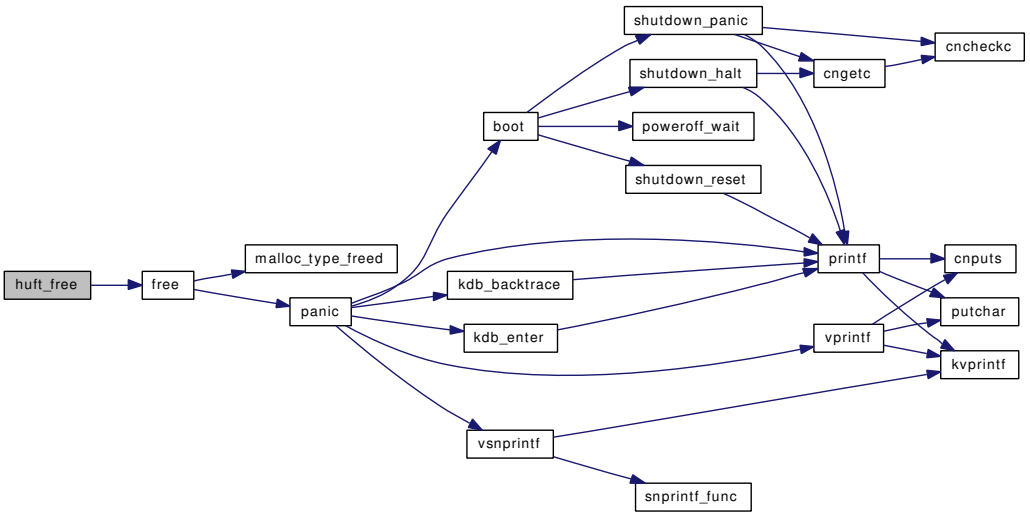
9.12.2.3 static int huft_free (struct inflate *, struct huft *) [static]

Definition at line 618 of file inflate.c.

References free(), huft::t, and huft::v.

Referenced by huft_build(), inflate(), inflate_dynamic(), and inflate_fixed().

Here is the call graph for this function:



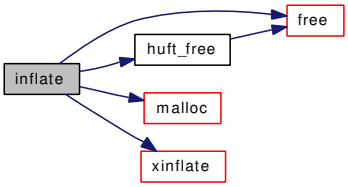
9.12.2.4 int inflate (struct inflate * *glbl*)

Definition at line 1043 of file inflate.c.

References free(), huft_free(), malloc(), and xinflate().

Referenced by exec_gzip_imgact().

Here is the call graph for this function:



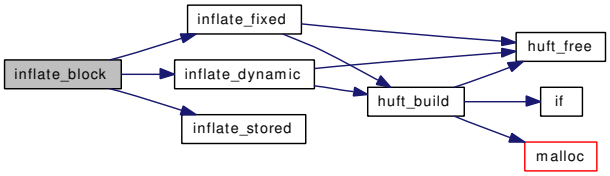
9.12.2.5 static int inflate_block (struct inflate *, int *) [static]

Definition at line 971 of file inflate.c.

References huft::b, DUMPBITS, inflate_dynamic(), inflate_fixed(), inflate_stored(), NEEDBITS, huft::t, and ulg.

Referenced by xinflate().

Here is the call graph for this function:



9.12.2.6 static int inflate_codes (struct inflate *, struct huft *, struct huft *, int, int) [static]

Definition at line 640 of file inflate.c.

References huft::b, DUMPBITS, huft::e, FLUSH, if(), mask, huft::n, NEEDBITS, huft::t, td, uch, ulg, and huft::v.

Here is the call graph for this function:

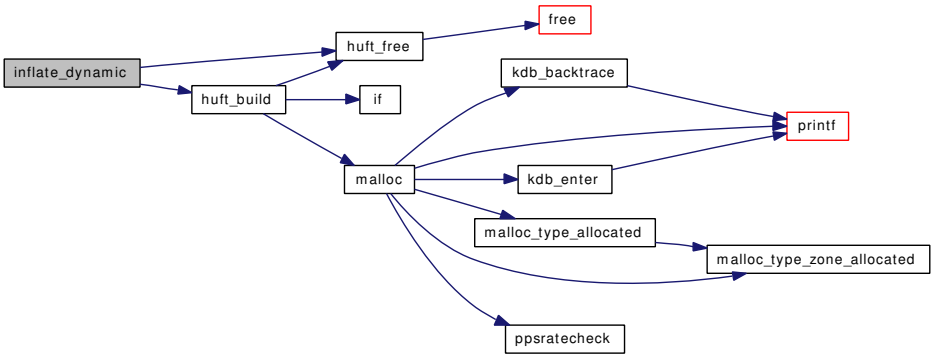
9.12.2.7 static int inflate_dynamic (struct inflate *) [static]

Definition at line 825 of file inflate.c.

References huft::b, border, cpdext, cpdist, cplens, cplext, dbits, DUMPBITS, FPRINTF, huft_build(), huft_free(), lbits, mask, huft::n, NEEDBITS, qflag, td, and ulg.

Referenced by inflate_block().

Here is the call graph for this function:



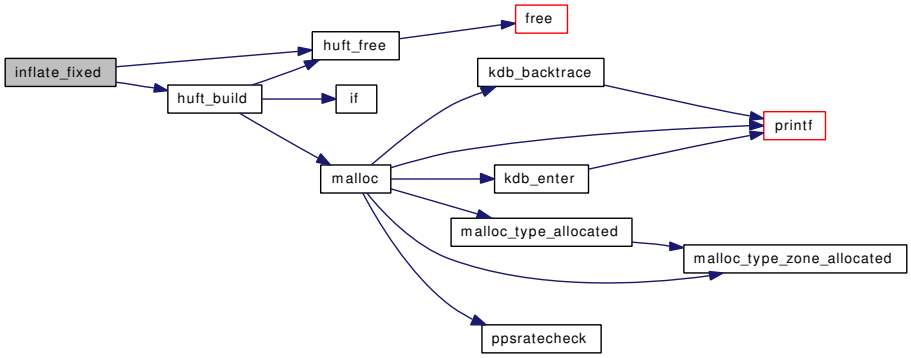
9.12.2.8 static int inflate_fixed (struct inflate *) [static]

Definition at line 783 of file inflate.c.

References cpdext, cpdist, cplens, cplext, huft_build(), and huft_free().

Referenced by inflate_block().

Here is the call graph for this function:



9.12.2.9 static int inflate_stored (struct inflate *) [static]

Definition at line 736 of file inflate.c.

References huft::b, DUMPBITS, FLUSH, huft::n, NEEDBITS, uch, and ulg.

Referenced by inflate_block().

9.12.2.10 static MALLOC_DEFINE (M_GZIP, "gzip_trees", "Gzip trees") [static]

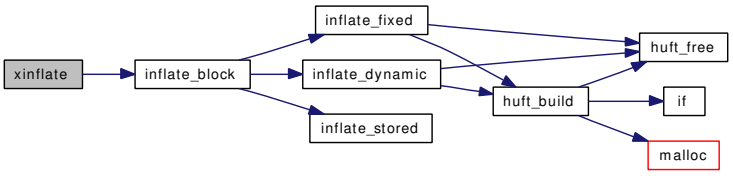
9.12.2.11 static int xinflate (struct inflate *) [static]

Definition at line 1010 of file inflate.c.

References huft::e, FLUSH, and inflate_block().

Referenced by inflate().

Here is the call graph for this function:



9.12.3 Variable Documentation

9.12.3.1 const unsigned border[ ] [static]

Initial value:

```
{
        16, 17, 18, 0, 8, 7, 9, 6, 10, 5, 11, 4, 12, 3, 13, 2, 14, 1, 15}
```

Definition at line 298 of file inflate.c.

Referenced by inflate_dynamic().

9.12.3.2 const ush cpdext[ ] [static]

Initial value:

```
{
        0, 0, 0, 0, 1, 1, 2, 2, 3, 3, 4, 4, 5, 5, 6, 6,
        7, 7, 8, 8, 9, 9, 10, 10, 11, 11,
        12, 12, 13, 13}
```

Definition at line 315 of file inflate.c.

Referenced by inflate_dynamic(), and inflate_fixed().

9.12.3.3 const ush cpdist[ ] [static]

Initial value:

```
{
        1, 2, 3, 4, 5, 7, 9, 13, 17, 25, 33, 49, 65, 97, 129, 193,
        257, 385, 513, 769, 1025, 1537, 2049, 3073, 4097, 6145,
        8193, 12289, 16385, 24577}
```

Definition at line 310 of file inflate.c.

Referenced by inflate_dynamic(), and inflate_fixed().

9.12.3.4 const ush cplens[ ] [static]

Initial value:

```
{
        3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 15, 17, 19, 23, 27, 31,
        35, 43, 51, 59, 67, 83, 99, 115, 131, 163, 195, 227, 258, 0, 0}
```

Definition at line 301 of file inflate.c.

Referenced by inflate_dynamic(), and inflate_fixed().

9.12.3.5 const ush cplext[ ] [static]

Initial value:

```
{
        0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 2, 2, 2, 2,
        3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 0, 99, 99}
```

Definition at line 306 of file inflate.c.

Referenced by inflate_dynamic(), and inflate_fixed().

9.12.3.6 const int dbits = 6 [static]

Definition at line 398 of file inflate.c.

Referenced by inflate_dynamic().

9.12.3.7 const int lbits = 9 [static]

Definition at line 397 of file inflate.c.

Referenced by inflate_dynamic().

9.12.3.8 const ush mask[ ] [static]

Initial value:

```
 {
        0x0000,
        0x0001, 0x0003, 0x0007, 0x000f, 0x001f, 0x003f, 0x007f, 0x00ff,
        0x01ff, 0x03ff, 0x07ff, 0x0fff, 0x1fff, 0x3fff, 0x7fff, 0xffff
}
```

Definition at line 321 of file inflate.c.

Referenced by blst_leaf_alloc(), blst_leaf_fill(), blst_leaf_free(), fd_first_free(), fd_last_used(), inflate_codes(), inflate_dynamic(), sigsuspend(), and vfs_bio_clrbuf().

9.12.3.9 const int qflag = 0 [static]

Definition at line 49 of file inflate.c.

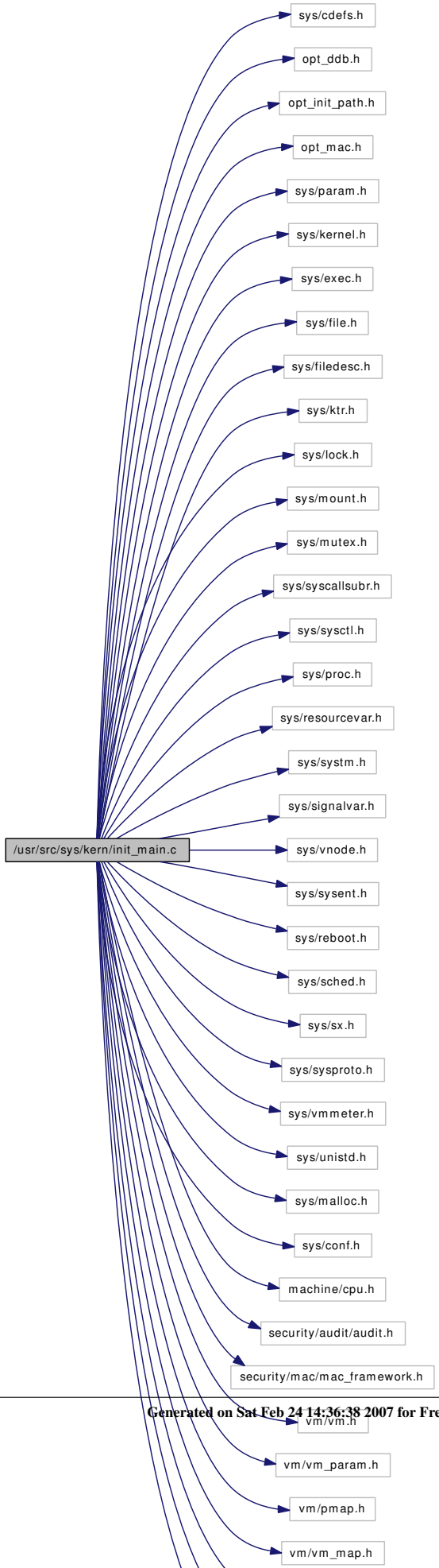
Referenced by inflate_dynamic(), and kvprintf().

9.13 /usr/src/sys/kern/init_main.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ddb.h"
#include "opt_init_path.h"
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/kernel.h>
#include <sys/exec.h>
#include <sys/file.h>
#include <sys/filedesc.h>
#include <sys/ktr.h>
#include <sys/lock.h>
#include <sys/mount.h>
#include <sys/mutex.h>
#include <sys/syscallsubr.h>
#include <sys/sysctl.h>
#include <sys/proc.h>
#include <sys/resourcevar.h>
#include <sys/systm.h>
#include <sys/signalvar.h>
#include <sys/vnode.h>
#include <sys/sysent.h>
#include <sys/reboot.h>
#include <sys/sched.h>
#include <sys/sx.h>
#include <sys/sysproto.h>
#include <sys/vmmeter.h>
#include <sys/unistd.h>
#include <sys/malloc.h>
#include <sys/conf.h>
#include <machine/cpu.h>
#include <security/audit/audit.h>
#include <security/mac/mac_framework.h>
#include <vm/vm.h>
#include <vm/vm_param.h>
#include <vm/pmap.h>
```

```
#include <vm/vm_map.h>
#include <sys/copyright.h>
#include <ddb/ddb.h>
#include <ddb/db_sym.h>
```

Include dependency graph for init_main.c:


/usr/src/sys/kern/init_main.c
sys/cdefs.h
opt_ddb.h
opt_init_path.h
opt_mac.h
sys/param.h
sys/kernel.h
sys/exec.h
sys/file.h
sys/filedesc.h
sys/ktr.h
sys/lock.h
sys/mount.h
sys/mutex.h
sys/syscallsubr.h
sys/sysctl.h
sys/proc.h
sys/resourcevar.h
sys/systm.h
sys/signalvar.h
sys/vnode.h
sys/sysent.h
sys/reboot.h
sys/sched.h
sys/sx.h
sys/sysproto.h
sys/vmmeter.h
sys/unistd.h
sys/malloc.h
sys/conf.h
machine/cpu.h
security/audit/audit.h
security/mac/mac_framework.h
vm/vm.h
vm/vm_param.h
vm/pmap.h
vm/vm_map.h

Defines

- #define INIT_SHUTDOWN_TIMEOUT 120

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/init_main.c,v 1.273 2007/01/23 08:46:50 jeff Exp $")
- void mi_startup (void)
- thread thread0 __aligned (8)
- SYSCTL_INT (_debug, OID_AUTO, boothowto, CTLFLAG_RD,&boothowto, 0,"")
- SYSCTL_INT (_debug, OID_AUTO, bootverbose, CTLFLAG_RW,&bootverbose, 0,"")
- SET_DECLARE (sysinit_set, struct sysinit)
- void sysinit_add (struct sysinit **set, struct sysinit **set_end)
- static void print_caddr_t (void *data __unused)
- static void set_boot_verbose (void *data __unused)
- static void proc0_init (void *dummy __unused)
- static void proc0_post (void *dummy __unused)
- SYSCTL_STRING (_kern, OID_AUTO, init_path, CTLFLAG_RD, init_path, 0,"Path used to search the init process")
- SYSCTL_INT (_kern, OID_AUTO, init_shutdown_timeout, CTLFLAG_RW,&init_shutdown_timeout, 0,"")
- static void start_init (void *dummy)
- static void create_init (const void *udata __unused)
- static void kick_init (const void *udata __unused)

Variables

- static struct session session0
- static struct pgrp pgrp0
- proc proc0
- vmspace vmspace0
- proc * initproc
- int boothowto = 0
- int bootverbose
- sysinit ** sysinit
- sysinit ** sysinit_end
- sysinit ** newsysinit
- sysinit ** newsysinit_end
- sysentvec null_sysvec
- static char init_path [MAXPATHLEN]
- static int init_shutdown_timeout = INIT_SHUTDOWN_TIMEOUT

9.13.1 Define Documentation

9.13.1.1 #define INIT_SHUTDOWN_TIMEOUT 120

Definition at line 557 of file init_main.c.

9.13.2 Function Documentation

9.13.2.1 struct thread thread0 __aligned (8)

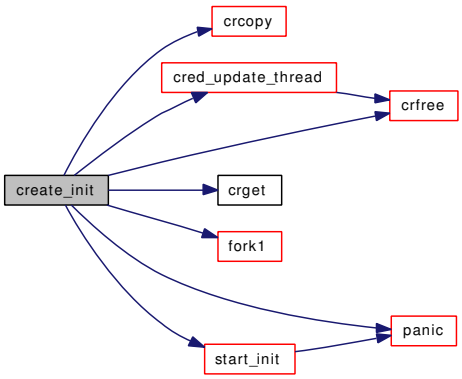
9.13.2.2 __FBSDID ("$FreeBSD: src/sys/kern/init_main. *c*, v 1.273 2007/01/23 08:46:50 jeff Exp $")

9.13.2.3 static void create_init (const void *udata *__unused*) [static]

Definition at line 690 of file init_main.c.

References crcopy(), cred_update_thread(), crfree(), crget(), fork1(), initproc, panic(), sched_lock, and start_init().

Here is the call graph for this function:

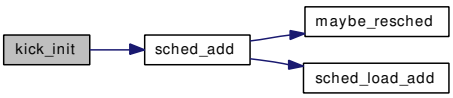


9.13.2.4 static void kick_init (const void *udata *__unused*) [static]

Definition at line 726 of file init_main.c.

References initproc, sched_add(), sched_lock, and td.

Here is the call graph for this function:

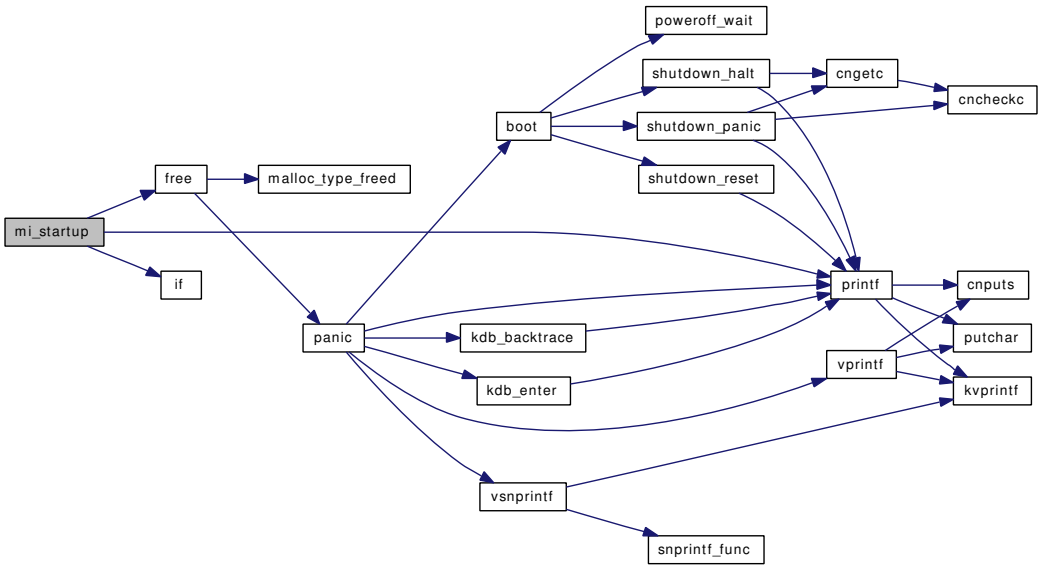


9.13.2.5 void mi_startup (void)

Definition at line 168 of file init_main.c.

References free(), if(), newsysinit, newsysinit_end, printf(), sysinit, and sysinit_end.

Here is the call graph for this function:

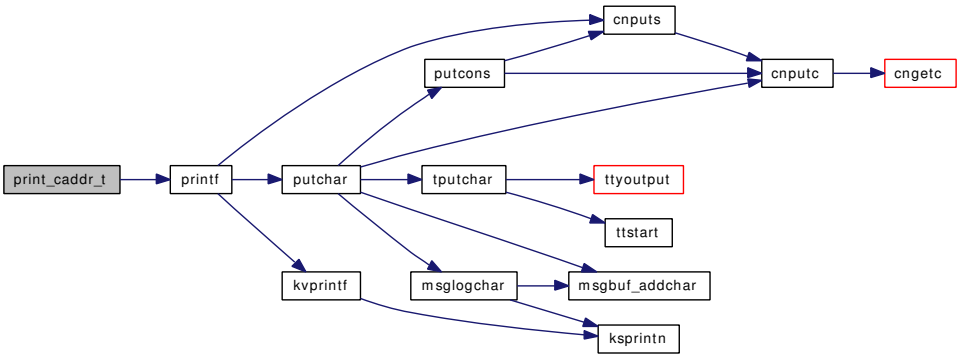


9.13.2.6 static void print_caddr_t (void *data *__unused*) [static]

Definition at line 286 of file init_main.c.

References printf().

Here is the call graph for this function:

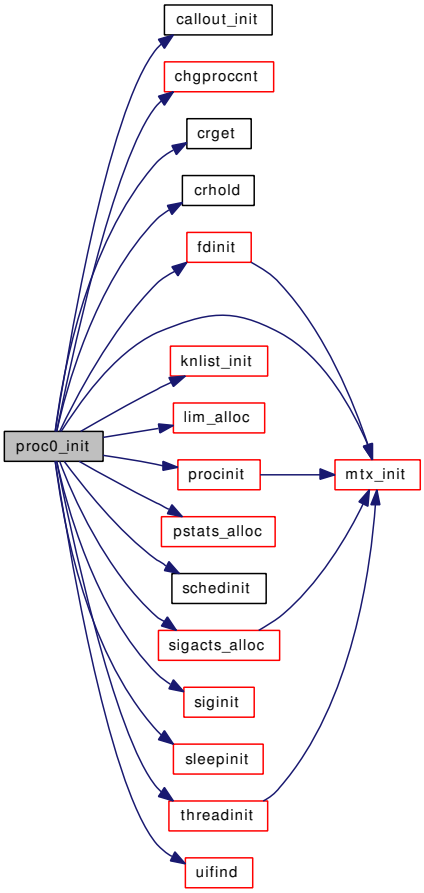


9.13.2.7 static void proc0_init (void *dummy *__unused*) [static]

Definition at line 361 of file init_main.c.

References allproc, callout_init(), chgproccnt(), crget(), crhold(), fdinit(), knlist_init(), lim_alloc(), maxfiles, maxproc, mtx_init(), null_sysvec, pgrp0, proc0, procinit(), pstats_alloc(), schedinit(), session0, sigacts_alloc(), siginit(), sleepinit(), td, threadinit(), uifind(), and vmspace0.

Here is the call graph for this function:

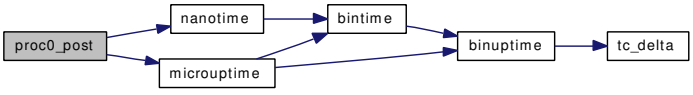


9.13.2.8 static void proc0_post (void *dummy *__unused*) [static]

Definition at line 495 of file init_main.c.

References allproc_lock, cpu_ticks, microuptime(), nanotime(), and ticks.

Here is the call graph for this function:



9.13.2.9 static void set_boot_verbose (void *data *__unused*) [static]

Definition at line 309 of file init_main.c.

References boothowto, and bootverbose.

9.13.2.10 SET_DECLARE (sysinit_set, struct *sysinit*)

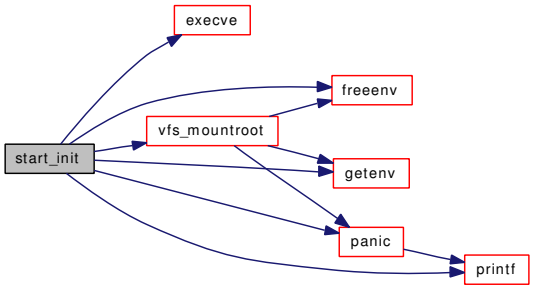
9.13.2.11 static void start_init (void * *dummy*) [static]

Definition at line 568 of file init_main.c.

References execve_args::argv, boothowto, bootverbose, execve_args::envv, execve(), execve_args::fname, freeenv(), getenv(), Giant, init_path, panic(), printf(), suword, td, ucp, and vfs_mountroot().

Referenced by create_init().

Here is the call graph for this function:



9.13.2.12 SYSCTL_INT (_kern, OID_AUTO, init_shutdown_timeout, CTLFLAG_RW, & *init_shutdown_timeout*, 0, "")

9.13.2.13 SYSCTL_INT (_debug, OID_AUTO, bootverbose, CTLFLAG_RW, & *bootverbose*, 0, "")

9.13.2.14 SYSCTL_INT (_debug, OID_AUTO, boothowto, CTLFLAG_RD, & *boothowto*, 0, "")

9.13.2.15 SYSCTL_STRING (_kern, OID_AUTO, init_path, CTLFLAG_RD, init_path, 0, "Path used to search the init process")

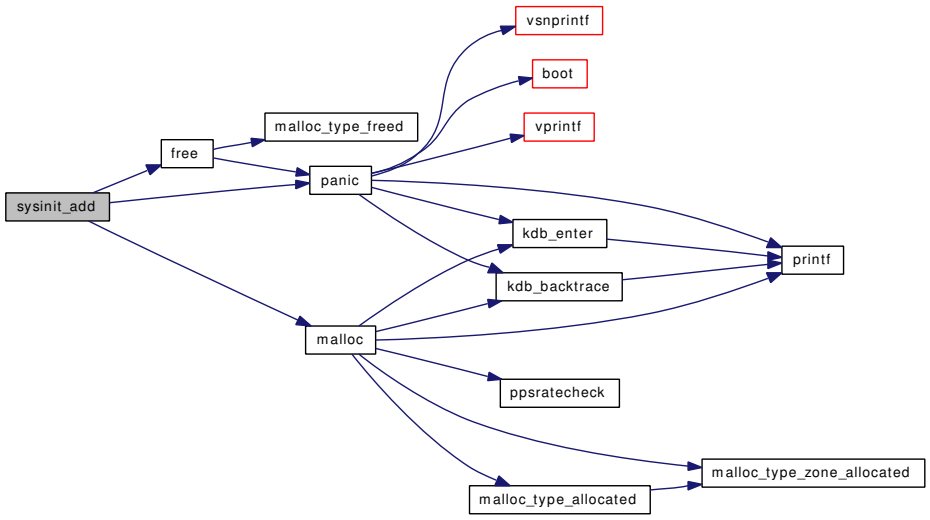
9.13.2.16 void sysinit_add (struct sysinit ** *set*, struct sysinit ** *set_end*)

Definition at line 126 of file init_main.c.

References free(), malloc(), and panic().

Referenced by linker_preload().

Here is the call graph for this function:



9.13.3 Variable Documentation

9.13.3.1 int boothowto = 0

Definition at line 101 of file init_main.c.

Referenced by set_boot_verbose(), start_init(), and vfs_mountroot().

9.13.3.2 int bootverbose

Definition at line 103 of file init_main.c.

Referenced by bus_setup_intr(), clock_register(), cpu_tick_calibrate(), devclass_alloc_unit(), device_probe_and_attach(), firmware_register(), linker_preload(), msgbuf_reinit(), power_profile_set_state(), pty_clone(), pty_maybecleanup(), set_boot_verbose(), start_init(), tc_init(), vfs_mount_destroy(), and vntblinit().

9.13.3.3 char init_path[MAXPATHLEN] [static]

Initial value:

```
"/sbin/init:/sbin/oinit:/sbin/init.bak:/rescue/init:/stand/sysinstall"
```

Definition at line 543 of file init_main.c.

Referenced by start_init().

9.13.3.4 int init_shutdown_timeout = INIT_SHUTDOWN_TIMEOUT [static]

Definition at line 559 of file init_main.c.

9.13.3.5 struct proc* initproc

Definition at line 99 of file init_main.c.

Referenced by create_init(), exit1(), fork1(), kern_ptrace(), kick_init(), kthread_exit(), and shutdown_nice().

9.13.3.6 struct sysinit** newsysinit

Definition at line 119 of file init_main.c.

Referenced by mi_startup().

9.13.3.7 struct sysinit ** newsysinit_end

Definition at line 119 of file init_main.c.

Referenced by mi_startup().

9.13.3.8 struct sysentvec null_sysvec

Initial value:

```
{
        0,
        NULL,
        0,
        0,
        NULL,
        0,
        NULL,
        NULL,
        NULL,
        NULL,
        NULL,
        NULL,
        NULL,
        "null",
        NULL,
        NULL,
        0,
        PAGE_SIZE,
        VM_MIN_ADDRESS,
        VM_MAXUSER_ADDRESS,
        USRSTACK,
        PS_STRINGS,
        VM_PROT_ALL,
        NULL,
        NULL,
        NULL
}
```

Definition at line 317 of file init_main.c.

Referenced by proc0_init().

9.13.3.9 struct pgrp pgrp0 [static]

Definition at line 95 of file init_main.c.

Referenced by proc0_init().

9.13.3.10 struct proc proc0

Definition at line 96 of file init_main.c.

Referenced by fork1(), kthread_create(), mutex_init(), proc0_init(), and schedinit().

9.13.3.11 struct session session0 [static]

Definition at line 94 of file init_main.c.

Referenced by proc0_init().

9.13.3.12 struct sysinit** sysinit

Definition at line 118 of file init_main.c.

Referenced by linker_file_sysinit(), linker_file_sysuninit(), linker_preload(), and mi_startup().

9.13.3.13 struct sysinit ** sysinit_end

Definition at line 118 of file init_main.c.

Referenced by mi_startup().

9.13.3.14 struct vmspace vmspace0

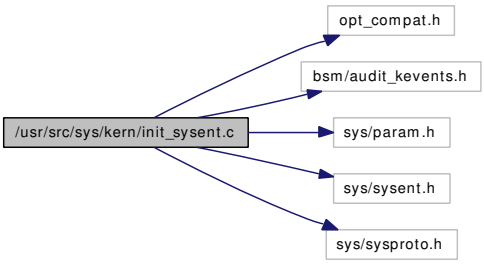
Definition at line 98 of file init_main.c.

Referenced by proc0_init().

9.14 /usr/src/sys/kern/init_sysent.c File Reference

```
#include "opt_compat.h"
#include <bsm/audit_kevents.h>
#include <sys/param.h>
#include <sys/sysent.h>
#include <sys/sysproto.h>
```

Include dependency graph for init_sysent.c:



Defines

- #define AS(name) (sizeof(struct name) / sizeof(register_t))
- #define compat(n, name) 0, (sy_call_t *)nosys
- #define compat4(n, name) 0, (sy_call_t *)nosys

Variables

- sysent sysent [ ]

9.14.1 Define Documentation

9.14.1.1 #define AS(name) (sizeof(struct name) / sizeof(register_t))

Definition at line 16 of file init_sysent.c.

9.14.1.2 #define compat(n, name) 0, (sy_call_t *)nosys

Definition at line 21 of file init_sysent.c.

9.14.1.3 #define compat4(n, name) 0, (sy_call_t *)nosys

Definition at line 27 of file init_sysent.c.

9.14.2 Variable Documentation

9.14.2.1 struct sysent sysent[ ]

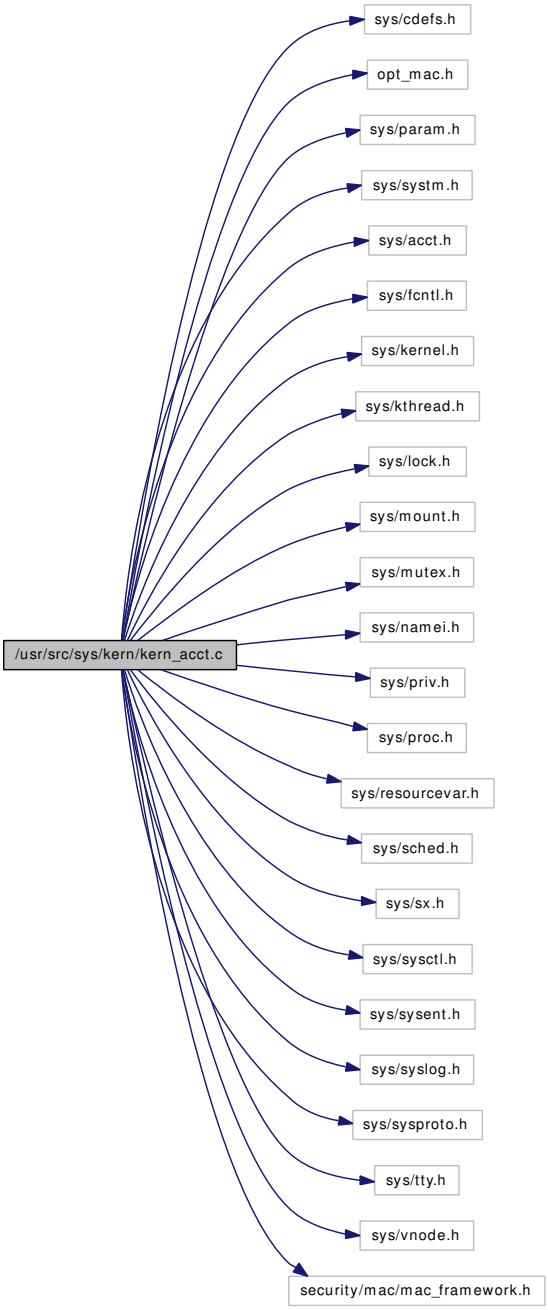
Definition at line 31 of file init_sysent.c.

Referenced by syscall_deregister(), and syscall_register().

9.15 /usr/src/sys/kern/kern_acct.c File Reference

```
#include <sys/cdefs.h>
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/acct.h>
#include <sys/fcntl.h>
#include <sys/kernel.h>
#include <sys/kthread.h>
#include <sys/lock.h>
#include <sys/mount.h>
#include <sys/mutex.h>
#include <sys/namei.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/resourcevar.h>
#include <sys/sched.h>
#include <sys/sx.h>
#include <sys/sysctl.h>
#include <sys/sysent.h>
#include <sys/syslog.h>
#include <sys/sysproto.h>
#include <sys/tty.h>
#include <sys/vnode.h>
#include <security/mac/mac_framework.h>
```

Include dependency graph for kern_acct.c:



Defines

- #define ACCT_RUNNING 1
- #define ACCT_EXITREQ 2
- #define MANTSIZE 13
- #define EXPSIZE 3
- #define MAXFRACT ((1 << MANTSIZE) - 1)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_acct.c,v 1.86 2007/01/08 20:35:13 rwatson Exp $")
- static comp_t encode_comp_t (u_long, u_long)
- static void acctwatch (void)
- static void acct_thread (void *)
- static int acct_disable (struct thread *)
- SX_SYSINIT (acct,&acct_sx,"acct_sx")
- SYSCTL_INT (_kern, OID_AUTO, acct_suspend, CTLFLAG_RW,&acctsuspend, 0,"percentage of free disk space below which accounting stops")
- SYSCTL_INT (_kern, OID_AUTO, acct_resume, CTLFLAG_RW,&acctresume, 0,"percentage of free disk space above which accounting resumes")
- static int sysctl_acct_chkfreq (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern, OID_AUTO, acct_chkfreq, CTLTYPE_INT|CTLFLAG_RW,&acctchkfreq, 0, sysctl_acct_chkfreq,"I","frequency for checking the free space")
- SYSCTL_INT (_kern, OID_AUTO, acct_configured, CTLFLAG_RD,&acct_configured, 0,"Accounting configured or not")
- SYSCTL_INT (_kern, OID_AUTO, acct_suspended, CTLFLAG_RD,&acct_suspended, 0,"Accounting suspended or not")
- int acct (struct thread *td, struct acct_args *uap)
- int acct_process (struct thread *td)

Variables

- static int acct_configured
- static int acct_suspended
- static struct vnode * acct_vp
- static struct ucred * acct_cred
- static int acct_flags
- static struct sx acct_sx
- static int acct_state
- static int acctsuspend = 2
- static int acctresume = 4
- static int acctchkfreq = 15

9.15.1 Define Documentation

9.15.1.1 #define ACCT_EXITREQ 2

Definition at line 140 of file kern_acct.c.

Referenced by acct(), acct_thread(), and acctwatch().

9.15.1.2 #define ACCT_RUNNING 1

Definition at line 139 of file kern_acct.c.

Referenced by acct(), and acct_thread().

9.15.1.3 #define EXPSIZE 3

Definition at line 440 of file kern_acct.c.

Referenced by encode_comp_t().

9.15.1.4 #define MANTSIZE 13

Definition at line 439 of file kern_acct.c.

Referenced by encode_comp_t().

9.15.1.5 #define MAXFRACT ((1 << MANTSIZE) - 1)

Definition at line 441 of file kern_acct.c.

Referenced by encode_comp_t().

9.15.2 Function Documentation

9.15.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_acct. *c*, v 1.86 2007/01/08 20:35:13 rwatson Exp $")

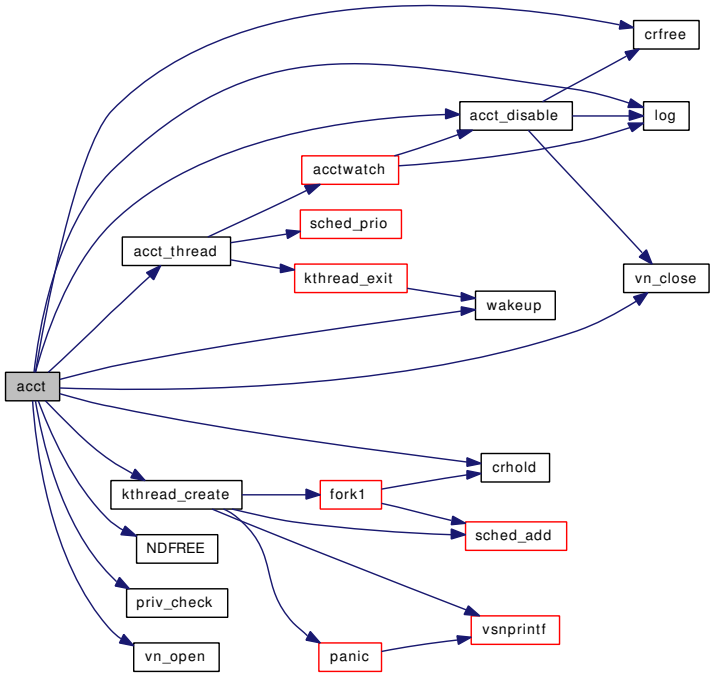
9.15.2.2 int acct (struct thread * *td*, struct acct_args * *uap*)

Definition at line 191 of file kern_acct.c.

References acct_configured, acct_cred, acct_disable(), ACCT_EXITREQ, acct_flags, ACCT_RUNNING, acct_state, acct_suspended, acct_sx, acct_thread(), acct_vp, crfree(), crhold(), kthread_create(), log(), NDFREE(), priv_check(), vn_close(), vn_open(), and wakeup().

Referenced by acct_process().

Here is the call graph for this function:



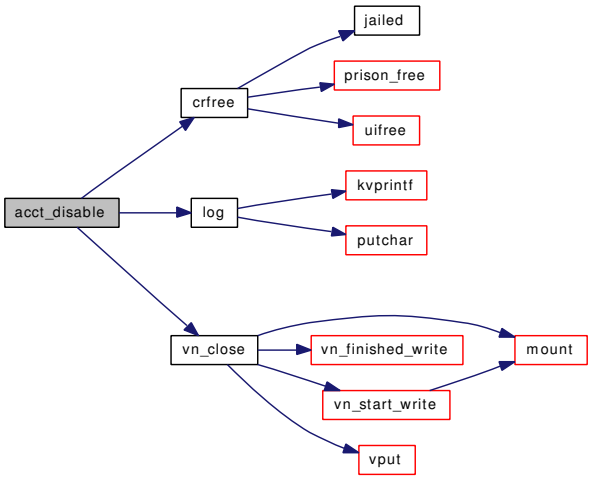
9.15.2.3 static int acct_disable (struct thread *) [static]

Definition at line 306 of file kern_acct.c.

References acct_configured, acct_cred, acct_flags, acct_sx, acct_vp, crfree(), log(), and vn_close().

Referenced by acct(), and acctwatch().

Here is the call graph for this function:



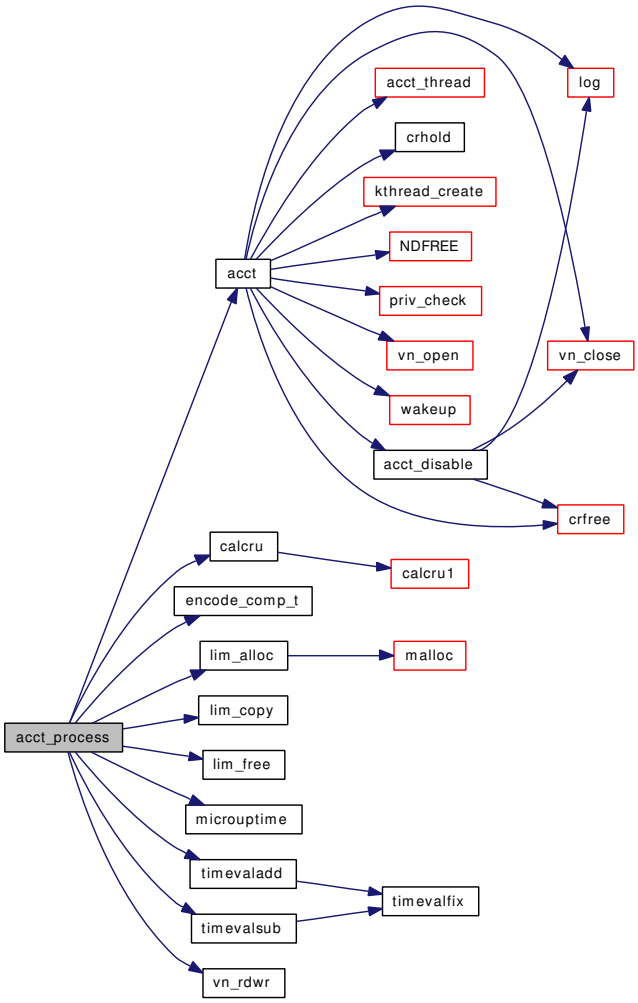
9.15.2.4 int acct_process (struct thread * *td*)

Definition at line 328 of file kern_acct.c.

References acct(), acct_cred, acct_suspended, acct_sx, acct_vp, boottime, calcru(), encode_comp_t(), hz, lim_alloc(), lim_copy(), lim_free(), microuptime(), ret, tick, timevaladd(), timevalsub(), and vn_rdwr().

Referenced by exit1().

Here is the call graph for this function:



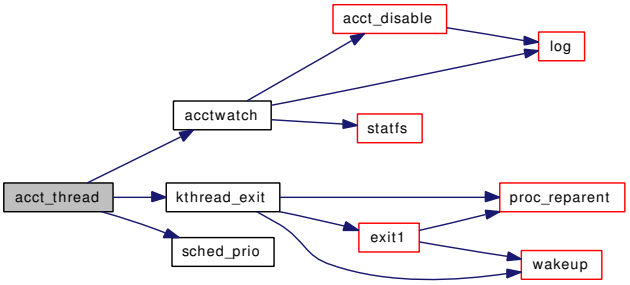
9.15.2.5 static void acct_thread (void *) [static]

Definition at line 537 of file kern_acct.c.

References ACCT_EXITREQ, ACCT_RUNNING, acct_state, acct_sx, acctchkfreq, acctwatch(), hz, kthread_exit(), sched_lock, and sched_prio().

Referenced by acct().

Here is the call graph for this function:



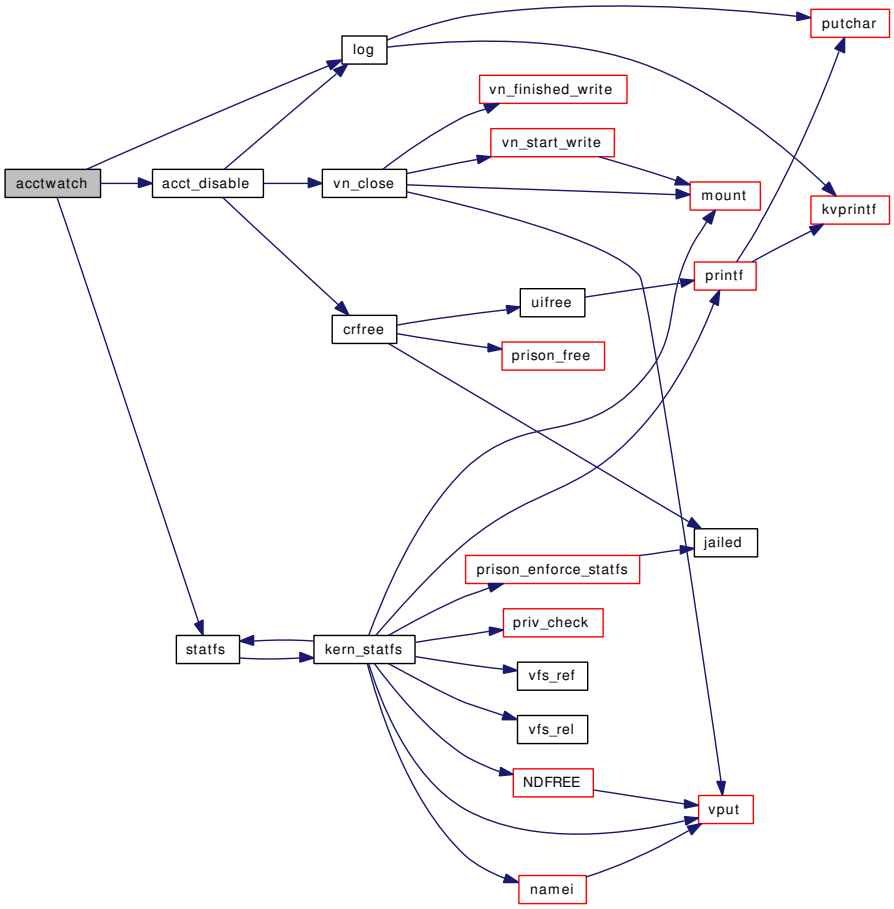
9.15.2.6 static void acctwatch (void) [static]

Definition at line 479 of file kern_acct.c.

References acct_disable(), ACCT_EXITREQ, acct_state, acct_suspended, acct_sx, acct_vp, acctresume, acctsuspend, log(), and statfs().

Referenced by acct_thread().

Here is the call graph for this function:



9.15.2.7 static comp_t encode_comp_t (u_long, u_long) [static]

Definition at line 444 of file kern_acct.c.

References EXPSIZE, MANTSIZE, and MAXFRACT.

Referenced by acct_process().

9.15.2.8 SX_SYSINIT (acct, & *acct_sx*, "acct_sx")

9.15.2.9 static int sysctl_acct_chkfreq (SYSCTL_HANDLER_ARGS) [static]

Definition at line 156 of file kern_acct.c.

References acctchkfreq.

9.15.2.10 SYSCTL_INT (_kern, OID_AUTO, acct_suspended, CTLFLAG_RD, & *acct_suspended*, 0, "Accounting suspended or not")

9.15.2.11 SYSCTL_INT (_kern, OID_AUTO, acct_configured, CTLFLAG_RD, & *acct_configured*, 0, "Accounting configured or not")

9.15.2.12 SYSCTL_INT (_kern, OID_AUTO, acct_resume, CTLFLAG_RW, & *acctresume*, 0, "percentage of free disk space above which accounting resumes")

9.15.2.13 SYSCTL_INT (_kern, OID_AUTO, acct_suspend, CTLFLAG_RW, & *acctsuspend*, 0, "percentage of free disk space below which accounting stops")

9.15.2.14 SYSCTL_PROC (_kern, OID_AUTO, acct_chkfreq, CTLTYPE_INT| *CTLFLAG_RW*, & *acctchkfreq*, 0, sysctl_acct_chkfreq, "I", "frequency for checking the free space")

9.15.3 Variable Documentation

9.15.3.1 int acct_configured [static]

Definition at line 125 of file kern_acct.c.

Referenced by acct(), and acct_disable().

9.15.3.2 struct ucred* acct_cred [static]

Definition at line 128 of file kern_acct.c.

Referenced by acct(), acct_disable(), and acct_process().

9.15.3.3 int acct_flags [static]

Definition at line 129 of file kern_acct.c.

Referenced by acct(), and acct_disable().

9.15.3.4 int acct_state [static]

Definition at line 137 of file kern_acct.c.

Referenced by acct(), acct_thread(), and acctwatch().

9.15.3.5 int acct_suspended [static]

Definition at line 126 of file kern_acct.c.

Referenced by acct(), acct_process(), and acctwatch().

9.15.3.6 struct sx acct_sx [static]

Definition at line 130 of file kern_acct.c.

Referenced by acct(), acct_disable(), acct_process(), acct_thread(), and acctwatch().

9.15.3.7 struct vnode* acct_vp [static]

Definition at line 127 of file kern_acct.c.

Referenced by acct(), acct_disable(), acct_process(), and acctwatch().

9.15.3.8 int acctchkfreq = 15 [static]

Definition at line 153 of file kern_acct.c.

Referenced by acct_thread(), and sysctl_acct_chkfreq().

9.15.3.9 int acctresume = 4 [static]

Definition at line 149 of file kern_acct.c.

Referenced by acctwatch().

9.15.3.10 int acctsuspend = 2 [static]

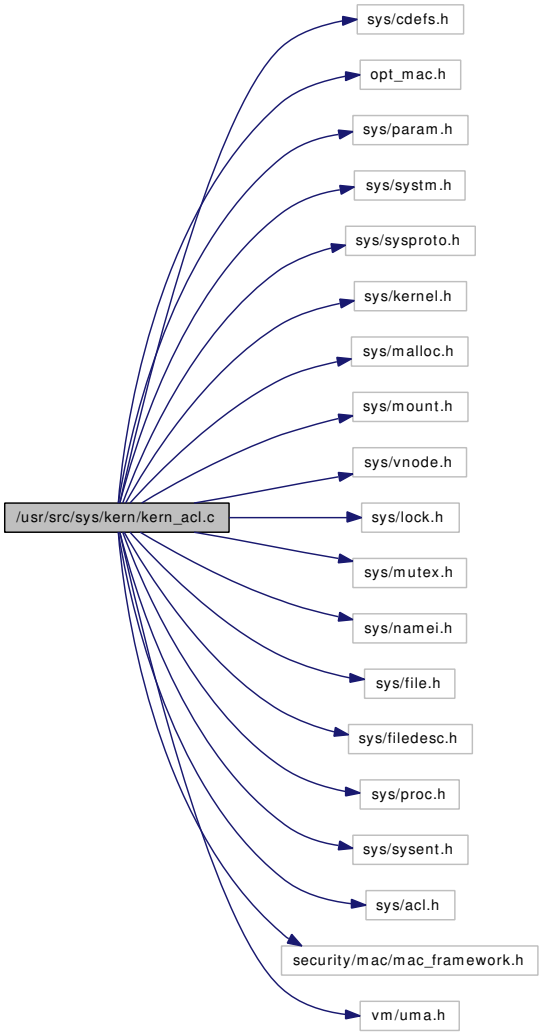
Definition at line 145 of file kern_acct.c.

Referenced by acctwatch().

9.16 /usr/src/sys/kern/kern_acl.c File Reference

```
#include <sys/cdefs.h>
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/sysproto.h>
#include <sys/kernel.h>
#include <sys/malloc.h>
#include <sys/mount.h>
#include <sys/vnode.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/namei.h>
#include <sys/file.h>
#include <sys/filedesc.h>
#include <sys/proc.h>
#include <sys/sysent.h>
#include <sys/acl.h>
#include <security/mac/mac_framework.h>
#include <vm/uma.h>
```

Include dependency graph for kern_acl.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_acl.c,v 1.50 2007/01/12 22:01:03 rwatson Exp $")
- static int vacl_set_acl (struct thread *td, struct vnode *vp, acl_type_t type, struct acl *aclp)
- static int vacl_get_acl (struct thread *td, struct vnode *vp, acl_type_t type, struct acl *aclp)
- static int vacl_aclcheck (struct thread *td, struct vnode *vp, acl_type_t type, struct acl *aclp)
- static int vacl_delete (struct thread *td, struct vnode *vp, acl_type_t type)
- int __acl_get_file (struct thread *td, struct __acl_get_file_args *uap)
- int __acl_get_link (struct thread *td, struct __acl_get_link_args *uap)
- int __acl_set_file (struct thread *td, struct __acl_set_file_args *uap)
- int __acl_set_link (struct thread *td, struct __acl_set_link_args *uap)
- int __acl_get_fd (struct thread *td, struct __acl_get_fd_args *uap)
- int __acl_set_fd (struct thread *td, struct __acl_set_fd_args *uap)
- int __acl_delete_file (struct thread *td, struct __acl_delete_file_args *uap)
- int __acl_delete_link (struct thread *td, struct __acl_delete_link_args *uap)
- int __acl_delete_fd (struct thread *td, struct __acl_delete_fd_args *uap)
- int __acl_aclcheck_file (struct thread *td, struct __acl_aclcheck_file_args *uap)

- int __acl_aclcheck_link (struct thread *td, struct __acl_aclcheck_link_args *uap)
- int __acl_aclcheck_fd (struct thread *td, struct __acl_aclcheck_fd_args *uap)
- static void aclinit (void *dummy __unused)

Variables

- uma_zone_t acl_zone

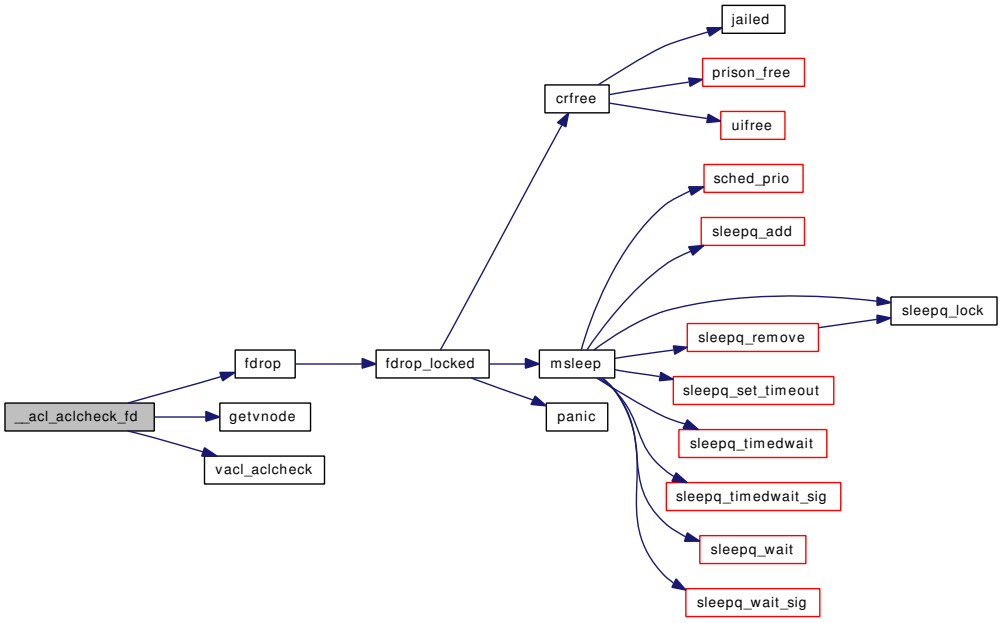
9.16.1 Function Documentation

9.16.1.1 int __acl_aclcheck_fd (struct thread * *td*, struct __acl_aclcheck_fd_args * *uap*)

Definition at line 431 of file kern_acl.c.

References fdrop(), getvnode(), and vacl_aclcheck().

Here is the call graph for this function:

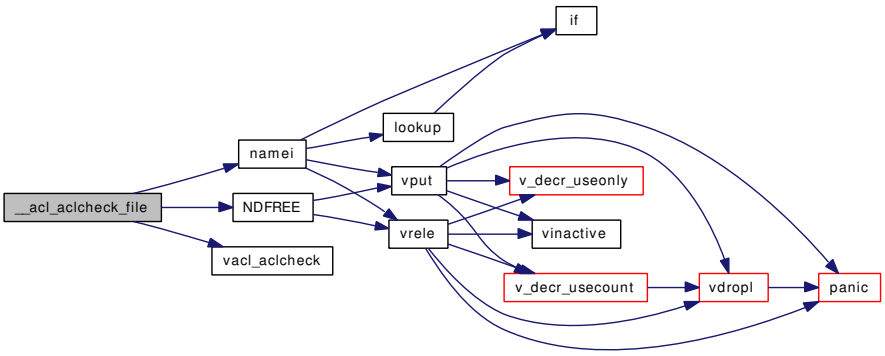


9.16.1.2 int __acl_aclcheck_file (struct thread * *td*, struct __acl_aclcheck_file_args * *uap*)

Definition at line 387 of file kern_acl.c.

References namei(), NDFREE(), and vacl_aclcheck().

Here is the call graph for this function:

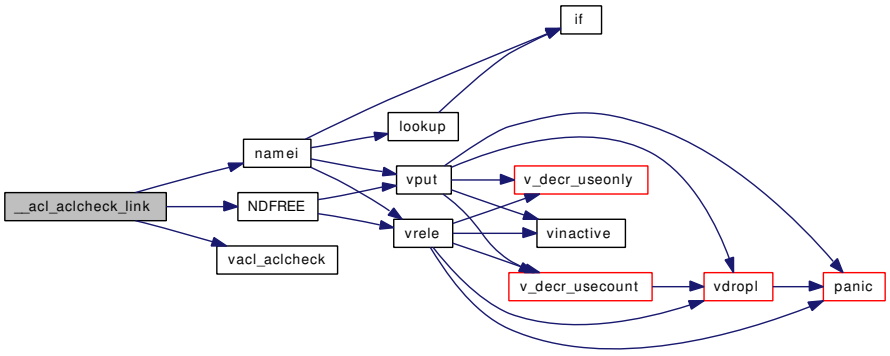


9.16.1.3 int __acl_aclcheck_link (struct thread * *td*, struct __acl_aclcheck_link_args * *uap*)

Definition at line 409 of file kern_acl.c.

References namei(), NDFREE(), and vacl_aclcheck().

Here is the call graph for this function:

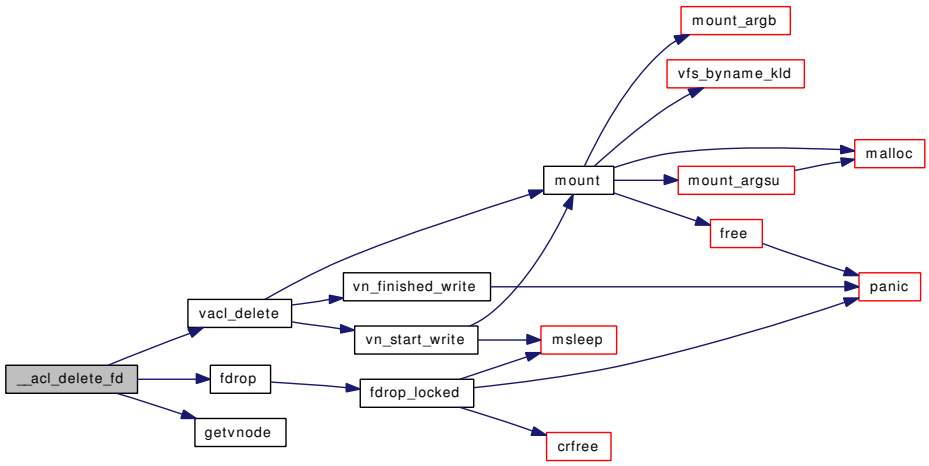


9.16.1.4 int __acl_delete_fd (struct thread * *td*, struct __acl_delete_fd_args * *uap*)

Definition at line 366 of file kern_acl.c.

References fdrop(), getvnode(), and vacl_delete().

Here is the call graph for this function:

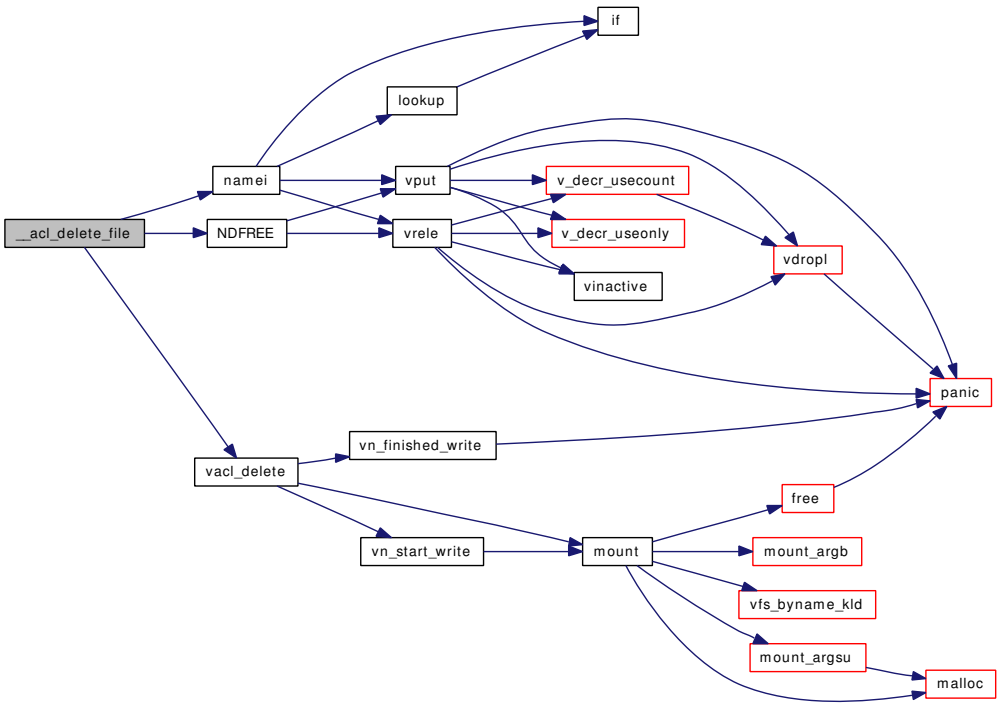


9.16.1.5 int __acl_delete_file (struct thread * *td*, struct __acl_delete_file_args * *uap*)

Definition at line 322 of file kern_acl.c.

References namei(), NDFREE(), and vacl_delete().

Here is the call graph for this function:

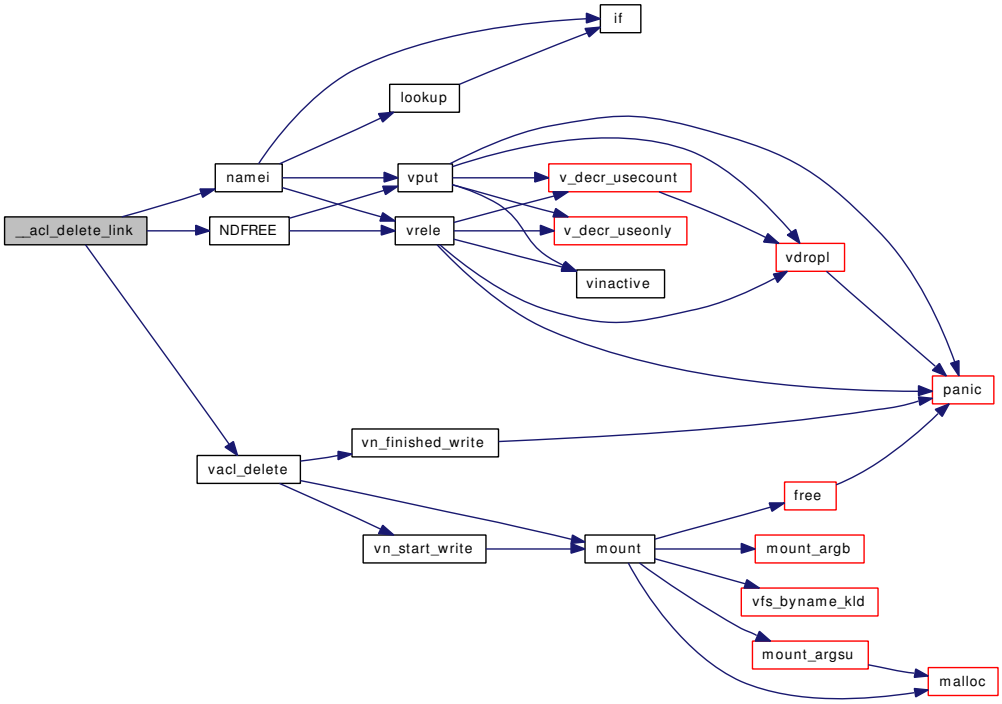


9.16.1.6 int __acl_delete_link (struct thread * *td*, struct __acl_delete_link_args * *uap*)

Definition at line 344 of file kern_acl.c.

References namei(), NDFREE(), and vacl_delete().

Here is the call graph for this function:

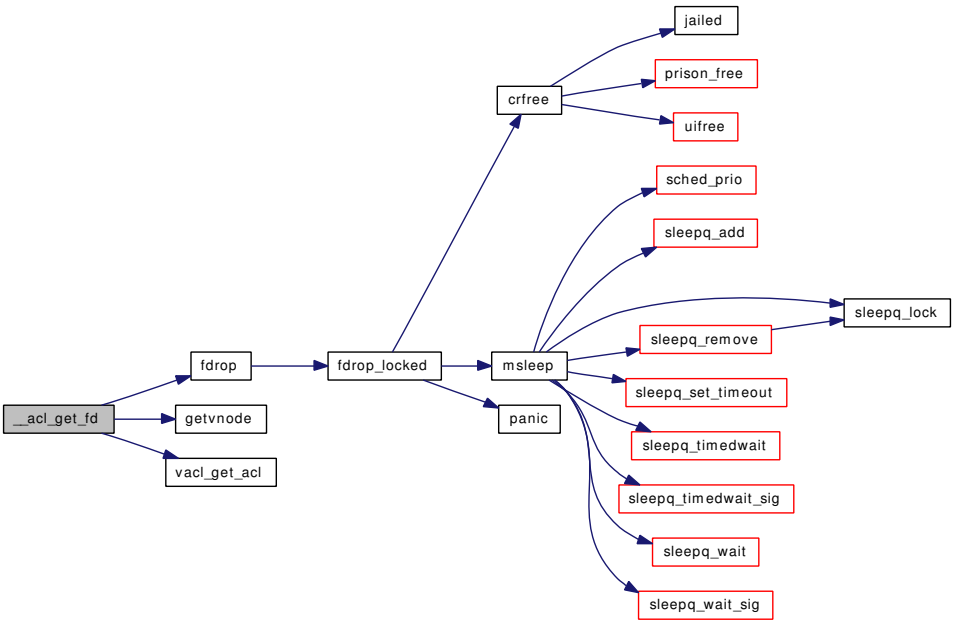


9.16.1.7 int __acl_get_fd (struct thread * *td*, struct __acl_get_fd_args * *uap*)

Definition at line 280 of file kern_acl.c.

References fdrop(), getvnode(), and vacl_get_acl().

Here is the call graph for this function:

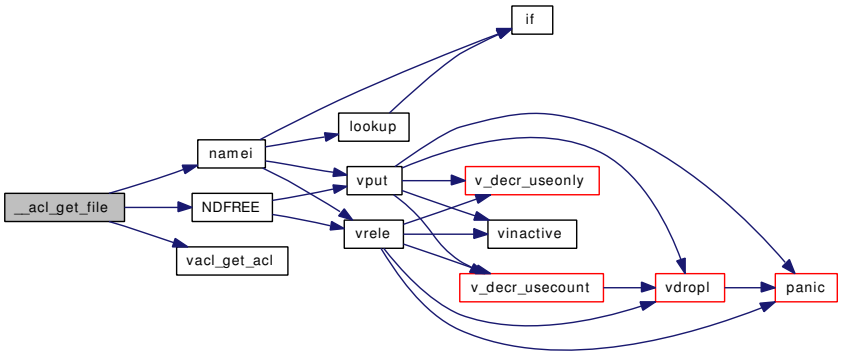


9.16.1.8 int __acl_get_file (struct thread * *td*, struct __acl_get_file_args * *uap*)

Definition at line 192 of file kern_acl.c.

References namei(), NDFREE(), and vacl_get_acl().

Here is the call graph for this function:

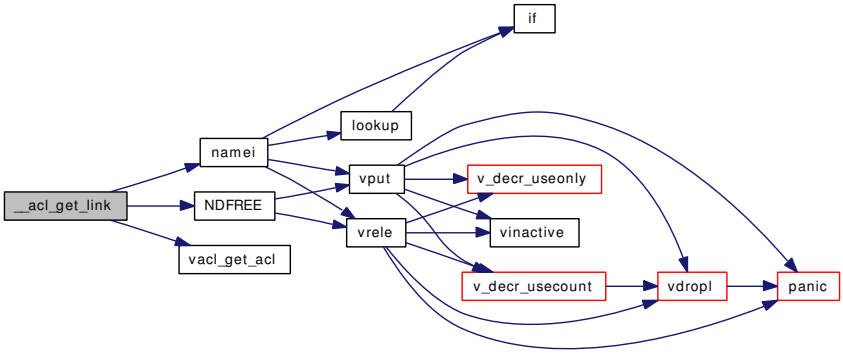


9.16.1.9 int __acl_get_link (struct thread * *td*, struct __acl_get_link_args * *uap*)

Definition at line 214 of file kern_acl.c.

References namei(), NDFREE(), and vacl_get_acl().

Here is the call graph for this function:

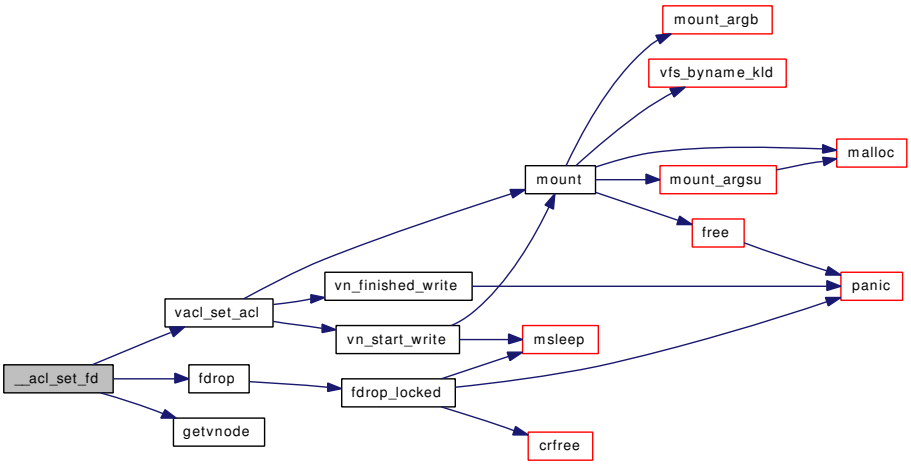


9.16.1.10 int __acl_set_fd (struct thread * *td*, struct __acl_set_fd_args * *uap*)

Definition at line 301 of file kern_acl.c.

References fdrop(), getvnode(), and vacl_set_acl().

Here is the call graph for this function:

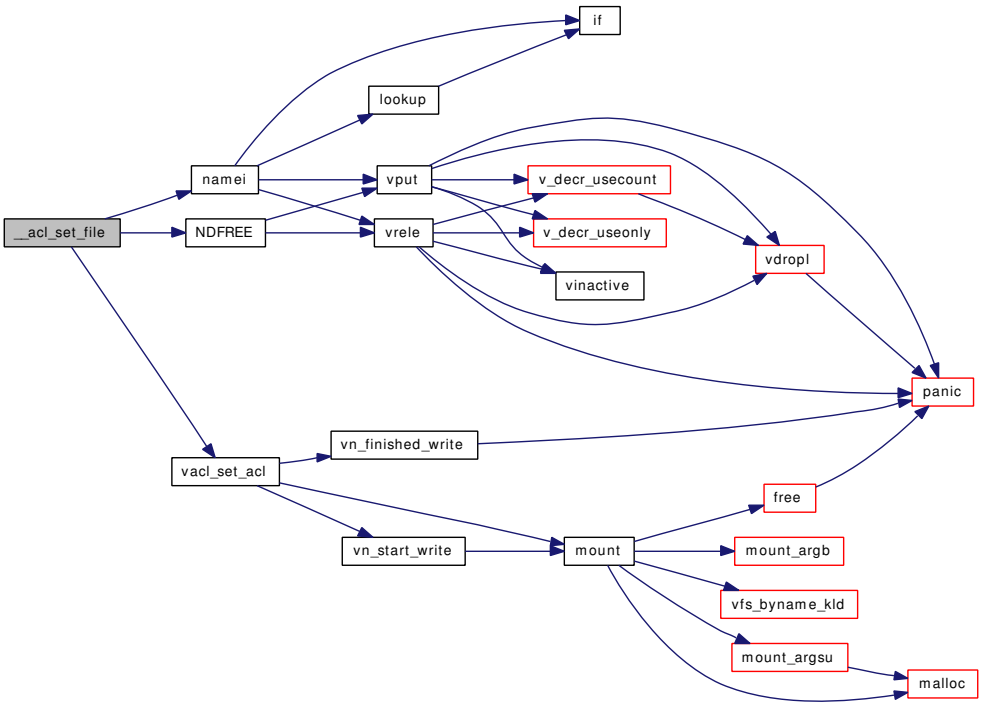


9.16.1.11 int __acl_set_file (struct thread * *td*, struct __acl_set_file_args * *uap*)

Definition at line 236 of file kern_acl.c.

References namei(), NDFREE(), and vacl_set_acl().

Here is the call graph for this function:

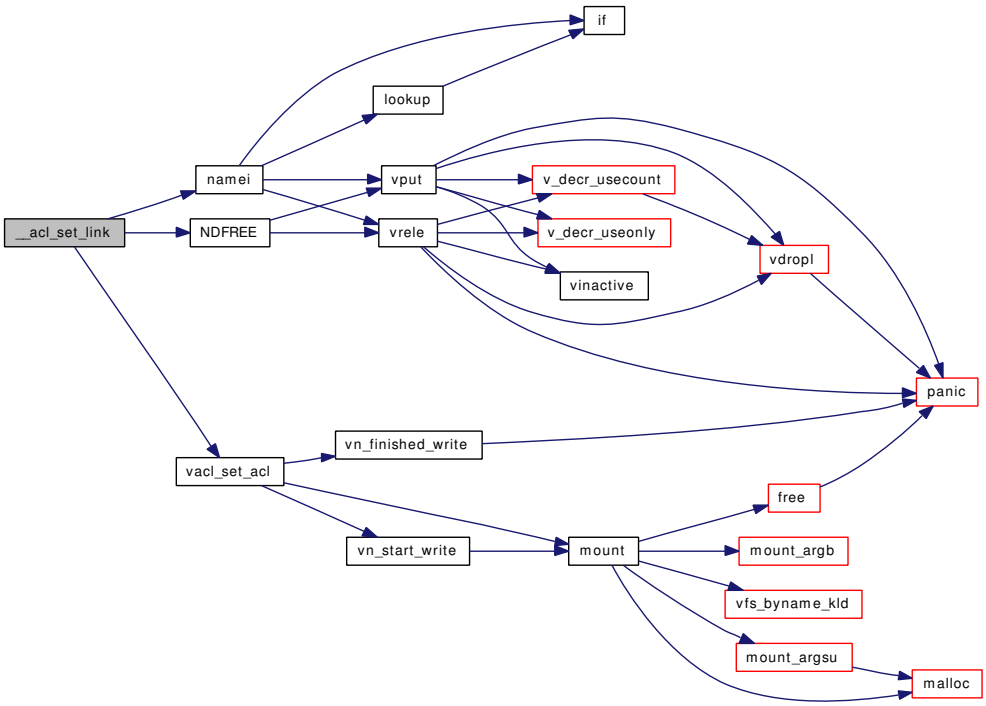


9.16.1.12 int __acl_set_link (struct thread * *td*, struct __acl_set_link_args * *uap*)

Definition at line 258 of file kern_acl.c.

References namei(), NDFREE(), and vacl_set_acl().

Here is the call graph for this function:



9.16.1.13 __FBSDID ("$FreeBSD: src/sys/kern/kern_acl. *c*, v 1.50 2007/01/12 22:01:03 rwatson Exp $")

9.16.1.14 static void aclinit (void ∗dummy *__unused*) [static]

Definition at line 449 of file kern_acl.c.

References acl_zone.

9.16.1.15 static int vacl_aclcheck (struct thread ∗ *td*, struct vnode ∗ *vp*, acl_type_t *type*, struct acl ∗ *aclp*) [static]

Definition at line 168 of file kern_acl.c.

Referenced by __acl_aclcheck_fd(), __acl_aclcheck_file(), and __acl_aclcheck_link().

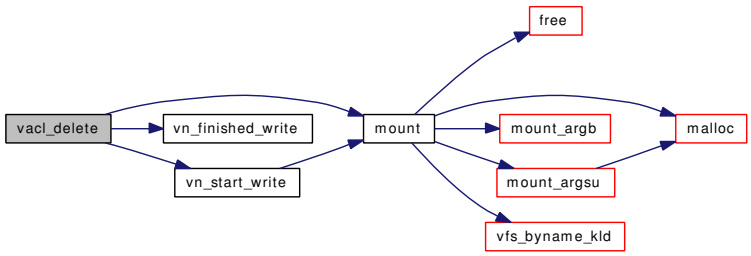
9.16.1.16 static int vacl_delete (struct thread ∗ *td*, struct vnode ∗ *vp*, acl_type_t *type*) [static]

Definition at line 140 of file kern_acl.c.

References mount(), vn_finished_write(), and vn_start_write().

Referenced by __acl_delete_fd(), __acl_delete_file(), and __acl_delete_link().

Here is the call graph for this function:



9.16.1.17 static int vacl_get_acl (struct thread * *td*, struct vnode * *vp*, acl_type_t *type*, struct acl * *aclp*) [static]

Definition at line 113 of file kern_acl.c.

Referenced by __acl_get_fd(), __acl_get_file(), and __acl_get_link().

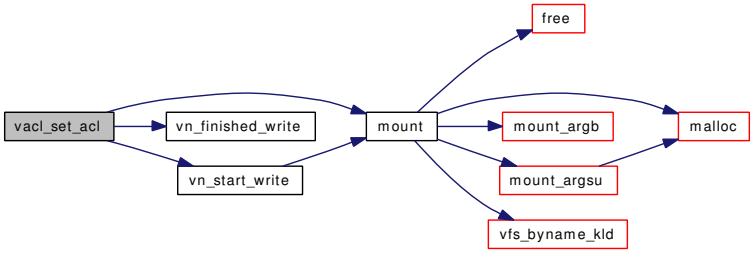
9.16.1.18 static int vacl_set_acl (struct thread * *td*, struct vnode * *vp*, acl_type_t *type*, struct acl * *aclp*) [static]

Definition at line 80 of file kern_acl.c.

References mount(), vn_finished_write(), and vn_start_write().

Referenced by __acl_set_fd(), __acl_set_file(), and __acl_set_link().

Here is the call graph for this function:



9.16.2 Variable Documentation

9.16.2.1 uma_zone_t acl_zone

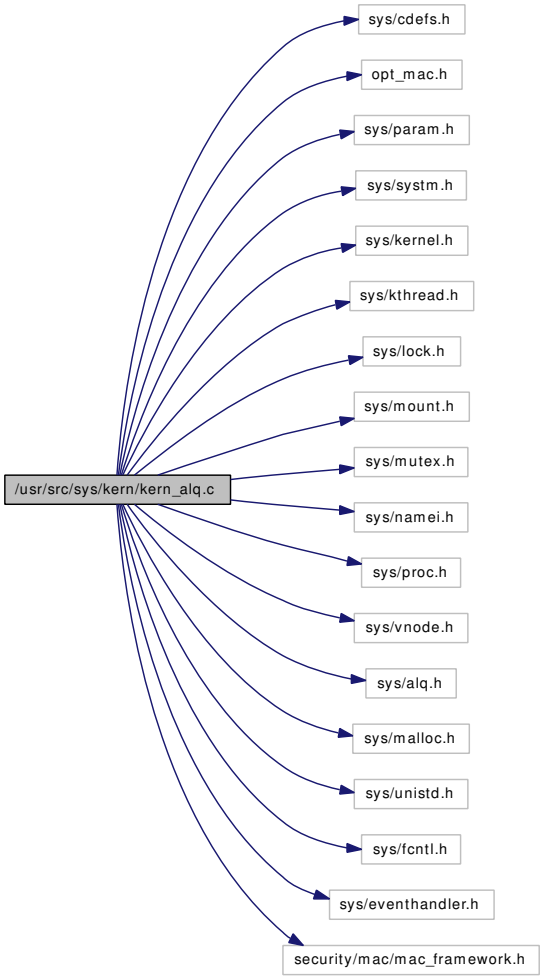
Definition at line 60 of file kern_acl.c.

Referenced by aclinit().

9.17 /usr/src/sys/kern/kern_alq.c File Reference

```
#include <sys/cdefs.h>
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/kthread.h>
#include <sys/lock.h>
#include <sys/mount.h>
#include <sys/mutex.h>
#include <sys/namei.h>
#include <sys/proc.h>
#include <sys/vnode.h>
#include <sys/alq.h>
#include <sys/malloc.h>
#include <sys/unistd.h>
#include <sys/fcntl.h>
#include <sys/eventhandler.h>
#include <security/mac/mac_framework.h>
```

Include dependency graph for kern_alq.c:



Data Structures

- struct alq

Defines

- #define AQ_WANTED 0x0001
- #define AQ_ACTIVE 0x0002
- #define AQ_FLUSHING 0x0004
- #define AQ_SHUTDOWN 0x0008
- #define ALQ_LOCK(alq) mtx_lock_spin(&(alq) → aq_mtx)
- #define ALQ_UNLOCK(alq) mtx_unlock_spin(&(alq) → aq_mtx)
- #define ALD_LOCK() mtx_lock(&ald_mtx)
- #define ALD_UNLOCK() mtx_unlock(&ald_mtx)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_alq.c,v 1.16 2006/10/22 11:52:12 rwatson Exp $")

- static MALLOC_DEFINE (M_ALD,"ALD","ALD")
- static LIST_HEAD (alq)
- static int ald_rem (struct alq *alq)
- static void ald_activate (struct alq *alq)
- static void ald_deactivate (struct alq *alq)
- static void ald_startup (void *unused)
- static void ald_daemon (void)
- static void ald_shutdown (void *arg, int howto)
- static void alq_shutdown (struct alq *alq)
- static int alq_doio (struct alq *alq)
- int alq_open (struct alq **alqp, const char *file, struct ucred *cred, int cmode, int size, int count)
- int alq_write (struct alq *alq, void *data, int waitok)
- ale * alq_get (struct alq *alq, int waitok)
- void alq_post (struct alq *alq, struct ale *ale)
- void alq_flush (struct alq *alq)
- void alq_close (struct alq *alq)

Variables

- static struct mtx ald_mtx
- static struct kproc_desc ald_kp

9.17.1 Define Documentation

9.17.1.1 #define ALD_LOCK() mtx_lock(&ald_mtx)

Referenced by ald_daemon(), ald_rem(), ald_shutdown(), alq_flush(), alq_post(), and LIST_HEAD().

9.17.1.2 #define ALD_UNLOCK() mtx_unlock(&ald_mtx)

Referenced by ald_daemon(), ald_rem(), ald_shutdown(), alq_flush(), and alq_post().

9.17.1.3 #define ALQ_LOCK(alq) mtx_lock_spin(&(alq) → aq_mtx)

Definition at line 71 of file kern_alq.c.

Referenced by ald_daemon(), alq_doio(), alq_flush(), alq_get(), and alq_shutdown().

9.17.1.4 #define ALQ_UNLOCK(alq) mtx_unlock_spin(&(alq) → aq_mtx)

Definition at line 72 of file kern_alq.c.

Referenced by ald_daemon(), alq_doio(), alq_flush(), alq_get(), alq_post(), and alq_shutdown().

9.17.1.5 #define AQ_ACTIVE 0x0002

Definition at line 67 of file kern_alq.c.

Referenced by ald_deactivate(), alq_flush(), alq_post(), and alq_shutdown().

9.17.1.6 #define AQ_FLUSHING 0x0004

Definition at line 68 of file kern_alq.c.

Referenced by alq_doio(), and alq_shutdown().

9.17.1.7 #define AQ_SHUTDOWN 0x0008

Definition at line 69 of file kern_alq.c.

Referenced by alq_get(), and alq_shutdown().

9.17.1.8 #define AQ_WANTED 0x0001

Definition at line 66 of file kern_alq.c.

Referenced by alq_doio(), alq_get(), and alq_shutdown().

9.17.2 Function Documentation

9.17.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_alq. *c*, v 1.16 2006/10/22 11:52:12 rwatson Exp $")

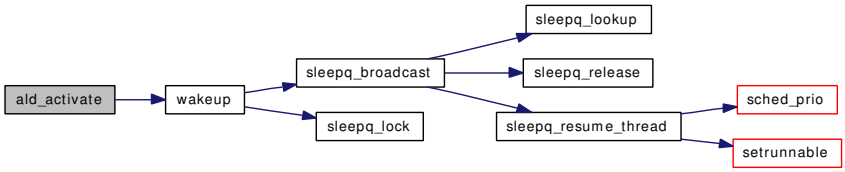
9.17.2.2 static void ald_activate (struct alq * *alq*) [static]

Definition at line 150 of file kern_alq.c.

References wakeup().

Referenced by alq_post().

Here is the call graph for this function:

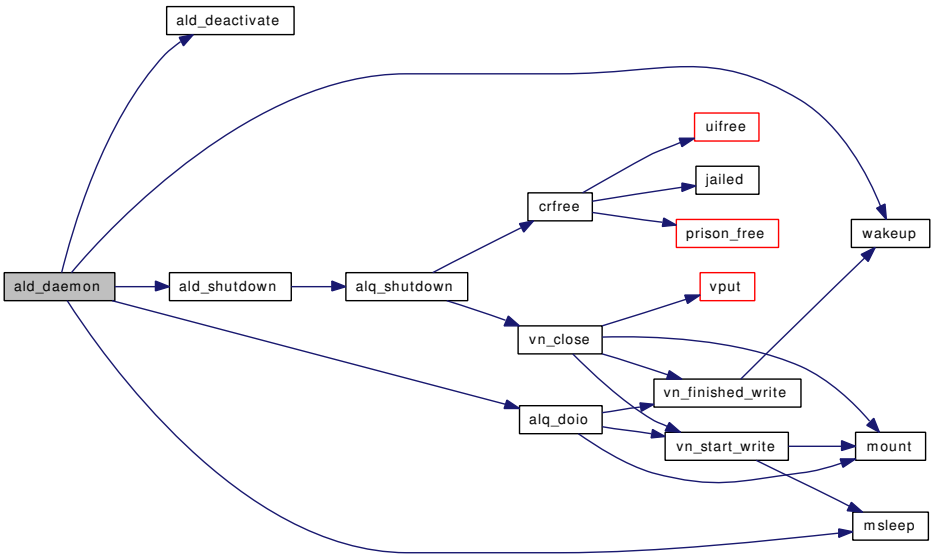


9.17.2.3 static void ald_daemon (void) [static]

Definition at line 172 of file kern_alq.c.

References ald_deactivate(), ALD_LOCK, ald_shutdown(), ALD_UNLOCK, alq_doio(), ALQ_LOCK, ALQ_UNLOCK, msleep(), and wakeup().

Here is the call graph for this function:



9.17.2.4 static void ald_deactivate (struct alq * *alq*) [static]

Definition at line 157 of file kern_alq.c.

References AQ_ACTIVE, and alq::aq_flags.

Referenced by ald_daemon(), and alq_flush().

9.17.2.5 static int ald_rem (struct alq * *alq*) [static]

Definition at line 129 of file kern_alq.c.

References ALD_LOCK, and ALD_UNLOCK.

Referenced by alq_close().

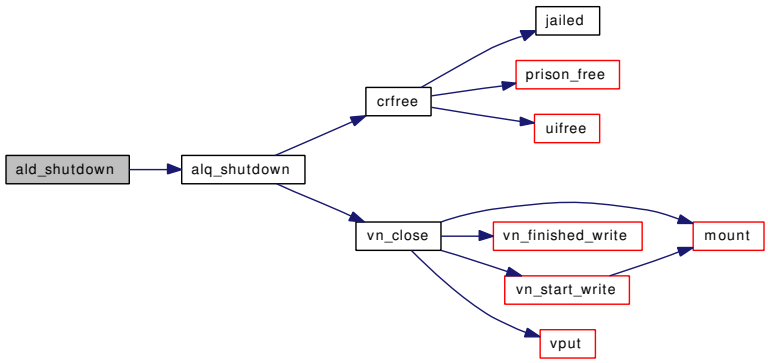
9.17.2.6 static void ald_shutdown (void * *arg*, int *howto*) [static]

Definition at line 200 of file kern_alq.c.

References ALD_LOCK, ALD_UNLOCK, and alq_shutdown().

Referenced by ald_daemon().

Here is the call graph for this function:

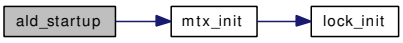


9.17.2.7 static void ald_startup (void * *unused*) [static]

Definition at line 164 of file kern_alq.c.

References mtx_init().

Here is the call graph for this function:

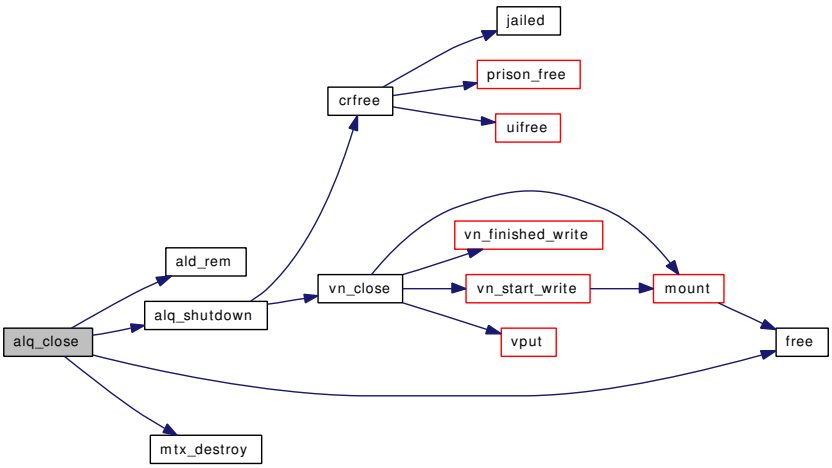


9.17.2.8 void alq_close (struct alq * *alq*)

Definition at line 501 of file kern_alq.c.

References ald_rem(), alq_shutdown(), alq::aq_entbuf, alq::aq_first, alq::aq_mtx, free(), and mtx_destroy().

Here is the call graph for this function:



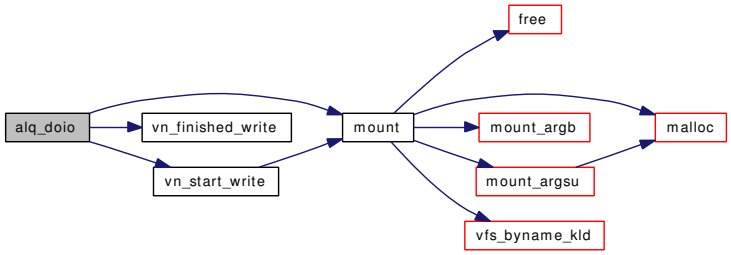
9.17.2.9 static int alq_doio (struct alq * *alq*) [static]

Definition at line 242 of file kern_alq.c.

References ALQ_LOCK, ALQ_UNLOCK, alq::aq_cred, alq::aq_entfree, alq::aq_entlen, alq::aq_entvalid, alq::aq_flags, AQ_FLUSHING, alq::aq_vp, AQ_WANTED, mount(), td, vn_finished_write(), and vn_start_write().

Referenced by ald_daemon(), and alq_flush().

Here is the call graph for this function:

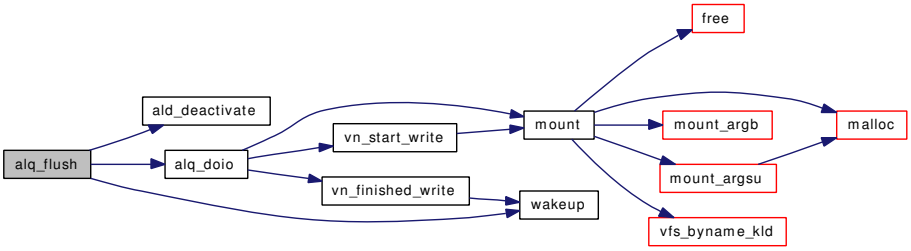


9.17.2.10 void alq_flush (struct alq * *alq*)

Definition at line 479 of file kern_alq.c.

References ald_deactivate(), ALD_LOCK, ALD_UNLOCK, alq_doio(), ALQ_LOCK, ALQ_UNLOCK, AQ_ACTIVE, alq::aq_flags, and wakeup().

Here is the call graph for this function:



9.17.2.11 struct ale* alq_get (struct alq * *alq*, int *waitok*)

Definition at line 422 of file kern_alq.c.

References ALQ_LOCK, ALQ_UNLOCK, alq::aq_entfree, alq::aq_flags, AQ_SHUTDOWN, and AQ_WANTED.

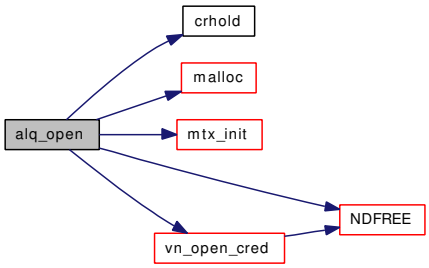
Referenced by alq_write(), and ktr_tracepoint().

9.17.2.12 int alq_open (struct alq ** *alqp*, const char * *file*, struct ucred * *cred*, int *cmode*, int *size*, int *count*)

Definition at line 340 of file kern_alq.c.

References crhold(), malloc(), mtx_init(), NDFREE(), td, and vn_open_cred().

Here is the call graph for this function:



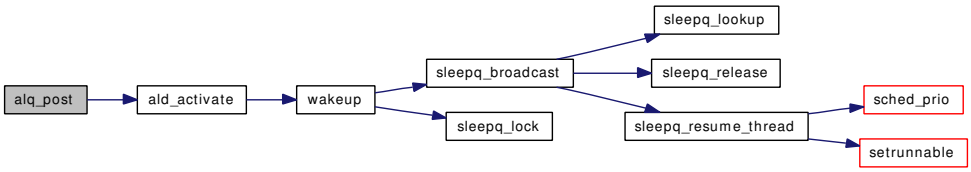
9.17.2.13 void alq_post (struct alq * *alq*, struct ale * *ale*)

Definition at line 455 of file kern_alq.c.

References ald_activate(), ALD_LOCK, ALD_UNLOCK, ALQ_UNLOCK, AQ_ACTIVE, alq::aq_entvalid, and alq::aq_flags.

Referenced by alq_write(), and ktr_tracepoint().

Here is the call graph for this function:



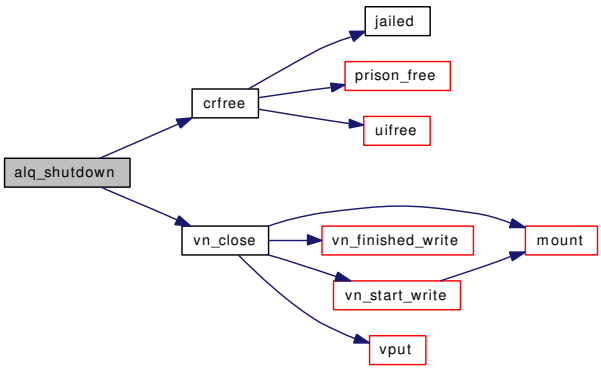
9.17.2.14 static void alq_shutdown (struct alq * *alq*) [static]

Definition at line 217 of file kern_alq.c.

References ALQ_LOCK, ALQ_UNLOCK, AQ_ACTIVE, alq::aq_cred, alq::aq_flags, AQ_FLUSHING, AQ_SHUTDOWN, alq::aq_vp, AQ_WANTED, crfree(), and vn_close().

Referenced by ald_shutdown(), and alq_close().

Here is the call graph for this function:

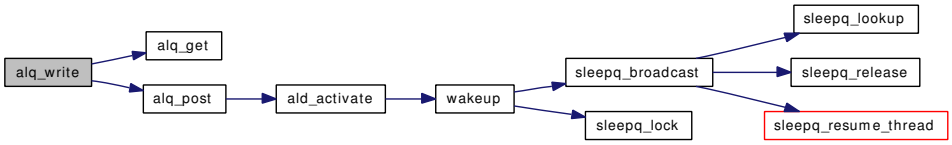


9.17.2.15 int alq_write (struct alq * *alq*, void * *data*, int *waitok*)

Definition at line 408 of file kern_alq.c.

References alq_get(), alq_post(), and alq::aq_entlen.

Here is the call graph for this function:



9.17.2.16 static LIST_HEAD (alq) [static]

Definition at line 80 of file kern_alq.c.

References ALD_LOCK.

Referenced by hashdestroy(), hashinit_flags(), and phashinit().

9.17.2.17 static MALLOC_DEFINE (M_ALD, "ALD", "ALD") [static]

9.17.3 Variable Documentation

9.17.3.1 struct kproc_desc ald_kp [static]

Initial value:

```
{
        "ALQ Daemon",
        ald_daemon,
        &ald_proc
}
```

Definition at line 324 of file kern_alq.c.

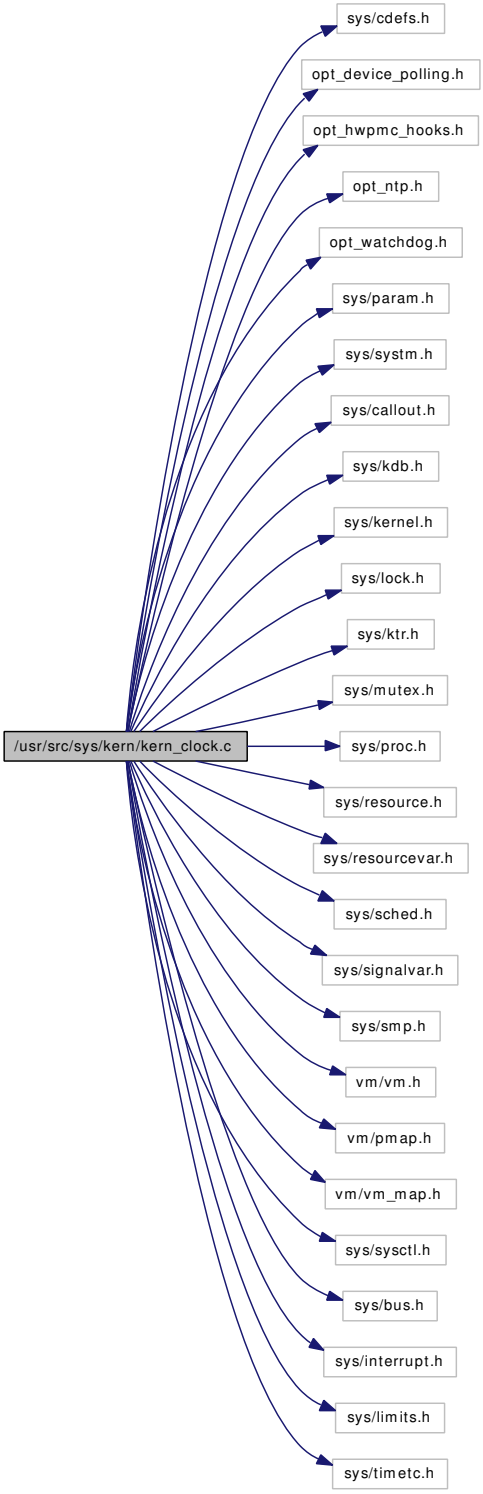
9.17.3.2 struct mtx ald_mtx [static]

Definition at line 79 of file kern_alq.c.

9.18 /usr/src/sys/kern/kern_clock.c File Reference

```
#include <sys/cdefs.h>
#include "opt_device_polling.h"
#include "opt_hwpmc_hooks.h"
#include "opt_ntp.h"
#include "opt_watchdog.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/callout.h>
#include <sys/kdb.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/ktr.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/resource.h>
#include <sys/resourcevar.h>
#include <sys/sched.h>
#include <sys/signalvar.h>
#include <sys/smp.h>
#include <vm/vm.h>
#include <vm/pmap.h>
#include <vm/vm_map.h>
#include <sys/sysctl.h>
#include <sys/bus.h>
#include <sys/interrupt.h>
#include <sys/limits.h>
#include <sys/timetc.h>
```

Include dependency graph for kern_clock.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_clock.c,v 1.193 2006/12/15 21:44:49 n_hibma Exp $")
- static void initclocks (void *dummy)

- static int sysctl_kern_cp_time (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern, OID_AUTO, cp_time, CTLTYPE_LONG|CTLFLAG_RD, 0, 0, sysctl_-kern_cp_time,"LU","CPU time statistics")
- void hardclock_cpu (int usermode)
- void hardclock (int usermode, uintfptr_t pc)
- int tvtohz (struct timeval ∗tv)
- void startprofclock (struct proc ∗p)
- void stopprofclock (struct proc ∗p)
- void statclock (int usermode)
- void profclock (int usermode, uintfptr_t pc)
- static int sysctl_kern_clockrate (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern, KERN_CLOCKRATE, clockrate, CTLTYPE_STRUCT|CTLFLAG_RD, 0, 0, sysctl_kern_clockrate,"S,clockinfo","Rate and period of various kernel clocks")

Variables

- long cp_time [CPUSTATES]
- int stathz
- int profhz
- int profprocs
- int ticks
- int psratio

9.18.1 Function Documentation

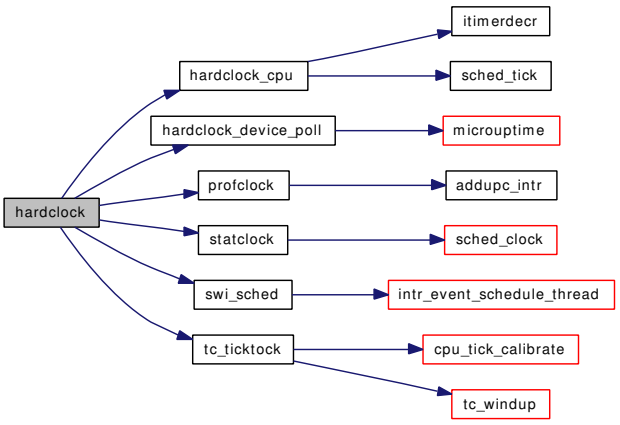
9.18.1.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_clock. *c*, v 1.193 2006/12/15 21:44:49 n_hibma Exp $")

9.18.1.2 void hardclock (int *usermode*, uintfptr_t *pc*)

Definition at line 236 of file kern_clock.c.

References callout_lock, callwheel, callwheelmask, hardclock_cpu(), hardclock_device_poll(), prof-clock(), softclock_ih, softticks, statclock(), stathz, swi_sched(), tc_ticktock(), and ticks.

Here is the call graph for this function:



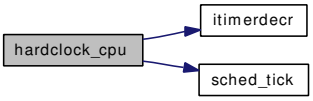
9.18.1.3 void hardclock_cpu (int *usermode*)

Definition at line 194 of file kern_clock.c.

References itimerdecr(), sched_lock, sched_tick(), td, and tick.

Referenced by hardclock().

Here is the call graph for this function:



9.18.1.4 static void initclocks (void * *dummy*) [static]

Definition at line 165 of file kern_clock.c.

References hz, profhz, psratio, and stathz.

9.18.1.5 void profclock (int *usermode*, uintfptr_t *pc*)

Definition at line 479 of file kern_clock.c.

References addupc_intr(), and td.

Referenced by hardclock().

Here is the call graph for this function:

9.18.1.6 void startprofclock (struct proc * *p*)

Definition at line 348 of file kern_clock.c.

References profprocs, and sched_lock.

Referenced by fork1(), and profil().

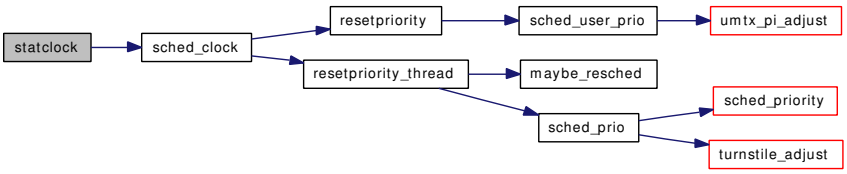
9.18.1.7 void statclock (int *usermode*)

Definition at line 404 of file kern_clock.c.

References cp_time, hz, sched_clock(), sched_lock, stathz, and td.

Referenced by hardclock().

Here is the call graph for this function:



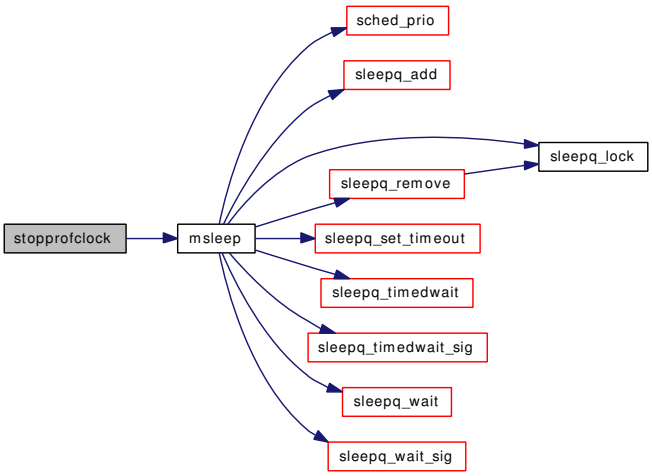
9.18.1.8 void stopprofclock (struct proc * *p*)

Definition at line 373 of file kern_clock.c.

References msleep(), profprocs, and sched_lock.

Referenced by addupc_task(), exit1(), and profil().

Here is the call graph for this function:



9.18.1.9 static int sysctl_kern_clockrate (SYSCTL_HANDLER_ARGS) [static]

Definition at line 518 of file kern_clock.c.

References hz, profhz, stathz, sysctl_handle_opaque(), and tick.

Here is the call graph for this function:

9.18.1.10 static int sysctl_kern_cp_time (SYSCTL_HANDLER_ARGS) [static]

Definition at line 87 of file kern_clock.c.

References cp_time.

9.18.1.11 SYSCTL_PROC (_kern, KERN_CLOCKRATE, clockrate, CTLTYPE_STRUCT| *CTLFLAG_RD*, 0, 0, sysctl_kern_clockrate, " *S*, clockinfo", "Rate and period of various kernel clocks")

9.18.1.12 SYSCTL_PROC (_kern, OID_AUTO, cp_time, CTLTYPE_LONG| *CTLFLAG_RD*, 0, 0, sysctl_kern_cp_time, "LU", "CPU time statistics")

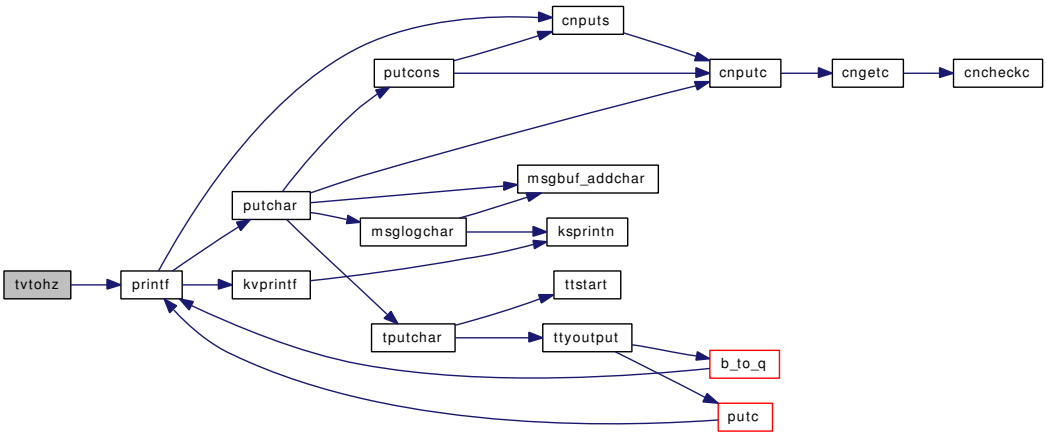
9.18.1.13 int tvtohz (struct timeval * *tv*)

Definition at line 286 of file kern_clock.c.

References hz, printf(), tick, and ticks.

Referenced by aio_suspend(), aio_waitcomplete(), do_cv_wait(), do_lock_umtx(), do_lock_umutex(), do_wait(), kern_nanosleep(), kern_select(), kern_sem_wait(), kern_setitimer(), kern_thr_suspend(), kqueue_scan(), kse_release(), mqueue_receive(), mqueue_send(), poll(), realitexpire(), realtimer_expire(), realtimer_settime(), timertoticks(), and ttread().

Here is the call graph for this function:



9.18.2 Variable Documentation

9.18.2.1 long cp_time[CPUSTATES]

Definition at line 84 of file kern_clock.c.

Referenced by statclock(), and sysctl_kern_cp_time().

9.18.2.2 int profhz

Definition at line 155 of file kern_clock.c.

Referenced by initclocks(), and sysctl_kern_clockrate().

9.18.2.3 int profprocs

Definition at line 156 of file kern_clock.c.

Referenced by startprofclock(), and stopprofclock().

9.18.2.4 int psratio

Definition at line 158 of file kern_clock.c.

Referenced by initclocks(), and userret().

9.18.2.5 int stathz

Definition at line 154 of file kern_clock.c.

Referenced by hardclock(), initclocks(), sched_initticks(), schedcpu(), statclock(), and sysctl_kern_clockrate().

9.18.2.6 int ticks

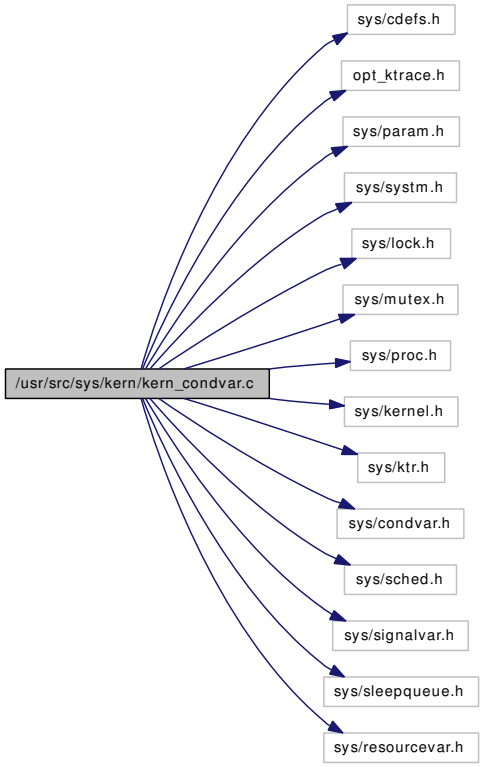
Definition at line 157 of file kern_clock.c.

Referenced by callout_reset(), hardclock(), mi_switch(), ppsratecheck(), proc0_post(), sched_clock(), sched_pctcpu(), sched_pctcpu_update(), sched_sleep(), sched_tick(), sched_wakeup(), schedinit(), softclock(), tdq_runq_rem(), thread_exit(), tvtohz(), and uiomove().

9.19 /usr/src/sys/kern/kern_condvar.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ktrace.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/kernel.h>
#include <sys/ktr.h>
#include <sys/condvar.h>
#include <sys/sched.h>
#include <sys/signalvar.h>
#include <sys/sleepqueue.h>
#include <sys/resourcevar.h>
```

Include dependency graph for kern_condvar.c:



Defines

- #define CV_ASSERT(cvp, mp, td)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_condvar.c,v 1.57 2006/12/16 06:54:08 kmacy Exp $")
- void cv_init (struct cv *cvp, const char *desc)
- void cv_destroy (struct cv *cvp)
- void cv_wait (struct cv *cvp, struct mtx *mp)
- void cv_wait_unlock (struct cv *cvp, struct mtx *mp)
- int cv_wait_sig (struct cv *cvp, struct mtx *mp)
- int cv_timedwait (struct cv *cvp, struct mtx *mp, int timo)
- int cv_timedwait_sig (struct cv *cvp, struct mtx *mp, int timo)
- void cv_signal (struct cv *cvp)
- void cv_broadcastpri (struct cv *cvp, int pri)

9.19.1 Define Documentation

9.19.1.1 #define CV_ASSERT(cvp, mp, td)

Value:

```
do {                                    \
        KASSERT((td) != NULL, ("%s: curthread NULL", __func__));        \
        KASSERT(TD_IS_RUNNING(td), ("%s: not TDS_RUNNING", __func__));  \
        KASSERT((cvp) != NULL, ("%s: cvp NULL", __func__));             \
        KASSERT((mp) != NULL, ("%s: mp NULL", __func__));               \
        mtx_assert((mp), MA_OWNED | MA_NOTRECURSED);                    \
} while (0)
```

Definition at line 52 of file kern_condvar.c.

Referenced by cv_timedwait(), cv_timedwait_sig(), cv_wait_sig(), and cv_wait_unlock().

9.19.2 Function Documentation

9.19.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_condvar. *c*, v 1.57 2006/12/16 06:54:08 kmacy Exp $")

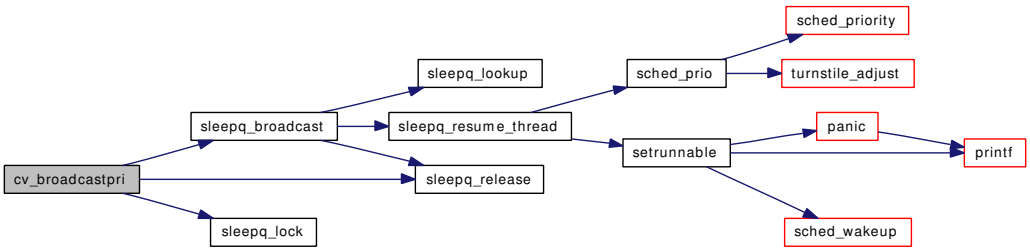
9.19.2.2 void cv_broadcastpri (struct cv * *cvp*, int *pri*)

Definition at line 352 of file kern_condvar.c.

References sleepq_broadcast(), sleepq_lock(), and sleepq_release().

Referenced by doselwakeup().

Here is the call graph for this function:



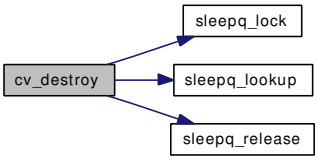
9.19.2.3 void cv_destroy (struct cv * *cvp*)

Definition at line 76 of file kern_condvar.c.

References sleepq_lock(), sleepq_lookup(), and sleepq_release().

Referenced by sem_free(), sema_destroy(), and sx_destroy().

Here is the call graph for this function:



9.19.2.4 void cv_init (struct cv * *cvp*, const char * *desc*)

Definition at line 64 of file kern_condvar.c.

Referenced by devinit(), selectinit(), sem_create(), sema_init(), and sx_init().

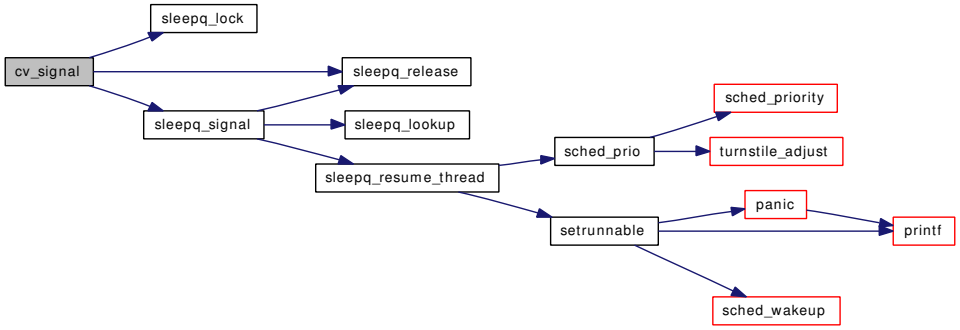
9.19.2.5 void cv_signal (struct cv * *cvp*)

Definition at line 336 of file kern_condvar.c.

References sleepq_lock(), sleepq_release(), and sleepq_signal().

Referenced by _sema_post(), _sx_sunlock(), _sx_xunlock(), and kern_sem_post().

Here is the call graph for this function:



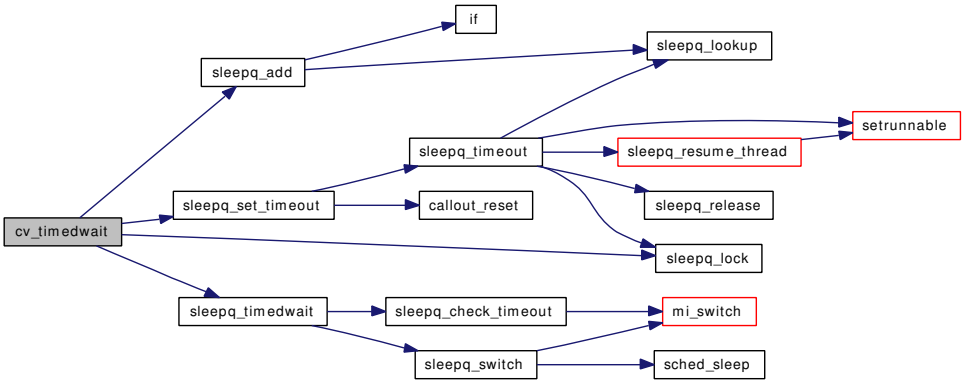
9.19.2.6 int cv_timedwait (struct cv * *cvp*, struct mtx * *mp*, int *timo*)

Definition at line 221 of file kern_condvar.c.

References CV_ASSERT, panicstr, sleepq_add(), sleepq_lock(), sleepq_set_timeout(), sleepq_timedwait(), and td.

Referenced by _sema_timedwait().

Here is the call graph for this function:



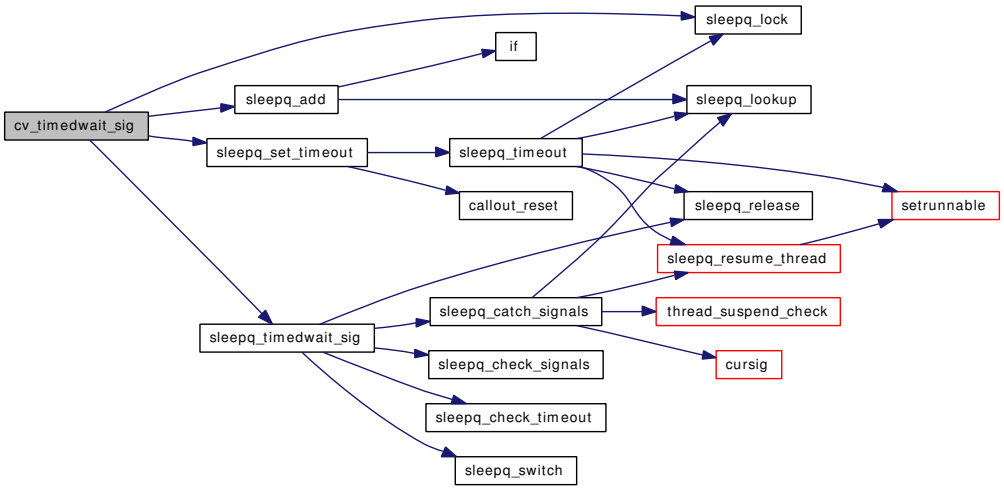
9.19.2.7 int cv_timedwait_sig (struct cv * *cvp*, struct mtx * *mp*, int *timo*)

Definition at line 277 of file kern_condvar.c.

References CV_ASSERT, panicstr, sleepq_add(), sleepq_lock(), sleepq_set_timeout(), sleepq_timedwait_-sig(), and td.

Referenced by kern_select(), kern_sem_wait(), and poll().

Here is the call graph for this function:



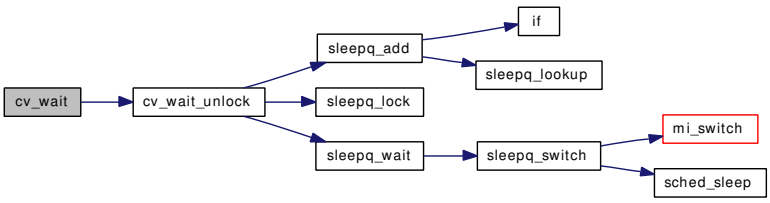
9.19.2.8 void cv_wait (struct cv * *cvp*, struct mtx * *mp*)

Definition at line 96 of file kern_condvar.c.

References cv_wait_unlock(), and panicstr.

Referenced by _sema_wait(), _sx_slock(), and _sx_xlock().

Here is the call graph for this function:



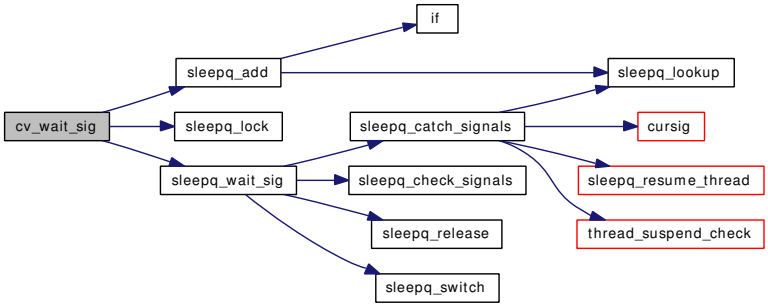
9.19.2.9 int cv_wait_sig (struct cv * *cvp*, struct mtx * *mp*)

Definition at line 166 of file kern_condvar.c.

References CV_ASSERT, panicstr, sleepq_add(), sleepq_lock(), sleepq_wait_sig(), and td.

Referenced by devread(), kern_select(), kern_sem_wait(), and poll().

Here is the call graph for this function:



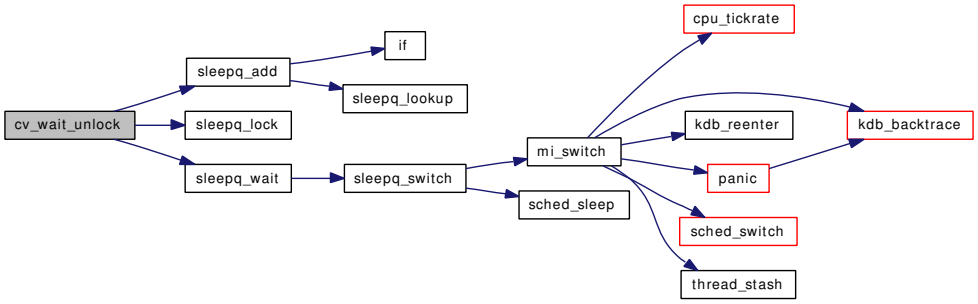
9.19.2.10 void cv_wait_unlock (struct cv * *cvp*, struct mtx * *mp*)

Definition at line 122 of file kern_condvar.c.

References CV_ASSERT, panicstr, sleepq_add(), sleepq_lock(), sleepq_wait(), and td.

Referenced by cv_wait().

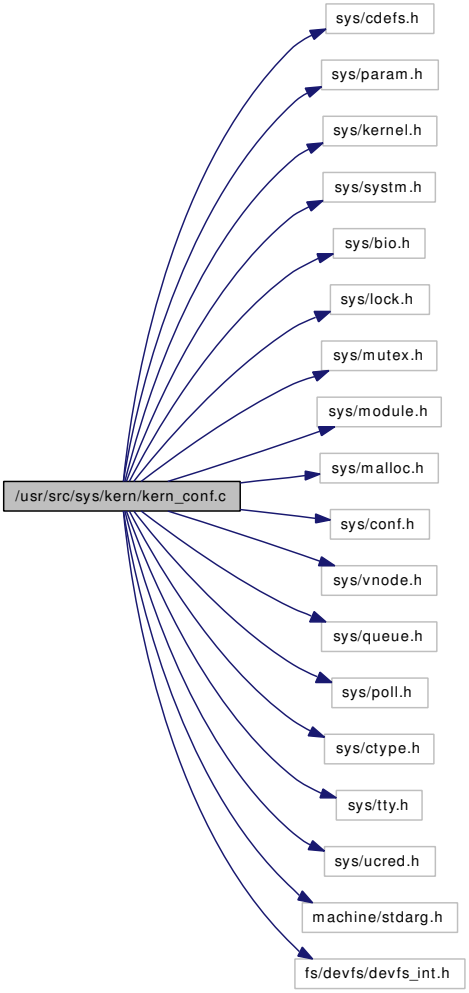
Here is the call graph for this function:



9.20 /usr/src/sys/kern/kern_conf.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/kernel.h>
#include <sys/systm.h>
#include <sys/bio.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/module.h>
#include <sys/malloc.h>
#include <sys/conf.h>
#include <sys/vnode.h>
#include <sys/queue.h>
#include <sys/poll.h>
#include <sys/ctype.h>
#include <sys/tty.h>
#include <sys/ucred.h>
#include <machine/stdarg.h>
#include <fs/devfs/devfs_int.h>
```

Include dependency graph for kern_conf.c:



Data Structures

- struct clonedevs

Defines

- #define dead_open (d_open_t *)enxio
- #define dead_close (d_close_t *)enxio
- #define dead_read (d_read_t *)enxio
- #define dead_write (d_write_t *)enxio
- #define dead_ioctl (d_ioctl_t *)enxio
- #define dead_poll (d_poll_t *)enodev
- #define dead_mmap (d_mmap_t *)enodev
- #define dead_dump (dumper_t *)enxio
- #define dead_kqfilter (d_kqfilter_t *)enxio
- #define null_open (d_open_t *)nullop
- #define null_close (d_close_t *)nullop
- #define no_read (d_read_t *)enodev

- #define no_write (d_write_t *)enodev
- #define no_ioctl (d_ioctl_t *)enodev
- #define no_mmap (d_mmap_t *)enodev
- #define no_kqfilter (d_kqfilter_t *)enodev
- #define no_dump (dumper_t *)enodev
- #define FIXUP(member, noop, giant)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_conf.c,v 1.201 2007/02/02 22:27:45 bms Exp $")
- static MALLOC_DEFINE (M_DEVT,"cdev","cdev storage")
- static void destroy_devl (struct cdev *dev)
- static struct cdev * make_dev_credv (struct cdevsw *devsw, int minornr, struct ucred *cr, uid_t uid, gid_t gid, int mode, const char *fmt, va_list ap)
- void dev_lock (void)
- void dev_unlock (void)
- void dev_ref (struct cdev *dev)
- void dev_refl (struct cdev *dev)
- void dev_rel (struct cdev *dev)
- cdevsw * dev_refthread (struct cdev *dev)
- cdevsw * devvn_refthread (struct vnode *vp, struct cdev **devp)
- void dev_relthread (struct cdev *dev)
- int nullop (void)
- int eopnotsupp (void)
- static int enxio (void)
- static int enodev (void)
- static void dead_strategy (struct bio *bp)
- static void no_strategy (struct bio *bp)
- static int no_poll (struct cdev *dev __unused, int events, struct thread *td __unused)
- static int giant_open (struct cdev *dev, int oflags, int devtype, struct thread *td)
- static int giant_fdopen (struct cdev *dev, int oflags, struct thread *td, int fdidx)
- static int giant_close (struct cdev *dev, int fflag, int devtype, struct thread *td)
- static void giant_strategy (struct bio *bp)
- static int giant_ioctl (struct cdev *dev, u_long cmd, caddr_t data, int fflag, struct thread *td)
- static int giant_read (struct cdev *dev, struct uio *uio, int ioflag)
- static int giant_write (struct cdev *dev, struct uio *uio, int ioflag)
- static int giant_poll (struct cdev *dev, int events, struct thread *td)
- static int giant_kqfilter (struct cdev *dev, struct knote *kn)
- static int giant_mmap (struct cdev *dev, vm_offset_t offset, vm_paddr_t *paddr, int nprot)
- int minor (struct cdev *x)
- int dev2unit (struct cdev *x)
- u_int minor2unit (u_int _minor)
- int unit2minor (int unit)
- static struct cdev * newdev (struct cdevsw *csw, int y, struct cdev *si)
- int uminor (dev_t dev)
- int umajor (dev_t dev)
- static void fini_cdevsw (struct cdevsw *devsw)
- static void prep_cdevsw (struct cdevsw *devsw)
- cdev * make_dev (struct cdevsw *devsw, int minornr, uid_t uid, gid_t gid, int mode, const char *fmt,...)

- cdev * make_dev_cred (struct cdevsw *devsw, int minornr, struct ucred *cr, uid_t uid, gid_t gid, int mode, const char *fmt,...)
- static void dev_dependsl (struct cdev *pdev, struct cdev *cdev)
- void dev_depends (struct cdev *pdev, struct cdev *cdev)
- cdev * make_dev_alias (struct cdev *pdev, const char *fmt,...)
- void destroy_dev (struct cdev *dev)
- const char * devtoname (struct cdev *dev)
- int dev_stdclone (char *name, char **namep, const char *stem, int *unit)
- void clone_setup (struct clonedevs **cdp)
- int clone_create (struct clonedevs **cdp, struct cdevsw *csw, int *up, struct cdev **dp, int extra)
- void clone_cleanup (struct clonedevs **cdp)

Variables

- mtx devmtx
- static struct cdevsw dead_cdevsw

9.20.1 Define Documentation

9.20.1.1 #define dead_close (d_close_t *)enxio

Definition at line 185 of file kern_conf.c.

Referenced by prep_cdevsw().

9.20.1.2 #define dead_dump (dumper_t *)enxio

Definition at line 199 of file kern_conf.c.

Referenced by prep_cdevsw().

9.20.1.3 #define dead_ioctl (d_ioctl_t *)enxio

Definition at line 188 of file kern_conf.c.

Referenced by prep_cdevsw().

9.20.1.4 #define dead_kqfilter (d_kqfilter_t *)enxio

Definition at line 200 of file kern_conf.c.

Referenced by prep_cdevsw().

9.20.1.5 #define dead_mmap (d_mmap_t *)enodev

Definition at line 190 of file kern_conf.c.

Referenced by prep_cdevsw().

9.20.1.6 #define dead_open (d_open_t *)enxio

Definition at line 184 of file kern_conf.c.

Referenced by prep_cdevsw().

9.20.1.7 #define dead_poll (d_poll_t *)enodev

Definition at line 189 of file kern_conf.c.

Referenced by prep_cdevsw().

9.20.1.8 #define dead_read (d_read_t *)enxio

Definition at line 186 of file kern_conf.c.

Referenced by prep_cdevsw().

9.20.1.9 #define dead_write (d_write_t *)enxio

Definition at line 187 of file kern_conf.c.

Referenced by prep_cdevsw().

9.20.1.10 #define FIXUP(member, noop, giant)

Value:

```
do {                                                    \
                if (devsw->member == NULL) {                    \
                        devsw->member = noop;                   \
                } else if (devsw->d_flags & D_NEEDGIANT)        \
                        devsw->member = giant;                  \
                }                                               \
        while (0)
```

Referenced by prep_cdevsw().

9.20.1.11 #define no_dump (dumper_t *)enodev

Definition at line 252 of file kern_conf.c.

Referenced by prep_cdevsw().

9.20.1.12 #define no_ioctl (d_ioctl_t *)enodev

Definition at line 224 of file kern_conf.c.

Referenced by prep_cdevsw().

9.20.1.13 #define no_kqfilter (d_kqfilter_t *)enodev

Definition at line 226 of file kern_conf.c.

Referenced by prep_cdevsw().

9.20.1.14 #define no_mmap (d_mmap_t *)enodev

Definition at line 225 of file kern_conf.c.

Referenced by prep_cdevsw().

9.20.1.15 #define no_read (d_read_t *)enodev

Definition at line 222 of file kern_conf.c.

Referenced by prep_cdevsw().

9.20.1.16 #define no_write (d_write_t *)enodev

Definition at line 223 of file kern_conf.c.

Referenced by prep_cdevsw().

9.20.1.17 #define null_close (d_close_t *)nullop

Definition at line 221 of file kern_conf.c.

Referenced by prep_cdevsw().

9.20.1.18 #define null_open (d_open_t *)nullop

Definition at line 220 of file kern_conf.c.

Referenced by prep_cdevsw().

9.20.2 Function Documentation

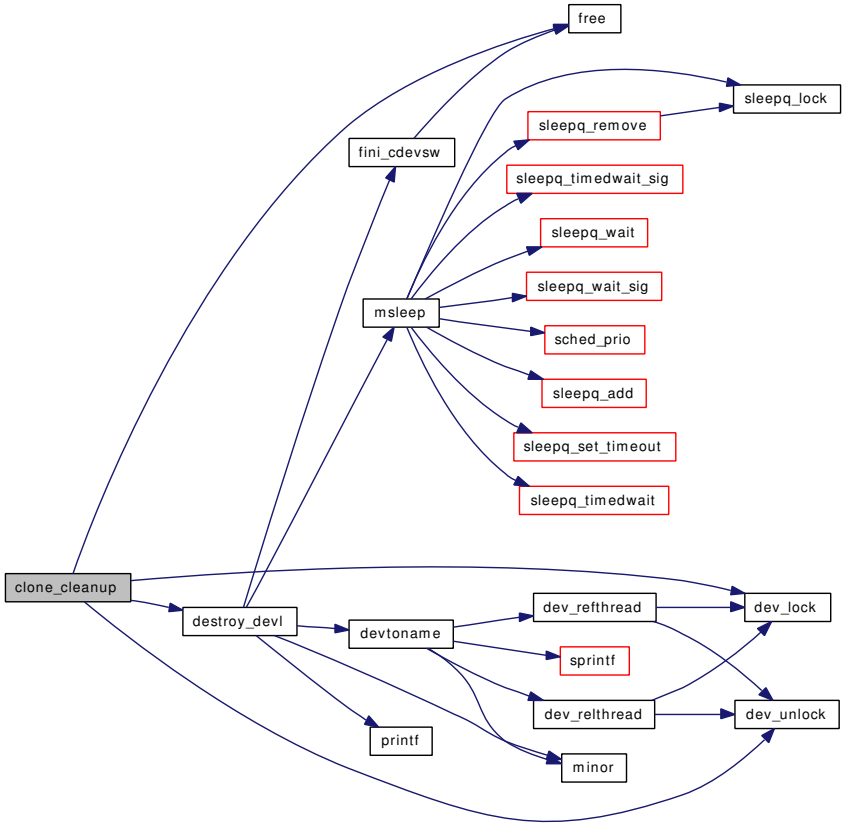
9.20.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_conf. *c*, v 1.201 2007/02/02 22:27:45 bms Exp $")

9.20.2.2 void clone_cleanup (struct clonedevs ** *cdp*)

Definition at line 888 of file kern_conf.c.

References destroy_devl(), dev_lock(), dev_unlock(), and free().

Here is the call graph for this function:

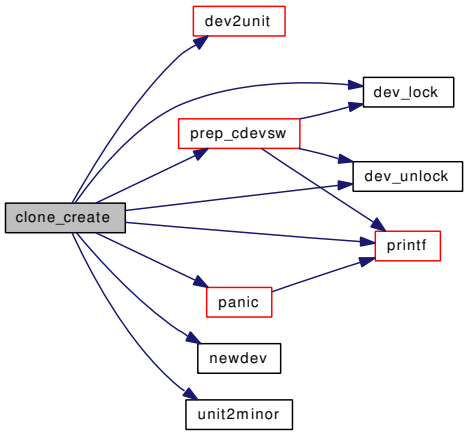


9.20.2.3 int clone_create (struct clonedevs ** *cdp*, struct cdevsw * *csw*, int * *up*, struct cdev ** *dp*, int *extra*)

Definition at line 806 of file kern_conf.c.

References dev2unit(), dev_lock(), dev_unlock(), newdev(), panic(), prep_cdevsw(), printf(), and unit2minor().

Here is the call graph for this function:

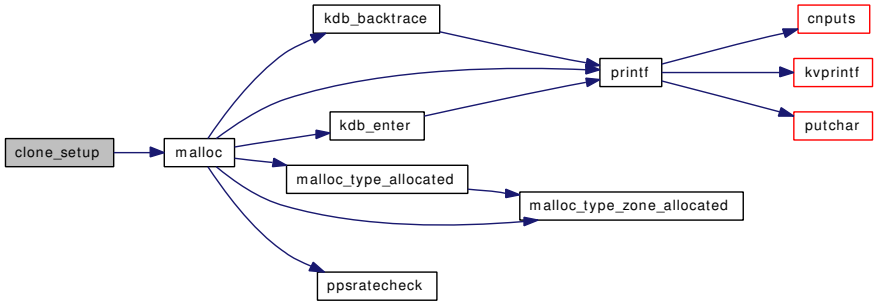


9.20.2.4 void clone_setup (struct clonedevs ** *cdp*)

Definition at line 798 of file kern_conf.c.

References malloc().

Here is the call graph for this function:



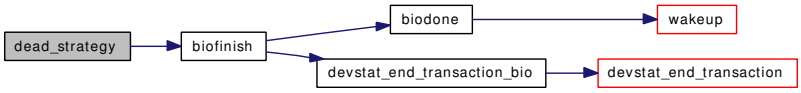
9.20.2.5 static void dead_strategy (struct bio * *bp*) [static]

Definition at line 193 of file kern_conf.c.

References biofinish().

Referenced by prep_cdevsw().

Here is the call graph for this function:



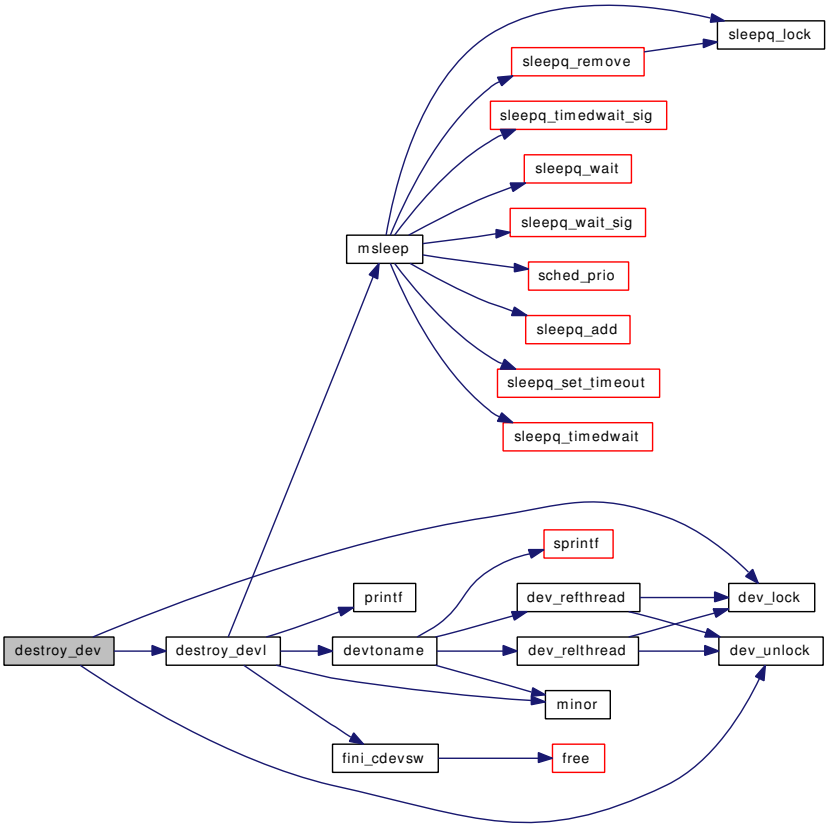
9.20.2.6 void destroy_dev (struct cdev * *dev*)

Definition at line 717 of file kern_conf.c.

References destroy_devl(), dev_lock(), and dev_unlock().

Referenced by pty_destroy_slave(), pty_maybecleanup(), and ttyfree().

Here is the call graph for this function:



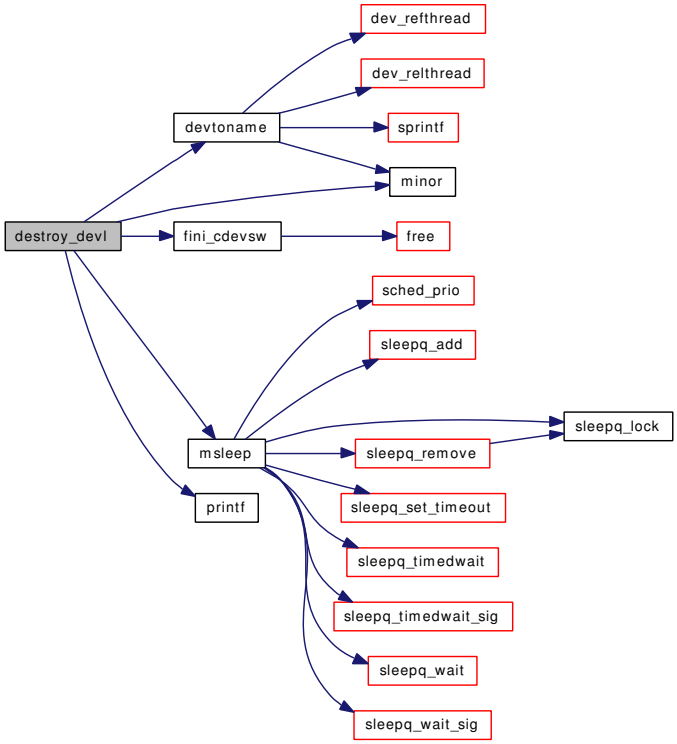
9.20.2.7 static void destroy_devl (struct cdev * *dev*) [static]

Definition at line 650 of file kern_conf.c.

References dead_cdevsw, devmtx, devtoname(), fini_cdevsw(), hz, minor(), msleep(), and printf().

Referenced by clone_cleanup(), and destroy_dev().

Here is the call graph for this function:



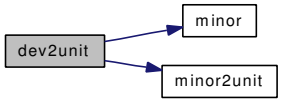
9.20.2.8 int dev2unit (struct cdev * *x*)

Definition at line 386 of file kern_conf.c.

References minor(), and minor2unit().

Referenced by clone_create(), and fdopen().

Here is the call graph for this function:



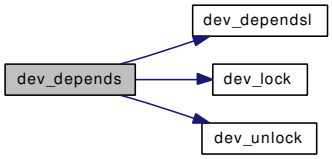
9.20.2.9 void dev_depends (struct cdev * *pdev*, struct cdev * *cdev*)

Definition at line 616 of file kern_conf.c.

References dev_dependsl(), dev_lock(), and dev_unlock().

Referenced by make_dev_alias(), and ttycreate().

Here is the call graph for this function:



9.20.2.10 static void dev_dependsl (struct cdev * *pdev*, struct cdev * *cdev*) [static]

Definition at line 606 of file kern_conf.c.

Referenced by dev_depends().

9.20.2.11 void dev_lock (void)

Definition at line 58 of file kern_conf.c.

References devmtx.

Referenced by clone_cleanup(), clone_create(), count_dev(), destroy_dev(), dev_depends(), dev_refthread(), dev_rel(), dev_relthread(), devvn_refthread(), make_dev_alias(), make_dev_credv(), prep_cdevsw(), v_decr_usecount(), v_decr_useonly(), v_incr_usecount(), v_upgrade_usecount(), vcount(), and vn_isdisk().

9.20.2.12 void dev_ref (struct cdev * *dev*)

Definition at line 72 of file kern_conf.c.

References devmtx.

Referenced by ctty_clone(), and pty_clone().

9.20.2.13 void dev_refl (struct cdev * *dev*)

Definition at line 82 of file kern_conf.c.

References devmtx.

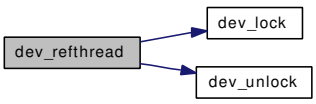
9.20.2.14 struct cdevsw* dev_refthread (struct cdev * *dev*)

Definition at line 115 of file kern_conf.c.

References dev_lock(), dev_unlock(), and devmtx.

Referenced by cn_devopen(), cnioctl(), cnkqfilter(), cnpoll(), cnread(), cnwrite(), dev_strategy(), devtoname(), and TAILQ_HEAD().

Here is the call graph for this function:

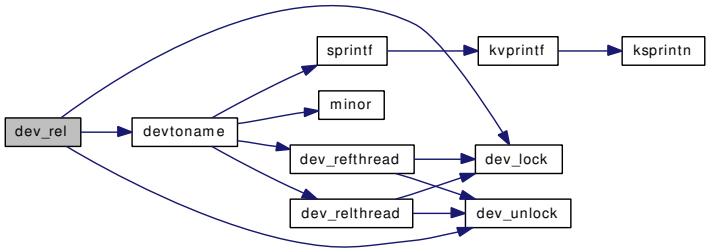


9.20.2.15 void dev_rel (struct cdev * *dev*)

Definition at line 90 of file kern_conf.c.

References dev_lock(), dev_unlock(), devmtx, and devtoname().

Here is the call graph for this function:



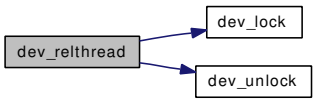
9.20.2.16 void dev_relthread (struct cdev * *dev*)

Definition at line 147 of file kern_conf.c.

References dev_lock(), dev_unlock(), and devmtx.

Referenced by cn_devopen(), cnioctl(), cnkqfilter(), cnpoll(), cnread(), cnwrite(), dev_strategy(), devtoname(), and TAILQ_HEAD().

Here is the call graph for this function:



9.20.2.17 int dev_stdclone (char * *name*, char ** *namep*, const char * *stem*, int * *unit*)

Definition at line 750 of file kern_conf.c.

9.20.2.18 void dev_unlock (void)

Definition at line 65 of file kern_conf.c.

References devmtx.

Referenced by clone_cleanup(), clone_create(), count_dev(), destroy_dev(), dev_depends(), dev_refthread(), dev_rel(), dev_relthread(), devvn_refthread(), make_dev_alias(), make_dev_credv(), prep_cdevsw(), v_decr_usecount(), v_decr_useonly(), v_incr_usecount(), v_upgrade_usecount(), vcount(), and vn_isdisk().

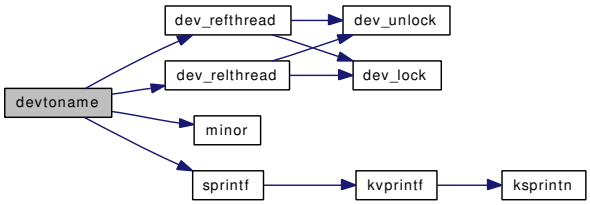
9.20.2.19 const char* devtoname (struct cdev * *dev*)

Definition at line 726 of file kern_conf.c.

References dev_refthread(), dev_relthread(), minor(), and sprintf().

Referenced by destroy_devl(), dev_rel(), dev_strategy(), disk_err(), make_dev_credv(), physio(), TAILQ_-HEAD(), ttyref(), ttyrel(), ttysioctl(), and ttysopen().

Here is the call graph for this function:

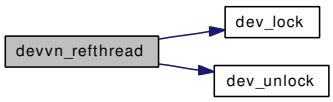


9.20.2.20 struct cdevsw* devvn_refthread (struct vnode * *vp*, struct cdev ** *devp*)

Definition at line 129 of file kern_conf.c.

References dev_lock(), dev_unlock(), and devmtx.

Here is the call graph for this function:



9.20.2.21 static int enodev (void) [static]

Definition at line 177 of file kern_conf.c.

9.20.2.22 static int enxio (void) [static]

Definition at line 171 of file kern_conf.c.

9.20.2.23 int eopnotsupp (void)

Definition at line 164 of file kern_conf.c.

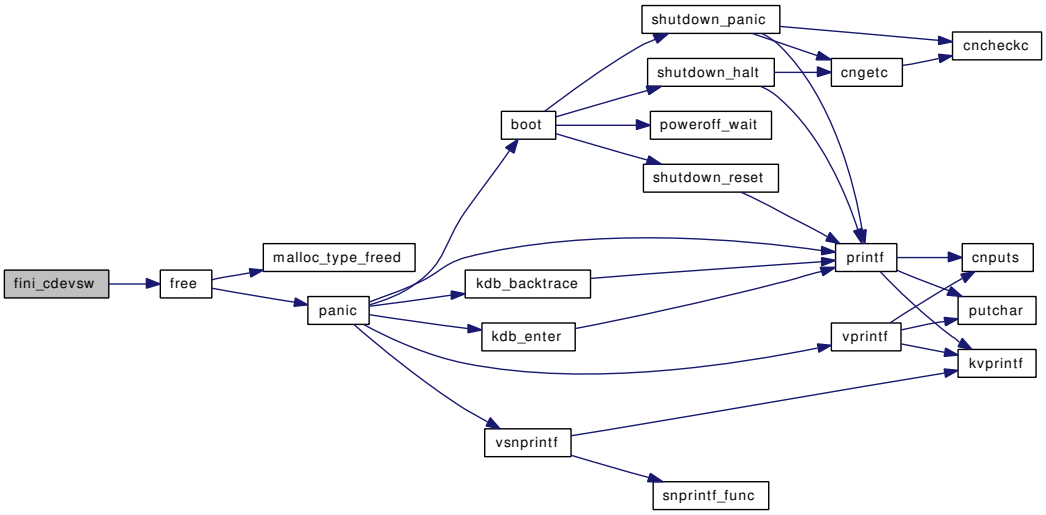
9.20.2.24 static void fini_cdevsw (struct cdevsw * *devsw*) [static]

Definition at line 443 of file kern_conf.c.

References free().

Referenced by destroy_devl().

Here is the call graph for this function:



9.20.2.25 static int giant_close (struct cdev * *dev*, int *fflag*, int *devtype*, struct thread * *td*) [static]

Definition at line 279 of file kern_conf.c.

References Giant.

Referenced by prep_cdevsw().

9.20.2.26 static int giant_fdopen (struct cdev * *dev*, int *oflags*, struct thread * *td*, int *fdidx*) [static]

Definition at line 267 of file kern_conf.c.

References Giant.

Referenced by prep_cdevsw().

9.20.2.27 static int giant_ioctl (struct cdev * *dev*, u_long *cmd*, caddr_t *data*, int *fflag*, struct thread * *td*) [static]

Definition at line 301 of file kern_conf.c.

References Giant.

Referenced by prep_cdevsw().

9.20.2.28 static int giant_kqfilter (struct cdev * *dev*, struct knote * *kn*) [static]

Definition at line 349 of file kern_conf.c.

References Giant.

Referenced by prep_cdevsw().

9.20.2.29 static int giant_mmap (struct cdev * *dev*, vm_offset_t *offset*, vm_paddr_t * *paddr*, int *nprot*) [static]

Definition at line 361 of file kern_conf.c.

References Giant.

Referenced by prep_cdevsw().

9.20.2.30 static int giant_open (struct cdev * *dev*, int *oflags*, int *devtype*, struct thread * *td*) [static]

Definition at line 255 of file kern_conf.c.

References Giant.

Referenced by prep_cdevsw().

9.20.2.31 static int giant_poll (struct cdev * *dev*, int *events*, struct thread * *td*) [static]

Definition at line 337 of file kern_conf.c.

References Giant.

Referenced by prep_cdevsw().

9.20.2.32 static int giant_read (struct cdev * *dev*, struct uio * *uio*, int *ioflag*) [static]

Definition at line 313 of file kern_conf.c.

References Giant.

Referenced by prep_cdevsw().

9.20.2.33 static void giant_strategy (struct bio * *bp*) [static]

Definition at line 291 of file kern_conf.c.

References Giant.

Referenced by prep_cdevsw().

9.20.2.34 static int giant_write (struct cdev * *dev*, struct uio * *uio*, int *ioflag*) [static]

Definition at line 325 of file kern_conf.c.

References Giant.

Referenced by prep_cdevsw().

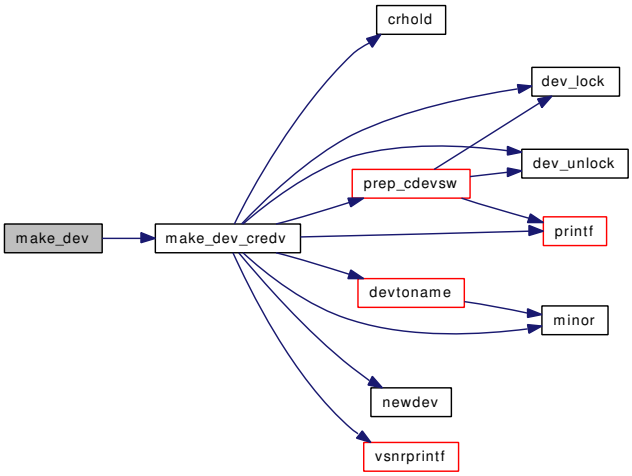
9.20.2.35 struct cdev* make_dev (struct cdevsw * *devsw*, int *minornr*, uid_t *uid*, gid_t *gid*, int *mode*, const char * *fmt*, ...)

Definition at line 579 of file kern_conf.c.

References make_dev_credv().

Referenced by cn_drvinit(), ctty_drvinit(), devinit(), devstat_alloc(), fildesc_drvinit(), log_drvinit(), and ttycreate().

Here is the call graph for this function:



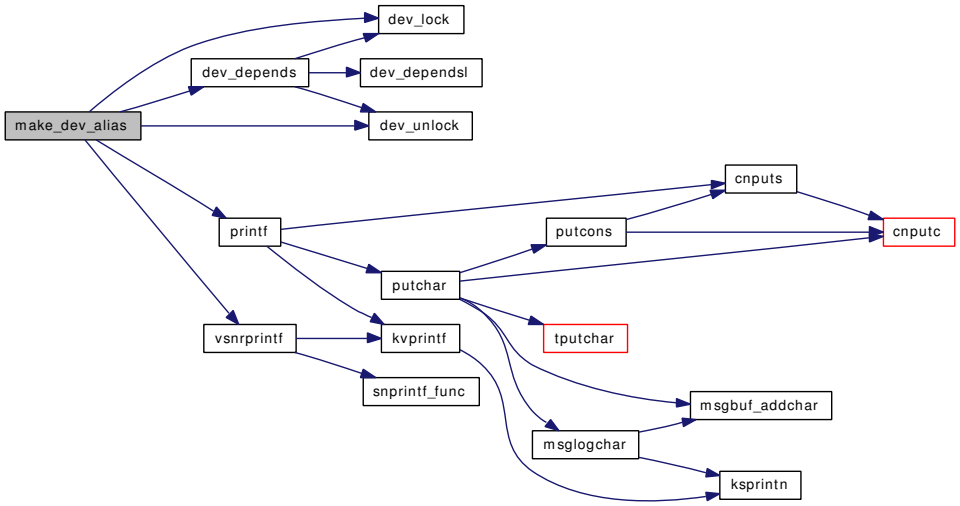
9.20.2.36 struct cdev* make_dev_alias (struct cdev * *pdev*, const char * *fmt*, ...)

Definition at line 625 of file kern_conf.c.

References dev_depends(), dev_lock(), dev_unlock(), printf(), and vsnrprintf().

Referenced by fildesc_drvinit().

Here is the call graph for this function:



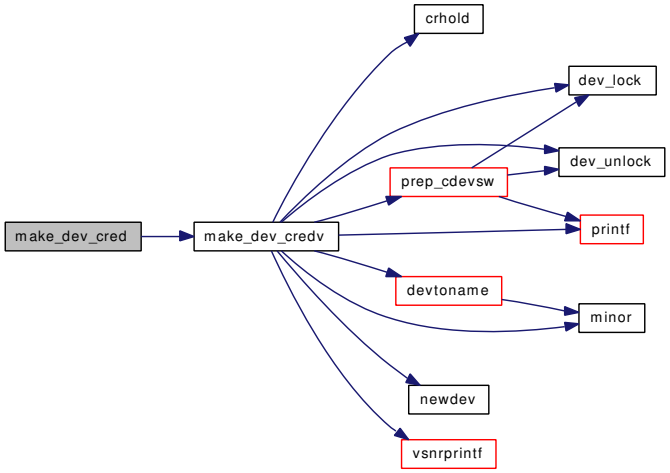
9.20.2.37 struct cdev* make_dev_cred (struct cdevsw * *devsw*, int *minornr*, struct ucred * *cr*, uid_t *uid*, gid_t *gid*, int *mode*, const char * *fmt*, ...)

Definition at line 592 of file kern_conf.c.

References make_dev_credv().

Referenced by ptcopen(), pty_clone(), and pty_create_slave().

Here is the call graph for this function:



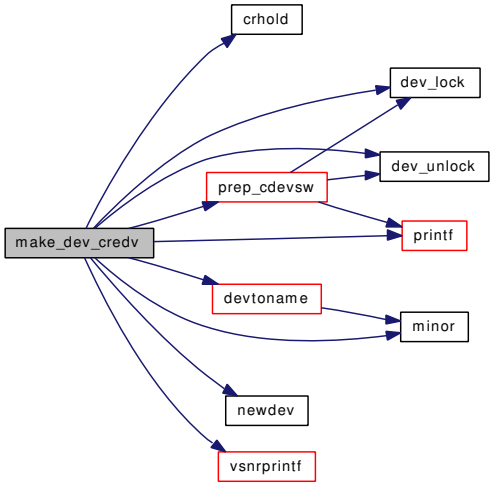
9.20.2.38 static struct cdev * make_dev_credv (struct cdevsw * *devsw*, int *minornr*, struct ucred * *cr*, uid_t *uid*, gid_t *gid*, int *mode*, const char * *fmt*, va_list *ap*) [static]

Definition at line 530 of file kern_conf.c.

References crhold(), dev_lock(), dev_unlock(), devtoname(), minor(), newdev(), prep_cdevsw(), printf(), and vsnrprintf().

Referenced by make_dev(), and make_dev_cred().

Here is the call graph for this function:



9.20.2.39 static MALLOC_DEFINE (M_DEVT, "cdev", "cdev storage") [static]

9.20.2.40 int minor (struct cdev * *x*)

Definition at line 378 of file kern_conf.c.

Referenced by destroy_devl(), dev2unit(), devtoname(), make_dev_credv(), ptcopen(), ptyinit(), and ttycreate().

9.20.2.41 u_int minor2unit (u_int *_minor*)

Definition at line 395 of file kern_conf.c.

Referenced by dev2unit().

9.20.2.42 static struct cdev* newdev (struct cdevsw * *csw*, int *y*, struct cdev * *si*) [static]

Definition at line 411 of file kern_conf.c.

References devmtx.

Referenced by clone_create(), and make_dev_credv().

9.20.2.43 static int no_poll (struct cdev *dev *__unused*, int *events*, struct thread *td *__unused*) [static]

Definition at line 236 of file kern_conf.c.

Referenced by prep_cdevsw().

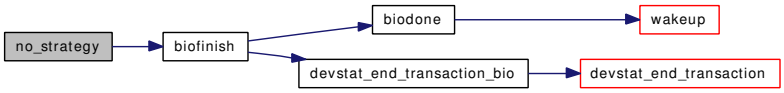
9.20.2.44 static void no_strategy (struct bio * *bp*) [static]

Definition at line 229 of file kern_conf.c.

References biofinish().

Referenced by prep_cdevsw().

Here is the call graph for this function:



9.20.2.45 int nullop (void)

Definition at line 157 of file kern_conf.c.

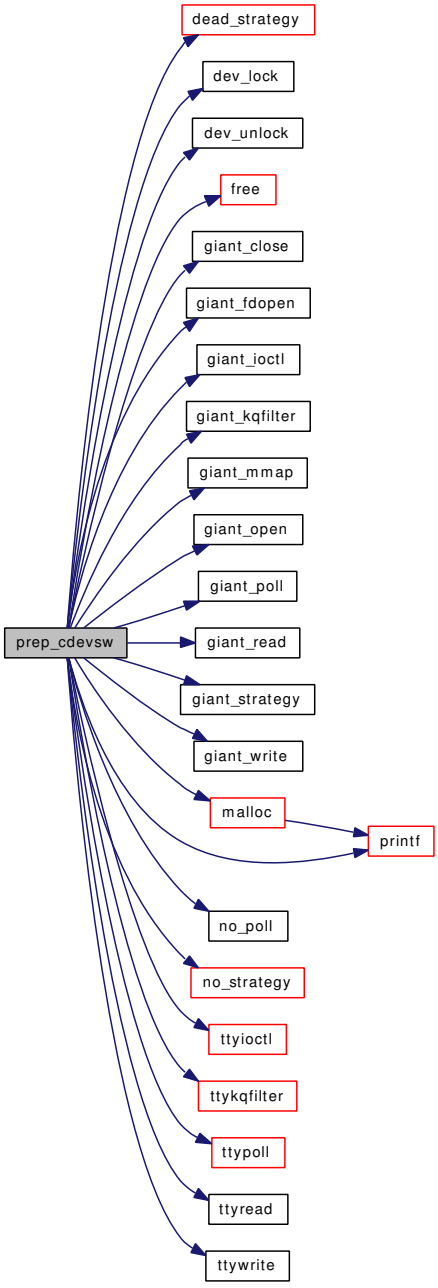
9.20.2.46 static void prep_cdevsw (struct cdevsw * *devsw*) [static]

Definition at line 457 of file kern_conf.c.

References dead_close, dead_dump, dead_ioctl, dead_kqfilter, dead_mmap, dead_open, dead_poll, dead_-read, dead_strategy(), dead_write, dev_lock(), dev_unlock(), FIXUP, free(), giant_close(), giant_fdopen(), giant_ioctl(), giant_kqfilter(), giant_mmap(), giant_open(), giant_poll(), giant_read(), giant_strategy(), giant_write(), malloc(), no_dump, no_ioctl, no_kqfilter, no_mmap, no_poll(), no_read, no_strategy(), no_-write, null_close, null_open, printf(), ttyioctl(), ttykqfilter(), ttypoll(), ttyread(), and ttywrite().

Referenced by clone_create(), and make_dev_credv().

Here is the call graph for this function:



9.20.2.47 int umajor (dev_t *dev*)

Definition at line 437 of file kern_conf.c.

9.20.2.48 int uminor (dev_t *dev*)

Definition at line 431 of file kern_conf.c.

9.20.2.49 int unit2minor (int *unit*)

Definition at line 403 of file kern_conf.c.

Referenced by clone_create(), and ttycreate().

9.20.3 Variable Documentation

9.20.3.1 struct cdevsw dead_cdevsw [static]

Initial value:

```
{
        .d_version =    D_VERSION,
        .d_flags =      D_NEEDGIANT,
        .d_open =       dead_open,
        .d_close =      dead_close,
        .d_read =       dead_read,
        .d_write =      dead_write,
        .d_ioctl =      dead_ioctl,
        .d_poll =       dead_poll,
        .d_mmap =       dead_mmap,
        .d_strategy =   dead_strategy,
        .d_name =       "dead",
        .d_dump =       dead_dump,
        .d_kqfilter =   dead_kqfilter
}
```

Definition at line 202 of file kern_conf.c.

Referenced by destroy_devl().

9.20.3.2 struct mtx devmtx

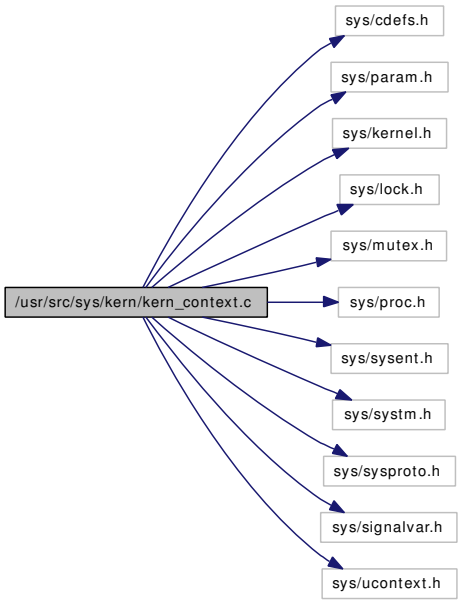
Definition at line 51 of file kern_conf.c.

Referenced by destroy_devl(), dev_lock(), dev_ref(), dev_refl(), dev_refthread(), dev_rel(), dev_relthread(), dev_unlock(), devvn_refthread(), mutex_init(), and newdev().

9.21 /usr/src/sys/kern/kern_context.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/sysent.h>
#include <sys/systm.h>
#include <sys/sysproto.h>
#include <sys/signalvar.h>
#include <sys/ucontext.h>
```

Include dependency graph for kern_context.c:



Data Structures

- struct getcontext_args
- struct swapcontext_args

Defines

- #define UC_COPY_SIZE offsetof(ucontext_t, uc_link)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_context.c,v 1.7 2003/11/09 20:31:03 marcel Exp $")
- if (uap → ucp==NULL) ret
- int setcontext (struct thread *td, struct setcontext_args *uap)
- int swapcontext (struct thread *td, struct swapcontext_args *uap)

Variables

- getcontext_args * ucp
- swapcontext_args * td
- swapcontext_args uc
- int ret

9.21.1 Define Documentation

9.21.1.1 #define UC_COPY_SIZE offsetof(ucontext_t, uc_link)

Definition at line 46 of file kern_context.c.

Referenced by setcontext(), and swapcontext().

9.21.2 Function Documentation

9.21.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_context. *c*, v 1.7 2003/11/09 20:31:03 marcel Exp $")

9.21.2.2 if (uap → *ucp* = =NULL)

Referenced by _do_lock_normal(), _do_lock_pi(), _do_lock_pp(), _do_lock_umtx(), _rw_rlock(), _rw_runlock(), acl_posix1e_check(), aio_suspend(), blst_meta_free(), buf_splay(), cf_set_method(), chroot_refuse_vdir_fds(), cluster_rbuild(), cluster_read(), coredump(), cpufreq_expand_set(), cr_seeothergids(), devclass_delete_driver(), devclass_quiesce_driver(), do_set_ceiling(), do_wait(), exec_aout_imgact(), exec_copyout_strings(), exec_gzip_imgact(), fdalloc(), fdcheckstd(), filt_aio(), filt_aioattach(), filt_aiodetach(), filt_lioattach(), filt_liodetach(), filt_vfsread(), find_instance(), getc(), huft_build(), inflate_codes(), ioctl(), issignal(), kern_msgrcv(), kern_sigtimedwait(), kern_timer_create(), lf_setlock(), link_elf_fix_link_set(), link_elf_init(), link_elf_lookup_set(), link_elf_reloc_local(), link_elf_unload_file(), linker_addmodules(), linker_file_lookup_symbol_internal(), linker_file_register_modules(), linker_load_dependencies(), linker_preload(), lookup(), m_adj(), m_copymdata(), m_length(), m_pulldown(), m_sanity(), m_split(), m_unshare(), mb_reclaim(), mbp_count(), mbp_destroy(), mi_startup(), mqfs_readdir(), namei(), pfctlinput(), pfctlinput2(), pffasttimo(), pffindproto(), pffindtype(), pfslowtimo(), physio(), pipe_build_write_buffer(), propagate_priority(), ptcread(), relocate_file(), removechild(), sched_setup(), semundo_adjust(), shmget_allocate_segment(), shminit(), sleepq_add(), socow_setup(), STAILQ_HEAD(), sysctl_devices(), sysctl_sysctl_next_ls(), sysctl_wire_old_buffer(), taskqueue_start_threads(), tc_windup(), timespec2fattime(), ttcompat(), ttyioctl(), uipc_send(), umtx_propagate_priority(), unp_addsockcred(), unp_scan(), uuid_node(), vaccess_acl_posix1e(), vfs_bio_clrbuf(), vfs_bio_set_validclean(), vfs_busy_pages(), vfs_clean_pages(), vfs_mount_destroy(), vfs_setdirty_locked_object(), vfs_unbusy_pages(), vfs_vmio_release(), vm_hold_free_pages(), witness_checkorder(), and witness_unlock().

9.21.2.3 int setcontext (struct thread * *td*, struct setcontext_args * *uap*)

Definition at line 86 of file kern_context.c.

References UC_COPY_SIZE.

9.21.2.4 int swapcontext (struct thread * *td*, struct swapcontext_args * *uap*)

Definition at line 109 of file kern_context.c.

References swapcontext_args::oucp, UC_COPY_SIZE, and swapcontext_args::ucp.

9.21.3 Variable Documentation

9.21.3.1 int ret

Definition at line 68 of file kern_context.c.

Referenced by _sema_trywait(), acct_process(), aio_kick(), aio_sendsig(), elf_obj_lookup(), filt_piperead(), fsetown(), getenv(), itimer_fire(), kern_umtx_wake(), ktrace(), mqfs_root(), resource_find_dev(), resource_find_match(), resource_string_copy(), sem_create(), sema_value(), sigqueue_add(), sleepq_catch_signals(), speedup_syncer(), sscanf(), sysctl_wire_old_buffer(), tdsignal(), umtxq_signal(), and vfs_scanopt().

9.21.3.2 struct swapcontext_args* td

Referenced by alq_doio(), alq_open(), ast(), bdwrite(), cv_timedwait(), cv_timedwait_sig(), cv_wait_sig(), cv_wait_unlock(), devfs_first(), do_aout_hdr(), doselwakeup(), exec_aout_imgact(), exec_check_permissions(), firmware_get(), flushbufqueues(), fork_exit(), hardclock_cpu(), idle_setup(), inflate_codes(), inflate_dynamic(), intr_event_schedule_thread(), ithread_create(), ithread_destroy(), ithread_loop(), ithread_update(), kern_symlink(), kick_init(), kthread_create(), kthread_exit(), ktr_tracepoint(), lf_setlock(), link_elf_load_file(), linker_hints_lookup(), linker_lookup_file(), lookup(), mi_switch(), mqfs_lookupx(), msleep(), msleep_spin(), namei(), panic(), poll_idle(), proc0_init(), proc_compare(), proc_dtor(), proc_init(), profclock(), relookup(), sched_nice(), sched_sync(), sched_tick(), schedcpu(), sigonstack(), sleepq_add(), sleepq_catch_signals(), sleepq_check_signals(), sleepq_check_timeout(), sleepq_set_timeout(), sleepq_signal(), sleepq_switch(), sleepq_timeout(), soreserve(), speedup_syncer(), start_init(), statclock(), sync_fsync(), syncer_shutdown(), sysctl_out_proc(), taskqueue_start_threads(), ttioctl(), ttread(), ttwrite(), ttycheckoutq(), ttyinfo(), turnstile_broadcast(), turnstile_claim(), turnstile_disown(), turnstile_signal(), turnstile_unpend(), turnstile_wait(), uio_yield(), uiomove(), umtx_key_get(), unp_externalize(), uprintf(), vfs_cache_lookup(), vfs_unmountall(), vfs_write_suspend(), vgonel(), vlrureclaim(), vn_open(), vn_open_cred(), vnlru_proc(), vput(), vrele(), vtryrecycle(), witness_checkorder(), witness_lock(), witness_unlock(), and witness_warn().

9.21.3.3 struct swapcontext_args uc

Referenced by sem_create(), sem_perm(), umtx_pi_insert(), umtx_pi_lookup(), umtx_pi_ref(), umtx_pi_unref(), umtxq_busy(), umtxq_count(), umtxq_count_pi(), umtxq_insert(), umtxq_lock(), umtxq_remove(), umtxq_signal(), umtxq_signal_thread(), umtxq_sleep(), umtxq_sleep_pi(), umtxq_unbusy(), and umtxq_unlock().

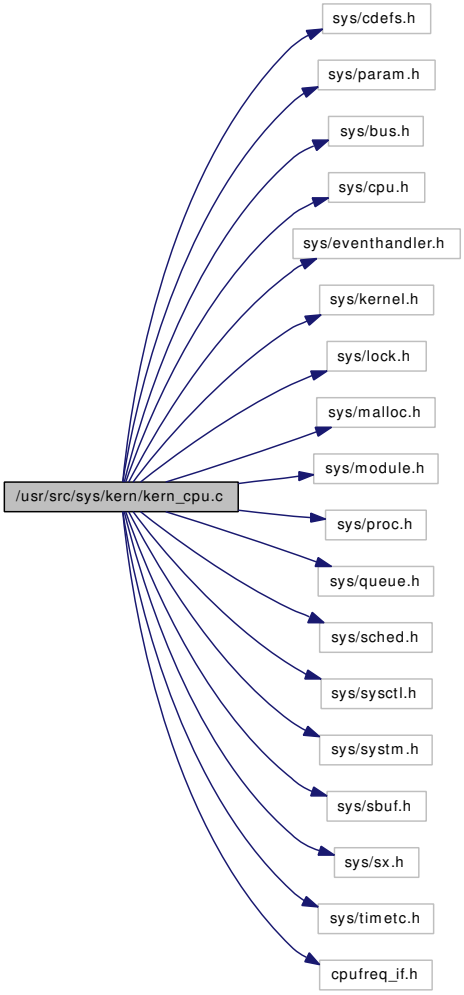
9.21.3.4 struct getcontext_args* ucp

Referenced by start_init().

9.22 /usr/src/sys/kern/kern_cpu.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/bus.h>
#include <sys/cpu.h>
#include <sys/eventhandler.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/module.h>
#include <sys/proc.h>
#include <sys/queue.h>
#include <sys/sched.h>
#include <sys/sysctl.h>
#include <sys/systm.h>
#include <sys/sbuf.h>
#include <sys/sx.h>
#include <sys/timetc.h>
#include "cpufreq_if.h"
```

Include dependency graph for kern_cpu.c:



Data Structures

- struct cf_saved_freq
- struct cpufreq_softc
- struct cf_setting_array

Defines

- #define CF_MAX_LEVELS 64
- #define CF_MTX_INIT(x) sx_init((x), "cpufreq lock")
- #define CF_MTX_LOCK(x) sx_xlock((x))
- #define CF_MTX_UNLOCK(x) sx_xunlock((x))
- #define CF_MTX_ASSERT(x) sx_assert((x), SX_XLOCKED)
- #define CF_DEBUG(msg...)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_cpu.c,v 1.23 2006/03/03 02:06:04 mnag Exp $")

- TAILQ_HEAD (cf_setting_lst, cf_setting_array)
- static int cpufreq_attach (device_t dev)
- static int cpufreq_detach (device_t dev)
- static void cpufreq_evaluate (void *arg)
- static int cf_set_method (device_t dev, const struct cf_level *level, int priority)
- static int cf_get_method (device_t dev, struct cf_level *level)
- static int cf_levels_method (device_t dev, struct cf_level *levels, int *count)
- static int cpufreq_insert_abs (struct cpufreq_softc *sc, struct cf_setting *sets, int count)
- static int cpufreq_expand_set (struct cpufreq_softc *sc, struct cf_setting_array *set_arr)
- static struct cf_level * cpufreq_dup_set (struct cpufreq_softc *sc, struct cf_level *dup, struct cf_-setting *set)
- static int cpufreq_curr_sysctl (SYSCTL_HANDLER_ARGS)
- static int cpufreq_levels_sysctl (SYSCTL_HANDLER_ARGS)
- static int cpufreq_settings_sysctl (SYSCTL_HANDLER_ARGS)
- DRIVER_MODULE (cpufreq, cpu, cpufreq_driver, cpufreq_dc, 0, 0)
- TUNABLE_INT ("debug.cpufreq.lowest",&cf_lowest_freq)
- TUNABLE_INT ("debug.cpufreq.verbose",&cf_verbose)
- SYSCTL_NODE (_debug, OID_AUTO, cpufreq, CTLFLAG_RD, NULL,"cpufreq debugging")
- SYSCTL_INT (_debug_cpufreq, OID_AUTO, lowest, CTLFLAG_RW,&cf_lowest_freq, 1,"Don't provide levels below this frequency.")
- SYSCTL_INT (_debug_cpufreq, OID_AUTO, verbose, CTLFLAG_RW,&cf_verbose, 1,"Print verbose debugging messages")
- int cpufreq_register (device_t dev)
- int cpufreq_unregister (device_t dev)

Variables

- static device_method_t cpufreq_methods [ ]
- static driver_t cpufreq_driver
- static devclass_t cpufreq_dc
- static eventhandler_tag cf_ev_tag
- static int cf_lowest_freq
- static int cf_verbose

9.22.1 Define Documentation

9.22.1.1 #define CF_DEBUG(msg...)

Value:

```
do {            \
        if (cf_verbose)                         \
                printf("cpufreq: " msg);        \
        } while (0)
```

Definition at line 91 of file kern_cpu.c.

Referenced by cf_get_method(), cf_levels_method(), cf_set_method(), cpufreq_attach(), cpufreq_detach(), cpufreq_dup_set(), cpufreq_expand_set(), and cpufreq_insert_abs().

9.22.1.2 #define CF_MAX_LEVELS 64

Definition at line 58 of file kern_cpu.c.

Referenced by cf_get_method(), cpufreq_curr_sysctl(), and cpufreq_levels_sysctl().

9.22.1.3 #define CF_MTX_ASSERT(x) sx_assert((x), SX_XLOCKED)

Definition at line 89 of file kern_cpu.c.

Referenced by cpufreq_dup_set(), cpufreq_expand_set(), and cpufreq_insert_abs().

9.22.1.4 #define CF_MTX_INIT(x) sx_init((x), "cpufreq lock")

Definition at line 86 of file kern_cpu.c.

Referenced by cpufreq_attach().

9.22.1.5 #define CF_MTX_LOCK(x) sx_xlock((x))

Definition at line 87 of file kern_cpu.c.

Referenced by cf_get_method(), cf_levels_method(), and cf_set_method().

9.22.1.6 #define CF_MTX_UNLOCK(x) sx_xunlock((x))

Definition at line 88 of file kern_cpu.c.

Referenced by cf_get_method(), and cf_levels_method().

9.22.2 Function Documentation

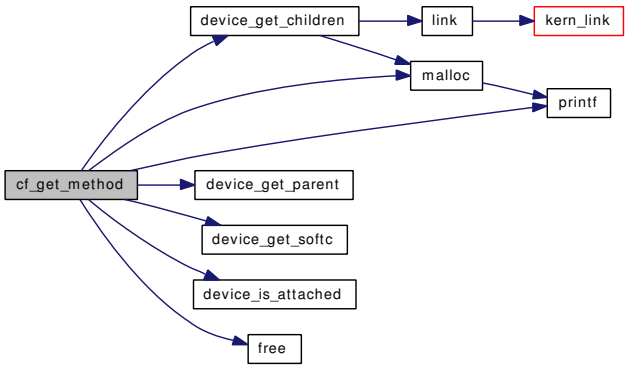
9.22.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_cpu. *c*, v 1.23 2006/03/03 02:06:04 mnag Exp $")

9.22.2.2 static int cf_get_method (device_t *dev*, struct cf_level * *level*) [static]

Definition at line 387 of file kern_cpu.c.

References CF_DEBUG, CF_MAX_LEVELS, CF_MTX_LOCK, CF_MTX_UNLOCK, cpufreq_softc::curr_level, device_get_children(), device_get_parent(), device_get_softc(), device_is_attached(), free(), cpufreq_softc::lock, malloc(), and printf().

Here is the call graph for this function:

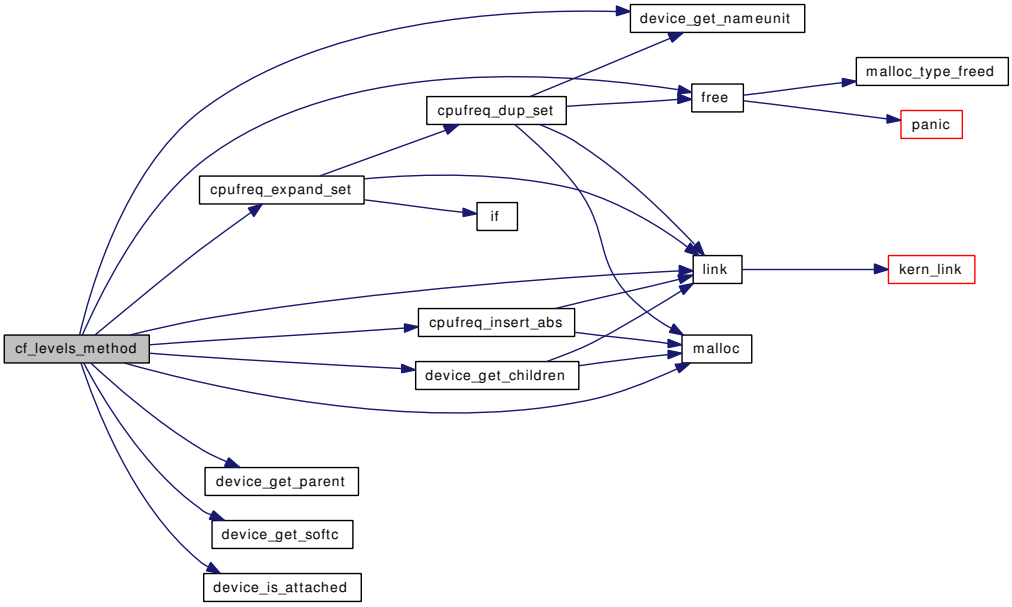


9.22.2.3 static int cf_levels_method (device_t *dev*, struct cf_level * *levels*, int * *count*) [static]

Definition at line 490 of file kern_cpu.c.

References CF_DEBUG, CF_MTX_LOCK, CF_MTX_UNLOCK, cpufreq_expand_set(), cpufreq_insert_abs(), device_get_children(), device_get_nameunit(), device_get_parent(), device_get_softc(), device_is_attached(), free(), link(), cpufreq_softc::lock, and malloc().

Here is the call graph for this function:



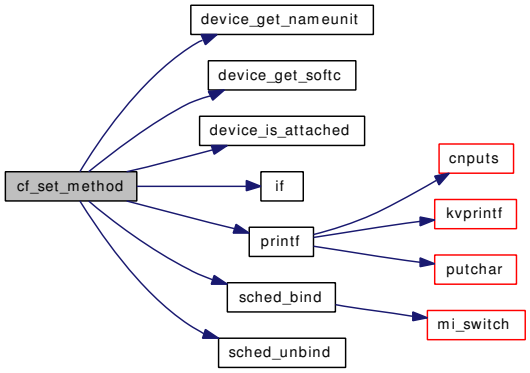
9.22.2.4 static int cf_set_method (device_t *dev*, const struct cf_level * *level*, int *priority*) [static]

Definition at line 218 of file kern_cpu.c.

References CF_DEBUG, CF_MTX_LOCK, cpufreq_softc::curr_level, cpufreq_softc::curr_priority,

device_get_nameunit(), device_get_softc(), device_is_attached(), if(), cf_saved_freq::level, cpufreq_-softc::lock, printf(), cf_saved_freq::priority, sched_bind(), sched_lock, sched_unbind(), and timecounter.

Here is the call graph for this function:

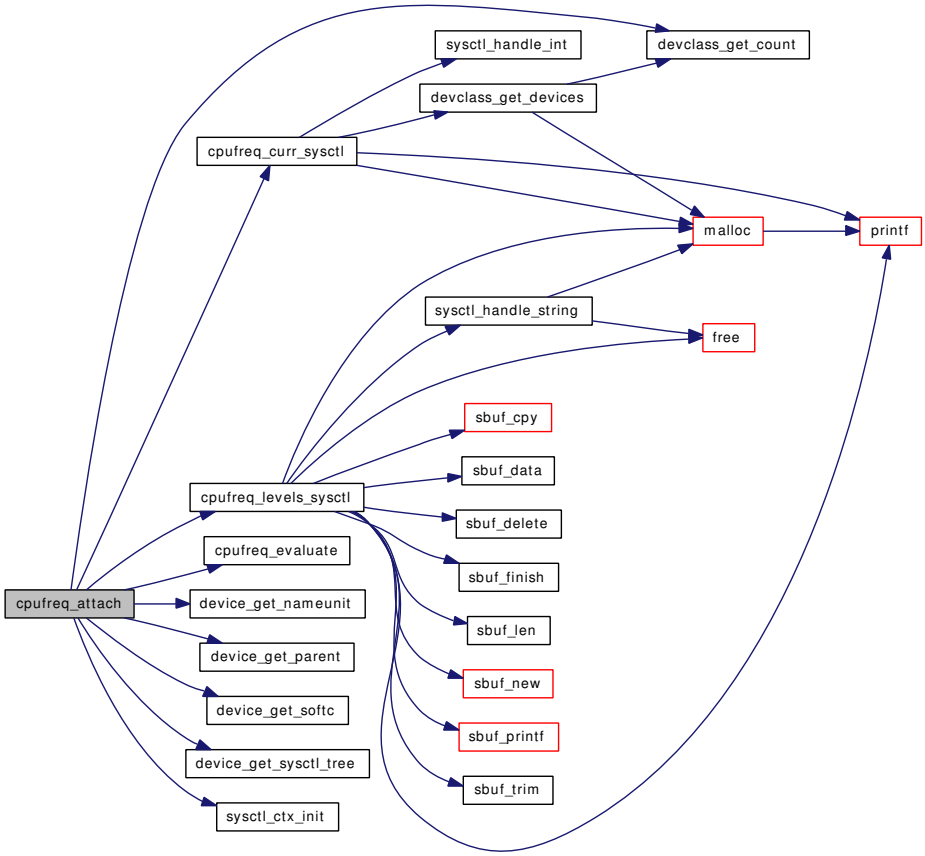


9.22.2.5 static int cpufreq_attach (device_t *dev*) [static]

Definition at line 143 of file kern_cpu.c.

References CF_DEBUG, CF_MTX_INIT, cpufreq_curr_sysctl(), cpufreq_evaluate(), cpufreq_levels_-sysctl(), cpufreq_softc::curr_level, devclass_get_count(), device_get_nameunit(), device_get_parent(), device_get_softc(), device_get_sysctl_tree(), cpufreq_softc::lock, and sysctl_ctx_init().

Here is the call graph for this function:



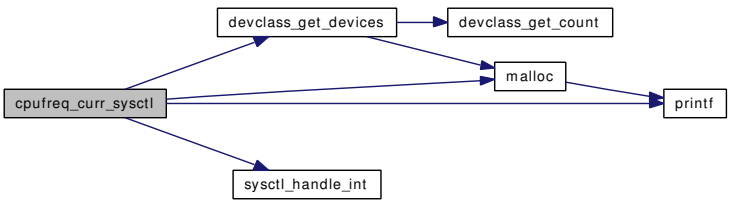
9.22.2.6 static int cpufreq_curr_sysctl (SYSCTL_HANDLER_ARGS) [static]

Definition at line 826 of file kern_cpu.c.

References CF_MAX_LEVELS, devclass_get_devices(), malloc(), printf(), and sysctl_handle_int().

Referenced by cpufreq_attach().

Here is the call graph for this function:

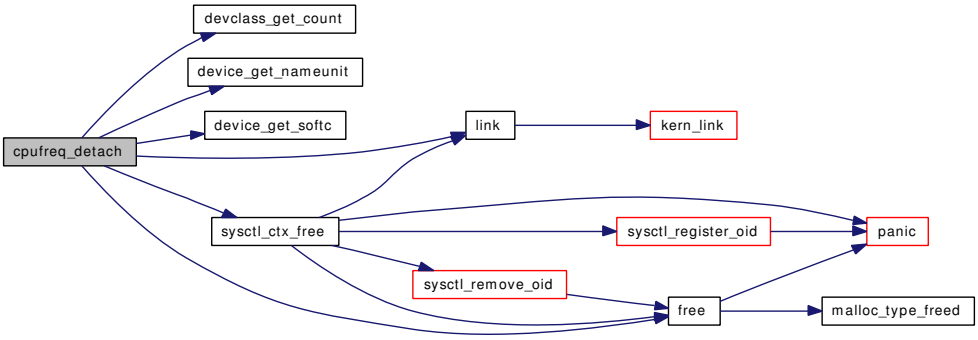


9.22.2.7 static int cpufreq_detach (device_t *dev*) [static]

Definition at line 186 of file kern_cpu.c.

References CF_DEBUG, devclass_get_count(), device_get_nameunit(), device_get_softc(), free(), link(), and sysctl_ctx_free().

Here is the call graph for this function:



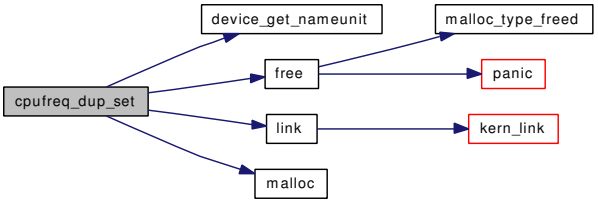
9.22.2.8 static struct cf_level * cpufreq_dup_set (struct cpufreq_softc * *sc*, struct cf_level * *dup*, struct cf_setting * *set*) [static]

Definition at line 731 of file kern_cpu.c.

References CF_DEBUG, CF_MTX_ASSERT, device_get_nameunit(), free(), link(), cpufreq_softc::lock, and malloc().

Referenced by cpufreq_expand_set().

Here is the call graph for this function:



9.22.2.9 static void cpufreq_evaluate (void * *arg*) [static]

Definition at line 212 of file kern_cpu.c.

Referenced by cpufreq_attach().

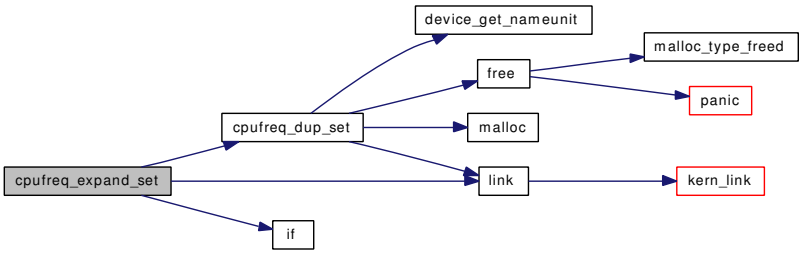
9.22.2.10 static int cpufreq_expand_set (struct cpufreq_softc * *sc*, struct cf_setting_array * *set_arr*) [static]

Definition at line 675 of file kern_cpu.c.

References CF_DEBUG, CF_MTX_ASSERT, cf_setting_array::count, cpufreq_dup_set(), if(), link(), cpufreq_softc::lock, and cf_setting_array::sets.

Referenced by cf_levels_method().

Here is the call graph for this function:



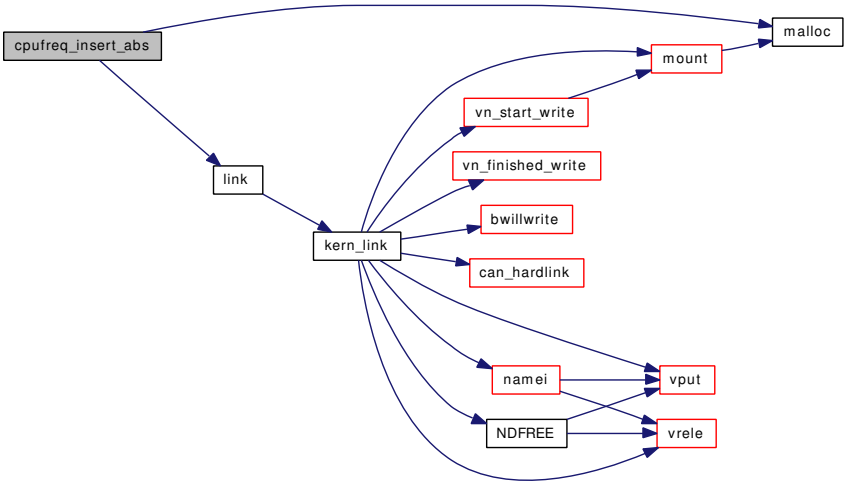
9.22.2.11 static int cpufreq_insert_abs (struct cpufreq_softc * *sc*, struct cf_setting * *sets*, int *count*) [static]

Definition at line 633 of file kern_cpu.c.

References CF_DEBUG, CF_MTX_ASSERT, link(), cpufreq_softc::lock, and malloc().

Referenced by cf_levels_method().

Here is the call graph for this function:



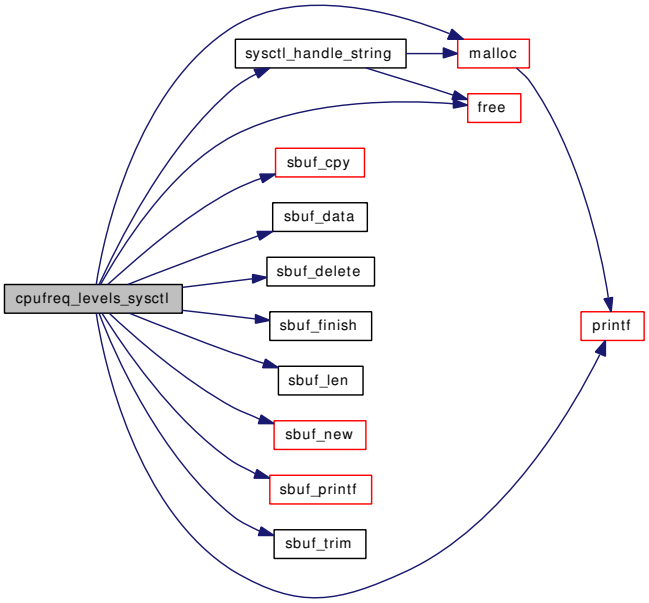
9.22.2.12 static int cpufreq_levels_sysctl (SYSCTL_HANDLER_ARGS) [static]

Definition at line 886 of file kern_cpu.c.

References CF_MAX_LEVELS, free(), malloc(), printf(), sbuf_cpy(), sbuf_data(), sbuf_delete(), sbuf_finish(), sbuf_len(), sbuf_new(), sbuf_printf(), sbuf_trim(), and sysctl_handle_string().

Referenced by cpufreq_attach().

Here is the call graph for this function:

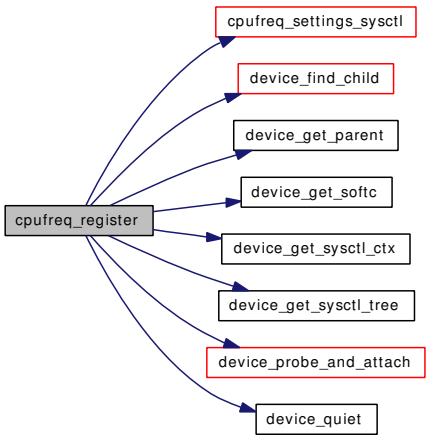


9.22.2.13 int cpufreq_register (device_t *dev*)

Definition at line 960 of file kern_cpu.c.

References cpufreq_settings_sysctl(), device_find_child(), device_get_parent(), device_get_softc(), device_get_sysctl_ctx(), device_get_sysctl_tree(), device_probe_and_attach(), and device_quiet().

Here is the call graph for this function:



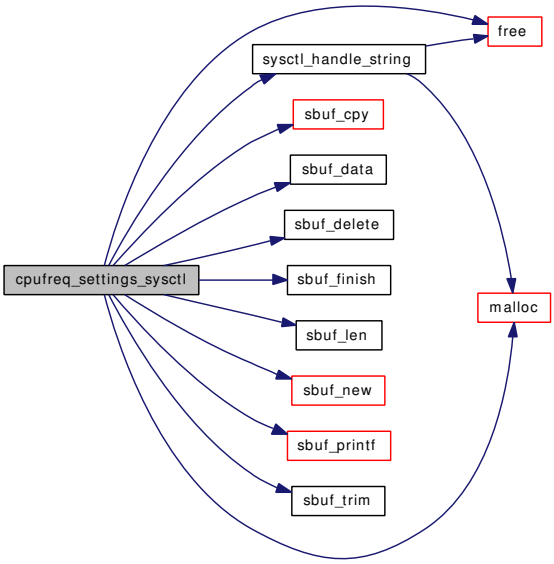
9.22.2.14 static int cpufreq_settings_sysctl (SYSCTL_HANDLER_ARGS) [static]

Definition at line 926 of file kern_cpu.c.

References free(), malloc(), sbuf_cpy(), sbuf_data(), sbuf_delete(), sbuf_finish(), sbuf_len(), sbuf_new(), sbuf_printf(), sbuf_trim(), and sysctl_handle_string().

Referenced by cpufreq_register().

Here is the call graph for this function:

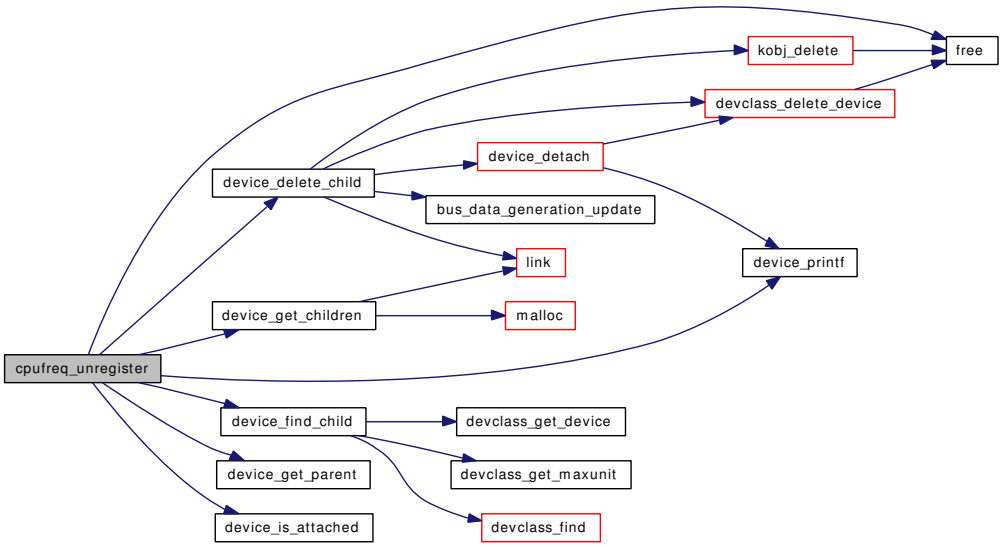


9.22.2.15 int cpufreq_unregister (device_t *dev*)

Definition at line 992 of file kern_cpu.c.

References device_delete_child(), device_find_child(), device_get_children(), device_get_parent(), device_is_attached(), device_printf(), and free().

Here is the call graph for this function:



9.22.2.16 DRIVER_MODULE (cpufreq, cpu, cpufreq_driver, cpufreq_dc, 0, 0)

9.22.2.17 SYSCTL_INT (_debug_cpufreq, OID_AUTO, verbose, CTLFLAG_RW, & *cf_verbose*, 1, "Print verbose debugging messages")

9.22.2.18 SYSCTL_INT (_debug_cpufreq, OID_AUTO, lowest, CTLFLAG_RW, & *cf_lowest_freq*, 1, "Don't provide levels below this frequency.")

9.22.2.19 SYSCTL_NODE (_debug, OID_AUTO, cpufreq, CTLFLAG_RD, NULL, "cpufreq debugging")

9.22.2.20 TAILQ_HEAD (cf_setting_lst, cf_setting_array)

Referenced by turnstile_unpend().

9.22.2.21 TUNABLE_INT ("debug.cpufreq.verbose", & *cf_verbose*)

9.22.2.22 TUNABLE_INT ("debug.cpufreq.lowest", & *cf_lowest_freq*)

9.22.3 Variable Documentation

9.22.3.1 eventhandler_tag cf_ev_tag [static]

Definition at line 130 of file kern_cpu.c.

9.22.3.2 int cf_lowest_freq [static]

Definition at line 132 of file kern_cpu.c.

9.22.3.3 int cf_verbose [static]

Definition at line 133 of file kern_cpu.c.

9.22.3.4 devclass_t cpufreq_dc [static]

Definition at line 127 of file kern_cpu.c.

9.22.3.5 driver_t cpufreq_driver [static]

Initial value:

```
{
        "cpufreq", cpufreq_methods, sizeof(struct cpufreq_softc)
}
```

Definition at line 124 of file kern_cpu.c.

9.22.3.6 device_method_t cpufreq_methods[ ] [static]

Initial value:

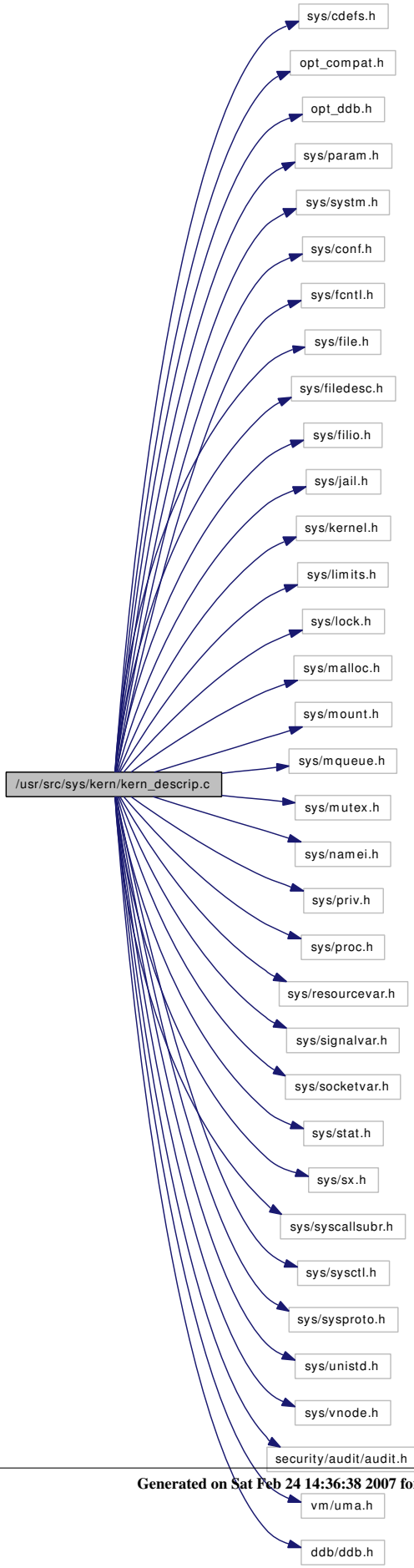
```
{
        DEVMETHOD(device_probe,         bus_generic_probe),
        DEVMETHOD(device_attach,        cpufreq_attach),
        DEVMETHOD(device_detach,        cpufreq_detach),

        DEVMETHOD(cpufreq_set,          cf_set_method),
        DEVMETHOD(cpufreq_get,          cf_get_method),
        DEVMETHOD(cpufreq_levels,       cf_levels_method),
        {0, 0}
}
```

Definition at line 114 of file kern_cpu.c.

9.23 /usr/src/sys/kern/kern_descrip.c File Reference

```
#include <sys/cdefs.h>
#include "opt_compat.h"
#include "opt_ddb.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/conf.h>
#include <sys/fcntl.h>
#include <sys/file.h>
#include <sys/filedesc.h>
#include <sys/filio.h>
#include <sys/jail.h>
#include <sys/kernel.h>
#include <sys/limits.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mount.h>
#include <sys/mqueue.h>
#include <sys/mutex.h>
#include <sys/namei.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/resourcevar.h>
#include <sys/signalvar.h>
#include <sys/socketvar.h>
#include <sys/stat.h>
#include <sys/sx.h>
#include <sys/syscallsubr.h>
#include <sys/sysctl.h>
#include <sys/sysproto.h>
#include <sys/unistd.h>
#include <sys/vnode.h>
#include <security/audit/audit.h>
#include <vm/uma.h>
#include <ddb/ddb.h>
```

Include dependency graph for kern_descrip.c:


/usr/src/sys/kern/kern_descrip.c
sys/cdefs.h
opt_compat.h
opt_ddb.h
sys/param.h
sys/systm.h
sys/conf.h
sys/fcntl.h
sys/file.h
sys/filedesc.h
sys/filio.h
sys/jail.h
sys/kernel.h
sys/limits.h
sys/lock.h
sys/malloc.h
sys/mount.h
sys/mqueue.h
sys/mutex.h
sys/namei.h
sys/priv.h
sys/proc.h
sys/resourcevar.h
sys/signalvar.h
sys/socketvar.h
sys/stat.h
sys/sx.h
sys/syscallsubr.h
sys/sysctl.h
sys/sysproto.h
sys/unistd.h
sys/vnode.h
security/audit/audit.h
vm/uma.h
ddb/ddb.h

Data Structures

- struct filedesc0
- struct getdtablesize_args
- struct dup2_args
- struct dup_args
- struct fcntl_args
- struct close_args
- struct fstat_args
- struct nfstat_args
- struct fpathconf_args
- struct flock_args

Defines

- #define NDFILE 20
- #define NDSLOTSIZE sizeof(NDSLOTTYPE)
- #define NDENTRIES (NDSLOTSIZE * __CHAR_BIT)
- #define NDSLOT(x) ((x) / NDENTRIES)
- #define NDBIT(x) ((NDSLOTTYPE)1 << ((x) % NDENTRIES))
- #define NDSLOTS(x) (((x) + NDENTRIES - 1) / NDENTRIES)
- #define OFILESIZE (sizeof(struct file *) + sizeof(char))

Enumerations

- enum dup_type { DUP_VARIABLE, DUP_FIXED }

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_descrip.c,v 1.303 2007/02/15 10:55:43 rwatson Exp $")
- static MALLOC_DEFINE (M_FILEDESC,"filedesc","Open file descriptor table")
- static MALLOC_DEFINE (M_FILEDESC_TO_LEADER,"filedesc_to_leader","file desc to leader structures")
- static MALLOC_DEFINE (M_SIGIO,"sigio","sigio structures")
- static int do_dup (struct thread *td, enum dup_type type, int old, int new, register_t *retval)
- static int fd_first_free (struct filedesc *, int, int)
- static int fd_last_used (struct filedesc *, int, int)
- static void fdgrowtable (struct filedesc *, int)
- static int fdrop_locked (struct file *fp, struct thread *td)
- static void fdunused (struct filedesc *fdp, int fd)
- static void fdused (struct filedesc *fdp, int fd)
- static int fdisused (struct filedesc *fdp, int fd)
- int getdtablesize (struct thread *td, struct getdtablesize_args *uap)
- int dup2 (struct thread *td, struct dup2_args *uap)
- int dup (struct thread *td, struct dup_args *uap)
- int fcntl (struct thread *td, struct fcntl_args *uap)
- int kern_fcntl (struct thread *td, int fd, int cmd, intptr_t arg)
- void funsetown (struct sigio **sigiop)
- void funsetownlst (struct sigiolst *sigiolst)

- int fsetown (pid_t pgid, struct sigio **sigiop)
- pid_t fgetown (struct sigio **sigiop)
- int close (struct thread *td, struct close_args *uap)
- int kern_close (struct thread *td, int fd)
- int fstat (struct thread *td, struct fstat_args *uap)
- int kern_fstat (struct thread *td, int fd, struct stat *sbp)
- int nfstat (struct thread *td, struct nfstat_args *uap)
- int fpathconf (struct thread *td, struct fpathconf_args *uap)
- int fdalloc (struct thread *td, int minfd, int *result)
- int fdavail (struct thread *td, int n)
- int falloc (struct thread *td, struct file **resultfp, int *resultfd)
- filedesc * fdinit (struct filedesc *fdp)
- static struct filedesc * fdhold (struct proc *p)
- static void fddrop (struct filedesc *fdp)
- filedesc * fdshare (struct filedesc *fdp)
- void fdunshare (struct proc *p, struct thread *td)
- filedesc * fdcopy (struct filedesc *fdp)
- void fdfree (struct thread *td)
- static int is_unsafe (struct file *fp)
- void setugidsafety (struct thread *td)
- void fdclose (struct filedesc *fdp, struct file *fp, int idx, struct thread *td)
- void fdcloseexec (struct thread *td)
- int fdcheckstd (struct thread *td)
- int closef (struct file *fp, struct thread *td)
- static __inline int _fget (struct thread *td, int fd, struct file **fpp, int flags, int hold)
- int fget (struct thread *td, int fd, struct file **fpp)
- int fget_read (struct thread *td, int fd, struct file **fpp)
- int fget_write (struct thread *td, int fd, struct file **fpp)
- static __inline int _fgetvp (struct thread *td, int fd, struct vnode **vpp, int flags)
- int fgetvp (struct thread *td, int fd, struct vnode **vpp)
- int fgetvp_read (struct thread *td, int fd, struct vnode **vpp)
- int fgetsock (struct thread *td, int fd, struct socket **spp, u_int *fflagp)
- void fputsock (struct socket *so)
- int fdrop (struct file *fp, struct thread *td)
- int flock (struct thread *td, struct flock_args *uap)
- int dupfdopen (struct thread *td, struct filedesc *fdp, int indx, int dfd, int mode, int error)
- void mountcheckdirs (struct vnode *olddp, struct vnode *newdp)
- filedesc_to_leader * filedesc_to_leader_alloc (struct filedesc_to_leader *old, struct filedesc *fdp, struct proc *leader)
- static int sysctl_kern_file (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern, KERN_FILE, file, CTLTYPE_OPAQUE|CTLFLAG_RD, 0, 0, sysctl_kern_file,"S,xfile","Entire file table")
- SYSCTL_INT (_kern, KERN_MAXFILESPERPROC, maxfilesperproc, CTLFLAG_RW,&maxfilesperproc, 0,"Maximum files allowed open per process")
- SYSCTL_INT (_kern, KERN_MAXFILES, maxfiles, CTLFLAG_RW,&maxfiles, 0,"Maximum number of files")
- SYSCTL_INT (_kern, OID_AUTO, openfiles, CTLFLAG_RD,&openfiles, 0,"System-wide number of open files")
- static void filelistinit (void *dummy)
- static int badfo_readwrite (struct file *fp, struct uio *uio, struct ucred *active_cred, int flags, struct thread *td)

- static int badfo_ioctl (struct file *fp, u_long com, void *data, struct ucred *active_cred, struct thread *td)
- static int badfo_poll (struct file *fp, int events, struct ucred *active_cred, struct thread *td)
- static int badfo_kqfilter (struct file *fp, struct knote *kn)
- static int badfo_stat (struct file *fp, struct stat *sb, struct ucred *active_cred, struct thread *td)
- static int badfo_close (struct file *fp, struct thread *td)
- static int fdopen (struct cdev *dev, int mode, int type, struct thread *td)
- static void fildesc_drvinit (void *unused)

Variables

- static uma_zone_t file_zone
- filelist filehead
- int openfiles
- sx filelist_lock
- mtx sigio_lock
- void(*) mq_fdclose (struct thread *td, int fd, struct file *fp)
- static struct mtx fdesc_mtx
- fileops badfileops
- static struct cdevsw fildesc_cdevsw

9.23.1 Define Documentation

9.23.1.1 #define NDBIT(x) ((NDSLOTTYPE)1 << ((x) % NDENTRIES))

Definition at line 111 of file kern_descrip.c.

Referenced by fdisused(), fdunused(), and fdused().

9.23.1.2 #define NDENTRIES (NDSLOTSIZE * __CHAR_BIT)

Definition at line 109 of file kern_descrip.c.

Referenced by fd_first_free(), fd_last_used(), and fdgrowtable().

9.23.1.3 #define NDFILE 20

Definition at line 107 of file kern_descrip.c.

Referenced by fdfree(), fdgrowtable(), and fdinit().

9.23.1.4 #define NDSLOT(x) ((x) / NDENTRIES)

Definition at line 110 of file kern_descrip.c.

Referenced by fd_first_free(), fd_last_used(), fdisused(), fdunused(), and fdused().

9.23.1.5 #define NDSLOTS(x) (((x) + NDENTRIES - 1) / NDENTRIES)

Definition at line 112 of file kern_descrip.c.

Referenced by fd_first_free(), fdfree(), and fdgrowtable().

9.23.1.6 #define NDSLOTSIZE sizeof(NDSLOTTYPE)

Definition at line 108 of file kern_descrip.c.

Referenced by fdgrowtable().

9.23.1.7 #define OFILESIZE (sizeof(struct file *) + sizeof(char))

Definition at line 117 of file kern_descrip.c.

Referenced by fdgrowtable().

9.23.2 Enumeration Type Documentation

9.23.2.1 enum dup_type

Enumerator:

DUP_VARIABLE

DUP_FIXED

Definition at line 88 of file kern_descrip.c.

9.23.3 Function Documentation

9.23.3.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_descrip. *c*, v 1.303 2007/02/15 10:55:43 rwatson Exp $")

9.23.3.2 static __inline int _fget (struct thread * *td*, int *fd*, struct file ** *fpp*, int *flags*, int *hold*) [static]

Definition at line 2001 of file kern_descrip.c.

References badfileops.

Referenced by _fgetvp(), fget(), fget_read(), fget_write(), and fgetsock().

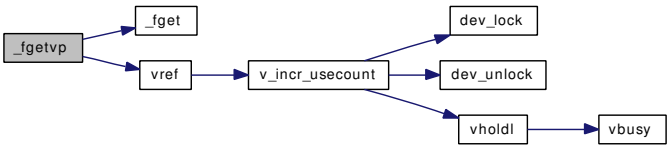
9.23.3.3 static __inline int _fgetvp (struct thread * *td*, int *fd*, struct vnode ** *vpp*, int *flags*) [static]

Definition at line 2065 of file kern_descrip.c.

References _fget(), and vref().

Referenced by fgetvp(), and fgetvp_read().

Here is the call graph for this function:



9.23.3.4 static int badfo_close (struct file * *fp*, struct thread * *td*) [static]

Definition at line 2685 of file kern_descrip.c.

9.23.3.5 static int badfo_ioctl (struct file * *fp*, u_long *com*, void * *data*, struct ucred * *active_cred*, struct thread * *td*) [static]

Definition at line 2657 of file kern_descrip.c.

9.23.3.6 static int badfo_kqfilter (struct file * *fp*, struct knote * *kn*) [static]

Definition at line 2671 of file kern_descrip.c.

9.23.3.7 static int badfo_poll (struct file * *fp*, int *events*, struct ucred * *active_cred*, struct thread * *td*) [static]

Definition at line 2664 of file kern_descrip.c.

9.23.3.8 static int badfo_readwrite (struct file * *fp*, struct uio * *uio*, struct ucred * *active_cred*, int *flags*, struct thread * *td*) [static]

Definition at line 2650 of file kern_descrip.c.

9.23.3.9 static int badfo_stat (struct file * *fp*, struct stat * *sb*, struct ucred * *active_cred*, struct thread * *td*) [static]

Definition at line 2678 of file kern_descrip.c.

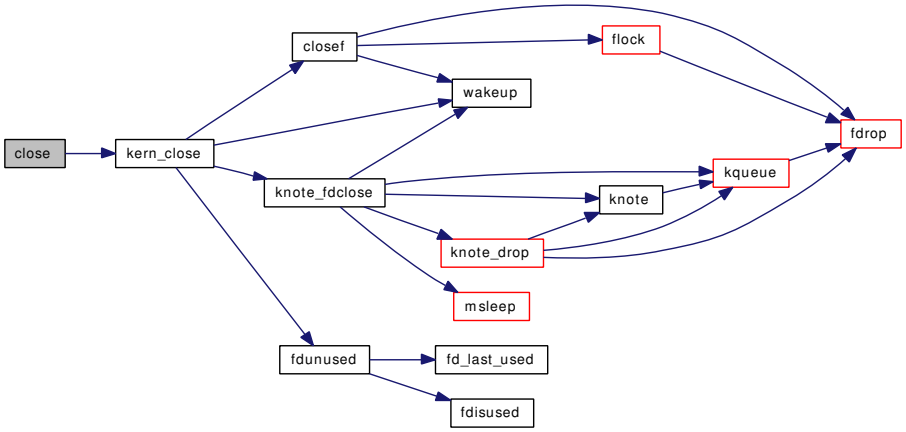
9.23.3.10 int close (struct thread * *td*, struct close_args * *uap*)

Definition at line 973 of file kern_descrip.c.

References close_args::fd, and kern_close().

Referenced by coredump().

Here is the call graph for this function:



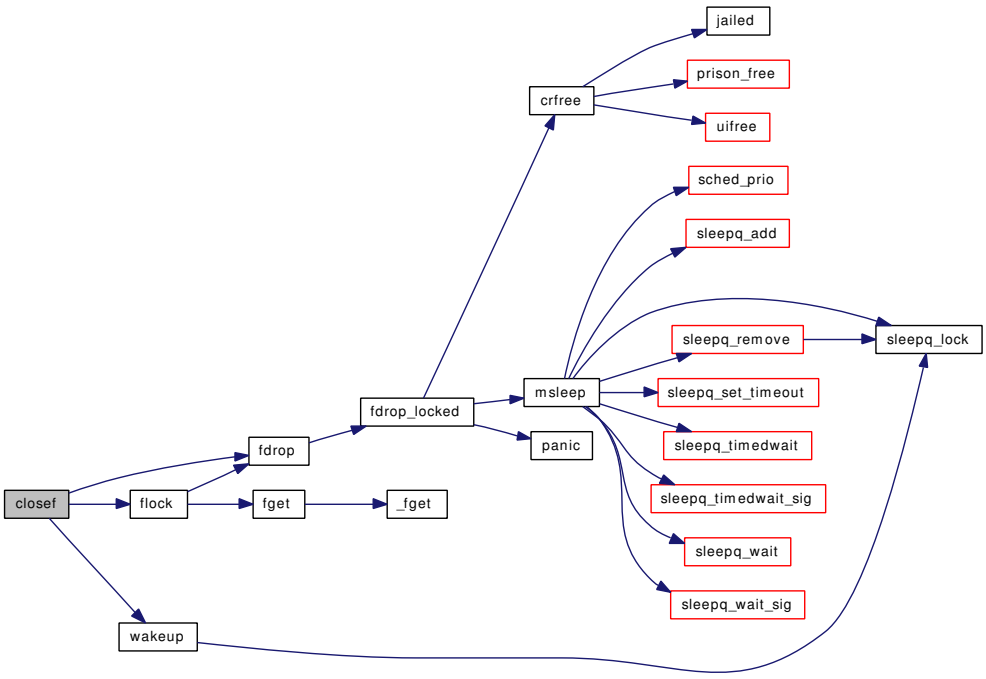
9.23.3.11 int closef (struct file * *fp*, struct thread * *td*)

Definition at line 1912 of file kern_descrip.c.

References fdrop(), flock(), and wakeup().

Referenced by do_dup(), fdcloseexec(), fdfree(), kern_close(), setugidsafety(), unp_discard(), and unp_-gc().

Here is the call graph for this function:



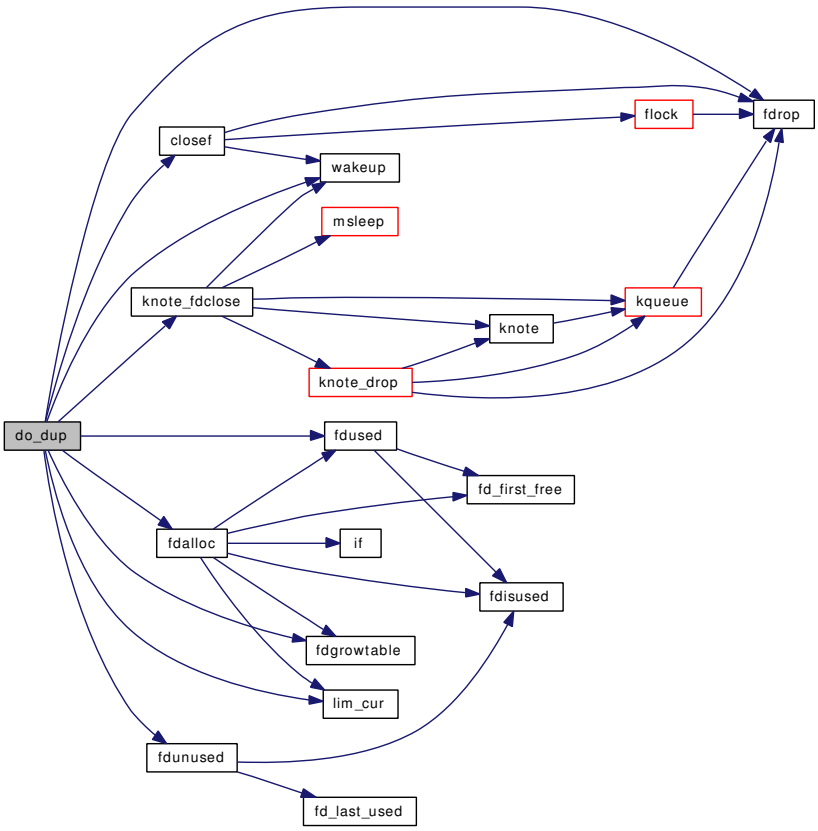
9.23.3.12 static int do_dup (struct thread * *td*, enum dup_type *type*, int *old*, int *new*, register_t * *retval*) [static]

Definition at line 608 of file kern_descrip.c.

References closef(), DUP_FIXED, DUP_VARIABLE, fdalloc(), fdgrowtable(), fdrop(), fdunused(), fdused(), knote_fdclose(), lim_cur(), maxfilesperproc, mq_fdclose, and wakeup().

Referenced by dup(), dup2(), fdcheckstd(), and kern_fcntl().

Here is the call graph for this function:

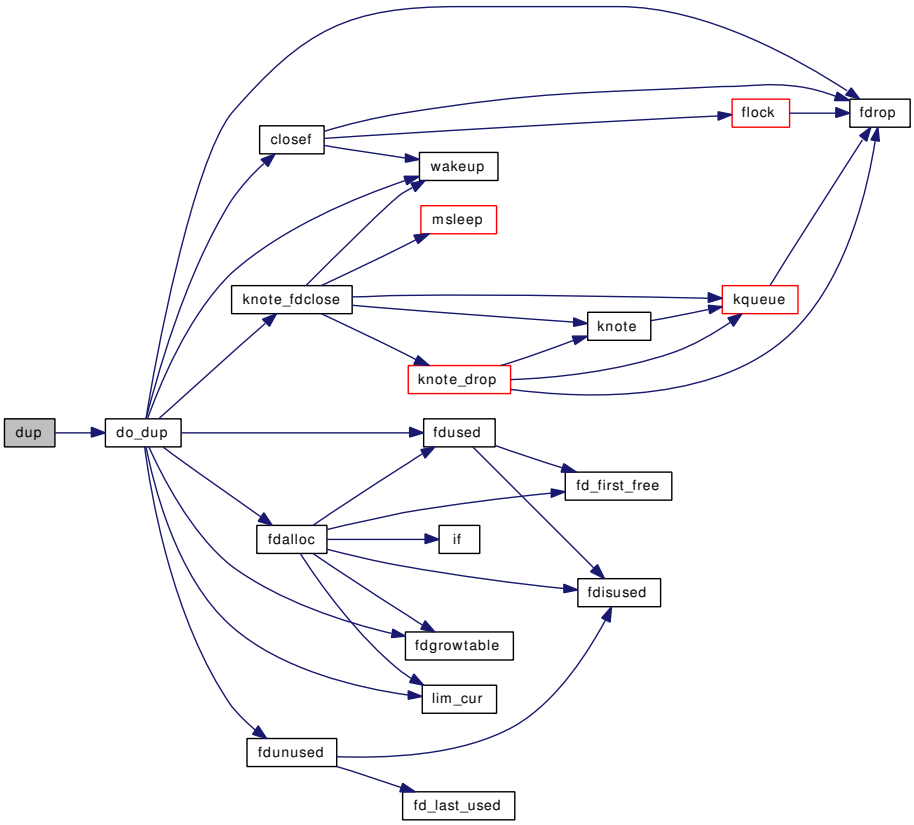


9.23.3.13 int dup (struct thread * *td*, struct dup_args * *uap*)

Definition at line 303 of file kern_descrip.c.

References do_dup(), DUP_VARIABLE, and dup_args::fd.

Here is the call graph for this function:

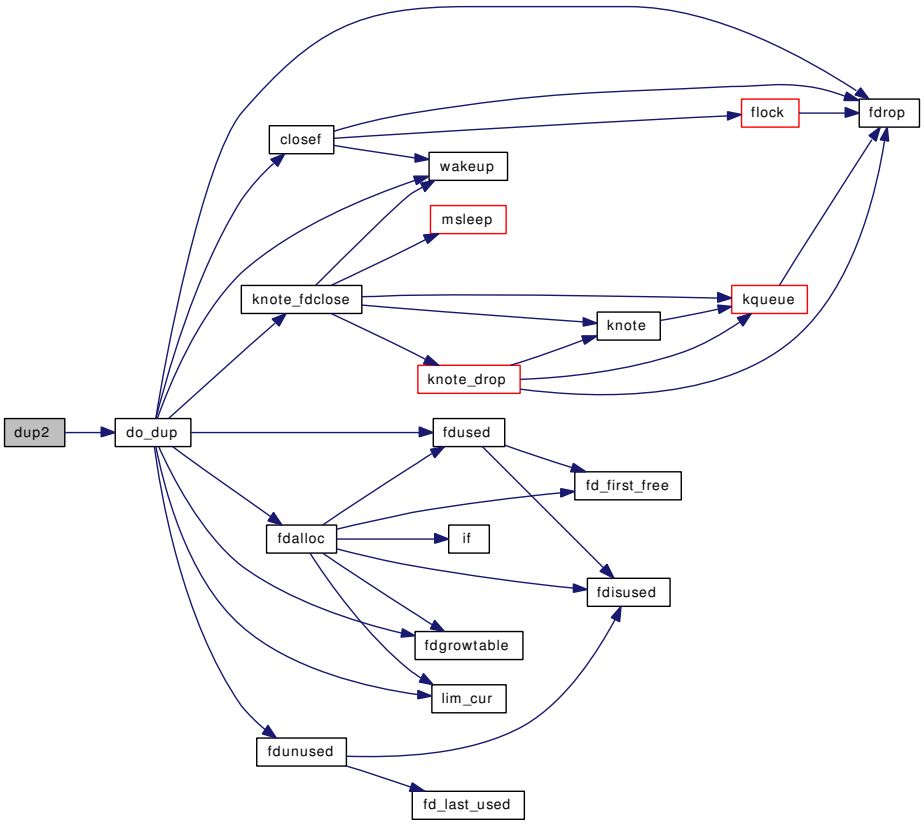


9.23.3.14 int dup2 (struct thread * *td*, struct dup2_args * *uap*)

Definition at line 283 of file kern_descrip.c.

References do_dup(), DUP_FIXED, dup2_args::from, and dup2_args::to.

Here is the call graph for this function:



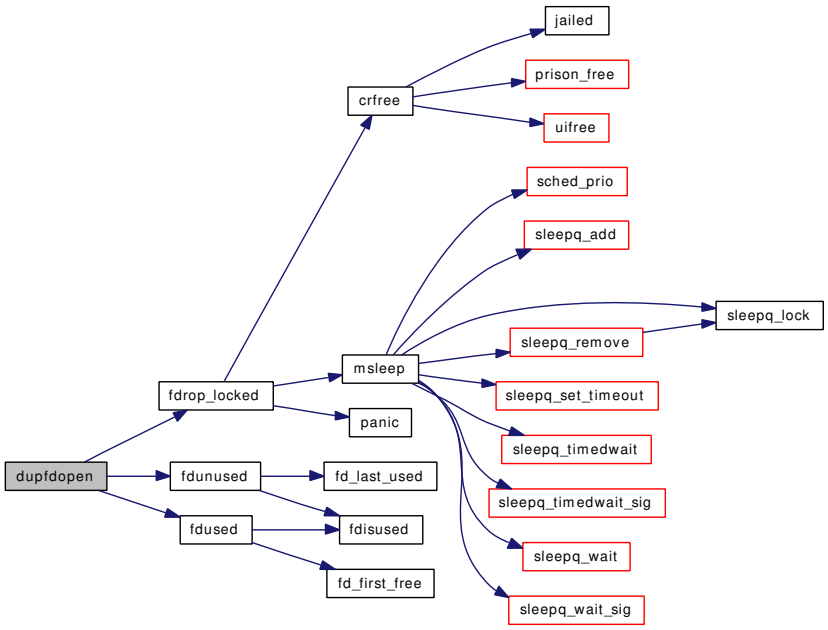
9.23.3.15 int dupfdopen (struct thread * *td*, struct filedesc * *fdp*, int *indx*, int *dfd*, int *mode*, int *error*)

Definition at line 2280 of file kern_descrip.c.

References fdrop_locked(), fdunused(), and fdused().

Referenced by kern_open().

Here is the call graph for this function:



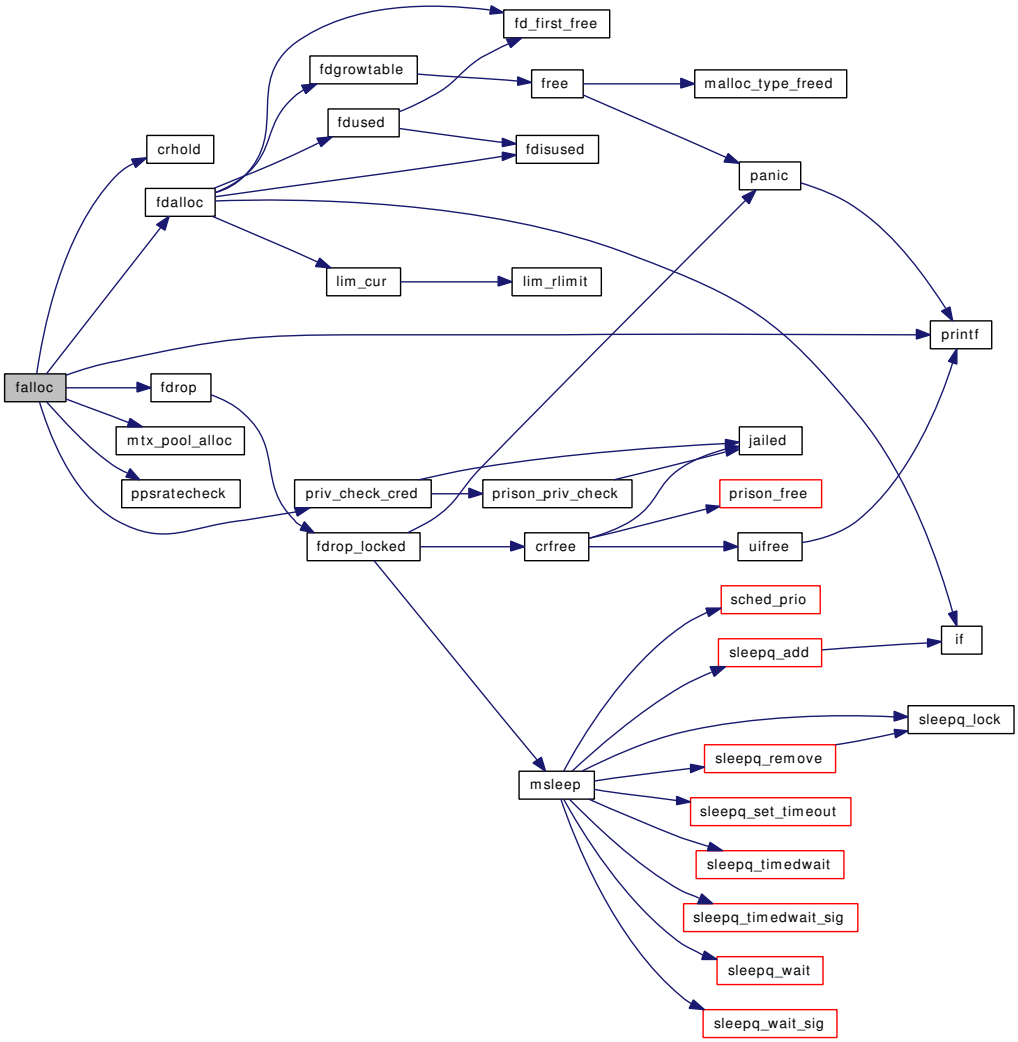
9.23.3.16 int falloc (struct thread * *td*, struct file ** *resultfp*, int * *resultfd*)

Definition at line 1342 of file kern_descrip.c.

References badfileops, crhold(), fdalloc(), fdrop(), file_zone, filehead, filelist_lock, maxfiles, mtx_pool_-alloc(), mtxpool_sleep, openfiles, ppsratecheck(), printf(), and priv_check_cred().

Referenced by fdcheckstd(), fhopen(), kern_accept(), kern_open(), kmq_open(), kqueue(), pipe(), sctp_-peeloff(), socket(), and socketpair().

Here is the call graph for this function:

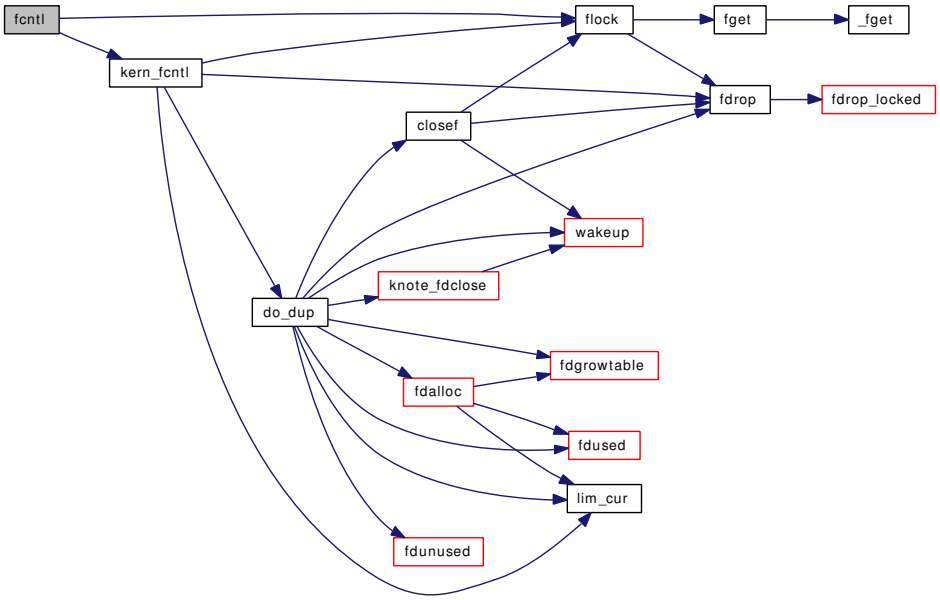


9.23.3.17 int fcntl (struct thread * *td*, struct fcntl_args * *uap*)

Definition at line 324 of file kern_descrip.c.

References fcntl_args::arg, fcntl_args::cmd, fcntl_args::fd, flock(), and kern_fcntl().

Here is the call graph for this function:



9.23.3.18 static int fd_first_free (struct filedesc *, int, int) [static]

Definition at line 151 of file kern_descrip.c.

References mask, NDENTRIES, NDSLOT, and NDSLOTS.

Referenced by fdalloc(), and fdused().

9.23.3.19 static int fd_last_used (struct filedesc *, int, int) [static]

Definition at line 178 of file kern_descrip.c.

References mask, NDENTRIES, and NDSLOT.

Referenced by fdunused().

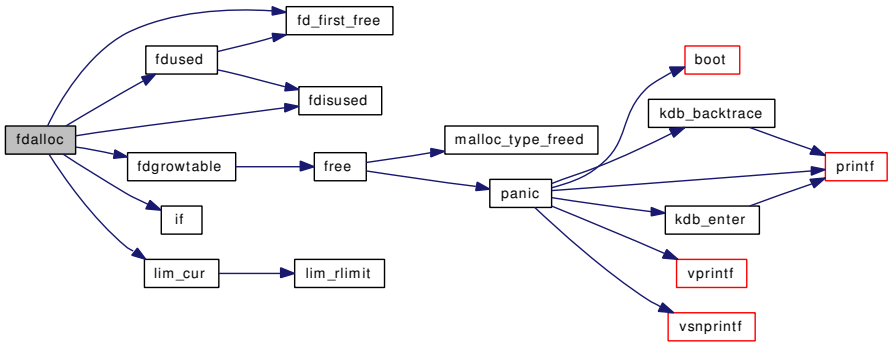
9.23.3.20 int fdalloc (struct thread * *td*, int *minfd*, int * *result*)

Definition at line 1261 of file kern_descrip.c.

References fd_first_free(), fdgrowtable(), fdisused(), fdused(), if(), lim_cur(), and maxfilesperproc.

Referenced by do_dup(), falloc(), and unp_externalize().

Here is the call graph for this function:



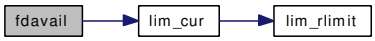
9.23.3.21 int fdavail (struct thread * *td*, int *n*)

Definition at line 1310 of file kern_descrip.c.

References lim_cur(), and maxfilesperproc.

Referenced by unp_externalize().

Here is the call graph for this function:

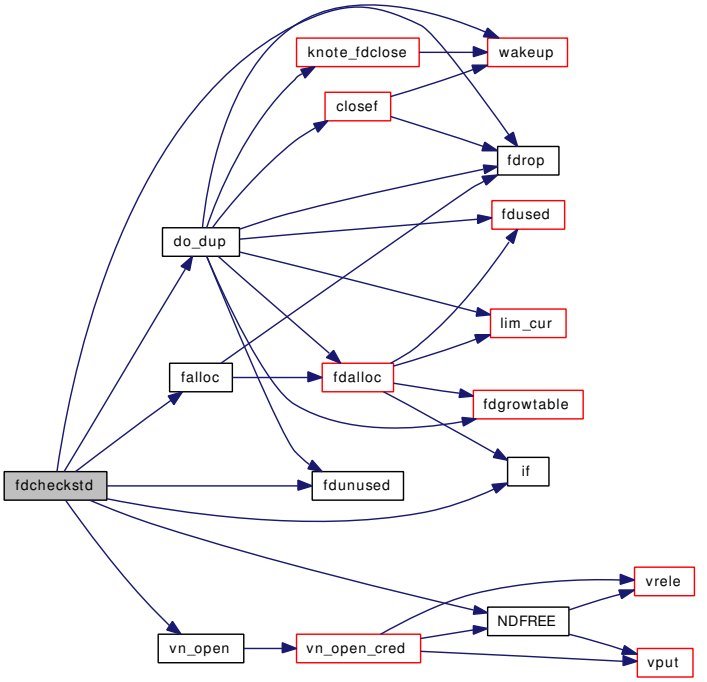


9.23.3.22 int fdcheckstd (struct thread * *td*)

Definition at line 1836 of file kern_descrip.c.

References badfileops, do_dup(), DUP_FIXED, falloc(), fdrop(), fdunused(), if(), NDFREE(), vn_open(), and vnops.

Here is the call graph for this function:



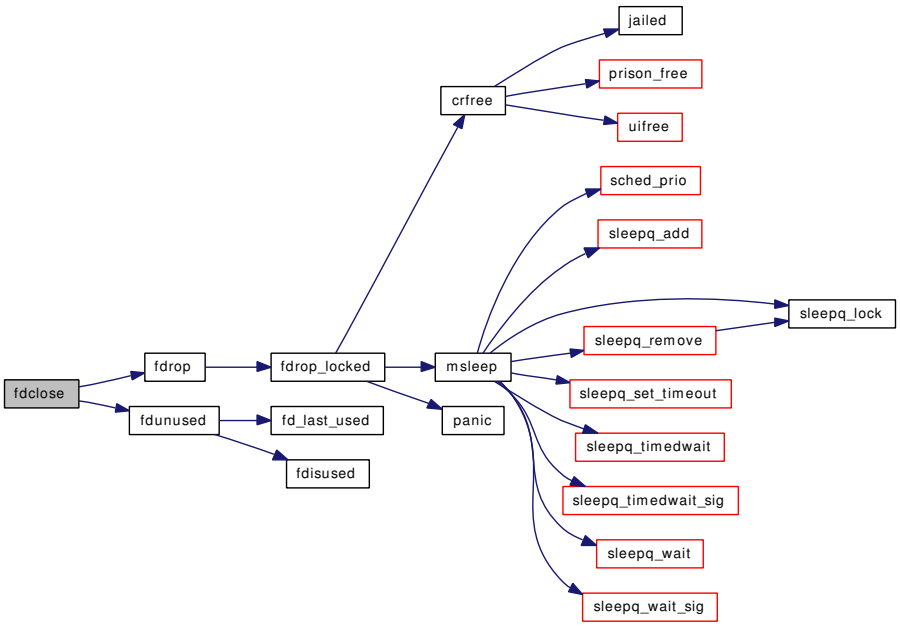
9.23.3.23 void fdclose (struct filedesc * *fdp*, struct file * *fp*, int *idx*, struct thread * *td*)

Definition at line 1769 of file kern_descrip.c.

References fdrop(), and fdunused().

Referenced by accept1(), fhopen(), kern_accept(), kern_open(), kmq_open(), pipe(), sctp_peeloff(), socket(), and socketpair().

Here is the call graph for this function:

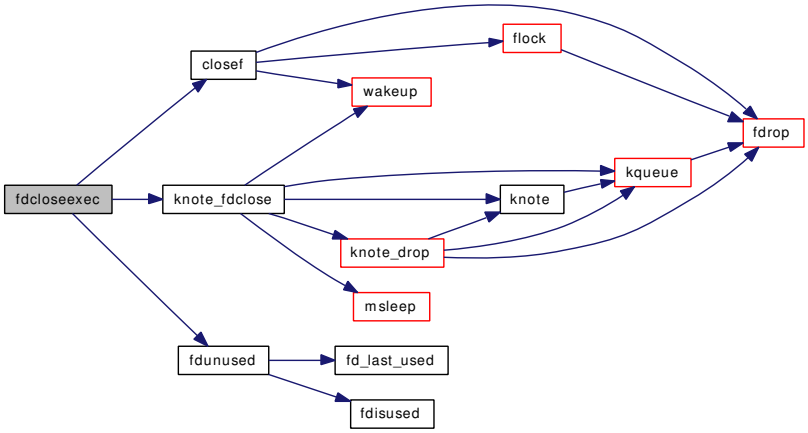


9.23.3.24 void fdcloseexec (struct thread * *td*)

Definition at line 1787 of file kern_descrip.c.

References closef(), fdunused(), knote_fdclose(), and mq_fdclose.

Here is the call graph for this function:



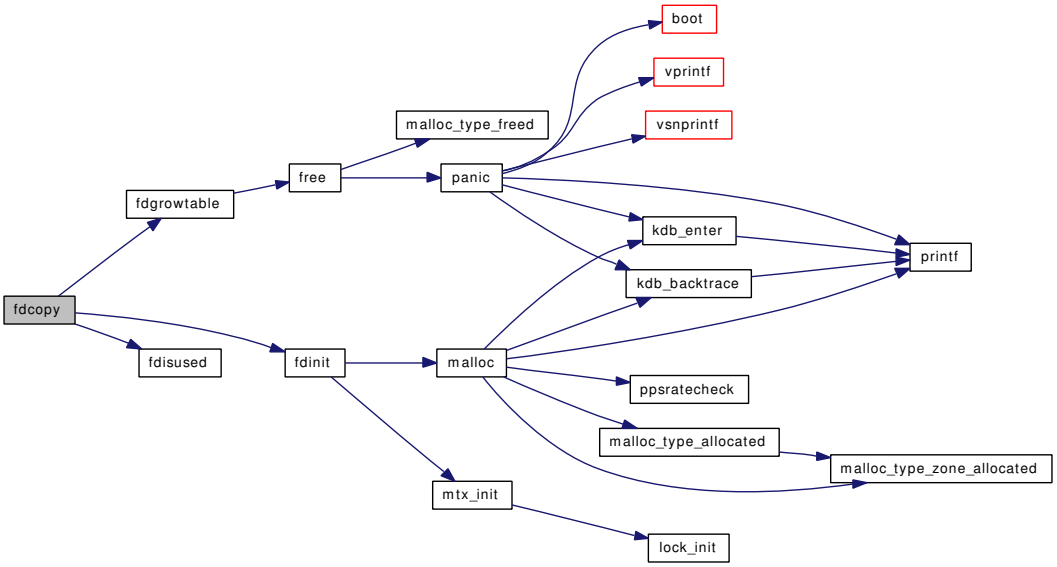
9.23.3.25 struct filedesc* fdcopy (struct filedesc * *fdp*)

Definition at line 1505 of file kern_descrip.c.

References fdgrowtable(), fdinit(), and fdisused().

Referenced by fdunshare(), and fork1().

Here is the call graph for this function:



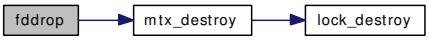
9.23.3.26 static void fddrop (struct filedesc * *fdp*) [static]

Definition at line 1454 of file kern_descrip.c.

References fdesc_mtx, and mtx_destroy().

Referenced by fdfree(), mountcheckdirs(), and sysctl_kern_file().

Here is the call graph for this function:



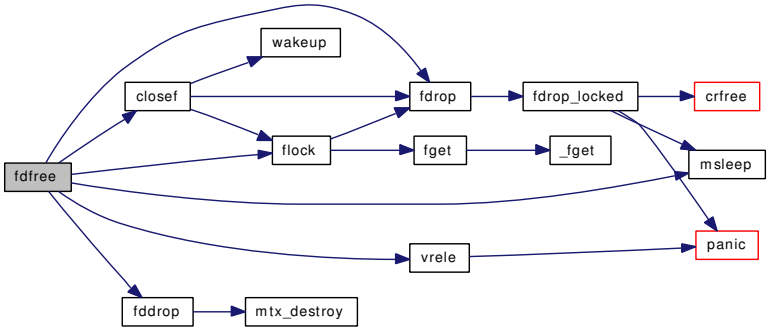
9.23.3.27 void fdfree (struct thread * *td*)

Definition at line 1555 of file kern_descrip.c.

References closef(), fddrop(), fdesc_mtx, fdrop(), flock(), msleep(), NDFILE, NDSLOTS, and vrele().

Referenced by exit1(), fdunshare(), and fork1().

Here is the call graph for this function:



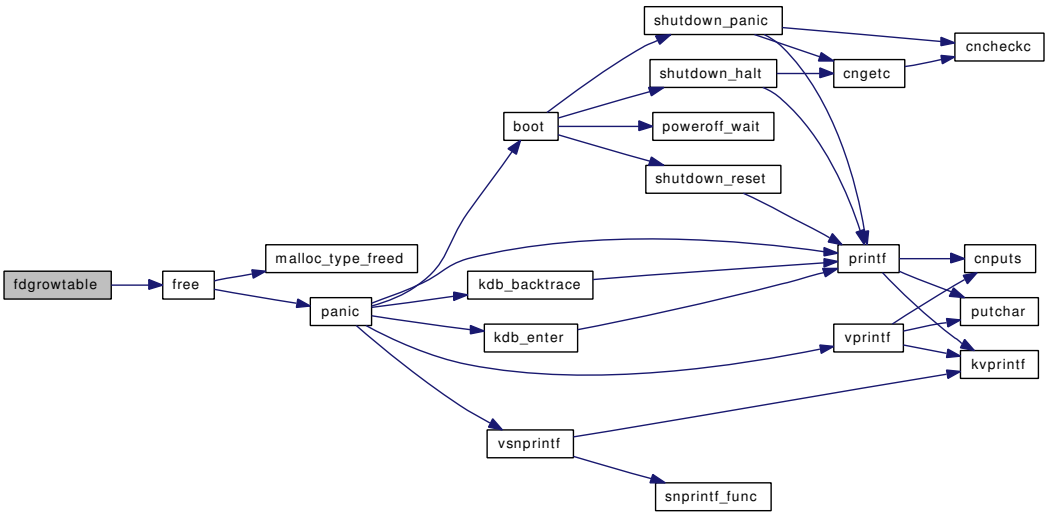
9.23.3.28 static void fdgrowtable (struct filedesc *, int) [static]

Definition at line 1199 of file kern_descrip.c.

References free(), NDENTRIES, NDFILE, NDSLOTS, NDSLOTSIZE, and OFILESIZE.

Referenced by do_dup(), fdalloc(), and fdcopy().

Here is the call graph for this function:



9.23.3.29 static struct filedesc* fdhold (struct proc * *p*) [static]

Definition at line 1441 of file kern_descrip.c.

References fdesc_mtx.

Referenced by mountcheckdirs(), and sysctl_kern_file().

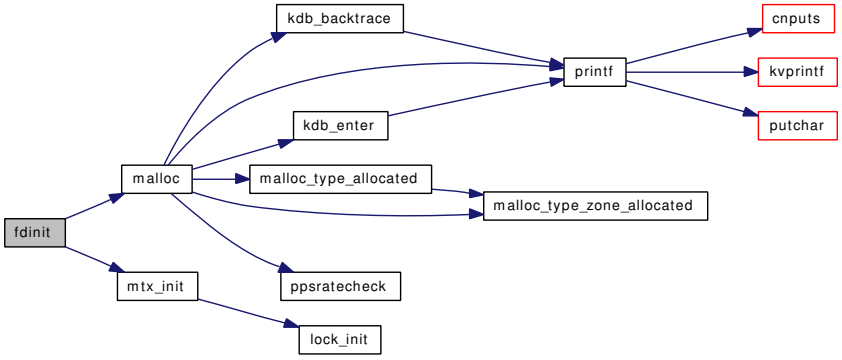
9.23.3.30 struct filedesc* fdinit (struct filedesc * *fdp*)

Definition at line 1408 of file kern_descrip.c.

References malloc(), mtx_init(), and NDFILE.

Referenced by fdcopy(), fork1(), and proc0_init().

Here is the call graph for this function:



9.23.3.31 static int fdisused (struct filedesc * *fdp*, int *fd*) [static]

Definition at line 201 of file kern_descrip.c.

References NDBIT, and NDSLOT.

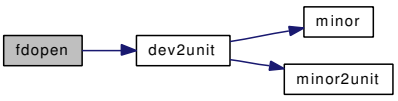
Referenced by fdalloc(), fdcopy(), fdunused(), and fdused().

9.23.3.32 static int fdopen (struct cdev * *dev*, int *mode*, int *type*, struct thread * *td*) [static]

Definition at line 2717 of file kern_descrip.c.

References dev2unit().

Here is the call graph for this function:



9.23.3.33 int fdrop (struct file * *fp*, struct thread * *td*)

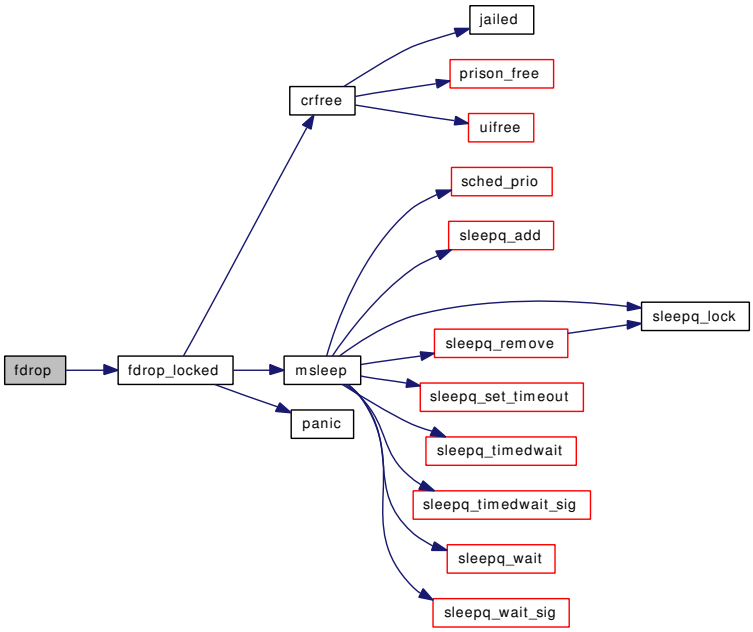
Definition at line 2161 of file kern_descrip.c.

References fdrop_locked().

Referenced by __acl_aclcheck_fd(), __acl_delete_fd(), __acl_get_fd(), __acl_set_fd(), _getmq(), accept1(), aio_aqueue(), aio_cancel(), aio_free_entry(), closef(), do_dup(), extattr_delete_fd(), extattr_get_fd(), extattr_list_fd(), extattr_set_fd(), falloc(), fchdir(), fchflags(), fchmod(), fchown(), fdcheckstd(), fdclose(), fdfree(), fhopen(), flock(), fpathconf(), fsync(), ftruncate(), getdirentries(), kern_accept(), kern_bind(), kern_connect(), kern_fcntl(), kern_fstat(), kern_fstatfs(), kern_futimes(), kern_getpeername(), kern_getsockname(), kern_getsockopt(), kern_ioctl(), kern_kevent(), kern_open(), kern_preadv(), kern_pwritev(), kern_readv(), kern_recvit(), kern_sendfile(), kern_setsockopt(), kern_writev(),

kmq_notify(), kmq_open(), kmq_setattr(), kmq_timedreceive(), kmq_timedsend(), knote_drop(), kqfd_-register(), kqueue(), kqueue_register(), listen(), lseek(), pipe(), sctp_generic_sendmsg(), sctp_peeloff(), shutdown(), socket(), and socketpair().

Here is the call graph for this function:



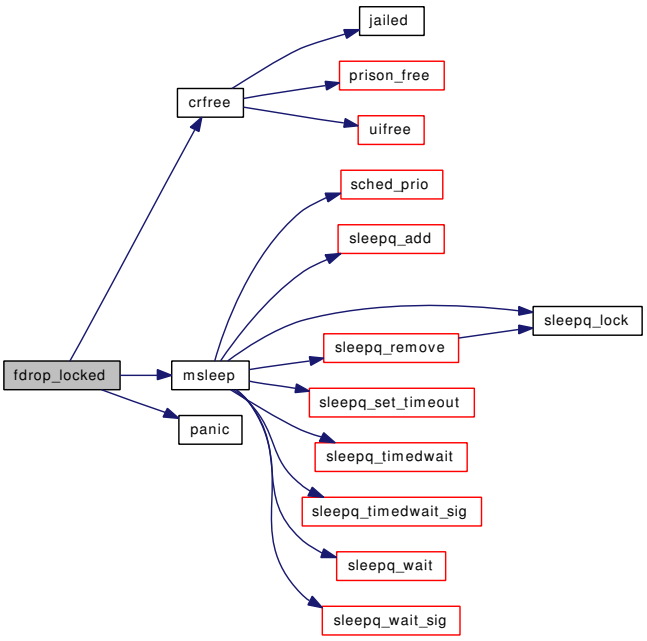
9.23.3.34 static int fdrop_locked (struct file * *fp*, struct thread * *td*) [static]

Definition at line 2174 of file kern_descrip.c.

References badfileops, crfree(), file_zone, filelist_lock, msleep(), openfiles, and panic().

Referenced by dupfdopen(), and fdrop().

Here is the call graph for this function:



9.23.3.35 struct filedesc* fdshare (struct filedesc * *fdp*)

Definition at line 1472 of file kern_descrip.c.

Referenced by fork1().

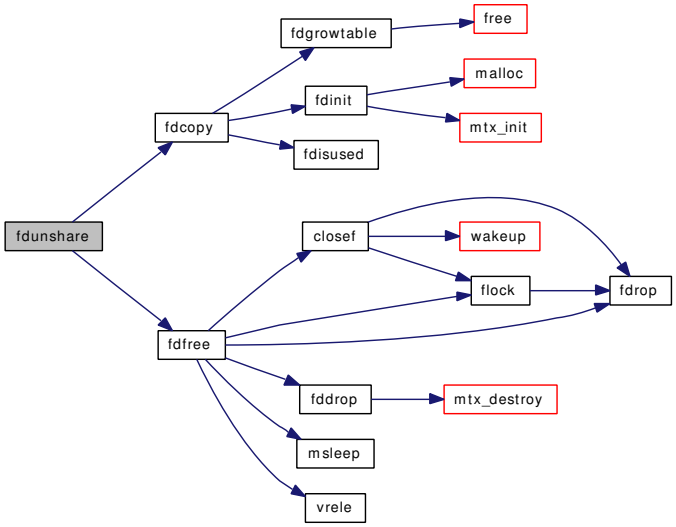
9.23.3.36 void fdunshare (struct proc * *p*, struct thread * *td*)

Definition at line 1484 of file kern_descrip.c.

References fdcopy(), and fdfree().

Referenced by fork1().

Here is the call graph for this function:



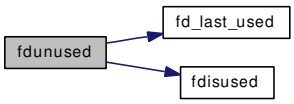
9.23.3.37 static void fdunused (struct filedesc * *fdp*, int *fd*) [static]

Definition at line 228 of file kern_descrip.c.

References fd_last_used(), fdisused(), NDBIT, and NDSLOT.

Referenced by do_dup(), dupfdopen(), fdcheckstd(), fdclose(), fdcloseexec(), kern_close(), and setugidsafety().

Here is the call graph for this function:



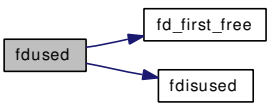
9.23.3.38 static void fdused (struct filedesc * *fdp*, int *fd*) [static]

Definition at line 212 of file kern_descrip.c.

References fd_first_free(), fdisused(), NDBIT, and NDSLOT.

Referenced by do_dup(), dupfdopen(), and fdalloc().

Here is the call graph for this function:



9.23.3.39 int fget (struct thread * *td*, int *fd*, struct file ** *fpp*)

Definition at line 2037 of file kern_descrip.c.

References _fget().

Referenced by aio_aqueue(), aio_cancel(), flock(), fpathconf(), getmq(), kern_fstat(), kern_ioctl(), kern_kevent(), kqfd_register(), kqueue_register(), and lseek().

Here is the call graph for this function:

9.23.3.40 int fget_read (struct thread * *td*, int *fd*, struct file ** *fpp*)

Definition at line 2044 of file kern_descrip.c.

References _fget().

Referenced by aio_aqueue(), getmq_read(), kern_preadv(), and kern_readv().

Here is the call graph for this function:

9.23.3.41 int fget_write (struct thread * *td*, int *fd*, struct file ** *fpp*)

Definition at line 2051 of file kern_descrip.c.

References _fget().

Referenced by aio_aqueue(), getmq_write(), kern_pwritev(), and kern_writev().

Here is the call graph for this function:

9.23.3.42 pid_t fgetown (struct sigio ** *sigiop*)

Definition at line 949 of file kern_descrip.c.

Referenced by kern_accept(), kqueue_ioctl(), logioctl(), pipe_ioctl(), sctp_peeloff(), soo_ioctl(), and ttioctl().

9.23.3.43 int fgetsock (struct thread * *td*, int *fd*, struct socket ** *spp*, u_int * *fflagp*)

Definition at line 2118 of file kern_descrip.c.

References _fget().

Referenced by sctp_peeloff().

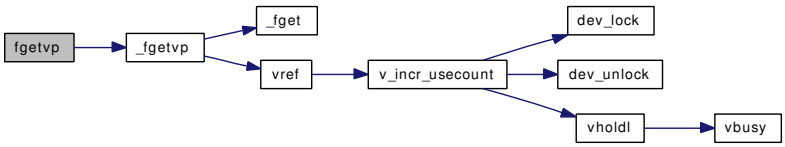
Here is the call graph for this function:

9.23.3.44 int fgetvp (struct thread * *td*, int *fd*, struct vnode ** *vpp*)

Definition at line 2084 of file kern_descrip.c.

References _fgetvp().

Here is the call graph for this function:



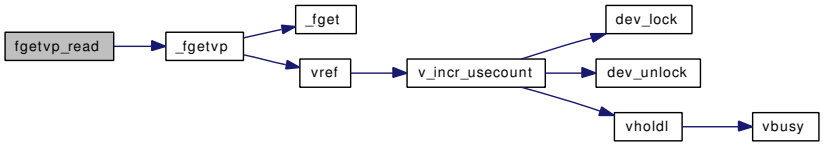
9.23.3.45 int fgetvp_read (struct thread * *td*, int *fd*, struct vnode ** *vpp*)

Definition at line 2091 of file kern_descrip.c.

References _fgetvp().

Referenced by kern_sendfile().

Here is the call graph for this function:

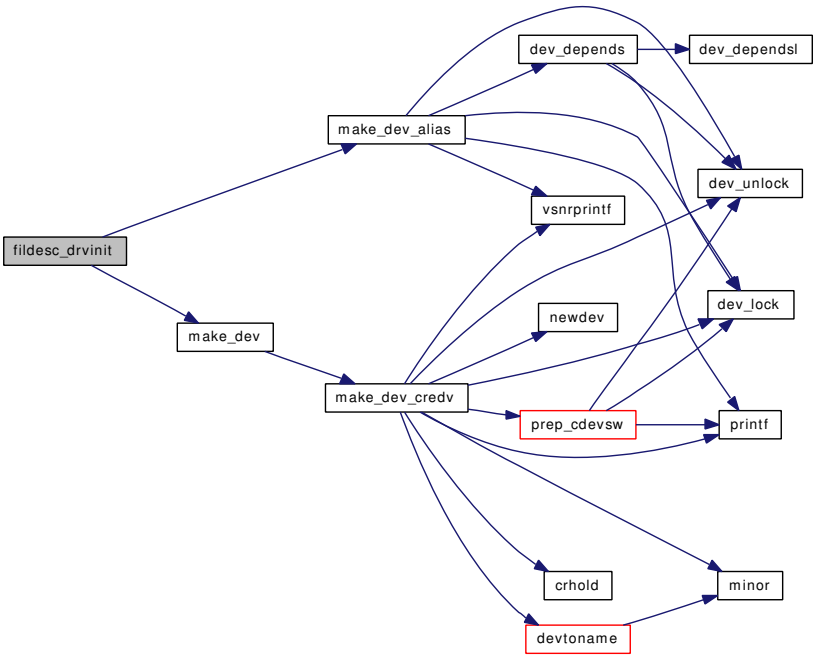


9.23.3.46 static void fildesc_drvinit (void * *unused*) [static]

Definition at line 2740 of file kern_descrip.c.

References make_dev(), and make_dev_alias().

Here is the call graph for this function:



9.23.3.47 struct filedesc_to_leader* filedesc_to_leader_alloc (struct filedesc_to_leader * *old*, struct filedesc * *fdp*, struct proc * *leader*)

Definition at line 2417 of file kern_descrip.c.

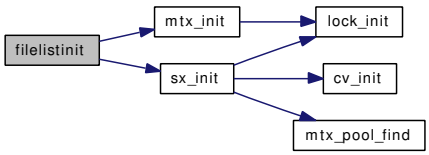
Referenced by fork1().

9.23.3.48 static void filelistinit (void * *dummy*) [static]

Definition at line 2636 of file kern_descrip.c.

References fdesc_mtx, file_zone, filelist_lock, mtx_init(), sigio_lock, and sx_init().

Here is the call graph for this function:



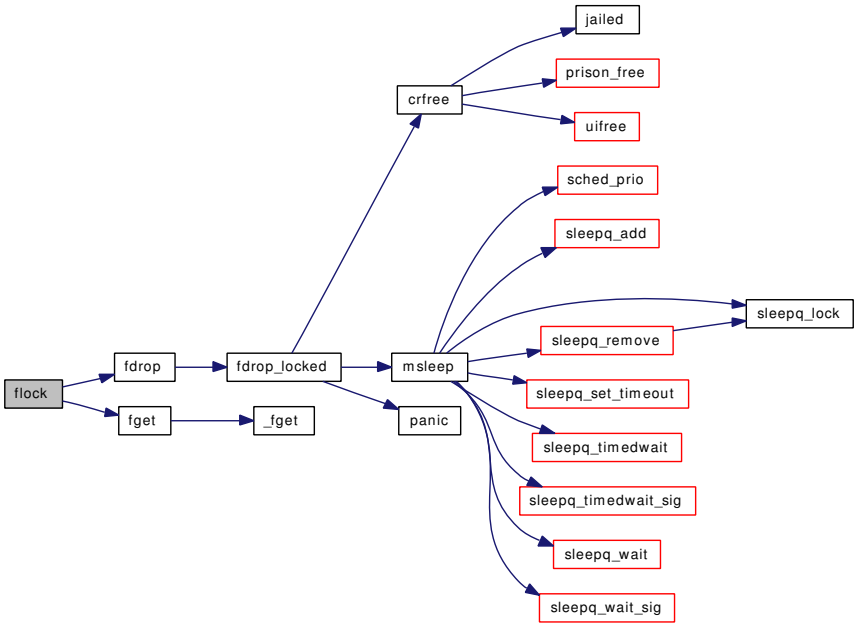
9.23.3.49 int flock (struct thread * *td*, struct flock_args * *uap*)

Definition at line 2231 of file kern_descrip.c.

References flock_args::fd, fdrop(), fget(), Giant, and flock_args::how.

Referenced by closef(), coredump(), fcntl(), fdfree(), fhopen(), kern_fcntl(), kern_open(), lf_advlock(), and vn_closefile().

Here is the call graph for this function:

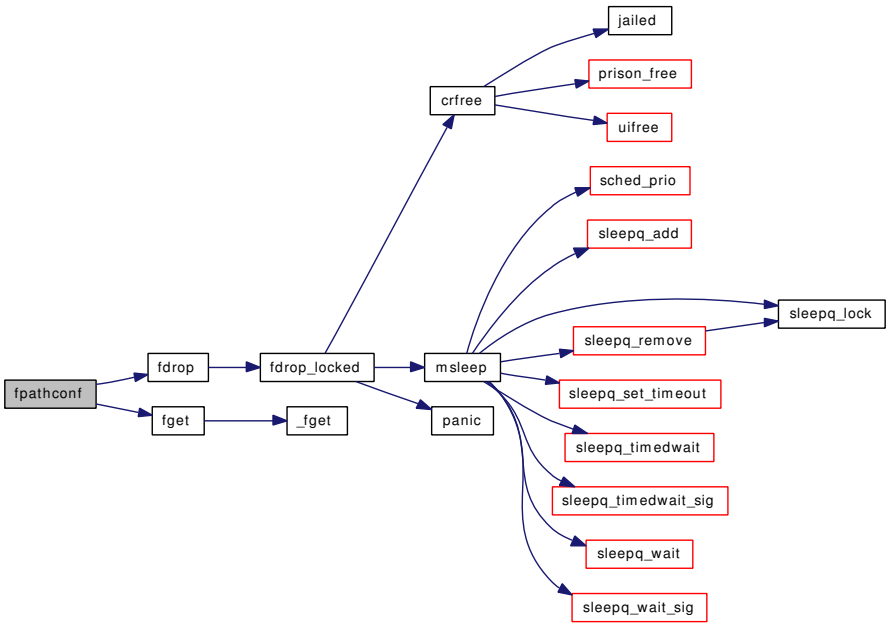


9.23.3.50 int fpathconf (struct thread * *td*, struct fpathconf_args * *uap*)

Definition at line 1156 of file kern_descrip.c.

References async_io_version, fpathconf_args::fd, fdrop(), fget(), and fpathconf_args::name.

Here is the call graph for this function:



9.23.3.51 void fputsock (struct socket * *so*)

Definition at line 2151 of file kern_descrip.c.

Referenced by sctp_peeloff().

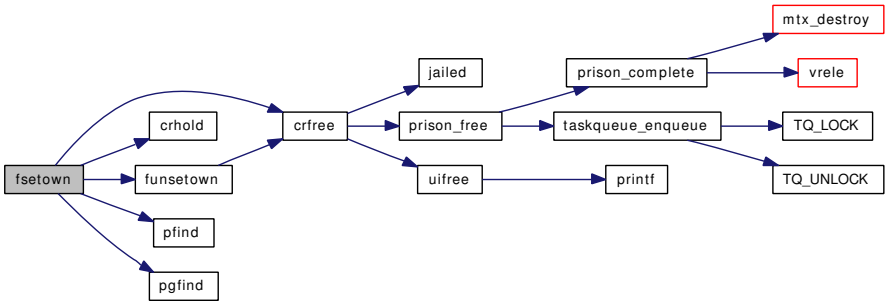
9.23.3.52 int fsetown (pid_t *pgid*, struct sigio ** *sigiop*)

Definition at line 844 of file kern_descrip.c.

References crfree(), crhold(), funsetown(), pfind(), pgfind(), proctree_lock, and ret.

Referenced by kern_accept(), kqueue_ioctl(), logioctl(), logopen(), pipe_ioctl(), sctp_peeloff(), soo_ioctl(), and ttioctl().

Here is the call graph for this function:

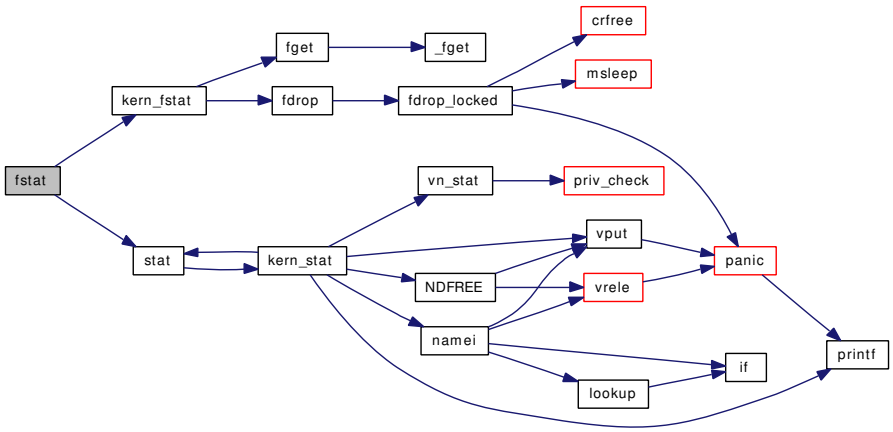


9.23.3.53 int fstat (struct thread * *td*, struct fstat_args * *uap*)

Definition at line 1085 of file kern_descrip.c.

References fstat_args::fd, kern_fstat(), fstat_args::sb, and stat().

Here is the call graph for this function:



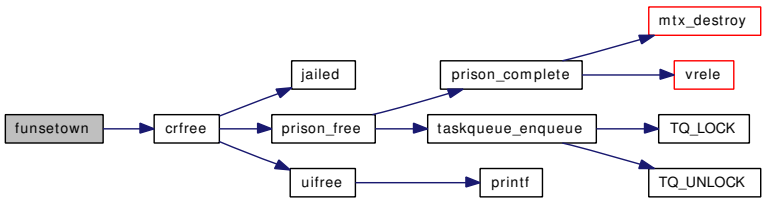
9.23.3.54 void funsetown (struct sigio ** *sigiop*)

Definition at line 747 of file kern_descrip.c.

References crfree().

Referenced by fsetown(), logclose(), pipe_close(), soclose(), and tty_close().

Here is the call graph for this function:



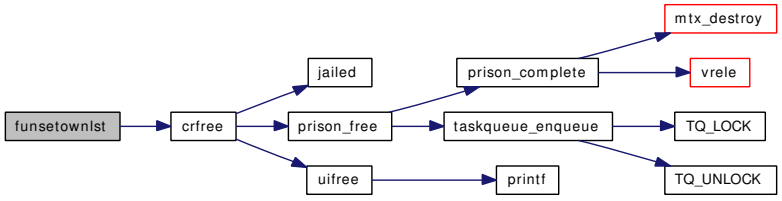
9.23.3.55 void funsetownlst (struct sigiolst * *sigiolst*)

Definition at line 783 of file kern_descrip.c.

References crfree().

Referenced by exit1(), and pgdelete().

Here is the call graph for this function:

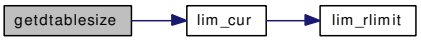


9.23.3.56 int getdtablesize (struct thread * *td*, struct getdtablesize_args * *uap*)

Definition at line 255 of file kern_descrip.c.

References lim_cur(), and maxfilesperproc.

Here is the call graph for this function:



9.23.3.57 static int is_unsafe (struct file * *fp*) [static]

Definition at line 1707 of file kern_descrip.c.

Referenced by setugidsafety().

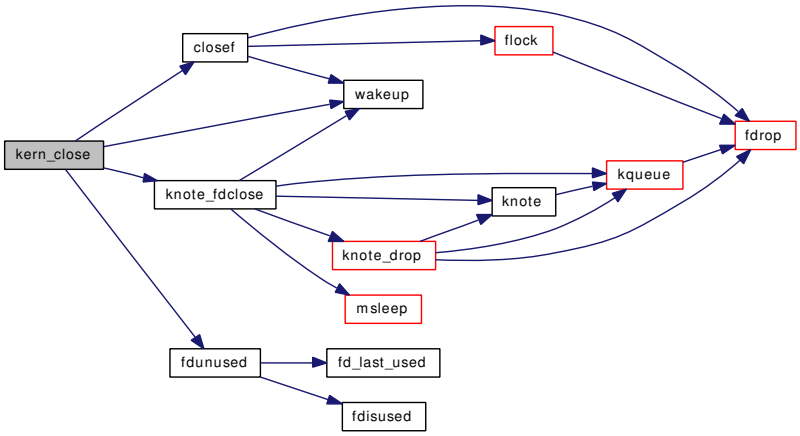
9.23.3.58 int kern_close (struct thread * *td*, int *fd*)

Definition at line 982 of file kern_descrip.c.

References closef(), fdunused(), knote_fdclose(), mq_fdclose, and wakeup().

Referenced by close().

Here is the call graph for this function:



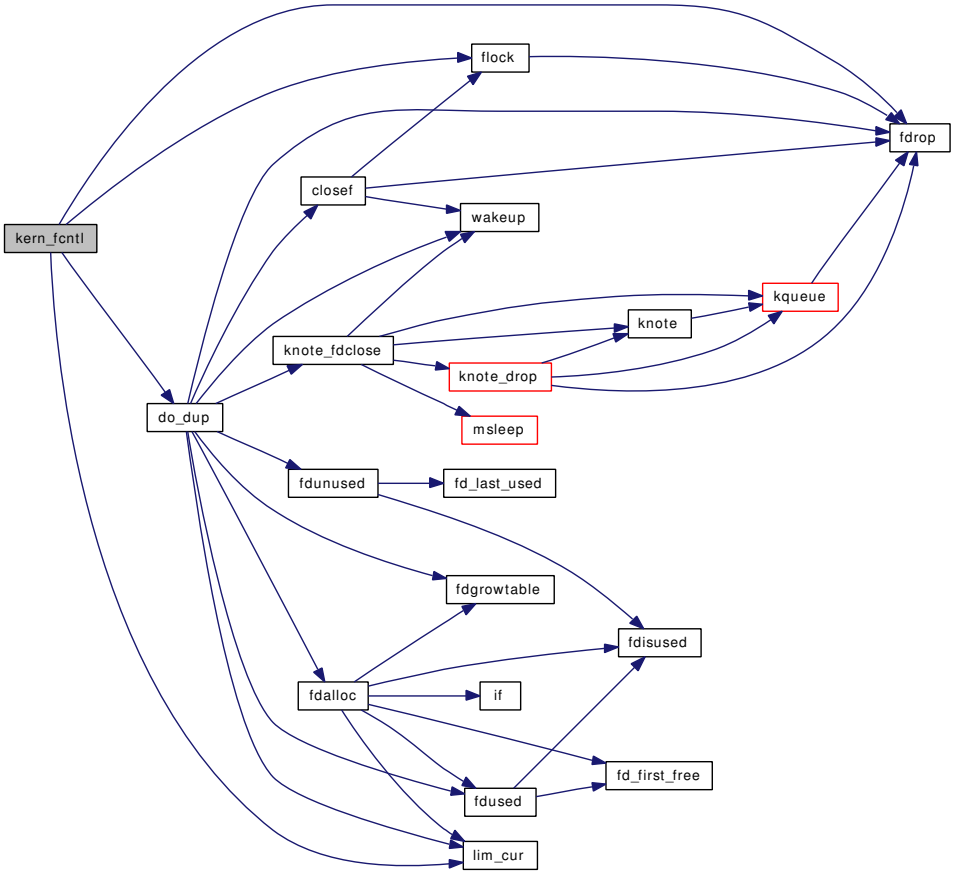
9.23.3.59 int kern_fcntl (struct thread * *td*, int *fd*, int *cmd*, intptr_t *arg*)

Definition at line 353 of file kern_descrip.c.

References do_dup(), DUP_VARIABLE, fdrop(), flock(), Giant, lim_cur(), and maxfilesperproc.

Referenced by fcntl().

Here is the call graph for this function:



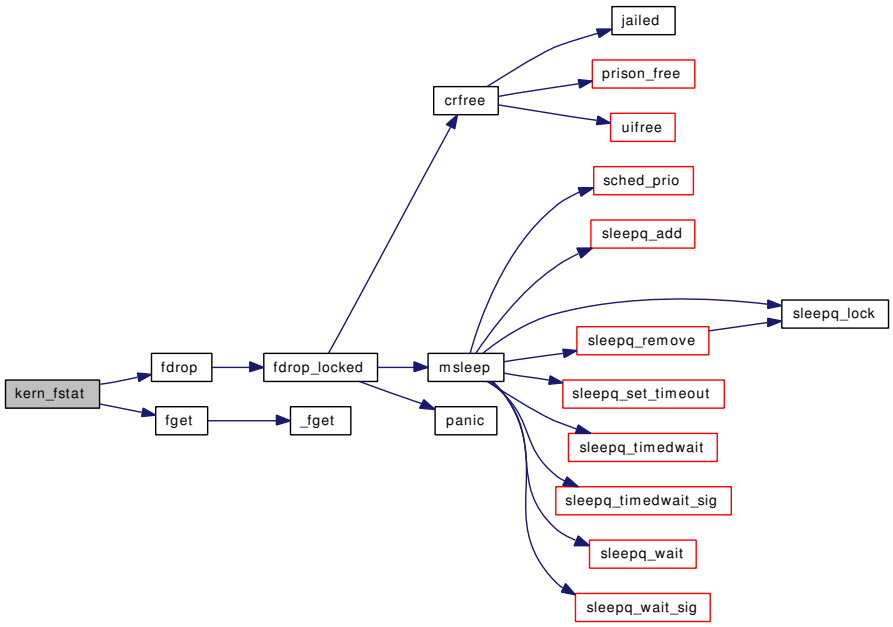
9.23.3.60 int kern_fstat (struct thread * *td*, int *fd*, struct stat * *sbp*)

Definition at line 1097 of file kern_descrip.c.

References fdrop(), and fget().

Referenced by fstat(), and nfstat().

Here is the call graph for this function:



9.23.3.61 static MALLOC_DEFINE (M_SIGIO, "sigio", "sigio structures") [static]

9.23.3.62 static MALLOC_DEFINE (M_FILEDESC_TO_LEADER, "filedesc_to_leader", "file desc to leader structures") [static]

9.23.3.63 static MALLOC_DEFINE (M_FILEDESC, "filedesc", "Open file descriptor table") [static]

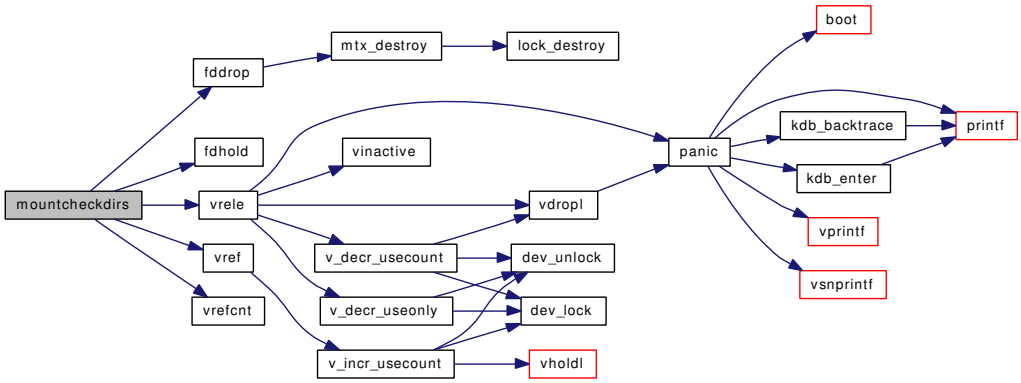
9.23.3.64 void mountcheckdirs (struct vnode * *olddp*, struct vnode * *newdp*)

Definition at line 2378 of file kern_descrip.c.

References allproc_lock, fddrop(), fdhold(), rootvnode, vref(), vrefcnt(), and vrele().

Referenced by dounmount(), and vfs_domount().

Here is the call graph for this function:

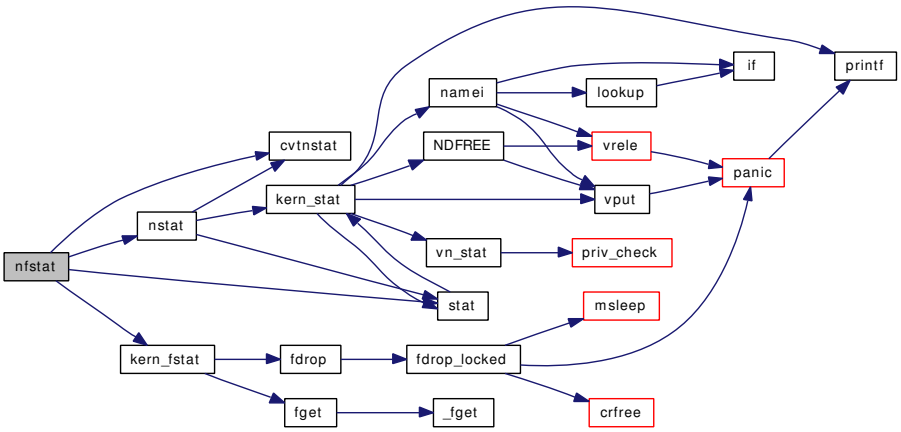


9.23.3.65 int nfstat (struct thread * *td*, struct nfstat_args * *uap*)

Definition at line 1128 of file kern_descrip.c.

References cvtnstat(), nfstat_args::fd, kern_fstat(), nstat(), nfstat_args::sb, and stat().

Here is the call graph for this function:

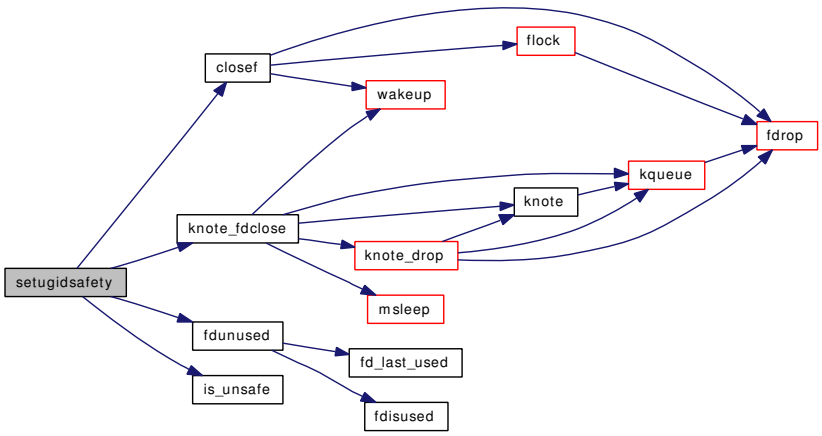


9.23.3.66 void setugidsafety (struct thread * *td*)

Definition at line 1722 of file kern_descrip.c.

References closef(), fdunused(), is_unsafe(), and knote_fdclose().

Here is the call graph for this function:



9.23.3.67 SYSCTL_INT (_kern, OID_AUTO, openfiles, CTLFLAG_RD, & *openfiles*, 0, "System-wide number of open files")

9.23.3.68 SYSCTL_INT (_kern, KERN_MAXFILES, maxfiles, CTLFLAG_RW, & *maxfiles*, 0, "Maximum number of files")

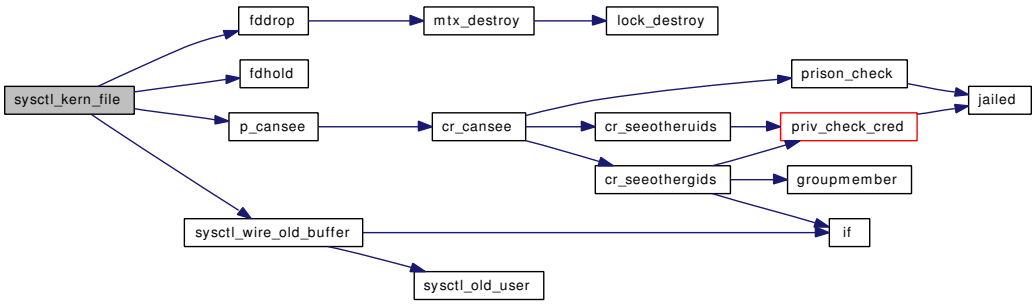
9.23.3.69 SYSCTL_INT (_kern, KERN_MAXFILESPERPROC, maxfilesperproc, CTLFLAG_RW, & *maxfilesperproc*, 0, "Maximum files allowed open per process")

9.23.3.70 static int sysctl_kern_file (SYSCTL_HANDLER_ARGS) [static]

Definition at line 2447 of file kern_descrip.c.

References allproc_lock, fddrop(), fdhold(), filehead, filelist_lock, p_cansee(), and sysctl_wire_old_-buffer().

Here is the call graph for this function:



9.23.3.71 SYSCTL_PROC (_kern, KERN_FILE, file, CTLTYPE_OPAQUE| *CTLFLAG_RD*, 0, 0, sysctl_kern_file, " *S*, xfile", "Entire file table")

9.23.4 Variable Documentation

9.23.4.1 struct fileops badfileops

Initial value:

```
 {
        .fo_read = badfo_readwrite,
        .fo_write = badfo_readwrite,
        .fo_ioctl = badfo_ioctl,
        .fo_poll = badfo_poll,
        .fo_kqfilter = badfo_kqfilter,
        .fo_stat = badfo_stat,
        .fo_close = badfo_close,
}
```

Definition at line 2691 of file kern_descrip.c.

Referenced by _fget(), falloc(), fdcheckstd(), fdrop_locked(), kern_open(), mqf_close(), pipe_close(), soo_close(), and vn_closefile().

9.23.4.2 struct mtx fdesc_mtx [static]

Definition at line 144 of file kern_descrip.c.

Referenced by fddrop(), fdfree(), fdhold(), and filelistinit().

9.23.4.3 struct cdevsw fildesc_cdevsw [static]

Initial value:

```
{
        .d_version =    D_VERSION,
        .d_flags =      D_NEEDGIANT,
        .d_open =       fdopen,
        .d_name =       "FD",
}
```

Definition at line 2732 of file kern_descrip.c.

9.23.4.4 uma_zone_t file_zone [static]

Definition at line 84 of file kern_descrip.c.

Referenced by falloc(), fdrop_locked(), and filelistinit().

9.23.4.5 struct filelist filehead

Definition at line 137 of file kern_descrip.c.

Referenced by falloc(), sysctl_kern_file(), and unp_gc().

9.23.4.6 struct sx filelist_lock

Definition at line 139 of file kern_descrip.c.

Referenced by falloc(), fdrop_locked(), filelistinit(), sysctl_kern_file(), and unp_gc().

9.23.4.7 void(*) mq_fdclose(struct thread *td, int fd, struct file *fp)

Definition at line 141 of file kern_descrip.c.

Referenced by do_dup(), fdcloseexec(), kern_close(), and mqfs_init().

9.23.4.8 int openfiles

Definition at line 138 of file kern_descrip.c.

Referenced by falloc(), fdrop_locked(), and unp_gc().

9.23.4.9 struct mtx sigio_lock

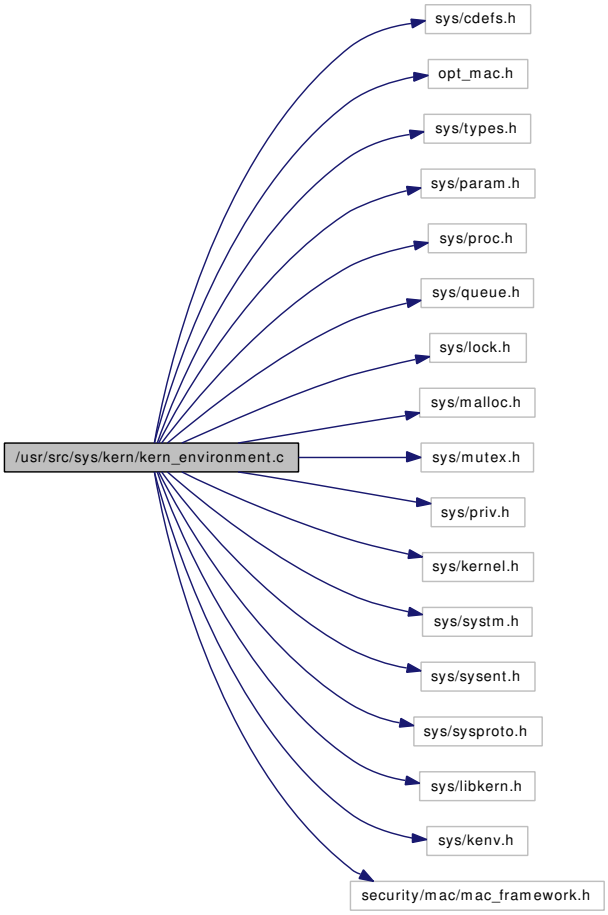
Definition at line 140 of file kern_descrip.c.

Referenced by filelistinit().

9.24 /usr/src/sys/kern/kern_environment.c File Reference

```
#include <sys/cdefs.h>
#include "opt_mac.h"
#include <sys/types.h>
#include <sys/param.h>
#include <sys/proc.h>
#include <sys/queue.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/priv.h>
#include <sys/kernel.h>
#include <sys/systm.h>
#include <sys/sysent.h>
#include <sys/sysproto.h>
#include <sys/libkern.h>
#include <sys/kenv.h>
#include <security/mac/mac_framework.h>
```

Include dependency graph for kern_environment.c:



Defines

- #define KENV_SIZE 512
- #define KENV_CHECK

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_environment.c,v 1.46 2006/11/06 13:42:00 rwatson Exp $")
- static MALLOC_DEFINE (M_KENV,"kenv","kernel environment")
- static char * kernenv_next (char *)
- int kenv (struct thread *td, struct kenv_args *uap)
- static void init_dynamic_kenv (void *data __unused)
- SYSINIT (kenv, SI_SUB_KMEM, SI_ORDER_ANY, init_dynamic_kenv, NULL)
- void freeenv (char *env)
- static char * _getenv_dynamic (const char *name, int *idx)
- static char * _getenv_static (const char *name)
- char * getenv (const char *name)
- int testenv (const char *name)
- int setenv (const char *name, const char *value)
- int unsetenv (const char *name)

- int getenv_string (const char *name, char *data, int size)
- int getenv_int (const char *name, int *data)
- long getenv_long (const char *name, long *data)
- unsigned long getenv_ulong (const char *name, unsigned long *data)
- int getenv_quad (const char *name, quad_t *data)
- void tunable_int_init (void *data)
- void tunable_long_init (void *data)
- void tunable_ulong_init (void *data)
- void tunable_str_init (void *data)

Variables

- char * kern_envp
- char ** kenvp
- mtx kenv_lock
- int dynamic_kenv = 0

9.24.1 Define Documentation

9.24.1.1 #define KENV_CHECK

Value:

```
if (!dynamic_kenv) \
                            panic("%s: called before SI_SUB_KMEM", __func__)
```

Definition at line 77 of file kern_environment.c.

Referenced by setenv(), and unsetenv().

9.24.1.2 #define KENV_SIZE 512

Definition at line 61 of file kern_environment.c.

Referenced by init_dynamic_kenv(), and setenv().

9.24.2 Function Documentation

9.24.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_environment. *c*, v 1.46 2006/11/06 13:42:00 rwatson Exp $")

9.24.2.2 static char* _getenv_dynamic (const char * *name*, int * *idx*) [static]

Definition at line 251 of file kern_environment.c.

References kenv_lock, and kenvp.

Referenced by getenv(), setenv(), testenv(), and unsetenv().

9.24.2.3 static char* _getenv_static (const char * *name*) [static]

Definition at line 270 of file kern_environment.c.

References kern_envp, and kernenv_next().

Referenced by getenv(), and testenv().

Here is the call graph for this function:

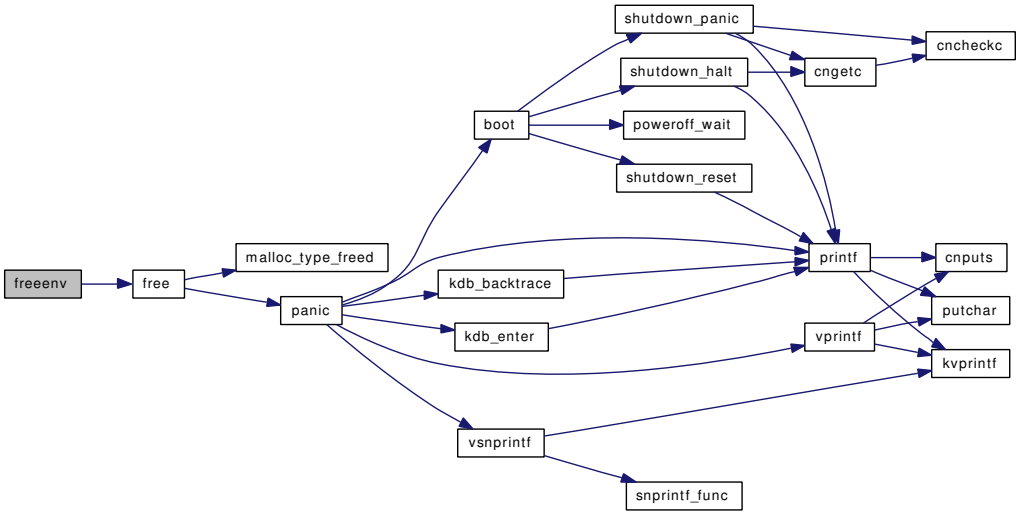
9.24.2.4 void freeenv (char * *env*)

Definition at line 240 of file kern_environment.c.

References dynamic_kenv, and free().

Referenced by getenv_quad(), getenv_string(), kenv(), start_init(), and vfs_mountroot().

Here is the call graph for this function:



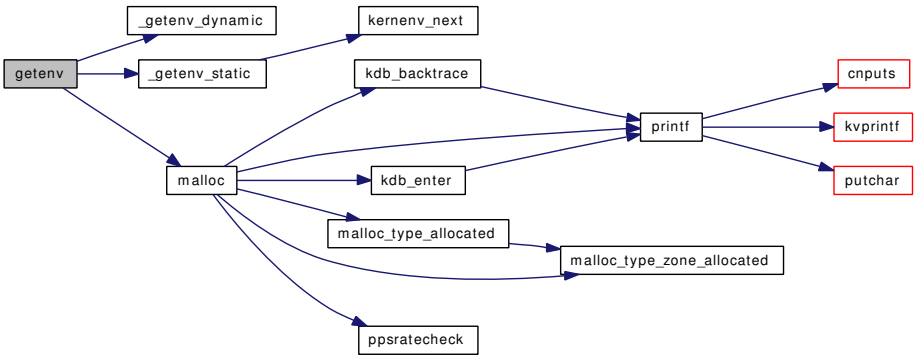
9.24.2.5 char* getenv (const char * *name*)

Definition at line 295 of file kern_environment.c.

References _getenv_dynamic(), _getenv_static(), buf, dynamic_kenv, kenv_lock, malloc(), and ret.

Referenced by getenv_quad(), getenv_string(), kenv(), start_init(), and vfs_mountroot().

Here is the call graph for this function:

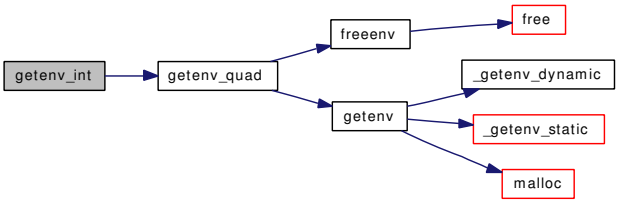


9.24.2.6 int getenv_int (const char * *name*, int * *data*)

Definition at line 431 of file kern_environment.c.

References getenv_quad().

Here is the call graph for this function:

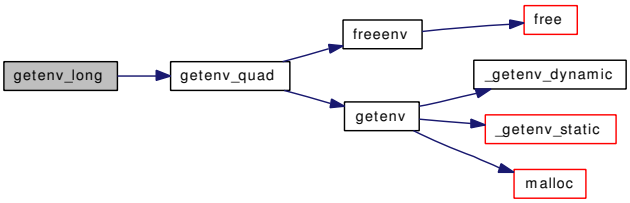


9.24.2.7 long getenv_long (const char * *name*, long * *data*)

Definition at line 446 of file kern_environment.c.

References getenv_quad().

Here is the call graph for this function:



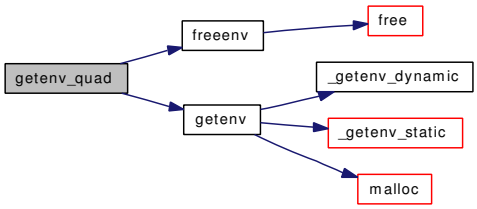
9.24.2.8 int getenv_quad (const char * *name*, quad_t * *data*)

Definition at line 476 of file kern_environment.c.

References freeenv(), and getenv().

Referenced by getenv_int(), getenv_long(), and getenv_ulong().

Here is the call graph for this function:

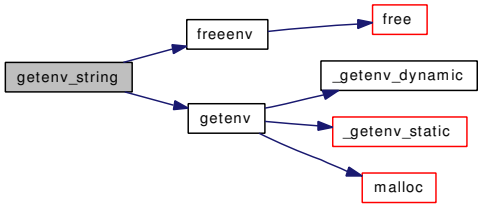


9.24.2.9 int getenv_string (const char * *name*, char * *data*, int *size*)

Definition at line 414 of file kern_environment.c.

References freeenv(), and getenv().

Here is the call graph for this function:

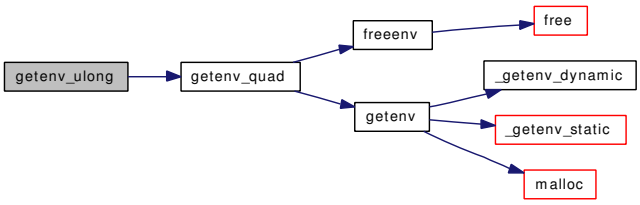


9.24.2.10 unsigned long getenv_ulong (const char * *name*, unsigned long * *data*)

Definition at line 461 of file kern_environment.c.

References getenv_quad().

Here is the call graph for this function:

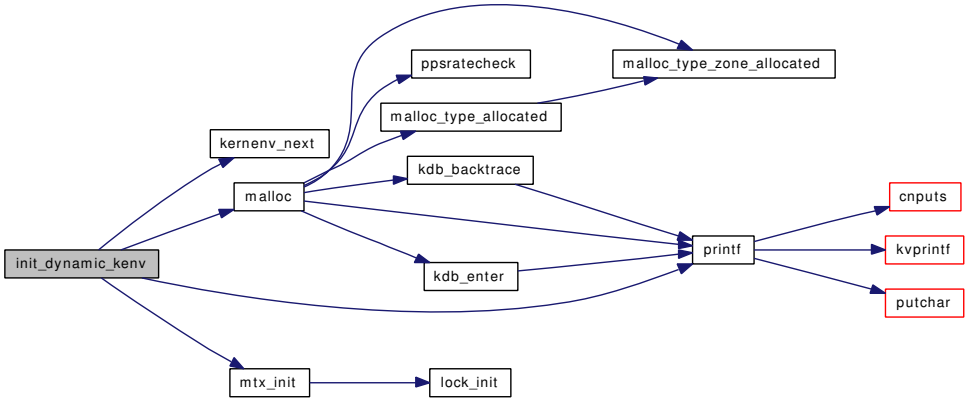


9.24.2.11 static void init_dynamic_kenv (void *data *__unused*) [static]

Definition at line 214 of file kern_environment.c.

References dynamic_kenv, kenv_lock, KENV_SIZE, kenvp, kern_envp, kernenv_next(), malloc(), mtx_init(), and printf().

Here is the call graph for this function:

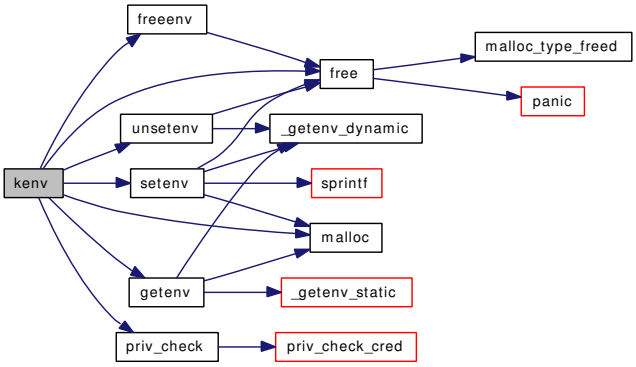


9.24.2.12 int kenv (struct thread * *td*, struct kenv_args * *uap*)

Definition at line 81 of file kern_environment.c.

References dynamic_kenv, free(), freeenv(), getenv(), kenv_lock, kenvp, malloc(), priv_check(), setenv(), and unsetenv().

Here is the call graph for this function:



9.24.2.13 static char * kernenv_next (char *) [static]

Definition at line 515 of file kern_environment.c.

Referenced by _getenv_static(), and init_dynamic_kenv().

9.24.2.14 static MALLOC_DEFINE (M_KENV, "kenv", "kernel environment") [static]

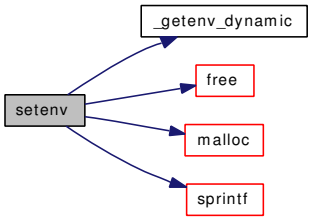
9.24.2.15 int setenv (const char * *name*, const char * *value*)

Definition at line 342 of file kern_environment.c.

References _getenv_dynamic(), buf, free(), KENV_CHECK, kenv_lock, KENV_SIZE, kenvp, malloc(), and sprintf().

Referenced by kenv().

Here is the call graph for this function:



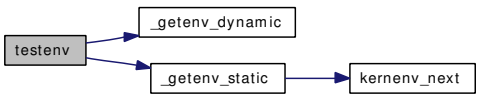
9.24.2.16 SYSINIT (kenv, SI_SUB_KMEM, SI_ORDER_ANY, init_dynamic_kenv, NULL)

9.24.2.17 int testenv (const char * *name*)

Definition at line 323 of file kern_environment.c.

References _getenv_dynamic(), _getenv_static(), dynamic_kenv, and kenv_lock.

Here is the call graph for this function:



9.24.2.18 void tunable_int_init (void * *data*)

Definition at line 529 of file kern_environment.c.

9.24.2.19 void tunable_long_init (void * *data*)

Definition at line 537 of file kern_environment.c.

9.24.2.20 void tunable_str_init (void * *data*)

Definition at line 553 of file kern_environment.c.

9.24.2.21 void tunable_ulong_init (void * *data*)

Definition at line 545 of file kern_environment.c.

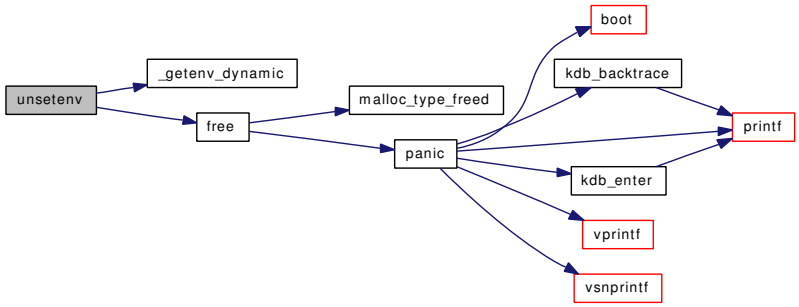
9.24.2.22 int unsetenv (const char * *name*)

Definition at line 388 of file kern_environment.c.

References _getenv_dynamic(), free(), KENV_CHECK, kenv_lock, and kenvp.

Referenced by kenv().

Here is the call graph for this function:



9.24.3 Variable Documentation

9.24.3.1 int dynamic_kenv = 0

Definition at line 75 of file kern_environment.c.

Referenced by freeenv(), getenv(), init_dynamic_kenv(), kenv(), res_find(), and testenv().

9.24.3.2 struct mtx kenv_lock

Definition at line 69 of file kern_environment.c.

Referenced by _getenv_dynamic(), getenv(), init_dynamic_kenv(), kenv(), res_find(), setenv(), testenv(), and unsetenv().

9.24.3.3 char** kenvp

Definition at line 68 of file kern_environment.c.

Referenced by _getenv_dynamic(), init_dynamic_kenv(), kenv(), res_find(), setenv(), and unsetenv().

9.24.3.4 char* kern_envp

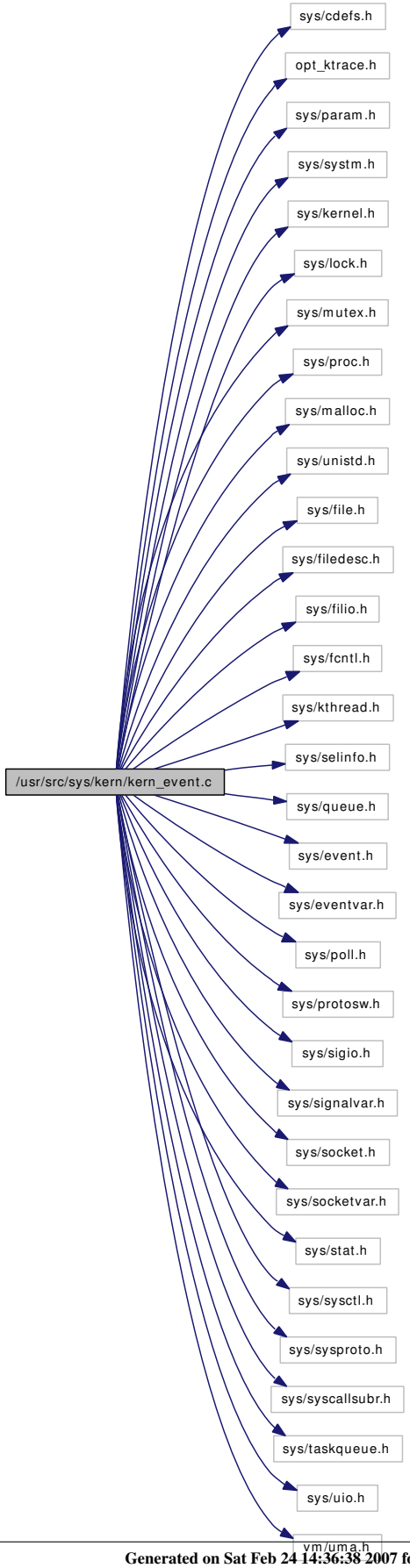
Definition at line 64 of file kern_environment.c.

Referenced by _getenv_static(), init_dynamic_kenv(), and res_find().

9.25 /usr/src/sys/kern/kern_event.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ktrace.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/malloc.h>
#include <sys/unistd.h>
#include <sys/file.h>
#include <sys/filedesc.h>
#include <sys/filio.h>
#include <sys/fcntl.h>
#include <sys/kthread.h>
#include <sys/selinfo.h>
#include <sys/queue.h>
#include <sys/event.h>
#include <sys/eventvar.h>
#include <sys/poll.h>
#include <sys/protosw.h>
#include <sys/sigio.h>
#include <sys/signalvar.h>
#include <sys/socket.h>
#include <sys/socketvar.h>
#include <sys/stat.h>
#include <sys/sysctl.h>
#include <sys/sysproto.h>
#include <sys/syscallsubr.h>
#include <sys/taskqueue.h>
#include <sys/uio.h>
#include <vm/uma.h>
```

Include dependency graph for kern_event.c:


/usr/src/sys/kern/kern_event.c
sys/cdefs.h
opt_ktrace.h
sys/param.h
sys/systm.h
sys/kernel.h
sys/lock.h
sys/mutex.h
sys/proc.h
sys/malloc.h
sys/unistd.h
sys/file.h
sys/filedesc.h
sys/filio.h
sys/fcntl.h
sys/kthread.h
sys/selinfo.h
sys/queue.h
sys/event.h
sys/eventvar.h
sys/poll.h
sys/protosw.h
sys/sigio.h
sys/signalvar.h
sys/socket.h
sys/socketvar.h
sys/stat.h
sys/sysctl.h
sys/sysproto.h
sys/syscallsubr.h
sys/taskqueue.h
sys/uio.h
vm/uma.h

Data Structures

- struct kevent_args

Defines

- #define KQ_GLOBAL_LOCK(lck, haslck)
- #define KQ_GLOBAL_UNLOCK(lck, haslck)
- #define KNOTE_ACTIVATE(kn, islock)
- #define KQ_LOCK(kq)
- #define KQ_FLUX_WAKEUP(kq)
- #define KQ_UNLOCK_FLUX(kq)
- #define KQ_UNLOCK(kq)
- #define KQ_OWNED(kq)
- #define KQ_NOTOWNED(kq)
- #define KN_LIST_LOCK(kn)
- #define KN_LIST_UNLOCK(kn)
- #define KNL_ASSERT_LOCK(knl, islocked)
- #define KNL_ASSERT_LOCKED(knl) do {} while(0)
- #define KNL_ASSERT_UNLOCKED(knl) do {} while (0)
- #define KN_HASHSIZE 64
- #define KN_HASH(val, mask) (((val) ^ (val >> 8)) & (mask))

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_event.c,v 1.107 2006/11/20 22:17:50 jhb Exp $")
- static MALLOC_DEFINE (M_KQUEUE,"kqueue","memory for kqueue system")
- MTX_SYSINIT (kq_global,&kq_global,"kqueue order", MTX_DEF)
- TASKQUEUE_DEFINE_THREAD (kqueue)
- static int kevent_copyout (void *arg, struct kevent *kevp, int count)
- static int kevent_copyin (void *arg, struct kevent *kevp, int count)
- static int kqueue_register (struct kqueue *kq, struct kevent *kev, struct thread *td, int waitok)
- static int kqueue_aquire (struct file *fp, struct kqueue **kqp)
- static void kqueue_release (struct kqueue *kq, int locked)
- static int kqueue_expand (struct kqueue *kq, struct filterops *fops, uintptr_t ident, int waitok)
- static void kqueue_task (void *arg, int pending)
- static int kqueue_scan (struct kqueue *kq, int maxevents, struct kevent_copyops *k_ops, const struct timespec *timeout, struct kevent *keva, struct thread *td)
- static void kqueue_wakeup (struct kqueue *kq)
- static struct filterops * kqueue_fo_find (int filt)
- static void kqueue_fo_release (int filt)
- static int knote_attach (struct knote *kn, struct kqueue *kq)
- static void knote_drop (struct knote *kn, struct thread *td)
- static void knote_enqueue (struct knote *kn)
- static void knote_dequeue (struct knote *kn)
- static void knote_init (void)
- static struct knote * knote_alloc (int waitok)
- static void knote_free (struct knote *kn)
- static void filt_kqdetach (struct knote *kn)
- static int filt_kqueue (struct knote *kn, long hint)

- static int filt_procattach (struct knote *kn)
- static void filt_procdetach (struct knote *kn)
- static int filt_proc (struct knote *kn, long hint)
- static int filt_fileattach (struct knote *kn)
- static void filt_timerexpire (void *knx)
- static int filt_timerattach (struct knote *kn)
- static void filt_timerdetach (struct knote *kn)
- static int filt_timer (struct knote *kn, long hint)
- SYSCTL_INT (_kern, OID_AUTO, kq_calloutmax, CTLFLAG_RW,&kq_calloutmax, 0,"Maximum number of callouts allocated for kqueue")
- static int filt_nullattach (struct knote *kn)
- MTX_SYSINIT (kqueue_filterops,&filterops_lock,"protect sysfilt_ops", MTX_DEF)
- static int kqueue_kqfilter (struct file *fp, struct knote *kn)
- static int timertoticks (intptr_t data)
- int kqueue (struct thread *td, struct kqueue_args *uap)
- int kevent (struct thread *td, struct kevent_args *uap)
- int kern_kevent (struct thread *td, int fd, int nchanges, int nevents, struct kevent_copyops *k_ops, const struct timespec *timeout)
- int kqueue_add_filteropts (int filt, struct filterops *filtops)
- int kqueue_del_filteropts (int filt)
- static void kqueue_schedtask (struct kqueue *kq)
- static int kqueue_read (struct file *fp, struct uio *uio, struct ucred *active_cred, int flags, struct thread *td)
- static int kqueue_write (struct file *fp, struct uio *uio, struct ucred *active_cred, int flags, struct thread *td)
- static int kqueue_ioctl (struct file *fp, u_long cmd, void *data, struct ucred *active_cred, struct thread *td)
- static int kqueue_poll (struct file *fp, int events, struct ucred *active_cred, struct thread *td)
- static int kqueue_stat (struct file *fp, struct stat *st, struct ucred *active_cred, struct thread *td)
- static int kqueue_close (struct file *fp, struct thread *td)
- void knote (struct knlist *list, long hint, int islocked)
- void knlist_add (struct knlist *knl, struct knote *kn, int islocked)
- static void knlist_remove_kq (struct knlist *knl, struct knote *kn, int knlislocked, int kqislocked)
- void knlist_remove (struct knlist *knl, struct knote *kn, int islocked)
- void knlist_remove_inevent (struct knlist *knl, struct knote *kn)
- int knlist_empty (struct knlist *knl)
- MTX_SYSINIT (knlist_lock,&knlist_lock,"knlist lock for lockless objects", MTX_DEF)
- static void knlist_mtx_lock (void *arg)
- static void knlist_mtx_unlock (void *arg)
- static int knlist_mtx_locked (void *arg)
- void knlist_init (struct knlist *knl, void *lock, void(*kl_lock)(void *), void(*kl_unlock)(void *), int(*kl_locked)(void *))
- void knlist_destroy (struct knlist *knl)
- void knlist_cleardel (struct knlist *knl, struct thread *td, int islocked, int killkn)
- void knote_fdclose (struct thread *td, int fd)
- int kqfd_register (int fd, struct kevent *kev, struct thread *td, int waitok)

Variables

- static struct mtx kq_global
- static fo_rdwr_t kqueue_read
- static fo_rdwr_t kqueue_write
- static fo_ioctl_t kqueue_ioctl
- static fo_poll_t kqueue_poll
- static fo_kqfilter_t kqueue_kqfilter
- static fo_stat_t kqueue_stat
- static fo_close_t kqueue_close
- static struct fileops kqueueops
- static struct filterops file_filtops
- static struct filterops kqread_filtops
- static struct filterops proc_filtops
- static struct filterops timer_filtops
- static uma_zone_t knote_zone
- static int kq_ncallouts = 0
- static int kq_calloutmax = (4 ∗ 1024)
- filterops null_filtops
- filterops sig_filtops
- filterops fs_filtops
- static struct mtx filterops_lock
- struct {
  filterops ∗ for_fop
  int for_refcnt
  } sysfilt_ops [EVFILT_SYSCOUNT]

- static struct mtx knlist_lock

9.25.1 Define Documentation

9.25.1.1 #define KN_HASH(val, mask) (((val) ^ (val >> 8)) & (mask))

Definition at line 222 of file kern_event.c.

Referenced by knote_attach(), knote_drop(), and kqueue_register().

9.25.1.2 #define KN_HASHSIZE 64

Definition at line 221 of file kern_event.c.

Referenced by kqueue_expand().

9.25.1.3 #define KN_LIST_LOCK(kn)

Value:

```
do {                                                    \
        if (kn->kn_knlist != NULL)                                      \
                kn->kn_knlist->kl_lock(kn->kn_knlist->kl_lockarg);      \
} while (0)
```

Definition at line 193 of file kern_event.c.

Referenced by kqueue_register(), and kqueue_scan().

9.25.1.4 #define KN_LIST_UNLOCK(kn)

Value:

```
do {                                                    \
        if (kn->kn_knlist != NULL)                                      \
                kn->kn_knlist->kl_unlock(kn->kn_knlist->kl_lockarg);    \
} while (0)
```

Definition at line 197 of file kern_event.c.

Referenced by kqueue_register(), and kqueue_scan().

9.25.1.5 #define KNL_ASSERT_LOCK(knl, islocked)

Value:

```
do {                                \
        if (islocked)                                                   \
                KNL_ASSERT_LOCKED(knl);                         \
        else                                                            \
                KNL_ASSERT_UNLOCKED(knl);                               \
} while (0)
```

Definition at line 201 of file kern_event.c.

Referenced by knlist_add(), knlist_remove_kq(), and knote().

9.25.1.6 #define KNL_ASSERT_LOCKED(knl) do {} while(0)

Definition at line 217 of file kern_event.c.

Referenced by knlist_cleardel(), and knlist_empty().

9.25.1.7 #define KNL_ASSERT_UNLOCKED(knl) do {} while (0)

Definition at line 218 of file kern_event.c.

Referenced by knlist_cleardel().

9.25.1.8 #define KNOTE_ACTIVATE(kn, islock)

Value:

```
do {                                \
        if ((islock))                                                   \
                mtx_assert(&(kn)->kn_kq->kq_lock, MA_OWNED);            \
        else                                                            \
                KQ_LOCK((kn)->kn_kq);                                   \
        (kn)->kn_status |= KN_ACTIVE;                                   \
        if (((kn)->kn_status & (KN_QUEUED | KN_DISABLED)) == 0)         \
```

```
                knote_enqueue((kn));                                    \
        if (!(islock))                                                  \
                KQ_UNLOCK((kn)->kn_kq);                                 \
} while(0)
```

Definition at line 160 of file kern_event.c.

Referenced by filt_procattach(), filt_timerexpire(), knote(), and kqueue_register().

9.25.1.9 #define KQ_FLUX_WAKEUP(kq)

Value:

```
do {                                                    \
        if (((kq)->kq_state & KQ_FLUXWAIT) == KQ_FLUXWAIT) {            \
                (kq)->kq_state &= ~KQ_FLUXWAIT;                         \
                wakeup((kq));                                           \
        }                                                               \
} while (0)
```

Definition at line 174 of file kern_event.c.

Referenced by kqueue_scan().

9.25.1.10 #define KQ_GLOBAL_LOCK(lck, haslck)

Value:

```
do {    \
        if (!haslck)                            \
                mtx_lock(lck);                  \
        haslck = 1;                             \
} while (0)
```

Definition at line 76 of file kern_event.c.

Referenced by kqueue_register(), kqueue_scan(), and kqueue_task().

9.25.1.11 #define KQ_GLOBAL_UNLOCK(lck, haslck)

Value:

```
do {    \
        if (haslck)                             \
                mtx_unlock(lck);                        \
        haslck = 0;                             \
} while (0)
```

Definition at line 81 of file kern_event.c.

Referenced by kqueue_register(), kqueue_scan(), and kqueue_task().

9.25.1.12 #define KQ_LOCK(kq)

Value:

```
do {                                                \
        mtx_lock(&(kq)->kq_lock);                                       \
} while (0)
```

Definition at line 171 of file kern_event.c.

Referenced by knlist_add(), knlist_cleardel(), knlist_remove_kq(), knote(), knote_drop(), knote_fdclose(), kqueue_aquire(), kqueue_close(), kqueue_expand(), kqueue_poll(), kqueue_register(), kqueue_release(), kqueue_scan(), and kqueue_task().

9.25.1.13 #define KQ_NOTOWNED(kq)

Value:

```
do {                                                \
        mtx_assert(&(kq)->kq_lock, MA_NOTOWNED);                        \
} while (0)
```

Definition at line 190 of file kern_event.c.

Referenced by knlist_add(), knote_drop(), kqueue_expand(), and kqueue_scan().

9.25.1.14 #define KQ_OWNED(kq)

Value:

```
do {                                                \
        mtx_assert(&(kq)->kq_lock, MA_OWNED);                           \
} while (0)
```

Definition at line 187 of file kern_event.c.

Referenced by knote_attach(), knote_dequeue(), knote_enqueue(), kqueue_release(), kqueue_scan(), kqueue_schedtask(), and kqueue_wakeup().

9.25.1.15 #define KQ_UNLOCK(kq)

Value:

```
do {                                                \
        mtx_unlock(&(kq)->kq_lock);                                     \
} while (0)
```

Definition at line 184 of file kern_event.c.

Referenced by knlist_add(), knlist_cleardel(), knlist_remove_kq(), knote(), knote_fdclose(), kqueue_aquire(), kqueue_close(), kqueue_expand(), kqueue_poll(), kqueue_register(), kqueue_release(), kqueue_scan(), and kqueue_task().

9.25.1.16 #define KQ_UNLOCK_FLUX(kq)

Value:

```
do {                                                    \
        KQ_FLUX_WAKEUP(kq);                                                     \
        mtx_unlock(&(kq)->kq_lock);                                             \
} while (0)
```

Definition at line 180 of file kern_event.c.

Referenced by knote_drop(), knote_fdclose(), kqueue_register(), and kqueue_scan().

9.25.2 Function Documentation

9.25.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_event. *c*, v 1.107 2006/11/20 22:17:50 jhb Exp $")

9.25.2.2 static int filt_fileattach (struct knote * *kn*) [static]

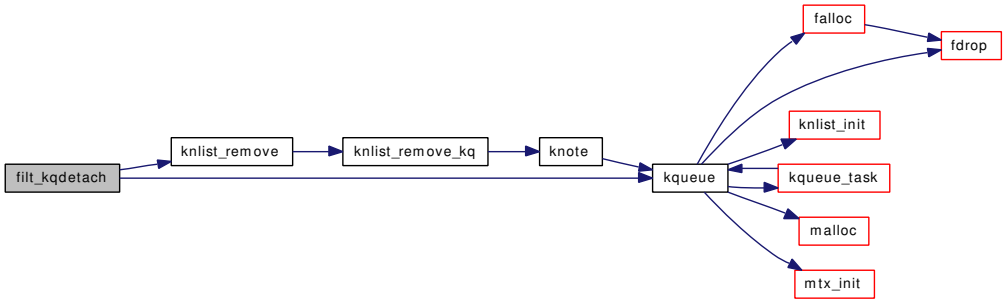
Definition at line 265 of file kern_event.c.

9.25.2.3 static void filt_kqdetach (struct knote * *kn*) [static]

Definition at line 288 of file kern_event.c.

References knlist_remove(), and kqueue().

Here is the call graph for this function:

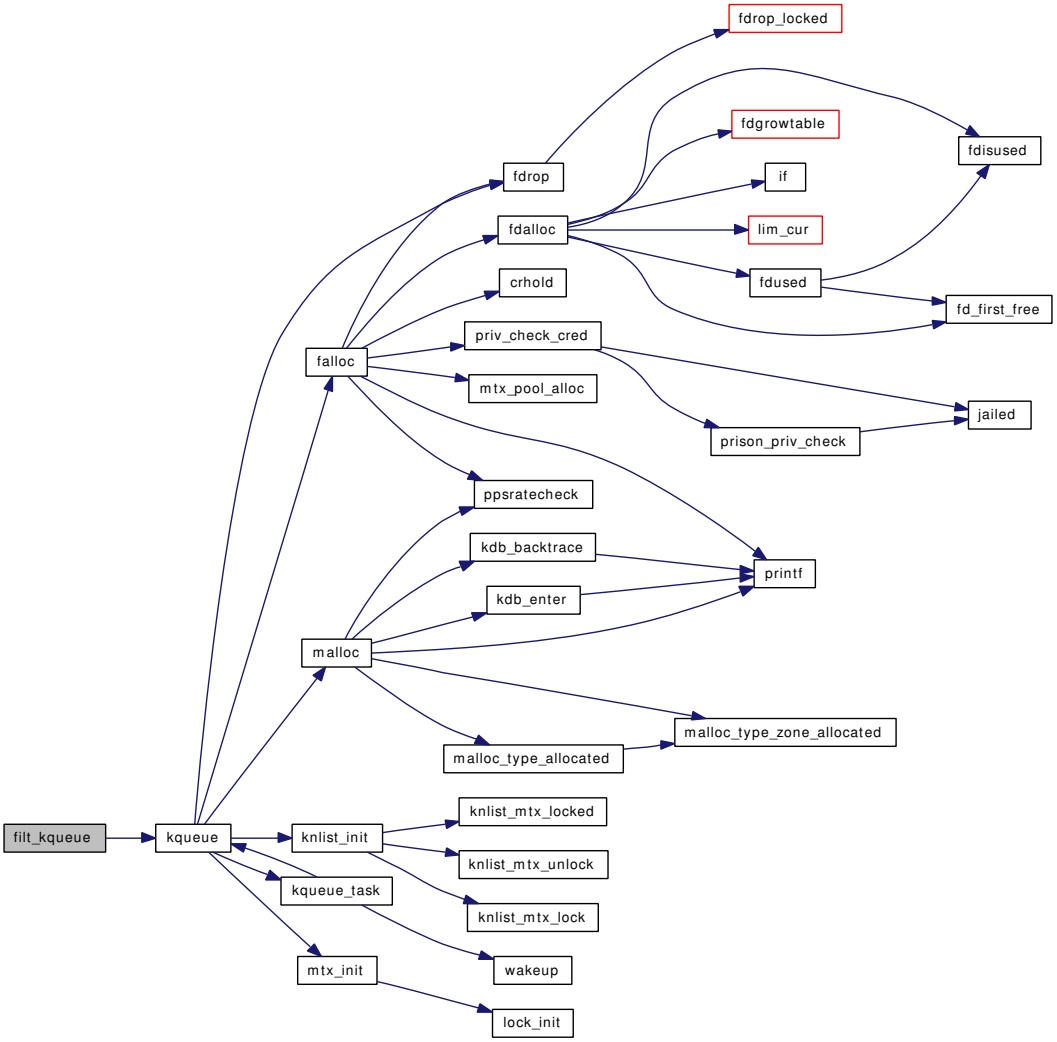


9.25.2.4 static int filt_kqueue (struct knote * *kn*, long *hint*) [static]

Definition at line 297 of file kern_event.c.

References kqueue().

Here is the call graph for this function:



9.25.2.5 static int filt_nullattach (struct knote * *kn*) [static]

Definition at line 225 of file kern_event.c.

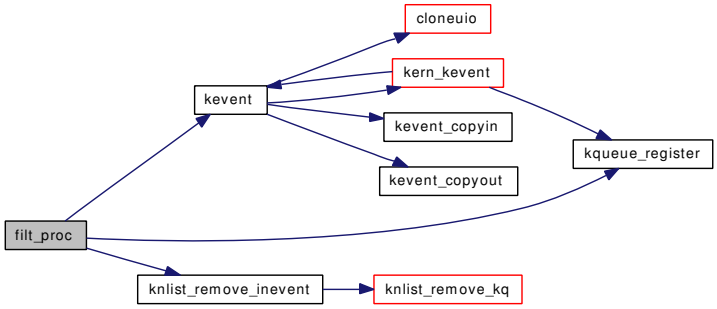
9.25.2.6 static int filt_proc (struct knote * *kn*, long *hint*) [static]

Definition at line 376 of file kern_event.c.

References kevent(), knlist_remove_inevent(), and kqueue_register().

Referenced by filt_procattach().

Here is the call graph for this function:

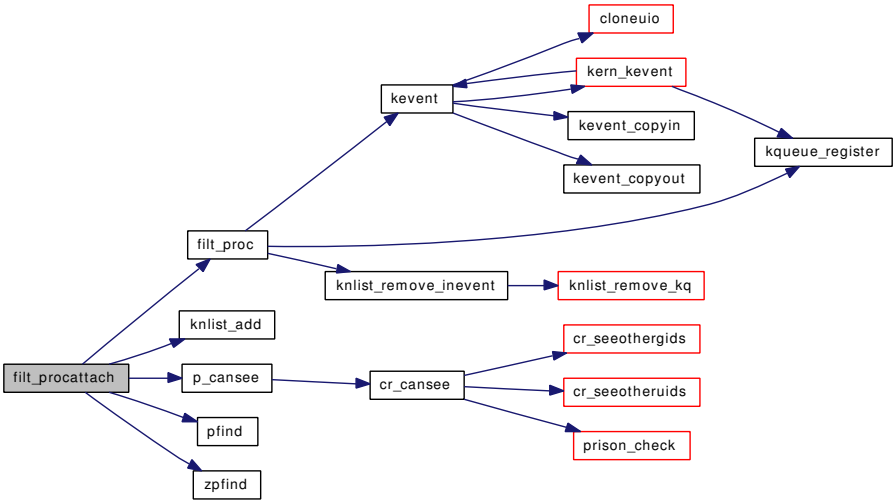


9.25.2.7 static int filt_procattach (struct knote * *kn*) [static]

Definition at line 307 of file kern_event.c.

References filt_proc(), knlist_add(), KNOTE_ACTIVATE, p_cansee(), pfind(), and zpfind().

Here is the call graph for this function:

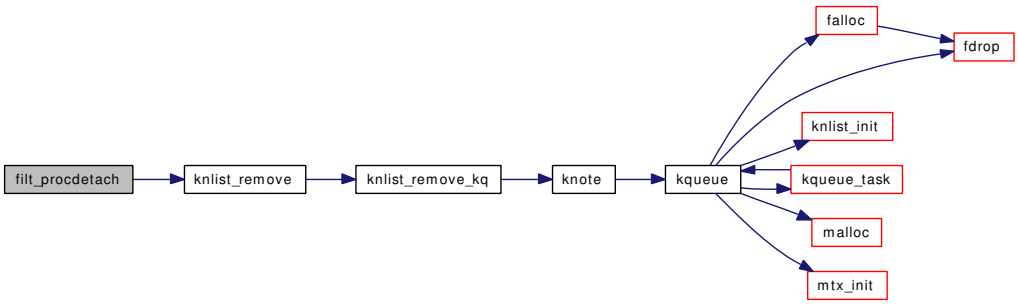


9.25.2.8 static void filt_procdetach (struct knote * *kn*) [static]

Definition at line 365 of file kern_event.c.

References knlist_remove().

Here is the call graph for this function:



9.25.2.9 static int filt_timer (struct knote * *kn*, long *hint*) [static]

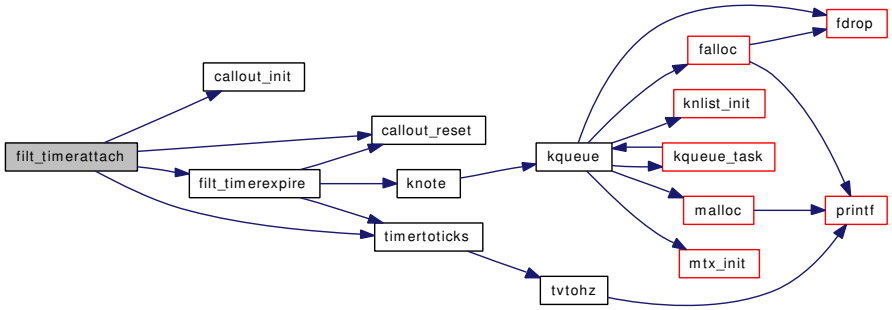
Definition at line 503 of file kern_event.c.

9.25.2.10 static int filt_timerattach (struct knote * *kn*) [static]

Definition at line 465 of file kern_event.c.

References callout, callout_init(), callout_reset(), filt_timerexpire(), kq_calloutmax, kq_ncallouts, and timertoticks().

Here is the call graph for this function:



9.25.2.11 static void filt_timerdetach (struct knote * *kn*) [static]

Definition at line 490 of file kern_event.c.

References callout, and kq_ncallouts.

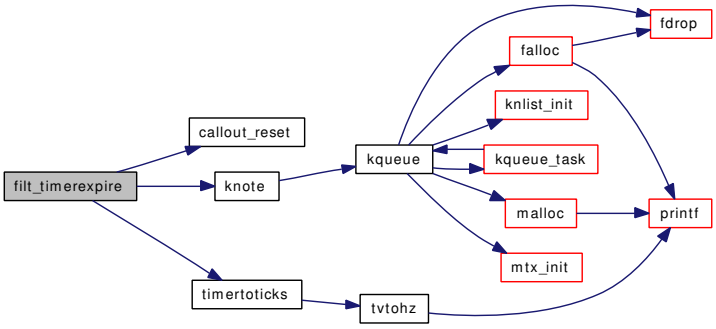
9.25.2.12 static void filt_timerexpire (void * *knx*) [static]

Definition at line 445 of file kern_event.c.

References callout, callout_reset(), knote(), KNOTE_ACTIVATE, and timertoticks().

Referenced by filt_timerattach().

Here is the call graph for this function:



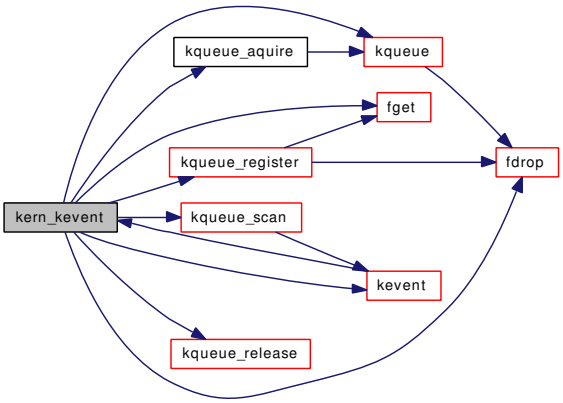
9.25.2.13 int kern_kevent (struct thread * *td*, int *fd*, int *nchanges*, int *nevents*, struct kevent_copyops * *k_ops*, const struct timespec * *timeout*)

Definition at line 652 of file kern_event.c.

References fdrop(), fget(), kevent(), kqueue(), kqueue_aquire(), kqueue_register(), kqueue_release(), and kqueue_scan().

Referenced by kevent().

Here is the call graph for this function:



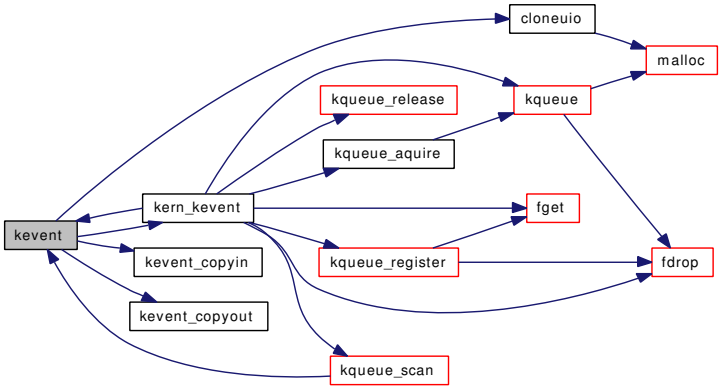
9.25.2.14 int kevent (struct thread * *td*, struct kevent_args * *uap*)

Definition at line 564 of file kern_event.c.

References kevent_args::changelist, cloneuio(), kevent_args::eventlist, kevent_args::fd, kern_kevent(), kevent_copyin(), kevent_copyout(), kevent_args::nchanges, kevent_args::nevents, and kevent_-args::timeout.

Referenced by aio_aqueue(), do_lio_listio(), filt_proc(), kern_kevent(), and kqueue_scan().

Here is the call graph for this function:



9.25.2.15 static int kevent_copyin (void * *arg*, struct kevent * *kevp*, int *count*) [static]

Definition at line 637 of file kern_event.c.

References kevent_args::changelist.

Referenced by kevent().

9.25.2.16 static int kevent_copyout (void * *arg*, struct kevent * *kevp*, int *count*) [static]

Definition at line 619 of file kern_event.c.

References kevent_args::eventlist.

Referenced by kevent().

9.25.2.17 void knlist_add (struct knlist * *knl*, struct knote * *kn*, int *islocked*)

Definition at line 1594 of file kern_event.c.

References KNL_ASSERT_LOCK, KQ_LOCK, KQ_NOTOWNED, and KQ_UNLOCK.

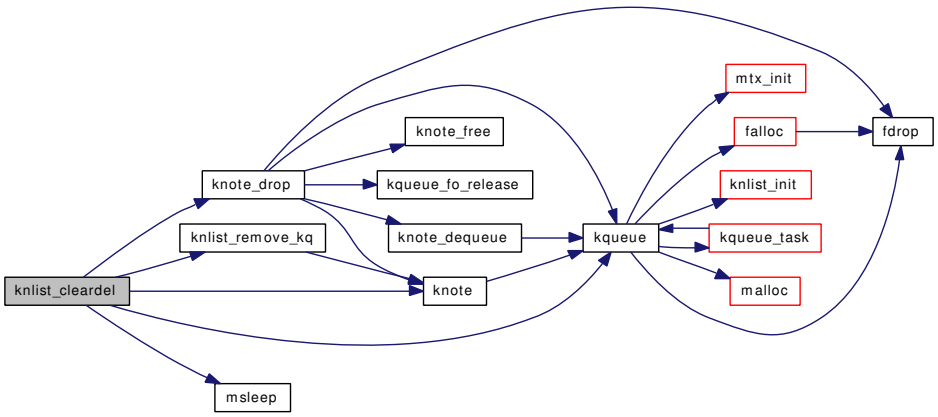
Referenced by filt_aioattach(), filt_fsattach(), filt_lioattach(), filt_procattach(), filt_sigattach(), kqueue_kqfilter(), mqf_kqfilter(), pipe_kqfilter(), soo_kqfilter(), ttykqfilter(), and vfs_kqfilter().

9.25.2.18 void knlist_cleardel (struct knlist * *knl*, struct thread * *td*, int *islocked*, int *killkn*)

Definition at line 1734 of file kern_event.c.

References KNL_ASSERT_LOCKED, KNL_ASSERT_UNLOCKED, knlist_remove_kq(), knote(), knote_drop(), KQ_LOCK, KQ_UNLOCK, kqueue(), and msleep().

Here is the call graph for this function:



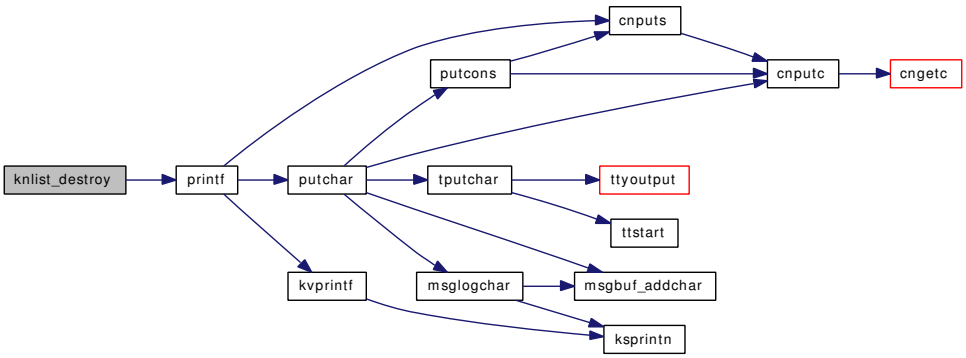
9.25.2.19 void knlist_destroy (struct knlist * *knl*)

Definition at line 1713 of file kern_event.c.

References printf().

Referenced by exit1(), mqueue_free(), pipeclose(), sofree(), ttyrel(), and vdestroy().

Here is the call graph for this function:



9.25.2.20 int knlist_empty (struct knlist * *knl*)

Definition at line 1655 of file kern_event.c.

References KNL_ASSERT_LOCKED.

Referenced by filt_aiodetach(), filt_liodetach(), filt_sordetach(), filt_sowdetach(), kqueue_close(), and kqueue_wakeup().

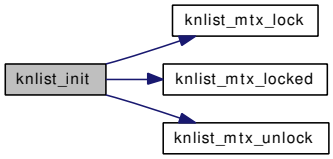
9.25.2.21 void knlist_init (struct knlist * *knl*, void * *lock*, void(*)(void *) *kl_lock*, void(*)(void *) *kl_unlock*, int(*)(void *) *kl_locked*)

Definition at line 1687 of file kern_event.c.

References knlist_lock, knlist_mtx_lock(), knlist_mtx_locked(), and knlist_mtx_unlock().

Referenced by aio_aqueue(), do_lio_listio(), fork1(), kqueue(), mqueue_alloc(), pipe(), proc0_init(), socreate(), sonewconn(), ttyalloc(), v_addpollinfo(), and vfs_event_init().

Here is the call graph for this function:



9.25.2.22 static void knlist_mtx_lock (void * *arg*) [static]

Definition at line 1669 of file kern_event.c.

Referenced by knlist_init().

9.25.2.23 static int knlist_mtx_locked (void * *arg*) [static]

Definition at line 1681 of file kern_event.c.

Referenced by knlist_init().

9.25.2.24 static void knlist_mtx_unlock (void * *arg*) [static]

Definition at line 1675 of file kern_event.c.

Referenced by knlist_init().

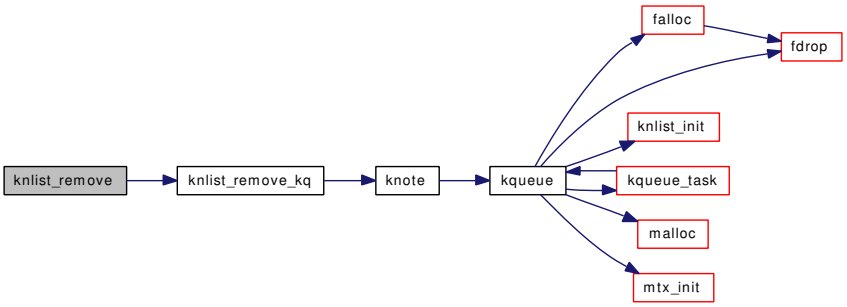
9.25.2.25 void knlist_remove (struct knlist * *knl*, struct knote * *kn*, int *islocked*)

Definition at line 1637 of file kern_event.c.

References knlist_remove_kq().

Referenced by filt_aiodetach(), filt_fsdetach(), filt_kqdetach(), filt_liodetach(), filt_mqdetach(), filt_pipedetach(), filt_procdetach(), filt_sigdetach(), filt_sordetach(), filt_sowdetach(), filt_ttyrdetach(), filt_ttywdetach(), and filt_vfsdetach().

Here is the call graph for this function:



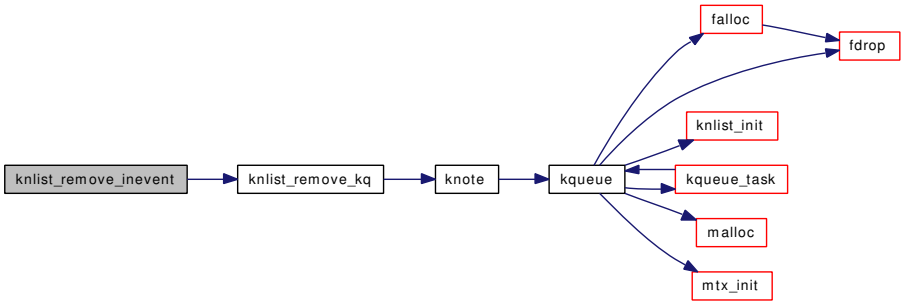
9.25.2.26 void knlist_remove_inevent (struct knlist * *knl*, struct knote * *kn*)

Definition at line 1647 of file kern_event.c.

References knlist_remove_kq().

Referenced by filt_proc().

Here is the call graph for this function:



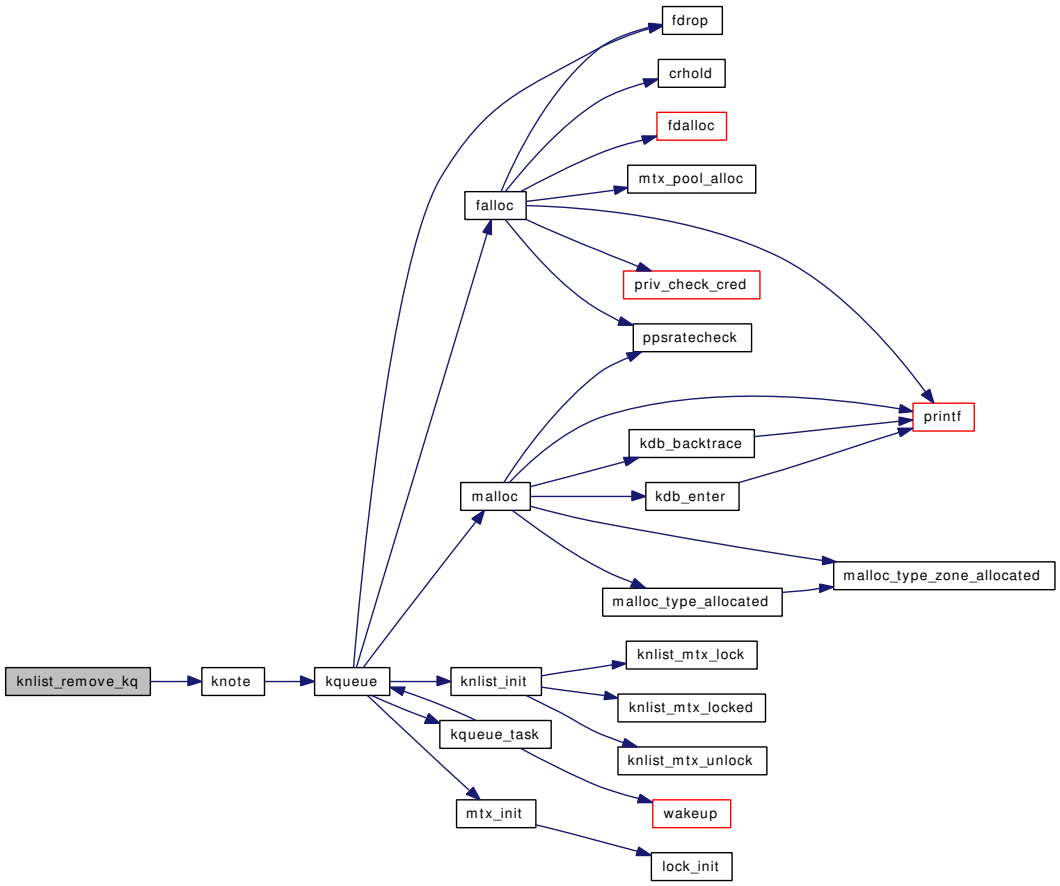
9.25.2.27 static void knlist_remove_kq (struct knlist * *knl*, struct knote * *kn*, int *knlislocked*, int *kqislocked*) [static]

Definition at line 1612 of file kern_event.c.

References KNL_ASSERT_LOCK, knote(), KQ_LOCK, and KQ_UNLOCK.

Referenced by knlist_cleardel(), knlist_remove(), and knlist_remove_inevent().

Here is the call graph for this function:



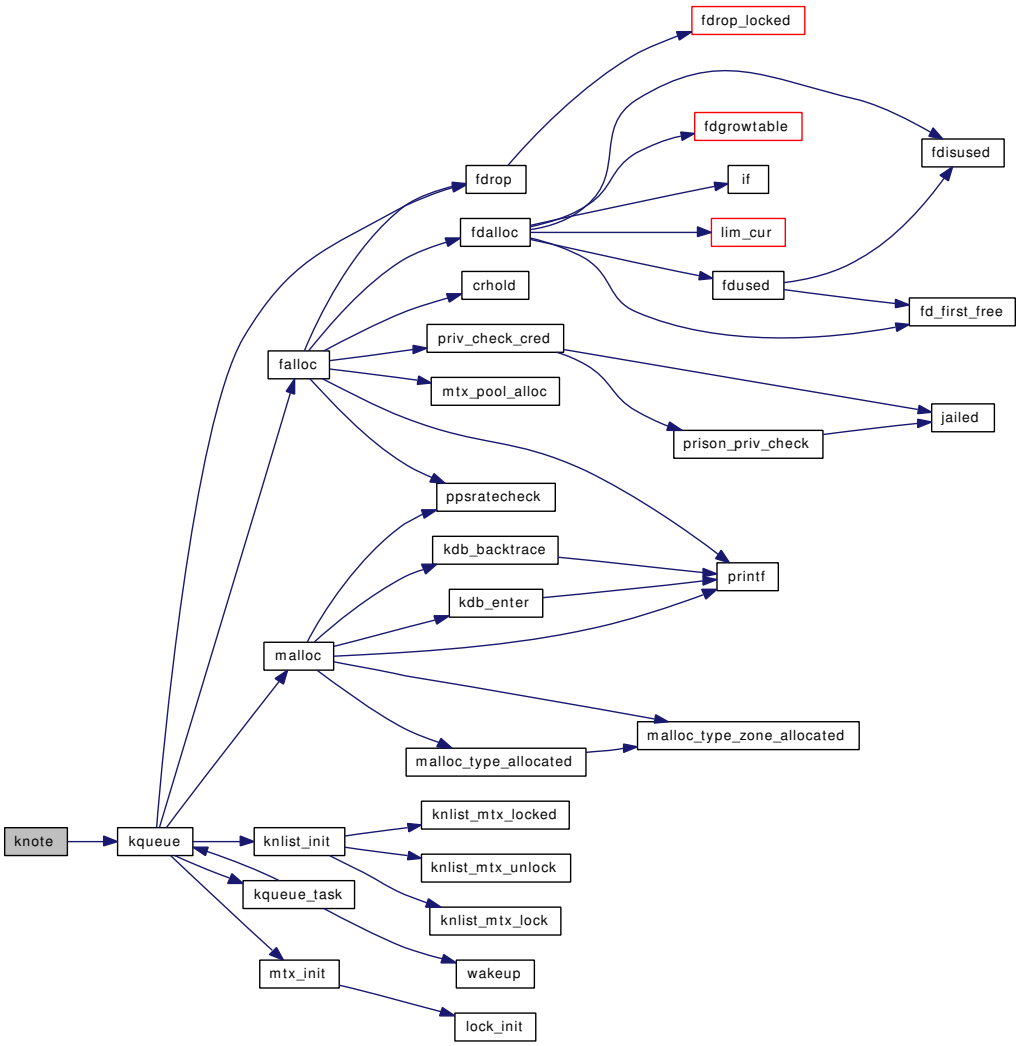
9.25.2.28 void knote (struct knlist * *list*, long *hint*, int *islocked*)

Definition at line 1551 of file kern_event.c.

References KNL_ASSERT_LOCK, KNOTE_ACTIVATE, KQ_LOCK, KQ_UNLOCK, and kqueue().

Referenced by filt_timerexpire(), knlist_cleardel(), knlist_remove_kq(), knote_alloc(), knote_drop(), knote_fdclose(), knote_init(), kqueue_close(), kqueue_register(), kqueue_scan(), and vfs_kqfilter().

Here is the call graph for this function:



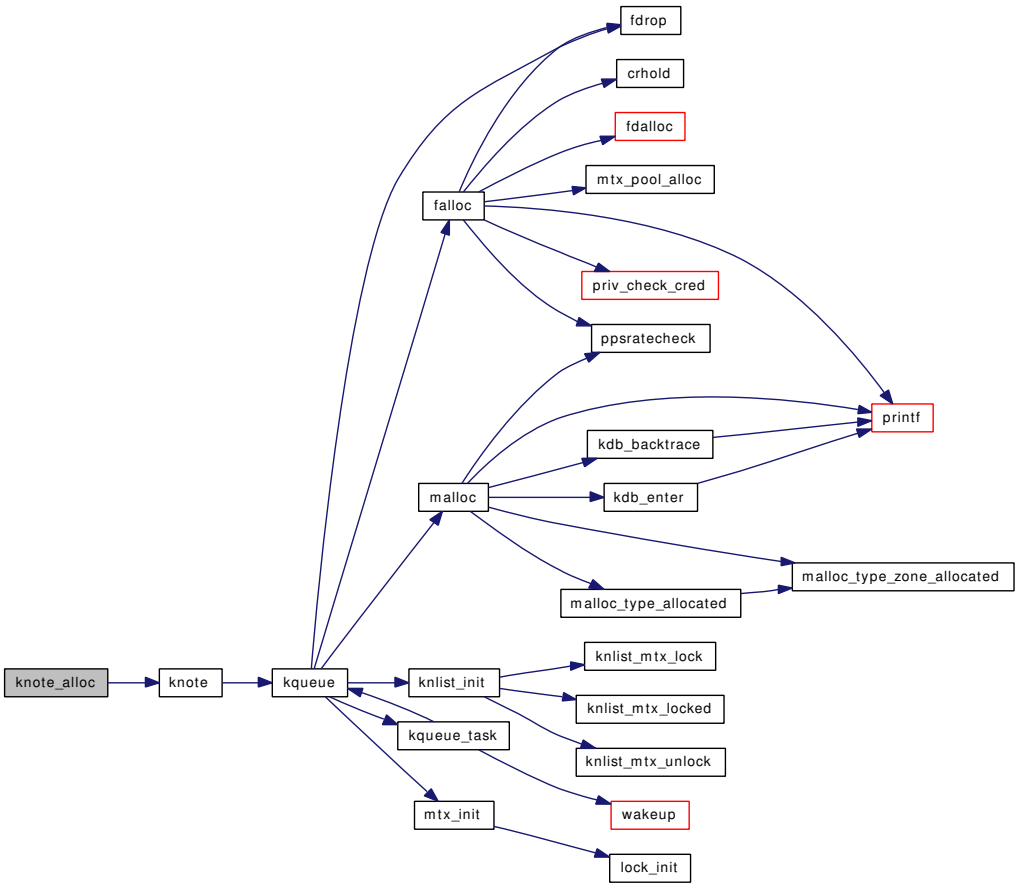
9.25.2.29 static struct knote * knote_alloc (int *waitok*) [static]

Definition at line 1933 of file kern_event.c.

References knote(), and knote_zone.

Referenced by kqueue_register(), and kqueue_scan().

Here is the call graph for this function:



9.25.2.30 static int knote_attach (struct knote * *kn*, struct kqueue * *kq*) [static]

Definition at line 1836 of file kern_event.c.

References KN_HASH, and KQ_OWNED.

Referenced by kqueue_register().

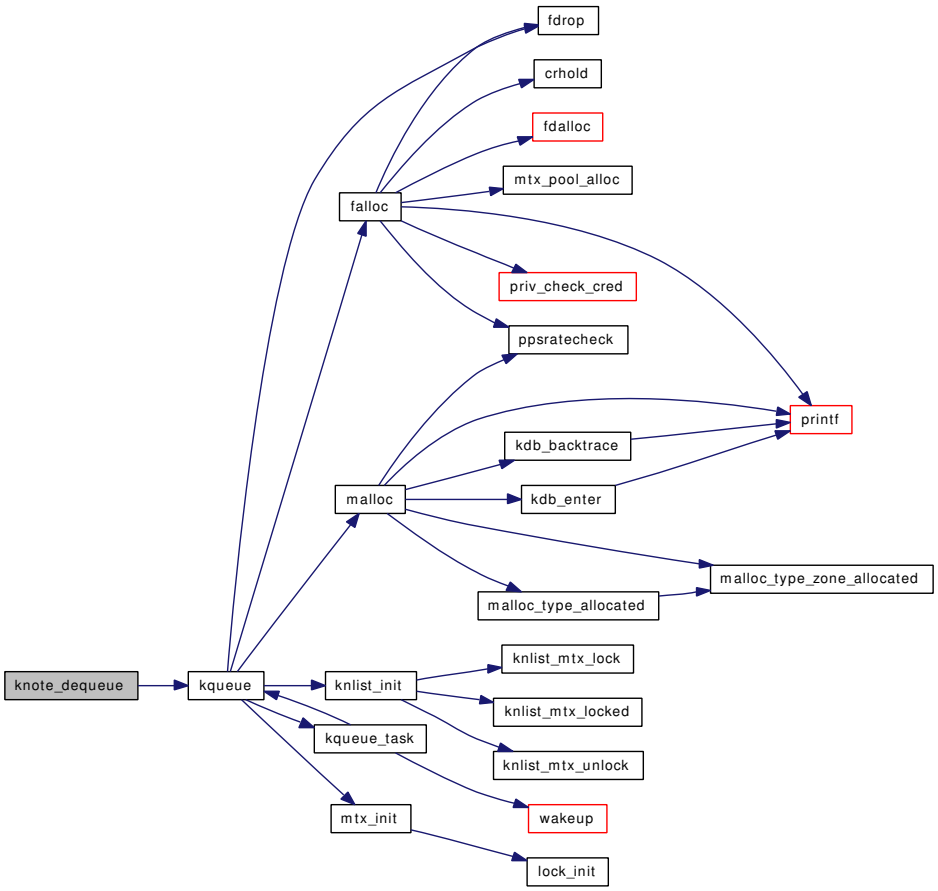
9.25.2.31 static void knote_dequeue (struct knote * *kn*) [static]

Definition at line 1911 of file kern_event.c.

References KQ_OWNED, and kqueue().

Referenced by knote_drop().

Here is the call graph for this function:



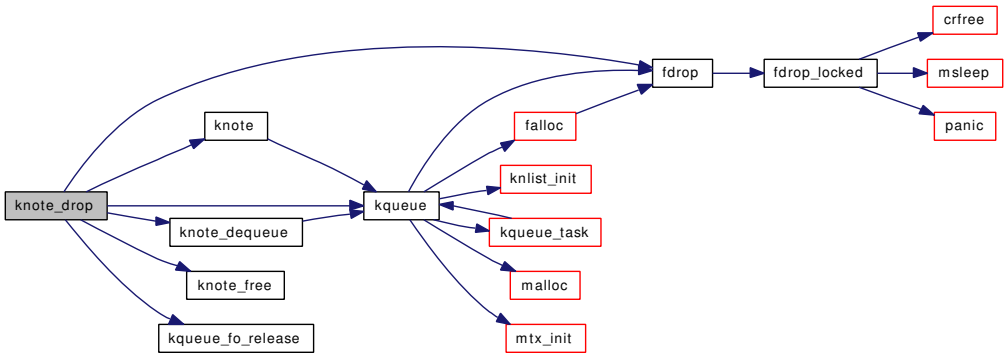
9.25.2.32 static void knote_drop (struct knote * *kn*, struct thread * *td*) [static]

Definition at line 1864 of file kern_event.c.

References fdrop(), KN_HASH, knote(), knote_dequeue(), knote_free(), KQ_LOCK, KQ_NOTOWNED, KQ_UNLOCK_FLUX, kqueue(), and kqueue_fo_release().

Referenced by knlist_cleardel(), knote_fdclose(), kqueue_close(), kqueue_register(), and kqueue_scan().

Here is the call graph for this function:



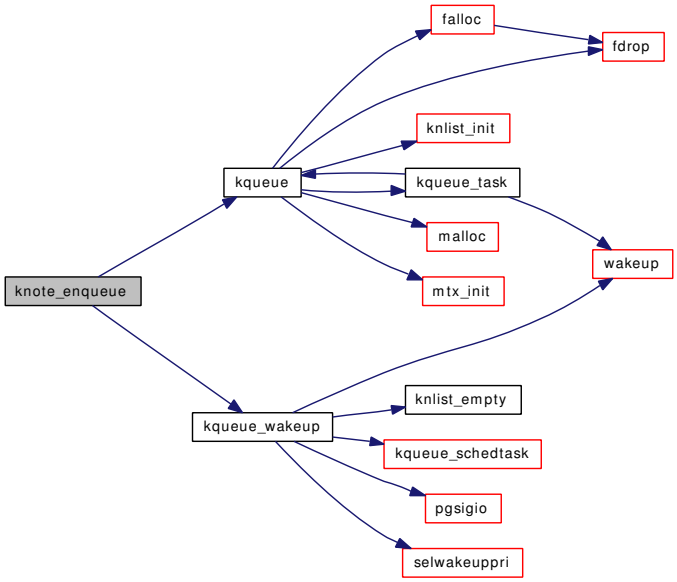
9.25.2.33 static void knote_enqueue (struct knote * *kn*) [static]

Definition at line 1897 of file kern_event.c.

References KQ_OWNED, kqueue(), and kqueue_wakeup().

Referenced by kqueue_register().

Here is the call graph for this function:



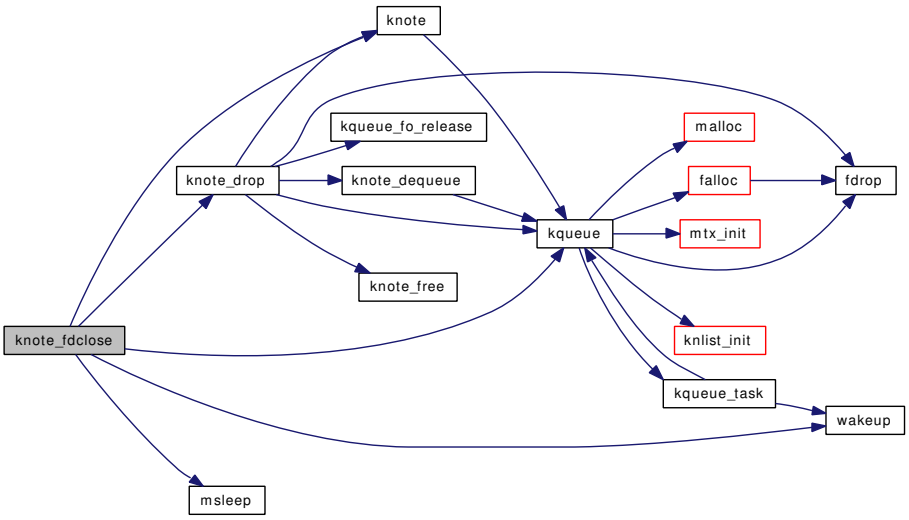
9.25.2.34 void knote_fdclose (struct thread * *td*, int *fd*)

Definition at line 1795 of file kern_event.c.

References knote(), knote_drop(), KQ_LOCK, KQ_UNLOCK, KQ_UNLOCK_FLUX, kqueue(), msleep(), and wakeup().

Referenced by do_dup(), fdcloseexec(), kern_close(), and setugidsafety().

Here is the call graph for this function:



9.25.2.35 static void knote_free (struct knote * *kn*) [static]

Definition at line 1940 of file kern_event.c.

References knote_zone.

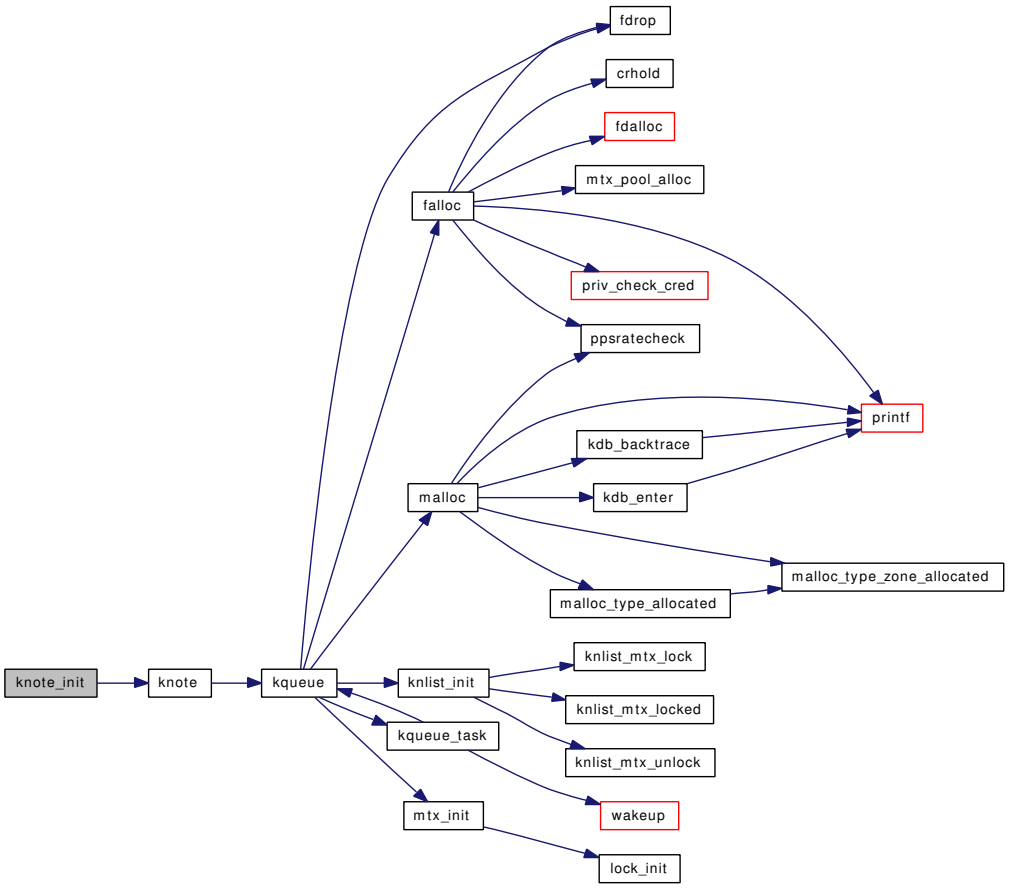
Referenced by knote_drop(), kqueue_register(), and kqueue_scan().

9.25.2.36 static void knote_init (void) [static]

Definition at line 1924 of file kern_event.c.

References knote(), and knote_zone.

Here is the call graph for this function:



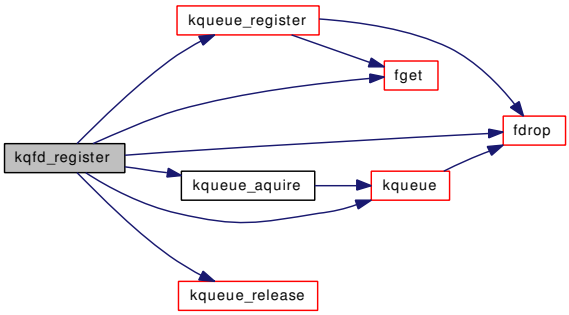
9.25.2.37 int kqfd_register (int *fd*, struct kevent * *kev*, struct thread * *td*, int *waitok*)

Definition at line 1950 of file kern_event.c.

References fdrop(), fget(), kqueue(), kqueue_aquire(), kqueue_register(), and kqueue_release().

Referenced by aio_aqueue(), and do_lio_listio().

Here is the call graph for this function:



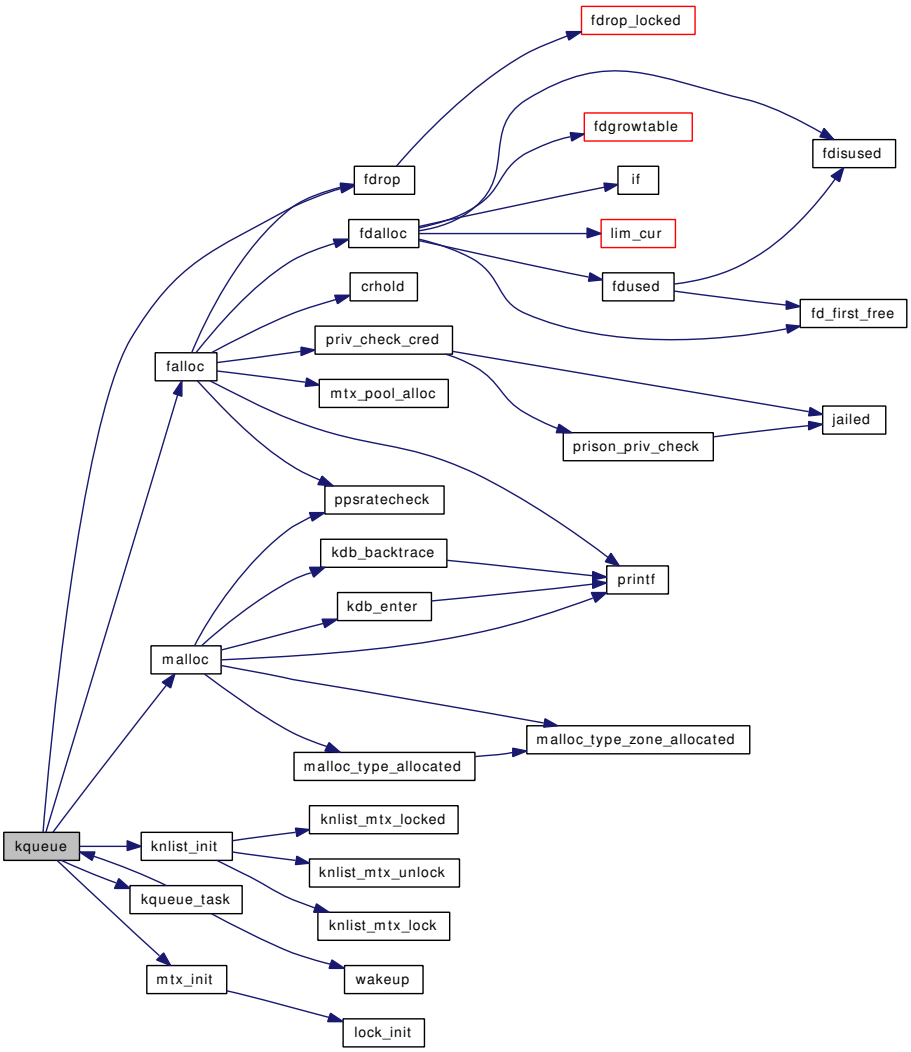
9.25.2.38 int kqueue (struct thread * *td*, struct kqueue_args * *uap*)

Definition at line 513 of file kern_event.c.

References falloc(), fdrop(), knlist_init(), kqueue_task(), kqueueops, malloc(), and mtx_init().

Referenced by filt_kqdetach(), filt_kqueue(), kern_kevent(), knlist_cleardel(), knote(), knote_dequeue(), knote_drop(), knote_enqueue(), knote_fdclose(), kqfd_register(), kqueue_aquire(), kqueue_close(), kqueue_ioctl(), kqueue_kqfilter(), kqueue_poll(), and kqueue_task().

Here is the call graph for this function:



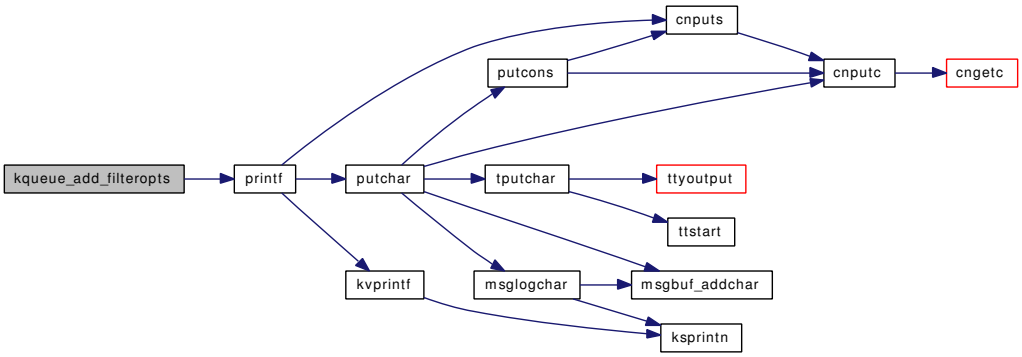
9.25.2.39 int kqueue_add_filteropts (int *filt*, struct filterops * *filtops*)

Definition at line 711 of file kern_event.c.

References filterops_lock, for_fop, null_filtops, printf(), and sysfilt_ops.

Referenced by aio_onceonly().

Here is the call graph for this function:



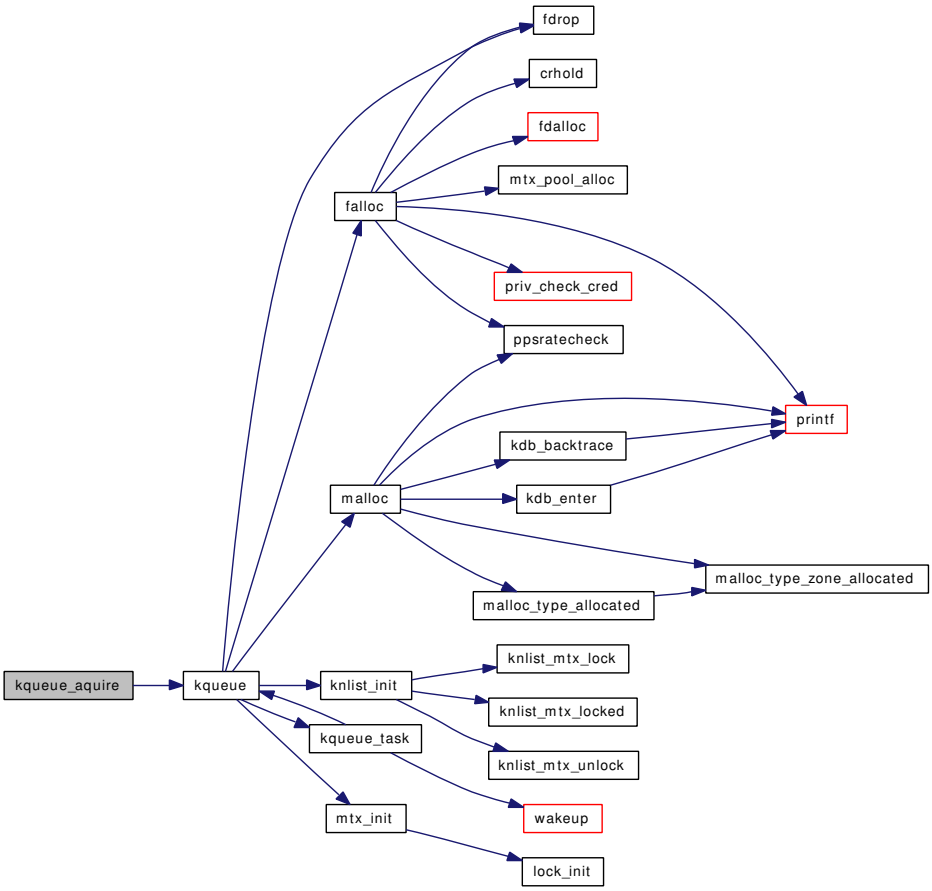
9.25.2.40 static int kqueue_aquire (struct file * *fp*, struct kqueue ** *kqp*) [static]

Definition at line 993 of file kern_event.c.

References KQ_LOCK, KQ_UNLOCK, and kqueue().

Referenced by kern_kevent(), kqfd_register(), kqueue_close(), and kqueue_poll().

Here is the call graph for this function:

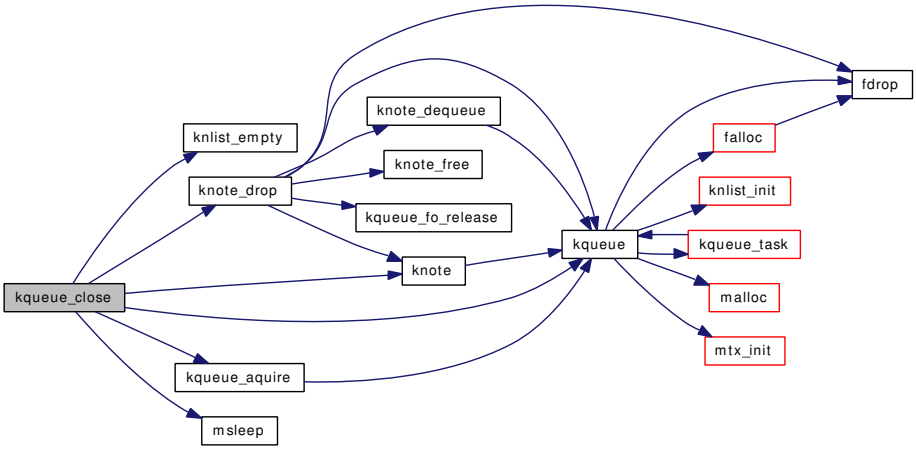


9.25.2.41 static int kqueue_close (struct file * *fp*, struct thread * *td*) [static]

Definition at line 1438 of file kern_event.c.

References knlist_empty(), knote(), knote_drop(), KQ_LOCK, KQ_UNLOCK, kqueue(), kqueue_aquire(), and msleep().

Here is the call graph for this function:



9.25.2.42 int kqueue_del_filteropts (int *filt*)

Definition at line 735 of file kern_event.c.

References filterops_lock, for_fop, for_refcnt, null_filtops, and sysfilt_ops.

Referenced by aio_unload().

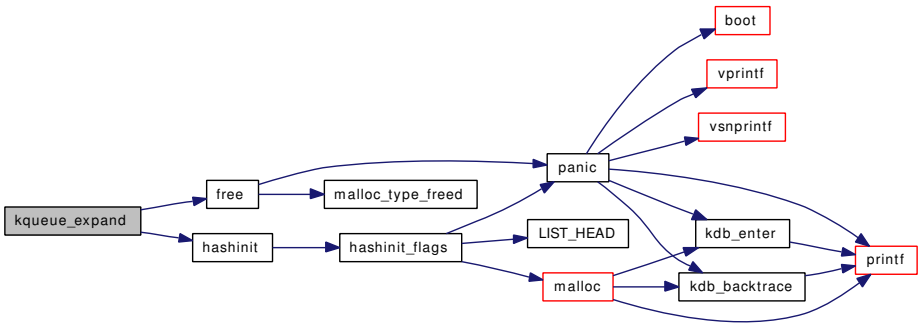
9.25.2.43 static int kqueue_expand (struct kqueue * *kq*, struct filterops * *fops*, uintptr_t *ident*, int *waitok*) [static]

Definition at line 1060 of file kern_event.c.

References free(), hashinit(), KN_HASHSIZE, KQ_LOCK, KQ_NOTOWNED, and KQ_UNLOCK.

Referenced by kqueue_register().

Here is the call graph for this function:



9.25.2.44 static struct filterops * kqueue_fo_find (int *filt*) [static]

Definition at line 759 of file kern_event.c.

References filterops_lock, for_fop, null_filtops, and sysfilt_ops.

Referenced by kqueue_register().

9.25.2.45 static void kqueue_fo_release (int *filt*) [static]

Definition at line 775 of file kern_event.c.

References filterops_lock, for_refcnt, and sysfilt_ops.

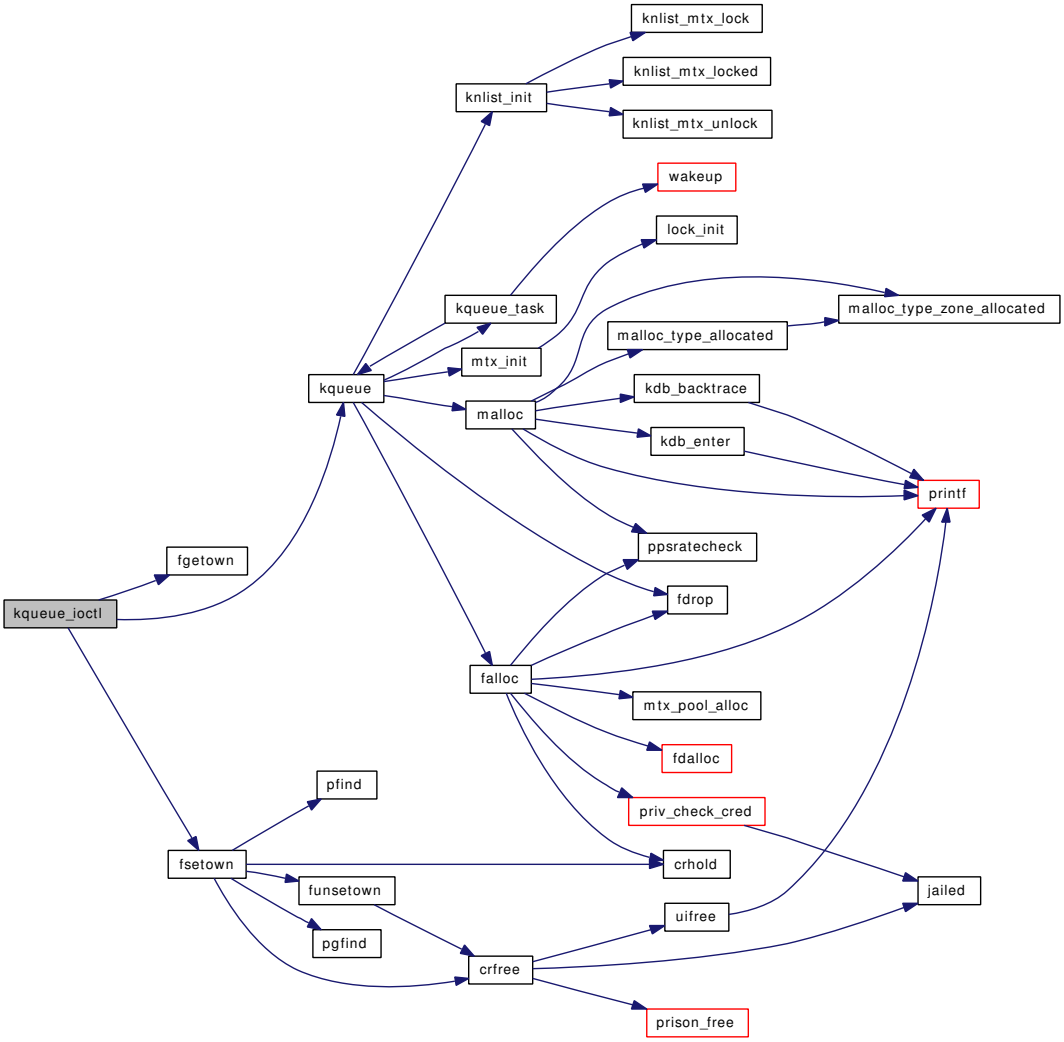
Referenced by knote_drop(), and kqueue_register().

9.25.2.46 static int kqueue_ioctl (struct file * *fp*, u_long *cmd*, void * *data*, struct ucred * *active_cred*, struct thread * *td*) [static]

Definition at line 1346 of file kern_event.c.

References fgetown(), fsetown(), and kqueue().

Here is the call graph for this function:

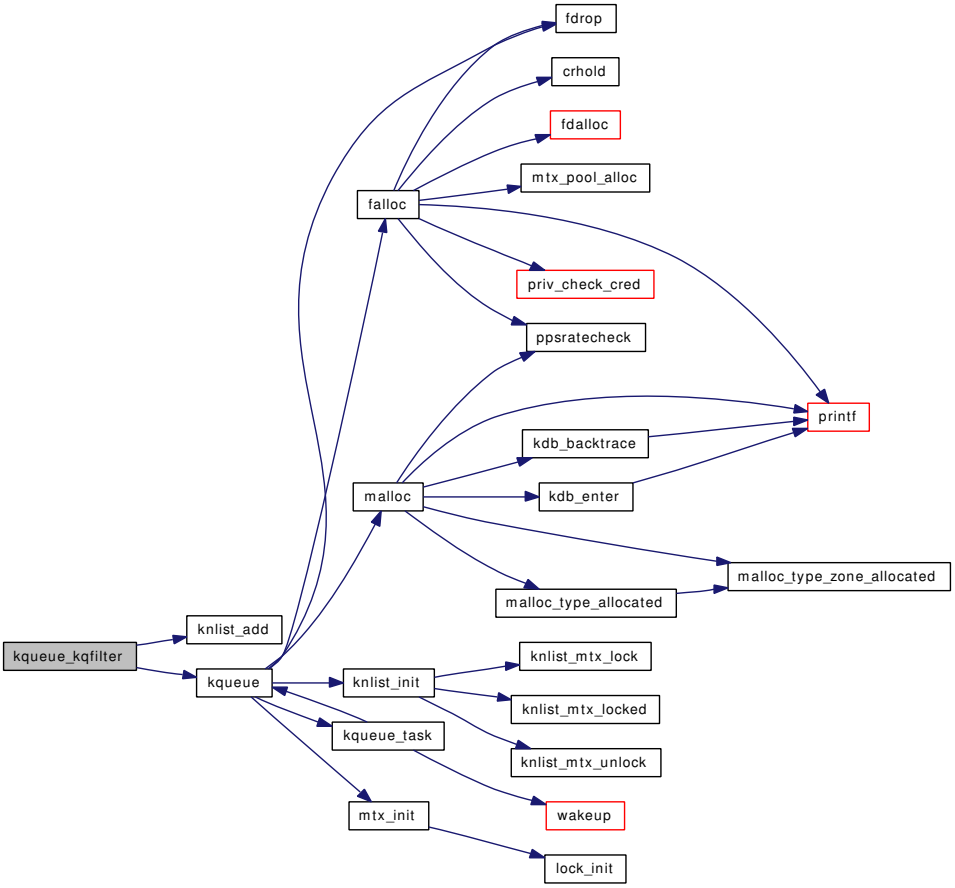


9.25.2.47 static int kqueue_kqfilter (struct file * *fp*, struct knote * *kn*) [static]

Definition at line 273 of file kern_event.c.

References knlist_add(), kqread_filtops, and kqueue().

Here is the call graph for this function:

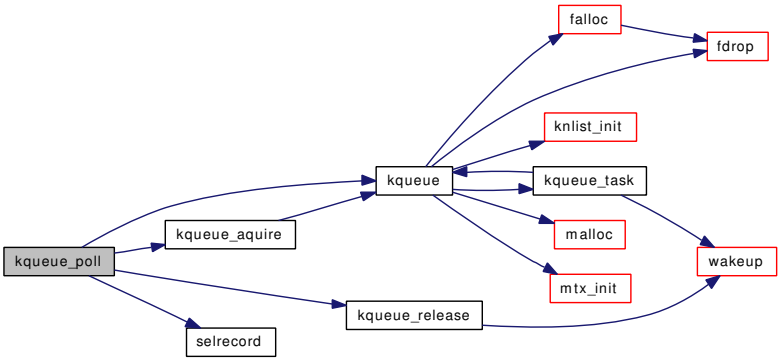


9.25.2.48 static int kqueue_poll (struct file * *fp*, int *events*, struct ucred * *active_cred*, struct thread * *td*) [static]

Definition at line 1394 of file kern_event.c.

References KQ_LOCK, KQ_UNLOCK, kqueue(), kqueue_aquire(), kqueue_release(), and selrecord().

Here is the call graph for this function:



9.25.2.49 static int kqueue_read (struct file * *fp*, struct uio * *uio*, struct ucred * *active_cred*, int *flags*, struct thread * *td*) [static]

Definition at line 1330 of file kern_event.c.

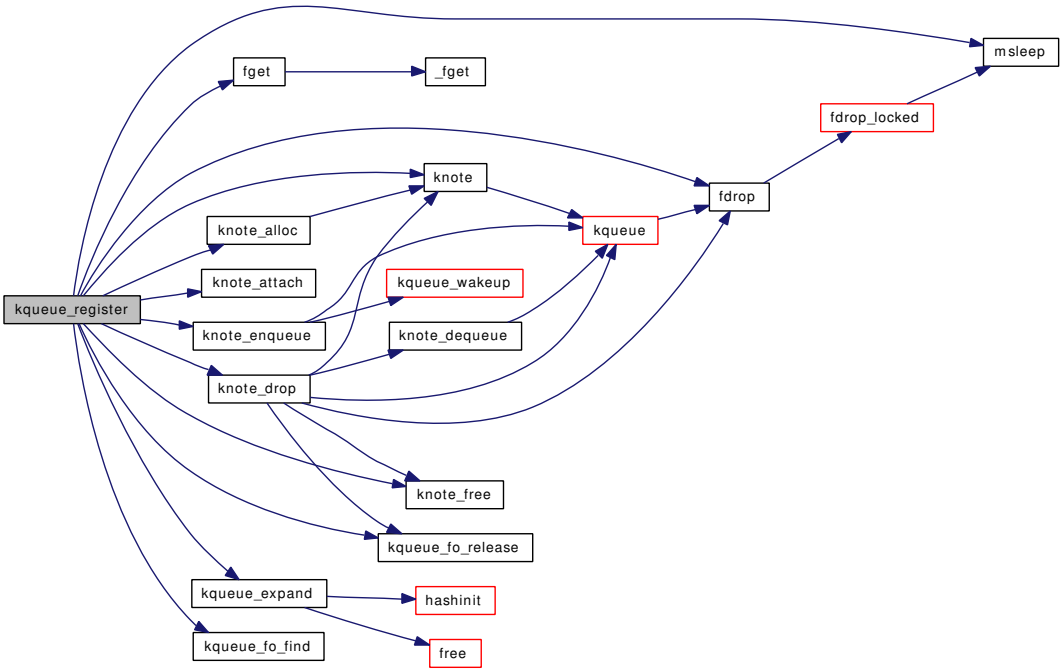
9.25.2.50 static int kqueue_register (struct kqueue * *kq*, struct kevent * *kev*, struct thread * *td*, int *waitok*) [static]

Definition at line 794 of file kern_event.c.

References fdrop(), fget(), KN_HASH, KN_LIST_LOCK, KN_LIST_UNLOCK, knote(), KNOTE_ACTIVATE, knote_alloc(), knote_attach(), knote_drop(), knote_enqueue(), knote_free(), kq_global, KQ_GLOBAL_LOCK, KQ_GLOBAL_UNLOCK, KQ_LOCK, KQ_UNLOCK, KQ_UNLOCK_FLUX, kqueue_expand(), kqueue_fo_find(), kqueue_fo_release(), and msleep().

Referenced by filt_proc(), kern_kevent(), and kqfd_register().

Here is the call graph for this function:



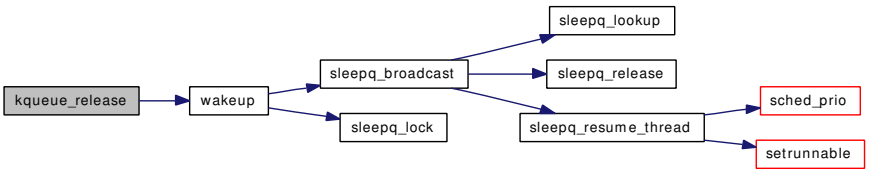
9.25.2.51 static void kqueue_release (struct kqueue * *kq*, int *locked*) [static]

Definition at line 1023 of file kern_event.c.

References KQ_LOCK, KQ_OWNED, KQ_UNLOCK, and wakeup().

Referenced by kern_kevent(), kqfd_register(), and kqueue_poll().

Here is the call graph for this function:



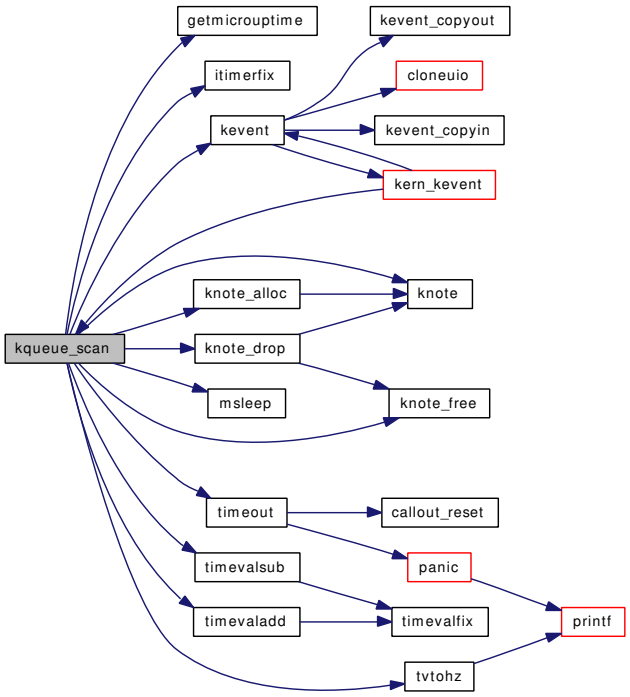
9.25.2.52 static int kqueue_scan (struct kqueue * *kq*, int *maxevents*, struct kevent_copyops * *k_ops*, const struct timespec * *timeout*, struct kevent * *keva*, struct thread * *td*) [static]

Definition at line 1148 of file kern_event.c.

References getmicrouptime(), hz, itimerfix(), kevent(), KN_LIST_LOCK, KN_LIST_UNLOCK, knote(), knote_alloc(), knote_drop(), knote_free(), KQ_FLUX_WAKEUP, kq_global, KQ_GLOBAL_LOCK, KQ_GLOBAL_UNLOCK, KQ_LOCK, KQ_NOTOWNED, KQ_OWNED, KQ_UNLOCK, KQ_UNLOCK_FLUX, msleep(), timeout(), timevaladd(), timevalsub(), and tvtohz().

Referenced by kern_kevent().

Here is the call graph for this function:



9.25.2.53 static void kqueue_schedtask (struct kqueue * *kq*) [static]

Definition at line 1037 of file kern_event.c.

References KQ_OWNED, and taskqueue_enqueue().

Referenced by kqueue_wakeup().

Here is the call graph for this function:

9.25.2.54 static int kqueue_stat (struct file * *fp*, struct stat * *st*, struct ucred * *active_cred*, struct thread * *td*) [static]

Definition at line 1420 of file kern_event.c.

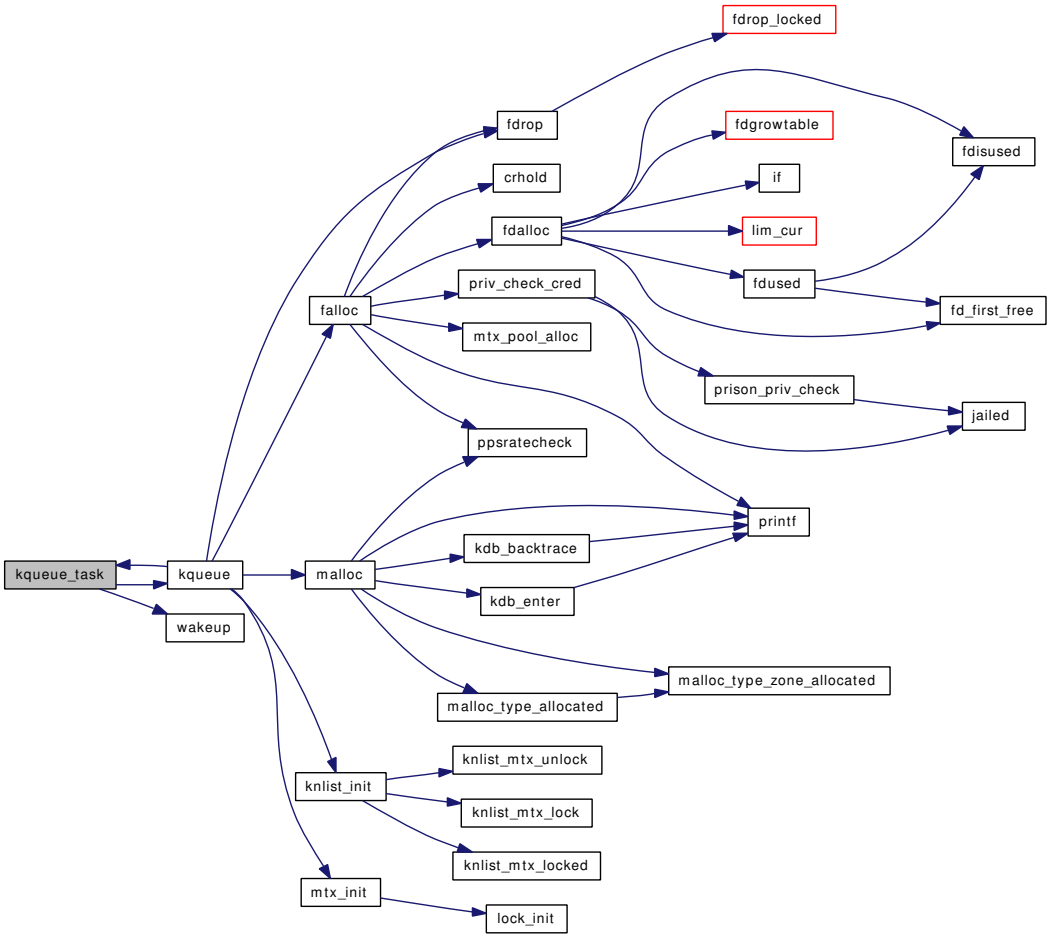
9.25.2.55 static void kqueue_task (void * *arg*, int *pending*) [static]

Definition at line 1122 of file kern_event.c.

References kq_global, KQ_GLOBAL_LOCK, KQ_GLOBAL_UNLOCK, KQ_LOCK, KQ_UNLOCK, kqueue(), and wakeup().

Referenced by kqueue().

Here is the call graph for this function:



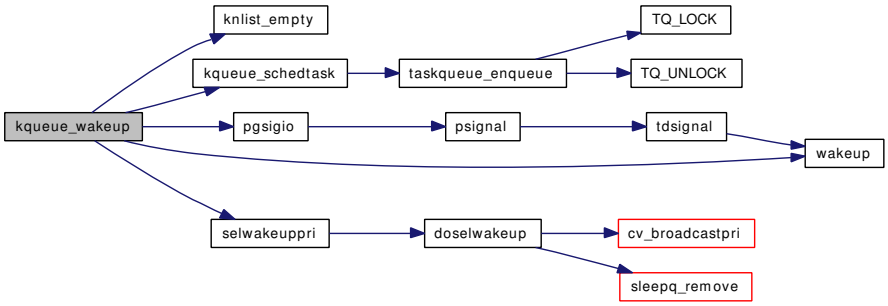
9.25.2.56 static void kqueue_wakeup (struct kqueue * *kq*) [static]

Definition at line 1523 of file kern_event.c.

References knlist_empty(), KQ_OWNED, kqueue_schedtask(), pgsigio(), selwakeuppri(), and wakeup().

Referenced by knote_enqueue().

Here is the call graph for this function:



9.25.2.57 static int kqueue_write (struct file * *fp*, struct uio * *uio*, struct ucred * *active_cred*, int *flags*, struct thread * *td*) [static]

Definition at line 1338 of file kern_event.c.

9.25.2.58 static MALLOC_DEFINE (M_KQUEUE, "kqueue", "memory for kqueue system") [static]

9.25.2.59 MTX_SYSINIT (knlist_lock, & *knlist_lock*, "knlist lock for lockless objects", MTX_DEF)

9.25.2.60 MTX_SYSINIT (kqueue_filterops, & *filterops_lock*, "protect sysfilt_ops", MTX_DEF)

9.25.2.61 MTX_SYSINIT (kq_global, & *kq_global*, "kqueue order", MTX_DEF)

9.25.2.62 SYSCTL_INT (_kern, OID_AUTO, kq_calloutmax, CTLFLAG_RW, & *kq_calloutmax*, 0, "Maximum number of callouts allocated for kqueue")

9.25.2.63 TASKQUEUE_DEFINE_THREAD (kqueue)

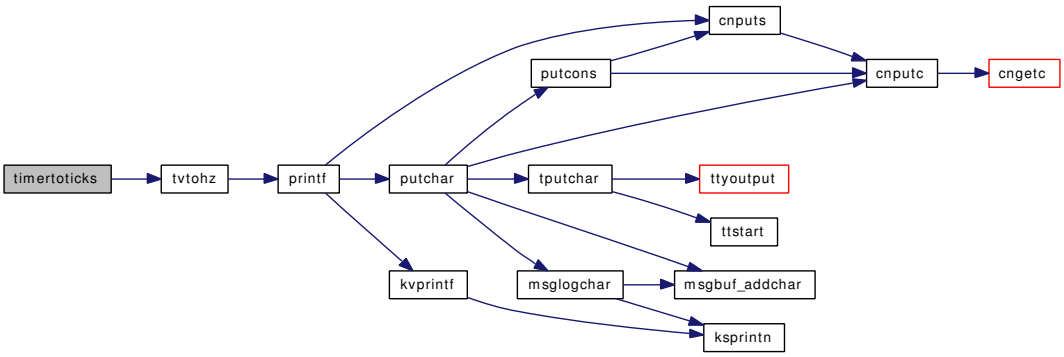
9.25.2.64 static int timertoticks (intptr_t *data*) [static]

Definition at line 431 of file kern_event.c.

References tvtohz().

Referenced by filt_timerattach(), and filt_timerexpire().

Here is the call graph for this function:



9.25.3 Variable Documentation

9.25.3.1 struct filterops file_filtops [static]

Initial value:

```
{ 1, filt_fileattach, NULL, NULL }
```

Definition at line 143 of file kern_event.c.

9.25.3.2 struct mtx filterops_lock [static]

Definition at line 241 of file kern_event.c.

Referenced by kqueue_add_filteropts(), kqueue_del_filteropts(), kqueue_fo_find(), and kqueue_fo_release().

9.25.3.3 struct filterops* for_fop

Definition at line 245 of file kern_event.c.

Referenced by kqueue_add_filteropts(), kqueue_del_filteropts(), and kqueue_fo_find().

9.25.3.4 int for_refcnt

Definition at line 246 of file kern_event.c.

Referenced by kqueue_del_filteropts(), and kqueue_fo_release().

9.25.3.5 struct filterops fs_filtops

Definition at line 3665 of file vfs_subr.c.

9.25.3.6 struct mtx knlist_lock [static]

Definition at line 1661 of file kern_event.c.

Referenced by knlist_init().

9.25.3.7 uma_zone_t knote_zone [static]

Definition at line 153 of file kern_event.c.

Referenced by knote_alloc(), knote_free(), and knote_init().

9.25.3.8 int kq_calloutmax = (4 * 1024) [static]

Definition at line 155 of file kern_event.c.

Referenced by filt_timerattach().

9.25.3.9 struct mtx kq_global [static]

Definition at line 74 of file kern_event.c.

Referenced by kqueue_register(), kqueue_scan(), and kqueue_task().

9.25.3.10 int kq_ncallouts = 0 [static]

Definition at line 154 of file kern_event.c.

Referenced by filt_timerattach(), and filt_timerdetach().

9.25.3.11 struct filterops kqread_filtops [static]

Initial value:

```
{ 1, NULL, filt_kqdetach, filt_kqueue }
```

Definition at line 145 of file kern_event.c.

Referenced by kqueue_kqfilter().

9.25.3.12 fo_close_t kqueue_close [static]

Definition at line 112 of file kern_event.c.

9.25.3.13 fo_ioctl_t kqueue_ioctl [static]

Definition at line 108 of file kern_event.c.

9.25.3.14 fo_kqfilter_t kqueue_kqfilter [static]

Definition at line 110 of file kern_event.c.

9.25.3.15 fo_poll_t kqueue_poll [static]

Definition at line 109 of file kern_event.c.

9.25.3.16 fo_rdwr_t kqueue_read [static]

Definition at line 106 of file kern_event.c.

9.25.3.17 fo_stat_t kqueue_stat [static]

Definition at line 111 of file kern_event.c.

9.25.3.18 fo_rdwr_t kqueue_write [static]

Definition at line 107 of file kern_event.c.

9.25.3.19 struct fileops kqueueops [static]

Initial value:

```
 {
        .fo_read = kqueue_read,
        .fo_write = kqueue_write,
        .fo_ioctl = kqueue_ioctl,
        .fo_poll = kqueue_poll,
        .fo_kqfilter = kqueue_kqfilter,
        .fo_stat = kqueue_stat,
        .fo_close = kqueue_close,
}
```

Definition at line 114 of file kern_event.c.

Referenced by kqueue().

9.25.3.20 struct filterops null_filtops

Initial value:

```
        { 0, filt_nullattach, NULL, NULL }
```

Definition at line 231 of file kern_event.c.

Referenced by kqueue_add_filteropts(), kqueue_del_filteropts(), and kqueue_fo_find().

9.25.3.21 struct filterops proc_filtops [static]

Initial value:

```
        { 0, filt_procattach, filt_procdetach, filt_proc }
```

Definition at line 148 of file kern_event.c.

9.25.3.22 struct filterops sig_filtops

Definition at line 103 of file kern_sig.c.

9.25.3.23 struct { ... } sysfilt_ops[EVFILT_SYSCOUNT] [static]

Referenced by kqueue_add_filteropts(), kqueue_del_filteropts(), kqueue_fo_find(), and kqueue_fo_release().

9.25.3.24 struct filterops timer_filtops [static]

Initial value:

```
{ 0, filt_timerattach, filt_timerdetach, filt_timer }
```

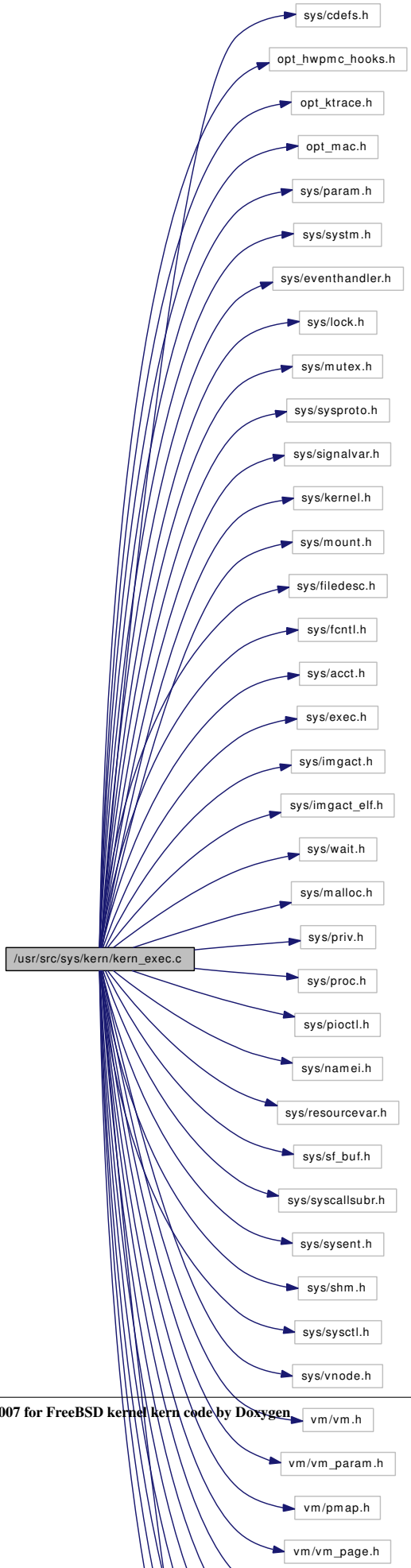
Definition at line 150 of file kern_event.c.

9.26 /usr/src/sys/kern/kern_exec.c File Reference

```
#include <sys/cdefs.h>
#include "opt_hwpmc_hooks.h"
#include "opt_ktrace.h"
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/eventhandler.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/sysproto.h>
#include <sys/signalvar.h>
#include <sys/kernel.h>
#include <sys/mount.h>
#include <sys/filedesc.h>
#include <sys/fcntl.h>
#include <sys/acct.h>
#include <sys/exec.h>
#include <sys/imgact.h>
#include <sys/imgact_elf.h>
#include <sys/wait.h>
#include <sys/malloc.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/pioctl.h>
#include <sys/namei.h>
#include <sys/resourcevar.h>
#include <sys/sf_buf.h>
#include <sys/syscallsubr.h>
#include <sys/sysent.h>
#include <sys/shm.h>
#include <sys/sysctl.h>
#include <sys/vnode.h>
#include <vm/vm.h>
#include <vm/vm_param.h>
#include <vm/pmap.h>
```

```
#include <vm/vm_page.h>
#include <vm/vm_map.h>
#include <vm/vm_kern.h>
#include <vm/vm_extern.h>
#include <vm/vm_object.h>
#include <vm/vm_pager.h>
#include <machine/reg.h>
#include <security/audit/audit.h>
#include <security/mac/mac_framework.h>
```

Include dependency graph for kern_exec.c:


/usr/src/sys/kern/kern_exec.c
sys/cdefs.h
opt_hwpmc_hooks.h
opt_ktrace.h
opt_mac.h
sys/param.h
sys/systm.h
sys/eventhandler.h
sys/lock.h
sys/mutex.h
sys/sysproto.h
sys/signalvar.h
sys/kernel.h
sys/mount.h
sys/filedesc.h
sys/fcntl.h
sys/acct.h
sys/exec.h
sys/imgact.h
sys/imgact_elf.h
sys/wait.h
sys/malloc.h
sys/priv.h
sys/proc.h
sys/pioctl.h
sys/namei.h
sys/resourcevar.h
sys/sf_buf.h
sys/syscallsubr.h
sys/sysent.h
sys/shm.h
sys/sysctl.h
sys/vnode.h
vm/vm.h
vm/vm_param.h
vm/pmap.h
vm/vm_page.h

Data Structures

- struct execve_args
- struct __mac_execve_args

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_exec.c,v 1.299 2006/11/06 13:42:00 rwatson Exp $")
- MALLOC_DEFINE (M_PARGS,"proc-args","Process arguments")
- static int sysctl_kern_ps_strings (SYSCTL_HANDLER_ARGS)
- static int sysctl_kern_usrstack (SYSCTL_HANDLER_ARGS)
- static int sysctl_kern_stackprot (SYSCTL_HANDLER_ARGS)
- static int do_execve (struct thread *td, struct image_args *args, struct mac *mac_p)
- static void exec_free_args (struct image_args *)
- SYSCTL_PROC (_kern, KERN_PS_STRINGS, ps_strings, CTLTYPE_ULONG|CTLFLAG_RD, NULL, 0, sysctl_kern_ps_strings,"LU","")
- SYSCTL_PROC (_kern, KERN_USRSTACK, usrstack, CTLTYPE_ULONG|CTLFLAG_RD, NULL, 0, sysctl_kern_usrstack,"LU","")
- SYSCTL_PROC (_kern, OID_AUTO, stackprot, CTLTYPE_INT|CTLFLAG_RD, NULL, 0, sysctl_kern_stackprot,"I","")
- SYSCTL_ULONG (_kern, OID_AUTO, ps_arg_cache_limit, CTLFLAG_RW,&ps_arg_cache_limit, 0,"")
- int execve (struct thread *td, struct execve_args *uap)
- int __mac_execve (struct thread *td, struct __mac_execve_args *uap)
- int kern_execve (struct thread *td, struct image_args *args, struct mac *mac_p)
- int exec_map_first_page (struct image_params *imgp)
- void exec_unmap_first_page (struct image_params *imgp)
- int exec_new_vmspace (struct image_params *imgp, struct sysentvec *sv)
- int exec_copyin_args (struct image_args *args, char *fname, enum uio_seg segflg, char **argv, char **envv)
- register_t * exec_copyout_strings (struct image_params *imgp)
- int exec_check_permissions (struct image_params *imgp)
- int exec_register (struct execsw *execsw_arg) const
- int exec_unregister (struct execsw *execsw_arg) const

Variables

- u_long ps_arg_cache_limit = PAGE_SIZE / 16
- static struct execsw ** execsw

9.26.1 Function Documentation

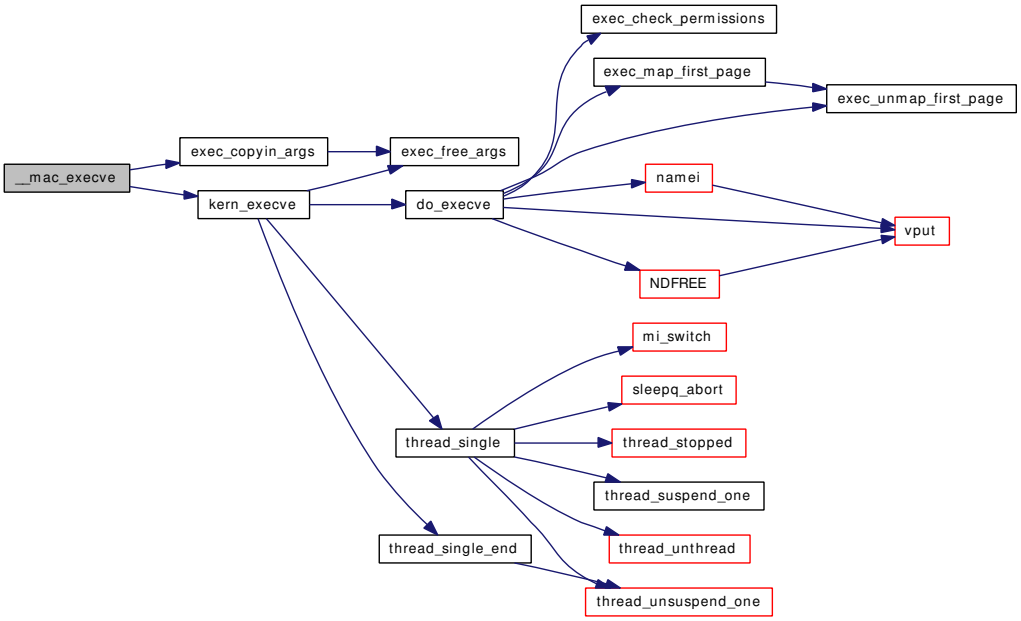
9.26.1.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_exec. *c*, v 1.299 2006/11/06 13:42:00 rwatson Exp $")

9.26.1.2 int __mac_execve (struct thread * *td*, struct __mac_execve_args * *uap*)

Definition at line 206 of file kern_exec.c.

References __mac_execve_args::argv, __mac_execve_args::envv, exec_copyin_args(), __mac_execve_-args::fname, kern_execve(), and __mac_execve_args::mac_p.

Here is the call graph for this function:



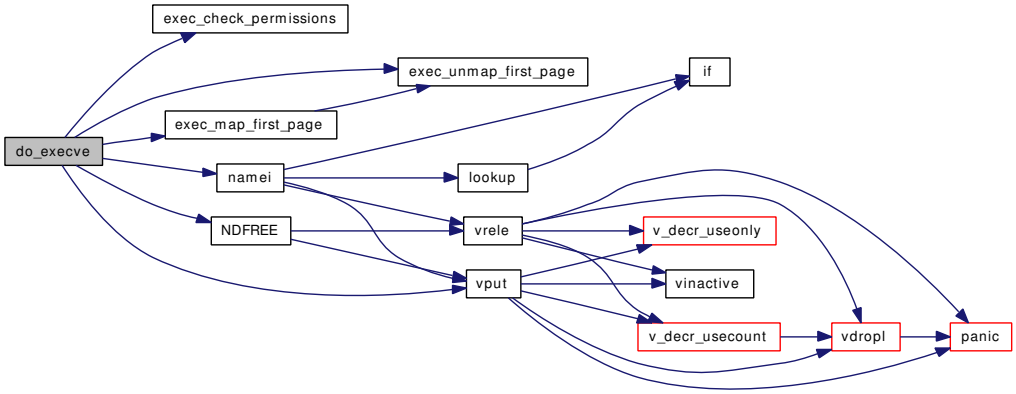
9.26.1.3 static int do_execve (struct thread * *td*, struct image_args * *args*, struct mac * *mac_p*) [static]

Definition at line 284 of file kern_exec.c.

References exec_check_permissions(), exec_map_first_page(), exec_unmap_first_page(), execsw, namei(), NDFREE(), and vput().

Referenced by kern_execve().

Here is the call graph for this function:



9.26.1.4 int exec_check_permissions (struct image_params * *imgp*)

Definition at line 1186 of file kern_exec.c.

References td.

Referenced by do_execve().

9.26.1.5 int exec_copyin_args (struct image_args * *args*, char * *fname*, enum uio_seg *segflg*, char ** *argv*, char ** *envv*)

Definition at line 965 of file kern_exec.c.

References exec_free_args().

Referenced by __mac_execve(), execve(), and kse_thr_interrupt().

Here is the call graph for this function:

9.26.1.6 register_t* exec_copyout_strings (struct image_params * *imgp*)

Definition at line 1069 of file kern_exec.c.

References if(), and suword.

Here is the call graph for this function:

9.26.1.7 static void exec_free_args (struct image_args *) [static]

Definition at line 1053 of file kern_exec.c.

Referenced by exec_copyin_args(), and kern_execve().

9.26.1.8 int exec_map_first_page (struct image_params * *imgp*)

Definition at line 810 of file kern_exec.c.

References exec_unmap_first_page().

Referenced by do_execve().

Here is the call graph for this function:

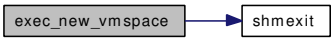
9.26.1.9 int exec_new_vmspace (struct image_params * *imgp*, struct sysentvec * *sv*)

Definition at line 892 of file kern_exec.c.

References maxssiz, sgrowsiz, and shmexit().

Referenced by do_aout_hdr(), and exec_aout_imgact().

Here is the call graph for this function:

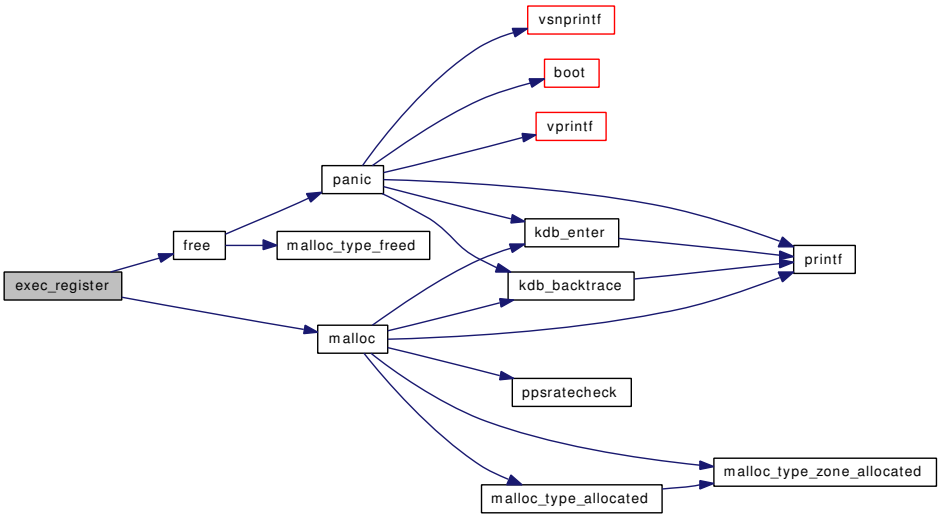


9.26.1.10 int exec_register (struct execsw * *execsw_arg*) const

Definition at line 1252 of file kern_exec.c.

References execsw, free(), and malloc().

Here is the call graph for this function:



9.26.1.11 void exec_unmap_first_page (struct image_params * *imgp*)

Definition at line 871 of file kern_exec.c.

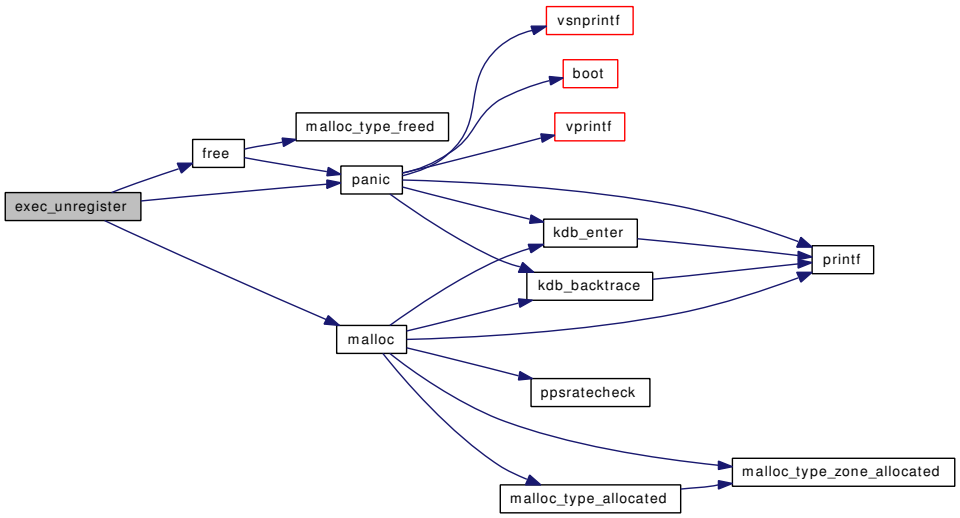
Referenced by do_execve(), and exec_map_first_page().

9.26.1.12 int exec_unregister (struct execsw * *execsw_arg*) const

Definition at line 1277 of file kern_exec.c.

References execsw, free(), malloc(), and panic().

Here is the call graph for this function:



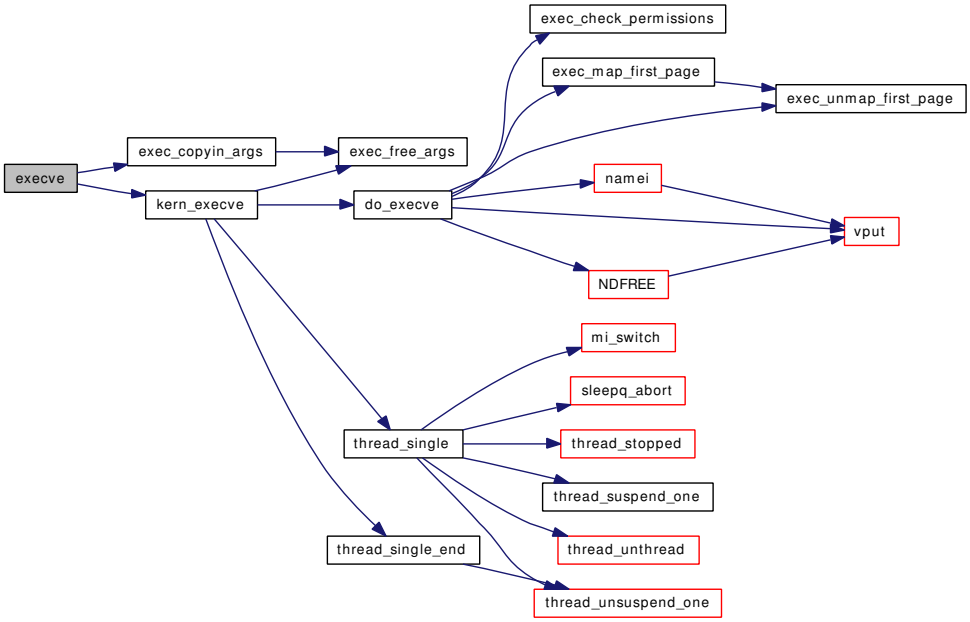
9.26.1.13 int execve (struct thread * *td*, struct execve_args * *uap*)

Definition at line 175 of file kern_exec.c.

References execve_args::argv, execve_args::envv, exec_copyin_args(), execve_args::fname, and kern_-execve().

Referenced by start_init().

Here is the call graph for this function:



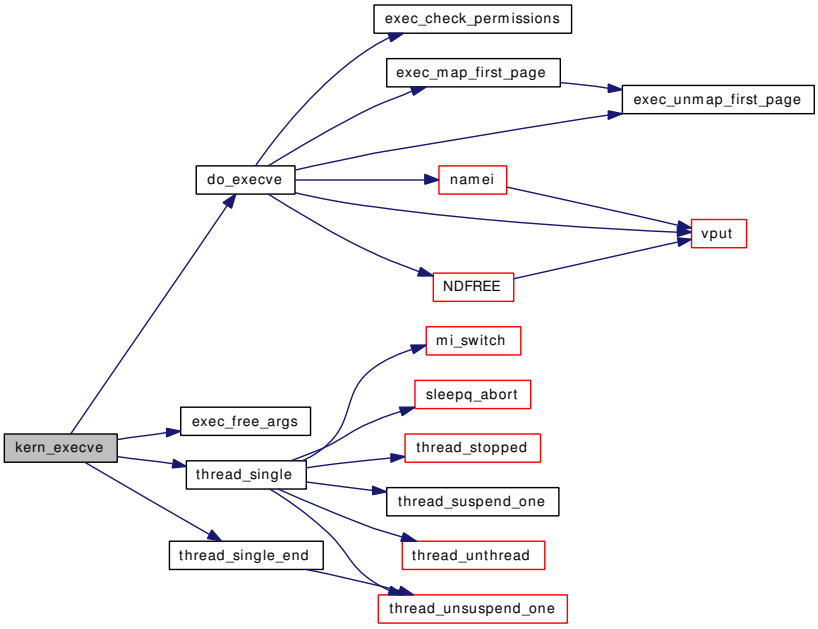
9.26.1.14 int kern_execve (struct thread * *td*, struct image_args * *args*, struct mac * *mac_p*)

Definition at line 237 of file kern_exec.c.

References do_execve(), exec_free_args(), thread_single(), and thread_single_end().

Referenced by __mac_execve(), execve(), and kse_thr_interrupt().

Here is the call graph for this function:



9.26.1.15 MALLOC_DEFINE (M_PARGS, "proc-args", "Process arguments")

9.26.1.16 static int sysctl_kern_ps_strings (SYSCTL_HANDLER_ARGS) [static]

Definition at line 110 of file kern_exec.c.

9.26.1.17 static int sysctl_kern_stackprot (SYSCTL_HANDLER_ARGS) [static]

Definition at line 148 of file kern_exec.c.

9.26.1.18 static int sysctl_kern_usrstack (SYSCTL_HANDLER_ARGS) [static]

Definition at line 129 of file kern_exec.c.

9.26.1.19 SYSCTL_PROC (_kern, OID_AUTO, stackprot, CTLTYPE_INT| *CTLFLAG_RD*, NULL, 0, sysctl_kern_stackprot, "I", "")

9.26.1.20 SYSCTL_PROC (_kern, KERN_USRSTACK, usrstack, CTLTYPE_ULONG| *CTLFLAG_RD*, NULL, 0, sysctl_kern_usrstack, "LU", "")

9.26.1.21 SYSCTL_PROC (_kern, KERN_PS_STRINGS, ps_strings, CTLTYPE_ULONG| *CTLFLAG_RD*, NULL, 0, sysctl_kern_ps_strings, "LU", "")

9.26.1.22 SYSCTL_ULONG (_kern, OID_AUTO, ps_arg_cache_limit, CTLFLAG_RW, & *ps_arg_cache_limit*, 0, "")

9.26.2 Variable Documentation

9.26.2.1 struct execsw** execsw [static]

Definition at line 161 of file kern_exec.c.

Referenced by do_execve(), exec_register(), and exec_unregister().

9.26.2.2 u_long ps_arg_cache_limit = PAGE_SIZE / 16

Definition at line 105 of file kern_exec.c.

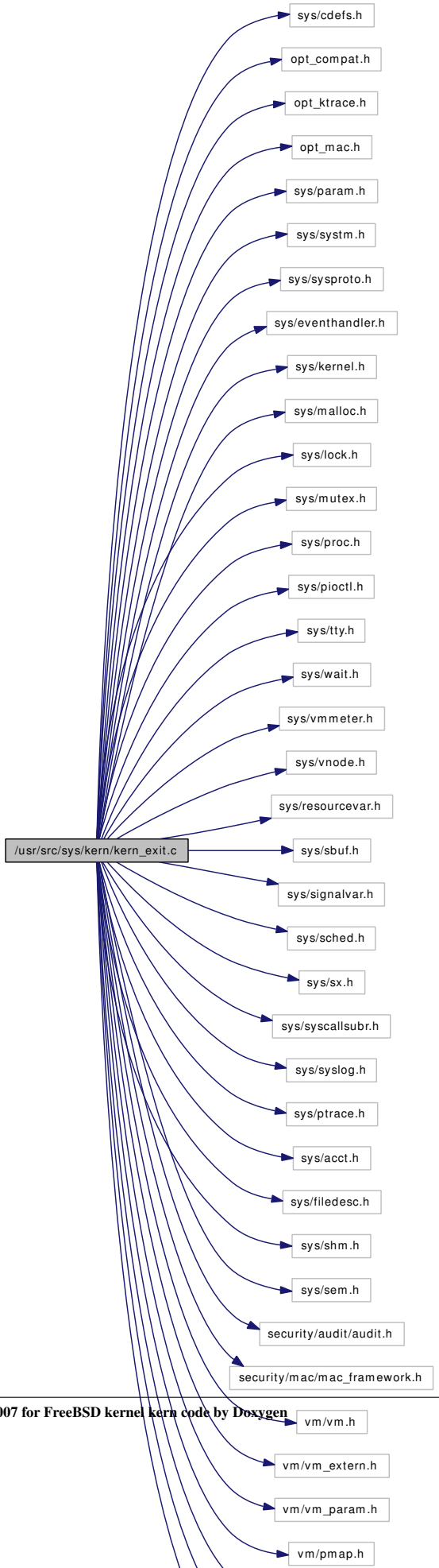
Referenced by sysctl_kern_proc_args().

9.27 /usr/src/sys/kern/kern_exit.c File Reference

```
#include <sys/cdefs.h>
#include "opt_compat.h"
#include "opt_ktrace.h"
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/sysproto.h>
#include <sys/eventhandler.h>
#include <sys/kernel.h>
#include <sys/malloc.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/pioctl.h>
#include <sys/tty.h>
#include <sys/wait.h>
#include <sys/vmmeter.h>
#include <sys/vnode.h>
#include <sys/resourcevar.h>
#include <sys/sbuf.h>
#include <sys/signalvar.h>
#include <sys/sched.h>
#include <sys/sx.h>
#include <sys/syscallsubr.h>
#include <sys/syslog.h>
#include <sys/ptrace.h>
#include <sys/acct.h>
#include <sys/filedesc.h>
#include <sys/shm.h>
#include <sys/sem.h>
#include <security/audit/audit.h>
#include <security/mac/mac_framework.h>
#include <vm/vm.h>
#include <vm/vm_extern.h>
#include <vm/vm_param.h>
```

```
#include <vm/pmap.h>
#include <vm/vm_map.h>
#include <vm/vm_page.h>
#include <vm/uma.h>
```

Include dependency graph for kern_exit.c:


/usr/src/sys/kern/kern_exit.c
sys/cdefs.h
opt_compat.h
opt_ktrace.h
opt_mac.h
sys/param.h
sys/systm.h
sys/sysproto.h
sys/eventhandler.h
sys/kernel.h
sys/malloc.h
sys/lock.h
sys/mutex.h
sys/proc.h
sys/pioctl.h
sys/tty.h
sys/wait.h
sys/vmmeter.h
sys/vnode.h
sys/resourcevar.h
sys/sbuf.h
sys/signalvar.h
sys/sched.h
sys/sx.h
sys/syscallsubr.h
sys/syslog.h
sys/ptrace.h
sys/acct.h
sys/filedesc.h
sys/shm.h
sys/sem.h
security/audit/audit.h
security/mac/mac_framework.h
vm/vm.h
vm/vm_extern.h
vm/vm_param.h
vm/pmap.h

Data Structures

- struct abort2_args

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_exit.c,v 1.294 2006/10/25 06:18:04 davidxu Exp $")
- MALLOC_DEFINE (M_ZOMBIE,"zombie","zombie proc status")
- void sys_exit (struct thread *td, struct sys_exit_args *uap)
- void exit1 (struct thread *td, int rv)
- int abort2 (struct thread *td, struct abort2_args *uap)
- int wait4 (struct thread *td, struct wait_args *uap)
- int kern_wait (struct thread *td, pid_t pid, int *status, int options, struct rusage *rusage)
- void proc_reparent (struct proc *child, struct proc *parent)

Variables

- void(*) nlminfo_release_p (struct proc *p)

9.27.1 Function Documentation

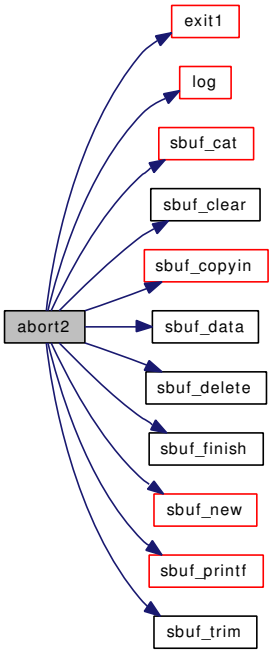
9.27.1.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_exit. *c*, v 1.294 2006/10/25 06:18:04 davidxu Exp $")

9.27.1.2 int abort2 (struct thread * *td*, struct abort2_args * *uap*)

Definition at line 560 of file kern_exit.c.

References abort2_args::args, exit1(), log(), abort2_args::nargs, sbuf_cat(), sbuf_clear(), sbuf_copyin(), sbuf_data(), sbuf_delete(), sbuf_finish(), sbuf_new(), sbuf_printf(), sbuf_trim(), and abort2_args::why.

Here is the call graph for this function:



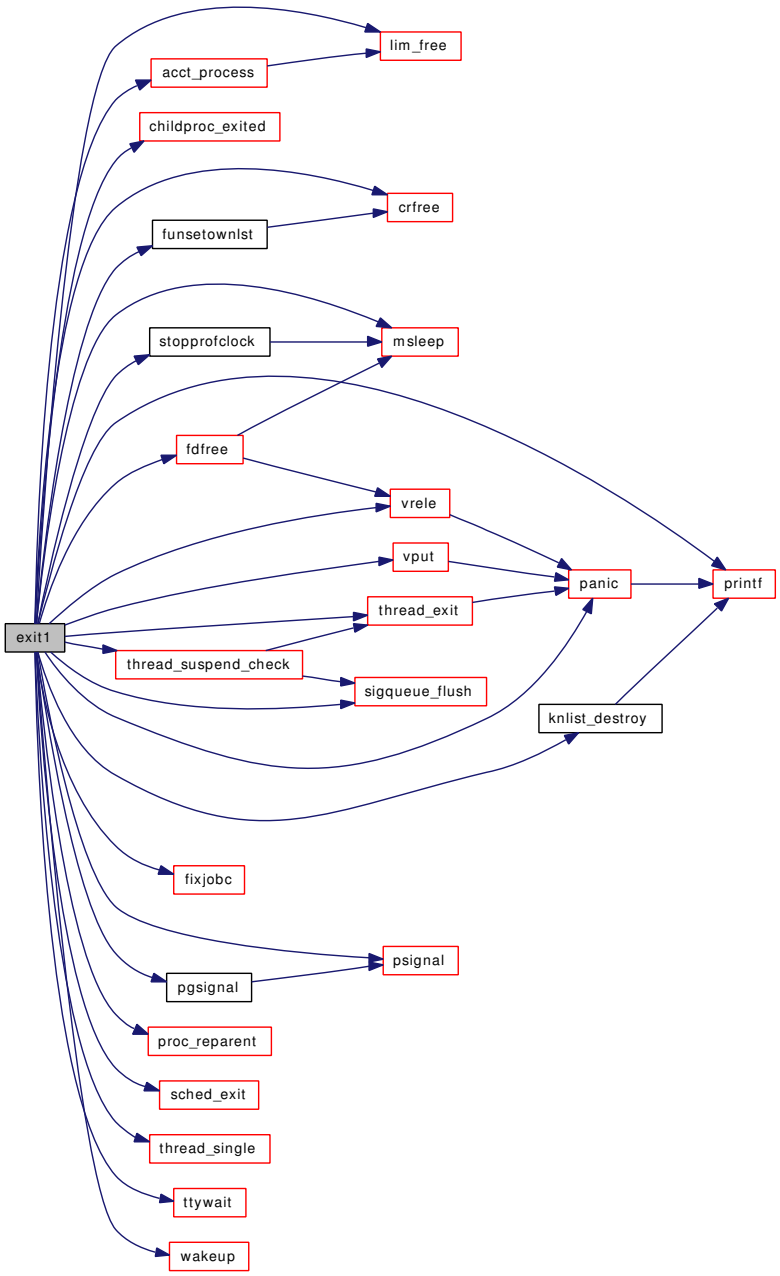
9.27.1.3 void exit1 (struct thread * *td*, int *rv*)

Definition at line 111 of file kern_exit.c.

References acct_process(), allproc_lock, childproc_exited(), crfree(), fdfree(), fixjobc(), funsetownlst(), Giant, initproc, knlist_destroy(), lim_free(), msleep(), nlminfo_release_p, panic(), pgsignal(), ppeers_lock, printf(), proc_reparent(), proctree_lock, psignal(), sched_exit(), sched_lock, sigqueue_flush(), stopprofclock(), thread_exit(), thread_single(), thread_suspend_check(), ttywait(), vput(), vrele(), wakeup(), and zombproc.

Referenced by abort2(), kse_exit(), kse_release(), kthread_exit(), sigexit(), and sys_exit().

Here is the call graph for this function:



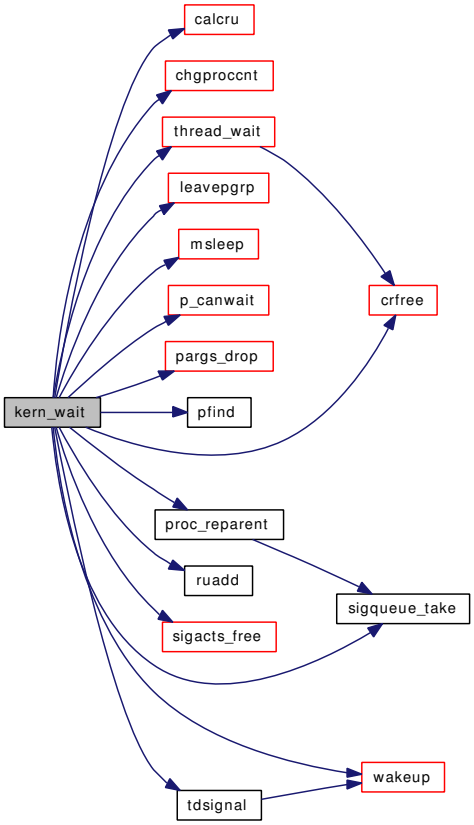
9.27.1.4 int kern_wait (struct thread * *td*, pid_t *pid*, int * *status*, int *options*, struct rusage * *rusage*)

Definition at line 670 of file kern_exit.c.

References allproc_lock, calcru(), chgproccnt(), crfree(), leavepgrp(), msleep(), nprocs, p_canwait(), pargs_drop(), pfind(), proc_reparent(), proc_zone, proctree_lock, ruadd(), sched_lock, sigacts_free(), sigqueue_take(), tdsignal(), thread_wait(), and wakeup().

Referenced by wait4().

Here is the call graph for this function:



9.27.1.5 MALLOC_DEFINE (M_ZOMBIE, "zombie", "zombie proc status")

9.27.1.6 void proc_reparent (struct proc * *child*, struct proc * *parent*)

Definition at line 894 of file kern_exit.c.

References proctree_lock, and sigqueue_take().

Referenced by exit1(), kern_ptrace(), kern_wait(), and kthread_exit().

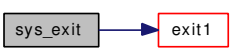
Here is the call graph for this function:

9.27.1.7 void sys_exit (struct thread * *td*, struct sys_exit_args * *uap*)

Definition at line 98 of file kern_exit.c.

References exit1().

Here is the call graph for this function:

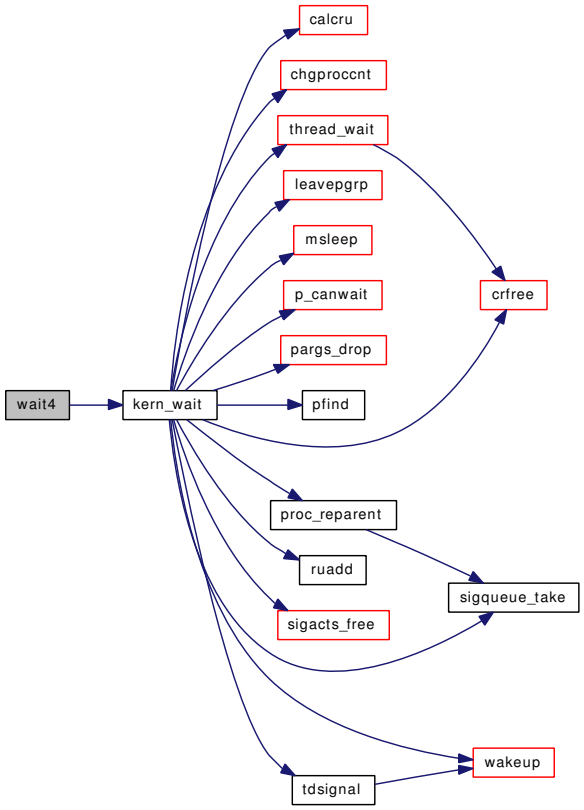


9.27.1.8 int wait4 (struct thread * *td*, struct wait_args * *uap*)

Definition at line 652 of file kern_exit.c.

References kern_wait().

Here is the call graph for this function:



9.27.2 Variable Documentation

9.27.2.1 void(*) nlminfo_release_p(struct proc *p)

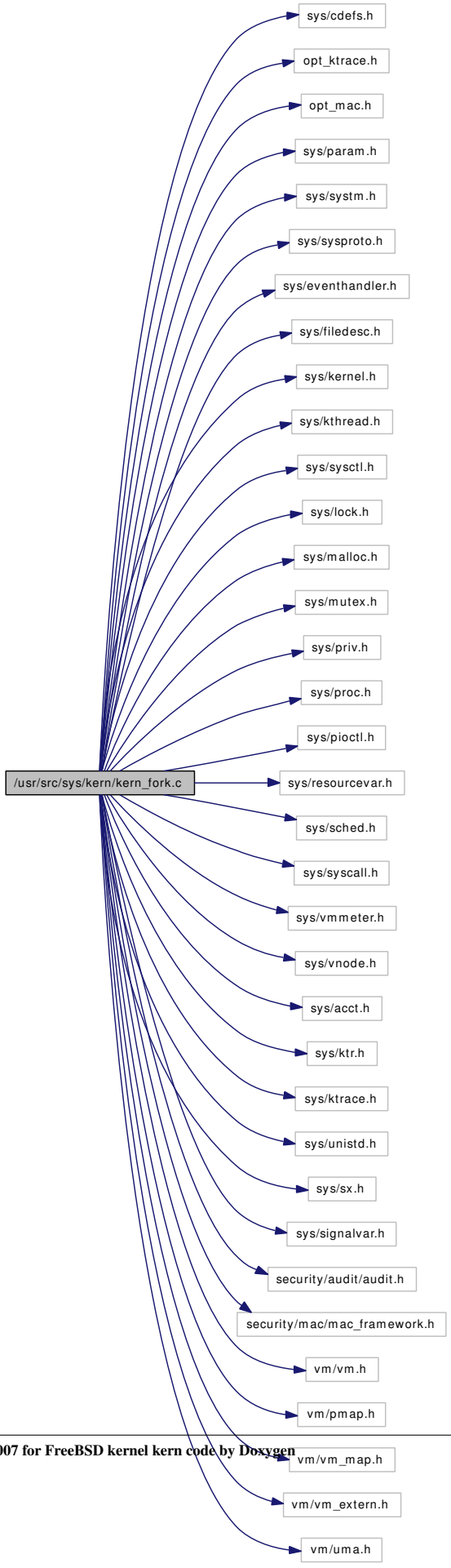
Definition at line 89 of file kern_exit.c.

Referenced by exit1().

9.28 /usr/src/sys/kern/kern_fork.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ktrace.h"
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/sysproto.h>
#include <sys/eventhandler.h>
#include <sys/filedesc.h>
#include <sys/kernel.h>
#include <sys/kthread.h>
#include <sys/sysctl.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/pioctl.h>
#include <sys/resourcevar.h>
#include <sys/sched.h>
#include <sys/syscall.h>
#include <sys/vmmeter.h>
#include <sys/vnode.h>
#include <sys/acct.h>
#include <sys/ktr.h>
#include <sys/ktrace.h>
#include <sys/unistd.h>
#include <sys/sx.h>
#include <sys/signalvar.h>
#include <security/audit/audit.h>
#include <security/mac/mac_framework.h>
#include <vm/vm.h>
#include <vm/pmap.h>
#include <vm/vm_map.h>
#include <vm/vm_extern.h>
#include <vm/uma.h>
```

Include dependency graph for kern_fork.c:


/usr/src/sys/kern/kern_fork.c
sys/cdefs.h
opt_ktrace.h
opt_mac.h
sys/param.h
sys/systm.h
sys/sysproto.h
sys/eventhandler.h
sys/filedesc.h
sys/kernel.h
sys/kthread.h
sys/sysctl.h
sys/lock.h
sys/malloc.h
sys/mutex.h
sys/priv.h
sys/proc.h
sys/pioctl.h
sys/resourcevar.h
sys/sched.h
sys/syscall.h
sys/vmmeter.h
sys/vnode.h
sys/acct.h
sys/ktr.h
sys/ktrace.h
sys/unistd.h
sys/sx.h
sys/signalvar.h
security/audit/audit.h
security/mac/mac_framework.h
vm/vm.h
vm/pmap.h
vm/vm_map.h
vm/vm_extern.h
vm/uma.h

Data Structures

- struct fork_args

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_fork.c,v 1.266 2007/01/23 08:46:50 jeff Exp $")
- int fork (struct thread *td, struct fork_args *uap)
- int vfork (struct thread *td, struct vfork_args *uap)
- int rfork (struct thread *td, struct rfork_args *uap)
- SYSCTL_INT (_kern, OID_AUTO, lastpid, CTLFLAG_RD,&lastpid, 0,"Last used PID")
- static int sysctl_kern_randompid (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern, OID_AUTO, randompid, CTLTYPE_INT|CTLFLAG_RW, 0, 0, sysctl_kern_randompid,"I","Random PID modulus")
- int fork1 (struct thread *td, int flags, int pages, struct proc **procp)
- void fork_exit (void *callout, void *arg, struct trapframe *frame)
- void fork_return (struct thread *td, struct trapframe *frame)

Variables

- static int forksleep
- int nprocs = 1
- int lastpid = 0
- static int randompid = 0

9.28.1 Function Documentation

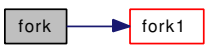
9.28.1.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_fork. *c*, v 1.266 2007/01/23 08:46:50 jeff Exp $")

9.28.1.2 int fork (struct thread * *td*, struct fork_args * *uap*)

Definition at line 92 of file kern_fork.c.

References fork1().

Here is the call graph for this function:



9.28.1.3 int fork1 (struct thread * *td*, int *flags*, int *pages*, struct proc ** *procp*)

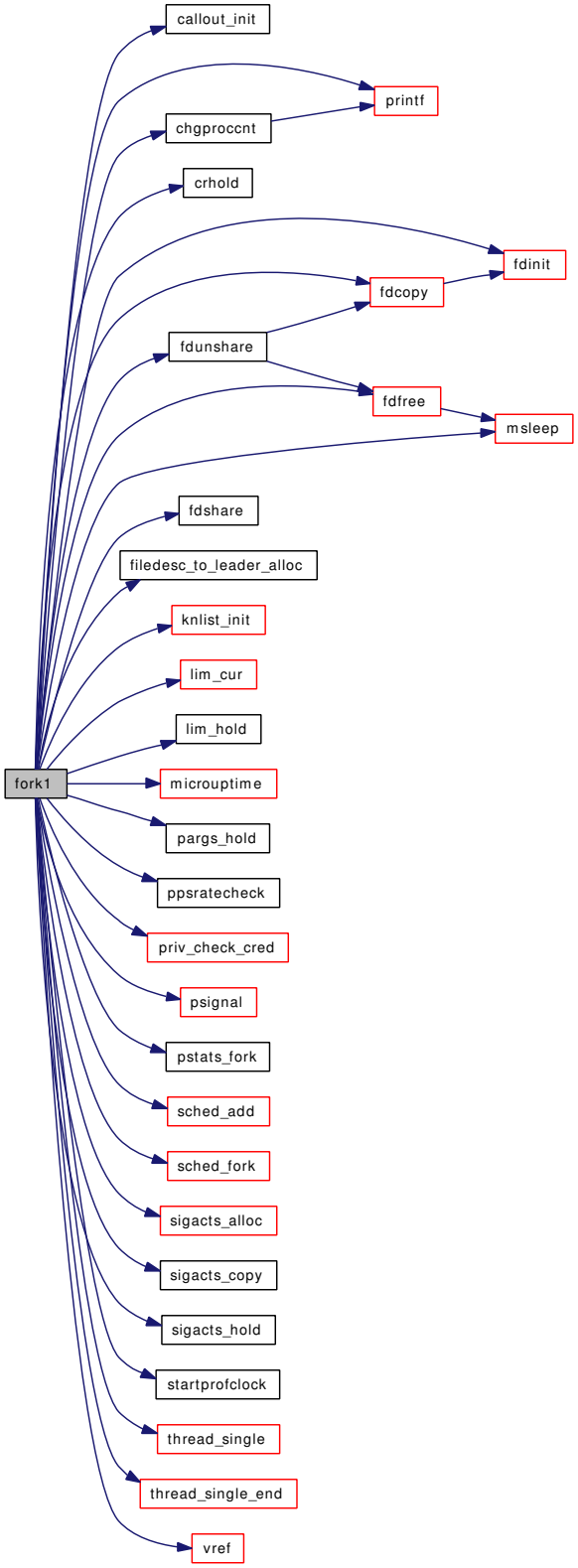
Definition at line 194 of file kern_fork.c.

References allproc, allproc_lock, callout_init(), chgproccnt(), crhold(), fdcopy(), fdfree(), fdinit(), fdshare(), fdunshare(), filedesc_to_leader_alloc(), forksleep, hz, initproc, knlist_init(), lastpid, lim_cur(), lim_hold(), maxproc, microuptime(), msleep(), nprocs, pargs_hold(), ppeers_lock, ppsratecheck(), printf(), priv_check_cred(), proc0, proc_zone, proctree_lock, psignal(), pstats_fork(), randompid, sched_add(),

sched_fork(), sched_lock, sigacts_alloc(), sigacts_copy(), sigacts_hold(), startprofclock(), thread_single(), thread_single_end(), vref(), and zombproc.

Referenced by create_init(), fork(), kthread_create(), rfork(), and vfork().

Here is the call graph for this function:

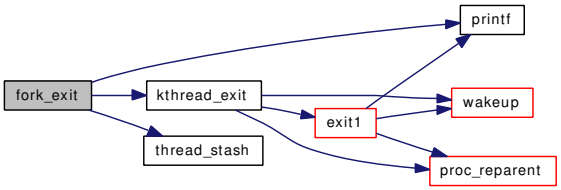

callout_init
printf
chgproccnt
crhold
fdinit
fdcopy
fdunshare
fdfree
msleep
fdshare
filedesc_to_leader_alloc
knlist_init
lim_cur
lim_hold
fork1
microuptime
pargs_hold
ppsratecheck
priv_check_cred
psignal
pstats_fork
sched_add
sched_fork
sigacts_alloc
sigacts_copy
sigacts_hold
startprofclock
thread_single
thread_single_end
vref

9.28.1.4 void fork_exit (void * *callout*, void * *arg*, struct trapframe * *frame*)

Definition at line 775 of file kern_fork.c.

References callout, Giant, kthread_exit(), printf(), sched_lock, td, and thread_stash().

Here is the call graph for this function:

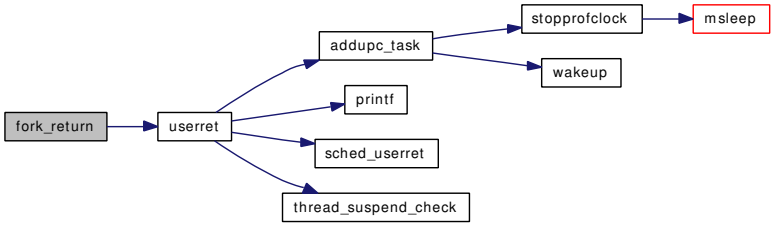


9.28.1.5 void fork_return (struct thread * *td*, struct trapframe * *frame*)

Definition at line 839 of file kern_fork.c.

References Giant, and userret().

Here is the call graph for this function:



9.28.1.6 int rfork (struct thread * *td*, struct rfork_args * *uap*)

Definition at line 131 of file kern_fork.c.

References fork1().

Here is the call graph for this function:

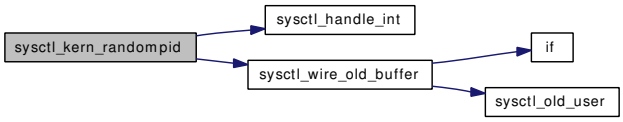
9.28.1.7 SYSCTL_INT (_kern, OID_AUTO, lastpid, CTLFLAG_RD, & *lastpid*, 0, "Last used PID")

9.28.1.8 static int sysctl_kern_randompid (SYSCTL_HANDLER_ARGS) [static]

Definition at line 167 of file kern_fork.c.

References allproc_lock, randompid, sysctl_handle_int(), and sysctl_wire_old_buffer().

Here is the call graph for this function:



9.28.1.9 SYSCTL_PROC (_kern, OID_AUTO, randompid, CTLTYPE_INT| *CTLFLAG_RW*, 0, 0, sysctl_kern_randompid, "I", "Random PID modulus")

9.28.1.10 int vfork (struct thread * *td*, struct vfork_args * *uap*)

Definition at line 112 of file kern_fork.c.

References fork1().

Here is the call graph for this function:

9.28.2 Variable Documentation

9.28.2.1 int forksleep [static]

Definition at line 85 of file kern_fork.c.

Referenced by fork1().

9.28.2.2 int lastpid = 0

Definition at line 152 of file kern_fork.c.

Referenced by fork1().

9.28.2.3 int nprocs = 1

Definition at line 151 of file kern_fork.c.

Referenced by fork1(), and kern_wait().

9.28.2.4 int randompid = 0 [static]

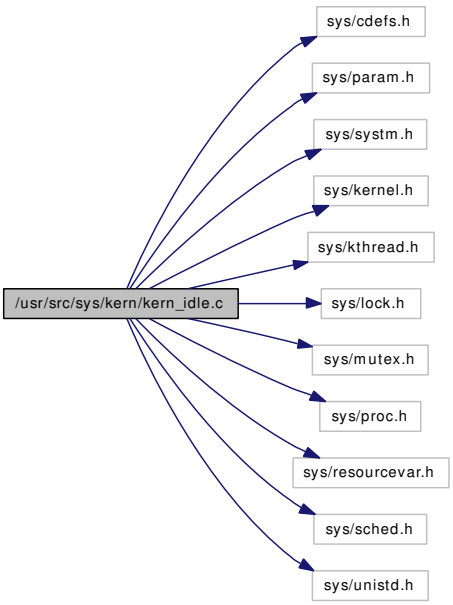
Definition at line 164 of file kern_fork.c.

Referenced by fork1(), and sysctl_kern_randompid().

9.29 /usr/src/sys/kern/kern_idle.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/kthread.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/resourcevar.h>
#include <sys/sched.h>
#include <sys/unistd.h>
```

Include dependency graph for kern_idle.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_idle.c,v 1.47 2007/01/23 08:46:50 jeff Exp $")
- static void idle_setup (void *dummy)

9.29.1 Function Documentation

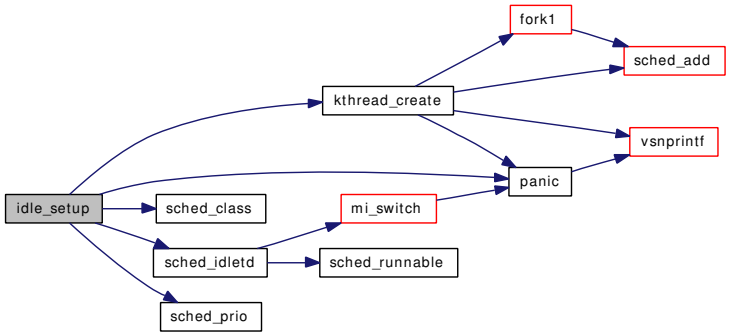
9.29.1.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_idle. *c*, v 1.47 2007/01/23 08:46:50 jeff Exp $")

9.29.1.2 static void idle_setup (void * *dummy*) [static]

Definition at line 52 of file kern_idle.c.

References cpuhead, kthread_create(), panic(), sched_class(), sched_idletd(), sched_lock, sched_prio(), and td.

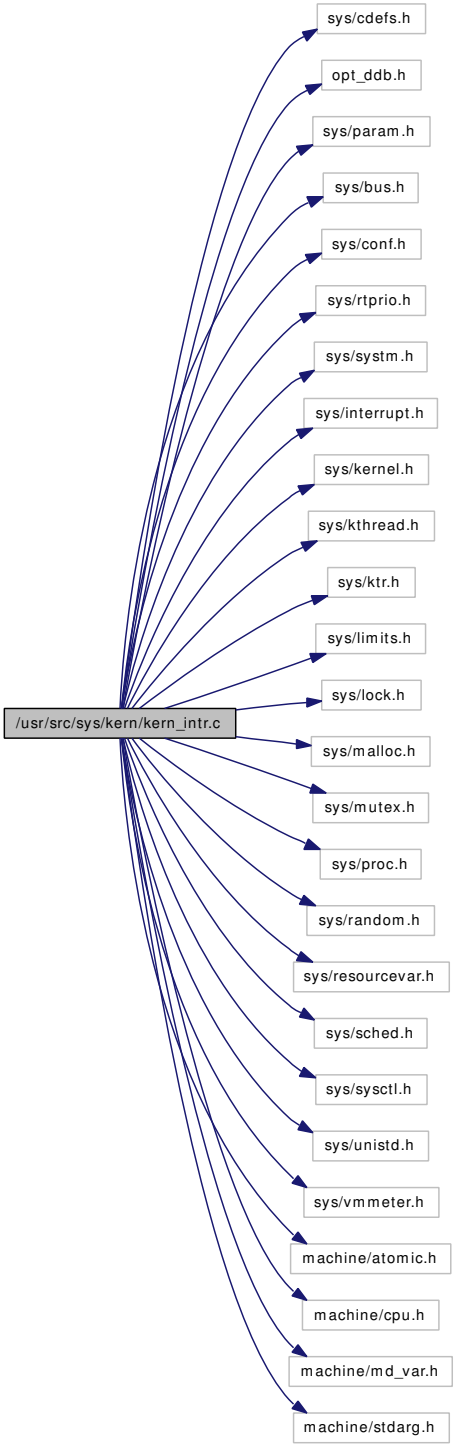
Here is the call graph for this function:



9.30 /usr/src/sys/kern/kern_intr.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ddb.h"
#include <sys/param.h>
#include <sys/bus.h>
#include <sys/conf.h>
#include <sys/rtprio.h>
#include <sys/systm.h>
#include <sys/interrupt.h>
#include <sys/kernel.h>
#include <sys/kthread.h>
#include <sys/ktr.h>
#include <sys/limits.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/random.h>
#include <sys/resourcevar.h>
#include <sys/sched.h>
#include <sys/sysctl.h>
#include <sys/unistd.h>
#include <sys/vmmeter.h>
#include <machine/atomic.h>
#include <machine/cpu.h>
#include <machine/md_var.h>
#include <machine/stdarg.h>
```

Include dependency graph for kern_intr.c:



Data Structures

- struct intr_thread
- struct intr_entropy

Defines

- #define IT_DEAD 0x000001

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_intr.c,v 1.140 2007/02/23 12:19:01 piso Exp $")
- static MALLOC_DEFINE (M_ITHREAD,"ithread","Interrupt Threads")
- TUNABLE_INT ("hw.intr_storm_threshold",&intr_storm_threshold)
- SYSCTL_INT (_hw, OID_AUTO, intr_storm_threshold, CTLFLAG_RW,&intr_storm_threshold, 0,"Number of consecutive interrupts before storm protection is enabled")
- static TAILQ_HEAD (intr_event)
- static void ithread_update (struct intr_thread *ithd)
- static void intr_event_update (struct intr_event *ie)
- int intr_event_create (struct intr_event **event, void *source, int flags, void(*enable)(void *), const char *fmt,...)
- int intr_event_destroy (struct intr_event *ie)
- static struct intr_thread * ithread_create (const char *name)
- static void ithread_destroy (struct intr_thread *ithread)
- int intr_event_add_handler (struct intr_event *ie, const char *name, driver_filter_t filter, driver_intr_t handler, void *arg, u_char pri, enum intr_type flags, void **cookiep)
- void * intr_handler_source (void *cookie)
- int intr_event_remove_handler (void *cookie)
- int intr_event_schedule_thread (struct intr_event *ie)
- int swi_add (struct intr_event **eventp, const char *name, driver_intr_t handler, void *arg, int pri, enum intr_type flags, void **cookiep)
- void swi_sched (void *cookie, int flags)
- int swi_remove (void *cookie)
- static void ithread_execute_handlers (struct proc *p, struct intr_event *ie)
- static void ithread_loop (void *arg)
- static void start_softintr (void *dummy)
- static int sysctl_intrnames (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_hw, OID_AUTO, intrnames, CTLTYPE_OPAQUE|CTLFLAG_RD, NULL, 0, sysctl_intrnames,"","Interrupt Names")
- static int sysctl_intrcnt (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_hw, OID_AUTO, intrcnt, CTLTYPE_OPAQUE|CTLFLAG_RD, NULL, 0, sysctl_intrcnt,"","Interrupt Counts")

Variables

- intr_event * clk_intr_event
- intr_event * tty_intr_event
- void * softclock_ih
- void * vm_ih
- static int intr_storm_threshold = 500

9.30.1 Define Documentation

9.30.1.1 #define IT_DEAD 0x000001

Definition at line 72 of file kern_intr.c.

Referenced by ithread_destroy(), and ithread_loop().

9.30.2 Function Documentation

9.30.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_intr. *c*, v 1.140 2007/02/23 12:19:01 piso Exp $")

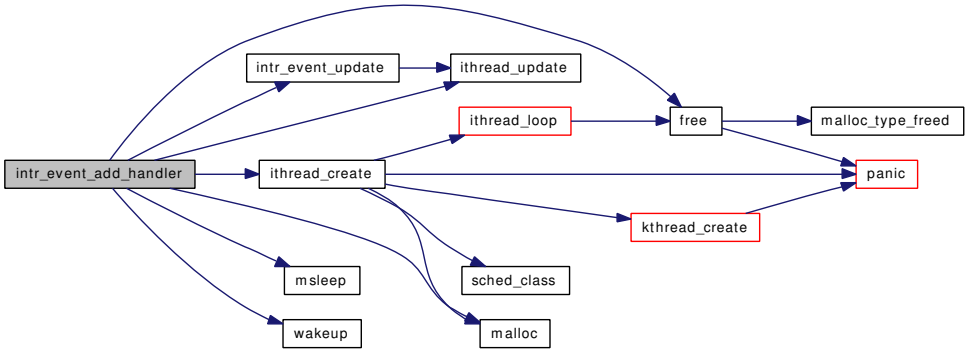
9.30.2.2 int intr_event_add_handler (struct intr_event * *ie*, const char * *name*, driver_filter_t *filter*, driver_intr_t *handler*, void * *arg*, u_char *pri*, enum intr_type *flags*, void ** *cookiep*)

Definition at line 326 of file kern_intr.c.

References free(), intr_event_update(), intr_thread::it_event, ithread_create(), ithread_update(), malloc(), msleep(), and wakeup().

Referenced by swi_add().

Here is the call graph for this function:



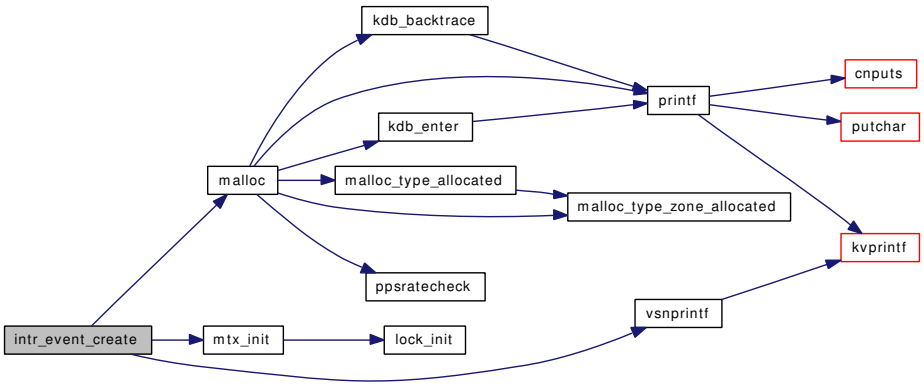
9.30.2.3 int intr_event_create (struct intr_event ** *event*, void * *source*, int *flags*, void(*)(void *) *enable*, const char * *fmt*, ...)

Definition at line 231 of file kern_intr.c.

References malloc(), mtx_init(), mtxpool_sleep, and vsnprintf().

Referenced by swi_add().

Here is the call graph for this function:

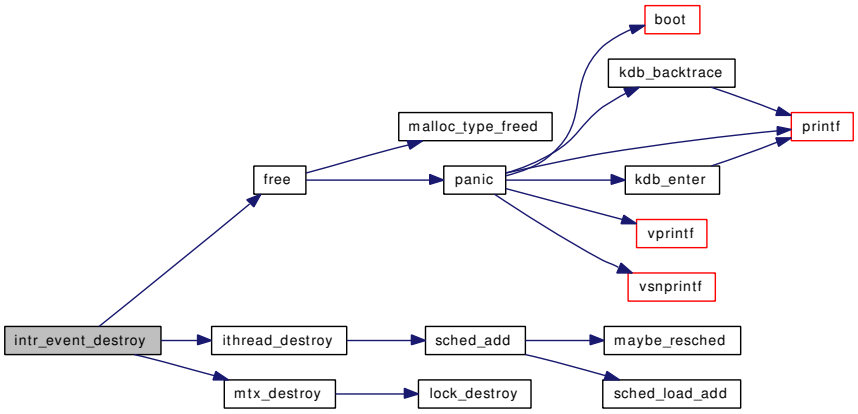


9.30.2.4 int intr_event_destroy (struct intr_event * *ie*)

Definition at line 261 of file kern_intr.c.

References free(), ithread_destroy(), mtx_destroy(), and mtxpool_sleep.

Here is the call graph for this function:



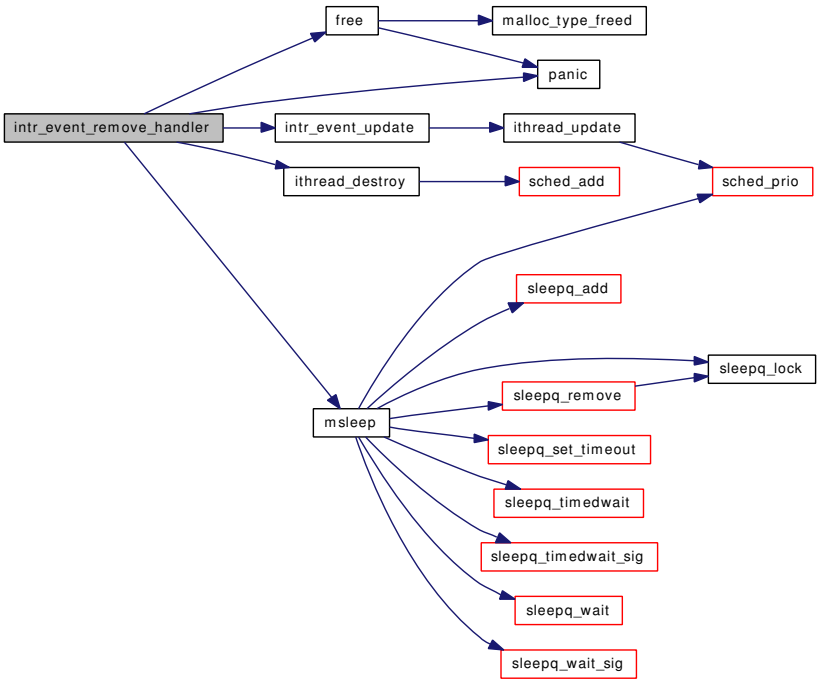
9.30.2.5 int intr_event_remove_handler (void * *cookie*)

Definition at line 419 of file kern_intr.c.

References free(), intr_event_update(), ithread_destroy(), msleep(), panic(), and sched_lock.

Referenced by swi_remove().

Here is the call graph for this function:



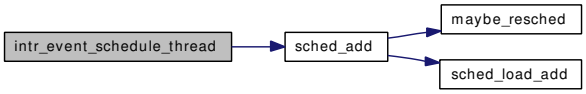
9.30.2.6 int intr_event_schedule_thread (struct intr_event * *ie*)

Definition at line 508 of file kern_intr.c.

References intr_entropy::event, intr_thread::it_need, intr_thread::it_thread, sched_add(), sched_lock, intr_entropy::td, and td.

Referenced by swi_sched().

Here is the call graph for this function:



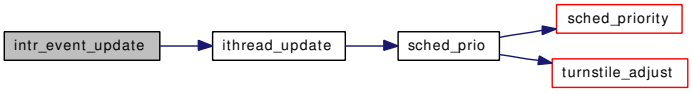
9.30.2.7 static void intr_event_update (struct intr_event * *ie*) [static]

Definition at line 175 of file kern_intr.c.

References ithread_update().

Referenced by intr_event_add_handler(), and intr_event_remove_handler().

Here is the call graph for this function:



9.30.2.8 void* intr_handler_source (void * *cookie*)

Definition at line 403 of file kern_intr.c.

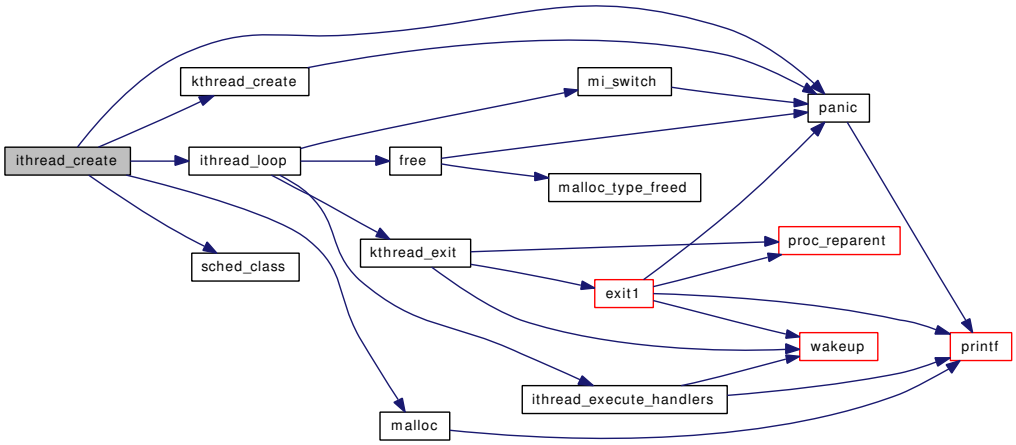
9.30.2.9 static struct intr_thread* ithread_create (const char * *name*) [static]

Definition at line 285 of file kern_intr.c.

References intr_thread::it_thread, ithread_loop(), kthread_create(), malloc(), panic(), sched_class(), sched_lock, and td.

Referenced by intr_event_add_handler().

Here is the call graph for this function:



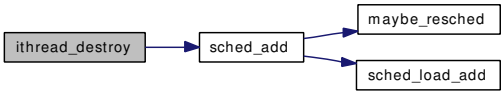
9.30.2.10 static void ithread_destroy (struct intr_thread * *ithread*) [static]

Definition at line 310 of file kern_intr.c.

References IT_DEAD, intr_thread::it_event, intr_thread::it_flags, intr_thread::it_thread, sched_add(), sched_lock, and td.

Referenced by intr_event_destroy(), and intr_event_remove_handler().

Here is the call graph for this function:



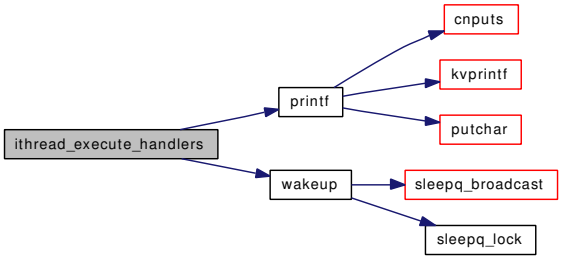
9.30.2.11 static void ithread_execute_handlers (struct proc * *p*, struct intr_event * *ie*) [static]

Definition at line 637 of file kern_intr.c.

References Giant, intr_storm_threshold, printf(), and wakeup().

Referenced by ithread_loop().

Here is the call graph for this function:



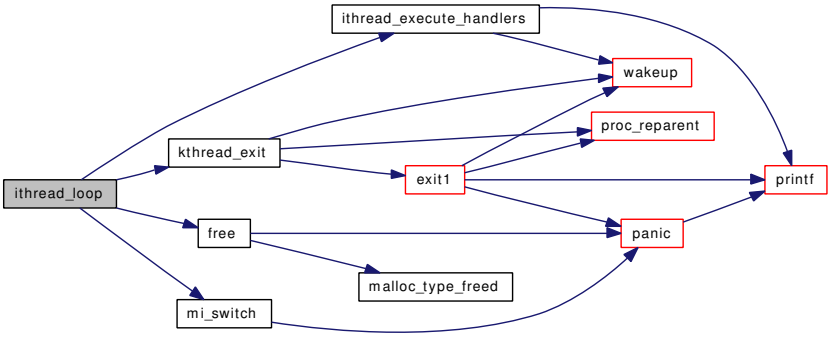
9.30.2.12 static void ithread_loop (void * *arg*) [static]

Definition at line 718 of file kern_intr.c.

References free(), Giant, IT_DEAD, intr_thread::it_event, intr_thread::it_flags, intr_thread::it_need, intr_thread::it_thread, ithread_execute_handlers(), kthread_exit(), mi_switch(), sched_lock, and td.

Referenced by ithread_create().

Here is the call graph for this function:



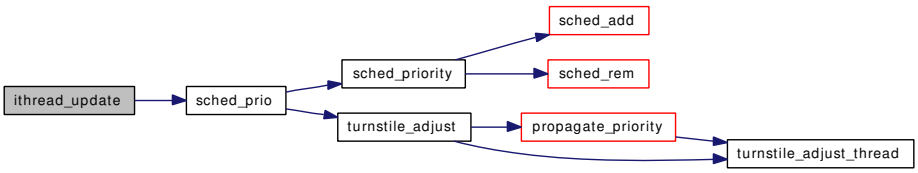
9.30.2.13 static void ithread_update (struct intr_thread * *ithd*) [static]

Definition at line 148 of file kern_intr.c.

References intr_thread::it_event, intr_thread::it_thread, sched_lock, sched_prio(), and td.

Referenced by intr_event_add_handler(), and intr_event_update().

Here is the call graph for this function:



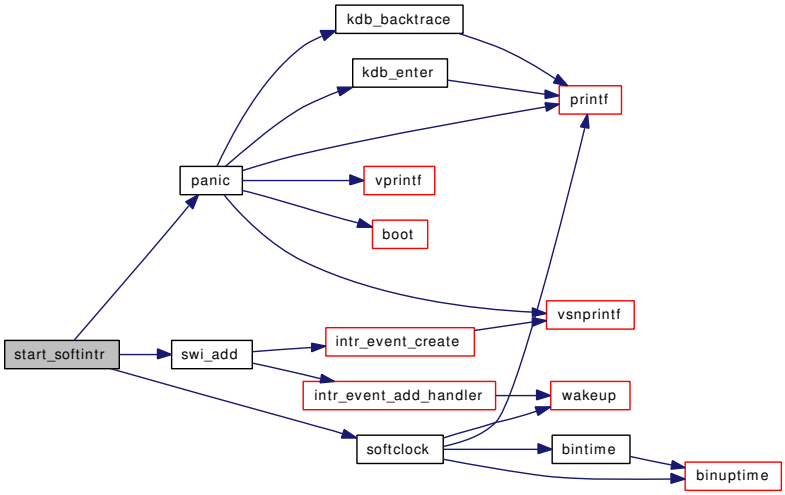
9.30.2.14 static MALLOC_DEFINE (M_ITHREAD, "ithread", "Interrupt Threads") [static]

9.30.2.15 static void start_softintr (void * *dummy*) [static]

Definition at line 938 of file kern_intr.c.

References clk_intr_event, panic(), softclock(), softclock_ih, swi_add(), and vm_ih.

Here is the call graph for this function:



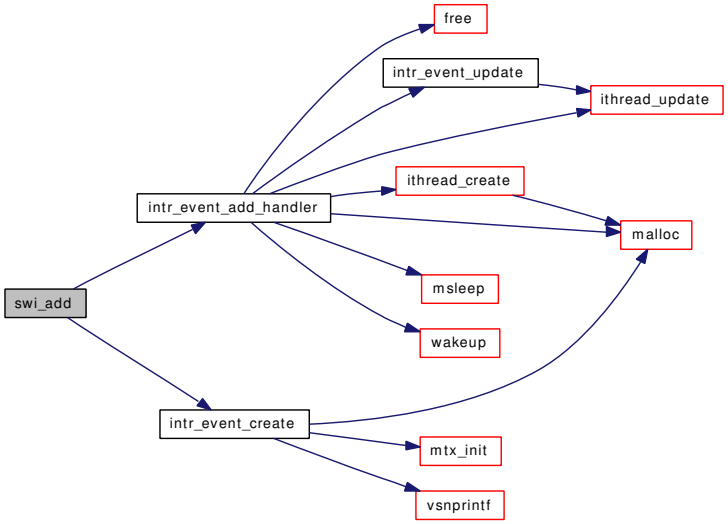
9.30.2.16 int swi_add (struct intr_event ** *eventp*, const char * *name*, driver_intr_t *handler*, void * *arg*, int *pri*, enum intr_type *flags*, void ** *cookiep*)

Definition at line 569 of file kern_intr.c.

References intr_event_add_handler(), and intr_event_create().

Referenced by start_softintr().

Here is the call graph for this function:

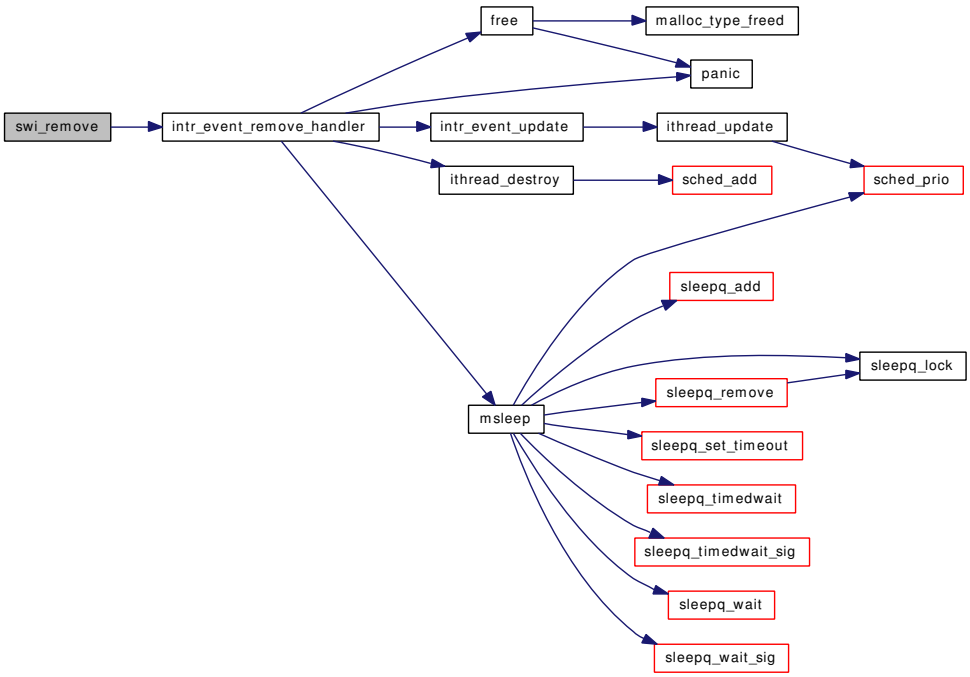


9.30.2.17 int swi_remove (void * *cookie*)

Definition at line 630 of file kern_intr.c.

References intr_event_remove_handler().

Here is the call graph for this function:



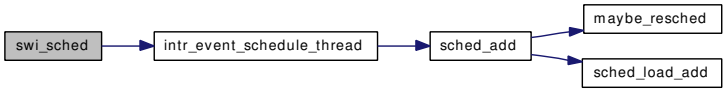
9.30.2.18 void swi_sched (void * *cookie*, int *flags*)

Definition at line 600 of file kern_intr.c.

References intr_event_schedule_thread().

Referenced by hardclock(), taskqueue_fast_enqueue(), taskqueue_swi_enqueue(), and taskqueue_swi_-giant_enqueue().

Here is the call graph for this function:



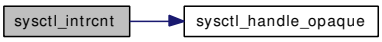
9.30.2.19 SYSCTL_INT (_hw, OID_AUTO, intr_storm_threshold, CTLFLAG_RW, & *intr_storm_threshold*, 0, "Number of consecutive interrupts before storm protection is enabled")

9.30.2.20 static int sysctl_intrcnt (SYSCTL_HANDLER_ARGS) [static]

Definition at line 974 of file kern_intr.c.

References sysctl_handle_opaque().

Here is the call graph for this function:



9.30.2.21 static int sysctl_intrnames (SYSCTL_HANDLER_ARGS) [static]

Definition at line 964 of file kern_intr.c.

References sysctl_handle_opaque().

Here is the call graph for this function:

9.30.2.22 SYSCTL_PROC (_hw, OID_AUTO, intrcnt, CTLTYPE_OPAQUE| *CTLFLAG_RD*, NULL, 0, sysctl_intrcnt, "", "Interrupt Counts")

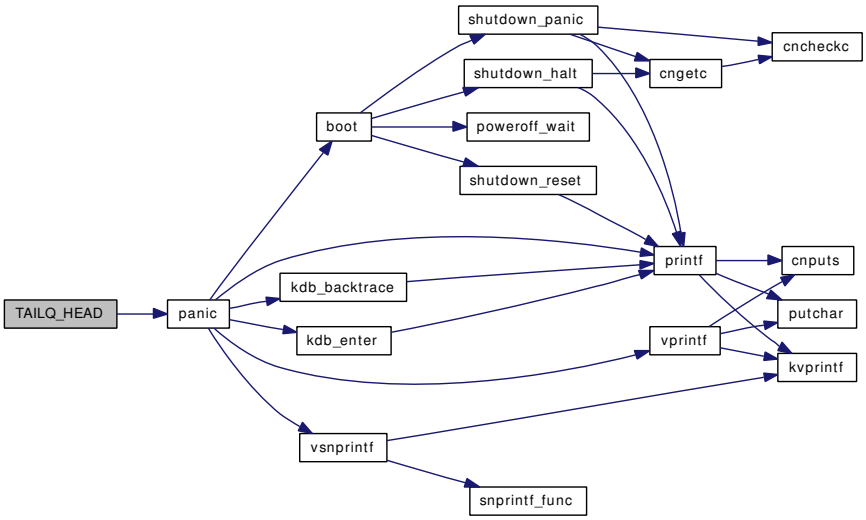
9.30.2.23 SYSCTL_PROC (_hw, OID_AUTO, intrnames, CTLTYPE_OPAQUE| *CTLFLAG_RD*, NULL, 0, sysctl_intrnames, "", "Interrupt Names")

9.30.2.24 static TAILQ_HEAD (intr_event) [static]

Definition at line 91 of file kern_intr.c.

References panic().

Here is the call graph for this function:



9.30.2.25 TUNABLE_INT ("hw.intr_storm_threshold", & *intr_storm_threshold*)

9.30.3 Variable Documentation

9.30.3.1 struct intr_event* clk_intr_event

Definition at line 79 of file kern_intr.c.

Referenced by start_softintr().

9.30.3.2 int intr_storm_threshold = 500 [static]

Definition at line 86 of file kern_intr.c.

Referenced by ithread_execute_handlers().

9.30.3.3 void* softclock_ih

Definition at line 81 of file kern_intr.c.

Referenced by hardclock(), and start_softintr().

9.30.3.4 struct intr_event* tty_intr_event

Definition at line 80 of file kern_intr.c.

9.30.3.5 void* vm_ih

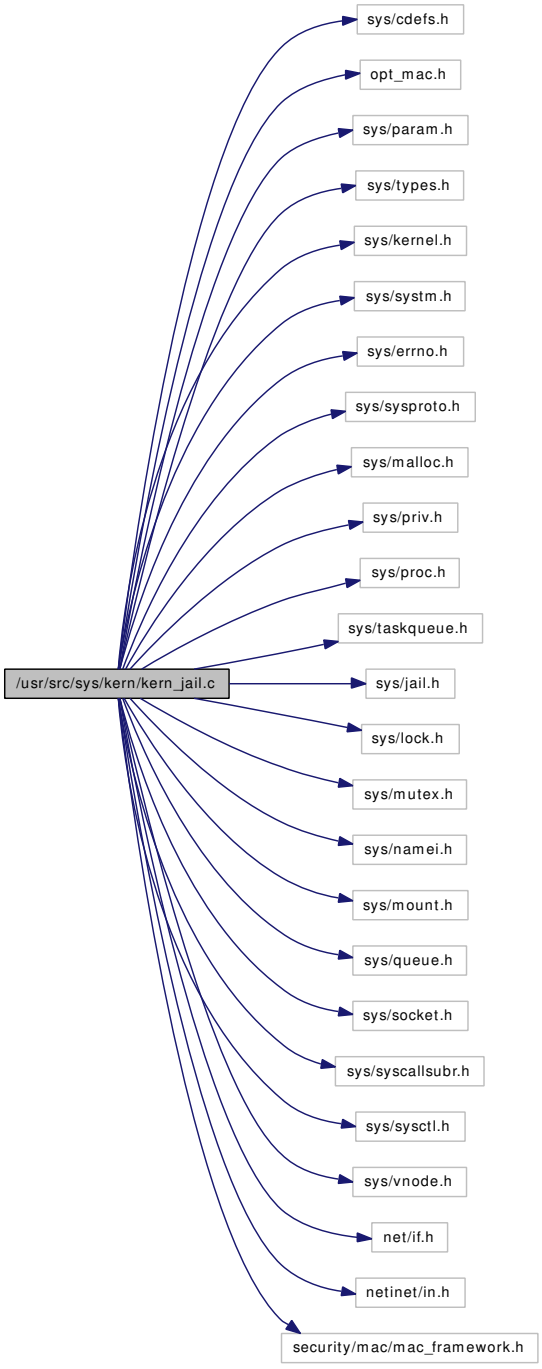
Definition at line 82 of file kern_intr.c.

Referenced by start_softintr().

9.31 /usr/src/sys/kern/kern_jail.c File Reference

```
#include <sys/cdefs.h>
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/types.h>
#include <sys/kernel.h>
#include <sys/systm.h>
#include <sys/errno.h>
#include <sys/sysproto.h>
#include <sys/malloc.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/taskqueue.h>
#include <sys/jail.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/namei.h>
#include <sys/mount.h>
#include <sys/queue.h>
#include <sys/socket.h>
#include <sys/syscallsubr.h>
#include <sys/sysctl.h>
#include <sys/vnode.h>
#include <net/if.h>
#include <netinet/in.h>
#include <security/mac/mac_framework.h>
```

Include dependency graph for kern_jail.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_jail.c,v 1.58 2007/02/20 00:12:52 rwatson Exp $")
- MALLOC_DEFINE (M_PRISON,"prison","Prison structures")
- SYSCTL_NODE (_security, OID_AUTO, jail, CTLFLAG_RW, 0,"Jail rules")
- SYSCTL_INT (_security_jail, OID_AUTO, set_hostname_allowed, CTLFLAG_RW,&jail_set_hostname_allowed, 0,"Processes in jail can set their hostnames")

- SYSCTL_INT (_security_jail, OID_AUTO, socket_unixiproute_only, CTLFLAG_RW,&jail_socket_unixiproute_only, 0,"Processes in jail are limited to creating UNIX/IPv4/route sockets only")
- SYSCTL_INT (_security_jail, OID_AUTO, sysvipc_allowed, CTLFLAG_RW,&jail_sysvipc_allowed, 0,"Processes in jail can use System V IPC primitives")
- SYSCTL_INT (_security_jail, OID_AUTO, enforce_statfs, CTLFLAG_RW,&jail_enforce_statfs, 0,"Processes in jail cannot see all mounted file systems")
- SYSCTL_INT (_security_jail, OID_AUTO, allow_raw_sockets, CTLFLAG_RW,&jail_allow_raw_sockets, 0,"Prison root can create raw sockets")
- SYSCTL_INT (_security_jail, OID_AUTO, chflags_allowed, CTLFLAG_RW,&jail_chflags_allowed, 0,"Processes in jail can alter system file flags")
- static void init_prison (void *)
- static void prison_complete (void *context, int pending)
- static struct prison * prison_find (int)
- static int sysctl_jail_list (SYSCTL_HANDLER_ARGS)
- static void init_prison (void *data __unused)
- SYSINIT (prison, SI_SUB_INTRINSIC, SI_ORDER_ANY, init_prison, NULL)
- int jail (struct thread *td, struct jail_args *uap)
- int jail_attach (struct thread *td, struct jail_attach_args *uap)
- void prison_free (struct prison *pr)
- void prison_hold (struct prison *pr)
- u_int32_t prison_getip (struct ucred *cred)
- int prison_ip (struct ucred *cred, int flag, u_int32_t *ip)
- void prison_remote_ip (struct ucred *cred, int flag, u_int32_t *ip)
- int prison_if (struct ucred *cred, struct sockaddr *sa)
- int prison_check (struct ucred *cred1, struct ucred *cred2)
- int jailed (struct ucred *cred)
- void getcredhostname (struct ucred *cred, char *buf, size_t size)
- int prison_canseemount (struct ucred *cred, struct mount *mp)
- void prison_enforce_statfs (struct ucred *cred, struct mount *mp, struct statfs *sp)
- int prison_priv_check (struct ucred *cred, int priv)
- SYSCTL_OID (_security_jail, OID_AUTO, list, CTLTYPE_STRUCT|CTLFLAG_RD, NULL, 0, sysctl_jail_list,"S","List of active jails")
- static int sysctl_jail_jailed (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_security_jail, OID_AUTO, jailed, CTLTYPE_INT|CTLFLAG_RD, NULL, 0, sysctl_jail_jailed,"I","Process in jail?")

Variables

- int jail_set_hostname_allowed = 1
- int jail_socket_unixiproute_only = 1
- int jail_sysvipc_allowed = 0
- static int jail_enforce_statfs = 2
- int jail_allow_raw_sockets = 0
- int jail_chflags_allowed = 0
- prisonlist allprison
- mtx allprison_mtx
- int lastprid = 0
- int prisoncount = 0

9.31.1 Function Documentation

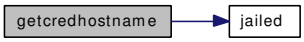
9.31.1.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_jail. *c*, v 1.58 2007/02/20 00:12:52 rwatson Exp $")

9.31.1.2 void getcredhostname (struct ucred * *cred*, char * *buf*, size_t *size*)

Definition at line 428 of file kern_jail.c.

References hostname, and jailed().

Here is the call graph for this function:

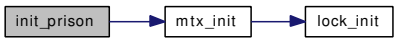


9.31.1.3 static void init_prison (void *data *__unused*) [static]

Definition at line 87 of file kern_jail.c.

References allprison, allprison_mtx, and mtx_init().

Here is the call graph for this function:



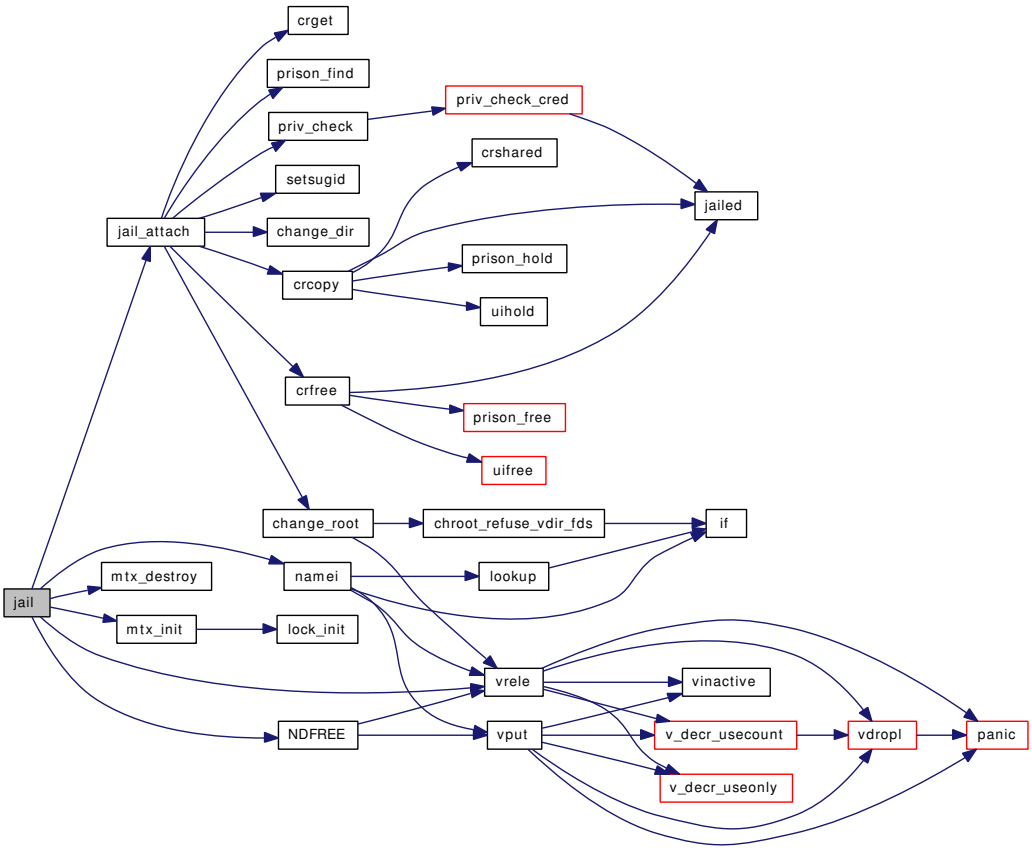
9.31.1.4 static void init_prison (void *) [static]

9.31.1.5 int jail (struct thread * *td*, struct jail_args * *uap*)

Definition at line 104 of file kern_jail.c.

References allprison, allprison_mtx, jail_attach(), lastprid, mtx_destroy(), mtx_init(), namei(), NDFREE(), pr, prisoncount, securelevel, and vrele().

Here is the call graph for this function:



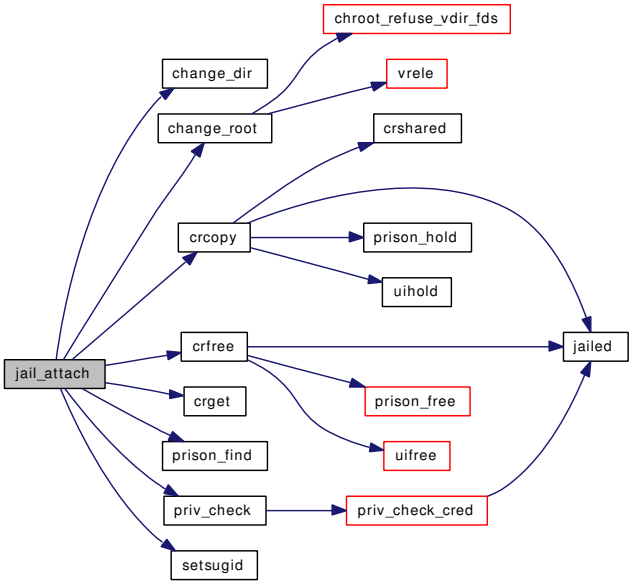
9.31.1.6 int jail_attach (struct thread * *td*, struct jail_attach_args * *uap*)

Definition at line 194 of file kern_jail.c.

References allprison_mtx, change_dir(), change_root(), crcopy(), crfree(), crget(), pr, prison_find(), priv_-check(), and setsugid().

Referenced by jail().

Here is the call graph for this function:



9.31.1.7 int jailed (struct ucred * *cred*)

Definition at line 418 of file kern_jail.c.

Referenced by crcopy(), crfree(), fill_kinfo_proc_only(), getcredhostname(), kern_msgctl(), kern_msgrcv(), kern_msgsnd(), kern_semctl(), kern_shmat(), kern_shmctl(), msgget(), msgsys(), prison_canseemount(), prison_check(), prison_enforce_statfs(), prison_ip(), prison_priv_check(), prison_remote_ip(), priv_check_cred(), quotactl(), semget(), semop(), semsys(), shmdt(), shmget(), shmsys(), socreate(), sysctl_jail_jailed(), sysctl_jail_list(), unmount(), and vfs_domount().

9.31.1.8 MALLOC_DEFINE (M_PRISON, "prison", "Prison structures")

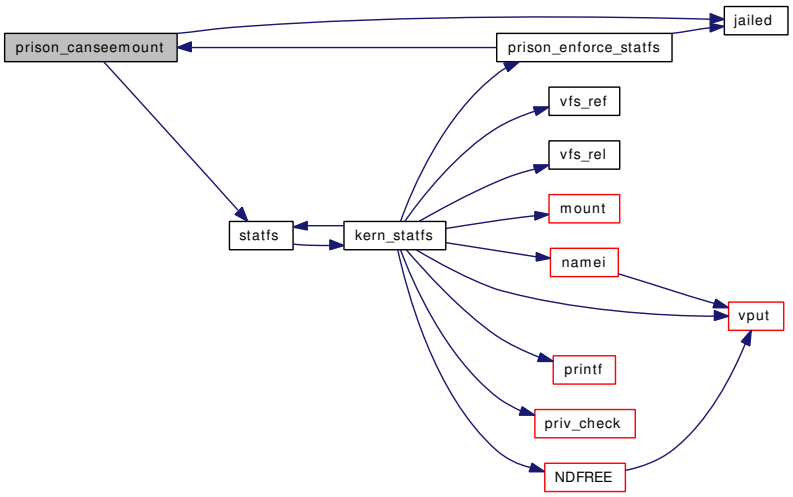
9.31.1.9 int prison_canseemount (struct ucred * *cred*, struct mount * *mp*)

Definition at line 447 of file kern_jail.c.

References jail_enforce_statfs, jailed(), pr, and statfs().

Referenced by kern_fhstatfs(), kern_getfsstat(), and prison_enforce_statfs().

Here is the call graph for this function:



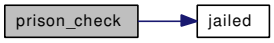
9.31.1.10 int prison_check (struct ucred * *cred1*, struct ucred * *cred2*)

Definition at line 401 of file kern_jail.c.

References jailed().

Referenced by cr_cansee(), cr_canseesocket(), cr_cansignal(), p_candebug(), p_cansched(), and p_-canwait().

Here is the call graph for this function:



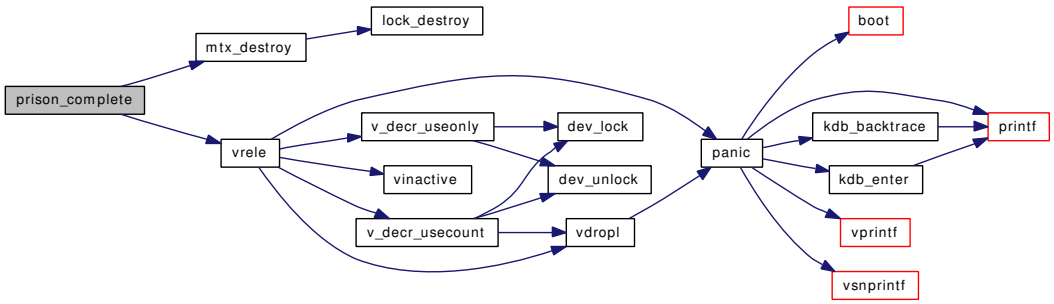
9.31.1.11 static void prison_complete (void * *context*, int *pending*) [static]

Definition at line 295 of file kern_jail.c.

References mtx_destroy(), pr, and vrele().

Referenced by prison_free().

Here is the call graph for this function:



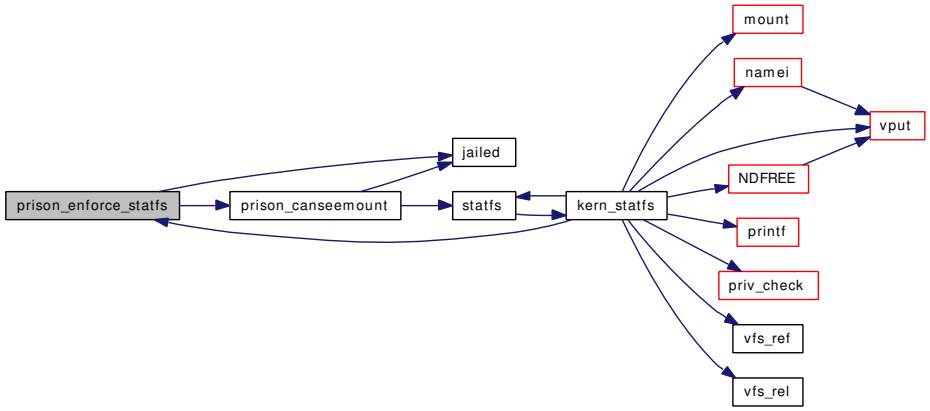
9.31.1.12 void prison_enforce_statfs (struct ucred * *cred*, struct mount * *mp*, struct statfs * *sp*)

Definition at line 482 of file kern_jail.c.

References jail_enforce_statfs, jailed(), pr, and prison_canseemount().

Referenced by kern_fstatfs(), and kern_statfs().

Here is the call graph for this function:



9.31.1.13 static struct prison * prison_find (int) [static]

Definition at line 259 of file kern_jail.c.

References allprison, allprison_mtx, and pr.

Referenced by jail_attach().

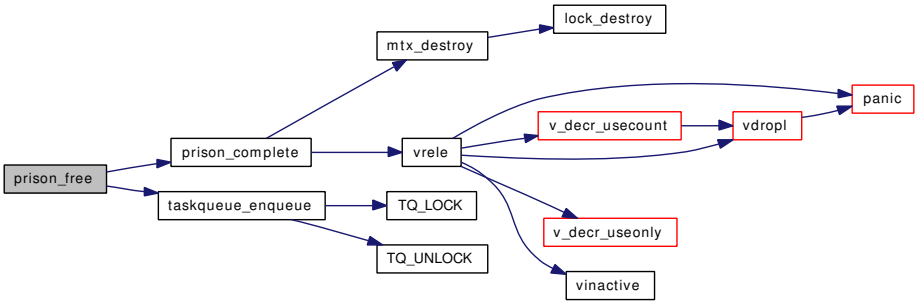
9.31.1.14 void prison_free (struct prison * *pr*)

Definition at line 274 of file kern_jail.c.

References allprison_mtx, prison_complete(), prisoncount, and taskqueue_enqueue().

Referenced by crfree().

Here is the call graph for this function:



9.31.1.15 u_int32_t prison_getip (struct ucred * *cred*)

Definition at line 322 of file kern_jail.c.

9.31.1.16 void prison_hold (struct prison * *pr*)

Definition at line 313 of file kern_jail.c.

Referenced by crcopy().

9.31.1.17 int prison_if (struct ucred * *cred*, struct sockaddr * *sa*)

Definition at line 380 of file kern_jail.c.

References jail_socket_unixiproute_only.

9.31.1.18 int prison_ip (struct ucred * *cred*, int *flag*, u_int32_t * *ip*)

Definition at line 329 of file kern_jail.c.

References jailed().

Here is the call graph for this function:

9.31.1.19 int prison_priv_check (struct ucred * *cred*, int *priv*)

Definition at line 532 of file kern_jail.c.

References jail_allow_raw_sockets, jail_chflags_allowed, and jailed().

Referenced by priv_check_cred().

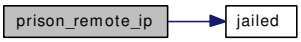
Here is the call graph for this function:

9.31.1.20 void prison_remote_ip (struct ucred * *cred*, int *flag*, u_int32_t * *ip*)

Definition at line 359 of file kern_jail.c.

References jailed().

Here is the call graph for this function:



9.31.1.21 SYSCTL_INT (_security_jail, OID_AUTO, chflags_allowed, CTLFLAG_RW, & *jail_chflags_allowed*, 0, "Processes in jail can alter system file flags")

9.31.1.22 SYSCTL_INT (_security_jail, OID_AUTO, allow_raw_sockets, CTLFLAG_RW, & *jail_allow_raw_sockets*, 0, "Prison root can create raw sockets")

9.31.1.23 SYSCTL_INT (_security_jail, OID_AUTO, enforce_statfs, CTLFLAG_RW, & *jail_enforce_statfs*, 0, "Processes in jail cannot see all mounted file systems")

9.31.1.24 SYSCTL_INT (_security_jail, OID_AUTO, sysvipc_allowed, CTLFLAG_RW, & *jail_sysvipc_allowed*, 0, "Processes in jail can use System V IPC primitives")

9.31.1.25 SYSCTL_INT (_security_jail, OID_AUTO, socket_unixiproute_only, CTLFLAG_RW, & *jail_socket_unixiproute_only*, 0, "Processes in jail are limited to creating UNIX/IPv4/route sockets only")

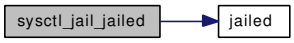
9.31.1.26 SYSCTL_INT (_security_jail, OID_AUTO, set_hostname_allowed, CTLFLAG_RW, & *jail_set_hostname_allowed*, 0, "Processes in jail can set their hostnames")

9.31.1.27 static int sysctl_jail_jailed (SYSCTL_HANDLER_ARGS) [static]

Definition at line 739 of file kern_jail.c.

References jailed().

Here is the call graph for this function:

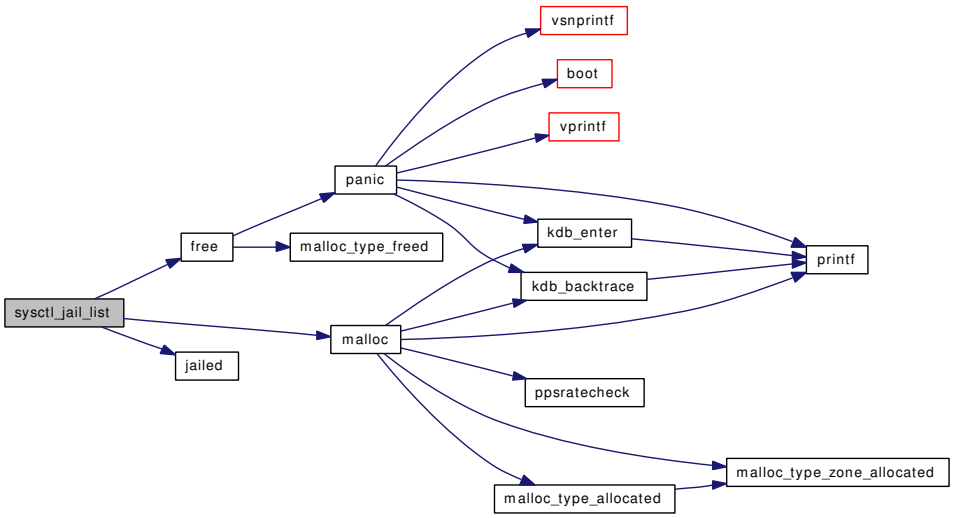


9.31.1.28 static int sysctl_jail_list (SYSCTL_HANDLER_ARGS) [static]

Definition at line 692 of file kern_jail.c.

References allprison, allprison_mtx, free(), jailed(), malloc(), pr, and prisoncount.

Here is the call graph for this function:



9.31.1.29 SYSCTL_NODE (_security, OID_AUTO, jail, CTLFLAG_RW, 0, "Jail rules")

9.31.1.30 SYSCTL_OID (_security_jail, OID_AUTO, list, CTLTYPE_STRUCT| *CTLFLAG_RD*, NULL, 0, sysctl_jail_list, "S", "List of active jails")

9.31.1.31 SYSCTL_PROC (_security_jail, OID_AUTO, jailed, CTLTYPE_INT| *CTLFLAG_RD*, NULL, 0, sysctl_jail_jailed, "I", "Process in jail?")

9.31.1.32 SYSINIT (prison, SI_SUB_INTRINSIC, SI_ORDER_ANY, init_prison, NULL)

9.31.2 Variable Documentation

9.31.2.1 struct prisonlist allprison

Definition at line 76 of file kern_jail.c.

Referenced by init_prison(), jail(), prison_find(), and sysctl_jail_list().

9.31.2.2 struct mtx allprison_mtx

Definition at line 77 of file kern_jail.c.

Referenced by init_prison(), jail(), jail_attach(), prison_find(), prison_free(), and sysctl_jail_list().

9.31.2.3 int jail_allow_raw_sockets = 0

Definition at line 65 of file kern_jail.c.

Referenced by prison_priv_check().

9.31.2.4 int jail_chflags_allowed = 0

Definition at line 70 of file kern_jail.c.

Referenced by prison_priv_check().

9.31.2.5 int jail_enforce_statfs = 2 [static]

Definition at line 60 of file kern_jail.c.

Referenced by prison_canseemount(), and prison_enforce_statfs().

9.31.2.6 int jail_set_hostname_allowed = 1

Definition at line 45 of file kern_jail.c.

Referenced by sysctl_hostname().

9.31.2.7 int jail_socket_unixiproute_only = 1

Definition at line 50 of file kern_jail.c.

Referenced by prison_if(), and socreate().

9.31.2.8 int jail_sysvipc_allowed = 0

Definition at line 55 of file kern_jail.c.

Referenced by kern_msgctl(), kern_msgrcv(), kern_msgsnd(), kern_semctl(), kern_shmat(), kern_shmctl(), msgget(), msgsys(), semget(), semop(), semsys(), shmdt(), shmget(), and shmsys().

9.31.2.9 int lastprid = 0

Definition at line 78 of file kern_jail.c.

Referenced by jail().

9.31.2.10 int prisoncount = 0

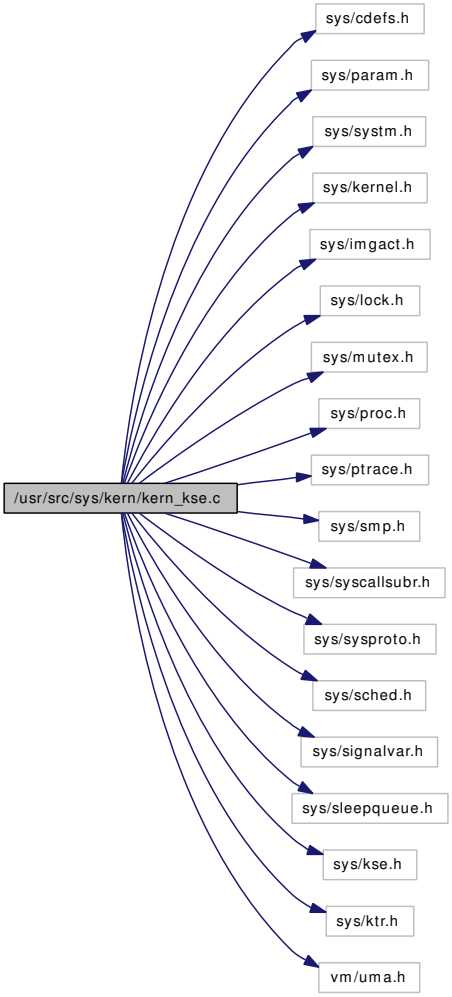
Definition at line 79 of file kern_jail.c.

Referenced by jail(), prison_free(), and sysctl_jail_list().

9.32 /usr/src/sys/kern/kern_kse.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/imgact.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/ptrace.h>
#include <sys/smp.h>
#include <sys/syscallsubr.h>
#include <sys/sysproto.h>
#include <sys/sched.h>
#include <sys/signalvar.h>
#include <sys/sleepqueue.h>
#include <sys/kse.h>
#include <sys/ktr.h>
#include <vm/uma.h>
```

Include dependency graph for kern_kse.c:



Data Structures

- struct kse_switchin_args

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_kse.c,v 1.227 2007/01/23 08:46:50 jeff Exp $")
- int kse_switchin (struct thread *td, struct kse_switchin_args *uap)
- int kse_thr_interrupt (struct thread *td, struct kse_thr_interrupt_args *uap)
- int kse_exit (struct thread *td, struct kse_exit_args *uap)
- int kse_release (struct thread *td, struct kse_release_args *uap)
- int kse_wakeup (struct thread *td, struct kse_wakeup_args *uap)
- int kse_create (struct thread *td, struct kse_create_args *uap)

9.32.1 Function Documentation

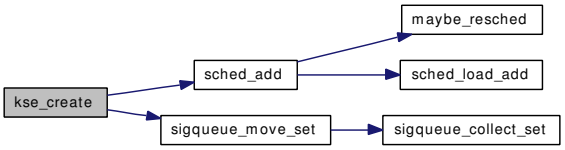
9.32.1.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_kse. *c*, v 1.227 2007/01/23 08:46:50 jeff Exp $")

9.32.1.2 int kse_create (struct thread * *td*, struct kse_create_args * *uap*)

Definition at line 539 of file kern_kse.c.

References mp_ncpus, sched_add(), sched_lock, sigqueue_move_set(), and tick.

Here is the call graph for this function:

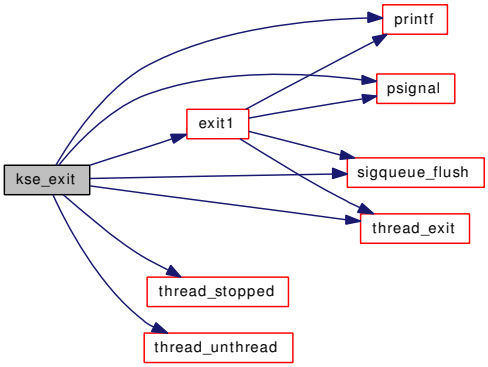


9.32.1.3 int kse_exit (struct thread * *td*, struct kse_exit_args * *uap*)

Definition at line 301 of file kern_kse.c.

References exit1(), printf(), psignal(), sched_lock, sigqueue_flush(), thread_exit(), thread_stopped(), and thread_unthread().

Here is the call graph for this function:

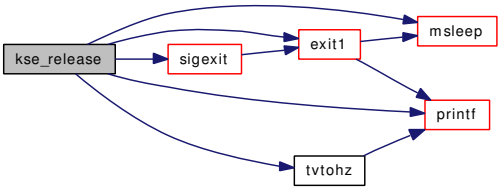


9.32.1.4 int kse_release (struct thread * *td*, struct kse_release_args * *uap*)

Definition at line 396 of file kern_kse.c.

References exit1(), msleep(), printf(), sched_lock, sigexit(), and tvtohz().

Here is the call graph for this function:



9.32.1.5 int kse_switchin (struct thread * *td*, struct kse_switchin_args * *uap*)

Definition at line 128 of file kern_kse.c.

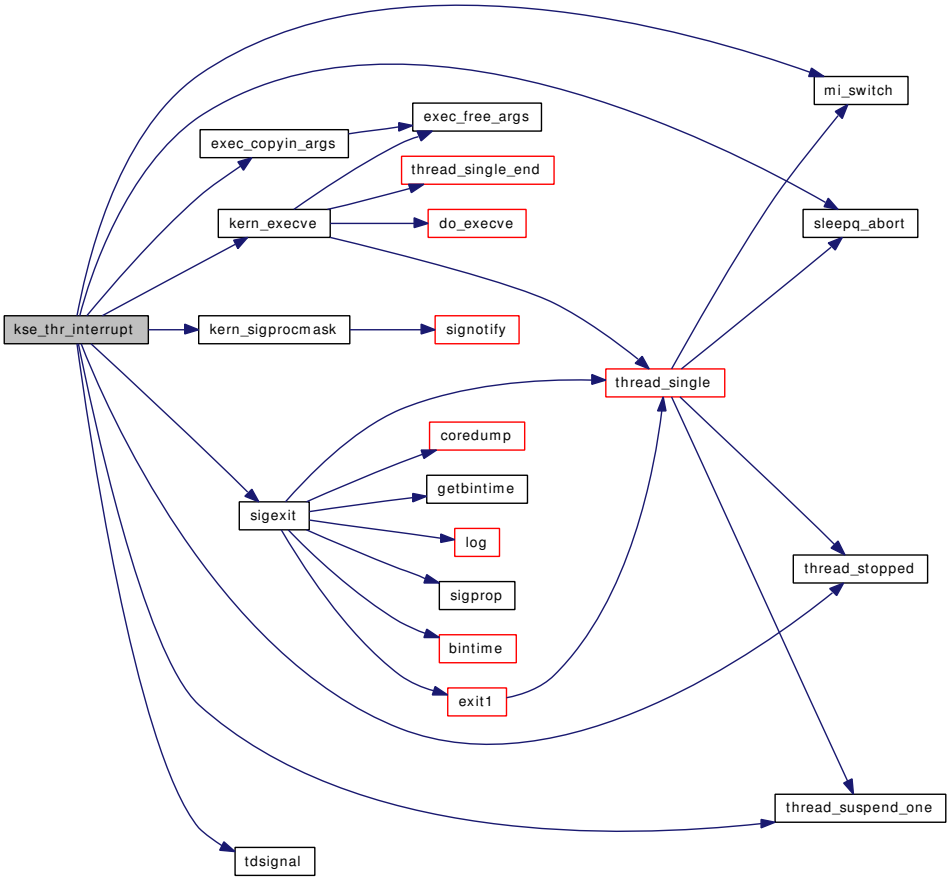
References kse_switchin_args::flags, sched_lock, suword, and kse_switchin_args::tmbx.

9.32.1.6 int kse_thr_interrupt (struct thread * *td*, struct kse_thr_interrupt_args * *uap*)

Definition at line 183 of file kern_kse.c.

References exec_copyin_args(), kern_execve(), kern_sigprocmask(), mi_switch(), sched_lock, sigexit(), sleepq_abort(), tdsignal(), thread_stopped(), and thread_suspend_one().

Here is the call graph for this function:

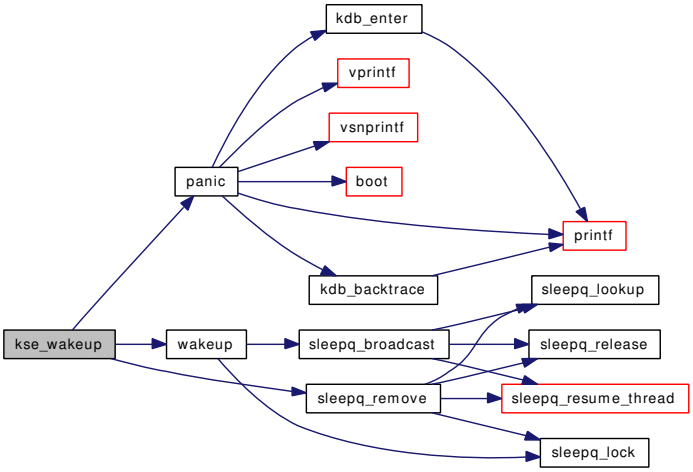


9.32.1.7 int kse_wakeup (struct thread * *td*, struct kse_wakeup_args * *uap*)

Definition at line 469 of file kern_kse.c.

References panic(), sched_lock, sleepq_remove(), and wakeup().

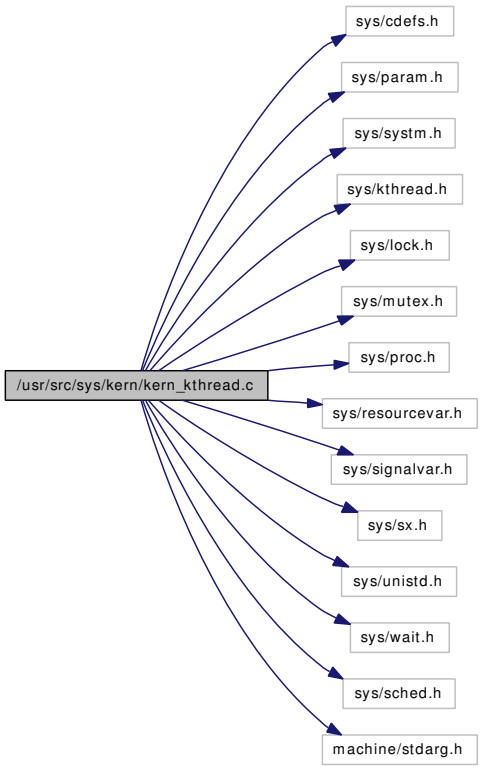
Here is the call graph for this function:


kdb_enter
vprintf
vsnprintf
panic
boot
printf
kdb_backtrace
sleepq_lookup
kse_wakeup
wakeup
sleepq_broadcast
sleepq_release
sleepq_remove
sleepq_resume_thread
sleepq_lock

9.33 /usr/src/sys/kern/kern_kthread.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kthread.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/resourcevar.h>
#include <sys/signalvar.h>
#include <sys/sx.h>
#include <sys/unistd.h>
#include <sys/wait.h>
#include <sys/sched.h>
#include <machine/stdarg.h>
```

Include dependency graph for kern_kthread.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_kthread.c,v 1.37 2007/01/23 08:46:50 jeff Exp $")
- void kproc_start (void *udata) const
- int kthread_create (void(*func)(void *), void *arg, struct proc **newpp, int flags, int pages, const char *fmt,...)
- void kthread_exit (int ecode)
- int kthread_suspend (struct proc *p, int timo)
- int kthread_resume (struct proc *p)
- void kthread_suspend_check (struct proc *p)

9.33.1 Function Documentation

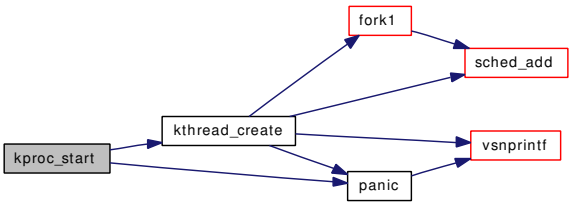
9.33.1.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_kthread. *c*, v 1.37 2007/01/23 08:46:50 jeff Exp $")

9.33.1.2 void kproc_start (void * *udata*) const

Definition at line 53 of file kern_kthread.c.

References kthread_create(), and panic().

Here is the call graph for this function:



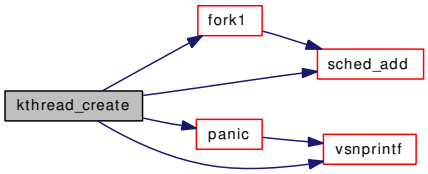
9.33.1.3 int kthread_create (void(*)(void *) *func*, void * *arg*, struct proc ** *newpp*, int *flags*, int *pages*, const char * *fmt*, ...)

Definition at line 76 of file kern_kthread.c.

References fork1(), panic(), proc0, sched_add(), sched_lock, td, and vsnprintf().

Referenced by acct(), aio_newproc(), idle_setup(), ithread_create(), kproc_start(), and taskqueue_start_threads().

Here is the call graph for this function:



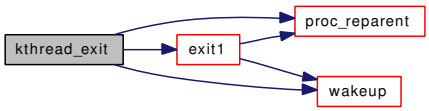
9.33.1.4 void kthread_exit (int *ecode*)

Definition at line 125 of file kern_kthread.c.

References exit1(), initproc, proc_reparent(), proctree_lock, td, and wakeup().

Referenced by acct_thread(), aio_daemon(), fork_exit(), ithread_loop(), and taskqueue_thread_loop().

Here is the call graph for this function:

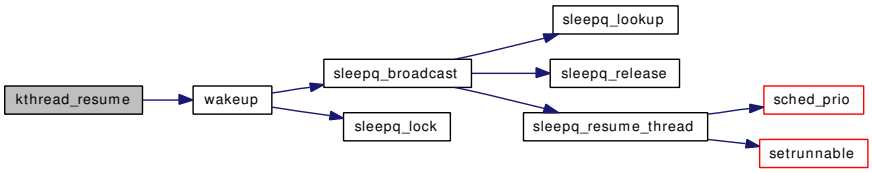


9.33.1.5 int kthread_resume (struct proc * *p*)

Definition at line 174 of file kern_kthread.c.

References wakeup().

Here is the call graph for this function:



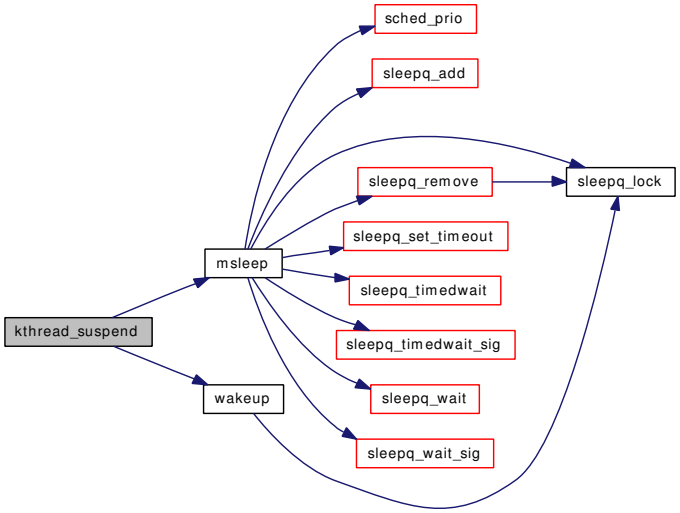
9.33.1.6 int kthread_suspend (struct proc * *p*, int *timo*)

Definition at line 157 of file kern_kthread.c.

References msleep(), and wakeup().

Referenced by kproc_shutdown().

Here is the call graph for this function:



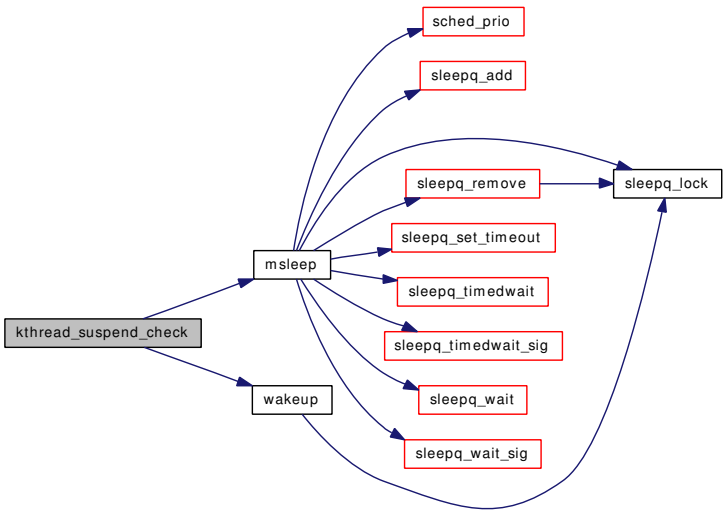
9.33.1.7 void kthread_suspend_check (struct proc * *p*)

Definition at line 192 of file kern_kthread.c.

References msleep(), and wakeup().

Referenced by buf_daemon(), sched_sync(), and vnlru_proc().

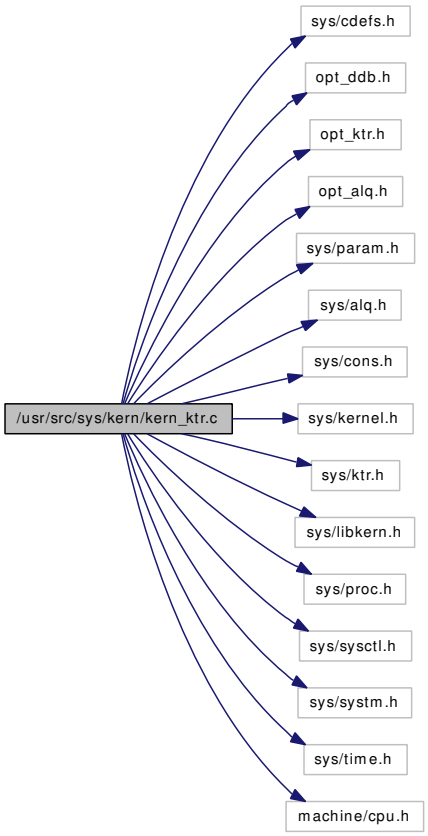
Here is the call graph for this function:



9.34 /usr/src/sys/kern/kern_ktr.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ddb.h"
#include "opt_ktr.h"
#include "opt_alq.h"
#include <sys/param.h>
#include <sys/alq.h>
#include <sys/cons.h>
#include <sys/kernel.h>
#include <sys/ktr.h>
#include <sys/libkern.h>
#include <sys/proc.h>
#include <sys/sysctl.h>
#include <sys/systm.h>
#include <sys/time.h>
#include <machine/cpu.h>
```

Include dependency graph for kern_ktr.c:



Defines

- #define KTR_ENTRIES 1024
- #define KTR_MASK (KTR_GEN)
- #define KTR_CPUMASK (~0)
- #define KTR_TIME get_cyclecount()
- #define KTR_CPU PCPU_GET(cpuid)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_ktr.c,v 1.53 2006/09/09 16:09:01 rwatson Exp $")
- SYSCTL_NODE (_debug, OID_AUTO, ktr, CTLFLAG_RD, 0,"KTR options")
- TUNABLE_INT ("debug.ktr.cpumask",&ktr_cpumask)
- SYSCTL_INT (_debug_ktr, OID_AUTO, cpumask, CTLFLAG_RW,&ktr_cpumask, 0,"")
- TUNABLE_INT ("debug.ktr.mask",&ktr_mask)
- SYSCTL_INT (_debug_ktr, OID_AUTO, mask, CTLFLAG_RW,&ktr_mask, 0,"")
- SYSCTL_INT (_debug_ktr, OID_AUTO, compile, CTLFLAG_RD,&ktr_compile, 0,"")
- SYSCTL_INT (_debug_ktr, OID_AUTO, entries, CTLFLAG_RD,&ktr_entries, 0,"")
- SYSCTL_INT (_debug_ktr, OID_AUTO, version, CTLFLAG_RD,&ktr_version, 0,"")
- static int sysctl_debug_ktr_clear (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_debug_ktr, OID_AUTO, clear, CTLTYPE_INT|CTLFLAG_RW, 0, 0, sysctl_-debug_ktr_clear,"I","Clear KTR Buffer")
- void ktr_tracepoint (u_int mask, const char *file, int line, const char *format, u_long arg1, u_long arg2, u_long arg3, u_long arg4, u_long arg5, u_long arg6)

Variables

- int ktr_cpumask = KTR_CPUMASK
- int ktr_mask = KTR_MASK
- int ktr_compile = KTR_COMPILE
- int ktr_entries = KTR_ENTRIES
- int ktr_version = KTR_VERSION
- volatile int ktr_idx = 0
- ktr_entry ktr_buf [KTR_ENTRIES]

9.34.1 Define Documentation

9.34.1.1 #define KTR_CPU PCPU_GET(cpuid)

Definition at line 80 of file kern_ktr.c.

Referenced by ktr_tracepoint().

9.34.1.2 #define KTR_CPUMASK (~0)

Definition at line 72 of file kern_ktr.c.

9.34.1.3 #define KTR_ENTRIES 1024

Definition at line 64 of file kern_ktr.c.

Referenced by ktr_tracepoint().

9.34.1.4 #define KTR_MASK (KTR_GEN)

Definition at line 68 of file kern_ktr.c.

9.34.1.5 #define KTR_TIME get_cyclecount()

Definition at line 76 of file kern_ktr.c.

Referenced by ktr_tracepoint().

9.34.2 Function Documentation

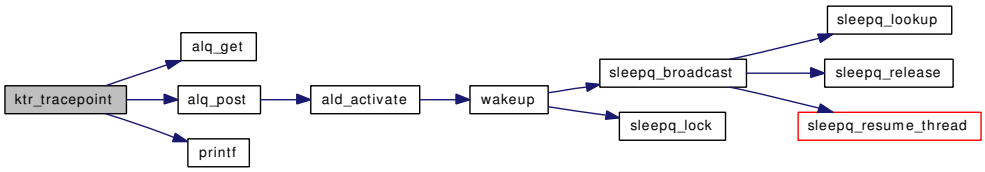
9.34.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_ktr. *c*, v 1.53 2006/09/09 16:09:01 rwatson Exp $")

9.34.2.2 void ktr_tracepoint (u_int *mask*, const char * *file*, int *line*, const char * *format*, u_long *arg1*, u_long *arg2*, u_long *arg3*, u_long *arg4*, u_long *arg5*, u_long *arg6*)

Definition at line 190 of file kern_ktr.c.

References alq_get(), alq_post(), ktr_buf, KTR_CPU, ktr_cpumask, KTR_ENTRIES, ktr_idx, ktr_mask, KTR_TIME, panicstr, printf(), and td.

Here is the call graph for this function:

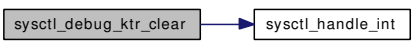


9.34.2.3 static int sysctl_debug_ktr_clear (SYSCTL_HANDLER_ARGS) [static]

Definition at line 106 of file kern_ktr.c.

References ktr_buf, ktr_idx, and sysctl_handle_int().

Here is the call graph for this function:



9.34.2.4 SYSCTL_INT (_debug_ktr, OID_AUTO, version, CTLFLAG_RD, & *ktr_version*, 0, "")

9.34.2.5 SYSCTL_INT (_debug_ktr, OID_AUTO, entries, CTLFLAG_RD, & *ktr_entries*, 0, "")

9.34.2.6 SYSCTL_INT (_debug_ktr, OID_AUTO, compile, CTLFLAG_RD, & *ktr_compile*, 0, "")

9.34.2.7 SYSCTL_INT (_debug_ktr, OID_AUTO, mask, CTLFLAG_RW, & *ktr_mask*, 0, "")

9.34.2.8 SYSCTL_INT (_debug_ktr, OID_AUTO, cpumask, CTLFLAG_RW, & *ktr_cpumask*, 0, "")

9.34.2.9 SYSCTL_NODE (_debug, OID_AUTO, ktr, CTLFLAG_RD, 0, "KTR options")

9.34.2.10 SYSCTL_PROC (_debug_ktr, OID_AUTO, clear, CTLTYPE_INT| *CTLFLAG_RW*, 0, 0, sysctl_debug_ktr_clear, "I", "Clear KTR Buffer")

9.34.2.11 TUNABLE_INT ("debug.ktr.mask", & *ktr_mask*)

9.34.2.12 TUNABLE_INT ("debug.ktr.cpumask", & *ktr_cpumask*)

9.34.3 Variable Documentation

9.34.3.1 struct ktr_entry ktr_buf[KTR_ENTRIES]

Definition at line 103 of file kern_ktr.c.

Referenced by ktr_tracepoint(), and sysctl_debug_ktr_clear().

9.34.3.2 int ktr_compile = KTR_COMPILE

Definition at line 93 of file kern_ktr.c.

9.34.3.3 int ktr_cpumask = KTR_CPUMASK

Definition at line 85 of file kern_ktr.c.

Referenced by ktr_tracepoint().

9.34.3.4 int ktr_entries = KTR_ENTRIES

Definition at line 96 of file kern_ktr.c.

9.34.3.5 volatile int ktr_idx = 0

Definition at line 102 of file kern_ktr.c.

Referenced by ktr_tracepoint(), and sysctl_debug_ktr_clear().

9.34.3.6 int ktr_mask = KTR_MASK

Definition at line 89 of file kern_ktr.c.

Referenced by ktr_tracepoint().

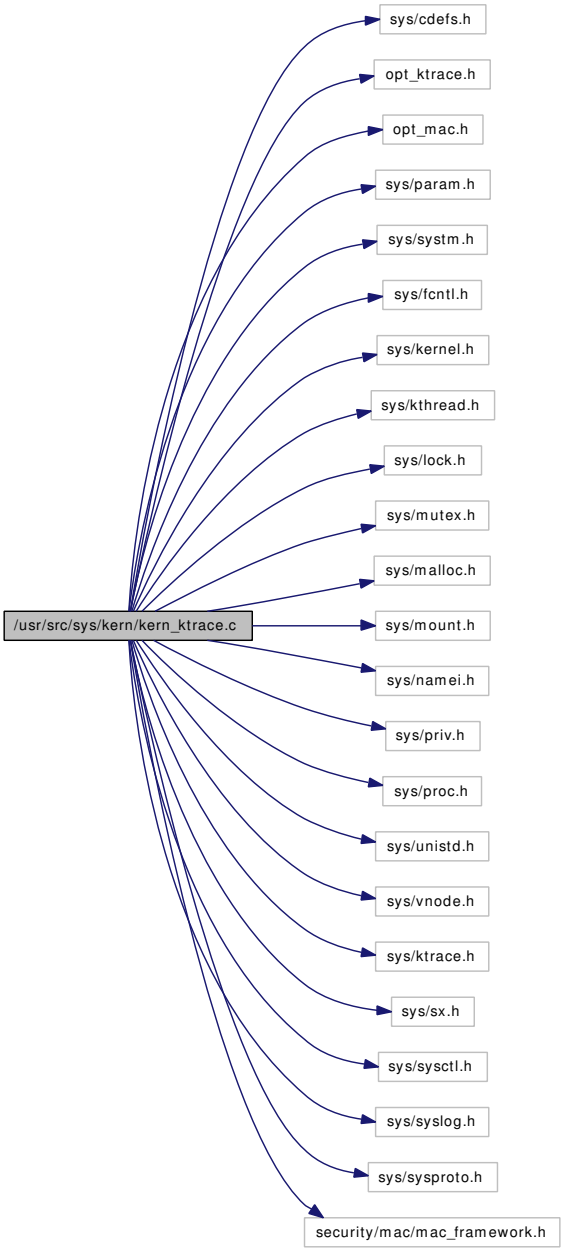
9.34.3.7 int ktr_version = KTR_VERSION

Definition at line 99 of file kern_ktr.c.

9.35 /usr/src/sys/kern/kern_ktrace.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ktrace.h"
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/fcntl.h>
#include <sys/kernel.h>
#include <sys/kthread.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/malloc.h>
#include <sys/mount.h>
#include <sys/namei.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/unistd.h>
#include <sys/vnode.h>
#include <sys/ktrace.h>
#include <sys/sx.h>
#include <sys/sysctl.h>
#include <sys/syslog.h>
#include <sys/sysproto.h>
#include <security/mac/mac_framework.h>
```

Include dependency graph for kern_ktrace.c:



Data Structures

- struct ktrace_args

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_ktrace.c,v 1.115 2007/02/13 00:20:13 mpp Exp $")
- static MALLOC_DEFINE (M_KTRACE,"KTRACE","KTRACE")
- int ktrace (struct thread *td, struct ktrace_args *uap)
- int utrace (struct thread *td, struct utrace_args *uap)

9.35.1 Function Documentation

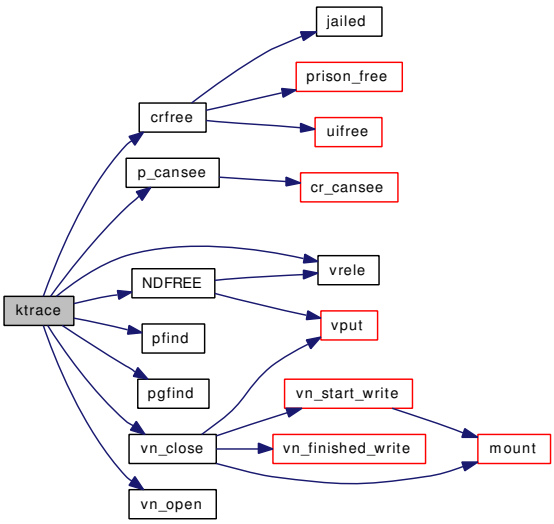
9.35.1.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_ktrace. *c*, v 1.115 2007/02/13 00:20:13 mpp Exp $")

9.35.1.2 int ktrace (struct thread * *td*, struct ktrace_args * *uap*)

Definition at line 580 of file kern_ktrace.c.

References allproc_lock, crfree(), NDFREE(), p_cansee(), pfind(), pgfind(), proctree_lock, ret, vn_close(), vn_open(), and vrele().

Here is the call graph for this function:



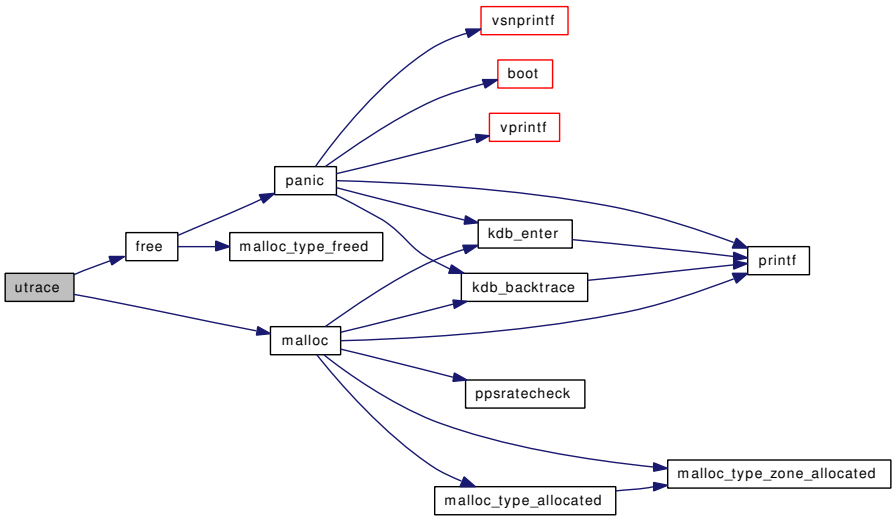
9.35.1.3 static MALLOC_DEFINE (M_KTRACE, "KTRACE", "KTRACE") [static]

9.35.1.4 int utrace (struct thread * *td*, struct utrace_args * *uap*)

Definition at line 747 of file kern_ktrace.c.

References free(), and malloc().

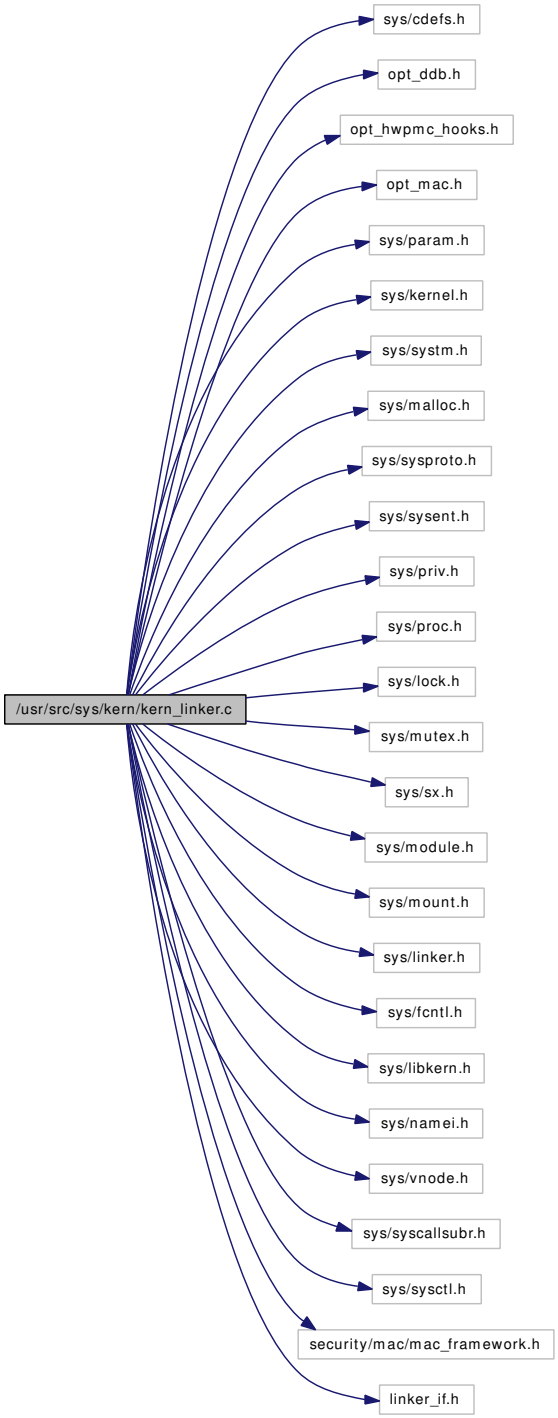
Here is the call graph for this function:


vsnprintf
boot
vprintf
panic
kdb_enter
printf
free
malloc_type_freed
kdb_backtrace
utrace
malloc
ppsratecheck
malloc_type_zone_allocated
malloc_type_allocated

9.36 /usr/src/sys/kern/kern_linker.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ddb.h"
#include "opt_hwpmc_hooks.h"
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/kernel.h>
#include <sys/systm.h>
#include <sys/malloc.h>
#include <sys/sysproto.h>
#include <sys/sysent.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/sx.h>
#include <sys/module.h>
#include <sys/mount.h>
#include <sys/linker.h>
#include <sys/fcntl.h>
#include <sys/libkern.h>
#include <sys/namei.h>
#include <sys/vnode.h>
#include <sys/syscallsubr.h>
#include <sys/sysctl.h>
#include <security/mac/mac_framework.h>
#include "linker_if.h"
```

Include dependency graph for kern_linker.c:



Defines

- #define KLD_LOCK() sx_xlock(&kld_sx)
- #define KLD_UNLOCK() sx_xunlock(&kld_sx)
- #define KLD_LOCKED() sx_xlocked(&kld_sx)
- #define KLD_LOCK_ASSERT()

- #define LINKER_GET_NEXT_FILE_ID(a)
- #define INT_ALIGN(base, ptr)

Typedefs

- typedef modlist * modlist_t

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_linker.c,v 1.145 2007/02/23 19:46:59 jhb Exp $")
- static const char * linker_basename (const char *path)
- static linker_file_t linker_find_file_by_name (const char *_filename)
- static linker_file_t linker_find_file_by_id (int _fileid)
- SET_DECLARE (modmetadata_set, struct mod_metadata)
- MALLOC_DEFINE (M_LINKER,"linker","kernel linker")
- typedef TAILQ_HEAD (modlist)
- static int linker_file_add_dependency (linker_file_t file, linker_file_t dep)
- static caddr_t linker_file_lookup_symbol_internal (linker_file_t file, const char *name, int deps)
- static int linker_load_module (const char *kldname, const char *modname, struct linker_file *parent, struct mod_depend *verinfo, struct linker_file **lfpp)
- static modlist_t modlist_lookup2 (const char *name, struct mod_depend *verinfo)
- static char * linker_strdup (const char *str)
- static void linker_init (void *arg)
- static void linker_stop_class_add (void *arg)
- int linker_add_class (linker_class_t lc)
- static void linker_file_sysinit (linker_file_t lf)
- static void linker_file_sysuninit (linker_file_t lf)
- static void linker_file_register_sysctls (linker_file_t lf)
- static void linker_file_unregister_sysctls (linker_file_t lf)
- static int linker_file_register_modules (linker_file_t lf)
- static void linker_init_kernel_modules (void)
- static int linker_load_file (const char *filename, linker_file_t *result)
- int linker_reference_module (const char *modname, struct mod_depend *verinfo, linker_file_t *result)
- int linker_release_module (const char *modname, struct mod_depend *verinfo, linker_file_t lf)
- int linker_file_foreach (linker_predicate_t *predicate, void *context)
- linker_file_t linker_make_file (const char *pathname, linker_class_t lc)
- int linker_file_unload (linker_file_t file, int flags)
- int linker_file_lookup_set (linker_file_t file, const char *name, void *firstp, void *lastp, int *countp)
- caddr_t linker_file_lookup_symbol (linker_file_t file, const char *name, int deps)
- int kern_kldload (struct thread *td, const char *file, int *fileid)
- int kldload (struct thread *td, struct kldload_args *uap)
- int kern_kldunload (struct thread *td, int fileid, int flags)
- int kldunload (struct thread *td, struct kldunload_args *uap)
- int kldunloadf (struct thread *td, struct kldunloadf_args *uap)
- int kldfind (struct thread *td, struct kldfind_args *uap)
- int kldnext (struct thread *td, struct kldnext_args *uap)
- int kldstat (struct thread *td, struct kldstat_args *uap)
- int kldfirstmod (struct thread *td, struct kldfirstmod_args *uap)

- int kldsym (struct thread *td, struct kldsym_args *uap)
- static modlist_t modlist_lookup (const char *name, int ver)
- static modlist_t modlist_newmodule (const char *modname, int version, linker_file_t container)
- static void linker_addmodules (linker_file_t lf, struct mod_metadata **start, struct mod_metadata **stop, int preload)
- static void linker_preload (void *arg)
- SYSCTL_STRING (_kern, OID_AUTO, module_path, CTLFLAG_RW, linker_path, sizeof(linker_path),"module load search path")
- TUNABLE_STR ("module_path", linker_path, sizeof(linker_path))
- static char * linker_lookup_file (const char *path, int pathlen, const char *name, int namelen, struct vattr *vap)
- static char * linker_hints_lookup (const char *path, int pathlen, const char *modname, int modnamelen, struct mod_depend *verinfo)
- static char * linker_search_module (const char *modname, int modnamelen, struct mod_depend *verinfo)
- static char * linker_search_kld (const char *name)
- int linker_load_dependencies (linker_file_t lf)
- static int sysctl_kern_function_list_iterate (const char *name, void *opaque)
- static int sysctl_kern_function_list (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern, OID_AUTO, function_list, CTLFLAG_RD, NULL, 0, sysctl_kern_function_list,"","kernel function list")

Variables

- linker_file_t linker_kernel_file
- static struct sx kld_sx
- static linker_class_list_t classes
- static linker_file_list_t linker_files
- static int next_file_id = 1
- static int linker_no_more_classes = 0
- static modlisthead_t found_modules
- static char linker_hintfile [ ] = "linker.hints"
- static char linker_path [MAXPATHLEN] = "/boot/kernel;/boot/modules"
- static char * linker_ext_list [ ]

9.36.1 Define Documentation

9.36.1.1 #define INT_ALIGN(base, ptr)

Value:

```
ptr =                                           \
        (base) + (((ptr) - (base) + sizeof(int) - 1) & ~(sizeof(int) - 1))
```

Definition at line 1563 of file kern_linker.c.

Referenced by linker_hints_lookup().

9.36.1.2 #define KLD_LOCK() sx_xlock(&kld_sx)

Definition at line 67 of file kern_linker.c.

Referenced by kern_kldload(), kern_kldunload(), kldfind(), kldfirstmod(), kldnext(), kldstat(), kldsym(), linker_file_foreach(), linker_file_lookup_set(), linker_file_lookup_symbol(), linker_load_file(), linker_reference_module(), linker_release_module(), and sysctl_kern_function_list().

9.36.1.3 #define KLD_LOCK_ASSERT()

Value:

```
do {                                                \
        if (!cold)                                                      \
                sx_assert(&kld_sx, SX_XLOCKED);                         \
} while (0)
```

Definition at line 70 of file kern_linker.c.

Referenced by linker_file_add_dependency(), linker_file_lookup_symbol_internal(), linker_file_unload(), linker_find_file_by_id(), linker_find_file_by_name(), linker_load_dependencies(), linker_load_file(), linker_load_module(), and linker_make_file().

9.36.1.4 #define KLD_LOCKED() sx_xlocked(&kld_sx)

Definition at line 69 of file kern_linker.c.

Referenced by linker_file_lookup_set(), and linker_file_lookup_symbol().

9.36.1.5 #define KLD_UNLOCK() sx_xunlock(&kld_sx)

Definition at line 68 of file kern_linker.c.

Referenced by kern_kldload(), kern_kldunload(), kldfind(), kldfirstmod(), kldnext(), kldstat(), kldsym(), linker_file_foreach(), linker_file_lookup_set(), linker_file_lookup_symbol(), linker_load_file(), linker_reference_module(), linker_release_module(), and sysctl_kern_function_list().

9.36.1.6 #define LINKER_GET_NEXT_FILE_ID(a)

Value:

```
do {                                    \
        linker_file_t lftmp;                                            \
                                                                        \
        KLD_LOCK_ASSERT();                                              \
retry:                                                                  \
        TAILQ_FOREACH(lftmp, &linker_files, link) {                     \
                if (next_file_id == lftmp->id) {                        \
                        next_file_id++;                                 \
                        goto retry;                                     \
                }                                                       \
        }                                                               \
        (a) = next_file_id;                                             \
} while(0)
```

Definition at line 105 of file kern_linker.c.

Referenced by linker_make_file().

9.36.2 Typedef Documentation

9.36.2.1 typedef struct modlist* modlist_t

Definition at line 128 of file kern_linker.c.

9.36.3 Function Documentation

9.36.3.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_linker. c, v 1.145 2007/02/23 19:46:59 jhb Exp $")

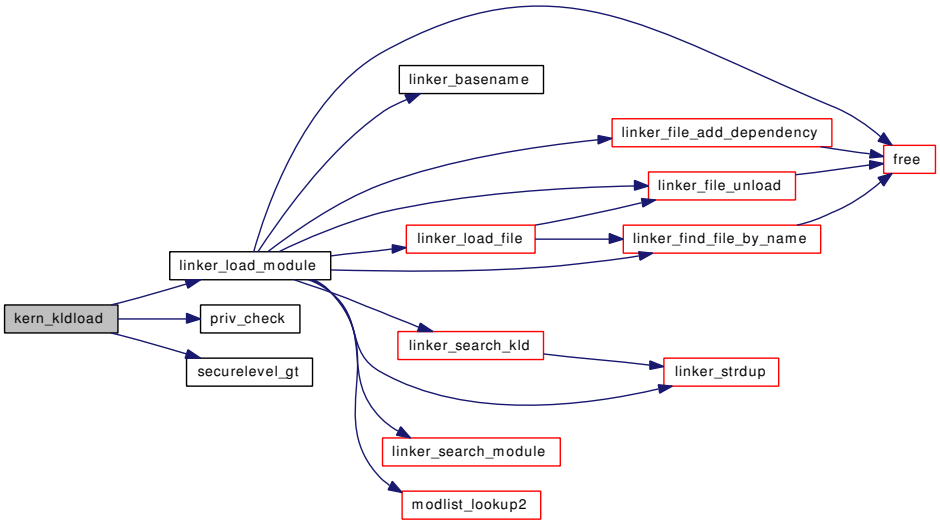
9.36.3.2 int kern_kldload (struct thread * *td*, const char * *file*, int * *fileid*)

Definition at line 848 of file kern_linker.c.

References KLD_LOCK, KLD_UNLOCK, linker_load_module(), priv_check(), and securelevel_gt().

Referenced by kldload(), and vfs_byname_kld().

Here is the call graph for this function:



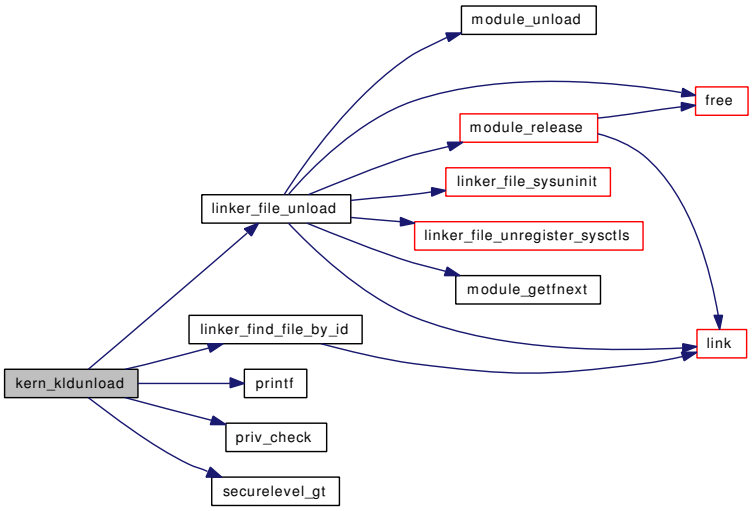
9.36.3.3 int kern_kldunload (struct thread * *td*, int *fileid*, int *flags*)

Definition at line 916 of file kern_linker.c.

References KLD_LOCK, KLD_UNLOCK, linker_file_unload(), linker_find_file_by_id(), printf(), priv_check(), and securelevel_gt().

Referenced by kldunload(), kldunloadf(), and vfs_byname_kld().

Here is the call graph for this function:

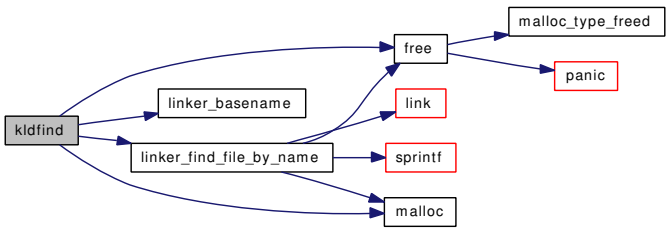


9.36.3.4 int kldfind (struct thread * *td*, struct kldfind_args * *uap*)

Definition at line 990 of file kern_linker.c.

References free(), KLD_LOCK, KLD_UNLOCK, linker_basename(), linker_find_file_by_name(), and malloc().

Here is the call graph for this function:

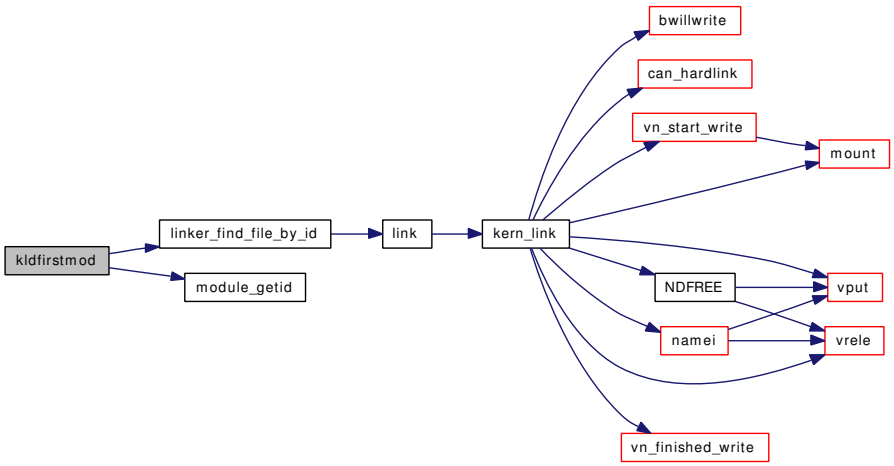


9.36.3.5 int kldfirstmod (struct thread * *td*, struct kldfirstmod_args * *uap*)

Definition at line 1113 of file kern_linker.c.

References KLD_LOCK, KLD_UNLOCK, linker_find_file_by_id(), and module_getid().

Here is the call graph for this function:

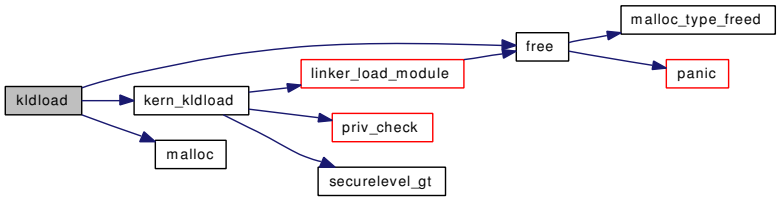


9.36.3.6 int kldload (struct thread * *td*, struct kldload_args * *uap*)

Definition at line 894 of file kern_linker.c.

References free(), kern_kldload(), and malloc().

Here is the call graph for this function:

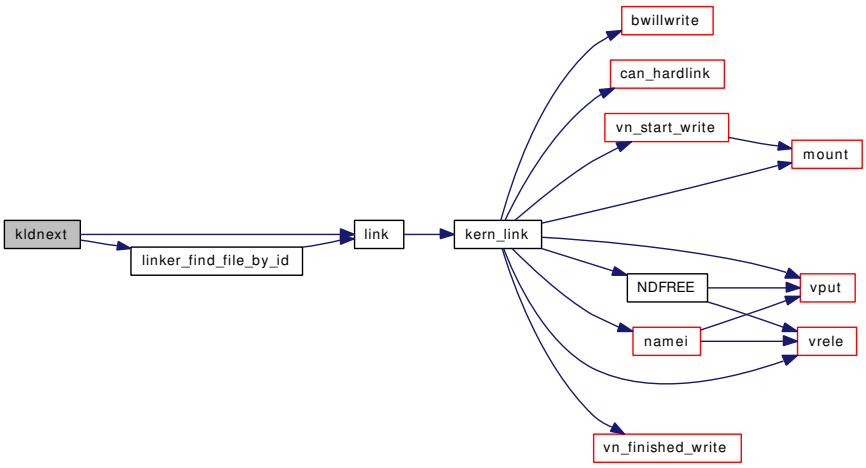


9.36.3.7 int kldnext (struct thread * *td*, struct kldnext_args * *uap*)

Definition at line 1026 of file kern_linker.c.

References KLD_LOCK, KLD_UNLOCK, link(), linker_files, and linker_find_file_by_id().

Here is the call graph for this function:

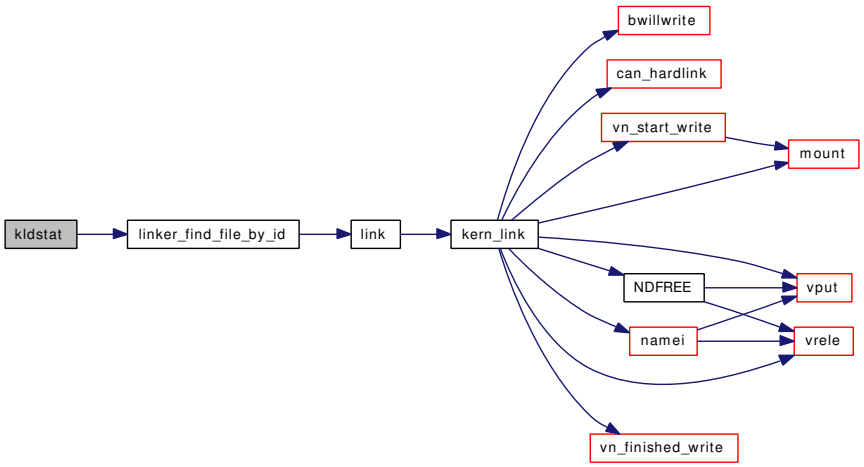


9.36.3.8 int kldstat (struct thread * *td*, struct kldstat_args * *uap*)

Definition at line 1066 of file kern_linker.c.

References KLD_LOCK, KLD_UNLOCK, and linker_find_file_by_id().

Here is the call graph for this function:

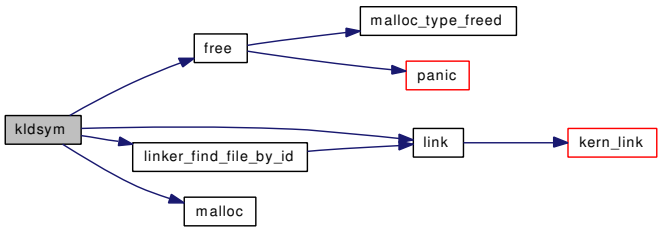


9.36.3.9 int kldsym (struct thread * *td*, struct kldsym_args * *uap*)

Definition at line 1145 of file kern_linker.c.

References free(), KLD_LOCK, KLD_UNLOCK, link(), linker_files, linker_find_file_by_id(), and malloc().

Here is the call graph for this function:

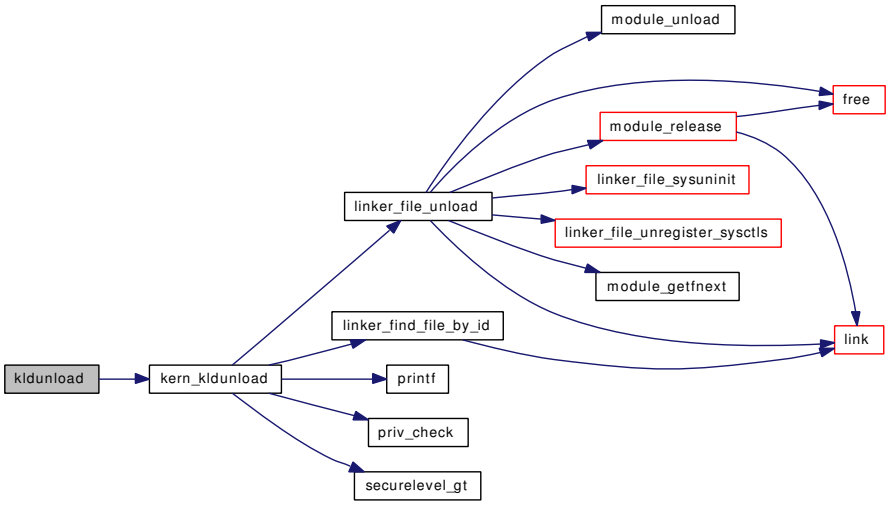


9.36.3.10 int kldunload (struct thread * *td*, struct kldunload_args * *uap*)

Definition at line 967 of file kern_linker.c.

References kern_kldunload().

Here is the call graph for this function:

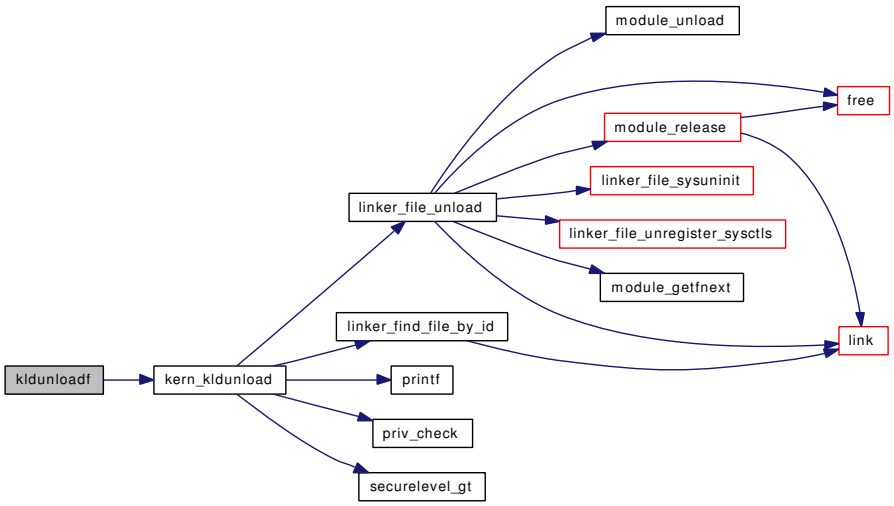


9.36.3.11 int kldunloadf (struct thread * *td*, struct kldunloadf_args * *uap*)

Definition at line 977 of file kern_linker.c.

References kern_kldunload().

Here is the call graph for this function:



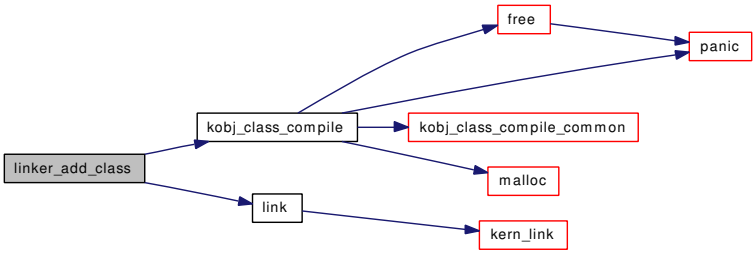
9.36.3.12 int linker_add_class (linker_class_t *lc*)

Definition at line 171 of file kern_linker.c.

References classes, kobj_class_compile(), link(), and linker_no_more_classes.

Referenced by link_elf_init().

Here is the call graph for this function:



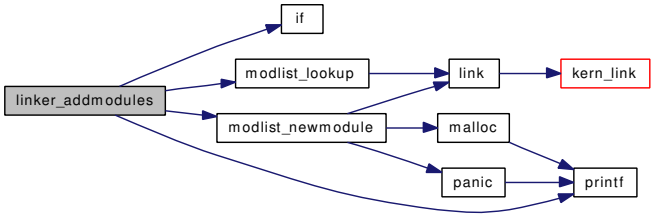
9.36.3.13 static void linker_addmodules (linker_file_t *lf*, struct mod_metadata ** *start*, struct mod_metadata ** *stop*, int *preload*) [static]

Definition at line 1256 of file kern_linker.c.

References if(), modlist_lookup(), modlist_newmodule(), and printf().

Referenced by linker_load_dependencies(), and linker_preload().

Here is the call graph for this function:



9.36.3.14 static const char * linker_basename (const char * *path*) [static]

Definition at line 1756 of file kern_linker.c.

Referenced by kldfind(), linker_load_module(), and linker_make_file().

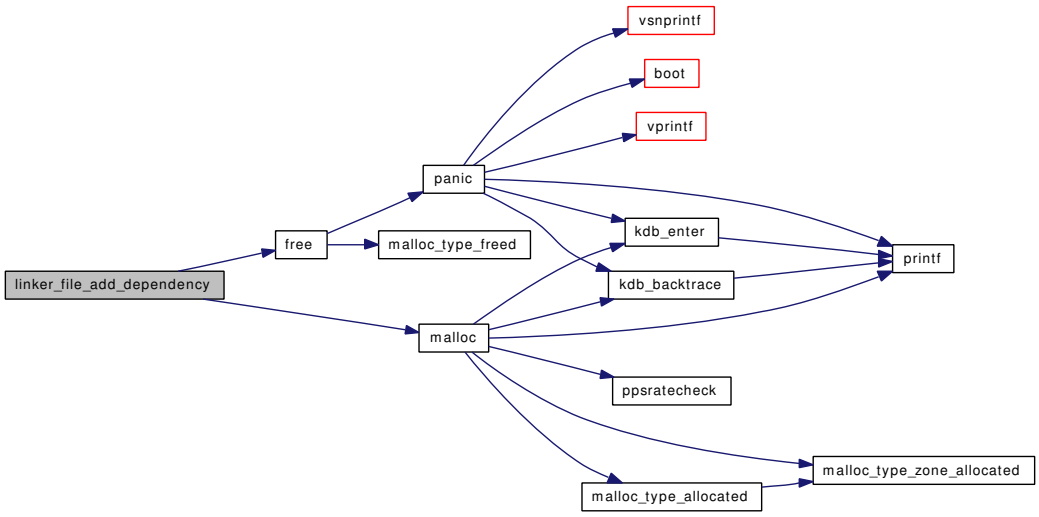
9.36.3.15 static int linker_file_add_dependency (linker_file_t *file*, linker_file_t *dep*) [static]

Definition at line 642 of file kern_linker.c.

References free(), KLD_LOCK_ASSERT, and malloc().

Referenced by linker_load_dependencies(), linker_load_module(), and linker_preload().

Here is the call graph for this function:

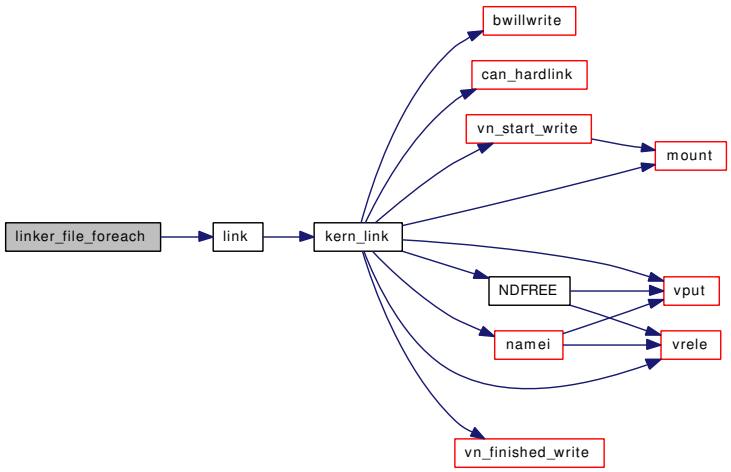


9.36.3.16 int linker_file_foreach (linker_predicate_t * *predicate*, void * *context*)

Definition at line 513 of file kern_linker.c.

References KLD_LOCK, KLD_UNLOCK, link(), and linker_files.

Here is the call graph for this function:



9.36.3.17 int linker_file_lookup_set (linker_file_t *file*, const char * *name*, void * *firstp*, void * *lastp*, int * *countp*)

Definition at line 670 of file kern_linker.c.

References KLD_LOCK, KLD_LOCKED, and KLD_UNLOCK.

Referenced by linker_file_register_modules(), linker_file_register_sysctls(), linker_file_sysinit(), linker_file_sysuninit(), linker_file_unregister_sysctls(), linker_load_dependencies(), and linker_preload().

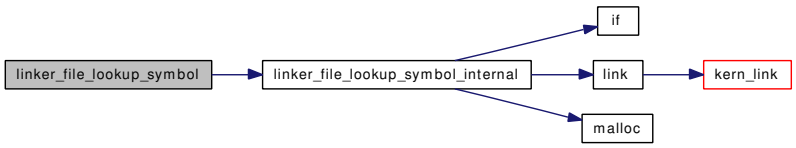
9.36.3.18 caddr_t linker_file_lookup_symbol (linker_file_t *file*, const char * *name*, int *deps*)

Definition at line 685 of file kern_linker.c.

References KLD_LOCK, KLD_LOCKED, KLD_UNLOCK, and linker_file_lookup_symbol_internal().

Referenced by elf_lookup(), and elf_obj_lookup().

Here is the call graph for this function:



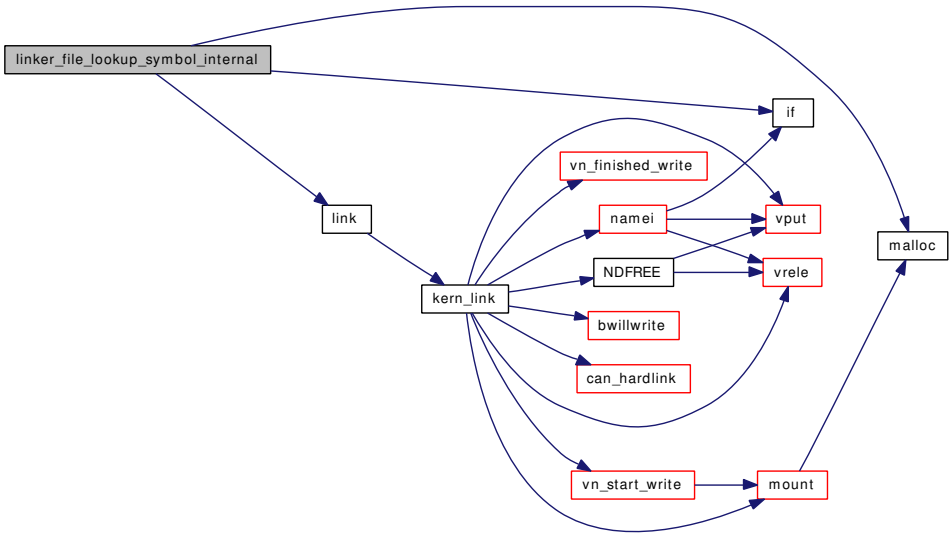
9.36.3.19 static caddr_t linker_file_lookup_symbol_internal (linker_file_t *file*, const char * *name*, int *deps*) [static]

Definition at line 700 of file kern_linker.c.

References if(), KLD_LOCK_ASSERT, link(), and malloc().

Referenced by linker_file_lookup_symbol().

Here is the call graph for this function:



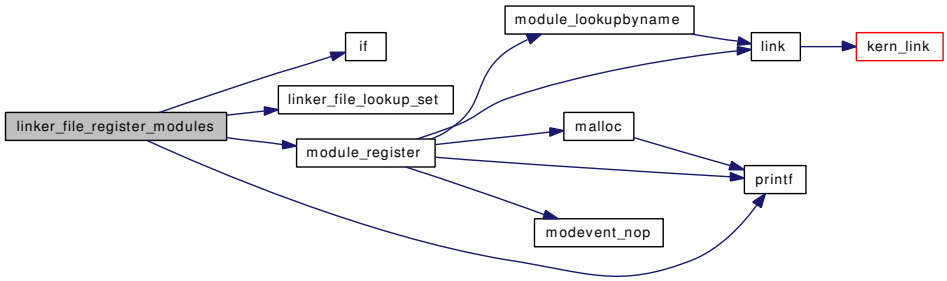
9.36.3.20 static int linker_file_register_modules (linker_file_t *lf*) [static]

Definition at line 313 of file kern_linker.c.

References if(), linker_file_lookup_set(), linker_kernel_file, module_register(), and printf().

Referenced by linker_init_kernel_modules(), linker_load_file(), and linker_preload().

Here is the call graph for this function:



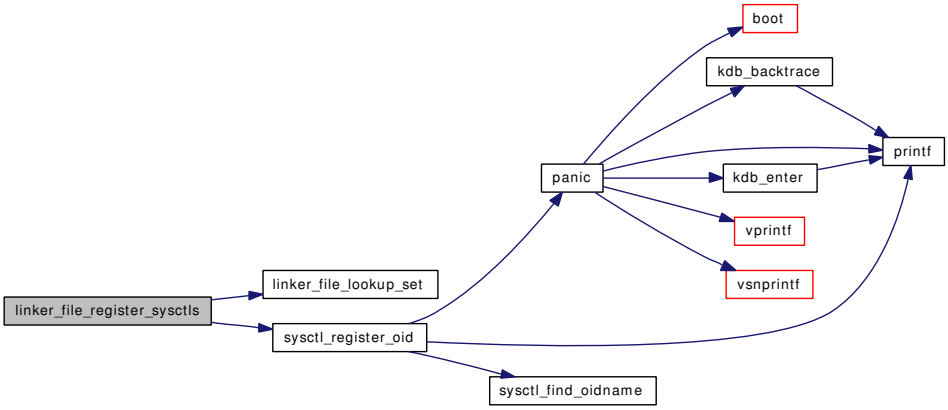
9.36.3.21 static void linker_file_register_sysctls (linker_file_t *lf*) [static]

Definition at line 278 of file kern_linker.c.

References Giant, linker_file_lookup_set(), and sysctl_register_oid().

Referenced by linker_load_file(), and linker_preload().

Here is the call graph for this function:



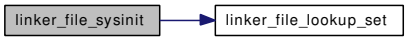
9.36.3.22 static void linker_file_sysinit (linker_file_t *lf*) [static]

Definition at line 188 of file kern_linker.c.

References Giant, linker_file_lookup_set(), and sysinit.

Referenced by linker_load_file().

Here is the call graph for this function:



9.36.3.23 static void linker_file_sysuninit (linker_file_t *lf*) [static]

Definition at line 232 of file kern_linker.c.

References Giant, linker_file_lookup_set(), and sysinit.

Referenced by linker_file_unload().

Here is the call graph for this function:

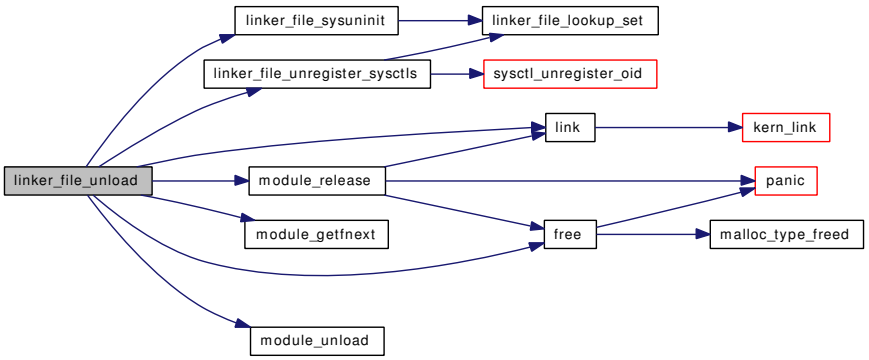
9.36.3.24 int linker_file_unload (linker_file_t *file*, int *flags*)

Definition at line 555 of file kern_linker.c.

References found_modules, free(), KLD_LOCK_ASSERT, link(), linker_file_sysuninit(), linker_file_-unregister_sysctls(), linker_files, module_getfnext(), module_release(), module_unload(), and securelevel.

Referenced by kern_kldunload(), link_elf_link_preload(), link_elf_load_file(), linker_load_file(), linker_-load_module(), linker_preload(), and linker_release_module().

Here is the call graph for this function:



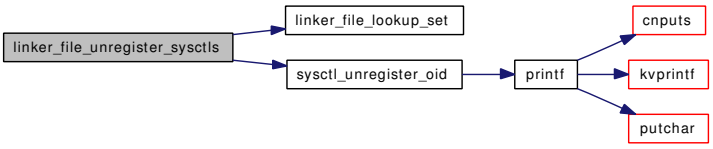
9.36.3.25 static void linker_file_unregister_sysctls (linker_file_t *lf*) [static]

Definition at line 296 of file kern_linker.c.

References Giant, linker_file_lookup_set(), and sysctl_unregister_oid().

Referenced by linker_file_unload().

Here is the call graph for this function:



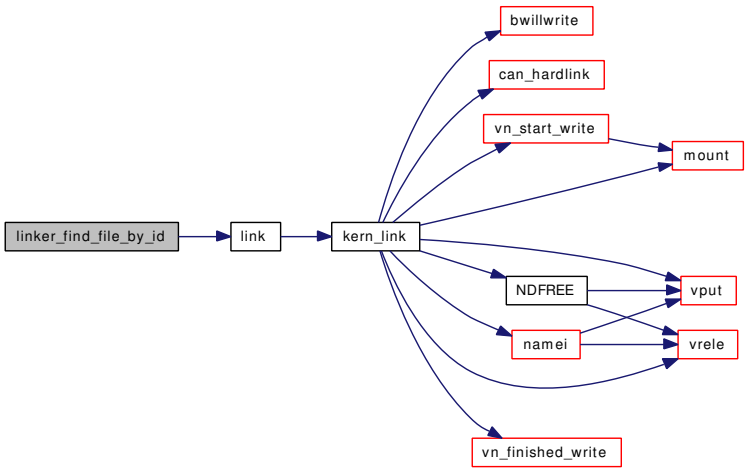
9.36.3.26 static linker_file_t linker_find_file_by_id (int *_fileid*) [static]

Definition at line 501 of file kern_linker.c.

References KLD_LOCK_ASSERT, link(), and linker_files.

Referenced by kern_kldunload(), kldfirstmod(), kldnext(), kldstat(), and kldsym().

Here is the call graph for this function:



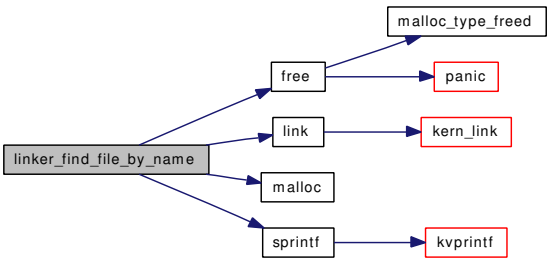
9.36.3.27 static linker_file_t linker_find_file_by_name (const char * *filename*) [static]

Definition at line 481 of file kern_linker.c.

References free(), KLD_LOCK_ASSERT, link(), linker_files, malloc(), and sprintf().

Referenced by kldfind(), linker_load_file(), and linker_load_module().

Here is the call graph for this function:



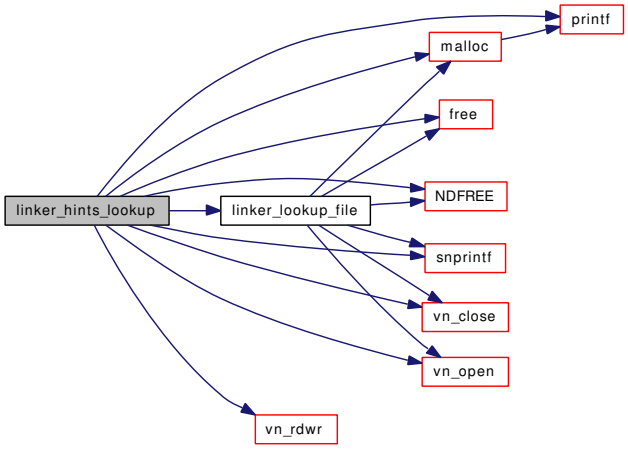
9.36.3.28 static char* linker_hints_lookup (const char * *path*, int *pathlen*, const char * *modname*, int *modnamelen*, struct mod_depend * *verinfo*) [static]

Definition at line 1572 of file kern_linker.c.

References free(), INT_ALIGN, linker_hintfile, linker_lookup_file(), malloc(), NDFREE(), printf(), snprintf(), td, vn_close(), vn_open(), and vn_rdwr().

Referenced by linker_search_module().

Here is the call graph for this function:

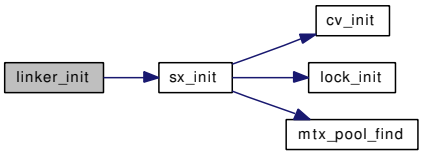


9.36.3.29 static void linker_init (void * *arg*) [static]

Definition at line 151 of file kern_linker.c.

References classes, kld_sx, linker_files, and sx_init().

Here is the call graph for this function:

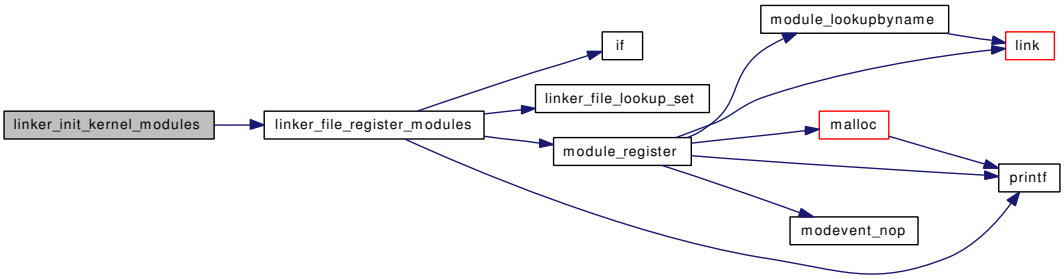


9.36.3.30 static void linker_init_kernel_modules (void) [static]

Definition at line 354 of file kern_linker.c.

References linker_file_register_modules(), and linker_kernel_file.

Here is the call graph for this function:



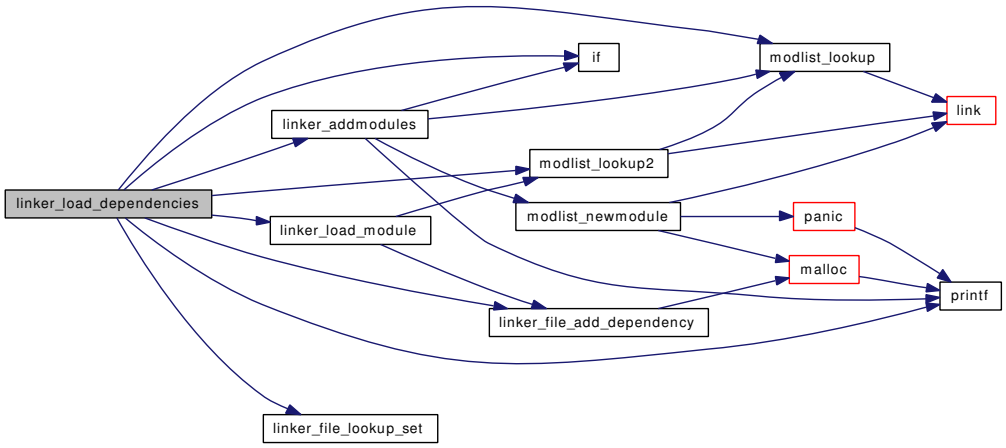
9.36.3.31 int linker_load_dependencies (linker_file_t *lf*)

Definition at line 1903 of file kern_linker.c.

References if(), KLD_LOCK_ASSERT, linker_addmodules(), linker_file_add_dependency(), linker_file_-lookup_set(), linker_kernel_file, linker_load_module(), modlist_lookup(), modlist_lookup2(), and printf().

Referenced by link_elf_load_file().

Here is the call graph for this function:



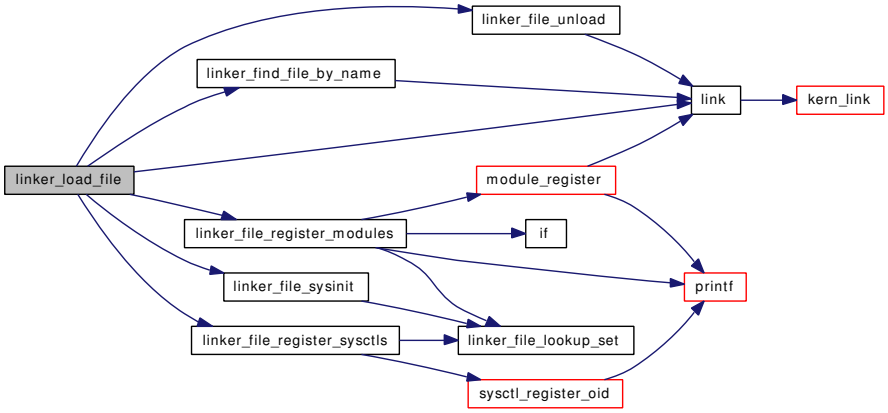
9.36.3.32 static int linker_load_file (const char * *filename*, linker_file_t * *result*) [static]

Definition at line 363 of file kern_linker.c.

References classes, KLD_LOCK, KLD_LOCK_ASSERT, KLD_UNLOCK, link(), linker_file_register_-modules(), linker_file_register_sysctls(), linker_file_sysinit(), linker_file_unload(), linker_find_file_by_-name(), and securelevel.

Referenced by linker_load_module().

Here is the call graph for this function:



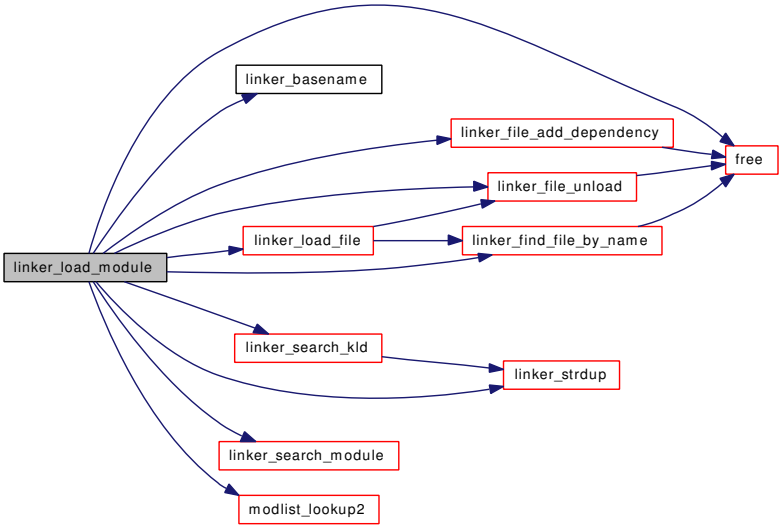
9.36.3.33 static int linker_load_module (const char * *kldname*, const char * *modname*, struct linker_file * *parent*, struct mod_depend * *verinfo*, struct linker_file ** *lfpp*) [static]

Definition at line 1833 of file kern_linker.c.

References free(), KLD_LOCK_ASSERT, linker_basename(), linker_file_add_dependency(), linker_file_unload(), linker_find_file_by_name(), linker_load_file(), linker_search_kld(), linker_search_module(), linker_strdup(), modlist_lookup2(), and rootvnode.

Referenced by kern_kldload(), link_elf_link_preload_finish(), link_elf_load_file(), linker_load_dependencies(), and linker_reference_module().

Here is the call graph for this function:



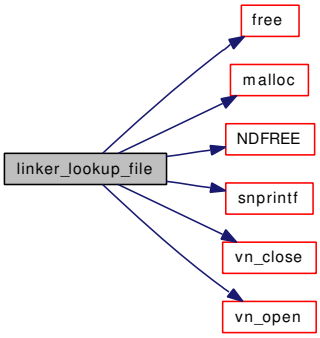
9.36.3.34 static char* linker_lookup_file (const char * *path*, int *pathlen*, const char * *name*, int *namelen*, struct vattr * *vap*) [static]

Definition at line 1516 of file kern_linker.c.

References free(), linker_ext_list, malloc(), NDFREE(), snprintf(), td, vn_close(), and vn_open().

Referenced by linker_hints_lookup(), and linker_search_kld().

Here is the call graph for this function:



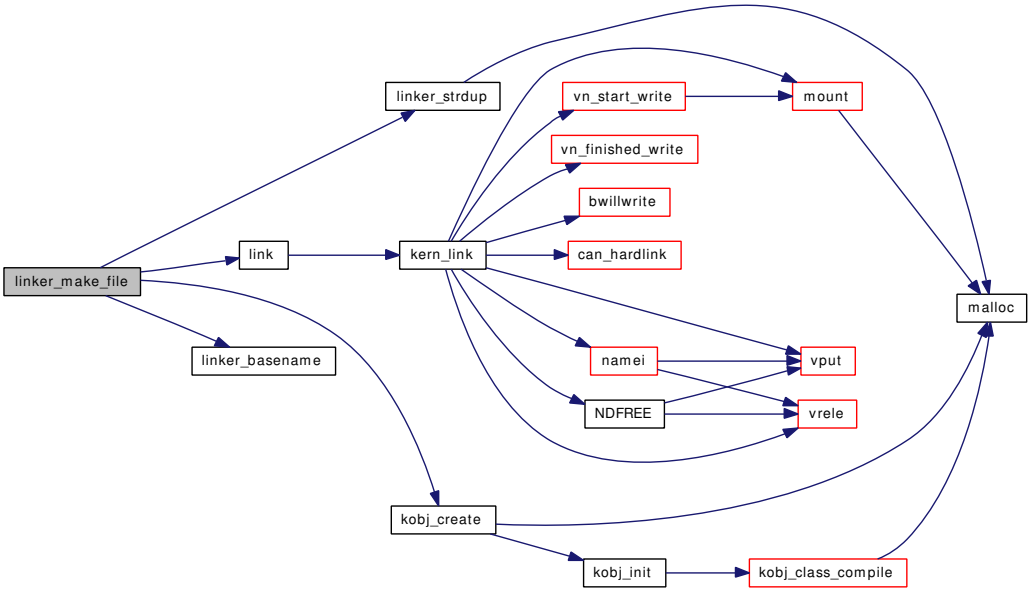
9.36.3.35 linker_file_t linker_make_file (const char * *pathname*, linker_class_t *lc*)

Definition at line 529 of file kern_linker.c.

References KLD_LOCK_ASSERT, kobj_create(), link(), linker_basename(), linker_files, LINKER_-GET_NEXT_FILE_ID, and linker_strdup().

Referenced by link_elf_init(), link_elf_link_preload(), and link_elf_load_file().

Here is the call graph for this function:



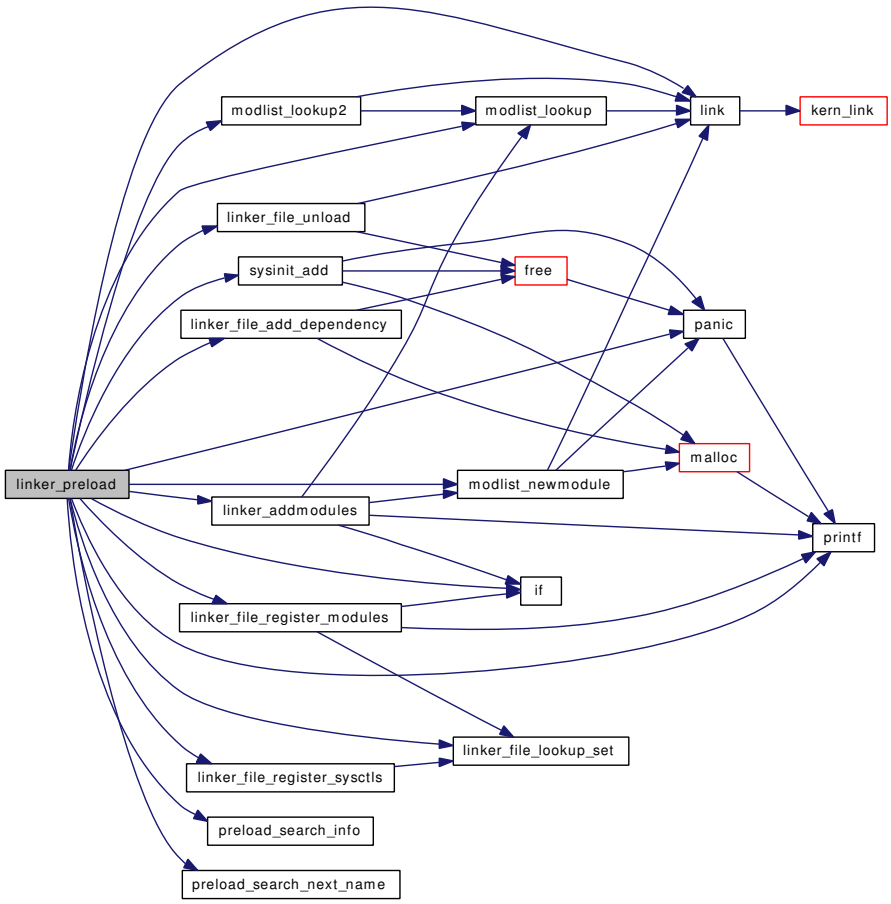
9.36.3.36 static void linker_preload (void * *arg*) [static]

Definition at line 1279 of file kern_linker.c.

References bootverbose, classes, found_modules, if(), link(), linker_addmodules(), linker_file_-add_dependency(), linker_file_lookup_set(), linker_file_register_modules(), linker_file_register_sysctls(),

linker_file_unload(), linker_kernel_file, modlist_lookup(), modlist_lookup2(), modlist_newmodule(), panic(), preload_search_info(), preload_search_next_name(), printf(), sysinit, and sysinit_add().

Here is the call graph for this function:



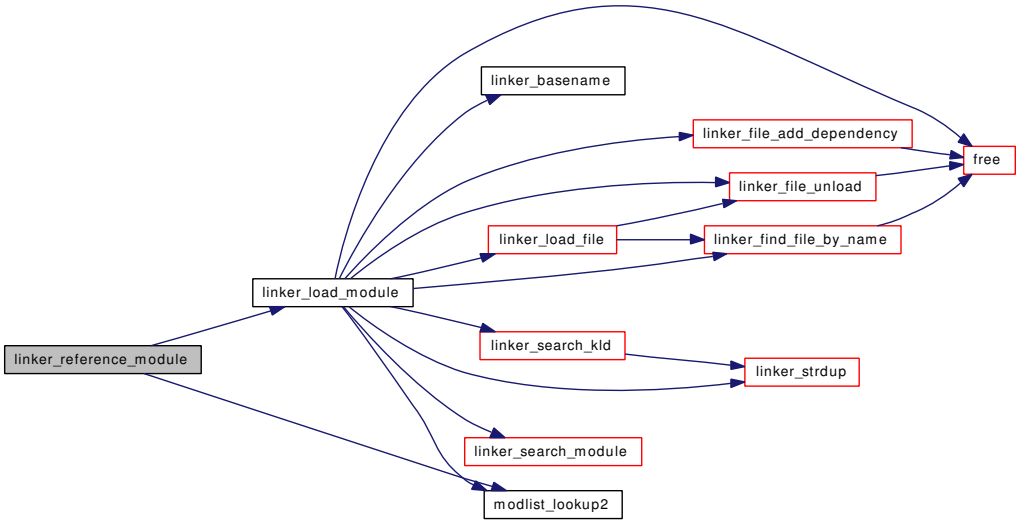
9.36.3.37 int linker_reference_module (const char * *modname*, struct mod_depend * *verinfo*, linker_file_t * *result*)

Definition at line 436 of file kern_linker.c.

References KLD_LOCK, KLD_UNLOCK, linker_load_module(), and modlist_lookup2().

Referenced by firmware_get().

Here is the call graph for this function:



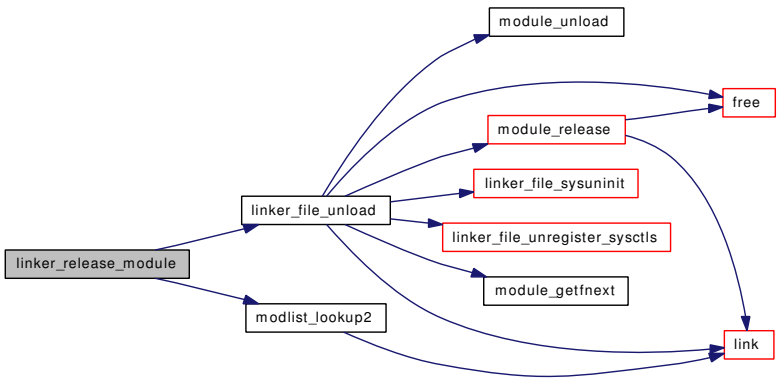
9.36.3.38 int linker_release_module (const char * *modname*, struct mod_depend * *verinfo*, linker_file_t *lf*)

Definition at line 456 of file kern_linker.c.

References KLD_LOCK, KLD_UNLOCK, linker_file_unload(), and modlist_lookup2().

Referenced by firmware_get(), and unloadentry().

Here is the call graph for this function:



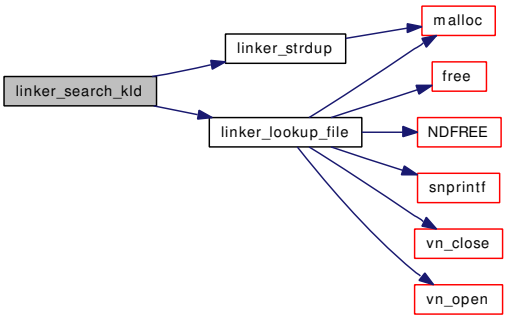
9.36.3.39 static char* linker_search_kld (const char * *name*) [static]

Definition at line 1733 of file kern_linker.c.

References linker_lookup_file(), linker_path, and linker_strdup().

Referenced by linker_load_module().

Here is the call graph for this function:



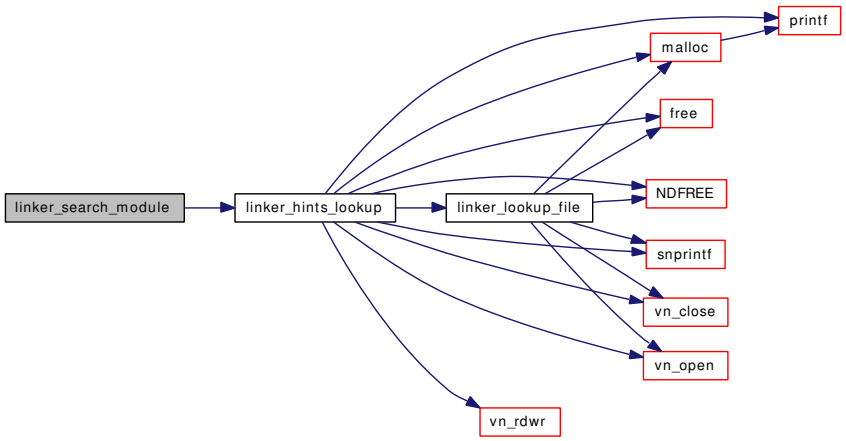
9.36.3.40 static char* linker_search_module (const char * *modname*, int *modnamelen*, struct mod_depend * *verinfo*) [static]

Definition at line 1708 of file kern_linker.c.

References linker_hints_lookup(), and linker_path.

Referenced by linker_load_module().

Here is the call graph for this function:



9.36.3.41 static void linker_stop_class_add (void * *arg*) [static]

Definition at line 162 of file kern_linker.c.

References linker_no_more_classes.

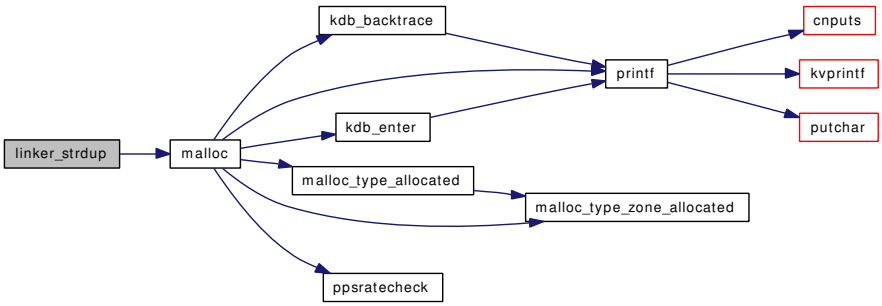
9.36.3.42 static char* linker_strdup (const char * *str*) [static]

Definition at line 141 of file kern_linker.c.

References malloc().

Referenced by linker_load_module(), linker_make_file(), and linker_search_kld().

Here is the call graph for this function:



9.36.3.43 MALLOC_DEFINE (M_LINKER, "linker", "kernel linker")

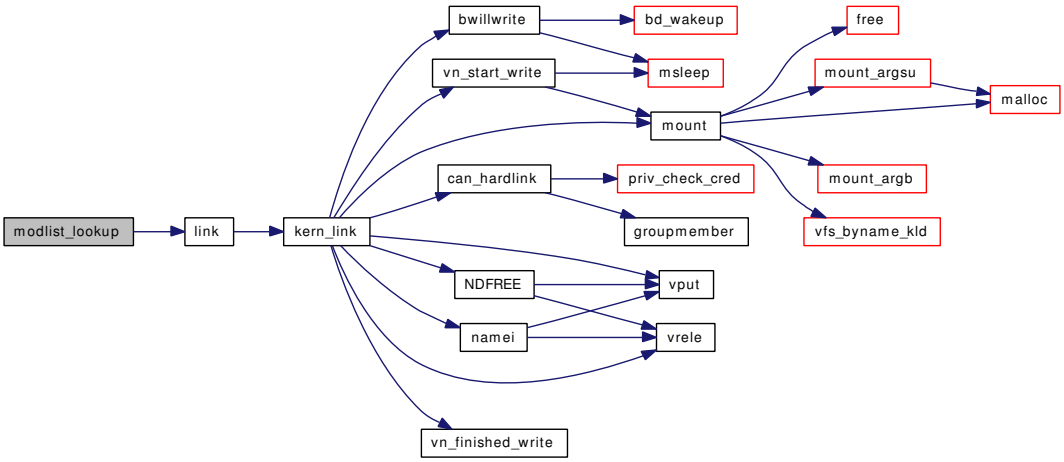
9.36.3.44 static modlist_t modlist_lookup (const char * *name*, int *ver*) [static]

Definition at line 1205 of file kern_linker.c.

References found_modules, and link().

Referenced by linker_addmodules(), linker_load_dependencies(), linker_preload(), and modlist_-lookup2().

Here is the call graph for this function:



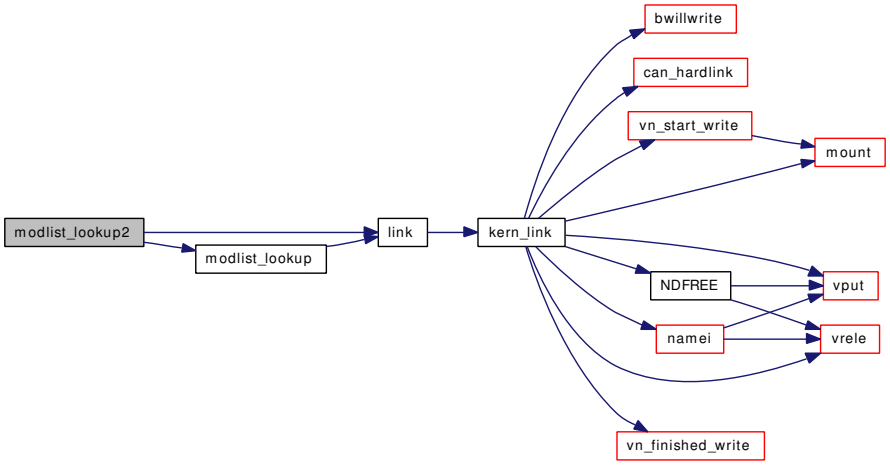
9.36.3.45 static modlist_t modlist_lookup2 (const char * *name*, struct mod_depend * *verinfo*) [static]

Definition at line 1218 of file kern_linker.c.

References found_modules, link(), and modlist_lookup().

Referenced by linker_load_dependencies(), linker_load_module(), linker_preload(), linker_reference_-module(), and linker_release_module().

Here is the call graph for this function:



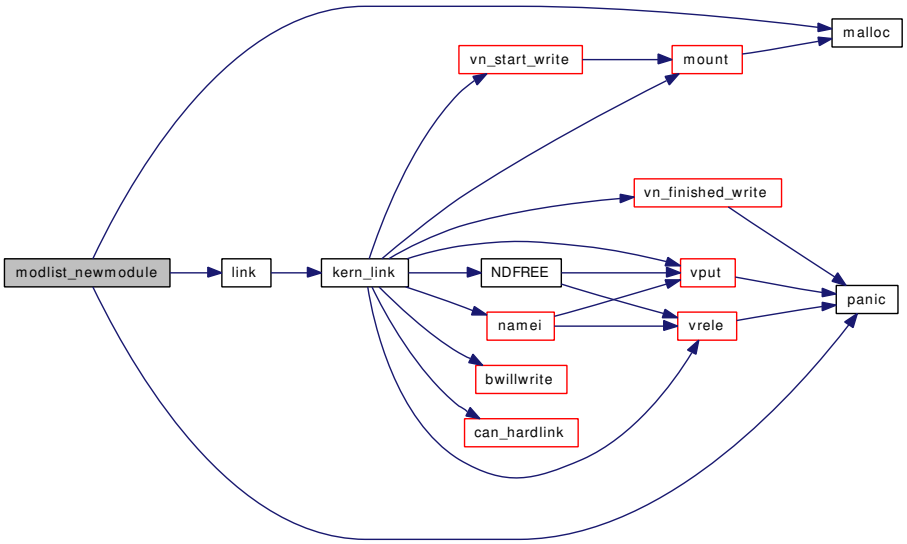
9.36.3.46 static modlist_t modlist_newmodule (const char * *modname*, int *version*, linker_file_t *container*) [static]

Definition at line 1241 of file kern_linker.c.

References found_modules, link(), malloc(), and panic().

Referenced by linker_addmodules(), and linker_preload().

Here is the call graph for this function:



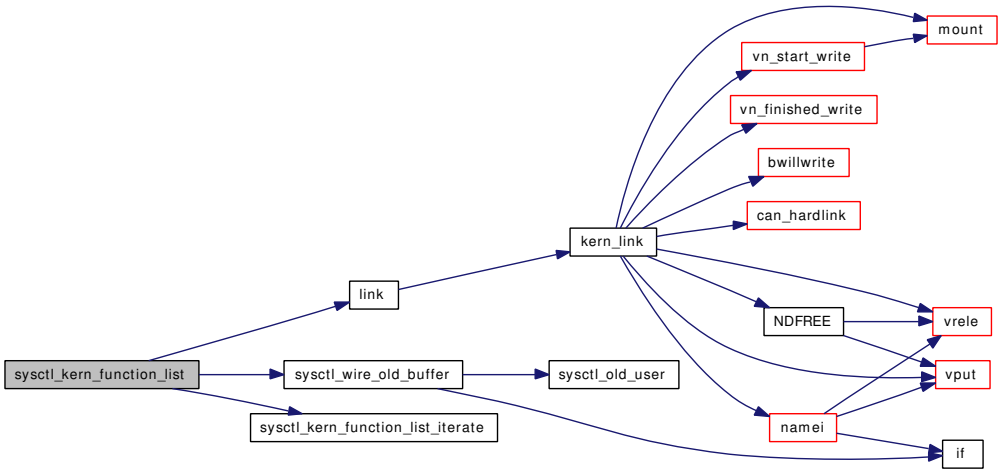
9.36.3.47 SET_DECLARE (modmetadata_set, struct *mod_metadata*)

9.36.3.48 static int sysctl_kern_function_list (SYSCTL_HANDLER_ARGS) [static]

Definition at line 1995 of file kern_linker.c.

References KLD_LOCK, KLD_UNLOCK, link(), linker_files, sysctl_kern_function_list_iterate(), and sysctl_wire_old_buffer().

Here is the call graph for this function:



9.36.3.49 static int sysctl_kern_function_list_iterate (const char * *name*, void * *opaque*) [static]

Definition at line 1982 of file kern_linker.c.

Referenced by sysctl_kern_function_list().

9.36.3.50 SYSCTL_PROC (_kern, OID_AUTO, function_list, CTLFLAG_RD, NULL, 0, sysctl_kern_function_list, "", "kernel function list")

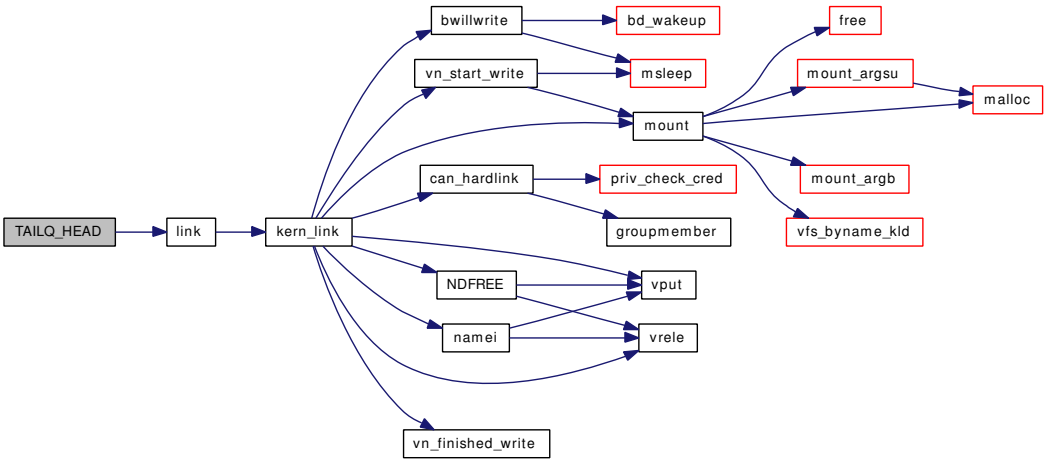
9.36.3.51 SYSCTL_STRING (_kern, OID_AUTO, module_path, CTLFLAG_RW, linker_path, sizeof(linker_path), "module load search path")

9.36.3.52 typedef TAILQ_HEAD (modlist)

Definition at line 121 of file kern_linker.c.

References link().

Here is the call graph for this function:



9.36.3.53 TUNABLE_STR ("module_path", linker_path, sizeof(linker_path))

9.36.4 Variable Documentation

9.36.4.1 linker_class_list_t classes [static]

Definition at line 100 of file kern_linker.c.

Referenced by linker_add_class(), linker_init(), linker_load_file(), and linker_preload().

9.36.4.2 modlisthead_t found_modules [static]

Definition at line 129 of file kern_linker.c.

Referenced by linker_file_unload(), linker_preload(), modlist_lookup(), modlist_lookup2(), and modlist_-newmodule().

9.36.4.3 struct sx kld_sx [static]

Definition at line 98 of file kern_linker.c.

Referenced by linker_init().

9.36.4.4 char* linker_ext_list[ ] [static]

Initial value:

```
{
        "",
        ".ko",
        NULL
}
```

Definition at line 1504 of file kern_linker.c.

Referenced by linker_lookup_file().

9.36.4.5 linker_file_list_t linker_files [static]

Definition at line 101 of file kern_linker.c.

Referenced by kldnext(), kldsym(), linker_file_foreach(), linker_file_unload(), linker_find_file_by_id(), linker_find_file_by_name(), linker_init(), linker_make_file(), and sysctl_kern_function_list().

9.36.4.6 char linker_hintfile[ ] = "linker.hints" [static]

Definition at line 1496 of file kern_linker.c.

Referenced by linker_hints_lookup().

9.36.4.7 linker_file_t linker_kernel_file

Definition at line 96 of file kern_linker.c.

Referenced by link_elf_init(), linker_file_register_modules(), linker_init_kernel_modules(), linker_load_dependencies(), and linker_preload().

9.36.4.8 int linker_no_more_classes = 0 [static]

Definition at line 103 of file kern_linker.c.

Referenced by linker_add_class(), and linker_stop_class_add().

9.36.4.9 char linker_path[MAXPATHLEN] = "/boot/kernel;/boot/modules" [static]

Definition at line 1497 of file kern_linker.c.

Referenced by linker_search_kld(), and linker_search_module().

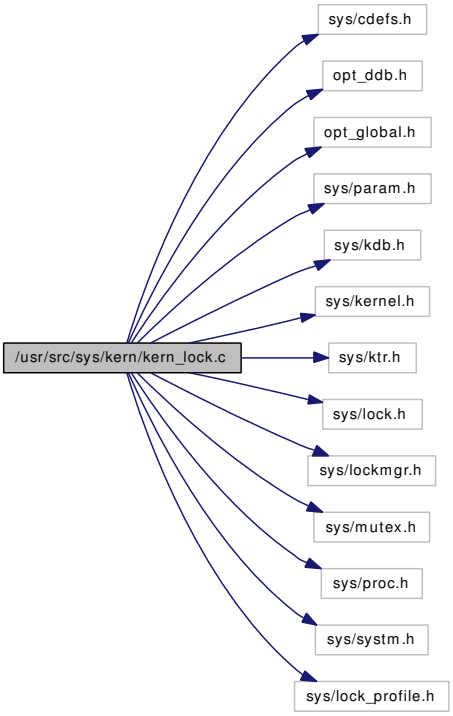
9.36.4.10 int next_file_id = 1 [static]

Definition at line 102 of file kern_linker.c.

9.37 /usr/src/sys/kern/kern_lock.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ddb.h"
#include "opt_global.h"
#include <sys/param.h>
#include <sys/kdb.h>
#include <sys/kernel.h>
#include <sys/ktr.h>
#include <sys/lock.h>
#include <sys/lockmgr.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/systm.h>
#include <sys/lock_profile.h>
```

Include dependency graph for kern_lock.c:



Defines

- #define COUNT(td, x) if ((td)) (td) → td_locks += (x)
- #define LK_ALL

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_lock.c,v 1.103 2006/11/13 05:41:46 kmacy Exp $")
- static int acquire (struct lock **lkpp, int extflags, int wanted)
- static int acquiredrain (struct lock *lkp, int extflags)
- static __inline void sharelock (struct thread *td, struct lock *lkp, int incr)
- static __inline void shareunlock (struct thread *td, struct lock *lkp, int decr)
- int _lockmgr (struct lock *lkp, int flags, struct mtx *interlkp, struct thread *td, char *file, int line)
- void transferlockers (struct lock *from, struct lock *to)
- void lockinit (struct lock *lkp, int prio, const char *wmesg, int timo, int flags)
- void lockdestroy (struct lock *lkp)
- int lockstatus (struct lock *lkp, struct thread *td)
- int lockcount (struct lock *lkp)
- int lockwaiters (struct lock *lkp)
- void lockmgr_printinfo (struct lock *lkp)

Variables

- lock_class lock_class_lockmgr

9.37.1 Define Documentation

9.37.1.1 #define COUNT(td, x) if ((td)) (td) → td_locks += (x)

Definition at line 83 of file kern_lock.c.

Referenced by _lockmgr(), sharelock(), and shareunlock().

9.37.1.2 #define LK_ALL

Value:

```
(LK_HAVE_EXCL | LK_WANT_EXCL | LK_WANT_UPGRADE | \
        LK_SHARE_NONZERO | LK_WAIT_NONZERO)
```

Definition at line 84 of file kern_lock.c.

Referenced by acquiredrain().

9.37.2 Function Documentation

9.37.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_lock. *c*, v 1.103 2006/11/13 05:41:46 kmacy Exp $")

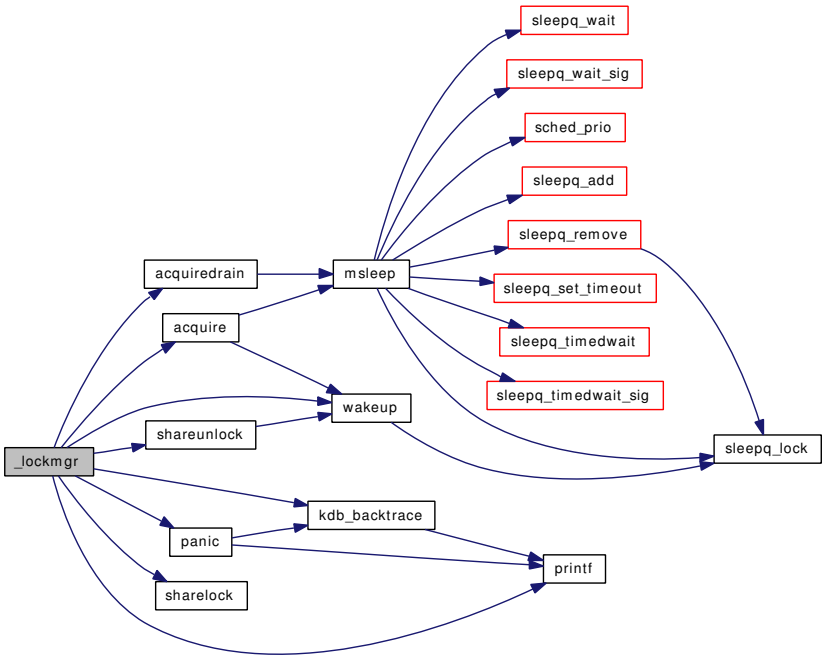
9.37.2.2 int _lockmgr (struct lock * *lkp*, int *flags*, struct mtx * *interlkp*, struct thread * *td*, char * *file*, int *line*)

Definition at line 164 of file kern_lock.c.

References acquire(), acquiredrain(), COUNT, kdb_backtrace(), panic(), panicstr, printf(), sharelock(), shareunlock(), and wakeup().

Referenced by vop_stdlock().

Here is the call graph for this function:



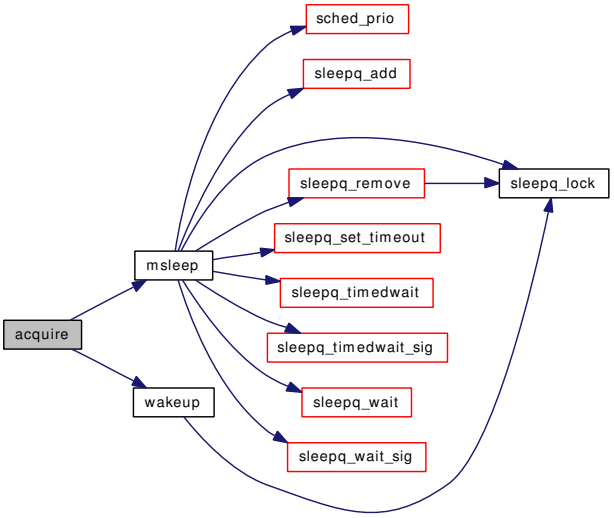
9.37.2.3 static int acquire (struct lock ** *lkpp*, int *extflags*, int *wanted*) [static]

Definition at line 115 of file kern_lock.c.

References msleep(), and wakeup().

Referenced by _lockmgr().

Here is the call graph for this function:



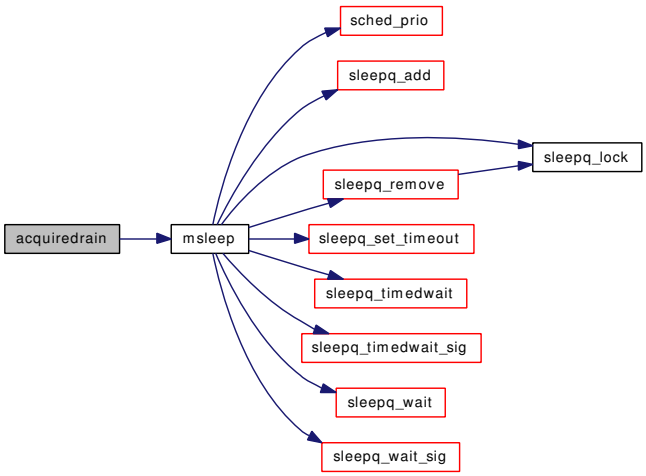
9.37.2.4 static int acquiredrain (struct lock * *lkp*, int *extflags*) [static]

Definition at line 460 of file kern_lock.c.

References LK_ALL, and msleep().

Referenced by _lockmgr().

Here is the call graph for this function:



9.37.2.5 int lockcount (struct lock * *lkp*)

Definition at line 582 of file kern_lock.c.

9.37.2.6 void lockdestroy (struct lock * *lkp*)

Definition at line 542 of file kern_lock.c.

Referenced by mount_fini(), and vdestroy().

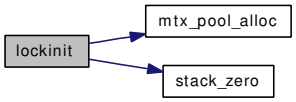
9.37.2.7 void lockinit (struct lock * *lkp*, int *prio*, const char * *wmesg*, int *timo*, int *flags*)

Definition at line 512 of file kern_lock.c.

References lock_class_lockmgr, mtx_pool_alloc(), mtxpool_lockbuilder, and stack_zero().

Referenced by getnewvnode(), and mount_init().

Here is the call graph for this function:



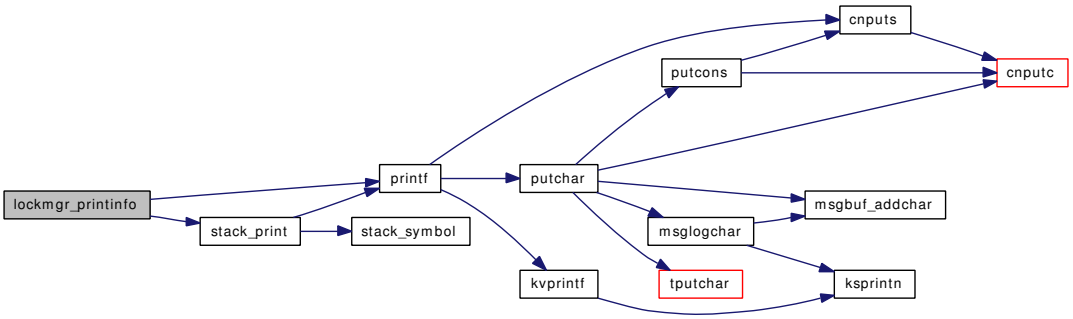
9.37.2.8 void lockmgr_printinfo (struct lock * *lkp*)

Definition at line 613 of file kern_lock.c.

References printf(), and stack_print().

Referenced by vn_printf().

Here is the call graph for this function:



9.37.2.9 int lockstatus (struct lock * *lkp*, struct thread * *td*)

Definition at line 554 of file kern_lock.c.

References kdb_active.

Referenced by vop_stdislocked().

9.37.2.10 int lockwaiters (struct lock * *lkp*)

Definition at line 597 of file kern_lock.c.

9.37.2.11 static __inline void sharelock (struct thread * *td*, struct lock * *lkp*, int *incr*) [static]

Definition at line 91 of file kern_lock.c.

References COUNT.

Referenced by _lockmgr().

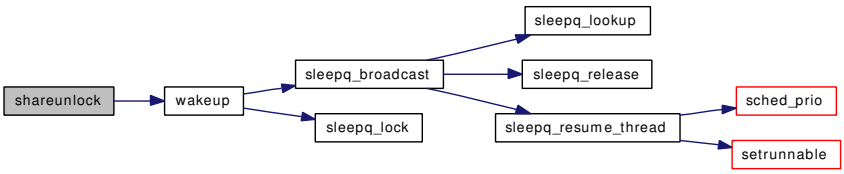
9.37.2.12 static __inline void shareunlock (struct thread * *td*, struct lock * *lkp*, int *decr*) [static]

Definition at line 98 of file kern_lock.c.

References COUNT, and wakeup().

Referenced by _lockmgr().

Here is the call graph for this function:

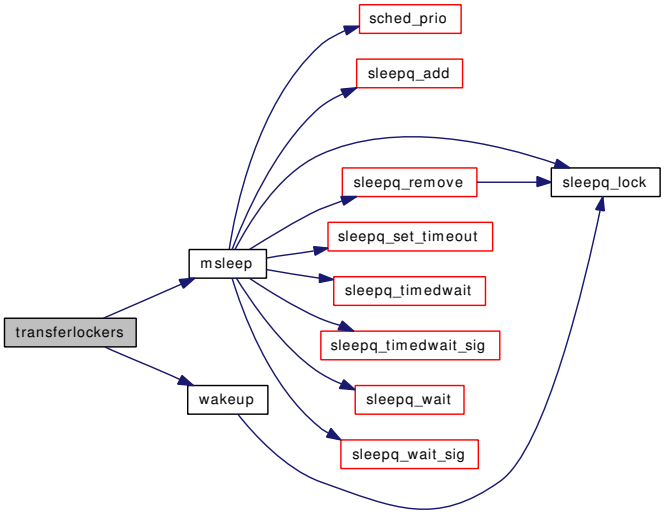


9.37.2.13 void transferlockers (struct lock * *from*, struct lock * *to*)

Definition at line 484 of file kern_lock.c.

References msleep(), and wakeup().

Here is the call graph for this function:



9.37.3 Variable Documentation

9.37.3.1 struct lock_class lock_class_lockmgr

Initial value:

```
{
        "lockmgr",
        LC_SLEEPLOCK | LC_SLEEPABLE | LC_RECURSABLE | LC_UPGRADABLE,


}
```

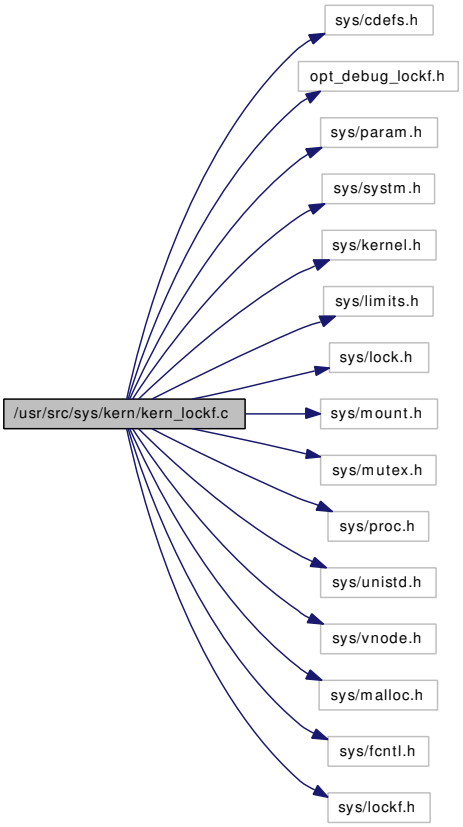
Definition at line 69 of file kern_lock.c.

Referenced by lockinit().

9.38 /usr/src/sys/kern/kern_lockf.c File Reference

```
#include <sys/cdefs.h>
#include "opt_debug_lockf.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/limits.h>
#include <sys/lock.h>
#include <sys/mount.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/unistd.h>
#include <sys/vnode.h>
#include <sys/malloc.h>
#include <sys/fcntl.h>
#include <sys/lockf.h>
```

Include dependency graph for kern_lockf.c:



Defines

- #define NOLOCKF (struct lockf ∗)0
- #define SELF 0x1
- #define OTHERS 0x2

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_lockf.c,v 1.54 2005/03/29 08:13:01 phk Exp $")
- MALLOC_DEFINE (M_LOCKF,"lockf","Byte-range locking structures")
- static int lf_clearlock (struct lockf ∗)
- static int lf_findoverlap (struct lockf ∗, struct lockf ∗, int, struct lockf ∗∗∗, struct lockf ∗∗)
- static struct lockf ∗ lf_getblock (struct lockf ∗)
- static int lf_getlock (struct lockf ∗, struct flock ∗)
- static int lf_setlock (struct lockf ∗)
- static void lf_split (struct lockf ∗, struct lockf ∗)
- static void lf_wakelock (struct lockf ∗)
- int lf_advlock (struct vop_advlock_args ∗ap, struct lockf ∗∗head, u_quad_t size)

Variables

- static int maxlockdepth = MAXDEPTH

9.38.1 Define Documentation

9.38.1.1 #define NOLOCKF (struct lockf ∗)0

Definition at line 73 of file kern_lockf.c.

Referenced by lf_clearlock(), lf_findoverlap(), lf_getblock(), lf_setlock(), and lf_wakelock().

9.38.1.2 #define OTHERS 0x2

Definition at line 75 of file kern_lockf.c.

Referenced by lf_findoverlap(), and lf_getblock().

9.38.1.3 #define SELF 0x1

Definition at line 74 of file kern_lockf.c.

Referenced by lf_clearlock(), lf_findoverlap(), and lf_setlock().

9.38.2 Function Documentation

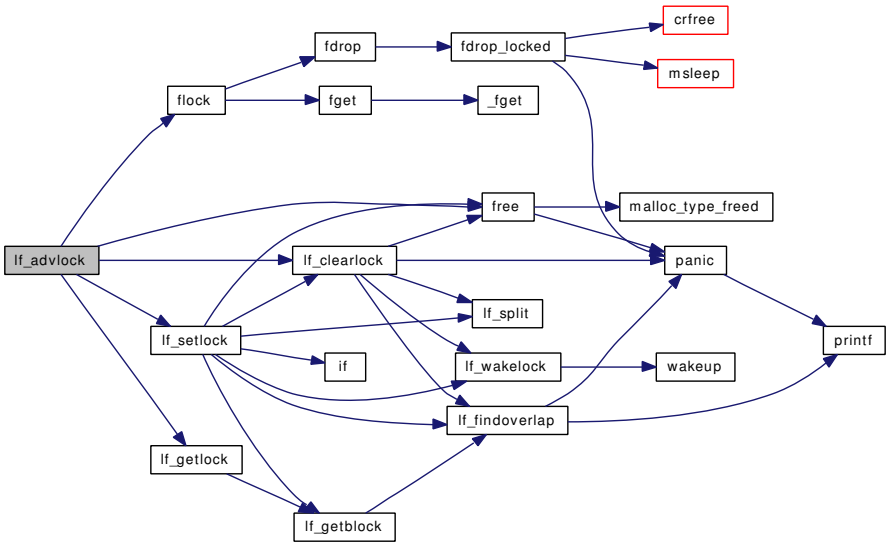
9.38.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_lockf. *c*, v 1.54 2005/03/29 08:13:01 phk Exp $")

9.38.2.2 int lf_advlock (struct vop_advlock_args ∗ *ap*, struct lockf ∗∗ *head*, u_quad_t *size*)

Definition at line 94 of file kern_lockf.c.

References flock(), free(), Giant, lf_clearlock(), lf_getlock(), and lf_setlock().

Here is the call graph for this function:



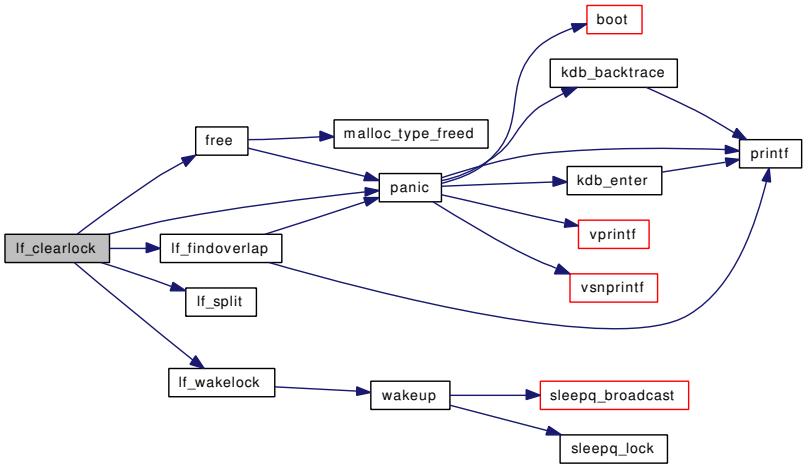
9.38.2.3 static int lf_clearlock (struct lockf *) [static]

Definition at line 474 of file kern_lockf.c.

References free(), lf_findoverlap(), lf_split(), lf_wakelock(), NOLOCKF, panic(), and SELF.

Referenced by lf_advlock(), and lf_setlock().

Here is the call graph for this function:



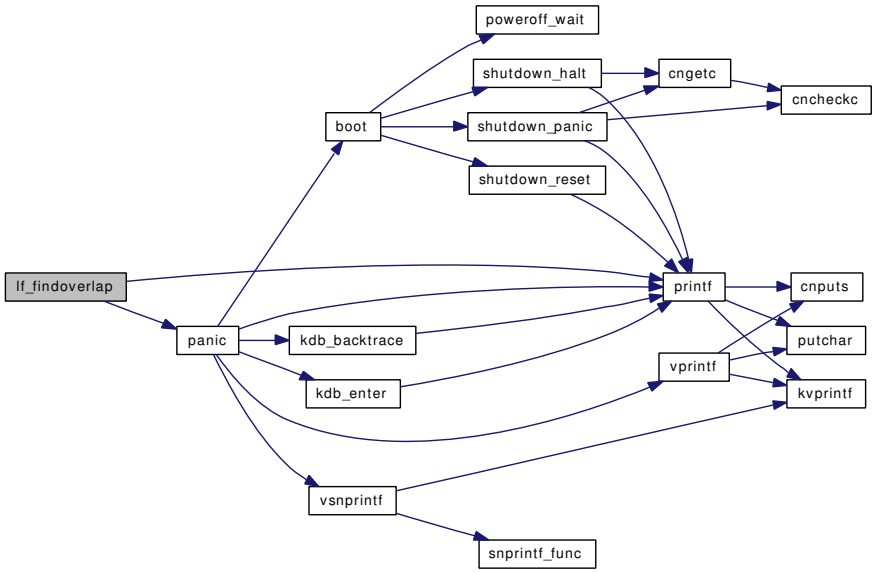
9.38.2.4 static int lf_findoverlap (struct lockf *, struct lockf *, int, struct lockf ***, struct lockf **) [static]

Definition at line 607 of file kern_lockf.c.

References NOLOCKF, OTHERS, panic(), printf(), and SELF.

Referenced by lf_clearlock(), lf_getblock(), and lf_setlock().

Here is the call graph for this function:



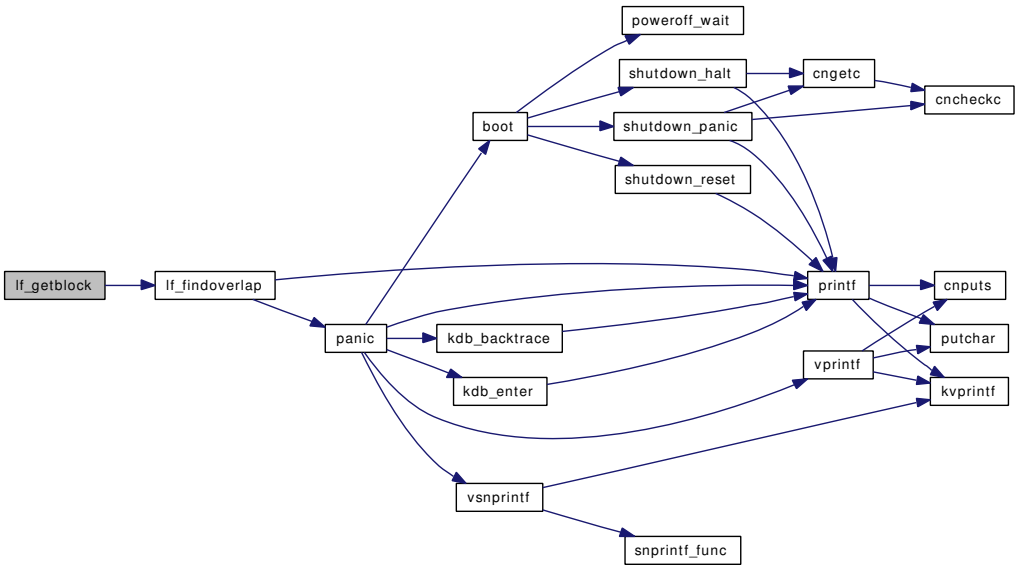
9.38.2.5 static struct lockf * lf_getblock (struct lockf *) [static]

Definition at line 577 of file kern_lockf.c.

References lf_findoverlap(), NOLOCKF, and OTHERS.

Referenced by lf_getlock(), and lf_setlock().

Here is the call graph for this function:



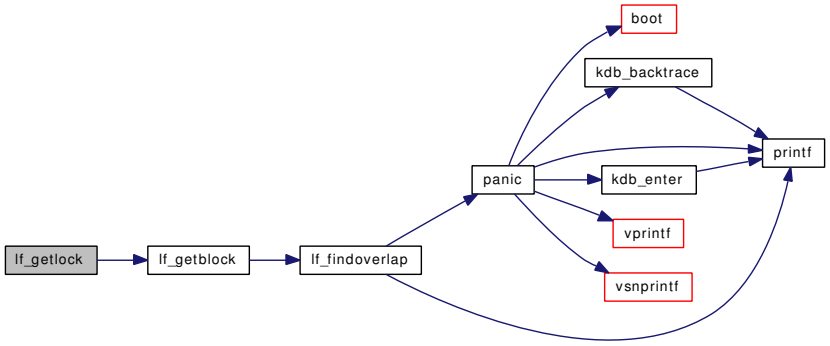
9.38.2.6 static int lf_getlock (struct lockf *, struct flock *) [static]

Definition at line 543 of file kern_lockf.c.

References lf_getblock().

Referenced by lf_advlock().

Here is the call graph for this function:



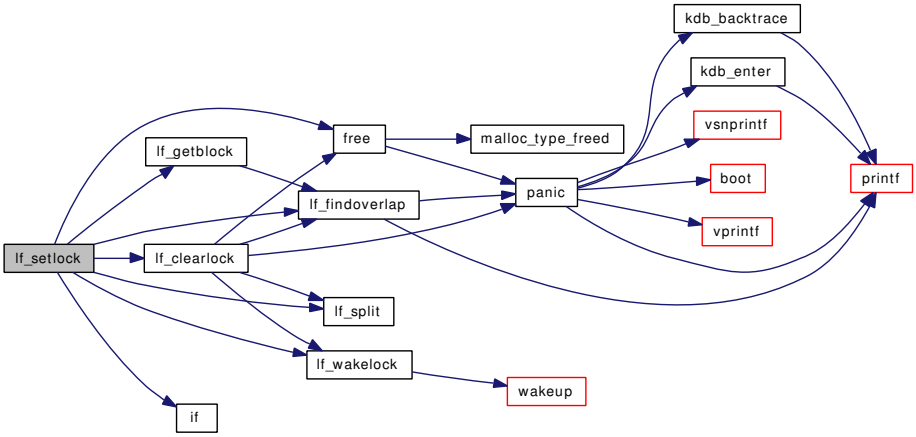
9.38.2.7 static int lf_setlock (struct lockf *) [static]

Definition at line 225 of file kern_lockf.c.

References free(), if(), lf_clearlock(), lf_findoverlap(), lf_getblock(), lf_split(), lf_wakelock(), maxlockdepth, NOLOCKF, sched_lock, SELF, and td.

Referenced by lf_advlock().

Here is the call graph for this function:



9.38.2.8 static void lf_split (struct lockf *, struct lockf *) [static]

Definition at line 717 of file kern_lockf.c.

Referenced by lf_clearlock(), and lf_setlock().

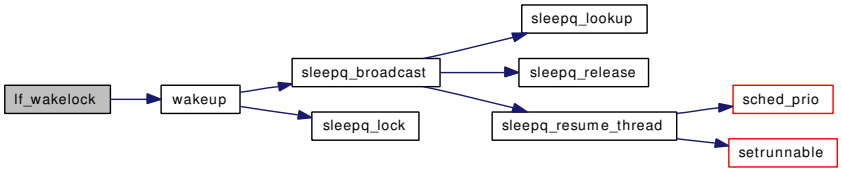
9.38.2.9 static void lf_wakelock (struct lockf *) [static]

Definition at line 764 of file kern_lockf.c.

References NOLOCKF, and wakeup().

Referenced by lf_clearlock(), and lf_setlock().

Here is the call graph for this function:



9.38.2.10 MALLOC_DEFINE (M_LOCKF, "lockf", "Byte-range locking structures")

9.38.3 Variable Documentation

9.38.3.1 int maxlockdepth = MAXDEPTH [static]

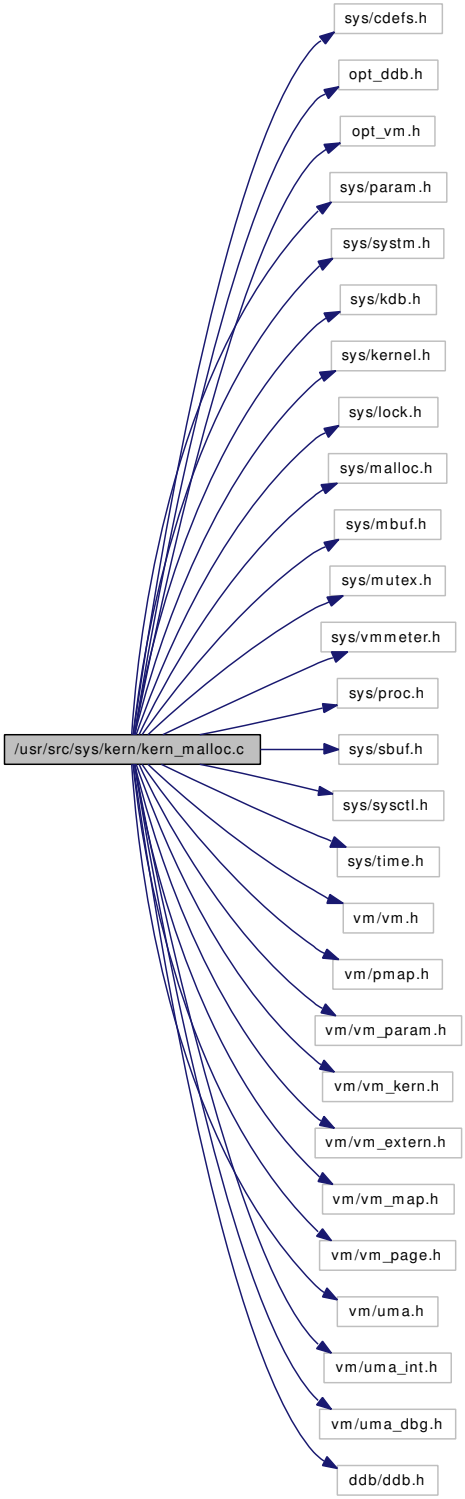
Definition at line 58 of file kern_lockf.c.

Referenced by lf_setlock().

9.39 /usr/src/sys/kern/kern_malloc.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ddb.h"
#include "opt_vm.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kdb.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mbuf.h>
#include <sys/mutex.h>
#include <sys/vmmeter.h>
#include <sys/proc.h>
#include <sys/sbuf.h>
#include <sys/sysctl.h>
#include <sys/time.h>
#include <vm/vm.h>
#include <vm/pmap.h>
#include <vm/vm_param.h>
#include <vm/vm_kern.h>
#include <vm/vm_extern.h>
#include <vm/vm_map.h>
#include <vm/vm_page.h>
#include <vm/uma.h>
#include <vm/uma_int.h>
#include <vm/uma_dbg.h>
#include <ddb/ddb.h>
```

Include dependency graph for kern_malloc.c:



Defines

- #define REALLOC_FRACTION 1
- #define KMEM_ZSHIFT 4

- #define KMEM_ZBASE 16
- #define KMEM_ZMASK (KMEM_ZBASE - 1)
- #define KMEM_ZMAX PAGE_SIZE
- #define KMEM_ZSIZE (KMEM_ZMAX >> KMEM_ZSHIFT)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_malloc.c,v 1.156 2006/10/26 10:17:13 rwatson Exp $")
- MALLOC_DEFINE (M_CACHE,"cache","Various Dynamically allocated caches")
- MALLOC_DEFINE (M_DEVBUF,"devbuf","device driver memory")
- MALLOC_DEFINE (M_TEMP,"temp","misc temporary data buffers")
- MALLOC_DEFINE (M_IP6OPT,"ip6opt","IPv6 options")
- MALLOC_DEFINE (M_IP6NDP,"ip6ndp","IPv6 Neighbor Discovery")
- static void kmeminit (void *)
- static MALLOC_DEFINE (M_FREE,"free","should be on free list")
- SYSCTL_UINT (_vm, OID_AUTO, kmem_size, CTLFLAG_RD,&vm_kmem_size, 0,"Size of kernel memory")
- SYSCTL_UINT (_vm, OID_AUTO, kmem_size_max, CTLFLAG_RD,&vm_kmem_size_max, 0,"Maximum size of kernel memory")
- SYSCTL_UINT (_vm, OID_AUTO, kmem_size_scale, CTLFLAG_RD,&vm_kmem_size_scale, 0,"Scale factor for kernel memory size")
- static int sysctl_kern_malloc_stats (SYSCTL_HANDLER_ARGS)
- int malloc_last_fail (void)
- static void malloc_type_zone_allocated (struct malloc_type *mtp, unsigned long size, int zindx)
- void malloc_type_allocated (struct malloc_type *mtp, unsigned long size)
- void malloc_type_freed (struct malloc_type *mtp, unsigned long size)
- void * malloc (unsigned long size, struct malloc_type *mtp, int flags)
- void free (void *addr, struct malloc_type *mtp)
- void * realloc (void *addr, unsigned long size, struct malloc_type *mtp, int flags)
- void * reallocf (void *addr, unsigned long size, struct malloc_type *mtp, int flags)
- void malloc_init (void *data)
- void malloc_uninit (void *data)
- malloc_type * malloc_desc2type (const char *desc)
- SYSCTL_PROC (_kern, OID_AUTO, malloc_stats, CTLFLAG_RD|CTLTYPE_STRUCT, 0, 0, sysctl_kern_malloc_stats,"s,malloc_type_ustats","Return malloc types")
- SYSCTL_INT (_kern, OID_AUTO, malloc_count, CTLFLAG_RD,&kmemcount, 0,"Count of kernel malloc types")

Variables

- static struct malloc_type * kmemstatistics
- static char * kmembase
- static char * kmemlimit
- static int kmemcount
- static u_int8_t kmemsize [KMEM_ZSIZE+1]
- struct {
  int kz_size
  char * kz_name
  uma_zone_t kz_zone
  } kmemzones [ ]

- static uma_zone_t mt_zone
- u_int vm_kmem_size
- u_int vm_kmem_size_max
- u_int vm_kmem_size_scale
- mtx malloc_mtx
- static time_t t_malloc_fail

9.39.1 Define Documentation

9.39.1.1 #define KMEM_ZBASE 16

Definition at line 120 of file kern_malloc.c.

Referenced by kmeminit(), and malloc().

9.39.1.2 #define KMEM_ZMASK (KMEM_ZBASE - 1)

Definition at line 121 of file kern_malloc.c.

Referenced by malloc().

9.39.1.3 #define KMEM_ZMAX PAGE_SIZE

Definition at line 123 of file kern_malloc.c.

Referenced by malloc().

9.39.1.4 #define KMEM_ZSHIFT 4

Definition at line 119 of file kern_malloc.c.

Referenced by kmeminit(), and malloc().

9.39.1.5 #define KMEM_ZSIZE (KMEM_ZMAX >> KMEM_ZSHIFT)

Definition at line 124 of file kern_malloc.c.

9.39.1.6 #define REALLOC_FRACTION 1

Definition at line 96 of file kern_malloc.c.

Referenced by realloc().

9.39.2 Function Documentation

9.39.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_malloc. *c*, v 1.156 2006/10/26 10:17:13 rwatson Exp $")

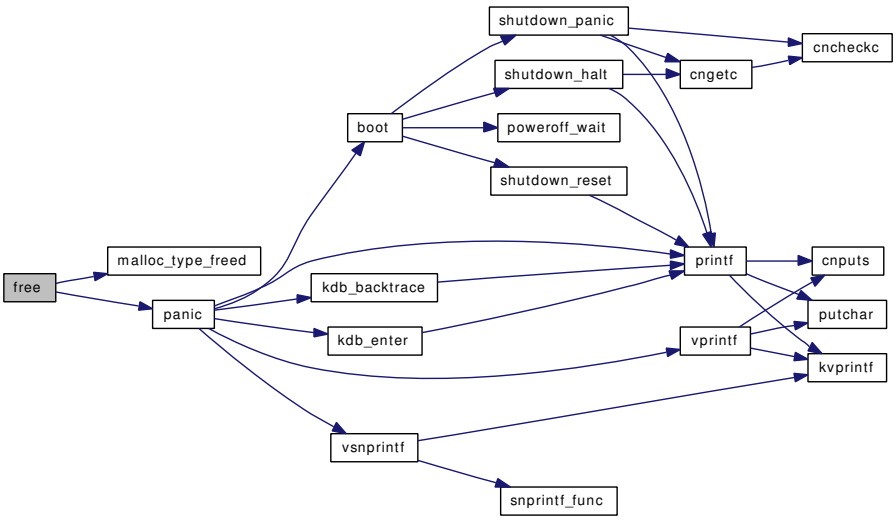
9.39.2.2 void free (void * *addr*, struct malloc_type * *mtp*)

Definition at line 391 of file kern_malloc.c.

References malloc_type_freed(), and panic().

Referenced by __mnt_vnode_first(), __mnt_vnode_markerfree(), accept1(), allocbuf(), alq_close(), bind(), blist_destroy(), cblock_free_cblocks(), cf_get_method(), cf_levels_method(), clone_cleanup(), cluster_write(), connect(), constty_clear(), copyiniov(), copyinuio(), coredump(), cpufreq_detach(), cpufreq_dup_set(), cpufreq_levels_sysctl(), cpufreq_settings_sysctl(), cpufreq_unregister(), devadded(), devaddq(), devclass_add_device(), devclass_alloc_unit(), devclass_delete_device(), device_set_desc_internal(), device_set_driver(), device_set_softc(), device_sysctl_handler(), devread(), devremoved(), do_sendfile(), eventhandler_deregister(), eventhandler_prune_list(), exec_register(), exec_unregister(), expand_name(), fdgrowtable(), fini_cdevsw(), free_mntarg(), freebsd_fixup(), freeenv(), getpeername1(), getsockname1(), hashdestroy(), huft_free(), inflate(), int_rman_release_resource(), intr_event_add_handler(), intr_event_destroy(), intr_event_remove_handler(), ioctl(), ithread_loop(), itimers_event_hook_exit(), kdb_sysctl_available(), kenv(), kern___getcwd(), kern_alternate_path(), kern_getpeername(), kern_getsockname(), kern_ptrace(), kern_select(), kldfind(), kldload(), kldsym(), kobj_class_compile(), kobj_class_free(), kobj_delete(), kqueue_expand(), lf_advlock(), lf_clearlock(), lf_setlock(), lim_free(), link_elf_load_file(), link_elf_lookup_set(), link_elf_unload_file(), linker_file_add_dependency(), linker_file_unload(), linker_find_file_by_name(), linker_hints_lookup(), linker_load_module(), linker_lookup_file(), log_console(), m_tag_free_default(), mbp_alloc_page(), mbp_destroy(), mi_startup(), module_release(), mount(), msgunload(), nmount(), poll(), preadv(), prep_cdevsw(), pstats_free(), pwritev(), readv(), realloc(), reallocf(), recvmsg(), resource_list_delete(), resource_list_free(), resource_list_purge(), rman_fini(), rman_manage_region(), rman_reserve_resource_bound(), root_mount_rel(), sctp_generic_sendmsg(), sem_create(), sem_enter(), sem_forkhook(), sem_free(), sem_leave(), semop(), semunload(), sendmsg(), setenv(), shmexit_myhook(), shmrealloc(), shmunload(), sigacts_free(), sleepq_free(), stack_destroy(), sysctl_ctx_entry_del(), sysctl_ctx_free(), sysctl_devctl_disable(), sysctl_handle_string(), sysctl_jail_list(), sysctl_kern_malloc_stats(), sysctl_kern_proc_pathname(), sysctl_remove_oid(), sysctl_sysctl_name2oid(), SYSINIT(), sysinit_add(), taskqueue_free(), ttyrel(), turnstile_free(), uifind(), uipc_bind(), umtx_pi_unref(), umtxq_free(), unmount(), unp_connect(), unp_gc(), unp_pcblist(), unsetenv(), utrace(), uuidgen(), vfs_export(), vfs_free_addrlist(), vfs_free_netcred(), vfs_freeopt(), vfs_freeopts(), vfs_mountroot_try(), vfs_read_dirent(), vn_fullpath(), and writev().

Here is the call graph for this function:

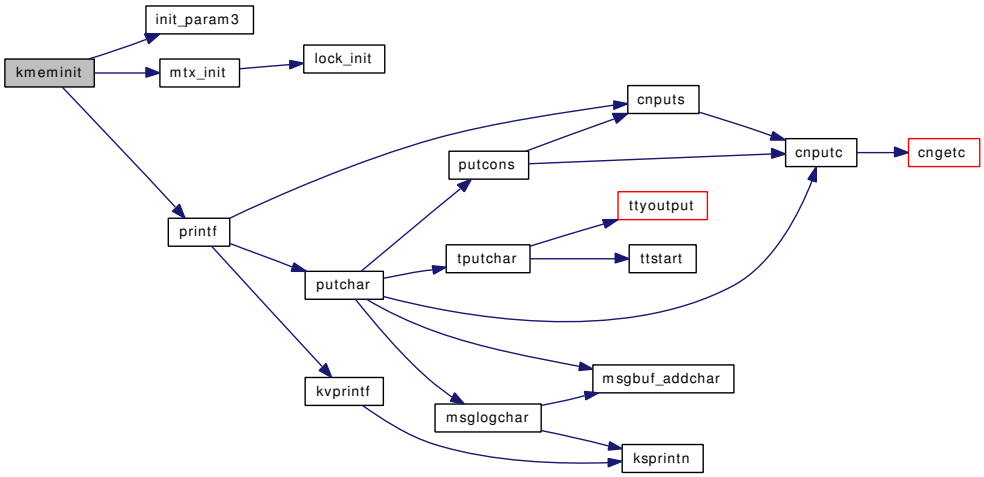


9.39.2.3 static void kmeminit (void *) [static]

Definition at line 529 of file kern_malloc.c.

References init_param3(), KMEM_ZBASE, KMEM_ZSHIFT, kmemzones, kz_size, mtx_init(), nmbclusters, and printf().

Here is the call graph for this function:



9.39.2.4 void* malloc (unsigned long *size*, struct malloc_type * *mtp*, int *flags*)

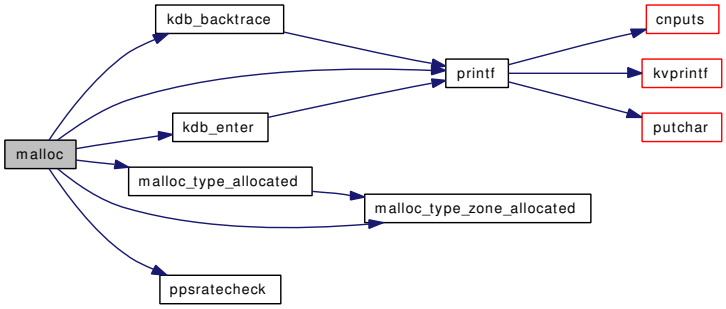
Definition at line 294 of file kern_malloc.c.

References kdb_backtrace(), kdb_enter(), KMEM_ZBASE, KMEM_ZMASK, KMEM_ZMAX, KMEM_ZSHIFT, kmemzones, malloc_type_allocated(), malloc_type_zone_allocated(), ppsratecheck(), printf(), and time_uptime.

Referenced by __mnt_vnode_first(), _taskqueue_create(), allocbuf(), alq_open(), blist_create(), cblock_alloc_cblocks(), cf_get_method(), cf_levels_method(), clone_setup(), cloneuio(), cluster_collectbufs(), constty_set(), copyiniov(), copyinuio(), coredump(), cpufreq_curr_sysctl(), cpufreq_dup_set(), cpufreq_insert_abs(), cpufreq_levels_sysctl(), cpufreq_settings_sysctl(), devadded(), devaddq(), devclass_add_device(), devclass_add_driver(), devclass_alloc_unit(), devclass_find_internal(), devclass_get_devices(), devclass_get_drivers(), devctl_notify(), devctl_queue_data(), devfs_first(), device_get_children(), device_set_desc_internal(), device_set_driver(), device_sysctl_handler(), devremoved(), devstat_alloc(), exec_register(), exec_unregister(), expand_name(), fdinit(), getenv(), hashinit_flags(), huft_build(), inflate(), init_dynamic_kenv(), int_alloc_resource(), intr_event_add_handler(), intr_event_create(), ioctl(), ithread_create(), itimers_alloc(), kdb_sysctl_available(), kenv(), kern___getcwd(), kern_alternate_path(), kern_getfsstat(), kern_ptrace(), kern_select(), kern_semctl(), kern_shmat(), kldfind(), kldload(), kldsym(), kobj_class_compile(), kobj_create(), kqueue(), lim_alloc(), link_elf_link_common_finish(), link_elf_load_file(), link_elf_lookup_set(), linker_file_add_dependency(), linker_file_lookup_symbol_internal(), linker_find_file_by_name(), linker_hints_lookup(), linker_lookup_file(), linker_strdup(), LIST_HEAD(), log_console(), m_tag_alloc(), make_device(), mbp_create(), modlist_newmodule(), module_register(), mount(), mount_arg(), mount_argf(), mount_argsu(), msginit(), phashinit(), poll(), prep_cdevsw(), pstats_alloc(), ptyinit(), realloc(), resource_list_add(), rman_init(), rman_reserve_resource_bound(), sem_create(), sem_enter(), sem_forkhook(), seminit(), semop(), setenv(), shmfork_myhook(), shminit(), shmrealloc(), sigacts_alloc(), sleepq_alloc(), sodupsockaddr(), stack_create(), sysctl_add_oid(), sysctl_ctx_entry_add(), sysctl_handle_string(), sysctl_jail_list(), sysctl_kern_malloc_stats(), sysctl_sysctl_name2oid(), SYSINIT(), sysinit_add(), taskqueue_start_threads(), ttyalloc(), turnstile_alloc(), uifind(), uipc_accept(), uipc_bind(), uipc_peeraddr(), uipc_sockaddr(), umtxq_alloc(), unmount(), unp_connect(), unp_gc(), unp_pcblist(), utrace(), uuidgen(), vfs_buildopts(), vfs_donmount(), vfs_export(), vfs_hang_-

addrlist(), vfs_mergeopts(), vfs_mountroot_try(), and vn_fullpath().

Here is the call graph for this function:



9.39.2.5 static MALLOC_DEFINE (M_FREE, "free", "should be on free list") [static]

9.39.2.6 MALLOC_DEFINE (M_IP6NDP, "ip6ndp", "IPv6 Neighbor Discovery")

9.39.2.7 MALLOC_DEFINE (M_IP6OPT, "ip6opt", "IPv6 options")

9.39.2.8 MALLOC_DEFINE (M_TEMP, "temp", "misc temporary data buffers")

9.39.2.9 MALLOC_DEFINE (M_DEVBUF, "devbuf", "device driver memory")

9.39.2.10 MALLOC_DEFINE (M_CACHE, "cache", "Various Dynamically allocated caches")

9.39.2.11 struct malloc_type* malloc_desc2type (const char * *desc*)

Definition at line 702 of file kern_malloc.c.

9.39.2.12 void malloc_init (void * *data*)

Definition at line 636 of file kern_malloc.c.

9.39.2.13 int malloc_last_fail (void)

Definition at line 226 of file kern_malloc.c.

References time_uptime.

9.39.2.14 void malloc_type_allocated (struct malloc_type * *mtp*, unsigned long *size*)

Definition at line 258 of file kern_malloc.c.

References malloc_type_zone_allocated().

Referenced by malloc().

Here is the call graph for this function:

9.39.2.15 void malloc_type_freed (struct malloc_type * *mtp*, unsigned long *size*)

Definition at line 272 of file kern_malloc.c.

Referenced by free().

9.39.2.16 static void malloc_type_zone_allocated (struct malloc_type * *mtp*, unsigned long *size*, int *zindx*) [static]

Definition at line 239 of file kern_malloc.c.

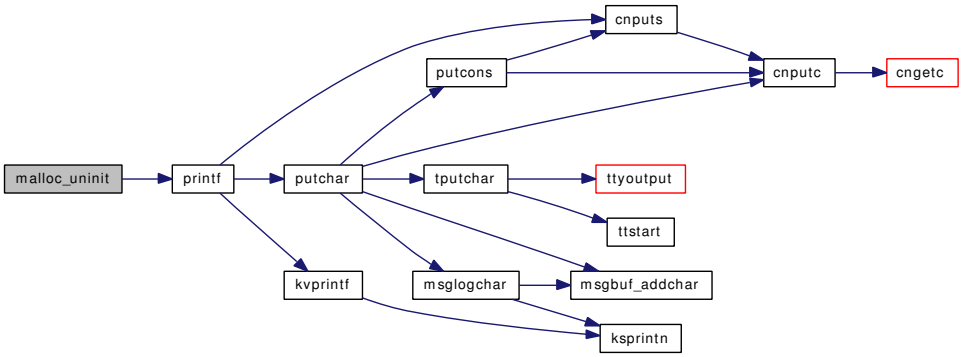
Referenced by malloc(), and malloc_type_allocated().

9.39.2.17 void malloc_uninit (void * *data*)

Definition at line 655 of file kern_malloc.c.

References printf().

Here is the call graph for this function:



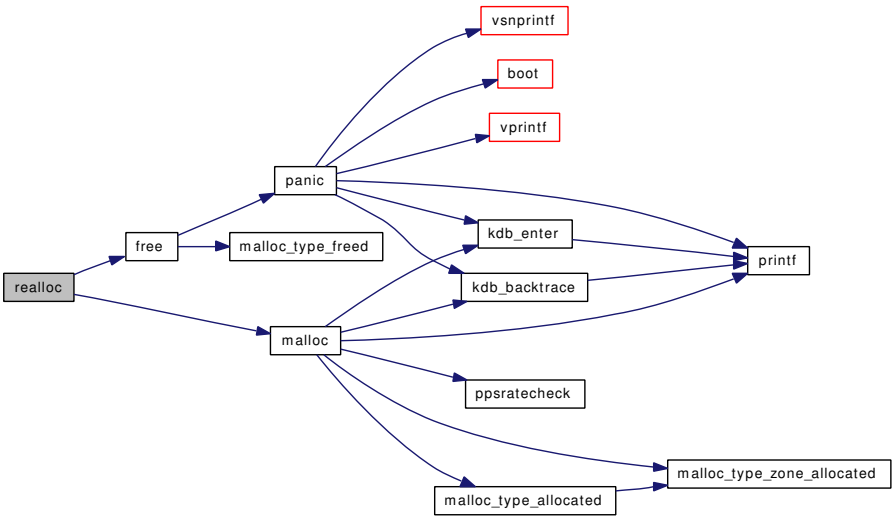
9.39.2.18 void* realloc (void * *addr*, unsigned long *size*, struct malloc_type * *mtp*, int *flags*)

Definition at line 453 of file kern_malloc.c.

References free(), malloc(), and REALLOC_FRACTION.

Referenced by mount_arg(), mount_argf(), reallocf(), and vfs_read_dirent().

Here is the call graph for this function:

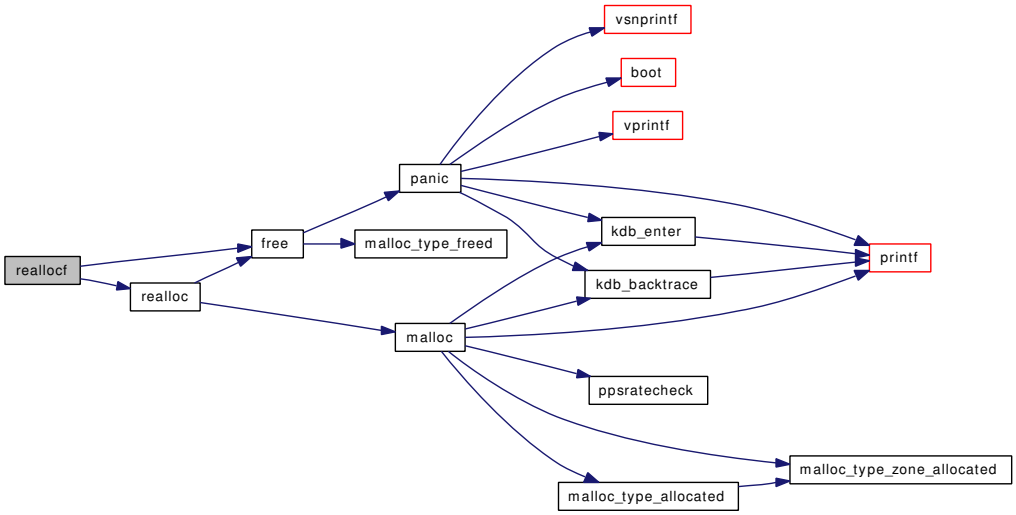


9.39.2.19 void* reallocf (void * *addr*, unsigned long *size*, struct malloc_type * *mtp*, int *flags*)

Definition at line 515 of file kern_malloc.c.

References free(), and realloc().

Here is the call graph for this function:



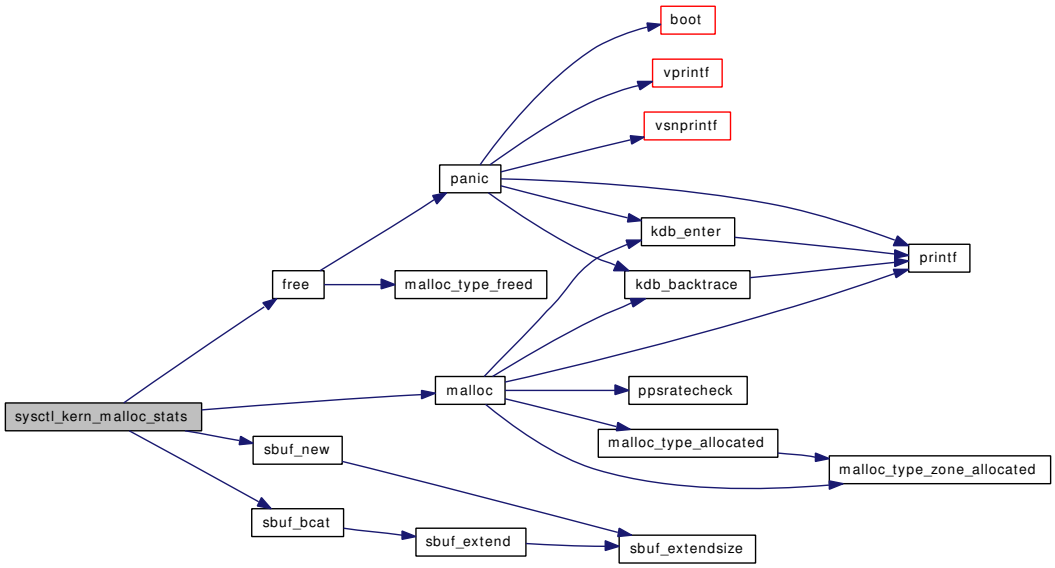
9.39.2.20 SYSCTL_INT (_kern, OID_AUTO, malloc_count, CTLFLAG_RD, & *kmemcount*, 0, "Count of kernel malloc types")

9.39.2.21 static int sysctl_kern_malloc_stats (SYSCTL_HANDLER_ARGS) [static]

Definition at line 715 of file kern_malloc.c.

References free(), malloc(), sbuf_bcat(), and sbuf_new().

Here is the call graph for this function:



9.39.2.22 SYSCTL_PROC (_kern, OID_AUTO, malloc_stats, CTLFLAG_RD| *CTLTYPE_STRUCT*, 0, 0, sysctl_kern_malloc_stats, " *s*, malloc_type_ustats", "Return malloc types")

9.39.2.23 SYSCTL_UINT (_vm, OID_AUTO, kmem_size_scale, CTLFLAG_RD, & *vm_kmem_size_scale*, 0, "Scale factor for kernel memory size")

9.39.2.24 SYSCTL_UINT (_vm, OID_AUTO, kmem_size_max, CTLFLAG_RD, & *vm_kmem_size_max*, 0, "Maximum size of kernel memory")

9.39.2.25 SYSCTL_UINT (_vm, OID_AUTO, kmem_size, CTLFLAG_RD, & *vm_kmem_size*, 0, "Size of kernel memory")

9.39.3 Variable Documentation

9.39.3.1 char* kmembase [static]

Definition at line 115 of file kern_malloc.c.

9.39.3.2 int kmemcount [static]

Definition at line 117 of file kern_malloc.c.

9.39.3.3 char* kmemlimit [static]

Definition at line 116 of file kern_malloc.c.

9.39.3.4 u_int8_t kmemsize[KMEM_ZSIZE+1] [static]

Definition at line 125 of file kern_malloc.c.

9.39.3.5 struct malloc_type* kmemstatistics [static]

Definition at line 114 of file kern_malloc.c.

9.39.3.6 struct { ... } kmemzones[ ]

Referenced by kmeminit(), and malloc().

9.39.3.7 char* kz_name

Definition at line 137 of file kern_malloc.c.

9.39.3.8 int kz_size

Definition at line 136 of file kern_malloc.c.

Referenced by kmeminit().

9.39.3.9 uma_zone_t kz_zone

Definition at line 138 of file kern_malloc.c.

9.39.3.10 struct mtx malloc_mtx

Definition at line 192 of file kern_malloc.c.

9.39.3.11 uma_zone_t mt_zone [static]

Definition at line 175 of file kern_malloc.c.

9.39.3.12 time_t t_malloc_fail [static]

Definition at line 205 of file kern_malloc.c.

9.39.3.13 u_int vm_kmem_size

Definition at line 177 of file kern_malloc.c.

Referenced by vntblinit().

9.39.3.14 u_int vm_kmem_size_max

Definition at line 181 of file kern_malloc.c.

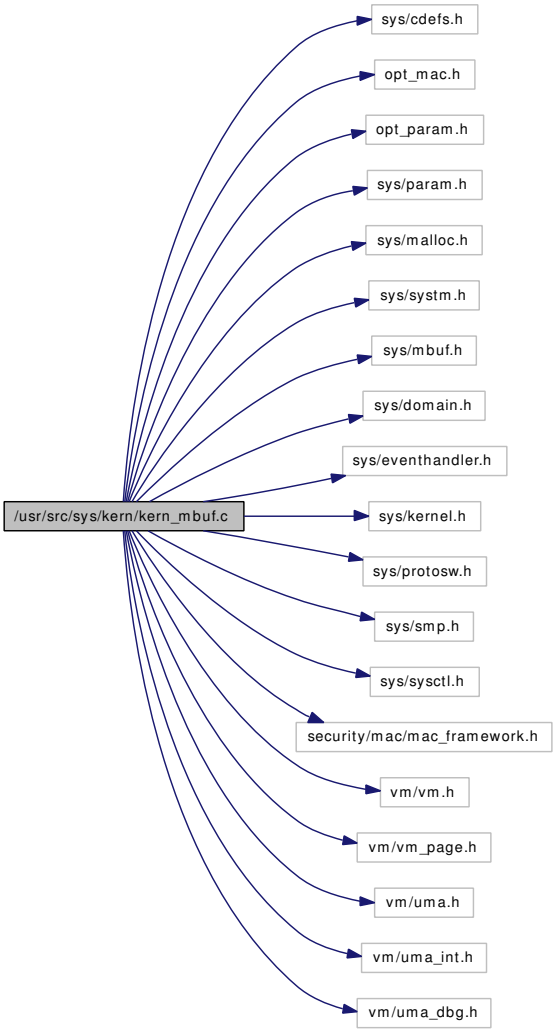
9.39.3.15 u_int vm_kmem_size_scale

Definition at line 185 of file kern_malloc.c.

9.40 /usr/src/sys/kern/kern_mbuf.c File Reference

```
#include <sys/cdefs.h>
#include "opt_mac.h"
#include "opt_param.h"
#include <sys/param.h>
#include <sys/malloc.h>
#include <sys/systm.h>
#include <sys/mbuf.h>
#include <sys/domain.h>
#include <sys/eventhandler.h>
#include <sys/kernel.h>
#include <sys/protosw.h>
#include <sys/smp.h>
#include <sys/sysctl.h>
#include <security/mac/mac_framework.h>
#include <vm/vm.h>
#include <vm/vm_page.h>
#include <vm/uma.h>
#include <vm/uma_int.h>
#include <vm/uma_dbg.h>
```

Include dependency graph for kern_mbuf.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_mbuf.c,v 1.28 2007/01/25 01:05:23 mohans Exp $")
- static void tunable_mbinit (void *dummy)
- SYSINIT (tunable_mbinit, SI_SUB_TUNABLES, SI_ORDER_ANY, tunable_mbinit, NULL)
- static int sysctl_nmbclusters (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern_ipc, OID_AUTO, nmbclusters, CTLTYPE_INT|CTLFLAG_RW,&nmbclusters, 0, sysctl_nmbclusters,"IU","Maximum number of mbuf clusters allowed")
- SYSCTL_INT (_kern_ipc, OID_AUTO, nmbjumbop, CTLFLAG_RW,&nmbjumbop, 0,"Maximum number of mbuf page size jumbo clusters allowed")
- SYSCTL_INT (_kern_ipc, OID_AUTO, nmbjumbo9, CTLFLAG_RW,&nmbjumbo9, 0,"Maximum number of mbuf 9k jumbo clusters allowed")
- SYSCTL_INT (_kern_ipc, OID_AUTO, nmbjumbo16, CTLFLAG_RW,&nmbjumbo16, 0,"Maximum number of mbuf 16k jumbo clusters allowed")
- SYSCTL_STRUCT (_kern_ipc, OID_AUTO, mbstat, CTLFLAG_RD,&mbstat, mbstat,"Mbuf general information and statistics")
- static int mb_ctor_mbuf (void *, int, void *, int)
- static int mb_ctor_clust (void *, int, void *, int)

- static int mb_ctor_pack (void *, int, void *, int)
- static void mb_dtor_mbuf (void *, int, void *)
- static void mb_dtor_clust (void *, int, void *)
- static void mb_dtor_pack (void *, int, void *)
- static int mb_zinit_pack (void *, int, int)
- static void mb_zfini_pack (void *, int)
- static void mb_reclaim (void *)
- static void mbuf_init (void *)
- CTASSERT ((((MSIZE-1)^MSIZE)+1) >> 1==MSIZE)

Variables

- int nmbclusters
- int nmbjumbop
- int nmbjumbo9
- int nmbjumbo16
- mbstat mbstat
- uma_zone_t zone_mbuf
- uma_zone_t zone_clust
- uma_zone_t zone_pack
- uma_zone_t zone_jumbop
- uma_zone_t zone_jumbo9
- uma_zone_t zone_jumbo16
- uma_zone_t zone_ext_refcnt

9.40.1 Function Documentation

9.40.1.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_mbuf. *c*, v 1.28 2007/01/25 01:05:23 mohans Exp $")

9.40.1.2 CTASSERT ((((MSIZE-1)^MSIZE)+1) >> *1* = =MSIZE)

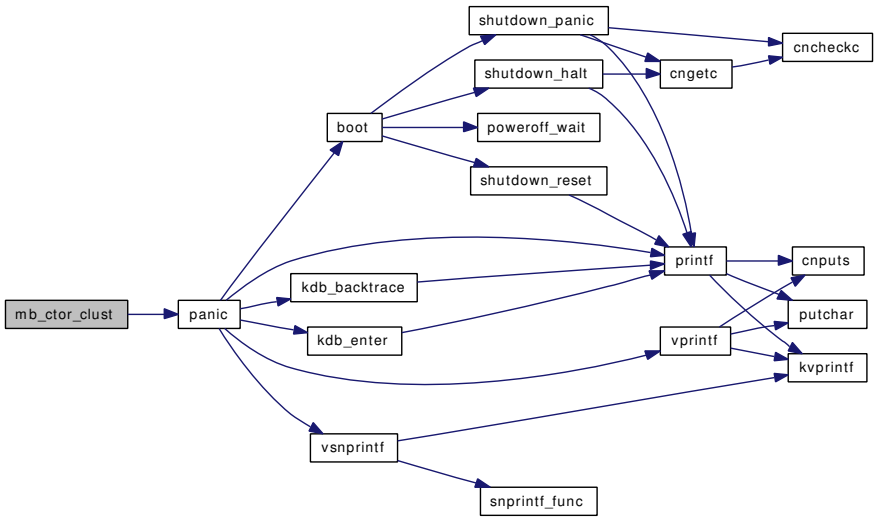
9.40.1.3 static int mb_ctor_clust (void *, int, void *, int) [static]

Definition at line 394 of file kern_mbuf.c.

References panic(), and zone_clust.

Referenced by mbuf_init().

Here is the call graph for this function:



9.40.1.4 static int mb_ctor_mbuf (void *, int, void *, int) [static]

Definition at line 284 of file kern_mbuf.c.

Referenced by mbuf_init().

9.40.1.5 static int mb_ctor_pack (void *, int, void *, int) [static]

Definition at line 498 of file kern_mbuf.c.

Referenced by mbuf_init().

9.40.1.6 static void mb_dtor_clust (void *, int, void *) [static]

Definition at line 444 of file kern_mbuf.c.

References zone_clust.

Referenced by mbuf_init().

9.40.1.7 static void mb_dtor_mbuf (void *, int, void *) [static]

Definition at line 339 of file kern_mbuf.c.

References m_tag_delete_chain().

Referenced by mbuf_init().

Here is the call graph for this function:

9.40.1.8 static void mb_dtor_pack (void *, int, void *) [static]

Definition at line 356 of file kern_mbuf.c.

References m_tag_delete_chain(), zone_clust, and zone_pack.

Referenced by mbuf_init().

Here is the call graph for this function:

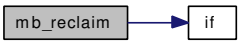
9.40.1.9 static void mb_reclaim (void *) [static]

Definition at line 552 of file kern_mbuf.c.

References if(), and pr.

Referenced by mbuf_init().

Here is the call graph for this function:



9.40.1.10 static void mb_zfini_pack (void *, int) [static]

Definition at line 480 of file kern_mbuf.c.

References zone_clust.

Referenced by mbuf_init().

9.40.1.11 static int mb_zinit_pack (void *, int, int) [static]

Definition at line 460 of file kern_mbuf.c.

References zone_clust.

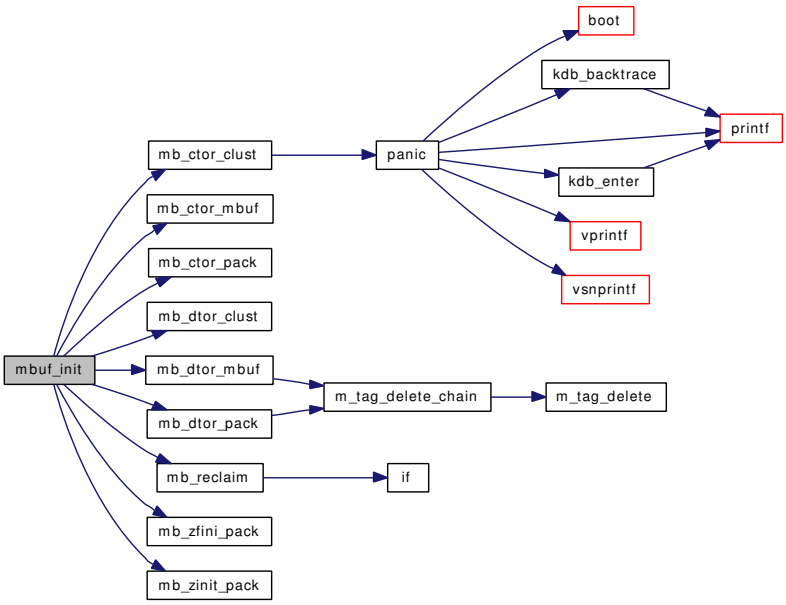
Referenced by mbuf_init().

9.40.1.12 static void mbuf_init (void *) [static]

Definition at line 178 of file kern_mbuf.c.

References mb_ctor_clust(), mb_ctor_mbuf(), mb_ctor_pack(), mb_dtor_clust(), mb_dtor_mbuf(), mb_dtor_pack(), mb_reclaim(), mb_zfini_pack(), mb_zinit_pack(), mbstat, nmbclusters, nmbjumbo16, nmbjumbo9, nmbjumbop, zone_clust, zone_ext_refcnt, zone_jumbo16, zone_jumbo9, zone_jumbop, zone_mbuf, and zone_pack.

Here is the call graph for this function:



9.40.1.13 SYSCTL_INT (_kern_ipc, OID_AUTO, nmbjumbo16, CTLFLAG_RW, & *nmbjumbo16*, 0, "Maximum number of mbuf 16k jumbo clusters allowed")

9.40.1.14 SYSCTL_INT (_kern_ipc, OID_AUTO, nmbjumbo9, CTLFLAG_RW, & *nmbjumbo9*, 0, "Maximum number of mbuf 9k jumbo clusters allowed")

9.40.1.15 SYSCTL_INT (_kern_ipc, OID_AUTO, nmbjumbop, CTLFLAG_RW, & *nmbjumbop*, 0, "Maximum number of mbuf page size jumbo clusters allowed")

9.40.1.16 static int sysctl_nmbclusters (SYSCTL_HANDLER_ARGS) [static]

Definition at line 116 of file kern_mbuf.c.

References nmbclusters, sysctl_handle_int(), and zone_clust.

Here is the call graph for this function:

9.40.1.17 SYSCTL_PROC (_kern_ipc, OID_AUTO, nmbclusters, CTLTYPE_INT| *CTLFLAG_RW*, & *nmbclusters*, 0, sysctl_nmbclusters, "IU", "Maximum number of mbuf clusters allowed")

9.40.1.18 SYSCTL_STRUCT (_kern_ipc, OID_AUTO, mbstat, CTLFLAG_RD, & *mbstat*, mbstat, "Mbuf general information and statistics")

9.40.1.19 SYSINIT (tunable_mbinit, SI_SUB_TUNABLES, SI_ORDER_ANY, tunable_mbinit, NULL)

9.40.1.20 static void tunable_mbinit (void * *dummy*) [static]

Definition at line 105 of file kern_mbuf.c.

References maxusers, and nmbclusters.

9.40.2 Variable Documentation

9.40.2.1 struct mbstat mbstat

Definition at line 102 of file kern_mbuf.c.

Referenced by kern_sendfile(), m_copym(), m_copypacket(), m_dup(), m_pullup(), and mbuf_init().

9.40.2.2 int nmbclusters

Definition at line 98 of file kern_mbuf.c.

Referenced by init_maxsockets(), kmeminit(), mbuf_init(), sysctl_nmbclusters(), and tunable_mbinit().

9.40.2.3 int nmbjumbo16

Definition at line 101 of file kern_mbuf.c.

Referenced by mbuf_init().

9.40.2.4 int nmbjumbo9

Definition at line 100 of file kern_mbuf.c.

Referenced by mbuf_init().

9.40.2.5 int nmbjumbop

Definition at line 99 of file kern_mbuf.c.

Referenced by mbuf_init().

9.40.2.6 uma_zone_t zone_clust

Definition at line 148 of file kern_mbuf.c.

Referenced by mb_ctor_clust(), mb_dtor_clust(), mb_dtor_pack(), mb_free_ext(), mb_zfini_pack(), mb_-zinit_pack(), mbuf_init(), and sysctl_nmbclusters().

9.40.2.7 uma_zone_t zone_ext_refcnt

Definition at line 153 of file kern_mbuf.c.

Referenced by m_extadd(), mb_free_ext(), and mbuf_init().

9.40.2.8 uma_zone_t zone_jumbo16

Definition at line 152 of file kern_mbuf.c.

Referenced by mb_free_ext(), and mbuf_init().

9.40.2.9 uma_zone_t zone_jumbo9

Definition at line 151 of file kern_mbuf.c.

Referenced by mb_free_ext(), and mbuf_init().

9.40.2.10 uma_zone_t zone_jumbop

Definition at line 150 of file kern_mbuf.c.

Referenced by mb_free_ext(), and mbuf_init().

9.40.2.11 uma_zone_t zone_mbuf

Definition at line 147 of file kern_mbuf.c.

Referenced by mb_free_ext(), and mbuf_init().

9.40.2.12 uma_zone_t zone_pack

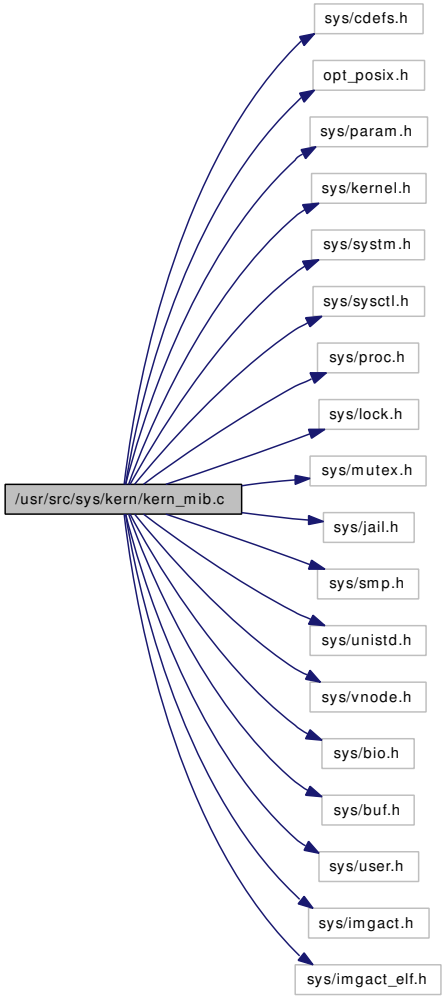
Definition at line 149 of file kern_mbuf.c.

Referenced by mb_dtor_pack(), mb_free_ext(), and mbuf_init().

9.41 /usr/src/sys/kern/kern_mib.c File Reference

```
#include <sys/cdefs.h>
#include "opt_posix.h"
#include <sys/param.h>
#include <sys/kernel.h>
#include <sys/systm.h>
#include <sys/sysctl.h>
#include <sys/proc.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/jail.h>
#include <sys/smp.h>
#include <sys/unistd.h>
#include <sys/vnode.h>
#include <sys/bio.h>
#include <sys/buf.h>
#include <sys/user.h>
#include <sys/imgact.h>
#include <sys/imgact_elf.h>
```

Include dependency graph for kern_mib.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_mib.c,v 1.76 2005/08/21 18:03:31 pjd Exp $")
- SYSCTL_NODE (, 0, sysctl, CTLFLAG_RW, 0,"Sysctl internal magic")
- SYSCTL_NODE (, CTL_KERN, kern, CTLFLAG_RW, 0,"High kernel, proc, limits &c")
- SYSCTL_NODE (, CTL_VM, vm, CTLFLAG_RW, 0,"Virtual memory")
- SYSCTL_NODE (, CTL_VFS, vfs, CTLFLAG_RW, 0,"File system")
- SYSCTL_NODE (, CTL_NET, net, CTLFLAG_RW, 0,"Network, (see socket.h)")
- SYSCTL_NODE (, CTL_DEBUG, debug, CTLFLAG_RW, 0,"Debugging")
- SYSCTL_NODE (_debug, OID_AUTO, sizeof, CTLFLAG_RW, 0,"Sizeof various things")
- SYSCTL_NODE (, CTL_HW, hw, CTLFLAG_RW, 0,"hardware")
- SYSCTL_NODE (, CTL_MACHDEP, machdep, CTLFLAG_RW, 0,"machine dependent")
- SYSCTL_NODE (, CTL_USER, user, CTLFLAG_RW, 0,"user-level")
- SYSCTL_NODE (, CTL_P1003_1B, p1003_1b, CTLFLAG_RW, 0,"p1003_1b, (see p1003_1b.h)")
- SYSCTL_NODE (, OID_AUTO, compat, CTLFLAG_RW, 0,"Compatibility code")
- SYSCTL_NODE (, OID_AUTO, security, CTLFLAG_RW, 0,"Security")
- SYSCTL_STRING (_kern, OID_AUTO, ident, CTLFLAG_RD, kern_ident, 0,"Kernel identifier")
- SYSCTL_STRING (_kern, KERN_OSRELEASE, osrelease, CTLFLAG_RD, osrelease, 0,"Operating system release")

- SYSCTL_INT (_kern, KERN_OSREV, osrevision, CTLFLAG_RD, 0, BSD,"Operating system revision")
- SYSCTL_STRING (_kern, KERN_VERSION, version, CTLFLAG_RD, version, 0,"Kernel version")
- SYSCTL_STRING (_kern, KERN_OSTYPE, ostype, CTLFLAG_RD, ostype, 0,"Operating system type")
- SYSCTL_INT (_kern, KERN_OSRELDATE, osreldate, CTLFLAG_RD,&osreldate, 0,"Kernel release date")
- SYSCTL_INT (_kern, KERN_MAXPROC, maxproc, CTLFLAG_RDTUN,&maxproc, 0,"Maximum number of processes")
- SYSCTL_INT (_kern, KERN_MAXPROCPERUID, maxprocperuid, CTLFLAG_RW,&maxprocperuid, 0,"Maximum processes allowed per userid")
- SYSCTL_INT (_kern, OID_AUTO, maxusers, CTLFLAG_RDTUN,&maxusers, 0,"Hint for kernel tuning")
- SYSCTL_INT (_kern, KERN_ARGMAX, argmax, CTLFLAG_RD, 0, ARG_MAX,"Maximum bytes of argument to execve(2)")
- SYSCTL_INT (_kern, KERN_POSIX1, posix1version, CTLFLAG_RD, 0, _POSIX_VERSION,"Version of POSIX attempting to comply to")
- SYSCTL_INT (_kern, KERN_NGROUPS, ngroups, CTLFLAG_RD, 0, NGROUPS_MAX,"Maximum number of groups a user can belong to")
- SYSCTL_INT (_kern, KERN_JOB_CONTROL, job_control, CTLFLAG_RD, 0, 1,"Whether job control is available")
- SYSCTL_INT (_kern, KERN_SAVED_IDS, saved_ids, CTLFLAG_RD, 0, 0,"Whether saved set-group/user ID is available")
- SYSCTL_STRING (_kern, KERN_BOOTFILE, bootfile, CTLFLAG_RW, kernelname, sizeof kernelname,"Name of kernel file booted")
- SYSCTL_INT (_hw, HW_NCPU, ncpu, CTLFLAG_RD,&mp_ncpus, 0,"Number of active CPUs")
- SYSCTL_INT (_hw, HW_BYTEORDER, byteorder, CTLFLAG_RD, 0, BYTE_ORDER,"System byte order")
- SYSCTL_INT (_hw, HW_PAGESIZE, pagesize, CTLFLAG_RD, 0, PAGE_SIZE,"System memory page size")
- static int sysctl_hw_physmem (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_hw, HW_PHYSMEM, physmem, CTLTYPE_ULONG|CTLFLAG_RD, 0, 0, sysctl_hw_physmem,"LU","")
- static int sysctl_hw_realmem (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_hw, HW_REALMEM, realmem, CTLTYPE_ULONG|CTLFLAG_RD, 0, 0, sysctl_hw_realmem,"LU","")
- static int sysctl_hw_usermem (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_hw, HW_USERMEM, usermem, CTLTYPE_ULONG|CTLFLAG_RD, 0, 0, sysctl_hw_usermem,"LU","")
- SYSCTL_ULONG (_hw, OID_AUTO, availpages, CTLFLAG_RD,&physmem, 0,"")
- SYSCTL_STRING (_hw, HW_MACHINE_ARCH, machine_arch, CTLFLAG_RD, machine_arch, 0,"System architecture")
- static int sysctl_hostname (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern, KERN_HOSTNAME, hostname, CTLTYPE_STRING|CTLFLAG_RW|CTLFLAG_PRISON, 0, 0, sysctl_hostname,"A","Hostname")
- MTX_SYSINIT (securelevel_lock,&securelevel_mtx,"securelevel mutex lock", MTX_DEF)
- static int sysctl_kern_securelvl (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern, KERN_SECURELVL, securelevel, CTLTYPE_INT|CTLFLAG_RW|CTLFLAG_PRISON, 0, 0, sysctl_kern_securelvl,"I","Current secure level")
- SYSCTL_STRING (_kern, KERN_NISDOMAINNAME, domainname, CTLFLAG_RW,&domainname, sizeof(domainname),"Name of the current YP/NIS domain")

- SYSCTL_ULONG (_kern, KERN_HOSTID, hostid, CTLFLAG_RW,&hostid, 0,"Host ID")
- SYSCTL_STRING (_user, USER_CS_PATH, cs_path, CTLFLAG_RD,"", 0,"PATH that finds all the standard utilities")
- SYSCTL_INT (_user, USER_BC_BASE_MAX, bc_base_max, CTLFLAG_RD, 0, 0,"Max ibase/obase values in bc(1)")
- SYSCTL_INT (_user, USER_BC_DIM_MAX, bc_dim_max, CTLFLAG_RD, 0, 0,"Max array size in bc(1)")
- SYSCTL_INT (_user, USER_BC_SCALE_MAX, bc_scale_max, CTLFLAG_RD, 0, 0,"Max scale value in bc(1)")
- SYSCTL_INT (_user, USER_BC_STRING_MAX, bc_string_max, CTLFLAG_RD, 0, 0,"Max string length in bc(1)")
- SYSCTL_INT (_user, USER_COLL_WEIGHTS_MAX, coll_weights_max, CTLFLAG_RD, 0, 0,"Maximum number of weights assigned to an LC_COLLATE locale entry")
- SYSCTL_INT (_user, USER_EXPR_NEST_MAX, expr_nest_max, CTLFLAG_RD, 0, 0,"")
- SYSCTL_INT (_user, USER_LINE_MAX, line_max, CTLFLAG_RD, 0, 0,"Max length (bytes) of a text-processing utility's input line")
- SYSCTL_INT (_user, USER_RE_DUP_MAX, re_dup_max, CTLFLAG_RD, 0, 0,"Maximum number of repeats of a regexp permitted")
- SYSCTL_INT (_user, USER_POSIX2_VERSION, posix2_version, CTLFLAG_RD, 0, 0,"The version of POSIX 1003.2 with which the system attempts to comply")
- SYSCTL_INT (_user, USER_POSIX2_C_BIND, posix2_c_bind, CTLFLAG_RD, 0, 0,"Whether C development supports the C bindings option")
- SYSCTL_INT (_user, USER_POSIX2_C_DEV, posix2_c_dev, CTLFLAG_RD, 0, 0,"Whether system supports the C development utilities option")
- SYSCTL_INT (_user, USER_POSIX2_CHAR_TERM, posix2_char_term, CTLFLAG_RD, 0, 0,"")
- SYSCTL_INT (_user, USER_POSIX2_FORT_DEV, posix2_fort_dev, CTLFLAG_RD, 0, 0,"Whether system supports FORTRAN development utilities")
- SYSCTL_INT (_user, USER_POSIX2_FORT_RUN, posix2_fort_run, CTLFLAG_RD, 0, 0,"Whether system supports FORTRAN runtime utilities")
- SYSCTL_INT (_user, USER_POSIX2_LOCALEDEF, posix2_localedef, CTLFLAG_RD, 0, 0,"Whether system supports creation of locales")
- SYSCTL_INT (_user, USER_POSIX2_SW_DEV, posix2_sw_dev, CTLFLAG_RD, 0, 0,"Whether system supports software development utilities")
- SYSCTL_INT (_user, USER_POSIX2_UPE, posix2_upe, CTLFLAG_RD, 0, 0,"Whether system supports the user portability utilities")
- SYSCTL_INT (_user, USER_STREAM_MAX, stream_max, CTLFLAG_RD, 0, 0,"Min Maximum number of streams a process may have open at one time")
- SYSCTL_INT (_user, USER_TZNAME_MAX, tzname_max, CTLFLAG_RD, 0, 0,"Min Maximum number of types supported for timezone names")
- SYSCTL_INT (_debug_sizeof, OID_AUTO, vnode, CTLFLAG_RD, 0, sizeof(struct vnode),"sizeof(struct vnode)")
- SYSCTL_INT (_debug_sizeof, OID_AUTO, proc, CTLFLAG_RD, 0, sizeof(struct proc),"sizeof(struct proc)")
- SYSCTL_INT (_debug_sizeof, OID_AUTO, bio, CTLFLAG_RD, 0, sizeof(struct bio),"sizeof(struct bio)")
- SYSCTL_INT (_debug_sizeof, OID_AUTO, buf, CTLFLAG_RD, 0, sizeof(struct buf),"sizeof(struct buf)")
- SYSCTL_INT (_debug_sizeof, OID_AUTO, kinfo_proc, CTLFLAG_RD, 0, sizeof(struct kinfo_proc),"sizeof(struct kinfo_proc)")
- SYSCTL_INT (_kern, OID_AUTO, fallback_elf_brand, CTLFLAG_RW,&__elfN(fallback_brand), sizeof(__elfN(fallback_brand)),"compatibility for kern.fallback_elf_brand")

Variables

- int osreldate
- char kernelname [MAXPATHLEN] = "/kernel"
- static char machine_arch [ ] = MACHINE_ARCH
- char hostname [MAXHOSTNAMELEN]
- static int regression_securelevel_nonmonotonic = 0
- int securelevel = -1
- static struct mtx securelevel_mtx
- char domainname [MAXHOSTNAMELEN]
- u_long hostid

9.41.1 Function Documentation

9.41.1.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_mib. *c*, v 1.76 2005/08/21 18:03:31 pjd Exp $")

9.41.1.2 MTX_SYSINIT (securelevel_lock, & *securelevel_mtx*, "securelevel mutex lock", MTX_DEF)

9.41.1.3 static int sysctl_hostname (SYSCTL_HANDLER_ARGS) [static]

Definition at line 194 of file kern_mib.c.

References hostname, jail_set_hostname_allowed, pr, and sysctl_handle_string().

Here is the call graph for this function:

9.41.1.4 static int sysctl_hw_physmem (SYSCTL_HANDLER_ARGS) [static]

Definition at line 153 of file kern_mib.c.

References sysctl_handle_long().

Here is the call graph for this function:

9.41.1.5 static int sysctl_hw_realmem (SYSCTL_HANDLER_ARGS) [static]

Definition at line 165 of file kern_mib.c.

References sysctl_handle_long().

Here is the call graph for this function:

9.41.1.6 static int sysctl_hw_usermem (SYSCTL_HANDLER_ARGS) [static]

Definition at line 174 of file kern_mib.c.

References sysctl_handle_long().

Here is the call graph for this function:

9.41.1.7 SYSCTL_INT (_kern, OID_AUTO, fallback_elf_brand, CTLFLAG_RW, & *__elfN*(fallback_brand), sizeof(__elfN(fallback_brand)), "compatibility for kern.fallback_elf_brand")

9.41.1.8 SYSCTL_INT (_debug_sizeof, OID_AUTO, kinfo_proc, CTLFLAG_RD, 0, sizeof(struct kinfo_proc), "sizeof(struct kinfo_proc)")

9.41.1.9 SYSCTL_INT (_debug_sizeof, OID_AUTO, buf, CTLFLAG_RD, 0, sizeof(struct buf), "sizeof(struct buf)")

9.41.1.10 SYSCTL_INT (_debug_sizeof, OID_AUTO, bio, CTLFLAG_RD, 0, sizeof(struct bio), "sizeof(struct bio)")

9.41.1.11 SYSCTL_INT (_debug_sizeof, OID_AUTO, proc, CTLFLAG_RD, 0, sizeof(struct proc), "sizeof(struct proc)")

9.41.1.12 SYSCTL_INT (_debug_sizeof, OID_AUTO, vnode, CTLFLAG_RD, 0, sizeof(struct vnode), "sizeof(struct vnode)")

9.41.1.13 SYSCTL_INT (_user, USER_TZNAME_MAX, tzname_max, CTLFLAG_RD, 0, 0, "Min Maximum number of types supported for timezone names")

9.41.1.14 SYSCTL_INT (_user, USER_STREAM_MAX, stream_max, CTLFLAG_RD, 0, 0, "Min Maximum number of streams a process may have open at one time")

9.41.1.15 SYSCTL_INT (_user, USER_POSIX2_UPE, posix2_upe, CTLFLAG_RD, 0, 0, "Whether system supports the user portability utilities")

9.41.1.16 SYSCTL_INT (_user, USER_POSIX2_SW_DEV, posix2_sw_dev, CTLFLAG_RD, 0, 0, "Whether system supports software development utilities")

9.41.1.17 SYSCTL_INT (_user, USER_POSIX2_LOCALEDEF, posix2_localedef, CTLFLAG_RD, 0, 0, "Whether system supports creation of locales")

9.41.1.18 SYSCTL_INT (_user, USER_POSIX2_FORT_RUN, posix2_fort_run, CTLFLAG_RD, 0, 0, "Whether system supports FORTRAN runtime utilities")

9.41.1.19 SYSCTL_INT (_user, USER_POSIX2_FORT_DEV, posix2_fort_dev, CTLFLAG_RD, 0, 0, "Whether system supports FORTRAN development utilities")

9.41.1.20 SYSCTL_INT (_user, USER_POSIX2_CHAR_TERM, posix2_char_term, CTLFLAG_RD, 0, 0, "")

9.41.1.21 SYSCTL_INT (_user, USER_POSIX2_C_DEV, posix2_c_dev, CTLFLAG_RD, 0, 0, "Whether system supports the C development utilities option")

9.41.1.22 SYSCTL_INT (_user, USER_POSIX2_C_BIND, posix2_c_bind, CTLFLAG_RD, 0, 0, "Whether C development supports the C bindings option")

9.41.1.23 SYSCTL_INT (_user, USER_POSIX2_VERSION, posix2_version, CTLFLAG_RD, 0, 0, "The version of POSIX 1003.2 with which the system attempts to comply")

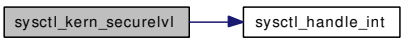
9.41.1.24 SYSCTL_INT (_user, USER_RE_DUP_MAX, re_dup_max, CTLFLAG_RD, 0, 0, "Maximum number of repeats of a regexp permitted")

9.41.1.25 SYSCTL_INT (_user, USER_LINE_MAX, line_max, CTLFLAG_RD, 0, 0, "Max length (bytes) of a text-processing utility's input line")

9.41.1.26 SYSCTL_INT (_user, USER_EXPR_NEST_MAX, expr_nest_max, CTLFLAG_RD, 0, 0, "")

References pr, regression_securelevel_nonmonotonic, securelevel, securelevel_mtx, and sysctl_handle_-int().

Here is the call graph for this function:



9.41.1.46 SYSCTL_NODE (OID_AUTO, security, CTLFLAG_RW, 0, "Security")

9.41.1.47 SYSCTL_NODE (OID_AUTO, compat, CTLFLAG_RW, 0, "Compatibility code")

9.41.1.48 SYSCTL_NODE (CTL_P1003_1B, p1003_1b, CTLFLAG_RW, 0, " *p1003_1b*, (see p1003_1b.h)")

9.41.1.49 SYSCTL_NODE (CTL_USER, user, CTLFLAG_RW, 0, "user-level")

9.41.1.50 SYSCTL_NODE (CTL_MACHDEP, machdep, CTLFLAG_RW, 0, "machine dependent")

9.41.1.51 SYSCTL_NODE (CTL_HW, hw, CTLFLAG_RW, 0, "hardware")

9.41.1.52 SYSCTL_NODE (_debug, OID_AUTO, sizeof, CTLFLAG_RW, 0, "Sizeof various things")

9.41.1.53 SYSCTL_NODE (CTL_DEBUG, debug, CTLFLAG_RW, 0, "Debugging")

9.41.1.54 SYSCTL_NODE (CTL_NET, net, CTLFLAG_RW, 0, " *Network*, (see socket.h)")

9.41.1.55 SYSCTL_NODE (CTL_VFS, vfs, CTLFLAG_RW, 0, "File system")

9.41.1.56 SYSCTL_NODE (CTL_VM, vm, CTLFLAG_RW, 0, "Virtual memory")

9.41.1.57 SYSCTL_NODE (CTL_KERN, kern, CTLFLAG_RW, 0, "High *kernel*, proc, limits &c")

9.41.1.58 SYSCTL_NODE (0, sysctl, CTLFLAG_RW, 0, "Sysctl internal magic")

9.41.1.59 SYSCTL_PROC (_kern, KERN_SECURELVL, securelevel, CTLTYPE_-INT|CTLFLAG_RW| *CTLFLAG_PRISON*, 0, 0, sysctl_kern_securelvl, "I", "Current secure level")

9.41.1.60 SYSCTL_PROC (_kern, KERN_HOSTNAME, hostname, CTLTYPE_-STRING|CTLFLAG_RW| *CTLFLAG_PRISON*, 0, 0, sysctl_hostname, "A", "Hostname")

9.41.1.61 SYSCTL_PROC (_hw, HW_USERMEM, usermem, CTLTYPE_ULONG| *CTLFLAG_RD*, 0, 0, sysctl_hw_usermem, "LU", "")

9.41.1.62 SYSCTL_PROC (_hw, HW_REALMEM, realmem, CTLTYPE_ULONG| *CTLFLAG_RD*, 0, 0, sysctl_hw_realmem, "LU", "")

9.41.1.63 SYSCTL_PROC (_hw, HW_PHYSMEM, physmem, CTLTYPE_ULONG| *CTLFLAG_RD*, 0, 0, sysctl_hw_physmem, "LU", "")

9.41.1.64 SYSCTL_STRING (_user, USER_CS_PATH, cs_path, CTLFLAG_RD, "", 0, "PATH that finds all the standard utilities")

9.41.1.65 SYSCTL_STRING (_kern, KERN_NISDOMAINNAME, domainname, CTLFLAG_RW, & *domainname*, sizeof(domainname), "Name of the current YP/NIS domain")

9.41.1.66 SYSCTL_STRING (_hw, HW_MACHINE_ARCH, machine_arch, CTLFLAG_RD, machine_arch, 0, "System architecture")

9.41.1.67 SYSCTL_STRING (_kern, KERN_BOOTFILE, bootfile, CTLFLAG_RW, kernelname, sizeof *kernelname*, "Name of kernel file booted")

9.41.1.68 SYSCTL_STRING (_kern, KERN_OSTYPE, ostype, CTLFLAG_RD, ostype, 0, "Operating system type")

Referenced by getdomainname(), and setdomainname().

9.41.2.2 u_long hostid

Definition at line 302 of file kern_mib.c.

9.41.2.3 char hostname[MAXHOSTNAMELEN]

Definition at line 191 of file kern_mib.c.

Referenced by getcredhostname(), and sysctl_hostname().

9.41.2.4 char kernelname[MAXPATHLEN] = "/kernel"

Definition at line 138 of file kern_mib.c.

9.41.2.5 char machine_arch[ ] = MACHINE_ARCH [static]

Definition at line 187 of file kern_mib.c.

9.41.2.6 int osreldate

9.41.2.7 int regression_securelevel_nonmonotonic = 0 [static]

Definition at line 235 of file kern_mib.c.

Referenced by sysctl_kern_securelvl().

9.41.2.8 int securelevel = -1

Definition at line 242 of file kern_mib.c.

Referenced by jail(), linker_file_unload(), linker_load_file(), securelevel_ge(), securelevel_gt(), and sysctl_kern_securelvl().

9.41.2.9 struct mtx securelevel_mtx [static]

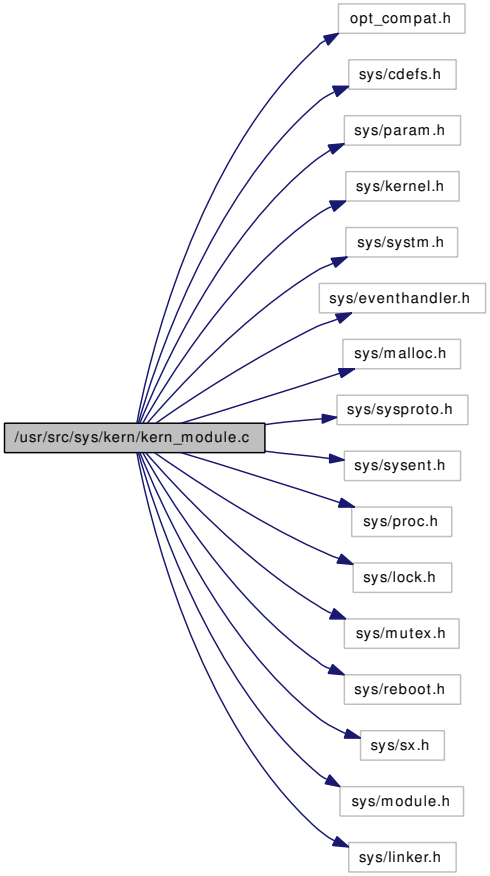
Definition at line 243 of file kern_mib.c.

Referenced by sysctl_kern_securelvl().

9.42 /usr/src/sys/kern/kern_module.c File Reference

```
#include "opt_compat.h"
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/kernel.h>
#include <sys/systm.h>
#include <sys/eventhandler.h>
#include <sys/malloc.h>
#include <sys/sysproto.h>
#include <sys/sysent.h>
#include <sys/proc.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/reboot.h>
#include <sys/sx.h>
#include <sys/module.h>
#include <sys/linker.h>
```

Include dependency graph for kern_module.c:



Data Structures

- struct module_stat_v1

Defines

- #define MOD_EVENT(mod, type) (mod) → handler((mod), (type), (mod) → arg)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_module.c,v 1.51 2006/06/26 18:34:45 jhb Exp $")
- static MALLOC_DEFINE (M_MODULE,"module","module data structures")
- typedef TAILQ_HEAD (module)
- static void module_shutdown (void *, int)
- static int modevent_nop (module_t mod, int what, void *arg)
- static void module_init (void *arg)
- void module_register_init (const void *arg)
- int module_register (const moduledata_t *data, linker_file_t container)
- void module_reference (module_t mod)
- void module_release (module_t mod)
- module_t module_lookupbyname (const char *name)

- module_t module_lookupbyid (int modid)
- int module_unload (module_t mod, int flags)
- int module_getid (module_t mod)
- module_t module_getfnext (module_t mod)
- void module_setspecific (module_t mod, modspecific_t *datap)
- linker_file_t module_file (module_t mod)
- int modnext (struct thread *td, struct modnext_args *uap)
- int modfnext (struct thread *td, struct modfnext_args *uap)
- int modstat (struct thread *td, struct modstat_args *uap)
- int modfind (struct thread *td, struct modfind_args *uap)

Variables

- static modulelist_t modules
- sx modules_sx
- static int nextid = 1

9.42.1 Define Documentation

9.42.1.1 #define MOD_EVENT(mod, type) (mod) → handler((mod), (type), (mod) → arg)

Definition at line 62 of file kern_module.c.

Referenced by module_register_init(), module_shutdown(), and module_unload().

9.42.2 Function Documentation

9.42.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_module. *c*, v 1.51 2006/06/26 18:34:45 jhb Exp $")

9.42.2.2 static MALLOC_DEFINE (M_MODULE, "module", "module data structures") [static]

9.42.2.3 static int modevent_nop (module_t *mod*, int *what*, void * *arg*) [static]

Definition at line 70 of file kern_module.c.

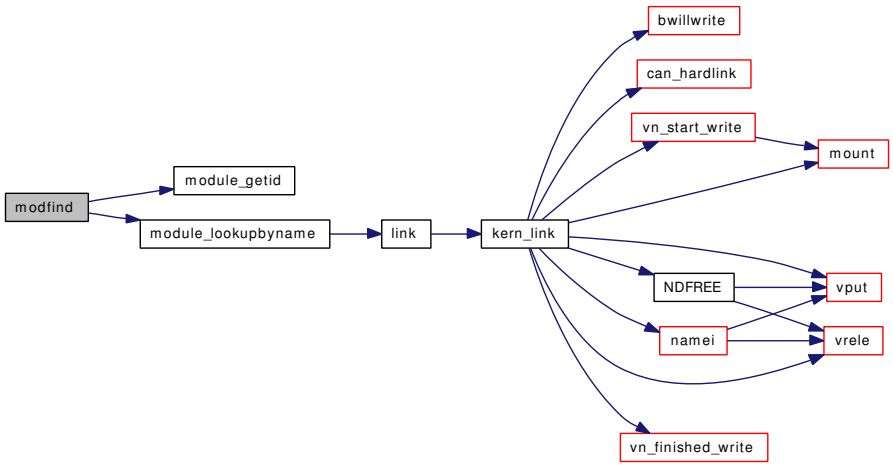
Referenced by module_register().

9.42.2.4 int modfind (struct thread * *td*, struct modfind_args * *uap*)

Definition at line 410 of file kern_module.c.

References module_getid(), and module_lookupbyname().

Here is the call graph for this function:

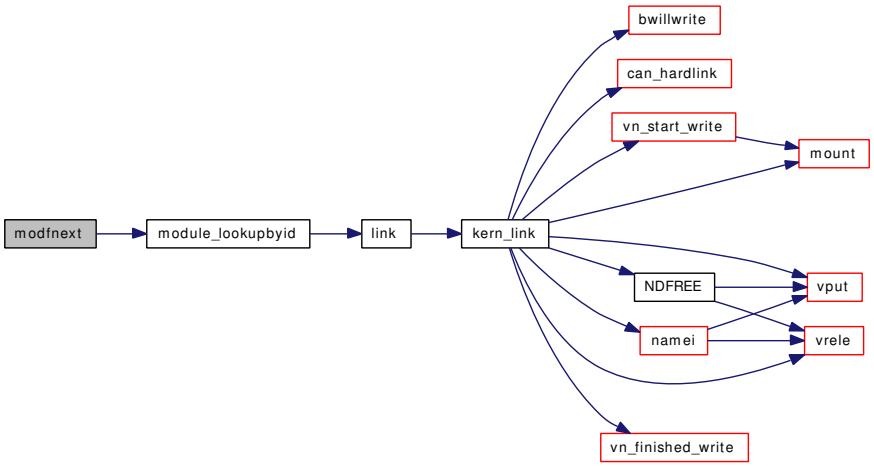


9.42.2.5 int modfnext (struct thread * *td*, struct modfnext_args * *uap*)

Definition at line 321 of file kern_module.c.

References module_lookupbyid().

Here is the call graph for this function:

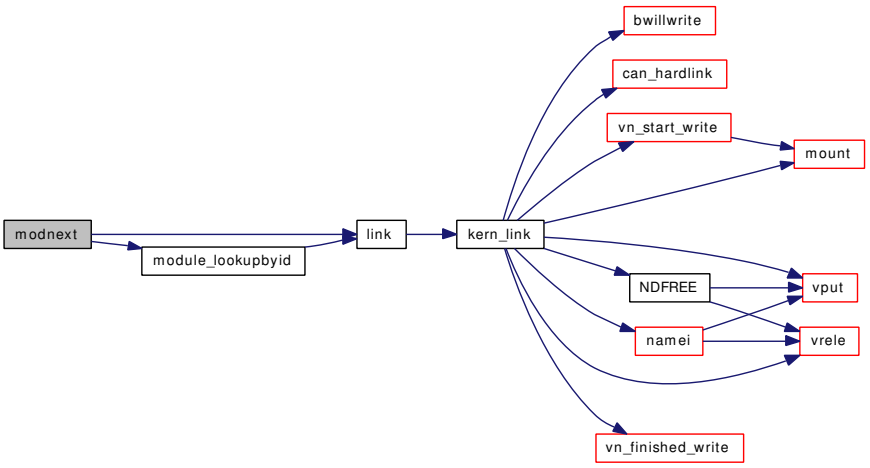


9.42.2.6 int modnext (struct thread * *td*, struct modnext_args * *uap*)

Definition at line 287 of file kern_module.c.

References link(), module_lookupbyid(), and modules.

Here is the call graph for this function:

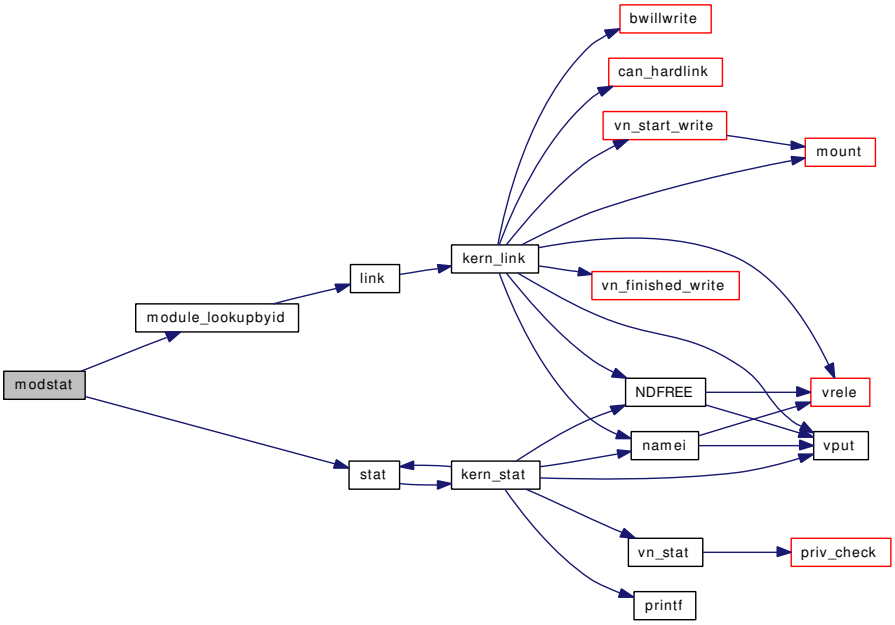


9.42.2.7 int modstat (struct thread * *td*, struct modstat_args * *uap*)

Definition at line 354 of file kern_module.c.

References module_lookupbyid(), and stat().

Here is the call graph for this function:



9.42.2.8 linker_file_t module_file (module_t *mod*)

Definition at line 274 of file kern_module.c.

9.42.2.9 module_t module_getfnext (module_t *mod*)

Definition at line 258 of file kern_module.c.

Referenced by linker_file_unload().

9.42.2.10 int module_getid (module_t *mod*)

Definition at line 250 of file kern_module.c.

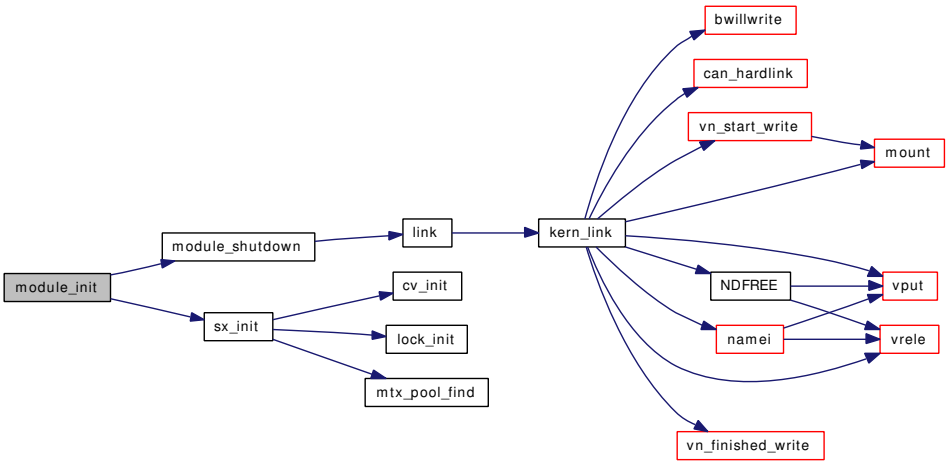
Referenced by kldfirstmod(), and modfind().

9.42.2.11 static void module_init (void * *arg*) [static]

Definition at line 84 of file kern_module.c.

References module_shutdown(), modules, modules_sx, and sx_init().

Here is the call graph for this function:



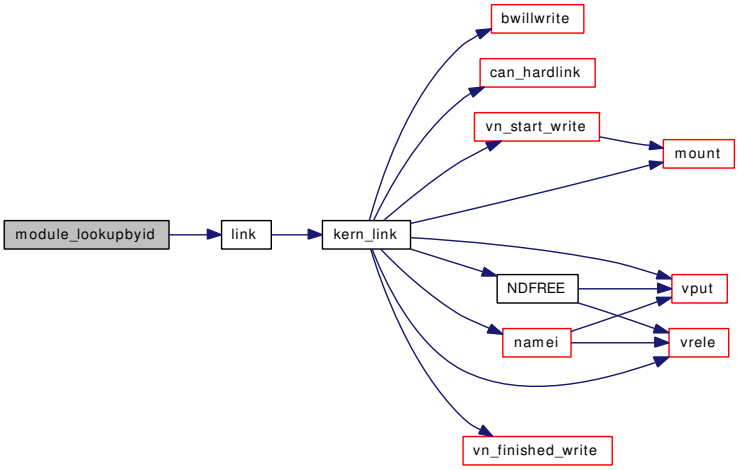
9.42.2.12 module_t module_lookupbyid (int *modid*)

Definition at line 222 of file kern_module.c.

References link(), and modules.

Referenced by modfnext(), modnext(), and modstat().

Here is the call graph for this function:



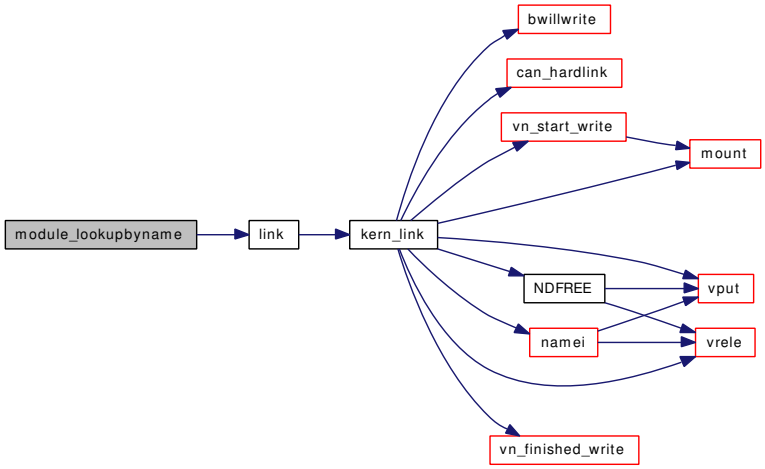
9.42.2.13 module_t module_lookupbyname (const char * *name*)

Definition at line 206 of file kern_module.c.

References link(), and modules.

Referenced by modfind(), module_register(), and module_register_init().

Here is the call graph for this function:



9.42.2.14 void module_reference (module_t *mod*)

Definition at line 174 of file kern_module.c.

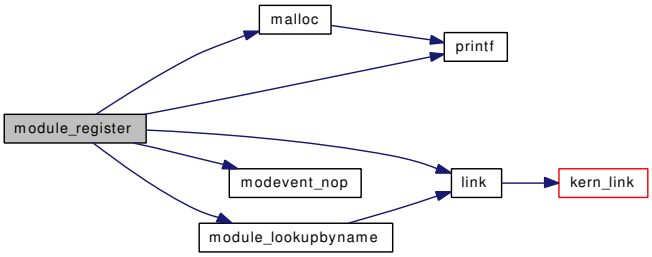
9.42.2.15 int module_register (const moduledata_t * *data*, linker_file_t *container*)

Definition at line 138 of file kern_module.c.

References link(), malloc(), modevent_nop(), module_lookupbyname(), modules, nextid, and printf().

Referenced by linker_file_register_modules().

Here is the call graph for this function:

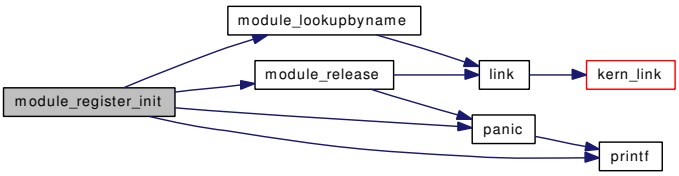


9.42.2.16 void module_register_init (const void * *arg*)

Definition at line 111 of file kern_module.c.

References Giant, MOD_EVENT, module_lookupbyname(), module_release(), panic(), and printf().

Here is the call graph for this function:



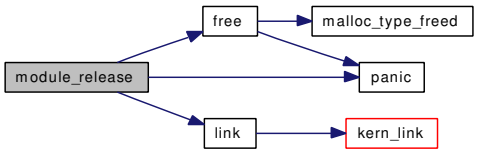
9.42.2.17 void module_release (module_t *mod*)

Definition at line 184 of file kern_module.c.

References free(), link(), modules, and panic().

Referenced by linker_file_unload(), and module_register_init().

Here is the call graph for this function:



9.42.2.18 void module_setspecific (module_t *mod*, modspecific_t * *datap*)

Definition at line 266 of file kern_module.c.

Referenced by syscall_module_handler().

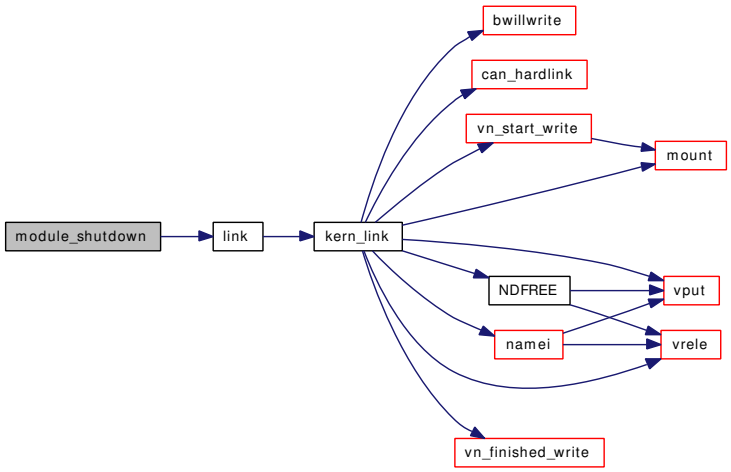
9.42.2.19 static void module_shutdown (void *, int) [static]

Definition at line 96 of file kern_module.c.

References Giant, link(), MOD_EVENT, and modules.

Referenced by module_init().

Here is the call graph for this function:



9.42.2.20 int module_unload (module_t *mod*, int *flags*)

Definition at line 235 of file kern_module.c.

References Giant, and MOD_EVENT.

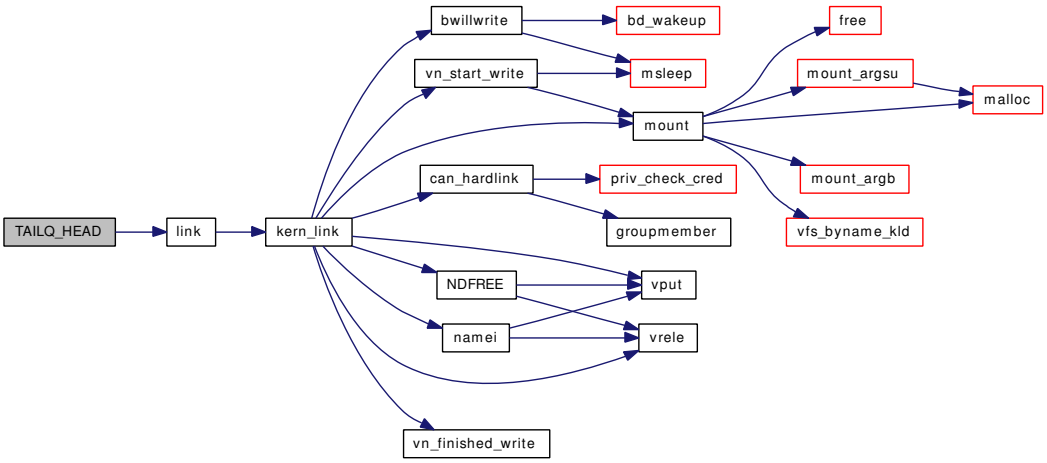
Referenced by linker_file_unload().

9.42.2.21 typedef TAILQ_HEAD (module)

Definition at line 49 of file kern_module.c.

References link().

Here is the call graph for this function:



9.42.3 Variable Documentation

9.42.3.1 modulelist_t modules [static]

Definition at line 64 of file kern_module.c.

Referenced by modnext(), module_init(), module_lookupbyid(), module_lookupbyname(), module_register(), module_release(), and module_shutdown().

9.42.3.2 struct sx modules_sx

Definition at line 65 of file kern_module.c.

Referenced by module_init().

9.42.3.3 int nextid = 1 [static]

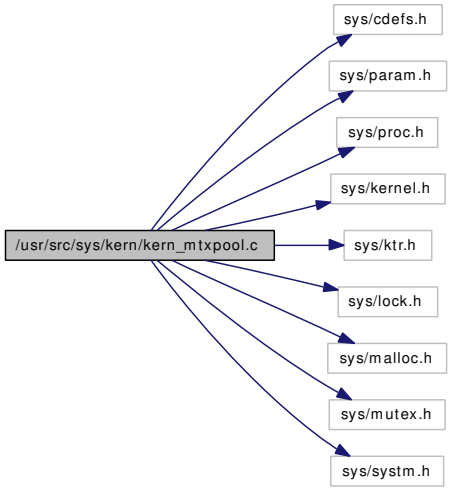
Definition at line 66 of file kern_module.c.

Referenced by module_register().

9.43 /usr/src/sys/kern/kern_mtxpool.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/proc.h>
#include <sys/kernel.h>
#include <sys/ktr.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/systm.h>
```

Include dependency graph for kern_mtxpool.c:



Data Structures

- struct mtxpool_header
- struct mtx_pool
- struct mtx_pool_lockbuilder

Defines

- #define MTX_POOL_LOCKBUILDER_SIZE 128
- #define MTX_POOL_SLEEP_SIZE 128
- #define mtx_pool_size mtx_pool_header.mtxpool_size
- #define mtx_pool_mask mtx_pool_header.mtxpool_mask
- #define mtx_pool_shift mtx_pool_header.mtxpool_shift
- #define mtx_pool_next mtx_pool_header.mtxpool_next
- #define POINTER_BITS 64
- #define HASH_MULTIPLIER 11400714819323198485u

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_mtxpool.c,v 1.11 2005/02/10 12:02:37 phk Exp $")
- static MALLOC_DEFINE (M_MTXPOOL,"mtx_pool","mutex pool")
- mtx * mtx_pool_find (struct mtx_pool *pool, void *ptr)
- static void mtx_pool_initialize (struct mtx_pool *pool, const char *mtx_name, int pool_size, int opts)
- mtx_pool * mtx_pool_create (const char *mtx_name, int pool_size, int opts)
- void mtx_pool_destroy (struct mtx_pool **poolp)
- static void mtx_pool_setup_static (void *dummy __unused)
- static void mtx_pool_setup_dynamic (void *dummy __unused)
- mtx * mtx_pool_alloc (struct mtx_pool *pool)
- SYSINIT (mtxpooli1, SI_SUB_MTX_POOL_STATIC, SI_ORDER_FIRST, mtx_pool_setup_static, NULL)
- SYSINIT (mtxpooli2, SI_SUB_MTX_POOL_DYNAMIC, SI_ORDER_FIRST, mtx_pool_setup_dynamic, NULL)

Variables

- static struct mtx_pool_lockbuilder lockbuilder_pool
- mtx_pool * mtxpool_sleep
- mtx_pool * mtxpool_lockbuilder

9.43.1 Define Documentation

9.43.1.1 #define HASH_MULTIPLIER 11400714819323198485u

Definition at line 96 of file kern_mtxpool.c.

Referenced by mtx_pool_find().

9.43.1.2 #define MTX_POOL_LOCKBUILDER_SIZE 128

Definition at line 63 of file kern_mtxpool.c.

Referenced by mtx_pool_setup_static().

9.43.1.3 #define mtx_pool_mask mtx_pool_header.mtxpool_mask

Definition at line 87 of file kern_mtxpool.c.

9.43.1.4 #define mtx_pool_next mtx_pool_header.mtxpool_next

Definition at line 89 of file kern_mtxpool.c.

9.43.1.5 #define mtx_pool_shift mtx_pool_header.mtxpool_shift

Definition at line 88 of file kern_mtxpool.c.

9.43.1.6 #define mtx_pool_size mtx_pool_header.mtxpool_size

Definition at line 86 of file kern_mtxpool.c.

9.43.1.7 #define MTX_POOL_SLEEP_SIZE 128

Definition at line 66 of file kern_mtxpool.c.

Referenced by mtx_pool_setup_dynamic().

9.43.1.8 #define POINTER_BITS 64

Definition at line 95 of file kern_mtxpool.c.

Referenced by mtx_pool_initialize().

9.43.2 Function Documentation

9.43.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_mtxpool. *c*, v 1.11 2005/02/10 12:02:37 phk Exp $")

9.43.2.2 static MALLOC_DEFINE (M_MTXPOOL, "mtx_pool", "mutex pool") [static]

9.43.2.3 struct mtx* mtx_pool_alloc (struct mtx_pool * *pool*)

Definition at line 190 of file kern_mtxpool.c.

References mtx_pool::mtx_pool_ary.

Referenced by falloc(), lockinit(), and uifind().

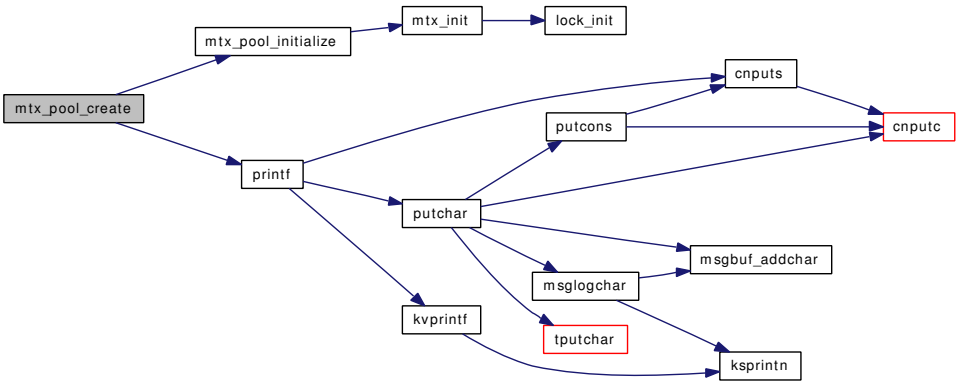
9.43.2.4 struct mtx_pool* mtx_pool_create (const char * *mtx_name*, int *pool_size*, int *opts*)

Definition at line 140 of file kern_mtxpool.c.

References mtx_pool_initialize(), and printf().

Referenced by mtx_pool_setup_dynamic().

Here is the call graph for this function:

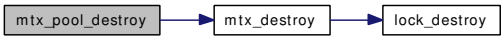


9.43.2.5 void mtx_pool_destroy (struct mtx_pool ** *poolp*)

Definition at line 157 of file kern_mtxpool.c.

References mtx_destroy(), and mtx_pool::mtx_pool_ary.

Here is the call graph for this function:



9.43.2.6 struct mtx* mtx_pool_find (struct mtx_pool * *pool*, void * *ptr*)

Definition at line 109 of file kern_mtxpool.c.

References HASH_MULTIPLIER, and mtx_pool::mtx_pool_ary.

Referenced by sx_init().

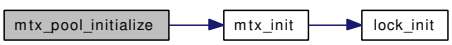
9.43.2.7 static void mtx_pool_initialize (struct mtx_pool * *pool*, const char * *mtx_name*, int *pool_size*, int *opts*) [static]

Definition at line 124 of file kern_mtxpool.c.

References mtx_init(), mtx_pool::mtx_pool_ary, and POINTER_BITS.

Referenced by mtx_pool_create(), and mtx_pool_setup_static().

Here is the call graph for this function:

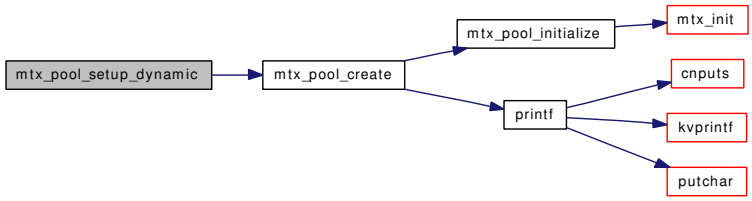


9.43.2.8 static void mtx_pool_setup_dynamic (void *dummy *__unused*) [static]

Definition at line 178 of file kern_mtxpool.c.

References mtx_pool_create(), MTX_POOL_SLEEP_SIZE, and mtxpool_sleep.

Here is the call graph for this function:

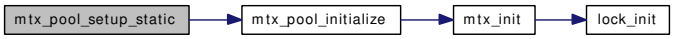


9.43.2.9 static void mtx_pool_setup_static (void *dummy *__unused*) [static]

Definition at line 169 of file kern_mtxpool.c.

References lockbuilder_pool, mtx_pool_initialize(), MTX_POOL_LOCKBUILDER_SIZE, and mtxpool_lockbuilder.

Here is the call graph for this function:



9.43.2.10 SYSINIT (mtxpooli2, SI_SUB_MTX_POOL_DYNAMIC, SI_ORDER_FIRST, mtx_pool_setup_dynamic, NULL)

9.43.2.11 SYSINIT (mtxpooli1, SI_SUB_MTX_POOL_STATIC, SI_ORDER_FIRST, mtx_pool_setup_static, NULL)

9.43.3 Variable Documentation

9.43.3.1 struct mtx_pool_lockbuilder lockbuilder_pool [static]

Referenced by mtx_pool_setup_static().

9.43.3.2 struct mtx_pool* mtxpool_lockbuilder

Definition at line 92 of file kern_mtxpool.c.

Referenced by lockinit(), mtx_pool_setup_static(), and sx_init().

9.43.3.3 struct mtx_pool* mtxpool_sleep

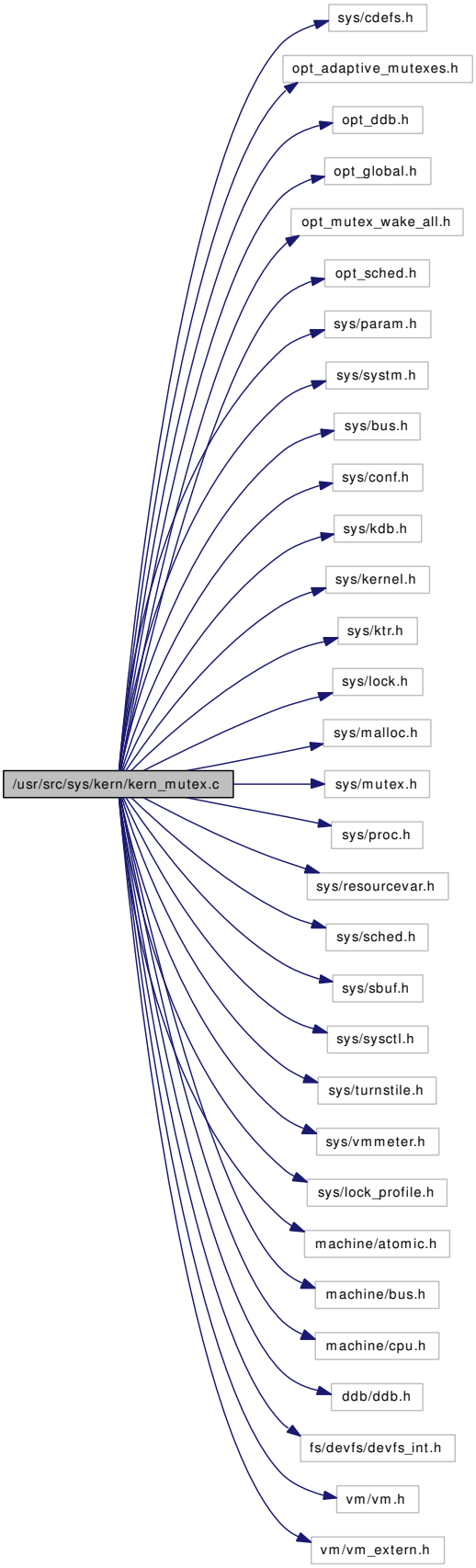
Definition at line 91 of file kern_mtxpool.c.

Referenced by falloc(), intr_event_create(), intr_event_destroy(), mtx_pool_setup_dynamic(), and uifind().

9.44 /usr/src/sys/kern/kern_mutex.c File Reference

```
#include <sys/cdefs.h>
#include "opt_adaptive_mutexes.h"
#include "opt_ddb.h"
#include "opt_global.h"
#include "opt_mutex_wake_all.h"
#include "opt_sched.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/bus.h>
#include <sys/conf.h>
#include <sys/kdb.h>
#include <sys/kernel.h>
#include <sys/ktr.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/resourcevar.h>
#include <sys/sched.h>
#include <sys/sbuf.h>
#include <sys/sysctl.h>
#include <sys/turnstile.h>
#include <sys/vmmeter.h>
#include <sys/lock_profile.h>
#include <machine/atomic.h>
#include <machine/bus.h>
#include <machine/cpu.h>
#include <ddb/ddb.h>
#include <fs/devfs/devfs_int.h>
#include <vm/vm.h>
#include <vm/vm_extern.h>
```

Include dependency graph for kern_mutex.c:


/usr/src/sys/kern/kern_mutex.c
sys/cdefs.h
opt_adaptive_mutexes.h
opt_ddb.h
opt_global.h
opt_mutex_wake_all.h
opt_sched.h
sys/param.h
sys/systm.h
sys/bus.h
sys/conf.h
sys/kdb.h
sys/kernel.h
sys/ktr.h
sys/lock.h
sys/malloc.h
sys/mutex.h
sys/proc.h
sys/resourcevar.h
sys/sched.h
sys/sbuf.h
sys/sysctl.h
sys/turnstile.h
sys/vmmeter.h
sys/lock_profile.h
machine/atomic.h
machine/bus.h
machine/cpu.h
ddb/ddb.h
fs/devfs/devfs_int.h
vm/vm.h
vm/vm_extern.h

Defines

- #define mtx_unowned(m) ((m) → mtx_lock == MTX_UNOWNED)
- #define mtx_owner(m) ((struct thread ∗)((m) → mtx_lock & ∼MTX_FLAGMASK))

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_mutex.c,v 1.179 2006/12/16 02:37:57 kmacy Exp $")
- static void lock_profile_init (void)
- void _mtx_lock_flags (struct mtx ∗m, int opts, const char ∗file, int line)
- void _mtx_unlock_flags (struct mtx ∗m, int opts, const char ∗file, int line)
- void _mtx_lock_spin_flags (struct mtx ∗m, int opts, const char ∗file, int line)
- void _mtx_unlock_spin_flags (struct mtx ∗m, int opts, const char ∗file, int line)
- int _mtx_trylock (struct mtx ∗m, int opts, const char ∗file, int line)
- void _mtx_lock_sleep (struct mtx ∗m, uintptr_t tid, int opts, const char ∗file, int line)
- void _mtx_unlock_sleep (struct mtx ∗m, int opts, const char ∗file, int line)
- void mtx_sysinit (void ∗arg)
- void mtx_init (struct mtx ∗m, const char ∗name, const char ∗type, int opts)
- void mtx_destroy (struct mtx ∗m)
- void mutex_init (void)

Variables

- lock_class lock_class_mtx_sleep
- lock_class lock_class_mtx_spin
- mtx sched_lock
- mtx Giant

9.44.1 Define Documentation

9.44.1.1 #define mtx_owner(m) ((struct thread ∗)((m) → mtx_lock & ∼MTX_FLAGMASK))

Definition at line 89 of file kern_mutex.c.

Referenced by _mtx_lock_sleep().

9.44.1.2 #define mtx_unowned(m) ((m) → mtx_lock == MTX_UNOWNED)

Definition at line 87 of file kern_mutex.c.

Referenced by mtx_destroy().

9.44.2 Function Documentation

9.44.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_mutex. *c*, v 1.179 2006/12/16 02:37:57 kmacy Exp $")

9.44.2.2 void _mtx_lock_flags (struct mtx ∗ *m*, int *opts*, const char ∗ *file*, int *line*)

Definition at line 138 of file kern_mutex.c.

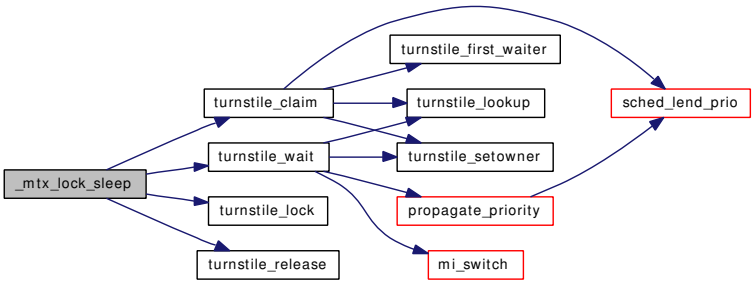
References lock_class_mtx_sleep, and waittime.

9.44.2.3 void _mtx_lock_sleep (struct mtx * *m*, uintptr_t *tid*, int *opts*, const char * *file*, int *line*)

Definition at line 264 of file kern_mutex.c.

References Giant, mtx_owner, turnstile_claim(), turnstile_lock(), turnstile_release(), and turnstile_wait().

Here is the call graph for this function:



9.44.2.4 void _mtx_lock_spin_flags (struct mtx * *m*, int *opts*, const char * *file*, int *line*)

Definition at line 181 of file kern_mutex.c.

References lock_class_mtx_spin, and waittime.

9.44.2.5 int _mtx_trylock (struct mtx * *m*, int *opts*, const char * *file*, int *line*)

Definition at line 226 of file kern_mutex.c.

References lock_class_mtx_sleep, and waittime.

9.44.2.6 void _mtx_unlock_flags (struct mtx * *m*, int *opts*, const char * *file*, int *line*)

Definition at line 161 of file kern_mutex.c.

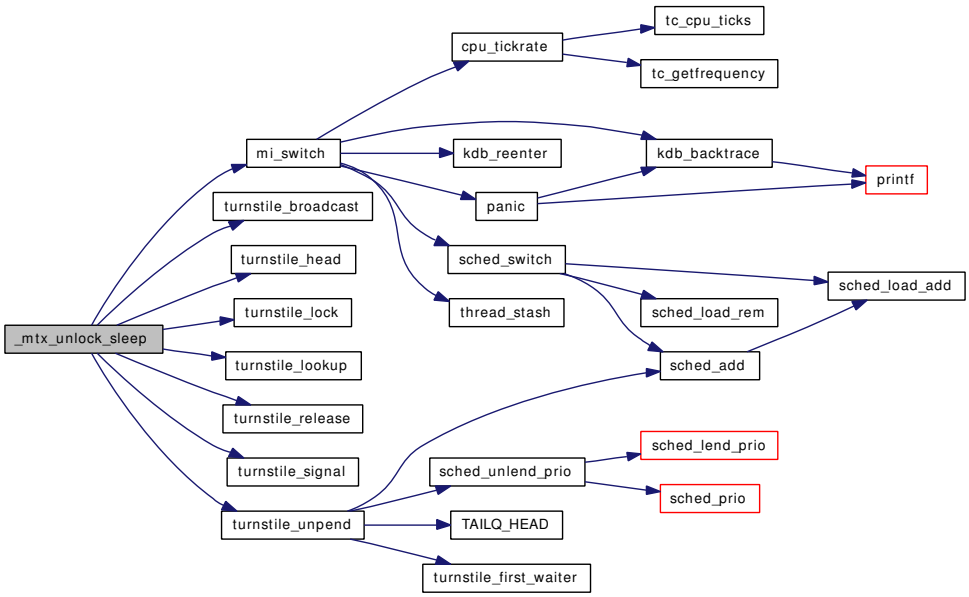
References lock_class_mtx_sleep.

9.44.2.7 void _mtx_unlock_sleep (struct mtx * *m*, int *opts*, const char * *file*, int *line*)

Definition at line 454 of file kern_mutex.c.

References mi_switch(), sched_lock, turnstile_broadcast(), turnstile_head(), turnstile_lock(), turnstile_lookup(), turnstile_release(), turnstile_signal(), and turnstile_unpend().

Here is the call graph for this function:



9.44.2.8 void _mtx_unlock_spin_flags (struct mtx * *m*, int *opts*, const char * *file*, int *line*)

Definition at line 203 of file kern_mutex.c.

References lock_class_mtx_spin.

9.44.2.9 static void lock_profile_init (void) [inline, static]

Definition at line 130 of file kern_mutex.c.

Referenced by mutex_init().

9.44.2.10 void mtx_destroy (struct mtx * *m*)

Definition at line 684 of file kern_mutex.c.

References lock_class_mtx_spin, lock_destroy(), and mtx_unowned.

Referenced by aio_init_aioinfo(), aio_unload(), alq_close(), fddrop(), intr_event_destroy(), itimer_fini(), jail(), mbp_destroy(), mount_fini(), mqueue_free(), msgunload(), mtx_pool_destroy(), pgdelete(), pipe_zone_fini(), prison_complete(), proc_fini(), rman_fini(), sem_modload(), sema_destroy(), semunload(), sessrele(), sigacts_free(), taskqueue_free(), ttyrel(), and vdestroy().

Here is the call graph for this function:

9.44.2.11 void mtx_init (struct mtx * *m*, const char * *name*, const char * *type*, int *opts*)

Definition at line 639 of file kern_mutex.c.

References lock_class_mtx_sleep, lock_class_mtx_spin, and lock_init().

Referenced by _taskqueue_create(), aio_init_aioinfo(), aio_onceonly(), ald_startup(), alq_open(), bufinit(), cn_drvinit(), devinit(), devstat_new_entry(), enterpgrp(), eventhandler_init(), fdinit(), filelistinit(), getnewvnode(), init_device_poll(), init_dynamic_kenv(), init_prison(), init_sleepqueues(), init_taskqueue_list(), init_turnstiles(), intr_event_create(), itimer_init(), jail(), kern_timeout_callwheel_init(), kmeminit(), kobj_init_mutex(), kqueue(), LIST_HEAD(), mbp_create(), mount_init(), mqueue_alloc(), msginit(), mtx_pool_initialize(), mtx_sysinit(), mutex_init(), pipe_zone_init(), proc0_init(), proc_init(), procinit(), pty_drvinit(), rman_init(), selectinit(), sem_modload(), sema_init(), seminit(), sigacts_alloc(), STAILQ_HEAD(), SYSINIT(), threadinit(), ttyalloc(), uihashinit(), umtxq_sysinit(), v_addpollinfo(), and vntblinit().

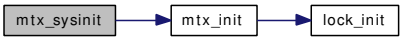
Here is the call graph for this function:

9.44.2.12 void mtx_sysinit (void * *arg*)

Definition at line 625 of file kern_mutex.c.

References mtx_init().

Here is the call graph for this function:

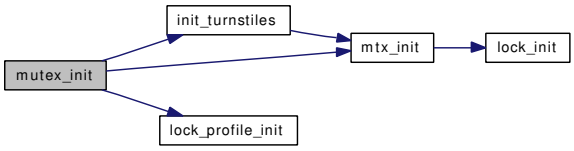


9.44.2.13 void mutex_init (void)

Definition at line 714 of file kern_mutex.c.

References devmtx, Giant, init_turnstiles(), lock_profile_init(), mtx_init(), proc0, and sched_lock.

Here is the call graph for this function:



9.44.3 Variable Documentation

9.44.3.1 struct mtx Giant

Definition at line 117 of file kern_mutex.c.

Referenced by __sysctl(), _mtx_lock_sleep(), ast(), buf_daemon(), callout_init(), doenterpgrp(), dounmount(), enterpgrp(), exit1(), flock(), fork_exit(), fork_return(), getdomainname(), giant_close(), giant_fdopen(), giant_ioctl(), giant_kqfilter(), giant_mmap(), giant_open(), giant_poll(), giant_read(), giant_strategy(), giant_write(), ithread_execute_handlers(), ithread_loop(), kern___getcwd(), kern_adjtime(),

kern_fcntl(), kern_shmat(), kern_shmctl(), kern_stat(), kern_statfs(), leavepgrp(), lf_advlock(), link_elf_load_file(), linker_file_register_sysctls(), linker_file_sysinit(), linker_file_sysuninit(), linker_file_unregister_sysctls(), mi_switch(), module_register_init(), module_shutdown(), module_unload(), mount(), msleep(), mutex_init(), namei(), ntp_adjtime(), ntp_gettime(), pf_proto_register(), pf_proto_unregister(), pgdelete(), reboot(), sched_idletd(), sched_sync(), semget(), setdomainname(), settime(), shmdt(), shmexit_myhook(), shmfork_myhook(), shmget(), shmsys(), softclock(), STAILQ_HEAD(), start_init(), thread_exit(), thread_single(), thread_suspend_check(), thread_wait(), tprintf(), ttycreate(), ttyfree(), uipc_bind(), uname(), unmount(), unp_connect(), uprintf(), vfs_domount(), vfs_donmount(), vnlru_proc(), witness_checkorder(), witness_warn(), and yield().

9.44.3.2 struct lock_class lock_class_mtx_sleep

Initial value:

```
{
        "sleep mutex",
        LC_SLEEPLOCK | LC_RECURSABLE,


}
```

Definition at line 98 of file kern_mutex.c.

Referenced by _mtx_lock_flags(), _mtx_trylock(), _mtx_unlock_flags(), and mtx_init().

9.44.3.3 struct lock_class lock_class_mtx_spin

Initial value:

```
{
        "spin mutex",
        LC_SPINLOCK | LC_RECURSABLE,


}
```

Definition at line 105 of file kern_mutex.c.

Referenced by _mtx_lock_spin_flags(), _mtx_unlock_spin_flags(), mtx_destroy(), and mtx_init().

9.44.3.4 struct mtx sched_lock

Definition at line 116 of file kern_mutex.c.

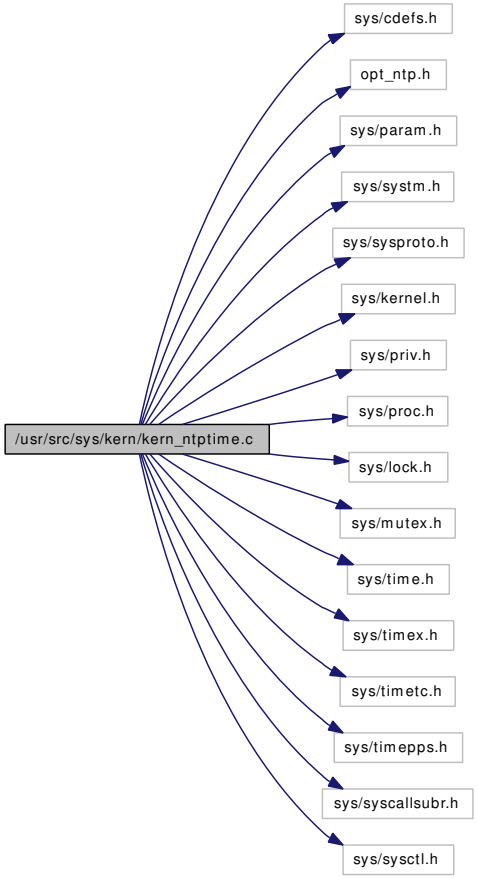
Referenced by _do_lock_pp(), _mtx_unlock_sleep(), acct_thread(), addupc_intr(), ast(), calcru(), cf_set_method(), create_init(), create_thread(), cursig(), do_unlock_pi(), do_unlock_pp(), donice(), doselwakeup(), exit1(), fill_kinfo_proc(), fill_kinfo_proc_only(), fork1(), fork_exit(), getscheduler(), hardclock_cpu(), idle_setup(), intr_event_remove_handler(), intr_event_schedule_thread(), issignal(), ithread_create(), ithread_destroy(), ithread_loop(), ithread_update(), kern_getitimer(), kern_ptrace(), kern_select(), kern_setitimer(), kern_setrlimit(), kern_thr_suspend(), kern_wait(), kick_init(), krunq_choose(), ksched_getparam(), ksched_setscheduler(), kse_create(), kse_exit(), kse_release(), kse_switchin(), kse_thr_interrupt(), kse_wakeup(), kseq_choose(), kthread_create(), lf_setlock(), maybe_resched(), mi_switch(), msleep(), mutex_init(), panic(), poll(), poll_idle(), pri_to_rtp(), proc_compare(),

profil(), propagate_priority(), ptracestop(), roundrobin(), rtp_to_pri(), rtprio(), rtprio_thread(), sched_add(), sched_bind(), sched_choose(), sched_class(), sched_clock(), sched_exit(), sched_fork(), sched_idletd(), sched_initticks(), sched_is_bound(), sched_nice(), sched_pctcpu(), sched_priority(), sched_relinquish(), sched_rem(), sched_setup(), sched_sleep(), sched_switch(), sched_thread_priority(), sched_tick(), sched_unbind(), sched_userret(), sched_wakeup(), schedcpu(), setrunnable(), sig_suspend_threads(), signotify(), sigqueue_delete_set_proc(), sigtd(), sleepq_abort(), sleepq_broadcast(), sleepq_catch_signals(), sleepq_check_signals(), sleepq_check_timeout(), sleepq_remove(), sleepq_resume_thread(), sleepq_signal(), sleepq_switch(), sleepq_timedwait(), sleepq_timedwait_sig(), sleepq_timeout(), sleepq_wait(), sleepq_wait_sig(), startprofclock(), statclock(), stopprofclock(), sysctl_kern_proc(), sysctl_out_proc(), taskqueue_start_threads(), tdq_choose(), tdq_load_add(), tdq_load_rem(), tdsigwakeup(), thr_exit(), thr_wake(), thread_exit(), thread_find(), thread_single(), thread_single_end(), thread_stopped(), thread_suspend_check(), thread_suspend_one(), thread_unlink(), thread_unsuspend(), thread_unsuspend_one(), trapsignal(), ttyinfo(), turnstile_adjust(), turnstile_adjust_thread(), turnstile_claim(), turnstile_disown(), turnstile_unpend(), turnstile_wait(), uio_yield(), umtx_pi_adjust(), umtx_pi_adjust_thread(), umtx_pi_claim(), umtx_pi_setowner(), umtx_pi_unref(), umtx_propagate_priority(), umtx_thread_cleanup(), umtx_unpropagate_priority(), umtxq_sleep_pi(), and userret().

9.45 /usr/src/sys/kern/kern_ntptime.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ntp.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/sysproto.h>
#include <sys/kernel.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/time.h>
#include <sys/timex.h>
#include <sys/timetc.h>
#include <sys/timepps.h>
#include <sys/syscallsubr.h>
#include <sys/sysctl.h>
```

Include dependency graph for kern_ntptime.c:



Data Structures

- struct ntp_gettime_args
- struct ntp_adjtime_args
- struct adjtime_args

Defines

- #define L_ADD(v, u) ((v) += (u))
- #define L_SUB(v, u) ((v) -= (u))
- #define L_ADDHI(v, a) ((v) += (int64_t)(a) << 32)
- #define L_NEG(v) ((v) = -(v))
- #define L_RSHIFT(v, n)
- #define L_MPY(v, a) ((v) *= (a))
- #define L_CLR(v) ((v) = 0)
- #define L_ISNEG(v) ((v) < 0)
- #define L_LINT(v, a) ((v) = (int64_t)(a) << 32)
- #define L_GINT(v) ((v) < 0 ? -(-(v) >> 32) : (v) >> 32)
- #define SHIFT_PLL 4
- #define SHIFT_FLL 2

Typedefs

- typedef int64_t l_fp

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_ntptime.c,v 1.61 2007/01/12 07:40:30 imp Exp $")
- static void ntp_init (void)
- static void hardupdate (long offset)
- static void ntp_gettime1 (struct ntptimeval *ntvp)
- int ntp_gettime (struct thread *td, struct ntp_gettime_args *uap)
- static int ntp_sysctl (SYSCTL_HANDLER_ARGS)
- SYSCTL_NODE (_kern, OID_AUTO, ntp_pll, CTLFLAG_RW, 0,"")
- SYSCTL_PROC (_kern_ntp_pll, OID_AUTO, gettime, CTLTYPE_OPAQUE|CTLFLAG_RD, 0, sizeof(struct ntptimeval), ntp_sysctl,"S,ntptimeval","")
- int ntp_adjtime (struct thread *td, struct ntp_adjtime_args *uap)
- void ntp_update_second (int64_t *adjustment, time_t *newsec)
- int adjtime (struct thread *td, struct adjtime_args *uap)
- int kern_adjtime (struct thread *td, struct timeval *delta, struct timeval *olddelta)

Variables

- static int time_state = TIME_OK
- static int time_status = STA_UNSYNC
- static long time_tai
- static long time_monitor
- static long time_constant
- static long time_precision = 1
- static long time_maxerror = MAXPHASE / 1000
- static long time_esterror = MAXPHASE / 1000
- static long time_reftime
- static l_fp time_offset
- static l_fp time_freq
- static l_fp time_adj
- static int64_t time_adjtime

9.45.1 Define Documentation

9.45.1.1 #define L_ADD(v, u) ((v) += (u))

Definition at line 57 of file kern_ntptime.c.

Referenced by hardupdate(), and ntp_update_second().

9.45.1.2 #define L_ADDHI(v, a) ((v) += (int64_t)(a) << 32)

Definition at line 59 of file kern_ntptime.c.

9.45.1.3 #define L_CLR(v) ((v) = 0)

Definition at line 69 of file kern_ntptime.c.

Referenced by ntp_init().

9.45.1.4 #define L_GINT(v) ((v) < 0 ? -(-(v) >> 32) : (v) >> 32)

Definition at line 72 of file kern_ntptime.c.

Referenced by hardupdate(), and ntp_adjtime().

9.45.1.5 #define L_ISNEG(v) ((v) < 0)

Definition at line 70 of file kern_ntptime.c.

9.45.1.6 #define L_LINT(v, a) ((v) = (int64_t)(a) << 32)

Definition at line 71 of file kern_ntptime.c.

Referenced by hardupdate(), ntp_adjtime(), and ntp_update_second().

9.45.1.7 #define L_MPY(v, a) ((v) *= (a))

Definition at line 68 of file kern_ntptime.c.

Referenced by hardupdate().

9.45.1.8 #define L_NEG(v) ((v) = -(v))

Definition at line 60 of file kern_ntptime.c.

9.45.1.9 #define L_RSHIFT(v, n)

Value:

```
do { \
                if ((v) < 0) \
                        (v) = -(-(v) >> (n)); \
                else \
                        (v) = (v) >> (n); \
        } while (0)
```

Definition at line 61 of file kern_ntptime.c.

Referenced by hardupdate(), and ntp_update_second().

9.45.1.10 #define L_SUB(v, u) ((v) -= (u))

Definition at line 58 of file kern_ntptime.c.

Referenced by ntp_update_second().

9.45.1.11 #define SHIFT_FLL 2

Definition at line 143 of file kern_ntptime.c.

Referenced by hardupdate().

9.45.1.12 #define SHIFT_PLL 4

Definition at line 142 of file kern_ntptime.c.

Referenced by hardupdate(), and ntp_update_second().

9.45.2 Typedef Documentation

9.45.2.1 typedef int64_t l_fp

Definition at line 56 of file kern_ntptime.c.

9.45.3 Function Documentation

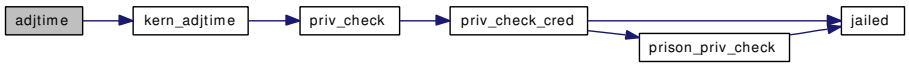
9.45.3.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_ntptime. *c*, v 1.61 2007/01/12 07:40:30 imp Exp $")

9.45.3.2 int adjtime (struct thread * *td*, struct adjtime_args * *uap*)

Definition at line 935 of file kern_ntptime.c.

References adjtime_args::delta, kern_adjtime(), and adjtime_args::olddelta.

Here is the call graph for this function:



9.45.3.3 static void hardupdate (long *offset*) [static]

Definition at line 660 of file kern_ntptime.c.

References L_ADD, L_GINT, L_LINT, L_MPY, L_RSHIFT, SHIFT_FLL, SHIFT_PLL, time_constant, time_freq, time_monitor, time_offset, time_reftime, time_second, and time_status.

Referenced by ntp_adjtime().

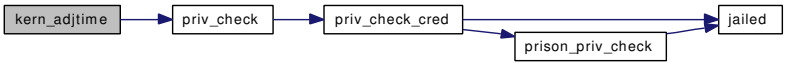
9.45.3.4 int kern_adjtime (struct thread * *td*, struct timeval * *delta*, struct timeval * *olddelta*)

Definition at line 954 of file kern_ntptime.c.

References Giant, priv_check(), and time_adjtime.

Referenced by adjtime().

Here is the call graph for this function:

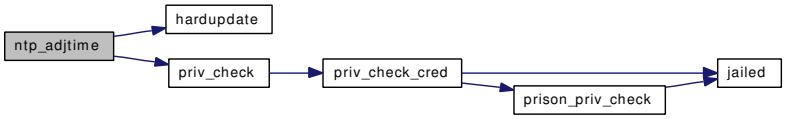


9.45.3.5 int ntp_adjtime (struct thread * *td*, struct ntp_adjtime_args * *uap*)

Definition at line 314 of file kern_ntptime.c.

References Giant, hardupdate(), L_GINT, L_LINT, priv_check(), time_constant, time_esterror, time_freq, time_maxerror, time_offset, time_precision, time_state, time_status, time_tai, and ntp_adjtime_args::tp.

Here is the call graph for this function:

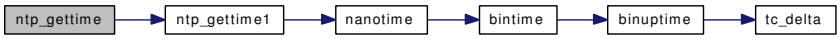


9.45.3.6 int ntp_gettime (struct thread * *td*, struct ntp_gettime_args * *uap*)

Definition at line 263 of file kern_ntptime.c.

References Giant, ntp_gettime1(), and ntp_gettime_args::ntvp.

Here is the call graph for this function:



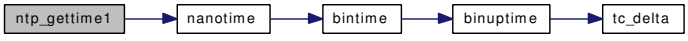
9.45.3.7 static void ntp_gettime1 (struct ntptimeval * *ntvp*) [static]

Definition at line 203 of file kern_ntptime.c.

References nanotime(), time_esterror, time_maxerror, time_state, time_status, and time_tai.

Referenced by ntp_gettime(), and ntp_sysctl().

Here is the call graph for this function:



9.45.3.8 static void ntp_init (void) [static]

Definition at line 616 of file kern_ntptime.c.

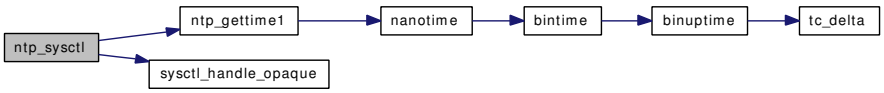
References L_CLR, time_freq, and time_offset.

9.45.3.9 static int ntp_sysctl (SYSCTL_HANDLER_ARGS) [static]

Definition at line 276 of file kern_ntptime.c.

References ntp_gettime1(), and sysctl_handle_opaque().

Here is the call graph for this function:



9.45.3.10 void ntp_update_second (int64_t * *adjustment*, time_t * *newsec*)

Definition at line 477 of file kern_ntptime.c.

References L_ADD, L_LINT, L_RSHIFT, L_SUB, SHIFT_PLL, time_adj, time_adjtime, time_constant, time_freq, time_maxerror, time_offset, time_state, time_status, and time_tai.

Referenced by tc_windup().

9.45.3.11 SYSCTL_NODE (_kern, OID_AUTO, ntp_pll, CTLFLAG_RW, 0, "")

9.45.3.12 SYSCTL_PROC (_kern_ntp_pll, OID_AUTO, gettime, CTLTYPE_OPAQUE| *CTLFLAG_RD*, 0, sizeof(struct ntptimeval), ntp_sysctl, " *S*, ntptimeval", "")

9.45.4 Variable Documentation

9.45.4.1 l_fp time_adj [static]

Definition at line 156 of file kern_ntptime.c.

Referenced by ntp_update_second().

9.45.4.2 int64_t time_adjtime [static]

Definition at line 158 of file kern_ntptime.c.

Referenced by kern_adjtime(), and ntp_update_second().

9.45.4.3 long time_constant [static]

Definition at line 149 of file kern_ntptime.c.

Referenced by hardupdate(), ntp_adjtime(), and ntp_update_second().

9.45.4.4 long time_esterror = MAXPHASE / 1000 [static]

Definition at line 152 of file kern_ntptime.c.

Referenced by ntp_adjtime(), and ntp_gettime1().

9.45.4.5 l_fp time_freq [static]

Definition at line 155 of file kern_ntptime.c.

Referenced by hardupdate(), ntp_adjtime(), ntp_init(), and ntp_update_second().

9.45.4.6 long time_maxerror = MAXPHASE / 1000 [static]

Definition at line 151 of file kern_ntptime.c.

Referenced by ntp_adjtime(), ntp_gettime1(), and ntp_update_second().

9.45.4.7 long time_monitor [static]

Definition at line 148 of file kern_ntptime.c.

Referenced by hardupdate().

9.45.4.8 l_fp time_offset [static]

Definition at line 154 of file kern_ntptime.c.

Referenced by hardupdate(), ntp_adjtime(), ntp_init(), and ntp_update_second().

9.45.4.9 long time_precision = 1 [static]

Definition at line 150 of file kern_ntptime.c.

Referenced by ntp_adjtime().

9.45.4.10 long time_reftime [static]

Definition at line 153 of file kern_ntptime.c.

Referenced by hardupdate().

9.45.4.11 int time_state = TIME_OK [static]

Definition at line 145 of file kern_ntptime.c.

Referenced by ntp_adjtime(), ntp_gettime1(), and ntp_update_second().

9.45.4.12 int time_status = STA_UNSYNC [static]

Definition at line 146 of file kern_ntptime.c.

Referenced by hardupdate(), ntp_adjtime(), ntp_gettime1(), and ntp_update_second().

9.45.4.13 long time_tai [static]

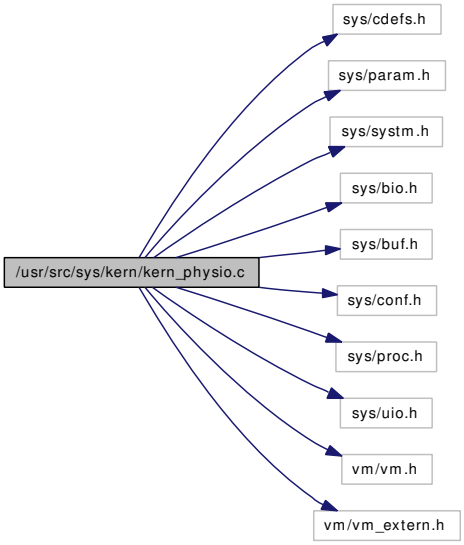
Definition at line 147 of file kern_ntptime.c.

Referenced by ntp_adjtime(), ntp_gettime1(), and ntp_update_second().

9.46 /usr/src/sys/kern/kern_physio.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/bio.h>
#include <sys/buf.h>
#include <sys/conf.h>
#include <sys/proc.h>
#include <sys/uio.h>
#include <vm/vm.h>
#include <vm/vm_extern.h>
```

Include dependency graph for kern_physio.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_physio.c,v 1.68 2005/01/06 23:35:39 imp Exp $")
- int physio (struct cdev *dev, struct uio *uio, int ioflag)

9.46.1 Function Documentation

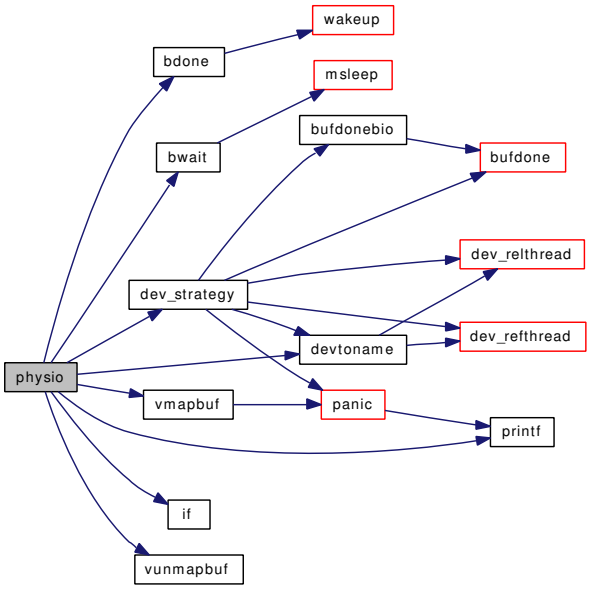
9.46.1.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_physio. *c*, v 1.68 2005/01/06 23:35:39 imp Exp $")

9.46.1.2 int physio (struct cdev * *dev*, struct uio * *uio*, int *ioflag*)

Definition at line 35 of file kern_physio.c.

References bdone(), buf, bwait(), dev_strategy(), devtoname(), if(), printf(), vmapbuf(), and vunmapbuf().

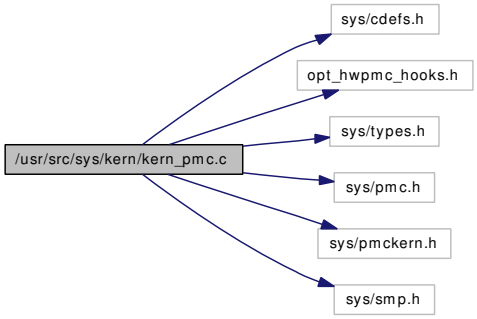
Here is the call graph for this function:



9.47 /usr/src/sys/kern/kern_pmc.c File Reference

```
#include <sys/cdefs.h>
#include "opt_hwpmc_hooks.h"
#include <sys/types.h>
#include <sys/pmc.h>
#include <sys/pmckern.h>
#include <sys/smp.h>
```

Include dependency graph for kern_pmc.c:



Defines

- #define PMC_KERNEL_VERSION 0

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_pmc.c,v 1.6 2005/12/04 02:12:43 ru Exp $")
- SX_SYSINIT (pmc,&pmc_sx,"pmc shared lock")
- int pmc_cpu_is_disabled (int cpu)
- int pmc_cpu_is_logical (int cpu)

Variables

- const int pmc_kernel_version = PMC_KERNEL_VERSION
- int(∗) pmc_hook (struct thread ∗td, int function, void ∗arg) = NULL
- int(∗) pmc_intr (int cpu, uintptr_t pc, int usermode) = NULL
- volatile cpumask_t pmc_cpumask
- volatile int pmc_ss_count
- sx pmc_sx

9.47.1 Define Documentation

9.47.1.1 #define PMC_KERNEL_VERSION 0

Definition at line 39 of file kern_pmc.c.

9.47.2 Function Documentation

9.47.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_pmc. *c*, v 1.6 2005/12/04 02:12:43 ru Exp $")

9.47.2.2 int pmc_cpu_is_disabled (int *cpu*)

Definition at line 76 of file kern_pmc.c.

9.47.2.3 int pmc_cpu_is_logical (int *cpu*)

Definition at line 86 of file kern_pmc.c.

9.47.2.4 SX_SYSINIT (pmc, & *pmc_sx*, "pmc shared lock")

9.47.3 Variable Documentation

9.47.3.1 volatile cpumask_t pmc_cpumask

Definition at line 51 of file kern_pmc.c.

9.47.3.2 int(∗) pmc_hook(struct thread ∗td, int function, void ∗arg) = NULL

Definition at line 45 of file kern_pmc.c.

9.47.3.3 int(∗) pmc_intr(int cpu, uintptr_t pc, int usermode) = NULL

Definition at line 48 of file kern_pmc.c.

9.47.3.4 const int pmc_kernel_version = PMC_KERNEL_VERSION

Definition at line 42 of file kern_pmc.c.

9.47.3.5 volatile int pmc_ss_count

Definition at line 57 of file kern_pmc.c.

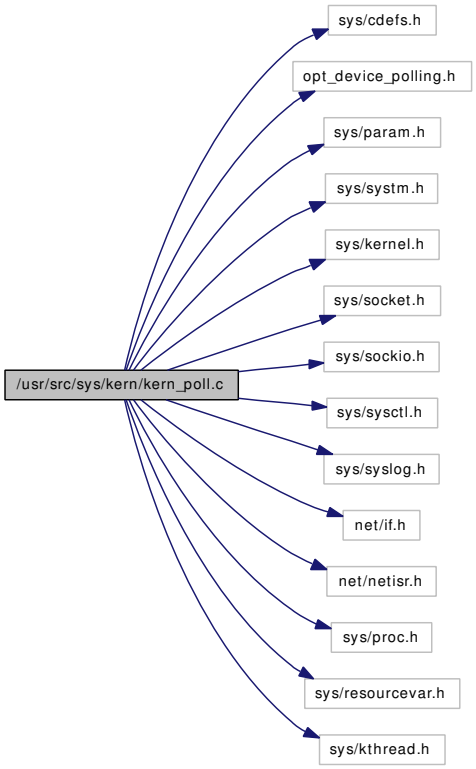
9.47.3.6 struct sx pmc_sx

Definition at line 68 of file kern_pmc.c.

9.48 /usr/src/sys/kern/kern_poll.c File Reference

```
#include <sys/cdefs.h>
#include "opt_device_polling.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/socket.h>
#include <sys/sockio.h>
#include <sys/sysctl.h>
#include <sys/syslog.h>
#include <net/if.h>
#include <net/netisr.h>
#include <sys/proc.h>
#include <sys/resourcevar.h>
#include <sys/kthread.h>
```

Include dependency graph for kern_poll.c:



Data Structures

- struct pollrec

Defines

- #define MIN_POLL_BURST_MAX 10
- #define MAX_POLL_BURST_MAX 1000
- #define POLL_LIST_LEN 128

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_poll.c,v 1.28 2006/12/06 06:34:55 julian Exp $")
- static void netisr_poll (void)
- static void netisr_pollmore (void)
- static int poll_switch (SYSCTL_HANDLER_ARGS)
- void hardclock_device_poll (void)
- void ether_poll (int)
- SYSCTL_NODE (_kern, OID_AUTO, polling, CTLFLAG_RW, 0,"Device polling parameters")
- SYSCTL_UINT (_kern_polling, OID_AUTO, burst, CTLFLAG_RD,&poll_burst, 0,"Current polling burst size")
- static int poll_burst_max_sysctl (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern_polling, OID_AUTO, burst_max, CTLTYPE_UINT|CTLFLAG_RW, 0, sizeof(uint32_t), poll_burst_max_sysctl,"I","Max Polling burst size")
- static int poll_each_burst_sysctl (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern_polling, OID_AUTO, each_burst, CTLTYPE_UINT|CTLFLAG_RW, 0, sizeof(uint32_t), poll_each_burst_sysctl,"I","Max size of each burst")
- SYSCTL_UINT (_kern_polling, OID_AUTO, idle_poll, CTLFLAG_RW,&poll_in_idle_loop, 0,"Enable device polling in idle loop")
- static int user_frac_sysctl (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern_polling, OID_AUTO, user_frac, CTLTYPE_UINT|CTLFLAG_RW, 0, sizeof(uint32_t), user_frac_sysctl,"I","Desired user fraction of cpu time")
- static int reg_frac_sysctl (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern_polling, OID_AUTO, reg_frac, CTLTYPE_UINT|CTLFLAG_RW, 0, sizeof(uint32_t), reg_frac_sysctl,"I","Every this many cycles check registers")
- SYSCTL_UINT (_kern_polling, OID_AUTO, short_ticks, CTLFLAG_RD,&short_ticks, 0,"Hardclock ticks shorter than they should be")
- SYSCTL_UINT (_kern_polling, OID_AUTO, lost_polls, CTLFLAG_RD,&lost_polls, 0,"How many times we would have lost a poll tick")
- SYSCTL_UINT (_kern_polling, OID_AUTO, pending_polls, CTLFLAG_RD,&pending_polls, 0,"Do we need to poll again")
- SYSCTL_INT (_kern_polling, OID_AUTO, residual_burst, CTLFLAG_RD,&residual_burst, 0,"# of residual cycles in burst")
- SYSCTL_UINT (_kern_polling, OID_AUTO, handlers, CTLFLAG_RD,&poll_handlers, 0,"Number of registered poll handlers")
- SYSCTL_PROC (_kern_polling, OID_AUTO, enable, CTLTYPE_UINT|CTLFLAG_RW, 0, sizeof(int), poll_switch,"I","Switch polling for all interfaces")
- SYSCTL_UINT (_kern_polling, OID_AUTO, phase, CTLFLAG_RD,&phase, 0,"Polling phase")
- SYSCTL_UINT (_kern_polling, OID_AUTO, suspect, CTLFLAG_RD,&suspect, 0,"suspect event")
- SYSCTL_UINT (_kern_polling, OID_AUTO, stalled, CTLFLAG_RD,&stalled, 0,"potential stalls")

- SYSCTL_UINT (_kern_polling, OID_AUTO, idlepoll_sleeping, CTLFLAG_RD,&idlepoll_-sleeping, 0,"idlepoll is sleeping")
- static void init_device_poll (void)
- int ether_poll_register (poll_handler_t *h, struct ifnet *ifp)
- int ether_poll_deregister (struct ifnet *ifp)
- static void poll_idle (void)

Variables

- static struct mtx poll_mtx
- static uint32_t poll_burst = 5
- static uint32_t poll_burst_max = 150
- static uint32_t poll_each_burst = 5
- static uint32_t poll_in_idle_loop = 0
- static uint32_t user_frac = 50
- static uint32_t reg_frac_count = 0
- static uint32_t reg_frac = 20
- static uint32_t short_ticks
- static uint32_t lost_polls
- static uint32_t pending_polls
- static int residual_burst = 0
- static uint32_t poll_handlers
- static int polling = 0
- static uint32_t phase
- static uint32_t suspect
- static uint32_t stalled
- static uint32_t idlepoll_sleeping
- static struct pollrec pr [POLL_LIST_LEN]
- static struct timeval poll_start_t
- static struct proc * idlepoll
- static struct kproc_desc idlepoll_kp

9.48.1 Define Documentation

9.48.1.1 #define MAX_POLL_BURST_MAX 1000

Definition at line 99 of file kern_poll.c.

Referenced by poll_burst_max_sysctl().

9.48.1.2 #define MIN_POLL_BURST_MAX 10

Definition at line 98 of file kern_poll.c.

Referenced by poll_burst_max_sysctl().

9.48.1.3 #define POLL_LIST_LEN 128

Definition at line 252 of file kern_poll.c.

Referenced by ether_poll_register().

9.48.2 Function Documentation

9.48.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_poll. *c*, v 1.28 2006/12/06 06:34:55 julian Exp $")

9.48.2.2 void ether_poll (int)

Definition at line 328 of file kern_poll.c.

References poll_each_burst, poll_mtx, and pr.

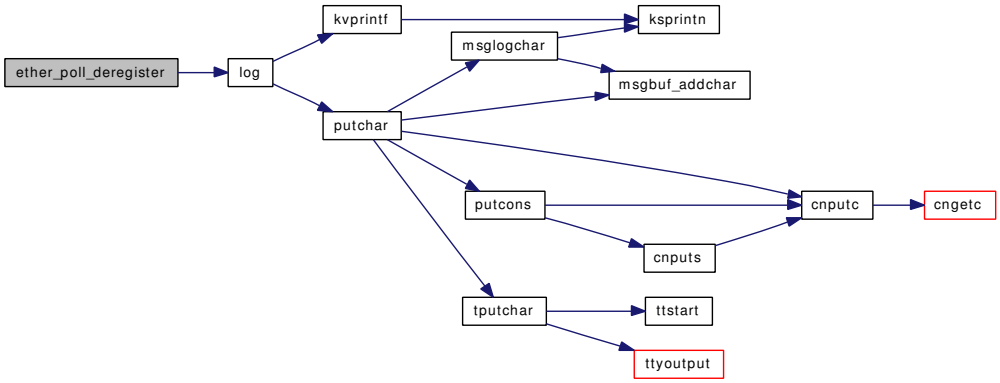
Referenced by poll_idle().

9.48.2.3 int ether_poll_deregister (struct ifnet * *ifp*)

Definition at line 501 of file kern_poll.c.

References pollrec::ifp, log(), poll_mtx, and pr.

Here is the call graph for this function:

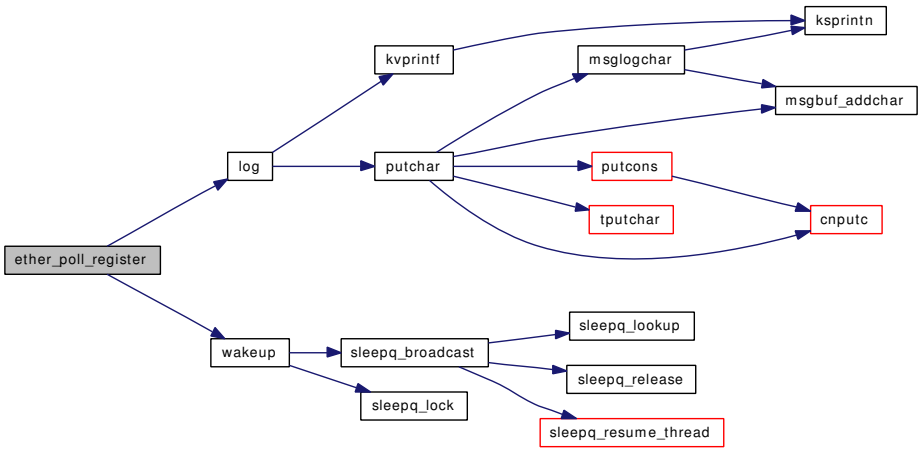


9.48.2.4 int ether_poll_register (poll_handler_t * *h*, struct ifnet * *ifp*)

Definition at line 452 of file kern_poll.c.

References pollrec::handler, pollrec::ifp, log(), POLL_LIST_LEN, poll_mtx, pr, and wakeup().

Here is the call graph for this function:



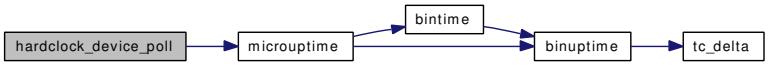
9.48.2.5 void hardclock_device_poll (void)

Definition at line 287 of file kern_poll.c.

References hz, and microuptime().

Referenced by hardclock().

Here is the call graph for this function:

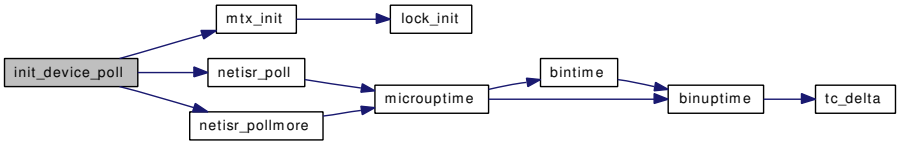


9.48.2.6 static void init_device_poll (void) [static]

Definition at line 261 of file kern_poll.c.

References mtx_init(), netisr_poll(), netisr_pollmore(), and poll_mtx.

Here is the call graph for this function:



9.48.2.7 static void netisr_poll (void) [static]

Definition at line 415 of file kern_poll.c.

References microuptime(), poll_burst, poll_each_burst, poll_mtx, and pr.

Referenced by init_device_poll().

Here is the call graph for this function:

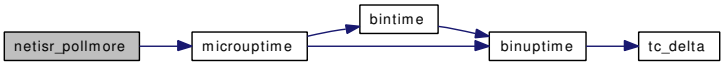
9.48.2.8 void netisr_pollmore (void) [static]

Definition at line 364 of file kern_poll.c.

References hz, microuptime(), poll_burst, poll_burst_max, and poll_mtx.

Referenced by init_device_poll().

Here is the call graph for this function:



9.48.2.9 static int poll_burst_max_sysctl (SYSCTL_HANDLER_ARGS) [static]

Definition at line 111 of file kern_poll.c.

References MAX_POLL_BURST_MAX, MIN_POLL_BURST_MAX, poll_burst, poll_burst_max, poll_each_burst, poll_mtx, and sysctl_handle_int().

Here is the call graph for this function:

9.48.2.10 static int poll_each_burst_sysctl (SYSCTL_HANDLER_ARGS) [static]

Definition at line 135 of file kern_poll.c.

References poll_burst_max, poll_each_burst, poll_mtx, and sysctl_handle_int().

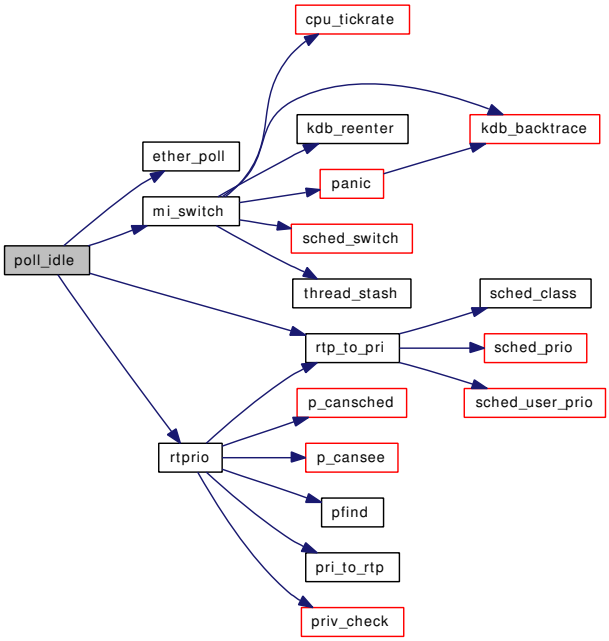
Here is the call graph for this function:

9.48.2.11 static void poll_idle (void) [static]

Definition at line 576 of file kern_poll.c.

References ether_poll(), hz, mi_switch(), poll_each_burst, rtp_to_pri(), rtprio(), sched_lock, and td.

Here is the call graph for this function:

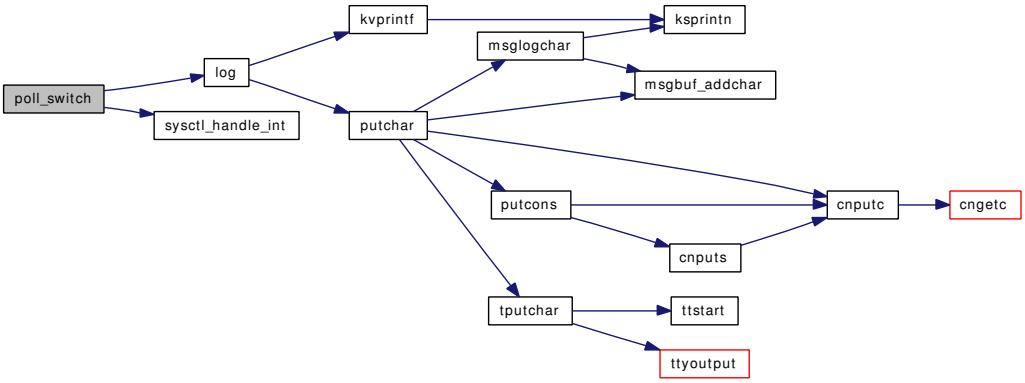


9.48.2.12 static int poll_switch (SYSCTL_HANDLER_ARGS) [static]

Definition at line 532 of file kern_poll.c.

References log(), and sysctl_handle_int().

Here is the call graph for this function:

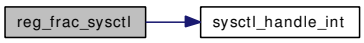


9.48.2.13 static int reg_frac_sysctl (SYSCTL_HANDLER_ARGS) [static]

Definition at line 188 of file kern_poll.c.

References hz, poll_mtx, and sysctl_handle_int().

Here is the call graph for this function:



9.48.2.14 SYSCTL_INT (_kern_polling, OID_AUTO, residual_burst, CTLFLAG_RD, & *residual_burst*, 0, "# of residual cycles in burst")

9.48.2.15 SYSCTL_NODE (_kern, OID_AUTO, polling, CTLFLAG_RW, 0, "Device polling parameters")

9.48.2.16 SYSCTL_PROC (_kern_polling, OID_AUTO, enable, CTLTYPE_UINT| *CTLFLAG_RW*, 0, sizeof(int), poll_switch, "I", "Switch polling for all interfaces")

9.48.2.17 SYSCTL_PROC (_kern_polling, OID_AUTO, reg_frac, CTLTYPE_UINT| *CTLFLAG_RW*, 0, sizeof(uint32_t), reg_frac_sysctl, "I", "Every this many cycles check registers")

9.48.2.18 SYSCTL_PROC (_kern_polling, OID_AUTO, user_frac, CTLTYPE_UINT| *CTLFLAG_RW*, 0, sizeof(uint32_t), user_frac_sysctl, "I", "Desired user fraction of cpu time")

9.48.2.19 SYSCTL_PROC (_kern_polling, OID_AUTO, each_burst, CTLTYPE_UINT| *CTLFLAG_RW*, 0, sizeof(uint32_t), poll_each_burst_sysctl, "I", "Max size of each burst")

9.48.2.20 SYSCTL_PROC (_kern_polling, OID_AUTO, burst_max, CTLTYPE_UINT| *CTLFLAG_RW*, 0, sizeof(uint32_t), poll_burst_max_sysctl, "I", "Max Polling burst size")

9.48.2.21 SYSCTL_UINT (_kern_polling, OID_AUTO, idlepoll_sleeping, CTLFLAG_RD, & *idlepoll_sleeping*, 0, "idlepoll is sleeping")

9.48.2.22 SYSCTL_UINT (_kern_polling, OID_AUTO, stalled, CTLFLAG_RD, & *stalled*, 0, "potential stalls")

9.48.2.23 SYSCTL_UINT (_kern_polling, OID_AUTO, suspect, CTLFLAG_RD, & *suspect*, 0, "suspect event")

9.48.2.24 SYSCTL_UINT (_kern_polling, OID_AUTO, phase, CTLFLAG_RD, & *phase*, 0, "Polling phase")

9.48.2.25 SYSCTL_UINT (_kern_polling, OID_AUTO, handlers, CTLFLAG_RD, & *poll_handlers*, 0, "Number of registered poll handlers")

9.48.2.26 SYSCTL_UINT (_kern_polling, OID_AUTO, pending_polls, CTLFLAG_RD, & *pending_polls*, 0, "Do we need to poll again")

9.48.2.27 SYSCTL_UINT (_kern_polling, OID_AUTO, lost_polls, CTLFLAG_RD, & *lost_polls*, 0, "How many times we would have lost a poll tick")

9.48.2.28 SYSCTL_UINT (_kern_polling, OID_AUTO, short_ticks, CTLFLAG_RD, & *short_ticks*, 0, "Hardclock ticks shorter than they should be")

9.48.2.29 SYSCTL_UINT (_kern_polling, OID_AUTO, idle_poll, CTLFLAG_RW, & *poll_in_idle_loop*, 0, "Enable device polling in idle loop")

9.48.2.30 SYSCTL_UINT (_kern_polling, OID_AUTO, burst, CTLFLAG_RD, & *poll_burst*, 0, "Current polling burst size")

9.48.2.31 static int user_frac_sysctl (SYSCTL_HANDLER_ARGS) `[static]`

Definition at line 165 of file kern_poll.c.

References poll_mtx, and sysctl_handle_int().

Here is the call graph for this function:

9.48.3 Variable Documentation

9.48.3.1 struct proc* idlepoll [static]

Definition at line 601 of file kern_poll.c.

9.48.3.2 struct kproc_desc idlepoll_kp [static]

Initial value:

```
{
        "idlepoll",
        poll_idle,
        &idlepoll
}
```

Definition at line 602 of file kern_poll.c.

9.48.3.3 uint32_t idlepoll_sleeping [static]

Definition at line 247 of file kern_poll.c.

9.48.3.4 uint32_t lost_polls [static]

Definition at line 215 of file kern_poll.c.

9.48.3.5 uint32_t pending_polls [static]

Definition at line 219 of file kern_poll.c.

9.48.3.6 uint32_t phase [static]

Definition at line 235 of file kern_poll.c.

9.48.3.7 uint32_t poll_burst = 5 [static]

Definition at line 101 of file kern_poll.c.

Referenced by netisr_poll(), netisr_pollmore(), and poll_burst_max_sysctl().

9.48.3.8 uint32_t poll_burst_max = 150 [static]

Definition at line 102 of file kern_poll.c.

Referenced by netisr_pollmore(), poll_burst_max_sysctl(), and poll_each_burst_sysctl().

9.48.3.9 uint32_t poll_each_burst = 5 [static]

Definition at line 103 of file kern_poll.c.

Referenced by ether_poll(), netisr_poll(), poll_burst_max_sysctl(), poll_each_burst_sysctl(), and poll_-idle().

9.48.3.10 uint32_t poll_handlers [static]

Definition at line 227 of file kern_poll.c.

9.48.3.11 uint32_t poll_in_idle_loop = 0 [static]

Definition at line 160 of file kern_poll.c.

9.48.3.12 struct mtx poll_mtx [static]

Definition at line 55 of file kern_poll.c.

Referenced by ether_poll(), ether_poll_deregister(), ether_poll_register(), init_device_poll(), netisr_poll(), netisr_pollmore(), poll_burst_max_sysctl(), poll_each_burst_sysctl(), reg_frac_sysctl(), and user_frac_-sysctl().

9.48.3.13 struct timeval poll_start_t [static]

Definition at line 361 of file kern_poll.c.

9.48.3.14 int polling = 0 [static]

Definition at line 231 of file kern_poll.c.

9.48.3.15 struct pollrec pr[POLL_LIST_LEN] [static]

Definition at line 258 of file kern_poll.c.

Referenced by ether_poll(), ether_poll_deregister(), ether_poll_register(), jail(), jail_attach(), mb_-reclaim(), net_init_domain(), netisr_poll(), pf_proto_register(), pf_proto_unregister(), pfctlinput(), pfctlin-put2(), pffasttimo(), pffindproto(), pffindtype(), pfslowtimo(), prison_canseemount(), prison_complete(), prison_enforce_statfs(), prison_find(), sofree(), soreceive_generic(), soreceive_rcvoob(), sorflush(), soshutdown(), sysctl_hostname(), sysctl_jail_list(), and sysctl_kern_securelvl().

9.48.3.16 uint32_t reg_frac = 20 [static]

Definition at line 187 of file kern_poll.c.

9.48.3.17 uint32_t reg_frac_count = 0 [static]

Definition at line 186 of file kern_poll.c.

9.48.3.18 int residual_burst = 0 [static]

Definition at line 223 of file kern_poll.c.

9.48.3.19 uint32_t short_ticks [static]

Definition at line 211 of file kern_poll.c.

9.48.3.20 uint32_t stalled [static]

Definition at line 243 of file kern_poll.c.

9.48.3.21 uint32_t suspect [static]

Definition at line 239 of file kern_poll.c.

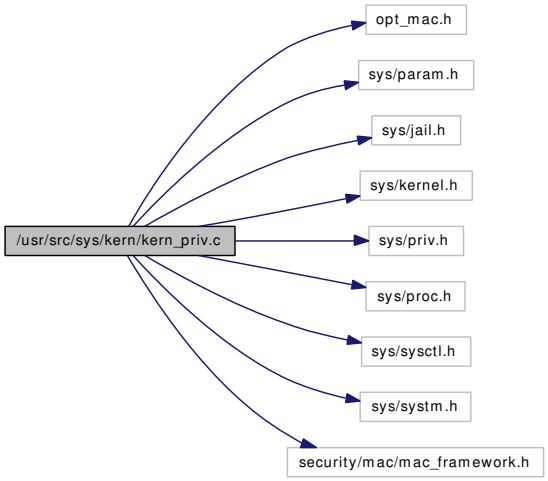
9.48.3.22 uint32_t user_frac = 50 [static]

Definition at line 164 of file kern_poll.c.

9.49 /usr/src/sys/kern/kern_priv.c File Reference

```
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/jail.h>
#include <sys/kernel.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/sysctl.h>
#include <sys/systm.h>
#include <security/mac/mac_framework.h>
```

Include dependency graph for kern_priv.c:



Functions

- SYSCTL_INT (_security_bsd, OID_AUTO, suser_enabled, CTLFLAG_RW,&suser_enabled, 0,"processes with uid 0 have privilege")
- TUNABLE_INT ("security.bsd.suser_enabled",&suser_enabled)
- int priv_check_cred (struct ucred *cred, int priv, int flags)
- int priv_check (struct thread *td, int priv)
- int suser_cred (struct ucred *cred, int flags)
- int suser (struct thread *td)

Variables

- int suser_enabled = 1

9.49.1 Function Documentation

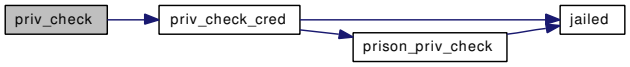
9.49.1.1 int priv_check (struct thread * *td*, int *priv*)

Definition at line 127 of file kern_priv.c.

References priv_check_cred().

Referenced by _do_lock_pp(), acct(), cnioctl(), create_thread(), do_unlock_pp(), donice(), fhopen(), fhstat(), firmware_get(), getfh(), jail_attach(), kenv(), kern_adjtime(), kern_clock_settime(), kern_fhstatfs(), kern_fstatfs(), kern_kldload(), kern_kldunload(), kern_mknod(), kern_settimeofday(), kern_statfs(), lgetfh(), ntp_adjtime(), ptsopen(), reboot(), rtprio(), rtprio_thread(), sem_perm(), setdomainname(), sysctl_kern_msgbuf(), sysctl_root(), ttioctl(), ttyopen(), ttysioctl(), unmount(), vfs_domount(), vfs_suser(), and vn_stat().

Here is the call graph for this function:



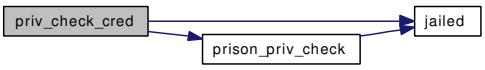
9.49.1.2 int priv_check_cred (struct ucred * *cred*, int *priv*, int *flags*)

Definition at line 64 of file kern_priv.c.

References jailed(), prison_priv_check(), and suser_enabled.

Referenced by can_hardlink(), chroot(), cr_cansignal(), cr_seeothergids(), cr_seeotheruids(), do_unlink(), extattr_check_cred(), falloc(), fork1(), ipcperm(), kern_msgctl(), kern_setgroups(), kern_setrlimit(), mqfs_setattr(), p_candebug(), p_cansched(), priv_check(), revoke(), setegid(), seteuid(), setfflags(), setgid(), setlogin(), setregid(), setresgid(), setresuid(), setreuid(), setuid(), suser_cred(), sysctl_root(), vaccess(), vaccess_acl_posix1e(), and vfs_domount().

Here is the call graph for this function:

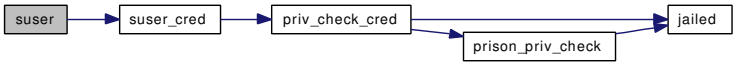


9.49.1.3 int suser (struct thread * *td*)

Definition at line 148 of file kern_priv.c.

References suser_cred().

Here is the call graph for this function:



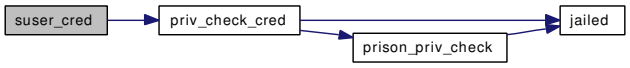
9.49.1.4 int suser_cred (struct ucred * *cred*, int *flags*)

Definition at line 141 of file kern_priv.c.

References priv_check_cred().

Referenced by suser().

Here is the call graph for this function:



9.49.1.5 SYSCTL_INT (_security_bsd, OID_AUTO, suser_enabled, CTLFLAG_RW, & *suser_enabled*, 0, "processes with uid 0 have privilege")

9.49.1.6 TUNABLE_INT ("security.bsd.suser_enabled", & *suser_enabled*)

9.49.2 Variable Documentation

9.49.2.1 int suser_enabled = 1

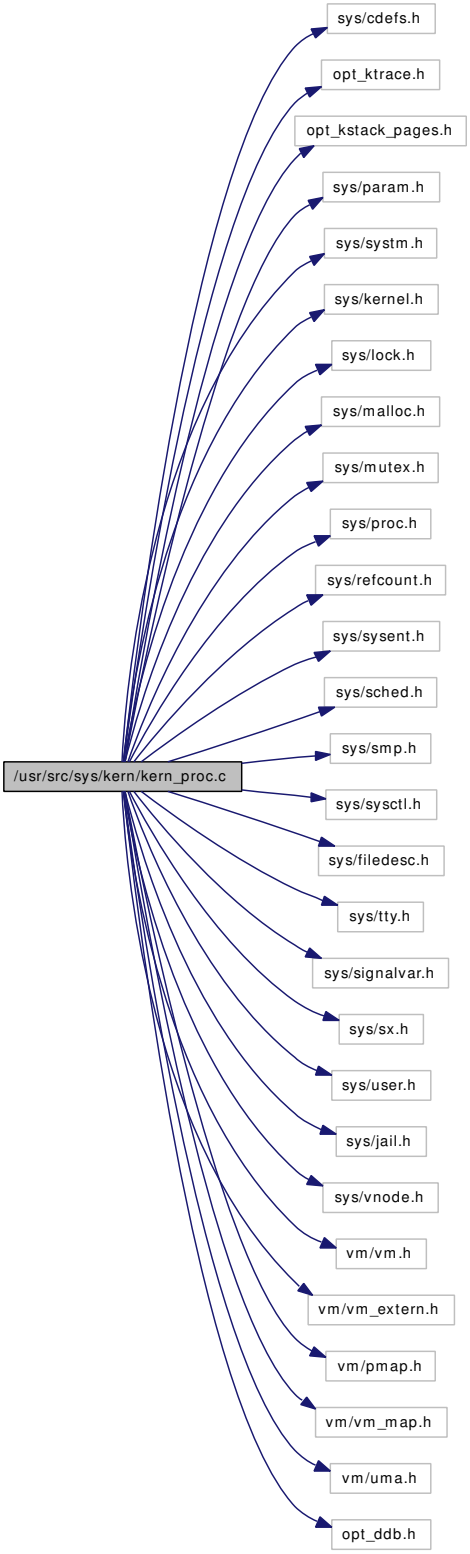
Definition at line 54 of file kern_priv.c.

Referenced by priv_check_cred().

9.50 /usr/src/sys/kern/kern_proc.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ktrace.h"
#include "opt_kstack_pages.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/refcount.h>
#include <sys/sysent.h>
#include <sys/sched.h>
#include <sys/smp.h>
#include <sys/sysctl.h>
#include <sys/filedesc.h>
#include <sys/tty.h>
#include <sys/signalvar.h>
#include <sys/sx.h>
#include <sys/user.h>
#include <sys/jail.h>
#include <sys/vnode.h>
#include <vm/vm.h>
#include <vm/vm_extern.h>
#include <vm/pmap.h>
#include <vm/vm_map.h>
#include <vm/uma.h>
#include "opt_ddb.h"
```

Include dependency graph for kern_proc.c:


/usr/src/sys/kern/kern_proc.c
sys/cdefs.h
opt_ktrace.h
opt_kstack_pages.h
sys/param.h
sys/systm.h
sys/kernel.h
sys/lock.h
sys/malloc.h
sys/mutex.h
sys/proc.h
sys/refcount.h
sys/sysent.h
sys/sched.h
sys/smp.h
sys/sysctl.h
sys/filedesc.h
sys/tty.h
sys/signalvar.h
sys/sx.h
sys/user.h
sys/jail.h
sys/vnode.h
vm/vm.h
vm/vm_extern.h
vm/pmap.h
vm/vm_map.h
vm/uma.h
opt_ddb.h

Defines

- #define KERN_PROC_ZOMBMASK 0x3
- #define KERN_PROC_NOTHREADS 0x4

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_proc.c,v 1.246 2006/12/06 06:34:55 julian Exp $")
- MALLOC_DEFINE (M_PGRP,"pgrp","process group header")
- MALLOC_DEFINE (M_SESSION,"session","session header")
- static MALLOC_DEFINE (M_PROC,"proc","Proc structures")
- MALLOC_DEFINE (M_SUBPROC,"subproc","Proc sub-structures")
- static void doenterpgrp (struct proc *, struct pgrp *)
- static void orphanpg (struct pgrp *pg)
- static void fill_kinfo_proc_only (struct proc *p, struct kinfo_proc *kp)
- static void fill_kinfo_thread (struct thread *td, struct kinfo_proc *kp)
- static void pgadjustjobc (struct pgrp *pgrp, int entering)
- static void pgdelete (struct pgrp *)
- static int proc_ctor (void *mem, int size, void *arg, int flags)
- static void proc_dtor (void *mem, int size, void *arg)
- static int proc_init (void *mem, int size, int flags)
- static void proc_fini (void *mem, int size)
- SYSCTL_INT (_kern, OID_AUTO, kstack_pages, CTLFLAG_RD,&kstack_pages, 0,"")
- CTASSERT (sizeof(struct kinfo_proc)==KINFO_PROC_SIZE)
- void procinit ()
- int inferior (struct proc *p)
- proc * pfind (pid_t pid)
- pgrp * pgfind (pid_t pgid)
- int enterpgrp (struct proc *p, pid_t pgid, struct pgrp *pgrp, struct session *sess)
- int enterthispgrp (struct proc *p, struct pgrp *pgrp)
- int leavepgrp (struct proc *p)
- void fixjobc (struct proc *p, struct pgrp *pgrp, int entering)
- void sessrele (struct session *s)
- void fill_kinfo_proc (struct proc *p, struct kinfo_proc *kp)
- pstats * pstats_alloc (void)
- void pstats_fork (struct pstats *src, struct pstats *dst)
- void pstats_free (struct pstats *ps)
- proc * zpfind (pid_t pid)
- static int sysctl_out_proc (struct proc *p, struct sysctl_req *req, int flags)
- static int sysctl_kern_proc (SYSCTL_HANDLER_ARGS)
- pargs * pargs_alloc (int len)
- void pargs_free (struct pargs *pa)
- void pargs_hold (struct pargs *pa)
- void pargs_drop (struct pargs *pa)
- static int sysctl_kern_proc_args (SYSCTL_HANDLER_ARGS)
- static int sysctl_kern_proc_pathname (SYSCTL_HANDLER_ARGS)
- static int sysctl_kern_proc_sv_name (SYSCTL_HANDLER_ARGS)
- static SYSCTL_NODE (_kern, KERN_PROC, proc, CTLFLAG_RD, 0,"Process table")
- SYSCTL_PROC (_kern_proc, KERN_PROC_ALL, all, CTLFLAG_RD|CTLTYPE_STRUCT, 0, 0, sysctl_kern_proc,"S,proc","Return entire process table")

- static SYSCTL_NODE (_kern_proc, KERN_PROC_GID, gid, CTLFLAG_RD, sysctl_kern_proc,"Process table")
- static SYSCTL_NODE (_kern_proc, KERN_PROC_PGRP, pgrp, CTLFLAG_RD, sysctl_kern_proc,"Process table")
- static SYSCTL_NODE (_kern_proc, KERN_PROC_RGID, rgid, CTLFLAG_RD, sysctl_kern_proc,"Process table")
- static SYSCTL_NODE (_kern_proc, KERN_PROC_SESSION, sid, CTLFLAG_RD, sysctl_kern_proc,"Process table")
- static SYSCTL_NODE (_kern_proc, KERN_PROC_TTY, tty, CTLFLAG_RD, sysctl_kern_proc,"Process table")
- static SYSCTL_NODE (_kern_proc, KERN_PROC_UID, uid, CTLFLAG_RD, sysctl_kern_proc,"Process table")
- static SYSCTL_NODE (_kern_proc, KERN_PROC_RUID, ruid, CTLFLAG_RD, sysctl_kern_proc,"Process table")
- static SYSCTL_NODE (_kern_proc, KERN_PROC_PID, pid, CTLFLAG_RD, sysctl_kern_proc,"Process table")
- static SYSCTL_NODE (_kern_proc, KERN_PROC_PROC, proc, CTLFLAG_RD, sysctl_kern_proc,"Return process table, no threads")
- static SYSCTL_NODE (_kern_proc, KERN_PROC_ARGS, args, CTLFLAG_RW|CTLFLAG_ANYBODY, sysctl_kern_proc_args,"Process argument list")
- static SYSCTL_NODE (_kern_proc, KERN_PROC_PATHNAME, pathname, CTLFLAG_RD, sysctl_kern_proc_pathname,"Process executable path")
- static SYSCTL_NODE (_kern_proc, KERN_PROC_SV_NAME, sv_name, CTLFLAG_RD, sysctl_kern_proc_sv_name,"Process syscall vector name (ABI type)")
- static SYSCTL_NODE (_kern_proc,(KERN_PROC_GID|KERN_PROC_INC_THREAD), gid_td, CTLFLAG_RD, sysctl_kern_proc,"Process table")
- static SYSCTL_NODE (_kern_proc,(KERN_PROC_PGRP|KERN_PROC_INC_THREAD), pgrp_td, CTLFLAG_RD, sysctl_kern_proc,"Process table")
- static SYSCTL_NODE (_kern_proc,(KERN_PROC_RGID|KERN_PROC_INC_THREAD), rgid_td, CTLFLAG_RD, sysctl_kern_proc,"Process table")
- static SYSCTL_NODE (_kern_proc,(KERN_PROC_SESSION|KERN_PROC_INC_THREAD), sid_td, CTLFLAG_RD, sysctl_kern_proc,"Process table")
- static SYSCTL_NODE (_kern_proc,(KERN_PROC_TTY|KERN_PROC_INC_THREAD), tty_td, CTLFLAG_RD, sysctl_kern_proc,"Process table")
- static SYSCTL_NODE (_kern_proc,(KERN_PROC_UID|KERN_PROC_INC_THREAD), uid_td, CTLFLAG_RD, sysctl_kern_proc,"Process table")
- static SYSCTL_NODE (_kern_proc,(KERN_PROC_RUID|KERN_PROC_INC_THREAD), ruid_td, CTLFLAG_RD, sysctl_kern_proc,"Process table")
- static SYSCTL_NODE (_kern_proc,(KERN_PROC_PID|KERN_PROC_INC_THREAD), pid_td, CTLFLAG_RD, sysctl_kern_proc,"Process table")
- static SYSCTL_NODE (_kern_proc,(KERN_PROC_PROC|KERN_PROC_INC_THREAD), proc_td, CTLFLAG_RD, sysctl_kern_proc,"Return process table, no threads")

Variables

- pidhashhead * pidhashtbl
- u_long pidhash
- pgrphashhead * pgrphashtbl
- u_long pgrphash
- proclist allproc
- proclist zombproc

- sx allproc_lock
- sx proctree_lock
- mtx ppeers_lock
- uma_zone_t proc_zone
- uma_zone_t ithread_zone
- int kstack_pages = KSTACK_PAGES

9.50.1 Define Documentation

9.50.1.1 #define KERN_PROC_NOTHREADS 0x4

Definition at line 886 of file kern_proc.c.

Referenced by sysctl_kern_proc(), and sysctl_out_proc().

9.50.1.2 #define KERN_PROC_ZOMBMASK 0x3

Definition at line 885 of file kern_proc.c.

Referenced by sysctl_out_proc().

9.50.2 Function Documentation

9.50.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_proc. *c*, v 1.246 2006/12/06 06:34:55 julian Exp $")

9.50.2.2 CTASSERT (sizeof(struct kinfo_proc) = =KINFO_PROC_SIZE)

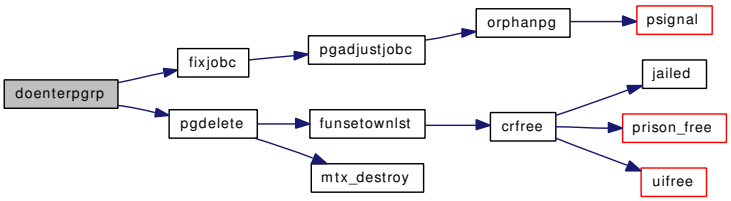
9.50.2.3 static void doenterpgrp (struct proc *, struct pgrp *) [static]

Definition at line 375 of file kern_proc.c.

References fixjobc(), Giant, pgdelete(), and proctree_lock.

Referenced by enterpgrp(), and enterthispgrp().

Here is the call graph for this function:



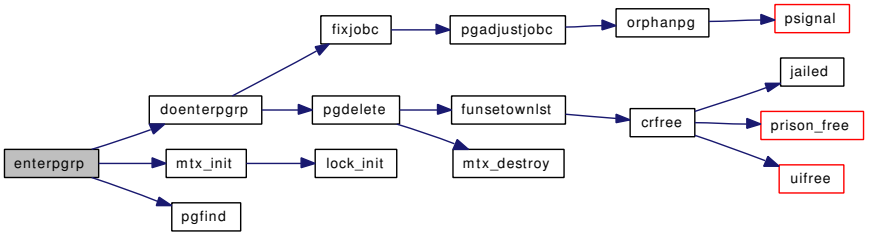
9.50.2.4 int enterpgrp (struct proc * *p*, pid_t *pgid*, struct pgrp * *pgrp*, struct session * *sess*)

Definition at line 275 of file kern_proc.c.

References doenterpgrp(), Giant, mtx_init(), pgfind(), and proctree_lock.

Referenced by setpgid(), and setsid().

Here is the call graph for this function:



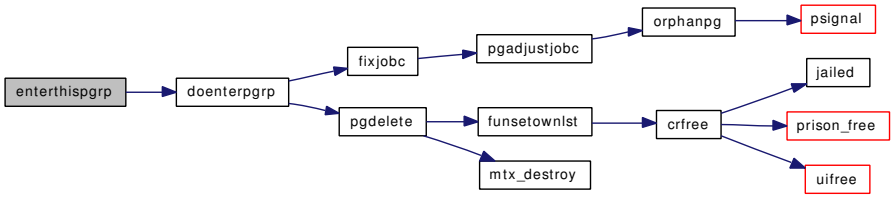
9.50.2.5 int enterthispgrp (struct proc * *p*, struct pgrp * *pgrp*)

Definition at line 348 of file kern_proc.c.

References doenterpgrp(), and proctree_lock.

Referenced by setpgid().

Here is the call graph for this function:

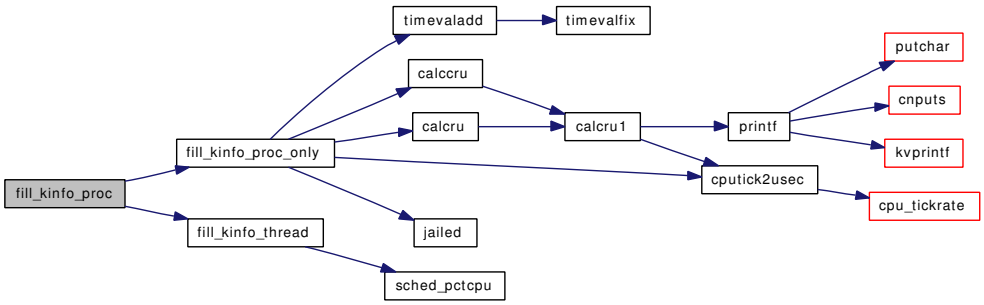


9.50.2.6 void fill_kinfo_proc (struct proc * *p*, struct kinfo_proc * *kp*)

Definition at line 830 of file kern_proc.c.

References fill_kinfo_proc_only(), fill_kinfo_thread(), and sched_lock.

Here is the call graph for this function:



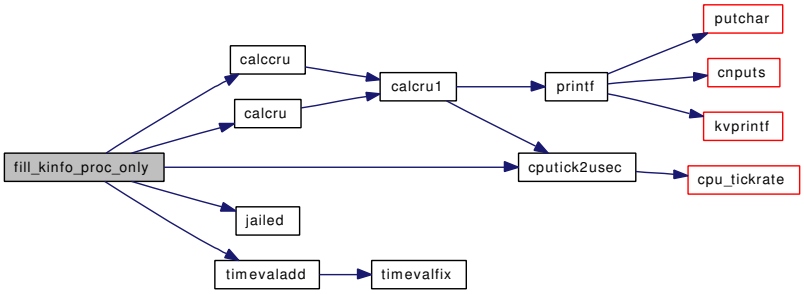
9.50.2.7 static void fill_kinfo_proc_only (struct proc * *p*, struct kinfo_proc * *kp*) [static]

Definition at line 621 of file kern_proc.c.

References boottime, calccru(), calcru(), cputick2usec(), jailed(), sched_lock, and timevaladd().

Referenced by fill_kinfo_proc(), and sysctl_out_proc().

Here is the call graph for this function:



9.50.2.8 static void fill_kinfo_thread (struct thread * *td*, struct kinfo_proc * *kp*) [static]

Definition at line 760 of file kern_proc.c.

References sched_pctcpu().

Referenced by fill_kinfo_proc(), and sysctl_out_proc().

Here is the call graph for this function:

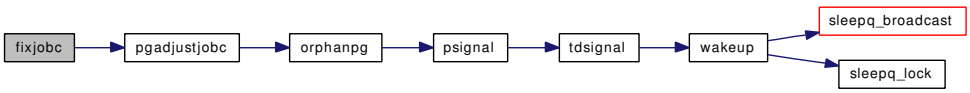
9.50.2.9 void fixjobc (struct proc * *p*, struct pgrp * *pgrp*, int *entering*)

Definition at line 497 of file kern_proc.c.

References pgadjustjobc(), and proctree_lock.

Referenced by doenterpgrp(), and exit1().

Here is the call graph for this function:



9.50.2.10 int inferior (struct proc * *p*)

Definition at line 211 of file kern_proc.c.

References proctree_lock.

Referenced by setpgid().

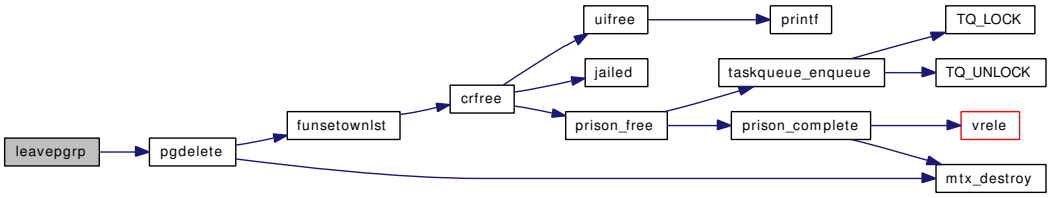
9.50.2.11 int leavepgrp (struct proc * *p*)

Definition at line 416 of file kern_proc.c.

References Giant, pgdelete(), and proctree_lock.

Referenced by kern_wait().

Here is the call graph for this function:



9.50.2.12 MALLOC_DEFINE (M_SUBPROC, "subproc", "Proc sub-structures")

9.50.2.13 static MALLOC_DEFINE (M_PROC, "proc", "Proc structures") [static]

9.50.2.14 MALLOC_DEFINE (M_SESSION, "session", "session header")

9.50.2.15 MALLOC_DEFINE (M_PGRP, "pgrp", "process group header")

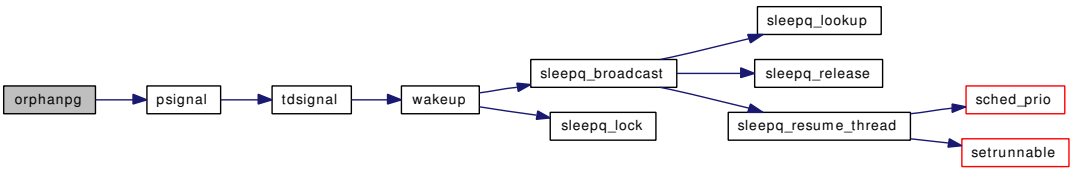
9.50.2.16 static void orphanpg (struct pgrp * *pg*) [static]

Definition at line 545 of file kern_proc.c.

References psignal().

Referenced by pgadjustjobc().

Here is the call graph for this function:



9.50.2.17 struct pargs* pargs_alloc (int *len*)

Definition at line 1117 of file kern_proc.c.

Referenced by sysctl_kern_proc_args().

9.50.2.18 void pargs_drop (struct pargs * *pa*)

Definition at line 1145 of file kern_proc.c.

References pargs_free().

Referenced by kern_wait(), and sysctl_kern_proc_args().

Here is the call graph for this function:

9.50.2.19 void pargs_free (struct pargs * *pa*)

Definition at line 1129 of file kern_proc.c.

Referenced by pargs_drop(), and sysctl_kern_proc_args().

9.50.2.20 void pargs_hold (struct pargs * *pa*)

Definition at line 1136 of file kern_proc.c.

Referenced by fork1(), and sysctl_kern_proc_args().

9.50.2.21 struct proc* pfind (pid_t *pid*)

Definition at line 229 of file kern_proc.c.

References allproc_lock.

Referenced by filt_procattach(), fsetown(), getpgid(), getpriority(), getsid(), kern_ptrace(), kern_wait(), kill(), ktrace(), rtprio(), setpgid(), setpriority(), sigqueue(), sysctl_kern_proc(), sysctl_kern_proc_args(), sysctl_kern_proc_pathname(), sysctl_kern_proc_sv_name(), and sysctl_out_proc().

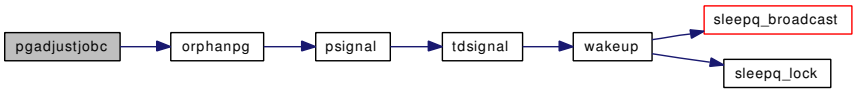
9.50.2.22 static void pgadjustjobc (struct pgrp * *pgrp*, int *entering*) [static]

Definition at line 470 of file kern_proc.c.

References orphanpg().

Referenced by fixjobc().

Here is the call graph for this function:



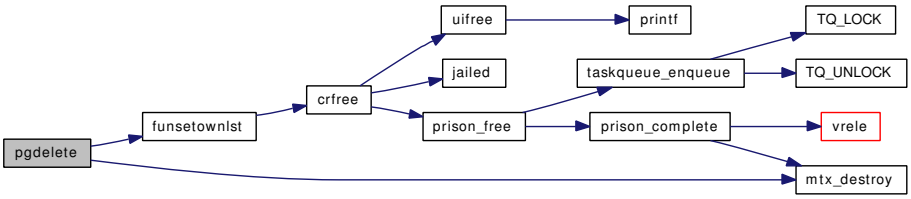
9.50.2.23 static void pgdelete (struct pgrp *) [static]

Definition at line 440 of file kern_proc.c.

References funsetownlst(), Giant, mtx_destroy(), and proctree_lock.

Referenced by doenterpgrp(), and leavepgrp().

Here is the call graph for this function:



9.50.2.24 struct pgrp* pgfind (pid_t *pgid*)

Definition at line 253 of file kern_proc.c.

References proctree_lock.

Referenced by enterpgrp(), fsetown(), getpriority(), gsignal(), killpg1(), ktrace(), setpgid(), setpriority(), setsid(), and ttioctl().

9.50.2.25 static int proc_ctor (void * *mem*, int *size*, void * *arg*, int *flags*) [static]

Definition at line 128 of file kern_proc.c.

Referenced by procinit().

9.50.2.26 static void proc_dtor (void * *mem*, int *size*, void * *arg*) [static]

Definition at line 140 of file kern_proc.c.

References td.

Referenced by procinit().

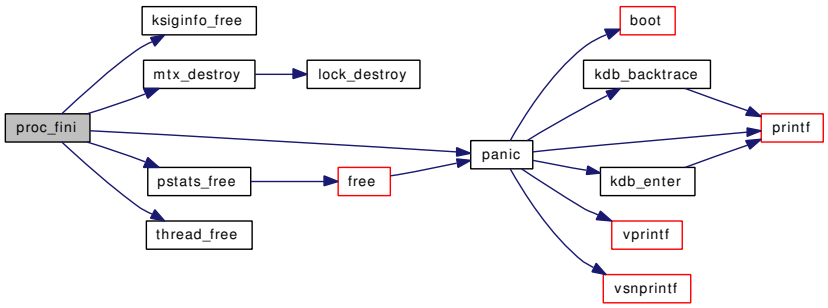
9.50.2.27 static void proc_fini (void * *mem*, int *size*) [static]

Definition at line 191 of file kern_proc.c.

References ksiginfo_free(), mtx_destroy(), panic(), pstats_free(), and thread_free().

Referenced by procinit().

Here is the call graph for this function:



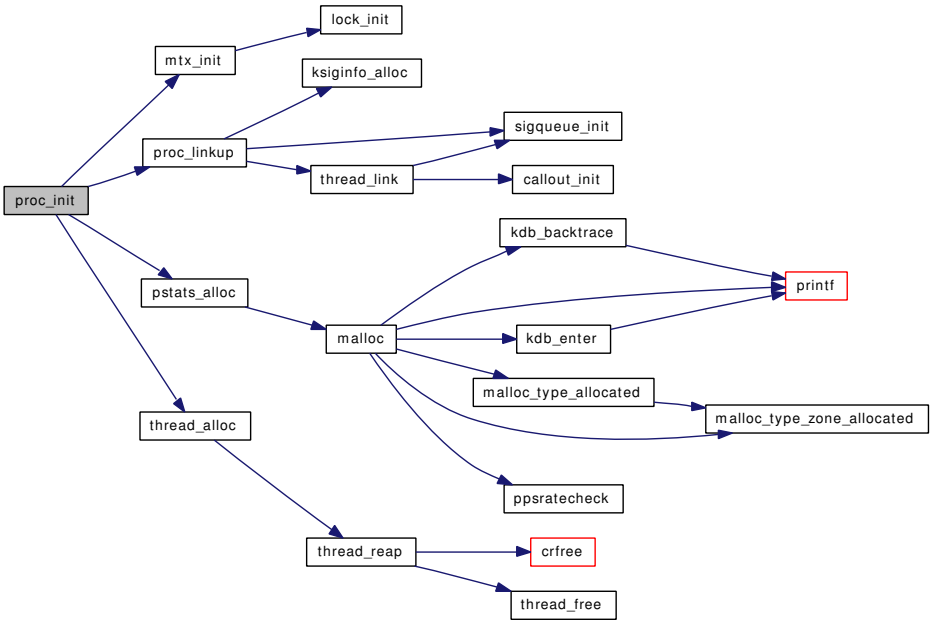
9.50.2.28 static int proc_init (void * *mem*, int *size*, int *flags*) [static]

Definition at line 170 of file kern_proc.c.

References mtx_init(), proc_linkup(), pstats_alloc(), td, and thread_alloc().

Referenced by procinit().

Here is the call graph for this function:



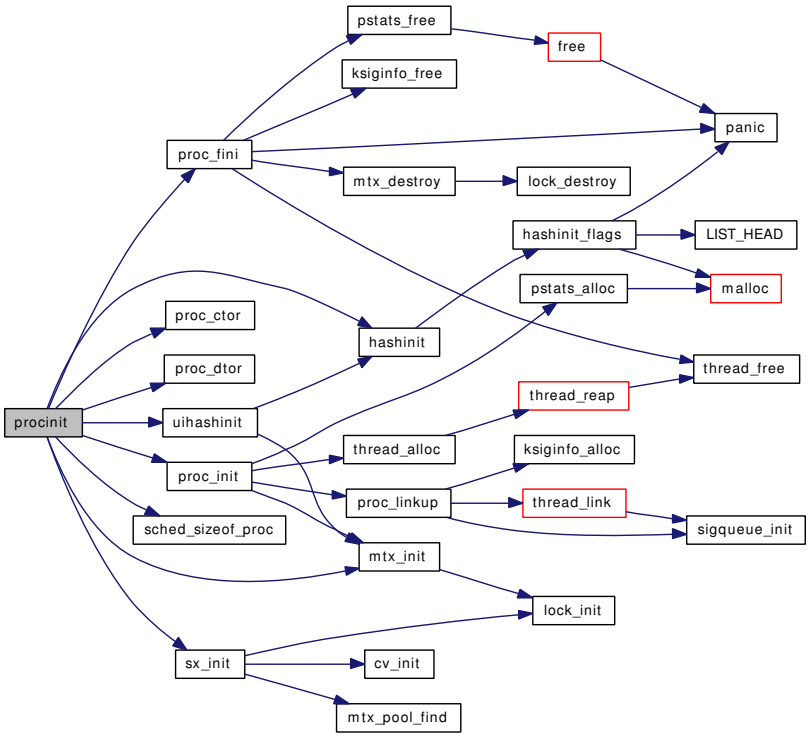
9.50.2.29 void procinit ()

Definition at line 108 of file kern_proc.c.

References allproc, allproc_lock, hashinit(), maxproc, mtx_init(), pgrphash, pgrphashtbl, pidhash, pidhashtbl, ppeers_lock, proc_ctor(), proc_dtor(), proc_fini(), proc_init(), proc_zone, proctree_lock, sched_sizeof_proc(), sx_init(), uihashinit(), and zombproc.

Referenced by proc0_init().

Here is the call graph for this function:



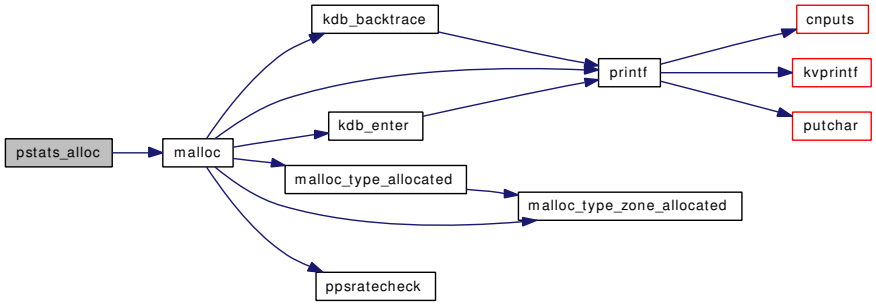
9.50.2.30 struct pstats* pstats_alloc (void)

Definition at line 841 of file kern_proc.c.

References malloc().

Referenced by proc0_init(), and proc_init().

Here is the call graph for this function:



9.50.2.31 void pstats_fork (struct pstats * *src*, struct pstats * *dst*)

Definition at line 851 of file kern_proc.c.

Referenced by fork1().

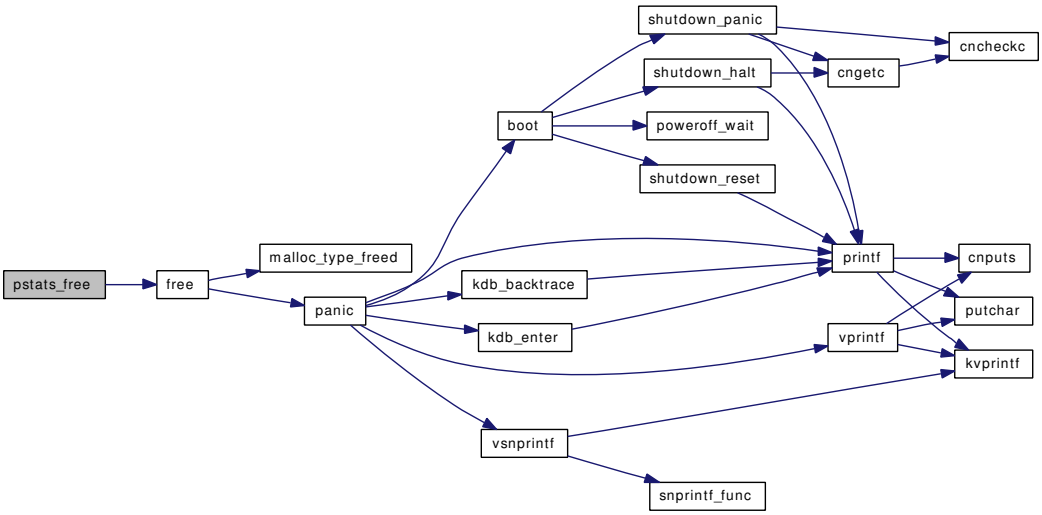
9.50.2.32 void pstats_free (struct pstats * *ps*)

Definition at line 861 of file kern_proc.c.

References free().

Referenced by proc_fini().

Here is the call graph for this function:

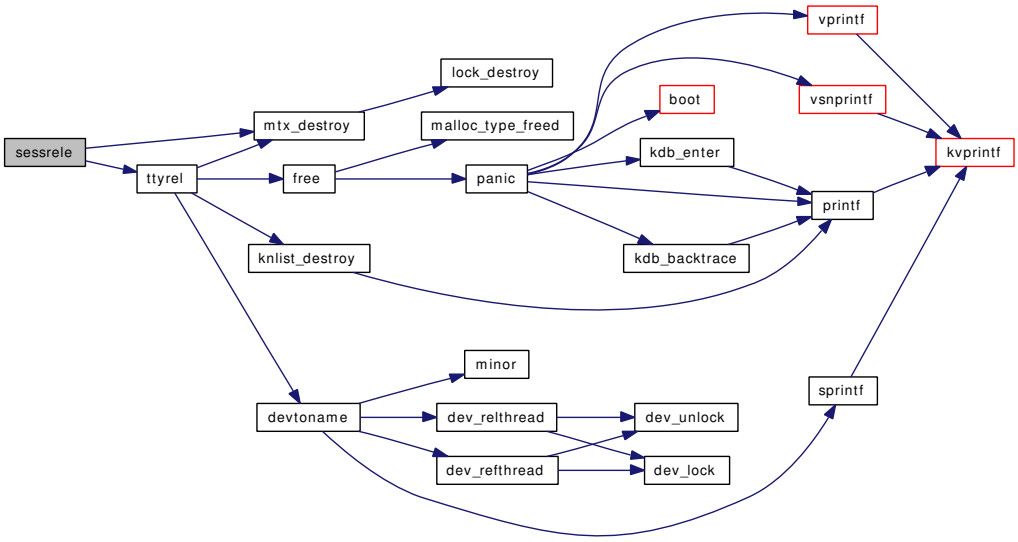


9.50.2.33 void sessrele (struct session * *s*)

Definition at line 569 of file kern_proc.c.

References mtx_destroy(), and ttyrel().

Here is the call graph for this function:



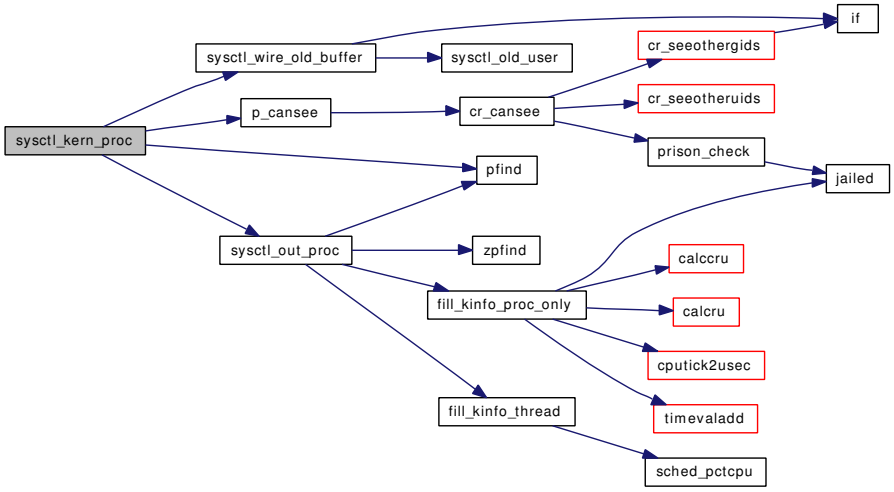
9.50.2.34 SYSCTL_INT (_kern, OID_AUTO, kstack_pages, CTLFLAG_RD, & *kstack_pages*, 0, "")

9.50.2.35 static int sysctl_kern_proc (SYSCTL_HANDLER_ARGS) [static]

Definition at line 946 of file kern_proc.c.

References allproc, allproc_lock, KERN_PROC_NOTHREADS, p_cansee(), pfind(), sched_lock, sysctl_-out_proc(), sysctl_wire_old_buffer(), and zombproc.

Here is the call graph for this function:

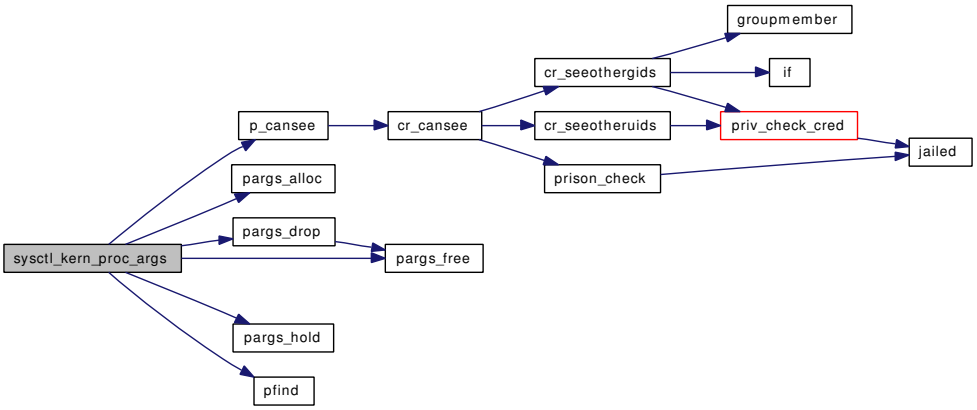


9.50.2.36 static int sysctl_kern_proc_args (SYSCTL_HANDLER_ARGS) [static]

Definition at line 1161 of file kern_proc.c.

References p_cansee(), pargs_alloc(), pargs_drop(), pargs_free(), pargs_hold(), pfind(), and ps_arg_cache_limit.

Here is the call graph for this function:

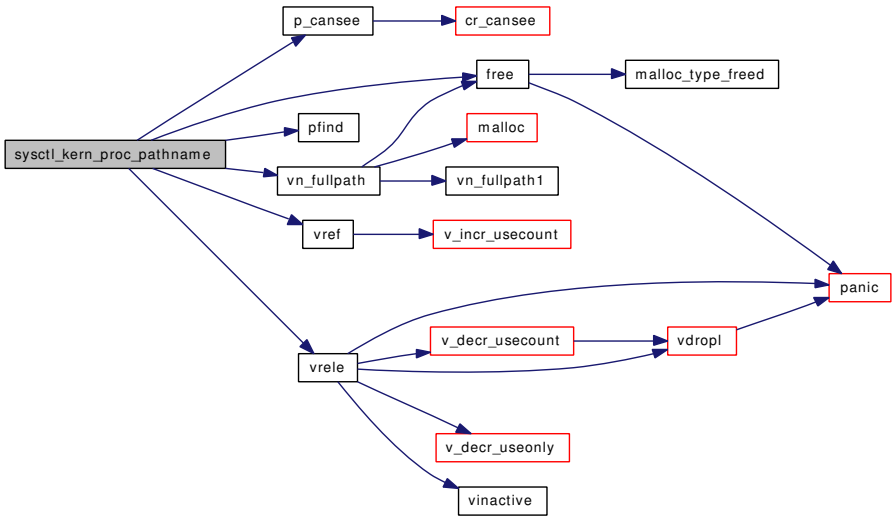


9.50.2.37 static int sysctl_kern_proc_pathname (SYSCTL_HANDLER_ARGS) [static]

Definition at line 1216 of file kern_proc.c.

References free(), p_cansee(), pfind(), vn_fullpath(), vref(), and vrele().

Here is the call graph for this function:

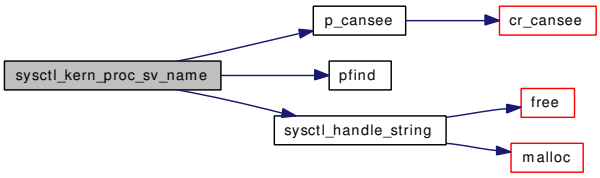


9.50.2.38 static int sysctl_kern_proc_sv_name (SYSCTL_HANDLER_ARGS) [static]

Definition at line 1253 of file kern_proc.c.

References p_cansee(), pfind(), and sysctl_handle_string().

Here is the call graph for this function:



9.50.2.39 static SYSCTL_NODE (_kern_proc, (KERN_PROC_PROC|KERN_PROC_INC_-THREAD), proc_td, CTLFLAG_RD, sysctl_kern_proc, "Return process *table*, no threads") [static]

9.50.2.40 static SYSCTL_NODE (_kern_proc, (KERN_PROC_PID|KERN_PROC_-INC_THREAD), pid_td, CTLFLAG_RD, sysctl_kern_proc, "Process table") [static]

9.50.2.41 static SYSCTL_NODE (_kern_proc, (KERN_PROC_RUID|KERN_PROC_-INC_THREAD), ruid_td, CTLFLAG_RD, sysctl_kern_proc, "Process table") [static]

9.50.2.42 static SYSCTL_NODE (_kern_proc, (KERN_PROC_UID|KERN_PROC_-INC_THREAD), uid_td, CTLFLAG_RD, sysctl_kern_proc, "Process table") [static]

9.50.2.43 static SYSCTL_NODE (_kern_proc, (KERN_PROC_TTY|KERN_PROC_-INC_THREAD), tty_td, CTLFLAG_RD, sysctl_kern_proc, "Process table") [static]

9.50.2.44 static SYSCTL_NODE (_kern_proc, (KERN_PROC_SESSION|KERN_PROC_-INC_THREAD), sid_td, CTLFLAG_RD, sysctl_kern_proc, "Process table") [static]

9.50.2.45 static SYSCTL_NODE (_kern_proc, (KERN_PROC_RGID|KERN_PROC_-INC_THREAD), rgid_td, CTLFLAG_RD, sysctl_kern_proc, "Process table") [static]

9.50.2.46 static SYSCTL_NODE (_kern_proc, (KERN_PROC_PGRP|KERN_PROC_-INC_THREAD), pgrp_td, CTLFLAG_RD, sysctl_kern_proc, "Process table") [static]

9.50.2.47 static SYSCTL_NODE (_kern_proc, (KERN_PROC_GID|KERN_PROC_-INC_THREAD), gid_td, CTLFLAG_RD, sysctl_kern_proc, "Process table") [static]

9.50.2.48 static SYSCTL_NODE (_kern_proc, KERN_PROC_SV_NAME, sv_name, CTLFLAG_RD, sysctl_kern_proc_sv_name, "Process syscall vector name (ABI type)") [static]

9.50.2.49 static SYSCTL_NODE (_kern_proc, KERN_PROC_PATHNAME, pathname, CTLFLAG_RD, sysctl_kern_proc_pathname, "Process executable path") [static]

9.50.2.50 static SYSCTL_NODE (_kern_proc, KERN_PROC_ARGS, args, CTLFLAG_RW| *CTLFLAG_ANYBODY*, sysctl_kern_proc_args, "Process argument list") [static]

9.50.2.51 static SYSCTL_NODE (_kern_proc, KERN_PROC_PROC, proc, CTLFLAG_RD, sysctl_kern_proc, "Return process *table*, no threads") [static]

9.50.2.52 static SYSCTL_NODE (_kern_proc, KERN_PROC_PID, pid, CTLFLAG_RD, sysctl_kern_proc, "Process table") [static]

9.50.2.53 static SYSCTL_NODE (_kern_proc, KERN_PROC_RUID, ruid, CTLFLAG_RD, sysctl_kern_proc, "Process table") [static]

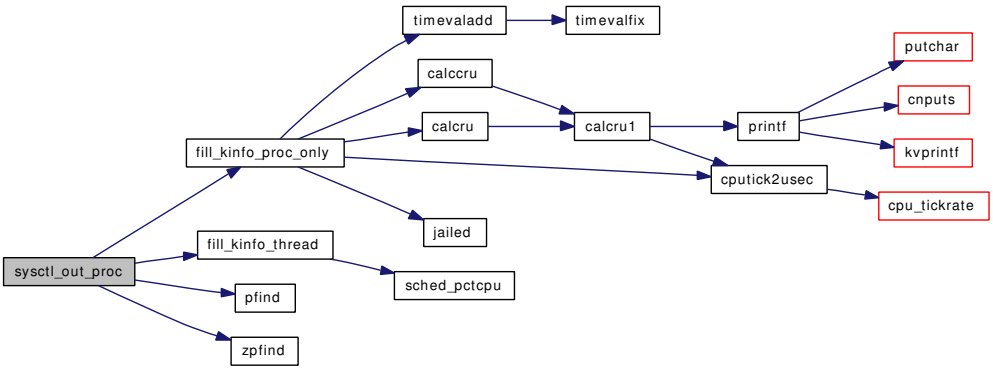
9.50.2.54 static SYSCTL_NODE (_kern_proc, KERN_PROC_UID, uid, CTLFLAG_RD, sysctl_kern_proc, "Process table") [static]

9.50.2.55 static SYSCTL_NODE (_kern_proc, KERN_PROC_TTY, tty, CTLFLAG_RD, sysctl_kern_proc, "Process table") [static]

References fill_kinfo_proc_only(), fill_kinfo_thread(), KERN_PROC_NOTHREADS, KERN_PROC_-ZOMBMASK, pfind(), sched_lock, td, and zpfind().

Referenced by sysctl_kern_proc().

Here is the call graph for this function:



9.50.2.62 SYSCTL_PROC (_kern_proc, KERN_PROC_ALL, all, CTLFLAG_RD| *CTLTYPE_STRUCT*, 0, 0, sysctl_kern_proc, " *S*, proc", "Return entire process table")

9.50.2.63 struct proc* zpfind (pid_t *pid*)

Definition at line 871 of file kern_proc.c.

References allproc_lock, and zombproc.

Referenced by filt_procattach(), kill(), sigqueue(), and sysctl_out_proc().

9.50.3 Variable Documentation

9.50.3.1 struct proclist allproc

Definition at line 91 of file kern_proc.c.

Referenced by fork1(), kdb_thr_first(), kdb_thr_from_pid(), proc0_init(), procinit(), and sysctl_kern_-proc().

9.50.3.2 struct sx allproc_lock

Definition at line 93 of file kern_proc.c.

Referenced by brand_inuse(), exit1(), fork1(), getpriority(), kern_ptrace(), kern_wait(), killpg1(), ktrace(), mountcheckdirs(), pfind(), proc0_post(), procinit(), schedcpu(), setpriority(), sysctl_kern_file(), sysctl_-kern_proc(), sysctl_kern_randompid(), and zpfind().

9.50.3.3 uma_zone_t ithread_zone

Definition at line 97 of file kern_proc.c.

9.50.3.4 int kstack_pages = KSTACK_PAGES

Definition at line 99 of file kern_proc.c.

9.50.3.5 u_long pgrphash

Definition at line 90 of file kern_proc.c.

Referenced by procinit().

9.50.3.6 struct pgrphashhead* pgrphashtbl

Definition at line 89 of file kern_proc.c.

Referenced by procinit().

9.50.3.7 u_long pidhash

Definition at line 88 of file kern_proc.c.

Referenced by procinit().

9.50.3.8 struct pidhashhead* pidhashtbl

Definition at line 87 of file kern_proc.c.

Referenced by procinit().

9.50.3.9 struct mtx ppeers_lock

Definition at line 95 of file kern_proc.c.

Referenced by exit1(), fork1(), and procinit().

9.50.3.10 uma_zone_t proc_zone

Definition at line 96 of file kern_proc.c.

Referenced by fork1(), kern_wait(), and procinit().

9.50.3.11 struct sx proctree_lock

Definition at line 94 of file kern_proc.c.

Referenced by doenterpgrp(), enterpgrp(), enterthispgrp(), exit1(), fixjobc(), fork1(), fsetown(), getpriority(), gsignal(), inferior(), kern_ptrace(), kern_wait(), killpg1(), kthread_exit(), ktrace(), leavepgrp(), pgdelete(), pgfind(), proc_reparent(), procinit(), setpgid(), setpriority(), setsid(), ttioctl(), ttread(), ttwrite(), and ttymodem().

9.50.3.12 struct proclist zombproc

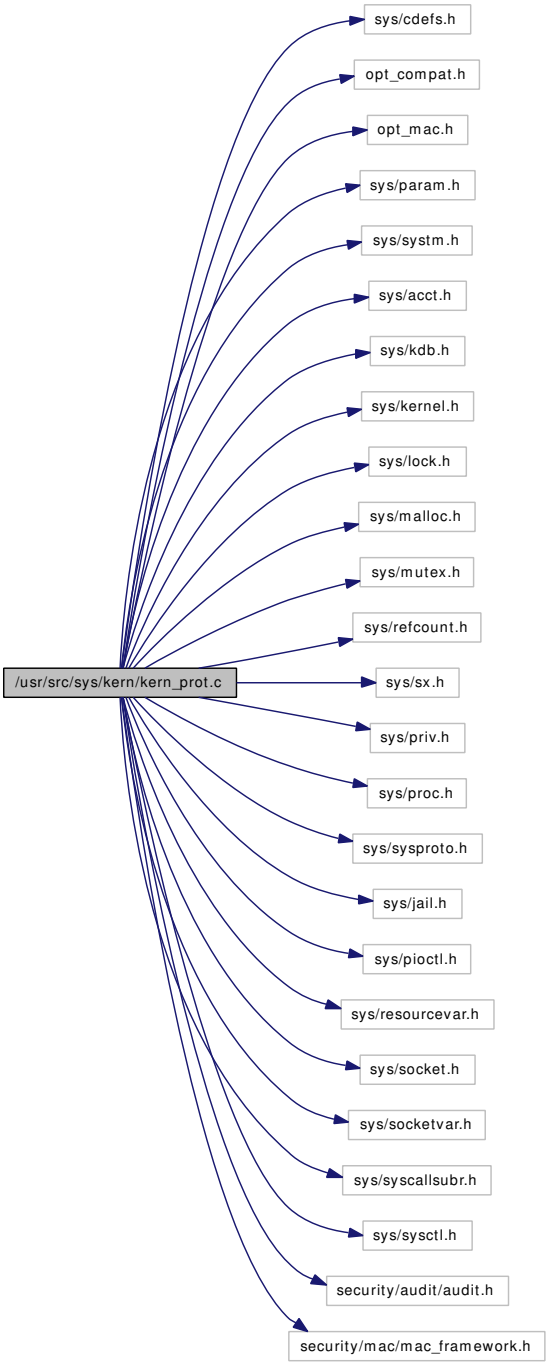
Definition at line 92 of file kern_proc.c.

Referenced by exit1(), fork1(), procinit(), sysctl_kern_proc(), and zpfind().

9.51 /usr/src/sys/kern/kern_prot.c File Reference

```
#include <sys/cdefs.h>
#include "opt_compat.h"
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/acct.h>
#include <sys/kdb.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/refcount.h>
#include <sys/sx.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/sysproto.h>
#include <sys/jail.h>
#include <sys/pioctl.h>
#include <sys/resourcevar.h>
#include <sys/socket.h>
#include <sys/socketvar.h>
#include <sys/syscallsubr.h>
#include <sys/sysctl.h>
#include <security/audit/audit.h>
#include <security/mac/mac_framework.h>
```

Include dependency graph for kern_prot.c:



Data Structures

- struct getpid_args
- struct getppid_args
- struct getpgrp_args
- struct getpgid_args
- struct getsid_args
- struct getuid_args

- struct geteuid_args
- struct getgid_args
- struct getegid_args
- struct getgroups_args
- struct setsid_args
- struct setpgid_args
- struct setuid_args
- struct seteuid_args
- struct setgid_args
- struct setegid_args
- struct setgroups_args
- struct setreuid_args
- struct setregid_args
- struct setresuid_args
- struct setresgid_args
- struct getresuid_args
- struct getresgid_args
- struct issetugid_args
- struct getlogin_args
- struct setlogin_args

Defines

- #define POSIX_APPENDIX_B_4_2_2

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_prot.c,v 1.207 2007/01/08 20:37:02 rwatson Exp $")
- static MALLOC_DEFINE (M_CRED,"cred","credentials")
- SYSCTL_NODE (_security, OID_AUTO, bsd, CTLFLAG_RW, 0,"BSD security policy")
- int getpid (struct thread *td, struct getpid_args *uap)
- int getppid (struct thread *td, struct getppid_args *uap)
- int getpgrp (struct thread *td, struct getpgrp_args *uap)
- int getpgid (struct thread *td, struct getpgid_args *uap)
- int getsid (struct thread *td, struct getsid_args *uap)
- int getuid (struct thread *td, struct getuid_args *uap)
- int geteuid (struct thread *td, struct geteuid_args *uap)
- int getgid (struct thread *td, struct getgid_args *uap)
- int getegid (struct thread *td, struct getegid_args *uap)
- int getgroups (struct thread *td, register struct getgroups_args *uap)
- int kern_getgroups (struct thread *td, u_int *ngrp, gid_t *groups)
- int setsid (register struct thread *td, struct setsid_args *uap)
- int setpgid (struct thread *td, register struct setpgid_args *uap)
- int setuid (struct thread *td, struct setuid_args *uap)
- int seteuid (struct thread *td, struct seteuid_args *uap)
- int setgid (struct thread *td, struct setgid_args *uap)
- int setegid (struct thread *td, struct setegid_args *uap)
- int setgroups (struct thread *td, struct setgroups_args *uap)
- int kern_setgroups (struct thread *td, u_int ngrp, gid_t *groups)

- int setreuid (register struct thread *td, struct setreuid_args *uap)
- int setregid (register struct thread *td, struct setregid_args *uap)
- int setresuid (register struct thread *td, struct setresuid_args *uap)
- int setresgid (register struct thread *td, struct setresgid_args *uap)
- int getresuid (register struct thread *td, struct getresuid_args *uap)
- int getresgid (register struct thread *td, struct getresgid_args *uap)
- int issetugid (register struct thread *td, struct issetugid_args *uap)
- int __setugid (struct thread *td, struct __setugid_args *uap)
- int groupmember (gid_t gid, struct ucred *cred)
- int securelevel_gt (struct ucred *cr, int level)
- int securelevel_ge (struct ucred *cr, int level)
- SYSCTL_INT (_security_bsd, OID_AUTO, see_other_uids, CTLFLAG_RW,&see_other_uids, 0,"Unprivileged processes may see subjects/objects with different real uid")
- static int cr_seeotheruids (struct ucred *u1, struct ucred *u2)
- SYSCTL_INT (_security_bsd, OID_AUTO, see_other_gids, CTLFLAG_RW,&see_other_gids, 0,"Unprivileged processes may see subjects/objects with different real gid")
- static int cr_seeothergids (struct ucred *u1, struct ucred *u2)
- int cr_cansee (struct ucred *u1, struct ucred *u2)
- int p_cansee (struct thread *td, struct proc *p)
- SYSCTL_INT (_security_bsd, OID_AUTO, conservative_signals, CTLFLAG_RW,&conservative_signals, 0,"Unprivileged processes prevented from ""sending certain signals to processes whose credentials have changed")
- int cr_cansignal (struct ucred *cred, struct proc *proc, int signum)
- int p_cansignal (struct thread *td, struct proc *p, int signum)
- int p_cansched (struct thread *td, struct proc *p)
- SYSCTL_INT (_security_bsd, OID_AUTO, unprivileged_proc_debug, CTLFLAG_RW,&unprivileged_proc_debug, 0,"Unprivileged processes may use process debugging facilities")
- int p_candebug (struct thread *td, struct proc *p)
- int cr_canseesocket (struct ucred *cred, struct socket *so)
- int p_canwait (struct thread *td, struct proc *p)
- ucred * crget (void)
- ucred * crhold (struct ucred *cr)
- void crfree (struct ucred *cr)
- int crshared (struct ucred *cr)
- void crcopy (struct ucred *dest, struct ucred *src)
- ucred * crdup (struct ucred *cr)
- void cru2x (struct ucred *cr, struct xucred *xcr)
- void cred_update_thread (struct thread *td)
- int getlogin (struct thread *td, struct getlogin_args *uap)
- int setlogin (struct thread *td, struct setlogin_args *uap)
- void setsugid (struct proc *p)
- void change_euid (struct ucred *newcred, struct uidinfo *euip)
- void change_egid (struct ucred *newcred, gid_t egid)
- void change_ruid (struct ucred *newcred, struct uidinfo *ruip)
- void change_rgid (struct ucred *newcred, gid_t rgid)
- void change_svuid (struct ucred *newcred, uid_t svuid)
- void change_svgid (struct ucred *newcred, gid_t svgid)

Variables

- static int see_other_uids = 1
- static int see_other_gids = 1
- static int conservative_signals = 1
- static int unprivileged_proc_debug = 1

9.51.1 Define Documentation

9.51.1.1 #define POSIX_APPENDIX_B_4_2_2

Definition at line 496 of file kern_prot.c.

Referenced by setgid(), and setuid().

9.51.2 Function Documentation

9.51.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_prot. *c*, v 1.207 2007/01/08 20:37:02 rwatson Exp $")

9.51.2.2 int __setugid (struct thread * *td*, struct __setugid_args * *uap*)

Definition at line 1296 of file kern_prot.c.

9.51.2.3 void change_egid (struct ucred * *newcred*, gid_t *egid*)

Definition at line 2081 of file kern_prot.c.

Referenced by setegid(), setgid(), setregid(), and setresgid().

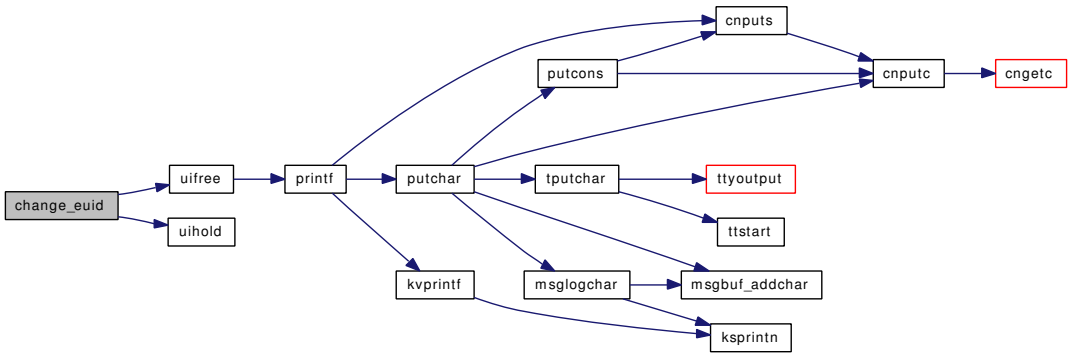
9.51.2.4 void change_euid (struct ucred * *newcred*, struct uidinfo * *euip*)

Definition at line 2065 of file kern_prot.c.

References uifree(), and uihold().

Referenced by seteuid(), setresuid(), setreuid(), and setuid().

Here is the call graph for this function:



9.51.2.5 void change_rgid (struct ucred * *newcred*, gid_t *rgid*)

Definition at line 2114 of file kern_prot.c.

Referenced by setgid(), setregid(), and setresgid().

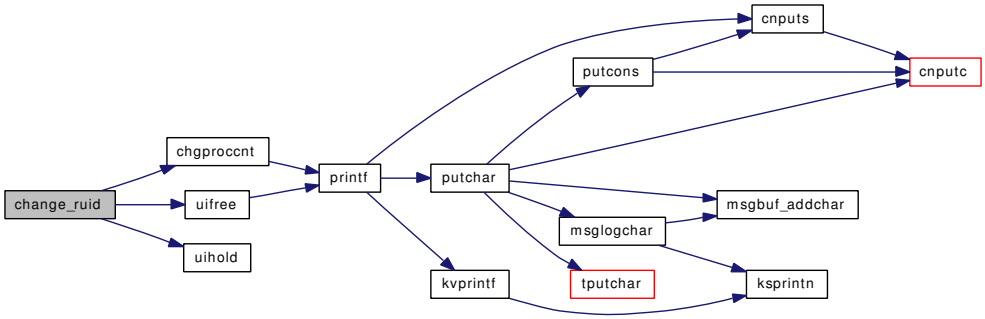
9.51.2.6 void change_ruid (struct ucred * *newcred*, struct uidinfo * *ruip*)

Definition at line 2096 of file kern_prot.c.

References chgproccnt(), uifree(), and uihold().

Referenced by setresuid(), setreuid(), and setuid().

Here is the call graph for this function:



9.51.2.7 void change_svgid (struct ucred * *newcred*, gid_t *svgid*)

Definition at line 2140 of file kern_prot.c.

Referenced by setgid(), setregid(), and setresgid().

9.51.2.8 void change_svuid (struct ucred * *newcred*, uid_t *svuid*)

Definition at line 2127 of file kern_prot.c.

Referenced by setresuid(), setreuid(), and setuid().

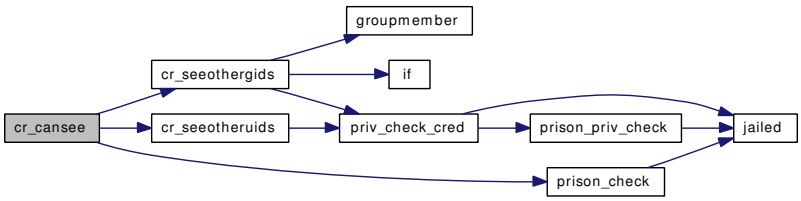
9.51.2.9 int cr_cansee (struct ucred * *u1*, struct ucred * *u2*)

Definition at line 1457 of file kern_prot.c.

References cr_seeothergids(), cr_seeotheruids(), and prison_check().

Referenced by p_cansee(), and unp_pcblist().

Here is the call graph for this function:

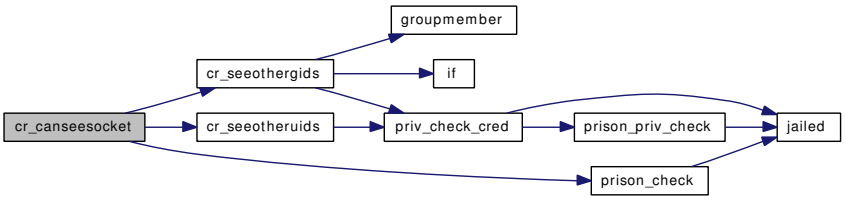


9.51.2.10 int cr_canseesocket (struct ucred * *cred*, struct socket * *so*)

Definition at line 1784 of file kern_prot.c.

References cr_seeothergids(), cr_seeotheruids(), and prison_check().

Here is the call graph for this function:



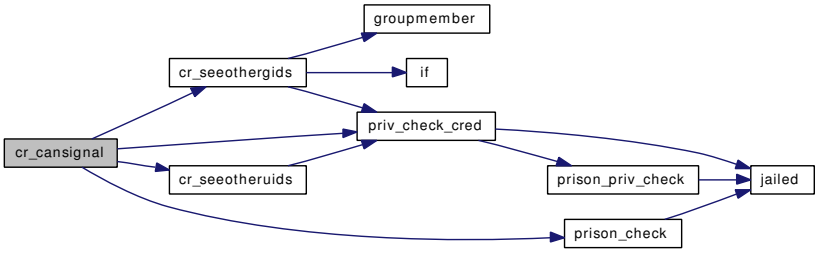
9.51.2.11 int cr_cansignal (struct ucred * *cred*, struct proc * *proc*, int *signum*)

Definition at line 1512 of file kern_prot.c.

References conservative_signals, cr_seeothergids(), cr_seeotheruids(), prison_check(), and priv_check_-cred().

Referenced by p_cansignal().

Here is the call graph for this function:



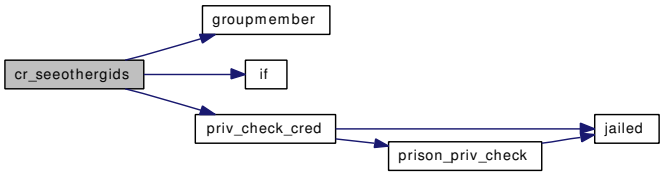
9.51.2.12 static int cr_seeothergids (struct ucred * *u1*, struct ucred * *u2*) [static]

Definition at line 1428 of file kern_prot.c.

References groupmember(), if(), priv_check_cred(), and see_other_gids.

Referenced by cr_cansee(), cr_canseesocket(), cr_cansignal(), p_candebug(), and p_cansched().

Here is the call graph for this function:



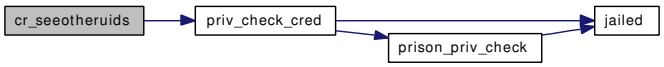
9.51.2.13 static int cr_seeotheruids (struct ucred * *u1*, struct ucred * *u2*) [static]

Definition at line 1397 of file kern_prot.c.

References priv_check_cred(), and see_other_uids.

Referenced by cr_cansee(), cr_canseesocket(), cr_cansignal(), p_candebug(), p_cansched(), and p_-canwait().

Here is the call graph for this function:



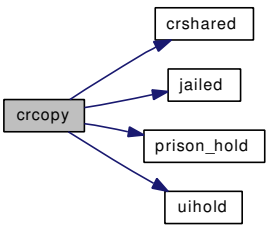
9.51.2.14 void crcopy (struct ucred * *dest*, struct ucred * *src*)

Definition at line 1915 of file kern_prot.c.

References crshared(), jailed(), prison_hold(), and uihold().

Referenced by crdup(), create_init(), jail_attach(), kern_setgroups(), setegid(), seteuid(), setgid(), se-tregid(), setresgid(), setresuid(), setreuid(), and setuid().

Here is the call graph for this function:



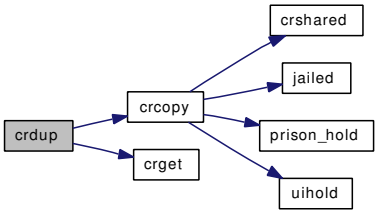
9.51.2.15 struct ucred* crdup (struct ucred * *cr*)

Definition at line 1936 of file kern_prot.c.

References crcopy(), and crget().

Referenced by kern_access(), and vfs_mount_alloc().

Here is the call graph for this function:



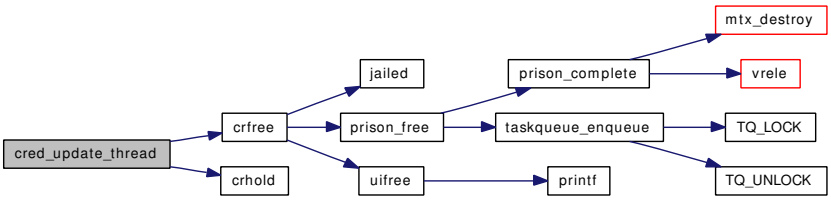
9.51.2.16 void cred_update_thread (struct thread * *td*)

Definition at line 1966 of file kern_prot.c.

References crfree(), and crhold().

Referenced by ast(), and create_init().

Here is the call graph for this function:



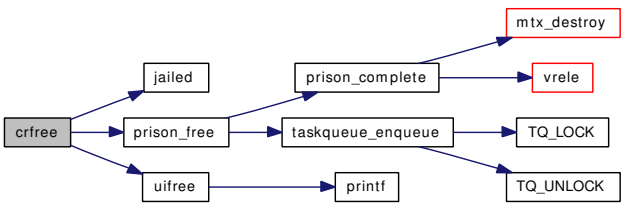
9.51.2.17 void crfree (struct ucred * *cr*)

Definition at line 1872 of file kern_prot.c.

References jailed(), prison_free(), and uifree().

Referenced by acct(), acct_disable(), aio_free_entry(), alq_shutdown(), create_init(), create_thread(), cred_update_thread(), exit1(), fdrop_locked(), fsetown(), funsetown(), funsetownlst(), getnewbuf(), jail_attach(), kern_access(), kern_setgroups(), kern_wait(), ktrace(), setegid(), seteuid(), setgid(), setregid(), setresgid(), setresuid(), setreuid(), setuid(), sodealloc(), thread_reap(), and thread_wait().

Here is the call graph for this function:



9.51.2.18 struct ucred* crget (void)

Definition at line 1842 of file kern_prot.c.

Referenced by crdup(), create_init(), jail_attach(), kern_setgroups(), proc0_init(), setegid(), seteuid(), setgid(), setregid(), setresgid(), setresuid(), setreuid(), and setuid().

9.51.2.19 struct ucred* crhold (struct ucred * *cr*)

Definition at line 1859 of file kern_prot.c.

Referenced by acct(), aio_aqueue(), alq_open(), breada(), breadn(), cluster_wbuild(), create_thread(), cred_update_thread(), falloc(), fork1(), fsetown(), make_dev_credv(), proc0_init(), socreate(), and sonewconn().

9.51.2.20 int crshared (struct ucred * *cr*)

Definition at line 1904 of file kern_prot.c.

Referenced by crcopy().

9.51.2.21 void cru2x (struct ucred * *cr*, struct xucred * *xcr*)

Definition at line 1950 of file kern_prot.c.

Referenced by unp_connect(), and unp_listen().

9.51.2.22 int getegid (struct thread * *td*, struct getegid_args * *uap*)

Definition at line 283 of file kern_prot.c.

9.51.2.23 int geteuid (struct thread * *td*, struct geteuid_args * *uap*)

Definition at line 241 of file kern_prot.c.

9.51.2.24 int getgid (struct thread * *td*, struct getgid_args * *uap*)

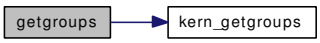
Definition at line 258 of file kern_prot.c.

9.51.2.25 int getgroups (struct thread * *td*, register struct getgroups_args * *uap*)

Definition at line 300 of file kern_prot.c.

References kern_getgroups().

Here is the call graph for this function:



9.51.2.26 int getlogin (struct thread * *td*, struct getlogin_args * *uap*)

Definition at line 1994 of file kern_prot.c.

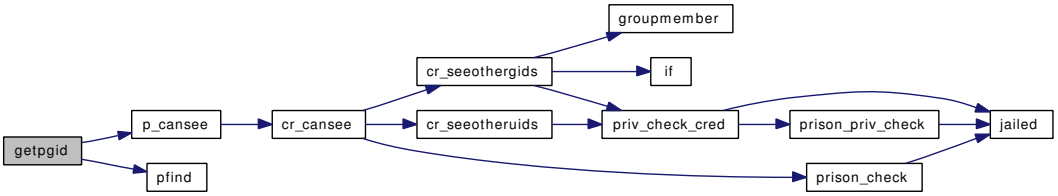
References getlogin_args::namebuf, and getlogin_args::namelen.

9.51.2.27 int getpgid (struct thread * *td*, struct getpgid_args * *uap*)

Definition at line 153 of file kern_prot.c.

References p_cansee(), pfind(), and getpgid_args::pid.

Here is the call graph for this function:



9.51.2.28 int getpgrp (struct thread * *td*, struct getpgrp_args * *uap*)

Definition at line 133 of file kern_prot.c.

9.51.2.29 int getpid (struct thread * *td*, struct getpid_args * *uap*)

Definition at line 88 of file kern_prot.c.

9.51.2.30 int getppid (struct thread * *td*, struct getppid_args * *uap*)

Definition at line 111 of file kern_prot.c.

9.51.2.31 int getresgid (register struct thread * *td*, struct getresgid_args * *uap*)

Definition at line 1246 of file kern_prot.c.

References getresgid_args::egid, getresgid_args::rgid, and getresgid_args::sgid.

9.51.2.32 int getresuid (register struct thread * *td*, struct getresuid_args * *uap*)

Definition at line 1216 of file kern_prot.c.

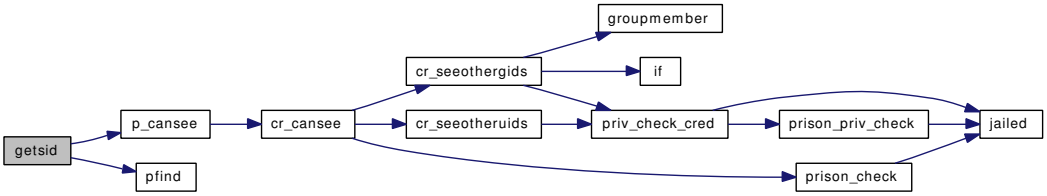
References getresuid_args::euid, getresuid_args::ruid, and getresuid_args::suid.

9.51.2.33 int getsid (struct thread * *td*, struct getsid_args * *uap*)

Definition at line 188 of file kern_prot.c.

References p_cansee(), pfind(), and getsid_args::pid.

Here is the call graph for this function:



9.51.2.34 int getuid (struct thread * *td*, struct getuid_args * *uap*)

Definition at line 221 of file kern_prot.c.

9.51.2.35 int groupmember (gid_t *gid*, struct ucred * *cred*)

Definition at line 1328 of file kern_prot.c.

Referenced by can_hardlink(), cr_seeothergids(), ipcperm(), mqfs_setattr(), p_candebug(), and vaccess().

9.51.2.36 int issetugid (register struct thread * *td*, struct issetugid_args * *uap*)

Definition at line 1274 of file kern_prot.c.

9.51.2.37 int kern_getgroups (struct thread * *td*, u_int * *ngrp*, gid_t * *groups*)

Definition at line 318 of file kern_prot.c.

Referenced by getgroups().

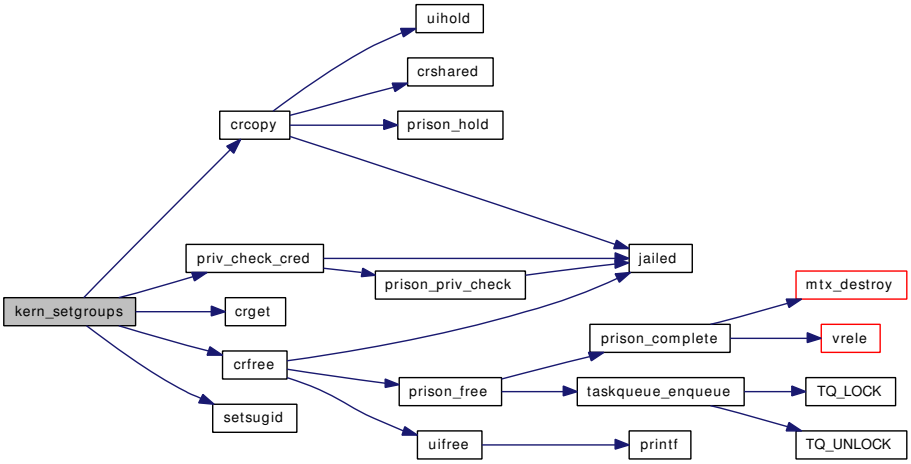
9.51.2.38 int kern_setgroups (struct thread * *td*, u_int *ngrp*, gid_t * *groups*)

Definition at line 852 of file kern_prot.c.

References crcopy(), crfree(), crget(), priv_check_cred(), and setsugid().

Referenced by setgroups().

Here is the call graph for this function:



9.51.2.39 static MALLOC_DEFINE (M_CRED, "cred", "credentials") [static]

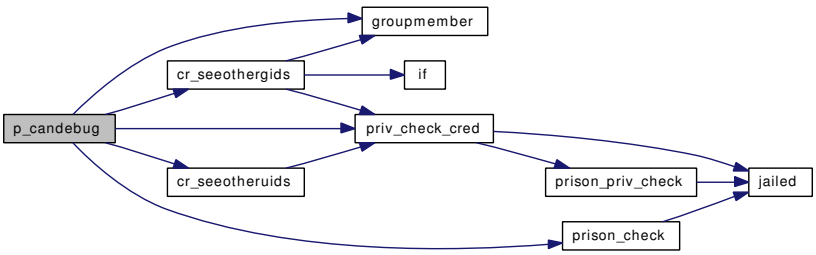
9.51.2.40 int p_candebug (struct thread * *td*, struct proc * *p*)

Definition at line 1687 of file kern_prot.c.

References cr_seeothergids(), cr_seeotheruids(), groupmember(), prison_check(), priv_check_cred(), and unprivileged_proc_debug.

Referenced by kern_ptrace().

Here is the call graph for this function:



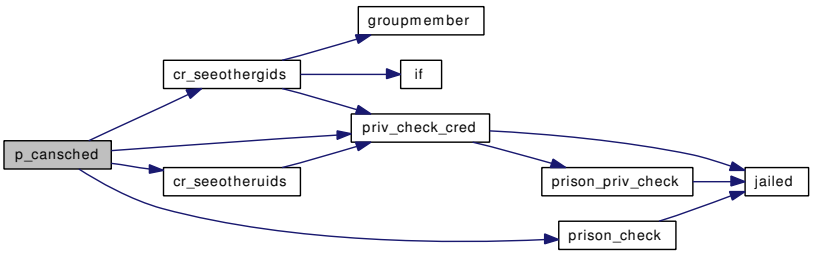
9.51.2.41 int p_cansched (struct thread * *td*, struct proc * *p*)

Definition at line 1634 of file kern_prot.c.

References cr_seeothergids(), cr_seeotheruids(), prison_check(), and priv_check_cred().

Referenced by donice(), rtprio(), and rtprio_thread().

Here is the call graph for this function:



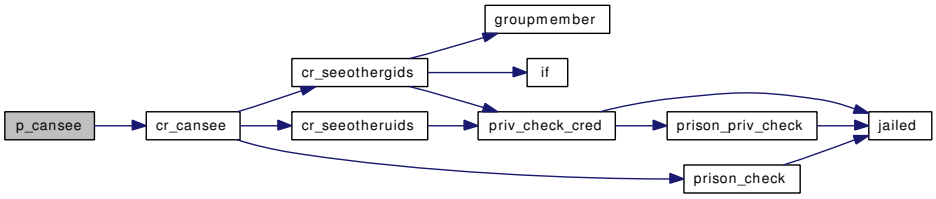
9.51.2.42 int p_cansee (struct thread * *td*, struct proc * *p*)

Definition at line 1482 of file kern_prot.c.

References cr_cansee().

Referenced by filt_procattach(), getpgid(), getpriority(), getsid(), kern_ptrace(), ktrace(), rtprio(), rtprio_thread(), setpgid(), setpriority(), sysctl_kern_file(), sysctl_kern_proc(), sysctl_kern_proc_args(), sysctl_kern_proc_pathname(), and sysctl_kern_proc_sv_name().

Here is the call graph for this function:



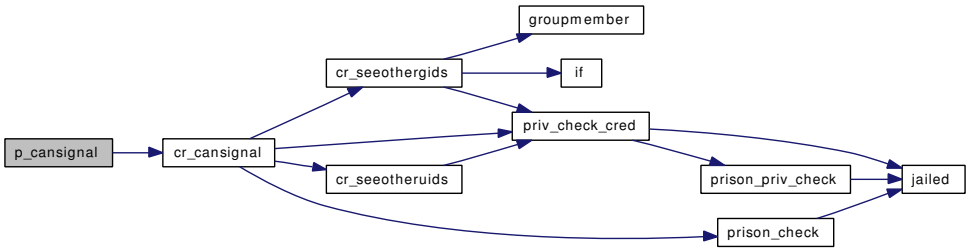
9.51.2.43 int p_cansignal (struct thread * *td*, struct proc * *p*, int *signum*)

Definition at line 1593 of file kern_prot.c.

References cr_cansignal().

Referenced by kill(), killpg1(), and sigqueue().

Here is the call graph for this function:



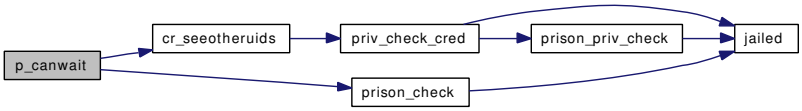
9.51.2.44 int p_canwait (struct thread * *td*, struct proc * *p*)

Definition at line 1816 of file kern_prot.c.

References cr_seeotheruids(), and prison_check().

Referenced by kern_wait().

Here is the call graph for this function:



9.51.2.45 int securelevel_ge (struct ucred * *cr*, int *level*)

Definition at line 1365 of file kern_prot.c.

References securelevel.

9.51.2.46 int securelevel_gt (struct ucred * *cr*, int *level*)

Definition at line 1352 of file kern_prot.c.

References securelevel.

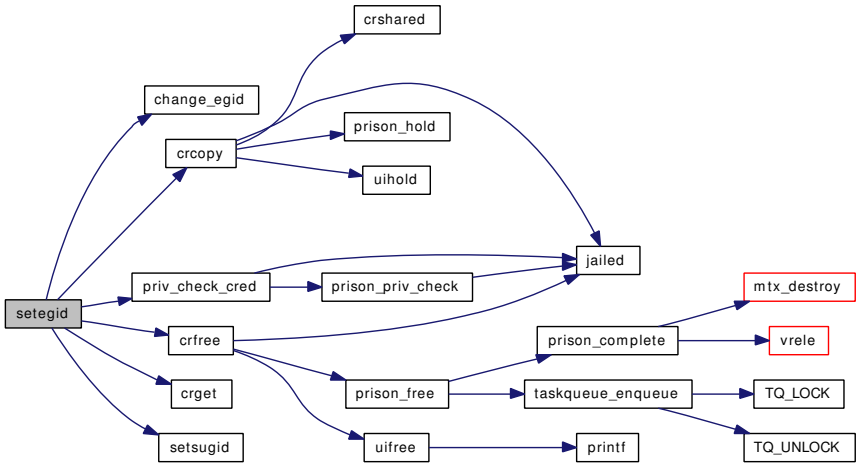
Referenced by firmware_get(), kern_kldload(), kern_kldunload(), settime(), and sysctl_root().

9.51.2.47 int setegid (struct thread * *td*, struct setegid_args * *uap*)

Definition at line 786 of file kern_prot.c.

References change_egid(), crcopy(), crfree(), crget(), setegid_args::egid, priv_check_cred(), and setsugid().

Here is the call graph for this function:

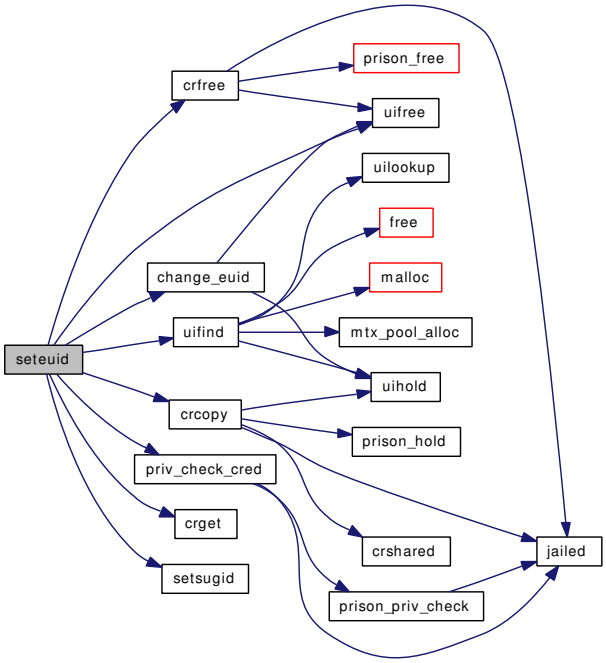


9.51.2.48 int seteuid (struct thread * *td*, struct seteuid_args * *uap*)

Definition at line 624 of file kern_prot.c.

References change_euid(), crcopy(), crfree(), crget(), seteuid_args::euid, priv_check_cred(), setsugid(), uifind(), and uifree().

Here is the call graph for this function:

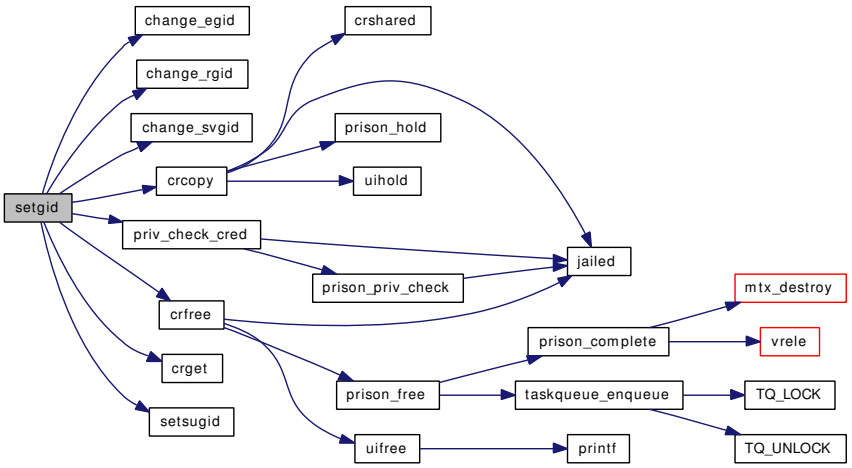


9.51.2.49 int setgid (struct thread * *td*, struct setgid_args * *uap*)

Definition at line 683 of file kern_prot.c.

References change_egid(), change_rgid(), change_svgid(), crcopy(), crfree(), crget(), setgid_args::gid, POSIX_APPENDIX_B_4_2_2, priv_check_cred(), and setsugid().

Here is the call graph for this function:

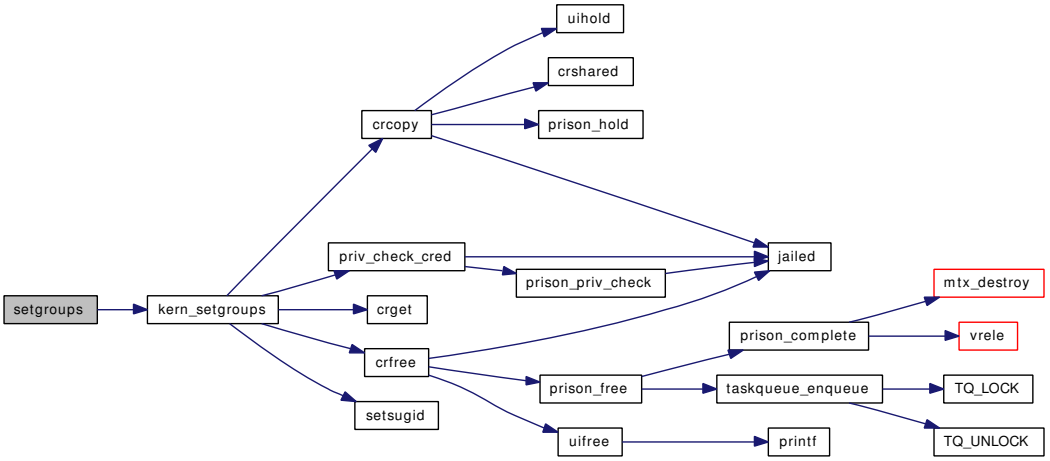


9.51.2.50 int setgroups (struct thread * *td*, struct setgroups_args * *uap*)

Definition at line 838 of file kern_prot.c.

References setgroups_args::gidset, setgroups_args::gidsetsize, and kern_setgroups().

Here is the call graph for this function:

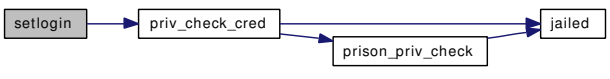


9.51.2.51 int setlogin (struct thread * *td*, struct setlogin_args * *uap*)

Definition at line 2024 of file kern_prot.c.

References setlogin_args::namebuf, and priv_check_cred().

Here is the call graph for this function:

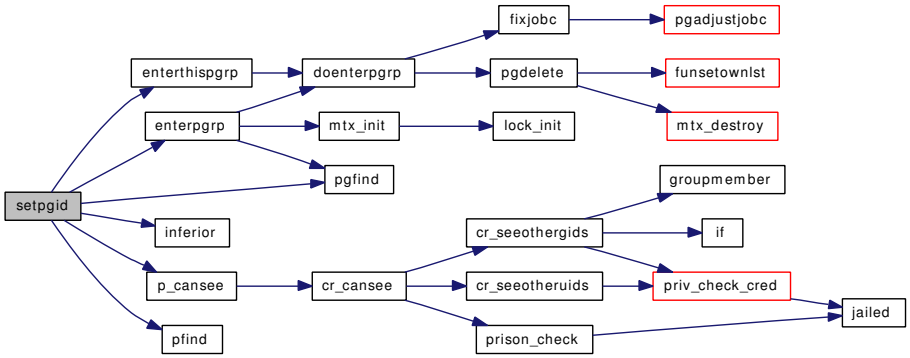


9.51.2.52 int setpgid (struct thread * *td*, register struct setpgid_args * *uap*)

Definition at line 405 of file kern_prot.c.

References enterpgrp(), enterthispgrp(), inferior(), p_cansee(), pfind(), pgfind(), and proctree_lock.

Here is the call graph for this function:

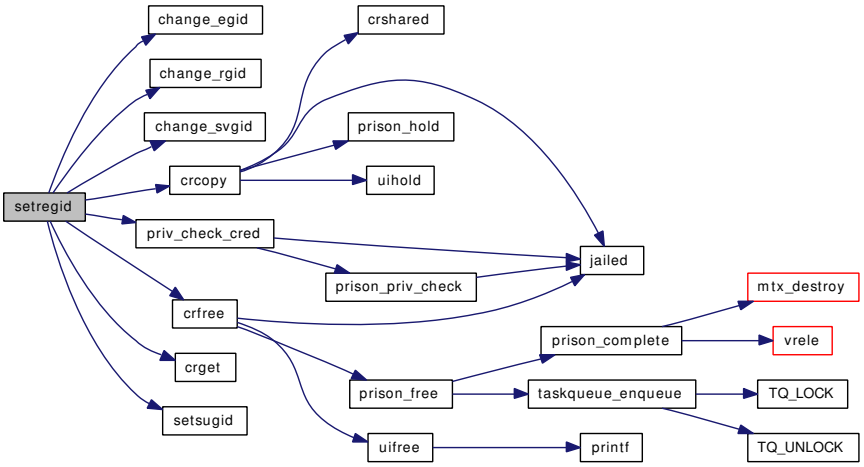


9.51.2.53 int setregid (register struct thread * *td*, struct setregid_args * *uap*)

Definition at line 988 of file kern_prot.c.

References change_egid(), change_rgid(), change_svgid(), crcopy(), crfree(), crget(), setregid_args::egid, priv_check_cred(), setregid_args::rgid, and setsugid().

Here is the call graph for this function:

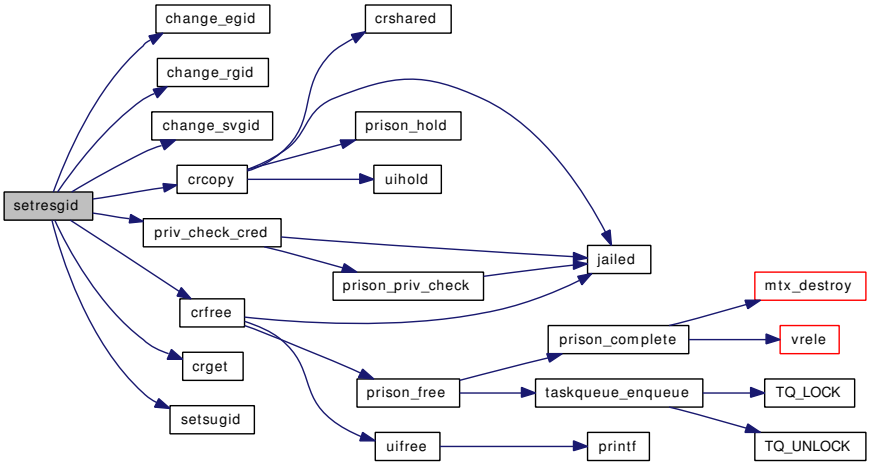


9.51.2.54 int setresgid (register struct thread * *td*, struct setresgid_args * *uap*)

Definition at line 1144 of file kern_prot.c.

References change_egid(), change_rgid(), change_svgid(), crcopy(), crfree(), crget(), setresgid_args::egid, priv_check_cred(), setresgid_args::rgid, setsugid(), and setresgid_args::sgid.

Here is the call graph for this function:

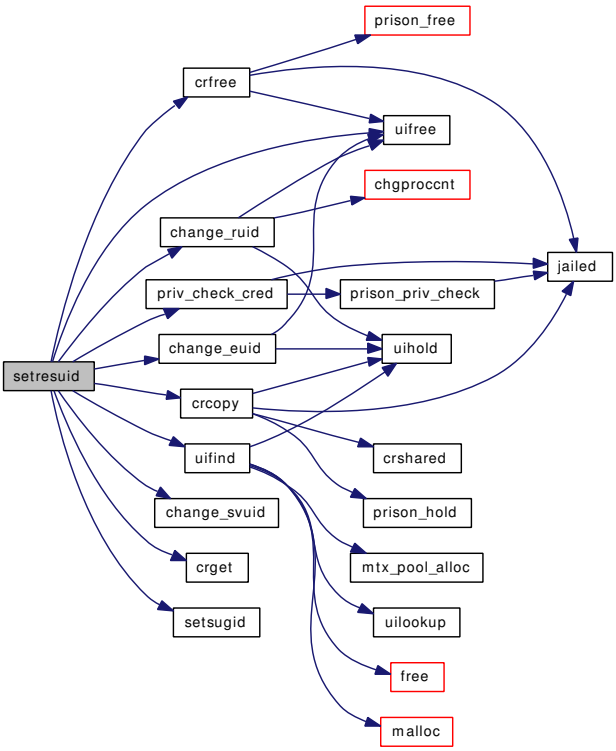


9.51.2.55 int setresuid (register struct thread * *td*, struct setresuid_args * *uap*)

Definition at line 1059 of file kern_prot.c.

References change_euid(), change_ruid(), change_svuid(), crcopy(), crfree(), crget(), setresuid_args::euid, priv_check_cred(), setresuid_args::ruid, setsugid(), setresuid_args::suid, uifind(), and uifree().

Here is the call graph for this function:

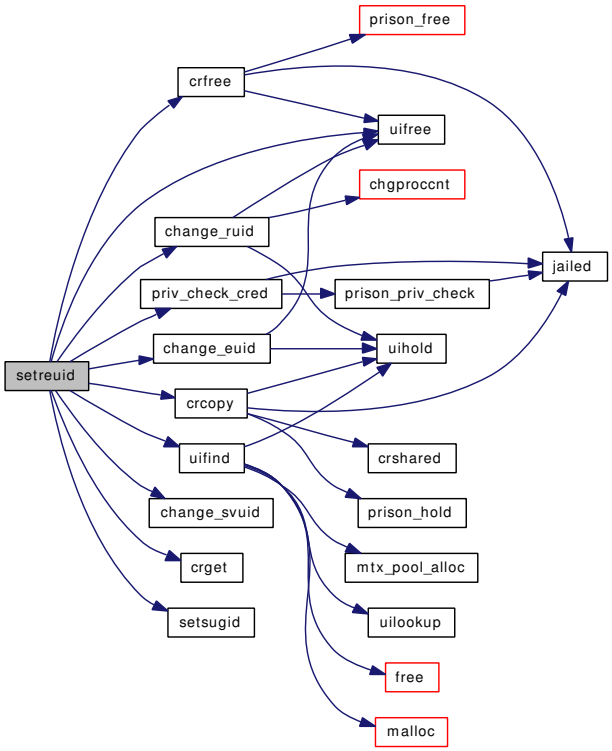


9.51.2.56 int setreuid (register struct thread * *td*, struct setreuid_args * *uap*)

Definition at line 916 of file kern_prot.c.

References change_euid(), change_ruid(), change_svuid(), crcopy(), crfree(), crget(), setreuid_args::euid, priv_check_cred(), setreuid_args::ruid, setsugid(), uifind(), and uifree().

Here is the call graph for this function:



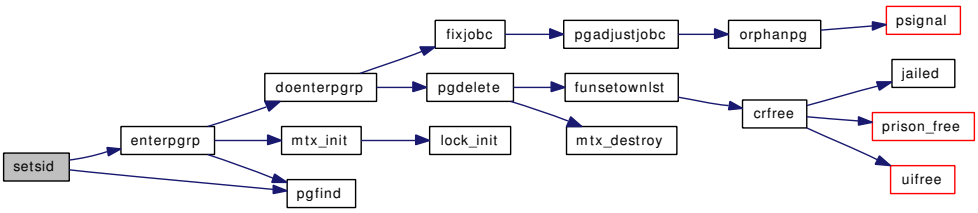
9.51.2.57 int setsid (register struct thread * *td*, struct setsid_args * *uap*)

Definition at line 344 of file kern_prot.c.

References enterpgrp(), pgfind(), and proctree_lock.

Referenced by aio_daemon().

Here is the call graph for this function:



9.51.2.58 void setsugid (struct proc * *p*)

Definition at line 2049 of file kern_prot.c.

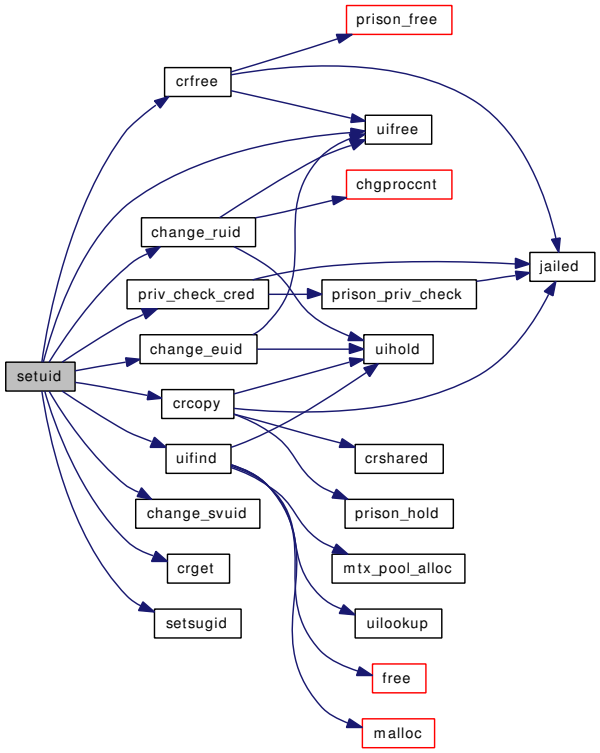
Referenced by jail_attach(), kern_setgroups(), setegid(), seteuid(), setgid(), setregid(), setresgid(), setresuid(), setreuid(), and setuid().

9.51.2.59 int setuid (struct thread * *td*, struct setuid_args * *uap*)

Definition at line 508 of file kern_prot.c.

References change_euid(), change_ruid(), change_svuid(), crcopy(), crfree(), crget(), POSIX_APPENDIX_B_4_2_2, priv_check_cred(), setsugid(), setuid_args::uid, uifind(), and uifree().

Here is the call graph for this function:



9.51.2.60 SYSCTL_INT (_security_bsd, OID_AUTO, unprivileged_proc_debug, CTLFLAG_RW, & *unprivileged_proc_debug*, 0, "Unprivileged processes may use process debugging facilities")

9.51.2.61 SYSCTL_INT (_security_bsd, OID_AUTO, conservative_signals, CTLFLAG_RW, & *conservative_signals*, 0, "Unprivileged processes prevented from ""sending certain signals to processes whose credentials have changed")

9.51.2.62 SYSCTL_INT (_security_bsd, OID_AUTO, see_other_gids, CTLFLAG_RW, & *see_other_gids*, 0, "Unprivileged processes may see subjects/objects with different real gid")

9.51.2.63 SYSCTL_INT (_security_bsd, OID_AUTO, see_other_uids, CTLFLAG_RW, & *see_other_uids*, 0, "Unprivileged processes may see subjects/objects with different real uid")

9.51.2.64 SYSCTL_NODE (_security, OID_AUTO, bsd, CTLFLAG_RW, 0, "BSD security policy")

9.51.3 Variable Documentation

9.51.3.1 int conservative_signals = 1 [static]

Definition at line 1501 of file kern_prot.c.

Referenced by cr_cansignal().

9.51.3.2 int see_other_gids = 1 [static]

Definition at line 1414 of file kern_prot.c.

Referenced by cr_seeothergids().

9.51.3.3 int see_other_uids = 1 [static]

Definition at line 1383 of file kern_prot.c.

Referenced by cr_seeotheruids().

9.51.3.4 int unprivileged_proc_debug = 1 [static]

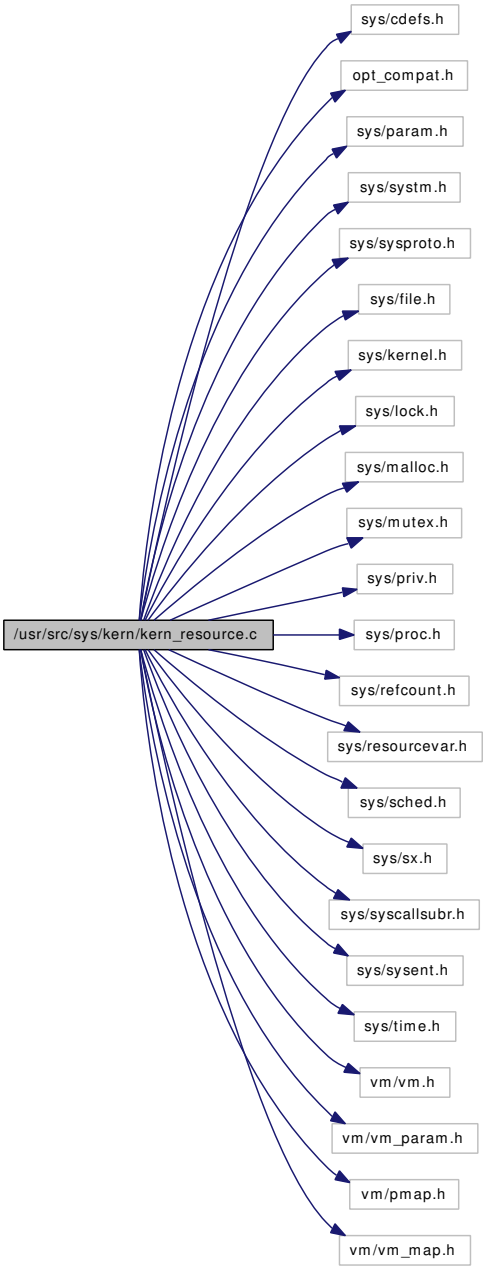
Definition at line 1673 of file kern_prot.c.

Referenced by p_candebug().

9.52 /usr/src/sys/kern/kern_resource.c File Reference

```
#include <sys/cdefs.h>
#include "opt_compat.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/sysproto.h>
#include <sys/file.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/refcount.h>
#include <sys/resourcevar.h>
#include <sys/sched.h>
#include <sys/sx.h>
#include <sys/syscallsubr.h>
#include <sys/sysent.h>
#include <sys/time.h>
#include <vm/vm.h>
#include <vm/vm_param.h>
#include <vm/pmap.h>
#include <vm/vm_map.h>
```

Include dependency graph for kern_resource.c:



Data Structures

- struct setpriority_args
- struct rtprio_thread_args
- struct rtprio_args
- struct __setrlimit_args
- struct __getrlimit_args
- struct getrusage_args

Defines

- #define UIHASH(uid) (&uihashtbl[(uid) & uihash])

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_resource.c,v 1.166 2007/02/19 13:22:36 rwatson Exp $")
- static MALLOC_DEFINE (M_PLIMIT,"plimit","plimit structures")
- static MALLOC_DEFINE (M_UIDINFO,"uidinfo","uidinfo structures")
- static LIST_HEAD (uihashhead, uidinfo)
- int getpriority (struct thread *td, struct getpriority_args *uap)
- int setpriority (struct thread *td, struct setpriority_args *uap)
- static int donice (struct thread *td, struct proc *p, int n)
- int rtprio_thread (struct thread *td, struct rtprio_thread_args *uap)
- int rtprio (struct thread *td, struct rtprio_args *uap)
- int rtp_to_pri (struct rtprio *rtp, struct thread *td)
- void pri_to_rtp (struct thread *td, struct rtprio *rtp)
- int setrlimit (struct thread *td, struct __setrlimit_args *uap)
- int kern_setrlimit (struct thread *td, u_int which, struct rlimit *limp)
- int getrlimit (struct thread *td, struct __getrlimit_args *uap)
- void calccru (struct proc *p, struct timeval *up, struct timeval *sp)
- void calcru (struct proc *p, struct timeval *up, struct timeval *sp)
- static void calcru1 (struct proc *p, struct rusage_ext *ruxp, struct timeval *up, struct timeval *sp)
- int getrusage (struct thread *td, struct getrusage_args *uap)
- int kern_getrusage (struct thread *td, int who, struct rusage *rup)
- void ruadd (struct rusage *ru, struct rusage_ext *rux, struct rusage *ru2, struct rusage_ext *rux2)
- plimit * lim_alloc ()
- plimit * lim_hold (struct plimit *limp)
- void lim_free (struct plimit *limp)
- void lim_copy (struct plimit *dst, struct plimit *src)
- rlim_t lim_max (struct proc *p, int which)
- rlim_t lim_cur (struct proc *p, int which)
- void lim_rlimit (struct proc *p, int which, struct rlimit *rlp)
- void uihashinit ()
- static struct uidinfo * uilookup (uid_t uid)
- uidinfo * uifind (uid_t uid)
- void uihold (struct uidinfo *uip)
- void uifree (struct uidinfo *uip)
- int chgproccnt (struct uidinfo *uip, int diff, int max)
- int chgsbsize (struct uidinfo *uip, u_int *hiwat, u_int to, rlim_t max)

Variables

- static struct mtx uihashtbl_mtx

9.52.1 Define Documentation

9.52.1.1 #define UIHASH(uid) (&uihashtbl[(uid) & uihash])

Definition at line 68 of file kern_resource.c.

Referenced by uifind(), and uilookup().

9.52.2 Function Documentation

9.52.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_resource. *c*, v 1.166 2007/02/19 13:22:36 rwatson Exp $")

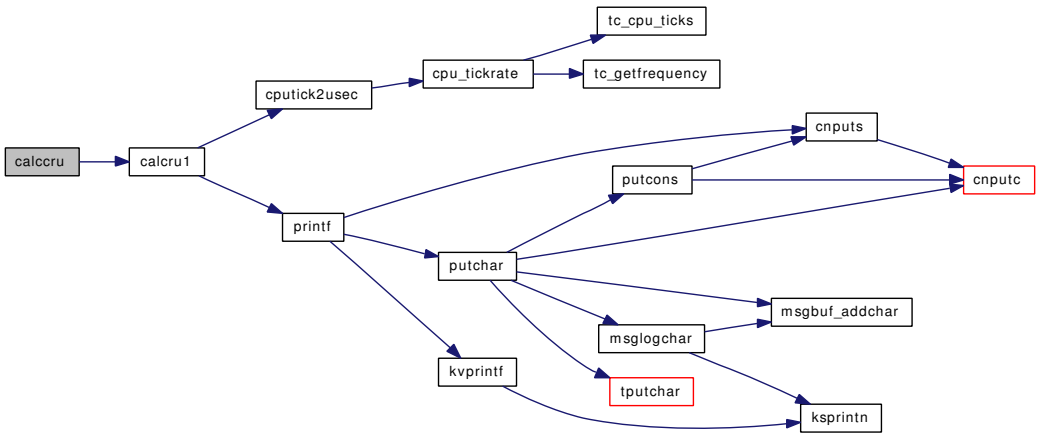
9.52.2.2 void calccru (struct proc * *p*, struct timeval * *up*, struct timeval * *sp*)

Definition at line 799 of file kern_resource.c.

References calcru1().

Referenced by fill_kinfo_proc_only(), and kern_getrusage().

Here is the call graph for this function:



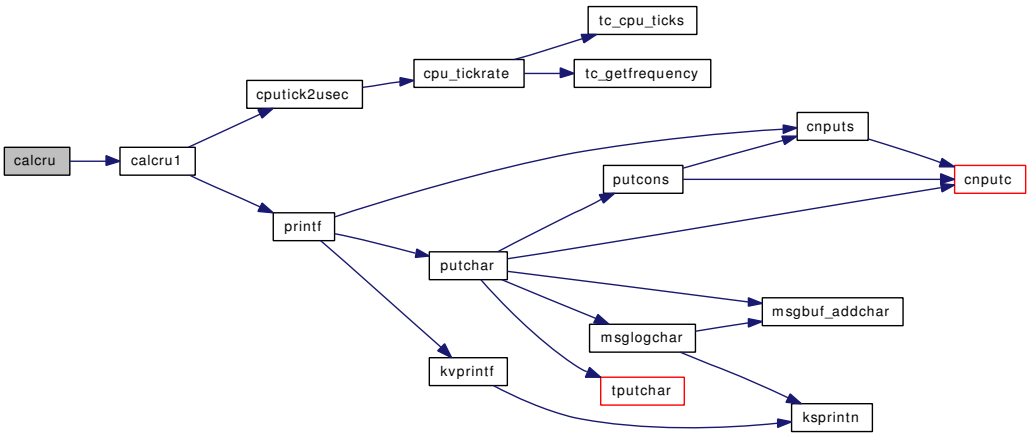
9.52.2.3 void calcru (struct proc * *p*, struct timeval * *up*, struct timeval * *sp*)

Definition at line 815 of file kern_resource.c.

References calcru1(), cpu_ticks, and sched_lock.

Referenced by acct_process(), fill_kinfo_proc_only(), kern_clock_gettime(), kern_getrusage(), kern_wait(), and ttyinfo().

Here is the call graph for this function:



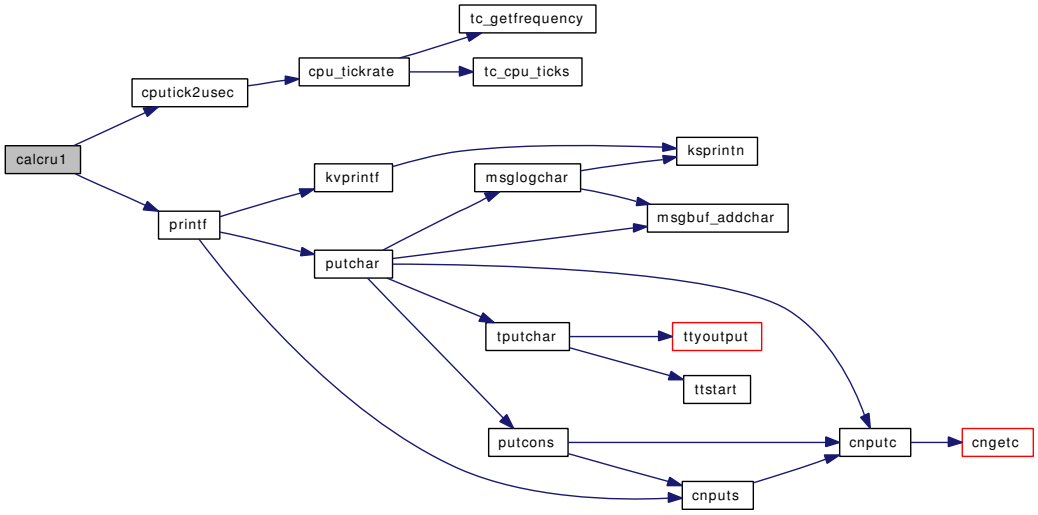
9.52.2.4 static void calcru1 (struct proc * *p*, struct rusage_ext * *ruxp*, struct timeval * *up*, struct timeval * *sp*) [static]

Definition at line 854 of file kern_resource.c.

References cputick2usec(), and printf().

Referenced by calccru(), and calcru().

Here is the call graph for this function:



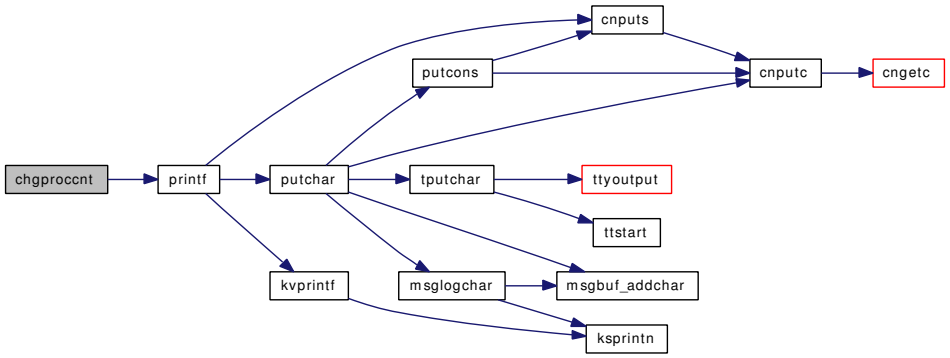
9.52.2.5 int chgproccnt (struct uidinfo * *uip*, int *diff*, int *max*)

Definition at line 1238 of file kern_resource.c.

References printf().

Referenced by change_ruid(), fork1(), kern_wait(), and proc0_init().

Here is the call graph for this function:



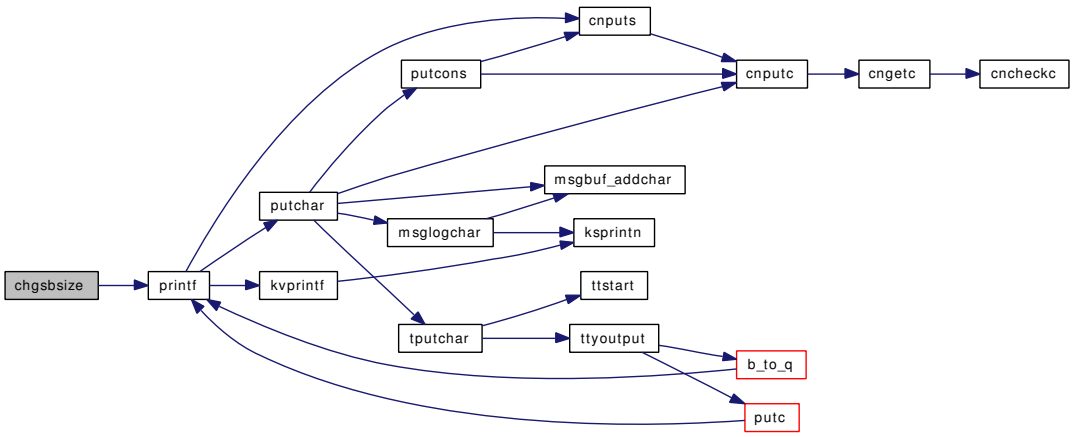
9.52.2.6 int chgsbsize (struct uidinfo * *uip*, u_int * *hiwat*, u_int *to*, rlim_t *max*)

Definition at line 1261 of file kern_resource.c.

References printf().

Referenced by sbrelease_internal(), sbreserve_locked(), sodealloc(), uipc_rcvd(), and uipc_send().

Here is the call graph for this function:



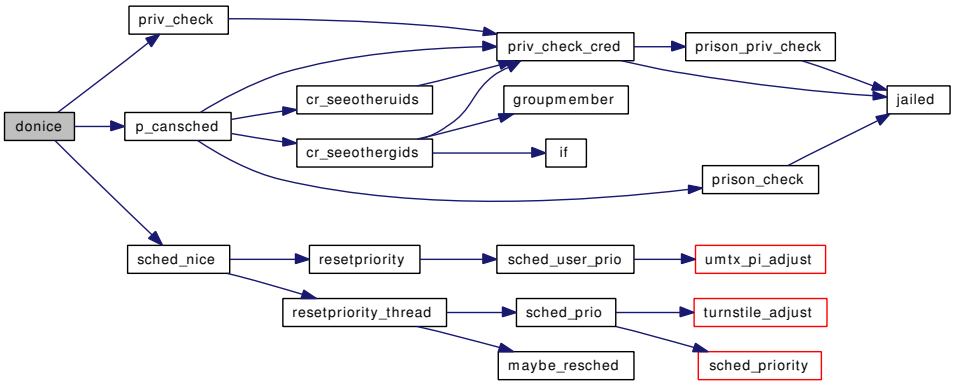
9.52.2.7 static int donice (struct thread * *td*, struct proc * *p*, int *n*) [static]

Definition at line 257 of file kern_resource.c.

References p_cansched(), priv_check(), sched_lock, and sched_nice().

Referenced by setpriority().

Here is the call graph for this function:

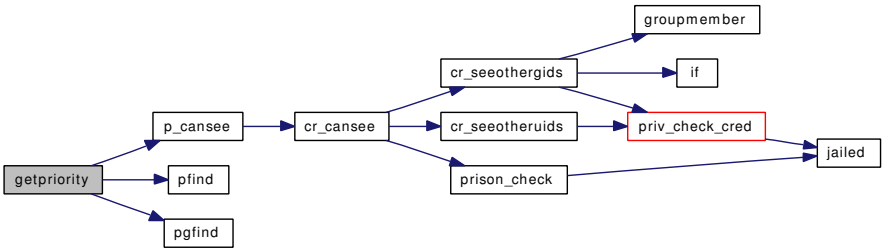


9.52.2.8 int getpriority (struct thread * *td*, struct getpriority_args * *uap*)

Definition at line 92 of file kern_resource.c.

References allproc_lock, p_cansee(), pfind(), pgfind(), and proctree_lock.

Here is the call graph for this function:



9.52.2.9 int getrlimit (struct thread * *td*, struct __getrlimit_args * *uap*)

Definition at line 776 of file kern_resource.c.

References lim_rlimit().

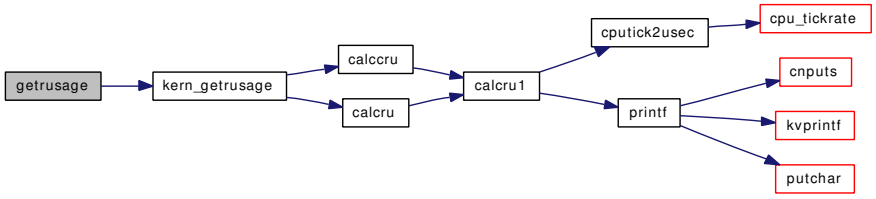
Here is the call graph for this function:

9.52.2.10 int getrusage (struct thread * *td*, struct getrusage_args * *uap*)

Definition at line 938 of file kern_resource.c.

References kern_getrusage().

Here is the call graph for this function:



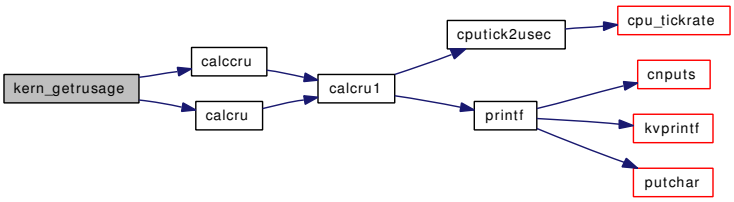
9.52.2.11 int kern_getrusage (struct thread * *td*, int *who*, struct rusage * *rup*)

Definition at line 952 of file kern_resource.c.

References calccru(), and calcru().

Referenced by getrusage().

Here is the call graph for this function:



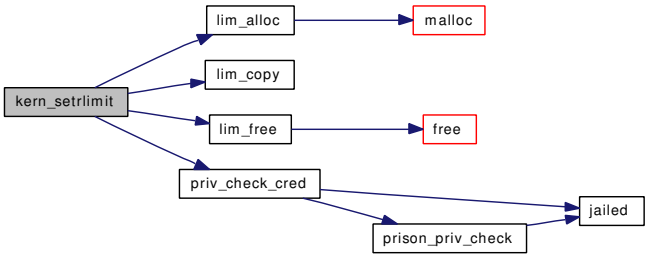
9.52.2.12 int kern_setrlimit (struct thread * *td*, u_int *which*, struct rlimit * *limp*)

Definition at line 642 of file kern_resource.c.

References lim_alloc(), lim_copy(), lim_free(), maxdsiz, maxfilesperproc, maxprocperuid, maxssiz, priv_check_cred(), and sched_lock.

Referenced by setrlimit().

Here is the call graph for this function:



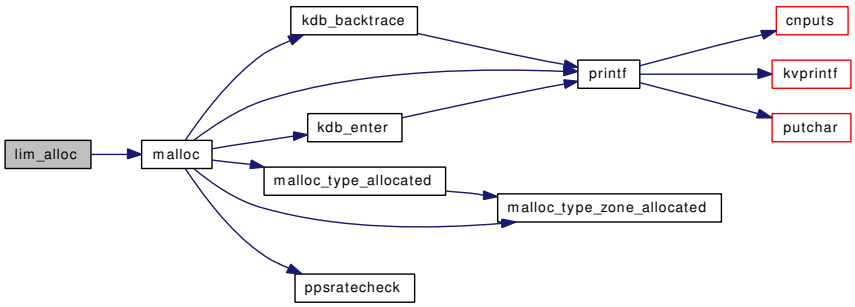
9.52.2.13 struct plimit* lim_alloc ()

Definition at line 1011 of file kern_resource.c.

References malloc().

Referenced by acct_process(), kern_setrlimit(), and proc0_init().

Here is the call graph for this function:



9.52.2.14 void lim_copy (struct plimit * *dst*, struct plimit * *src*)

Definition at line 1044 of file kern_resource.c.

Referenced by acct_process(), and kern_setrlimit().

9.52.2.15 rlim_t lim_cur (struct proc * *p*, int *which*)

Definition at line 1070 of file kern_resource.c.

References lim_rlimit().

Referenced by coredump(), do_aout_hdr(), do_dup(), exec_aout_imgact(), fdalloc(), fdavail(), fork1(), getdtablesize(), kern_fcntl(), poll(), and sbreserve_locked().

Here is the call graph for this function:

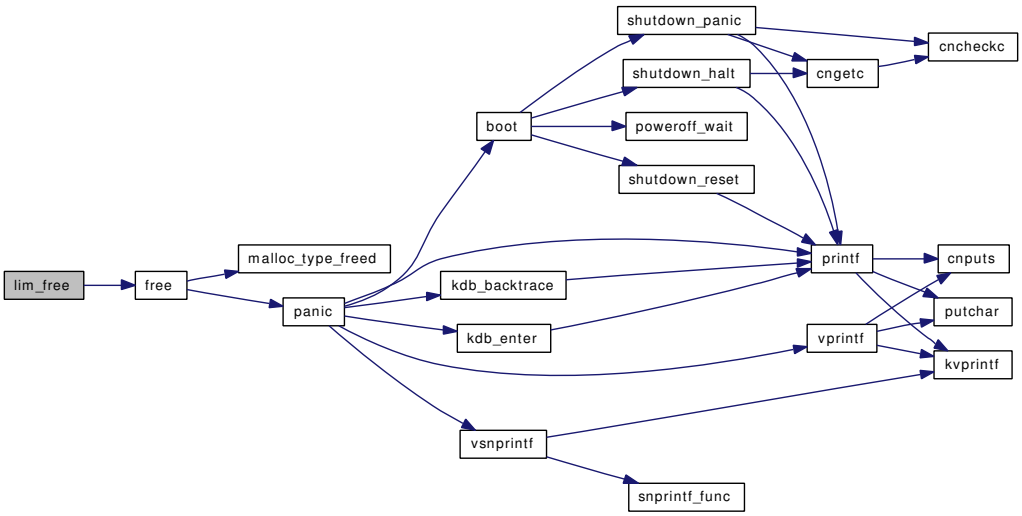
9.52.2.16 void lim_free (struct plimit * *limp*)

Definition at line 1030 of file kern_resource.c.

References free().

Referenced by acct_process(), exit1(), and kern_setrlimit().

Here is the call graph for this function:



9.52.2.17 struct plimit* lim_hold (struct plimit * *limp*)

Definition at line 1021 of file kern_resource.c.

Referenced by fork1().

9.52.2.18 rlim_t lim_max (struct proc * *p*, int *which*)

Definition at line 1057 of file kern_resource.c.

References lim_rlimit().

Here is the call graph for this function:

9.52.2.19 void lim_rlimit (struct proc * *p*, int *which*, struct rlimit * *rlp*)

Definition at line 1083 of file kern_resource.c.

Referenced by ast(), getrlimit(), lim_cur(), and lim_max().

9.52.2.20 static LIST_HEAD (uihashhead, uidinfo) [static]

Definition at line 70 of file kern_resource.c.

9.52.2.21 static MALLOC_DEFINE (M_UIDINFO, "uidinfo", "uidinfo structures") [static]

9.52.2.22 static MALLOC_DEFINE (M_PLIMIT, "plimit", "plimit structures") [static]

9.52.2.23 void pri_to_rtp (struct thread * *td*, struct rtprio * *rtp*)

Definition at line 529 of file kern_resource.c.

References sched_lock.

Referenced by getscheduler(), ksched_getparam(), rtprio(), and rtprio_thread().

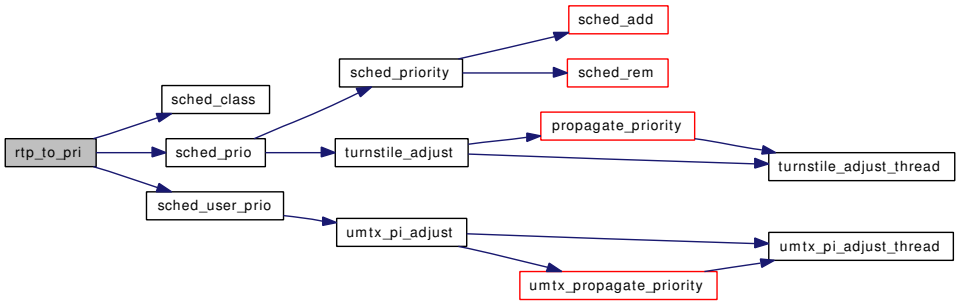
9.52.2.24 int rtp_to_pri (struct rtprio * *rtp*, struct thread * *td*)

Definition at line 501 of file kern_resource.c.

References sched_class(), sched_lock, sched_prio(), and sched_user_prio().

Referenced by create_thread(), ksched_setscheduler(), poll_idle(), rtprio(), and rtprio_thread().

Here is the call graph for this function:



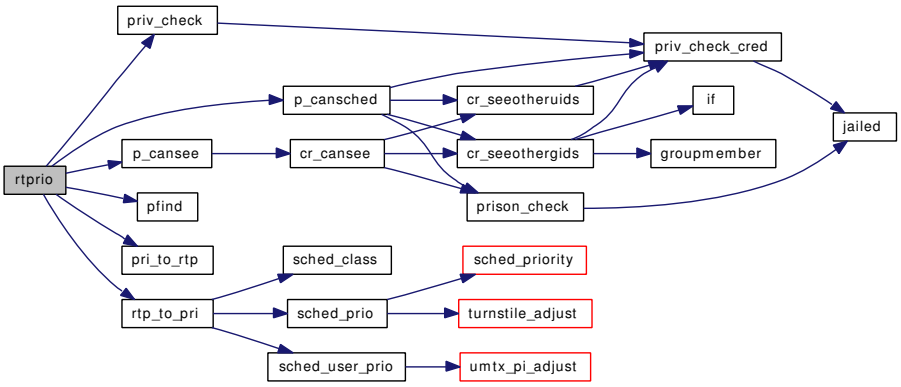
9.52.2.25 int rtprio (struct thread * *td*, struct rtprio_args * *uap*)

Definition at line 385 of file kern_resource.c.

References p_cansched(), p_cansee(), pfind(), pri_to_rtp(), priv_check(), rtp_to_pri(), and sched_lock.

Referenced by getscheduler(), kern_thr_new(), ksched_getparam(), ksched_setscheduler(), poll_idle(), and rtprio_thread().

Here is the call graph for this function:

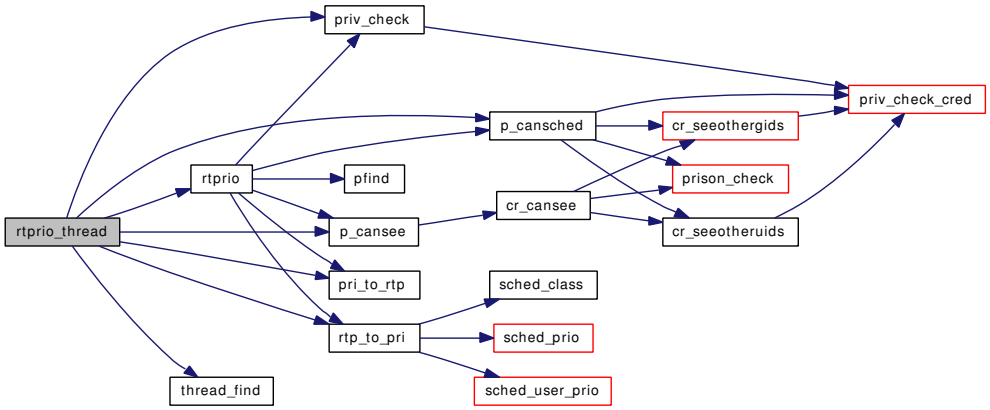


9.52.2.26 int rtprio_thread (struct thread * *td*, struct rtprio_thread_args * *uap*)

Definition at line 290 of file kern_resource.c.

References rtprio_thread_args::function, rtprio_thread_args::lwpid, p_cansched(), p_cansee(), pri_to_rtp(), priv_check(), rtprio_thread_args::rtp, rtp_to_pri(), rtprio(), sched_lock, and thread_find().

Here is the call graph for this function:



9.52.2.27 void ruadd (struct rusage * *ru*, struct rusage_ext * *rux*, struct rusage * *ru2*, struct rusage_ext * *rux2*)

Definition at line 982 of file kern_resource.c.

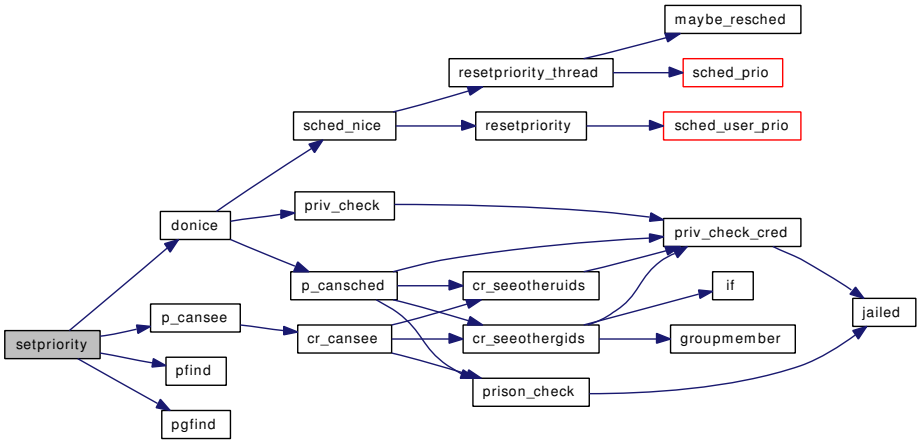
Referenced by kern_wait(), and thread_exit().

9.52.2.28 int setpriority (struct thread * *td*, struct setpriority_args * *uap*)

Definition at line 178 of file kern_resource.c.

References allproc_lock, donice(), p_cansee(), pfind(), pgfind(), setpriority_args::prio, proctree_lock, setpriority_args::which, and setpriority_args::who.

Here is the call graph for this function:

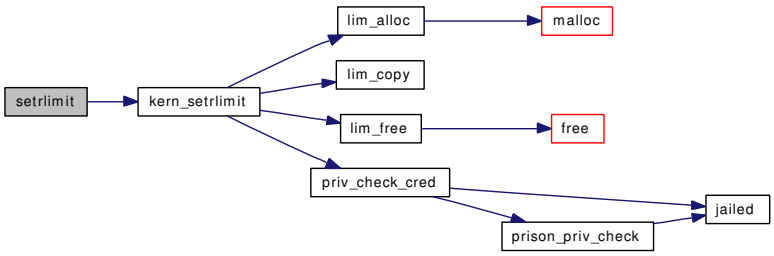


9.52.2.29 int setrlimit (struct thread * *td*, struct __setrlimit_args * *uap*)

Definition at line 628 of file kern_resource.c.

References kern_setrlimit().

Here is the call graph for this function:



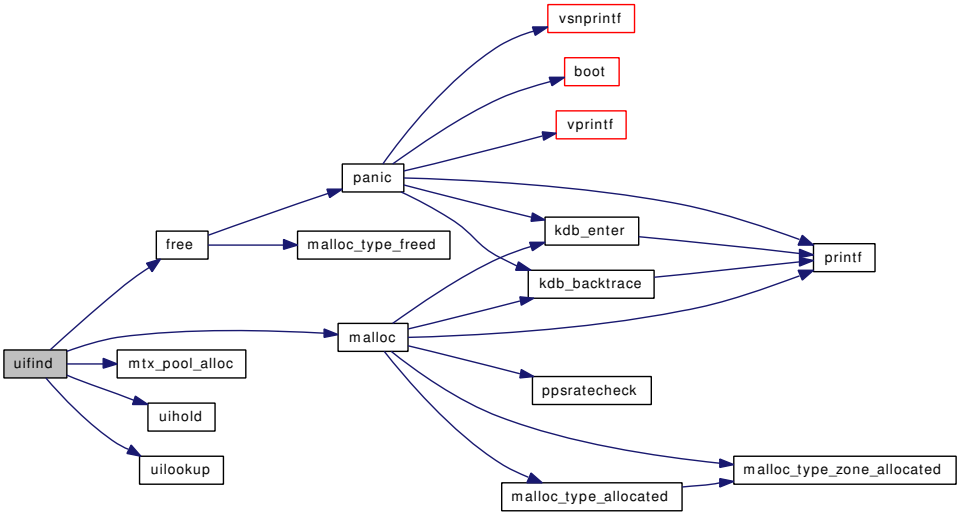
9.52.2.30 struct uidinfo* uifind (uid_t *uid*)

Definition at line 1131 of file kern_resource.c.

References free(), malloc(), mtx_pool_alloc(), mtxpool_sleep, UIHASH, uihashtbl_mtx, uihold(), and uilookup().

Referenced by proc0_init(), seteuid(), setresuid(), setreuid(), and setuid().

Here is the call graph for this function:



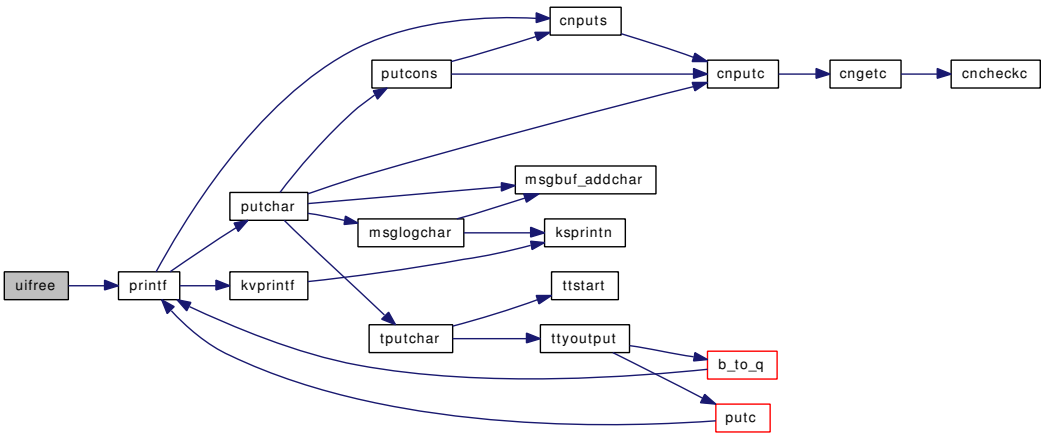
9.52.2.31 void uifree (struct uidinfo * *uip*)

Definition at line 1191 of file kern_resource.c.

References printf(), and uihashtbl_mtx.

Referenced by change_euid(), change_ruid(), crfree(), seteuid(), setresuid(), setreuid(), and setuid().

Here is the call graph for this function:



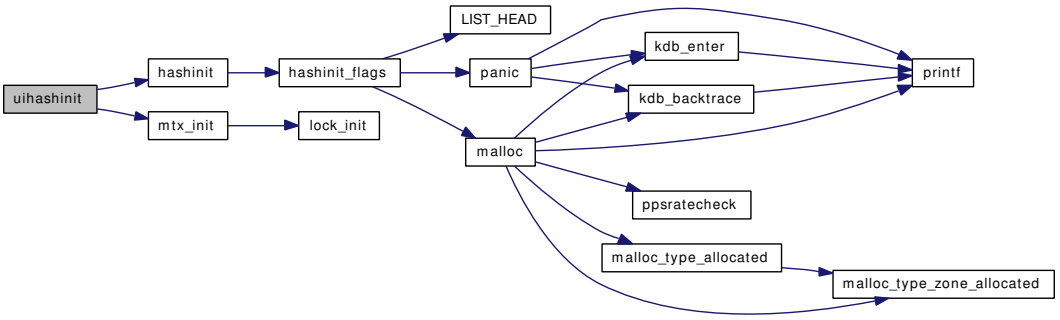
9.52.2.32 void uihashinit ()

Definition at line 1098 of file kern_resource.c.

References hashinit(), maxproc, mtx_init(), and uihashtbl_mtx.

Referenced by procinit().

Here is the call graph for this function:



9.52.2.33 void uihold (struct uidinfo * *uip*)

Definition at line 1166 of file kern_resource.c.

Referenced by change_euid(), change_ruid(), crcopy(), and uifind().

9.52.2.34 static struct uidinfo* uilookup (uid_t *uid*) [static]

Definition at line 1110 of file kern_resource.c.

References UIHASH, and uihashtbl_mtx.

Referenced by uifind().

9.52.3 Variable Documentation

9.52.3.1 struct mtx uihashtbl_mtx [static]

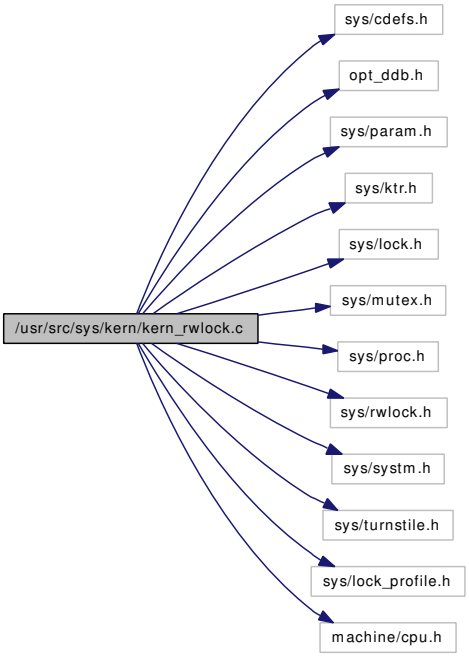
Definition at line 69 of file kern_resource.c.

Referenced by uifind(), uifree(), uihashinit(), and uilookup().

9.53 /usr/src/sys/kern/kern_rwlock.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ddb.h"
#include <sys/param.h>
#include <sys/ktr.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/rwlock.h>
#include <sys/systm.h>
#include <sys/turnstile.h>
#include <sys/lock_profile.h>
#include <machine/cpu.h>
```

Include dependency graph for kern_rwlock.c:



Defines

- #define rw_wowner(rw)
- #define rw_owner(rw) rw_wowner(rw)
- #define _rw_assert(rw, what, file, line)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_rwlock.c,v 1.12 2006/11/13 05:41:46 kmacy Exp $")
- void rw_init (struct rwlock *rw, const char *name)
- void rw_destroy (struct rwlock *rw)
- void rw_sysinit (void *arg)
- void _rw_wlock (struct rwlock *rw, const char *file, int line)
- void _rw_wunlock (struct rwlock *rw, const char *file, int line)
- void _rw_rlock (struct rwlock *rw, const char *file, int line)
- void _rw_runlock (struct rwlock *rw, const char *file, int line)
- void _rw_wlock_hard (struct rwlock *rw, uintptr_t tid, const char *file, int line)
- void _rw_wunlock_hard (struct rwlock *rw, uintptr_t tid, const char *file, int line)
- int _rw_try_upgrade (struct rwlock *rw, const char *file, int line)
- void _rw_downgrade (struct rwlock *rw, const char *file, int line)

Variables

- lock_class lock_class_rw

9.53.1 Define Documentation

9.53.1.1 #define _rw_assert(rw, what, file, line)

Definition at line 80 of file kern_rwlock.c.

Referenced by _rw_downgrade(), _rw_runlock(), _rw_try_upgrade(), and _rw_wunlock().

9.53.1.2 #define rw_owner(rw) rw_wowner(rw)

Definition at line 77 of file kern_rwlock.c.

Referenced by _rw_rlock(), and _rw_wlock_hard().

9.53.1.3 #define rw_wowner(rw)

Value:

```
((rw)->rw_lock & RW_LOCK_READ ? NULL :                          \
            (struct thread *)RW_OWNER((rw)->rw_lock))
```

Definition at line 68 of file kern_rwlock.c.

Referenced by _rw_rlock(), and _rw_wlock().

9.53.2 Function Documentation

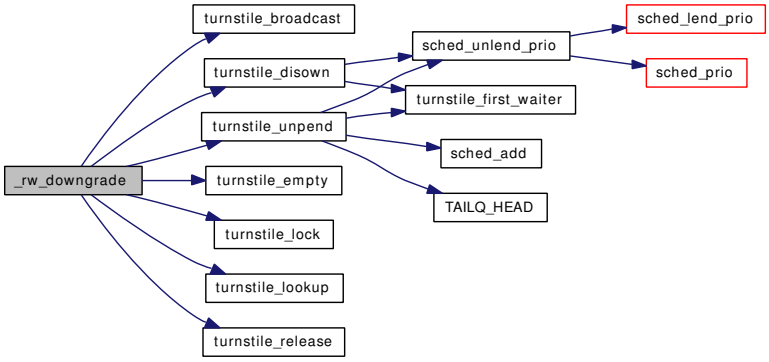
9.53.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_rwlock. *c*, v 1.12 2006/11/13 05:41:46 kmacy Exp $")

9.53.2.2 void _rw_downgrade (struct rwlock * *rw*, const char * *file*, int *line*)

Definition at line 674 of file kern_rwlock.c.

References _rw_assert, turnstile_broadcast(), turnstile_disown(), turnstile_empty(), turnstile_lock(), turnstile_lookup(), turnstile_release(), and turnstile_unpend().

Here is the call graph for this function:

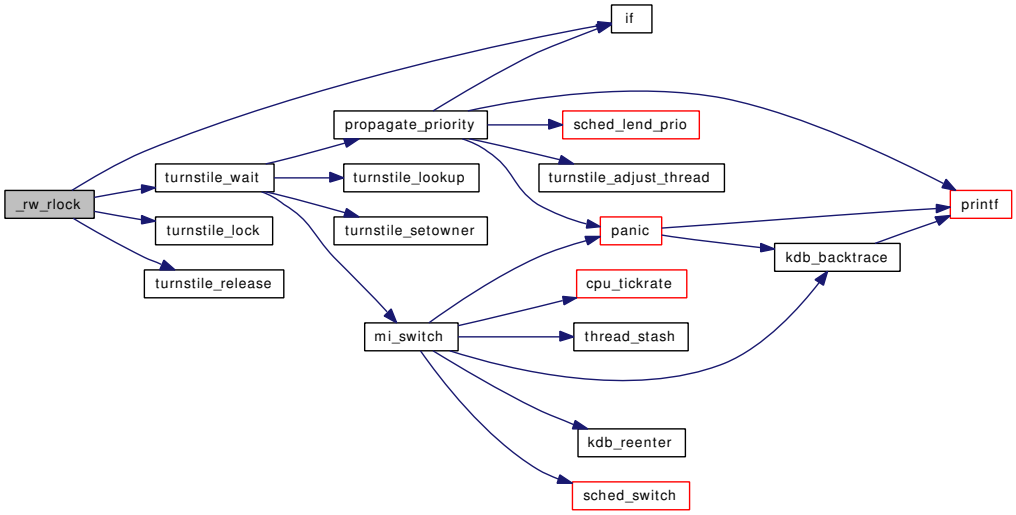


9.53.2.3 void _rw_rlock (struct rwlock * *rw*, const char * *file*, int *line*)

Definition at line 144 of file kern_rwlock.c.

References if(), rw_owner, rw_wowner, turnstile_lock(), turnstile_release(), and turnstile_wait().

Here is the call graph for this function:

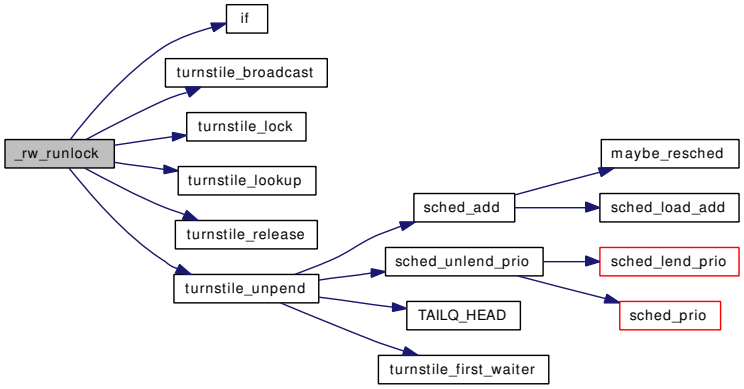


9.53.2.4 void _rw_runlock (struct rwlock * *rw*, const char * *file*, int *line*)

Definition at line 287 of file kern_rwlock.c.

References _rw_assert, if(), turnstile_broadcast(), turnstile_lock(), turnstile_lookup(), turnstile_release(), and turnstile_unpend().

Here is the call graph for this function:

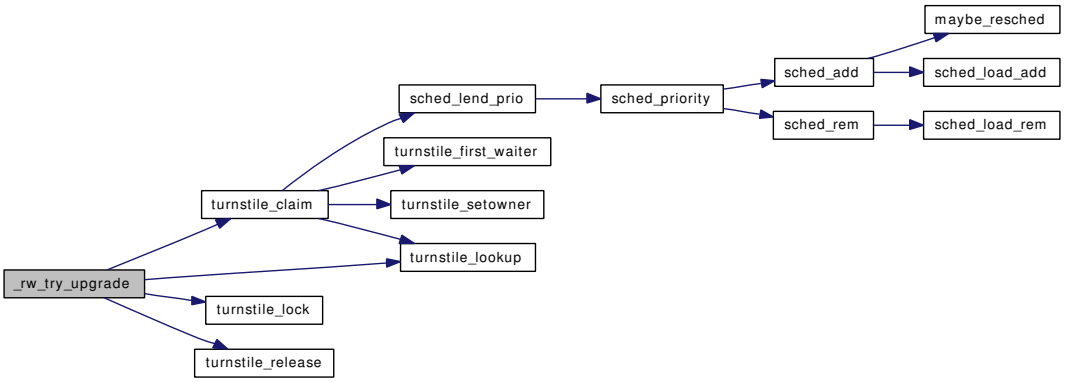


9.53.2.5 int _rw_try_upgrade (struct rwlock * *rw*, const char * *file*, int *line*)

Definition at line 616 of file kern_rwlock.c.

References _rw_assert, turnstile_claim(), turnstile_lock(), turnstile_lookup(), and turnstile_release().

Here is the call graph for this function:



9.53.2.6 void _rw_wlock (struct rwlock * *rw*, const char * *file*, int *line*)

Definition at line 112 of file kern_rwlock.c.

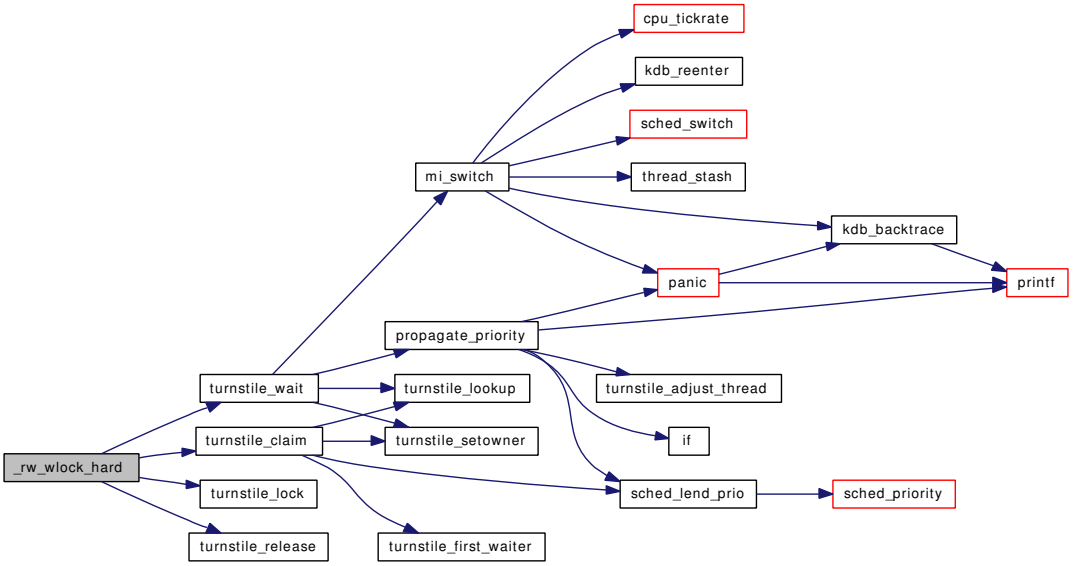
References rw_wowner.

9.53.2.7 void _rw_wlock_hard (struct rwlock * *rw*, uintptr_t *tid*, const char * *file*, int *line*)

Definition at line 409 of file kern_rwlock.c.

References rw_owner, turnstile_claim(), turnstile_lock(), turnstile_release(), and turnstile_wait().

Here is the call graph for this function:



9.53.2.8 void _rw_wunlock (struct rwlock * *rw*, const char * *file*, int *line*)

Definition at line 131 of file kern_rwlock.c.

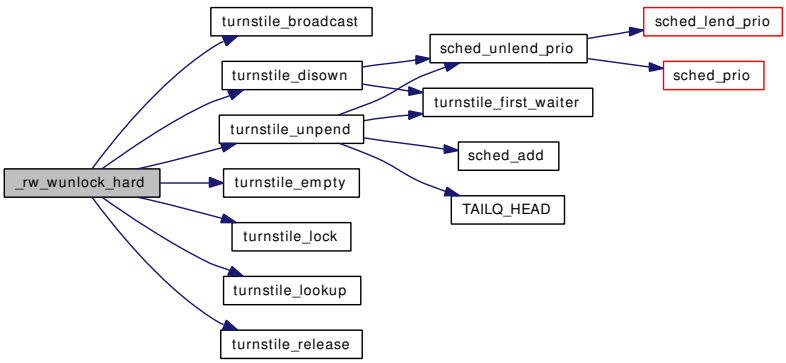
References _rw_assert.

9.53.2.9 void _rw_wunlock_hard (struct rwlock * *rw*, uintptr_t *tid*, const char * *file*, int *line*)

Definition at line 519 of file kern_rwlock.c.

References turnstile_broadcast(), turnstile_disown(), turnstile_empty(), turnstile_lock(), turnstile_-lookup(), turnstile_release(), and turnstile_unpend().

Here is the call graph for this function:



9.53.2.10 void rw_destroy (struct rwlock * *rw*)

Definition at line 95 of file kern_rwlock.c.

References lock_destroy().

Here is the call graph for this function:

9.53.2.11 void rw_init (struct rwlock * *rw*, const char * *name*)

Definition at line 84 of file kern_rwlock.c.

References lock_class_rw, and lock_init().

Referenced by rw_sysinit().

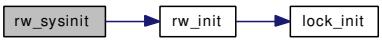
Here is the call graph for this function:

9.53.2.12 void rw_sysinit (void * *arg*)

Definition at line 104 of file kern_rwlock.c.

References rw_init().

Here is the call graph for this function:



9.53.3 Variable Documentation

9.53.3.1 struct lock_class lock_class_rw

Initial value:

```
{
        "rw",
        LC_SLEEPLOCK | LC_RECURSABLE | LC_UPGRADABLE,


}
```

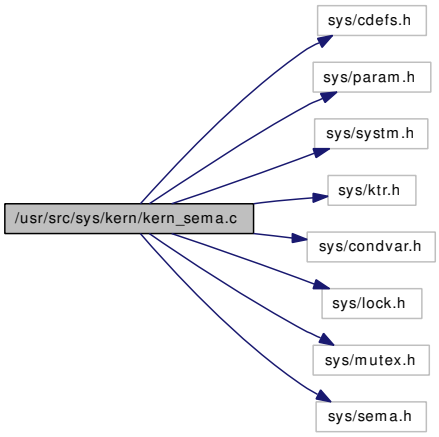
Definition at line 56 of file kern_rwlock.c.

Referenced by rw_init().

9.54 /usr/src/sys/kern/kern_sema.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/ktr.h>
#include <sys/condvar.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/sema.h>
```

Include dependency graph for kern_sema.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_sema.c,v 1.6 2005/01/06 23:35:39 imp Exp $")
- void sema_init (struct sema *sema, int value, const char *description)
- void sema_destroy (struct sema *sema)
- void _sema_post (struct sema *sema, const char *file, int line)
- void _sema_wait (struct sema *sema, const char *file, int line)
- int _sema_timedwait (struct sema *sema, int timo, const char *file, int line)
- int _sema_trywait (struct sema *sema, const char *file, int line)
- int sema_value (struct sema *sema)

9.54.1 Function Documentation

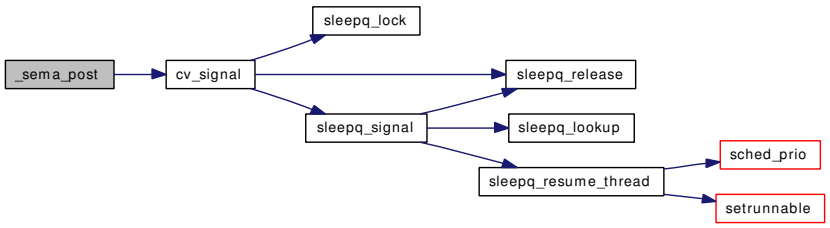
9.54.1.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_sema. *c*, v 1.6 2005/01/06 23:35:39 imp Exp $")

9.54.1.2 void _sema_post (struct sema * *sema*, const char * *file*, int *line*)

Definition at line 76 of file kern_sema.c.

References cv_signal().

Here is the call graph for this function:

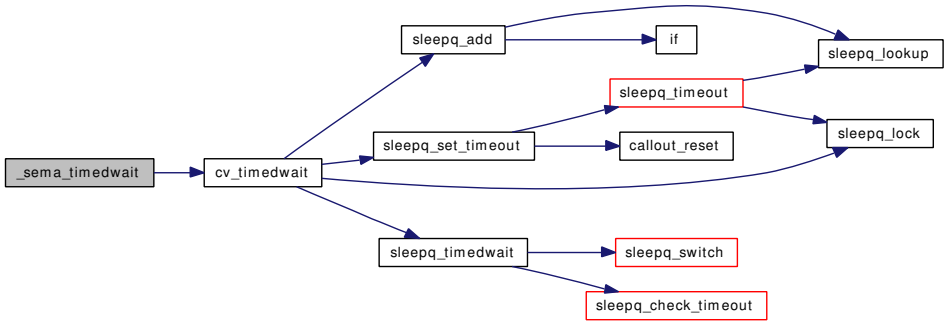


9.54.1.3 int _sema_timedwait (struct sema * *sema*, int *timo*, const char * *file*, int *line*)

Definition at line 109 of file kern_sema.c.

References cv_timedwait().

Here is the call graph for this function:



9.54.1.4 int _sema_trywait (struct sema * *sema*, const char * *file*, int *line*)

Definition at line 143 of file kern_sema.c.

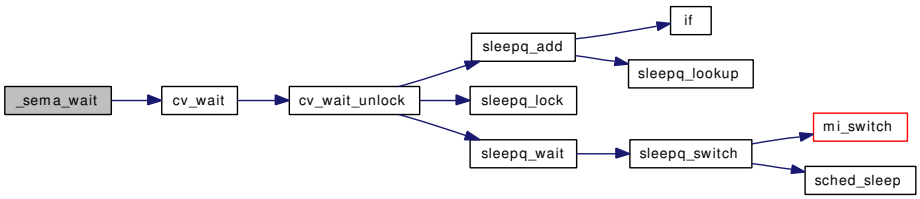
References ret.

9.54.1.5 void _sema_wait (struct sema * *sema*, const char * *file*, int *line*)

Definition at line 91 of file kern_sema.c.

References cv_wait().

Here is the call graph for this function:



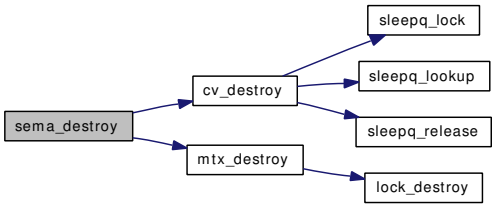
9.54.1.6 void sema_destroy (struct sema * *sema*)

Definition at line 63 of file kern_sema.c.

References cv_destroy(), and mtx_destroy().

Referenced by aio_unload().

Here is the call graph for this function:



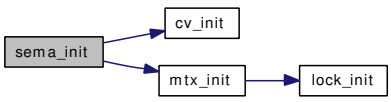
9.54.1.7 void sema_init (struct sema * *sema*, int *value*, const char * *description*)

Definition at line 48 of file kern_sema.c.

References cv_init(), and mtx_init().

Referenced by aio_onceonly().

Here is the call graph for this function:



9.54.1.8 int sema_value (struct sema * *sema*)

Definition at line 168 of file kern_sema.c.

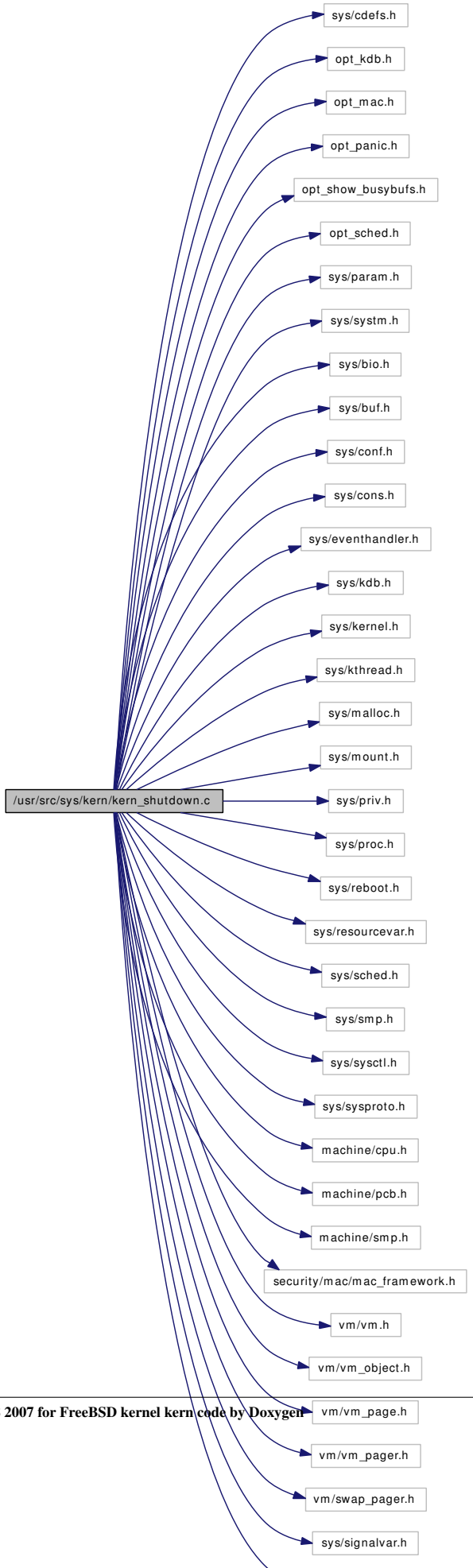
References ret.

9.55 /usr/src/sys/kern/kern_shutdown.c File Reference

```
#include <sys/cdefs.h>
#include "opt_kdb.h"
#include "opt_mac.h"
#include "opt_panic.h"
#include "opt_show_busybufs.h"
#include "opt_sched.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/bio.h>
#include <sys/buf.h>
#include <sys/conf.h>
#include <sys/cons.h>
#include <sys/eventhandler.h>
#include <sys/kdb.h>
#include <sys/kernel.h>
#include <sys/kthread.h>
#include <sys/malloc.h>
#include <sys/mount.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/reboot.h>
#include <sys/resourcevar.h>
#include <sys/sched.h>
#include <sys/smp.h>
#include <sys/sysctl.h>
#include <sys/sysproto.h>
#include <machine/cpu.h>
#include <machine/pcb.h>
#include <machine/smp.h>
#include <security/mac/mac_framework.h>
#include <vm/vm.h>
#include <vm/vm_object.h>
#include <vm/vm_page.h>
#include <vm/vm_pager.h>
#include <vm/swap_pager.h>
```

```
#include <sys/signalvar.h>
#include <machine/stdarg.h>
```

Include dependency graph for kern_shutdown.c:


/usr/src/sys/kern/kern_shutdown.c
sys/cdefs.h
opt_kdb.h
opt_mac.h
opt_panic.h
opt_show_busybufs.h
opt_sched.h
sys/param.h
sys/systm.h
sys/bio.h
sys/buf.h
sys/conf.h
sys/cons.h
sys/eventhandler.h
sys/kdb.h
sys/kernel.h
sys/kthread.h
sys/malloc.h
sys/mount.h
sys/priv.h
sys/proc.h
sys/reboot.h
sys/resourcevar.h
sys/sched.h
sys/smp.h
sys/sysctl.h
sys/sysproto.h
machine/cpu.h
machine/pcb.h
machine/smp.h
security/mac/mac_framework.h
vm/vm.h
vm/vm_object.h
vm/vm_page.h
vm/vm_pager.h
vm/swap_pager.h
sys/signalvar.h

Defines

- #define PANIC_REBOOT_WAIT_TIME 15
- #define POWEROFF_DELAY 5000

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_shutdown.c,v 1.180 2006/11/06 13:42:00 rwatson Exp $")
- SYSCTL_INT (_kern, OID_AUTO, sync_on_panic, CTLFLAG_RW,&sync_on_panic, 0,"Do a sync before rebooting from a panic")
- SYSCTL_NODE (_kern, OID_AUTO, shutdown, CTLFLAG_RW, 0,"Shutdown environment")
- static void boot (int)
- int reboot (struct thread *td, struct reboot_args *uap)
- void shutdown_nice (int howto)
- static void print_uptime (void)
- static void doadump (void)
- static int isbufbusy (struct buf *bp)
- static void shutdown_halt (void *junk, int howto)
- static void shutdown_panic (void *junk, int howto)
- static void shutdown_reset (void *junk, int howto)
- void panic (const char *fmt,...)
- SYSCTL_INT (_kern_shutdown, OID_AUTO, poweroff_delay, CTLFLAG_RW,&poweroff_delay, 0,"")
- static void poweroff_wait (void *junk, int howto)
- SYSCTL_INT (_kern_shutdown, OID_AUTO, kproc_shutdown_wait, CTLFLAG_RW,&kproc_shutdown_wait, 0,"")
- void kproc_shutdown (void *arg, int howto)
- int set_dumper (struct dumperinfo *di)

Variables

- int sync_on_panic = 0
- const char * panicstr
- int dumping
- int rebooting
- static struct dumperinfo dumper
- static struct pcb dumppcb
- static lwpid_t dumptid
- static int shutdown_howto = 0
- static int waittime = -1
- static int poweroff_delay = POWEROFF_DELAY
- static int kproc_shutdown_wait = 60

9.55.1 Define Documentation

9.55.1.1 #define PANIC_REBOOT_WAIT_TIME 15

Definition at line 82 of file kern_shutdown.c.

Referenced by shutdown_panic().

9.55.1.2 #define POWEROFF_DELAY 5000

Definition at line 574 of file kern_shutdown.c.

9.55.2 Function Documentation

9.55.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_shutdown. *c*, v 1.180 2006/11/06 13:42:00 rwatson Exp $")

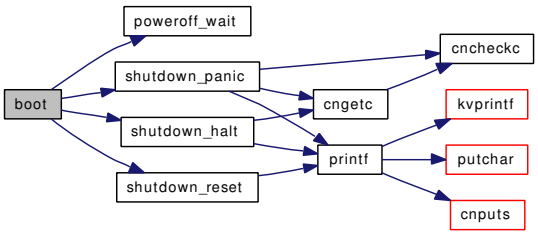
9.55.2.2 static void boot (int *howto*) [static]

Definition at line 129 of file kern_shutdown.c.

References poweroff_wait(), shutdown_halt(), shutdown_panic(), and shutdown_reset().

Referenced by panic(), reboot(), and shutdown_nice().

Here is the call graph for this function:

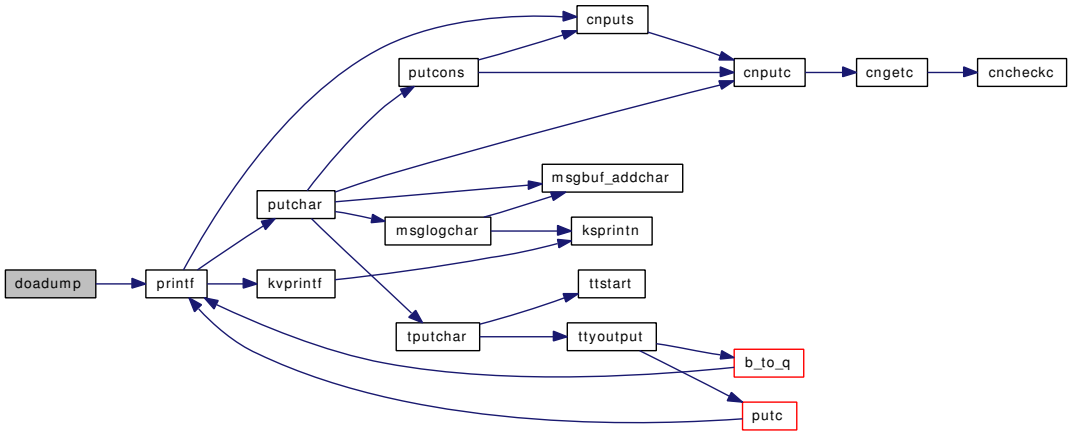


9.55.2.3 static void doadump (void) [static]

Definition at line 229 of file kern_shutdown.c.

References dumper, dumping, dumppcb, dumptid, and printf().

Here is the call graph for this function:



9.55.2.4 static int isbufbusy (struct buf * *bp*) [static]

Definition at line 249 of file kern_shutdown.c.

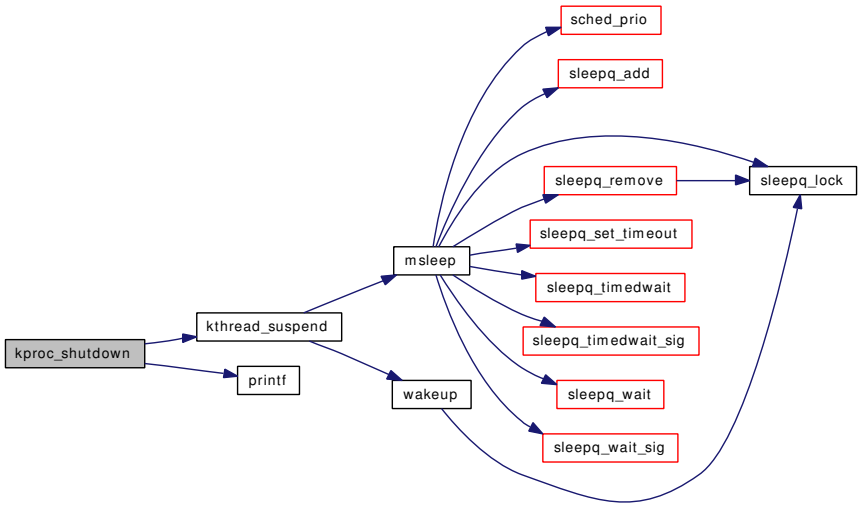
9.55.2.5 void kproc_shutdown (void * *arg*, int *howto*)

Definition at line 602 of file kern_shutdown.c.

References hz, kthread_suspend(), panicstr, and printf().

Referenced by buf_daemon(), syncer_shutdown(), and vnlru_proc().

Here is the call graph for this function:



9.55.2.6 void panic (const char * *fmt*, ...)

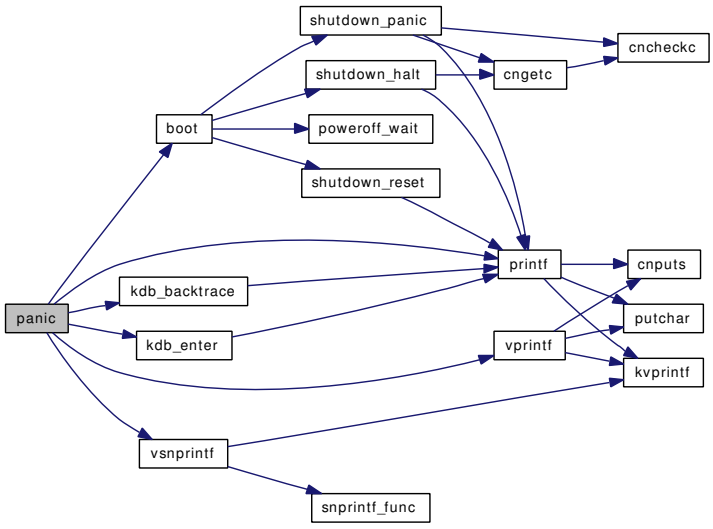
Definition at line 502 of file kern_shutdown.c.

References boot(), buf, kdb_backtrace(), kdb_enter(), panicstr, printf(), sched_lock, sync_on_panic, td, vprintf(), and vsnprintf().

Referenced by _lockmgr(), acl_posix1e_acl_to_mode(), aio_proc_rundown(), aio_swake_cb(), allocbuf(), blst_leaf_free(), blst_meta_alloc(), blst_meta_fill(), blst_meta_free(), bqrelse(), brelse(), brelvp(), bufbdflush(), bufdone_finish(), bufobj_invalbuf(), bufwrite(), cblock_alloc(), clist_free_cblocks(), clone_create(), config_intrhook_disestablish(), copyinfrom(), copyinstrfrom(), create_init(), dev_strategy(), devclass_delete_device(), device_busy(), device_probe_child(), device_unbusy(), enroll(), exec_unregister(), exit1(), fdrop_locked(), filt_mqdetach(), free(), getblk(), getnewbuf(), hashdestroy(), hashinit_flags(), idle_setup(), intr_event_remove_handler(), ithread_create(), itimers_event_hook_exit(), itismychild(), kdb_sysctl_panic(), kern_getsockopt(), kern_mknod(), kern_msgctl(), kern_msgrcv(), kern_setsockopt(), kobj_class_compile(), kproc_start(), kse_wakeup(), kthread_create(), lf_clearlock(), lf_findoverlap(), link_elf_init(), link_elf_reloc_local(), linker_preload(), lookup(), m_pulldown(), mb_ctor_clust(), mb_reserve(), mi_switch(), modlist_newmodule(), module_register_init(), module_release(), mqfs_allocv(), mqfs_create(), mqfs_readdir(), msg_freehdr(), msginit(), net_init_domain(), phashinit(), proc_fini(), propagate_priority(), reassignbuf(), relocate_file(), relookup(), resource_list_add(), resource_list_alloc(), resource_list_delete(), resource_list_free(), resource_list_release(), rman_await_resource(),

rman_init(), root_setup_intr(), sbappendaddr_locked(), sbappendcontrol_locked(), sbdrop_internal(), sbflush_internal(), sched_add(), sched_setup(), semexit_myhook(), semu_alloc(), set_rootvnode(), setrunnable(), shminit(), soopt_mcopyin(), STAILQ_HEAD(), start_init(), start_softintr(), sysctl_ctx_free(), sysctl_register_oid(), sysinit_add(), TAILQ_HEAD(), thread_dtor(), thread_exit(), timeout(), ttyrub(), uipc_attach(), uipc_rcvd(), uipc_send(), umtx_pi_setowner(), unp_connect2(), unp_externalize(), unp_init(), ureadc(), vdropl(), vfs_busy(), vfs_domount(), vfs_freeopt(), vfs_mountroot(), vfs_unbusy_pages(), vget(), vgonel(), vmapbuf(), vn_finished_secondary_write(), vn_finished_write(), vop_panic(), vop_stdfsync(), vput(), vrele(), witness_assert(), witness_checkorder(), witness_destroy(), witness_downgrade(), witness_init(), witness_restore(), witness_save(), witness_unlock(), witness_upgrade(), and witness_warn().

Here is the call graph for this function:



9.55.2.7 static void poweroff_wait (void * *junk*, int *howto*) [static]

Definition at line 582 of file kern_shutdown.c.

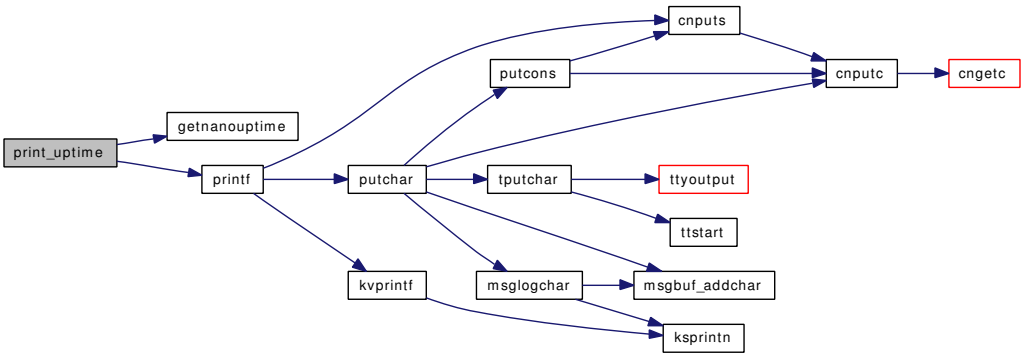
Referenced by boot().

9.55.2.8 static void print_uptime (void) [static]

Definition at line 202 of file kern_shutdown.c.

References getnanouptime(), and printf().

Here is the call graph for this function:

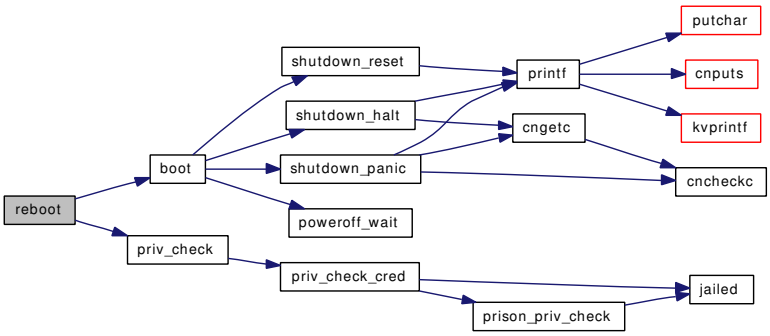


9.55.2.9 int reboot (struct thread * *td*, struct reboot_args * *uap*)

Definition at line 159 of file kern_shutdown.c.

References boot(), Giant, and priv_check().

Here is the call graph for this function:



9.55.2.10 int set_dumper (struct dumperinfo * *di*)

Definition at line 625 of file kern_shutdown.c.

References dumper.

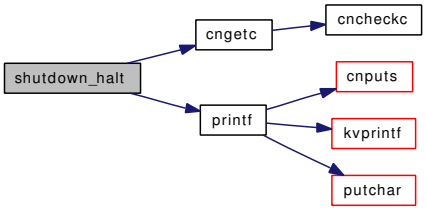
9.55.2.11 static void shutdown_halt (void * *junk*, int *howto*) [static]

Definition at line 424 of file kern_shutdown.c.

References cngetc(), and printf().

Referenced by boot().

Here is the call graph for this function:

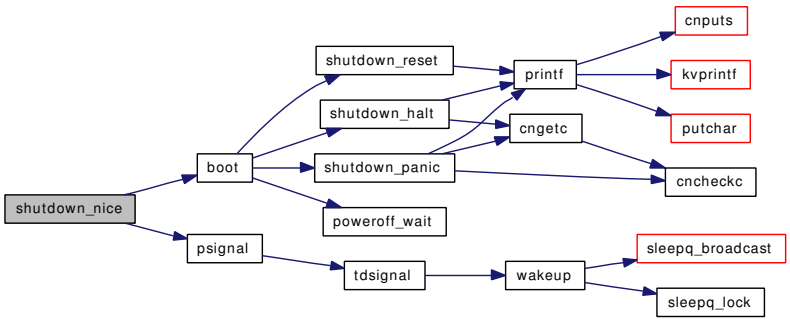


9.55.2.12 void shutdown_nice (int *howto*)

Definition at line 183 of file kern_shutdown.c.

References boot(), initproc, psignal(), and shutdown_howto.

Here is the call graph for this function:



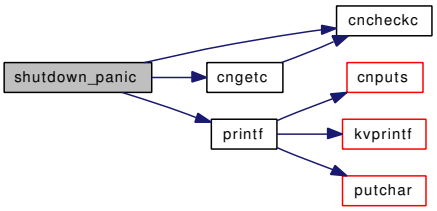
9.55.2.13 static void shutdown_panic (void * *junk*, int *howto*) [static]

Definition at line 447 of file kern_shutdown.c.

References cncheckc(), cngetc(), PANIC_REBOOT_WAIT_TIME, and printf().

Referenced by boot().

Here is the call graph for this function:



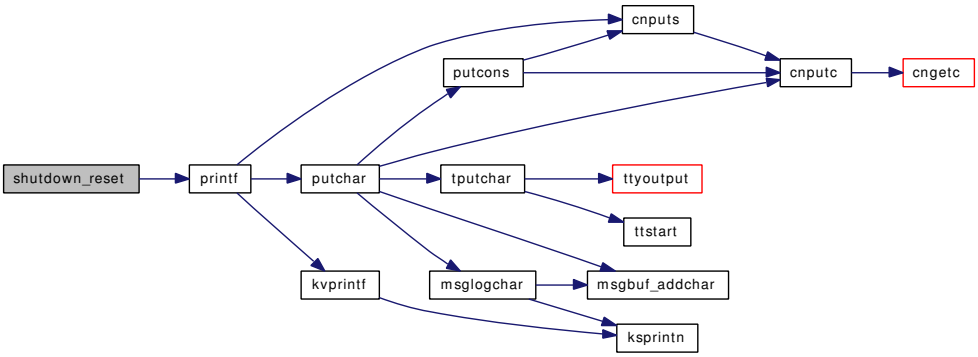
9.55.2.14 static void shutdown_reset (void * *junk*, int *howto*) [static]

Definition at line 480 of file kern_shutdown.c.

References printf().

Referenced by boot().

Here is the call graph for this function:



9.55.2.15 SYSCTL_INT (_kern_shutdown, OID_AUTO, kproc_shutdown_wait, CTLFLAG_RW, & *kproc_shutdown_wait*, 0, "")

9.55.2.16 SYSCTL_INT (_kern_shutdown, OID_AUTO, poweroff_delay, CTLFLAG_RW, & *poweroff_delay*, 0, "")

9.55.2.17 SYSCTL_INT (_kern, OID_AUTO, sync_on_panic, CTLFLAG_RW, & *sync_on_panic*, 0, "Do a sync before rebooting from a panic")

9.55.2.18 SYSCTL_NODE (_kern, OID_AUTO, shutdown, CTLFLAG_RW, 0, "Shutdown environment")

9.55.3 Variable Documentation

9.55.3.1 struct dumperinfo dumper [static]

Definition at line 123 of file kern_shutdown.c.

Referenced by doadump(), and set_dumper().

9.55.3.2 int dumping

Definition at line 121 of file kern_shutdown.c.

Referenced by doadump().

9.55.3.3 struct pcb dumppcb [static]

Definition at line 126 of file kern_shutdown.c.

Referenced by doadump().

9.55.3.4 lwpid_t dumptid [static]

Definition at line 127 of file kern_shutdown.c.

Referenced by doadump().

9.55.3.5 int kproc_shutdown_wait = 60 [static]

Definition at line 597 of file kern_shutdown.c.

9.55.3.6 const char* panicstr

Definition at line 119 of file kern_shutdown.c.

Referenced by _lockmgr(), cv_timedwait(), cv_timedwait_sig(), cv_wait(), cv_wait_sig(), cv_wait_unlock(), enroll(), kproc_shutdown(), ktr_tracepoint(), mi_switch(), panic(), printf(), putchar(), sched_preempt(), vprintf(), witness_assert(), witness_checkorder(), witness_defineorder(), witness_downgrade(), witness_init(), witness_lock(), witness_restore(), witness_save(), witness_unlock(), witness_upgrade(), and witness_warn().

9.55.3.7 int poweroff_delay = POWEROFF_DELAY [static]

Definition at line 576 of file kern_shutdown.c.

9.55.3.8 int rebooting

Definition at line 122 of file kern_shutdown.c.

Referenced by vfs_mount_destroy().

9.55.3.9 int shutdown_howto = 0 [static]

Definition at line 180 of file kern_shutdown.c.

Referenced by shutdown_nice().

9.55.3.10 int sync_on_panic = 0

Definition at line 109 of file kern_shutdown.c.

Referenced by panic().

9.55.3.11 int waittime = -1 [static]

Definition at line 199 of file kern_shutdown.c.

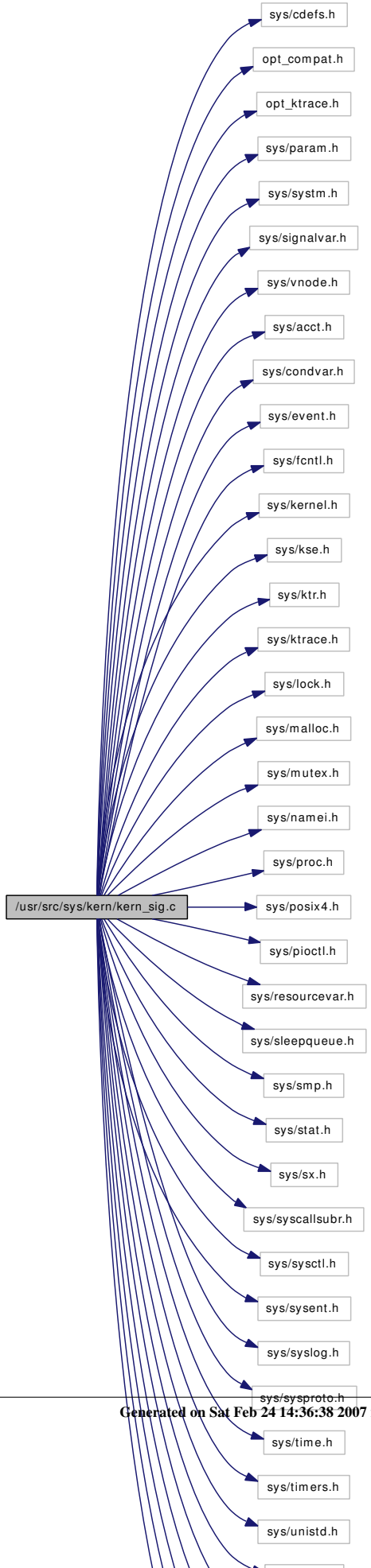
Referenced by _mtx_lock_flags(), _mtx_lock_spin_flags(), _mtx_trylock(), _sx_slock(), and _sx_xlock().

9.56 /usr/src/sys/kern/kern_sig.c File Reference

```
#include <sys/cdefs.h>
#include "opt_compat.h"
#include "opt_ktrace.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/signalvar.h>
#include <sys/vnode.h>
#include <sys/acct.h>
#include <sys/condvar.h>
#include <sys/event.h>
#include <sys/fcntl.h>
#include <sys/kernel.h>
#include <sys/kse.h>
#include <sys/ktr.h>
#include <sys/ktrace.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/namei.h>
#include <sys/proc.h>
#include <sys/posix4.h>
#include <sys/pioctl.h>
#include <sys/resourcevar.h>
#include <sys/sleepqueue.h>
#include <sys/smp.h>
#include <sys/stat.h>
#include <sys/sx.h>
#include <sys/syscallsubr.h>
#include <sys/sysctl.h>
#include <sys/sysent.h>
#include <sys/syslog.h>
#include <sys/sysproto.h>
#include <sys/time.h>
#include <sys/timers.h>
#include <sys/unistd.h>
```

```
#include <sys/wait.h>
#include <vm/vm.h>
#include <vm/vm_extern.h>
#include <vm/uma.h>
#include <machine/cpu.h>
#include <security/audit/audit.h>
```

Include dependency graph for kern_sig.c:


/usr/src/sys/kern/kern_sig.c
sys/cdefs.h
opt_compat.h
opt_ktrace.h
sys/param.h
sys/systm.h
sys/signalvar.h
sys/vnode.h
sys/acct.h
sys/condvar.h
sys/event.h
sys/fcntl.h
sys/kernel.h
sys/kse.h
sys/ktr.h
sys/ktrace.h
sys/lock.h
sys/malloc.h
sys/mutex.h
sys/namei.h
sys/proc.h
sys/posix4.h
sys/pioctl.h
sys/resourcevar.h
sys/sleepqueue.h
sys/smp.h
sys/stat.h
sys/sx.h
sys/syscallsubr.h
sys/sysctl.h
sys/sysent.h
sys/syslog.h
sys/sysproto.h
sys/time.h
sys/timers.h
sys/unistd.h

Data Structures

- struct sigaction_args
- struct sigprocmask_args
- struct sigpending_args
- struct sigsuspend_args
- struct sigaltstack_args
- struct kill_args
- struct sigqueue_args
- struct nosys_args

Defines

- #define ONSIG 32
- #define CANSIGIO(cr1, cr2)
- #define SA_KILL 0x01
- #define SA_CORE 0x02
- #define SA_STOP 0x04
- #define SA_TTYSTOP 0x08
- #define SA_IGNORE 0x10
- #define SA_CONT 0x20
- #define SA_CANTMASK 0x40
- #define SA_PROC 0x80

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_sig.c,v 1.339 2007/02/09 17:48:28 delphij Exp $")
- static int coredump (struct thread *)
- static char * expand_name (const char *, uid_t, pid_t)
- static int killpg1 (struct thread *td, int sig, int pgid, int all)
- static int issignal (struct thread *p)
- static int sigprop (int sig)
- static void tdsigwakeup (struct thread *, int, sig_t, int)
- static void sig_suspend_threads (struct thread *, struct proc *, int)
- static int filt_sigattach (struct knote *kn)
- static void filt_sigdetach (struct knote *kn)
- static int filt_signal (struct knote *kn, long hint)
- static struct thread * sigtd (struct proc *p, int sig, int prop)
- static void sigqueue_start (void)
- SYSCTL_INT (_kern, KERN_LOGSIGEXIT, logsigexit, CTLFLAG_RW,&kern_logsigexit, 0,"Log processes quitting on abnormal signals to syslog(3)")
- SYSCTL_INT (_kern, OID_AUTO, forcesigexit, CTLFLAG_RW,&kern_forcesigexit, 0,"Force trap signal to be handled")
- SYSCTL_NODE (_kern, OID_AUTO, sigqueue, CTLFLAG_RW, 0,"POSIX real time signal")
- SYSCTL_INT (_kern_sigqueue, OID_AUTO, max_pending_per_proc, CTLFLAG_RW,&max_pending_per_proc, 0,"Max pending signals per proc")
- TUNABLE_INT ("kern.sigqueue.preallocate",&preallocate_siginfo)
- SYSCTL_INT (_kern_sigqueue, OID_AUTO, preallocate, CTLFLAG_RD,&preallocate_siginfo, 0,"Preallocated signal memory size")

- SYSCTL_INT (_kern_sigqueue, OID_AUTO, overflow, CTLFLAG_RD,&signal_overflow, 0,"Number of signals overflew")
- SYSCTL_INT (_kern_sigqueue, OID_AUTO, alloc_fail, CTLFLAG_RD,&signal_alloc_fail, 0,"signals failed to be allocated")
- SYSINIT (signal, SI_SUB_P1003_1B, SI_ORDER_FIRST+3, sigqueue_start, NULL)
- SYSCTL_INT (_kern, OID_AUTO, sugid_coredump, CTLFLAG_RW,&sugid_coredump, 0,"Enable coredumping set user/group ID processes")
- SYSCTL_INT (_kern, OID_AUTO, coredump, CTLFLAG_RW,&do_coredump, 0,"Enable/Disable coredumps")
- SYSCTL_INT (_kern, OID_AUTO, nodump_coredump, CTLFLAG_RW,&set_core_nodump_flag, 0,"Enable setting the NODUMP flag on coredump files")
- ksiginfo_t * ksiginfo_alloc (int wait)
- void ksiginfo_free (ksiginfo_t *ksi)
- static __inline int ksiginfo_tryfree (ksiginfo_t *ksi)
- void sigqueue_init (sigqueue_t *list, struct proc *p)
- int sigqueue_get (sigqueue_t *sq, int signo, ksiginfo_t *si)
- void sigqueue_take (ksiginfo_t *ksi)
- int sigqueue_add (sigqueue_t *sq, int signo, ksiginfo_t *si)
- void sigqueue_flush (sigqueue_t *sq)
- void sigqueue_collect_set (sigqueue_t *sq, sigset_t *set)
- void sigqueue_move_set (sigqueue_t *src, sigqueue_t *dst, sigset_t *setp)
- void sigqueue_move (sigqueue_t *src, sigqueue_t *dst, int signo)
- void sigqueue_delete_set (sigqueue_t *sq, sigset_t *set)
- void sigqueue_delete (sigqueue_t *sq, int signo)
- void sigqueue_delete_set_proc (struct proc *p, sigset_t *set)
- void sigqueue_delete_proc (struct proc *p, int signo)
- void sigqueue_delete_stopmask_proc (struct proc *p)
- int cursig (struct thread *td)
- void signotify (struct thread *td)
- int sigonstack (size_t sp)
- int sig_ffs (sigset_t *set)
- int kern_sigaction (struct thread *td, int sig, struct sigaction *act, struct sigaction *oact, int flags)
- int sigaction (struct thread *td, struct sigaction_args *uap)
- void siginit (struct proc *p)
- void execsigs (struct proc *p)
- int kern_sigprocmask (struct thread *td, int how, sigset_t *set, sigset_t *oset, int old)
- int sigprocmask (struct thread *td, struct sigprocmask_args *uap)
- int sigwait (struct thread *td, struct sigwait_args *uap)
- int sigtimedwait (struct thread *td, struct sigtimedwait_args *uap)
- int sigwaitinfo (struct thread *td, struct sigwaitinfo_args *uap)
- int kern_sigtimedwait (struct thread *td, sigset_t waitset, ksiginfo_t *ksi, struct timespec *timeout)
- int sigpending (struct thread *td, struct sigpending_args *uap)
- int sigsuspend (struct thread *td, struct sigsuspend_args *uap)
- int kern_sigsuspend (struct thread *td, sigset_t mask)
- int sigaltstack (struct thread *td, struct sigaltstack_args *uap)
- int kern_sigaltstack (struct thread *td, stack_t *ss, stack_t *oss)
- int kill (struct thread *td, struct kill_args *uap)
- int sigqueue (struct thread *td, struct sigqueue_args *uap)
- void gsignal (int pgid, int sig)
- void pgsignal (struct pgrp *pgrp, int sig, int checkctty)

- void trapsignal (struct thread *td, ksiginfo_t *ksi)
- void psignal (struct proc *p, int sig)
- int psignal_event (struct proc *p, struct sigevent *sigev, ksiginfo_t *ksi)
- int tdsignal (struct proc *p, struct thread *td, int sig, ksiginfo_t *ksi)
- int ptracestop (struct thread *td, int sig)
- void thread_stopped (struct proc *p)
- void postsig (int sig)
- void killproc (struct proc *p, char *why)
- void sigexit (struct thread *td, int sig)
- static void sigparent (struct proc *p, int reason, int status)
- static void childproc_jobstate (struct proc *p, int reason, int status)
- void childproc_stopped (struct proc *p, int reason)
- void childproc_continued (struct proc *p)
- void childproc_exited (struct proc *p)
- SYSCTL_STRING (_kern, OID_AUTO, corefile, CTLFLAG_RW, corefilename, sizeof(corefilename),"process corefile name format string")
- static char * expand_name (char *name, uid_t uid, pid_t pid) const
- int nosys (struct thread *td, struct nosys_args *args)
- void pgsigio (struct sigio **sigiop, int sig, int checkctty)
- sigacts * sigacts_alloc (void)
- void sigacts_free (struct sigacts *ps)
- sigacts * sigacts_hold (struct sigacts *ps)
- void sigacts_copy (struct sigacts *dest, struct sigacts *src)
- int sigacts_shared (struct sigacts *ps)

Variables

- static uma_zone_t ksiginfo_zone = NULL
- filterops sig_filtops
- static int kern_logsigexit = 1
- static int kern_forcesigexit = 1
- static int max_pending_per_proc = 128
- static int preallocate_siginfo = 1024
- static int signal_overflow = 0
- static int signal_alloc_fail = 0
- int sugid_coredump
- static int do_coredump = 1
- static int set_core_nodump_flag = 0
- static int sigproptbl [NSIG]
- static char corefilename [MAXPATHLEN] = {"%N.core"}

9.56.1 Define Documentation

9.56.1.1 #define CANSIGIO(cr1, cr2)

Value:

```
((cr1)->cr_uid == 0 || \
            (cr1)->cr_ruid == (cr2)->cr_ruid || \
            (cr1)->cr_uid == (cr2)->cr_ruid || \
            (cr1)->cr_ruid == (cr2)->cr_uid || \
            (cr1)->cr_uid == (cr2)->cr_uid)
```

Definition at line 141 of file kern_sig.c.

Referenced by pgsigio().

9.56.1.2 #define ONSIG 32

Definition at line 84 of file kern_sig.c.

9.56.1.3 #define SA_CANTMASK 0x40

Definition at line 171 of file kern_sig.c.

9.56.1.4 #define SA_CONT 0x20

Definition at line 170 of file kern_sig.c.

9.56.1.5 #define SA_CORE 0x02

Definition at line 166 of file kern_sig.c.

Referenced by sigexit().

9.56.1.6 #define SA_IGNORE 0x10

Definition at line 169 of file kern_sig.c.

Referenced by execsigs(), issignal(), kern_sigaction(), postsig(), siginit(), and trapsignal().

9.56.1.7 #define SA_KILL 0x01

Definition at line 165 of file kern_sig.c.

Referenced by tdsigwakeup().

9.56.1.8 #define SA_PROC 0x80

Definition at line 172 of file kern_sig.c.

9.56.1.9 #define SA_STOP 0x04

Definition at line 167 of file kern_sig.c.

9.56.1.10 #define SA_TTYSTOP 0x08

Definition at line 168 of file kern_sig.c.

9.56.2 Function Documentation

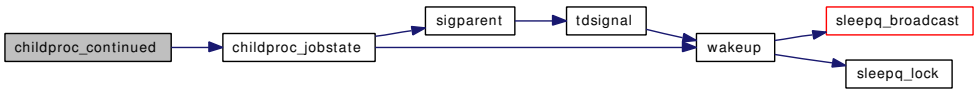
9.56.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_sig. *c*, v 1.339 2007/02/09 17:48:28 delphij Exp $")

9.56.2.2 void childproc_continued (struct proc * *p*)

Definition at line 3001 of file kern_sig.c.

References childproc_jobstate().

Here is the call graph for this function:



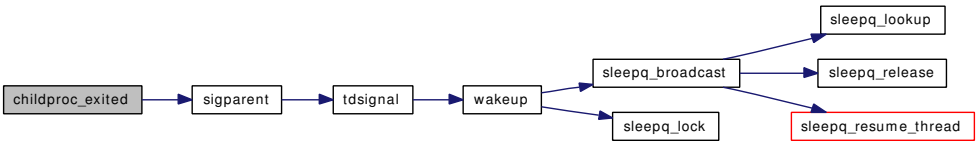
9.56.2.3 void childproc_exited (struct proc * *p*)

Definition at line 3007 of file kern_sig.c.

References sigparent().

Referenced by exit1().

Here is the call graph for this function:



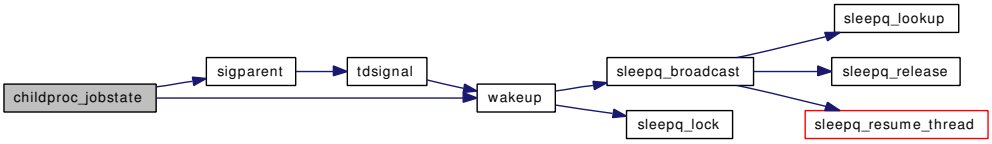
9.56.2.4 static void childproc_jobstate (struct proc * *p*, int *reason*, int *status*) [static]

Definition at line 2969 of file kern_sig.c.

References sigparent(), and wakeup().

Referenced by childproc_continued(), and childproc_stopped().

Here is the call graph for this function:



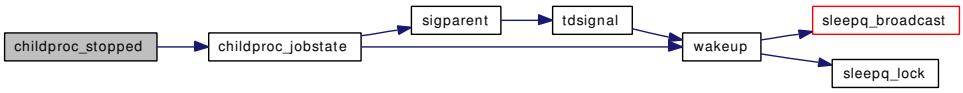
9.56.2.5 void childproc_stopped (struct proc * *p*, int *reason*)

Definition at line 2995 of file kern_sig.c.

References childproc_jobstate().

Referenced by thread_stopped().

Here is the call graph for this function:

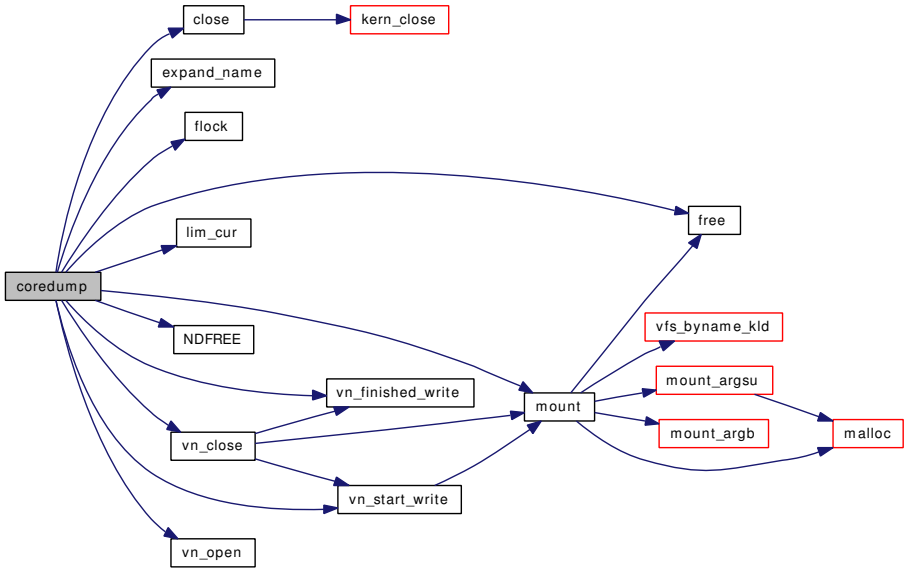


9.56.2.6 static int coredump (struct thread *) [static]

Definition at line 3109 of file kern_sig.c.

References close(), do_coredump, expand_name(), flock(), free(), lim_cur(), mount(), NDFREE(), set_-core_nodump_flag, sugid_coredump, vn_close(), vn_finished_write(), vn_open(), and vn_start_write().

Here is the call graph for this function:



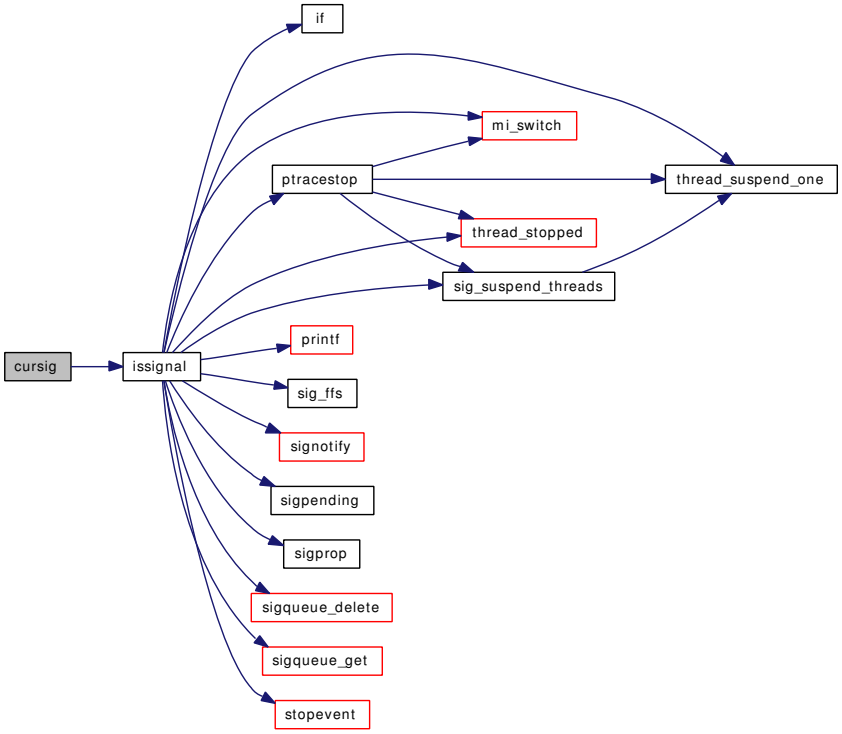
9.56.2.7 int cursig (struct thread * *td*)

Definition at line 554 of file kern_sig.c.

References issignal(), and sched_lock.

Referenced by ast(), and sleepq_catch_signals().

Here is the call graph for this function:

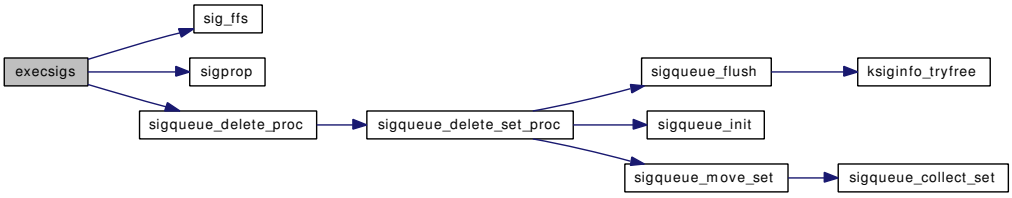


9.56.2.8 void execsigs (struct proc * *p*)

Definition at line 950 of file kern_sig.c.

References SA_IGNORE, sig_ffs(), sigprop(), and sigqueue_delete_proc().

Here is the call graph for this function:

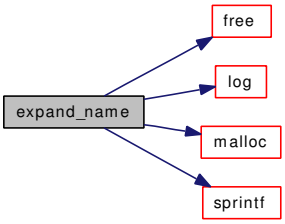


9.56.2.9 static char* expand_name (char * *name*, uid_t *uid*, pid_t *pid*) const [static]

Definition at line 3041 of file kern_sig.c.

References buf, free(), log(), malloc(), and sprintf().

Here is the call graph for this function:



9.56.2.10 static char* expand_name (const char *, uid_t, pid_t) [static]

Referenced by coredump().

9.56.2.11 static int filt_sigattach (struct knote * *kn*) [static]

Definition at line 3281 of file kern_sig.c.

References knlist_add().

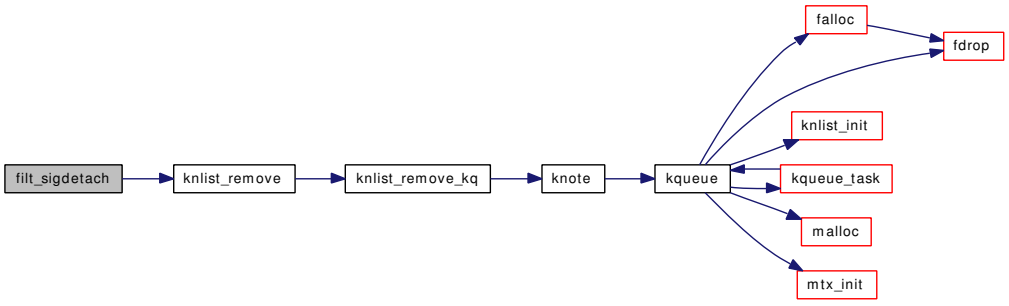
Here is the call graph for this function:

9.56.2.12 static void filt_sigdetach (struct knote * *kn*) [static]

Definition at line 3294 of file kern_sig.c.

References knlist_remove().

Here is the call graph for this function:



9.56.2.13 static int filt_signal (struct knote * *kn*, long *hint*) [static]

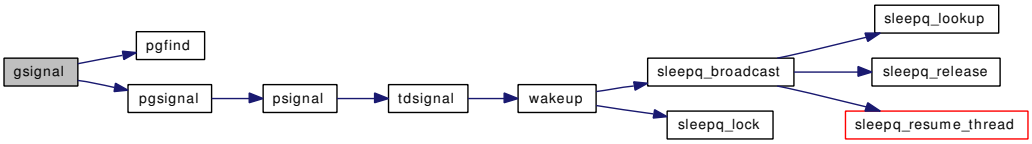
Definition at line 3308 of file kern_sig.c.

9.56.2.14 void gsignal (int *pgid*, int *sig*)

Definition at line 1850 of file kern_sig.c.

References pgfind(), pgsignal(), and proctree_lock.

Here is the call graph for this function:



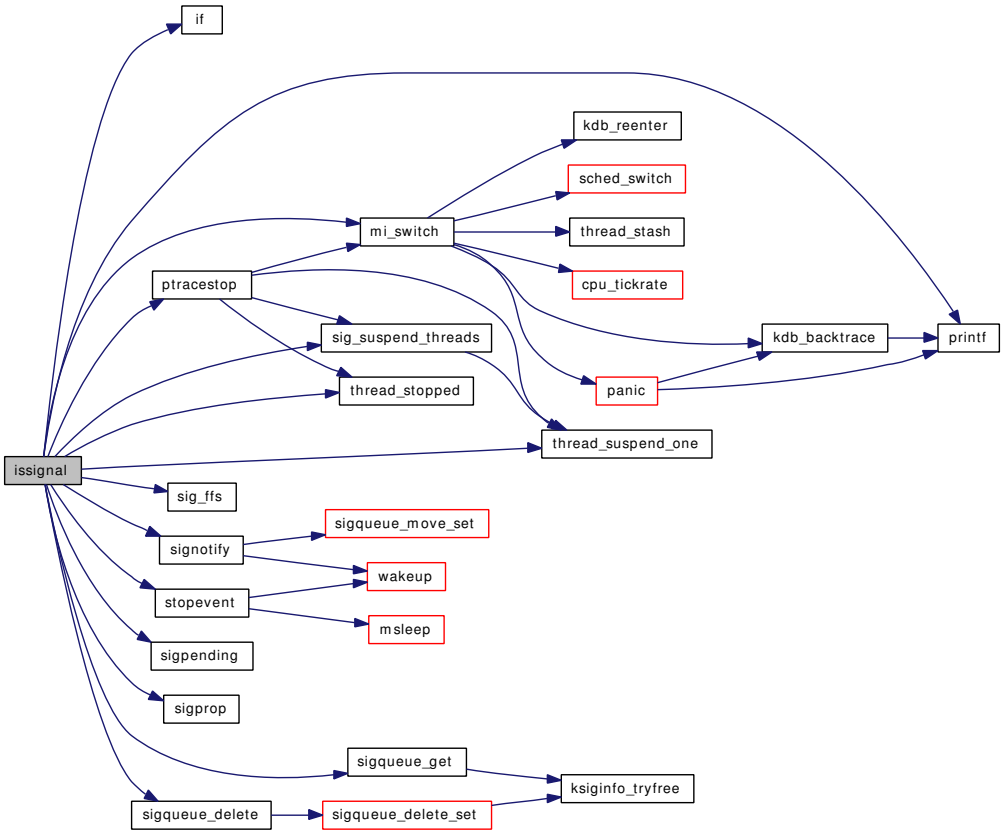
9.56.2.15 static int issignal (struct thread * *p*) [static]

Definition at line 2553 of file kern_sig.c.

References if(), mi_switch(), printf(), ptracestop(), SA_IGNORE, sched_lock, sig_ffs(), sig_suspend_threads(), signotify(), sigpending(), sigprop(), sigqueue_delete(), sigqueue_get(), stopevent(), thread_stopped(), and thread_suspend_one().

Referenced by cursig().

Here is the call graph for this function:



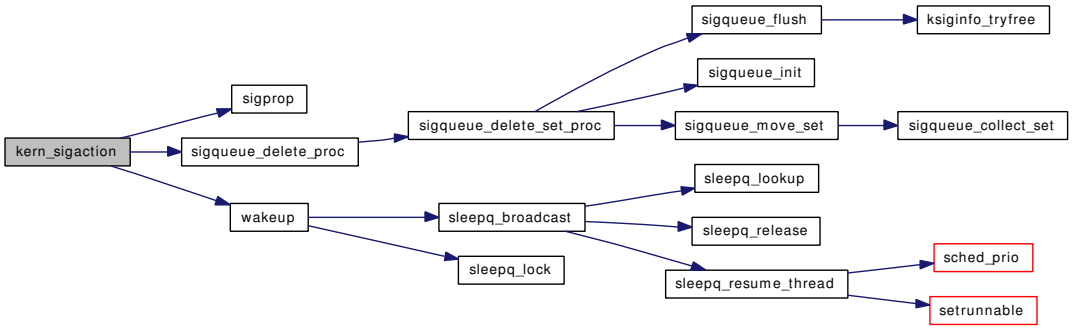
9.56.2.16 int kern_sigaction (struct thread * *td*, int *sig*, struct sigaction * *act*, struct sigaction * *oact*, int *flags*)

Definition at line 654 of file kern_sig.c.

References SA_IGNORE, sigprop(), sigqueue_delete_proc(), and wakeup().

Referenced by sigaction().

Here is the call graph for this function:



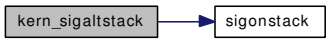
9.56.2.17 int kern_sigaltstack (struct thread * *td*, stack_t * *ss*, stack_t * *oss*)

Definition at line 1631 of file kern_sig.c.

References sigonstack().

Referenced by sigaltstack().

Here is the call graph for this function:



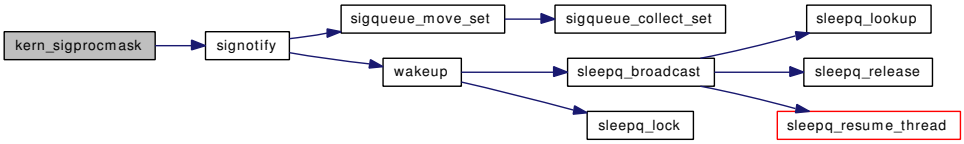
9.56.2.18 int kern_sigprocmask (struct thread * *td*, int *how*, sigset_t * *set*, sigset_t * *oset*, int *old*)

Definition at line 998 of file kern_sig.c.

References signotify().

Referenced by kse_thr_interrupt(), and sigprocmask().

Here is the call graph for this function:



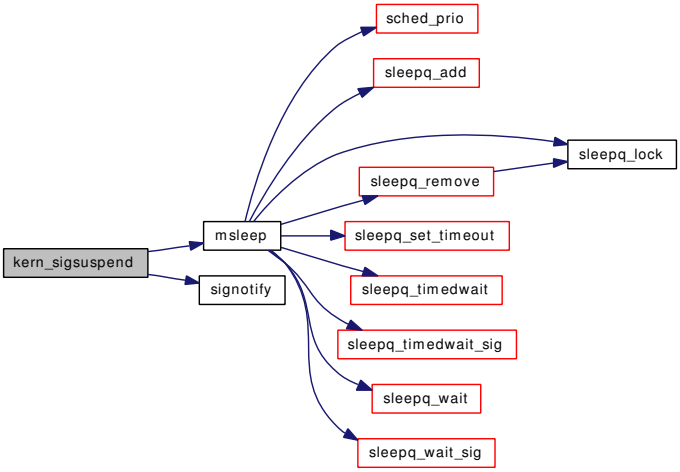
9.56.2.19 int kern_sigsuspend (struct thread * *td*, sigset_t *mask*)

Definition at line 1498 of file kern_sig.c.

References msleep(), and signotify().

Referenced by sigsuspend().

Here is the call graph for this function:



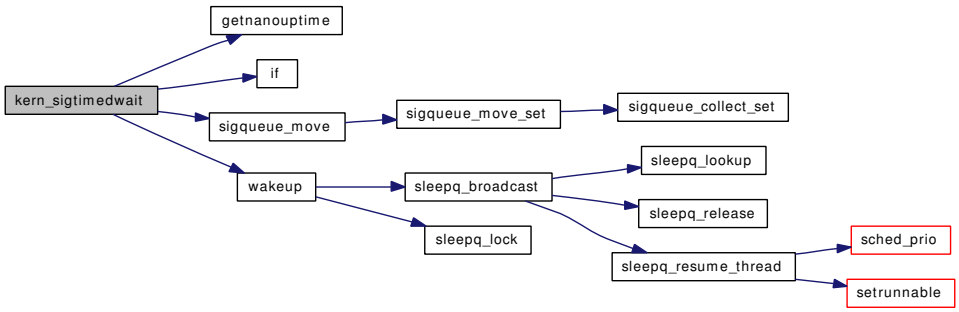
9.56.2.20 int kern_sigtimedwait (struct thread * *td*, sigset_t *waitset*, ksiginfo_t * *ksi*, struct timespec * *timeout*)

Definition at line 1189 of file kern_sig.c.

References getnanouptime(), hz, if(), sigqueue_move(), and wakeup().

Referenced by sigtimedwait(), sigwait(), and sigwaitinfo().

Here is the call graph for this function:

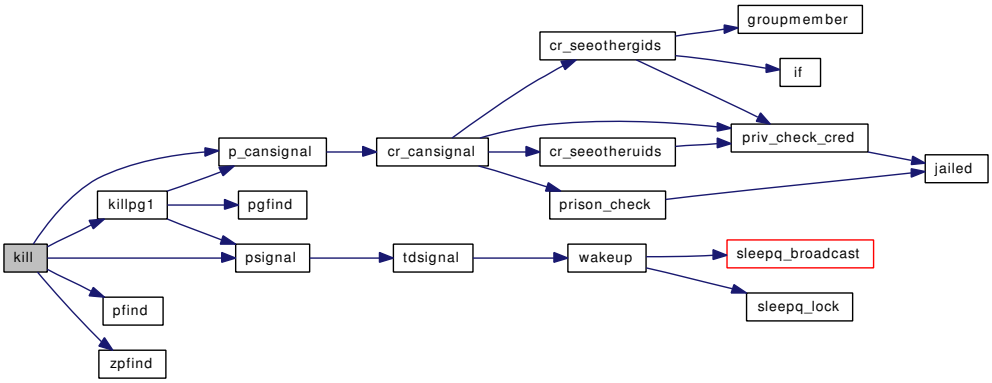


9.56.2.21 int kill (struct thread * *td*, struct kill_args * *uap*)

Definition at line 1741 of file kern_sig.c.

References killpg1(), p_cansignal(), pfind(), psignal(), and zpfind().

Here is the call graph for this function:



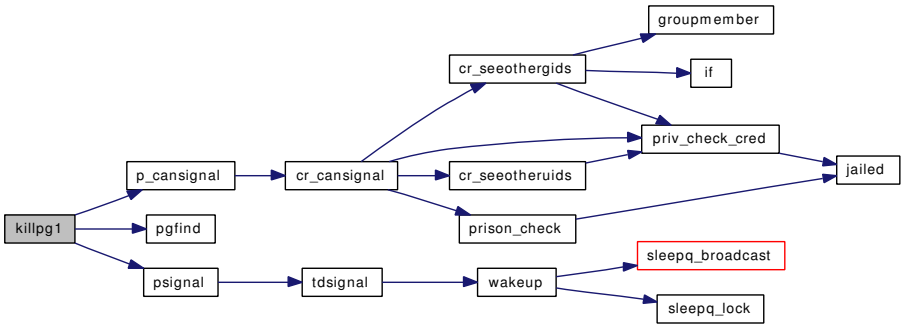
9.56.2.22 static int killpg1 (struct thread * *td*, int *sig*, int *pgid*, int *all*) [static]

Definition at line 1667 of file kern_sig.c.

References allproc_lock, p_cansignal(), pgfind(), proctree_lock, and psignal().

Referenced by kill().

Here is the call graph for this function:



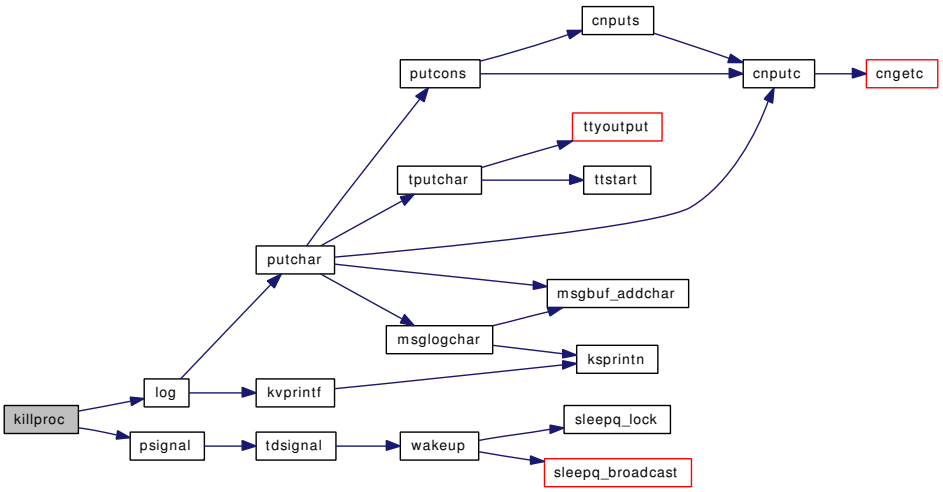
9.56.2.23 void killproc (struct proc * *p*, char * *why*)

Definition at line 2876 of file kern_sig.c.

References log(), and psignal().

Referenced by ast().

Here is the call graph for this function:



9.56.2.24 ksiginfo_t* ksiginfo_alloc (int *wait*)

Definition at line 220 of file kern_sig.c.

References ksiginfo_zone.

Referenced by proc_linkup(), and sigqueue_add().

9.56.2.25 void ksiginfo_free (ksiginfo_t * *ksi*)

Definition at line 233 of file kern_sig.c.

References ksiginfo_zone.

Referenced by proc_fini().

9.56.2.26 static __inline int ksiginfo_tryfree (ksiginfo_t * *ksi*) [static]

Definition at line 239 of file kern_sig.c.

References ksiginfo_zone.

Referenced by sigqueue_delete_set(), sigqueue_flush(), and sigqueue_get().

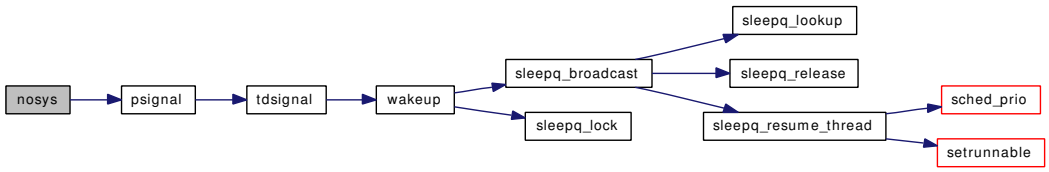
9.56.2.27 int nosys (struct thread * *td*, struct nosys_args * *args*)

Definition at line 3230 of file kern_sig.c.

References psignal().

Referenced by lkmnosys(), lkmressys(), and shmsys().

Here is the call graph for this function:



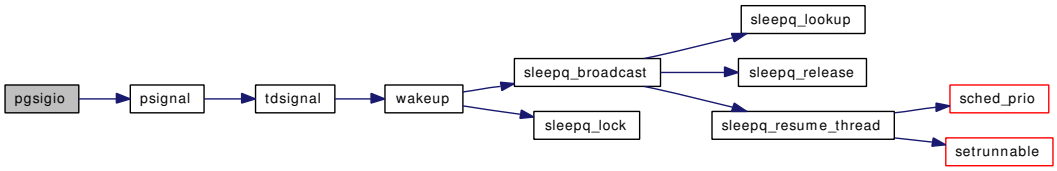
9.56.2.28 void pgsigio (struct sigio ∗∗ *sigiop*, int *sig*, int *checkctty*)

Definition at line 3247 of file kern_sig.c.

References CANSIGIO, and psignal().

Referenced by kqueue_wakeup(), logtimeout(), pipeselwakeup(), sohasoutofband(), sowakeup(), ttwakeup(), and ttwwakeup().

Here is the call graph for this function:



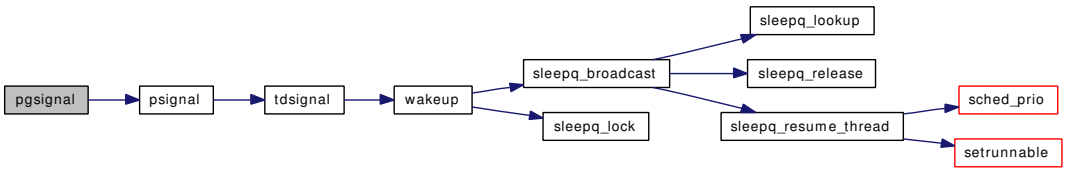
9.56.2.29 void pgsignal (struct pgrp ∗ *pgrp*, int *sig*, int *checkctty*)

Definition at line 1871 of file kern_sig.c.

References psignal().

Referenced by exit1(), gsignal(), ptcioctl(), ttioctl(), ttread(), ttwrite(), and ttyinput().

Here is the call graph for this function:



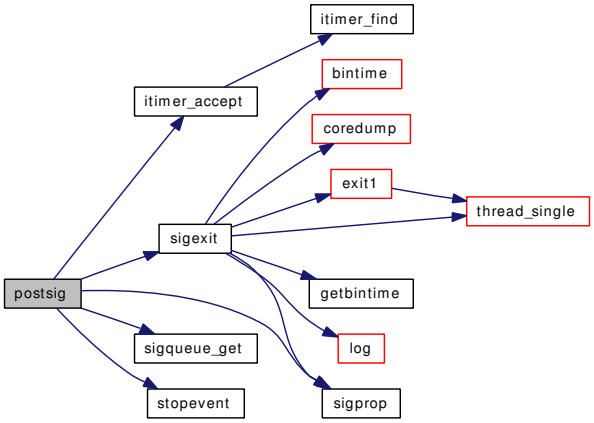
9.56.2.30 void postsig (int *sig*)

Definition at line 2764 of file kern_sig.c.

References itimer_accept(), SA_IGNORE, sigexit(), sigprop(), sigqueue_get(), and stopevent().

Referenced by ast().

Here is the call graph for this function:



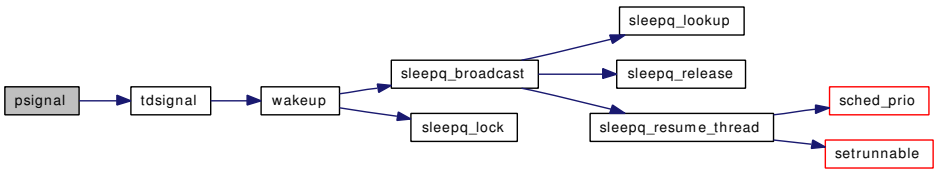
9.56.2.31 void psignal (struct proc * *p*, int *sig*)

Definition at line 2049 of file kern_sig.c.

References tdsignal().

Referenced by aio_process(), ast(), devctl_queue_data(), dofilewrite(), exit1(), fork1(), kern_ptrace(), kill(), killpg1(), killproc(), kse_exit(), nosys(), orphanpg(), pgsigio(), pgsignal(), realitexpire(), sctp_generic_sendmsg(), shutdown_nice(), soo_write(), and ttymodem().

Here is the call graph for this function:



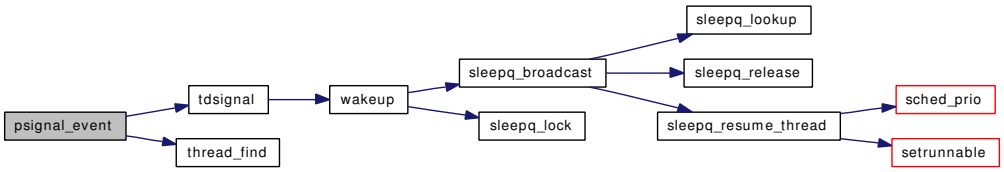
9.56.2.32 int psignal_event (struct proc * *p*, struct sigevent * *sigev*, ksiginfo_t * *ksi*)

Definition at line 2055 of file kern_sig.c.

References tdsignal(), and thread_find().

Referenced by aio_sendsig(), itimer_fire(), and mqueue_send_notification().

Here is the call graph for this function:



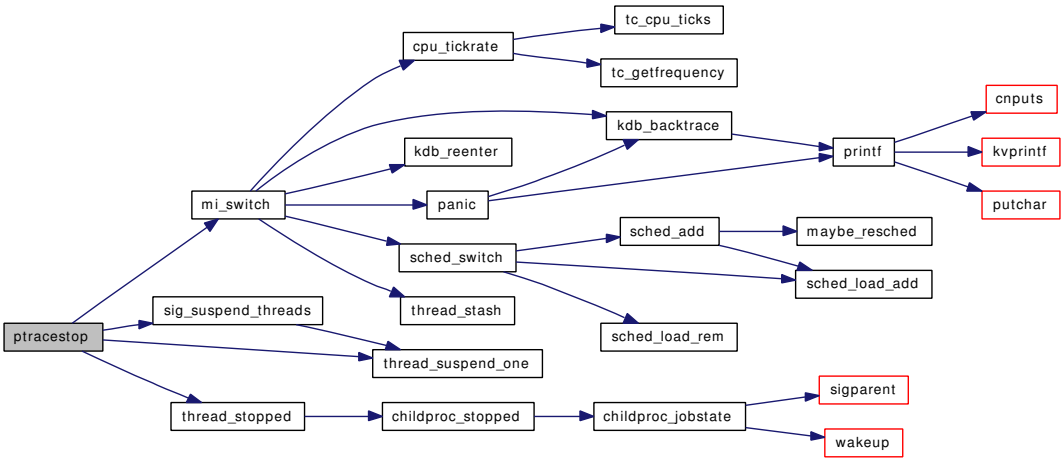
9.56.2.33 int ptracestop (struct thread * *td*, int *sig*)

Definition at line 2491 of file kern_sig.c.

References mi_switch(), sched_lock, sig_suspend_threads(), thread_stopped(), and thread_suspend_one().

Referenced by issignal().

Here is the call graph for this function:



9.56.2.34 int sig_ffs (sigset_t * *set*)

Definition at line 635 of file kern_sig.c.

Referenced by execsigs(), and issignal().

9.56.2.35 static void sig_suspend_threads (struct thread *, struct proc *, int) [static]

Definition at line 2467 of file kern_sig.c.

References sched_lock, and thread_suspend_one().

Referenced by issignal(), and ptracestop().

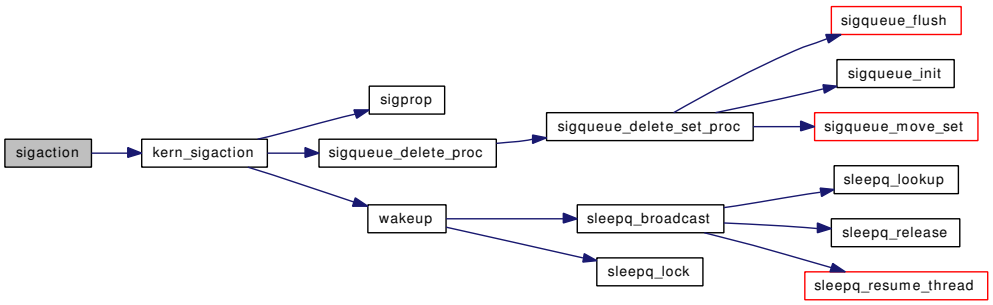
Here is the call graph for this function:

9.56.2.36 int sigaction (struct thread * *td*, struct sigaction_args * *uap*)

Definition at line 811 of file kern_sig.c.

References kern_sigaction().

Here is the call graph for this function:



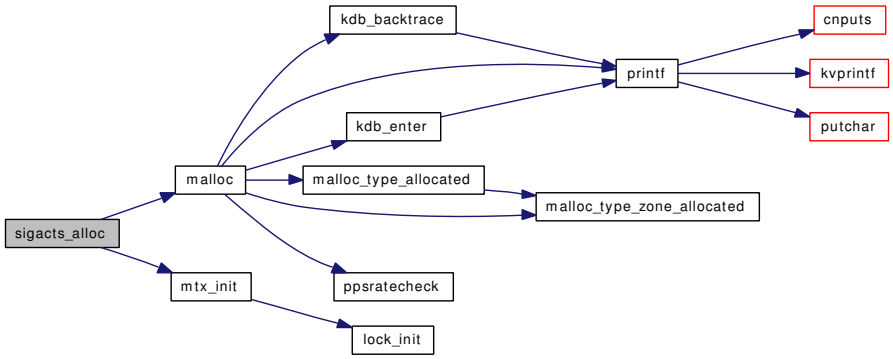
9.56.2.37 struct sigacts* sigacts_alloc (void)

Definition at line 3321 of file kern_sig.c.

References malloc(), and mtx_init().

Referenced by fork1(), and proc0_init().

Here is the call graph for this function:



9.56.2.38 void sigacts_copy (struct sigacts * *dest*, struct sigacts * *src*)

Definition at line 3354 of file kern_sig.c.

Referenced by fork1().

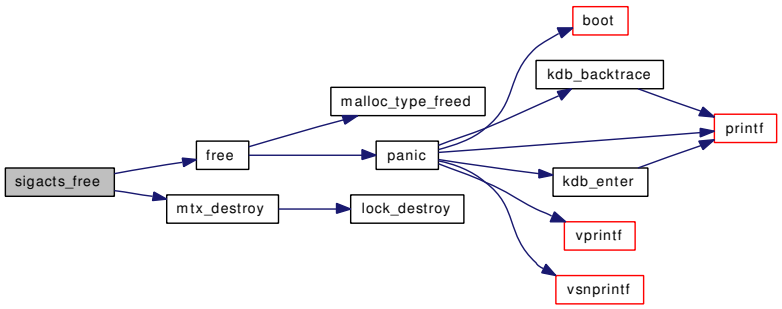
9.56.2.39 void sigacts_free (struct sigacts * *ps*)

Definition at line 3332 of file kern_sig.c.

References free(), and mtx_destroy().

Referenced by kern_wait().

Here is the call graph for this function:



9.56.2.40 struct sigacts* sigacts_hold (struct sigacts * *ps*)

Definition at line 3345 of file kern_sig.c.

Referenced by fork1().

9.56.2.41 int sigacts_shared (struct sigacts * *ps*)

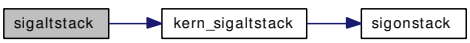
Definition at line 3364 of file kern_sig.c.

9.56.2.42 int sigaltstack (struct thread * *td*, struct sigaltstack_args * *uap*)

Definition at line 1609 of file kern_sig.c.

References kern_sigaltstack().

Here is the call graph for this function:



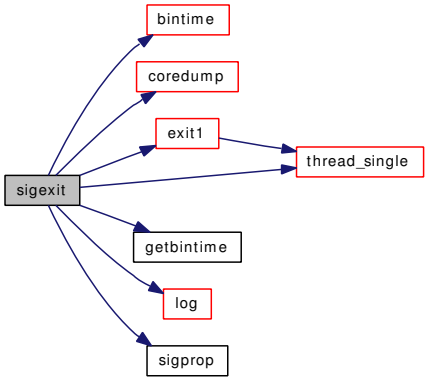
9.56.2.43 void sigexit (struct thread * *td*, int *sig*)

Definition at line 2900 of file kern_sig.c.

References bintime(), coredump(), exit1(), getbintime(), kern_logsigexit, log(), SA_CORE, sigprop(), and thread_single().

Referenced by kse_release(), kse_thr_interrupt(), postsig(), and trapsignal().

Here is the call graph for this function:



9.56.2.44 void siginit (struct proc * *p*)

Definition at line 930 of file kern_sig.c.

References SA_IGNORE, and sigprop().

Referenced by proc0_init().

Here is the call graph for this function:

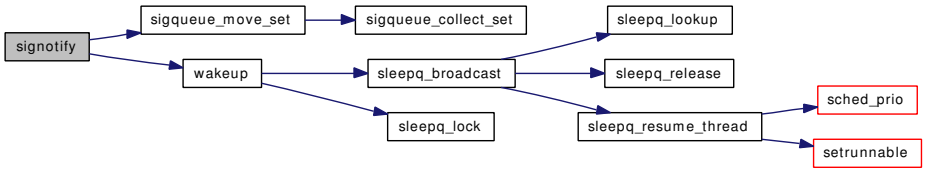
9.56.2.45 void signotify (struct thread * *td*)

Definition at line 568 of file kern_sig.c.

References sched_lock, sigqueue_move_set(), and wakeup().

Referenced by issignal(), kern_sigprocmask(), and kern_sigsuspend().

Here is the call graph for this function:



9.56.2.46 int sigonstack (size_t *sp*)

Definition at line 610 of file kern_sig.c.

References td.

Referenced by kern_sigaltstack().

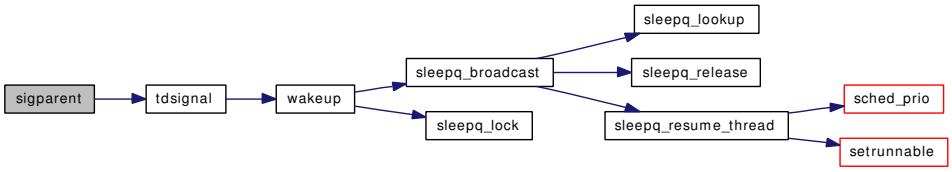
9.56.2.47 static void sigparent (struct proc * *p*, int *reason*, int *status*) [static]

Definition at line 2951 of file kern_sig.c.

References tdsignal().

Referenced by childproc_exited(), and childproc_jobstate().

Here is the call graph for this function:



9.56.2.48 int sigpending (struct thread * *td*, struct sigpending_args * *uap*)

Definition at line 1328 of file kern_sig.c.

References sigpending_args::set.

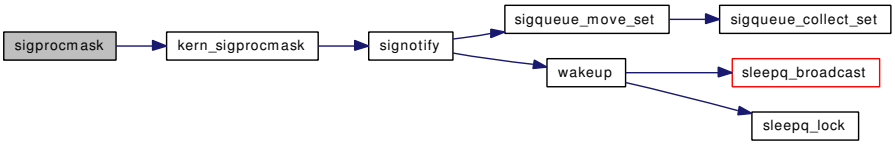
Referenced by issignal().

9.56.2.49 int sigprocmask (struct thread * *td*, struct sigprocmask_args * *uap*)

Definition at line 1050 of file kern_sig.c.

References sigprocmask_args::how, kern_sigprocmask(), sigprocmask_args::oset, and sigprocmask_-args::set.

Here is the call graph for this function:



9.56.2.50 static __inline int sigprop (int *sig*) [static]

Definition at line 626 of file kern_sig.c.

References sigproptbl.

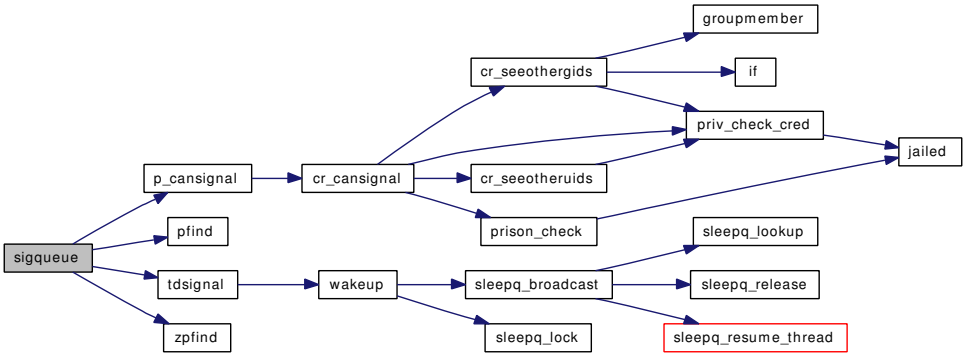
Referenced by execsigs(), issignal(), kern_sigaction(), postsig(), sigexit(), siginit(), tdsigwakeup(), and trapsignal().

9.56.2.51 int sigqueue (struct thread * *td*, struct sigqueue_args * *uap*)

Definition at line 1812 of file kern_sig.c.

References p_cansignal(), pfind(), sigqueue_args::pid, sigqueue_args::signum, tdsignal(), sigqueue_-args::value, and zpfind().

Here is the call graph for this function:



9.56.2.52 int sigqueue_add (sigqueue_t * *sq*, int *signo*, ksiginfo_t * *si*)

Definition at line 327 of file kern_sig.c.

References ksiginfo_alloc(), ksiginfo_zone, max_pending_per_proc, ret, signal_alloc_fail, and signal_-overflow.

Here is the call graph for this function:

9.56.2.53 void sigqueue_collect_set (sigqueue_t * *sq*, sigset_t * *set*)

Definition at line 405 of file kern_sig.c.

Referenced by sigqueue_delete_set(), and sigqueue_move_set().

9.56.2.54 void sigqueue_delete (sigqueue_t * *sq*, int *signo*)

Definition at line 494 of file kern_sig.c.

References sigqueue_delete_set().

Referenced by issignal(), and tdsigwakeup().

Here is the call graph for this function:

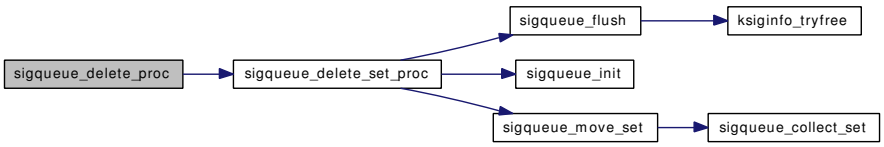
9.56.2.55 void sigqueue_delete_proc (struct proc * *p*, int *signo*)

Definition at line 524 of file kern_sig.c.

References sigqueue_delete_set_proc().

Referenced by execsigs(), and kern_sigaction().

Here is the call graph for this function:



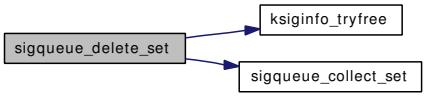
9.56.2.56 void sigqueue_delete_set (sigqueue_t * *sq*, sigset_t * *set*)

Definition at line 471 of file kern_sig.c.

References ksiginfo_tryfree(), and sigqueue_collect_set().

Referenced by sigqueue_delete().

Here is the call graph for this function:



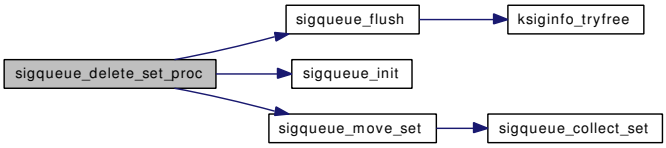
9.56.2.57 void sigqueue_delete_set_proc (struct proc * *p*, sigset_t * *set*)

Definition at line 505 of file kern_sig.c.

References sched_lock, sigqueue_flush(), sigqueue_init(), and sigqueue_move_set().

Referenced by sigqueue_delete_proc(), and sigqueue_delete_stopmask_proc().

Here is the call graph for this function:

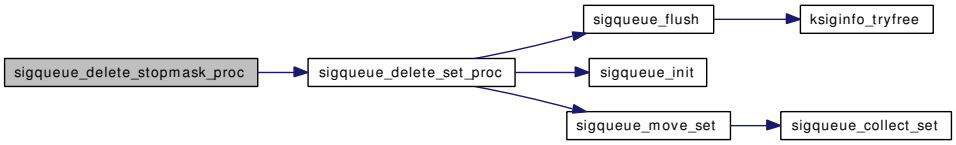


9.56.2.58 void sigqueue_delete_stopmask_proc (struct proc * *p*)

Definition at line 534 of file kern_sig.c.

References sigqueue_delete_set_proc().

Here is the call graph for this function:



9.56.2.59 void sigqueue_flush (sigqueue_t * *sq*)

Definition at line 383 of file kern_sig.c.

References ksiginfo_tryfree().

Referenced by exit1(), kse_exit(), sigqueue_delete_set_proc(), thr_exit(), and thread_suspend_check().

Here is the call graph for this function:

9.56.2.60 int sigqueue_get (sigqueue_t * *sq*, int *signo*, ksiginfo_t * *si*)

Definition at line 265 of file kern_sig.c.

References ksiginfo_tryfree().

Referenced by issignal(), and postsig().

Here is the call graph for this function:

9.56.2.61 void sigqueue_init (sigqueue_t * *list*, struct proc * *p*)

Definition at line 249 of file kern_sig.c.

Referenced by proc_linkup(), sigqueue_delete_set_proc(), and thread_link().

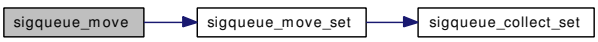
9.56.2.62 void sigqueue_move (sigqueue_t * *src*, sigqueue_t * *dst*, int *signo*)

Definition at line 461 of file kern_sig.c.

References sigqueue_move_set().

Referenced by kern_sigtimedwait().

Here is the call graph for this function:



9.56.2.63 void sigqueue_move_set (sigqueue_t * *src*, sigqueue_t * *dst*, sigset_t * *setp*)

Definition at line 417 of file kern_sig.c.

References sigqueue_collect_set().

Referenced by kse_create(), signotify(), sigqueue_delete_set_proc(), and sigqueue_move().

Here is the call graph for this function:

9.56.2.64 static void sigqueue_start (void) [static]

Definition at line 209 of file kern_sig.c.

References ksiginfo_zone, max_pending_per_proc, p31b_setcfg(), and preallocate_siginfo.

Here is the call graph for this function:

9.56.2.65 void sigqueue_take (ksiginfo_t * *ksi*)

Definition at line 302 of file kern_sig.c.

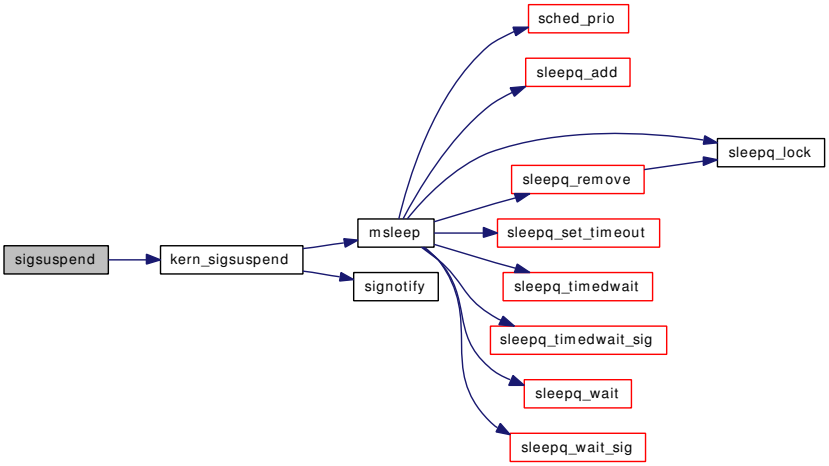
Referenced by aio_free_entry(), aio_proc_rundown(), do_lio_listio(), kern_ptrace(), kern_timer_delete(), kern_wait(), kmq_notify(), notifier_remove(), and proc_reparent().

9.56.2.66 int sigsuspend (struct thread * *td*, struct sigsuspend_args * *uap*)

Definition at line 1484 of file kern_sig.c.

References kern_sigsuspend(), mask, and sigsuspend_args::sigmask.

Here is the call graph for this function:



9.56.2.67 static struct thread * sigtd (struct proc * *p*, int *sig*, int *prop*) [static]

Definition at line 2007 of file kern_sig.c.

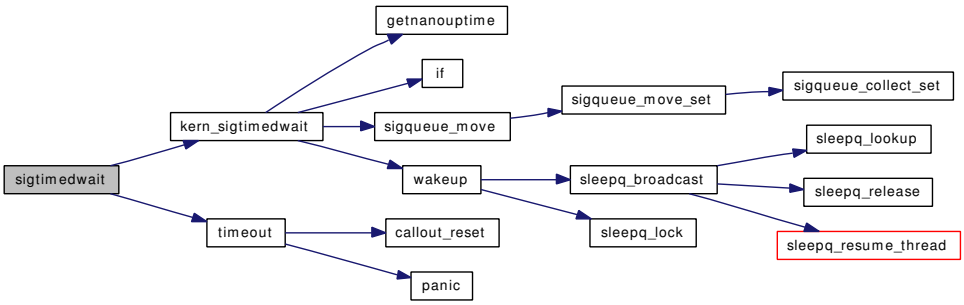
References sched_lock.

9.56.2.68 int sigtimedwait (struct thread * *td*, struct sigtimedwait_args * *uap*)

Definition at line 1129 of file kern_sig.c.

References kern_sigtimedwait(), and timeout().

Here is the call graph for this function:

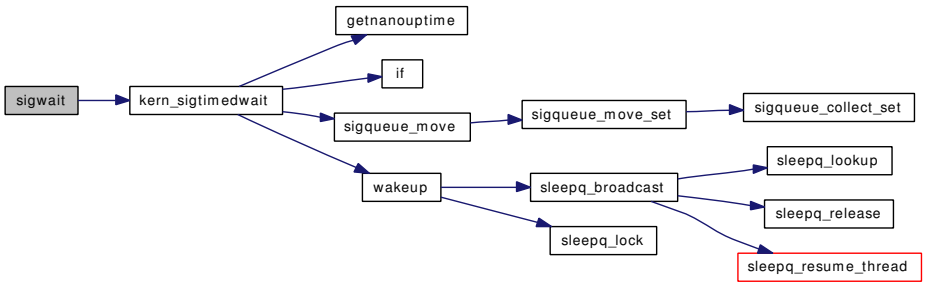


9.56.2.69 int sigwait (struct thread * *td*, struct sigwait_args * *uap*)

Definition at line 1101 of file kern_sig.c.

References kern_sigtimedwait().

Here is the call graph for this function:

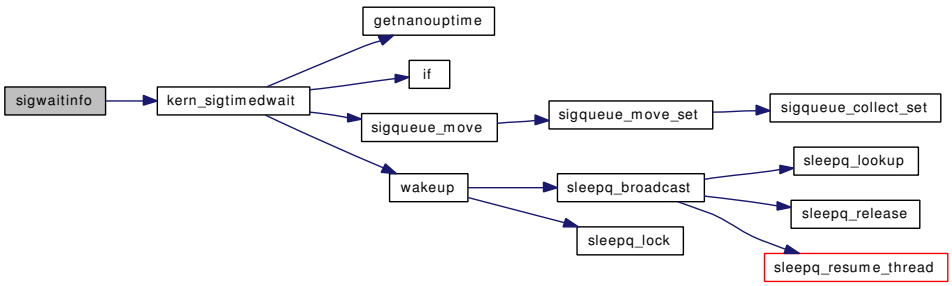


9.56.2.70 int sigwaitinfo (struct thread * *td*, struct sigwaitinfo_args * *uap*)

Definition at line 1166 of file kern_sig.c.

References kern_sigtimedwait().

Here is the call graph for this function:



9.56.2.71 SYSCTL_INT (_kern, OID_AUTO, nodump_coredump, CTLFLAG_RW, & *set_core_nodump_flag*, 0, "Enable setting the NODUMP flag on coredump files")

9.56.2.72 SYSCTL_INT (_kern, OID_AUTO, coredump, CTLFLAG_RW, & *do_coredump*, 0, "Enable/Disable coredumps")

9.56.2.73 SYSCTL_INT (_kern, OID_AUTO, sugid_coredump, CTLFLAG_RW, & *sugid_coredump*, 0, "Enable coredumping set user/group ID processes")

9.56.2.74 SYSCTL_INT (_kern_sigqueue, OID_AUTO, alloc_fail, CTLFLAG_RD, & *signal_alloc_fail*, 0, "signals failed to be allocated")

9.56.2.75 SYSCTL_INT (_kern_sigqueue, OID_AUTO, overflow, CTLFLAG_RD, & *signal_overflow*, 0, "Number of signals overflew")

9.56.2.76 SYSCTL_INT (_kern_sigqueue, OID_AUTO, preallocate, CTLFLAG_RD, & *preallocate_siginfo*, 0, "Preallocated signal memory size")

9.56.2.77 SYSCTL_INT (_kern_sigqueue, OID_AUTO, max_pending_per_proc, CTLFLAG_RW, & *max_pending_per_proc*, 0, "Max pending signals per proc")

9.56.2.78 SYSCTL_INT (_kern, OID_AUTO, forcesigexit, CTLFLAG_RW, & *kern_forcesigexit*, 0, "Force trap signal to be handled")

9.56.2.79 SYSCTL_INT (_kern, KERN_LOGSIGEXIT, logsigexit, CTLFLAG_RW, & *kern_logsigexit*, 0, "Log processes quitting on abnormal signals to syslog(3)")

9.56.2.80 SYSCTL_NODE (_kern, OID_AUTO, sigqueue, CTLFLAG_RW, 0, "POSIX real time signal")

9.56.2.81 SYSCTL_STRING (_kern, OID_AUTO, corefile, CTLFLAG_RW, corefilename, sizeof(corefilename), "process corefile name format string")

9.56.2.82 SYSINIT (signal, SI_SUB_P1003_1B, SI_ORDER_FIRST+ *3*, sigqueue_start, NULL)

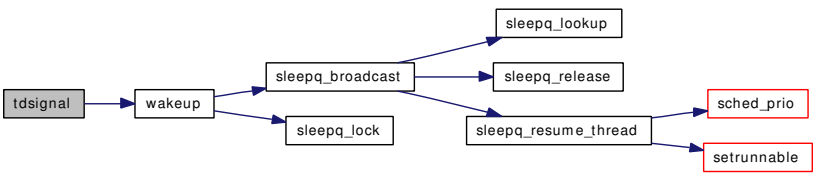
9.56.2.83 int tdsignal (struct proc * *p*, struct thread * *td*, int *sig*, ksiginfo_t * *ksi*)

Definition at line 2081 of file kern_sig.c.

References ret, and wakeup().

Referenced by kern_wait(), kse_thr_interrupt(), psignal(), psignal_event(), sigparent(), sigqueue(), thr_kill(), and trapsignal().

Here is the call graph for this function:

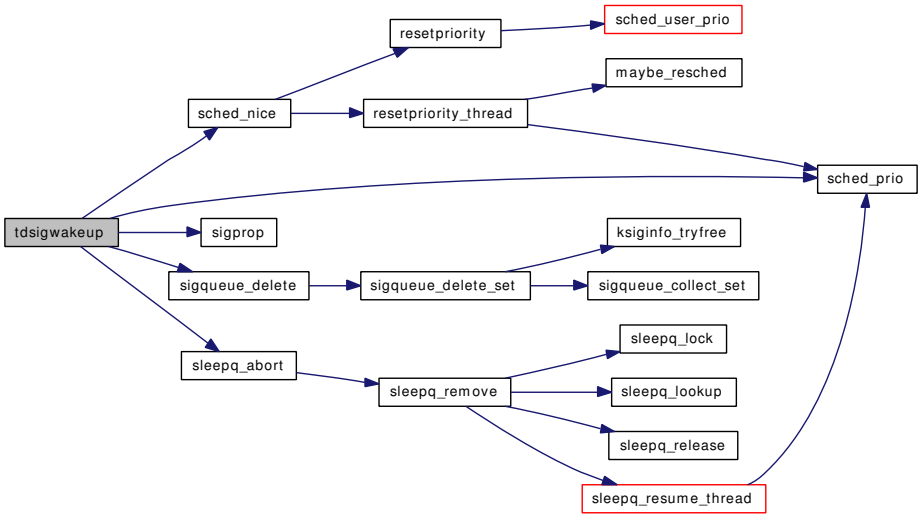


9.56.2.84 static void tdsigwakeup (struct thread *, int, sig_t, int) [static]

Definition at line 2400 of file kern_sig.c.

References SA_KILL, sched_lock, sched_nice(), sched_prio(), sigprop(), sigqueue_delete(), and sleepq_abort().

Here is the call graph for this function:



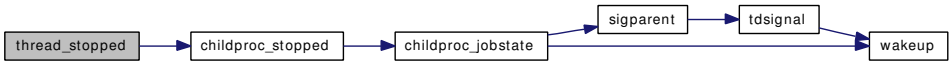
9.56.2.85 void thread_stopped (struct proc * *p*)

Definition at line 2739 of file kern_sig.c.

References childproc_stopped(), and sched_lock.

Referenced by issignal(), kse_exit(), kse_thr_interrupt(), ptracestop(), thr_exit(), thread_single(), and thread_suspend_check().

Here is the call graph for this function:



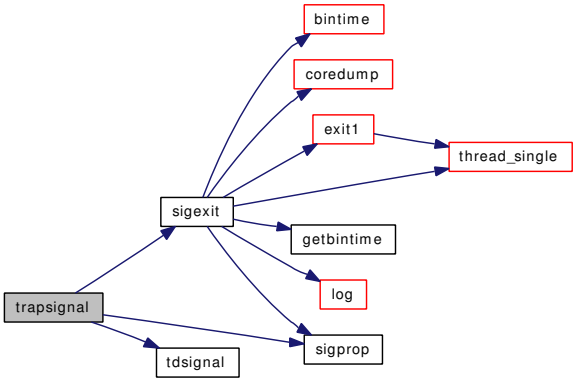
9.56.2.86 void trapsignal (struct thread * *td*, ksiginfo_t * *ksi*)

Definition at line 1896 of file kern_sig.c.

References kern_forcesigexit, SA_IGNORE, sched_lock, sigexit(), sigprop(), and tdsignal().

Referenced by ast().

Here is the call graph for this function:



9.56.2.87 TUNABLE_INT ("kern.sigqueue.preallocate", & *preallocate_siginfo*)

9.56.3 Variable Documentation

9.56.3.1 char corefilename[MAXPATHLEN] = {"%N.core"} [static]

Definition at line 3024 of file kern_sig.c.

9.56.3.2 int do_coredump = 1 [static]

Definition at line 152 of file kern_sig.c.

Referenced by coredump().

9.56.3.3 int kern_forcesigexit = 1 [static]

Definition at line 111 of file kern_sig.c.

Referenced by trapsignal().

9.56.3.4 int kern_logsigexit = 1 [static]

Definition at line 106 of file kern_sig.c.

Referenced by sigexit().

9.56.3.5 uma_zone_t ksiginfo_zone = NULL [static]

Definition at line 102 of file kern_sig.c.

Referenced by ksiginfo_alloc(), ksiginfo_free(), ksiginfo_tryfree(), sigqueue_add(), and sigqueue_start().

9.56.3.6 int max_pending_per_proc = 128 [static]

Definition at line 117 of file kern_sig.c.

Referenced by sigqueue_add(), and sigqueue_start().

9.56.3.7 int preallocate_siginfo = 1024 [static]

Definition at line 121 of file kern_sig.c.

Referenced by sigqueue_start().

9.56.3.8 int set_core_nodump_flag = 0 [static]

Definition at line 156 of file kern_sig.c.

Referenced by coredump().

9.56.3.9 struct filterops sig_filtops

Initial value:

```
{ 0, filt_sigattach, filt_sigdetach, filt_signal }
```

Definition at line 103 of file kern_sig.c.

9.56.3.10 int signal_alloc_fail = 0 [static]

Definition at line 130 of file kern_sig.c.

Referenced by sigqueue_add().

9.56.3.11 int signal_overflow = 0 [static]

Definition at line 126 of file kern_sig.c.

Referenced by sigqueue_add().

9.56.3.12 int sigproptbl[NSIG] [static]

Definition at line 174 of file kern_sig.c.

Referenced by sigprop().

9.56.3.13 int sugid_coredump

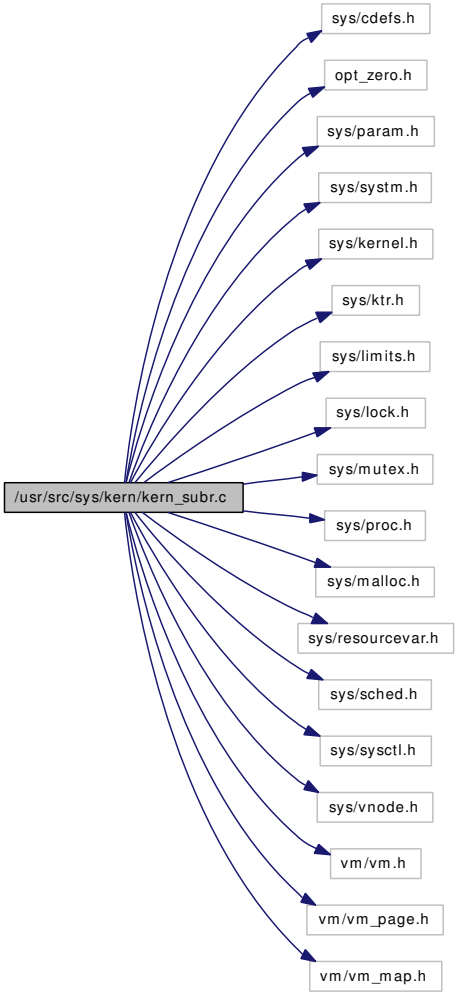
Definition at line 148 of file kern_sig.c.

Referenced by coredump().

9.57 /usr/src/sys/kern/kern_subr.c File Reference

```
#include <sys/cdefs.h>
#include "opt_zero.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/ktr.h>
#include <sys/limits.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/malloc.h>
#include <sys/resourcevar.h>
#include <sys/sched.h>
#include <sys/sysctl.h>
#include <sys/vnode.h>
#include <vm/vm.h>
#include <vm/vm_page.h>
#include <vm/vm_map.h>
```

Include dependency graph for kern_subr.c:



Defines

- #define NPRIMES (sizeof(primes) / sizeof(primes[0]))

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_subr.c,v 1.102 2007/01/16 11:40:55 rrs Exp $")
- SYSCTL_INT (_kern, KERN_IOV_MAX, iov_max, CTLFLAG_RD, NULL, UIO_-MAXIOV,"Maximum number of elements in an I/O vector; sysconf(_SC_IOV_MAX)")
- int uiomove (void *cp, int n, struct uio *uio)
- int uiomove_frombuf (void *buf, int buflen, struct uio *uio)
- int ureadc (int c, struct uio *uio)
- void * hashinit_flags (int elements, struct malloc_type *type, u_long *hashmask, int flags)
- void * hashinit (int elements, struct malloc_type *type, u_long *hashmask)
- void hashdestroy (void *vhashtbl, struct malloc_type *type, u_long hashmask)
- void * phashinit (int elements, struct malloc_type *type, u_long *nentries)
- void uio_yield (void)
- int copyinfrom (const void *__restrict src, void *__restrict dst, size_t len, int seg)

- int copyinstrfrom (const void *__restrict src, void *__restrict dst, size_t len, size_t *__restrict copied, int seg)
- int copyiniov (struct iovec *iovp, u_int iovcnt, struct iovec **iov, int error)
- int copyinuio (struct iovec *iovp, u_int iovcnt, struct uio **uiop)
- uio * cloneuio (struct uio *uiop)

Variables

- static int primes [ ]

9.57.1 Define Documentation

9.57.1.1 #define NPRIMES (sizeof(primes) / sizeof(primes[0]))

Definition at line 422 of file kern_subr.c.

Referenced by phashinit().

9.57.2 Function Documentation

9.57.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_subr. *c*, v 1.102 2007/01/16 11:40:55 rrs Exp $")

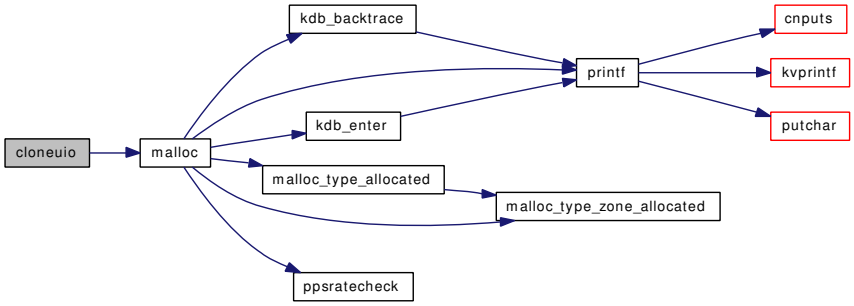
9.57.2.2 struct uio* cloneuio (struct uio * *uiop*)

Definition at line 557 of file kern_subr.c.

References malloc().

Referenced by dofileread(), dofilewrite(), kevent(), and log_console().

Here is the call graph for this function:

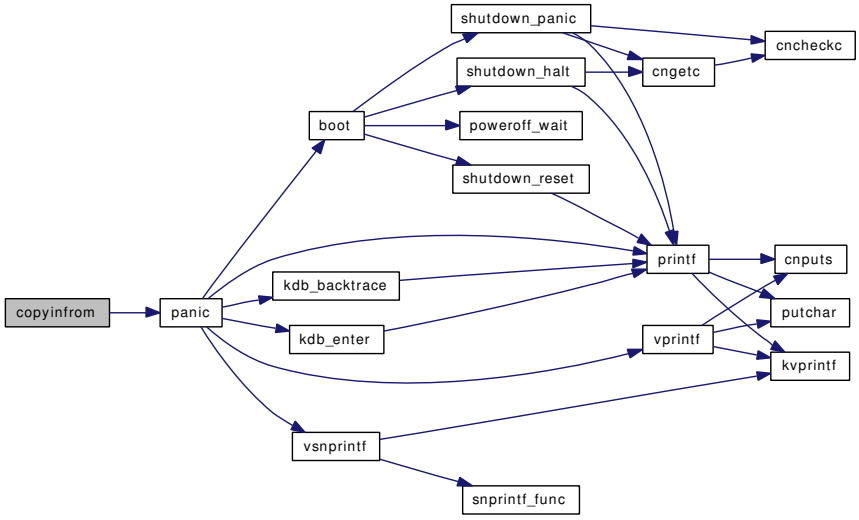


9.57.2.3 int copyinfrom (const void *__restrict *src*, void *__restrict *dst*, size_t *len*, int *seg*)

Definition at line 465 of file kern_subr.c.

References panic().

Here is the call graph for this function:



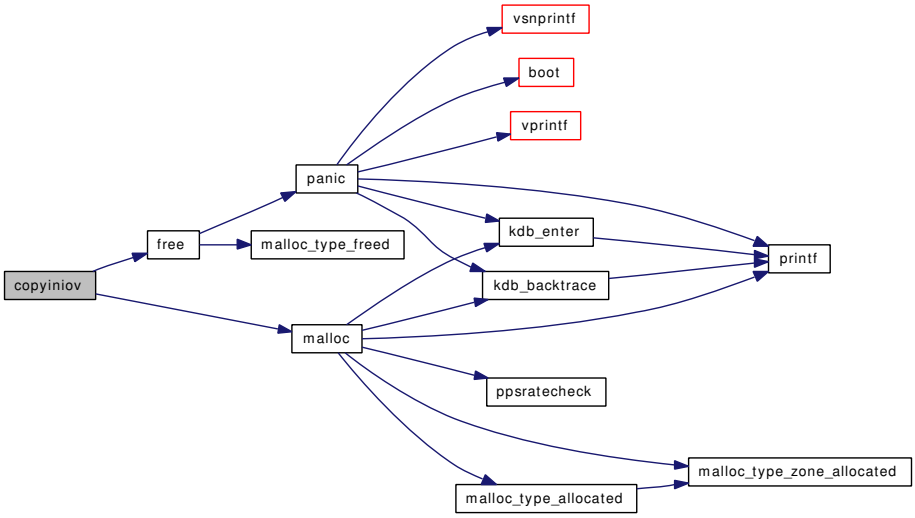
9.57.2.4 int copyiniov (struct iovec * *iovp*, u_int *iovcnt*, struct iovec ** *iov*, int *error*)

Definition at line 503 of file kern_subr.c.

References free(), and malloc().

Referenced by recvmsg(), sctp_generic_recvmsg(), sctp_generic_sendmsg_iov(), and sendmsg().

Here is the call graph for this function:

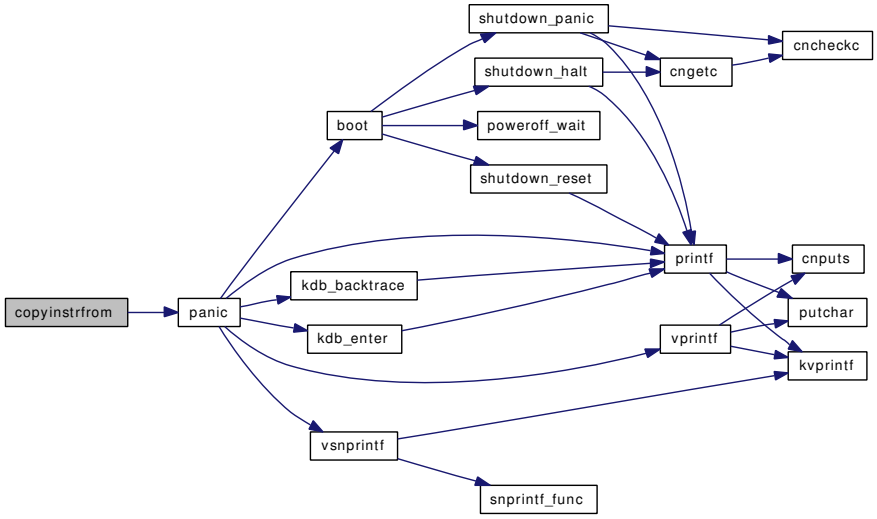


9.57.2.5 int copyinstrfrom (const void *__restrict *src*, void *__restrict *dst*, size_t *len*, size_t *__restrict *copied*, int *seg*)

Definition at line 484 of file kern_subr.c.

References panic().

Here is the call graph for this function:



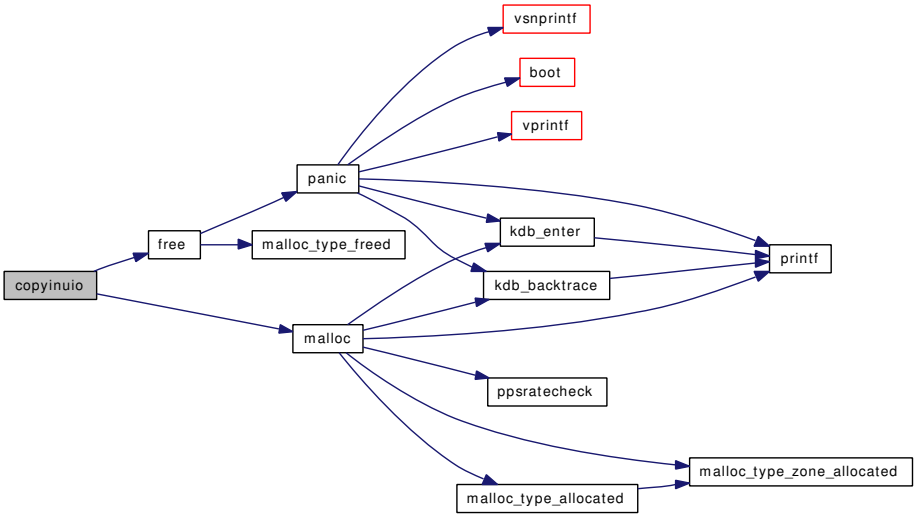
9.57.2.6 int copyinuio (struct iovec * *iovp*, u_int *iovcnt*, struct uio ** *uiop*)

Definition at line 521 of file kern_subr.c.

References free(), and malloc().

Referenced by do_sendfile(), nmount(), preadv(), pwritev(), readv(), and writev().

Here is the call graph for this function:

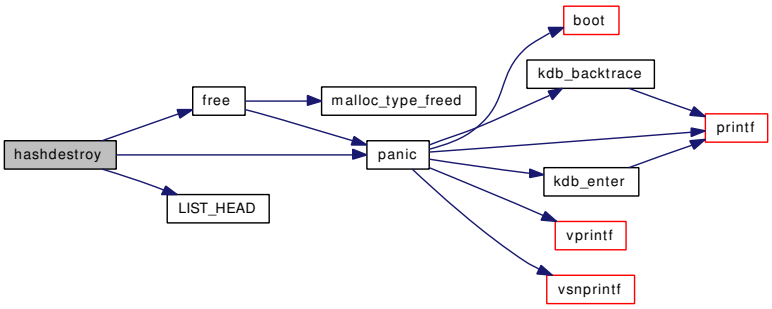


9.57.2.7 void hashdestroy (void * *vhashtbl*, struct malloc_type * *type*, u_long *hashmask*)

Definition at line 408 of file kern_subr.c.

References free(), LIST_HEAD(), and panic().

Here is the call graph for this function:



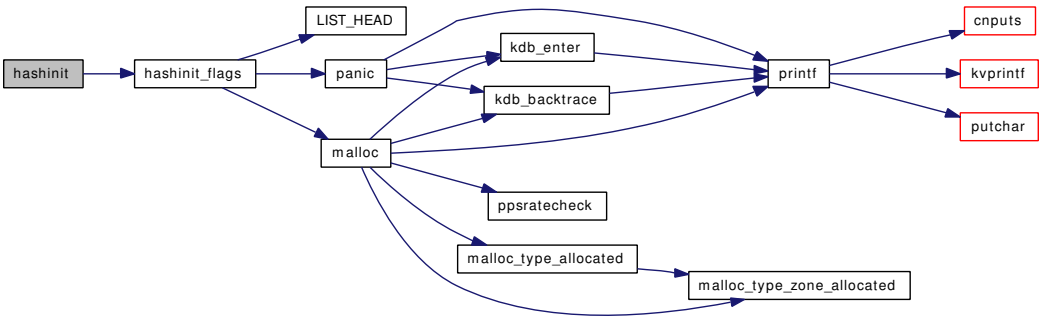
9.57.2.8 void* hashinit (int *elements*, struct malloc_type * *type*, u_long * *hashmask*)

Definition at line 401 of file kern_subr.c.

References hashinit_flags().

Referenced by kqueue_expand(), LIST_HEAD(), nchinit(), procinit(), uihashinit(), and vntblinit().

Here is the call graph for this function:



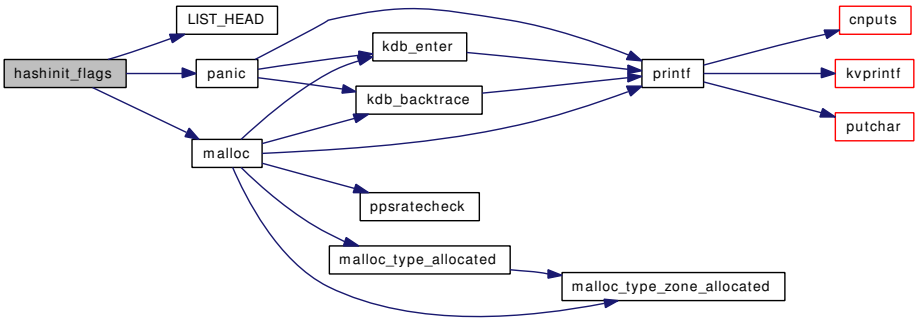
9.57.2.9 void* hashinit_flags (int *elements*, struct malloc_type * *type*, u_long * *hashmask*, int *flags*)

Definition at line 364 of file kern_subr.c.

References LIST_HEAD(), malloc(), and panic().

Referenced by hashinit().

Here is the call graph for this function:

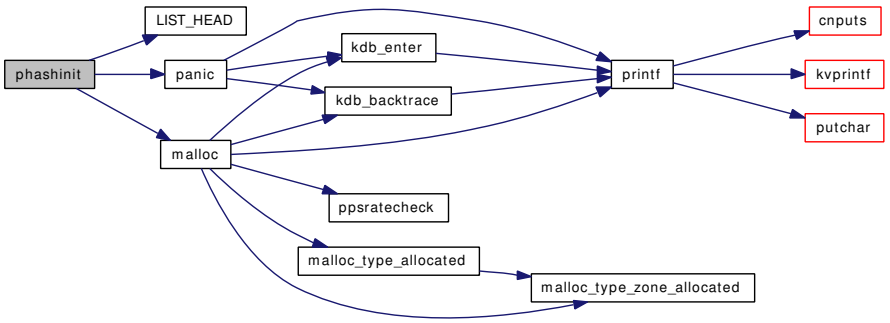


9.57.2.10 void* phashinit (int *elements*, struct malloc_type * *type*, u_long * *nentries*)

Definition at line 428 of file kern_subr.c.

References LIST_HEAD(), malloc(), NPRIMES, and panic().

Here is the call graph for this function:



9.57.2.11 SYSCTL_INT (_kern, KERN_IOV_MAX, iov_max, CTLFLAG_RD, NULL, UIO_MAXIOV, "Maximum number of elements in an I/O vector; sysconf(_SC_IOV_MAX)")

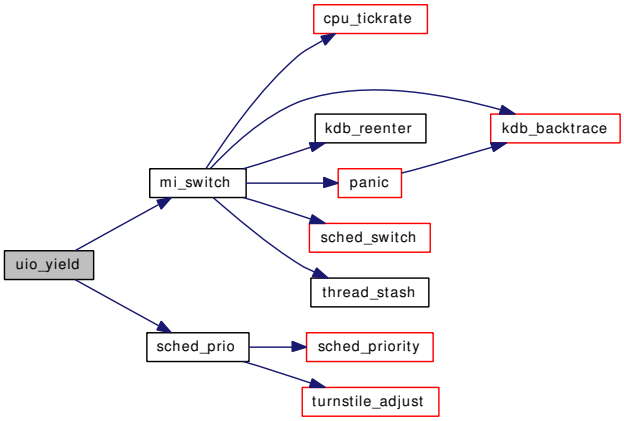
9.57.2.12 void uio_yield (void)

Definition at line 451 of file kern_subr.c.

References mi_switch(), sched_lock, sched_prio(), and td.

Referenced by buf_daemon(), uiomove(), vlrureclaim(), vn_rdwr_inchunks(), and vnlru_proc().

Here is the call graph for this function:



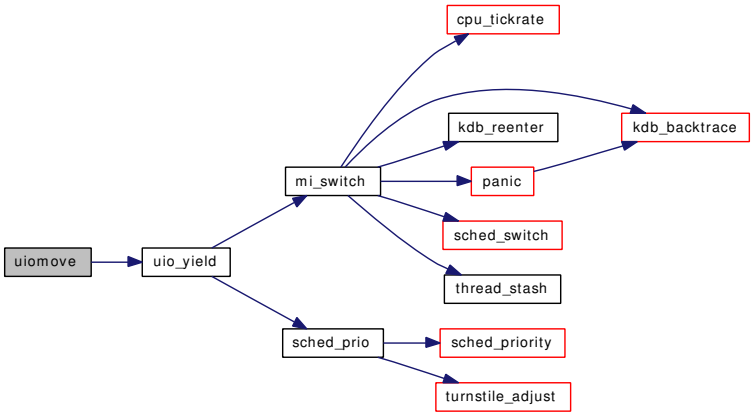
9.57.2.13 int uiomove (void * *cp*, int *n*, struct uio * *uio*)

Definition at line 133 of file kern_subr.c.

References hogticks, td, ticks, and uio_yield().

Referenced by devread(), log_console(), logread(), m_uiotombuf(), pipe_read(), pipe_write(), ptcread(), ptcwrite(), sbuf_uionew(), soreceive_rcvoob(), ttread(), ttwrite(), uiomove_frombuf(), and vfs_read_dirent().

Here is the call graph for this function:



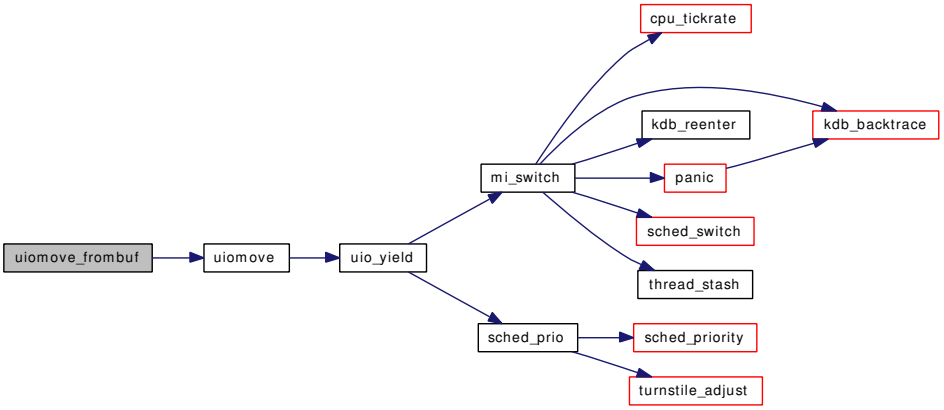
9.57.2.14 int uiomove_frombuf (void * *buf*, int *buflen*, struct uio * *uio*)

Definition at line 205 of file kern_subr.c.

References uiomove().

Referenced by mqfs_read().

Here is the call graph for this function:



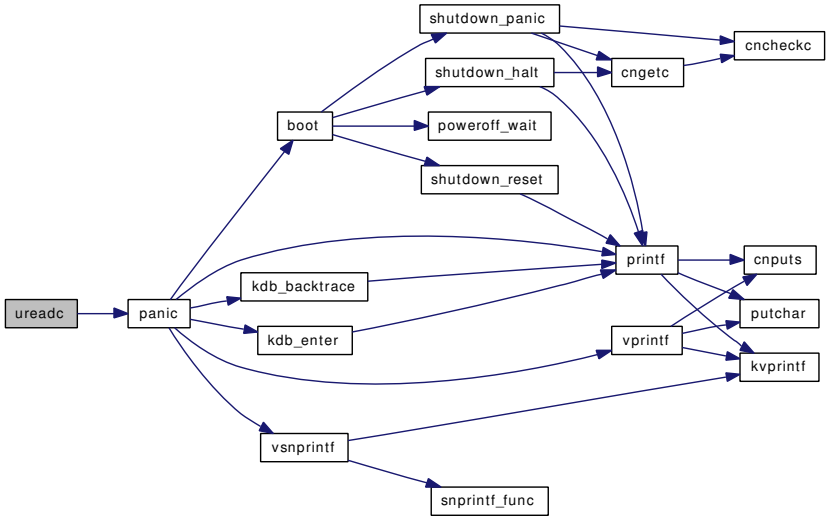
9.57.2.15 int ureadc (int *c*, struct uio * *uio*)

Definition at line 323 of file kern_subr.c.

References panic().

Referenced by ptcread(), and ttread().

Here is the call graph for this function:



9.57.3 Variable Documentation

9.57.3.1 int primes[ ] [static]

Initial value:

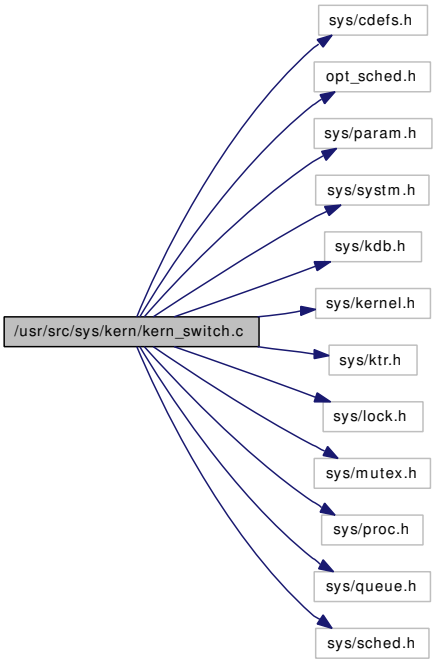
```
{ 1, 13, 31, 61, 127, 251, 509, 761, 1021, 1531, 2039,
                      2557, 3067, 3583, 4093, 4603, 5119, 5623, 6143, 6653,
                      7159, 7673, 8191, 12281, 16381, 24571, 32749 }
```

Definition at line 419 of file kern_subr.c.

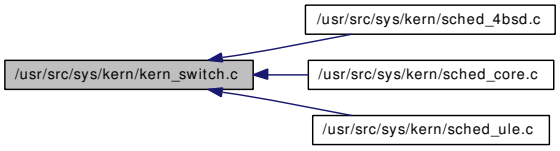
9.58 /usr/src/sys/kern/kern_switch.c File Reference

```
#include <sys/cdefs.h>
#include "opt_sched.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kdb.h>
#include <sys/kernel.h>
#include <sys/ktr.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/queue.h>
#include <sys/sched.h>
```

Include dependency graph for kern_switch.c:



This graph shows which files directly or indirectly include this file:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_switch.c,v 1.129 2007/02/08 01:52:25 jeff Exp $")

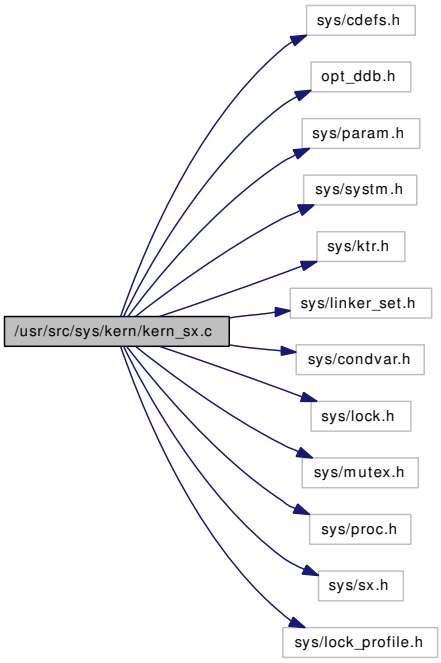
9.58.1 Function Documentation

9.58.1.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_switch. *c*, v 1.129 2007/02/08 01:52:25 jeff Exp $")

9.59 /usr/src/sys/kern/kern_sx.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ddb.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/ktr.h>
#include <sys/linker_set.h>
#include <sys/condvar.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/sx.h>
#include <sys/lock_profile.h>
```

Include dependency graph for kern_sx.c:



Defines

- #define _sx_assert(sx, what, file, line)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_sx.c,v 1.32 2006/11/13 05:41:46 kmacy Exp $")

- void sx_sysinit (void *arg)
- void sx_init (struct sx *sx, const char *description)
- void sx_destroy (struct sx *sx)
- void _sx_slock (struct sx *sx, const char *file, int line)
- int _sx_try_slock (struct sx *sx, const char *file, int line)
- void _sx_xlock (struct sx *sx, const char *file, int line)
- int _sx_try_xlock (struct sx *sx, const char *file, int line)
- void _sx_sunlock (struct sx *sx, const char *file, int line)
- void _sx_xunlock (struct sx *sx, const char *file, int line)
- int _sx_try_upgrade (struct sx *sx, const char *file, int line)
- void _sx_downgrade (struct sx *sx, const char *file, int line)

Variables

- lock_class lock_class_sx

9.59.1 Define Documentation

9.59.1.1 #define _sx_assert(sx, what, file, line)

Definition at line 67 of file kern_sx.c.

Referenced by _sx_downgrade(), _sx_sunlock(), _sx_try_upgrade(), and _sx_xunlock().

9.59.2 Function Documentation

9.59.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_sx. *c*, v 1.32 2006/11/13 05:41:46 kmacy Exp $")

9.59.2.2 void _sx_downgrade (struct sx * *sx*, const char * *file*, int *line*)

Definition at line 318 of file kern_sx.c.

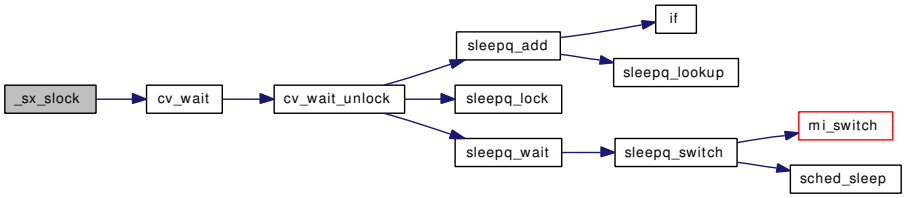
References _sx_assert.

9.59.2.3 void _sx_slock (struct sx * *sx*, const char * *file*, int *line*)

Definition at line 111 of file kern_sx.c.

References cv_wait(), and waittime.

Here is the call graph for this function:

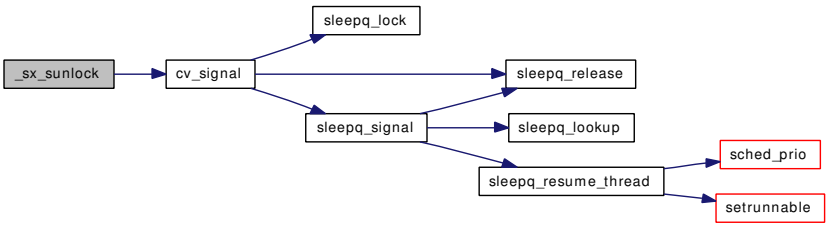


9.59.2.4 void _sx_sunlock (struct sx * *sx*, const char * *file*, int *line*)

Definition at line 233 of file kern_sx.c.

References _sx_assert, and cv_signal().

Here is the call graph for this function:



9.59.2.5 int _sx_try_slock (struct sx * *sx*, const char * *file*, int *line*)

Definition at line 148 of file kern_sx.c.

9.59.2.6 int _sx_try_upgrade (struct sx * *sx*, const char * *file*, int *line*)

Definition at line 294 of file kern_sx.c.

References _sx_assert.

9.59.2.7 int _sx_try_xlock (struct sx * *sx*, const char * *file*, int *line*)

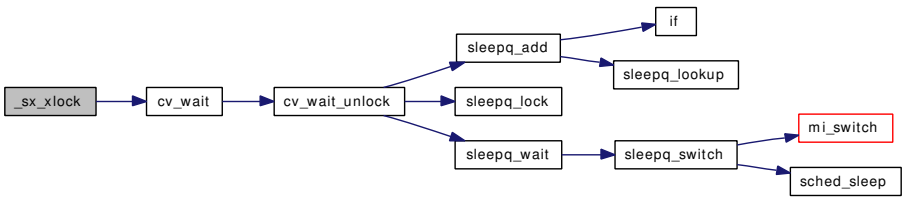
Definition at line 212 of file kern_sx.c.

9.59.2.8 void _sx_xlock (struct sx * *sx*, const char * *file*, int *line*)

Definition at line 167 of file kern_sx.c.

References cv_wait(), and waittime.

Here is the call graph for this function:

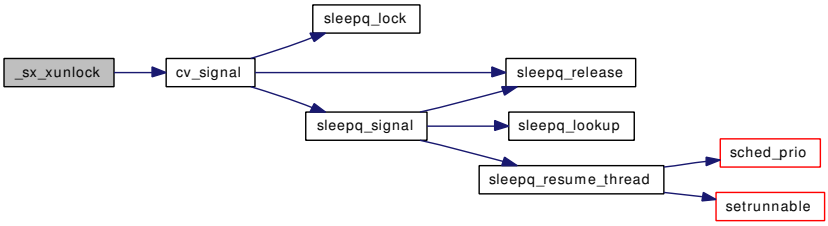


9.59.2.9 void _sx_xunlock (struct sx * *sx*, const char * *file*, int *line*)

Definition at line 265 of file kern_sx.c.

References _sx_assert, and cv_signal().

Here is the call graph for this function:



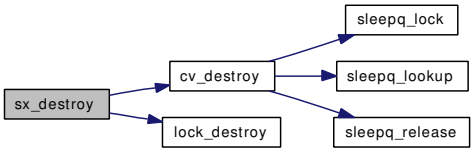
9.59.2.10 void sx_destroy (struct sx * *sx*)

Definition at line 95 of file kern_sx.c.

References cv_destroy(), and lock_destroy().

Referenced by mqfs_uninit().

Here is the call graph for this function:



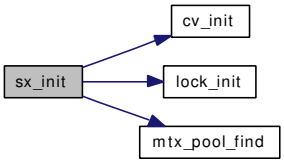
9.59.2.11 void sx_init (struct sx * *sx*, const char * *description*)

Definition at line 79 of file kern_sx.c.

References cv_init(), lock_class_sx, lock_init(), mtx_pool_find(), and mtxpool_lockbuilder.

Referenced by filelistinit(), linker_init(), module_init(), mqfs_init(), procinit(), and sx_sysinit().

Here is the call graph for this function:

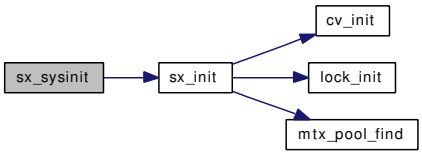


9.59.2.12 void sx_sysinit (void * *arg*)

Definition at line 71 of file kern_sx.c.

References sx_init().

Here is the call graph for this function:



9.59.3 Variable Documentation

9.59.3.1 struct lock_class lock_class_sx

Initial value:

```
{
        "sx",
        LC_SLEEPLOCK | LC_SLEEPABLE | LC_RECURSABLE | LC_UPGRADABLE,


}
```

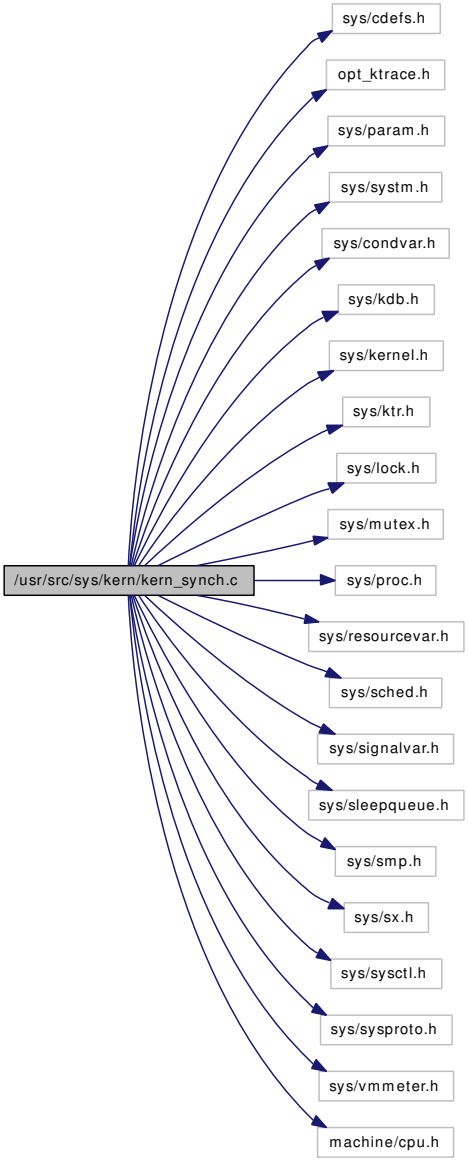
Definition at line 58 of file kern_sx.c.

Referenced by sx_init().

9.60 /usr/src/sys/kern/kern_synch.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ktrace.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/condvar.h>
#include <sys/kdb.h>
#include <sys/kernel.h>
#include <sys/ktr.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/resourcevar.h>
#include <sys/sched.h>
#include <sys/signalvar.h>
#include <sys/sleepqueue.h>
#include <sys/smp.h>
#include <sys/sx.h>
#include <sys/sysctl.h>
#include <sys/sysproto.h>
#include <sys/vmmeter.h>
#include <machine/cpu.h>
```

Include dependency graph for kern_synch.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_synch.c,v 1.288 2007/02/23 16:22:09 jhb Exp $")
- static void synch_setup (void *dummy)
- SYSCTL_INT (_kern, OID_AUTO, fscale, CTLFLAG_RD, 0, FSCALE,"")
- static void loadav (void *arg)
- static void lboltcb (void *arg)
- void sleepinit (void)
- int msleep (void *ident, struct mtx *mtx, int priority, const char *wmesg, int timo)
- int msleep_spin (void *ident, struct mtx *mtx, const char *wmesg, int timo)
- int pause (char *wmesg, int timo) const
- void wakeup (void *ident)
- void wakeup_one (void *ident)
- void mi_switch (int flags, struct thread *newtd)

- void setrunnable (struct thread *td)
- int yield (struct thread *td, struct yield_args *uap)

Variables

- int hogticks
- int lbolt
- static int pause_wchan
- static struct callout loadav_callout
- static struct callout lbolt_callout
- loadavg averunnable
- static fixpt_t cexp [3]
- static int fscale __unused = FSCALE

9.60.1 Function Documentation

9.60.1.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_synch. *c*, v 1.288 2007/02/23 16:22:09 jhb Exp $")

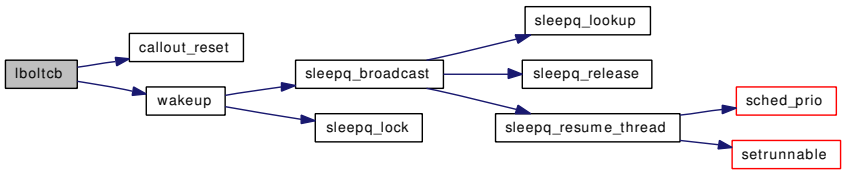
9.60.1.2 static void lboltcb (void * *arg*) [static]

Definition at line 548 of file kern_synch.c.

References callout_reset(), hz, lbolt, lbolt_callout, and wakeup().

Referenced by synch_setup().

Here is the call graph for this function:



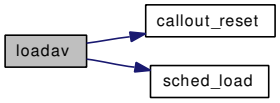
9.60.1.3 static void loadav (void * *arg*) [static]

Definition at line 526 of file kern_synch.c.

References averunnable, callout_reset(), cexp, hz, loadav_callout, and sched_load().

Referenced by synch_setup().

Here is the call graph for this function:



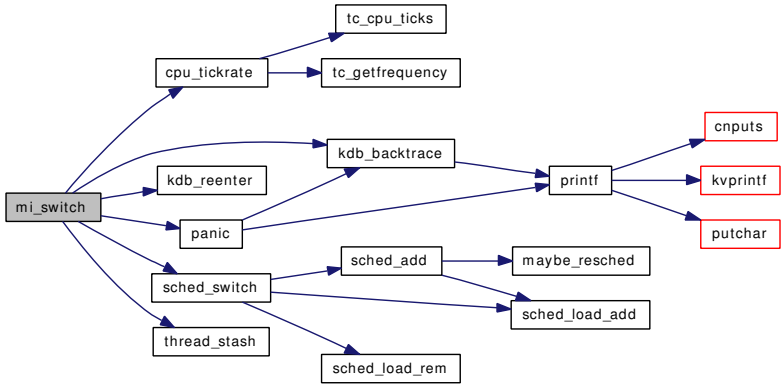
9.60.1.4 void mi_switch (int *flags*, struct thread * *newtd*)

Definition at line 360 of file kern_synch.c.

References cpu_tickrate(), cpu_ticks, Giant, kdb_active, kdb_backtrace(), kdb_reenter(), panic(), panicstr, sched_lock, sched_switch(), td, thread_stash(), and ticks.

Referenced by _mtx_unlock_sleep(), ast(), issignal(), ithread_loop(), kse_thr_interrupt(), poll_idle(), ptracestop(), sched_bind(), sched_idletd(), sched_preempt(), sched_relinquish(), sleepq_check_timeout(), sleepq_switch(), thread_single(), thread_suspend_check(), turnstile_wait(), and uio_yield().

Here is the call graph for this function:



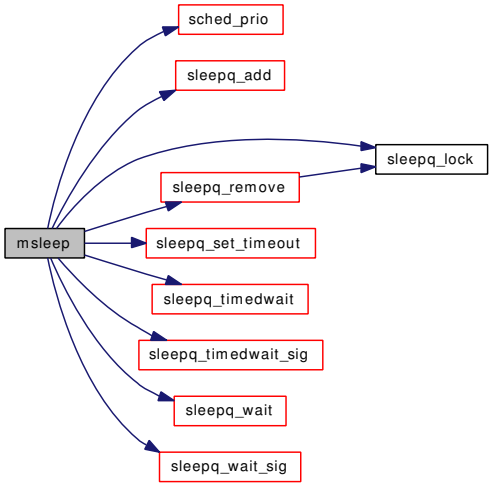
9.60.1.5 int msleep (void * *ident*, struct mtx * *mtx*, int *priority*, const char * *wmesg*, int *timo*)

Definition at line 120 of file kern_synch.c.

References Giant, lbolt, pause_wchan, sched_lock, sched_prio(), sleepq_add(), sleepq_lock(), sleepq_remove(), sleepq_set_timeout(), sleepq_timedwait(), sleepq_timedwait_sig(), sleepq_wait(), sleepq_wait_sig(), and td.

Referenced by _mqueue_recv(), _mqueue_send(), acquire(), acquiredrain(), aio_daemon(), aio_proc_rundown(), aio_suspend(), aio_waitcomplete(), ald_daemon(), biowait(), buf_daemon(), bufobj_wwait(), bunpin_wait(), bwait(), bwillwrite(), destroy_devl(), do_lio_listio(), exit1(), fdfree(), fdrop_locked(), fork1(), getnewbuf(), getnewvnode(), intr_event_add_handler(), intr_event_remove_handler(), kern_accept(), kern_connect(), kern_msgrcv(), kern_sigsuspend(), kern_thr_suspend(), kern_timer_delete(), kern_wait(), knlist_cleardel(), knote_fdclose(), kqueue_close(), kqueue_register(), kqueue_scan(), kse_release(), kthread_suspend(), kthread_suspend_check(), pipe_direct_write(), pipe_read(), pipe_write(), pipeclose(), pipelock(), rman_await_resource(), root_mount_wait(), sb_lock(), sbwait(), sched_sync(), stopevent(), stopprofclock(), TAILQ_HEAD(), taskqueue_drain(), TQ_SLEEP(), transferlockers(), umtxq_busy(), umtxq_sleep(), umtxq_sleep_pi(), vfs_busy(), vfs_mount_destroy(), vfs_write_suspend(), vn_read(), vn_start_secondary_write(), vn_start_write(), vn_write_suspend_wait(), vnlru_proc(), and waitrunningbufspace().

Here is the call graph for this function:



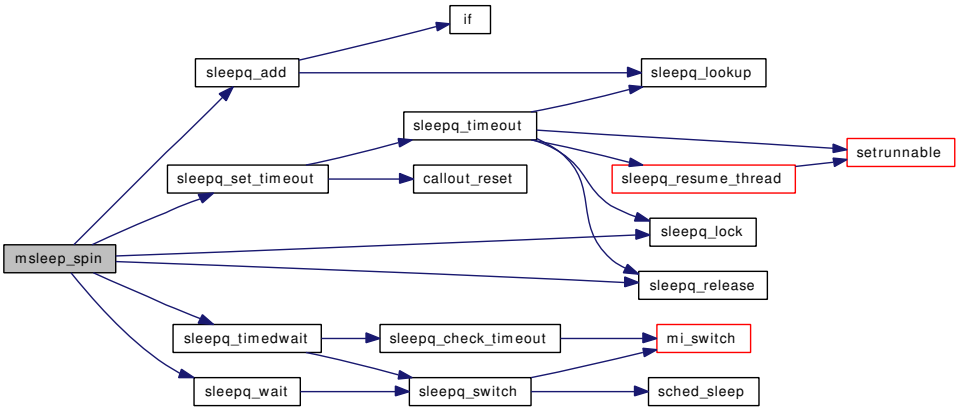
9.60.1.6 int msleep_spin (void * *ident*, struct mtx * *mtx*, const char * *wmesg*, int *timo*)

Definition at line 233 of file kern_synch.c.

References sleepq_add(), sleepq_lock(), sleepq_release(), sleepq_set_timeout(), sleepq_timedwait(), sleepq_wait(), and td.

Referenced by _callout_stop_safe(), taskqueue_drain(), and TQ_SLEEP().

Here is the call graph for this function:



9.60.1.7 int pause (char * *wmesg*, int *timo*) const

Definition at line 321 of file kern_synch.c.

References pause_wchan.

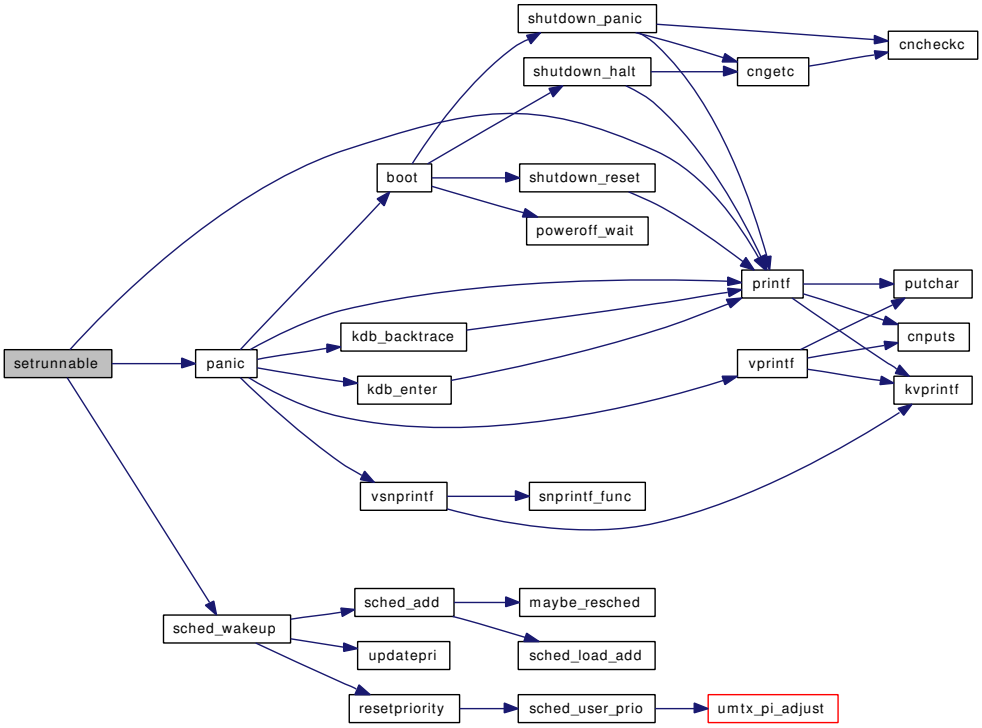
9.60.1.8 void setrunnable (struct thread * *td*)

Definition at line 475 of file kern_synch.c.

References panic(), printf(), sched_lock, and sched_wakeup().

Referenced by sleepq_resume_thread(), sleepq_timeout(), and thread_unsuspend_one().

Here is the call graph for this function:



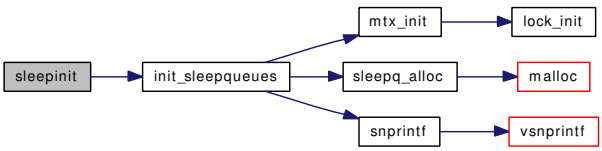
9.60.1.9 void sleepinit (void)

Definition at line 97 of file kern_synch.c.

References hogticks, hz, and init_sleepqueues().

Referenced by proc0_init().

Here is the call graph for this function:

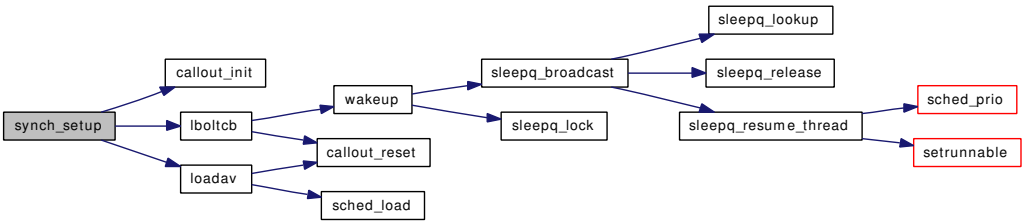


9.60.1.10 static void synch_setup (void * *dummy*) [static]

Definition at line 556 of file kern_synch.c.

References callout_init(), lbolt_callout, lboltcb(), loadav(), and loadav_callout.

Here is the call graph for this function:



9.60.1.11 SYSCTL_INT (_kern, OID_AUTO, fscale, CTLFLAG_RD, 0, FSCALE, "")

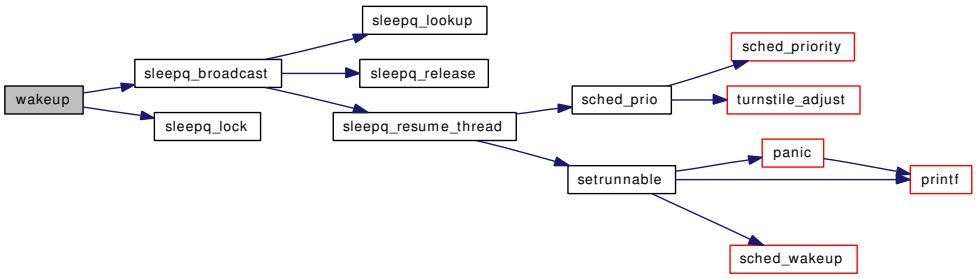
9.60.1.12 void wakeup (void * *ident*)

Definition at line 334 of file kern_synch.c.

References sleepq_broadcast(), and sleepq_lock().

Referenced by __mnt_vnode_first(), __mnt_vnode_markerfree(), _lockmgr(), acct(), acquire(), addupc_task(), aio_bio_done_notify(), aio_kick(), aio_kick_nowait(), ald_activate(), ald_daemon(), alq_flush(), bd_wakeup(), bdone(), biodone(), bufcountwakeup(), bufobj_wdrop(), bufspacewakeup(), bunpin(), childproc_jobstate(), closef(), config_intrhook_disestablish(), do_dup(), dounmount(), ether_poll_register(), exit1(), getnewvnode(), int_rman_deactivate_resource(), intr_event_add_handler(), ithread_execute_handlers(), itimer_leave(), kern_close(), kern_msgctl(), kern_msgrcv(), kern_semctl(), kern_sigaction(), kern_sigtimedwait(), kern_wait(), knote_fdclose(), kqueue_release(), kqueue_task(), kqueue_wakeup(), kse_wakeup(), kthread_exit(), kthread_resume(), kthread_suspend(), kthread_suspend_check(), lboltcb(), lf_wakelock(), logtimeout(), numdirtywakeup(), pipe_direct_write(), pipe_read(), pipe_write(), pipeclose(), pipeunlock(), ptcwakeup(), ptcwrite(), realitexpire(), root_mount_rel(), runningbufwakeup(), semexit_myhook(), shareunlock(), shmget_allocate_segment(), signotify(), softclock(), soisconnected(), soisdisconnected(), soisdisconnecting(), sowakeup(), stopevent(), taskqueue_run(), taskqueue_terminate(), tdsignal(), thr_wake(), transferlockers(), ttioctl(), ttwakeup(), ttwwakeup(), ttydtrwaitwakeup(), ttygone(), ttymodem(), umtxq_signal(), umtxq_signal_thread(), unp_gc(), vfs_write_resume(), vn_finished_secondary_write(), vn_finished_write(), vn_read(), and vnlru_proc().

Here is the call graph for this function:



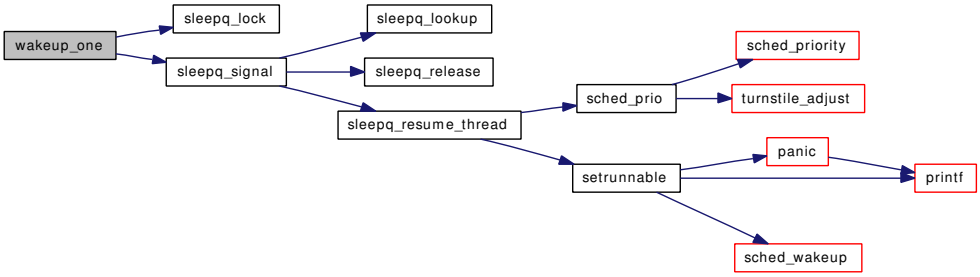
9.60.1.13 void wakeup_one (void * *ident*)

Definition at line 348 of file kern_synch.c.

References sleepq_lock(), and sleepq_signal().

Referenced by _mqueue_recv(), _mqueue_send(), soisconnected(), sonewconn(), taskqueue_thread_enqueue(), taskqueue_thread_loop(), and umtxq_unbusy().

Here is the call graph for this function:



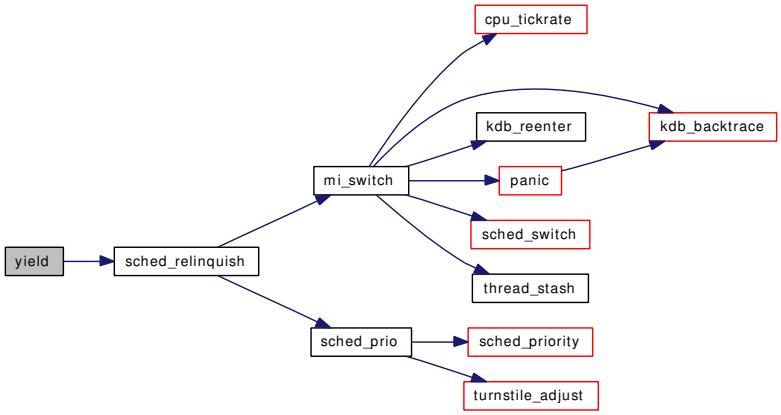
9.60.1.14 int yield (struct thread * *td*, struct yield_args * *uap*)

Definition at line 571 of file kern_synch.c.

References Giant, and sched_relinquish().

Referenced by vlrureclaim().

Here is the call graph for this function:



9.60.2 Variable Documentation

9.60.2.1 int fscale __unused = FSCALE [static]

Definition at line 90 of file kern_synch.c.

9.60.2.2 struct loadavg averunnable

Initial value:

```
{ {0, 0, 0}, FSCALE }
```

Definition at line 77 of file kern_synch.c.

Referenced by loadav(), schedcpu(), ttyinfo(), and updatepri().

9.60.2.3 fixpt_t cexp[3] [static]

Initial value:

```
 {
        0.9200444146293232 * FSCALE,
        0.9834714538216174 * FSCALE,
        0.9944598480048967 * FSCALE,
}
```

Definition at line 83 of file kern_synch.c.

Referenced by loadav().

9.60.2.4 int hogticks

Definition at line 70 of file kern_synch.c.

Referenced by sched_setup(), sleepinit(), sysctl_kern_quantum(), and uiomove().

9.60.2.5 int lbolt

Definition at line 71 of file kern_synch.c.

Referenced by lboltcb(), msleep(), sched_sync(), speedup_syncer(), syncer_shutdown(), ttioctl(), ttread(), and ttwrite().

9.60.2.6 struct callout lbolt_callout [static]

Definition at line 75 of file kern_synch.c.

Referenced by lboltcb(), and synch_setup().

9.60.2.7 struct callout loadav_callout [static]

Definition at line 74 of file kern_synch.c.

Referenced by loadav(), and synch_setup().

9.60.2.8 int pause_wchan [static]

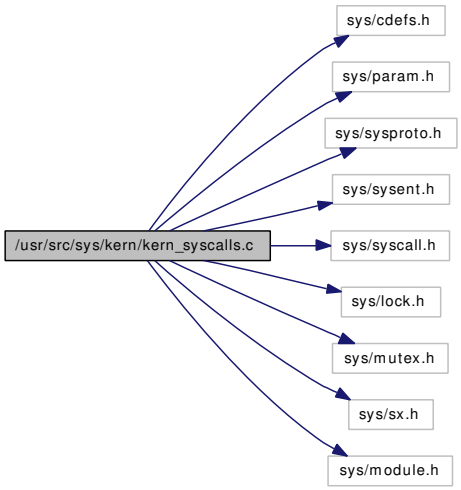
Definition at line 72 of file kern_synch.c.

Referenced by msleep(), and pause().

9.61 /usr/src/sys/kern/kern_syscalls.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/sysproto.h>
#include <sys/sysent.h>
#include <sys/syscall.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/sx.h>
#include <sys/module.h>
```

Include dependency graph for kern_syscalls.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_syscalls.c,v 1.12 2006/08/01 16:32:20 jhb Exp $")
- int lkmnosys (struct thread *td, struct nosys_args *args)
- int lkmressys (struct thread *td, struct nosys_args *args)
- int syscall_register (int *offset, struct sysent *new_sysent, struct sysent *old_sysent)
- int syscall_deregister (int *offset, struct sysent *old_sysent)
- int syscall_module_handler (struct module *mod, int what, void *arg)

9.61.1 Function Documentation

9.61.1.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_syscalls. *c*, v 1.12 2006/08/01 16:32:20 jhb Exp $")

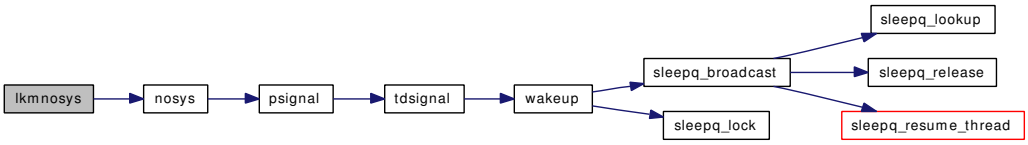
9.61.1.2 int lkmnosys (struct thread * *td*, struct nosys_args * *args*)

Definition at line 46 of file kern_syscalls.c.

References nosys().

Referenced by syscall_register().

Here is the call graph for this function:



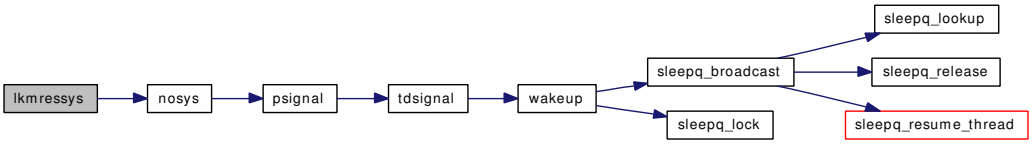
9.61.1.3 int lkmressys (struct thread * *td*, struct nosys_args * *args*)

Definition at line 52 of file kern_syscalls.c.

References nosys().

Referenced by syscall_register().

Here is the call graph for this function:



9.61.1.4 int syscall_deregister (int * *offset*, struct sysent * *old_sysent*)

Definition at line 82 of file kern_syscalls.c.

References sysent.

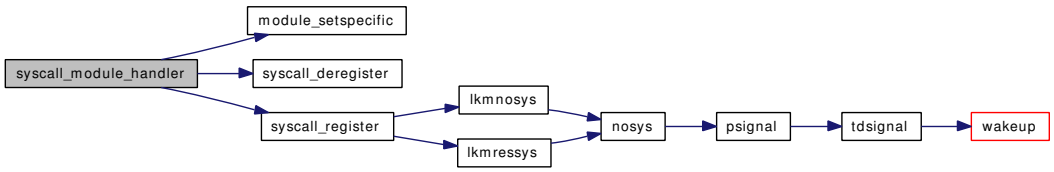
Referenced by syscall_module_handler().

9.61.1.5 int syscall_module_handler (struct module * *mod*, int *what*, void * *arg*)

Definition at line 90 of file kern_syscalls.c.

References module_setspecific(), syscall_deregister(), and syscall_register().

Here is the call graph for this function:



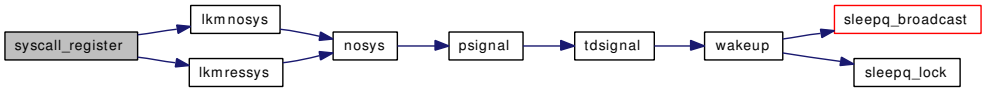
9.61.1.6 int syscall_register (int * *offset*, struct sysent * *new_sysent*, struct sysent * *old_sysent*)

Definition at line 58 of file kern_syscalls.c.

References lkmnosys(), lkmressys(), and sysent.

Referenced by syscall_module_handler().

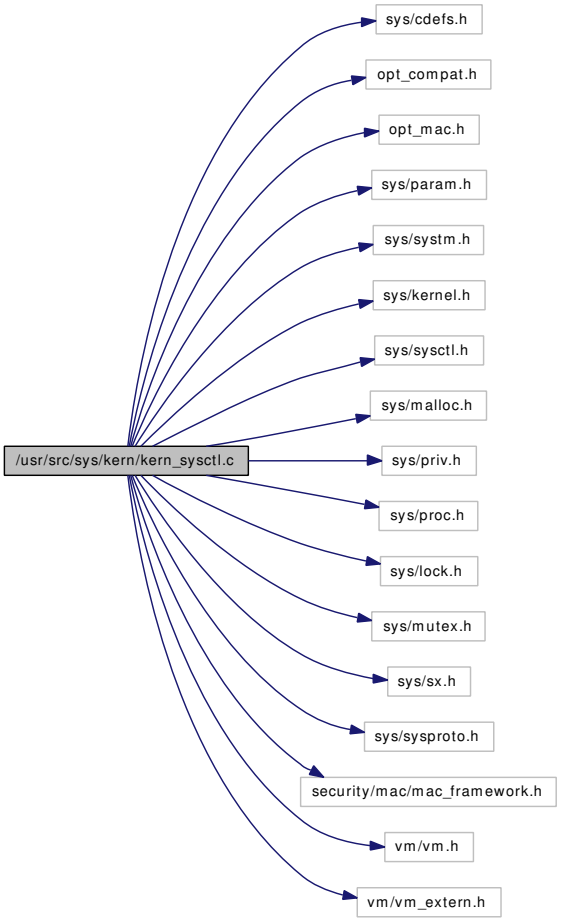
Here is the call graph for this function:



9.62 /usr/src/sys/kern/kern_sysctl.c File Reference

```
#include <sys/cdefs.h>
#include "opt_compat.h"
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/sysctl.h>
#include <sys/malloc.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/sx.h>
#include <sys/sysproto.h>
#include <security/mac/mac_framework.h>
#include <vm/vm.h>
#include <vm/vm_extern.h>
```

Include dependency graph for kern_sysctl.c:



Data Structures

- struct sysctl_args

Defines

- #define SYSCTL_LOCK() sx_xlock(&sysctllock)
- #define SYSCTL_UNLOCK() sx_xunlock(&sysctllock)
- #define SYSCTL_INIT() sx_init(&sysctllock, "sysctl lock")

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_sysctl.c,v 1.172 2006/11/06 13:42:01 rwatson Exp $")
- static MALLOC_DEFINE (M_SYSCTL,"sysctl","sysctl internal magic")
- static MALLOC_DEFINE (M_SYSCTLOID,"sysctloid","sysctl dynamic oids")
- static MALLOC_DEFINE (M_SYSCTLTMP,"sysctltmp","sysctl temp output buffer")
- static int sysctl_root (SYSCTL_HANDLER_ARGS)
- static struct sysctl_oid * sysctl_find_oidname (const char *name, struct sysctl_oid_list *list)
- void sysctl_register_oid (struct sysctl_oid *oidp)
- void sysctl_unregister_oid (struct sysctl_oid *oidp)

- int sysctl_ctx_init (struct sysctl_ctx_list *c)
- int sysctl_ctx_free (struct sysctl_ctx_list *clist)
- sysctl_ctx_entry * sysctl_ctx_entry_add (struct sysctl_ctx_list *clist, struct sysctl_oid *oidp)
- sysctl_ctx_entry * sysctl_ctx_entry_find (struct sysctl_ctx_list *clist, struct sysctl_oid *oidp)
- int sysctl_ctx_entry_del (struct sysctl_ctx_list *clist, struct sysctl_oid *oidp)
- int sysctl_remove_oid (struct sysctl_oid *oidp, int del, int recurse)
- sysctl_oid * sysctl_add_oid (struct sysctl_ctx_list *clist, struct sysctl_oid_list *parent, int number, const char *name, int kind, void *arg1, int arg2, int(*handler)(SYSCTL_HANDLER_ARGS), const char *fmt, const char *descr)
- int sysctl_move_oid (struct sysctl_oid *oid, struct sysctl_oid_list *parent)
- SET_DECLARE (sysctl_set, struct sysctl_oid)
- static void sysctl_register_all (void *arg)
- SYSINIT (sysctl, SI_SUB_KMEM, SI_ORDER_ANY, sysctl_register_all, 0)
- static int sysctl_sysctl_name (SYSCTL_HANDLER_ARGS)
- static SYSCTL_NODE (_sysctl, 1, name, CTLFLAG_RD, sysctl_sysctl_name,"")
- static int sysctl_sysctl_next_ls (struct sysctl_oid_list *lsp, int *name, u_int namelen, int *next, int *len, int level, struct sysctl_oid **oidpp)
- static int sysctl_sysctl_next (SYSCTL_HANDLER_ARGS)
- static SYSCTL_NODE (_sysctl, 2, next, CTLFLAG_RD, sysctl_sysctl_next,"")
- static int name2oid (char *name, int *oid, int *len, struct sysctl_oid **oidpp)
- static int sysctl_sysctl_name2oid (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_sysctl, 3, name2oid, CTLFLAG_RW|CTLFLAG_ANYBODY, 0, 0, sysctl_-sysctl_name2oid,"I","")
- static int sysctl_sysctl_oidfmt (SYSCTL_HANDLER_ARGS)
- static SYSCTL_NODE (_sysctl, 4, oidfmt, CTLFLAG_RD, sysctl_sysctl_oidfmt,"")
- static int sysctl_sysctl_oiddescr (SYSCTL_HANDLER_ARGS)
- static SYSCTL_NODE (_sysctl, 5, oiddescr, CTLFLAG_RD, sysctl_sysctl_oiddescr,"")
- int sysctl_handle_int (SYSCTL_HANDLER_ARGS)
- int sysctl_msec_to_ticks (SYSCTL_HANDLER_ARGS)
- int sysctl_handle_long (SYSCTL_HANDLER_ARGS)
- int sysctl_handle_string (SYSCTL_HANDLER_ARGS)
- int sysctl_handle_opaque (SYSCTL_HANDLER_ARGS)
- static int sysctl_old_kernel (struct sysctl_req *req, const void *p, size_t l)
- static int sysctl_new_kernel (struct sysctl_req *req, void *p, size_t l)
- int kernel_sysctl (struct thread *td, int *name, u_int namelen, void *old, size_t *oldlenp, void *new, size_t newlen, size_t *retval, int flags)
- int kernel_sysctlbyname (struct thread *td, char *name, void *old, size_t *oldlenp, void *new, size_t newlen, size_t *retval, int flags)
- static int sysctl_old_user (struct sysctl_req *req, const void *p, size_t l)
- static int sysctl_new_user (struct sysctl_req *req, void *p, size_t l)
- int sysctl_wire_old_buffer (struct sysctl_req *req, size_t len)
- int sysctl_find_oid (int *name, u_int namelen, struct sysctl_oid **noid, int *nindx, struct sysctl_req *req)
- int __sysctl (struct thread *td, struct sysctl_args *uap)
- int userland_sysctl (struct thread *td, int *name, u_int namelen, void *old, size_t *oldlenp, int inker-nel, void *new, size_t newlen, size_t *retval, int flags)

Variables

- static struct sx sysctllock
- sysctl_oid_list sysctl__children

9.62.1 Define Documentation

9.62.1.1 #define SYSCTL_INIT() sx_init(&sysctllock, "sysctl lock")

Definition at line 72 of file kern_sysctl.c.

Referenced by sysctl_register_all().

9.62.1.2 #define SYSCTL_LOCK() sx_xlock(&sysctllock)

Definition at line 70 of file kern_sysctl.c.

Referenced by kernel_sysctl(), and userland_sysctl().

9.62.1.3 #define SYSCTL_UNLOCK() sx_xunlock(&sysctllock)

Definition at line 71 of file kern_sysctl.c.

Referenced by kernel_sysctl(), and userland_sysctl().

9.62.2 Function Documentation

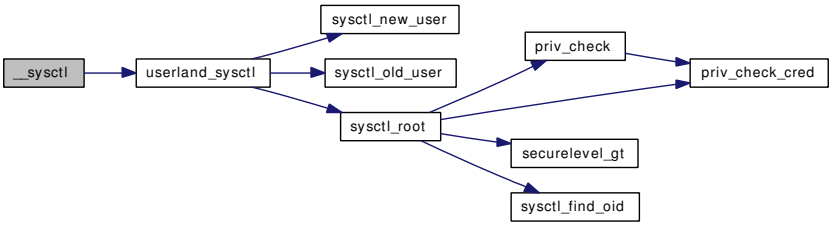
9.62.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_sysctl. *c*, v 1.172 2006/11/06 13:42:01 rwatson Exp $")

9.62.2.2 int __sysctl (struct thread * *td*, struct sysctl_args * *uap*)

Definition at line 1302 of file kern_sysctl.c.

References Giant, sysctl_args::name, sysctl_args::namelen, sysctl_args::new, sysctl_args::newlen, sysctl_-args::old, sysctl_args::oldlenp, and userland_sysctl().

Here is the call graph for this function:



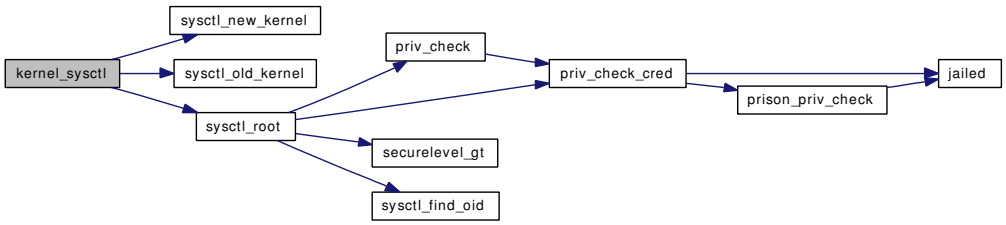
9.62.2.3 int kernel_sysctl (struct thread * *td*, int * *name*, u_int *namelen*, void * *old*, size_t * *oldlenp*, void * *new*, size_t *newlen*, size_t * *retval*, int *flags*)

Definition at line 1013 of file kern_sysctl.c.

References SYSCTL_LOCK, sysctl_new_kernel(), sysctl_old_kernel(), sysctl_root(), and SYSCTL_-UNLOCK.

Referenced by kernel_sysctlbyname().

Here is the call graph for this function:

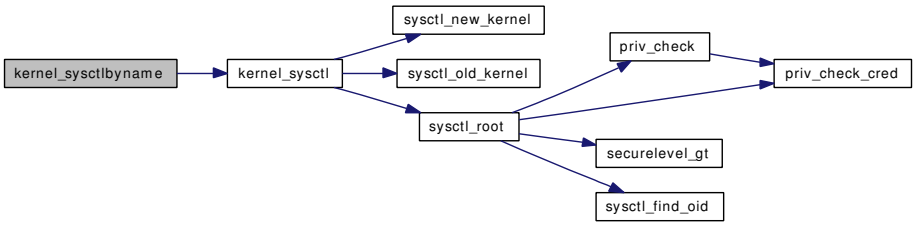


9.62.2.4 int kernel_sysctlbyname (struct thread * *td*, char * *name*, void * *old*, size_t * *oldlenp*, void * *new*, size_t *newlen*, size_t * *retval*, int *flags*)

Definition at line 1064 of file kern_sysctl.c.

References kernel_sysctl().

Here is the call graph for this function:



9.62.2.5 static MALLOC_DEFINE (M_SYSCTLTMP, "sysctltmp", "sysctl temp output buffer") [static]

9.62.2.6 static MALLOC_DEFINE (M_SYSCTLOID, "sysctloid", "sysctl dynamic oids") [static]

9.62.2.7 static MALLOC_DEFINE (M_SYSCTL, "sysctl", "sysctl internal magic") [static]

9.62.2.8 static int name2oid (char * *name*, int * *oid*, int * *len*, struct sysctl_oid ** *oidpp*) [static]

Definition at line 661 of file kern_sysctl.c.

References sysctl__children.

Referenced by sysctl_sysctl_name2oid().

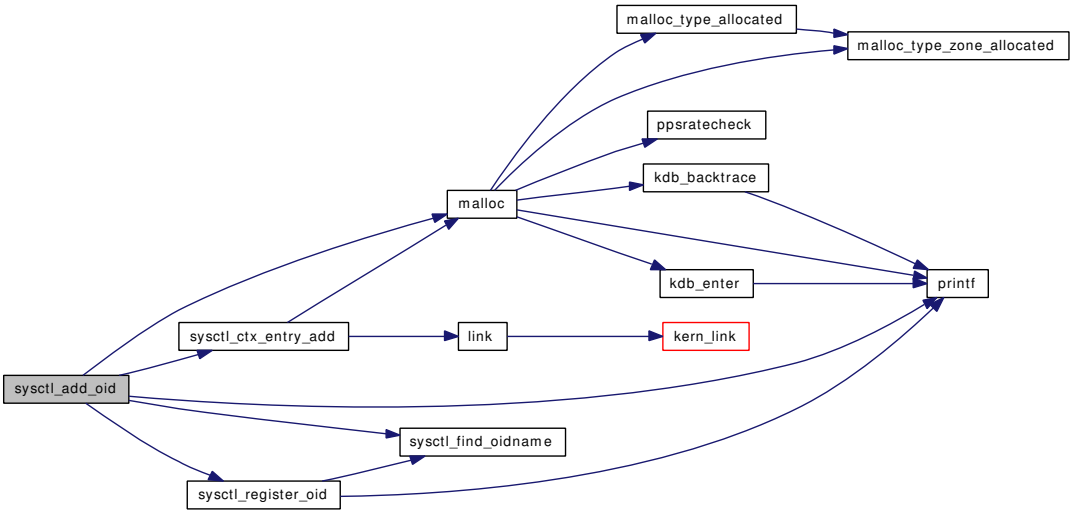
9.62.2.9 SET_DECLARE (sysctl_set, struct *sysctl_oid*)

9.62.2.10 struct sysctl_oid* sysctl_add_oid (struct sysctl_ctx_list * *clist*, struct sysctl_oid_list * *parent*, int *number*, const char * *name*, int *kind*, void * *arg1*, int *arg2*, int(*)(SYSCTL_HANDLER_ARGS) *handler*, const char * *fmt*, const char * *descr*)

Definition at line 356 of file kern_sysctl.c.

References malloc(), printf(), sysctl_ctx_entry_add(), sysctl_find_oidname(), and sysctl_register_oid().

Here is the call graph for this function:



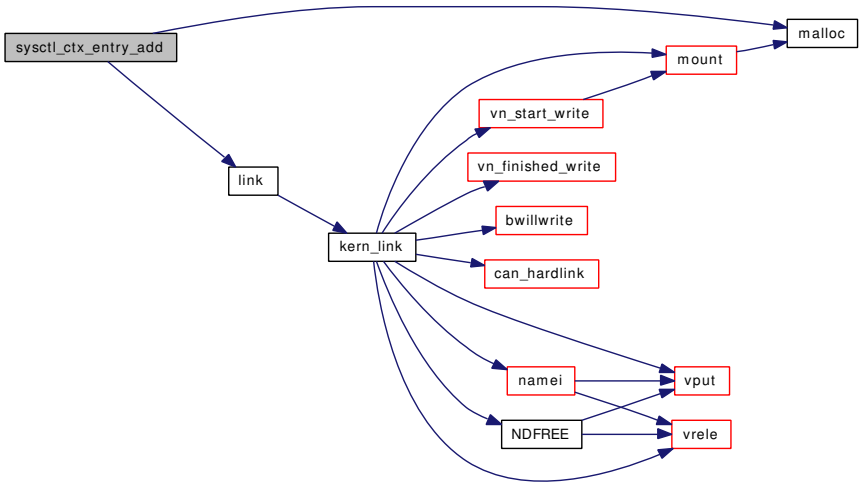
9.62.2.11 struct sysctl_ctx_entry* sysctl_ctx_entry_add (struct sysctl_ctx_list * *clist*, struct sysctl_oid * *oidp*)

Definition at line 245 of file kern_sysctl.c.

References link(), and malloc().

Referenced by sysctl_add_oid().

Here is the call graph for this function:

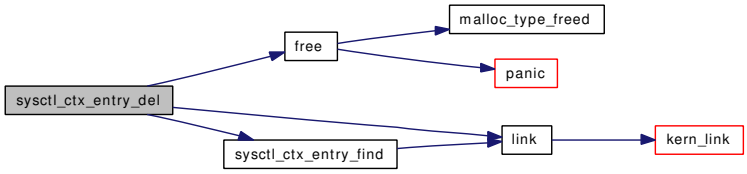


9.62.2.12 int sysctl_ctx_entry_del (struct sysctl_ctx_list * *clist*, struct sysctl_oid * *oidp*)

Definition at line 278 of file kern_sysctl.c.

References free(), link(), and sysctl_ctx_entry_find().

Here is the call graph for this function:



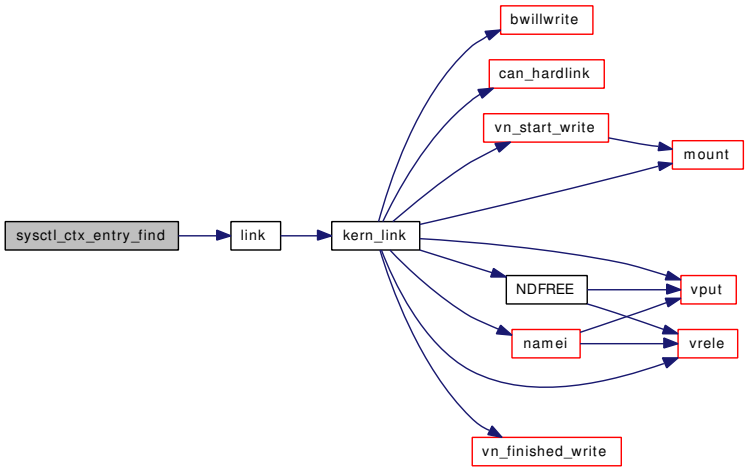
9.62.2.13 struct sysctl_ctx_entry* sysctl_ctx_entry_find (struct sysctl_ctx_list * *clist*, struct sysctl_oid * *oidp*)

Definition at line 259 of file kern_sysctl.c.

References link().

Referenced by sysctl_ctx_entry_del().

Here is the call graph for this function:



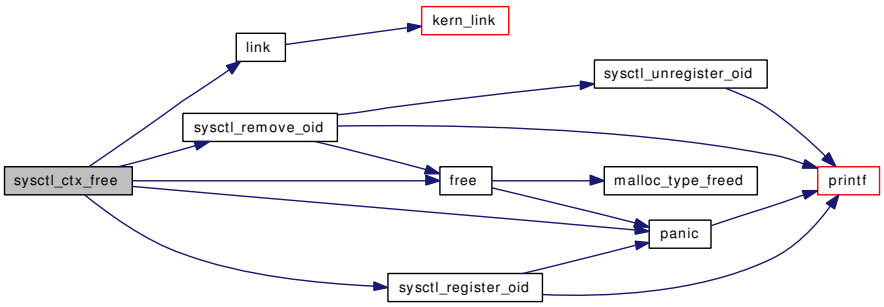
9.62.2.14 int sysctl_ctx_free (struct sysctl_ctx_list * *clist*)

Definition at line 197 of file kern_sysctl.c.

References free(), link(), panic(), sysctl_register_oid(), and sysctl_remove_oid().

Referenced by cpufreq_detach(), and device_sysctl_fini().

Here is the call graph for this function:



9.62.2.15 int sysctl_ctx_init (struct sysctl_ctx_list * *c*)

Definition at line 185 of file kern_sysctl.c.

Referenced by cpufreq_attach(), devclass_sysctl_init(), and device_sysctl_init().

9.62.2.16 int sysctl_find_oid (int * *name*, u_int *namelen*, struct sysctl_oid ** *noid*, int * *nindx*, struct sysctl_req * *req*)

Definition at line 1178 of file kern_sysctl.c.

References sysctl__children.

Referenced by sysctl_root(), sysctl_sysctl_oiddescr(), and sysctl_sysctl_oidfmt().

9.62.2.17 static struct sysctl_oid* sysctl_find_oidname (const char * *name*, struct sysctl_oid_list * *list*) [static]

Definition at line 79 of file kern_sysctl.c.

Referenced by sysctl_add_oid(), sysctl_move_oid(), and sysctl_register_oid().

9.62.2.18 int sysctl_handle_int (SYSCTL_HANDLER_ARGS)

Definition at line 805 of file kern_sysctl.c.

Referenced by cpufreq_curr_sysctl(), kdb_sysctl_enter(), kdb_sysctl_panic(), kdb_sysctl_trap(), kdb_sysctl_trap_code(), poll_burst_max_sysctl(), poll_each_burst_sysctl(), poll_switch(), reg_frac_sysctl(), sysctl_debug_ktr_clear(), sysctl_debug_witness_watch(), sysctl_devctl_disable(), sysctl_kern_consmute(), sysctl_kern_msgbuf_clear(), sysctl_kern_quantum(), sysctl_kern_randompid(), sysctl_kern_securelvl(), sysctl_kern_timecounter_freq(), sysctl_kern_timecounter_get(), sysctl_machdep_adjkerntz(), sysctl_maxsockets(), sysctl_msec_to_ticks(), sysctl_nmbclusters(), sysctl_somaxconn(), TAILQ_HEAD(), and user_frac_sysctl().

9.62.2.19 int sysctl_handle_long (SYSCTL_HANDLER_ARGS)

Definition at line 859 of file kern_sysctl.c.

Referenced by sysctl_handle_sb_max(), sysctl_hw_physmem(), sysctl_hw_realmem(), and sysctl_hw_usermem().

9.62.2.20 int sysctl_handle_opaque (SYSCTL_HANDLER_ARGS)

Definition at line 944 of file kern_sysctl.c.

Referenced by ntp_sysctl(), sysctl_intrcnt(), sysctl_intrnames(), sysctl_kern_clockrate(), and sysctl_kern_-msgbuf().

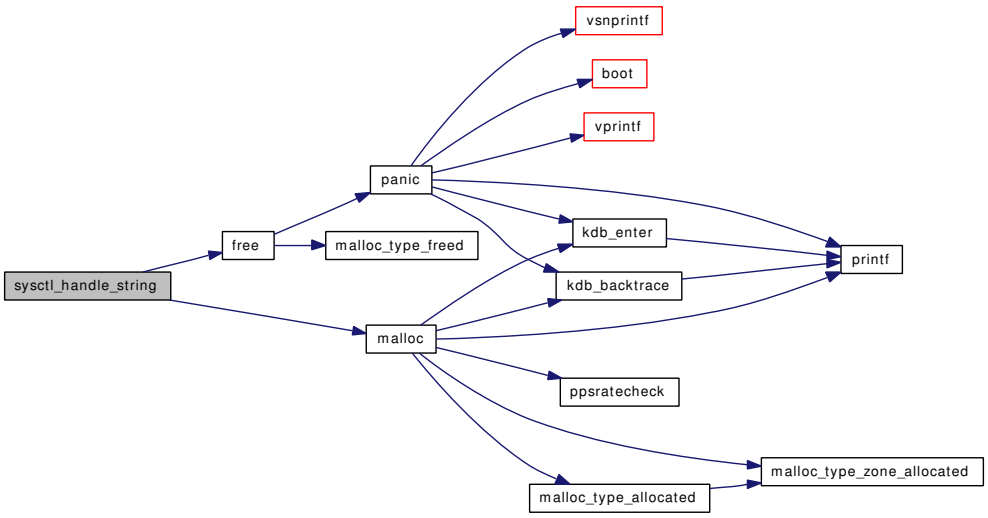
9.62.2.21 int sysctl_handle_string (SYSCTL_HANDLER_ARGS)

Definition at line 902 of file kern_sysctl.c.

References free(), and malloc().

Referenced by cpufreq_levels_sysctl(), cpufreq_settings_sysctl(), kdb_sysctl_available(), kdb_sysctl_-current(), sysctl_hostname(), sysctl_kern_console(), sysctl_kern_proc_sv_name(), and sysctl_kern_-timecounter_hardware().

Here is the call graph for this function:

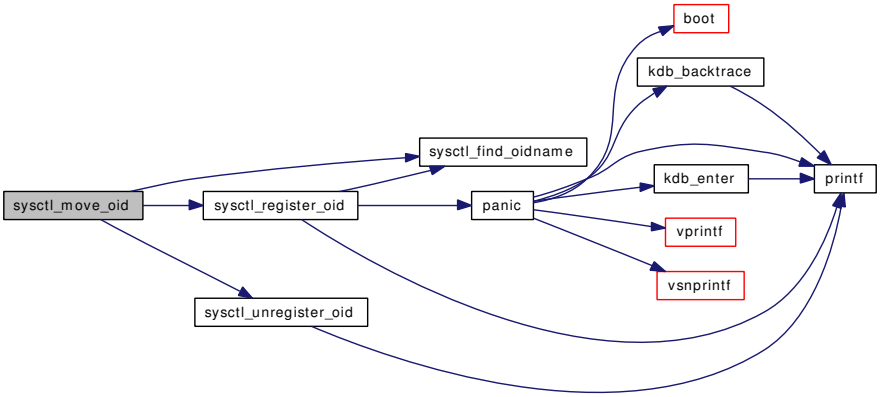


9.62.2.22 int sysctl_move_oid (struct sysctl_oid ∗ *oid*, struct sysctl_oid_list ∗ *parent*)

Definition at line 421 of file kern_sysctl.c.

References sysctl_find_oidname(), sysctl_register_oid(), and sysctl_unregister_oid().

Here is the call graph for this function:



9.62.2.23 int sysctl_msec_to_ticks (SYSCTL_HANDLER_ARGS)

Definition at line 834 of file kern_sysctl.c.

References hz, and sysctl_handle_int().

Here is the call graph for this function:

9.62.2.24 static int sysctl_new_kernel (struct sysctl_req * *req*, void * *p*, size_t *l*) [static]

Definition at line 1001 of file kern_sysctl.c.

Referenced by kernel_sysctl().

9.62.2.25 static int sysctl_new_user (struct sysctl_req * *req*, void * *p*, size_t *l*) [static]

Definition at line 1123 of file kern_sysctl.c.

Referenced by userland_sysctl().

9.62.2.26 static SYSCTL_NODE (_sysctl, 5, oiddescr, CTLFLAG_RD, sysctl_sysctl_oiddescr, "") [static]

9.62.2.27 static SYSCTL_NODE (_sysctl, 4, oidfmt, CTLFLAG_RD, sysctl_sysctl_oidfmt, "") [static]

9.62.2.28 static SYSCTL_NODE (_sysctl, 2, next, CTLFLAG_RD, sysctl_sysctl_next, "") [static]

9.62.2.29 static SYSCTL_NODE (_sysctl, 1, name, CTLFLAG_RD, sysctl_sysctl_name, "") [static]

9.62.2.30 static int sysctl_old_kernel (struct sysctl_req * *req*, const void * *p*, size_t *l*) [static]

Definition at line 980 of file kern_sysctl.c.

Referenced by kernel_sysctl().

9.62.2.31 static int sysctl_old_user (struct sysctl_req * *req*, const void * *p*, size_t *l*) [static]

Definition at line 1089 of file kern_sysctl.c.

Referenced by sysctl_wire_old_buffer(), and userland_sysctl().

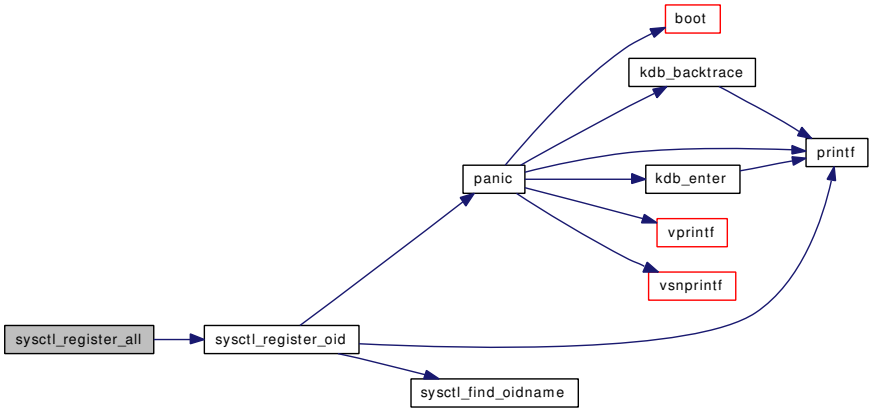
9.62.2.32 SYSCTL_PROC (_sysctl, 3, name2oid, CTLFLAG_RW| *CTLFLAG_ANYBODY*, 0, 0, sysctl_sysctl_name2oid, "I", "")

9.62.2.33 static void sysctl_register_all (void * *arg*) [static]

Definition at line 443 of file kern_sysctl.c.

References SYSCTL_INIT, and sysctl_register_oid().

Here is the call graph for this function:



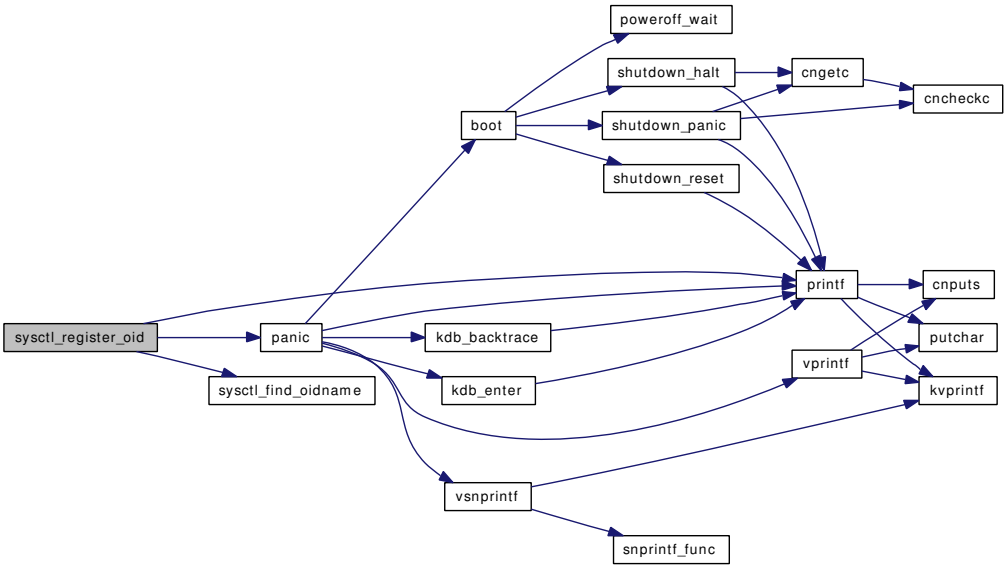
9.62.2.34 void sysctl_register_oid (struct sysctl_oid * *oidp*)

Definition at line 98 of file kern_sysctl.c.

References panic(), printf(), and sysctl_find_oidname().

Referenced by linker_file_register_sysctls(), sysctl_add_oid(), sysctl_ctx_free(), sysctl_move_oid(), sysctl_register_all(), and vfs_register().

Here is the call graph for this function:



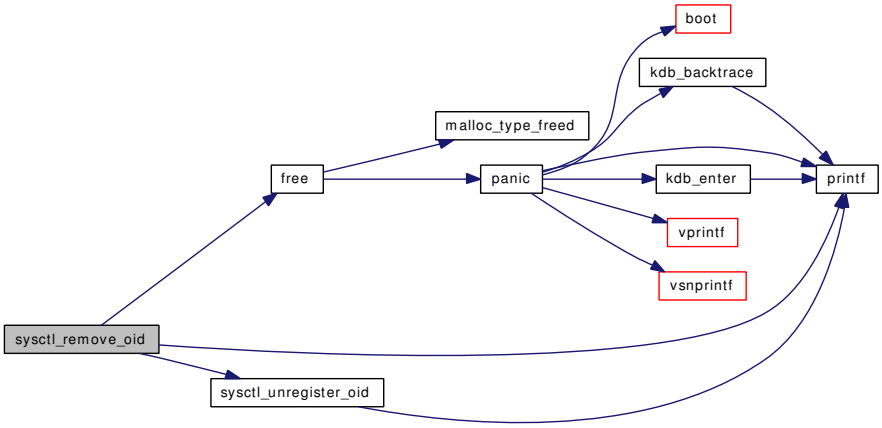
9.62.2.35 int sysctl_remove_oid (struct sysctl_oid * *oidp*, int *del*, int *recurse*)

Definition at line 300 of file kern_sysctl.c.

References free(), printf(), and sysctl_unregister_oid().

Referenced by sysctl_ctx_free().

Here is the call graph for this function:



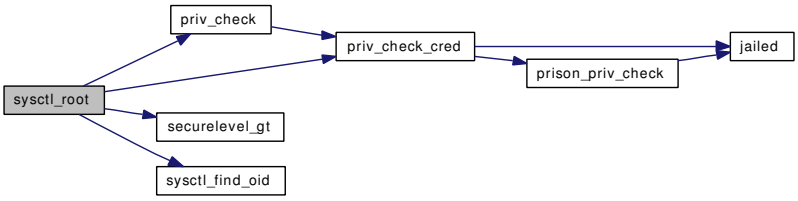
9.62.2.36 static int sysctl_root (SYSCTL_HANDLER_ARGS) [static]

Definition at line 1222 of file kern_sysctl.c.

References priv_check(), priv_check_cred(), securelevel_gt(), and sysctl_find_oid().

Referenced by kernel_sysctl(), and userland_sysctl().

Here is the call graph for this function:

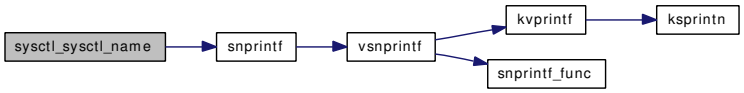


9.62.2.37 static int sysctl_sysctl_name (SYSCTL_HANDLER_ARGS) [static]

Definition at line 528 of file kern_sysctl.c.

References buf, snprintf(), and sysctl__children.

Here is the call graph for this function:

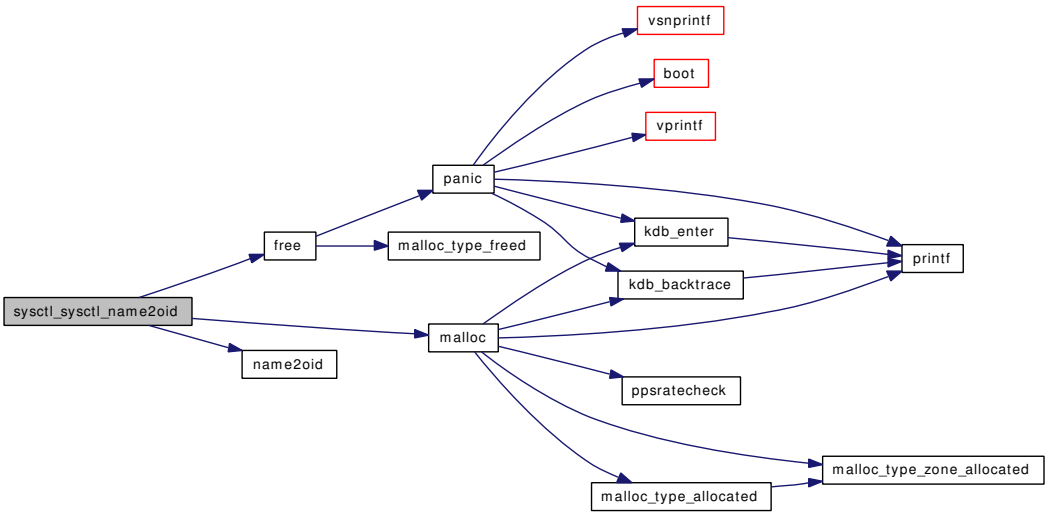


9.62.2.38 static int sysctl_sysctl_name2oid (SYSCTL_HANDLER_ARGS) [static]

Definition at line 718 of file kern_sysctl.c.

References free(), malloc(), and name2oid().

Here is the call graph for this function:

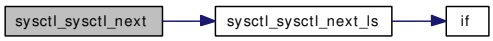


9.62.2.39 static int sysctl_sysctl_next (SYSCTL_HANDLER_ARGS) [static]

Definition at line 642 of file kern_sysctl.c.

References sysctl__children, and sysctl_sysctl_next_ls().

Here is the call graph for this function:



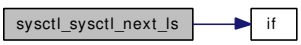
9.62.2.40 static int sysctl_sysctl_next_ls (struct sysctl_oid_list * *lsp*, int * *name*, u_int *namelen*, int * *next*, int * *len*, int *level*, struct sysctl_oid ** *oidpp*) [static]

Definition at line 583 of file kern_sysctl.c.

References if().

Referenced by sysctl_sysctl_next().

Here is the call graph for this function:



9.62.2.41 static int sysctl_sysctl_oiddescr (SYSCTL_HANDLER_ARGS) [static]

Definition at line 776 of file kern_sysctl.c.

References sysctl_find_oid().

Here is the call graph for this function:

9.62.2.42 static int sysctl_sysctl_oidfmt (SYSCTL_HANDLER_ARGS) [static]

Definition at line 754 of file kern_sysctl.c.

References sysctl_find_oid().

Here is the call graph for this function:

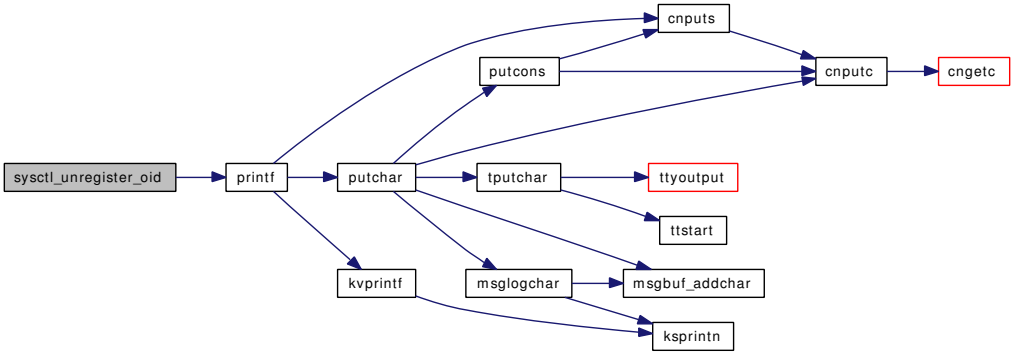
9.62.2.43 void sysctl_unregister_oid (struct sysctl_oid * *oidp*)

Definition at line 155 of file kern_sysctl.c.

References printf().

Referenced by linker_file_unregister_sysctls(), sysctl_move_oid(), sysctl_remove_oid(), and vfs_-register().

Here is the call graph for this function:



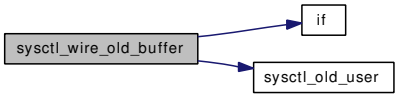
9.62.2.44 int sysctl_wire_old_buffer (struct sysctl_req * *req*, size_t *len*)

Definition at line 1143 of file kern_sysctl.c.

References dummy, if(), ret, and sysctl_old_user().

Referenced by kdb_sysctl_enter(), kdb_sysctl_panic(), kdb_sysctl_trap(), kdb_sysctl_trap_code(), sysctl_-kern_file(), sysctl_kern_function_list(), sysctl_kern_proc(), and sysctl_kern_randompid().

Here is the call graph for this function:



9.62.2.45 SYSINIT (sysctl, SI_SUB_KMEM, SI_ORDER_ANY, sysctl_register_all, 0)

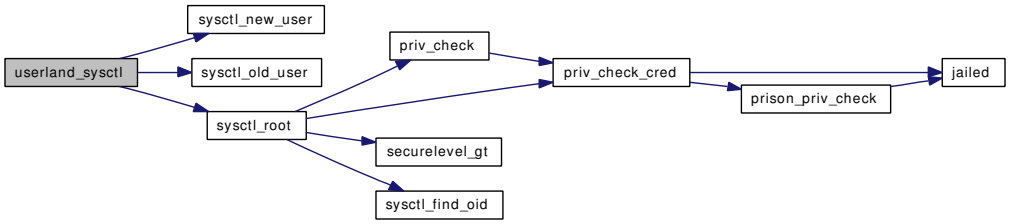
9.62.2.46 int userland_sysctl (struct thread * *td*, int * *name*, u_int *namelen*, void * *old*, size_t * *oldlenp*, int *inkernel*, void * *new*, size_t *newlen*, size_t * *retval*, int *flags*)

Definition at line 1336 of file kern_sysctl.c.

References SYSCTL_LOCK, sysctl_new_user(), sysctl_old_user(), sysctl_root(), and SYSCTL_-UNLOCK.

Referenced by __sysctl(), and uname().

Here is the call graph for this function:



9.62.3 Variable Documentation

9.62.3.1 struct sysctl_oid_list sysctl__children

Definition at line 76 of file kern_sysctl.c.

Referenced by name2oid(), sysctl_find_oid(), sysctl_sysctl_name(), and sysctl_sysctl_next().

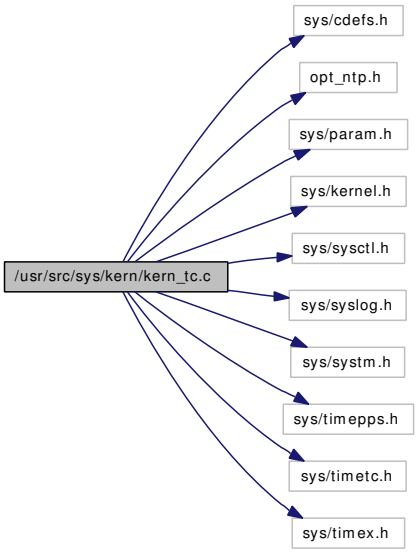
9.62.3.2 struct sx sysctllock [static]

Definition at line 68 of file kern_sysctl.c.

9.63 /usr/src/sys/kern/kern_tc.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ntp.h"
#include <sys/param.h>
#include <sys/kernel.h>
#include <sys/sysctl.h>
#include <sys/syslog.h>
#include <sys/systm.h>
#include <sys/timepps.h>
#include <sys/timetc.h>
#include <sys/timex.h>
```

Include dependency graph for kern_tc.c:



Data Structures

- struct timehands

Defines

- #define LARGE_STEP 200
- #define TC_STATS(foo)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_tc.c,v 1.177 2006/08/04 07:56:35 yar Exp $")
- static u_int dummy_get_timecount (struct timecounter *tc)

- static int sysctl_kern_boottime (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern, KERN_BOOTTIME, boottime, CTLTYPE_STRUCT|CTLFLAG_RD, NULL, 0, sysctl_kern_boottime,"S,timeval","System boottime")
- SYSCTL_NODE (_kern, OID_AUTO, timecounter, CTLFLAG_RW, 0,"")
- SYSCTL_NODE (_kern_timecounter, OID_AUTO, tc, CTLFLAG_RW, 0,"")
- SYSCTL_INT (_kern_timecounter, OID_AUTO, stepwarnings, CTLFLAG_RW,&timestepwarnings, 0,"")
- TC_STATS (nbinuptime)
- TC_STATS (nnanouptime)
- TC_STATS (nmicrouptime)
- TC_STATS (nbintime)
- TC_STATS (nnanotime)
- TC_STATS (nmicrotime)
- TC_STATS (ngetbinuptime)
- TC_STATS (ngetnanouptime)
- TC_STATS (ngetmicrouptime)
- TC_STATS (ngetbintime)
- TC_STATS (ngetnanotime)
- TC_STATS (ngetmicrotime)
- TC_STATS (nsetclock)
- static void tc_windup (void)
- static void cpu_tick_calibrate (int)
- static int sysctl_kern_timecounter_get (SYSCTL_HANDLER_ARGS)
- static int sysctl_kern_timecounter_freq (SYSCTL_HANDLER_ARGS)
- static __inline u_int tc_delta (struct timehands *th)
- void binuptime (struct bintime *bt)
- void nanouptime (struct timespec *tsp)
- void microuptime (struct timeval *tvp)
- void bintime (struct bintime *bt)
- void nanotime (struct timespec *tsp)
- void microtime (struct timeval *tvp)
- void getbinuptime (struct bintime *bt)
- void getnanouptime (struct timespec *tsp)
- void getmicrouptime (struct timeval *tvp)
- void getbintime (struct bintime *bt)
- void getnanotime (struct timespec *tsp)
- void getmicrotime (struct timeval *tvp)
- void tc_init (struct timecounter *tc)
- u_int64_t tc_getfrequency (void)
- void tc_setclock (struct timespec *ts)
- static int sysctl_kern_timecounter_hardware (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern_timecounter, OID_AUTO, hardware, CTLTYPE_STRING|CTLFLAG_RW, 0, 0, sysctl_kern_timecounter_hardware,"A","")
- static int sysctl_kern_timecounter_choice (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern_timecounter, OID_AUTO, choice, CTLTYPE_STRING|CTLFLAG_RD, 0, 0, sysctl_kern_timecounter_choice,"A","")
- int pps_ioctl (u_long cmd, caddr_t data, struct pps_state *pps)
- void pps_init (struct pps_state *pps)
- void pps_capture (struct pps_state *pps)
- void pps_event (struct pps_state *pps, int event)

- SYSCTL_INT (_kern_timecounter, OID_AUTO, tick, CTLFLAG_RD,&tc_tick, 0,"")
- void tc_ticktock (void)
- static void inittimecounter (void *dummy)
- static uint64_t tc_cpu_ticks (void)
- void set_cputicker (cpu_tick_f *func, uint64_t freq, unsigned var)
- uint64_t cpu_tickrate (void)
- uint64_t cputick2usec (uint64_t tick)

Variables

- static struct timecounter dummy_timecounter
- static struct timehands th0
- static struct timehands th9 = { NULL, 0, 0, 0, {0, 0}, {0, 0}, {0, 0}, 0, &th0}
- static struct timehands th8 = { NULL, 0, 0, 0, {0, 0}, {0, 0}, {0, 0}, 0, &th9}
- static struct timehands th7 = { NULL, 0, 0, 0, {0, 0}, {0, 0}, {0, 0}, 0, &th8}
- static struct timehands th6 = { NULL, 0, 0, 0, {0, 0}, {0, 0}, {0, 0}, 0, &th7}
- static struct timehands th5 = { NULL, 0, 0, 0, {0, 0}, {0, 0}, {0, 0}, 0, &th6}
- static struct timehands th4 = { NULL, 0, 0, 0, {0, 0}, {0, 0}, {0, 0}, 0, &th5}
- static struct timehands th3 = { NULL, 0, 0, 0, {0, 0}, {0, 0}, {0, 0}, 0, &th4}
- static struct timehands th2 = { NULL, 0, 0, 0, {0, 0}, {0, 0}, {0, 0}, 0, &th3}
- static struct timehands th1 = { NULL, 0, 0, 0, {0, 0}, {0, 0}, {0, 0}, 0, &th2}
- static struct timehands th0
- static struct timehands *volatile timehands = &th0
- timecounter * timecounter = &dummy_timecounter
- static struct timecounter * timecounters = &dummy_timecounter
- time_t time_second = 1
- time_t time_uptime = 1
- static struct bintime boottimebin
- timeval boottime
- static int timestepwarnings
- static int tc_tick
- static int cpu_tick_variable
- static uint64_t cpu_tick_frequency
- cpu_tick_f * cpu_ticks = tc_cpu_ticks

9.63.1 Define Documentation

9.63.1.1 #define LARGE_STEP 200

Definition at line 30 of file kern_tc.c.

Referenced by tc_windup().

9.63.1.2 #define TC_STATS(foo)

Value:

```
static u_int foo; \
        SYSCTL_UINT(_kern_timecounter, OID_AUTO, foo, CTLFLAG_RD, &foo, 0, "");\
        struct __hack
```

Definition at line 106 of file kern_tc.c.

9.63.2 Function Documentation

9.63.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_tc. *c*, v 1.177 2006/08/04 07:56:35 yar Exp $")

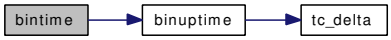
9.63.2.2 void bintime (struct bintime * *bt*)

Definition at line 213 of file kern_tc.c.

References binuptime().

Referenced by cpu_tick_calibrate(), devstat_end_transaction(), init_va_filerev(), microtime(), microuptime(), nanotime(), nanouptime(), pps_event(), sigexit(), softclock(), tc_setclock(), tc_windup(), and uuid_time().

Here is the call graph for this function:



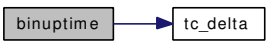
9.63.2.3 void binuptime (struct bintime * *bt*)

Definition at line 178 of file kern_tc.c.

References tc_delta(), timehands::th_generation, timehands::th_offset, timehands::th_scale, and timehands.

Referenced by bintime(), devstat_add_entry(), devstat_end_transaction(), devstat_new_entry(), devstat_start_transaction(), devstat_start_transaction_bio(), microuptime(), nanouptime(), softclock(), and tc_setclock().

Here is the call graph for this function:



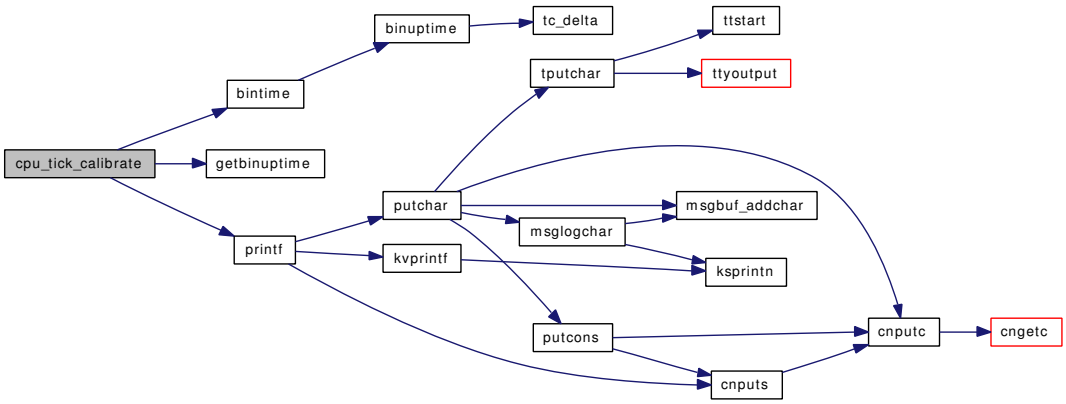
9.63.2.4 static void cpu_tick_calibrate (int) [static]

Definition at line 868 of file kern_tc.c.

References bintime(), bootverbose, cpu_tick_frequency, cpu_tick_variable, cpu_ticks, getbinuptime(), and printf().

Referenced by tc_setclock(), and tc_ticktock().

Here is the call graph for this function:



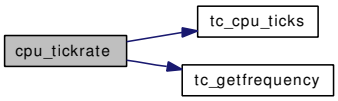
9.63.2.5 uint64_t cpu_tickrate (void)

Definition at line 950 of file kern_tc.c.

References cpu_tick_frequency, cpu_ticks, tc_cpu_ticks(), and tc_getfrequency().

Referenced by ast(), cputick2usec(), and mi_switch().

Here is the call graph for this function:



9.63.2.6 uint64_t cputick2usec (uint64_t *tick*)

Definition at line 969 of file kern_tc.c.

References cpu_tickrate().

Referenced by calcru1(), fill_kinfo_proc_only(), and sched_timestamp().

Here is the call graph for this function:

9.63.2.7 static u_int dummy_get_timecount (struct timecounter * *tc*) [static]

Definition at line 39 of file kern_tc.c.

9.63.2.8 void getbintime (struct bintime * *bt*)

Definition at line 284 of file kern_tc.c.

References timehands::th_generation, timehands::th_offset, and timehands.

Referenced by sigexit().

9.63.2.9 void getbinuptime (struct bintime * *bt*)

Definition at line 242 of file kern_tc.c.

References timehands::th_generation, timehands::th_offset, and timehands.

Referenced by cpu_tick_calibrate(), and init_va_filerev().

9.63.2.10 void getmicrotime (struct timeval * *tvp*)

Definition at line 313 of file kern_tc.c.

References timehands::th_generation, timehands::th_microtime, and timehands.

Referenced by ttread().

9.63.2.11 void getmicrouptime (struct timeval * *tvp*)

Definition at line 270 of file kern_tc.c.

References timehands::th_generation, timehands::th_offset, and timehands.

Referenced by kern_getitimer(), kern_select(), kern_setitimer(), kqueue_scan(), poll(), ratecheck(), and realitexpire().

9.63.2.12 void getnanotime (struct timespec * *tsp*)

Definition at line 299 of file kern_tc.c.

References timehands::th_generation, timehands::th_nanotime, and timehands.

Referenced by kern_clock_gettime(), kern_sem_wait(), mqfs_create_node(), mqueue_receive(), mqueue_send(), realtimer_clocktime(), resettodr(), and vfs_timestamp().

9.63.2.13 void getnanouptime (struct timespec * *tsp*)

Definition at line 256 of file kern_tc.c.

References timehands::th_generation, timehands::th_offset, and timehands.

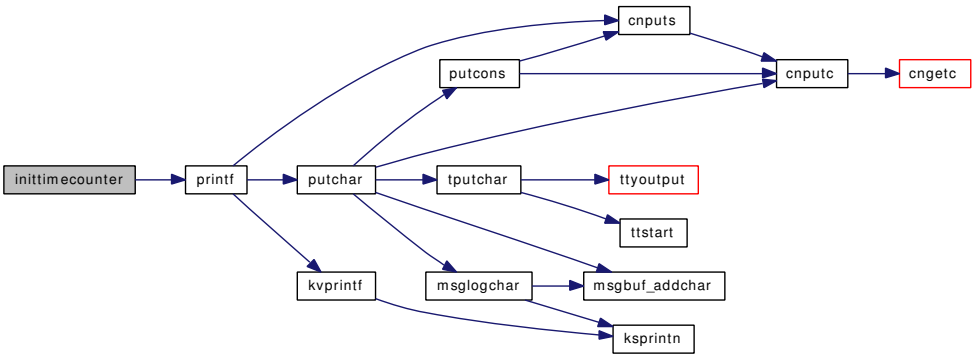
Referenced by do_cv_wait(), do_lock_umtx(), do_lock_umutex(), do_wait(), kern_clock_gettime(), kern_nanosleep(), kern_sigtimedwait(), print_uptime(), and realtimer_clocktime().

9.63.2.14 static void inittimecounter (void * *dummy*) [static]

Definition at line 812 of file kern_tc.c.

References hz, and printf().

Here is the call graph for this function:



9.63.2.15 void microtime (struct timeval * *tvp*)

Definition at line 232 of file kern_tc.c.

References bintime().

Referenced by gettimeofday(), getutimes(), settime(), unp_internalize(), and vfs_timestamp().

Here is the call graph for this function:

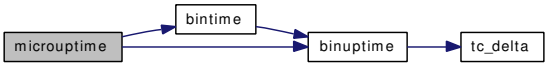
9.63.2.16 void microuptime (struct timeval * *tvp*)

Definition at line 203 of file kern_tc.c.

References bintime(), and binuptime().

Referenced by acct_process(), fork1(), hardclock_device_poll(), netisr_poll(), netisr_pollmore(), and proc0_post().

Here is the call graph for this function:



9.63.2.17 void nanotime (struct timespec * *tsp*)

Definition at line 222 of file kern_tc.c.

References bintime().

Referenced by kern_clock_gettime(), ntp_gettime1(), proc0_post(), tc_setclock(), and vfs_timestamp().

Here is the call graph for this function:

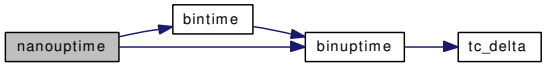
9.63.2.18 void nanouptime (struct timespec * *tsp*)

Definition at line 193 of file kern_tc.c.

References bintime(), and binuptime().

Referenced by kern_clock_gettime().

Here is the call graph for this function:



9.63.2.19 void pps_capture (struct pps_state * *pps*)

Definition at line 683 of file kern_tc.c.

References timehands::th_counter, timehands::th_generation, and timehands.

9.63.2.20 void pps_event (struct pps_state * *pps*, int *event*)

Definition at line 697 of file kern_tc.c.

References bintime().

Here is the call graph for this function:

9.63.2.21 void pps_init (struct pps_state * *pps*)

Definition at line 673 of file kern_tc.c.

9.63.2.22 int pps_ioctl (u_long *cmd*, caddr_t *data*, struct pps_state * *pps*)

Definition at line 615 of file kern_tc.c.

Referenced by ttioctl().

9.63.2.23 void set_cputicker (cpu_tick_f * *func*, uint64_t *freq*, unsigned *var*)

Definition at line 937 of file kern_tc.c.

References cpu_tick_frequency, cpu_tick_variable, cpu_ticks, and tc_cpu_ticks().

Here is the call graph for this function:

9.63.2.24 SYSCTL_INT (_kern_timecounter, OID_AUTO, tick, CTLFLAG_RD, & *tc_tick*, 0, "")

9.63.2.25 SYSCTL_INT (_kern_timecounter, OID_AUTO, stepwarnings, CTLFLAG_RW, & *timestepwarnings*, 0, "")

9.63.2.26 static int sysctl_kern_boottime (SYSCTL_HANDLER_ARGS) [static]

Definition at line 123 of file kern_tc.c.

9.63.2.27 static int sysctl_kern_timecounter_choice (SYSCTL_HANDLER_ARGS) [static]

Definition at line 590 of file kern_tc.c.

References buf, and sprintf().

Here is the call graph for this function:

9.63.2.28 static int sysctl_kern_timecounter_freq (SYSCTL_HANDLER_ARGS) [static]

Definition at line 148 of file kern_tc.c.

References sysctl_handle_int().

Referenced by tc_init().

Here is the call graph for this function:

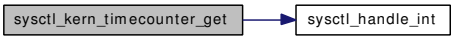
9.63.2.29 static int sysctl_kern_timecounter_get (SYSCTL_HANDLER_ARGS) [static]

Definition at line 138 of file kern_tc.c.

References sysctl_handle_int().

Referenced by tc_init().

Here is the call graph for this function:

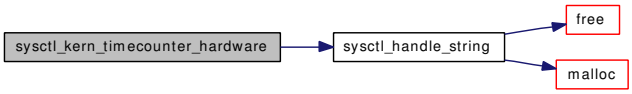


9.63.2.30 static int sysctl_kern_timecounter_hardware (SYSCTL_HANDLER_ARGS) [static]

Definition at line 557 of file kern_tc.c.

References sysctl_handle_string().

Here is the call graph for this function:



9.63.2.31 SYSCTL_NODE (_kern_timecounter, OID_AUTO, tc, CTLFLAG_RW, 0, "")

9.63.2.32 SYSCTL_NODE (_kern, OID_AUTO, timecounter, CTLFLAG_RW, 0, "")

9.63.2.33 SYSCTL_PROC (_kern_timecounter, OID_AUTO, choice, CTLTYPE_STRING| *CTLFLAG_RD*, 0, 0, sysctl_kern_timecounter_choice, "A", "")

9.63.2.34 SYSCTL_PROC (_kern_timecounter, OID_AUTO, hardware, CTLTYPE_STRING| *CTLFLAG_RW*, 0, 0, sysctl_kern_timecounter_hardware, "A", "")

9.63.2.35 SYSCTL_PROC (_kern, KERN_BOOTTIME, boottime, CTLTYPE_STRUCT| *CTLFLAG_RD*, NULL, 0, sysctl_kern_boottime, " *S*, timeval", "System boottime")

9.63.2.36 static uint64_t tc_cpu_ticks (void) [static]

Definition at line 844 of file kern_tc.c.

References timehands::th_counter, and timehands.

Referenced by cpu_tickrate(), and set_cputicker().

9.63.2.37 static __inline u_int tc_delta (struct timehands * *th*) [static]

Definition at line 162 of file kern_tc.c.

References timehands::th_counter, and timehands::th_offset_count.

Referenced by binuptime(), and tc_windup().

9.63.2.38 u_int64_t tc_getfrequency (void)

Definition at line 391 of file kern_tc.c.

References timehands::th_counter, and timehands.

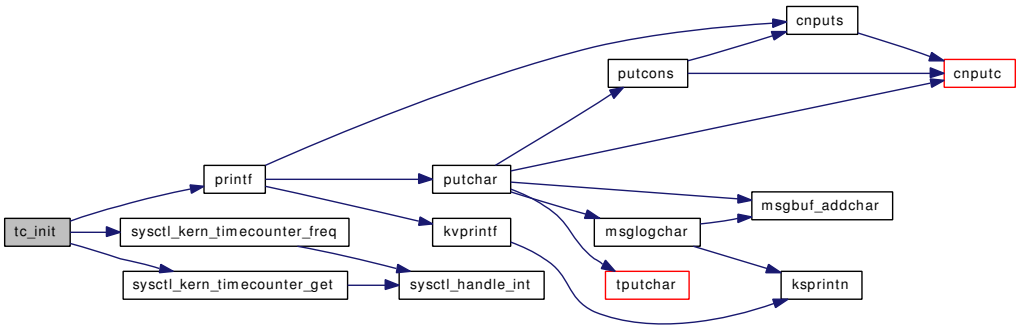
Referenced by cpu_tickrate(), and kern_clock_getres().

9.63.2.39 void tc_init (struct timecounter * *tc*)

Definition at line 330 of file kern_tc.c.

References bootverbose, hz, printf(), sysctl_kern_timecounter_freq(), and sysctl_kern_timecounter_get().

Here is the call graph for this function:



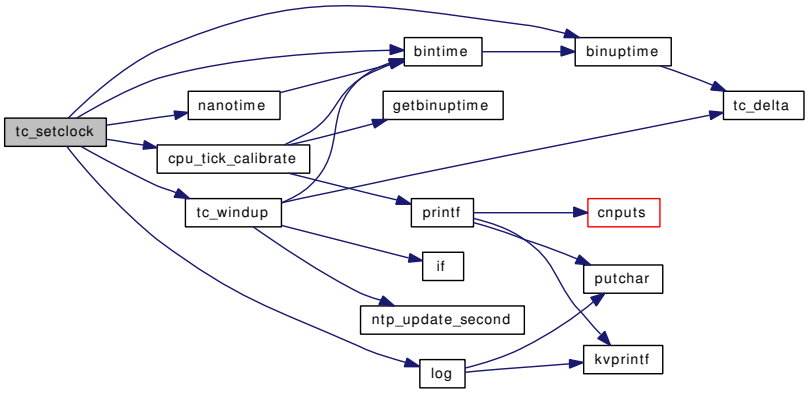
9.63.2.40 void tc_setclock (struct timespec * *ts*)

Definition at line 403 of file kern_tc.c.

References bintime(), binuptime(), cpu_tick_calibrate(), log(), nanotime(), and tc_windup().

Referenced by inittodr(), and settime().

Here is the call graph for this function:



9.63.2.41 TC_STATS (nsetclock)

9.63.2.42 TC_STATS (ngetmicrotime)

9.63.2.43 TC_STATS (ngetnanotime)

9.63.2.44 TC_STATS (ngetbintime)

9.63.2.45 TC_STATS (ngetmicrouptime)

9.63.2.46 TC_STATS (ngetnanouptime)

9.63.2.47 TC_STATS (ngetbinuptime)

9.63.2.48 TC_STATS (nmicrotime)

9.63.2.49 TC_STATS (nnanotime)

9.63.2.50 TC_STATS (nbintime)

9.63.2.51 TC_STATS (nmicrouptime)

9.63.2.52 TC_STATS (nnanouptime)

9.63.2.53 TC_STATS (nbinuptime)

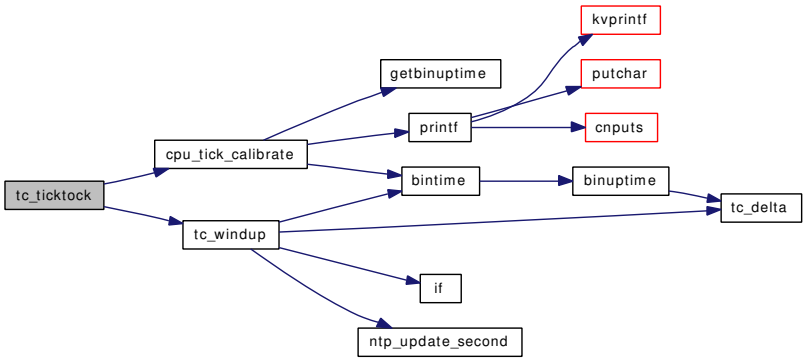
9.63.2.54 void tc_ticktock (void)

Definition at line 796 of file kern_tc.c.

References cpu_tick_calibrate(), and tc_windup().

Referenced by hardclock().

Here is the call graph for this function:



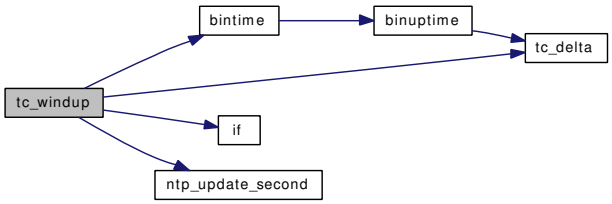
9.63.2.55 static void tc_windup (void) [static]

Definition at line 437 of file kern_tc.c.

References bintime(), if(), LARGE_STEP, ntp_update_second(), tc_delta(), timehands::th_adjustment, timehands::th_counter, timehands::th_generation, timehands::th_microtime, timehands::th_nanotime, timehands::th_next, timehands::th_offset, timehands::th_offset_count, timehands::th_scale, and timehands.

Referenced by tc_setclock(), and tc_ticktock().

Here is the call graph for this function:



9.63.3 Variable Documentation

9.63.3.1 struct timeval boottime

Definition at line 94 of file kern_tc.c.

Referenced by acct_process(), and fill_kinfo_proc_only().

9.63.3.2 struct bintime boottimebin [static]

Definition at line 93 of file kern_tc.c.

9.63.3.3 uint64_t cpu_tick_frequency [static]

Definition at line 841 of file kern_tc.c.

Referenced by cpu_tick_calibrate(), cpu_tickrate(), and set_cputicker().

9.63.3.4 int cpu_tick_variable [static]

Definition at line 840 of file kern_tc.c.

Referenced by cpu_tick_calibrate(), and set_cputicker().

9.63.3.5 cpu_tick_f* cpu_ticks = tc_cpu_ticks

Definition at line 980 of file kern_tc.c.

Referenced by calcru(), cpu_tick_calibrate(), cpu_tickrate(), mi_switch(), proc0_post(), sched_timestamp(), set_cputicker(), and thread_exit().

9.63.3.6 struct timecounter dummy_timecounter [static]

Initial value:

```
 {
        dummy_get_timecount, 0, ~0u, 1000000, "dummy", -1000000
}
```

Definition at line 46 of file kern_tc.c.

9.63.3.7 int tc_tick [static]

Definition at line 792 of file kern_tc.c.

9.63.3.8 struct timehands th0 [static]

Initial value:

```
 {
        &dummy_timecounter,
        0,
        (uint64_t)-1 / 1000000,
        0,
        {1, 0},
        {0, 0},
        {0, 0},
        1,
        &th1
}
```

Definition at line 74 of file kern_tc.c.

9.63.3.9 struct timehands th0 [static]

Definition at line 64 of file kern_tc.c.

9.63.3.10 struct timehands th1 = { NULL, 0, 0, 0, {0, 0}, {0, 0}, {0, 0}, 0, &th2} [static]

Definition at line 73 of file kern_tc.c.

9.63.3.11 struct timehands th2 = { NULL, 0, 0, 0, {0, 0}, {0, 0}, {0, 0}, 0, &th3} [static]

Definition at line 72 of file kern_tc.c.

9.63.3.12 struct timehands th3 = { NULL, 0, 0, 0, {0, 0}, {0, 0}, {0, 0}, 0, &th4} [static]

Definition at line 71 of file kern_tc.c.

9.63.3.13 struct timehands th4 = { NULL, 0, 0, 0, {0, 0}, {0, 0}, {0, 0}, 0, &th5} [static]

Definition at line 70 of file kern_tc.c.

9.63.3.14 struct timehands th5 = { NULL, 0, 0, 0, {0, 0}, {0, 0}, {0, 0}, 0, &th6} [static]

Definition at line 69 of file kern_tc.c.

9.63.3.15 struct timehands th6 = { NULL, 0, 0, 0, {0, 0}, {0, 0}, {0, 0}, 0, &th7} [static]

Definition at line 68 of file kern_tc.c.

9.63.3.16 struct timehands th7 = { NULL, 0, 0, 0, {0, 0}, {0, 0}, {0, 0}, 0, &th8} [static]

Definition at line 67 of file kern_tc.c.

9.63.3.17 struct timehands th8 = { NULL, 0, 0, 0, {0, 0}, {0, 0}, {0, 0}, 0, &th9} [static]

Definition at line 66 of file kern_tc.c.

9.63.3.18 struct timehands th9 = { NULL, 0, 0, 0, {0, 0}, {0, 0}, {0, 0}, 0, &th0} [static]

Definition at line 65 of file kern_tc.c.

9.63.3.19 time_t time_second = 1

Definition at line 90 of file kern_tc.c.

Referenced by hardupdate(), kern_clock_gettime(), kern_msgctl(), kern_msgrcv(), kern_semctl(), kern_shmctl(), shm_delete_mapping(), shmget_allocate_segment(), and vfs_timestamp().

9.63.3.20 time_t time_uptime = 1

Definition at line 91 of file kern_tc.c.

Referenced by malloc(), malloc_last_fail(), and sched_sync().

9.63.3.21 struct timecounter* timecounter = &dummy_timecounter

Definition at line 87 of file kern_tc.c.

Referenced by cf_set_method().

9.63.3.22 struct timecounter* timecounters = &dummy_timecounter [static]

Definition at line 88 of file kern_tc.c.

9.63.3.23 struct timehands* volatile timehands = &th0 [static]

Definition at line 86 of file kern_tc.c.

Referenced by binuptime(), getbintime(), getbinuptime(), getmicrotime(), getmicrouptime(), getnanotime(), getnanouptime(), pps_capture(), tc_cpu_ticks(), tc_getfrequency(), and tc_windup().

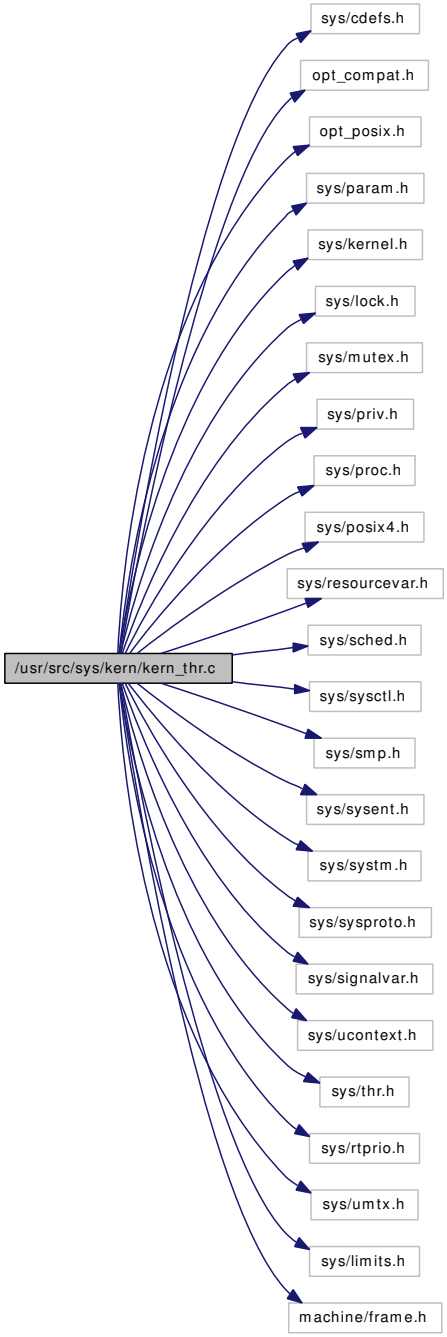
9.63.3.24 int timestepwarnings [static]

Definition at line 102 of file kern_tc.c.

9.64 /usr/src/sys/kern/kern_thr.c File Reference

```
#include <sys/cdefs.h>
#include "opt_compat.h"
#include "opt_posix.h"
#include <sys/param.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/posix4.h>
#include <sys/resourcevar.h>
#include <sys/sched.h>
#include <sys/sysctl.h>
#include <sys/smp.h>
#include <sys/sysent.h>
#include <sys/systm.h>
#include <sys/sysproto.h>
#include <sys/signalvar.h>
#include <sys/ucontext.h>
#include <sys/thr.h>
#include <sys/rtprio.h>
#include <sys/umtx.h>
#include <sys/limits.h>
#include <machine/frame.h>
```

Include dependency graph for kern_thr.c:



Defines

- #define suword_lwpid suword

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_thr.c,v 1.59 2007/01/23 08:46:50 jeff Exp $")

- static int create_thread (struct thread *td, mcontext_t *ctx, void(*start_func)(void *), void *arg, char *stack_base, size_t stack_size, char *tls_base, long *child_tid, long *parent_tid, int flags, struct rtprio *rtp)
- int thr_create (struct thread *td, struct thr_create_args *uap)
- int thr_new (struct thread *td, struct thr_new_args *uap)
- int kern_thr_new (struct thread *td, struct thr_param *param)
- int thr_self (struct thread *td, struct thr_self_args *uap)
- int thr_exit (struct thread *td, struct thr_exit_args *uap)
- int thr_kill (struct thread *td, struct thr_kill_args *uap)
- int thr_suspend (struct thread *td, struct thr_suspend_args *uap)
- int kern_thr_suspend (struct thread *td, struct timespec *tsp)
- int thr_wake (struct thread *td, struct thr_wake_args *uap)
- int thr_set_name (struct thread *td, struct thr_set_name_args *uap)

Variables

- int max_threads_per_proc

9.64.1 Define Documentation

9.64.1.1 #define suword_lwpid suword

Definition at line 72 of file kern_thr.c.

Referenced by create_thread(), thr_exit(), and thr_self().

9.64.2 Function Documentation

9.64.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_thr. *c*, v 1.59 2007/01/23 08:46:50 jeff Exp $")

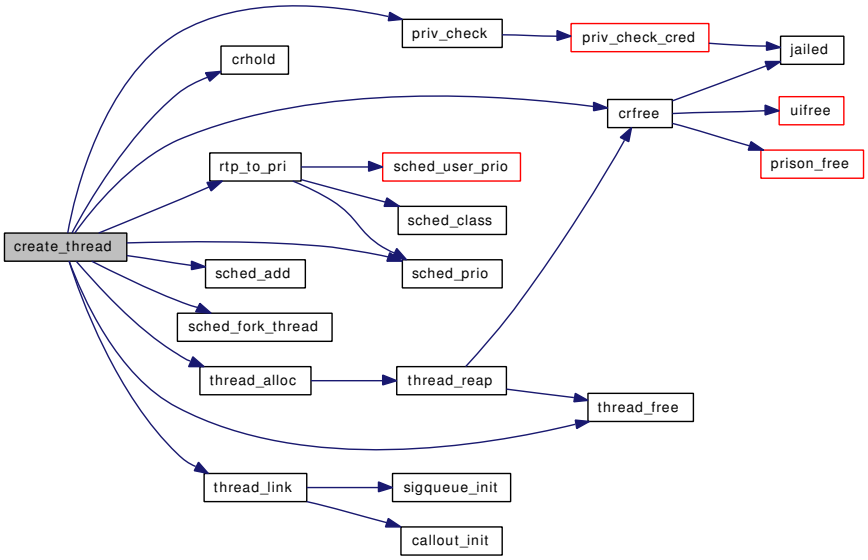
9.64.2.2 static int create_thread (struct thread * *td*, mcontext_t * *ctx*, void(*)(void *) *start_func*, void * *arg*, char * *stack_base*, size_t *stack_size*, char * *tls_base*, long * *child_tid*, long * *parent_tid*, int *flags*, struct rtprio * *rtp*) [static]

Definition at line 136 of file kern_thr.c.

References crfree(), crhold(), priv_check(), rtp_to_pri(), sched_add(), sched_fork_thread(), sched_lock, sched_prio(), suword_lwpid, thread_alloc(), thread_free(), and thread_link().

Referenced by kern_thr_new(), and thr_create().

Here is the call graph for this function:



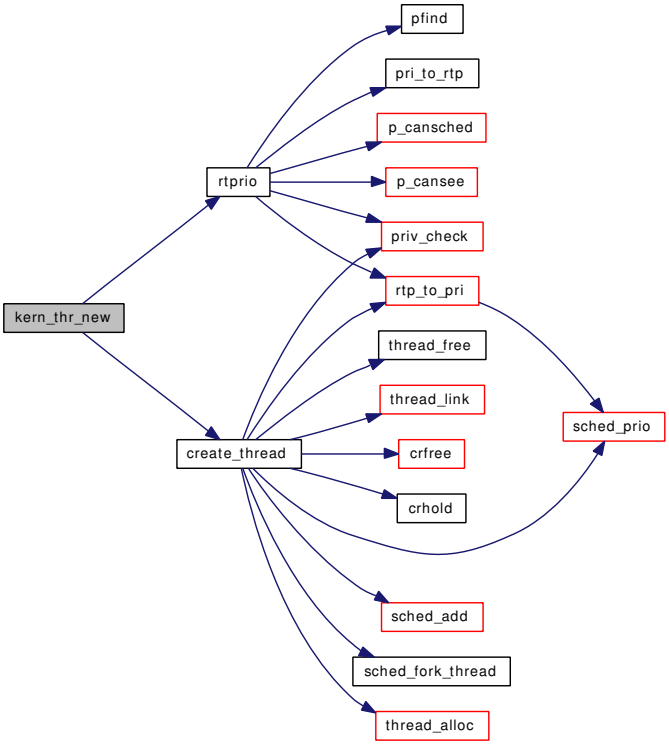
9.64.2.3 int kern_thr_new (struct thread * *td*, struct thr_param * *param*)

Definition at line 118 of file kern_thr.c.

References create_thread(), and rtprio().

Referenced by thr_new().

Here is the call graph for this function:



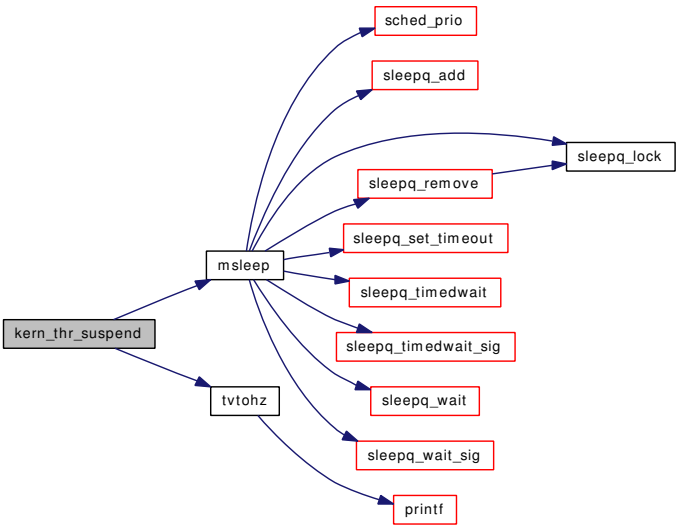
9.64.2.4 int kern_thr_suspend (struct thread * *td*, struct timespec * *tsp*)

Definition at line 358 of file kern_thr.c.

References hz, msleep(), sched_lock, and tvtohz().

Referenced by thr_suspend().

Here is the call graph for this function:

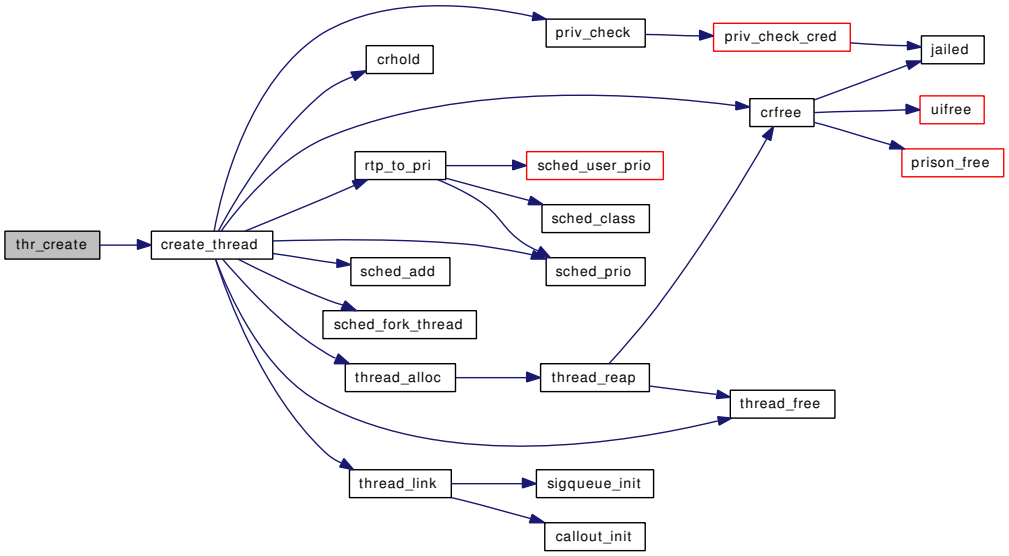


9.64.2.5 int thr_create (struct thread * *td*, struct thr_create_args * *uap*)

Definition at line 88 of file kern_thr.c.

References create_thread().

Here is the call graph for this function:

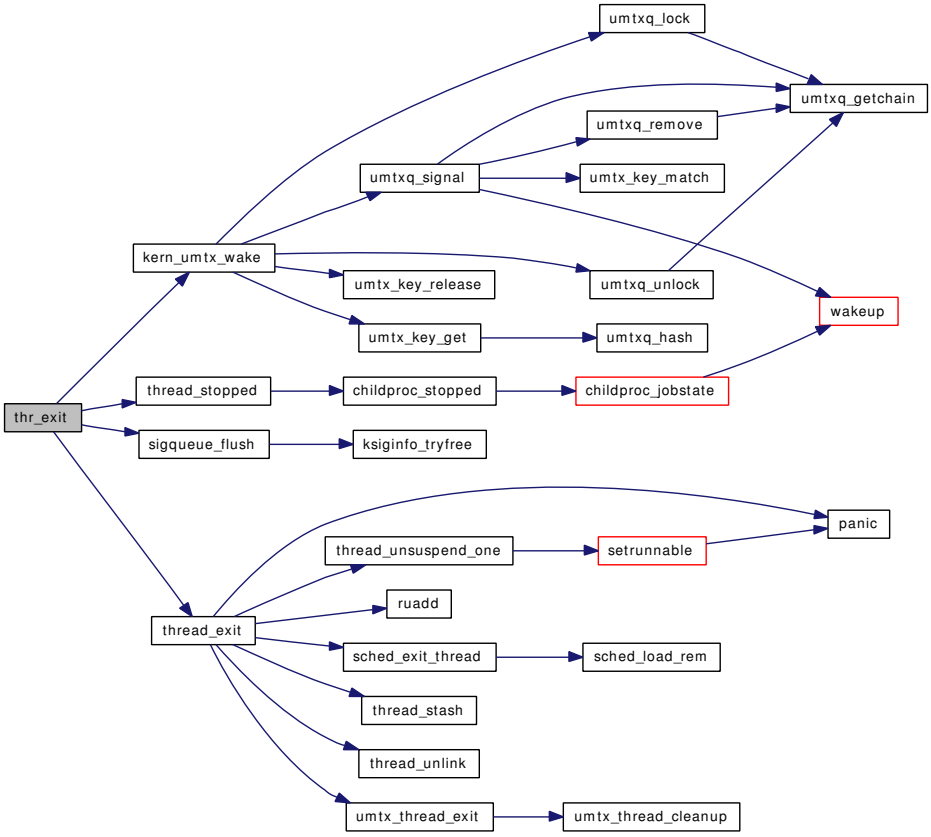


9.64.2.6 int thr_exit (struct thread * *td*, struct thr_exit_args * *uap*)

Definition at line 263 of file kern_thr.c.

References kern_umtx_wake(), sched_lock, sigqueue_flush(), suword_lwpid, thread_exit(), and thread_-stopped().

Here is the call graph for this function:

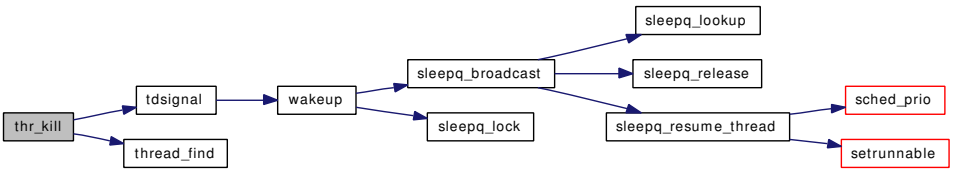


9.64.2.7 int thr_kill (struct thread * *td*, struct thr_kill_args * *uap*)

Definition at line 295 of file kern_thr.c.

References tdsignal(), and thread_find().

Here is the call graph for this function:

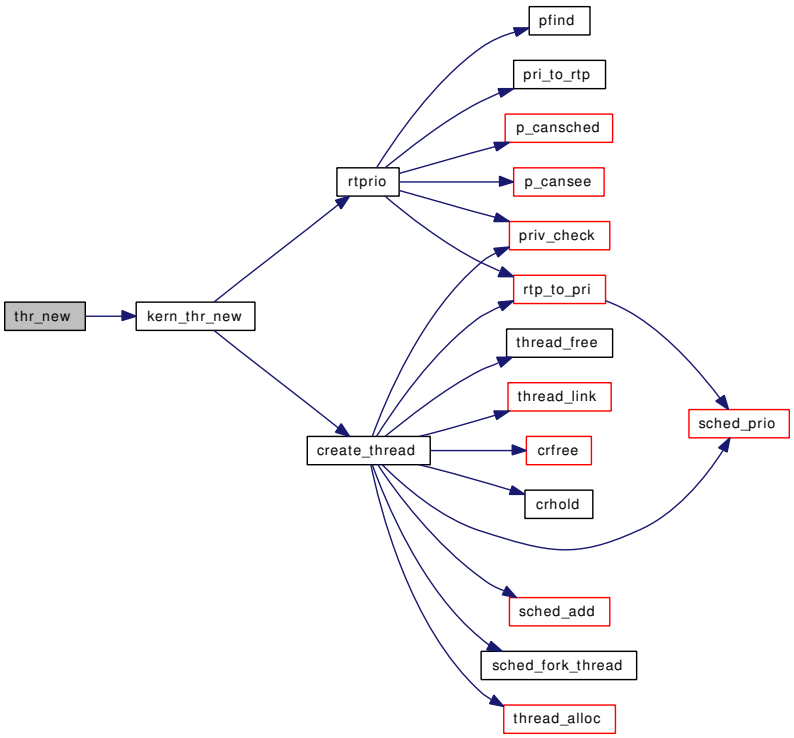


9.64.2.8 int thr_new (struct thread * *td*, struct thr_new_args * *uap*)

Definition at line 103 of file kern_thr.c.

References kern_thr_new().

Here is the call graph for this function:



9.64.2.9 int thr_self (struct thread * *td*, struct thr_self_args * *uap*)

Definition at line 251 of file kern_thr.c.

References suword_lwpid.

9.64.2.10 int thr_set_name (struct thread * *td*, struct thr_set_name_args * *uap*)

Definition at line 426 of file kern_thr.c.

References thread_find().

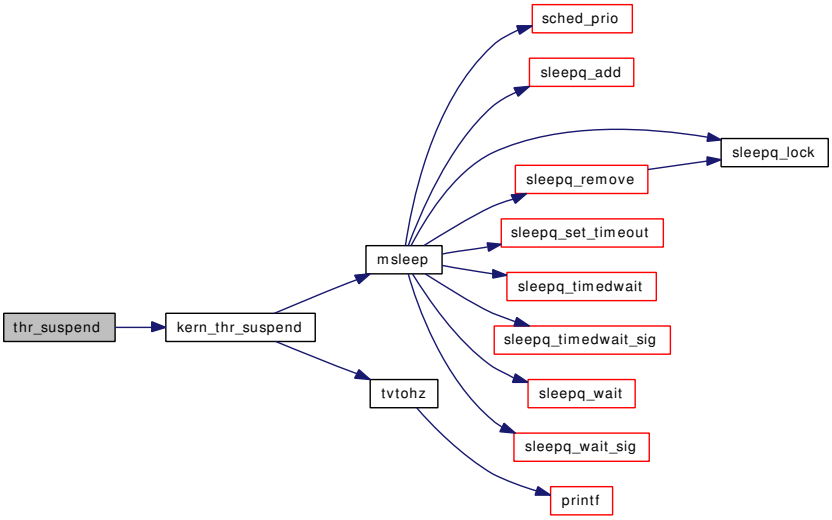
Here is the call graph for this function:

9.64.2.11 int thr_suspend (struct thread * *td*, struct thr_suspend_args * *uap*)

Definition at line 338 of file kern_thr.c.

References kern_thr_suspend().

Here is the call graph for this function:

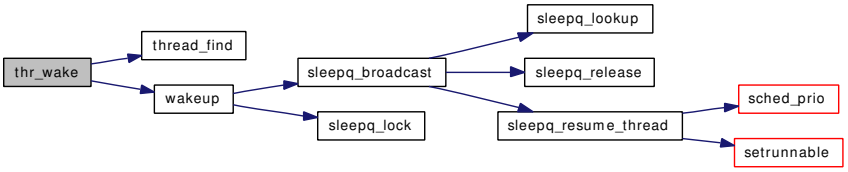


9.64.2.12 int thr_wake (struct thread * *td*, struct thr_wake_args * *uap*)

Definition at line 399 of file kern_thr.c.

References sched_lock, thread_find(), and wakeup().

Here is the call graph for this function:



9.64.3 Variable Documentation

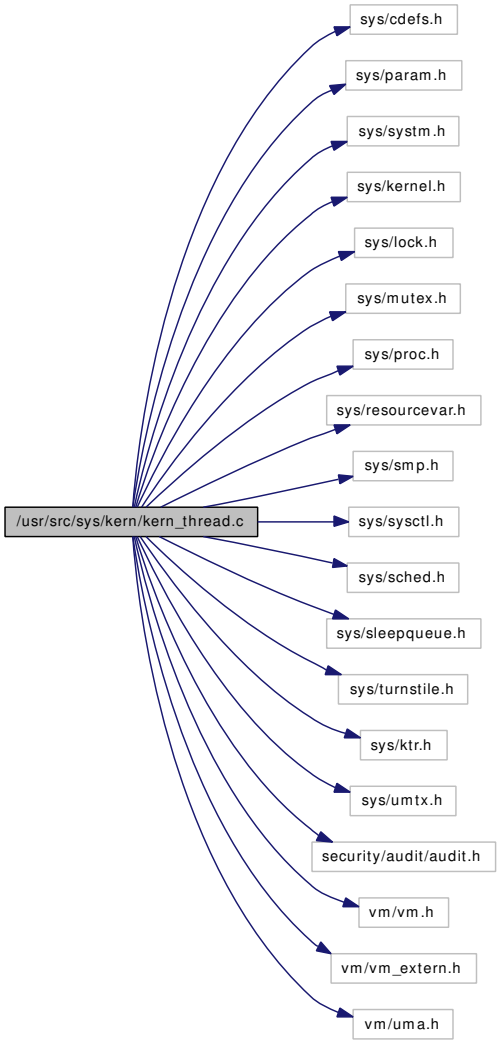
9.64.3.1 int max_threads_per_proc

Definition at line 60 of file kern_thread.c.

9.65 /usr/src/sys/kern/kern_thread.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/resourcevar.h>
#include <sys/smp.h>
#include <sys/sysctl.h>
#include <sys/sched.h>
#include <sys/sleepqueue.h>
#include <sys/turnstile.h>
#include <sys/ktr.h>
#include <sys/umtx.h>
#include <security/audit/audit.h>
#include <vm/vm.h>
#include <vm/vm_extern.h>
#include <vm/uma.h>
```

Include dependency graph for kern_thread.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_thread.c,v 1.240 2006/12/31 15:56:04 rwatson Exp $")
- SYSCTL_NODE (_kern, OID_AUTO, threads, CTLFLAG_RW, 0,"thread allocation")
- SYSCTL_INT (_kern_threads, OID_AUTO, max_threads_per_proc, CTLFLAG_RW,&max_threads_per_proc, 0,"Limit on threads per proc")
- SYSCTL_INT (_kern_threads, OID_AUTO, max_threads_hits, CTLFLAG_RD,&max_threads_hits, 0,"")
- TAILQ_HEAD (thread)
- static void thread_dtor (void *mem, int size, void *arg)
- static int thread_init (void *mem, int size, int flags)
- static void thread_fini (void *mem, int size)
- void proc_linkup (struct proc *p, struct thread *td)
- void threadinit (void)
- void thread_stash (struct thread *td)
- void thread_reap (void)
- thread * thread_alloc (void)

- void thread_free (struct thread *td)
- void thread_exit (void)
- void thread_wait (struct proc *p)
- void thread_link (struct thread *td, struct proc *p)
- void thread_unthread (struct thread *td)
- void thread_unlink (struct thread *td)
- int thread_single (int mode)
- int thread_suspend_check (int return_instead)
- void thread_suspend_one (struct thread *td)
- void thread_unsuspend_one (struct thread *td)
- void thread_unsuspend (struct proc *p)
- void thread_single_end (void)
- thread * thread_find (struct proc *p, lwpid_t tid)

Variables

- static uma_zone_t thread_zone
- int max_threads_per_proc = 1500
- int max_threads_hits

9.65.1 Function Documentation

9.65.1.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_thread. *c*, v 1.240 2006/12/31 15:56:04 rwatson Exp $")

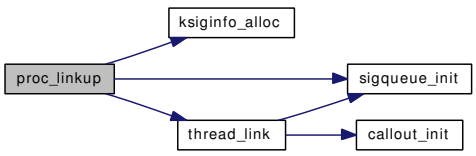
9.65.1.2 void proc_linkup (struct proc * *p*, struct thread * *td*)

Definition at line 222 of file kern_thread.c.

References ksiginfo_alloc(), sigqueue_init(), and thread_link().

Referenced by proc_init().

Here is the call graph for this function:



9.65.1.3 SYSCTL_INT (_kern_threads, OID_AUTO, max_threads_hits, CTLFLAG_RD, & *max_threads_hits*, 0, "")

9.65.1.4 SYSCTL_INT (_kern_threads, OID_AUTO, max_threads_per_proc, CTLFLAG_RW, & *max_threads_per_proc*, 0, "Limit on threads per proc")

9.65.1.5 SYSCTL_NODE (_kern, OID_AUTO, threads, CTLFLAG_RW, 0, "thread allocation")

9.65.1.6 TAILQ_HEAD (thread)

Definition at line 72 of file kern_thread.c.

References mp_ncpus, and sysctl_handle_int().

Here is the call graph for this function:

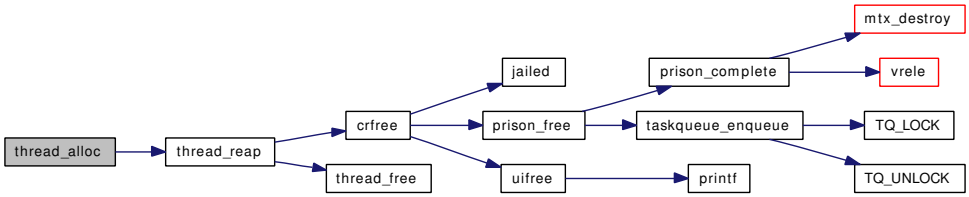
9.65.1.7 struct thread* thread_alloc (void)

Definition at line 299 of file kern_thread.c.

References thread_reap(), and thread_zone.

Referenced by create_thread(), and proc_init().

Here is the call graph for this function:



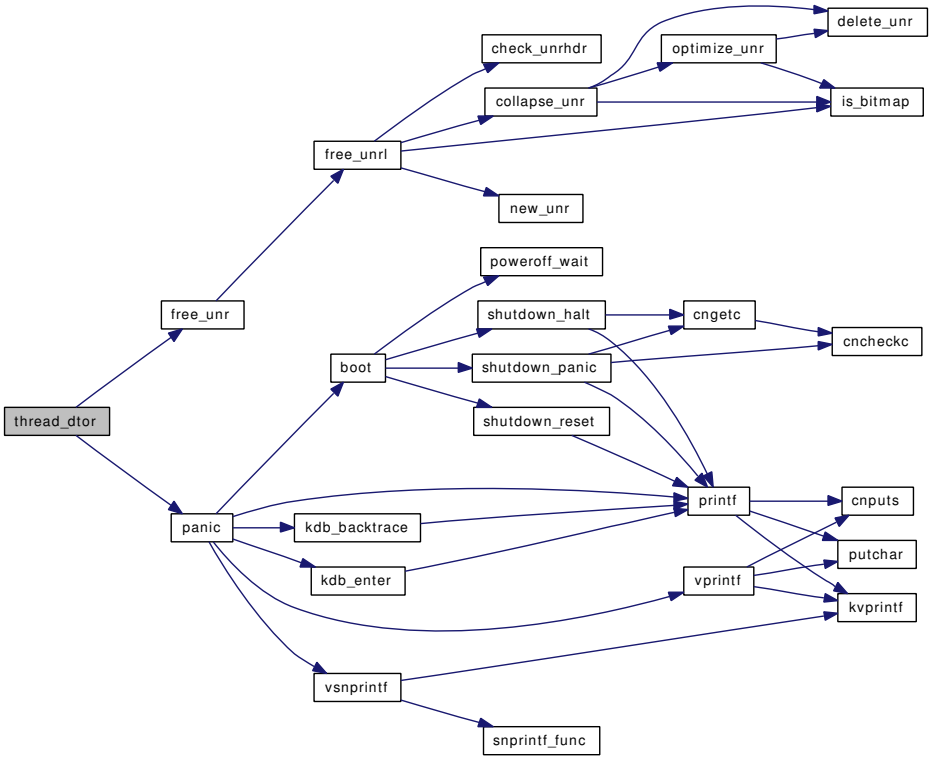
9.65.1.8 static void thread_dtor (void * *mem*, int *size*, void * *arg*) [static]

Definition at line 145 of file kern_thread.c.

References free_unr(), and panic().

Referenced by threadinit().

Here is the call graph for this function:



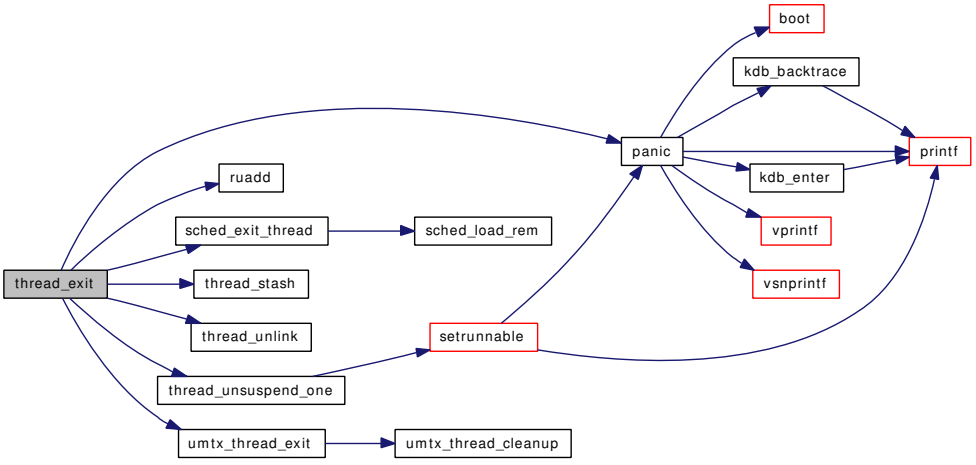
9.65.1.9 void thread_exit (void)

Definition at line 350 of file kern_thread.c.

References cpu_ticks, Giant, panic(), ruadd(), sched_exit_thread(), sched_lock, thread_stash(), thread_unlink(), thread_unsuspend_one(), ticks, and umtx_thread_exit().

Referenced by exit1(), kse_exit(), thr_exit(), and thread_suspend_check().

Here is the call graph for this function:



9.65.1.10 struct thread* thread_find (struct proc * *p*, lwpid_t *tid*)

Definition at line 918 of file kern_thread.c.

References sched_lock.

Referenced by psignal_event(), rtprio_thread(), thr_kill(), thr_set_name(), thr_wake(), and umtxq_sleep_pi().

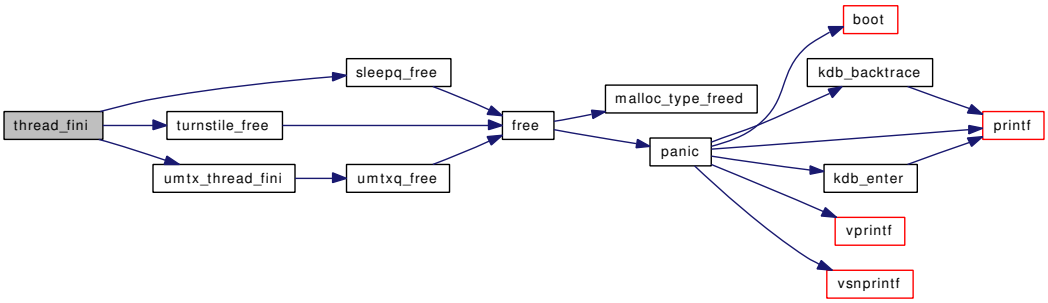
9.65.1.11 static void thread_fini (void * *mem*, int *size*) [static]

Definition at line 202 of file kern_thread.c.

References sleepq_free(), turnstile_free(), and umtx_thread_fini().

Referenced by threadinit().

Here is the call graph for this function:



9.65.1.12 void thread_free (struct thread * *td*)

Definition at line 311 of file kern_thread.c.

References thread_zone.

Referenced by create_thread(), proc_fini(), thread_reap(), and thread_wait().

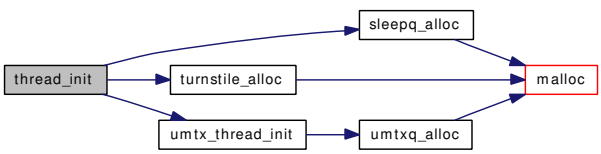
9.65.1.13 static int thread_init (void * *mem*, int *size*, int *flags*) [static]

Definition at line 182 of file kern_thread.c.

References sleepq_alloc(), turnstile_alloc(), and umtx_thread_init().

Referenced by threadinit().

Here is the call graph for this function:



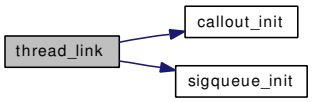
9.65.1.14 void thread_link (struct thread * *td*, struct proc * *p*)

Definition at line 528 of file kern_thread.c.

References callout_init(), and sigqueue_init().

Referenced by create_thread(), and proc_linkup().

Here is the call graph for this function:



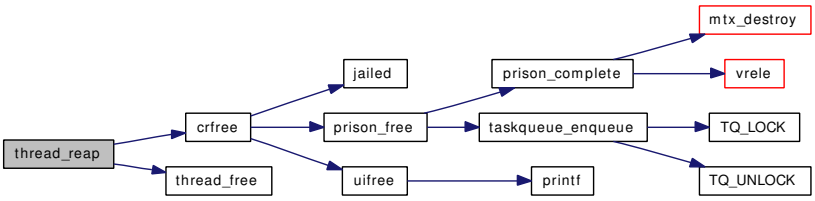
9.65.1.15 void thread_reap (void)

Definition at line 271 of file kern_thread.c.

References crfree(), and thread_free().

Referenced by thread_alloc(), and thread_wait().

Here is the call graph for this function:



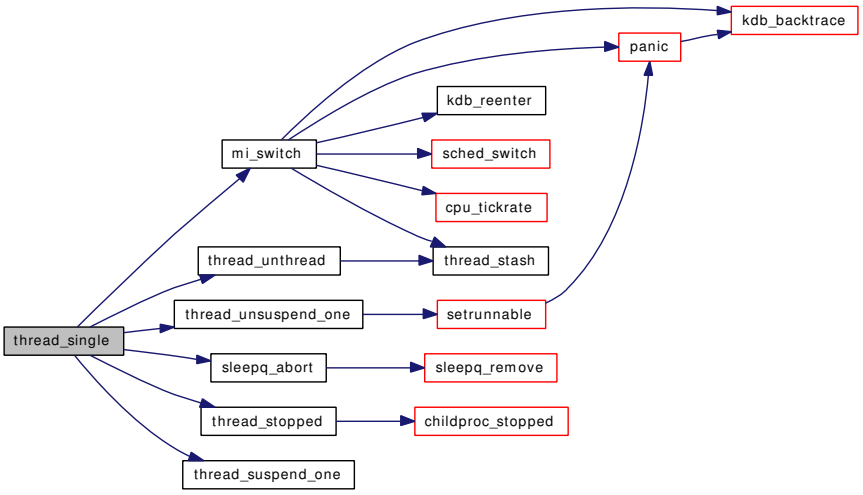
9.65.1.16 int thread_single (int *mode*)

Definition at line 599 of file kern_thread.c.

References Giant, mi_switch(), sched_lock, sleepq_abort(), thread_stopped(), thread_suspend_one(), thread_unsuspend_one(), and thread_unthread().

Referenced by exit1(), fork1(), kern_execve(), and sigexit().

Here is the call graph for this function:



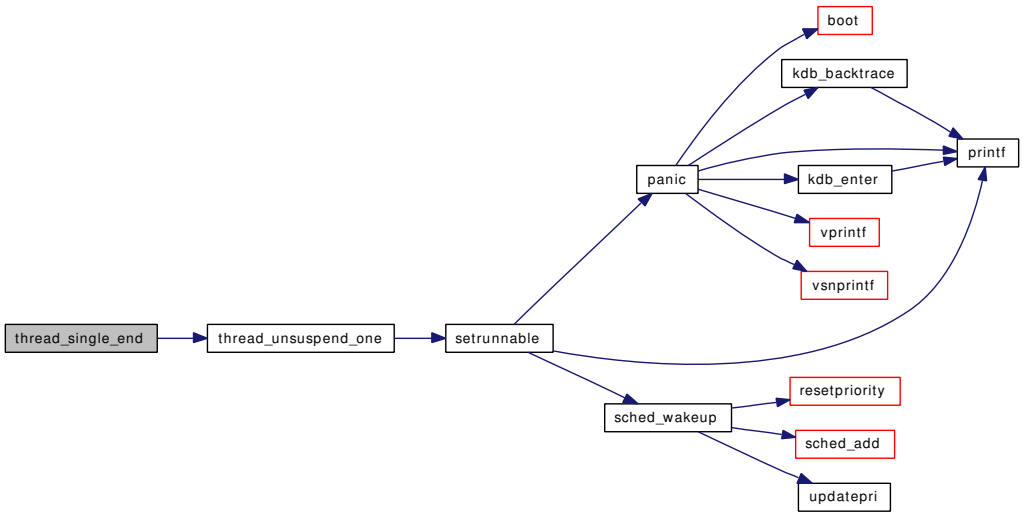
9.65.1.17 void thread_single_end (void)

Definition at line 890 of file kern_thread.c.

References sched_lock, and thread_unsuspend_one().

Referenced by fork1(), and kern_execve().

Here is the call graph for this function:



9.65.1.18 void thread_stash (struct thread * *td*)

Definition at line 260 of file kern_thread.c.

Referenced by fork_exit(), mi_switch(), thread_exit(), and thread_unthread().

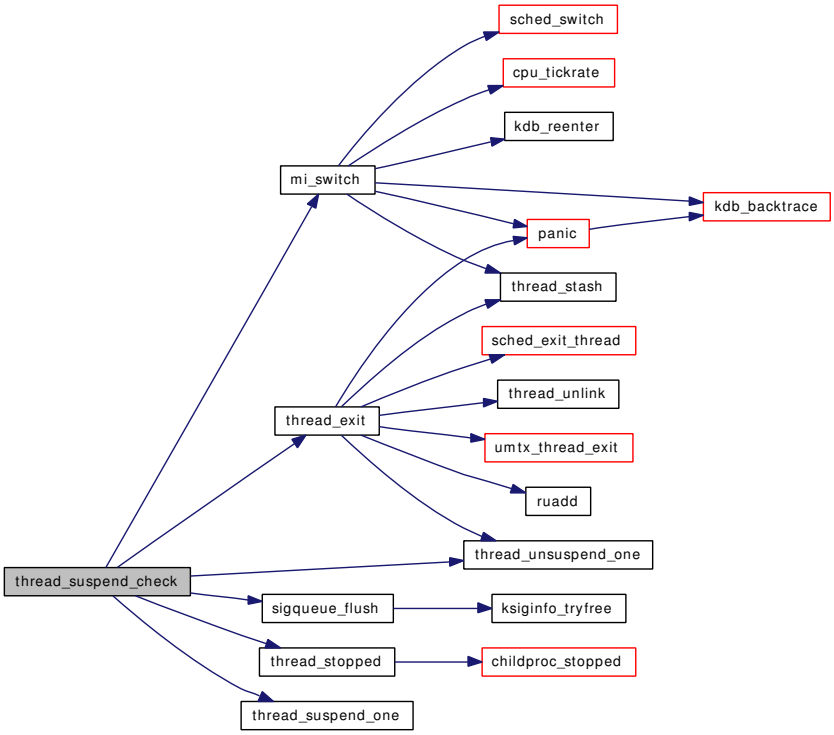
9.65.1.19 int thread_suspend_check (int *return_instead*)

Definition at line 764 of file kern_thread.c.

References Giant, mi_switch(), sched_lock, sigqueue_flush(), thread_exit(), thread_stopped(), thread_-suspend_one(), and thread_unsuspend_one().

Referenced by exit1(), sleepq_catch_signals(), and userret().

Here is the call graph for this function:



9.65.1.20 void thread_suspend_one (struct thread * *td*)

Definition at line 835 of file kern_thread.c.

References sched_lock.

Referenced by issignal(), kse_thr_interrupt(), ptracestop(), sig_suspend_threads(), thread_single(), and thread_suspend_check().

9.65.1.21 void thread_unlink (struct thread * *td*)

Definition at line 574 of file kern_thread.c.

References sched_lock.

Referenced by thread_exit().

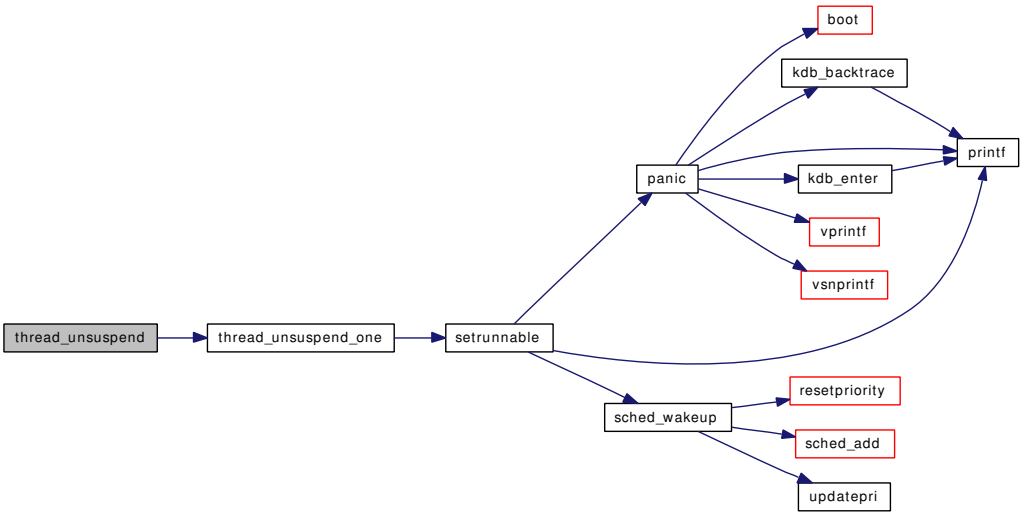
9.65.1.22 void thread_unsuspend (struct proc * *p*)

Definition at line 863 of file kern_thread.c.

References sched_lock, and thread_unsuspend_one().

Referenced by kern_ptrace().

Here is the call graph for this function:



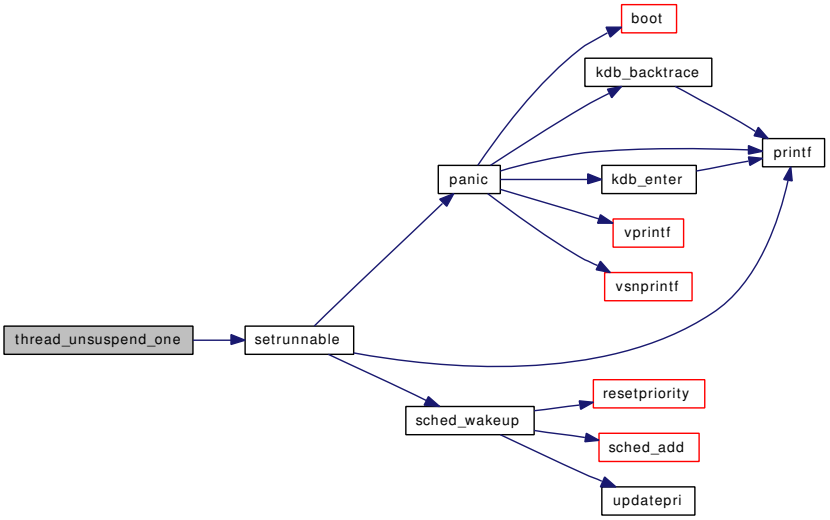
9.65.1.23 void thread_unsuspend_one (struct thread * *td*)

Definition at line 847 of file kern_thread.c.

References sched_lock, and setrunnable().

Referenced by thread_exit(), thread_single(), thread_single_end(), thread_suspend_check(), and thread_-unsuspend().

Here is the call graph for this function:



9.65.1.24 void thread_unthread (struct thread * *td*)

Definition at line 549 of file kern_thread.c.

References thread_stash().

Referenced by kse_exit(), and thread_single().

Here is the call graph for this function:

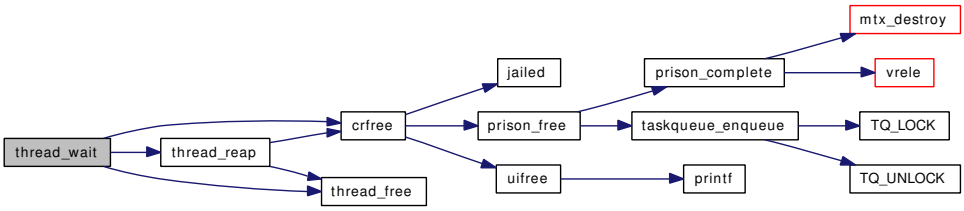
9.65.1.25 void thread_wait (struct proc * *p*)

Definition at line 492 of file kern_thread.c.

References crfree(), Giant, thread_free(), and thread_reap().

Referenced by kern_wait().

Here is the call graph for this function:



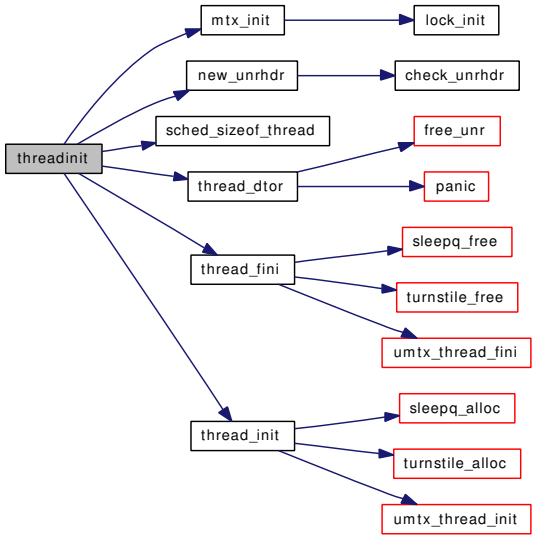
9.65.1.26 void threadinit (void)

Definition at line 241 of file kern_thread.c.

References mtx_init(), new_unrhdr(), sched_sizeof_thread(), thread_dtor(), thread_fini(), thread_init(), and thread_zone.

Referenced by proc0_init().

Here is the call graph for this function:



9.65.2 Variable Documentation

9.65.2.1 int max_threads_hits

Definition at line 64 of file kern_thread.c.

9.65.2.2 int max_threads_per_proc = 1500

Definition at line 60 of file kern_thread.c.

9.65.2.3 uma_zone_t thread_zone [static]

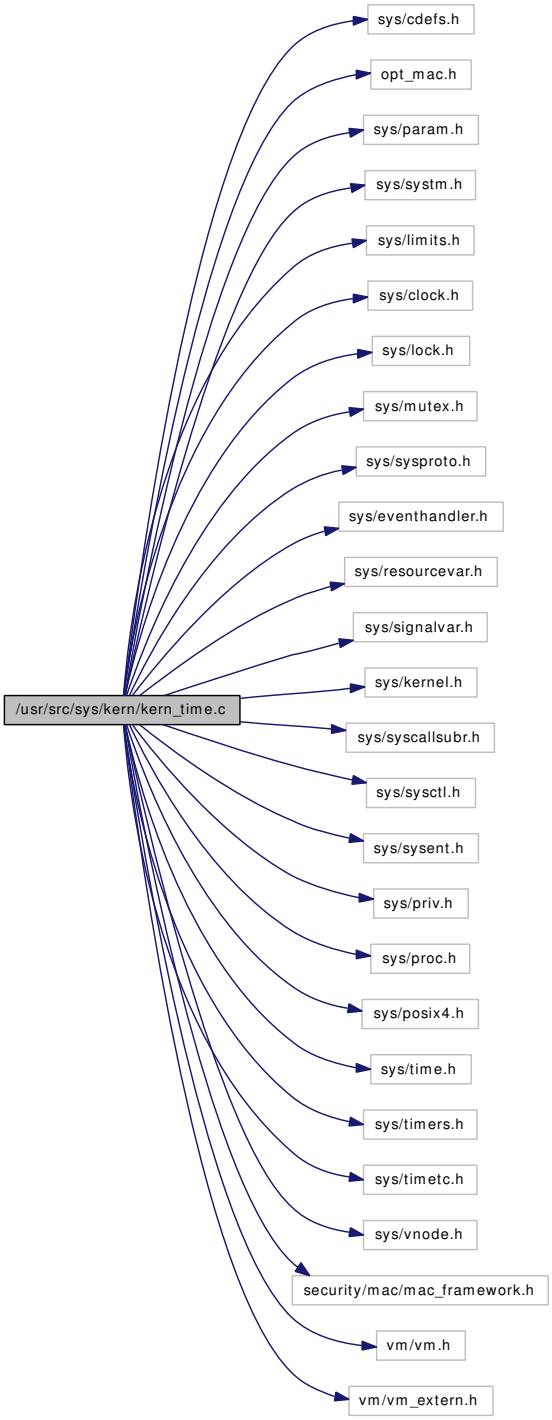
Definition at line 56 of file kern_thread.c.

Referenced by thread_alloc(), thread_free(), and threadinit().

9.66 /usr/src/sys/kern/kern_time.c File Reference

```
#include <sys/cdefs.h>
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/limits.h>
#include <sys/clock.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/sysproto.h>
#include <sys/eventhandler.h>
#include <sys/resourcevar.h>
#include <sys/signalvar.h>
#include <sys/kernel.h>
#include <sys/syscallsubr.h>
#include <sys/sysctl.h>
#include <sys/sysent.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/posix4.h>
#include <sys/time.h>
#include <sys/timers.h>
#include <sys/timetc.h>
#include <sys/vnode.h>
#include <security/mac/mac_framework.h>
#include <vm/vm.h>
#include <vm/vm_extern.h>
```

Include dependency graph for kern_time.c:



Data Structures

- struct clock_gettime_args
- struct clock_settime_args
- struct clock_getres_args
- struct nanosleep_args

- struct gettimeofday_args
- struct settimeofday_args
- struct getitimer_args
- struct setitimer_args
- struct ktimer_create_args
- struct ktimer_delete_args
- struct ktimer_settime_args
- struct ktimer_gettime_args
- struct timer_getoverrun_args

Defines

- #define MAX_CLOCKS (CLOCK_MONOTONIC+1)
- #define CLOCK_CALL(clock, call, arglist) ((*posix_clocks[clock].call) arglist)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_time.c,v 1.137 2006/11/28 03:24:34 davidxu Exp $")
- static int settime (struct thread *, struct timeval *)
- static void timevalfix (struct timeval *)
- static void no_lease_updatetime (int)
- static void itimer_start (void)
- static int itimer_init (void *, int, int)
- static void itimer_fini (void *, int)
- static void itimer_enter (struct itimer *)
- static void itimer_leave (struct itimer *)
- static struct itimer * itimer_find (struct proc *, int)
- static void itimers_alloc (struct proc *)
- static void itimers_event_hook_exec (void *arg, struct proc *p, struct image_params *imgp)
- static void itimers_event_hook_exit (void *arg, struct proc *p)
- static int realtimer_create (struct itimer *)
- static int realtimer_gettime (struct itimer *, struct itimerspec *)
- static int realtimer_settime (struct itimer *, int, struct itimerspec *, struct itimerspec *)
- static int realtimer_delete (struct itimer *)
- static void realtimer_clocktime (clockid_t, struct timespec *)
- static void realtimer_expire (void *)
- static int kern_timer_create (struct thread *, clockid_t, struct sigevent *, int *, int)
- static int kern_timer_delete (struct thread *, int)
- int register_posix_clock (int, struct kclock *)
- void itimer_fire (struct itimer *it)
- int itimespecfix (struct timespec *ts)
- SYSINIT (posix_timer, SI_SUB_P1003_1B, SI_ORDER_FIRST+4, itimer_start, NULL)
- int clock_gettime (struct thread *td, struct clock_gettime_args *uap)
- int kern_clock_gettime (struct thread *td, clockid_t clock_id, struct timespec *ats)
- int clock_settime (struct thread *td, struct clock_settime_args *uap)
- int kern_clock_settime (struct thread *td, clockid_t clock_id, struct timespec *ats)
- int clock_getres (struct thread *td, struct clock_getres_args *uap)
- int kern_clock_getres (struct thread *td, clockid_t clock_id, struct timespec *ts)
- int kern_nanosleep (struct thread *td, struct timespec *rqt, struct timespec *rmt)

- int nanosleep (struct thread *td, struct nanosleep_args *uap)
- int gettimeofday (struct thread *td, struct gettimeofday_args *uap)
- int settimeofday (struct thread *td, struct settimeofday_args *uap)
- int kern_settimeofday (struct thread *td, struct timeval *tv, struct timezone *tzp)
- int getitimer (struct thread *td, struct getitimer_args *uap)
- int kern_getitimer (struct thread *td, u_int which, struct itimerval *aitv)
- int setitimer (struct thread *td, struct setitimer_args *uap)
- int kern_setitimer (struct thread *td, u_int which, struct itimerval *aitv, struct itimerval *oitv)
- void realitexpire (void *arg)
- int itimerfix (struct timeval *tv)
- int itimerdecr (struct itimerval *itp, int usec)
- void timevaladd (struct timeval *t1, const struct timeval *t2)
- void timevalsub (struct timeval *t1, const struct timeval *t2)
- int ratecheck (struct timeval *lasttime, const struct timeval *mininterval)
- int ppsratecheck (struct timeval *lasttime, int *curpps, int maxpps)
- int ktimer_create (struct thread *td, struct ktimer_create_args *uap)
- int ktimer_delete (struct thread *td, struct ktimer_delete_args *uap)
- int ktimer_settime (struct thread *td, struct ktimer_settime_args *uap)
- int ktimer_gettime (struct thread *td, struct ktimer_gettime_args *uap)
- int ktimer_getoverrun (struct thread *td, struct ktimer_getoverrun_args *uap)
- int itimer_accept (struct proc *p, int timerid, ksiginfo_t *ksi)
- static void itimers_event_hook_exec (void *arg, struct proc *p, struct image_params *imgp __unused)

Variables

- static struct kclock posix_clocks [MAX_CLOCKS]
- static uma_zone_t itimer_zone = NULL
- void(*) lease_updatetime (int) = no_lease_updatetime
- static int nanowait

9.66.1 Define Documentation

9.66.1.1 #define CLOCK_CALL(clock, call, arglist) ((*posix_clocks[clock].call) arglist)

Definition at line 107 of file kern_time.c.

Referenced by itimers_event_hook_exit(), kern_timer_create(), kern_timer_delete(), ktimer_gettime(), and ktimer_settime().

9.66.1.2 #define MAX_CLOCKS (CLOCK_MONOTONIC+1)

Definition at line 64 of file kern_time.c.

Referenced by itimers_event_hook_exit(), kern_timer_create(), and register_posix_clock().

9.66.2 Function Documentation

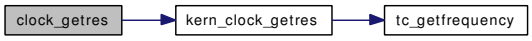
9.66.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_time. *c*, v 1.137 2006/11/28 03:24:34 davidxu Exp $")

9.66.2.2 int clock_getres (struct thread * *td*, struct clock_getres_args * *uap*)

Definition at line 308 of file kern_time.c.

References clock_getres_args::clock_id, kern_clock_getres(), and clock_getres_args::tp.

Here is the call graph for this function:

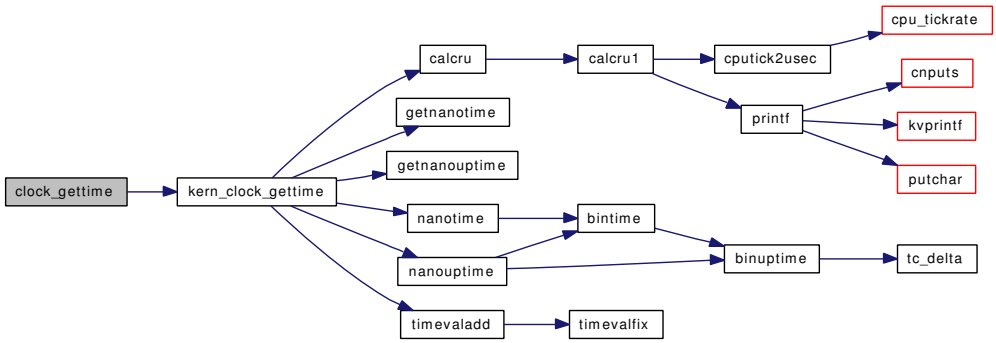


9.66.2.3 int clock_gettime (struct thread * *td*, struct clock_gettime_args * *uap*)

Definition at line 195 of file kern_time.c.

References clock_gettime_args::clock_id, kern_clock_gettime(), and clock_gettime_args::tp.

Here is the call graph for this function:

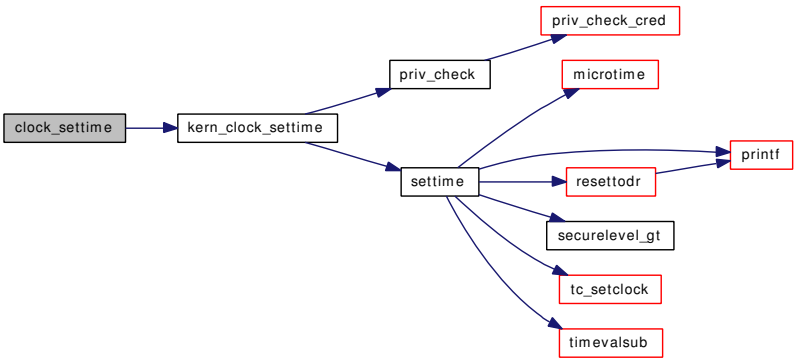


9.66.2.4 int clock_settime (struct thread * *td*, struct clock_settime_args * *uap*)

Definition at line 267 of file kern_time.c.

References clock_settime_args::clock_id, kern_clock_settime(), and clock_settime_args::tp.

Here is the call graph for this function:



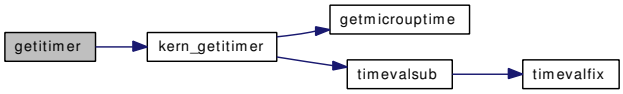
9.66.2.5 int getitimer (struct thread * *td*, struct getitimer_args * *uap*)

Definition at line 553 of file kern_time.c.

References getitimer_args::itv, kern_getitimer(), and getitimer_args::which.

Referenced by setitimer().

Here is the call graph for this function:

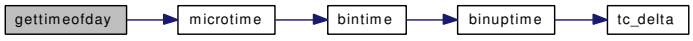


9.66.2.6 int gettimeofday (struct thread * *td*, struct gettimeofday_args * *uap*)

Definition at line 444 of file kern_time.c.

References microtime(), gettimeofday_args::tp, tz_dsttime, tz_minuteswest, and gettimeofday_args::tzp.

Here is the call graph for this function:



9.66.2.7 int itimer_accept (struct proc * *p*, int *timerid*, ksiginfo_t * *ksi*)

Definition at line 1354 of file kern_time.c.

References itimer_find().

Referenced by postsig().

Here is the call graph for this function:

9.66.2.8 static void itimer_enter (struct itimer *) [static]

Definition at line 931 of file kern_time.c.

Referenced by ktimer_gettime(), and ktimer_settime().

9.66.2.9 static struct itimer * itimer_find (struct proc *, int) [static]

Definition at line 1104 of file kern_time.c.

Referenced by itimer_accept(), kern_timer_delete(), ktimer_getoverrun(), ktimer_gettime(), and ktimer_settime().

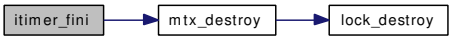
9.66.2.10 static void itimer_fini (void *, int) [static]

Definition at line 922 of file kern_time.c.

References mtx_destroy().

Referenced by itimer_start().

Here is the call graph for this function:



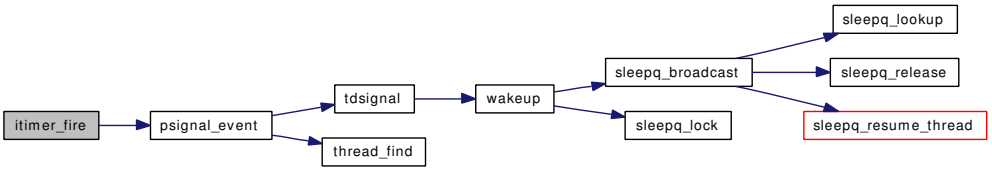
9.66.2.11 void itimer_fire (struct itimer * *it*)

Definition at line 1431 of file kern_time.c.

References psignal_event(), and ret.

Referenced by realtimer_expire().

Here is the call graph for this function:



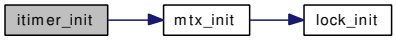
9.66.2.12 static int itimer_init (void *, int, int) [static]

Definition at line 912 of file kern_time.c.

References mtx_init().

Referenced by itimer_start().

Here is the call graph for this function:



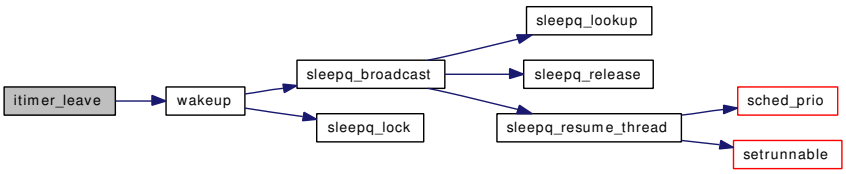
9.66.2.13 static void itimer_leave (struct itimer *) [static]

Definition at line 939 of file kern_time.c.

References wakeup().

Referenced by ktimer_gettime(), and ktimer_settime().

Here is the call graph for this function:

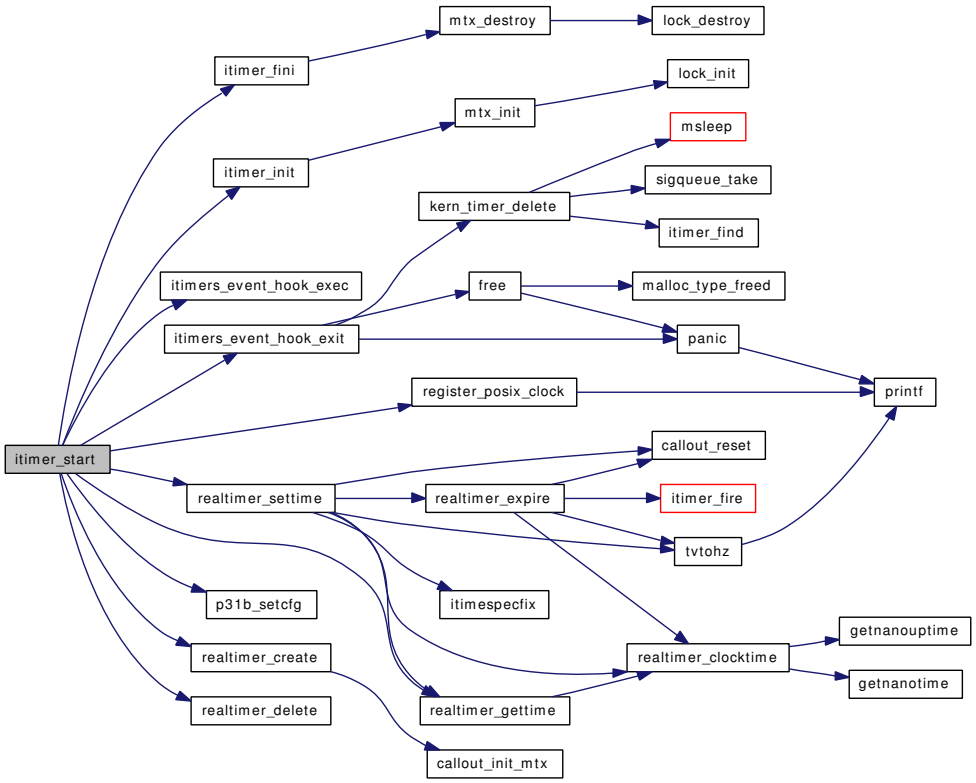


9.66.2.14 static void itimer_start (void) [static]

Definition at line 877 of file kern_time.c.

References itimer_fini(), itimer_init(), itimer_zone, itimers_event_hook_exec(), itimers_event_hook_exit(), p31b_setcfg(), realtimer_create(), realtimer_delete(), realtimer_gettime(), realtimer_settime(), and register_posix_clock().

Here is the call graph for this function:



9.66.2.15 int itimerdecr (struct itimerval * *itp*, int *usec*)

Definition at line 746 of file kern_time.c.

Referenced by hardclock_cpu().

9.66.2.16 int itimerfix (struct timeval * *tv*)

Definition at line 725 of file kern_time.c.

References tick.

Referenced by aio_suspend(), aio_waitcomplete(), kern_select(), kern_setitimer(), kqueue_scan(), and poll().

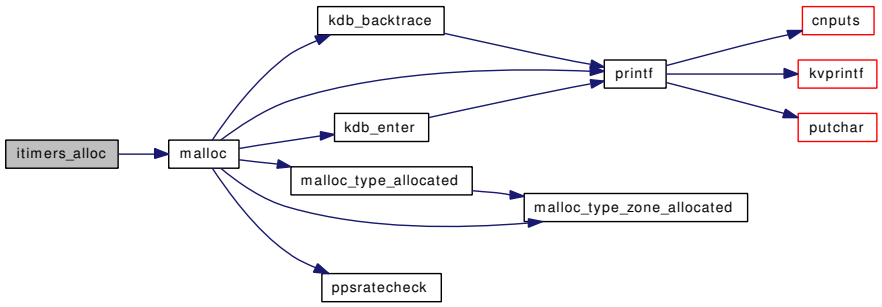
9.66.2.17 static void itimers_alloc (struct proc *) [static]

Definition at line 1467 of file kern_time.c.

References malloc().

Referenced by kern_timer_create().

Here is the call graph for this function:

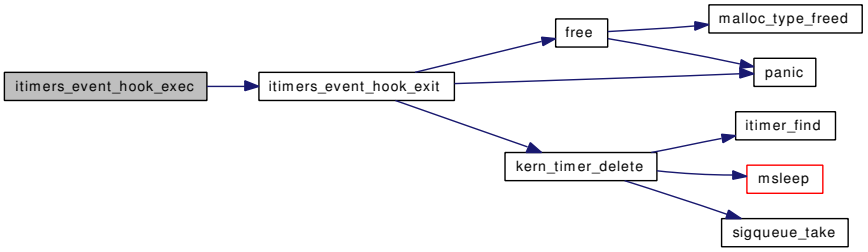


9.66.2.18 static void itimers_event_hook_exec (void * *arg*, struct proc * *p*, struct image_params *imgp *__unused*) [static]

Definition at line 1490 of file kern_time.c.

References itimers_event_hook_exit().

Here is the call graph for this function:



9.66.2.19 static void itimers_event_hook_exec (void * *arg*, struct proc * *p*, struct image_params * *imgp*) [static]

Referenced by itimer_start().

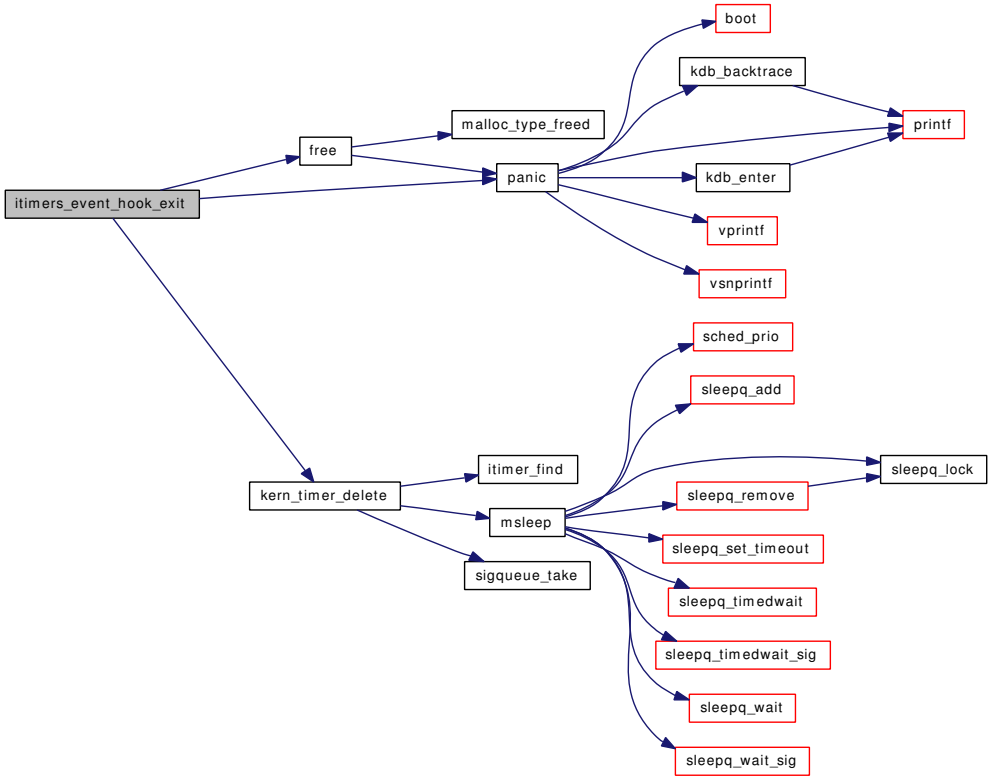
9.66.2.20 static void itimers_event_hook_exit (void * *arg*, struct proc * *p*) [static]

Definition at line 1497 of file kern_time.c.

References CLOCK_CALL, free(), kern_timer_delete(), MAX_CLOCKS, panic(), and posix_clocks.

Referenced by itimer_start(), and itimers_event_hook_exec().

Here is the call graph for this function:



9.66.2.21 int itimespecfix (struct timespec * *ts*)

Definition at line 1371 of file kern_time.c.

References tick.

Referenced by realtimer_settime().

9.66.2.22 int kern_clock_getres (struct thread * *td*, clockid_t *clock_id*, struct timespec * *ts*)

Definition at line 323 of file kern_time.c.

References hz, and tc_getfrequency().

Referenced by clock_getres().

Here is the call graph for this function:

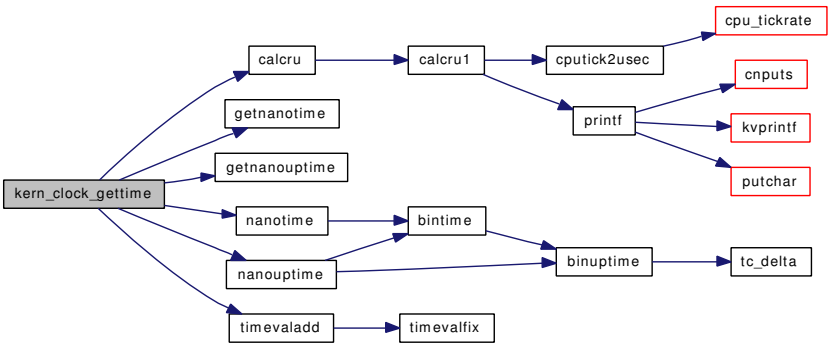
9.66.2.23 int kern_clock_gettime (struct thread * *td*, clockid_t *clock_id*, struct timespec * *ats*)

Definition at line 208 of file kern_time.c.

References calcru(), getnanotime(), getnanouptime(), nanotime(), nanouptime(), time_second, and timevaladd().

Referenced by clock_gettime().

Here is the call graph for this function:



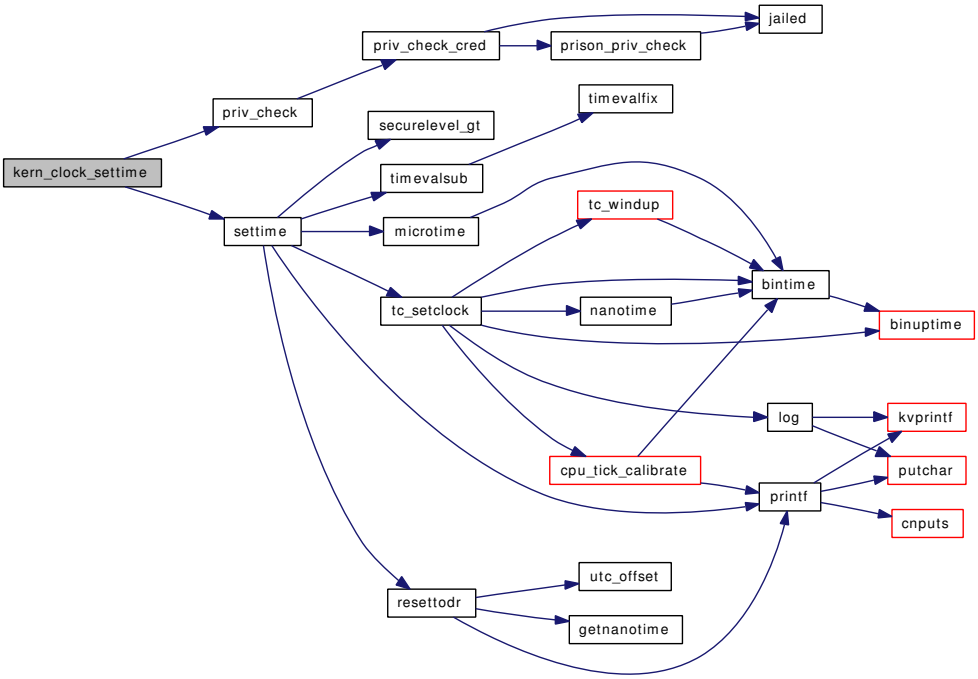
9.66.2.24 int kern_clock_settime (struct thread * *td*, clockid_t *clock_id*, struct timespec * *ats*)

Definition at line 278 of file kern_time.c.

References priv_check(), and settime().

Referenced by clock_settime().

Here is the call graph for this function:



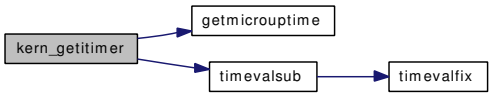
9.66.2.25 int kern_getitimer (struct thread * *td*, u_int *which*, struct itimerval * *aitv*)

Definition at line 565 of file kern_time.c.

References getmicrouptime(), sched_lock, and timevalsub().

Referenced by getitimer(), and kern_setitimer().

Here is the call graph for this function:



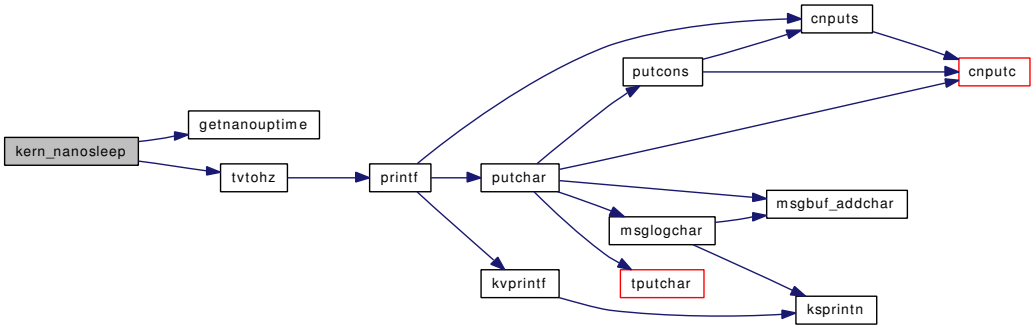
9.66.2.26 int kern_nanosleep (struct thread * *td*, struct timespec * *rqt*, struct timespec * *rmt*)

Definition at line 362 of file kern_time.c.

References getnanouptime(), nanowait, and tvtohz().

Referenced by nanosleep().

Here is the call graph for this function:



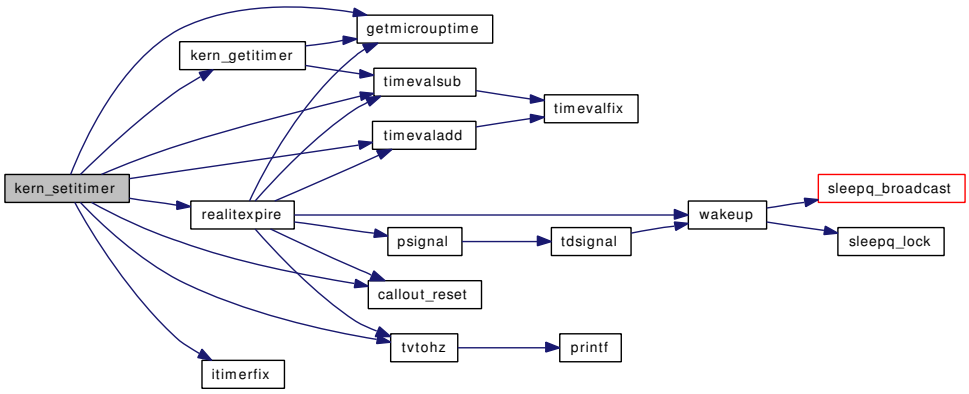
9.66.2.27 int kern_setitimer (struct thread * *td*, u_int *which*, struct itimerval * *aitv*, struct itimerval * *oitv*)

Definition at line 628 of file kern_time.c.

References callout_reset(), getmicrouptime(), itimerfix(), kern_getitimer(), realitexpire(), sched_lock, timevaladd(), timevalsub(), and tvtohz().

Referenced by setitimer().

Here is the call graph for this function:



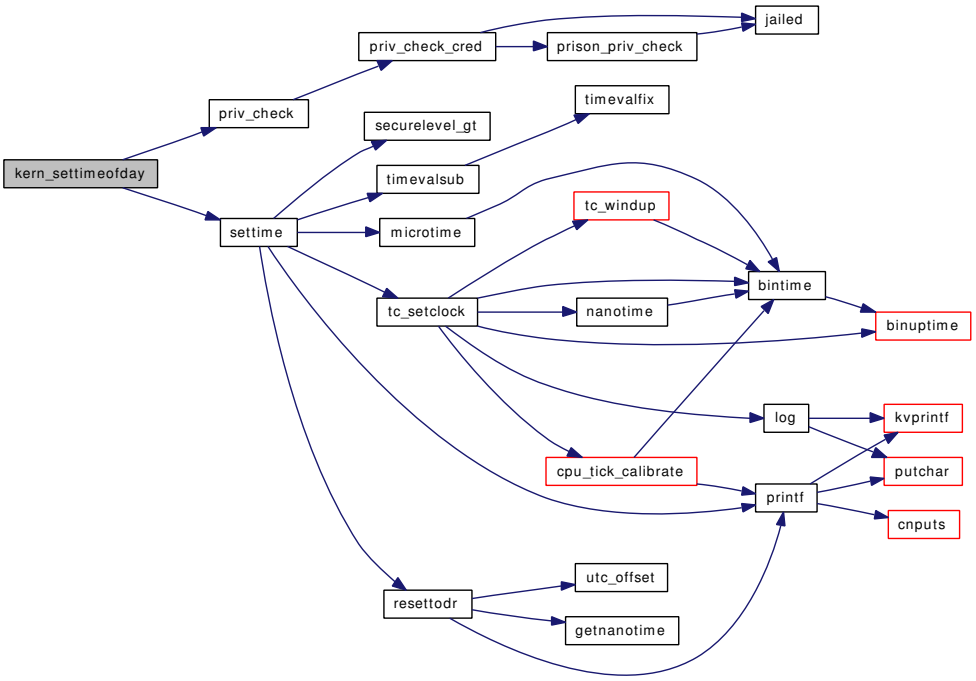
9.66.2.28 int kern_settimeofday (struct thread * *td*, struct timeval * *tv*, struct timezone * *tzp*)

Definition at line 497 of file kern_time.c.

References priv_check(), settime(), tz_dsttime, and tz_minuteswest.

Referenced by settimeofday().

Here is the call graph for this function:



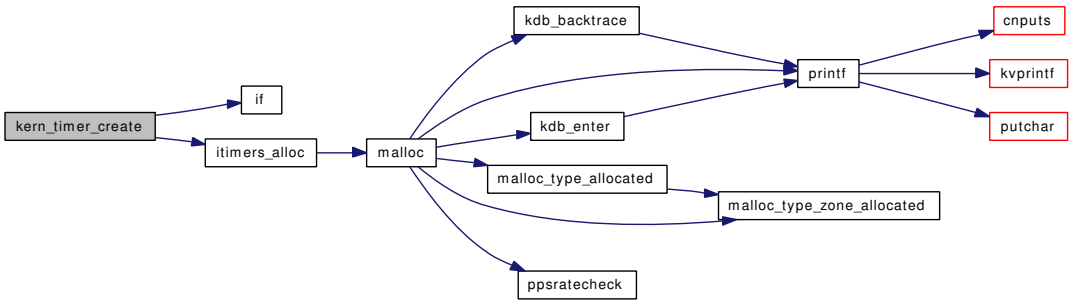
9.66.2.29 static int kern_timer_create (struct thread *, clockid_t, struct sigevent *, int *, int) [static]

Definition at line 983 of file kern_time.c.

References CLOCK_CALL, if(), itimer_zone, itimers_alloc(), MAX_CLOCKS, and posix_clocks.

Referenced by ktimer_create().

Here is the call graph for this function:



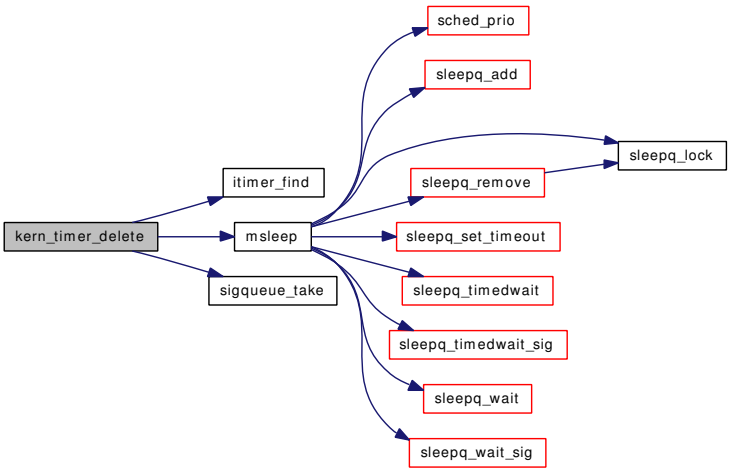
9.66.2.30 static int kern_timer_delete (struct thread *, int) [static]

Definition at line 1122 of file kern_time.c.

References CLOCK_CALL, itimer_find(), itimer_zone, msleep(), and sigqueue_take().

Referenced by itimers_event_hook_exit(), ktimer_create(), and ktimer_delete().

Here is the call graph for this function:

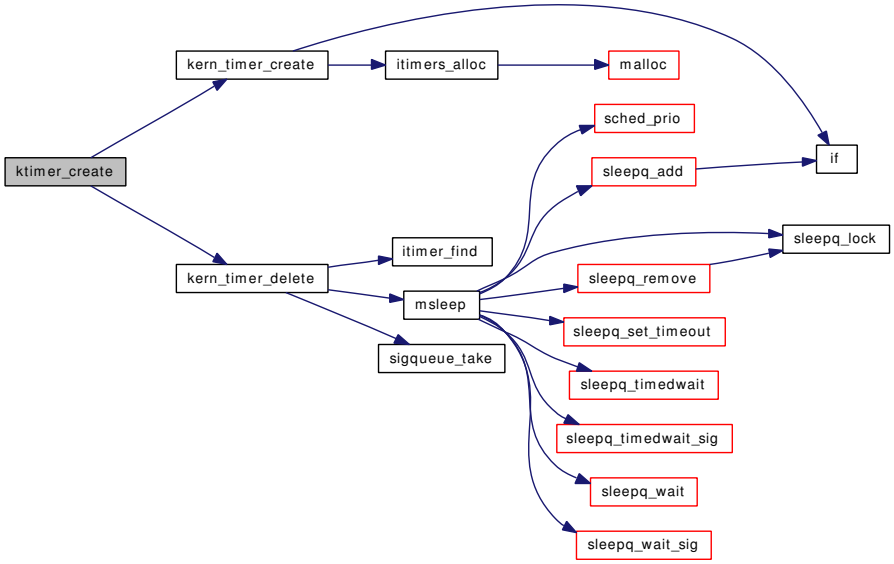


9.66.2.31 int ktimer_create (struct thread * *td*, struct ktimer_create_args * *uap*)

Definition at line 958 of file kern_time.c.

References ktimer_create_args::clock_id, ktimer_create_args::evp, kern_timer_create(), kern_timer_-delete(), and ktimer_create_args::timerid.

Here is the call graph for this function:

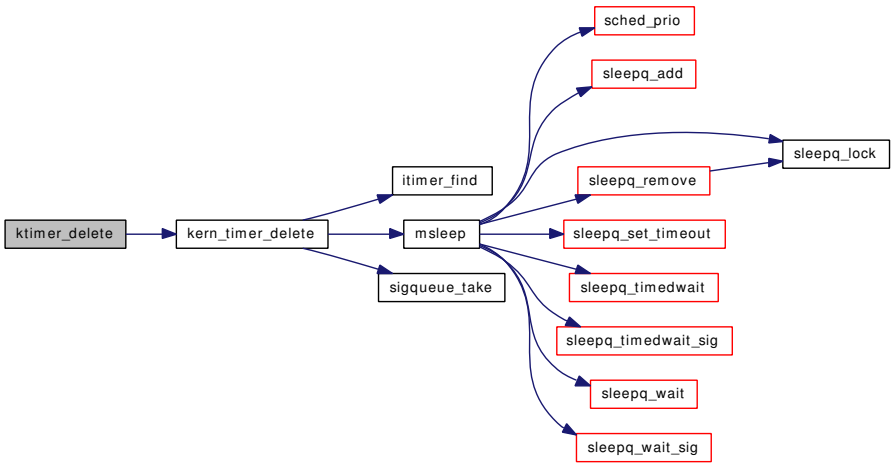


9.66.2.32 int ktimer_delete (struct thread * *td*, struct ktimer_delete_args * *uap*)

Definition at line 1098 of file kern_time.c.

References kern_timer_delete(), and ktimer_delete_args::timerid.

Here is the call graph for this function:

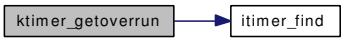


9.66.2.33 int ktimer_getoverrun (struct thread * *td*, struct ktimer_getoverrun_args * *uap*)

Definition at line 1237 of file kern_time.c.

References itimer_find().

Here is the call graph for this function:

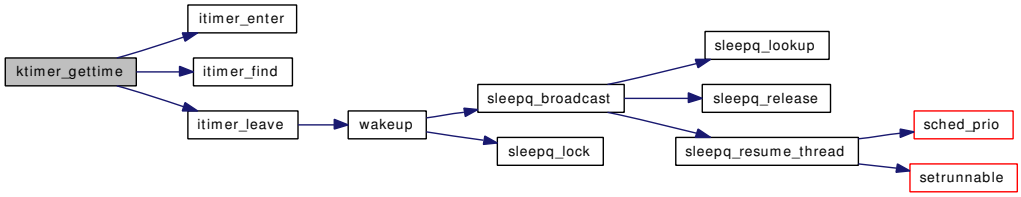


9.66.2.34 int ktimer_gettime (struct thread * *td*, struct ktimer_gettime_args * *uap*)

Definition at line 1205 of file kern_time.c.

References CLOCK_CALL, itimer_enter(), itimer_find(), itimer_leave(), ktimer_gettime_args::timerid, and ktimer_gettime_args::value.

Here is the call graph for this function:

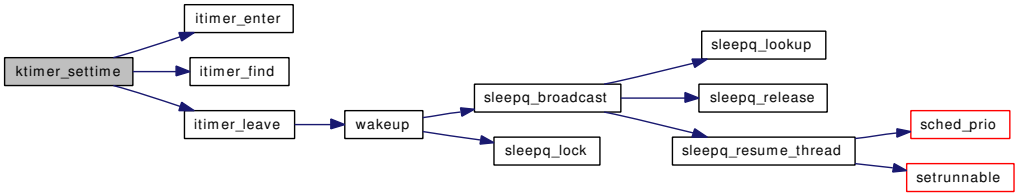


9.66.2.35 int ktimer_settime (struct thread * *td*, struct ktimer_settime_args * *uap*)

Definition at line 1163 of file kern_time.c.

References CLOCK_CALL, ktimer_settime_args::flags, itimer_enter(), itimer_find(), itimer_leave(), ktimer_settime_args::ovalue, ktimer_settime_args::timerid, and ktimer_settime_args::value.

Here is the call graph for this function:

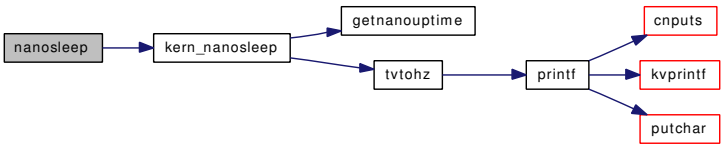


9.66.2.36 int nanosleep (struct thread * *td*, struct nanosleep_args * *uap*)

Definition at line 410 of file kern_time.c.

References kern_nanosleep(), nanosleep_args::rmtp, and nanosleep_args::rqtp.

Here is the call graph for this function:



9.66.2.37 static void no_lease_updatetime (int) [static]

Definition at line 114 of file kern_time.c.

9.66.2.38 int ppsratecheck (struct timeval * *lasttime*, int * *curpps*, int *maxpps*)

Definition at line 856 of file kern_time.c.

References hz, and ticks.

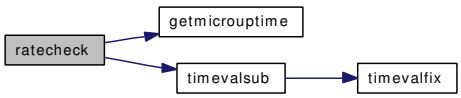
Referenced by falloc(), fork1(), malloc(), and pipespace_new().

9.66.2.39 int ratecheck (struct timeval * *lasttime*, const struct timeval * *mininterval*)

Definition at line 819 of file kern_time.c.

References getmicrouptime(), and timevalsub().

Here is the call graph for this function:



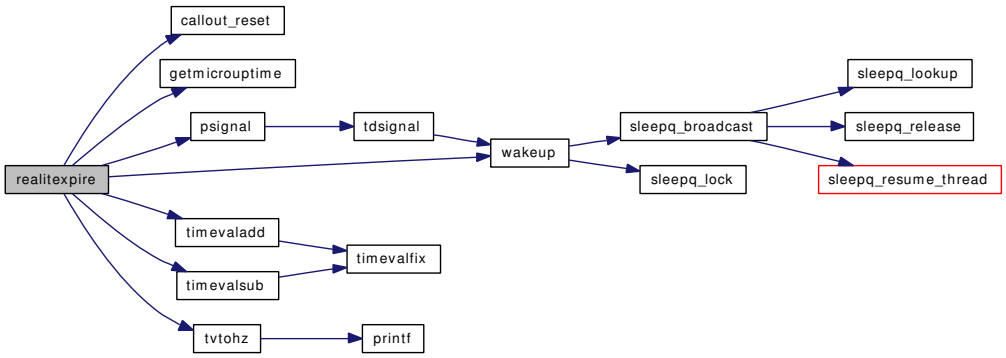
9.66.2.40 void realitexpire (void * *arg*)

Definition at line 687 of file kern_time.c.

References callout_reset(), getmicrouptime(), psignal(), timevaladd(), timevalsub(), tvtohz(), and wakeup().

Referenced by kern_setitimer().

Here is the call graph for this function:



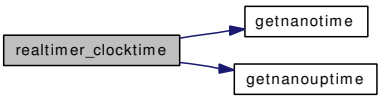
9.66.2.41 static void realtimer_clocktime (clockid_t, struct timespec *) [static]

Definition at line 1345 of file kern_time.c.

References getnanotime(), and getnanouptime().

Referenced by realtimer_expire(), realtimer_gettime(), and realtimer_settime().

Here is the call graph for this function:



9.66.2.42 static int realtimer_create (struct itimer *) [static]

Definition at line 1258 of file kern_time.c.

References callout_init_mtx().

Referenced by itimer_start().

Here is the call graph for this function:

9.66.2.43 static int realtimer_delete (struct itimer *) [static]

Definition at line 1265 of file kern_time.c.

Referenced by itimer_start().

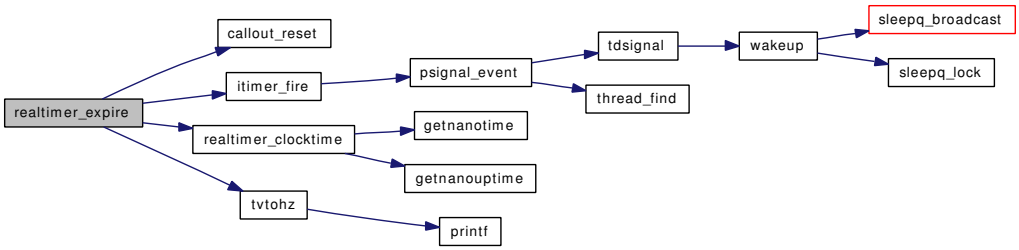
9.66.2.44 static void realtimer_expire (void *) [static]

Definition at line 1383 of file kern_time.c.

References callout_reset(), itimer_fire(), realtimer_clocktime(), and tvtohz().

Referenced by realtimer_settime().

Here is the call graph for this function:



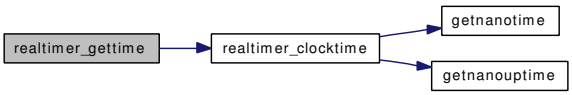
9.66.2.45 static int realtimer_gettime (struct itimer *, struct itimerspec *) [static]

Definition at line 1276 of file kern_time.c.

References realtimer_clocktime().

Referenced by itimer_start(), and realtimer_settime().

Here is the call graph for this function:



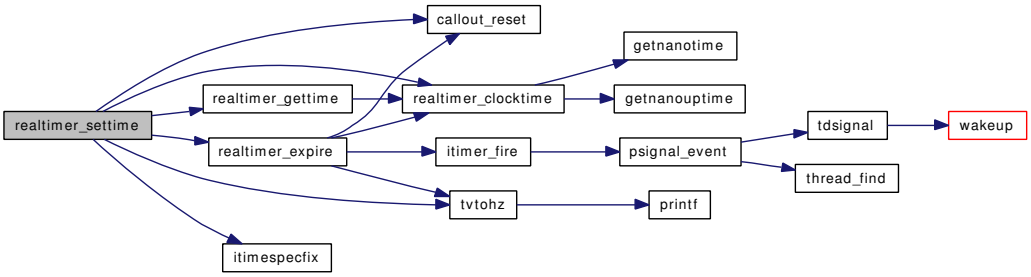
9.66.2.46 static int realtimer_settime (struct itimer *, int, struct itimerspec *, struct itimerspec *) [static]

Definition at line 1297 of file kern_time.c.

References callout_reset(), itimespecfix(), realtimer_clocktime(), realtimer_expire(), realtimer_gettime(), and tvtohz().

Referenced by itimer_start().

Here is the call graph for this function:



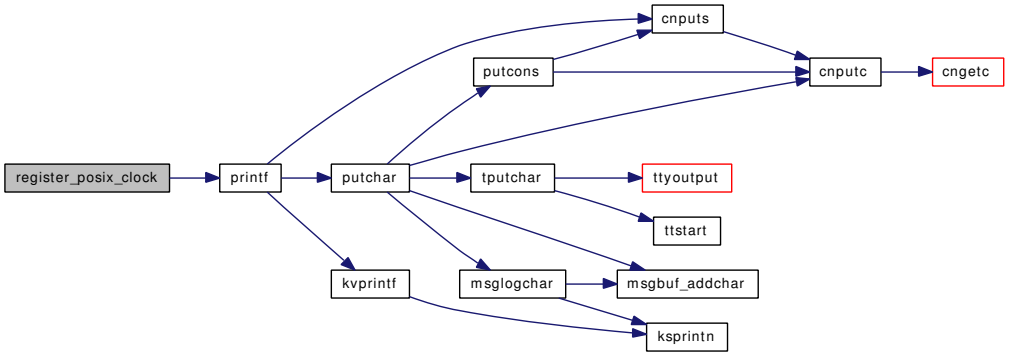
9.66.2.47 int register_posix_clock (int, struct kclock *)

Definition at line 901 of file kern_time.c.

References MAX_CLOCKS, posix_clocks, and printf().

Referenced by itimer_start().

Here is the call graph for this function:

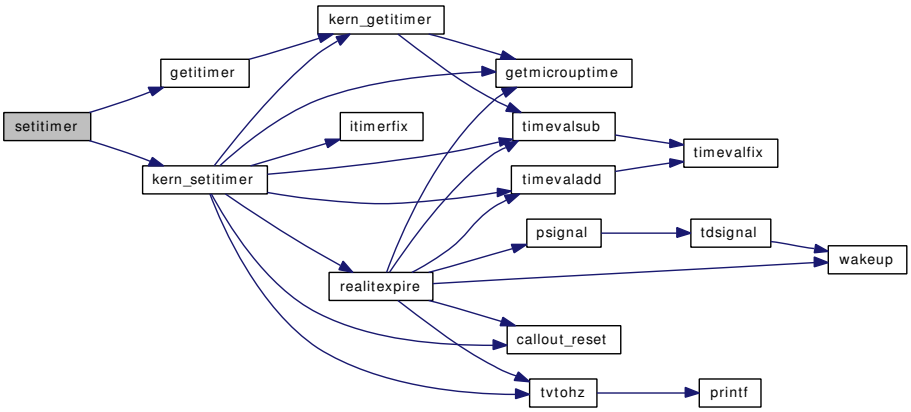


9.66.2.48 int setitimer (struct thread * *td*, struct setitimer_args * *uap*)

Definition at line 609 of file kern_time.c.

References getitimer(), setitimer_args::itv, kern_setitimer(), setitimer_args::oitv, and setitimer_-args::which.

Here is the call graph for this function:



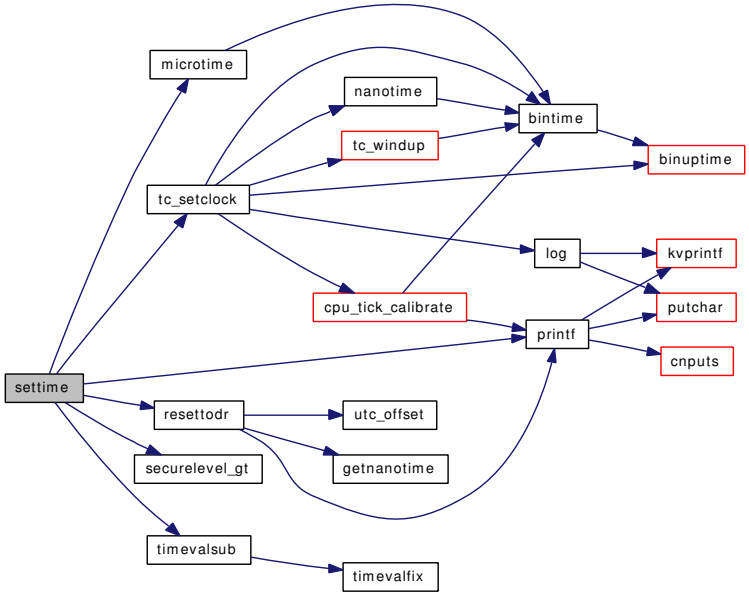
9.66.2.49 static int settime (struct thread *, struct timeval *) [static]

Definition at line 122 of file kern_time.c.

References Giant, lease_updatetime, microtime(), printf(), resettodr(), securelevel_gt(), tc_setclock(), and timevalsub().

Referenced by kern_clock_settime(), and kern_settimeofday().

Here is the call graph for this function:

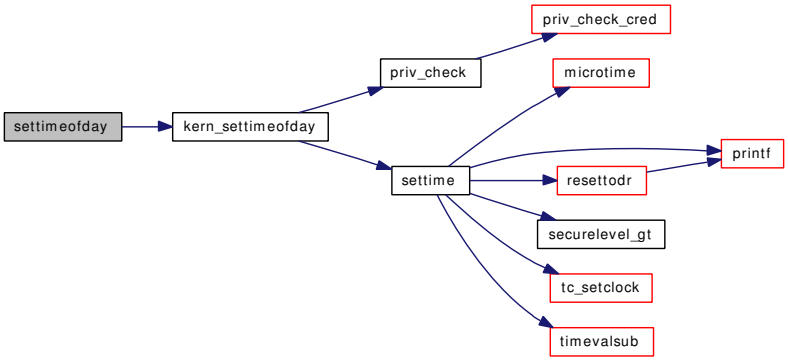


9.66.2.50 int settimeofday (struct thread * *td*, struct settimeofday_args * *uap*)

Definition at line 473 of file kern_time.c.

References kern_settimeofday(), settimeofday_args::tv, and settimeofday_args::tzp.

Here is the call graph for this function:



9.66.2.51 SYSINIT (posix_timer, SI_SUB_P1003_1B, SI_ORDER_FIRST+ *4*, itimer_start, NULL)

9.66.2.52 void timevaladd (struct timeval * *t1*, const struct timeval * *t2*)

Definition at line 784 of file kern_time.c.

References timevalfix().

Referenced by acct_process(), fill_kinfo_proc_only(), kern_clock_gettime(), kern_select(), kern_setitimer(), kqueue_scan(), poll(), and realitexpire().

Here is the call graph for this function:

9.66.2.53 static void timevalfix (struct timeval *) [static]

Definition at line 802 of file kern_time.c.

Referenced by timevaladd(), and timevalsub().

9.66.2.54 void timevalsub (struct timeval * *t1*, const struct timeval * *t2*)

Definition at line 793 of file kern_time.c.

References timevalfix().

Referenced by acct_process(), kern_getitimer(), kern_select(), kern_setitimer(), kqueue_scan(), poll(), ratecheck(), realitexpire(), and settime().

Here is the call graph for this function:

9.66.3 Variable Documentation

9.66.3.1 uma_zone_t itimer_zone = NULL [static]

Definition at line 67 of file kern_time.c.

Referenced by itimer_start(), kern_timer_create(), and kern_timer_delete().

9.66.3.2 void(*) lease_updatetime(int) = no_lease_updatetime

Definition at line 119 of file kern_time.c.

Referenced by settime().

9.66.3.3 int nanowait [static]

Definition at line 359 of file kern_time.c.

Referenced by kern_nanosleep().

9.66.3.4 struct kclock posix_clocks[MAX_CLOCKS] [static]

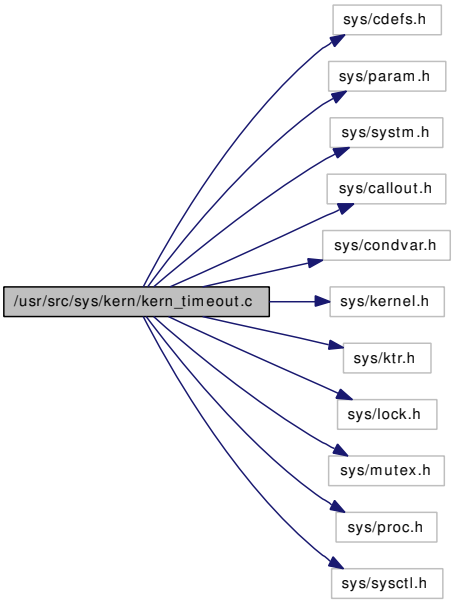
Definition at line 66 of file kern_time.c.

Referenced by itimers_event_hook_exit(), kern_timer_create(), and register_posix_clock().

9.67 /usr/src/sys/kern/kern_timeout.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/callout.h>
#include <sys/condvar.h>
#include <sys/kernel.h>
#include <sys/ktr.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/sysctl.h>
```

Include dependency graph for kern_timeout.c:



Defines

- #define MAX_SOFTCLOCK_STEPS 100

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_timeout.c,v 1.102 2006/10/11 14:57:03 glebius Exp $")
- SYSCTL_INT (_debug, OID_AUTO, to_avg_depth, CTLFLAG_RD,&avg_depth, 0,"Average number of items examined per softclock call. Units = 1/1000")
- SYSCTL_INT (_debug, OID_AUTO, to_avg_gcalls, CTLFLAG_RD,&avg_gcalls, 0,"Average number of Giant callouts made per softclock call. Units = 1/1000")

- SYSCTL_INT (_debug, OID_AUTO, to_avg_mtxcalls, CTLFLAG_RD,&avg_mtxcalls, 0,"Average number of mtx callouts made per softclock call. Units = 1/1000")
- SYSCTL_INT (_debug, OID_AUTO, to_avg_mpcalls, CTLFLAG_RD,&avg_mpcalls, 0,"Average number of MP callouts made per softclock call. Units = 1/1000")
- caddr_t kern_timeout_callwheel_alloc (caddr_t v)
- void kern_timeout_callwheel_init (void)
- void softclock (void *dummy)
- callout_handle timeout (timeout_t *ftn, void *arg, int to_ticks)
- void untimeout (timeout_t *ftn, void *arg, struct callout_handle handle)
- void callout_handle_init (struct callout_handle *handle)
- int callout_reset (struct callout *c, int to_ticks, void(*ftn)(void *), void *arg)
- int _callout_stop_safe (struct callout *c, int safe)
- void callout_init (struct callout *c, int mpsafe)
- void callout_init_mtx (struct callout *c, struct mtx *mtx, int flags)

Variables

- static int avg_depth
- static int avg_gcalls
- static int avg_mtxcalls
- static int avg_mpcalls
- callout * callout
- callout_list callfree
- int callwheelsize
- int callwheelbits
- int callwheelmask
- callout_tailq * callwheel
- int softticks
- mtx callout_lock
- static struct callout * nextsoftcheck
- static struct callout * curr_callout
- static int curr_cancelled
- static int callout_wait

9.67.1 Define Documentation

9.67.1.1 #define MAX_SOFTCLOCK_STEPS 100

Referenced by softclock().

9.67.2 Function Documentation

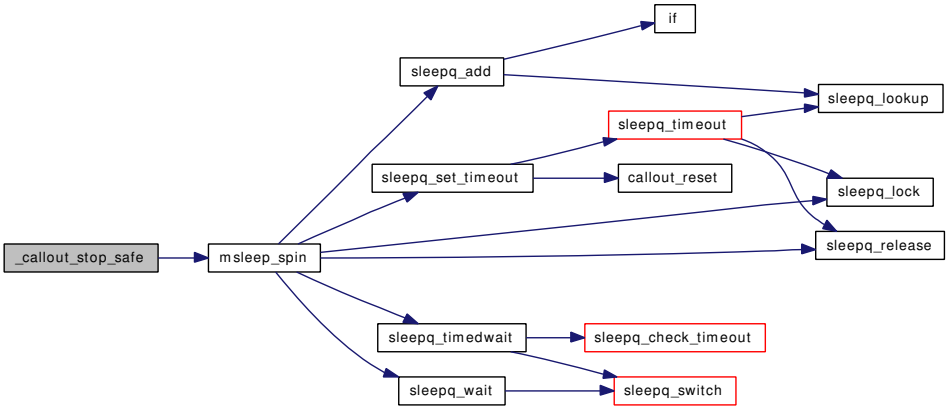
9.67.2.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_timeout. *c*, v 1.102 2006/10/11 14:57:03 glebius Exp $")

9.67.2.2 int _callout_stop_safe (struct callout * *c*, int *safe*)

Definition at line 478 of file kern_timeout.c.

References callfree, callout_lock, callout_wait, callwheel, callwheelmask, curr_callout, curr_cancelled, msleep_spin(), and nextsoftcheck.

Here is the call graph for this function:



9.67.2.3 void callout_handle_init (struct callout_handle * *handle*)

Definition at line 381 of file kern_timeout.c.

9.67.2.4 void callout_init (struct callout * *c*, int *mpsafe*)

Definition at line 565 of file kern_timeout.c.

References Giant.

Referenced by constty_set(), domaininit(), filt_timerattach(), fork1(), kern_timeout_callwheel_init(), logopen(), proc0_init(), sched_setup(), synch_setup(), and thread_link().

9.67.2.5 void callout_init_mtx (struct callout * *c*, struct mtx * *mtx*, int *flags*)

Definition at line 580 of file kern_timeout.c.

Referenced by realtimer_create().

9.67.2.6 int callout_reset (struct callout * *c*, int *to_ticks*, void (*)(void *) *ftn*, void * *arg*)

Definition at line 403 of file kern_timeout.c.

References callout_lock, callout_wait, callwheel, callwheelmask, curr_callout, curr_cancelled, nextsoftcheck, and ticks.

Referenced by constty_timeout(), domainfinalize(), filt_timerattach(), filt_timerexpire(), kern_setitimer(), lboltcb(), loadav(), logopen(), logtimeout(), pffasttimo(), pfslowtimo(), realitexpire(), realtimer_expire(), realtimer_settime(), roundrobin(), sleepq_set_timeout(), and timeout().

9.67.2.7 caddr_t kern_timeout_callwheel_alloc (caddr_t *v*)

Definition at line 105 of file kern_timeout.c.

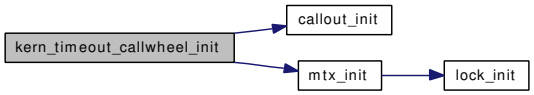
References callout, callwheel, callwheelbits, callwheelmask, callwheelsize, and ncallout.

9.67.2.8 void kern_timeout_callwheel_init (void)

Definition at line 131 of file kern_timeout.c.

References callfree, callout, callout_init(), callout_lock, callwheel, callwheelsize, mtx_init(), and ncallout.

Here is the call graph for this function:



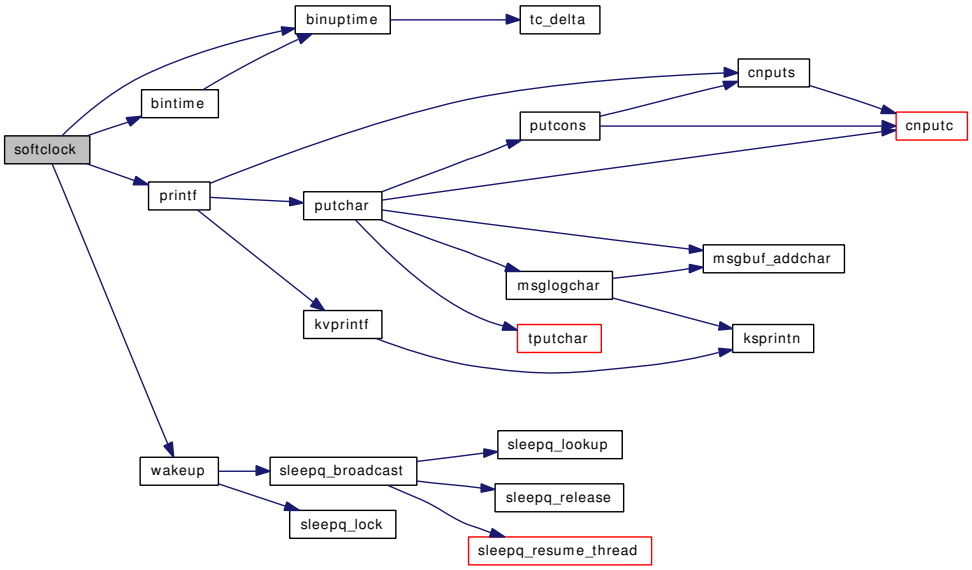
9.67.2.9 void softclock (void * *dummy*)

Definition at line 164 of file kern_timeout.c.

References avg_depth, avg_gcalls, avg_mpcalls, avg_mtxcalls, bintime(), binuptime(), callfree, callout, callout_lock, callout_wait, callwheel, callwheelmask, curr_callout, curr_cancelled, Giant, MAX_SOFTCLOCK_STEPS, nextsoftcheck, printf(), softticks, ticks, and wakeup().

Referenced by start_softintr().

Here is the call graph for this function:



9.67.2.10 SYSCTL_INT (_debug, OID_AUTO, to_avg_mpcalls, CTLFLAG_RD, & *avg_mpcalls*, 0)

9.67.2.11 SYSCTL_INT (_debug, OID_AUTO, to_avg_mtxcalls, CTLFLAG_RD, & *avg_mtxcalls*, 0)

9.67.2.12 SYSCTL_INT (_debug, OID_AUTO, to_avg_gcalls, CTLFLAG_RD, & *avg_gcalls*, 0)

9.67.2.13 SYSCTL_INT (_debug, OID_AUTO, to_avg_depth, CTLFLAG_RD, & *avg_depth*, 0)

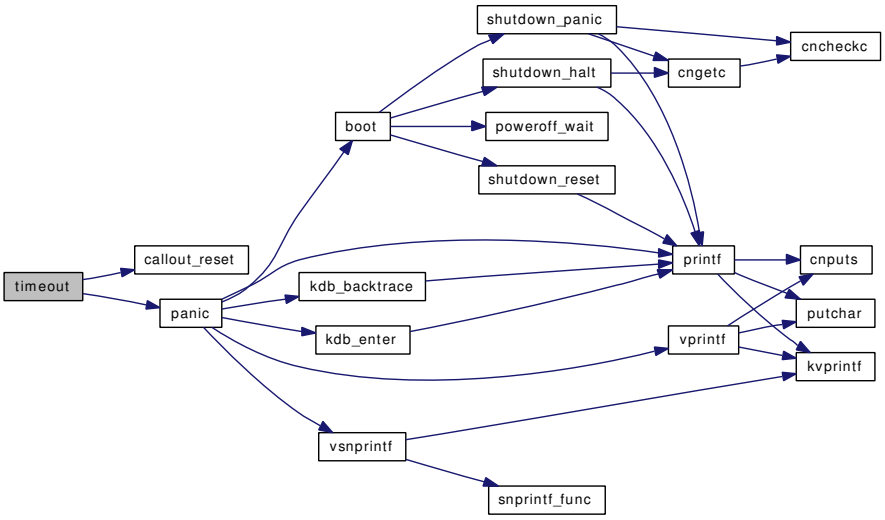
9.67.2.14 struct callout_handle timeout (timeout_t * *ftn*, void * *arg*, int *to_ticks*)

Definition at line 335 of file kern_timeout.c.

References callfree, callout, callout_lock, callout_reset(), and panic().

Referenced by kqueue_scan(), sigtimedwait(), sysctl_kern_ttys(), and ttydtrwaitstart().

Here is the call graph for this function:



9.67.2.15 void untimeout (timeout_t * *ftn*, void * *arg*, struct callout_handle *handle*)

Definition at line 360 of file kern_timeout.c.

References callout_lock.

9.67.3 Variable Documentation

9.67.3.1 int avg_depth [static]

Definition at line 51 of file kern_timeout.c.

Referenced by softclock().

9.67.3.2 int avg_gcalls [static]

Definition at line 54 of file kern_timeout.c.

Referenced by softclock().

9.67.3.3 int avg_mpcalls [static]

Definition at line 60 of file kern_timeout.c.

Referenced by softclock().

9.67.3.4 int avg_mtxcalls [static]

Definition at line 57 of file kern_timeout.c.

Referenced by softclock().

9.67.3.5 struct callout_list callfree

Definition at line 70 of file kern_timeout.c.

Referenced by _callout_stop_safe(), kern_timeout_callwheel_init(), softclock(), and timeout().

9.67.3.6 struct callout* callout

Definition at line 69 of file kern_timeout.c.

Referenced by filt_timerattach(), filt_timerdetach(), filt_timerexpire(), fork_exit(), kern_timeout_callwheel_alloc(), kern_timeout_callwheel_init(), softclock(), and timeout().

9.67.3.7 struct mtx callout_lock

Definition at line 74 of file kern_timeout.c.

Referenced by _callout_stop_safe(), callout_reset(), hardclock(), kern_timeout_callwheel_init(), softclock(), timeout(), and untimeout().

9.67.3.8 int callout_wait [static]

Definition at line 96 of file kern_timeout.c.

Referenced by _callout_stop_safe(), callout_reset(), and softclock().

9.67.3.9 struct callout_tailq* callwheel

Definition at line 72 of file kern_timeout.c.

Referenced by _callout_stop_safe(), callout_reset(), hardclock(), kern_timeout_callwheel_alloc(), kern_timeout_callwheel_init(), and softclock().

9.67.3.10 int callwheelbits

Definition at line 71 of file kern_timeout.c.

Referenced by kern_timeout_callwheel_alloc().

9.67.3.11 int callwheelmask

Definition at line 71 of file kern_timeout.c.

Referenced by _callout_stop_safe(), callout_reset(), hardclock(), kern_timeout_callwheel_alloc(), and softclock().

9.67.3.12 int callwheelsize

Definition at line 71 of file kern_timeout.c.

Referenced by kern_timeout_callwheel_alloc(), and kern_timeout_callwheel_init().

9.67.3.13 struct callout* curr_callout [static]

Locked by callout_lock: curr_callout - If a callout is in progress, it is curr_callout. If curr_callout is non-NULL, threads waiting in callout_drain() will be woken up as soon as the relevant callout completes. curr_cancelled - Changing to 1 with both callout_lock and c_mtx held guarantees that the current callout will not run. The softclock() function sets this to 0 before it drops callout_lock to acquire c_mtx, and it calls the handler only if curr_cancelled is still 0 after c_mtx is successfully acquired. callout_wait - If a thread is waiting in callout_drain(), then callout_wait is nonzero. Set only when curr_callout is non-NULL.

Definition at line 94 of file kern_timeout.c.

Referenced by _callout_stop_safe(), callout_reset(), and softclock().

9.67.3.14 int curr_cancelled [static]

Definition at line 95 of file kern_timeout.c.

Referenced by _callout_stop_safe(), callout_reset(), and softclock().

9.67.3.15 struct callout* nextsoftcheck [static]

Definition at line 76 of file kern_timeout.c.

Referenced by _callout_stop_safe(), callout_reset(), and softclock().

9.67.3.16 int softticks

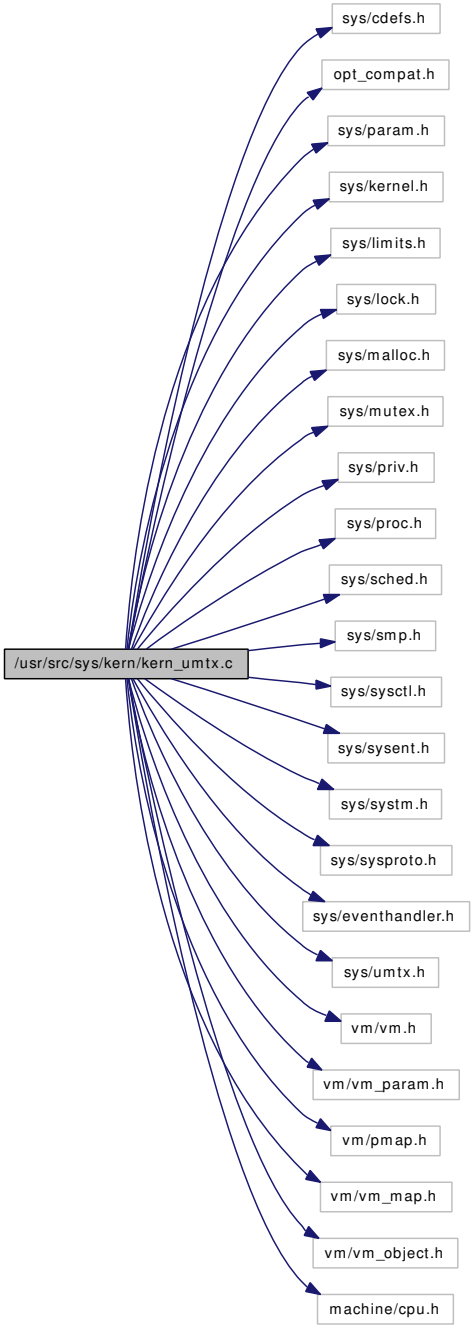
Definition at line 73 of file kern_timeout.c.

Referenced by hardclock(), and softclock().

9.68 /usr/src/sys/kern/kern_umtx.c File Reference

```
#include <sys/cdefs.h>
#include "opt_compat.h"
#include <sys/param.h>
#include <sys/kernel.h>
#include <sys/limits.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/sched.h>
#include <sys/smp.h>
#include <sys/sysctl.h>
#include <sys/sysent.h>
#include <sys/systm.h>
#include <sys/sysproto.h>
#include <sys/eventhandler.h>
#include <sys/umtx.h>
#include <vm/vm.h>
#include <vm/vm_param.h>
#include <vm/pmap.h>
#include <vm/vm_map.h>
#include <vm/vm_object.h>
#include <machine/cpu.h>
```

Include dependency graph for kern_umtx.c:



Data Structures

- struct umtx_key
- struct umtx_pi
- struct umtx_q
- struct umtxq_chain

Defines

- #define TYPE_SIMPLE_LOCK 0
- #define TYPE_SIMPLE_WAIT 1
- #define TYPE_NORMAL_UMUTEX 2
- #define TYPE_PI_UMUTEX 3
- #define TYPE_PP_UMUTEX 4
- #define TYPE_CV 5
- #define UQF_UMTXQ 0x0001
- #define UMTXQ_LOCKED_ASSERT(uc) mtx_assert(&(uc) → uc_lock, MA_OWNED)
- #define UPRI(td)
- #define GOLDEN_RATIO_PRIME 2654404609U
- #define UMTX_CHAINS 128
- #define UMTX_SHIFTS (__WORD_BIT - 7)
- #define THREAD_SHARE 0
- #define PROCESS_SHARE 1
- #define AUTO_SHARE 2
- #define GET_SHARE(flags) (((flags) & USYNC_PROCESS_SHARED) == 0 ? THREAD_SHARE : PROCESS_SHARE)

Typedefs

- typedef int(∗) _umtx_op_func (struct thread ∗td, struct _umtx_op_args ∗uap)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_umtx.c,v 1.58 2006/12/20 04:40:39 davidxu Exp $")
- TAILQ_HEAD (umtxq_head, umtx_q)
- static MALLOC_DEFINE (M_UMTX,"umtx","UMTX queue memory")
- SYSCTL_NODE (_debug, OID_AUTO, umtx, CTLFLAG_RW, 0,"umtx debug")
- SYSCTL_INT (_debug_umtx, OID_AUTO, umtx_pi_allocated, CTLFLAG_RD,&umtx_pi_allocated, 0,"Allocated umtx_pi")
- SYSCTL_DECL (_kern_threads)
- SYSCTL_INT (_kern_threads, OID_AUTO, umtx_dflt_spins, CTLFLAG_RW,&umtx_dflt_spins, 0,"default umtx spin count")
- SYSCTL_INT (_kern_threads, OID_AUTO, umtx_max_spins, CTLFLAG_RW,&umtx_max_spins, 0,"max umtx spin count")
- static void umtxq_sysinit (void ∗)
- static void umtxq_hash (struct umtx_key ∗key)
- static struct umtxq_chain ∗ umtxq_getchain (struct umtx_key ∗key)
- static void umtxq_lock (struct umtx_key ∗key)
- static void umtxq_unlock (struct umtx_key ∗key)
- static void umtxq_busy (struct umtx_key ∗key)
- static void umtxq_unbusy (struct umtx_key ∗key)
- static void umtxq_insert (struct umtx_q ∗uq)
- static void umtxq_remove (struct umtx_q ∗uq)
- static int umtxq_sleep (struct umtx_q ∗uq, const char ∗wmesg, int timo)
- static int umtxq_count (struct umtx_key ∗key)
- static int umtxq_signal (struct umtx_key ∗key, int nr_wakeup)
- static int umtx_key_match (const struct umtx_key ∗k1, const struct umtx_key ∗k2)

- static int umtx_key_get (void *addr, int type, int share, struct umtx_key *key)
- static void umtx_key_release (struct umtx_key *key)
- static struct umtx_pi * umtx_pi_alloc (int)
- static void umtx_pi_free (struct umtx_pi *pi)
- static int do_unlock_pp (struct thread *td, struct umutex *m, uint32_t flags)
- static void umtx_thread_cleanup (struct thread *td)
- static void umtx_exec_hook (void *arg __unused, struct proc *p __unused, struct image_params *imgp __unused)
- SYSINIT (umtx, SI_SUB_EVENTHANDLER+1, SI_ORDER_MIDDLE, umtxq_sysinit, NULL)
- static void umtxq_sysinit (void *arg __unused)
- umtx_q * umtxq_alloc (void)
- void umtxq_free (struct umtx_q *uq)
- static int umtxq_count_pi (struct umtx_key *key, struct umtx_q **first)
- static void umtxq_signal_thread (struct umtx_q *uq)
- static int _do_lock_umtx (struct thread *td, struct umtx *umtx, u_long id, int timo)
- static int do_lock_umtx (struct thread *td, struct umtx *umtx, u_long id, struct timespec *timeout)
- static int do_unlock_umtx (struct thread *td, struct umtx *umtx, u_long id)
- static int do_wait (struct thread *td, void *addr, u_long id, struct timespec *timeout, int compat32)
- int kern_umtx_wake (struct thread *td, void *uaddr, int n_wake)
- static int _do_lock_normal (struct thread *td, struct umutex *m, uint32_t flags, int timo, int try)
- static int do_unlock_normal (struct thread *td, struct umutex *m, uint32_t flags)
- static int umtx_pi_adjust_thread (struct umtx_pi *pi, struct thread *td)
- static void umtx_propagate_priority (struct thread *td)
- static void umtx_unpropagate_priority (struct umtx_pi *pi)
- static void umtx_pi_setowner (struct umtx_pi *pi, struct thread *owner)
- static int umtx_pi_claim (struct umtx_pi *pi, struct thread *owner)
- void umtx_pi_adjust (struct thread *td, u_char oldpri)
- static int umtxq_sleep_pi (struct umtx_q *uq, struct umtx_pi *pi, uint32_t owner, const char *wmesg, int timo)
- static void umtx_pi_ref (struct umtx_pi *pi)
- static void umtx_pi_unref (struct umtx_pi *pi)
- static struct umtx_pi * umtx_pi_lookup (struct umtx_key *key)
- static void umtx_pi_insert (struct umtx_pi *pi)
- static int _do_lock_pi (struct thread *td, struct umutex *m, uint32_t flags, int timo, int try)
- static int do_unlock_pi (struct thread *td, struct umutex *m, uint32_t flags)
- static int _do_lock_pp (struct thread *td, struct umutex *m, uint32_t flags, int timo, int try)
- static int do_set_ceiling (struct thread *td, struct umutex *m, uint32_t ceiling, uint32_t *old_ceiling)
- static int _do_lock_umutex (struct thread *td, struct umutex *m, int flags, int timo, int try)
- static int do_lock_umutex (struct thread *td, struct umutex *m, struct timespec *timeout, int try)
- static int do_unlock_umutex (struct thread *td, struct umutex *m)
- static int do_cv_wait (struct thread *td, struct ucond *cv, struct umutex *m, struct timespec *timeout, u_long wflags)
- static int do_cv_signal (struct thread *td, struct ucond *cv)
- static int do_cv_broadcast (struct thread *td, struct ucond *cv)
- int _umtx_lock (struct thread *td, struct _umtx_lock_args *uap)
- int _umtx_unlock (struct thread *td, struct _umtx_unlock_args *uap)
- static int __umtx_op_lock_umtx (struct thread *td, struct _umtx_op_args *uap)
- static int __umtx_op_unlock_umtx (struct thread *td, struct _umtx_op_args *uap)
- static int __umtx_op_wait (struct thread *td, struct _umtx_op_args *uap)
- static int __umtx_op_wake (struct thread *td, struct _umtx_op_args *uap)

- static int __umtx_op_lock_umutex (struct thread *td, struct _umtx_op_args *uap)
- static int __umtx_op_trylock_umutex (struct thread *td, struct _umtx_op_args *uap)
- static int __umtx_op_unlock_umutex (struct thread *td, struct _umtx_op_args *uap)
- static int __umtx_op_set_ceiling (struct thread *td, struct _umtx_op_args *uap)
- static int __umtx_op_cv_wait (struct thread *td, struct _umtx_op_args *uap)
- static int __umtx_op_cv_signal (struct thread *td, struct _umtx_op_args *uap)
- static int __umtx_op_cv_broadcast (struct thread *td, struct _umtx_op_args *uap)
- int _umtx_op (struct thread *td, struct _umtx_op_args *uap)
- void umtx_thread_init (struct thread *td)
- void umtx_thread_fini (struct thread *td)
- void umtx_thread_alloc (struct thread *td)
- void umtx_thread_exit (struct thread *td)

Variables

- static uma_zone_t umtx_pi_zone
- static struct umtxq_chain umtxq_chains [UMTX_CHAINS]
- static int umtx_pi_allocated
- static int umtx_dflt_spins = 0
- static int umtx_max_spins = 3000
- static _umtx_op_func op_table [ ]

9.68.1 Define Documentation

9.68.1.1 #define AUTO_SHARE 2

Definition at line 183 of file kern_umtx.c.

Referenced by _do_lock_umtx(), do_unlock_umtx(), do_wait(), kern_umtx_wake(), and umtx_key_get().

9.68.1.2 #define GET_SHARE(flags) (((flags) & USYNC_PROCESS_SHARED) == 0 ? THREAD_SHARE : PROCESS_SHARE)

Definition at line 185 of file kern_umtx.c.

Referenced by _do_lock_normal(), _do_lock_pi(), _do_lock_pp(), do_cv_broadcast(), do_cv_signal(), do_cv_wait(), do_set_ceiling(), do_unlock_normal(), do_unlock_pi(), and do_unlock_pp().

9.68.1.3 #define GOLDEN_RATIO_PRIME 2654404609U

Definition at line 177 of file kern_umtx.c.

Referenced by umtxq_hash().

9.68.1.4 #define PROCESS_SHARE 1

Definition at line 182 of file kern_umtx.c.

Referenced by umtx_key_get().

9.68.1.5 #define THREAD_SHARE 0

Definition at line 181 of file kern_umtx.c.

Referenced by umtx_key_get().

9.68.1.6 #define TYPE_CV 5

Definition at line 66 of file kern_umtx.c.

Referenced by do_cv_broadcast(), do_cv_signal(), and do_cv_wait().

9.68.1.7 #define TYPE_NORMAL_UMUTEX 2

Definition at line 63 of file kern_umtx.c.

Referenced by _do_lock_normal(), and do_unlock_normal().

9.68.1.8 #define TYPE_PI_UMUTEX 3

Definition at line 64 of file kern_umtx.c.

Referenced by _do_lock_pi(), and do_unlock_pi().

9.68.1.9 #define TYPE_PP_UMUTEX 4

Definition at line 65 of file kern_umtx.c.

Referenced by _do_lock_pp(), do_set_ceiling(), and do_unlock_pp().

9.68.1.10 #define TYPE_SIMPLE_LOCK 0

Definition at line 61 of file kern_umtx.c.

Referenced by _do_lock_umtx(), and do_unlock_umtx().

9.68.1.11 #define TYPE_SIMPLE_WAIT 1

Definition at line 62 of file kern_umtx.c.

Referenced by do_wait(), and kern_umtx_wake().

9.68.1.12 #define UMTX_CHAINS 128

Definition at line 178 of file kern_umtx.c.

Referenced by umtxq_hash(), and umtxq_sysinit().

9.68.1.13 #define UMTX_SHIFTS (__WORD_BIT - 7)

Definition at line 179 of file kern_umtx.c.

Referenced by umtxq_hash().

9.68.1.14 #define UMTXQ_LOCKED_ASSERT(uc) mtx_assert(&(uc) → uc_lock, MA_OWNED)

Definition at line 162 of file kern_umtx.c.

Referenced by umtx_pi_insert(), umtx_pi_lookup(), umtx_pi_ref(), umtx_pi_unref(), umtxq_count(), umtxq_count_pi(), umtxq_insert(), umtxq_remove(), umtxq_signal(), umtxq_signal_thread(), umtxq_sleep(), and umtxq_sleep_pi().

9.68.1.15 #define UPRI(td)

Value:

```
(((td)->td_user_pri >= PRI_MIN_TIMESHARE &&\
                         (td)->td_user_pri <= PRI_MAX_TIMESHARE) ?\
                        PRI_MAX_TIMESHARE : (td)->td_user_pri)
```

Definition at line 173 of file kern_umtx.c.

Referenced by _do_lock_pp(), do_unlock_pi(), do_unlock_pp(), umtx_pi_adjust(), umtx_pi_adjust_thread(), umtx_pi_claim(), umtx_propagate_priority(), umtx_unpropagate_priority(), and umtxq_sleep_pi().

9.68.1.16 #define UQF_UMTXQ 0x0001

Referenced by do_cv_wait(), do_wait(), umtxq_insert(), umtxq_remove(), umtxq_sleep(), and umtxq_sleep_pi().

9.68.2 Typedef Documentation

9.68.2.1 typedef int(∗) _umtx_op_func(struct thread ∗td, struct _umtx_op_args ∗uap)

Definition at line 2518 of file kern_umtx.c.

9.68.3 Function Documentation

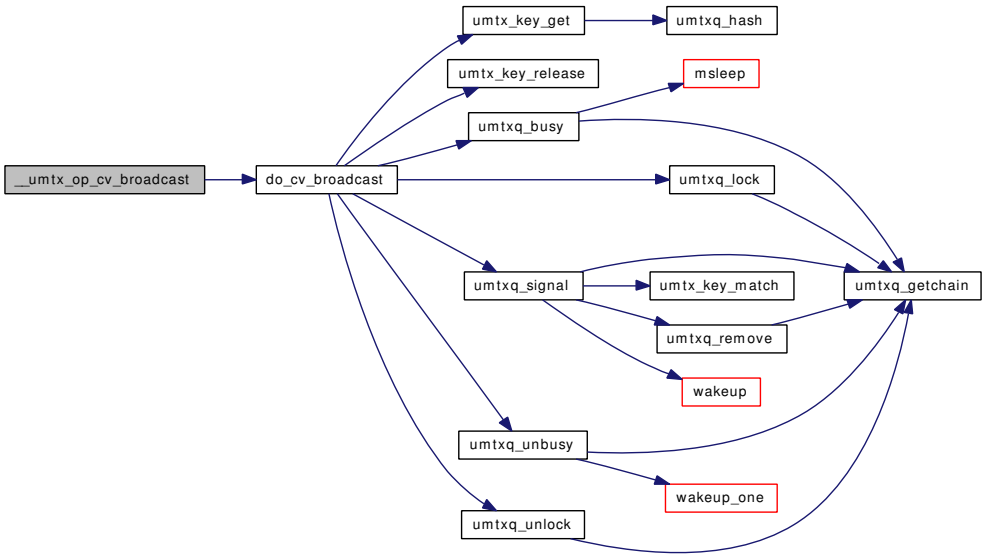
9.68.3.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_umtx. *c*, v 1.58 2006/12/20 04:40:39 davidxu Exp $")

9.68.3.2 static int __umtx_op_cv_broadcast (struct thread ∗ *td*, struct _umtx_op_args ∗ *uap*) [static]

Definition at line 2513 of file kern_umtx.c.

References do_cv_broadcast().

Here is the call graph for this function:

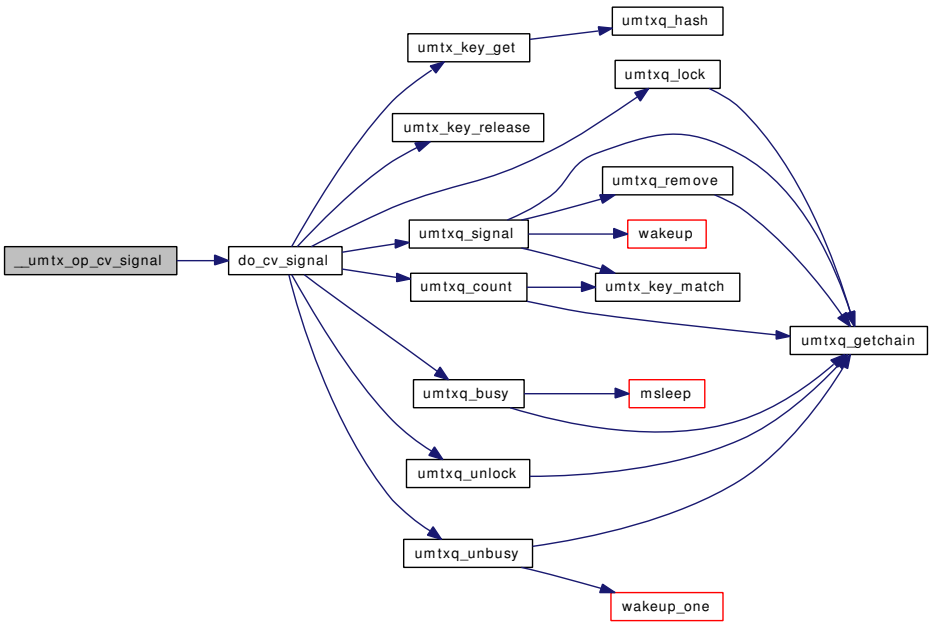


9.68.3.3 static int __umtx_op_cv_signal (struct thread * *td*, struct _umtx_op_args * *uap*) [static]

Definition at line 2507 of file kern_umtx.c.

References do_cv_signal().

Here is the call graph for this function:

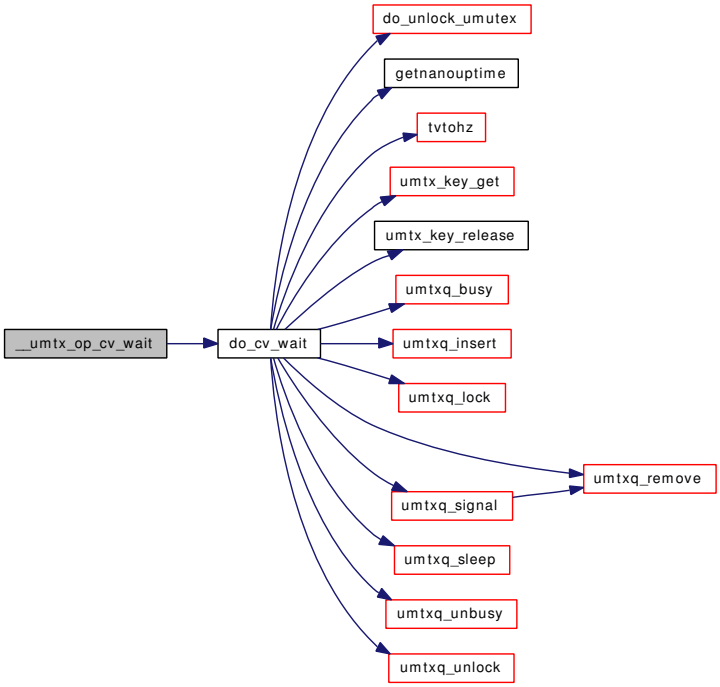


9.68.3.4 static int __umtx_op_cv_wait (struct thread * *td*, struct _umtx_op_args * *uap*) [static]

Definition at line 2484 of file kern_umtx.c.

References do_cv_wait().

Here is the call graph for this function:

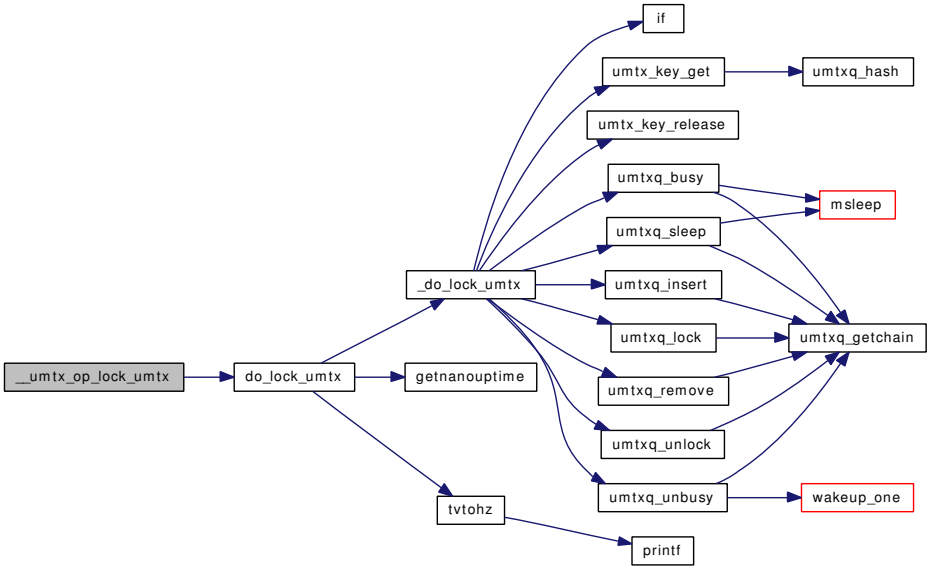


9.68.3.5 static int __umtx_op_lock_umtx (struct thread * *td*, struct _umtx_op_args * *uap*) [static]

Definition at line 2389 of file kern_umtx.c.

References do_lock_umtx().

Here is the call graph for this function:

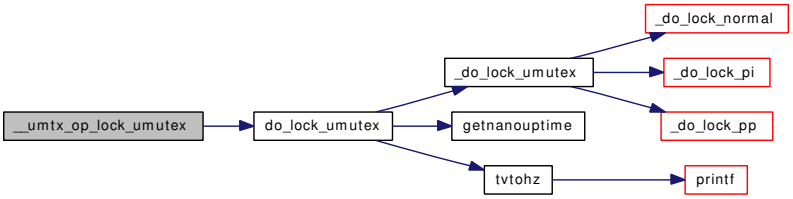


9.68.3.6 static int __umtx_op_lock_umutex (struct thread * *td*, struct _umtx_op_args * *uap*) [static]

Definition at line 2443 of file kern_umtx.c.

References do_lock_umutex().

Here is the call graph for this function:

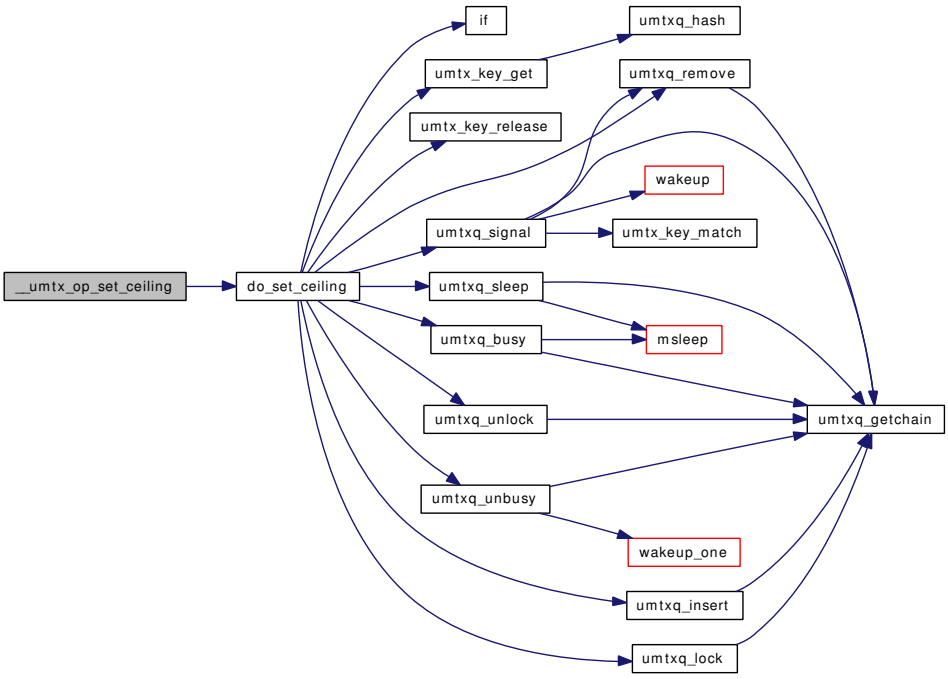


9.68.3.7 static int __umtx_op_set_ceiling (struct thread * *td*, struct _umtx_op_args * *uap*) [static]

Definition at line 2478 of file kern_umtx.c.

References do_set_ceiling().

Here is the call graph for this function:

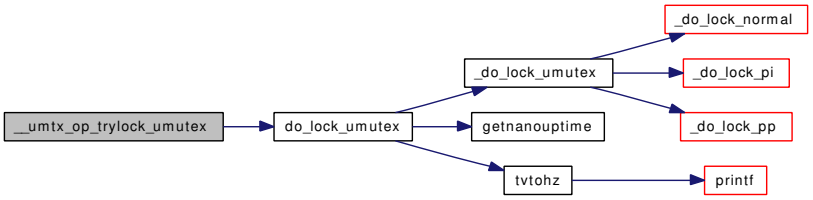


9.68.3.8 static int __umtx_op_trylock_umutex (struct thread * *td*, struct _umtx_op_args * *uap*) [static]

Definition at line 2466 of file kern_umtx.c.

References do_lock_umutex().

Here is the call graph for this function:

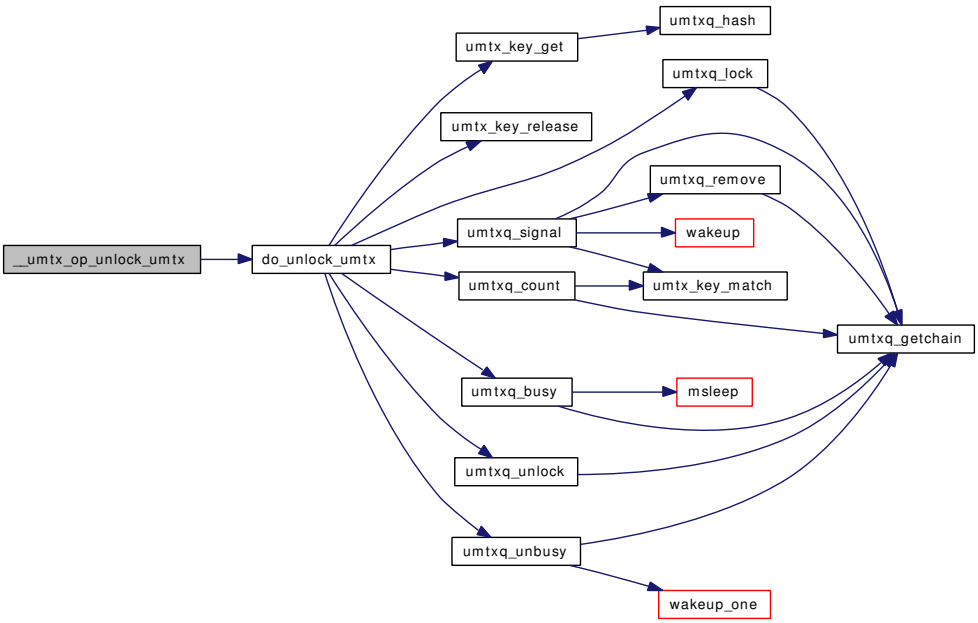


9.68.3.9 static int __umtx_op_unlock_umtx (struct thread * *td*, struct _umtx_op_args * *uap*) [static]

Definition at line 2411 of file kern_umtx.c.

References do_unlock_umtx().

Here is the call graph for this function:

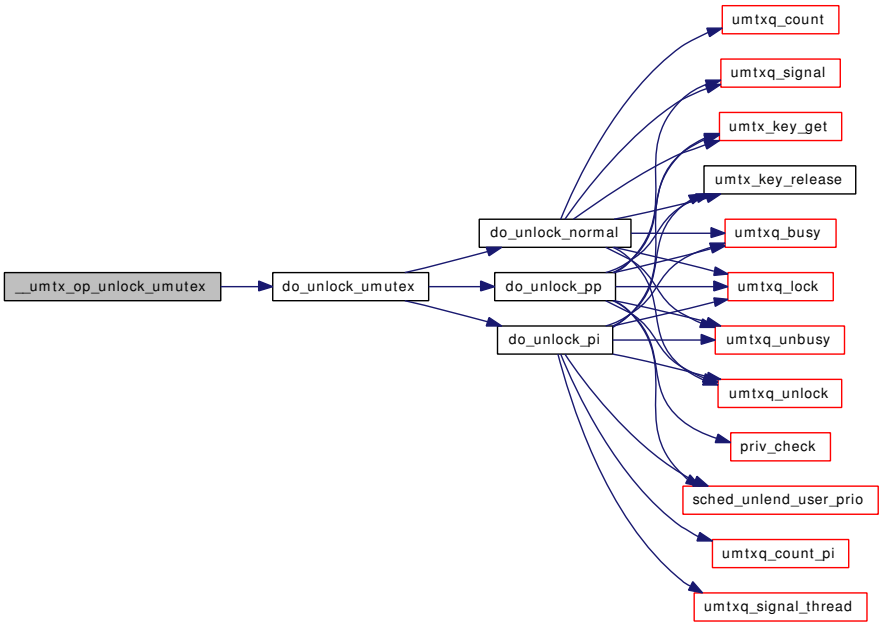


9.68.3.10 static int __umtx_op_unlock_umutex (struct thread * *td*, struct _umtx_op_args * *uap*) [static]

Definition at line 2472 of file kern_umtx.c.

References do_unlock_umutex().

Here is the call graph for this function:

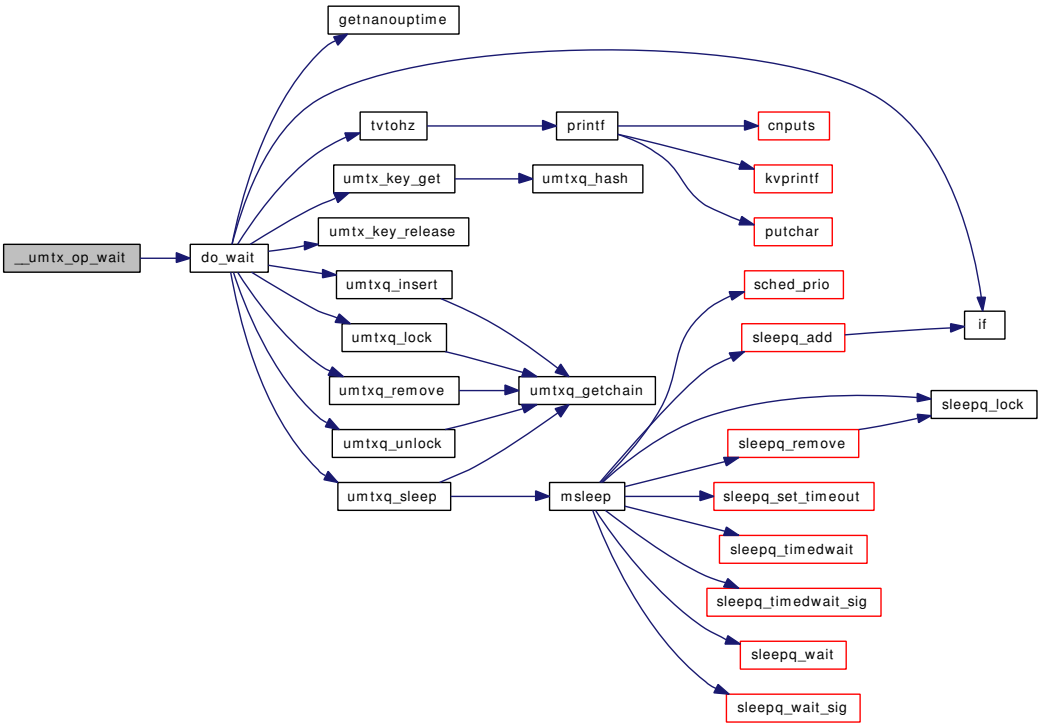


9.68.3.11 static int __umtx_op_wait (struct thread * *td*, struct _umtx_op_args * *uap*) [static]

Definition at line 2417 of file kern_umtx.c.

References do_wait().

Here is the call graph for this function:

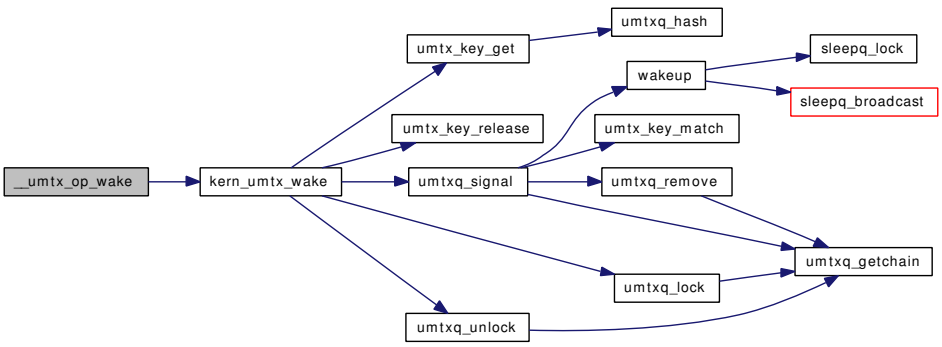


9.68.3.12 static int __umtx_op_wake (struct thread * *td*, struct _umtx_op_args * *uap*) [static]

Definition at line 2437 of file kern_umtx.c.

References kern_umtx_wake().

Here is the call graph for this function:



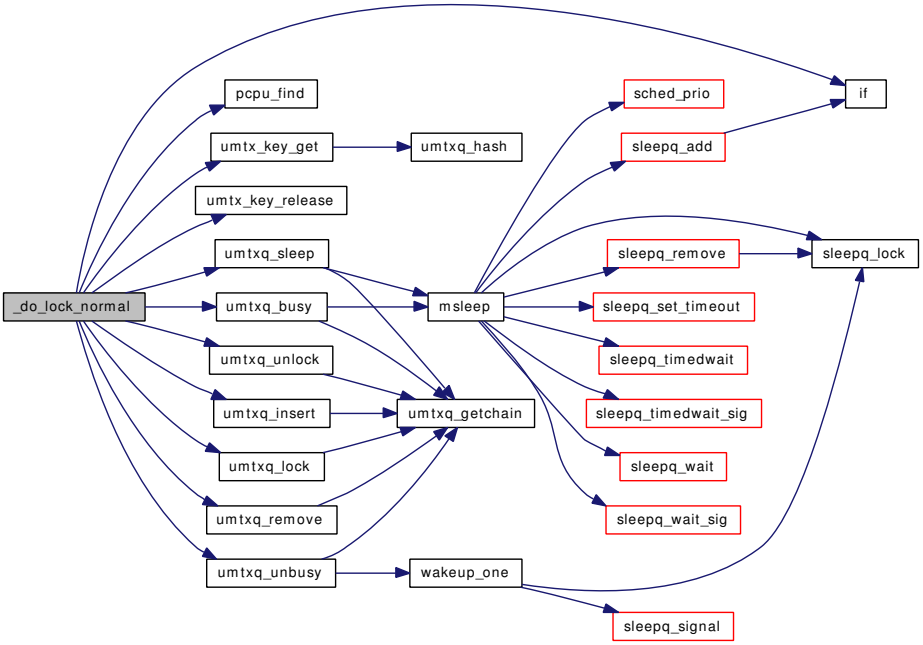
9.68.3.13 static int _do_lock_normal (struct thread * *td*, struct umutex * *m*, uint32_t *flags*, int *timo*, int *try*) [static]

Definition at line 1020 of file kern_umtx.c.

References GET_SHARE, if(), pcpu_find(), smp_cpus, TYPE_NORMAL_UMUTEX, umtx_dflt_spins, umtx_key_get(), umtx_key_release(), umtx_max_spins, umtxq_busy(), umtxq_insert(), umtxq_lock(), umtxq_remove(), umtxq_sleep(), umtxq_unbusy(), and umtxq_unlock().

Referenced by _do_lock_umutex().

Here is the call graph for this function:



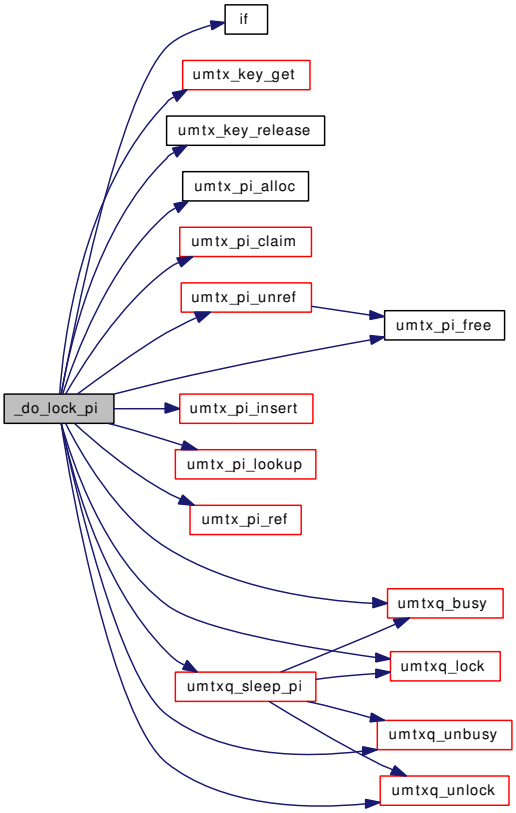
9.68.3.14 static int _do_lock_pi (struct thread * *td*, struct umutex * *m*, uint32_t *flags*, int *timo*, int *try*) [static]

Definition at line 1631 of file kern_umtx.c.

References GET_SHARE, if(), TYPE_PI_UMUTEX, umtx_key_get(), umtx_key_release(), umtx_pi_alloc(), umtx_pi_claim(), umtx_pi_free(), umtx_pi_insert(), umtx_pi_lookup(), umtx_pi_ref(), umtx_pi_unref(), umtxq_busy(), umtxq_lock(), umtxq_sleep_pi(), umtxq_unbusy(), and umtxq_unlock().

Referenced by _do_lock_umutex().

Here is the call graph for this function:



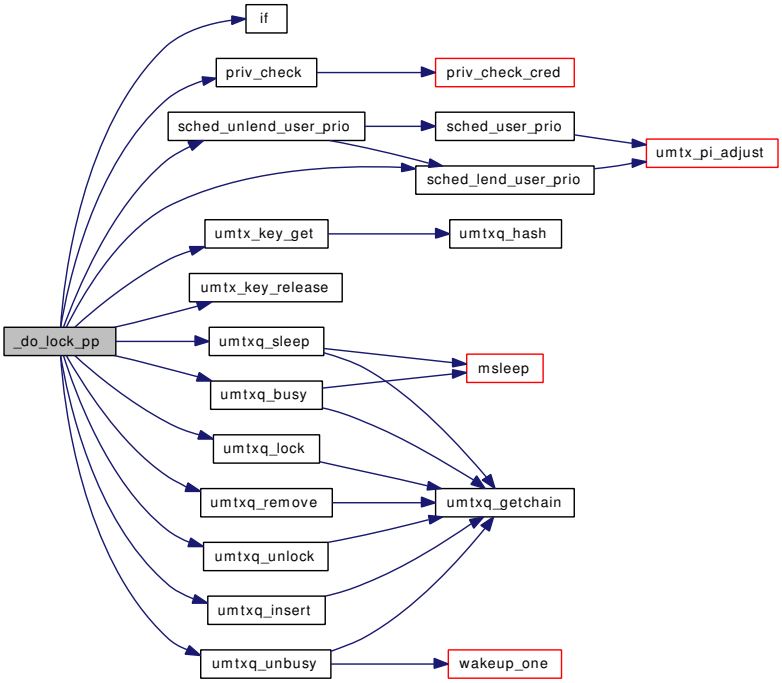
9.68.3.15 static int _do_lock_pp (struct thread * *td*, struct umutex * *m*, uint32_t *flags*, int *timo*, int *try*) [static]

Definition at line 1867 of file kern_umtx.c.

References GET_SHARE, if(), priv_check(), sched_lend_user_prio(), sched_lock, sched_unlend_user_prio(), TYPE_PP_UMUTEX, umtx_key_get(), umtx_key_release(), umtxq_busy(), umtxq_insert(), umtxq_lock(), umtxq_remove(), umtxq_sleep(), umtxq_unbusy(), umtxq_unlock(), and UPRI.

Referenced by _do_lock_umutex().

Here is the call graph for this function:



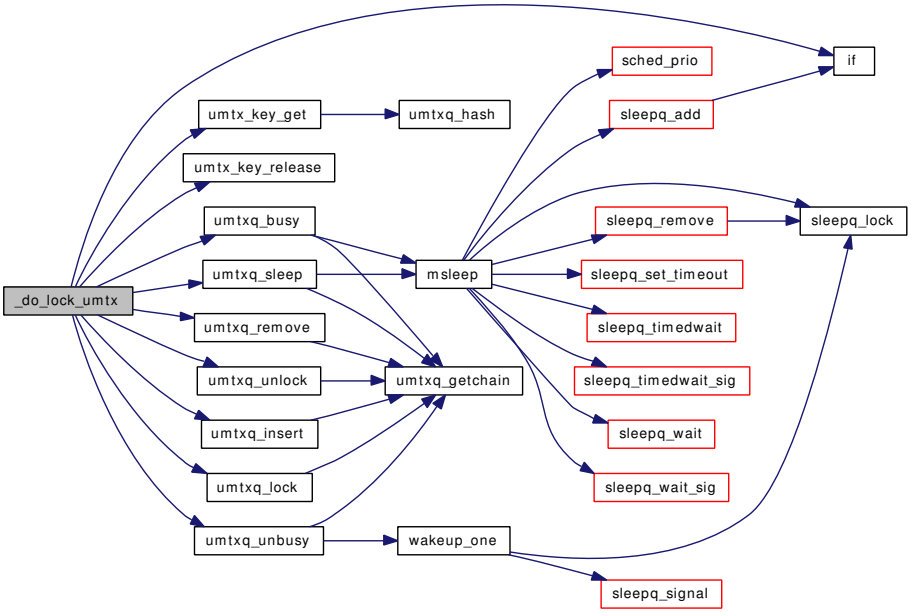
9.68.3.16 static int _do_lock_umtx (struct thread * *td*, struct umtx * *umtx*, u_long *id*, int *timo*) [static]

Definition at line 537 of file kern_umtx.c.

References AUTO_SHARE, if(), TYPE_SIMPLE_LOCK, umtx_key_get(), umtx_key_release(), umtxq_busy(), umtxq_insert(), umtxq_lock(), umtxq_remove(), umtxq_sleep(), umtxq_unbusy(), and umtxq_unlock().

Referenced by _umtx_lock(), and do_lock_umtx().

Here is the call graph for this function:



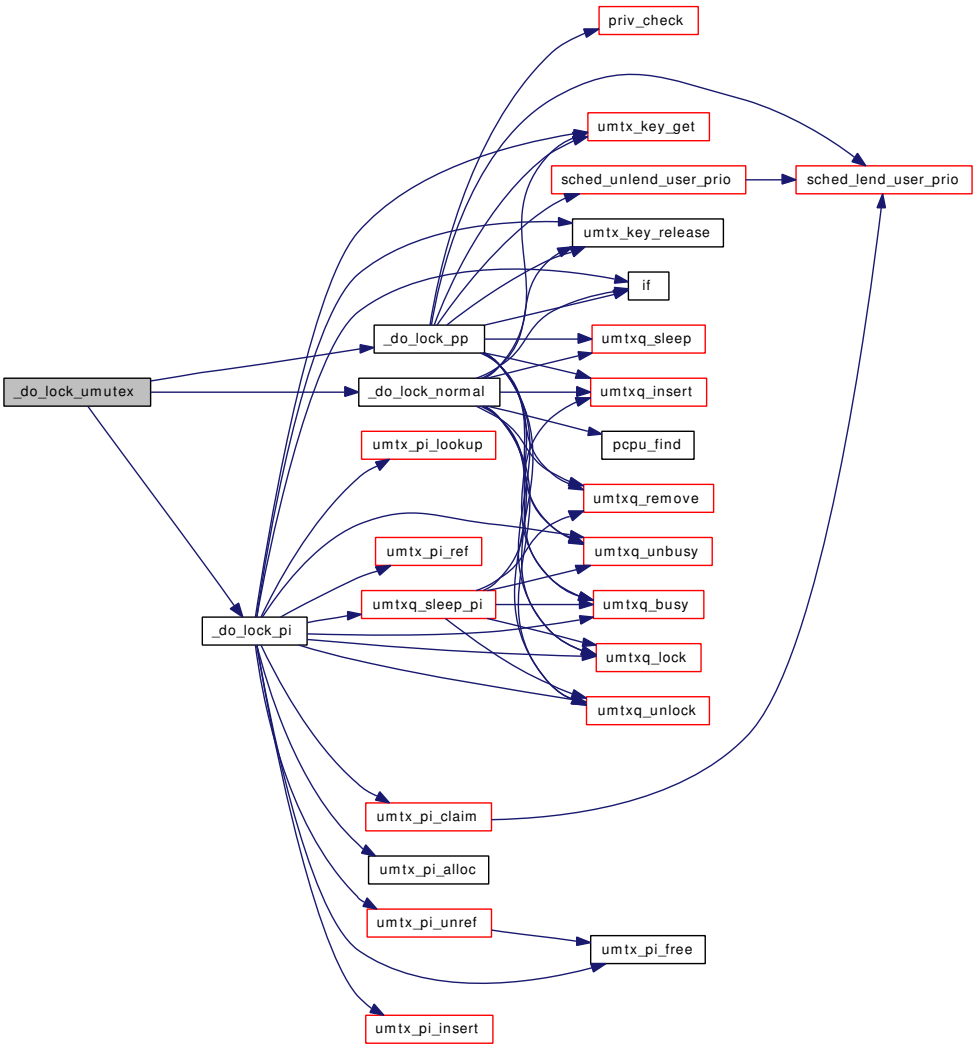
9.68.3.17 static int _do_lock_umutex (struct thread * *td*, struct umutex * *m*, int *flags*, int *timo*, int *try*) [static]

Definition at line 2152 of file kern_umtx.c.

References _do_lock_normal(), _do_lock_pi(), and _do_lock_pp().

Referenced by do_lock_umutex().

Here is the call graph for this function:

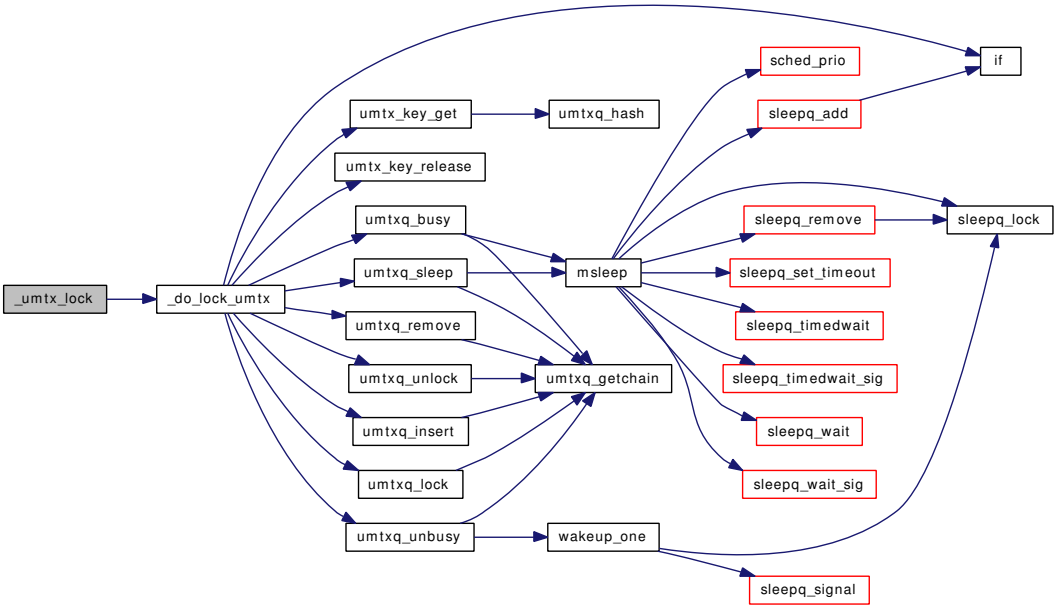


9.68.3.18 int _umtx_lock (struct thread * *td*, struct _umtx_lock_args * *uap*)

Definition at line 2375 of file kern_umtx.c.

References _do_lock_umtx().

Here is the call graph for this function:



9.68.3.19 int _umtx_op (struct thread * *td*, struct _umtx_op_args * *uap*)

Definition at line 2535 of file kern_umtx.c.

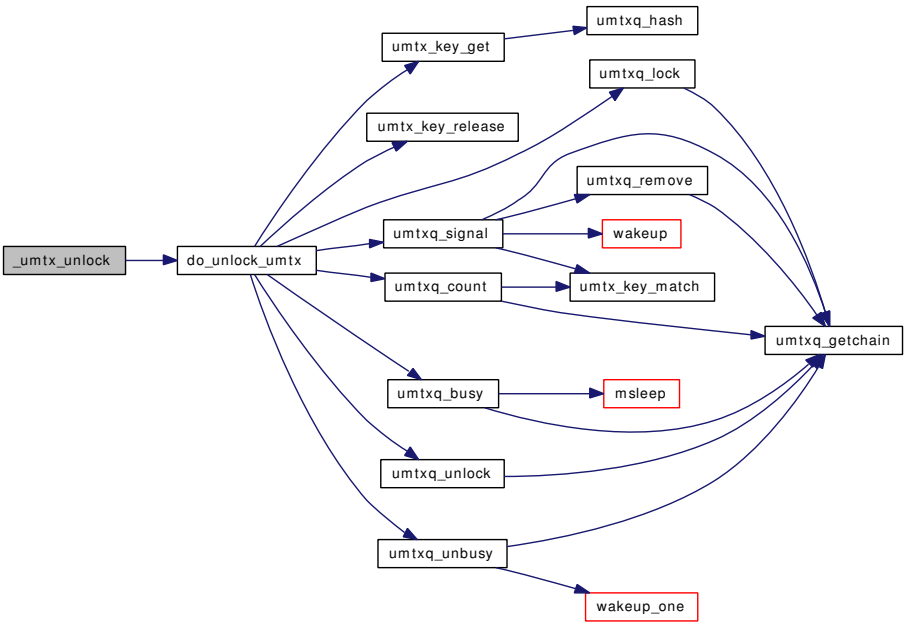
References op_table.

9.68.3.20 int _umtx_unlock (struct thread * *td*, struct _umtx_unlock_args * *uap*)

Definition at line 2382 of file kern_umtx.c.

References do_unlock_umtx().

Here is the call graph for this function:



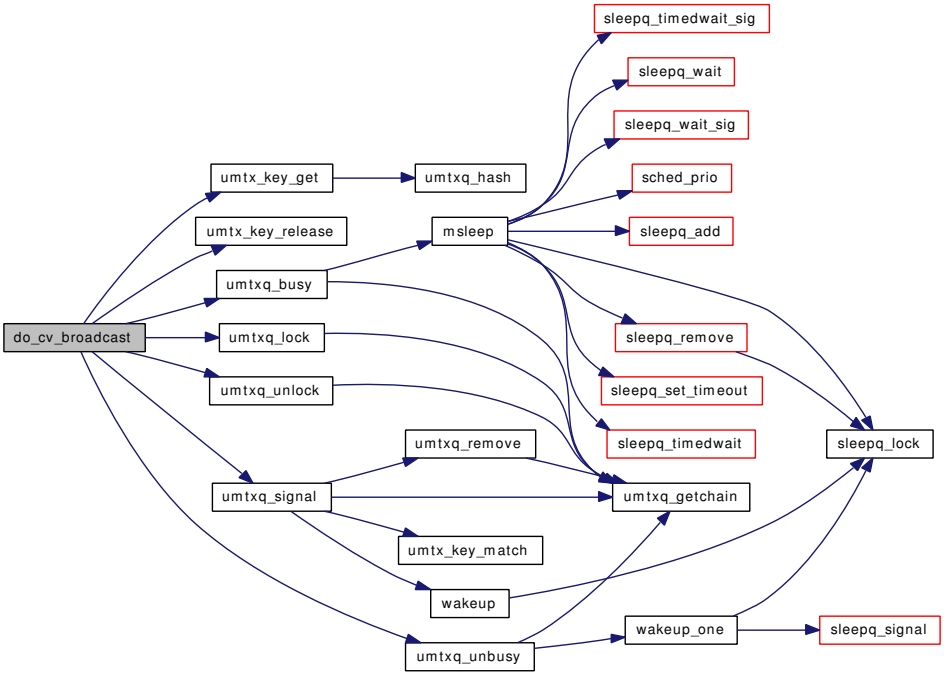
9.68.3.21 static int do_cv_broadcast (struct thread * *td*, struct ucond * *cv*) [static]

Definition at line 2349 of file kern_umtx.c.

References GET_SHARE, TYPE_CV, umtx_key_get(), umtx_key_release(), umtxq_busy(), umtxq_lock(), umtxq_signal(), umtxq_unbusy(), and umtxq_unlock().

Referenced by __umtx_op_cv_broadcast().

Here is the call graph for this function:



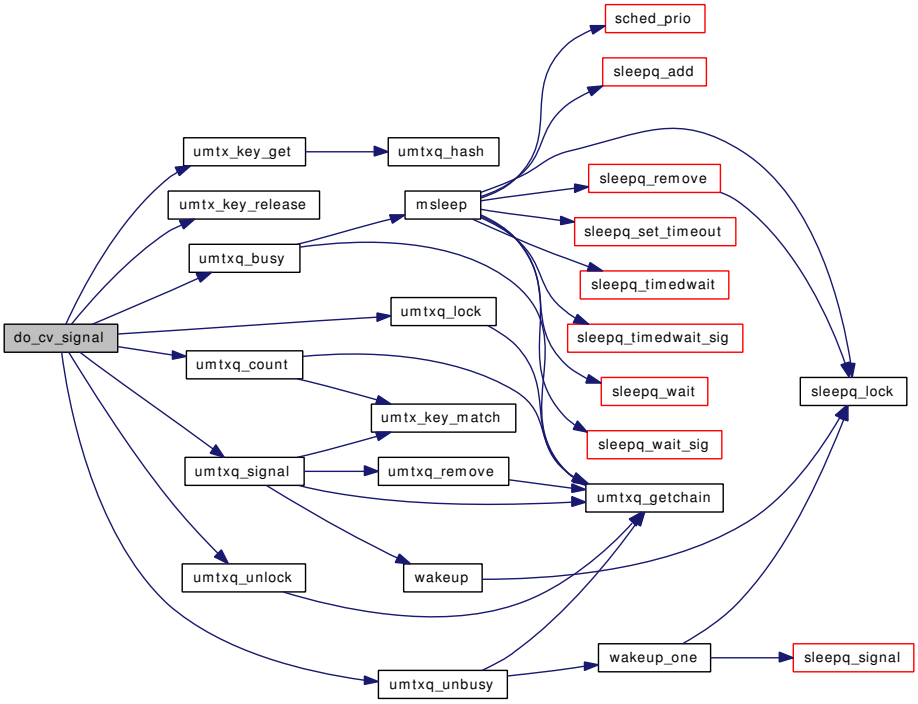
9.68.3.22 static int do_cv_signal (struct thread * *td*, struct ucond * *cv*) [static]

Definition at line 2323 of file kern_umtx.c.

References GET_SHARE, TYPE_CV, umtx_key_get(), umtx_key_release(), umtxq_busy(), umtxq_count(), umtxq_lock(), umtxq_signal(), umtxq_unbusy(), and umtxq_unlock().

Referenced by __umtx_op_cv_signal().

Here is the call graph for this function:



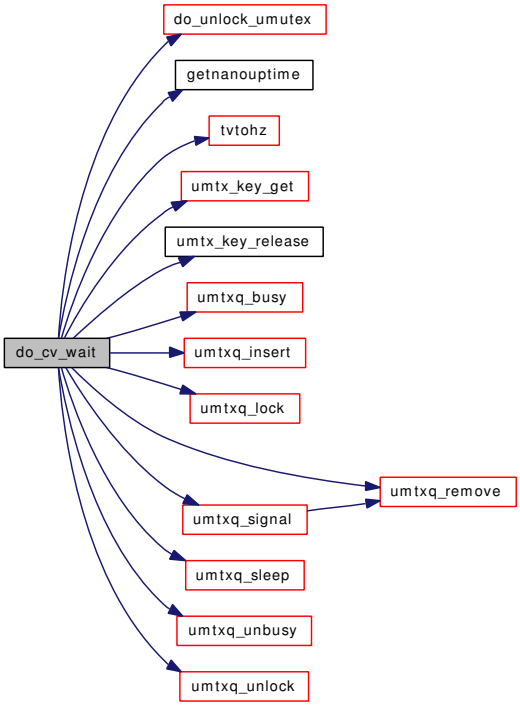
9.68.3.23 static int do_cv_wait (struct thread * *td*, struct ucond * *cv*, struct umutex * *m*, struct timespec * *timeout*, u_long *wflags*) [static]

Definition at line 2236 of file kern_umtx.c.

References do_unlock_umutex(), GET_SHARE, getnanouptime(), tvtohz(), TYPE_CV, umtx_key_get(), umtx_key_release(), umtxq_busy(), umtxq_insert(), umtxq_lock(), umtxq_remove(), umtxq_signal(), umtxq_sleep(), umtxq_unbusy(), umtxq_unlock(), and UQF_UMTXQ.

Referenced by __umtx_op_cv_wait().

Here is the call graph for this function:



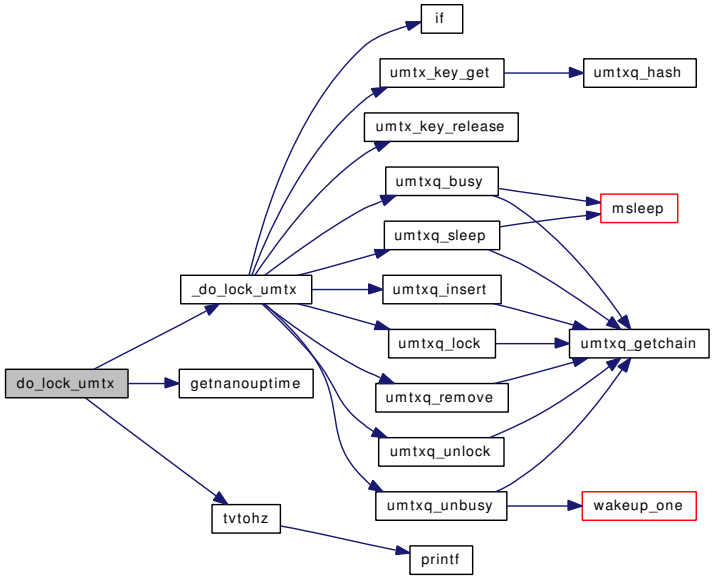
9.68.3.24 static int do_lock_umtx (struct thread * *td*, struct umtx * *umtx*, u_long *id*, struct timespec * *timeout*) [static]

Definition at line 634 of file kern_umtx.c.

References _do_lock_umtx(), getnanouptime(), and tvtohz().

Referenced by __umtx_op_lock_umtx().

Here is the call graph for this function:



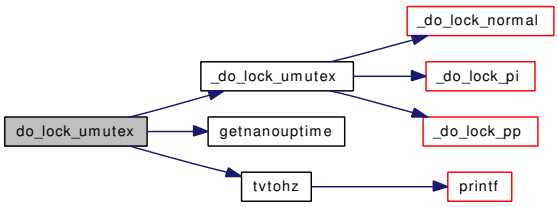
9.68.3.25 static int do_lock_umutex (struct thread * *td*, struct umutex * *m*, struct timespec * *timeout*, int *try*) [static]

Definition at line 2170 of file kern_umtx.c.

References _do_lock_umutex(), getnanouptime(), and tvtohz().

Referenced by __umtx_op_lock_umutex(), and __umtx_op_trylock_umutex().

Here is the call graph for this function:



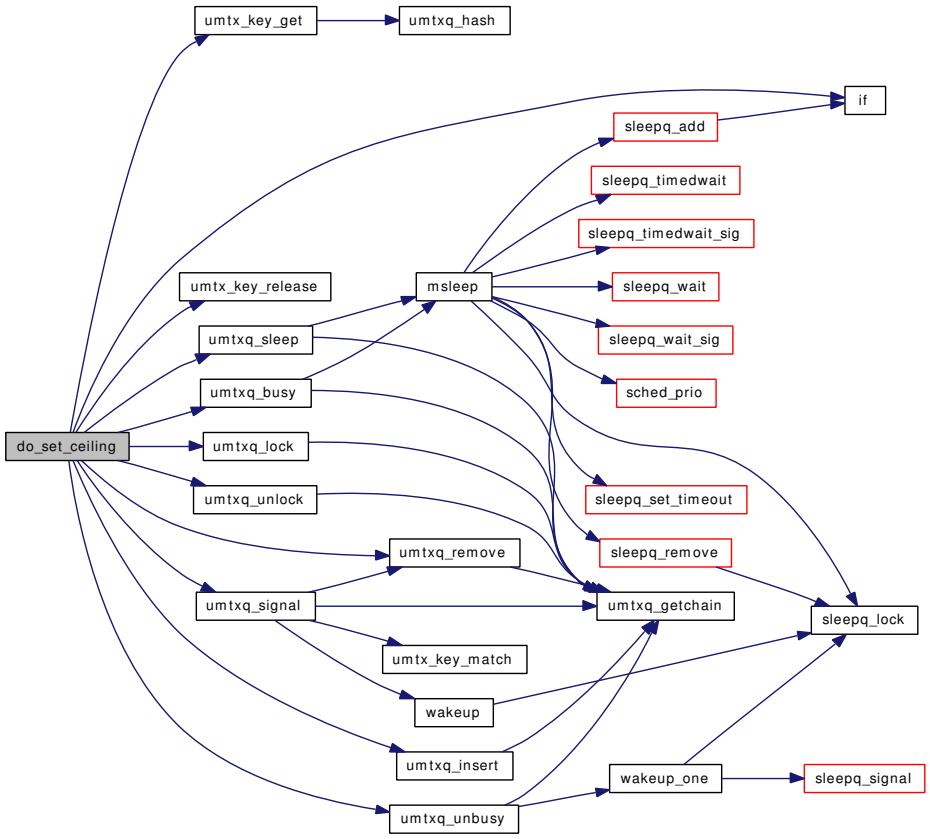
9.68.3.26 static int do_set_ceiling (struct thread * *td*, struct umutex * *m*, uint32_t *ceiling*, uint32_t * *old_ceiling*) [static]

Definition at line 2072 of file kern_umtx.c.

References GET_SHARE, if(), TYPE_PP_UMUTEX, umtx_key_get(), umtx_key_release(), umtxq_busy(), umtxq_insert(), umtxq_lock(), umtxq_remove(), umtxq_signal(), umtxq_sleep(), umtxq_unbusy(), and umtxq_unlock().

Referenced by __umtx_op_set_ceiling().

Here is the call graph for this function:



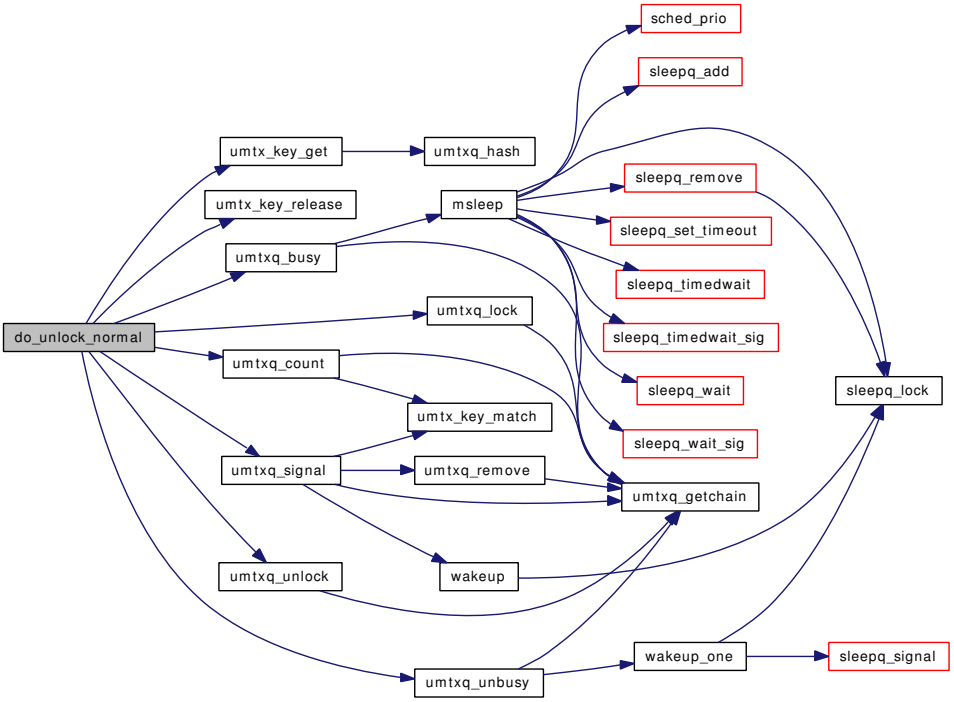
9.68.3.27 static int do_unlock_normal (struct thread * *td*, struct umutex * *m*, uint32_t *flags*) [static]

Definition at line 1188 of file kern_umtx.c.

References GET_SHARE, TYPE_NORMAL_UMUTEX, umtx_key_get(), umtx_key_release(), umtxq_busy(), umtxq_count(), umtxq_lock(), umtxq_signal(), umtxq_unbusy(), and umtxq_unlock().

Referenced by do_unlock_umutex().

Here is the call graph for this function:



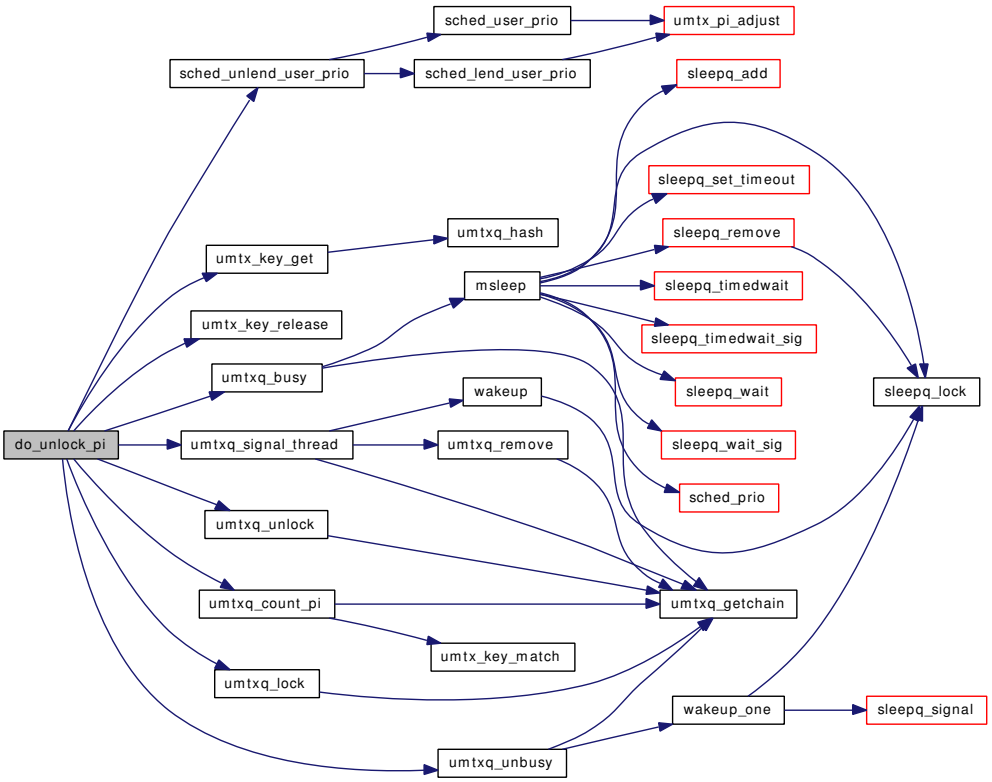
9.68.3.28 static int do_unlock_pi (struct thread * *td*, struct umutex * *m*, uint32_t *flags*) [static]

Definition at line 1777 of file kern_umtx.c.

References GET_SHARE, umtx_pi::pi_owner, sched_lock, sched_unlend_user_prio(), TYPE_PI_UMUTEX, umtx_key_get(), umtx_key_release(), umtxq_busy(), umtxq_count_pi(), umtxq_lock(), umtxq_signal_thread(), umtxq_unbusy(), umtxq_unlock(), and UPRI.

Referenced by do_unlock_umutex().

Here is the call graph for this function:



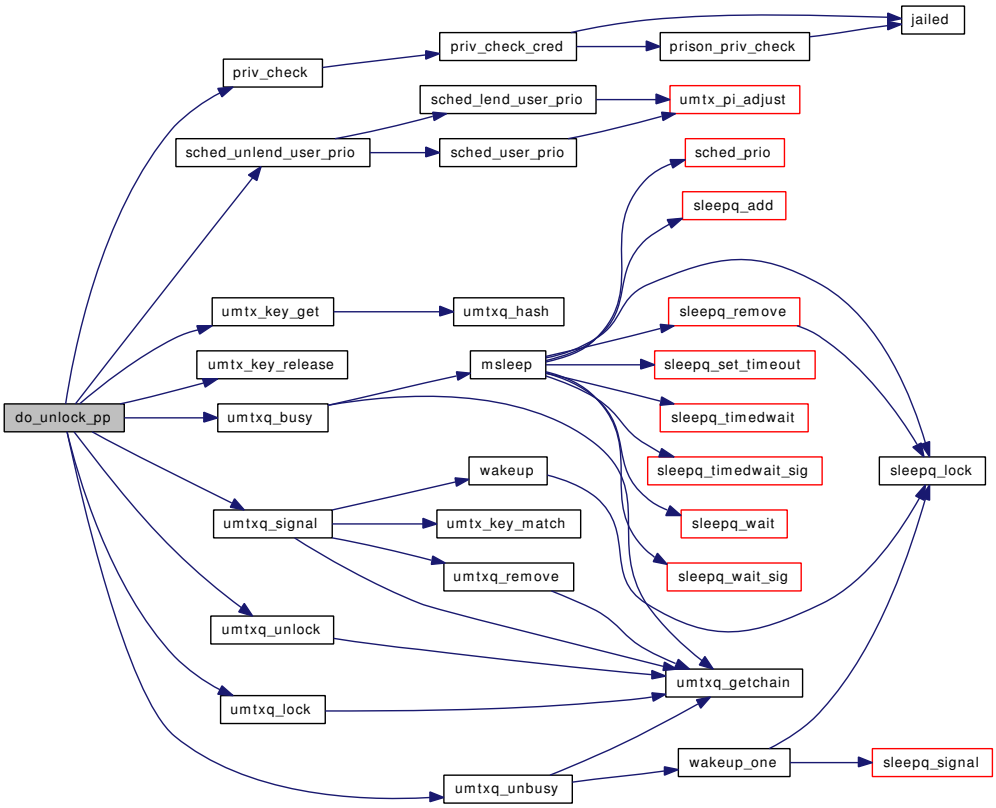
9.68.3.29 static int do_unlock_pp (struct thread * *td*, struct umutex * *m*, uint32_t *flags*) [static]

Definition at line 1991 of file kern_umtx.c.

References GET_SHARE, priv_check(), sched_lock, sched_unlend_user_prio(), TYPE_PP_UMUTEX, umtx_key_get(), umtx_key_release(), umtxq_busy(), umtxq_lock(), umtxq_signal(), umtxq_unbusy(), umtxq_unlock(), and UPRI.

Referenced by do_unlock_umutex().

Here is the call graph for this function:



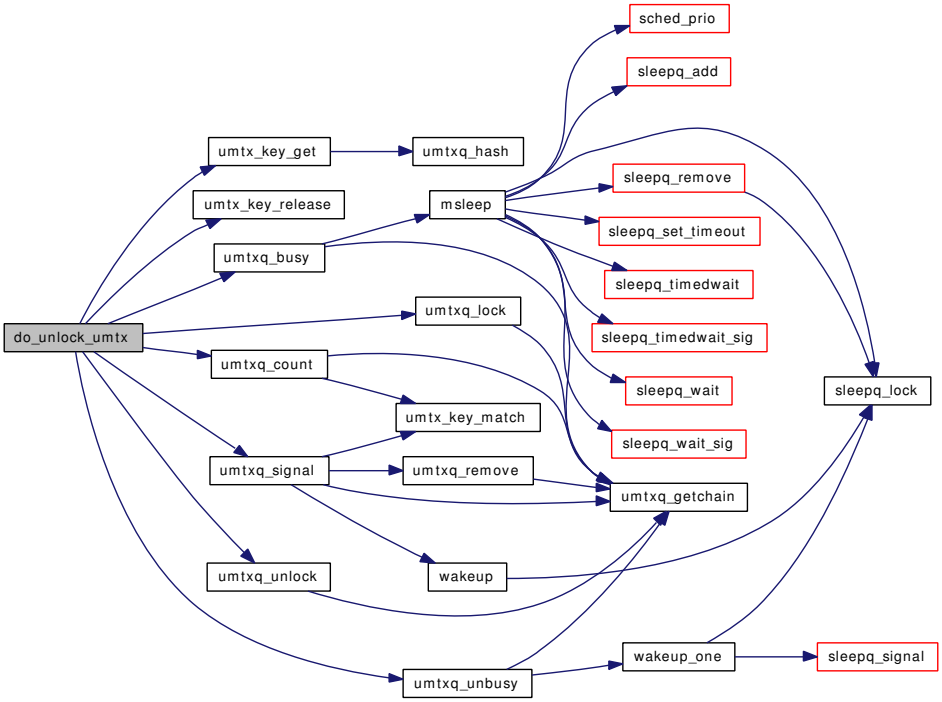
9.68.3.30 static int do_unlock_umtx (struct thread * *td*, struct umtx * *umtx*, u_long *id*) [static]

Definition at line 674 of file kern_umtx.c.

References AUTO_SHARE, TYPE_SIMPLE_LOCK, umtx_key_get(), umtx_key_release(), umtxq_busy(), umtxq_count(), umtxq_lock(), umtxq_signal(), umtxq_unbusy(), and umtxq_unlock().

Referenced by __umtx_op_unlock_umtx(), and _umtx_unlock().

Here is the call graph for this function:



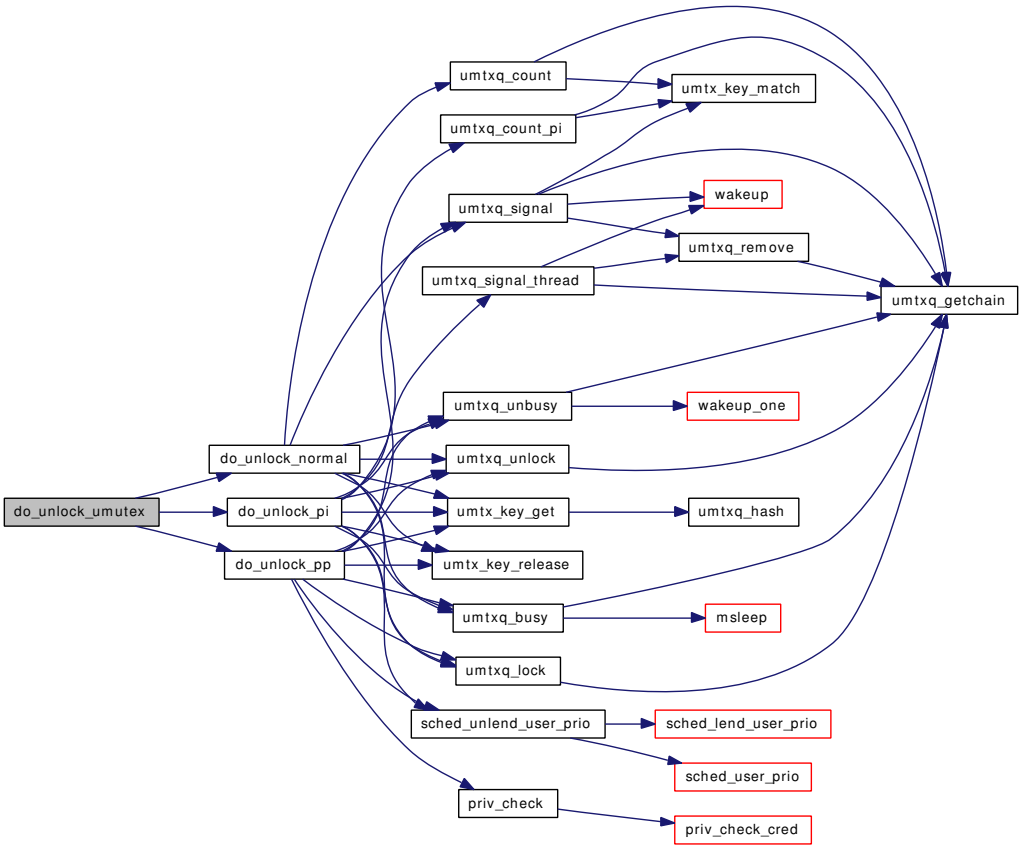
9.68.3.31 static int do_unlock_umutex (struct thread * *td*, struct umutex * *m*) [static]

Definition at line 2215 of file kern_umtx.c.

References do_unlock_normal(), do_unlock_pi(), and do_unlock_pp().

Referenced by __umtx_op_unlock_umutex(), and do_cv_wait().

Here is the call graph for this function:



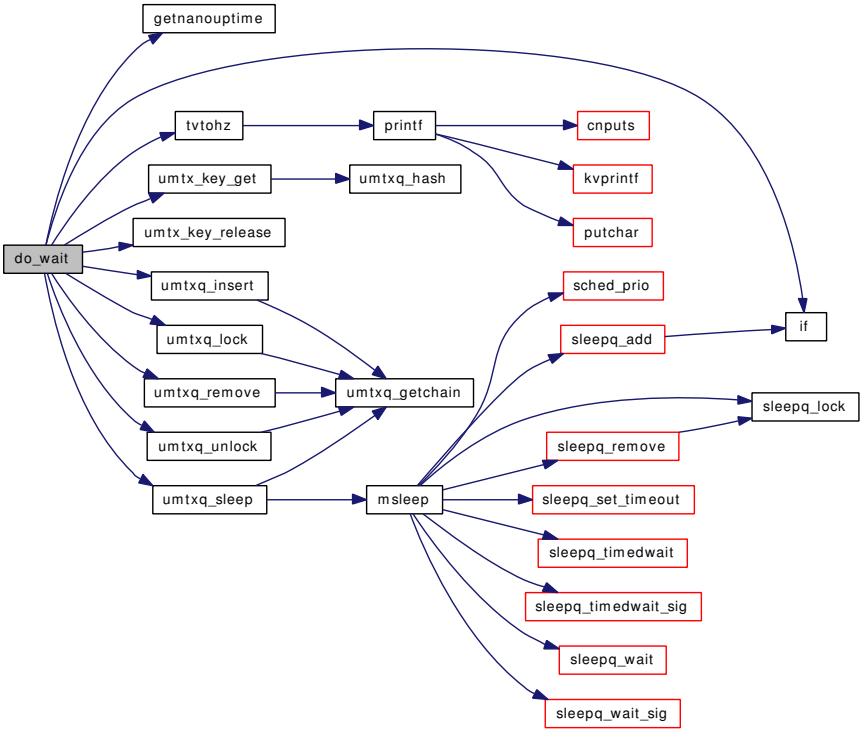
9.68.3.32 static int do_wait (struct thread * *td*, void * *addr*, u_long *id*, struct timespec * *timeout*, int *compat32*) [static]

Definition at line 935 of file kern_umtx.c.

References AUTO_SHARE, getnanouptime(), if(), tvtohz(), TYPE_SIMPLE_WAIT, umtx_key_get(), umtx_key_release(), umtxq_insert(), umtxq_lock(), umtxq_remove(), umtxq_sleep(), umtxq_unlock(), and UQF_UMTXQ.

Referenced by __umtx_op_wait().

Here is the call graph for this function:



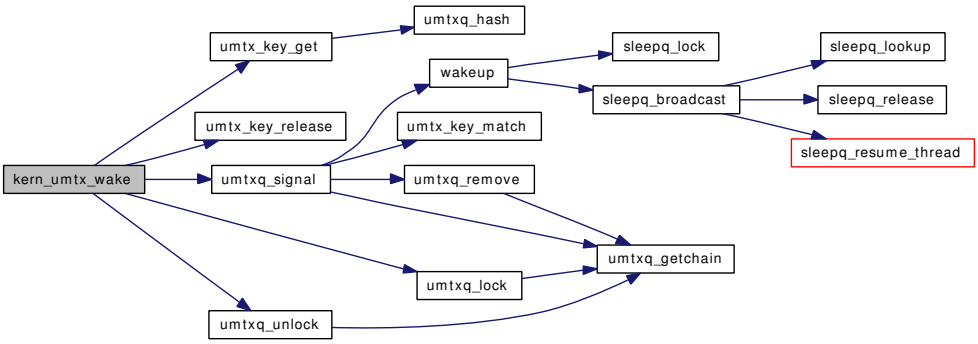
9.68.3.33 int kern_umtx_wake (struct thread * *td*, void * *uaddr*, int *n_wake*)

Definition at line 1001 of file kern_umtx.c.

References AUTO_SHARE, ret, TYPE_SIMPLE_WAIT, umtx_key_get(), umtx_key_release(), umtxq_lock(), umtxq_signal(), and umtxq_unlock().

Referenced by __umtx_op_wake(), and thr_exit().

Here is the call graph for this function:



9.68.3.34 static MALLOC_DEFINE (M_UMTX, "umtx", "UMTX queue memory") [static]

9.68.3.35 SYSCTL_DECL (_kern_threads)

9.68.3.36 SYSCTL_INT (_kern_threads, OID_AUTO, umtx_max_spins, CTLFLAG_RW, & *umtx_max_spins*, 0, "max umtx spin count")

9.68.3.37 SYSCTL_INT (_kern_threads, OID_AUTO, umtx_dflt_spins, CTLFLAG_RW, & *umtx_dflt_spins*, 0, "default umtx spin count")

9.68.3.38 SYSCTL_INT (_debug_umtx, OID_AUTO, umtx_pi_allocated, CTLFLAG_RD, & *umtx_pi_allocated*, 0, "Allocated umtx_pi")

9.68.3.39 SYSCTL_NODE (_debug, OID_AUTO, umtx, CTLFLAG_RW, 0, "umtx debug")

9.68.3.40 SYSINIT (umtx, SI_SUB_EVENTHANDLER+ *1*, SI_ORDER_MIDDLE, umtxq_sysinit, NULL)

9.68.3.41 TAILQ_HEAD (umtxq_head, umtx_q)

9.68.3.42 static void umtx_exec_hook (void *arg *__unused*, struct proc *p *__unused*, struct image_params *imgp *__unused*) [static]

Definition at line 2725 of file kern_umtx.c.

References umtx_thread_cleanup().

Referenced by umtxq_sysinit().

Here is the call graph for this function:

9.68.3.43 static int umtx_key_get (void * *addr*, int *type*, int *share*, struct umtx_key * *key*) [static]

Definition at line 481 of file kern_umtx.c.

References AUTO_SHARE, umtx_key::info, umtx_key::private, PROCESS_SHARE, umtx_key::shared, td, THREAD_SHARE, umtx_key::type, and umtxq_hash().

Referenced by _do_lock_normal(), _do_lock_pi(), _do_lock_pp(), _do_lock_umtx(), do_cv_broadcast(), do_cv_signal(), do_cv_wait(), do_set_ceiling(), do_unlock_normal(), do_unlock_pi(), do_unlock_pp(), do_unlock_umtx(), do_wait(), and kern_umtx_wake().

Here is the call graph for this function:

9.68.3.44 static int umtx_key_match (const struct umtx_key * *k1*, const struct umtx_key * *k2*) [inline, static]

Definition at line 272 of file kern_umtx.c.

References umtx_key::both, umtx_key::info, and umtx_key::type.

Referenced by umtx_pi_lookup(), umtxq_count(), umtxq_count_pi(), and umtxq_signal().

9.68.3.45 static void umtx_key_release (struct umtx_key * *key*) [inline, static]

Definition at line 527 of file kern_umtx.c.

References umtx_key::info, and umtx_key::shared.

Referenced by _do_lock_normal(), _do_lock_pi(), _do_lock_pp(), _do_lock_umtx(), do_cv_broadcast(), do_cv_signal(), do_cv_wait(), do_set_ceiling(), do_unlock_normal(), do_unlock_pi(), do_unlock_pp(), do_unlock_umtx(), do_wait(), and kern_umtx_wake().

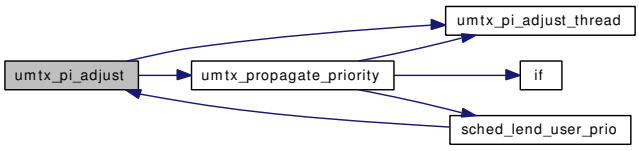
9.68.3.46 void umtx_pi_adjust (struct thread * *td*, u_char *oldpri*)

Definition at line 1438 of file kern_umtx.c.

References sched_lock, umtx_pi_adjust_thread(), umtx_propagate_priority(), and UPRI.

Referenced by sched_lend_user_prio(), and sched_user_prio().

Here is the call graph for this function:



9.68.3.47 static int umtx_pi_adjust_thread (struct umtx_pi * *pi*, struct thread * *td*) [static]

Definition at line 1268 of file kern_umtx.c.

References sched_lock, and UPRI.

Referenced by umtx_pi_adjust(), and umtx_propagate_priority().

9.68.3.48 static struct umtx_pi * umtx_pi_alloc (int) [inline, static]

Definition at line 1246 of file kern_umtx.c.

References umtx_pi_allocated, and umtx_pi_zone.

Referenced by _do_lock_pi().

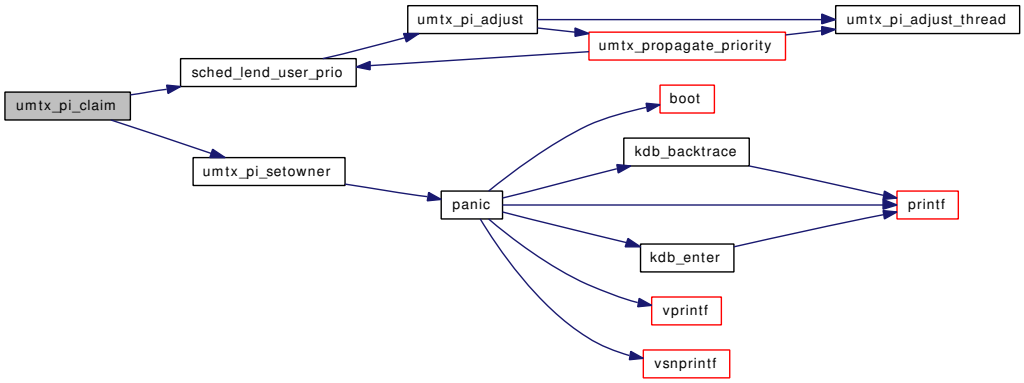
9.68.3.49 static int umtx_pi_claim (struct umtx_pi * *pi*, struct thread * *owner*) [static]

Definition at line 1402 of file kern_umtx.c.

References umtx_pi::pi_owner, sched_lend_user_prio(), sched_lock, umtx_pi_setowner(), and UPRI.

Referenced by _do_lock_pi().

Here is the call graph for this function:



9.68.3.50 static void umtx_pi_free (struct umtx_pi * *pi*) [inline, static]

Definition at line 1257 of file kern_umtx.c.

References umtx_pi_allocated, and umtx_pi_zone.

Referenced by _do_lock_pi(), and umtx_pi_unref().

9.68.3.51 static void umtx_pi_insert (struct umtx_pi * *pi*) [inline, static]

Definition at line 1618 of file kern_umtx.c.

References uc, umtxq_getchain(), and UMTXQ_LOCKED_ASSERT.

Referenced by _do_lock_pi().

Here is the call graph for this function:

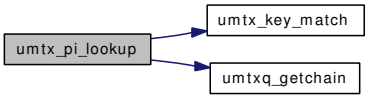
9.68.3.52 static struct umtx_pi* umtx_pi_lookup (struct umtx_key * *key*) [static]

Definition at line 1598 of file kern_umtx.c.

References uc, umtx_key_match(), umtxq_getchain(), and UMTXQ_LOCKED_ASSERT.

Referenced by _do_lock_pi().

Here is the call graph for this function:



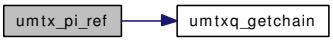
9.68.3.53 static void umtx_pi_ref (struct umtx_pi * *pi*) [static]

Definition at line 1555 of file kern_umtx.c.

References umtx_pi::pi_refcount, uc, umtxq_getchain(), and UMTXQ_LOCKED_ASSERT.

Referenced by _do_lock_pi().

Here is the call graph for this function:



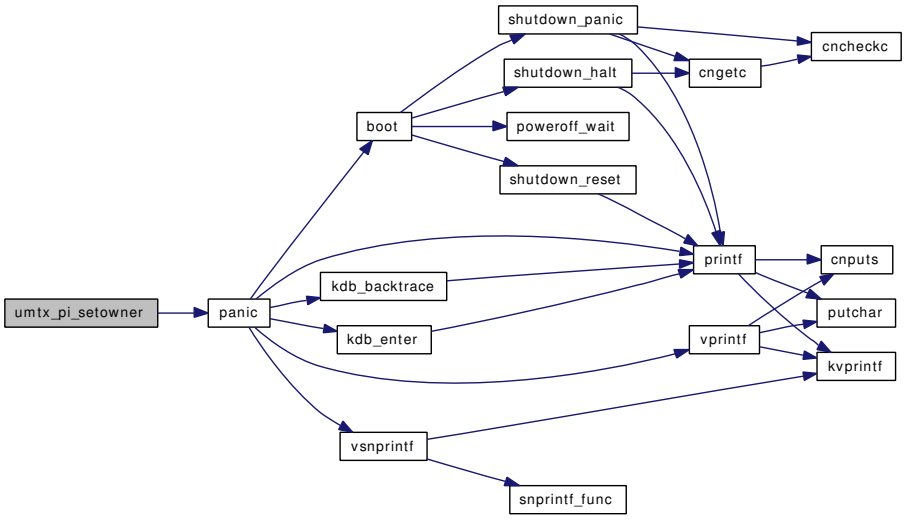
9.68.3.54 static void umtx_pi_setowner (struct umtx_pi * *pi*, struct thread * *owner*) [static]

Definition at line 1386 of file kern_umtx.c.

References panic(), umtx_pi::pi_owner, and sched_lock.

Referenced by umtx_pi_claim(), and umtxq_sleep_pi().

Here is the call graph for this function:



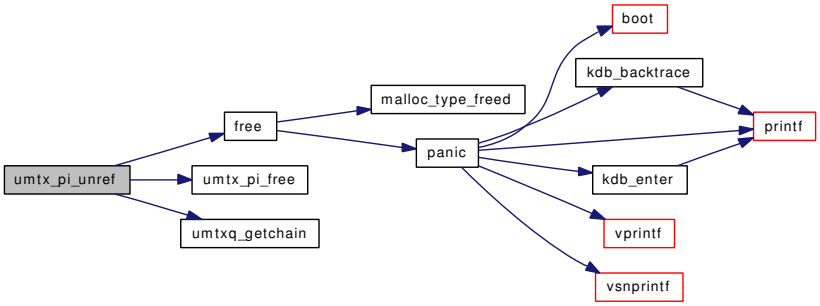
9.68.3.55 static void umtx_pi_unref (struct umtx_pi * *pi*) [static]

Definition at line 1569 of file kern_umtx.c.

References free(), umtx_pi::pi_owner, umtx_pi::pi_refcount, sched_lock, uc, umtx_pi_free(), umtxq_-getchain(), and UMTXQ_LOCKED_ASSERT.

Referenced by _do_lock_pi().

Here is the call graph for this function:



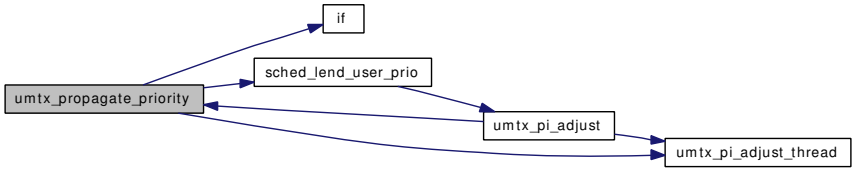
9.68.3.56 static void umtx_propagate_priority (struct thread * *td*) [static]

Definition at line 1313 of file kern_umtx.c.

References if(), umtx_pi::pi_owner, sched_lend_user_prio(), sched_lock, umtx_pi_adjust_thread(), and UPRI.

Referenced by umtx_pi_adjust(), and umtxq_sleep_pi().

Here is the call graph for this function:



9.68.3.57 void umtx_thread_alloc (struct thread * *td*)

Definition at line 2708 of file kern_umtx.c.

9.68.3.58 static void umtx_thread_cleanup (struct thread * *td*) [static]

Definition at line 2744 of file kern_umtx.c.

References umtx_pi::pi_owner, and sched_lock.

Referenced by umtx_exec_hook(), and umtx_thread_exit().

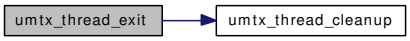
9.68.3.59 void umtx_thread_exit (struct thread * *td*)

Definition at line 2735 of file kern_umtx.c.

References umtx_thread_cleanup().

Referenced by thread_exit().

Here is the call graph for this function:



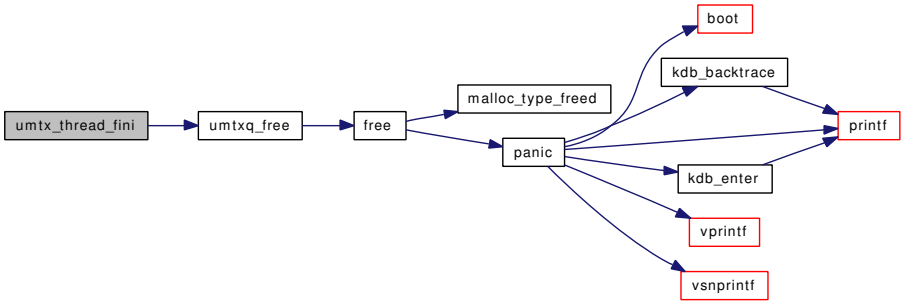
9.68.3.60 void umtx_thread_fini (struct thread * *td*)

Definition at line 2699 of file kern_umtx.c.

References umtxq_free().

Referenced by thread_fini().

Here is the call graph for this function:



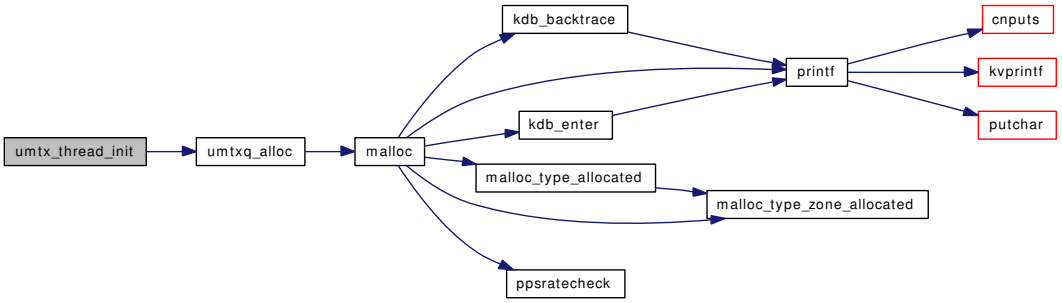
9.68.3.61 void umtx_thread_init (struct thread * *td*)

Definition at line 2692 of file kern_umtx.c.

References umtxq_alloc().

Referenced by thread_init().

Here is the call graph for this function:



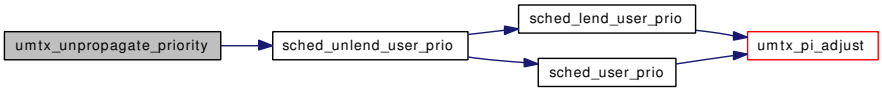
9.68.3.62 static void umtx_unpropagate_priority (struct umtx_pi * *pi*) [static]

Definition at line 1355 of file kern_umtx.c.

References umtx_pi::pi_owner, sched_lock, sched_unlend_user_prio(), and UPRI.

Referenced by umtxq_sleep_pi().

Here is the call graph for this function:



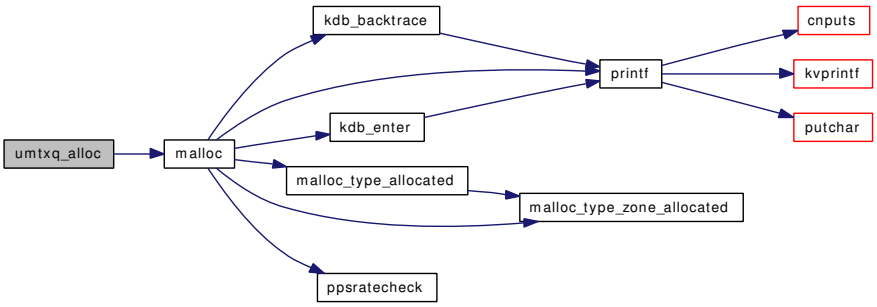
9.68.3.63 struct umtx_q* umtxq_alloc (void)

Definition at line 248 of file kern_umtx.c.

References malloc().

Referenced by umtx_thread_init().

Here is the call graph for this function:



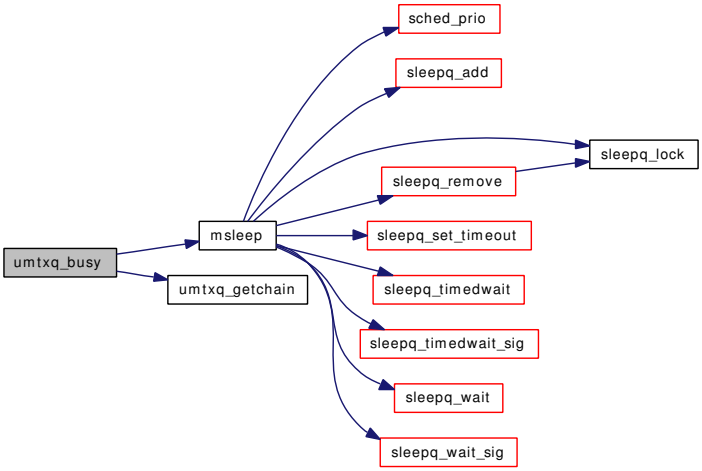
9.68.3.64 static void umtxq_busy (struct umtx_key * *key*) [inline, static]

Definition at line 290 of file kern_umtx.c.

References msleep(), uc, and umtxq_getchain().

Referenced by _do_lock_normal(), _do_lock_pi(), _do_lock_pp(), _do_lock_umtx(), do_cv_broadcast(), do_cv_signal(), do_cv_wait(), do_set_ceiling(), do_unlock_normal(), do_unlock_pi(), do_unlock_pp(), do_unlock_umtx(), and umtxq_sleep_pi().

Here is the call graph for this function:



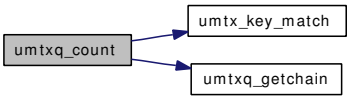
9.68.3.65 static int umtxq_count (struct umtx_key * *key*) [static]

Definition at line 378 of file kern_umtx.c.

References uc, umtx_key_match(), umtxq_getchain(), and UMTXQ_LOCKED_ASSERT.

Referenced by do_cv_signal(), do_unlock_normal(), and do_unlock_umtx().

Here is the call graph for this function:



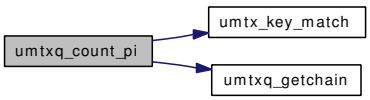
9.68.3.66 static int umtxq_count_pi (struct umtx_key * *key*, struct umtx_q ** *first*) [static]

Definition at line 400 of file kern_umtx.c.

References uc, umtx_key_match(), umtxq_getchain(), and UMTXQ_LOCKED_ASSERT.

Referenced by do_unlock_pi().

Here is the call graph for this function:



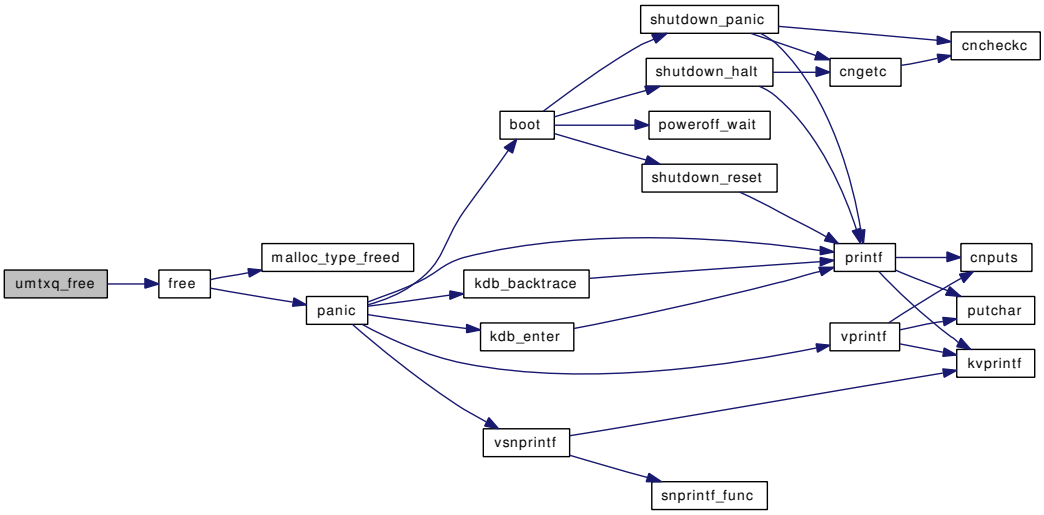
9.68.3.67 void umtxq_free (struct umtx_q * *uq*)

Definition at line 259 of file kern_umtx.c.

References free().

Referenced by umtx_thread_fini().

Here is the call graph for this function:



9.68.3.68 static struct umtxq_chain * umtxq_getchain (struct umtx_key * *key*) [inline, static]

Definition at line 280 of file kern_umtx.c.

References umtx_key::hash, and umtxq_chains.

Referenced by umtx_pi_insert(), umtx_pi_lookup(), umtx_pi_ref(), umtx_pi_unref(), umtxq_busy(), umtxq_count(), umtxq_count_pi(), umtxq_insert(), umtxq_lock(), umtxq_remove(), umtxq_signal(), umtxq_signal_thread(), umtxq_sleep(), umtxq_sleep_pi(), umtxq_unbusy(), and umtxq_unlock().

9.68.3.69 static void umtxq_hash (struct umtx_key * *key*) [inline, static]

Definition at line 265 of file kern_umtx.c.

References umtx_key::both, GOLDEN_RATIO_PRIME, umtx_key::hash, umtx_key::info, UMTX_-CHAINS, and UMTX_SHIFTS.

Referenced by umtx_key_get().

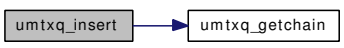
9.68.3.70 static void umtxq_insert (struct umtx_q * *uq*) [inline, static]

Definition at line 348 of file kern_umtx.c.

References uc, umtxq_getchain(), UMTXQ_LOCKED_ASSERT, and UQF_UMTXQ.

Referenced by _do_lock_normal(), _do_lock_pp(), _do_lock_umtx(), do_cv_wait(), do_set_ceiling(), do_-wait(), and umtxq_sleep_pi().

Here is the call graph for this function:



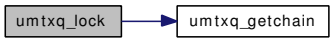
9.68.3.71 static void umtxq_lock (struct umtx_key * *key*) [inline, static]

Definition at line 324 of file kern_umtx.c.

References uc, and umtxq_getchain().

Referenced by _do_lock_normal(), _do_lock_pi(), _do_lock_pp(), _do_lock_umtx(), do_cv_broadcast(), do_cv_signal(), do_cv_wait(), do_set_ceiling(), do_unlock_normal(), do_unlock_pi(), do_unlock_pp(), do_unlock_umtx(), do_wait(), kern_umtx_wake(), and umtxq_sleep_pi().

Here is the call graph for this function:



9.68.3.72 static void umtxq_remove (struct umtx_q * *uq*) [inline, static]

Definition at line 362 of file kern_umtx.c.

References uc, umtxq_getchain(), UMTXQ_LOCKED_ASSERT, and UQF_UMTXQ.

Referenced by _do_lock_normal(), _do_lock_pp(), _do_lock_umtx(), do_cv_wait(), do_set_ceiling(), do_wait(), umtxq_signal(), umtxq_signal_thread(), and umtxq_sleep_pi().

Here is the call graph for this function:

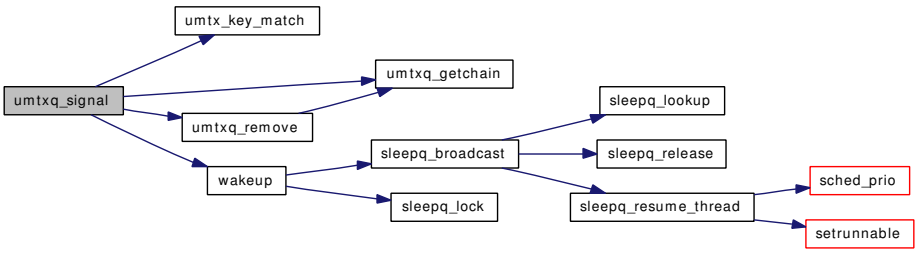
9.68.3.73 static int umtxq_signal (struct umtx_key * *key*, int *nr_wakeup*) [static]

Definition at line 423 of file kern_umtx.c.

References ret, uc, umtx_key_match(), umtxq_getchain(), UMTXQ_LOCKED_ASSERT, umtxq_remove(), and wakeup().

Referenced by do_cv_broadcast(), do_cv_signal(), do_cv_wait(), do_set_ceiling(), do_unlock_normal(), do_unlock_pp(), do_unlock_umtx(), and kern_umtx_wake().

Here is the call graph for this function:



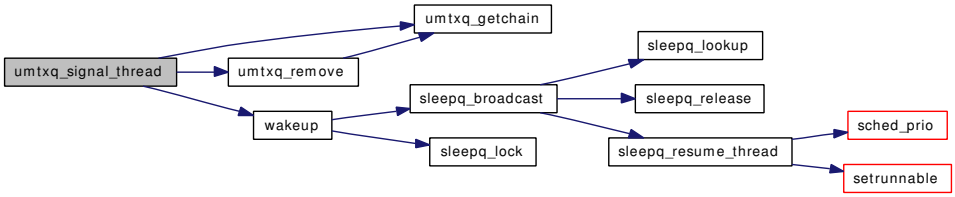
9.68.3.74 static void umtxq_signal_thread (struct umtx_q * *uq*) [inline, static]

Definition at line 447 of file kern_umtx.c.

References uc, umtxq_getchain(), UMTXQ_LOCKED_ASSERT, umtxq_remove(), and wakeup().

Referenced by do_unlock_pi().

Here is the call graph for this function:



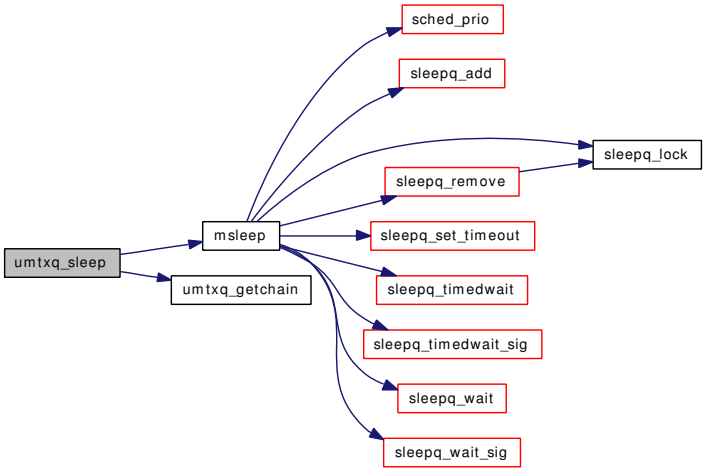
9.68.3.75 static int umtxq_sleep (struct umtx_q * *uq*, const char * *wmesg*, int *timo*) [inline, static]

Definition at line 462 of file kern_umtx.c.

References msleep(), uc, umtxq_getchain(), UMTXQ_LOCKED_ASSERT, and UQF_UMTXQ.

Referenced by _do_lock_normal(), _do_lock_pp(), _do_lock_umtx(), do_cv_wait(), do_set_ceiling(), and do_wait().

Here is the call graph for this function:



9.68.3.76 static int umtxq_sleep_pi (struct umtx_q * *uq*, struct umtx_pi * *pi*, uint32_t *owner*, const char * *wmesg*, int *timo*) [static]

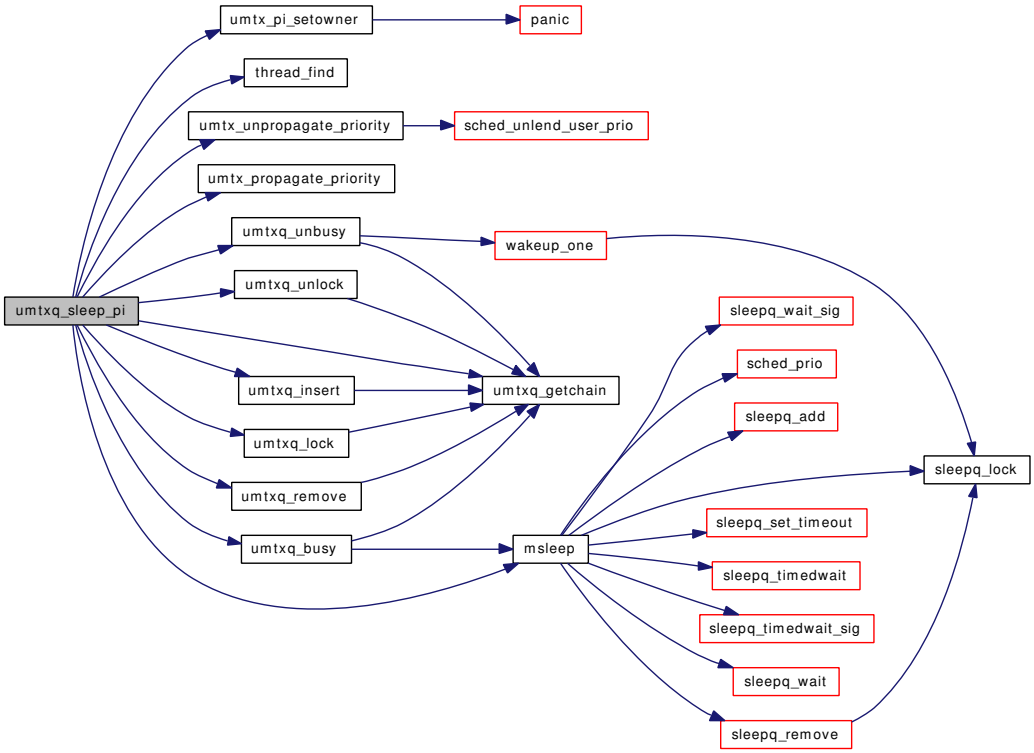
Definition at line 1470 of file kern_umtx.c.

References msleep(), umtx_pi::pi_owner, sched_lock, thread_find(), uc, umtx_pi_setowner(), umtx_propagate_priority(), umtx_unpropagate_priority(), umtxq_busy(), umtxq_getchain(), umtxq_insert(),

umtxq_lock(), UMTXQ_LOCKED_ASSERT, umtxq_remove(), umtxq_unbusy(), umtxq_unlock(), UPRI, and UQF_UMTXQ.

Referenced by _do_lock_pi().

Here is the call graph for this function:

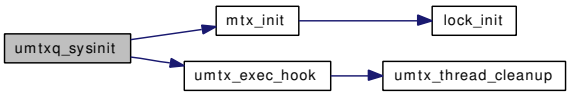


9.68.3.77 static void umtxq_sysinit (void *arg *__unused*) [static]

Definition at line 229 of file kern_umtx.c.

References mtx_init(), umtxq_chain::uc_busy, umtxq_chain::uc_waiters, UMTX_CHAINS, umtx_exec_hook(), umtx_pi_zone, and umtxq_chains.

Here is the call graph for this function:



9.68.3.78 static void umtxq_sysinit (void *) [static]

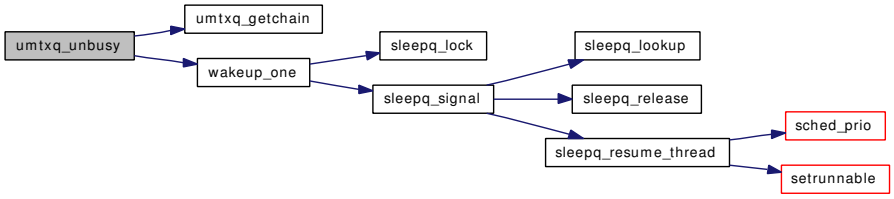
9.68.3.79 static void umtxq_unbusy (struct umtx_key * *key*) [inline, static]

Definition at line 308 of file kern_umtx.c.

References uc, umtxq_getchain(), and wakeup_one().

Referenced by _do_lock_normal(), _do_lock_pi(), _do_lock_pp(), _do_lock_umtx(), do_cv_broadcast(), do_cv_signal(), do_cv_wait(), do_set_ceiling(), do_unlock_normal(), do_unlock_pi(), do_unlock_pp(), do_unlock_umtx(), and umtxq_sleep_pi().

Here is the call graph for this function:



9.68.3.80 static void umtxq_unlock (struct umtx_key * *key*) [inline, static]

Definition at line 336 of file kern_umtx.c.

References uc, and umtxq_getchain().

Referenced by _do_lock_normal(), _do_lock_pi(), _do_lock_pp(), _do_lock_umtx(), do_cv_broadcast(), do_cv_signal(), do_cv_wait(), do_set_ceiling(), do_unlock_normal(), do_unlock_pi(), do_unlock_pp(), do_unlock_umtx(), do_wait(), kern_umtx_wake(), and umtxq_sleep_pi().

Here is the call graph for this function:

9.68.4 Variable Documentation

9.68.4.1 _umtx_op_func op_table[ ] [static]

Initial value:

```
{
        __umtx_op_lock_umtx,
        __umtx_op_unlock_umtx,
        __umtx_op_wait,
        __umtx_op_wake,
        __umtx_op_trylock_umutex,
        __umtx_op_lock_umutex,
        __umtx_op_unlock_umutex,
        __umtx_op_set_ceiling,
        __umtx_op_cv_wait,
        __umtx_op_cv_signal,
        __umtx_op_cv_broadcast
}
```

Definition at line 2520 of file kern_umtx.c.

Referenced by _umtx_op().

9.68.4.2 int umtx_dflt_spins = 0 [static]

Definition at line 197 of file kern_umtx.c.

Referenced by _do_lock_normal().

9.68.4.3 int umtx_max_spins = 3000 [static]

Definition at line 200 of file kern_umtx.c.

Referenced by _do_lock_normal().

9.68.4.4 int umtx_pi_allocated [static]

Definition at line 191 of file kern_umtx.c.

Referenced by umtx_pi_alloc(), and umtx_pi_free().

9.68.4.5 uma_zone_t umtx_pi_zone [static]

Definition at line 188 of file kern_umtx.c.

Referenced by umtx_pi_alloc(), umtx_pi_free(), and umtxq_sysinit().

9.68.4.6 struct umtxq_chain umtxq_chains[UMTX_CHAINS] [static]

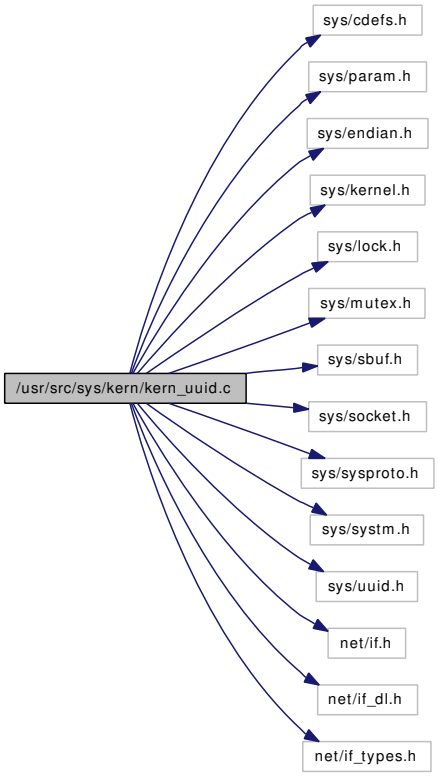
Definition at line 189 of file kern_umtx.c.

Referenced by umtxq_getchain(), and umtxq_sysinit().

9.69 /usr/src/sys/kern/kern_uuid.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/endian.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/sbuf.h>
#include <sys/socket.h>
#include <sys/sysproto.h>
#include <sys/systm.h>
#include <sys/uuid.h>
#include <net/if.h>
#include <net/if_dl.h>
#include <net/if_types.h>
```

Include dependency graph for kern_uuid.c:



Data Structures

- struct uuid_private
- struct uuidgen_args

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_uuid.c,v 1.11 2006/07/17 21:00:42 stefanf Exp $")
- CTASSERT (sizeof(struct uuid)==16)
- CTASSERT (sizeof(struct uuid_private)==16)
- MTX_SYSINIT (uuid_lock,&uuid_mutex,"UUID generator mutex lock", MTX_DEF)
- static void uuid_node (uint16_t *node)
- static uint64_t uuid_time (void)
- uuid * kern_uuidgen (struct uuid *store, size_t count)
- int uuidgen (struct thread *td, struct uuidgen_args *uap)
- int snprintf_uuid (char *buf, size_t sz, struct uuid *uuid)
- int printf_uuid (struct uuid *uuid)
- int sbuf_printf_uuid (struct sbuf *sb, struct uuid *uuid)
- void le_uuid_enc (void *buf, struct uuid const *uuid)
- void le_uuid_dec (void const *buf, struct uuid *uuid)
- void be_uuid_enc (void *buf, struct uuid const *uuid)
- void be_uuid_dec (void const *buf, struct uuid *uuid)
- int parse_uuid (const char *str, struct uuid *uuid)

Variables

- static struct uuid_private uuid_last
- static struct mtx uuid_mutex

9.69.1 Function Documentation

9.69.1.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_uuid. *c*, v 1.11 2006/07/17 21:00:42 stefanf Exp $")

9.69.1.2 void be_uuid_dec (void const * *buf*, struct uuid * *uuid*)

Definition at line 303 of file kern_uuid.c.

9.69.1.3 void be_uuid_enc (void * *buf*, struct uuid const * *uuid*)

Definition at line 287 of file kern_uuid.c.

9.69.1.4 CTASSERT (sizeof(struct uuid_private) = =16)

9.69.1.5 CTASSERT (sizeof(struct uuid) = =16)

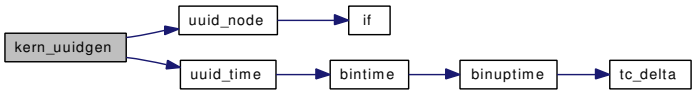
9.69.1.6 struct uuid* kern_uuidgen (struct uuid * *store*, size_t *count*)

Definition at line 135 of file kern_uuid.c.

References uuid_private::ll, uuid_private::node, uuid_private::seq, uuid_private::time, uuid_last, uuid_-node(), uuid_time(), and uuid_private::x.

Referenced by uuidgen().

Here is the call graph for this function:



9.69.1.7 void le_uuid_dec (void const * *buf*, struct uuid * *uuid*)

Definition at line 271 of file kern_uuid.c.

9.69.1.8 void le_uuid_enc (void * *buf*, struct uuid const * *uuid*)

Definition at line 255 of file kern_uuid.c.

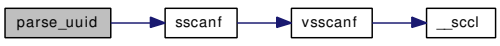
9.69.1.9 MTX_SYSINIT (uuid_lock, & *uuid_mutex*, "UUID generator mutex lock", MTX_DEF)

9.69.1.10 int parse_uuid (const char * *str*, struct uuid * *uuid*)

Definition at line 319 of file kern_uuid.c.

References sscanf().

Here is the call graph for this function:

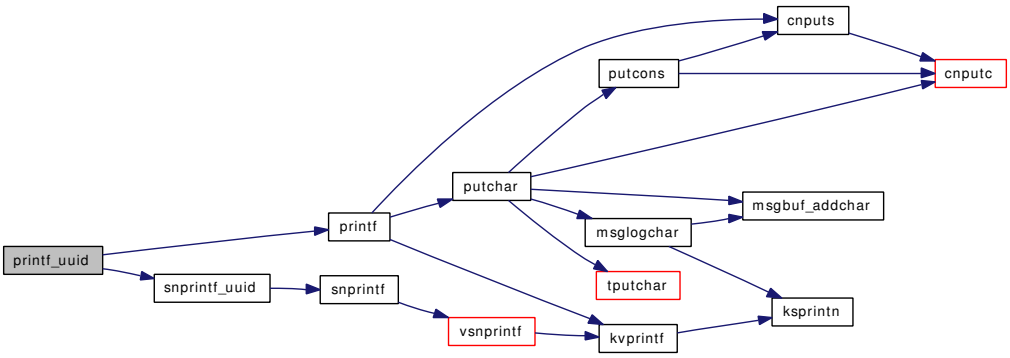


9.69.1.11 int printf_uuid (struct uuid * *uuid*)

Definition at line 220 of file kern_uuid.c.

References buf, printf(), and snprintf_uuid().

Here is the call graph for this function:

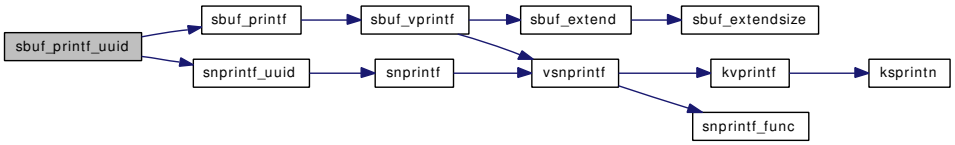


9.69.1.12 int sbuf_printf_uuid (struct sbuf * *sb*, struct uuid * *uuid*)

Definition at line 229 of file kern_uuid.c.

References buf, sbuf_printf(), and snprintf_uuid().

Here is the call graph for this function:



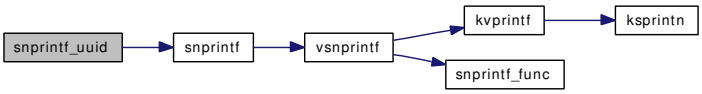
9.69.1.13 int snprintf_uuid (char * *buf*, size_t *sz*, struct uuid * *uuid*)

Definition at line 207 of file kern_uuid.c.

References uuid_private::node, uuid_private::seq, snprintf(), uuid_private::time, and uuid_private::x.

Referenced by printf_uuid(), and sbuf_printf_uuid().

Here is the call graph for this function:



9.69.1.14 static void uuid_node (uint16_t * *node*) [static]

Definition at line 88 of file kern_uuid.c.

References if().

Referenced by kern_uuidgen().

Here is the call graph for this function:

9.69.1.15 static uint64_t uuid_time (void) [static]

Definition at line 123 of file kern_uuid.c.

References bintime().

Referenced by kern_uuidgen().

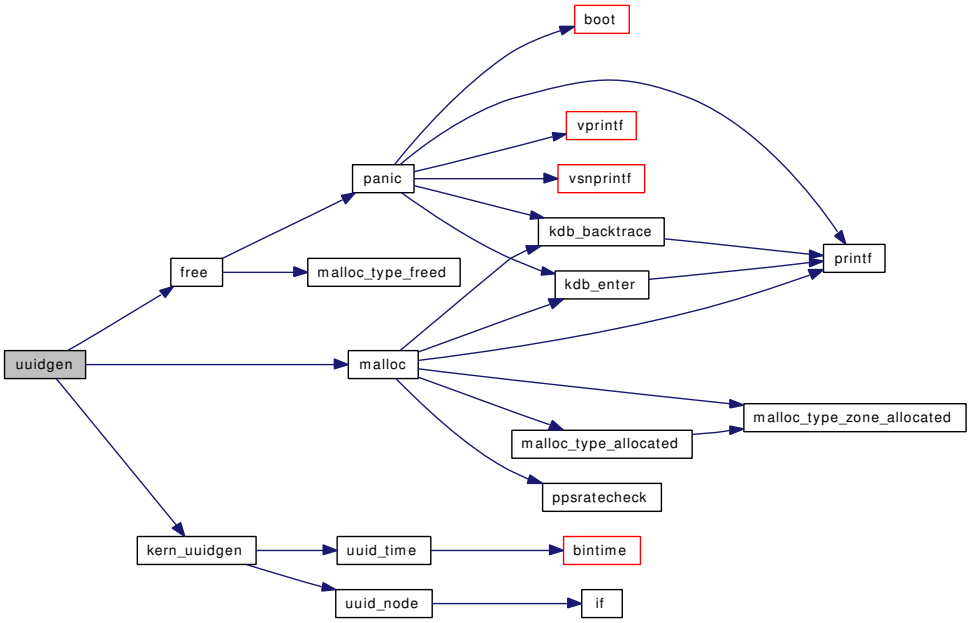
Here is the call graph for this function:

9.69.1.16 int uuidgen (struct thread * *td*, struct uuidgen_args * *uap*)

Definition at line 183 of file kern_uuid.c.

References uuidgen_args::count, free(), kern_uuidgen(), malloc(), and uuidgen_args::store.

Here is the call graph for this function:



9.69.2 Variable Documentation

9.69.2.1 struct uuid_private uuid_last [static]

Definition at line 72 of file kern_uuid.c.

Referenced by kern_uuidgen().

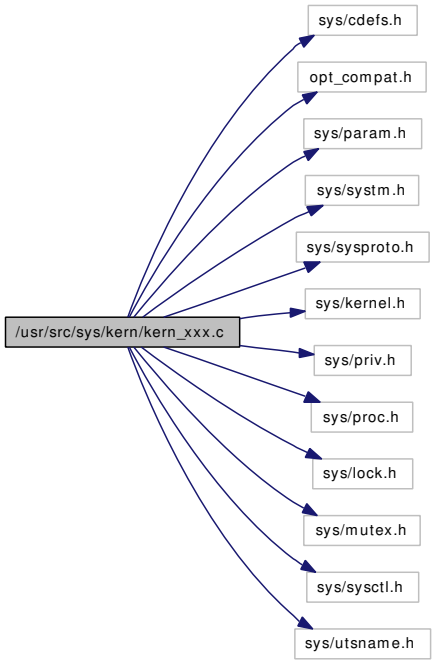
9.69.2.2 struct mtx uuid_mutex [static]

Definition at line 74 of file kern_uuid.c.

9.70 /usr/src/sys/kern/kern_xxx.c File Reference

```
#include <sys/cdefs.h>
#include "opt_compat.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/sysproto.h>
#include <sys/kernel.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/sysctl.h>
#include <sys/utsname.h>
```

Include dependency graph for kern_xxx.c:



Data Structures

- struct uname_args
- struct getdomainname_args
- struct setdomainname_args

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/kern_xxx.c,v 1.47 2006/11/06 13:42:01 rwatson Exp $")
- int uname (struct thread *td, struct uname_args *uap)
- int getdomainname (struct thread *td, struct getdomainname_args *uap)
- int setdomainname (struct thread *td, struct setdomainname_args *uap)

9.70.1 Function Documentation

9.70.1.1 __FBSDID ("$FreeBSD: src/sys/kern/kern_xxx. *c*, v 1.47 2006/11/06 13:42:01 rwatson Exp $")

9.70.1.2 int getdomainname (struct thread * *td*, struct getdomainname_args * *uap*)

Definition at line 266 of file kern_xxx.c.

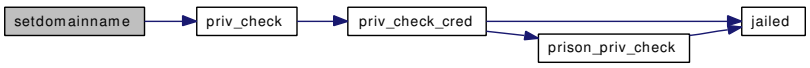
References getdomainname_args::domainname, domainname, Giant, and getdomainname_args::len.

9.70.1.3 int setdomainname (struct thread * *td*, struct setdomainname_args * *uap*)

Definition at line 294 of file kern_xxx.c.

References setdomainname_args::domainname, domainname, Giant, setdomainname_args::len, and priv_check().

Here is the call graph for this function:

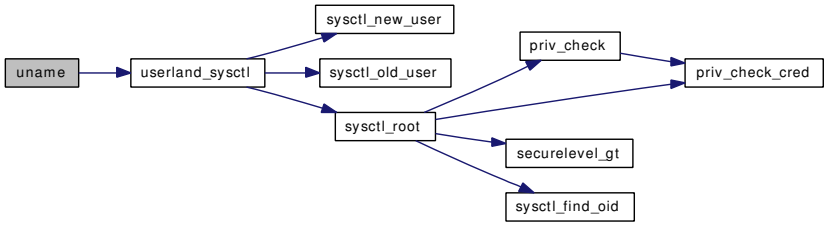


9.70.1.4 int uname (struct thread * *td*, struct uname_args * *uap*)

Definition at line 183 of file kern_xxx.c.

References Giant, uname_args::name, and userland_sysctl().

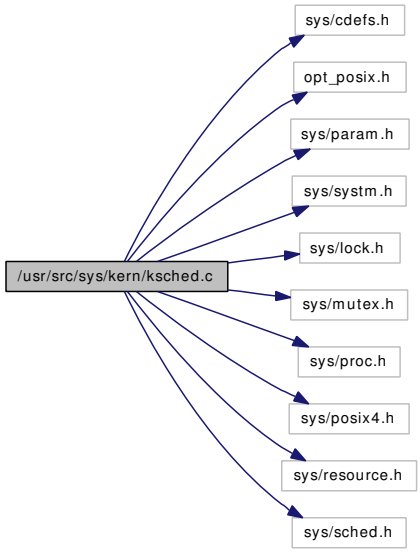
Here is the call graph for this function:



9.71 /usr/src/sys/kern/ksched.c File Reference

```
#include <sys/cdefs.h>
#include "opt_posix.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/posix4.h>
#include <sys/resource.h>
#include <sys/sched.h>
```

Include dependency graph for ksched.c:



Data Structures

- struct ksched

Defines

- #define p4prio_to_rtpprio(P) (RTP_PRIO_MAX - (P))
- #define rtpprio_to_p4prio(P) (RTP_PRIO_MAX - (P))
- #define P1B_PRIO_MIN rtpprio_to_p4prio(RTP_PRIO_MAX)
- #define P1B_PRIO_MAX rtpprio_to_p4prio(RTP_PRIO_MIN)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/ksched.c,v 1.35 2006/12/06 06:34:55 julian Exp $")
- int ksched_attach (struct ksched **p)
- int ksched_detach (struct ksched *ks)
- static __inline int getscheduler (struct ksched *ksched, struct thread *td, int *policy)
- int ksched_setparam (struct ksched *ksched, struct thread *td, const struct sched_param *param)
- int ksched_getparam (struct ksched *ksched, struct thread *td, struct sched_param *param)
- int ksched_setscheduler (struct ksched *ksched, struct thread *td, int policy, const struct sched_param *param)
- int ksched_getscheduler (struct ksched *ksched, struct thread *td, int *policy)
- int ksched_yield (struct ksched *ksched)
- int ksched_get_priority_max (struct ksched *ksched, int policy, int *prio)
- int ksched_get_priority_min (struct ksched *ksched, int policy, int *prio)
- int ksched_rr_get_interval (struct ksched *ksched, struct thread *td, struct timespec *timespec)

9.71.1 Define Documentation

9.71.1.1 #define P1B_PRIO_MAX rtpprio_to_p4prio(RTP_PRIO_MIN)

Definition at line 99 of file ksched.c.

Referenced by ksched_setscheduler().

9.71.1.2 #define P1B_PRIO_MIN rtpprio_to_p4prio(RTP_PRIO_MAX)

Definition at line 98 of file ksched.c.

Referenced by ksched_get_priority_min(), and ksched_setscheduler().

9.71.1.3 #define p4prio_to_rtpprio(P) (RTP_PRIO_MAX - (P))

Definition at line 93 of file ksched.c.

Referenced by ksched_setscheduler().

9.71.1.4 #define rtpprio_to_p4prio(P) (RTP_PRIO_MAX - (P))

Definition at line 94 of file ksched.c.

Referenced by ksched_getparam().

9.71.2 Function Documentation

9.71.2.1 __FBSDID ("$FreeBSD: src/sys/kern/ksched. *c*, v 1.35 2006/12/06 06:34:55 julian Exp $")

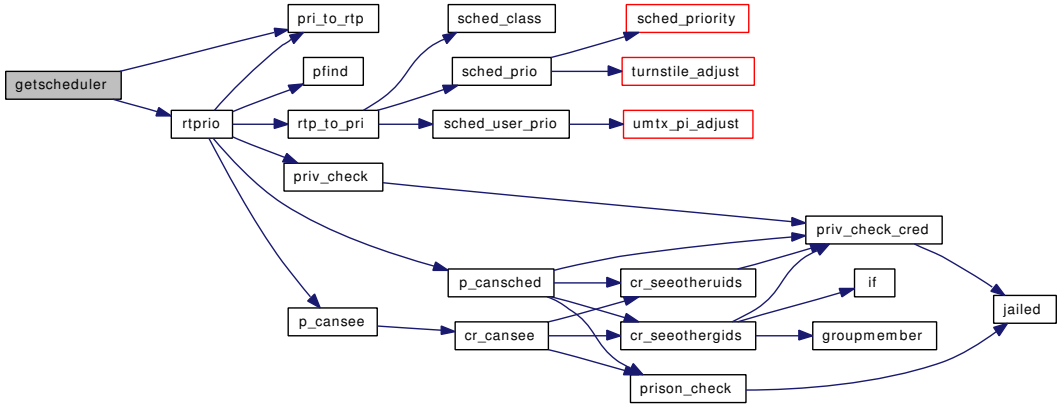
9.71.2.2 static __inline int getscheduler (struct ksched * *ksched*, struct thread * *td*, int * *policy*) [static]

Definition at line 102 of file ksched.c.

References pri_to_rtp(), rtprio(), and sched_lock.

Referenced by ksched_getscheduler(), and ksched_setparam().

Here is the call graph for this function:

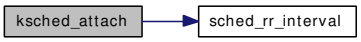


9.71.2.3 int ksched_attach (struct ksched ** *p*)

Definition at line 58 of file ksched.c.

References sched_rr_interval().

Here is the call graph for this function:



9.71.2.4 int ksched_detach (struct ksched * *ks*)

Definition at line 70 of file ksched.c.

9.71.2.5 int ksched_get_priority_max (struct ksched * *ksched*, int *policy*, int * *prio*)

Definition at line 233 of file ksched.c.

9.71.2.6 int ksched_get_priority_min (struct ksched * *ksched*, int *policy*, int * *prio*)

Definition at line 256 of file ksched.c.

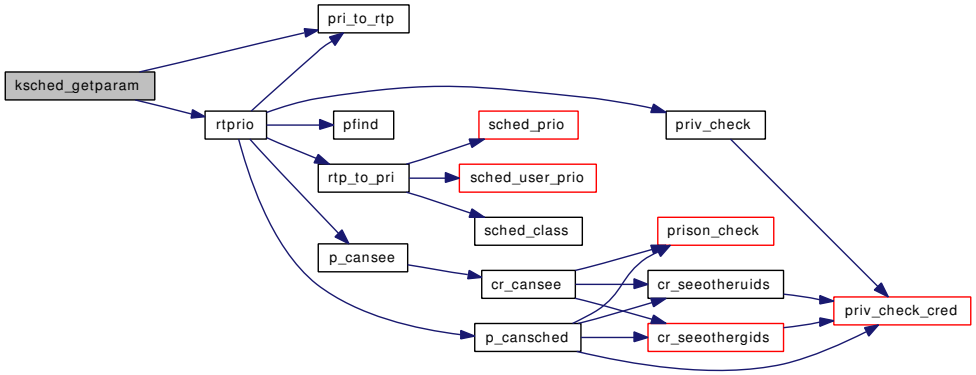
References P1B_PRIO_MIN.

9.71.2.7 int ksched_getparam (struct ksched * *ksched*, struct thread * *td*, struct sched_param * *param*)

Definition at line 149 of file ksched.c.

References pri_to_rtp(), rtpprio_to_p4prio, rtprio(), and sched_lock.

Here is the call graph for this function:

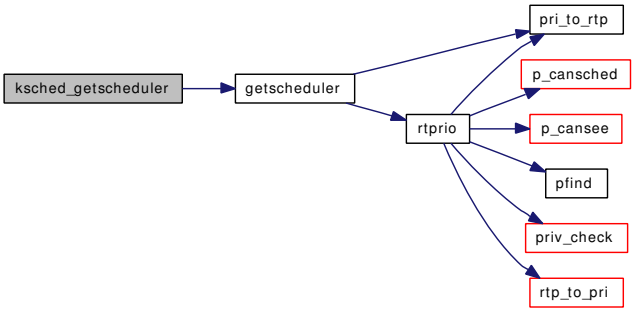


9.71.2.8 int ksched_getscheduler (struct ksched * *ksched*, struct thread * *td*, int * *policy*)

Definition at line 218 of file ksched.c.

References getscheduler().

Here is the call graph for this function:



9.71.2.9 int ksched_rr_get_interval (struct ksched * *ksched*, struct thread * *td*, struct timespec * *timespec*)

Definition at line 279 of file ksched.c.

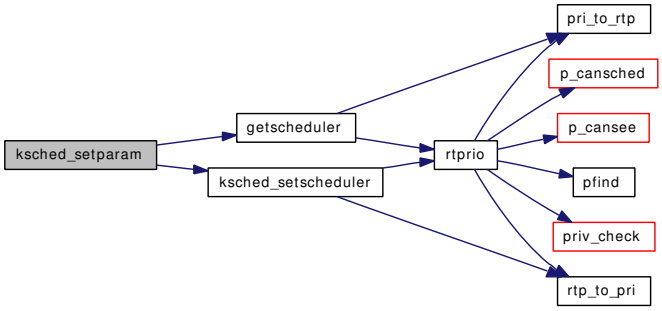
References ksched::rr_interval.

9.71.2.10 int ksched_setparam (struct ksched * *ksched*, struct thread * *td*, const struct sched_param * *param*)

Definition at line 129 of file ksched.c.

References getscheduler(), and ksched_setscheduler().

Here is the call graph for this function:



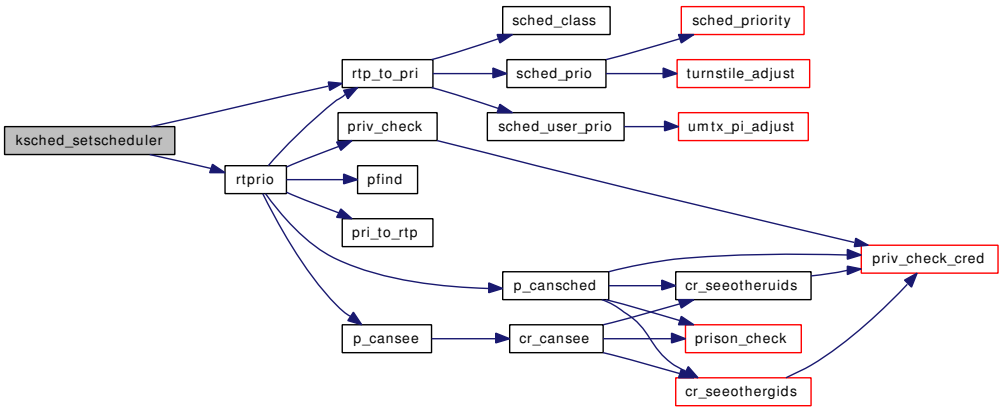
9.71.2.11 int ksched_setscheduler (struct ksched * *ksched*, struct thread * *td*, int *policy*, const struct sched_param * *param*)

Definition at line 171 of file ksched.c.

References P1B_PRIO_MAX, P1B_PRIO_MIN, p4prio_to_rtpprio, rtp_to_pri(), rtprio(), and sched_lock.

Referenced by ksched_setparam().

Here is the call graph for this function:

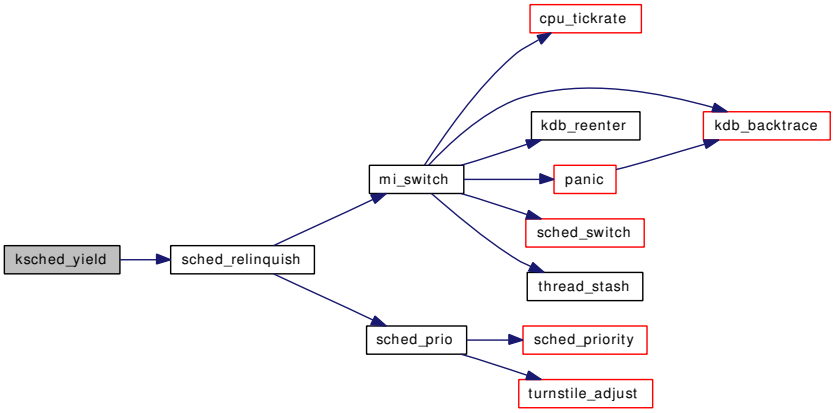


9.71.2.12 int ksched_yield (struct ksched * *ksched*)

Definition at line 226 of file ksched.c.

References sched_relinquish().

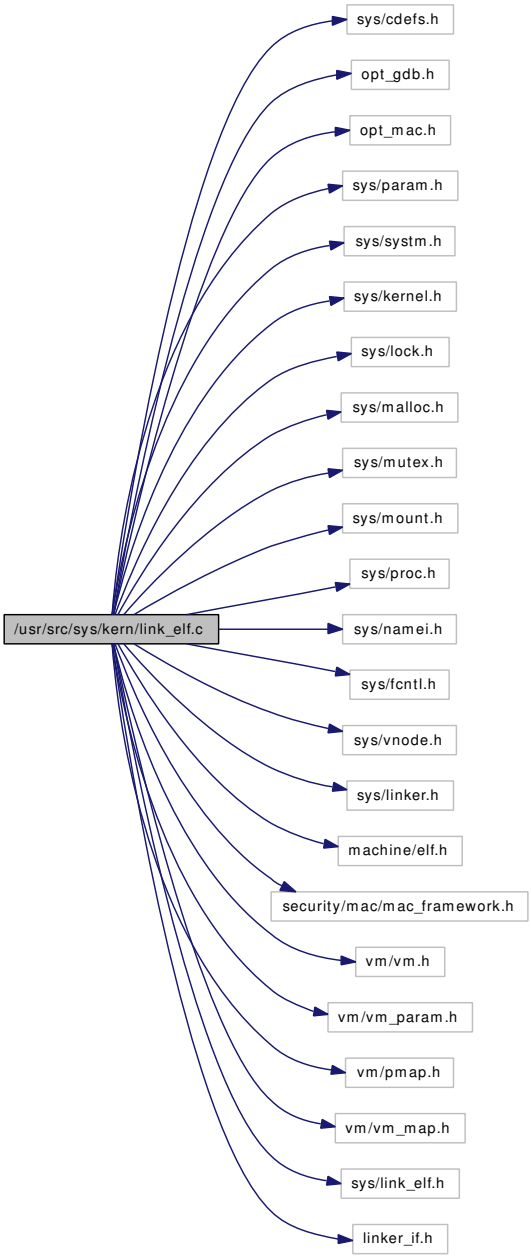
Here is the call graph for this function:


cpu_tickrate
kdb_reenter
kdb_backtrace
mi_switch
panic
sched_switch
ksched_yield
sched_relinquish
thread_stash
sched_prio
sched_priority
turnstile_adjust

9.72 /usr/src/sys/kern/link_elf.c File Reference

```
#include <sys/cdefs.h>
#include "opt_gdb.h"
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/mount.h>
#include <sys/proc.h>
#include <sys/namei.h>
#include <sys/fcntl.h>
#include <sys/vnode.h>
#include <sys/linker.h>
#include <machine/elf.h>
#include <security/mac/mac_framework.h>
#include <vm/vm.h>
#include <vm/vm_param.h>
#include <vm/pmap.h>
#include <vm/vm_map.h>
#include <sys/link_elf.h>
#include "linker_if.h"
```

Include dependency graph for link_elf.c:



Data Structures

- struct elf_file

Defines

- #define MAXSEGS 4

Typedefs

- typedef elf_file * elf_file_t

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/link_elf.c,v 1.91 2006/10/22 11:52:13 rwatson Exp $")
- static int link_elf_link_common_finish (linker_file_t)
- static int link_elf_link_preload (linker_class_t cls, const char *, linker_file_t *)
- static int link_elf_link_preload_finish (linker_file_t)
- static int link_elf_load_file (linker_class_t, const char *, linker_file_t *)
- static int link_elf_lookup_symbol (linker_file_t, const char *, c_linker_sym_t *)
- static int link_elf_symbol_values (linker_file_t, c_linker_sym_t, linker_symval_t *)
- static int link_elf_search_symbol (linker_file_t, caddr_t value, c_linker_sym_t *sym, long *diffp)
- static void link_elf_unload_file (linker_file_t)
- static void link_elf_unload_preload (linker_file_t)
- static int link_elf_lookup_set (linker_file_t, const char *, void ***, void ***, int *)
- static int link_elf_each_function_name (linker_file_t, int(*)(const char *, void *), void *)
- static void link_elf_reloc_local (linker_file_t)
- static Elf_Addr elf_lookup (linker_file_t lf, Elf_Size symidx, int deps)
- static int parse_dynamic (elf_file_t ef)
- static int relocate_file (elf_file_t ef)
- static int link_elf_preload_parse_symbols (elf_file_t ef)
- static void link_elf_error (const char *s)
- static void link_elf_init (void *arg)
- SYSINIT (link_elf, SI_SUB_KLD, SI_ORDER_THIRD, link_elf_init, 0)
- static const char * symbol_name (elf_file_t ef, Elf_Size r_info)
- static unsigned long elf_hash (const char *name)
- const Elf_Sym * elf_get_sym (linker_file_t lf, Elf_Size symidx)
- const char * elf_get_symname (linker_file_t lf, Elf_Size symidx)

Variables

- static kobj_method_t link_elf_methods [ ]
- static struct linker_class link_elf_class
- _dynamic _DYNAMIC

9.72.1 Define Documentation

9.72.1.1 #define MAXSEGS 4

Definition at line 67 of file link_elf.c.

Referenced by link_elf_load_file().

9.72.2 Typedef Documentation

9.72.2.1 typedef struct elf_file * elf_file_t

9.72.3 Function Documentation

9.72.3.1 __FBSDID ("$FreeBSD: src/sys/kern/link_elf. *c*, v 1.91 2006/10/22 11:52:13 rwatson Exp $")

9.72.3.2 const Elf_Sym* elf_get_sym (linker_file_t *lf*, Elf_Size *symidx*)

Definition at line 1240 of file link_elf.c.

References elf_file::nchains, and elf_file::symtab.

9.72.3.3 const char* elf_get_symname (linker_file_t *lf*, Elf_Size *symidx*)

Definition at line 1250 of file link_elf.c.

References elf_file::nchains, elf_file::strtab, and elf_file::symtab.

9.72.3.4 static unsigned long elf_hash (const char * *name*) [static]

Definition at line 1005 of file link_elf.c.

Referenced by link_elf_lookup_symbol().

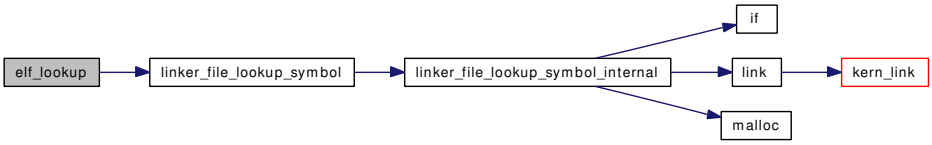
9.72.3.5 static Elf_Addr elf_lookup (linker_file_t *lf*, Elf_Size *symidx*, int *deps*) [static]

Definition at line 1269 of file link_elf.c.

References elf_file::address, linker_file_lookup_symbol(), elf_file::nchains, elf_file::strtab, and elf_file::symtab.

Referenced by link_elf_reloc_local(), and relocate_file().

Here is the call graph for this function:



9.72.3.6 static int link_elf_each_function_name (linker_file_t, int(*)(const char *, void *), void *) [static]

Definition at line 1206 of file link_elf.c.

References elf_file::ddbstrtab, and elf_file::ddbsymtab.

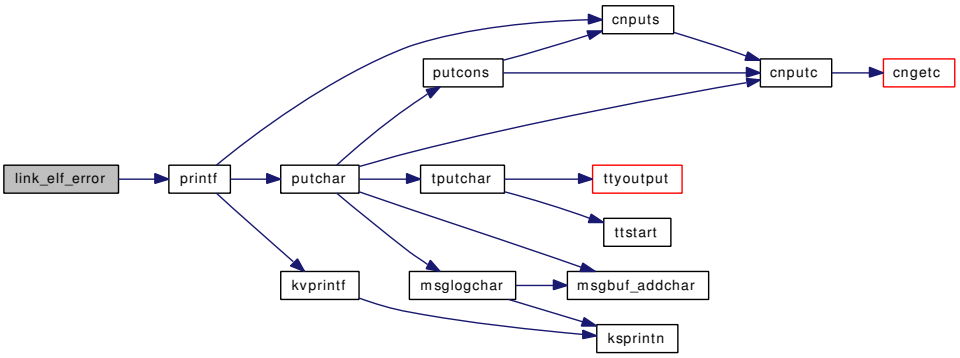
9.72.3.7 static void link_elf_error (const char * s) [static]

Definition at line 218 of file link_elf.c.

References printf().

Referenced by link_elf_load_file().

Here is the call graph for this function:

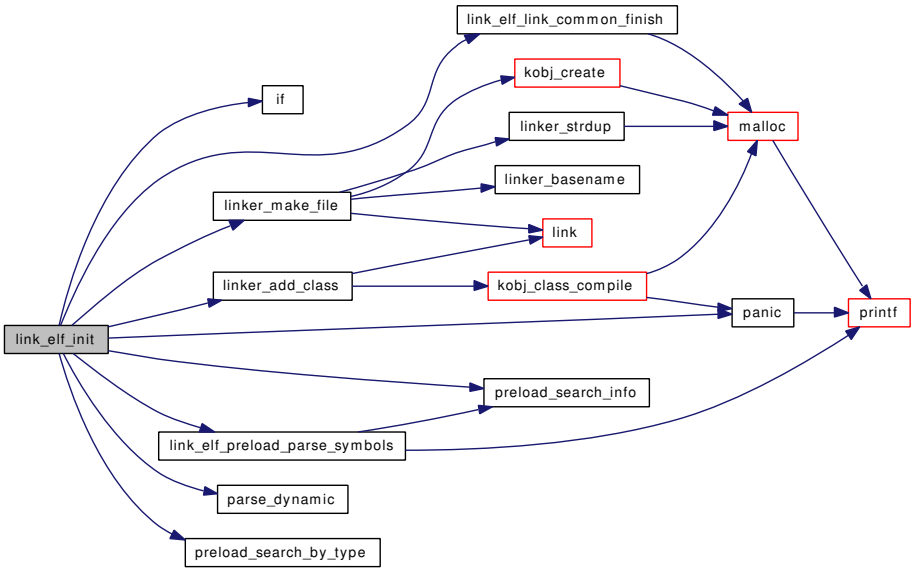


9.72.3.8 static void link_elf_init (void * arg) [static]

Definition at line 256 of file link_elf.c.

References __ELF_WORD_SIZE, elf_file::address, elf_file::dynamic, if(), link_elf_link_common_finish(), link_elf_preload_parse_symbols(), linker_add_class(), linker_kernel_file, linker_make_file(), elf_file::object, panic(), parse_dynamic(), preload_search_by_type(), preload_search_info(), and elf_file::preloaded.

Here is the call graph for this function:



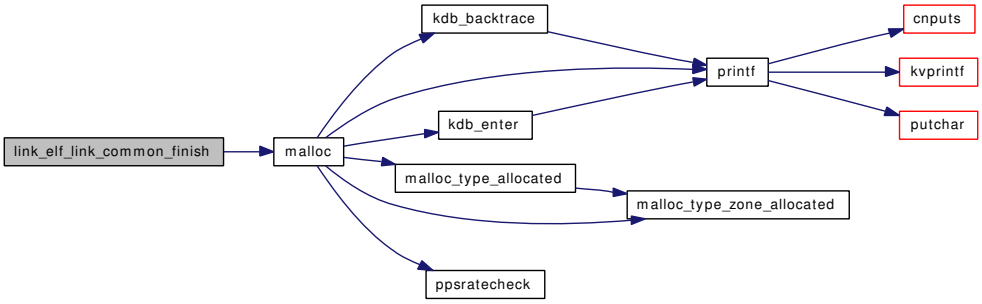
9.72.3.9 static int link_elf_link_common_finish (linker_file_t) [static]

Definition at line 228 of file link_elf.c.

References elf_file::address, elf_file::dynamic, and malloc().

Referenced by link_elf_init(), link_elf_link_preload_finish(), and link_elf_load_file().

Here is the call graph for this function:

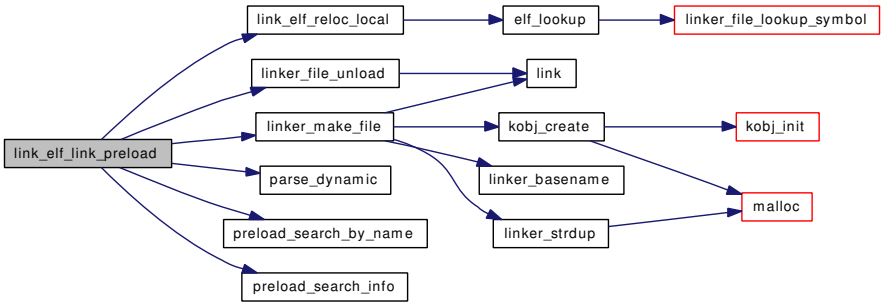


9.72.3.10 static int link_elf_link_preload (linker_class_t *cls*, const char *, linker_file_t *) [static]

Definition at line 454 of file link_elf.c.

References __ELF_WORD_SIZE, elf_file::address, elf_file::dynamic, link_elf_reloc_local(), linker_file_unload(), linker_make_file(), elf_file::modptr, elf_file::object, parse_dynamic(), preload_search_by_name(), preload_search_info(), and elf_file::preloaded.

Here is the call graph for this function:

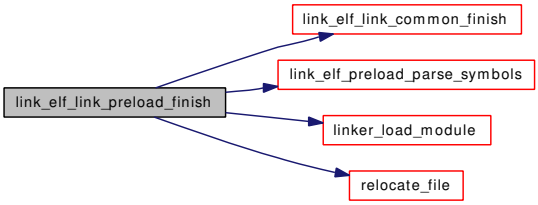


9.72.3.11 static int link_elf_link_preload_finish (linker_file_t) [static]

Definition at line 508 of file link_elf.c.

References elf_file::dynamic, link_elf_link_common_finish(), link_elf_preload_parse_symbols(), linker_load_module(), relocate_file(), and elf_file::strtab.

Here is the call graph for this function:

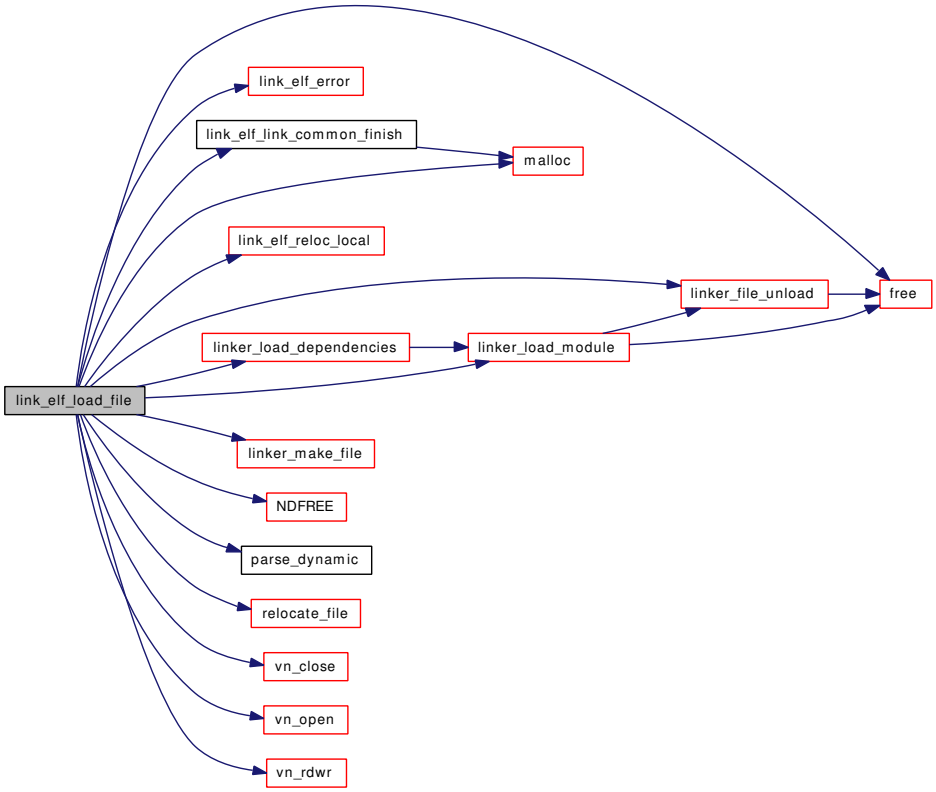


9.72.3.12 static int link_elf_load_file (linker_class_t, const char *, linker_file_t *) [static]

Definition at line 533 of file link_elf.c.

References elf_file::address, elf_file::ddbstrcnt, elf_file::ddbstrtab, elf_file::ddbsymcnt, elf_file::ddbsymtab, elf_file::dynamic, free(), Giant, link_elf_error(), link_elf_link_common_finish(), link_elf_reloc_local(), linker_file_unload(), linker_load_dependencies(), linker_load_module(), linker_make_file(), malloc(), MAXSEGS, NDFREE(), elf_file::object, parse_dynamic(), relocate_file(), elf_file::strbase, elf_file::strtab, elf_file::symbase, td, vn_close(), vn_open(), and vn_rdwr().

Here is the call graph for this function:

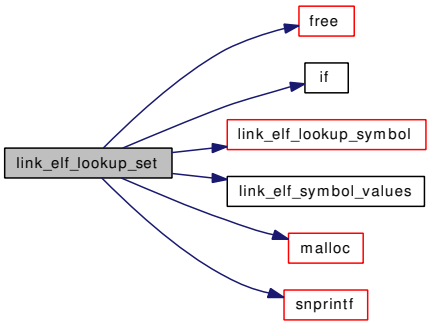


9.72.3.13 static int link_elf_lookup_set (linker_file_t, const char *, void ***, void ***, int *) [static]

Definition at line 1151 of file link_elf.c.

References free(), if(), link_elf_lookup_symbol(), link_elf_symbol_values(), malloc(), and snprintf().

Here is the call graph for this function:



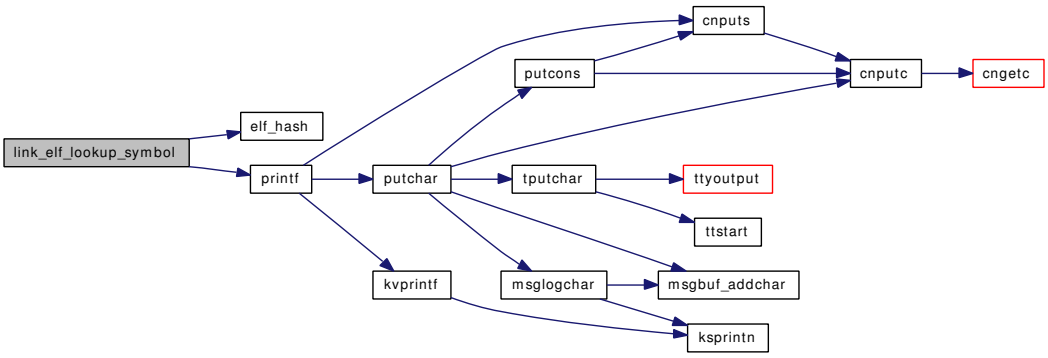
9.72.3.14 static int link_elf_lookup_symbol (linker_file_t, const char *, c_linker_sym_t *) [static]

Definition at line 1021 of file link_elf.c.

References elf_file::buckets, elf_file::chains, elf_file::ddbstrtab, elf_file::ddbsymtab, elf_hash(), elf_file::nbuckets, elf_file::nchains, printf(), elf_file::strtab, and elf_file::symtab.

Referenced by link_elf_lookup_set().

Here is the call graph for this function:



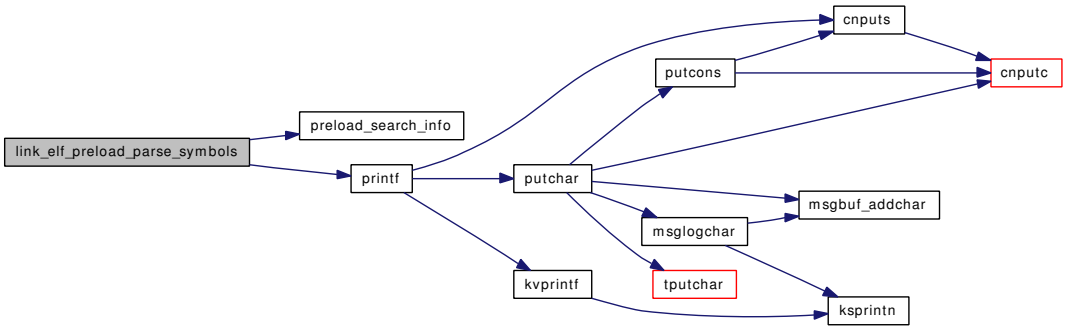
9.72.3.15 static int link_elf_preload_parse_symbols (elf_file_t *ef*) [static]

Definition at line 314 of file link_elf.c.

References elf_file::ddbstrcnt, elf_file::ddbstrtab, elf_file::ddbsymcnt, elf_file::ddbsymtab, elf_file::modptr, preload_search_info(), and printf().

Referenced by link_elf_init(), and link_elf_link_preload_finish().

Here is the call graph for this function:



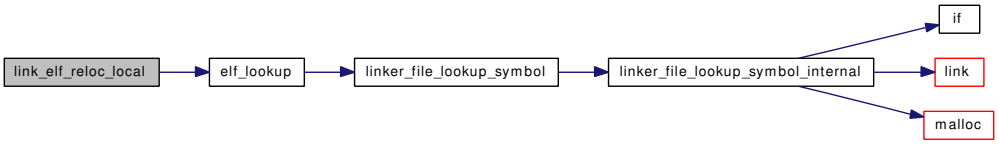
9.72.3.16 static void link_elf_reloc_local (linker_file_t) [static]

Definition at line 1309 of file link_elf.c.

References elf_file::address, elf_lookup(), elf_file::rel, elf_file::rela, elf_file::relasize, and elf_file::relsize.

Referenced by link_elf_link_preload(), and link_elf_load_file().

Here is the call graph for this function:



9.72.3.17 static int link_elf_search_symbol (linker_file_t, caddr_t *value*, c_linker_sym_t * *sym*, long * *diffp*) [static]

Definition at line 1112 of file link_elf.c.

References elf_file::address, elf_file::ddbsymtab, and diff.

9.72.3.18 static int link_elf_symbol_values (linker_file_t, c_linker_sym_t, linker_symval_t *) [static]

Definition at line 1089 of file link_elf.c.

References elf_file::address, elf_file::ddbstrtab, elf_file::ddbsymcnt, elf_file::ddbsymtab, elf_file::nchains, elf_file::strtab, and elf_file::symtab.

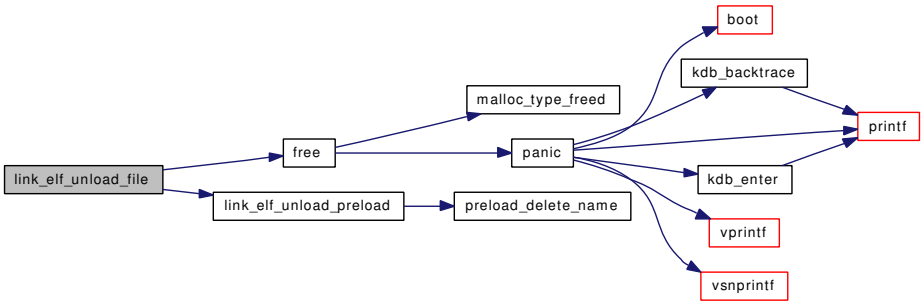
Referenced by link_elf_lookup_set().

9.72.3.19 static void link_elf_unload_file (linker_file_t) [static]

Definition at line 872 of file link_elf.c.

References elf_file::address, free(), link_elf_unload_preload(), elf_file::object, elf_file::preloaded, elf_file::strbase, and elf_file::symbase.

Here is the call graph for this function:



9.72.3.20 static void link_elf_unload_preload (linker_file_t) [static]

Definition at line 910 of file link_elf.c.

References preload_delete_name().

Referenced by link_elf_unload_file().

Here is the call graph for this function:

9.72.3.21 static int parse_dynamic (elf_file_t *ef*) [static]

Definition at line 365 of file link_elf.c.

References elf_file::address, elf_file::buckets, elf_file::chains, elf_file::ddbstrcnt, elf_file::ddbstrtab, elf_file::ddbsymcnt, elf_file::ddbsymtab, elf_file::dynamic, elf_file::got, elf_file::nbuckets, elf_file::nchains, elf_file::pltrel, elf_file::pltrela, elf_file::pltrelasize, elf_file::pltrelsize, elf_file::rel, elf_file::rela, elf_file::relasize, elf_file::relsize, elf_file::strsz, elf_file::strtab, and elf_file::symtab.

Referenced by link_elf_init(), link_elf_link_preload(), and link_elf_load_file().

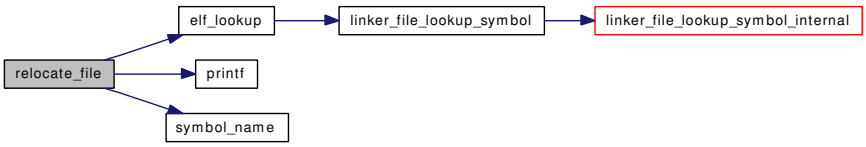
9.72.3.22 static int relocate_file (elf_file_t *ef*) [static]

Definition at line 929 of file link_elf.c.

References elf_file::address, elf_lookup(), elf_file::lf, elf_file::pltrel, elf_file::pltrela, elf_file::pltrelasize, elf_file::pltrelsize, printf(), elf_file::rel, elf_file::rela, elf_file::relasize, elf_file::relsize, and symbol_name().

Referenced by link_elf_link_preload_finish(), and link_elf_load_file().

Here is the call graph for this function:



9.72.3.23 static const char* symbol_name (elf_file_t *ef*, Elf_Size *r_info*) [static]

Definition at line 917 of file link_elf.c.

References elf_file::strtab, and elf_file::symtab.

Referenced by relocate_file().

9.72.3.24 SYSINIT (link_elf, SI_SUB_KLD, SI_ORDER_THIRD, link_elf_init, 0)

9.72.4 Variable Documentation

9.72.4.1 struct _dynamic _DYNAMIC

9.72.4.2 struct linker_class link_elf_class [static]

Initial value:

```
{
    "elf32",


    link_elf_methods, sizeof(struct elf_file)
}
```

Definition at line 140 of file link_elf.c.

9.72.4.3 kobj_method_t link_elf_methods[ ] [static]

Initial value:

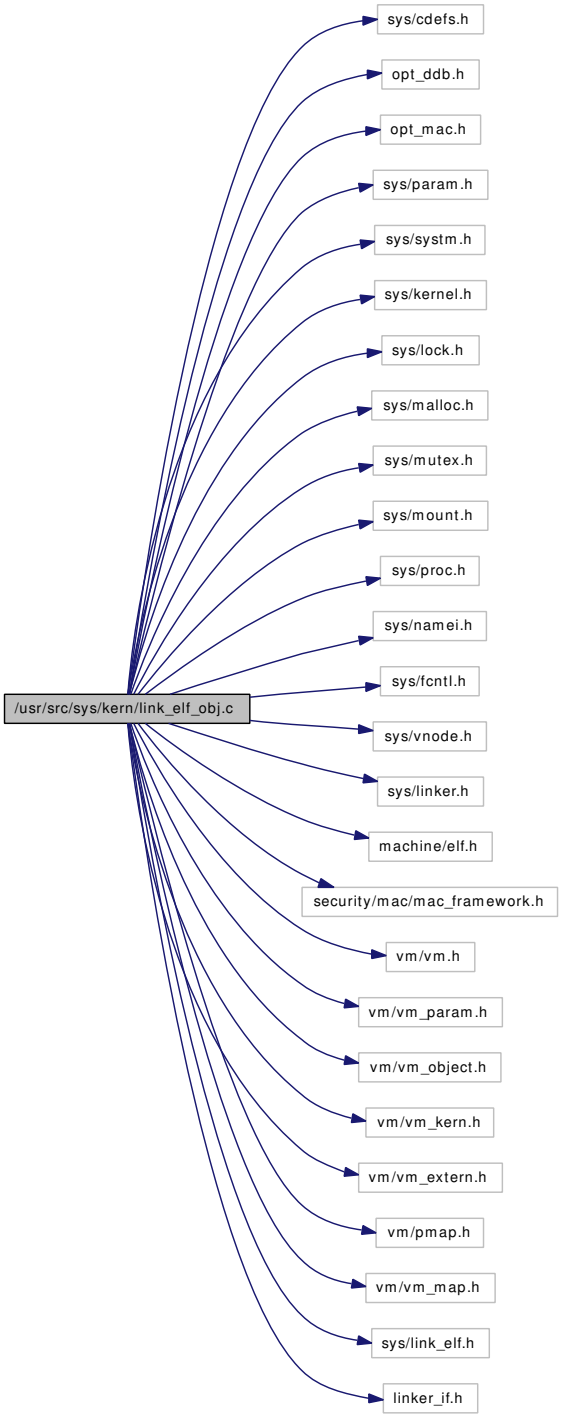
```
{
    KOBJMETHOD(linker_lookup_symbol,     link_elf_lookup_symbol),
    KOBJMETHOD(linker_symbol_values,     link_elf_symbol_values),
    KOBJMETHOD(linker_search_symbol,     link_elf_search_symbol),
    KOBJMETHOD(linker_unload,            link_elf_unload_file),
    KOBJMETHOD(linker_load_file,         link_elf_load_file),
    KOBJMETHOD(linker_link_preload,      link_elf_link_preload),
    KOBJMETHOD(linker_link_preload_finish, link_elf_link_preload_finish),
    KOBJMETHOD(linker_lookup_set,        link_elf_lookup_set),
    KOBJMETHOD(linker_each_function_name, link_elf_each_function_name),
    { 0, 0 }
}
```

Definition at line 127 of file link_elf.c.

9.73 /usr/src/sys/kern/link_elf_obj.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ddb.h"
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/mount.h>
#include <sys/proc.h>
#include <sys/namei.h>
#include <sys/fcntl.h>
#include <sys/vnode.h>
#include <sys/linker.h>
#include <machine/elf.h>
#include <security/mac/mac_framework.h>
#include <vm/vm.h>
#include <vm/vm_param.h>
#include <vm/vm_object.h>
#include <vm/vm_kern.h>
#include <vm/vm_extern.h>
#include <vm/pmap.h>
#include <vm/vm_map.h>
#include <sys/link_elf.h>
#include "linker_if.h"
```

Include dependency graph for link_elf_obj.c:



Data Structures

- struct Elf_progent
- struct Elf_relent
- struct Elf_relaent
- struct elf_file

Typedefs

- typedef elf_file * elf_file_t

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/link_elf_obj.c,v 1.94 2006/11/30 10:50:29 kib Exp $")
- static int link_elf_link_preload (linker_class_t cls, const char *, linker_file_t *)
- static int link_elf_link_preload_finish (linker_file_t)
- static int link_elf_load_file (linker_class_t, const char *, linker_file_t *)
- static int link_elf_lookup_symbol (linker_file_t, const char *, c_linker_sym_t *)
- static int link_elf_symbol_values (linker_file_t, c_linker_sym_t, linker_symval_t *)
- static int link_elf_search_symbol (linker_file_t, caddr_t value, c_linker_sym_t *sym, long *diffp)
- static void link_elf_unload_file (linker_file_t)
- static int link_elf_lookup_set (linker_file_t, const char *, void ***, void ***, int *)
- static int link_elf_each_function_name (linker_file_t, int(*)(const char *, void *), void *)
- static void link_elf_reloc_local (linker_file_t)
- static Elf_Addr elf_obj_lookup (linker_file_t lf, Elf_Size symidx, int deps)
- static int relocate_file (elf_file_t ef)
- static void link_elf_error (const char *s)
- static void link_elf_init (void *arg)
- SYSINIT (link_elf_obj, SI_SUB_KLD, SI_ORDER_SECOND, link_elf_init, 0)
- static const char * symbol_name (elf_file_t ef, Elf_Size r_info)
- static Elf_Addr findbase (elf_file_t ef, int sec)
- static void link_elf_fix_link_set (elf_file_t ef)

Variables

- static kobj_method_t link_elf_methods [ ]
- static struct linker_class link_elf_class

9.73.1 Typedef Documentation

9.73.1.1 typedef struct elf_file * elf_file_t

9.73.2 Function Documentation

9.73.2.1 __FBSDID ("$FreeBSD: src/sys/kern/link_elf_obj. *c*, v 1.94 2006/11/30 10:50:29 kib Exp $")

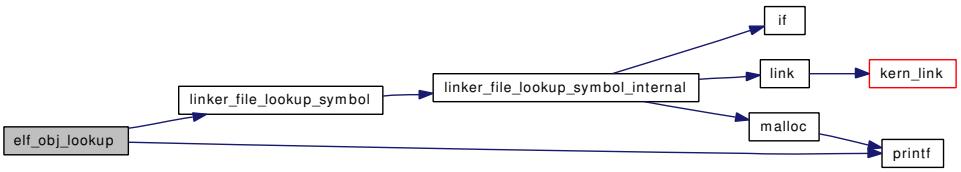
9.73.2.2 static Elf_Addr elf_obj_lookup (linker_file_t *lf*, Elf_Size *symidx*, int *deps*) [static]

Definition at line 1075 of file link_elf_obj.c.

References elf_file::ddbstrtab, elf_file::ddbsymcnt, elf_file::ddbsymtab, linker_file_lookup_symbol(), printf(), and ret.

Referenced by link_elf_reloc_local(), and relocate_file().

Here is the call graph for this function:



9.73.2.3 static Elf_Addr findbase (elf_file_t *ef*, int *sec*) [static]

Definition at line 861 of file link_elf_obj.c.

References Elf_progent::addr, elf_file::nprogtab, elf_file::progtab, and Elf_progent::sec.

Referenced by link_elf_reloc_local(), and relocate_file().

9.73.2.4 static int link_elf_each_function_name (linker_file_t, int(∗)(const char ∗, void ∗), void ∗) [static]

Definition at line 1048 of file link_elf_obj.c.

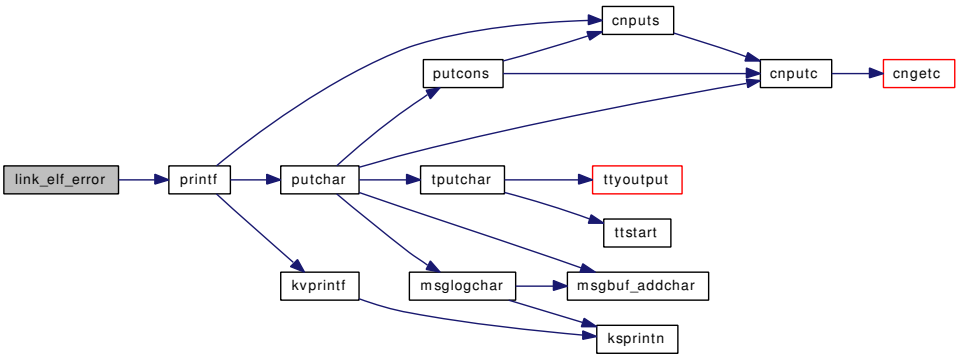
References elf_file::ddbstrtab, and elf_file::ddbsymtab.

9.73.2.5 static void link_elf_error (const char ∗ *s*) [static]

Definition at line 156 of file link_elf_obj.c.

References printf().

Here is the call graph for this function:



9.73.2.6 static void link_elf_fix_link_set (elf_file_t *ef*) [static]

Definition at line 1118 of file link_elf_obj.c.

References Elf_progent::addr, elf_file::ddbstrtab, elf_file::ddbsymcnt, elf_file::ddbsymtab, if(), Elf_progent::name, elf_file::nprogtab, elf_file::progtab, and Elf_progent::size.

Referenced by link_elf_reloc_local().

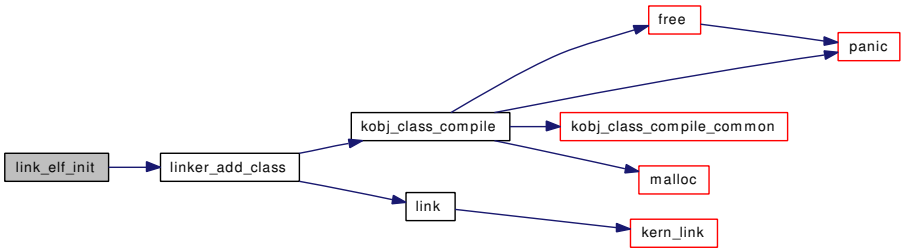
Here is the call graph for this function:

9.73.2.7 static void link_elf_init (void * *arg*) [static]

Definition at line 162 of file link_elf_obj.c.

References linker_add_class().

Here is the call graph for this function:

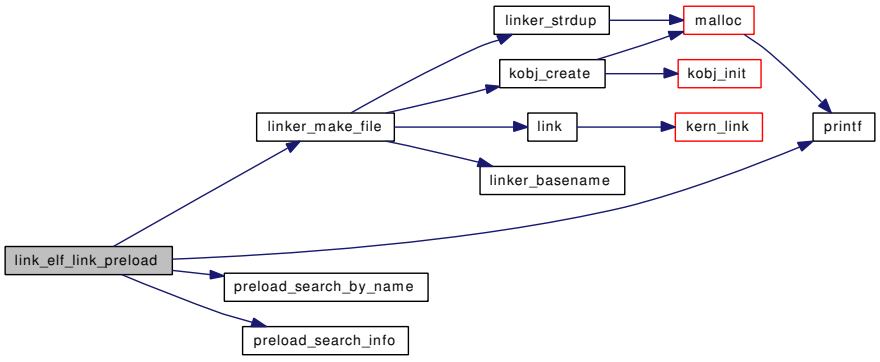


9.73.2.8 static int link_elf_link_preload (linker_class_t *cls*, const char *, linker_file_t *) [static]

Definition at line 171 of file link_elf_obj.c.

References __ELF_WORD_SIZE, elf_file::address, elf_file::e_shdr, elf_file::lf, linker_make_file(), elf_file::modptr, elf_file::nprogtab, elf_file::nrel, elf_file::nrela, preload_search_by_name(), preload_search_info(), elf_file::preloaded, and printf().

Here is the call graph for this function:

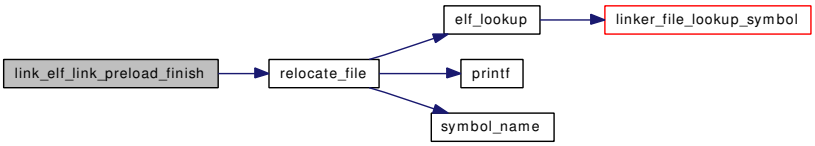


9.73.2.9 static int link_elf_link_preload_finish (linker_file_t) [static]

Definition at line 358 of file link_elf_obj.c.

References relocate_file().

Here is the call graph for this function:

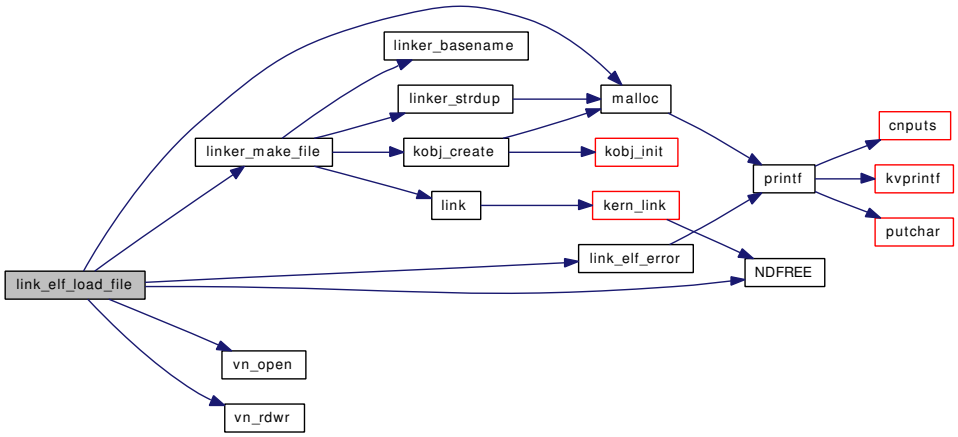


9.73.2.10 static int link_elf_load_file (linker_class_t, const char *, linker_file_t *) [static]

Definition at line 377 of file link_elf_obj.c.

References elf_file::e_shdr, link_elf_error(), linker_make_file(), malloc(), NDFREE(), elf_file::nprogtab, elf_file::nrel, elf_file::nrela, td, vn_open(), and vn_rdwr().

Here is the call graph for this function:



9.73.2.11 static int link_elf_lookup_set (linker_file_t, const char *, void ***, void ***, int *) [static]

Definition at line 1020 of file link_elf_obj.c.

References Elf_progent::addr, if(), Elf_progent::name, elf_file::nprogtab, elf_file::progtab, and Elf_-progent::size.

Here is the call graph for this function:

9.73.2.12 static int link_elf_lookup_symbol (linker_file_t, const char *, c_linker_sym_t *) [static]

Definition at line 947 of file link_elf_obj.c.

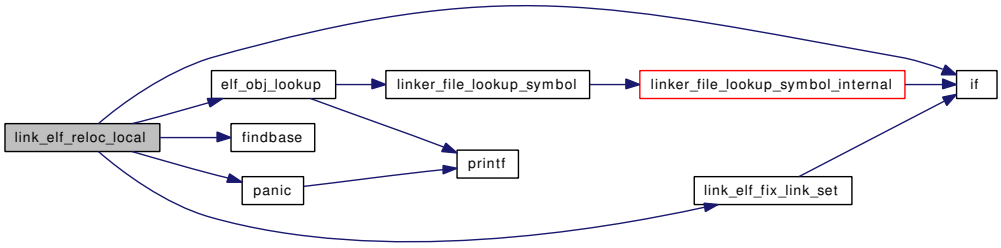
References elf_file::ddbstrtab, and elf_file::ddbsymtab.

9.73.2.13 static void link_elf_reloc_local (linker_file_t) [static]

Definition at line 1163 of file link_elf_obj.c.

References elf_file::ddbsymcnt, elf_file::ddbsymtab, elf_obj_lookup(), findbase(), if(), link_elf_fix_link_set(), Elf_relent::nrel, elf_file::nrel, Elf_relaent::nrela, elf_file::nrela, panic(), Elf_relent::rel, Elf_relaent::rela, elf_file::relatab, elf_file::reltab, Elf_relaent::sec, and Elf_relent::sec.

Here is the call graph for this function:



9.73.2.14 static int link_elf_search_symbol (linker_file_t, caddr_t *value*, c_linker_sym_t * *sym*, long * *diffp*) [static]

Definition at line 981 of file link_elf_obj.c.

References elf_file::ddbsymtab, and diff.

9.73.2.15 static int link_elf_symbol_values (linker_file_t, c_linker_sym_t, linker_symval_t *) [static]

Definition at line 965 of file link_elf_obj.c.

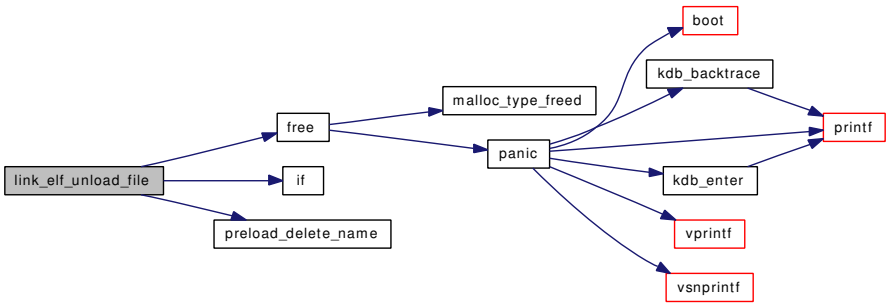
References elf_file::ddbstrtab, elf_file::ddbsymcnt, and elf_file::ddbsymtab.

9.73.2.16 static void link_elf_unload_file (linker_file_t) [static]

Definition at line 799 of file link_elf_obj.c.

References elf_file::address, free(), if(), elf_file::nrel, elf_file::nrela, elf_file::object, preload_delete_name(), elf_file::preloaded, elf_file::progtab, Elf_relent::rel, Elf_relaent::rela, elf_file::relatab, and elf_file::reltab.

Here is the call graph for this function:

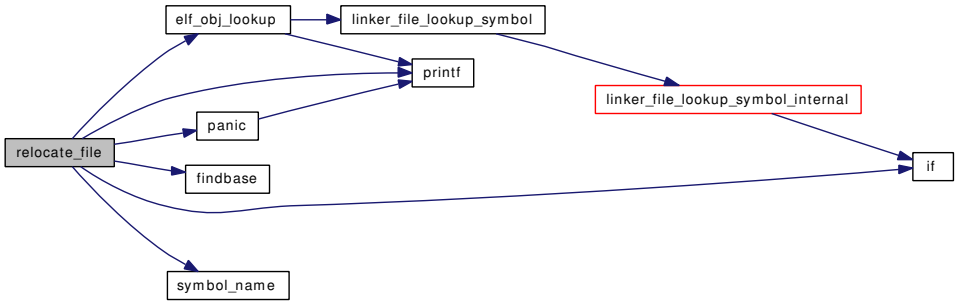


9.73.2.17 static int relocate_file (elf_file_t *ef*) [static]

Definition at line 876 of file link_elf_obj.c.

References elf_file::ddbsymcnt, elf_file::ddbsymtab, elf_obj_lookup(), findbase(), if(), elf_file::lf, Elf_relent::nrel, elf_file::nrel, panic(), printf(), Elf_relent::rel, elf_file::reltab, Elf_relent::sec, and symbol_name().

Here is the call graph for this function:



9.73.2.18 static const char* symbol_name (elf_file_t *ef*, Elf_Size *r_info*) [static]

Definition at line 849 of file link_elf_obj.c.

References elf_file::ddbstrtab, and elf_file::ddbsymtab.

9.73.2.19 SYSINIT (link_elf_obj, SI_SUB_KLD, SI_ORDER_SECOND, link_elf_init, 0)

9.73.3 Variable Documentation

9.73.3.1 struct linker_class link_elf_class [static]

Initial value:

```
{
        "elf32_obj",
```

```
        link_elf_methods, sizeof(struct elf_file)
}
```

Definition at line 144 of file link_elf_obj.c.

9.73.3.2 kobj_method_t link_elf_methods[ ] [static]

Initial value:

```
 {
        KOBJMETHOD(linker_lookup_symbol,        link_elf_lookup_symbol),
        KOBJMETHOD(linker_symbol_values,        link_elf_symbol_values),
        KOBJMETHOD(linker_search_symbol,        link_elf_search_symbol),
        KOBJMETHOD(linker_unload,               link_elf_unload_file),
        KOBJMETHOD(linker_load_file,            link_elf_load_file),
        KOBJMETHOD(linker_link_preload,         link_elf_link_preload),
        KOBJMETHOD(linker_link_preload_finish,  link_elf_link_preload_finish),
        KOBJMETHOD(linker_lookup_set,           link_elf_lookup_set),
        KOBJMETHOD(linker_each_function_name,   link_elf_each_function_name),
        { 0, 0 }
}
```

Definition at line 131 of file link_elf_obj.c.

9.74 /usr/src/sys/kern/linker_if.m File Reference

```
#include <sys/linker.h>
```

Include dependency graph for linker_if.m:

Variables

- INTERFACE linker

9.74.1 Variable Documentation

9.74.1.1 INTERFACE linker

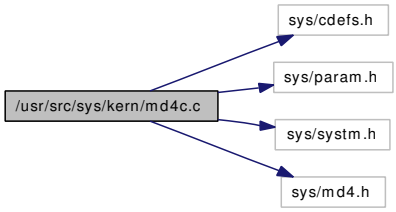
Definition at line 31 of file linker_if.m.

9.75 /usr/src/sys/kern/Make.tags.inc File Reference

9.76 /usr/src/sys/kern/md4c.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/md4.h>
```

Include dependency graph for md4c.c:



Defines

- #define PROTO_LIST(list) list
- #define S11 3
- #define S12 7
- #define S13 11
- #define S14 19
- #define S21 3
- #define S22 5
- #define S23 9
- #define S24 13
- #define S31 3
- #define S32 9
- #define S33 11
- #define S34 15
- #define F(x, y, z) (((x) & (y)) | ((~x) & (z)))
- #define G(x, y, z) (((x) & (y)) | ((x) & (z)) | ((y) & (z)))
- #define H(x, y, z) ((x) ^ (y) ^ (z))
- #define ROTATE_LEFT(x, n) (((x) << (n)) | ((x) >> (32-(n))))
- #define FF(a, b, c, d, x, s)
- #define GG(a, b, c, d, x, s)
- #define HH(a, b, c, d, x, s)

Typedefs

- typedef unsigned char * POINTER
- typedef u_int16_t UINT2
- typedef u_int32_t UINT4

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/md4c.c,v 1.3 2005/01/06 23:35:39 imp Exp $")
- static void MD4Transform PROTO_LIST ((UINT4[4], const unsigned char[64]))
- static void Encode PROTO_LIST ((unsigned char *, UINT4 *, unsigned int))
- static void Decode PROTO_LIST ((UINT4 *, const unsigned char *, unsigned int))
- void MD4Init (MD4_CTX *context)
- void MD4Update (MD4_CTX *context, const unsigned char *input, unsigned int inputLen)
- void MD4Pad (MD4_CTX *context)
- void MD4Final (digest, MD4_CTX *context)
- static void MD4Transform (state, block)
- static void Encode (unsigned char *output, UINT4 *input, unsigned int len)
- static void Decode (UINT4 *output, const unsigned char *input, unsigned int len)

Variables

- static unsigned char PADDING [64]

9.76.1 Define Documentation

9.76.1.1 #define F(x, y, z) (((x) & (y)) | ((~x) & (z)))

Definition at line 68 of file md4c.c.

9.76.1.2 #define FF(a, b, c, d, x, s)

Value:

```
{ \
    (a) += F ((b), (c), (d)) + (x); \
    (a) = ROTATE_LEFT ((a), (s)); \
  }
```

Definition at line 78 of file md4c.c.

Referenced by MD4Transform(), and MD5Transform().

9.76.1.3 #define G(x, y, z) (((x) & (y)) | ((x) & (z)) | ((y) & (z)))

Definition at line 69 of file md4c.c.

9.76.1.4 #define GG(a, b, c, d, x, s)

Value:

```
{ \
    (a) += G ((b), (c), (d)) + (x) + (UINT4)0x5a827999; \
    (a) = ROTATE_LEFT ((a), (s)); \
  }
```

Definition at line 82 of file md4c.c.

Referenced by MD4Transform(), and MD5Transform().

9.76.1.5 #define H(x, y, z) ((x) ^ (y) ^ (z))

Definition at line 70 of file md4c.c.

9.76.1.6 #define HH(a, b, c, d, x, s)

Value:

```
{ \
    (a) += H ((b), (c), (d)) + (x) + (UINT4)0x6ed9eba1; \
    (a) = ROTATE_LEFT ((a), (s)); \
  }
```

Definition at line 86 of file md4c.c.

Referenced by MD4Transform(), and MD5Transform().

9.76.1.7 #define PROTO_LIST(list) list

Definition at line 37 of file md4c.c.

9.76.1.8 #define ROTATE_LEFT(x, n) (((x) << (n)) | ((x) >> (32-(n))))

Definition at line 74 of file md4c.c.

9.76.1.9 #define S11 3

Definition at line 41 of file md4c.c.

Referenced by MD4Transform(), and MD5Transform().

9.76.1.10 #define S12 7

Definition at line 42 of file md4c.c.

Referenced by MD4Transform(), and MD5Transform().

9.76.1.11 #define S13 11

Definition at line 43 of file md4c.c.

Referenced by MD4Transform(), and MD5Transform().

9.76.1.12 #define S14 19

Definition at line 44 of file md4c.c.

Referenced by MD4Transform(), and MD5Transform().

9.76.1.13 #define S21 3

Definition at line 45 of file md4c.c.

Referenced by MD4Transform(), and MD5Transform().

9.76.1.14 #define S22 5

Definition at line 46 of file md4c.c.

Referenced by MD4Transform(), and MD5Transform().

9.76.1.15 #define S23 9

Definition at line 47 of file md4c.c.

Referenced by MD4Transform(), and MD5Transform().

9.76.1.16 #define S24 13

Definition at line 48 of file md4c.c.

Referenced by MD4Transform(), and MD5Transform().

9.76.1.17 #define S31 3

Definition at line 49 of file md4c.c.

Referenced by MD4Transform(), and MD5Transform().

9.76.1.18 #define S32 9

Definition at line 50 of file md4c.c.

Referenced by MD4Transform(), and MD5Transform().

9.76.1.19 #define S33 11

Definition at line 51 of file md4c.c.

Referenced by MD4Transform(), and MD5Transform().

9.76.1.20 #define S34 15

Definition at line 52 of file md4c.c.

Referenced by MD4Transform(), and MD5Transform().

9.76.2 Typedef Documentation

9.76.2.1 typedef unsigned char* POINTER

Definition at line 33 of file md4c.c.

9.76.2.2 typedef u_int16_t UINT2

Definition at line 34 of file md4c.c.

9.76.2.3 typedef u_int32_t UINT4

Definition at line 35 of file md4c.c.

9.76.3 Function Documentation

9.76.3.1 __FBSDID ("$FreeBSD: src/sys/kern/md4c. *c*, v 1.3 2005/01/06 23:35:39 imp Exp $")

9.76.3.2 static void Decode (UINT4 * *output*, const unsigned char * *input*, unsigned int *len*) [static]

Definition at line 277 of file md4c.c.

9.76.3.3 static void Encode (unsigned char * *output*, UINT4 * *input*, unsigned int *len*) [static]

Definition at line 259 of file md4c.c.

9.76.3.4 void MD4Final (digest, MD4_CTX * *context*)

Definition at line 167 of file md4c.c.

References Encode, MD4Pad(), and POINTER.

Here is the call graph for this function:

9.76.3.5 void MD4Init (MD4_CTX * *context*)

Definition at line 93 of file md4c.c.

9.76.3.6 void MD4Pad (MD4_CTX * *context*)

Definition at line 145 of file md4c.c.

References Encode, and MD4Update().

Referenced by MD4Final().

Here is the call graph for this function:

9.76.3.7 static void MD4Transform (state, block) [static]

Definition at line 184 of file md4c.c.

References Decode, FF, GG, HH, POINTER, S11, S12, S13, S14, S21, S22, S23, S24, S31, S32, S33, and S34.

Referenced by MD4Update().

9.76.3.8 void MD4Update (MD4_CTX * *context*, const unsigned char * *input*, unsigned int *inputLen*)

Definition at line 110 of file md4c.c.

References MD4Transform().

Referenced by MD4Pad().

Here is the call graph for this function:

9.76.3.9 static void Decode PROTO_LIST ((UINT4 *, const unsigned char *, unsigned int)) [static]

9.76.3.10 static void Encode PROTO_LIST ((unsigned char *, UINT4 *, unsigned int)) [static]

9.76.3.11 static void MD4Transform PROTO_LIST ((UINT4[4], const unsigned char[64])) [static]

9.76.4 Variable Documentation

9.76.4.1 unsigned char PADDING[64] [static]

Initial value:

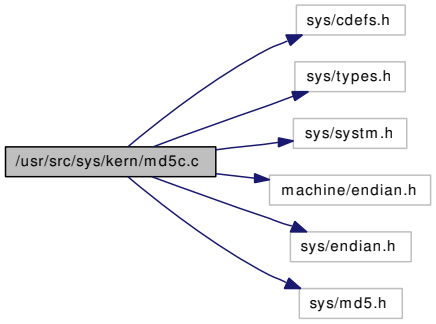
```
{
  0x80, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
}
```

Definition at line 60 of file md4c.c.

9.77 /usr/src/sys/kern/md5c.c File Reference

```
#include <sys/cdefs.h>
#include <sys/types.h>
#include <sys/systm.h>
#include <machine/endian.h>
#include <sys/endian.h>
#include <sys/md5.h>
```

Include dependency graph for md5c.c:



Defines

- #define Encode memcpy
- #define Decode memcpy
- #define F(x, y, z) (((x) & (y)) | ((~x) & (z)))
- #define G(x, y, z) (((x) & (z)) | ((y) & (~z)))
- #define H(x, y, z) ((x) ^ (y) ^ (z))
- #define I(x, y, z) ((y) ^ ((x) | (~z)))
- #define ROTATE_LEFT(x, n) (((x) << (n)) | ((x) >> (32-(n))))
- #define FF(a, b, c, d, x, s, ac)
- #define GG(a, b, c, d, x, s, ac)
- #define HH(a, b, c, d, x, s, ac)
- #define II(a, b, c, d, x, s, ac)
- #define S11 7
- #define S12 12
- #define S13 17
- #define S14 22
- #define S21 5
- #define S22 9
- #define S23 14
- #define S24 20
- #define S31 4
- #define S32 11
- #define S33 16
- #define S34 23
- #define S41 6

- #define S42 10
- #define S43 15
- #define S44 21

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/md5c.c,v 1.27 2006/03/30 18:45:50 pjd Exp $")
- static void MD5Transform (u_int32_t[4], const unsigned char[64])
- void MD5Init (MD5_CTX *context)
- void MD5Update (MD5_CTX *context, const void *in, unsigned int inputLen)
- static void MD5Pad (MD5_CTX *context)
- void MD5Final (digest, MD5_CTX *context)
- static void MD5Transform (state, block)

Variables

- static unsigned char PADDING [64]

9.77.1 Define Documentation

9.77.1.1 #define Decode memcpy

Definition at line 51 of file md5c.c.

Referenced by MD4Transform(), and MD5Transform().

9.77.1.2 #define Encode memcpy

Definition at line 50 of file md5c.c.

Referenced by MD4Final(), MD4Pad(), MD5Final(), and MD5Pad().

9.77.1.3 #define F(x, y, z) (((x) & (y)) | ((~x) & (z)))

Definition at line 98 of file md5c.c.

9.77.1.4 #define FF(a, b, c, d, x, s, ac)

Value:

```
{ \
        (a) += F ((b), (c), (d)) + (x) + (u_int32_t)(ac); \
        (a) = ROTATE_LEFT ((a), (s)); \
        (a) += (b); \
        }
```

Definition at line 110 of file md5c.c.

9.77.1.5 #define G(x, y, z) (((x) & (z)) | ((y) & (∼z)))

Definition at line 99 of file md5c.c.

9.77.1.6 #define GG(a, b, c, d, x, s, ac)

Value:

```
{ \
        (a) += G ((b), (c), (d)) + (x) + (u_int32_t)(ac); \
        (a) = ROTATE_LEFT ((a), (s)); \
        (a) += (b); \
        }
```

Definition at line 115 of file md5c.c.

9.77.1.7 #define H(x, y, z) ((x) ^ (y) ^ (z))

Definition at line 100 of file md5c.c.

9.77.1.8 #define HH(a, b, c, d, x, s, ac)

Value:

```
{ \
        (a) += H ((b), (c), (d)) + (x) + (u_int32_t)(ac); \
        (a) = ROTATE_LEFT ((a), (s)); \
        (a) += (b); \
        }
```

Definition at line 120 of file md5c.c.

9.77.1.9 #define I(x, y, z) ((y) ^ ((x) | (∼z)))

Definition at line 101 of file md5c.c.

9.77.1.10 #define II(a, b, c, d, x, s, ac)

Value:

```
{ \
        (a) += I ((b), (c), (d)) + (x) + (u_int32_t)(ac); \
        (a) = ROTATE_LEFT ((a), (s)); \
        (a) += (b); \
        }
```

Definition at line 125 of file md5c.c.

Referenced by MD5Transform().

9.77.1.11 #define ROTATE_LEFT(x, n) (((x) << (n)) | ((x) >> (32-(n))))

Definition at line 104 of file md5c.c.

9.77.1.12 #define S11 7

9.77.1.13 #define S12 12

9.77.1.14 #define S13 17

9.77.1.15 #define S14 22

9.77.1.16 #define S21 5

9.77.1.17 #define S22 9

9.77.1.18 #define S23 14

9.77.1.19 #define S24 20

9.77.1.20 #define S31 4

9.77.1.21 #define S32 11

9.77.1.22 #define S33 16

9.77.1.23 #define S34 23

9.77.1.24 #define S41 6

Referenced by MD5Transform().

9.77.1.25 #define S42 10

Referenced by MD5Transform().

9.77.1.26 #define S43 15

Referenced by MD5Transform().

9.77.1.27 #define S44 21

Referenced by MD5Transform().

9.77.2 Function Documentation

9.77.2.1 __FBSDID ("$FreeBSD: src/sys/kern/md5c. *c*, v 1.27 2006/03/30 18:45:50 pjd Exp $")

9.77.2.2 void MD5Final (digest, MD5_CTX * *context*)

Definition at line 220 of file md5c.c.

References Encode, and MD5Pad().

Here is the call graph for this function:

9.77.2.3 void MD5Init (MD5_CTX * *context*)

Definition at line 134 of file md5c.c.

9.77.2.4 static void MD5Pad (MD5_CTX * *context*) [static]

Definition at line 197 of file md5c.c.

References Encode, and MD5Update().

Referenced by MD5Final().

Here is the call graph for this function:

9.77.2.5 static void MD5Transform (state, block) [static]

Definition at line 237 of file md5c.c.

References Decode, FF, GG, HH, II, S11, S12, S13, S14, S21, S22, S23, S24, S31, S32, S33, S34, S41, S42, S43, and S44.

9.77.2.6 static void MD5Transform (u_int32_t[4], const unsigned *char*[64]) [static]

Referenced by MD5Update().

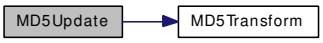
9.77.2.7 void MD5Update (MD5_CTX * *context*, const void * *in*, unsigned int *inputLen*)

Definition at line 154 of file md5c.c.

References MD5Transform().

Referenced by MD5Pad().

Here is the call graph for this function:



9.77.3 Variable Documentation

9.77.3.1 unsigned char PADDING[64] [static]

Initial value:

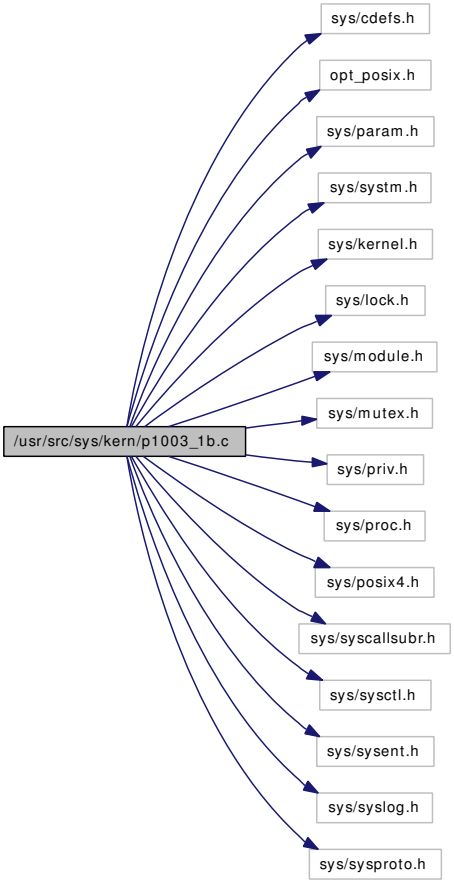
```
 {
  0x80, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
}
```

Definition at line 91 of file md5c.c.

9.78 /usr/src/sys/kern/p1003_1b.c File Reference

```
#include <sys/cdefs.h>
#include "opt_posix.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/module.h>
#include <sys/mutex.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/posix4.h>
#include <sys/syscallsubr.h>
#include <sys/sysctl.h>
#include <sys/sysent.h>
#include <sys/syslog.h>
#include <sys/sysproto.h>
```

Include dependency graph for p1003_1b.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/p1003_1b.c,v 1.33 2006/11/11 16:19:11 trhodes Exp $")
- MALLOC_DEFINE (M_P31B,"p1003.1b","Posix 1003.1B")
- int syscall_not_present (struct thread *td, const char *s, struct nosys_args *uap)
- static int sched_attach (void)
- static void p31binit (void *notused)
- SYSINIT (p31b, SI_SUB_P1003_1B, SI_ORDER_FIRST, p31binit, NULL)

9.78.1 Function Documentation

9.78.1.1 __FBSDID ("$FreeBSD: src/sys/kern/p1003_1b. *c*, v 1.33 2006/11/11 16:19:11 trhodes Exp $")

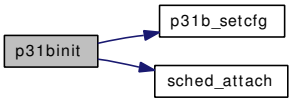
9.78.1.2 MALLOC_DEFINE (M_P31B, "p1003.1b", "Posix 1003.1B")

9.78.1.3 static void p31binit (void * *notused*) [static]

Definition at line 338 of file p1003_1b.c.

References p31b_setcfg(), and sched_attach().

Here is the call graph for this function:



9.78.1.4 static int sched_attach (void) [static]

Definition at line 80 of file p1003_1b.c.

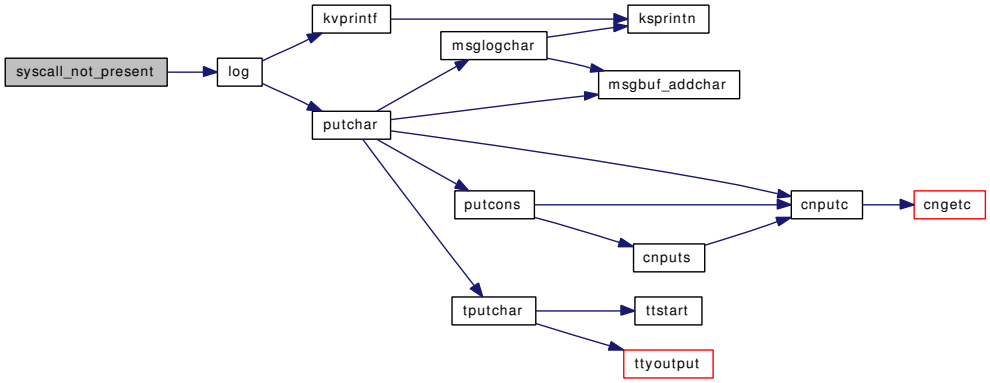
Referenced by p31binit().

9.78.1.5 int syscall_not_present (struct thread * *td*, const char * *s*, struct nosys_args * *uap*)

Definition at line 63 of file p1003_1b.c.

References log().

Here is the call graph for this function:

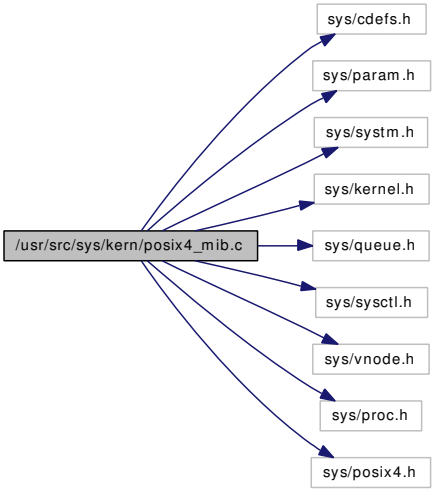


9.78.1.6 SYSINIT (p31b, SI_SUB_P1003_1B, SI_ORDER_FIRST, p31binit, NULL)

9.79 /usr/src/sys/kern/posix4_mib.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/queue.h>
#include <sys/sysctl.h>
#include <sys/vnode.h>
#include <sys/proc.h>
#include <sys/posix4.h>
```

Include dependency graph for posix4_mib.c:



Defines

- #define P1B_SYSCTL(num, name)
- #define P31B_VALID(num) ((num) >= 1 && (num) < CTL_P1003_1B_MAXID)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/posix4_mib.c,v 1.12 2006/11/12 03:34:03 trhodes Exp $")
- SYSCTL_DECL (_p1003_1b)
- SYSCTL_INT (_p1003_1b, CTL_P1003_1B_ASYNCHRONOUS_IO, asynchronous_io, CTLFLAG_RD,&async_io_version, 0,"")
- P1B_SYSCTL (CTL_P1003_1B_MAPPED_FILES, mapped_files)
- P1B_SYSCTL (CTL_P1003_1B_MEMLOCK, memlock)
- P1B_SYSCTL (CTL_P1003_1B_MEMLOCK_RANGE, memlock_range)
- P1B_SYSCTL (CTL_P1003_1B_MEMORY_PROTECTION, memory_protection)
- P1B_SYSCTL (CTL_P1003_1B_MESSAGE_PASSING, message_passing)

- P1B_SYSCTL (CTL_P1003_1B_PRIORITIZED_IO, prioritized_io)
- P1B_SYSCTL (CTL_P1003_1B_PRIORITY_SCHEDULING, priority_scheduling)
- P1B_SYSCTL (CTL_P1003_1B_REALTIME_SIGNALS, realtime_signals)
- P1B_SYSCTL (CTL_P1003_1B_SEMAPHORES, semaphores)
- P1B_SYSCTL (CTL_P1003_1B_FSYNC, fsync)
- P1B_SYSCTL (CTL_P1003_1B_SHARED_MEMORY_OBJECTS, shared_memory_objects)
- P1B_SYSCTL (CTL_P1003_1B_SYNCHRONIZED_IO, synchronized_io)
- P1B_SYSCTL (CTL_P1003_1B_TIMERS, timers)
- P1B_SYSCTL (CTL_P1003_1B_AIO_LISTIO_MAX, aio_listio_max)
- P1B_SYSCTL (CTL_P1003_1B_AIO_MAX, aio_max)
- P1B_SYSCTL (CTL_P1003_1B_AIO_PRIO_DELTA_MAX, aio_prio_delta_max)
- P1B_SYSCTL (CTL_P1003_1B_DELAYTIMER_MAX, delaytimer_max)
- P1B_SYSCTL (CTL_P1003_1B_MQ_OPEN_MAX, mq_open_max)
- P1B_SYSCTL (CTL_P1003_1B_PAGESIZE, pagesize)
- P1B_SYSCTL (CTL_P1003_1B_RTSIG_MAX, rtsig_max)
- P1B_SYSCTL (CTL_P1003_1B_SEM_NSEMS_MAX, sem_nsems_max)
- P1B_SYSCTL (CTL_P1003_1B_SEM_VALUE_MAX, sem_value_max)
- P1B_SYSCTL (CTL_P1003_1B_SIGQUEUE_MAX, sigqueue_max)
- P1B_SYSCTL (CTL_P1003_1B_TIMER_MAX, timer_max)
- void p31b_setcfg (int num, int value)
- int p31b_getcfg (int num)
- int p31b_iscfg (int num)
- static void p31b_set_standard (void *dummy)
- SYSINIT (p31b_set_standard, SI_SUB_P1003_1B, SI_ORDER_ANY, p31b_set_standard, 0)

Variables

- static int facility [CTL_P1003_1B_MAXID-1]
- static int facility_initialized [CTL_P1003_1B_MAXID-1]

9.79.1 Define Documentation

9.79.1.1 #define P1B_SYSCTL(num, name)

Value:

```
SYSCTL_INT(_p1003_1b, num, \
        name, CTLFLAG_RD, facility + num - 1, 0, "");
```

Definition at line 57 of file posix4_mib.c.

9.79.1.2 #define P31B_VALID(num) ((num) >= 1 && (num) < CTL_P1003_1B_MAXID)

Definition at line 99 of file posix4_mib.c.

Referenced by p31b_getcfg(), p31b_iscfg(), and p31b_setcfg().

9.79.2 Function Documentation

9.79.2.1 __FBSDID ("$FreeBSD: src/sys/kern/posix4_mib. *c*, v 1.12 2006/11/12 03:34:03 trhodes Exp $")

9.79.2.2 P1B_SYSCTL (CTL_P1003_1B_TIMER_MAX, timer_max)

9.79.2.3 P1B_SYSCTL (CTL_P1003_1B_SIGQUEUE_MAX, sigqueue_max)

9.79.2.4 P1B_SYSCTL (CTL_P1003_1B_SEM_VALUE_MAX, sem_value_max)

9.79.2.5 P1B_SYSCTL (CTL_P1003_1B_SEM_NSEMS_MAX, sem_nsems_max)

9.79.2.6 P1B_SYSCTL (CTL_P1003_1B_RTSIG_MAX, rtsig_max)

9.79.2.7 P1B_SYSCTL (CTL_P1003_1B_PAGESIZE, pagesize)

9.79.2.8 P1B_SYSCTL (CTL_P1003_1B_MQ_OPEN_MAX, mq_open_max)

9.79.2.9 P1B_SYSCTL (CTL_P1003_1B_DELAYTIMER_MAX, delaytimer_max)

9.79.2.10 P1B_SYSCTL (CTL_P1003_1B_AIO_PRIO_DELTA_MAX, aio_prio_delta_max)

9.79.2.11 P1B_SYSCTL (CTL_P1003_1B_AIO_MAX, aio_max)

9.79.2.12 P1B_SYSCTL (CTL_P1003_1B_AIO_LISTIO_MAX, aio_listio_max)

9.79.2.13 P1B_SYSCTL (CTL_P1003_1B_TIMERS, timers)

9.79.2.14 P1B_SYSCTL (CTL_P1003_1B_SYNCHRONIZED_IO, synchronized_io)

9.79.2.15 P1B_SYSCTL (CTL_P1003_1B_SHARED_MEMORY_OBJECTS, shared_memory_objects)

9.79.2.16 P1B_SYSCTL (CTL_P1003_1B_FSYNC, fsync)

9.79.2.17 P1B_SYSCTL (CTL_P1003_1B_SEMAPHORES, semaphores)

9.79.2.18 P1B_SYSCTL (CTL_P1003_1B_REALTIME_SIGNALS, realtime_signals)

9.79.2.19 P1B_SYSCTL (CTL_P1003_1B_PRIORITY_SCHEDULING, priority_scheduling)

9.79.2.20 P1B_SYSCTL (CTL_P1003_1B_PRIORITIZED_IO, prioritized_io)

9.79.2.21 P1B_SYSCTL (CTL_P1003_1B_MESSAGE_PASSING, message_passing)

9.79.2.22 P1B_SYSCTL (CTL_P1003_1B_MEMORY_PROTECTION, memory_protection)

9.79.2.23 P1B_SYSCTL (CTL_P1003_1B_MEMLOCK_RANGE, memlock_range)

9.79.2.24 P1B_SYSCTL (CTL_P1003_1B_MEMLOCK, memlock)

9.79.2.25 P1B_SYSCTL (CTL_P1003_1B_MAPPED_FILES, mapped_files)

9.79.2.26 int p31b_getcfg (int *num*)

Definition at line 114 of file posix4_mib.c.

References facility, and P31B_VALID.

Referenced by sem_create().

9.79.2.27 int p31b_iscfg (int *num*)

Definition at line 123 of file posix4_mib.c.

References facility_initialized, and P31B_VALID.

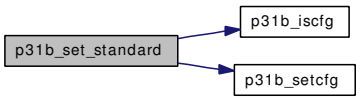
Referenced by p31b_set_standard().

9.79.2.28 static void p31b_set_standard (void * *dummy*) [static]

Definition at line 135 of file posix4_mib.c.

References p31b_iscfg(), and p31b_setcfg().

Here is the call graph for this function:



9.79.2.29 void p31b_setcfg (int *num*, int *value*)

Definition at line 104 of file posix4_mib.c.

References facility, facility_initialized, and P31B_VALID.

Referenced by aio_onceonly(), aio_unload(), itimer_start(), mqfs_init(), p31b_set_standard(), p31binit(), sem_modload(), and sigqueue_start().

9.79.2.30 SYSCTL_DECL (_p1003_1b)

9.79.2.31 SYSCTL_INT (_p1003_1b, CTL_P1003_1B_ASYNCHRONOUS_IO, asynchronous_io, CTLFLAG_RD, & *async_io_version*, 0, "")

9.79.2.32 SYSINIT (p31b_set_standard, SI_SUB_P1003_1B, SI_ORDER_ANY, p31b_set_standard, 0)

9.79.3 Variable Documentation

9.79.3.1 int facility[CTL_P1003_1B_MAXID-1] [static]

Definition at line 45 of file posix4_mib.c.

Referenced by p31b_getcfg(), and p31b_setcfg().

9.79.3.2 int facility_initialized[CTL_P1003_1B_MAXID-1] [static]

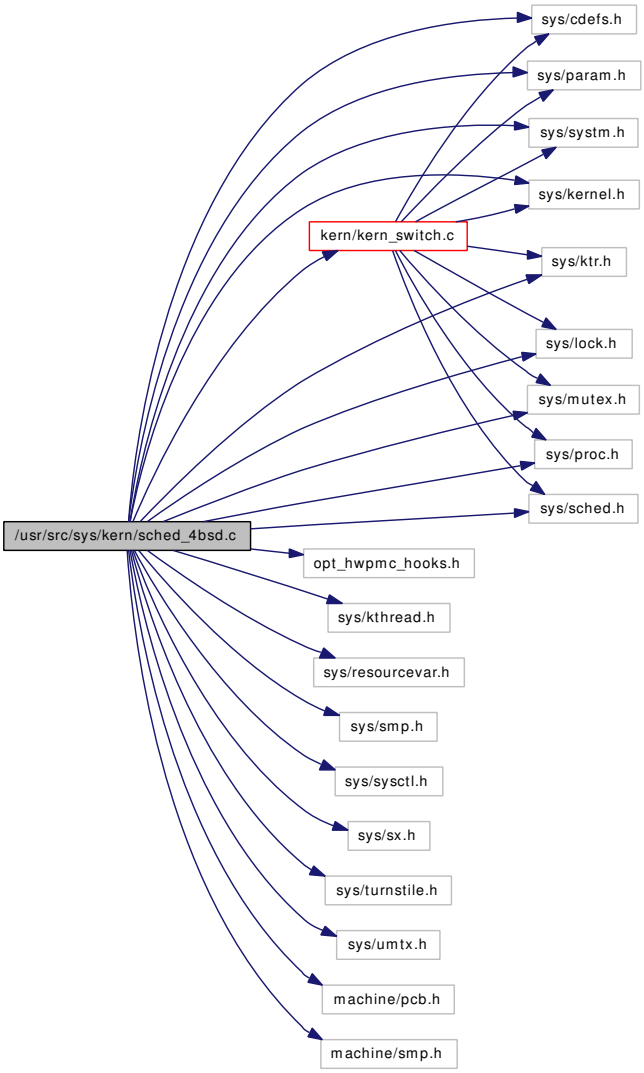
Definition at line 46 of file posix4_mib.c.

Referenced by p31b_iscfg(), and p31b_setcfg().

9.80 /usr/src/sys/kern/sched_4bsd.c File Reference

```
#include <sys/cdefs.h>
#include "opt_hwpmc_hooks.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/ktr.h>
#include <sys/lock.h>
#include <sys/kthread.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/resourcevar.h>
#include <sys/sched.h>
#include <sys/smp.h>
#include <sys/sysctl.h>
#include <sys/sx.h>
#include <sys/turnstile.h>
#include <sys/umtx.h>
#include <machine/pcb.h>
#include <machine/smp.h>
#include "kern/kern_switch.c"
```

Include dependency graph for sched_4bsd.c:



Data Structures

- struct td_sched

Defines

- #define ESTCPULIM(e)
- #define INVERSE_ESTCPU_WEIGHT 8
- #define NICE_WEIGHT 1
- #define TDF_DIDRUN TDF_SCHED0
- #define TDF_EXIT TDF_SCHED1
- #define TDF_BOUND TDF_SCHED2
- #define ts_flags ts_thread → td_flags
- #define TSF_DIDRUN TDF_DIDRUN
- #define TSF_EXIT TDF_EXIT
- #define TSF_BOUND TDF_BOUND

- #define SKE_RUNQ_PCPU(ts) ((ts) → ts_runq != 0 && (ts) → ts_runq != &runq)
- #define SCHED_QUANTUM (hz / 10)
- #define loadfactor(loadav) (2 ∗ (loadav))
- #define decay_cpu(loadfac, cpu) (((loadfac) ∗ (cpu)) / ((loadfac) + FSCALE))
- #define CCPU_SHIFT 11
- #define KERN_SWITCH_INCLUDE 1

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/sched_4bsd.c,v 1.96 2007/02/02 05:14:21 julian Exp $")
- static void setup_runqs (void)
- static void roundrobin (void ∗arg)
- static void schedcpu (void)
- static void schedcpu_thread (void)
- static void sched_priority (struct thread ∗td, u_char prio)
- static void sched_setup (void ∗dummy)
- static void maybe_resched (struct thread ∗td)
- static void updatepri (struct thread ∗td)
- static void resetpriority (struct thread ∗td)
- static void resetpriority_thread (struct thread ∗td)
- static int sysctl_kern_quantum (SYSCTL_HANDLER_ARGS)
- SYSCTL_NODE (_kern, OID_AUTO, sched, CTLFLAG_RD, 0,"Scheduler")
- SYSCTL_STRING (_kern_sched, OID_AUTO, name, CTLFLAG_RD,"4BSD", 0,"Scheduler name")
- SYSCTL_PROC (_kern_sched, OID_AUTO, quantum, CTLTYPE_INT|CTLFLAG_RW, 0, sizeof sched_quantum, sysctl_kern_quantum,"I","Roundrobin scheduling quantum in microseconds")
- static __inline void sched_load_add (void)
- static __inline void sched_load_rem (void)
- SYSCTL_INT (_kern, OID_AUTO, ccpu, CTLFLAG_RD,&ccpu, 0,"")
- void schedinit (void)
- int sched_runnable (void)
- int sched_rr_interval (void)
- void sched_clock (struct thread ∗td)
- void sched_exit (struct proc ∗p, struct thread ∗td)
- void sched_exit_thread (struct thread ∗td, struct thread ∗child)
- void sched_fork (struct thread ∗td, struct thread ∗childtd)
- void sched_fork_thread (struct thread ∗td, struct thread ∗childtd)
- void sched_nice (struct proc ∗p, int nice)
- void sched_class (struct thread ∗td, int class)
- void sched_lend_prio (struct thread ∗td, u_char prio)
- void sched_unlend_prio (struct thread ∗td, u_char prio)
- void sched_prio (struct thread ∗td, u_char prio)
- void sched_user_prio (struct thread ∗td, u_char prio)
- void sched_lend_user_prio (struct thread ∗td, u_char prio)
- void sched_unlend_user_prio (struct thread ∗td, u_char prio)
- void sched_sleep (struct thread ∗td)
- void sched_switch (struct thread ∗td, struct thread ∗newtd, int flags)
- void sched_wakeup (struct thread ∗td)
- void sched_add (struct thread ∗td, int flags)
- void sched_rem (struct thread ∗td)

- thread * sched_choose (void)
- void sched_userret (struct thread *td)
- void sched_bind (struct thread *td, int cpu)
- void sched_unbind (struct thread *td)
- int sched_is_bound (struct thread *td)
- void sched_relinquish (struct thread *td)
- int sched_load (void)
- int sched_sizeof_proc (void)
- int sched_sizeof_thread (void)
- fixpt_t sched_pctcpu (struct thread *td)
- void sched_tick (void)
- void sched_idletd (void *dummy)

Variables

- static struct td_sched td_sched0
- static int sched_tdcnt
- static int sched_quantum
- static struct callout roundrobin_callout
- static struct kproc_desc sched_kp
- static struct runq runq
- static fixpt_t ccpu = 0.95122942450071400909 * FSCALE

9.80.1 Define Documentation

9.80.1.1 #define CCPU_SHIFT 11

Definition at line 361 of file sched_4bsd.c.

Referenced by schedcpu().

9.80.1.2 #define decay_cpu(loadfac, cpu) (((loadfac) * (cpu)) / ((loadfac) + FSCALE))

Definition at line 343 of file sched_4bsd.c.

Referenced by schedcpu(), and updatepri().

9.80.1.3 #define ESTCPULIM(e)

Value:

```
min((e), INVERSE_ESTCPU_WEIGHT * (NICE_WEIGHT * (PRIO_MAX - PRIO_MIN) - \
    RQ_PPQ) + INVERSE_ESTCPU_WEIGHT - 1)
```

Definition at line 66 of file sched_4bsd.c.

Referenced by sched_clock(), and sched_exit_thread().

9.80.1.4 #define INVERSE_ESTCPU_WEIGHT 8

Definition at line 72 of file sched_4bsd.c.

Referenced by resetpriority(), and sched_clock().

9.80.1.5 #define KERN_SWITCH_INCLUDE 1

Definition at line 1367 of file sched_4bsd.c.

9.80.1.6 #define loadfactor(loadav) (2 ∗ (loadav))

Definition at line 342 of file sched_4bsd.c.

Referenced by schedcpu(), and updatepri().

9.80.1.7 #define NICE_WEIGHT 1

Definition at line 74 of file sched_4bsd.c.

Referenced by resetpriority().

9.80.1.8 #define SCHED_QUANTUM (hz / 10)

Definition at line 107 of file sched_4bsd.c.

Referenced by sched_rr_interval(), and sched_setup().

9.80.1.9 #define SKE_RUNQ_PCPU(ts) ((ts) → ts_runq != 0 && (ts) → ts_runq != &runq)

Definition at line 100 of file sched_4bsd.c.

Referenced by sched_add().

9.80.1.10 #define TDF_BOUND TDF_SCHED2

Definition at line 93 of file sched_4bsd.c.

9.80.1.11 #define TDF_DIDRUN TDF_SCHED0

Definition at line 91 of file sched_4bsd.c.

9.80.1.12 #define TDF_EXIT TDF_SCHED1

Definition at line 92 of file sched_4bsd.c.

9.80.1.13 #define ts_flags ts_thread → td_flags

Definition at line 95 of file sched_4bsd.c.

Referenced by sched_add().

9.80.1.14 #define TSF_BOUND TDF_BOUND

Definition at line 98 of file sched_4bsd.c.

Referenced by sched_add(), sched_bind(), sched_is_bound(), sched_tick(), and sched_unbind().

9.80.1.15 #define TSF_DIDRUN TDF_DIDRUN

Definition at line 96 of file sched_4bsd.c.

Referenced by sched_choose(), sched_switch(), and schedcpu().

9.80.1.16 #define TSF_EXIT TDF_EXIT

Definition at line 97 of file sched_4bsd.c.

9.80.2 Function Documentation

9.80.2.1 __FBSDID ("$FreeBSD: src/sys/kern/sched_4bsd. *c*, v 1.96 2007/02/02 05:14:21 julian Exp $")

9.80.2.2 static void maybe_resched (struct thread * *td*) [static]

Definition at line 248 of file sched_4bsd.c.

References sched_lock.

Referenced by resetpriority_thread(), and sched_add().

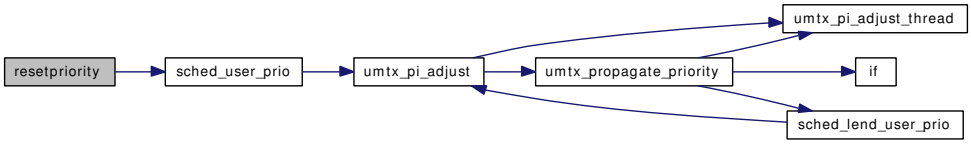
9.80.2.3 static void resetpriority (struct thread * *td*) [static]

Definition at line 513 of file sched_4bsd.c.

References INVERSE_ESTCPU_WEIGHT, NICE_WEIGHT, and sched_user_prio().

Referenced by sched_clock(), sched_nice(), sched_wakeup(), and schedcpu().

Here is the call graph for this function:



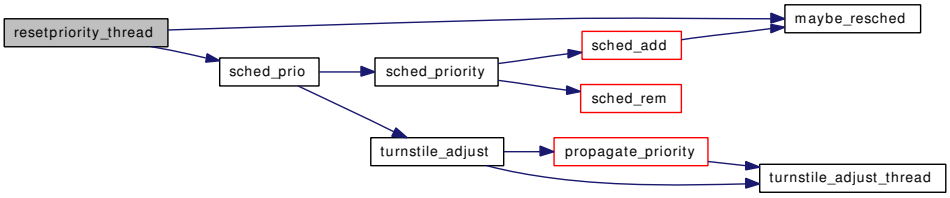
9.80.2.4 static void resetpriority_thread (struct thread * *td*) [static]

Definition at line 531 of file sched_4bsd.c.

References maybe_resched(), and sched_prio().

Referenced by sched_clock(), sched_nice(), and schedcpu().

Here is the call graph for this function:



9.80.2.5 static void roundrobin (void * *arg*) [static]

Definition at line 265 of file sched_4bsd.c.

References callout_reset(), roundrobin_callout, sched_lock, and sched_quantum.

Referenced by sched_setup().

Here is the call graph for this function:

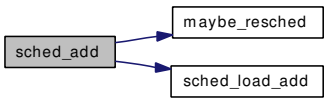
9.80.2.6 void sched_add (struct thread * *td*, int *flags*)

Definition at line 1074 of file sched_4bsd.c.

References maybe_resched(), sched_load_add(), sched_lock, SKE_RUNQ_PCPU, ts_flags, and TSF_BOUND.

Referenced by create_thread(), fork1(), intr_event_schedule_thread(), ithread_destroy(), kick_init(), kse_create(), kthread_create(), sched_priority(), sched_switch(), sched_thread_priority(), sched_wakeup(), taskqueue_start_threads(), and turnstile_unpend().

Here is the call graph for this function:



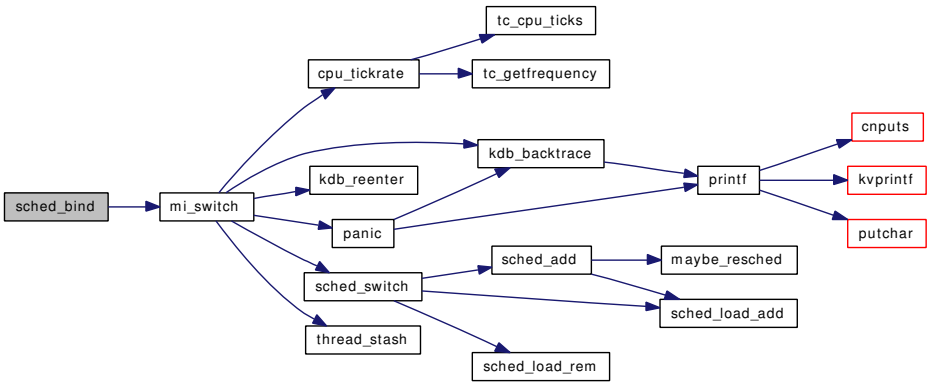
9.80.2.7 void sched_bind (struct thread * *td*, int *cpu*)

Definition at line 1268 of file sched_4bsd.c.

References mi_switch(), sched_lock, and TSF_BOUND.

Referenced by cf_set_method().

Here is the call graph for this function:



9.80.2.8 struct thread* sched_choose (void)

Definition at line 1206 of file sched_4bsd.c.

References td_sched::ts_thread, and TSF_DIDRUN.

9.80.2.9 void sched_class (struct thread * *td*, int *class*)

Definition at line 685 of file sched_4bsd.c.

References sched_lock.

Referenced by idle_setup(), ithread_create(), and rtp_to_pri().

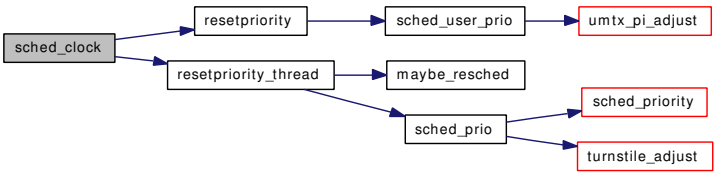
9.80.2.10 void sched_clock (struct thread * *td*)

Definition at line 615 of file sched_4bsd.c.

References ESTCPULIM, INVERSE_ESTCPU_WEIGHT, resetpriority(), resetpriority_thread(), and sched_lock.

Referenced by statclock().

Here is the call graph for this function:



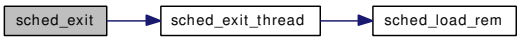
9.80.2.11 void sched_exit (struct proc * *p*, struct thread * *td*)

Definition at line 634 of file sched_4bsd.c.

References sched_exit_thread().

Referenced by exit1().

Here is the call graph for this function:



9.80.2.12 void sched_exit_thread (struct thread * *td*, struct thread * *child*)

Definition at line 644 of file sched_4bsd.c.

References ESTCPULIM, and sched_load_rem().

Referenced by sched_exit(), and thread_exit().

Here is the call graph for this function:

9.80.2.13 void sched_fork (struct thread * *td*, struct thread * *childtd*)

Definition at line 658 of file sched_4bsd.c.

References sched_fork_thread().

Referenced by fork1().

Here is the call graph for this function:

9.80.2.14 void sched_fork_thread (struct thread * *td*, struct thread * *childtd*)

Definition at line 664 of file sched_4bsd.c.

Referenced by create_thread(), and sched_fork().

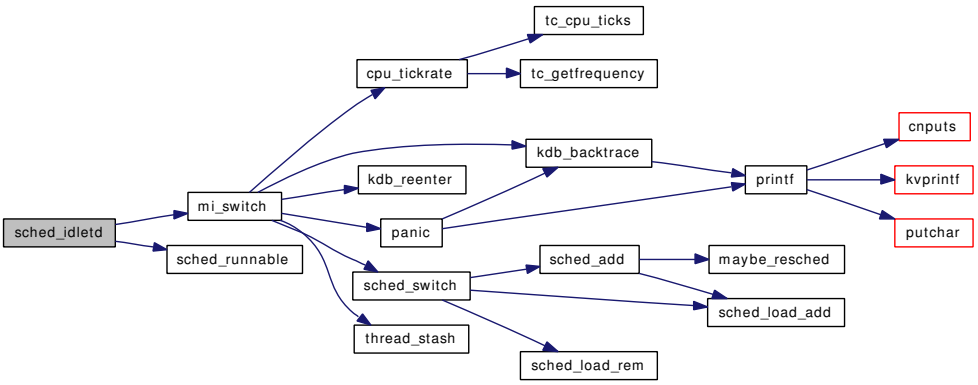
9.80.2.15 void sched_idletd (void * *dummy*)

Definition at line 1348 of file sched_4bsd.c.

References Giant, mi_switch(), sched_lock, and sched_runnable().

Referenced by idle_setup().

Here is the call graph for this function:



9.80.2.16 int sched_is_bound (struct thread * *td*)

Definition at line 1296 of file sched_4bsd.c.

References sched_lock, and TSF_BOUND.

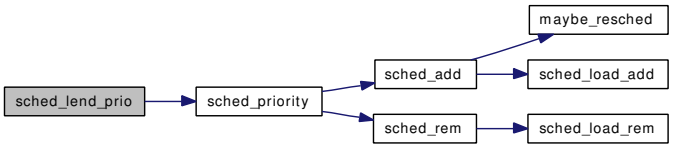
9.80.2.17 void sched_lend_prio (struct thread * *td*, u_char *prio*)

Definition at line 717 of file sched_4bsd.c.

References sched_priority().

Referenced by propagate_priority(), sched_unlend_prio(), and turnstile_claim().

Here is the call graph for this function:



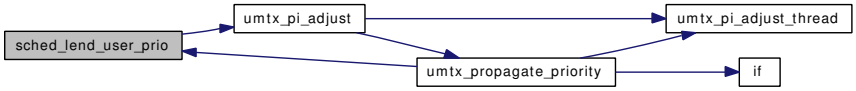
9.80.2.18 void sched_lend_user_prio (struct thread * *td*, u_char *prio*)

Definition at line 792 of file sched_4bsd.c.

References umtx_pi_adjust().

Referenced by _do_lock_pp(), sched_unlend_user_prio(), umtx_pi_claim(), and umtx_propagate_-priority().

Here is the call graph for this function:



9.80.2.19 int sched_load (void)

Definition at line 1313 of file sched_4bsd.c.

References sched_tdcnt.

Referenced by loadav().

9.80.2.20 static __inline void sched_load_add (void) [static]

Definition at line 231 of file sched_4bsd.c.

References sched_tdcnt.

Referenced by sched_add(), sched_setup(), and sched_switch().

9.80.2.21 static __inline void sched_load_rem (void) [static]

Definition at line 238 of file sched_4bsd.c.

References sched_tdcnt.

Referenced by sched_exit_thread(), sched_rem(), and sched_switch().

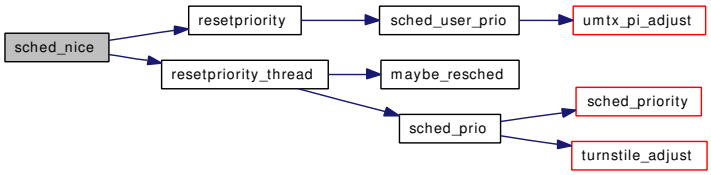
9.80.2.22 void sched_nice (struct proc * *p*, int *nice*)

Definition at line 671 of file sched_4bsd.c.

References resetpriority(), resetpriority_thread(), sched_lock, and td.

Referenced by donice(), and tdsigwakeup().

Here is the call graph for this function:



9.80.2.23 fixpt_t sched_pctcpu (struct thread * *td*)

Definition at line 1331 of file sched_4bsd.c.

Referenced by fill_kinfo_thread(), and ttyinfo().

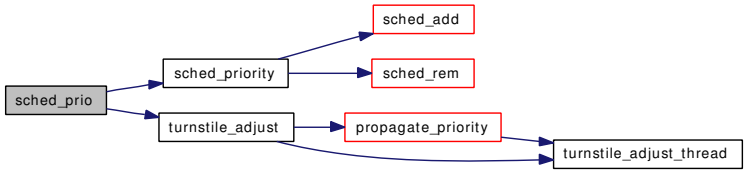
9.80.2.24 void sched_prio (struct thread * *td*, u_char *prio*)

Definition at line 750 of file sched_4bsd.c.

References sched_priority(), and turnstile_adjust().

Referenced by acct_thread(), ast(), create_thread(), idle_setup(), ithread_update(), msleep(), resetpriority_-thread(), rtp_to_pri(), sched_nice(), sched_relinquish(), sched_unlend_prio(), sleepq_resume_thread(), taskqueue_start_threads(), tdsigwakeup(), and uio_yield().

Here is the call graph for this function:



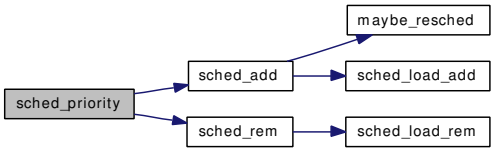
9.80.2.25 static void sched_priority (struct thread * *td*, u_char *prio*) [static]

Definition at line 695 of file sched_4bsd.c.

References sched_add(), sched_lock, and sched_rem().

Referenced by sched_add(), sched_clock(), sched_exit_thread(), sched_fork(), sched_lend_prio(), sched_-nice(), sched_prio(), sched_wakeup(), and tdq_runq_rem().

Here is the call graph for this function:



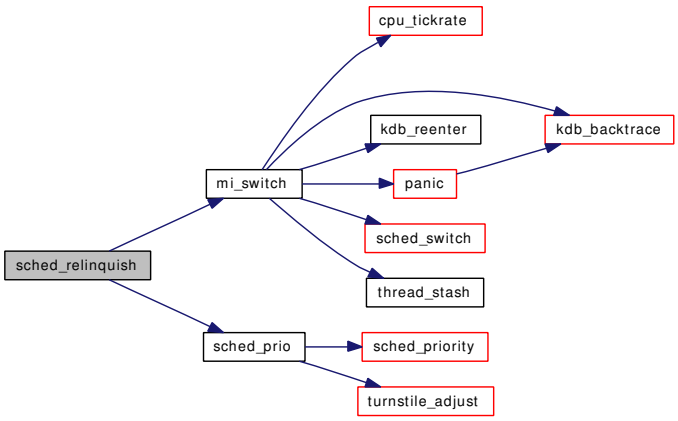
9.80.2.26 void sched_relinquish (struct thread * *td*)

Definition at line 1303 of file sched_4bsd.c.

References mi_switch(), sched_lock, and sched_prio().

Referenced by ksched_yield(), and yield().

Here is the call graph for this function:



9.80.2.27 void sched_rem (struct thread * *td*)

Definition at line 1181 of file sched_4bsd.c.

References sched_load_rem(), and sched_lock.

Referenced by sched_priority(), and sched_thread_priority().

Here is the call graph for this function:

9.80.2.28 int sched_rr_interval (void)

Definition at line 593 of file sched_4bsd.c.

References SCHED_QUANTUM, and sched_quantum.

Referenced by ksched_attach().

9.80.2.29 int sched_runnable (void)

Definition at line 583 of file sched_4bsd.c.

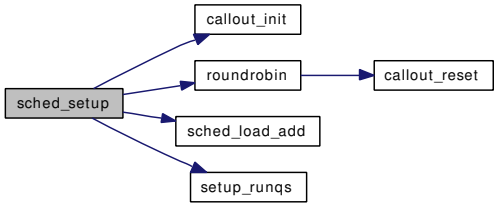
Referenced by sched_idletd().

9.80.2.30 static void sched_setup (void * *dummy*) [static]

Definition at line 547 of file sched_4bsd.c.

References callout_init(), hogticks, roundrobin(), roundrobin_callout, sched_load_add(), SCHED_-QUANTUM, sched_quantum, and setup_runqs().

Here is the call graph for this function:



9.80.2.31 int sched_sizeof_proc (void)

Definition at line 1319 of file sched_4bsd.c.

Referenced by procinit().

9.80.2.32 int sched_sizeof_thread (void)

Definition at line 1325 of file sched_4bsd.c.

Referenced by threadinit().

9.80.2.33 void sched_sleep (struct thread * *td*)

Definition at line 819 of file sched_4bsd.c.

References sched_lock.

Referenced by sleepq_switch().

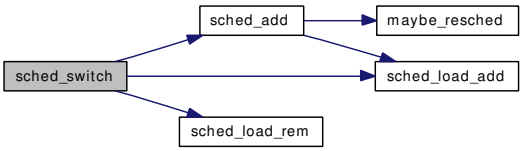
9.80.2.34 void sched_switch (struct thread * *td*, struct thread * *newtd*, int *flags*)

Definition at line 827 of file sched_4bsd.c.

References sched_add(), sched_load_add(), sched_load_rem(), sched_lock, and TSF_DIDRUN.

Referenced by mi_switch().

Here is the call graph for this function:



9.80.2.35 void sched_tick (void)

Definition at line 1340 of file sched_4bsd.c.

Referenced by hardclock_cpu(), and sched_fork_thread().

9.80.2.36 void sched_unbind (struct thread * *td*)

Definition at line 1289 of file sched_4bsd.c.

References sched_lock, and TSF_BOUND.

Referenced by cf_set_method(), and sched_bind().

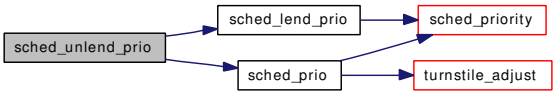
9.80.2.37 void sched_unlend_prio (struct thread * *td*, u_char *prio*)

Definition at line 733 of file sched_4bsd.c.

References sched_lend_prio(), and sched_prio().

Referenced by turnstile_disown(), and turnstile_unpend().

Here is the call graph for this function:



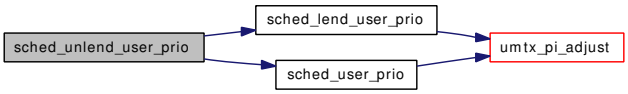
9.80.2.38 void sched_unlend_user_prio (struct thread * *td*, u_char *prio*)

Definition at line 806 of file sched_4bsd.c.

References sched_lend_user_prio(), and sched_user_prio().

Referenced by _do_lock_pp(), do_unlock_pi(), do_unlock_pp(), and umtx_unpropagate_priority().

Here is the call graph for this function:



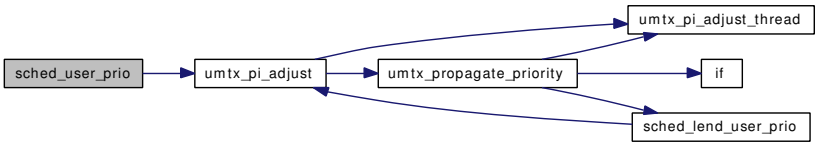
9.80.2.39 void sched_user_prio (struct thread * *td*, u_char *prio*)

Definition at line 777 of file sched_4bsd.c.

References umtx_pi_adjust().

Referenced by resetpriority(), rtp_to_pri(), sched_fork_thread(), sched_nice(), sched_priority(), sched_tick(), sched_unlend_user_prio(), and sched_wakeup().

Here is the call graph for this function:



9.80.2.40 void sched_userret (struct thread * *td*)

Definition at line 1246 of file sched_4bsd.c.

References sched_lock.

Referenced by userret().

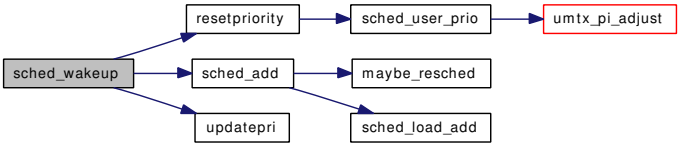
9.80.2.41 void sched_wakeup (struct thread * *td*)

Definition at line 939 of file sched_4bsd.c.

References resetpriority(), sched_add(), sched_lock, and updatepri().

Referenced by setrunnable().

Here is the call graph for this function:



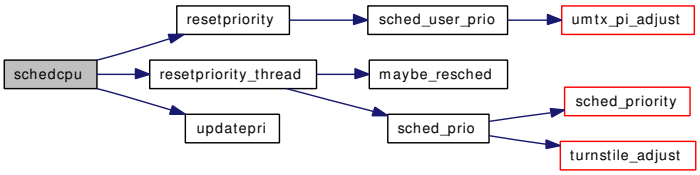
9.80.2.42 static void schedcpu (void) [static]

Definition at line 369 of file sched_4bsd.c.

References allproc_lock, averunnable, CCPU_SHIFT, decay_cpu, hz, loadfactor, realstathz, resetpriority(), resetpriority_thread(), sched_lock, stathz, td, TSF_DIDRUN, and updatepri().

Referenced by schedcpu_thread().

Here is the call graph for this function:

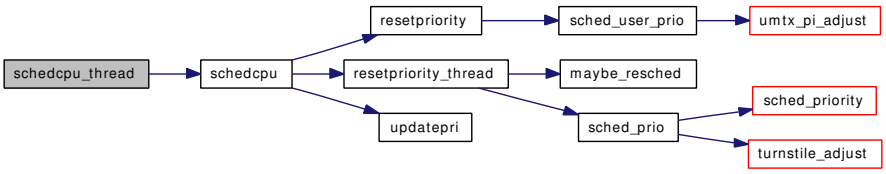


9.80.2.43 static void schedcpu_thread (void) [static]

Definition at line 474 of file sched_4bsd.c.

References hz, and schedcpu().

Here is the call graph for this function:



9.80.2.44 void schedinit (void)

Definition at line 572 of file sched_4bsd.c.

References proc0, td_sched0, and td_sched::ts_thread.

Referenced by proc0_init().

9.80.2.45 static void setup_runqs (void) [static]

Definition at line 146 of file sched_4bsd.c.

Referenced by sched_setup().

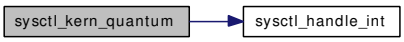
9.80.2.46 SYSCTL_INT (_kern, OID_AUTO, ccpu, CTLFLAG_RD, & *ccpu*, 0, "")

9.80.2.47 static int sysctl_kern_quantum (SYSCTL_HANDLER_ARGS) [static]

Definition at line 159 of file sched_4bsd.c.

References hogticks, sched_quantum, sysctl_handle_int(), and tick.

Here is the call graph for this function:



9.80.2.48 SYSCTL_NODE (_kern, OID_AUTO, sched, CTLFLAG_RD, 0, "Scheduler")

9.80.2.49 SYSCTL_PROC (_kern_sched, OID_AUTO, quantum, CTLTYPE_INT| *CTLFLAG_RW*, 0, sizeof *sched_quantum*, sysctl_kern_quantum, "I", "Roundrobin scheduling quantum in microseconds")

9.80.2.50 SYSCTL_STRING (_kern_sched, OID_AUTO, name, CTLFLAG_RD, "4BSD", 0, "Scheduler name")

9.80.2.51 static void updatepri (struct thread * *td*) [static]

Definition at line 490 of file sched_4bsd.c.

References averunnable, decay_cpu, and loadfactor.

Referenced by sched_wakeup(), and schedcpu().

9.80.3 Variable Documentation

9.80.3.1 fixpt_t ccpu = 0.95122942450071400909 * FSCALE [static]

Definition at line 346 of file sched_4bsd.c.

9.80.3.2 struct callout roundrobin_callout [static]

Definition at line 109 of file sched_4bsd.c.

Referenced by roundrobin(), and sched_setup().

9.80.3.3 struct runq runq [static]

Definition at line 136 of file sched_4bsd.c.

9.80.3.4 struct kproc_desc sched_kp [static]

Initial value:

```
 {
        "schedcpu",
        schedcpu_thread,
        NULL
}
```

Definition at line 125 of file sched_4bsd.c.

9.80.3.5 int sched_quantum [static]

Definition at line 106 of file sched_4bsd.c.

Referenced by roundrobin(), sched_rr_interval(), sched_setup(), and sysctl_kern_quantum().

9.80.3.6 int sched_tdcnt [static]

Definition at line 105 of file sched_4bsd.c.

Referenced by sched_load(), sched_load_add(), and sched_load_rem().

9.80.3.7 struct td_sched td_sched0 [static]

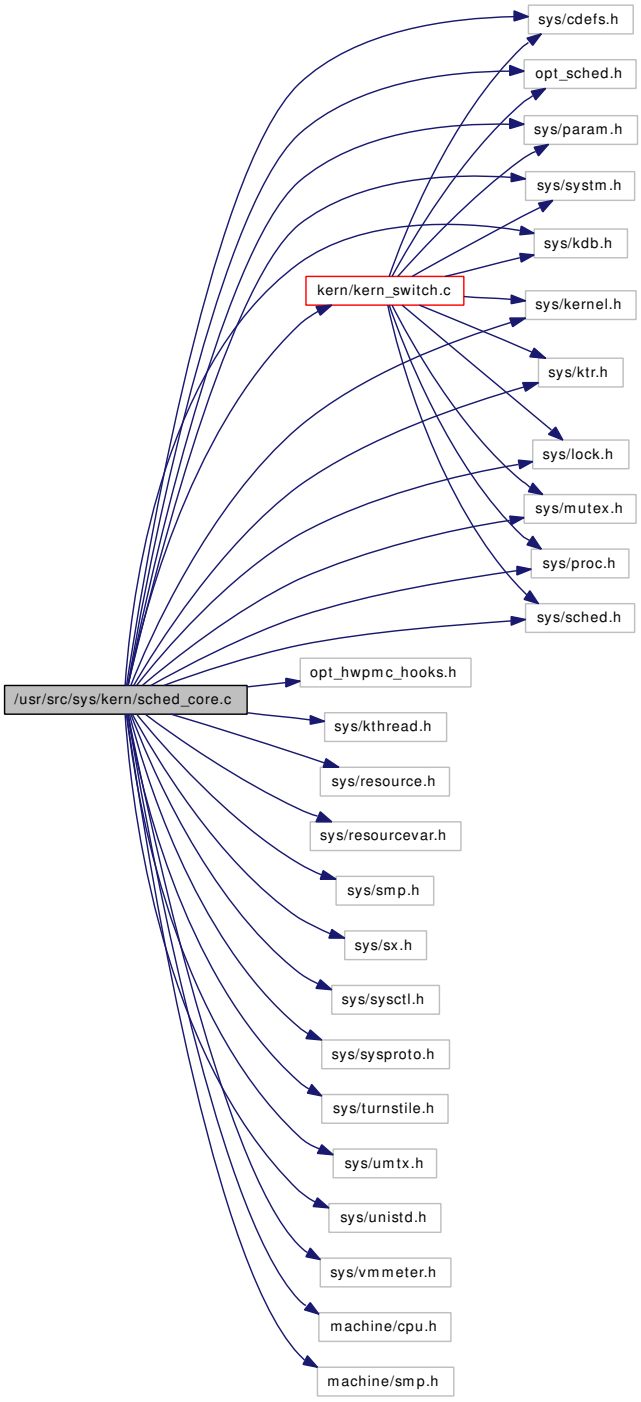
Definition at line 103 of file sched_4bsd.c.

Referenced by schedinit().

9.81 /usr/src/sys/kern/sched_core.c File Reference

```
#include <sys/cdefs.h>
#include "opt_hwpmc_hooks.h"
#include "opt_sched.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kdb.h>
#include <sys/kernel.h>
#include <sys/kthread.h>
#include <sys/ktr.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/resource.h>
#include <sys/resourcevar.h>
#include <sys/sched.h>
#include <sys/smp.h>
#include <sys/sx.h>
#include <sys/sysctl.h>
#include <sys/sysproto.h>
#include <sys/turnstile.h>
#include <sys/umtx.h>
#include <sys/unistd.h>
#include <sys/vmmeter.h>
#include <machine/cpu.h>
#include <machine/smp.h>
#include "kern/kern_switch.c"
```

Include dependency graph for sched_core.c:



Data Structures

- struct krqbits
- struct krunq
- struct td_sched
- struct kseq

Defines

- #define PROC_NICE(p) MIN((p) → p_nice, 19)
- #define NICE_TO_PRI(nice) (PUSER + 20 + (nice))
- #define PROC_PRI(p) NICE_TO_PRI(PROC_NICE(p))
- #define USER_PRI(pri) MIN((pri) - PUSER, 39)
- #define PROC_USER_PRI(p) (PROC_NICE(p) + 20)
- #define MAX_USER_PRI 40
- #define PUSER_MAX (PUSER + 39)
- #define NS_TO_HZ(n) ((n) / (1000000000 / hz))
- #define HZ_TO_NS(h) ((h) ∗ (1000000000 / hz))
- #define MS_TO_HZ(m) ((m) / (1000000 / hz))
- #define PRI_SCORE_RATIO 25
- #define MAX_SCORE (MAX_USER_PRI ∗ PRI_SCORE_RATIO / 100)
- #define MAX_SLEEP_TIME (def_timeslice ∗ MAX_SCORE)
- #define NS_MAX_SLEEP_TIME (HZ_TO_NS(MAX_SLEEP_TIME))
- #define STARVATION_TIME (MAX_SLEEP_TIME)
- #define CURRENT_SCORE(ts) (MAX_SCORE ∗ NS_TO_HZ((ts) → ts_slptime) / MAX_-SLEEP_TIME)
- #define SCALE_USER_PRI(x, upri) MAX(x ∗ (upri + 1) / (MAX_USER_PRI/2), min_timeslice)
- #define INTERACTIVE_BASE_SCORE (MAX_SCORE ∗ 20)/100
- #define INTERACTIVE_SCORE(ts) (PROC_NICE((ts) → ts_proc) ∗ MAX_SCORE / 40 + INTERACTIVE_BASE_SCORE)
- #define THREAD_IS_INTERACTIVE(ts)
- #define INTERACTIVE_SLEEP_TIME(ts)
- #define CHILD_WEIGHT 90
- #define PARENT_WEIGHT 90
- #define EXIT_WEIGHT 3
- #define SCHED_LOAD_SCALE 128UL
- #define IDLE 0
- #define IDLE_IDLE 1
- #define NOT_IDLE 2
- #define KQB_LEN (8)
- #define KQB_L2BPW (5)
- #define KQB_BPW (1<<KQB_L2BPW)
- #define KQB_BIT(pri) (1 << ((pri) & (KQB_BPW - 1)))
- #define KQB_WORD(pri) ((pri) >> KQB_L2BPW)
- #define KQB_FFS(word) (ffs(word) - 1)
- #define KQ_NQS 256
- #define td_sched td_sched
- #define ts_proc ts_thread → td_proc
- #define TSF_BOUND 0x0001
- #define TSF_PREEMPTED 0x0002
- #define TSF_MIGRATING 0x0004
- #define TSF_SLEEP 0x0008
- #define TSF_DIDRUN 0x0010
- #define TSF_EXIT 0x0020
- #define TSF_NEXTRQ 0x0400
- #define TSF_FIRST_SLICE 0x0800
- #define SCHED_CPU_TIME 10
- #define SCHED_CPU_TICKS (hz ∗ SCHED_CPU_TIME)
- #define KSEQ_SELF() (&kseq_global)
- #define KSEQ_CPU(x) (&kseq_global)
- #define KERN_SWITCH_INCLUDE 1

Typedefs

- typedef u_int32_t kqb_word_t

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/sched_core.c,v 1.12 2007/01/23 08:46:50 jeff Exp $")
- TAILQ_HEAD (krqhead, td_sched)
- SYSCTL_INT (_kern, OID_AUTO, ccpu, CTLFLAG_RD,&ccpu, 0,"")
- static void sched_setup (void *dummy)
- SYSINIT (sched_setup, SI_SUB_RUN_QUEUE, SI_ORDER_FIRST, sched_setup, NULL)
- static void sched_initticks (void *dummy)
- static SYSCTL_NODE (_kern, OID_AUTO, sched, CTLFLAG_RW, 0,"Scheduler")
- SYSCTL_STRING (_kern_sched, OID_AUTO, name, CTLFLAG_RD,"CORE", 0,"Scheduler name")
- static void krunq_add (struct krunq *, struct td_sched *)
- static struct td_sched * krunq_choose (struct krunq *)
- static void krunq_clrbit (struct krunq *rq, int pri)
- static int krunq_findbit (struct krunq *rq)
- static void krunq_init (struct krunq *)
- static void krunq_remove (struct krunq *, struct td_sched *)
- static struct td_sched * kseq_choose (struct kseq *)
- static void kseq_load_add (struct kseq *, struct td_sched *)
- static void kseq_load_rem (struct kseq *, struct td_sched *)
- static void kseq_runq_add (struct kseq *, struct td_sched *)
- static void kseq_runq_rem (struct kseq *, struct td_sched *)
- static void kseq_setup (struct kseq *)
- static int sched_is_timeshare (struct thread *td)
- static int sched_calc_pri (struct td_sched *ts)
- static int sched_starving (struct kseq *, unsigned, struct td_sched *)
- static void sched_pctcpu_update (struct td_sched *)
- static void sched_thread_priority (struct thread *, u_char)
- static uint64_t sched_timestamp (void)
- static int sched_recalc_pri (struct td_sched *ts, uint64_t now)
- static int sched_timeslice (struct td_sched *ts)
- static void sched_update_runtime (struct td_sched *ts, uint64_t now)
- static void sched_commit_runtime (struct td_sched *ts)
- static int krunq_check (struct krunq *rq)
- static void krunq_setbit (struct krunq *rq, int pri)
- void schedinit (void)
- int sched_rr_interval (void)
- void sched_lend_prio (struct thread *td, u_char prio)
- void sched_unlend_prio (struct thread *td, u_char prio)
- void sched_prio (struct thread *td, u_char prio)
- void sched_user_prio (struct thread *td, u_char prio)
- void sched_lend_user_prio (struct thread *td, u_char prio)
- void sched_unlend_user_prio (struct thread *td, u_char prio)
- void sched_switch (struct thread *td, struct thread *newtd, int flags)
- void sched_nice (struct proc *p, int nice)
- void sched_sleep (struct thread *td)

- void sched_wakeup (struct thread *td)
- void sched_fork (struct thread *td, struct thread *childtd)
- void sched_fork_thread (struct thread *td, struct thread *child)
- void sched_class (struct thread *td, int class)
- void sched_exit (struct proc *p, struct thread *childtd)
- void sched_exit_thread (struct thread *td, struct thread *childtd)
- static int sched_timeslice_split (struct td_sched *ts)
- void sched_tick (void)
- void sched_clock (struct thread *td)
- static int kseq_runnable (struct kseq *kseq)
- int sched_runnable (void)
- void sched_userret (struct thread *td)
- thread * sched_choose (void)
- void sched_add (struct thread *td, int flags)
- void sched_rem (struct thread *td)
- fixpt_t sched_pctcpu (struct thread *td)
- void sched_bind (struct thread *td, int cpu)
- void sched_unbind (struct thread *td)
- int sched_is_bound (struct thread *td)
- int sched_load (void)
- void sched_relinquish (struct thread *td)
- int sched_sizeof_proc (void)
- int sched_sizeof_thread (void)
- void sched_idletd (void *dummy)

Variables

- static struct td_sched kse0
- static int min_timeslice = 5
- static int def_timeslice = 100
- static int granularity = 10
- static int realstathz
- static int sched_tdcnt
- static struct kseq kseq_global
- static fixpt_t ccpu = 0.95122942450071400909 * FSCALE

9.81.1 Define Documentation

9.81.1.1 #define CHILD_WEIGHT 90

Definition at line 131 of file sched_core.c.

Referenced by sched_fork_thread().

9.81.1.2 #define CURRENT_SCORE(ts) (MAX_SCORE * NS_TO_HZ((ts) → ts_slptime) / MAX_SLEEP_TIME)

Definition at line 99 of file sched_core.c.

Referenced by sched_calc_pri(), sched_recalc_pri(), sched_timeslice_split(), and sched_update_runtime().

9.81.1.3 #define EXIT_WEIGHT 3

Definition at line 133 of file sched_core.c.

Referenced by sched_exit_thread().

9.81.1.4 #define HZ_TO_NS(h) ((h) ∗ (1000000000 / hz))

Definition at line 88 of file sched_core.c.

Referenced by sched_recalc_pri(), and sched_starving().

9.81.1.5 #define IDLE 0

Definition at line 137 of file sched_core.c.

9.81.1.6 #define IDLE_IDLE 1

Definition at line 138 of file sched_core.c.

9.81.1.7 #define INTERACTIVE_BASE_SCORE (MAX_SCORE ∗ 20)/100

Definition at line 109 of file sched_core.c.

9.81.1.8 #define INTERACTIVE_SCORE(ts) (PROC_NICE((ts) → ts_proc) ∗ MAX_SCORE / 40 + INTERACTIVE_BASE_SCORE)

Definition at line 115 of file sched_core.c.

9.81.1.9 #define INTERACTIVE_SLEEP_TIME(ts)

Value:

```
(HZ_TO_NS(MAX_SLEEP_TIME *              \
        (MAX_SCORE / 2 + INTERACTIVE_SCORE((ts)) + 1) / MAX_SCORE - 1))
```

Definition at line 127 of file sched_core.c.

Referenced by sched_recalc_pri().

9.81.1.10 #define KERN_SWITCH_INCLUDE 1

Definition at line 1750 of file sched_core.c.

9.81.1.11 #define KQ_NQS 256

Definition at line 149 of file sched_core.c.

Referenced by krunq_init().

9.81.1.12 #define KQB_BIT(pri) (1 << ((pri) & (KQB_BPW - 1)))

Definition at line 145 of file sched_core.c.

Referenced by krunq_clrbit(), and krunq_setbit().

9.81.1.13 #define KQB_BPW (1<<KQB_L2BPW)

Definition at line 143 of file sched_core.c.

9.81.1.14 #define KQB_FFS(word) (ffs(word) - 1)

Definition at line 147 of file sched_core.c.

Referenced by krunq_findbit().

9.81.1.15 #define KQB_L2BPW (5)

Definition at line 142 of file sched_core.c.

Referenced by krunq_findbit().

9.81.1.16 #define KQB_LEN (8)

Definition at line 141 of file sched_core.c.

Referenced by krunq_check(), and krunq_findbit().

9.81.1.17 #define KQB_WORD(pri) ((pri) >> KQB_L2BPW)

Definition at line 146 of file sched_core.c.

Referenced by krunq_clrbit(), and krunq_setbit().

9.81.1.18 #define KSEQ_CPU(x) (&kseq_global)

Definition at line 273 of file sched_core.c.

Referenced by sched_add(), sched_tick(), and sched_wakeup().

9.81.1.19 #define KSEQ_SELF() (&kseq_global)

Definition at line 272 of file sched_core.c.

Referenced by sched_choose(), sched_exit_thread(), sched_runnable(), sched_setup(), sched_switch(), and sched_wakeup().

9.81.1.20 #define MAX_SCORE (MAX_USER_PRI * PRI_SCORE_RATIO / 100)

Definition at line 94 of file sched_core.c.

Referenced by sched_calc_pri(), sched_recalc_pri(), and sched_timeslice_split().

9.81.1.21 #define MAX_SLEEP_TIME (def_timeslice * MAX_SCORE)

Definition at line 95 of file sched_core.c.

Referenced by sched_recalc_pri().

9.81.1.22 #define MAX_USER_PRI 40

Definition at line 81 of file sched_core.c.

9.81.1.23 #define MS_TO_HZ(m) ((m) / (1000000 / hz))

Definition at line 91 of file sched_core.c.

9.81.1.24 #define NICE_TO_PRI(nice) (PUSER + 20 + (nice))

Definition at line 69 of file sched_core.c.

9.81.1.25 #define NOT_IDLE 2

Definition at line 139 of file sched_core.c.

9.81.1.26 #define NS_MAX_SLEEP_TIME (HZ_TO_NS(MAX_SLEEP_TIME))

Definition at line 96 of file sched_core.c.

Referenced by sched_recalc_pri(), and sched_update_runtime().

9.81.1.27 #define NS_TO_HZ(n) ((n) / (1000000000 / hz))

Definition at line 87 of file sched_core.c.

9.81.1.28 #define PARENT_WEIGHT 90

Definition at line 132 of file sched_core.c.

Referenced by sched_fork_thread().

9.81.1.29 #define PRI_SCORE_RATIO 25

Definition at line 93 of file sched_core.c.

9.81.1.30 #define PROC_NICE(p) MIN((p) → p_nice, 19)

Definition at line 66 of file sched_core.c.

9.81.1.31 #define PROC_PRI(p) NICE_TO_PRI(PROC_NICE(p))

Definition at line 72 of file sched_core.c.

Referenced by sched_calc_pri().

9.81.1.32 #define PROC_USER_PRI(p) (PROC_NICE(p) + 20)

Definition at line 78 of file sched_core.c.

Referenced by sched_timeslice().

9.81.1.33 #define PUSER_MAX (PUSER + 39)

Definition at line 84 of file sched_core.c.

Referenced by sched_calc_pri(), and sched_prio().

9.81.1.34 #define SCALE_USER_PRI(x, upri) MAX(x ∗ (upri + 1) / (MAX_USER_PRI/2), min_timeslice)

Definition at line 102 of file sched_core.c.

Referenced by sched_timeslice().

9.81.1.35 #define SCHED_CPU_TICKS (hz ∗ SCHED_CPU_TIME)

Definition at line 233 of file sched_core.c.

Referenced by sched_clock(), sched_pctcpu(), and sched_pctcpu_update().

9.81.1.36 #define SCHED_CPU_TIME 10

Definition at line 232 of file sched_core.c.

Referenced by sched_pctcpu().

9.81.1.37 #define SCHED_LOAD_SCALE 128UL

Definition at line 135 of file sched_core.c.

9.81.1.38 #define STARVATION_TIME (MAX_SLEEP_TIME)

Definition at line 97 of file sched_core.c.

Referenced by sched_starving().

9.81.1.39 #define td_sched td_sched

Definition at line 212 of file sched_core.c.

9.81.1.40 #define THREAD_IS_INTERACTIVE(ts)

Value:

```
((ts)->ts_thread->td_user_pri <=        \
        PROC_PRI((ts)->ts_proc) - INTERACTIVE_SCORE(ts))
```

Definition at line 119 of file sched_core.c.

Referenced by sched_tick().

9.81.1.41 #define ts_proc ts_thread → td_proc

Definition at line 213 of file sched_core.c.

9.81.1.42 #define TSF_BOUND 0x0001

Definition at line 216 of file sched_core.c.

9.81.1.43 #define TSF_DIDRUN 0x0010

Definition at line 220 of file sched_core.c.

9.81.1.44 #define TSF_EXIT 0x0020

Definition at line 221 of file sched_core.c.

9.81.1.45 #define TSF_FIRST_SLICE 0x0800

Definition at line 223 of file sched_core.c.

Referenced by sched_exit_thread(), sched_fork_thread(), and sched_tick().

9.81.1.46 #define TSF_MIGRATING 0x0004

Definition at line 218 of file sched_core.c.

9.81.1.47 #define TSF_NEXTRQ 0x0400

Definition at line 222 of file sched_core.c.

Referenced by sched_add(), sched_fork_thread(), sched_relinquish(), sched_switch(), and sched_tick().

9.81.1.48 #define TSF_PREEMPTED 0x0002

Definition at line 217 of file sched_core.c.

Referenced by krunq_add(), sched_add(), and sched_choose().

9.81.1.49 #define TSF_SLEEP 0x0008

Definition at line 219 of file sched_core.c.

Referenced by sched_sleep(), and sched_wakeup().

9.81.1.50 #define USER_PRI(pri) MIN((pri) - PUSER, 39)

Definition at line 75 of file sched_core.c.

9.81.2 Typedef Documentation

9.81.2.1 typedef u_int32_t kqb_word_t

Definition at line 154 of file sched_core.c.

9.81.3 Function Documentation

9.81.3.1 __FBSDID ("$FreeBSD: src/sys/kern/sched_core. *c*, v 1.12 2007/01/23 08:46:50 jeff Exp $")

9.81.3.2 static void krunq_add (struct krunq *, struct td_sched *) [static]

Definition at line 438 of file sched_core.c.

References krunq_setbit(), krunq::rq_queues, td_sched::ts_thread, and TSF_PREEMPTED.

Referenced by kseq_runq_add(), and sched_thread_priority().

Here is the call graph for this function:

9.81.3.3 static int krunq_check (struct krunq * *rq*) [static]

Definition at line 407 of file sched_core.c.

References KQB_LEN, krunq::rq_status, and krqbits::rqb_bits.

Referenced by kseq_runnable().

9.81.3.4 static struct td_sched * krunq_choose (struct krunq *) [static]

Definition at line 457 of file sched_core.c.

References krunq_findbit(), krunq::rq_queues, and sched_lock.

Referenced by kseq_choose().

Here is the call graph for this function:

9.81.3.5 static void krunq_clrbit (struct krunq * *rq*, int *pri*) [inline, static]

Definition at line 377 of file sched_core.c.

References KQB_BIT, KQB_WORD, krunq::rq_status, and krqbits::rqb_bits.

Referenced by krunq_remove().

9.81.3.6 static int krunq_findbit (struct krunq * *rq*) [static]

Definition at line 390 of file sched_core.c.

References KQB_FFS, KQB_L2BPW, KQB_LEN, krunq::rq_status, and krqbits::rqb_bits.

Referenced by krunq_choose().

9.81.3.7 static void krunq_init (struct krunq *) [static]

Definition at line 363 of file sched_core.c.

References KQ_NQS.

Referenced by kseq_setup().

9.81.3.8 static void krunq_remove (struct krunq *, struct td_sched *) [static]

Definition at line 502 of file sched_core.c.

References krunq_clrbit(), and krunq::rq_queues.

Referenced by kseq_runq_rem(), and sched_thread_priority().

Here is the call graph for this function:

9.81.3.9 static void krunq_setbit (struct krunq * *rq*, int *pri*) [inline, static]

Definition at line 425 of file sched_core.c.

References KQB_BIT, KQB_WORD, krunq::rq_status, and krqbits::rqb_bits.

Referenced by krunq_add().

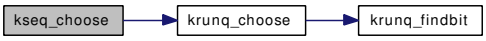
9.81.3.10 static struct td_sched * kseq_choose (struct kseq *) [static]

Definition at line 552 of file sched_core.c.

References krunq_choose(), kseq::ksq_curr, kseq::ksq_expired_nice, kseq::ksq_expired_tick, kseq::ksq_-idle, kseq::ksq_next, and sched_lock.

Referenced by sched_choose().

Here is the call graph for this function:



9.81.3.11 static void kseq_load_add (struct kseq *, struct td_sched *) [inline, static]

Definition at line 533 of file sched_core.c.

References kseq::ksq_load.

Referenced by sched_add(), sched_setup(), and sched_switch().

9.81.3.12 static void kseq_load_rem (struct kseq *, struct td_sched *) [inline, static]

Definition at line 541 of file sched_core.c.

References kseq::ksq_load.

Referenced by sched_exit_thread(), sched_rem(), and sched_switch().

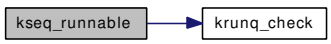
9.81.3.13 static int kseq_runnable (struct kseq * *kseq*) [static]

Definition at line 1316 of file sched_core.c.

References krunq_check(), kseq::ksq_curr, kseq::ksq_idle, and kseq::ksq_next.

Referenced by sched_runnable().

Here is the call graph for this function:



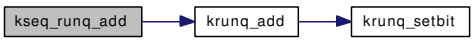
9.81.3.14 static void kseq_runq_add (struct kseq *, struct td_sched *) [inline, static]

Definition at line 518 of file sched_core.c.

References krunq_add().

Referenced by sched_add().

Here is the call graph for this function:



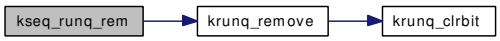
9.81.3.15 static void kseq_runq_rem (struct kseq *, struct td_sched *) [inline, static]

Definition at line 525 of file sched_core.c.

References krunq_remove().

Referenced by sched_choose(), and sched_rem().

Here is the call graph for this function:



9.81.3.16 static void kseq_setup (struct kseq *) [static]

Definition at line 698 of file sched_core.c.

References krunq_init(), kseq::ksq_curr, kseq::ksq_expired_nice, kseq::ksq_expired_tick, kseq::ksq_idle, kseq::ksq_next, and kseq::ksq_timeshare.

Referenced by sched_setup().

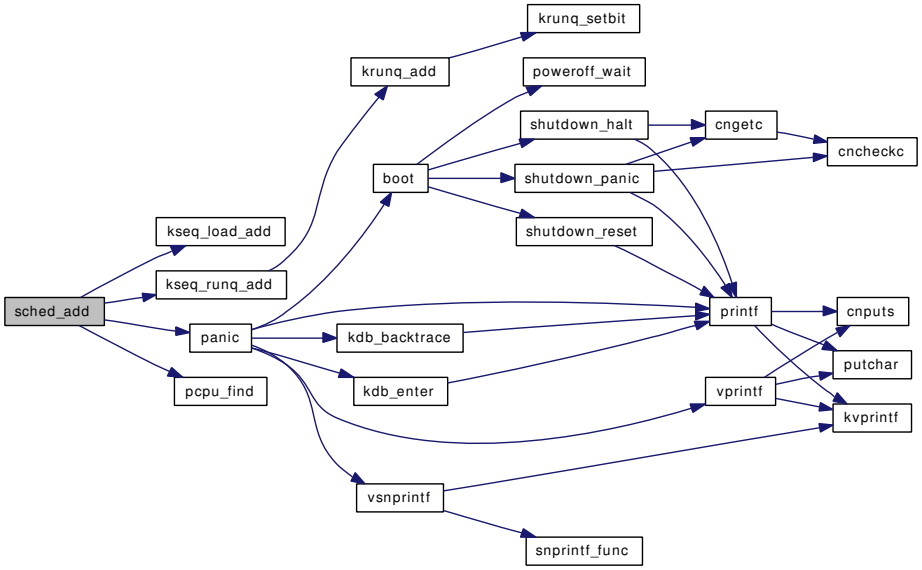
Here is the call graph for this function:

9.81.3.17 void sched_add (struct thread * *td*, int *flags*)

Definition at line 1467 of file sched_core.c.

References KSEQ_CPU, kseq_global, kseq_load_add(), kseq_runq_add(), kseq::ksq_curr, kseq::ksq_idle, kseq::ksq_next, panic(), pcpu_find(), sched_lock, ts_flags, TSF_BOUND, TSF_NEXTRQ, and TSF_-PREEMPTED.

Here is the call graph for this function:

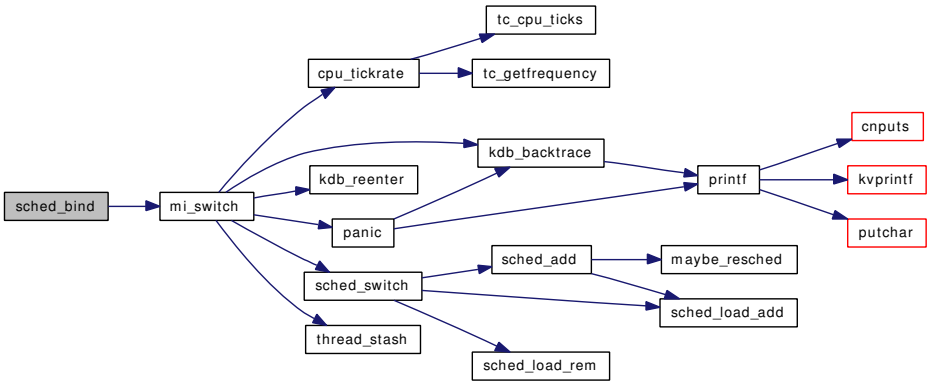


9.81.3.18 void sched_bind (struct thread * *td*, int *cpu*)

Definition at line 1652 of file sched_core.c.

References mi_switch(), and sched_lock.

Here is the call graph for this function:



9.81.3.19 static int sched_calc_pri (struct td_sched * *ts*) [static]

Definition at line 599 of file sched_core.c.

References CURRENT_SCORE, MAX_SCORE, PROC_PRI, PUSER_MAX, sched_is_timeshare(), and td_sched::ts_thread.

Referenced by sched_fork_thread(), sched_nice(), sched_recalc_pri(), and sched_tick().

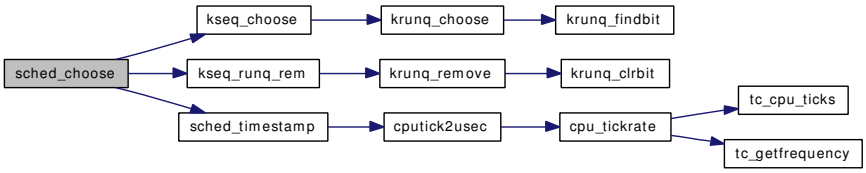
Here is the call graph for this function:

9.81.3.20 struct thread* sched_choose (void)

Definition at line 1348 of file sched_core.c.

References kseq_choose(), kseq_global, kseq_runq_rem(), KSEQ_SELF, sched_lock, sched_timestamp(), td_sched::ts_thread, and TSF_PREEMPTED.

Here is the call graph for this function:



9.81.3.21 void sched_class (struct thread * *td*, int *class*)

Definition at line 1133 of file sched_core.c.

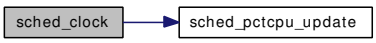
References sched_lock.

9.81.3.22 void sched_clock (struct thread * *td*)

Definition at line 1299 of file sched_core.c.

References SCHED_CPU_TICKS, sched_lock, sched_pctcpu_update(), and ticks.

Here is the call graph for this function:



9.81.3.23 static void sched_commit_runtime (struct td_sched * *ts*) [static]

Definition at line 687 of file sched_core.c.

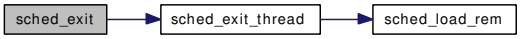
Referenced by sched_exit_thread(), sched_tick(), and sched_wakeup().

9.81.3.24 void sched_exit (struct proc * *p*, struct thread * *childtd*)

Definition at line 1143 of file sched_core.c.

References sched_exit_thread(), and sched_lock.

Here is the call graph for this function:

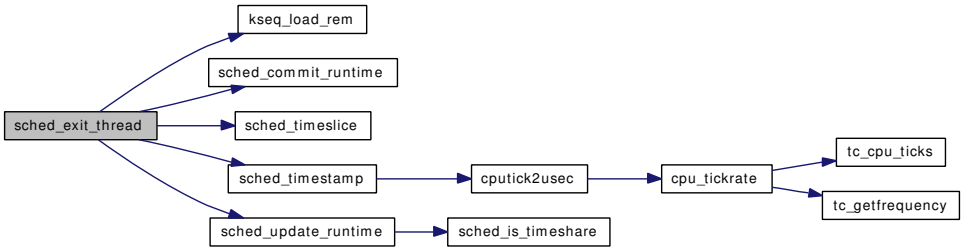


9.81.3.25 void sched_exit_thread (struct thread * *td*, struct thread * *childtd*)

Definition at line 1150 of file sched_core.c.

References EXIT_WEIGHT, kseq_load_rem(), KSEQ_SELF, sched_commit_runtime(), sched_timeslice(), sched_timestamp(), sched_update_runtime(), and TSF_FIRST_SLICE.

Here is the call graph for this function:



9.81.3.26 void sched_fork (struct thread * *td*, struct thread * *childtd*)

Definition at line 1095 of file sched_core.c.

References sched_fork_thread(), and sched_lock.

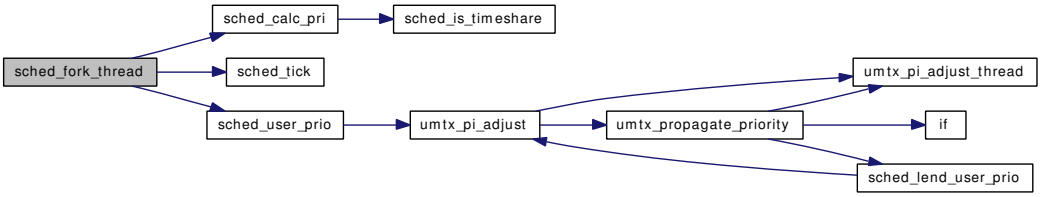
Here is the call graph for this function:

9.81.3.27 void sched_fork_thread (struct thread * *td*, struct thread * *child*)

Definition at line 1103 of file sched_core.c.

References CHILD_WEIGHT, PARENT_WEIGHT, sched_calc_pri(), sched_tick(), sched_user_prio(), TSF_FIRST_SLICE, and TSF_NEXTRQ.

Here is the call graph for this function:

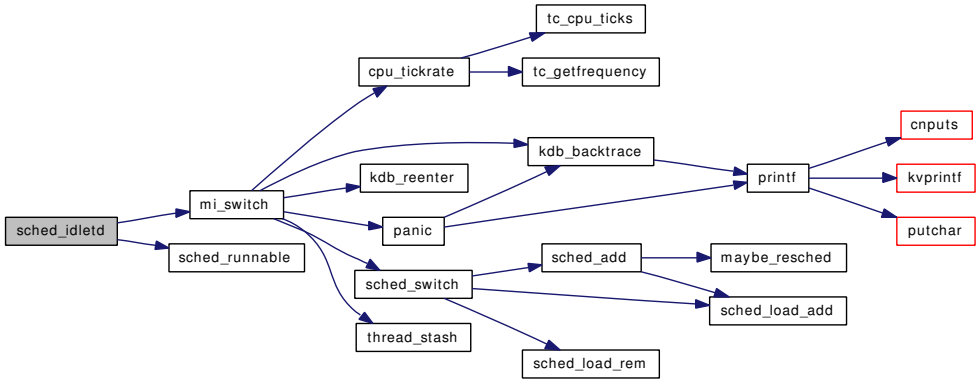


9.81.3.28 void sched_idletd (void * *dummy*)

Definition at line 1716 of file sched_core.c.

References Giant, mi_switch(), sched_lock, and sched_runnable().

Here is the call graph for this function:



9.81.3.29 static void sched_initticks (void * *dummy*) [static]

Definition at line 766 of file sched_core.c.

References hz, sched_lock, and stathz.

9.81.3.30 int sched_is_bound (struct thread * *td*)

Definition at line 1675 of file sched_core.c.

References sched_lock.

9.81.3.31 static int sched_is_timeshare (struct thread * *td*) [inline, static]

Definition at line 593 of file sched_core.c.

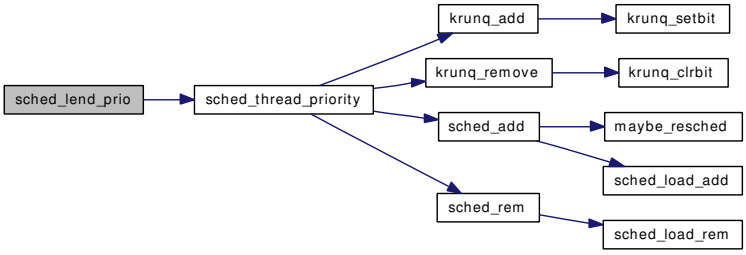
Referenced by sched_calc_pri(), sched_recalc_pri(), sched_relinquish(), sched_update_runtime(), and sched_wakeup().

9.81.3.32 void sched_lend_prio (struct thread * *td*, u_char *prio*)

Definition at line 862 of file sched_core.c.

References sched_thread_priority().

Here is the call graph for this function:

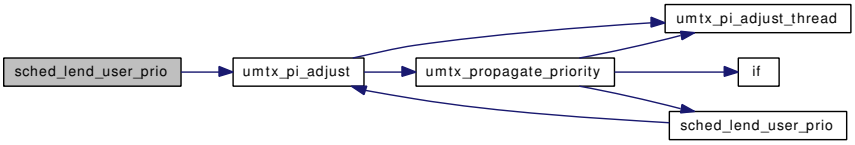


9.81.3.33 void sched_lend_user_prio (struct thread * *td*, u_char *prio*)

Definition at line 942 of file sched_core.c.

References umtx_pi_adjust().

Here is the call graph for this function:



9.81.3.34 int sched_load (void)

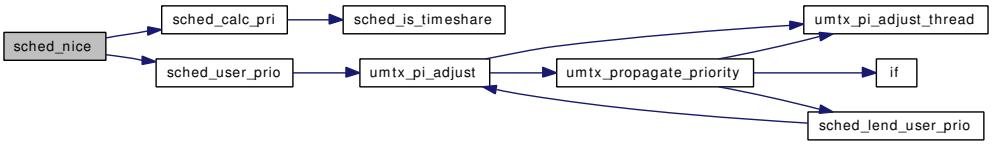
Definition at line 1682 of file sched_core.c.

9.81.3.35 void sched_nice (struct proc * *p*, int *nice*)

Definition at line 1034 of file sched_core.c.

References sched_calc_pri(), sched_lock, sched_user_prio(), and td.

Here is the call graph for this function:

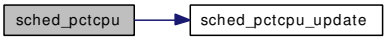


9.81.3.36 fixpt_t sched_pctcpu (struct thread * *td*)

Definition at line 1618 of file sched_core.c.

References hz, SCHED_CPU_TICKS, SCHED_CPU_TIME, sched_lock, sched_pctcpu_update(), and ticks.

Here is the call graph for this function:



9.81.3.37 static void sched_pctcpu_update (struct td_sched *) [static]

Definition at line 803 of file sched_core.c.

References SCHED_CPU_TICKS, and ticks.

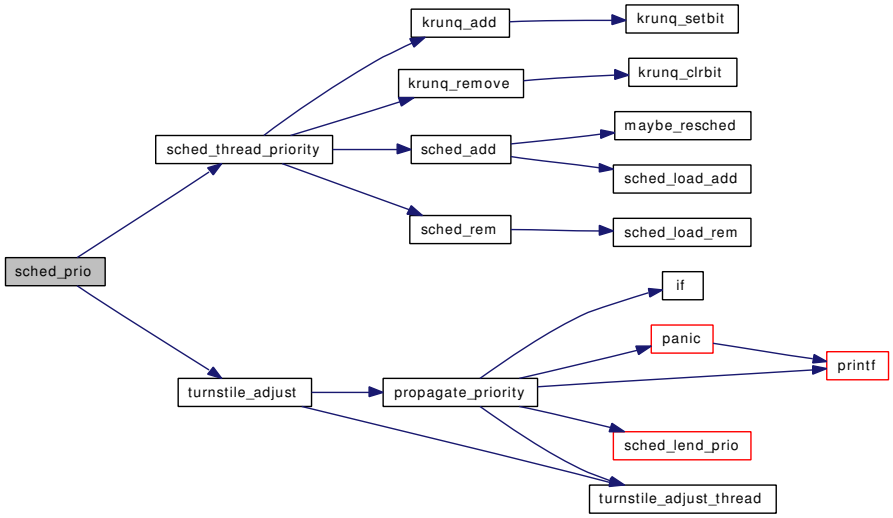
Referenced by sched_clock(), sched_pctcpu(), sched_tick(), sched_wakeup(), and tdq_runq_rem().

9.81.3.38 void sched_prio (struct thread * *td*, u_char *prio*)

Definition at line 895 of file sched_core.c.

References PUSER_MAX, sched_thread_priority(), and turnstile_adjust().

Here is the call graph for this function:



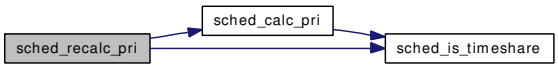
9.81.3.39 static int sched_recalc_pri (struct td_sched * *ts*, uint64_t *now*) [static]

Definition at line 616 of file sched_core.c.

References CURRENT_SCORE, HZ_TO_NS, INTERACTIVE_SLEEP_TIME, MAX_SCORE, MAX_-SLEEP_TIME, NS_MAX_SLEEP_TIME, sched_calc_pri(), sched_is_timeshare(), and td_sched::ts_-thread.

Referenced by sched_wakeup().

Here is the call graph for this function:

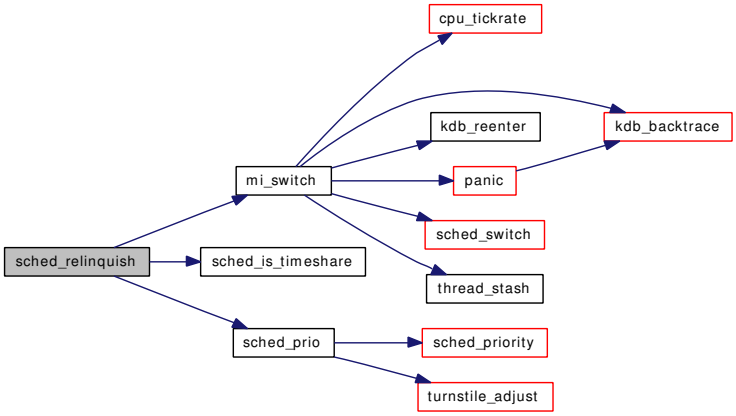


9.81.3.40 void sched_relinquish (struct thread * *td*)

Definition at line 1688 of file sched_core.c.

References mi_switch(), sched_is_timeshare(), sched_lock, sched_prio(), and TSF_NEXTRQ.

Here is the call graph for this function:

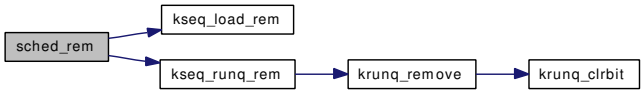


9.81.3.41 void sched_rem (struct thread * *td*)

Definition at line 1601 of file sched_core.c.

References kseq_load_rem(), kseq_runq_rem(), and sched_lock.

Here is the call graph for this function:



9.81.3.42 int sched_rr_interval (void)

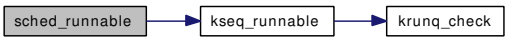
Definition at line 797 of file sched_core.c.

9.81.3.43 int sched_runnable (void)

Definition at line 1324 of file sched_core.c.

References kseq_global, kseq_runnable(), and KSEQ_SELF.

Here is the call graph for this function:

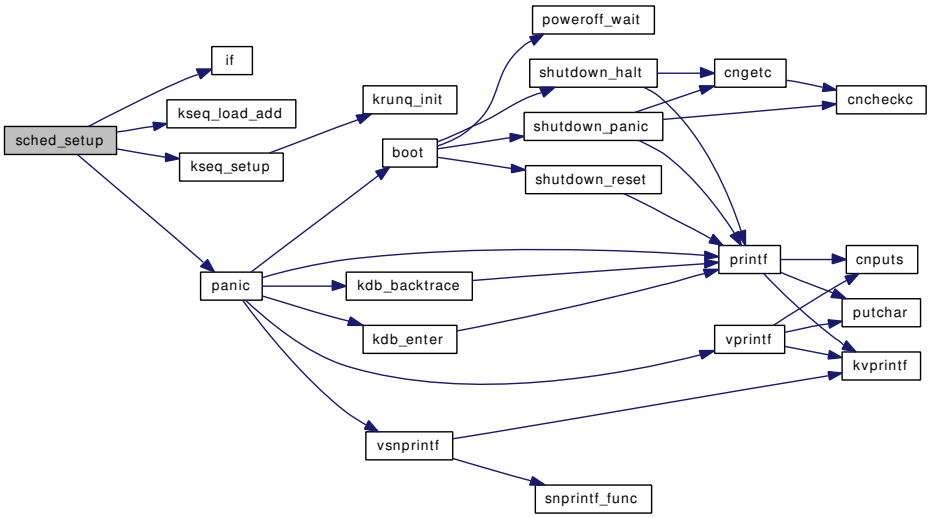


9.81.3.44 static void sched_setup (void * *dummy*) [static]

Definition at line 710 of file sched_core.c.

References hz, if(), kse0, kseq_global, kseq_load_add(), KSEQ_SELF, kseq_setup(), mp_ncpus, panic(), sched_lock, and smp_topology.

Here is the call graph for this function:



9.81.3.45 int sched_sizeof_proc (void)

Definition at line 1701 of file sched_core.c.

9.81.3.46 int sched_sizeof_thread (void)

Definition at line 1707 of file sched_core.c.

9.81.3.47 void sched_sleep (struct thread * *td*)

Definition at line 1050 of file sched_core.c.

References sched_lock, and TSF_SLEEP.

9.81.3.48 static int sched_starving (struct kseq *, unsigned, struct td_sched *) [static]

Definition at line 1173 of file sched_core.c.

References HZ_TO_NS, kseq::ksq_expired_nice, kseq::ksq_expired_tick, kseq::ksq_load, and STARVATION_TIME.

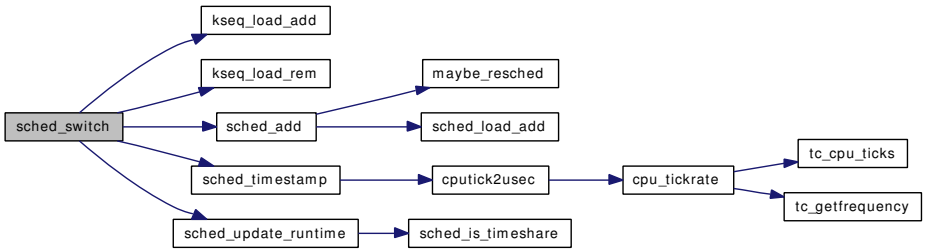
Referenced by sched_tick().

9.81.3.49 void sched_switch (struct thread * *td*, struct thread * *newtd*, int *flags*)

Definition at line 969 of file sched_core.c.

References kseq_load_add(), kseq_load_rem(), KSEQ_SELF, sched_add(), sched_lock, sched_timestamp(), sched_update_runtime(), tick, TSF_DIDRUN, and TSF_NEXTRQ.

Here is the call graph for this function:



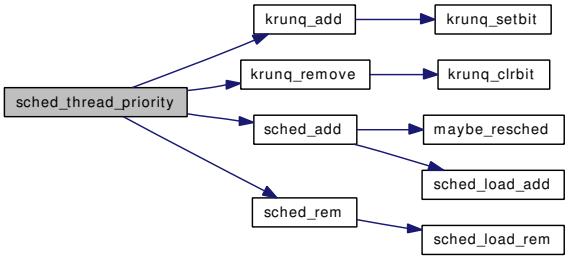
9.81.3.50 static void sched_thread_priority (struct thread *, u_char) [static]

Definition at line 824 of file sched_core.c.

References krunq_add(), krunq_remove(), sched_add(), sched_lock, and sched_rem().

Referenced by sched_lend_prio(), sched_prio(), and sched_unlend_prio().

Here is the call graph for this function:

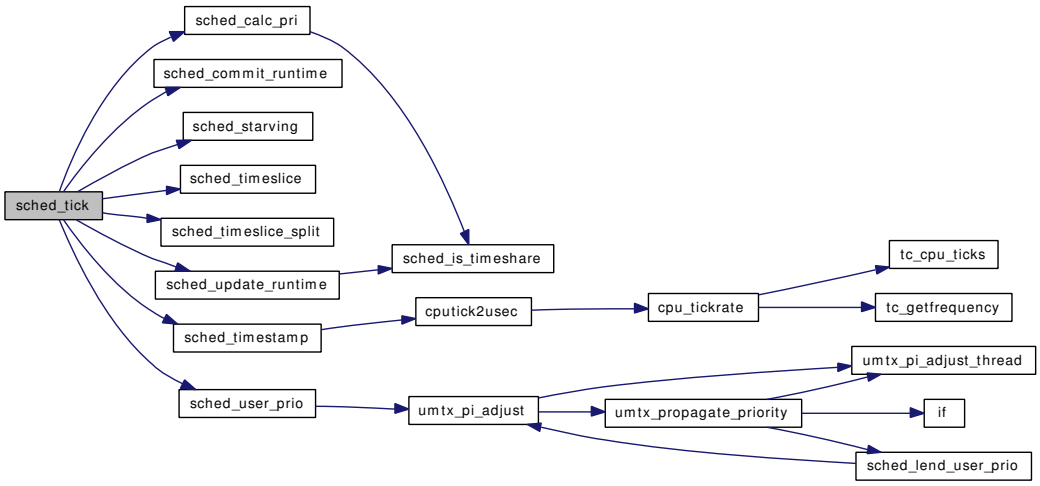


9.81.3.51 void sched_tick (void)

Definition at line 1208 of file sched_core.c.

References KSEQ_CPU, kseq_global, kseq::ksq_expired_nice, kseq::ksq_expired_tick, kseq::ksq_last_timestamp, sched_calc_pri(), sched_commit_runtime(), sched_lock, sched_starving(), sched_timeslice(), sched_timeslice_split(), sched_timestamp(), sched_update_runtime(), sched_user_prio(), td, THREAD_IS_INTERACTIVE, tick, TSF_BOUND, TSF_FIRST_SLICE, and TSF_NEXTRQ.

Here is the call graph for this function:



9.81.3.52 static int sched_timeslice (struct td_sched * *ts*) [inline, static]

Definition at line 582 of file sched_core.c.

References PROC_USER_PRI, and SCALE_USER_PRI.

Referenced by sched_exit_thread(), and sched_tick().

9.81.3.53 static int sched_timeslice_split (struct td_sched * *ts*) [inline, static]

Definition at line 1192 of file sched_core.c.

References CURRENT_SCORE, MAX_SCORE, and smp_cpus.

Referenced by sched_tick().

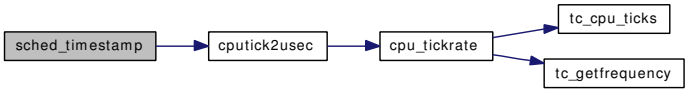
9.81.3.54 static uint64_t sched_timestamp (void) [inline, static]

Definition at line 575 of file sched_core.c.

References cpu_ticks, and cputick2usec().

Referenced by sched_choose(), sched_exit_thread(), sched_switch(), sched_tick(), and sched_wakeup().

Here is the call graph for this function:



9.81.3.55 void sched_unbind (struct thread * *td*)

Definition at line 1668 of file sched_core.c.

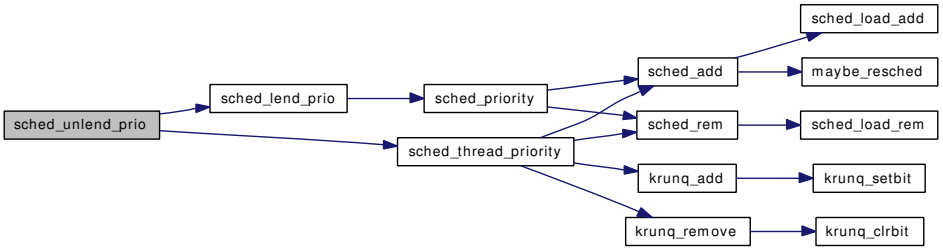
References sched_lock.

9.81.3.56 void sched_unlend_prio (struct thread * *td*, u_char *prio*)

Definition at line 878 of file sched_core.c.

References sched_lend_prio(), and sched_thread_priority().

Here is the call graph for this function:

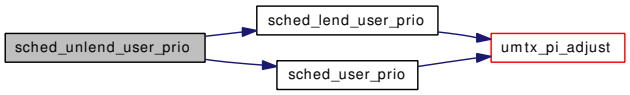


9.81.3.57 void sched_unlend_user_prio (struct thread * *td*, u_char *prio*)

Definition at line 956 of file sched_core.c.

References sched_lend_user_prio(), and sched_user_prio().

Here is the call graph for this function:



9.81.3.58 static void sched_update_runtime (struct td_sched * *ts*, uint64_t *now*) [static]

Definition at line 668 of file sched_core.c.

References CURRENT_SCORE, NS_MAX_SLEEP_TIME, sched_is_timeshare(), and td_sched::ts_thread.

Referenced by sched_exit_thread(), sched_switch(), and sched_tick().

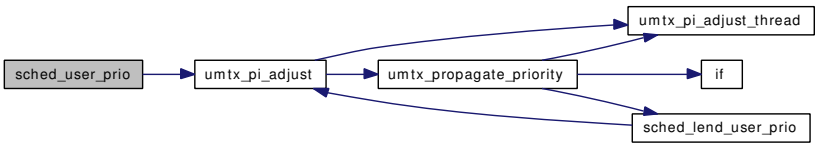
Here is the call graph for this function:

9.81.3.59 void sched_user_prio (struct thread * *td*, u_char *prio*)

Definition at line 925 of file sched_core.c.

References umtx_pi_adjust().

Here is the call graph for this function:



9.81.3.60 void sched_userret (struct thread * *td*)

Definition at line 1334 of file sched_core.c.

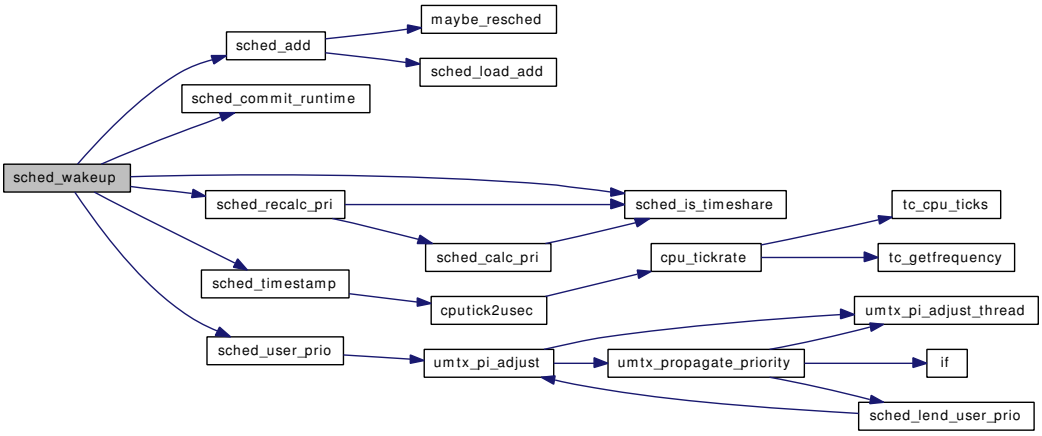
References sched_lock.

9.81.3.61 void sched_wakeup (struct thread * *td*)

Definition at line 1064 of file sched_core.c.

References KSEQ_CPU, KSEQ_SELF, kseq::ksq_last_timestamp, sched_add(), sched_commit_runtime(), sched_is_timeshare(), sched_lock, sched_recalc_pri(), sched_timestamp(), sched_user_prio(), and TSF_-SLEEP.

Here is the call graph for this function:



9.81.3.62 void schedinit (void)

Definition at line 780 of file sched_core.c.

References kse0, proc0, and td_sched::ts_thread.

9.81.3.63 SYSCTL_INT (_kern, OID_AUTO, ccpu, CTLFLAG_RD, & *ccpu*, 0, "")

9.81.3.64 static SYSCTL_NODE (_kern, OID_AUTO, sched, CTLFLAG_RW, 0, "Scheduler") [static]

9.81.3.65 SYSCTL_STRING (_kern_sched, OID_AUTO, name, CTLFLAG_RD, "CORE", 0, "Scheduler name")

9.81.3.66 SYSINIT (sched_setup, SI_SUB_RUN_QUEUE, SI_ORDER_FIRST, sched_setup, NULL)

9.81.3.67 TAILQ_HEAD (krqhead, td_sched)

9.81.4 Variable Documentation

9.81.4.1 fixpt_t ccpu = 0.95122942450071400909 * FSCALE [static]

Definition at line 277 of file sched_core.c.

9.81.4.2 int def_timeslice = 100 [static]

Definition at line 252 of file sched_core.c.

9.81.4.3 int granularity = 10 [static]

Definition at line 253 of file sched_core.c.

9.81.4.4 struct td_sched kse0 [static]

Definition at line 249 of file sched_core.c.

Referenced by sched_setup(), and schedinit().

9.81.4.5 struct kseq kseq_global [static]

Definition at line 256 of file sched_core.c.

Referenced by sched_add(), sched_choose(), sched_runnable(), sched_setup(), and sched_tick().

9.81.4.6 int min_timeslice = 5 [static]

Definition at line 251 of file sched_core.c.

9.81.4.7 int realstathz [static]

Definition at line 254 of file sched_core.c.

Referenced by sched_initticks(), sched_rr_interval(), sched_setup(), and schedcpu().

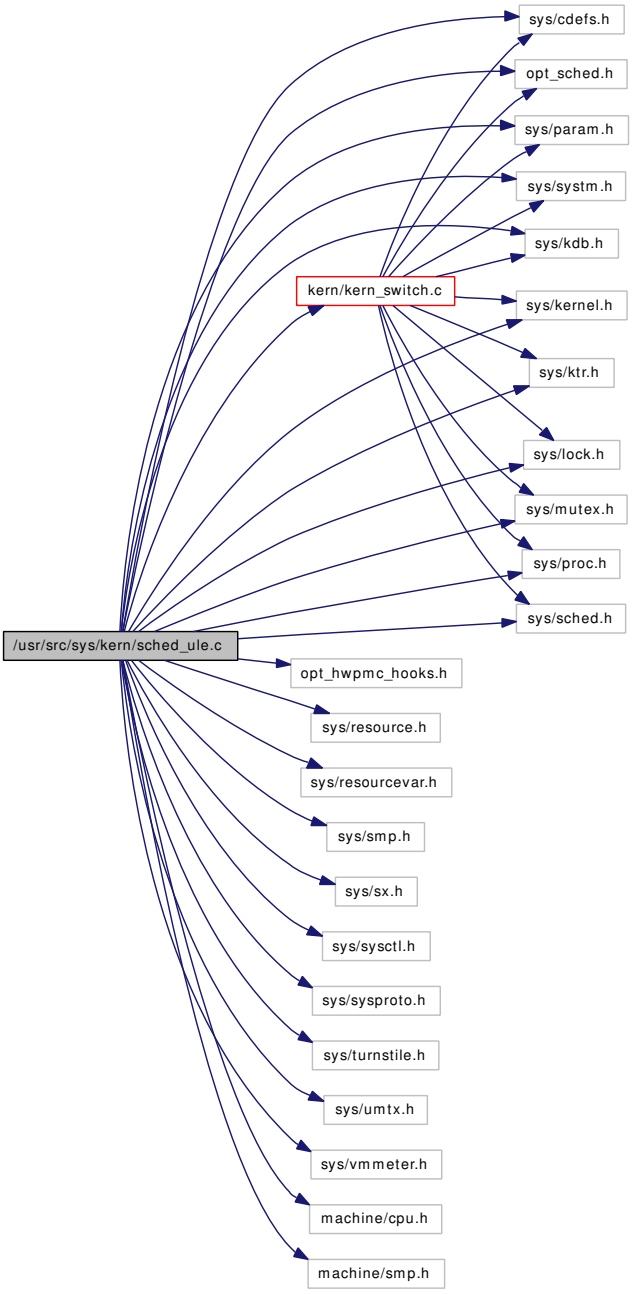
9.81.4.8 int sched_tdcnt [static]

Definition at line 255 of file sched_core.c.

9.82 /usr/src/sys/kern/sched_ule.c File Reference

```
#include <sys/cdefs.h>
#include "opt_hwpmc_hooks.h"
#include "opt_sched.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kdb.h>
#include <sys/kernel.h>
#include <sys/ktr.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/resource.h>
#include <sys/resourcevar.h>
#include <sys/sched.h>
#include <sys/smp.h>
#include <sys/sx.h>
#include <sys/sysctl.h>
#include <sys/sysproto.h>
#include <sys/turnstile.h>
#include <sys/umtx.h>
#include <sys/vmmeter.h>
#include <machine/cpu.h>
#include <machine/smp.h>
#include "kern/kern_switch.c"
```

Include dependency graph for sched_ule.c:



Data Structures

- struct td_sched
- struct tdq

Defines

- #define KTR_ULE 0x0
- #define TSF_BOUND 0x0001

- #define TSF_XFERABLE 0x0002
- #define SCHED_TICK_SECS 10
- #define SCHED_TICK_TARG (hz * SCHED_TICK_SECS)
- #define SCHED_TICK_MAX (SCHED_TICK_TARG + hz)
- #define SCHED_TICK_SHIFT 10
- #define SCHED_TICK_HZ(ts) ((ts)->ts_ticks >> SCHED_TICK_SHIFT)
- #define SCHED_TICK_TOTAL(ts) (max((ts)->ts_ltick - (ts)->ts_ftick, hz))
- #define SCHED_PRI_NRESV (PRIO_MAX - PRIO_MIN)
- #define SCHED_PRI_NHALF (SCHED_PRI_NRESV / 2)
- #define SCHED_PRI_MIN (PRI_MIN_TIMESHARE + SCHED_PRI_NHALF)
- #define SCHED_PRI_MAX (PRI_MAX_TIMESHARE - SCHED_PRI_NHALF)
- #define SCHED_PRI_RANGE (SCHED_PRI_MAX - SCHED_PRI_MIN + 1)
- #define SCHED_PRI_TICKS(ts)
- #define SCHED_PRI_NICE(nice) (nice)
- #define SCHED_SLP_RUN_MAX ((hz * 5) << SCHED_TICK_SHIFT)
- #define SCHED_SLP_RUN_FORK ((hz / 2) << SCHED_TICK_SHIFT)
- #define SCHED_INTERACT_MAX (100)
- #define SCHED_INTERACT_HALF (SCHED_INTERACT_MAX / 2)
- #define SCHED_INTERACT_THRESH (30)
- #define TDQF_BUSY 0x0001
- #define TDQ_SELF() (&tdq_cpu)
- #define TDQ_CPU(x) (&tdq_cpu)
- #define TS_RQ_PPQ (((PRI_MAX_TIMESHARE - PRI_MIN_TIMESHARE) + 1) / RQ_NQS)
- #define KERN_SWITCH_INCLUDE 1

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/sched_ule.c,v 1.187 2007/02/08 01:52:25 jeff Exp $")
- static void sched_priority (struct thread *)
- static void sched_thread_priority (struct thread *, u_char)
- static int sched_interact_score (struct thread *)
- static void sched_interact_update (struct thread *)
- static void sched_interact_fork (struct thread *)
- static void sched_pctcpu_update (struct td_sched *)
- static void sched_pin_td (struct thread *td)
- static void sched_unpin_td (struct thread *td)
- static struct td_sched * tdq_choose (struct tdq *)
- static void tdq_setup (struct tdq *)
- static void tdq_load_add (struct tdq *, struct td_sched *)
- static void tdq_load_rem (struct tdq *, struct td_sched *)
- static __inline void tdq_runq_add (struct tdq *, struct td_sched *, int)
- static __inline void tdq_runq_rem (struct tdq *, struct td_sched *)
- void tdq_print (int cpu)
- static void runq_print (struct runq *rq)
- static void sched_setup (void *dummy)
- static void sched_initticks (void *dummy)
- void schedinit (void)
- int sched_rr_interval (void)
- void sched_lend_prio (struct thread *td, u_char prio)
- void sched_unlend_prio (struct thread *td, u_char prio)

- void sched_prio (struct thread *td, u_char prio)
- void sched_user_prio (struct thread *td, u_char prio)
- void sched_lend_user_prio (struct thread *td, u_char prio)
- void sched_unlend_user_prio (struct thread *td, u_char prio)
- void sched_switch (struct thread *td, struct thread *newtd, int flags)
- void sched_nice (struct proc *p, int nice)
- void sched_sleep (struct thread *td)
- void sched_wakeup (struct thread *td)
- void sched_fork (struct thread *td, struct thread *child)
- void sched_fork_thread (struct thread *td, struct thread *child)
- void sched_class (struct thread *td, int class)
- void sched_exit (struct proc *p, struct thread *child)
- void sched_exit_thread (struct thread *td, struct thread *child)
- void sched_userret (struct thread *td)
- void sched_clock (struct thread *td)
- int sched_runnable (void)
- thread * sched_choose (void)
- static int sched_preempt (struct thread *td)
- void sched_add (struct thread *td, int flags)
- void sched_rem (struct thread *td)
- fixpt_t sched_pctcpu (struct thread *td)
- void sched_bind (struct thread *td, int cpu)
- void sched_unbind (struct thread *td)
- int sched_is_bound (struct thread *td)
- void sched_relinquish (struct thread *td)
- int sched_load (void)
- int sched_sizeof_proc (void)
- int sched_sizeof_thread (void)
- void sched_tick (void)
- void sched_idletd (void *dummy)
- static SYSCTL_NODE (_kern, OID_AUTO, sched, CTLFLAG_RW, 0,"Scheduler")
- SYSCTL_STRING (_kern_sched, OID_AUTO, name, CTLFLAG_RD,"ule", 0,"Scheduler name")
- SYSCTL_INT (_kern_sched, OID_AUTO, slice, CTLFLAG_RW,&sched_slice, 0,"")
- SYSCTL_INT (_kern_sched, OID_AUTO, interact, CTLFLAG_RW,&sched_interact, 0,"")
- SYSCTL_INT (_kern_sched, OID_AUTO, tickincr, CTLFLAG_RD,&tickincr, 0,"")
- SYSCTL_INT (_kern_sched, OID_AUTO, realstathz, CTLFLAG_RD,&realstathz, 0,"")
- SYSCTL_INT (_kern, OID_AUTO, ccpu, CTLFLAG_RD,&ccpu, 0,"")

Variables

- static struct td_sched td_sched0
- static int sched_interact = SCHED_INTERACT_THRESH
- static int realstathz
- static int tickincr
- static int sched_slice
- static struct tdq tdq_cpu
- static fixpt_t ccpu = 0.95122942450071400909 * FSCALE

9.82.1 Define Documentation

9.82.1.1 #define KERN_SWITCH_INCLUDE 1

Definition at line 2100 of file sched_ule.c.

9.82.1.2 #define KTR_ULE 0x0

Definition at line 73 of file sched_ule.c.

Referenced by sched_add().

9.82.1.3 #define SCHED_INTERACT_HALF (SCHED_INTERACT_MAX / 2)

Definition at line 159 of file sched_ule.c.

Referenced by sched_interact_score().

9.82.1.4 #define SCHED_INTERACT_MAX (100)

Definition at line 158 of file sched_ule.c.

9.82.1.5 #define SCHED_INTERACT_THRESH (30)

Definition at line 160 of file sched_ule.c.

9.82.1.6 #define SCHED_PRI_MAX (PRI_MAX_TIMESHARE - SCHED_PRI_NHALF)

Definition at line 137 of file sched_ule.c.

9.82.1.7 #define SCHED_PRI_MIN (PRI_MIN_TIMESHARE + SCHED_PRI_NHALF)

Definition at line 136 of file sched_ule.c.

Referenced by sched_priority().

9.82.1.8 #define SCHED_PRI_NHALF (SCHED_PRI_NRESV / 2)

Definition at line 135 of file sched_ule.c.

9.82.1.9 #define SCHED_PRI_NICE(nice) (nice)

Definition at line 142 of file sched_ule.c.

Referenced by sched_priority().

9.82.1.10 #define SCHED_PRI_NRESV (PRIO_MAX - PRIO_MIN)

Definition at line 134 of file sched_ule.c.

9.82.1.11 #define SCHED_PRI_RANGE (SCHED_PRI_MAX - SCHED_PRI_MIN + 1)

Definition at line 138 of file sched_ule.c.

9.82.1.12 #define SCHED_PRI_TICKS(ts)

Value:

```
(SCHED_TICK_HZ((ts)) /                                          \
    (roundup(SCHED_TICK_TOTAL((ts)), SCHED_PRI_RANGE) / SCHED_PRI_RANGE))
```

Definition at line 139 of file sched_ule.c.

Referenced by sched_priority().

9.82.1.13 #define SCHED_SLP_RUN_FORK ((hz / 2) << SCHED_TICK_SHIFT)

Definition at line 157 of file sched_ule.c.

Referenced by sched_interact_fork().

9.82.1.14 #define SCHED_SLP_RUN_MAX ((hz * 5) << SCHED_TICK_SHIFT)

Definition at line 156 of file sched_ule.c.

Referenced by sched_interact_update().

9.82.1.15 #define SCHED_TICK_HZ(ts) ((ts)→ts_ticks >> SCHED_TICK_SHIFT)

Definition at line 119 of file sched_ule.c.

Referenced by sched_pctcpu().

9.82.1.16 #define SCHED_TICK_MAX (SCHED_TICK_TARG + hz)

Definition at line 117 of file sched_ule.c.

Referenced by sched_pctcpu_update().

9.82.1.17 #define SCHED_TICK_SECS 10

Definition at line 115 of file sched_ule.c.

Referenced by sched_pctcpu().

9.82.1.18 #define SCHED_TICK_SHIFT 10

Definition at line 118 of file sched_ule.c.

Referenced by sched_initticks(), sched_setup(), sched_tick(), and sched_wakeup().

9.82.1.19 #define SCHED_TICK_TARG (hz ∗ SCHED_TICK_SECS)

Definition at line 116 of file sched_ule.c.

Referenced by sched_pctcpu_update().

9.82.1.20 #define SCHED_TICK_TOTAL(ts) (max((ts) → ts_ltick - (ts) → ts_ftick, hz))

Definition at line 120 of file sched_ule.c.

Referenced by sched_pctcpu_update().

9.82.1.21 #define TDQ_CPU(x) (&tdq_cpu)

Definition at line 253 of file sched_ule.c.

Referenced by sched_add(), sched_class(), sched_exit_thread(), sched_rem(), and tdq_print().

9.82.1.22 #define TDQ_SELF() (&tdq_cpu)

Definition at line 252 of file sched_ule.c.

Referenced by sched_add(), sched_choose(), sched_clock(), sched_load(), sched_runnable(), and sched_switch().

9.82.1.23 #define TDQF_BUSY 0x0001

Definition at line 194 of file sched_ule.c.

Referenced by tdq_runq_add(), and tdq_runq_rem().

9.82.1.24 #define TS_RQ_PPQ (((PRI_MAX_TIMESHARE - PRI_MIN_TIMESHARE) + 1) / RQ_NQS)

Referenced by tdq_runq_add().

9.82.1.25 #define TSF_BOUND 0x0001

Definition at line 100 of file sched_ule.c.

9.82.1.26 #define TSF_XFERABLE 0x0002

Definition at line 101 of file sched_ule.c.

Referenced by tdq_runq_add(), and tdq_runq_rem().

9.82.2 Function Documentation

9.82.2.1 __FBSDID ("$FreeBSD: src/sys/kern/sched_ule. *c*, v 1.187 2007/02/08 01:52:25 jeff Exp $")

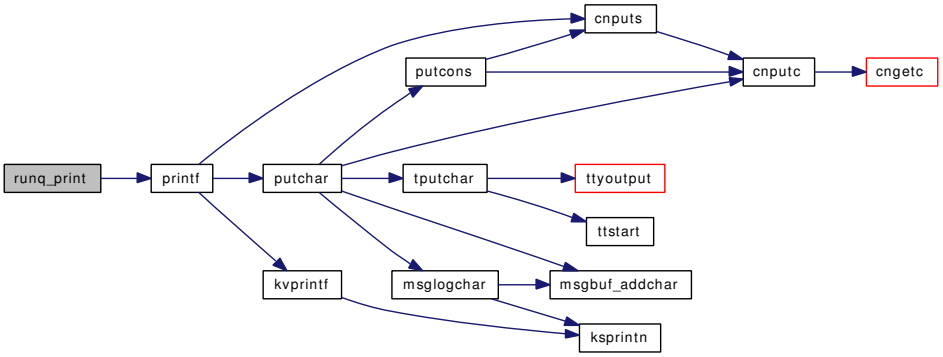
9.82.2.2 static void runq_print (struct runq * *rq*) [static]

Definition at line 310 of file sched_ule.c.

References printf().

Referenced by tdq_print().

Here is the call graph for this function:

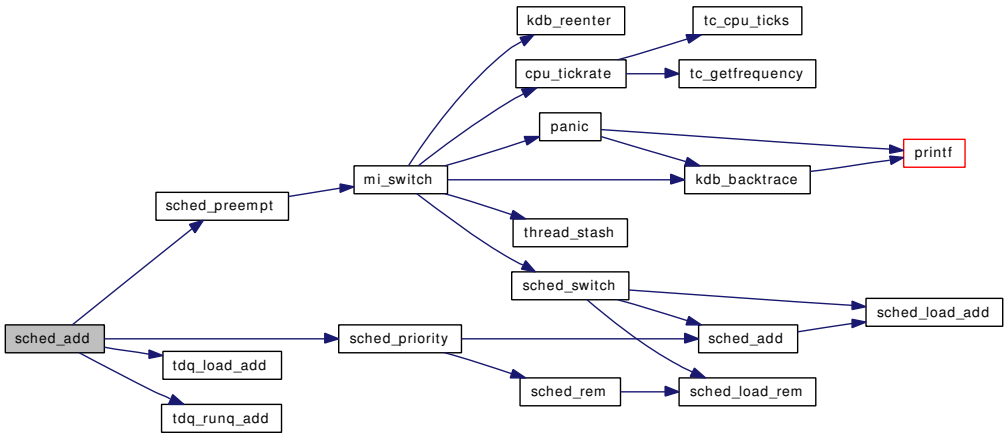


9.82.2.3 void sched_add (struct thread * *td*, int *flags*)

Definition at line 1811 of file sched_ule.c.

References KTR_ULE, sched_lock, sched_preempt(), sched_priority(), sched_slice, TDQ_CPU, tdq::tdq_idle, tdq_load_add(), tdq::tdq_realtime, tdq_runq_add(), TDQ_SELF, and tdq::tdq_timeshare.

Here is the call graph for this function:

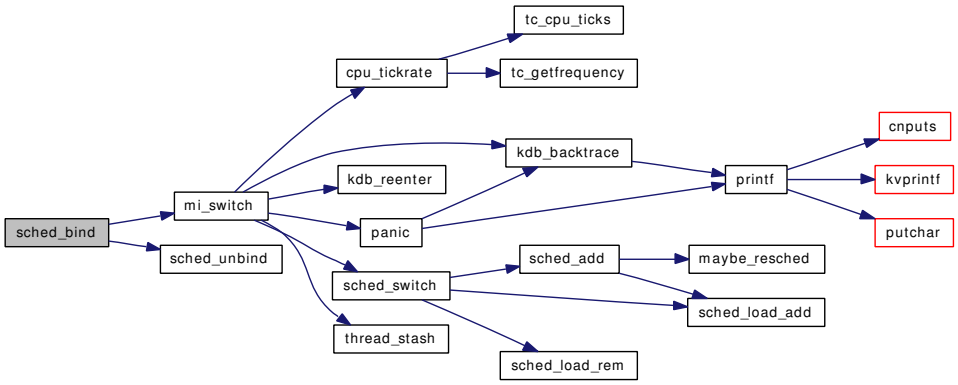


9.82.2.4 void sched_bind (struct thread * *td*, int *cpu*)

Definition at line 1958 of file sched_ule.c.

References mi_switch(), sched_lock, sched_unbind(), and TSF_BOUND.

Here is the call graph for this function:

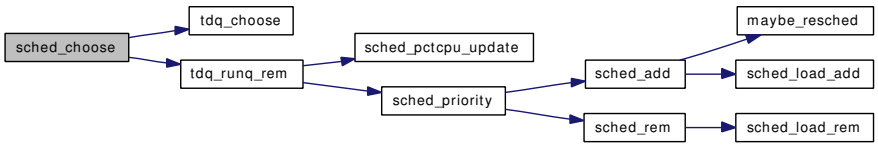


9.82.2.5 struct thread* sched_choose (void)

Definition at line 1748 of file sched_ule.c.

References sched_lock, tdq_choose(), tdq_runq_rem(), TDQ_SELF, and td_sched::ts_thread.

Here is the call graph for this function:



9.82.2.6 void sched_class (struct thread * *td*, int *class*)

Definition at line 1584 of file sched_ule.c.

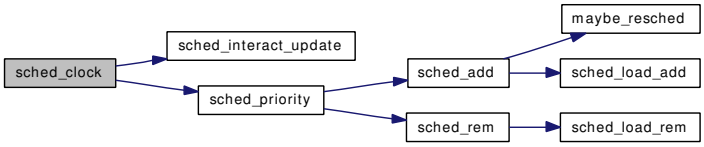
References sched_lock, and TDQ_CPU.

9.82.2.7 void sched_clock (struct thread * *td*)

Definition at line 1680 of file sched_ule.c.

References sched_interact_update(), sched_lock, sched_priority(), tdq::tdq_idx, tdq::tdq_ridx, TDQ_SELF, tdq::tdq_timeshare, and tickincr.

Here is the call graph for this function:

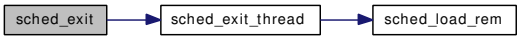


9.82.2.8 void sched_exit (struct proc * *p*, struct thread * *child*)

Definition at line 1619 of file sched_ule.c.

References sched_exit_thread().

Here is the call graph for this function:

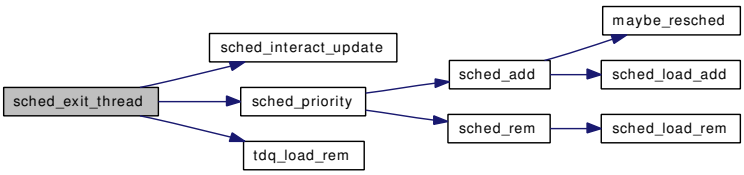


9.82.2.9 void sched_exit_thread (struct thread * *td*, struct thread * *child*)

Definition at line 1631 of file sched_ule.c.

References sched_interact_update(), sched_priority(), TDQ_CPU, and tdq_load_rem().

Here is the call graph for this function:

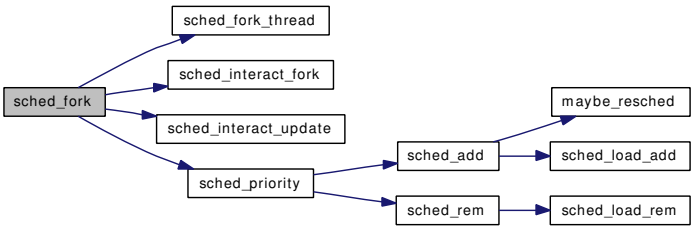


9.82.2.10 void sched_fork (struct thread * *td*, struct thread * *child*)

Definition at line 1539 of file sched_ule.c.

References sched_fork_thread(), sched_interact_fork(), sched_interact_update(), sched_lock, sched_-priority(), and tickincr.

Here is the call graph for this function:



9.82.2.11 void sched_fork_thread (struct thread * *td*, struct thread * *child*)

Definition at line 1554 of file sched_ule.c.

9.82.2.12 void sched_idletd (void * *dummy*)

Definition at line 2058 of file sched_ule.c.

References Giant.

9.82.2.13 static void sched_initticks (void * *dummy*) [static]

Definition at line 1077 of file sched_ule.c.

References realstathz, sched_lock, sched_slice, SCHED_TICK_SHIFT, stathz, and tickincr.

9.82.2.14 static void sched_interact_fork (struct thread *) [static]

Definition at line 1201 of file sched_ule.c.

References SCHED_SLP_RUN_FORK.

Referenced by sched_fork().

9.82.2.15 static int sched_interact_score (struct thread *) [static]

Definition at line 1215 of file sched_ule.c.

References SCHED_INTERACT_HALF.

Referenced by sched_priority().

9.82.2.16 static void sched_interact_update (struct thread *) [static]

Definition at line 1162 of file sched_ule.c.

References SCHED_SLP_RUN_MAX.

Referenced by sched_clock(), sched_exit_thread(), sched_fork(), and sched_wakeup().

9.82.2.17 int sched_is_bound (struct thread * *td*)

Definition at line 1993 of file sched_ule.c.

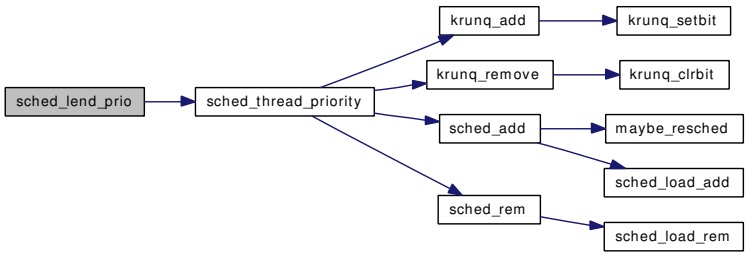
References sched_lock, and TSF_BOUND.

9.82.2.18 void sched_lend_prio (struct thread * *td*, u_char *prio*)

Definition at line 1318 of file sched_ule.c.

References sched_thread_priority().

Here is the call graph for this function:

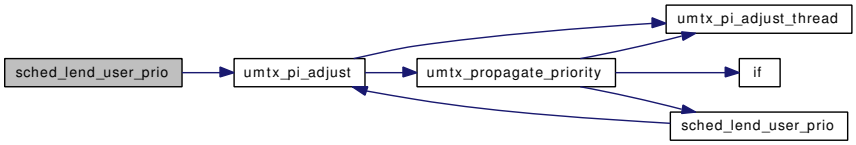


9.82.2.19 void sched_lend_user_prio (struct thread * *td*, u_char *prio*)

Definition at line 1393 of file sched_ule.c.

References umtx_pi_adjust().

Here is the call graph for this function:



9.82.2.20 int sched_load (void)

Definition at line 2010 of file sched_ule.c.

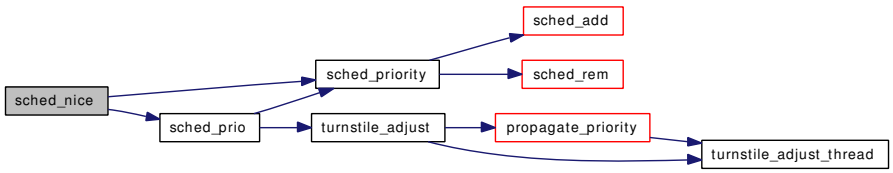
References TDQ_SELF.

9.82.2.21 void sched_nice (struct proc * *p*, int *nice*)

Definition at line 1483 of file sched_ule.c.

References sched_lock, sched_prio(), and sched_priority().

Here is the call graph for this function:

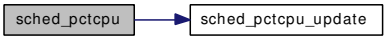


9.82.2.22 fixpt_t sched_pctcpu (struct thread * *td*)

Definition at line 1932 of file sched_ule.c.

References sched_lock, sched_pctcpu_update(), SCHED_TICK_HZ, and SCHED_TICK_SECS.

Here is the call graph for this function:



9.82.2.23 static void sched_pctcpu_update (struct td_sched *) [static]

Definition at line 1266 of file sched_ule.c.

References SCHED_TICK_MAX, SCHED_TICK_TARG, SCHED_TICK_TOTAL, and ticks.

9.82.2.24 static void sched_pin_td (struct thread * *td*) [inline, static]

Definition at line 298 of file sched_ule.c.

Referenced by sched_switch().

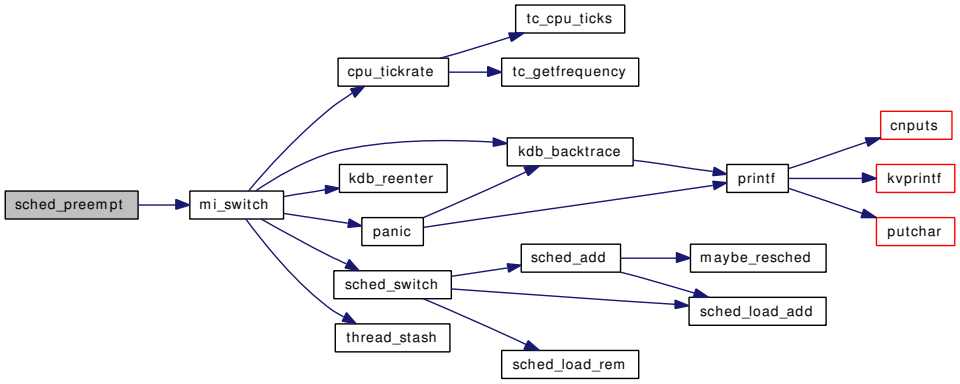
9.82.2.25 static int sched_preempt (struct thread * *td*) [static]

Definition at line 1776 of file sched_ule.c.

References mi_switch(), and panicstr.

Referenced by sched_add().

Here is the call graph for this function:

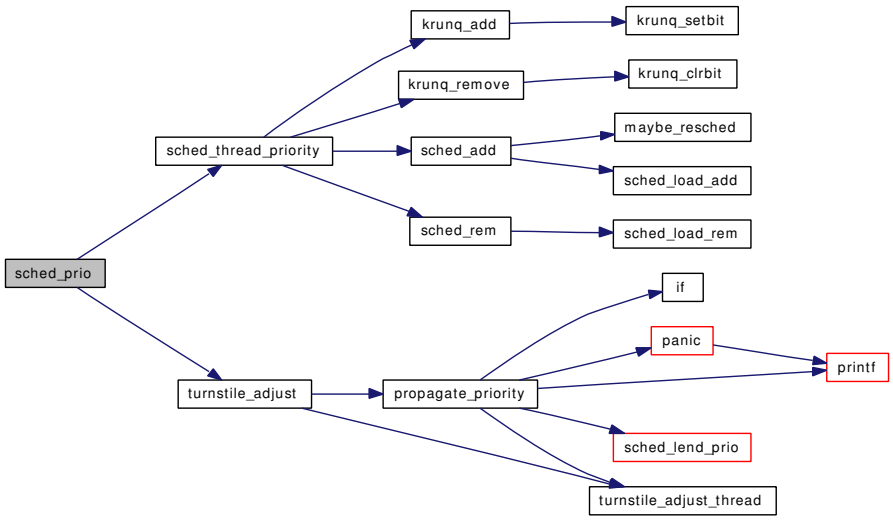


9.82.2.26 void sched_prio (struct thread * *td*, u_char *prio*)

Definition at line 1351 of file sched_ule.c.

References sched_thread_priority(), and turnstile_adjust().

Here is the call graph for this function:

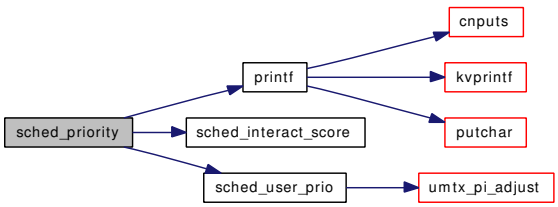


9.82.2.27 static void sched_priority (struct thread *) [static]

Definition at line 1106 of file sched_ule.c.

References printf(), sched_interact, sched_interact_score(), SCHED_PRI_MIN, SCHED_PRI_NICE, SCHED_PRI_TICKS, and sched_user_prio().

Here is the call graph for this function:

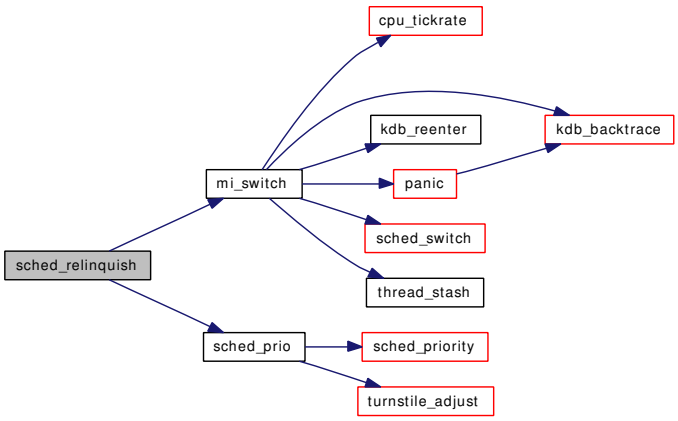


9.82.2.28 void sched_relinquish (struct thread * *td*)

Definition at line 2000 of file sched_ule.c.

References mi_switch(), sched_lock, and sched_prio().

Here is the call graph for this function:

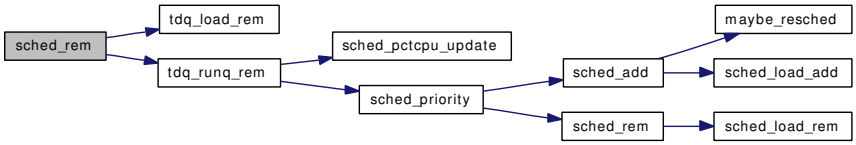


9.82.2.29 void sched_rem (struct thread * *td*)

Definition at line 1912 of file sched_ule.c.

References sched_lock, TDQ_CPU, tdq_load_rem(), and tdq_runq_rem().

Here is the call graph for this function:



9.82.2.30 int sched_rr_interval (void)

Definition at line 1258 of file sched_ule.c.

References realstathz, and sched_slice.

9.82.2.31 int sched_runnable (void)

Definition at line 1724 of file sched_ule.c.

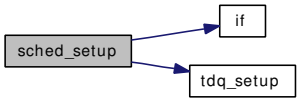
References tdq::tdq_load, and TDQ_SELF.

9.82.2.32 static void sched_setup (void * *dummy*) [static]

Definition at line 975 of file sched_ule.c.

References hz, if(), realstathz, sched_slice, SCHED_TICK_SHIFT, smp_topology, tdq_cpu, tdq_setup(), and tickincr.

Here is the call graph for this function:



9.82.2.33 int sched_sizeof_proc (void)

Definition at line 2026 of file sched_ule.c.

9.82.2.34 int sched_sizeof_thread (void)

Definition at line 2032 of file sched_ule.c.

9.82.2.35 void sched_sleep (struct thread * *td*)

Definition at line 1498 of file sched_ule.c.

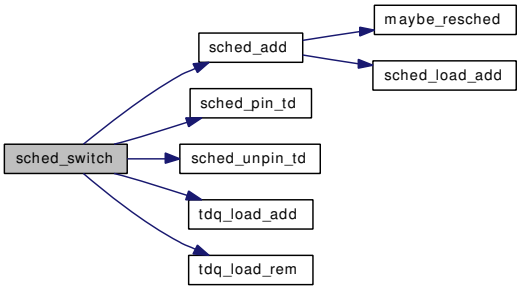
References sched_lock, and ticks.

9.82.2.36 void sched_switch (struct thread * *td*, struct thread * *newtd*, int *flags*)

Definition at line 1420 of file sched_ule.c.

References sched_add(), sched_lock, sched_pin_td(), sched_unpin_td(), tdq_load_add(), tdq_load_rem(), and TDQ_SELF.

Here is the call graph for this function:

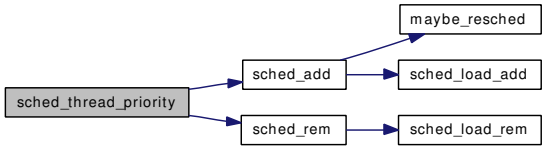


9.82.2.37 static void sched_thread_priority (struct thread *, u_char) [static]

Definition at line 1287 of file sched_ule.c.

References sched_add(), sched_lock, and sched_rem().

Here is the call graph for this function:

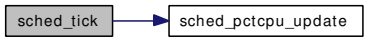


9.82.2.38 void sched_tick (void)

Definition at line 2038 of file sched_ule.c.

References sched_pctcpu_update(), SCHED_TICK_SHIFT, and ticks.

Here is the call graph for this function:



9.82.2.39 void sched_unbind (struct thread * *td*)

Definition at line 1978 of file sched_ule.c.

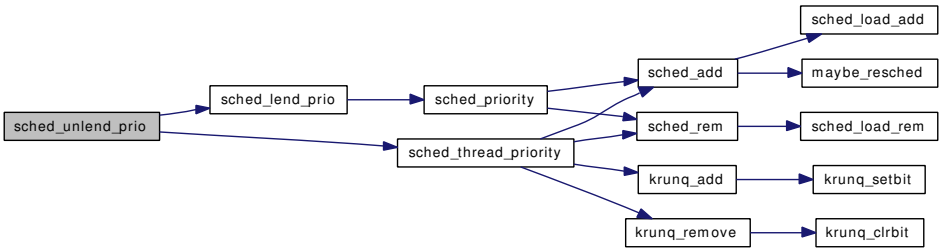
References sched_lock, and TSF_BOUND.

9.82.2.40 void sched_unlend_prio (struct thread * *td*, u_char *prio*)

Definition at line 1334 of file sched_ule.c.

References sched_lend_prio(), and sched_thread_priority().

Here is the call graph for this function:

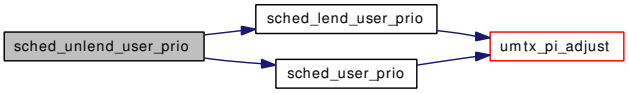


9.82.2.41 void sched_unlend_user_prio (struct thread * *td*, u_char *prio*)

Definition at line 1407 of file sched_ule.c.

References sched_lend_user_prio(), and sched_user_prio().

Here is the call graph for this function:



9.82.2.42 static void sched_unpin_td (struct thread * *td*) [inline, static]

Definition at line 304 of file sched_ule.c.

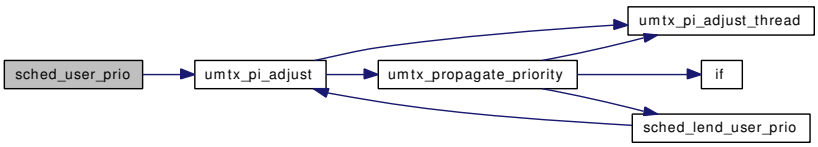
Referenced by sched_switch().

9.82.2.43 void sched_user_prio (struct thread * *td*, u_char *prio*)

Definition at line 1378 of file sched_ule.c.

References umtx_pi_adjust().

Here is the call graph for this function:



9.82.2.44 void sched_userret (struct thread * *td*)

Definition at line 1658 of file sched_ule.c.

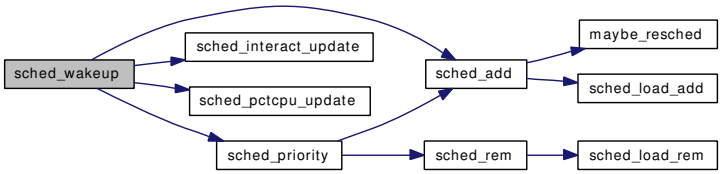
References sched_lock.

9.82.2.45 void sched_wakeup (struct thread * *td*)

Definition at line 1507 of file sched_ule.c.

References sched_add(), sched_interact_update(), sched_lock, sched_pctcpu_update(), sched_priority(), sched_slice, SCHED_TICK_SHIFT, and ticks.

Here is the call graph for this function:



9.82.2.46 void schedinit (void)

Definition at line 1239 of file sched_ule.c.

References proc0, td_sched0, ticks, and td_sched::ts_thread.

9.82.2.47 SYSCTL_INT (_kern, OID_AUTO, ccpu, CTLFLAG_RD, & *ccpu*, 0, "")

9.82.2.48 SYSCTL_INT (_kern_sched, OID_AUTO, realstathz, CTLFLAG_RD, & *realstathz*, 0, "")

9.82.2.49 SYSCTL_INT (_kern_sched, OID_AUTO, tickincr, CTLFLAG_RD, & *tickincr*, 0, "")

9.82.2.50 SYSCTL_INT (_kern_sched, OID_AUTO, interact, CTLFLAG_RW, & *sched_interact*, 0, "")

9.82.2.51 SYSCTL_INT (_kern_sched, OID_AUTO, slice, CTLFLAG_RW, & *sched_slice*, 0, "")

9.82.2.52 static SYSCTL_NODE (_kern, OID_AUTO, sched, CTLFLAG_RW, 0, "Scheduler") [static]

9.82.2.53 SYSCTL_STRING (_kern_sched, OID_AUTO, name, CTLFLAG_RD, "ule", 0, "Scheduler name")

9.82.2.54 static struct td_sched * tdq_choose (struct tdq *) [static]

Definition at line 932 of file sched_ule.c.

References sched_lock, tdq::tdq_idle, tdq::tdq_realtime, tdq::tdq_ridx, tdq::tdq_timeshare, and td_sched::ts_thread.

Referenced by sched_choose().

9.82.2.55 static void tdq_load_add (struct tdq *, struct td_sched *) [static]

Definition at line 431 of file sched_ule.c.

References sched_lock, tdq::tdq_load, and tdq::tdq_sysload.

Referenced by sched_add(), and sched_switch().

9.82.2.56 static void tdq_load_rem (struct tdq *, struct td_sched *) [static]

Definition at line 448 of file sched_ule.c.

References sched_lock, tdq::tdq_load, and tdq::tdq_sysload.

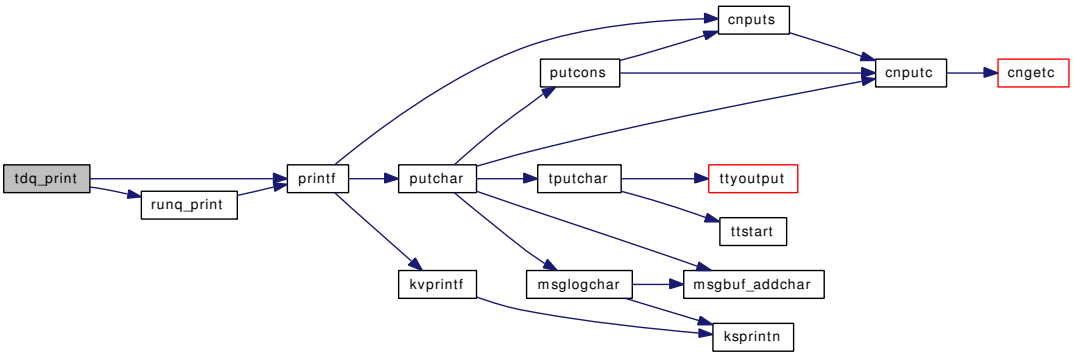
Referenced by sched_exit_thread(), sched_rem(), and sched_switch().

9.82.2.57 void tdq_print (int *cpu*)

Definition at line 334 of file sched_ule.c.

References printf(), runq_print(), TDQ_CPU, tdq::tdq_idle, tdq::tdq_idx, tdq::tdq_load, tdq::tdq_realtime, tdq::tdq_ridx, and tdq::tdq_timeshare.

Here is the call graph for this function:



9.82.2.58 static __inline void tdq_runq_add (struct tdq *, struct td_sched *, int) [static]

Definition at line 356 of file sched_ule.c.

References tdq::tdq_flags, tdq::tdq_idx, tdq::tdq_ridx, tdq::tdq_timeshare, TDQF_BUSY, TS_RQ_PPQ, td_sched::ts_thread, and TSF_XFERABLE.

Referenced by sched_add().

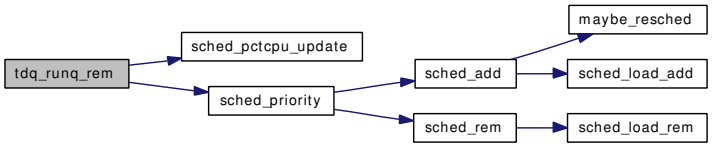
9.82.2.59 static __inline void tdq_runq_rem (struct tdq *, struct td_sched *) [static]

Definition at line 400 of file sched_ule.c.

References sched_pctcpu_update(), sched_priority(), tdq::tdq_flags, tdq::tdq_idx, tdq::tdq_ridx, tdq::tdq_timeshare, TDQF_BUSY, ticks, td_sched::ts_thread, and TSF_XFERABLE.

Referenced by sched_choose(), and sched_rem().

Here is the call graph for this function:



9.82.2.60 static void tdq_setup (struct tdq *) [static]

Definition at line 966 of file sched_ule.c.

References tdq::tdq_idle, tdq::tdq_load, tdq::tdq_realtime, and tdq::tdq_timeshare.

Referenced by sched_setup().

9.82.3 Variable Documentation

9.82.3.1 fixpt_t ccpu = 0.95122942450071400909 * FSCALE [static]

Definition at line 2096 of file sched_ule.c.

9.82.3.2 int realstathz [static]

Definition at line 170 of file sched_ule.c.

9.82.3.3 int sched_interact = SCHED_INTERACT_THRESH [static]

Definition at line 169 of file sched_ule.c.

Referenced by sched_priority().

9.82.3.4 int sched_slice [static]

Definition at line 172 of file sched_ule.c.

Referenced by sched_add(), sched_initticks(), sched_rr_interval(), sched_setup(), and sched_wakeup().

9.82.3.5 struct td_sched td_sched0 [static]

Definition at line 103 of file sched_ule.c.

9.82.3.6 struct tdq tdq_cpu [static]

Definition at line 250 of file sched_ule.c.

Referenced by sched_setup().

9.82.3.7 int tickincr [static]

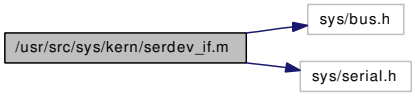
Definition at line 171 of file sched_ule.c.

Referenced by sched_clock(), sched_fork(), sched_initticks(), and sched_setup().

9.83 /usr/src/sys/kern/serdev_if.m File Reference

```
#include <sys/bus.h>
#include <sys/serial.h>
```

Include dependency graph for serdev_if.m:



Variables

- INTERFACE serdev

9.83.1 Variable Documentation

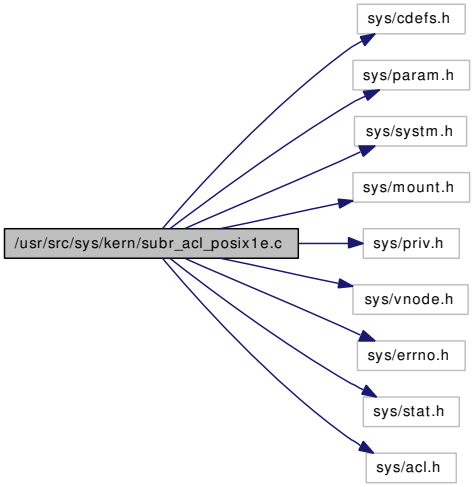
9.83.1.1 INTERFACE serdev

Definition at line 42 of file serdev_if.m.

9.84 /usr/src/sys/kern/subr_acl_posix1e.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/mount.h>
#include <sys/priv.h>
#include <sys/vnode.h>
#include <sys/errno.h>
#include <sys/stat.h>
#include <sys/acl.h>
```

Include dependency graph for subr_acl_posix1e.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_acl_posix1e.c,v 1.51 2006/11/06 13:42:01 rwatson Exp $")
- int vaccess_acl_posix1e (enum vtype type, uid_t file_uid, gid_t file_gid, struct acl *acl, mode_t acc_mode, struct ucred *cred, int *privused)
- acl_perm_t acl_posix1e_mode_to_perm (acl_tag_t tag, mode_t mode)
- acl_entry acl_posix1e_mode_to_entry (acl_tag_t tag, uid_t uid, gid_t gid, mode_t mode)
- mode_t acl_posix1e_perms_to_mode (struct acl_entry *acl_user_obj_entry, struct acl_entry *acl_-group_obj_entry, struct acl_entry *acl_other_entry)
- mode_t acl_posix1e_acl_to_mode (struct acl *acl)
- int acl_posix1e_check (struct acl *acl)
- mode_t acl_posix1e_newfilemode (mode_t cmode, struct acl *dacl)

9.84.1 Function Documentation

9.84.1.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_acl_posix1e. *c*, v 1.51 2006/11/06 13:42:01 rwatson Exp $")

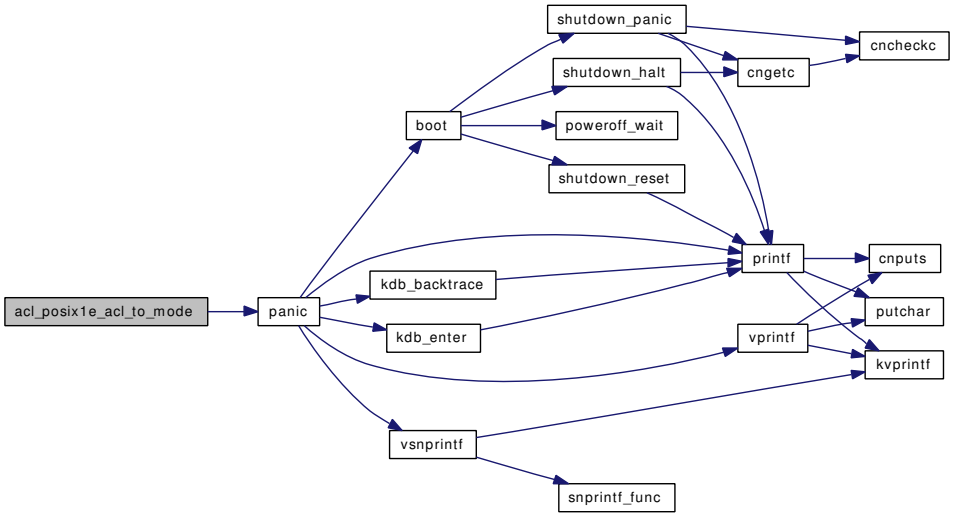
9.84.1.2 mode_t acl_posix1e_acl_to_mode (struct acl * *acl*)

Definition at line 473 of file subr_acl_posix1e.c.

References panic().

Referenced by acl_posix1e_newfilemode().

Here is the call graph for this function:



9.84.1.3 int acl_posix1e_check (struct acl * *acl*)

Definition at line 531 of file subr_acl_posix1e.c.

References if().

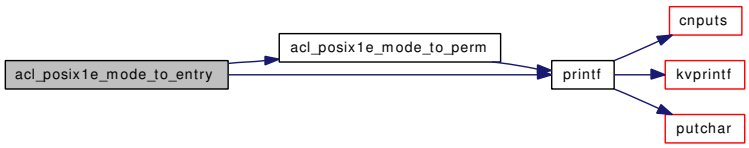
Here is the call graph for this function:

9.84.1.4 struct acl_entry acl_posix1e_mode_to_entry (acl_tag_t *tag*, uid_t *uid*, gid_t *gid*, mode_t *mode*)

Definition at line 408 of file subr_acl_posix1e.c.

References acl_posix1e_mode_to_perm(), and printf().

Here is the call graph for this function:



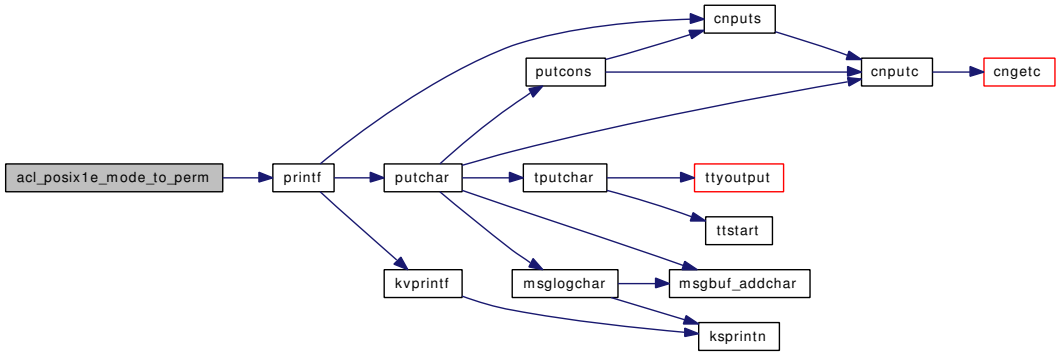
9.84.1.5 acl_perm_t acl_posix1e_mode_to_perm (acl_tag_t *tag*, mode_t *mode*)

Definition at line 365 of file subr_acl_posix1e.c.

References printf().

Referenced by acl_posix1e_mode_to_entry().

Here is the call graph for this function:

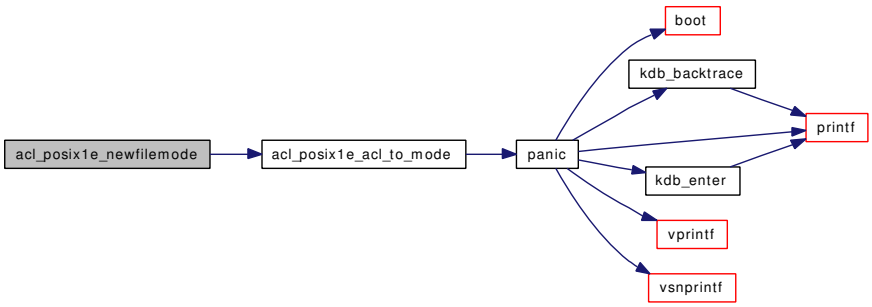


9.84.1.6 mode_t acl_posix1e_newfilemode (mode_t *cmode*, struct acl * *dacl*)

Definition at line 624 of file subr_acl_posix1e.c.

References acl_posix1e_acl_to_mode().

Here is the call graph for this function:



9.84.1.7 mode_t acl_posix1e_perms_to_mode (struct acl_entry * *acl_user_obj_entry*, struct acl_entry * *acl_group_obj_entry*, struct acl_entry * *acl_other_entry*)

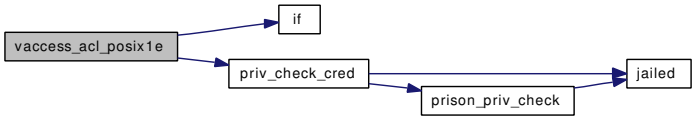
Definition at line 439 of file subr_acl_posix1e.c.

9.84.1.8 int vaccess_acl_posix1e (enum vtype *type*, uid_t *file_uid*, gid_t *file_gid*, struct acl * *acl*, mode_t *acc_mode*, struct ucred * *cred*, int * *privused*)

Definition at line 55 of file subr_acl_posix1e.c.

References if(), and priv_check_cred().

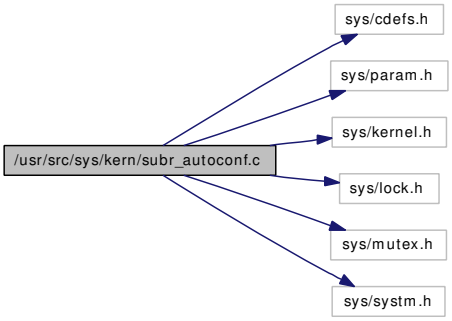
Here is the call graph for this function:



9.85 /usr/src/sys/kern/subr_autoconf.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/systm.h>
```

Include dependency graph for subr_autoconf.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_autoconf.c,v 1.23 2006/07/19 18:53:56 jhb Exp $")
- static TAILQ_HEAD (MTX_SYSINIT(intr_config_hook)
- SYSINIT (intr_config_hooks, SI_SUB_INT_CONFIG_HOOKS, SI_ORDER_FIRST, run_interrupt_driven_config_hooks, NULL)
- void config_intrhook_disestablish (struct intr_config_hook *hook)

9.85.1 Function Documentation

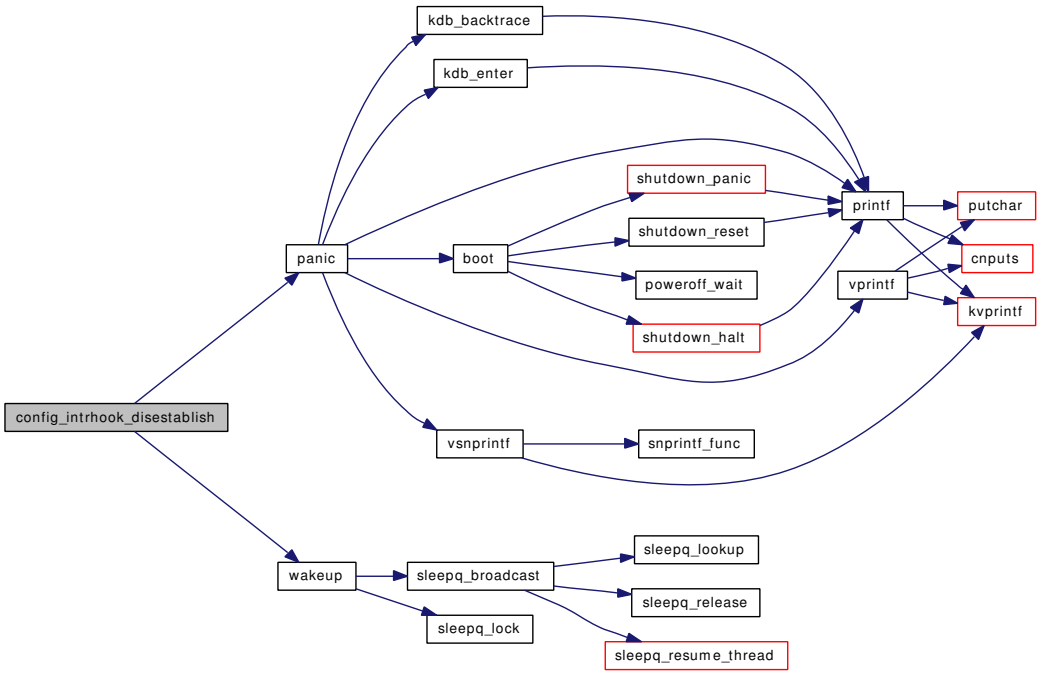
9.85.1.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_autoconf. *c*, v 1.23 2006/07/19 18:53:56 jhb Exp $")

9.85.1.2 void config_intrhook_disestablish (struct intr_config_hook * *hook*)

Definition at line 115 of file subr_autoconf.c.

References panic(), and wakeup().

Here is the call graph for this function:

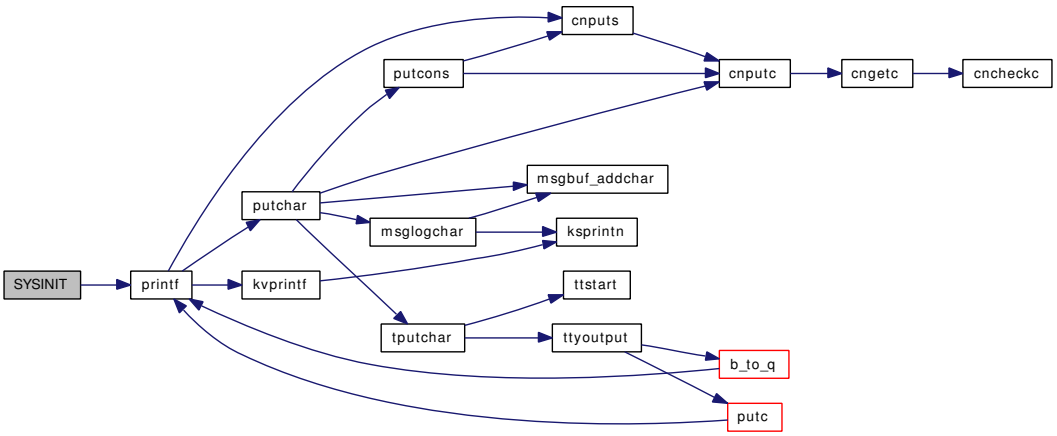


9.85.1.3 SYSINIT (intr_config_hooks, SI_SUB_INT_CONFIG_HOOKS, SI_ORDER_FIRST, run_interrupt_driven_config_hooks, NULL)

Definition at line 82 of file subr_autoconf.c.

References printf().

Here is the call graph for this function:

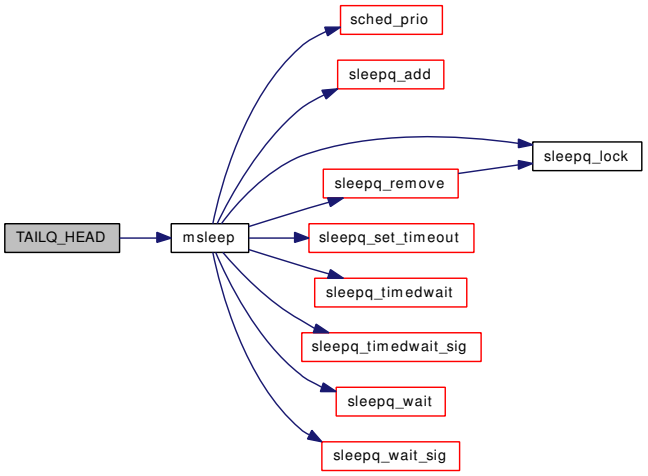


9.85.1.4 static TAILQ_HEAD (MTX_SYSINIT(*intr_config_hook*) [static]

Definition at line 53 of file subr_autoconf.c.

References msleep().

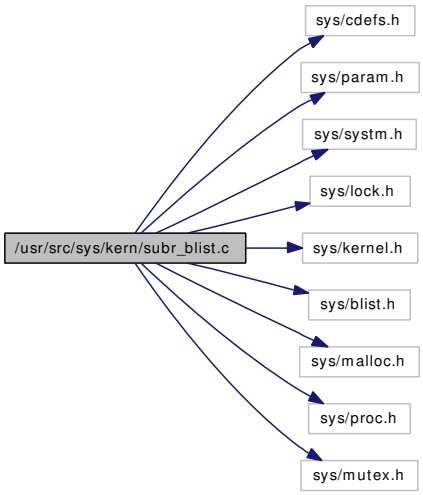
Here is the call graph for this function:



9.86 /usr/src/sys/kern/subr_blist.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/lock.h>
#include <sys/kernel.h>
#include <sys/blist.h>
#include <sys/malloc.h>
#include <sys/proc.h>
#include <sys/mutex.h>
```

Include dependency graph for subr_blist.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_blist.c,v 1.17 2004/06/04 04:03:25 alc Exp $")
- static daddr_t blst_leaf_alloc (blmeta_t *scan, daddr_t blk, int count)
- static daddr_t blst_meta_alloc (blmeta_t *scan, daddr_t blk, daddr_t count, daddr_t radix, int skip)
- static void blst_leaf_free (blmeta_t *scan, daddr_t relblk, int count)
- static void blst_meta_free (blmeta_t *scan, daddr_t freeBlk, daddr_t count, daddr_t radix, int skip, daddr_t blk)
- static void blst_copy (blmeta_t *scan, daddr_t blk, daddr_t radix, daddr_t skip, blist_t dest, daddr_t count)
- static int blst_leaf_fill (blmeta_t *scan, daddr_t blk, int count)
- static int blst_meta_fill (blmeta_t *scan, daddr_t allocBlk, daddr_t count, daddr_t radix, int skip, daddr_t blk)
- static daddr_t blst_radix_init (blmeta_t *scan, daddr_t radix, int skip, daddr_t count)
- static MALLOC_DEFINE (M_SWAP,"SWAP","Swap space")
- blist_t blist_create (daddr_t blocks)
- void blist_destroy (blist_t bl)

- daddr_t blist_alloc (blist_t bl, daddr_t count)
- void blist_free (blist_t bl, daddr_t blkno, daddr_t count)
- int blist_fill (blist_t bl, daddr_t blkno, daddr_t count)
- void blist_resize (blist_t *pbl, daddr_t count, int freenew)

9.86.1 Function Documentation

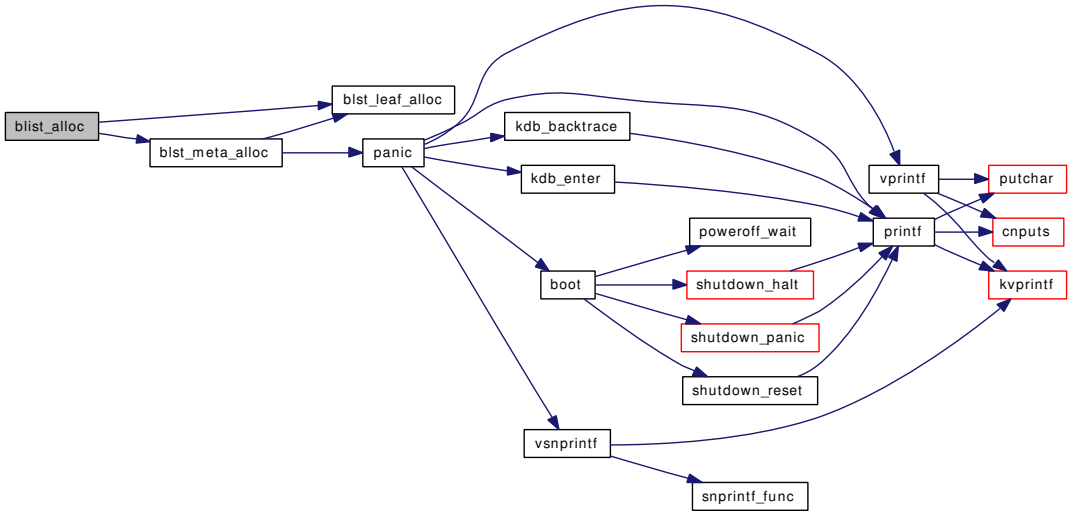
9.86.1.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_blist. *c*, v 1.17 2004/06/04 04:03:25 alc Exp $")

9.86.1.2 daddr_t blist_alloc (blist_t *bl*, daddr_t *count*)

Definition at line 217 of file subr_blist.c.

References blst_leaf_alloc(), and blst_meta_alloc().

Here is the call graph for this function:



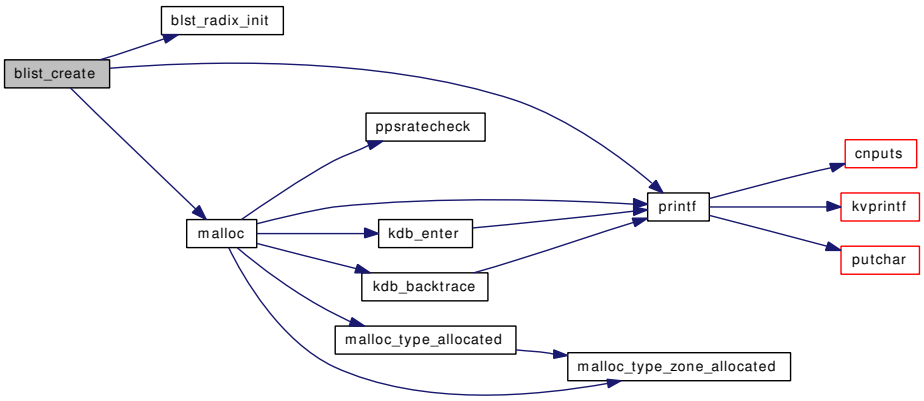
9.86.1.3 blist_t blist_create (daddr_t *blocks*)

Definition at line 162 of file subr_blist.c.

References blst_radix_init(), malloc(), and printf().

Referenced by blist_resize().

Here is the call graph for this function:



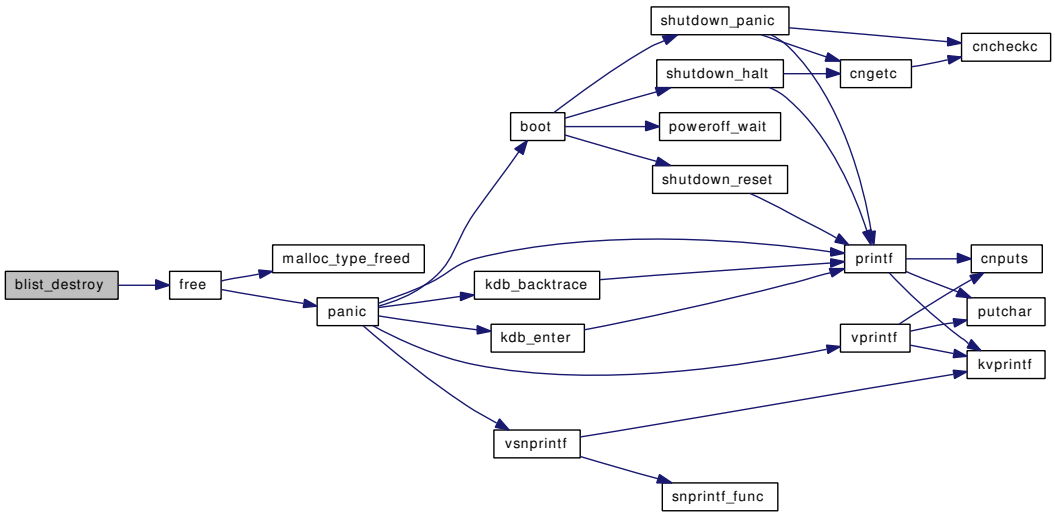
9.86.1.4 void blist_destroy (blist_t *bl*)

Definition at line 204 of file subr_blist.c.

References free().

Referenced by blist_resize().

Here is the call graph for this function:

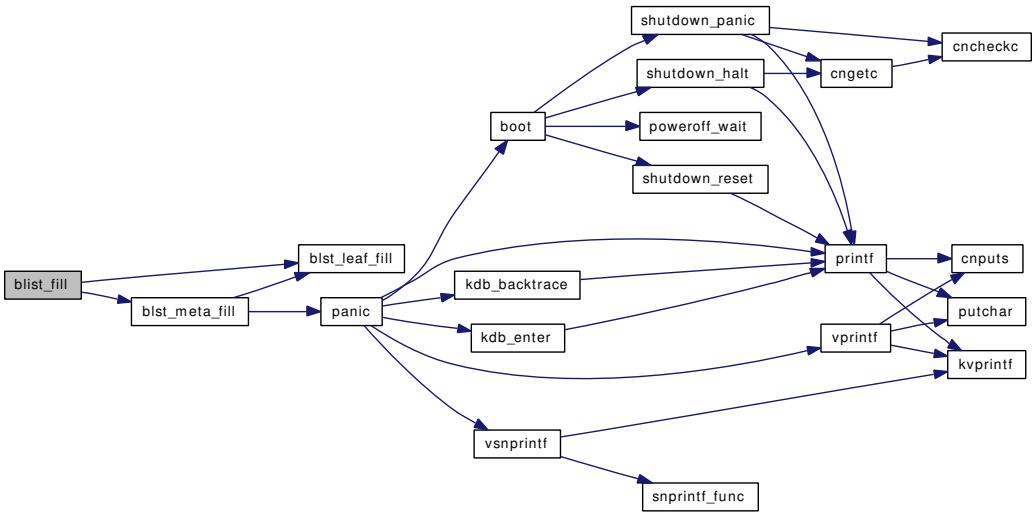


9.86.1.5 int blist_fill (blist_t *bl*, daddr_t *blkno*, daddr_t *count*)

Definition at line 258 of file subr_blist.c.

References blst_leaf_fill(), and blst_meta_fill().

Here is the call graph for this function:



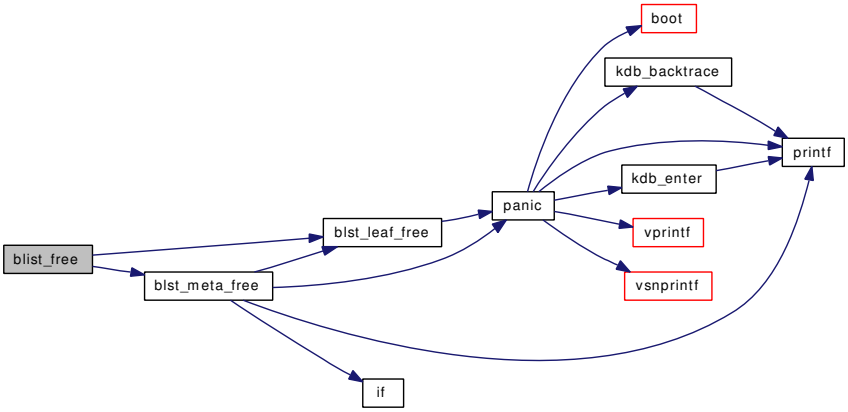
9.86.1.6 void blist_free (blist_t *bl*, daddr_t *blkno*, daddr_t *count*)

Definition at line 239 of file subr_blist.c.

References blst_leaf_free(), and blst_meta_free().

Referenced by blist_resize(), and blst_copy().

Here is the call graph for this function:

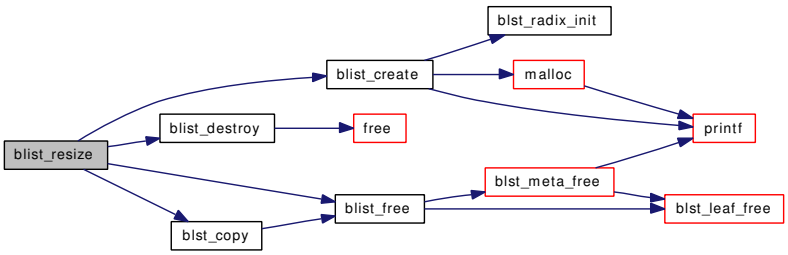


9.86.1.7 void blist_resize (blist_t * *pbl*, daddr_t *count*, int *freenew*)

Definition at line 283 of file subr_blist.c.

References blist_create(), blist_destroy(), blist_free(), and blst_copy().

Here is the call graph for this function:



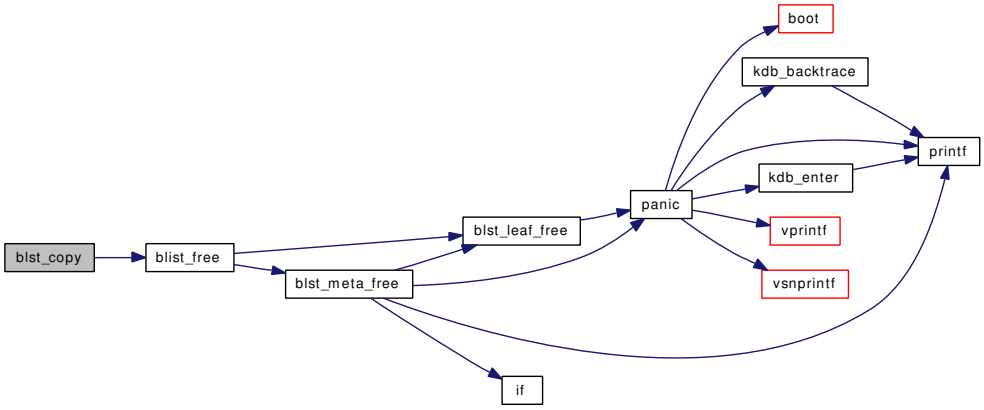
9.86.1.8 static void blst_copy (blmeta_t * *scan*, daddr_t *blk*, daddr_t *radix*, daddr_t *skip*, blist_t *dest*, daddr_t *count*) [static]

Definition at line 640 of file subr_blist.c.

References blist_free().

Referenced by blist_resize().

Here is the call graph for this function:



9.86.1.9 static daddr_t blst_leaf_alloc (blmeta_t * *scan*, daddr_t *blk*, int *count*) [static]

Definition at line 338 of file subr_blist.c.

References mask.

Referenced by blist_alloc(), and blst_meta_alloc().

9.86.1.10 static int blst_leaf_fill (blmeta_t * *scan*, daddr_t *blk*, int *count*) [static]

Definition at line 736 of file subr_blist.c.

References mask.

Referenced by blist_fill(), and blst_meta_fill().

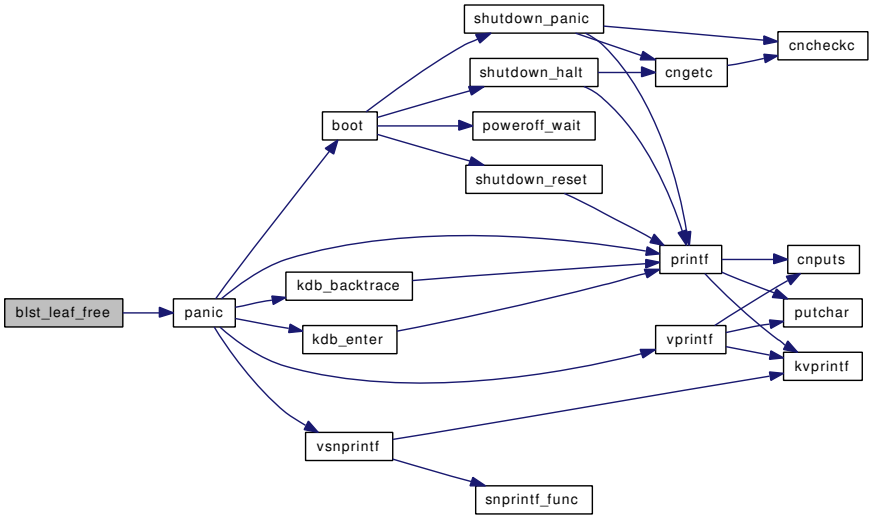
9.86.1.11 static void blst_leaf_free (blmeta_t * *scan*, daddr_t *relblk*, int *count*) [static]

Definition at line 500 of file subr_blist.c.

References mask, and panic().

Referenced by blist_free(), and blst_meta_free().

Here is the call graph for this function:



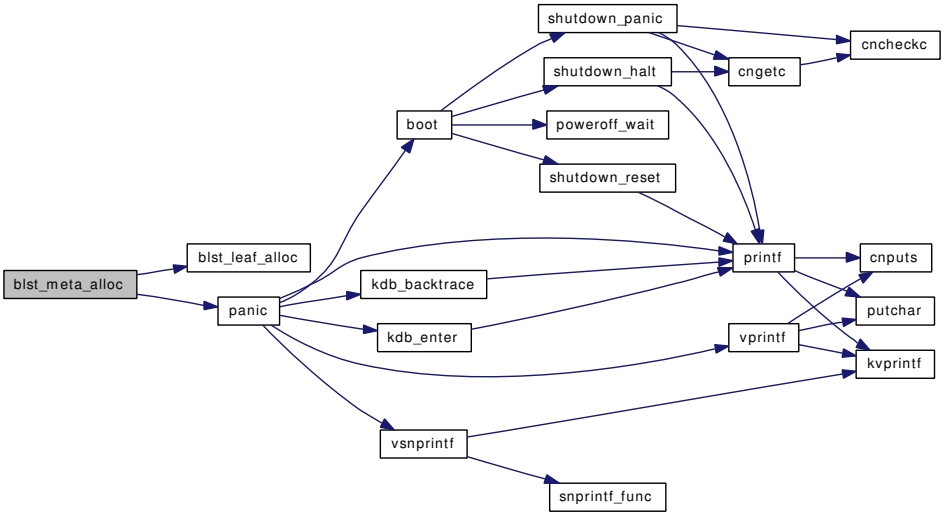
9.86.1.12 static daddr_t blst_meta_alloc (blmeta_t * *scan*, daddr_t *blk*, daddr_t *count*, daddr_t *radix*, int *skip*) [static]

Definition at line 414 of file subr_blist.c.

References blst_leaf_alloc(), and panic().

Referenced by blist_alloc().

Here is the call graph for this function:



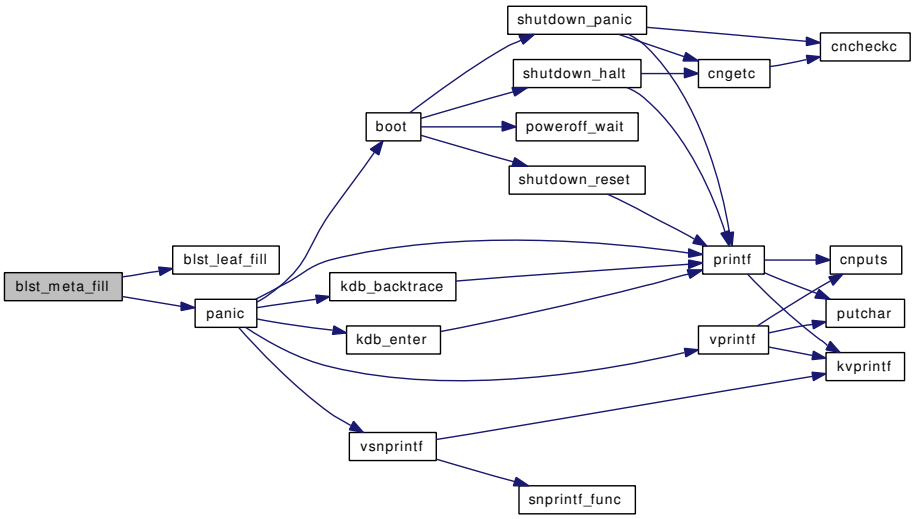
9.86.1.13 static int blst_meta_fill (blmeta_t * *scan*, daddr_t *allocBlk*, daddr_t *count*, daddr_t *radix*, int *skip*, daddr_t *blk*) [static]

Definition at line 763 of file subr_blist.c.

References blst_leaf_fill(), and panic().

Referenced by blist_fill().

Here is the call graph for this function:



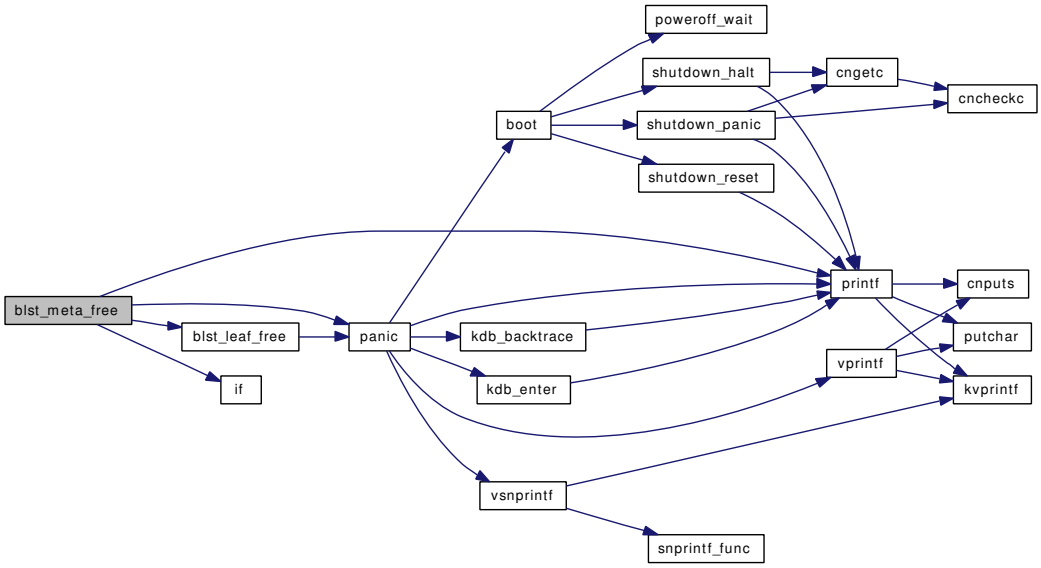
9.86.1.14 static void blst_meta_free (blmeta_t * *scan*, daddr_t *freeBlk*, daddr_t *count*, daddr_t *radix*, int *skip*, daddr_t *blk*) [static]

Definition at line 544 of file subr_blist.c.

References blst_leaf_free(), if(), panic(), and printf().

Referenced by blist_free().

Here is the call graph for this function:



9.86.1.15 static daddr_t blst_radix_init (blmeta_t * *scan*, daddr_t *radix*, int *skip*, daddr_t *count*) [static]

Definition at line 848 of file subr_blist.c.

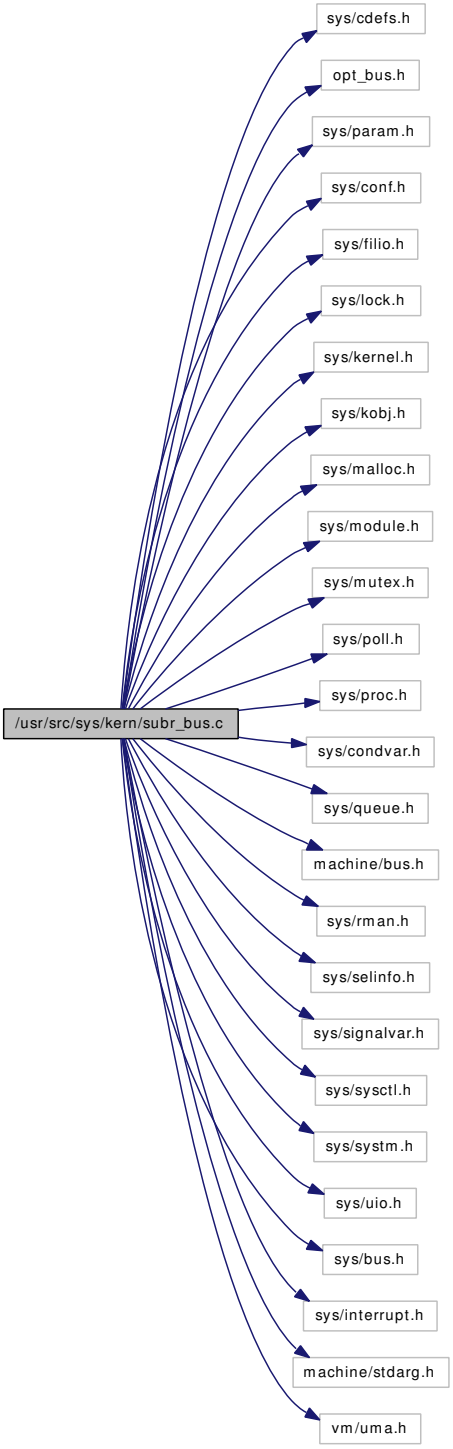
Referenced by blist_create().

9.86.1.16 static MALLOC_DEFINE (M_SWAP, "SWAP", "Swap space") [static]

9.87 /usr/src/sys/kern/subr_bus.c File Reference

```
#include <sys/cdefs.h>
#include "opt_bus.h"
#include <sys/param.h>
#include <sys/conf.h>
#include <sys/filio.h>
#include <sys/lock.h>
#include <sys/kernel.h>
#include <sys/kobj.h>
#include <sys/malloc.h>
#include <sys/module.h>
#include <sys/mutex.h>
#include <sys/poll.h>
#include <sys/proc.h>
#include <sys/condvar.h>
#include <sys/queue.h>
#include <machine/bus.h>
#include <sys/rman.h>
#include <sys/selinfo.h>
#include <sys/signalvar.h>
#include <sys/sysctl.h>
#include <sys/systm.h>
#include <sys/uio.h>
#include <sys/bus.h>
#include <sys/interrupt.h>
#include <machine/stdarg.h>
#include <vm/uma.h>
```

Include dependency graph for subr_bus.c:



Data Structures

- struct driverlink
- struct device

  Implementation of device.

- struct dev_event_info
- struct dev_softc

Defines

- #define DF_ENABLED 1
- #define DF_FIXEDCLASS 2
- #define DF_WILDCARD 4
- #define DF_DESCMALLOCED 8
- #define DF_QUIET 16
- #define DF_DONENOMATCH 32
- #define DF_EXTERNALSOFTC 64
- #define DF_REBID 128
- #define PDEBUG(a)
- #define DEVICENAME(d)
- #define DRIVERNAME(d)
- #define DEVCLANAME(d)
- #define print_device_short(d, i)
- #define print_device(d, i)
- #define print_device_tree_short(d, i)
- #define print_device_tree(d, i)
- #define print_driver_short(d, i)
- #define print_driver(d, i)
- #define print_driver_list(d, i)
- #define print_devclass_short(d, i)
- #define print_devclass(d, i)
- #define print_devclass_list_short()
- #define print_devclass_list()

Typedefs

- typedef driverlink * driverlink_t

Enumerations

- enum { DEVCLASS_SYSCTL_PARENT }
- enum {

  DEVICE_SYSCTL_DESC, DEVICE_SYSCTL_DRIVER, DEVICE_SYSCTL_LOCATION, DEVICE_SYSCTL_PNPINFO,

  DEVICE_SYSCTL_PARENT }

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_bus.c,v 1.198 2007/02/23 12:19:01 piso Exp $")
- SYSCTL_NODE (_hw, OID_AUTO, bus, CTLFLAG_RW, NULL, NULL)
- SYSCTL_NODE (, OID_AUTO, dev, CTLFLAG_RW, NULL, NULL)
- typedef TAILQ_HEAD (devclass_list, devclass)
- static MALLOC_DEFINE (M_BUS,"bus","Bus data structures")
- static MALLOC_DEFINE (M_BUS_SC,"bus-sc","Bus data structures, softc")
- static int devclass_sysctl_handler (SYSCTL_HANDLER_ARGS)
- static void devclass_sysctl_init (devclass_t dc)
- static int device_sysctl_handler (SYSCTL_HANDLER_ARGS)
- static void device_sysctl_init (device_t dev)
- static void device_sysctl_fini (device_t dev)
- static int sysctl_devctl_disable (SYSCTL_HANDLER_ARGS)
- TUNABLE_INT ("hw.bus.devctl_disable",&devctl_disable)
- SYSCTL_PROC (_hw_bus, OID_AUTO, devctl_disable, CTLTYPE_INT|CTLFLAG_RW, 0, 0, sysctl_devctl_disable,"I","devctl disable")
- TAILQ_HEAD (devq, dev_event_info)
- static void devinit (void)
- static int devopen (struct cdev *dev, int oflags, int devtype, d_thread_t *td)
- static int devclose (struct cdev *dev, int fflag, int devtype, d_thread_t *td)
- static int devread (struct cdev *dev, struct uio *uio, int ioflag)
- static int devioctl (struct cdev *dev, u_long cmd, caddr_t data, int fflag, d_thread_t *td)
- static int devpoll (struct cdev *dev, int events, d_thread_t *td)
- void devctl_queue_data (char *data)

  Queue data to be read from the devctl device.

- void devctl_notify (const char *system, const char *subsystem, const char *type, const char *data)

  Send a 'notification' to userland, using standard ways.

- static void devaddq (const char *type, const char *what, device_t dev)
- static void devadded (device_t dev)
- static void devremoved (device_t dev)
- static void devnomatch (device_t dev)
- TAILQ_HEAD (device)
- DEFINE_CLASS (null, null_methods, 0)
- static devclass_t devclass_find_internal (const char *classname, const char *parentname, int create)

  Find or create a device class.

- devclass_t devclass_create (const char *classname)

  Create a device class.

- devclass_t devclass_find (const char *classname)

  Find a device class.

- int devclass_add_driver (devclass_t dc, driver_t *driver)

  Add a device driver to a device class.

- int devclass_delete_driver (devclass_t busclass, driver_t *driver)

  Delete a device driver from a device class.

- int devclass_quiesce_driver (devclass_t busclass, driver_t *driver)

  Quiesces a set of device drivers from a device class.

- static driverlink_t devclass_find_driver_internal (devclass_t dc, const char *classname)
- kobj_class_t devclass_find_driver (devclass_t dc, const char *classname)

  Search a devclass for a driver.

- const char * devclass_get_name (devclass_t dc)

  Return the name of the devclass.

- device_t devclass_get_device (devclass_t dc, int unit)

  Find a device given a unit number.

- void * devclass_get_softc (devclass_t dc, int unit)

  Find the softc field of a device given a unit number.

- int devclass_get_devices (devclass_t dc, device_t **devlistp, int *devcountp)

  Get a list of devices in the devclass.

- int devclass_get_drivers (devclass_t dc, driver_t ***listp, int *countp)

  Get a list of drivers in the devclass.

- int devclass_get_count (devclass_t dc)

  Get the number of devices in a devclass.

- int devclass_get_maxunit (devclass_t dc)

  Get the maximum unit number used in a devclass.

- int devclass_find_free_unit (devclass_t dc, int unit)

  Find a free unit number in a devclass.

- void devclass_set_parent (devclass_t dc, devclass_t pdc)

  Set the parent of a devclass.

- devclass_t devclass_get_parent (devclass_t dc)

  Get the parent of a devclass.

- sysctl_ctx_list * devclass_get_sysctl_ctx (devclass_t dc)
- sysctl_oid * devclass_get_sysctl_tree (devclass_t dc)
- static int devclass_alloc_unit (devclass_t dc, int *unitp)

  Allocate a unit number.

- static int devclass_add_device (devclass_t dc, device_t dev)

  Add a device to a devclass.

- static int devclass_delete_device (devclass_t dc, device_t dev)

  Delete a device from a devclass.

- static device_t make_device (device_t parent, const char *name, int unit)

Make a new device and add it as a child of `parent`.

- static int device_print_child (device_t dev, device_t child)

  Print a description of a device.

- device_t device_add_child (device_t dev, const char *name, int unit)

  Create a new device.

- device_t device_add_child_ordered (device_t dev, int order, const char *name, int unit)

  Create a new device.

- int device_delete_child (device_t dev, device_t child)

  Delete a device.

- device_t device_find_child (device_t dev, const char *classname, int unit)

  Find a device given a unit number.

- static driverlink_t first_matching_driver (devclass_t dc, device_t dev)
- static driverlink_t next_matching_driver (devclass_t dc, device_t dev, driverlink_t last)
- int device_probe_child (device_t dev, device_t child)
- device_t device_get_parent (device_t dev)

  Return the parent of a device.

- int device_get_children (device_t dev, device_t **devlistp, int *devcountp)

  Get a list of children of a device.

- driver_t * device_get_driver (device_t dev)

  Return the current driver for the device or `NULL` *if there is no driver currently attached.*

- devclass_t device_get_devclass (device_t dev)

  Return the current devclass for the device or `NULL` *if there is none.*

- const char * device_get_name (device_t dev)

  Return the name of the device's devclass or `NULL` *if there is none.*

- const char * device_get_nameunit (device_t dev)

  Return a string containing the device's devclass name followed by an ascii representation of the device's unit number (e.g. `"foo2"`*).*

- int device_get_unit (device_t dev)

  Return the device's unit number.

- const char * device_get_desc (device_t dev)

  Return the device's description string.

- u_int32_t device_get_flags (device_t dev)

  Return the device's flags.

- sysctl_ctx_list * device_get_sysctl_ctx (device_t dev)
- sysctl_oid * device_get_sysctl_tree (device_t dev)

- int device_print_prettyname (device_t dev)

  Print the name of the device followed by a colon and a space.

- int device_printf (device_t dev, const char *fmt,...)

  Print the name of the device followed by a colon, a space and the result of calling vprintf() with the value of `fmt` *and the following arguments.*

- static void device_set_desc_internal (device_t dev, const char *desc, int copy)
- void device_set_desc (device_t dev, const char *desc)

  Set the device's description.

- void device_set_desc_copy (device_t dev, const char *desc)

  Set the device's description.

- void device_set_flags (device_t dev, u_int32_t flags)

  Set the device's flags.

- void * device_get_softc (device_t dev)

  Return the device's softc field.

- void device_set_softc (device_t dev, void *softc)

  Set the device's softc field.

- void * device_get_ivars (device_t dev)

  Get the device's ivars field.

- void device_set_ivars (device_t dev, void *ivars)

  Set the device's ivars field.

- device_state_t device_get_state (device_t dev)

  Return the device's state.

- void device_enable (device_t dev)

  Set the DF_ENABLED flag for the device.

- void device_disable (device_t dev)

  Clear the DF_ENABLED flag for the device.

- void device_busy (device_t dev)

  Increment the busy counter for the device.

- void device_unbusy (device_t dev)

  Decrement the busy counter for the device.

- void device_quiet (device_t dev)

  Set the DF_QUIET flag for the device.

- void device_verbose (device_t dev)

  Clear the DF_QUIET flag for the device.

- int device_is_quiet (device_t dev)

  Return non-zero if the DF_QUIET flag is set on the device.

- int device_is_enabled (device_t dev)

  Return non-zero if the DF_ENABLED flag is set on the device.

- int device_is_alive (device_t dev)

  Return non-zero if the device was successfully probed.

- int device_is_attached (device_t dev)

  Return non-zero if the device currently has a driver attached to it.

- int device_set_devclass (device_t dev, const char *classname)

  Set the devclass of a device.

- int device_set_driver (device_t dev, driver_t *driver)

  Set the driver of a device.

- int device_probe_and_attach (device_t dev)

  Probe a device and attach a driver if possible.

- int device_attach (device_t dev)

  Attach a device driver to a device.

- int device_detach (device_t dev)

  Detach a driver from a device.

- int device_quiesce (device_t dev)

  Tells a driver to quiesce itself.

- int device_shutdown (device_t dev)

  Notify a device of system shutdown.

- int device_set_unit (device_t dev, int unit)

  Set the unit number of a device.

- void resource_list_init (struct resource_list *rl)

  Initialise a resource list.

- void resource_list_free (struct resource_list *rl)

  Reclaim memory used by a resource list.

- int resource_list_add_next (struct resource_list *rl, int type, u_long start, u_long end, u_long count)

  Add a resource entry.

- resource_list_entry * resource_list_add (struct resource_list *rl, int type, int rid, u_long start, u_long end, u_long count)

  Add or modify a resource entry.

- resource_list_entry * resource_list_find (struct resource_list *rl, int type, int rid)

  Find a resource entry by type and rid.

- void resource_list_delete (struct resource_list *rl, int type, int rid)

  Delete a resource entry.

- resource * resource_list_alloc (struct resource_list *rl, device_t bus, device_t child, int type, int *rid, u_long start, u_long end, u_long count, u_int flags)

  Helper function for implementing BUS_ALLOC_RESOURCE().

- int resource_list_release (struct resource_list *rl, device_t bus, device_t child, int type, int rid, struct resource *res)

  Helper function for implementing BUS_RELEASE_RESOURCE().

- int resource_list_print_type (struct resource_list *rl, const char *name, int type, const char *format)

  Print a description of resources in a resource list.

- void resource_list_purge (struct resource_list *rl)

  Releases all the resources in a list.

- device_t bus_generic_add_child (device_t dev, int order, const char *name, int unit)
- int bus_generic_probe (device_t dev)

  Helper function for implementing DEVICE_PROBE().

- int bus_generic_attach (device_t dev)

  Helper function for implementing DEVICE_ATTACH().

- int bus_generic_detach (device_t dev)

  Helper function for implementing DEVICE_DETACH().

- int bus_generic_shutdown (device_t dev)

  Helper function for implementing DEVICE_SHUTDOWN().

- int bus_generic_suspend (device_t dev)

  Helper function for implementing DEVICE_SUSPEND().

- int bus_generic_resume (device_t dev)

  Helper function for implementing DEVICE_RESUME().

- int bus_print_child_header (device_t dev, device_t child)

  Helper function for implementing BUS_PRINT_CHILD().

- int bus_print_child_footer (device_t dev, device_t child)

  Helper function for implementing BUS_PRINT_CHILD().

- int bus_generic_print_child (device_t dev, device_t child)

  Helper function for implementing BUS_PRINT_CHILD().

- int bus_generic_read_ivar (device_t dev, device_t child, int index, uintptr_t *result)

Stub function for implementing BUS_READ_IVAR().

- int bus_generic_write_ivar (device_t dev, device_t child, int index, uintptr_t value)
  Stub function for implementing BUS_WRITE_IVAR().
- resource_list * bus_generic_get_resource_list (device_t dev, device_t child)
  Stub function for implementing BUS_GET_RESOURCE_LIST().
- void bus_generic_driver_added (device_t dev, driver_t *driver)
  Helper function for implementing BUS_DRIVER_ADDED().
- int bus_generic_setup_intr (device_t dev, device_t child, struct resource *irq, int flags, driver_filter_t *filter, driver_intr_t *intr, void *arg, void **cookiep)
  Helper function for implementing BUS_SETUP_INTR().
- int bus_generic_teardown_intr (device_t dev, device_t child, struct resource *irq, void *cookie)
  Helper function for implementing BUS_TEARDOWN_INTR().
- resource * bus_generic_alloc_resource (device_t dev, device_t child, int type, int *rid, u_long start, u_long end, u_long count, u_int flags)
  Helper function for implementing BUS_ALLOC_RESOURCE().
- int bus_generic_release_resource (device_t dev, device_t child, int type, int rid, struct resource *r)
  Helper function for implementing BUS_RELEASE_RESOURCE().
- int bus_generic_activate_resource (device_t dev, device_t child, int type, int rid, struct resource *r)
  Helper function for implementing BUS_ACTIVATE_RESOURCE().
- int bus_generic_deactivate_resource (device_t dev, device_t child, int type, int rid, struct resource *r)
  Helper function for implementing BUS_DEACTIVATE_RESOURCE().
- int bus_generic_config_intr (device_t dev, int irq, enum intr_trigger trig, enum intr_polarity pol)
  Helper function for implementing BUS_CONFIG_INTR().
- bus_dma_tag_t bus_generic_get_dma_tag (device_t dev, device_t child)
  Helper function for implementing BUS_GET_DMA_TAG().
- int bus_generic_rl_get_resource (device_t dev, device_t child, int type, int rid, u_long *startp, u_long *countp)
  Helper function for implementing BUS_GET_RESOURCE().
- int bus_generic_rl_set_resource (device_t dev, device_t child, int type, int rid, u_long start, u_long count)
  Helper function for implementing BUS_SET_RESOURCE().
- void bus_generic_rl_delete_resource (device_t dev, device_t child, int type, int rid)
  Helper function for implementing BUS_DELETE_RESOURCE().
- int bus_generic_rl_release_resource (device_t dev, device_t child, int type, int rid, struct resource *r)

Helper function for implementing BUS_RELEASE_RESOURCE().

- resource * bus_generic_rl_alloc_resource (device_t dev, device_t child, int type, int *rid, u_long start, u_long end, u_long count, u_int flags)

  Helper function for implementing BUS_ALLOC_RESOURCE().

- int bus_generic_child_present (device_t dev, device_t child)

  Helper function for implementing BUS_CHILD_PRESENT().

- int bus_alloc_resources (device_t dev, struct resource_spec *rs, struct resource **res)
- void bus_release_resources (device_t dev, const struct resource_spec *rs, struct resource **res)
- resource * bus_alloc_resource (device_t dev, int type, int *rid, u_long start, u_long end, u_long count, u_int flags)

  Wrapper function for BUS_ALLOC_RESOURCE().

- int bus_activate_resource (device_t dev, int type, int rid, struct resource *r)

  Wrapper function for BUS_ACTIVATE_RESOURCE().

- int bus_deactivate_resource (device_t dev, int type, int rid, struct resource *r)

  Wrapper function for BUS_DEACTIVATE_RESOURCE().

- int bus_release_resource (device_t dev, int type, int rid, struct resource *r)

  Wrapper function for BUS_RELEASE_RESOURCE().

- int bus_setup_intr (device_t dev, struct resource *r, int flags, driver_filter_t filter, driver_intr_t handler, void *arg, void **cookiep)

  Wrapper function for BUS_SETUP_INTR().

- int bus_teardown_intr (device_t dev, struct resource *r, void *cookie)

  Wrapper function for BUS_TEARDOWN_INTR().

- int bus_set_resource (device_t dev, int type, int rid, u_long start, u_long count)

  Wrapper function for BUS_SET_RESOURCE().

- int bus_get_resource (device_t dev, int type, int rid, u_long *startp, u_long *countp)

  Wrapper function for BUS_GET_RESOURCE().

- u_long bus_get_resource_start (device_t dev, int type, int rid)

  Wrapper function for BUS_GET_RESOURCE().

- u_long bus_get_resource_count (device_t dev, int type, int rid)

  Wrapper function for BUS_GET_RESOURCE().

- void bus_delete_resource (device_t dev, int type, int rid)

  Wrapper function for BUS_DELETE_RESOURCE().

- int bus_child_present (device_t child)

  Wrapper function for BUS_CHILD_PRESENT().

- int bus_child_pnpinfo_str (device_t child, char *buf, size_t buflen)

Wrapper function for BUS_CHILD_PNPINFO_STR().

- int bus_child_location_str (device_t child, char *buf, size_t buflen)

  Wrapper function for BUS_CHILD_LOCATION_STR().

- bus_dma_tag_t bus_get_dma_tag (device_t dev)

  Wrapper function for BUS_GET_DMA_TAG().

- static int root_resume (device_t dev)
- static int root_print_child (device_t dev, device_t child)
- static int root_setup_intr (device_t dev, device_t child, driver_intr_t *intr, void *arg, void **cookiep)
- static int root_child_present (device_t dev, device_t child)
- static int root_bus_module_handler (module_t mod, int what, void *arg)
- DECLARE_MODULE (rootbus, root_bus_mod, SI_SUB_DRIVERS, SI_ORDER_FIRST)
- void root_bus_configure (void)

  Automatically configure devices.

- int driver_module_handler (module_t mod, int what, void *arg)

  Module handler for registering device drivers.

- void bus_enumerate_hinted_children (device_t bus)

  Enumerate all hinted devices for this bus.

- static int sysctl_bus (SYSCTL_HANDLER_ARGS)
- SYSCTL_NODE (_hw_bus, OID_AUTO, info, CTLFLAG_RW, sysctl_bus,"bus-related data")
- static int sysctl_devices (SYSCTL_HANDLER_ARGS)
- SYSCTL_NODE (_hw_bus, OID_AUTO, devices, CTLFLAG_RD, sysctl_devices,"system device tree")
- int bus_data_generation_check (int generation)
- void bus_data_generation_update (void)
- int bus_free_resource (device_t dev, int type, struct resource *r)

Variables

- static int devctl_disable = 0
- static d_open_t devopen
- static d_close_t devclose
- static d_read_t devread
- static d_ioctl_t devioctl
- static d_poll_t devpoll
- static struct cdevsw dev_cdevsw
- static struct dev_softc devsoftc
- static struct cdev * devctl_dev
- static devclass_list_t devclasses = TAILQ_HEAD_INITIALIZER(devclasses)
- static kobj_method_t root_methods [ ]
- static driver_t root_driver
- device_t root_bus
- devclass_t root_devclass
- static moduledata_t root_bus_mod

9.87.1 Define Documentation

9.87.1.1 #define DEVCLANAME(d)

Definition at line 175 of file subr_bus.c.

Referenced by devclass_add_device(), devclass_alloc_unit(), devclass_delete_device(), devclass_delete_driver(), devclass_find_driver_internal(), and devclass_quiesce_driver().

9.87.1.2 #define DEVICENAME(d)

Definition at line 173 of file subr_bus.c.

Referenced by devclass_add_device(), devclass_delete_device(), device_add_child_ordered(), device_delete_child(), device_detach(), device_quiesce(), and make_device().

9.87.1.3 #define DF_DESCMALLOCED 8

Referenced by device_set_desc_internal().

9.87.1.4 #define DF_DONENOMATCH 32

Referenced by device_probe_and_attach().

9.87.1.5 #define DF_ENABLED 1

Referenced by device_disable(), device_enable(), device_is_enabled(), device_probe_and_attach(), and make_device().

9.87.1.6 #define DF_EXTERNALSOFTC 64

Referenced by device_set_driver(), and device_set_softc().

9.87.1.7 #define DF_FIXEDCLASS 2

Referenced by device_detach(), and make_device().

9.87.1.8 #define DF_QUIET 16

Referenced by device_is_quiet(), device_quiet(), and device_verbose().

9.87.1.9 #define DF_REBID 128

Referenced by bus_generic_driver_added(), device_probe_and_attach(), and device_probe_child().

9.87.1.10 #define DF_WILDCARD 4

Referenced by devclass_delete_device(), and make_device().

9.87.1.11 #define DRIVERNAME(d)

Definition at line 174 of file subr_bus.c.

Referenced by devclass_add_driver(), device_probe_child(), and driver_module_handler().

9.87.1.12 #define PDEBUG(a)

Definition at line 172 of file subr_bus.c.

Referenced by devclass_add_device(), devclass_add_driver(), devclass_alloc_unit(), devclass_delete_device(), devclass_delete_driver(), devclass_find_driver_internal(), devclass_find_internal(), devclass_quiesce_driver(), device_add_child_ordered(), device_delete_child(), device_detach(), device_probe_child(), device_quiesce(), driver_module_handler(), make_device(), and root_bus_configure().

9.87.1.13 #define print_devclass(d, i)

Definition at line 185 of file subr_bus.c.

9.87.1.14 #define print_devclass_list()

Definition at line 187 of file subr_bus.c.

9.87.1.15 #define print_devclass_list_short()

Definition at line 186 of file subr_bus.c.

9.87.1.16 #define print_devclass_short(d, i)

Definition at line 184 of file subr_bus.c.

9.87.1.17 #define print_device(d, i)

Definition at line 178 of file subr_bus.c.

9.87.1.18 #define print_device_short(d, i)

Definition at line 177 of file subr_bus.c.

9.87.1.19 #define print_device_tree(d, i)

Definition at line 180 of file subr_bus.c.

9.87.1.20 #define print_device_tree_short(d, i)

Definition at line 179 of file subr_bus.c.

9.87.1.21 #define print_driver(d, i)

Definition at line 182 of file subr_bus.c.

9.87.1.22 #define print_driver_list(d, i)

Definition at line 183 of file subr_bus.c.

9.87.1.23 #define print_driver_short(d, i)

Definition at line 181 of file subr_bus.c.

9.87.2 Typedef Documentation

9.87.2.1 typedef struct driverlink* driverlink_t

Definition at line 65 of file subr_bus.c.

9.87.3 Enumeration Type Documentation

9.87.3.1 anonymous enum

Enumerator:

DEVICE_SYSCTL_DESC
DEVICE_SYSCTL_DRIVER
DEVICE_SYSCTL_LOCATION
DEVICE_SYSCTL_PNPINFO
DEVICE_SYSCTL_PARENT

Definition at line 230 of file subr_bus.c.

9.87.3.2 anonymous enum

Enumerator:

DEVCLASS_SYSCTL_PARENT

Definition at line 194 of file subr_bus.c.

9.87.4 Function Documentation

9.87.4.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_bus. *c*, v 1.198 2007/02/23 12:19:01 piso Exp $")

9.87.4.2 int bus_activate_resource (device_t *dev*, int *type*, int *rid*, struct resource * *r*)

Wrapper function for BUS_ACTIVATE_RESOURCE().

This function simply calls the BUS_ACTIVATE_RESOURCE() method of the parent of dev.

Definition at line 3419 of file subr_bus.c.

9.87.4.3 struct resource* bus_alloc_resource (device_t *dev*, int *type*, int * *rid*, u_long *start*, u_long *end*, u_long *count*, u_int *flags*)

Wrapper function for BUS_ALLOC_RESOURCE().

This function simply calls the BUS_ALLOC_RESOURCE() method of the parent of dev.

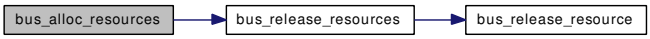
Definition at line 3403 of file subr_bus.c.

9.87.4.4 int bus_alloc_resources (device_t *dev*, struct resource_spec * *rs*, struct resource ** *res*)

Definition at line 3364 of file subr_bus.c.

References bus_release_resources().

Here is the call graph for this function:



9.87.4.5 int bus_child_location_str (device_t *child*, char * *buf*, size_t *buflen*)

Wrapper function for BUS_CHILD_LOCATION_STR().

This function simply calls the BUS_CHILD_LOCATION_STR() method of the parent of dev.

Definition at line 3620 of file subr_bus.c.

References device_get_parent().

Referenced by devaddq(), device_sysctl_handler(), and sysctl_devices().

Here is the call graph for this function:

9.87.4.6 int bus_child_pnpinfo_str (device_t *child*, char * *buf*, size_t *buflen*)

Wrapper function for BUS_CHILD_PNPINFO_STR().

This function simply calls the BUS_CHILD_PNPINFO_STR() method of the parent of dev.

Definition at line 3601 of file subr_bus.c.

References device_get_parent().

Referenced by devadded(), devaddq(), device_sysctl_handler(), devremoved(), and sysctl_devices().

Here is the call graph for this function:

9.87.4.7 int bus_child_present (device_t *child*)

Wrapper function for BUS_CHILD_PRESENT().

This function simply calls the BUS_CHILD_PRESENT() method of the parent of `dev`.

Definition at line 3589 of file subr_bus.c.

References device_get_parent().

Here is the call graph for this function:

9.87.4.8 int bus_data_generation_check (int *generation*)

Definition at line 4120 of file subr_bus.c.

Referenced by sysctl_devices(), and sysctl_rman().

9.87.4.9 void bus_data_generation_update (void)

Definition at line 4130 of file subr_bus.c.

Referenced by devclass_add_driver(), devclass_find_internal(), device_add_child_ordered(), device_delete_child(), device_probe_child(), device_set_desc_internal(), device_set_devclass(), device_set_driver(), device_set_unit(), and make_device().

9.87.4.10 int bus_deactivate_resource (device_t *dev*, int *type*, int *rid*, struct resource * *r*)

Wrapper function for BUS_DEACTIVATE_RESOURCE().

This function simply calls the BUS_DEACTIVATE_RESOURCE() method of the parent of `dev`.

Definition at line 3433 of file subr_bus.c.

9.87.4.11 void bus_delete_resource (device_t *dev*, int *type*, int *rid*)

Wrapper function for BUS_DELETE_RESOURCE().

This function simply calls the BUS_DELETE_RESOURCE() method of the parent of `dev`.

Definition at line 3577 of file subr_bus.c.

References device_get_parent().

Here is the call graph for this function:

9.87.4.12 void bus_enumerate_hinted_children (device_t *bus*)

Enumerate all hinted devices for this bus.

Walks throught he hints for this bus and calls the bus_hinted_child routine for each one it fines. It searches first for the specific bus that's being probed for hinted children (eg isa0), and then for generic children (eg isa).

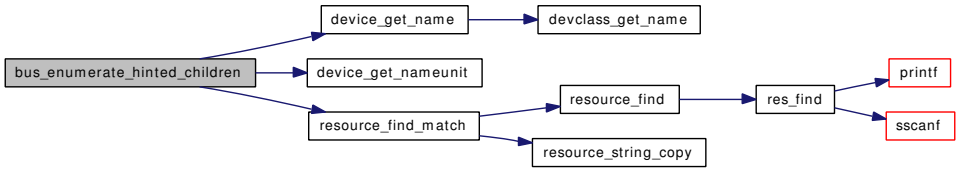
Parameters:

dev bus device to enumerate

Definition at line 3861 of file subr_bus.c.

References device_get_name(), device_get_nameunit(), and resource_find_match().

Here is the call graph for this function:

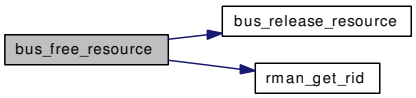


9.87.4.13 int bus_free_resource (device_t *dev*, int *type*, struct resource * *r*)

Definition at line 4136 of file subr_bus.c.

References bus_release_resource(), and rman_get_rid().

Here is the call graph for this function:



9.87.4.14 int bus_generic_activate_resource (device_t *dev*, device_t *child*, int *type*, int *rid*, struct resource * *r*)

Helper function for implementing BUS_ACTIVATE_RESOURCE().

This simple implementation of BUS_ACTIVATE_RESOURCE() simply calls the BUS_ACTIVATE_-RESOURCE() method of the parent of `dev`.

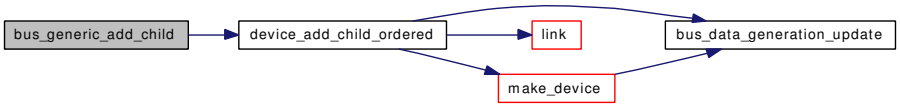
Definition at line 3166 of file subr_bus.c.

9.87.4.15 device_t bus_generic_add_child (device_t *dev*, int *order*, const char * *name*, int *unit*)

Definition at line 2844 of file subr_bus.c.

References device_add_child_ordered().

Here is the call graph for this function:



9.87.4.16 struct resource* bus_generic_alloc_resource (device_t *dev*, device_t *child*, int *type*, int * *rid*, u_long *start*, u_long *end*, u_long *count*, u_int *flags*)

Helper function for implementing BUS_ALLOC_RESOURCE().

This simple implementation of BUS_ALLOC_RESOURCE() simply calls the BUS_ALLOC_-RESOURCE() method of the parent of `dev`.

Definition at line 3132 of file subr_bus.c.

9.87.4.17 int bus_generic_attach (device_t *dev*)

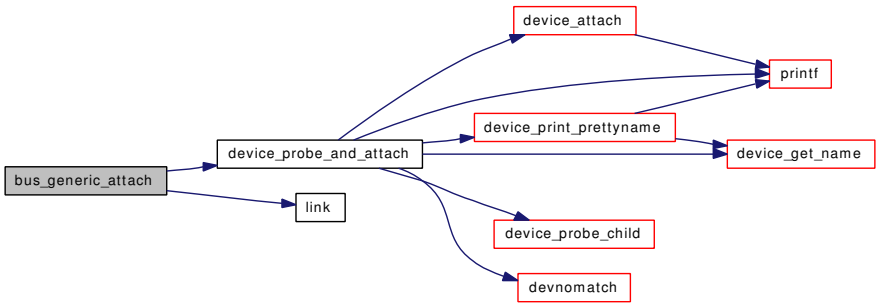
Helper function for implementing DEVICE_ATTACH().

This function can be used to help implement the DEVICE_ATTACH() for a bus. It calls device_probe_-and_attach() for each of the device's children.

Definition at line 2879 of file subr_bus.c.

References device_probe_and_attach(), and link().

Here is the call graph for this function:



9.87.4.18 int bus_generic_child_present (device_t *dev*, device_t *child*)

Helper function for implementing BUS_CHILD_PRESENT().

This simple implementation of BUS_CHILD_PRESENT() simply calls the BUS_CHILD_PRESENT() method of the parent of `dev`.

Definition at line 3350 of file subr_bus.c.

References device_get_parent().

Here is the call graph for this function:

9.87.4.19 int bus_generic_config_intr (device_t *dev*, int *irq*, enum intr_trigger *trig*, enum intr_polarity *pol*)

Helper function for implementing BUS_CONFIG_INTR().

This simple implementation of BUS_CONFIG_INTR() simply calls the BUS_CONFIG_INTR() method of the parent of `dev`.

Definition at line 3200 of file subr_bus.c.

9.87.4.20 int bus_generic_deactivate_resource (device_t *dev*, device_t *child*, int *type*, int *rid*, struct resource * *r*)

Helper function for implementing BUS_DEACTIVATE_RESOURCE().

This simple implementation of BUS_DEACTIVATE_RESOURCE() simply calls the BUS_-DEACTIVATE_RESOURCE() method of the parent of `dev`.

Definition at line 3183 of file subr_bus.c.

9.87.4.21 int bus_generic_detach (device_t *dev*)

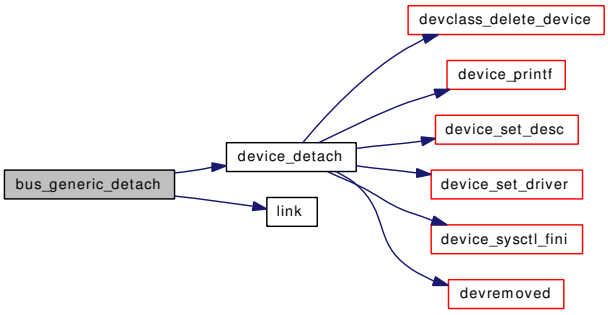
Helper function for implementing DEVICE_DETACH().

This function can be used to help implement the DEVICE_DETACH() for a bus. It calls device_detach() for each of the device's children.

Definition at line 2898 of file subr_bus.c.

References device_detach(), and link().

Here is the call graph for this function:



9.87.4.22 void bus_generic_driver_added (device_t *dev*, driver_t * *driver*)

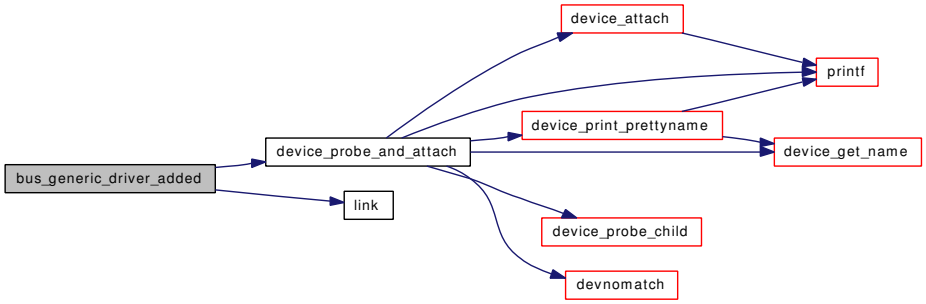
Helper function for implementing BUS_DRIVER_ADDED().

This implementation of BUS_DRIVER_ADDED() simply calls the driver's DEVICE_IDENTIFY() method to allow it to add new children to the bus and then calls device_probe_and_attach() for each unattached child.

Definition at line 3079 of file subr_bus.c.

References device_probe_and_attach(), DF_REBID, and link().

Here is the call graph for this function:



9.87.4.23 bus_dma_tag_t bus_generic_get_dma_tag (device_t *dev*, device_t *child*)

Helper function for implementing BUS_GET_DMA_TAG().

This simple implementation of BUS_GET_DMA_TAG() simply calls the BUS_GET_DMA_TAG() method of the parent of `dev`.

Definition at line 3217 of file subr_bus.c.

9.87.4.24 struct resource_list* bus_generic_get_resource_list (device_t *dev*, device_t *child*)

Stub function for implementing BUS_GET_RESOURCE_LIST().

Returns:

NULL

Definition at line 3066 of file subr_bus.c.

9.87.4.25 int bus_generic_print_child (device_t *dev*, device_t *child*)

Helper function for implementing BUS_PRINT_CHILD().

This function simply calls bus_print_child_header() followed by bus_print_child_footer().

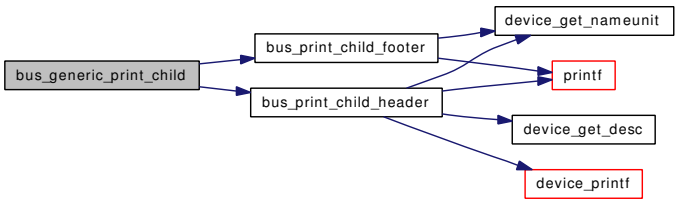
Returns:

the number of characters printed

Definition at line 3026 of file subr_bus.c.

References bus_print_child_footer(), and bus_print_child_header().

Here is the call graph for this function:



9.87.4.26 int bus_generic_probe (device_t *dev*)

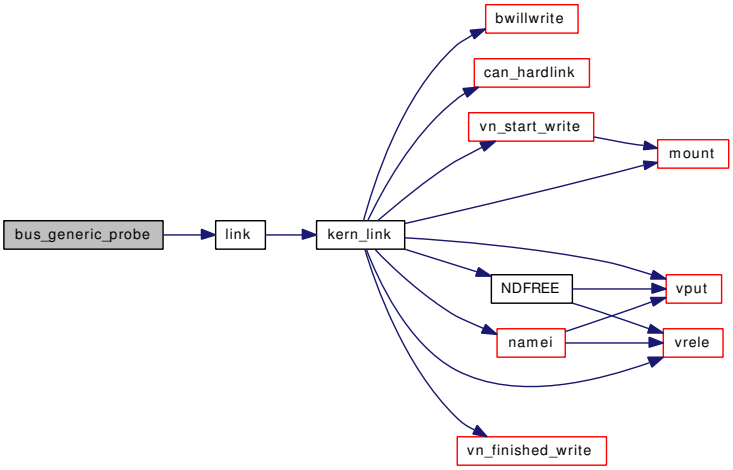
Helper function for implementing DEVICE_PROBE().

This function can be used to help implement the DEVICE_PROBE() for a bus (i.e. a device which has other devices attached to it). It calls the DEVICE_IDENTIFY() method of each driver in the device's devclass.

Definition at line 2859 of file subr_bus.c.

References driverlink::driver, and link().

Here is the call graph for this function:



9.87.4.27 int bus_generic_read_ivar (device_t *dev*, device_t *child*, int *index*, uintptr_t * *result*)

Stub function for implementing BUS_READ_IVAR().

Returns:

ENOENT

Definition at line 3042 of file subr_bus.c.

9.87.4.28 int bus_generic_release_resource (device_t *dev*, device_t *child*, int *type*, int *rid*, struct resource * *r*)

Helper function for implementing BUS_RELEASE_RESOURCE().

This simple implementation of BUS_RELEASE_RESOURCE() simply calls the BUS_RELEASE_-RESOURCE() method of the parent of `dev`.

Definition at line 3149 of file subr_bus.c.

9.87.4.29 int bus_generic_resume (device_t *dev*)

Helper function for implementing DEVICE_RESUME().

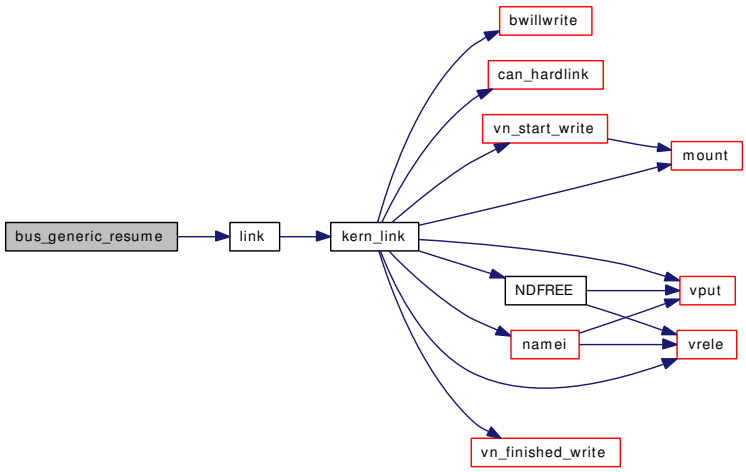
This function can be used to help implement the DEVICE_RESUME() for a bus. It calls DEVICE_-RESUME() on each of the device's children.

Definition at line 2968 of file subr_bus.c.

References link().

Referenced by root_resume().

Here is the call graph for this function:



9.87.4.30 struct resource* bus_generic_rl_alloc_resource (device_t *dev*, device_t *child*, int *type*, int * *rid*, u_long *start*, u_long *end*, u_long *count*, u_int *flags*)

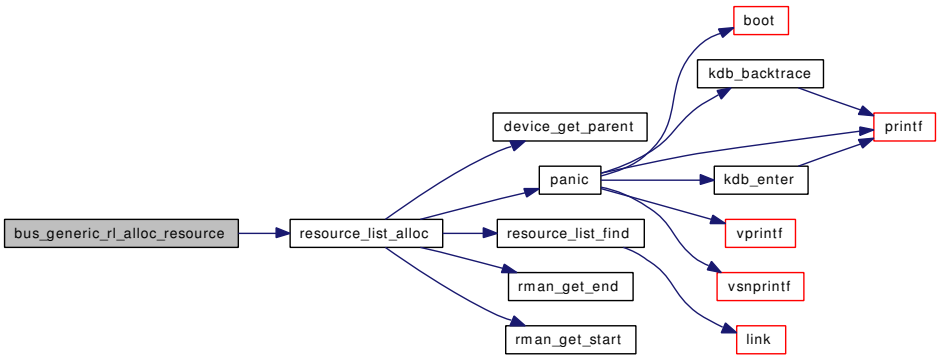
Helper function for implementing BUS_ALLOC_RESOURCE().

This implementation of BUS_ALLOC_RESOURCE() uses the resource_list_alloc() function to do most of the work. It calls BUS_GET_RESOURCE_LIST() to find a suitable resource list.

Definition at line 3330 of file subr_bus.c.

References resource_list_alloc().

Here is the call graph for this function:



9.87.4.31 void bus_generic_rl_delete_resource (device_t *dev*, device_t *child*, int *type*, int *rid*)

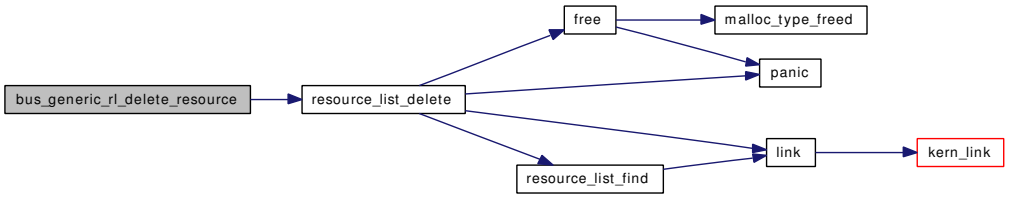
Helper function for implementing BUS_DELETE_RESOURCE().

This implementation of BUS_DELETE_RESOURCE() uses the resource_list_delete() function to do most of the work. It calls BUS_GET_RESOURCE_LIST() to find a suitable resource list to edit.

Definition at line 3289 of file subr_bus.c.

References resource_list_delete().

Here is the call graph for this function:



9.87.4.32 int bus_generic_rl_get_resource (device_t *dev*, device_t *child*, int *type*, int *rid*, u_long * *startp*, u_long * *countp*)

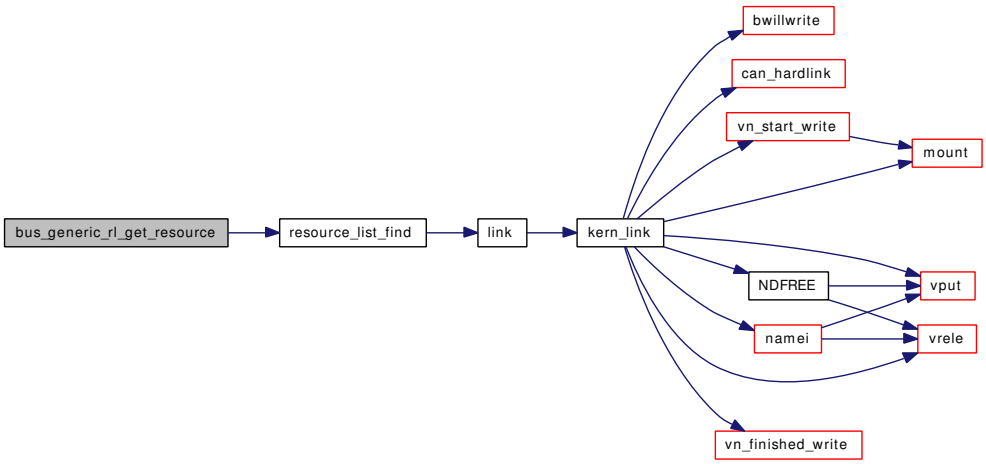
Helper function for implementing BUS_GET_RESOURCE().

This implementation of BUS_GET_RESOURCE() uses the resource_list_find() function to do most of the work. It calls BUS_GET_RESOURCE_LIST() to find a suitable resource list to search.

Definition at line 3235 of file subr_bus.c.

References resource_list_find().

Here is the call graph for this function:



9.87.4.33 int bus_generic_rl_release_resource (device_t *dev*, device_t *child*, int *type*, int *rid*, struct resource * *r*)

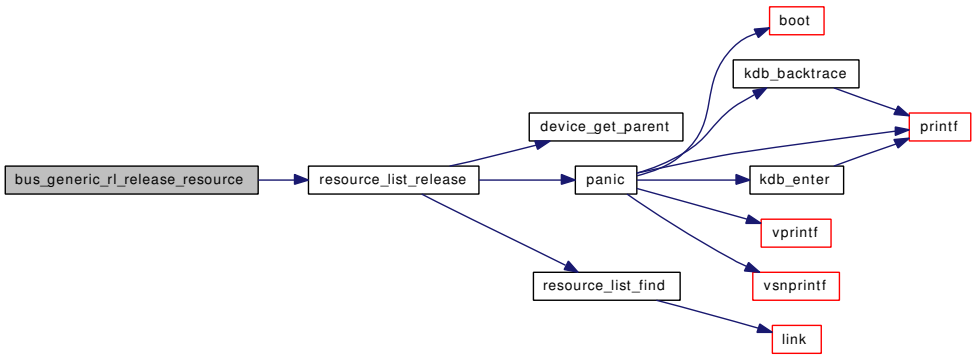
Helper function for implementing BUS_RELEASE_RESOURCE().

This implementation of BUS_RELEASE_RESOURCE() uses the resource_list_release() function to do most of the work. It calls BUS_GET_RESOURCE_LIST() to find a suitable resource list.

Definition at line 3310 of file subr_bus.c.

References resource_list_release().

Here is the call graph for this function:



9.87.4.34 int bus_generic_rl_set_resource (device_t *dev*, device_t *child*, int *type*, int *rid*, u_long *start*, u_long *count*)

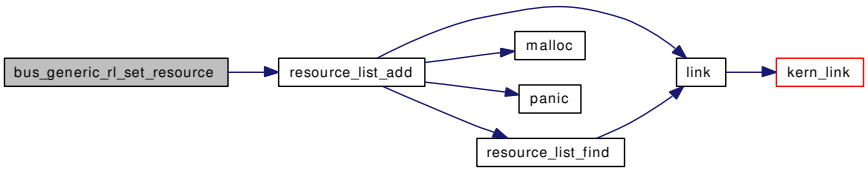
Helper function for implementing BUS_SET_RESOURCE().

This implementation of BUS_SET_RESOURCE() uses the resource_list_add() function to do most of the work. It calls BUS_GET_RESOURCE_LIST() to find a suitable resource list to edit.

Definition at line 3266 of file subr_bus.c.

References resource_list_add().

Here is the call graph for this function:



9.87.4.35 int bus_generic_setup_intr (device_t *dev*, device_t *child*, struct resource * *irq*, int *flags*, driver_filter_t * *filter*, driver_intr_t * *intr*, void * *arg*, void ** *cookiep*)

Helper function for implementing BUS_SETUP_INTR().

This simple implementation of BUS_SETUP_INTR() simply calls the BUS_SETUP_INTR() method of the parent of `dev`.

Definition at line 3098 of file subr_bus.c.

9.87.4.36 int bus_generic_shutdown (device_t *dev*)

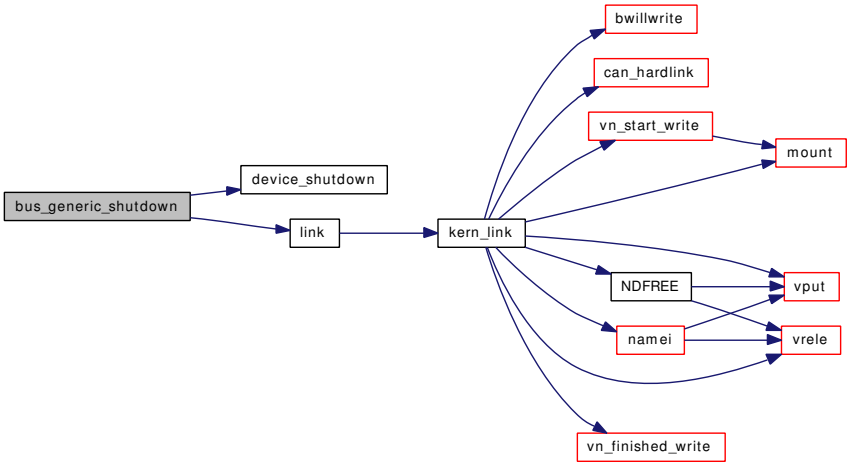
Helper function for implementing DEVICE_SHUTDOWN().

This function can be used to help implement the DEVICE_SHUTDOWN() for a bus. It calls device_shutdown() for each of the device's children.

Definition at line 2922 of file subr_bus.c.

References device_shutdown(), and link().

Here is the call graph for this function:



9.87.4.37 int bus_generic_suspend (device_t *dev*)

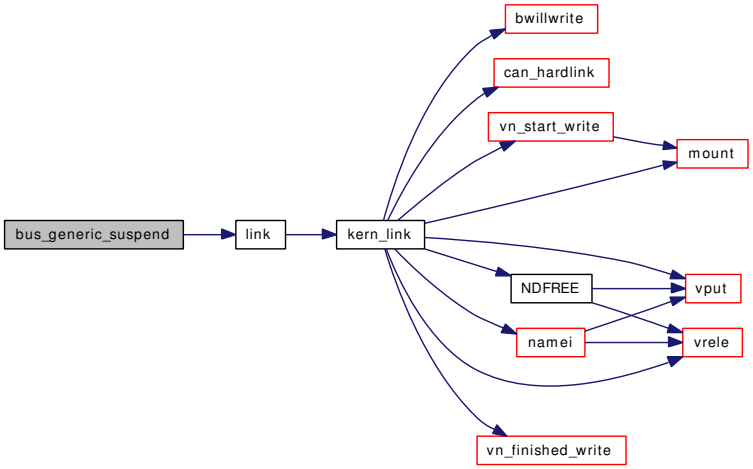
Helper function for implementing DEVICE_SUSPEND().

This function can be used to help implement the DEVICE_SUSPEND() for a bus. It calls DEVICE_-SUSPEND() for each of the device's children. If any call to DEVICE_SUSPEND() fails, the suspend operation is aborted and any devices which were suspended are resumed immediately by calling their DEVICE_RESUME() methods.

Definition at line 2943 of file subr_bus.c.

References link().

Here is the call graph for this function:



9.87.4.38 int bus_generic_teardown_intr (device_t *dev*, device_t *child*, struct resource * *irq*, void * *cookie*)

Helper function for implementing BUS_TEARDOWN_INTR().

This simple implementation of BUS_TEARDOWN_INTR() simply calls the BUS_TEARDOWN_INTR() method of the parent of `dev`.

Definition at line 3116 of file subr_bus.c.

9.87.4.39 int bus_generic_write_ivar (device_t *dev*, device_t *child*, int *index*, uintptr_t *value*)

Stub function for implementing BUS_WRITE_IVAR().

Returns:

ENOENT

Definition at line 3054 of file subr_bus.c.

9.87.4.40 bus_dma_tag_t bus_get_dma_tag (device_t *dev*)

Wrapper function for BUS_GET_DMA_TAG().

This function simply calls the BUS_GET_DMA_TAG() method of the parent of dev.

Definition at line 3639 of file subr_bus.c.

References device_get_parent().

Here is the call graph for this function:

9.87.4.41 int bus_get_resource (device_t *dev*, int *type*, int *rid*, u_long * *startp*, u_long * *countp*)

Wrapper function for BUS_GET_RESOURCE().

This function simply calls the BUS_GET_RESOURCE() method of the parent of dev.

Definition at line 3525 of file subr_bus.c.

References device_get_parent().

Here is the call graph for this function:

9.87.4.42 u_long bus_get_resource_count (device_t *dev*, int *type*, int *rid*)

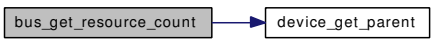
Wrapper function for BUS_GET_RESOURCE().

This function simply calls the BUS_GET_RESOURCE() method of the parent of dev and returns the count value.

Definition at line 3558 of file subr_bus.c.

References device_get_parent().

Here is the call graph for this function:



9.87.4.43 u_long bus_get_resource_start (device_t *dev*, int *type*, int *rid*)

Wrapper function for BUS_GET_RESOURCE().

This function simply calls the BUS_GET_RESOURCE() method of the parent of dev and returns the start value.

Definition at line 3539 of file subr_bus.c.

References device_get_parent().

Here is the call graph for this function:

9.87.4.44 int bus_print_child_footer (device_t *dev*, device_t *child*)

Helper function for implementing BUS_PRINT_CHILD().

This function prints the last part of the ascii representation of `child`, which consists of the string `" on "` followed by the name and unit of the `dev`.

Returns:

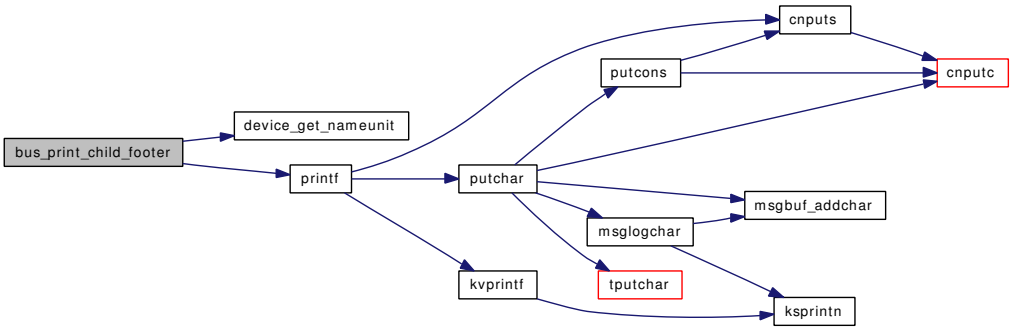
the number of characters printed

Definition at line 3012 of file subr_bus.c.

References device_get_nameunit(), and printf().

Referenced by bus_generic_print_child().

Here is the call graph for this function:



9.87.4.45 int bus_print_child_header (device_t *dev*, device_t *child*)

Helper function for implementing BUS_PRINT_CHILD().

This function prints the first part of the ascii representation of `child`, including its name, unit and description (if any - see device_set_desc()).

Returns:

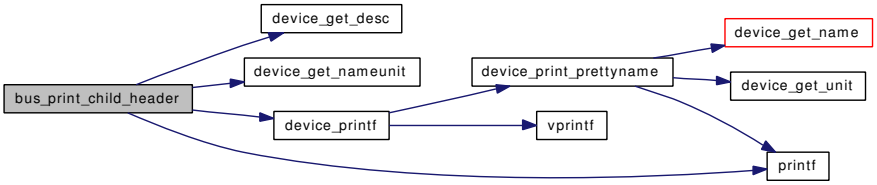
the number of characters printed

Definition at line 2989 of file subr_bus.c.

References device_get_desc(), device_get_nameunit(), device_printf(), and printf().

Referenced by bus_generic_print_child(), and root_print_child().

Here is the call graph for this function:



9.87.4.46 int bus_release_resource (device_t *dev*, int *type*, int *rid*, struct resource * *r*)

Wrapper function for BUS_RELEASE_RESOURCE().

This function simply calls the BUS_RELEASE_RESOURCE() method of the parent of `dev`.

Definition at line 3447 of file subr_bus.c.

Referenced by bus_free_resource(), bus_release_resources(), and resource_list_purge().

9.87.4.47 void bus_release_resources (device_t *dev*, const struct resource_spec * *rs*, struct resource ** *res*)

Definition at line 3383 of file subr_bus.c.

References bus_release_resource().

Referenced by bus_alloc_resources().

Here is the call graph for this function:

9.87.4.48 int bus_set_resource (device_t *dev*, int *type*, int *rid*, u_long *start*, u_long *count*)

Wrapper function for BUS_SET_RESOURCE().

This function simply calls the BUS_SET_RESOURCE() method of the parent of `dev`.

Definition at line 3511 of file subr_bus.c.

References device_get_parent().

Here is the call graph for this function:

9.87.4.49 int bus_setup_intr (device_t *dev*, struct resource * *r*, int *flags*, driver_filter_t *filter*, driver_intr_t *handler*, void * *arg*, void ** *cookiep*)

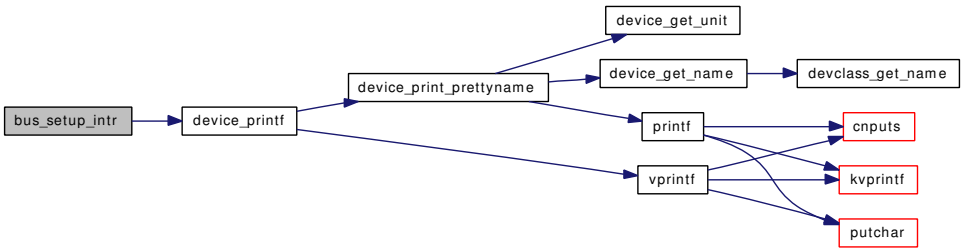
Wrapper function for BUS_SETUP_INTR().

This function simply calls the BUS_SETUP_INTR() method of the parent of `dev`.

Definition at line 3461 of file subr_bus.c.

References bootverbose, and device_printf().

Here is the call graph for this function:



9.87.4.50 int bus_teardown_intr (device_t *dev*, struct resource * *r*, void * *cookie*)

Wrapper function for BUS_TEARDOWN_INTR().

This function simply calls the BUS_TEARDOWN_INTR() method of the parent of `dev`.

Definition at line 3497 of file subr_bus.c.

9.87.4.51 DECLARE_MODULE (rootbus, root_bus_mod, SI_SUB_DRIVERS, SI_ORDER_FIRST)

9.87.4.52 DEFINE_CLASS (null, null_methods, 0)

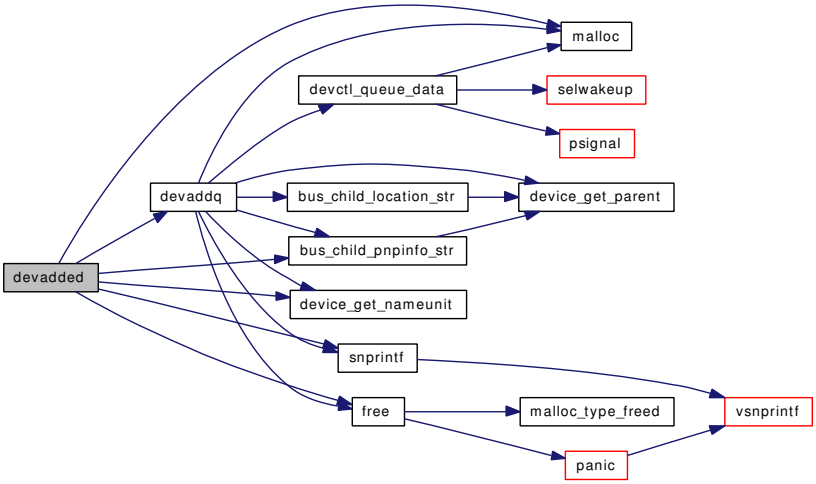
9.87.4.53 static void devadded (device_t *dev*) `[static]`

Definition at line 635 of file subr_bus.c.

References bus_child_pnpinfo_str(), devaddq(), device_get_nameunit(), free(), malloc(), and snprintf().

Referenced by device_attach().

Here is the call graph for this function:



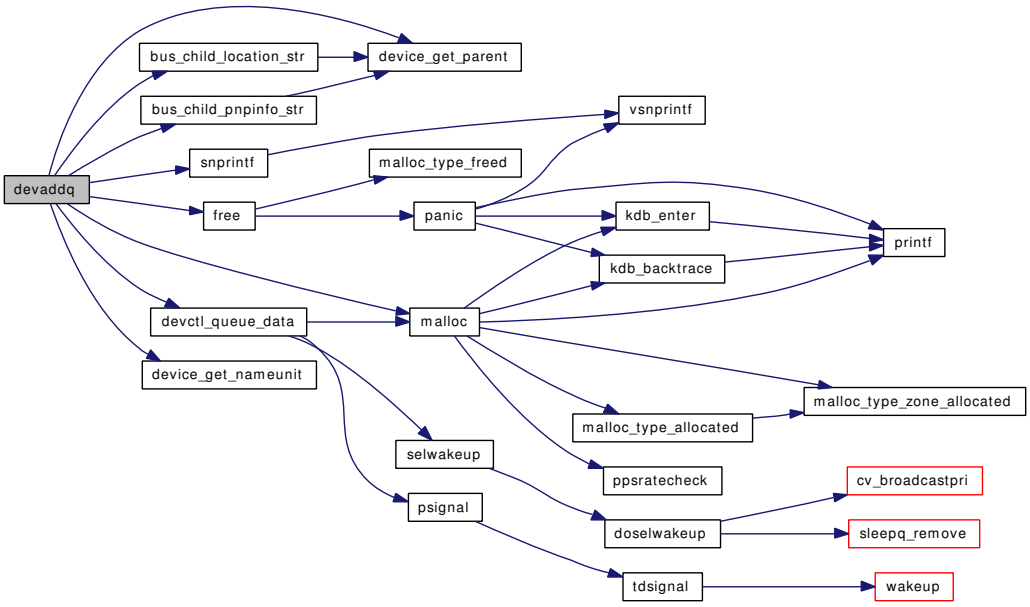
9.87.4.54 static void devaddq (const char * *type*, const char * *what*, device_t *dev*) [static]

Definition at line 581 of file subr_bus.c.

References bus_child_location_str(), bus_child_pnpinfo_str(), devctl_disable, devctl_queue_data(), device_get_nameunit(), device_get_parent(), free(), malloc(), and snprintf().

Referenced by devadded(), devnomatch(), and devremoved().

Here is the call graph for this function:



9.87.4.55 static int devclass_add_device (devclass_t *dc*, device_t *dev*) [static]

Add a device to a devclass.

For internal use only.

A unit number is allocated for the device (using the device's preferred unit number if any) and the device is registered in the devclass. This allows the device to be looked up by its unit number, e.g. by decoding a dev_t minor number.

Parameters:

dc the devclass to add to

dev the device to add

Return values:

0 success

EEXIST the requested unit number is already allocated

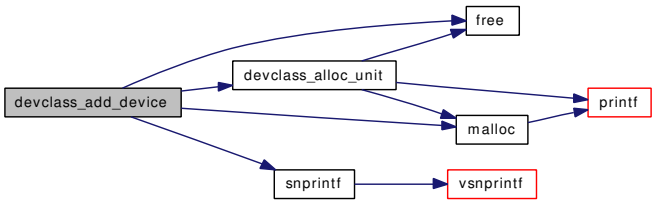
ENOMEM memory allocation failure

Definition at line 1363 of file subr_bus.c.

References DEVCLANAME, devclass_alloc_unit(), DEVICENAME, free(), malloc(), PDEBUG, and snprintf().

Referenced by device_set_devclass(), device_set_unit(), and make_device().

Here is the call graph for this function:



9.87.4.56 int devclass_add_driver (devclass_t *dc*, driver_t * *driver*)

Add a device driver to a device class.

Add a device driver to a devclass. This is normally called automatically by DRIVER_MODULE(). The BUS_DRIVER_ADDED() method of all devices in the devclass will be called to allow them to attempt to re-probe any unmatched children.

Parameters:

dc the devclass to edit

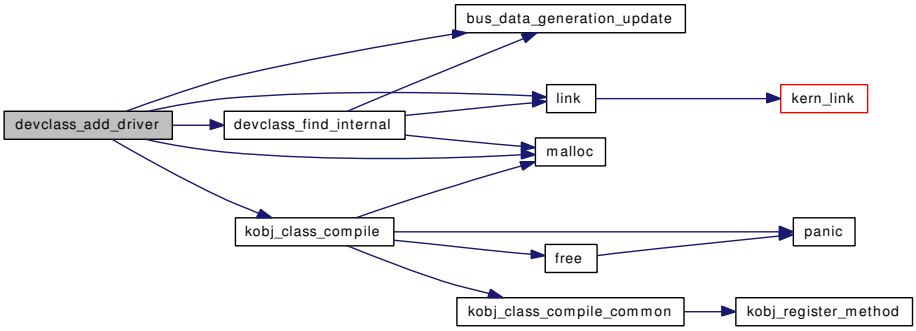
driver the driver to register

Definition at line 842 of file subr_bus.c.

References bus_data_generation_update(), devclass_find_internal(), DRIVERNAME, kobj_class_compile(), link(), malloc(), and PDEBUG.

Referenced by driver_module_handler().

Here is the call graph for this function:



9.87.4.57 static int devclass_alloc_unit (devclass_t *dc*, int * *unitp*) [static]

Allocate a unit number.

For internal use only.

On entry, `*unitp` is the desired unit number (or `-1` if any will do). The allocated unit number is returned in `*unitp`.

Parameters:

dc the devclass to allocate from

unitp points at the location for the allocated unit number

Return values:

0 success

EEXIST the requested unit number is already allocated

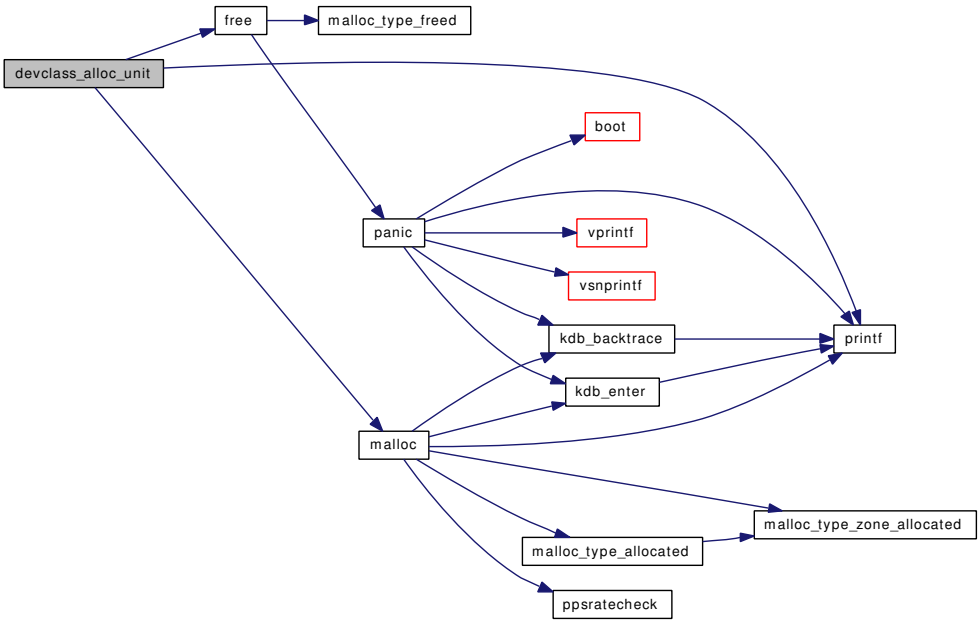
ENOMEM memory allocation failure

Definition at line 1296 of file subr_bus.c.

References bootverbose, DEVCLANAME, free(), malloc(), PDEBUG, and printf().

Referenced by devclass_add_device().

Here is the call graph for this function:



9.87.4.58 devclass_t devclass_create (const char * *classname*)

Create a device class.

If a device class with the name `classname` exists, return it, otherwise create and return a new device class.

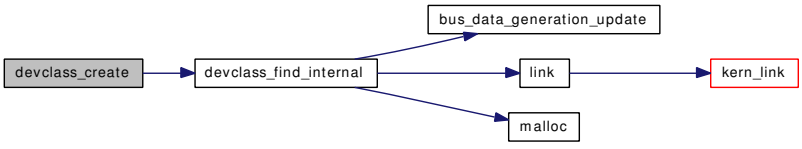
Parameters:

classname the devclass name to find or create

Definition at line 811 of file subr_bus.c.

References devclass_find_internal().

Here is the call graph for this function:



9.87.4.59 static int devclass_delete_device (devclass_t *dc*, device_t *dev*) [static]

Delete a device from a devclass.

For internal use only.

The device is removed from the devclass's device list and its unit number is freed.

Parameters:

dc the devclass to delete from

dev the device to delete

Return values:

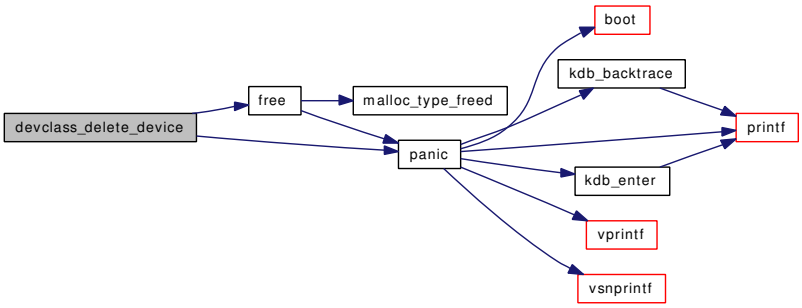
0 success

Definition at line 1401 of file subr_bus.c.

References DEVCLANAME, DEVICENAME, DF_WILDCARD, free(), panic(), and PDEBUG.

Referenced by device_delete_child(), device_detach(), device_set_devclass(), and device_set_unit().

Here is the call graph for this function:



9.87.4.60 int devclass_delete_driver (devclass_t *busclass*, driver_t * *driver*)

Delete a device driver from a device class.

Delete a device driver from a devclass. This is normally called automatically by DRIVER_MODULE().

If the driver is currently attached to any devices, devclass_delete_driver() will first attempt to detach from each device. If one of the detach calls fails, the driver will not be deleted.

Parameters:

dc the devclass to edit

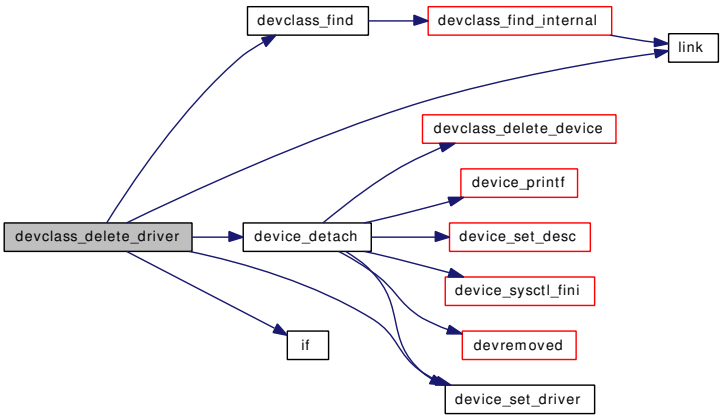
driver the driver to unregister

Definition at line 896 of file subr_bus.c.

References DEVCLANAME, devclass_find(), device_detach(), device_set_driver(), driverlink::driver, if(), link(), and PDEBUG.

Referenced by driver_module_handler().

Here is the call graph for this function:



9.87.4.61 devclass_t devclass_find (const char * *classname*)

Find a device class.

If a device class with the name `classname` exists, return it, otherwise return `NULL`.

Parameters:

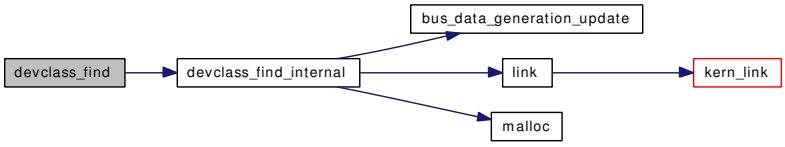
classname the devclass name to find

Definition at line 825 of file subr_bus.c.

References devclass_find_internal().

Referenced by devclass_delete_driver(), devclass_quiesce_driver(), and device_find_child().

Here is the call graph for this function:



9.87.4.62 kobj_class_t devclass_find_driver (devclass_t *dc*, const char * *classname*)

Search a devclass for a driver.

This function searches the devclass's list of drivers and returns the first driver whose name is `classname` or `NULL` if there is no driver of that name.

Parameters:

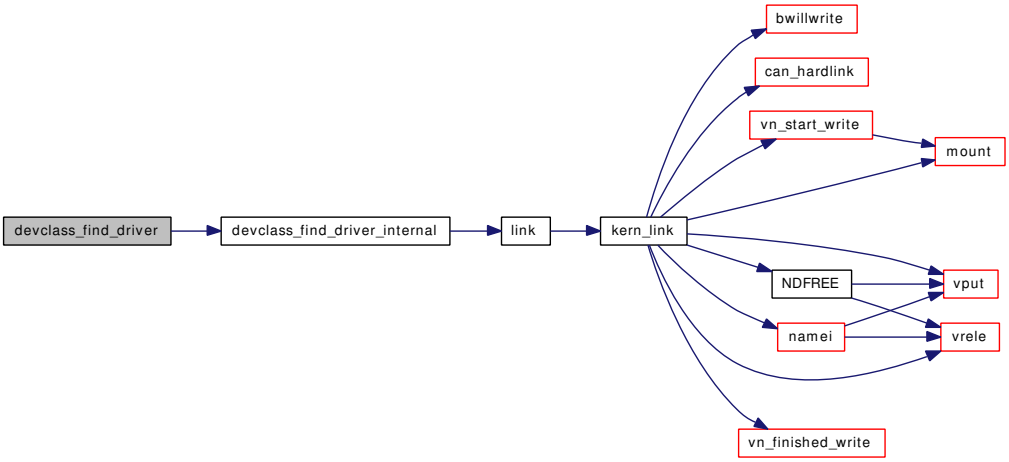
dc the devclass to search

classname the driver name to search for

Definition at line 1052 of file subr_bus.c.

References devclass_find_driver_internal(), and driverlink::driver.

Here is the call graph for this function:



9.87.4.63 static driverlink_t devclass_find_driver_internal (devclass_t *dc*, const char * *classname*) [static]

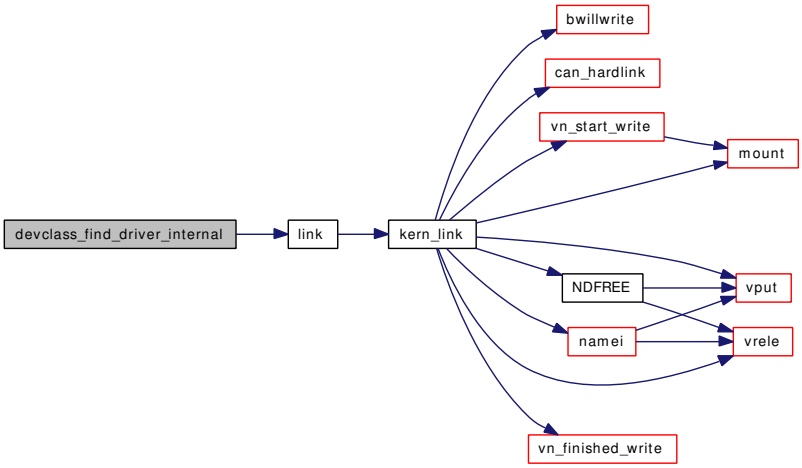
For internal use only.

Definition at line 1026 of file subr_bus.c.

References DEVCLANAME, driverlink::driver, link(), and PDEBUG.

Referenced by devclass_find_driver(), and first_matching_driver().

Here is the call graph for this function:



9.87.4.64 int devclass_find_free_unit (devclass_t *dc*, int *unit*)

Find a free unit number in a devclass.

This function searches for the first unused unit number greater that or equal to `unit`.

Parameters:

dc the devclass to examine
unit the first unit number to check

Definition at line 1233 of file subr_bus.c.

9.87.4.65 static devclass_t devclass_find_internal (const char * *classname*, const char * *parentname*, int *create*) [static]

Find or create a device class.

For internal use only.

If a device class with the name `classname` exists, return it, otherwise if `create` is non-zero create and return a new device class.
If `parentname` is non-NULL, the parent of the devclass is set to the devclass of that name.

Parameters:

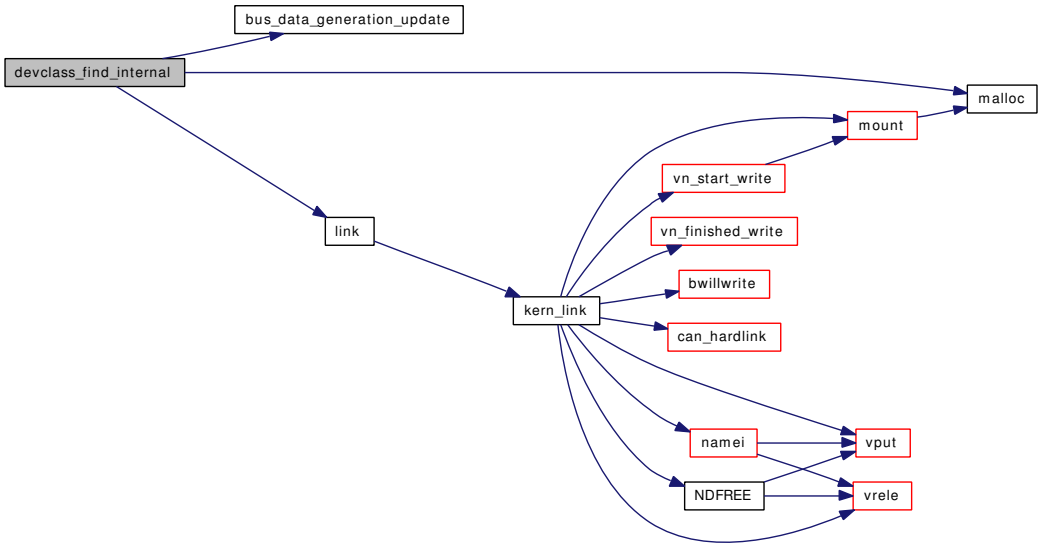
classname the devclass name to find or create
parentname the parent devclass name or `NULL`
create non-zero to create a devclass

Definition at line 757 of file subr_bus.c.

References bus_data_generation_update(), devclasses, link(), malloc(), and PDEBUG.

Referenced by devclass_add_driver(), devclass_create(), devclass_find(), device_set_devclass(), driver_-module_handler(), make_device(), and root_bus_module_handler().

Here is the call graph for this function:



9.87.4.66 int devclass_get_count (devclass_t *dc*)

Get the number of devices in a devclass.

Parameters:

dc the devclass to examine

Definition at line 1198 of file subr_bus.c.

Referenced by cpufreq_attach(), cpufreq_detach(), and devclass_get_devices().

9.87.4.67 device_t devclass_get_device (devclass_t *dc*, int *unit*)

Find a device given a unit number.

Parameters:

dc the devclass to search

unit the unit number to search for

Returns:

the device with the given unit number or `NULL` if there is no such device

Definition at line 1081 of file subr_bus.c.

Referenced by devclass_get_softc(), and device_find_child().

9.87.4.68 int devclass_get_devices (devclass_t *dc*, device_t ** *devlistp*, int * *devcountp*)

Get a list of devices in the devclass.

An array containing a list of all the devices in the given devclass is allocated and returned in `*devlistp`. The number of devices in the array is returned in `*devcountp`. The caller should free the array using `free(p, M_TEMP)`, even if `*devcountp` is 0.

Parameters:

dc the devclass to examine

devlistp points at location for array pointer return value

devcountp points at location for array size return value

Return values:

0 success

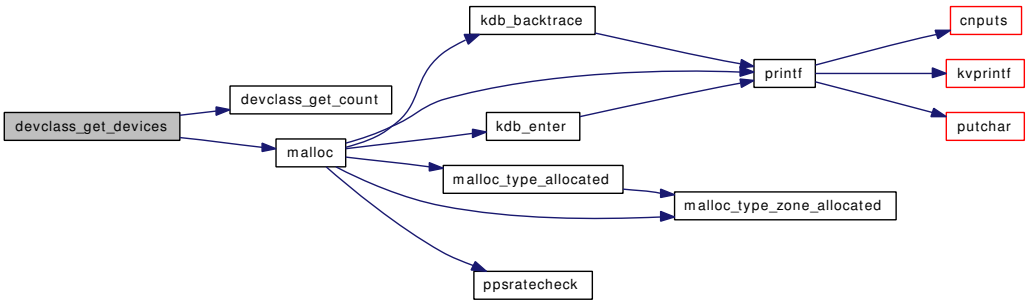
ENOMEM the array allocation failed

Definition at line 1127 of file subr_bus.c.

References devclass_get_count(), and malloc().

Referenced by cpufreq_curr_sysctl().

Here is the call graph for this function:



9.87.4.69 int devclass_get_drivers (devclass_t *dc*, driver_t *** *listp*, int * *countp*)

Get a list of drivers in the devclass.

An array containing a list of pointers to all the drivers in the given devclass is allocated and returned in `*listp`. The number of drivers in the array is returned in `*countp`. The caller should free the array using `free(p, M_TEMP)`.

Parameters:

dc the devclass to examine

listp gives location for array pointer return value

countp gives location for number of array elements return value

Return values:

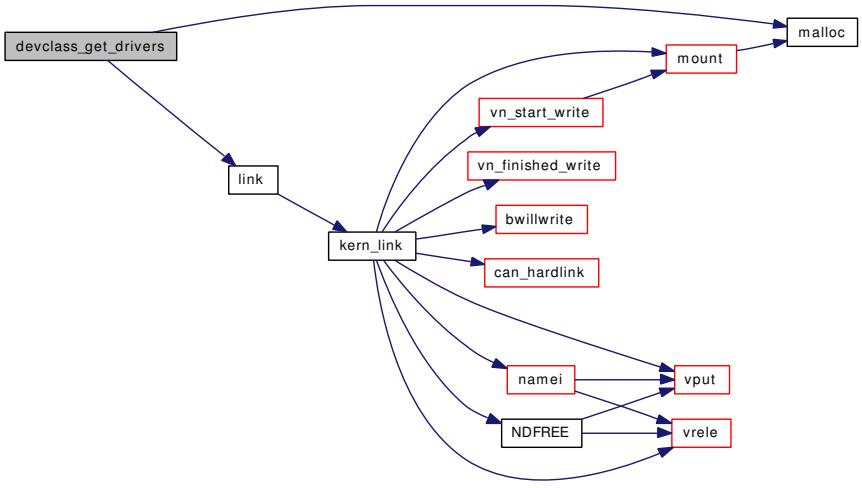
0 success

ENOMEM the array allocation failed

Definition at line 1168 of file subr_bus.c.

References driverlink::driver, link(), and malloc().

Here is the call graph for this function:



9.87.4.70 int devclass_get_maxunit (devclass_t *dc*)

Get the maximum unit number used in a devclass.

Note that this is one greater than the highest currently-allocated unit.

Parameters:

dc the devclass to examine

Definition at line 1218 of file subr_bus.c.

Referenced by device_find_child().

9.87.4.71 const char* devclass_get_name (devclass_t *dc*)

Return the name of the devclass.

Definition at line 1066 of file subr_bus.c.

Referenced by device_get_name().

9.87.4.72 devclass_t devclass_get_parent (devclass_t *dc*)

Get the parent of a devclass.

Parameters:

dc the devclass to examine

Definition at line 1263 of file subr_bus.c.

9.87.4.73 void* devclass_get_softc (devclass_t *dc*, int *unit*)

Find the softc field of a device given a unit number.

Parameters:

dc the devclass to search

unit the unit number to search for

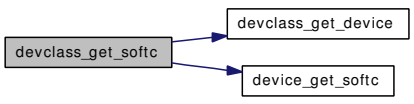
Returns:

the softc field of the device with the given unit number or `NULL` if there is no such device

Definition at line 1099 of file subr_bus.c.

References devclass_get_device(), and device_get_softc().

Here is the call graph for this function:



9.87.4.74 struct sysctl_ctx_list* devclass_get_sysctl_ctx (devclass_t *dc*)

Definition at line 1269 of file subr_bus.c.

9.87.4.75 struct sysctl_oid* devclass_get_sysctl_tree (devclass_t *dc*)

Definition at line 1275 of file subr_bus.c.

9.87.4.76 int devclass_quiesce_driver (devclass_t *busclass*, driver_t * *driver*)

Quiesces a set of device drivers from a device class.

Quiesce a device driver from a devclass. This is normally called automatically by DRIVER_MODULE().

If the driver is currently attached to any devices, devclass_quiesece_driver() will first attempt to quiesce each device.

Parameters:

dc the devclass to edit

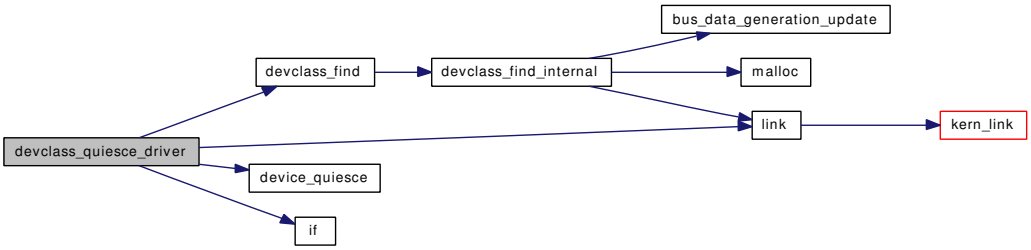
driver the driver to unregister

Definition at line 971 of file subr_bus.c.

References DEVCLANAME, devclass_find(), device_quiesce(), driverlink::driver, if(), link(), and PDEBUG.

Referenced by driver_module_handler().

Here is the call graph for this function:



9.87.4.77 void devclass_set_parent (devclass_t *dc*, devclass_t *pdc*)

Set the parent of a devclass.

The parent class is normally initialised automatically by DRIVER_MODULE().

Parameters:

dc the devclass to edit

pdc the new parent devclass

Definition at line 1252 of file subr_bus.c.

9.87.4.78 static int devclass_sysctl_handler (SYSCTL_HANDLER_ARGS) [static]

Definition at line 199 of file subr_bus.c.

References DEVCLASS_SYSCTL_PARENT.

Referenced by devclass_sysctl_init().

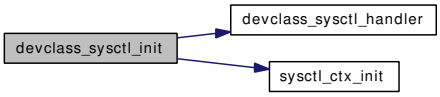
9.87.4.79 static void devclass_sysctl_init (devclass_t *dc*) [static]

Definition at line 215 of file subr_bus.c.

References devclass_sysctl_handler(), DEVCLASS_SYSCTL_PARENT, and sysctl_ctx_init().

Referenced by device_sysctl_init().

Here is the call graph for this function:



9.87.4.80 static int devclose (struct cdev * *dev*, int *fflag*, int *devtype*, d_thread_t * *td*) [static]

Definition at line 406 of file subr_bus.c.

References dev_softc::cv, devsoftc, dev_softc::inuse, and dev_softc::mtx.

9.87.4.81 void devctl_notify (const char * *system*, const char * *subsystem*, const char * *type*, const char * *data*)

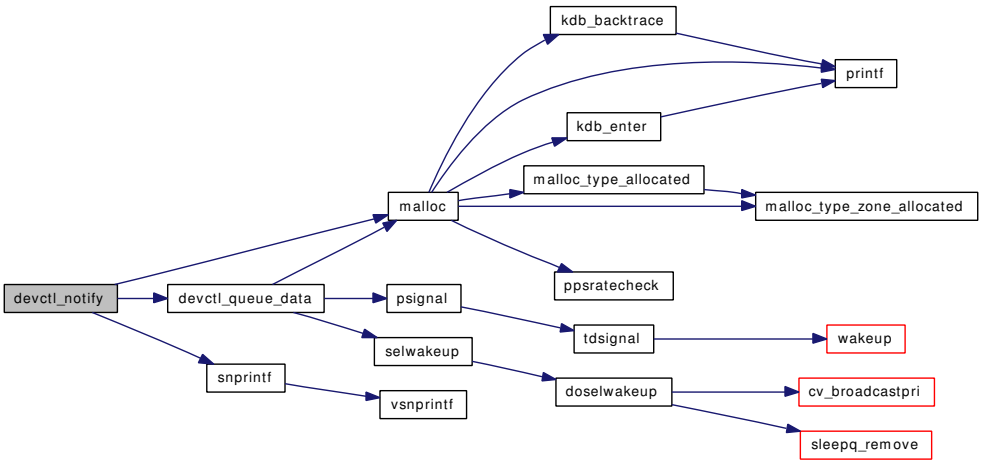
Send a 'notification' to userland, using standard ways.

Definition at line 535 of file subr_bus.c.

References devctl_queue_data(), malloc(), and snprintf().

Referenced by root_resume().

Here is the call graph for this function:



9.87.4.82 void devctl_queue_data (char * *data*)

Queue data to be read from the devctl device.

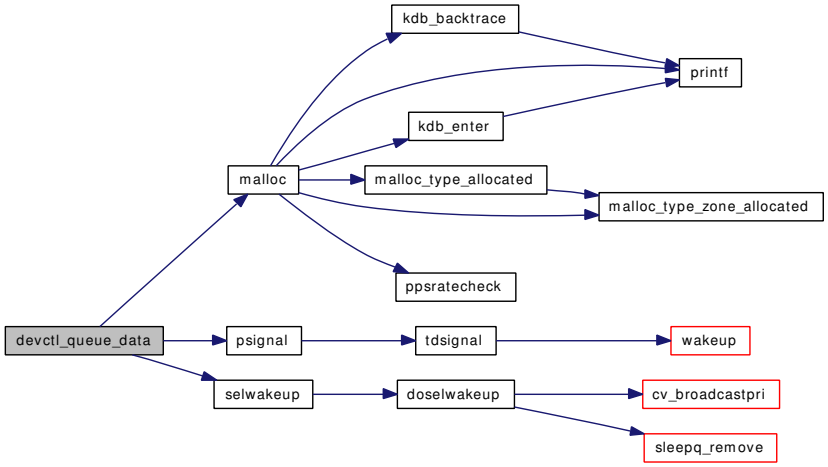
Generic interface to queue data to the devctl device. It is assumed that `data` is properly formatted. It is further assumed that `data` is allocated using the M_BUS malloc type.

Definition at line 509 of file subr_bus.c.

References dev_softc::async_proc, dev_softc::cv, dev_softc::devq, devsoftc, malloc(), dev_softc::mtx, psignal(), dev_softc::sel, and selwakeup().

Referenced by devaddq(), and devctl_notify().

Here is the call graph for this function:



9.87.4.83 device_t device_add_child (device_t *dev*, const char * *name*, int *unit*)

Create a new device.

This creates a new device and adds it as a child of an existing parent device. The new device will be added after the last existing child with order zero.

Parameters:

dev the device which will be the parent of the new child device

name devclass name for new device or `NULL` if not specified

unit unit number for new device or `-1` if not specified

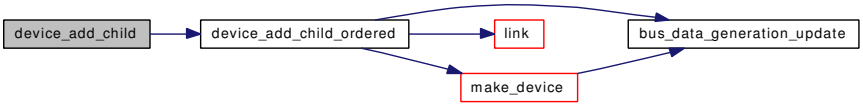
Returns:

the new device

Definition at line 1521 of file subr_bus.c.

References device_add_child_ordered().

Here is the call graph for this function:



9.87.4.84 device_t device_add_child_ordered (device_t *dev*, int *order*, const char * *name*, int *unit*)

Create a new device.

This creates a new device and adds it as a child of an existing parent device. The new device will be added after the last existing child with the same order.

Parameters:

dev the device which will be the parent of the new child device

order a value which is used to partially sort the children of `dev` - devices created using lower values of `order` appear first in `dev's` list of children

name devclass name for new device or `NULL` if not specified

unit unit number for new device or `-1` if not specified

Returns:

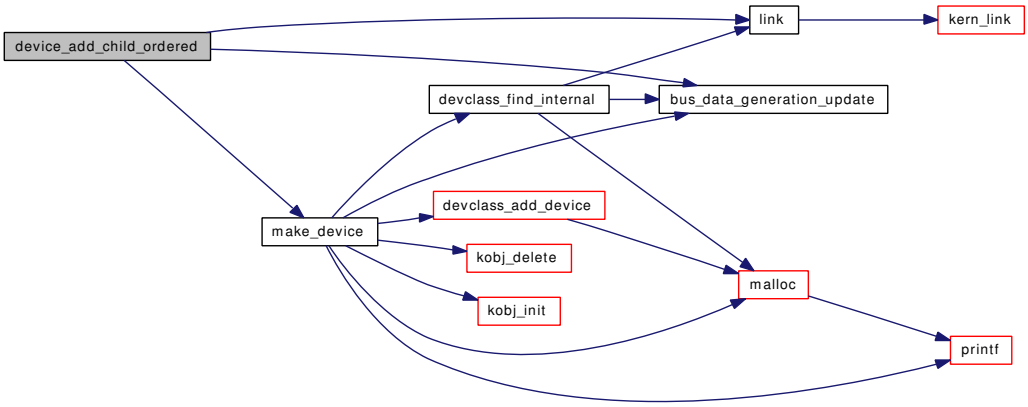
the new device

Definition at line 1547 of file subr_bus.c.

References bus_data_generation_update(), DEVICENAME, link(), make_device(), and PDEBUG.

Referenced by bus_generic_add_child(), and device_add_child().

Here is the call graph for this function:



9.87.4.85 int device_attach (device_t *dev*)

Attach a device driver to a device.

This function is a wrapper around the DEVICE_ATTACH() driver method. In addition to calling DEVICE_ATTACH(), it initialises the device's sysctl tree, optionally prints a description of the device and queues a notification event for user-based device management services.

Normally this function is only called internally from device_probe_and_attach().

Parameters:

dev the device to initialise

Return values:

0 success

ENXIO no driver was found

ENOMEM memory allocation failure

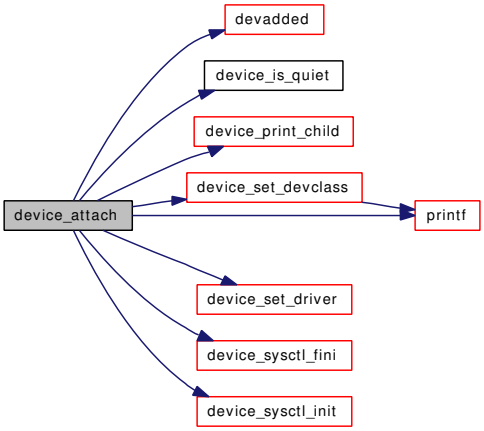
non-zero some other unix error code

Definition at line 2372 of file subr_bus.c.

References devadded(), device_is_quiet(), device_print_child(), device_set_devclass(), device_set_driver(), device_sysctl_fini(), device_sysctl_init(), and printf().

Referenced by device_probe_and_attach().

Here is the call graph for this function:



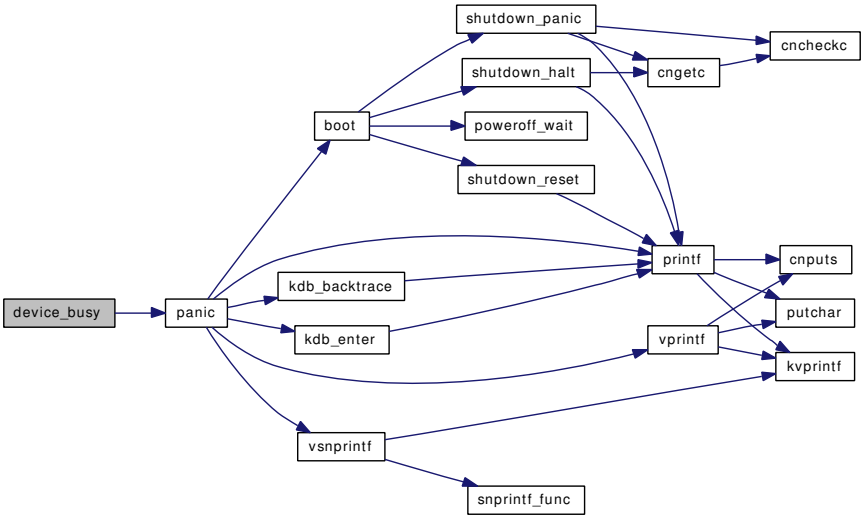
9.87.4.86 void device_busy (device_t *dev*)

Increment the busy counter for the device.

Definition at line 2141 of file subr_bus.c.

References panic().

Here is the call graph for this function:



9.87.4.87 int device_delete_child (device_t *dev*, device_t *child*)

Delete a device.

This function deletes a device along with all of its children. If the device currently has a driver attached to it, the device is detached first using device_detach().

Parameters:

dev the parent device

child the device to delete

Return values:

0 success

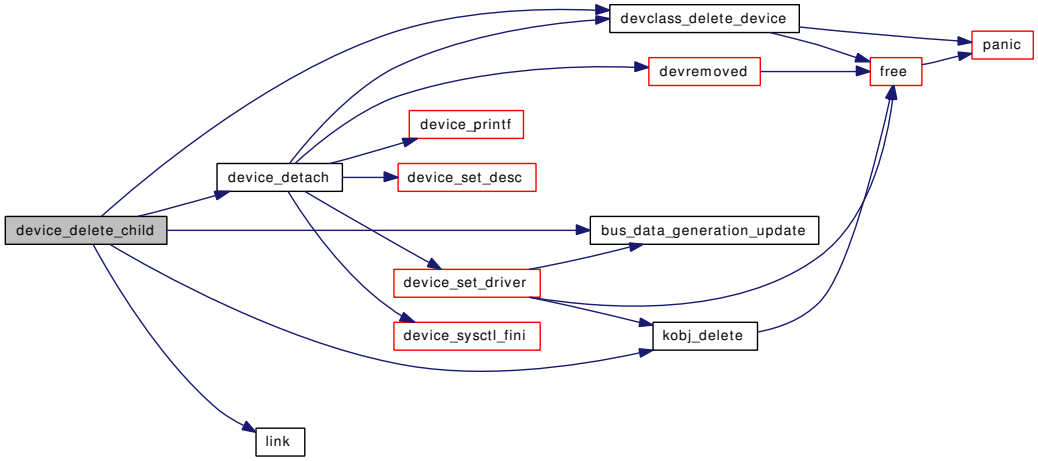
non-zero a unit error code describing the error

Definition at line 1597 of file subr_bus.c.

References bus_data_generation_update(), devclass_delete_device(), device_detach(), DEVICENAME, kobj_delete(), link(), and PDEBUG.

Referenced by cpufreq_unregister().

Here is the call graph for this function:



9.87.4.88 int device_detach (device_t *dev*)

Detach a driver from a device.

This function is a wrapper around the DEVICE_DETACH() driver method. If the call to DEVICE_-DETACH() succeeds, it calls BUS_CHILD_DETACHED() for the parent of `dev`, queues a notification event for user-based device management services and cleans up the device's sysctl tree.

Parameters:

dev the device to un-initialise

Return values:

0 success

ENXIO no driver was found

ENOMEM memory allocation failure

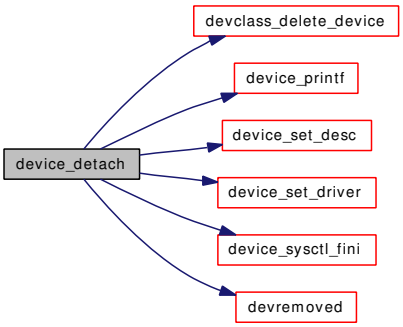
non-zero some other unix error code

Definition at line 2412 of file subr_bus.c.

References devclass_delete_device(), device_printf(), device_set_desc(), device_set_driver(), device_sysctl_fini(), DEVICENAME, devremoved(), DF_FIXEDCLASS, and PDEBUG.

Referenced by bus_generic_detach(), devclass_delete_driver(), device_delete_child(), and device_probe_child().

Here is the call graph for this function:



9.87.4.89 void device_disable (device_t *dev*)

Clear the DF_ENABLED flag for the device.

Definition at line 2132 of file subr_bus.c.

References DF_ENABLED.

9.87.4.90 void device_enable (device_t *dev*)

Set the DF_ENABLED flag for the device.

Definition at line 2123 of file subr_bus.c.

References DF_ENABLED.

9.87.4.91 device_t device_find_child (device_t *dev*, const char * *classname*, int *unit*)

Find a device given a unit number.

This is similar to devclass_get_devices() but only searches for devices which have `dev` as a parent.

Parameters:

dev the parent device to search

unit the unit number to search for. If the unit is -1, return the first child of `dev` which has name `classname` (that is, the one with the lowest unit.)

Returns:

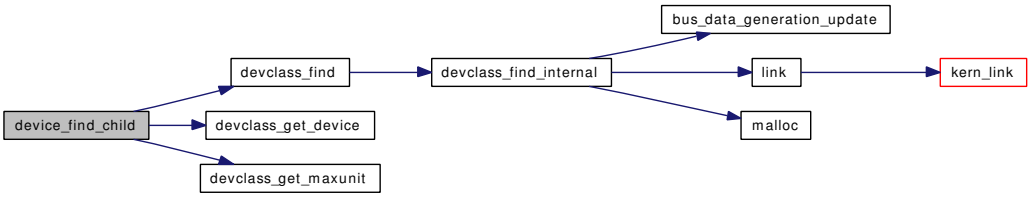
the device with the given unit number or `NULL` if there is no such device

Definition at line 1638 of file subr_bus.c.

References devclass_find(), devclass_get_device(), and devclass_get_maxunit().

Referenced by cpufreq_register(), and cpufreq_unregister().

Here is the call graph for this function:



9.87.4.92 int device_get_children (device_t *dev*, device_t ** *devlistp*, int * *devcountp*)

Get a list of children of a device.

An array containing a list of all the children of the given device is allocated and returned in `*devlistp`. The number of devices in the array is returned in `*devcountp`. The caller should free the array using `free(p, M_TEMP)`.

Parameters:

dev the device to examine

devlistp points at location for array pointer return value

devcountp points at location for array size return value

Return values:

0 success

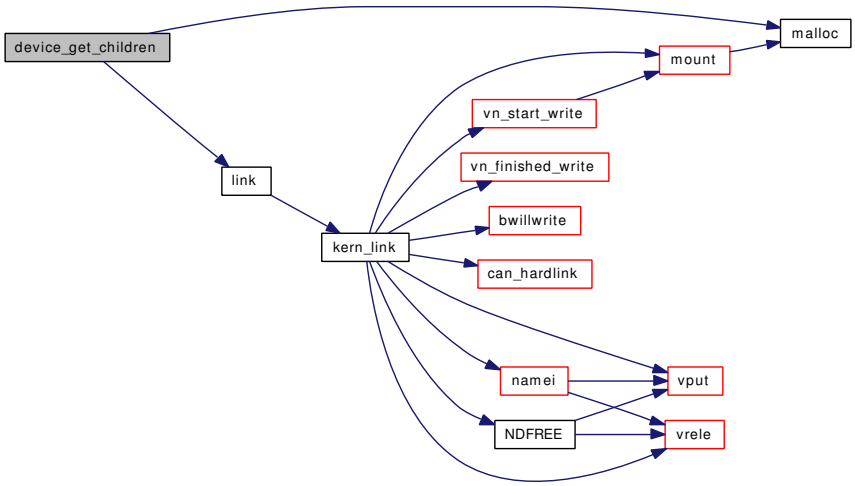
ENOMEM the array allocation failed

Definition at line 1850 of file subr_bus.c.

References link(), and malloc().

Referenced by cf_get_method(), cf_levels_method(), and cpufreq_unregister().

Here is the call graph for this function:



9.87.4.93 const char* device_get_desc (device_t *dev*)

Return the device's description string.

Definition at line 1933 of file subr_bus.c.

Referenced by bus_print_child_header().

9.87.4.94 devclass_t device_get_devclass (device_t *dev*)

Return the current devclass for the device or `NULL` if there is none.

Definition at line 1892 of file subr_bus.c.

Referenced by device_set_unit().

9.87.4.95 driver_t* device_get_driver (device_t *dev*)

Return the current driver for the device or `NULL` if there is no driver currently attached.

Definition at line 1882 of file subr_bus.c.

9.87.4.96 u_int32_t device_get_flags (device_t *dev*)

Return the device's flags.

Definition at line 1942 of file subr_bus.c.

9.87.4.97 void* device_get_ivars (device_t *dev*)

Get the device's ivars field.

The ivars field is used by the parent device to store per-device state (e.g. the physical location of the device or a list of resources).

Definition at line 2092 of file subr_bus.c.

9.87.4.98 const char* device_get_name (device_t *dev*)

Return the name of the device's devclass or `NULL` if there is none.

Definition at line 1902 of file subr_bus.c.

References devclass_get_name().

Referenced by bus_enumerate_hinted_children(), clock_register(), device_print_prettyname(), device_probe_and_attach(), and sysctl_rman().

Here is the call graph for this function:

9.87.4.99 const char* device_get_nameunit (device_t *dev*)

Return a string containing the device's devclass name followed by an ascii representation of the device's unit number (e.g. `"foo2"`).

Definition at line 1915 of file subr_bus.c.

Referenced by bus_enumerate_hinted_children(), bus_print_child_footer(), bus_print_child_header(), cf_levels_method(), cf_set_method(), cpufreq_attach(), cpufreq_detach(), cpufreq_dup_set(), devadded(), devaddq(), device_unbusy(), devremoved(), and rman_reserve_resource_bound().

9.87.4.100 device_t device_get_parent (device_t *dev*)

Return the parent of a device.

Definition at line 1828 of file subr_bus.c.

Referenced by bus_child_location_str(), bus_child_pnpinfo_str(), bus_child_present(), bus_delete_resource(), bus_generic_child_present(), bus_get_dma_tag(), bus_get_resource(), bus_get_resource_count(), bus_get_resource_start(), bus_set_resource(), cf_get_method(), cf_levels_method(), cpufreq_attach(), cpufreq_register(), cpufreq_unregister(), devaddq(), resource_list_alloc(), and resource_list_release().

9.87.4.101 void* device_get_softc (device_t *dev*)

Return the device's softc field.

The softc is allocated and zeroed when a driver is attached, based on the size field of the driver.

Definition at line 2061 of file subr_bus.c.

Referenced by cf_get_method(), cf_levels_method(), cf_set_method(), cpufreq_attach(), cpufreq_detach(), cpufreq_register(), and devclass_get_softc().

9.87.4.102 device_state_t device_get_state (device_t *dev*)

Return the device's state.

Definition at line 2114 of file subr_bus.c.

9.87.4.103 struct sysctl_ctx_list* device_get_sysctl_ctx (device_t *dev*)

Definition at line 1948 of file subr_bus.c.

Referenced by cpufreq_register().

9.87.4.104 struct sysctl_oid* device_get_sysctl_tree (device_t *dev*)

Definition at line 1954 of file subr_bus.c.

Referenced by cpufreq_attach(), and cpufreq_register().

9.87.4.105 int device_get_unit (device_t *dev*)

Return the device's unit number.

Definition at line 1924 of file subr_bus.c.

Referenced by device_print_prettyname(), and sysctl_rman().

9.87.4.106 int device_is_alive (device_t *dev*)

Return non-zero if the device was successfully probed.

Definition at line 2208 of file subr_bus.c.

Referenced by device_print_child().

9.87.4.107 int device_is_attached (device_t *dev*)

Return non-zero if the device currently has a driver attached to it.

Definition at line 2218 of file subr_bus.c.

Referenced by cf_get_method(), cf_levels_method(), cf_set_method(), and cpufreq_unregister().

9.87.4.108 int device_is_enabled (device_t *dev*)

Return non-zero if the DF_ENABLED flag is set on the device.

Definition at line 2199 of file subr_bus.c.

References DF_ENABLED.

9.87.4.109 int device_is_quiet (device_t *dev*)

Return non-zero if the DF_QUIET flag is set on the device.

Definition at line 2190 of file subr_bus.c.

References DF_QUIET.

Referenced by device_attach().

9.87.4.110 static int device_print_child (device_t *dev*, device_t *child*) [static]

Print a description of a device.

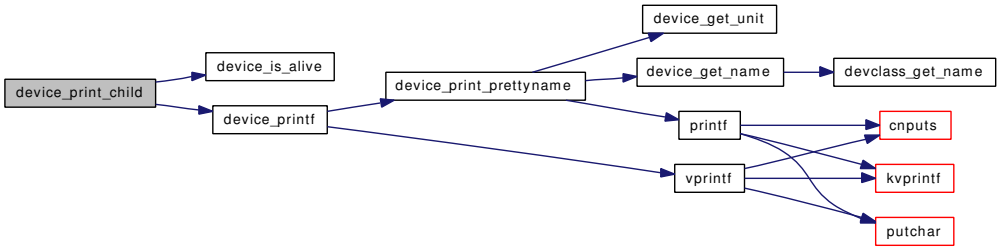
For internal use only.

Definition at line 1492 of file subr_bus.c.

References device_is_alive(), and device_printf().

Referenced by device_attach().

Here is the call graph for this function:



9.87.4.111 int device_print_prettyname (device_t *dev*)

Print the name of the device followed by a colon and a space.

Returns:

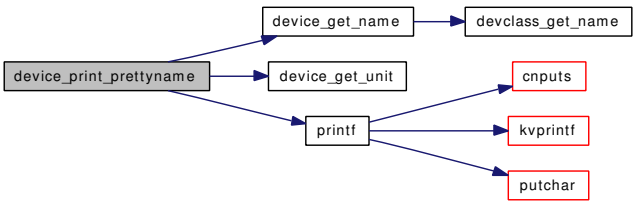
the number of characters printed

Definition at line 1965 of file subr_bus.c.

References device_get_name(), device_get_unit(), and printf().

Referenced by device_printf(), and device_probe_and_attach().

Here is the call graph for this function:



9.87.4.112 int device_printf (device_t *dev*, const char * *fmt*, ...)

Print the name of the device followed by a colon, a space and the result of calling vprintf() with the value of `fmt` and the following arguments.

Returns:

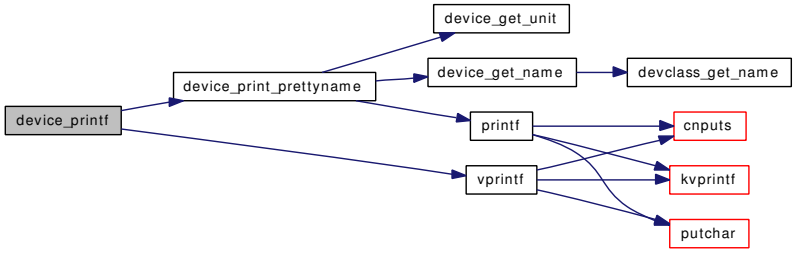
the number of characters printed

Definition at line 1982 of file subr_bus.c.

References device_print_prettyname(), and vprintf().

Referenced by bus_print_child_header(), bus_setup_intr(), clock_register(), cpufreq_unregister(), device_-detach(), and device_print_child().

Here is the call graph for this function:



9.87.4.113 int device_probe_and_attach (device_t *dev*)

Probe a device and attach a driver if possible.

This function is the core of the device autoconfiguration system. Its purpose is to select a suitable driver for a device and then call that driver to initialise the hardware appropriately. The driver is selected by calling the DEVICE_PROBE() method of a set of candidate drivers and then choosing the driver which returned the best value. This driver is then attached to the device using device_attach().

The set of suitable drivers is taken from the list of drivers in the parent device's devclass. If the device was originally created with a specific class name (see device_add_child()), only drivers with that name are probed, otherwise all drivers in the devclass are probed. If no drivers return successful probe values in the parent devclass, the search continues in the parent of that devclass (see devclass_get_parent()) if any.

Parameters:

dev the device to initialise

Return values:

0 success

ENXIO no driver was found

ENOMEM memory allocation failure

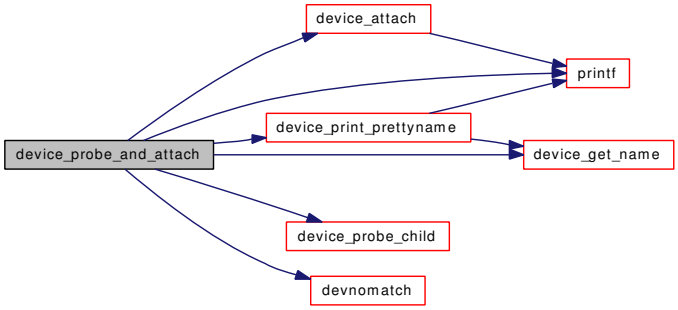
non-zero some other unix error code

Definition at line 2323 of file subr_bus.c.

References bootverbose, device_attach(), device_get_name(), device_print_prettyname(), device_probe_-child(), devnomatch(), DF_DONENOMATCH, DF_ENABLED, DF_REBID, and printf().

Referenced by bus_generic_attach(), bus_generic_driver_added(), cpufreq_register(), and root_bus_-configure().

Here is the call graph for this function:



9.87.4.114 int device_probe_child (device_t *dev*, device_t *child*)

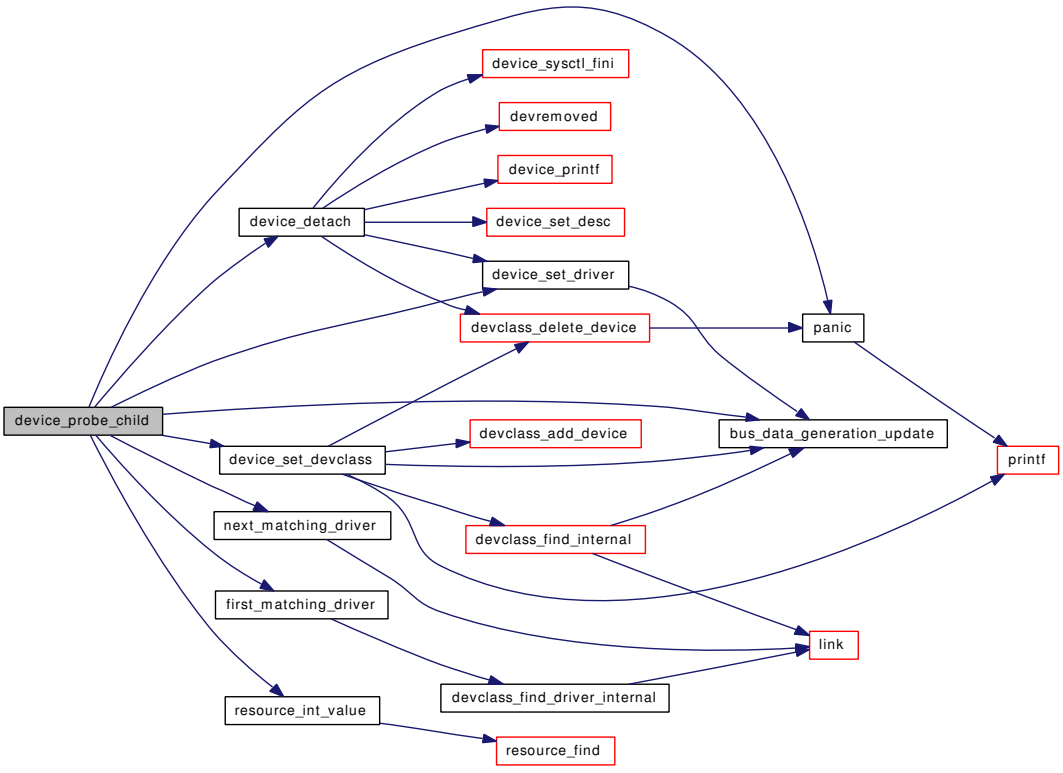
For internal use only.

Definition at line 1692 of file subr_bus.c.

References bus_data_generation_update(), device_detach(), device_set_devclass(), device_set_driver(), DF_REBID, driverlink::driver, DRIVERNAME, first_matching_driver(), next_matching_driver(), panic(), PDEBUG, and resource_int_value().

Referenced by device_probe_and_attach().

Here is the call graph for this function:



9.87.4.115 int device_quiesce (device_t *dev*)

Tells a driver to quiesce itself.

This function is a wrapper around the DEVICE_QUIESCE() driver method. If the call to DEVICE_-QUIESCE() succeeds.

Parameters:

dev the device to quiesce

Return values:

0 success

ENXIO no driver was found

ENOMEM memory allocation failure

non-zero some other unix error code

Definition at line 2456 of file subr_bus.c.

References DEVICENAME, and PDEBUG.

Referenced by devclass_quiesce_driver().

9.87.4.116 void device_quiet (device_t *dev*)

Set the DF_QUIET flag for the device.

Definition at line 2172 of file subr_bus.c.

References DF_QUIET.

Referenced by cpufreq_register().

9.87.4.117 void device_set_desc (device_t *dev*, const char * *desc*)

Set the device's description.

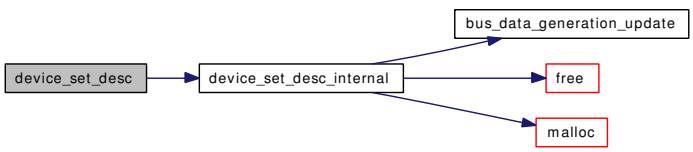
The value of `desc` should be a string constant that will not change (at least until the description is changed in a subsequent call to device_set_desc() or device_set_desc_copy()).

Definition at line 2028 of file subr_bus.c.

References device_set_desc_internal().

Referenced by device_detach().

Here is the call graph for this function:



9.87.4.118 void device_set_desc_copy (device_t *dev*, const char * *desc*)

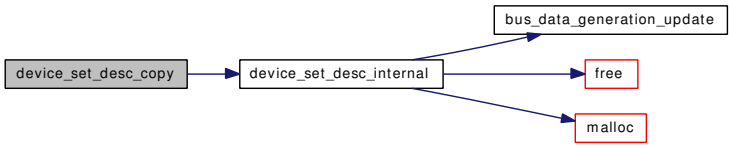
Set the device's description.

The string pointed to by `desc` is copied. Use this function if the device description is generated, (e.g. with sprintf()).

Definition at line 2040 of file subr_bus.c.

References device_set_desc_internal().

Here is the call graph for this function:



9.87.4.119 static void device_set_desc_internal (device_t *dev*, const char * *desc*, int *copy*) [static]

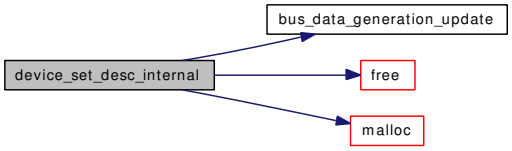
For internal use only.

Definition at line 1998 of file subr_bus.c.

References bus_data_generation_update(), DF_DESCMALLOCED, free(), and malloc().

Referenced by device_set_desc(), and device_set_desc_copy().

Here is the call graph for this function:



9.87.4.120 int device_set_devclass (device_t *dev*, const char * *classname*)

Set the devclass of a device.

See also:

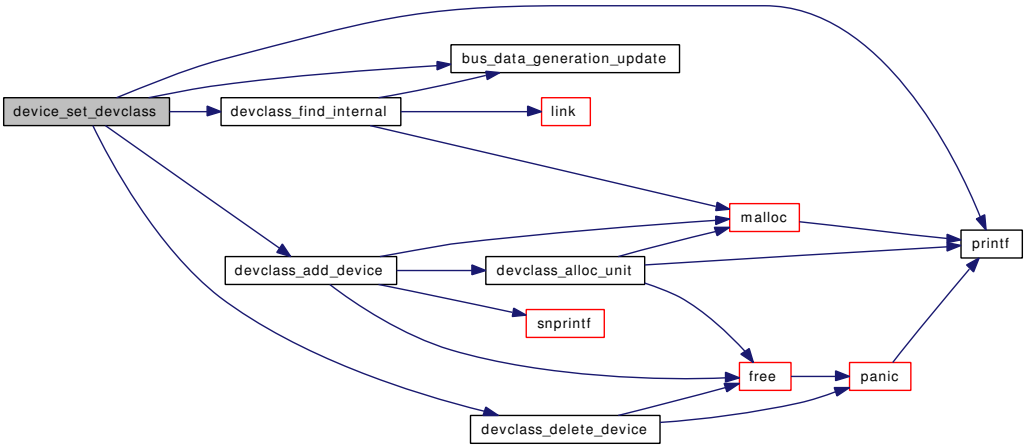
devclass_add_device().

Definition at line 2228 of file subr_bus.c.

References bus_data_generation_update(), devclass_add_device(), devclass_delete_device(), devclass_find_internal(), and printf().

Referenced by device_attach(), and device_probe_child().

Here is the call graph for this function:



9.87.4.121 int device_set_driver (device_t *dev*, driver_t * *driver*)

Set the driver of a device.

Return values:

0 success

EBUSY the device already has a driver attached

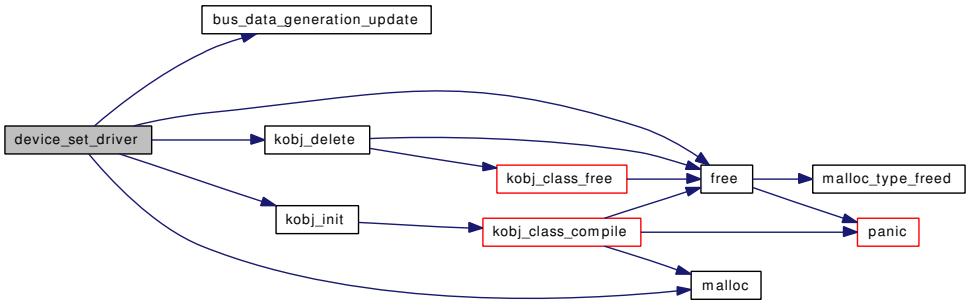
ENOMEM a memory allocation failure occurred

Definition at line 2262 of file subr_bus.c.

References bus_data_generation_update(), DF_EXTERNALSOFTC, free(), kobj_delete(), kobj_init(), and malloc().

Referenced by devclass_delete_driver(), device_attach(), device_detach(), and device_probe_child().

Here is the call graph for this function:



9.87.4.122 void device_set_flags (device_t *dev*, u_int32_t *flags*)

Set the device's flags.

Definition at line 2049 of file subr_bus.c.

9.87.4.123 void device_set_ivars (device_t *dev*, void * *ivars*)

Set the device's ivars field.

Definition at line 2103 of file subr_bus.c.

9.87.4.124 void device_set_softc (device_t *dev*, void * *softc*)

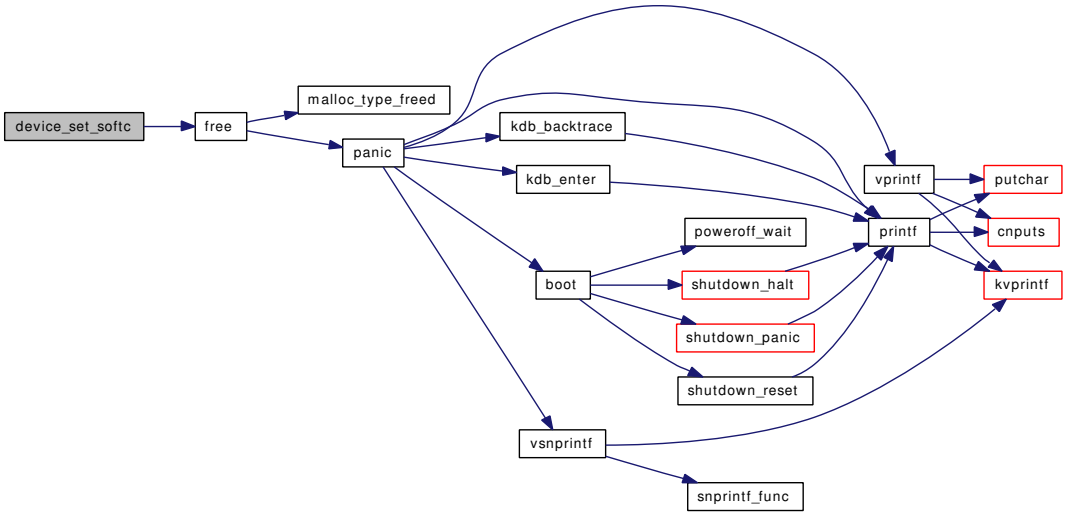
Set the device's softc field.

Most drivers do not need to use this since the softc is allocated automatically when the driver is attached.

Definition at line 2073 of file subr_bus.c.

References DF_EXTERNALSOFTC, and free().

Here is the call graph for this function:



9.87.4.125 int device_set_unit (device_t *dev*, int *unit*)

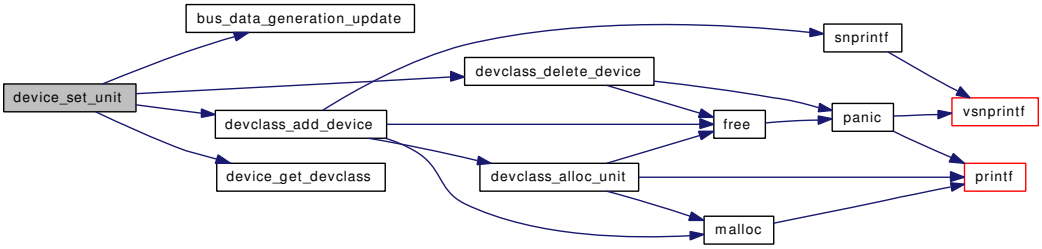
Set the unit number of a device.

This function can be used to override the unit number used for a device (e.g. to wire a device to a pre-configured unit number).

Definition at line 2491 of file subr_bus.c.

References bus_data_generation_update(), devclass_add_device(), devclass_delete_device(), and device_-get_devclass().

Here is the call graph for this function:



9.87.4.126 int device_shutdown (device_t *dev*)

Notify a device of system shutdown.

This function calls the DEVICE_SHUTDOWN() driver method if the device currently has an attached driver.

Returns:

the value returned by DEVICE_SHUTDOWN()

Definition at line 2477 of file subr_bus.c.

Referenced by bus_generic_shutdown(), and root_bus_module_handler().

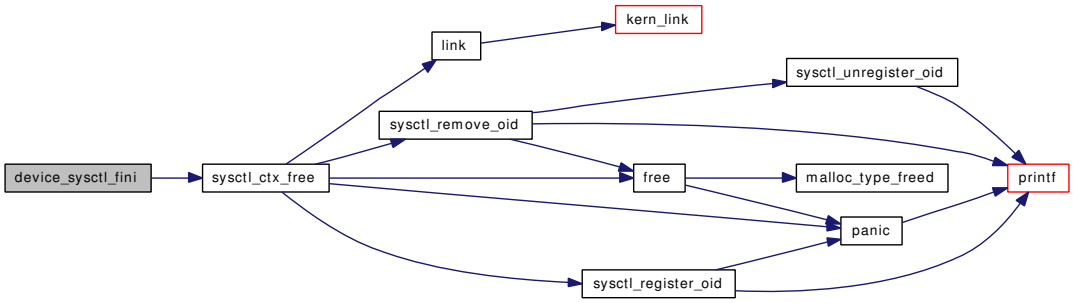
9.87.4.127 static void device_sysctl_fini (device_t *dev*) [static]

Definition at line 310 of file subr_bus.c.

References sysctl_ctx_free().

Referenced by device_attach(), and device_detach().

Here is the call graph for this function:



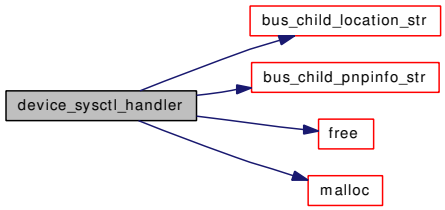
9.87.4.128 static int device_sysctl_handler (SYSCTL_HANDLER_ARGS) [static]

Definition at line 239 of file subr_bus.c.

References buf, bus_child_location_str(), bus_child_pnpinfo_str(), DEVICE_SYSCTL_DESC, DEVICE_SYSCTL_DRIVER, DEVICE_SYSCTL_LOCATION, DEVICE_SYSCTL_PARENT, DEVICE_SYSCTL_PNPINFO, free(), and malloc().

Referenced by device_sysctl_init().

Here is the call graph for this function:



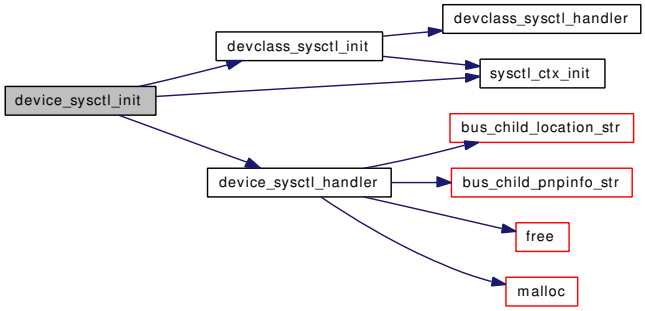
9.87.4.129 static void device_sysctl_init (device_t *dev*) [static]

Definition at line 275 of file subr_bus.c.

References devclass_sysctl_init(), DEVICE_SYSCTL_DESC, DEVICE_SYSCTL_DRIVER, device_sysctl_handler(), DEVICE_SYSCTL_LOCATION, DEVICE_SYSCTL_PARENT, DEVICE_SYSCTL_PNPINFO, and sysctl_ctx_init().

Referenced by device_attach().

Here is the call graph for this function:



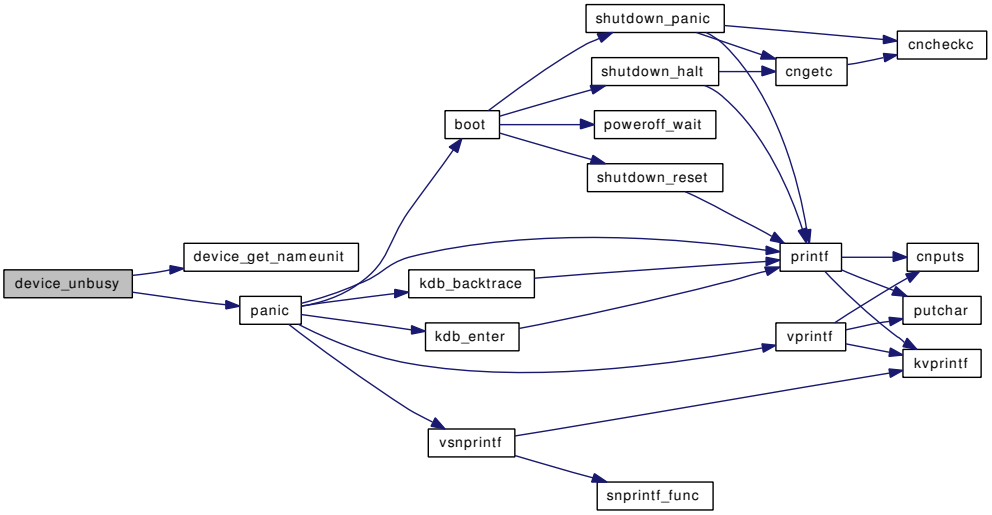
9.87.4.130 void device_unbusy (device_t *dev*)

Decrement the busy counter for the device.

Definition at line 2155 of file subr_bus.c.

References device_get_nameunit(), and panic().

Here is the call graph for this function:



9.87.4.131 void device_verbose (device_t *dev*)

Clear the DF_QUIET flag for the device.

Definition at line 2181 of file subr_bus.c.

References DF_QUIET.

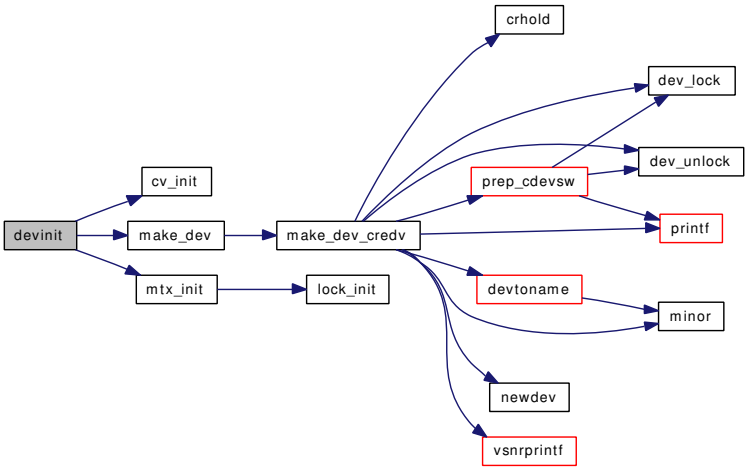
9.87.4.132 static void devinit (void) [static]

Definition at line 384 of file subr_bus.c.

References dev_softc::cv, cv_init(), devctl_dev, dev_softc::devq, devsoftc, make_dev(), dev_softc::mtx, and mtx_init().

Referenced by root_bus_module_handler().

Here is the call graph for this function:



9.87.4.133 static int devioctl (struct cdev * *dev*, u_long *cmd*, caddr_t *data*, int *fflag*, d_thread_t * *td*) [static]

Definition at line 455 of file subr_bus.c.

References dev_softc::async_proc, devsoftc, and dev_softc::nonblock.

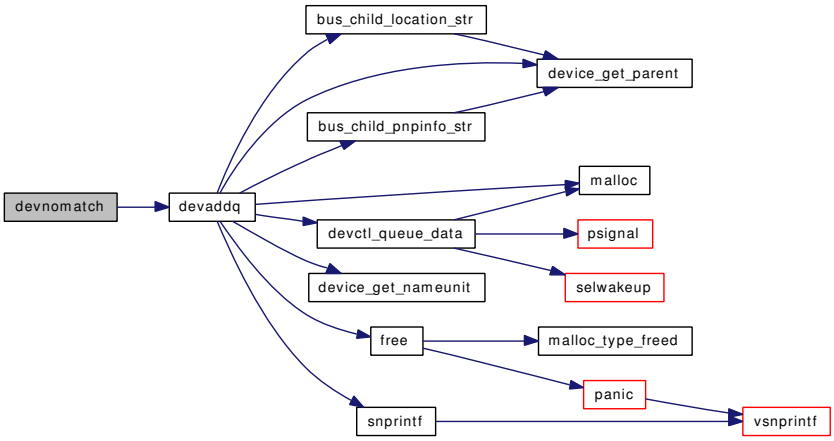
9.87.4.134 static void devnomatch (device_t *dev*) [static]

Definition at line 694 of file subr_bus.c.

References devaddq().

Referenced by device_probe_and_attach().

Here is the call graph for this function:



9.87.4.135 static int devopen (struct cdev * *dev*, int *oflags*, int *devtype*, d_thread_t * *td*) [static]

Definition at line 394 of file subr_bus.c.

References dev_softc::async_proc, devsoftc, dev_softc::inuse, and dev_softc::nonblock.

9.87.4.136 static int devpoll (struct cdev * *dev*, int *events*, d_thread_t * *td*) [static]

Definition at line 485 of file subr_bus.c.

References dev_softc::devq, devsoftc, dev_softc::mtx, dev_softc::sel, and selrecord().

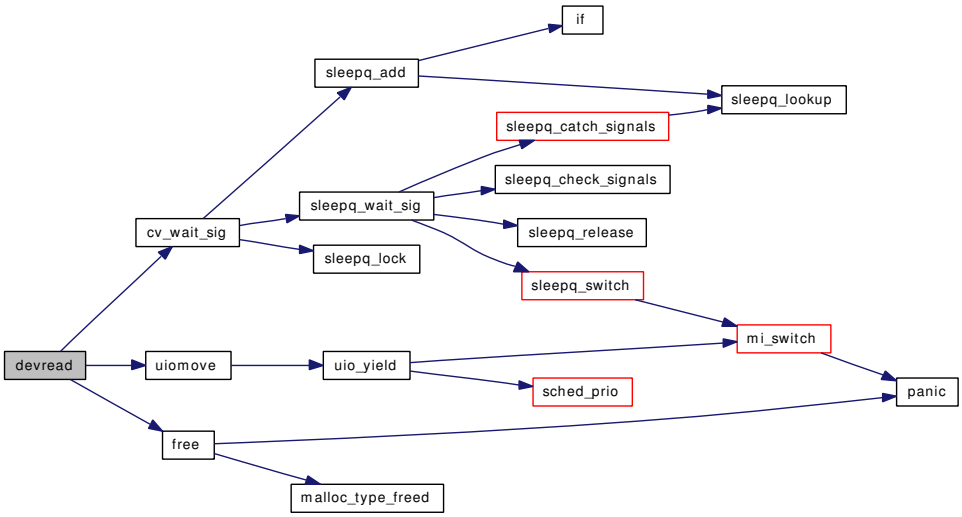
Here is the call graph for this function:

9.87.4.137 static int devread (struct cdev * *dev*, struct uio * *uio*, int *ioflag*) [static]

Definition at line 425 of file subr_bus.c.

References dev_softc::cv, cv_wait_sig(), dev_event_info::dei_data, dev_softc::devq, devsoftc, free(), dev_softc::mtx, dev_softc::nonblock, and uiomove().

Here is the call graph for this function:



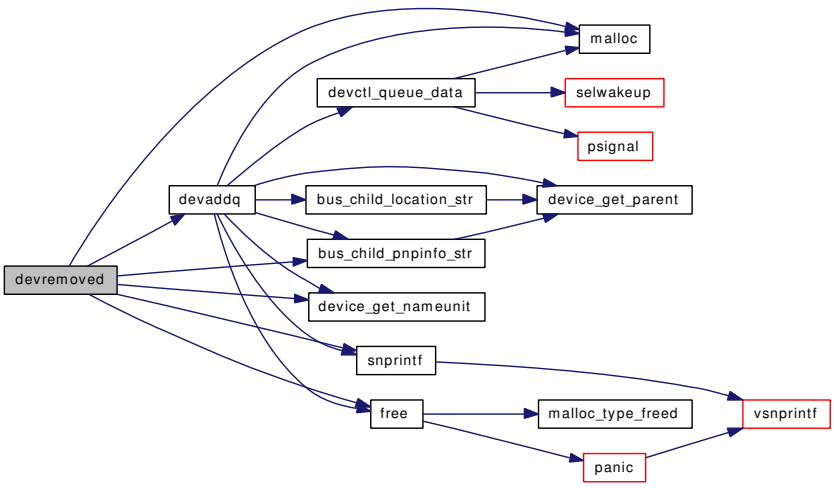
9.87.4.138 static void devremoved (device_t *dev*) [static]

Definition at line 663 of file subr_bus.c.

References bus_child_pnpinfo_str(), devaddq(), device_get_nameunit(), free(), malloc(), and snprintf().

Referenced by device_detach().

Here is the call graph for this function:



9.87.4.139 int driver_module_handler (module_t *mod*, int *what*, void * *arg*)

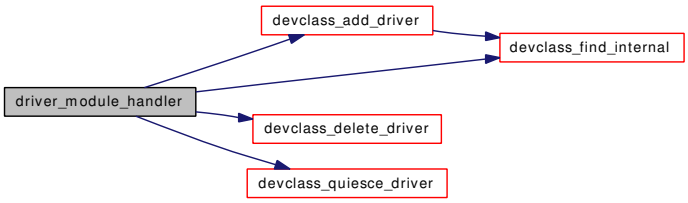
Module handler for registering device drivers.

This module handler is used to automatically register device drivers when modules are loaded. If `what` is MOD_LOAD, it calls devclass_add_driver() for the driver described by the driver_module_data structure pointed to by `arg`

Definition at line 3780 of file subr_bus.c.

References devclass_add_driver(), devclass_delete_driver(), devclass_find_internal(), devclass_quiesce_-driver(), DRIVERNAME, and PDEBUG.

Here is the call graph for this function:



9.87.4.140 static driverlink_t first_matching_driver (devclass_t *dc*, device_t *dev*) [static]

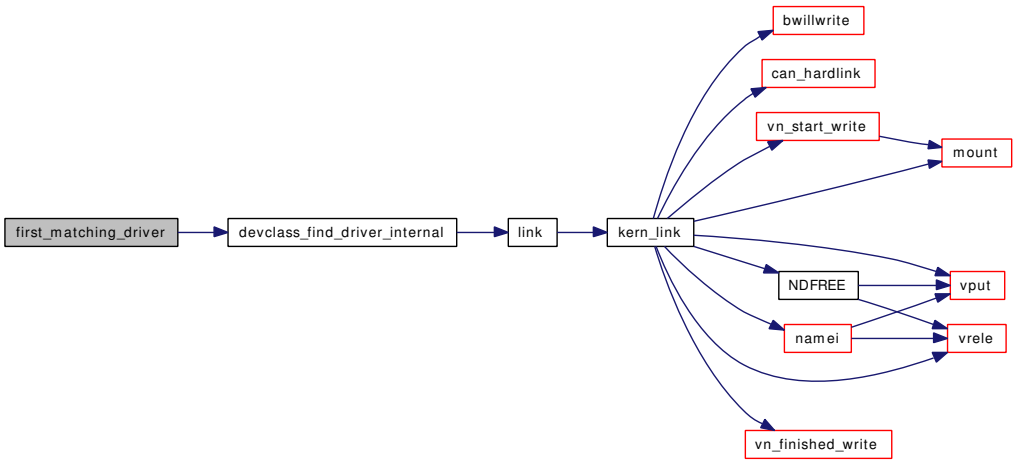
For internal use only.

Definition at line 1665 of file subr_bus.c.

References devclass_find_driver_internal().

Referenced by device_probe_child().

Here is the call graph for this function:



9.87.4.141 static device_t make_device (device_t *parent*, const char * *name*, int *unit*) [static]

Make a new device and add it as a child of `parent`.

For internal use only.

Parameters:

parent the parent of the new device

name the devclass name of the new device or `NULL` to leave the devclass unspecified unit the unit number of the new device of `-1` to leave the unit number unspecified

Returns:

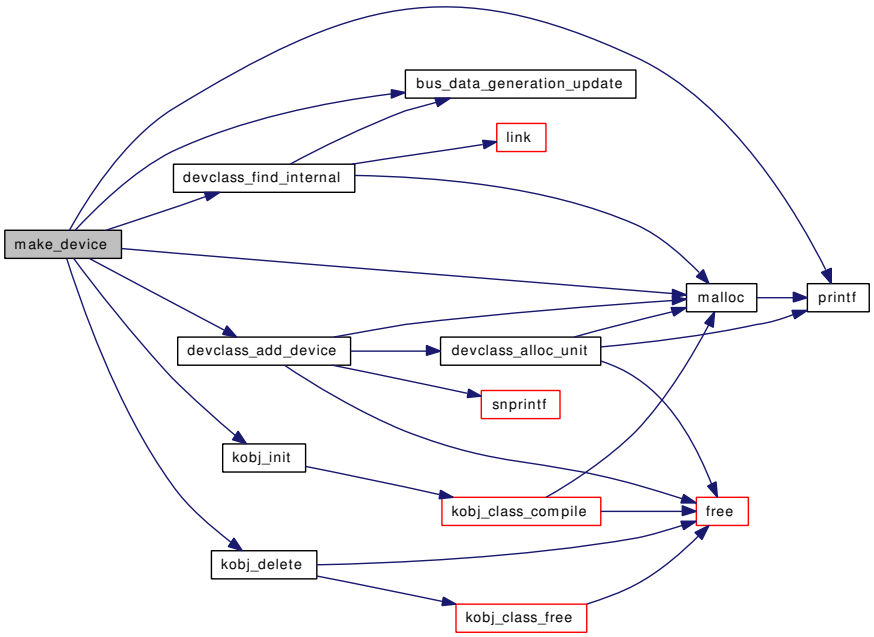
the new device

Definition at line 1433 of file subr_bus.c.

References bus_data_generation_update(), devclass_add_device(), devclass_find_internal(), DEVICENAME, DF_ENABLED, DF_FIXEDCLASS, DF_WILDCARD, kobj_delete(), kobj_init(), malloc(), PDEBUG, and printf().

Referenced by device_add_child_ordered(), and root_bus_module_handler().

Here is the call graph for this function:



9.87.4.142 static MALLOC_DEFINE (M_BUS_SC, "bus-sc", "Bus data *structures*, softc") [static]

9.87.4.143 static MALLOC_DEFINE (M_BUS, "bus", "Bus data structures") [static]

9.87.4.144 static driverlink_t next_matching_driver (devclass_t *dc*, device_t *dev*, driverlink_t *last*) [static]

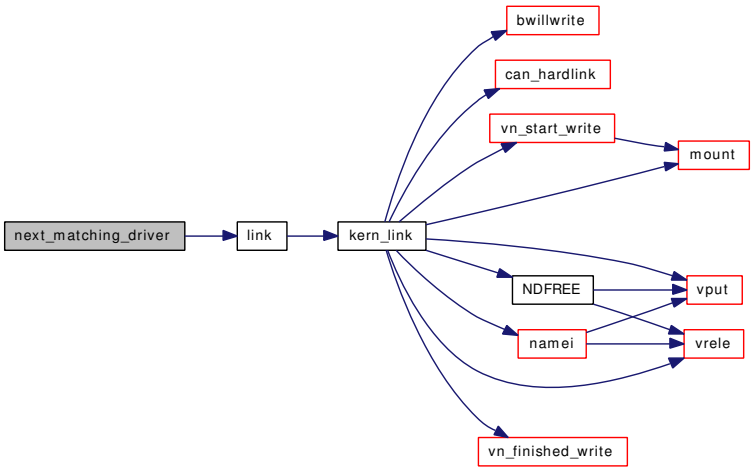
For internal use only.

Definition at line 1676 of file subr_bus.c.

References driverlink::driver, and link().

Referenced by device_probe_child().

Here is the call graph for this function:



9.87.4.145 struct resource_list_entry* resource_list_add (struct resource_list * *rl*, int *type*, int *rid*, u_long *start*, u_long *end*, u_long *count*)

Add or modify a resource entry.

If an existing entry exists with the same type and rid, it will be modified using the given values of `start`, `end` and `count`. If no entry exists, a new one will be created using the given values. The resource list entry that matches is then returned.

Parameters:

rl the resource list to edit

type the resource entry type (e.g. SYS_RES_MEMORY)

rid the resource identifier

start the start address of the resource

end the end address of the resource

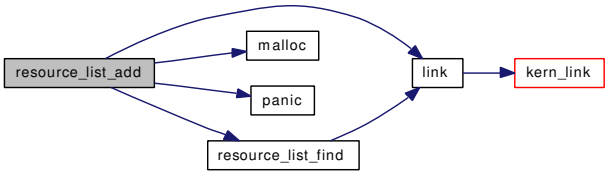
count XXX end-start+1

Definition at line 2590 of file subr_bus.c.

References link(), malloc(), panic(), and resource_list_find().

Referenced by bus_generic_rl_set_resource(), and resource_list_add_next().

Here is the call graph for this function:



9.87.4.146 int resource_list_add_next (struct resource_list * *rl*, int *type*, u_long *start*, u_long *end*, u_long *count*)

Add a resource entry.

This function adds a resource entry using the given `type`, `start`, `end` and `count` values. A rid value is chosen by searching sequentially for the first unused rid starting at zero.

Parameters:

rl the resource list to edit

type the resource entry type (e.g. SYS_RES_MEMORY)

start the start address of the resource

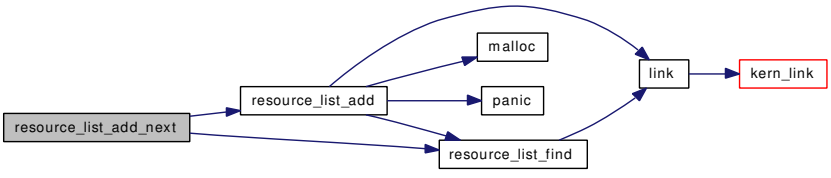
end the end address of the resource

count XXX end-start+1

Definition at line 2562 of file subr_bus.c.

References resource_list_add(), and resource_list_find().

Here is the call graph for this function:



9.87.4.147 struct resource* resource_list_alloc (struct resource_list * *rl*, device_t *bus*, device_t *child*, int *type*, int * *rid*, u_long *start*, u_long *end*, u_long *count*, u_int *flags*)

Helper function for implementing BUS_ALLOC_RESOURCE().

Implement BUS_ALLOC_RESOURCE() by looking up a resource from the list and passing the allocation up to the parent of `bus`. This assumes that the first entry of `device_get_ivars(child)` is a struct

resource_list. This also handles 'passthrough' allocations where a child is a remote descendant of bus by passing the allocation up to the parent of bus.

Typically, a bus driver would store a list of child resources somewhere in the child device's ivars (see device_get_ivars()) and its implementation of BUS_ALLOC_RESOURCE() would find that list and then call resource_list_alloc() to perform the allocation.

Parameters:

rl the resource list to allocate from

bus the parent device of `child`

child the device which is requesting an allocation

type the type of resource to allocate

rid a pointer to the resource identifier

start hint at the start of the resource range - pass `0UL` for any start address

end hint at the end of the resource range - pass ~`0UL` for any end address

count hint at the size of range required - pass `1` for any size

flags any extra flags to control the resource allocation - see `RF_XXX` flags in <sys/rman.h> for details

Returns:

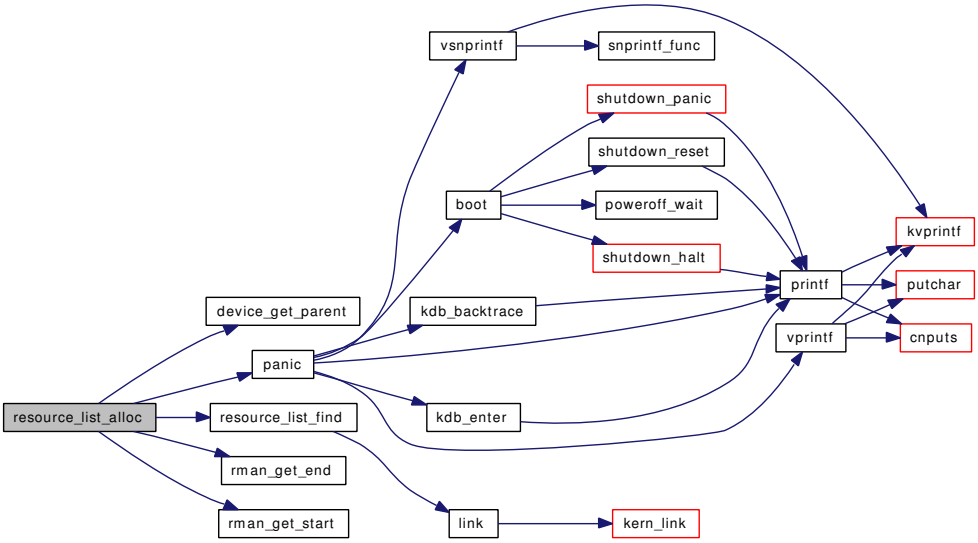
the resource which was allocated or `NULL` if no resource could be allocated

Definition at line 2692 of file subr_bus.c.

References device_get_parent(), panic(), resource_list_find(), rman_get_end(), and rman_get_start().

Referenced by bus_generic_rl_alloc_resource().

Here is the call graph for this function:



9.87.4.148 void resource_list_delete (struct resource_list * *rl*, int *type*, int *rid*)

Delete a resource entry.

Parameters:

rl the resource list to edit

type the resource entry type (e.g. SYS_RES_MEMORY)

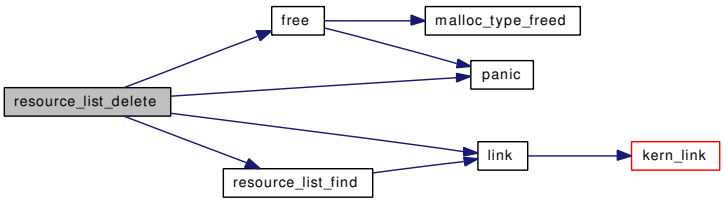
rid the resource identifier

Definition at line 2646 of file subr_bus.c.

References free(), link(), panic(), and resource_list_find().

Referenced by bus_generic_rl_delete_resource().

Here is the call graph for this function:



9.87.4.149 struct resource_list_entry* resource_list_find (struct resource_list * *rl*, int *type*, int *rid*)

Find a resource entry by type and rid.

Parameters:

rl the resource list to search

type the resource entry type (e.g. SYS_RES_MEMORY)

rid the resource identifier

Returns:

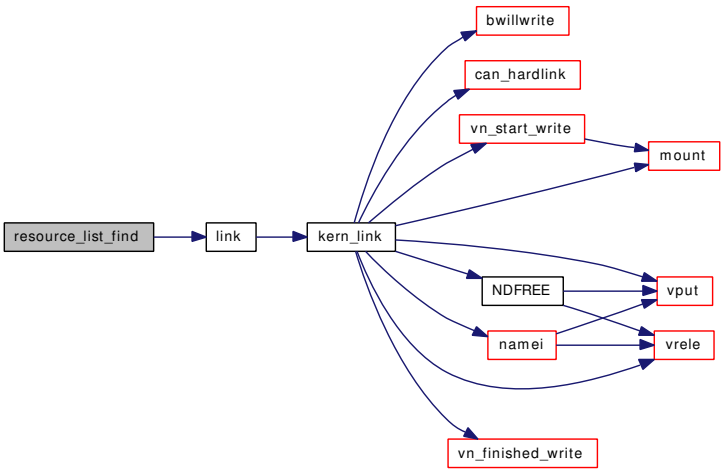
the resource entry pointer or NULL if there is no such entry.

Definition at line 2627 of file subr_bus.c.

References link().

Referenced by bus_generic_rl_get_resource(), resource_list_add(), resource_list_add_next(), resource_list_alloc(), resource_list_delete(), and resource_list_release().

Here is the call graph for this function:



9.87.4.150 void resource_list_free (struct resource_list * *rl*)

Reclaim memory used by a resource list.

This function frees the memory for all resource entries on the list (if any).

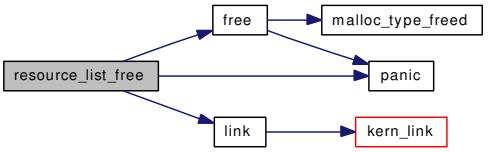
Parameters:

rl the resource list to free

Definition at line 2536 of file subr_bus.c.

References free(), link(), and panic().

Here is the call graph for this function:



9.87.4.151 void resource_list_init (struct resource_list * *rl*)

Initialise a resource list.

Parameters:

rl the resource list to initialise

Definition at line 2522 of file subr_bus.c.

9.87.4.152 int resource_list_print_type (struct resource_list * *rl*, const char * *name*, int *type*, const char * *format*)

Print a description of resources in a resource list.

Print all resources of a specified type, for use in BUS_PRINT_CHILD(). The name is printed if at least one resource of the given type is available. The format is used to print resource start and end.

Parameters:

rl the resource list to print

name the name of `type`, e.g. `"memory"`

type type type of resource entry to print

format printf(9) format string to print resource start and end values

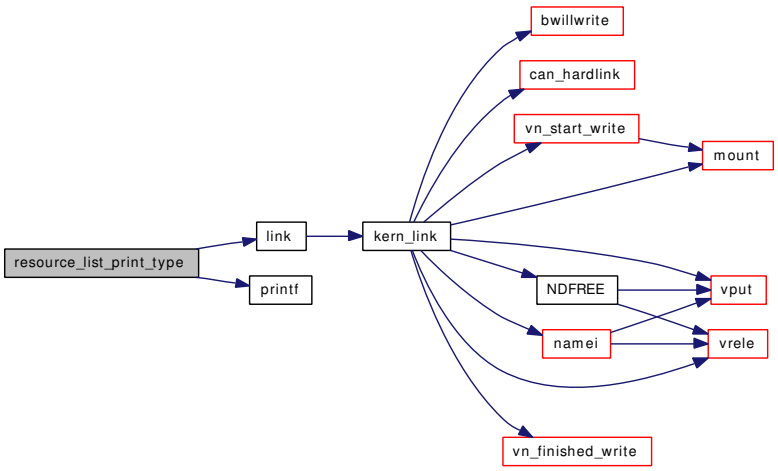
Returns:

the number of characters printed

Definition at line 2795 of file subr_bus.c.

References link(), and printf().

Here is the call graph for this function:



9.87.4.153 void resource_list_purge (struct resource_list * *rl*)

Releases all the resources in a list.

Parameters:

rl The resource list to purge.

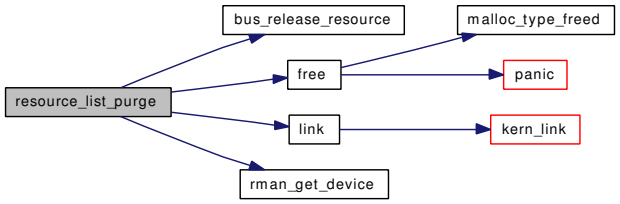
Returns:

nothing

Definition at line 2830 of file subr_bus.c.

References bus_release_resource(), free(), link(), and rman_get_device().

Here is the call graph for this function:



9.87.4.154 int resource_list_release (struct resource_list * *rl*, device_t *bus*, device_t *child*, int *type*, int *rid*, struct resource * *res*)

Helper function for implementing BUS_RELEASE_RESOURCE().

Implement BUS_RELEASE_RESOURCE() using a resource list. Normally used with resource_list_alloc().

Parameters:

rl the resource list which was allocated from

bus the parent device of `child`

child the device which is requesting a release

type the type of resource to allocate

rid the resource identifier

res the resource to release

Return values:

0 success

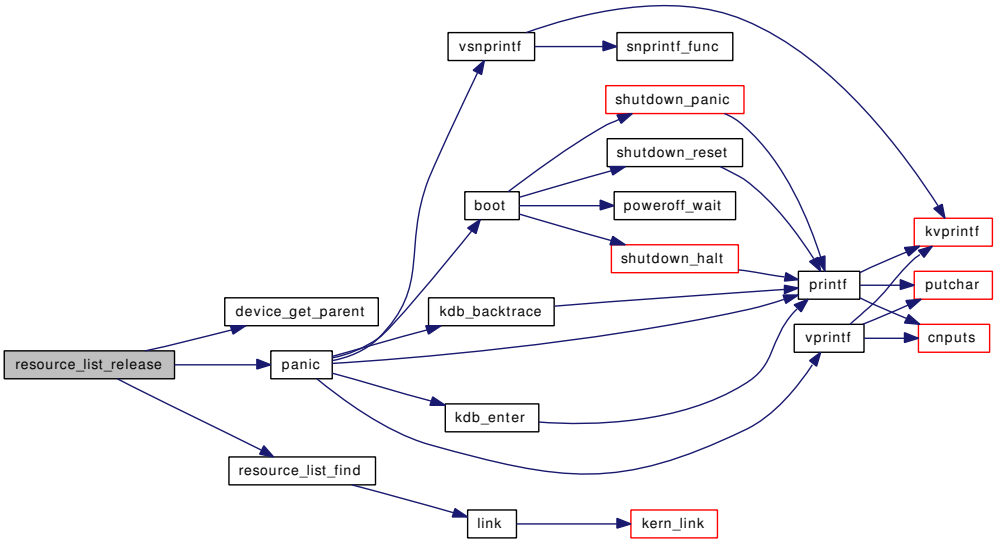
non-zero a standard unix error code indicating what error condition prevented the operation

Definition at line 2751 of file subr_bus.c.

References device_get_parent(), panic(), and resource_list_find().

Referenced by bus_generic_rl_release_resource().

Here is the call graph for this function:



9.87.4.155 void root_bus_configure (void)

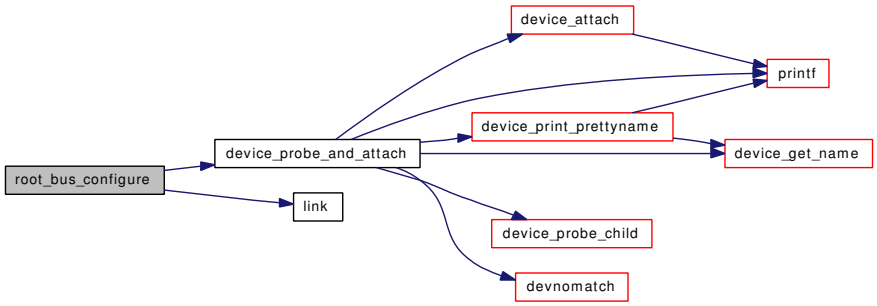
Automatically configure devices.

This function begins the autoconfiguration process by calling device_probe_and_attach() for each child of the `root0` device.

Definition at line 3760 of file subr_bus.c.

References device_probe_and_attach(), link(), and PDEBUG.

Here is the call graph for this function:

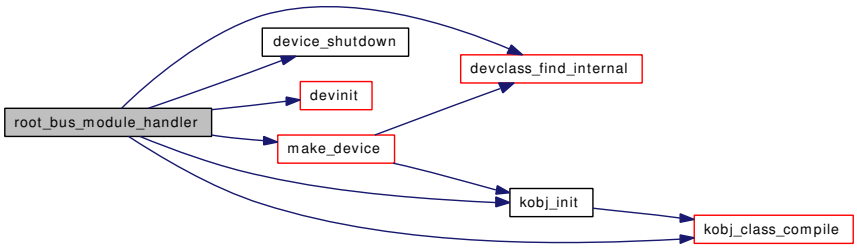


9.87.4.156 static int root_bus_module_handler (module_t *mod*, int *what*, void * *arg*) [static]

Definition at line 3721 of file subr_bus.c.

References devclass_find_internal(), device_shutdown(), devinit(), kobj_class_compile(), kobj_init(), and make_device().

Here is the call graph for this function:



9.87.4.157 static int root_child_present (device_t *dev*, device_t *child*) [static]

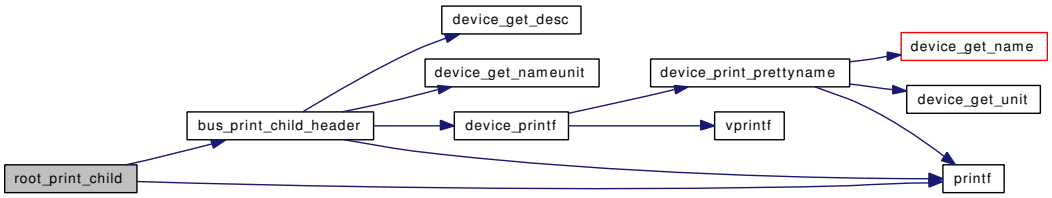
Definition at line 3690 of file subr_bus.c.

9.87.4.158 static int root_print_child (device_t *dev*, device_t *child*) [static]

Definition at line 3662 of file subr_bus.c.

References bus_print_child_header(), and printf().

Here is the call graph for this function:

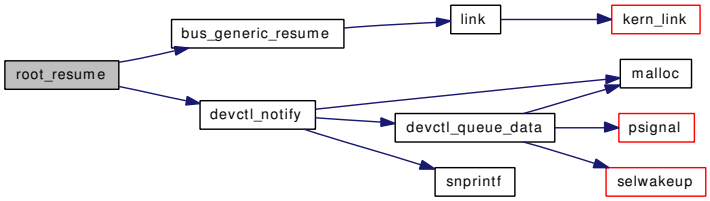


9.87.4.159 static int root_resume (device_t *dev*) [static]

Definition at line 3651 of file subr_bus.c.

References bus_generic_resume(), and devctl_notify().

Here is the call graph for this function:

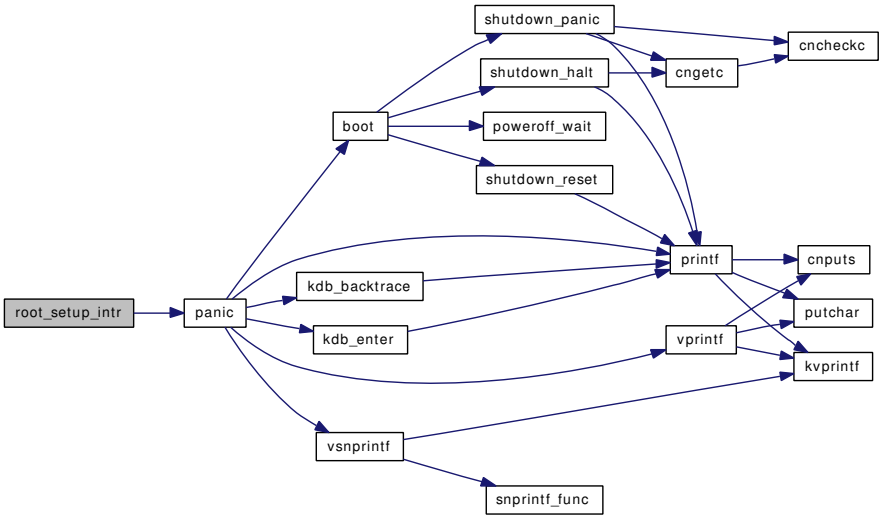


9.87.4.160 static int root_setup_intr (device_t *dev*, device_t *child*, driver_intr_t * *intr*, void * *arg*, void ** *cookiep*) [static]

Definition at line 3673 of file subr_bus.c.

References panic().

Here is the call graph for this function:



9.87.4.161 static int sysctl_bus (SYSCTL_HANDLER_ARGS) [static]

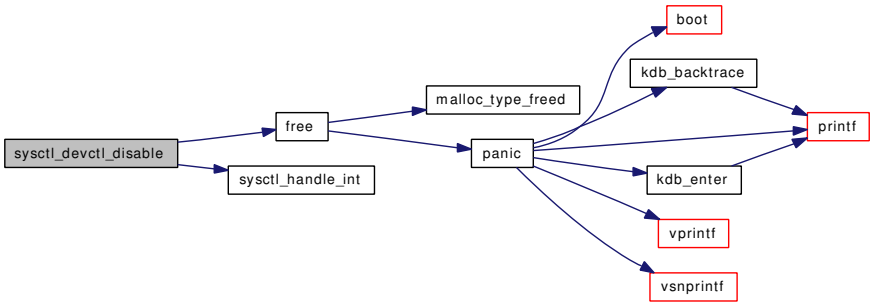
Definition at line 4054 of file subr_bus.c.

9.87.4.162 static int sysctl_devctl_disable (SYSCTL_HANDLER_ARGS) [static]

Definition at line 700 of file subr_bus.c.

References dev_event_info::dei_data, devctl_disable, dev_softc::devq, devsoftc, free(), dev_softc::mtx, and sysctl_handle_int().

Here is the call graph for this function:

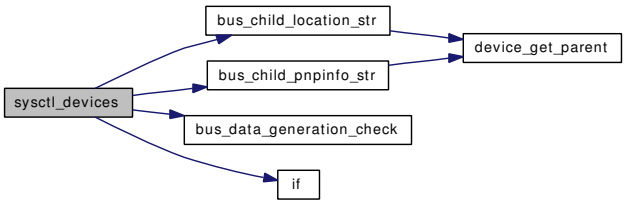


9.87.4.163 static int sysctl_devices (SYSCTL_HANDLER_ARGS) [static]

Definition at line 4067 of file subr_bus.c.

References bus_child_location_str(), bus_child_pnpinfo_str(), bus_data_generation_check(), and if().

Here is the call graph for this function:



9.87.4.164 SYSCTL_NODE (_hw_bus, OID_AUTO, devices, CTLFLAG_RD, sysctl_devices, "system device tree")

9.87.4.165 SYSCTL_NODE (_hw_bus, OID_AUTO, info, CTLFLAG_RW, sysctl_bus, "bus-related data")

9.87.4.166 SYSCTL_NODE (OID_AUTO, dev, CTLFLAG_RW, NULL, NULL)

9.87.4.167 SYSCTL_NODE (_hw, OID_AUTO, bus, CTLFLAG_RW, NULL, NULL)

9.87.4.168 SYSCTL_PROC (_hw_bus, OID_AUTO, devctl_disable, CTLTYPE_INT| *CTLFLAG_RW*, 0, 0, sysctl_devctl_disable, "I", "devctl disable")

9.87.4.169 TAILQ_HEAD (device)

Definition at line 725 of file subr_bus.c.

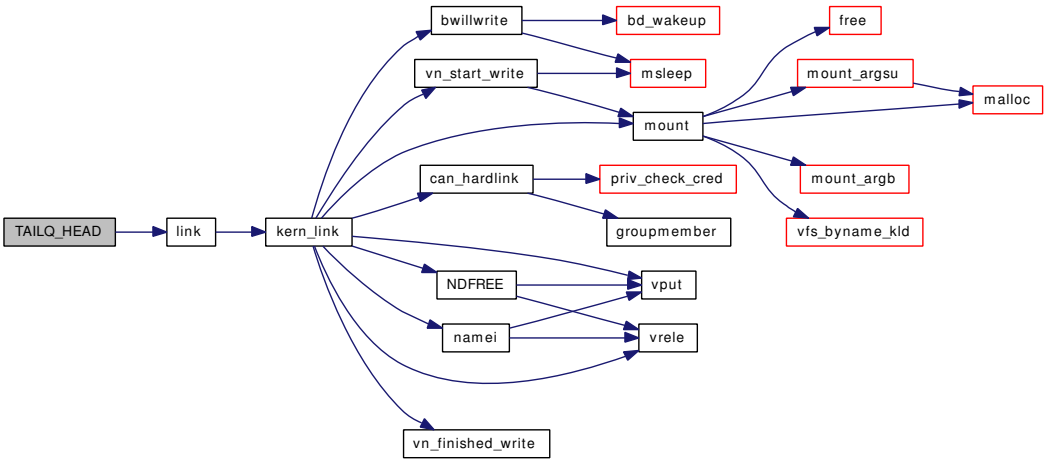
9.87.4.170 TAILQ_HEAD (devq, dev_event_info)

9.87.4.171 typedef TAILQ_HEAD (devclass_list, devclass)

Definition at line 74 of file subr_bus.c.

References link().

Here is the call graph for this function:



9.87.4.172 TUNABLE_INT ("hw.bus.devctl_disable", & *devctl_disable*)

9.87.5 Variable Documentation

9.87.5.1 struct cdevsw dev_cdevsw [static]

Initial value:

```
{
        .d_version =    D_VERSION,
        .d_flags =      D_NEEDGIANT,
        .d_open =       devopen,
        .d_close =      devclose,
        .d_read =       devread,
        .d_ioctl =      devioctl,
        .d_poll =       devpoll,
        .d_name =       "devctl",
}
```

Definition at line 351 of file subr_bus.c.

9.87.5.2 devclass_list_t devclasses = TAILQ_HEAD_INITIALIZER(devclasses) [static]

Definition at line 738 of file subr_bus.c.

Referenced by devclass_find_internal().

9.87.5.3 d_close_t devclose [static]

Definition at line 346 of file subr_bus.c.

9.87.5.4 struct cdev* devctl_dev [static]

Definition at line 381 of file subr_bus.c.

Referenced by devinit().

9.87.5.5 int devctl_disable = 0 [static]

Definition at line 340 of file subr_bus.c.

Referenced by devaddq(), and sysctl_devctl_disable().

9.87.5.6 d_ioctl_t devioctl [static]

Definition at line 348 of file subr_bus.c.

9.87.5.7 d_open_t devopen [static]

Definition at line 345 of file subr_bus.c.

9.87.5.8 d_poll_t devpoll [static]

Definition at line 349 of file subr_bus.c.

9.87.5.9 d_read_t devread [static]

Definition at line 347 of file subr_bus.c.

9.87.5.10 struct dev_softc devsoftc [static]

Referenced by devclose(), devctl_queue_data(), devinit(), devioctl(), devopen(), devpoll(), devread(), and sysctl_devctl_disable().

9.87.5.11 device_t root_bus

Definition at line 3717 of file subr_bus.c.

9.87.5.12 moduledata_t root_bus_mod [static]

Initial value:

```
{
        "rootbus",
        root_bus_module_handler,
        0
}
```

Definition at line 3746 of file subr_bus.c.

9.87.5.13 devclass_t root_devclass

Definition at line 3718 of file subr_bus.c.

9.87.5.14 driver_t root_driver [static]

Initial value:

```
{
        "root",
        root_methods,
        1,
}
```

Definition at line 3711 of file subr_bus.c.

9.87.5.15 kobj_method_t root_methods[ ] [static]

Initial value:

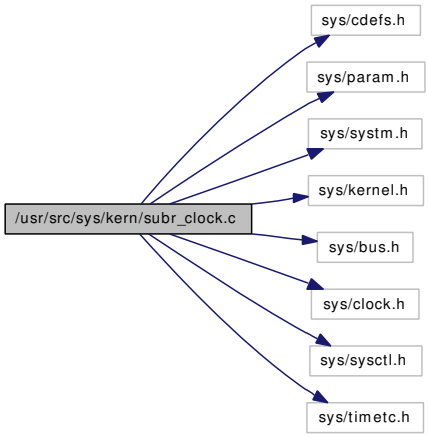
```
{

        KOBJMETHOD(device_shutdown,     bus_generic_shutdown),
        KOBJMETHOD(device_suspend,      bus_generic_suspend),
        KOBJMETHOD(device_resume,       root_resume),


        KOBJMETHOD(bus_print_child,     root_print_child),
        KOBJMETHOD(bus_read_ivar,       bus_generic_read_ivar),
        KOBJMETHOD(bus_write_ivar,      bus_generic_write_ivar),
        KOBJMETHOD(bus_setup_intr,      root_setup_intr),
        KOBJMETHOD(bus_child_present,   root_child_present),

        { 0, 0 }
}
```

Definition at line 3695 of file subr_bus.c.

9.88 /usr/src/sys/kern/subr_clock.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/bus.h>
#include <sys/clock.h>
#include <sys/sysctl.h>
#include <sys/timetc.h>
```

Include dependency graph for subr_clock.c:



Defines

- #define FEBRUARY 2
- #define days_in_year(y) (leapyear(y) ? 366 : 365)
- #define days_in_month(y, m) (month_days[(m) - 1] + (m == FEBRUARY ? leapyear(y) : 0))
- #define day_of_week(days) (((days) + 4) % 7)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_clock.c,v 1.11 2006/10/02 18:23:37 phk Exp $")
- SYSCTL_INT (_machdep, OID_AUTO, disable_rtc_set, CTLFLAG_RW,&disable_rtc_set, 0,"")
- SYSCTL_INT (_machdep, OID_AUTO, wall_cmos_clock, CTLFLAG_RW,&wall_cmos_clock, 0,"")
- static int sysctl_machdep_adjkerntz (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_machdep, OID_AUTO, adjkerntz, CTLTYPE_INT|CTLFLAG_RW,&adjkerntz, 0, sysctl_machdep_adjkerntz,"I","")
- static __inline int leapyear (int year)
- int clock_ct_to_ts (struct clocktime *ct, struct timespec *ts)
- void clock_ts_to_ct (struct timespec *ts, struct clocktime *ct)
- int utc_offset (void)

Variables

- static int adjkerntz
- static int wall_cmos_clock
- int disable_rtc_set
- int tz_minuteswest
- int tz_dsttime
- static const int month_days [12]

9.88.1 Define Documentation

9.88.1.1 #define day_of_week(days) (((days) + 4) % 7)

Definition at line 95 of file subr_clock.c.

Referenced by clock_ct_to_ts(), and clock_ts_to_ct().

9.88.1.2 #define days_in_month(y, m) (month_days[(m) - 1] + (m == FEBRUARY ? leapyear(y) : 0))

Definition at line 92 of file subr_clock.c.

Referenced by clock_ct_to_ts(), and clock_ts_to_ct().

9.88.1.3 #define days_in_year(y) (leapyear(y) ? 366 : 365)

Definition at line 91 of file subr_clock.c.

Referenced by clock_ct_to_ts(), and clock_ts_to_ct().

9.88.1.4 #define FEBRUARY 2

Definition at line 90 of file subr_clock.c.

9.88.2 Function Documentation

9.88.2.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_clock. *c*, v 1.11 2006/10/02 18:23:37 phk Exp $")

9.88.2.2 int clock_ct_to_ts (struct clocktime * *ct*, struct timespec * *ts*)

Definition at line 127 of file subr_clock.c.

References day_of_week, days, days_in_month, and days_in_year.

9.88.2.3 void clock_ts_to_ct (struct timespec * *ts*, struct clocktime * *ct*)

Definition at line 167 of file subr_clock.c.

References day_of_week, days, days_in_month, and days_in_year.

9.88.2.4 static __inline int leapyear (int *year*) [static]

Definition at line 111 of file subr_clock.c.

9.88.2.5 SYSCTL_INT (_machdep, OID_AUTO, wall_cmos_clock, CTLFLAG_RW, & *wall_cmos_clock*, 0, "")

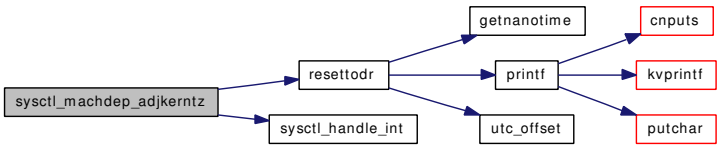
9.88.2.6 SYSCTL_INT (_machdep, OID_AUTO, disable_rtc_set, CTLFLAG_RW, & *disable_rtc_set*, 0, "")

9.88.2.7 static int sysctl_machdep_adjkerntz (SYSCTL_HANDLER_ARGS) [static]

Definition at line 70 of file subr_clock.c.

References resettodr(), and sysctl_handle_int().

Here is the call graph for this function:



9.88.2.8 SYSCTL_PROC (_machdep, OID_AUTO, adjkerntz, CTLTYPE_INT| *CTLFLAG_RW*, & *adjkerntz*, 0, sysctl_machdep_adjkerntz, "I", "")

9.88.2.9 int utc_offset (void)

Definition at line 202 of file subr_clock.c.

References adjkerntz, tz_minuteswest, and wall_cmos_clock.

Referenced by fattime2timespec(), inittodr(), resettodr(), and timespec2fattime().

9.88.3 Variable Documentation

9.88.3.1 int adjkerntz [static]

Definition at line 52 of file subr_clock.c.

Referenced by utc_offset().

9.88.3.2 int disable_rtc_set

Definition at line 54 of file subr_clock.c.

Referenced by resettodr().

9.88.3.3 const int month_days[12] [static]

Initial value:

```
 {
        31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31
}
```

Definition at line 97 of file subr_clock.c.

9.88.3.4 int tz_dsttime

Definition at line 57 of file subr_clock.c.

Referenced by gettimeofday(), and kern_settimeofday().

9.88.3.5 int tz_minuteswest

Definition at line 56 of file subr_clock.c.

Referenced by gettimeofday(), kern_settimeofday(), and utc_offset().

9.88.3.6 int wall_cmos_clock [static]

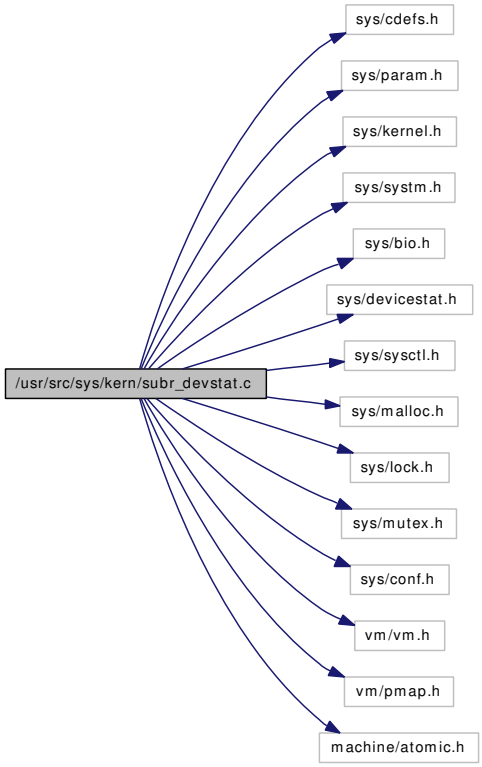
Definition at line 53 of file subr_clock.c.

Referenced by utc_offset().

9.89 /usr/src/sys/kern/subr_devstat.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/kernel.h>
#include <sys/systm.h>
#include <sys/bio.h>
#include <sys/devicestat.h>
#include <sys/sysctl.h>
#include <sys/malloc.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/conf.h>
#include <vm/vm.h>
#include <vm/pmap.h>
#include <machine/atomic.h>
```

Include dependency graph for subr_devstat.c:



Data Structures

- struct statspage

Defines

- #define statsperpage (PAGE_SIZE / sizeof(struct devstat))

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_devstat.c,v 1.51 2005/05/03 10:58:05 jeff Exp $")
- static struct devstat * devstat_alloc (void)
- static void devstat_free (struct devstat *)
- static void devstat_add_entry (struct devstat *ds, const void *dev_name, int unit_number, u_int32_t block_size, devstat_support_flags flags, devstat_type_flags device_type, devstat_priority priority)
- devstat * devstat_new_entry (const void *dev_name, int unit_number, u_int32_t block_size, devstat_support_flags flags, devstat_type_flags device_type, devstat_priority priority)
- void devstat_remove_entry (struct devstat *ds)
- void devstat_start_transaction (struct devstat *ds, struct bintime *now)
- void devstat_start_transaction_bio (struct devstat *ds, struct bio *bp)
- void devstat_end_transaction (struct devstat *ds, u_int32_t bytes, devstat_tag_type tag_type, devstat_trans_flags flags, struct bintime *now, struct bintime *then)
- void devstat_end_transaction_bio (struct devstat *ds, struct bio *bp)
- static int sysctl_devstat (SYSCTL_HANDLER_ARGS)
- SYSCTL_NODE (_kern, OID_AUTO, devstat, CTLFLAG_RD, 0,"Device Statistics")
- SYSCTL_PROC (_kern_devstat, OID_AUTO, all, CTLFLAG_RD|CTLTYPE_OPAQUE, 0, 0, sysctl_devstat,"S,devstat","All devices in the devstat list")
- SYSCTL_INT (_kern_devstat, OID_AUTO, numdevs, CTLFLAG_RD,&devstat_num_devs, 0,"Number of devices in the devstat list")
- SYSCTL_LONG (_kern_devstat, OID_AUTO, generation, CTLFLAG_RD,&devstat_generation, 0,"Devstat list generation")
- SYSCTL_INT (_kern_devstat, OID_AUTO, version, CTLFLAG_RD,&devstat_version, 0,"Devstat list version number")
- static TAILQ_HEAD (statspage)
- SYSCTL_INT (_debug_sizeof, OID_AUTO, devstat, CTLFLAG_RD, 0, sizeof(struct devstat),"sizeof(struct devstat)")

Variables

- static int devstat_num_devs
- static long devstat_generation = 1
- static int devstat_version = DEVSTAT_VERSION
- static int devstat_current_devnumber
- static struct mtx devstat_mutex
- static struct devstatlist device_statq
- static d_mmap_t devstat_mmap
- static struct cdevsw devstat_cdevsw

9.89.1 Define Documentation

9.89.1.1 #define statsperpage (PAGE_SIZE / sizeof(struct devstat))

Definition at line 431 of file subr_devstat.c.

Referenced by devstat_alloc(), and devstat_free().

9.89.2 Function Documentation

9.89.2.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_devstat. *c*, v 1.51 2005/05/03 10:58:05 jeff Exp $")

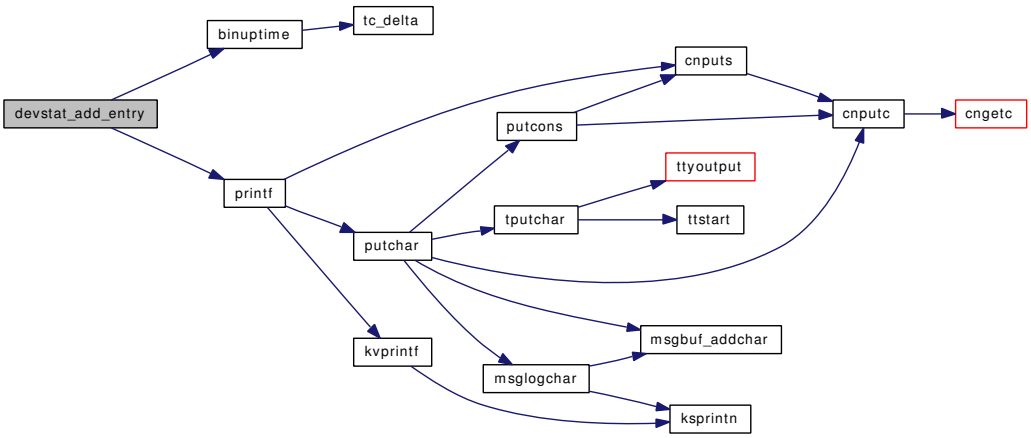
9.89.2.2 static void devstat_add_entry (struct devstat * *ds*, const void * *dev_name*, int *unit_number*, u_int32_t *block_size*, devstat_support_flags *flags*, devstat_type_flags *device_type*, devstat_priority *priority*) [static]

Definition at line 101 of file subr_devstat.c.

References binuptime(), device_statq, devstat_current_devnumber, devstat_generation, devstat_mutex, devstat_num_devs, and printf().

Referenced by devstat_new_entry().

Here is the call graph for this function:



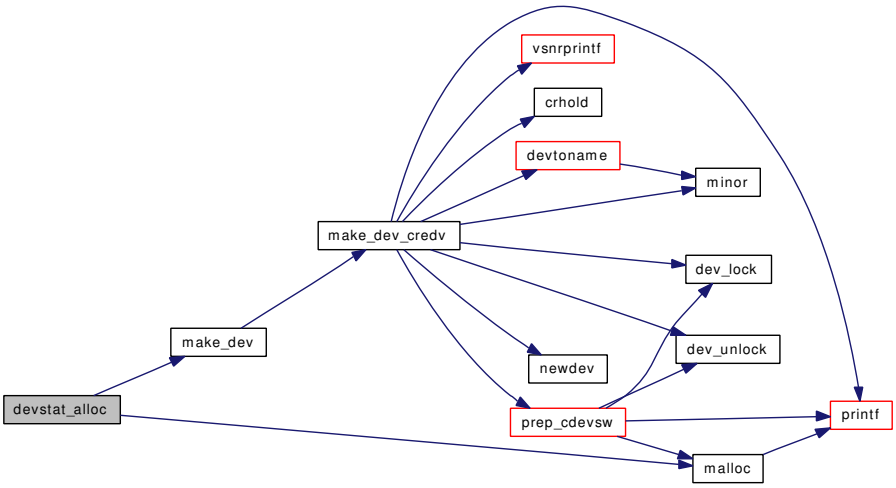
9.89.2.3 static struct devstat * devstat_alloc (void) [static]

Definition at line 469 of file subr_devstat.c.

References devstat_cdevsw, devstat_mutex, make_dev(), malloc(), and statsperpage.

Referenced by devstat_new_entry().

Here is the call graph for this function:



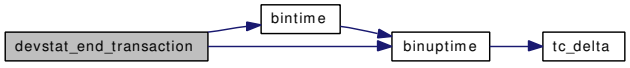
9.89.2.4 void devstat_end_transaction (struct devstat * *ds*, u_int32_t *bytes*, devstat_tag_type *tag_type*, devstat_trans_flags *flags*, struct bintime * *now*, struct bintime * *then*)

Definition at line 278 of file subr_devstat.c.

References bintime(), and binuptime().

Referenced by devstat_end_transaction_bio().

Here is the call graph for this function:



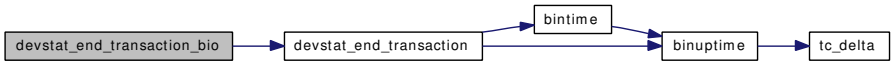
9.89.2.5 void devstat_end_transaction_bio (struct devstat * *ds*, struct bio * *bp*)

Definition at line 323 of file subr_devstat.c.

References devstat_end_transaction().

Referenced by biofinish().

Here is the call graph for this function:



9.89.2.6 static void devstat_free (struct devstat *) [static]

Definition at line 522 of file subr_devstat.c.

References devstat_mutex, and statsperpage.

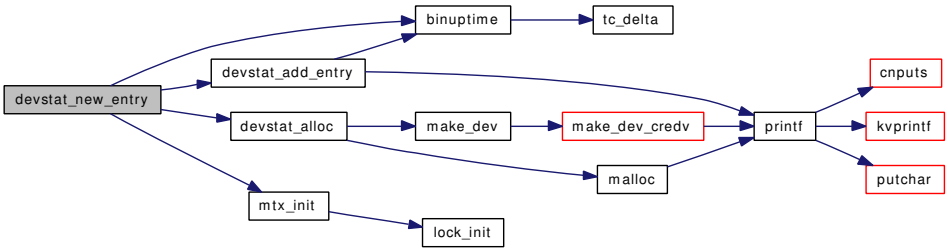
Referenced by devstat_remove_entry().

9.89.2.7 struct devstat* devstat_new_entry (const void * *dev_name*, int *unit_number*, u_int32_t *block_size*, devstat_support_flags *flags*, devstat_type_flags *device_type*, devstat_priority *priority*)

Definition at line 66 of file subr_devstat.c.

References binuptime(), device_statq, devstat_add_entry(), devstat_alloc(), devstat_generation, devstat_-mutex, and mtx_init().

Here is the call graph for this function:



9.89.2.8 void devstat_remove_entry (struct devstat * *ds*)

Definition at line 182 of file subr_devstat.c.

References device_statq, devstat_free(), devstat_generation, devstat_mutex, and devstat_num_devs.

Here is the call graph for this function:

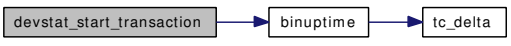
9.89.2.9 void devstat_start_transaction (struct devstat * *ds*, struct bintime * *now*)

Definition at line 212 of file subr_devstat.c.

References binuptime(), and devstat_mutex.

Referenced by devstat_start_transaction_bio().

Here is the call graph for this function:

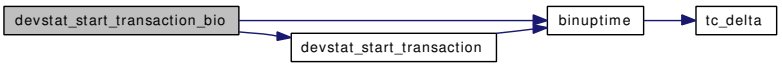


9.89.2.10 void devstat_start_transaction_bio (struct devstat * *ds*, struct bio * *bp*)

Definition at line 238 of file subr_devstat.c.

References binuptime(), devstat_mutex, and devstat_start_transaction().

Here is the call graph for this function:



9.89.2.11 static int sysctl_devstat (SYSCTL_HANDLER_ARGS) [static]

Definition at line 355 of file subr_devstat.c.

References device_statq, devstat_generation, devstat_mutex, and devstat_num_devs.

9.89.2.12 SYSCTL_INT (_debug_sizeof, OID_AUTO, devstat, CTLFLAG_RD, 0, sizeof(struct devstat), "sizeof(struct devstat)")

9.89.2.13 SYSCTL_INT (_kern_devstat, OID_AUTO, version, CTLFLAG_RD, & *devstat_version*, 0, "Devstat list version number")

9.89.2.14 SYSCTL_INT (_kern_devstat, OID_AUTO, numdevs, CTLFLAG_RD, & *devstat_num_devs*, 0, "Number of devices in the devstat list")

9.89.2.15 SYSCTL_LONG (_kern_devstat, OID_AUTO, generation, CTLFLAG_RD, & *devstat_generation*, 0, "Devstat list generation")

9.89.2.16 SYSCTL_NODE (_kern, OID_AUTO, devstat, CTLFLAG_RD, 0, "Device Statistics")

9.89.2.17 SYSCTL_PROC (_kern_devstat, OID_AUTO, all, CTLFLAG_RD| *CTLTYPE_OPAQUE*, 0, 0, sysctl_devstat, " *S*, devstat", "All devices in the devstat list")

9.89.2.18 static TAILQ_HEAD (statspage) [static]

Definition at line 448 of file subr_devstat.c.

9.89.3 Variable Documentation

9.89.3.1 struct devstatlist device_statq [static]

Definition at line 53 of file subr_devstat.c.

Referenced by devstat_add_entry(), devstat_new_entry(), devstat_remove_entry(), and sysctl_devstat().

9.89.3.2 struct cdevsw devstat_cdevsw [static]

Initial value:

```
{
        .d_version =    D_VERSION,
        .d_flags =      D_NEEDGIANT,
```

```
        .d_mmap =       devstat_mmap,
        .d_name =       "devstat",
}
```

Definition at line 435 of file subr_devstat.c.

Referenced by devstat_alloc().

9.89.3.3 int devstat_current_devnumber [static]

Definition at line 50 of file subr_devstat.c.

Referenced by devstat_add_entry().

9.89.3.4 long devstat_generation = 1 [static]

Definition at line 48 of file subr_devstat.c.

Referenced by devstat_add_entry(), devstat_new_entry(), devstat_remove_entry(), and sysctl_devstat().

9.89.3.5 d_mmap_t devstat_mmap [static]

Definition at line 433 of file subr_devstat.c.

9.89.3.6 struct mtx devstat_mutex [static]

Definition at line 51 of file subr_devstat.c.

Referenced by devstat_add_entry(), devstat_alloc(), devstat_free(), devstat_new_entry(), devstat_remove_-entry(), devstat_start_transaction(), devstat_start_transaction_bio(), and sysctl_devstat().

9.89.3.7 int devstat_num_devs [static]

Definition at line 47 of file subr_devstat.c.

Referenced by devstat_add_entry(), devstat_remove_entry(), and sysctl_devstat().

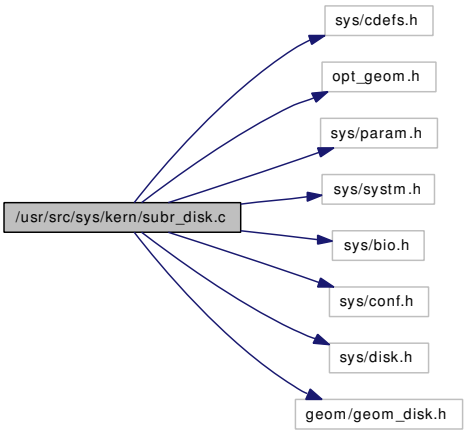
9.89.3.8 int devstat_version = DEVSTAT_VERSION [static]

Definition at line 49 of file subr_devstat.c.

9.90 /usr/src/sys/kern/subr_disk.c File Reference

```
#include <sys/cdefs.h>
#include "opt_geom.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/bio.h>
#include <sys/conf.h>
#include <sys/disk.h>
#include <geom/geom_disk.h>
```

Include dependency graph for subr_disk.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_disk.c,v 1.88 2006/10/31 21:11:21 pjd Exp $")
- void disk_err (struct bio *bp, const char *what, int blkdone, int nl)
- void bioq_init (struct bio_queue_head *head)
- void bioq_remove (struct bio_queue_head *head, struct bio *bp)
- void bioq_flush (struct bio_queue_head *head, struct devstat *stp, int error)
- void bioq_insert_head (struct bio_queue_head *head, struct bio *bp)
- void bioq_insert_tail (struct bio_queue_head *head, struct bio *bp)
- bio * bioq_first (struct bio_queue_head *head)
- bio * bioq_takefirst (struct bio_queue_head *head)
- void bioq_disksort (struct bio_queue_head *bioq, struct bio *bp)

9.90.1 Function Documentation

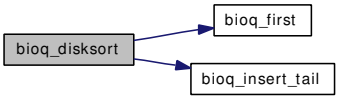
9.90.1.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_disk. *c*, v 1.88 2006/10/31 21:11:21 pjd Exp $")

9.90.1.2 void bioq_disksort (struct bio_queue_head * *bioq*, struct bio * *bp*)

Definition at line 147 of file subr_disk.c.

References bioq_first(), and bioq_insert_tail().

Here is the call graph for this function:



9.90.1.3 struct bio* bioq_first (struct bio_queue_head * *head*)

Definition at line 118 of file subr_disk.c.

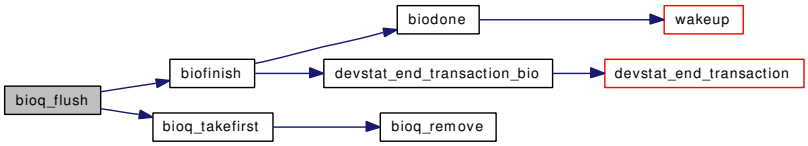
Referenced by bioq_disksort().

9.90.1.4 void bioq_flush (struct bio_queue_head * *head*, struct devstat * *stp*, int *error*)

Definition at line 91 of file subr_disk.c.

References biofinish(), and bioq_takefirst().

Here is the call graph for this function:



9.90.1.5 void bioq_init (struct bio_queue_head * *head*)

Definition at line 69 of file subr_disk.c.

9.90.1.6 void bioq_insert_head (struct bio_queue_head * *head*, struct bio * *bp*)

Definition at line 100 of file subr_disk.c.

9.90.1.7 void bioq_insert_tail (struct bio_queue_head * *head*, struct bio * *bp*)

Definition at line 109 of file subr_disk.c.

Referenced by bioq_disksort().

9.90.1.8 void bioq_remove (struct bio_queue_head * *head*, struct bio * *bp*)

Definition at line 77 of file subr_disk.c.

Referenced by bioq_takefirst().

9.90.1.9 struct bio* bioq_takefirst (struct bio_queue_head * *head*)

Definition at line 125 of file subr_disk.c.

References bioq_remove().

Referenced by bioq_flush().

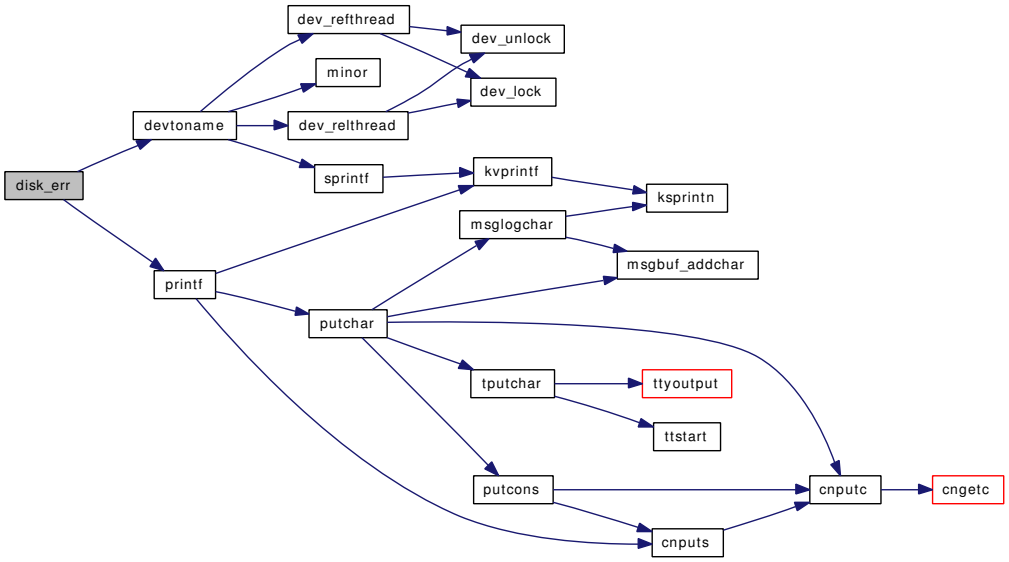
Here is the call graph for this function:

9.90.1.10 void disk_err (struct bio * *bp*, const char * *what*, int *blkdone*, int *nl*)

Definition at line 30 of file subr_disk.c.

References devtoname(), and printf().

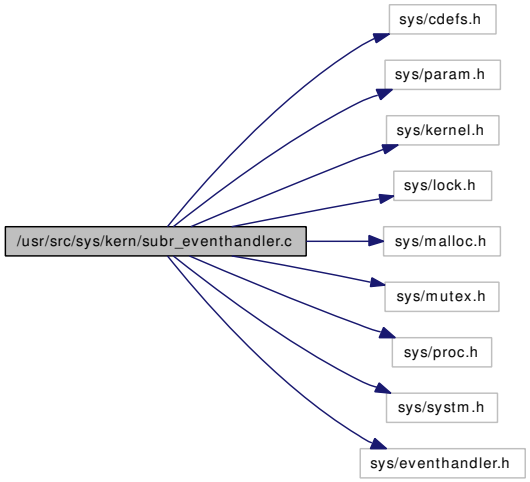
Here is the call graph for this function:



9.91 /usr/src/sys/kern/subr_eventhandler.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/systm.h>
#include <sys/eventhandler.h>
```

Include dependency graph for subr_eventhandler.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_eventhandler.c,v 1.23 2005/02/23 19:32:29 sam Exp $")
- static MALLOC_DEFINE (M_EVENTHANDLER,"eventhandler","Event handler records")
- static TAILQ_HEAD (eventhandler_list)
- static struct eventhandler_list * _eventhandler_find_list (const char *name)
- static void eventhandler_init (void *dummy __unused)
- SYSINIT (eventhandlers, SI_SUB_EVENTHANDLER, SI_ORDER_FIRST, eventhandler_init, NULL)
- void eventhandler_deregister (struct eventhandler_list *list, eventhandler_tag tag)
- eventhandler_list * eventhandler_find_list (const char *name)
- void eventhandler_prune_list (struct eventhandler_list *list)

9.91.1 Function Documentation

9.91.1.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_eventhandler. *c*, v 1.23 2005/02/23 19:32:29 sam Exp $")

9.91.1.2 static struct eventhandler_list * _eventhandler_find_list (const char * *name*) [static]

Definition at line 188 of file subr_eventhandler.c.

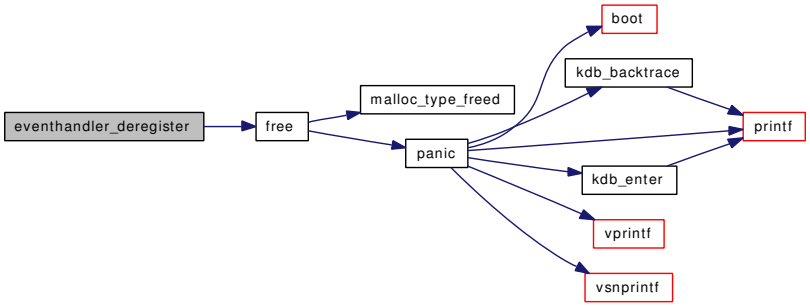
Referenced by eventhandler_find_list(), and SYSINIT().

9.91.1.3 void eventhandler_deregister (struct eventhandler_list * *list*, eventhandler_tag *tag*)

Definition at line 147 of file subr_eventhandler.c.

References free().

Here is the call graph for this function:



9.91.1.4 struct eventhandler_list* eventhandler_find_list (const char * *name*)

Definition at line 204 of file subr_eventhandler.c.

References _eventhandler_find_list().

Here is the call graph for this function:

9.91.1.5 static void eventhandler_init (void *dummy *__unused*) [static]

Definition at line 58 of file subr_eventhandler.c.

References mtx_init().

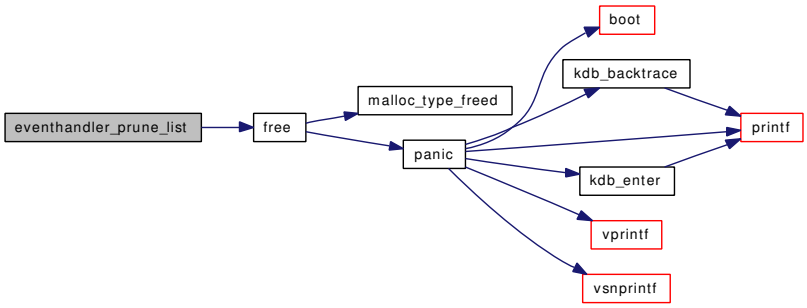
Here is the call graph for this function:

9.91.1.6 void eventhandler_prune_list (struct eventhandler_list * *list*)

Definition at line 225 of file subr_eventhandler.c.

References free().

Here is the call graph for this function:



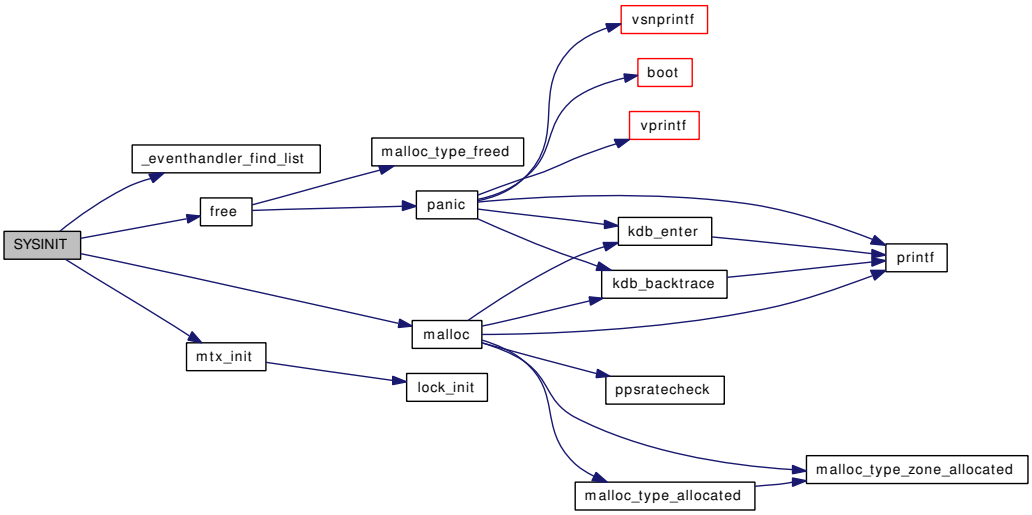
9.91.1.7 static MALLOC_DEFINE (M_EVENTHANDLER, "eventhandler", "Event handler records") [static]

9.91.1.8 SYSINIT (eventhandlers, SI_SUB_EVENTHANDLER, SI_ORDER_FIRST, eventhandler_init, NULL)

Definition at line 64 of file subr_eventhandler.c.

References _eventhandler_find_list(), free(), malloc(), and mtx_init().

Here is the call graph for this function:



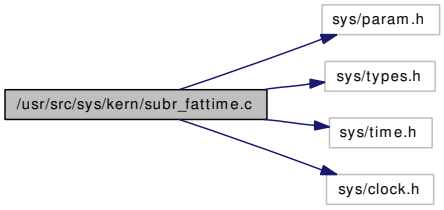
9.91.1.9 static TAILQ_HEAD (eventhandler_list) [static]

Definition at line 42 of file subr_eventhandler.c.

9.92 /usr/src/sys/kern/subr_fattime.c File Reference

```
#include <sys/param.h>
#include <sys/types.h>
#include <sys/time.h>
#include <sys/clock.h>
```

Include dependency graph for subr_fattime.c:



Defines

- #define DAY (24 ∗ 60 ∗ 60)
- #define YEAR 365
- #define LYC (4 ∗ YEAR + 1)
- #define T1980 (10 ∗ 365 + 2)
- #define JAN 31
- #define FEB (JAN + 28)
- #define MAR (FEB + 31)
- #define APR (MAR + 30)
- #define MAY (APR + 31)
- #define JUN (MAY + 30)
- #define JUL (JUN + 31)
- #define AUG (JUL + 31)
- #define SEP (AUG + 30)
- #define OCT (SEP + 31)
- #define NOV (OCT + 30)
- #define DEC (NOV + 31)
- #define ENC(y, m) (((y) << 9) | ((m) << 5))
- #define DCOD(m, y, l) ((m) + YEAR ∗ (y) + (l))

Functions

- void timespec2fattime (struct timespec ∗tsp, int utc, u_int16_t ∗ddp, u_int16_t ∗dtp, u_int8_t ∗dhp)
- void fattime2timespec (unsigned dd, unsigned dt, unsigned dh, int utc, struct timespec ∗tsp)

Variables

- struct {
  uint16_t days
  uint16_t coded

} mtab [48]

- static const uint16_t daytab [64]

9.92.1 Define Documentation

9.92.1.1 #define APR (MAR + 30)

Definition at line 87 of file subr_fattime.c.

9.92.1.2 #define AUG (JUL + 31)

Definition at line 91 of file subr_fattime.c.

9.92.1.3 #define DAY (24 * 60 * 60)

Definition at line 78 of file subr_fattime.c.

Referenced by fattime2timespec(), and timespec2fattime().

9.92.1.4 #define DCOD(m, y, l) ((m) + YEAR * (y) + (l))

Definition at line 197 of file subr_fattime.c.

9.92.1.5 #define DEC (NOV + 31)

Definition at line 95 of file subr_fattime.c.

9.92.1.6 #define ENC(y, m) (((y) << 9) | ((m) << 5))

Definition at line 99 of file subr_fattime.c.

9.92.1.7 #define FEB (JAN + 28)

Definition at line 85 of file subr_fattime.c.

Referenced by fattime2timespec(), and timespec2fattime().

9.92.1.8 #define JAN 31

Definition at line 84 of file subr_fattime.c.

9.92.1.9 #define JUL (JUN + 31)

Definition at line 90 of file subr_fattime.c.

9.92.1.10 #define JUN (MAY + 30)

Definition at line 89 of file subr_fattime.c.

9.92.1.11 #define LYC (4 ∗ YEAR + 1)

Definition at line 80 of file subr_fattime.c.

Referenced by fattime2timespec(), and timespec2fattime().

9.92.1.12 #define MAR (FEB + 31)

Definition at line 86 of file subr_fattime.c.

9.92.1.13 #define MAY (APR + 31)

Definition at line 88 of file subr_fattime.c.

9.92.1.14 #define NOV (OCT + 30)

Definition at line 94 of file subr_fattime.c.

9.92.1.15 #define OCT (SEP + 31)

Definition at line 93 of file subr_fattime.c.

9.92.1.16 #define SEP (AUG + 30)

Definition at line 92 of file subr_fattime.c.

9.92.1.17 #define T1980 (10 ∗ 365 + 2)

Definition at line 81 of file subr_fattime.c.

Referenced by fattime2timespec(), and timespec2fattime().

9.92.1.18 #define YEAR 365

Definition at line 79 of file subr_fattime.c.

9.92.2 Function Documentation

9.92.2.1 void fattime2timespec (unsigned *dd*, unsigned *dt*, unsigned *dh*, int *utc*, struct timespec ∗ *tsp*)

Definition at line 218 of file subr_fattime.c.

References DAY, daytab, FEB, LYC, T1980, and utc_offset().

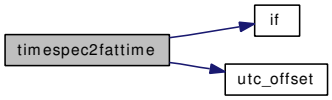
Here is the call graph for this function:

9.92.2.2 void timespec2fattime (struct timespec * *tsp*, int *utc*, u_int16_t * *ddp*, u_int16_t * *dtp*, u_int8_t * *dhp*)

Definition at line 138 of file subr_fattime.c.

References DAY, days, FEB, if(), LYC, mtab, T1980, and utc_offset().

Here is the call graph for this function:



9.92.3 Variable Documentation

9.92.3.1 uint16_t coded

Definition at line 103 of file subr_fattime.c.

9.92.3.2 uint16_t days

Definition at line 102 of file subr_fattime.c.

Referenced by clock_ct_to_ts(), clock_ts_to_ct(), and timespec2fattime().

9.92.3.3 const uint16_t daytab[64] [static]

Initial value:

```
{
        0,                DCOD(  0, 0, 0), DCOD(JAN, 0, 0), DCOD(FEB, 0, 1),
        DCOD(MAR, 0, 1), DCOD(APR, 0, 1), DCOD(MAY, 0, 1), DCOD(JUN, 0, 1),
        DCOD(JUL, 0, 1), DCOD(AUG, 0, 1), DCOD(SEP, 0, 1), DCOD(OCT, 0, 1),
        DCOD(NOV, 0, 1), DCOD(DEC, 0, 1), 0,                0,
        0,                DCOD(  0, 1, 1), DCOD(JAN, 1, 1), DCOD(FEB, 1, 1),
        DCOD(MAR, 1, 1), DCOD(APR, 1, 1), DCOD(MAY, 1, 1), DCOD(JUN, 1, 1),
        DCOD(JUL, 1, 1), DCOD(AUG, 1, 1), DCOD(SEP, 1, 1), DCOD(OCT, 1, 1),
        DCOD(NOV, 1, 1), DCOD(DEC, 1, 1), 0,                0,
        0,                DCOD(  0, 2, 1), DCOD(JAN, 2, 1), DCOD(FEB, 2, 1),
        DCOD(MAR, 2, 1), DCOD(APR, 2, 1), DCOD(MAY, 2, 1), DCOD(JUN, 2, 1),
        DCOD(JUL, 2, 1), DCOD(AUG, 2, 1), DCOD(SEP, 2, 1), DCOD(OCT, 2, 1),
        DCOD(NOV, 2, 1), DCOD(DEC, 2, 1), 0,                0,
        0,                DCOD(  0, 3, 1), DCOD(JAN, 3, 1), DCOD(FEB, 3, 1),
        DCOD(MAR, 3, 1), DCOD(APR, 3, 1), DCOD(MAY, 3, 1), DCOD(JUN, 3, 1),
        DCOD(JUL, 3, 1), DCOD(AUG, 3, 1), DCOD(SEP, 3, 1), DCOD(OCT, 3, 1),
        DCOD(NOV, 3, 1), DCOD(DEC, 3, 1), 0,                0
}
```

Definition at line 198 of file subr_fattime.c.

Referenced by fattime2timespec().

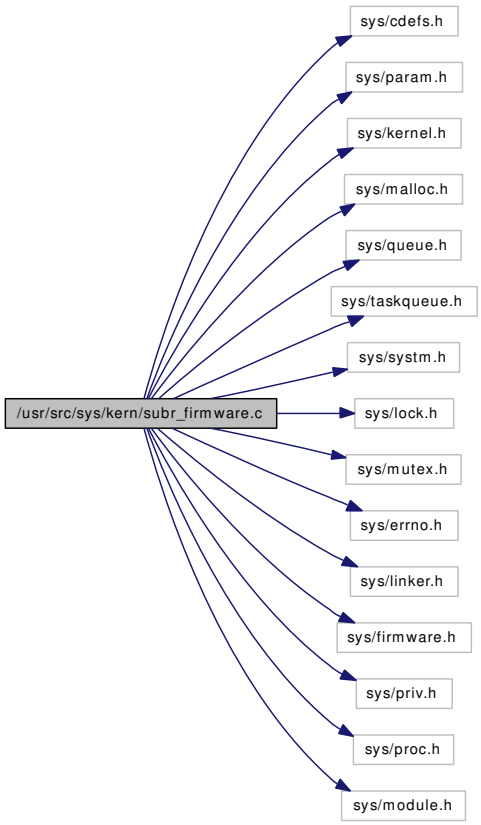
9.92.3.4 struct { ... } mtab[48] [static]

Referenced by timespec2fattime().

9.93 /usr/src/sys/kern/subr_firmware.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/kernel.h>
#include <sys/malloc.h>
#include <sys/queue.h>
#include <sys/taskqueue.h>
#include <sys/systm.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/errno.h>
#include <sys/linker.h>
#include <sys/firmware.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/module.h>
```

Include dependency graph for subr_firmware.c:



Data Structures

- struct priv_fw

Defines

- #define FW_INUSE(p) ((p) → file != NULL || (p) → fw.name != NULL)
- #define FW_UNLOAD 0x100
- #define PRIV_FW(x)
- #define FIRMWARE_MAX 30

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_firmware.c,v 1.9 2007/02/15 17:21:31 luigi Exp $")
- MTX_SYSINIT (firmware,&firmware_mtx,"firmware table", MTX_DEF)
- static struct priv_fw ∗ lookup (const char ∗name, struct priv_fw ∗∗empty_slot)
- firmware ∗ firmware_register (const char ∗imagename, const void ∗data, size_t datasize, unsigned int version, const struct firmware ∗parent)
- int firmware_unregister (const char ∗imagename)
- firmware ∗ firmware_get (const char ∗imagename)
- void firmware_put (const struct firmware ∗p, int flags)
- static void unloadentry (void ∗unused1, int unused2)
- static int firmware_modevent (module_t mod, int type, void ∗unused)
- DECLARE_MODULE (firmware, firmware_mod, SI_SUB_DRIVERS, SI_ORDER_FIRST)
- MODULE_VERSION (firmware, 1)

Variables

- static struct priv_fw firmware_table [FIRMWARE_MAX]
- task firmware_task
- mtx firmware_mtx
- static moduledata_t firmware_mod

9.93.1 Define Documentation

9.93.1.1 #define FIRMWARE_MAX 30

Definition at line 120 of file subr_firmware.c.

Referenced by firmware_modevent(), lookup(), and unloadentry().

9.93.1.2 #define FW_INUSE(p) ((p) → file != NULL || (p) → fw.name != NULL)

Definition at line 54 of file subr_firmware.c.

Referenced by lookup().

9.93.1.3 #define FW_UNLOAD 0x100

Definition at line 87 of file subr_firmware.c.

Referenced by firmware_modevent(), firmware_put(), and unloadentry().

9.93.1.4 #define PRIV_FW(x)

Value:

```
((struct priv_fw *)             \
        ((intptr_t)(x) - offsetof(struct priv_fw, fw)) )
```

Definition at line 111 of file subr_firmware.c.

Referenced by firmware_put(), and firmware_register().

9.93.2 Function Documentation

9.93.2.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_firmware. *c*, v 1.9 2007/02/15 17:21:31 luigi Exp $")

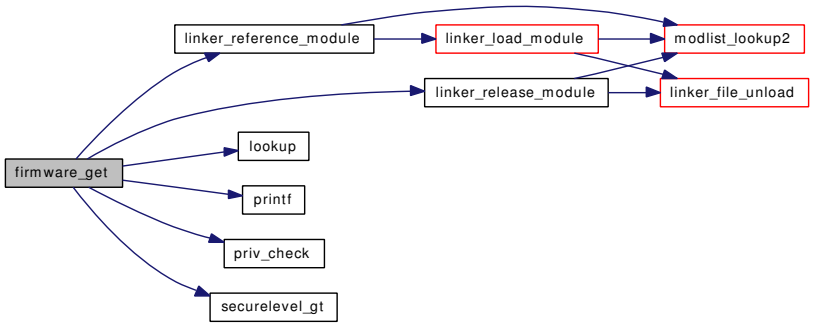
9.93.2.2 DECLARE_MODULE (firmware, firmware_mod, SI_SUB_DRIVERS, SI_ORDER_FIRST)

9.93.2.3 struct firmware* firmware_get (const char * *imagename*)

Definition at line 255 of file subr_firmware.c.

References priv_fw::file, firmware_mtx, priv_fw::fw, linker_reference_module(), linker_release_module(), lookup(), printf(), priv_check(), priv_fw::refcnt, securelevel_gt(), and td.

Here is the call graph for this function:

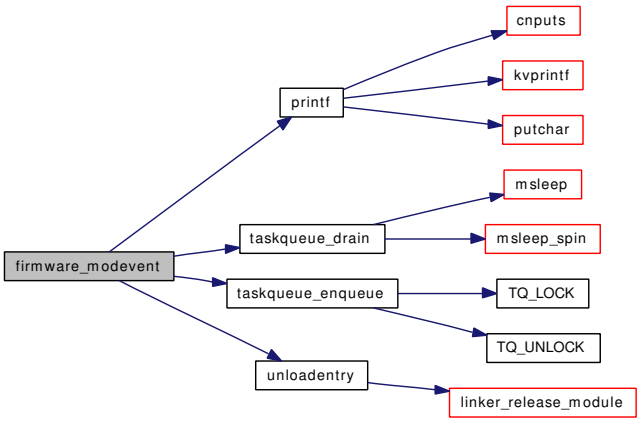


9.93.2.4 static int firmware_modevent (module_t *mod*, int *type*, void * *unused*) [static]

Definition at line 383 of file subr_firmware.c.

References FIRMWARE_MAX, firmware_mtx, firmware_table, firmware_task, priv_fw::flags, priv_-fw::fw, FW_UNLOAD, printf(), priv_fw::refcnt, taskqueue_drain(), taskqueue_enqueue(), and unloaden-try().

Here is the call graph for this function:

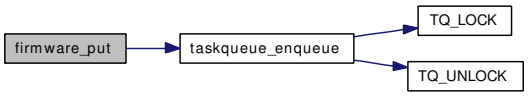


9.93.2.5 void firmware_put (const struct firmware * *p*, int *flags*)

Definition at line 307 of file subr_firmware.c.

References priv_fw::file, firmware_mtx, firmware_task, priv_fw::flags, FW_UNLOAD, PRIV_FW, priv_-fw::refcnt, and taskqueue_enqueue().

Here is the call graph for this function:

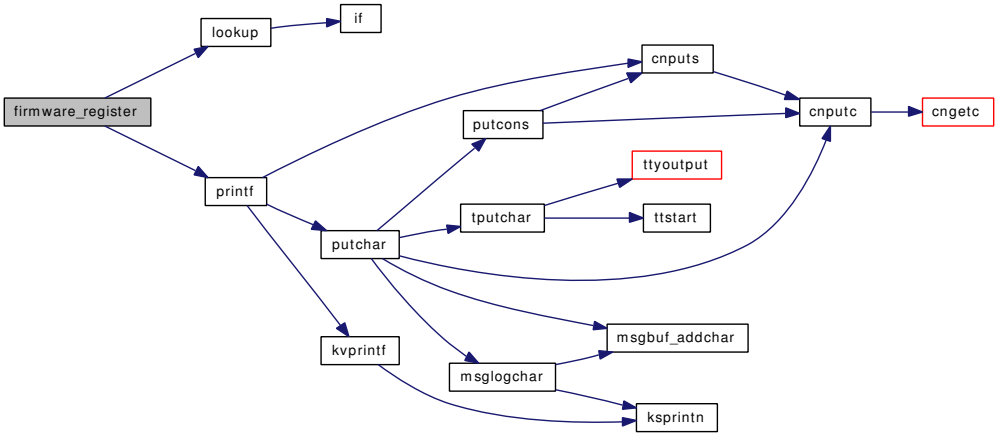


9.93.2.6 struct firmware* firmware_register (const char * *imagename*, const void * *data*, size_t *datasize*, unsigned int *version*, const struct firmware * *parent*)

Definition at line 168 of file subr_firmware.c.

References bootverbose, firmware_mtx, lookup(), printf(), and PRIV_FW.

Here is the call graph for this function:



9.93.2.7 int firmware_unregister (const char * *imagename*)

Definition at line 212 of file subr_firmware.c.

References priv_fw::file, firmware_mtx, lookup(), priv_fw::parent, and priv_fw::refcnt.

Here is the call graph for this function:

9.93.2.8 static struct priv_fw* lookup (const char * *name*, struct priv_fw ** *empty_slot*) [static]

Definition at line 142 of file subr_firmware.c.

References dummy, FIRMWARE_MAX, firmware_table, priv_fw::fw, FW_INUSE, and if().

Referenced by firmware_get(), firmware_register(), firmware_unregister(), and namei().

Here is the call graph for this function:

9.93.2.9 MODULE_VERSION (firmware, 1)

9.93.2.10 MTX_SYSINIT (firmware, &*firmware_mtx*, "firmware table", MTX_DEF)

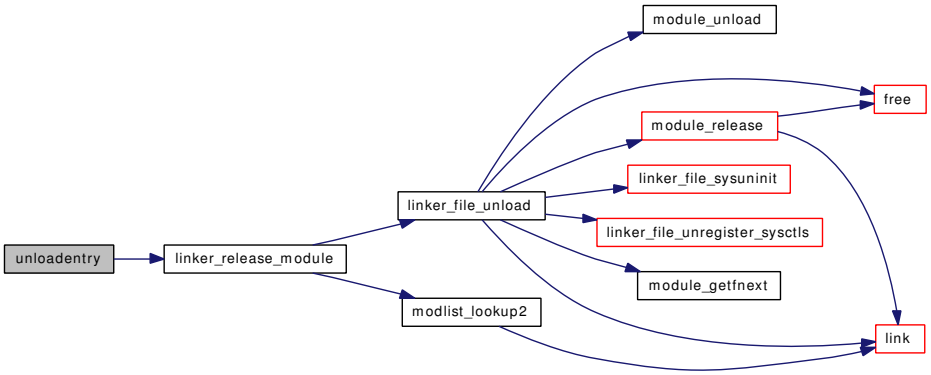
9.93.2.11 static void unloadentry (void * *unused1*, int *unused2*) [static]

Definition at line 330 of file subr_firmware.c.

References FIRMWARE_MAX, firmware_mtx, firmware_table, FW_UNLOAD, and linker_release_module().

Referenced by firmware_modevent().

Here is the call graph for this function:



9.93.3 Variable Documentation

9.93.3.1 moduledata_t firmware_mod [static]

Initial value:

```
{
        "firmware",
        firmware_modevent,
        0
}
```

Definition at line 417 of file subr_firmware.c.

9.93.3.2 struct mtx firmware_mtx

Definition at line 131 of file subr_firmware.c.

Referenced by firmware_get(), firmware_modevent(), firmware_put(), firmware_register(), firmware_-unregister(), and unloadentry().

9.93.3.3 struct priv_fw firmware_table[FIRMWARE_MAX] [static]

Definition at line 121 of file subr_firmware.c.

Referenced by firmware_modevent(), lookup(), and unloadentry().

9.93.3.4 struct task firmware_task

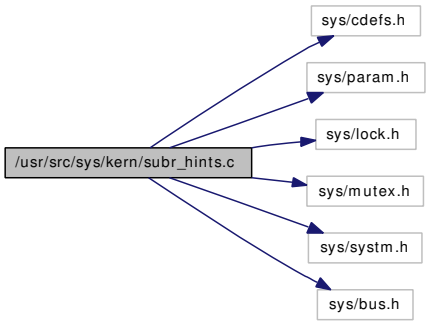
Definition at line 126 of file subr_firmware.c.

Referenced by firmware_modevent(), and firmware_put().

9.94 /usr/src/sys/kern/subr_hints.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/systm.h>
#include <sys/bus.h>
```

Include dependency graph for subr_hints.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_hints.c,v 1.13 2006/07/09 21:42:58 scottl Exp $")
- static int res_find (int *line, int *startln, const char *name, int *unit, const char *resname, const char *value, const char **ret_name, int *ret_namelen, int *ret_unit, const char **ret_resname, int *ret_resnamelen, const char **ret_value)
- static int resource_find (int *line, int *startln, const char *name, int *unit, const char *resname, const char *value, const char **ret_name, int *ret_namelen, int *ret_unit, const char **ret_resname, int *ret_resnamelen, const char **ret_value)
- int resource_int_value (const char *name, int unit, const char *resname, int *result)
- int resource_long_value (const char *name, int unit, const char *resname, long *result)
- int resource_string_value (const char *name, int unit, const char *resname, const char **result)
- static const char * resource_string_copy (const char *s, int len)
- int resource_find_match (int *anchor, const char **name, int *unit, const char *resname, const char *value)
- int resource_find_dev (int *anchor, const char *name, int *unit, const char *resname, const char *value)
- int resource_disabled (const char *name, int unit)

Variables

- static int checkmethod = 1
- static int use_kenv
- static char * hintp

9.94.1 Function Documentation

9.94.1.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_hints. *c*, v 1.13 2006/07/09 21:42:58 scottl Exp $")

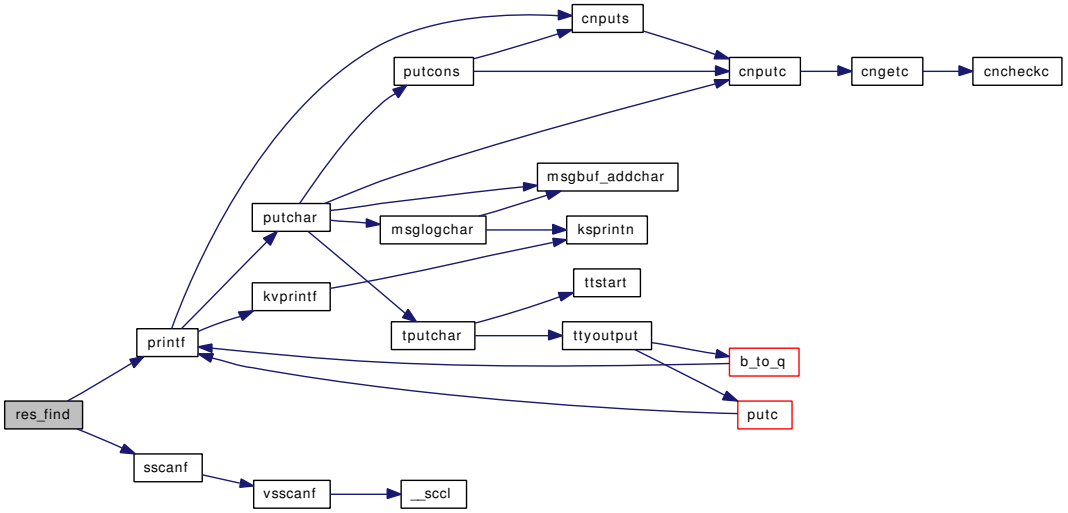
9.94.1.2 static int res_find (int * *line*, int * *startln*, const char * *name*, int * *unit*, const char * *resname*, const char * *value*, const char ** *ret_name*, int * *ret_namelen*, int * *ret_unit*, const char ** *ret_resname*, int * *ret_resnamelen*, const char ** *ret_value*) [static]

Definition at line 50 of file subr_hints.c.

References checkmethod, dynamic_kenv, hintp, kenv_lock, kenvp, kern_envp, printf(), sscanf(), and use_-kenv.

Referenced by resource_find().

Here is the call graph for this function:

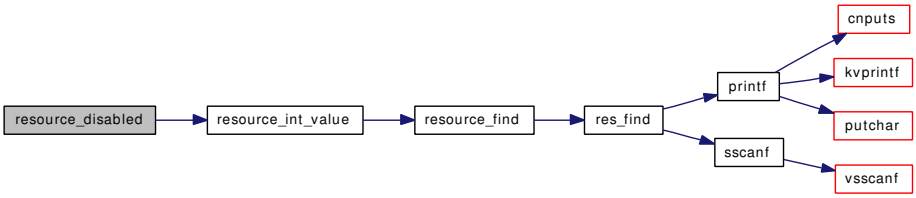


9.94.1.3 int resource_disabled (const char * *name*, int *unit*)

Definition at line 375 of file subr_hints.c.

References resource_int_value().

Here is the call graph for this function:



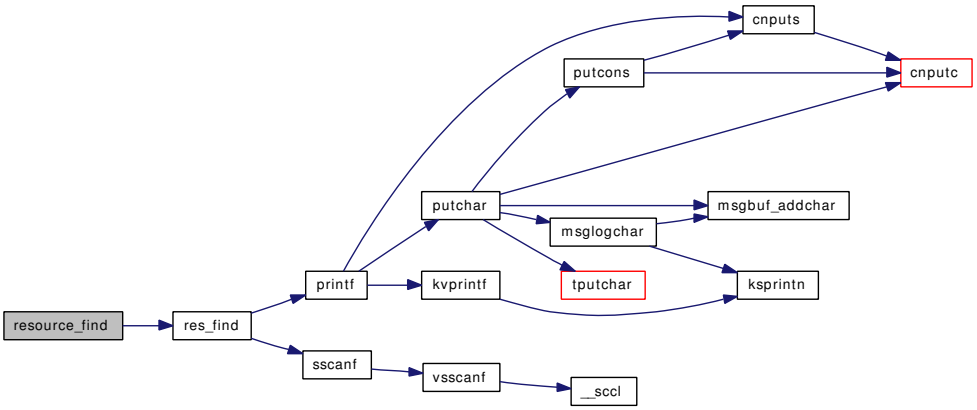
9.94.1.4 static int resource_find (int * *line*, int * *startln*, const char * *name*, int * *unit*, const char * *resname*, const char * *value*, const char ** *ret_name*, int * *ret_namelen*, int * *ret_unit*, const char ** *ret_resname*, int * *ret_resnamelen*, const char ** *ret_value*) [static]

Definition at line 201 of file subr_hints.c.

References res_find().

Referenced by resource_find_dev(), resource_find_match(), resource_int_value(), resource_long_value(), and resource_string_value().

Here is the call graph for this function:

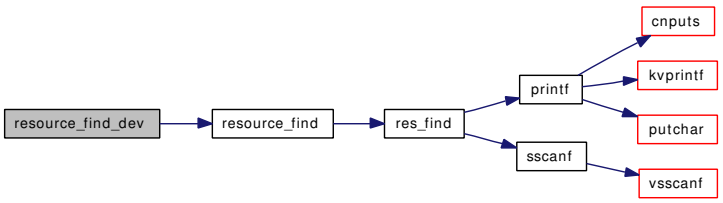


9.94.1.5 int resource_find_dev (int * *anchor*, const char * *name*, int * *unit*, const char * *resname*, const char * *value*)

Definition at line 354 of file subr_hints.c.

References resource_find(), and ret.

Here is the call graph for this function:



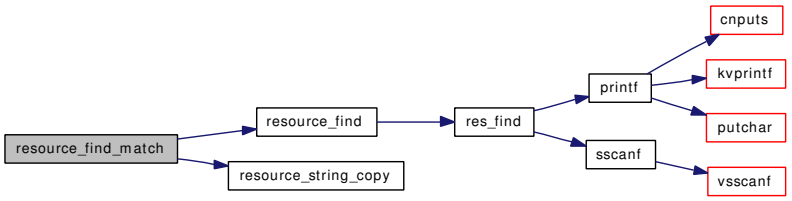
9.94.1.6 int resource_find_match (int * *anchor*, const char ** *name*, int * *unit*, const char * *resname*, const char * *value*)

Definition at line 325 of file subr_hints.c.

References resource_find(), resource_string_copy(), and ret.

Referenced by bus_enumerate_hinted_children().

Here is the call graph for this function:



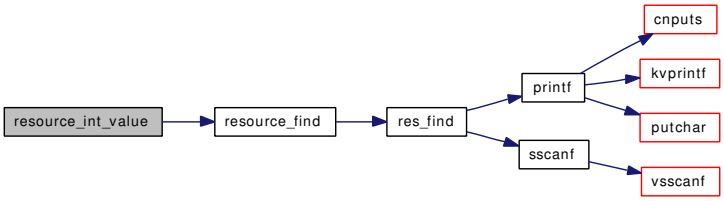
9.94.1.7 int resource_int_value (const char * *name*, int *unit*, const char * *resname*, int * *result*)

Definition at line 230 of file subr_hints.c.

References resource_find().

Referenced by device_probe_child(), and resource_disabled().

Here is the call graph for this function:

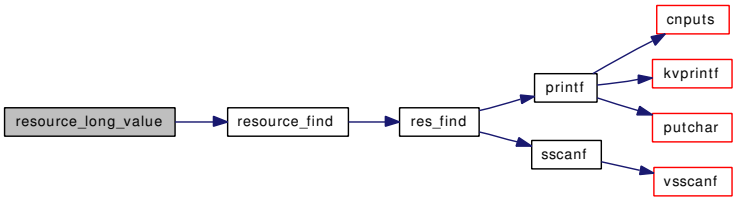


9.94.1.8 int resource_long_value (const char * *name*, int *unit*, const char * *resname*, long * *result*)

Definition at line 253 of file subr_hints.c.

References resource_find().

Here is the call graph for this function:



9.94.1.9 static const char* resource_string_copy (const char * *s*, int *len*) [static]

Definition at line 297 of file subr_hints.c.

References ret.

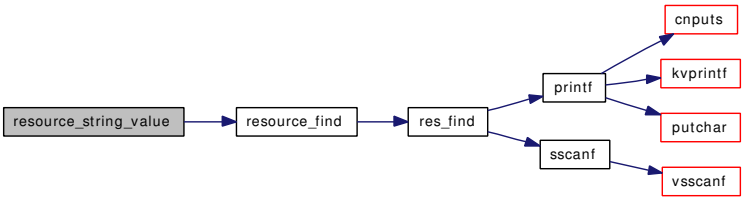
Referenced by resource_find_match().

9.94.1.10 int resource_string_value (const char * *name*, int *unit*, const char * *resname*, const char ** *result*)

Definition at line 277 of file subr_hints.c.

References resource_find().

Here is the call graph for this function:



9.94.2 Variable Documentation

9.94.2.1 int checkmethod = 1 [static]

Definition at line 40 of file subr_hints.c.

Referenced by res_find().

9.94.2.2 char* hintp [static]

Definition at line 42 of file subr_hints.c.

Referenced by res_find().

9.94.2.3 int use_kenv [static]

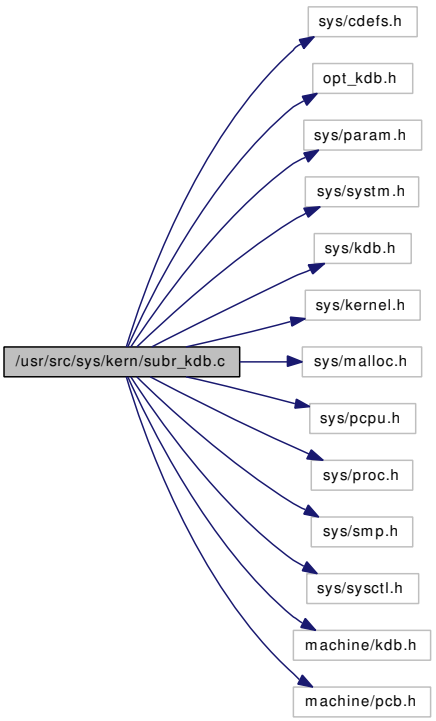
Definition at line 41 of file subr_hints.c.

Referenced by res_find().

9.95 /usr/src/sys/kern/subr_kdb.c File Reference

```
#include <sys/cdefs.h>
#include "opt_kdb.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kdb.h>
#include <sys/kernel.h>
#include <sys/malloc.h>
#include <sys/pcpu.h>
#include <sys/proc.h>
#include <sys/smp.h>
#include <sys/sysctl.h>
#include <machine/kdb.h>
#include <machine/pcb.h>
```

Include dependency graph for subr_kdb.c:



Defines

- #define KEY_CR 13
- #define KEY_TILDE 126
- #define KEY_CRTLB 2

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_kdb.c,v 1.23 2006/06/18 12:27:59 yar Exp $")
- KDB_BACKEND (null, NULL, NULL, NULL)
- SET_DECLARE (kdb_dbbe_set, struct kdb_dbbe)
- static int kdb_sysctl_available (SYSCTL_HANDLER_ARGS)
- static int kdb_sysctl_current (SYSCTL_HANDLER_ARGS)
- static int kdb_sysctl_enter (SYSCTL_HANDLER_ARGS)
- static int kdb_sysctl_panic (SYSCTL_HANDLER_ARGS)
- static int kdb_sysctl_trap (SYSCTL_HANDLER_ARGS)
- static int kdb_sysctl_trap_code (SYSCTL_HANDLER_ARGS)
- SYSCTL_NODE (_debug, OID_AUTO, kdb, CTLFLAG_RW, NULL,"KDB nodes")
- SYSCTL_PROC (_debug_kdb, OID_AUTO, available, CTLTYPE_STRING|CTLFLAG_RD, 0, 0, kdb_sysctl_available,"A","list of available KDB backends")
- SYSCTL_PROC (_debug_kdb, OID_AUTO, current, CTLTYPE_STRING|CTLFLAG_RW, 0, 0, kdb_sysctl_current,"A","currently selected KDB backend")
- SYSCTL_PROC (_debug_kdb, OID_AUTO, enter, CTLTYPE_INT|CTLFLAG_RW, 0, 0, kdb_-sysctl_enter,"I","set to enter the debugger")
- SYSCTL_PROC (_debug_kdb, OID_AUTO, panic, CTLTYPE_INT|CTLFLAG_RW, 0, 0, kdb_-sysctl_panic,"I","set to panic the kernel")
- SYSCTL_PROC (_debug_kdb, OID_AUTO, trap, CTLTYPE_INT|CTLFLAG_RW, 0, 0, kdb_-sysctl_trap,"I","set to cause a page fault via data access")
- SYSCTL_PROC (_debug_kdb, OID_AUTO, trap_code, CTLTYPE_INT|CTLFLAG_RW, 0, 0, kdb_sysctl_trap_code,"I","set to cause a page fault via code access")
- int kdb_alt_break (int key, int *state)
- void kdb_backtrace ()
- int kdb_dbbe_select (const char *name)
- void kdb_enter (const char *msg)
- void kdb_init ()
- void * kdb_jmpbuf (jmp_buf new)
- void kdb_reenter (void)
- pcb * kdb_thr_ctx (struct thread *thr)
- thread * kdb_thr_first (void)
- thread * kdb_thr_from_pid (pid_t pid)
- thread * kdb_thr_lookup (lwpid_t tid)
- thread * kdb_thr_next (struct thread *thr)
- int kdb_thr_select (struct thread *thr)
- int kdb_trap (int type, int code, struct trapframe *tf)

Variables

- int kdb_active = 0
- void * kdb_jmpbufp = NULL
- kdb_dbbe * kdb_dbbe = NULL
- pcb kdb_pcb
- pcb * kdb_thrctx = NULL
- thread * kdb_thread = NULL
- trapframe * kdb_frame = NULL

9.95.1 Define Documentation

9.95.1.1 #define KEY_CR 13

Definition at line 230 of file subr_kdb.c.

Referenced by kdb_alt_break().

9.95.1.2 #define KEY_CRTLB 2

Definition at line 232 of file subr_kdb.c.

Referenced by kdb_alt_break().

9.95.1.3 #define KEY_TILDE 126

Definition at line 231 of file subr_kdb.c.

Referenced by kdb_alt_break().

9.95.2 Function Documentation

9.95.2.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_kdb. *c*, v 1.23 2006/06/18 12:27:59 yar Exp $")

9.95.2.2 int kdb_alt_break (int *key*, int * *state*)

Definition at line 235 of file subr_kdb.c.

References KEY_CR, KEY_CRTLB, and KEY_TILDE.

9.95.2.3 KDB_BACKEND (null, NULL, NULL, NULL)

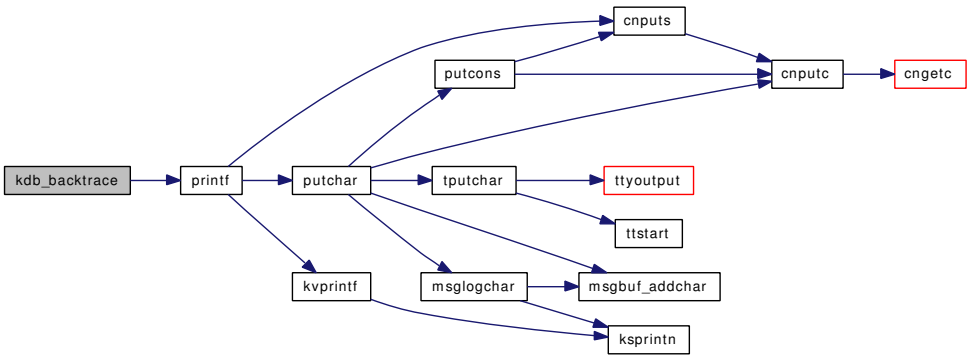
9.95.2.4 void kdb_backtrace ()

Definition at line 266 of file subr_kdb.c.

References kdb_dbbe, and printf().

Referenced by _lockmgr(), malloc(), mi_switch(), panic(), and witness_warn().

Here is the call graph for this function:



9.95.2.5 int kdb_dbbe_select (const char * *name*)

Definition at line 280 of file subr_kdb.c.

References kdb_dbbe.

Referenced by kdb_sysctl_current().

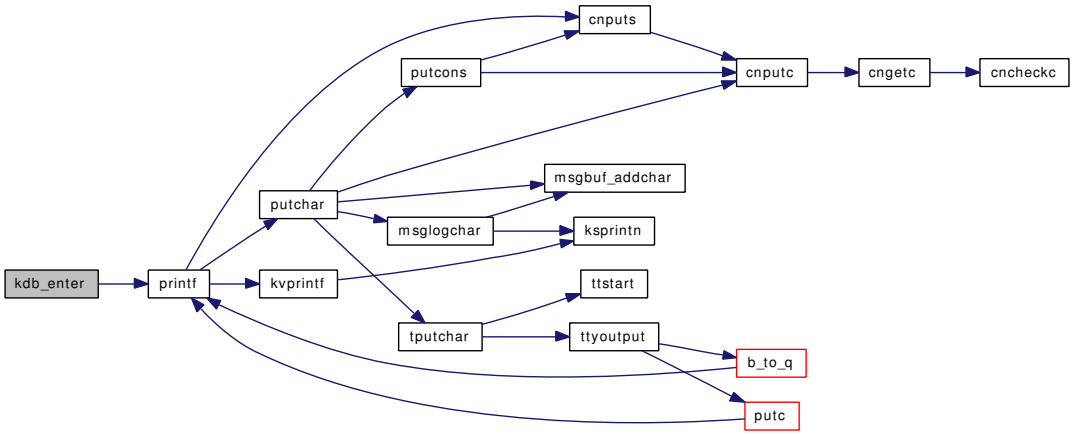
9.95.2.6 void kdb_enter (const char * *msg*)

Definition at line 302 of file subr_kdb.c.

References kdb_active, kdb_dbbe, and printf().

Referenced by kdb_sysctl_enter(), malloc(), panic(), vop_strategy_pre(), and witness_warn().

Here is the call graph for this function:

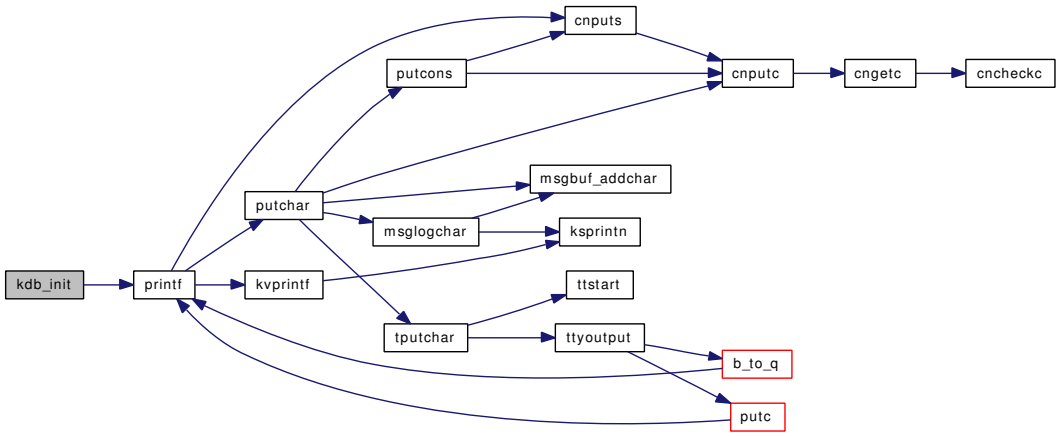


9.95.2.7 void kdb_init ()

Definition at line 317 of file subr_kdb.c.

References kdb_active, kdb_dbbe, and printf().

Here is the call graph for this function:



9.95.2.8 void* kdb_jmpbuf (jmp_buf *new*)

Definition at line 352 of file subr_kdb.c.

References kdb_jmpbufp.

9.95.2.9 void kdb_reenter (void)

Definition at line 362 of file subr_kdb.c.

References kdb_active, and kdb_jmpbufp.

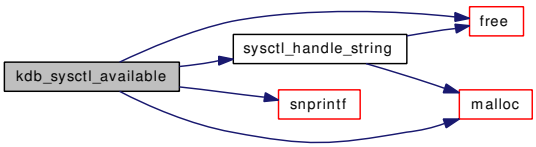
Referenced by mi_switch().

9.95.2.10 static int kdb_sysctl_available (SYSCTL_HANDLER_ARGS) [static]

Definition at line 101 of file subr_kdb.c.

References free(), kdb_dbbe, malloc(), snprintf(), and sysctl_handle_string().

Here is the call graph for this function:

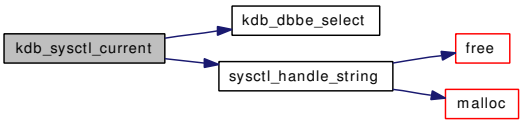


9.95.2.11 static int kdb_sysctl_current (SYSCTL_HANDLER_ARGS) [static]

Definition at line 134 of file subr_kdb.c.

References buf, kdb_active, kdb_dbbe, kdb_dbbe_select(), and sysctl_handle_string().

Here is the call graph for this function:

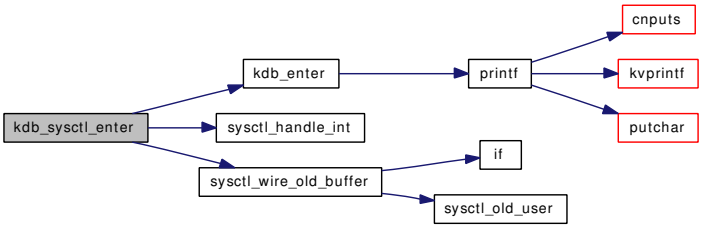


9.95.2.12 static int kdb_sysctl_enter (SYSCTL_HANDLER_ARGS) [static]

Definition at line 153 of file subr_kdb.c.

References kdb_active, kdb_enter(), sysctl_handle_int(), and sysctl_wire_old_buffer().

Here is the call graph for this function:

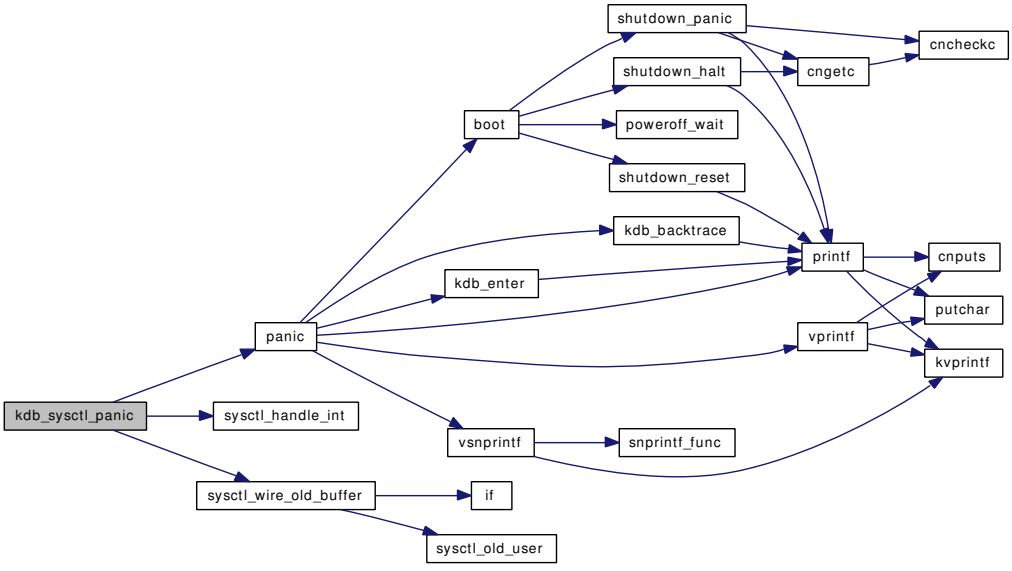


9.95.2.13 static int kdb_sysctl_panic (SYSCTL_HANDLER_ARGS) [static]

Definition at line 171 of file subr_kdb.c.

References panic(), sysctl_handle_int(), and sysctl_wire_old_buffer().

Here is the call graph for this function:

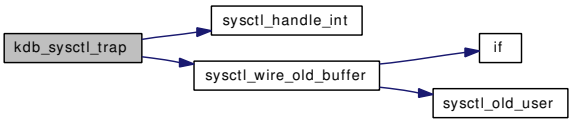


9.95.2.14 static int kdb_sysctl_trap (SYSCTL_HANDLER_ARGS) [static]

Definition at line 187 of file subr_kdb.c.

References sysctl_handle_int(), and sysctl_wire_old_buffer().

Here is the call graph for this function:

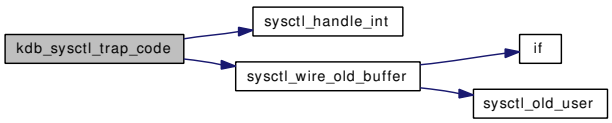


9.95.2.15 static int kdb_sysctl_trap_code (SYSCTL_HANDLER_ARGS) [static]

Definition at line 203 of file subr_kdb.c.

References sysctl_handle_int(), and sysctl_wire_old_buffer().

Here is the call graph for this function:



9.95.2.16 struct pcb* kdb_thr_ctx (struct thread * *thr*)

Definition at line 377 of file subr_kdb.c.

References cpuhead, and kdb_pcb.

Referenced by kdb_thr_select().

9.95.2.17 struct thread* kdb_thr_first (void)

Definition at line 396 of file subr_kdb.c.

References allproc.

Referenced by kdb_thr_lookup().

9.95.2.18 struct thread* kdb_thr_from_pid (pid_t *pid*)

Definition at line 414 of file subr_kdb.c.

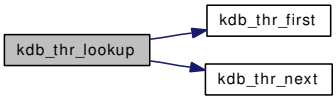
References allproc.

9.95.2.19 struct thread* kdb_thr_lookup (lwpid_t *tid*)

Definition at line 428 of file subr_kdb.c.

References kdb_thr_first(), and kdb_thr_next().

Here is the call graph for this function:



9.95.2.20 struct thread* kdb_thr_next (struct thread * *thr*)

Definition at line 439 of file subr_kdb.c.

Referenced by kdb_thr_lookup().

9.95.2.21 int kdb_thr_select (struct thread * *thr*)

Definition at line 456 of file subr_kdb.c.

References kdb_thr_ctx(), kdb_thrctx, and kdb_thread.

Referenced by kdb_trap().

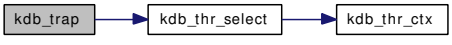
Here is the call graph for this function:

9.95.2.22 int kdb_trap (int *type*, int *code*, struct trapframe * *tf*)

Definition at line 470 of file subr_kdb.c.

References kdb_active, kdb_dbbe, kdb_frame, kdb_pcb, and kdb_thr_select().

Here is the call graph for this function:



9.95.2.23 SET_DECLARE (kdb_dbbe_set, struct *kdb_dbbe*)

9.95.2.24 SYSCTL_NODE (_debug, OID_AUTO, kdb, CTLFLAG_RW, NULL, "KDB nodes")

9.95.2.25 SYSCTL_PROC (_debug_kdb, OID_AUTO, trap_code, CTLTYPE_INT| *CTLFLAG_RW*, 0, 0, kdb_sysctl_trap_code, "I", "set to cause a page fault via code access")

9.95.2.26 SYSCTL_PROC (_debug_kdb, OID_AUTO, trap, CTLTYPE_INT| *CTLFLAG_RW*, 0, 0, kdb_sysctl_trap, "I", "set to cause a page fault via data access")

9.95.2.27 SYSCTL_PROC (_debug_kdb, OID_AUTO, panic, CTLTYPE_INT| *CTLFLAG_RW*, 0, 0, kdb_sysctl_panic, "I", "set to panic the kernel")

9.95.2.28 SYSCTL_PROC (_debug_kdb, OID_AUTO, enter, CTLTYPE_INT| *CTLFLAG_RW*, 0, 0, kdb_sysctl_enter, "I", "set to enter the debugger")

9.95.2.29 SYSCTL_PROC (_debug_kdb, OID_AUTO, current, CTLTYPE_STRING| *CTLFLAG_RW*, 0, 0, kdb_sysctl_current, "A", "currently selected KDB backend")

9.95.2.30 SYSCTL_PROC (_debug_kdb, OID_AUTO, available, CTLTYPE_STRING| *CTLFLAG_RD*, 0, 0, kdb_sysctl_available, "A", "list of available KDB backends")

9.95.3 Variable Documentation

9.95.3.1 int kdb_active = 0

Definition at line 49 of file subr_kdb.c.

Referenced by cncheckc(), cnputc(), kdb_enter(), kdb_init(), kdb_reenter(), kdb_sysctl_current(), kdb_sysctl_enter(), kdb_trap(), lockstatus(), mi_switch(), putchar(), and witness_checkorder().

9.95.3.2 struct kdb_dbbe* kdb_dbbe = NULL

Definition at line 51 of file subr_kdb.c.

Referenced by kdb_backtrace(), kdb_dbbe_select(), kdb_enter(), kdb_init(), kdb_sysctl_available(), kdb_sysctl_current(), and kdb_trap().

9.95.3.3 struct trapframe* kdb_frame = NULL

Definition at line 55 of file subr_kdb.c.

Referenced by kdb_trap().

9.95.3.4 void* kdb_jmpbufp = NULL

Definition at line 50 of file subr_kdb.c.

Referenced by kdb_jmpbuf(), and kdb_reenter().

9.95.3.5 struct pcb kdb_pcb

Definition at line 52 of file subr_kdb.c.

Referenced by kdb_thr_ctx(), and kdb_trap().

9.95.3.6 struct pcb* kdb_thrctx = NULL

Definition at line 53 of file subr_kdb.c.

Referenced by kdb_thr_select().

9.95.3.7 struct thread* kdb_thread = NULL

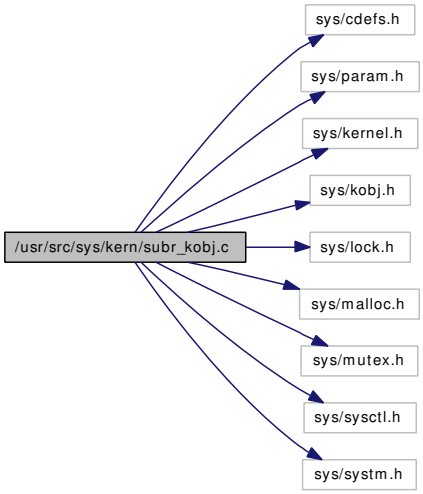
Definition at line 54 of file subr_kdb.c.

Referenced by kdb_thr_select().

9.96 /usr/src/sys/kern/subr_kobj.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/kernel.h>
#include <sys/kobj.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/sysctl.h>
#include <sys/systm.h>
```

Include dependency graph for subr_kobj.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_kobj.c,v 1.10 2005/12/29 18:00:42 jhb Exp $")
- static MALLOC_DEFINE (M_KOBJ,"kobj","Kernel object structures")
- SYSCTL_UINT (_kern, OID_AUTO, kobj_methodcount, CTLFLAG_RD,&kobj_next_id, 0,"")
- static void kobj_init_mutex (void *arg)
- SYSINIT (kobj, SI_SUB_LOCK, SI_ORDER_ANY, kobj_init_mutex, NULL)
- void kobj_machdep_init (void)
- int kobj_error_method (void)
- static void kobj_register_method (struct kobjop_desc *desc)
- static void kobj_unregister_method (struct kobjop_desc *desc)
- static void kobj_class_compile_common (kobj_class_t cls, kobj_ops_t ops)
- void kobj_class_compile (kobj_class_t cls)
- void kobj_class_compile_static (kobj_class_t cls, kobj_ops_t ops)
- static kobj_method_t * kobj_lookup_method_class (kobj_class_t cls, kobjop_desc_t desc)
- static kobj_method_t * kobj_lookup_method_mi (kobj_class_t cls, kobjop_desc_t desc)
- kobj_method_t * kobj_lookup_method (kobj_class_t cls, kobj_method_t **cep, kobjop_desc_t desc)

- void kobj_class_free (kobj_class_t cls)
- kobj_t kobj_create (kobj_class_t cls, struct malloc_type *mtype, int mflags)
- void kobj_init (kobj_t obj, kobj_class_t cls)
- void kobj_delete (kobj_t obj, struct malloc_type *mtype)

Variables

- static struct mtx kobj_mtx
- static int kobj_mutex_inited
- static int kobj_next_id = 1
- static struct kobj_method null_method

9.96.1 Function Documentation

9.96.1.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_kobj. *c*, v 1.10 2005/12/29 18:00:42 jhb Exp $")

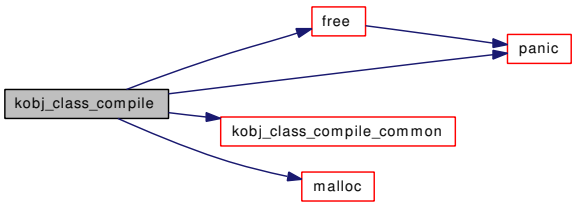
9.96.1.2 void kobj_class_compile (kobj_class_t *cls*)

Definition at line 144 of file subr_kobj.c.

References free(), kobj_class_compile_common(), kobj_mtx, malloc(), and panic().

Referenced by devclass_add_driver(), kobj_init(), linker_add_class(), and root_bus_module_handler().

Here is the call graph for this function:



9.96.1.3 static void kobj_class_compile_common (kobj_class_t *cls*, kobj_ops_t *ops*) [static]

Definition at line 115 of file subr_kobj.c.

References kobj_mtx, kobj_register_method(), and null_method.

Referenced by kobj_class_compile(), and kobj_class_compile_static().

Here is the call graph for this function:

9.96.1.4 void kobj_class_compile_static (kobj_class_t *cls*, kobj_ops_t *ops*)

Definition at line 175 of file subr_kobj.c.

References kobj_class_compile_common(), and kobj_mtx.

Here is the call graph for this function:

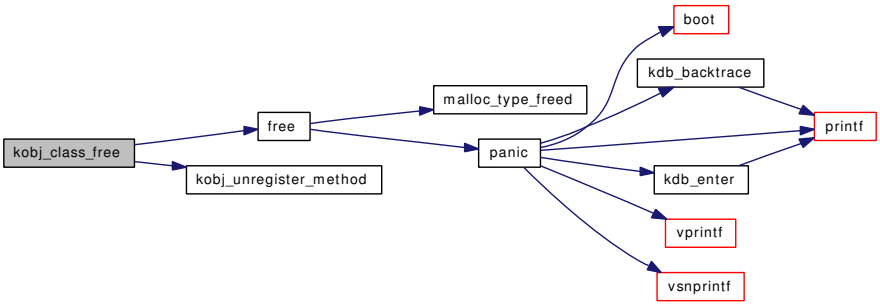
9.96.1.5 void kobj_class_free (kobj_class_t *cls*)

Definition at line 251 of file subr_kobj.c.

References free(), kobj_mtx, and kobj_unregister_method().

Referenced by kobj_delete().

Here is the call graph for this function:



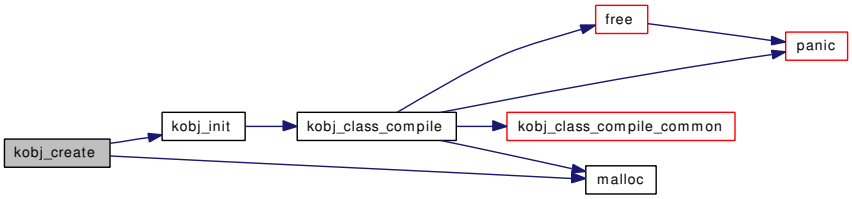
9.96.1.6 kobj_t kobj_create (kobj_class_t *cls*, struct malloc_type * *mtype*, int *mflags*)

Definition at line 285 of file subr_kobj.c.

References kobj_init(), and malloc().

Referenced by linker_make_file().

Here is the call graph for this function:



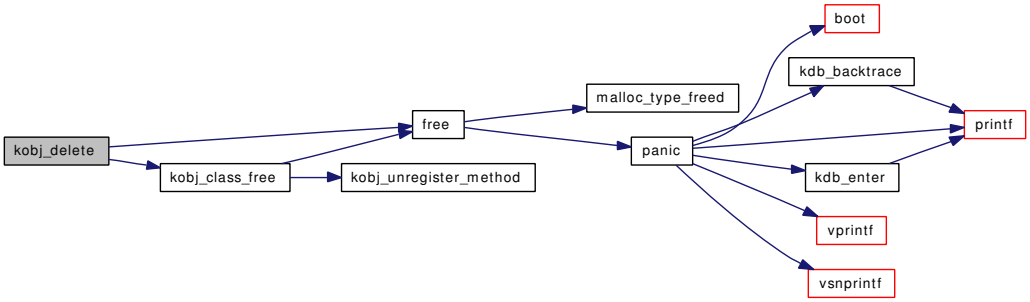
9.96.1.7 void kobj_delete (kobj_t *obj*, struct malloc_type * *mtype*)

Definition at line 330 of file subr_kobj.c.

References free(), kobj_class_free(), and kobj_mtx.

Referenced by device_delete_child(), device_set_driver(), and make_device().

Here is the call graph for this function:



9.96.1.8 int kobj_error_method (void)

Definition at line 93 of file subr_kobj.c.

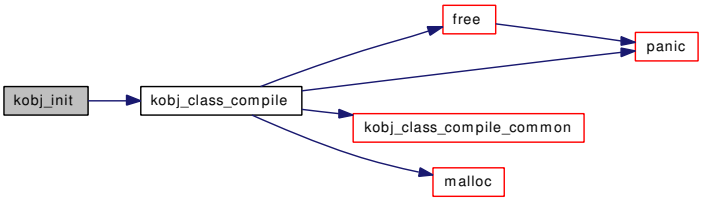
9.96.1.9 void kobj_init (kobj_t *obj*, kobj_class_t *cls*)

Definition at line 303 of file subr_kobj.c.

References kobj_class_compile(), and kobj_mtx.

Referenced by device_set_driver(), kobj_create(), make_device(), and root_bus_module_handler().

Here is the call graph for this function:



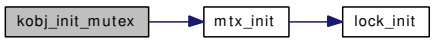
9.96.1.10 static void kobj_init_mutex (void * *arg*) [static]

Definition at line 67 of file subr_kobj.c.

References kobj_mtx, kobj_mutex_inited, and mtx_init().

Referenced by kobj_machdep_init().

Here is the call graph for this function:

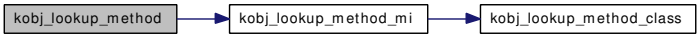


9.96.1.11 kobj_method_t* kobj_lookup_method (kobj_class_t *cls*, kobj_method_t ** *cep*, kobjop_desc_t *desc*)

Definition at line 228 of file subr_kobj.c.

References kobj_lookup_method_mi().

Here is the call graph for this function:



9.96.1.12 static kobj_method_t* kobj_lookup_method_class (kobj_class_t *cls*, kobjop_desc_t *desc*) [static]

Definition at line 190 of file subr_kobj.c.

Referenced by kobj_lookup_method_mi().

9.96.1.13 static kobj_method_t* kobj_lookup_method_mi (kobj_class_t *cls*, kobjop_desc_t *desc*) [static]

Definition at line 205 of file subr_kobj.c.

References kobj_lookup_method_class().

Referenced by kobj_lookup_method().

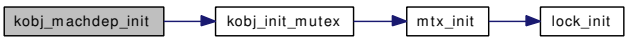
Here is the call graph for this function:

9.96.1.14 void kobj_machdep_init (void)

Definition at line 78 of file subr_kobj.c.

References kobj_init_mutex().

Here is the call graph for this function:



9.96.1.15 static void kobj_register_method (struct kobjop_desc * *desc*) [static]

Definition at line 100 of file subr_kobj.c.

References kobj_mtx, and kobj_next_id.

Referenced by kobj_class_compile_common().

9.96.1.16 static void kobj_unregister_method (struct kobjop_desc * *desc*) [static]

Definition at line 110 of file subr_kobj.c.

Referenced by kobj_class_free().

9.96.1.17 static MALLOC_DEFINE (M_KOBJ, "kobj", "Kernel object structures") [static]

9.96.1.18 SYSCTL_UINT (_kern, OID_AUTO, kobj_methodcount, CTLFLAG_RD, & *kobj_next_id*, 0, "")

9.96.1.19 SYSINIT (kobj, SI_SUB_LOCK, SI_ORDER_ANY, kobj_init_mutex, NULL)

9.96.2 Variable Documentation

9.96.2.1 struct mtx kobj_mtx [static]

Definition at line 59 of file subr_kobj.c.

Referenced by kobj_class_compile(), kobj_class_compile_common(), kobj_class_compile_static(), kobj_class_free(), kobj_delete(), kobj_init(), kobj_init_mutex(), and kobj_register_method().

9.96.2.2 int kobj_mutex_inited [static]

Definition at line 60 of file subr_kobj.c.

Referenced by kobj_init_mutex().

9.96.2.3 int kobj_next_id = 1 [static]

Definition at line 61 of file subr_kobj.c.

Referenced by kobj_register_method().

9.96.2.4 struct kobj_method null_method [static]

Initial value:

```
{
        0, 0,
}
```

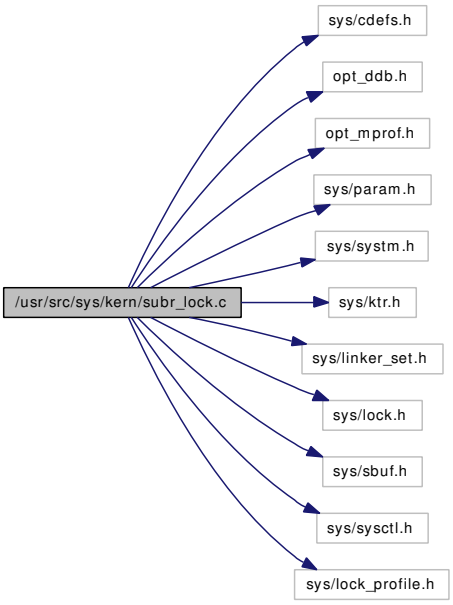
Definition at line 88 of file subr_kobj.c.

Referenced by kobj_class_compile_common().

9.97 /usr/src/sys/kern/subr_lock.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ddb.h"
#include "opt_mprof.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/ktr.h>
#include <sys/linker_set.h>
#include <sys/lock.h>
#include <sys/sbuf.h>
#include <sys/sysctl.h>
#include <sys/lock_profile.h>
```

Include dependency graph for subr_lock.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_lock.c,v 1.12 2006/12/04 22:15:50 kmacy Exp $")
- CTASSERT (LOCK_CLASS_MAX==15)
- void lock_init (struct lock_object *lock, struct lock_class *class, const char *name, const char *type, int flags)
- void lock_destroy (struct lock_object *lock)

Variables

- lock_class * lock_classes [LOCK_CLASS_MAX+1]

9.97.1 Function Documentation

9.97.1.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_lock. *c*, v 1.12 2006/12/04 22:15:50 kmacy Exp $")

9.97.1.2 CTASSERT (LOCK_CLASS_MAX = =15)

9.97.1.3 void lock_destroy (struct lock_object * *lock*)

Definition at line 221 of file subr_lock.c.

Referenced by mtx_destroy(), rw_destroy(), and sx_destroy().

9.97.1.4 void lock_init (struct lock_object * *lock*, struct lock_class * *class*, const char * *name*, const char * *type*, int *flags*)

Definition at line 195 of file subr_lock.c.

References lock_classes.

Referenced by mtx_init(), rw_init(), and sx_init().

9.97.2 Variable Documentation

9.97.2.1 struct lock_class* lock_classes[LOCK_CLASS_MAX+1]

Initial value:

```
 {
        &lock_class_mtx_spin,
        &lock_class_mtx_sleep,
        &lock_class_sx,
        &lock_class_rw,
        &lock_class_lockmgr,
}
```

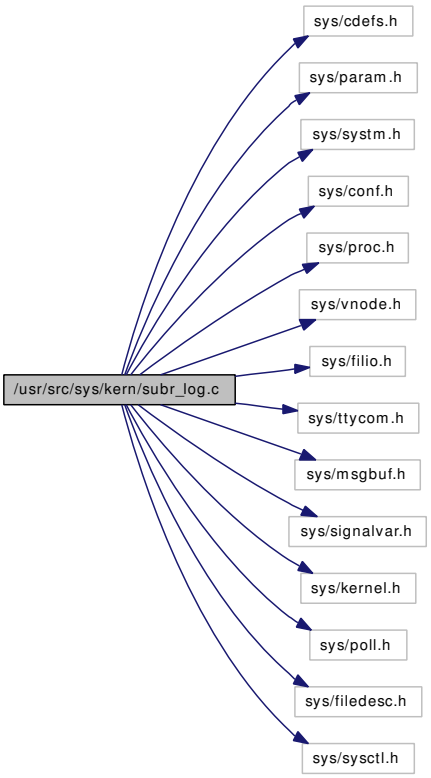
Definition at line 56 of file subr_lock.c.

Referenced by lock_init().

9.98 /usr/src/sys/kern/subr_log.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/conf.h>
#include <sys/proc.h>
#include <sys/vnode.h>
#include <sys/filio.h>
#include <sys/ttycom.h>
#include <sys/msgbuf.h>
#include <sys/signalvar.h>
#include <sys/kernel.h>
#include <sys/poll.h>
#include <sys/filedesc.h>
#include <sys/sysctl.h>
```

Include dependency graph for subr_log.c:



Data Structures

- struct logsoftc

Defines

- #define LOG_RDPRI (PZERO + 1)
- #define LOG_ASYNC 0x04
- #define LOG_RDWAIT 0x08

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_log.c,v 1.64 2005/02/27 22:01:09 phk Exp $")
- static void logtimeout (void *arg)
- SYSCTL_INT (_kern, OID_AUTO, log_wakeups_per_second, CTLFLAG_RW,&log_wakeups_per_second, 0,"")
- static int logopen (struct cdev *dev, int flags, int mode, struct thread *td)
- static int logclose (struct cdev *dev, int flag, int mode, struct thread *td)
- static int logread (struct cdev *dev, struct uio *uio, int flag)
- static int logpoll (struct cdev *dev, int events, struct thread *td)
- static int logioctl (struct cdev *dev, u_long com, caddr_t data, int flag, struct thread *td)
- static void log_drvinit (void *unused)

Variables

- static d_open_t logopen
- static d_close_t logclose
- static d_read_t logread
- static d_ioctl_t logioctl
- static d_poll_t logpoll
- static struct cdevsw log_cdevsw
- int log_open
- static int log_wakeups_per_second = 5

9.98.1 Define Documentation

9.98.1.1 #define LOG_ASYNC 0x04

Definition at line 55 of file subr_log.c.

Referenced by logioctl(), and logtimeout().

9.98.1.2 #define LOG_RDPRI (PZERO + 1)

Definition at line 53 of file subr_log.c.

Referenced by logread(), and logtimeout().

9.98.1.3 #define LOG_RDWAIT 0x08

Definition at line 56 of file subr_log.c.

Referenced by logread(), and logtimeout().

9.98.2 Function Documentation

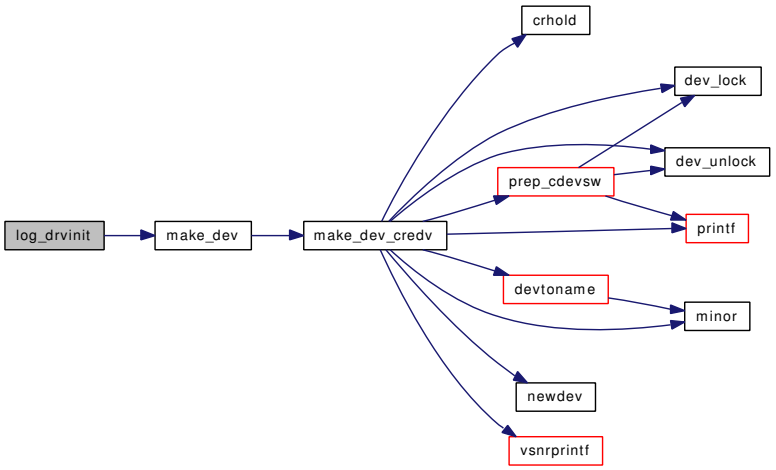
9.98.2.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_log. *c*, v 1.64 2005/02/27 22:01:09 phk Exp $")

9.98.2.2 static void log_drvinit (void * *unused*) [static]

Definition at line 247 of file subr_log.c.

References log_cdevsw, and make_dev().

Here is the call graph for this function:

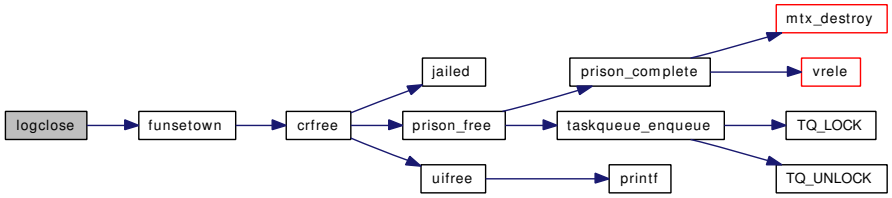


9.98.2.3 static int logclose (struct cdev * *dev*, int *flag*, int *mode*, struct thread * *td*) [static]

Definition at line 111 of file subr_log.c.

References funsetown(), and log_open.

Here is the call graph for this function:

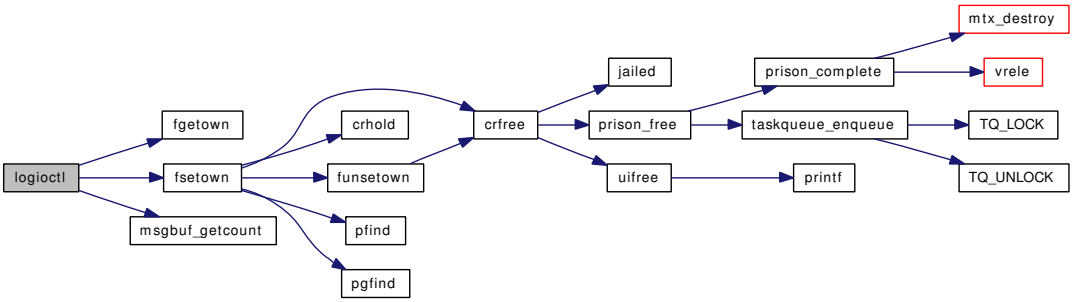


9.98.2.4 static int logioctl (struct cdev * *dev*, u_long *com*, caddr_t *data*, int *flag*, struct thread * *td*) [static]

Definition at line 204 of file subr_log.c.

References fgetown(), fsetown(), LOG_ASYNC, and msgbuf_getcount().

Here is the call graph for this function:

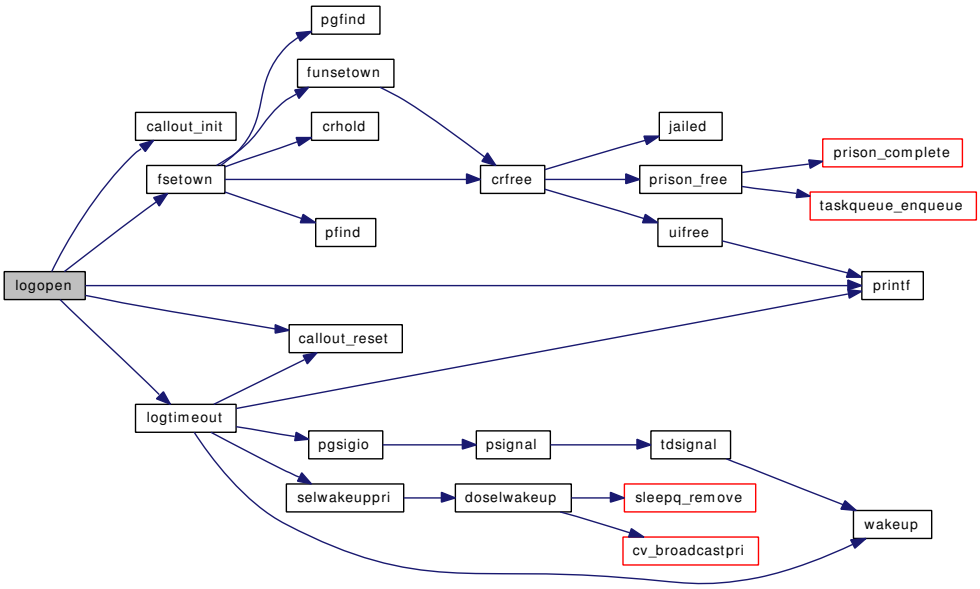


9.98.2.5 static int logopen (struct cdev * *dev*, int *flags*, int *mode*, struct thread * *td*) [static]

Definition at line 93 of file subr_log.c.

References callout_init(), callout_reset(), fsetown(), hz, log_open, log_wakeups_per_second, logtimeout(), and printf().

Here is the call graph for this function:

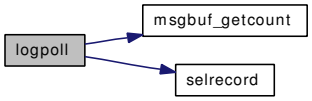


9.98.2.6 static int logpoll (struct cdev * *dev*, int *events*, struct thread * *td*) [static]

Definition at line 158 of file subr_log.c.

References msgbuf_getcount(), and selrecord().

Here is the call graph for this function:

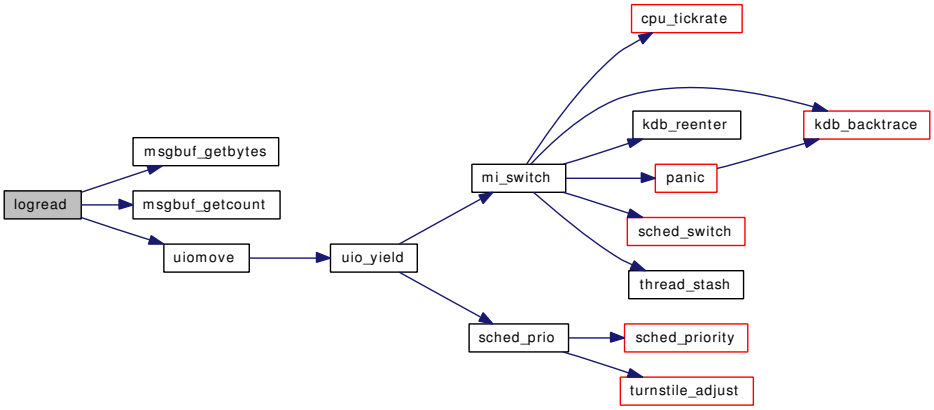


9.98.2.7 static int logread (struct cdev * *dev*, struct uio * *uio*, int *flag*) [static]

Definition at line 123 of file subr_log.c.

References buf, LOG_RDPRI, LOG_RDWAIT, msgbuf_getbytes(), msgbuf_getcount(), and uiomove().

Here is the call graph for this function:



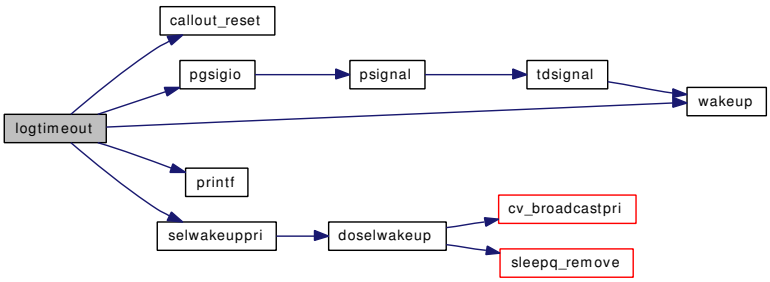
9.98.2.8 static void logtimeout (void * *arg*) [static]

Definition at line 176 of file subr_log.c.

References callout_reset(), hz, LOG_ASYNC, log_open, LOG_RDPRI, LOG_RDWAIT, log_wakeups_-per_second, msgbuftrigger, pgsigio(), printf(), selwakeuppri(), and wakeup().

Referenced by logopen().

Here is the call graph for this function:



9.98.2.9 SYSCTL_INT (_kern, OID_AUTO, log_wakeups_per_second, CTLFLAG_RW, & *log_wakeups_per_second*, 0, "")

9.98.3 Variable Documentation

9.98.3.1 struct cdevsw log_cdevsw [static]

Initial value:

```
{
        .d_version =    D_VERSION,
        .d_flags =      D_NEEDGIANT,
        .d_open =       logopen,
        .d_close =      logclose,
        .d_read =       logread,
        .d_ioctl =      logioctl,
        .d_poll =       logpoll,
        .d_name =       "log",
}
```

Definition at line 66 of file subr_log.c.

Referenced by log_drvinit().

9.98.3.2 int log_open

Definition at line 84 of file subr_log.c.

Referenced by log(), logclose(), logopen(), and logtimeout().

9.98.3.3 int log_wakeups_per_second = 5 [static]

Definition at line 87 of file subr_log.c.

Referenced by logopen(), and logtimeout().

9.98.3.4 d_close_t logclose [static]

Definition at line 59 of file subr_log.c.

9.98.3.5 d_ioctl_t logioctl [static]

Definition at line 61 of file subr_log.c.

9.98.3.6 d_open_t logopen [static]

Definition at line 58 of file subr_log.c.

9.98.3.7 d_poll_t logpoll [static]

Definition at line 62 of file subr_log.c.

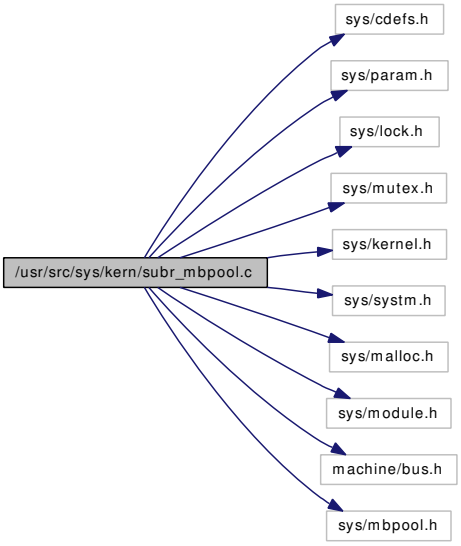
9.98.3.8 d_read_t logread [static]

Definition at line 60 of file subr_log.c.

9.99 /usr/src/sys/kern/subr_mbpool.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/kernel.h>
#include <sys/systm.h>
#include <sys/malloc.h>
#include <sys/module.h>
#include <machine/bus.h>
#include <sys/mbpool.h>
```

Include dependency graph for subr_mbpool.c:



Data Structures

- struct mbtrail
- struct mbfree
- struct mbpage
- struct mbpool

Defines

- #define MBP_CARD 0x8000
- #define MBP_USED 0x4000
- #define MBP_PMSK 0x3fff
- #define MBP_CMSK 0x01ff

- #define C2T(P, C)
- #define N2C(P, PG, C)
- #define HMAKE(P, C) ((((P) & MBP_PMSK) << 16) | ((C) << 7))
- #define HPAGE(H) (((H) >> 16) & MBP_PMSK)
- #define HCHUNK(H) (((H) >> 7) & MBP_CMSK)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_mbpool.c,v 1.3 2005/01/06 23:35:39 imp Exp $")
- MODULE_VERSION (libmbpool, 1)
- static MALLOC_DEFINE (M_MBPOOL,"mbpools","mbuf pools")
- int mbp_create (struct mbpool **pp, const char *name, bus_dma_tag_t dmat, u_int max_pages, size_t page_size, size_t chunk_size)
- void mbp_destroy (struct mbpool *p)
- static void mbp_callback (void *arg, bus_dma_segment_t *segs, int nsegs, int error)
- static void mbp_alloc_page (struct mbpool *p)
- void * mbp_alloc (struct mbpool *p, bus_addr_t *pap, uint32_t *hp)
- void mbp_free (struct mbpool *p, void *ptr)
- void mbp_ext_free (void *buf, void *arg)
- void mbp_card_free (struct mbpool *p)
- void mbp_count (struct mbpool *p, u_int *used, u_int *card, u_int *free)
- void * mbp_get (struct mbpool *p, uint32_t h)
- void * mbp_get_keep (struct mbpool *p, uint32_t h)
- void mbp_sync (struct mbpool *p, uint32_t h, bus_addr_t off, bus_size_t len, u_int op)

9.99.1 Define Documentation

9.99.1.1 #define C2T(P, C)

Value:

```
((struct mbtrail *)((char *)(C) + (P)->chunk_size - \
                            sizeof(struct mbtrail)))
```

Definition at line 98 of file subr_mbpool.c.

Referenced by mbp_alloc(), mbp_alloc_page(), mbp_card_free(), mbp_count(), mbp_destroy(), mbp_free(), mbp_get(), and mbp_get_keep().

9.99.1.2 #define HCHUNK(H) (((H) >> 7) & MBP_CMSK)

Definition at line 112 of file subr_mbpool.c.

Referenced by mbp_get(), mbp_get_keep(), and mbp_sync().

9.99.1.3 #define HMAKE(P, C) ((((P) & MBP_PMSK) << 16) | ((C) << 7))

Definition at line 110 of file subr_mbpool.c.

Referenced by mbp_alloc().

9.99.1.4 #define HPAGE(H) (((H) >> 16) & MBP_PMSK)

Definition at line 111 of file subr_mbpool.c.

Referenced by mbp_get(), mbp_get_keep(), and mbp_sync().

9.99.1.5 #define MBP_CARD 0x8000

Definition at line 64 of file subr_mbpool.c.

Referenced by mbp_alloc(), mbp_card_free(), mbp_count(), mbp_destroy(), mbp_free(), mbp_get(), and mbp_get_keep().

9.99.1.6 #define MBP_CMSK 0x01ff

Definition at line 67 of file subr_mbpool.c.

9.99.1.7 #define MBP_PMSK 0x3fff

Definition at line 66 of file subr_mbpool.c.

Referenced by mbp_card_free().

9.99.1.8 #define MBP_USED 0x4000

Definition at line 65 of file subr_mbpool.c.

Referenced by mbp_alloc(), mbp_count(), mbp_destroy(), and mbp_free().

9.99.1.9 #define N2C(P, PG, C)

Value:

```
((struct mbfree *)((char *)(PG)->va + \
                           (C) * (P)->chunk_size))
```

Definition at line 104 of file subr_mbpool.c.

Referenced by mbp_alloc_page(), mbp_card_free(), mbp_count(), mbp_destroy(), mbp_get(), and mbp_get_keep().

9.99.2 Function Documentation

9.99.2.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_mbpool. *c*, v 1.3 2005/01/06 23:35:39 imp Exp $")

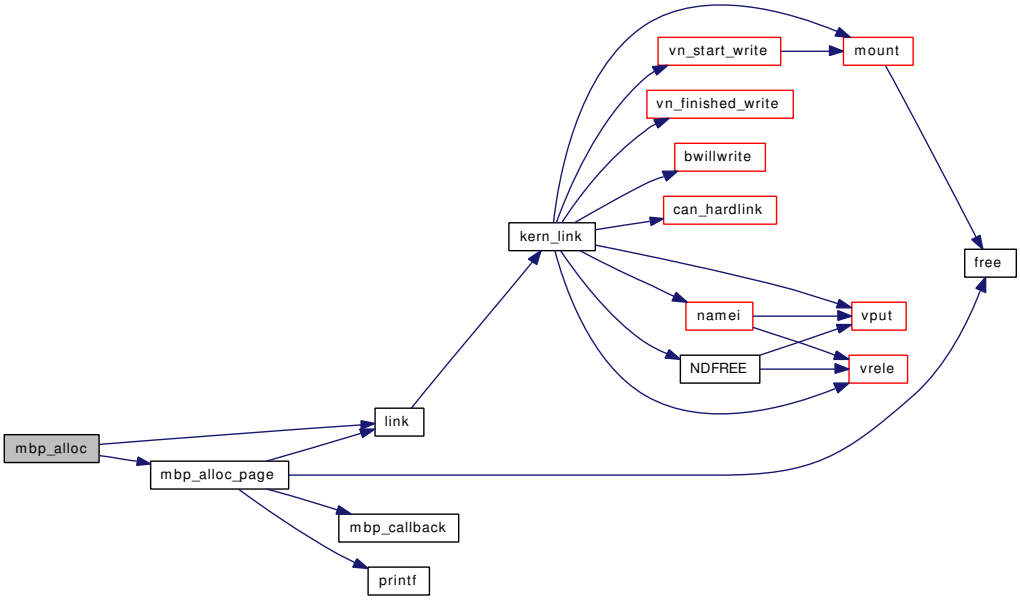
9.99.2.2 static MALLOC_DEFINE (M_MBPOOL, "mbpools", "mbuf pools") [static]

9.99.2.3 void* mbp_alloc (struct mbpool * *p*, bus_addr_t * *pap*, uint32_t * *hp*)

Definition at line 240 of file subr_mbpool.c.

References C2T, mbtrail::chunk, mbpool::chunk_size, mbpool::free_lock, HMAKE, link(), mbp_alloc_-page(), MBP_CARD, MBP_USED, and mbtrail::page.

Here is the call graph for this function:



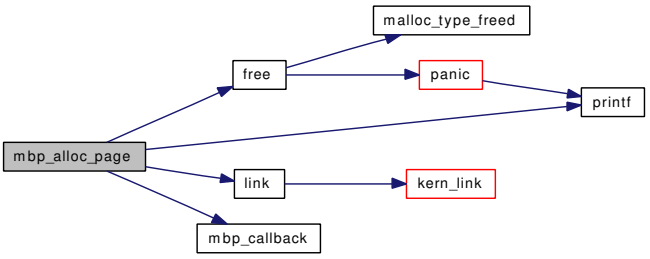
9.99.2.4 static void mbp_alloc_page (struct mbpool * *p*) [static]

Definition at line 194 of file subr_mbpool.c.

References C2T, mbtrail::chunk, mbpool::dmat, free(), link(), mbpage::map, mbpool::max_pages, mbp_-callback(), N2C, mbpool::name, mbtrail::page, mbpool::page_size, mbpage::phy, printf(), and mbpage::va.

Referenced by mbp_alloc().

Here is the call graph for this function:



9.99.2.5 static void mbp_callback (void * *arg*, bus_dma_segment_t * *segs*, int *nsegs*, int *error*) [static]

Definition at line 184 of file subr_mbpool.c.

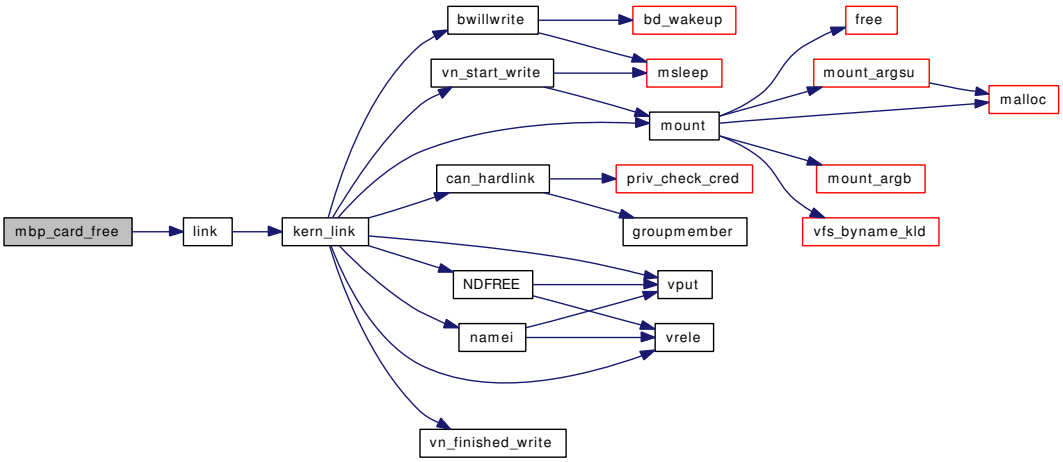
Referenced by mbp_alloc_page().

9.99.2.6 void mbp_card_free (struct mbpool * *p*)

Definition at line 295 of file subr_mbpool.c.

References C2T, mbpool::free_lock, link(), MBP_CARD, MBP_PMSK, N2C, and mbtrail::page.

Here is the call graph for this function:

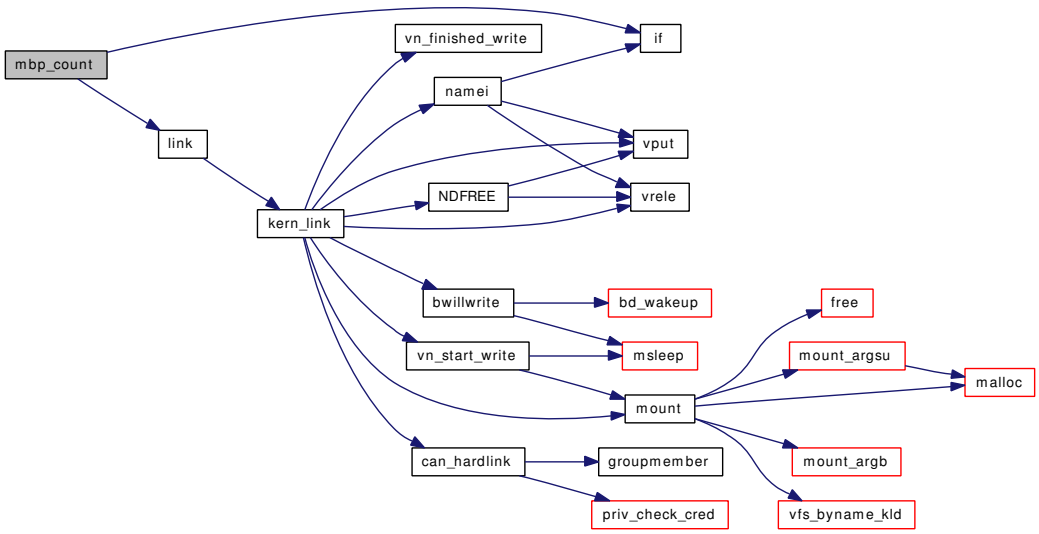


9.99.2.7 void mbp_count (struct mbpool * *p*, u_int * *used*, u_int * *card*, u_int * *free*)

Definition at line 321 of file subr_mbpool.c.

References C2T, mbpool::free_lock, if(), link(), MBP_CARD, MBP_USED, N2C, and mbtrail::page.

Here is the call graph for this function:

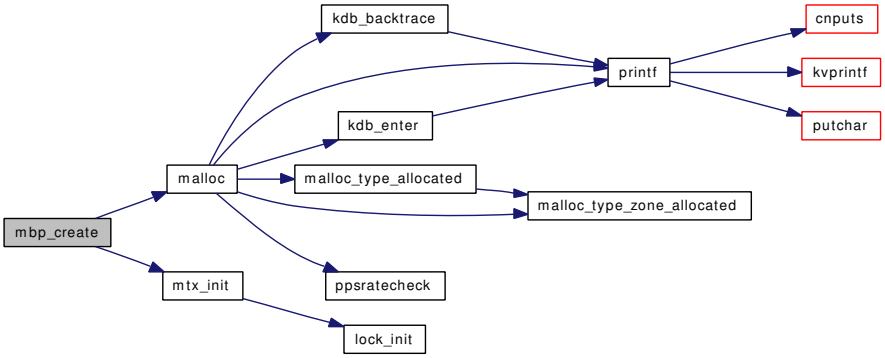


9.99.2.8 int mbp_create (struct mbpool ** *pp*, const char * *name*, bus_dma_tag_t *dmat*, u_int *max_pages*, size_t *page_size*, size_t *chunk_size*)

Definition at line 118 of file subr_mbpool.c.

References malloc(), and mtx_init().

Here is the call graph for this function:

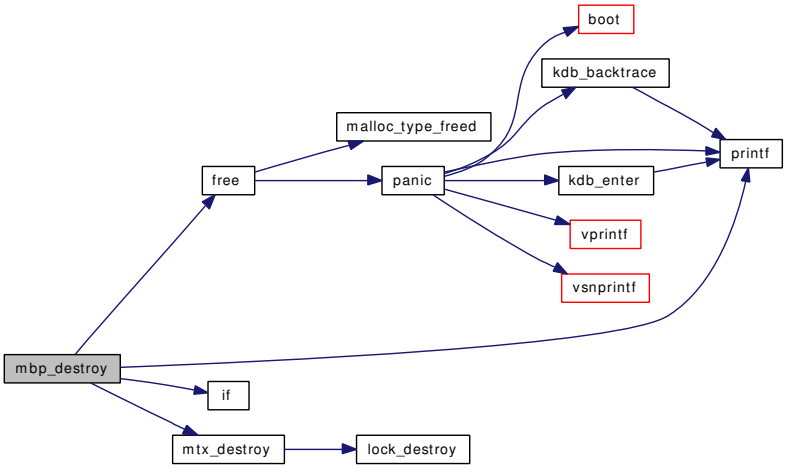


9.99.2.9 void mbp_destroy (struct mbpool * *p*)

Definition at line 150 of file subr_mbpool.c.

References C2T, mbpool::dmat, free(), mbpool::free_lock, if(), mbpage::map, MBP_CARD, MBP_USED, mtx_destroy(), N2C, mbpool::name, mbtrail::page, printf(), and mbpage::va.

Here is the call graph for this function:

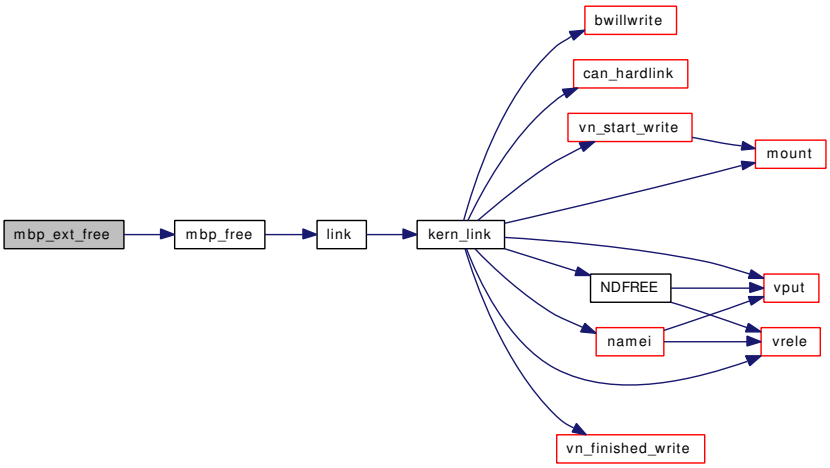


9.99.2.10 void mbp_ext_free (void * *buf*, void * *arg*)

Definition at line 286 of file subr_mbpool.c.

References mbp_free().

Here is the call graph for this function:



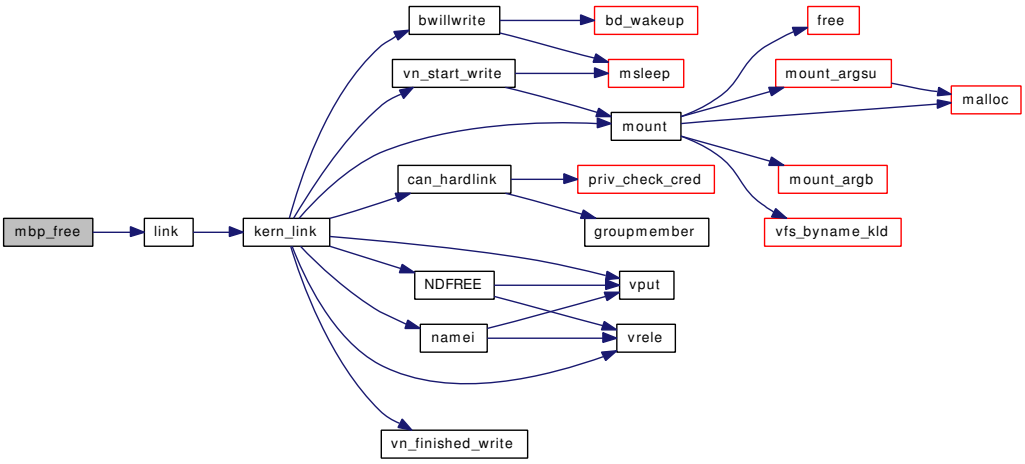
9.99.2.11 void mbp_free (struct mbpool * *p*, void * *ptr*)

Definition at line 271 of file subr_mbpool.c.

References C2T, mbpool::free_lock, link(), MBP_CARD, MBP_USED, and mbtrail::page.

Referenced by mbp_ext_free().

Here is the call graph for this function:

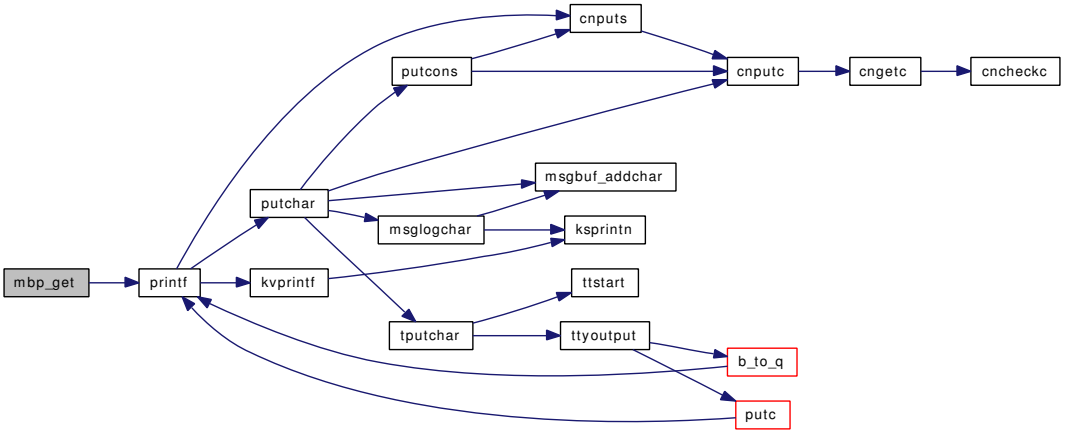


9.99.2.12 void* mbp_get (struct mbpool * *p*, uint32_t *h*)

Definition at line 349 of file subr_mbpool.c.

References C2T, HCHUNK, HPAGE, MBP_CARD, N2C, mbpool::name, mbtrail::page, and printf().

Here is the call graph for this function:

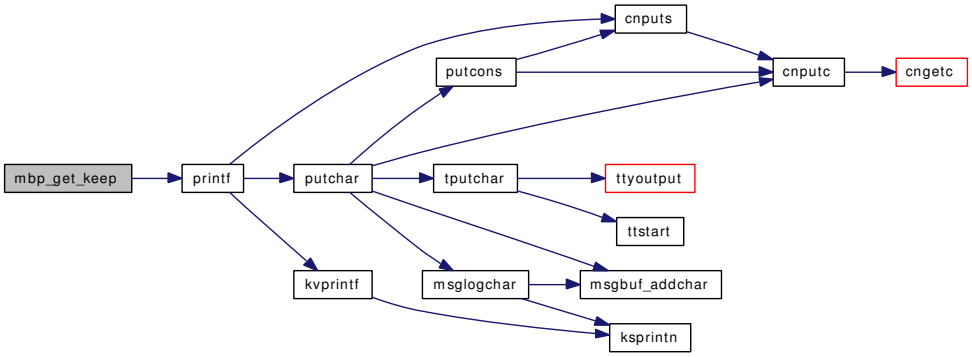


9.99.2.13 void* mbp_get_keep (struct mbpool * *p*, uint32_t *h*)

Definition at line 371 of file subr_mbpool.c.

References C2T, HCHUNK, HPAGE, MBP_CARD, N2C, mbpool::name, mbtrail::page, and printf().

Here is the call graph for this function:



9.99.2.14 void mbp_sync (struct mbpool * *p*, uint32_t *h*, bus_addr_t *off*, bus_size_t *len*, u_int *op*)

Definition at line 392 of file subr_mbpool.c.

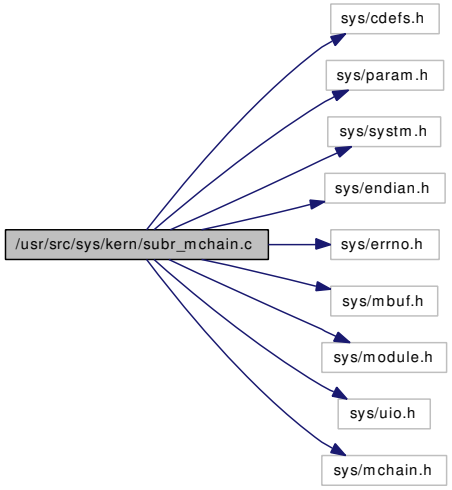
References mbpool::chunk_size, mbpool::dmat, HCHUNK, and HPAGE.

9.99.2.15 MODULE_VERSION (libmbpool, 1)

9.100 /usr/src/sys/kern/subr_mchain.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/endian.h>
#include <sys/errno.h>
#include <sys/mbuf.h>
#include <sys/module.h>
#include <sys/uio.h>
#include <sys/mchain.h>
```

Include dependency graph for subr_mchain.c:



Defines

- #define MBERROR(format,)
- #define MBPANIC(format,)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_mchain.c,v 1.18 2005/07/29 13:22:36 imura Exp $")
- MODULE_VERSION (libmchain, 1)
- int mb_init (struct mbchain *mbp)
- void mb_initm (struct mbchain *mbp, struct mbuf *m)
- void mb_done (struct mbchain *mbp)
- mbuf * mb_detach (struct mbchain *mbp)
- int mb_fixhdr (struct mbchain *mbp)
- caddr_t mb_reserve (struct mbchain *mbp, int size)
- int mb_put_uint8 (struct mbchain *mbp, u_int8_t x)

- int mb_put_uint16be (struct mbchain *mbp, u_int16_t x)
- int mb_put_uint16le (struct mbchain *mbp, u_int16_t x)
- int mb_put_uint32be (struct mbchain *mbp, u_int32_t x)
- int mb_put_uint32le (struct mbchain *mbp, u_int32_t x)
- int mb_put_int64be (struct mbchain *mbp, int64_t x)
- int mb_put_int64le (struct mbchain *mbp, int64_t x)
- int mb_put_mem (struct mbchain *mbp, c_caddr_t source, int size, int type)
- int mb_put_mbuf (struct mbchain *mbp, struct mbuf *m)
- int mb_put_uio (struct mbchain *mbp, struct uio *uiop, int size)
- int md_init (struct mdchain *mdp)
- void md_initm (struct mdchain *mdp, struct mbuf *m)
- void md_done (struct mdchain *mdp)
- void md_append_record (struct mdchain *mdp, struct mbuf *top)
- int md_next_record (struct mdchain *mdp)
- int md_get_uint8 (struct mdchain *mdp, u_int8_t *x)
- int md_get_uint16 (struct mdchain *mdp, u_int16_t *x)
- int md_get_uint16le (struct mdchain *mdp, u_int16_t *x)
- int md_get_uint16be (struct mdchain *mdp, u_int16_t *x)
- int md_get_uint32 (struct mdchain *mdp, u_int32_t *x)
- int md_get_uint32be (struct mdchain *mdp, u_int32_t *x)
- int md_get_uint32le (struct mdchain *mdp, u_int32_t *x)
- int md_get_int64 (struct mdchain *mdp, int64_t *x)
- int md_get_int64be (struct mdchain *mdp, int64_t *x)
- int md_get_int64le (struct mdchain *mdp, int64_t *x)
- int md_get_mem (struct mdchain *mdp, caddr_t target, int size, int type)
- int md_get_mbuf (struct mdchain *mdp, int size, struct mbuf **ret)
- int md_get_uio (struct mdchain *mdp, struct uio *uiop, int size)

9.100.1 Define Documentation

9.100.1.1 #define MBERROR(format)

Value:

```
printf("%s(%d): "format, __func__ , \
                                    __LINE__ , ## __VA_ARGS__)
```

Definition at line 45 of file subr_mchain.c.

Referenced by md_get_mem().

9.100.1.2 #define MBPANIC(format)

Value:

```
printf("%s(%d): "format, __func__ , \
                                    __LINE__ , ## __VA_ARGS__)
```

Definition at line 48 of file subr_mchain.c.

9.100.2 Function Documentation

9.100.2.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_mchain. *c*, v 1.18 2005/07/29 13:22:36 imura Exp $")

9.100.2.2 struct mbuf* mb_detach (struct mbchain * *mbp*)

Definition at line 85 of file subr_mchain.c.

9.100.2.3 void mb_done (struct mbchain * *mbp*)

Definition at line 76 of file subr_mchain.c.

References m_freem().

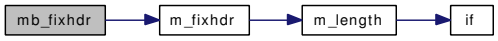
Here is the call graph for this function:

9.100.2.4 int mb_fixhdr (struct mbchain * *mbp*)

Definition at line 95 of file subr_mchain.c.

References m_fixhdr().

Here is the call graph for this function:



9.100.2.5 int mb_init (struct mbchain * *mbp*)

Definition at line 55 of file subr_mchain.c.

References mb_initm().

Here is the call graph for this function:

9.100.2.6 void mb_initm (struct mbchain * *mbp*, struct mbuf * *m*)

Definition at line 68 of file subr_mchain.c.

Referenced by mb_init().

9.100.2.7 int mb_put_int64be (struct mbchain * *mbp*, int64_t *x*)

Definition at line 166 of file subr_mchain.c.

References mb_put_mem().

Here is the call graph for this function:

9.100.2.8 int mb_put_int64le (struct mbchain * *mbp*, int64_t *x*)

Definition at line 173 of file subr_mchain.c.

References mb_put_mem().

Here is the call graph for this function:

9.100.2.9 int mb_put_mbuf (struct mbchain * *mbp*, struct mbuf * *m*)

Definition at line 241 of file subr_mchain.c.

9.100.2.10 int mb_put_mem (struct mbchain * *mbp*, c_caddr_t *source*, int *size*, int *type*)

Definition at line 180 of file subr_mchain.c.

Referenced by mb_put_int64be(), mb_put_int64le(), mb_put_uint16be(), mb_put_uint16le(), mb_put_-uint32be(), mb_put_uint32le(), mb_put_uint8(), and mb_put_uio().

9.100.2.11 int mb_put_uint16be (struct mbchain * *mbp*, u_int16_t *x*)

Definition at line 138 of file subr_mchain.c.

References mb_put_mem().

Here is the call graph for this function:

9.100.2.12 int mb_put_uint16le (struct mbchain * *mbp*, u_int16_t *x*)

Definition at line 145 of file subr_mchain.c.

References mb_put_mem().

Here is the call graph for this function:

9.100.2.13 int mb_put_uint32be (struct mbchain * *mbp*, u_int32_t *x*)

Definition at line 152 of file subr_mchain.c.

References mb_put_mem().

Here is the call graph for this function:

9.100.2.14 int mb_put_uint32le (struct mbchain * *mbp*, u_int32_t *x*)

Definition at line 159 of file subr_mchain.c.

References mb_put_mem().

Here is the call graph for this function:

9.100.2.15 int mb_put_uint8 (struct mbchain * *mbp*, u_int8_t *x*)

Definition at line 132 of file subr_mchain.c.

References mb_put_mem().

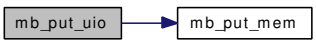
Here is the call graph for this function:

9.100.2.16 int mb_put_uio (struct mbchain * *mbp*, struct uio * *uiop*, int *size*)

Definition at line 259 of file subr_mchain.c.

References mb_put_mem().

Here is the call graph for this function:

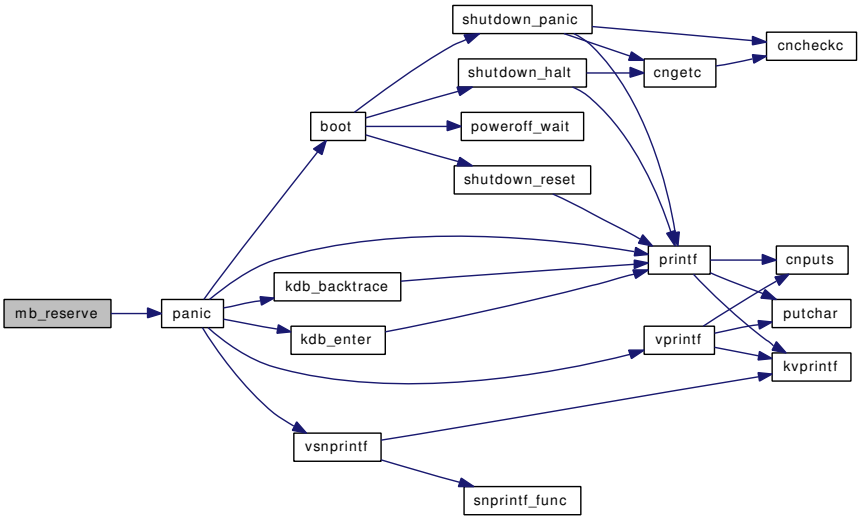


9.100.2.17 caddr_t mb_reserve (struct mbchain * *mbp*, int *size*)

Definition at line 107 of file subr_mchain.c.

References panic().

Here is the call graph for this function:



9.100.2.18 void md_append_record (struct mdchain * *mdp*, struct mbuf * *top*)

Definition at line 328 of file subr_mchain.c.

References md_initm().

Here is the call graph for this function:

9.100.2.19 void md_done (struct mdchain * *mdp*)

Definition at line 315 of file subr_mchain.c.

References m_freem().

Referenced by md_next_record().

Here is the call graph for this function:

9.100.2.20 int md_get_int64 (struct mdchain * *mdp*, int64_t * *x*)

Definition at line 426 of file subr_mchain.c.

References md_get_mem().

Referenced by md_get_int64be(), and md_get_int64le().

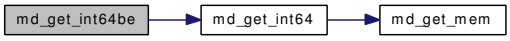
Here is the call graph for this function:

9.100.2.21 int md_get_int64be (struct mdchain * *mdp*, int64_t * *x*)

Definition at line 432 of file subr_mchain.c.

References md_get_int64().

Here is the call graph for this function:

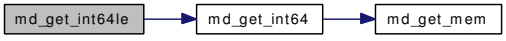


9.100.2.22 int md_get_int64le (struct mdchain * *mdp*, int64_t * *x*)

Definition at line 444 of file subr_mchain.c.

References md_get_int64().

Here is the call graph for this function:

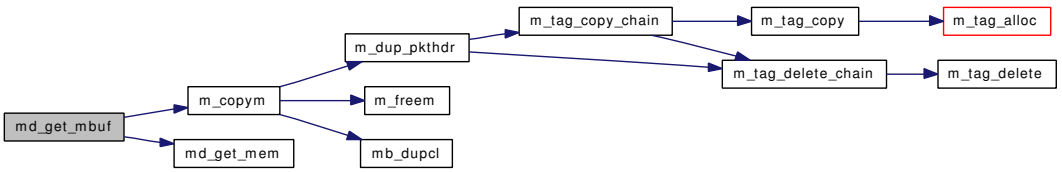


9.100.2.23 int md_get_mbuf (struct mdchain * *mdp*, int *size*, struct mbuf ** *ret*)

Definition at line 502 of file subr_mchain.c.

References m_copym(), and md_get_mem().

Here is the call graph for this function:



9.100.2.24 int md_get_mem (struct mdchain * *mdp*, caddr_t *target*, int *size*, int *type*)

Definition at line 456 of file subr_mchain.c.

References MBERROR.

Referenced by md_get_int64(), md_get_mbuf(), md_get_uint16(), md_get_uint32(), md_get_uint8(), and md_get_uio().

9.100.2.25 int md_get_uint16 (struct mdchain * *mdp*, u_int16_t * *x*)

Definition at line 369 of file subr_mchain.c.

References md_get_mem().

Referenced by md_get_uint16be(), and md_get_uint16le().

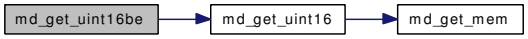
Here is the call graph for this function:

9.100.2.26 int md_get_uint16be (struct mdchain * *mdp*, u_int16_t * *x*)

Definition at line 386 of file subr_mchain.c.

References md_get_uint16().

Here is the call graph for this function:

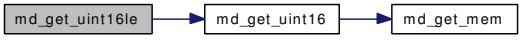


9.100.2.27 int md_get_uint16le (struct mdchain * *mdp*, u_int16_t * *x*)

Definition at line 375 of file subr_mchain.c.

References md_get_uint16().

Here is the call graph for this function:



9.100.2.28 int md_get_uint32 (struct mdchain * *mdp*, u_int32_t * *x*)

Definition at line 396 of file subr_mchain.c.

References md_get_mem().

Referenced by md_get_uint32be(), and md_get_uint32le().

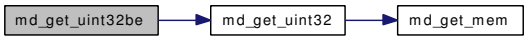
Here is the call graph for this function:

9.100.2.29 int md_get_uint32be (struct mdchain * *mdp*, u_int32_t * *x*)

Definition at line 402 of file subr_mchain.c.

References md_get_uint32().

Here is the call graph for this function:

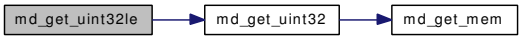


9.100.2.30 int md_get_uint32le (struct mdchain * *mdp*, u_int32_t * *x*)

Definition at line 414 of file subr_mchain.c.

References md_get_uint32().

Here is the call graph for this function:



9.100.2.31 int md_get_uint8 (struct mdchain * *mdp*, u_int8_t * *x*)

Definition at line 363 of file subr_mchain.c.

References md_get_mem().

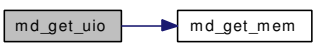
Here is the call graph for this function:

9.100.2.32 int md_get_uio (struct mdchain * *mdp*, struct uio * *uiop*, int *size*)

Definition at line 515 of file subr_mchain.c.

References md_get_mem().

Here is the call graph for this function:



9.100.2.33 int md_init (struct mdchain * *mdp*)

Definition at line 294 of file subr_mchain.c.

References md_initm().

Here is the call graph for this function:

9.100.2.34 void md_initm (struct mdchain * *mdp*, struct mbuf * *m*)

Definition at line 307 of file subr_mchain.c.

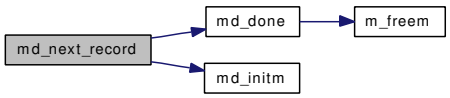
Referenced by md_append_record(), md_init(), and md_next_record().

9.100.2.35 int md_next_record (struct mdchain * *mdp*)

Definition at line 348 of file subr_mchain.c.

References md_done(), and md_initm().

Here is the call graph for this function:

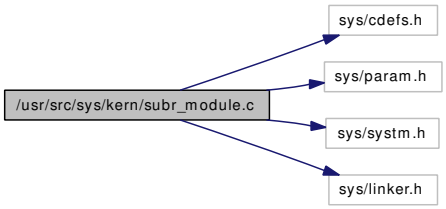


9.100.2.36 MODULE_VERSION (libmchain, 1)

9.101 /usr/src/sys/kern/subr_module.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/linker.h>
```

Include dependency graph for subr_module.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_module.c,v 1.8 2003/06/11 00:56:57 obrien Exp $")
- caddr_t preload_search_by_name (const char *name)
- caddr_t preload_search_by_type (const char *type)
- caddr_t preload_search_next_name (caddr_t base)
- caddr_t preload_search_info (caddr_t mod, int inf)
- void preload_delete_name (const char *name)
- void preload_bootstrap_relocate (vm_offset_t offset)

Variables

- caddr_t preload_metadata

9.101.1 Function Documentation

9.101.1.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_module. *c*, v 1.8 2003/06/11 00:56:57 obrien Exp $")

9.101.1.2 void preload_bootstrap_relocate (vm_offset_t *offset*)

Definition at line 235 of file subr_module.c.

References preload_metadata.

9.101.1.3 void preload_delete_name (const char * *name*)

Definition at line 199 of file subr_module.c.

References preload_metadata.

Referenced by link_elf_unload_file(), and link_elf_unload_preload().

9.101.1.4 caddr_t preload_search_by_name (const char * *name*)

Definition at line 44 of file subr_module.c.

References preload_metadata.

Referenced by link_elf_link_preload().

9.101.1.5 caddr_t preload_search_by_type (const char * *type*)

Definition at line 76 of file subr_module.c.

References preload_metadata.

Referenced by link_elf_init().

9.101.1.6 caddr_t preload_search_info (caddr_t *mod*, int *inf*)

Definition at line 155 of file subr_module.c.

Referenced by link_elf_init(), link_elf_link_preload(), link_elf_preload_parse_symbols(), and linker_preload().

9.101.1.7 caddr_t preload_search_next_name (caddr_t *base*)

Definition at line 113 of file subr_module.c.

References preload_metadata.

Referenced by linker_preload().

9.101.2 Variable Documentation

9.101.2.1 caddr_t preload_metadata

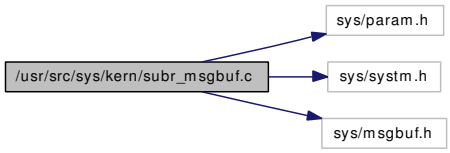
Definition at line 38 of file subr_module.c.

Referenced by preload_bootstrap_relocate(), preload_delete_name(), preload_search_by_name(), preload_search_by_type(), and preload_search_next_name().

9.102 /usr/src/sys/kern/subr_msgbuf.c File Reference

```
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/msgbuf.h>
```

Include dependency graph for subr_msgbuf.c:



Defines

- #define SEQMOD(size) ((size) ∗ 16)

Functions

- static u_int msgbuf_cksum (struct msgbuf ∗mbp)
- void msgbuf_init (struct msgbuf ∗mbp, void ∗ptr, int size)
- void msgbuf_reinit (struct msgbuf ∗mbp, void ∗ptr, int size)
- void msgbuf_clear (struct msgbuf ∗mbp)
- int msgbuf_getcount (struct msgbuf ∗mbp)
- void msgbuf_addchar (struct msgbuf ∗mbp, int c)
- int msgbuf_getchar (struct msgbuf ∗mbp)
- int msgbuf_getbytes (struct msgbuf ∗mbp, char ∗buf, int buflen)
- int msgbuf_peekbytes (struct msgbuf ∗mbp, char ∗buf, int buflen, u_int ∗seqp)
- void msgbuf_copy (struct msgbuf ∗src, struct msgbuf ∗dst)

9.102.1 Define Documentation

9.102.1.1 #define SEQMOD(size) ((size) ∗ 16)

Definition at line 37 of file subr_msgbuf.c.

Referenced by msgbuf_init(), and msgbuf_reinit().

9.102.2 Function Documentation

9.102.2.1 void msgbuf_addchar (struct msgbuf ∗ *mbp*, int *c*)

Definition at line 120 of file subr_msgbuf.c.

Referenced by msgbuf_copy(), msglogchar(), and putchar().

9.102.2.2 static u_int msgbuf_cksum (struct msgbuf * *mbp*) [static]

Definition at line 222 of file subr_msgbuf.c.

Referenced by msgbuf_reinit().

9.102.2.3 void msgbuf_clear (struct msgbuf * *mbp*)

Definition at line 89 of file subr_msgbuf.c.

Referenced by msgbuf_init(), msgbuf_reinit(), and sysctl_kern_msgbuf_clear().

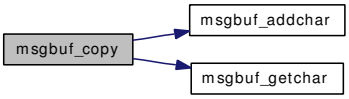
9.102.2.4 void msgbuf_copy (struct msgbuf * *src*, struct msgbuf * *dst*)

Definition at line 236 of file subr_msgbuf.c.

References msgbuf_addchar(), and msgbuf_getchar().

Referenced by msgbufinit().

Here is the call graph for this function:



9.102.2.5 int msgbuf_getbytes (struct msgbuf * *mbp*, char * *buf*, int *buflen*)

Definition at line 160 of file subr_msgbuf.c.

Referenced by logread().

9.102.2.6 int msgbuf_getchar (struct msgbuf * *mbp*)

Definition at line 139 of file subr_msgbuf.c.

Referenced by constty_clear(), constty_timeout(), and msgbuf_copy().

9.102.2.7 int msgbuf_getcount (struct msgbuf * *mbp*)

Definition at line 102 of file subr_msgbuf.c.

Referenced by logioctl(), logpoll(), and logread().

9.102.2.8 void msgbuf_init (struct msgbuf * *mbp*, void * *ptr*, int *size*)

Definition at line 46 of file subr_msgbuf.c.

References msgbuf_clear(), and SEQMOD.

Referenced by constty_set(), and msgbuf_reinit().

Here is the call graph for this function:

9.102.2.9 int msgbuf_peekbytes (struct msgbuf * *mbp*, char * *buf*, int *buflen*, u_int * *seqp*)

Definition at line 192 of file subr_msgbuf.c.

Referenced by sysctl_kern_msgbuf().

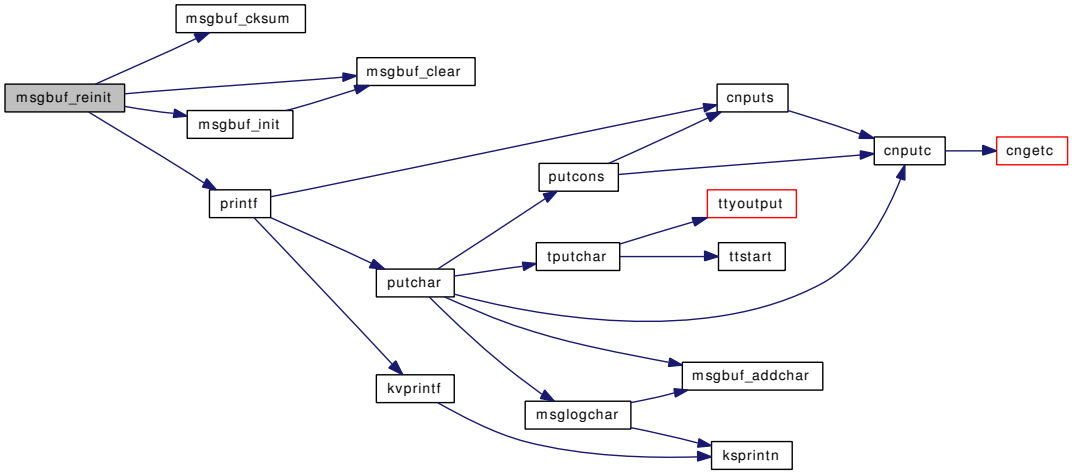
9.102.2.10 void msgbuf_reinit (struct msgbuf * *mbp*, void * *ptr*, int *size*)

Definition at line 62 of file subr_msgbuf.c.

References bootverbose, msgbuf_cksum(), msgbuf_clear(), msgbuf_init(), printf(), and SEQMOD.

Referenced by msgbufinit().

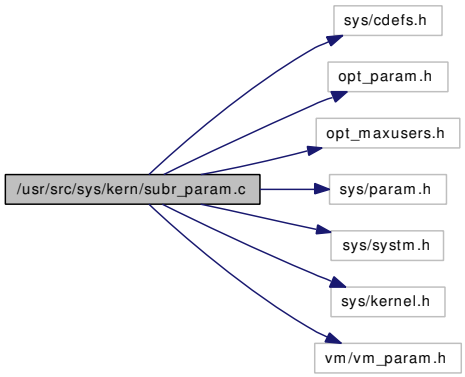
Here is the call graph for this function:



9.103 /usr/src/sys/kern/subr_param.c File Reference

```
#include <sys/cdefs.h>
#include "opt_param.h"
#include "opt_maxusers.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <vm/vm_param.h>
```

Include dependency graph for subr_param.c:



Defines

- #define HZ 100
- #define NPROC (20 + 16 ∗ maxusers)
- #define NBUF 0
- #define MAXFILES (maxproc ∗ 2)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_param.c,v 1.73 2005/10/16 03:58:10 kris Exp $")
- void init_param1 (void)
- void init_param2 (long physpages)
- void init_param3 (long kmempages)

Variables

- int hz
- int tick
- int maxusers
- int maxproc
- int maxprocperuid
- int maxfiles

- int maxfilesperproc
- int ncallout
- int nbuf
- int nswbuf
- int maxswzone
- int maxbcache
- int maxpipekva
- u_long maxtsiz
- u_long dfldsiz
- u_long maxdsiz
- u_long dflssiz
- u_long maxssiz
- u_long sgrowsiz
- buf ∗ swbuf

9.103.1 Define Documentation

9.103.1.1 #define HZ 100

Definition at line 57 of file subr_param.c.

Referenced by init_param1().

9.103.1.2 #define MAXFILES (maxproc ∗ 2)

Definition at line 65 of file subr_param.c.

Referenced by init_param2().

9.103.1.3 #define NBUF 0

Definition at line 62 of file subr_param.c.

Referenced by init_param2().

9.103.1.4 #define NPROC (20 + 16 ∗ maxusers)

Definition at line 60 of file subr_param.c.

Referenced by init_param2().

9.103.2 Function Documentation

9.103.2.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_param. *c*, v 1.73 2005/10/16 03:58:10 kris Exp $")

9.103.2.2 void init_param1 (void)

Definition at line 99 of file subr_param.c.

References dfldsiz, dflssiz, HZ, hz, maxbcache, maxdsiz, maxssiz, maxswzone, maxtsiz, sgrowsiz, and tick.

9.103.2.3 void init_param2 (long *physpages*)

Definition at line 133 of file subr_param.c.

References MAXFILES, maxfiles, maxfilesperproc, maxproc, maxprocperuid, maxusers, NBUF, nbuf, ncallout, and NPROC.

9.103.2.4 void init_param3 (long *kmempages*)

Definition at line 178 of file subr_param.c.

References maxpipekva.

Referenced by kmeminit().

9.103.3 Variable Documentation

9.103.3.1 u_long dfldsiz

Definition at line 82 of file subr_param.c.

Referenced by init_param1().

9.103.3.2 u_long dflssiz

Definition at line 84 of file subr_param.c.

Referenced by init_param1().

9.103.3.3 int hz

Definition at line 68 of file subr_param.c.

Referenced by acct_process(), acct_thread(), aio_proc_rundown(), biowait(), buf_daemon(), constty_timeout(), destroy_devl(), dev_strategy(), fork1(), getnewvnode(), hardclock_device_poll(), init_param1(), initclocks(), inittimecounter(), kern_clock_getres(), kern_select(), kern_sigtimedwait(), kern_thr_suspend(), kproc_shutdown(), kqueue_scan(), lboltcb(), loadav(), logopen(), logtimeout(), netisr_pollmore(), pffasttimo(), pfslowtimo(), poll(), poll_idle(), ppsratecheck(), reg_frac_sysctl(), root_mount_wait(), sched_initticks(), sched_pctcpu(), sched_setup(), sched_sync(), schedcpu(), schedcpu_thread(), sleepinit(), soclose(), sogetopt(), sosetopt(), statclock(), sysctl_kern_clockrate(), sysctl_msec_to_ticks(), tc_init(), ttioctl(), tty_open(), ttyalloc(), ttycheckoutq(), ttyflush(), tvtohz(), vfs_mount_destroy(), and vnlru_proc().

9.103.3.4 int maxbcache

Definition at line 79 of file subr_param.c.

Referenced by init_param1(), and kern_vfs_bio_buffer_alloc().

9.103.3.5 u_long maxdsiz

Definition at line 83 of file subr_param.c.

Referenced by init_param1(), and kern_setrlimit().

9.103.3.6 int maxfiles

Definition at line 73 of file subr_param.c.

Referenced by falloc(), init_maxsockets(), init_param2(), proc0_init(), and sysctl_maxsockets().

9.103.3.7 int maxfilesperproc

Definition at line 74 of file subr_param.c.

Referenced by do_dup(), fdalloc(), fdavail(), getdtablesize(), init_param2(), kern_fcntl(), kern_setrlimit(), and sysctl_maxsockets().

9.103.3.8 int maxpipekva

Definition at line 80 of file subr_param.c.

Referenced by init_param3(), pipe_create(), pipe_read(), and pipe_write().

9.103.3.9 int maxproc

Definition at line 71 of file subr_param.c.

Referenced by fork1(), init_param2(), proc0_init(), procinit(), uihashinit(), and vntblinit().

9.103.3.10 int maxprocperuid

Definition at line 72 of file subr_param.c.

Referenced by init_param2(), and kern_setrlimit().

9.103.3.11 u_long maxssiz

Definition at line 85 of file subr_param.c.

Referenced by exec_new_vmspace(), init_param1(), and kern_setrlimit().

9.103.3.12 int maxswzone

Definition at line 78 of file subr_param.c.

Referenced by init_param1().

9.103.3.13 u_long maxtsiz

Definition at line 81 of file subr_param.c.

Referenced by do_aout_hdr(), exec_aout_imgact(), and init_param1().

9.103.3.14 int maxusers

Definition at line 70 of file subr_param.c.

Referenced by init_param2(), and tunable_mbinit().

9.103.3.15 int nbuf

Definition at line 76 of file subr_param.c.

Referenced by bufinit(), cluster_read(), init_param2(), kern_vfs_bio_buffer_alloc(), kvprintf(), and msglogchar().

9.103.3.16 int ncallout

Definition at line 75 of file subr_param.c.

Referenced by init_param2(), kern_timeout_callwheel_alloc(), and kern_timeout_callwheel_init().

9.103.3.17 int nswbuf

Definition at line 77 of file subr_param.c.

Referenced by kern_vfs_bio_buffer_alloc().

9.103.3.18 u_long sgrowsiz

Definition at line 86 of file subr_param.c.

Referenced by exec_new_vmspace(), and init_param1().

9.103.3.19 struct buf* swbuf

Definition at line 93 of file subr_param.c.

Referenced by kern_vfs_bio_buffer_alloc().

9.103.3.20 int tick

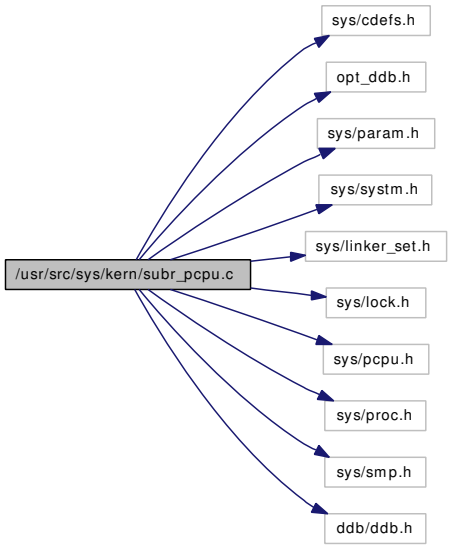
Definition at line 69 of file subr_param.c.

Referenced by acct_process(), hardclock_cpu(), init_param1(), itimerfix(), itimespecfix(), kse_create(), sched_switch(), sched_tick(), sogetopt(), sosetopt(), sysctl_kern_clockrate(), sysctl_kern_quantum(), and tvtohz().

9.104 /usr/src/sys/kern/subr_pcpu.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ddb.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/linker_set.h>
#include <sys/lock.h>
#include <sys/pcpu.h>
#include <sys/proc.h>
#include <sys/smp.h>
#include <ddb/ddb.h>
```

Include dependency graph for subr_pcpu.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_pcpu.c,v 1.8 2005/11/03 21:06:29 jhb Exp $")
- void pcpu_init (struct pcpu *pcpu, int cpuid, size_t size)
- void pcpu_destroy (struct pcpu *pcpu)
- pcpu * pcpu_find (u_int cpuid)

Variables

- static struct pcpu * cpuid_to_pcpu [MAXCPU]
- cpuhead cpuhead = SLIST_HEAD_INITIALIZER(cpuhead)

9.104.1 Function Documentation

9.104.1.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_pcpu. *c*, v 1.8 2005/11/03 21:06:29 jhb Exp $")

9.104.1.2 void pcpu_destroy (struct pcpu * *pcpu*)

Definition at line 83 of file subr_pcpu.c.

References cpuhead, and cpuid_to_pcpu.

9.104.1.3 struct pcpu* pcpu_find (u_int *cpuid*)

Definition at line 94 of file subr_pcpu.c.

References cpuid_to_pcpu.

Referenced by _do_lock_normal(), sched_add(), and witness_display_spinlock().

9.104.1.4 void pcpu_init (struct pcpu * *pcpu*, int *cpuid*, size_t *size*)

Definition at line 66 of file subr_pcpu.c.

References cpuhead, and cpuid_to_pcpu.

9.104.2 Variable Documentation

9.104.2.1 struct cpuhead cpuhead = SLIST_HEAD_INITIALIZER(cpuhead)

Definition at line 60 of file subr_pcpu.c.

Referenced by idle_setup(), kdb_thr_ctx(), pcpu_destroy(), and pcpu_init().

9.104.2.2 struct pcpu* cpuid_to_pcpu[MAXCPU] [static]

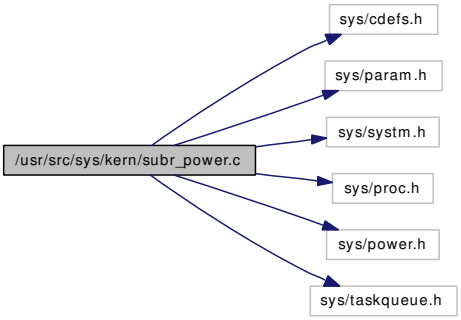
Definition at line 59 of file subr_pcpu.c.

Referenced by pcpu_destroy(), pcpu_find(), and pcpu_init().

9.105 /usr/src/sys/kern/subr_power.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/proc.h>
#include <sys/power.h>
#include <sys/taskqueue.h>
```

Include dependency graph for subr_power.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_power.c,v 1.8 2005/11/09 16:22:56 imp Exp $")
- static void power_pm_deferred_fn (void *arg, int pending)
- int power_pm_register (u_int pm_type, power_pm_fn_t pm_fn, void *pm_arg)
- u_int power_pm_get_type (void)
- void power_pm_suspend (int state)
- int power_profile_get_state (void)
- void power_profile_set_state (int state)

Variables

- static u_int power_pm_type = POWER_PM_TYPE_NONE
- static power_pm_fn_t power_pm_fn = NULL
- static void * power_pm_arg = NULL
- static struct task power_pm_task
- static int power_profile_state = POWER_PROFILE_PERFORMANCE

9.105.1 Function Documentation

9.105.1.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_power. *c*, v 1.8 2005/11/09 16:22:56 imp Exp $")

9.105.1.2 static void power_pm_deferred_fn (void * *arg*, int *pending*) [static]

Definition at line 43 of file subr_power.c.

References power_pm_arg, and power_pm_fn.

Referenced by power_pm_register().

9.105.1.3 u_int power_pm_get_type (void)

Definition at line 70 of file subr_power.c.

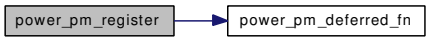
References power_pm_type.

9.105.1.4 int power_pm_register (u_int *pm_type*, power_pm_fn_t *pm_fn*, void * *pm_arg*)

Definition at line 51 of file subr_power.c.

References power_pm_arg, power_pm_deferred_fn(), power_pm_fn, power_pm_task, and power_pm_type.

Here is the call graph for this function:



9.105.1.5 void power_pm_suspend (int *state*)

Definition at line 77 of file subr_power.c.

References power_pm_fn, power_pm_task, and taskqueue_enqueue().

Here is the call graph for this function:

9.105.1.6 int power_profile_get_state (void)

Definition at line 97 of file subr_power.c.

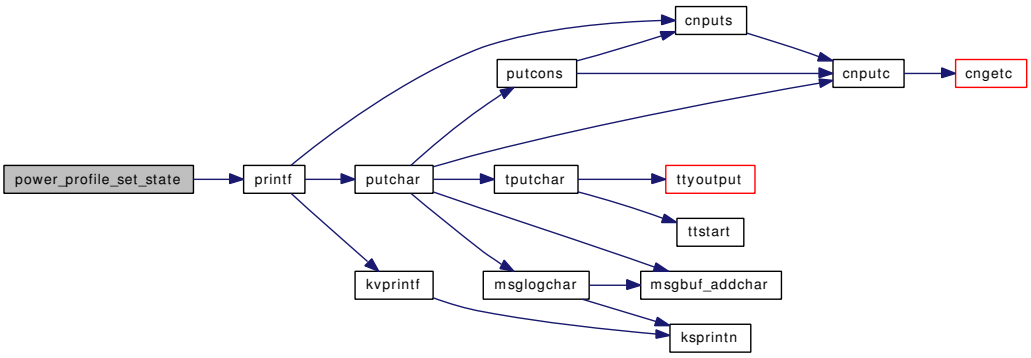
References power_profile_state.

9.105.1.7 void power_profile_set_state (int *state*)

Definition at line 103 of file subr_power.c.

References bootverbose, power_profile_state, and printf().

Here is the call graph for this function:



9.105.2 Variable Documentation

9.105.2.1 void* power_pm_arg = NULL [static]

Definition at line 39 of file subr_power.c.

Referenced by power_pm_deferred_fn(), and power_pm_register().

9.105.2.2 power_pm_fn_t power_pm_fn = NULL [static]

Definition at line 38 of file subr_power.c.

Referenced by power_pm_deferred_fn(), power_pm_register(), and power_pm_suspend().

9.105.2.3 struct task power_pm_task [static]

Definition at line 40 of file subr_power.c.

Referenced by power_pm_register(), and power_pm_suspend().

9.105.2.4 u_int power_pm_type = POWER_PM_TYPE_NONE [static]

Definition at line 37 of file subr_power.c.

Referenced by power_pm_get_type(), and power_pm_register().

9.105.2.5 int power_profile_state = POWER_PROFILE_PERFORMANCE [static]

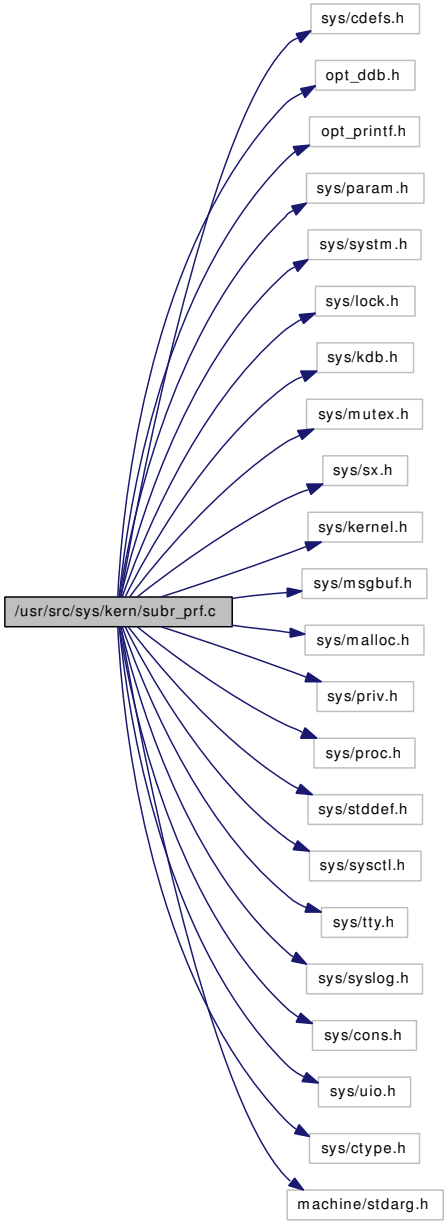
Definition at line 94 of file subr_power.c.

Referenced by power_profile_get_state(), and power_profile_set_state().

9.106 /usr/src/sys/kern/subr_prf.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ddb.h"
#include "opt_printf.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/lock.h>
#include <sys/kdb.h>
#include <sys/mutex.h>
#include <sys/sx.h>
#include <sys/kernel.h>
#include <sys/msgbuf.h>
#include <sys/malloc.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/stddef.h>
#include <sys/sysctl.h>
#include <sys/tty.h>
#include <sys/syslog.h>
#include <sys/cons.h>
#include <sys/uio.h>
#include <sys/ctype.h>
#include <machine/stdarg.h>
```

Include dependency graph for subr_prf.c:



Data Structures

- struct putchar_arg
- struct snprintf_arg

Defines

- #define TOCONS 0x01
- #define TOTTY 0x02
- #define TOLOG 0x04
- #define MAXNBUF (sizeof(intmax_t) * NBBY + 1)

- #define CONSCHUNK 128
- #define PCHAR(c) {int cc=(c); if (func) (*func)(cc,arg); else *d++ = cc; retval++; }

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_prf.c,v 1.129 2006/11/30 07:25:52 jb Exp $")
- static void msglogchar (int c, int pri)
- static void putchar (int ch, void *arg)
- static char * ksprintn (char *nbuf, uintmax_t num, int base, int *len, int upper)
- static void snprintf_func (int ch, void *arg)
- TUNABLE_INT ("kern.log_console_output",&log_console_output)
- SYSCTL_INT (_kern, OID_AUTO, log_console_output, CTLFLAG_RW,&log_console_output, 0,"Duplicate console output to the syslog.")
- TUNABLE_INT ("kern.always_console_output",&always_console_output)
- SYSCTL_INT (_kern, OID_AUTO, always_console_output, CTLFLAG_RW,&always_console_output, 0,"Always output to console despite TIOCCONS.")
- void tablefull (const char *tab)
- int uprintf (const char *fmt,...)
- void tprintf (struct proc *p, int pri, const char *fmt,...)
- int ttyprintf (struct tty *tp, const char *fmt,...)
- void log (int level, const char *fmt,...)
- void log_console (struct uio *uio)
- int printf (const char *fmt,...)
- int vprintf (const char *fmt, va_list ap)
- static void putcons (int c, struct putchar_arg *ap)
- int sprintf (char *buf, const char *cfmt,...)
- int vsprintf (char *buf, const char *cfmt, va_list ap)
- int snprintf (char *str, size_t size, const char *format,...)
- int vsnprintf (char *str, size_t size, const char *format, va_list ap)
- int vsnrprintf (char *str, size_t size, int radix, const char *format, va_list ap)
- int kvprintf (char const *fmt, void(*func)(int, void *), void *arg, int radix, va_list ap)
- void msgbufinit (void *ptr, int size)
- SYSCTL_INT (_security_bsd, OID_AUTO, unprivileged_read_msgbuf, CTLFLAG_RW,&unprivileged_read_msgbuf, 0,"Unprivileged processes may read the kernel message buffer")
- static int sysctl_kern_msgbuf (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern, OID_AUTO, msgbuf, CTLTYPE_STRING|CTLFLAG_RD, 0, 0, sysctl_kern_msgbuf,"A","Contents of kernel message buffer")
- static int sysctl_kern_msgbuf_clear (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern, OID_AUTO, msgbuf_clear, CTLTYPE_INT|CTLFLAG_RW|CTLFLAG_SECURE,&msgbuf_clearflag, 0, sysctl_kern_msgbuf_clear,"I","Clear kernel message buffer")
- void hexdump (const void *ptr, int length, const char *hdr, int flags)

Variables

- int log_open
- static int msgbufmapped
- int msgbuftrigger
- static int log_console_output = 1
- static int always_console_output = 0
- static int unprivileged_read_msgbuf = 1
- static int msgbuf_clearflag

9.106.1 Define Documentation

9.106.1.1 #define CONSCHUNK 128

Definition at line 251 of file subr_prf.c.

Referenced by log_console().

9.106.1.2 #define MAXNBUF (sizeof(intmax_t) * NBBY + 1)

Definition at line 77 of file subr_prf.c.

Referenced by kvprintf(), and msglogchar().

9.106.1.3 #define PCHAR(c) {int cc=(c); if (func) (*func)(cc,arg); else *d++ = cc; retval++; }

Referenced by kvprintf().

9.106.1.4 #define TOCONS 0x01

Definition at line 72 of file subr_prf.c.

Referenced by log(), printf(), putchar(), and vprintf().

9.106.1.5 #define TOLOG 0x04

Definition at line 74 of file subr_prf.c.

Referenced by log(), printf(), putchar(), tprintf(), and vprintf().

9.106.1.6 #define TOTTY 0x02

Definition at line 73 of file subr_prf.c.

Referenced by putchar(), tprintf(), ttyprintf(), and uprintf().

9.106.2 Function Documentation

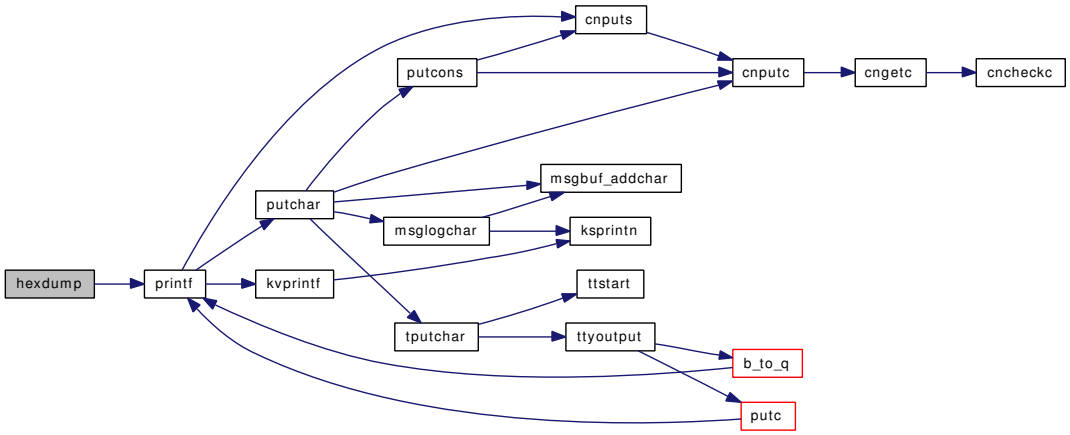
9.106.2.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_prf. *c*, v 1.129 2006/11/30 07:25:52 jb Exp $")

9.106.2.2 void hexdump (const void * *ptr*, int *length*, const char * *hdr*, int *flags*)

Definition at line 1002 of file subr_prf.c.

References printf().

Here is the call graph for this function:



9.106.2.3 static char * ksprintn (char * *nbuf*, uintmax_t *num*, int *base*, int * *len*, int *upper*) [static]

Definition at line 524 of file subr_prf.c.

Referenced by kvprintf(), and msglogchar().

9.106.2.4 int kvprintf (char const * *fmt*, void(*)(int, void *) *func*, void * *arg*, int *radix*, va_list *ap*)

Definition at line 566 of file subr_prf.c.

References ksprintn(), MAXNBUF, nbuf, PCHAR, and qflag.

Referenced by log(), printf(), sprintf(), tprintf(), ttyprintf(), uprintf(), vprintf(), vsnprintf(), vsnrprintf(), and vsprintf().

Here is the call graph for this function:

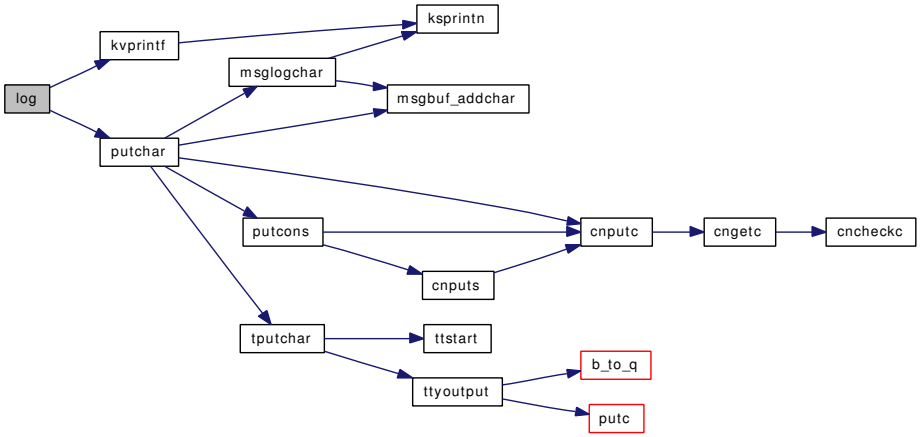
9.106.2.5 void log (int *level*, const char * *fmt*, ...)

Definition at line 234 of file subr_prf.c.

References putchar_arg::flags, kvprintf(), log_open, msgbuftrigger, putchar_arg::p_bufr, putchar_arg::pri, putchar(), TOCONS, TOLOG, and putchar_arg::tty.

Referenced by abort2(), acct(), acct_disable(), acctwatch(), ether_poll_deregister(), ether_poll_register(), expand_name(), killproc(), poll_switch(), sigexit(), syscall_not_present(), tablefull(), and tc_setclock().

Here is the call graph for this function:



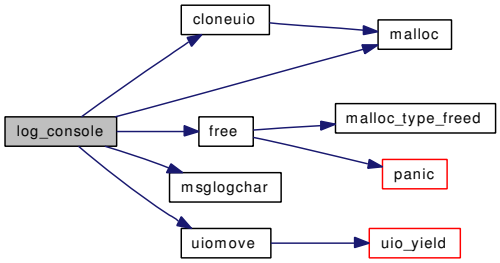
9.106.2.6 void log_console (struct uio * *uio*)

Definition at line 254 of file subr_prf.c.

References cloneuio(), CONSCHUNK, free(), log_console_output, malloc(), msgbuftrigger, msglogchar(), putchar_arg::pri, and uiomove().

Referenced by cnwrite().

Here is the call graph for this function:

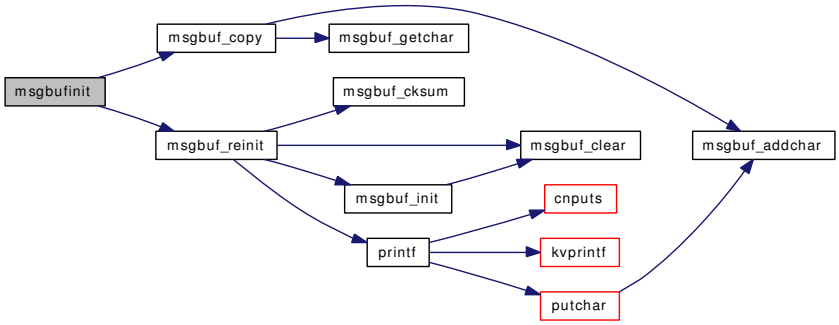


9.106.2.7 void msgbufinit (void * *ptr*, int *size*)

Definition at line 913 of file subr_prf.c.

References msgbuf_copy(), msgbuf_reinit(), and msgbufmapped.

Here is the call graph for this function:



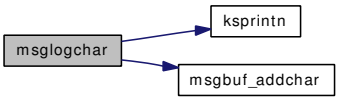
9.106.2.8 static void msglogchar (int *c*, int *pri*) [static]

Definition at line 881 of file subr_prf.c.

References ksprintn(), MAXNBUF, msgbuf_addchar(), msgbufmapped, and nbuf.

Referenced by log_console(), and putchar().

Here is the call graph for this function:



9.106.2.9 int printf (const char * *fmt*, ...)

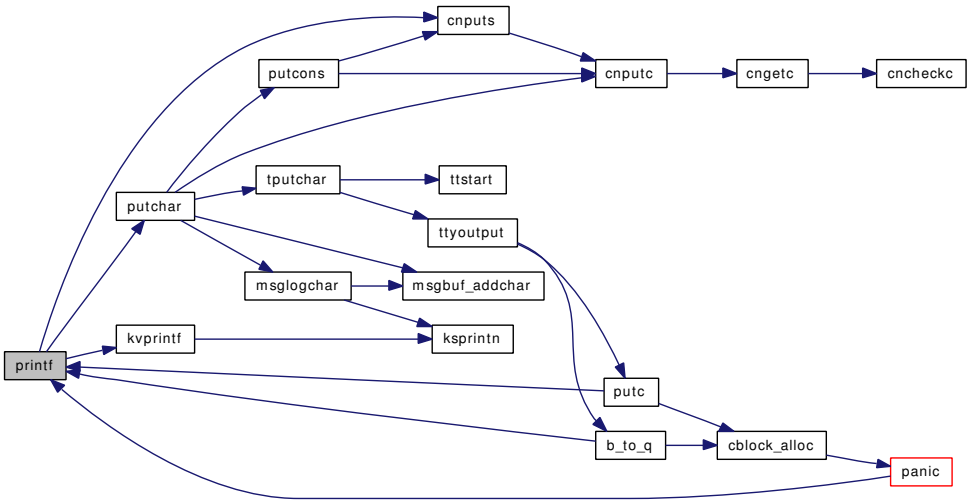
Definition at line 290 of file subr_prf.c.

References cnputs(), putchar_arg::flags, kvprintf(), msgbuftrigger, putchar_arg::n_bufr, putchar_arg::p_bufr, putchar_arg::p_next, panicstr, putchar_arg::pri, putchar(), putchar_arg::remain, TOCONS, TOLOG, and putchar_arg::tty.

Referenced by _lockmgr(), acl_posix1e_mode_to_entry(), acl_posix1e_mode_to_perm(), aio_daemon(), b_to_q(), blist_create(), blst_meta_free(), brelse(), bufdone_finish(), bus_print_child_footer(), bus_print_child_header(), calcru1(), cblock_alloc_cblocks(), cf_get_method(), cf_set_method(), chgproccnt(), chgsbsize(), clone_create(), cluster_rbuild(), cnadd(), cpu_tick_calibrate(), cpufreq_curr_sysctl(), cpufreq_levels_sysctl(), destroy_devl(), devclass_alloc_unit(), devfs_first(), devfs_fixup(), device_attach(), device_print_prettyname(), device_probe_and_attach(), device_set_devclass(), devstat_add_entry(), disk_err(), doadump(), elf_obj_lookup(), enroll(), exec_gzip_imgact(), exit1(), falloc(), firmware_get(), firmware_modevent(), firmware_register(), fork1(), fork_exit(), getblk(), getnewbuf(), getnewvnode(), hexdump(), init_dynamic_kenv(), inittimecounter(), inittodr(), ioctl(), issignal(), ithread_execute_handlers(), kdb_backtrace(), kdb_enter(), kdb_init(), kern_kldunload(), kern_ptrace(), kern_stat(), kern_statfs(), kern_vfs_bio_buffer_alloc(), kmeminit(), knlist_destroy(), kproc_shutdown(), kqueue_add_filteropts(), kse_exit(), kse_release(), ktr_tracepoint(), lf_findoverlap(), link_elf_error(), link_elf_link_preload(), link_elf_lookup_symbol(), link_elf_preload_parse_symbols(), linker_addmodules(), linker_file_register_modules(), linker_hints_lookup(), linker_load_dependencies(), linker_preload(), lockmgr_printinfo(), logopen(), logtimeout(), lookup(), m_print(), m_pulldown(), make_dev_alias(), make_dev_credv(), make_device(), malloc(), malloc_uninit(), mbp_alloc_page(), mbp_destroy(), mbp_get(), mbp_get_keep(), mi_startup(), module_register(), module_register_init(), msgbuf_reinit(), mtx_pool_create(),

net_add_domain(), panic(), physio(), pipespace_new(), power_profile_set_state(), prep_cdevsw(), print_caddr_t(), print_uptime(), printf_uuid(), propagate_priority(), pty_clone(), pty_maybecleanup(), putc(), register_posix_clock(), relocate_file(), relookup(), res_find(), resettodr(), resource_list_print_type(), root_mount_wait(), root_print_child(), runq_print(), sched_priority(), sched_sync(), setrunnable(), settime(), shutdown_halt(), shutdown_panic(), shutdown_reset(), softclock(), stack_print(), start_init(), sysctl_add_oid(), sysctl_register_oid(), sysctl_remove_oid(), sysctl_unregister_oid(), SYSINIT(), taskqueue_start_threads(), tc_init(), tdq_print(), ttcompat(), ttcompatgetflags(), ttyrub(), tvtohz(), uifree(), userret(), vfs_filteropt(), vfs_mount_destroy(), vfs_mountroot_ask(), vfs_mountroot_try(), vfs_register(), vfs_sysctl(), vfs_unmountall(), vm_hold_free_pages(), vn_printf(), vnlru_proc(), vntblinit(), vop_nostrategy(), vop_strategy_pre(), witness_checkorder(), witness_child_get(), witness_get(), witness_list_lock(), witness_lock_list_get(), witness_unlock(), and witness_warn().

Here is the call graph for this function:



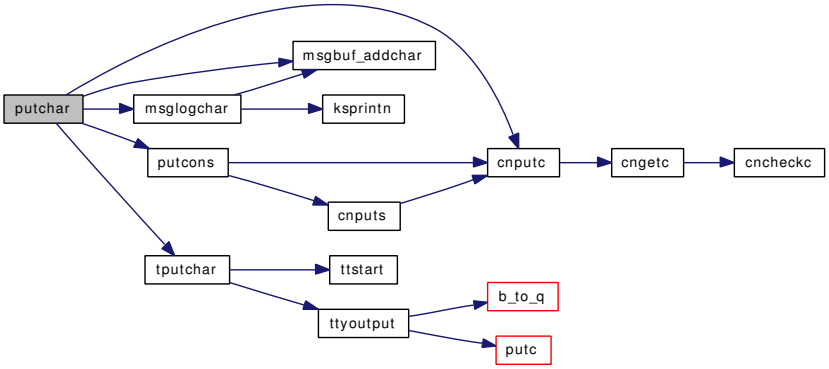
9.106.2.10 static void putchar (int *ch*, void * *arg*) [static]

Definition at line 401 of file subr_prf.c.

References always_console_output, cnputc(), putchar_arg::flags, kdb_active, msgbuf_addchar(), msglogchar(), panicstr, putchar_arg::pri, putcons(), TOCONS, TOLOG, TOTTY, tputchar(), and putchar_arg::tty.

Referenced by log(), printf(), tprintf(), ttyprintf(), uprintf(), and vprintf().

Here is the call graph for this function:



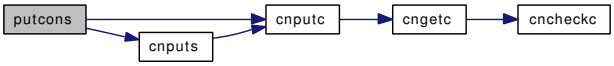
9.106.2.11 static void putcons (int *c*, struct putchar_arg * *ap*) [static]

Definition at line 367 of file subr_prf.c.

References cnputc(), cnputs(), putchar_arg::n_bufr, putchar_arg::p_bufr, putchar_arg::p_next, and putchar_arg::remain.

Referenced by putchar().

Here is the call graph for this function:



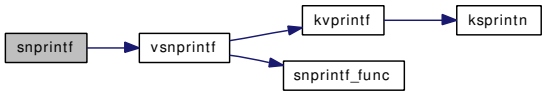
9.106.2.12 int snprintf (char * *str*, size_t *size*, const char * *format*, ...)

Definition at line 461 of file subr_prf.c.

References vsnprintf().

Referenced by cn_devopen(), devadded(), devaddq(), devclass_add_device(), devctl_notify(), devremoved(), init_sleepqueues(), kdb_sysctl_available(), link_elf_lookup_set(), linker_hints_lookup(), linker_lookup_file(), mqfs_read(), snprintf_uuid(), sysctl_rman(), and sysctl_sysctl_name().

Here is the call graph for this function:



9.106.2.13 static void snprintf_func (int *ch*, void * *arg*) [static]

Definition at line 507 of file subr_prf.c.

References snprintf_arg::remain, and snprintf_arg::str.

Referenced by vsnprintf(), and vsnrprintf().

9.106.2.14 int sprintf (char * *buf*, const char * *cfmt*, ...)

Definition at line 432 of file subr_prf.c.

References kvprintf().

Referenced by devtoname(), expand_name(), linker_find_file_by_name(), setenv(), sysctl_kern_console(), sysctl_kern_timecounter_choice(), and vfs_mountroot_try().

Here is the call graph for this function:

9.106.2.15 SYSCTL_INT (_security_bsd, OID_AUTO, unprivileged_read_msgbuf, CTLFLAG_RW, & *unprivileged_read_msgbuf*, 0, "Unprivileged processes may read the kernel message buffer")

9.106.2.16 SYSCTL_INT (_kern, OID_AUTO, always_console_output, CTLFLAG_RW, & *always_console_output*, 0, "Always output to console despite TIOCCONS.")

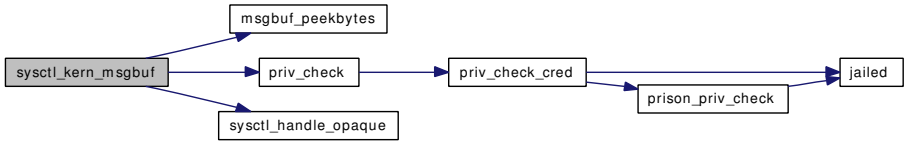
9.106.2.17 SYSCTL_INT (_kern, OID_AUTO, log_console_output, CTLFLAG_RW, & *log_console_output*, 0, "Duplicate console output to the syslog.")

9.106.2.18 static int sysctl_kern_msgbuf (SYSCTL_HANDLER_ARGS) [static]

Definition at line 935 of file subr_prf.c.

References buf, msgbuf_peekbytes(), priv_check(), and sysctl_handle_opaque().

Here is the call graph for this function:

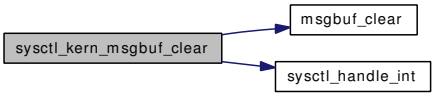


9.106.2.19 static int sysctl_kern_msgbuf_clear (SYSCTL_HANDLER_ARGS) [static]

Definition at line 963 of file subr_prf.c.

References msgbuf_clear(), and sysctl_handle_int().

Here is the call graph for this function:



9.106.2.20 SYSCTL_PROC (_kern, OID_AUTO, msgbuf_clear, CTLTYPE_INT|CTLFLAG_RW| *CTLFLAG_SECURE*, & *msgbuf_clearflag*, 0, sysctl_kern_msgbuf_clear, "I", "Clear kernel message buffer")

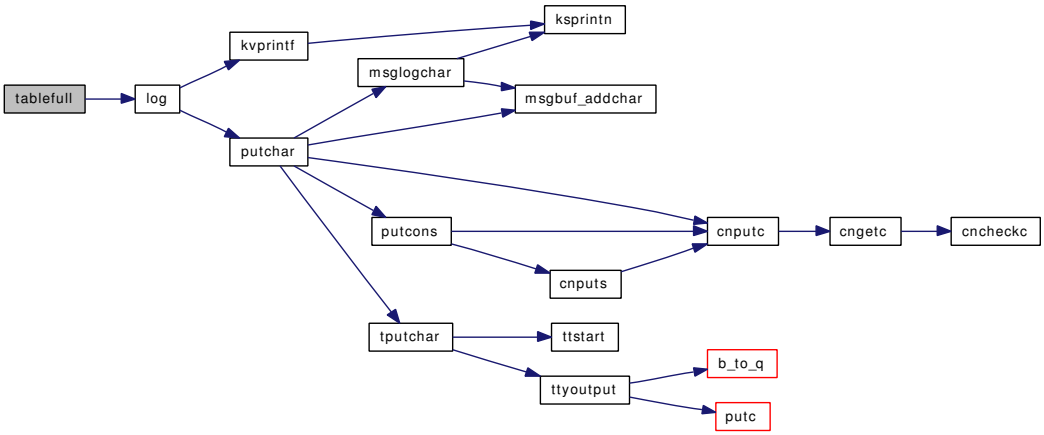
9.106.2.21 SYSCTL_PROC (_kern, OID_AUTO, msgbuf, CTLTYPE_STRING| *CTLFLAG_RD*, 0, 0, sysctl_kern_msgbuf, "A", "Contents of kernel message buffer")

9.106.2.22 void tablefull (const char * *tab*)

Definition at line 118 of file subr_prf.c.

References log().

Here is the call graph for this function:

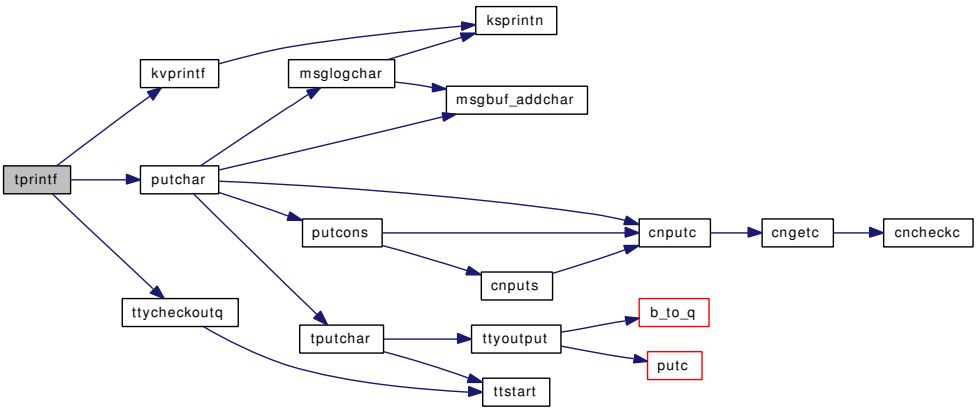


9.106.2.23 void tprintf (struct proc * *p*, int *pri*, const char * *fmt*, ...)

Definition at line 169 of file subr_prf.c.

References putchar_arg::flags, Giant, kvprintf(), msgbuftrigger, putchar_arg::pri, putchar(), TOLOG, TOTTY, putchar_arg::tty, and ttycheckoutq().

Here is the call graph for this function:



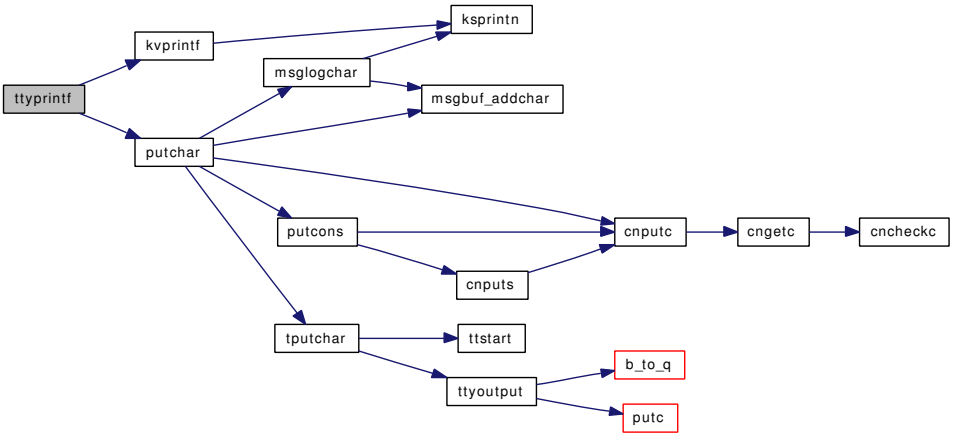
9.106.2.24 int ttyprintf (struct tty * *tp*, const char * *fmt*, ...)

Definition at line 214 of file subr_prf.c.

References putchar_arg::flags, kvprintf(), putchar(), TOTTY, and putchar_arg::tty.

Referenced by ttyinfo().

Here is the call graph for this function:



9.106.2.25 TUNABLE_INT ("kern.always_console_output", & *always_console_output*)

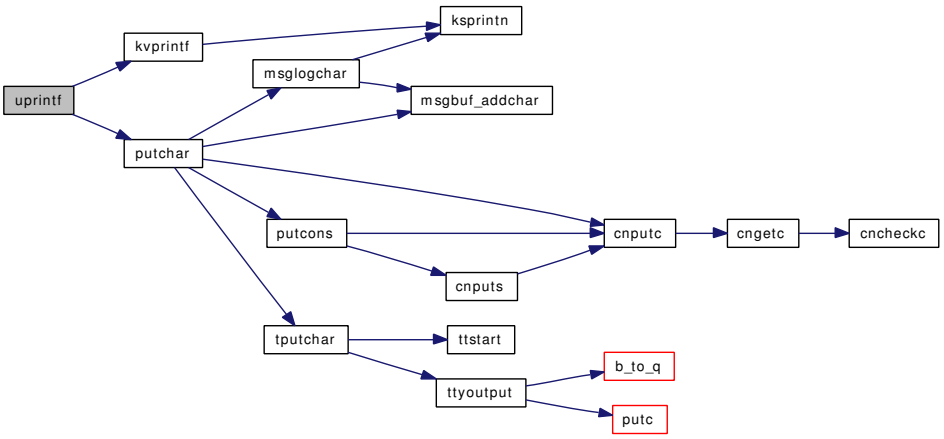
9.106.2.26 TUNABLE_INT ("kern.log_console_output", & *log_console_output*)

9.106.2.27 int uprintf (const char * *fmt*, ...)

Definition at line 128 of file subr_prf.c.

References putchar_arg::flags, Giant, kvprintf(), putchar(), td, TOTTY, and putchar_arg::tty.

Here is the call graph for this function:



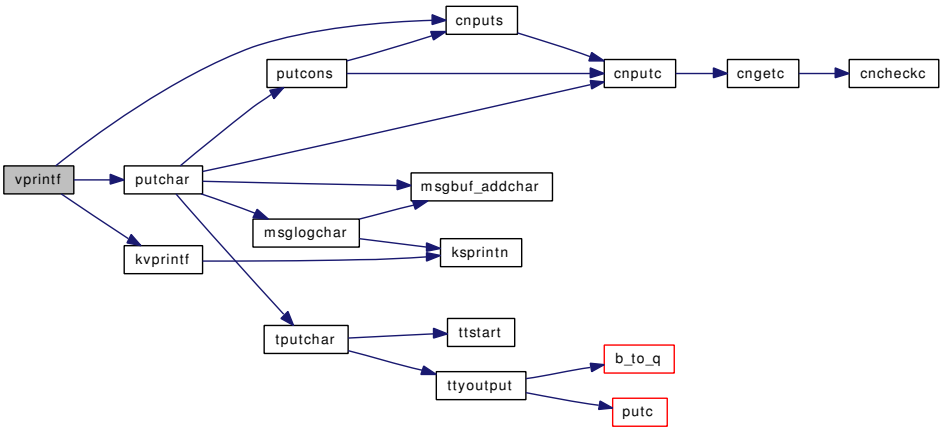
9.106.2.28 int vprintf (const char * *fmt*, va_list *ap*)

Definition at line 330 of file subr_prf.c.

References cnputs(), putchar_arg::flags, kvprintf(), msgbuftrigger, putchar_arg::n_bufr, putchar_arg::p_-bufr, putchar_arg::p_next, panicstr, putchar_arg::pri, putchar(), putchar_arg::remain, TOCONS, TOLOG, and putchar_arg::tty.

Referenced by device_printf(), panic(), vn_printf(), and witness_warn().

Here is the call graph for this function:



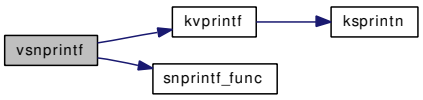
9.106.2.29 int vsnprintf (char * *str*, size_t *size*, const char * *format*, va_list *ap*)

Definition at line 476 of file subr_prf.c.

References kvprintf(), snprintf_arg::remain, snprintf_func(), and snprintf_arg::str.

Referenced by intr_event_create(), kthread_create(), panic(), sbuf_vprintf(), snprintf(), taskqueue_start_-threads(), and vfs_mount_error().

Here is the call graph for this function:



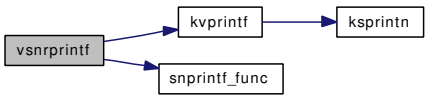
9.106.2.30 int vsnrprintf (char * *str*, size_t *size*, int *radix*, const char * *format*, va_list *ap*)

Definition at line 493 of file subr_prf.c.

References kvprintf(), snprintf_arg::remain, snprintf_func(), and snprintf_arg::str.

Referenced by make_dev_alias(), make_dev_credv(), and ttycreate().

Here is the call graph for this function:

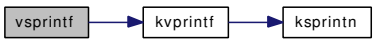


9.106.2.31 int vsprintf (char * *buf*, const char * *cfmt*, va_list *ap*)

Definition at line 448 of file subr_prf.c.

References kvprintf().

Here is the call graph for this function:



9.106.3 Variable Documentation

9.106.3.1 int always_console_output = 0 [static]

Definition at line 109 of file subr_prf.c.

Referenced by putchar().

9.106.3.2 int log_console_output = 1 [static]

Definition at line 104 of file subr_prf.c.

Referenced by log_console().

9.106.3.3 int log_open

Definition at line 84 of file subr_log.c.

Referenced by log(), logclose(), logopen(), and logtimeout().

9.106.3.4 int msgbuf_clearflag [static]

Definition at line 960 of file subr_prf.c.

9.106.3.5 int msgbufmapped [static]

Definition at line 101 of file subr_prf.c.

Referenced by msgbufinit(), and msglogchar().

9.106.3.6 int msgbuftrigger

Definition at line 102 of file subr_prf.c.

Referenced by log(), log_console(), logtimeout(), printf(), tprintf(), and vprintf().

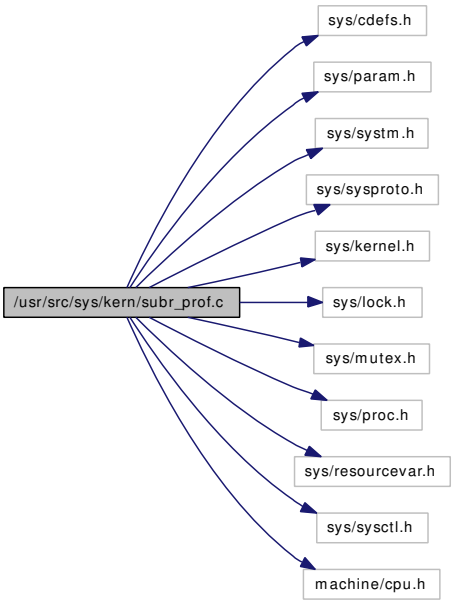
9.106.3.7 int unprivileged_read_msgbuf = 1 [static]

Definition at line 928 of file subr_prf.c.

9.107 /usr/src/sys/kern/subr_prof.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/sysproto.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/resourcevar.h>
#include <sys/sysctl.h>
#include <machine/cpu.h>
```

Include dependency graph for subr_prof.c:



Data Structures

- struct profil_args

Defines

- #define PC_TO_INDEX(pc, prof)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_prof.c,v 1.76 2005/12/16 22:08:32 jhb Exp $")
- int profil (struct thread *td, struct profil_args *uap)
- void addupc_intr (struct thread *td, uintfptr_t pc, u_int ticks)
- void addupc_task (struct thread *td, uintfptr_t pc, u_int ticks)

9.107.1 Define Documentation

9.107.1.1 #define PC_TO_INDEX(pc, prof)

Value:

```
((int)(((u_quad_t)((pc) - (prof)->pr_off) * \
            (u_quad_t)((prof)->pr_scale)) >> 16) & ~1)
```

Definition at line 446 of file subr_prof.c.

Referenced by addupc_intr(), and addupc_task().

9.107.2 Function Documentation

9.107.2.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_prof. *c*, v 1.76 2005/12/16 22:08:32 jhb Exp $")

9.107.2.2 void addupc_intr (struct thread * *td*, uintfptr_t *pc*, u_int *ticks*)

Definition at line 464 of file subr_prof.c.

References PC_TO_INDEX, and sched_lock.

Referenced by profclock().

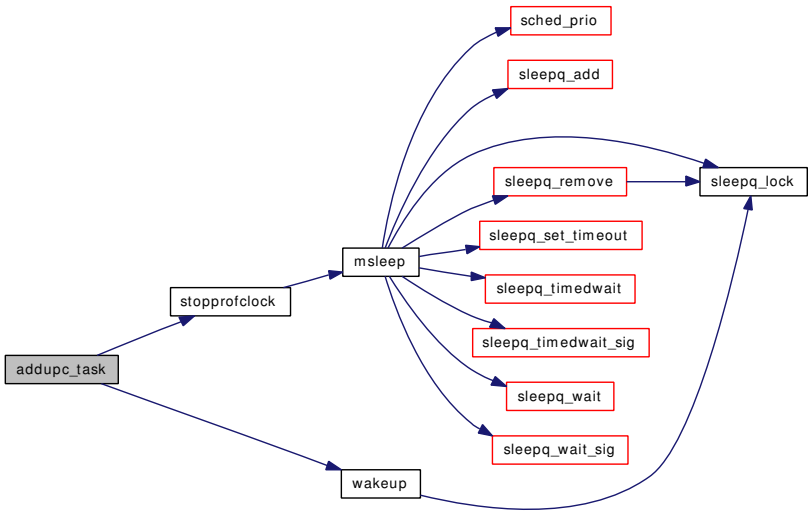
9.107.2.3 void addupc_task (struct thread * *td*, uintfptr_t *pc*, u_int *ticks*)

Definition at line 498 of file subr_prof.c.

References PC_TO_INDEX, stopprofclock(), and wakeup().

Referenced by ast(), and userret().

Here is the call graph for this function:

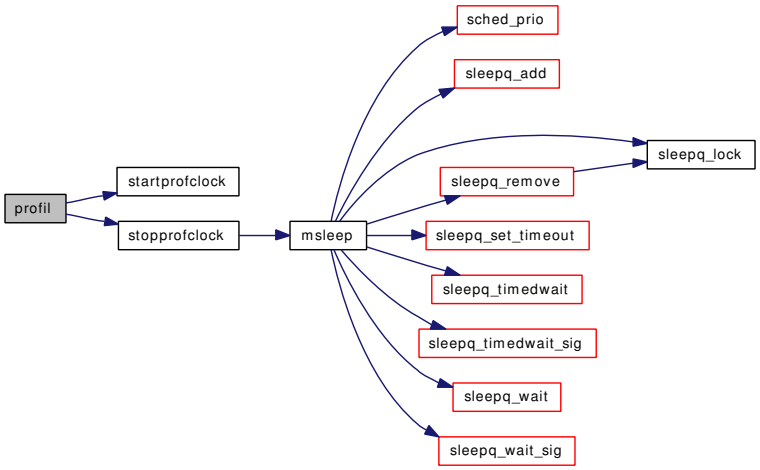


9.107.2.4 int profil (struct thread * *td*, struct profil_args * *uap*)

Definition at line 410 of file subr_prof.c.

References sched_lock, startprofclock(), and stopprofclock().

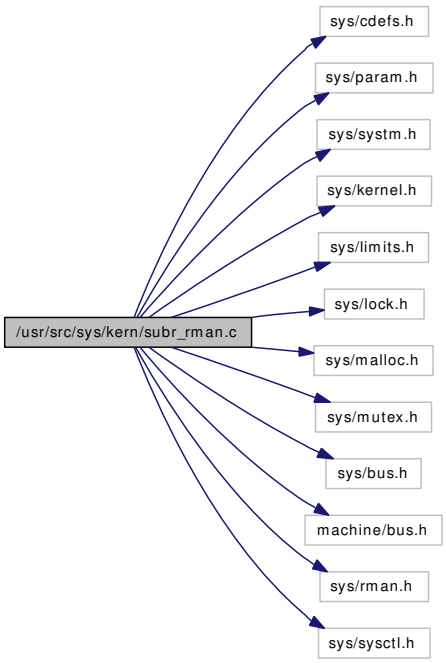
Here is the call graph for this function:



9.108 /usr/src/sys/kern/subr_rman.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/limits.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/bus.h>
#include <machine/bus.h>
#include <sys/rman.h>
#include <sys/sysctl.h>
```

Include dependency graph for subr_rman.c:



Data Structures

- struct resource_i

Defines

- #define DPRINTF(params) if (rman_debug) printf params

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_rman.c,v 1.55 2007/02/23 22:53:56 scottl Exp $")
- TUNABLE_INT ("debug.rman_debug",&rman_debug)
- SYSCTL_INT (_debug, OID_AUTO, rman_debug, CTLFLAG_RW,&rman_debug, 0,"rman debug")
- static MALLOC_DEFINE (M_RMAN,"rman","Resource manager")
- static int int_rman_activate_resource (struct rman *rm, struct resource_i *r, struct resource_i **whohas)
- static int int_rman_deactivate_resource (struct resource_i *r)
- static int int_rman_release_resource (struct rman *rm, struct resource_i *r)
- static __inline struct resource_i * int_alloc_resource (int malloc_flag)
- int rman_init (struct rman *rm)
- int rman_manage_region (struct rman *rm, u_long start, u_long end)
- int rman_init_from_resource (struct rman *rm, struct resource *r)
- int rman_fini (struct rman *rm)
- resource * rman_reserve_resource_bound (struct rman *rm, u_long start, u_long end, u_long count, u_long bound, u_int flags, struct device *dev)
- resource * rman_reserve_resource (struct rman *rm, u_long start, u_long end, u_long count, u_int flags, struct device *dev)
- int rman_activate_resource (struct resource *re)
- int rman_await_resource (struct resource *re, int pri, int timo)
- int rman_deactivate_resource (struct resource *r)
- int rman_release_resource (struct resource *re)
- uint32_t rman_make_alignment_flags (uint32_t size)
- u_long rman_get_start (struct resource *r)
- u_long rman_get_end (struct resource *r)
- u_long rman_get_size (struct resource *r)
- u_int rman_get_flags (struct resource *r)
- void rman_set_virtual (struct resource *r, void *v)
- void * rman_get_virtual (struct resource *r)
- void rman_set_bustag (struct resource *r, bus_space_tag_t t)
- bus_space_tag_t rman_get_bustag (struct resource *r)
- void rman_set_bushandle (struct resource *r, bus_space_handle_t h)
- bus_space_handle_t rman_get_bushandle (struct resource *r)
- void rman_set_rid (struct resource *r, int rid)
- void rman_set_start (struct resource *r, u_long start)
- void rman_set_end (struct resource *r, u_long end)
- int rman_get_rid (struct resource *r)
- device * rman_get_device (struct resource *r)
- void rman_set_device (struct resource *r, struct device *dev)
- int rman_is_region_manager (struct resource *r, struct rman *rm)
- static int sysctl_rman (SYSCTL_HANDLER_ARGS)
- SYSCTL_NODE (_hw_bus, OID_AUTO, rman, CTLFLAG_RD, sysctl_rman,"kernel resource manager")

Variables

- int rman_debug = 0
- rman_head rman_head
- static struct mtx rman_mtx

9.108.1 Define Documentation

9.108.1.1 #define DPRINTF(params) if (rman_debug) printf params

Definition at line 101 of file subr_rman.c.

Referenced by kern_msgctl(), kern_msgrcv(), kern_msgsnd(), kern_semctl(), msgctl(), msgget(), msginit(), msgrcv(), msgsnd(), rman_manage_region(), rman_reserve_resource_bound(), semexit_myhook(), semget(), and semop().

9.108.2 Function Documentation

9.108.2.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_rman. *c*, v 1.55 2007/02/23 22:53:56 scottl Exp $")

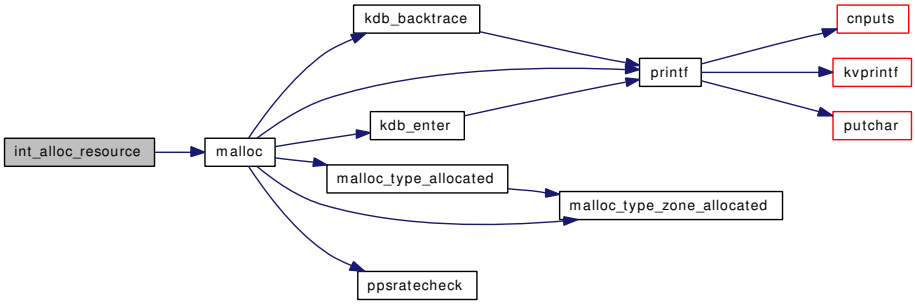
9.108.2.2 static __inline struct resource_i* int_alloc_resource (int *malloc_flag*) [static]

Definition at line 113 of file subr_rman.c.

References malloc().

Referenced by rman_manage_region(), and rman_reserve_resource_bound().

Here is the call graph for this function:



9.108.2.3 static int int_rman_activate_resource (struct rman * *rm*, struct resource_i * *r*, struct resource_i ** *whohas*) [static]

Definition at line 494 of file subr_rman.c.

Referenced by rman_activate_resource(), rman_await_resource(), and rman_reserve_resource_bound().

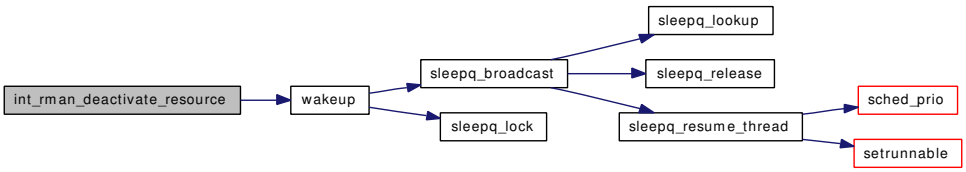
9.108.2.4 static int int_rman_deactivate_resource (struct resource_i * *r*) [static]

Definition at line 570 of file subr_rman.c.

References wakeup().

Referenced by int_rman_release_resource(), and rman_deactivate_resource().

Here is the call graph for this function:



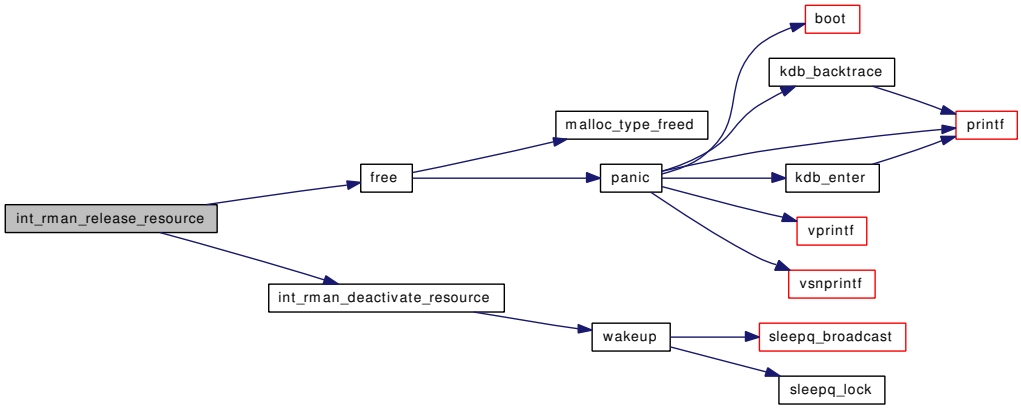
9.108.2.5 static int int_rman_release_resource (struct rman * *rm*, struct resource_i * *r*) [static]

Definition at line 594 of file subr_rman.c.

References free(), and int_rman_deactivate_resource().

Referenced by rman_release_resource(), and rman_reserve_resource_bound().

Here is the call graph for this function:



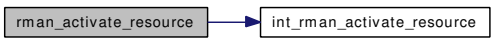
9.108.2.6 static MALLOC_DEFINE (M_RMAN, "rman", "Resource manager") [static]

9.108.2.7 int rman_activate_resource (struct resource * *re*)

Definition at line 529 of file subr_rman.c.

References int_rman_activate_resource().

Here is the call graph for this function:

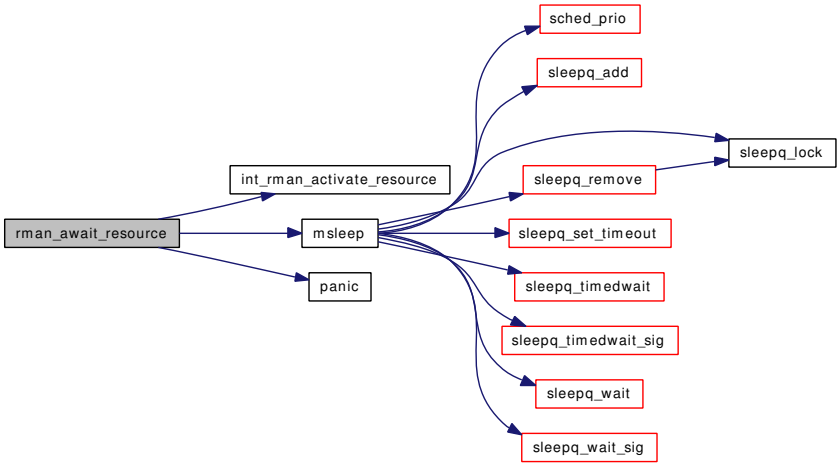


9.108.2.8 int rman_await_resource (struct resource * *re*, int *pri*, int *timo*)

Definition at line 544 of file subr_rman.c.

References int_rman_activate_resource(), msleep(), and panic().

Here is the call graph for this function:

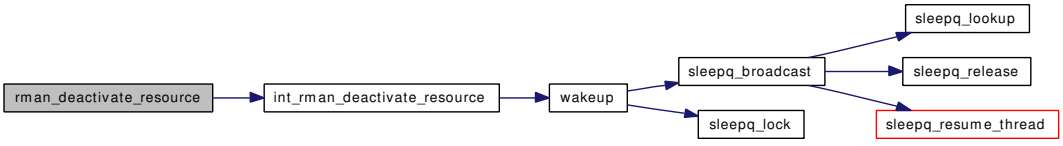


9.108.2.9 int rman_deactivate_resource (struct resource * *r*)

Definition at line 582 of file subr_rman.c.

References int_rman_deactivate_resource().

Here is the call graph for this function:

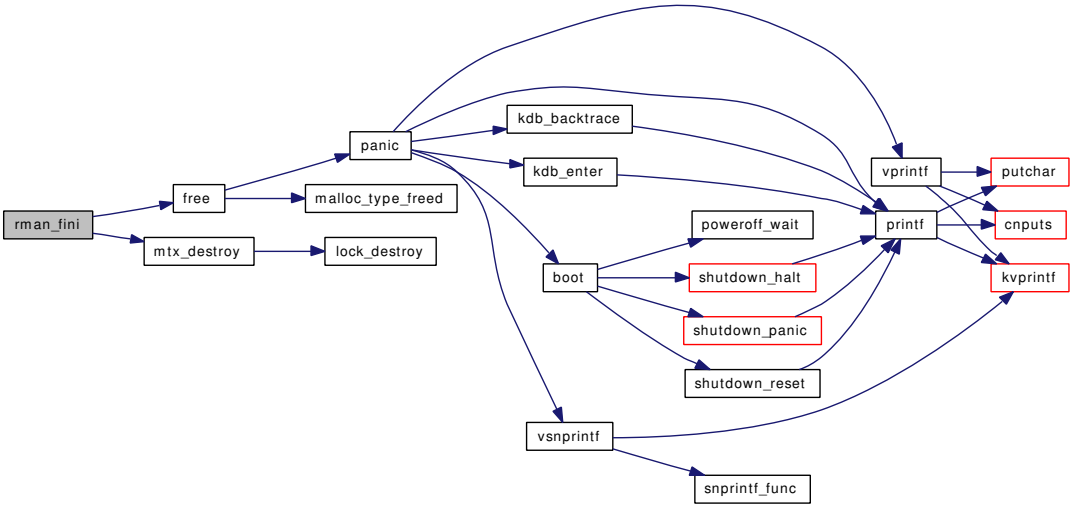


9.108.2.10 int rman_fini (struct rman * *rm*)

Definition at line 238 of file subr_rman.c.

References free(), and mtx_destroy().

Here is the call graph for this function:



9.108.2.11 bus_space_handle_t rman_get_bushandle (struct resource * r)

Definition at line 774 of file subr_rman.c.

9.108.2.12 bus_space_tag_t rman_get_bustag (struct resource * r)

Definition at line 762 of file subr_rman.c.

9.108.2.13 struct device* rman_get_device (struct resource * r)

Definition at line 804 of file subr_rman.c.

Referenced by resource_list_purge().

9.108.2.14 u_long rman_get_end (struct resource * r)

Definition at line 726 of file subr_rman.c.

Referenced by resource_list_alloc().

9.108.2.15 u_int rman_get_flags (struct resource * r)

Definition at line 738 of file subr_rman.c.

9.108.2.16 int rman_get_rid (struct resource * r)

Definition at line 798 of file subr_rman.c.

Referenced by bus_free_resource().

9.108.2.17 u_long rman_get_size (struct resource * *r*)

Definition at line 732 of file subr_rman.c.

9.108.2.18 u_long rman_get_start (struct resource * *r*)

Definition at line 720 of file subr_rman.c.

Referenced by resource_list_alloc().

9.108.2.19 void* rman_get_virtual (struct resource * *r*)

Definition at line 750 of file subr_rman.c.

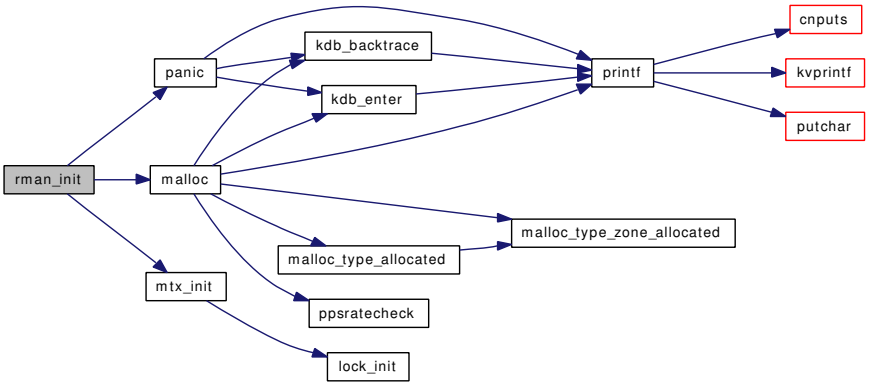
9.108.2.20 int rman_init (struct rman * *rm*)

Definition at line 125 of file subr_rman.c.

References malloc(), mtx_init(), and panic().

Referenced by rman_init_from_resource().

Here is the call graph for this function:

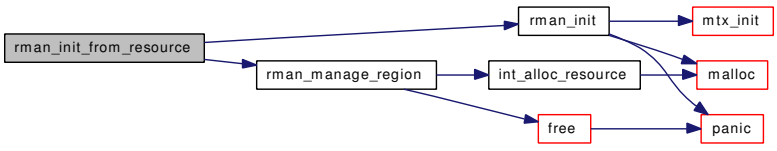


9.108.2.21 int rman_init_from_resource (struct rman * *rm*, struct resource * *r*)

Definition at line 228 of file subr_rman.c.

References rman_init(), and rman_manage_region().

Here is the call graph for this function:



9.108.2.22 int rman_is_region_manager (struct resource * *r*, struct rman * *rm*)

Definition at line 816 of file subr_rman.c.

9.108.2.23 uint32_t rman_make_alignment_flags (uint32_t *size*)

Definition at line 702 of file subr_rman.c.

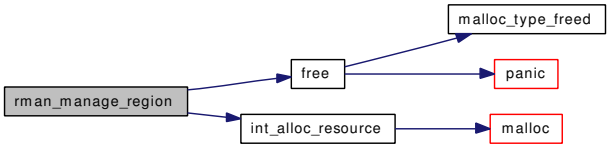
9.108.2.24 int rman_manage_region (struct rman * *rm*, u_long *start*, u_long *end*)

Definition at line 157 of file subr_rman.c.

References DPRINTF, free(), and int_alloc_resource().

Referenced by rman_init_from_resource().

Here is the call graph for this function:

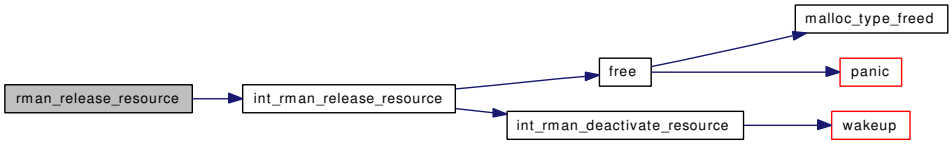


9.108.2.25 int rman_release_resource (struct resource * *re*)

Definition at line 687 of file subr_rman.c.

References int_rman_release_resource().

Here is the call graph for this function:

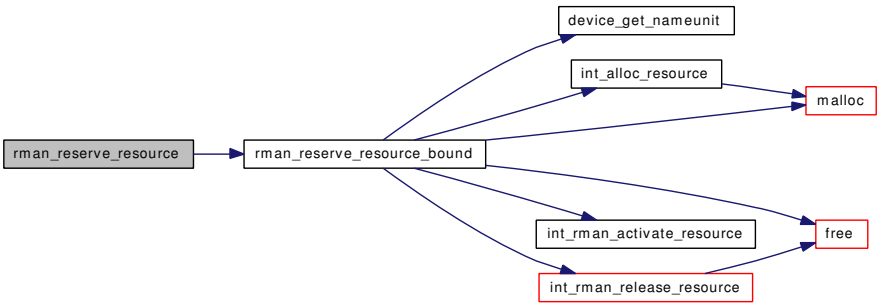


9.108.2.26 struct resource* rman_reserve_resource (struct rman * *rm*, u_long *start*, u_long *end*, u_long *count*, u_int *flags*, struct device * *dev*)

Definition at line 485 of file subr_rman.c.

References rman_reserve_resource_bound().

Here is the call graph for this function:



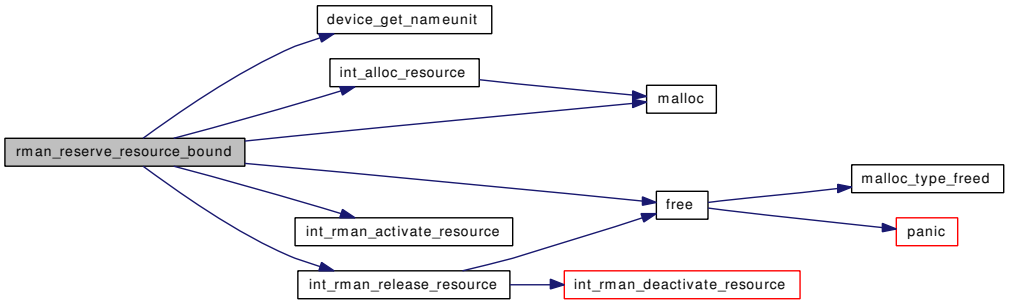
9.108.2.27 struct resource* rman_reserve_resource_bound (struct rman * *rm*, u_long *start*, u_long *end*, u_long *count*, u_long *bound*, u_int *flags*, struct device * *dev*)

Definition at line 270 of file subr_rman.c.

References device_get_nameunit(), DPRINTF, free(), int_alloc_resource(), int_rman_activate_resource(), int_rman_release_resource(), malloc(), and resource_i::r_r.

Referenced by rman_reserve_resource().

Here is the call graph for this function:



9.108.2.28 void rman_set_bushandle (struct resource * *r*, bus_space_handle_t *h*)

Definition at line 768 of file subr_rman.c.

9.108.2.29 void rman_set_bustag (struct resource * *r*, bus_space_tag_t *t*)

Definition at line 756 of file subr_rman.c.

9.108.2.30 void rman_set_device (struct resource * *r*, struct device * *dev*)

Definition at line 810 of file subr_rman.c.

9.108.2.31 void rman_set_end (struct resource * *r*, u_long *end*)

Definition at line 792 of file subr_rman.c.

9.108.2.32 void rman_set_rid (struct resource * *r*, int *rid*)

Definition at line 780 of file subr_rman.c.

9.108.2.33 void rman_set_start (struct resource * *r*, u_long *start*)

Definition at line 786 of file subr_rman.c.

9.108.2.34 void rman_set_virtual (struct resource * *r*, void * *v*)

Definition at line 744 of file subr_rman.c.

9.108.2.35 SYSCTL_INT (_debug, OID_AUTO, rman_debug, CTLFLAG_RW, & *rman_debug*, 0, "rman debug")

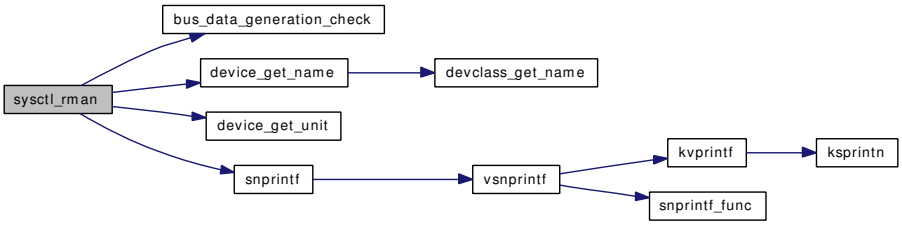
9.108.2.36 SYSCTL_NODE (_hw_bus, OID_AUTO, rman, CTLFLAG_RD, sysctl_rman, "kernel resource manager")

9.108.2.37 static int sysctl_rman (SYSCTL_HANDLER_ARGS) [static]

Definition at line 829 of file subr_rman.c.

References bus_data_generation_check(), device_get_name(), device_get_unit(), and snprintf().

Here is the call graph for this function:



9.108.2.38 TUNABLE_INT ("debug.rman_debug", & *rman_debug*)

9.108.3 Variable Documentation

9.108.3.1 int rman_debug = 0

Definition at line 96 of file subr_rman.c.

9.108.3.2 struct rman_head rman_head

Definition at line 105 of file subr_rman.c.

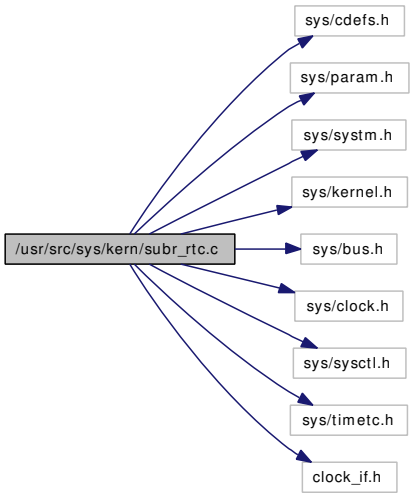
9.108.3.3 struct mtx rman_mtx [static]

Definition at line 106 of file subr_rman.c.

9.109 /usr/src/sys/kern/subr_rtc.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/bus.h>
#include <sys/clock.h>
#include <sys/sysctl.h>
#include <sys/timetc.h>
#include "clock_if.h"
```

Include dependency graph for subr_rtc.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_rtc.c,v 1.9 2006/10/02 18:23:37 phk Exp $")
- void clock_register (device_t dev, long res)
- void inittodr (time_t base)
- void resettodr ()

Variables

- static device_t clock_dev = NULL
- static long clock_res

9.109.1 Function Documentation

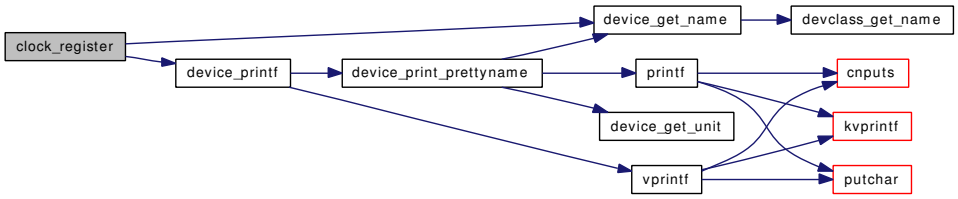
9.109.1.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_rtc. *c*, v 1.9 2006/10/02 18:23:37 phk Exp $")

9.109.1.2 void clock_register (device_t *dev*, long *res*)

Definition at line 67 of file subr_rtc.c.

References bootverbose, clock_dev, clock_res, device_get_name(), and device_printf().

Here is the call graph for this function:



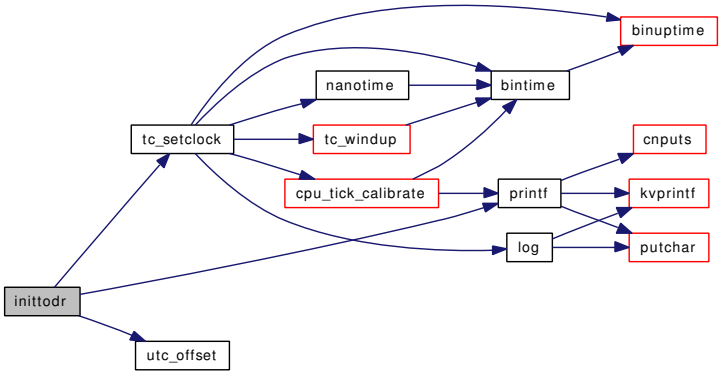
9.109.1.3 void inittodr (time_t *base*)

Definition at line 105 of file subr_rtc.c.

References clock_dev, printf(), tc_setclock(), and utc_offset().

Referenced by vfs_mountroot_try().

Here is the call graph for this function:



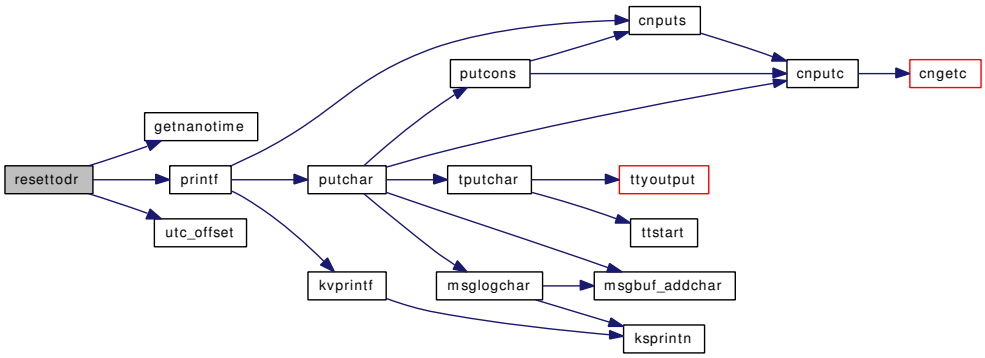
9.109.1.4 void resettodr ()

Definition at line 151 of file subr_rtc.c.

References clock_dev, disable_rtc_set, getnanotime(), printf(), and utc_offset().

Referenced by settime(), and sysctl_machdep_adjkerntz().

Here is the call graph for this function:



9.109.2 Variable Documentation

9.109.2.1 device_t clock_dev = NULL [static]

Definition at line 63 of file subr_rtc.c.

Referenced by clock_register(), inittodr(), and resettodr().

9.109.2.2 long clock_res [static]

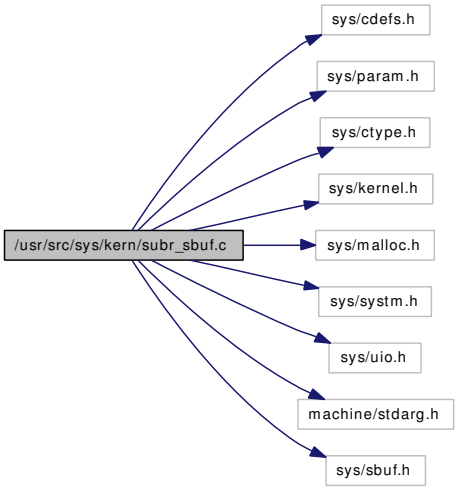
Definition at line 64 of file subr_rtc.c.

Referenced by clock_register().

9.110 /usr/src/sys/kern/subr_sbuf.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/ctype.h>
#include <sys/kernel.h>
#include <sys/malloc.h>
#include <sys/systm.h>
#include <sys/uio.h>
#include <machine/stdarg.h>
#include <sys/sbuf.h>
```

Include dependency graph for subr_sbuf.c:



Defines

- #define SBMALLOC(size) malloc(size, M_SBUF, M_WAITOK)
- #define SBFREE(buf) free(buf, M_SBUF)
- #define SBUF_ISDYNAMIC(s) ((s) → s_flags & SBUF_DYNAMIC)
- #define SBUF_ISDYNSTRUCT(s) ((s) → s_flags & SBUF_DYNSTRUCT)
- #define SBUF_ISFINISHED(s) ((s) → s_flags & SBUF_FINISHED)
- #define SBUF_HASOVERFLOWED(s) ((s) → s_flags & SBUF_OVERFLOWED)
- #define SBUF_HASROOM(s) ((s) → s_len < (s) → s_size - 1)
- #define SBUF_FREESPACE(s) ((s) → s_size - (s) → s_len - 1)
- #define SBUF_CANEXTEND(s) ((s) → s_flags & SBUF_AUTOEXTEND)
- #define SBUF_SETFLAG(s, f) do { (s) → s_flags |= (f); } while (0)
- #define SBUF_CLEARFLAG(s, f) do { (s) → s_flags &= ∼(f); } while (0)
- #define SBUF_MINEXTENDSIZE 16
- #define SBUF_MAXEXTENDSIZE PAGE_SIZE
- #define SBUF_MAXEXTENDINCR PAGE_SIZE
- #define assert_sbuf_integrity(s) do { } while (0)
- #define assert_sbuf_state(s, i) do { } while (0)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_sbuf.c,v 1.30 2005/12/23 11:49:53 phk Exp $")
- static MALLOC_DEFINE (M_SBUF,"sbuf","string buffers")
- static int sbuf_extendsize (int size)
- static int sbuf_extend (struct sbuf *s, int addlen)
- sbuf * sbuf_new (struct sbuf *s, char *buf, int length, int flags)
- sbuf * sbuf_uionew (struct sbuf *s, struct uio *uio, int *error)
- void sbuf_clear (struct sbuf *s)
- int sbuf_setpos (struct sbuf *s, int pos)
- int sbuf_bcat (struct sbuf *s, const void *buf, size_t len)
- int sbuf_bcopyin (struct sbuf *s, const void *uaddr, size_t len)
- int sbuf_bcpy (struct sbuf *s, const void *buf, size_t len)
- int sbuf_cat (struct sbuf *s, const char *str)
- int sbuf_copyin (struct sbuf *s, const void *uaddr, size_t len)
- int sbuf_cpy (struct sbuf *s, const char *str)
- int sbuf_vprintf (struct sbuf *s, const char *fmt, va_list ap)
- int sbuf_printf (struct sbuf *s, const char *fmt,...)
- int sbuf_putc (struct sbuf *s, int c)
- int sbuf_trim (struct sbuf *s)
- int sbuf_overflowed (struct sbuf *s)
- void sbuf_finish (struct sbuf *s)
- char * sbuf_data (struct sbuf *s)
- int sbuf_len (struct sbuf *s)
- void sbuf_delete (struct sbuf *s)
- int sbuf_done (struct sbuf *s)

9.110.1 Define Documentation

9.110.1.1 #define assert_sbuf_integrity(s) do { } while (0)

Definition at line 108 of file subr_sbuf.c.

Referenced by sbuf_bcat(), sbuf_bcopyin(), sbuf_bcpy(), sbuf_cat(), sbuf_clear(), sbuf_copyin(), sbuf_cpy(), sbuf_data(), sbuf_delete(), sbuf_finish(), sbuf_len(), sbuf_putc(), sbuf_setpos(), sbuf_trim(), and sbuf_vprintf().

9.110.1.2 #define assert_sbuf_state(s, i) do { } while (0)

Definition at line 109 of file subr_sbuf.c.

Referenced by sbuf_bcat(), sbuf_bcopyin(), sbuf_bcpy(), sbuf_cat(), sbuf_copyin(), sbuf_cpy(), sbuf_data(), sbuf_finish(), sbuf_putc(), sbuf_setpos(), sbuf_trim(), and sbuf_vprintf().

9.110.1.3 #define SBFREE(buf) free(buf, M_SBUF)

Definition at line 54 of file subr_sbuf.c.

Referenced by sbuf_delete(), sbuf_extend(), and sbuf_new().

9.110.1.4 #define SBMALLOC(size) malloc(size, M_SBUF, M_WAITOK)

Definition at line 53 of file subr_sbuf.c.

Referenced by sbuf_extend(), and sbuf_new().

9.110.1.5 #define SBUF_CANEXTEND(s) ((s) → s_flags & SBUF_AUTOEXTEND)

Definition at line 71 of file subr_sbuf.c.

Referenced by sbuf_extend(), and sbuf_vprintf().

9.110.1.6 #define SBUF_CLEARFLAG(s, f) do { (s) → s_flags &= ∼(f); } while (0)

Definition at line 77 of file subr_sbuf.c.

Referenced by sbuf_clear(), and sbuf_finish().

9.110.1.7 #define SBUF_FREESPACE(s) ((s) → s_size - (s) → s_len - 1)

Definition at line 70 of file subr_sbuf.c.

Referenced by sbuf_bcopyin(), sbuf_copyin(), and sbuf_vprintf().

9.110.1.8 #define SBUF_HASOVERFLOWED(s) ((s) → s_flags & SBUF_OVERFLOWED)

Definition at line 68 of file subr_sbuf.c.

Referenced by sbuf_bcat(), sbuf_bcopyin(), sbuf_cat(), sbuf_copyin(), sbuf_len(), sbuf_overflowed(), sbuf_putc(), sbuf_trim(), and sbuf_vprintf().

9.110.1.9 #define SBUF_HASROOM(s) ((s) → s_len < (s) → s_size - 1)

Definition at line 69 of file subr_sbuf.c.

Referenced by sbuf_bcat(), sbuf_cat(), sbuf_putc(), and sbuf_vprintf().

9.110.1.10 #define SBUF_ISDYNAMIC(s) ((s) → s_flags & SBUF_DYNAMIC)

Definition at line 65 of file subr_sbuf.c.

Referenced by sbuf_delete(), and sbuf_extend().

9.110.1.11 #define SBUF_ISDYNSTRUCT(s) ((s) → s_flags & SBUF_DYNSTRUCT)

Definition at line 66 of file subr_sbuf.c.

Referenced by sbuf_delete(), and sbuf_new().

9.110.1.12 #define SBUF_ISFINISHED(s) ((s) → s_flags & SBUF_FINISHED)

Definition at line 67 of file subr_sbuf.c.

Referenced by sbuf_done().

9.110.1.13 #define SBUF_MAXEXTENDINCR PAGE_SIZE

Definition at line 81 of file subr_sbuf.c.

Referenced by sbuf_extendsize().

9.110.1.14 #define SBUF_MAXEXTENDSIZE PAGE_SIZE

Definition at line 80 of file subr_sbuf.c.

Referenced by sbuf_extendsize().

9.110.1.15 #define SBUF_MINEXTENDSIZE 16

Definition at line 79 of file subr_sbuf.c.

Referenced by sbuf_extendsize().

9.110.1.16 #define SBUF_SETFLAG(s, f) do { (s) → s_flags |= (f); } while (0)

Definition at line 76 of file subr_sbuf.c.

Referenced by sbuf_bcat(), sbuf_cat(), sbuf_copyin(), sbuf_extend(), sbuf_finish(), sbuf_new(), sbuf_-putc(), and sbuf_vprintf().

9.110.2 Function Documentation

9.110.2.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_sbuf. *c*, v 1.30 2005/12/23 11:49:53 phk Exp $")

9.110.2.2 static MALLOC_DEFINE (M_SBUF, "sbuf", "string buffers") [static]

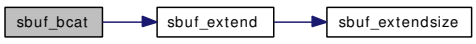
9.110.2.3 int sbuf_bcat (struct sbuf ∗ *s*, const void ∗ *buf*, size_t *len*)

Definition at line 264 of file subr_sbuf.c.

References assert_sbuf_integrity, assert_sbuf_state, sbuf_extend(), SBUF_HASOVERFLOWED, SBUF_-HASROOM, and SBUF_SETFLAG.

Referenced by sbuf_bcpy(), and sysctl_kern_malloc_stats().

Here is the call graph for this function:



9.110.2.4 int sbuf_bcopyin (struct sbuf ∗ *s*, const void ∗ *uaddr*, size_t *len*)

Definition at line 291 of file subr_sbuf.c.

References assert_sbuf_integrity, assert_sbuf_state, sbuf_extend(), SBUF_FREESPACE, and SBUF_-HASOVERFLOWED.

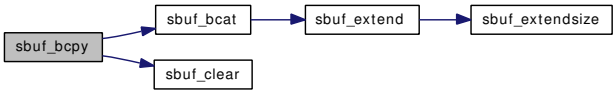
Here is the call graph for this function:

9.110.2.5 int sbuf_bcpy (struct sbuf * *s*, const void * *buf*, size_t *len*)

Definition at line 317 of file subr_sbuf.c.

References assert_sbuf_integrity, assert_sbuf_state, sbuf_bcat(), and sbuf_clear().

Here is the call graph for this function:



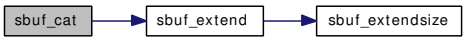
9.110.2.6 int sbuf_cat (struct sbuf * *s*, const char * *str*)

Definition at line 330 of file subr_sbuf.c.

References assert_sbuf_integrity, assert_sbuf_state, sbuf_extend(), SBUF_HASOVERFLOWED, SBUF_-HASROOM, and SBUF_SETFLAG.

Referenced by abort2(), and sbuf_cpy().

Here is the call graph for this function:



9.110.2.7 void sbuf_clear (struct sbuf * *s*)

Definition at line 229 of file subr_sbuf.c.

References assert_sbuf_integrity, and SBUF_CLEARFLAG.

Referenced by abort2(), sbuf_bcpy(), and sbuf_cpy().

9.110.2.8 int sbuf_copyin (struct sbuf * *s*, const void * *uaddr*, size_t *len*)

Definition at line 355 of file subr_sbuf.c.

References assert_sbuf_integrity, assert_sbuf_state, sbuf_extend(), SBUF_FREESPACE, SBUF_-HASOVERFLOWED, and SBUF_SETFLAG.

Referenced by abort2().

Here is the call graph for this function:

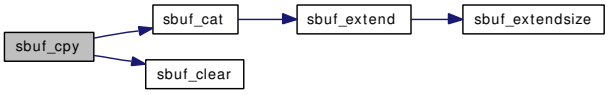
9.110.2.9 int sbuf_cpy (struct sbuf * *s*, const char * *str*)

Definition at line 390 of file subr_sbuf.c.

References assert_sbuf_integrity, assert_sbuf_state, sbuf_cat(), and sbuf_clear().

Referenced by cpufreq_levels_sysctl(), and cpufreq_settings_sysctl().

Here is the call graph for this function:



9.110.2.10 char* sbuf_data (struct sbuf * *s*)

Definition at line 527 of file subr_sbuf.c.

References assert_sbuf_integrity, and assert_sbuf_state.

Referenced by abort2(), cpufreq_levels_sysctl(), cpufreq_settings_sysctl(), and mount_argf().

9.110.2.11 void sbuf_delete (struct sbuf * *s*)

Definition at line 553 of file subr_sbuf.c.

References assert_sbuf_integrity, SBFREE, SBUF_ISDYNAMIC, and SBUF_ISDYNSTRUCT.

Referenced by abort2(), cpufreq_levels_sysctl(), cpufreq_settings_sysctl(), mount_argf(), and sbuf_-uionew().

9.110.2.12 int sbuf_done (struct sbuf * *s*)

Definition at line 572 of file subr_sbuf.c.

References SBUF_ISFINISHED.

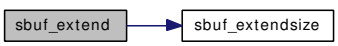
9.110.2.13 static int sbuf_extend (struct sbuf * *s*, int *addlen*) [static]

Definition at line 133 of file subr_sbuf.c.

References SBFREE, SBMALLOC, SBUF_CANEXTEND, sbuf_extendsize(), SBUF_ISDYNAMIC, and SBUF_SETFLAG.

Referenced by sbuf_bcat(), sbuf_bcopyin(), sbuf_cat(), sbuf_copyin(), sbuf_putc(), and sbuf_vprintf().

Here is the call graph for this function:



9.110.2.14 static int sbuf_extendsize (int *size*) [static]

Definition at line 113 of file subr_sbuf.c.

References SBUF_MAXEXTENDINCR, SBUF_MAXEXTENDSIZE, and SBUF_MINEXTENDSIZE.

Referenced by sbuf_extend(), and sbuf_new().

9.110.2.15 void sbuf_finish (struct sbuf * *s*)

Definition at line 513 of file subr_sbuf.c.

References assert_sbuf_integrity, assert_sbuf_state, SBUF_CLEARFLAG, and SBUF_SETFLAG.

Referenced by abort2(), cpufreq_levels_sysctl(), cpufreq_settings_sysctl(), and mount_argf().

9.110.2.16 int sbuf_len (struct sbuf * *s*)

Definition at line 539 of file subr_sbuf.c.

References assert_sbuf_integrity, and SBUF_HASOVERFLOWED.

Referenced by cpufreq_levels_sysctl(), cpufreq_settings_sysctl(), and mount_argf().

9.110.2.17 struct sbuf* sbuf_new (struct sbuf * *s*, char * *buf*, int *length*, int *flags*)

Definition at line 161 of file subr_sbuf.c.

References SBFREE, SBMALLOC, sbuf_extendsize(), SBUF_ISDYNSTRUCT, and SBUF_SETFLAG.

Referenced by abort2(), cpufreq_levels_sysctl(), cpufreq_settings_sysctl(), mount_argf(), sbuf_uionew(), and sysctl_kern_malloc_stats().

Here is the call graph for this function:

9.110.2.18 int sbuf_overflowed (struct sbuf * *s*)

Definition at line 504 of file subr_sbuf.c.

References SBUF_HASOVERFLOWED.

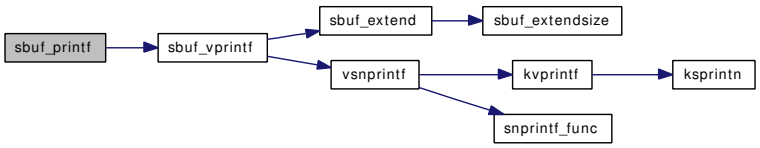
9.110.2.19 int sbuf_printf (struct sbuf * *s*, const char * *fmt*, ...)

Definition at line 450 of file subr_sbuf.c.

References sbuf_vprintf().

Referenced by abort2(), cpufreq_levels_sysctl(), cpufreq_settings_sysctl(), sbuf_printf_uuid(), and stack_-sbuf_print().

Here is the call graph for this function:

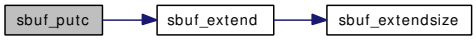


9.110.2.20 int sbuf_putc (struct sbuf * *s*, int *c*)

Definition at line 465 of file subr_sbuf.c.

References assert_sbuf_integrity, assert_sbuf_state, sbuf_extend(), SBUF_HASOVERFLOWED, SBUF_-HASROOM, and SBUF_SETFLAG.

Here is the call graph for this function:



9.110.2.21 int sbuf_setpos (struct sbuf * *s*, int *pos*)

Definition at line 244 of file subr_sbuf.c.

References assert_sbuf_integrity, and assert_sbuf_state.

9.110.2.22 int sbuf_trim (struct sbuf * *s*)

Definition at line 486 of file subr_sbuf.c.

References assert_sbuf_integrity, assert_sbuf_state, and SBUF_HASOVERFLOWED.

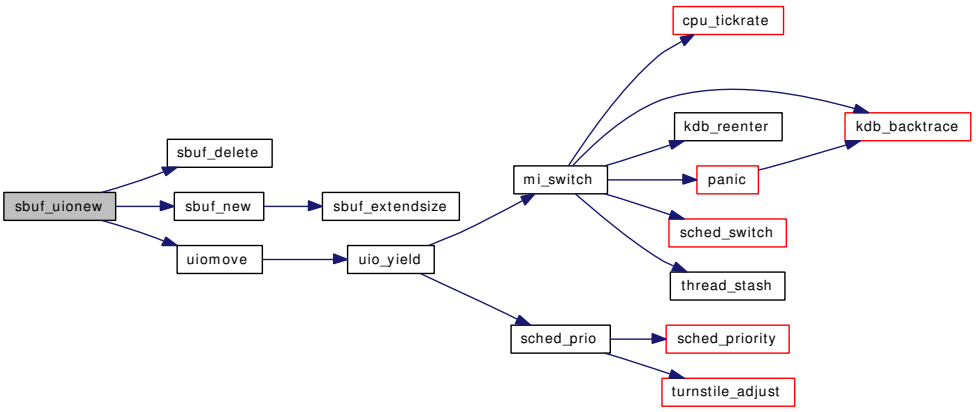
Referenced by abort2(), cpufreq_levels_sysctl(), and cpufreq_settings_sysctl().

9.110.2.23 struct sbuf* sbuf_uionew (struct sbuf * *s*, struct uio * *uio*, int * *error*)

Definition at line 202 of file subr_sbuf.c.

References sbuf_delete(), sbuf_new(), and uiomove().

Here is the call graph for this function:



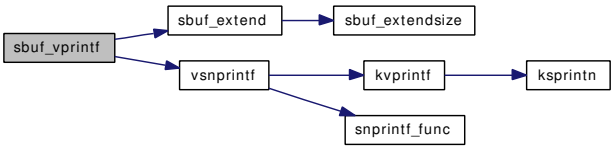
9.110.2.24 int sbuf_vprintf (struct sbuf * *s*, const char * *fmt*, va_list *ap*)

Definition at line 403 of file subr_sbuf.c.

References assert_sbuf_integrity, assert_sbuf_state, SBUF_CANEXTEND, sbuf_extend(), SBUF_FREESPACE, SBUF_HASOVERFLOWED, SBUF_HASROOM, SBUF_SETFLAG, and vsnprintf().

Referenced by mount_argf(), and sbuf_printf().

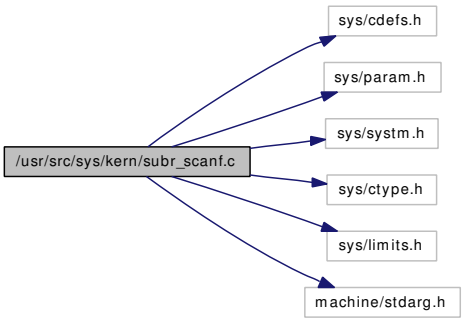
Here is the call graph for this function:



9.111 /usr/src/sys/kern/subr_scanf.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/ctype.h>
#include <sys/limits.h>
#include <machine/stdarg.h>
```

Include dependency graph for subr_scanf.c:



Defines

- #define BUF 32
- #define LONG 0x01
- #define SHORT 0x04
- #define SUPPRESS 0x08
- #define POINTER 0x10
- #define NOSKIP 0x20
- #define QUAD 0x400
- #define SIGNOK 0x40
- #define NDIGITS 0x80
- #define DPTOK 0x100
- #define EXPOK 0x200
- #define PFXOK 0x100
- #define NZDIGITS 0x200
- #define CT_CHAR 0
- #define CT_CCL 1
- #define CT_STRING 2
- #define CT_INT 3

Typedefs

- typedef u_quad_t(∗) ccfntype (const char ∗, char ∗∗, int)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_scanf.c,v 1.17 2004/04/05 21:03:35 imp Exp $")
- static const u_char * __sccl (char *, const u_char *)
- int sscanf (const char *ibuf, const char *fmt,...)
- int vsscanf (const char *inp, char const *fmt0, va_list ap)

9.111.1 Define Documentation

9.111.1.1 #define BUF 32

Definition at line 51 of file subr_scanf.c.

Referenced by vsscanf().

9.111.1.2 #define CT_CCL 1

Definition at line 81 of file subr_scanf.c.

Referenced by vsscanf().

9.111.1.3 #define CT_CHAR 0

Definition at line 80 of file subr_scanf.c.

Referenced by vsscanf().

9.111.1.4 #define CT_INT 3

Definition at line 83 of file subr_scanf.c.

Referenced by vsscanf().

9.111.1.5 #define CT_STRING 2

Definition at line 82 of file subr_scanf.c.

Referenced by vsscanf().

9.111.1.6 #define DPTOK 0x100

Definition at line 71 of file subr_scanf.c.

9.111.1.7 #define EXPOK 0x200

Definition at line 72 of file subr_scanf.c.

9.111.1.8 #define LONG 0x01

Definition at line 56 of file subr_scanf.c.

Referenced by vsscanf().

9.111.1.9 #define NDIGITS 0x80

Definition at line 69 of file subr_scanf.c.

Referenced by vsscanf().

9.111.1.10 #define NOSKIP 0x20

Definition at line 60 of file subr_scanf.c.

Referenced by vsscanf().

9.111.1.11 #define NZDIGITS 0x200

Definition at line 75 of file subr_scanf.c.

Referenced by vsscanf().

9.111.1.12 #define PFXOK 0x100

Definition at line 74 of file subr_scanf.c.

Referenced by vsscanf().

9.111.1.13 #define POINTER 0x10

Definition at line 59 of file subr_scanf.c.

Referenced by MD4Final(), MD4Transform(), and vsscanf().

9.111.1.14 #define QUAD 0x400

Definition at line 61 of file subr_scanf.c.

Referenced by vsscanf().

9.111.1.15 #define SHORT 0x04

Definition at line 57 of file subr_scanf.c.

Referenced by vsscanf().

9.111.1.16 #define SIGNOK 0x40

Definition at line 68 of file subr_scanf.c.

Referenced by vsscanf().

9.111.1.17 #define SUPPRESS 0x08

Definition at line 58 of file subr_scanf.c.

Referenced by vsscanf().

9.111.2 Typedef Documentation

9.111.2.1 typedef u_quad_t(∗) ccfntype(const char ∗, char ∗∗, int)

Definition at line 84 of file subr_scanf.c.

9.111.3 Function Documentation

9.111.3.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_scanf. *c*, v 1.17 2004/04/05 21:03:35 imp Exp $")

9.111.3.2 static const u_char ∗ __sccl (char ∗, const u_char ∗) [static]

Definition at line 542 of file subr_scanf.c.

Referenced by vsscanf().

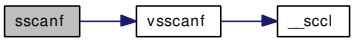
9.111.3.3 int sscanf (const char ∗ *ibuf*, const char ∗ *fmt*, ...)

Definition at line 89 of file subr_scanf.c.

References ret, and vsscanf().

Referenced by parse_uuid(), res_find(), unmount(), and vfs_mountroot_try().

Here is the call graph for this function:



9.111.3.4 int vsscanf (const char ∗ *inp*, char const ∗ *fmt0*, va_list *ap*)

Definition at line 101 of file subr_scanf.c.

References __sccl(), BUF, buf, CT_CCL, CT_CHAR, CT_INT, CT_STRING, LONG, NDIGITS, NOSKIP, NZDIGITS, PFXOK, POINTER, QUAD, SHORT, SIGNOK, and SUPPRESS.

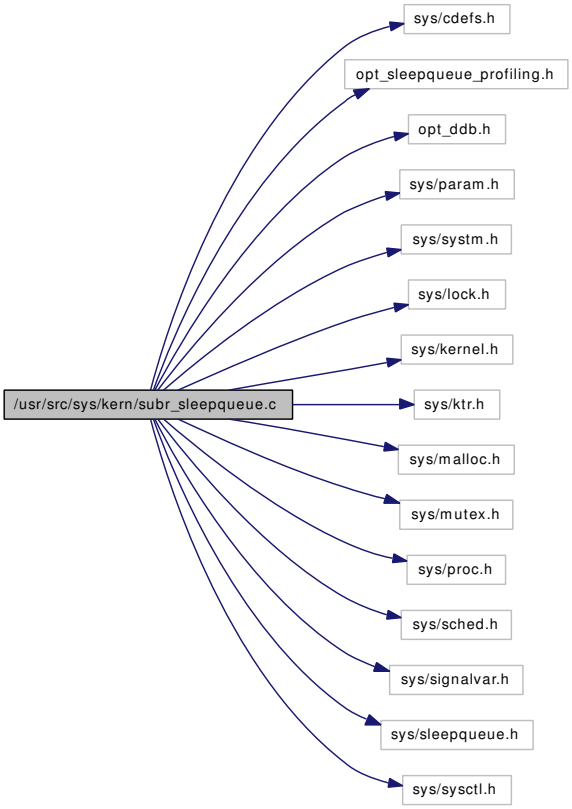
Referenced by sscanf(), and vfs_scanopt().

Here is the call graph for this function:

9.112 /usr/src/sys/kern/subr_sleepqueue.c File Reference

```
#include <sys/cdefs.h>
#include "opt_sleepqueue_profiling.h"
#include "opt_ddb.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/lock.h>
#include <sys/kernel.h>
#include <sys/ktr.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/sched.h>
#include <sys/signalvar.h>
#include <sys/sleepqueue.h>
#include <sys/sysctl.h>
```

Include dependency graph for subr_sleepqueue.c:



Data Structures

- struct sleepqueue
- struct sleepqueue_chain

Defines

- #define SC_TABLESIZE 128
- #define SC_MASK (SC_TABLESIZE - 1)
- #define SC_SHIFT 8
- #define SC_HASH(wc) (((uintptr_t)(wc) >> SC_SHIFT) & SC_MASK)
- #define SC_LOOKUP(wc) &sleepq_chains[SC_HASH(wc)]
- #define NR_SLEEPQS 2

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_sleepqueue.c,v 1.35 2006/12/17 00:14:20 kmacy Exp $")
- static MALLOC_DEFINE (M_SLEEPQUEUE,"sleepqueue","sleep queues")
- static int sleepq_catch_signals (void *wchan)
- static int sleepq_check_signals (void)
- static int sleepq_check_timeout (void)
- static void sleepq_switch (void *wchan)
- static void sleepq_timeout (void *arg)
- static void sleepq_resume_thread (struct sleepqueue *sq, struct thread *td, int pri)
- void init_sleepqueues (void)
- sleepqueue * sleepq_alloc (void)
- void sleepq_free (struct sleepqueue *sq)
- void sleepq_lock (void *wchan)
- sleepqueue * sleepq_lookup (void *wchan)
- void sleepq_release (void *wchan)
- void sleepq_add (void *wchan, struct lock_object *lock, const char *wmesg, int flags, int queue)
- void sleepq_set_timeout (void *wchan, int timo)
- void sleepq_wait (void *wchan)
- int sleepq_wait_sig (void *wchan)
- int sleepq_timedwait (void *wchan)
- int sleepq_timedwait_sig (void *wchan)
- void sleepq_signal (void *wchan, int flags, int pri, int queue)
- void sleepq_broadcast (void *wchan, int flags, int pri, int queue)
- void sleepq_remove (struct thread *td, void *wchan)
- void sleepq_abort (struct thread *td, int intrval)

Variables

- static struct sleepqueue_chain sleepq_chains [SC_TABLESIZE]

9.112.1 Define Documentation

9.112.1.1 #define NR_SLEEPQS 2

Definition at line 97 of file subr_sleepqueue.c.

Referenced by sleepq_add(), sleepq_alloc(), sleepq_broadcast(), sleepq_free(), sleepq_resume_thread(), and sleepq_signal().

9.112.1.2 #define SC_HASH(wc) (((uintptr_t)(wc) >> SC_SHIFT) & SC_MASK)

Definition at line 95 of file subr_sleepqueue.c.

9.112.1.3 #define SC_LOOKUP(wc) &sleepq_chains[SC_HASH(wc)]

Definition at line 96 of file subr_sleepqueue.c.

Referenced by sleepq_add(), sleepq_catch_signals(), sleepq_lock(), sleepq_lookup(), sleepq_release(), sleepq_resume_thread(), sleepq_set_timeout(), and sleepq_switch().

9.112.1.4 #define SC_MASK (SC_TABLESIZE - 1)

Definition at line 93 of file subr_sleepqueue.c.

9.112.1.5 #define SC_SHIFT 8

Definition at line 94 of file subr_sleepqueue.c.

9.112.1.6 #define SC_TABLESIZE 128

Definition at line 92 of file subr_sleepqueue.c.

Referenced by init_sleepqueues().

9.112.2 Function Documentation

9.112.2.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_sleepqueue. *c*, v 1.35 2006/12/17 00:14:20 kmacy Exp $")

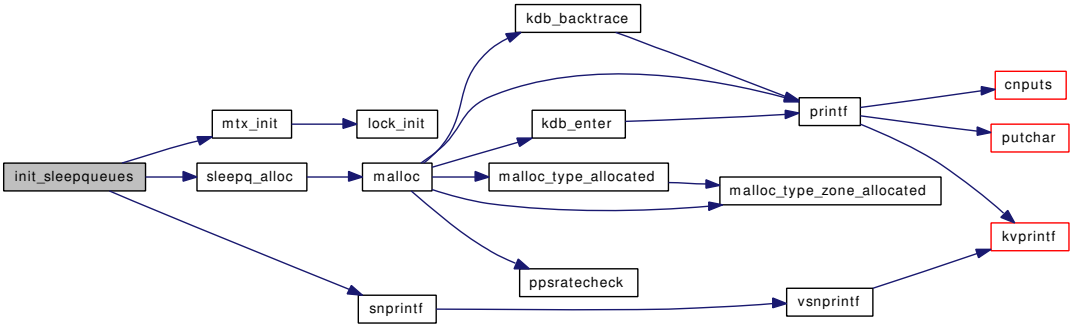
9.112.2.2 void init_sleepqueues (void)

Definition at line 163 of file subr_sleepqueue.c.

References mtx_init(), SC_TABLESIZE, sleepq_alloc(), sleepq_chains, and snprintf().

Referenced by sleepinit().

Here is the call graph for this function:



9.112.2.3 static MALLOC_DEFINE (M_SLEEPQUEUE, "sleepqueue", "sleep queues") [static]

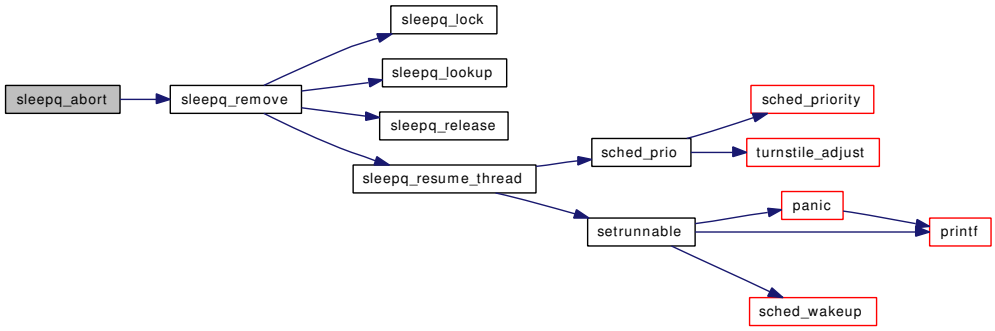
9.112.2.4 void sleepq_abort (struct thread * *td*, int *intrval*)

Definition at line 845 of file subr_sleepqueue.c.

References sched_lock, and sleepq_remove().

Referenced by kse_thr_interrupt(), tdsigwakeup(), and thread_single().

Here is the call graph for this function:



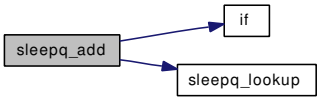
9.112.2.5 void sleepq_add (void * *wchan*, struct lock_object * *lock*, const char * *wmesg*, int *flags*, int *queue*)

Definition at line 271 of file subr_sleepqueue.c.

References if(), NR_SLEEPQS, SC_LOOKUP, sleepq_lookup(), and td.

Referenced by cv_timedwait(), cv_timedwait_sig(), cv_wait_sig(), cv_wait_unlock(), msleep(), and msleep_spin().

Here is the call graph for this function:



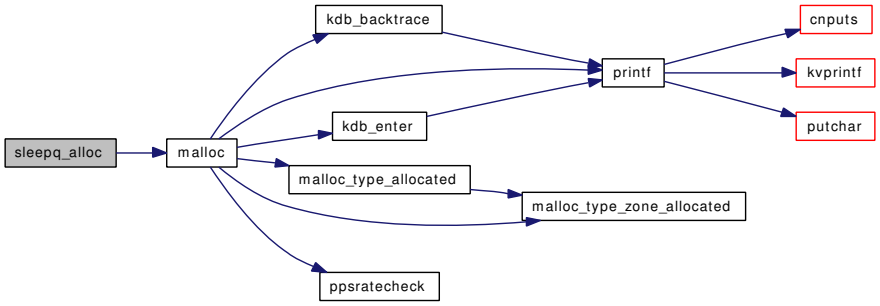
9.112.2.6 struct sleepqueue* sleepq_alloc (void)

Definition at line 194 of file subr_sleepqueue.c.

References malloc(), and NR_SLEEPQS.

Referenced by init_sleepqueues(), and thread_init().

Here is the call graph for this function:



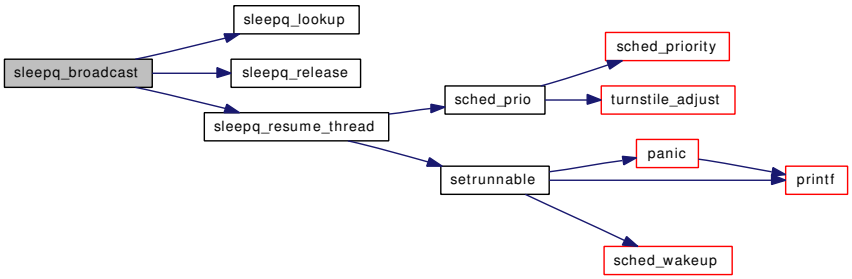
9.112.2.7 void sleepq_broadcast (void * *wchan*, int *flags*, int *pri*, int *queue*)

Definition at line 711 of file subr_sleepqueue.c.

References NR_SLEEPQS, sched_lock, sleepq_lookup(), sleepq_release(), and sleepq_resume_thread().

Referenced by cv_broadcastpri(), and wakeup().

Here is the call graph for this function:



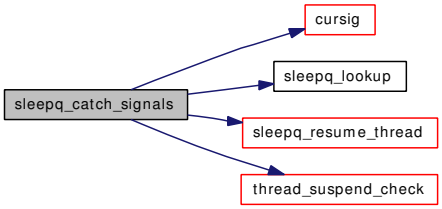
9.112.2.8 static int sleepq_catch_signals (void * *wchan*) [static]

Definition at line 366 of file subr_sleepqueue.c.

References cursig(), ret, SC_LOOKUP, sched_lock, sleepq_lookup(), sleepq_resume_thread(), td, and thread_suspend_check().

Referenced by sleepq_timedwait_sig(), and sleepq_wait_sig().

Here is the call graph for this function:



9.112.2.9 static int sleepq_check_signals (void) [static]

Definition at line 512 of file subr_sleepqueue.c.

References sched_lock, and td.

Referenced by sleepq_timedwait_sig(), and sleepq_wait_sig().

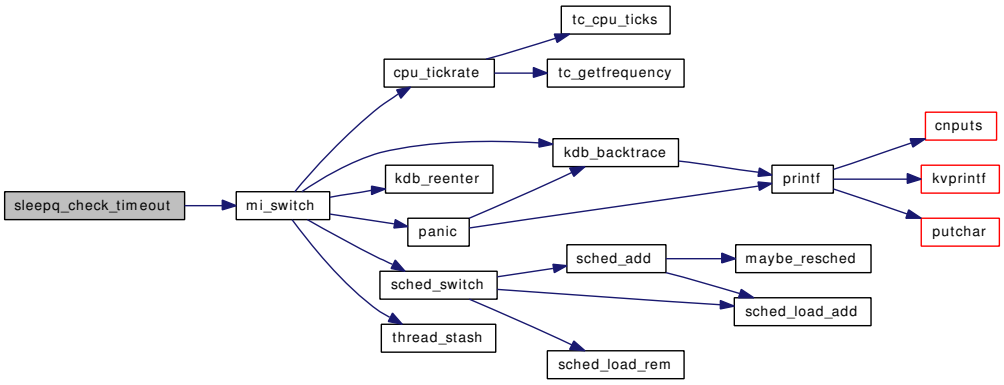
9.112.2.10 static int sleepq_check_timeout (void) [static]

Definition at line 473 of file subr_sleepqueue.c.

References mi_switch(), sched_lock, and td.

Referenced by sleepq_timedwait(), and sleepq_timedwait_sig().

Here is the call graph for this function:



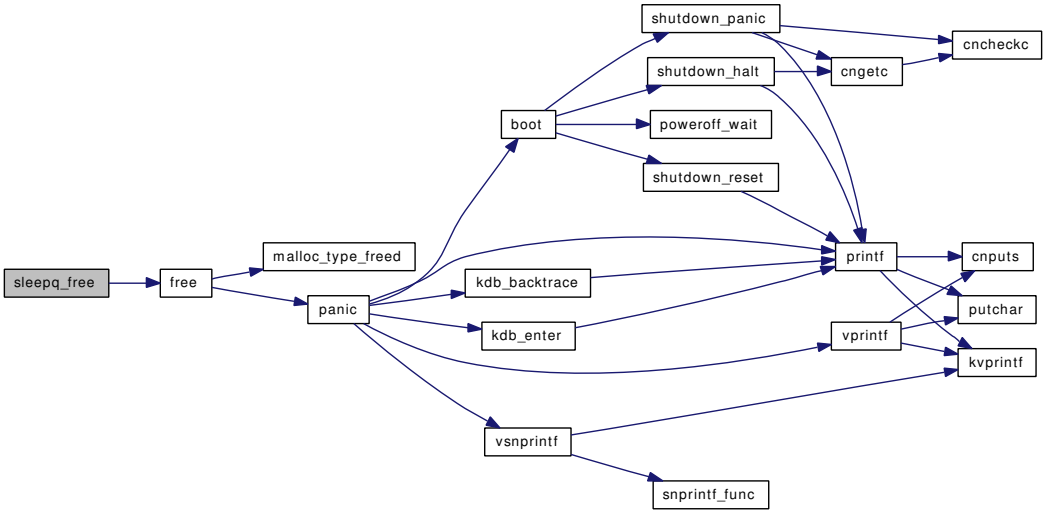
9.112.2.11 void sleepq_free (struct sleepqueue * *sq*)

Definition at line 210 of file subr_sleepqueue.c.

References free(), and NR_SLEEPQS.

Referenced by thread_fini().

Here is the call graph for this function:



9.112.2.12 void sleepq_lock (void * *wchan*)

Definition at line 224 of file subr_sleepqueue.c.

References SC_LOOKUP.

Referenced by cv_broadcastpri(), cv_destroy(), cv_signal(), cv_timedwait(), cv_timedwait_sig(), cv_wait_sig(), cv_wait_unlock(), msleep(), msleep_spin(), sleepq_remove(), sleepq_timeout(), wakeup(), and wakeup_one().

9.112.2.13 struct sleepqueue* sleepq_lookup (void * *wchan*)

Definition at line 238 of file subr_sleepqueue.c.

References SC_LOOKUP.

Referenced by cv_destroy(), sleepq_add(), sleepq_broadcast(), sleepq_catch_signals(), sleepq_remove(), sleepq_signal(), and sleepq_timeout().

9.112.2.14 void sleepq_release (void * *wchan*)

Definition at line 256 of file subr_sleepqueue.c.

References SC_LOOKUP.

Referenced by cv_broadcastpri(), cv_destroy(), cv_signal(), msleep_spin(), sleepq_broadcast(), sleepq_remove(), sleepq_signal(), sleepq_timedwait_sig(), sleepq_timeout(), and sleepq_wait_sig().

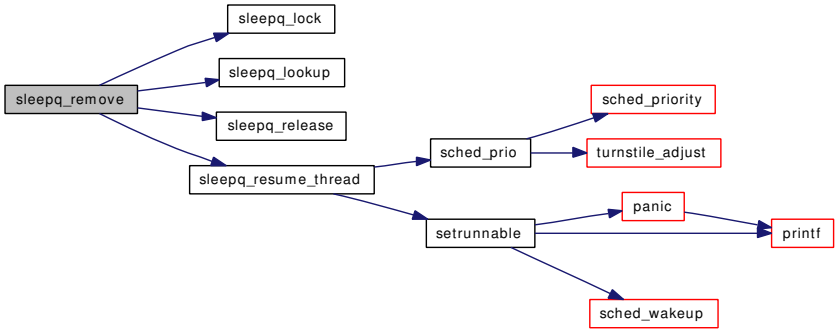
9.112.2.15 void sleepq_remove (struct thread * *td*, void * *wchan*)

Definition at line 811 of file subr_sleepqueue.c.

References sched_lock, sleepq_lock(), sleepq_lookup(), sleepq_release(), and sleepq_resume_thread().

Referenced by doselwakeup(), kse_wakeup(), msleep(), sleepq_abort(), speedup_syncer(), and syncer_-shutdown().

Here is the call graph for this function:



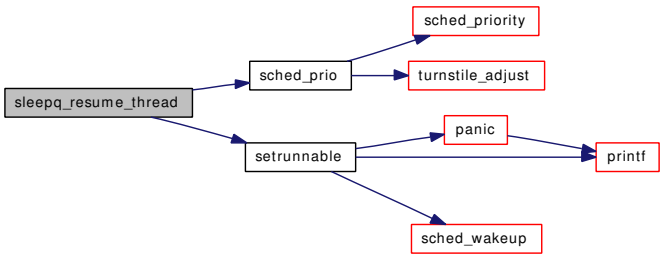
9.112.2.16 static void sleepq_resume_thread (struct sleepqueue * *sq*, struct thread * *td*, int *pri*) [static]

Definition at line 615 of file subr_sleepqueue.c.

References NR_SLEEPQS, SC_LOOKUP, sched_lock, sched_prio(), and setrunnable().

Referenced by sleepq_broadcast(), sleepq_catch_signals(), sleepq_remove(), sleepq_signal(), and sleepq_-timeout().

Here is the call graph for this function:



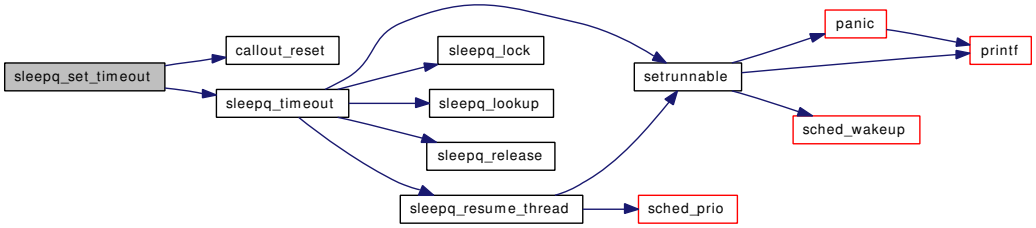
9.112.2.17 void sleepq_set_timeout (void * *wchan*, int *timo*)

Definition at line 346 of file subr_sleepqueue.c.

References callout_reset(), SC_LOOKUP, sleepq_timeout(), and td.

Referenced by cv_timedwait(), cv_timedwait_sig(), msleep(), and msleep_spin().

Here is the call graph for this function:



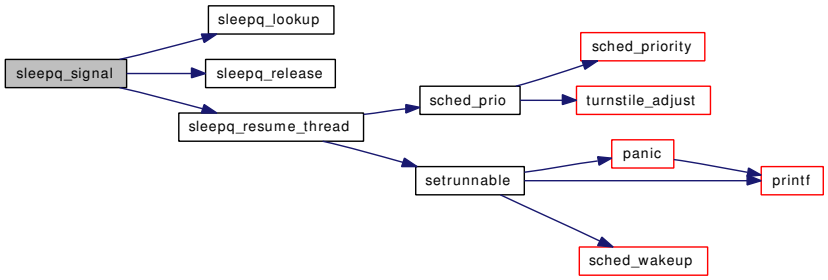
9.112.2.18 void sleepq_signal (void * *wchan*, int *flags*, int *pri*, int *queue*)

Definition at line 673 of file subr_sleepqueue.c.

References NR_SLEEPQS, sched_lock, sleepq_lookup(), sleepq_release(), sleepq_resume_thread(), and td.

Referenced by cv_signal(), and wakeup_one().

Here is the call graph for this function:



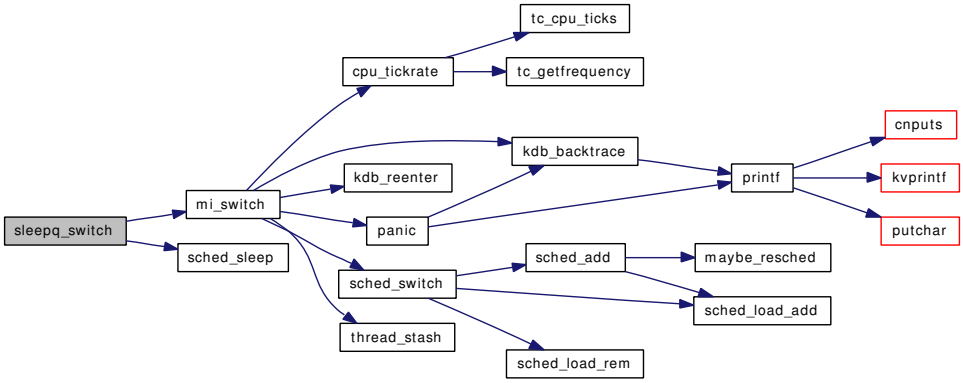
9.112.2.19 static void sleepq_switch (void * *wchan*) [static]

Definition at line 437 of file subr_sleepqueue.c.

References mi_switch(), SC_LOOKUP, sched_lock, sched_sleep(), and td.

Referenced by sleepq_timedwait(), sleepq_timedwait_sig(), sleepq_wait(), and sleepq_wait_sig().

Here is the call graph for this function:



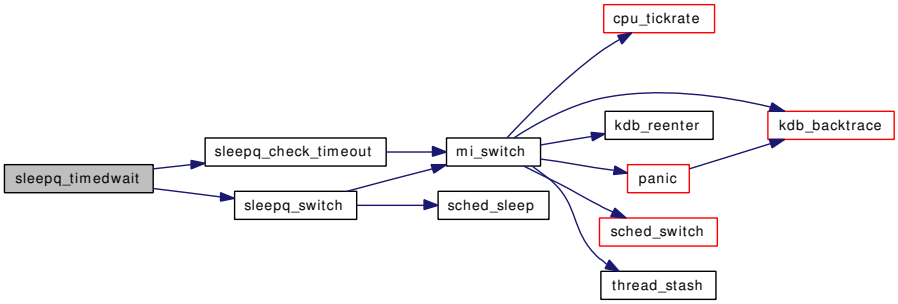
9.112.2.20 int sleepq_timedwait (void * *wchan*)

Definition at line 574 of file subr_sleepqueue.c.

References sched_lock, sleepq_check_timeout(), and sleepq_switch().

Referenced by cv_timedwait(), msleep(), and msleep_spin().

Here is the call graph for this function:



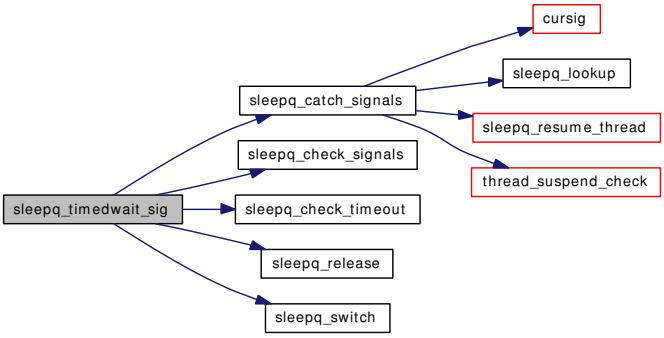
9.112.2.21 int sleepq_timedwait_sig (void * *wchan*)

Definition at line 591 of file subr_sleepqueue.c.

References sched_lock, sleepq_catch_signals(), sleepq_check_signals(), sleepq_check_timeout(), sleepq_-release(), and sleepq_switch().

Referenced by cv_timedwait_sig(), and msleep().

Here is the call graph for this function:



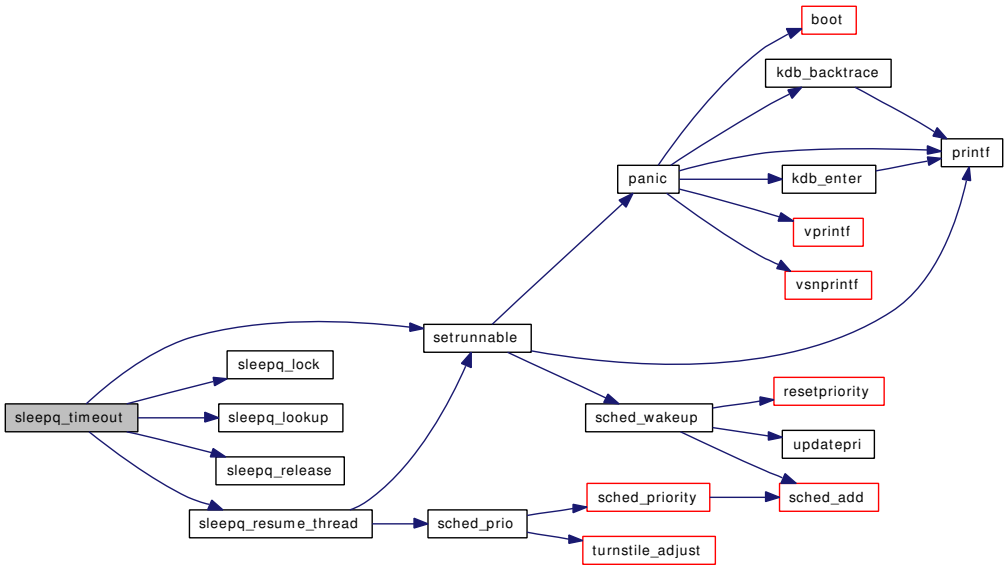
9.112.2.22 static void sleepq_timeout (void * *arg*) [static]

Definition at line 740 of file subr_sleepqueue.c.

References sched_lock, setrunnable(), sleepq_lock(), sleepq_lookup(), sleepq_release(), sleepq_resume_-thread(), and td.

Referenced by sleepq_set_timeout().

Here is the call graph for this function:



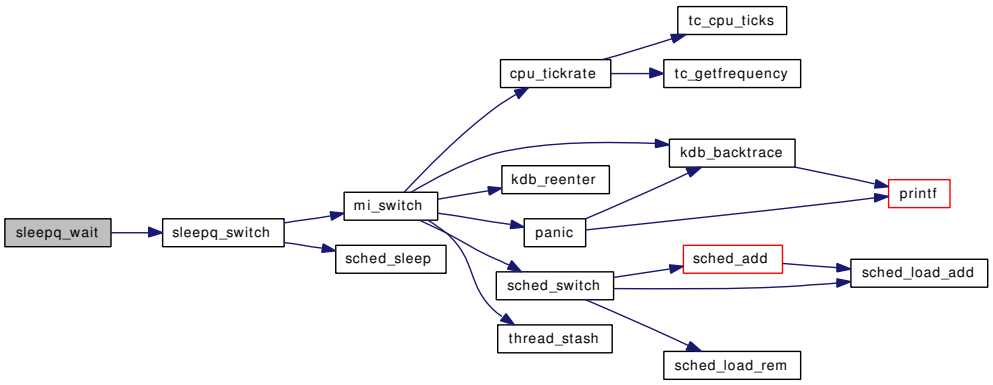
9.112.2.23 void sleepq_wait (void * *wchan*)

Definition at line 538 of file subr_sleepqueue.c.

References sched_lock, and sleepq_switch().

Referenced by cv_wait_unlock(), msleep(), and msleep_spin().

Here is the call graph for this function:



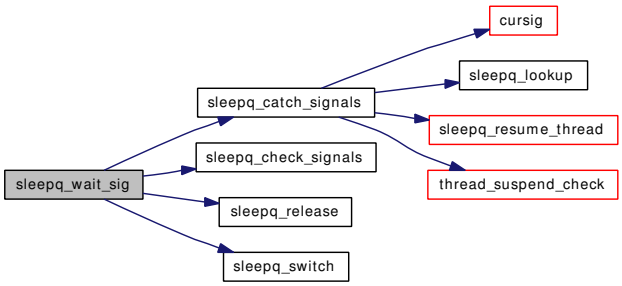
9.112.2.24 int sleepq_wait_sig (void * *wchan*)

Definition at line 552 of file subr_sleepqueue.c.

References sched_lock, sleepq_catch_signals(), sleepq_check_signals(), sleepq_release(), and sleepq_switch().

Referenced by cv_wait_sig(), and msleep().

Here is the call graph for this function:



9.112.3 Variable Documentation

9.112.3.1 struct sleepqueue_chain sleepq_chains[SC_TABLESIZE] [static]

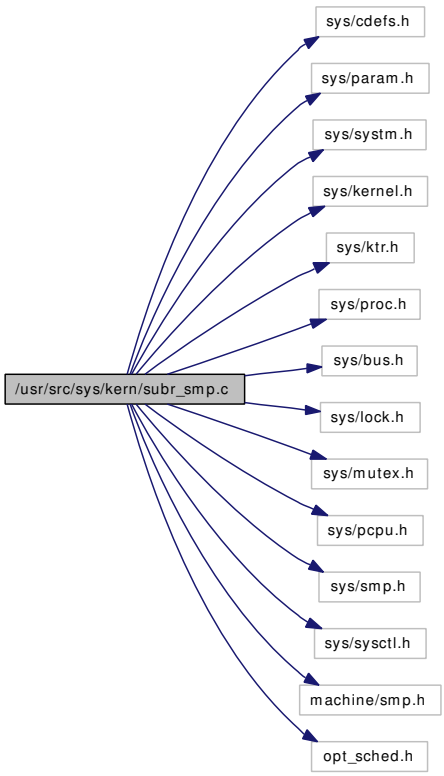
Definition at line 144 of file subr_sleepqueue.c.

Referenced by init_sleepqueues().

9.113 /usr/src/sys/kern/subr_smp.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/ktr.h>
#include <sys/proc.h>
#include <sys/bus.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/pcpu.h>
#include <sys/smp.h>
#include <sys/sysctl.h>
#include <machine/smp.h>
#include "opt_sched.h"
```

Include dependency graph for subr_smp.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_smp.c,v 1.197 2005/10/24 21:04:17 jhb Exp $")
- SYSCTL_NODE (_kern, OID_AUTO, smp, CTLFLAG_RD, NULL,"Kernel SMP")
- SYSCTL_INT (_kern_smp, OID_AUTO, maxcpus, CTLFLAG_RD,&mp_maxcpus, 0,"Max number of CPUs that the system was compiled for.")
- SYSCTL_INT (_kern_smp, OID_AUTO, active, CTLFLAG_RW,&smp_active, 0,"Number of Auxillary Processors (APs) that were successfully started")
- SYSCTL_INT (_kern_smp, OID_AUTO, disabled, CTLFLAG_RDTUN,&smp_disabled, 0,"SMP has been disabled from the loader")
- TUNABLE_INT ("kern.smp.disabled",&smp_disabled)
- SYSCTL_INT (_kern_smp, OID_AUTO, cpus, CTLFLAG_RD,&smp_cpus, 0,"Number of CPUs online")
- static void mp_setvariables_for_up (void *dummy)
- SYSINIT (cpu_mp_setvariables, SI_SUB_TUNABLES, SI_ORDER_FIRST, mp_setvariables_for_up, NULL)

Variables

- cpumask_t all_cpus
- int mp_ncpus
- int mp_maxcpus = MAXCPU
- cpu_top * smp_topology
- volatile int smp_started
- u_int mp_maxid
- int smp_active = 0
- int smp_disabled = 0
- int smp_cpus = 1

9.113.1 Function Documentation

9.113.1.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_smp. *c*, v 1.197 2005/10/24 21:04:17 jhb Exp $")

9.113.1.2 static void mp_setvariables_for_up (void * *dummy*) [static]

Definition at line 365 of file subr_smp.c.

References all_cpus, mp_maxid, and mp_ncpus.

9.113.1.3 SYSCTL_INT (_kern_smp, OID_AUTO, cpus, CTLFLAG_RD, & *smp_cpus*, 0, "Number of CPUs online")

9.113.1.4 SYSCTL_INT (_kern_smp, OID_AUTO, disabled, CTLFLAG_RDTUN, & *smp_disabled*, 0, "SMP has been disabled from the loader")

9.113.1.5 SYSCTL_INT (_kern_smp, OID_AUTO, active, CTLFLAG_RW, & *smp_active*, 0, "Number of Auxillary Processors (APs) that were successfully started")

9.113.1.6 SYSCTL_INT (_kern_smp, OID_AUTO, maxcpus, CTLFLAG_RD, & *mp_maxcpus*, 0, "Max number of CPUs that the system was compiled for.")

9.113.1.7 SYSCTL_NODE (_kern, OID_AUTO, smp, CTLFLAG_RD, NULL, "Kernel SMP")

9.113.1.8 SYSINIT (cpu_mp_setvariables, SI_SUB_TUNABLES, SI_ORDER_FIRST, mp_setvariables_for_up, NULL)

Definition at line 372 of file subr_smp.c.

9.113.1.9 TUNABLE_INT ("kern.smp.disabled", & *smp_disabled*)

9.113.2 Variable Documentation

9.113.2.1 cpumask_t all_cpus

Definition at line 64 of file subr_smp.c.

Referenced by mp_setvariables_for_up().

9.113.2.2 int mp_maxcpus = MAXCPU

Definition at line 68 of file subr_smp.c.

9.113.2.3 u_int mp_maxid

Definition at line 72 of file subr_smp.c.

Referenced by mp_setvariables_for_up().

9.113.2.4 int mp_ncpus

Definition at line 66 of file subr_smp.c.

Referenced by kse_create(), mp_setvariables_for_up(), sched_setup(), and TAILQ_HEAD().

9.113.2.5 int smp_active = 0

Definition at line 79 of file subr_smp.c.

9.113.2.6 int smp_cpus = 1

Definition at line 88 of file subr_smp.c.

Referenced by _do_lock_normal(), and sched_timeslice_split().

9.113.2.7 int smp_disabled = 0

Definition at line 83 of file subr_smp.c.

9.113.2.8 volatile int smp_started

Definition at line 71 of file subr_smp.c.

9.113.2.9 struct cpu_top* smp_topology

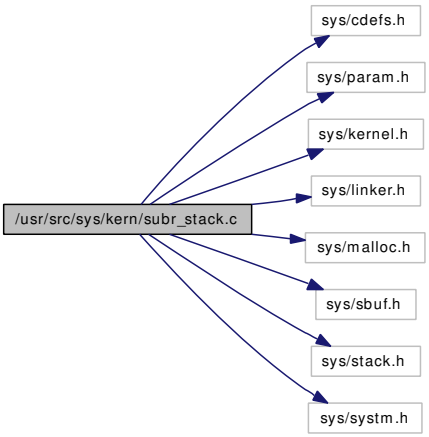
Definition at line 70 of file subr_smp.c.

Referenced by sched_setup().

9.114 /usr/src/sys/kern/subr_stack.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/kernel.h>
#include <sys/linker.h>
#include <sys/malloc.h>
#include <sys/sbuf.h>
#include <sys/stack.h>
#include <sys/systm.h>
```

Include dependency graph for subr_stack.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_stack.c,v 1.3 2006/05/28 22:15:28 kris Exp $")
- MALLOC_DEFINE (M_STACK,"stack","Stack Traces")
- static void stack_symbol (vm_offset_t pc, const char **name, long *offset)
- stack * stack_create (void)
- void stack_destroy (struct stack *st)
- int stack_put (struct stack *st, vm_offset_t pc)
- void stack_copy (struct stack *src, struct stack *dst)
- void stack_zero (struct stack *st)
- void stack_print (struct stack *st)
- void stack_sbuf_print (struct sbuf *sb, struct stack *st)

9.114.1 Function Documentation

9.114.1.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_stack. *c*, v 1.3 2006/05/28 22:15:28 kris Exp $")

9.114.1.2 MALLOC_DEFINE (M_STACK, "stack", "Stack Traces")

9.114.1.3 void stack_copy (struct stack ∗ *src*, struct stack ∗ *dst*)

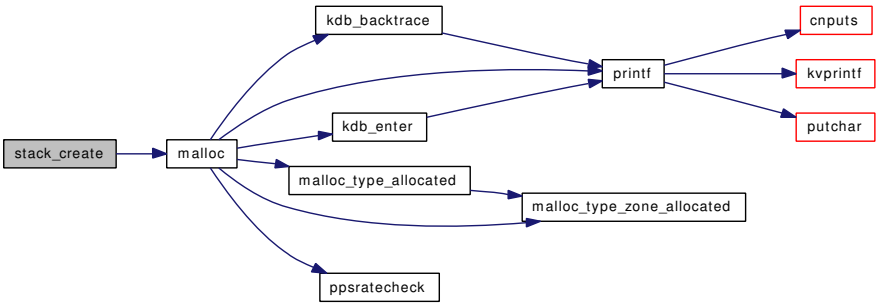
Definition at line 73 of file subr_stack.c.

9.114.1.4 struct stack∗ stack_create (void)

Definition at line 46 of file subr_stack.c.

References malloc().

Here is the call graph for this function:

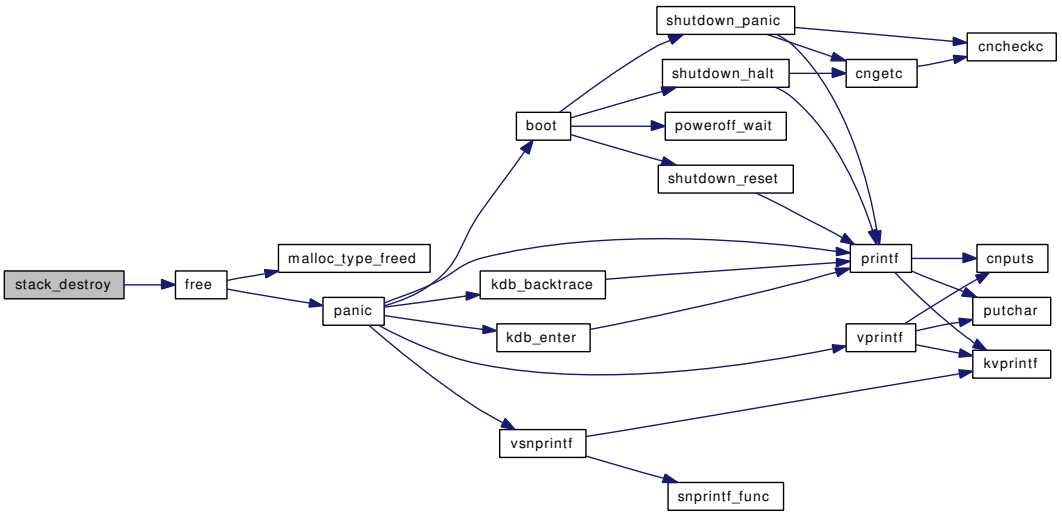


9.114.1.5 void stack_destroy (struct stack ∗ *st*)

Definition at line 55 of file subr_stack.c.

References free().

Here is the call graph for this function:



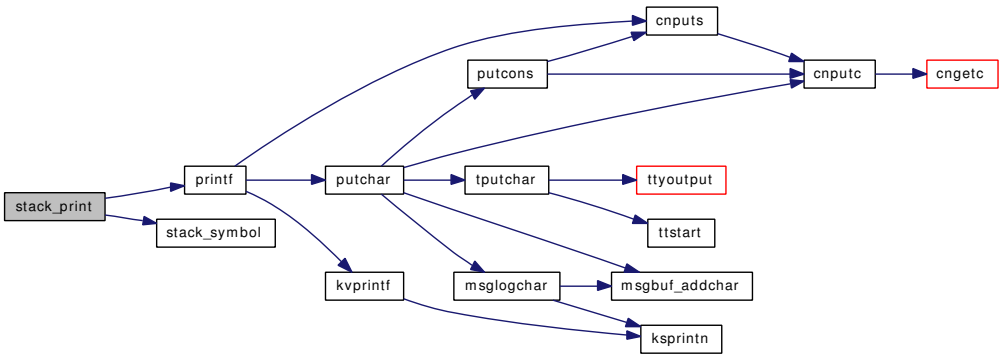
9.114.1.6 void stack_print (struct stack * *st*)

Definition at line 87 of file subr_stack.c.

References printf(), and stack_symbol().

Referenced by lockmgr_printinfo().

Here is the call graph for this function:



9.114.1.7 int stack_put (struct stack * *st*, vm_offset_t *pc*)

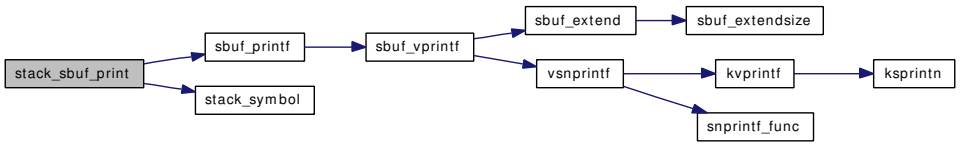
Definition at line 62 of file subr_stack.c.

9.114.1.8 void stack_sbuf_print (struct sbuf * *sb*, struct stack * *st*)

Definition at line 102 of file subr_stack.c.

References sbuf_printf(), and stack_symbol().

Here is the call graph for this function:



9.114.1.9 static void stack_symbol (vm_offset_t *pc*, const char ** *name*, long * *offset*) [static]

Definition at line 153 of file subr_stack.c.

Referenced by stack_print(), and stack_sbuf_print().

9.114.1.10 void stack_zero (struct stack * *st*)

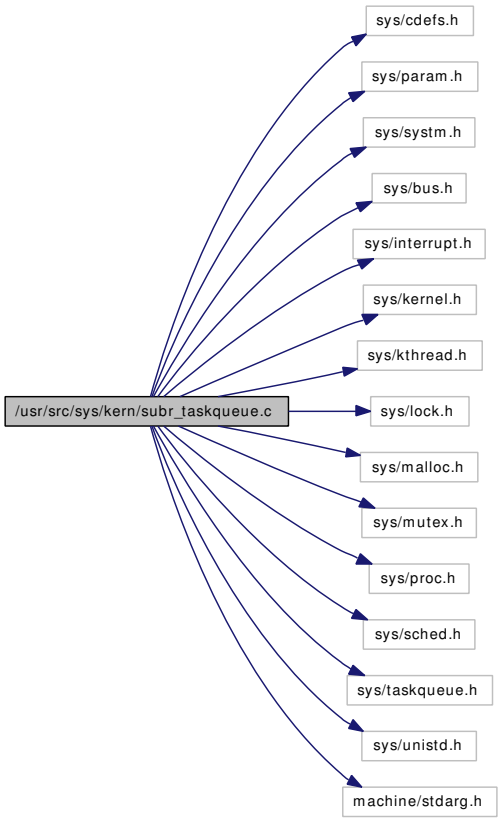
Definition at line 80 of file subr_stack.c.

Referenced by lockinit().

9.115 /usr/src/sys/kern/subr_taskqueue.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/bus.h>
#include <sys/interrupt.h>
#include <sys/kernel.h>
#include <sys/kthread.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/sched.h>
#include <sys/taskqueue.h>
#include <sys/unistd.h>
#include <machine/stdarg.h>
```

Include dependency graph for subr_taskqueue.c:



Defines

- #define TQ_FLAGS_ACTIVE (1 << 0)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_taskqueue.c,v 1.38 2007/01/23 08:46:50 jeff Exp $")
- static MALLOC_DEFINE (M_TASKQUEUE,"taskqueue","Task Queues")
- static STAILQ_HEAD (taskqueue_list, taskqueue)
- static __inline void TQ_LOCK (struct taskqueue *tq)
- static __inline void TQ_UNLOCK (struct taskqueue *tq)
- static void init_taskqueue_list (void *data)
- static __inline int TQ_SLEEP (struct taskqueue *tq, void *p, struct mtx *m, int pri, const char *wm, int t)
- static void init_taskqueue_list (void *data __unused)
- SYSINIT (taskqueue_list, SI_SUB_INTRINSIC, SI_ORDER_ANY, init_taskqueue_list, NULL)
- static struct taskqueue * _taskqueue_create (const char *name, int mflags, taskqueue_enqueue_fn enqueue, void *context, int mtxflags, const char *mtxname)
- taskqueue * taskqueue_create (const char *name, int mflags, taskqueue_enqueue_fn enqueue, void *context)
- static void taskqueue_terminate (struct proc **pp, struct taskqueue *tq)
- void taskqueue_free (struct taskqueue *queue)
- taskqueue * taskqueue_find (const char *name)
- int taskqueue_enqueue (struct taskqueue *queue, struct task *task)
- void taskqueue_run (struct taskqueue *queue)
- void taskqueue_drain (struct taskqueue *queue, struct task *task)
- static void taskqueue_swi_enqueue (void *context)
- static void taskqueue_swi_run (void *dummy)
- static void taskqueue_swi_giant_enqueue (void *context)
- static void taskqueue_swi_giant_run (void *dummy)
- int taskqueue_start_threads (struct taskqueue **tqp, int count, int pri, const char *name,...)
- void taskqueue_thread_loop (void *arg)
- void taskqueue_thread_enqueue (void *context)
- TASKQUEUE_DEFINE (swi, taskqueue_swi_enqueue, 0, swi_add(NULL,"task queue", taskqueue_swi_run, NULL, SWI_TQ, INTR_MPSAFE,&taskqueue_ih))
- TASKQUEUE_DEFINE (swi_giant, taskqueue_swi_giant_enqueue, 0, swi_add(NULL,"Giant taskq", taskqueue_swi_giant_run, NULL, SWI_TQ_GIANT, 0,&taskqueue_giant_ih))
- TASKQUEUE_DEFINE_THREAD (thread)
- taskqueue * taskqueue_create_fast (const char *name, int mflags, taskqueue_enqueue_fn enqueue, void *context)
- int taskqueue_enqueue_fast (struct taskqueue *queue, struct task *task)
- static void taskqueue_fast_enqueue (void *context)
- static void taskqueue_fast_run (void *dummy)
- TASKQUEUE_FAST_DEFINE (fast, taskqueue_fast_enqueue, 0, swi_add(NULL,"Fast task queue", taskqueue_fast_run, NULL, SWI_TQ_FAST, INTR_MPSAFE,&taskqueue_fast_ih))

Variables

- static void * taskqueue_giant_ih
- static void * taskqueue_ih
- static void * taskqueue_fast_ih

9.115.1 Define Documentation

9.115.1.1 #define TQ_FLAGS_ACTIVE (1 << 0)

Definition at line 65 of file subr_taskqueue.c.

Referenced by _taskqueue_create(), taskqueue_free(), and taskqueue_thread_loop().

9.115.2 Function Documentation

9.115.2.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_taskqueue. *c*, v 1.38 2007/01/23 08:46:50 jeff Exp $")

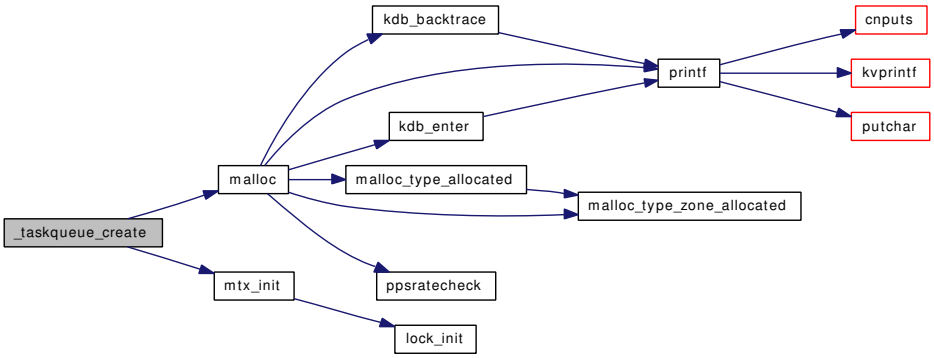
9.115.2.2 static struct taskqueue* _taskqueue_create (const char * *name*, int *mflags*, taskqueue_enqueue_fn *enqueue*, void * *context*, int *mtxflags*, const char * *mtxname*) [static]

Definition at line 107 of file subr_taskqueue.c.

References malloc(), mtx_init(), and TQ_FLAGS_ACTIVE.

Referenced by taskqueue_create(), and taskqueue_create_fast().

Here is the call graph for this function:

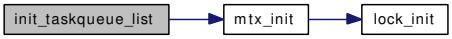


9.115.2.3 static void init_taskqueue_list (void *data *__unused*) [static]

Definition at line 97 of file subr_taskqueue.c.

References mtx_init().

Here is the call graph for this function:



9.115.2.4 static void init_taskqueue_list (void * *data*) [static]

9.115.2.5 static MALLOC_DEFINE (M_TASKQUEUE, "taskqueue", "Task Queues") [static]

9.115.2.6 static STAILQ_HEAD (taskqueue_list, taskqueue) [static]

Definition at line 48 of file subr_taskqueue.c.

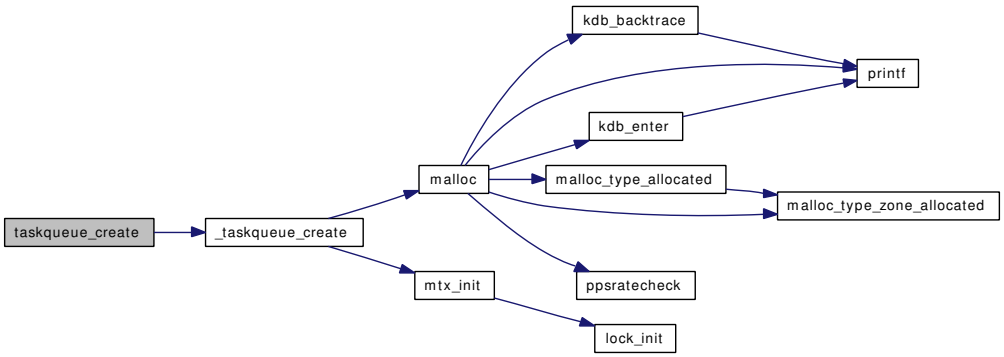
9.115.2.7 SYSINIT (taskqueue_list, SI_SUB_INTRINSIC, SI_ORDER_ANY, init_taskqueue_list, NULL)

9.115.2.8 struct taskqueue* taskqueue_create (const char * *name*, int *mflags*, taskqueue_enqueue_fn *enqueue*, void * *context*)

Definition at line 133 of file subr_taskqueue.c.

References _taskqueue_create().

Here is the call graph for this function:

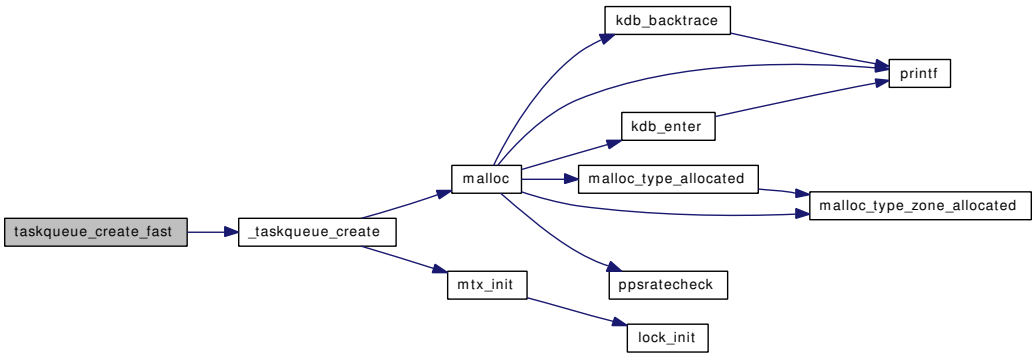


9.115.2.9 struct taskqueue* taskqueue_create_fast (const char * *name*, int *mflags*, taskqueue_enqueue_fn *enqueue*, void * *context*)

Definition at line 408 of file subr_taskqueue.c.

References _taskqueue_create().

Here is the call graph for this function:



9.115.2.10 TASKQUEUE_DEFINE (swi_giant, taskqueue_swi_giant_enqueue, 0, swi_add(NULL,"Giant taskq", taskqueue_swi_giant_run, NULL, SWI_TQ_GIANT, 0,&taskqueue_giant_ih))

9.115.2.11 TASKQUEUE_DEFINE (swi, taskqueue_swi_enqueue, 0, swi_add(NULL,"task queue", taskqueue_swi_run, NULL, SWI_TQ, INTR_MPSAFE,&taskqueue_ih))

9.115.2.12 TASKQUEUE_DEFINE_THREAD (thread)

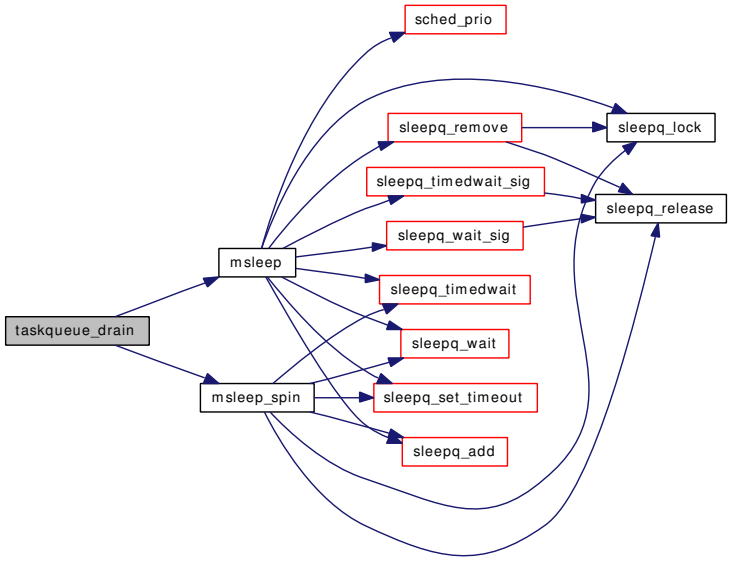
9.115.2.13 void taskqueue_drain (struct taskqueue * *queue*, struct task * *task*)

Definition at line 271 of file subr_taskqueue.c.

References msleep(), and msleep_spin().

Referenced by aio_proc_rundown(), and firmware_modevent().

Here is the call graph for this function:



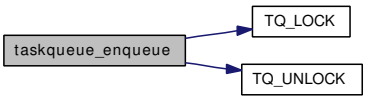
9.115.2.14 int taskqueue_enqueue (struct taskqueue * *queue*, struct task * *task*)

Definition at line 191 of file subr_taskqueue.c.

References TQ_LOCK(), and TQ_UNLOCK().

Referenced by aio_kick_nowait(), aio_physwakeup(), do_unlink(), firmware_modevent(), firmware_put(), kqueue_schedtask(), power_pm_suspend(), prison_free(), taskqueue_enqueue_fast(), and uipc_detach().

Here is the call graph for this function:



9.115.2.15 int taskqueue_enqueue_fast (struct taskqueue * *queue*, struct task * *task*)

Definition at line 417 of file subr_taskqueue.c.

References taskqueue_enqueue().

Here is the call graph for this function:

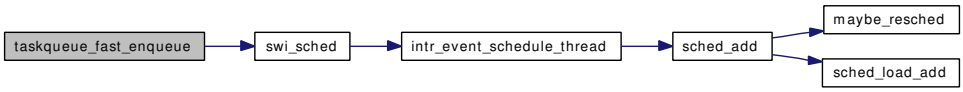
9.115.2.16 TASKQUEUE_FAST_DEFINE (fast, taskqueue_fast_enqueue, 0, swi_add(NULL,"Fast task queue", taskqueue_fast_run, NULL, SWI_TQ_FAST, INTR_MPSAFE,&taskqueue_fast_ih))

9.115.2.17 static void taskqueue_fast_enqueue (void * *context*) [static]

Definition at line 425 of file subr_taskqueue.c.

References swi_sched().

Here is the call graph for this function:

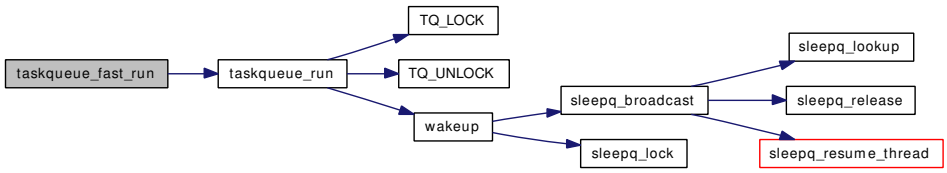


9.115.2.18 static void taskqueue_fast_run (void * *dummy*) [static]

Definition at line 431 of file subr_taskqueue.c.

References taskqueue_run().

Here is the call graph for this function:



9.115.2.19 struct taskqueue* taskqueue_find (const char * *name*)

Definition at line 174 of file subr_taskqueue.c.

References TQ_LOCK().

Here is the call graph for this function:

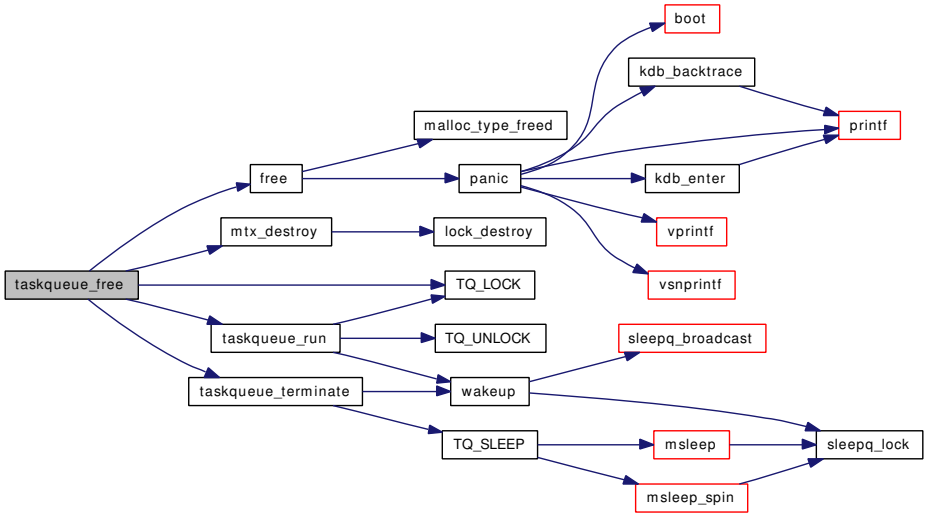
9.115.2.20 void taskqueue_free (struct taskqueue * *queue*)

Definition at line 154 of file subr_taskqueue.c.

References free(), mtx_destroy(), taskqueue_run(), taskqueue_terminate(), TQ_FLAGS_ACTIVE, and TQ_LOCK().

Referenced by aio_unload().

Here is the call graph for this function:



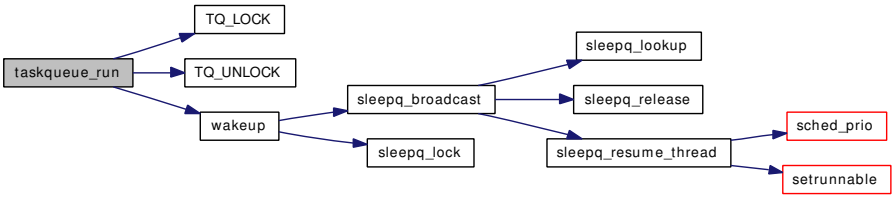
9.115.2.21 void taskqueue_run (struct taskqueue * *queue*)

Definition at line 235 of file subr_taskqueue.c.

References TQ_LOCK(), TQ_UNLOCK(), and wakeup().

Referenced by taskqueue_fast_run(), taskqueue_free(), taskqueue_swi_giant_run(), taskqueue_swi_run(), and taskqueue_thread_loop().

Here is the call graph for this function:

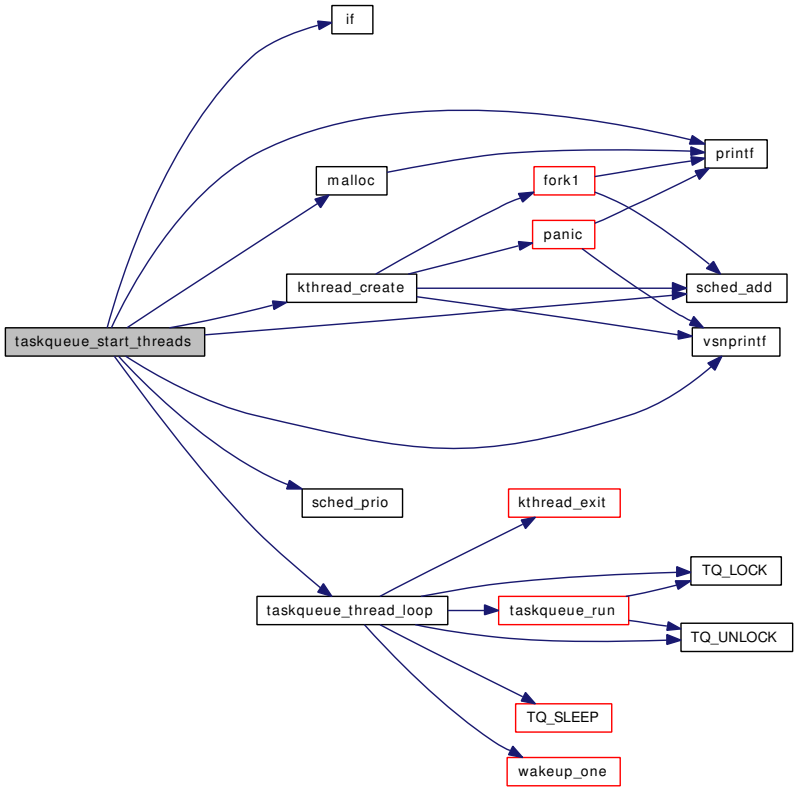


9.115.2.22 int taskqueue_start_threads (struct taskqueue ** *tqp*, int *count*, int *pri*, const char * *name*, ...)

Definition at line 313 of file subr_taskqueue.c.

References if(), kthread_create(), malloc(), printf(), sched_add(), sched_lock, sched_prio(), taskqueue_-thread_loop(), td, and vsnprintf().

Here is the call graph for this function:

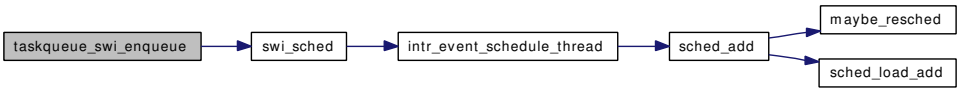


9.115.2.23 static void taskqueue_swi_enqueue (void * *context*) [static]

Definition at line 289 of file subr_taskqueue.c.

References swi_sched(), and taskqueue_ih.

Here is the call graph for this function:

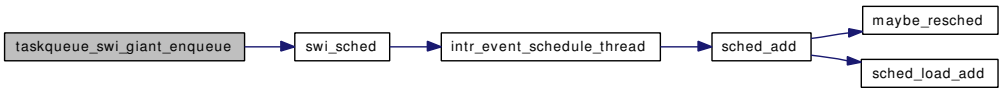


9.115.2.24 static void taskqueue_swi_giant_enqueue (void * *context*) [static]

Definition at line 301 of file subr_taskqueue.c.

References swi_sched(), and taskqueue_giant_ih.

Here is the call graph for this function:

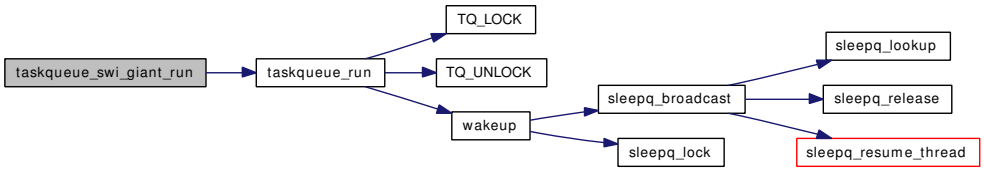


9.115.2.25 static void taskqueue_swi_giant_run (void * *dummy*) [static]

Definition at line 307 of file subr_taskqueue.c.

References taskqueue_run().

Here is the call graph for this function:

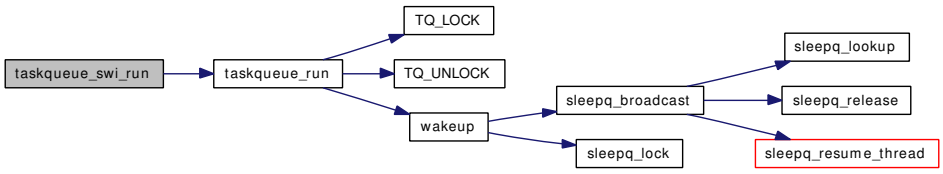


9.115.2.26 static void taskqueue_swi_run (void * *dummy*) [static]

Definition at line 295 of file subr_taskqueue.c.

References taskqueue_run().

Here is the call graph for this function:



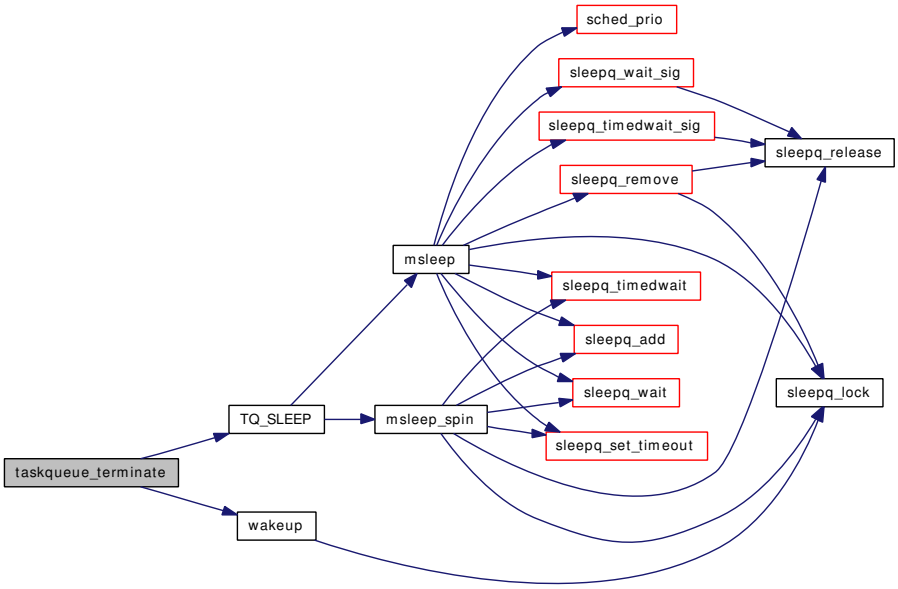
9.115.2.27 static void taskqueue_terminate (struct proc ** *pp*, struct taskqueue * *tq*) [static]

Definition at line 144 of file subr_taskqueue.c.

References TQ_SLEEP(), and wakeup().

Referenced by taskqueue_free().

Here is the call graph for this function:

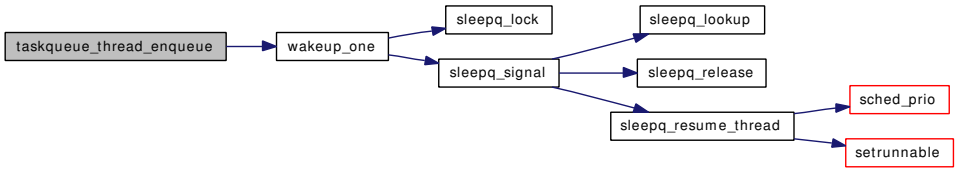


9.115.2.28 void taskqueue_thread_enqueue (void * *context*)

Definition at line 386 of file subr_taskqueue.c.

References wakeup_one().

Here is the call graph for this function:



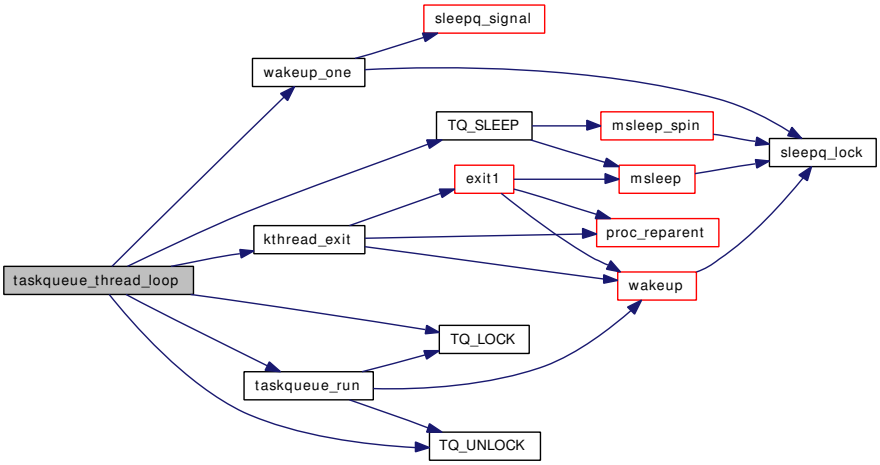
9.115.2.29 void taskqueue_thread_loop (void * *arg*)

Definition at line 366 of file subr_taskqueue.c.

References kthread_exit(), taskqueue_run(), TQ_FLAGS_ACTIVE, TQ_LOCK(), TQ_SLEEP(), TQ_-UNLOCK(), and wakeup_one().

Referenced by taskqueue_start_threads().

Here is the call graph for this function:



9.115.2.30 static __inline void TQ_LOCK (struct taskqueue * *tq*) [static]

Definition at line 68 of file subr_taskqueue.c.

Referenced by taskqueue_enqueue(), taskqueue_find(), taskqueue_free(), taskqueue_run(), and taskqueue_thread_loop().

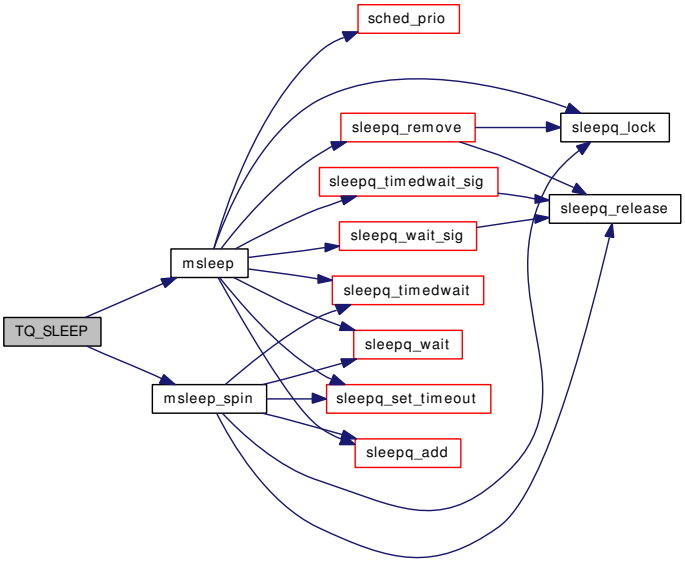
9.115.2.31 static __inline int TQ_SLEEP (struct taskqueue * *tq*, void * *p*, struct mtx * *m*, int *pri*, const char * *wm*, int *t*) [static]

Definition at line 88 of file subr_taskqueue.c.

References msleep(), and msleep_spin().

Referenced by taskqueue_terminate(), and taskqueue_thread_loop().

Here is the call graph for this function:



9.115.2.32 static __inline void TQ_UNLOCK (struct taskqueue * *tq*) [static]

Definition at line 77 of file subr_taskqueue.c.

Referenced by taskqueue_enqueue(), taskqueue_run(), and taskqueue_thread_loop().

9.115.3 Variable Documentation

9.115.3.1 void* taskqueue_fast_ih [static]

Definition at line 422 of file subr_taskqueue.c.

9.115.3.2 void* taskqueue_giant_ih [static]

Definition at line 46 of file subr_taskqueue.c.

Referenced by taskqueue_swi_giant_enqueue().

9.115.3.3 void* taskqueue_ih [static]

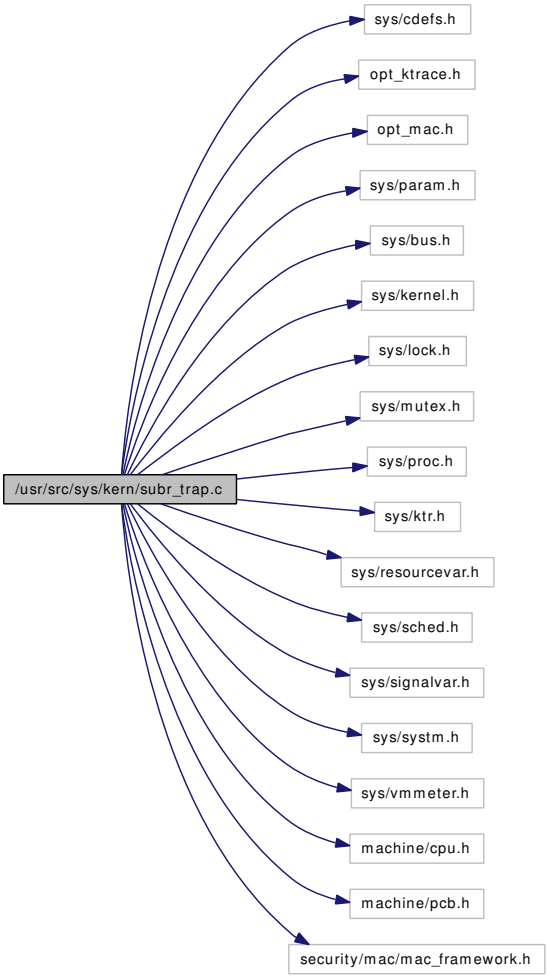
Definition at line 47 of file subr_taskqueue.c.

Referenced by taskqueue_swi_enqueue().

9.116 /usr/src/sys/kern/subr_trap.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ktrace.h"
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/bus.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/ktr.h>
#include <sys/resourcevar.h>
#include <sys/sched.h>
#include <sys/signalvar.h>
#include <sys/systm.h>
#include <sys/vmmeter.h>
#include <machine/cpu.h>
#include <machine/pcb.h>
#include <security/mac/mac_framework.h>
```

Include dependency graph for subr_trap.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_trap.c,v 1.290 2006/12/06 06:34:55 julian Exp $")
- void userret (struct thread *td, struct trapframe *frame)
- void ast (struct trapframe *framep)

9.116.1 Function Documentation

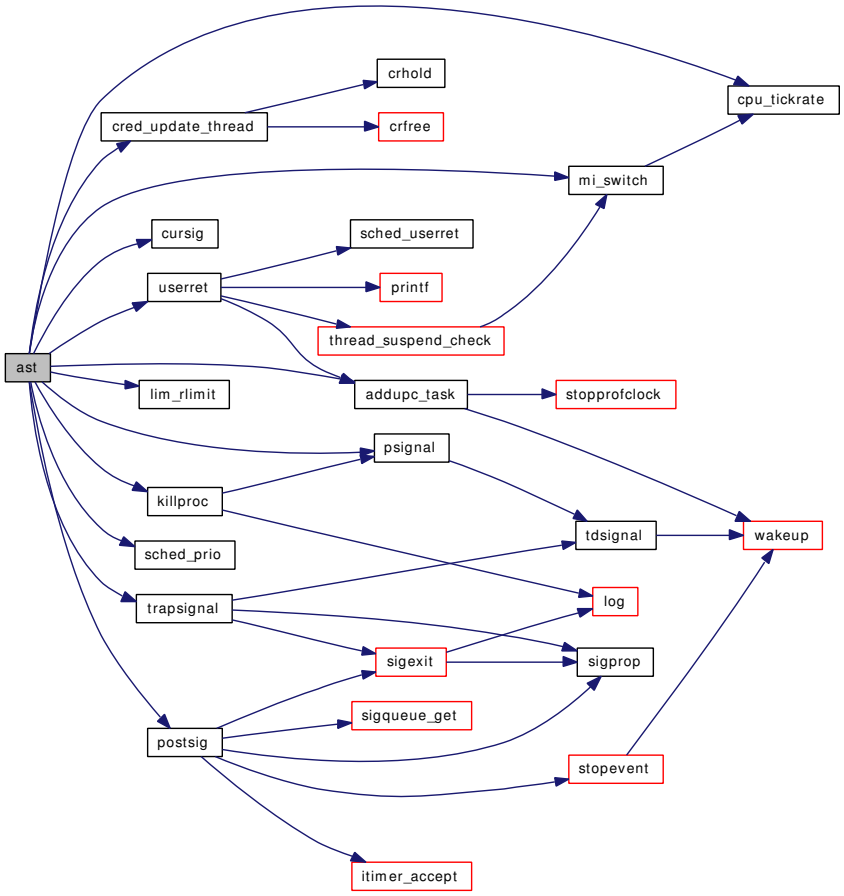
9.116.1.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_trap. *c*, v 1.290 2006/12/06 06:34:55 julian Exp $")

9.116.1.2 void ast (struct trapframe * *framep*)

Definition at line 148 of file subr_trap.c.

References addupc_task(), cpu_tickrate(), cred_update_thread(), cursig(), Giant, killproc(), lim_rlimit(), mi_switch(), postsig(), psignal(), sched_lock, sched_prio(), td, trapsignal(), and userret().

Here is the call graph for this function:



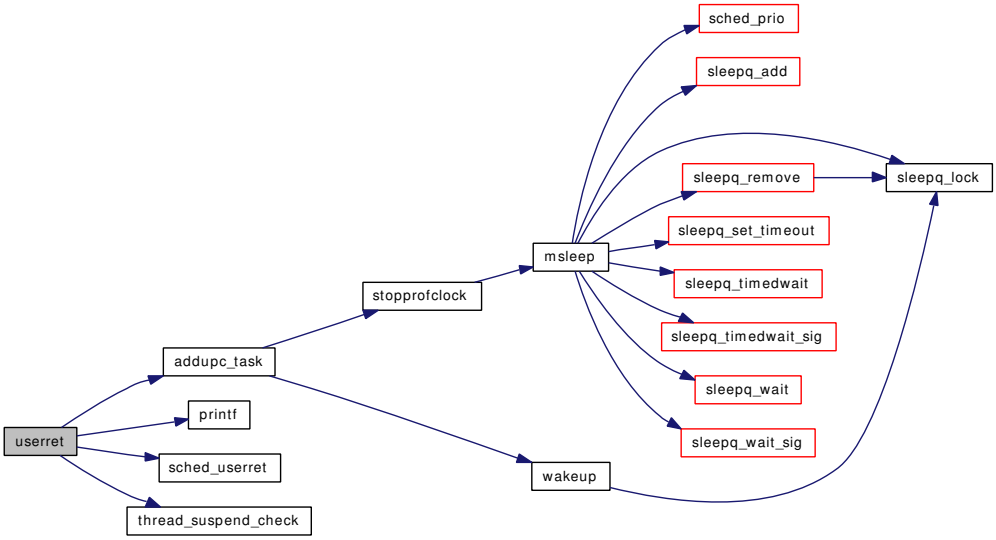
9.116.1.3 void userret (struct thread * *td*, struct trapframe * *frame*)

Definition at line 78 of file subr_trap.c.

References addupc_task(), printf(), psratio, sched_lock, sched_userret(), and thread_suspend_check().

Referenced by ast(), and fork_return().

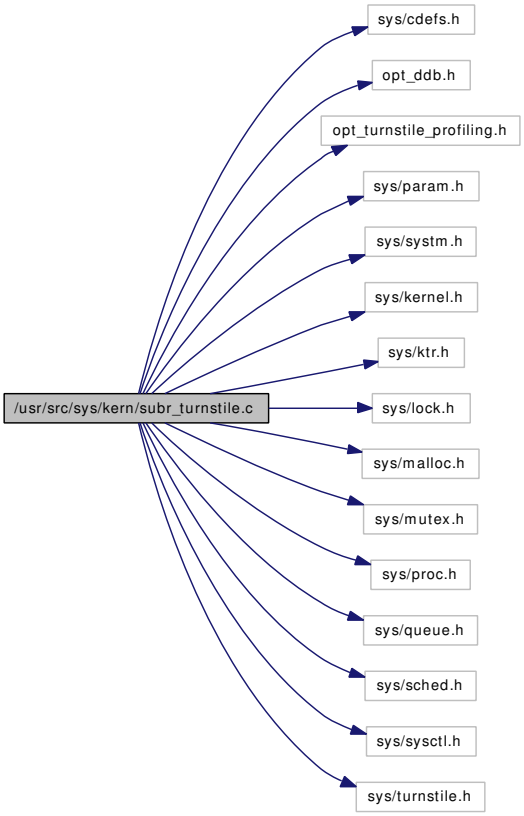
Here is the call graph for this function:


sched_prio
sleepq_add
sleepq_remove
sleepq_lock
sleepq_set_timeout
msleep
sleepq_timedwait
stopprofclock
sleepq_timedwait_sig
addupc_task
sleepq_wait
printf
sleepq_wait_sig
userret
sched_userret
wakeup
thread_suspend_check

9.117 /usr/src/sys/kern/subr_turnstile.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ddb.h"
#include "opt_turnstile_profiling.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/ktr.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/queue.h>
#include <sys/sched.h>
#include <sys/sysctl.h>
#include <sys/turnstile.h>
```

Include dependency graph for subr_turnstile.c:



Data Structures

- struct turnstile
- struct turnstile_chain

Defines

- #define TC_TABLESIZE 128
- #define TC_MASK (TC_TABLESIZE - 1)
- #define TC_SHIFT 8
- #define TC_HASH(lock) (((uintptr_t)(lock) >> TC_SHIFT) & TC_MASK)
- #define TC_LOOKUP(lock) &turnstile_chains[TC_HASH(lock)]

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_turnstile.c,v 1.166 2007/01/23 08:46:50 jeff Exp $")
- static MALLOC_DEFINE (M_TURNSTILE,"turnstiles","turnstiles")
- static void init_turnstile0 (void *dummy)
- static void propagate_priority (struct thread *td)
- static int turnstile_adjust_thread (struct turnstile *ts, struct thread *td)
- static struct thread * turnstile_first_waiter (struct turnstile *ts)
- static void turnstile_setowner (struct turnstile *ts, struct thread *owner)
- void init_turnstiles (void)
- SYSINIT (turnstile0, SI_SUB_LOCK, SI_ORDER_ANY, init_turnstile0, NULL)
- void turnstile_adjust (struct thread *td, u_char oldpri)
- turnstile * turnstile_alloc (void)
- void turnstile_free (struct turnstile *ts)
- void turnstile_lock (struct lock_object *lock)
- turnstile * turnstile_lookup (struct lock_object *lock)
- void turnstile_release (struct lock_object *lock)
- void turnstile_claim (struct lock_object *lock)
- void turnstile_wait (struct lock_object *lock, struct thread *owner, int queue)
- int turnstile_signal (struct turnstile *ts, int queue)
- void turnstile_broadcast (struct turnstile *ts, int queue)
- void turnstile_unpend (struct turnstile *ts, int owner_type)
- void turnstile_disown (struct turnstile *ts)
- thread * turnstile_head (struct turnstile *ts, int queue)
- int turnstile_empty (struct turnstile *ts, int queue)

Variables

- static struct mtx td_contested_lock
- static struct turnstile_chain turnstile_chains [TC_TABLESIZE]

9.117.1 Define Documentation

9.117.1.1 #define TC_HASH(lock) (((uintptr_t)(lock) >> TC_SHIFT) & TC_MASK)

Definition at line 94 of file subr_turnstile.c.

9.117.1.2 #define TC_LOOKUP(lock) &turnstile_chains[TC_HASH(lock)]

Definition at line 95 of file subr_turnstile.c.

Referenced by propagate_priority(), turnstile_adjust(), turnstile_adjust_thread(), turnstile_broadcast(), turnstile_claim(), turnstile_disown(), turnstile_empty(), turnstile_head(), turnstile_lock(), turnstile_-lookup(), turnstile_release(), turnstile_signal(), turnstile_unpend(), and turnstile_wait().

9.117.1.3 #define TC_MASK (TC_TABLESIZE - 1)

Definition at line 92 of file subr_turnstile.c.

9.117.1.4 #define TC_SHIFT 8

Definition at line 93 of file subr_turnstile.c.

9.117.1.5 #define TC_TABLESIZE 128

Definition at line 91 of file subr_turnstile.c.

Referenced by init_turnstiles().

9.117.2 Function Documentation

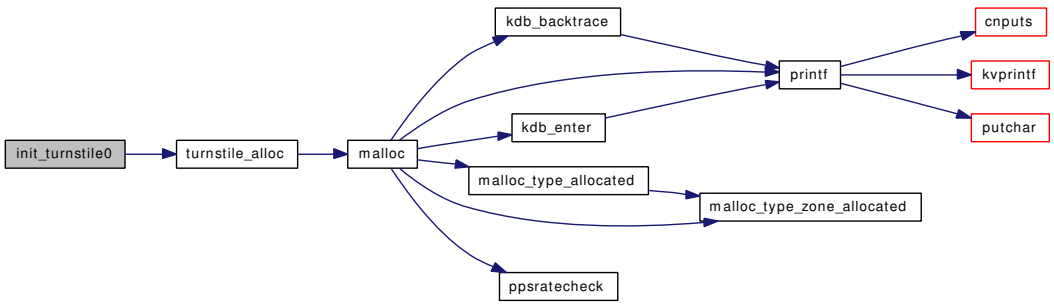
9.117.2.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_turnstile. *c*, v 1.166 2007/01/23 08:46:50 jeff Exp $")

9.117.2.2 static void init_turnstile0 (void * *dummy*) [static]

Definition at line 379 of file subr_turnstile.c.

References turnstile_alloc().

Here is the call graph for this function:



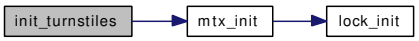
9.117.2.3 void init_turnstiles (void)

Definition at line 339 of file subr_turnstile.c.

References mtx_init(), TC_TABLESIZE, and turnstile_chains.

Referenced by mutex_init().

Here is the call graph for this function:



9.117.2.4 static MALLOC_DEFINE (M_TURNSTILE, "turnstiles", "turnstiles") [static]

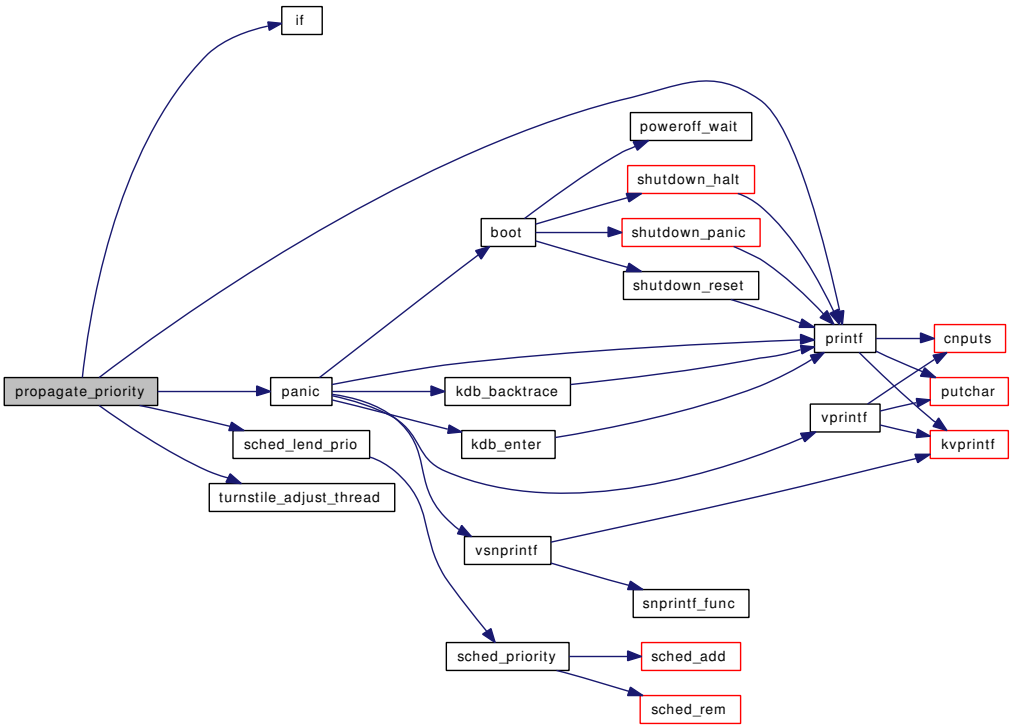
9.117.2.5 static void propagate_priority (struct thread * *td*) [static]

Definition at line 168 of file subr_turnstile.c.

References if(), panic(), printf(), sched_lend_prio(), sched_lock, TC_LOOKUP, and turnstile_adjust_thread().

Referenced by turnstile_adjust(), and turnstile_wait().

Here is the call graph for this function:



9.117.2.6 SYSINIT (turnstile0, SI_SUB_LOCK, SI_ORDER_ANY, init_turnstile0, NULL)

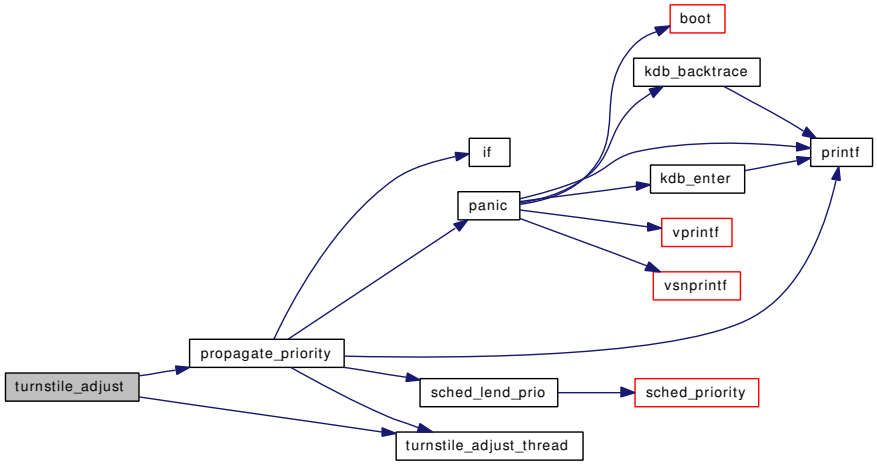
9.117.2.7 void turnstile_adjust (struct thread * *td*, u_char *oldpri*)

Definition at line 391 of file subr_turnstile.c.

References propagate_priority(), sched_lock, TC_LOOKUP, turnstile::ts_blocked, and turnstile_adjust_thread().

Referenced by sched_prio().

Here is the call graph for this function:



9.117.2.8 static int turnstile_adjust_thread (struct turnstile * *ts*, struct thread * *td*) [static]

Definition at line 267 of file subr_turnstile.c.

References sched_lock, TC_LOOKUP, and turnstile::ts_blocked.

Referenced by propagate_priority(), and turnstile_adjust().

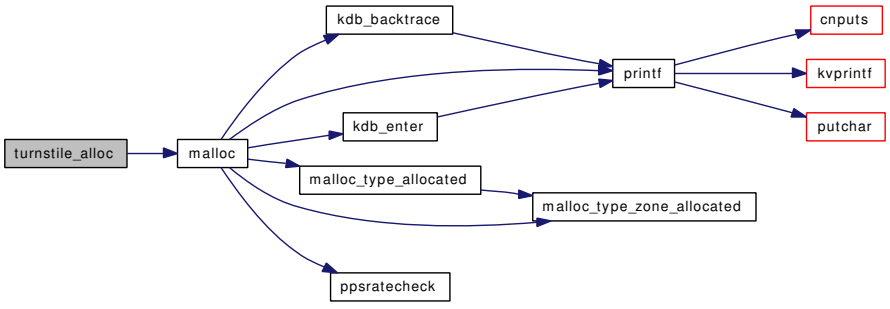
9.117.2.9 struct turnstile* turnstile_alloc (void)

Definition at line 454 of file subr_turnstile.c.

References malloc(), turnstile::ts_blocked, and turnstile::ts_pending.

Referenced by init_turnstile0(), and thread_init().

Here is the call graph for this function:



9.117.2.10 void turnstile_broadcast (struct turnstile * *ts*, int *queue*)

Definition at line 754 of file subr_turnstile.c.

References TC_LOOKUP, td, turnstile::ts_blocked, and turnstile::ts_pending.

Referenced by _mtx_unlock_sleep(), _rw_downgrade(), _rw_runlock(), and _rw_wunlock_hard().

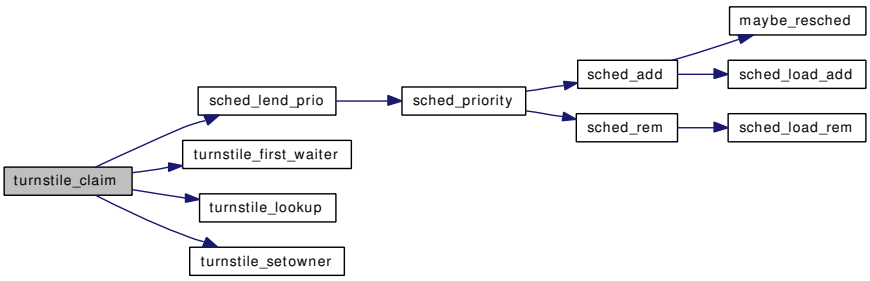
9.117.2.11 void turnstile_claim (struct lock_object * *lock*)

Definition at line 544 of file subr_turnstile.c.

References sched_lend_prio(), sched_lock, TC_LOOKUP, td, turnstile_first_waiter(), turnstile_lookup(), and turnstile_setowner().

Referenced by _mtx_lock_sleep(), _rw_try_upgrade(), and _rw_wlock_hard().

Here is the call graph for this function:



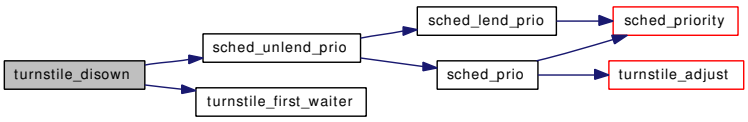
9.117.2.12 void turnstile_disown (struct turnstile * *ts*)

Definition at line 893 of file subr_turnstile.c.

References sched_lock, sched_unlend_prio(), TC_LOOKUP, td, turnstile::ts_blocked, turnstile::ts_pending, and turnstile_first_waiter().

Referenced by _rw_downgrade(), and _rw_wunlock_hard().

Here is the call graph for this function:



9.117.2.13 int turnstile_empty (struct turnstile * *ts*, int *queue*)

Definition at line 959 of file subr_turnstile.c.

References TC_LOOKUP, and turnstile::ts_blocked.

Referenced by _rw_downgrade(), and _rw_wunlock_hard().

9.117.2.14 static struct thread * turnstile_first_waiter (struct turnstile * *ts*) [static]

Definition at line 528 of file subr_turnstile.c.

References turnstile::ts_blocked.

Referenced by turnstile_claim(), turnstile_disown(), and turnstile_unpend().

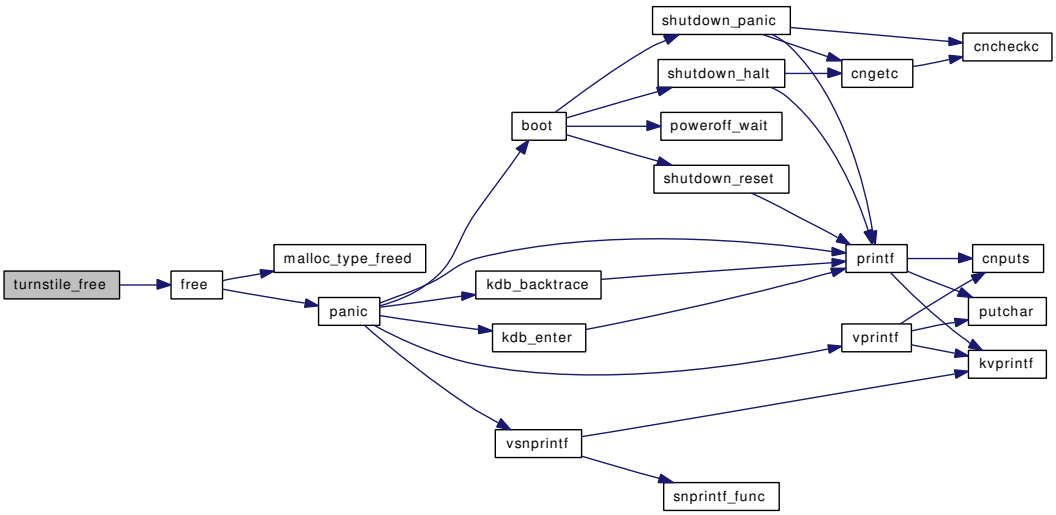
9.117.2.15 void turnstile_free (struct turnstile * *ts*)

Definition at line 470 of file subr_turnstile.c.

References free(), turnstile::ts_blocked, and turnstile::ts_pending.

Referenced by thread_fini().

Here is the call graph for this function:



9.117.2.16 struct thread* turnstile_head (struct turnstile * *ts*, int *queue*)

Definition at line 942 of file subr_turnstile.c.

References TC_LOOKUP, and turnstile::ts_blocked.

Referenced by _mtx_unlock_sleep().

9.117.2.17 void turnstile_lock (struct lock_object * *lock*)

Definition at line 484 of file subr_turnstile.c.

References TC_LOOKUP.

Referenced by _mtx_lock_sleep(), _mtx_unlock_sleep(), _rw_downgrade(), _rw_rlock(), _rw_runlock(), _rw_try_upgrade(), _rw_wlock_hard(), and _rw_wunlock_hard().

9.117.2.18 struct turnstile* turnstile_lookup (struct lock_object * *lock*)

Definition at line 498 of file subr_turnstile.c.

References TC_LOOKUP.

Referenced by _mtx_unlock_sleep(), _rw_downgrade(), _rw_runlock(), _rw_try_upgrade(), _rw_-wunlock_hard(), turnstile_claim(), and turnstile_wait().

9.117.2.19 void turnstile_release (struct lock_object * *lock*)

Definition at line 515 of file subr_turnstile.c.

References TC_LOOKUP.

Referenced by _mtx_lock_sleep(), _mtx_unlock_sleep(), _rw_downgrade(), _rw_rlock(), _rw_runlock(), _rw_try_upgrade(), _rw_wlock_hard(), and _rw_wunlock_hard().

9.117.2.20 static void turnstile_setowner (struct turnstile * *ts*, struct thread * *owner*) [static]

Definition at line 435 of file subr_turnstile.c.

Referenced by turnstile_claim(), and turnstile_wait().

9.117.2.21 int turnstile_signal (struct turnstile * *ts*, int *queue*)

Definition at line 703 of file subr_turnstile.c.

References TC_LOOKUP, td, turnstile::ts_blocked, and turnstile::ts_pending.

Referenced by _mtx_unlock_sleep().

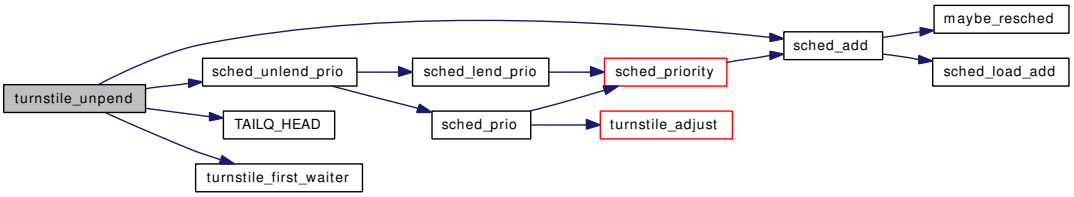
9.117.2.22 void turnstile_unpend (struct turnstile * *ts*, int *owner_type*)

Definition at line 800 of file subr_turnstile.c.

References sched_add(), sched_lock, sched_unlend_prio(), TAILQ_HEAD(), TC_LOOKUP, td, turnstile::ts_blocked, turnstile::ts_pending, and turnstile_first_waiter().

Referenced by _mtx_unlock_sleep(), _rw_downgrade(), _rw_runlock(), and _rw_wunlock_hard().

Here is the call graph for this function:



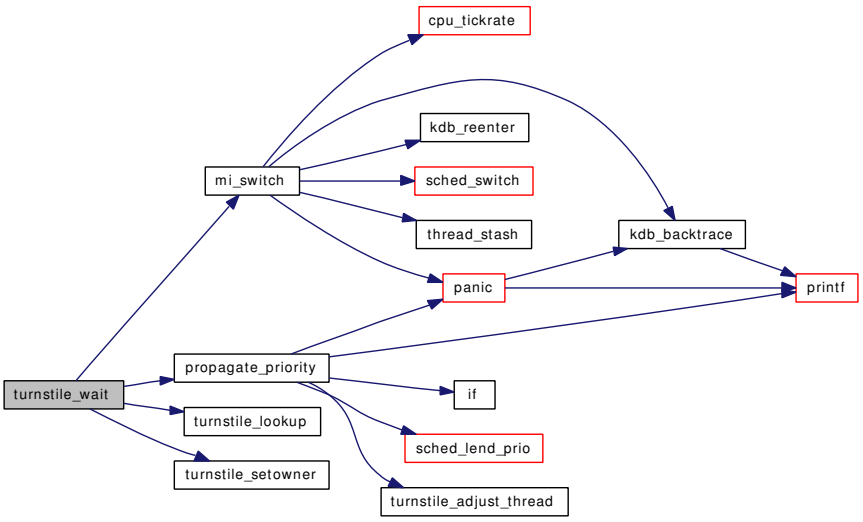
9.117.2.23 void turnstile_wait (struct lock_object * *lock*, struct thread * *owner*, int *queue*)

Definition at line 581 of file subr_turnstile.c.

References mi_switch(), propagate_priority(), sched_lock, TC_LOOKUP, td, turnstile::ts_blocked, turnstile::ts_pending, turnstile_lookup(), and turnstile_setowner().

Referenced by _mtx_lock_sleep(), _rw_rlock(), and _rw_wlock_hard().

Here is the call graph for this function:



9.117.3 Variable Documentation

9.117.3.1 struct mtx td_contested_lock [static]

Definition at line 144 of file subr_turnstile.c.

9.117.3.2 struct turnstile_chain turnstile_chains[TC_TABLESIZE] [static]

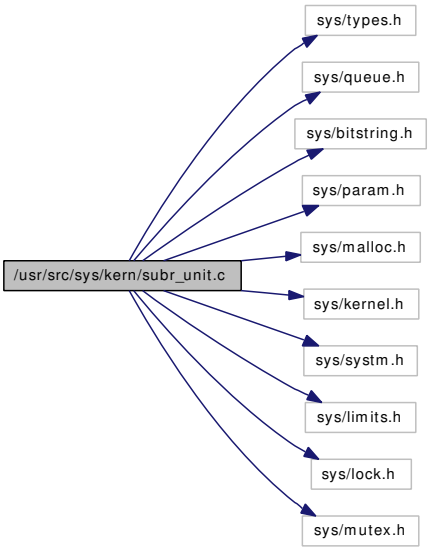
Definition at line 145 of file subr_turnstile.c.

Referenced by init_turnstiles().

9.118 /usr/src/sys/kern/subr_unit.c File Reference

```
#include <sys/types.h>
#include <sys/queue.h>
#include <sys/bitstring.h>
#include <sys/param.h>
#include <sys/malloc.h>
#include <sys/kernel.h>
#include <sys/systm.h>
#include <sys/limits.h>
#include <sys/lock.h>
#include <sys/mutex.h>
```

Include dependency graph for subr_unit.c:



Data Structures

- struct unr
- struct unrb
- struct unrhdr

Defines

- #define Malloc(foo) malloc(foo, M_UNIT, M_WAITOK | M_ZERO)
- #define Free(foo) free(foo, M_UNIT)
- #define NBITS ((int)sizeof(((struct unrb *)NULL) → map) * 8)

Functions

- static MALLOC_DEFINE (M_UNIT,"Unitno","Unit number allocation")
- MTX_SYSINIT (unit,&unitmtx,"unit# allocation", MTX_DEF)
- CTASSERT (sizeof(struct unr)==sizeof(struct unrb))
- static __inline void check_unrhdr (struct unrhdr *uh, int line)
- static __inline void * new_unr (struct unrhdr *uh, void **p1, void **p2)
- static __inline void delete_unr (struct unrhdr *uh, void *ptr)
- unrhdr * new_unrhdr (int low, int high, struct mtx *mutex)
- void delete_unrhdr (struct unrhdr *uh)
- static __inline int is_bitmap (struct unrhdr *uh, struct unr *up)
- static int optimize_unr (struct unrhdr *uh)
- static void collapse_unr (struct unrhdr *uh, struct unr *up)
- int alloc_unrl (struct unrhdr *uh)
- int alloc_unr (struct unrhdr *uh)
- static void free_unrl (struct unrhdr *uh, u_int item, void **p1, void **p2)
- void free_unr (struct unrhdr *uh, u_int item)

Variables

- static struct mtx unitmtx

9.118.1 Define Documentation

9.118.1.1 #define Free(foo) free(foo, M_UNIT)

Definition at line 93 of file subr_unit.c.

Referenced by delete_unr(), delete_unrhdr(), and free_unr().

9.118.1.2 #define Malloc(foo) malloc(foo, M_UNIT, M_WAITOK | M_ZERO)

Definition at line 92 of file subr_unit.c.

Referenced by free_unr(), and new_unrhdr().

9.118.1.3 #define NBITS ((int)sizeof(((struct unrb *)NULL) → map) * 8)

Definition at line 187 of file subr_unit.c.

Referenced by optimize_unr().

9.118.2 Function Documentation

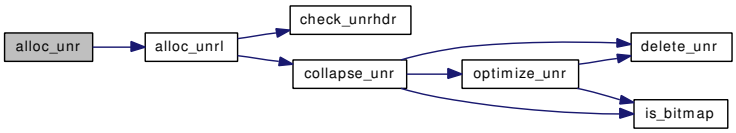
9.118.2.1 int alloc_unr (struct unrhdr * *uh*)

Definition at line 588 of file subr_unit.c.

References alloc_unrl().

Referenced by aio_newproc(), mqfs_fileno_alloc(), and ttycreate().

Here is the call graph for this function:



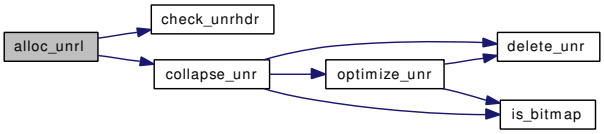
9.118.2.2 int alloc_unrl (struct unrhdr * *uh*)

Definition at line 538 of file subr_unit.c.

References unrb::busy, check_unrhdr(), collapse_unr(), and unrb::map.

Referenced by alloc_unr().

Here is the call graph for this function:



9.118.2.3 static __inline void check_unrhdr (struct unrhdr * *uh*, int *line*) [static]

Definition at line 250 of file subr_unit.c.

Referenced by alloc_unrl(), delete_unrhdr(), free_unrl(), and new_unrhdr().

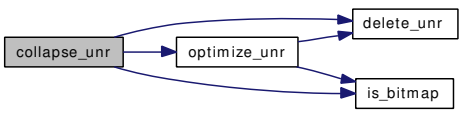
9.118.2.4 static void collapse_unr (struct unrhdr * *uh*, struct unr * *up*) [static]

Definition at line 466 of file subr_unit.c.

References unrb::busy, delete_unr(), is_bitmap(), and optimize_unr().

Referenced by alloc_unrl(), and free_unrl().

Here is the call graph for this function:



9.118.2.5 CTASSERT (sizeof(struct unr) = =sizeof(struct unrb))

9.118.2.6 static __inline void delete_unr (struct unrhdr * *uh*, void * *ptr*) [static]

Definition at line 282 of file subr_unit.c.

References Free.

Referenced by collapse_unr(), and optimize_unr().

9.118.2.7 void delete_unrhdr (struct unrhdr * *uh*)

Definition at line 317 of file subr_unit.c.

References check_unrhdr(), and Free.

Referenced by aio_unload(), and mqfs_fileno_uninit().

Here is the call graph for this function:

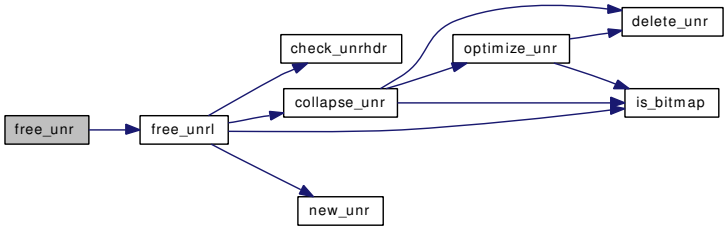
9.118.2.8 void free_unr (struct unrhdr * *uh*, u_int *item*)

Definition at line 713 of file subr_unit.c.

References Free, free_unrl(), and Malloc.

Referenced by aio_daemon(), aio_newproc(), mqfs_fileno_free(), thread_dtor(), and ttyfree().

Here is the call graph for this function:



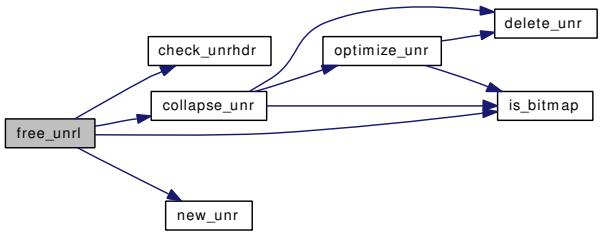
9.118.2.9 static void free_unrl (struct unrhdr * *uh*, u_int *item*, void ** *p1*, void ** *p2*) [static]

Definition at line 604 of file subr_unit.c.

References unrb::busy, check_unrhdr(), collapse_unr(), is_bitmap(), unrb::map, and new_unr().

Referenced by free_unr().

Here is the call graph for this function:



9.118.2.10 static __inline int is_bitmap (struct unrhdr * *uh*, struct unr * *up*) [static]

Definition at line 327 of file subr_unit.c.

Referenced by collapse_unr(), free_unrl(), and optimize_unr().

9.118.2.11 static MALLOC_DEFINE (M_UNIT, "Unitno", "Unit number allocation") [static]

9.118.2.12 MTX_SYSINIT (unit, & *unitmtx*, "unit# allocation", MTX_DEF)

9.118.2.13 static __inline void* new_unr (struct unrhdr * *uh*, void ** *p1*, void ** *p2*) [static]

Definition at line 264 of file subr_unit.c.

Referenced by free_unrl().

9.118.2.14 struct unrhdr* new_unrhdr (int *low*, int *high*, struct mtx * *mutex*)

Definition at line 296 of file subr_unit.c.

References check_unrhdr(), Malloc, and unitmtx.

Referenced by aio_onceonly(), mqfs_fileno_init(), threadinit(), and ttycreate().

Here is the call graph for this function:

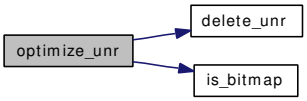
9.118.2.15 static int optimize_unr (struct unrhdr * *uh*) [static]

Definition at line 342 of file subr_unit.c.

References unrb::busy, delete_unr(), is_bitmap(), unrb::map, and NBITS.

Referenced by collapse_unr().

Here is the call graph for this function:



9.118.3 Variable Documentation

9.118.3.1 struct mtx unitmtx [static]

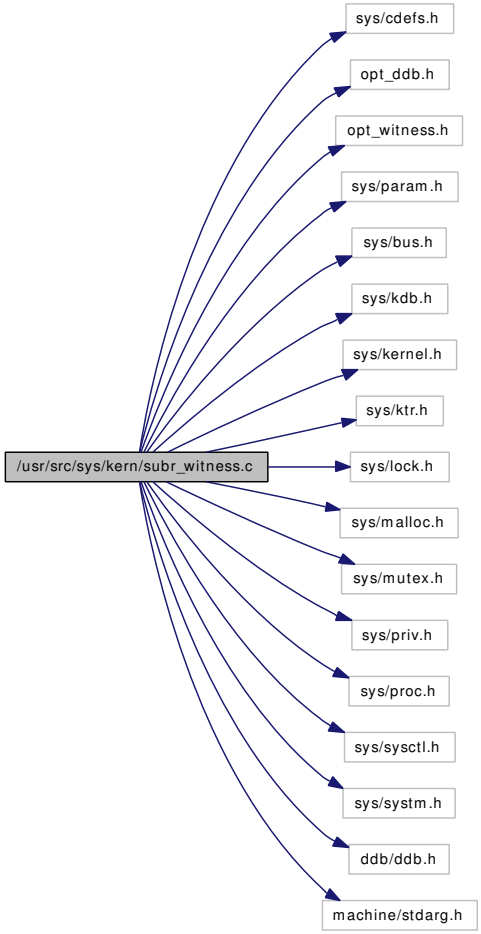
Definition at line 95 of file subr_unit.c.

Referenced by new_unrhdr().

9.119 /usr/src/sys/kern/subr_witness.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ddb.h"
#include "opt_witness.h"
#include <sys/param.h>
#include <sys/bus.h>
#include <sys/kdb.h>
#include <sys/kernel.h>
#include <sys/ktr.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mutex.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/sysctl.h>
#include <sys/systm.h>
#include <ddb/ddb.h>
#include <machine/stdarg.h>
```

Include dependency graph for subr_witness.c:



Data Structures

- struct witness
- struct witness_child_list_entry
- struct witness_order_list_entry

Defines

- #define KTR_WITNESS 0
- #define lo_list lo_witness_data.lod_list
- #define lo_witness lo_witness_data.lod_witness
- #define WITNESS_COUNT 1024
- #define WITNESS_CHILDCOUNT (WITNESS_COUNT ∗ 4)
- #define LOCK_CHILDCOUNT (MAXCPU + 1024) ∗ 2
- #define WITNESS_NCHILDREN 6

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/subr_witness.c,v 1.225 2007/02/20 23:49:30 rwatson Exp $")

- STAILQ_HEAD (witness_list, witness)
- static int depart (struct witness *w)
- static struct witness * enroll (const char *description, struct lock_class *lock_class)
- static int insertchild (struct witness *parent, struct witness *child)
- static int isitmychild (struct witness *parent, struct witness *child)
- static int isitmydescendant (struct witness *parent, struct witness *child)
- static int itismychild (struct witness *parent, struct witness *child)
- static void removechild (struct witness *parent, struct witness *child)
- static int sysctl_debug_witness_watch (SYSCTL_HANDLER_ARGS)
- static const char * fixup_filename (const char *file)
- static struct witness * witness_get (void)
- static void witness_free (struct witness *m)
- static struct witness_child_list_entry * witness_child_get (void)
- static void witness_child_free (struct witness_child_list_entry *wcl)
- static struct lock_list_entry * witness_lock_list_get (void)
- static void witness_lock_list_free (struct lock_list_entry *lle)
- static struct lock_instance * find_instance (struct lock_list_entry *lock_list, struct lock_object *lock)
- static void witness_list_lock (struct lock_instance *instance)
- SYSCTL_NODE (_debug, OID_AUTO, witness, CTLFLAG_RW, 0,"Witness Locking")
- TUNABLE_INT ("debug.witness.watch",&witness_watch)
- SYSCTL_PROC (_debug_witness, OID_AUTO, watch, CTLFLAG_RW|CTLTYPE_INT, NULL, 0, sysctl_debug_witness_watch,"I","witness is watching lock operations")
- TUNABLE_INT ("debug.witness.skipspin",&witness_skipspin)
- SYSCTL_INT (_debug_witness, OID_AUTO, skipspin, CTLFLAG_RDTUN,&witness_skipspin, 0,"")
- SYSCTL_INT (_debug_witness, OID_AUTO, free_cnt, CTLFLAG_RD,&w_free_cnt, 0,"")
- SYSCTL_INT (_debug_witness, OID_AUTO, spin_cnt, CTLFLAG_RD,&w_spin_cnt, 0,"")
- SYSCTL_INT (_debug_witness, OID_AUTO, sleep_cnt, CTLFLAG_RD,&w_sleep_cnt, 0,"")
- SYSCTL_INT (_debug_witness, OID_AUTO, child_free_cnt, CTLFLAG_RD,&w_child_free_cnt, 0,"")
- SYSCTL_INT (_debug_witness, OID_AUTO, child_cnt, CTLFLAG_RD,&w_child_cnt, 0,"")
- STAILQ_HEAD (lock_object)
- void witness_init (struct lock_object *lock)
- void witness_destroy (struct lock_object *lock)
- int witness_defineorder (struct lock_object *lock1, struct lock_object *lock2)
- void witness_checkorder (struct lock_object *lock, int flags, const char *file, int line)
- void witness_lock (struct lock_object *lock, int flags, const char *file, int line)
- void witness_upgrade (struct lock_object *lock, int flags, const char *file, int line)
- void witness_downgrade (struct lock_object *lock, int flags, const char *file, int line)
- void witness_unlock (struct lock_object *lock, int flags, const char *file, int line)
- int witness_warn (int flags, struct lock_object *lock, const char *fmt,...)
- const char * witness_file (struct lock_object *lock)
- int witness_line (struct lock_object *lock)
- int witness_list_locks (struct lock_list_entry **lock_list)
- void witness_display_spinlock (struct lock_object *lock, struct thread *owner)
- void witness_save (struct lock_object *lock, const char **filep, int *linep)
- void witness_restore (struct lock_object *lock, const char *file, int line)
- void witness_assert (struct lock_object *lock, int flags, const char *file, int line)

Variables

- static int witness_watch = 1
- int witness_skipspin = 0
- static struct mtx w_mtx
- static struct witness_list w_free = STAILQ_HEAD_INITIALIZER(w_free)
- static struct witness_list w_all = STAILQ_HEAD_INITIALIZER(w_all)
- static struct witness_list w_spin = STAILQ_HEAD_INITIALIZER(w_spin)
- static struct witness_list w_sleep = STAILQ_HEAD_INITIALIZER(w_sleep)
- static struct witness_child_list_entry * w_child_free = NULL
- static struct lock_list_entry * w_lock_list_free = NULL
- static int w_free_cnt
- static int w_spin_cnt
- static int w_sleep_cnt
- static int w_child_free_cnt
- static int w_child_cnt
- static struct witness w_data [WITNESS_COUNT]
- static struct witness_child_list_entry w_childdata [WITNESS_CHILDCOUNT]
- static struct lock_list_entry w_locklistdata [LOCK_CHILDCOUNT]
- static struct witness_order_list_entry order_lists [ ]

9.119.1 Define Documentation

9.119.1.1 #define KTR_WITNESS 0

Definition at line 111 of file subr_witness.c.

Referenced by STAILQ_HEAD(), witness_defineorder(), witness_lock(), and witness_unlock().

9.119.1.2 #define lo_list lo_witness_data.lod_list

Definition at line 115 of file subr_witness.c.

Referenced by STAILQ_HEAD(), witness_destroy(), and witness_init().

9.119.1.3 #define lo_witness lo_witness_data.lod_witness

Definition at line 116 of file subr_witness.c.

9.119.1.4 #define LOCK_CHILDCOUNT (MAXCPU + 1024) * 2

Definition at line 128 of file subr_witness.c.

9.119.1.5 #define WITNESS_CHILDCOUNT (WITNESS_COUNT * 4)

Definition at line 122 of file subr_witness.c.

Referenced by STAILQ_HEAD().

9.119.1.6 #define WITNESS_COUNT 1024

Definition at line 121 of file subr_witness.c.

Referenced by STAILQ_HEAD(), and witness_checkorder().

9.119.1.7 #define WITNESS_NCHILDREN 6

Definition at line 130 of file subr_witness.c.

Referenced by insertchild().

9.119.2 Function Documentation

9.119.2.1 __FBSDID ("$FreeBSD: src/sys/kern/subr_witness. *c*, v 1.225 2007/02/20 23:49:30 rwatson Exp $")

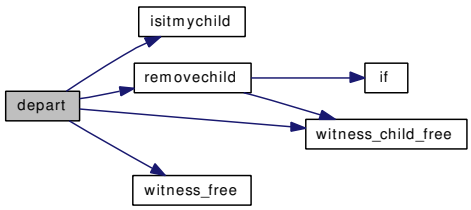
9.119.2.2 static int depart (struct witness * *w*) [static]

Definition at line 1452 of file subr_witness.c.

References isitmychild(), removechild(), w_child_cnt, witness::w_class, w_sleep_cnt, w_spin_cnt, witness_child_list_entry::wcl_next, witness_child_free(), and witness_free().

Referenced by witness_destroy().

Here is the call graph for this function:



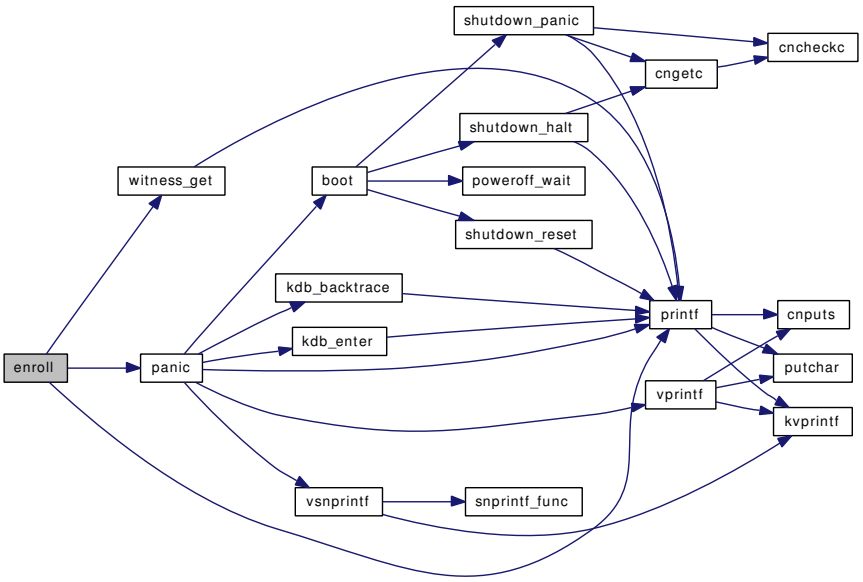
9.119.2.3 static struct witness * enroll (const char * *description*, struct lock_class * *lock_class*) [static]

Definition at line 1395 of file subr_witness.c.

References panic(), panicstr, printf(), witness::w_class, witness::w_name, w_sleep_cnt, w_spin_cnt, witness_get(), witness_skipspin, and witness_watch.

Referenced by STAILQ_HEAD(), and witness_init().

Here is the call graph for this function:



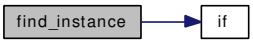
9.119.2.4 static struct lock_instance * find_instance (struct lock_list_entry * *lock_list*, struct lock_object * *lock*) [static]

Definition at line 1725 of file subr_witness.c.

References if().

Referenced by witness_assert(), witness_checkorder(), witness_display_spinlock(), witness_downgrade(), witness_lock(), witness_restore(), witness_save(), and witness_upgrade().

Here is the call graph for this function:



9.119.2.5 static const char * fixup_filename (const char * *file*) [static]

Definition at line 762 of file subr_witness.c.

Referenced by witness_assert(), witness_checkorder(), witness_downgrade(), witness_lock(), witness_unlock(), and witness_upgrade().

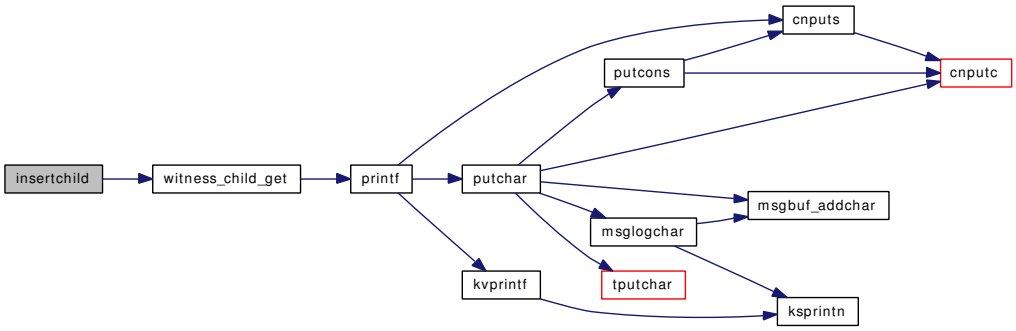
9.119.2.6 static int insertchild (struct witness * *parent*, struct witness * *child*) [static]

Definition at line 1503 of file subr_witness.c.

References w_child_cnt, witness_child_list_entry::wcl_count, witness_child_list_entry::wcl_next, witness_child_get(), and WITNESS_NCHILDREN.

Referenced by itismychild().

Here is the call graph for this function:



9.119.2.7 static int isitmychild (struct witness * *parent*, struct witness * *child*) [static]

Definition at line 1576 of file subr_witness.c.

References witness_child_list_entry::wcl_children, witness_child_list_entry::wcl_count, and witness_child_list_entry::wcl_next.

Referenced by depart(), isitmydescendant(), and witness_checkorder().

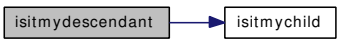
9.119.2.8 static int isitmydescendant (struct witness * *parent*, struct witness * *child*) [static]

Definition at line 1591 of file subr_witness.c.

References isitmychild(), witness_child_list_entry::wcl_children, witness_child_list_entry::wcl_count, and witness_child_list_entry::wcl_next.

Referenced by witness_checkorder(), and witness_defineorder().

Here is the call graph for this function:



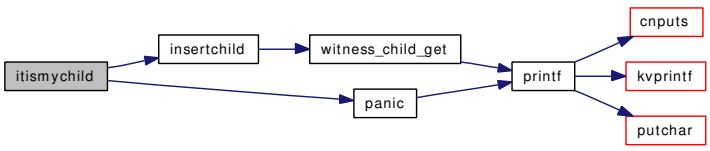
9.119.2.9 static int itismychild (struct witness * *parent*, struct witness * *child*) [static]

Definition at line 1528 of file subr_witness.c.

References insertchild(), panic(), and witness::w_class.

Referenced by STAILQ_HEAD(), and witness_defineorder().

Here is the call graph for this function:



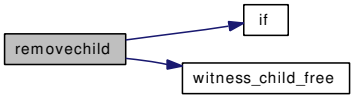
9.119.2.10 static void removechild (struct witness * *parent*, struct witness * *child*) [static]

Definition at line 1551 of file subr_witness.c.

References if(), w_child_cnt, witness_child_list_entry::wcl_count, witness_child_list_entry::wcl_next, and witness_child_free().

Referenced by depart().

Here is the call graph for this function:

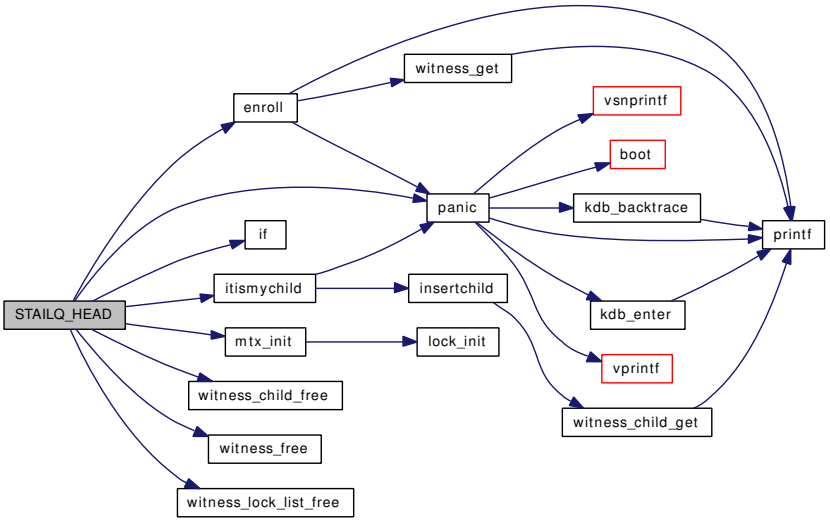


9.119.2.11 STAILQ_HEAD (lock_object)

Definition at line 454 of file subr_witness.c.

References enroll(), Giant, if(), itismychild(), KTR_WITNESS, lo_list, mtx_init(), order_lists, panic(), w_-childdata, witness_order_list_entry::w_class, w_data, witness_order_list_entry::w_name, witness_child_-free(), WITNESS_CHILDCOUNT, WITNESS_COUNT, witness_free(), and witness_lock_list_free().

Here is the call graph for this function:



9.119.2.12 STAILQ_HEAD (witness_list, witness)

9.119.2.13 static int sysctl_debug_witness_watch (SYSCTL_HANDLER_ARGS) [static]

Definition at line 534 of file subr_witness.c.

References sysctl_handle_int(), and witness_watch.

Here is the call graph for this function:

9.119.2.14 SYSCTL_INT (_debug_witness, OID_AUTO, child_cnt, CTLFLAG_RD, & *w_child_cnt*, 0, "")

9.119.2.15 SYSCTL_INT (_debug_witness, OID_AUTO, child_free_cnt, CTLFLAG_RD, & *w_child_free_cnt*, 0, "")

9.119.2.16 SYSCTL_INT (_debug_witness, OID_AUTO, sleep_cnt, CTLFLAG_RD, & *w_sleep_cnt*, 0, "")

9.119.2.17 SYSCTL_INT (_debug_witness, OID_AUTO, spin_cnt, CTLFLAG_RD, & *w_spin_cnt*, 0, "")

9.119.2.18 SYSCTL_INT (_debug_witness, OID_AUTO, free_cnt, CTLFLAG_RD, & *w_free_cnt*, 0, "")

9.119.2.19 SYSCTL_INT (_debug_witness, OID_AUTO, skipspin, CTLFLAG_RDTUN, & *witness_skipspin*, 0, "")

9.119.2.20 SYSCTL_NODE (_debug, OID_AUTO, witness, CTLFLAG_RW, 0, "Witness Locking")

9.119.2.21 SYSCTL_PROC (_debug_witness, OID_AUTO, watch, CTLFLAG_RW| *CTLTYPE_INT*, NULL, 0, sysctl_debug_witness_watch, "I", "witness is watching lock operations")

9.119.2.22 TUNABLE_INT ("debug.witness.skipspin", & *witness_skipspin*)

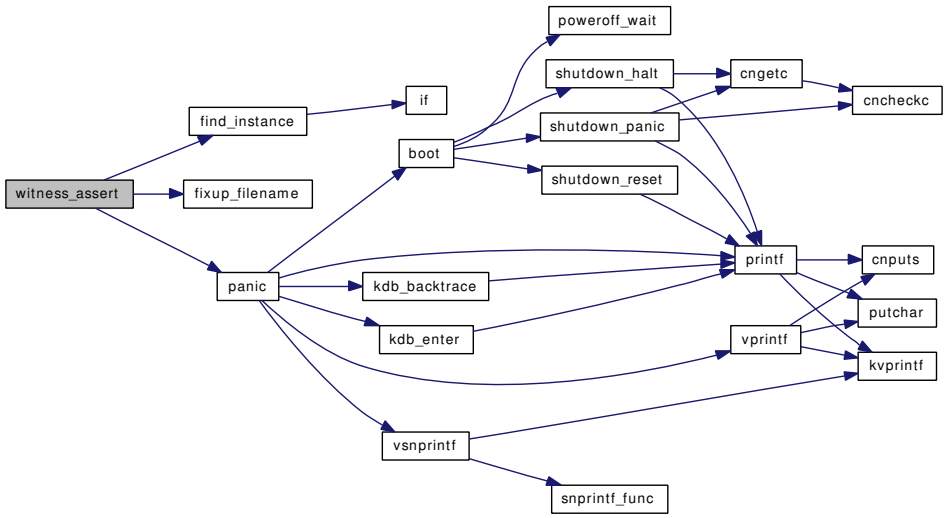
9.119.2.23 TUNABLE_INT ("debug.witness.watch", & *witness_watch*)

9.119.2.24 void witness_assert (struct lock_object * *lock*, int *flags*, const char * *file*, int *line*)

Definition at line 1867 of file subr_witness.c.

References find_instance(), fixup_filename(), panic(), panicstr, and witness_watch.

Here is the call graph for this function:

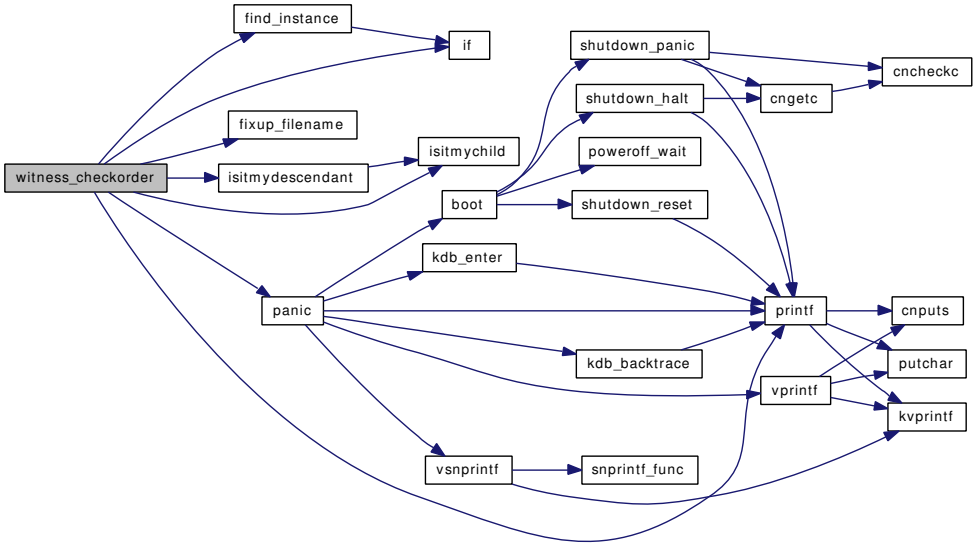


9.119.2.25 void witness_checkorder (struct lock_object * *lock*, int *flags*, const char * *file*, int *line*)

Definition at line 806 of file subr_witness.c.

References find_instance(), fixup_filename(), Giant, if(), isitmychild(), isitmydescendant(), kdb_active, panic(), panicstr, printf(), td, WITNESS_COUNT, and witness_watch.

Here is the call graph for this function:



9.119.2.26 static void witness_child_free (struct witness_child_list_entry * *wcl*) [static]

Definition at line 1685 of file subr_witness.c.

References w_child_free, w_child_free_cnt, and witness_child_list_entry::wcl_next.

Referenced by depart(), removechild(), and STAILQ_HEAD().

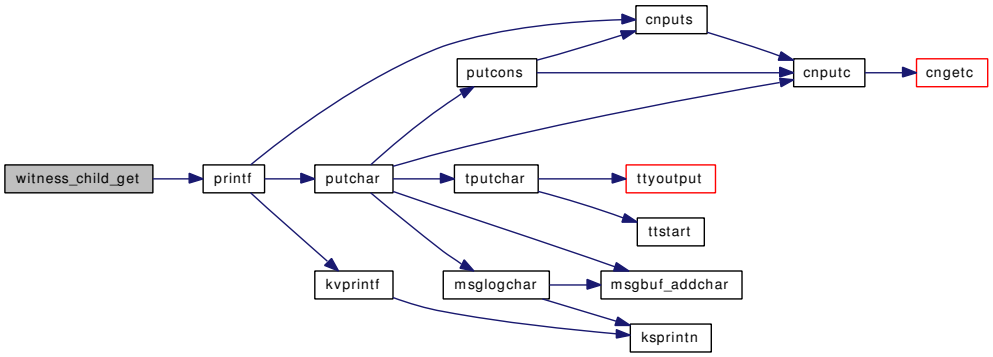
9.119.2.27 static struct witness_child_list_entry * witness_child_get (void) [static]

Definition at line 1663 of file subr_witness.c.

References printf(), w_child_free, w_child_free_cnt, witness_child_list_entry::wcl_next, and witness_-watch.

Referenced by insertchild().

Here is the call graph for this function:

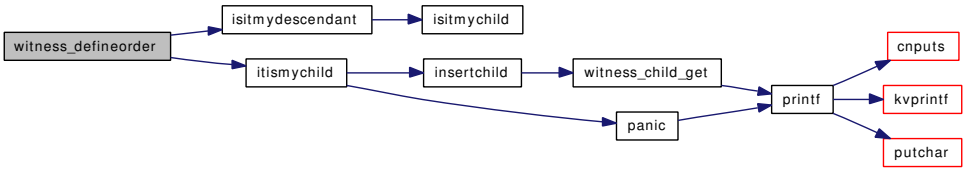


9.119.2.28 int witness_defineorder (struct lock_object * *lock1*, struct lock_object * *lock2*)

Definition at line 773 of file subr_witness.c.

References isitmydescendant(), itismychild(), KTR_WITNESS, panicstr, and witness_watch.

Here is the call graph for this function:

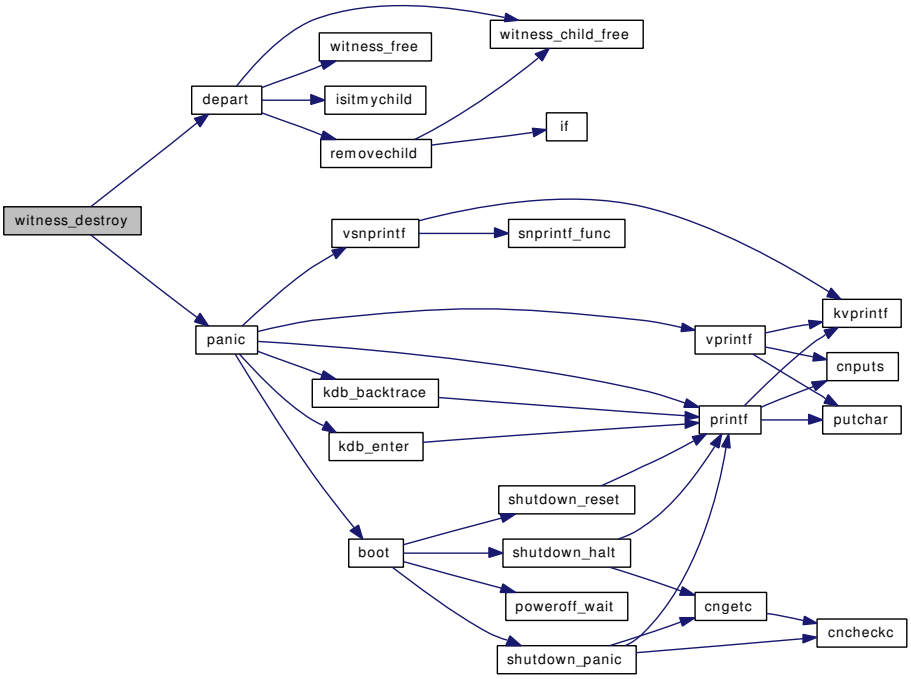


9.119.2.29 void witness_destroy (struct lock_object * *lock*)

Definition at line 588 of file subr_witness.c.

References depart(), lo_list, and panic().

Here is the call graph for this function:

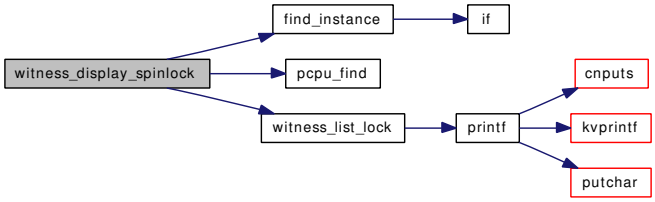


9.119.2.30 void witness_display_spinlock (struct lock_object * *lock*, struct thread * *owner*)

Definition at line 1799 of file subr_witness.c.

References find_instance(), pcpu_find(), and witness_list_lock().

Here is the call graph for this function:

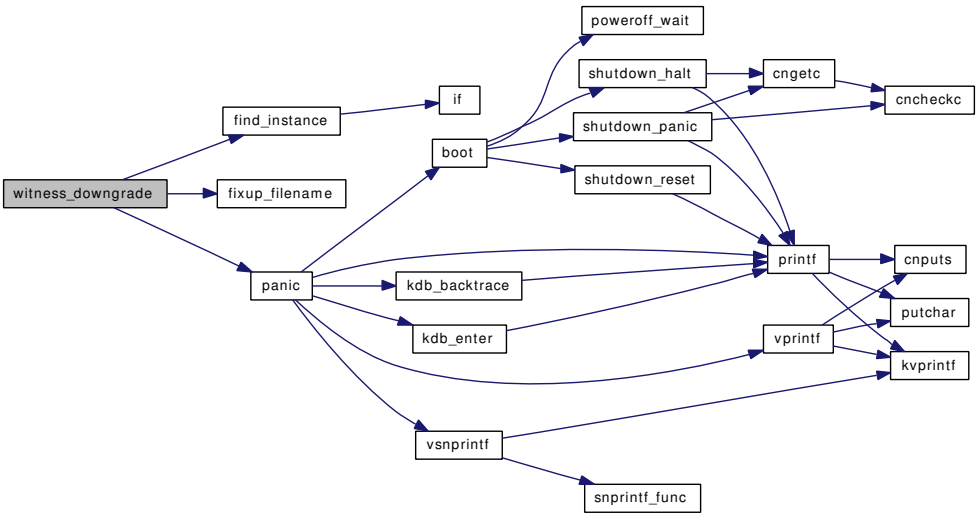


9.119.2.31 void witness_downgrade (struct lock_object * *lock*, int *flags*, const char * *file*, int *line*)

Definition at line 1191 of file subr_witness.c.

References find_instance(), fixup_filename(), panic(), panicstr, and witness_watch.

Here is the call graph for this function:



9.119.2.32 const char* witness_file (struct lock_object * *lock*)

Definition at line 1373 of file subr_witness.c.

References witness_watch.

9.119.2.33 static void witness_free (struct witness * *m*) [static]

Definition at line 1655 of file subr_witness.c.

References w_free_cnt.

Referenced by depart(), and STAILQ_HEAD().

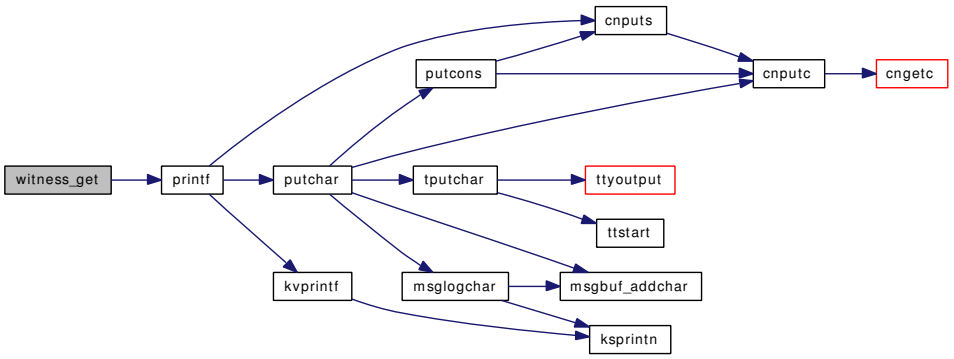
9.119.2.34 static struct witness * witness_get (void) [static]

Definition at line 1633 of file subr_witness.c.

References printf(), w_free_cnt, and witness_watch.

Referenced by enroll().

Here is the call graph for this function:

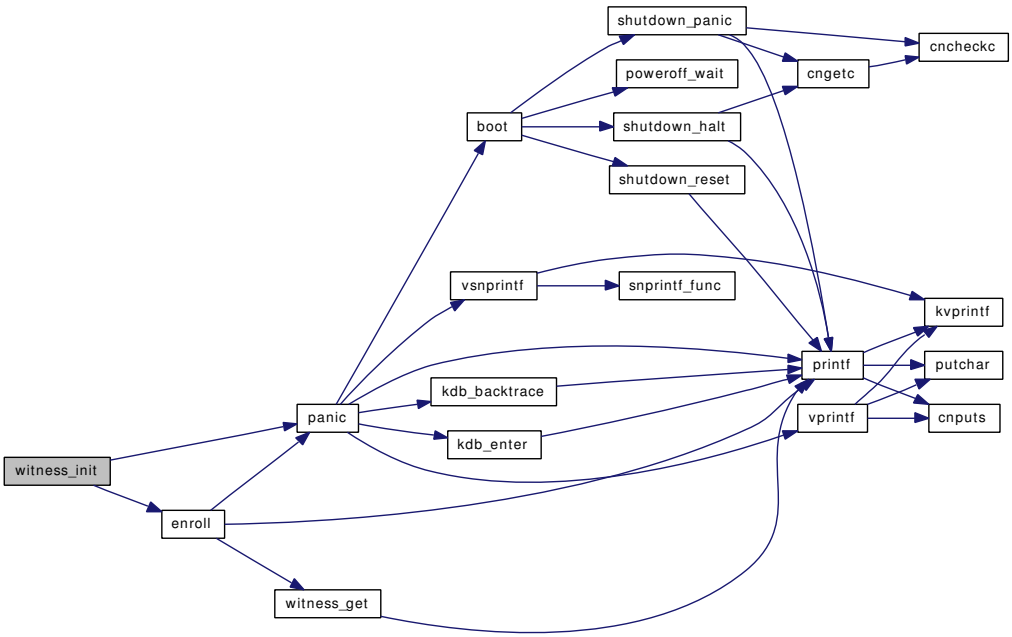


9.119.2.35 void witness_init (struct lock_object * *lock*)

Definition at line 551 of file subr_witness.c.

References enroll(), lo_list, panic(), panicstr, and witness_watch.

Here is the call graph for this function:



9.119.2.36 int witness_line (struct lock_object * *lock*)

Definition at line 1384 of file subr_witness.c.

References witness_watch.

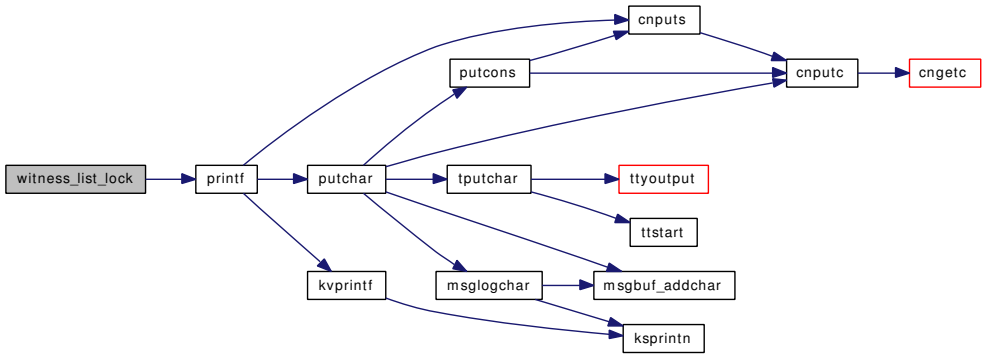
9.119.2.37 static void witness_list_lock (struct lock_instance * *instance*) [static]

Definition at line 1741 of file subr_witness.c.

References printf().

Referenced by witness_display_spinlock(), witness_list_locks(), and witness_warn().

Here is the call graph for this function:



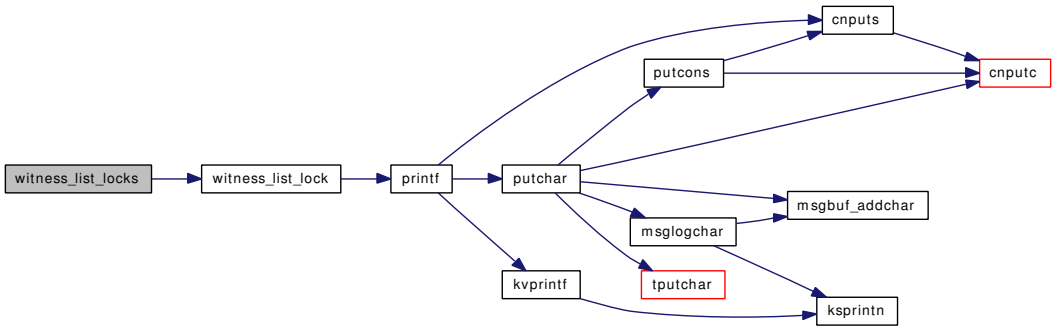
9.119.2.38 int witness_list_locks (struct lock_list_entry ** *lock_list*)

Definition at line 1777 of file subr_witness.c.

References witness_list_lock().

Referenced by witness_warn().

Here is the call graph for this function:

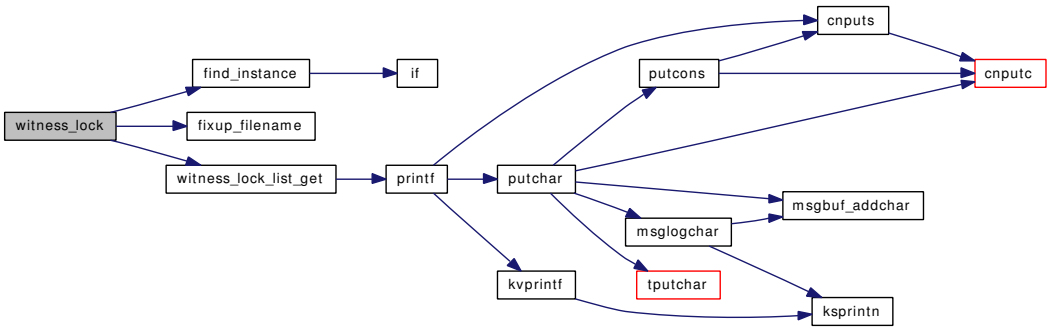


9.119.2.39 void witness_lock (struct lock_object * *lock*, int *flags*, const char * *file*, int *line*)

Definition at line 1097 of file subr_witness.c.

References find_instance(), fixup_filename(), KTR_WITNESS, panicstr, td, witness_lock_list_get(), and witness_watch.

Here is the call graph for this function:



9.119.2.40 static void witness_lock_list_free (struct lock_list_entry * *lle*) [static]

Definition at line 1715 of file subr_witness.c.

References w_lock_list_free.

Referenced by STAILQ_HEAD(), and witness_unlock().

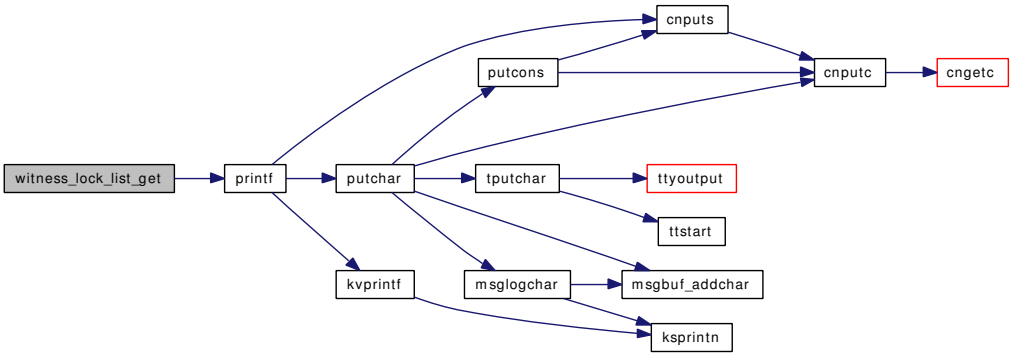
9.119.2.41 static struct lock_list_entry * witness_lock_list_get (void) [static]

Definition at line 1694 of file subr_witness.c.

References printf(), w_lock_list_free, and witness_watch.

Referenced by witness_lock().

Here is the call graph for this function:

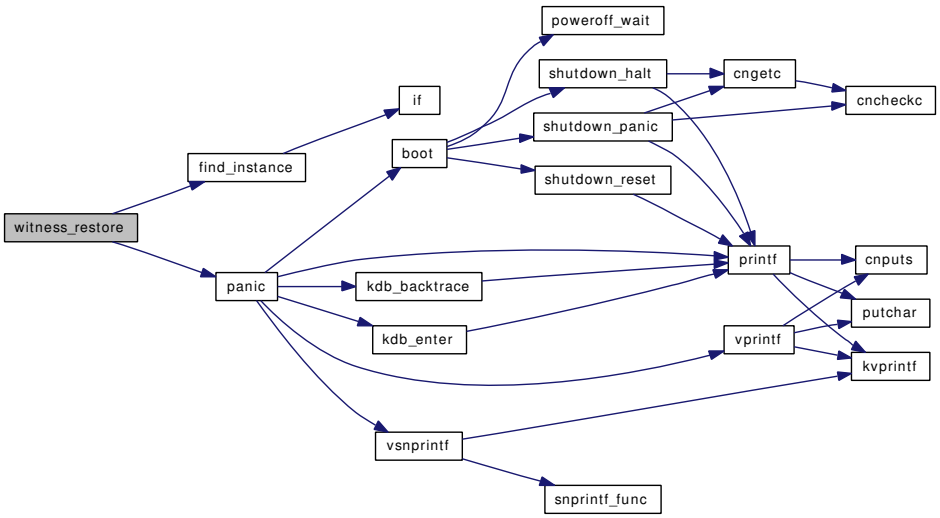


9.119.2.42 void witness_restore (struct lock_object * *lock*, const char * *file*, int *line*)

Definition at line 1839 of file subr_witness.c.

References find_instance(), panic(), panicstr, witness_skipspin, and witness_watch.

Here is the call graph for this function:

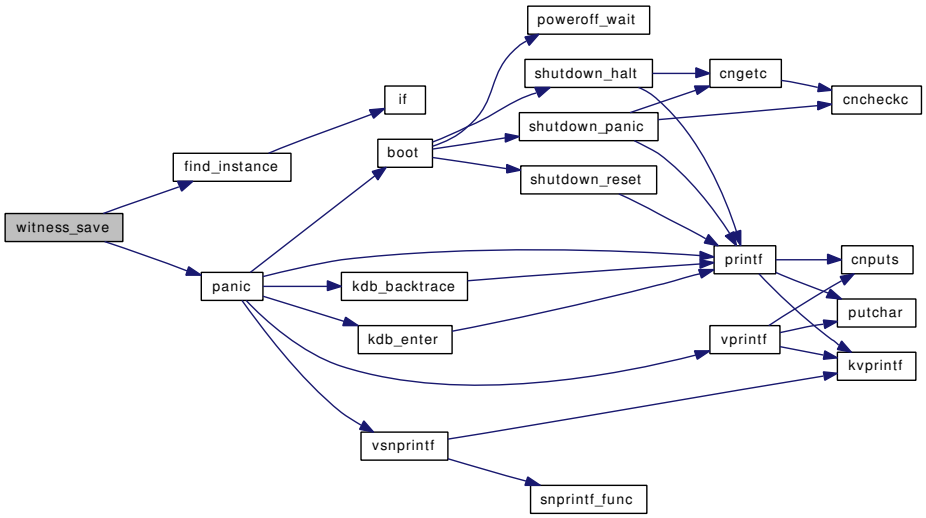


9.119.2.43 void witness_save (struct lock_object * *lock*, const char ** *filep*, int * *linep*)

Definition at line 1813 of file subr_witness.c.

References find_instance(), panic(), panicstr, witness_skipspin, and witness_watch.

Here is the call graph for this function:

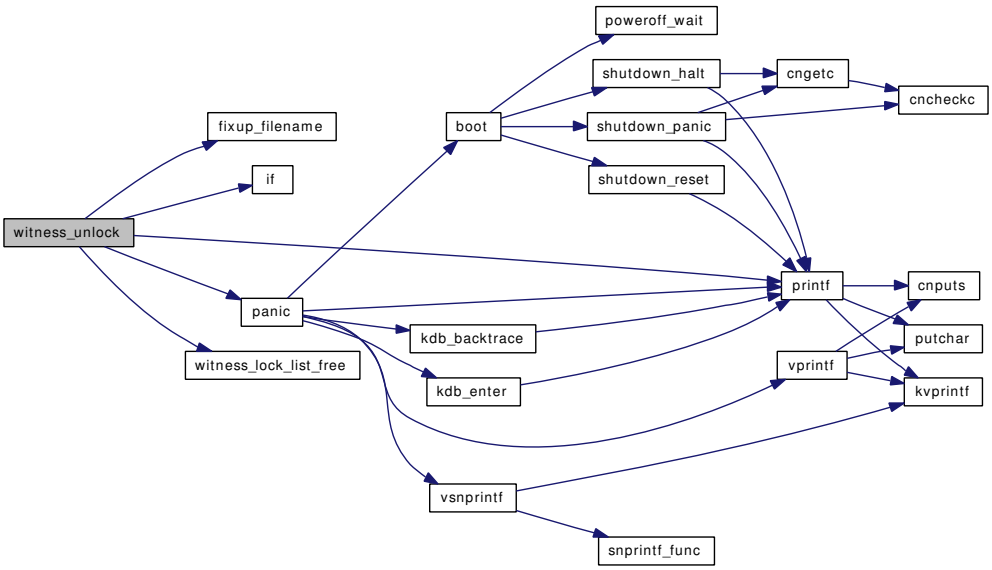


9.119.2.44 void witness_unlock (struct lock_object * *lock*, int *flags*, const char * *file*, int *line*)

Definition at line 1223 of file subr_witness.c.

References fixup_filename(), if(), KTR_WITNESS, panic(), panicstr, printf(), td, witness_lock_list_free(), and witness_watch.

Here is the call graph for this function:

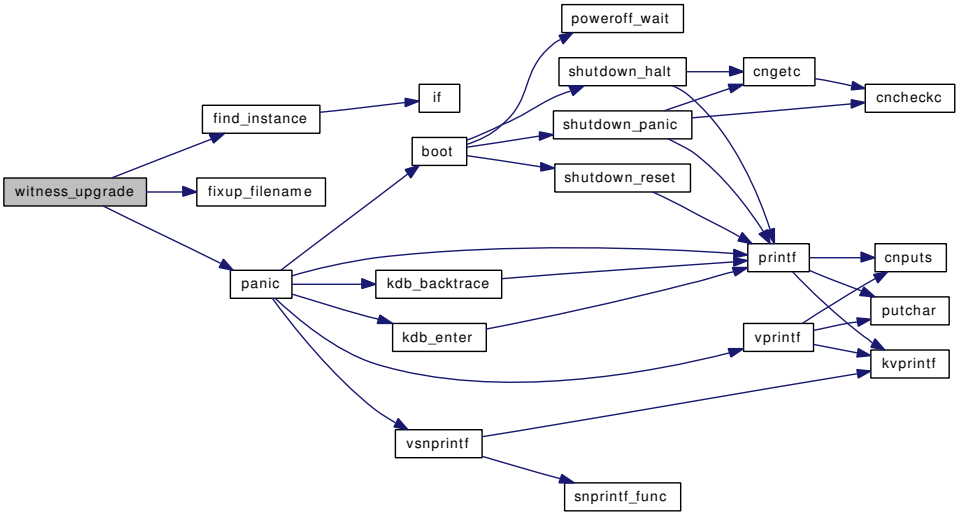


9.119.2.45 void witness_upgrade (struct lock_object * *lock*, int *flags*, const char * *file*, int *line*)

Definition at line 1157 of file subr_witness.c.

References find_instance(), fixup_filename(), panic(), panicstr, and witness_watch.

Here is the call graph for this function:

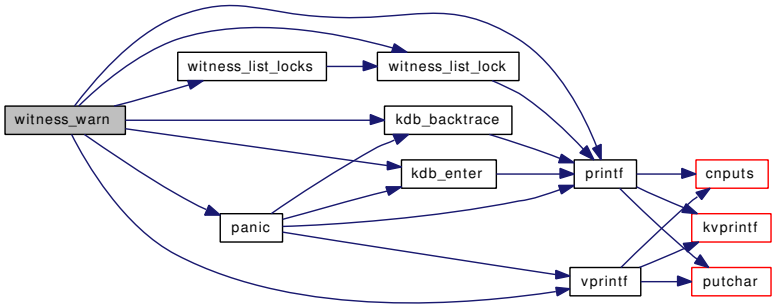


9.119.2.46 int witness_warn (int *flags*, struct lock_object * *lock*, const char * *fmt*, ...)

Definition at line 1310 of file subr_witness.c.

References Giant, kdb_backtrace(), kdb_enter(), panic(), panicstr, printf(), td, vprintf(), witness_list_-lock(), witness_list_locks(), and witness_watch.

Here is the call graph for this function:



9.119.3 Variable Documentation

9.119.3.1 struct witness_order_list_entry order_lists[ ] [static]

Definition at line 273 of file subr_witness.c.

Referenced by STAILQ_HEAD().

9.119.3.2 struct witness_list w_all = STAILQ_HEAD_INITIALIZER(w_all) [static]

Definition at line 253 of file subr_witness.c.

9.119.3.3 int w_child_cnt [static]

Definition at line 259 of file subr_witness.c.

Referenced by depart(), insertchild(), and removechild().

9.119.3.4 struct witness_child_list_entry* w_child_free = NULL [static]

Definition at line 256 of file subr_witness.c.

Referenced by witness_child_free(), and witness_child_get().

9.119.3.5 int w_child_free_cnt [static]

Definition at line 259 of file subr_witness.c.

Referenced by witness_child_free(), and witness_child_get().

9.119.3.6 struct witness_child_list_entry w_childdata[WITNESS_CHILDCOUNT] [static]

Definition at line 270 of file subr_witness.c.

Referenced by STAILQ_HEAD().

9.119.3.7 struct witness w_data[WITNESS_COUNT] [static]

Definition at line 269 of file subr_witness.c.

Referenced by STAILQ_HEAD().

9.119.3.8 struct witness_list w_free = STAILQ_HEAD_INITIALIZER(w_free) [static]

Definition at line 252 of file subr_witness.c.

9.119.3.9 int w_free_cnt [static]

Definition at line 259 of file subr_witness.c.

Referenced by witness_free(), and witness_get().

9.119.3.10 struct lock_list_entry* w_lock_list_free = NULL [static]

Definition at line 257 of file subr_witness.c.

Referenced by witness_lock_list_free(), and witness_lock_list_get().

9.119.3.11 struct lock_list_entry w_locklistdata[LOCK_CHILDCOUNT] [static]

Definition at line 271 of file subr_witness.c.

9.119.3.12 struct mtx w_mtx [static]

Definition at line 251 of file subr_witness.c.

9.119.3.13 struct witness_list w_sleep = STAILQ_HEAD_INITIALIZER(w_sleep) [static]

Definition at line 255 of file subr_witness.c.

9.119.3.14 int w_sleep_cnt [static]

Definition at line 259 of file subr_witness.c.

Referenced by depart(), and enroll().

9.119.3.15 struct witness_list w_spin = STAILQ_HEAD_INITIALIZER(w_spin) [static]

Definition at line 254 of file subr_witness.c.

9.119.3.16 int w_spin_cnt [static]

Definition at line 259 of file subr_witness.c.

Referenced by depart(), and enroll().

9.119.3.17 int witness_skipspin = 0

Definition at line 245 of file subr_witness.c.

Referenced by enroll(), witness_restore(), and witness_save().

9.119.3.18 int witness_watch = 1 [static]

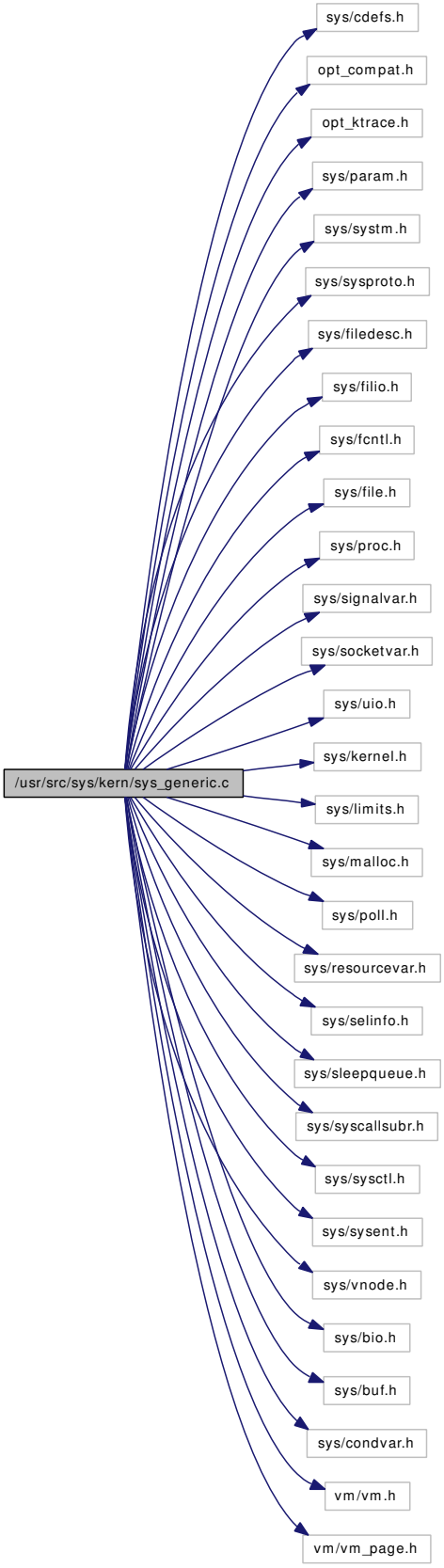
Definition at line 211 of file subr_witness.c.

Referenced by enroll(), sysctl_debug_witness_watch(), witness_assert(), witness_checkorder(), witness_child_get(), witness_defineorder(), witness_downgrade(), witness_file(), witness_get(), witness_init(), witness_line(), witness_lock(), witness_lock_list_get(), witness_restore(), witness_save(), witness_unlock(), witness_upgrade(), and witness_warn().

9.120 /usr/src/sys/kern/sys_generic.c File Reference

```
#include <sys/cdefs.h>
#include "opt_compat.h"
#include "opt_ktrace.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/sysproto.h>
#include <sys/filedesc.h>
#include <sys/filio.h>
#include <sys/fcntl.h>
#include <sys/file.h>
#include <sys/proc.h>
#include <sys/signalvar.h>
#include <sys/socketvar.h>
#include <sys/uio.h>
#include <sys/kernel.h>
#include <sys/limits.h>
#include <sys/malloc.h>
#include <sys/poll.h>
#include <sys/resourcevar.h>
#include <sys/selinfo.h>
#include <sys/sleepqueue.h>
#include <sys/syscallsubr.h>
#include <sys/sysctl.h>
#include <sys/sysent.h>
#include <sys/vnode.h>
#include <sys/bio.h>
#include <sys/buf.h>
#include <sys/condvar.h>
#include <vm/vm.h>
#include <vm/vm_page.h>
```

Include dependency graph for sys_generic.c:


/usr/src/sys/kern/sys_generic.c
sys/cdefs.h
opt_compat.h
opt_ktrace.h
sys/param.h
sys/systm.h
sys/sysproto.h
sys/filedesc.h
sys/filio.h
sys/fcntl.h
sys/file.h
sys/proc.h
sys/signalvar.h
sys/socketvar.h
sys/uio.h
sys/kernel.h
sys/limits.h
sys/malloc.h
sys/poll.h
sys/resourcevar.h
sys/selinfo.h
sys/sleepqueue.h
sys/syscallsubr.h
sys/sysctl.h
sys/sysent.h
sys/vnode.h
sys/bio.h
sys/buf.h
sys/condvar.h
vm/vm.h
vm/vm_page.h

Data Structures

- struct read_args
- struct pread_args
- struct readv_args
- struct preadv_args
- struct write_args
- struct pwrite_args
- struct writev_args
- struct pwritev_args
- struct ioctl_args
- struct select_args
- struct poll_args
- struct openbsd_poll_args

Defines

- #define getbits(name, x)
- #define putbits(name, x)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/sys_generic.c,v 1.151 2006/10/14 19:01:55 ru Exp $")
- static MALLOC_DEFINE (M_IOCTLOPS,"ioctlops","ioctl data buffer")
- static MALLOC_DEFINE (M_SELECT,"select","select() buffer")
- MALLOC_DEFINE (M_IOV,"iov","large iov's")
- static int pollscan (struct thread *, struct pollfd *, u_int)
- static int selscan (struct thread *, fd_mask **, fd_mask **, int)
- static int dofileread (struct thread *, int, struct file *, struct uio *, off_t, int)
- static int dofilewrite (struct thread *, int, struct file *, struct uio *, off_t, int)
- static void doselwakeup (struct selinfo *, int)
- int read (struct thread *td, struct read_args *uap)
- int pread (struct thread *td, struct pread_args *uap)
- int readv (struct thread *td, struct readv_args *uap)
- int kern_readv (struct thread *td, int fd, struct uio *auio)
- int preadv (struct thread *td, struct preadv_args *uap)
- int kern_preadv (struct thread *td, int fd, struct uio *auio, off_t offset)
- int write (struct thread *td, struct write_args *uap)
- int pwrite (struct thread *td, struct pwrite_args *uap)
- int writev (struct thread *td, struct writev_args *uap)
- int kern_writev (struct thread *td, int fd, struct uio *auio)
- int pwritev (struct thread *td, struct pwritev_args *uap)
- int kern_pwritev (struct thread *td, int fd, struct uio *auio, off_t offset)
- int ioctl (struct thread *td, struct ioctl_args *uap)
- int kern_ioctl (struct thread *td, int fd, u_long com, caddr_t data)
- SYSCTL_UINT (_kern, OID_AUTO, nselcoll, CTLFLAG_RD,&nselcoll, 0,"")
- int select (struct thread *td, struct select_args *uap)
- int kern_select (struct thread *td, int nd, fd_set *fd_in, fd_set *fd_ou, fd_set *fd_ex, struct timeval *tvp)
- int poll (struct thread *td, struct poll_args *uap)

- int openbsd_poll (struct thread *td, struct openbsd_poll_args *uap)
- void clear_selinfo_list (struct thread *td)
- void selrecord (struct thread *selector, struct selinfo *sip)
- void selwakeup (struct selinfo *sip)
- void selwakeuppri (struct selinfo *sip, int pri)
- static void selectinit (void *)

Variables

- mtx sellock
- cv selwait
- u_int nselcoll

9.120.1 Define Documentation

9.120.1.1 #define getbits(name, x)

Value:

```
do {                                                            \
                if (name == NULL)                                       \
                        ibits[x] = NULL;                                \
                else {                                                  \
                        ibits[x] = sbp + nbufbytes / 2 / sizeof *sbp;   \
                        obits[x] = sbp;                                 \
                        sbp += ncpbytes / sizeof *sbp;                  \
                        error = copyin(name, ibits[x], ncpbytes);       \
                        if (error != 0)                                 \
                                goto done_nosellock;                    \
                }                                                       \
        } while (0)
```

Referenced by kern_select().

9.120.1.2 #define putbits(name, x)

Value:

```
if (name && (error2 = copyout(obits[x], name, ncpbytes))) \
                error = error2;
```

Referenced by kern_select().

9.120.2 Function Documentation

9.120.2.1 __FBSDID ("$FreeBSD: src/sys/kern/sys_generic. *c*, v 1.151 2006/10/14 19:01:55 ru Exp $")

9.120.2.2 void clear_selinfo_list (struct thread * *td*)

Definition at line 1084 of file sys_generic.c.

References sellock.

Referenced by kern_select(), and poll().

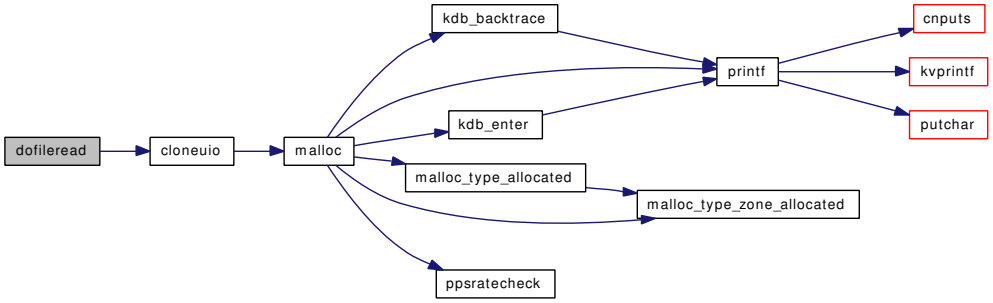
9.120.2.3 static int dofileread (struct thread *, int, struct file *, struct uio *, off_t, int) [static]

Definition at line 253 of file sys_generic.c.

References cloneuio().

Referenced by kern_preadv(), and kern_readv().

Here is the call graph for this function:



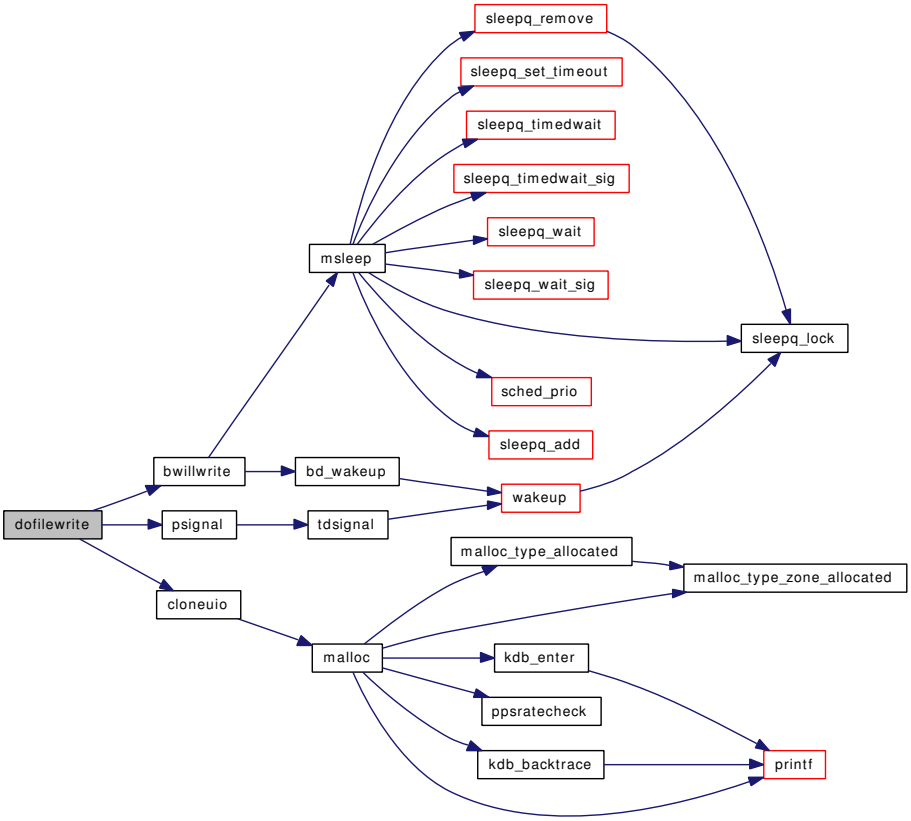
9.120.2.4 static int dofilewrite (struct thread *, int, struct file *, struct uio *, off_t, int) [static]

Definition at line 463 of file sys_generic.c.

References bwillwrite(), cloneuio(), and psignal().

Referenced by kern_pwritev(), and kern_writev().

Here is the call graph for this function:



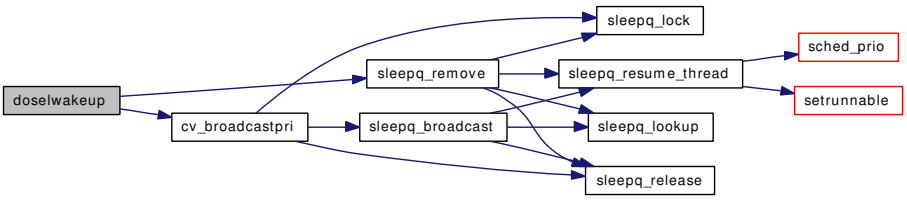
9.120.2.5 static void doselwakeup (struct selinfo *, int) [static]

Definition at line 1146 of file sys_generic.c.

References cv_broadcastpri(), nselcoll, sched_lock, sellock, selwait, sleepq_remove(), and td.

Referenced by selwakeup(), and selwakeuppri().

Here is the call graph for this function:

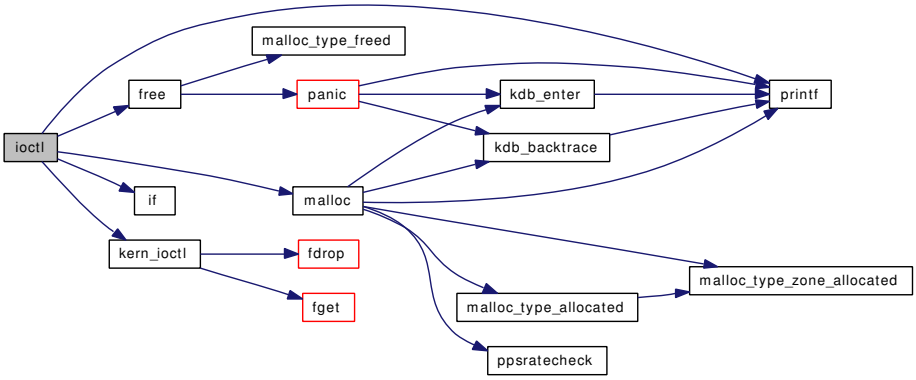


9.120.2.6 int ioctl (struct thread * *td*, struct ioctl_args * *uap*)

Definition at line 524 of file sys_generic.c.

References ioctl_args::com, ioctl_args::data, ioctl_args::fd, free(), if(), kern_ioctl(), malloc(), and printf().

Here is the call graph for this function:



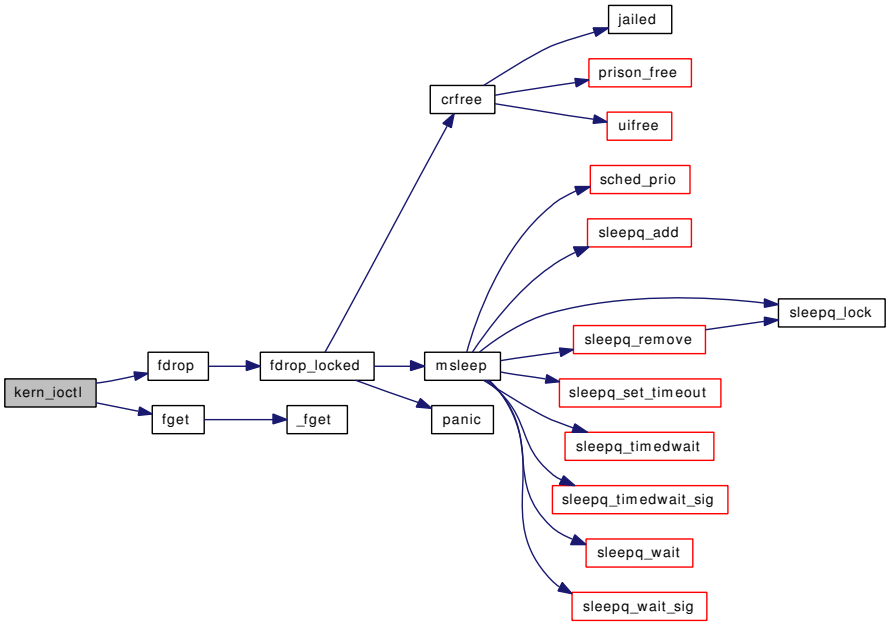
9.120.2.7 int kern_ioctl (struct thread * *td*, int *fd*, u_long *com*, caddr_t *data*)

Definition at line 591 of file sys_generic.c.

References fdrop(), and fget().

Referenced by ioctl().

Here is the call graph for this function:



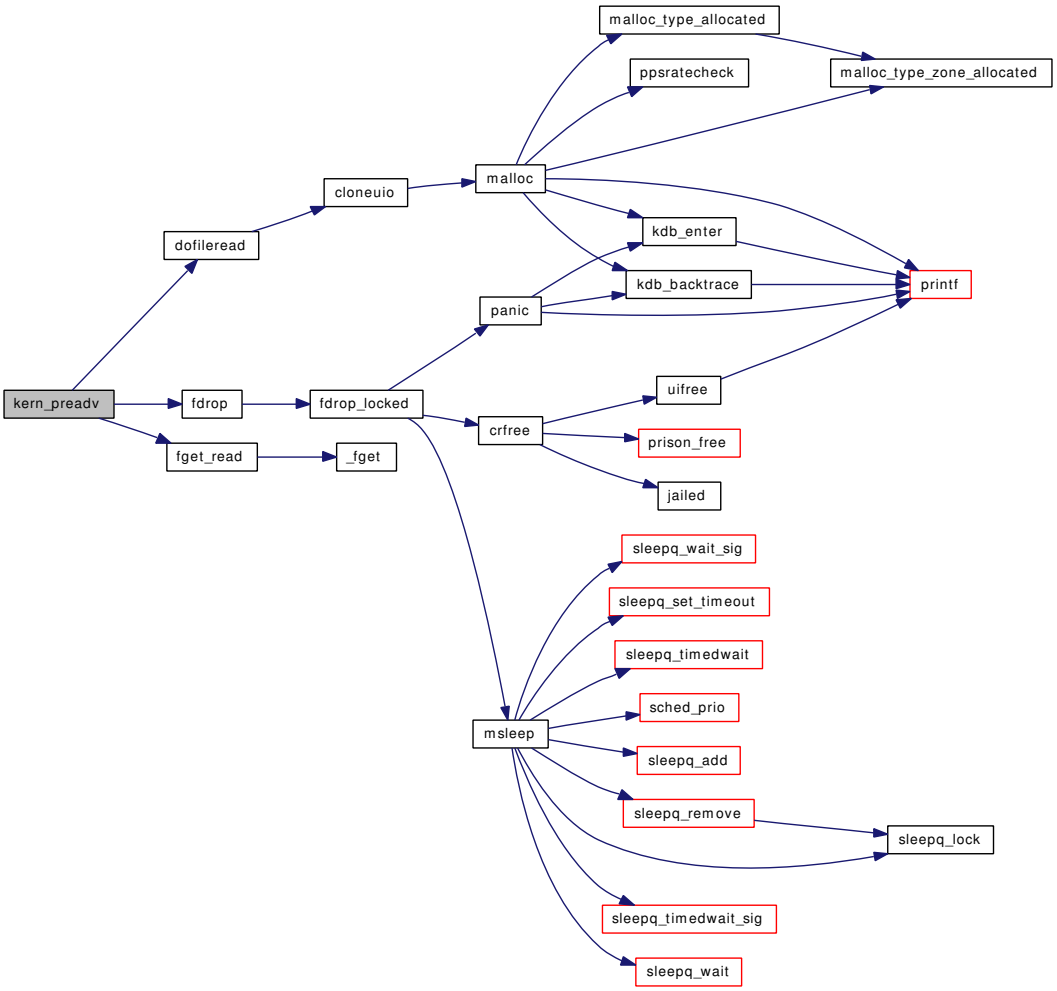
9.120.2.8 int kern_preadv (struct thread * *td*, int *fd*, struct uio * *auio*, off_t *offset*)

Definition at line 226 of file sys_generic.c.

References dofileread(), fdrop(), and fget_read().

Referenced by pread(), and preadv().

Here is the call graph for this function:



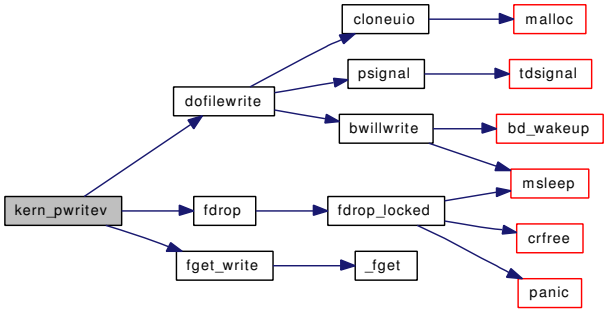
9.120.2.9 int kern_pwritev (struct thread * *td*, int *fd*, struct uio * *auio*, off_t *offset*)

Definition at line 436 of file sys_generic.c.

References dofilewrite(), fdrop(), and fget_write().

Referenced by pwrite(), and pwritev().

Here is the call graph for this function:



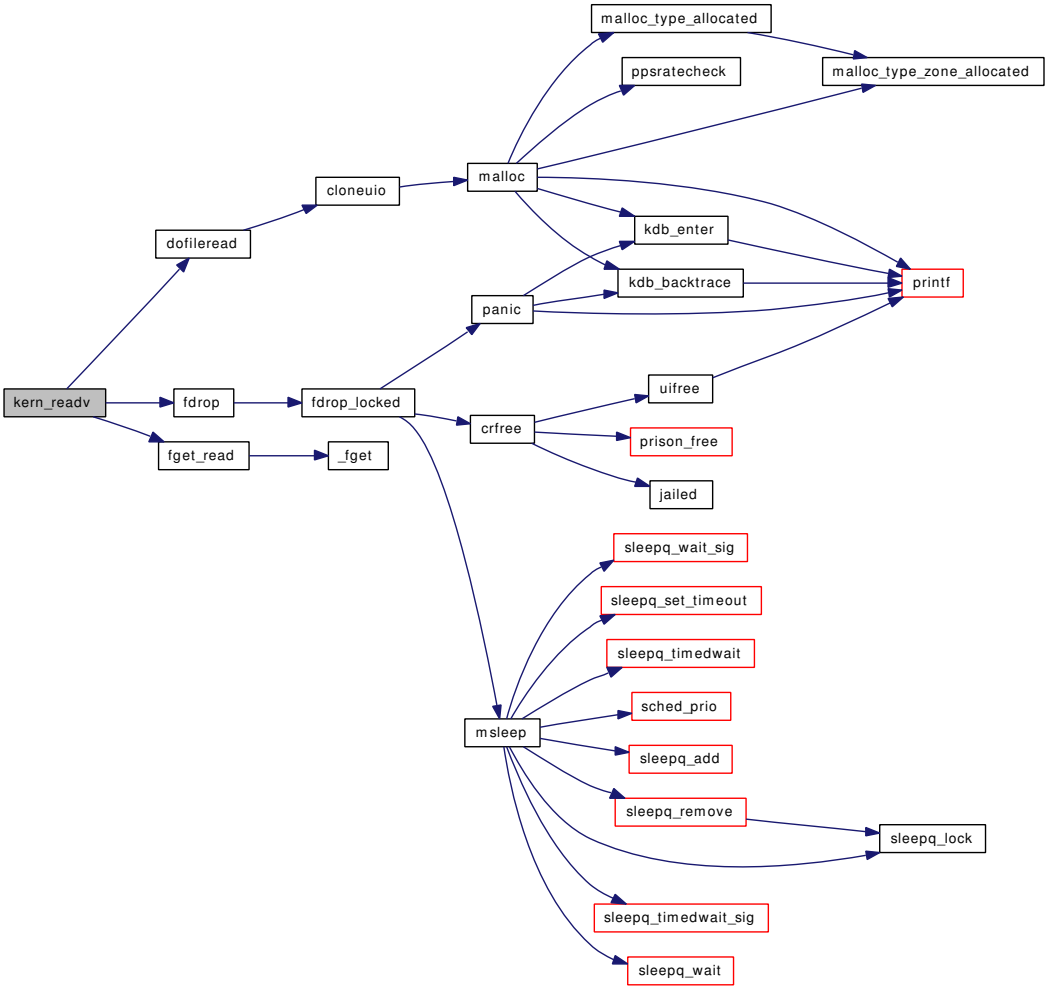
9.120.2.10 int kern_readv (struct thread * *td*, int *fd*, struct uio * *auio*)

Definition at line 184 of file sys_generic.c.

References dofileread(), fdrop(), and fget_read().

Referenced by read(), and readv().

Here is the call graph for this function:



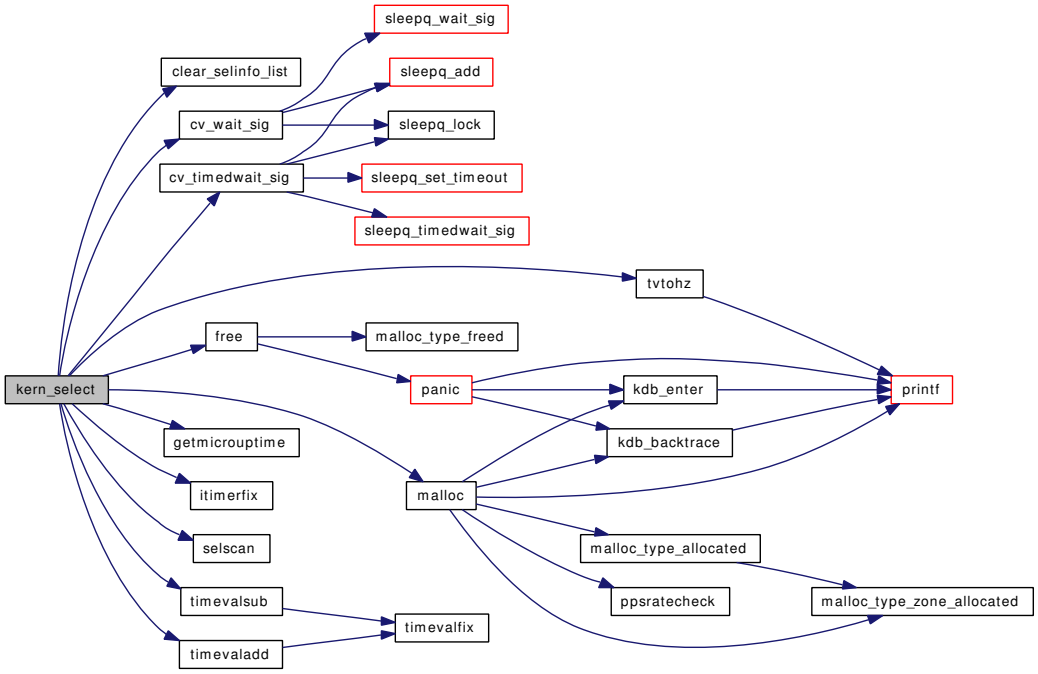
9.120.2.11 int kern_select (struct thread * *td*, int *nd*, fd_set * *fd_in*, fd_set * *fd_ou*, fd_set * *fd_ex*, struct timeval * *tvp*)

Definition at line 683 of file sys_generic.c.

References clear_selinfo_list(), cv_timedwait_sig(), cv_wait_sig(), free(), getbits, getmicrouptime(), hz, itimerfix(), malloc(), nselcoll, putbits, sched_lock, sellock, selscan(), selwait, timevaladd(), timevalsub(), and tvtohz().

Referenced by select().

Here is the call graph for this function:



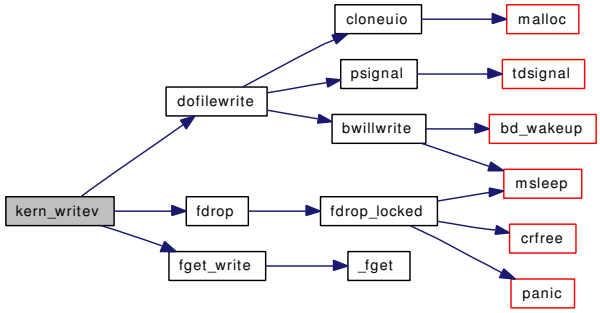
9.120.2.12 int kern_writev (struct thread * *td*, int *fd*, struct uio * *auio*)

Definition at line 394 of file sys_generic.c.

References dofilewrite(), fdrop(), and fget_write().

Referenced by kern_sendfile(), write(), and writev().

Here is the call graph for this function:



9.120.2.13 MALLOC_DEFINE (M_IOV, "iov", "large iov's")

9.120.2.14 static MALLOC_DEFINE (M_SELECT, "select", "select() buffer") [static]

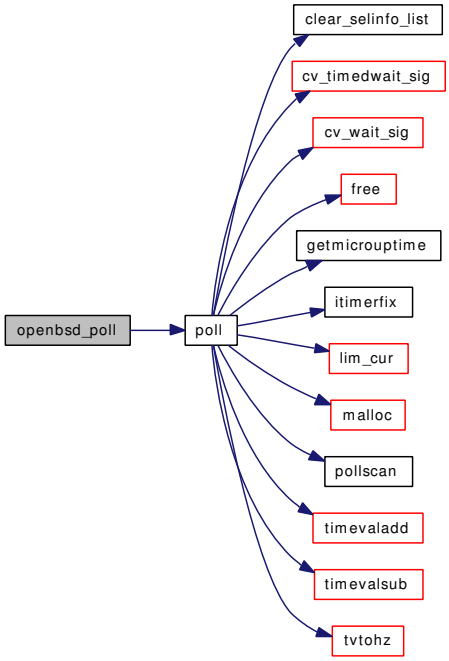
9.120.2.15 static MALLOC_DEFINE (M_IOCTLOPS, "ioctlops", "ioctl data buffer") [static]

9.120.2.16 int openbsd_poll (struct thread * *td*, struct openbsd_poll_args * *uap*)

Definition at line 1068 of file sys_generic.c.

References poll().

Here is the call graph for this function:



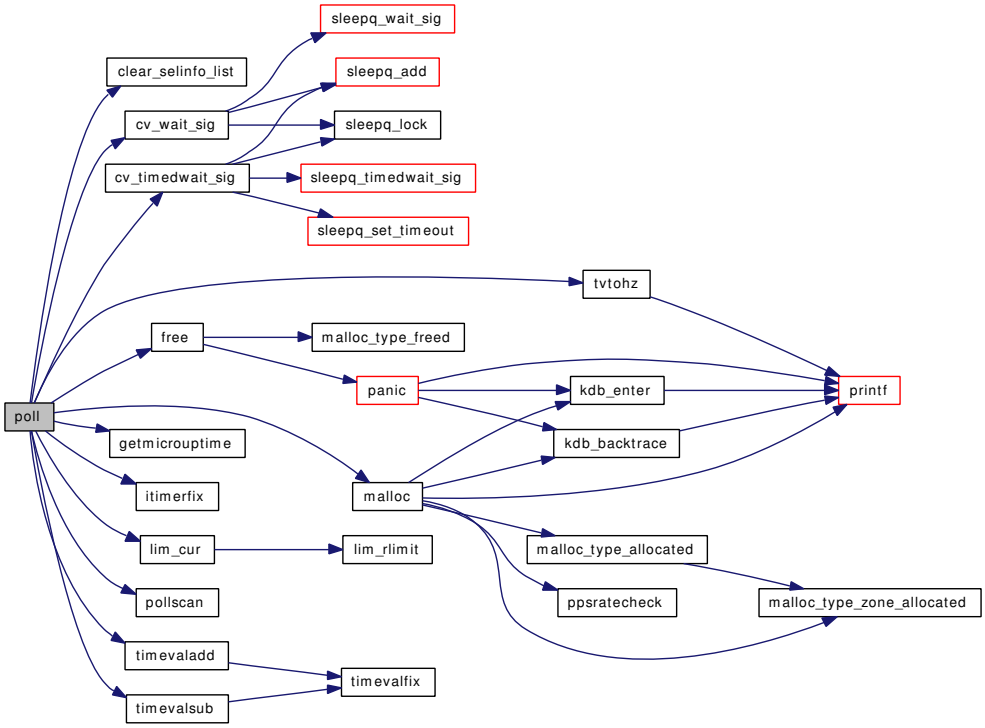
9.120.2.17 int poll (struct thread * *td*, struct poll_args * *uap*)

Definition at line 896 of file sys_generic.c.

References clear_selinfo_list(), cv_timedwait_sig(), cv_wait_sig(), poll_args::fds, free(), getmicrouptime(), hz, itimerfix(), lim_cur(), malloc(), poll_args::nfds, nselcoll, pollscan(), sched_lock, sellock, selwait, poll_args::timeout, timevaladd(), timevalsub(), and tvtohz().

Referenced by openbsd_poll().

Here is the call graph for this function:



9.120.2.18 static int pollscan (struct thread *, struct pollfd *, u_int) [static]

Definition at line 1014 of file sys_generic.c.

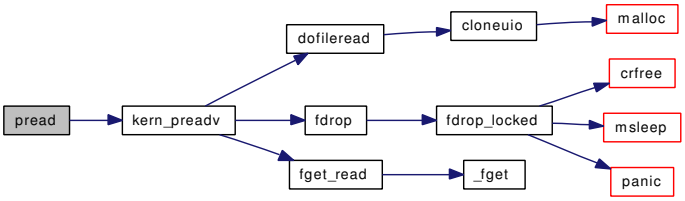
Referenced by poll().

9.120.2.19 int pread (struct thread * *td*, struct pread_args * *uap*)

Definition at line 136 of file sys_generic.c.

References pread_args::buf, pread_args::fd, kern_preadv(), pread_args::nbyte, and pread_args::offset.

Here is the call graph for this function:

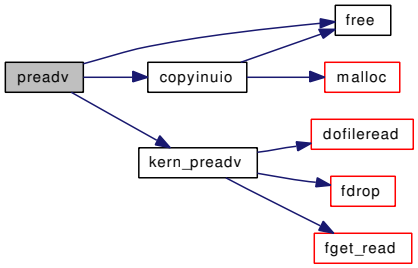


9.120.2.20 int preadv (struct thread * *td*, struct preadv_args * *uap*)

Definition at line 212 of file sys_generic.c.

References copyinuio(), preadv_args::fd, free(), preadv_args::iovcnt, preadv_args::iovp, kern_preadv(), and preadv_args::offset.

Here is the call graph for this function:

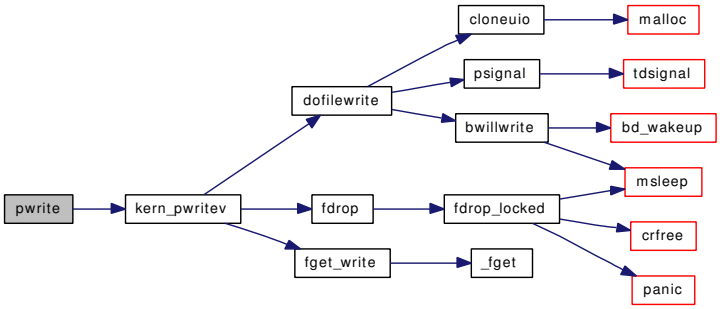


9.120.2.21 int pwrite (struct thread * *td*, struct pwrite_args * *uap*)

Definition at line 346 of file sys_generic.c.

References pwrite_args::buf, pwrite_args::fd, kern_pwritev(), pwrite_args::nbyte, and pwrite_args::offset.

Here is the call graph for this function:

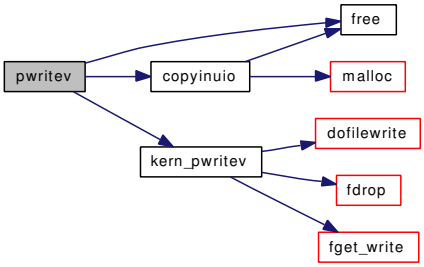


9.120.2.22 int pwritev (struct thread * *td*, struct pwritev_args * *uap*)

Definition at line 422 of file sys_generic.c.

References copyinuio(), pwritev_args::fd, free(), pwritev_args::iovcnt, pwritev_args::iovp, kern_pwritev(), and pwritev_args::offset.

Here is the call graph for this function:



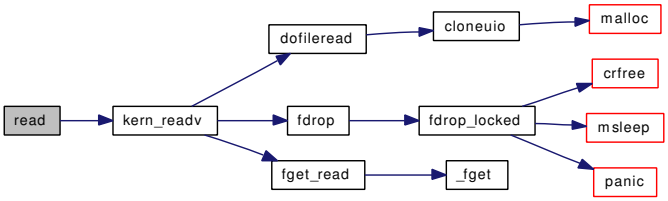
9.120.2.23 int read (struct thread * *td*, struct read_args * *uap*)

Definition at line 100 of file sys_generic.c.

References read_args::buf, read_args::fd, kern_readv(), and read_args::nbyte.

Referenced by ttread().

Here is the call graph for this function:

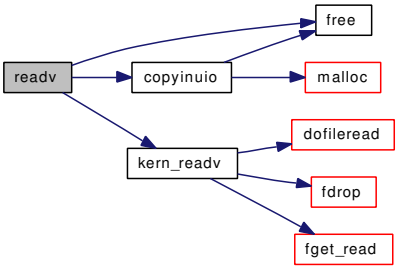


9.120.2.24 int readv (struct thread * *td*, struct readv_args * *uap*)

Definition at line 170 of file sys_generic.c.

References copyinuio(), readv_args::fd, free(), readv_args::iovcnt, readv_args::iovp, and kern_readv().

Here is the call graph for this function:

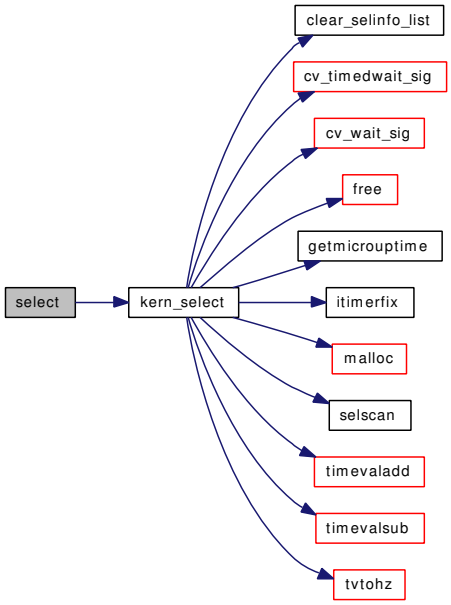


9.120.2.25 int select (struct thread * *td*, struct select_args * *uap*)

Definition at line 664 of file sys_generic.c.

References kern_select().

Here is the call graph for this function:

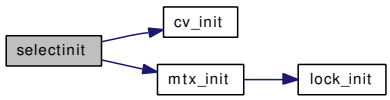


9.120.2.26 static void selectinit (void *) [static]

Definition at line 1177 of file sys_generic.c.

References cv_init(), mtx_init(), sellock, and selwait.

Here is the call graph for this function:



9.120.2.27 void selrecord (struct thread * *selector*, struct selinfo * *sip*)

Definition at line 1099 of file sys_generic.c.

References sellock.

Referenced by devpoll(), kqueue_poll(), logpoll(), mqf_poll(), pipe_poll(), ptcpoll(), sopoll_generic(), ttypoll(), and vn_pollrecord().

9.120.2.28 static int selscan (struct thread *, fd_mask **, fd_mask **, int) [static]

Definition at line 841 of file sys_generic.c.

Referenced by kern_select().

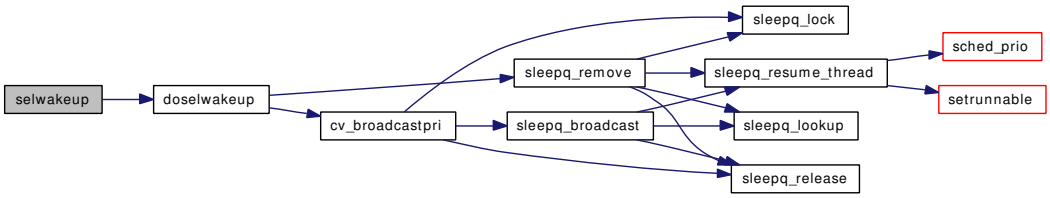
9.120.2.29 void selwakeup (struct selinfo * *sip*)

Definition at line 1127 of file sys_generic.c.

References doselwakeup().

Referenced by _mqueue_recv(), _mqueue_send(), devctl_queue_data(), mqueue_fdclose(), and ptcwakeup().

Here is the call graph for this function:



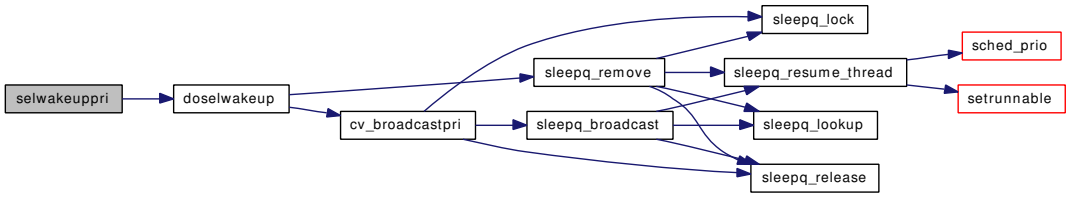
9.120.2.30 void selwakeuppri (struct selinfo * *sip*, int *pri*)

Definition at line 1135 of file sys_generic.c.

References doselwakeup().

Referenced by kqueue_wakeup(), logtimeout(), pipeselwakeup(), ptcwakeup(), sohasoutofband(), sowakeup(), ttwakeup(), and ttwwakeup().

Here is the call graph for this function:



9.120.2.31 SYSCTL_UINT (_kern, OID_AUTO, nselcoll, CTLFLAG_RD, & *nselcoll*, 0, "")

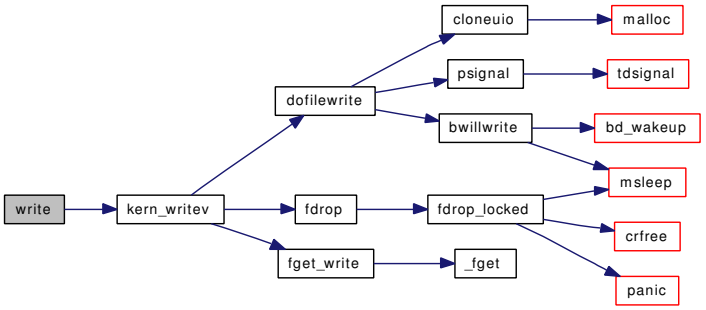
9.120.2.32 int write (struct thread * *td*, struct write_args * *uap*)

Definition at line 310 of file sys_generic.c.

References write_args::buf, write_args::fd, kern_writev(), and write_args::nbyte.

Referenced by kern_ptrace().

Here is the call graph for this function:

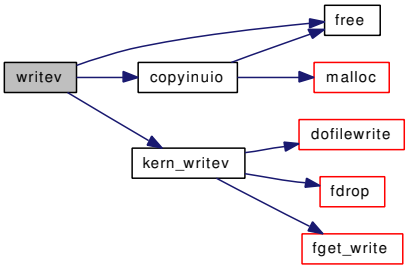


9.120.2.33 int writev (struct thread * *td*, struct writev_args * *uap*)

Definition at line 380 of file sys_generic.c.

References copyinuio(), writev_args::fd, free(), writev_args::iovcnt, writev_args::iovp, and kern_writev().

Here is the call graph for this function:



9.120.3 Variable Documentation

9.120.3.1 u_int nselcoll

Definition at line 647 of file sys_generic.c.

Referenced by doselwakeup(), kern_select(), and poll().

9.120.3.2 struct mtx sellock

Definition at line 645 of file sys_generic.c.

Referenced by clear_selinfo_list(), doselwakeup(), kern_select(), poll(), selectinit(), and selrecord().

9.120.3.3 struct cv selwait

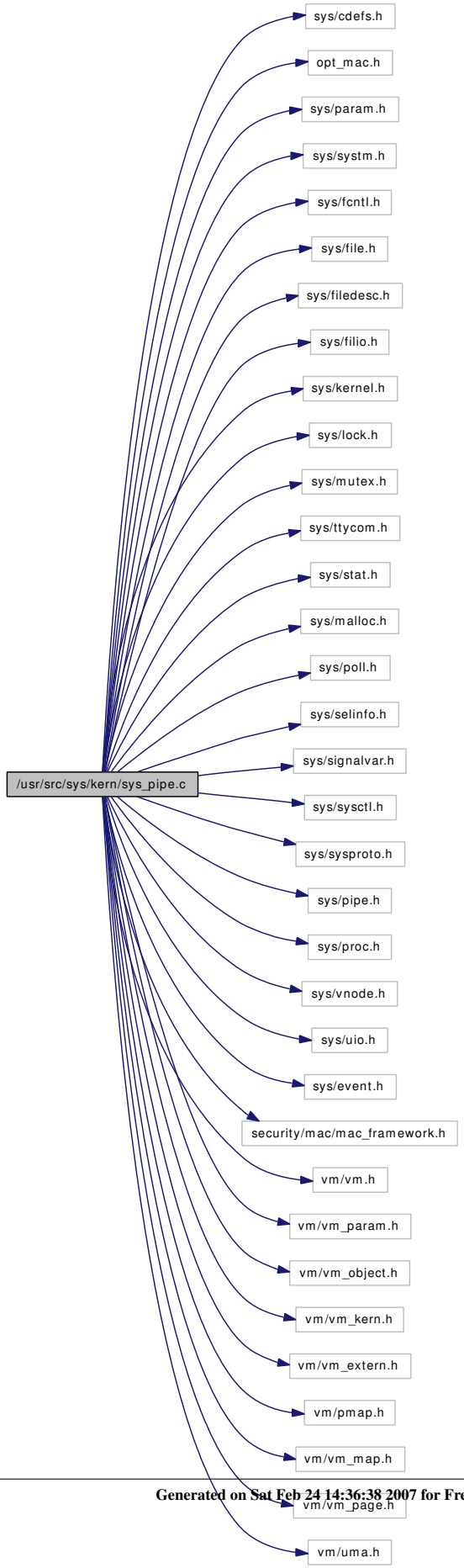
Definition at line 646 of file sys_generic.c.

Referenced by doselwakeup(), kern_select(), poll(), and selectinit().

9.121 /usr/src/sys/kern/sys_pipe.c File Reference

```
#include <sys/cdefs.h>
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/fcntl.h>
#include <sys/file.h>
#include <sys/filedesc.h>
#include <sys/filio.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/ttycom.h>
#include <sys/stat.h>
#include <sys/malloc.h>
#include <sys/poll.h>
#include <sys/selinfo.h>
#include <sys/signalvar.h>
#include <sys/sysctl.h>
#include <sys/sysproto.h>
#include <sys/pipe.h>
#include <sys/proc.h>
#include <sys/vnode.h>
#include <sys/uio.h>
#include <sys/event.h>
#include <security/mac/mac_framework.h>
#include <vm/vm.h>
#include <vm/vm_param.h>
#include <vm/vm_object.h>
#include <vm/vm_kern.h>
#include <vm/vm_extern.h>
#include <vm/pmap.h>
#include <vm/vm_map.h>
#include <vm/vm_page.h>
#include <vm/uma.h>
```

Include dependency graph for sys_pipe.c:


/usr/src/sys/kern/sys_pipe.c
sys/cdefs.h
opt_mac.h
sys/param.h
sys/systm.h
sys/fcntl.h
sys/file.h
sys/filedesc.h
sys/filio.h
sys/kernel.h
sys/lock.h
sys/mutex.h
sys/ttycom.h
sys/stat.h
sys/malloc.h
sys/poll.h
sys/selinfo.h
sys/signalvar.h
sys/sysctl.h
sys/sysproto.h
sys/pipe.h
sys/proc.h
sys/vnode.h
sys/uio.h
sys/event.h
security/mac/mac_framework.h
vm/vm.h
vm/vm_param.h
vm/vm_object.h
vm/vm_kern.h
vm/vm_extern.h
vm/pmap.h
vm/vm_map.h
vm/vm_page.h
vm/uma.h

Defines

- #define MINPIPESIZE (PIPE_SIZE/3)
- #define MAXPIPESIZE (2∗PIPE_SIZE/3)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/sys_pipe.c,v 1.189 2006/12/19 12:52:22 pjd Exp $")
- static void filt_pipedetach (struct knote ∗kn)
- static int filt_piperead (struct knote ∗kn, long hint)
- static int filt_pipewrite (struct knote ∗kn, long hint)
- SYSCTL_INT (_kern_ipc, OID_AUTO, maxpipekva, CTLFLAG_RDTUN,&maxpipekva, 0,"Pipe KVA limit")
- SYSCTL_INT (_kern_ipc, OID_AUTO, pipes, CTLFLAG_RD,&amountpipes, 0,"Current # of pipes")
- SYSCTL_INT (_kern_ipc, OID_AUTO, pipekva, CTLFLAG_RD,&amountpipekva, 0,"Pipe KVA usage")
- SYSCTL_INT (_kern_ipc, OID_AUTO, pipefragretry, CTLFLAG_RD,&pipefragretry, 0,"Pipe allocation retries due to fragmentation")
- SYSCTL_INT (_kern_ipc, OID_AUTO, pipeallocfail, CTLFLAG_RD,&pipeallocfail, 0,"Pipe allocation failures")
- SYSCTL_INT (_kern_ipc, OID_AUTO, piperesizefail, CTLFLAG_RD,&piperesizefail, 0,"Pipe resize failures")
- SYSCTL_INT (_kern_ipc, OID_AUTO, piperesizeallowed, CTLFLAG_RW,&piperesizeallowed, 0,"Pipe resizing allowed")
- static void pipeinit (void ∗dummy __unused)
- static void pipeclose (struct pipe ∗cpipe)
- static void pipe_free_kmem (struct pipe ∗cpipe)
- static int pipe_create (struct pipe ∗pipe, int backing)
- static __inline int pipelock (struct pipe ∗cpipe, int catch)
- static __inline void pipeunlock (struct pipe ∗cpipe)
- static __inline void pipeselwakeup (struct pipe ∗cpipe)
- static int pipe_build_write_buffer (struct pipe ∗wpipe, struct uio ∗uio)
- static void pipe_destroy_write_buffer (struct pipe ∗wpipe)
- static int pipe_direct_write (struct pipe ∗wpipe, struct uio ∗uio)
- static void pipe_clone_write_buffer (struct pipe ∗wpipe)
- static int pipespace (struct pipe ∗cpipe, int size)
- static int pipespace_new (struct pipe ∗cpipe, int size)
- static int pipe_zone_ctor (void ∗mem, int size, void ∗arg, int flags)
- static void pipe_zone_dtor (void ∗mem, int size, void ∗arg)
- static int pipe_zone_init (void ∗mem, int size, int flags)
- static void pipe_zone_fini (void ∗mem, int size)
- SYSINIT (vfs, SI_SUB_VFS, SI_ORDER_ANY, pipeinit, NULL)
- int pipe (struct thread ∗td, struct pipe_args ∗uap)
- static int pipe_read (struct file ∗fp, struct uio ∗uio, struct ucred ∗active_cred, int flags, struct thread ∗td)
- static int pipe_write (struct file ∗fp, struct uio ∗uio, struct ucred ∗active_cred, int flags, struct thread ∗td)
- static int pipe_ioctl (struct file ∗fp, u_long cmd, void ∗data, struct ucred ∗active_cred, struct thread ∗td)
- static int pipe_poll (struct file ∗fp, int events, struct ucred ∗active_cred, struct thread ∗td)

- static int pipe_stat (struct file *fp, struct stat *ub, struct ucred *active_cred, struct thread *td)
- static int pipe_close (struct file *fp, struct thread *td)
- static int pipe_kqfilter (struct file *fp, struct knote *kn)

Variables

- static fo_rdwr_t pipe_read
- static fo_rdwr_t pipe_write
- static fo_ioctl_t pipe_ioctl
- static fo_poll_t pipe_poll
- static fo_kqfilter_t pipe_kqfilter
- static fo_stat_t pipe_stat
- static fo_close_t pipe_close
- static struct fileops pipeops
- static struct filterops pipe_rfiltops
- static struct filterops pipe_wfiltops
- static int amountpipes
- static int amountpipekva
- static int pipefragretry
- static int pipeallocfail
- static int piperesizefail
- static int piperesizeallowed = 1
- static uma_zone_t pipe_zone

9.121.1 Define Documentation

9.121.1.1 #define MAXPIPESIZE (2*PIPE_SIZE/3)

Definition at line 176 of file sys_pipe.c.

9.121.1.2 #define MINPIPESIZE (PIPE_SIZE/3)

Definition at line 175 of file sys_pipe.c.

Referenced by pipe_read().

9.121.2 Function Documentation

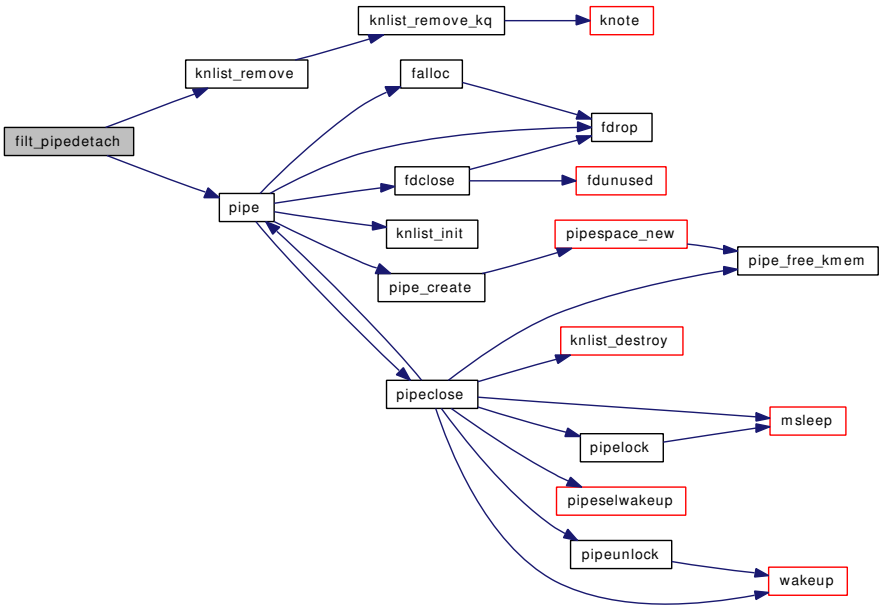
9.121.2.1 __FBSDID ("$FreeBSD: src/sys/kern/sys_pipe. *c*, v 1.189 2006/12/19 12:52:22 pjd Exp $")

9.121.2.2 static void filt_pipedetach (struct knote * *kn*) [static]

Definition at line 1571 of file sys_pipe.c.

References knlist_remove(), and pipe().

Here is the call graph for this function:

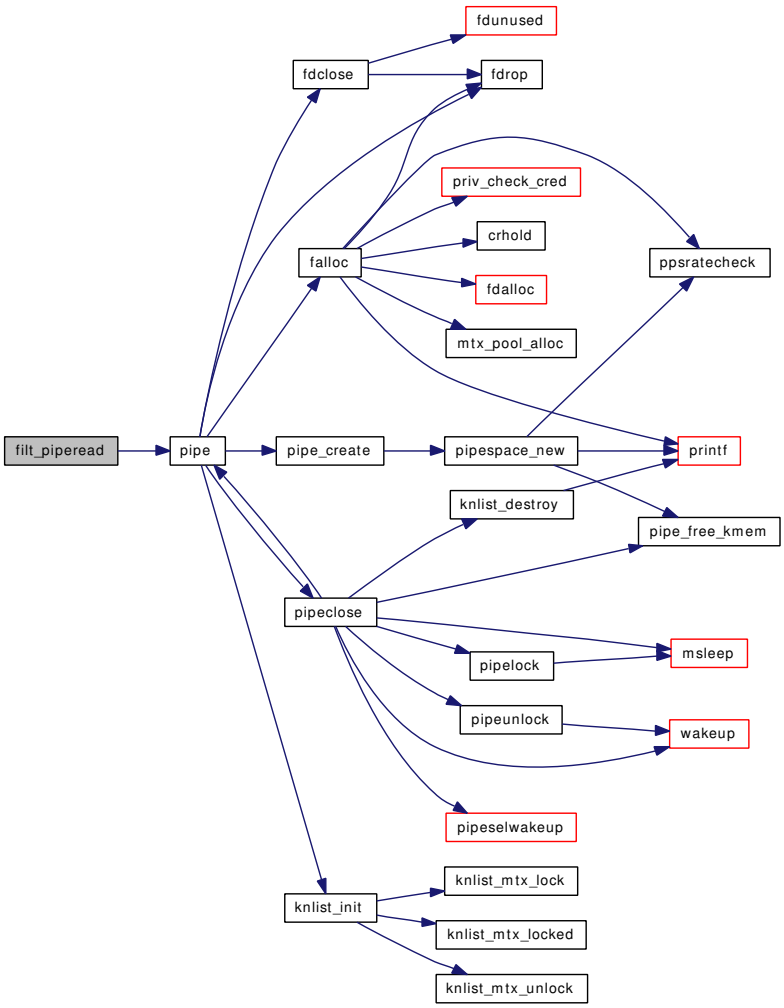


9.121.2.3 static int filt_piperead (struct knote * *kn*, long *hint*) [static]

Definition at line 1589 of file sys_pipe.c.

References pipe(), and ret.

Here is the call graph for this function:

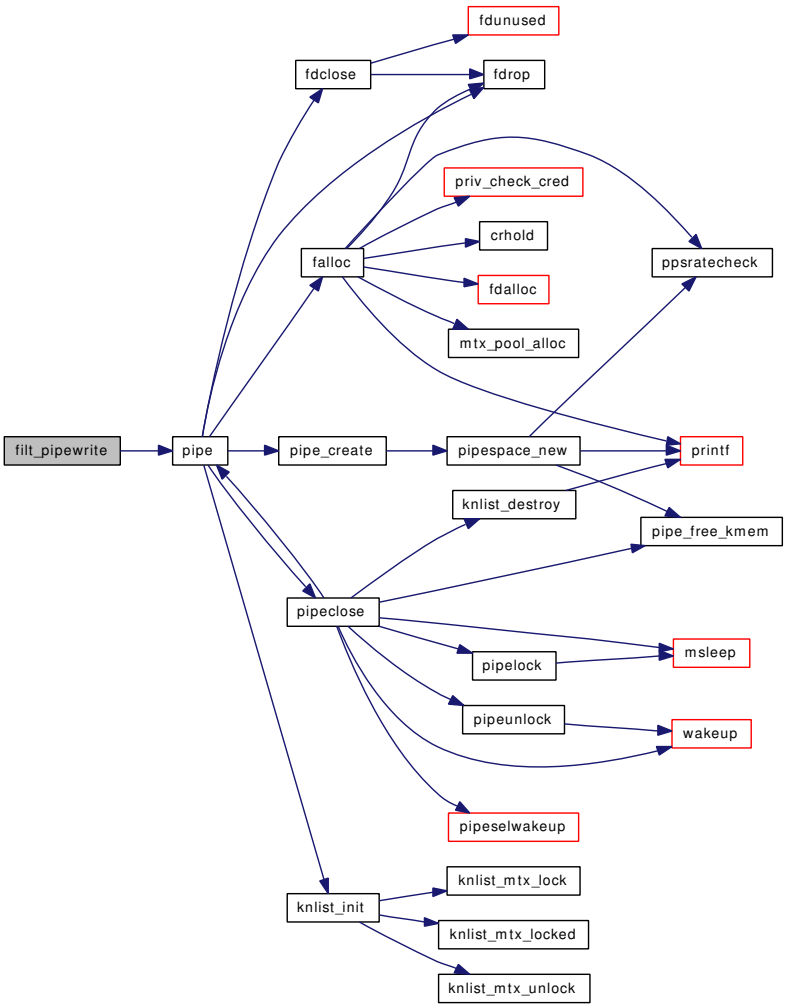


9.121.2.4 static int filt_pipewrite (struct knote * *kn*, long *hint*) [static]

Definition at line 1613 of file sys_pipe.c.

References pipe().

Here is the call graph for this function:



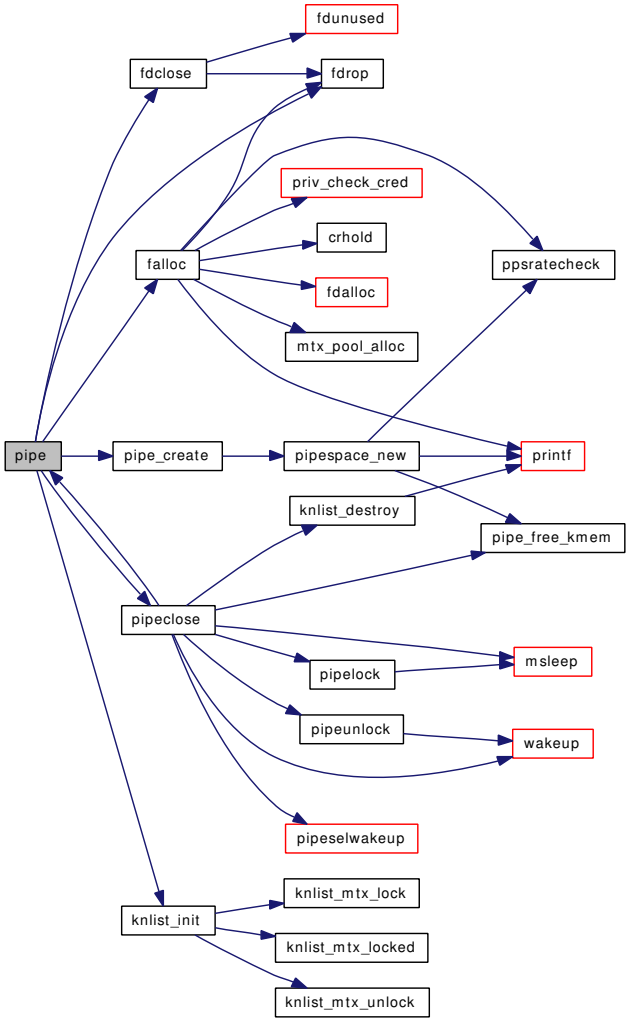
9.121.2.5 int pipe (struct thread * *td*, struct pipe_args * *uap*)

Definition at line 328 of file sys_pipe.c.

References falloc(), fdclose(), fdrop(), knlist_init(), pipe_create(), pipe_zone, pipeclose(), and pipeops.

Referenced by filt_pipedetach(), filt_piperead(), filt_pipewrite(), pipe_close(), pipe_ioctl(), pipe_kqfilter(), pipe_poll(), pipe_read(), pipe_stat(), pipe_write(), pipe_zone_ctor(), and pipeclose().

Here is the call graph for this function:



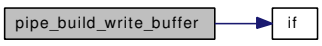
9.121.2.6 static int pipe_build_write_buffer (struct pipe * *wpipe*, struct uio * *uio*) [static]

Definition at line 756 of file sys_pipe.c.

References if().

Referenced by pipe_direct_write().

Here is the call graph for this function:



9.121.2.7 static void pipe_clone_write_buffer (struct pipe * *wpipe*) [static]

Definition at line 841 of file sys_pipe.c.

References pipe_destroy_write_buffer().

Referenced by pipe_direct_write().

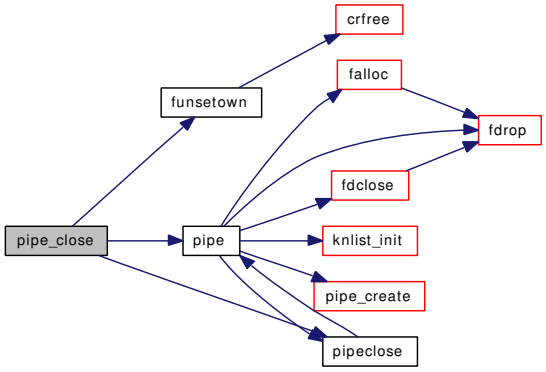
Here is the call graph for this function:

9.121.2.8 static int pipe_close (struct file * *fp*, struct thread * *td*) [static]

Definition at line 1430 of file sys_pipe.c.

References badfileops, funsetown(), pipe(), and pipeclose().

Here is the call graph for this function:



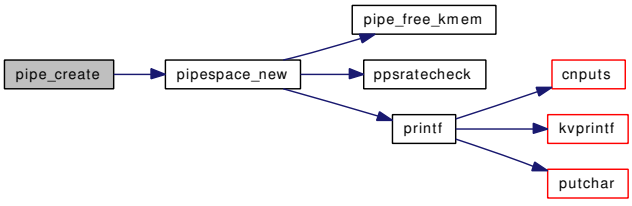
9.121.2.9 static int pipe_create (struct pipe * *pipe*, int *backing*) [static]

Definition at line 558 of file sys_pipe.c.

References amountpipekva, maxpipekva, and pipespace_new().

Referenced by pipe().

Here is the call graph for this function:



9.121.2.10 static void pipe_destroy_write_buffer (struct pipe * *wpipe*) [static]

Definition at line 821 of file sys_pipe.c.

Referenced by pipe_clone_write_buffer(), and pipe_direct_write().

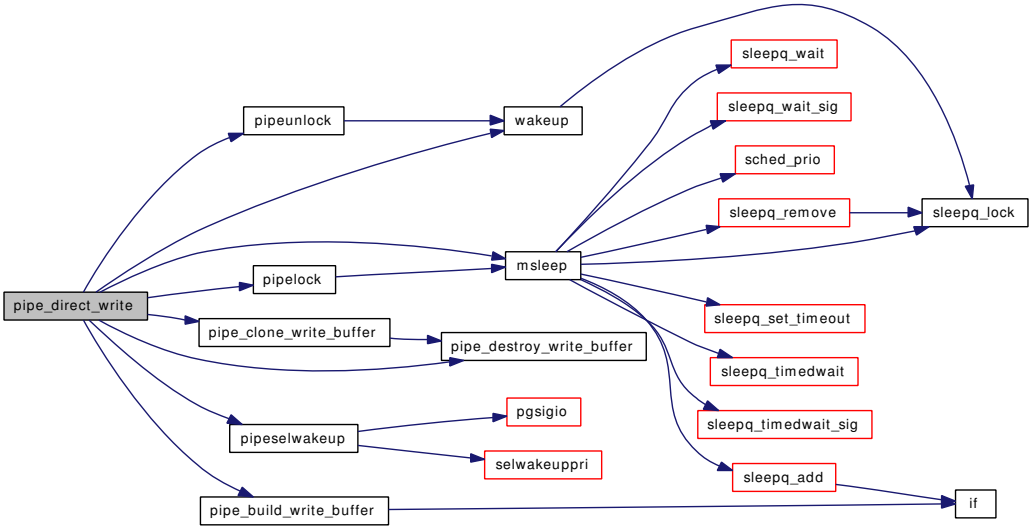
9.121.2.11 static int pipe_direct_write (struct pipe * *wpipe*, struct uio * *uio*) [static]

Definition at line 881 of file sys_pipe.c.

References msleep(), pipe_build_write_buffer(), pipe_clone_write_buffer(), pipe_destroy_write_buffer(), pipelock(), pipeselwakeup(), pipeunlock(), and wakeup().

Referenced by pipe_write().

Here is the call graph for this function:



9.121.2.12 static void pipe_free_kmem (struct pipe * *cpipe*) [static]

Definition at line 1444 of file sys_pipe.c.

References amountpipekva.

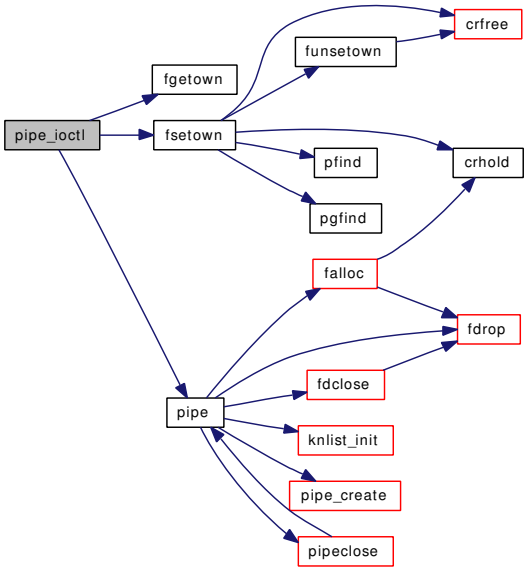
Referenced by pipeclose(), and pipespace_new().

9.121.2.13 static int pipe_ioctl (struct file * *fp*, u_long *cmd*, void * *data*, struct ucred * *active_cred*, struct thread * *td*) [static]

Definition at line 1260 of file sys_pipe.c.

References fgetown(), fsetown(), and pipe().

Here is the call graph for this function:

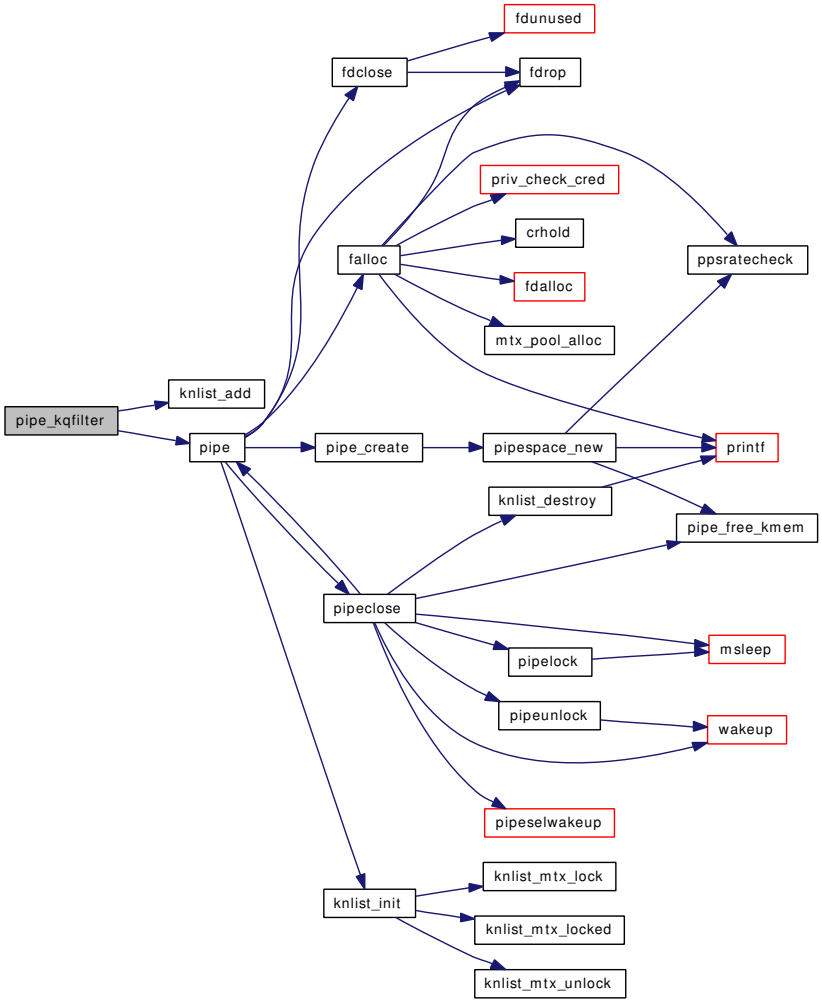


9.121.2.14 static int pipe_kqfilter (struct file * *fp*, struct knote * *kn*) [static]

Definition at line 1541 of file sys_pipe.c.

References knlist_add(), pipe(), pipe_rfiltops, and pipe_wfiltops.

Here is the call graph for this function:

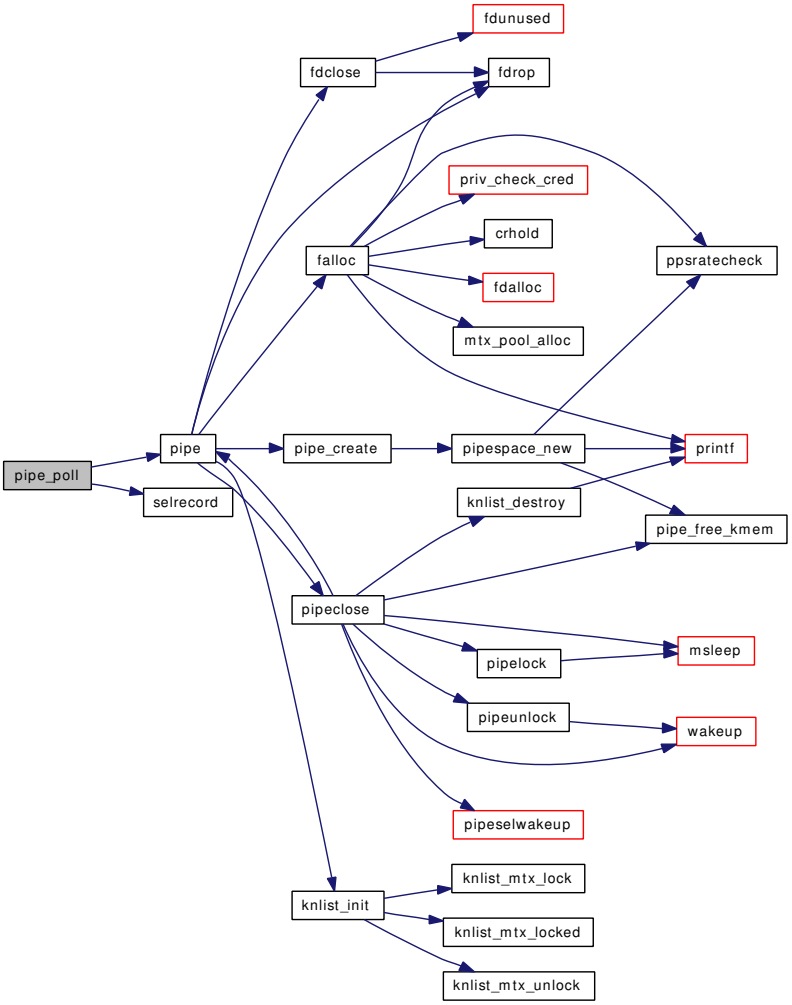


9.121.2.15 static int pipe_poll (struct file * *fp*, int *events*, struct ucred * *active_cred*, struct thread * *td*) [static]

Definition at line 1331 of file sys_pipe.c.

References pipe(), and selrecord().

Here is the call graph for this function:

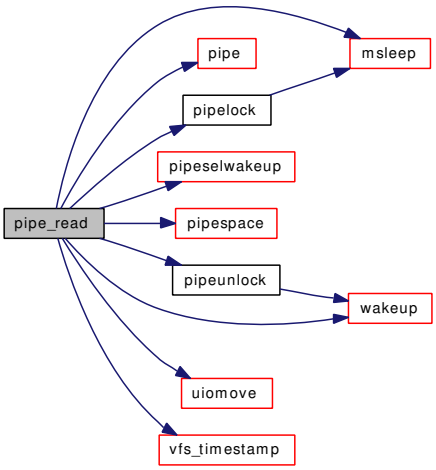


9.121.2.16 static int pipe_read (struct file * *fp*, struct uio * *uio*, struct ucred * *active_cred*, int *flags*, struct thread * *td*) [static]

Definition at line 578 of file sys_pipe.c.

References amountpipekva, maxpipekva, MINPIPESIZE, msleep(), pipe(), pipelock(), piperesizeallowed, pipeselwakeup(), pipespace(), pipeunlock(), uiomove(), vfs_timestamp(), and wakeup().

Here is the call graph for this function:

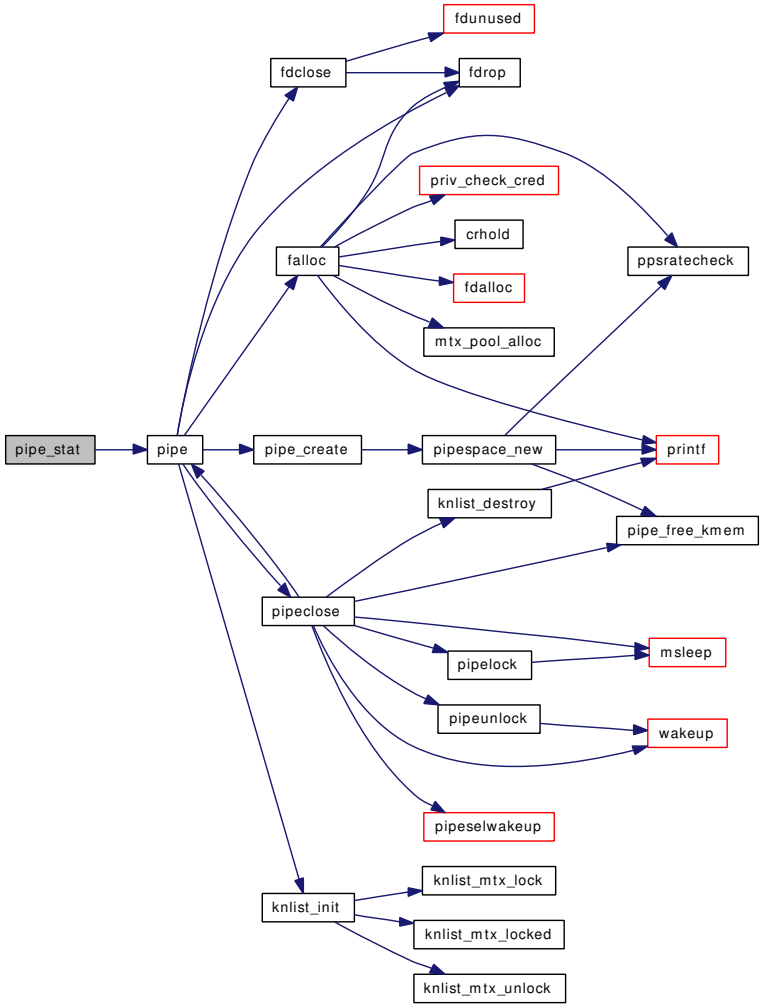


9.121.2.17 static int pipe_stat (struct file * *fp*, struct stat * *ub*, struct ucred * *active_cred*, struct thread * *td*) [static]

Definition at line 1392 of file sys_pipe.c.

References pipe().

Here is the call graph for this function:

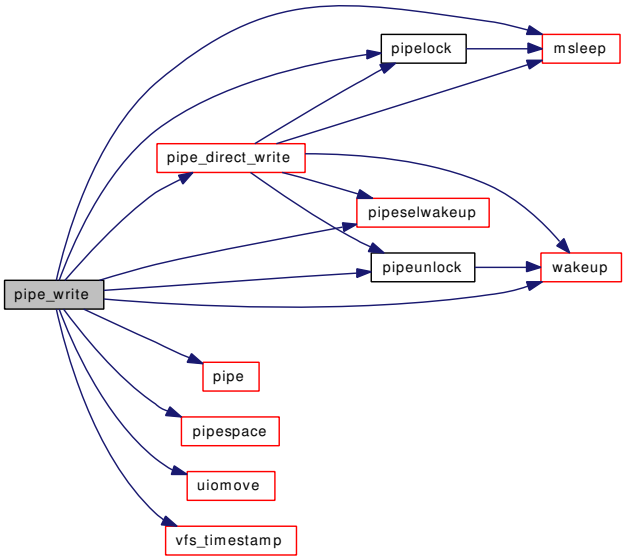


9.121.2.18 static int pipe_write (struct file * *fp*, struct uio * *uio*, struct ucred * *active_cred*, int *flags*, struct thread * *td*) [static]

Definition at line 978 of file sys_pipe.c.

References amountpipekva, maxpipekva, msleep(), pipe(), pipe_direct_write(), pipelock(), piperesizeallowed, pipeselwakeup(), pipespace(), pipeunlock(), uiomove(), vfs_timestamp(), and wakeup().

Here is the call graph for this function:



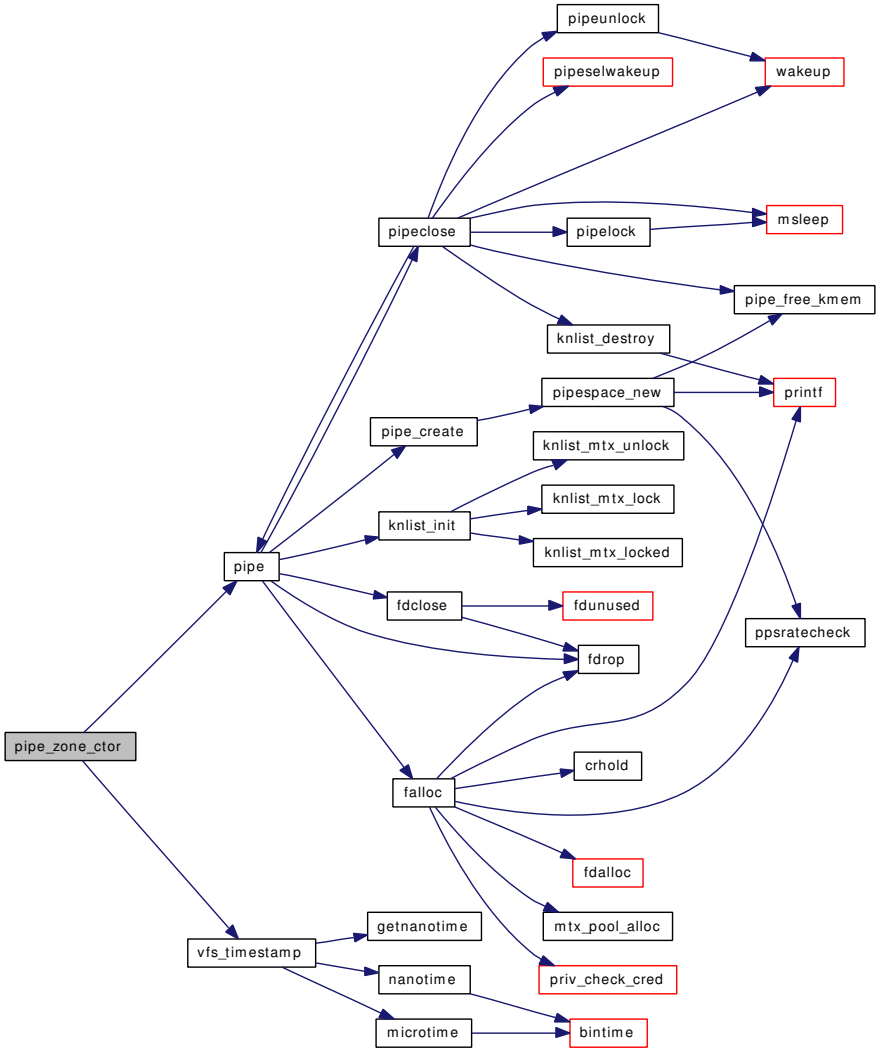
9.121.2.19 static int pipe_zone_ctor (void * *mem*, int *size*, void * *arg*, int *flags*) [static]

Definition at line 236 of file sys_pipe.c.

References amountpipes, pipe(), and vfs_timestamp().

Referenced by pipeinit().

Here is the call graph for this function:



9.121.2.20 static void pipe_zone_dtor (void * *mem*, int *size*, void * *arg*) [static]

Definition at line 285 of file sys_pipe.c.

References amountpipes.

Referenced by pipeinit().

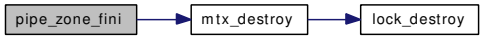
9.121.2.21 static void pipe_zone_fini (void * *mem*, int *size*) [static]

Definition at line 310 of file sys_pipe.c.

References mtx_destroy().

Referenced by pipeinit().

Here is the call graph for this function:



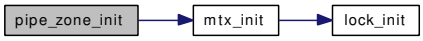
9.121.2.22 static int pipe_zone_init (void * *mem*, int *size*, int *flags*) [static]

Definition at line 297 of file sys_pipe.c.

References mtx_init().

Referenced by pipeinit().

Here is the call graph for this function:



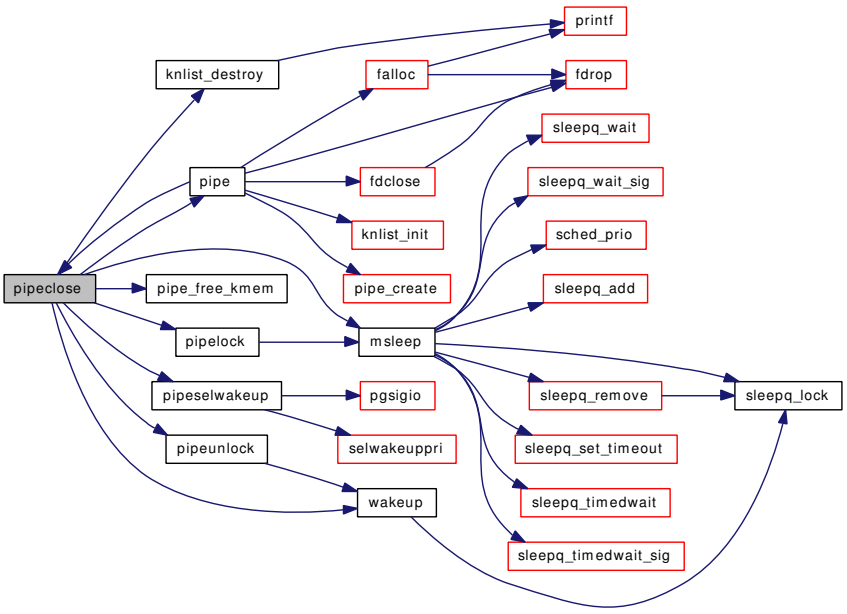
9.121.2.23 static void pipeclose (struct pipe * *cpipe*) [static]

Definition at line 1471 of file sys_pipe.c.

References knlist_destroy(), msleep(), pipe(), pipe_free_kmem(), pipe_zone, pipelock(), pipeselwakeup(), pipeunlock(), and wakeup().

Referenced by pipe(), and pipe_close().

Here is the call graph for this function:

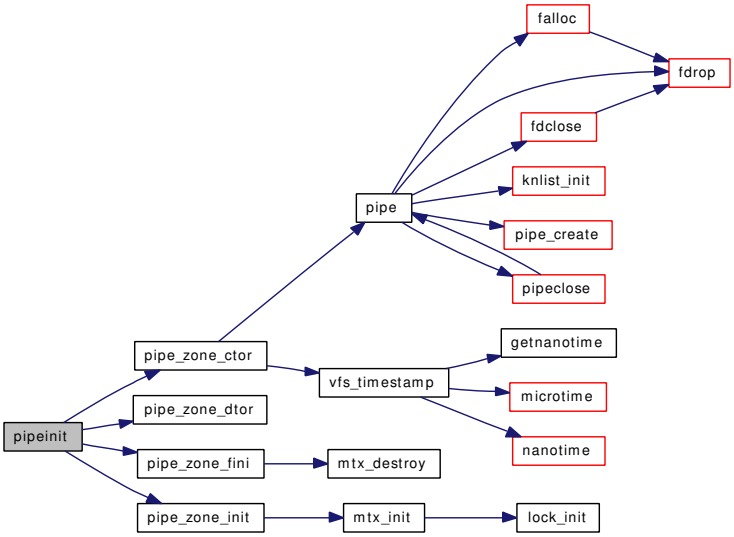


9.121.2.24 static void pipeinit (void *dummy *__unused*) [static]

Definition at line 226 of file sys_pipe.c.

References pipe_zone, pipe_zone_ctor(), pipe_zone_dtor(), pipe_zone_fini(), and pipe_zone_init().

Here is the call graph for this function:



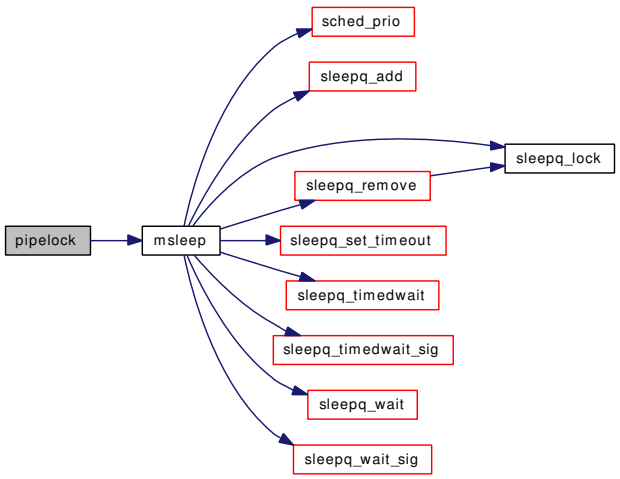
9.121.2.25 static __inline int pipelock (struct pipe * *cpipe*, int *catch*) [static]

Definition at line 501 of file sys_pipe.c.

References msleep().

Referenced by pipe_direct_write(), pipe_read(), pipe_write(), and pipeclose().

Here is the call graph for this function:



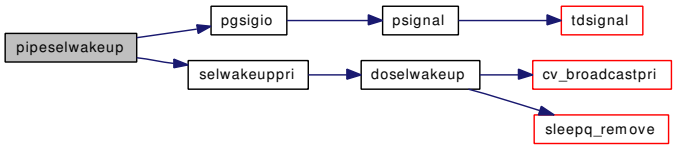
9.121.2.26 static __inline void pipeselwakeup (struct pipe * *cpipe*) [static]

Definition at line 539 of file sys_pipe.c.

References pgsigio(), and selwakeuppri().

Referenced by pipe_direct_write(), pipe_read(), pipe_write(), and pipeclose().

Here is the call graph for this function:



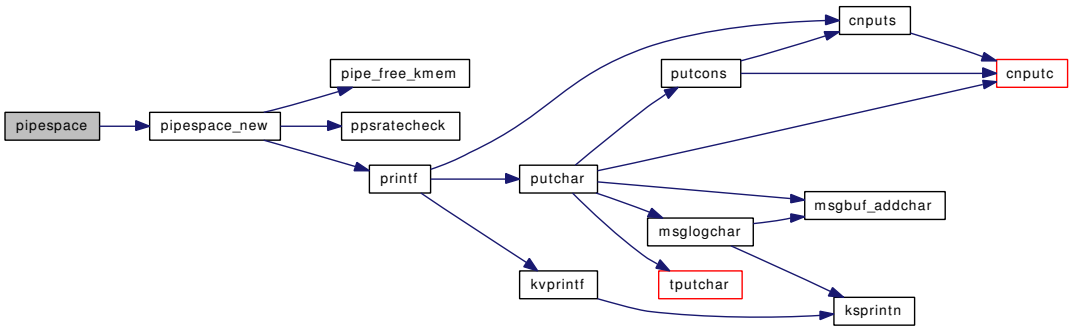
9.121.2.27 static int pipespace (struct pipe * *cpipe*, int *size*) [static]

Definition at line 487 of file sys_pipe.c.

References pipespace_new().

Referenced by pipe_read(), and pipe_write().

Here is the call graph for this function:



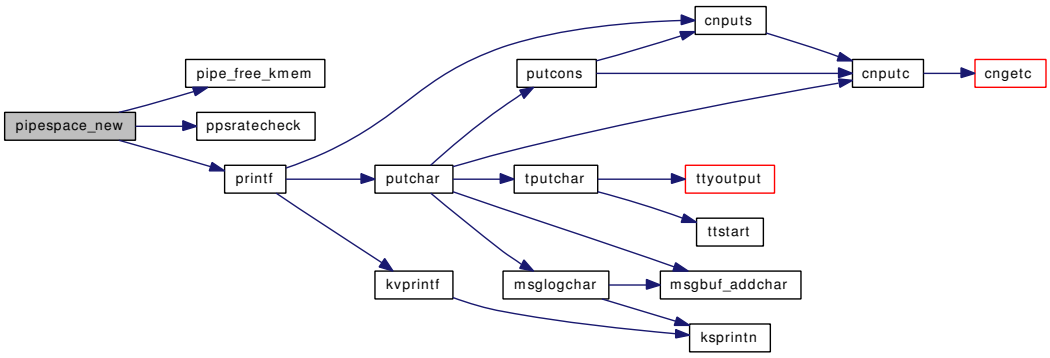
9.121.2.28 static int pipespace_new (struct pipe * *cpipe*, int *size*) [static]

Definition at line 419 of file sys_pipe.c.

References amountpipekva, pipe_free_kmem(), pipeallocfail, pipefragretry, piperesizefail, ppsratecheck(), and printf().

Referenced by pipe_create(), and pipespace().

Here is the call graph for this function:



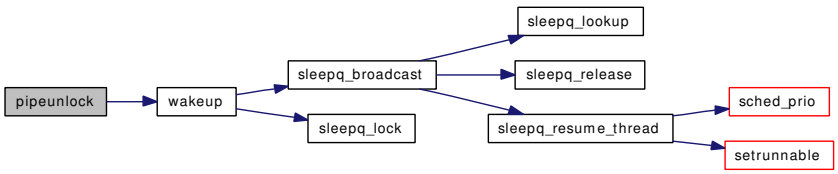
9.121.2.29 static __inline void pipeunlock (struct pipe * *cpipe*) [static]

Definition at line 524 of file sys_pipe.c.

References wakeup().

Referenced by pipe_direct_write(), pipe_read(), pipe_write(), and pipeclose().

Here is the call graph for this function:



9.121.2.30 SYSCTL_INT (_kern_ipc, OID_AUTO, piperesizeallowed, CTLFLAG_RW, & *piperesizeallowed*, 0, "Pipe resizing allowed")

9.121.2.31 SYSCTL_INT (_kern_ipc, OID_AUTO, piperesizefail, CTLFLAG_RD, & *piperesizefail*, 0, "Pipe resize failures")

9.121.2.32 SYSCTL_INT (_kern_ipc, OID_AUTO, pipeallocfail, CTLFLAG_RD, & *pipeallocfail*, 0, "Pipe allocation failures")

9.121.2.33 SYSCTL_INT (_kern_ipc, OID_AUTO, pipefragretry, CTLFLAG_RD, & *pipefragretry*, 0, "Pipe allocation retries due to fragmentation")

9.121.2.34 SYSCTL_INT (_kern_ipc, OID_AUTO, pipekva, CTLFLAG_RD, & *amountpipekva*, 0, "Pipe KVA usage")

9.121.2.35 SYSCTL_INT (_kern_ipc, OID_AUTO, pipes, CTLFLAG_RD, & *amountpipes*, 0, "Current # of pipes")

9.121.2.36 SYSCTL_INT (_kern_ipc, OID_AUTO, maxpipekva, CTLFLAG_RDTUN, & *maxpipekva*, 0, "Pipe KVA limit")

9.121.2.37 SYSINIT (vfs, SI_SUB_VFS, SI_ORDER_ANY, pipeinit, NULL)

9.121.3 Variable Documentation

9.121.3.1 int amountpipekva [static]

Definition at line 179 of file sys_pipe.c.

Referenced by pipe_create(), pipe_free_kmem(), pipe_read(), pipe_write(), and pipespace_new().

9.121.3.2 int amountpipes [static]

Definition at line 178 of file sys_pipe.c.

Referenced by pipe_zone_ctor(), and pipe_zone_dtor().

9.121.3.3 fo_close_t pipe_close [static]

Definition at line 147 of file sys_pipe.c.

9.121.3.4 fo_ioctl_t pipe_ioctl [static]

Definition at line 143 of file sys_pipe.c.

9.121.3.5 fo_kqfilter_t pipe_kqfilter [static]

Definition at line 145 of file sys_pipe.c.

9.121.3.6 fo_poll_t pipe_poll [static]

Definition at line 144 of file sys_pipe.c.

9.121.3.7 fo_rdwr_t pipe_read [static]

Definition at line 141 of file sys_pipe.c.

9.121.3.8 struct filterops pipe_rfiltops [static]

Initial value:

```
{ 1, NULL, filt_pipedetach, filt_piperead }
```

Definition at line 164 of file sys_pipe.c.

Referenced by pipe_kqfilter().

9.121.3.9 fo_stat_t pipe_stat [static]

Definition at line 146 of file sys_pipe.c.

9.121.3.10 struct filterops pipe_wfiltops [static]

Initial value:

```
{ 1, NULL, filt_pipedetach, filt_pipewrite }
```

Definition at line 166 of file sys_pipe.c.

Referenced by pipe_kqfilter().

9.121.3.11 fo_rdwr_t pipe_write [static]

Definition at line 142 of file sys_pipe.c.

9.121.3.12 uma_zone_t pipe_zone [static]

Definition at line 221 of file sys_pipe.c.

Referenced by pipe(), pipeclose(), and pipeinit().

9.121.3.13 int pipeallocfail [static]

Definition at line 181 of file sys_pipe.c.

Referenced by pipespace_new().

9.121.3.14 int pipefragretry [static]

Definition at line 180 of file sys_pipe.c.

Referenced by pipespace_new().

9.121.3.15 struct fileops pipeops [static]

Initial value:

```
{
        .fo_read = pipe_read,
        .fo_write = pipe_write,
        .fo_ioctl = pipe_ioctl,
        .fo_poll = pipe_poll,
        .fo_kqfilter = pipe_kqfilter,
        .fo_stat = pipe_stat,
        .fo_close = pipe_close,
        .fo_flags = DFLAG_PASSABLE
}
```

Definition at line 149 of file sys_pipe.c.

Referenced by pipe().

9.121.3.16 int piperesizeallowed = 1 [static]

Definition at line 183 of file sys_pipe.c.

Referenced by pipe_read(), and pipe_write().

9.121.3.17 int piperesizefail [static]

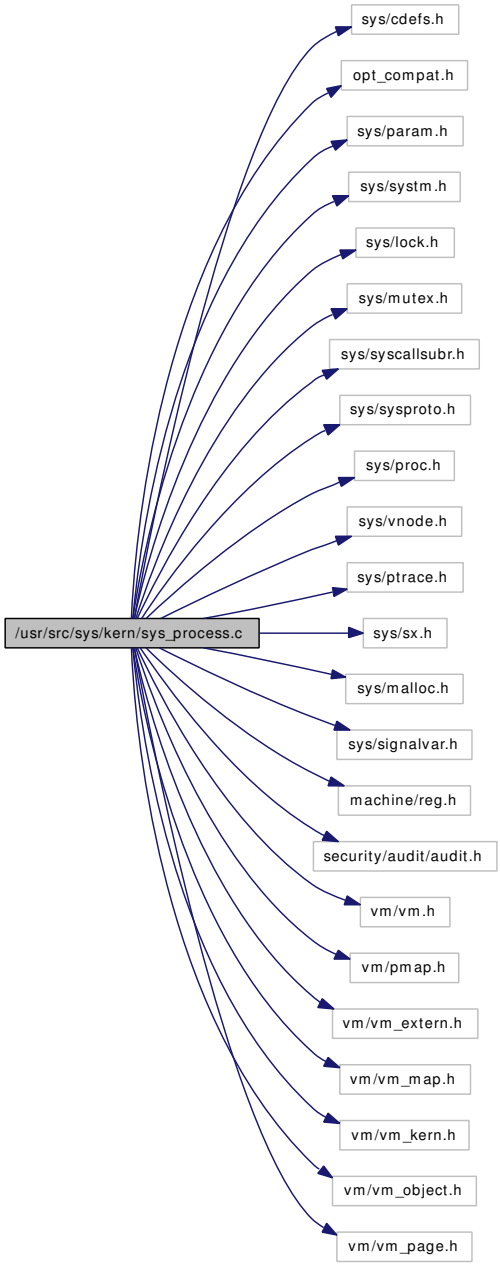
Definition at line 182 of file sys_pipe.c.

Referenced by pipespace_new().

9.122 /usr/src/sys/kern/sys_process.c File Reference

```
#include <sys/cdefs.h>
#include "opt_compat.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/syscallsubr.h>
#include <sys/sysproto.h>
#include <sys/proc.h>
#include <sys/vnode.h>
#include <sys/ptrace.h>
#include <sys/sx.h>
#include <sys/malloc.h>
#include <sys/signalvar.h>
#include <machine/reg.h>
#include <security/audit/audit.h>
#include <vm/vm.h>
#include <vm/pmap.h>
#include <vm/vm_extern.h>
#include <vm/vm_map.h>
#include <vm/vm_kern.h>
#include <vm/vm_object.h>
#include <vm/vm_page.h>
```

Include dependency graph for sys_process.c:



Data Structures

- struct ptrace_args

Defines

- #define PROC_ACTION(action)
- #define COPYIN(u, k, s) copyin(u, k, s)
- #define COPYOUT(k, u, s) copyout(k, u, s)
- #define PROC_READ(w, t, a) proc_read_ ## w (t, a)

- #define PROC_WRITE(w, t, a) proc_write_ ## w (t, a)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/sys_process.c,v 1.141 2006/10/26 21:42:20 jb Exp $")
- int proc_read_regs (struct thread *td, struct reg *regs)
- int proc_write_regs (struct thread *td, struct reg *regs)
- int proc_read_dbregs (struct thread *td, struct dbreg *dbregs)
- int proc_write_dbregs (struct thread *td, struct dbreg *dbregs)
- int proc_read_fpregs (struct thread *td, struct fpreg *fpregs)
- int proc_write_fpregs (struct thread *td, struct fpreg *fpregs)
- int proc_sstep (struct thread *td)
- int proc_rwmem (struct proc *p, struct uio *uio)
- int ptrace (struct thread *td, struct ptrace_args *uap)
- int kern_ptrace (struct thread *td, int req, pid_t pid, void *addr, int data)
- void stopevent (struct proc *p, unsigned int event, unsigned int val)

9.122.1 Define Documentation

9.122.1.1 #define COPYIN(u, k, s) copyin(u, k, s)

Definition at line 368 of file sys_process.c.

Referenced by ptrace().

9.122.1.2 #define COPYOUT(k, u, s) copyout(k, u, s)

Definition at line 369 of file sys_process.c.

Referenced by ptrace().

9.122.1.3 #define PROC_ACTION(action)

Value:

```
do {                                    \
        int error;                                                      \
                                                                        \
        PROC_LOCK_ASSERT(td->td_proc, MA_OWNED);                        \
        if ((td->td_proc->p_sflag & PS_INMEM) == 0)                     \
                error = EIO;                                            \
        else                                                            \
                error = (action);                                       \
        return (error);                                                 \
} while(0)
```

Definition at line 103 of file sys_process.c.

Referenced by proc_read_dbregs(), proc_read_fpregs(), proc_read_regs(), proc_sstep(), proc_write_dbregs(), proc_write_fpregs(), and proc_write_regs().

9.122.1.4 #define PROC_READ(w, t, a) proc_read_ ## w (t, a)

Definition at line 477 of file sys_process.c.

Referenced by kern_ptrace().

9.122.1.5 #define PROC_WRITE(w, t, a) proc_write_ ## w (t, a)

Definition at line 478 of file sys_process.c.

Referenced by kern_ptrace().

9.122.2 Function Documentation

9.122.2.1 __FBSDID ("$FreeBSD: src/sys/kern/sys_process. *c*, v 1.141 2006/10/26 21:42:20 jb Exp $")

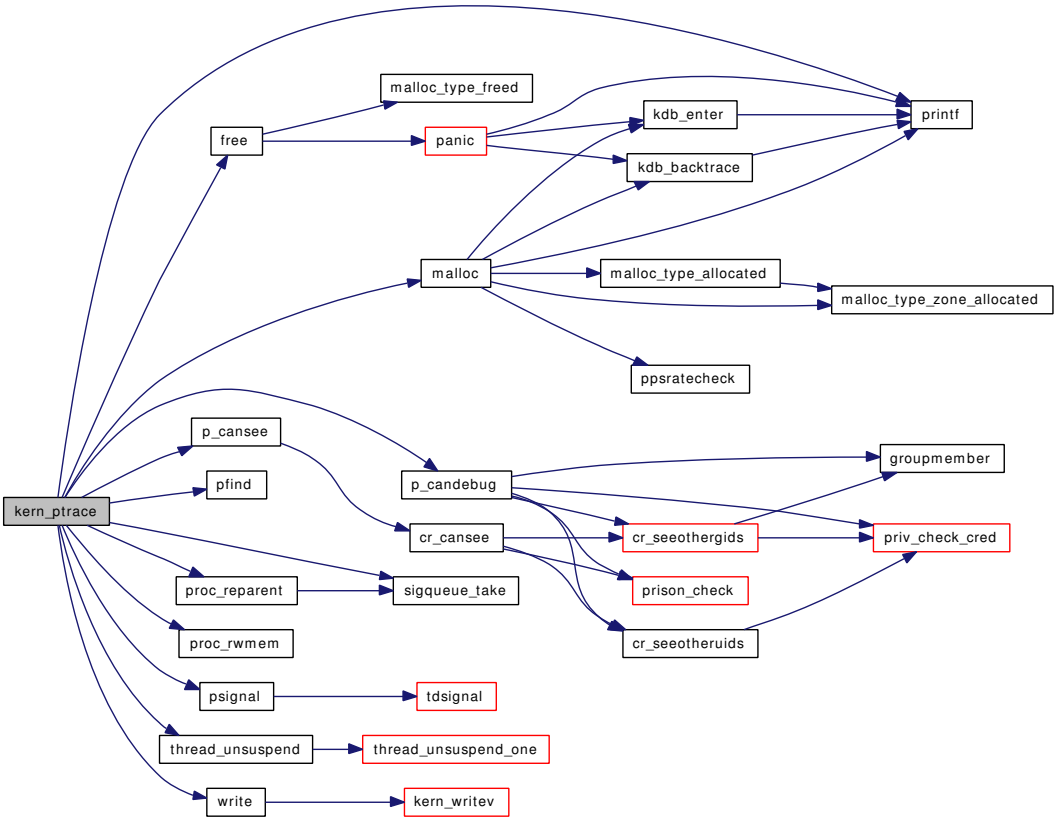
9.122.2.2 int kern_ptrace (struct thread * *td*, int *req*, pid_t *pid*, void * *addr*, int *data*)

Definition at line 482 of file sys_process.c.

References allproc_lock, buf, free(), initproc, malloc(), p_candebug(), p_cansee(), pfind(), printf(), PROC_READ, proc_reparent(), proc_rwmem(), PROC_WRITE, proctree_lock, psignal(), sched_lock, sigqueue_take(), thread_unsuspend(), and write().

Referenced by ptrace().

Here is the call graph for this function:



9.122.2.3 int proc_read_dbregs (struct thread * *td*, struct dbreg * *dbregs*)

Definition at line 129 of file sys_process.c.

References PROC_ACTION.

9.122.2.4 int proc_read_fpregs (struct thread * *td*, struct fpreg * *fpregs*)

Definition at line 147 of file sys_process.c.

References PROC_ACTION.

9.122.2.5 int proc_read_regs (struct thread * *td*, struct reg * *regs*)

Definition at line 115 of file sys_process.c.

References PROC_ACTION.

9.122.2.6 int proc_rwmem (struct proc * *p*, struct uio * *uio*)

Definition at line 213 of file sys_process.c.

Referenced by kern_ptrace().

9.122.2.7 int proc_sstep (struct thread * *td*)

Definition at line 206 of file sys_process.c.

References PROC_ACTION.

9.122.2.8 int proc_write_dbregs (struct thread * *td*, struct dbreg * *dbregs*)

Definition at line 136 of file sys_process.c.

References PROC_ACTION.

9.122.2.9 int proc_write_fpregs (struct thread * *td*, struct fpreg * *fpregs*)

Definition at line 154 of file sys_process.c.

References PROC_ACTION.

9.122.2.10 int proc_write_regs (struct thread * *td*, struct reg * *regs*)

Definition at line 122 of file sys_process.c.

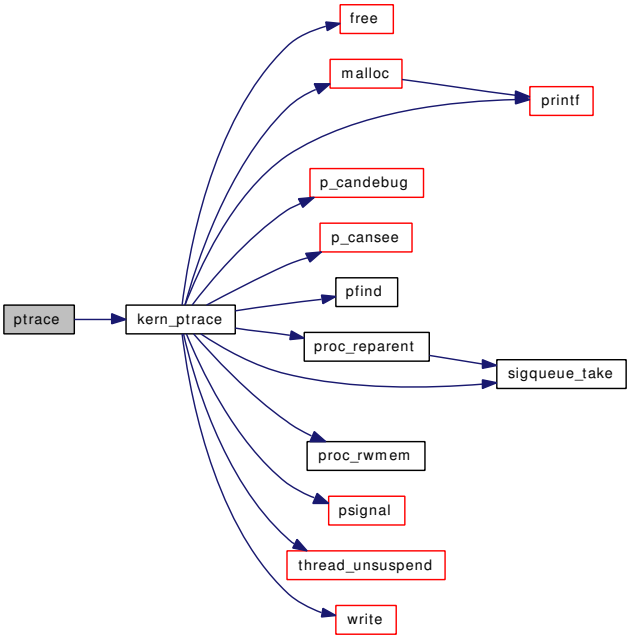
References PROC_ACTION.

9.122.2.11 int ptrace (struct thread * *td*, struct ptrace_args * *uap*)

Definition at line 375 of file sys_process.c.

References ptrace_args::addr, COPYIN, COPYOUT, ptrace_args::data, kern_ptrace(), ptrace_args::pid, and ptrace_args::req.

Here is the call graph for this function:



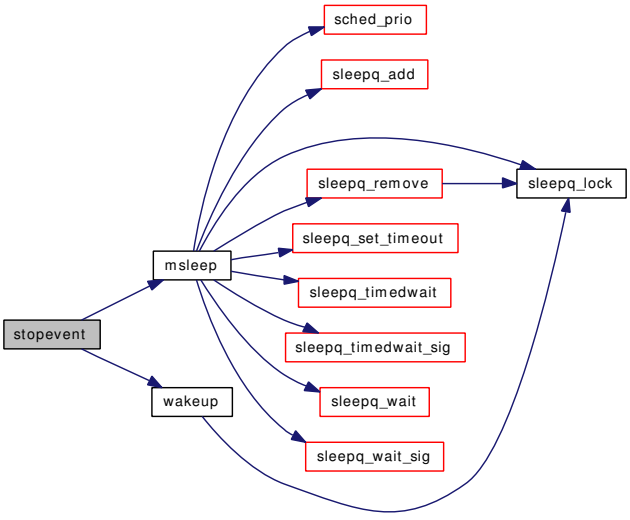
9.122.2.12 void stopevent (struct proc * *p*, unsigned int *event*, unsigned int *val*)

Definition at line 1020 of file sys_process.c.

References msleep(), and wakeup().

Referenced by issignal(), and postsig().

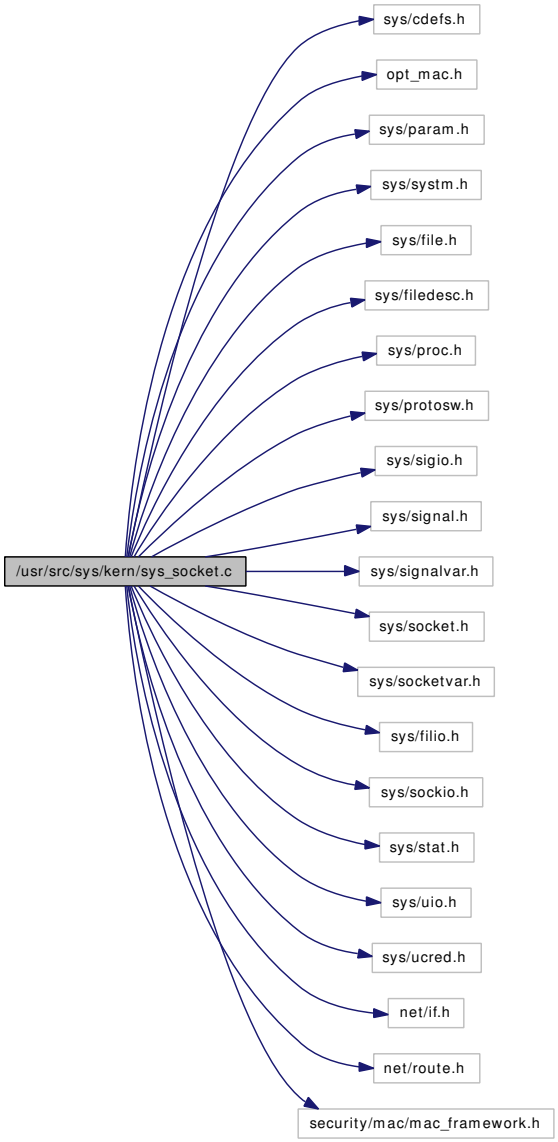
Here is the call graph for this function:



9.123 /usr/src/sys/kern/sys_socket.c File Reference

```
#include <sys/cdefs.h>
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/file.h>
#include <sys/filedesc.h>
#include <sys/proc.h>
#include <sys/protosw.h>
#include <sys/sigio.h>
#include <sys/signal.h>
#include <sys/signalvar.h>
#include <sys/socket.h>
#include <sys/socketvar.h>
#include <sys/filio.h>
#include <sys/sockio.h>
#include <sys/stat.h>
#include <sys/uio.h>
#include <sys/ucred.h>
#include <net/if.h>
#include <net/route.h>
#include <security/mac/mac_framework.h>
```

Include dependency graph for sys_socket.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/sys_socket.c,v 1.71 2006/10/22 11:52:13 rwatson Exp $")
- int soo_read (struct file *fp, struct uio *uio, struct ucred *active_cred, int flags, struct thread *td)
- int soo_write (struct file *fp, struct uio *uio, struct ucred *active_cred, int flags, struct thread *td)
- int soo_ioctl (struct file *fp, u_long cmd, void *data, struct ucred *active_cred, struct thread *td)
- int soo_poll (struct file *fp, int events, struct ucred *active_cred, struct thread *td)
- int soo_stat (struct file *fp, struct stat *ub, struct ucred *active_cred, struct thread *td)
- int soo_close (struct file *fp, struct thread *td)

Variables

- fileops socketops

9.123.1 Function Documentation

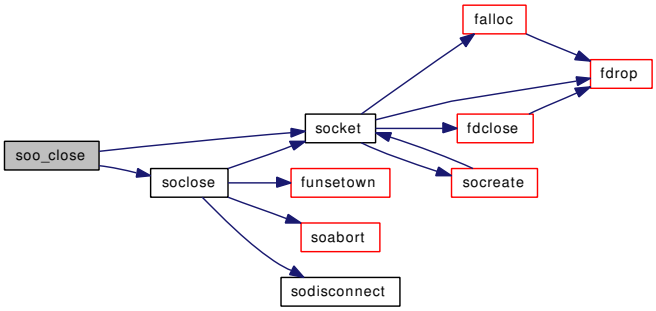
9.123.1.1 __FBSDID ("$FreeBSD: src/sys/kern/sys_socket. *c*, v 1.71 2006/10/22 11:52:13 rwatson Exp $")

9.123.1.2 int soo_close (struct file * *fp*, struct thread * *td*)

Definition at line 303 of file sys_socket.c.

References badfileops, socket(), and soclose().

Here is the call graph for this function:

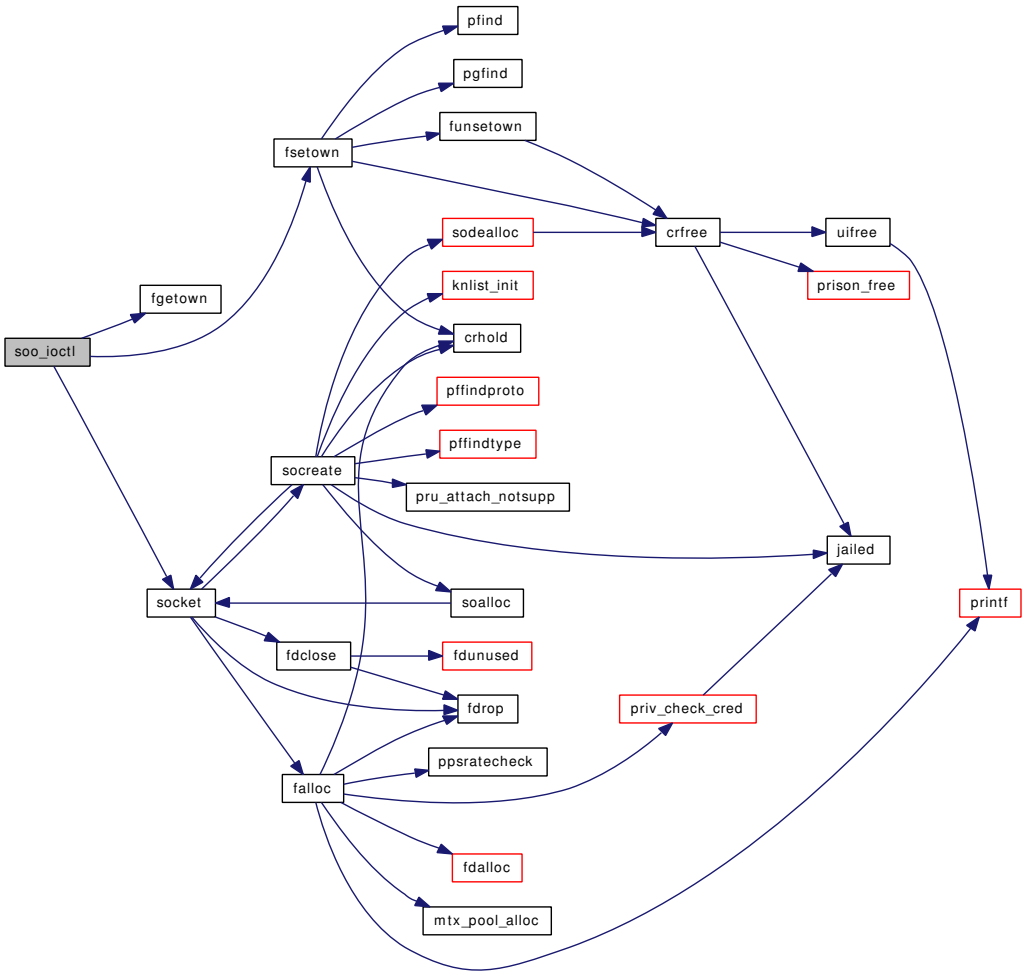


9.123.1.3 int soo_ioctl (struct file * *fp*, u_long *cmd*, void * *data*, struct ucred * *active_cred*, struct thread * *td*)

Definition at line 130 of file sys_socket.c.

References fgetown(), fsetown(), and socket().

Here is the call graph for this function:

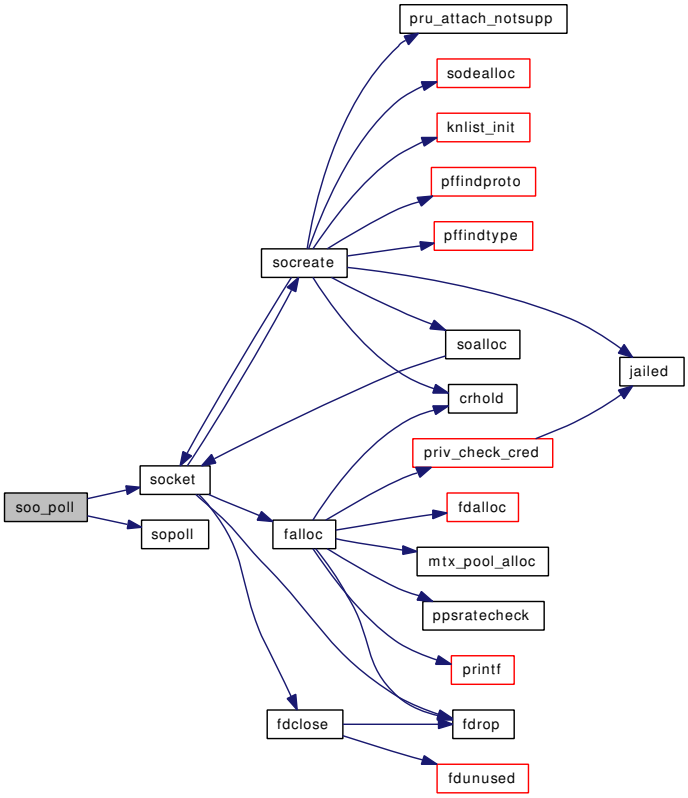


9.123.1.4 int soo_poll (struct file * *fp*, int *events*, struct ucred * *active_cred*, struct thread * *td*)

Definition at line 227 of file sys_socket.c.

References socket(), and sopoll().

Here is the call graph for this function:

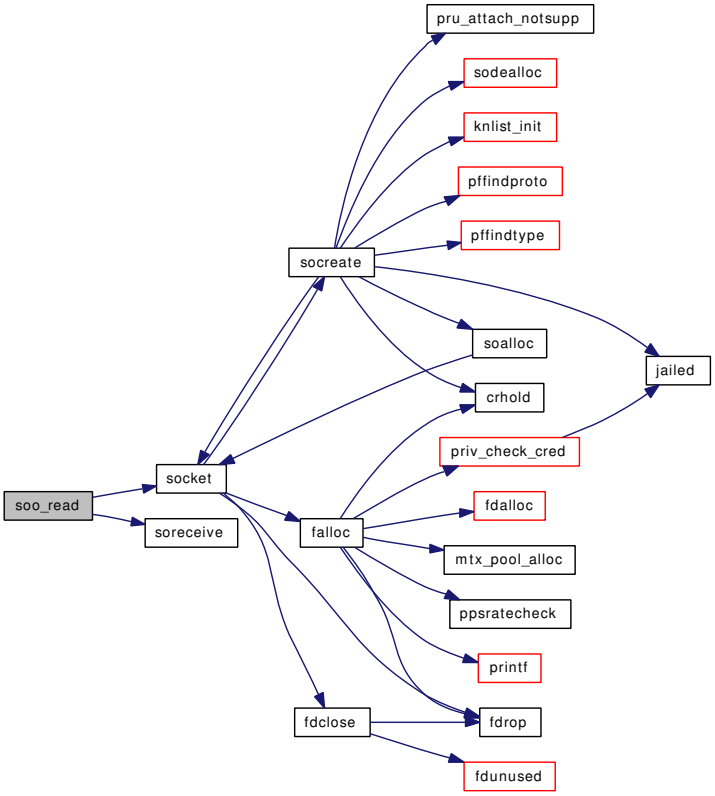


9.123.1.5 int soo_read (struct file * *fp*, struct uio * *uio*, struct ucred * *active_cred*, int *flags*, struct thread * *td*)

Definition at line 72 of file sys_socket.c.

References socket(), and soreceive().

Here is the call graph for this function:

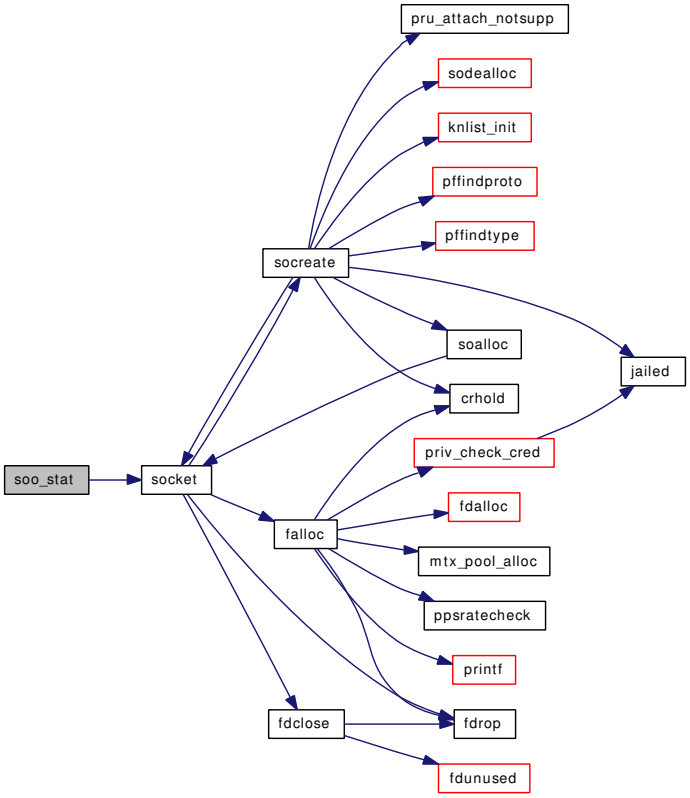


9.123.1.6 int soo_stat (struct file * *fp*, struct stat * *ub*, struct ucred * *active_cred*, struct thread * *td*)

Definition at line 253 of file sys_socket.c.

References socket().

Here is the call graph for this function:

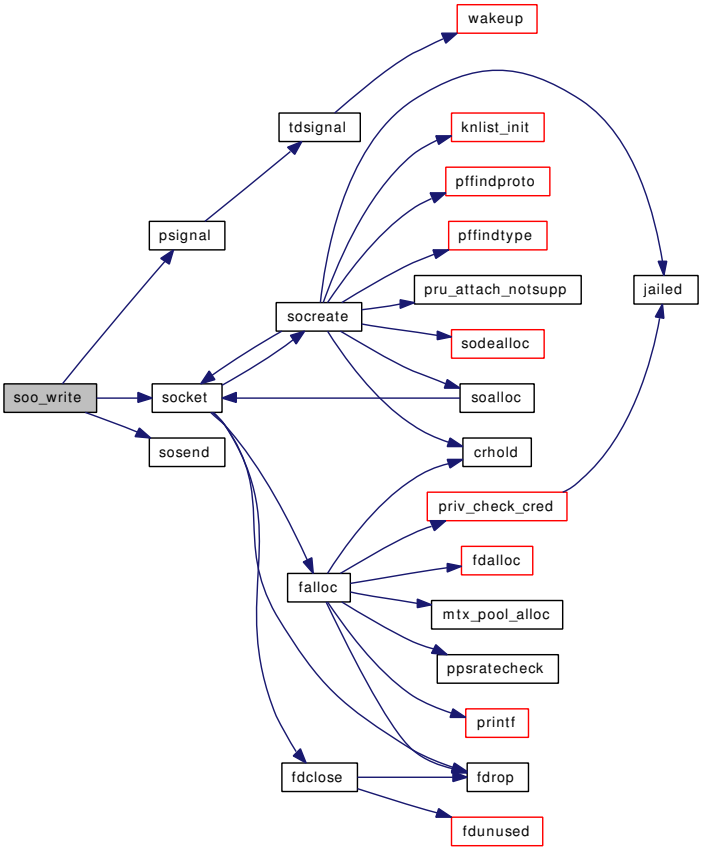


9.123.1.7 int soo_write (struct file * *fp*, struct uio * *uio*, struct ucred * *active_cred*, int *flags*, struct thread * *td*)

Definition at line 99 of file sys_socket.c.

References psignal(), socket(), and sosend().

Here is the call graph for this function:



9.123.2 Variable Documentation

9.123.2.1 struct fileops socketops

Initial value:

```
{
        .fo_read = soo_read,
        .fo_write = soo_write,
        .fo_ioctl = soo_ioctl,
        .fo_poll = soo_poll,
        .fo_kqfilter = soo_kqfilter,
        .fo_stat = soo_stat,
        .fo_close = soo_close,
        .fo_flags = DFLAG_PASSABLE
}
```

Definition at line 59 of file sys_socket.c.

Referenced by kern_accept(), sctp_peeloff(), socket(), and socketpair().

9.124 /usr/src/sys/kern/syscalls.c File Reference

Variables

- const char * syscallnames [ ]

9.124.1 Variable Documentation

9.124.1.1 const char* syscallnames[ ]

Definition at line 9 of file syscalls.c.

9.125 /usr/src/sys/kern/systrace_args.c File Reference

Functions

- static void systrace_args (int sysnum, void *params, u_int64_t *uarg, int *n_args)

9.125.1 Function Documentation

9.125.1.1 static void systrace_args (int *sysnum*, void * *params*, u_int64_t * *uarg*, int * *n_args*) [static]

Definition at line 10 of file systrace_args.c.

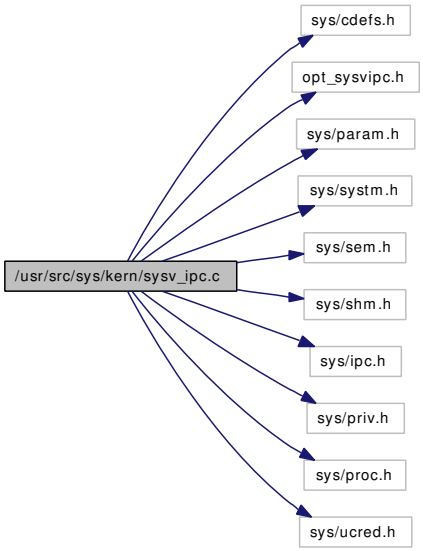
References ksem_timedwait_args::abstime, sigaction_args::act, ptrace_args::addr, __semctl_args::arg, quotactl_args::arg, fcntl_args::arg, abort2_args::args, __mac_execve_args::argv, execve_args::argv, getdirentries_args::basep, fhstatfs_args::buf, fstatfs_args::buf, statfs_args::buf, getfsstat_args::buf, __getcwd_args::buf, getdents_args::buf, shmctl_args::buf, msgctl_args::buf, getdirentries_args::buf, pwrite_args::buf, pread_args::buf, readlink_args::buf, write_args::buf, read_args::buf, __getcwd_args::buflen, getfsstat_args::bufsize, kevent_args::changelist, ktimer_create_args::clock_id, clock_getres_args::clock_id, clock_settime_args::clock_id, clock_gettime_args::clock_id, shmctl_args::cmd, msgctl_args::cmd, __semctl_args::cmd, quotactl_args::cmd, fcntl_args::cmd, ioctl_args::com, uuidgen_args::count, getdents_args::count, getdirentries_args::count, readlink_args::count, ioctl_args::data, ptrace_args::data, mount_args::data, adjtime_args::delta, mknod_args::dev, setdomainname_args::domainname, getdomainname_args::domainname, getresgid_args::egid, setresgid_args::egid, setegid_args::egid, setregid_args::egid, __mac_execve_args::envv, execve_args::envv, getresuid_args::euid, setresuid_args::euid, seteuid_args::euid, setreuid_args::euid, kevent_args::eventlist, ktimer_create_args::evp, select_args::ex, ktrace_args::facs, fstatfs_args::fd, kevent_args::fd, pwritev_args::fd, preadv_args::fd, nfstat_args::fd, getdents_args::fd, futimes_args::fd, ftruncate_args::fd, lseek_args::fd, getdirentries_args::fd, fpathconf_args::fd, fstat_args::fd, pwrite_args::fd, pread_args::fd, flock_args::fd, fchmod_args::fd, fchown_args::fd, writev_args::fd, readv_args::fd, fsync_args::fd, fcntl_args::fd, ioctl_args::fd, dup_args::fd, fchflags_args::fd, fchdir_args::fd, close_args::fd, write_args::fd, read_args::fd, openbsd_poll_args::fds, poll_args::fds, getfh_args::fhp, lgetfh_args::fhp, kse_switchin_args::flags, getfsstat_args::flags, eaccess_args::flags, fhopen_args::flags, ktimer_settime_args::flags, fchflags_args::flags, chflags_args::flags, access_args::flags, unmount_args::flags, mount_args::flags, open_args::flags, __mac_execve_args::fname, getfh_args::fname, lgetfh_args::fname, execve_args::fname, ktrace_args::fname, rename_args::from, dup2_args::from, rtprio_thread_args::function, rtprio_args::function, lchown_args::gid, setgid_args::gid, fchown_args::gid, chown_args::gid, setgroups_args::gidset, getgroups_args::gidset, setgroups_args::gidsetsize, getgroups_args::gidsetsize, sigprocmask_args::how, flock_args::how, ksem_timedwait_args::id, ksem_destroy_args::id, ksem_getvalue_args::id, ksem_trywait_args::id, ksem_wait_args::id, ksem_post_args::id, ksem_close_args::id, ksem_open_args::idp, ksem_init_args::idp, select_args::in, pwritev_args::iovcnt, preadv_args::iovcnt, writev_args::iovcnt, readv_args::iovcnt, pwritev_args::iovp, preadv_args::iovp, writev_args::iovp, readv_args::iovp, getitimer_args::itv, setitimer_args::itv, shmget_args::key, msgget_args::key, semget_args::key, setdomainname_args::len, getdomainname_args::len, ftruncate_args::length, truncate_args::length, symlink_args::link, link_args::link, rtprio_thread_args::lwpid, __mac_execve_args::mac_p, ksem_open_args::mode, lchmod_args::mode, mkdir_args::mode, mkfifo_args::mode, fchmod_args::mode, chmod_args::mode, mknod_args::mode, open_args::mode, msgrcv_args::msgflg, msgsnd_args::msgflg, msgget_args::msgflg, msgrcv_args::msgp, msgsnd_args::msgp, msgrcv_args::msgsz, msgsnd_args::msgsz, msgrcv_args::msgtyp, msgrcv_args::msqid, msgsnd_args::msqid, msgctl_args::msqid, ksem_unlink_args::name, ksem_open_args::name, sysctl_args::name, fpathconf_args::name, pathconf_args::name, uname_args::name, setlogin_args::namebuf, getlogin_args::namebuf, sysctl_args::namelen, getlogin_args::namelen, abort2_args::nargs, pwrite_args::nbyte, pread_args::nbyte, write_args::nbyte, read_args::nbyte, kevent_args::nchanges, select_args::nd, kevent_args::nevents, sysctl_args::new, sysctl_args::newlen, umask_args::newmask, openbsd_poll_args::nfds, poll_args::nfds,

semget_args::nsems, semop_args::nsops, ntp_gettime_args::ntvp, sigaction_args::oact, pwritev_-args::offset, preadv_args::offset, lseek_args::offset, pwrite_args::offset, pread_args::offset, profil_-args::offset, ksem_open_args::oflag, setitimer_args::oitv, sysctl_args::old, adjtime_args::olddelta, sysctl_args::oldlenp, ktrace_args::ops, sigprocmask_args::oset, sigaltstack_args::oss, select_args::ou, swapcontext_args::oucp, ktimer_settime_args::ovalue, ftruncate_args::pad, truncate_args::pad, lseek_-args::pad, pwrite_args::pad, pread_args::pad, statfs_args::path, eaccess_args::path, nstat_args::path, lutimes_args::path, lchmod_args::path, lchown_args::path, truncate_args::path, pathconf_args::path, lstat_args::path, stat_args::path, quotactl_args::path, utimes_args::path, rmdir_args::path, mkdir_-args::path, mkfifo_args::path, chroot_args::path, readlink_args::path, symlink_args::path, revoke_-args::path, chflags_args::path, access_args::path, unmount_args::path, mount_args::path, chown_-args::path, chmod_args::path, mknod_args::path, chdir_args::path, unlink_args::path, link_args::path, open_args::path, setpgid_args::pgid, sigqueue_args::pid, getsid_args::pid, getpgid_args::pid, rtprio_-args::pid, setpgid_args::pid, ktrace_args::pid, kill_args::pid, ptrace_args::pid, setpriority_args::prio, ptrace_args::req, getresgid_args::rgid, setresgid_args::rgid, setregid_args::rgid, __setrlimit_args::rlp, __getrlimit_args::rlp, nanosleep_args::rmtp, nanosleep_args::rqtp, rtprio_thread_args::rtp, rtprio_-args::rtp, getresuid_args::ruid, setresuid_args::ruid, setreuid_args::ruid, getrusage_args::rusage, profil_-args::samples, fhstat_args::sb, nfstat_args::sb, fstat_args::sb, profil_args::scale, semget_args::semflg, semop_args::semid, __semctl_args::semid, __semctl_args::semnum, sigpending_args::set, sigprocmask_-args::set, getresgid_args::sgid, setresgid_args::sgid, shmdt_args::shmaddr, shmat_args::shmaddr, shmget_args::shmflg, shmat_args::shmflg, shmctl_args::shmid, shmat_args::shmid, sigaction_args::sig, sigsuspend_args::sigmask, sigqueue_args::signum, kill_args::signum, shmget_args::size, profil_args::size, semop_args::sops, sigaltstack_args::ss, uuidgen_args::store, getresuid_args::suid, setresuid_args::suid, kevent_args::timeout, openbsd_poll_args::timeout, poll_args::timeout, ktimer_gettime_args::timerid, ktimer_settime_args::timerid, ktimer_delete_args::timerid, ktimer_create_args::timerid, kse_switchin_-args::tmbx, rename_args::to, dup2_args::to, clock_getres_args::tp, clock_settime_args::tp, clock_-gettime_args::tp, ntp_adjtime_args::tp, gettimeofday_args::tp, lutimes_args::tptr, futimes_args::tptr, utimes_args::tptr, settimeofday_args::tv, select_args::tv, mount_args::type, settimeofday_args::tzp, gettimeofday_args::tzp, fhstatfs_args::u_fhp, fhstat_args::u_fhp, fhopen_args::u_fhp, nstat_args::ub, lstat_args::ub, stat_args::ub, swapcontext_args::ucp, getcontext_args::ucp, lchown_args::uid, quotactl_-args::uid, fchown_args::uid, setuid_args::uid, chown_args::uid, ksem_getvalue_args::val, sigqueue_-args::value, ksem_open_args::value, ksem_init_args::value, ktimer_gettime_args::value, ktimer_-settime_args::value, lseek_args::whence, __setrlimit_args::which, __getrlimit_args::which, setpriority_-args::which, getitimer_args::which, setitimer_args::which, getrusage_args::who, setpriority_args::who, and abort2_args::why.

9.126 /usr/src/sys/kern/sysv_ipc.c File Reference

```
#include <sys/cdefs.h>
#include "opt_sysvipc.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/sem.h>
#include <sys/shm.h>
#include <sys/ipc.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/ucred.h>
```

Include dependency graph for sysv_ipc.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/sysv_ipc.c,v 1.33 2007/02/20 00:06:59 rwatson Exp $")
- void shmfork (struct proc *p1, struct proc *p2)
- void shmexit (struct vmspace *vm)
- int ipcperm (struct thread *td, struct ipc_perm *perm, int acc_mode)

Variables

- void(*) shmfork_hook (struct proc *, struct proc *) = NULL
- void(*) shmexit_hook (struct vmspace *) = NULL

9.126.1 Function Documentation

9.126.1.1 __FBSDID ("$FreeBSD: src/sys/kern/sysv_ipc. *c*, v 1.33 2007/02/20 00:06:59 rwatson Exp $")

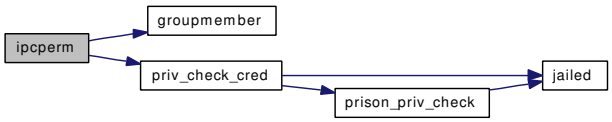
9.126.1.2 int ipcperm (struct thread * *td*, struct ipc_perm * *perm*, int *acc_mode*)

Definition at line 85 of file sysv_ipc.c.

References groupmember(), and priv_check_cred().

Referenced by kern_msgctl(), kern_msgrcv(), kern_msgsnd(), kern_semctl(), kern_shmat(), kern_shmctl(), msgget(), and semget().

Here is the call graph for this function:



9.126.1.3 void shmexit (struct vmspace * *vm*)

Definition at line 66 of file sysv_ipc.c.

References shmexit_hook.

Referenced by exec_new_vmspace().

9.126.1.4 void shmfork (struct proc * *p1*, struct proc * *p2*)

Definition at line 55 of file sysv_ipc.c.

References shmfork_hook.

9.126.2 Variable Documentation

9.126.2.1 void(*) shmexit_hook(struct vmspace *) = NULL

Definition at line 51 of file sysv_ipc.c.

Referenced by shmexit(), shminit(), and shmunload().

9.126.2.2 void(*) shmfork_hook(struct proc *, struct proc *) = NULL

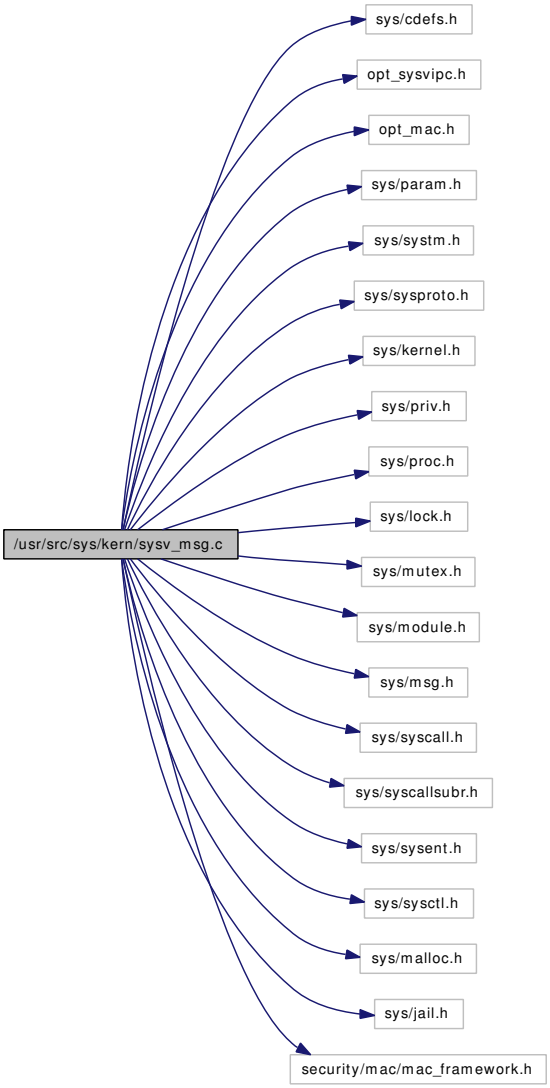
Definition at line 50 of file sysv_ipc.c.

Referenced by shmfork(), shminit(), and shmunload().

9.127 /usr/src/sys/kern/sysv_msg.c File Reference

```
#include <sys/cdefs.h>
#include "opt_sysvipc.h"
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/sysproto.h>
#include <sys/kernel.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/module.h>
#include <sys/msg.h>
#include <sys/syscall.h>
#include <sys/syscallsubr.h>
#include <sys/sysent.h>
#include <sys/sysctl.h>
#include <sys/malloc.h>
#include <sys/jail.h>
#include <security/mac/mac_framework.h>
```

Include dependency graph for sysv_msg.c:



Data Structures

- struct msgmap
- struct msgctl_args
- struct msgget_args
- struct msgsnd_args
- struct msgrcv_args

Defines

- #define DPRINTF(a)
- #define MSGSSZ 8
- #define MSGSEG 2048
- #define MSGMAX (MSGSSZ∗MSGSEG)
- #define MSGMNB 2048

- #define MSGMNI 40
- #define MSGTQL 40
- #define MSQID(ix, ds) ((ix) & 0xffff | (((ds).msg_perm.seq << 16) & 0xffff0000))
- #define MSQID_IX(id) ((id) & 0xffff)
- #define MSQID_SEQ(id) (((id) >> 16) & 0xffff)
- #define MSG_LOCKED 01000

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/sysv_msg.c,v 1.67 2007/02/19 13:23:45 rwatson Exp $")
- static MALLOC_DEFINE (M_MSG,"msg","SVID compatible message queues")
- static void msginit (void)
- static int msgunload (void)
- static int sysvmsg_modload (struct module *, int, void *)
- static void msg_freehdr (struct msg *msghdr)
- SYSCALL_MODULE_HELPER (msgsys)
- SYSCALL_MODULE_HELPER (msgctl)
- SYSCALL_MODULE_HELPER (msgget)
- SYSCALL_MODULE_HELPER (msgsnd)
- SYSCALL_MODULE_HELPER (msgrcv)
- DECLARE_MODULE (sysvmsg, sysvmsg_mod, SI_SUB_SYSV_MSG, SI_ORDER_FIRST)
- MODULE_VERSION (sysvmsg, 1)
- int msgsys (struct thread *td, struct msgsys_args *uap)
- int msgctl (struct thread *td, struct msgctl_args *uap)
- int kern_msgctl (struct thread *td, int msqid, int cmd, struct msqid_ds *msqbuf)
- int msgget (struct thread *td, struct msgget_args *uap)
- int kern_msgsnd (struct thread *td, int msqid, const void *msgp, size_t msgsz, int msgflg, long mtype)
- int msgsnd (struct thread *td, struct msgsnd_args *uap)
- int kern_msgrcv (struct thread *td, int msqid, void *msgp, size_t msgsz, long msgtyp, int msgflg, long *mtype)
- int msgrcv (struct thread *td, struct msgrcv_args *uap)
- static int sysctl_msqids (SYSCTL_HANDLER_ARGS)
- SYSCTL_INT (_kern_ipc, OID_AUTO, msgmax, CTLFLAG_RD,&msginfo.msgmax, 0,"Maximum message size")
- SYSCTL_INT (_kern_ipc, OID_AUTO, msgmni, CTLFLAG_RDTUN,&msginfo.msgmni, 0,"Number of message queue identifiers")
- SYSCTL_INT (_kern_ipc, OID_AUTO, msgmnb, CTLFLAG_RDTUN,&msginfo.msgmnb, 0,"Maximum number of bytes in a queue")
- SYSCTL_INT (_kern_ipc, OID_AUTO, msgtql, CTLFLAG_RDTUN,&msginfo.msgtql, 0,"Maximum number of messages in the system")
- SYSCTL_INT (_kern_ipc, OID_AUTO, msgssz, CTLFLAG_RDTUN,&msginfo.msgssz, 0,"Size of a message segment")
- SYSCTL_INT (_kern_ipc, OID_AUTO, msgseg, CTLFLAG_RDTUN,&msginfo.msgseg, 0,"Number of message segments")
- SYSCTL_PROC (_kern_ipc, OID_AUTO, msqids, CTLFLAG_RD, NULL, 0, sysctl_msqids,"","Message queue IDs")

Variables

- static sy_call_t * msgcalls [ ]
- msginfo msginfo
- static int nfree_msgmaps
- static short free_msgmaps
- static struct msg * free_msghdrs
- static char * msgpool
- static struct msgmap * msgmaps
- static struct msg * msghdrs
- static struct msqid_kernel * msqids
- static struct mtx msq_mtx
- static moduledata_t sysvmsg_mod

9.127.1 Define Documentation

9.127.1.1 #define DPRINTF(a)

Definition at line 84 of file sysv_msg.c.

9.127.1.2 #define MSG_LOCKED 01000

Definition at line 150 of file sysv_msg.c.

Referenced by kern_msgsnd(), and msgunload().

9.127.1.3 #define MSGMAX (MSGSSZ*MSGSEG)

Definition at line 101 of file sysv_msg.c.

9.127.1.4 #define MSGMNB 2048

Definition at line 103 of file sysv_msg.c.

9.127.1.5 #define MSGMNI 40

Definition at line 106 of file sysv_msg.c.

9.127.1.6 #define MSGSEG 2048

Definition at line 99 of file sysv_msg.c.

9.127.1.7 #define MSGSSZ 8

Definition at line 96 of file sysv_msg.c.

9.127.1.8 #define MSGTQL 40

Definition at line 109 of file sysv_msg.c.

9.127.1.9 #define MSQID(ix, ds) ((ix) & 0xffff | (((ds).msg_perm.seq << 16) & 0xffff0000))

Definition at line 136 of file sysv_msg.c.

9.127.1.10 #define MSQID_IX(id) ((id) & 0xffff)

Definition at line 137 of file sysv_msg.c.

9.127.1.11 #define MSQID_SEQ(id) (((id) >> 16) & 0xffff)

Definition at line 138 of file sysv_msg.c.

9.127.2 Function Documentation

9.127.2.1 __FBSDID ("$FreeBSD: src/sys/kern/sysv_msg. *c*, v 1.67 2007/02/19 13:23:45 rwatson Exp $")

9.127.2.2 DECLARE_MODULE (sysvmsg, sysvmsg_mod, SI_SUB_SYSV_MSG, SI_ORDER_FIRST)

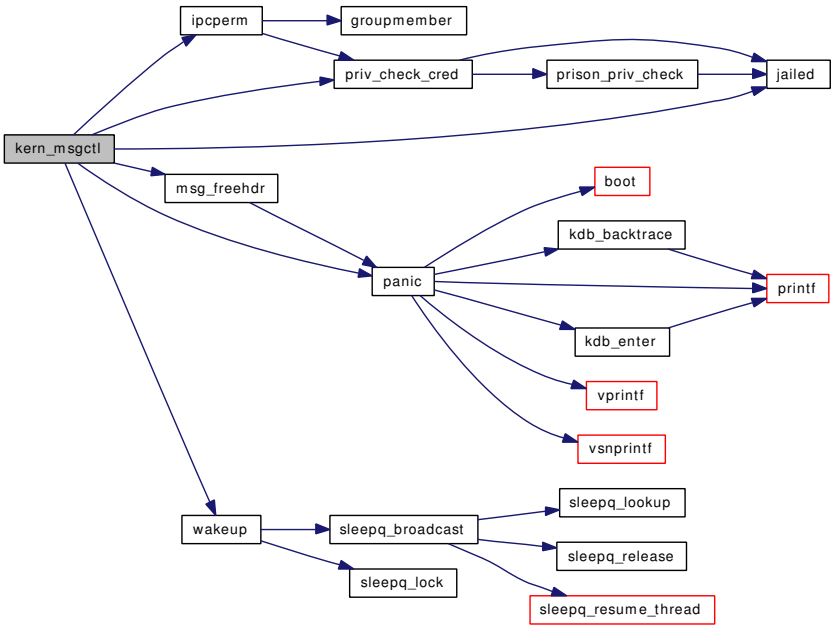
9.127.2.3 int kern_msgctl (struct thread * *td*, int *msqid*, int *cmd*, struct msqid_ds * *msqbuf*)

Definition at line 410 of file sysv_msg.c.

References DPRINTF, ipcperm(), jail_sysvipc_allowed, jailed(), msg_freehdr(), msginfo, msq_mtx, msqids, panic(), priv_check_cred(), time_second, and wakeup().

Referenced by msgctl().

Here is the call graph for this function:



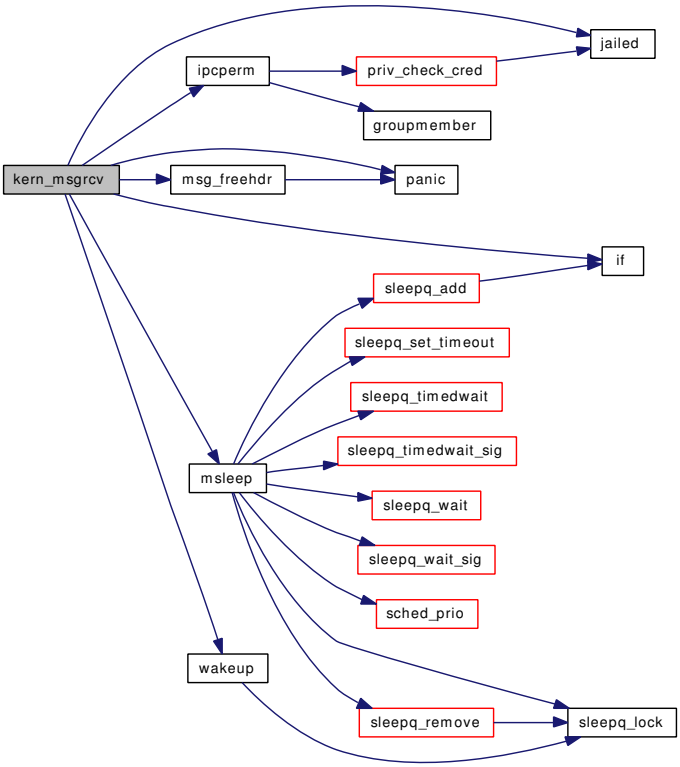
9.127.2.4 int kern_msgrcv (struct thread * *td*, int *msqid*, void * *msgp*, size_t *msgsz*, long *msgtyp*, int *msgflg*, long * *mtype*)

Definition at line 1016 of file sysv_msg.c.

References DPRINTF, if(), ipcperm(), jail_sysvipc_allowed, jailed(), msg_freehdr(), msginfo, msgmaps, msgpool, msleep(), msq_mtx, msqids, msgmap::next, panic(), time_second, and wakeup().

Referenced by msgrcv().

Here is the call graph for this function:



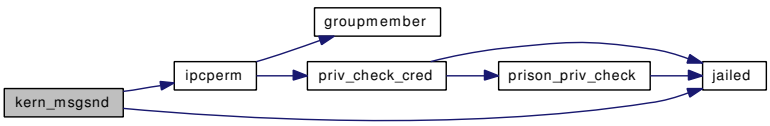
9.127.2.5 int kern_msgsnd (struct thread * *td*, int *msqid*, const void * *msgp*, size_t *msgsz*, int *msgflg*, long *mtype*)

Definition at line 676 of file sysv_msg.c.

References DPRINTF, ipcperm(), jail_sysvipc_allowed, jailed(), MSG_LOCKED, msginfo, msq_mtx, and msqids.

Referenced by msgsnd().

Here is the call graph for this function:



9.127.2.6 static MALLOC_DEFINE (M_MSG, "msg", "SVID compatible message queues") [static]

9.127.2.7 MODULE_VERSION (sysvmsg, 1)

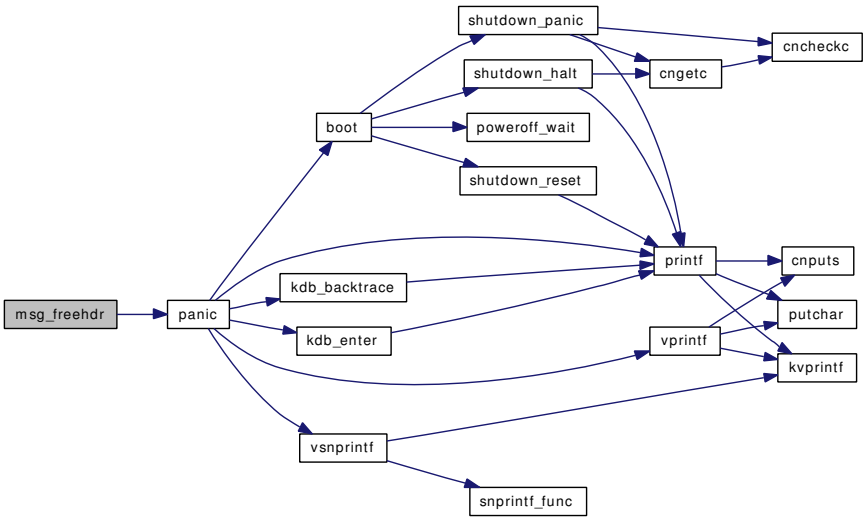
9.127.2.8 static void msg_freehdr (struct msg * *msghdr*) [static]

Definition at line 352 of file sysv_msg.c.

References free_msghdrs, free_msgmaps, msginfo, msgmaps, msgmap::next, nfree_msgmaps, and panic().

Referenced by kern_msgctl(), and kern_msgrcv().

Here is the call graph for this function:

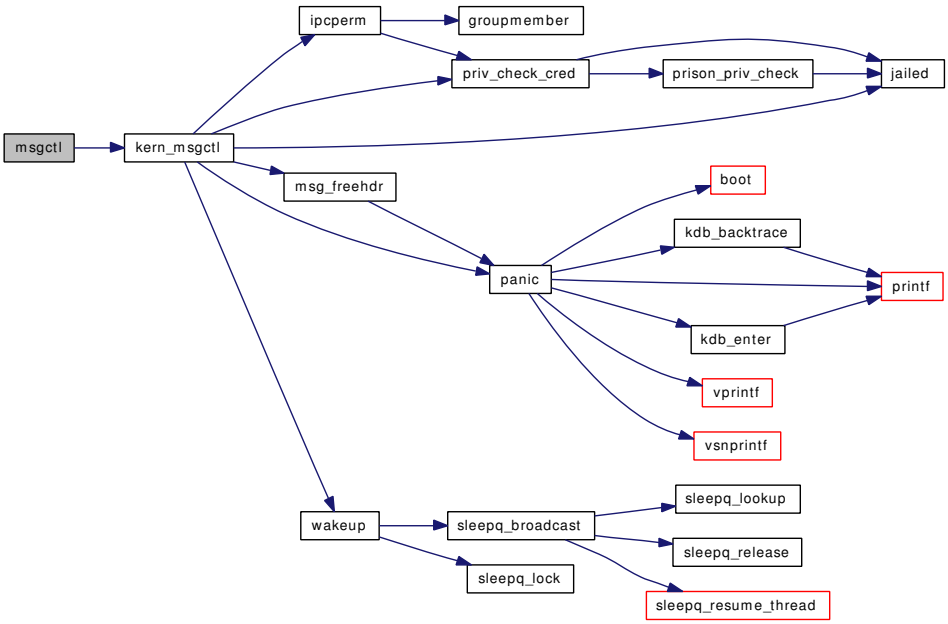


9.127.2.9 int msgctl (struct thread * *td*, struct msgctl_args * *uap*)

Definition at line 390 of file sysv_msg.c.

References DPRINTF, and kern_msgctl().

Here is the call graph for this function:

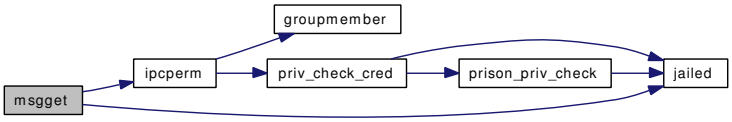


9.127.2.10 int msgget (struct thread * *td*, struct msgget_args * *uap*)

Definition at line 566 of file sysv_msg.c.

References DPRINTF, ipcperm(), jail_sysvipc_allowed, jailed(), msginfo, msq_mtx, and msqids.

Here is the call graph for this function:



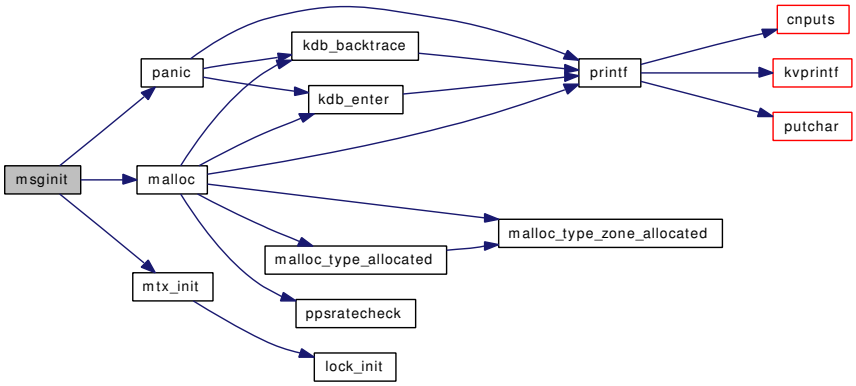
9.127.2.11 static void msginit (void) [static]

Definition at line 162 of file sysv_msg.c.

References DPRINTF, free_msghdrs, free_msgmaps, malloc(), msghdrs, msginfo, msgmaps, msgpool, msq_mtx, msqids, mtx_init(), msgmap::next, nfree_msgmaps, and panic().

Referenced by sysvmsg_modload().

Here is the call graph for this function:

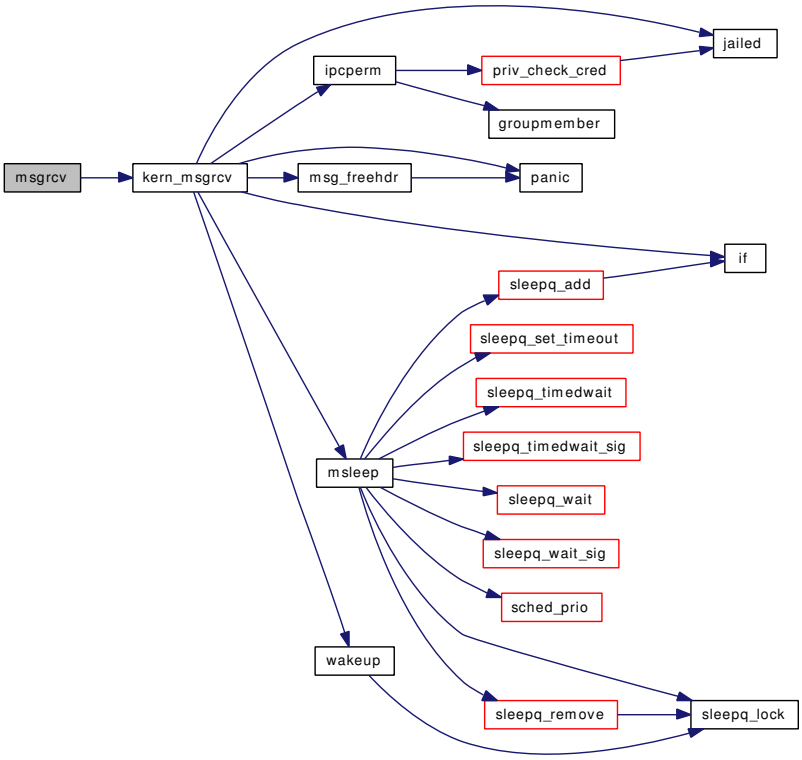


9.127.2.12 int msgrcv (struct thread * *td*, struct msgrcv_args * *uap*)

Definition at line 1272 of file sysv_msg.c.

References DPRINTF, and kern_msgrcv().

Here is the call graph for this function:

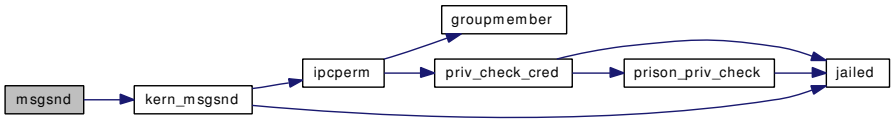


9.127.2.13 int msgsnd (struct thread * *td*, struct msgsnd_args * *uap*)

Definition at line 986 of file sysv_msg.c.

References DPRINTF, and kern_msgsnd().

Here is the call graph for this function:



9.127.2.14 int msgsys (struct thread * *td*, struct msgsys_args * *uap*)

Definition at line 328 of file sysv_msg.c.

References jail_sysvipc_allowed, jailed(), and msgcalls.

Here is the call graph for this function:

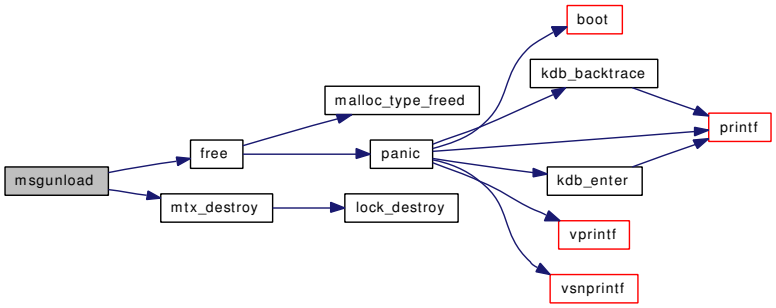
9.127.2.15 static int msgunload (void) [static]

Definition at line 247 of file sysv_msg.c.

References free(), MSG_LOCKED, msghdrs, msginfo, msgmaps, msgpool, msq_mtx, msqids, and mtx_destroy().

Referenced by sysvmsg_modload().

Here is the call graph for this function:



9.127.2.16 SYSCALL_MODULE_HELPER (msgrcv)

9.127.2.17 SYSCALL_MODULE_HELPER (msgsnd)

9.127.2.18 SYSCALL_MODULE_HELPER (msgget)

9.127.2.19 SYSCALL_MODULE_HELPER (msgctl)

9.127.2.20 SYSCALL_MODULE_HELPER (msgsys)

9.127.2.21 SYSCTL_INT (_kern_ipc, OID_AUTO, msgseg, CTLFLAG_RDTUN, &msginfo. *msgseg*, 0, "Number of message segments")

9.127.2.22 SYSCTL_INT (_kern_ipc, OID_AUTO, msgssz, CTLFLAG_RDTUN, &msginfo. *msgssz*, 0, "Size of a message segment")

9.127.2.23 SYSCTL_INT (_kern_ipc, OID_AUTO, msgtql, CTLFLAG_RDTUN, &msginfo. *msgtql*, 0, "Maximum number of messages in the system")

9.127.2.24 SYSCTL_INT (_kern_ipc, OID_AUTO, msgmnb, CTLFLAG_RDTUN, &msginfo. *msgmnb*, 0, "Maximum number of bytes in a queue")

9.127.2.25 SYSCTL_INT (_kern_ipc, OID_AUTO, msgmni, CTLFLAG_RDTUN, &msginfo. *msgmni*, 0, "Number of message queue identifiers")

9.127.2.26 SYSCTL_INT (_kern_ipc, OID_AUTO, msgmax, CTLFLAG_RD, &msginfo. *msgmax*, 0, "Maximum message size")

9.127.2.27 static int sysctl_msqids (SYSCTL_HANDLER_ARGS) [static]

Definition at line 1292 of file sysv_msg.c.

References msginfo, and msqids.

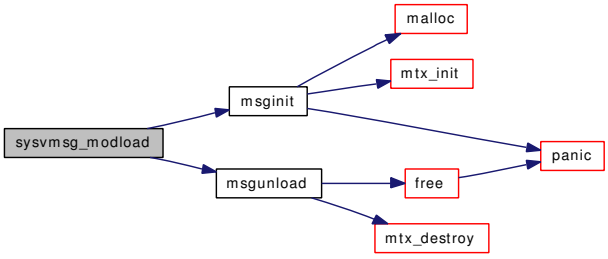
9.127.2.28 SYSCTL_PROC (_kern_ipc, OID_AUTO, msqids, CTLFLAG_RD, NULL, 0, sysctl_msqids, "", "Message queue IDs")

9.127.2.29 static int sysvmsg_modload (struct module *, int, void *) [static]

Definition at line 286 of file sysv_msg.c.

References msginit(), and msgunload().

Here is the call graph for this function:



9.127.3 Variable Documentation

9.127.3.1 struct msg* free_msghdrs [static]

Definition at line 154 of file sysv_msg.c.

Referenced by msg_freehdr(), and msginit().

9.127.3.2 short free_msgmaps [static]

Definition at line 153 of file sysv_msg.c.

Referenced by msg_freehdr(), and msginit().

9.127.3.3 sy_call_t* msgcalls[ ] [static]

Initial value:

```
 {
        (sy_call_t *)msgctl, (sy_call_t *)msgget,
        (sy_call_t *)msgsnd, (sy_call_t *)msgrcv
}
```

Definition at line 90 of file sysv_msg.c.

Referenced by msgsys().

9.127.3.4 struct msg* msghdrs [static]

Definition at line 157 of file sysv_msg.c.

Referenced by msginit(), and msgunload().

9.127.3.5 struct msginfo msginfo

Initial value:

```
 {
                MSGMAX,
                MSGMNI,
                MSGMNB,
                MSGTQL,
                MSGSSZ,

                MSGSEG
}
```

Definition at line 122 of file sysv_msg.c.

Referenced by kern_msgctl(), kern_msgrcv(), kern_msgsnd(), msg_freehdr(), msgget(), msginit(), msgunload(), and sysctl_msqids().

9.127.3.6 struct msgmap* msgmaps [static]

Definition at line 156 of file sysv_msg.c.

Referenced by kern_msgrcv(), msg_freehdr(), msginit(), and msgunload().

9.127.3.7 char* msgpool [static]

Definition at line 155 of file sysv_msg.c.

Referenced by kern_msgrcv(), msginit(), and msgunload().

9.127.3.8 struct mtx msq_mtx [static]

Definition at line 159 of file sysv_msg.c.

Referenced by kern_msgctl(), kern_msgrcv(), kern_msgsnd(), msgget(), msginit(), and msgunload().

9.127.3.9 struct msqid_kernel* msqids [static]

Definition at line 158 of file sysv_msg.c.

Referenced by kern_msgctl(), kern_msgrcv(), kern_msgsnd(), msgget(), msginit(), msgunload(), and sysctl_msqids().

9.127.3.10 int nfree_msgmaps [static]

Definition at line 152 of file sysv_msg.c.

Referenced by msg_freehdr(), and msginit().

9.127.3.11 moduledata_t sysvmsg_mod [static]

Initial value:

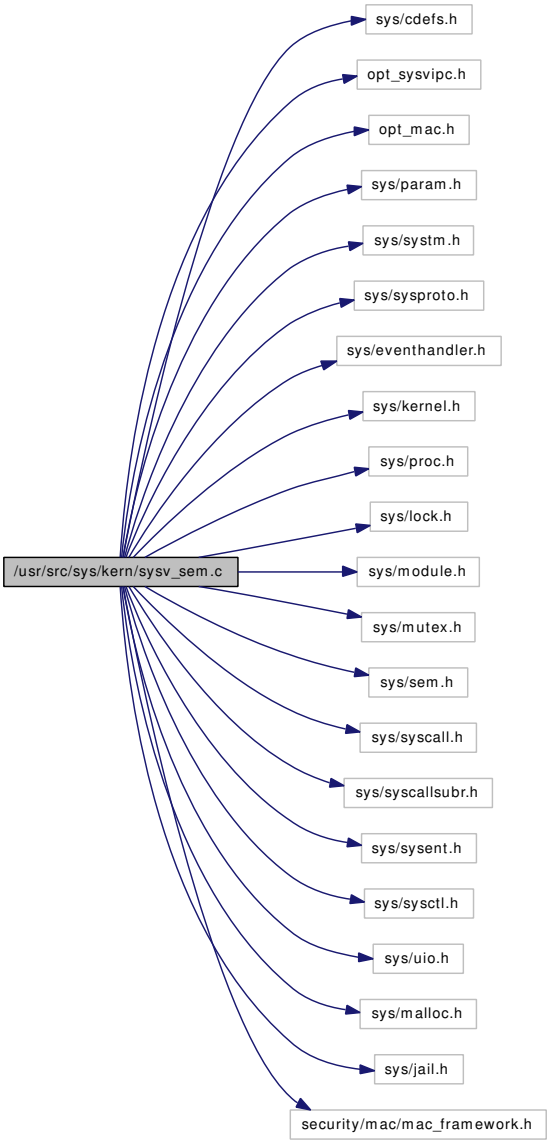
```
{
        "sysvmsg",
        &sysvmsg_modload,
        NULL
}
```

Definition at line 306 of file sysv_msg.c.

9.128 /usr/src/sys/kern/sysv_sem.c File Reference

```
#include <sys/cdefs.h>
#include "opt_sysvipc.h"
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/sysproto.h>
#include <sys/eventhandler.h>
#include <sys/kernel.h>
#include <sys/proc.h>
#include <sys/lock.h>
#include <sys/module.h>
#include <sys/mutex.h>
#include <sys/sem.h>
#include <sys/syscall.h>
#include <sys/syscallsubr.h>
#include <sys/sysent.h>
#include <sys/sysctl.h>
#include <sys/uio.h>
#include <sys/malloc.h>
#include <sys/jail.h>
#include <security/mac/mac_framework.h>
```

Include dependency graph for sysv_sem.c:



Data Structures

- struct sem_undo
- struct __semctl_args
- struct semget_args
- struct semop_args

Defines

- #define DPRINTF(a)
- #define SEMUNDO_MTX sem_mtx
- #define SEMUNDO_LOCK() mtx_lock(&SEMUNDO_MTX);
- #define SEMUNDO_UNLOCK() mtx_unlock(&SEMUNDO_MTX);
- #define SEMUNDO_LOCKASSERT(how) mtx_assert(&SEMUNDO_MTX, (how));

- #define SEMMNI 10
- #define SEMMNS 60
- #define SEMUME 10
- #define SEMMNU 30
- #define SEMMAP 30
- #define SEMMSL SEMMNS
- #define SEMOPM 100
- #define SEMVMX 32767
- #define SEMAEM 16384
- #define SEM_ALIGN(bytes) (((bytes) + (sizeof(long) - 1)) & ~(sizeof(long) - 1))
- #define SEMUSZ SEM_ALIGN(offsetof(struct sem_undo, un_ent[SEMUME]))
- #define SEMU(ix) ((struct sem_undo *)(((intptr_t)semu)+ix * seminfo.semusz))
- #define SMALL_SOPS 8

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/sysv_sem.c,v 1.85 2006/10/22 11:52:13 rwatson Exp $")
- static MALLOC_DEFINE (M_SEM,"sem","SVID compatible semaphores")
- static void seminit (void)
- static int sysvsem_modload (struct module *, int, void *)
- static int semunload (void)
- static void semexit_myhook (void *arg, struct proc *p)
- static int sysctl_sema (SYSCTL_HANDLER_ARGS)
- static int semvalid (int semid, struct semid_kernel *semakptr)
- int __semctl (struct thread *td, struct __semctl_args *uap)
- int semget (struct thread *td, struct semget_args *uap)
- int semop (struct thread *td, struct semop_args *uap)
- static struct sem_undo * semu_alloc (struct thread *td)
- static int semundo_adjust (struct thread *td, struct sem_undo **supptr, int semid, int semnum, int adjval)
- static void semundo_clear (int semid, int semnum)
- SLIST_HEAD (sem_undo)
- SYSCTL_INT (_kern_ipc, OID_AUTO, semmap, CTLFLAG_RW,&seminfo.semmap, 0,"Number of entries in the semaphore map")
- SYSCTL_INT (_kern_ipc, OID_AUTO, semmni, CTLFLAG_RDTUN,&seminfo.semmni, 0,"Number of semaphore identifiers")
- SYSCTL_INT (_kern_ipc, OID_AUTO, semmns, CTLFLAG_RDTUN,&seminfo.semmns, 0,"Maximum number of semaphores in the system")
- SYSCTL_INT (_kern_ipc, OID_AUTO, semmnu, CTLFLAG_RDTUN,&seminfo.semmnu, 0,"Maximum number of undo structures in the system")
- SYSCTL_INT (_kern_ipc, OID_AUTO, semmsl, CTLFLAG_RW,&seminfo.semmsl, 0,"Max semaphores per id")
- SYSCTL_INT (_kern_ipc, OID_AUTO, semopm, CTLFLAG_RDTUN,&seminfo.semopm, 0,"Max operations per semop call")
- SYSCTL_INT (_kern_ipc, OID_AUTO, semume, CTLFLAG_RDTUN,&seminfo.semume, 0,"Max undo entries per process")
- SYSCTL_INT (_kern_ipc, OID_AUTO, semusz, CTLFLAG_RDTUN,&seminfo.semusz, 0,"Size in bytes of undo structure")
- SYSCTL_INT (_kern_ipc, OID_AUTO, semvmx, CTLFLAG_RW,&seminfo.semvmx, 0,"Semaphore maximum value")

- SYSCTL_INT (_kern_ipc, OID_AUTO, semaem, CTLFLAG_RW,&seminfo.semaem, 0,"Adjust on exit max value")
- SYSCTL_PROC (_kern_ipc, OID_AUTO, sema, CTLFLAG_RD, NULL, 0, sysctl_sema,"","")
- SYSCALL_MODULE_HELPER (semsys)
- SYSCALL_MODULE_HELPER (__semctl)
- SYSCALL_MODULE_HELPER (semget)
- SYSCALL_MODULE_HELPER (semop)
- DECLARE_MODULE (sysvsem, sysvsem_mod, SI_SUB_SYSV_SEM, SI_ORDER_FIRST)
- MODULE_VERSION (sysvsem, 1)
- int semsys (struct thread *td, struct semsys_args *uap)
- int kern_semctl (struct thread *td, int semid, int semnum, int cmd, union semun *arg, register_t *rval)

Variables

- static sy_call_t * semcalls [ ]
- static struct mtx sem_mtx
- static int semtot = 0
- static struct semid_kernel * sema
- static struct mtx * sema_mtx
- static struct sem * sem
- seminfo seminfo
- static moduledata_t sysvsem_mod

9.128.1 Define Documentation

9.128.1.1 #define DPRINTF(a)

Definition at line 70 of file sysv_sem.c.

9.128.1.2 #define SEM_ALIGN(bytes) (((bytes) + (sizeof(long) - 1)) & ~(sizeof(long) - 1))

Definition at line 170 of file sysv_sem.c.

9.128.1.3 #define SEMAEM 16384

Definition at line 163 of file sysv_sem.c.

9.128.1.4 #define SEMMAP 30

Definition at line 153 of file sysv_sem.c.

9.128.1.5 #define SEMMNI 10

Definition at line 139 of file sysv_sem.c.

9.128.1.6 #define SEMMNS 60

Definition at line 142 of file sysv_sem.c.

9.128.1.7 #define SEMMNU 30

Definition at line 148 of file sysv_sem.c.

9.128.1.8 #define SEMMSL SEMMNS

Definition at line 156 of file sysv_sem.c.

9.128.1.9 #define SEMOPM 100

Definition at line 159 of file sysv_sem.c.

9.128.1.10 #define SEMU(ix) ((struct sem_undo *)(((intptr_t)semu)+ix * seminfo.semusz))

Definition at line 178 of file sysv_sem.c.

Referenced by seminit(), and semu_alloc().

9.128.1.11 #define SEMUME 10

Definition at line 145 of file sysv_sem.c.

9.128.1.12 #define SEMUNDO_LOCK() mtx_lock(&SEMUNDO_MTX);

Referenced by kern_semctl(), and semexit_myhook().

9.128.1.13 #define SEMUNDO_LOCKASSERT(how) mtx_assert(&SEMUNDO_MTX, (how));

Referenced by semu_alloc(), semundo_adjust(), and semundo_clear().

9.128.1.14 #define SEMUNDO_MTX sem_mtx

9.128.1.15 #define SEMUNDO_UNLOCK() mtx_unlock(&SEMUNDO_MTX);

Referenced by kern_semctl(), and semexit_myhook().

9.128.1.16 #define SEMUSZ SEM_ALIGN(offsetof(struct sem_undo, un_ent[SEMUME]))

Definition at line 173 of file sysv_sem.c.

9.128.1.17 #define SEMVMX 32767

Definition at line 162 of file sysv_sem.c.

9.128.1.18 #define SMALL_SOPS 8

9.128.2 Function Documentation

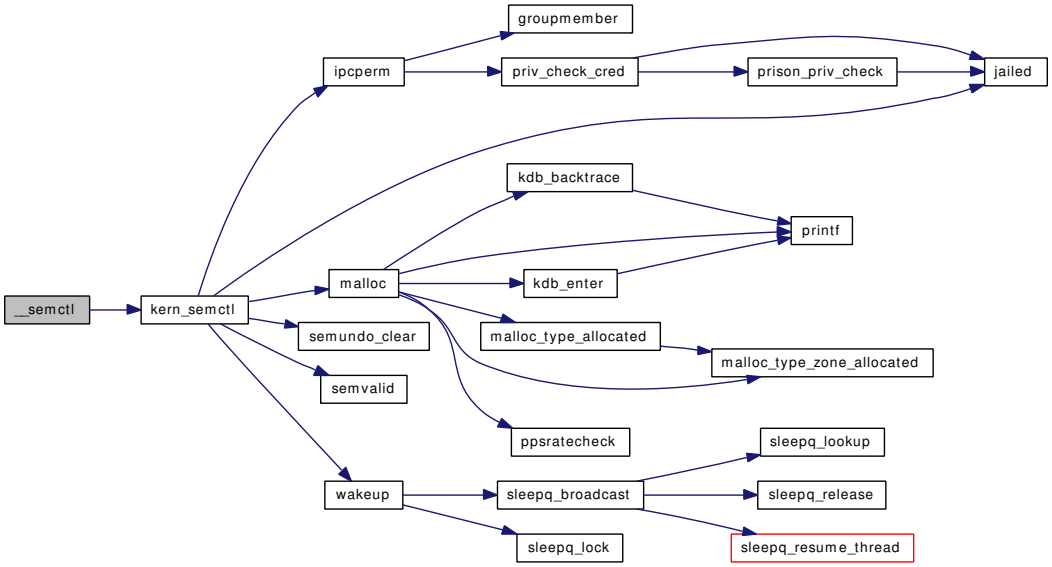
9.128.2.1 __FBSDID ("$FreeBSD: src/sys/kern/sysv_sem. *c*, v 1.85 2006/10/22 11:52:13 rwatson Exp $")

9.128.2.2 int __semctl (struct thread * *td*, struct __semctl_args * *uap*)

Definition at line 551 of file sysv_sem.c.

References __semctl_args::arg, __semctl_args::cmd, kern_semctl(), __semctl_args::semid, and __semctl_-args::semnum.

Here is the call graph for this function:



9.128.2.3 DECLARE_MODULE (sysvsem, sysvsem_mod, SI_SUB_SYSV_SEM, SI_ORDER_FIRST)

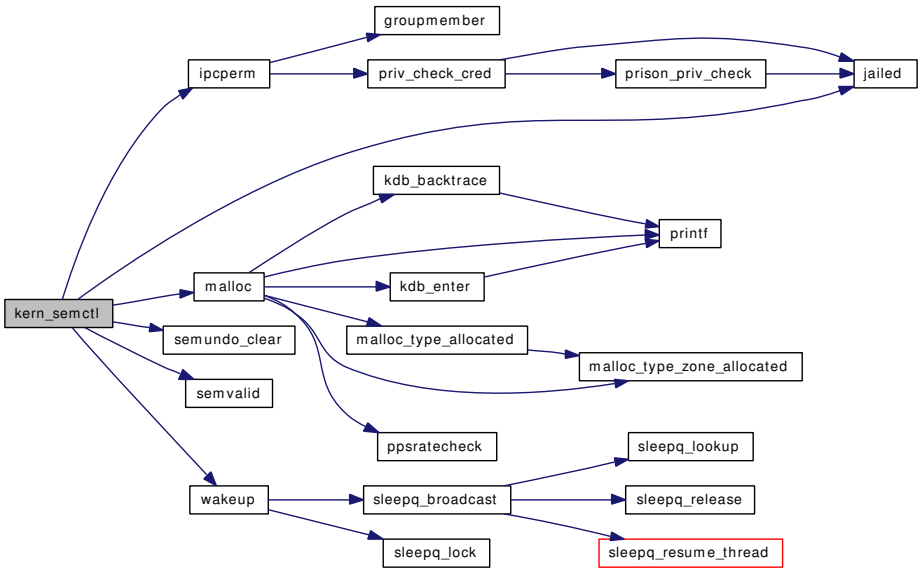
9.128.2.4 int kern_semctl (struct thread * *td*, int *semid*, int *semnum*, int *cmd*, union semun * *arg*, register_t * *rval*)

Definition at line 611 of file sysv_sem.c.

References DPRINTF, ipcperm(), jail_sysvipc_allowed, jailed(), malloc(), semundo_clear(), SEMUNDO_LOCK, SEMUNDO_UNLOCK, semvalid(), time_second, and wakeup().

Referenced by __semctl().

Here is the call graph for this function:



9.128.2.5 static MALLOC_DEFINE (M_SEM, "sem", "SVID compatible semaphores") [static]

9.128.2.6 MODULE_VERSION (sysvsem, 1)

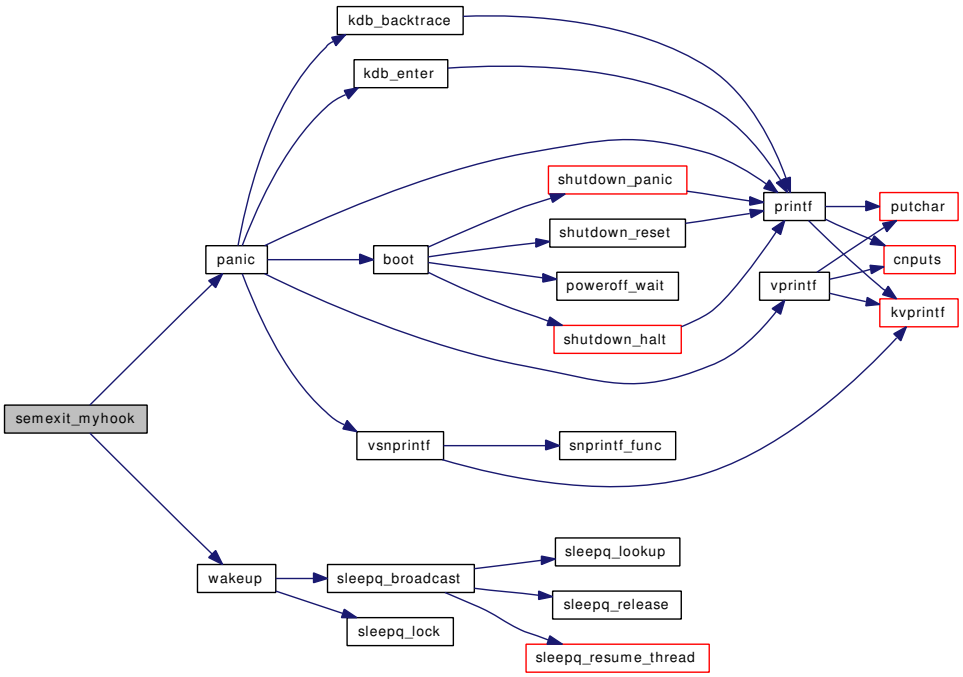
9.128.2.7 static void semexit_myhook (void * *arg*, struct proc * *p*) [static]

Definition at line 1291 of file sysv_sem.c.

References DPRINTF, panic(), SEMUNDO_LOCK, SEMUNDO_UNLOCK, sem_undo::un_ent, and wakeup().

Referenced by seminit().

Here is the call graph for this function:

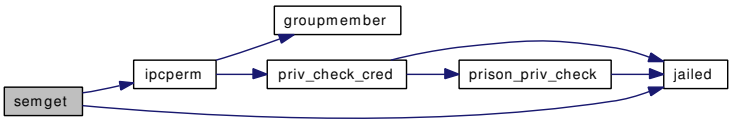


9.128.2.8 int semget (struct thread * *td*, struct semget_args * *uap*)

Definition at line 881 of file sysv_sem.c.

References DPRINTF, Giant, ipcperm(), jail_sysvipc_allowed, jailed(), semget_args::key, semget_-args::nsems, and semget_args::semflg.

Here is the call graph for this function:



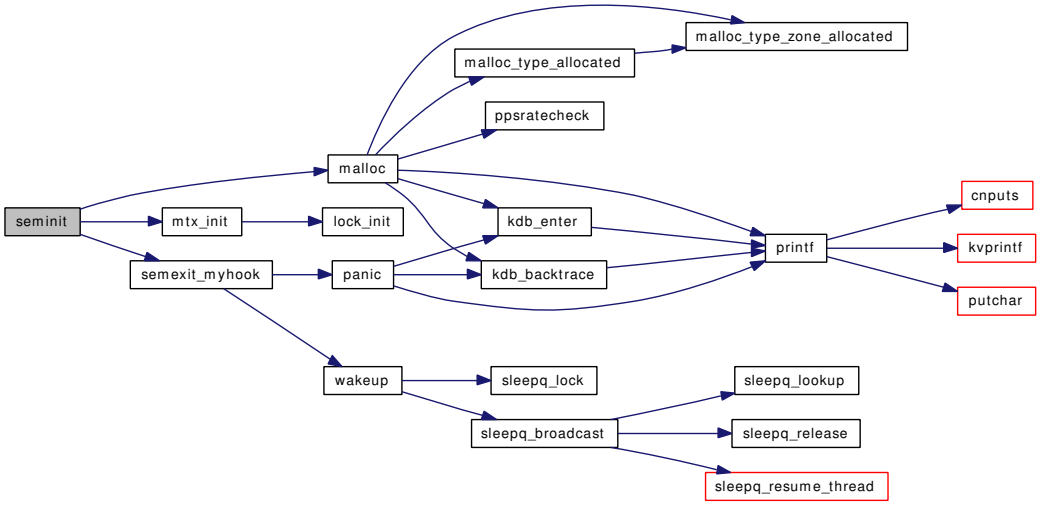
9.128.2.9 static void seminit (void) [static]

Definition at line 221 of file sysv_sem.c.

References malloc(), mtx_init(), semexit_myhook(), and SEMU.

Referenced by sysvsem_modload().

Here is the call graph for this function:

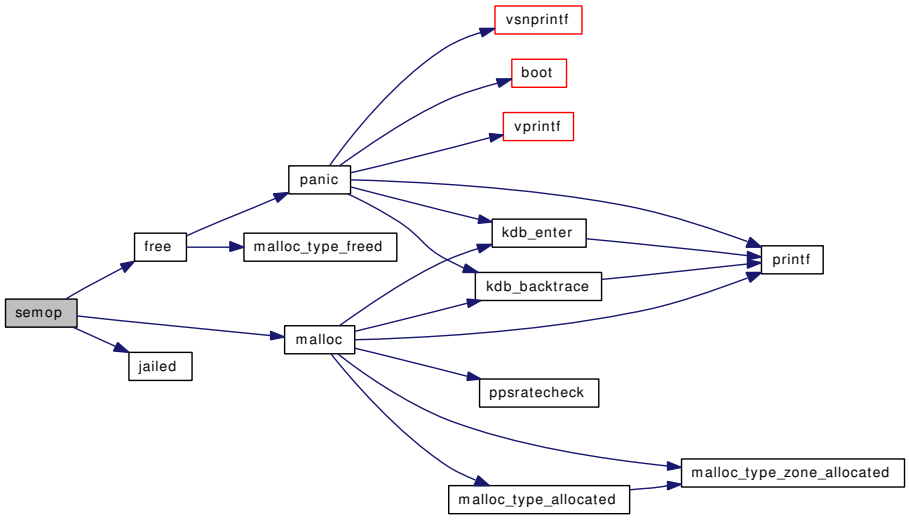


9.128.2.10 int semop (struct thread * *td*, struct semop_args * *uap*)

Definition at line 997 of file sysv_sem.c.

References DPRINTF, free(), jail_sysvipc_allowed, jailed(), malloc(), semop_args::nsops, semop_-args::semid, and semop_args::sops.

Here is the call graph for this function:



9.128.2.11 int semsys (struct thread * *td*, struct semsys_args * *uap*)

Definition at line 327 of file sysv_sem.c.

References jail_sysvipc_allowed, jailed(), and semcalls.

Here is the call graph for this function:

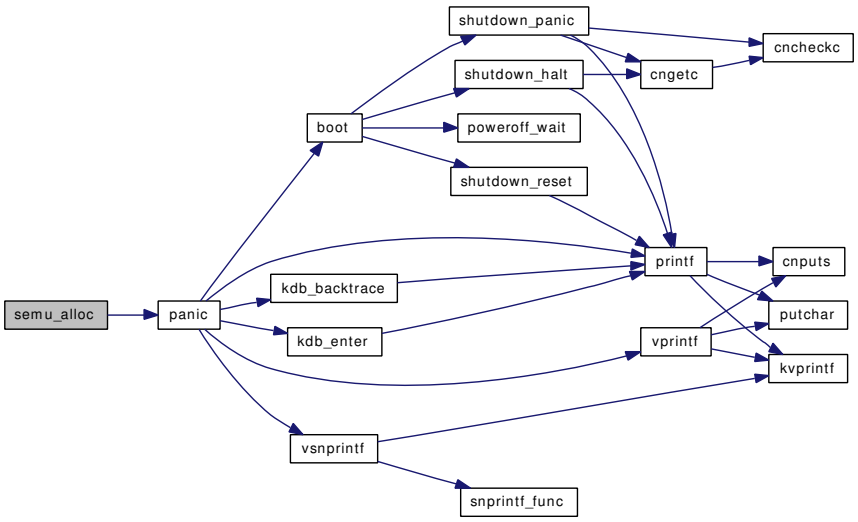
9.128.2.12 static struct sem_undo * semu_alloc (struct thread * *td*) [static]

Definition at line 355 of file sysv_sem.c.

References panic(), SEMU, and SEMUNDO_LOCKASSERT.

Referenced by semundo_adjust().

Here is the call graph for this function:

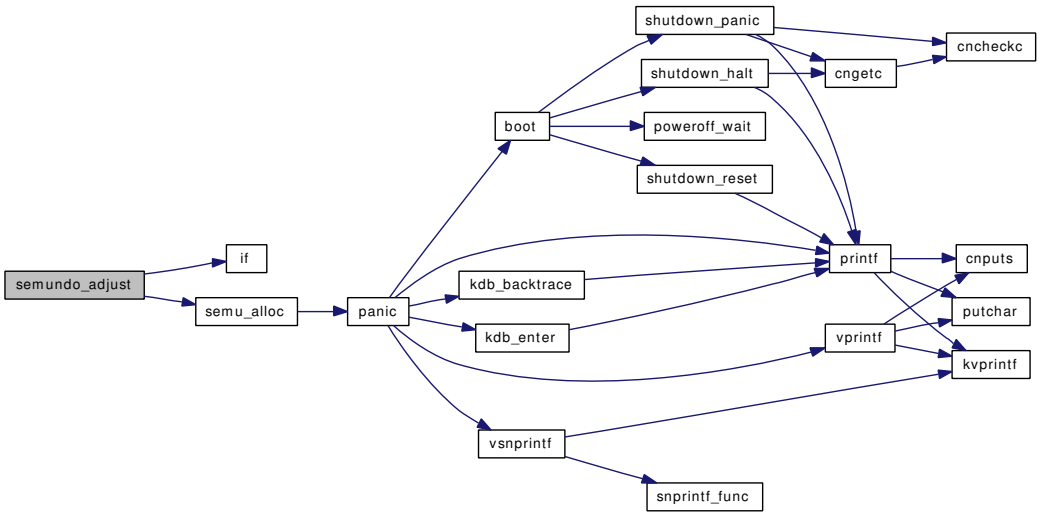


9.128.2.13 static int semundo_adjust (struct thread * *td*, struct sem_undo ** *supptr*, int *semid*, int *semnum*, int *adjval*) [static]

Definition at line 425 of file sysv_sem.c.

References if(), semu_alloc(), SEMUNDO_LOCKASSERT, and sem_undo::un_ent.

Here is the call graph for this function:



9.128.2.14 static void semundo_clear (int *semid*, int *semnum*) [static]

Definition at line 497 of file sysv_sem.c.

References SEMUNDO_LOCKASSERT, and sem_undo::un_ent.

Referenced by kern_semctl().

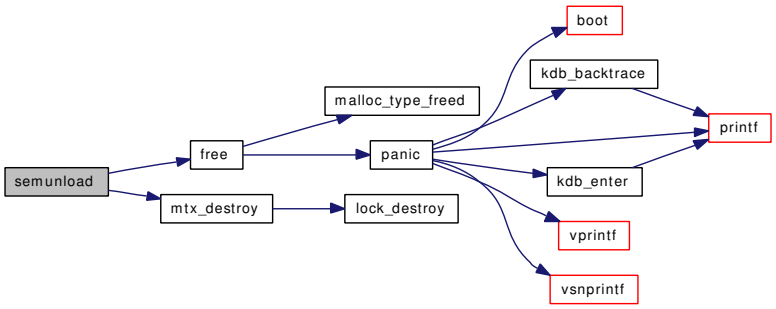
9.128.2.15 static int semunload (void) [static]

Definition at line 264 of file sysv_sem.c.

References free(), and mtx_destroy().

Referenced by sysvsem_modload().

Here is the call graph for this function:



9.128.2.16 static int semvalid (int *semid*, struct semid_kernel * *semakptr*) [static]

Definition at line 526 of file sysv_sem.c.

Referenced by kern_semctl().

9.128.2.17 SLIST_HEAD (sem_undo)

Definition at line 105 of file sysv_sem.c.

9.128.2.18 SYSCALL_MODULE_HELPER (semop)

9.128.2.19 SYSCALL_MODULE_HELPER (semget)

9.128.2.20 SYSCALL_MODULE_HELPER (__semctl)

9.128.2.21 SYSCALL_MODULE_HELPER (semsys)

9.128.2.22 SYSCTL_INT (_kern_ipc, OID_AUTO, semaem, CTLFLAG_RW, &seminfo. *semaem*, 0, "Adjust on exit max value")

9.128.2.23 SYSCTL_INT (_kern_ipc, OID_AUTO, semvmx, CTLFLAG_RW, &seminfo. *semvmx*, 0, "Semaphore maximum value")

9.128.2.24 SYSCTL_INT (_kern_ipc, OID_AUTO, semusz, CTLFLAG_RDTUN, &seminfo. *semusz*, 0, "Size in bytes of undo structure")

9.128.2.25 SYSCTL_INT (_kern_ipc, OID_AUTO, semume, CTLFLAG_RDTUN, &seminfo. *semume*, 0, "Max undo entries per process")

9.128.2.26 SYSCTL_INT (_kern_ipc, OID_AUTO, semopm, CTLFLAG_RDTUN, &seminfo. *semopm*, 0, "Max operations per semop call")

9.128.2.27 SYSCTL_INT (_kern_ipc, OID_AUTO, semmsl, CTLFLAG_RW, &seminfo. *semmsl*, 0, "Max semaphores per id")

9.128.2.28 SYSCTL_INT (_kern_ipc, OID_AUTO, semmnu, CTLFLAG_RDTUN, &seminfo. *semmnu*, 0, "Maximum number of undo structures in the system")

9.128.2.29 SYSCTL_INT (_kern_ipc, OID_AUTO, semmns, CTLFLAG_RDTUN, &seminfo. *semmns*, 0, "Maximum number of semaphores in the system")

9.128.2.30 SYSCTL_INT (_kern_ipc, OID_AUTO, semmni, CTLFLAG_RDTUN, &seminfo. *semmni*, 0, "Number of semaphore identifiers")

9.128.2.31 SYSCTL_INT (_kern_ipc, OID_AUTO, semmap, CTLFLAG_RW, &seminfo. *semmap*, 0, "Number of entries in the semaphore map")

9.128.2.32 SYSCTL_PROC (_kern_ipc, OID_AUTO, sema, CTLFLAG_RD, NULL, 0, sysctl_sema, "", "")

9.128.2.33 static int sysctl_sema (SYSCTL_HANDLER_ARGS) [static]

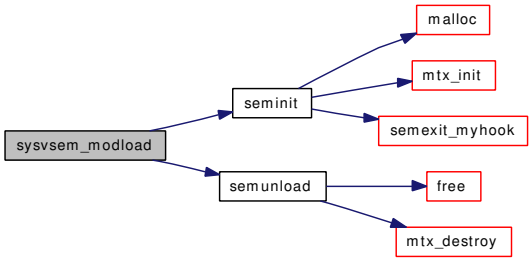
Definition at line 1370 of file sysv_sem.c.

9.128.2.34 static int sysvsem_modload (struct module *, int, void *) [static]

Definition at line 286 of file sysv_sem.c.

References seminit(), and semunload().

Here is the call graph for this function:



9.128.3 Variable Documentation

9.128.3.1 struct sem* sem [static]

Definition at line 104 of file sysv_sem.c.

9.128.3.2 struct mtx sem_mtx [static]

Definition at line 100 of file sysv_sem.c.

9.128.3.3 struct semid_kernel* sema [static]

Definition at line 102 of file sysv_sem.c.

9.128.3.4 struct mtx* sema_mtx [static]

Definition at line 103 of file sysv_sem.c.

9.128.3.5 sy_call_t* semcalls[ ] [static]

Initial value:

```
 {
        (sy_call_t *)__semctl, (sy_call_t *)semget,
        (sy_call_t *)semop
}
```

Definition at line 95 of file sysv_sem.c.

Referenced by semsys().

9.128.3.6 struct seminfo seminfo

Initial value:

```
 {
                SEMMAP,
                SEMMNI,
                SEMMNS,
                SEMMNU,
                SEMMSL,
                SEMOPM,
                SEMUME,
                SEMUSZ,
                SEMVMX,
                SEMAEM
}
```

Definition at line 184 of file sysv_sem.c.

9.128.3.7 int semtot = 0 [static]

Definition at line 101 of file sysv_sem.c.

9.128.3.8 moduledata_t sysvsem_mod [static]

Initial value:

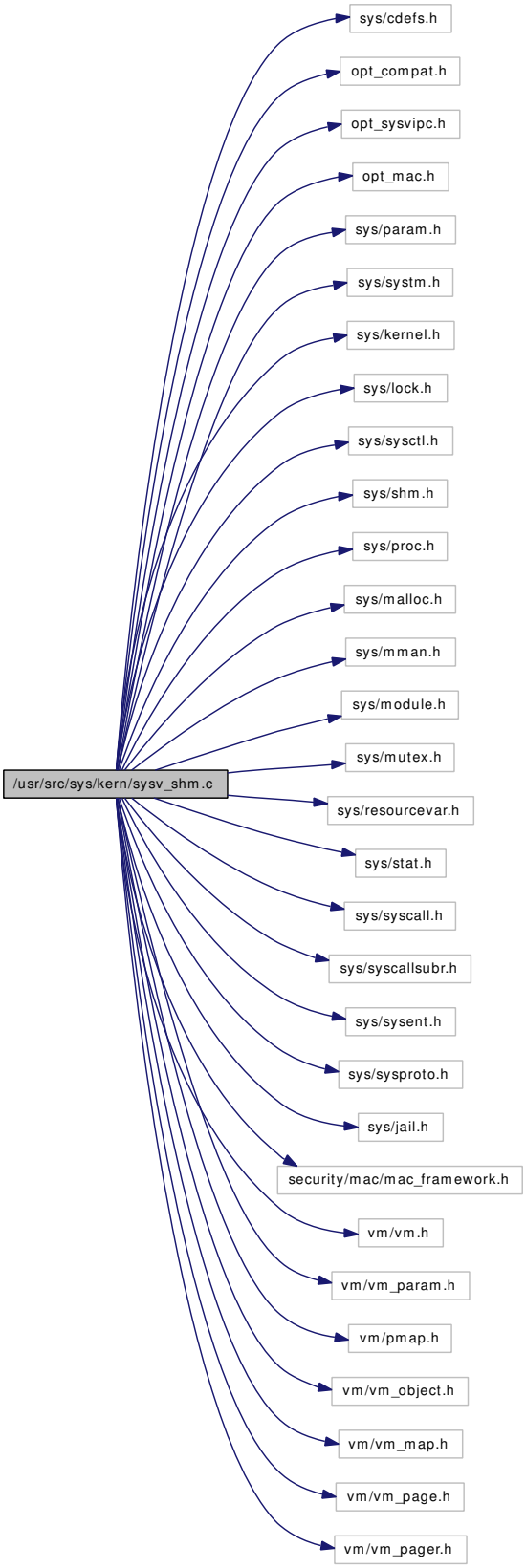
```
 {
        "sysvsem",
        &sysvsem_modload,
        NULL
}
```

Definition at line 306 of file sysv_sem.c.

9.129 /usr/src/sys/kern/sysv_shm.c File Reference

```
#include <sys/cdefs.h>
#include "opt_compat.h"
#include "opt_sysvipc.h"
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/sysctl.h>
#include <sys/shm.h>
#include <sys/proc.h>
#include <sys/malloc.h>
#include <sys/mman.h>
#include <sys/module.h>
#include <sys/mutex.h>
#include <sys/resourcevar.h>
#include <sys/stat.h>
#include <sys/syscall.h>
#include <sys/syscallsubr.h>
#include <sys/sysent.h>
#include <sys/sysproto.h>
#include <sys/jail.h>
#include <security/mac/mac_framework.h>
#include <vm/vm.h>
#include <vm/vm_param.h>
#include <vm/pmap.h>
#include <vm/vm_object.h>
#include <vm/vm_map.h>
#include <vm/vm_page.h>
#include <vm/vm_pager.h>
```

Include dependency graph for sysv_shm.c:


/usr/src/sys/kern/sysv_shm.c
sys/cdefs.h
opt_compat.h
opt_sysvipc.h
opt_mac.h
sys/param.h
sys/systm.h
sys/kernel.h
sys/lock.h
sys/sysctl.h
sys/shm.h
sys/proc.h
sys/malloc.h
sys/mman.h
sys/module.h
sys/mutex.h
sys/resourcevar.h
sys/stat.h
sys/syscall.h
sys/syscallsubr.h
sys/sysent.h
sys/sysproto.h
sys/jail.h
security/mac/mac_framework.h
vm/vm.h
vm/vm_param.h
vm/pmap.h
vm/vm_object.h
vm/vm_map.h
vm/vm_page.h
vm/vm_pager.h

Data Structures

- struct shmmap_state
- struct shmdt_args
- struct shmat_args
- struct shmctl_args
- struct shmget_args

Defines

- #define SHMSEG_FREE 0x0200
- #define SHMSEG_REMOVED 0x0400
- #define SHMSEG_ALLOCATED 0x0800
- #define SHMSEG_WANTED 0x1000
- #define SHMMAXPGS 8192
- #define SHMMAX (SHMMAXPGS∗PAGE_SIZE)
- #define SHMMIN 1
- #define SHMMNI 192
- #define SHMSEG 128
- #define SHMALL (SHMMAXPGS)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/sysv_shm.c,v 1.109 2007/02/19 22:56:10 rwatson Exp $")
- static MALLOC_DEFINE (M_SHM,"shm","SVID compatible shared memory segments")
- static int shmget_allocate_segment (struct thread ∗td, struct shmget_args ∗uap, int mode)
- static int shmget_existing (struct thread ∗td, struct shmget_args ∗uap, int mode, int segnum)
- static void shm_deallocate_segment (struct shmid_kernel ∗)
- static int shm_find_segment_by_key (key_t)
- static struct shmid_kernel ∗ shm_find_segment_by_shmid (int)
- static struct shmid_kernel ∗ shm_find_segment_by_shmidx (int)
- static int shm_delete_mapping (struct vmspace ∗vm, struct shmmap_state ∗)
- static void shmrealloc (void)
- static void shminit (void)
- static int sysvshm_modload (struct module ∗, int, void ∗)
- static int shmunload (void)
- static void shmexit_myhook (struct vmspace ∗vm)
- static void shmfork_myhook (struct proc ∗p1, struct proc ∗p2)
- static int sysctl_shmsegs (SYSCTL_HANDLER_ARGS)
- SYSCTL_ULONG (_kern_ipc, OID_AUTO, shmmax, CTLFLAG_RW,&shminfo.shmmax, 0,"Maximum shared memory segment size")
- SYSCTL_ULONG (_kern_ipc, OID_AUTO, shmmin, CTLFLAG_RW,&shminfo.shmmin, 0,"Minimum shared memory segment size")
- SYSCTL_ULONG (_kern_ipc, OID_AUTO, shmmni, CTLFLAG_RDTUN,&shminfo.shmmni, 0,"Number of shared memory identifiers")
- SYSCTL_ULONG (_kern_ipc, OID_AUTO, shmseg, CTLFLAG_RDTUN,&shminfo.shmseg, 0,"Number of segments per process")
- SYSCTL_ULONG (_kern_ipc, OID_AUTO, shmall, CTLFLAG_RW,&shminfo.shmall, 0,"Maximum number of pages available for shared memory")

- SYSCTL_INT (_kern_ipc, OID_AUTO, shm_use_phys, CTLFLAG_RW,&shm_use_phys, 0,"Enable/Disable locking of shared memory pages in core")
- SYSCTL_INT (_kern_ipc, OID_AUTO, shm_allow_removed, CTLFLAG_RW,&shm_allow_removed, 0,"Enable/Disable attachment to attached segments marked for removal")
- SYSCTL_PROC (_kern_ipc, OID_AUTO, shmsegs, CTLFLAG_RD, NULL, 0, sysctl_shmsegs,"","Current number of shared memory segments allocated")
- int shmdt (struct thread *td, struct shmdt_args *uap)
- int kern_shmat (struct thread *td, int shmid, const void *shmaddr, int shmflg)
- int shmat (struct thread *td, struct shmat_args *uap)
- int kern_shmctl (struct thread *td, int shmid, int cmd, void *buf, size_t *bufsz)
- int shmctl (struct thread *td, struct shmctl_args *uap)
- int shmget (struct thread *td, struct shmget_args *uap)
- int shmsys (struct thread *td, struct shmsys_args *uap)
- SYSCALL_MODULE_HELPER (shmsys)
- SYSCALL_MODULE_HELPER (shmat)
- SYSCALL_MODULE_HELPER (shmctl)
- SYSCALL_MODULE_HELPER (shmdt)
- SYSCALL_MODULE_HELPER (shmget)
- DECLARE_MODULE (sysvshm, sysvshm_mod, SI_SUB_SYSV_SHM, SI_ORDER_FIRST)
- MODULE_VERSION (sysvshm, 1)

Variables

- static int shm_last_free
- static int shm_nused
- static int shm_committed
- static int shmalloced
- static struct shmid_kernel * shmsegs
- shminfo shminfo
- static int shm_use_phys
- static int shm_allow_removed
- static moduledata_t sysvshm_mod

9.129.1 Define Documentation

9.129.1.1 #define SHMALL (SHMMAXPGS)

Definition at line 164 of file sysv_shm.c.

9.129.1.2 #define SHMMAX (SHMMAXPGS*PAGE_SIZE)

Definition at line 152 of file sysv_shm.c.

9.129.1.3 #define SHMMAXPGS 8192

Definition at line 149 of file sysv_shm.c.

9.129.1.4 #define SHMMIN 1

Definition at line 155 of file sysv_shm.c.

9.129.1.5 #define SHMMNI 192

Definition at line 158 of file sysv_shm.c.

9.129.1.6 #define SHMSEG 128

Definition at line 161 of file sysv_shm.c.

9.129.1.7 #define SHMSEG_ALLOCATED 0x0800

Definition at line 121 of file sysv_shm.c.

Referenced by shm_find_segment_by_key(), shm_find_segment_by_shmid(), shm_find_segment_by_-shmidx(), and shmget_allocate_segment().

9.129.1.8 #define SHMSEG_FREE 0x0200

Definition at line 119 of file sysv_shm.c.

Referenced by shm_deallocate_segment(), shmget_allocate_segment(), shminit(), and shmrealloc().

9.129.1.9 #define SHMSEG_REMOVED 0x0400

Definition at line 120 of file sysv_shm.c.

Referenced by kern_shmctl(), shm_delete_mapping(), shm_find_segment_by_shmid(), shm_find_-segment_by_shmidx(), shmget_allocate_segment(), and shmget_existing().

9.129.1.10 #define SHMSEG_WANTED 0x1000

Definition at line 122 of file sysv_shm.c.

Referenced by shmget_allocate_segment(), and shmget_existing().

9.129.2 Function Documentation

9.129.2.1 __FBSDID ("$FreeBSD: src/sys/kern/sysv_shm. *c*, v 1.109 2007/02/19 22:56:10 rwatson Exp $")

9.129.2.2 DECLARE_MODULE (sysvshm, sysvshm_mod, SI_SUB_SYSV_SHM, SI_ORDER_FIRST)

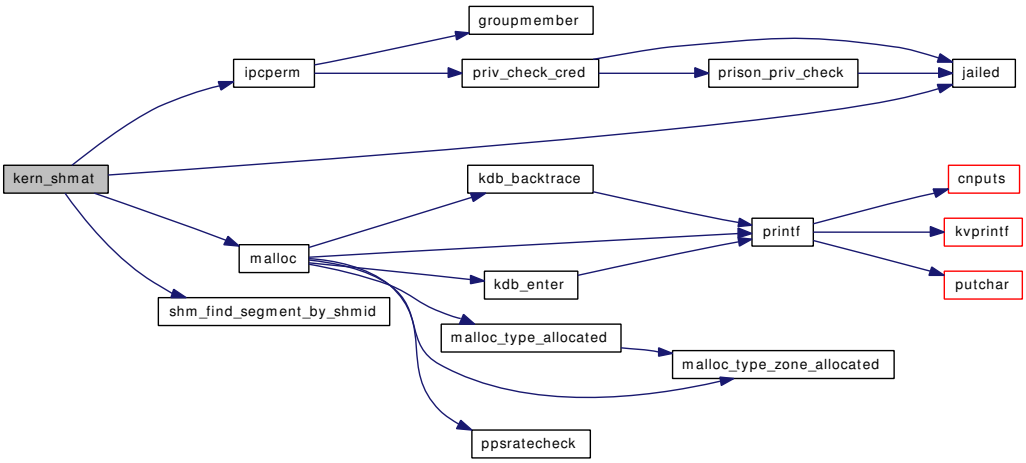
9.129.2.3 int kern_shmat (struct thread * *td*, int *shmid*, const void * *shmaddr*, int *shmflg*)

Definition at line 351 of file sysv_shm.c.

References Giant, ipcperm(), jail_sysvipc_allowed, jailed(), malloc(), shm_find_segment_by_shmid(), and shmmap_state::shmid.

Referenced by shmat().

Here is the call graph for this function:



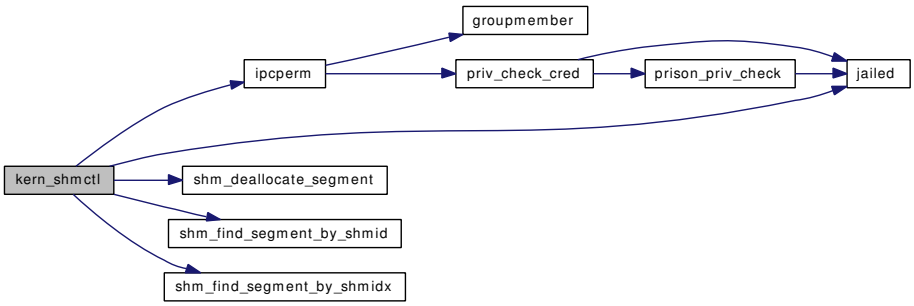
9.129.2.4 int kern_shmctl (struct thread * *td*, int *shmid*, int *cmd*, void * *buf*, size_t * *bufsz*)

Definition at line 549 of file sysv_shm.c.

References Giant, ipcperm(), jail_sysvipc_allowed, jailed(), shm_deallocate_segment(), shm_find_segment_by_shmid(), shm_find_segment_by_shmidx(), SHMSEG_REMOVED, and time_second.

Referenced by shmctl().

Here is the call graph for this function:



9.129.2.5 static MALLOC_DEFINE (M_SHM, "shm", "SVID compatible shared memory segments") [static]

9.129.2.6 MODULE_VERSION (sysvshm, 1)

9.129.2.7 static void shm_deallocate_segment (struct shmid_kernel *) [static]

Definition at line 244 of file sysv_shm.c.

References SHMSEG_FREE.

Referenced by kern_shmctl(), and shm_delete_mapping().

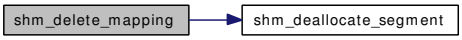
9.129.2.8 static int shm_delete_mapping (struct vmspace * *vm*, struct shmmap_state *) [static]

Definition at line 263 of file sysv_shm.c.

References shm_deallocate_segment(), shmmap_state::shmid, SHMSEG_REMOVED, time_second, and shmmap_state::va.

Referenced by shmexit_myhook().

Here is the call graph for this function:



9.129.2.9 static int shm_find_segment_by_key (key_t) [static]

Definition at line 198 of file sysv_shm.c.

References SHMSEG_ALLOCATED.

Referenced by shmget().

9.129.2.10 static struct shmid_kernel * shm_find_segment_by_shmid (int) [static]

Definition at line 211 of file sysv_shm.c.

References SHMSEG_ALLOCATED, and SHMSEG_REMOVED.

Referenced by kern_shmat(), and kern_shmctl().

9.129.2.11 static struct shmid_kernel * shm_find_segment_by_shmidx (int) [static]

Definition at line 229 of file sysv_shm.c.

References SHMSEG_ALLOCATED, and SHMSEG_REMOVED.

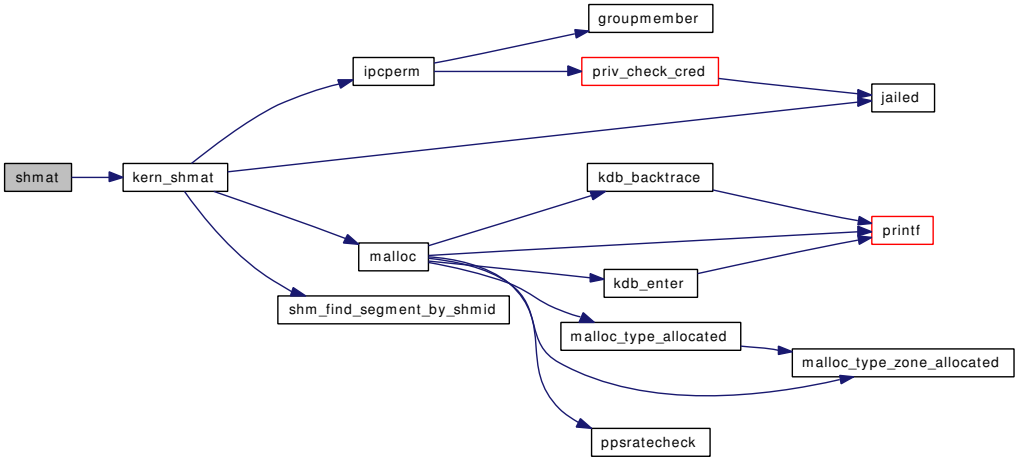
Referenced by kern_shmctl().

9.129.2.12 int shmat (struct thread * *td*, struct shmat_args * *uap*)

Definition at line 454 of file sysv_shm.c.

References kern_shmat(), shmat_args::shmaddr, shmat_args::shmflg, and shmat_args::shmid.

Here is the call graph for this function:

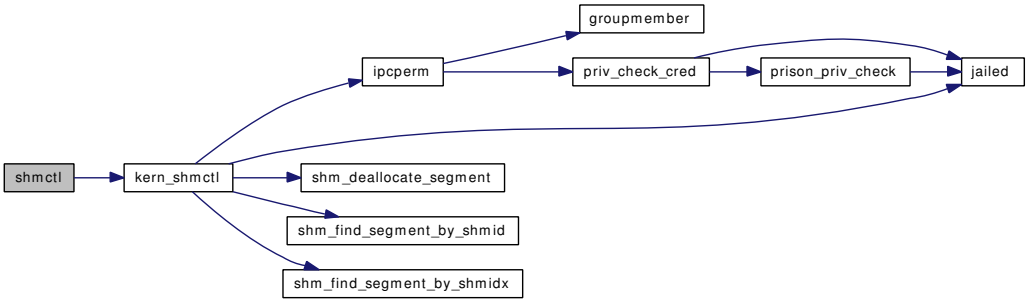


9.129.2.13 int shmctl (struct thread * *td*, struct shmctl_args * *uap*)

Definition at line 650 of file sysv_shm.c.

References shmctl_args::buf, shmctl_args::cmd, kern_shmctl(), and shmctl_args::shmid.

Here is the call graph for this function:



9.129.2.14 int shmdt (struct thread * *td*, struct shmdt_args * *uap*)

Definition at line 297 of file sysv_shm.c.

References Giant, jail_sysvipc_allowed, jailed(), shmdt_args::shmaddr, shmmap_state::shmid, and shmmap_state::va.

Here is the call graph for this function:

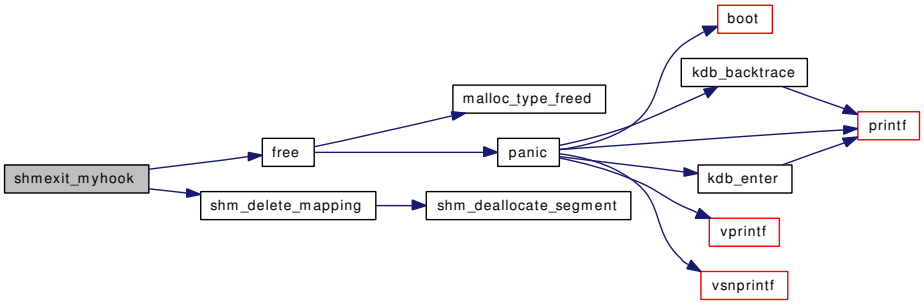
9.129.2.15 static void shmexit_myhook (struct vmspace * *vm*) [static]

Definition at line 902 of file sysv_shm.c.

References free(), Giant, shm_delete_mapping(), and shmmap_state::shmid.

Referenced by shminit().

Here is the call graph for this function:



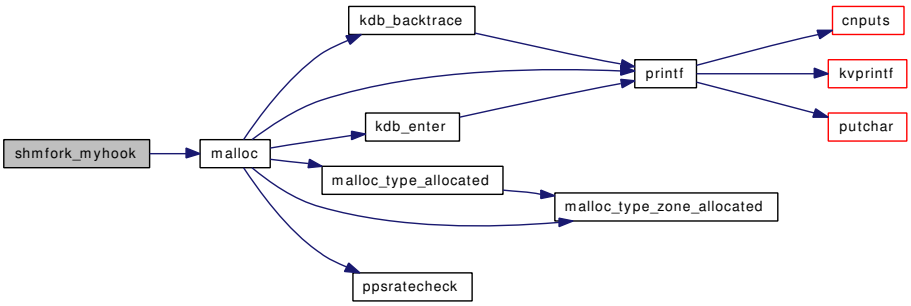
9.129.2.16 static void shmfork_myhook (struct proc * *p1*, struct proc * *p2*) [static]

Definition at line 883 of file sysv_shm.c.

References Giant, malloc(), and shmmap_state::shmid.

Referenced by shminit().

Here is the call graph for this function:

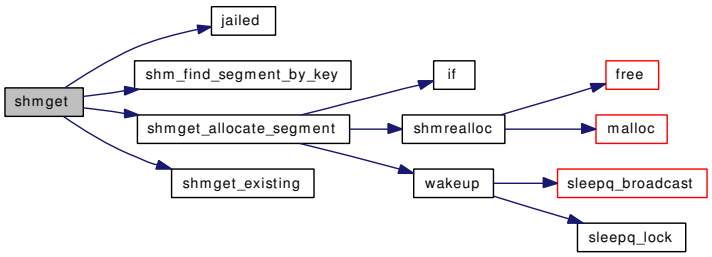


9.129.2.17 int shmget (struct thread * *td*, struct shmget_args * *uap*)

Definition at line 820 of file sysv_shm.c.

References Giant, jail_sysvipc_allowed, jailed(), shmget_args::key, shm_find_segment_by_key(), shmget_args::shmflg, shmget_allocate_segment(), and shmget_existing().

Here is the call graph for this function:



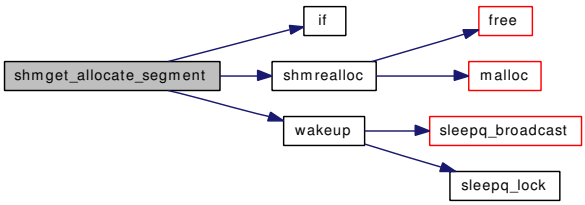
9.129.2.18 static int shmget_allocate_segment (struct thread * *td*, struct shmget_args * *uap*, int *mode*) [static]

Definition at line 732 of file sysv_shm.c.

References if(), shmget_args::key, shmrealloc(), SHMSEG_ALLOCATED, SHMSEG_FREE, SHMSEG_REMOVED, SHMSEG_WANTED, shmget_args::size, time_second, and wakeup().

Referenced by shmget().

Here is the call graph for this function:



9.129.2.19 static int shmget_existing (struct thread * *td*, struct shmget_args * *uap*, int *mode*, int *segnum*) [static]

Definition at line 696 of file sysv_shm.c.

References shmget_args::shmflg, SHMSEG_REMOVED, SHMSEG_WANTED, and shmget_args::size.

Referenced by shmget().

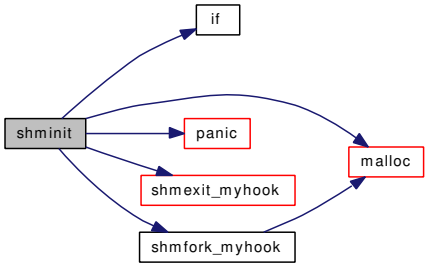
9.129.2.20 static void shminit (void) [static]

Definition at line 946 of file sysv_shm.c.

References if(), malloc(), panic(), shmexit_hook, shmexit_myhook(), shmfork_hook, shmfork_myhook(), and SHMSEG_FREE.

Referenced by sysvshm_modload().

Here is the call graph for this function:



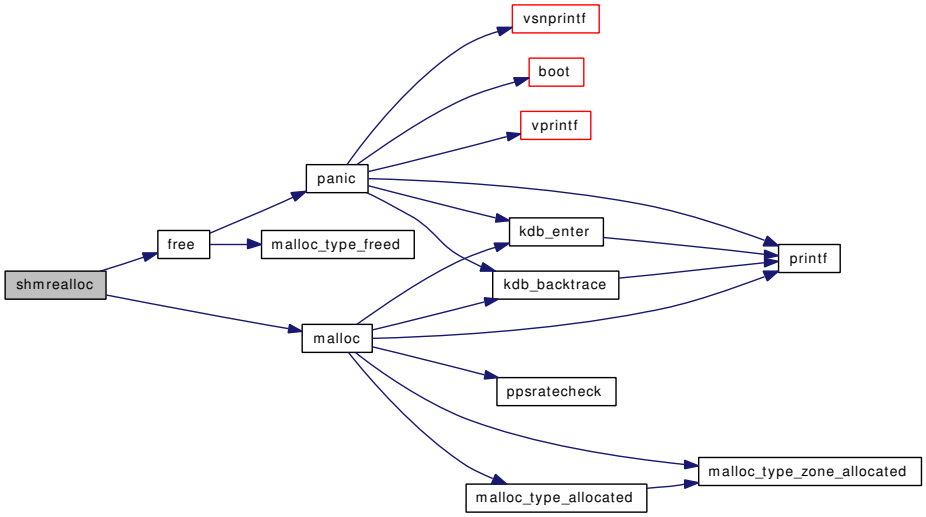
9.129.2.21 static void shmrealloc (void) [static]

Definition at line 920 of file sysv_shm.c.

References free(), malloc(), and SHMSEG_FREE.

Referenced by shmget_allocate_segment().

Here is the call graph for this function:

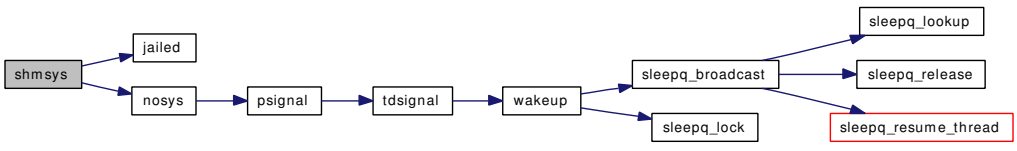


9.129.2.22 int shmsys (struct thread * *td*, struct shmsys_args * *uap*)

Definition at line 855 of file sysv_shm.c.

References Giant, jail_sysvipc_allowed, jailed(), and nosys().

Here is the call graph for this function:



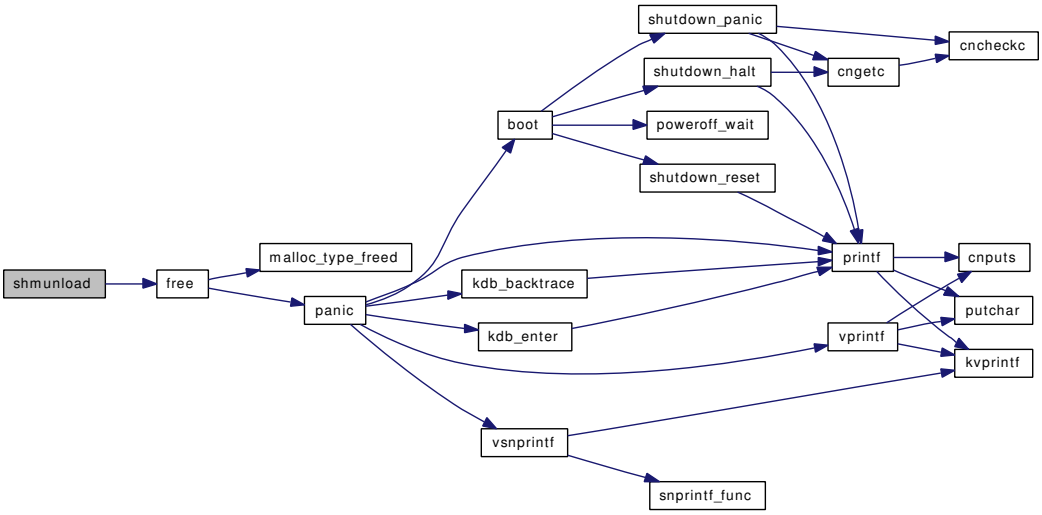
9.129.2.23 static int shmunload (void) [static]

Definition at line 980 of file sysv_shm.c.

References free(), shmexit_hook, and shmfork_hook.

Referenced by sysvshm_modload().

Here is the call graph for this function:



9.129.2.24 SYSCALL_MODULE_HELPER (shmget)

9.129.2.25 SYSCALL_MODULE_HELPER (shmdt)

9.129.2.26 SYSCALL_MODULE_HELPER (shmctl)

9.129.2.27 SYSCALL_MODULE_HELPER (shmat)

9.129.2.28 SYSCALL_MODULE_HELPER (shmsys)

9.129.2.29 SYSCTL_INT (_kern_ipc, OID_AUTO, shm_allow_removed, CTLFLAG_RW, & *shm_allow_removed*, 0, "Enable/Disable attachment to attached segments marked for removal")

9.129.2.30 SYSCTL_INT (_kern_ipc, OID_AUTO, shm_use_phys, CTLFLAG_RW, & *shm_use_phys*, 0, "Enable/Disable locking of shared memory pages in core")

9.129.2.31 SYSCTL_PROC (_kern_ipc, OID_AUTO, shmsegs, CTLFLAG_RD, NULL, 0, sysctl_shmsegs, "", "Current number of shared memory segments allocated")

9.129.2.32 static int sysctl_shmsegs (SYSCTL_HANDLER_ARGS) [static]

Definition at line 1000 of file sysv_shm.c.

9.129.2.33 SYSCTL_ULONG (_kern_ipc, OID_AUTO, shmall, CTLFLAG_RW, &shminfo. *shmall*, 0, "Maximum number of pages available for shared memory")

9.129.2.34 SYSCTL_ULONG (_kern_ipc, OID_AUTO, shmseg, CTLFLAG_RDTUN, &shminfo. *shmseg*, 0, "Number of segments per process")

9.129.2.35 SYSCTL_ULONG (_kern_ipc, OID_AUTO, shmmni, CTLFLAG_RDTUN, &shminfo. *shmmni*, 0, "Number of shared memory identifiers")

9.129.2.36 SYSCTL_ULONG (_kern_ipc, OID_AUTO, shmmin, CTLFLAG_RW, &shminfo. *shmmin*, 0, "Minimum shared memory segment size")

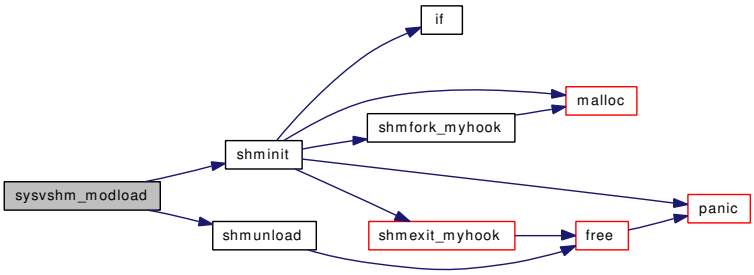
9.129.2.37 SYSCTL_ULONG (_kern_ipc, OID_AUTO, shmmax, CTLFLAG_RW, &shminfo. *shmmax*, 0, "Maximum shared memory segment size")

9.129.2.38 static int sysvshm_modload (struct module *, int, void *) [static]

Definition at line 1007 of file sysv_shm.c.

References shminit(), and shmunload().

Here is the call graph for this function:



9.129.3 Variable Documentation

9.129.3.1 int shm_allow_removed [static]

Definition at line 176 of file sysv_shm.c.

9.129.3.2 int shm_committed [static]

Definition at line 124 of file sysv_shm.c.

9.129.3.3 int shm_last_free [static]

Definition at line 124 of file sysv_shm.c.

9.129.3.4 int shm_nused [static]

Definition at line 124 of file sysv_shm.c.

9.129.3.5 int shm_use_phys [static]

Definition at line 175 of file sysv_shm.c.

9.129.3.6 int shmalloced [static]

Definition at line 124 of file sysv_shm.c.

9.129.3.7 struct shminfo shminfo

Initial value:

```
{
        SHMMAX,
        SHMMIN,
        SHMMNI,
        SHMSEG,
        SHMALL
}
```

Definition at line 167 of file sysv_shm.c.

9.129.3.8 struct shmid_kernel* shmsegs [static]

Definition at line 125 of file sysv_shm.c.

9.129.3.9 moduledata_t sysvshm_mod [static]

Initial value:

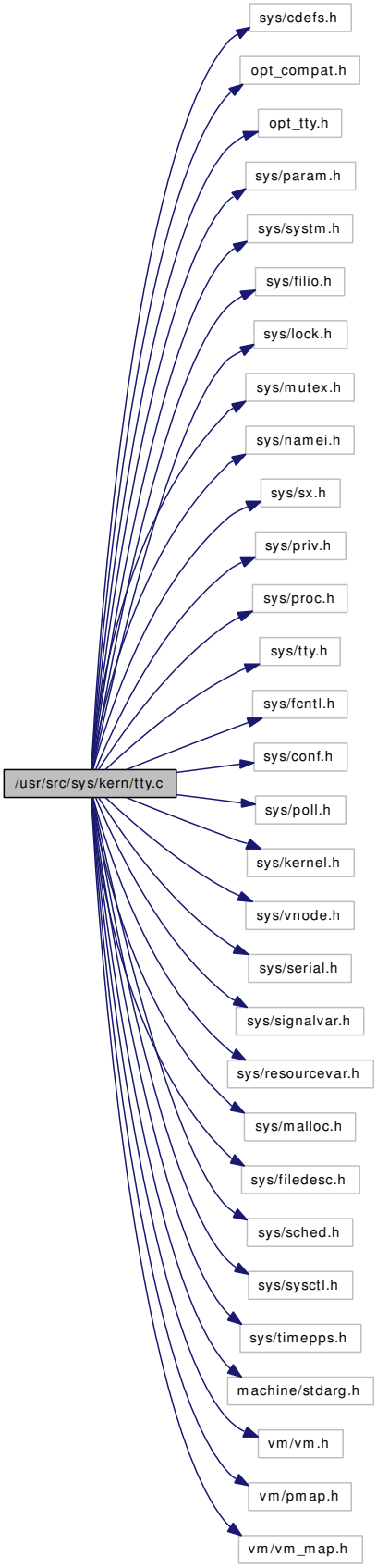
```
{
        "sysvshm",
        &sysvshm_modload,
        NULL
}
```

Definition at line 1027 of file sysv_shm.c.

9.130 /usr/src/sys/kern/tty.c File Reference

```
#include <sys/cdefs.h>
#include "opt_compat.h"
#include "opt_tty.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/filio.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/namei.h>
#include <sys/sx.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/tty.h>
#include <sys/fcntl.h>
#include <sys/conf.h>
#include <sys/poll.h>
#include <sys/kernel.h>
#include <sys/vnode.h>
#include <sys/serial.h>
#include <sys/signalvar.h>
#include <sys/resourcevar.h>
#include <sys/malloc.h>
#include <sys/filedesc.h>
#include <sys/sched.h>
#include <sys/sysctl.h>
#include <sys/timepps.h>
#include <machine/stdarg.h>
#include <vm/vm.h>
#include <vm/pmap.h>
#include <vm/vm_map.h>
```

Include dependency graph for tty.c:


/usr/src/sys/kern/tty.c
sys/cdefs.h
opt_compat.h
opt_tty.h
sys/param.h
sys/systm.h
sys/filio.h
sys/lock.h
sys/mutex.h
sys/namei.h
sys/sx.h
sys/priv.h
sys/proc.h
sys/tty.h
sys/fcntl.h
sys/conf.h
sys/poll.h
sys/kernel.h
sys/vnode.h
sys/serial.h
sys/signalvar.h
sys/resourcevar.h
sys/malloc.h
sys/filedesc.h
sys/sched.h
sys/sysctl.h
sys/timepps.h
machine/stdarg.h
vm/vm.h
vm/pmap.h
vm/vm_map.h

Defines

- #define TTYDEFCHARS
- #define E 0x00
- #define O 0x80
- #define PARITY(c) (char_type[c] & O)
- #define ALPHA 0x40
- #define ISALPHA(c) (char_type[(c) & TTY_CHARMASK] & ALPHA)
- #define CCLASSMASK 0x3f
- #define CCLASS(c) (char_type[c] & CCLASSMASK)
- #define BS BACKSPACE
- #define CC CONTROL
- #define CR RETURN
- #define NA ORDINARY | ALPHA
- #define NL NEWLINE
- #define NO ORDINARY
- #define TB TAB
- #define VT VTAB
- #define SET(t, f) (t) |= (f)
- #define CLR(t, f) (t) &= ~(f)
- #define ISSET(t, f) ((t) & (f))
- #define MAX_INPUT TTYHOG
- #define FLUSHQ(q)
- #define TTBREAKC(c, lflag)
- #define diff(t1, t2)
- #define PANICSTR "ttyrub: would panic c = %d, val = %d\n"
- #define CLAMP(x, h, l) ((x) > h ? h : ((x) < l) ? l : (x))
- #define ISRUN(p, val)
- #define TESTAB(a, b) ((a)<<1 | (b))
- #define ONLYA 2
- #define ONLYB 1
- #define BOTH 3
- #define XT_COPY(field) xt.xt_##field = tp → t_##field

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/tty.c,v 1.268 2006/12/20 02:49:59 mbr Exp $")
- MALLOC_DEFINE (M_TTYS,"ttys","tty data structures")
- static int proc_compare (struct proc *p1, struct proc *p2)
- static int ttnread (struct tty *tp)
- static void ttyecho (int c, struct tty *tp)
- static int ttyoutput (int c, struct tty *tp)
- static void ttypend (struct tty *tp)
- static void ttyretype (struct tty *tp)
- static void ttyrub (int c, struct tty *tp)
- static void ttyrubo (struct tty *tp, int cnt)
- static void ttyunblock (struct tty *tp)
- static int ttywflush (struct tty *tp)
- static int filt_ttyread (struct knote *kn, long hint)
- static void filt_ttyrdetach (struct knote *kn)

- static int filt_ttywrite (struct knote *kn, long hint)
- static void filt_ttywdetach (struct knote *kn)
- static TAILQ_HEAD (tty)
- int tty_open (struct cdev *device, struct tty *tp)
- int tty_close (struct tty *tp)
- int ttyinput (int c, struct tty *tp)
- int ttioctl (struct tty *tp, u_long cmd, void *data, int flag)
- int ttypoll (struct cdev *dev, int events, struct thread *td)
- int ttykqfilter (struct cdev *dev, struct knote *kn)
- int ttywait (struct tty *tp)
- void ttyflush (struct tty *tp, int rw)
- void termioschars (struct termios *t)
- void ttychars (struct tty *tp)
- void ttyblock (struct tty *tp)
- int ttstart (struct tty *tp)
- int ttylclose (struct tty *tp, int flag)
- int ttymodem (struct tty *tp, int flag)
- int ttread (struct tty *tp, struct uio *uio, int flag)
- int ttycheckoutq (struct tty *tp, int wait)
- int ttwrite (struct tty *tp, struct uio *uio, int flag)
- void ttwakeup (struct tty *tp)
- void ttwwakeup (struct tty *tp)
- int ttspeedtab (int speed, struct speedtab *table)
- void ttsetwater (struct tty *tp)
- void ttyinfo (struct tty *tp)
- int tputchar (int c, struct tty *tp)
- int ttysleep (struct tty *tp, void *chan, int pri, char *wmesg, int timo)
- int ttyref (struct tty *tp)
- int ttyrel (struct tty *tp)
- tty * ttyalloc ()
- static void ttypurge (struct cdev *dev)
- int ttycreate (struct tty *tp, int flags, const char *fmt,...)
- void ttygone (struct tty *tp)
- void ttyfree (struct tty *tp)
- static int sysctl_kern_ttys (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern, OID_AUTO, ttys, CTLTYPE_OPAQUE|CTLFLAG_RD, 0, 0, sysctl_kern_ttys,"S,xtty","All ttys")
- SYSCTL_LONG (_kern, OID_AUTO, tty_nin, CTLFLAG_RD,&tk_nin, 0,"Total TTY in characters")
- SYSCTL_LONG (_kern, OID_AUTO, tty_nout, CTLFLAG_RD,&tk_nout, 0,"Total TTY out characters")
- void nottystop (struct tty *tp, int rw)
- int ttyopen (struct cdev *dev, int flag, int mode, struct thread *td)
- int ttyclose (struct cdev *dev, int flag, int mode, struct thread *td)
- int ttyread (struct cdev *dev, struct uio *uio, int flag)
- int ttywrite (struct cdev *dev, struct uio *uio, int flag)
- int ttyioctl (struct cdev *dev, u_long cmd, caddr_t data, int flag, struct thread *td)
- void ttyldoptim (struct tty *tp)
- static void ttydtrwaitwakeup (void *arg)
- void ttydtrwaitstart (struct tty *tp)

- int ttydtrwaitsleep (struct tty *tp)
- static int ttysopen (struct cdev *dev, int flag, int mode, struct thread *td)
- static int ttysclose (struct cdev *dev, int flag, int mode, struct thread *td)
- static int ttysrdwr (struct cdev *dev, struct uio *uio, int flag)
- static int ttysioctl (struct cdev *dev, u_long cmd, caddr_t data, int flag, struct thread *td)
- void ttyinitmode (struct tty *tp, int echo, int speed)
- void ttyconsolemode (struct tty *tp, int speed)
- CTASSERT (SER_DTR==TIOCM_DTR/2)
- CTASSERT (SER_RTS==TIOCM_RTS/2)
- CTASSERT (SER_STX==TIOCM_ST/2)
- CTASSERT (SER_SRX==TIOCM_SR/2)
- CTASSERT (SER_CTS==TIOCM_CTS/2)
- CTASSERT (SER_DCD==TIOCM_DCD/2)
- CTASSERT (SER_RI==TIOCM_RI/2)
- CTASSERT (SER_DSR==TIOCM_DSR/2)

Variables

- long tk_cancc
- long tk_nin
- long tk_nout
- long tk_rawcc
- static d_open_t ttysopen
- static d_close_t ttysclose
- static d_read_t ttysrdwr
- static d_ioctl_t ttysioctl
- static d_purge_t ttypurge
- static struct cdevsw tty_cdevsw
- static struct cdevsw ttys_cdevsw
- static u_char const char_type [ ]
- static struct filterops ttyread_filtops
- static struct filterops ttywrite_filtops

9.130.1 Define Documentation

9.130.1.1 #define ALPHA 0x40

Definition at line 176 of file tty.c.

9.130.1.2 #define BOTH 3

Definition at line 2655 of file tty.c.

Referenced by proc_compare().

9.130.1.3 #define BS BACKSPACE

Definition at line 182 of file tty.c.

9.130.1.4 #define CC CONTROL

Definition at line 183 of file tty.c.

9.130.1.5 #define CCLASS(c) (char_type[c] & CCLASSMASK)

Definition at line 180 of file tty.c.

Referenced by ttyoutput(), and ttyrub().

9.130.1.6 #define CCLASSMASK 0x3f

Definition at line 179 of file tty.c.

Referenced by ttwrite().

9.130.1.7 #define CLAMP(x, h, l) ((x) > h ? h : ((x) < l) ? l : (x))

Referenced by ttsetwater().

9.130.1.8 #define CLR(t, f) (t) &= ~(f)

Definition at line 240 of file tty.c.

Referenced by ttioctl(), ttwwakeup(), ttyblock(), ttyecho(), ttyflush(), ttyinput(), ttymodem(), ttyoutput(), ttypend(), ttyretype(), ttyrub(), and ttyunblock().

9.130.1.9 #define CR RETURN

Definition at line 184 of file tty.c.

9.130.1.10 #define diff(t1, t2)

Value:

```
(((t1).tv_sec - (t2).tv_sec) * 1000000 + \
                        ((t1).tv_usec - (t2).tv_usec))
```

Referenced by link_elf_search_symbol(), and ttread().

9.130.1.11 #define E 0x00

Definition at line 172 of file tty.c.

9.130.1.12 #define FLUSHQ(q)

Value:

```
{                                                                       \
        if ((q)->c_cc)                                                  \
                ndflush(q, (q)->c_cc);                                  \
}
```

Definition at line 349 of file tty.c.

Referenced by ttyflush(), and ttyinput().

9.130.1.13 #define ISALPHA(c) (char_type[(c) & TTY_CHARMASK] & ALPHA)

Definition at line 177 of file tty.c.

Referenced by ttyinput().

9.130.1.14 #define ISRUN(p, val)

Value:

```
do {                                                                    \
        struct thread *td;                                              \
        val = 0;                                                        \
        FOREACH_THREAD_IN_PROC(p, td) {                                 \
                if (TD_ON_RUNQ(td) ||                                   \
                    TD_IS_RUNNING(td)) {                                \
                        val = 1;                                        \
                        break;                                          \
                }                                                       \
        }                                                               \
} while (0)
```

Definition at line 2639 of file tty.c.

Referenced by proc_compare().

9.130.1.15 #define ISSET(t, f) ((t) & (f))

Definition at line 241 of file tty.c.

Referenced by filt_ttyread(), filt_ttywrite(), tputchar(), ttioctl(), ttnread(), ttread(), ttwakeup(), ttwrite(), ttwwakeup(), tty_close(), tty_open(), ttyblock(), ttyecho(), ttyflush(), ttyinput(), ttymodem(), ttyoutput(), ttypoll(), ttyrub(), ttyunblock(), and ttywait().

9.130.1.16 #define MAX_INPUT TTYHOG

Definition at line 244 of file tty.c.

Referenced by ttyinput(), and vop_stdpathconf().

9.130.1.17 #define NA ORDINARY | ALPHA

Definition at line 185 of file tty.c.

9.130.1.18 #define NL NEWLINE

Definition at line 186 of file tty.c.

9.130.1.19 #define NO ORDINARY

Definition at line 187 of file tty.c.

9.130.1.20 #define O 0x80

Definition at line 173 of file tty.c.

9.130.1.21 #define ONLYA 2

Definition at line 2653 of file tty.c.

Referenced by proc_compare().

9.130.1.22 #define ONLYB 1

Definition at line 2654 of file tty.c.

Referenced by proc_compare().

9.130.1.23 #define PANICSTR "ttyrub: would panic c = %d, val = %d\n"

Referenced by ttyrub().

9.130.1.24 #define PARITY(c) (char_type[c] & O)

Definition at line 174 of file tty.c.

9.130.1.25 #define SET(t, f) (t) |= (f)

Definition at line 239 of file tty.c.

Referenced by ttioctl(), ttwrite(), tty_open(), ttyblock(), ttycheckoutq(), ttyflush(), ttyinput(), ttymodem(), ttypend(), ttyrub(), ttyunblock(), and ttywait().

9.130.1.26 #define TB TAB

Definition at line 188 of file tty.c.

9.130.1.27 #define TESTAB(a, b) ((a)<<1 | (b))

Definition at line 2652 of file tty.c.

Referenced by proc_compare().

9.130.1.28 #define TTBREAKC(c, lflag)

Value:

```
((c) == '\n' || (((c) == cc[VEOF] ||                              \
          (c) == cc[VEOL] || ((c) == cc[VEOL2] && lflag & IEXTEN)) &&   \
         (c) != _POSIX_VDISABLE))
```

Definition at line 355 of file tty.c.

Referenced by ttread(), and ttyinput().

9.130.1.29 #define TTYDEFCHARS

Definition at line 91 of file tty.c.

9.130.1.30 #define VT VTAB

Definition at line 189 of file tty.c.

9.130.1.31 #define XT_COPY(field) xt.xt_##field = tp → t_##field

Referenced by sysctl_kern_ttys().

9.130.2 Function Documentation

9.130.2.1 __FBSDID ("$FreeBSD: src/sys/kern/tty. *c*, v 1.268 2006/12/20 02:49:59 mbr Exp $")

9.130.2.2 CTASSERT (SER_DSR = =TIOCM_DSR/2)

9.130.2.3 CTASSERT (SER_RI = =TIOCM_RI/2)

9.130.2.4 CTASSERT (SER_DCD = =TIOCM_DCD/2)

9.130.2.5 CTASSERT (SER_CTS = =TIOCM_CTS/2)

9.130.2.6 CTASSERT (SER_SRX = =TIOCM_SR/2)

9.130.2.7 CTASSERT (SER_STX = =TIOCM_ST/2)

9.130.2.8 CTASSERT (SER_RTS = =TIOCM_RTS/2)

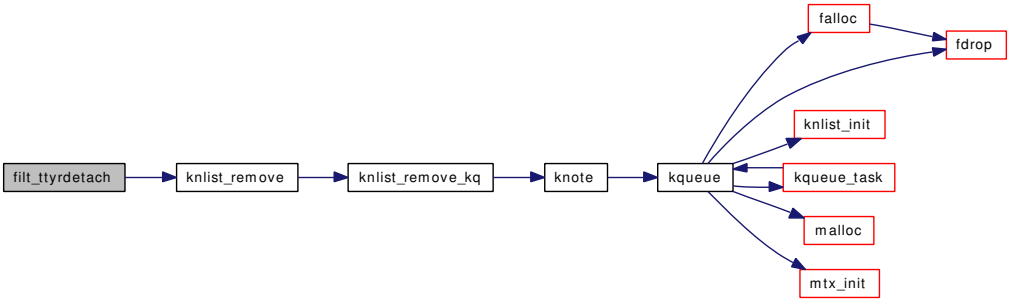
9.130.2.9 CTASSERT (SER_DTR = =TIOCM_DTR/2)

9.130.2.10 static void filt_ttyrdetach (struct knote * *kn*) [static]

Definition at line 1344 of file tty.c.

References knlist_remove().

Here is the call graph for this function:

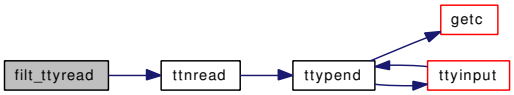


9.130.2.11 static int filt_ttyread (struct knote * *kn*, long *hint*) [static]

Definition at line 1354 of file tty.c.

References ISSET, and ttnread().

Here is the call graph for this function:

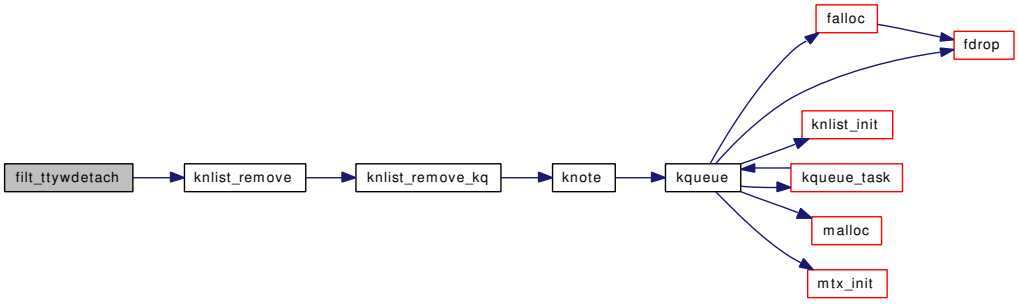


9.130.2.12 static void filt_ttywdetach (struct knote * *kn*) [static]

Definition at line 1367 of file tty.c.

References knlist_remove().

Here is the call graph for this function:



9.130.2.13 static int filt_ttywrite (struct knote * *kn*, long *hint*) [static]

Definition at line 1377 of file tty.c.

References ISSET.

9.130.2.14 MALLOC_DEFINE (M_TTYS, "ttys", "tty data structures")

9.130.2.15 void nottystop (struct tty * *tp*, int *rw*)

Definition at line 3052 of file tty.c.

9.130.2.16 static int proc_compare (struct proc * *p1*, struct proc * *p2*) [static]

Definition at line 2658 of file tty.c.

References BOTH, ISRUN, ONLYA, ONLYB, sched_lock, td, and TESTAB.

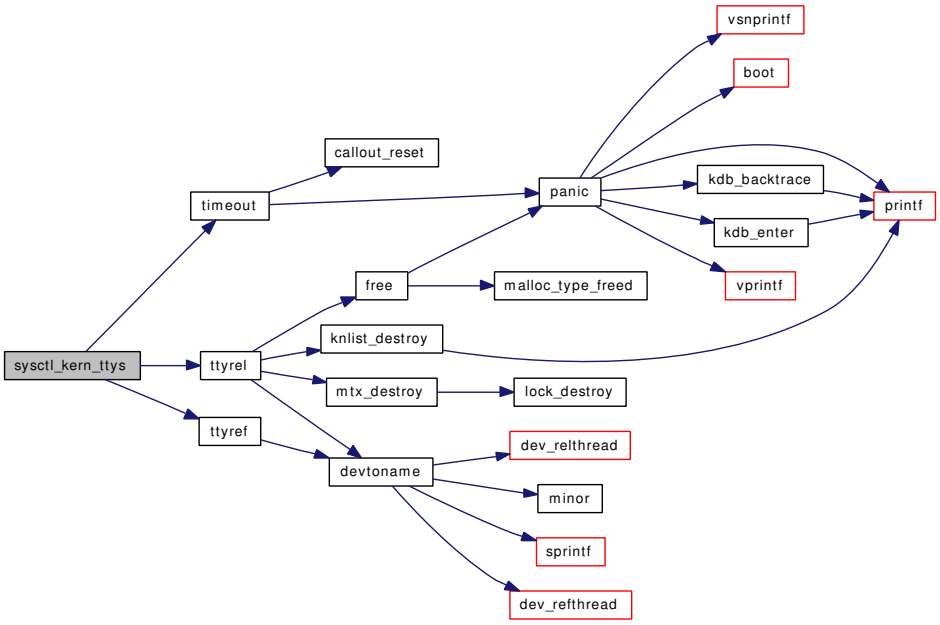
Referenced by ttyinfo().

9.130.2.17 static int sysctl_kern_ttys (SYSCTL_HANDLER_ARGS) [static]

Definition at line 2986 of file tty.c.

References timeout(), ttyref(), ttyrel(), and XT_COPY.

Here is the call graph for this function:



9.130.2.18 SYSCTL_LONG (_kern, OID_AUTO, tty_nout, CTLFLAG_RD, & *tk_nout*, 0, "Total TTY out characters")

9.130.2.19 SYSCTL_LONG (_kern, OID_AUTO, tty_nin, CTLFLAG_RD, & *tk_nin*, 0, "Total TTY in characters")

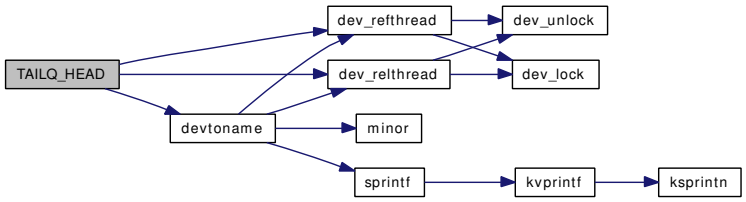
9.130.2.20 SYSCTL_PROC (_kern, OID_AUTO, ttys, CTLTYPE_OPAQUE| *CTLFLAG_RD*, 0, 0, sysctl_kern_ttys, " *S*, xtty", "All ttys")

9.130.2.21 static TAILQ_HEAD (tty) [static]

Definition at line 253 of file tty.c.

References dev_refthread(), dev_relthread(), and devtoname().

Here is the call graph for this function:



9.130.2.22 void termioschars (struct termios * *t*)

Definition at line 1526 of file tty.c.

Referenced by ttychars(), and ttyinitmode().

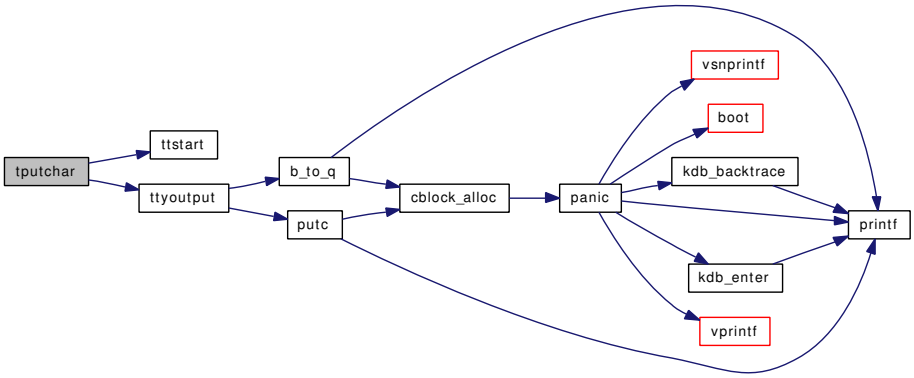
9.130.2.23 int tputchar (int *c*, struct tty * *tp*)

Definition at line 2728 of file tty.c.

References ISSET, ttstart(), and ttyoutput().

Referenced by constty_timeout(), and putchar().

Here is the call graph for this function:



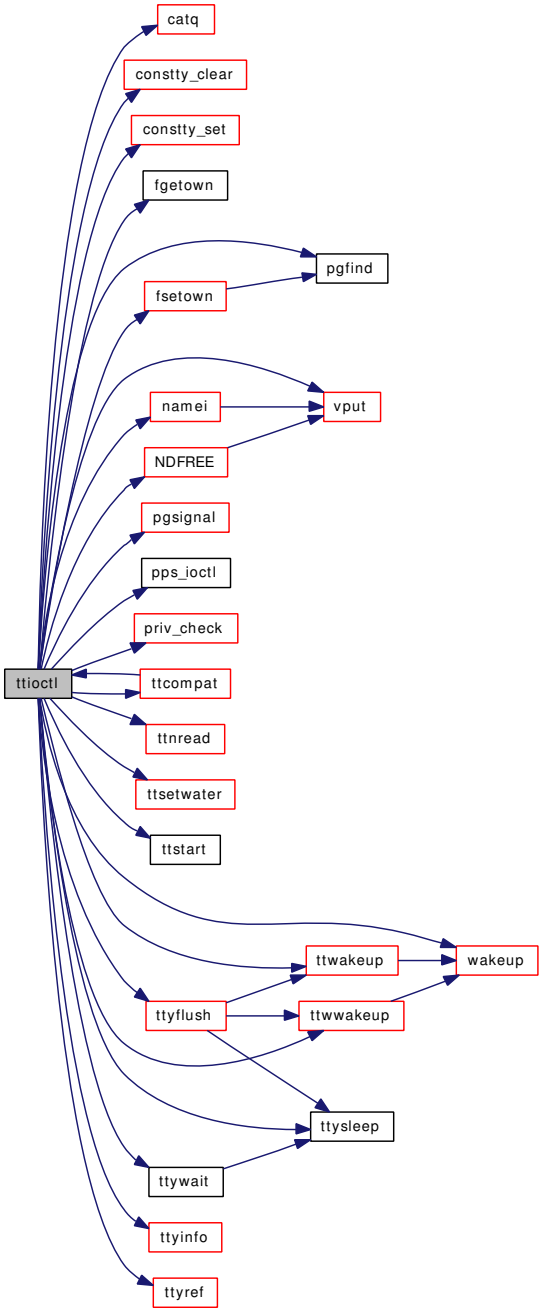
9.130.2.24 int ttioctl (struct tty * *tp*, u_long *cmd*, void * *data*, int *flag*)

Definition at line 803 of file tty.c.

References catq(), CLR, constty_clear(), constty_set(), fgetown(), fsetown(), hz, ISSET, lbolt, namei(), NDFREE(), nlinesw, pgfind(), pgsignal(), pps_ioctl(), priv_check(), proctree_lock, SET, td, ttcompat(), ttnread(), ttsetwater(), ttstart(), ttwakeup(), ttwwakeup(), ttyflush(), ttyinfo(), ttyref(), ttysleep(), ttywait(), vput(), and wakeup().

Referenced by ptsioctl(), ttcompat(), and ttyioctl().

Here is the call graph for this function:



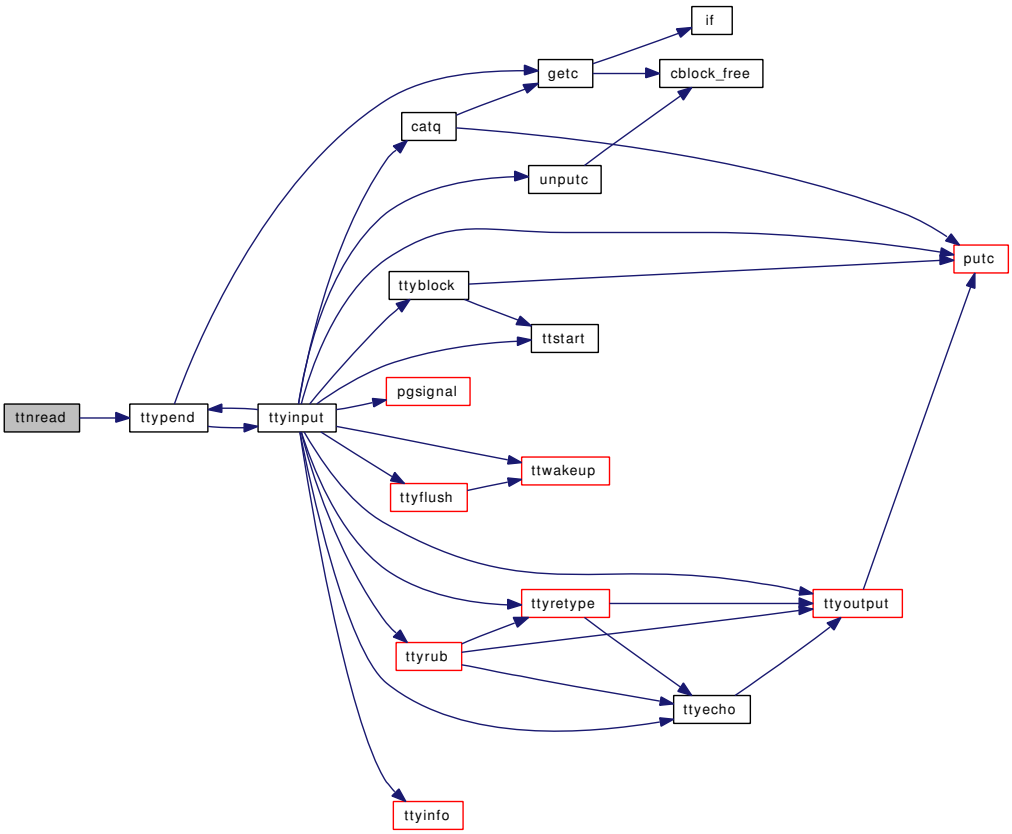
9.130.2.25 static int ttnread (struct tty * *tp*) [static]

Definition at line 1392 of file tty.c.

References ISSET, and ttypend().

Referenced by filt_ttyread(), ttioctl(), and ttypoll().

Here is the call graph for this function:

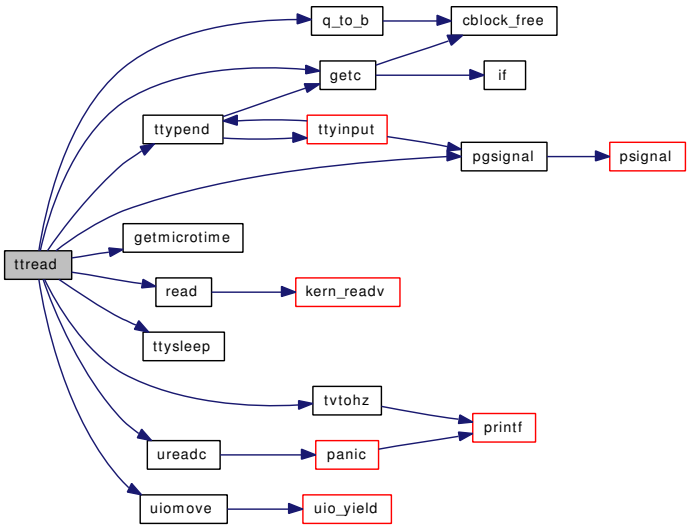


9.130.2.26 int ttread (struct tty * *tp*, struct uio * *uio*, int *flag*)

Definition at line 1710 of file tty.c.

References diff, getc(), getmicrotime(), ISSET, lbolt, pgsignal(), proctree_lock, q_to_b(), read(), td, TTBREAKC, ttypend(), ttysleep(), tvtohz(), uiomove(), and ureadc().

Here is the call graph for this function:



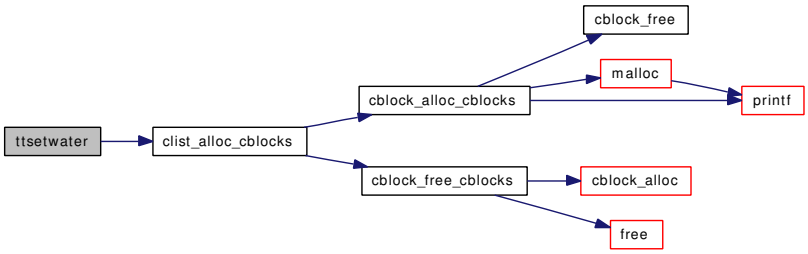
9.130.2.27 void ttsetwater (struct tty * *tp*)

Definition at line 2469 of file tty.c.

References CLAMP, and clist_alloc_cblocks().

Referenced by ttioctl(), tty_open(), and ttyconsolemode().

Here is the call graph for this function:



9.130.2.28 int ttspeedtab (int *speed*, struct speedtab * *table*)

Definition at line 2451 of file tty.c.

9.130.2.29 int ttstart (struct tty * *tp*)

Definition at line 1598 of file tty.c.

Referenced by tputchar(), ttioctl(), ttwrite(), ttyblock(), ttycheckoutq(), ttyinput(), ttymodem(), and ttyunblock().

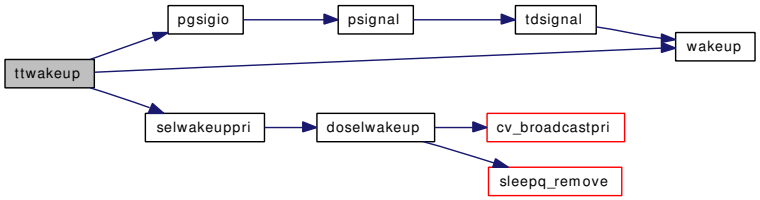
9.130.2.30 void ttwakeup (struct tty * *tp*)

Definition at line 2411 of file tty.c.

References ISSET, pgsigio(), selwakeuppri(), and wakeup().

Referenced by ttioctl(), ttyflush(), ttyinput(), and ttymodem().

Here is the call graph for this function:

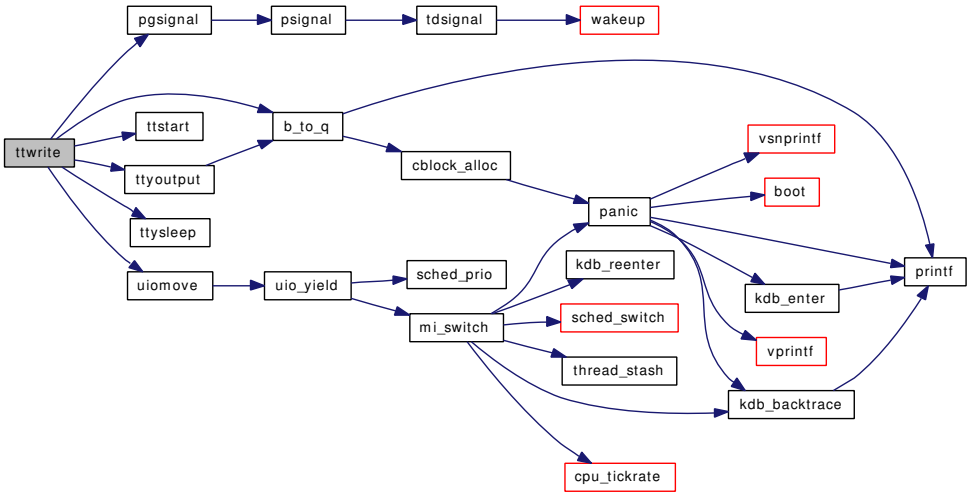


9.130.2.31 int ttwrite (struct tty * *tp*, struct uio * *uio*, int *flag*)

Definition at line 2036 of file tty.c.

References b_to_q(), CCLASSMASK, char_type, ISSET, lbolt, pgsignal(), proctree_lock, SET, td, tk_nout, ttstart(), ttyoutput(), ttysleep(), and uiomove().

Here is the call graph for this function:



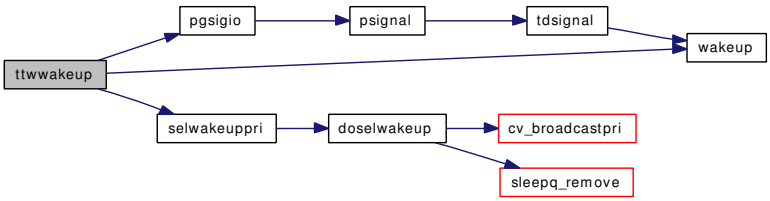
9.130.2.32 void ttwwakeup (struct tty * *tp*)

Definition at line 2426 of file tty.c.

References CLR, ISSET, pgsigio(), selwakeuppri(), and wakeup().

Referenced by ttioctl(), ttyflush(), and ttymodem().

Here is the call graph for this function:



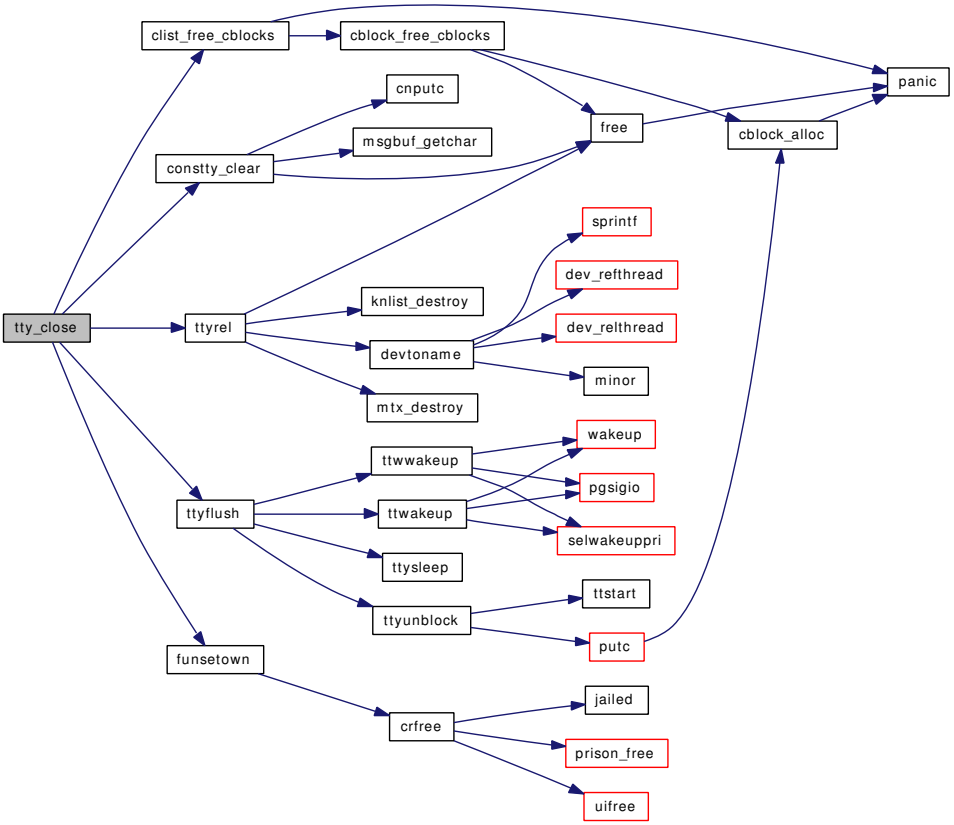
9.130.2.33 int tty_close (struct tty * *tp*)

Definition at line 315 of file tty.c.

References clist_free_cblocks(), constty_clear(), funsetown(), ISSET, ttyflush(), and ttyrel().

Referenced by ptsclose(), and ttyclose().

Here is the call graph for this function:

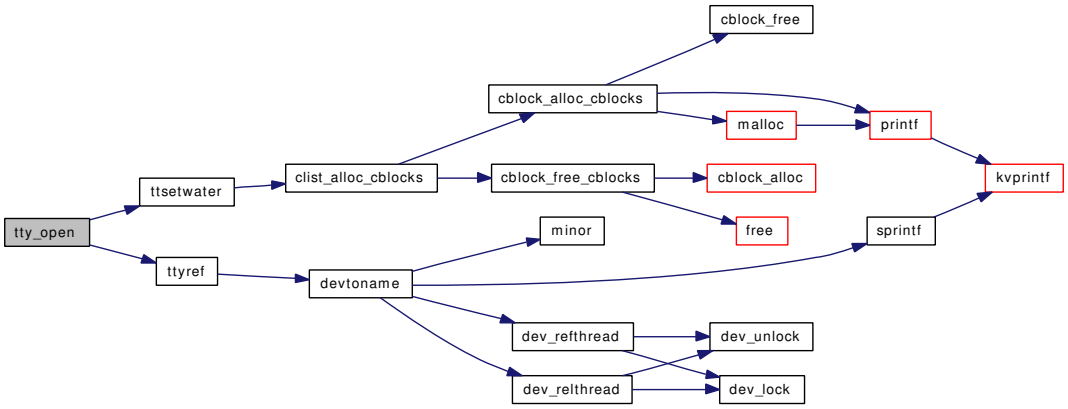


9.130.2.34 int tty_open (struct cdev * *device*, struct tty * *tp*)

Definition at line 284 of file tty.c.

References hz, ISSET, SET, ttsetwater(), and ttyref().

Here is the call graph for this function:



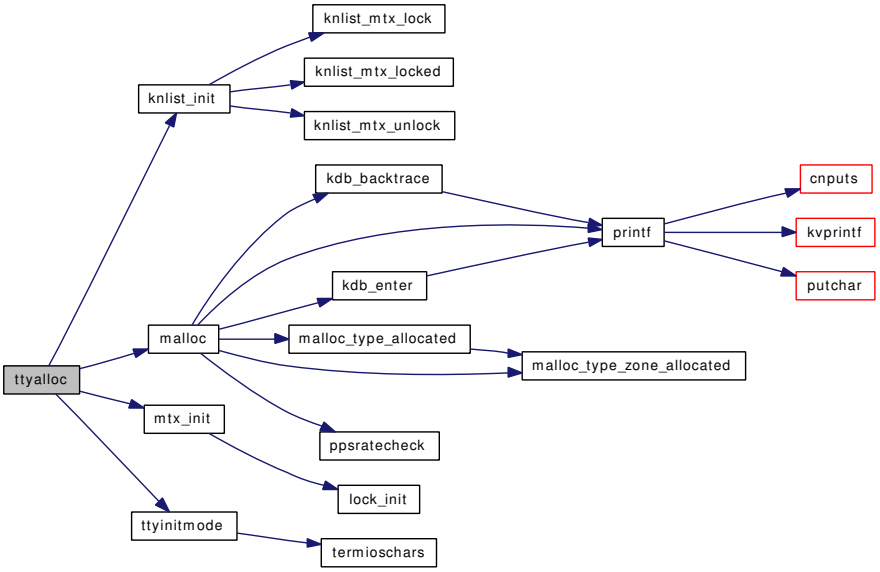
9.130.2.35 struct tty* ttyalloc ()

Definition at line 2818 of file tty.c.

References hz, knlist_init(), malloc(), mtx_init(), and ttyinitmode().

Referenced by ptcopen(), and ptyinit().

Here is the call graph for this function:



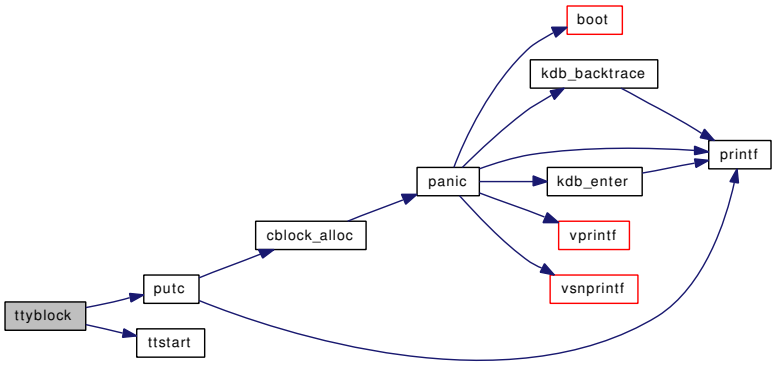
9.130.2.36 void ttyblock (struct tty * *tp*)

Definition at line 1548 of file tty.c.

References CLR, ISSET, putc(), SET, and ttstart().

Referenced by ttyinput().

Here is the call graph for this function:

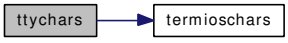


9.130.2.37 void ttychars (struct tty * *tp*)

Definition at line 1536 of file tty.c.

References termioschars().

Here is the call graph for this function:



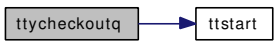
9.130.2.38 int ttycheckoutq (struct tty * *tp*, int *wait*)

Definition at line 1992 of file tty.c.

References hz, SET, td, and ttstart().

Referenced by tprintf(), and ttyinfo().

Here is the call graph for this function:

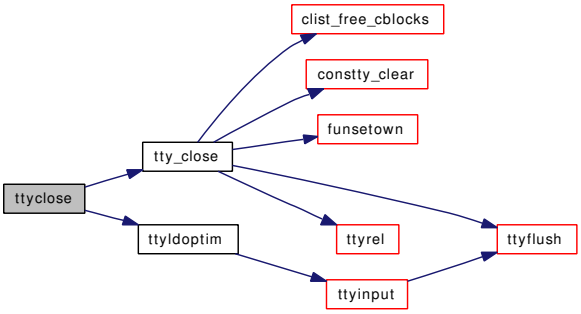


9.130.2.39 int ttyclose (struct cdev * *dev*, int *flag*, int *mode*, struct thread * *td*)

Definition at line 3142 of file tty.c.

References tty_close(), and ttyldoptim().

Here is the call graph for this function:

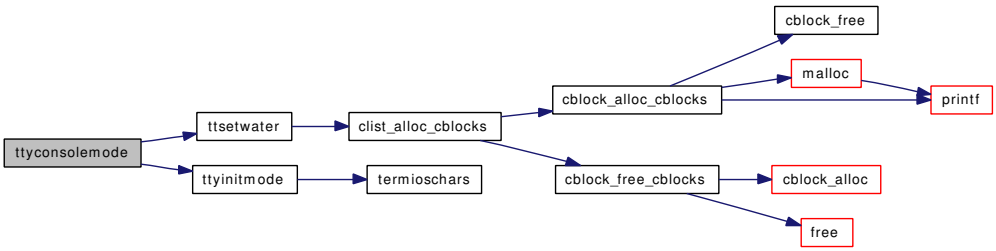


9.130.2.40 void ttyconsolemode (struct tty * *tp*, int *speed*)

Definition at line 3369 of file tty.c.

References ttsetwater(), and ttyinitmode().

Here is the call graph for this function:

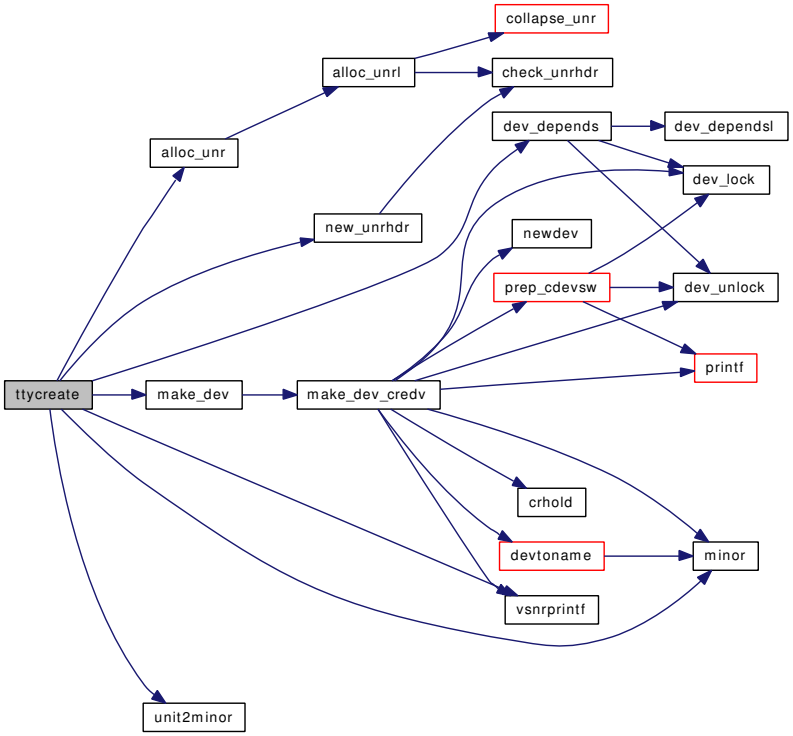


9.130.2.41 int ttycreate (struct tty * *tp*, int *flags*, const char * *fmt*, ...)

Definition at line 2869 of file tty.c.

References alloc_unr(), dev_depends(), Giant, make_dev(), minor(), new_unrhdr(), tty_cdevsw, ttypurge, ttys_cdevsw, unit2minor(), and vsnrprintf().

Here is the call graph for this function:



9.130.2.42 int ttydtrwaitsleep (struct tty * *tp*)

Definition at line 3262 of file tty.c.

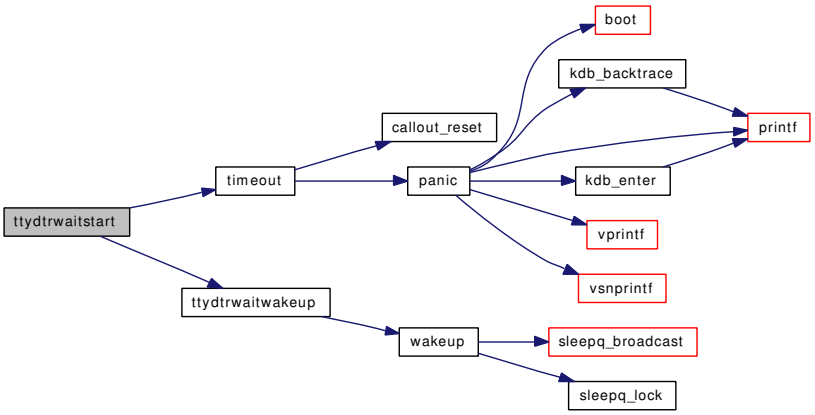
Referenced by ttyopen().

9.130.2.43 void ttydtrwaitstart (struct tty * *tp*)

Definition at line 3250 of file tty.c.

References timeout(), and ttydtrwaitwakeup().

Here is the call graph for this function:



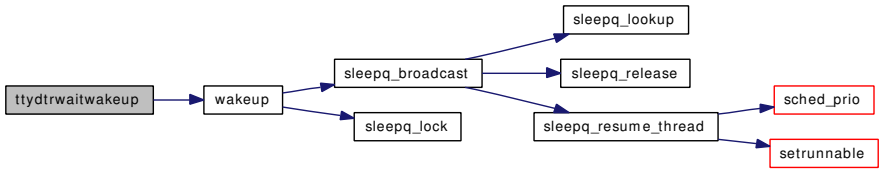
9.130.2.44 static void ttydtrwaitwakeup (void * *arg*) [static]

Definition at line 3239 of file tty.c.

References wakeup().

Referenced by ttydtrwaitstart().

Here is the call graph for this function:



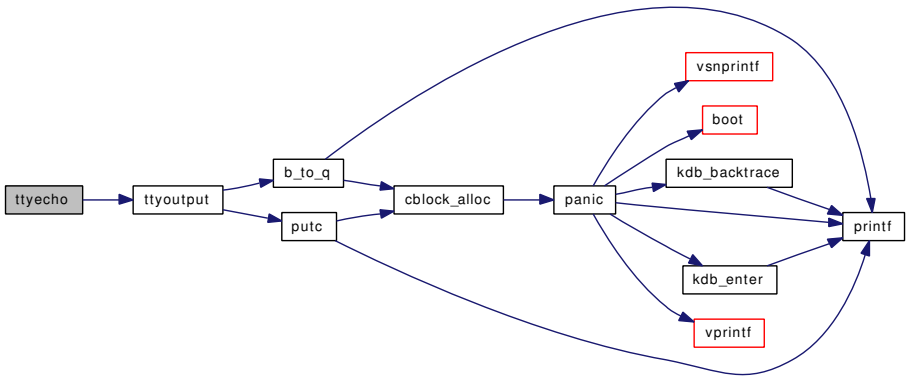
9.130.2.45 static void ttyecho (int *c*, struct tty * *tp*) [static]

Definition at line 2385 of file tty.c.

References CLR, ISSET, and ttyoutput().

Referenced by ttyinput(), ttyretype(), and ttyrub().

Here is the call graph for this function:



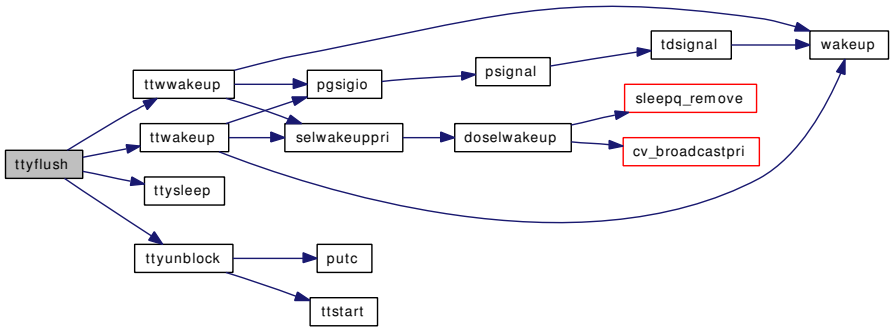
9.130.2.46 void ttyflush (struct tty * *tp*, int *rw*)

Definition at line 1457 of file tty.c.

References CLR, FLUSHQ, hz, ISSET, SET, ttwakeup(), ttwwakeup(), ttysleep(), and ttyunblock().

Referenced by ptcclose(), ptcioctl(), ttioctl(), tty_close(), ttyinput(), ttylclose(), ttymodem(), and ttywflush().

Here is the call graph for this function:

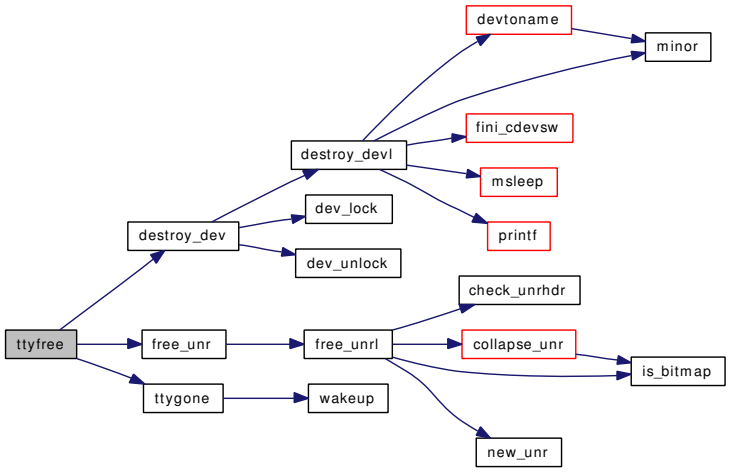


9.130.2.47 void ttyfree (struct tty * *tp*)

Definition at line 2974 of file tty.c.

References destroy_dev(), free_unr(), Giant, and ttygone().

Here is the call graph for this function:



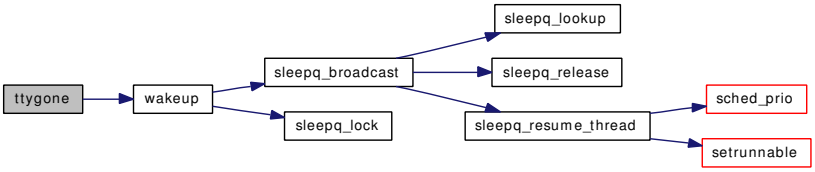
9.130.2.48 void ttygone (struct tty * *tp*)

Definition at line 2953 of file tty.c.

References wakeup().

Referenced by ttyfree(), and ttypurge().

Here is the call graph for this function:



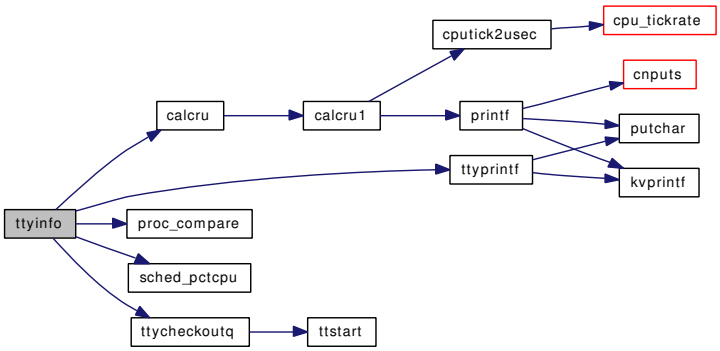
9.130.2.49 void ttyinfo (struct tty * *tp*)

Definition at line 2527 of file tty.c.

References averunnable, calcru(), proc_compare(), sched_lock, sched_pctcpu(), td, ttycheckoutq(), and ttyprintf().

Referenced by ptcioctl(), ttioctl(), and ttyinput().

Here is the call graph for this function:



9.130.2.50 void ttyinitmode (struct tty * *tp*, int *echo*, int *speed*)

Definition at line 3346 of file tty.c.

References termioschars().

Referenced by ptsopen(), ttyalloc(), and ttyconsolemode().

Here is the call graph for this function:

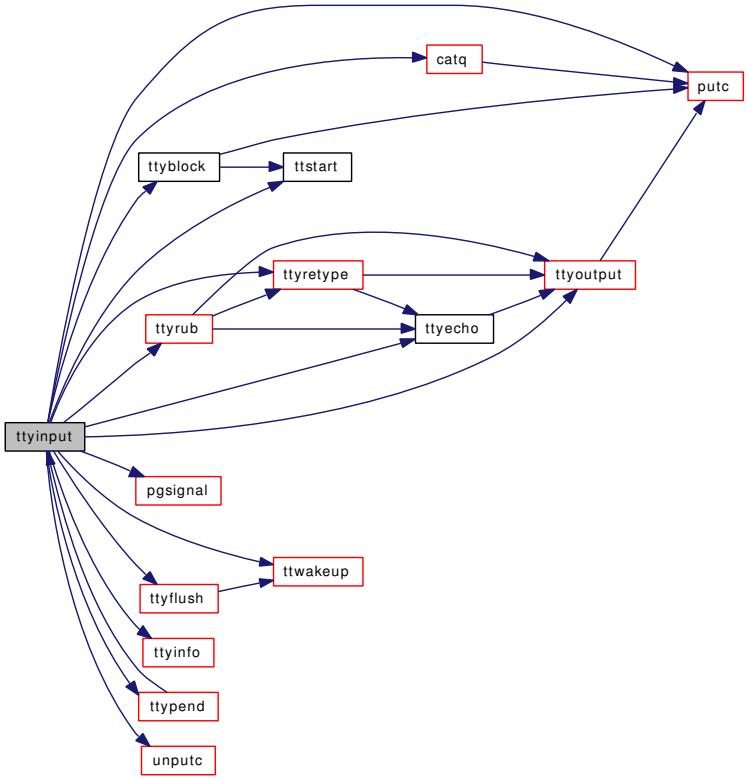
9.130.2.51 int ttyinput (int *c*, struct tty * *tp*)

Definition at line 364 of file tty.c.

References catq(), CLR, FLUSHQ, ISALPHA, ISSET, MAX_INPUT, pgsignal(), putc(), SET, tk_cancc, tk_nin, tk_rawcc, TTBREAKC, ttstart(), ttwakeup(), ttyblock(), ttyecho(), ttyflush(), ttyinfo(), ttyoutput(), ttypend(), ttyretype(), ttyrub(), and unputc().

Referenced by ttyldoptim(), and ttypend().

Here is the call graph for this function:



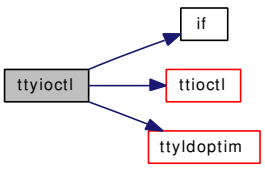
9.130.2.52 int ttyioctl (struct cdev * *dev*, u_long *cmd*, caddr_t *data*, int *flag*, struct thread * *td*)

Definition at line 3182 of file tty.c.

References if(), ttioctl(), and ttyldoptim().

Referenced by prep_cdevsw(), and ptsioctl().

Here is the call graph for this function:



9.130.2.53 int ttykqfilter (struct cdev * *dev*, struct knote * *kn*)

Definition at line 1313 of file tty.c.

References knlist_add(), ttyread_filtops, and ttywrite_filtops.

Referenced by prep_cdevsw().

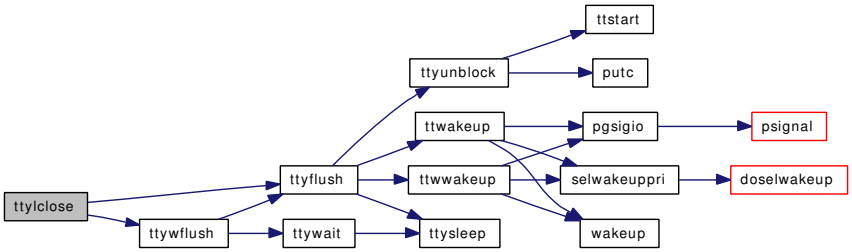
Here is the call graph for this function:

9.130.2.54 int ttylclose (struct tty * *tp*, int *flag*)

Definition at line 1609 of file tty.c.

References ttyflush(), and ttywflush().

Here is the call graph for this function:



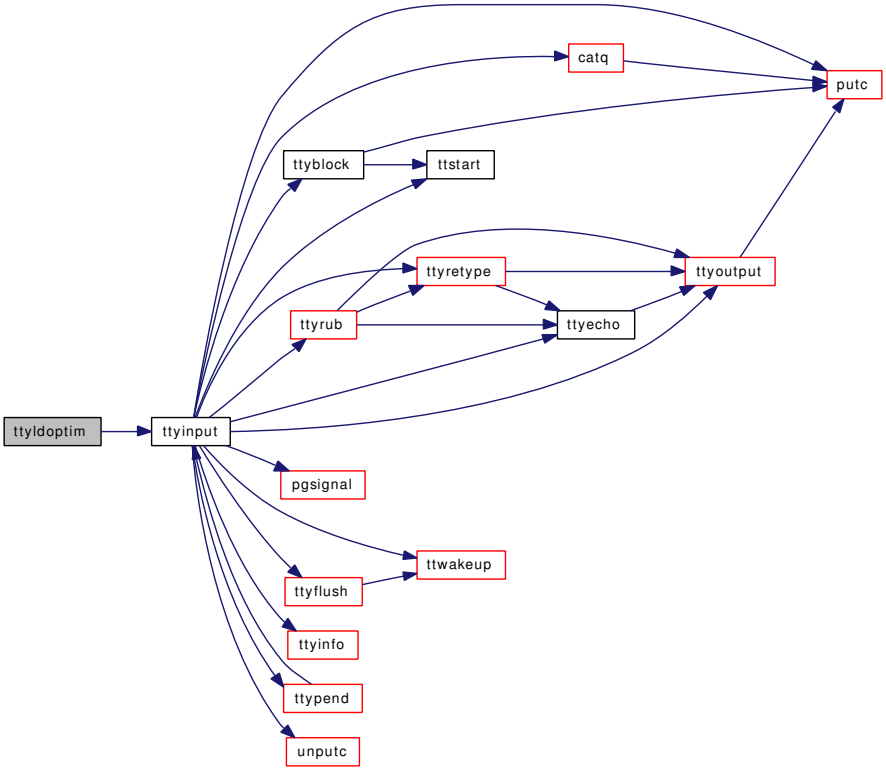
9.130.2.55 void ttyldoptim (struct tty * *tp*)

Definition at line 3222 of file tty.c.

References linesw, and ttyinput().

Referenced by ttyclose(), ttyioctl(), and ttyopen().

Here is the call graph for this function:

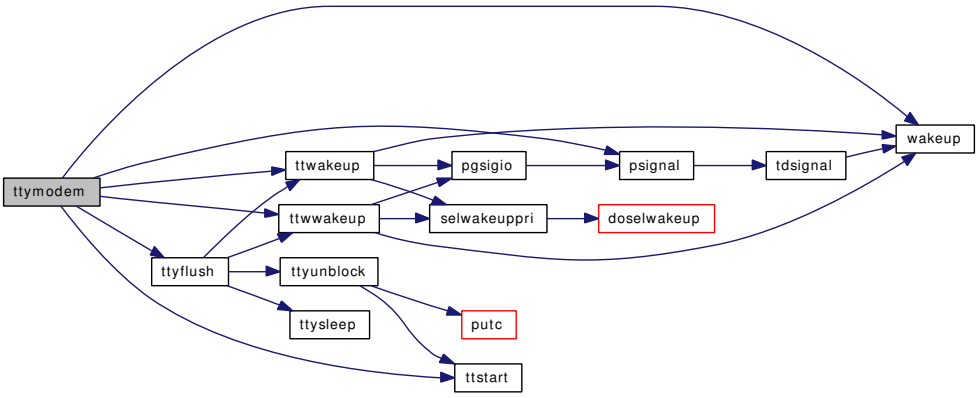


9.130.2.56 int ttymodem (struct tty * *tp*, int *flag*)

Definition at line 1623 of file tty.c.

References CLR, ISSET, proctree_lock, psignal(), SET, ttstart(), ttwakeup(), ttwwakeup(), ttyflush(), and wakeup().

Here is the call graph for this function:

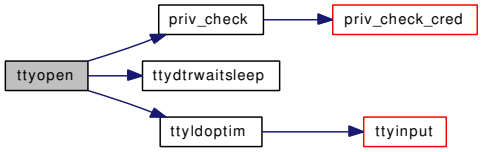


9.130.2.57 int ttyopen (struct cdev * *dev*, int *flag*, int *mode*, struct thread * *td*)

Definition at line 3059 of file tty.c.

References priv_check(), ttydtrwaitsleep(), and ttyldoptim().

Here is the call graph for this function:



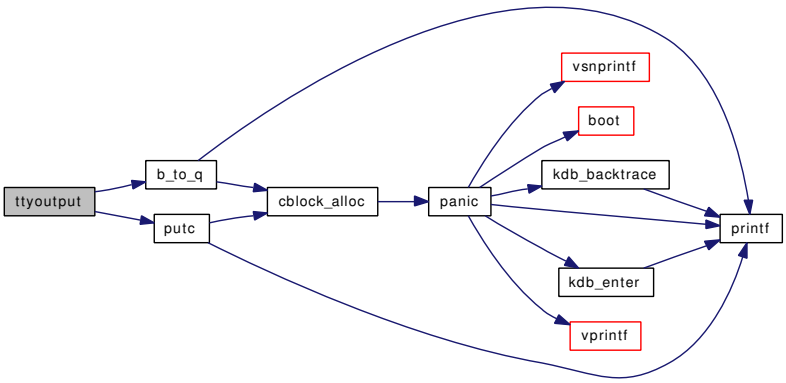
9.130.2.58 static int ttyoutput (int *c*, struct tty * *tp*) [static]

Definition at line 710 of file tty.c.

References b_to_q(), CCLASS, CLR, ISSET, putc(), and tk_nout.

Referenced by tputchar(), ttwrite(), ttyecho(), ttyinput(), ttyretype(), ttyrub(), and ttyrubo().

Here is the call graph for this function:



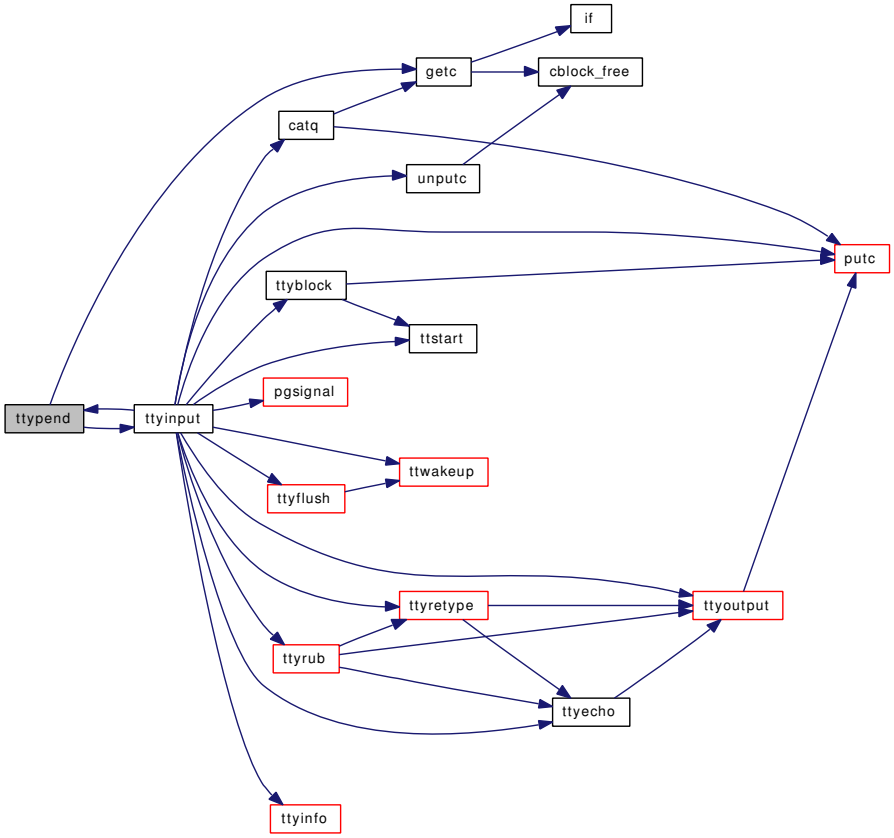
9.130.2.59 static void ttypend (struct tty * *tp*) [static]

Definition at line 1684 of file tty.c.

References CLR, getc(), SET, and ttyinput().

Referenced by ttnread(), ttread(), and ttyinput().

Here is the call graph for this function:



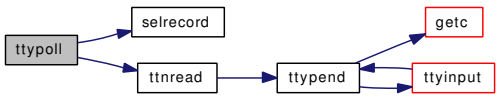
9.130.2.60 int ttypoll (struct cdev * *dev*, int *events*, struct thread * *td*)

Definition at line 1272 of file tty.c.

References ISSET, selrecord(), and ttnread().

Referenced by prep_cdevsw().

Here is the call graph for this function:

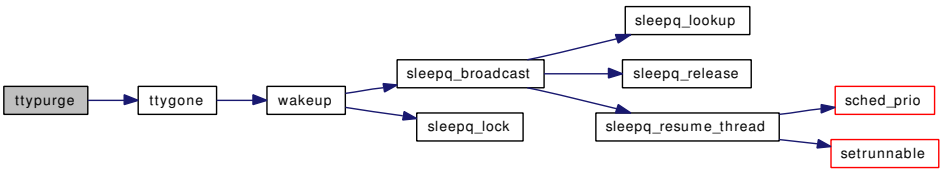


9.130.2.61 static void ttypurge (struct cdev * *dev*) [static]

Definition at line 2847 of file tty.c.

References ttygone().

Here is the call graph for this function:



9.130.2.62 int ttyread (struct cdev * *dev*, struct uio * *uio*, int *flag*)

Definition at line 3158 of file tty.c.

Referenced by prep_cdevsw().

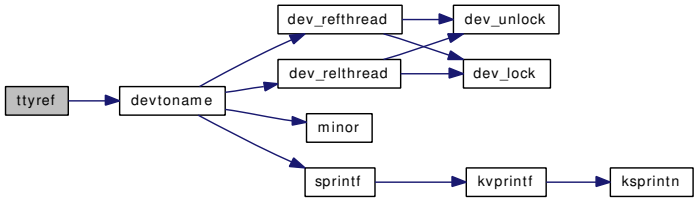
9.130.2.63 int ttyref (struct tty * *tp*)

Definition at line 2770 of file tty.c.

References devtoname().

Referenced by sysctl_kern_ttys(), ttioctl(), and tty_open().

Here is the call graph for this function:



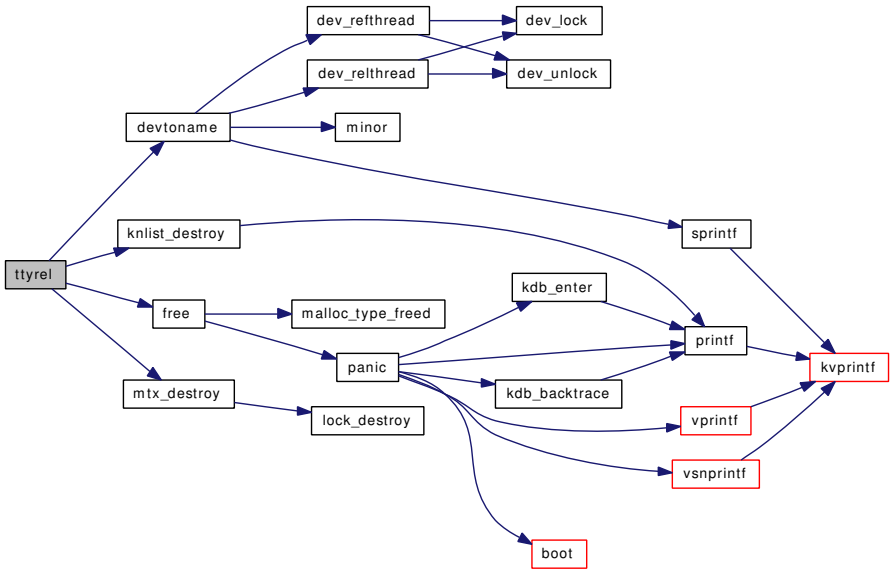
9.130.2.64 int ttyrel (struct tty * *tp*)

Definition at line 2788 of file tty.c.

References devtoname(), free(), knlist_destroy(), and mtx_destroy().

Referenced by pty_destroy_slave(), pty_maybecleanup(), sessrele(), sysctl_kern_ttys(), and tty_close().

Here is the call graph for this function:



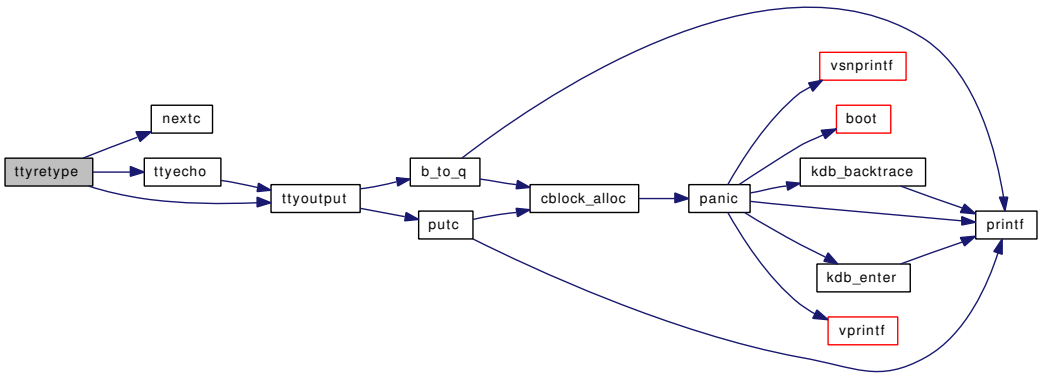
9.130.2.65 static void ttyretype (struct tty * *tp*) [static]

Definition at line 2351 of file tty.c.

References CLR, nextc(), ttyecho(), and ttyoutput().

Referenced by ttyinput(), and ttyrub().

Here is the call graph for this function:



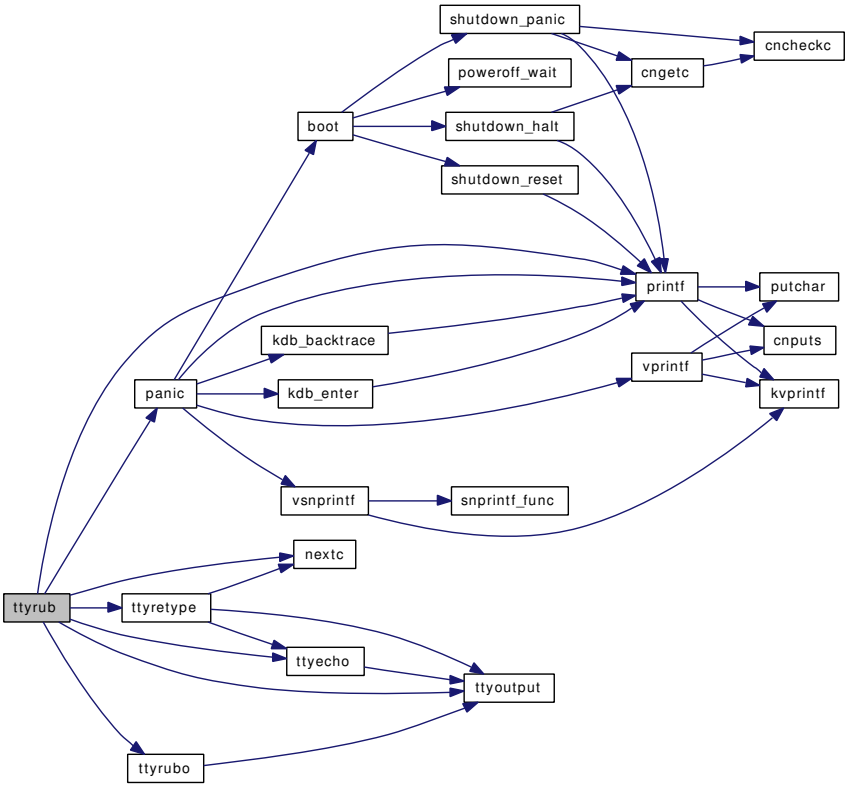
9.130.2.66 static void ttyrub (int *c*, struct tty * *tp*) [static]

Definition at line 2243 of file tty.c.

References CCLASS, CLR, ISSET, nextc(), panic(), PANICSTR, printf(), SET, ttyecho(), ttyoutput(), ttyretype(), and ttyrubo().

Referenced by ttyinput().

Here is the call graph for this function:



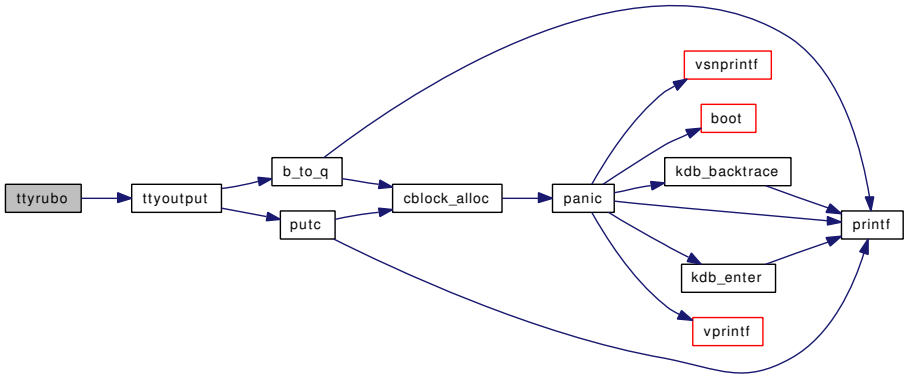
9.130.2.67 static void ttyrubo (struct tty * *tp*, int *cnt*) [static]

Definition at line 2335 of file tty.c.

References ttyoutput().

Referenced by ttyrub().

Here is the call graph for this function:



9.130.2.68 static int ttysclose (struct cdev * *dev*, int *flag*, int *mode*, struct thread * *td*) [static]

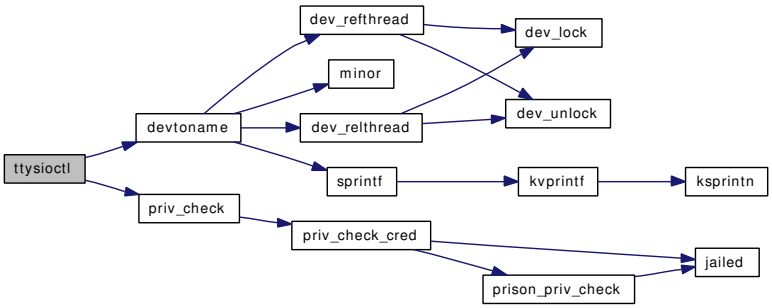
Definition at line 3293 of file tty.c.

9.130.2.69 static int ttysioctl (struct cdev * *dev*, u_long *cmd*, caddr_t *data*, int *flag*, struct thread * *td*) [static]

Definition at line 3307 of file tty.c.

References devtoname(), and priv_check().

Here is the call graph for this function:



9.130.2.70 int ttysleep (struct tty * *tp*, void * *chan*, int *pri*, char * *wmesg*, int *timo*)

Definition at line 2752 of file tty.c.

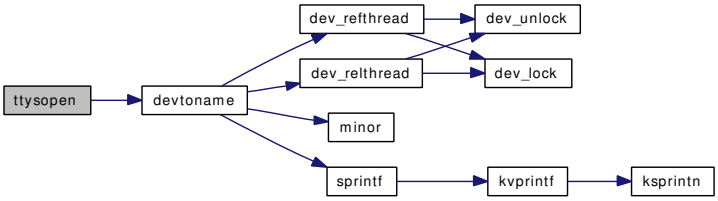
Referenced by ptsopen(), ttioctl(), tread(), ttwrite(), ttyflush(), and ttywait().

9.130.2.71 static int ttysopen (struct cdev * *dev*, int *flag*, int *mode*, struct thread * *td*) [static]

Definition at line 3280 of file tty.c.

References devtoname().

Here is the call graph for this function:



9.130.2.72 static int ttysrdwr (struct cdev * *dev*, struct uio * *uio*, int *flag*) [static]

Definition at line 3300 of file tty.c.

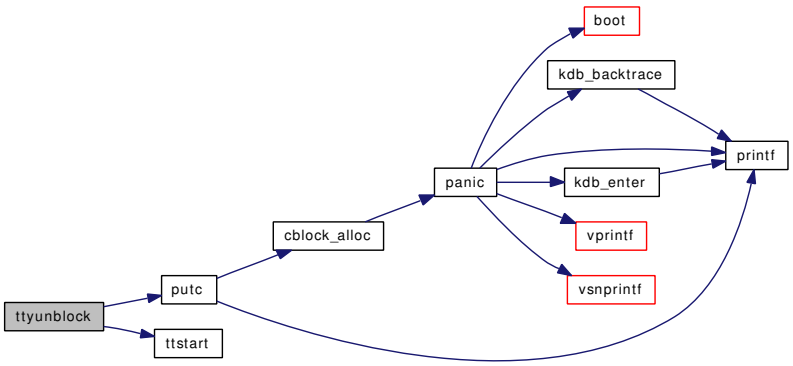
9.130.2.73 static void ttyunblock (struct tty * *tp*) [static]

Definition at line 1564 of file tty.c.

References CLR, ISSET, putc(), SET, and ttstart().

Referenced by ttyflush().

Here is the call graph for this function:



9.130.2.74 int ttywait (struct tty * *tp*)

Definition at line 1411 of file tty.c.

References ISSET, SET, and ttysleep().

Referenced by exit1(), ttioctl(), and ttywflush().

Here is the call graph for this function:

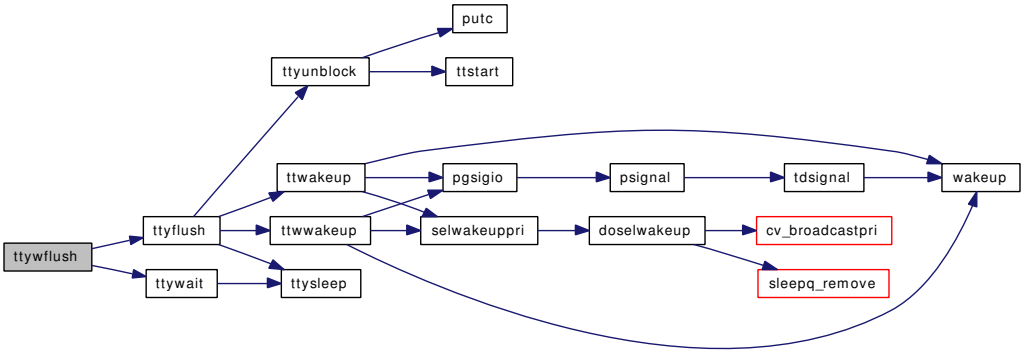
9.130.2.75 static int ttywflush (struct tty * *tp*) [static]

Definition at line 1444 of file tty.c.

References ttyflush(), and ttywait().

Referenced by ttylclose().

Here is the call graph for this function:



9.130.2.76 int ttywrite (struct cdev * *dev*, struct uio * *uio*, int *flag*)

Definition at line 3170 of file tty.c.

Referenced by prep_cdevsw().

9.130.3 Variable Documentation

9.130.3.1 u_char const char_type[ ] [static]

Definition at line 191 of file tty.c.

Referenced by ttwrite().

9.130.3.2 long tk_cancc

Definition at line 116 of file tty.c.

Referenced by ttyinput().

9.130.3.3 long tk_nin

Definition at line 117 of file tty.c.

Referenced by ttyinput().

9.130.3.4 long tk_nout

Definition at line 118 of file tty.c.

Referenced by ttwrite(), and ttyoutput().

9.130.3.5 long tk_rawcc

Definition at line 119 of file tty.c.

Referenced by ttyinput().

9.130.3.6 struct cdevsw tty_cdevsw [static]

Initial value:

```
{
        .d_version =    D_VERSION,
        .d_open =       ttyopen,
        .d_close =      ttyclose,
        .d_ioctl =      ttyioctl,
        .d_purge =      ttypurge,
        .d_name =       "ttydrv",
        .d_flags =      D_TTY | D_NEEDGIANT,
}
```

Definition at line 128 of file tty.c.

Referenced by ttycreate().

9.130.3.7 d_purge_t ttypurge [static]

Definition at line 125 of file tty.c.

Referenced by ttycreate().

9.130.3.8 struct filterops ttyread_filtops [static]

Initial value:

```
        { 1, NULL, filt_ttyrdetach, filt_ttyread }
```

Definition at line 1307 of file tty.c.

Referenced by ttykqfilter().

9.130.3.9 struct cdevsw ttys_cdevsw [static]

Initial value:

```
{
        .d_version =    D_VERSION,
        .d_open =       ttysopen,
        .d_close =      ttysclose,
```

```
        .d_read =       ttysrdwr,
        .d_write =      ttysrdwr,
        .d_ioctl =      ttysioctl,
        .d_name =       "TTYS",
        .d_flags =      D_TTY | D_NEEDGIANT,
}
```

Definition at line 139 of file tty.c.

Referenced by ttycreate().

9.130.3.10 d_close_t ttysclose [static]

Definition at line 122 of file tty.c.

9.130.3.11 d_ioctl_t ttysioctl [static]

Definition at line 124 of file tty.c.

9.130.3.12 d_open_t ttysopen [static]

Definition at line 121 of file tty.c.

9.130.3.13 d_read_t ttysrdwr [static]

Definition at line 123 of file tty.c.

9.130.3.14 struct filterops ttywrite_filtops [static]

Initial value:

```
{ 1, NULL, filt_ttywdetach, filt_ttywrite }
```

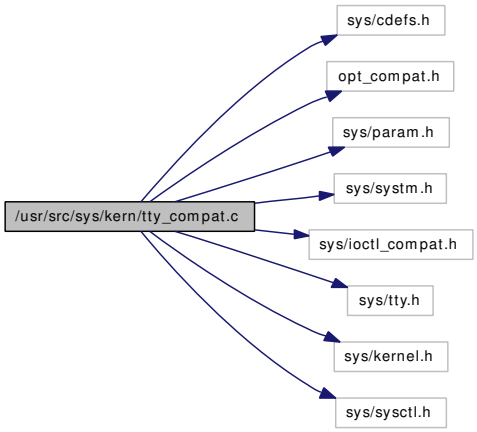
Definition at line 1309 of file tty.c.

Referenced by ttykqfilter().

9.131 /usr/src/sys/kern/tty_compat.c File Reference

```
#include <sys/cdefs.h>
#include "opt_compat.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/ioctl_compat.h>
#include <sys/tty.h>
#include <sys/kernel.h>
#include <sys/sysctl.h>
```

Include dependency graph for tty_compat.c:



Defines

- #define MAX_SPEED 17

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/tty_compat.c,v 1.39 2006/01/10 09:19:09 phk Exp $")
- static int ttcompatgetflags (struct tty *tp)
- static void ttcompatsetflags (struct tty *tp, struct termios *t)
- static void ttcompatsetlflags (struct tty *tp, struct termios *t)
- static int ttcompatspeedtab (int speed, struct speedtab *table)
- SYSCTL_INT (_debug, OID_AUTO, ttydebug, CTLFLAG_RW,&ttydebug, 0,"")
- static int ttsetcompat (struct tty *tp, u_long *com, caddr_t data, struct termios *term)
- int ttcompat (struct tty *tp, u_long com, caddr_t data, int flag)

Variables

- static int ttydebug = 0
- static struct speedtab compatspeeds [ ]
- static int compatspcodes [ ]

9.131.1 Define Documentation

9.131.1.1 #define MAX_SPEED 17

Referenced by ttsetcompat().

9.131.2 Function Documentation

9.131.2.1 __FBSDID ("$FreeBSD: src/sys/kern/tty_compat. *c*, v 1.39 2006/01/10 09:19:09 phk Exp $")

9.131.2.2 SYSCTL_INT (_debug, OID_AUTO, ttydebug, CTLFLAG_RW, & *ttydebug*, 0, "")

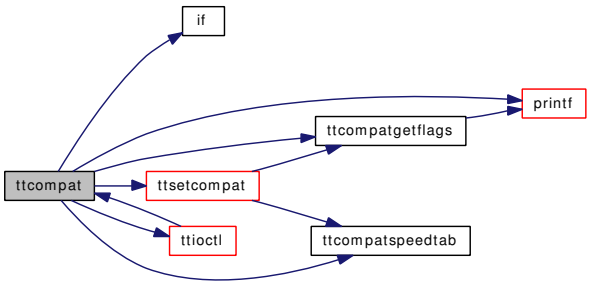
9.131.2.3 int ttcompat (struct tty * *tp*, u_long *com*, caddr_t *data*, int *flag*)

Definition at line 174 of file tty_compat.c.

References compatspeeds, if(), printf(), ttcompatgetflags(), ttcompatspeedtab(), ttioctl(), ttsetcompat(), and ttydebug.

Referenced by ttioctl().

Here is the call graph for this function:



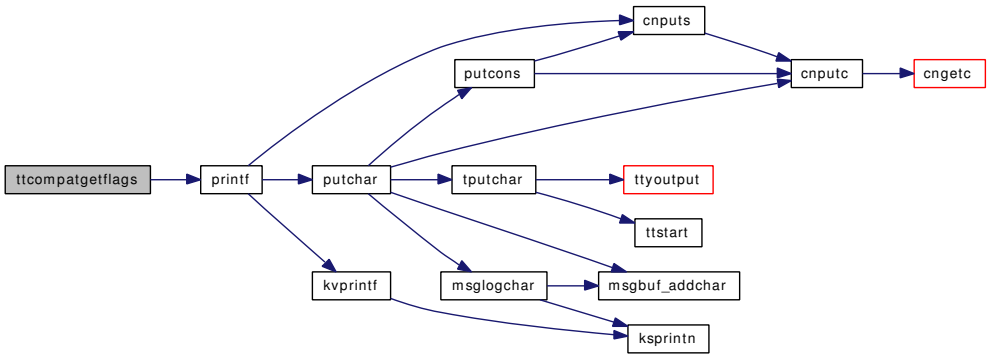
9.131.2.4 static int ttcompatgetflags (struct tty * *tp*) [static]

Definition at line 261 of file tty_compat.c.

References printf(), and ttydebug.

Referenced by ttcompat(), and ttsetcompat().

Here is the call graph for this function:



9.131.2.5 static void ttcompatsetflags (struct tty * *tp*, struct termios * *t*) [static]

Definition at line 321 of file tty_compat.c.

Referenced by ttsetcompat().

9.131.2.6 static void ttcompatsetlflags (struct tty * *tp*, struct termios * *t*) [static]

Definition at line 398 of file tty_compat.c.

Referenced by ttsetcompat().

9.131.2.7 static int ttcompatspeedtab (int *speed*, struct speedtab * *table*) [static]

Definition at line 84 of file tty_compat.c.

Referenced by ttcompat(), and ttsetcompat().

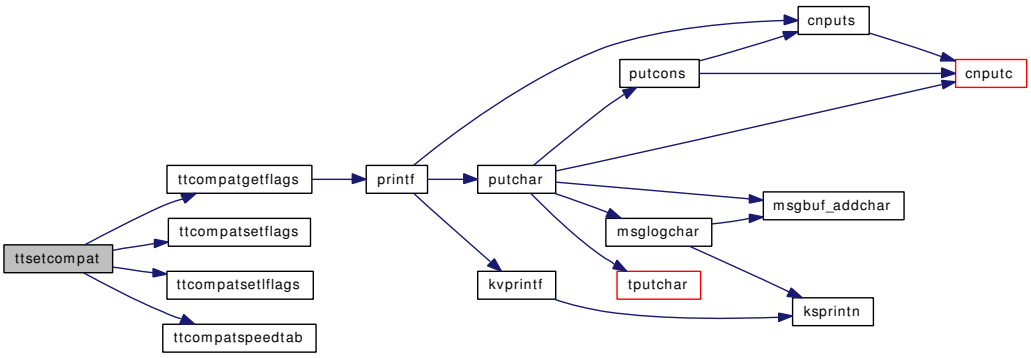
9.131.2.8 static int ttsetcompat (struct tty * *tp*, u_long * *com*, caddr_t *data*, struct termios * *term*) [static]

Definition at line 95 of file tty_compat.c.

References compatspcodes, compatspeeds, MAX_SPEED, ttcompatgetflags(), ttcompatsetflags(), ttcompatsetlflags(), and ttcompatspeedtab().

Referenced by ttcompat().

Here is the call graph for this function:



9.131.3 Variable Documentation

9.131.3.1 int compatspcodes[ ] [static]

Initial value:

```
 {
        0, 50, 75, 110, 134, 150, 200, 300, 600, 1200,
        1800, 2400, 4800, 9600, 19200, 38400, 57600, 115200,
}
```

Definition at line 78 of file tty_compat.c.

Referenced by ttsetcompat().

9.131.3.2 struct speedtab compatspeeds[ ] [static]

Initial value:

```
 {
#define MAX_SPEED
        { 115200, 17 },
        { 57600, 16 },
        { 38400, 15 },
        { 19200, 14 },
        { 9600, 13 },
        { 4800, 12 },
        { 2400, 11 },
        { 1800, 10 },
        { 1200, 9 },
        { 600,  8 },
        { 300,  7 },
        { 200,  6 },
        { 150,  5 },
        { 134,  4 },
        { 110,  3 },
        { 75,   2 },
        { 50,   1 },
        { 0,    0 },
        { -1,   -1 },
}
```

Definition at line 56 of file tty_compat.c.

Referenced by ttcompat(), and ttsetcompat().

9.131.3.3 int ttydebug = 0 [static]

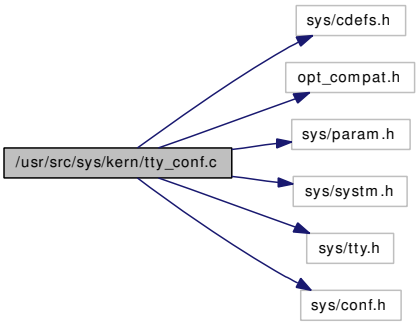
Definition at line 53 of file tty_compat.c.

Referenced by ttcompat(), and ttcompatgetflags().

9.132 /usr/src/sys/kern/tty_conf.c File Reference

```
#include <sys/cdefs.h>
#include "opt_compat.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/tty.h>
#include <sys/conf.h>
```

Include dependency graph for tty_conf.c:



Defines

- #define MAXLDISC 9
- #define ntty_disc nodisc
- #define LOADABLE_LDISC 7

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/tty_conf.c,v 1.24 2004/07/15 20:47:40 phk Exp $")
- int ldisc_register (int discipline, struct linesw *linesw_p)
- void ldisc_deregister (int discipline)
- static int l_noopen (struct cdev *dev, struct tty *tp)
- static int l_noclose (struct tty *tp, int flag)
- int l_noread (struct tty *tp, struct uio *uio, int flag)
- int l_nowrite (struct tty *tp, struct uio *uio, int flag)
- static int l_norint (int c, struct tty *tp)
- static int l_nostart (struct tty *tp)
- int l_nullioctl (struct tty *tp, u_long cmd, char *data, int flags, struct thread *td)

Variables

- static l_open_t l_noopen
- static l_close_t l_noclose
- static l_rint_t l_norint
- static l_start_t l_nostart

- static struct linesw nodisc
- static struct linesw termios_disc
- linesw * linesw [MAXLDISC]
- int nlinesw = sizeof (linesw) / sizeof (linesw[0])

9.132.1 Define Documentation

9.132.1.1 #define LOADABLE_LDISC 7

Definition at line 106 of file tty_conf.c.

Referenced by ldisc_register().

9.132.1.2 #define MAXLDISC 9

Definition at line 49 of file tty_conf.c.

Referenced by ldisc_deregister(), and ldisc_register().

9.132.1.3 #define ntty_disc nodisc

Definition at line 89 of file tty_conf.c.

9.132.2 Function Documentation

9.132.2.1 __FBSDID ("$FreeBSD: src/sys/kern/tty_conf. *c*, v 1.24 2004/07/15 20:47:40 phk Exp $")

9.132.2.2 static int l_noclose (struct tty * *tp*, int *flag*) [static]

Definition at line 166 of file tty_conf.c.

9.132.2.3 static int l_noopen (struct cdev * *dev*, struct tty * *tp*) [static]

Definition at line 159 of file tty_conf.c.

9.132.2.4 int l_noread (struct tty * *tp*, struct uio * *uio*, int *flag*)

Definition at line 173 of file tty_conf.c.

9.132.2.5 static int l_norint (int *c*, struct tty * *tp*) [static]

Definition at line 187 of file tty_conf.c.

9.132.2.6 static int l_nostart (struct tty * *tp*) [static]

Definition at line 194 of file tty_conf.c.

9.132.2.7 int l_nowrite (struct tty * *tp*, struct uio * *uio*, int *flag*)

Definition at line 180 of file tty_conf.c.

9.132.2.8 int l_nullioctl (struct tty * *tp*, u_long *cmd*, char * *data*, int *flags*, struct thread * *td*)

Definition at line 201 of file tty_conf.c.

9.132.2.9 void ldisc_deregister (int *discipline*)

Definition at line 147 of file tty_conf.c.

References MAXLDISC, and nodisc.

9.132.2.10 int ldisc_register (int *discipline*, struct linesw * *linesw_p*)

Definition at line 118 of file tty_conf.c.

References LOADABLE_LDISC, MAXLDISC, and nodisc.

9.132.3 Variable Documentation

9.132.3.1 l_close_t l_noclose [static]

Definition at line 53 of file tty_conf.c.

9.132.3.2 l_open_t l_noopen [static]

Definition at line 52 of file tty_conf.c.

9.132.3.3 l_rint_t l_norint [static]

Definition at line 54 of file tty_conf.c.

9.132.3.4 l_start_t l_nostart [static]

Definition at line 55 of file tty_conf.c.

9.132.3.5 struct linesw* linesw[MAXLDISC]

Initial value:

```
{
        &termios_disc,
        &nodisc,
        &ntty_disc,
        &nodisc,
        &nodisc,
        &nodisc,
        &nodisc,
```

```
        &nodisc,
        &nodisc,
}
```

Definition at line 92 of file tty_conf.c.

Referenced by ttyldoptim().

9.132.3.6 int nlinesw = sizeof (linesw) / sizeof (linesw[0])

Definition at line 104 of file tty_conf.c.

Referenced by ttioctl().

9.132.3.7 struct linesw nodisc [static]

Initial value:

```
 {
        .l_open =       l_noopen,
        .l_close =      l_noclose,
        .l_read =       l_noread,
        .l_write =      l_nowrite,
        .l_ioctl =      l_nullioctl,
        .l_rint =       l_norint,
        .l_start =      l_nostart,
        .l_modem =      ttymodem
}
```

Definition at line 64 of file tty_conf.c.

Referenced by ldisc_deregister(), and ldisc_register().

9.132.3.8 struct linesw termios_disc [static]

Initial value:

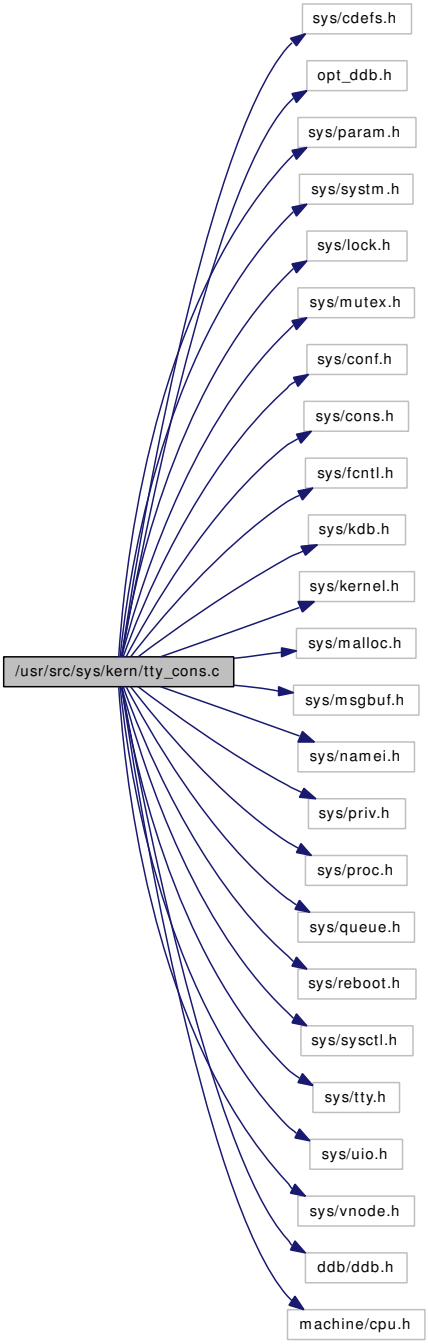
```
 {
        .l_open =       tty_open,
        .l_close =      ttylclose,
        .l_read =       ttread,
        .l_write =      ttwrite,
        .l_ioctl =      l_nullioctl,
        .l_rint =       ttyinput,
        .l_start =      ttstart,
        .l_modem =      ttymodem
}
```

Definition at line 75 of file tty_conf.c.

9.133 /usr/src/sys/kern/tty_cons.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ddb.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/conf.h>
#include <sys/cons.h>
#include <sys/fcntl.h>
#include <sys/kdb.h>
#include <sys/kernel.h>
#include <sys/malloc.h>
#include <sys/msgbuf.h>
#include <sys/namei.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/queue.h>
#include <sys/reboot.h>
#include <sys/sysctl.h>
#include <sys/tty.h>
#include <sys/uio.h>
#include <sys/vnode.h>
#include <ddb/ddb.h>
#include <machine/cpu.h>
```

Include dependency graph for tty_cons.c:



Data Structures

- struct cn_device

Defines

- #define CNDEVPATHMAX 32
- #define CNDEVTAB_SIZE 4

- #define CND_INVALID(cnd, td)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/tty_cons.c,v 1.138 2006/11/06 13:42:01 rwatson Exp $")
- static STAILQ_HEAD (cn_device)
- void cninit_finish ()
- int cnadd (struct consdev *cn)
- void cnremove (struct consdev *cn)
- void cnselect (struct consdev *cn)
- void cnavailable (struct consdev *cn, int available)
- int cnunavailable (void)
- static int sysctl_kern_console (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern, OID_AUTO, console, CTLTYPE_STRING|CTLFLAG_RW, 0, 0, sysctl_kern_console,"A","Console device control")
- static int sysctl_kern_consmute (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern, OID_AUTO, consmute, CTLTYPE_INT|CTLFLAG_RW, 0, sizeof(cn_mute), sysctl_kern_consmute,"I","")
- static int cn_devopen (struct cn_device *cnd, struct thread *td, int forceopen)
- static int cnopen (struct cdev *dev, int flag, int mode, struct thread *td)
- static int cnclose (struct cdev *dev, int flag, int mode, struct thread *td)
- static int cnread (struct cdev *dev, struct uio *uio, int flag)
- static int cnwrite (struct cdev *dev, struct uio *uio, int flag)
- static int cnioctl (struct cdev *dev, u_long cmd, caddr_t data, int flag, struct thread *td)
- static int cnpoll (struct cdev *dev, int events, struct thread *td)
- static int cnkqfilter (struct cdev *dev, struct knote *kn)
- int cngetc (void)
- int cncheckc (void)
- void cnputc (int c)
- void cnputs (char *p)
- SYSCTL_INT (_kern, OID_AUTO, consmsgbuf_size, CTLFLAG_RW,&consmsgbuf_size, 0,"")
- void constty_set (struct tty *tp)
- void constty_clear (void)
- SYSCTL_INT (_kern, OID_AUTO, constty_wakeups_per_second, CTLFLAG_RW,&constty_wakeups_per_second, 0,"")
- static void constty_timeout (void *arg)
- static void cn_drvinit (void *unused)

Variables

- static d_open_t cnopen
- static d_close_t cnclose
- static d_read_t cnread
- static d_write_t cnwrite
- static d_ioctl_t cnioctl
- static d_poll_t cnpoll
- static d_kqfilter_t cnkqfilter
- static struct cdevsw cn_cdevsw
- static struct cn_device cn_devtab [CNDEVTAB_SIZE]
- static int consmsgbuf_size = 8192
- static int constty_wakeups_per_second = 5

9.133.1 Define Documentation

9.133.1.1 #define CND_INVALID(cnd, td)

Value:

```
(cnd == NULL || cnd->cnd_vp == NULL ||                          \
            (cnd->cnd_vp->v_type == VBAD && !cn_devopen(cnd, td, 1)))
```

Referenced by cnioctl(), cnkqfilter(), cnpoll(), cnread(), and cnwrite().

9.133.1.2 #define CNDEVPATHMAX 32

Definition at line 94 of file tty_cons.c.

Referenced by cn_devopen(), and sysctl_kern_console().

9.133.1.3 #define CNDEVTAB_SIZE 4

Definition at line 95 of file tty_cons.c.

Referenced by cnadd(), cnavailable(), and cnremove().

9.133.2 Function Documentation

9.133.2.1 __FBSDID ("$FreeBSD: src/sys/kern/tty_cons. *c*, v 1.138 2006/11/06 13:42:01 rwatson Exp $")

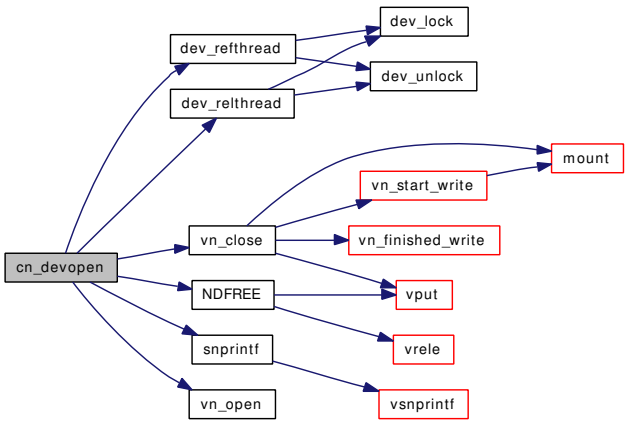
9.133.2.2 static int cn_devopen (struct cn_device * *cnd*, struct thread * *td*, int *forceopen*) [static]

Definition at line 386 of file tty_cons.c.

References CNDEVPATHMAX, dev_refthread(), dev_relthread(), NDFREE(), snprintf(), vn_close(), and vn_open().

Referenced by cnopen().

Here is the call graph for this function:

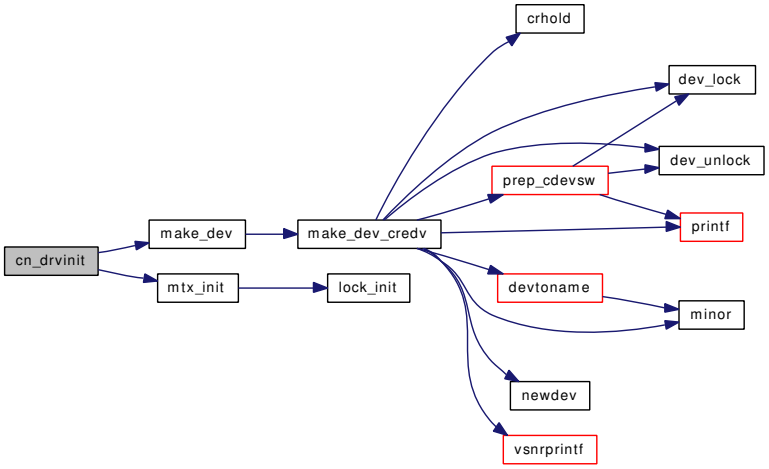


9.133.2.3 static void cn_drvinit (void * *unused*) [static]

Definition at line 730 of file tty_cons.c.

References cn_cdevsw, make_dev(), and mtx_init().

Here is the call graph for this function:



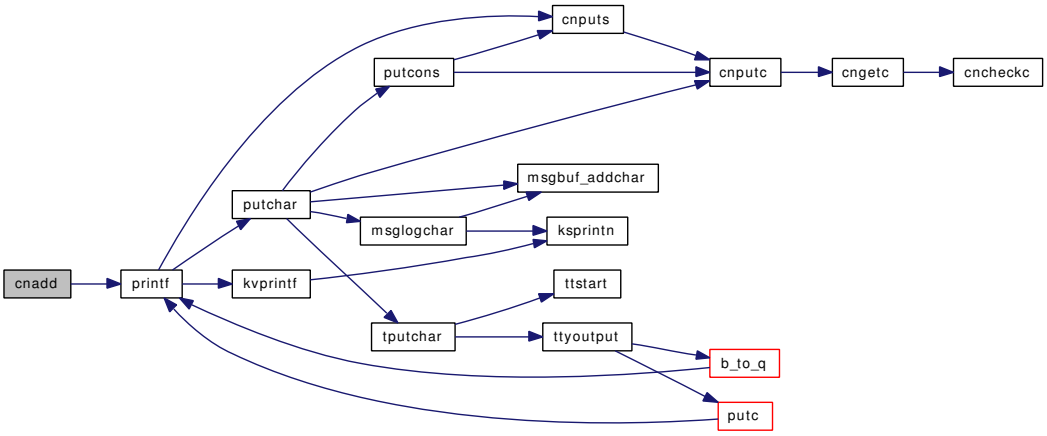
9.133.2.4 int cnadd (struct consdev * *cn*)

Definition at line 192 of file tty_cons.c.

References cn_devtab, CNDEVTAB_SIZE, and printf().

Referenced by sysctl_kern_console().

Here is the call graph for this function:



9.133.2.5 void cnavailable (struct consdev * *cn*, int *available*)

Definition at line 271 of file tty_cons.c.

References cn_devtab, and CNDEVTAB_SIZE.

9.133.2.6 int cncheckc (void)

Definition at line 595 of file tty_cons.c.

References kdb_active.

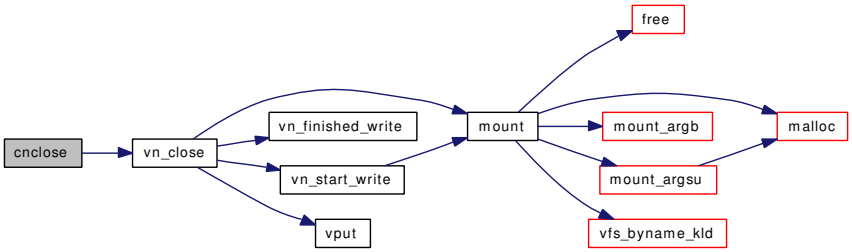
Referenced by cngetc(), and shutdown_panic().

9.133.2.7 static int cnclose (struct cdev * *dev*, int *flag*, int *mode*, struct thread * *td*) [static]

Definition at line 437 of file tty_cons.c.

References vn_close().

Here is the call graph for this function:



9.133.2.8 int cngetc (void)

Definition at line 581 of file tty_cons.c.

References cncheckc().

Referenced by cnputc(), shutdown_halt(), and shutdown_panic().

Here is the call graph for this function:

9.133.2.9 void cninit_finish ()

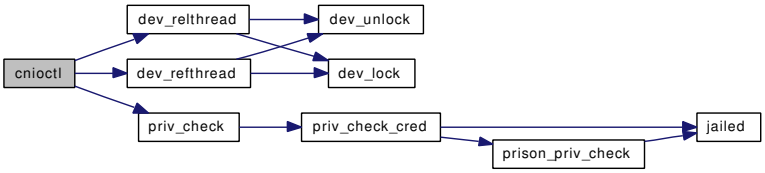
Definition at line 185 of file tty_cons.c.

9.133.2.10 static int cnioctl (struct cdev * *dev*, u_long *cmd*, caddr_t *data*, int *flag*, struct thread * *td*) [static]

Definition at line 500 of file tty_cons.c.

References CND_INVALID, dev_refthread(), dev_relthread(), and priv_check().

Here is the call graph for this function:

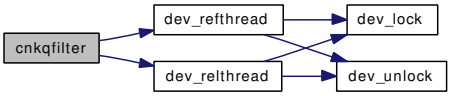


9.133.2.11 static int cnkqfilter (struct cdev * *dev*, struct knote * *kn*) [static]

Definition at line 557 of file tty_cons.c.

References CND_INVALID, dev_refthread(), and dev_relthread().

Here is the call graph for this function:

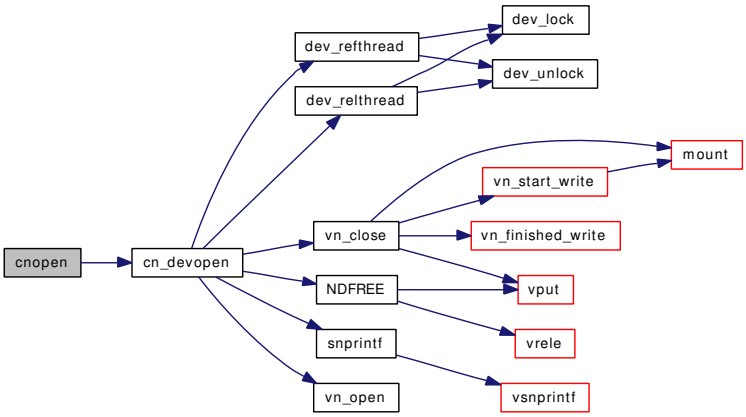


9.133.2.12 static int cnopen (struct cdev * *dev*, int *flag*, int *mode*, struct thread * *td*) [static]

Definition at line 423 of file tty_cons.c.

References cn_devopen().

Here is the call graph for this function:

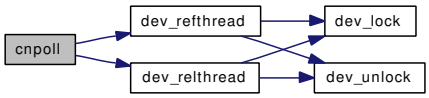


9.133.2.13 static int cnpoll (struct cdev * *dev*, int *events*, struct thread * *td*) [static]

Definition at line 536 of file tty_cons.c.

References CND_INVALID, dev_refthread(), and dev_relthread().

Here is the call graph for this function:



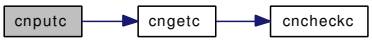
9.133.2.14 void cnputc (int *c*)

Definition at line 619 of file tty_cons.c.

References cngetc(), and kdb_active.

Referenced by cnputs(), constty_clear(), putchar(), and putcons().

Here is the call graph for this function:



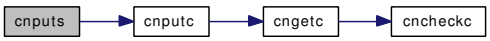
9.133.2.15 void cnputs (char * *p*)

Definition at line 648 of file tty_cons.c.

References cnputc().

Referenced by printf(), putcons(), and vprintf().

Here is the call graph for this function:

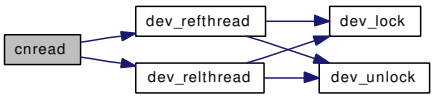


9.133.2.16 static int cnread (struct cdev * *dev*, struct uio * *uio*, int *flag*) [static]

Definition at line 453 of file tty_cons.c.

References CND_INVALID, dev_refthread(), and dev_relthread().

Here is the call graph for this function:



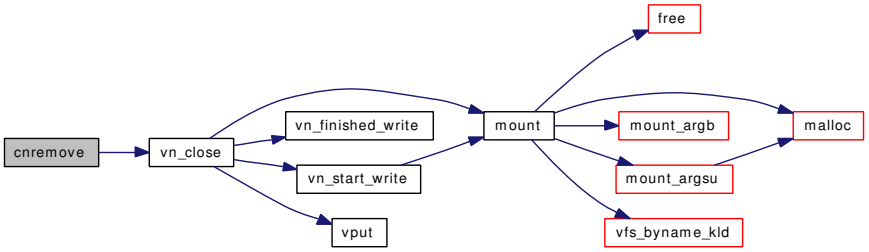
9.133.2.17 void cnremove (struct consdev * *cn*)

Definition at line 221 of file tty_cons.c.

References cn_devtab, CNDEVTAB_SIZE, and vn_close().

Referenced by sysctl_kern_console().

Here is the call graph for this function:



9.133.2.18 void cnselect (struct consdev * *cn*)

Definition at line 255 of file tty_cons.c.

Referenced by sysctl_kern_console().

9.133.2.19 int cnunavailable (void)

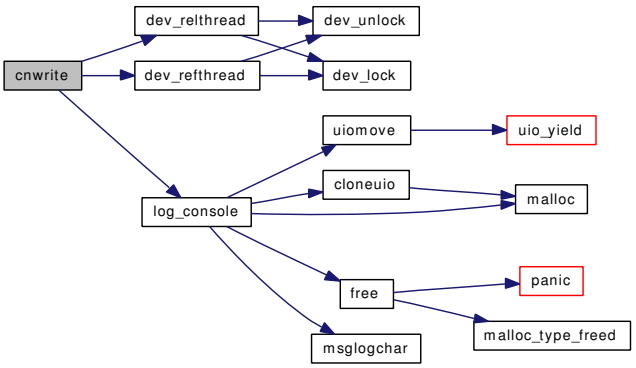
Definition at line 291 of file tty_cons.c.

9.133.2.20 static int cnwrite (struct cdev * *dev*, struct uio * *uio*, int *flag*) [static]

Definition at line 472 of file tty_cons.c.

References CND_INVALID, dev_refthread(), dev_relthread(), and log_console().

Here is the call graph for this function:



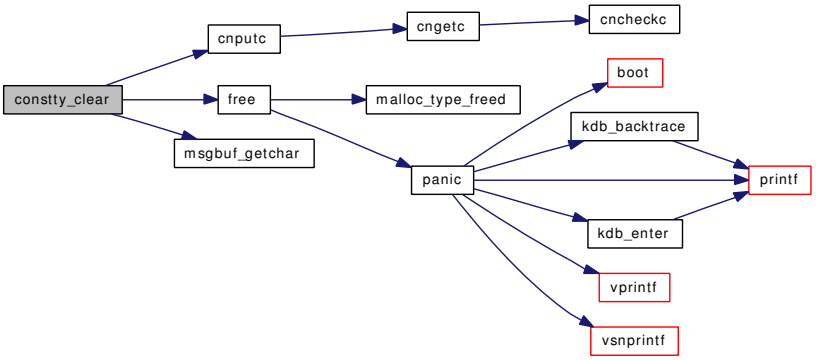
9.133.2.21 void constty_clear (void)

Definition at line 692 of file tty_cons.c.

References cnputc(), free(), and msgbuf_getchar().

Referenced by constty_timeout(), ttioctl(), and tty_close().

Here is the call graph for this function:



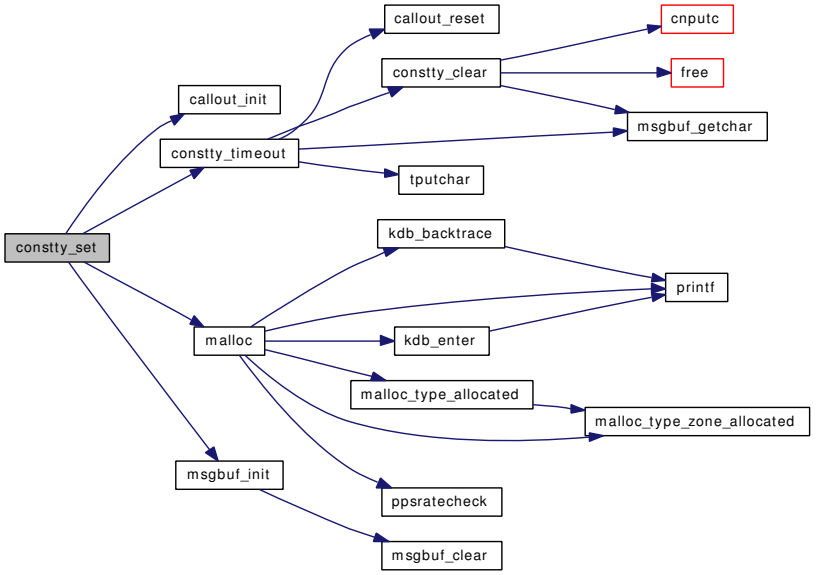
9.133.2.22 void constty_set (struct tty * *tp*)

Definition at line 673 of file tty_cons.c.

References callout_init(), constty_timeout(), malloc(), and msgbuf_init().

Referenced by ttioctl().

Here is the call graph for this function:



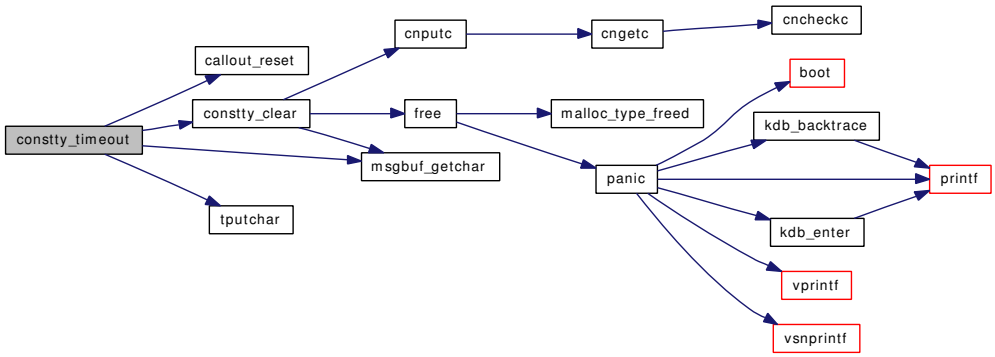
9.133.2.23 static void constty_timeout (void * *arg*) [static]

Definition at line 712 of file tty_cons.c.

References callout_reset(), constty_clear(), hz, msgbuf_getchar(), and tputchar().

Referenced by constty_set().

Here is the call graph for this function:



9.133.2.24 static STAILQ_HEAD (cn_device) [static]

Definition at line 97 of file tty_cons.c.

9.133.2.25 SYSCTL_INT (_kern, OID_AUTO, constty_wakeups_per_second, CTLFLAG_RW, & *constty_wakeups_per_second*, 0, "")

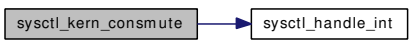
9.133.2.26 SYSCTL_INT (_kern, OID_AUTO, consmsgbuf_size, CTLFLAG_RW, & *consmsgbuf_size*, 0, "")

9.133.2.27 static int sysctl_kern_consmute (SYSCTL_HANDLER_ARGS) [static]

Definition at line 364 of file tty_cons.c.

References cnclose, cnopen, and sysctl_handle_int().

Here is the call graph for this function:

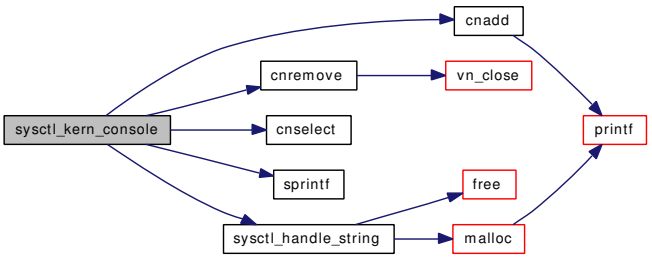


9.133.2.28 static int sysctl_kern_console (SYSCTL_HANDLER_ARGS) [static]

Definition at line 302 of file tty_cons.c.

References cnadd(), CNDEVPATHMAX, cnremove(), cnselect(), sprintf(), and sysctl_handle_string().

Here is the call graph for this function:



9.133.2.29 SYSCTL_PROC (_kern, OID_AUTO, consmute, CTLTYPE_INT| *CTLFLAG_RW*, 0, sizeof(cn_mute), sysctl_kern_consmute, "I", "")

9.133.2.30 SYSCTL_PROC (_kern, OID_AUTO, console, CTLTYPE_STRING| *CTLFLAG_RW*, 0, 0, sysctl_kern_console, "A", "Console device control")

9.133.3 Variable Documentation

9.133.3.1 struct cdevsw cn_cdevsw [static]

Initial value:

```
{
        .d_version =    D_VERSION,
        .d_open =       cnopen,
        .d_close =      cnclose,
        .d_read =       cnread,
        .d_write =      cnwrite,
        .d_ioctl =      cnioctl,
        .d_poll =       cnpoll,
        .d_name =       "console",
        .d_flags =      D_TTY | D_NEEDGIANT,
        .d_kqfilter =   cnkqfilter,
}
```

Definition at line 75 of file tty_cons.c.

Referenced by cn_drvinit().

9.133.3.2 struct cn_device cn_devtab[CNDEVTAB_SIZE] [static]

Definition at line 96 of file tty_cons.c.

Referenced by cnadd(), cnavailable(), and cnremove().

9.133.3.3 d_close_t cnclose [static]

Definition at line 68 of file tty_cons.c.

Referenced by sysctl_kern_consmute().

9.133.3.4 d_ioctl_t cnioctl [static]

Definition at line 71 of file tty_cons.c.

9.133.3.5 d_kqfilter_t cnkqfilter [static]

Definition at line 73 of file tty_cons.c.

9.133.3.6 d_open_t cnopen [static]

Definition at line 67 of file tty_cons.c.

Referenced by sysctl_kern_consmute().

9.133.3.7 d_poll_t cnpoll [static]

Definition at line 72 of file tty_cons.c.

9.133.3.8 d_read_t cnread [static]

Definition at line 69 of file tty_cons.c.

9.133.3.9 d_write_t cnwrite [static]

Definition at line 70 of file tty_cons.c.

9.133.3.10 int consmsgbuf_size = 8192 [static]

Definition at line 665 of file tty_cons.c.

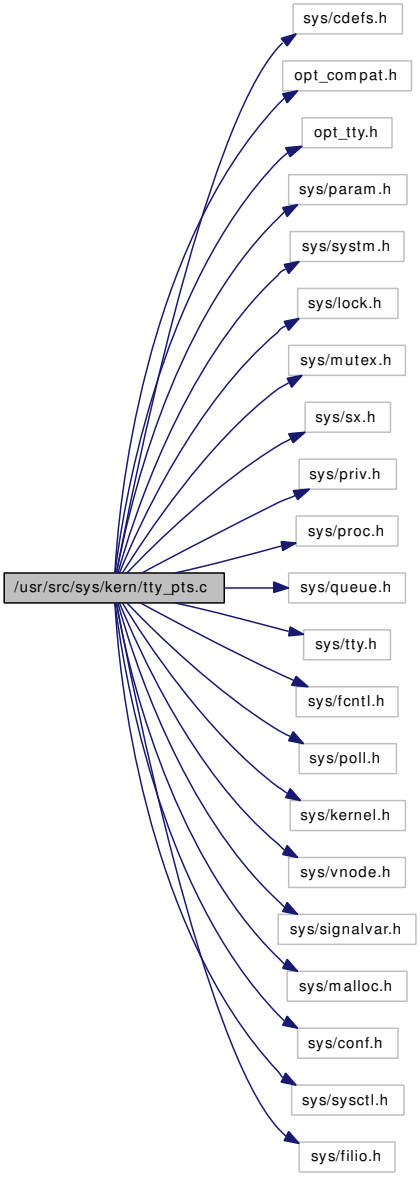
9.133.3.11 int constty_wakeups_per_second = 5 [static]

Definition at line 707 of file tty_cons.c.

9.134 /usr/src/sys/kern/tty_pts.c File Reference

```
#include <sys/cdefs.h>
#include "opt_compat.h"
#include "opt_tty.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/sx.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/queue.h>
#include <sys/tty.h>
#include <sys/fcntl.h>
#include <sys/poll.h>
#include <sys/kernel.h>
#include <sys/vnode.h>
#include <sys/signalvar.h>
#include <sys/malloc.h>
#include <sys/conf.h>
#include <sys/sysctl.h>
#include <sys/filio.h>
```

Include dependency graph for tty_pts.c:



Data Structures

- struct pt_desc

Defines

- #define BUFSIZ 100
- #define TSA_PTC_READ(tp) ((void *)&(tp) → t_outq.c_cf)
- #define TSA_PTC_WRITE(tp) ((void *)&(tp) → t_rawq.c_cl)
- #define TSA_PTS_READ(tp) ((void *)&(tp) → t_canq)
- #define NUM_TO_MINOR(c) ((c & 0xff) | ((c & ~0xff) << 16))
- #define PF_PKT 0x008
- #define PF_STOPPED 0x010

- #define PF_NOSTOP 0x040
- #define PF_UCNTL 0x080

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/tty_pts.c,v 1.14 2007/01/08 17:49:59 rwatson Exp $")
- static MALLOC_DEFINE (M_PTY,"ptys","pty data structures")
- static void ptsstart (struct tty *tp)
- static void ptsstop (struct tty *tp, int rw)
- static void ptcwakeup (struct tty *tp, int flag)
- static LIST_HEAD (pt_desc)
- static void pty_release (struct pt_desc *pt)
- static void pty_maybecleanup (struct pt_desc *pt)
- static int ptsopen (struct cdev *dev, int flag, int devtype, struct thread *td)
- static int ptsclose (struct cdev *dev, int flag, int mode, struct thread *td)
- static int ptsread (struct cdev *dev, struct uio *uio, int flag)
- static int ptswrite (struct cdev *dev, struct uio *uio, int flag)
- static int ptcopen (struct cdev *dev, int flag, int devtype, struct thread *td)
- static int ptcclose (struct cdev *dev, int flags, int fmt, struct thread *td)
- static int ptcread (struct cdev *dev, struct uio *uio, int flag)
- static int ptcpoll (struct cdev *dev, int events, struct thread *td)
- static int ptcwrite (struct cdev *dev, struct uio *uio, int flag)
- static int ptcioctl (struct cdev *dev, u_long cmd, caddr_t data, int flag, struct thread *td)
- static int ptsioctl (struct cdev *dev, u_long cmd, caddr_t data, int flag, struct thread *td)
- static void pty_clone (void *arg, struct ucred *cred, char *name, int namelen, struct cdev **dev)
- static void pty_drvinit (void *unused)

Variables

- static d_open_t ptsopen
- static d_close_t ptsclose
- static d_read_t ptsread
- static d_write_t ptswrite
- static d_ioctl_t ptsioctl
- static d_ioctl_t ptcioctl
- static d_open_t ptcopen
- static d_close_t ptcclose
- static d_read_t ptcread
- static d_write_t ptcwrite
- static d_poll_t ptcpoll
- static struct cdevsw pts_cdevsw
- static struct cdevsw ptc_cdevsw
- static struct mtx pt_mtx

9.134.1 Define Documentation

9.134.1.1 #define BUFSIZ 100

Definition at line 117 of file tty_pts.c.

Referenced by ptcread(), and ptcwrite().

9.134.1.2 #define NUM_TO_MINOR(c) ((c & 0xff) | ((c & ~0xff) << 16))

Definition at line 123 of file tty_pts.c.

Referenced by ptcopen(), and pty_clone().

9.134.1.3 #define PF_NOSTOP 0x040

Referenced by ptsioctl().

9.134.1.4 #define PF_PKT 0x008

Referenced by ptcioctl(), ptcpoll(), ptcread(), and ptsioctl().

9.134.1.5 #define PF_STOPPED 0x010

Referenced by ptsstart(), and ptsstop().

9.134.1.6 #define PF_UCNTL 0x080

Referenced by ptcioctl(), ptcpoll(), and ptsioctl().

9.134.1.7 #define TSA_PTC_READ(tp) ((void *)&(tp) → t_outq.c_cf)

Definition at line 119 of file tty_pts.c.

Referenced by ptcwakeup().

9.134.1.8 #define TSA_PTC_WRITE(tp) ((void *)&(tp) → t_rawq.c_cl)

Definition at line 120 of file tty_pts.c.

Referenced by ptcwakeup(), and ptcwrite().

9.134.1.9 #define TSA_PTS_READ(tp) ((void *)&(tp) → t_canq)

Definition at line 121 of file tty_pts.c.

9.134.2 Function Documentation

9.134.2.1 __FBSDID ("$FreeBSD: src/sys/kern/tty_pts. *c*, v 1.14 2007/01/08 17:49:59 rwatson Exp $")

9.134.2.2 static LIST_HEAD (pt_desc) [static]

Definition at line 152 of file tty_pts.c.

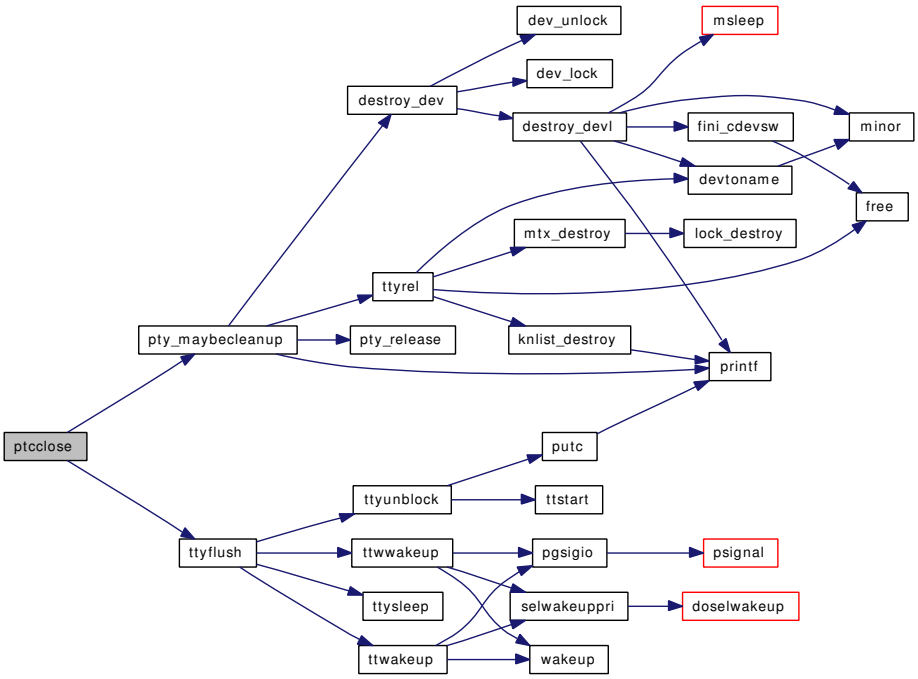
9.134.2.3 static MALLOC_DEFINE (M_PTY, "ptys", "pty data structures") [static]

9.134.2.4 static int ptcclose (struct cdev * *dev*, int *flags*, int *fmt*, struct thread * *td*) [static]

Definition at line 436 of file tty_pts.c.

References pty_maybecleanup(), and ttyflush().

Here is the call graph for this function:

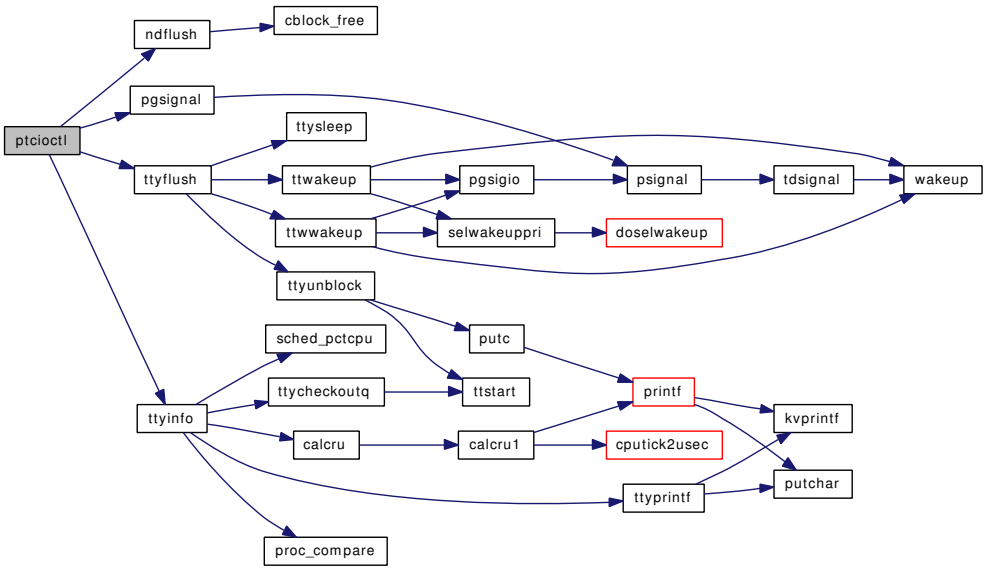


9.134.2.5 static int ptcioctl (struct cdev * *dev*, u_long *cmd*, caddr_t *data*, int *flag*, struct thread * *td*) [static]

Definition at line 657 of file tty_pts.c.

References ndflush(), PF_PKT, PF_UCNTL, pgsignal(), pt_desc::pt_num, ptsioctl, ttyflush(), and ttyinfo().

Here is the call graph for this function:

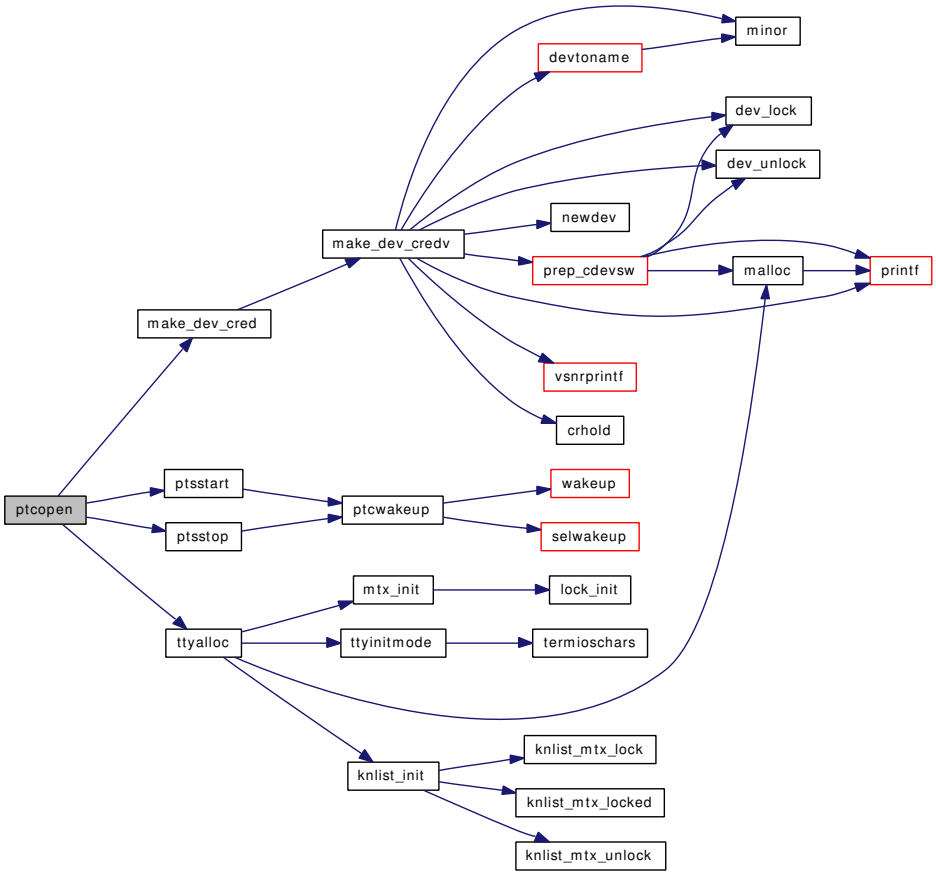


9.134.2.6 static int ptcopen (struct cdev * *dev*, int *flag*, int *devtype*, struct thread * *td*) [static]

Definition at line 387 of file tty_pts.c.

References make_dev_cred(), NUM_TO_MINOR, pt_desc::pt_num, pts_cdevsw, ptsstart(), ptsstop(), and ttyalloc().

Here is the call graph for this function:



9.134.2.7 static int ptcpoll (struct cdev * *dev*, int *events*, struct thread * *td*) [static]

Definition at line 545 of file tty_pts.c.

References PF_PKT, PF_UCNTL, and selrecord().

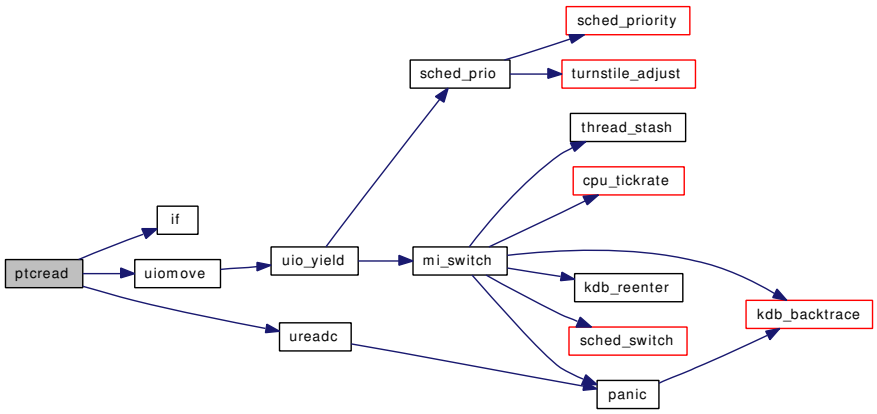
Here is the call graph for this function:

9.134.2.8 static int ptcread (struct cdev * *dev*, struct uio * *uio*, int *flag*) [static]

Definition at line 465 of file tty_pts.c.

References buf, BUFSIZ, if(), PF_PKT, uiomove(), and ureadc().

Here is the call graph for this function:



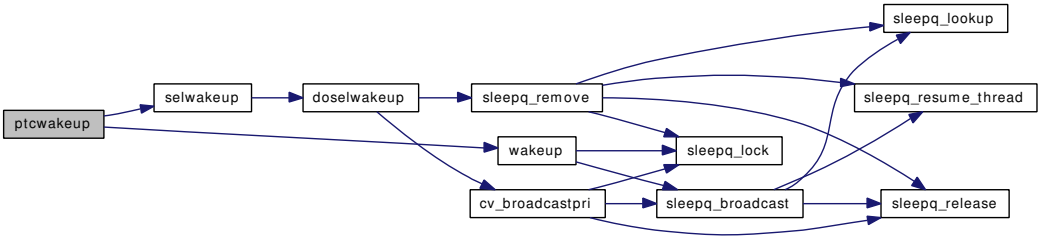
9.134.2.9 static void ptcwakeup (struct tty * *tp*, int *flag*) [static]

Definition at line 360 of file tty_pts.c.

References selwakeup(), TSA_PTC_READ, TSA_PTC_WRITE, and wakeup().

Referenced by ptsioctl(), ptsopen(), ptsread(), ptsstart(), and ptsstop().

Here is the call graph for this function:

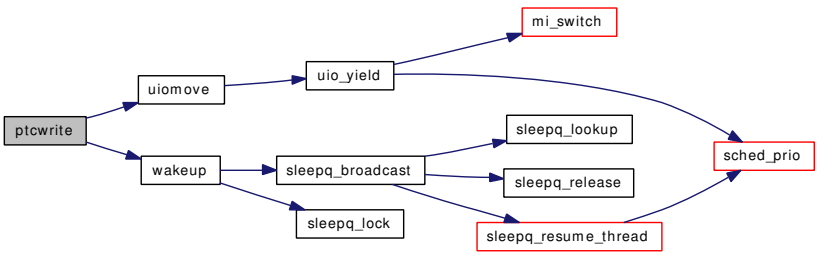


9.134.2.10 static int ptcwrite (struct cdev * *dev*, struct uio * *uio*, int *flag*) [static]

Definition at line 591 of file tty_pts.c.

References BUFSIZ, TSA_PTC_WRITE, uiomove(), and wakeup().

Here is the call graph for this function:

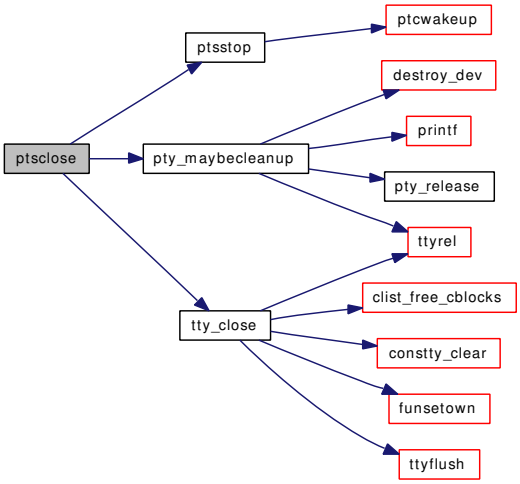


9.134.2.11 static int ptsclose (struct cdev * *dev*, int *flag*, int *mode*, struct thread * *td*) [static]

Definition at line 298 of file tty_pts.c.

References ptsstop(), pty_maybecleanup(), and tty_close().

Here is the call graph for this function:

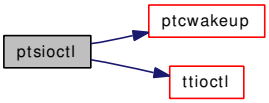


9.134.2.12 static int ptsioctl (struct cdev * *dev*, u_long *cmd*, caddr_t *data*, int *flag*, struct thread * *td*) [static]

Definition at line 761 of file tty_pts.c.

References PF_NOSTOP, PF_PKT, PF_UCNTL, ptcwakeup(), and ttioctl().

Here is the call graph for this function:

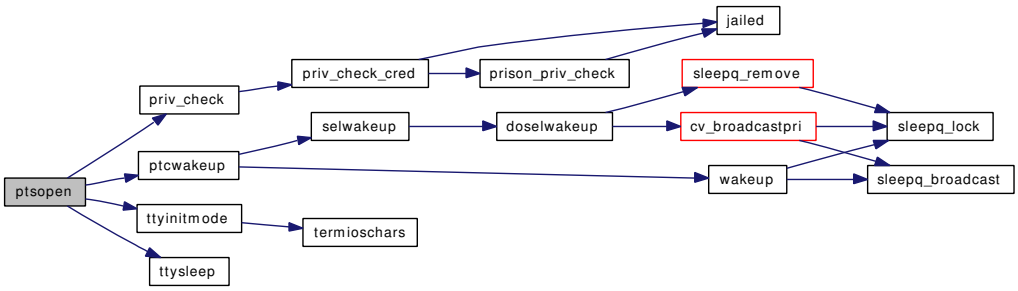


9.134.2.13 static int ptsopen (struct cdev * *dev*, int *flag*, int *devtype*, struct thread * *td*) [static]

Definition at line 262 of file tty_pts.c.

References priv_check(), ptcwakeup(), ttyinitmode(), and ttysleep().

Here is the call graph for this function:

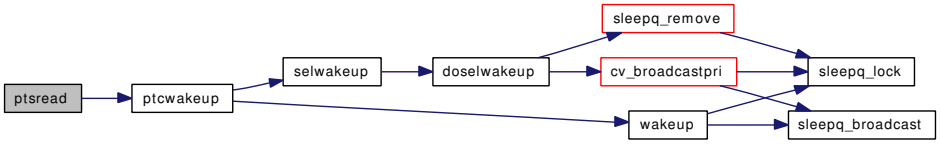


9.134.2.14 static int ptsread (struct cdev * *dev*, struct uio * *uio*, int *flag*) [static]

Definition at line 314 of file tty_pts.c.

References ptcwakeup().

Here is the call graph for this function:



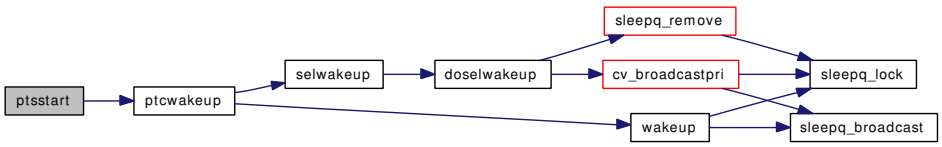
9.134.2.15 static void ptsstart (struct tty * *tp*) [static]

Definition at line 346 of file tty_pts.c.

References PF_STOPPED, and ptcwakeup().

Referenced by ptcopen().

Here is the call graph for this function:



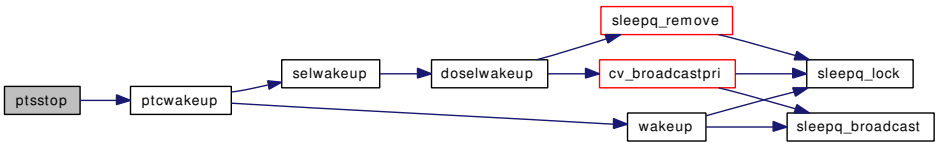
9.134.2.16 static void ptsstop (struct tty * *tp*, int *rw*) [static]

Definition at line 523 of file tty_pts.c.

References PF_STOPPED, and ptcwakeup().

Referenced by ptcopen(), and ptsclose().

Here is the call graph for this function:



9.134.2.17 static int ptswrite (struct cdev * *dev*, struct uio * *uio*, int *flag*) [static]

Definition at line 331 of file tty_pts.c.

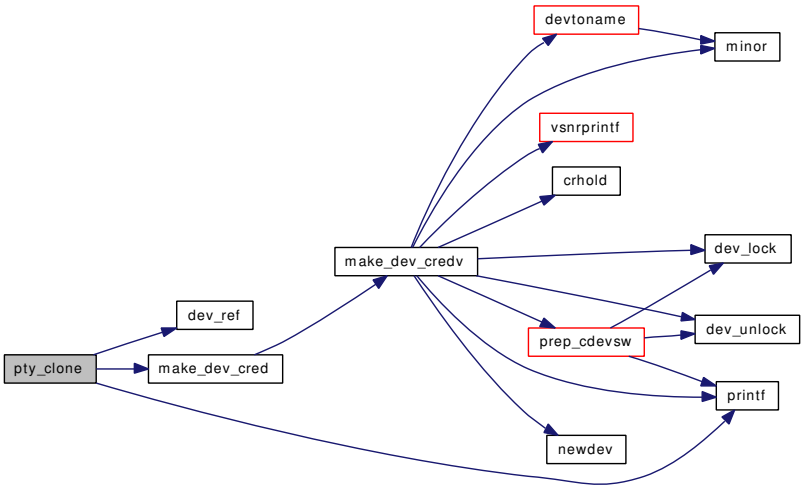
9.134.2.18 static void pty_clone (void * *arg*, struct ucred * *cred*, char * *name*, int *namelen*, struct cdev ** *dev*) [static]

Definition at line 854 of file tty_pts.c.

References bootverbose, dev_ref(), make_dev_cred(), NUM_TO_MINOR, printf(), pt_desc::pt_num, and ptc_cdevsw.

Referenced by ptc_drvinit(), and pty_drvinit().

Here is the call graph for this function:

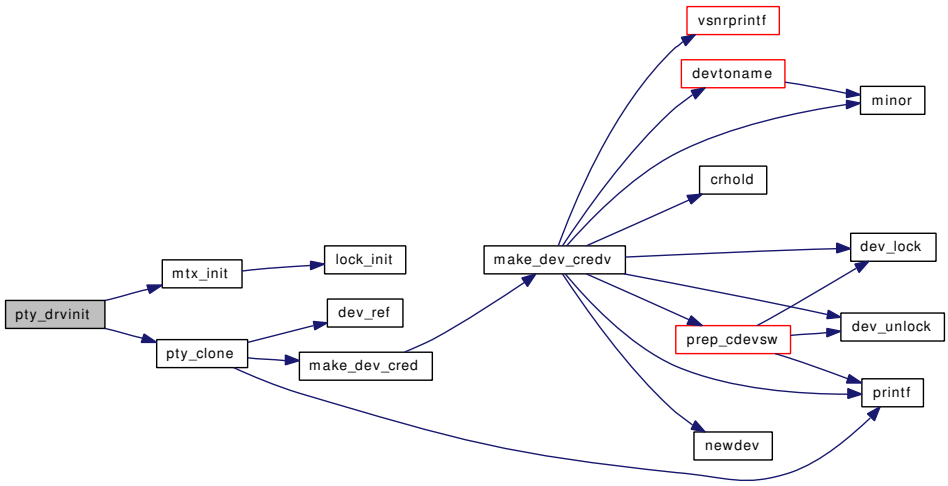


9.134.2.19 static void pty_drvinit (void * *unused*) [static]

Definition at line 901 of file tty_pts.c.

References mtx_init(), and pty_clone().

Here is the call graph for this function:



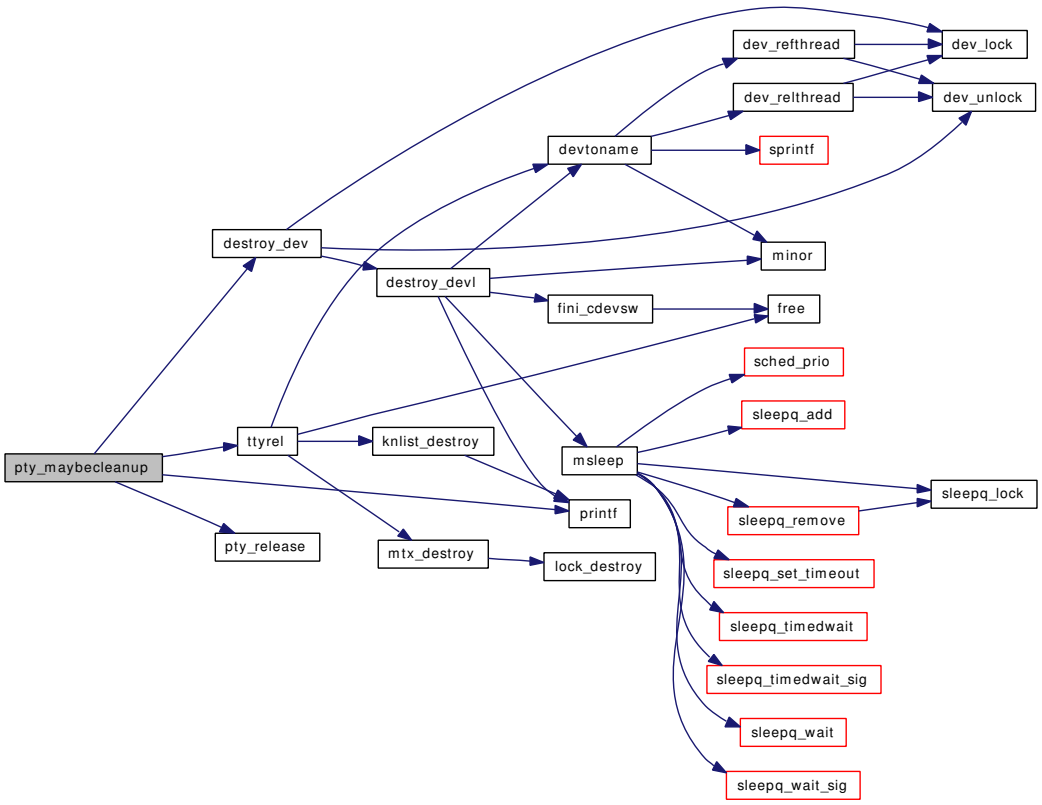
9.134.2.20 static void pty_maybecleanup (struct pt_desc * *pt*) [static]

Definition at line 236 of file tty_pts.c.

References bootverbose, destroy_dev(), printf(), pt_desc::pt_num, pty_release(), and ttyrel().

Referenced by ptcclose(), and ptsclose().

Here is the call graph for this function:



9.134.2.21 static void pty_release (struct pt_desc * *pt*) [static]

Definition at line 217 of file tty_pts.c.

References pt_desc::pt_num.

Referenced by pty_maybecleanup().

9.134.3 Variable Documentation

9.134.3.1 struct mtx pt_mtx [static]

Definition at line 151 of file tty_pts.c.

9.134.3.2 struct cdevsw ptc_cdevsw [static]

Initial value:

```
{
        .d_version =    D_VERSION,
        .d_open =       ptcopen,
        .d_close =      ptcclose,
        .d_read =       ptcread,
        .d_write =      ptcwrite,
```

```
        .d_ioctl =      ptcioctl,
        .d_poll =       ptcpoll,
        .d_name =       "ptc",
        .d_flags =      D_TTY | D_NEEDGIANT,
        .d_kqfilter =   ttykqfilter,
}
```

Definition at line 104 of file tty_pts.c.

Referenced by pty_clone().

9.134.3.3 d_close_t ptcclose [static]

Definition at line 86 of file tty_pts.c.

9.134.3.4 d_ioctl_t ptcioctl [static]

Definition at line 84 of file tty_pts.c.

9.134.3.5 d_open_t ptcopen [static]

Definition at line 85 of file tty_pts.c.

9.134.3.6 d_poll_t ptcpoll [static]

Definition at line 89 of file tty_pts.c.

9.134.3.7 d_read_t ptcread [static]

Definition at line 87 of file tty_pts.c.

9.134.3.8 d_write_t ptcwrite [static]

Definition at line 88 of file tty_pts.c.

9.134.3.9 struct cdevsw pts_cdevsw [static]

Initial value:

```
 {
        .d_version =    D_VERSION,
        .d_open =       ptsopen,
        .d_close =      ptsclose,
        .d_read =       ptsread,
        .d_write =      ptswrite,
        .d_ioctl =      ptsioctl,
        .d_poll =       ttypoll,
        .d_name =       "pts",
        .d_flags =      D_TTY | D_NEEDGIANT,
        .d_kqfilter =   ttykqfilter,
}
```

Definition at line 91 of file tty_pts.c.

Referenced by ptcopen(), and pty_create_slave().

9.134.3.10 d_close_t ptsclose [static]

Definition at line 80 of file tty_pts.c.

9.134.3.11 d_ioctl_t ptsioctl [static]

Definition at line 83 of file tty_pts.c.

Referenced by ptcioctl().

9.134.3.12 d_open_t ptsopen [static]

Definition at line 79 of file tty_pts.c.

9.134.3.13 d_read_t ptsread [static]

Definition at line 81 of file tty_pts.c.

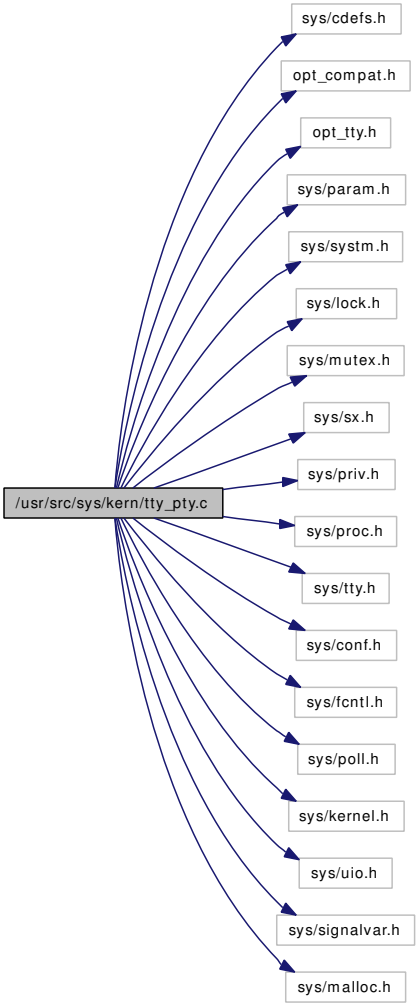
9.134.3.14 d_write_t ptswrite [static]

Definition at line 82 of file tty_pts.c.

9.135 /usr/src/sys/kern/tty_pty.c File Reference

```
#include <sys/cdefs.h>
#include "opt_compat.h"
#include "opt_tty.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/sx.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/tty.h>
#include <sys/conf.h>
#include <sys/fcntl.h>
#include <sys/poll.h>
#include <sys/kernel.h>
#include <sys/uio.h>
#include <sys/signalvar.h>
#include <sys/malloc.h>
```

Include dependency graph for tty_pty.c:



Data Structures

- struct ptsc

Defines

- #define BUFSIZ 100
- #define PF_PKT 0x08
- #define PF_STOPPED 0x10
- #define PF_NOSTOP 0x40
- #define PF_UCNTL 0x80
- #define TSA_PTC_READ(tp) ((void *)&(tp) → t_outq.c_cf)
- #define TSA_PTC_WRITE(tp) ((void *)&(tp) → t_rawq.c_cl)
- #define TSA_PTS_READ(tp) ((void *)&(tp) → t_canq)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/tty_pty.c,v 1.151 2006/11/06 13:42:01 rwatson Exp $")

- static MALLOC_DEFINE (M_PTY,"ptys","pty data structures")
- static void ptsstart (struct tty *tp)
- static void ptsstop (struct tty *tp, int rw)
- static void ptcwakeup (struct tty *tp, int flag)
- static struct cdev * ptyinit (struct cdev *cdev, struct thread *td)
- static void pty_create_slave (struct ucred *cred, struct ptsc *pt, int n)
- static void pty_destroy_slave (struct ptsc *pt)
- static void pty_maybe_destroy_slave (struct ptsc *pt)
- static int ptsopen (struct cdev *dev, int flag, int devtype, struct thread *td)
- static int ptsclose (struct cdev *dev, int flag, int mode, struct thread *td)
- static int ptsread (struct cdev *dev, struct uio *uio, int flag)
- static int ptswrite (struct cdev *dev, struct uio *uio, int flag)
- static int ptcopen (struct cdev *dev, int flag, int devtype, struct thread *td)
- static int ptcclose (struct cdev *dev, int flags, int fmt, struct thread *td)
- static int ptcread (struct cdev *dev, struct uio *uio, int flag)
- static int ptcpoll (struct cdev *dev, int events, struct thread *td)
- static int ptcwrite (struct cdev *dev, struct uio *uio, int flag)
- static int ptcioctl (struct cdev *dev, u_long cmd, caddr_t data, int flag, struct thread *td)
- static int ptsioctl (struct cdev *dev, u_long cmd, caddr_t data, int flag, struct thread *td)
- static void pty_clone (void *arg, struct ucred *cr, char *name, int namelen, struct cdev **dev)
- static void ptc_drvinit (void *unused)

Variables

- static d_open_t ptsopen
- static d_close_t ptsclose
- static d_read_t ptsread
- static d_write_t ptswrite
- static d_ioctl_t ptsioctl
- static d_open_t ptcopen
- static d_close_t ptcclose
- static d_read_t ptcread
- static d_ioctl_t ptcioctl
- static d_write_t ptcwrite
- static d_poll_t ptcpoll
- static struct cdevsw pts_cdevsw
- static struct cdevsw ptc_cdevsw
- static char * names = "pqrsPQRS"

9.135.1 Define Documentation

9.135.1.1 #define BUFSIZ 100

Definition at line 102 of file tty_pty.c.

9.135.1.2 #define PF_NOSTOP 0x40

Definition at line 117 of file tty_pty.c.

9.135.1.3 #define PF_PKT 0x08

Definition at line 115 of file tty_pty.c.

9.135.1.4 #define PF_STOPPED 0x10

Definition at line 116 of file tty_pty.c.

9.135.1.5 #define PF_UCNTL 0x80

Definition at line 118 of file tty_pty.c.

9.135.1.6 #define TSA_PTC_READ(tp) ((void ∗)&(tp) → t_outq.c_cf)

Definition at line 120 of file tty_pty.c.

9.135.1.7 #define TSA_PTC_WRITE(tp) ((void ∗)&(tp) → t_rawq.c_cl)

Definition at line 121 of file tty_pty.c.

9.135.1.8 #define TSA_PTS_READ(tp) ((void ∗)&(tp) → t_canq)

Definition at line 122 of file tty_pty.c.

9.135.2 Function Documentation

9.135.2.1 __FBSDID ("$FreeBSD: src/sys/kern/tty_pty. *c*, v 1.151 2006/11/06 13:42:01 rwatson Exp $")

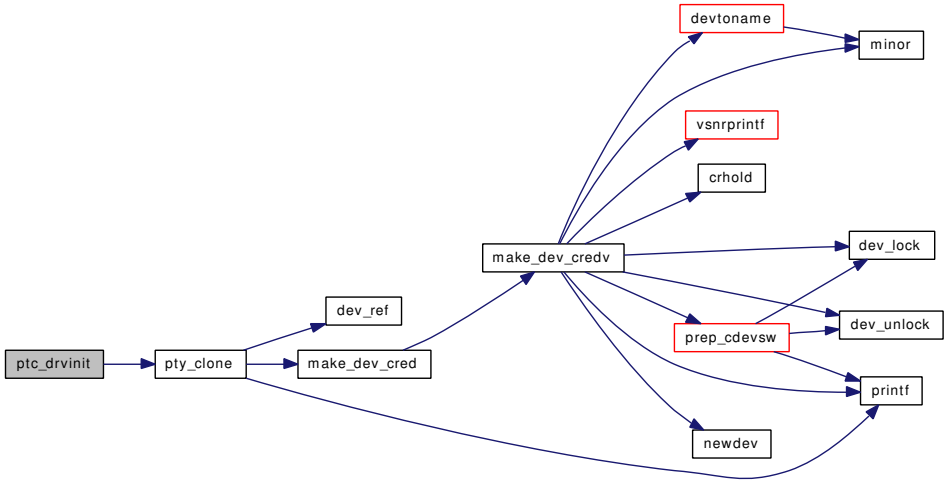
9.135.2.2 static MALLOC_DEFINE (M_PTY, "ptys", "pty data structures") [static]

9.135.2.3 static void ptc_drvinit (void ∗ *unused*) [static]

Definition at line 796 of file tty_pty.c.

References pty_clone().

Here is the call graph for this function:

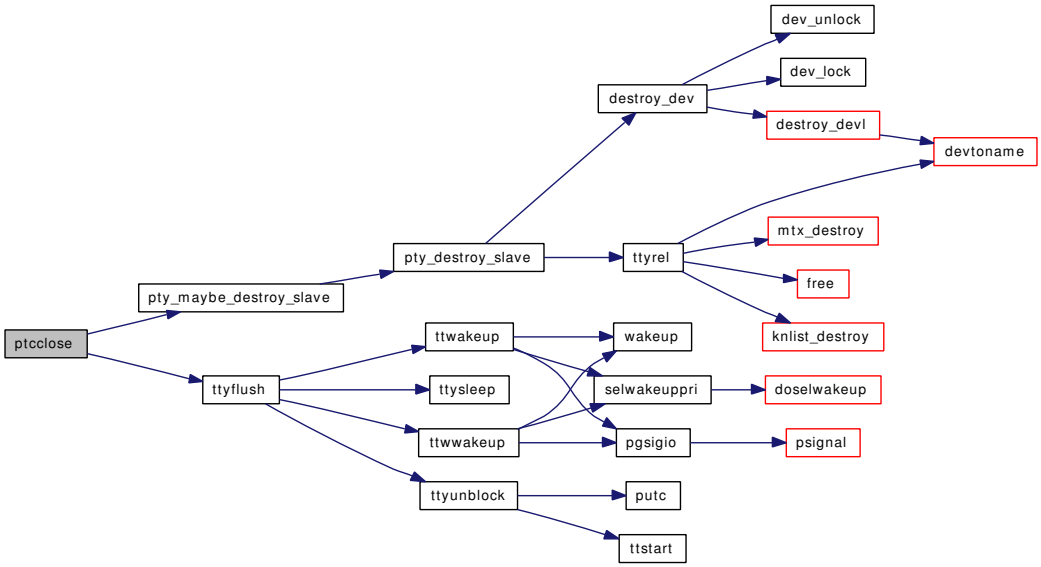


9.135.2.4 static int ptcclose (struct cdev * *dev*, int *flags*, int *fmt*, struct thread * *td*) [static]

Definition at line 361 of file tty_pty.c.

References ptsc::pt_devc_open, pty_maybe_destroy_slave(), and ttyflush().

Here is the call graph for this function:

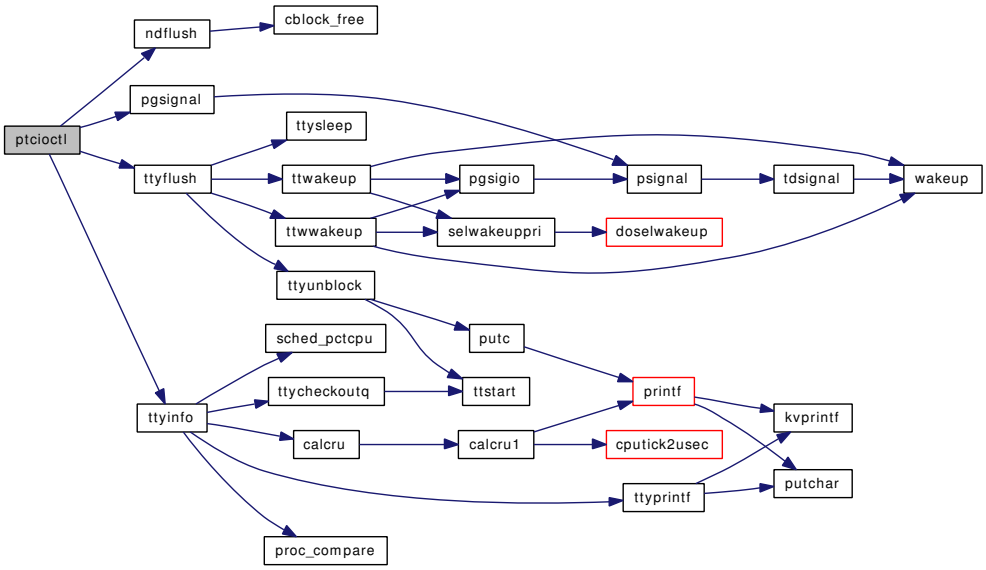


9.135.2.5 static int ptcioctl (struct cdev * *dev*, u_long *cmd*, caddr_t *data*, int *flag*, struct thread * *td*) [static]

Definition at line 583 of file tty_pty.c.

References ndflush(), PF_PKT, PF_UCNTL, pgsignal(), ptsc::pt_flags, ptsioctl, ttyflush(), and ttyinfo().

Here is the call graph for this function:

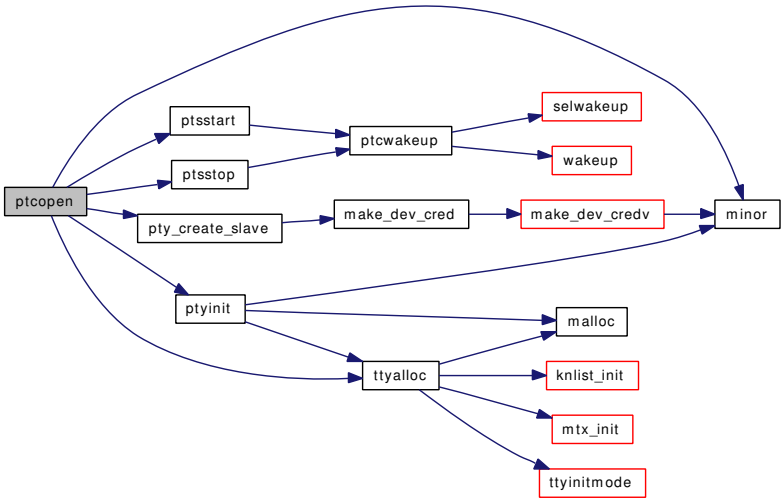


9.135.2.6 static int ptcopen (struct cdev * *dev*, int *flag*, int *devtype*, struct thread * *td*) [static]

Definition at line 319 of file tty_pty.c.

References ptsc::devs, minor(), ptsc::pt_devc_open, ptsc::pt_flags, ptsc::pt_prison, ptsc::pt_send, ptsc::pt_tty, ptsc::pt_ucntl, ptsstart(), ptsstop(), pty_create_slave(), ptyinit(), and ttyalloc().

Here is the call graph for this function:



9.135.2.7 static int ptcpoll (struct cdev * *dev*, int *events*, struct thread * *td*) [static]

Definition at line 470 of file tty_pty.c.

References PF_PKT, PF_UCNTL, ptsc::pt_flags, ptsc::pt_selw, ptsc::pt_send, ptsc::pt_ucntl, and selrecord().

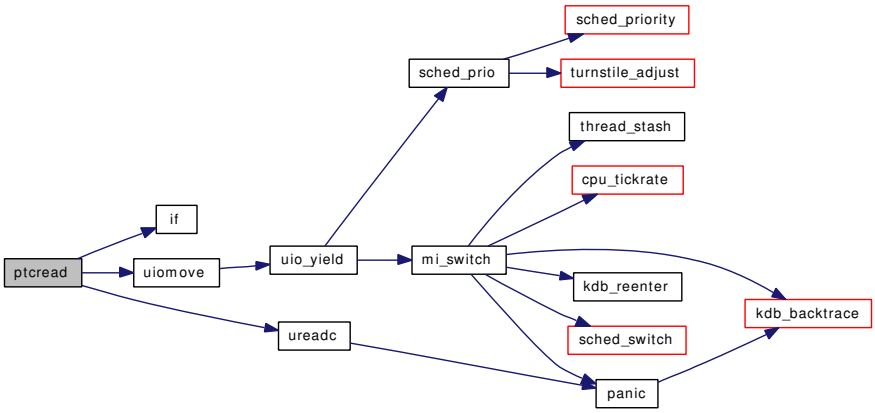
Here is the call graph for this function:

9.135.2.8 static int ptcread (struct cdev * *dev*, struct uio * *uio*, int *flag*) [static]

Definition at line 390 of file tty_pty.c.

References buf, BUFSIZ, if(), PF_PKT, ptsc::pt_flags, ptsc::pt_send, uiomove(), and ureadc().

Here is the call graph for this function:

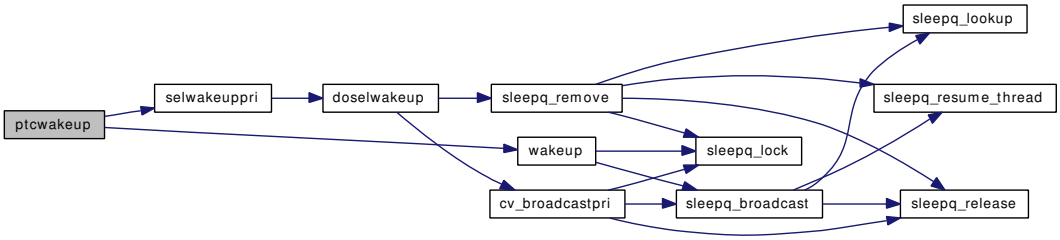


9.135.2.9 static void ptcwakeup (struct tty * *tp*, int *flag*) [static]

Definition at line 304 of file tty_pty.c.

References ptsc::pt_selw, selwakeuppri(), TSA_PTC_READ, TSA_PTC_WRITE, and wakeup().

Here is the call graph for this function:

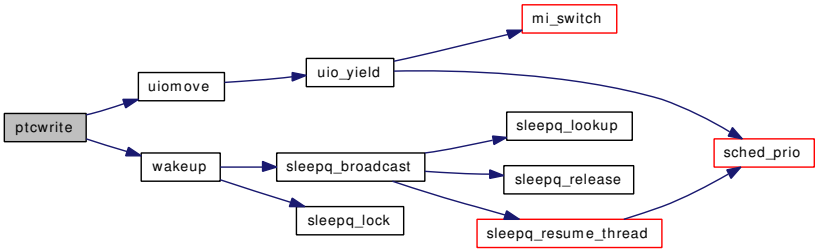


9.135.2.10 static int ptcwrite (struct cdev * *dev*, struct uio * *uio*, int *flag*) [static]

Definition at line 516 of file tty_pty.c.

References BUFSIZ, TSA_PTC_WRITE, uiomove(), and wakeup().

Here is the call graph for this function:

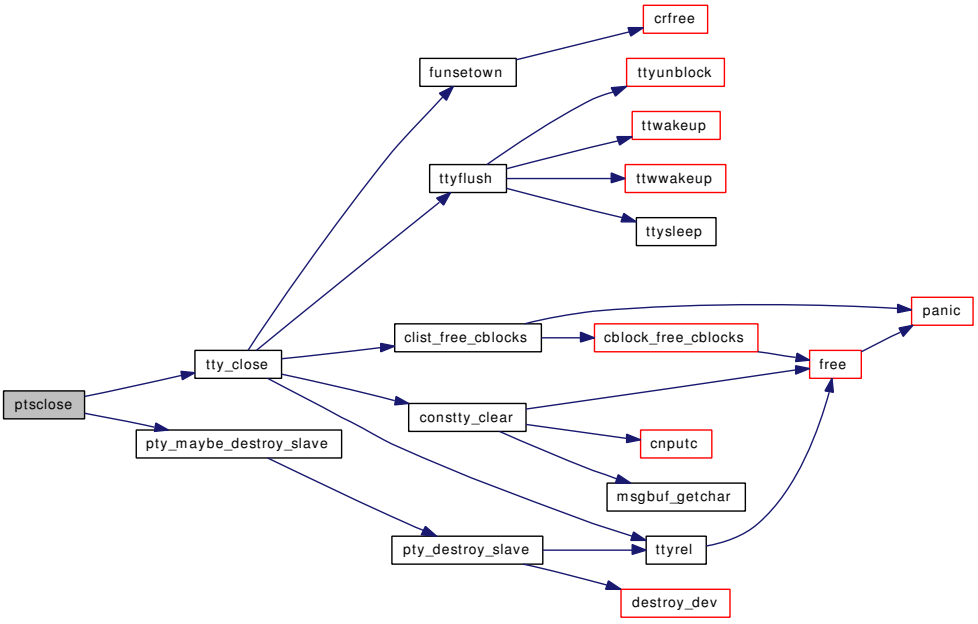


9.135.2.11 static int ptsclose (struct cdev * *dev*, int *flag*, int *mode*, struct thread * *td*) [static]

Definition at line 237 of file tty_pty.c.

References ptsc::devs, ptsc::pt_devs_open, pty_maybe_destroy_slave(), and tty_close().

Here is the call graph for this function:

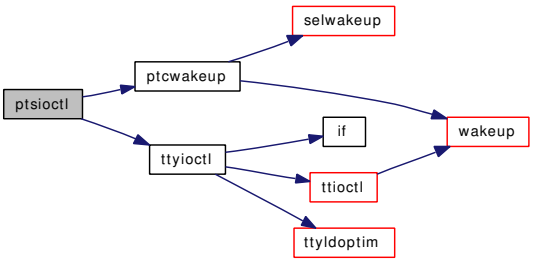


9.135.2.12 static int ptsioctl (struct cdev * *dev*, u_long *cmd*, caddr_t *data*, int *flag*, struct thread * *td*) [static]

Definition at line 674 of file tty_pty.c.

References PF_NOSTOP, PF_PKT, PF_UCNTL, ptsc::pt_flags, ptsc::pt_send, ptsc::pt_ucntl, ptcwakeup(), and ttyioctl().

Here is the call graph for this function:

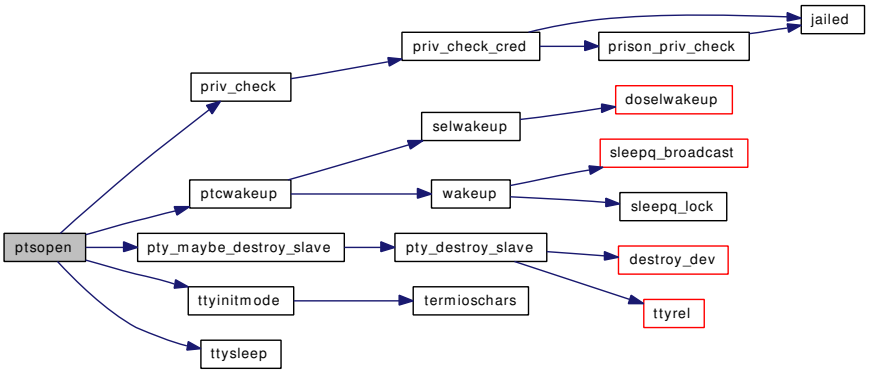


9.135.2.13 static int ptsopen (struct cdev * *dev*, int *flag*, int *devtype*, struct thread * *td*) [static]

Definition at line 198 of file tty_pty.c.

References priv_check(), ptsc::pt_devs_open, ptsc::pt_prison, ptcwakeup(), pty_maybe_destroy_slave(), ttyinitmode(), and ttysleep().

Here is the call graph for this function:

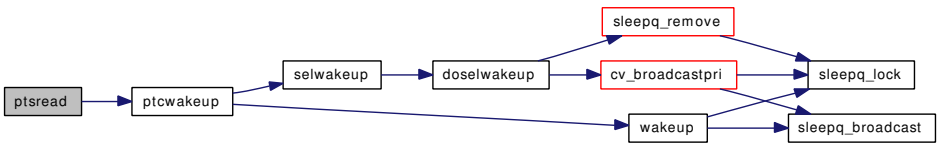


9.135.2.14 static int ptsread (struct cdev * *dev*, struct uio * *uio*, int *flag*) [static]

Definition at line 258 of file tty_pty.c.

References ptcwakeup().

Here is the call graph for this function:

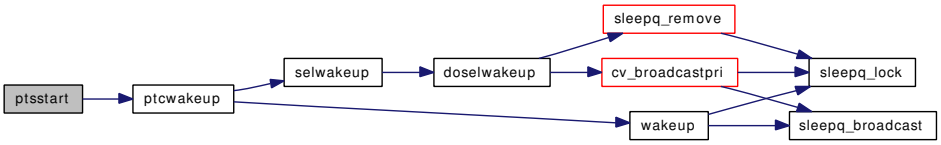


9.135.2.15 static void ptsstart (struct tty * *tp*) [static]

Definition at line 290 of file tty_pty.c.

References PF_STOPPED, ptsc::pt_flags, ptsc::pt_send, and ptcwakeup().

Here is the call graph for this function:

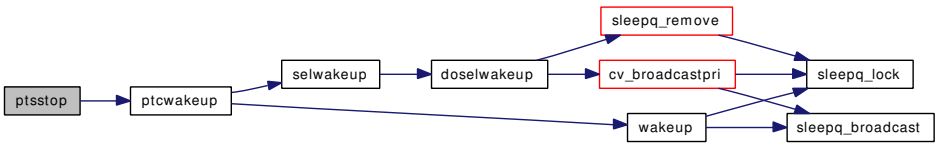


9.135.2.16 static void ptsstop (struct tty * *tp*, int *rw*) [static]

Definition at line 448 of file tty_pty.c.

References PF_STOPPED, ptsc::pt_flags, ptsc::pt_send, and ptcwakeup().

Here is the call graph for this function:



9.135.2.17 static int ptswrite (struct cdev * *dev*, struct uio * *uio*, int *flag*) [static]

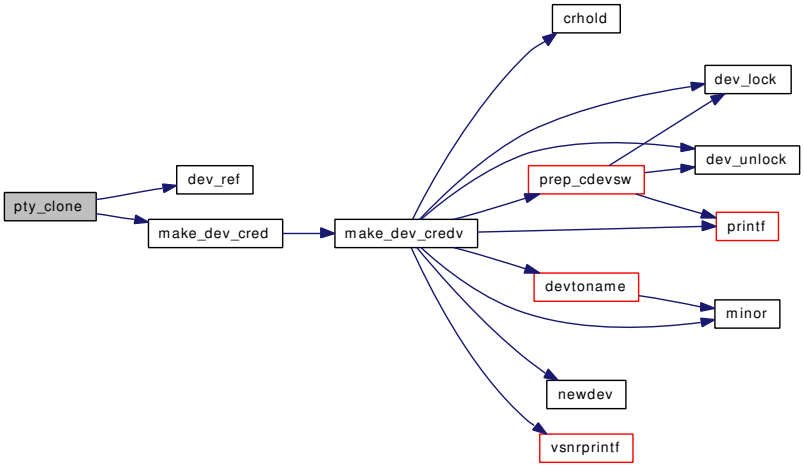
Definition at line 275 of file tty_pty.c.

9.135.2.18 static void pty_clone (void * *arg*, struct ucred * *cr*, char * *name*, int *namelen*, struct cdev ** *dev*) [static]

Definition at line 760 of file tty_pty.c.

References dev_ref(), make_dev_cred(), names, and ptc_cdevsw.

Here is the call graph for this function:



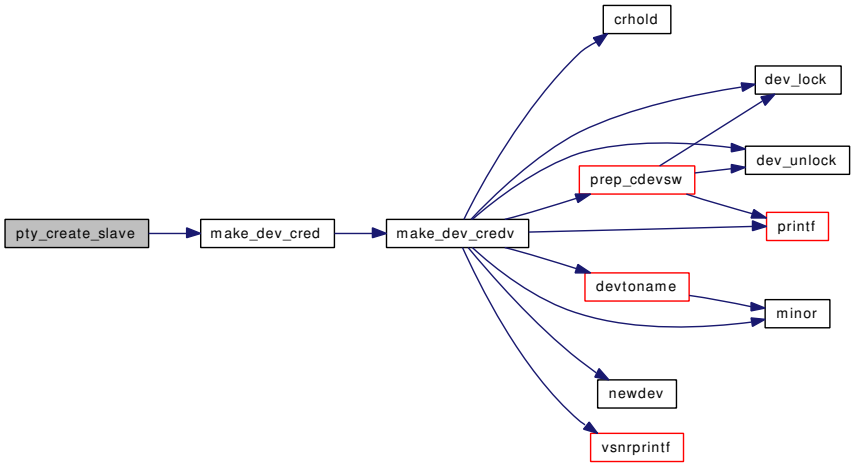
9.135.2.19 static void pty_create_slave (struct ucred * *cred*, struct ptsc * *pt*, int *n*) [static]

Definition at line 161 of file tty_pty.c.

References ptsc::devs, make_dev_cred(), names, ptsc::pt_tty, and pts_cdevsw.

Referenced by ptcopen().

Here is the call graph for this function:



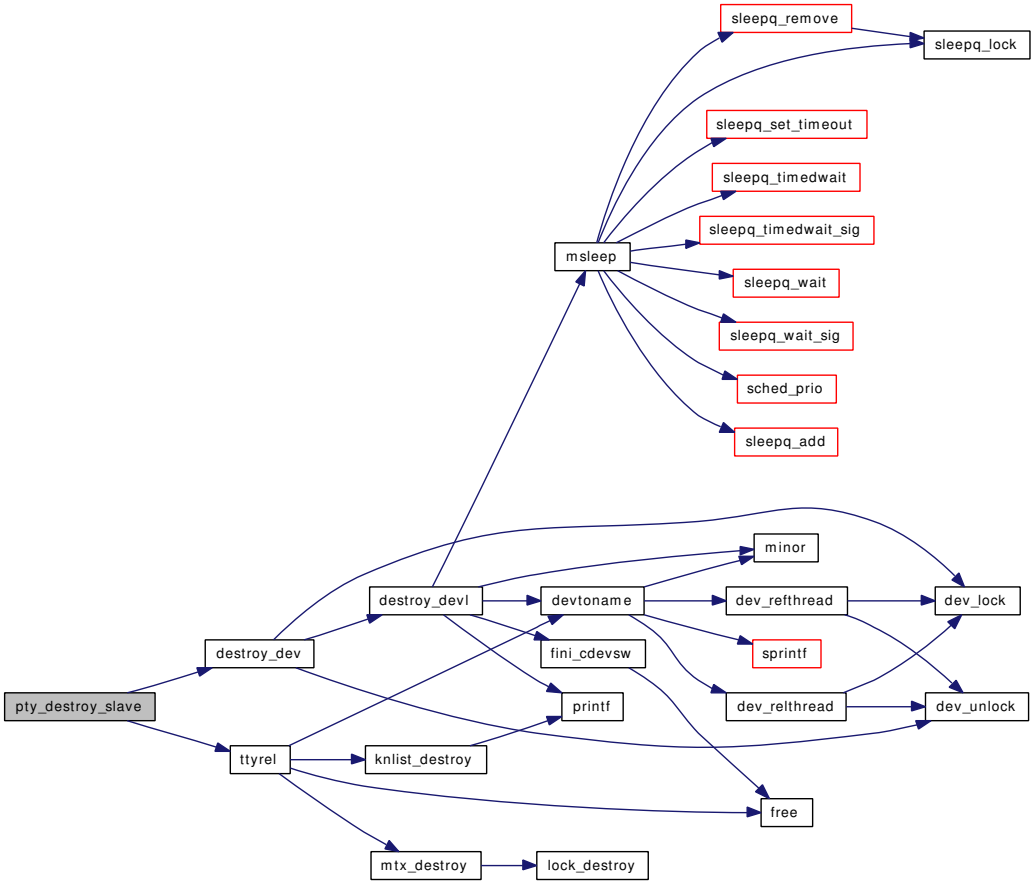
9.135.2.20 static void pty_destroy_slave (struct ptsc * *pt*) [static]

Definition at line 172 of file tty_pty.c.

References destroy_dev(), ptsc::devs, ptsc::pt_tty, and ttyrel().

Referenced by pty_maybe_destroy_slave().

Here is the call graph for this function:



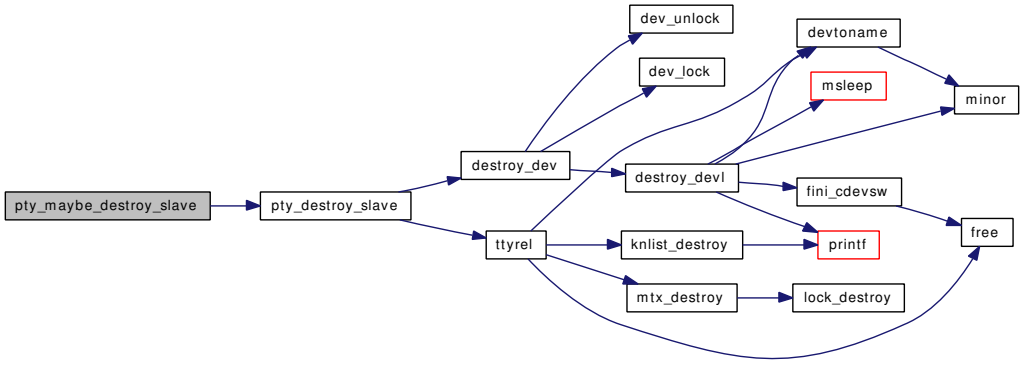
9.135.2.21 static void pty_maybe_destroy_slave (struct ptsc * *pt*) [static]

Definition at line 185 of file tty_pty.c.

References ptsc::pt_devc_open, ptsc::pt_devs_open, and pty_destroy_slave().

Referenced by ptcclose(), ptsclose(), and ptsopen().

Here is the call graph for this function:



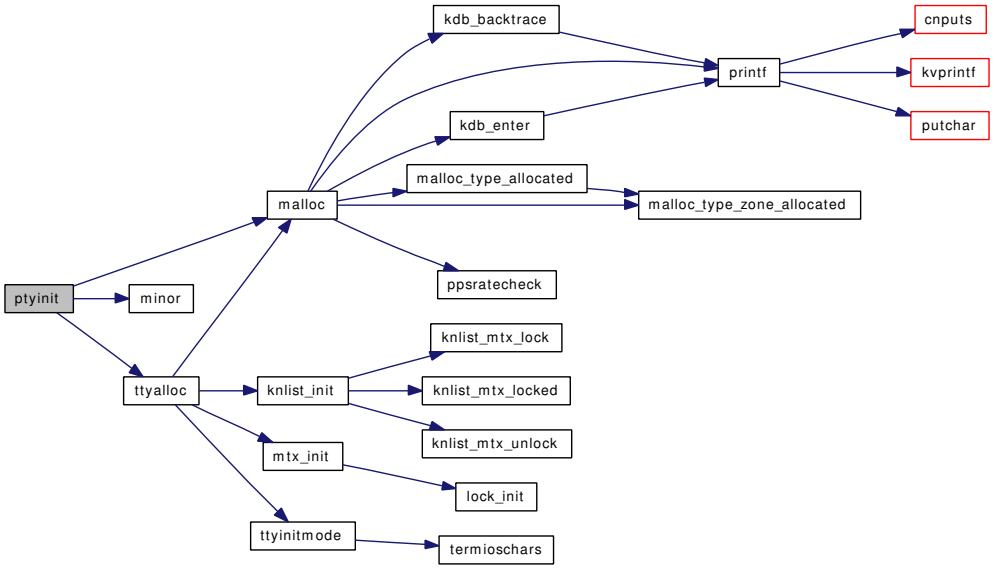
9.135.2.22 static struct cdev * ptyinit (struct cdev * *cdev*, struct thread * *td*) [static]

Definition at line 135 of file tty_pty.c.

References malloc(), minor(), and ttyalloc().

Referenced by ptcopen().

Here is the call graph for this function:



9.135.3 Variable Documentation

9.135.3.1 char* names = "pqrsPQRS" [static]

Definition at line 124 of file tty_pty.c.

Referenced by pty_clone(), and pty_create_slave().

9.135.3.2 struct cdevsw ptc_cdevsw [static]

Initial value:

```
{
        .d_version =    D_VERSION,
        .d_open =       ptcopen,
        .d_close =      ptcclose,
        .d_read =       ptcread,
        .d_write =      ptcwrite,
        .d_ioctl =      ptcioctl,
        .d_poll =       ptcpoll,
        .d_name =       "ptc",
        .d_flags =      D_TTY | D_NEEDGIANT,
}
```

Definition at line 90 of file tty_pty.c.

9.135.3.3 d_close_t ptcclose [static]

Definition at line 73 of file tty_pty.c.

9.135.3.4 d_ioctl_t ptcioctl [static]

Definition at line 75 of file tty_pty.c.

9.135.3.5 d_open_t ptcopen [static]

Definition at line 72 of file tty_pty.c.

9.135.3.6 d_poll_t ptcpoll [static]

Definition at line 77 of file tty_pty.c.

9.135.3.7 d_read_t ptcread [static]

Definition at line 74 of file tty_pty.c.

9.135.3.8 d_write_t ptcwrite [static]

Definition at line 76 of file tty_pty.c.

9.135.3.9 struct cdevsw pts_cdevsw [static]

Initial value:

```
 {
        .d_version =    D_VERSION,
        .d_open =       ptsopen,
        .d_close =      ptsclose,
        .d_read =       ptsread,
        .d_write =      ptswrite,
        .d_ioctl =      ptsioctl,
        .d_name =       "pts",
        .d_flags =      D_TTY | D_NEEDGIANT,
}
```

Definition at line 79 of file tty_pty.c.

9.135.3.10 d_close_t ptsclose [static]

Definition at line 68 of file tty_pty.c.

9.135.3.11 d_ioctl_t ptsioctl [static]

Definition at line 71 of file tty_pty.c.

9.135.3.12 d_open_t ptsopen [static]

Definition at line 67 of file tty_pty.c.

9.135.3.13 d_read_t ptsread [static]

Definition at line 69 of file tty_pty.c.

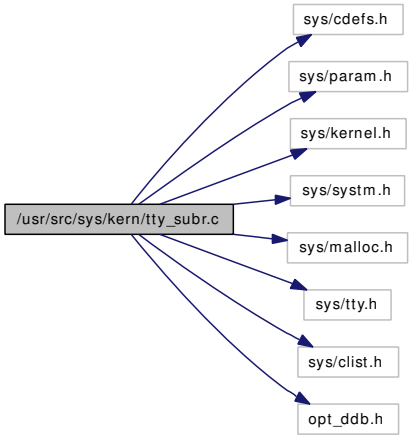
9.135.3.14 d_write_t ptswrite [static]

Definition at line 70 of file tty_pty.c.

9.136 /usr/src/sys/kern/tty_subr.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/kernel.h>
#include <sys/systm.h>
#include <sys/malloc.h>
#include <sys/tty.h>
#include <sys/clist.h>
#include "opt_ddb.h"
```

Include dependency graph for tty_subr.c:



Defines

- #define INITIAL_CBLOCKS 50

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/tty_subr.c,v 1.43 2005/01/06 23:35:40 imp Exp $")
- static void clist_init (void *)
- static struct cblock * cblock_alloc (void)
- static void cblock_alloc_cblocks (int number)
- static void cblock_free (struct cblock *cblockp)
- static void cblock_free_cblocks (int number)
- void clist_alloc_cblocks (struct clist *clistp, int ccmax, int ccreserved)
- void clist_free_cblocks (struct clist *clistp)
- int getc (struct clist *clistp)
- int q_to_b (struct clist *clistp, char *dest, int amount)
- void ndflush (struct clist *clistp, int amount)
- int putc (int chr, struct clist *clistp)
- int b_to_q (char *src, int amount, struct clist *clistp)

- char * nextc (struct clist *clistp, char *cp, int *dst)
- int unputc (struct clist *clistp)
- void catq (struct clist *src_clistp, struct clist *dest_clistp)

Variables

- static struct cblock * cfreelist = 0
- int cfreecount = 0
- static int cslushcount
- static int ctotcount

9.136.1 Define Documentation

9.136.1.1 #define INITIAL_CBLOCKS 50

Definition at line 51 of file tty_subr.c.

Referenced by clist_init().

9.136.2 Function Documentation

9.136.2.1 __FBSDID ("$FreeBSD: src/sys/kern/tty_subr. *c*, v 1.43 2005/01/06 23:35:40 imp Exp $")

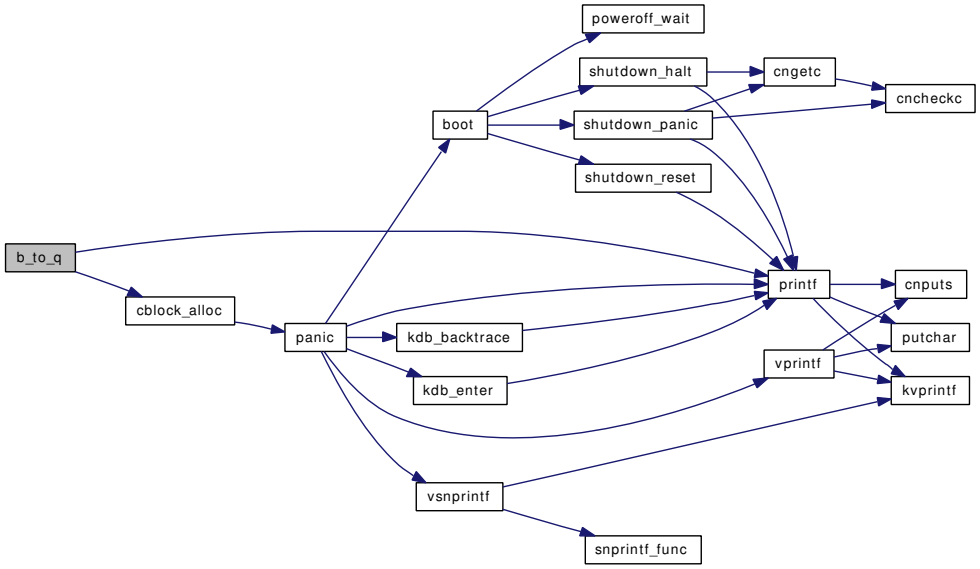
9.136.2.2 int b_to_q (char * *src*, int *amount*, struct clist * *clistp*)

Definition at line 430 of file tty_subr.c.

References cblock_alloc(), and printf().

Referenced by ttwrite(), and ttyoutput().

Here is the call graph for this function:



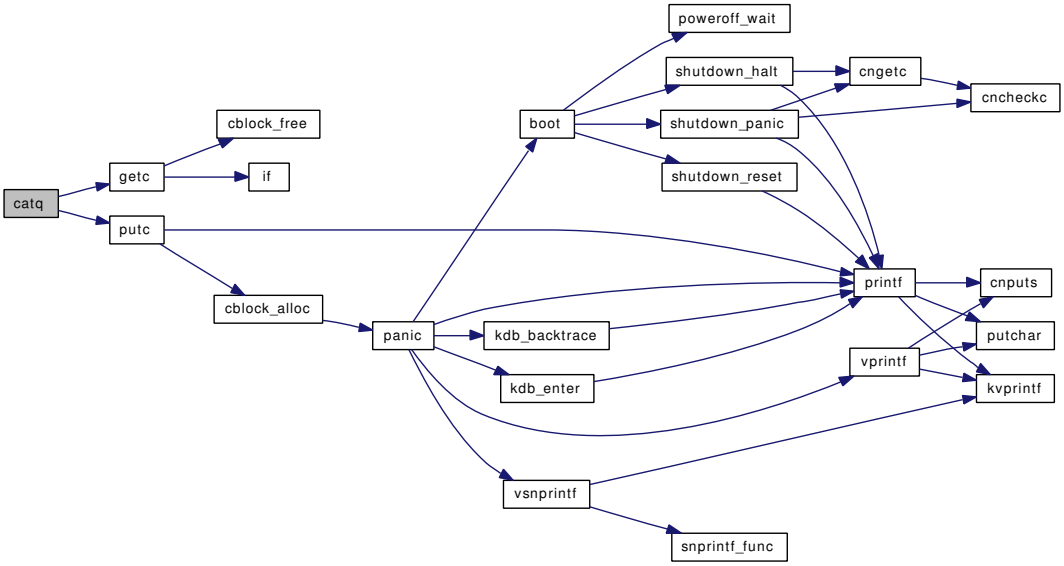
9.136.2.3 void catq (struct clist * *src_clistp*, struct clist * *dest_clistp*)

Definition at line 662 of file tty_subr.c.

References getc(), and putc().

Referenced by ttioctl(), and ttyinput().

Here is the call graph for this function:



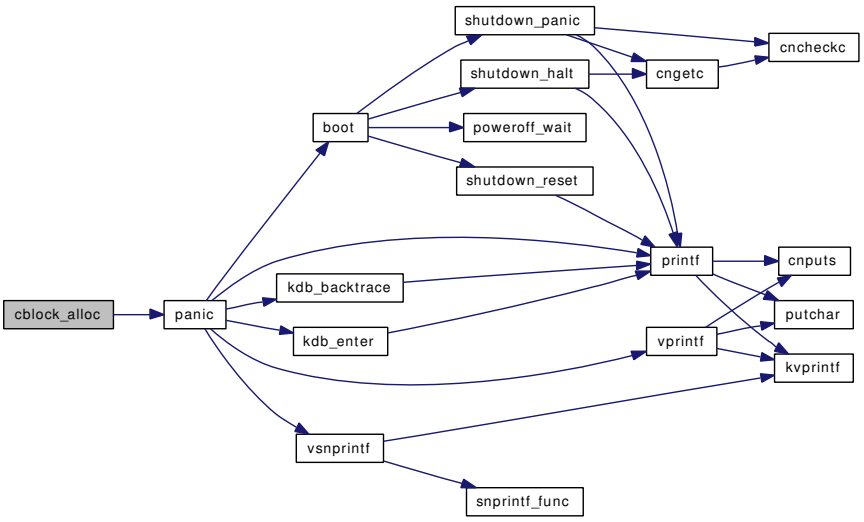
9.136.2.4 static __inline struct cblock * cblock_alloc (void) [static]

Definition at line 98 of file tty_subr.c.

References panic().

Referenced by b_to_q(), cblock_free_cblocks(), and putc().

Here is the call graph for this function:



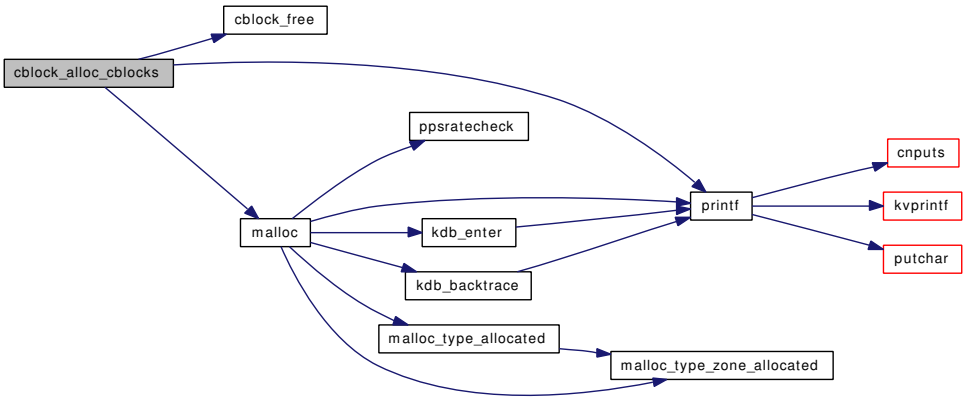
9.136.2.5 static void cblock_alloc_cblocks (int *number*) [static]

Definition at line 129 of file tty_subr.c.

References cblock_free(), malloc(), and printf().

Referenced by clist_alloc_cblocks(), and clist_init().

Here is the call graph for this function:



9.136.2.6 static __inline void cblock_free (struct cblock * *cblockp*) [static]

Definition at line 115 of file tty_subr.c.

Referenced by cblock_alloc_cblocks(), getc(), ndflush(), q_to_b(), and unputc().

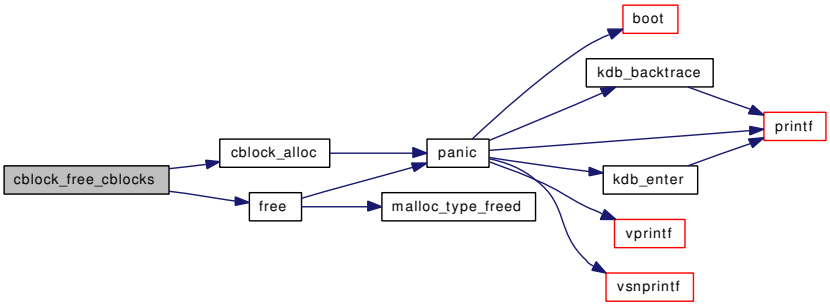
9.136.2.7 static void cblock_free_cblocks (int *number*) [static]

Definition at line 189 of file tty_subr.c.

References cblock_alloc(), and free().

Referenced by clist_alloc_cblocks(), and clist_free_cblocks().

Here is the call graph for this function:



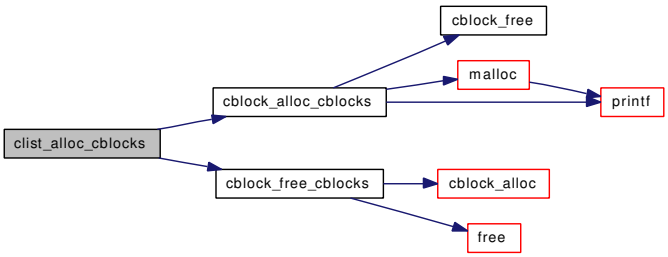
9.136.2.8 void clist_alloc_cblocks (struct clist * *clistp*, int *ccmax*, int *ccreserved*)

Definition at line 157 of file tty_subr.c.

References cblock_alloc_cblocks(), and cblock_free_cblocks().

Referenced by ttsetwater().

Here is the call graph for this function:



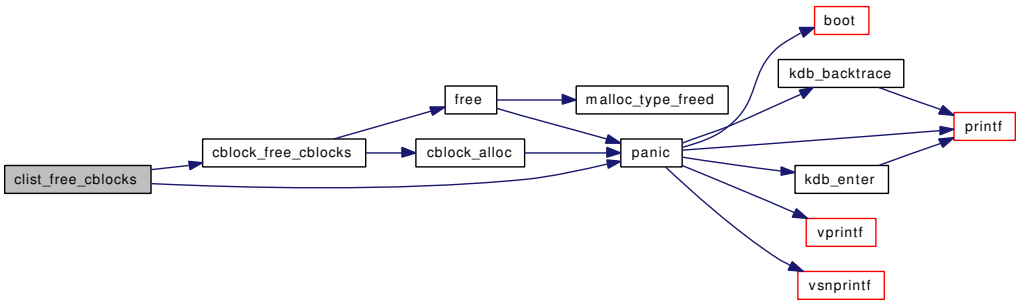
9.136.2.9 void clist_free_cblocks (struct clist * *clistp*)

Definition at line 204 of file tty_subr.c.

References cblock_free_cblocks(), and panic().

Referenced by tty_close().

Here is the call graph for this function:

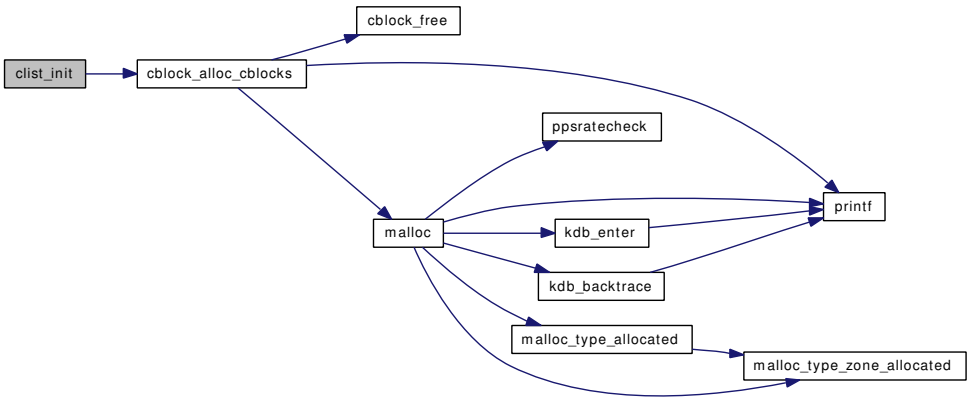


9.136.2.10 static void clist_init (void *) [static]

Definition at line 79 of file tty_subr.c.

References cblock_alloc_cblocks(), and INITIAL_CBLOCKS.

Here is the call graph for this function:



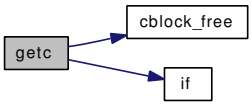
9.136.2.11 int getc (struct clist * *clistp*)

Definition at line 218 of file tty_subr.c.

References cblock_free(), and if().

Referenced by catq(), ttread(), and ttypend().

Here is the call graph for this function:



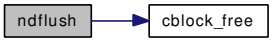
9.136.2.12 void ndflush (struct clist * *clistp*, int *amount*)

Definition at line 321 of file tty_subr.c.

References cblock_free().

Referenced by ptcioctl().

Here is the call graph for this function:



9.136.2.13 char* nextc (struct clist * *clistp*, char * *cp*, int * *dst*)

Definition at line 557 of file tty_subr.c.

Referenced by ttyretype(), and ttyrub().

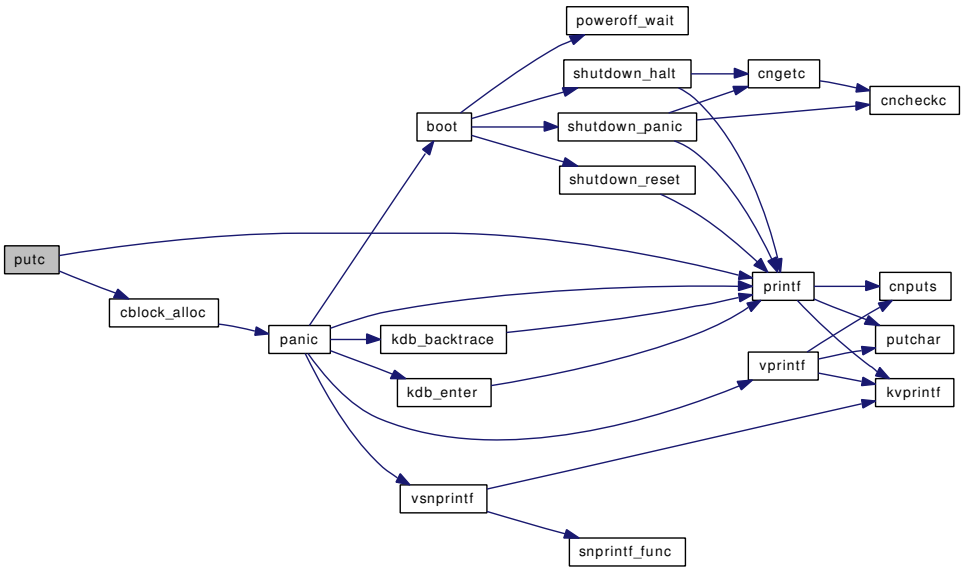
9.136.2.14 int putc (int *chr*, struct clist * *clistp*)

Definition at line 366 of file tty_subr.c.

References cblock_alloc(), and printf().

Referenced by catq(), ttyblock(), ttyinput(), ttyoutput(), and ttyunblock().

Here is the call graph for this function:



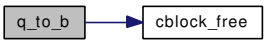
9.136.2.15 int q_to_b (struct clist * *clistp*, char * *dest*, int *amount*)

Definition at line 272 of file tty_subr.c.

References cblock_free().

Referenced by ttread().

Here is the call graph for this function:



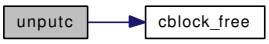
9.136.2.16 int unputc (struct clist * *clistp*)

Definition at line 594 of file tty_subr.c.

References cblock_free().

Referenced by ttyinput().

Here is the call graph for this function:



9.136.3 Variable Documentation

9.136.3.1 int cfreecount = 0

Definition at line 46 of file tty_subr.c.

9.136.3.2 struct cblock* cfreelist = 0 [static]

Definition at line 45 of file tty_subr.c.

9.136.3.3 int cslushcount [static]

Definition at line 47 of file tty_subr.c.

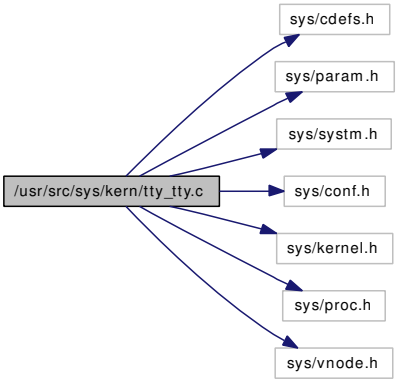
9.136.3.4 int ctotcount [static]

Definition at line 48 of file tty_subr.c.

9.137 /usr/src/sys/kern/tty_tty.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/conf.h>
#include <sys/kernel.h>
#include <sys/proc.h>
#include <sys/vnode.h>
```

Include dependency graph for tty_tty.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/tty_tty.c,v 1.59 2006/09/27 16:41:15 mbr Exp $")
- static int cttyopen (struct cdev *dev, int flag, int mode, struct thread *td)
- static void ctty_clone (void *arg, struct ucred *cred, char *name, int namelen, struct cdev **dev)
- static void ctty_drvinit (void *unused)

Variables

- static d_open_t cttyopen
- static struct cdevsw ctty_cdevsw
- static struct cdev * ctty

9.137.1 Function Documentation

9.137.1.1 __FBSDID ("$FreeBSD: src/sys/kern/tty_tty. *c*, v 1.59 2006/09/27 16:41:15 mbr Exp $")

9.137.1.2 static void ctty_clone (void * *arg*, struct ucred * *cred*, char * *name*, int *namelen*, struct cdev ** *dev*) [static]

Definition at line 55 of file tty_tty.c.

References ctty, and dev_ref().

Referenced by ctty_drvinit().

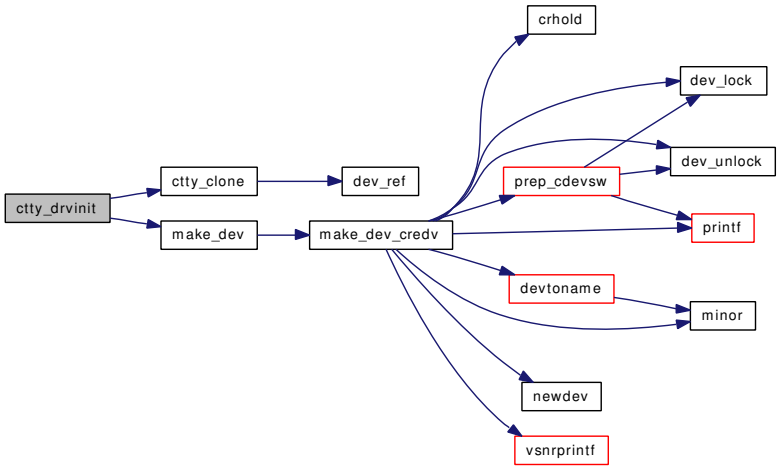
Here is the call graph for this function:

9.137.1.3 static void ctty_drvinit (void * *unused*) [static]

Definition at line 77 of file tty_tty.c.

References ctty, ctty_cdevsw, ctty_clone(), and make_dev().

Here is the call graph for this function:



9.137.1.4 static int cttyopen (struct cdev * *dev*, int *flag*, int *mode*, struct thread * *td*) [static]

Definition at line 48 of file tty_tty.c.

9.137.2 Variable Documentation

9.137.2.1 struct cdev* ctty [static]

Definition at line 45 of file tty_tty.c.

Referenced by ctty_clone(), and ctty_drvinit().

9.137.2.2 struct cdevsw ctty_cdevsw [static]

Initial value:

```
{
        .d_version =    D_VERSION,
        .d_open =       cttyopen,
```

```
        .d_name =       "ctty",
        .d_flags =      D_TTY | D_NEEDGIANT,
}
```

Definition at line 38 of file tty_tty.c.

Referenced by ctty_drvinit().

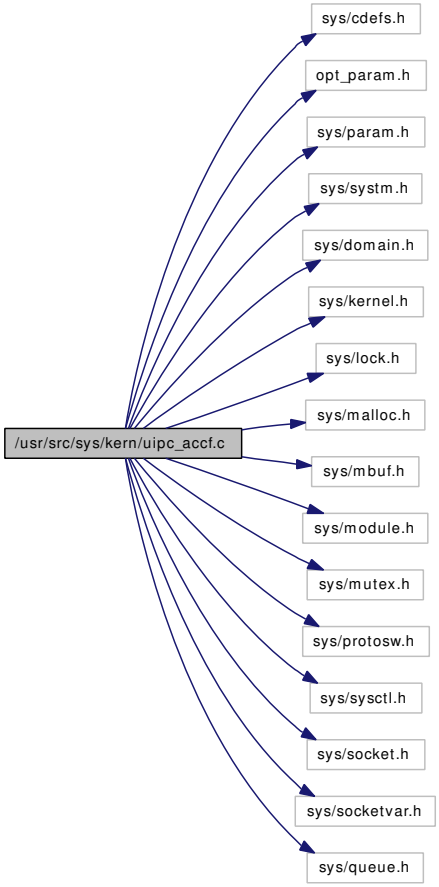
9.137.2.3 d_open_t cttyopen [static]

Definition at line 36 of file tty_tty.c.

9.138 /usr/src/sys/kern/uipc_accf.c File Reference

```
#include <sys/cdefs.h>
#include "opt_param.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/domain.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mbuf.h>
#include <sys/module.h>
#include <sys/mutex.h>
#include <sys/protosw.h>
#include <sys/sysctl.h>
#include <sys/socket.h>
#include <sys/socketvar.h>
#include <sys/queue.h>
```

Include dependency graph for uipc_accf.c:



Defines

- #define ACCEPT_FILTER_MOD
- #define ACCEPT_FILTER_LOCK() mtx_lock(&accept_filter_mtx)
- #define ACCEPT_FILTER_UNLOCK() mtx_unlock(&accept_filter_mtx)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/uipc_accf.c,v 1.19 2005/06/11 11:59:47 maxim Exp $")
- MTX_SYSINIT (accept_filter,&accept_filter_mtx,"accept_filter_mtx", MTX_DEF)
- static SLIST_HEAD (accept_filter)
- int accept_filt_del (char *name)
- accept_filter * accept_filt_get (char *name)
- int accept_filt_generic_mod_event (module_t mod, int event, void *data)
- int do_getopt_accept_filter (struct socket *so, struct sockopt *sopt)
- int do_setopt_accept_filter (struct socket *so, struct sockopt *sopt)

Variables

- static struct mtx accept_filter_mtx

9.138.1 Define Documentation

9.138.1.1 #define ACCEPT_FILTER_LOCK() mtx_lock(&accept_filter_mtx)

Definition at line 53 of file uipc_accf.c.

Referenced by accept_filt_get(), and SLIST_HEAD().

9.138.1.2 #define ACCEPT_FILTER_MOD

Definition at line 32 of file uipc_accf.c.

9.138.1.3 #define ACCEPT_FILTER_UNLOCK() mtx_unlock(&accept_filter_mtx)

Definition at line 54 of file uipc_accf.c.

Referenced by accept_filt_get(), and SLIST_HEAD().

9.138.2 Function Documentation

9.138.2.1 __FBSDID ("$FreeBSD: src/sys/kern/uipc_accf. *c*, v 1.19 2005/06/11 11:59:47 maxim Exp $")

9.138.2.2 int accept_filt_del (char * *name*)

Definition at line 99 of file uipc_accf.c.

References accept_filt_get().

Referenced by accept_filt_generic_mod_event().

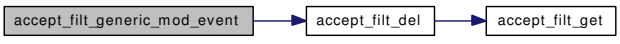
Here is the call graph for this function:

9.138.2.3 int accept_filt_generic_mod_event (module_t *mod*, int *event*, void * *data*)

Definition at line 126 of file uipc_accf.c.

References accept_filt_del(), and unloadable.

Here is the call graph for this function:



9.138.2.4 struct accept_filter* accept_filt_get (char * *name*)

Definition at line 112 of file uipc_accf.c.

References ACCEPT_FILTER_LOCK, and ACCEPT_FILTER_UNLOCK.

Referenced by accept_filt_del(), and do_setopt_accept_filter().

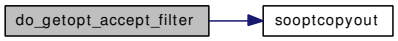
9.138.2.5 int do_getopt_accept_filter (struct socket * *so*, struct sockopt * *sopt*)

Definition at line 166 of file uipc_accf.c.

References sooptcopyout().

Referenced by sogetopt().

Here is the call graph for this function:



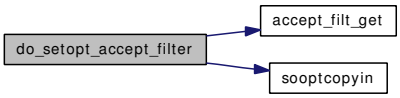
9.138.2.6 int do_setopt_accept_filter (struct socket * *so*, struct sockopt * *sopt*)

Definition at line 195 of file uipc_accf.c.

References accept_filt_get(), and sooptcopyin().

Referenced by sodealloc(), and sosetopt().

Here is the call graph for this function:



9.138.2.7 MTX_SYSINIT (accept_filter, & *accept_filter_mtx*, "accept_filter_mtx", MTX_DEF)

9.138.2.8 static SLIST_HEAD (accept_filter) [static]

Definition at line 56 of file uipc_accf.c.

References ACCEPT_FILTER_LOCK, and ACCEPT_FILTER_UNLOCK.

9.138.3 Variable Documentation

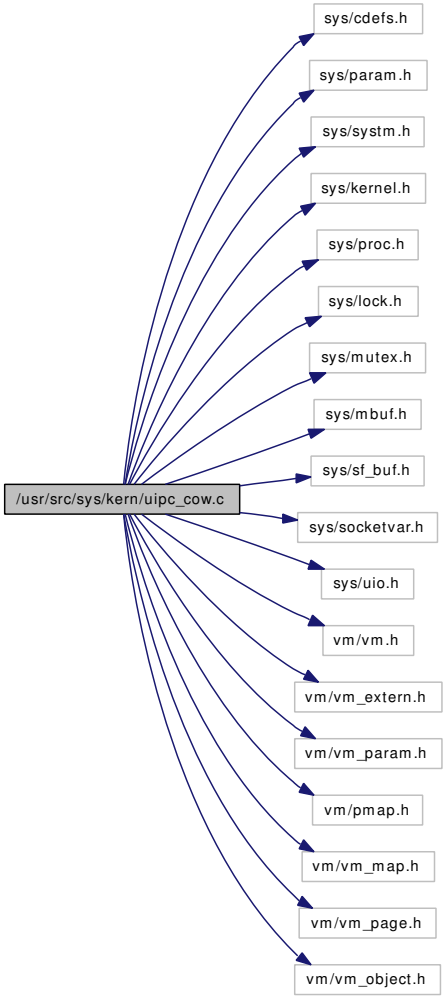
9.138.3.1 struct mtx accept_filter_mtx [static]

Definition at line 50 of file uipc_accf.c.

9.139 /usr/src/sys/kern/uipc_cow.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/proc.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/mbuf.h>
#include <sys/sf_buf.h>
#include <sys/socketvar.h>
#include <sys/uio.h>
#include <vm/vm.h>
#include <vm/vm_extern.h>
#include <vm/vm_param.h>
#include <vm/pmap.h>
#include <vm/vm_map.h>
#include <vm/vm_page.h>
#include <vm/vm_object.h>
```

Include dependency graph for uipc_cow.c:



Data Structures

- struct netsend_cow_stats

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/uipc_cow.c,v 1.26 2005/10/23 07:41:56 alc Exp $")
- static void socow_iodone (void *addr, void *args)
- int socow_setup (struct mbuf *m0, struct uio *uio)

Variables

- static struct netsend_cow_stats socow_stats

9.139.1 Function Documentation

9.139.1.1 __FBSDID ("$FreeBSD: src/sys/kern/uipc_cow. *c*, v 1.26 2005/10/23 07:41:56 alc Exp $")

9.139.1.2 static void socow_iodone (void * *addr*, void * *args*) [static]

Definition at line 74 of file uipc_cow.c.

References netsend_cow_stats::iodone, and socow_stats.

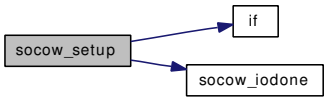
Referenced by socow_setup().

9.139.1.3 int socow_setup (struct mbuf * *m0*, struct uio * *uio*)

Definition at line 98 of file uipc_cow.c.

References netsend_cow_stats::attempted, netsend_cow_stats::fail_not_mapped, netsend_cow_-stats::fail_sf_buf, if(), socow_iodone(), socow_stats, and netsend_cow_stats::success.

Here is the call graph for this function:



9.139.2 Variable Documentation

9.139.2.1 struct netsend_cow_stats socow_stats [static]

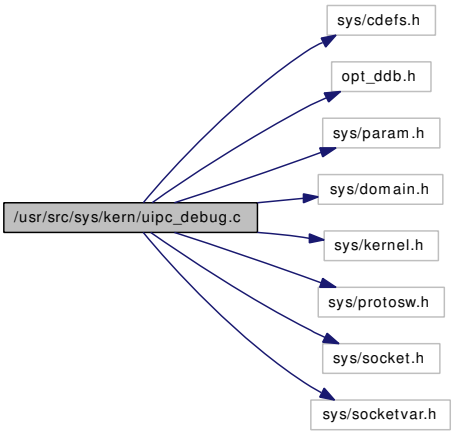
Definition at line 69 of file uipc_cow.c.

Referenced by socow_iodone(), and socow_setup().

9.140 /usr/src/sys/kern/uipc_debug.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ddb.h"
#include <sys/param.h>
#include <sys/domain.h>
#include <sys/kernel.h>
#include <sys/protosw.h>
#include <sys/socket.h>
#include <sys/socketvar.h>
```

Include dependency graph for uipc_debug.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/uipc_debug.c,v 1.1 2007/02/15 01:28:22 rwatson Exp $")

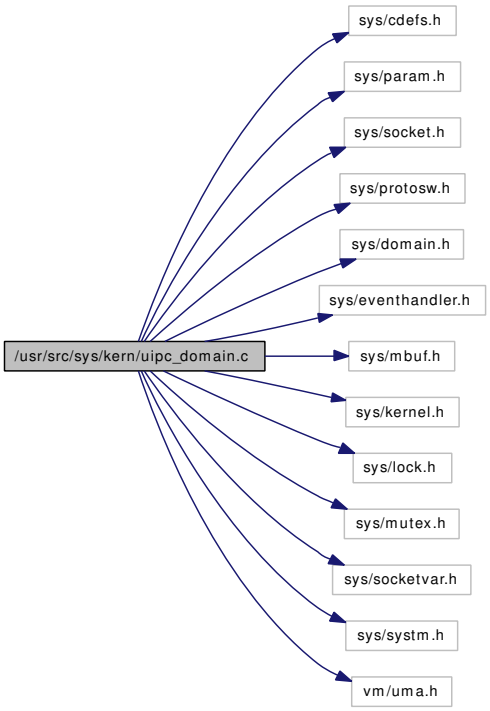
9.140.1 Function Documentation

9.140.1.1 __FBSDID ("$FreeBSD: src/sys/kern/uipc_debug. *c*, v 1.1 2007/02/15 01:28:22 rwatson Exp $")

9.141 /usr/src/sys/kern/uipc_domain.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/socket.h>
#include <sys/protosw.h>
#include <sys/domain.h>
#include <sys/eventhandler.h>
#include <sys/mbuf.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/socketvar.h>
#include <sys/systm.h>
#include <vm/uma.h>
```

Include dependency graph for uipc_domain.c:



Defines

- #define DEFAULT(foo, bar) if ((foo) == NULL) (foo) = (bar)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/uipc_domain.c,v 1.48 2006/07/24 15:20:06 rwatson Exp $")
- static void domaininit (void *)
- static void domainfinalize (void *)
- SYSINIT (domainfin, SI_SUB_PROTO_IFATTACHDOMAIN, SI_ORDER_FIRST, domainfinalize, NULL)
- static void protosw_init (struct protosw *pr)
- static void net_init_domain (struct domain *dp)
- void net_add_domain (void *data)
- static void socket_zone_change (void *tag)
- protosw * pffindtype (int family, int type)
- protosw * pffindproto (int family, int protocol, int type)
- int pf_proto_register (int family, struct protosw *npr)
- int pf_proto_unregister (int family, int protocol, int type)
- void pfctlinput (int cmd, struct sockaddr *sa)
- void pfctlinput2 (int cmd, struct sockaddr *sa, void *ctlparam)
- static void pfslowtimo (void *arg)
- static void pffasttimo (void *arg)

9.141.1 Define Documentation

9.141.1.1 #define DEFAULT(foo, bar) if ((foo) == NULL) (foo) = (bar)

Referenced by protosw_init().

9.141.2 Function Documentation

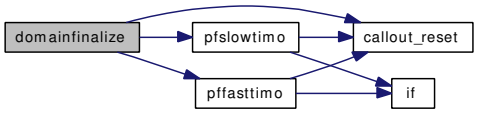
9.141.2.1 __FBSDID ("$FreeBSD: src/sys/kern/uipc_domain. *c*, v 1.48 2006/07/24 15:20:06 rwatson Exp $")

9.141.2.2 static void domainfinalize (void *) [static]

Definition at line 230 of file uipc_domain.c.

References callout_reset(), pffasttimo(), and pfslowtimo().

Here is the call graph for this function:

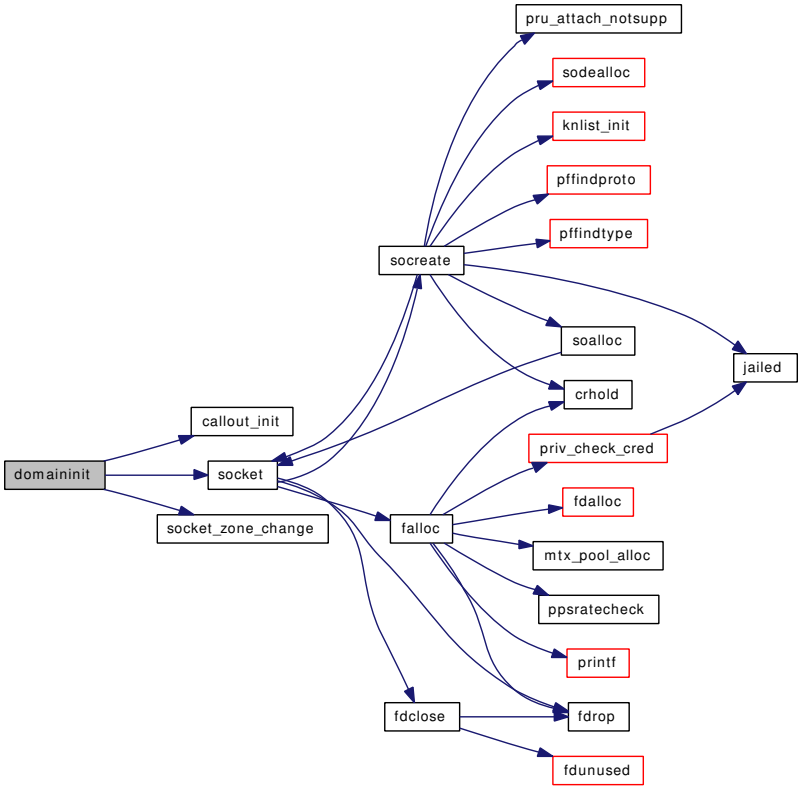


9.141.2.3 static void domaininit (void *) [static]

Definition at line 198 of file uipc_domain.c.

References callout_init(), max_linkhdr, maxsockets, socket(), socket_zone, and socket_zone_change().

Here is the call graph for this function:

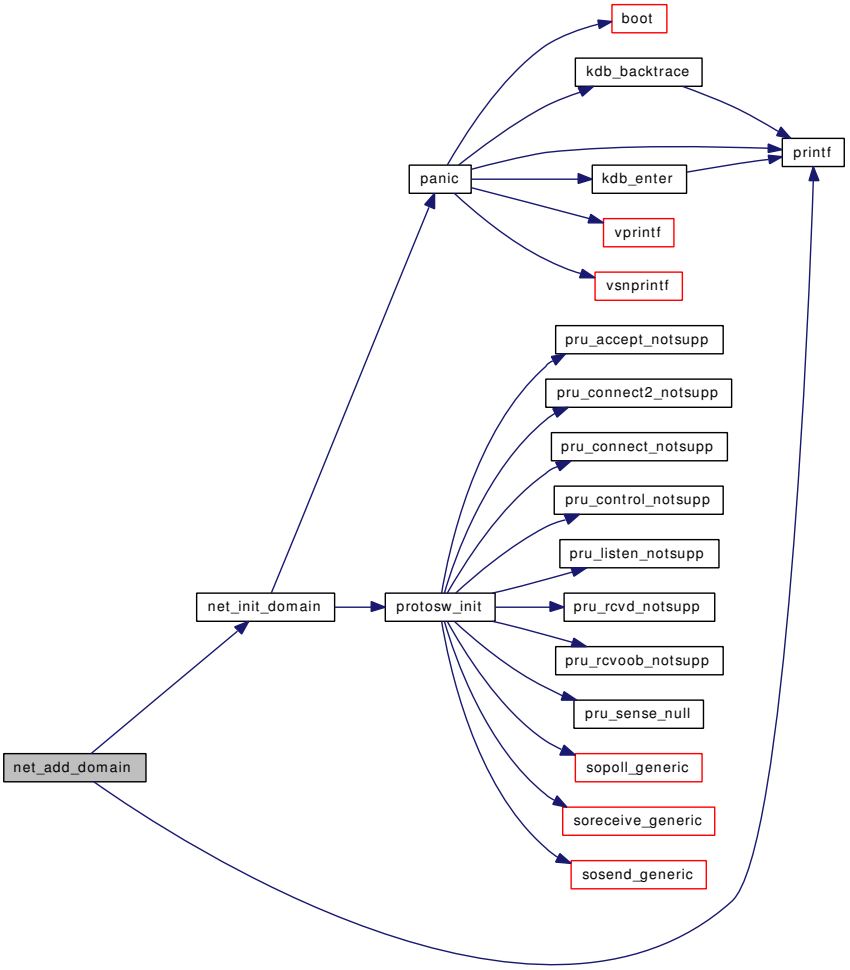


9.141.2.4 void net_add_domain (void * *data*)

Definition at line 159 of file uipc_domain.c.

References net_init_domain(), and printf().

Here is the call graph for this function:



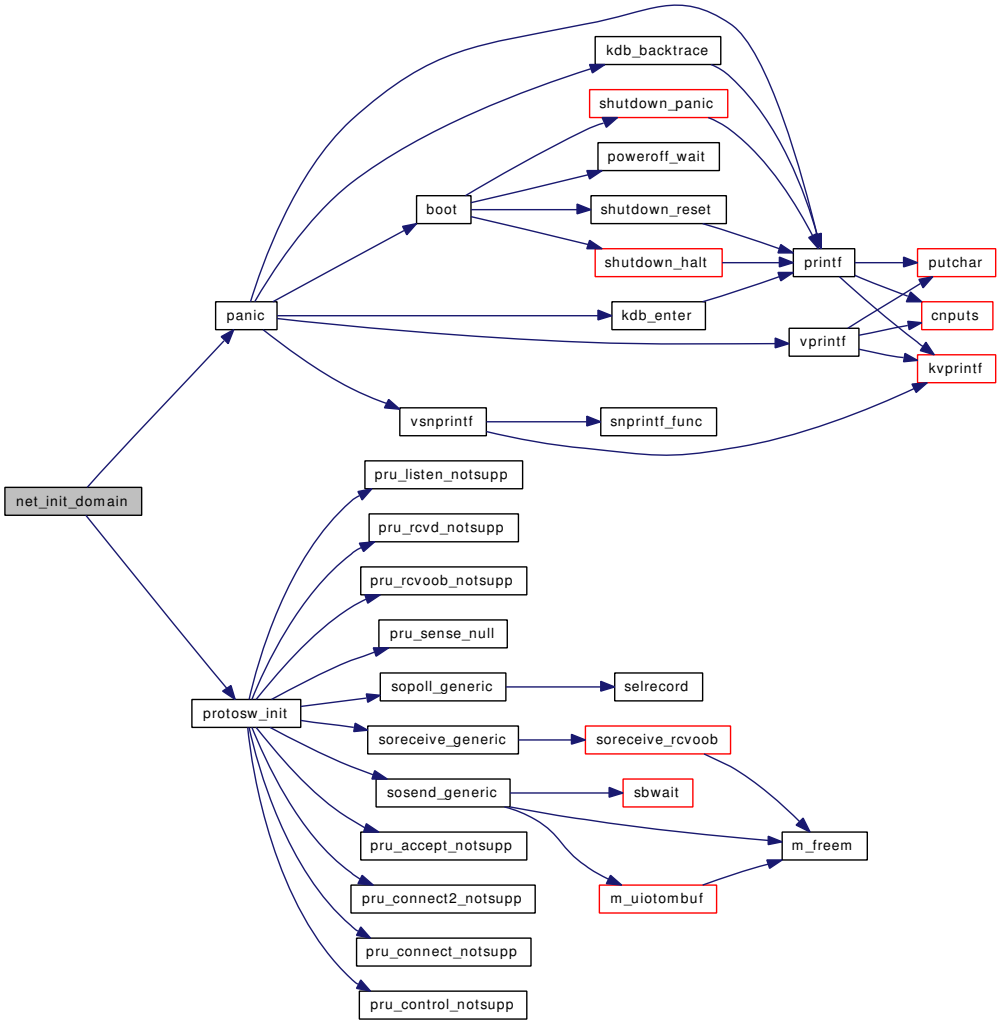
9.141.2.5 static void net_init_domain (struct domain * *dp*) [static]

Definition at line 136 of file uipc_domain.c.

References max_datalen, max_hdr, max_linkhdr, max_protohdr, panic(), pr, and protosw_init().

Referenced by net_add_domain().

Here is the call graph for this function:

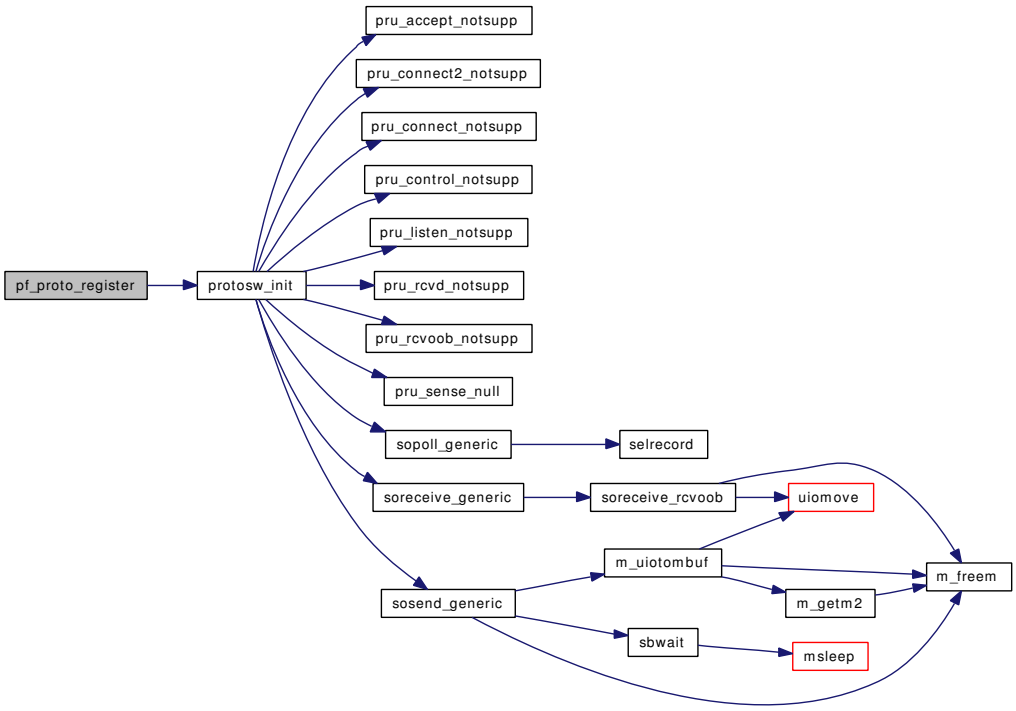


9.141.2.6 int pf_proto_register (int *family*, struct protosw * *npr*)

Definition at line 293 of file uipc_domain.c.

References Giant, pr, and protosw_init().

Here is the call graph for this function:



9.141.2.7 int pf_proto_unregister (int *family*, int *protocol*, int *type*)

Definition at line 362 of file uipc_domain.c.

References Giant, and pr.

9.141.2.8 void pfctlinput (int *cmd*, struct sockaddr * *sa*)

Definition at line 430 of file uipc_domain.c.

References if(), and pr.

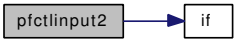
Here is the call graph for this function:

9.141.2.9 void pfctlinput2 (int *cmd*, struct sockaddr * *sa*, void * *ctlparam*)

Definition at line 444 of file uipc_domain.c.

References if(), and pr.

Here is the call graph for this function:



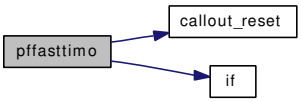
9.141.2.10 static void pffasttimo (void * *arg*) [static]

Definition at line 486 of file uipc_domain.c.

References callout_reset(), hz, if(), and pr.

Referenced by domainfinalize().

Here is the call graph for this function:



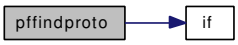
9.141.2.11 struct protosw* pffindproto (int *family*, int *protocol*, int *type*)

Definition at line 261 of file uipc_domain.c.

References if(), and pr.

Referenced by socreate().

Here is the call graph for this function:



9.141.2.12 struct protosw* pffindtype (int *family*, int *type*)

Definition at line 242 of file uipc_domain.c.

References if(), and pr.

Referenced by socreate().

Here is the call graph for this function:

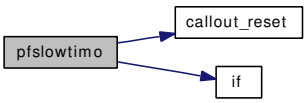
9.141.2.13 static void pfslowtimo (void * *arg*) [static]

Definition at line 470 of file uipc_domain.c.

References callout_reset(), hz, if(), and pr.

Referenced by domainfinalize().

Here is the call graph for this function:



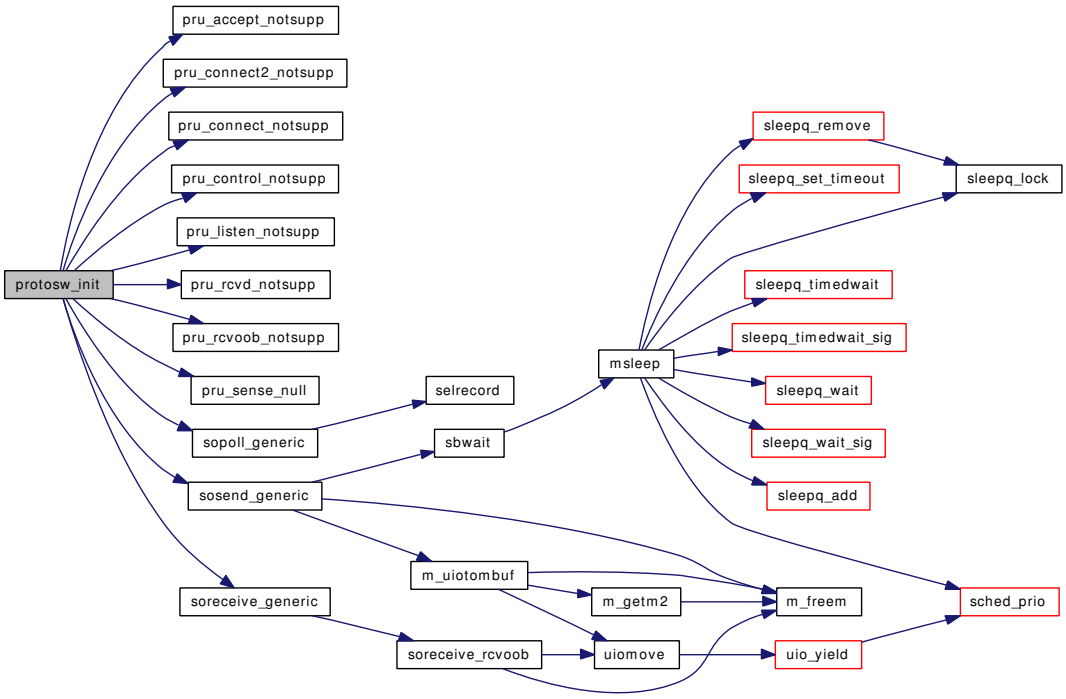
9.141.2.14 static void protosw_init (struct protosw * *pr*) [static]

Definition at line 104 of file uipc_domain.c.

References DEFAULT, pru_accept_notsupp(), pru_connect2_notsupp(), pru_connect_notsupp(), pru_control_notsupp(), pru_listen_notsupp(), pru_rcvd_notsupp(), pru_rcvoob_notsupp(), pru_sense_null(), sopoll_generic(), soreceive_generic(), and sosend_generic().

Referenced by net_init_domain(), and pf_proto_register().

Here is the call graph for this function:



9.141.2.15 static void socket_zone_change (void * *tag*) [static]

Definition at line 190 of file uipc_domain.c.

References maxsockets, and socket_zone.

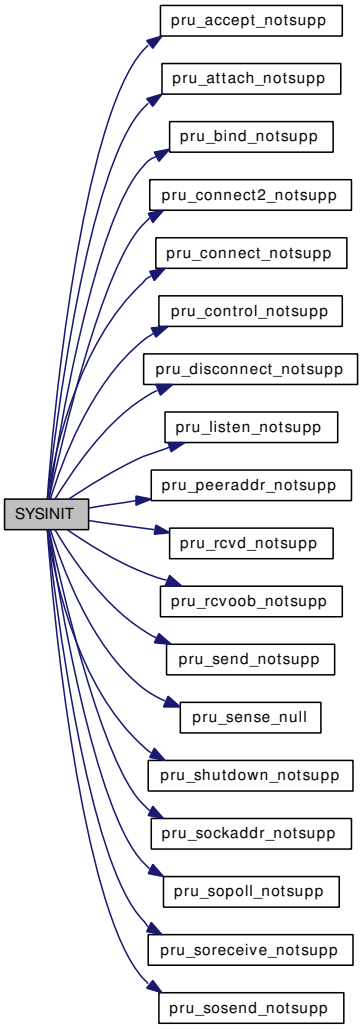
Referenced by domaininit().

9.141.2.16 SYSINIT (domainfin, SI_SUB_PROTO_IFATTACHDOMAIN, SI_ORDER_FIRST, domainfinalize, NULL)

Definition at line 64 of file uipc_domain.c.

References pru_accept_notsupp(), pru_attach_notsupp(), pru_bind_notsupp(), pru_connect2_notsupp(), pru_connect_notsupp(), pru_control_notsupp(), pru_disconnect_notsupp(), pru_listen_notsupp(), pru_peeraddr_notsupp(), pru_rcvd_notsupp(), pru_rcvoob_notsupp(), pru_send_notsupp(), pru_sense_null(), pru_shutdown_notsupp(), pru_sockaddr_notsupp(), pru_sopoll_notsupp(), pru_soreceive_notsupp(), and pru_sosend_notsupp().

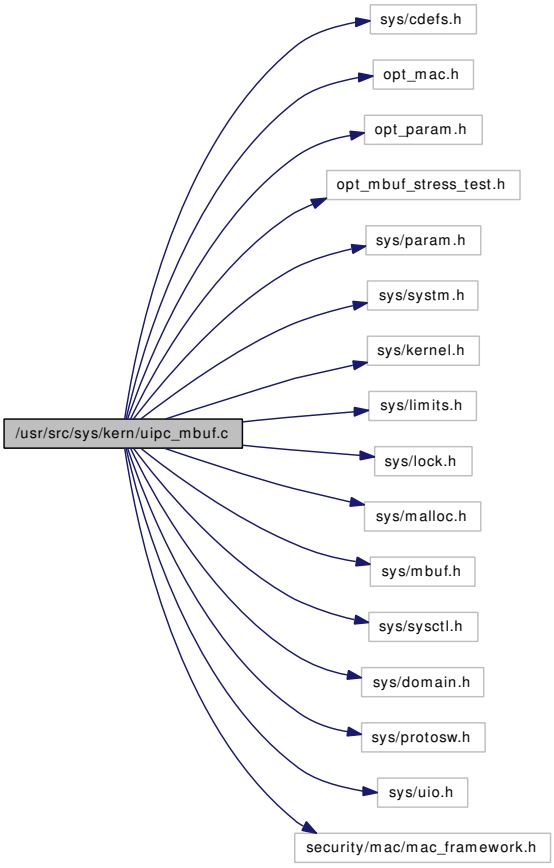
Here is the call graph for this function:



9.142 /usr/src/sys/kern/uipc_mbuf.c File Reference

```
#include <sys/cdefs.h>
#include "opt_mac.h"
#include "opt_param.h"
#include "opt_mbuf_stress_test.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/limits.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mbuf.h>
#include <sys/sysctl.h>
#include <sys/domain.h>
#include <sys/protosw.h>
#include <sys/uio.h>
#include <security/mac/mac_framework.h>
```

Include dependency graph for uipc_mbuf.c:



Defines

- #define M_SANITY_ACTION(s) return (0)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/uipc_mbuf.c,v 1.171 2007/01/22 14:50:28 andre Exp $")
- SYSCTL_INT (_kern_ipc, KIPC_MAX_LINKHDR, max_linkhdr, CTLFLAG_RD,&max_linkhdr, 0,"Size of largest link layer header")
- SYSCTL_INT (_kern_ipc, KIPC_MAX_PROTOHDR, max_protohdr, CTLFLAG_RD,&max_protohdr, 0,"Size of largest protocol layer header")
- SYSCTL_INT (_kern_ipc, KIPC_MAX_HDR, max_hdr, CTLFLAG_RD,&max_hdr, 0,"Size of largest link plus protocol header")
- SYSCTL_INT (_kern_ipc, KIPC_MAX_DATALEN, max_datalen, CTLFLAG_RD,&max_datalen, 0,"Minimum space left in mbuf after max_hdr")
- mbuf * m_getm2 (struct mbuf *m, int len, int how, short type, int flags)
- void m_freem (struct mbuf *mb)
- void m_extadd (struct mbuf *mb, caddr_t buf, u_int size, void(*freef)(void *, void *), void *args, int flags, int type)
- void mb_free_ext (struct mbuf *m)
- static void mb_dupcl (struct mbuf *n, struct mbuf *m)
- void m_demote (struct mbuf *m0, int all)

- int m_sanity (struct mbuf *m0, int sanitize)
- void m_move_pkthdr (struct mbuf *to, struct mbuf *from)
- int m_dup_pkthdr (struct mbuf *to, struct mbuf *from, int how)
- mbuf * m_prepend (struct mbuf *m, int len, int how)
- mbuf * m_copym (struct mbuf *m, int off0, int len, int wait)
- static int m_bcopyxxx (void *s, void *t, u_int len)
- mbuf * m_copymdata (struct mbuf *m, struct mbuf *n, int off, int len, int prep, int how)
- mbuf * m_copypacket (struct mbuf *m, int how)
- void m_copydata (const struct mbuf *m, int off, int len, caddr_t cp)
- mbuf * m_dup (struct mbuf *m, int how)
- void m_cat (struct mbuf *m, struct mbuf *n)
- void m_adj (struct mbuf *mp, int req_len)
- mbuf * m_pullup (struct mbuf *n, int len)
- mbuf * m_copyup (struct mbuf *n, int len, int dstoff)
- mbuf * m_split (struct mbuf *m0, int len0, int wait)
- mbuf * m_devget (char *buf, int totlen, int off, struct ifnet *ifp, void(*copy)(char *from, caddr_t to, u_int len))
- void m_copyback (struct mbuf *m0, int off, int len, c_caddr_t cp)
- int m_append (struct mbuf *m0, int len, c_caddr_t cp)
- int m_apply (struct mbuf *m, int off, int len, int(*f)(void *, void *, u_int), void *arg)
- mbuf * m_getptr (struct mbuf *m, int loc, int *off)
- void m_print (const struct mbuf *m, int maxlen)
- u_int m_fixhdr (struct mbuf *m0)
- u_int m_length (struct mbuf *m0, struct mbuf **last)
- mbuf * m_defrag (struct mbuf *m0, int how)
- mbuf * m_uiotombuf (struct uio *uio, int how, int len, int align, int flags)
- void m_align (struct mbuf *m, int len)
- mbuf * m_unshare (struct mbuf *m0, int how)

Variables

- int max_linkhdr
- int max_protohdr
- int max_hdr
- int max_datalen
- int MSFail

9.142.1 Define Documentation

9.142.1.1 #define M_SANITY_ACTION(s) return (0)

Referenced by m_sanity().

9.142.2 Function Documentation

9.142.2.1 __FBSDID ("$FreeBSD: src/sys/kern/uipc_mbuf. *c*, v 1.171 2007/01/22 14:50:28 andre Exp $")

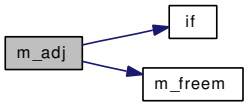
9.142.2.2 void m_adj (struct mbuf * *mp*, int *req_len*)

Definition at line 919 of file uipc_mbuf.c.

References if(), and m_freem().

Referenced by m_pulldown().

Here is the call graph for this function:



9.142.2.3 void m_align (struct mbuf * *m*, int *len*)

Definition at line 1681 of file uipc_mbuf.c.

9.142.2.4 int m_append (struct mbuf * *m0*, int *len*, c_caddr_t *cp*)

Definition at line 1294 of file uipc_mbuf.c.

9.142.2.5 int m_apply (struct mbuf * *m*, int *off*, int *len*, int(*)(void *, void *, u_int) *f*, void * *arg*)

Definition at line 1337 of file uipc_mbuf.c.

Referenced by m_copymdata().

9.142.2.6 static int m_bcopyxxx (void * *s*, void * *t*, u_int *len*) [static]

Definition at line 596 of file uipc_mbuf.c.

Referenced by m_copymdata().

9.142.2.7 void m_cat (struct mbuf * *m*, struct mbuf * *n*)

Definition at line 899 of file uipc_mbuf.c.

Referenced by kern_sendfile(), and m_defrag().

9.142.2.8 void m_copyback (struct mbuf * *m0*, int *off*, int *len*, c_caddr_t *cp*)

Definition at line 1242 of file uipc_mbuf.c.

9.142.2.9 void m_copydata (const struct mbuf * *m*, int *off*, int *len*, caddr_t *cp*)

Definition at line 794 of file uipc_mbuf.c.

Referenced by m_defrag(), m_dup1(), and m_pulldown().

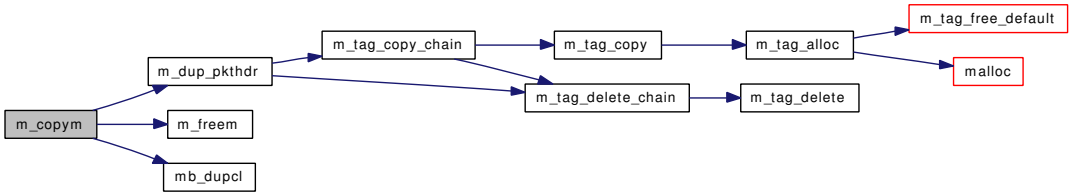
9.142.2.10 struct mbuf* m_copym (struct mbuf * *m*, int *off0*, int *len*, int *wait*)

Definition at line 518 of file uipc_mbuf.c.

References m_dup_pkthdr(), m_freem(), mb_dupcl(), and mbstat.

Referenced by md_get_mbuf().

Here is the call graph for this function:

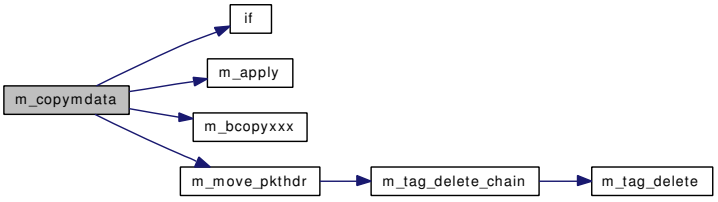


9.142.2.11 struct mbuf* m_copymdata (struct mbuf * *m*, struct mbuf * *n*, int *off*, int *len*, int *prep*, int *how*)

Definition at line 603 of file uipc_mbuf.c.

References if(), m_apply(), m_bcopyxxx(), and m_move_pkthdr().

Here is the call graph for this function:

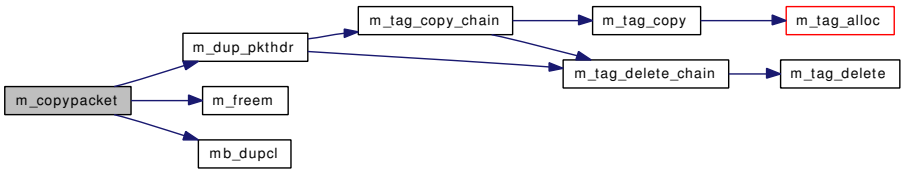


9.142.2.12 struct mbuf* m_copypacket (struct mbuf * *m*, int *how*)

Definition at line 742 of file uipc_mbuf.c.

References m_dup_pkthdr(), m_freem(), mb_dupcl(), and mbstat.

Here is the call graph for this function:



9.142.2.13 struct mbuf* m_copyup (struct mbuf * *n*, int *len*, int *dstoff*)

Definition at line 1063 of file uipc_mbuf.c.

References m_freem(), and max_protohdr.

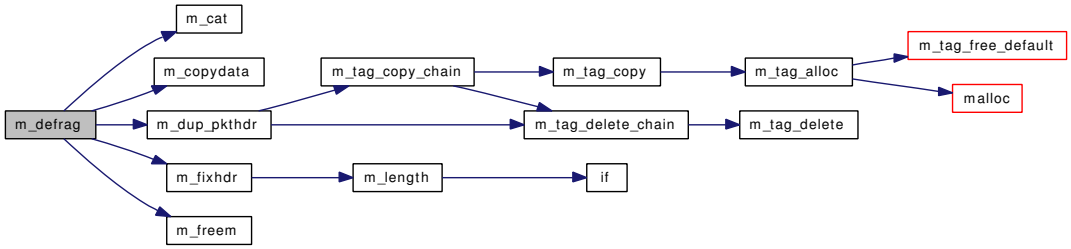
Here is the call graph for this function:

9.142.2.14 struct mbuf* m_defrag (struct mbuf * *m0*, int *how*)

Definition at line 1463 of file uipc_mbuf.c.

References m_cat(), m_copydata(), m_dup_pkthdr(), m_fixhdr(), and m_freem().

Here is the call graph for this function:

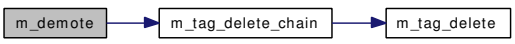


9.142.2.15 void m_demote (struct mbuf * *m0*, int *all*)

Definition at line 301 of file uipc_mbuf.c.

References m_tag_delete_chain().

Here is the call graph for this function:



9.142.2.16 struct mbuf* m_devget (char * *buf*, int *totlen*, int *off*, struct ifnet * *ifp*, void(*)(char *from, caddr_t to, u_int len) *copy*)

Definition at line 1176 of file uipc_mbuf.c.

References m_freem(), and max_linkhdr.

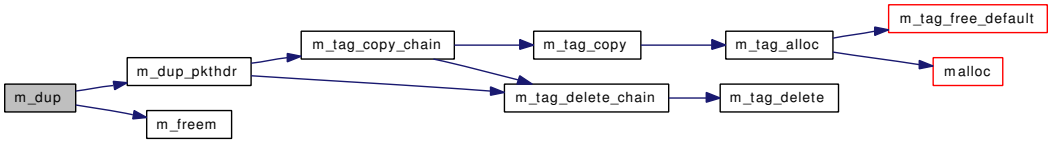
Here is the call graph for this function:

9.142.2.17 struct mbuf* m_dup (struct mbuf * *m*, int *how*)

Definition at line 824 of file uipc_mbuf.c.

References m_dup_pkthdr(), m_freem(), and mbstat.

Here is the call graph for this function:



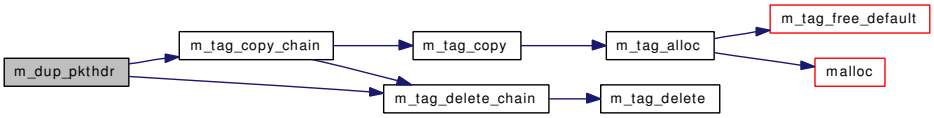
9.142.2.18 int m_dup_pkthdr (struct mbuf * *to*, struct mbuf * *from*, int *how*)

Definition at line 449 of file uipc_mbuf.c.

References m_tag_copy_chain(), and m_tag_delete_chain().

Referenced by m_copym(), m_copypacket(), m_defrag(), m_dup(), and m_dup1().

Here is the call graph for this function:



9.142.2.19 void m_extadd (struct mbuf * *mb*, caddr_t *buf*, u_int *size*, void(*)(void *, void *) *freef*, void * *args*, int *flags*, int *type*)

Definition at line 188 of file uipc_mbuf.c.

References zone_ext_refcnt.

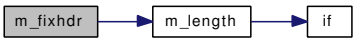
9.142.2.20 u_int m_fixhdr (struct mbuf * *m0*)

Definition at line 1425 of file uipc_mbuf.c.

References m_length().

Referenced by m_defrag(), and mb_fixhdr().

Here is the call graph for this function:



9.142.2.21 void m_freem (struct mbuf * *mb*)

Definition at line 159 of file uipc_mbuf.c.

Referenced by kern_sendfile(), m_adj(), m_copym(), m_copypacket(), m_copyup(), m_defrag(), m_devget(), m_dup(), m_getm2(), m_prepend(), m_pulldown(), m_pullup(), m_sanity(), m_uiotombuf(), m_unshare(), mb_done(), md_done(), soopt_getm(), soopt_mcopyin(), soopt_mcopyout(), soreceive_rcvoob(), sosend_dgram(), sosend_generic(), uipc_send(), and unp_externalize().

9.142.2.22 struct mbuf* m_getm2 (struct mbuf * *m*, int *len*, int *how*, short *type*, int *flags*)

Definition at line 97 of file uipc_mbuf.c.

References m_freem().

Referenced by m_uiotombuf().

Here is the call graph for this function:

9.142.2.23 struct mbuf* m_getptr (struct mbuf * *m*, int *loc*, int * *off*)

Definition at line 1369 of file uipc_mbuf.c.

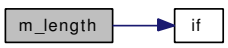
9.142.2.24 u_int m_length (struct mbuf * *m0*, struct mbuf ** *last*)

Definition at line 1435 of file uipc_mbuf.c.

References if().

Referenced by kern_sendfile(), m_fixhdr(), sbappendaddr_locked(), and sbappendcontrol_locked().

Here is the call graph for this function:



9.142.2.25 void m_move_pkthdr (struct mbuf * *to*, struct mbuf * *from*)

Definition at line 418 of file uipc_mbuf.c.

References m_tag_delete_chain().

Referenced by m_copymdata().

Here is the call graph for this function:

9.142.2.26 struct mbuf* m_prepend (struct mbuf * *m*, int *len*, int *how*)

Definition at line 483 of file uipc_mbuf.c.

References m_freem().

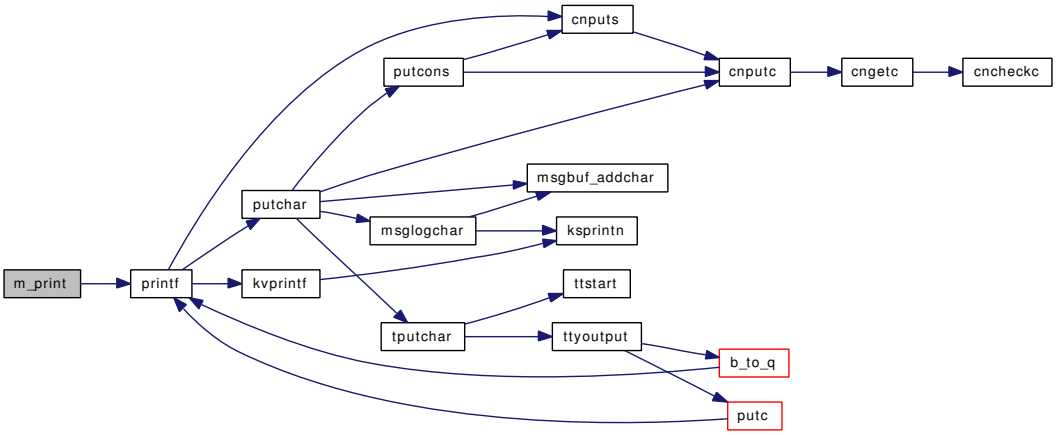
Here is the call graph for this function:

9.142.2.27 void m_print (const struct mbuf * *m*, int *maxlen*)

Definition at line 1394 of file uipc_mbuf.c.

References printf().

Here is the call graph for this function:



9.142.2.28 struct mbuf* m_pullup (struct mbuf * *n*, int *len*)

Definition at line 1001 of file uipc_mbuf.c.

References m_freem(), max_protohdr, and mbstat.

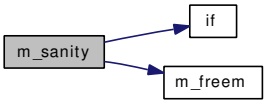
Here is the call graph for this function:

9.142.2.29 int m_sanity (struct mbuf * *m0*, int *sanitize*)

Definition at line 327 of file uipc_mbuf.c.

References if(), m_freem(), and M_SANITY_ACTION.

Here is the call graph for this function:



9.142.2.30 struct mbuf* m_split (struct mbuf * *m0*, int *len0*, int *wait*)

Definition at line 1114 of file uipc_mbuf.c.

References if().

Here is the call graph for this function:

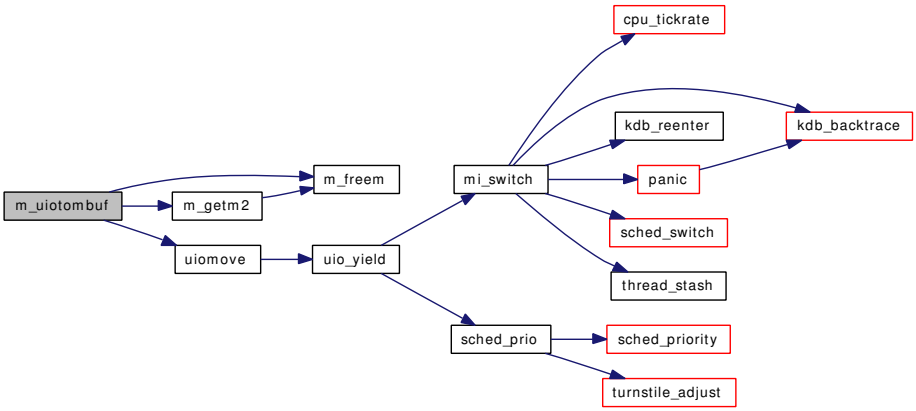
9.142.2.31 struct mbuf* m_uiotombuf (struct uio * *uio*, int *how*, int *len*, int *align*, int *flags*)

Definition at line 1622 of file uipc_mbuf.c.

References m_freem(), m_getm2(), and uiomove().

Referenced by kern_sendfile(), sosend_dgram(), and sosend_generic().

Here is the call graph for this function:

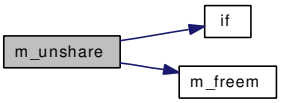


9.142.2.32 struct mbuf* m_unshare (struct mbuf * *m0*, int *how*)

Definition at line 1704 of file uipc_mbuf.c.

References if(), and m_freem().

Here is the call graph for this function:



9.142.2.33 static void mb_dupcl (struct mbuf * *n*, struct mbuf * *m*) [static]

Definition at line 276 of file uipc_mbuf.c.

Referenced by m_copym(), and m_copypacket().

9.142.2.34 void mb_free_ext (struct mbuf * *m*)

Definition at line 212 of file uipc_mbuf.c.

References zone_clust, zone_ext_refcnt, zone_jumbo16, zone_jumbo9, zone_jumbop, zone_mbuf, and zone_pack.

9.142.2.35 SYSCTL_INT (_kern_ipc, KIPC_MAX_DATALEN, max_datalen, CTLFLAG_RD, & *max_datalen*, 0, "Minimum space left in mbuf after max_hdr")

9.142.2.36 SYSCTL_INT (_kern_ipc, KIPC_MAX_HDR, max_hdr, CTLFLAG_RD, & *max_hdr*, 0, "Size of largest link plus protocol header")

9.142.2.37 SYSCTL_INT (_kern_ipc, KIPC_MAX_PROTOHDR, max_protohdr, CTLFLAG_RD, & *max_protohdr*, 0, "Size of largest protocol layer header")

9.142.2.38 SYSCTL_INT (_kern_ipc, KIPC_MAX_LINKHDR, max_linkhdr, CTLFLAG_RD, & *max_linkhdr*, 0, "Size of largest link layer header")

9.142.3 Variable Documentation

9.142.3.1 int max_datalen

Definition at line 56 of file uipc_mbuf.c.

Referenced by net_init_domain().

9.142.3.2 int max_hdr

Definition at line 55 of file uipc_mbuf.c.

Referenced by net_init_domain(), sosend_dgram(), and sosend_generic().

9.142.3.3 int max_linkhdr

Definition at line 53 of file uipc_mbuf.c.

Referenced by domaininit(), m_devget(), and net_init_domain().

9.142.3.4 int max_protohdr

Definition at line 54 of file uipc_mbuf.c.

Referenced by m_copyup(), m_pullup(), and net_init_domain().

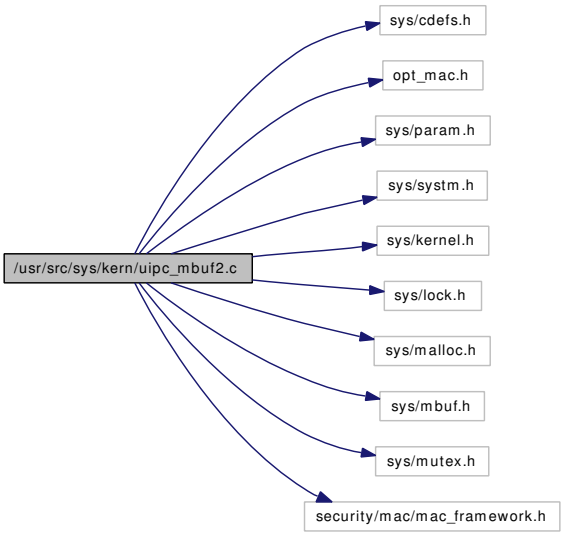
9.142.3.5 int MSFail

Definition at line 1060 of file uipc_mbuf.c.

9.143 /usr/src/sys/kern/uipc_mbuf2.c File Reference

```
#include <sys/cdefs.h>
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mbuf.h>
#include <sys/mutex.h>
#include <security/mac/mac_framework.h>
```

Include dependency graph for uipc_mbuf2.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/uipc_mbuf2.c,v 1.33 2006/10/22 11:52:13 rwatson Exp $")
- static MALLOC_DEFINE (M_PACKET_TAGS, MBUF_TAG_MEM_NAME,"packet-attached information")
- static struct mbuf * m_dup1 (struct mbuf *, int, int, int)
- mbuf * m_pulldown (struct mbuf *m, int off, int len, int *offp)
- void m_tag_free_default (struct m_tag *t)
- m_tag * m_tag_alloc (u_int32_t cookie, int type, int len, int wait)
- void m_tag_delete (struct mbuf *m, struct m_tag *t)
- void m_tag_delete_chain (struct mbuf *m, struct m_tag *t)
- void m_tag_delete_nonpersistent (struct mbuf *m)
- m_tag * m_tag_locate (struct mbuf *m, u_int32_t cookie, int type, struct m_tag *t)
- m_tag * m_tag_copy (struct m_tag *t, int how)
- int m_tag_copy_chain (struct mbuf *to, struct mbuf *from, int how)

9.143.1 Function Documentation

9.143.1.1 __FBSDID ("$FreeBSD: src/sys/kern/uipc_mbuf2. *c*, v 1.33 2006/10/22 11:52:13 rwatson Exp $")

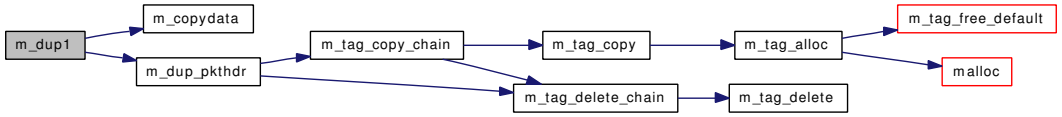
9.143.1.2 static struct mbuf * m_dup1 (struct mbuf *, int, int, int) [static]

Definition at line 272 of file uipc_mbuf2.c.

References m_copydata(), and m_dup_pkthdr().

Referenced by m_pulldown().

Here is the call graph for this function:

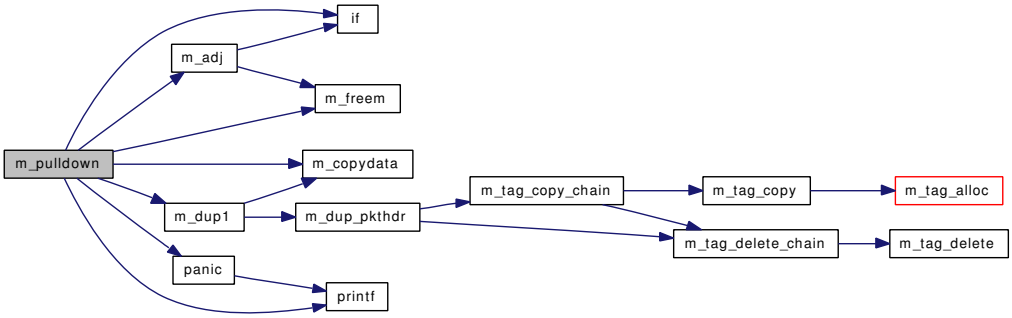


9.143.1.3 struct mbuf* m_pulldown (struct mbuf * *m*, int *off*, int *len*, int * *offp*)

Definition at line 97 of file uipc_mbuf2.c.

References if(), m_adj(), m_copydata(), m_dup1(), m_freem(), panic(), and printf().

Here is the call graph for this function:



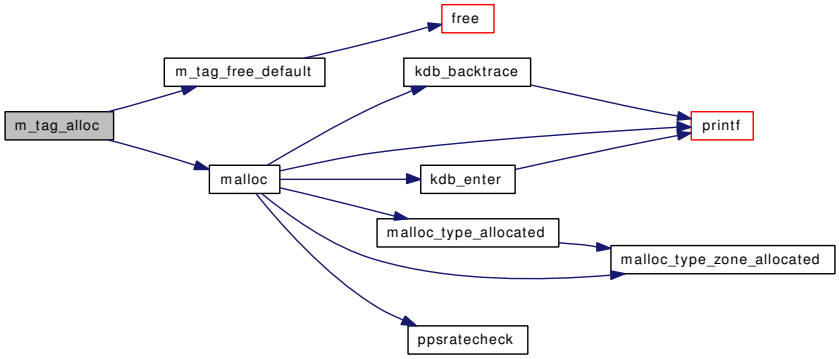
9.143.1.4 struct m_tag* m_tag_alloc (u_int32_t *cookie*, int *type*, int *len*, int *wait*)

Definition at line 319 of file uipc_mbuf2.c.

References m_tag_free_default(), and malloc().

Referenced by m_tag_copy().

Here is the call graph for this function:



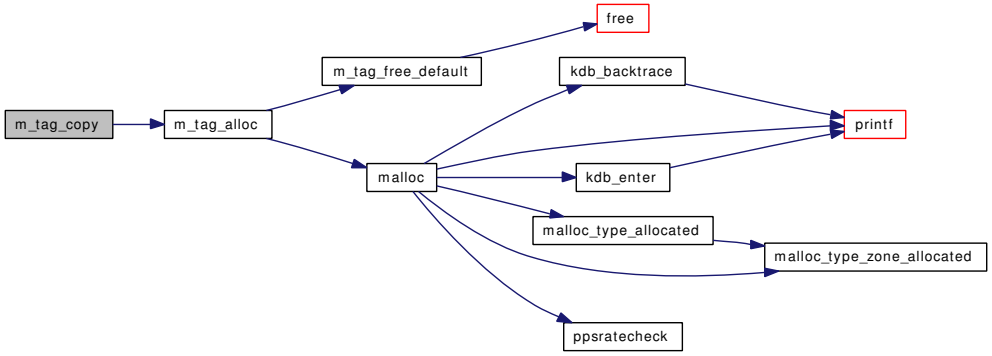
9.143.1.5 struct m_tag* m_tag_copy (struct m_tag * *t*, int *how*)

Definition at line 400 of file uipc_mbuf2.c.

References m_tag_alloc().

Referenced by m_tag_copy_chain().

Here is the call graph for this function:



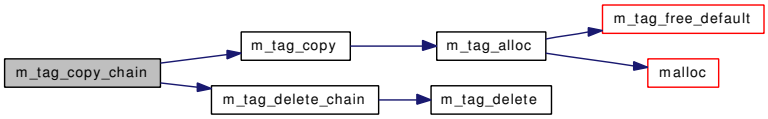
9.143.1.6 int m_tag_copy_chain (struct mbuf * *to*, struct mbuf * *from*, int *how*)

Definition at line 434 of file uipc_mbuf2.c.

References m_tag_copy(), and m_tag_delete_chain().

Referenced by m_dup_pkthdr().

Here is the call graph for this function:



9.143.1.7 void m_tag_delete (struct mbuf * *m*, struct m_tag * *t*)

Definition at line 336 of file uipc_mbuf2.c.

Referenced by m_tag_delete_chain(), and m_tag_delete_nonpersistent().

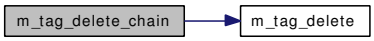
9.143.1.8 void m_tag_delete_chain (struct mbuf * *m*, struct m_tag * *t*)

Definition at line 346 of file uipc_mbuf2.c.

References m_tag_delete().

Referenced by m_demote(), m_dup_pkthdr(), m_move_pkthdr(), m_tag_copy_chain(), mb_dtor_mbuf(), and mb_dtor_pack().

Here is the call graph for this function:



9.143.1.9 void m_tag_delete_nonpersistent (struct mbuf * *m*)

Definition at line 370 of file uipc_mbuf2.c.

References m_tag_delete().

Here is the call graph for this function:

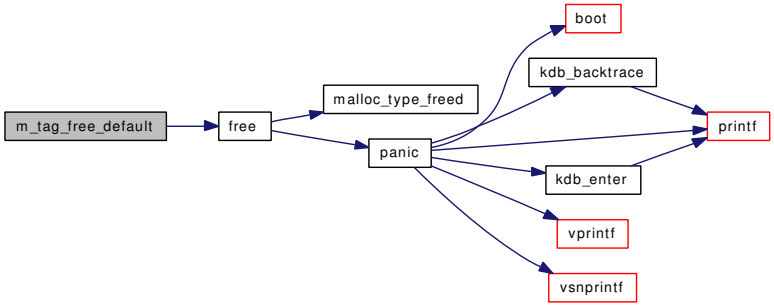
9.143.1.10 void m_tag_free_default (struct m_tag * *t*)

Definition at line 308 of file uipc_mbuf2.c.

References free().

Referenced by m_tag_alloc().

Here is the call graph for this function:



9.143.1.11 struct m_tag* m_tag_locate (struct mbuf * *m*, u_int32_t *cookie*, int *type*, struct m_tag * *t*)

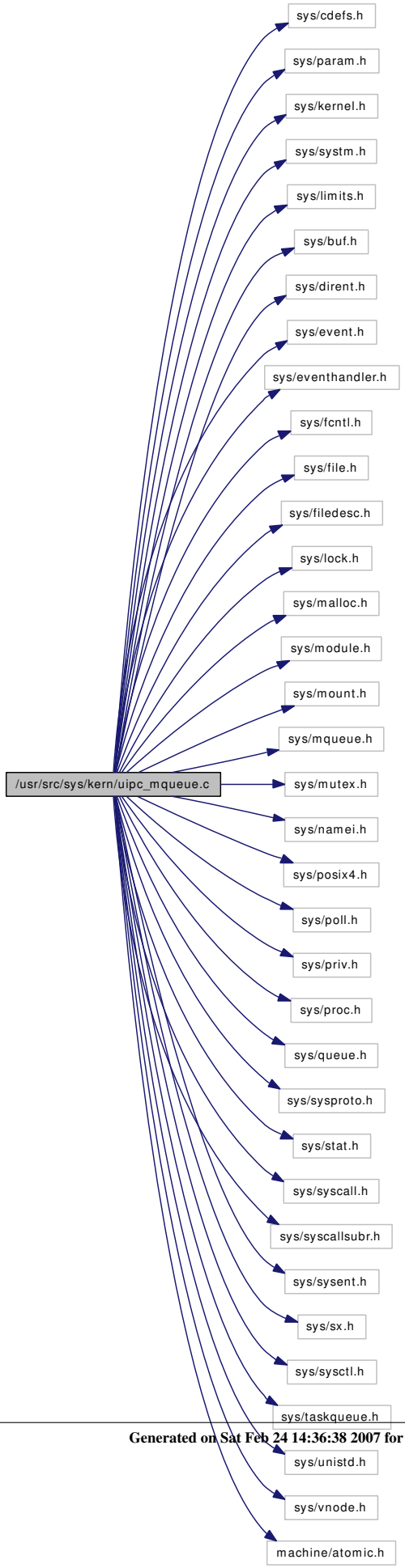
Definition at line 381 of file uipc_mbuf2.c.

9.143.1.12 static MALLOC_DEFINE (M_PACKET_TAGS, MBUF_TAG_MEM_NAME, "packet-attached information") [static]

9.144 /usr/src/sys/kern/uipc_mqueue.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/kernel.h>
#include <sys/systm.h>
#include <sys/limits.h>
#include <sys/buf.h>
#include <sys/dirent.h>
#include <sys/event.h>
#include <sys/eventhandler.h>
#include <sys/fcntl.h>
#include <sys/file.h>
#include <sys/filedesc.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/module.h>
#include <sys/mount.h>
#include <sys/mqueue.h>
#include <sys/mutex.h>
#include <sys/namei.h>
#include <sys/posix4.h>
#include <sys/poll.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/queue.h>
#include <sys/sysproto.h>
#include <sys/stat.h>
#include <sys/syscall.h>
#include <sys/syscallsubr.h>
#include <sys/sysent.h>
#include <sys/sx.h>
#include <sys/sysctl.h>
#include <sys/taskqueue.h>
#include <sys/unistd.h>
#include <sys/vnode.h>
#include <machine/atomic.h>
```

Include dependency graph for uipc_mqueue.c:


/usr/src/sys/kern/uipc_mqueue.c
sys/cdefs.h
sys/param.h
sys/kernel.h
sys/systm.h
sys/limits.h
sys/buf.h
sys/dirent.h
sys/event.h
sys/eventhandler.h
sys/fcntl.h
sys/file.h
sys/filedesc.h
sys/lock.h
sys/malloc.h
sys/module.h
sys/mount.h
sys/mqueue.h
sys/mutex.h
sys/namei.h
sys/posix4.h
sys/poll.h
sys/priv.h
sys/proc.h
sys/queue.h
sys/sysproto.h
sys/stat.h
sys/syscall.h
sys/syscallsubr.h
sys/sysent.h
sys/sx.h
sys/sysctl.h
sys/taskqueue.h
sys/unistd.h
sys/vnode.h
machine/atomic.h

Data Structures

- struct mqfs_info
- struct mqfs_vdata
- struct mqfs_node
- struct mqueue_notifier
- struct mqueue
- struct mqueue_msg

Defines

- #define MQFS_NAMELEN NAME_MAX
- #define MQFS_DELEN (8 + MQFS_NAMELEN)
- #define VTON(vp) (((struct mqfs_vdata *)((vp) → v_data)) → mv_node)
- #define VTOMQ(vp) ((struct mqueue *)(VTON(vp) → mn_data))
- #define VFSTOMQFS(m) ((struct mqfs_info *)((m) → mnt_data))
- #define FPTOMQ(fp)
- #define MQ_RSEL 0x01
- #define MQ_WSEL 0x02

Typedefs

- typedef int(*) _fgetf (struct thread *, int, struct file **)

Enumerations

- enum mqfs_type_t {
  mqfstype_none = 0, mqfstype_root, mqfstype_dir, mqfstype_this,
  mqfstype_parent, mqfstype_file, mqfstype_symlink }

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/uipc_mqueue.c,v 1.18 2006/11/11 16:26:56 trhodes Exp $")
- TAILQ_HEAD (msgq, mqueue_msg)
- SYSCTL_NODE (_kern, OID_AUTO, mqueue, CTLFLAG_RW, 0,"POSIX real time message queue")
- SYSCTL_INT (_kern_mqueue, OID_AUTO, maxmsg, CTLFLAG_RW,&maxmsg, 0,"Default maximum messages in queue")
- SYSCTL_INT (_kern_mqueue, OID_AUTO, maxmsgsize, CTLFLAG_RW,&maxmsgsize, 0,"Default maximum message size")
- SYSCTL_INT (_kern_mqueue, OID_AUTO, maxmq, CTLFLAG_RW,&maxmq, 0,"maximum message queues")
- SYSCTL_INT (_kern_mqueue, OID_AUTO, curmq, CTLFLAG_RW,&curmq, 0,"current message queue number")
- static MALLOC_DEFINE (M_MQUEUEDATA,"mqdata","mqueue data")
- static struct mqfs_node * mqfs_create_file (struct mqfs_node *parent, const char *name, int namelen, struct ucred *cred, int mode)
- static int mqfs_destroy (struct mqfs_node *mn)
- static void mqfs_fileno_alloc (struct mqfs_info *mi, struct mqfs_node *mn)

- static void mqfs_fileno_free (struct mqfs_info *mi, struct mqfs_node *mn)
- static int mqfs_allocv (struct mount *mp, struct vnode **vpp, struct mqfs_node *pn)
- static struct mqueue * mqueue_alloc (const struct mq_attr *attr)
- static void mqueue_free (struct mqueue *mq)
- static int mqueue_send (struct mqueue *mq, const char *msg_ptr, size_t msg_len, unsigned msg_-prio, int waitok, const struct timespec *abs_timeout)
- static int mqueue_receive (struct mqueue *mq, char *msg_ptr, size_t msg_len, unsigned *msg_prio, int waitok, const struct timespec *abs_timeout)
- static int _mqueue_send (struct mqueue *mq, struct mqueue_msg *msg, int timo)
- static int _mqueue_recv (struct mqueue *mq, struct mqueue_msg **msg, int timo)
- static void mqueue_send_notification (struct mqueue *mq)
- static void mqueue_fdclose (struct thread *td, int fd, struct file *fp)
- static void mq_proc_exit (void *arg, struct proc *p)
- static void filt_mqdetach (struct knote *kn)
- static int filt_mqread (struct knote *kn, long hint)
- static int filt_mqwrite (struct knote *kn, long hint)
- static void mqfs_fileno_init (struct mqfs_info *mi)
- static void mqfs_fileno_uninit (struct mqfs_info *mi)
- static __inline struct mqfs_node * mqnode_alloc (void)
- static __inline void mqnode_free (struct mqfs_node *node)
- static __inline void mqnode_addref (struct mqfs_node *node)
- static __inline void mqnode_release (struct mqfs_node *node)
- static int mqfs_add_node (struct mqfs_node *parent, struct mqfs_node *node)
- static struct mqfs_node * mqfs_create_node (const char *name, int namelen, struct ucred *cred, int mode, int nodetype)
- static int mqfs_fixup_dir (struct mqfs_node *parent)
- static int mqfs_mount (struct mount *mp, struct thread *td)
- static int mqfs_unmount (struct mount *mp, int mntflags, struct thread *td)
- static int mqfs_root (struct mount *mp, int flags, struct vnode **vpp, struct thread *td)
- static int mqfs_statfs (struct mount *mp, struct statfs *sbp, struct thread *td)
- static int mqfs_init (struct vfsconf *vfc)
- static int mqfs_uninit (struct vfsconf *vfc)
- static void do_recycle (void *context, int pending __unused)
- static struct mqfs_node * mqfs_search (struct mqfs_node *pd, const char *name, int len)
- static int mqfs_lookupx (struct vop_cachedlookup_args *ap)
- static int mqfs_lookup (struct vop_cachedlookup_args *ap)
- static int mqfs_create (struct vop_create_args *ap)
- static int do_unlink (struct mqfs_node *pn, struct ucred *ucred)
- static int mqfs_remove (struct vop_remove_args *ap)
- static int mqfs_inactive (struct vop_inactive_args *ap)
- static int mqfs_reclaim (struct vop_reclaim_args *ap)
- static int mqfs_open (struct vop_open_args *ap)
- static int mqfs_close (struct vop_close_args *ap)
- static int mqfs_access (struct vop_access_args *ap)
- static int mqfs_getattr (struct vop_getattr_args *ap)
- static int mqfs_setattr (struct vop_setattr_args *ap)
- static int mqfs_read (struct vop_read_args *ap)
- static int mqfs_readdir (struct vop_readdir_args *ap)
- static struct mqueue_msg * mqueue_loadmsg (const char *msg_ptr, size_t msg_size, int msg_prio)
- static int mqueue_savemsg (struct mqueue_msg *msg, char *msg_ptr, int *msg_prio)

- static __inline void mqueue_freemsg (struct mqueue_msg *msg)
- static __inline struct mqueue_notifier * notifier_alloc (void)
- static __inline void notifier_free (struct mqueue_notifier *p)
- static struct mqueue_notifier * notifier_search (struct proc *p, int fd)
- static __inline void notifier_insert (struct proc *p, struct mqueue_notifier *nt)
- static __inline void notifier_delete (struct proc *p, struct mqueue_notifier *nt)
- static void notifier_remove (struct proc *p, struct mqueue *mq, int fd)
- int kmq_open (struct thread *td, struct kmq_open_args *uap)
- int kmq_unlink (struct thread *td, struct kmq_unlink_args *uap)
- static int _getmq (struct thread *td, int fd, _fgetf func, struct file **fpp, struct mqfs_node **ppn, struct mqueue **pmq)
- static __inline int getmq (struct thread *td, int fd, struct file **fpp, struct mqfs_node **ppn, struct mqueue **pmq)
- static __inline int getmq_read (struct thread *td, int fd, struct file **fpp, struct mqfs_node **ppn, struct mqueue **pmq)
- static __inline int getmq_write (struct thread *td, int fd, struct file **fpp, struct mqfs_node **ppn, struct mqueue **pmq)
- int kmq_setattr (struct thread *td, struct kmq_setattr_args *uap)
- int kmq_timedreceive (struct thread *td, struct kmq_timedreceive_args *uap)
- int kmq_timedsend (struct thread *td, struct kmq_timedsend_args *uap)
- int kmq_notify (struct thread *td, struct kmq_notify_args *uap)
- static void mq_proc_exit (void *arg __unused, struct proc *p)
- static int mqf_read (struct file *fp, struct uio *uio, struct ucred *active_cred, int flags, struct thread *td)
- static int mqf_write (struct file *fp, struct uio *uio, struct ucred *active_cred, int flags, struct thread *td)
- static int mqf_ioctl (struct file *fp, u_long cmd, void *data, struct ucred *active_cred, struct thread *td)
- static int mqf_poll (struct file *fp, int events, struct ucred *active_cred, struct thread *td)
- static int mqf_close (struct file *fp, struct thread *td)
- static int mqf_stat (struct file *fp, struct stat *st, struct ucred *active_cred, struct thread *td)
- static int mqf_kqfilter (struct file *fp, struct knote *kn)
- SYSCALL_MODULE_HELPER (kmq_open)
- SYSCALL_MODULE_HELPER (kmq_setattr)
- SYSCALL_MODULE_HELPER (kmq_timedsend)
- SYSCALL_MODULE_HELPER (kmq_timedreceive)
- SYSCALL_MODULE_HELPER (kmq_notify)
- SYSCALL_MODULE_HELPER (kmq_unlink)
- VFS_SET (mqfs_vfsops, mqueuefs, VFCF_SYNTHETIC)
- MODULE_VERSION (mqueuefs, 1)

Variables

- static int default_maxmsg = 10
- static int default_msgsize = 1024
- static int maxmsg = 100
- static int maxmsgsize = 16384
- static int maxmq = 100
- static int curmq = 0
- static int unloadable = 0

- static eventhandler_tag exit_tag
- static struct mqfs_info mqfs_data
- static uma_zone_t mqnode_zone
- static uma_zone_t mqueue_zone
- static uma_zone_t mvdata_zone
- static uma_zone_t mqnoti_zone
- static struct vop_vector mqfs_vnodeops
- static struct fileops mqueueops
- filterops mq_rfiltops
- filterops mq_wfiltops
- static struct fileops mqueueops
- static struct vop_vector mqfs_vnodeops
- static struct vfsops mqfs_vfsops

9.144.1 Define Documentation

9.144.1.1 #define FPTOMQ(fp)

Value:

```
((struct mqueue *)(((struct mqfs_node *) \
                                (fp)->f_data)->mn_data))
```

Definition at line 146 of file uipc_mqueue.c.

Referenced by filt_mqdetach(), filt_mqread(), filt_mqwrite(), mq_proc_exit(), mqf_kqfilter(), mqf_poll(), and mqueue_fdclose().

9.144.1.2 #define MQ_RSEL 0x01

Definition at line 175 of file uipc_mqueue.c.

Referenced by _mqueue_send(), mqf_poll(), and mqueue_fdclose().

9.144.1.3 #define MQ_WSEL 0x02

Definition at line 176 of file uipc_mqueue.c.

Referenced by _mqueue_recv(), mqf_poll(), and mqueue_fdclose().

9.144.1.4 #define MQFS_DELEN (8 + MQFS_NAMELEN)

Definition at line 88 of file uipc_mqueue.c.

9.144.1.5 #define MQFS_NAMELEN NAME_MAX

Definition at line 87 of file uipc_mqueue.c.

Referenced by kmq_open(), kmq_unlink(), mqfs_lookupx(), and mqfs_readdir().

9.144.1.6 #define VFSTOMQFS(m) ((struct mqfs_info *)((m) → mnt_data))

Definition at line 145 of file uipc_mqueue.c.

Referenced by mqfs_create(), mqfs_lookup(), mqfs_readdir(), mqfs_reclaim(), mqfs_remove(), and mqfs_root().

9.144.1.7 #define VTOMQ(vp) ((struct mqueue *)(VTON(vp) → mn_data))

Definition at line 144 of file uipc_mqueue.c.

Referenced by mqfs_read().

9.144.1.8 #define VTON(vp) (((struct mqfs_vdata *)((vp) → v_data)) → mv_node)

Definition at line 143 of file uipc_mqueue.c.

Referenced by mqfs_create(), mqfs_getattr(), mqfs_inactive(), mqfs_lookupx(), mqfs_read(), mqfs_readdir(), mqfs_remove(), and mqfs_setattr().

9.144.2 Typedef Documentation

9.144.2.1 typedef int(*) _fgetf(struct thread *, int, struct file **)

Definition at line 2048 of file uipc_mqueue.c.

9.144.3 Enumeration Type Documentation

9.144.3.1 enum mqfs_type_t

Enumerator:

mqfstype_none
mqfstype_root
mqfstype_dir
mqfstype_this
mqfstype_parent
mqfstype_file
mqfstype_symlink

Definition at line 91 of file uipc_mqueue.c.

9.144.4 Function Documentation

9.144.4.1 __FBSDID ("$FreeBSD: src/sys/kern/uipc_mqueue. *c*, v 1.18 2006/11/11 16:26:56 trhodes Exp $")

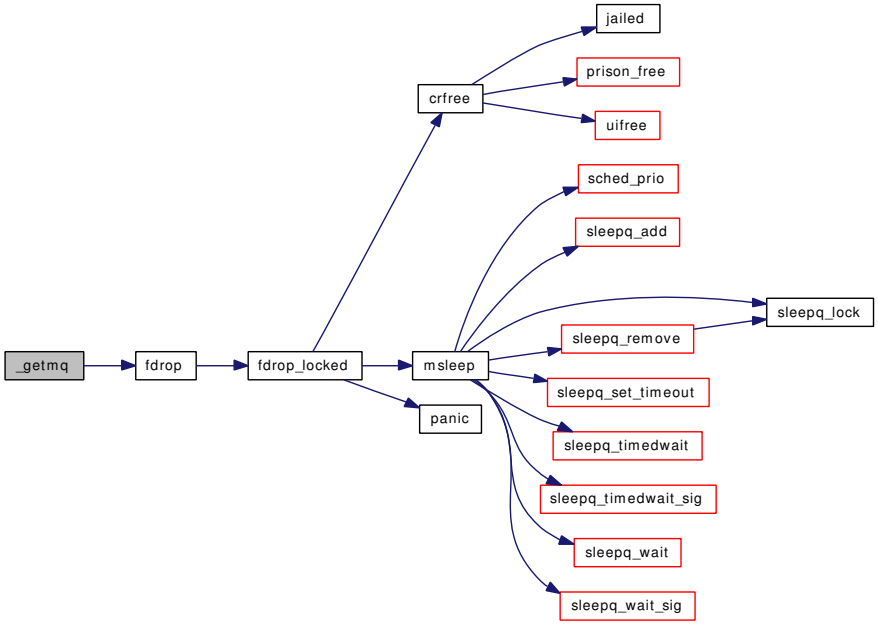
9.144.4.2 static int _getmq (struct thread * *td*, int *fd*, _fgetf *func*, struct file ** *fpp*, struct mqfs_node ** *ppn*, struct mqueue ** *pmq*) [static]

Definition at line 2054 of file uipc_mqueue.c.

References fdrop().

Referenced by getmq(), getmq_read(), and getmq_write().

Here is the call graph for this function:



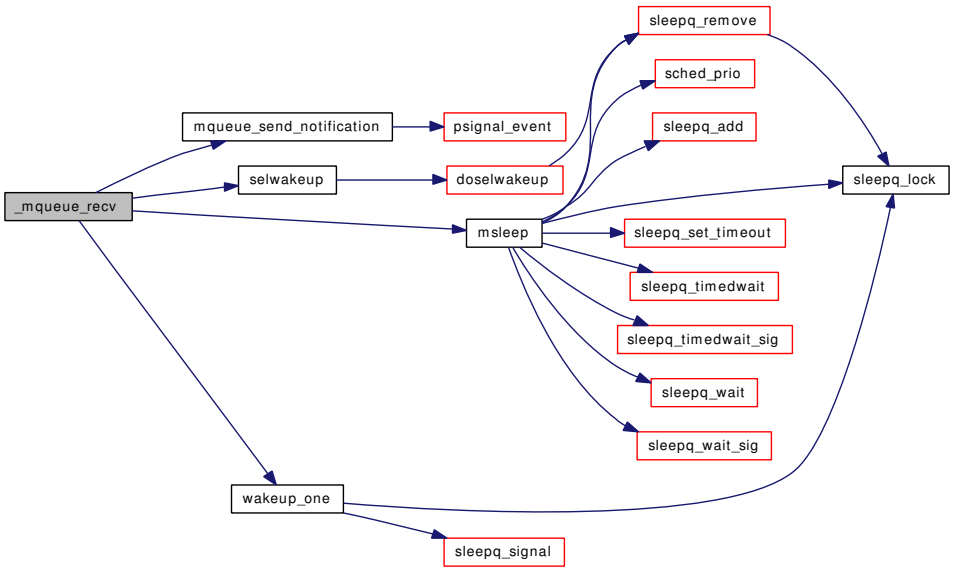
9.144.4.3 static int _mqueue_recv (struct mqueue * *mq*, struct mqueue_msg ** *msg*, int *timo*) [static]

Definition at line 1814 of file uipc_mqueue.c.

References mqueue::mq_curmsgs, mqueue::mq_flags, mqueue::mq_msgq, mqueue::mq_mutex, mqueue::mq_notifier, mqueue::mq_receivers, mqueue::mq_senders, mqueue::mq_totalbytes, mqueue::mq_wsel, MQ_WSEL, mqueue_send_notification(), msleep(), selwakeup(), and wakeup_one().

Referenced by mqueue_receive().

Here is the call graph for this function:



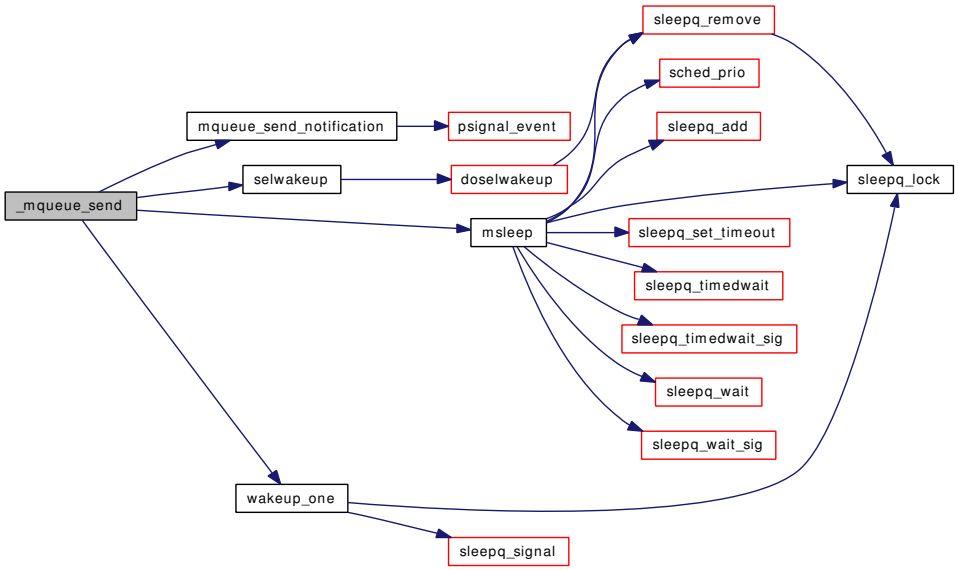
9.144.4.4 static int _mqueue_send (struct mqueue * *mq*, struct mqueue_msg * *msg*, int *timo*) [static]

Definition at line 1661 of file uipc_mqueue.c.

References mqueue::mq_curmsgs, mqueue::mq_maxmsg, mqueue::mq_msgq, mqueue::mq_mutex, mqueue::mq_notifier, mqueue::mq_receivers, MQ_RSEL, mqueue::mq_senders, mqueue::mq_totalbytes, mqueue_send_notification(), msleep(), selwakeup(), and wakeup_one().

Referenced by mqueue_send().

Here is the call graph for this function:



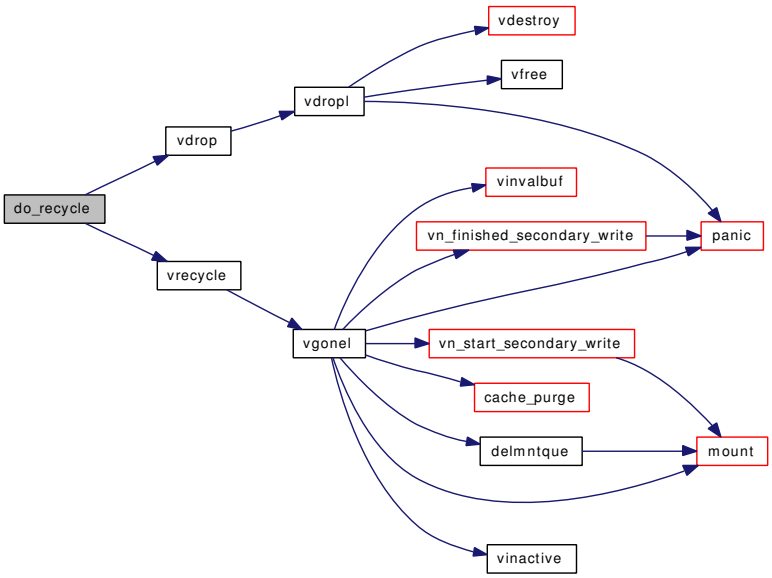
9.144.4.5 static void do_recycle (void * *context*, int pending *__unused*) [static]

Definition at line 685 of file uipc_mqueue.c.

References vdrop(), and vrecycle().

Referenced by mqfs_allocv().

Here is the call graph for this function:



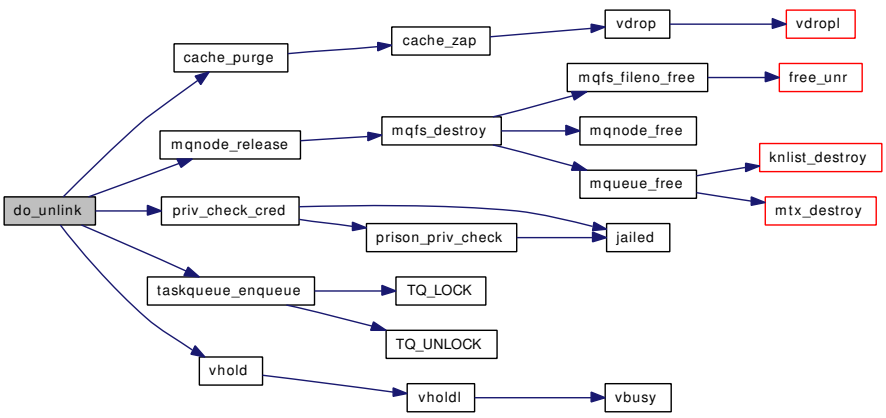
9.144.4.6 static int do_unlink (struct mqfs_node * *pn*, struct ucred * *ucred*) [static]

Definition at line 951 of file uipc_mqueue.c.

References cache_purge(), mqfs_info::mi_lock, mqfs_node::mn_info, mqfs_node::mn_parent, mqnode_-release(), priv_check_cred(), taskqueue_enqueue(), and vhold().

Referenced by kmq_unlink(), and mqfs_remove().

Here is the call graph for this function:

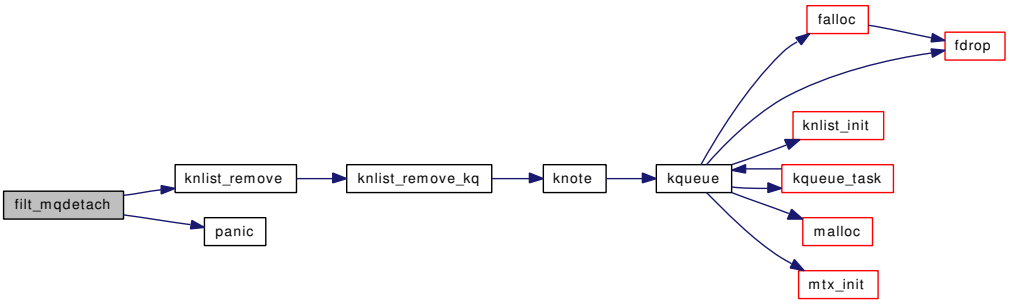


9.144.4.7 static void filt_mqdetach (struct knote * *kn*) [static]

Definition at line 2419 of file uipc_mqueue.c.

References FPTOMQ, knlist_remove(), mqueue::mq_rsel, mqueue::mq_wsel, and panic().

Here is the call graph for this function:



9.144.4.8 static int filt_mqread (struct knote * *kn*, long *hint*) [static]

Definition at line 2432 of file uipc_mqueue.c.

References FPTOMQ, mqueue::mq_curmsgs, and mqueue::mq_mutex.

9.144.4.9 static int filt_mqwrite (struct knote * *kn*, long *hint*) [static]

Definition at line 2441 of file uipc_mqueue.c.

References FPTOMQ, mqueue::mq_curmsgs, mqueue::mq_maxmsg, and mqueue::mq_mutex.

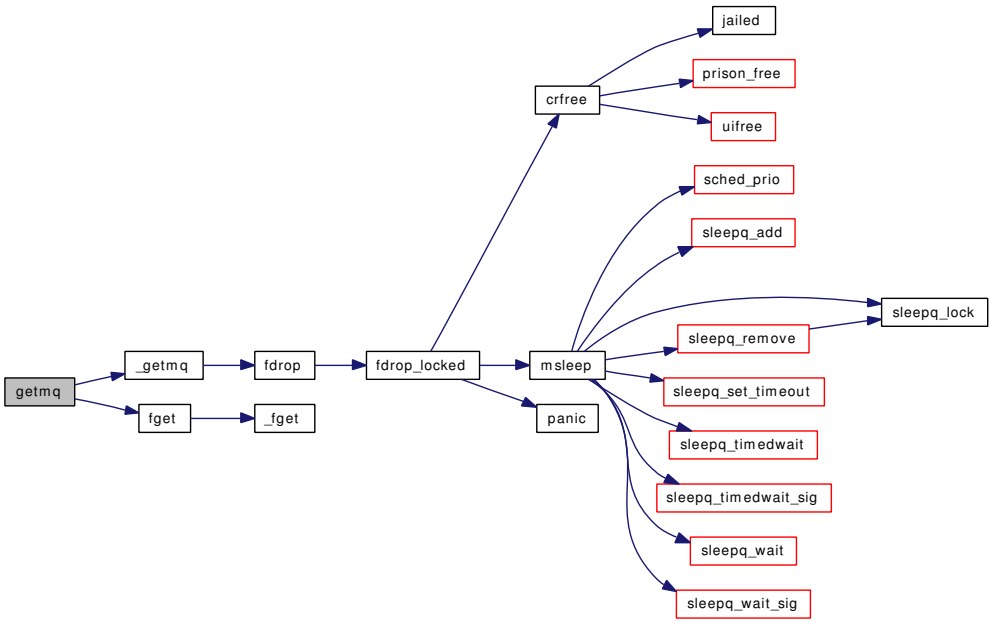
9.144.4.10 static __inline int getmq (struct thread * *td*, int *fd*, struct file ** *fpp*, struct mqfs_node ** *ppn*, struct mqueue ** *pmq*) [static]

Definition at line 2076 of file uipc_mqueue.c.

References _getmq(), and fget().

Referenced by kmq_notify(), and kmq_setattr().

Here is the call graph for this function:



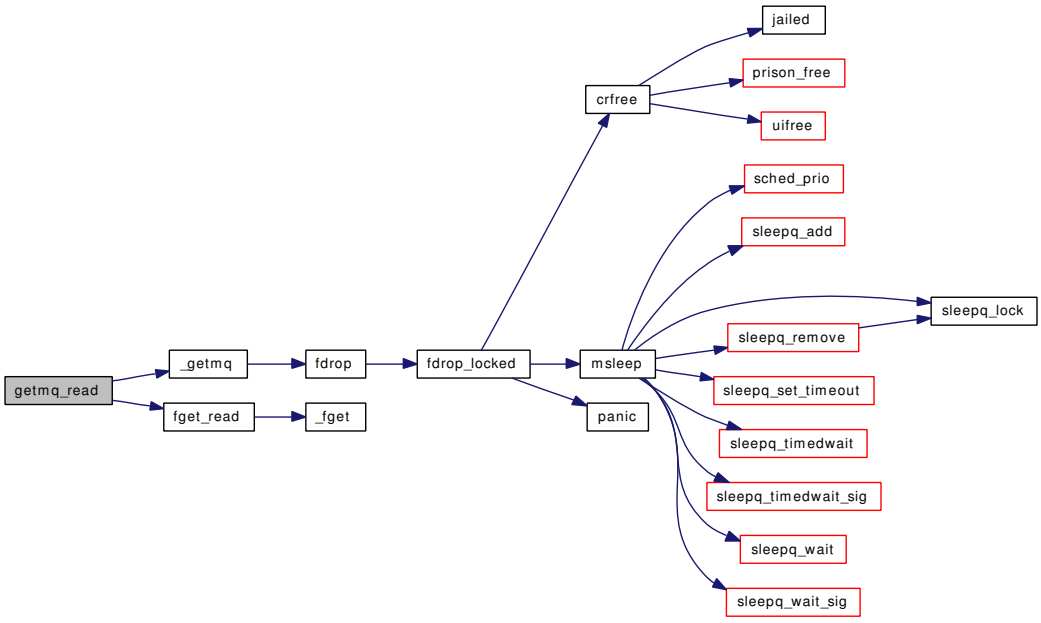
9.144.4.11 static __inline int getmq_read (struct thread * *td*, int *fd*, struct file ** *fpp*, struct mqfs_node ** *ppn*, struct mqueue ** *pmq*) [static]

Definition at line 2083 of file uipc_mqueue.c.

References _getmq(), and fget_read().

Referenced by kmq_timedreceive().

Here is the call graph for this function:



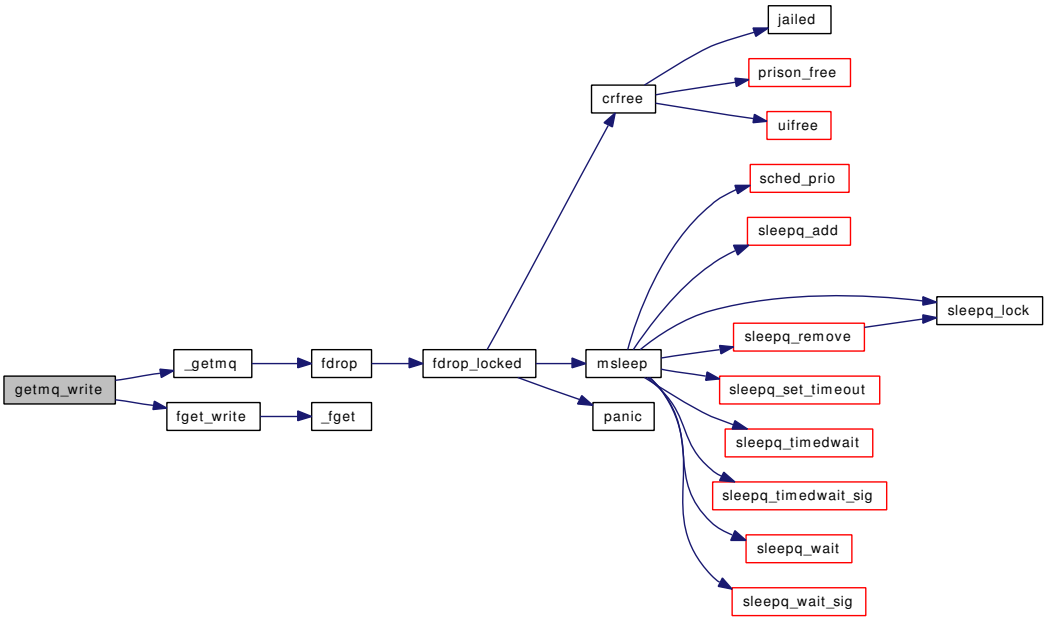
9.144.4.12 static __inline int getmq_write (struct thread * *td*, int *fd*, struct file ** *fpp*, struct mqfs_node ** *ppn*, struct mqueue ** *pmq*) [static]

Definition at line 2090 of file uipc_mqueue.c.

References _getmq(), and fget_write().

Referenced by kmq_timedsend().

Here is the call graph for this function:

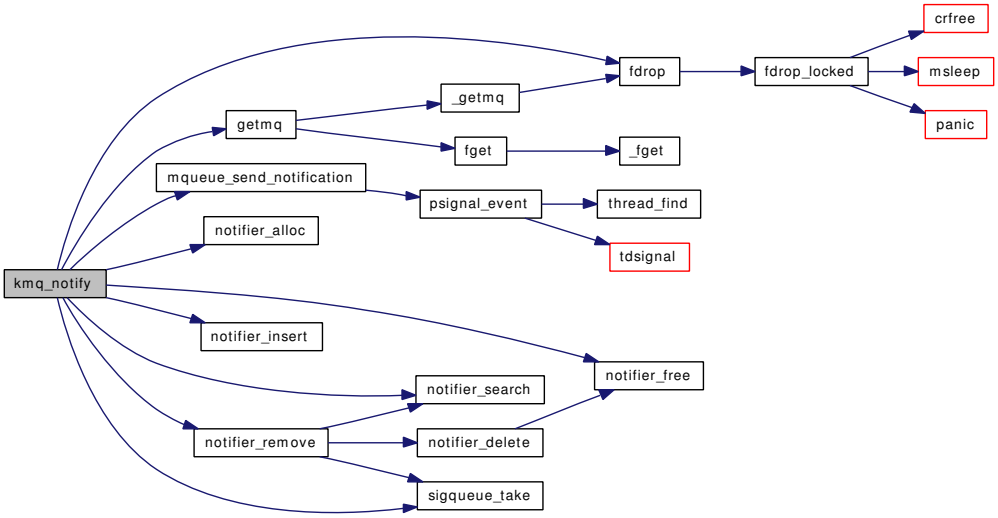


9.144.4.13 int kmq_notify (struct thread * *td*, struct kmq_notify_args * *uap*)

Definition at line 2178 of file uipc_mqueue.c.

References fdrop(), getmq(), mqueue::mq_msgq, mqueue::mq_mutex, mqueue::mq_notifier, mqueue::mq_receivers, mqueue_send_notification(), notifier_alloc(), notifier_free(), notifier_insert(), notifier_remove(), notifier_search(), and sigqueue_take().

Here is the call graph for this function:

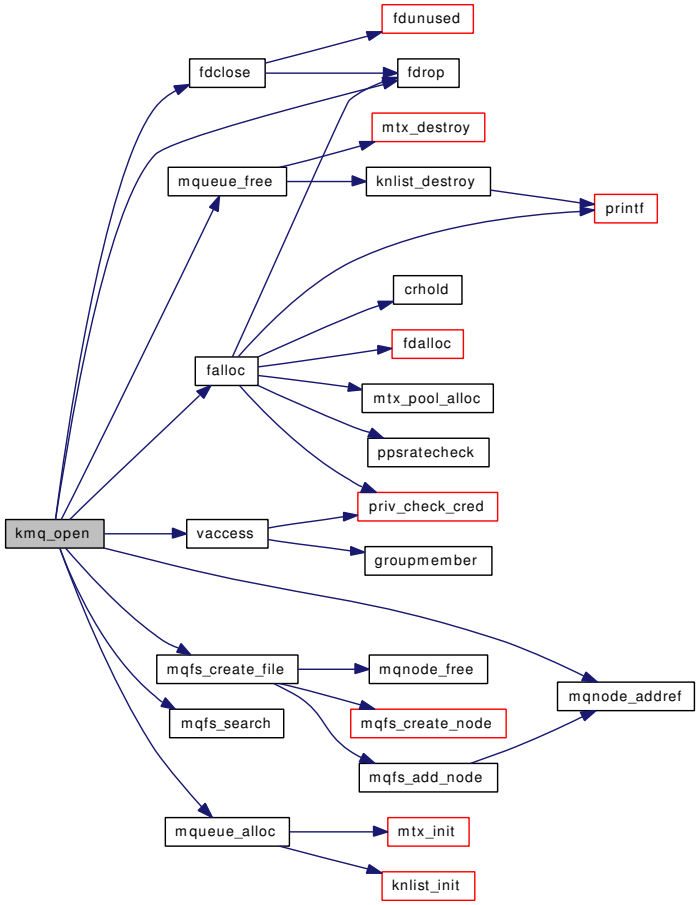


9.144.4.14 int kmq_open (struct thread * *td*, struct kmq_open_args * *uap*)

Definition at line 1910 of file uipc_mqueue.c.

References falloc(), fdclose(), fdrop(), maxmsg, maxmsgsize, mqfs_info::mi_lock, mqfs_info::mi_root, mqfs_create_file(), mqfs_data, MQFS_NAMELEN, mqfs_search(), mqnode_addref(), mqueue_alloc(), mqueue_free(), and vaccess().

Here is the call graph for this function:

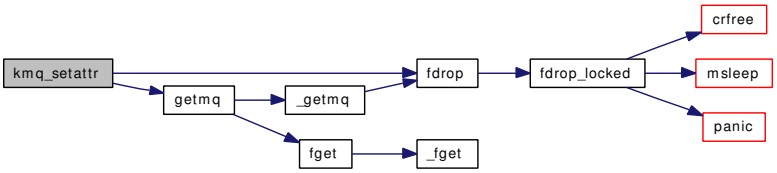


9.144.4.15 int kmq_setattr (struct thread * *td*, struct kmq_setattr_args * *uap*)

Definition at line 2100 of file uipc_mqueue.c.

References fdrop(), getmq(), mqueue::mq_curmsgs, mqueue::mq_maxmsg, and mqueue::mq_msgsize.

Here is the call graph for this function:

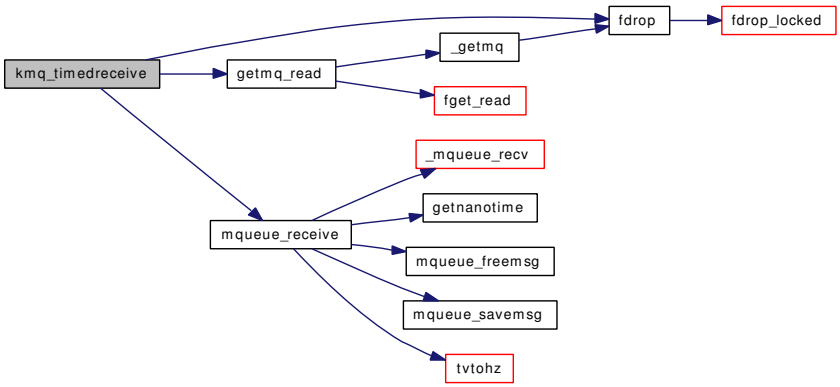


9.144.4.16 int kmq_timedreceive (struct thread * *td*, struct kmq_timedreceive_args * *uap*)

Definition at line 2137 of file uipc_mqueue.c.

References fdrop(), getmq_read(), and mqueue_receive().

Here is the call graph for this function:

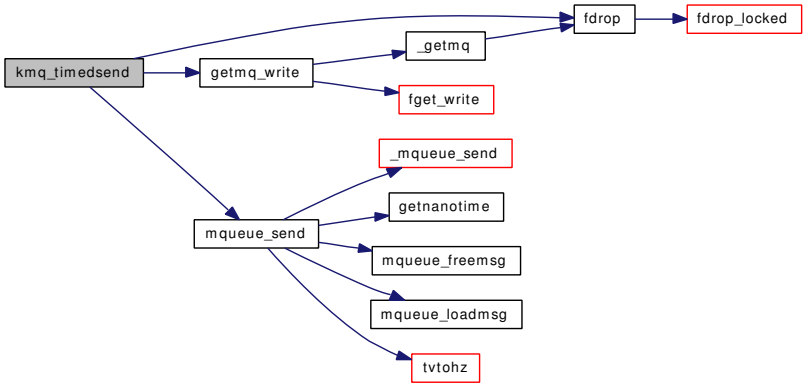


9.144.4.17 int kmq_timedsend (struct thread * *td*, struct kmq_timedsend_args * *uap*)

Definition at line 2158 of file uipc_mqueue.c.

References fdrop(), getmq_write(), and mqueue_send().

Here is the call graph for this function:

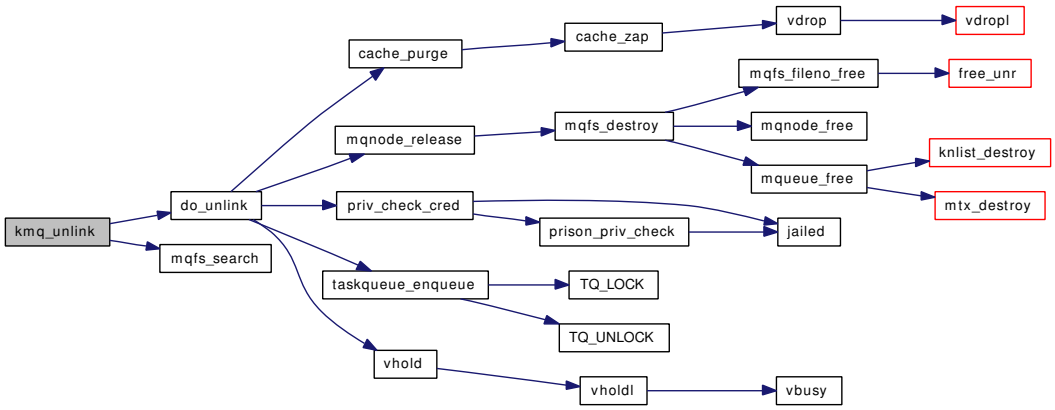


9.144.4.18 int kmq_unlink (struct thread * *td*, struct kmq_unlink_args * *uap*)

Definition at line 2024 of file uipc_mqueue.c.

References do_unlink(), mqfs_info::mi_lock, mqfs_info::mi_root, mqfs_data, MQFS_NAMELEN, and mqfs_search().

Here is the call graph for this function:



9.144.4.19 static MALLOC_DEFINE (M_MQUEUEDATA, "mqdata", "mqueue data") [static]

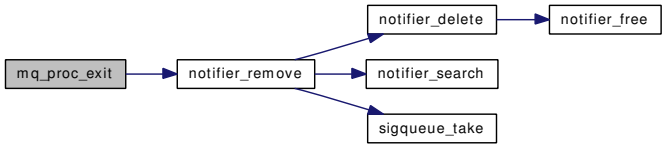
9.144.4.20 MODULE_VERSION (mqueuefs, 1)

9.144.4.21 static void mq_proc_exit (void *arg __*unused*, struct proc * *p*) [static]

Definition at line 2297 of file uipc_mqueue.c.

References FPTOMQ, mqueue::mq_mutex, and notifier_remove().

Here is the call graph for this function:



9.144.4.22 static void mq_proc_exit (void * *arg*, struct proc * *p*) [static]

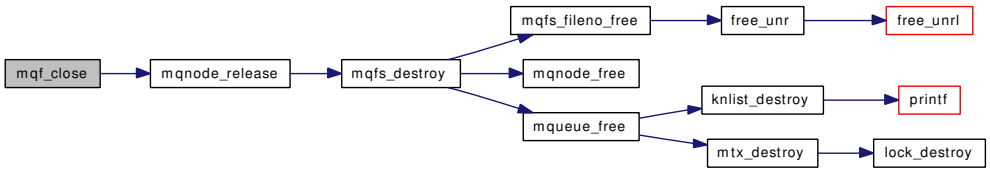
Referenced by mqfs_init().

9.144.4.23 static int mqf_close (struct file * *fp*, struct thread * *td*) [static]

Definition at line 2369 of file uipc_mqueue.c.

References badfileops, mqfs_info::mi_lock, mqfs_data, and mqnode_release().

Here is the call graph for this function:



9.144.4.24 static int mqf_ioctl (struct file * *fp*, u_long *cmd*, void * *data*, struct ucred * *active_cred*, struct thread * *td*) [static]

Definition at line 2334 of file uipc_mqueue.c.

9.144.4.25 static int mqf_kqfilter (struct file * *fp*, struct knote * *kn*) [static]

Definition at line 2402 of file uipc_mqueue.c.

References FPTOMQ, knlist_add(), mqueue::mq_rsel, and mqueue::mq_wsel.

Here is the call graph for this function:

9.144.4.26 static int mqf_poll (struct file * *fp*, int *events*, struct ucred * *active_cred*, struct thread * *td*) [static]

Definition at line 2341 of file uipc_mqueue.c.

References FPTOMQ, mqueue::mq_curmsgs, mqueue::mq_flags, mqueue::mq_maxmsg, mqueue::mq_-mutex, mqueue::mq_rsel, MQ_RSEL, mqueue::mq_wsel, MQ_WSEL, and selrecord().

Here is the call graph for this function:

9.144.4.27 static int mqf_read (struct file * *fp*, struct uio * *uio*, struct ucred * *active_cred*, int *flags*, struct thread * *td*) [static]

Definition at line 2320 of file uipc_mqueue.c.

9.144.4.28 static int mqf_stat (struct file * *fp*, struct stat * *st*, struct ucred * *active_cred*, struct thread * *td*) [static]

Definition at line 2385 of file uipc_mqueue.c.

9.144.4.29 static int mqf_write (struct file * *fp*, struct uio * *uio*, struct ucred * *active_cred*, int *flags*, struct thread * *td*) [static]

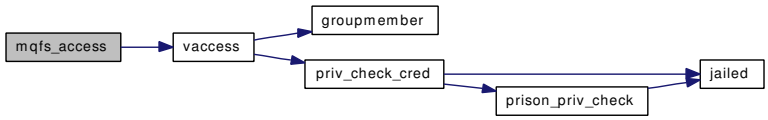
Definition at line 2327 of file uipc_mqueue.c.

9.144.4.30 static int mqfs_access (struct vop_access_args * *ap*) [static]

Definition at line 1101 of file uipc_mqueue.c.

References vaccess().

Here is the call graph for this function:



9.144.4.31 static int mqfs_add_node (struct mqfs_node * *parent*, struct mqfs_node * *node*) [static]

Definition at line 393 of file uipc_mqueue.c.

References mqfs_node::mn_info, mqfs_node::mn_parent, mqfstype_dir, mqfstype_root, and mqnode_addref().

Referenced by mqfs_create_file(), and mqfs_fixup_dir().

Here is the call graph for this function:

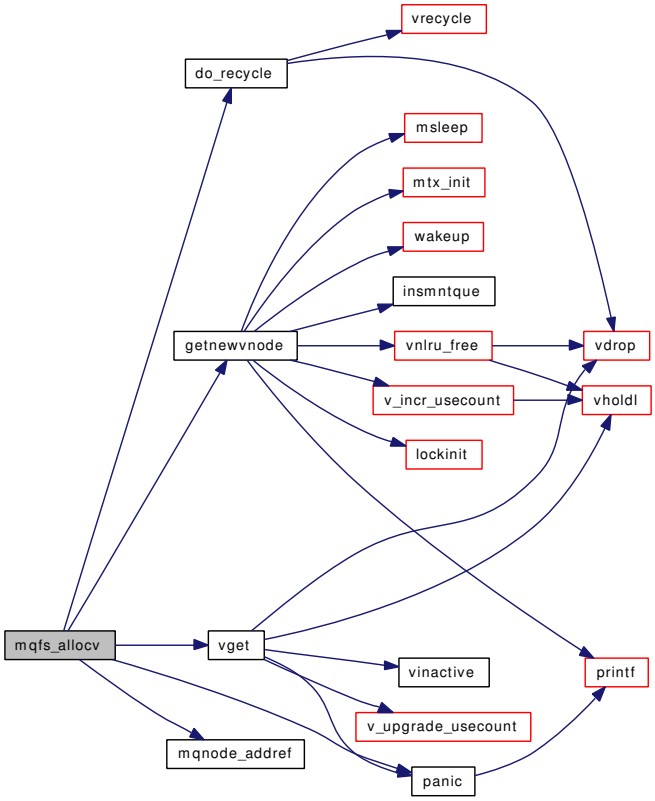
9.144.4.32 static int mqfs_allocv (struct mount * *mp*, struct vnode ** *vpp*, struct mqfs_node * *pn*) [static]

Definition at line 697 of file uipc_mqueue.c.

References do_recycle(), getnewvnode(), mqfs_node::mn_name, mqfstype_dir, mqfstype_file, mqfstype_none, mqfstype_parent, mqfstype_root, mqfstype_symlink, mqfstype_this, mqnode_addref(), mvdata_zone, panic(), and vget().

Referenced by mqfs_create(), mqfs_lookupx(), and mqfs_root().

Here is the call graph for this function:



9.144.4.33 static int mqfs_close (struct vop_close_args * *ap*) [static]

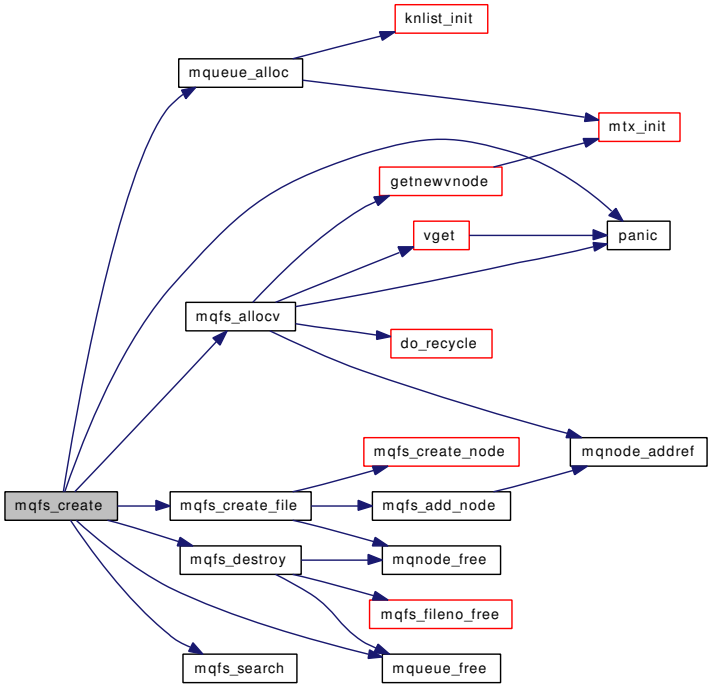
Definition at line 1082 of file uipc_mqueue.c.

9.144.4.34 static int mqfs_create (struct vop_create_args * *ap*) [static]

Definition at line 902 of file uipc_mqueue.c.

References mqfs_info::mi_lock, mqfs_allocv(), mqfs_create_file(), mqfs_destroy(), mqfs_search(), mqfstype_dir, mqfstype_root, mqueue_alloc(), mqueue_free(), panic(), VFSTOMQFS, and VTON.

Here is the call graph for this function:



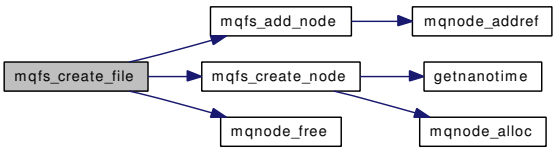
9.144.4.35 static struct mqfs_node * mqfs_create_file (struct mqfs_node * *parent*, const char * *name*, int *namelen*, struct ucred * *cred*, int *mode*) [static]

Definition at line 434 of file uipc_mqueue.c.

References mqfs_add_node(), mqfs_create_node(), mqfstype_file, and mqnode_free().

Referenced by kmq_open(), and mqfs_create().

Here is the call graph for this function:



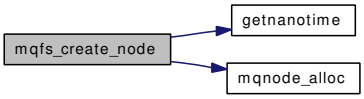
9.144.4.36 static struct mqfs_node* mqfs_create_node (const char * *name*, int *namelen*, struct ucred * *cred*, int *mode*, int *nodetype*) [static]

Definition at line 412 of file uipc_mqueue.c.

References getnanotime(), mqfs_node::mn_name, and mqnode_alloc().

Referenced by mqfs_create_file(), and mqfs_init().

Here is the call graph for this function:



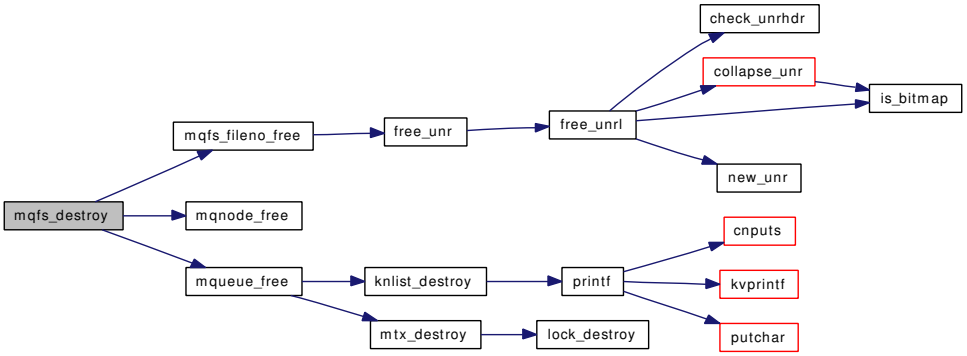
9.144.4.37 static int mqfs_destroy (struct mqfs_node * *mn*) [static]

Definition at line 524 of file uipc_mqueue.c.

References mqfs_node::mn_info, mqfs_node::mn_parent, mqfs_fileno_free(), mqfstype_dir, mqfstype_root, mqnode_free(), and mqueue_free().

Referenced by mqfs_create(), mqfs_uninit(), and mqnode_release().

Here is the call graph for this function:



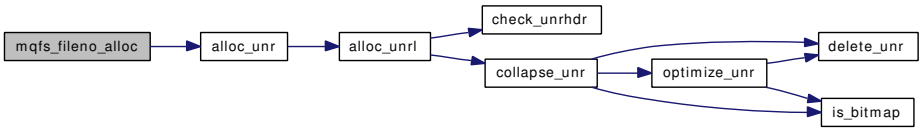
9.144.4.38 static void mqfs_fileno_alloc (struct mqfs_info * *mi*, struct mqfs_node * *mn*) [static]

Definition at line 294 of file uipc_mqueue.c.

References alloc_unr(), mqfs_info::mi_root, mqfs_info::mi_unrhdr, mqfs_node::mn_parent, mqfstype_dir, mqfstype_file, mqfstype_parent, mqfstype_root, mqfstype_symlink, and mqfstype_this.

Referenced by mqfs_init(), and mqfs_readdir().

Here is the call graph for this function:



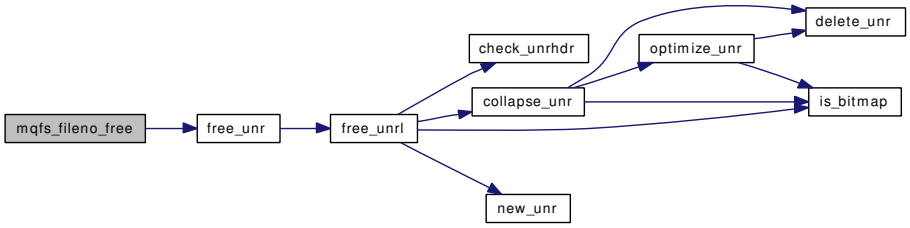
9.144.4.39 static void mqfs_fileno_free (struct mqfs_info * *mi*, struct mqfs_node * *mn*) [static]

Definition at line 335 of file uipc_mqueue.c.

References free_unr(), mqfs_info::mi_unrhdr, mqfstype_dir, mqfstype_file, mqfstype_parent, mqfstype_root, mqfstype_symlink, and mqfstype_this.

Referenced by mqfs_destroy().

Here is the call graph for this function:



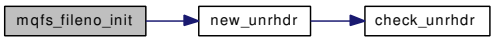
9.144.4.40 static void mqfs_fileno_init (struct mqfs_info * *mi*) [static]

Definition at line 269 of file uipc_mqueue.c.

References mqfs_info::mi_unrhdr, and new_unrhdr().

Referenced by mqfs_init().

Here is the call graph for this function:



9.144.4.41 static void mqfs_fileno_uninit (struct mqfs_info * *mi*) [static]

Definition at line 281 of file uipc_mqueue.c.

References delete_unrhdr(), and mqfs_info::mi_unrhdr.

Referenced by mqfs_uninit().

Here is the call graph for this function:

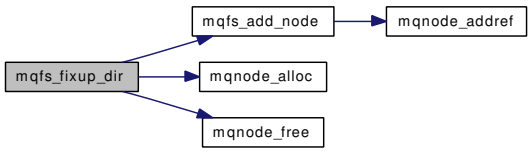
9.144.4.42 static int mqfs_fixup_dir (struct mqfs_node * *parent*) [static]

Definition at line 451 of file uipc_mqueue.c.

References mqfs_node::mn_name, mqfs_add_node(), mqfstype_parent, mqfstype_this, mqnode_alloc(), and mqnode_free().

Referenced by mqfs_init().

Here is the call graph for this function:



9.144.4.43 static int mqfs_getattr (struct vop_getattr_args * *ap*) [static]

Definition at line 1129 of file uipc_mqueue.c.

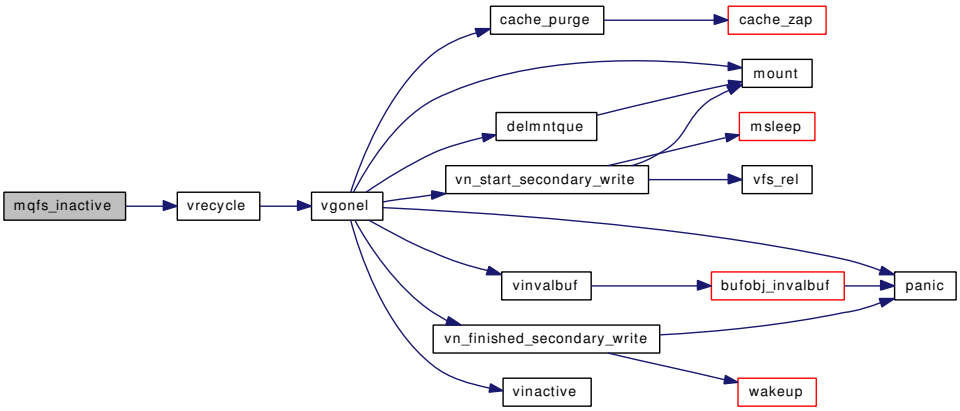
References VTON.

9.144.4.44 static int mqfs_inactive (struct vop_inactive_args * *ap*) [static]

Definition at line 1018 of file uipc_mqueue.c.

References vrecycle(), and VTON.

Here is the call graph for this function:

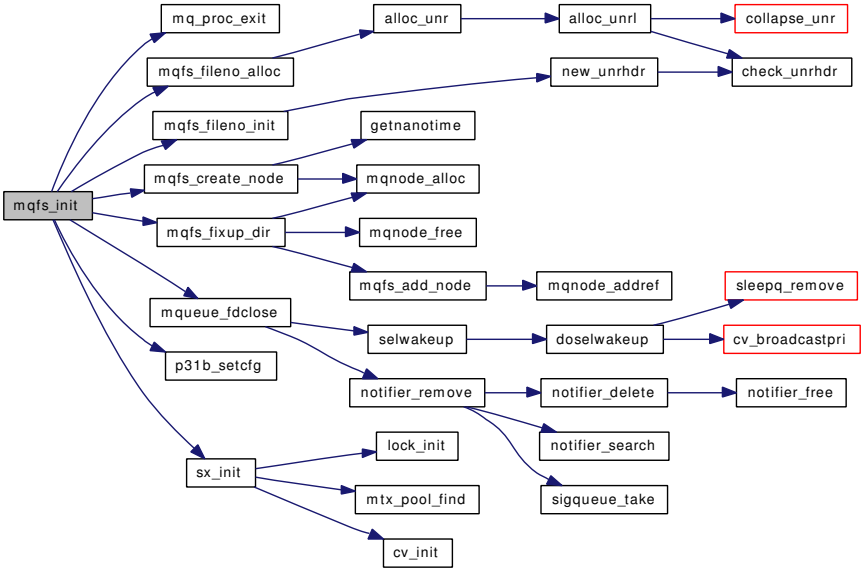


9.144.4.45 static int mqfs_init (struct vfsconf * *vfc*) [static]

Definition at line 625 of file uipc_mqueue.c.

References exit_tag, mqfs_info::mi_lock, mqfs_info::mi_root, mqfs_node::mn_info, mq_fdclose, mq_proc_exit(), mqfs_create_node(), mqfs_data, mqfs_fileno_alloc(), mqfs_fileno_init(), mqfs_fixup_dir(), mqfstype_root, mqnode_zone, mqnoti_zone, mqueue_fdclose(), mqueue_zone, mvdata_zone, p31b_setcfg(), and sx_init().

Here is the call graph for this function:

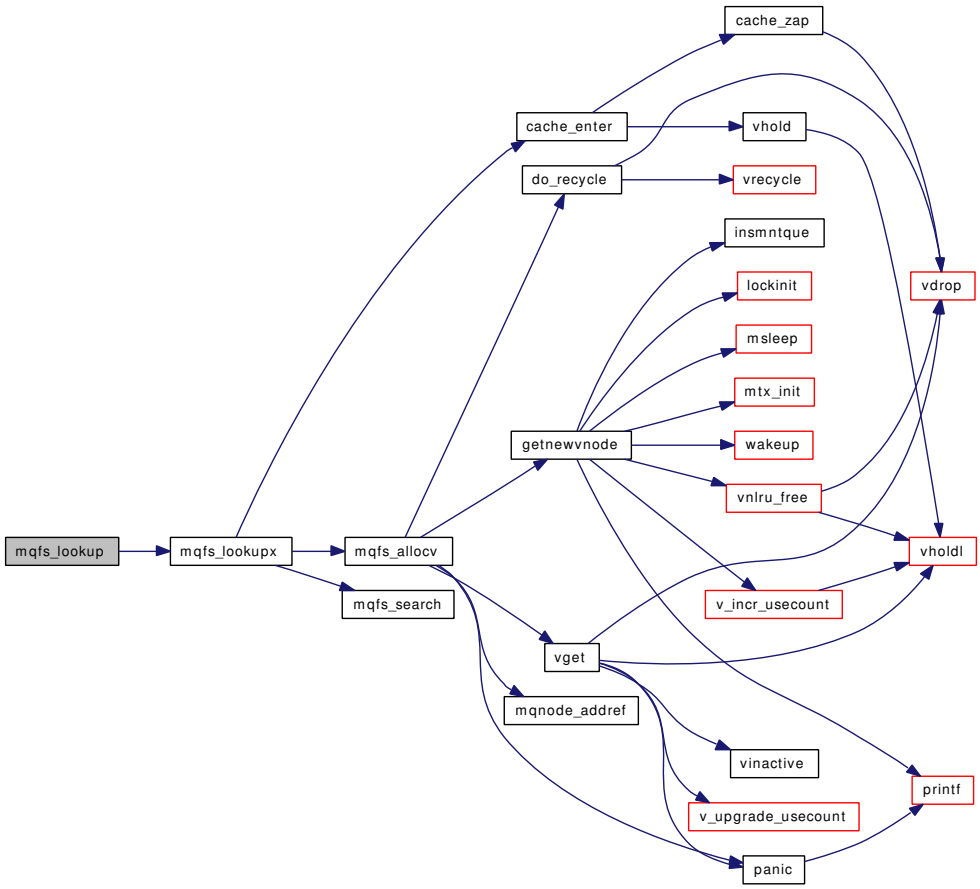


9.144.4.46 static int mqfs_lookup (struct vop_cachedlookup_args * *ap*) [static]

Definition at line 878 of file uipc_mqueue.c.

References mqfs_info::mi_lock, mqfs_lookupx(), and VFSTOMQFS.

Here is the call graph for this function:



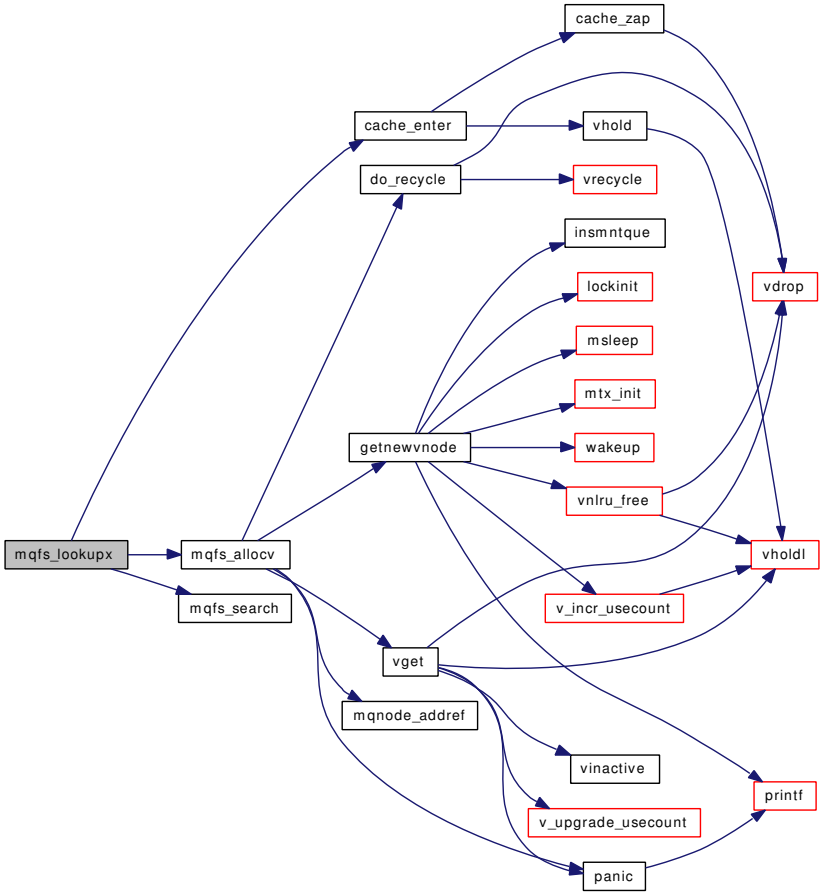
9.144.4.47 static int mqfs_lookupx (struct vop_cachedlookup_args * *ap*) [static]

Definition at line 770 of file uipc_mqueue.c.

References cache_enter(), mqfs_node::mn_parent, mqfs_allocv(), MQFS_NAMELEN, mqfs_search(), td, and VTON.

Referenced by mqfs_lookup().

Here is the call graph for this function:

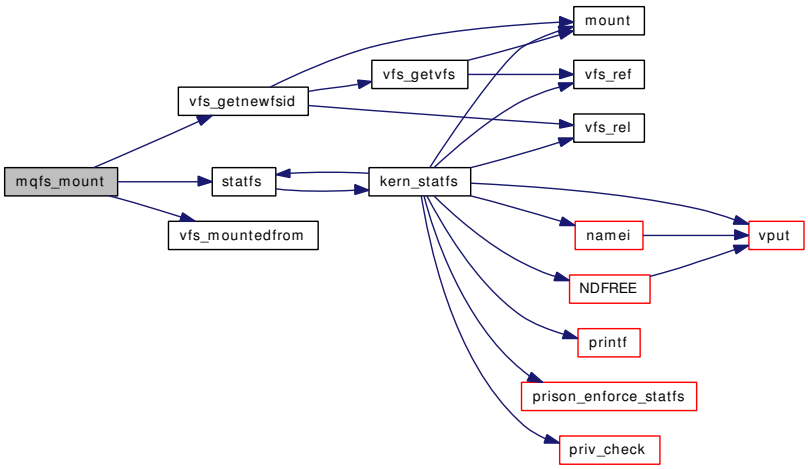


9.144.4.48 static int mqfs_mount (struct mount * *mp*, struct thread * *td*) [static]

Definition at line 557 of file uipc_mqueue.c.

References mqfs_data, statfs(), vfs_getnewfsid(), and vfs_mountedfrom().

Here is the call graph for this function:



9.144.4.49 static int mqfs_open (struct vop_open_args * *ap*) [static]

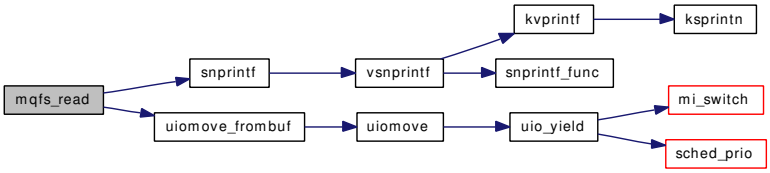
Definition at line 1066 of file uipc_mqueue.c.

9.144.4.50 static int mqfs_read (struct vop_read_args * *ap*) [static]

Definition at line 1274 of file uipc_mqueue.c.

References buf, mqueue::mq_curmsgs, mqueue::mq_maxmsg, mqueue::mq_msgsize, mqueue::mq_-totalbytes, snprintf(), uiomove_frombuf(), VTOMQ, and VTON.

Here is the call graph for this function:

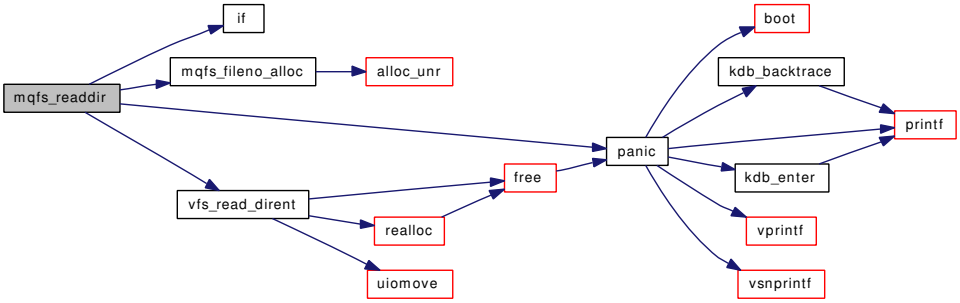


9.144.4.51 static int mqfs_readdir (struct vop_readdir_args * *ap*) [static]

Definition at line 1316 of file uipc_mqueue.c.

References if(), mqfs_info::mi_lock, mqfs_node::mn_name, mqfs_fileno_alloc(), MQFS_NAMELEN, mqfstype_dir, mqfstype_file, mqfstype_parent, mqfstype_root, mqfstype_symlink, mqfstype_this, panic(), vfs_read_dirent(), VFSTOMQFS, and VTON.

Here is the call graph for this function:

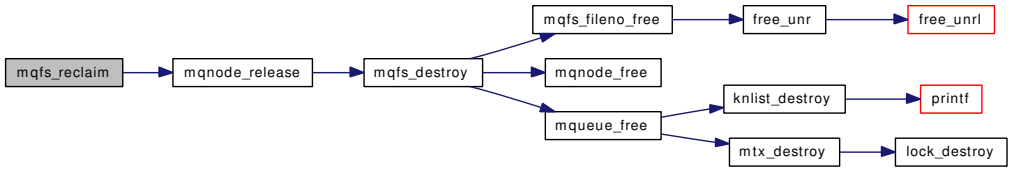


9.144.4.52 static int mqfs_reclaim (struct vop_reclaim_args * *ap*) [static]

Definition at line 1036 of file uipc_mqueue.c.

References mqfs_info::mi_lock, mqnode_release(), mvdata_zone, and VFSTOMQFS.

Here is the call graph for this function:

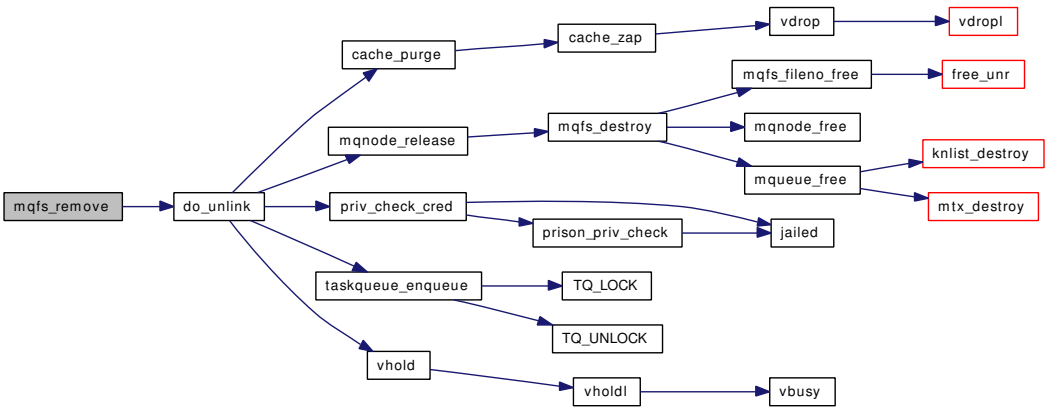


9.144.4.53 static int mqfs_remove (struct vop_remove_args * *ap*) [static]

Definition at line 995 of file uipc_mqueue.c.

References do_unlink(), mqfs_info::mi_lock, VFSTOMQFS, and VTON.

Here is the call graph for this function:

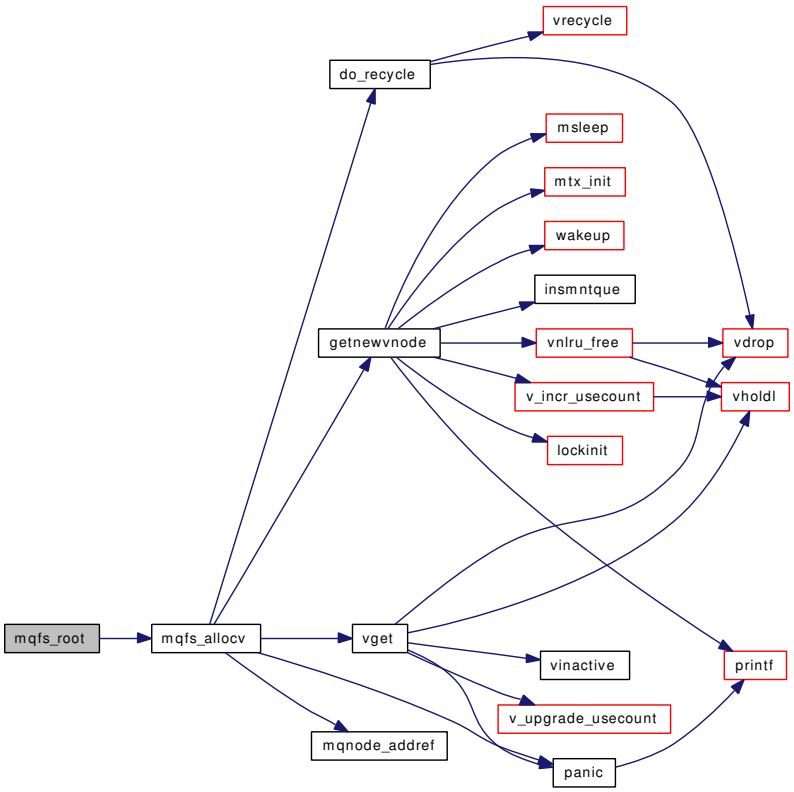


9.144.4.54 static int mqfs_root (struct mount * *mp*, int *flags*, struct vnode ** *vpp*, struct thread * *td*) [static]

Definition at line 599 of file uipc_mqueue.c.

References mqfs_info::mi_lock, mqfs_info::mi_root, mqfs_allocv(), ret, and VFSTOMQFS.

Here is the call graph for this function:



9.144.4.55 static struct mqfs_node* mqfs_search (struct mqfs_node * *pd*, const char * *name*, int *len*) [static]

Definition at line 755 of file uipc_mqueue.c.

References mqfs_node::mn_name.

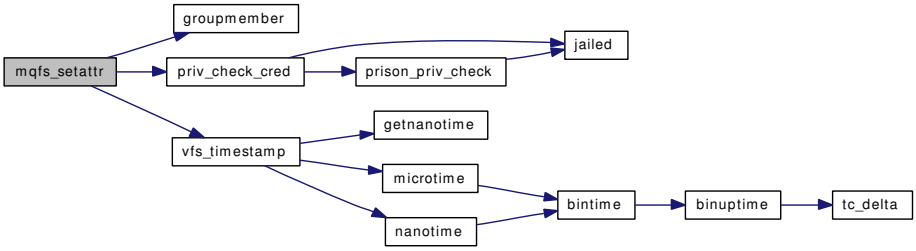
Referenced by kmq_open(), kmq_unlink(), mqfs_create(), and mqfs_lookupx().

9.144.4.56 static int mqfs_setattr (struct vop_setattr_args * *ap*) [static]

Definition at line 1173 of file uipc_mqueue.c.

References groupmember(), priv_check_cred(), vfs_timestamp(), and VTON.

Here is the call graph for this function:



9.144.4.57 static int mqfs_statfs (struct mount * *mp*, struct statfs * *sbp*, struct thread * *td*) [static]

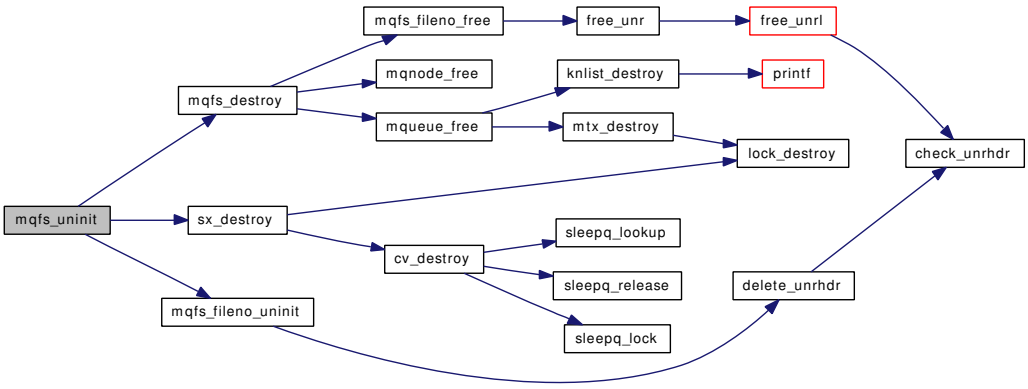
Definition at line 615 of file uipc_mqueue.c.

9.144.4.58 static int mqfs_uninit (struct vfsconf * *vfc*) [static]

Definition at line 662 of file uipc_mqueue.c.

References exit_tag, mqfs_info::mi_lock, mqfs_info::mi_root, mqfs_data, mqfs_destroy(), mqfs_fileno_-uninit(), mqnode_zone, mqnoti_zone, mqueue_zone, mvdata_zone, sx_destroy(), and unloadable.

Here is the call graph for this function:

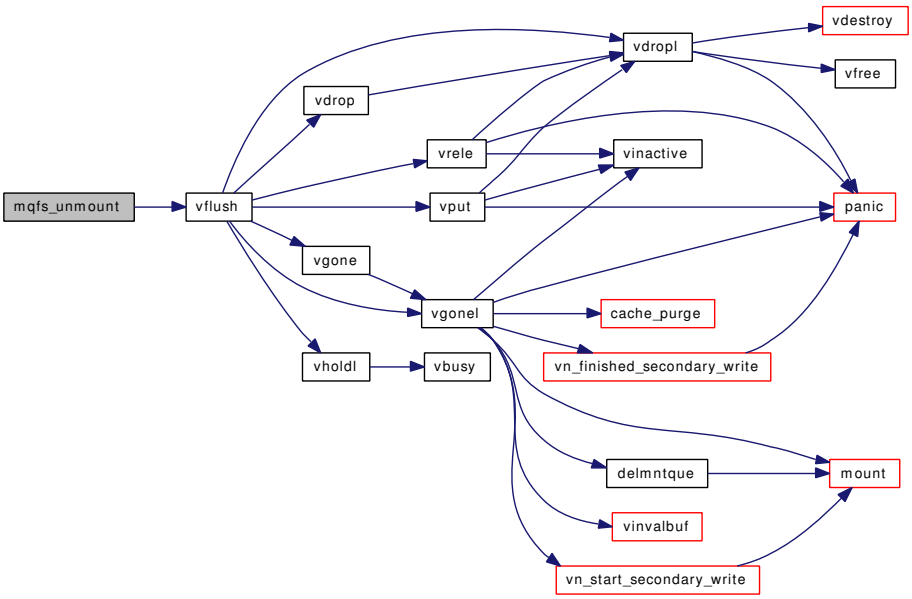


9.144.4.59 static int mqfs_unmount (struct mount * *mp*, int *mntflags*, struct thread * *td*) [static]

Definition at line 587 of file uipc_mqueue.c.

References vflush().

Here is the call graph for this function:



9.144.4.60 static __inline void mqnode_addref (struct mqfs_node * *node*) [static]

Definition at line 369 of file uipc_mqueue.c.

Referenced by kmq_open(), mqfs_add_node(), and mqfs_allocv().

9.144.4.61 static __inline struct mqfs_node* mqnode_alloc (void) [static]

Definition at line 357 of file uipc_mqueue.c.

References mqnode_zone.

Referenced by mqfs_create_node(), and mqfs_fixup_dir().

9.144.4.62 static __inline void mqnode_free (struct mqfs_node * *node*) [static]

Definition at line 363 of file uipc_mqueue.c.

References mqnode_zone.

Referenced by mqfs_create_file(), mqfs_destroy(), and mqfs_fixup_dir().

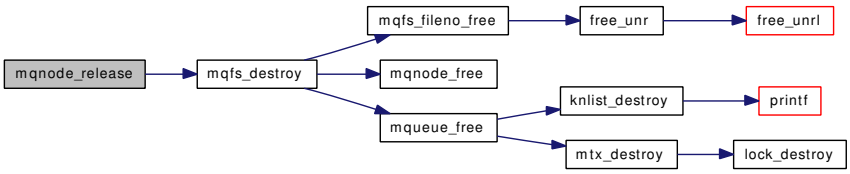
9.144.4.63 static __inline void mqnode_release (struct mqfs_node * *node*) [static]

Definition at line 375 of file uipc_mqueue.c.

References mqfs_destroy(), mqfstype_dir, and mqfstype_root.

Referenced by do_unlink(), mqf_close(), and mqfs_reclaim().

Here is the call graph for this function:



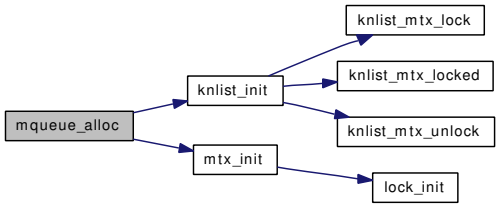
9.144.4.64 static struct mqueue * mqueue_alloc (const struct mq_attr * *attr*) [static]

Definition at line 1493 of file uipc_mqueue.c.

References curmq, default_maxmsg, default_msgsize, knlist_init(), maxmq, mqueue::mq_maxmsg, mqueue::mq_msgq, mqueue::mq_msgsize, mqueue::mq_mutex, mqueue::mq_rsel, mqueue::mq_wsel, mqueue_zone, and mtx_init().

Referenced by kmq_open(), and mqfs_create().

Here is the call graph for this function:



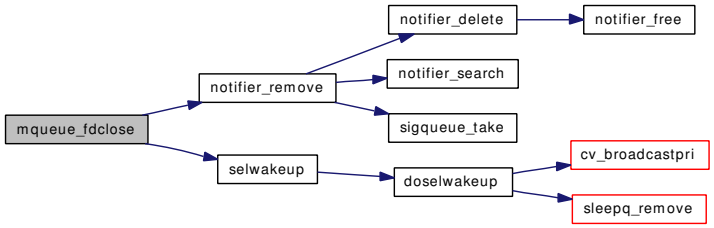
9.144.4.65 static void mqueue_fdclose (struct thread * *td*, int *fd*, struct file * *fp*) [static]

Definition at line 2271 of file uipc_mqueue.c.

References FPTOMQ, mqueue::mq_flags, mqueue::mq_mutex, mqueue::mq_rsel, MQ_RSEL, mqueue::mq_wsel, MQ_WSEL, notifier_remove(), and selwakeup().

Referenced by mqfs_init().

Here is the call graph for this function:



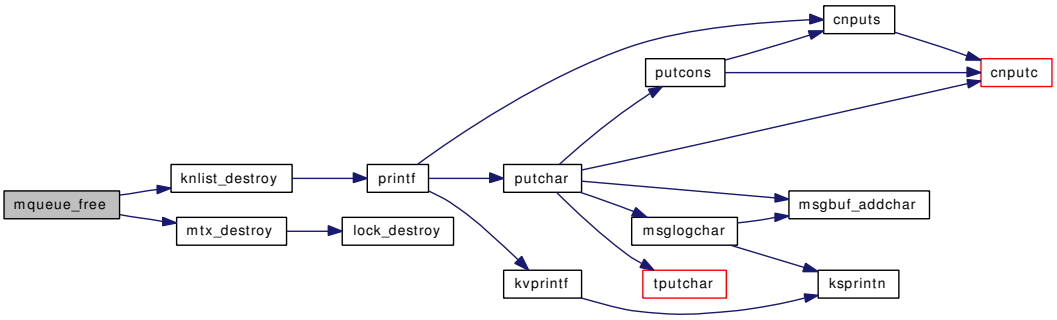
9.144.4.66 static void mqueue_free (struct mqueue * *mq*) [static]

Definition at line 1519 of file uipc_mqueue.c.

References curmq, knlist_destroy(), mqueue::mq_msgq, mqueue::mq_mutex, mqueue::mq_rsel, mqueue::mq_wsel, mqueue_zone, and mtx_destroy().

Referenced by kmq_open(), mqfs_create(), and mqfs_destroy().

Here is the call graph for this function:



9.144.4.67 static __inline void mqueue_freemsg (struct mqueue_msg * *msg*) [static]

Definition at line 1578 of file uipc_mqueue.c.

Referenced by mqueue_receive(), and mqueue_send().

9.144.4.68 static struct mqueue_msg* mqueue_loadmsg (const char * *msg_ptr*, size_t *msg_size*, int *msg_prio*) [static]

Definition at line 1539 of file uipc_mqueue.c.

Referenced by mqueue_send().

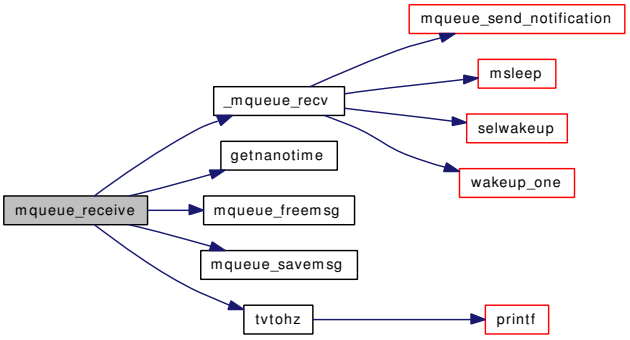
9.144.4.69 int mqueue_receive (struct mqueue * *mq*, char * *msg_ptr*, size_t *msg_len*, unsigned * *msg_prio*, int *waitok*, const struct timespec * *abs_timeout*) [static]

Definition at line 1740 of file uipc_mqueue.c.

References _mqueue_recv(), getnanotime(), mqueue_freemsg(), mqueue_savemsg(), and tvtohz().

Referenced by kmq_timedreceive().

Here is the call graph for this function:



9.144.4.70 static int mqueue_savemsg (struct mqueue_msg * *msg*, char * *msg_ptr*, int * *msg_prio*) [static]

Definition at line 1563 of file uipc_mqueue.c.

Referenced by mqueue_receive().

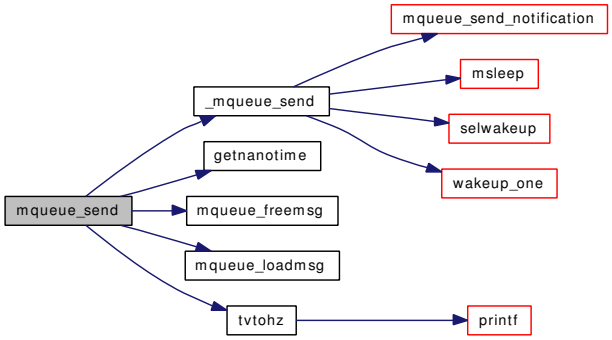
9.144.4.71 int mqueue_send (struct mqueue * *mq*, const char * *msg_ptr*, size_t *msg_len*, unsigned *msg_prio*, int *waitok*, const struct timespec * *abs_timeout*) [static]

Definition at line 1589 of file uipc_mqueue.c.

References _mqueue_send(), getnanotime(), mqueue::mq_msgsize, mqueue_freemsg(), mqueue_loadmsg(), and tvtohz().

Referenced by kmq_timedsend().

Here is the call graph for this function:



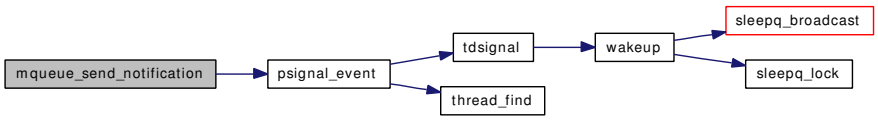
9.144.4.72 static void mqueue_send_notification (struct mqueue * *mq*) [static]

Definition at line 1717 of file uipc_mqueue.c.

References mqueue::mq_mutex, mqueue::mq_notifier, and psignal_event().

Referenced by _mqueue_recv(), _mqueue_send(), and kmq_notify().

Here is the call graph for this function:



9.144.4.73 static __inline struct mqueue_notifier* notifier_alloc (void) [static]

Definition at line 1853 of file uipc_mqueue.c.

References mqnoti_zone.

Referenced by kmq_notify().

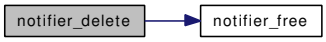
9.144.4.74 static __inline void notifier_delete (struct proc * *p*, struct mqueue_notifier * *nt*) [static]

Definition at line 1883 of file uipc_mqueue.c.

References notifier_free().

Referenced by notifier_remove().

Here is the call graph for this function:



9.144.4.75 static __inline void notifier_free (struct mqueue_notifier * *p*) [static]

Definition at line 1859 of file uipc_mqueue.c.

References mqnoti_zone.

Referenced by kmq_notify(), and notifier_delete().

9.144.4.76 static __inline void notifier_insert (struct proc * *p*, struct mqueue_notifier * *nt*) [static]

Definition at line 1877 of file uipc_mqueue.c.

Referenced by kmq_notify().

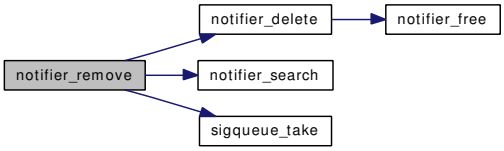
9.144.4.77 static void notifier_remove (struct proc * *p*, struct mqueue * *mq*, int *fd*) [static]

Definition at line 1890 of file uipc_mqueue.c.

References mqueue::mq_mutex, mqueue::mq_notifier, notifier_delete(), notifier_search(), and sigqueue_-take().

Referenced by kmq_notify(), mq_proc_exit(), and mqueue_fdclose().

Here is the call graph for this function:



9.144.4.78 static struct mqueue_notifier* notifier_search (struct proc * *p*, int *fd*) [static]

Definition at line 1865 of file uipc_mqueue.c.

Referenced by kmq_notify(), and notifier_remove().

9.144.4.79 SYSCALL_MODULE_HELPER (kmq_unlink)

9.144.4.80 SYSCALL_MODULE_HELPER (kmq_notify)

9.144.4.81 SYSCALL_MODULE_HELPER (kmq_timedreceive)

9.144.4.82 SYSCALL_MODULE_HELPER (kmq_timedsend)

9.144.4.83 SYSCALL_MODULE_HELPER (kmq_setattr)

9.144.4.84 SYSCALL_MODULE_HELPER (kmq_open)

9.144.4.85 SYSCTL_INT (_kern_mqueue, OID_AUTO, curmq, CTLFLAG_RW, & *curmq*, 0, "current message queue number")

9.144.4.86 SYSCTL_INT (_kern_mqueue, OID_AUTO, maxmq, CTLFLAG_RW, & *maxmq*, 0, "maximum message queues")

9.144.4.87 SYSCTL_INT (_kern_mqueue, OID_AUTO, maxmsgsize, CTLFLAG_RW, & *maxmsgsize*, 0, "Default maximum message size")

9.144.4.88 SYSCTL_INT (_kern_mqueue, OID_AUTO, maxmsg, CTLFLAG_RW, & *maxmsg*, 0, "Default maximum messages in queue")

9.144.4.89 SYSCTL_NODE (_kern, OID_AUTO, mqueue, CTLFLAG_RW, 0, "POSIX real time message queue")

9.144.4.90 TAILQ_HEAD (msgq, mqueue_msg)

9.144.4.91 VFS_SET (mqfs_vfsops, mqueuefs, VFCF_SYNTHETIC)

9.144.5 Variable Documentation

9.144.5.1 int curmq = 0 [static]

Definition at line 200 of file uipc_mqueue.c.

Referenced by mqueue_alloc(), and mqueue_free().

9.144.5.2 int default_maxmsg = 10 [static]

Definition at line 188 of file uipc_mqueue.c.

Referenced by mqueue_alloc().

9.144.5.3 int default_msgsize = 1024 [static]

Definition at line 189 of file uipc_mqueue.c.

Referenced by mqueue_alloc().

9.144.5.4 eventhandler_tag exit_tag [static]

Definition at line 206 of file uipc_mqueue.c.

Referenced by aio_onceonly(), aio_unload(), mqfs_init(), and mqfs_uninit().

9.144.5.5 int maxmq = 100 [static]

Definition at line 197 of file uipc_mqueue.c.

Referenced by mqueue_alloc().

9.144.5.6 int maxmsg = 100 [static]

Definition at line 191 of file uipc_mqueue.c.

Referenced by kmq_open().

9.144.5.7 int maxmsgsize = 16384 [static]

Definition at line 194 of file uipc_mqueue.c.

Referenced by kmq_open().

9.144.5.8 struct filterops mq_rfiltops

Initial value:

```
{ 1, NULL, filt_mqdetach, filt_mqread }
```

Definition at line 260 of file uipc_mqueue.c.

9.144.5.9 struct filterops mq_wfiltops

Initial value:

```
{ 1, NULL, filt_mqdetach, filt_mqwrite }
```

Definition at line 262 of file uipc_mqueue.c.

9.144.5.10 struct mqfs_info mqfs_data [static]

Definition at line 209 of file uipc_mqueue.c.

Referenced by kmq_open(), kmq_unlink(), mqf_close(), mqfs_init(), mqfs_mount(), and mqfs_uninit().

9.144.5.11 struct vfsops mqfs_vfsops [static]

Initial value:

```
{
        .vfs_init               = mqfs_init,
        .vfs_uninit             = mqfs_uninit,
        .vfs_mount              = mqfs_mount,
        .vfs_unmount            = mqfs_unmount,
        .vfs_root               = mqfs_root,
        .vfs_statfs             = mqfs_statfs,
}
```

Definition at line 2479 of file uipc_mqueue.c.

9.144.5.12 struct vop_vector mqfs_vnodeops [static]

Initial value:

```
{
        .vop_default            = &default_vnodeops,
        .vop_access             = mqfs_access,
        .vop_cachedlookup       = mqfs_lookup,
        .vop_lookup             = vfs_cache_lookup,
        .vop_reclaim            = mqfs_reclaim,
        .vop_create             = mqfs_create,
        .vop_remove             = mqfs_remove,
        .vop_inactive           = mqfs_inactive,
        .vop_open               = mqfs_open,
        .vop_close              = mqfs_close,
        .vop_getattr            = mqfs_getattr,
        .vop_setattr            = mqfs_setattr,
        .vop_read               = mqfs_read,
        .vop_write              = VOP_EOPNOTSUPP,
        .vop_readdir            = mqfs_readdir,
        .vop_mkdir              = VOP_EOPNOTSUPP,
        .vop_rmdir              = VOP_EOPNOTSUPP
}
```

Definition at line 2459 of file uipc_mqueue.c.

9.144.5.13 struct vop_vector mqfs_vnodeops [static]

Definition at line 214 of file uipc_mqueue.c.

9.144.5.14 uma_zone_t mqnode_zone [static]

Definition at line 210 of file uipc_mqueue.c.

Referenced by mqfs_init(), mqfs_uninit(), mqnode_alloc(), and mqnode_free().

9.144.5.15 uma_zone_t mqnoti_zone [static]

Definition at line 213 of file uipc_mqueue.c.

Referenced by mqfs_init(), mqfs_uninit(), notifier_alloc(), and notifier_free().

9.144.5.16 uma_zone_t mqueue_zone [static]

Definition at line 211 of file uipc_mqueue.c.

Referenced by mqfs_init(), mqfs_uninit(), mqueue_alloc(), and mqueue_free().

9.144.5.17 struct fileops mqueueops [static]

Initial value:

```
{
        .fo_read                = mqf_read,
        .fo_write               = mqf_write,
        .fo_ioctl               = mqf_ioctl,
        .fo_poll                = mqf_poll,
        .fo_kqfilter            = mqf_kqfilter,
        .fo_stat                = mqf_stat,
        .fo_close               = mqf_close
}
```

Definition at line 2449 of file uipc_mqueue.c.

9.144.5.18 struct fileops mqueueops [static]

Definition at line 215 of file uipc_mqueue.c.

9.144.5.19 uma_zone_t mvdata_zone [static]

Definition at line 212 of file uipc_mqueue.c.

Referenced by mqfs_allocv(), mqfs_init(), mqfs_reclaim(), and mqfs_uninit().

9.144.5.20 int unloadable = 0 [static]

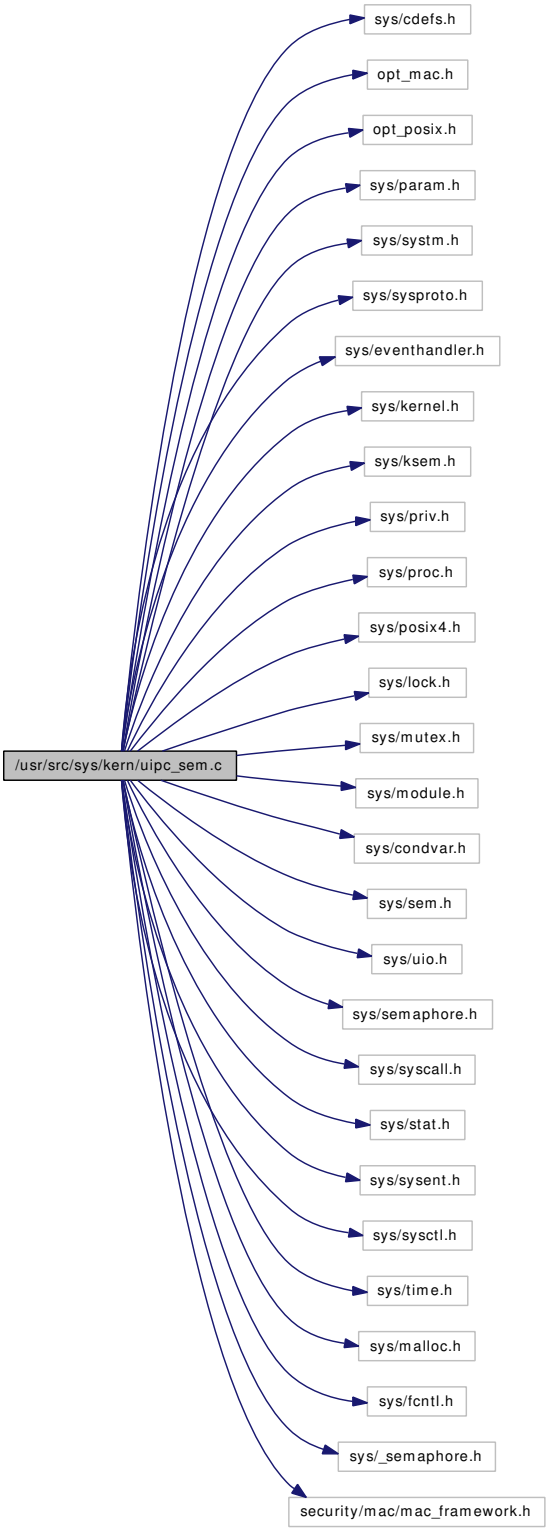
Definition at line 203 of file uipc_mqueue.c.

Referenced by accept_filt_generic_mod_event(), aio_unload(), and mqfs_uninit().

9.145 /usr/src/sys/kern/uipc_sem.c File Reference

```
#include <sys/cdefs.h>
#include "opt_mac.h"
#include "opt_posix.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/sysproto.h>
#include <sys/eventhandler.h>
#include <sys/kernel.h>
#include <sys/ksem.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/posix4.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/module.h>
#include <sys/condvar.h>
#include <sys/sem.h>
#include <sys/uio.h>
#include <sys/semaphore.h>
#include <sys/syscall.h>
#include <sys/stat.h>
#include <sys/sysent.h>
#include <sys/sysctl.h>
#include <sys/time.h>
#include <sys/malloc.h>
#include <sys/fcntl.h>
#include <sys/_semaphore.h>
#include <security/mac/mac_framework.h>
```

Include dependency graph for uipc_sem.c:



Data Structures

- struct ksem_init_args

- struct ksem_open_args
- struct ksem_unlink_args
- struct ksem_close_args
- struct ksem_post_args
- struct ksem_wait_args
- struct ksem_timedwait_args
- struct ksem_trywait_args
- struct ksem_getvalue_args
- struct ksem_destroy_args

Defines

- #define SEM_MAX 30
- #define SEM_MAX_NAMELEN 14
- #define SEM_TO_ID(x) ((intptr_t)(x))
- #define ID_TO_SEM(x) id_to_sem(x)
- #define DP(x)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/uipc_sem.c,v 1.27 2006/11/11 16:26:56 trhodes Exp $")
- static int sem_count_proc (struct proc *p)
- static struct ksem * sem_lookup_byname (const char *name)
- static int sem_create (struct thread *td, const char *name, struct ksem **ksret, mode_t mode, unsigned int value)
- static void sem_free (struct ksem *ksnew)
- static int sem_perm (struct thread *td, struct ksem *ks)
- static void sem_enter (struct proc *p, struct ksem *ks)
- static int sem_leave (struct proc *p, struct ksem *ks)
- static void sem_exechook (void *arg, struct proc *p, struct image_params *imgp)
- static void sem_exithook (void *arg, struct proc *p)
- static void sem_forkhook (void *arg, struct proc *p1, struct proc *p2, int flags)
- static int sem_hasopen (struct thread *td, struct ksem *ks)
- static int kern_sem_close (struct thread *td, semid_t id)
- static int kern_sem_post (struct thread *td, semid_t id)
- static int kern_sem_wait (struct thread *td, semid_t id, int tryflag, struct timespec *abstime)
- static int kern_sem_init (struct thread *td, int dir, unsigned int value, semid_t *idp)
- static int kern_sem_open (struct thread *td, int dir, const char *name, int oflag, mode_t mode, unsigned int value, semid_t *idp)
- static int kern_sem_unlink (struct thread *td, const char *name)
- LIST_HEAD (ksem)
- static __inline void sem_rel (struct ksem *ks)
- static __inline struct ksem * id_to_sem (semid_t id)
- int ksem_init (struct thread *td, struct ksem_init_args *uap)
- int ksem_open (struct thread *td, struct ksem_open_args *uap)
- static __inline struct kuser * sem_getuser (struct proc *p, struct ksem *ks)
- int ksem_unlink (struct thread *td, struct ksem_unlink_args *uap)
- int ksem_close (struct thread *td, struct ksem_close_args *uap)
- int ksem_post (struct thread *td, struct ksem_post_args *uap)

- int ksem_wait (struct thread *td, struct ksem_wait_args *uap)
- int ksem_timedwait (struct thread *td, struct ksem_timedwait_args *uap)
- int ksem_trywait (struct thread *td, struct ksem_trywait_args *uap)
- int ksem_getvalue (struct thread *td, struct ksem_getvalue_args *uap)
- int ksem_destroy (struct thread *td, struct ksem_destroy_args *uap)
- static void sem_exechook (void *arg, struct proc *p, struct image_params *imgp __unused)
- static int sem_modload (struct module *module, int cmd, void *arg)
- SYSCALL_MODULE_HELPER (ksem_init)
- SYSCALL_MODULE_HELPER (ksem_open)
- SYSCALL_MODULE_HELPER (ksem_unlink)
- SYSCALL_MODULE_HELPER (ksem_close)
- SYSCALL_MODULE_HELPER (ksem_post)
- SYSCALL_MODULE_HELPER (ksem_wait)
- SYSCALL_MODULE_HELPER (ksem_timedwait)
- SYSCALL_MODULE_HELPER (ksem_trywait)
- SYSCALL_MODULE_HELPER (ksem_getvalue)
- SYSCALL_MODULE_HELPER (ksem_destroy)
- DECLARE_MODULE (sem, sem_mod, SI_SUB_SYSV_SEM, SI_ORDER_FIRST)
- MODULE_VERSION (sem, 1)

Variables

- static moduledata_t sem_mod

9.145.1 Define Documentation

9.145.1.1 #define DP(x)

Referenced by id_to_sem(), kern_sem_open(), kern_sem_unlink(), kern_sem_wait(), ksem_open(), LIST_HEAD(), sem_create(), sem_leave(), sem_perm(), and sem_rel().

9.145.1.2 #define ID_TO_SEM(x) id_to_sem(x)

Definition at line 98 of file uipc_sem.c.

Referenced by kern_sem_close(), kern_sem_post(), kern_sem_wait(), ksem_destroy(), and ksem_-getvalue().

9.145.1.3 #define SEM_MAX 30

Definition at line 92 of file uipc_sem.c.

Referenced by sem_modload().

9.145.1.4 #define SEM_MAX_NAMELEN 14

Definition at line 95 of file uipc_sem.c.

Referenced by ksem_open(), ksem_unlink(), and sem_create().

9.145.1.5 #define SEM_TO_ID(x) ((intptr_t)(x))

Definition at line 97 of file uipc_sem.c.

Referenced by kern_sem_init(), and kern_sem_open().

9.145.2 Function Documentation

9.145.2.1 __FBSDID ("$FreeBSD: src/sys/kern/uipc_sem. *c*, v 1.27 2006/11/11 16:26:56 trhodes Exp $")

9.145.2.2 DECLARE_MODULE (sem, sem_mod, SI_SUB_SYSV_SEM, SI_ORDER_FIRST)

9.145.2.3 static __inline struct ksem * id_to_sem (semid_t *id*) [static]

Definition at line 150 of file uipc_sem.c.

References DP.

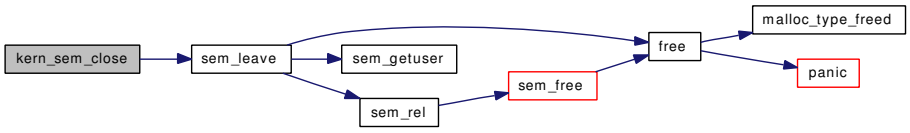
9.145.2.4 static int kern_sem_close (struct thread * *td*, semid_t *id*) [static]

Definition at line 578 of file uipc_sem.c.

References ID_TO_SEM, and sem_leave().

Referenced by ksem_close().

Here is the call graph for this function:



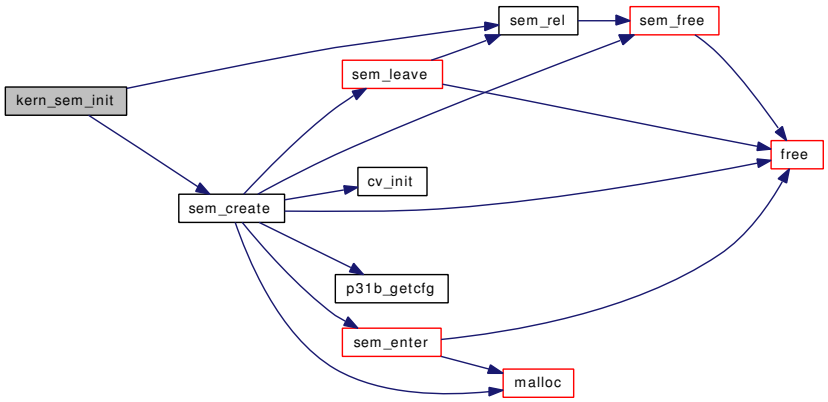
9.145.2.5 static int kern_sem_init (struct thread * *td*, int *dir*, unsigned int *value*, semid_t * *idp*) [static]

Definition at line 254 of file uipc_sem.c.

References sem_create(), sem_rel(), and SEM_TO_ID.

Referenced by ksem_init().

Here is the call graph for this function:



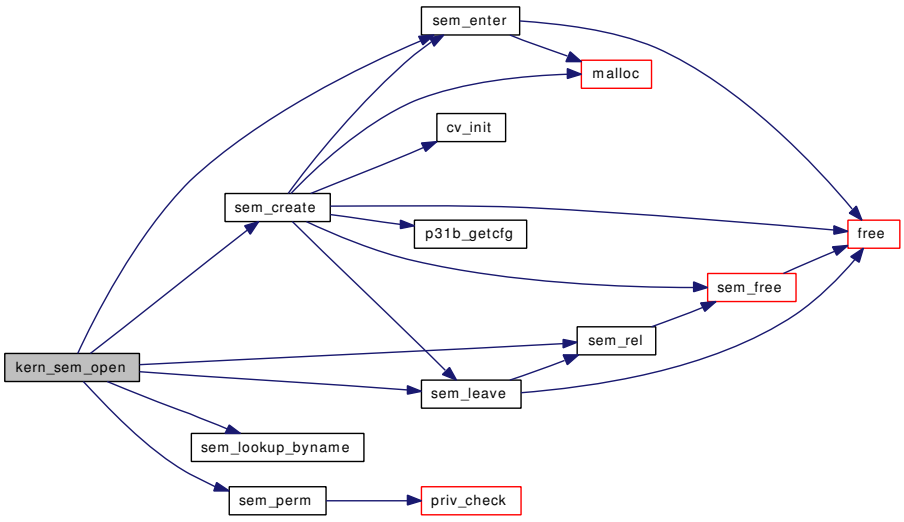
9.145.2.6 static int kern_sem_open (struct thread * *td*, int *dir*, const char * *name*, int *oflag*, mode_t *mode*, unsigned int *value*, semid_t * *idp*) [static]

Definition at line 310 of file uipc_sem.c.

References DP, sem_create(), sem_enter(), sem_leave(), sem_lookup_byname(), sem_perm(), sem_rel(), and SEM_TO_ID.

Referenced by ksem_open().

Here is the call graph for this function:



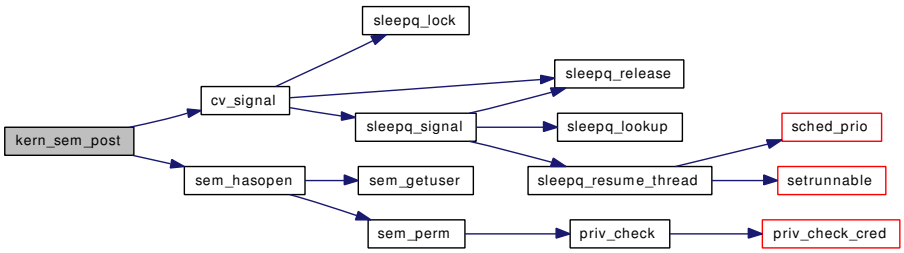
9.145.2.7 static int kern_sem_post (struct thread * *td*, semid_t *id*) [static]

Definition at line 607 of file uipc_sem.c.

References cv_signal(), ID_TO_SEM, and sem_hasopen().

Referenced by ksem_post().

Here is the call graph for this function:



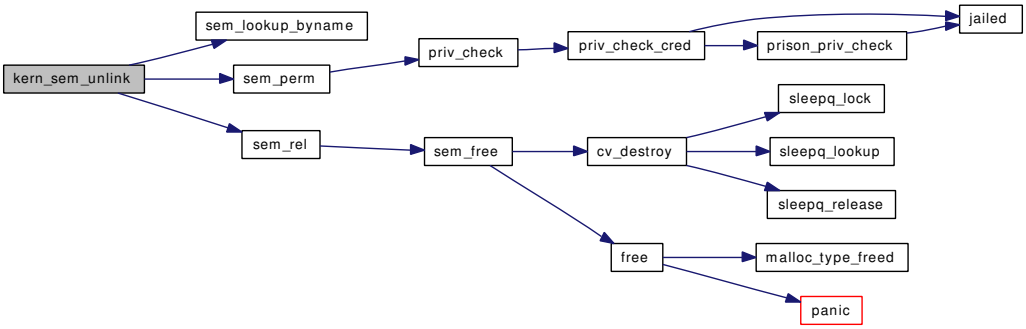
9.145.2.8 static int kern_sem_unlink (struct thread * *td*, const char * *name*) [static]

Definition at line 535 of file uipc_sem.c.

References DP, sem_lookup_byname(), sem_perm(), and sem_rel().

Referenced by ksem_unlink().

Here is the call graph for this function:



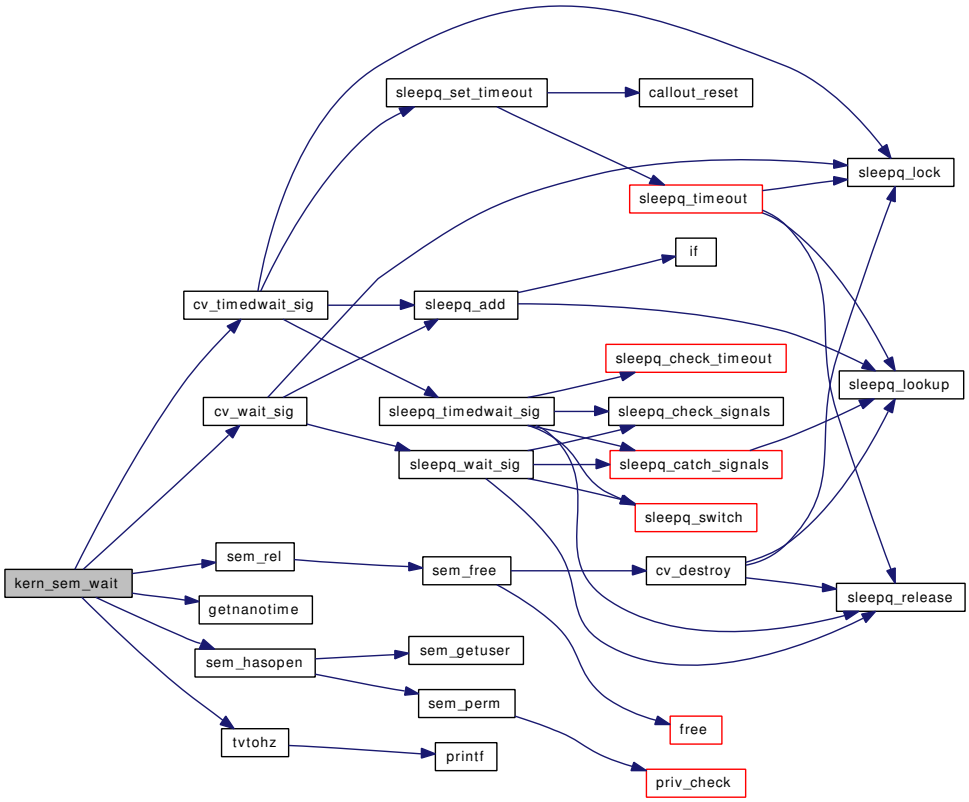
9.145.2.9 static int kern_sem_wait (struct thread * *td*, semid_t *id*, int *tryflag*, struct timespec * *abstime*) [static]

Definition at line 692 of file uipc_sem.c.

References cv_timedwait_sig(), cv_wait_sig(), DP, getnanotime(), ID_TO_SEM, sem_hasopen(), sem_-rel(), and tvtohz().

Referenced by ksem_timedwait(), ksem_trywait(), and ksem_wait().

Here is the call graph for this function:

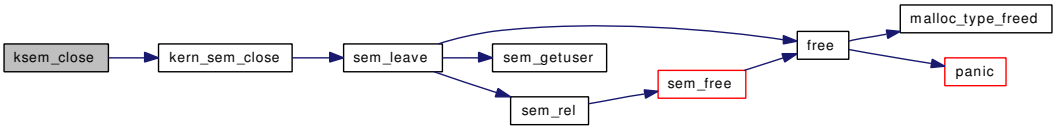


9.145.2.10 int ksem_close (struct thread * *td*, struct ksem_close_args * *uap*)

Definition at line 571 of file uipc_sem.c.

References ksem_close_args::id, and kern_sem_close().

Here is the call graph for this function:

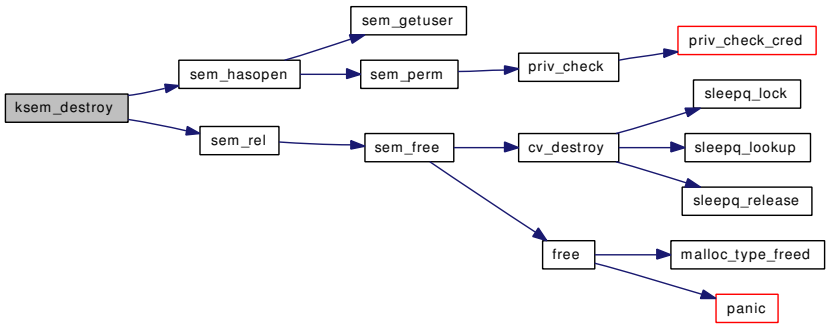


9.145.2.11 int ksem_destroy (struct thread * *td*, struct ksem_destroy_args * *uap*)

Definition at line 797 of file uipc_sem.c.

References ksem_destroy_args::id, ID_TO_SEM, sem_hasopen(), and sem_rel().

Here is the call graph for this function:

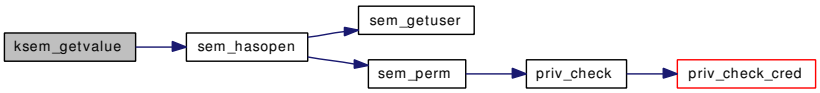


9.145.2.12 int ksem_getvalue (struct thread * *td*, struct ksem_getvalue_args * *uap*)

Definition at line 766 of file uipc_sem.c.

References ksem_getvalue_args::id, ID_TO_SEM, sem_hasopen(), and ksem_getvalue_args::val.

Here is the call graph for this function:

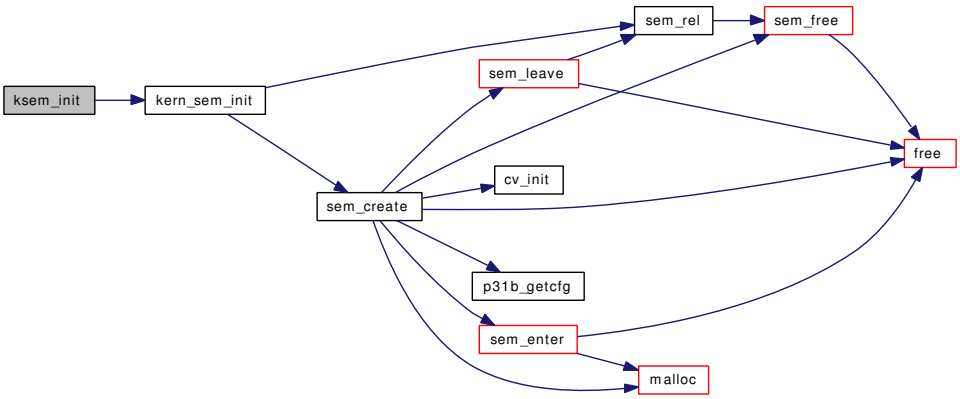


9.145.2.13 int ksem_init (struct thread * *td*, struct ksem_init_args * *uap*)

Definition at line 245 of file uipc_sem.c.

References ksem_init_args::idp, kern_sem_init(), and ksem_init_args::value.

Here is the call graph for this function:

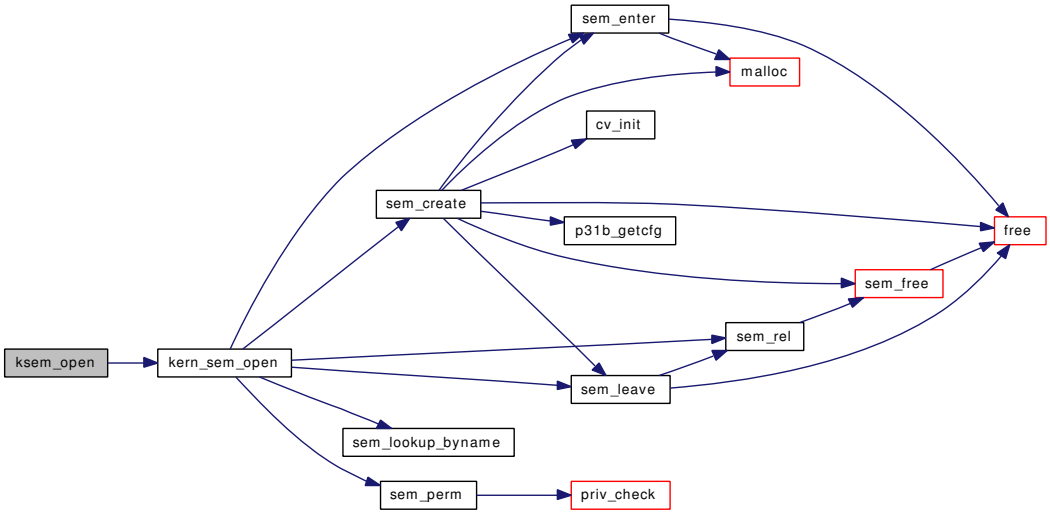


9.145.2.14 int ksem_open (struct thread * *td*, struct ksem_open_args * *uap*)

Definition at line 293 of file uipc_sem.c.

References DP, ksem_open_args::idp, kern_sem_open(), ksem_open_args::mode, ksem_open_args::name, ksem_open_args::oflag, SEM_MAX_NAMELEN, and ksem_open_args::value.

Here is the call graph for this function:

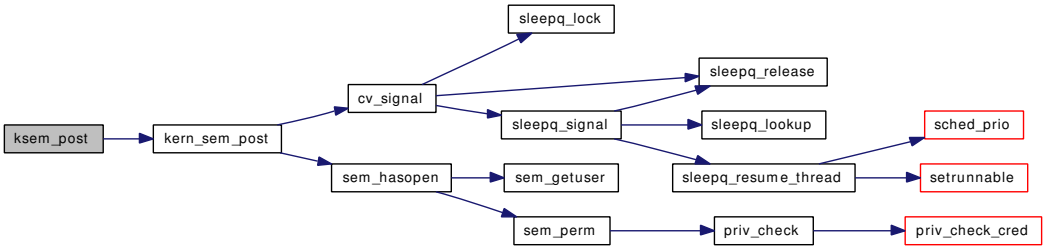


9.145.2.15 int ksem_post (struct thread * *td*, struct ksem_post_args * *uap*)

Definition at line 600 of file uipc_sem.c.

References ksem_post_args::id, and kern_sem_post().

Here is the call graph for this function:

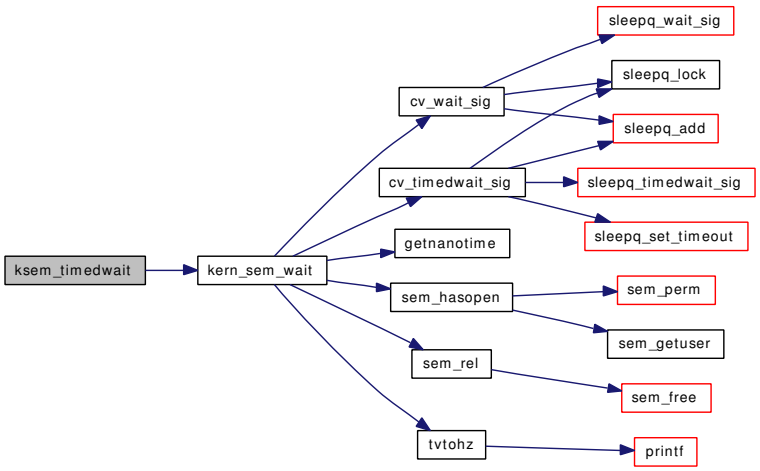


9.145.2.16 int ksem_timedwait (struct thread * *td*, struct ksem_timedwait_args * *uap*)

Definition at line 658 of file uipc_sem.c.

References ksem_timedwait_args::abstime, ksem_timedwait_args::id, and kern_sem_wait().

Here is the call graph for this function:

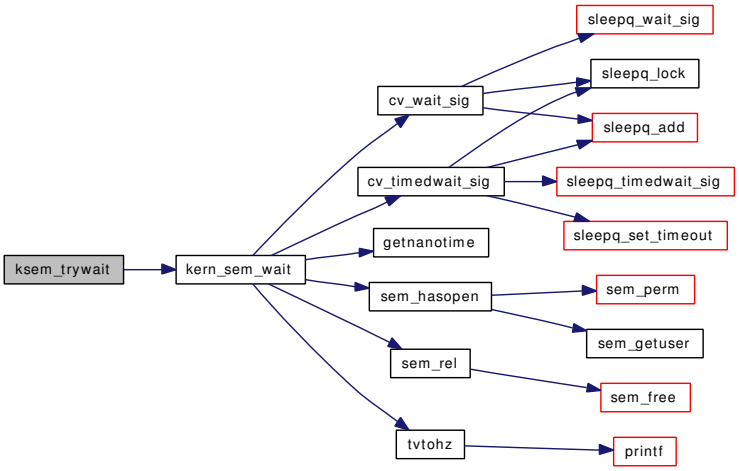


9.145.2.17 int ksem_trywait (struct thread * *td*, struct ksem_trywait_args * *uap*)

Definition at line 685 of file uipc_sem.c.

References ksem_trywait_args::id, and kern_sem_wait().

Here is the call graph for this function:

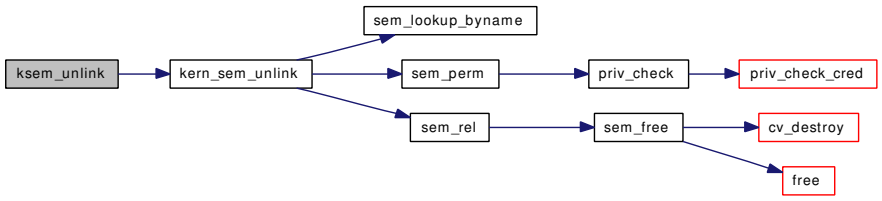


9.145.2.18 int ksem_unlink (struct thread * *td*, struct ksem_unlink_args * *uap*)

Definition at line 523 of file uipc_sem.c.

References kern_sem_unlink(), ksem_unlink_args::name, and SEM_MAX_NAMELEN.

Here is the call graph for this function:

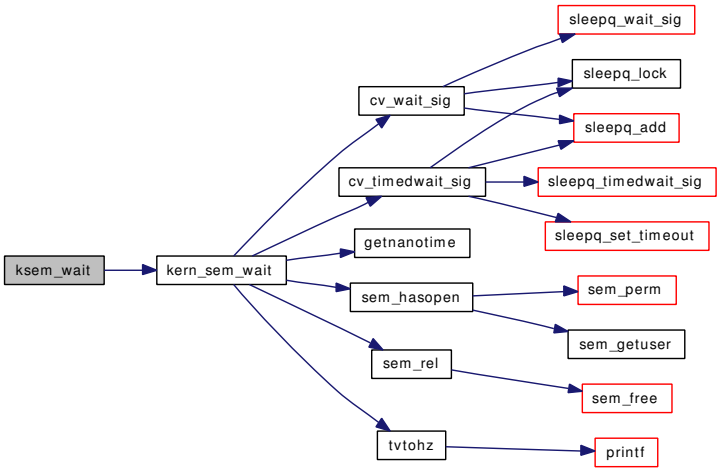


9.145.2.19 int ksem_wait (struct thread * *td*, struct ksem_wait_args * *uap*)

Definition at line 644 of file uipc_sem.c.

References ksem_wait_args::id, and kern_sem_wait().

Here is the call graph for this function:



9.145.2.20 LIST_HEAD (ksem)

Definition at line 104 of file uipc_sem.c.

References DP.

9.145.2.21 MODULE_VERSION (sem, 1)

9.145.2.22 static int sem_count_proc (struct proc * *p*) [static]

Definition at line 830 of file uipc_sem.c.

Referenced by sem_forkhook().

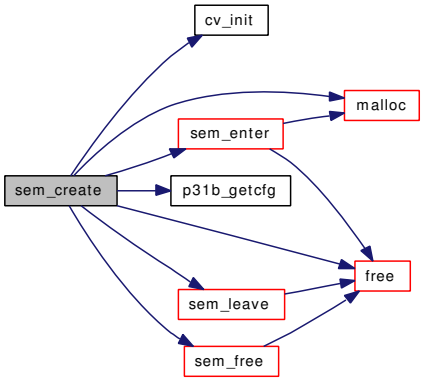
9.145.2.23 static int sem_create (struct thread * *td*, const char * *name*, struct ksem ** *ksret*, mode_t *mode*, unsigned int *value*) [static]

Definition at line 177 of file uipc_sem.c.

References cv_init(), DP, free(), malloc(), p31b_getcfg(), ret, sem_enter(), sem_free(), sem_leave(), SEM_MAX_NAMELEN, and uc.

Referenced by kern_sem_init(), and kern_sem_open().

Here is the call graph for this function:



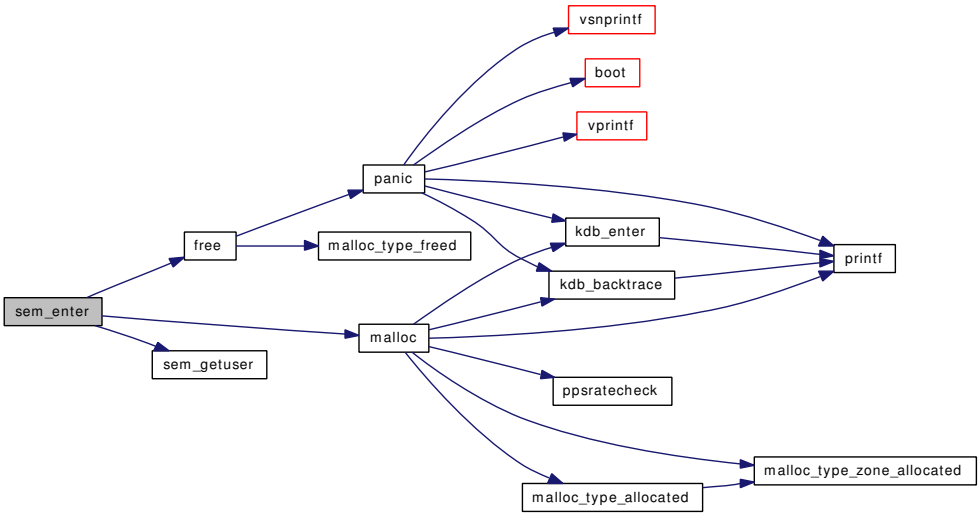
9.145.2.24 static void sem_enter (struct proc * *p*, struct ksem * *ks*) [static]

Definition at line 495 of file uipc_sem.c.

References free(), malloc(), and sem_getuser().

Referenced by kern_sem_open(), sem_create(), and sem_forkhook().

Here is the call graph for this function:

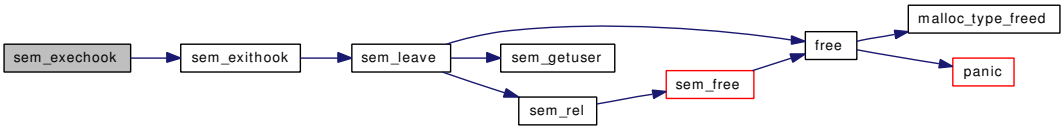


9.145.2.25 static void sem_exechook (void * *arg*, struct proc * *p*, struct image_params *imgp *__unused*) [static]

Definition at line 932 of file uipc_sem.c.

References sem_exithook().

Here is the call graph for this function:



9.145.2.26 static void sem_exechook (void * *arg*, struct proc * *p*, struct image_params * *imgp*) [static]

Referenced by sem_modload().

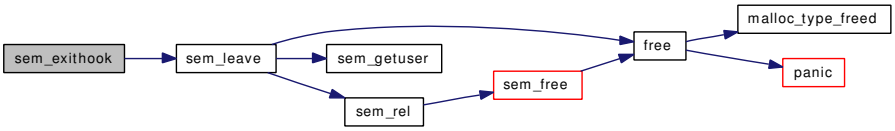
9.145.2.27 static void sem_exithook (void * *arg*, struct proc * *p*) [static]

Definition at line 938 of file uipc_sem.c.

References sem_leave().

Referenced by sem_exechook(), and sem_modload().

Here is the call graph for this function:



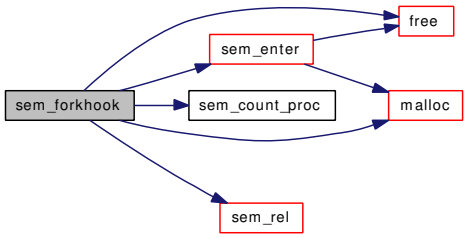
9.145.2.28 static void sem_forkhook (void * *arg*, struct proc * *p1*, struct proc * *p2*, int *flags*) [static]

Definition at line 864 of file uipc_sem.c.

References free(), malloc(), sem_count_proc(), sem_enter(), and sem_rel().

Referenced by sem_modload().

Here is the call graph for this function:



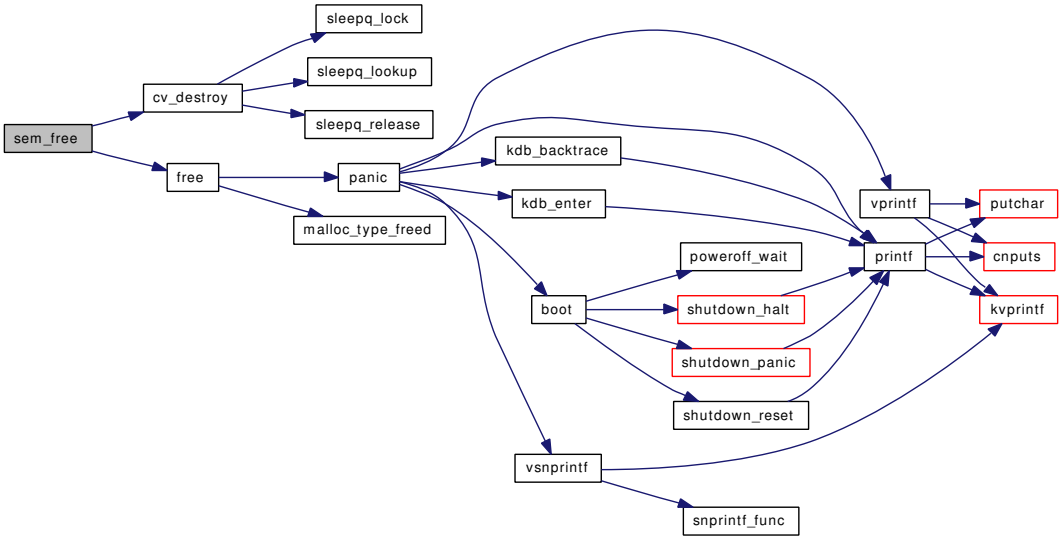
9.145.2.29 static void sem_free (struct ksem * *ksnew*) [static]

Definition at line 442 of file uipc_sem.c.

References cv_destroy(), and free().

Referenced by sem_create(), and sem_rel().

Here is the call graph for this function:



9.145.2.30 static __inline struct kuser * sem_getuser (struct proc * *p*, struct ksem * *ks*) [static]

Definition at line 457 of file uipc_sem.c.

Referenced by sem_enter(), sem_hasopen(), and sem_leave().

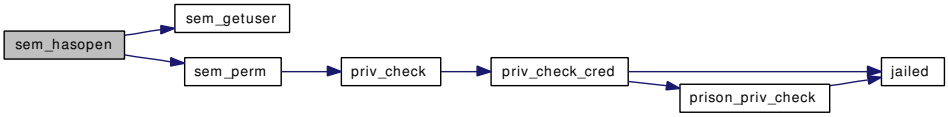
9.145.2.31 static int sem_hasopen (struct thread * *td*, struct ksem * *ks*) [static]

Definition at line 468 of file uipc_sem.c.

References sem_getuser(), and sem_perm().

Referenced by kern_sem_post(), kern_sem_wait(), ksem_destroy(), and ksem_getvalue().

Here is the call graph for this function:



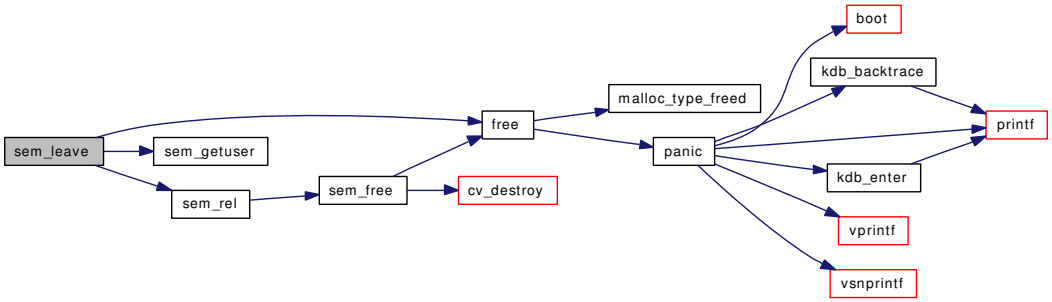
9.145.2.32 static int sem_leave (struct proc * *p*, struct ksem * *ks*) [static]

Definition at line 476 of file uipc_sem.c.

References DP, free(), sem_getuser(), and sem_rel().

Referenced by kern_sem_close(), kern_sem_open(), sem_create(), and sem_exithook().

Here is the call graph for this function:



9.145.2.33 static struct ksem * sem_lookup_byname (const char * *name*) [static]

Definition at line 165 of file uipc_sem.c.

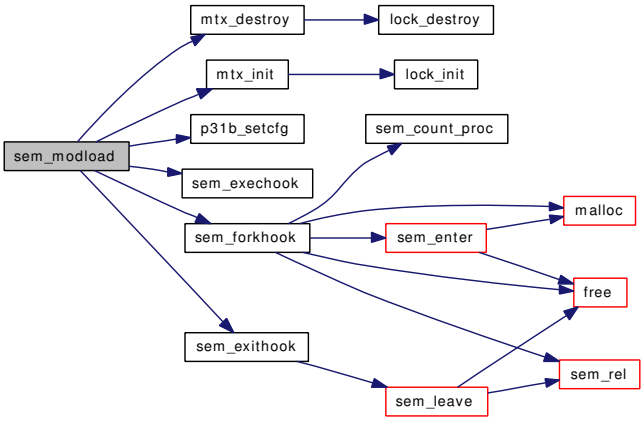
Referenced by kern_sem_open(), and kern_sem_unlink().

9.145.2.34 static int sem_modload (struct module * *module*, int *cmd*, void * *arg*) [static]

Definition at line 959 of file uipc_sem.c.

References mtx_destroy(), mtx_init(), p31b_setcfg(), sem_exechook(), sem_exithook(), sem_forkhook(), and SEM_MAX.

Here is the call graph for this function:



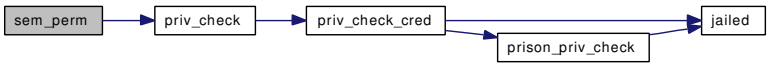
9.145.2.35 static int sem_perm (struct thread * *td*, struct ksem * *ks*) [static]

Definition at line 418 of file uipc_sem.c.

References DP, priv_check(), and uc.

Referenced by kern_sem_open(), kern_sem_unlink(), and sem_hasopen().

Here is the call graph for this function:



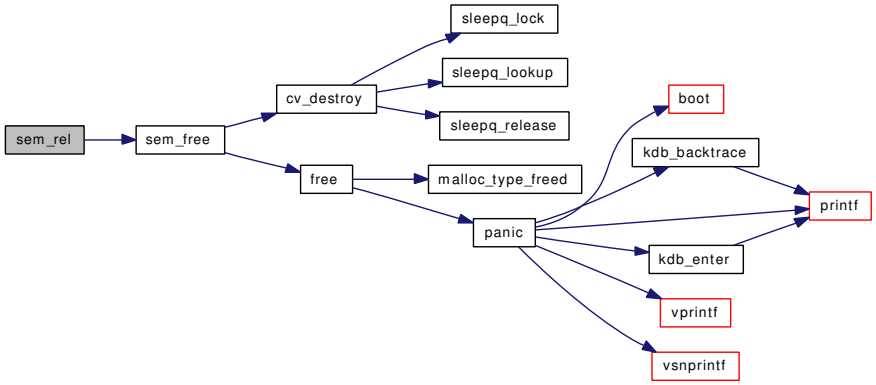
9.145.2.36 static __inline void sem_rel (struct ksem * *ks*) [static]

Definition at line 137 of file uipc_sem.c.

References DP, and sem_free().

Referenced by kern_sem_init(), kern_sem_open(), kern_sem_unlink(), kern_sem_wait(), ksem_destroy(), sem_forkhook(), and sem_leave().

Here is the call graph for this function:



9.145.2.37 SYSCALL_MODULE_HELPER (ksem_destroy)

9.145.2.38 SYSCALL_MODULE_HELPER (ksem_getvalue)

9.145.2.39 SYSCALL_MODULE_HELPER (ksem_trywait)

9.145.2.40 SYSCALL_MODULE_HELPER (ksem_timedwait)

9.145.2.41 SYSCALL_MODULE_HELPER (ksem_wait)

9.145.2.42 SYSCALL_MODULE_HELPER (ksem_post)

9.145.2.43 SYSCALL_MODULE_HELPER (ksem_close)

9.145.2.44 SYSCALL_MODULE_HELPER (ksem_unlink)

9.145.2.45 SYSCALL_MODULE_HELPER (ksem_open)

9.145.2.46 SYSCALL_MODULE_HELPER (ksem_init)

9.145.3 Variable Documentation

9.145.3.1 moduledata_t sem_mod [static]

Initial value:

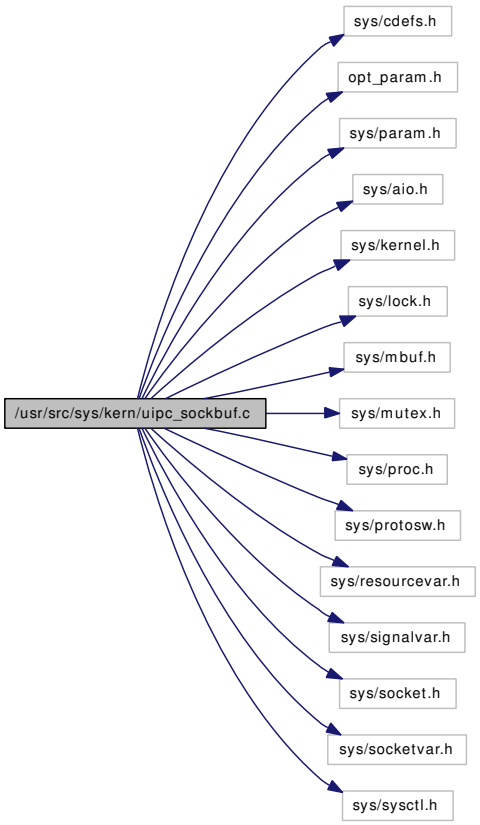
```
 {
        "sem",
        &sem_modload,
        NULL
}
```

Definition at line 993 of file uipc_sem.c.

9.146 /usr/src/sys/kern/uipc_sockbuf.c File Reference

```
#include <sys/cdefs.h>
#include "opt_param.h"
#include <sys/param.h>
#include <sys/aio.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/mbuf.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/protosw.h>
#include <sys/resourcevar.h>
#include <sys/signalvar.h>
#include <sys/socket.h>
#include <sys/socketvar.h>
#include <sys/sysctl.h>
```

Include dependency graph for uipc_sockbuf.c:



Defines

- #define SBLINKRECORD(sb, m0)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/uipc_sockbuf.c,v 1.165 2006/09/06 21:59:36 jhb Exp $")
- static void sbdrop_internal (struct sockbuf *sb, int len)
- static void sbflush_internal (struct sockbuf *sb)
- static void sbrelease_internal (struct sockbuf *sb, struct socket *so)
- void socantsendmore_locked (struct socket *so)
- void socantsendmore (struct socket *so)
- void socantrcvmore_locked (struct socket *so)
- void socantrcvmore (struct socket *so)
- int sbwait (struct sockbuf *sb)
- int sb_lock (struct sockbuf *sb)
- void sowakeup (struct socket *so, struct sockbuf *sb)
- int soreserve (struct socket *so, u_long sndcc, u_long rcvcc)
- static int sysctl_handle_sb_max (SYSCTL_HANDLER_ARGS)
- int sbreserve_locked (struct sockbuf *sb, u_long cc, struct socket *so, struct thread *td)
- int sbreserve (struct sockbuf *sb, u_long cc, struct socket *so, struct thread *td)
- void sbrelease_locked (struct sockbuf *sb, struct socket *so)
- void sbrelease (struct sockbuf *sb, struct socket *so)
- void sbdestroy (struct sockbuf *sb, struct socket *so)
- void sbappend_locked (struct sockbuf *sb, struct mbuf *m)
- void sbappend (struct sockbuf *sb, struct mbuf *m)
- void sbappendstream_locked (struct sockbuf *sb, struct mbuf *m)
- void sbappendstream (struct sockbuf *sb, struct mbuf *m)
- void sbappendrecord_locked (struct sockbuf *sb, struct mbuf *m0)
- void sbappendrecord (struct sockbuf *sb, struct mbuf *m0)
- int sbappendaddr_locked (struct sockbuf *sb, const struct sockaddr *asa, struct mbuf *m0, struct mbuf *control)
- int sbappendaddr (struct sockbuf *sb, const struct sockaddr *asa, struct mbuf *m0, struct mbuf *control)
- int sbappendcontrol_locked (struct sockbuf *sb, struct mbuf *m0, struct mbuf *control)
- int sbappendcontrol (struct sockbuf *sb, struct mbuf *m0, struct mbuf *control)
- void sbcompress (struct sockbuf *sb, struct mbuf *m, struct mbuf *n)
- void sbflush_locked (struct sockbuf *sb)
- void sbflush (struct sockbuf *sb)
- void sbdrop_locked (struct sockbuf *sb, int len)
- void sbdrop (struct sockbuf *sb, int len)
- void sbdroprecord_locked (struct sockbuf *sb)
- void sbdroprecord (struct sockbuf *sb)
- SYSCTL_INT (_kern, KERN_DUMMY, dummy, CTLFLAG_RW,&dummy, 0,"")
- SYSCTL_OID (_kern_ipc, KIPC_MAXSOCKBUF, maxsockbuf, CTLTYPE_-ULONG|CTLFLAG_RW,&sb_max, 0, sysctl_handle_sb_max,"LU","Maximum socket buffer size")
- SYSCTL_ULONG (_kern_ipc, KIPC_SOCKBUF_WASTE, sockbuf_waste_factor, CTLFLAG_-RW,&sb_efficiency, 0,"")

Variables

- void(∗) aio_swake (struct socket ∗, struct sockbuf ∗)
- u_long sb_max = SB_MAX
- static u_long sb_max_adj
- static u_long sb_efficiency = 8
- static int dummy

9.146.1 Define Documentation

9.146.1.1 #define SBLINKRECORD(sb, m0)

Value:

```
do {                                    \
        SOCKBUF_LOCK_ASSERT(sb);                                        \
        if ((sb)->sb_lastrecord != NULL)                                \
                (sb)->sb_lastrecord->m_nextpkt = (m0);                  \
        else                                                            \
                (sb)->sb_mb = (m0);                                     \
        (sb)->sb_lastrecord = (m0);                                     \
} while (/*CONSTCOND*/0)
```

Definition at line 426 of file uipc_sockbuf.c.

Referenced by sbappendaddr_locked(), sbappendcontrol_locked(), and sbappendrecord_locked().

9.146.2 Function Documentation

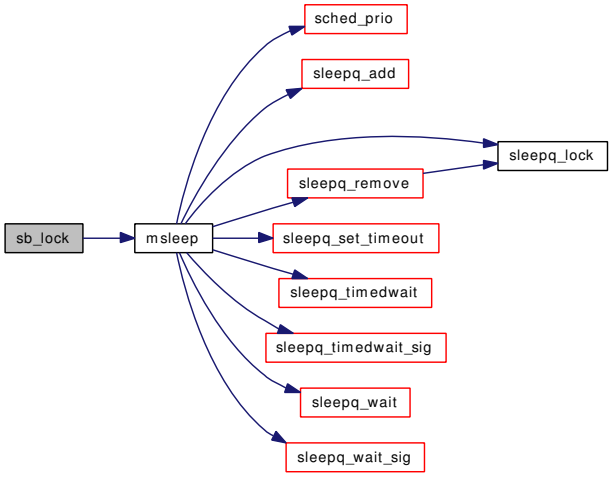
9.146.2.1 __FBSDID ("$FreeBSD: src/sys/kern/uipc_sockbuf. *c*, v 1.165 2006/09/06 21:59:36 jhb Exp $")

9.146.2.2 int sb_lock (struct sockbuf ∗ *sb*)

Definition at line 140 of file uipc_sockbuf.c.

References msleep().

Here is the call graph for this function:



9.146.2.3 void sbappend (struct sockbuf * *sb*, struct mbuf * *m*)

Definition at line 492 of file uipc_sockbuf.c.

References sbappend_locked().

Here is the call graph for this function:

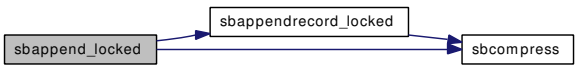
9.146.2.4 void sbappend_locked (struct sockbuf * *sb*, struct mbuf * *m*)

Definition at line 441 of file uipc_sockbuf.c.

References sbappendrecord_locked(), and sbcompress().

Referenced by sbappend(), and uipc_send().

Here is the call graph for this function:

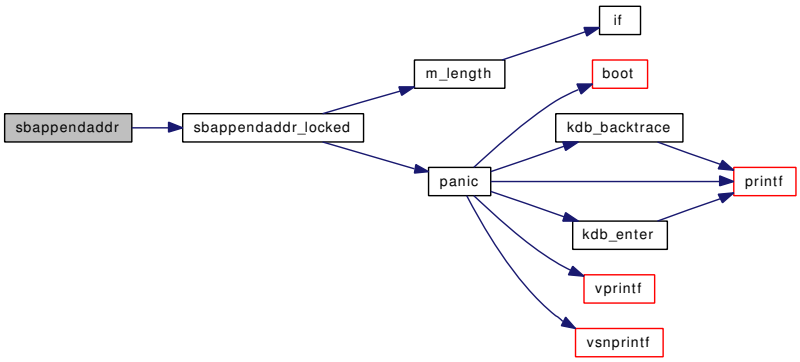


9.146.2.5 int sbappendaddr (struct sockbuf * *sb*, const struct sockaddr * *asa*, struct mbuf * *m0*, struct mbuf * *control*)

Definition at line 667 of file uipc_sockbuf.c.

References sbappendaddr_locked().

Here is the call graph for this function:



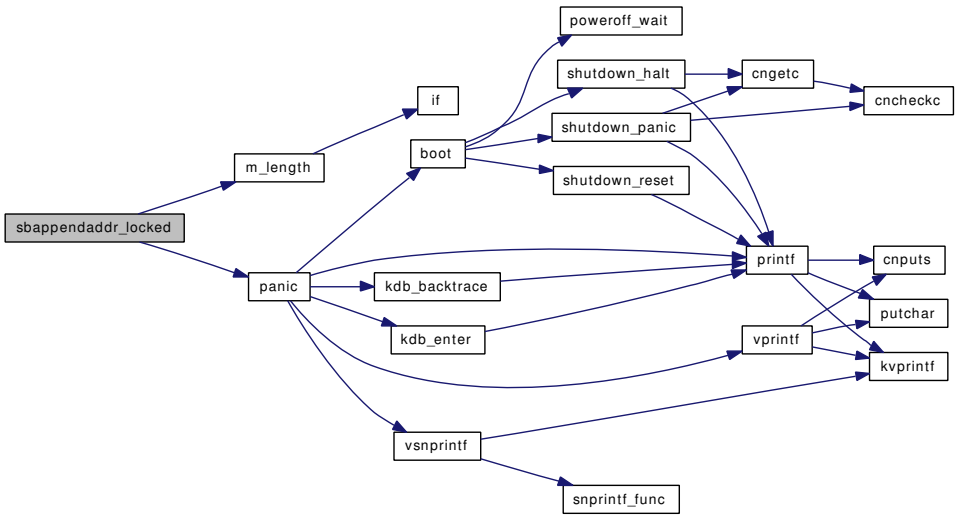
9.146.2.6 int sbappendaddr_locked (struct sockbuf * *sb*, const struct sockaddr * *asa*, struct mbuf * *m0*, struct mbuf * *control*)

Definition at line 617 of file uipc_sockbuf.c.

References m_length(), panic(), and SBLINKRECORD.

Referenced by sbappendaddr(), and uipc_send().

Here is the call graph for this function:

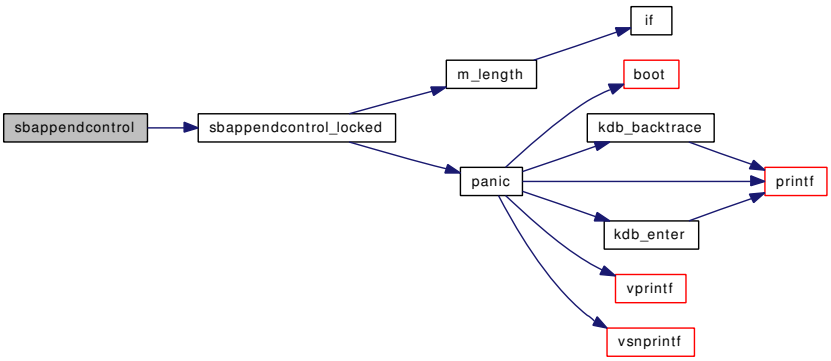


9.146.2.7 int sbappendcontrol (struct sockbuf * *sb*, struct mbuf * *m0*, struct mbuf * *control*)

Definition at line 711 of file uipc_sockbuf.c.

References sbappendcontrol_locked().

Here is the call graph for this function:



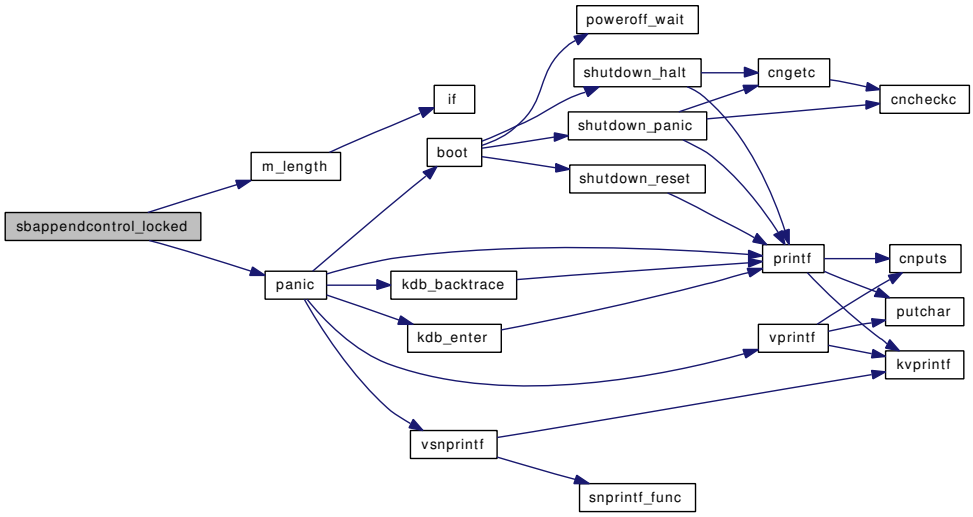
9.146.2.8 int sbappendcontrol_locked (struct sockbuf * *sb*, struct mbuf * *m0*, struct mbuf * *control*)

Definition at line 679 of file uipc_sockbuf.c.

References m_length(), panic(), and SBLINKRECORD.

Referenced by sbappendcontrol(), and uipc_send().

Here is the call graph for this function:

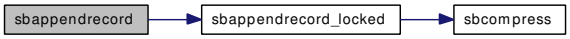


9.146.2.9 void sbappendrecord (struct sockbuf * *sb*, struct mbuf * *m0*)

Definition at line 602 of file uipc_sockbuf.c.

References sbappendrecord_locked().

Here is the call graph for this function:



9.146.2.10 void sbappendrecord_locked (struct sockbuf * *sb*, struct mbuf * *m0*)

Definition at line 566 of file uipc_sockbuf.c.

References sbcompress(), and SBLINKRECORD.

Referenced by sbappend_locked(), and sbappendrecord().

Here is the call graph for this function:

9.146.2.11 void sbappendstream (struct sockbuf * *sb*, struct mbuf * *m*)

Definition at line 527 of file uipc_sockbuf.c.

References sbappendstream_locked().

Here is the call graph for this function:

9.146.2.12 void sbappendstream_locked (struct sockbuf * *sb*, struct mbuf * *m*)

Definition at line 506 of file uipc_sockbuf.c.

References sbcompress().

Referenced by sbappendstream().

Here is the call graph for this function:

9.146.2.13 void sbcompress (struct sockbuf * *sb*, struct mbuf * *m*, struct mbuf * *n*)

Definition at line 742 of file uipc_sockbuf.c.

Referenced by sbappend_locked(), sbappendrecord_locked(), and sbappendstream_locked().

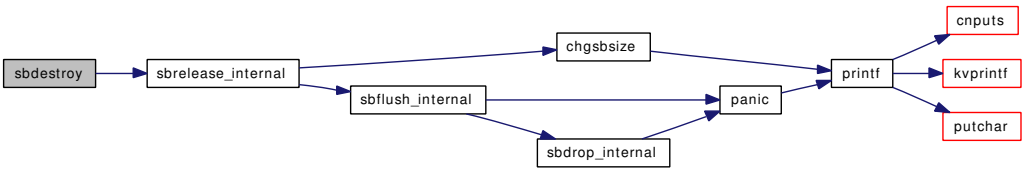
9.146.2.14 void sbdestroy (struct sockbuf * *sb*, struct socket * *so*)

Definition at line 347 of file uipc_sockbuf.c.

References sbrelease_internal().

Referenced by sofree().

Here is the call graph for this function:

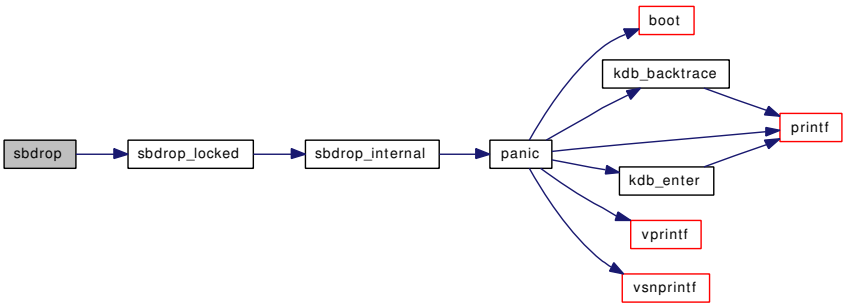


9.146.2.15 void sbdrop (struct sockbuf * *sb*, int *len*)

Definition at line 898 of file uipc_sockbuf.c.

References sbdrop_locked().

Here is the call graph for this function:



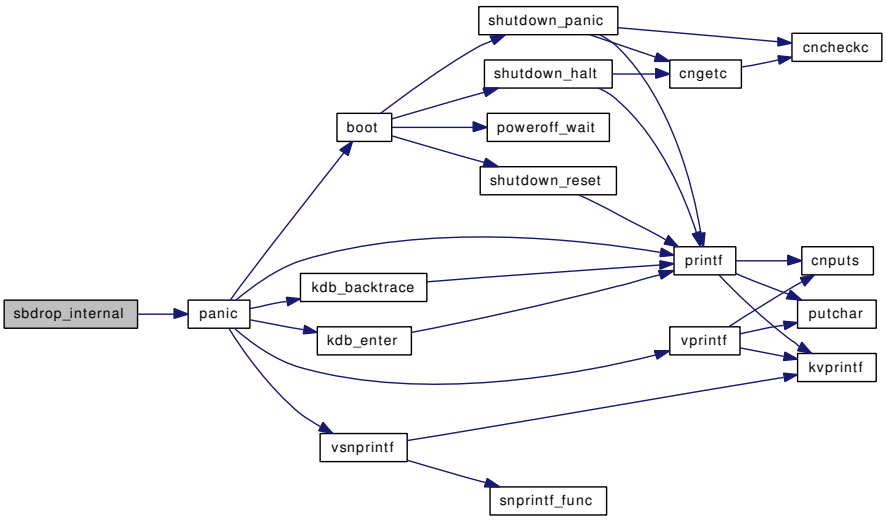
9.146.2.16 static void sbdrop_internal (struct sockbuf * *sb*, int *len*) [static]

Definition at line 837 of file uipc_sockbuf.c.

References panic().

Referenced by sbdrop_locked(), and sbflush_internal().

Here is the call graph for this function:



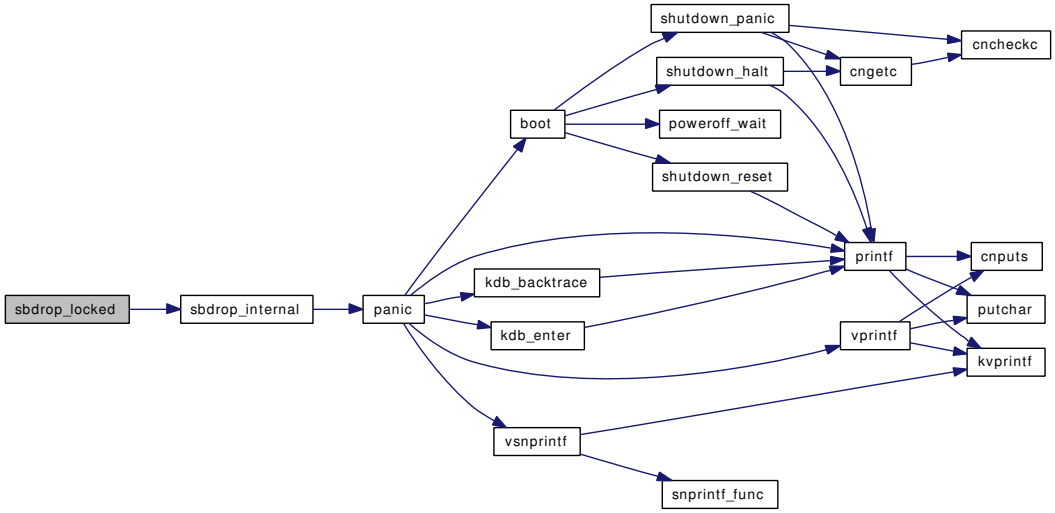
9.146.2.17 void sbdrop_locked (struct sockbuf * *sb*, int *len*)

Definition at line 889 of file uipc_sockbuf.c.

References sbdrop_internal().

Referenced by sbdrop(), and soisdisconnected().

Here is the call graph for this function:

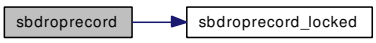


9.146.2.18 void sbdroprecord (struct sockbuf * *sb*)

Definition at line 933 of file uipc_sockbuf.c.

References sbdroprecord_locked().

Here is the call graph for this function:



9.146.2.19 void sbdroprecord_locked (struct sockbuf * *sb*)

Definition at line 911 of file uipc_sockbuf.c.

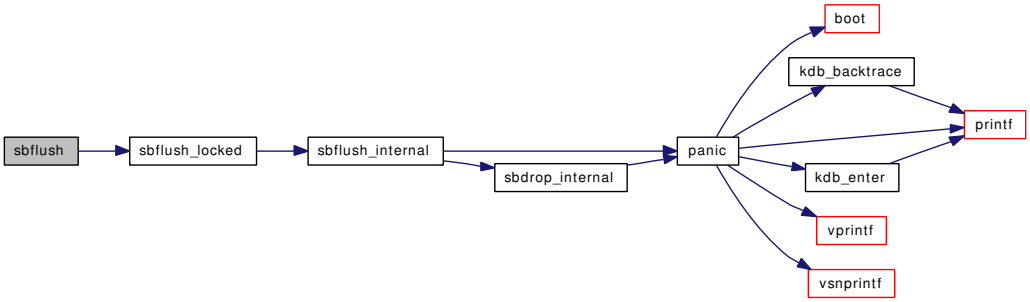
Referenced by sbdroprecord().

9.146.2.20 void sbflush (struct sockbuf * *sb*)

Definition at line 825 of file uipc_sockbuf.c.

References sbflush_locked().

Here is the call graph for this function:



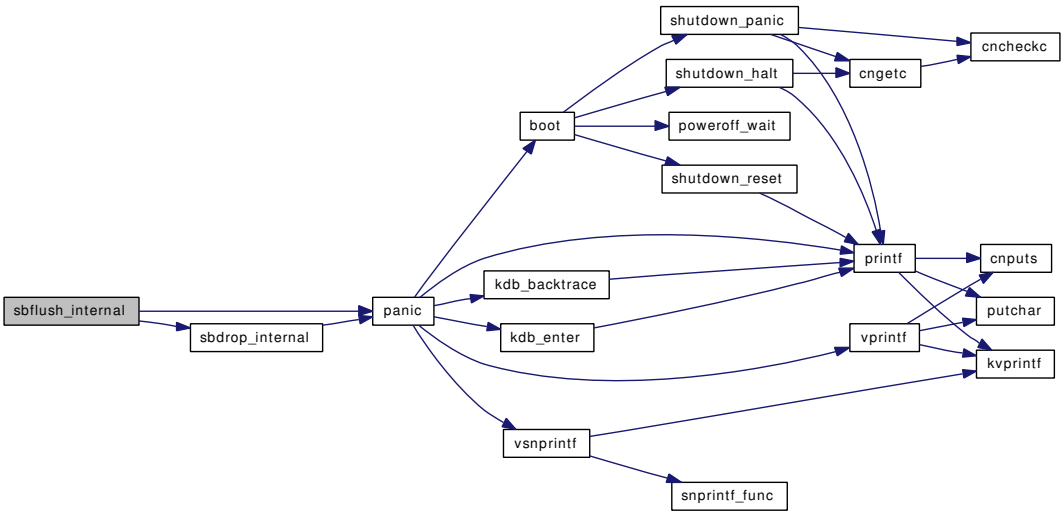
9.146.2.21 static void sbflush_internal (struct sockbuf * *sb*) [static]

Definition at line 797 of file uipc_sockbuf.c.

References panic(), and sbdrop_internal().

Referenced by sbflush_locked(), and sbrelease_internal().

Here is the call graph for this function:



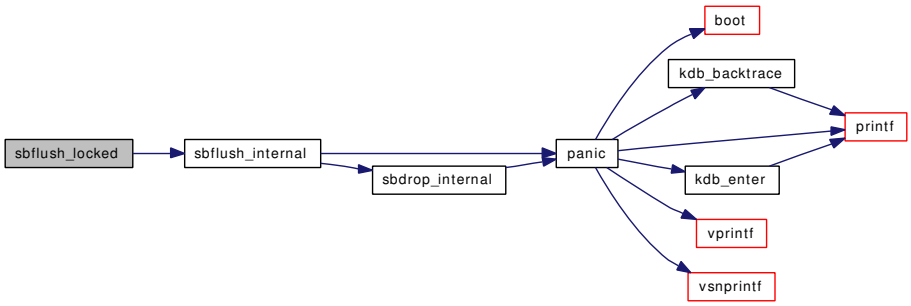
9.146.2.22 void sbflush_locked (struct sockbuf * *sb*)

Definition at line 817 of file uipc_sockbuf.c.

References sbflush_internal().

Referenced by sbflush().

Here is the call graph for this function:



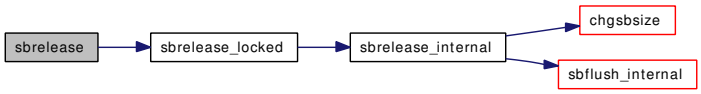
9.146.2.23 void sbrelease (struct sockbuf * *sb*, struct socket * *so*)

Definition at line 338 of file uipc_sockbuf.c.

References sbrelease_locked().

Referenced by sorflush().

Here is the call graph for this function:



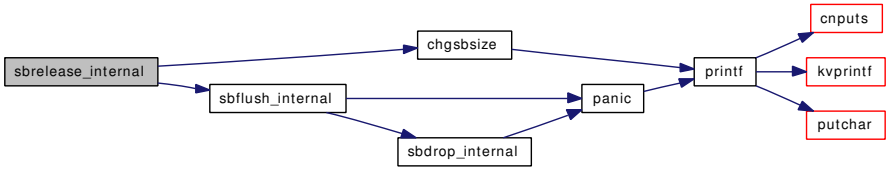
9.146.2.24 static void sbrelease_internal (struct sockbuf * *sb*, struct socket * *so*) [static]

Definition at line 319 of file uipc_sockbuf.c.

References chgsbsize(), and sbflush_internal().

Referenced by sbdestroy(), and sbrelease_locked().

Here is the call graph for this function:



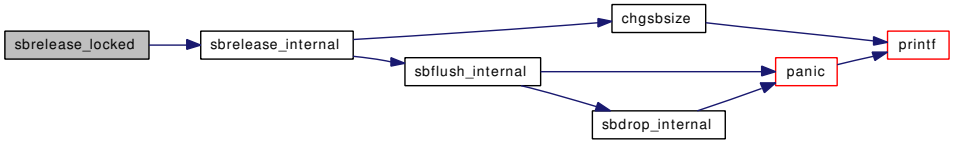
9.146.2.25 void sbrelease_locked (struct sockbuf * *sb*, struct socket * *so*)

Definition at line 329 of file uipc_sockbuf.c.

References sbrelease_internal().

Referenced by sbrelease(), and soreserve().

Here is the call graph for this function:



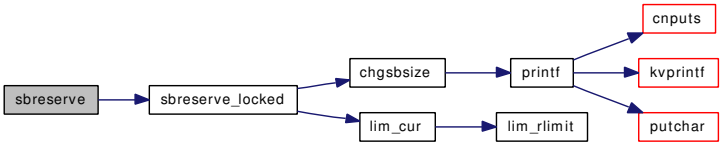
9.146.2.26 int sbreserve (struct sockbuf * *sb*, u_long *cc*, struct socket * *so*, struct thread * *td*)

Definition at line 304 of file uipc_sockbuf.c.

References sbreserve_locked().

Referenced by sosetopt().

Here is the call graph for this function:



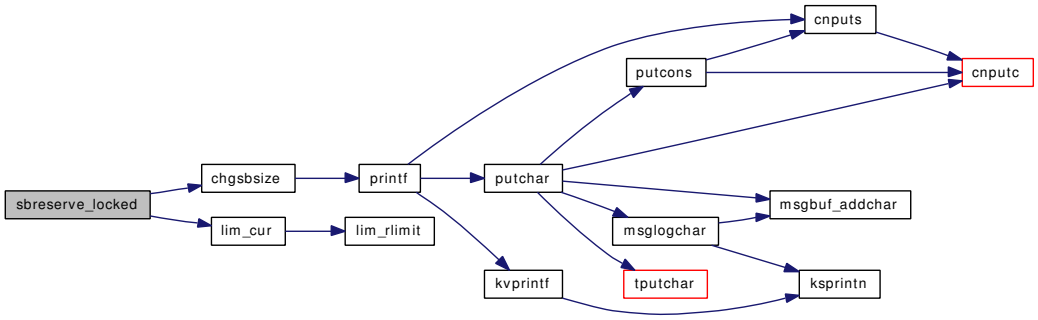
9.146.2.27 int sbreserve_locked (struct sockbuf * *sb*, u_long *cc*, struct socket * *so*, struct thread * *td*)

Definition at line 273 of file uipc_sockbuf.c.

References chgsbsize(), lim_cur(), sb_efficiency, sb_max, and sb_max_adj.

Referenced by sbreserve(), and soreserve().

Here is the call graph for this function:



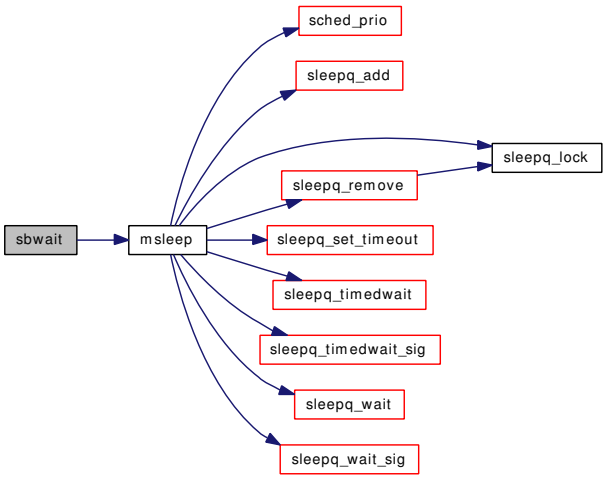
9.146.2.28 int sbwait (struct sockbuf * *sb*)

Definition at line 124 of file uipc_sockbuf.c.

References msleep().

Referenced by kern_sendfile(), and sosend_generic().

Here is the call graph for this function:



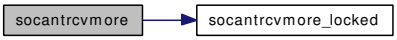
9.146.2.29 void socantrcvmore (struct socket * *so*)

Definition at line 112 of file uipc_sockbuf.c.

References socantrcvmore_locked().

Referenced by unp_shutdown().

Here is the call graph for this function:



9.146.2.30 void socantrcvmore_locked (struct socket * *so*)

Definition at line 101 of file uipc_sockbuf.c.

Referenced by socantrcvmore(), and sorflush().

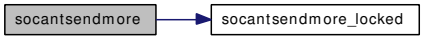
9.146.2.31 void socantsendmore (struct socket * *so*)

Definition at line 92 of file uipc_sockbuf.c.

References socantsendmore_locked().

Referenced by uipc_send(), and uipc_shutdown().

Here is the call graph for this function:



9.146.2.32 void socantsendmore_locked (struct socket * *so*)

Definition at line 81 of file uipc_sockbuf.c.

Referenced by socantsendmore().

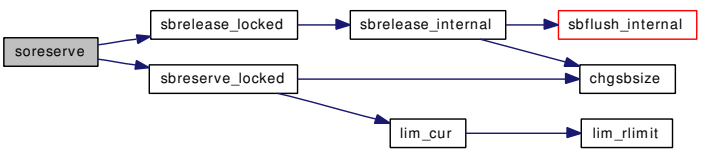
9.146.2.33 int soreserve (struct socket * *so*, u_long *sndcc*, u_long *rcvcc*)

Definition at line 225 of file uipc_sockbuf.c.

References sbrelease_locked(), sbreserve_locked(), and td.

Referenced by sonewconn(), and uipc_attach().

Here is the call graph for this function:

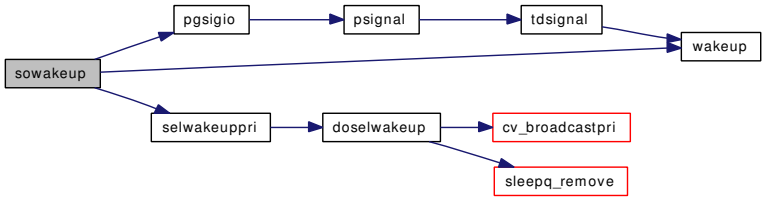


9.146.2.34 void sowakeup (struct socket * *so*, struct sockbuf * *sb*)

Definition at line 171 of file uipc_sockbuf.c.

References aio_swake, pgsigio(), selwakeuppri(), and wakeup().

Here is the call graph for this function:



9.146.2.35 static int sysctl_handle_sb_max (SYSCTL_HANDLER_ARGS) [static]

Definition at line 253 of file uipc_sockbuf.c.

References sb_max, sb_max_adj, and sysctl_handle_long().

Here is the call graph for this function:

9.146.2.36 SYSCTL_INT (_kern, KERN_DUMMY, dummy, CTLFLAG_RW, & *dummy*, 0, "")

9.146.2.37 SYSCTL_OID (_kern_ipc, KIPC_MAXSOCKBUF, maxsockbuf, CTLTYPE_ULONG| *CTLFLAG_RW*, & *sb_max*, 0, sysctl_handle_sb_max, "LU", "Maximum socket buffer size")

9.146.2.38 SYSCTL_ULONG (_kern_ipc, KIPC_SOCKBUF_WASTE, sockbuf_waste_factor, CTLFLAG_RW, & *sb_efficiency*, 0, "")

9.146.3 Variable Documentation

9.146.3.1 void(*) aio_swake(struct socket *, struct sockbuf *)

Definition at line 55 of file uipc_sockbuf.c.

Referenced by aio_onceonly(), aio_unload(), and sowakeup().

9.146.3.2 int dummy [static]

Definition at line 942 of file uipc_sockbuf.c.

Referenced by lookup(), and sysctl_wire_old_buffer().

9.146.3.3 u_long sb_efficiency = 8 [static]

Definition at line 65 of file uipc_sockbuf.c.

Referenced by sbreserve_locked().

9.146.3.4 u_long sb_max = SB_MAX

Definition at line 61 of file uipc_sockbuf.c.

Referenced by sbreserve_locked(), and sysctl_handle_sb_max().

9.146.3.5 u_long sb_max_adj [static]

Initial value:

```
SB_MAX * MCLBYTES / (MSIZE + MCLBYTES)
```

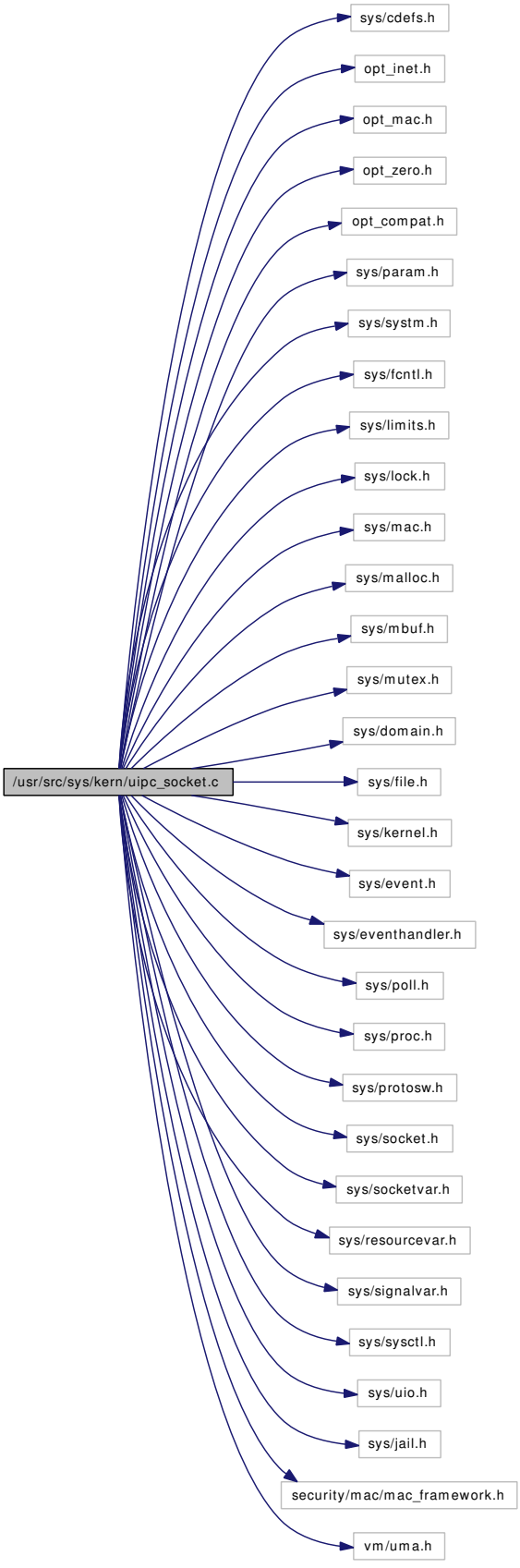
Definition at line 62 of file uipc_sockbuf.c.

Referenced by sbreserve_locked(), and sysctl_handle_sb_max().

9.147 /usr/src/sys/kern/uipc_socket.c File Reference

```
#include <sys/cdefs.h>
#include "opt_inet.h"
#include "opt_mac.h"
#include "opt_zero.h"
#include "opt_compat.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/fcntl.h>
#include <sys/limits.h>
#include <sys/lock.h>
#include <sys/mac.h>
#include <sys/malloc.h>
#include <sys/mbuf.h>
#include <sys/mutex.h>
#include <sys/domain.h>
#include <sys/file.h>
#include <sys/kernel.h>
#include <sys/event.h>
#include <sys/eventhandler.h>
#include <sys/poll.h>
#include <sys/proc.h>
#include <sys/protosw.h>
#include <sys/socket.h>
#include <sys/socketvar.h>
#include <sys/resourcevar.h>
#include <sys/signalvar.h>
#include <sys/sysctl.h>
#include <sys/uio.h>
#include <sys/jail.h>
#include <security/mac/mac_framework.h>
#include <vm/uma.h>
```

Include dependency graph for uipc_socket.c:


/usr/src/sys/kern/uipc_socket.c
sys/cdefs.h
opt_inet.h
opt_mac.h
opt_zero.h
opt_compat.h
sys/param.h
sys/systm.h
sys/fcntl.h
sys/limits.h
sys/lock.h
sys/mac.h
sys/malloc.h
sys/mbuf.h
sys/mutex.h
sys/domain.h
sys/file.h
sys/kernel.h
sys/event.h
sys/eventhandler.h
sys/poll.h
sys/proc.h
sys/protosw.h
sys/socket.h
sys/socketvar.h
sys/resourcevar.h
sys/signalvar.h
sys/sysctl.h
sys/uio.h
sys/jail.h
security/mac/mac_framework.h
vm/uma.h

Defines

- #define SBLOCKWAIT(f) (((f) & MSG_DONTWAIT) ? M_NOWAIT : M_WAITOK)
- #define snderr(errno) { error = (errno); goto release; }

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/uipc_socket.c,v 1.292 2007/02/15 10:11:00 rwatson Exp $")
- static int soreceive_rcvoob (struct socket *so, struct uio *uio, int flags)
- static void filt_sordetach (struct knote *kn)
- static int filt_soread (struct knote *kn, long hint)
- static void filt_sowdetach (struct knote *kn)
- static int filt_sowrite (struct knote *kn, long hint)
- static int filt_solisten (struct knote *kn, long hint)
- MALLOC_DEFINE (M_SONAME,"soname","socket name")
- MALLOC_DEFINE (M_PCB,"pcb","protocol control block")
- static int sysctl_somaxconn (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern_ipc, KIPC_SOMAXCONN, somaxconn, CTLTYPE_UINT|CTLFLAG_-RW, 0, sizeof(int), sysctl_somaxconn,"I","Maximum pending socket connection ""queue size")
- SYSCTL_INT (_kern_ipc, OID_AUTO, numopensockets, CTLFLAG_RD,&numopensockets, 0,"Number of open sockets")
- MTX_SYSINIT (accept_mtx,&accept_mtx,"accept", MTX_DEF)
- MTX_SYSINIT (so_global_mtx,&so_global_mtx,"so_glabel", MTX_DEF)
- SYSCTL_NODE (_kern, KERN_IPC, ipc, CTLFLAG_RW, 0,"IPC")
- static int sysctl_maxsockets (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_kern_ipc, OID_AUTO, maxsockets, CTLTYPE_INT|CTLFLAG_-RW,&maxsockets, 0, sysctl_maxsockets,"IU","Maximum number of sockets avaliable")
- static void init_maxsockets (void *ignored)
- SYSINIT (param, SI_SUB_TUNABLES, SI_ORDER_ANY, init_maxsockets, NULL)
- static struct socket * soalloc (int mflags)
- static void sodealloc (struct socket *so)
- int socreate (int dom, struct socket **aso, int type, int proto, struct ucred *cred, struct thread *td)
- socket * sonewconn (struct socket *head, int connstatus)
- int sobind (struct socket *so, struct sockaddr *nam, struct thread *td)
- int solisten (struct socket *so, int backlog, struct thread *td)
- int solisten_proto_check (struct socket *so)
- void solisten_proto (struct socket *so, int backlog)
- void sofree (struct socket *so)
- int soclose (struct socket *so)
- void soabort (struct socket *so)
- int soaccept (struct socket *so, struct sockaddr **nam)
- int soconnect (struct socket *so, struct sockaddr *nam, struct thread *td)
- int soconnect2 (struct socket *so1, struct socket *so2)
- int sodisconnect (struct socket *so)
- int sosend_dgram (struct socket *so, struct sockaddr *addr, struct uio *uio, struct mbuf *top, struct mbuf *control, int flags, struct thread *td)
- int sosend_generic (struct socket *so, struct sockaddr *addr, struct uio *uio, struct mbuf *top, struct mbuf *control, int flags, struct thread *td)
- int sosend (struct socket *so, struct sockaddr *addr, struct uio *uio, struct mbuf *top, struct mbuf *control, int flags, struct thread *td)

- static __inline void sockbuf_pushsync (struct sockbuf *sb, struct mbuf *nextrecord)
- int soreceive_generic (struct socket *so, struct sockaddr **psa, struct uio *uio, struct mbuf **mp0, struct mbuf **controlp, int *flagsp)
- int soreceive (struct socket *so, struct sockaddr **psa, struct uio *uio, struct mbuf **mp0, struct mbuf **controlp, int *flagsp)
- int soshutdown (struct socket *so, int how)
- void sorflush (struct socket *so)
- int sooptcopyin (struct sockopt *sopt, void *buf, size_t len, size_t minlen)
- int so_setsockopt (struct socket *so, int level, int optname, void *optval, size_t optlen)
- int sosetopt (struct socket *so, struct sockopt *sopt)
- int sooptcopyout (struct sockopt *sopt, const void *buf, size_t len)
- int sogetopt (struct socket *so, struct sockopt *sopt)
- int soopt_getm (struct sockopt *sopt, struct mbuf **mp)
- int soopt_mcopyin (struct sockopt *sopt, struct mbuf *m)
- int soopt_mcopyout (struct sockopt *sopt, struct mbuf *m)
- void sohasoutofband (struct socket *so)
- int sopoll (struct socket *so, int events, struct ucred *active_cred, struct thread *td)
- int sopoll_generic (struct socket *so, int events, struct ucred *active_cred, struct thread *td)
- int soo_kqfilter (struct file *fp, struct knote *kn)
- int socheckuid (struct socket *so, uid_t uid)

Variables

- static struct filterops solisten_filtops
- static struct filterops soread_filtops
- static struct filterops sowrite_filtops
- uma_zone_t socket_zone
- so_gen_t so_gencnt
- int maxsockets
- static int somaxconn = SOMAXCONN
- static int numopensockets
- mtx accept_mtx
- static struct mtx so_global_mtx

9.147.1 Define Documentation

9.147.1.1 #define SBLOCKWAIT(f) (((f) & MSG_DONTWAIT) ? M_NOWAIT : M_WAITOK)

Definition at line 957 of file uipc_socket.c.

Referenced by soreceive_generic(), and sosend_generic().

9.147.1.2 #define snderr(errno) { error = (errno); goto release; }

Definition at line 1139 of file uipc_socket.c.

Referenced by sosend_generic().

9.147.2 Function Documentation

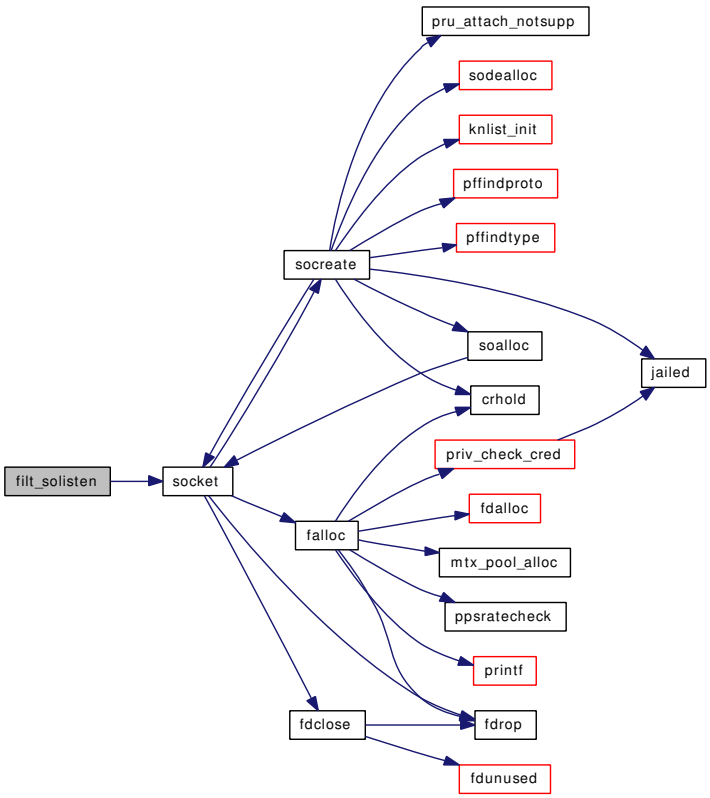
9.147.2.1 __FBSDID ("$FreeBSD: src/sys/kern/uipc_socket. *c*, v 1.292 2007/02/15 10:11:00 rwatson Exp $")

9.147.2.2 static int filt_solisten (struct knote * *kn*, long *hint*) [static]

Definition at line 2682 of file uipc_socket.c.

References socket().

Here is the call graph for this function:

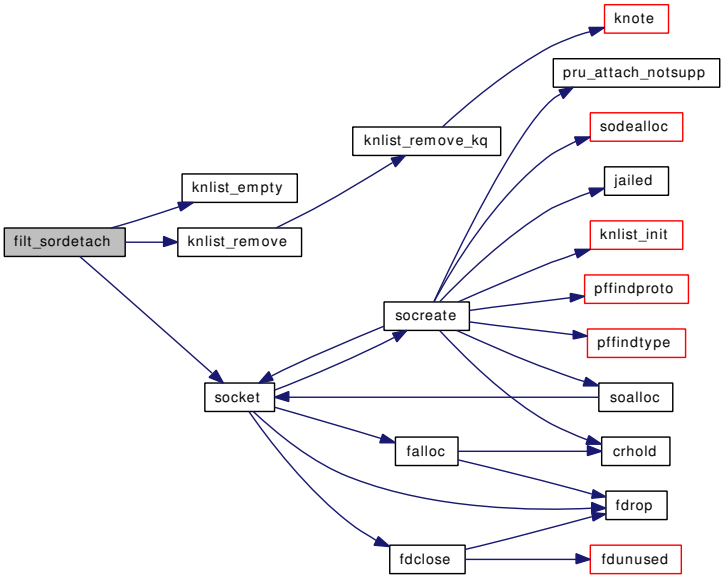


9.147.2.3 static void filt_sordetach (struct knote * *kn*) [static]

Definition at line 2611 of file uipc_socket.c.

References knlist_empty(), knlist_remove(), and socket().

Here is the call graph for this function:

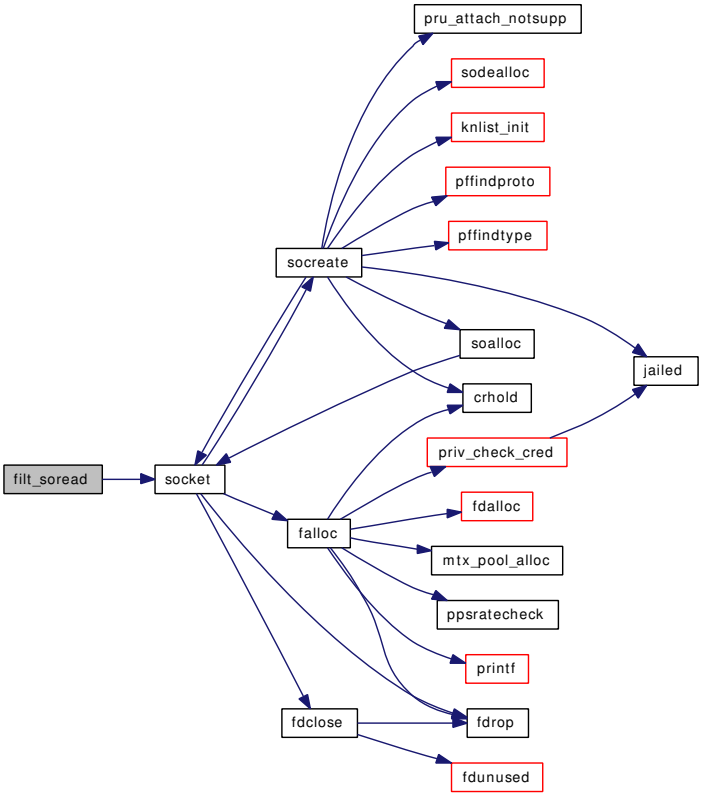


9.147.2.4 static int filt_soread (struct knote * *kn*, long *hint*) [static]

Definition at line 2624 of file uipc_socket.c.

References socket().

Here is the call graph for this function:

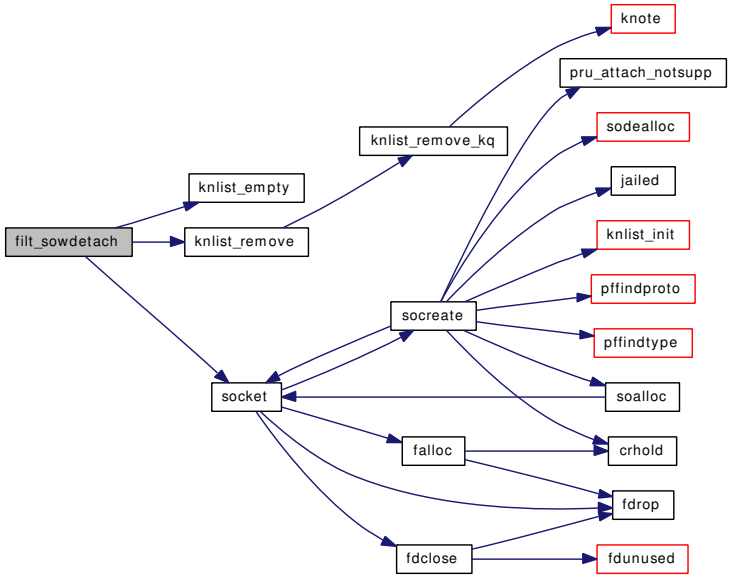


9.147.2.5 static void filt_sowdetach (struct knote * *kn*) [static]

Definition at line 2645 of file uipc_socket.c.

References knlist_empty(), knlist_remove(), and socket().

Here is the call graph for this function:

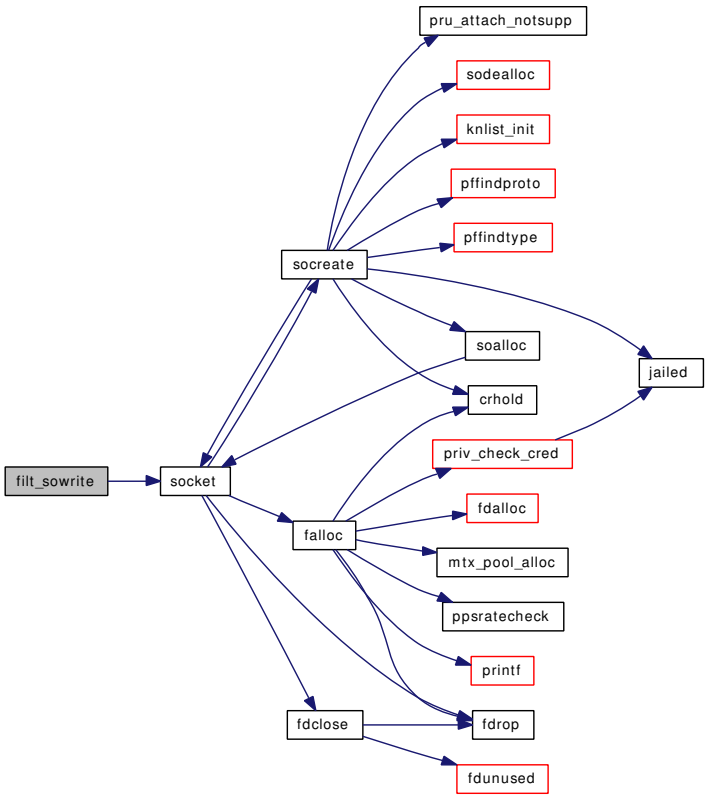


9.147.2.6 static int filt_sowrite (struct knote * *kn*, long *hint*) [static]

Definition at line 2658 of file uipc_socket.c.

References socket().

Here is the call graph for this function:



9.147.2.7 static void init_maxsockets (void * *ignored*) [static]

Definition at line 238 of file uipc_socket.c.

References maxfiles, maxsockets, and nmbclusters.

9.147.2.8 MALLOC_DEFINE (M_PCB, "pcb", "protocol control block")

9.147.2.9 MALLOC_DEFINE (M_SONAME, "soname", "socket name")

9.147.2.10 MTX_SYSINIT (so_global_mtx, & *so_global_mtx*, "so_glabel", MTX_DEF)

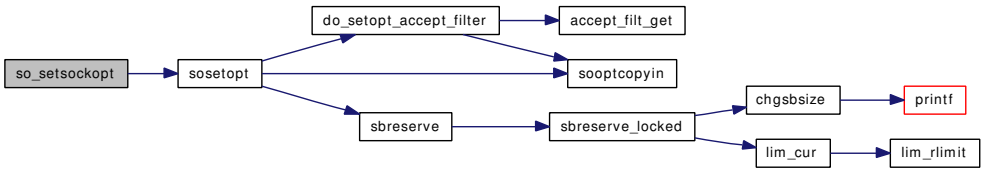
9.147.2.11 MTX_SYSINIT (accept_mtx, & *accept_mtx*, "accept", MTX_DEF)

9.147.2.12 int so_setsockopt (struct socket * *so*, int *level*, int *optname*, void * *optval*, size_t *optlen*)

Definition at line 2018 of file uipc_socket.c.

References sosetopt().

Here is the call graph for this function:



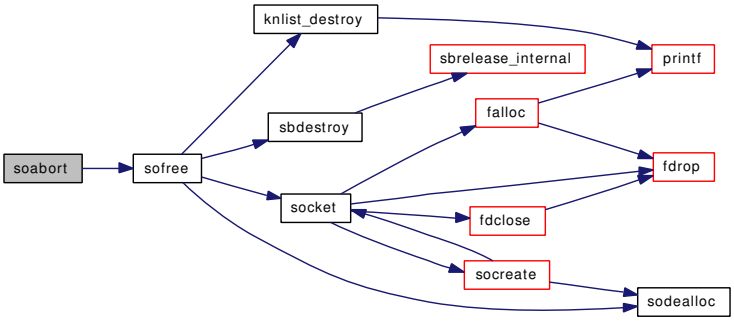
9.147.2.13 void soabort (struct socket * *so*)

Definition at line 722 of file uipc_socket.c.

References sofree().

Referenced by soclose(), and sonewconn().

Here is the call graph for this function:



9.147.2.14 int soaccept (struct socket * *so*, struct sockaddr ** *nam*)

Definition at line 746 of file uipc_socket.c.

Referenced by kern_accept().

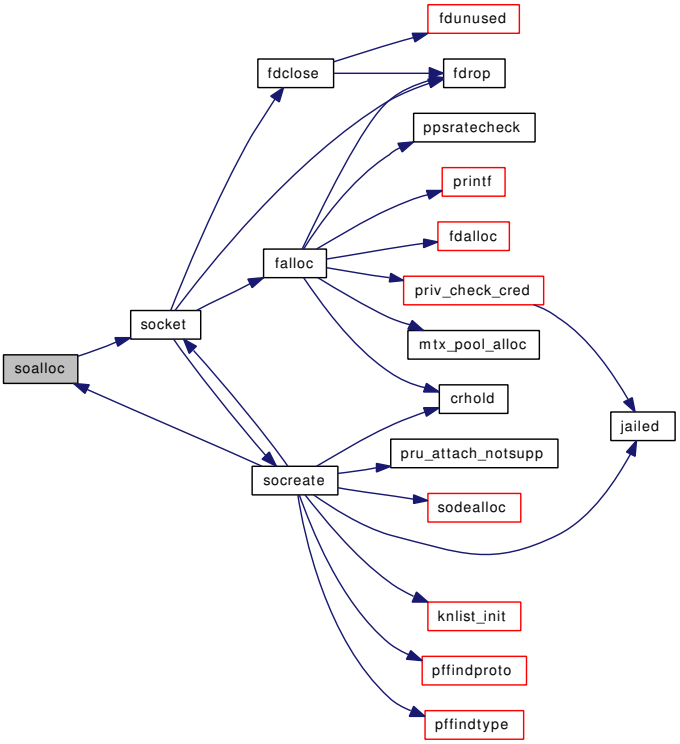
9.147.2.15 static struct socket* soalloc (int *mflags*) [static]

Definition at line 260 of file uipc_socket.c.

References so_gencnt, socket(), and socket_zone.

Referenced by socreate(), and sonewconn().

Here is the call graph for this function:



9.147.2.16 int sobind (struct socket * *so*, struct sockaddr * *nam*, struct thread * *td*)

Definition at line 489 of file uipc_socket.c.

Referenced by kern_bind().

9.147.2.17 int socheckuid (struct socket * *so*, uid_t *uid*)

Definition at line 2691 of file uipc_socket.c.

9.147.2.18 static __inline void sockbuf_pushsync (struct sockbuf * *sb*, struct mbuf * *nextrecord*) [static]

Definition at line 1402 of file uipc_socket.c.

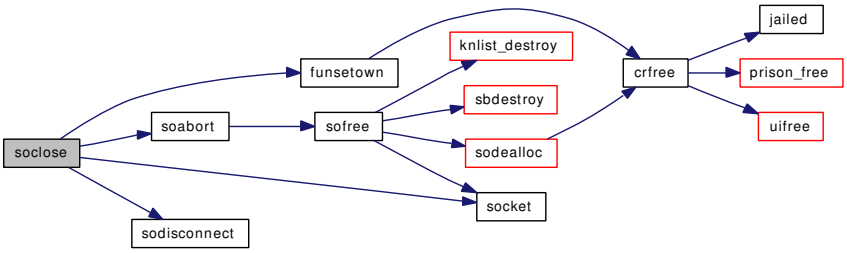
9.147.2.19 int soclose (struct socket * *so*)

Definition at line 646 of file uipc_socket.c.

References funsetown(), hz, soabort(), socket(), and sodisconnect().

Referenced by socketpair(), and soo_close().

Here is the call graph for this function:



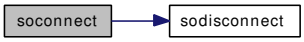
9.147.2.20 int soconnect (struct socket * *so*, struct sockaddr * *nam*, struct thread * *td*)

Definition at line 761 of file uipc_socket.c.

References sodisconnect().

Referenced by kern_connect().

Here is the call graph for this function:



9.147.2.21 int soconnect2 (struct socket * *so1*, struct socket * *so2*)

Definition at line 792 of file uipc_socket.c.

Referenced by socketpair().

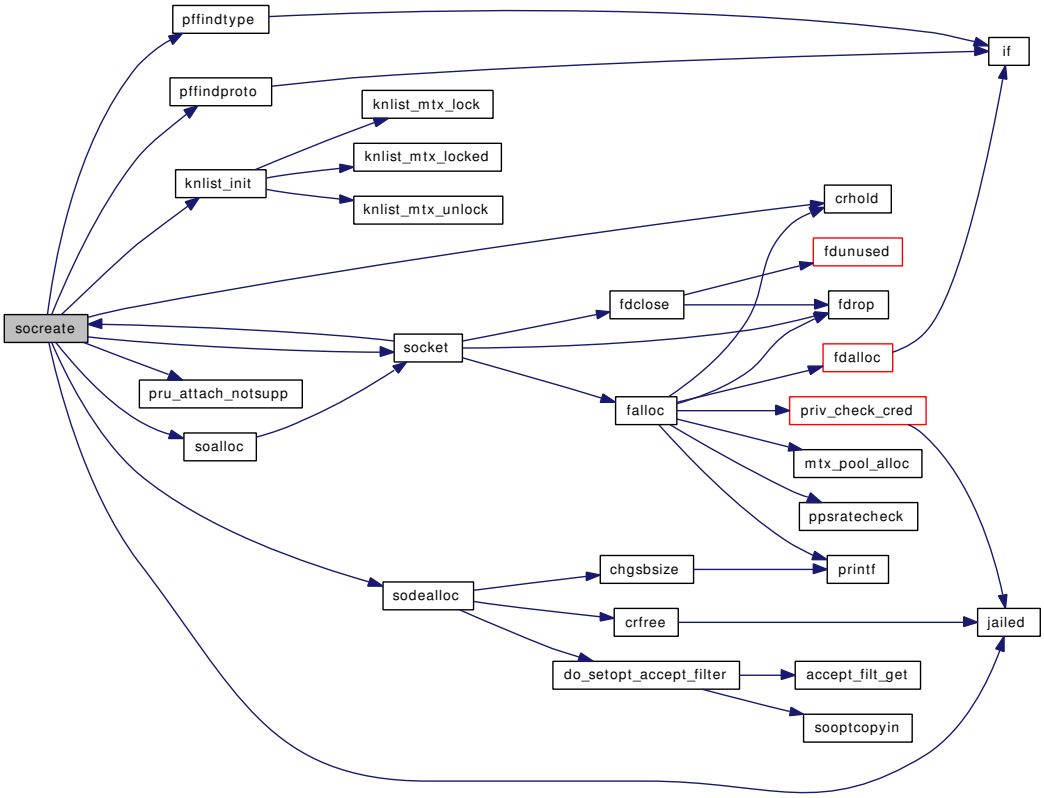
9.147.2.22 int socreate (int *dom*, struct socket ** *aso*, int *type*, int *proto*, struct ucred * *cred*, struct thread * *td*)

Definition at line 324 of file uipc_socket.c.

References crhold(), jail_socket_unixiproute_only, jailed(), knlist_init(), pffindproto(), pffindtype(), pru_attach_notsupp(), soalloc(), socket(), and sodealloc().

Referenced by socket(), and socketpair().

Here is the call graph for this function:



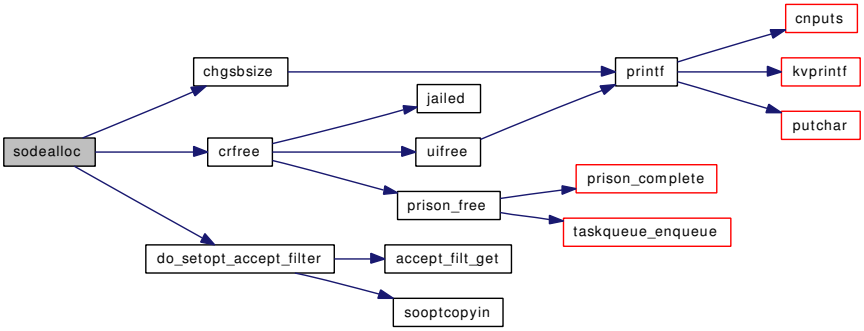
9.147.2.23 static void sodealloc (struct socket * *so*) [static]

Definition at line 289 of file uipc_socket.c.

References chgsbsize(), crfree(), do_setopt_accept_filter(), so_gencnt, and socket_zone.

Referenced by socreate(), sofree(), and sonewconn().

Here is the call graph for this function:



9.147.2.24 int sodisconnect (struct socket * *so*)

Definition at line 801 of file uipc_socket.c.

Referenced by soclose(), and soconnect().

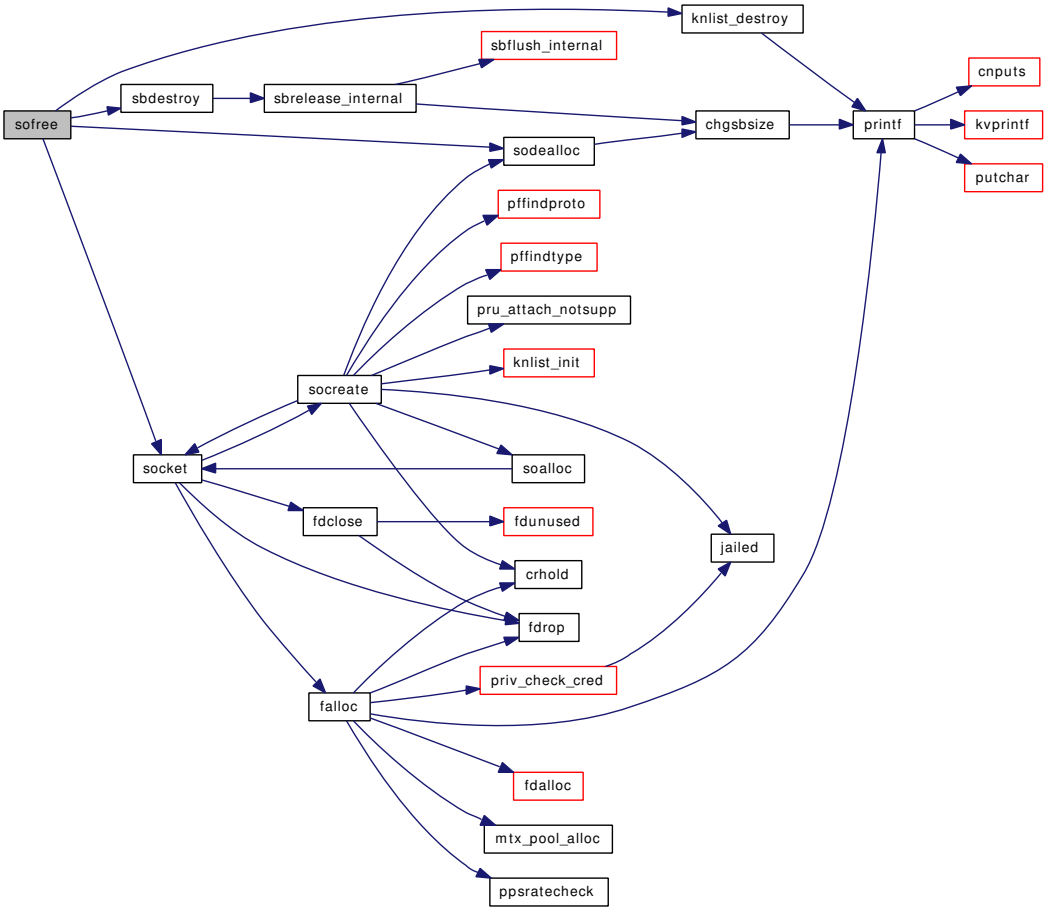
9.147.2.25 void sofree (struct socket * *so*)

Definition at line 568 of file uipc_socket.c.

References knlist_destroy(), pr, sbdestroy(), socket(), and sodealloc().

Referenced by soabort().

Here is the call graph for this function:



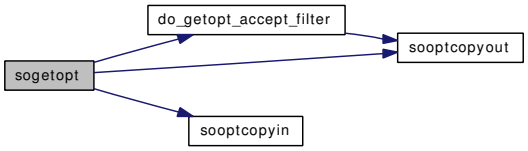
9.147.2.26 int sogetopt (struct socket * *so*, struct sockopt * *sopt*)

Definition at line 2250 of file uipc_socket.c.

References do_getopt_accept_filter(), hz, sooptcopyin(), sooptcopyout(), and tick.

Referenced by kern_getsockopt().

Here is the call graph for this function:

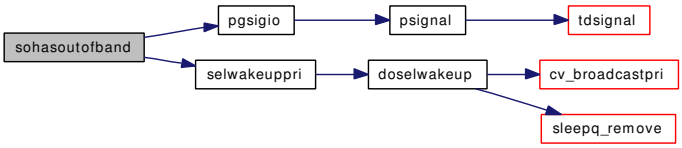


9.147.2.27 void sohasoutofband (struct socket * *so*)

Definition at line 2516 of file uipc_socket.c.

References pgsigio(), and selwakeuppri().

Here is the call graph for this function:



9.147.2.28 int solisten (struct socket * *so*, int *backlog*, struct thread * *td*)

Definition at line 511 of file uipc_socket.c.

Referenced by listen().

9.147.2.29 void solisten_proto (struct socket * *so*, int *backlog*)

Definition at line 534 of file uipc_socket.c.

References somaxconn.

Referenced by unp_listen().

9.147.2.30 int solisten_proto_check (struct socket * *so*)

Definition at line 521 of file uipc_socket.c.

Referenced by unp_listen().

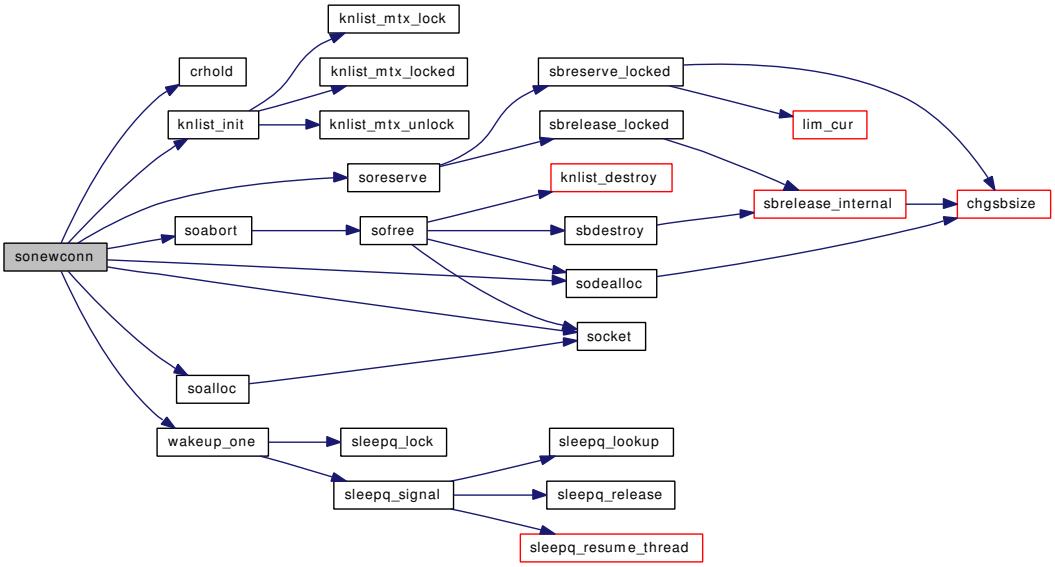
9.147.2.31 struct socket* sonewconn (struct socket * *head*, int *connstatus*)

Definition at line 403 of file uipc_socket.c.

References crhold(), knlist_init(), soabort(), soalloc(), socket(), sodealloc(), soreserve(), and wakeup_one().

Referenced by sctp_peeloff(), and unp_connect().

Here is the call graph for this function:

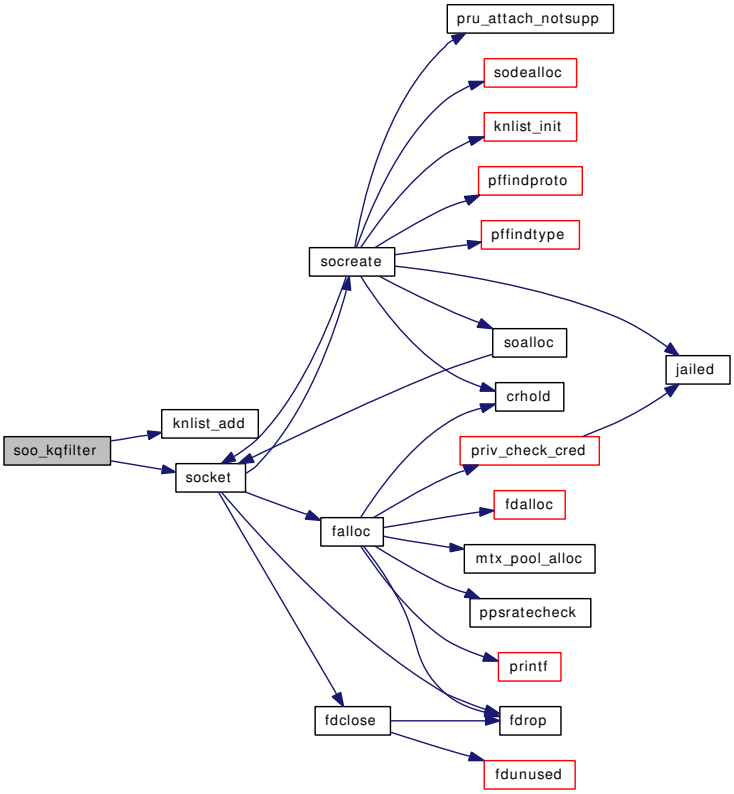


9.147.2.32 int soo_kqfilter (struct file * *fp*, struct knote * *kn*)

Definition at line 2582 of file uipc_socket.c.

References knlist_add(), socket(), solisten_filtops, soread_filtops, and sowrite_filtops.

Here is the call graph for this function:



9.147.2.33 int soopt_getm (struct sockopt * *sopt*, struct mbuf ** *mp*)

Definition at line 2400 of file uipc_socket.c.

References m_freem().

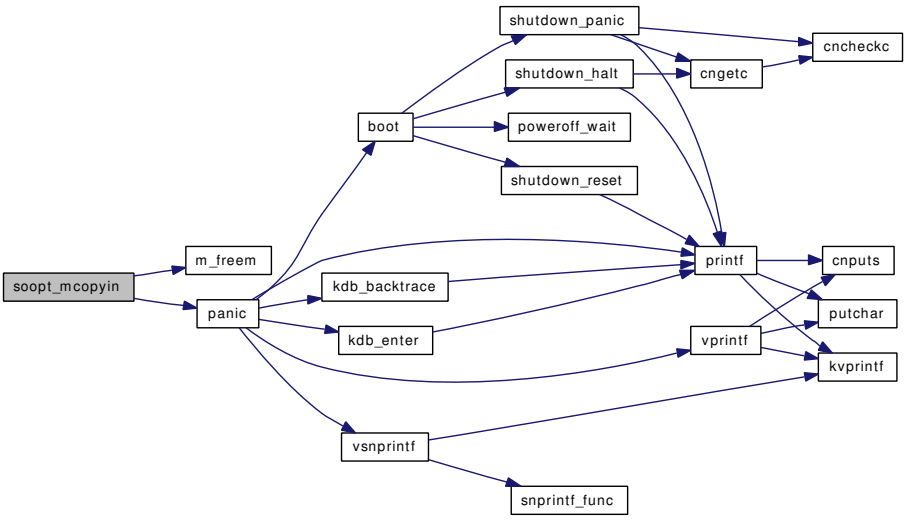
Here is the call graph for this function:

9.147.2.34 int soopt_mcopyin (struct sockopt * *sopt*, struct mbuf * *m*)

Definition at line 2449 of file uipc_socket.c.

References m_freem(), and panic().

Here is the call graph for this function:

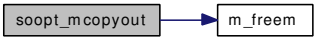


9.147.2.35 int soopt_mcopyout (struct sockopt * *sopt*, struct mbuf * *m*)

Definition at line 2478 of file uipc_socket.c.

References m_freem().

Here is the call graph for this function:



9.147.2.36 int sooptcopyin (struct sockopt * *sopt*, void * *buf*, size_t *len*, size_t *minlen*)

Definition at line 1986 of file uipc_socket.c.

Referenced by do_setopt_accept_filter(), sogetopt(), sosetopt(), and uipc_ctloutput().

9.147.2.37 int sooptcopyout (struct sockopt * *sopt*, const void * *buf*, size_t *len*)

Definition at line 2223 of file uipc_socket.c.

Referenced by do_getopt_accept_filter(), sogetopt(), and uipc_ctloutput().

9.147.2.38 int sopoll (struct socket * *so*, int *events*, struct ucred * *active_cred*, struct thread * *td*)

Definition at line 2525 of file uipc_socket.c.

Referenced by soo_poll().

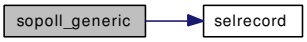
9.147.2.39 int sopoll_generic (struct socket * *so*, int *events*, struct ucred * *active_cred*, struct thread * *td*)

Definition at line 2538 of file uipc_socket.c.

References selrecord().

Referenced by protosw_init().

Here is the call graph for this function:



9.147.2.40 int soreceive (struct socket * *so*, struct sockaddr ** *psa*, struct uio * *uio*, struct mbuf ** *mp0*, struct mbuf ** *controlp*, int * *flagsp*)

Definition at line 1899 of file uipc_socket.c.

Referenced by soo_read().

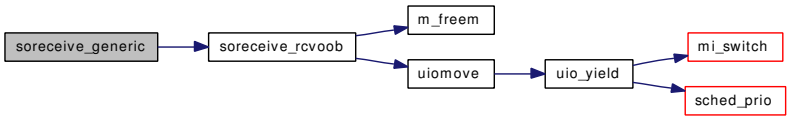
9.147.2.41 int soreceive_generic (struct socket * *so*, struct sockaddr ** *psa*, struct uio * *uio*, struct mbuf ** *mp0*, struct mbuf ** *controlp*, int * *flagsp*)

Definition at line 1446 of file uipc_socket.c.

References pr, SBLOCKWAIT, and soreceive_rcvoob().

Referenced by protosw_init().

Here is the call graph for this function:



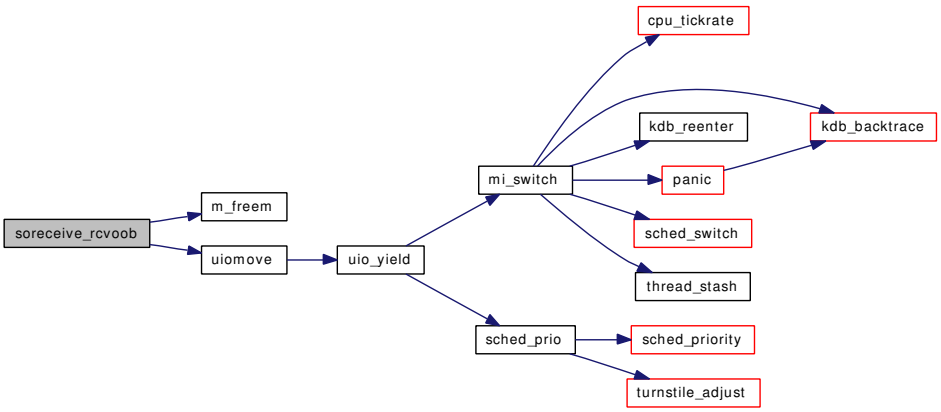
9.147.2.42 static int soreceive_rcvoob (struct socket * *so*, struct uio * *uio*, int *flags*) [static]

Definition at line 1350 of file uipc_socket.c.

References m_freem(), pr, and uiomove().

Referenced by soreceive_generic().

Here is the call graph for this function:



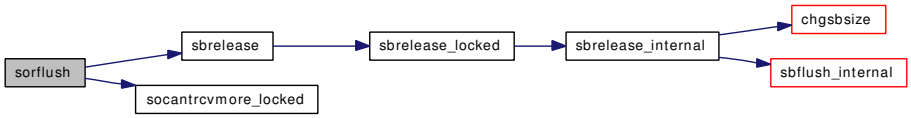
9.147.2.43 void sorflush (struct socket * *so*)

Definition at line 1934 of file uipc_socket.c.

References pr, sbrelease(), and socantrcvmore_locked().

Referenced by soshutdown(), and unp_gc().

Here is the call graph for this function:



9.147.2.44 int sosend (struct socket * *so*, struct sockaddr * *addr*, struct uio * *uio*, struct mbuf * *top*, struct mbuf * *control*, int *flags*, struct thread * *td*)

Definition at line 1323 of file uipc_socket.c.

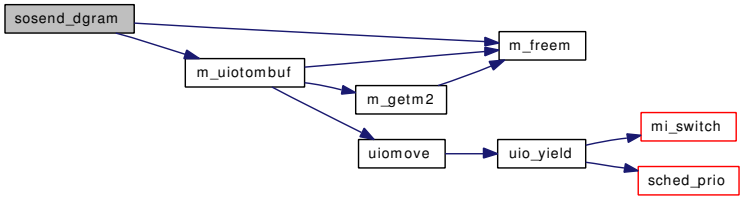
Referenced by soo_write().

9.147.2.45 int sosend_dgram (struct socket * *so*, struct sockaddr * *addr*, struct uio * *uio*, struct mbuf * *top*, struct mbuf * *control*, int *flags*, struct thread * *td*)

Definition at line 960 of file uipc_socket.c.

References m_freem(), m_uiotombuf(), and max_hdr.

Here is the call graph for this function:



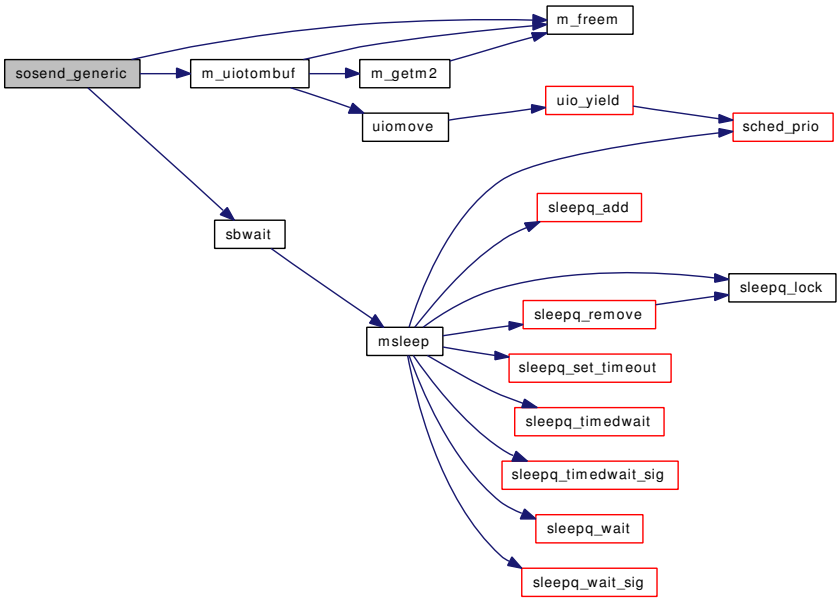
9.147.2.46 int sosend_generic (struct socket * *so*, struct sockaddr * *addr*, struct uio * *uio*, struct mbuf * *top*, struct mbuf * *control*, int *flags*, struct thread * *td*)

Definition at line 1141 of file uipc_socket.c.

References m_freem(), m_uiotombuf(), max_hdr, SBLOCKWAIT, sbwait(), and snderr.

Referenced by protosw_init().

Here is the call graph for this function:



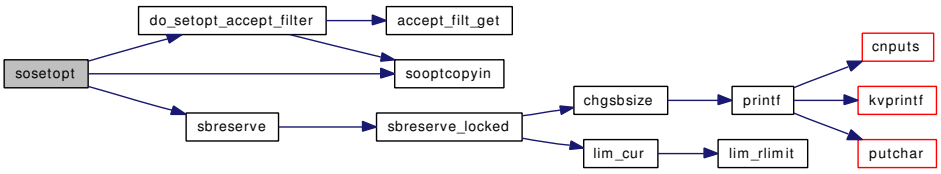
9.147.2.47 int sosetopt (struct socket * *so*, struct sockopt * *sopt*)

Definition at line 2033 of file uipc_socket.c.

References do_setopt_accept_filter(), hz, sbreserve(), sooptcopyin(), and tick.

Referenced by kern_setsockopt(), and so_setsockopt().

Here is the call graph for this function:



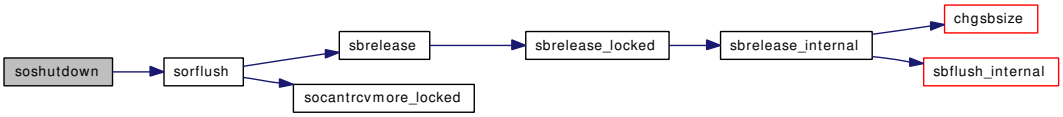
9.147.2.48 int soshutdown (struct socket * *so*, int *how*)

Definition at line 1917 of file uipc_socket.c.

References pr, and sorflush().

Referenced by shutdown().

Here is the call graph for this function:



9.147.2.49 SYSCTL_INT (_kern_ipc, OID_AUTO, numopensockets, CTLFLAG_RD, & *numopensockets*, 0, "Number of open sockets")

9.147.2.50 static int sysctl_maxsockets (SYSCTL_HANDLER_ARGS) [static]

Definition at line 211 of file uipc_socket.c.

References maxfiles, maxfilesperproc, maxsockets, and sysctl_handle_int().

Here is the call graph for this function:

9.147.2.51 SYSCTL_NODE (_kern, KERN_IPC, ipc, CTLFLAG_RW, 0, "IPC")

9.147.2.52 SYSCTL_PROC (_kern_ipc, OID_AUTO, maxsockets, CTLTYPE_INT| *CTLFLAG_RW*, & *maxsockets*, 0, sysctl_maxsockets, "IU", "Maximum number of sockets avaliable")

9.147.2.53 SYSCTL_PROC (_kern_ipc, KIPC_SOMAXCONN, somaxconn, CTLTYPE_UINT| *CTLFLAG_RW*, 0, sizeof(int), sysctl_somaxconn, "I", "Maximum pending socket connection ""queue size")

9.147.2.54 static int sysctl_somaxconn (SYSCTL_HANDLER_ARGS) [static]

Definition at line 2702 of file uipc_socket.c.

References somaxconn, and sysctl_handle_int().

Here is the call graph for this function:

9.147.2.55 SYSINIT (param, SI_SUB_TUNABLES, SI_ORDER_ANY, init_maxsockets, NULL)

9.147.3 Variable Documentation

9.147.3.1 struct mtx accept_mtx

Definition at line 190 of file uipc_socket.c.

Referenced by kern_accept().

9.147.3.2 int maxsockets

Definition at line 160 of file uipc_socket.c.

Referenced by domaininit(), init_maxsockets(), socket_zone_change(), sysctl_maxsockets(), unp_init(), and unp_zone_change().

9.147.3.3 int numopensockets [static]

Definition at line 171 of file uipc_socket.c.

9.147.3.4 so_gen_t so_gencnt

Definition at line 158 of file uipc_socket.c.

Referenced by soalloc(), sodealloc(), and unp_pcblist().

9.147.3.5 struct mtx so_global_mtx [static]

Definition at line 197 of file uipc_socket.c.

9.147.3.6 uma_zone_t socket_zone

Definition at line 157 of file uipc_socket.c.

Referenced by domaininit(), soalloc(), socket_zone_change(), and sodealloc().

9.147.3.7 struct filterops solisten_filtops [static]

Initial value:

```
{ 1, NULL, filt_sordetach, filt_solisten }
```

Definition at line 150 of file uipc_socket.c.

Referenced by soo_kqfilter().

9.147.3.8 int somaxconn = SOMAXCONN [static]

Definition at line 165 of file uipc_socket.c.

Referenced by solisten_proto(), and sysctl_somaxconn().

9.147.3.9 struct filterops soread_filtops [static]

Initial value:

```
{ 1, NULL, filt_sordetach, filt_soread }
```

Definition at line 152 of file uipc_socket.c.

Referenced by soo_kqfilter().

9.147.3.10 struct filterops sowrite_filtops [static]

Initial value:

```
{ 1, NULL, filt_sowdetach, filt_sowrite }
```

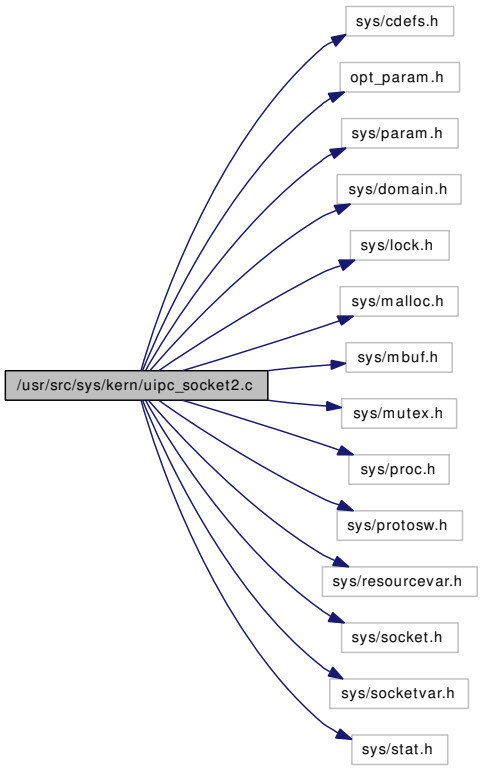
Definition at line 154 of file uipc_socket.c.

Referenced by soo_kqfilter().

9.148 /usr/src/sys/kern/uipc_socket2.c File Reference

```
#include <sys/cdefs.h>
#include "opt_param.h"
#include <sys/param.h>
#include <sys/domain.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mbuf.h>
#include <sys/mutex.h>
#include <sys/proc.h>
#include <sys/protosw.h>
#include <sys/resourcevar.h>
#include <sys/socket.h>
#include <sys/socketvar.h>
#include <sys/stat.h>
```

Include dependency graph for uipc_socket2.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/uipc_socket2.c,v 1.162 2006/08/02 16:23:52 rwatson Exp $")
- void soisconnecting (struct socket *so)
- void soisconnected (struct socket *so)
- void soisdisconnecting (struct socket *so)
- void soisdisconnected (struct socket *so)
- mbuf * sbcreatecontrol (caddr_t p, int size, int type, int level)
- int pru_accept_notsupp (struct socket *so, struct sockaddr **nam)
- int pru_attach_notsupp (struct socket *so, int proto, struct thread *td)
- int pru_bind_notsupp (struct socket *so, struct sockaddr *nam, struct thread *td)
- int pru_connect_notsupp (struct socket *so, struct sockaddr *nam, struct thread *td)
- int pru_connect2_notsupp (struct socket *so1, struct socket *so2)
- int pru_control_notsupp (struct socket *so, u_long cmd, caddr_t data, struct ifnet *ifp, struct thread *td)
- int pru_disconnect_notsupp (struct socket *so)
- int pru_listen_notsupp (struct socket *so, int backlog, struct thread *td)
- int pru_peeraddr_notsupp (struct socket *so, struct sockaddr **nam)
- int pru_rcvd_notsupp (struct socket *so, int flags)
- int pru_rcvoob_notsupp (struct socket *so, struct mbuf *m, int flags)
- int pru_send_notsupp (struct socket *so, int flags, struct mbuf *m, struct sockaddr *addr, struct mbuf *control, struct thread *td)
- int pru_sense_null (struct socket *so, struct stat *sb)
- int pru_shutdown_notsupp (struct socket *so)
- int pru_sockaddr_notsupp (struct socket *so, struct sockaddr **nam)
- int pru_sosend_notsupp (struct socket *so, struct sockaddr *addr, struct uio *uio, struct mbuf *top, struct mbuf *control, int flags, struct thread *td)
- int pru_soreceive_notsupp (struct socket *so, struct sockaddr **paddr, struct uio *uio, struct mbuf **mp0, struct mbuf **controlp, int *flagsp)
- int pru_sopoll_notsupp (struct socket *so, int events, struct ucred *cred, struct thread *td)
- sockaddr * sodupsockaddr (const struct sockaddr *sa, int mflags)
- void sotoxsocket (struct socket *so, struct xsocket *xso)
- void sbtoxsockbuf (struct sockbuf *sb, struct xsockbuf *xsb)

9.148.1 Function Documentation

9.148.1.1 __FBSDID ("$FreeBSD: src/sys/kern/uipc_socket2. *c*, v 1.162 2006/08/02 16:23:52 rwatson Exp $")

9.148.1.2 int pru_accept_notsupp (struct socket * *so*, struct sockaddr ** *nam*)

Definition at line 217 of file uipc_socket2.c.

Referenced by protosw_init(), and SYSINIT().

9.148.1.3 int pru_attach_notsupp (struct socket * *so*, int *proto*, struct thread * *td*)

Definition at line 223 of file uipc_socket2.c.

Referenced by socreate(), and SYSINIT().

9.148.1.4 int pru_bind_notsupp (struct socket * *so*, struct sockaddr * *nam*, struct thread * *td*)

Definition at line 229 of file uipc_socket2.c.

Referenced by SYSINIT().

9.148.1.5 int pru_connect2_notsupp (struct socket * *so1*, struct socket * *so2*)

Definition at line 241 of file uipc_socket2.c.

Referenced by protosw_init(), and SYSINIT().

9.148.1.6 int pru_connect_notsupp (struct socket * *so*, struct sockaddr * *nam*, struct thread * *td*)

Definition at line 235 of file uipc_socket2.c.

Referenced by protosw_init(), and SYSINIT().

9.148.1.7 int pru_control_notsupp (struct socket * *so*, u_long *cmd*, caddr_t *data*, struct ifnet * *ifp*, struct thread * *td*)

Definition at line 247 of file uipc_socket2.c.

Referenced by protosw_init(), and SYSINIT().

9.148.1.8 int pru_disconnect_notsupp (struct socket * *so*)

Definition at line 254 of file uipc_socket2.c.

Referenced by SYSINIT().

9.148.1.9 int pru_listen_notsupp (struct socket * *so*, int *backlog*, struct thread * *td*)

Definition at line 260 of file uipc_socket2.c.

Referenced by protosw_init(), and SYSINIT().

9.148.1.10 int pru_peeraddr_notsupp (struct socket * *so*, struct sockaddr ** *nam*)

Definition at line 266 of file uipc_socket2.c.

Referenced by SYSINIT().

9.148.1.11 int pru_rcvd_notsupp (struct socket * *so*, int *flags*)

Definition at line 272 of file uipc_socket2.c.

Referenced by protosw_init(), and SYSINIT().

9.148.1.12 int pru_rcvoob_notsupp (struct socket * *so*, struct mbuf * *m*, int *flags*)

Definition at line 278 of file uipc_socket2.c.

Referenced by protosw_init(), and SYSINIT().

9.148.1.13 int pru_send_notsupp (struct socket * *so*, int *flags*, struct mbuf * *m*, struct sockaddr * *addr*, struct mbuf * *control*, struct thread * *td*)

Definition at line 284 of file uipc_socket2.c.

Referenced by SYSINIT().

9.148.1.14 int pru_sense_null (struct socket * *so*, struct stat * *sb*)

Definition at line 295 of file uipc_socket2.c.

Referenced by protosw_init(), and SYSINIT().

9.148.1.15 int pru_shutdown_notsupp (struct socket * *so*)

Definition at line 302 of file uipc_socket2.c.

Referenced by SYSINIT().

9.148.1.16 int pru_sockaddr_notsupp (struct socket * *so*, struct sockaddr ** *nam*)

Definition at line 308 of file uipc_socket2.c.

Referenced by SYSINIT().

9.148.1.17 int pru_sopoll_notsupp (struct socket * *so*, int *events*, struct ucred * *cred*, struct thread * *td*)

Definition at line 329 of file uipc_socket2.c.

Referenced by SYSINIT().

9.148.1.18 int pru_soreceive_notsupp (struct socket * *so*, struct sockaddr ** *paddr*, struct uio * *uio*, struct mbuf ** *mp0*, struct mbuf ** *controlp*, int * *flagsp*)

Definition at line 321 of file uipc_socket2.c.

Referenced by SYSINIT().

9.148.1.19 int pru_sosend_notsupp (struct socket * *so*, struct sockaddr * *addr*, struct uio * *uio*, struct mbuf * *top*, struct mbuf * *control*, int *flags*, struct thread * *td*)

Definition at line 314 of file uipc_socket2.c.

Referenced by SYSINIT().

9.148.1.20 struct mbuf* sbcreatecontrol (caddr_t *p*, int *size*, int *type*, int *level*)

Definition at line 183 of file uipc_socket2.c.

Referenced by unp_addsockcred(), unp_externalize(), and unp_internalize().

9.148.1.21 void sbtoxsockbuf (struct sockbuf * *sb*, struct xsockbuf * *xsb*)

Definition at line 388 of file uipc_socket2.c.

Referenced by sotoxsocket().

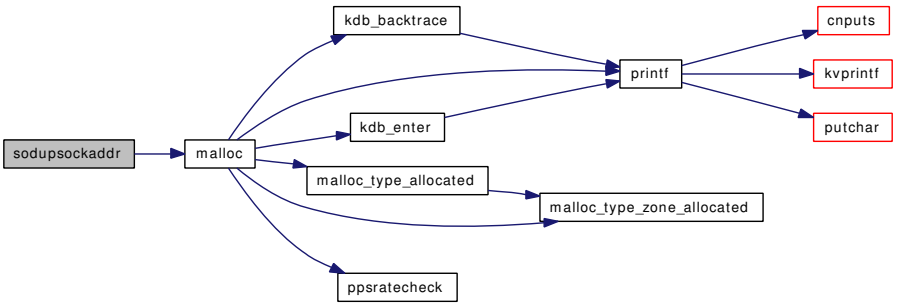
9.148.1.22 struct sockaddr* sodupsockaddr (const struct sockaddr * *sa*, int *mflags*)

Definition at line 339 of file uipc_socket2.c.

References malloc().

Referenced by uipc_bind().

Here is the call graph for this function:



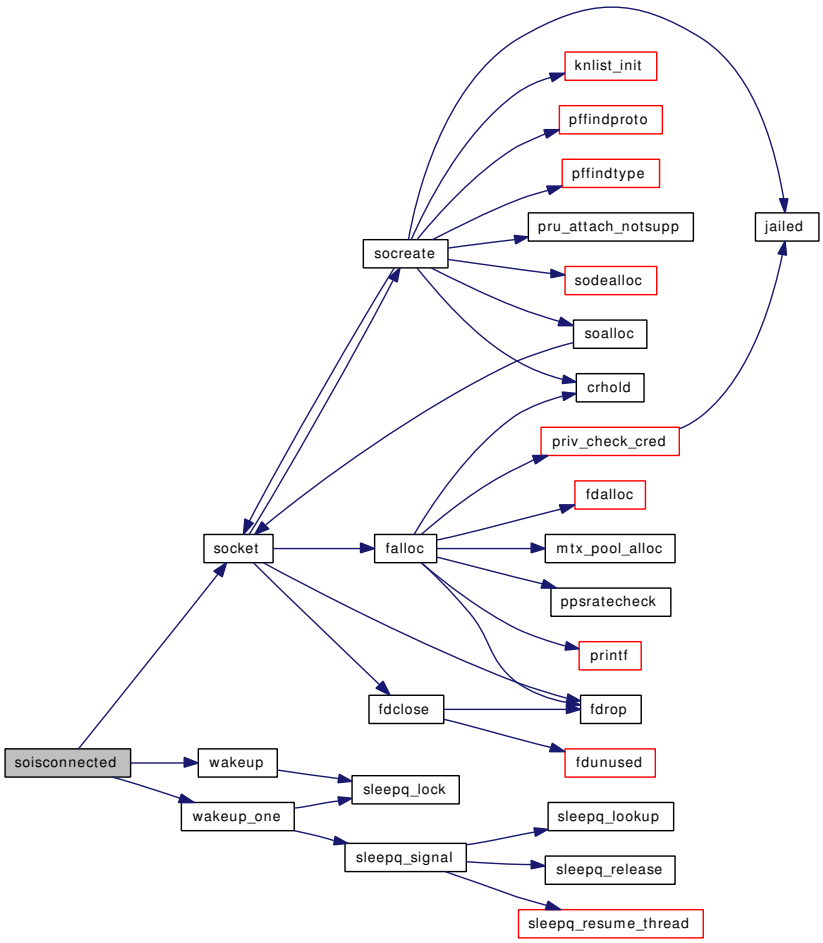
9.148.1.23 void soisconnected (struct socket * *so*)

Definition at line 96 of file uipc_socket2.c.

References socket(), wakeup(), and wakeup_one().

Referenced by unp_connect2().

Here is the call graph for this function:



9.148.1.24 void soisconnecting (struct socket * *so*)

Definition at line 85 of file uipc_socket2.c.

Referenced by unp_connect2().

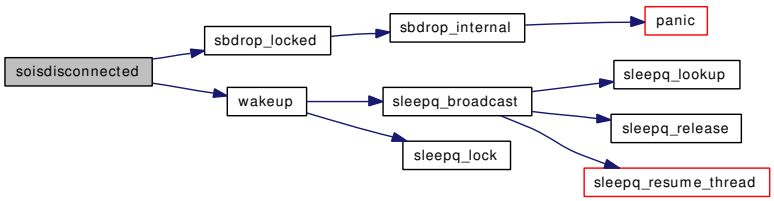
9.148.1.25 void soisdisconnected (struct socket * *so*)

Definition at line 158 of file uipc_socket2.c.

References sbdrop_locked(), and wakeup().

Referenced by unp_disconnect().

Here is the call graph for this function:

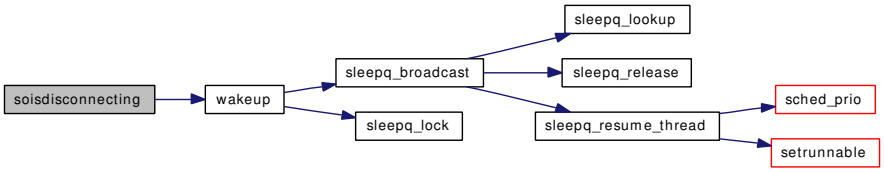


9.148.1.26 void soisdisconnecting (struct socket * *so*)

Definition at line 138 of file uipc_socket2.c.

References wakeup().

Here is the call graph for this function:



9.148.1.27 void sotoxsocket (struct socket * *so*, struct xsocket * *xso*)

Definition at line 358 of file uipc_socket2.c.

References sbtoxsockbuf().

Referenced by unp_pcblist().

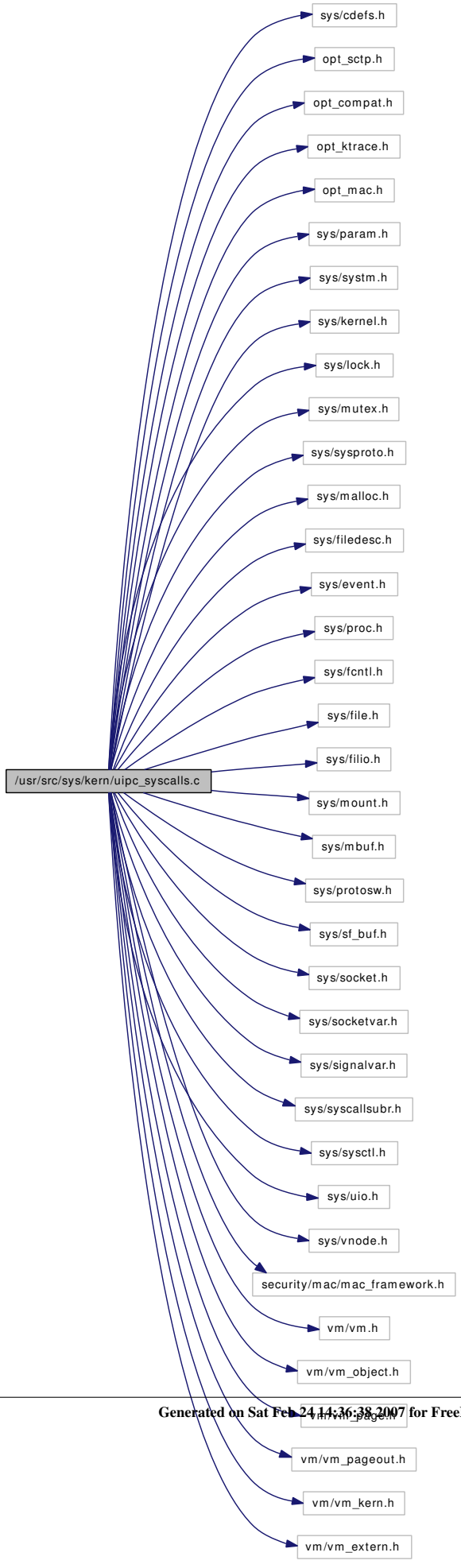
Here is the call graph for this function:

9.149 /usr/src/sys/kern/uipc_syscalls.c File Reference

```
#include <sys/cdefs.h>
#include "opt_sctp.h"
#include "opt_compat.h"
#include "opt_ktrace.h"
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/sysproto.h>
#include <sys/malloc.h>
#include <sys/filedesc.h>
#include <sys/event.h>
#include <sys/proc.h>
#include <sys/fcntl.h>
#include <sys/file.h>
#include <sys/filio.h>
#include <sys/mount.h>
#include <sys/mbuf.h>
#include <sys/protosw.h>
#include <sys/sf_buf.h>
#include <sys/socket.h>
#include <sys/socketvar.h>
#include <sys/signalvar.h>
#include <sys/syscallsubr.h>
#include <sys/sysctl.h>
#include <sys/uio.h>
#include <sys/vnode.h>
#include <security/mac/mac_framework.h>
#include <vm/vm.h>
#include <vm/vm_object.h>
#include <vm/vm_page.h>
#include <vm/vm_pageout.h>
#include <vm/vm_kern.h>
```

```
#include <vm/vm_extern.h>
```

Include dependency graph for uipc_syscalls.c:


/usr/src/sys/kern/uipc_syscalls.c
sys/cdefs.h
opt_sctp.h
opt_compat.h
opt_ktrace.h
opt_mac.h
sys/param.h
sys/systm.h
sys/kernel.h
sys/lock.h
sys/mutex.h
sys/sysproto.h
sys/malloc.h
sys/filedesc.h
sys/event.h
sys/proc.h
sys/fcntl.h
sys/file.h
sys/filio.h
sys/mount.h
sys/mbuf.h
sys/protosw.h
sys/sf_buf.h
sys/socket.h
sys/socketvar.h
sys/signalvar.h
sys/syscallsubr.h
sys/sysctl.h
sys/uio.h
sys/vnode.h
security/mac/mac_framework.h
vm/vm.h
vm/vm_object.h
vm/vm_page.h
vm/vm_pageout.h
vm/vm_kern.h
vm/vm_extern.h

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/uipc_syscalls.c,v 1.246 2007/01/24 12:59:56 rrs Exp $")
- static int sendit (struct thread *td, int s, struct msghdr *mp, int flags)
- static int recvit (struct thread *td, int s, struct msghdr *mp, void *namelenp)
- static int accept1 (struct thread *td, struct accept_args *uap, int compat)
- static int do_sendfile (struct thread *td, struct sendfile_args *uap, int compat)
- static int getsockname1 (struct thread *td, struct getsockname_args *uap, int compat)
- static int getpeername1 (struct thread *td, struct getpeername_args *uap, int compat)
- SYSCTL_INT (_kern_ipc, OID_AUTO, nsfbufs, CTLFLAG_RDTUN,&nsfbufs, 0,"Maximum number of sendfile(2) sf_bufs available")
- SYSCTL_INT (_kern_ipc, OID_AUTO, nsfbufspeak, CTLFLAG_RD,&nsfbufspeak, 0,"Number of sendfile(2) sf_bufs at peak usage")
- SYSCTL_INT (_kern_ipc, OID_AUTO, nsfbufsused, CTLFLAG_RD,&nsfbufsused, 0,"Number of sendfile(2) sf_bufs in use")
- static int getsock (struct filedesc *fdp, int fd, struct file **fpp, u_int *fflagp)
- int socket (struct thread *td, struct socket_args *uap)
- int bind (struct thread *td, struct bind_args *uap)
- int kern_bind (struct thread *td, int fd, struct sockaddr *sa)
- int listen (struct thread *td, struct listen_args *uap)
- int kern_accept (struct thread *td, int s, struct sockaddr **name, socklen_t *namelen, struct file **fp)
- int accept (struct thread *td, struct accept_args *uap)
- int connect (struct thread *td, struct connect_args *uap)
- int kern_connect (struct thread *td, int fd, struct sockaddr *sa)
- int socketpair (struct thread *td, struct socketpair_args *uap)
- int kern_sendit (struct thread *td, int s, struct msghdr *mp, int flags, struct mbuf *control, enum uio_seg segflg)
- int sendto (struct thread *td, struct sendto_args *uap)
- int sendmsg (struct thread *td, struct sendmsg_args *uap)
- int kern_recvit (struct thread *td, int s, struct msghdr *mp, enum uio_seg fromseg, struct mbuf **controlp)
- int recvfrom (struct thread *td, struct recvfrom_args *uap)
- int recvmsg (struct thread *td, struct recvmsg_args *uap)
- int shutdown (struct thread *td, struct shutdown_args *uap)
- int setsockopt (struct thread *td, struct setsockopt_args *uap)
- int kern_setsockopt (struct thread *td, int s, int level, int name, void *val, enum uio_seg valseg, socklen_t valsize)
- int getsockopt (struct thread *td, struct getsockopt_args *uap)
- int kern_getsockopt (struct thread *td, int s, int level, int name, void *val, enum uio_seg valseg, socklen_t *valsize)
- int kern_getsockname (struct thread *td, int fd, struct sockaddr **sa, socklen_t *alen)
- int getsockname (struct thread *td, struct getsockname_args *uap)
- int kern_getpeername (struct thread *td, int fd, struct sockaddr **sa, socklen_t *alen)
- int getpeername (struct thread *td, struct getpeername_args *uap)
- int sockargs (struct mbuf **mp, caddr_t buf, int buflen, int type)
- int getsockaddr (struct sockaddr **namp, caddr_t uaddr, size_t len)
- void sf_buf_mext (void *addr, void *args)
- int sendfile (struct thread *td, struct sendfile_args *uap)
- int kern_sendfile (struct thread *td, struct sendfile_args *uap, struct uio *hdr_uio, struct uio *trl_uio, int compat)
- int sctp_peeloff (struct thread *td, struct sctp_peeloff_args *uap)
- int sctp_generic_sendmsg (struct thread *td, struct sctp_generic_sendmsg_args *uap)
- int sctp_generic_sendmsg_iov (struct thread *td, struct sctp_generic_sendmsg_iov_args *uap)
- int sctp_generic_recvmsg (struct thread *td, struct sctp_generic_recvmsg_args *uap)

Variables

- int nsfbufs
- int nsfbufspeak
- int nsfbufsused

9.149.1 Function Documentation

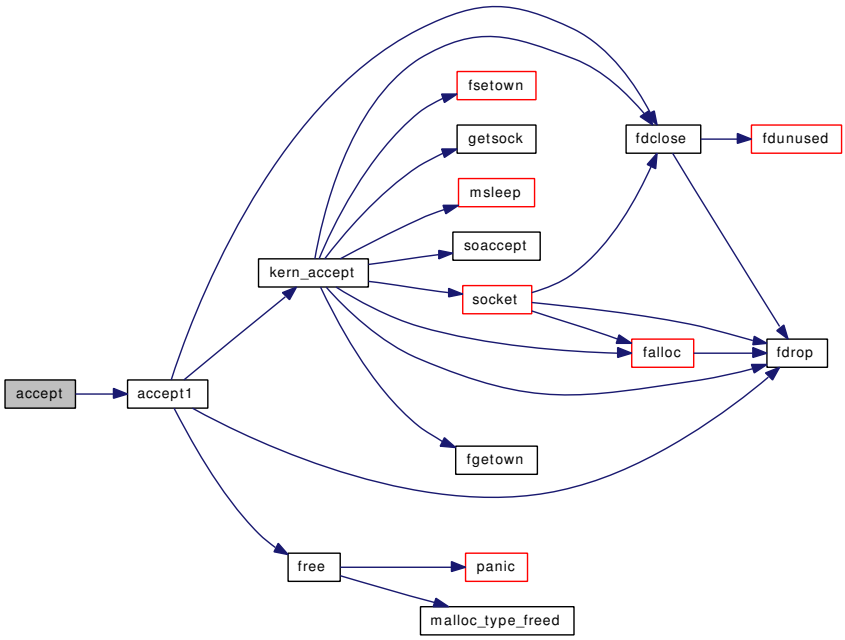
9.149.1.1 __FBSDID ("$FreeBSD: src/sys/kern/uipc_syscalls. *c*, v 1.246 2007/01/24 12:59:56 rrs Exp $")

9.149.1.2 int accept (struct thread * *td*, struct accept_args * *uap*)

Definition at line 513 of file uipc_syscalls.c.

References accept1().

Here is the call graph for this function:



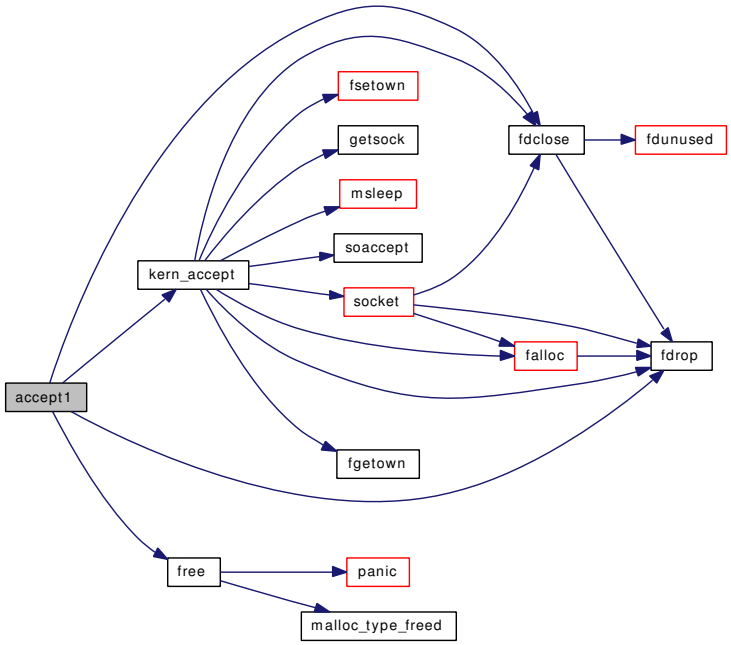
9.149.1.3 static int accept1 (struct thread * *td*, struct accept_args * *uap*, int *compat*) [static]

Definition at line 298 of file uipc_syscalls.c.

References fdclose(), fdrop(), free(), and kern_accept().

Referenced by accept().

Here is the call graph for this function:

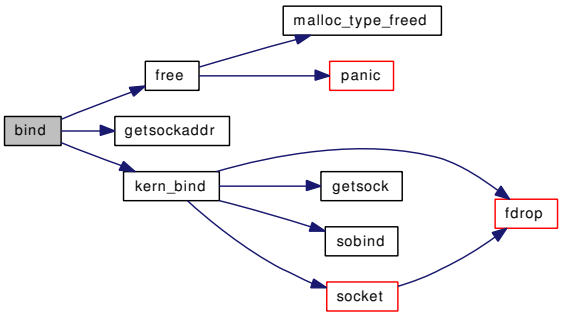


9.149.1.4 int bind (struct thread * *td*, struct bind_args * *uap*)

Definition at line 205 of file uipc_syscalls.c.

References free(), getsockaddr(), and kern_bind().

Here is the call graph for this function:

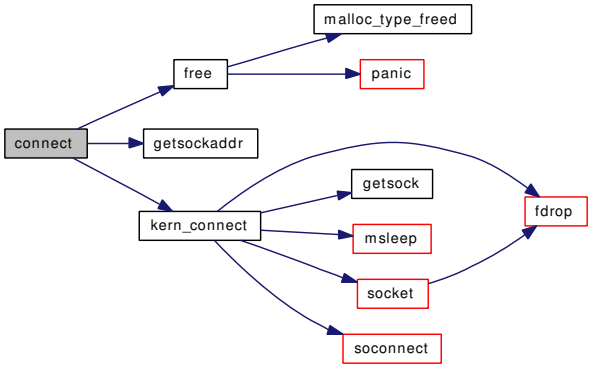


9.149.1.5 int connect (struct thread * *td*, struct connect_args * *uap*)

Definition at line 540 of file uipc_syscalls.c.

References free(), getsockaddr(), and kern_connect().

Here is the call graph for this function:



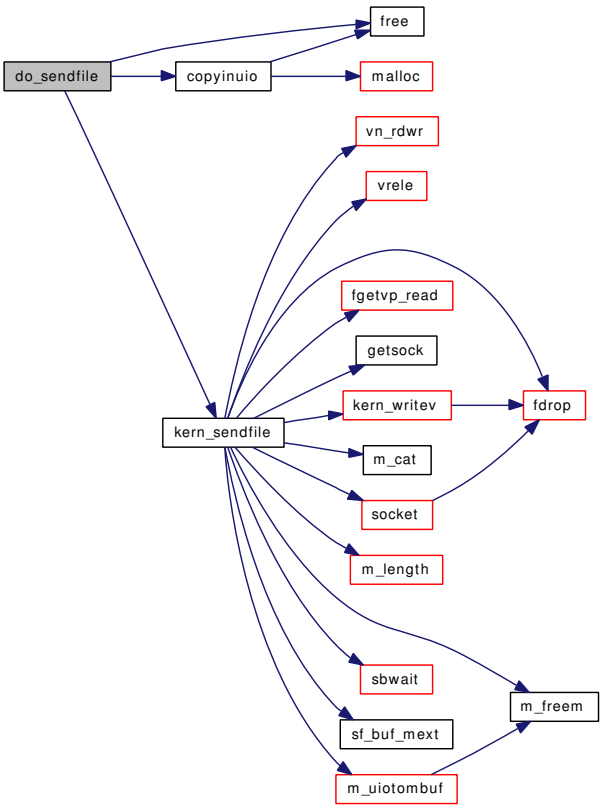
9.149.1.6 static int do_sendfile (struct thread * *td*, struct sendfile_args * *uap*, int *compat*) [static]

Definition at line 1831 of file uipc_syscalls.c.

References copyinuio(), free(), and kern_sendfile().

Referenced by sendfile().

Here is the call graph for this function:

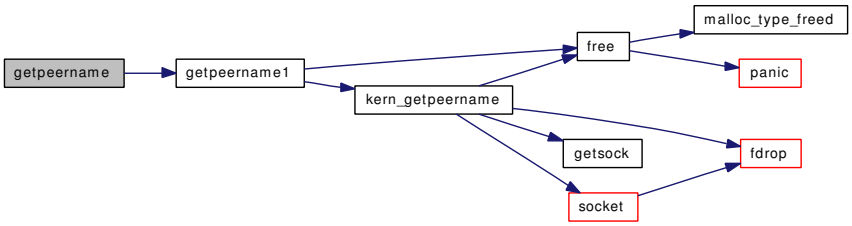


9.149.1.7 int getpeername (struct thread * *td*, struct getpeername_args * *uap*)

Definition at line 1688 of file uipc_syscalls.c.

References getpeername1().

Here is the call graph for this function:



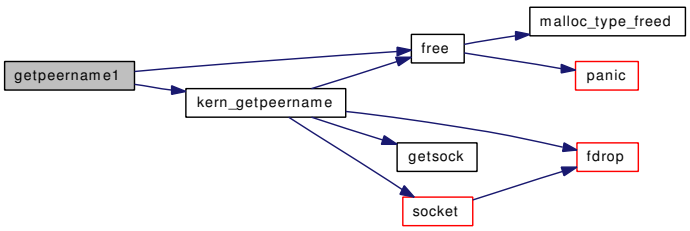
9.149.1.8 static int getpeername1 (struct thread * *td*, struct getpeername_args * *uap*, int *compat*) [static]

Definition at line 1608 of file uipc_syscalls.c.

References free(), and kern_getpeername().

Referenced by getpeername().

Here is the call graph for this function:



9.149.1.9 static int getsock (struct filedesc * *fdp*, int *fd*, struct file ** *fpp*, u_int * *fflagp*) [static]

Definition at line 118 of file uipc_syscalls.c.

Referenced by kern_accept(), kern_bind(), kern_connect(), kern_getpeername(), kern_getsockname(), kern_getsockopt(), kern_recvit(), kern_sendfile(), kern_sendit(), kern_setsockopt(), listen(), sctp_generic_recvmsg(), sctp_generic_sendmsg(), sctp_generic_sendmsg_iov(), and shutdown().

9.149.1.10 int getsockaddr (struct sockaddr ** *namp*, caddr_t *uaddr*, size_t *len*)

Definition at line 1760 of file uipc_syscalls.c.

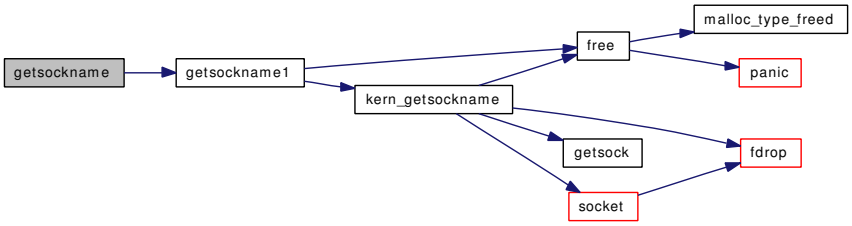
Referenced by bind(), connect(), sctp_generic_sendmsg(), sctp_generic_sendmsg_iov(), and sendit().

9.149.1.11 int getsockname (struct thread * *td*, struct getsockname_args * *uap*)

Definition at line 1579 of file uipc_syscalls.c.

References getsockname1().

Here is the call graph for this function:



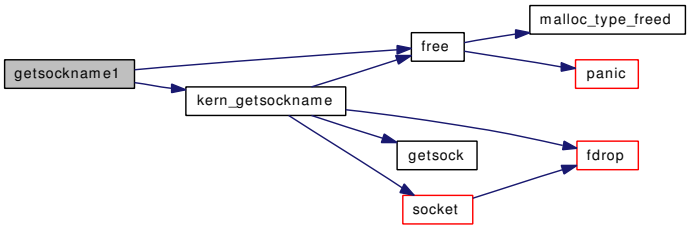
9.149.1.12 static int getsockname1 (struct thread * *td*, struct getsockname_args * *uap*, int *compat*) [static]

Definition at line 1504 of file uipc_syscalls.c.

References free(), and kern_getsockname().

Referenced by getsockname().

Here is the call graph for this function:

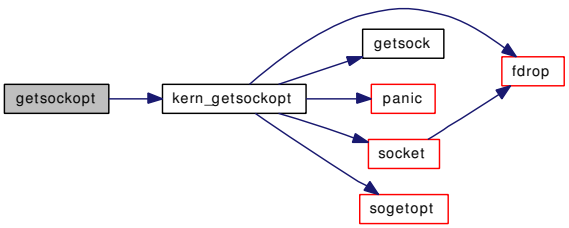


9.149.1.13 int getsockopt (struct thread * *td*, struct getsockopt_args * *uap*)

Definition at line 1418 of file uipc_syscalls.c.

References kern_getsockopt().

Here is the call graph for this function:



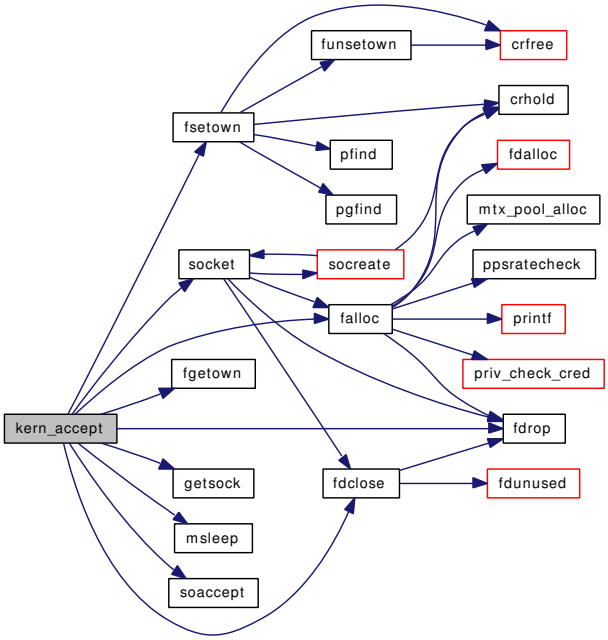
9.149.1.14 int kern_accept (struct thread * *td*, int *s*, struct sockaddr ** *name*, socklen_t * *namelen*, struct file ** *fp*)

Definition at line 350 of file uipc_syscalls.c.

References accept_mtx, falloc(), fdclose(), fdrop(), fgetown(), fsetown(), getsock(), msleep(), soaccept(), socket(), and socketops.

Referenced by accept1().

Here is the call graph for this function:



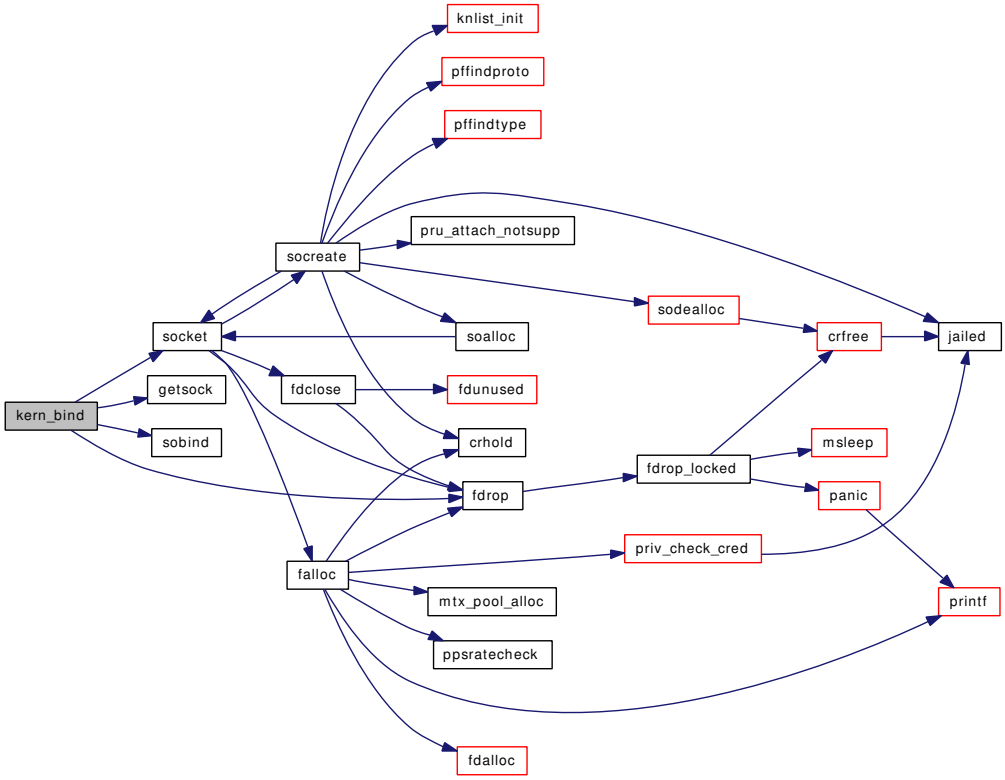
9.149.1.15 int kern_bind (struct thread * *td*, int *fd*, struct sockaddr * *sa*)

Definition at line 225 of file uipc_syscalls.c.

References fdrop(), getsock(), sobind(), and socket().

Referenced by bind().

Here is the call graph for this function:



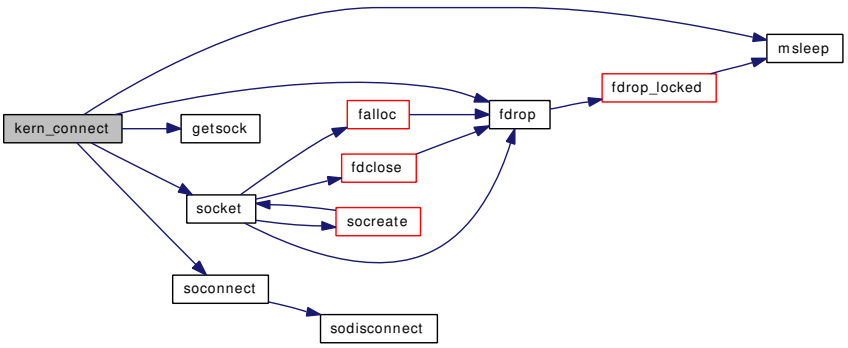
9.149.1.16 int kern_connect (struct thread * *td*, int *fd*, struct sockaddr * *sa*)

Definition at line 562 of file uipc_syscalls.c.

References fdrop(), getsock(), msleep(), socket(), and soconnect().

Referenced by connect().

Here is the call graph for this function:



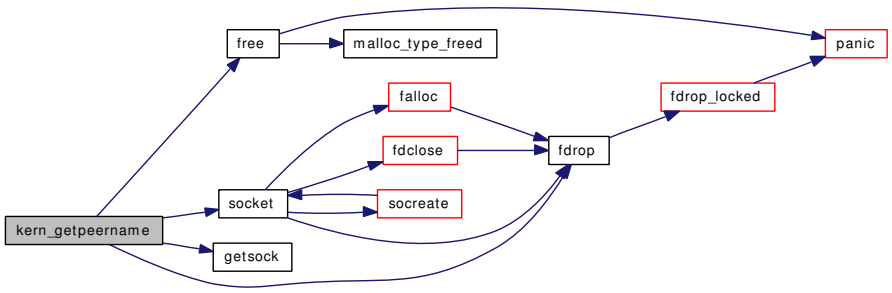
9.149.1.17 int kern_getpeername (struct thread * *td*, int *fd*, struct sockaddr ** *sa*, socklen_t * *alen*)

Definition at line 1643 of file uipc_syscalls.c.

References fdrop(), free(), getsock(), and socket().

Referenced by getpeername1().

Here is the call graph for this function:



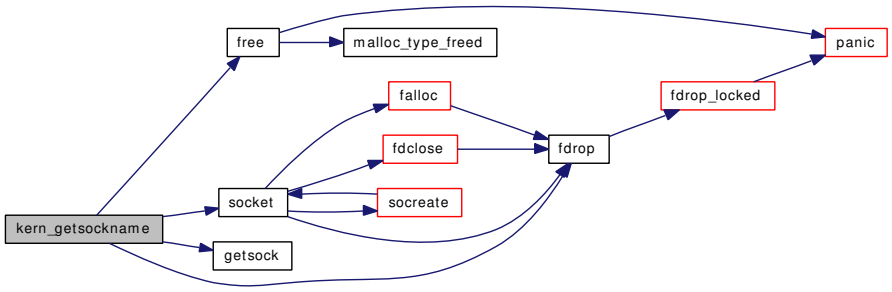
9.149.1.18 int kern_getsockname (struct thread * *td*, int *fd*, struct sockaddr ** *sa*, socklen_t * *alen*)

Definition at line 1539 of file uipc_syscalls.c.

References fdrop(), free(), getsock(), and socket().

Referenced by getsockname1().

Here is the call graph for this function:



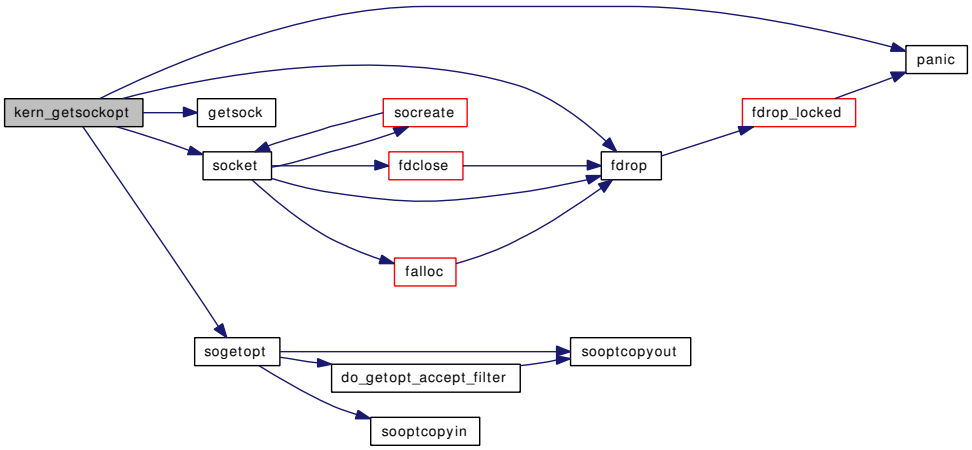
9.149.1.19 int kern_getsockopt (struct thread * *td*, int *s*, int *level*, int *name*, void * *val*, enum uio_seg *valseg*, socklen_t * *valsize*)

Definition at line 1450 of file uipc_syscalls.c.

References fdrop(), getsock(), panic(), socket(), and sogetopt().

Referenced by getsockopt().

Here is the call graph for this function:



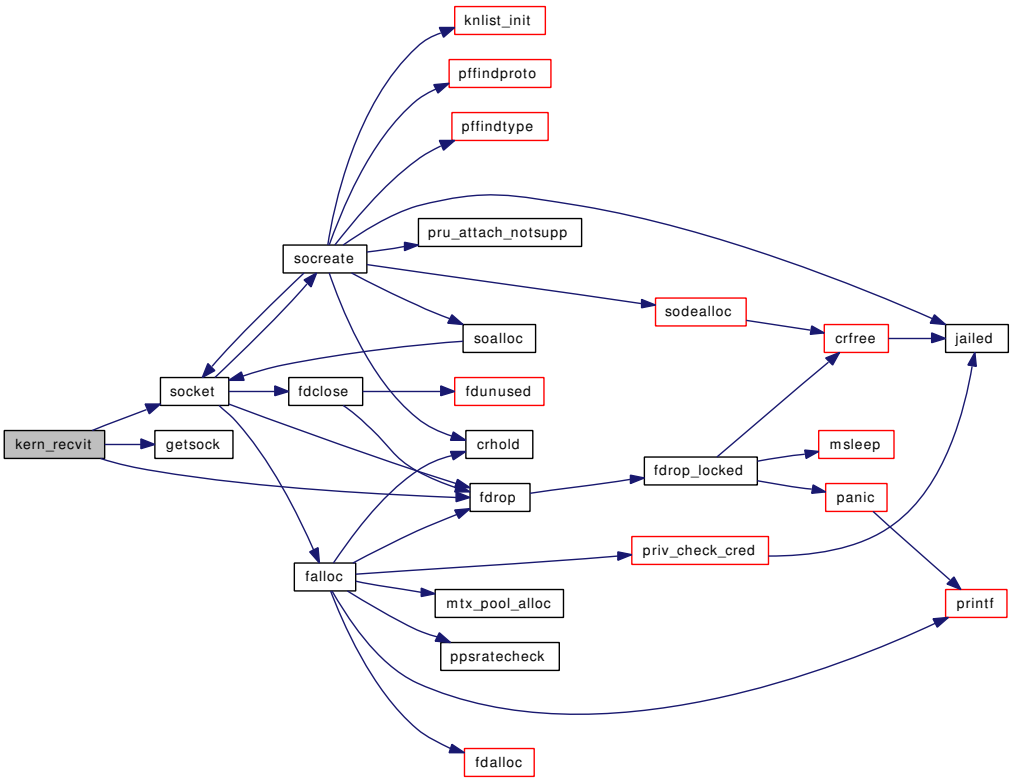
9.149.1.20 int kern_recvit (struct thread * *td*, int *s*, struct msghdr * *mp*, enum uio_seg *fromseg*, struct mbuf ** *controlp*)

Definition at line 976 of file uipc_syscalls.c.

References fdrop(), getsock(), and socket().

Referenced by recvit().

Here is the call graph for this function:



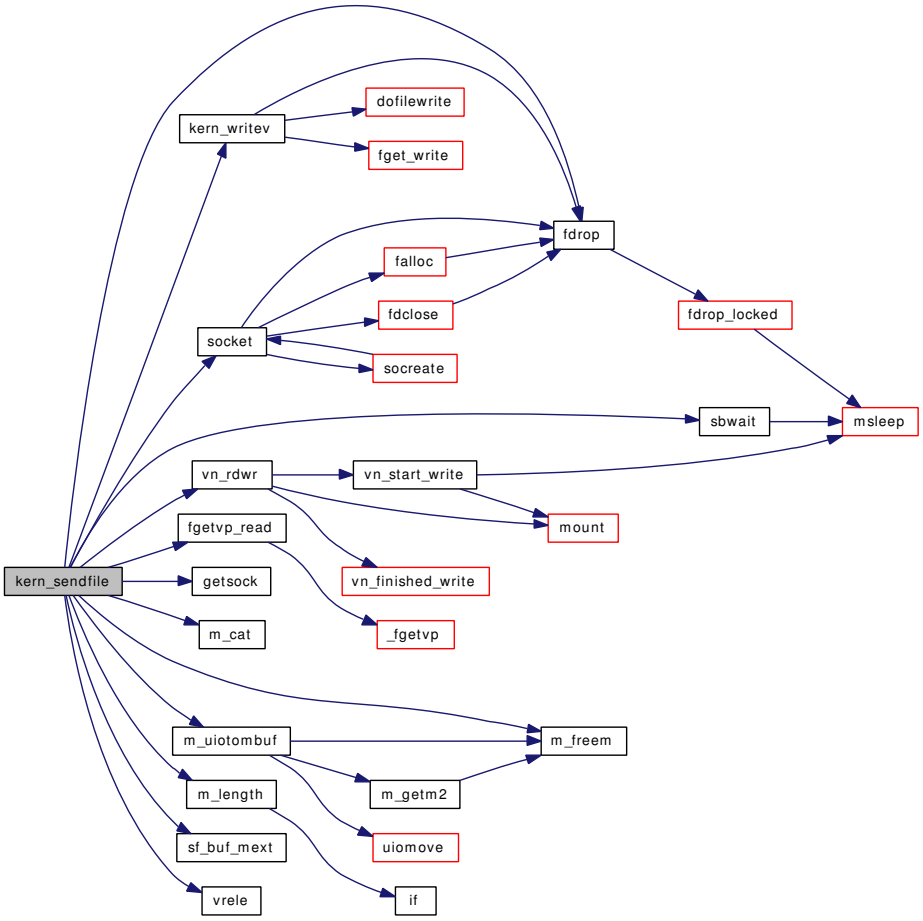
9.149.1.21 int kern_sendfile (struct thread * *td*, struct sendfile_args * *uap*, struct uio * *hdr_uio*, struct uio * *trl_uio*, int *compat*)

Definition at line 1884 of file uipc_syscalls.c.

References fdrop(), fgetvp_read(), getsock(), kern_writev(), m_cat(), m_freem(), m_length(), m_uiotombuf(), mbstat, sbwait(), sf_buf_mext(), socket(), vn_rdwr(), and vrele().

Referenced by do_sendfile().

Here is the call graph for this function:



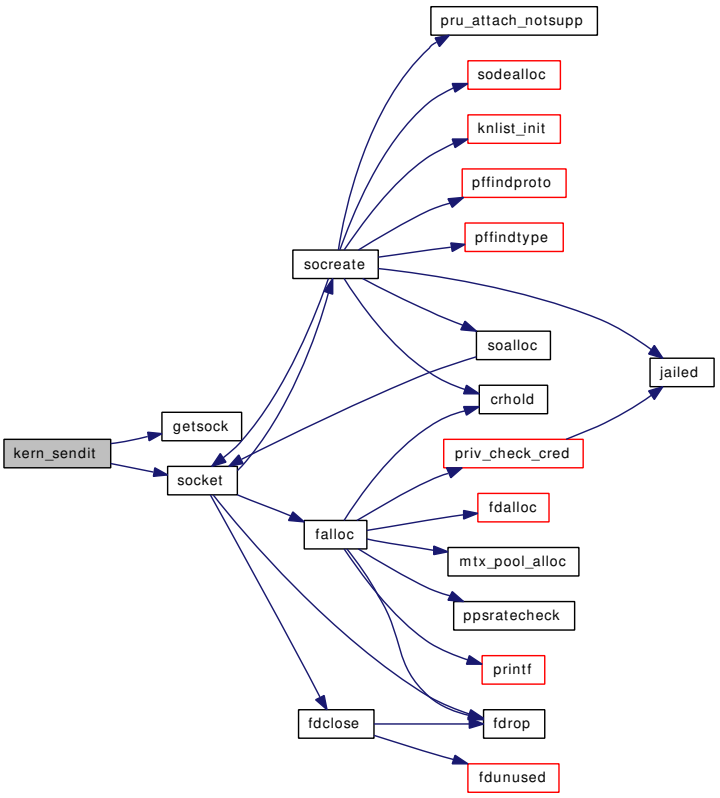
9.149.1.22 int kern_sendit (struct thread * *td*, int *s*, struct msghdr * *mp*, int *flags*, struct mbuf * *control*, enum uio_seg *segflg*)

Definition at line 772 of file uipc_syscalls.c.

References getsock(), and socket().

Referenced by sendit().

Here is the call graph for this function:



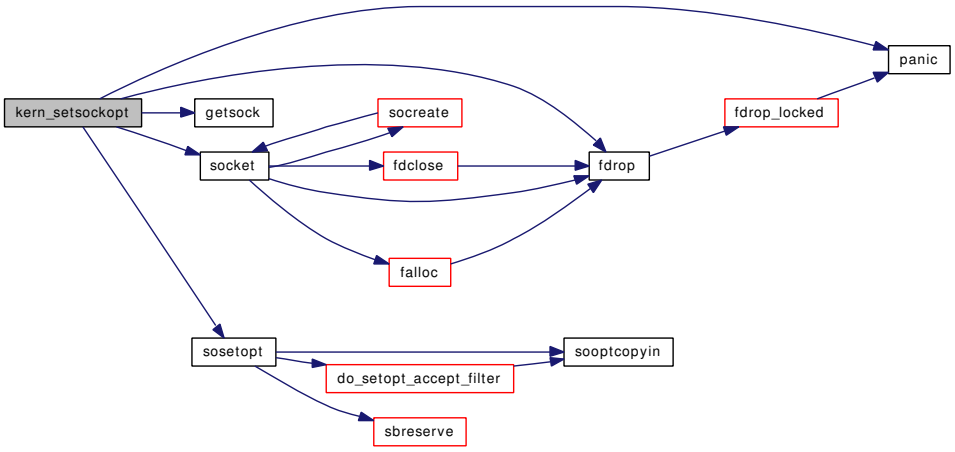
9.149.1.23 int kern_setsockopt (struct thread * *td*, int *s*, int *level*, int *name*, void * *val*, enum uio_seg *valseg*, socklen_t *valsize*)

Definition at line 1367 of file uipc_syscalls.c.

References fdrop(), getsock(), panic(), socket(), and sosetopt().

Referenced by setsockopt().

Here is the call graph for this function:

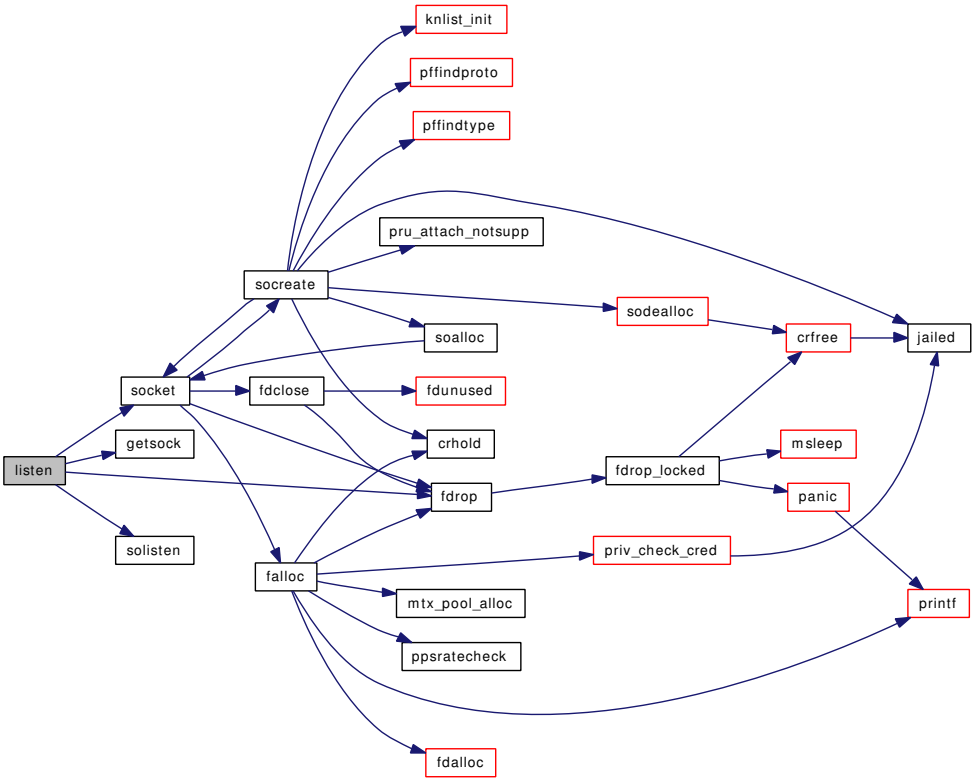


9.149.1.24 int listen (struct thread * *td*, struct listen_args * *uap*)

Definition at line 261 of file uipc_syscalls.c.

References fdrop(), getsock(), socket(), and solisten().

Here is the call graph for this function:

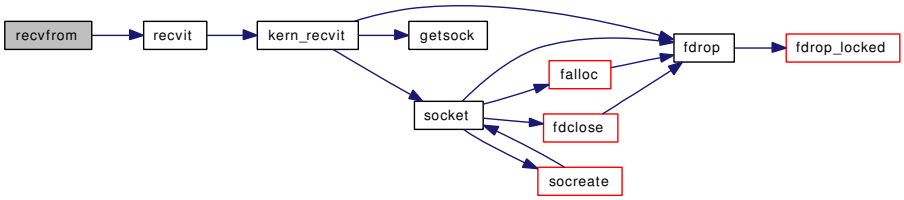


9.149.1.25 int recvfrom (struct thread * *td*, struct recvfrom_args * *uap*)

Definition at line 1164 of file uipc_syscalls.c.

References recvit().

Here is the call graph for this function:



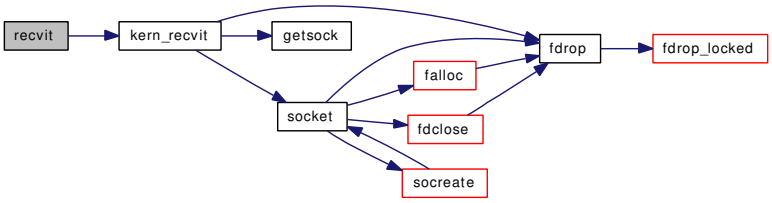
9.149.1.26 static int recvit (struct thread * *td*, int *s*, struct msghdr * *mp*, void * *namelenp*) [static]

Definition at line 1139 of file uipc_syscalls.c.

References kern_recvit().

Referenced by recvfrom(), and recvmsg().

Here is the call graph for this function:

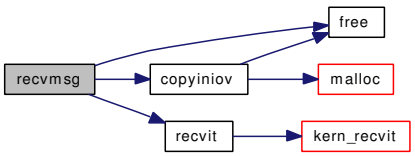


9.149.1.27 int recvmsg (struct thread * *td*, struct recvmsg_args * *uap*)

Definition at line 1286 of file uipc_syscalls.c.

References copyiniov(), free(), and recvit().

Here is the call graph for this function:

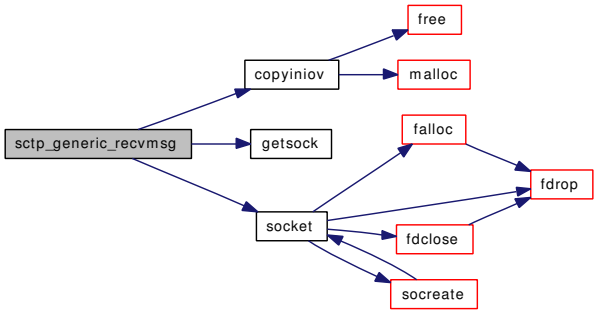


9.149.1.28 int sctp_generic_recvmsg (struct thread * *td*, struct sctp_generic_recvmsg_args * *uap*)

Definition at line 2630 of file uipc_syscalls.c.

References copyiniov(), getsock(), and socket().

Here is the call graph for this function:

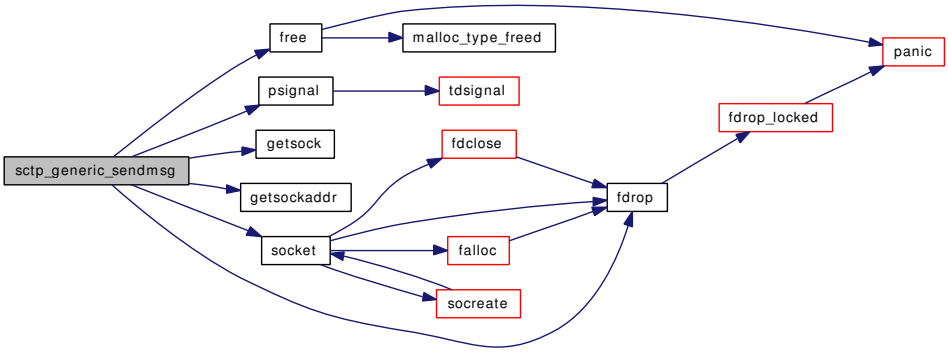


9.149.1.29 int sctp_generic_sendmsg (struct thread * *td*, struct sctp_generic_sendmsg_args * *uap*)

Definition at line 2424 of file uipc_syscalls.c.

References fdrop(), free(), getsock(), getsockaddr(), psignal(), and socket().

Here is the call graph for this function:

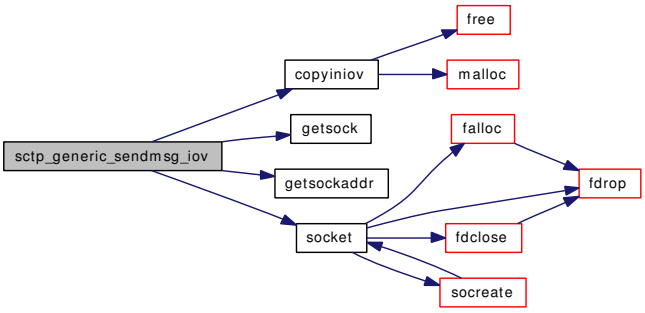


9.149.1.30 int sctp_generic_sendmsg_iov (struct thread * *td*, struct sctp_generic_sendmsg_iov_args * *uap*)

Definition at line 2522 of file uipc_syscalls.c.

References copyiniov(), getsock(), getsockaddr(), and socket().

Here is the call graph for this function:

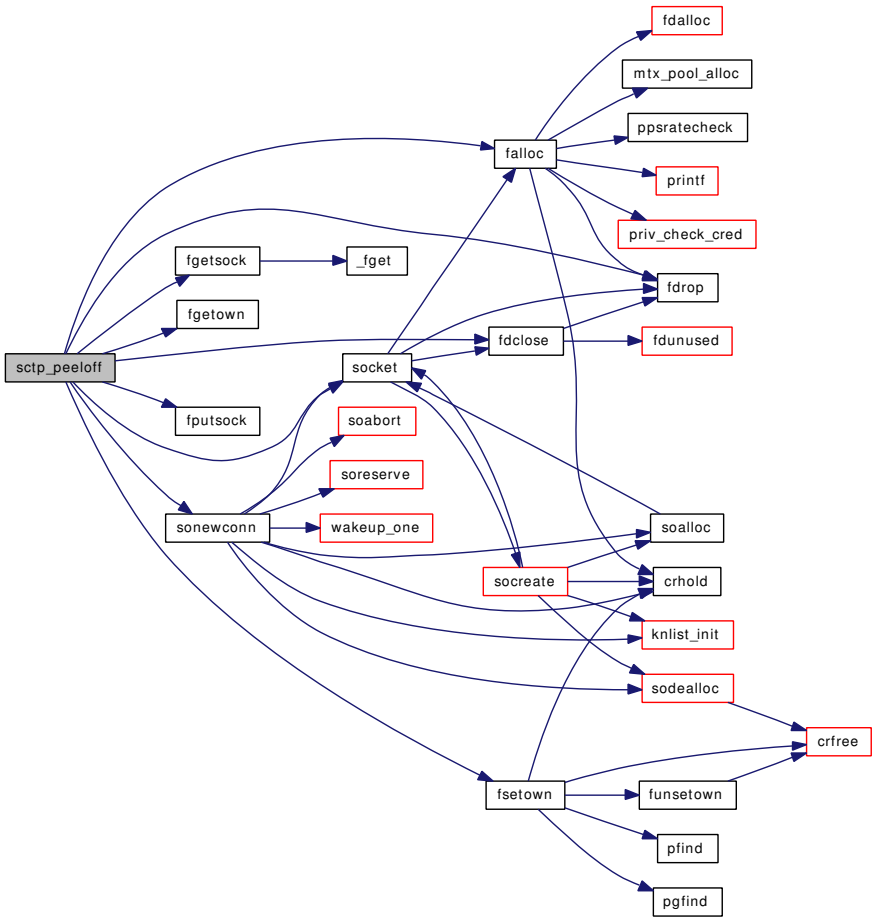


9.149.1.31 int sctp_peeloff (struct thread * *td*, struct sctp_peeloff_args * *uap*)

Definition at line 2332 of file uipc_syscalls.c.

References falloc(), fdclose(), fdrop(), fgetown(), fgetsock(), fputsock(), fsetown(), socket(), socketops, and sonewconn().

Here is the call graph for this function:

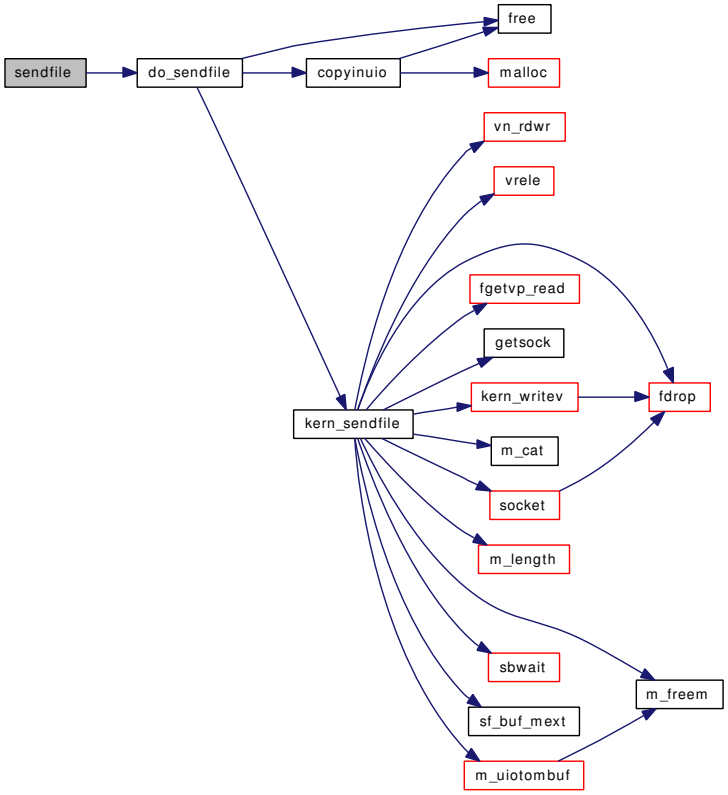


9.149.1.32 int sendfile (struct thread * *td*, struct sendfile_args * *uap*)

Definition at line 1824 of file uipc_syscalls.c.

References do_sendfile().

Here is the call graph for this function:



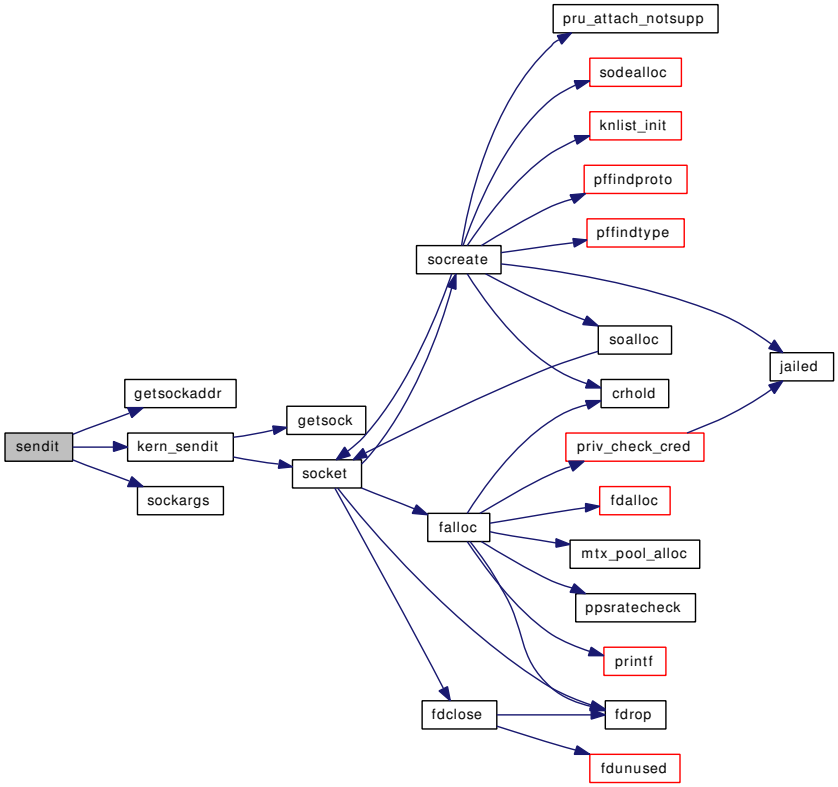
9.149.1.33 static int sendit (struct thread * *td*, int *s*, struct msghdr * *mp*, int *flags*) [static]

Definition at line 709 of file uipc_syscalls.c.

References getsockaddr(), kern_sendit(), and sockargs().

Referenced by sendmsg(), and sendto().

Here is the call graph for this function:

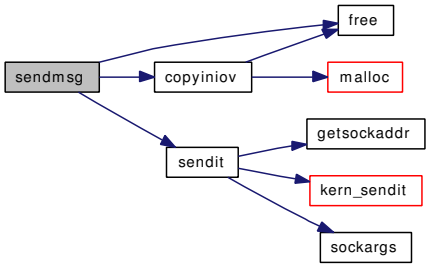


9.149.1.34 int sendmsg (struct thread * *td*, struct sendmsg_args * *uap*)

Definition at line 948 of file uipc_syscalls.c.

References copyiniov(), free(), and sendit().

Here is the call graph for this function:

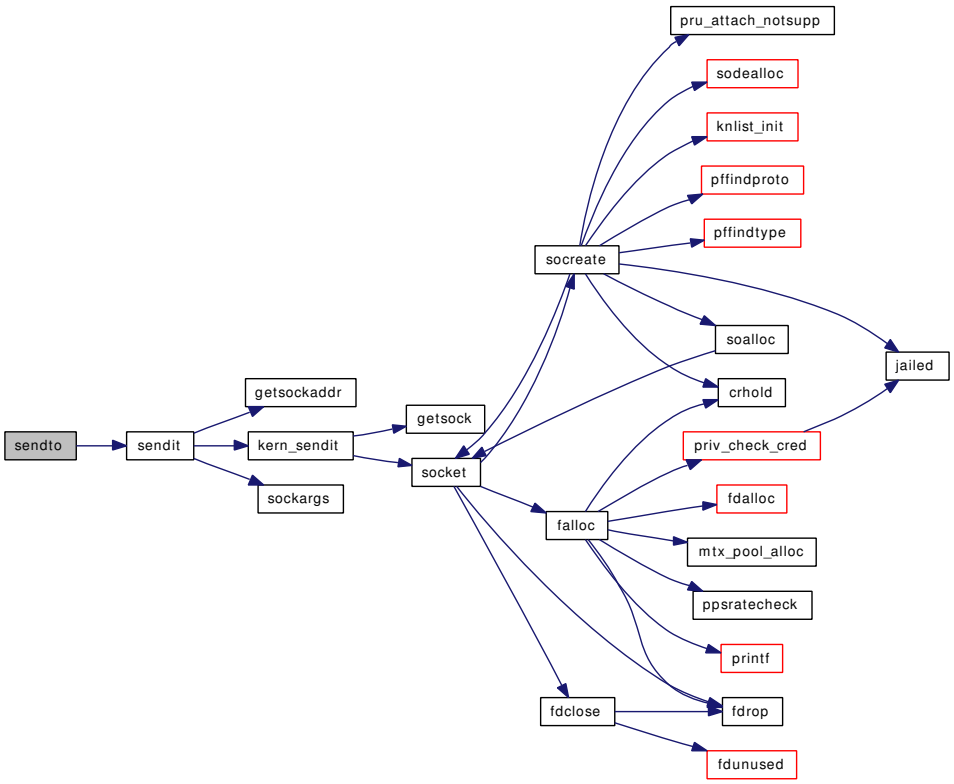


9.149.1.35 int sendto (struct thread * *td*, struct sendto_args * *uap*)

Definition at line 855 of file uipc_syscalls.c.

References sendit().

Here is the call graph for this function:

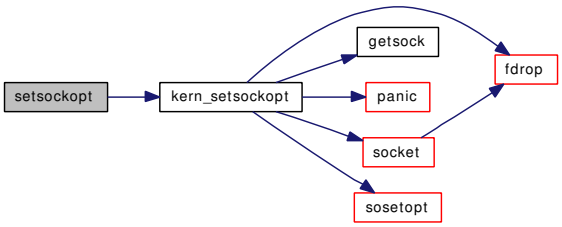


9.149.1.36 int setsockopt (struct thread * *td*, struct setsockopt_args * *uap*)

Definition at line 1351 of file uipc_syscalls.c.

References kern_setsockopt().

Here is the call graph for this function:



9.149.1.37 void sf_buf_mext (void * *addr*, void * *args*)

Definition at line 1791 of file uipc_syscalls.c.

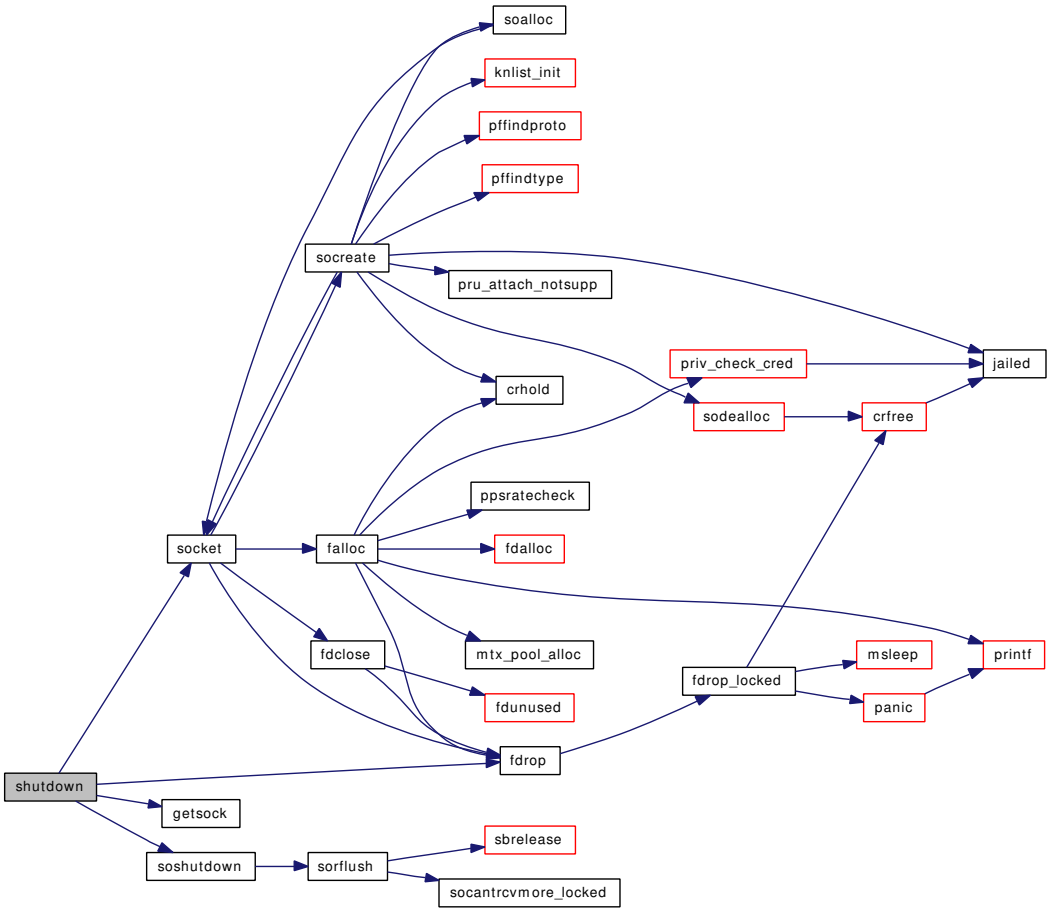
Referenced by kern_sendfile().

9.149.1.38 int shutdown (struct thread * *td*, struct shutdown_args * *uap*)

Definition at line 1324 of file uipc_syscalls.c.

References fdrop(), getsock(), socket(), and soshutdown().

Here is the call graph for this function:



9.149.1.39 int sockargs (struct mbuf ** *mp*, caddr_t *buf*, int *buflen*, int *type*)

Definition at line 1712 of file uipc_syscalls.c.

Referenced by sendit().

9.149.1.40 int socket (struct thread * *td*, struct socket_args * *uap*)

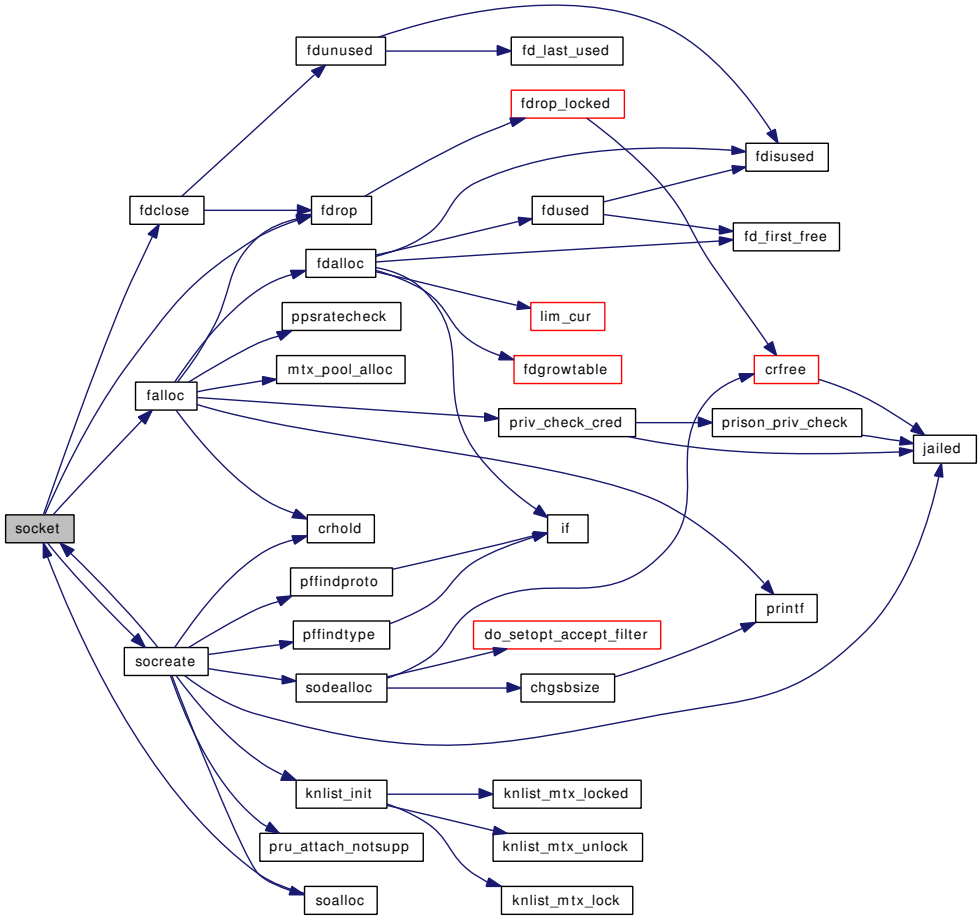
Definition at line 157 of file uipc_syscalls.c.

References falloc(), fdclose(), fdrop(), socketops, and socreate().

Referenced by aio_aqueue(), aio_cancel(), domaininit(), filt_solisten(), filt_sordetach(), filt_soread(), filt_sowdetach(), filt_sowrite(), kern_accept(), kern_bind(), kern_connect(), kern_getpeername(), kern_-getsockname(), kern_getsockopt(), kern_recvit(), kern_sendfile(), kern_sendit(), kern_setsockopt(), lis-

ten(), sctp_generic_recvmsg(), sctp_generic_sendmsg(), sctp_generic_sendmsg_iov(), sctp_peeloff(), shutdown(), soalloc(), socketpair(), soclose(), socreate(), sofree(), soisconnected(), sonewconn(), soo_close(), soo_ioctl(), soo_kqfilter(), soo_poll(), soo_read(), soo_stat(), soo_write(), uipc_rcvd(), uipc_send(), uipc_sense(), unp_connect(), unp_disconnect(), unp_drop(), unp_gc(), and unp_shutdown().

Here is the call graph for this function:

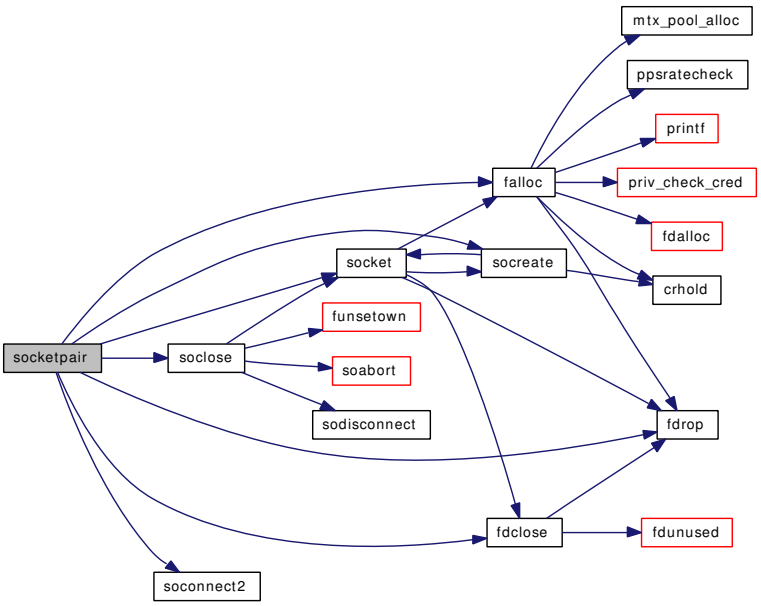


9.149.1.41 int socketpair (struct thread * *td*, struct socketpair_args * *uap*)

Definition at line 626 of file uipc_syscalls.c.

References falloc(), fdclose(), fdrop(), socket(), socketops, soclose(), soconnect2(), and socreate().

Here is the call graph for this function:



9.149.1.42 SYSCTL_INT (_kern_ipc, OID_AUTO, nsfbufsused, CTLFLAG_RD, & *nsfbufsused*, 0, "Number of sendfile(2) sf_bufs in use")

9.149.1.43 SYSCTL_INT (_kern_ipc, OID_AUTO, nsfbufspeak, CTLFLAG_RD, & *nsfbufspeak*, 0, "Number of sendfile(2) sf_bufs at peak usage")

9.149.1.44 SYSCTL_INT (_kern_ipc, OID_AUTO, nsfbufs, CTLFLAG_RDTUN, & *nsfbufs*, 0, "Maximum number of sendfile(2) sf_bufs available")

9.149.2 Variable Documentation

9.149.2.1 int nsfbufs

Definition at line 98 of file uipc_syscalls.c.

9.149.2.2 int nsfbufspeak

Definition at line 99 of file uipc_syscalls.c.

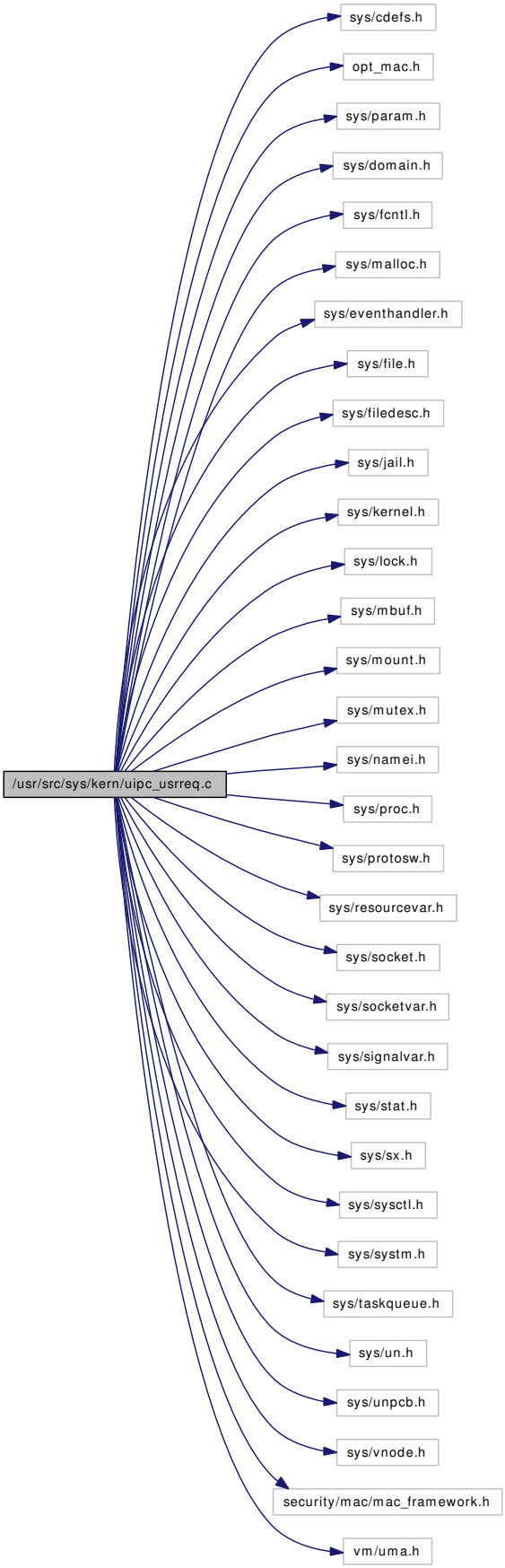
9.149.2.3 int nsfbufsused

Definition at line 100 of file uipc_syscalls.c.

9.150 /usr/src/sys/kern/uipc_usrreq.c File Reference

```
#include <sys/cdefs.h>
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/domain.h>
#include <sys/fcntl.h>
#include <sys/malloc.h>
#include <sys/eventhandler.h>
#include <sys/file.h>
#include <sys/filedesc.h>
#include <sys/jail.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/mbuf.h>
#include <sys/mount.h>
#include <sys/mutex.h>
#include <sys/namei.h>
#include <sys/proc.h>
#include <sys/protosw.h>
#include <sys/resourcevar.h>
#include <sys/socket.h>
#include <sys/socketvar.h>
#include <sys/signalvar.h>
#include <sys/stat.h>
#include <sys/sx.h>
#include <sys/sysctl.h>
#include <sys/systm.h>
#include <sys/taskqueue.h>
#include <sys/un.h>
#include <sys/unpcb.h>
#include <sys/vnode.h>
#include <security/mac/mac_framework.h>
#include <vm/uma.h>
```

Include dependency graph for uipc_usrreq.c:


/usr/src/sys/kern/uipc_usrreq.c
sys/cdefs.h
opt_mac.h
sys/param.h
sys/domain.h
sys/fcntl.h
sys/malloc.h
sys/eventhandler.h
sys/file.h
sys/filedesc.h
sys/jail.h
sys/kernel.h
sys/lock.h
sys/mbuf.h
sys/mount.h
sys/mutex.h
sys/namei.h
sys/proc.h
sys/protosw.h
sys/resourcevar.h
sys/socket.h
sys/socketvar.h
sys/signalvar.h
sys/stat.h
sys/sx.h
sys/sysctl.h
sys/systm.h
sys/taskqueue.h
sys/un.h
sys/unpcb.h
sys/vnode.h
security/mac/mac_framework.h
vm/uma.h

Defines

- #define PIPSIZ 8192
- #define UNP_LOCK_INIT() mtx_init(&unp_mtx, "unp", NULL, MTX_DEF | MTX_RECURSE)
- #define UNP_LOCK() mtx_lock(&unp_mtx)
- #define UNP_UNLOCK() mtx_unlock(&unp_mtx)
- #define UNP_LOCK_ASSERT() mtx_assert(&unp_mtx, MA_OWNED)
- #define UNP_UNLOCK_ASSERT() mtx_assert(&unp_mtx, MA_NOTOWNED)
- #define OPTSET(bit)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/uipc_usrreq.c,v 1.196 2007/02/22 09:37:44 rwatson Exp $")
- SYSCTL_NODE (_net, PF_LOCAL, local, CTLFLAG_RW, 0,"Local domain")
- SYSCTL_NODE (_net_local, SOCK_STREAM, stream, CTLFLAG_RW, 0,"SOCK_STREAM")
- SYSCTL_NODE (_net_local, SOCK_DGRAM, dgram, CTLFLAG_RW, 0,"SOCK_DGRAM")
- SYSCTL_ULONG (_net_local_stream, OID_AUTO, sendspace, CTLFLAG_RW,&unpst_sendspace, 0,"")
- SYSCTL_ULONG (_net_local_stream, OID_AUTO, recvspace, CTLFLAG_RW,&unpst_recvspace, 0,"")
- SYSCTL_ULONG (_net_local_dgram, OID_AUTO, maxdgram, CTLFLAG_RW,&unpdg_sendspace, 0,"")
- SYSCTL_ULONG (_net_local_dgram, OID_AUTO, recvspace, CTLFLAG_RW,&unpdg_recvspace, 0,"")
- SYSCTL_INT (_net_local, OID_AUTO, inflight, CTLFLAG_RD,&unp_rights, 0,"")
- static int unp_connect (struct socket *, struct sockaddr *, struct thread *)
- static int unp_connect2 (struct socket *so, struct socket *so2, int)
- static void unp_disconnect (struct unpcb *)
- static void unp_shutdown (struct unpcb *)
- static void unp_drop (struct unpcb *, int)
- static void unp_gc (__unused void *, int)
- static void unp_scan (struct mbuf *, void(*)(struct file *))
- static void unp_mark (struct file *)
- static void unp_discard (struct file *)
- static void unp_freerights (struct file **, int)
- static int unp_internalize (struct mbuf **, struct thread *)
- static int unp_listen (struct socket *, struct unpcb *, int, struct thread *)
- static struct mbuf * unp_addsockcred (struct thread *, struct mbuf *)
- DOMAIN_SET (local)
- static void uipc_abort (struct socket *so)
- static int uipc_accept (struct socket *so, struct sockaddr **nam)
- static int uipc_attach (struct socket *so, int proto, struct thread *td)
- static int uipc_bind (struct socket *so, struct sockaddr *nam, struct thread *td)
- static int uipc_connect (struct socket *so, struct sockaddr *nam, struct thread *td)
- static void uipc_close (struct socket *so)
- int uipc_connect2 (struct socket *so1, struct socket *so2)
- static void uipc_detach (struct socket *so)
- static int uipc_disconnect (struct socket *so)
- static int uipc_listen (struct socket *so, int backlog, struct thread *td)
- static int uipc_peeraddr (struct socket *so, struct sockaddr **nam)

- static int uipc_rcvd (struct socket *so, int flags)
- static int uipc_send (struct socket *so, int flags, struct mbuf *m, struct sockaddr *nam, struct mbuf *control, struct thread *td)
- static int uipc_sense (struct socket *so, struct stat *sb)
- static int uipc_shutdown (struct socket *so)
- static int uipc_sockaddr (struct socket *so, struct sockaddr **nam)
- int uipc_ctloutput (struct socket *so, struct sockopt *sopt)
- static int unp_pcblist (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_net_local_dgram, OID_AUTO, pcblist, CTLFLAG_RD,(caddr_t)(long) SOCK_DGRAM, 0, unp_pcblist,"S,xunpcb","List of active local datagram sockets")
- SYSCTL_PROC (_net_local_stream, OID_AUTO, pcblist, CTLFLAG_RD,(caddr_t)(long) SOCK_STREAM, 0, unp_pcblist,"S,xunpcb","List of active local stream sockets")
- int unp_externalize (struct mbuf *control, struct mbuf **controlp)
- static void unp_zone_change (void *tag)
- void unp_init (void)
- SYSCTL_INT (_net_local, OID_AUTO, taskcount, CTLFLAG_RD,&unp_taskcount, 0,"")
- SYSCTL_INT (_net_local, OID_AUTO, recycled, CTLFLAG_RD,&unp_recycled, 0,"")
- void unp_dispose (struct mbuf *m)

Variables

- static uma_zone_t unp_zone
- static unp_gen_t unp_gencnt
- static u_int unp_count
- static ino_t unp_ino
- static int unp_rights
- static struct unp_head unp_shead
- static struct unp_head unp_dhead
- static struct sockaddr sun_noname = { sizeof(sun_noname), AF_LOCAL }
- static struct task unp_gc_task
- static u_long unpst_sendspace = PIPSIZ
- static u_long unpst_recvspace = PIPSIZ
- static u_long unpdg_sendspace = 2*1024
- static u_long unpdg_recvspace = 4*1024
- static struct mtx unp_mtx
- static struct domain localdomain
- static struct protosw localsw [ ]
- static struct domain localdomain
- pr_usrreqs uipc_usrreqs
- static int unp_defer
- static int unp_taskcount
- static int unp_recycled

9.150.1 Define Documentation

9.150.1.1 #define OPTSET(bit)

Value:

```
if (optval) \
                unp->unp_flags |= bit; \
        else \
                unp->unp_flags &= ~bit;
```

Referenced by uipc_ctloutput().

9.150.1.2 #define PIPSIZ 8192

Definition at line 125 of file uipc_usrreq.c.

9.150.1.3 #define UNP_LOCK() mtx_lock(&unp_mtx)

Definition at line 169 of file uipc_usrreq.c.

Referenced by uipc_abort(), uipc_accept(), uipc_attach(), uipc_bind(), uipc_close(), uipc_connect(), uipc_connect2(), uipc_ctloutput(), uipc_detach(), uipc_disconnect(), uipc_listen(), uipc_peeraddr(), uipc_-rcvd(), uipc_send(), uipc_sense(), uipc_shutdown(), uipc_sockaddr(), unp_connect(), unp_discard(), and unp_pcblist().

9.150.1.4 #define UNP_LOCK_ASSERT() mtx_assert(&unp_mtx, MA_OWNED)

Definition at line 171 of file uipc_usrreq.c.

Referenced by unp_connect(), unp_connect2(), unp_disconnect(), unp_drop(), unp_listen(), and unp_-shutdown().

9.150.1.5 #define UNP_LOCK_INIT() mtx_init(&unp_mtx, "unp", NULL, MTX_DEF | MTX_RECURSE)

Definition at line 167 of file uipc_usrreq.c.

Referenced by unp_init().

9.150.1.6 #define UNP_UNLOCK() mtx_unlock(&unp_mtx)

Definition at line 170 of file uipc_usrreq.c.

Referenced by uipc_abort(), uipc_accept(), uipc_attach(), uipc_bind(), uipc_close(), uipc_connect(), uipc_connect2(), uipc_ctloutput(), uipc_detach(), uipc_disconnect(), uipc_listen(), uipc_rcvd(), uipc_-send(), uipc_sense(), uipc_shutdown(), uipc_sockaddr(), unp_connect(), unp_discard(), and unp_pcblist().

9.150.1.7 #define UNP_UNLOCK_ASSERT() mtx_assert(&unp_mtx, MA_NOTOWNED)

Definition at line 172 of file uipc_usrreq.c.

Referenced by unp_externalize(), and unp_internalize().

9.150.2 Function Documentation

9.150.2.1 __FBSDID ("$FreeBSD: src/sys/kern/uipc_usrreq. *c*, v 1.196 2007/02/22 09:37:44 rwatson Exp $")

9.150.2.2 DOMAIN_SET (local)

9.150.2.3 SYSCTL_INT (_net_local, OID_AUTO, recycled, CTLFLAG_RD, & *unp_recycled*, 0, "")

9.150.2.4 SYSCTL_INT (_net_local, OID_AUTO, taskcount, CTLFLAG_RD, & *unp_taskcount*, 0, "")

9.150.2.5 SYSCTL_INT (_net_local, OID_AUTO, inflight, CTLFLAG_RD, & *unp_rights*, 0, "")

9.150.2.6 SYSCTL_NODE (_net_local, SOCK_DGRAM, dgram, CTLFLAG_RW, 0, "SOCK_DGRAM")

9.150.2.7 SYSCTL_NODE (_net_local, SOCK_STREAM, stream, CTLFLAG_RW, 0, "SOCK_STREAM")

9.150.2.8 SYSCTL_NODE (_net, PF_LOCAL, local, CTLFLAG_RW, 0, "Local domain")

9.150.2.9 SYSCTL_PROC (_net_local_stream, OID_AUTO, pcblist, CTLFLAG_RD, (caddr_t)(long) *SOCK_STREAM*, 0, unp_pcblist, " *S*, xunpcb", "List of active local stream sockets")

9.150.2.10 SYSCTL_PROC (_net_local_dgram, OID_AUTO, pcblist, CTLFLAG_RD, (caddr_t)(long) *SOCK_DGRAM*, 0, unp_pcblist, " *S*, xunpcb", "List of active local datagram sockets")

9.150.2.11 SYSCTL_ULONG (_net_local_dgram, OID_AUTO, recvspace, CTLFLAG_RW, & *unpdg_recvspace*, 0, "")

9.150.2.12 SYSCTL_ULONG (_net_local_dgram, OID_AUTO, maxdgram, CTLFLAG_RW, & *unpdg_sendspace*, 0, "")

9.150.2.13 SYSCTL_ULONG (_net_local_stream, OID_AUTO, recvspace, CTLFLAG_RW, & *unpst_recvspace*, 0, "")

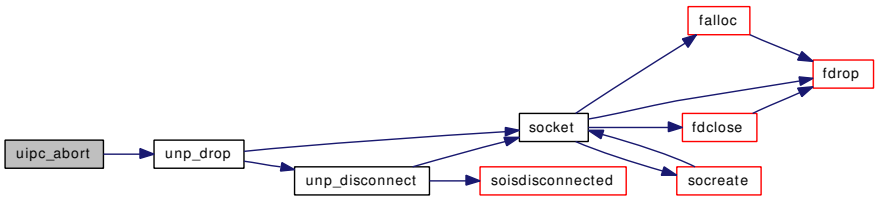
9.150.2.14 SYSCTL_ULONG (_net_local_stream, OID_AUTO, sendspace, CTLFLAG_RW, & *unpst_sendspace*, 0, "")

9.150.2.15 static void uipc_abort (struct socket * *so*) `[static]`

Definition at line 222 of file uipc_usrreq.c.

References unp_drop(), UNP_LOCK, and UNP_UNLOCK.

Here is the call graph for this function:

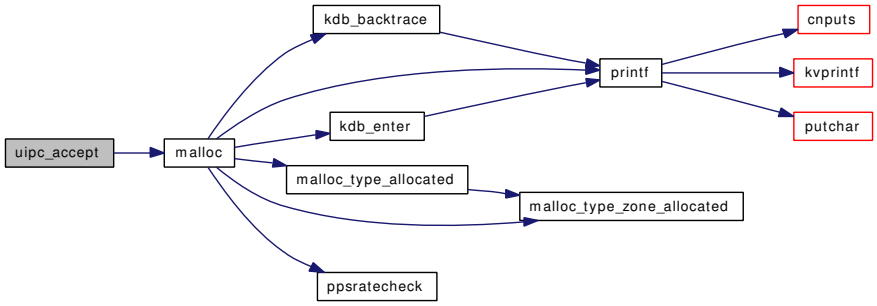


9.150.2.16 static int uipc_accept (struct socket * *so*, struct sockaddr ** *nam*) [static]

Definition at line 234 of file uipc_usrreq.c.

References malloc(), sun_noname, UNP_LOCK, and UNP_UNLOCK.

Here is the call graph for this function:

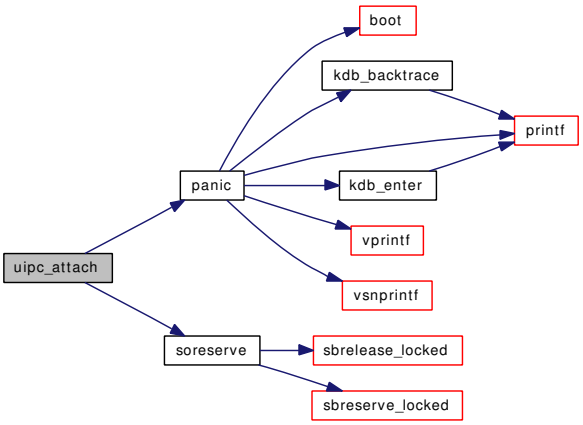


9.150.2.17 static int uipc_attach (struct socket * *so*, int *proto*, struct thread * *td*) [static]

Definition at line 257 of file uipc_usrreq.c.

References panic(), soreserve(), unp_count, unp_dhead, unp_gencnt, UNP_LOCK, unp_shead, UNP_-UNLOCK, and unp_zone.

Here is the call graph for this function:

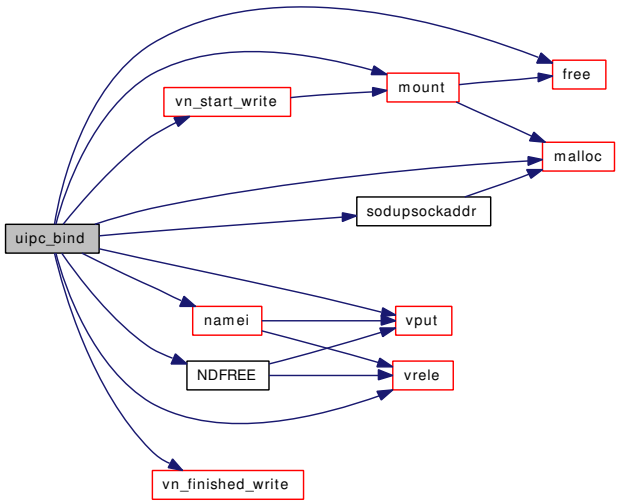


9.150.2.18 static int uipc_bind (struct socket * *so*, struct sockaddr * *nam*, struct thread * *td*) [static]

Definition at line 298 of file uipc_usrreq.c.

References buf, free(), Giant, malloc(), mount(), namei(), NDFREE(), sodupsockaddr(), UNP_LOCK, UNP_UNLOCK, vn_finished_write(), vn_start_write(), vput(), and vrele().

Here is the call graph for this function:

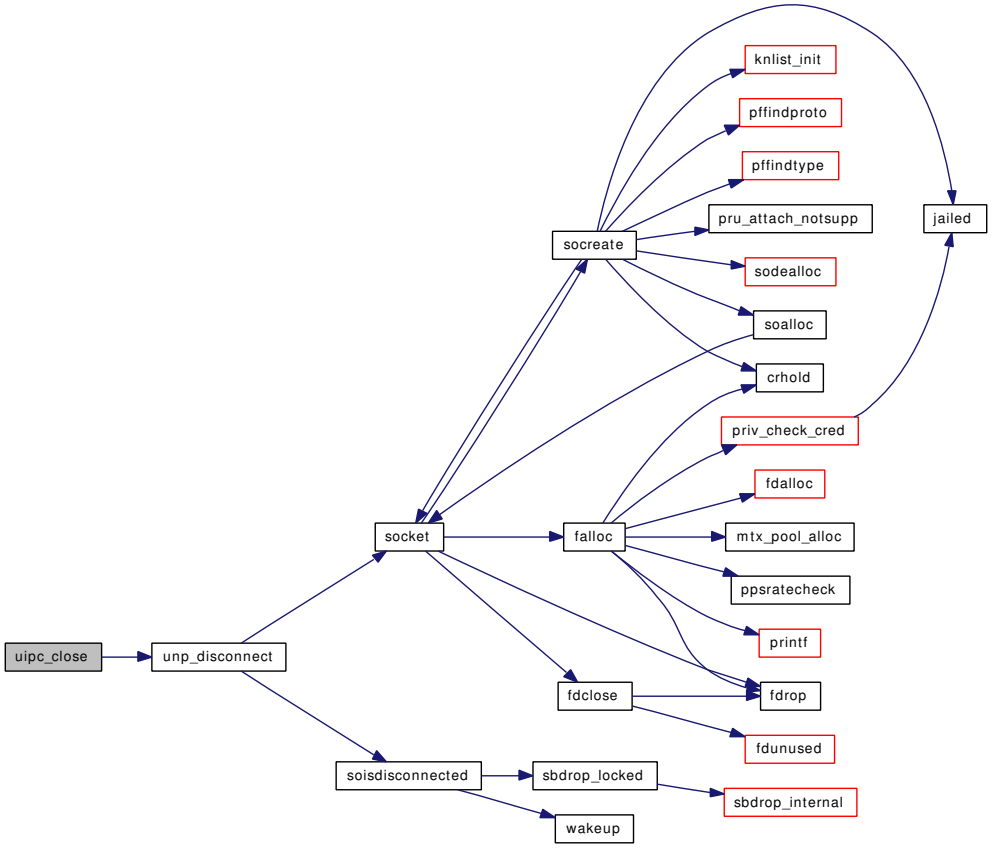


9.150.2.19 static void uipc_close (struct socket * *so*) [static]

Definition at line 423 of file uipc_usrreq.c.

References unp_disconnect(), UNP_LOCK, and UNP_UNLOCK.

Here is the call graph for this function:

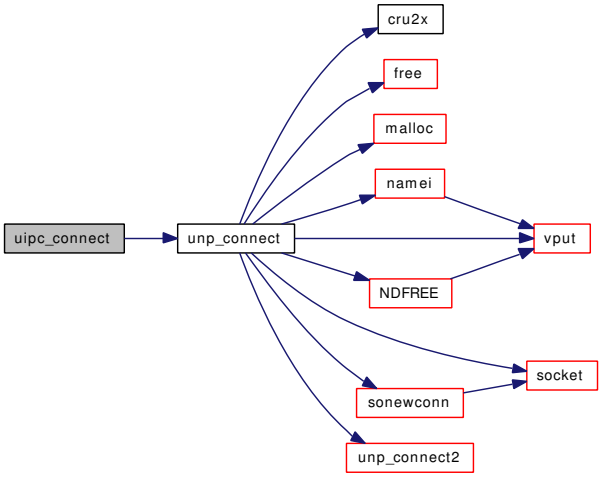


9.150.2.20 static int uipc_connect (struct socket * *so*, struct sockaddr * *nam*, struct thread * *td*) [static]

Definition at line 408 of file uipc_usrreq.c.

References unp_connect(), UNP_LOCK, and UNP_UNLOCK.

Here is the call graph for this function:

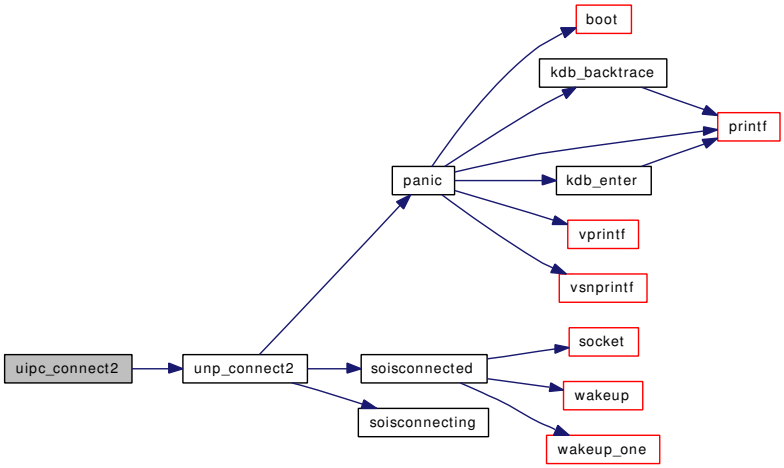


9.150.2.21 int uipc_connect2 (struct socket * *so1*, struct socket * *so2*)

Definition at line 435 of file uipc_usrreq.c.

References unp_connect2(), UNP_LOCK, and UNP_UNLOCK.

Here is the call graph for this function:

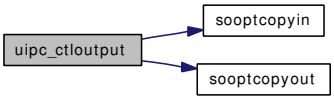


9.150.2.22 int uipc_ctloutput (struct socket * *so*, struct sockopt * *sopt*)

Definition at line 840 of file uipc_usrreq.c.

References OPTSET, sooptcopyin(), sooptcopyout(), UNP_LOCK, and UNP_UNLOCK.

Here is the call graph for this function:

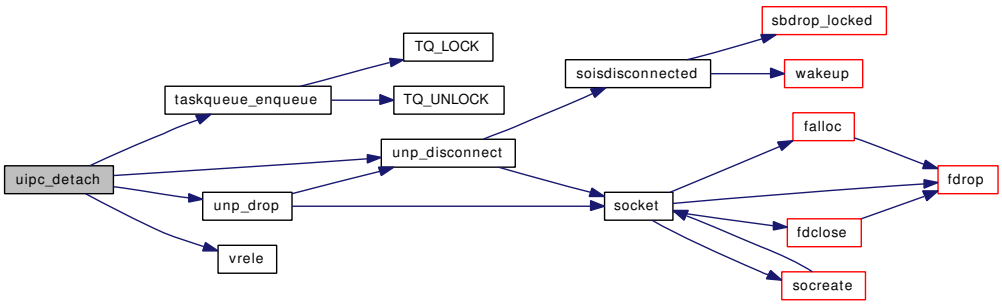


9.150.2.23 static void uipc_detach (struct socket * *so*) [static]

Definition at line 451 of file uipc_usrreq.c.

References taskqueue_enqueue(), unp_count, unp_disconnect(), unp_drop(), unp_gencnt, UNP_LOCK, unp_rights, UNP_UNLOCK, unp_zone, and vrele().

Here is the call graph for this function:

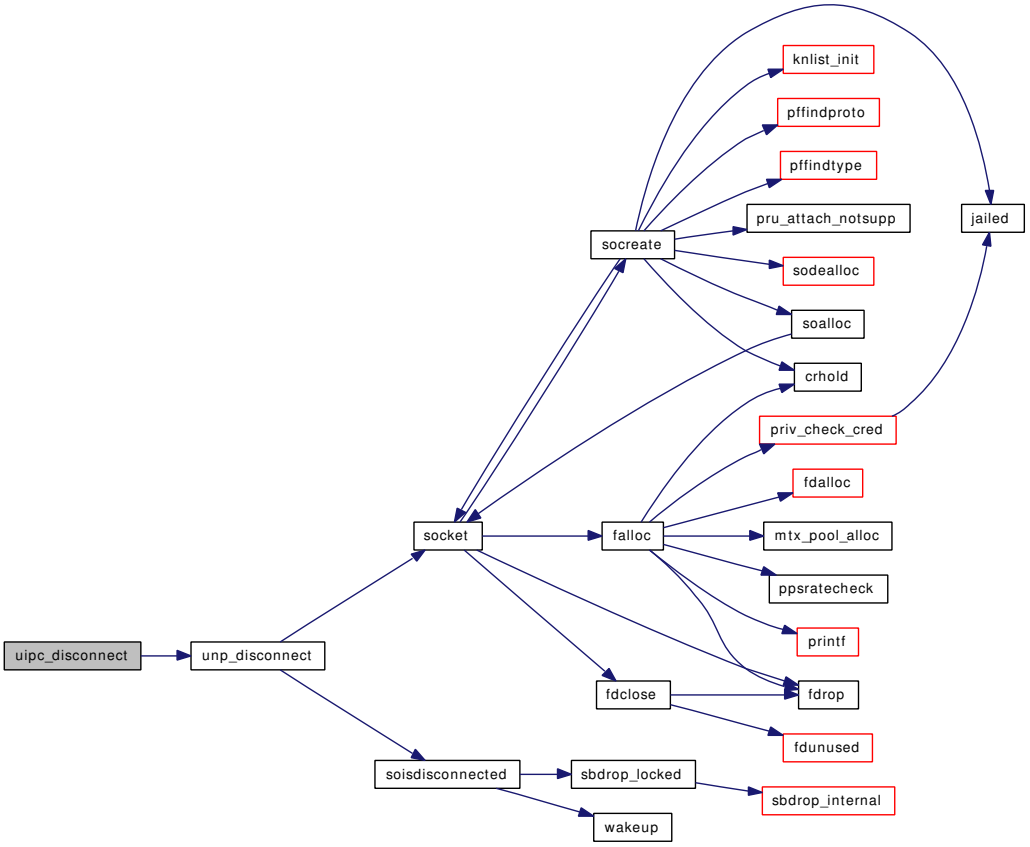


9.150.2.24 static int uipc_disconnect (struct socket * *so*) [static]

Definition at line 497 of file uipc_usrreq.c.

References unp_disconnect(), UNP_LOCK, and UNP_UNLOCK.

Here is the call graph for this function:

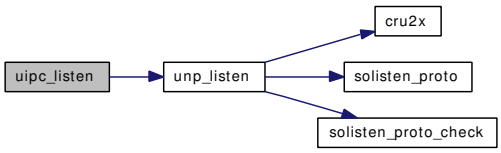


9.150.2.25 static int uipc_listen (struct socket * *so*, int *backlog*, struct thread * *td*) [static]

Definition at line 510 of file uipc_usrreq.c.

References unp_listen(), UNP_LOCK, and UNP_UNLOCK.

Here is the call graph for this function:

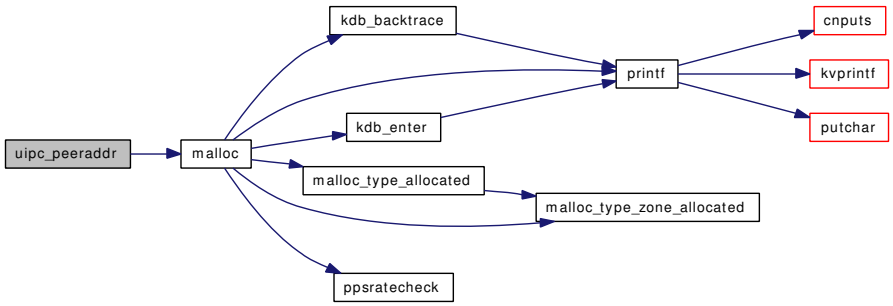


9.150.2.26 static int uipc_peeraddr (struct socket * *so*, struct sockaddr ** *nam*) [static]

Definition at line 528 of file uipc_usrreq.c.

References malloc(), sun_noname, and UNP_LOCK.

Here is the call graph for this function:

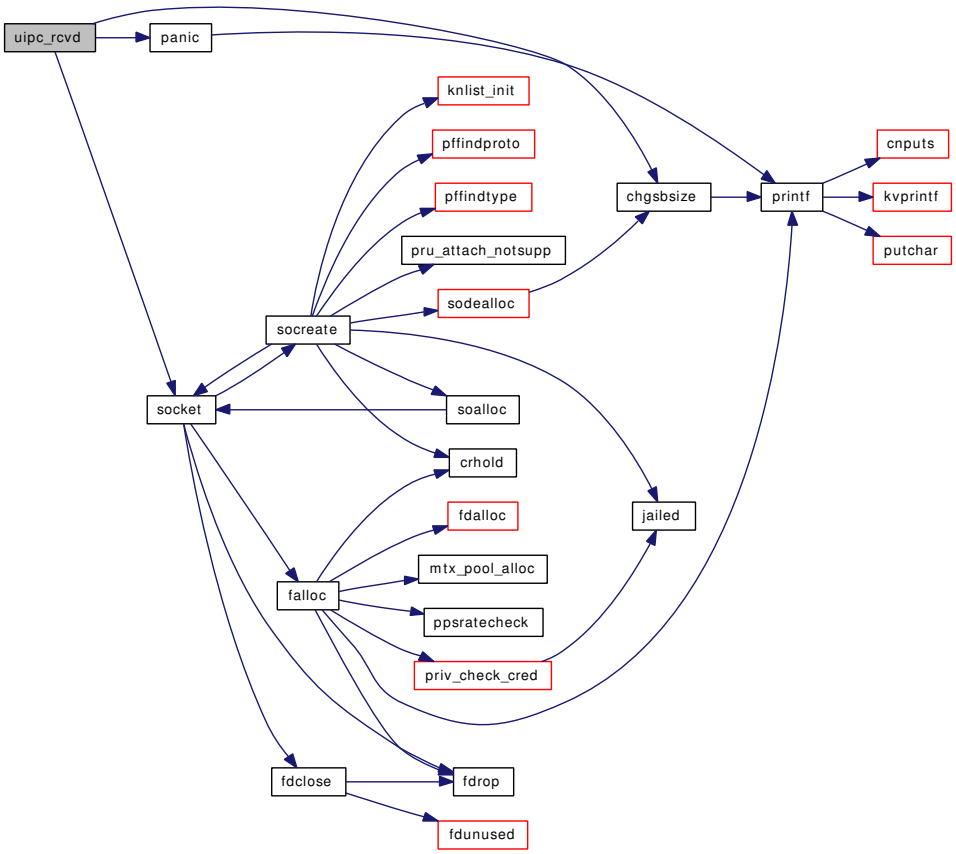


9.150.2.27 static int uipc_rcvd (struct socket * *so*, int *flags*) [static]

Definition at line 553 of file uipc_usrreq.c.

References chgsbsize(), panic(), socket(), UNP_LOCK, and UNP_UNLOCK.

Here is the call graph for this function:

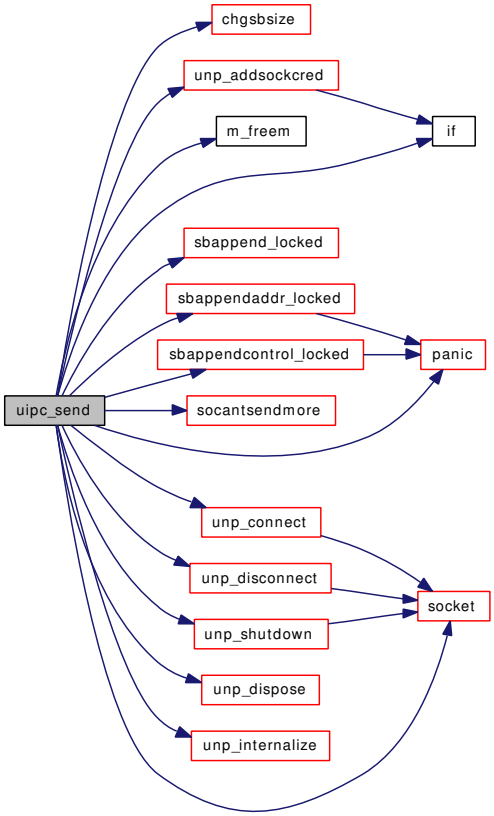


9.150.2.28 static int uipc_send (struct socket * *so*, int *flags*, struct mbuf * *m*, struct sockaddr * *nam*, struct mbuf * *control*, struct thread * *td*) [static]

Definition at line 602 of file uipc_usrreq.c.

References chgsbsize(), if(), m_freem(), panic(), sbappend_locked(), sbappendaddr_locked(), sbappendcontrol_locked(), socantsendmore(), socket(), sun_noname, unp_addsockcred(), unp_connect(), unp_disconnect(), unp_dispose(), unp_internalize(), UNP_LOCK, unp_shutdown(), and UNP_UNLOCK.

Here is the call graph for this function:

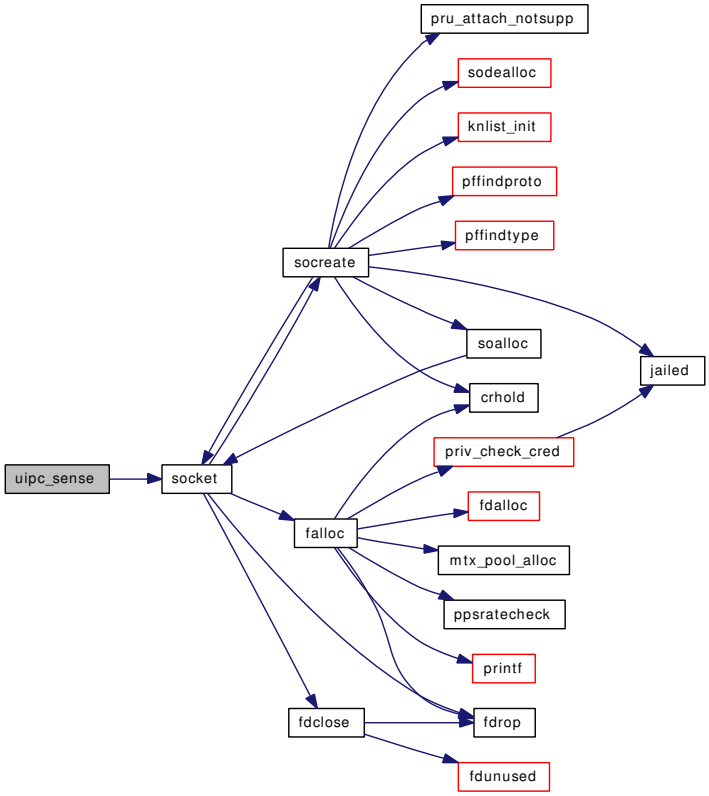


9.150.2.29 static int uipc_sense (struct socket * *so*, struct stat * *sb*) [static]

Definition at line 766 of file uipc_usrreq.c.

References socket(), unp_ino, UNP_LOCK, and UNP_UNLOCK.

Here is the call graph for this function:

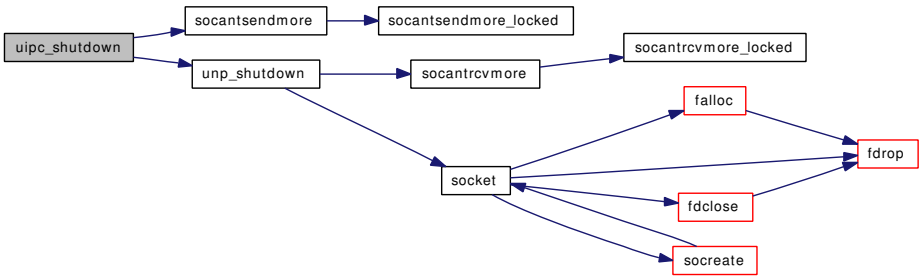


9.150.2.30 static int uipc_shutdown (struct socket * *so*) [static]

Definition at line 788 of file uipc_usrreq.c.

References socantsendmore(), UNP_LOCK, unp_shutdown(), and UNP_UNLOCK.

Here is the call graph for this function:

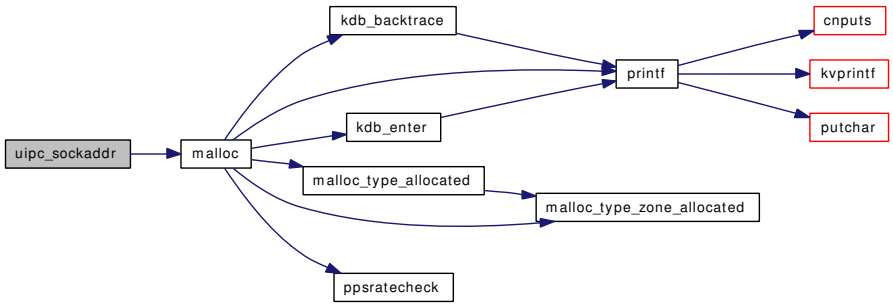


9.150.2.31 static int uipc_sockaddr (struct socket * *so*, struct sockaddr ** *nam*) [static]

Definition at line 802 of file uipc_usrreq.c.

References malloc(), sun_noname, UNP_LOCK, and UNP_UNLOCK.

Here is the call graph for this function:



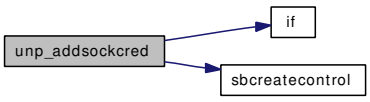
9.150.2.32 static struct mbuf * unp_addsockcred (struct thread *, struct mbuf *) [static]

Definition at line 1563 of file uipc_usrreq.c.

References if(), and sbcreatecontrol().

Referenced by uipc_send().

Here is the call graph for this function:



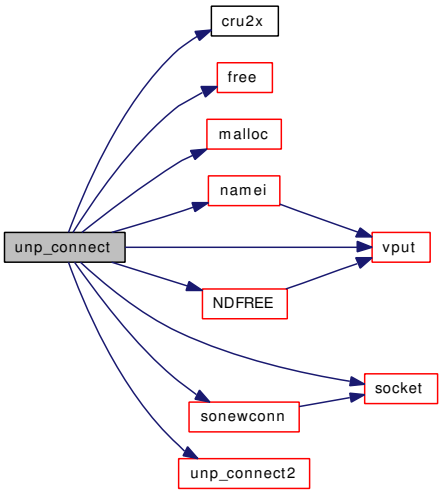
9.150.2.33 static int unp_connect (struct socket *, struct sockaddr *, struct thread *) [static]

Definition at line 926 of file uipc_usrreq.c.

References buf, cru2x(), free(), Giant, malloc(), namei(), NDFREE(), socket(), sonewconn(), unp_-connect2(), UNP_LOCK, UNP_LOCK_ASSERT, UNP_UNLOCK, and vput().

Referenced by uipc_connect(), and uipc_send().

Here is the call graph for this function:



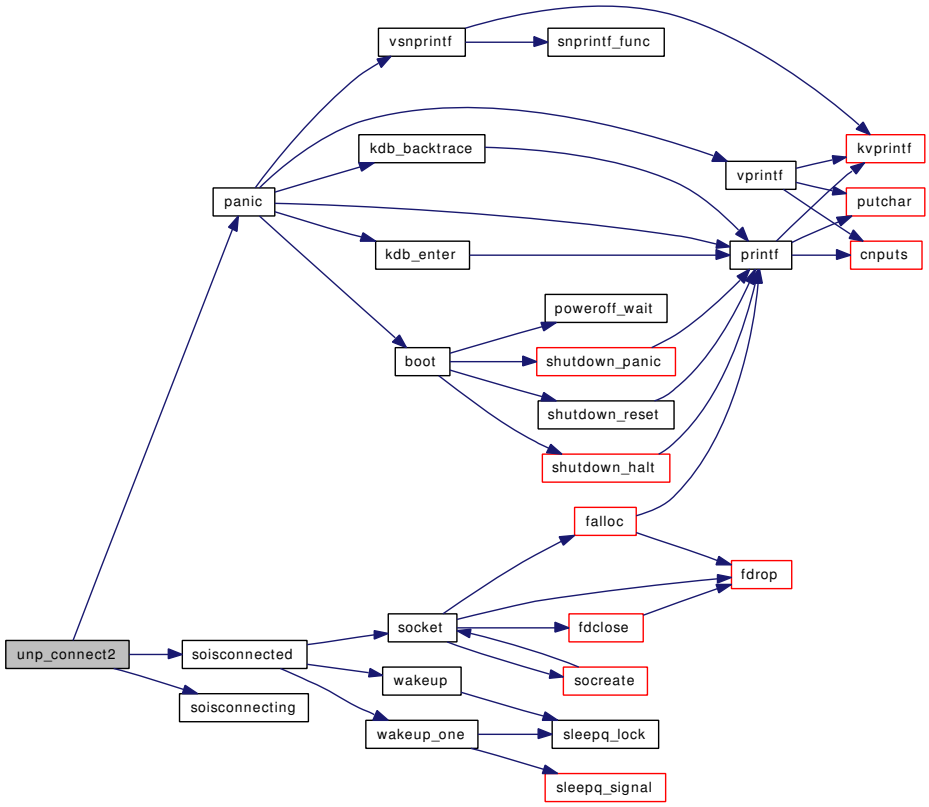
9.150.2.34 static int unp_connect2 (struct socket * *so*, struct socket * *so2*, int) [static]

Definition at line 1052 of file uipc_usrreq.c.

References panic(), soisconnected(), soisconnecting(), and UNP_LOCK_ASSERT.

Referenced by uipc_connect2(), and unp_connect().

Here is the call graph for this function:



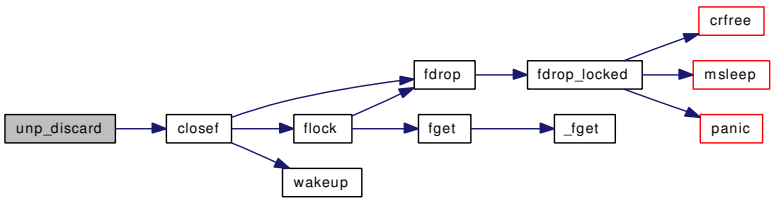
9.150.2.35 static void unp_discard (struct file *) [static]

Definition at line 1925 of file uipc_usrreq.c.

References closef(), UNP_LOCK, unp_rights, and UNP_UNLOCK.

Referenced by unp_dispose(), and unp_freerights().

Here is the call graph for this function:



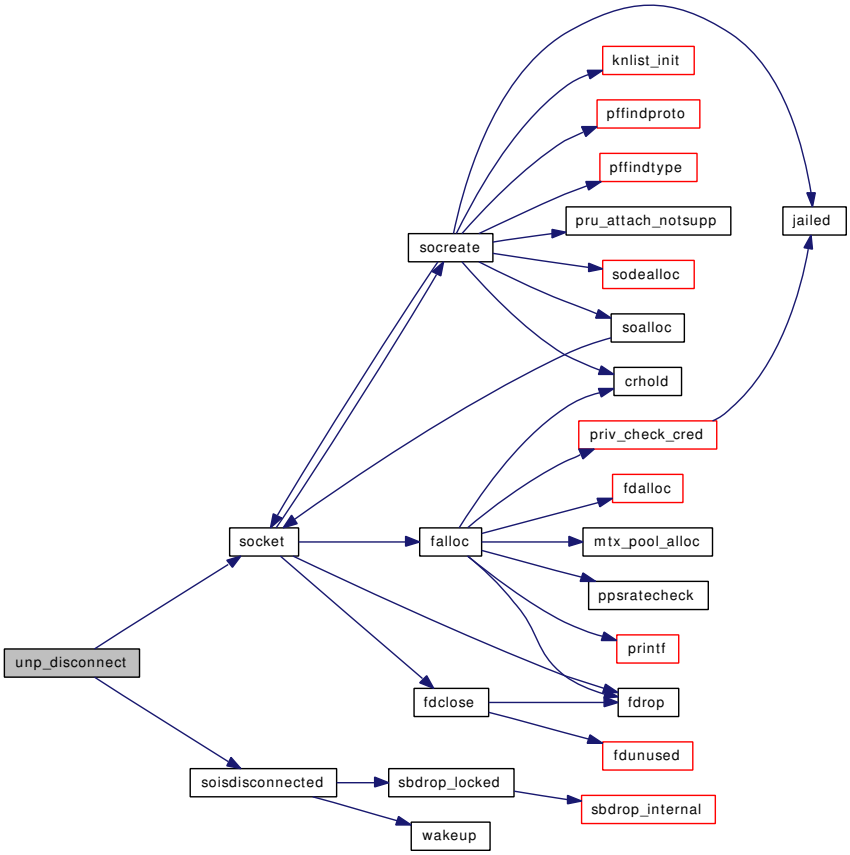
9.150.2.36 static void unp_disconnect (struct unpcb *) [static]

Definition at line 1087 of file uipc_usrreq.c.

References socket(), soisdisconnected(), and UNP_LOCK_ASSERT.

Referenced by uipc_close(), uipc_detach(), uipc_disconnect(), uipc_send(), and unp_drop().

Here is the call graph for this function:



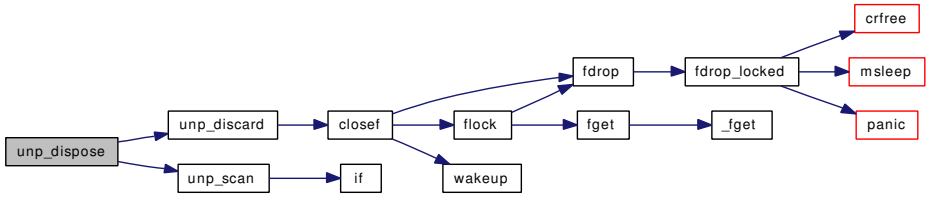
9.150.2.37 void unp_dispose (struct mbuf * *m*)

Definition at line 1840 of file uipc_usrreq.c.

References unp_discard(), and unp_scan().

Referenced by uipc_send().

Here is the call graph for this function:



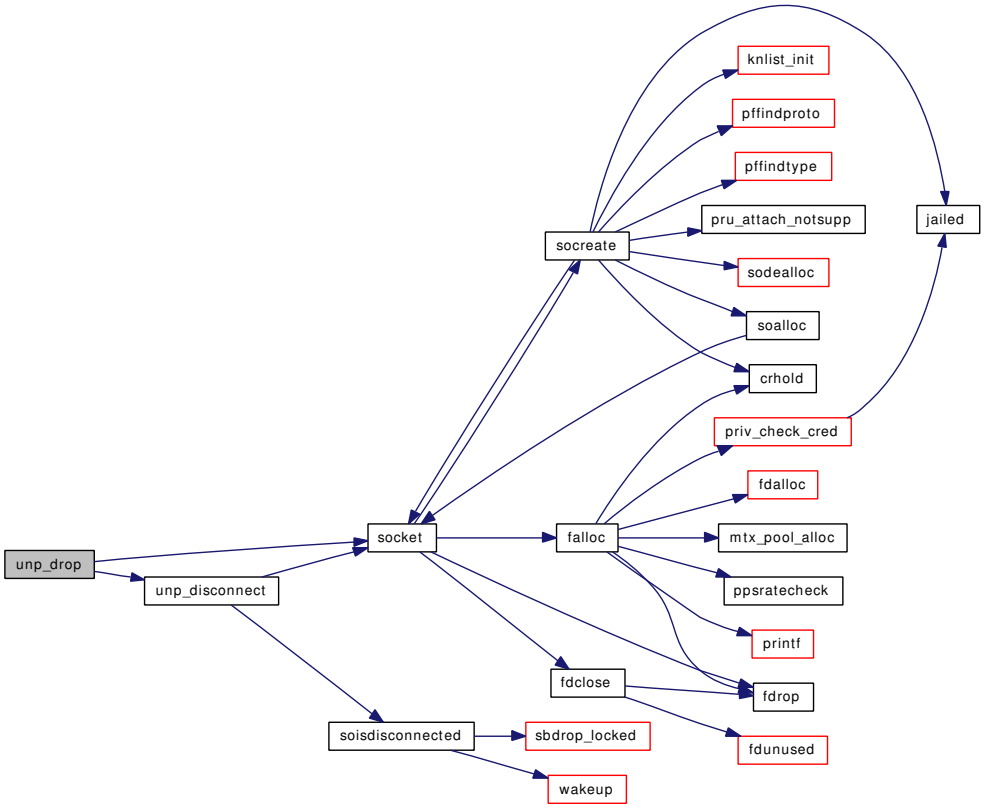
9.150.2.38 static void unp_drop (struct unpcb *, int) [static]

Definition at line 1254 of file uipc_usrreq.c.

References socket(), unp_disconnect(), and UNP_LOCK_ASSERT.

Referenced by uipc_abort(), and uipc_detach().

Here is the call graph for this function:

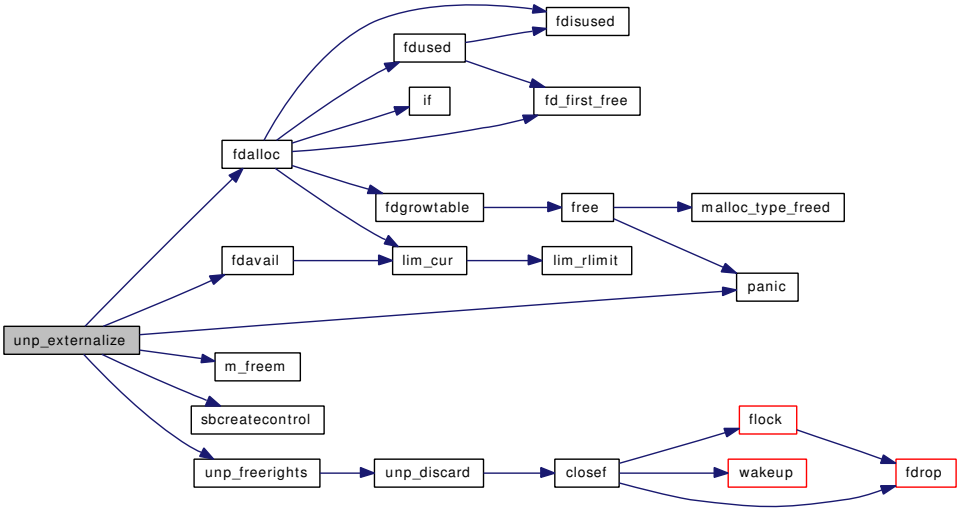


9.150.2.39 int unp_externalize (struct mbuf * *control*, struct mbuf ** *controlp*)

Definition at line 1284 of file uipc_usrreq.c.

References fdalloc(), fdavail(), m_freem(), panic(), sbcreatecontrol(), td, unp_freerights(), unp_rights, and UNP_UNLOCK_ASSERT.

Here is the call graph for this function:



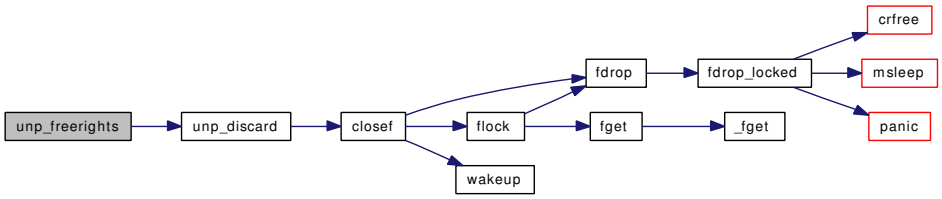
9.150.2.40 static void unp_freerights (struct file **, int) [static]

Definition at line 1265 of file uipc_usrreq.c.

References unp_discard().

Referenced by unp_externalize().

Here is the call graph for this function:



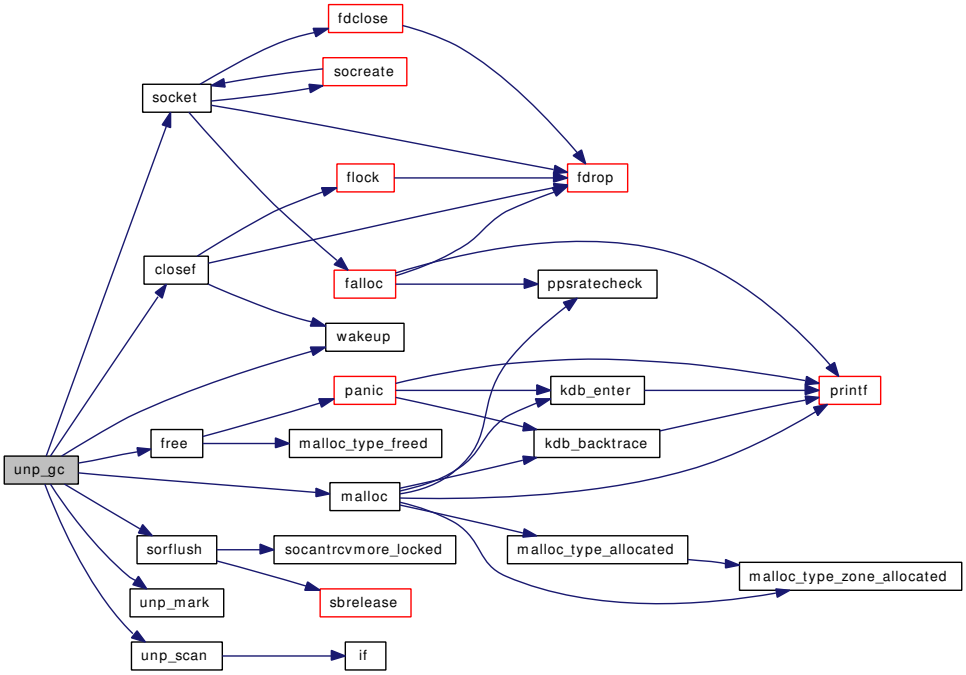
9.150.2.41 static void unp_gc (__unused void *, int) [static]

Definition at line 1627 of file uipc_usrreq.c.

References closef(), filehead, filelist_lock, free(), malloc(), openfiles, socket(), sorflush(), unp_mark(), unp_scan(), and wakeup().

Referenced by unp_init().

Here is the call graph for this function:

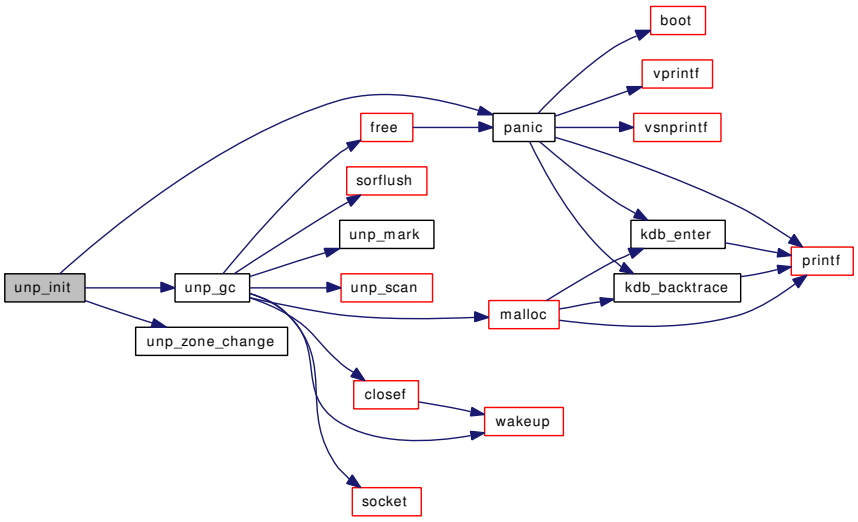


9.150.2.42 void unp_init (void)

Definition at line 1402 of file uipc_usrreq.c.

References maxsockets, panic(), unp_dhead, unp_gc(), UNP_LOCK_INIT, unp_shead, unp_zone, and unp_zone_change().

Here is the call graph for this function:



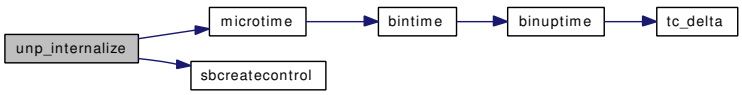
9.150.2.43 static int unp_internalize (struct mbuf **, struct thread *) [static]

Definition at line 1419 of file uipc_usrreq.c.

References microtime(), sbcreatecontrol(), unp_rights, and UNP_UNLOCK_ASSERT.

Referenced by uipc_send().

Here is the call graph for this function:



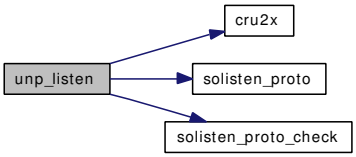
9.150.2.44 static int unp_listen (struct socket *, struct unpcb *, int, struct thread *) [static]

Definition at line 1848 of file uipc_usrreq.c.

References cru2x(), solisten_proto(), solisten_proto_check(), and UNP_LOCK_ASSERT.

Referenced by uipc_listen().

Here is the call graph for this function:



9.150.2.45 static void unp_mark (struct file *) [static]

Definition at line 1916 of file uipc_usrreq.c.

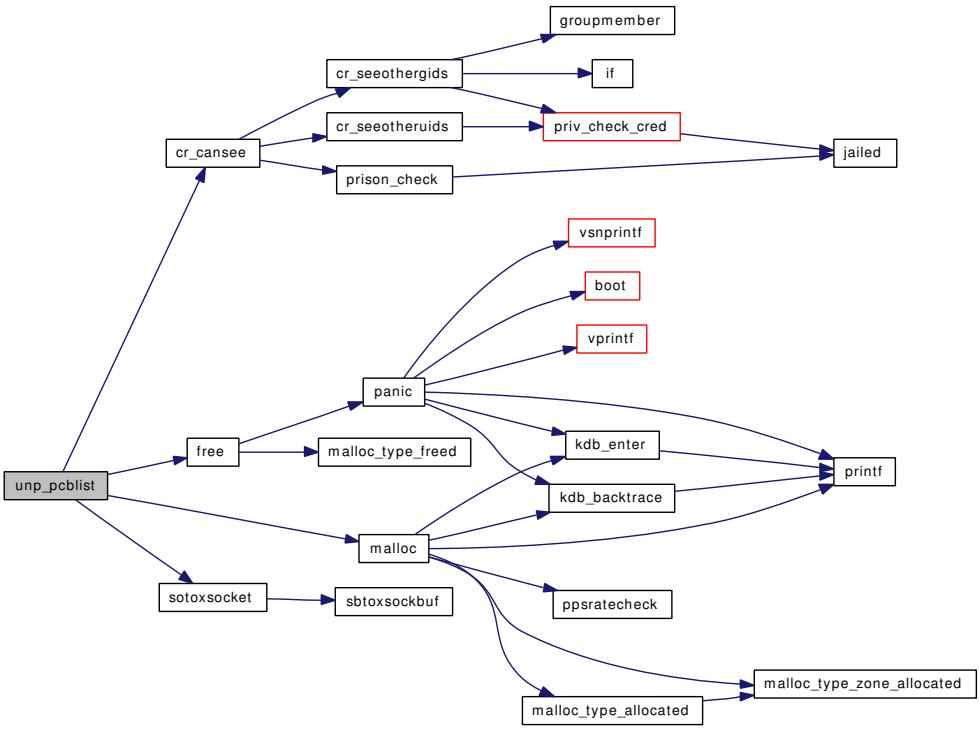
Referenced by unp_gc().

9.150.2.46 static int unp_pcblist (SYSCTL_HANDLER_ARGS) [static]

Definition at line 1123 of file uipc_usrreq.c.

References cr_cansee(), free(), malloc(), so_gencnt, sotoxsocket(), unp_count, unp_dhead, unp_gencnt, UNP_LOCK, unp_shead, UNP_UNLOCK, and unp_zone.

Here is the call graph for this function:



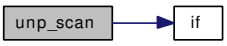
9.150.2.47 static void unp_scan (struct mbuf *, void(*)(struct file *)) [static]

Definition at line 1867 of file uipc_usrreq.c.

References if().

Referenced by unp_dispose(), and unp_gc().

Here is the call graph for this function:



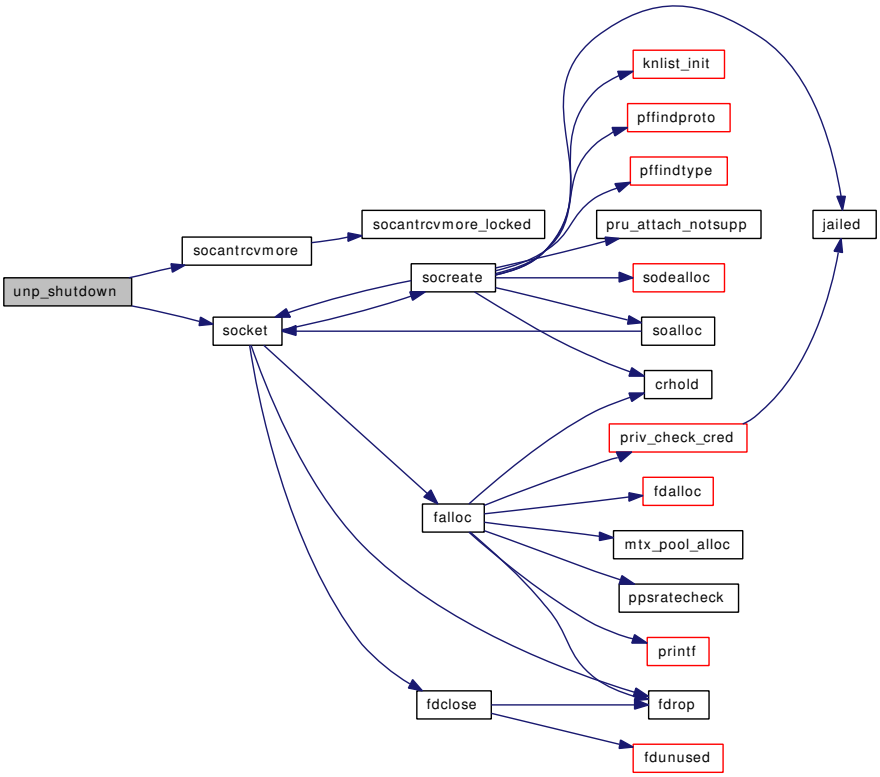
9.150.2.48 static void unp_shutdown (struct unpcb *) [static]

Definition at line 1242 of file uipc_usrreq.c.

References socantrcvmore(), socket(), and UNP_LOCK_ASSERT.

Referenced by uipc_send(), and uipc_shutdown().

Here is the call graph for this function:



9.150.2.49 static void unp_zone_change (void * *tag*) [static]

Definition at line 1395 of file uipc_usrreq.c.

References maxsockets, and unp_zone.

Referenced by unp_init().

9.150.3 Variable Documentation

9.150.3.1 struct domain localdomain [static]

Initial value:

```
{
        .dom_family =           AF_LOCAL,
        .dom_name =             "local",
        .dom_init =             unp_init,
        .dom_externalize =      unp_externalize,
        .dom_dispose =          unp_dispose,
        .dom_protosw =          localsw,
        .dom_protoswNPROTOSW =  &localsw[sizeof(localsw)/sizeof(localsw[0])]
}
```

Definition at line 210 of file uipc_usrreq.c.

9.150.3.2 struct domain localdomain [static]

Definition at line 193 of file uipc_usrreq.c.

9.150.3.3 struct protosw localsw[ ] [static]

Initial value:

```
 {
{
        .pr_type =              SOCK_STREAM,
        .pr_domain =            &localdomain,
        .pr_flags =             PR_CONNREQUIRED|PR_WANTRCVD|PR_RIGHTS,
        .pr_ctloutput =         &uipc_ctloutput,
        .pr_usrreqs =           &uipc_usrreqs
},
{
        .pr_type =              SOCK_DGRAM,
        .pr_domain =            &localdomain,
        .pr_flags =             PR_ATOMIC|PR_ADDR|PR_RIGHTS,
        .pr_usrreqs =           &uipc_usrreqs
},
}
```

Definition at line 194 of file uipc_usrreq.c.

9.150.3.4 struct sockaddr sun_noname = { sizeof(sun_noname), AF_LOCAL } [static]

Definition at line 105 of file uipc_usrreq.c.

Referenced by uipc_accept(), uipc_peeraddr(), uipc_send(), and uipc_sockaddr().

9.150.3.5 struct pr_usrreqs uipc_usrreqs

Initial value:

```
 {
        .pru_abort =            uipc_abort,
        .pru_accept =           uipc_accept,
        .pru_attach =           uipc_attach,
        .pru_bind =             uipc_bind,
        .pru_connect =          uipc_connect,
        .pru_connect2 =         uipc_connect2,
        .pru_detach =           uipc_detach,
        .pru_disconnect =       uipc_disconnect,
        .pru_listen =           uipc_listen,
        .pru_peeraddr =         uipc_peeraddr,
        .pru_rcvd =             uipc_rcvd,
        .pru_send =             uipc_send,
        .pru_sense =            uipc_sense,
        .pru_shutdown =         uipc_shutdown,
        .pru_sockaddr =         uipc_sockaddr,
        .pru_close =            uipc_close,
}
```

Definition at line 820 of file uipc_usrreq.c.

9.150.3.6 u_int unp_count [static]

Definition at line 99 of file uipc_usrreq.c.

Referenced by uipc_attach(), uipc_detach(), and unp_pcblist().

9.150.3.7 int unp_defer [static]

Definition at line 1618 of file uipc_usrreq.c.

9.150.3.8 struct unp_head unp_dhead [static]

Definition at line 103 of file uipc_usrreq.c.

Referenced by uipc_attach(), unp_init(), and unp_pcblist().

9.150.3.9 struct task unp_gc_task [static]

Definition at line 113 of file uipc_usrreq.c.

9.150.3.10 unp_gen_t unp_gencnt [static]

Definition at line 98 of file uipc_usrreq.c.

Referenced by uipc_attach(), uipc_detach(), and unp_pcblist().

9.150.3.11 ino_t unp_ino [static]

Definition at line 100 of file uipc_usrreq.c.

Referenced by uipc_sense().

9.150.3.12 struct mtx unp_mtx [static]

Definition at line 166 of file uipc_usrreq.c.

9.150.3.13 int unp_recycled [static]

Definition at line 1623 of file uipc_usrreq.c.

9.150.3.14 int unp_rights [static]

Definition at line 101 of file uipc_usrreq.c.

Referenced by uipc_detach(), unp_discard(), unp_externalize(), and unp_internalize().

9.150.3.15 struct unp_head unp_shead [static]

Definition at line 102 of file uipc_usrreq.c.

Referenced by uipc_attach(), unp_init(), and unp_pcblist().

9.150.3.16 int unp_taskcount [static]

Definition at line 1620 of file uipc_usrreq.c.

9.150.3.17 uma_zone_t unp_zone [static]

Definition at line 97 of file uipc_usrreq.c.

Referenced by uipc_attach(), uipc_detach(), unp_init(), unp_pcblist(), and unp_zone_change().

9.150.3.18 u_long unpdg_recvspace = 4∗1024 [static]

Definition at line 130 of file uipc_usrreq.c.

9.150.3.19 u_long unpdg_sendspace = 2∗1024 [static]

Definition at line 129 of file uipc_usrreq.c.

9.150.3.20 u_long unpst_recvspace = PIPSIZ [static]

Definition at line 128 of file uipc_usrreq.c.

9.150.3.21 u_long unpst_sendspace = PIPSIZ [static]

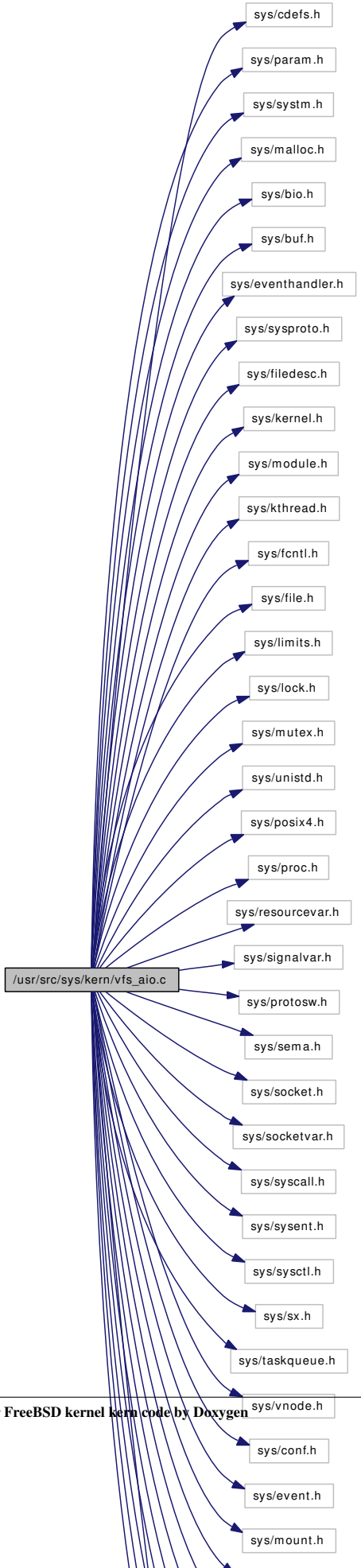
Definition at line 127 of file uipc_usrreq.c.

9.151 /usr/src/sys/kern/vfs_aio.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/malloc.h>
#include <sys/bio.h>
#include <sys/buf.h>
#include <sys/eventhandler.h>
#include <sys/sysproto.h>
#include <sys/filedesc.h>
#include <sys/kernel.h>
#include <sys/module.h>
#include <sys/kthread.h>
#include <sys/fcntl.h>
#include <sys/file.h>
#include <sys/limits.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/unistd.h>
#include <sys/posix4.h>
#include <sys/proc.h>
#include <sys/resourcevar.h>
#include <sys/signalvar.h>
#include <sys/protosw.h>
#include <sys/sema.h>
#include <sys/socket.h>
#include <sys/socketvar.h>
#include <sys/syscall.h>
#include <sys/sysent.h>
#include <sys/sysctl.h>
#include <sys/sx.h>
#include <sys/taskqueue.h>
#include <sys/vnode.h>
#include <sys/conf.h>
#include <sys/event.h>
#include <sys/mount.h>
```

```
#include <machine/atomic.h>
#include <vm/vm.h>
#include <vm/vm_extern.h>
#include <vm/pmap.h>
#include <vm/vm_map.h>
#include <vm/vm_object.h>
#include <vm/uma.h>
#include <sys/aio.h>
#include "opt_vfs_aio.h"
```

Include dependency graph for vfs_aio.c:


/usr/src/sys/kern/vfs_aio.c
sys/cdefs.h
sys/param.h
sys/systm.h
sys/malloc.h
sys/bio.h
sys/buf.h
sys/eventhandler.h
sys/sysproto.h
sys/filedesc.h
sys/kernel.h
sys/module.h
sys/kthread.h
sys/fcntl.h
sys/file.h
sys/limits.h
sys/lock.h
sys/mutex.h
sys/unistd.h
sys/posix4.h
sys/proc.h
sys/resourcevar.h
sys/signalvar.h
sys/protosw.h
sys/sema.h
sys/socket.h
sys/socketvar.h
sys/syscall.h
sys/sysent.h
sys/sysctl.h
sys/sx.h
sys/taskqueue.h
sys/vnode.h
sys/conf.h
sys/event.h
sys/mount.h

Data Structures

- struct oaiocb
- struct aiocblist
- struct aiothreadlist
- struct aioliojob
- struct kaioinfo

Defines

- #define JOBST_NULL 0
- #define JOBST_JOBQSOCK 1
- #define JOBST_JOBQGLOBAL 2
- #define JOBST_JOBRUNNING 3
- #define JOBST_JOBFINISHED 4
- #define JOBST_JOBQBUF 5
- #define JOBST_JOBQSYNC 6
- #define MAX_AIO_PER_PROC 32
- #define MAX_AIO_QUEUE_PER_PROC 256
- #define MAX_AIO_PROCS 32
- #define MAX_AIO_QUEUE 1024
- #define TARGET_AIO_PROCS 4
- #define MAX_BUF_AIO 16
- #define AIOD_TIMEOUT_DEFAULT (10 * hz)
- #define AIOD_LIFETIME_DEFAULT (30 * hz)
- #define AIOCBLIST_DONE 0x01
- #define AIOCBLIST_BUFDONE 0x02
- #define AIOCBLIST_RUNDOWN 0x04
- #define AIOCBLIST_CHECKSYNC 0x08
- #define AIOP_FREE 0x1
- #define LIOJ_SIGNAL 0x1
- #define LIOJ_SIGNAL_POSTED 0x2
- #define LIOJ_KEVENT_POSTED 0x4
- #define AIO_LOCK(ki) mtx_lock(&(ki) → kaio_mtx)
- #define AIO_UNLOCK(ki) mtx_unlock(&(ki) → kaio_mtx)
- #define AIO_LOCK_ASSERT(ki, f) mtx_assert(&(ki) → kaio_mtx, (f))
- #define AIO_MTX(ki) (&(ki) → kaio_mtx)
- #define KAIO_RUNDOWN 0x1
- #define KAIO_WAKEUP 0x2
- #define DONE_BUF 1
- #define DONE_QUEUE 2

Typedefs

- typedef oaiocb oaiocb_t

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/vfs_aio.c,v 1.229 2006/11/11 16:26:56 trhodes Exp $")
- static SYSCTL_NODE (_vfs, OID_AUTO, aio, CTLFLAG_RW, 0,"Async IO management")
- SYSCTL_INT (_vfs_aio, OID_AUTO, max_aio_procs, CTLFLAG_RW,&max_aio_procs, 0,"Maximum number of kernel threads to use for handling async IO ")
- SYSCTL_INT (_vfs_aio, OID_AUTO, num_aio_procs, CTLFLAG_RD,&num_aio_procs, 0,"Number of presently active kernel threads for async IO")
- SYSCTL_INT (_vfs_aio, OID_AUTO, target_aio_procs, CTLFLAG_RW,&target_aio_procs, 0,"Preferred number of ready kernel threads for async IO")
- SYSCTL_INT (_vfs_aio, OID_AUTO, max_aio_queue, CTLFLAG_RW,&max_queue_count, 0,"Maximum number of aio requests to queue, globally")
- SYSCTL_INT (_vfs_aio, OID_AUTO, num_queue_count, CTLFLAG_RD,&num_queue_count, 0,"Number of queued aio requests")
- SYSCTL_INT (_vfs_aio, OID_AUTO, num_buf_aio, CTLFLAG_RD,&num_buf_aio, 0,"Number of aio requests presently handled by the buf subsystem")
- SYSCTL_INT (_vfs_aio, OID_AUTO, aiod_timeout, CTLFLAG_RW,&aiod_timeout, 0,"Timeout value for synchronous aio operations")
- SYSCTL_INT (_vfs_aio, OID_AUTO, aiod_lifetime, CTLFLAG_RW,&aiod_lifetime, 0,"Maximum lifetime for idle aiod")
- SYSCTL_INT (_vfs_aio, OID_AUTO, unloadable, CTLFLAG_RW,&unloadable, 0,"Allow unload of aio (not recommended)")
- SYSCTL_INT (_vfs_aio, OID_AUTO, max_aio_per_proc, CTLFLAG_RW,&max_aio_per_proc, 0,"Maximum active aio requests per process (stored in the process)")
- SYSCTL_INT (_vfs_aio, OID_AUTO, max_aio_queue_per_proc, CTLFLAG_RW,&max_aio_queue_per_proc, 0,"Maximum queued aio requests per process (stored in the process)")
- SYSCTL_INT (_vfs_aio, OID_AUTO, max_buf_aio, CTLFLAG_RW,&max_buf_aio, 0,"Maximum buf aio requests per process (stored in the process)")
- static TAILQ_HEAD (aiothreadlist)
- TASKQUEUE_DEFINE_THREAD (aiod_bio)
- static int aio_modload (struct module *module, int cmd, void *arg)
- SYSCALL_MODULE_HELPER (aio_cancel)
- SYSCALL_MODULE_HELPER (aio_error)
- SYSCALL_MODULE_HELPER (aio_fsync)
- SYSCALL_MODULE_HELPER (aio_read)
- SYSCALL_MODULE_HELPER (aio_return)
- SYSCALL_MODULE_HELPER (aio_suspend)
- SYSCALL_MODULE_HELPER (aio_waitcomplete)
- SYSCALL_MODULE_HELPER (aio_write)
- SYSCALL_MODULE_HELPER (lio_listio)
- SYSCALL_MODULE_HELPER (oaio_read)
- SYSCALL_MODULE_HELPER (oaio_write)
- SYSCALL_MODULE_HELPER (olio_listio)
- DECLARE_MODULE (aio, aio_mod, SI_SUB_VFS, SI_ORDER_ANY)
- MODULE_VERSION (aio, 1)
- static void aio_onceonly (void)
- static int aio_unload (void)
- void aio_init_aioinfo (struct proc *p)
- static int aio_sendsig (struct proc *p, struct sigevent *sigev, ksiginfo_t *ksi)
- static int aio_free_entry (struct aiocblist *aiocbe)
- static void aio_proc_rundown_exec (void *arg, struct proc *p, struct image_params *imgp __unused)

- static void aio_proc_rundown (void *arg, struct proc *p)
- static struct aiocblist * aio_selectjob (struct aiothreadlist *aiop)
- static int aio_fsync_vnode (struct thread *td, struct vnode *vp)
- static void aio_process (struct aiocblist *aiocbe)
- static void aio_bio_done_notify (struct proc *userp, struct aiocblist *aiocbe, int type)
- static void aio_daemon (void *_id)
- static int aio_newproc (int *start)
- static int aio_qphysio (struct proc *p, struct aiocblist *aiocbe)
- static void aio_swake_cb (struct socket *so, struct sockbuf *sb)
- int aio_aqueue (struct thread *td, struct aiocb *job, struct aioliojob *lj, int type, int oldsigev)
- static void aio_kick_nowait (struct proc *userp)
- static int aio_kick (struct proc *userp)
- static void aio_kick_helper (void *context, int pending)
- int aio_return (struct thread *td, struct aio_return_args *uap)
- int aio_suspend (struct thread *td, struct aio_suspend_args *uap)
- int aio_cancel (struct thread *td, struct aio_cancel_args *uap)
- int aio_error (struct thread *td, struct aio_error_args *uap)
- int oaio_read (struct thread *td, struct oaio_read_args *uap)
- int aio_read (struct thread *td, struct aio_read_args *uap)
- int oaio_write (struct thread *td, struct oaio_write_args *uap)
- int aio_write (struct thread *td, struct aio_write_args *uap)
- int olio_listio (struct thread *td, struct olio_listio_args *uap)
- int lio_listio (struct thread *td, struct lio_listio_args *uap)
- static int do_lio_listio (struct thread *td, struct lio_listio_args *uap, int oldsigev)
- static void aio_physwakeup (struct buf *bp)
- static void biohelper (void *context, int pending)
- int aio_waitcomplete (struct thread *td, struct aio_waitcomplete_args *uap)
- int aio_fsync (struct thread *td, struct aio_fsync_args *uap)
- static int filt_aioattach (struct knote *kn)
- static void filt_aiodetach (struct knote *kn)
- static int filt_aio (struct knote *kn, long hint)
- static int filt_lioattach (struct knote *kn)
- static void filt_liodetach (struct knote *kn)
- static int filt_lio (struct knote *kn, long hint)

Variables

- static u_long jobrefid
- static uint64_t jobseqno
- static int max_aio_procs = MAX_AIO_PROCS
- static int num_aio_procs = 0
- static int target_aio_procs = TARGET_AIO_PROCS
- static int max_queue_count = MAX_AIO_QUEUE
- static int num_queue_count = 0
- static int num_buf_aio = 0
- static int num_aio_resv_start = 0
- static int aiod_timeout
- static int aiod_lifetime
- static int unloadable = 0
- static int max_aio_per_proc = MAX_AIO_PER_PROC

- static int max_aio_queue_per_proc = MAX_AIO_QUEUE_PER_PROC
- static int max_buf_aio = MAX_BUF_AIO
- static struct filterops lio_filtops
- static eventhandler_tag exit_tag
- static eventhandler_tag exec_tag
- static moduledata_t aio_mod

9.151.1 Define Documentation

9.151.1.1 #define AIO_LOCK(ki) mtx_lock(&(ki) → kaio_mtx)

Definition at line 301 of file vfs_aio.c.

Referenced by aio_aqueue(), aio_cancel(), aio_daemon(), aio_error(), aio_free_entry(), aio_proc_rundown(), aio_qphysio(), aio_return(), aio_suspend(), aio_waitcomplete(), biohelper(), and do_lio_listio().

9.151.1.2 #define AIO_LOCK_ASSERT(ki, f) mtx_assert(&(ki) → kaio_mtx, (f))

Definition at line 303 of file vfs_aio.c.

Referenced by aio_bio_done_notify(), and aio_free_entry().

9.151.1.3 #define AIO_MTX(ki) (&(ki) → kaio_mtx)

Definition at line 304 of file vfs_aio.c.

Referenced by aio_aqueue(), aio_proc_rundown(), aio_suspend(), aio_waitcomplete(), and do_lio_listio().

9.151.1.4 #define AIO_UNLOCK(ki) mtx_unlock(&(ki) → kaio_mtx)

Definition at line 302 of file vfs_aio.c.

Referenced by aio_aqueue(), aio_cancel(), aio_daemon(), aio_error(), aio_free_entry(), aio_proc_rundown(), aio_qphysio(), aio_return(), aio_waitcomplete(), biohelper(), and do_lio_listio().

9.151.1.5 #define AIOCBLIST_BUFDONE 0x02

Definition at line 245 of file vfs_aio.c.

Referenced by aio_bio_done_notify().

9.151.1.6 #define AIOCBLIST_CHECKSYNC 0x08

Definition at line 247 of file vfs_aio.c.

Referenced by aio_aqueue(), and aio_bio_done_notify().

9.151.1.7 #define AIOCBLIST_DONE 0x01

Definition at line 244 of file vfs_aio.c.

Referenced by aio_bio_done_notify().

9.151.1.8 #define AIOCBLIST_RUNDOWN 0x04

Definition at line 246 of file vfs_aio.c.

9.151.1.9 #define AIOD_LIFETIME_DEFAULT (30 ∗ hz)

Definition at line 119 of file vfs_aio.c.

Referenced by aio_onceonly().

9.151.1.10 #define AIOD_TIMEOUT_DEFAULT (10 ∗ hz)

Definition at line 115 of file vfs_aio.c.

Referenced by aio_onceonly().

9.151.1.11 #define AIOP_FREE 0x1

Definition at line 252 of file vfs_aio.c.

Referenced by aio_daemon(), aio_kick(), and aio_kick_nowait().

9.151.1.12 #define DONE_BUF 1

Referenced by biohelper().

9.151.1.13 #define DONE_QUEUE 2

Referenced by aio_bio_done_notify(), aio_cancel(), aio_daemon(), and aio_proc_rundown().

9.151.1.14 #define JOBST_JOBFINISHED 4

Definition at line 86 of file vfs_aio.c.

Referenced by aio_bio_done_notify(), aio_error(), aio_free_entry(), aio_proc_rundown(), aio_return(), aio_suspend(), aio_waitcomplete(), and filt_aio().

9.151.1.15 #define JOBST_JOBQBUF 5

Definition at line 87 of file vfs_aio.c.

Referenced by aio_qphysio().

9.151.1.16 #define JOBST_JOBQGLOBAL 2

Definition at line 84 of file vfs_aio.c.

Referenced by aio_aqueue(), aio_bio_done_notify(), aio_cancel(), and aio_proc_rundown().

9.151.1.17 #define JOBST_JOBQSOCK 1

Definition at line 83 of file vfs_aio.c.

Referenced by aio_aqueue(), aio_cancel(), aio_proc_rundown(), and aio_swake_cb().

9.151.1.18 #define JOBST_JOBQSYNC 6

Definition at line 88 of file vfs_aio.c.

Referenced by aio_aqueue(), aio_cancel(), and aio_proc_rundown().

9.151.1.19 #define JOBST_JOBRUNNING 3

Definition at line 85 of file vfs_aio.c.

Referenced by aio_selectjob().

9.151.1.20 #define JOBST_NULL 0

Definition at line 82 of file vfs_aio.c.

Referenced by aio_free_entry().

9.151.1.21 #define KAIO_RUNDOWN 0x1

Definition at line 306 of file vfs_aio.c.

Referenced by aio_bio_done_notify(), and aio_proc_rundown().

9.151.1.22 #define KAIO_WAKEUP 0x2

Definition at line 307 of file vfs_aio.c.

Referenced by aio_bio_done_notify(), aio_proc_rundown(), aio_suspend(), aio_waitcomplete(), and do_-lio_listio().

9.151.1.23 #define LIOJ_KEVENT_POSTED 0x4

Definition at line 275 of file vfs_aio.c.

Referenced by aio_bio_done_notify(), do_lio_listio(), and filt_lio().

9.151.1.24 #define LIOJ_SIGNAL 0x1

Definition at line 273 of file vfs_aio.c.

Referenced by aio_bio_done_notify(), and do_lio_listio().

9.151.1.25 #define LIOJ_SIGNAL_POSTED 0x2

Definition at line 274 of file vfs_aio.c.

Referenced by aio_bio_done_notify(), and do_lio_listio().

9.151.1.26 #define MAX_AIO_PER_PROC 32

Definition at line 91 of file vfs_aio.c.

9.151.1.27 #define MAX_AIO_PROCS 32

Definition at line 99 of file vfs_aio.c.

9.151.1.28 #define MAX_AIO_QUEUE 1024

Definition at line 103 of file vfs_aio.c.

Referenced by aio_onceonly().

9.151.1.29 #define MAX_AIO_QUEUE_PER_PROC 256

Definition at line 95 of file vfs_aio.c.

9.151.1.30 #define MAX_BUF_AIO 16

Definition at line 111 of file vfs_aio.c.

9.151.1.31 #define TARGET_AIO_PROCS 4

Definition at line 107 of file vfs_aio.c.

9.151.2 Typedef Documentation

9.151.2.1 typedef struct oaiocb oaiocb_t

9.151.3 Function Documentation

9.151.3.1 __FBSDID ("$FreeBSD: src/sys/kern/vfs_aio. *c*, v 1.229 2006/11/11 16:26:56 trhodes Exp $")

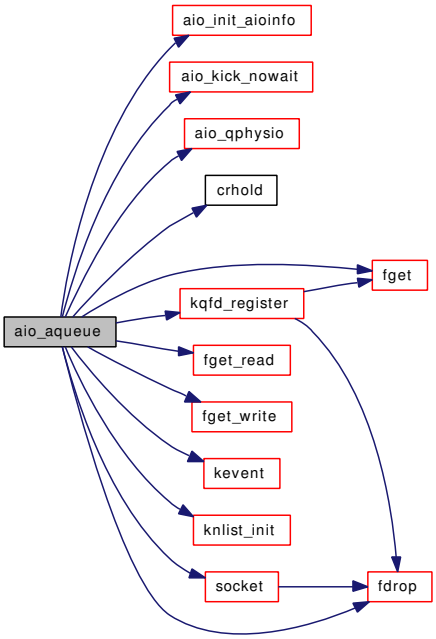
9.151.3.2 int aio_aqueue (struct thread * *td*, struct aiocb * *job*, struct aioliojob * *lj*, int *type*, int *oldsigev*)

Definition at line 1326 of file vfs_aio.c.

References aio_init_aioinfo(), aio_kick_nowait(), AIO_LOCK, AIO_MTX, aio_qphysio(), AIO_-UNLOCK, AIOCBLIST_CHECKSYNC, crhold(), fdrop(), fget(), fget_read(), fget_write(), jobrefid, jobseqno, JOBST_JOBQGLOBAL, JOBST_JOBQSOCK, JOBST_JOBQSYNC, kaioinfo::kaio_count, kaioinfo::kaio_qallowed_count, kevent(), knlist_init(), kqfd_register(), aioliojob::lioj_count, max_queue_-count, num_queue_count, socket(), and suword.

Referenced by aio_fsync(), aio_read(), aio_write(), do_lio_listio(), oaio_read(), and oaio_write().

Here is the call graph for this function:



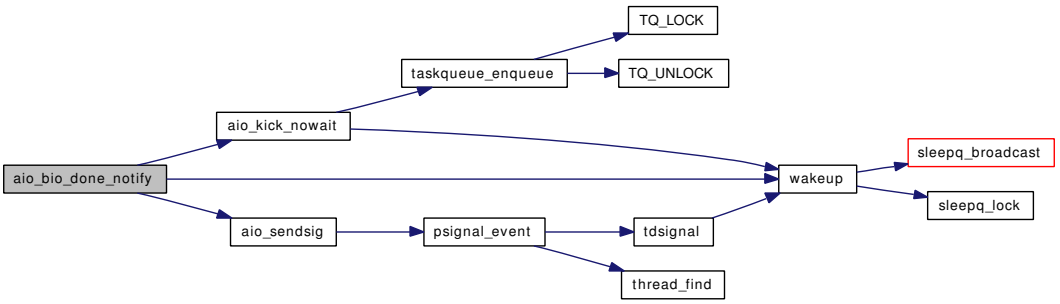
9.151.3.3 static void aio_bio_done_notify (struct proc * *userp*, struct aiocblist * *aiocbe*, int *type*) [static]

Definition at line 880 of file vfs_aio.c.

References aio_kick_nowait(), AIO_LOCK_ASSERT, aio_sendsig(), AIOCBLIST_BUFDONE, AIOCBLIST_CHECKSYNC, AIOCBLIST_DONE, DONE_QUEUE, JOBST_JOBFINISHED, JOBST_JOBQGLOBAL, kaioinfo::kaio_flags, KAIO_RUNDOWN, KAIO_WAKEUP, aioliojob::lioj_-count, aioliojob::lioj_finished_count, aioliojob::lioj_flags, LIOJ_KEVENT_POSTED, LIOJ_SIGNAL, aioliojob::lioj_signal, LIOJ_SIGNAL_POSTED, and wakeup().

Referenced by aio_cancel(), aio_daemon(), aio_proc_rundown(), and biohelper().

Here is the call graph for this function:

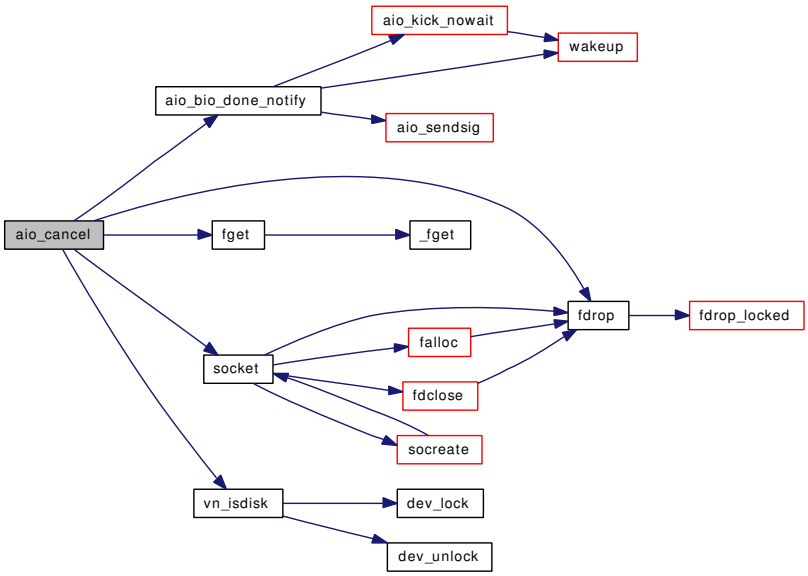


9.151.3.4 int aio_cancel (struct thread * *td*, struct aio_cancel_args * *uap*)

Definition at line 1778 of file vfs_aio.c.

References aio_bio_done_notify(), AIO_LOCK, AIO_UNLOCK, DONE_QUEUE, fdrop(), fget(), JOBST_JOBQGLOBAL, JOBST_JOBQSOCK, JOBST_JOBQSYNC, socket(), and vn_isdisk().

Here is the call graph for this function:



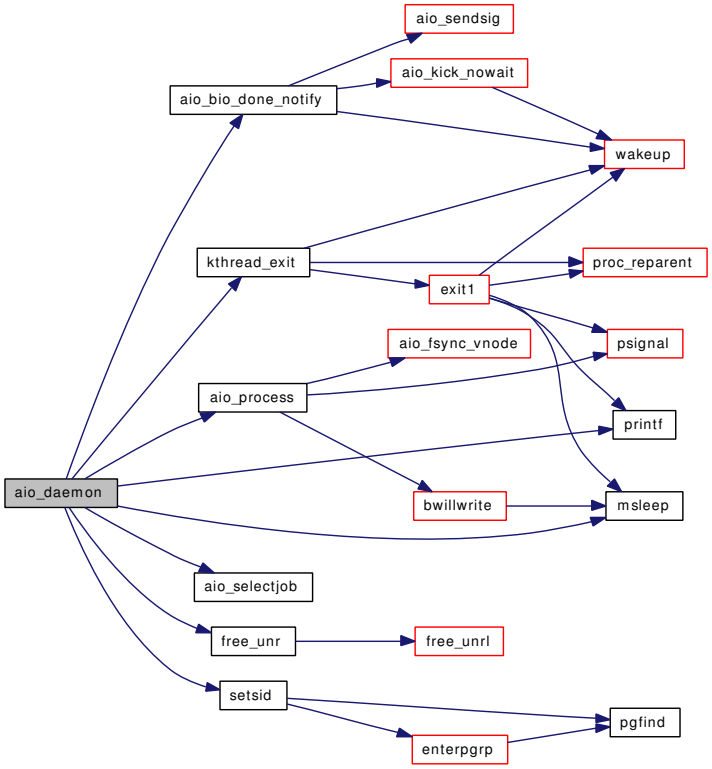
9.151.3.5 static void aio_daemon (void * *_id*) [static]

Definition at line 954 of file vfs_aio.c.

References aio_bio_done_notify(), AIO_LOCK, aio_process(), aio_selectjob(), AIO_UNLOCK, aiod_-lifetime, AIOP_FREE, aiothreadlist::aiothreadflags, DONE_QUEUE, free_unr(), kaioinfo::kaio_active_-count, kthread_exit(), msleep(), num_aio_procs, printf(), setsid(), and target_aio_procs.

Referenced by aio_newproc().

Here is the call graph for this function:



9.151.3.6 int aio_error (struct thread * *td*, struct aio_error_args * *uap*)

Definition at line 1877 of file vfs_aio.c.

References AIO_LOCK, AIO_UNLOCK, and JOBST_JOBFINISHED.

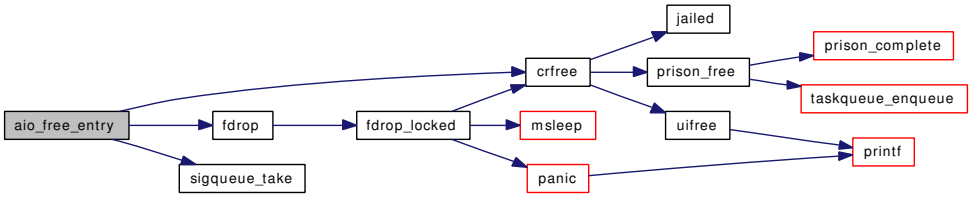
9.151.3.7 static int aio_free_entry (struct aiocblist * *aiocbe*) [static]

Definition at line 558 of file vfs_aio.c.

References AIO_LOCK, AIO_LOCK_ASSERT, AIO_UNLOCK, crfree(), fdrop(), JOBST_JOBFINISHED, JOBST_NULL, kaioinfo::kaio_count, aioliojob::lioj_count, aioliojob::lioj_finished_count, num_queue_count, and sigqueue_take().

Referenced by aio_proc_rundown(), aio_return(), and aio_waitcomplete().

Here is the call graph for this function:

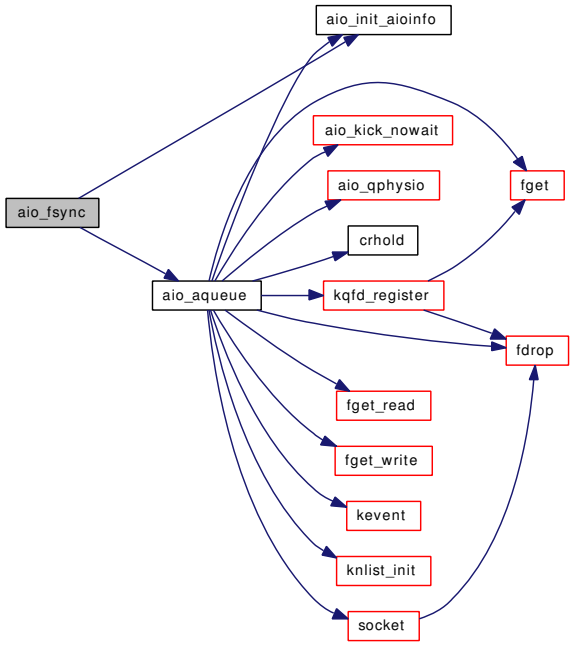


9.151.3.8 int aio_fsync (struct thread * *td*, struct aio_fsync_args * *uap*)

Definition at line 2227 of file vfs_aio.c.

References aio_aqueue(), and aio_init_aioinfo().

Here is the call graph for this function:



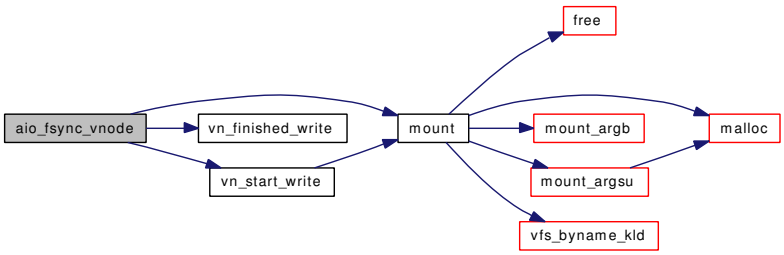
9.151.3.9 static int aio_fsync_vnode (struct thread * *td*, struct vnode * *vp*) [static]

Definition at line 757 of file vfs_aio.c.

References mount(), vn_finished_write(), and vn_start_write().

Referenced by aio_process().

Here is the call graph for this function:



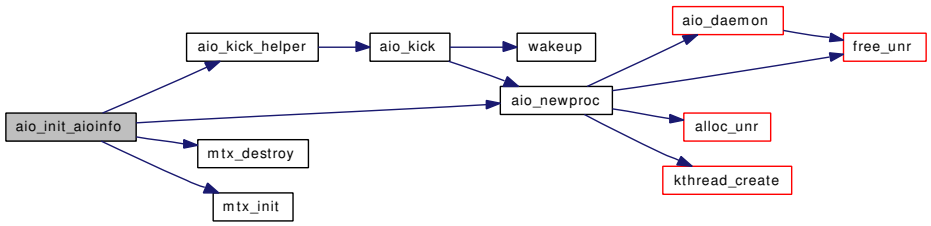
9.151.3.10 void aio_init_aioinfo (struct proc * *p*)

Definition at line 502 of file vfs_aio.c.

References aio_kick_helper(), aio_newproc(), kaioinfo::kaio_active_count, kaioinfo::kaio_ballowed_count, kaioinfo::kaio_buffer_count, kaioinfo::kaio_count, kaioinfo::kaio_flags, kaioinfo::kaio_maxactive_count, kaioinfo::kaio_mtx, kaioinfo::kaio_qallowed_count, max_aio_per_proc, max_aio_queue_per_proc, max_buf_aio, mtx_destroy(), mtx_init(), num_aio_procs, and target_aio_procs.

Referenced by aio_aqueue(), aio_fsync(), aio_waitcomplete(), and do_lio_listio().

Here is the call graph for this function:



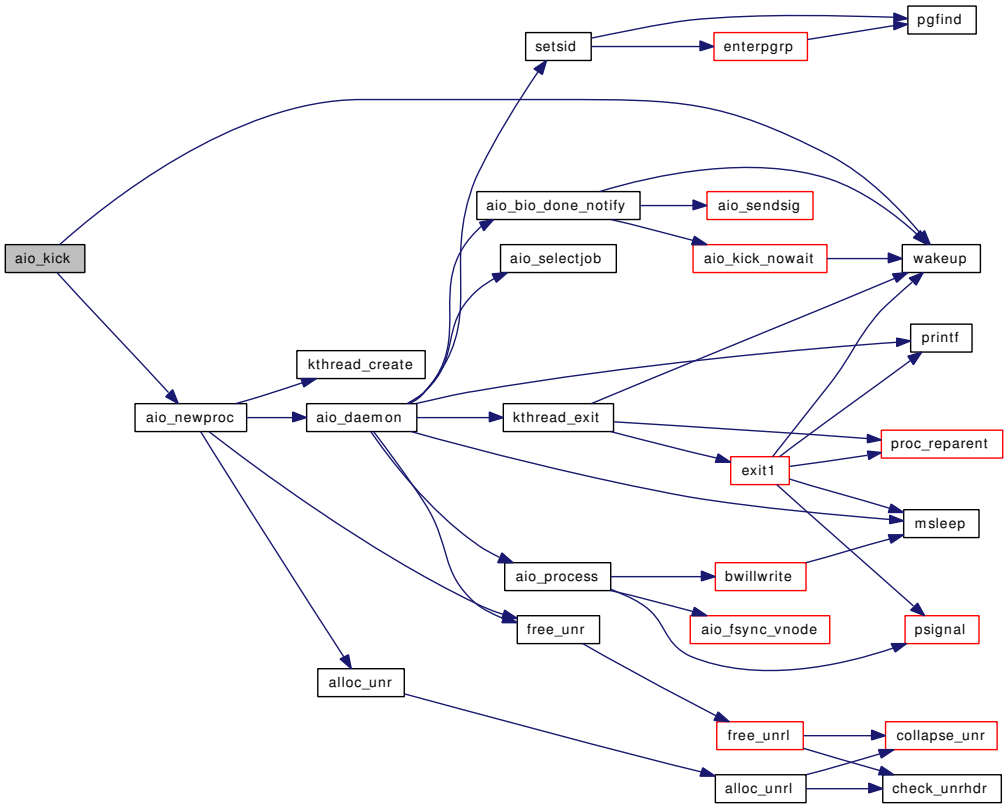
9.151.3.11 static int aio_kick (struct proc * *userp*) [static]

Definition at line 1596 of file vfs_aio.c.

References aio_newproc(), AIOP_FREE, aiothreadlist::aiothreadflags, kaioinfo::kaio_active_count, kaioinfo::kaio_maxactive_count, max_aio_procs, num_aio_procs, num_aio_resv_start, ret, and wakeup().

Referenced by aio_kick_helper().

Here is the call graph for this function:



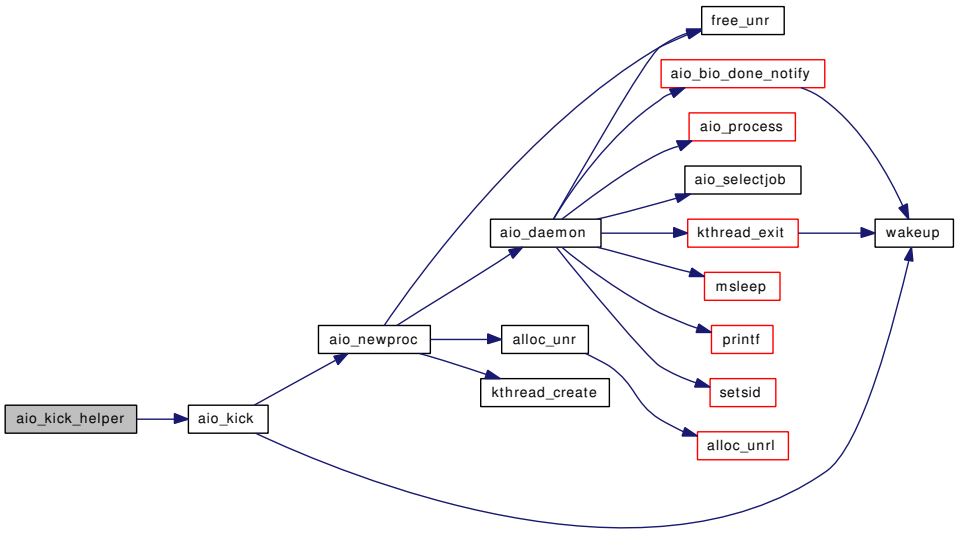
9.151.3.12 static void aio_kick_helper (void * *context*, int *pending*) [static]

Definition at line 1626 of file vfs_aio.c.

References aio_kick().

Referenced by aio_init_aioinfo().

Here is the call graph for this function:



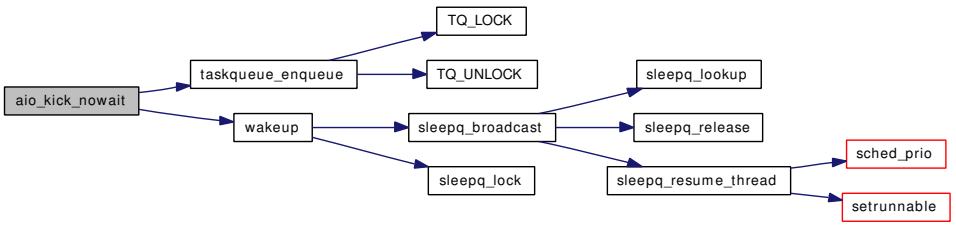
9.151.3.13 static void aio_kick_nowait (struct proc * *userp*) [static]

Definition at line 1578 of file vfs_aio.c.

References AIOP_FREE, aiothreadlist::aiothreadflags, kaioinfo::kaio_active_count, kaioinfo::kaio_-maxactive_count, max_aio_procs, num_aio_procs, num_aio_resv_start, taskqueue_enqueue(), and wakeup().

Referenced by aio_aqueue(), aio_bio_done_notify(), and aio_swake_cb().

Here is the call graph for this function:

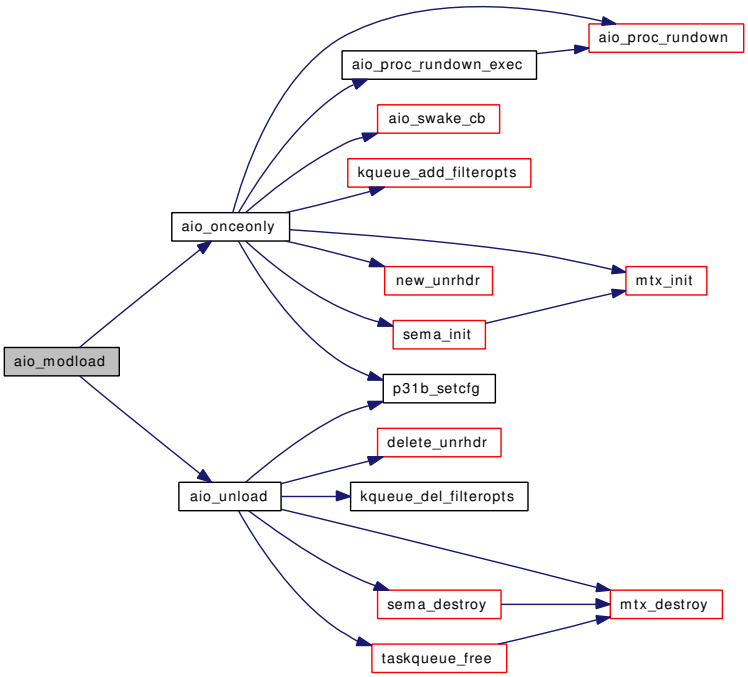


9.151.3.14 static int aio_modload (struct module * *module*, int *cmd*, void * *arg*) [static]

Definition at line 369 of file vfs_aio.c.

References aio_onceonly(), and aio_unload().

Here is the call graph for this function:



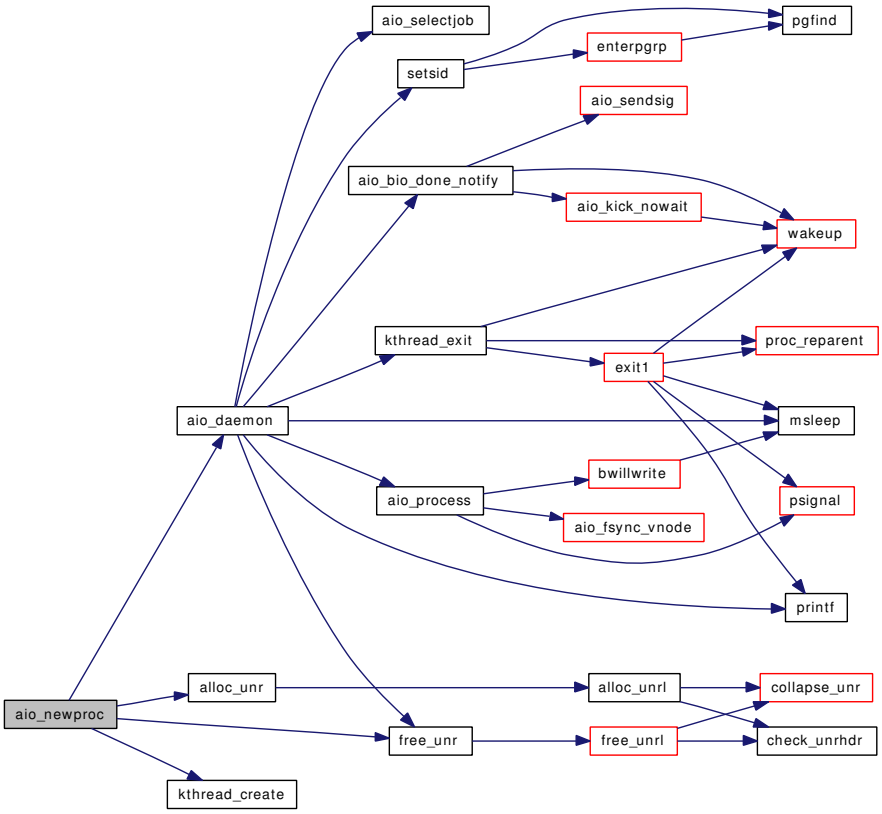
9.151.3.15 static int aio_newproc (int * *start*) [static]

Definition at line 1137 of file vfs_aio.c.

References aio_daemon(), alloc_unr(), free_unr(), kthread_create(), and num_aio_procs.

Referenced by aio_init_aioinfo(), and aio_kick().

Here is the call graph for this function:



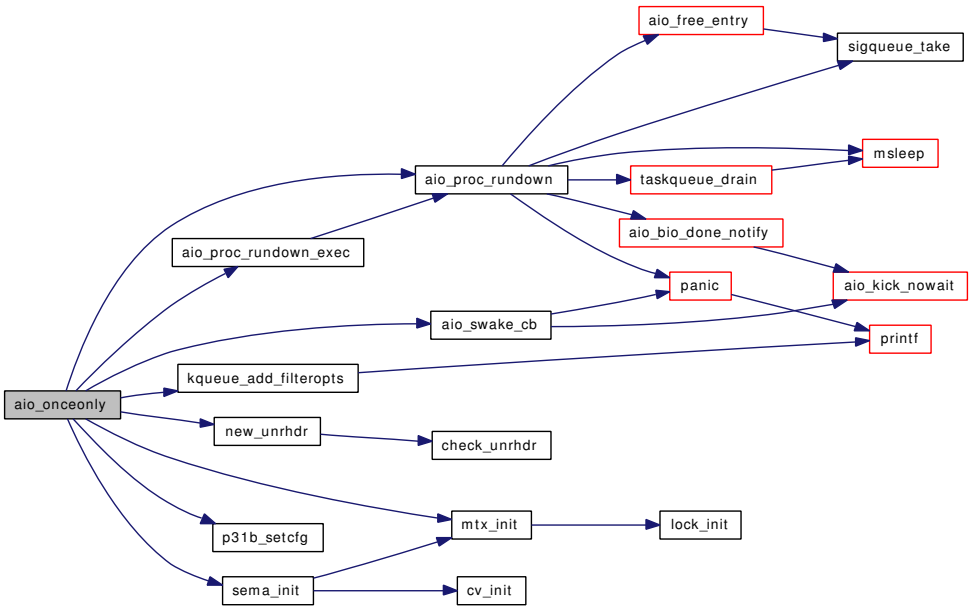
9.151.3.16 static void aio_onceonly (void) [static]

Definition at line 416 of file vfs_aio.c.

References aio_proc_rundown(), aio_proc_rundown_exec(), aio_swake, aio_swake_cb(), aiod_lifetime, AIOD_LIFETIME_DEFAULT, aiod_timeout, AIOD_TIMEOUT_DEFAULT, async_io_version, exec_tag, exit_tag, jobrefid, kqueue_add_filteropts(), MAX_AIO_QUEUE, mtx_init(), new_unrhdr(), p31b_setcfg(), and sema_init().

Referenced by aio_modload().

Here is the call graph for this function:



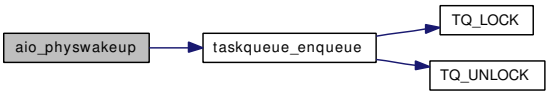
9.151.3.17 static void aio_physwakeup (struct buf * *bp*) [static]

Definition at line 2110 of file vfs_aio.c.

References taskqueue_enqueue().

Referenced by aio_qphysio().

Here is the call graph for this function:



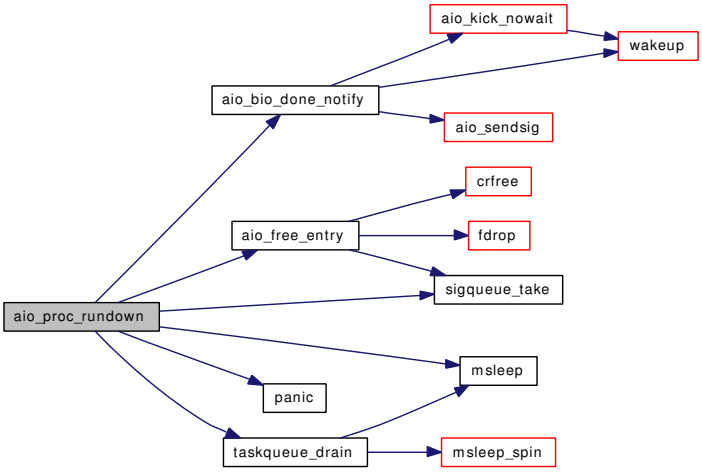
9.151.3.18 static void aio_proc_rundown (void * *arg*, struct proc * *p*) [static]

Definition at line 644 of file vfs_aio.c.

References aio_bio_done_notify(), aio_free_entry(), AIO_LOCK, AIO_MTX, AIO_UNLOCK, DONE_QUEUE, hz, JOBST_JOBFINISHED, JOBST_JOBQGLOBAL, JOBST_JOBQSOCK, JOBST_JOBQSYNC, kaioinfo::kaio_flags, KAIO_RUNDOWN, KAIO_WAKEUP, aioliojob::lioj_count, aioliojob::lioj_finished_count, msleep(), panic(), sigqueue_take(), and taskqueue_drain().

Referenced by aio_onceonly(), and aio_proc_rundown_exec().

Here is the call graph for this function:



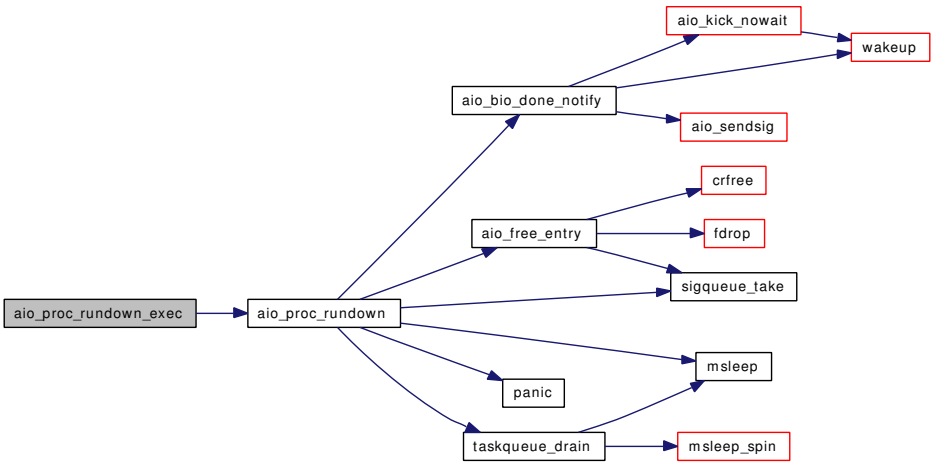
9.151.3.19 static void aio_proc_rundown_exec (void * *arg*, struct proc * *p*, struct image_params *imgp *__unused*) [static]

Definition at line 635 of file vfs_aio.c.

References aio_proc_rundown().

Referenced by aio_onceonly().

Here is the call graph for this function:



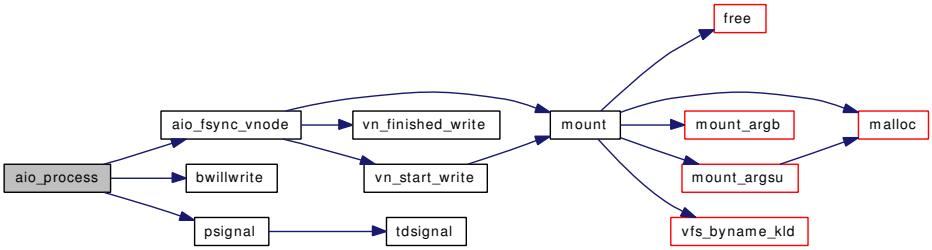
9.151.3.20 static void aio_process (struct aiocblist * *aiocbe*) [static]

Definition at line 790 of file vfs_aio.c.

References aio_fsync_vnode(), bwillwrite(), and psignal().

Referenced by aio_daemon().

Here is the call graph for this function:



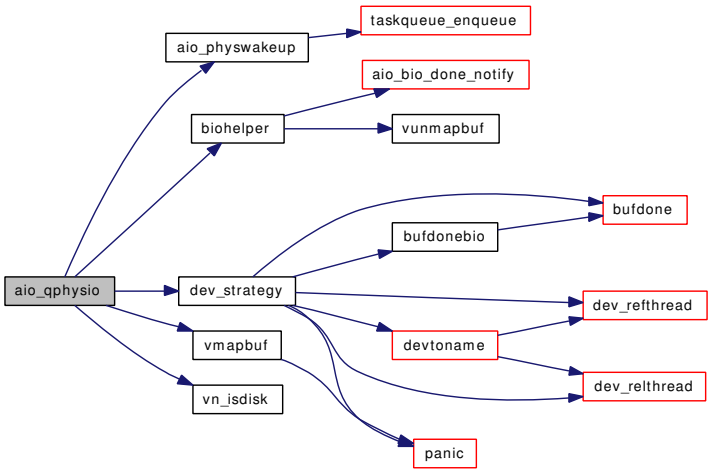
9.151.3.21 static int aio_qphysio (struct proc * *p*, struct aiocblist * *aiocbe*) [static]

Definition at line 1172 of file vfs_aio.c.

References AIO_LOCK, aio_physwakeup(), AIO_UNLOCK, biohelper(), buf, dev_strategy(), JOBST_JOBQBUF, kaioinfo::kaio_ballowed_count, kaioinfo::kaio_buffer_count, kaioinfo::kaio_count, aioliojob::lioj_count, num_buf_aio, num_queue_count, vmapbuf(), and vn_isdisk().

Referenced by aio_aqueue().

Here is the call graph for this function:

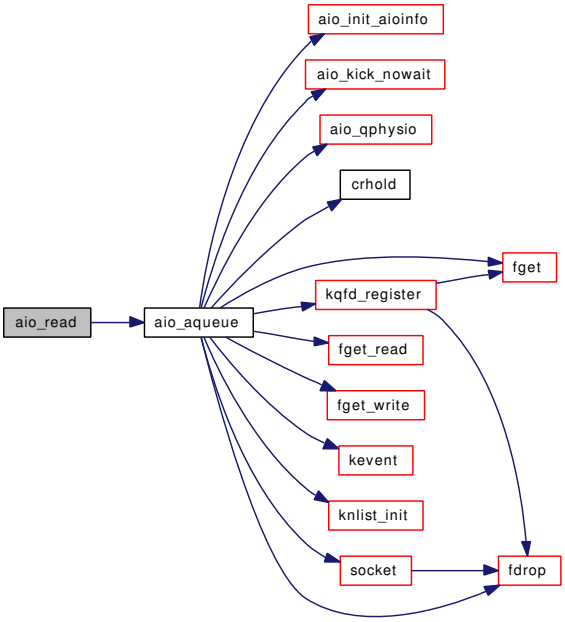


9.151.3.22 int aio_read (struct thread * *td*, struct aio_read_args * *uap*)

Definition at line 1926 of file vfs_aio.c.

References aio_aqueue().

Here is the call graph for this function:

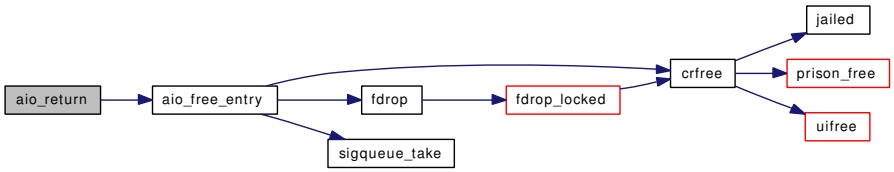


9.151.3.23 int aio_return (struct thread * *td*, struct aio_return_args * *uap*)

Definition at line 1643 of file vfs_aio.c.

References aio_free_entry(), AIO_LOCK, AIO_UNLOCK, JOBST_JOBFINISHED, and suword.

Here is the call graph for this function:



9.151.3.24 static struct aiocblist* aio_selectjob (struct aiothreadlist * *aiop*) [static]

Definition at line 730 of file vfs_aio.c.

References JOBST_JOBRUNNING, kaioinfo::kaio_active_count, and kaioinfo::kaio_maxactive_count.

Referenced by aio_daemon().

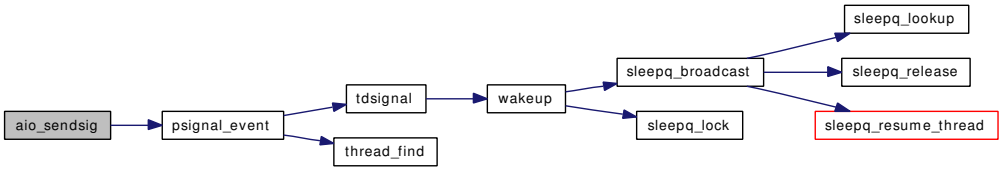
9.151.3.25 static int aio_sendsig (struct proc * *p*, struct sigevent * *sigev*, ksiginfo_t * *ksi*) [static]

Definition at line 538 of file vfs_aio.c.

References psignal_event(), and ret.

Referenced by aio_bio_done_notify(), and do_lio_listio().

Here is the call graph for this function:

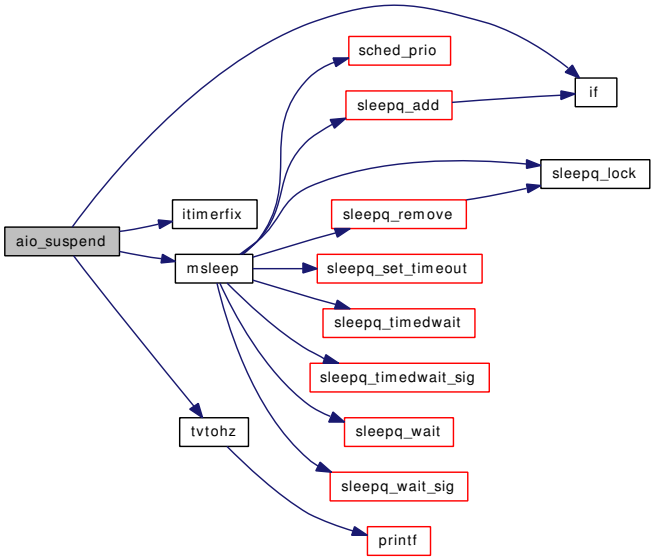


9.151.3.26 int aio_suspend (struct thread * *td*, struct aio_suspend_args * *uap*)

Definition at line 1688 of file vfs_aio.c.

References AIO_LOCK, AIO_MTX, if(), itimerfix(), JOBST_JOBFINISHED, KAIO_WAKEUP, msleep(), and tvtohz().

Here is the call graph for this function:



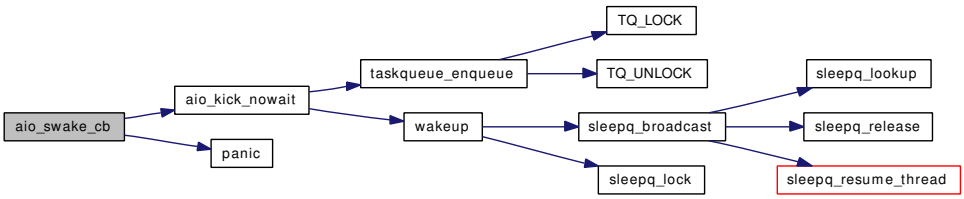
9.151.3.27 static void aio_swake_cb (struct socket * *so*, struct sockbuf * *sb*) [static]

Definition at line 1290 of file vfs_aio.c.

References aio_kick_nowait(), JOBST_JOBQSOCK, and panic().

Referenced by aio_onceonly().

Here is the call graph for this function:



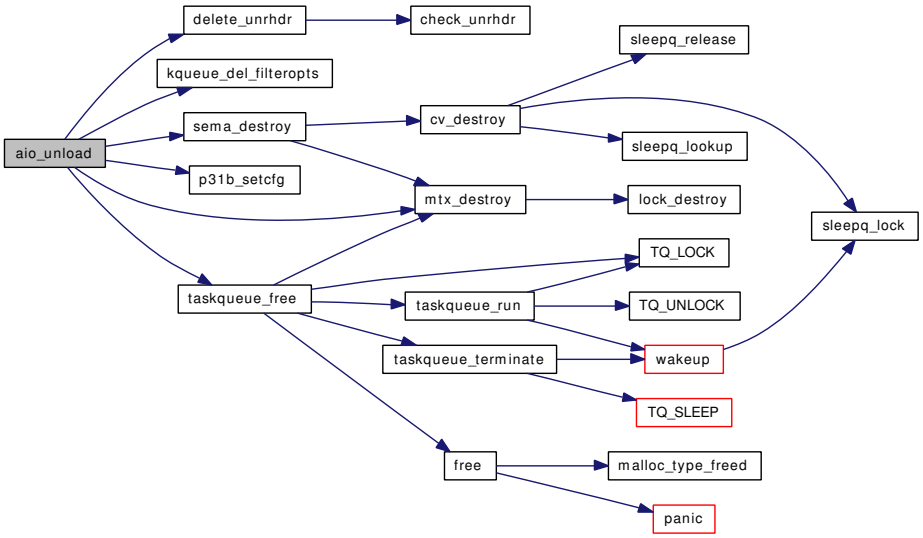
9.151.3.28 static int aio_unload (void) [static]

Definition at line 456 of file vfs_aio.c.

References aio_swake, async_io_version, delete_unrhdr(), exec_tag, exit_tag, kqueue_del_filteropts(), mtx_destroy(), p31b_setcfg(), sema_destroy(), taskqueue_free(), and unloadable.

Referenced by aio_modload().

Here is the call graph for this function:

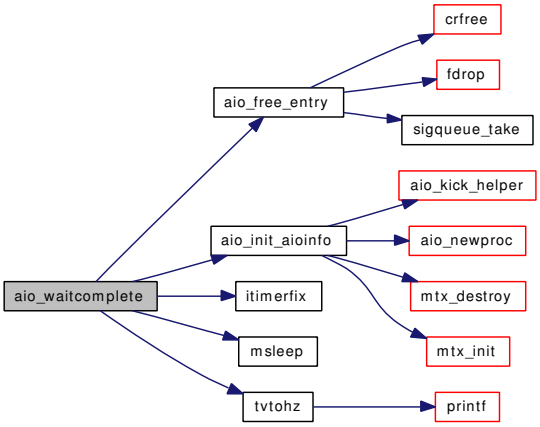


9.151.3.29 int aio_waitcomplete (struct thread * *td*, struct aio_waitcomplete_args * *uap*)

Definition at line 2157 of file vfs_aio.c.

References aio_free_entry(), aio_init_aioinfo(), AIO_LOCK, AIO_MTX, AIO_UNLOCK, itimerfix(), JOBST_JOBFINISHED, kaioinfo::kaio_flags, KAIO_WAKEUP, msleep(), suword, and tvtohz().

Here is the call graph for this function:

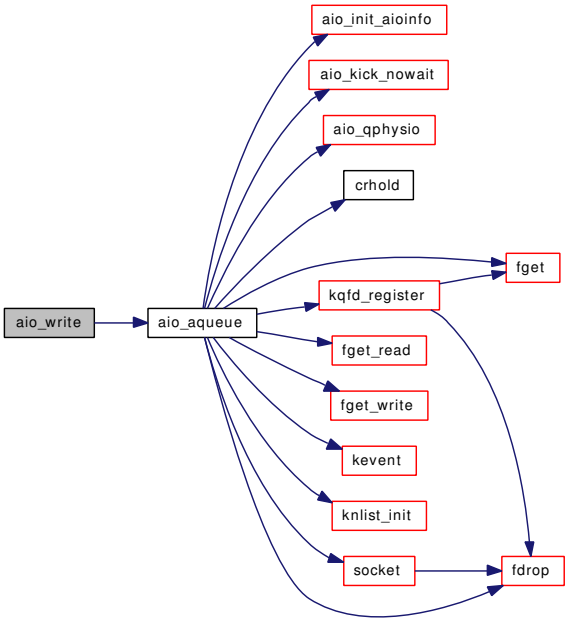


9.151.3.30 int aio_write (struct thread * *td*, struct aio_write_args * *uap*)

Definition at line 1941 of file vfs_aio.c.

References aio_aqueue().

Here is the call graph for this function:



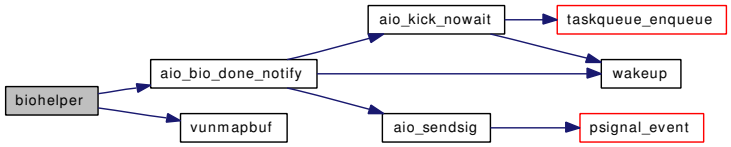
9.151.3.31 static void biohelper (void * *context*, int *pending*) [static]

Definition at line 2122 of file vfs_aio.c.

References aio_bio_done_notify(), AIO_LOCK, AIO_UNLOCK, buf, DONE_BUF, kaioinfo::kaio_-buffer_count, num_buf_aio, and vunmapbuf().

Referenced by aio_qphysio().

Here is the call graph for this function:



9.151.3.32 DECLARE_MODULE (aio, aio_mod, SI_SUB_VFS, SI_ORDER_ANY)

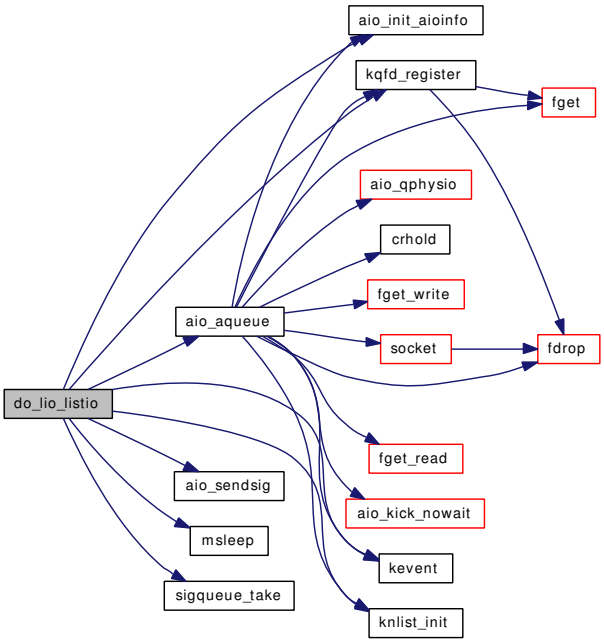
9.151.3.33 static int do_lio_listio (struct thread * *td*, struct lio_listio_args * *uap*, int *oldsigev*) [static]

Definition at line 1962 of file vfs_aio.c.

References aio_aqueue(), aio_init_aioinfo(), AIO_LOCK, AIO_MTX, aio_sendsig(), AIO_UNLOCK, kaioinfo::kaio_flags, KAIO_WAKEUP, kevent(), knlist_init(), kqfd_register(), aioliojob::lioj_count, aioliojob::lioj_finished_count, aioliojob::lioj_flags, LIOJ_KEVENT_POSTED, LIOJ_SIGNAL, aioliojob::lioj_signal, LIOJ_SIGNAL_POSTED, msleep(), and sigqueue_take().

Referenced by lio_listio(), and olio_listio().

Here is the call graph for this function:

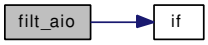


9.151.3.34 static int filt_aio (struct knote * *kn*, long *hint*) [static]

Definition at line 2273 of file vfs_aio.c.

References if(), and JOBST_JOBFINISHED.

Here is the call graph for this function:

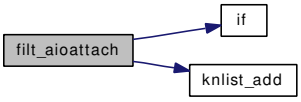


9.151.3.35 static int filt_aioattach (struct knote * *kn*) [static]

Definition at line 2242 of file vfs_aio.c.

References if(), and knlist_add().

Here is the call graph for this function:

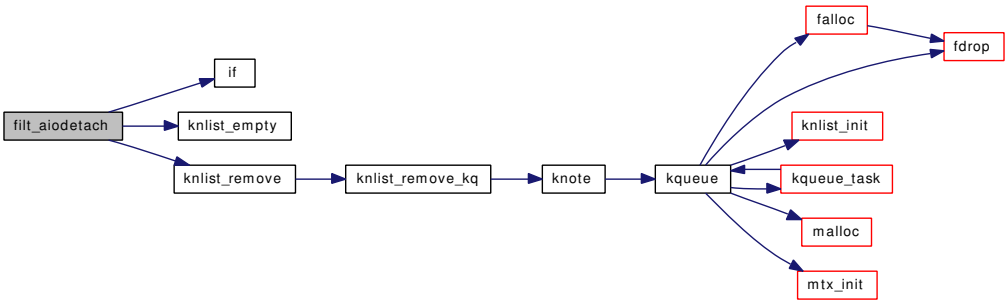


9.151.3.36 static void filt_aiodetach (struct knote * *kn*) [static]

Definition at line 2262 of file vfs_aio.c.

References if(), knlist_empty(), and knlist_remove().

Here is the call graph for this function:



9.151.3.37 static int filt_lio (struct knote * *kn*, long *hint*) [static]

Definition at line 2317 of file vfs_aio.c.

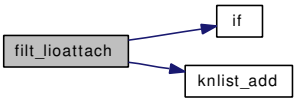
References aioliojob::lioj_flags, and LIOJ_KEVENT_POSTED.

9.151.3.38 static int filt_lioattach (struct knote * *kn*) [static]

Definition at line 2286 of file vfs_aio.c.

References if(), and knlist_add().

Here is the call graph for this function:

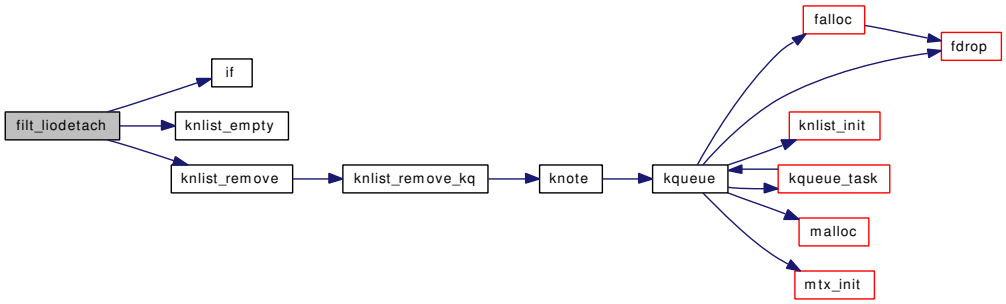


9.151.3.39 static void filt_liodetach (struct knote * *kn*) [static]

Definition at line 2306 of file vfs_aio.c.

References if(), knlist_empty(), and knlist_remove().

Here is the call graph for this function:

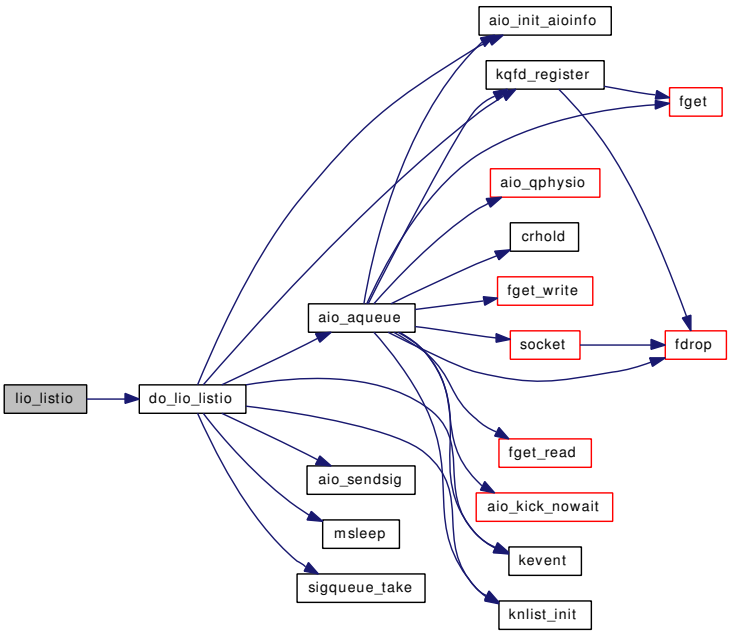


9.151.3.40 int lio_listio (struct thread * *td*, struct lio_listio_args * *uap*)

Definition at line 1956 of file vfs_aio.c.

References do_lio_listio().

Here is the call graph for this function:



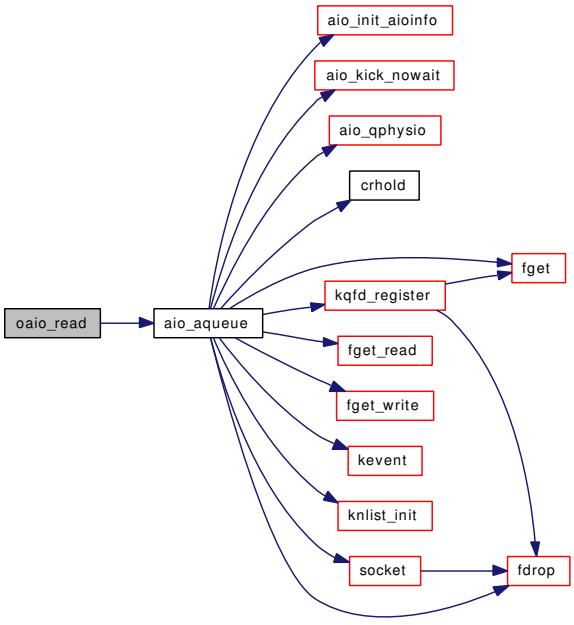
9.151.3.41 MODULE_VERSION (aio, 1)

9.151.3.42 int oaio_read (struct thread * *td*, struct oaio_read_args * *uap*)

Definition at line 1919 of file vfs_aio.c.

References aio_aqueue().

Here is the call graph for this function:

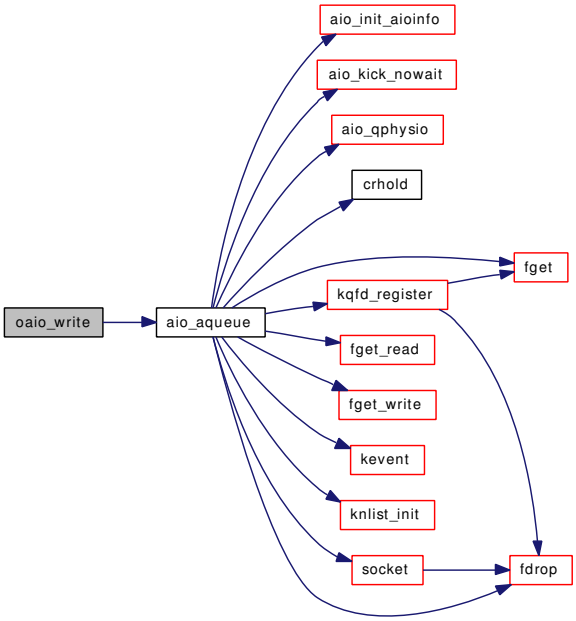


9.151.3.43 int oaio_write (struct thread * *td*, struct oaio_write_args * *uap*)

Definition at line 1934 of file vfs_aio.c.

References aio_aqueue().

Here is the call graph for this function:

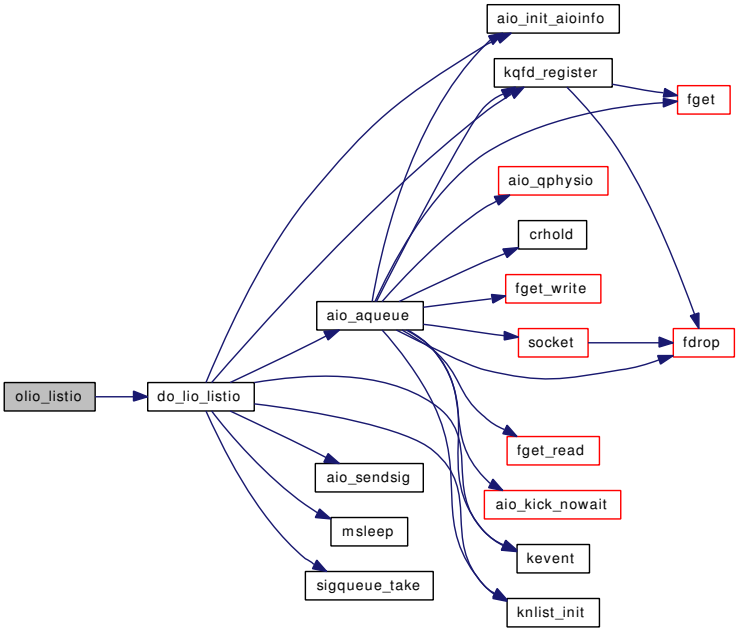


9.151.3.44 int olio_listio (struct thread * *td*, struct olio_listio_args * *uap*)

Definition at line 1949 of file vfs_aio.c.

References do_lio_listio().

Here is the call graph for this function:


aio_init_aioinfo
kqfd_register
fget
aio_qphysio
crhold
fget_write
aio_aqueue
socket
fdrop
olio_listio
do_lio_listio
fget_read
aio_sendsig
aio_kick_nowait
msleep
kevent
sigqueue_take
knlist_init

9.151.3.45 SYSCALL_MODULE_HELPER (olio_listio)

9.151.3.46 SYSCALL_MODULE_HELPER (oaio_write)

9.151.3.47 SYSCALL_MODULE_HELPER (oaio_read)

9.151.3.48 SYSCALL_MODULE_HELPER (lio_listio)

9.151.3.49 SYSCALL_MODULE_HELPER (aio_write)

9.151.3.50 SYSCALL_MODULE_HELPER (aio_waitcomplete)

9.151.3.51 SYSCALL_MODULE_HELPER (aio_suspend)

9.151.3.52 SYSCALL_MODULE_HELPER (aio_return)

9.151.3.53 SYSCALL_MODULE_HELPER (aio_read)

9.151.3.54 SYSCALL_MODULE_HELPER (aio_fsync)

9.151.3.55 SYSCALL_MODULE_HELPER (aio_error)

9.151.3.56 SYSCALL_MODULE_HELPER (aio_cancel)

9.151.3.57 SYSCTL_INT (_vfs_aio, OID_AUTO, max_buf_aio, CTLFLAG_RW, & *max_buf_aio*, 0, "Maximum buf aio requests per process (stored in the process)")

9.151.3.58 SYSCTL_INT (_vfs_aio, OID_AUTO, max_aio_queue_per_proc, CTLFLAG_RW, & *max_aio_queue_per_proc*, 0, "Maximum queued aio requests per process (stored in the process)")

9.151.3.59 SYSCTL_INT (_vfs_aio, OID_AUTO, max_aio_per_proc, CTLFLAG_RW, & *max_aio_per_proc*, 0, "Maximum active aio requests per process (stored in the process)")

9.151.3.60 SYSCTL_INT (_vfs_aio, OID_AUTO, unloadable, CTLFLAG_RW, & *unloadable*, 0, "Allow unload of aio (not recommended)")

9.151.3.61 SYSCTL_INT (_vfs_aio, OID_AUTO, aiod_lifetime, CTLFLAG_RW, & *aiod_lifetime*, 0, "Maximum lifetime for idle aiod")

9.151.3.62 SYSCTL_INT (_vfs_aio, OID_AUTO, aiod_timeout, CTLFLAG_RW, & *aiod_timeout*, 0, "Timeout value for synchronous aio operations")

9.151.3.63 SYSCTL_INT (_vfs_aio, OID_AUTO, num_buf_aio, CTLFLAG_RD, & *num_buf_aio*, 0, "Number of aio requests presently handled by the buf subsystem")

9.151.3.64 SYSCTL_INT (_vfs_aio, OID_AUTO, num_queue_count, CTLFLAG_RD, & *num_queue_count*, 0, "Number of queued aio requests")

9.151.3.65 SYSCTL_INT (_vfs_aio, OID_AUTO, max_aio_queue, CTLFLAG_RW, & *max_queue_count*, 0, "Maximum number of aio requests to *queue*, globally")

9.151.3.66 SYSCTL_INT (_vfs_aio, OID_AUTO, target_aio_procs, CTLFLAG_RW, & *target_aio_procs*, 0, "Preferred number of ready kernel threads for async IO")

9.151.3.67 SYSCTL_INT (_vfs_aio, OID_AUTO, num_aio_procs, CTLFLAG_RD, & *num_aio_procs*, 0, "Number of presently active kernel threads for async IO")

9.151.3.68 SYSCTL_INT (_vfs_aio, OID_AUTO, max_aio_procs, CTLFLAG_RW, & *max_aio_procs*, 0, "Maximum number of kernel threads to use for handling async IO ")

9.151.3.71 TASKQUEUE_DEFINE_THREAD (aiod_bio)

9.151.4 Variable Documentation

9.151.4.1 moduledata_t aio_mod [static]

Initial value:

```
{
        "aio",
        &aio_modload,
        NULL
}
```

Definition at line 389 of file vfs_aio.c.

9.151.4.2 int aiod_lifetime [static]

Definition at line 162 of file vfs_aio.c.

Referenced by aio_daemon(), and aio_onceonly().

9.151.4.3 int aiod_timeout [static]

Definition at line 158 of file vfs_aio.c.

Referenced by aio_onceonly().

9.151.4.4 eventhandler_tag exec_tag [static]

Definition at line 361 of file vfs_aio.c.

Referenced by aio_onceonly(), and aio_unload().

9.151.4.5 eventhandler_tag exit_tag [static]

Definition at line 361 of file vfs_aio.c.

9.151.4.6 u_long jobrefid [static]

Definition at line 75 of file vfs_aio.c.

Referenced by aio_aqueue(), and aio_onceonly().

9.151.4.7 uint64_t jobseqno [static]

Definition at line 80 of file vfs_aio.c.

Referenced by aio_aqueue().

9.151.4.8 struct filterops lio_filtops [static]

Initial value:

```
{ 0, filt_lioattach, filt_liodetach, filt_lio }
```

Definition at line 358 of file vfs_aio.c.

9.151.4.9 int max_aio_per_proc = MAX_AIO_PER_PROC [static]

Definition at line 171 of file vfs_aio.c.

Referenced by aio_init_aioinfo().

9.151.4.10 int max_aio_procs = MAX_AIO_PROCS [static]

Definition at line 124 of file vfs_aio.c.

Referenced by aio_kick(), and aio_kick_nowait().

9.151.4.11 int max_aio_queue_per_proc = MAX_AIO_QUEUE_PER_PROC [static]

Definition at line 175 of file vfs_aio.c.

Referenced by aio_init_aioinfo().

9.151.4.12 int max_buf_aio = MAX_BUF_AIO [static]

Definition at line 180 of file vfs_aio.c.

Referenced by aio_init_aioinfo().

9.151.4.13 int max_queue_count = MAX_AIO_QUEUE [static]

Definition at line 142 of file vfs_aio.c.

Referenced by aio_aqueue().

9.151.4.14 int num_aio_procs = 0 [static]

Definition at line 129 of file vfs_aio.c.

Referenced by aio_daemon(), aio_init_aioinfo(), aio_kick(), aio_kick_nowait(), and aio_newproc().

9.151.4.15 int num_aio_resv_start = 0 [static]

Definition at line 156 of file vfs_aio.c.

Referenced by aio_kick(), and aio_kick_nowait().

9.151.4.16 int num_buf_aio = 0 [static]

Definition at line 150 of file vfs_aio.c.

Referenced by aio_qphysio(), and biohelper().

9.151.4.17 int num_queue_count = 0 [static]

Definition at line 146 of file vfs_aio.c.

Referenced by aio_aqueue(), aio_free_entry(), and aio_qphysio().

9.151.4.18 int target_aio_procs = TARGET_AIO_PROCS [static]

Definition at line 138 of file vfs_aio.c.

Referenced by aio_daemon(), and aio_init_aioinfo().

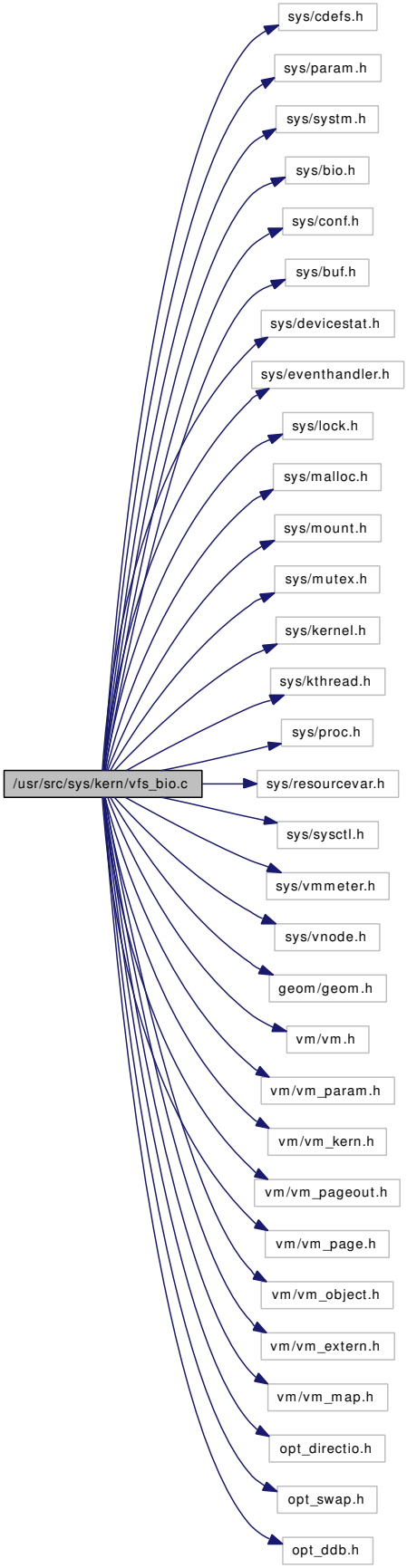
9.151.4.19 int unloadable = 0 [static]

Definition at line 166 of file vfs_aio.c.

9.152 /usr/src/sys/kern/vfs_bio.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/bio.h>
#include <sys/conf.h>
#include <sys/buf.h>
#include <sys/devicestat.h>
#include <sys/eventhandler.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mount.h>
#include <sys/mutex.h>
#include <sys/kernel.h>
#include <sys/kthread.h>
#include <sys/proc.h>
#include <sys/resourcevar.h>
#include <sys/sysctl.h>
#include <sys/vmmeter.h>
#include <sys/vnode.h>
#include <geom/geom.h>
#include <vm/vm.h>
#include <vm/vm_param.h>
#include <vm/vm_kern.h>
#include <vm/vm_pageout.h>
#include <vm/vm_page.h>
#include <vm/vm_object.h>
#include <vm/vm_extern.h>
#include <vm/vm_map.h>
#include "opt_directio.h"
#include "opt_swap.h"
#include "opt_ddb.h"
```

Include dependency graph for vfs_bio.c:


/usr/src/sys/kern/vfs_bio.c
sys/cdefs.h
sys/param.h
sys/systm.h
sys/bio.h
sys/conf.h
sys/buf.h
sys/devicestat.h
sys/eventhandler.h
sys/lock.h
sys/malloc.h
sys/mount.h
sys/mutex.h
sys/kernel.h
sys/kthread.h
sys/proc.h
sys/resourcevar.h
sys/sysctl.h
sys/vmmeter.h
sys/vnode.h
geom/geom.h
vm/vm.h
vm/vm_param.h
vm/vm_kern.h
vm/vm_pageout.h
vm/vm_page.h
vm/vm_object.h
vm/vm_extern.h
vm/vm_map.h
opt_directio.h
opt_swap.h
opt_ddb.h

Defines

- #define BUFFER_QUEUES 6
- #define QUEUE_NONE 0
- #define QUEUE_CLEAN 1
- #define QUEUE_DIRTY 2
- #define QUEUE_DIRTY_GIANT 3
- #define QUEUE_EMPTYKVA 4
- #define QUEUE_EMPTY 5
- #define VFS_BIO_NEED_ANY 0x01
- #define VFS_BIO_NEED_DIRTYFLUSH 0x02
- #define VFS_BIO_NEED_FREE 0x04
- #define VFS_BIO_NEED_BUFSPACE 0x08

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/vfs_bio.c,v 1.517 2007/02/22 14:52:59 delphij Exp $")
- static MALLOC_DEFINE (M_BIOBUF,"biobuf","BIO buffer")
- static int inmem (struct vnode *vp, daddr_t blkno)
- static void vm_hold_free_pages (struct buf *bp, vm_offset_t from, vm_offset_t to)
- static void vm_hold_load_pages (struct buf *bp, vm_offset_t from, vm_offset_t to)
- static void vfs_page_set_valid (struct buf *bp, vm_ooffset_t off, int pageno, vm_page_t m)
- static void vfs_clean_pages (struct buf *bp)
- static void vfs_setdirty (struct buf *bp)
- static void vfs_setdirty_locked_object (struct buf *bp)
- static void vfs_vmio_release (struct buf *bp)
- static int vfs_bio_clcheck (struct vnode *vp, int size, daddr_t lblkno, daddr_t blkno)
- static int flushbufqueues (int, int)
- static void buf_daemon (void)
- static void bremfreel (struct buf *bp)
- SYSCTL_INT (_vfs, OID_AUTO, vmiodirenable, CTLFLAG_RW,&vmiodirenable, 0,"Use the VM system for directory writes")
- SYSCTL_INT (_vfs, OID_AUTO, runningbufspace, CTLFLAG_RD,&runningbufspace, 0,"Amount of presently outstanding async buffer io")
- SYSCTL_INT (_vfs, OID_AUTO, bufspace, CTLFLAG_RD,&bufspace, 0,"KVA memory used for bufs")
- SYSCTL_INT (_vfs, OID_AUTO, maxbufspace, CTLFLAG_RD,&maxbufspace, 0,"Maximum allowed value of bufspace (including buf_daemon)")
- SYSCTL_INT (_vfs, OID_AUTO, bufmallocspace, CTLFLAG_RD,&bufmallocspace, 0,"Amount of malloced memory for buffers")
- SYSCTL_INT (_vfs, OID_AUTO, maxmallocbufspace, CTLFLAG_RW,&maxbufmallocspace, 0,"Maximum amount of malloced memory for buffers")
- SYSCTL_INT (_vfs, OID_AUTO, lobufspace, CTLFLAG_RD,&lobufspace, 0,"Minimum amount of buffers we want to have")
- SYSCTL_INT (_vfs, OID_AUTO, hibufspace, CTLFLAG_RD,&hibufspace, 0,"Maximum allowed value of bufspace (excluding buf_daemon)")
- SYSCTL_INT (_vfs, OID_AUTO, bufreusecnt, CTLFLAG_RW,&bufreusecnt, 0,"Number of times we have reused a buffer")
- SYSCTL_INT (_vfs, OID_AUTO, buffreekvacnt, CTLFLAG_RW,&buffreekvacnt, 0,"Number of times we have freed the KVA space from some buffer")

- SYSCTL_INT (_vfs, OID_AUTO, bufdefragcnt, CTLFLAG_RW,&bufdefragcnt, 0,"Number of times we have had to repeat buffer allocation to defragment")
- SYSCTL_INT (_vfs, OID_AUTO, lorunningspace, CTLFLAG_RW,&lorunningspace, 0,"Minimum preferred space used for in-progress I/O")
- SYSCTL_INT (_vfs, OID_AUTO, hirunningspace, CTLFLAG_RW,&hirunningspace, 0,"Maximum amount of space to use for in-progress I/O")
- SYSCTL_INT (_vfs, OID_AUTO, dirtybufferflushes, CTLFLAG_RW,&dirtybufferflushes, 0,"Number of bdwrite to bawrite conversions to limit dirty buffers")
- SYSCTL_INT (_vfs, OID_AUTO, bdwriteskip, CTLFLAG_RW,&bdwriteskip, 0,"Number of buffers supplied to bdwrite with snapshot deadlock risk")
- SYSCTL_INT (_vfs, OID_AUTO, altbufferflushes, CTLFLAG_RW,&altbufferflushes, 0,"Number of fsync flushes to limit dirty buffers")
- SYSCTL_INT (_vfs, OID_AUTO, recursiveflushes, CTLFLAG_RW,&recursiveflushes, 0,"Number of flushes skipped due to being recursive")
- SYSCTL_INT (_vfs, OID_AUTO, numdirtybuffers, CTLFLAG_RD,&numdirtybuffers, 0,"Number of buffers that are dirty (has unwritten changes) at the moment")
- SYSCTL_INT (_vfs, OID_AUTO, lodirtybuffers, CTLFLAG_RW,&lodirtybuffers, 0,"How many buffers we want to have free before bufdaemon can sleep")
- SYSCTL_INT (_vfs, OID_AUTO, hidirtybuffers, CTLFLAG_RW,&hidirtybuffers, 0,"When the number of dirty buffers is considered severe")
- SYSCTL_INT (_vfs, OID_AUTO, dirtybufthresh, CTLFLAG_RW,&dirtybufthresh, 0,"Number of bdwrite to bawrite conversions to clear dirty buffers")
- SYSCTL_INT (_vfs, OID_AUTO, numfreebuffers, CTLFLAG_RD,&numfreebuffers, 0,"Number of free buffers")
- SYSCTL_INT (_vfs, OID_AUTO, lofreebuffers, CTLFLAG_RW,&lofreebuffers, 0,"XXX Unused")
- SYSCTL_INT (_vfs, OID_AUTO, hifreebuffers, CTLFLAG_RW,&hifreebuffers, 0,"XXX Complicatedly unused")
- SYSCTL_INT (_vfs, OID_AUTO, getnewbufcalls, CTLFLAG_RW,&getnewbufcalls, 0,"Number of calls to getnewbuf")
- SYSCTL_INT (_vfs, OID_AUTO, getnewbufrestarts, CTLFLAG_RW,&getnewbufrestarts, 0,"Number of times getnewbuf has had to restart a buffer aquisition")
- static TAILQ_HEAD (bqueues, buf)
- static __inline void numdirtywakeup (int level)
- static __inline void bufspacewakeup (void)
- void runningbufwakeup (struct buf *bp)
- static __inline void bufcountwakeup (void)
- void waitrunningbufspace (void)
- static __inline void vfs_buf_test_cache (struct buf *bp, vm_ooffset_t foff, vm_offset_t off, vm_offset_t size, vm_page_t m)
- static __inline void bd_wakeup (int dirtybuflevel)
- static __inline void bd_speedup (void)
- caddr_t kern_vfs_bio_buffer_alloc (caddr_t v, long physmem_est)
- void bufinit (void)
- static void bfreekva (struct buf *bp)
- void bremfree (struct buf *bp)
- void bremfreef (struct buf *bp)
- int bread (struct vnode *vp, daddr_t blkno, int size, struct ucred *cred, struct buf **bpp)
- void breada (struct vnode *vp, daddr_t *rablkno, int *rabsize, int cnt, struct ucred *cred)
- int breadn (struct vnode *vp, daddr_t blkno, int size, daddr_t *rablkno, int *rabsize, int cnt, struct ucred *cred, struct buf **bpp)
- int bufwrite (struct buf *bp)

- void bufbdflush (struct bufobj *bo, struct buf *bp)
- void bdwrite (struct buf *bp)
- void bdirty (struct buf *bp)
- void bundirty (struct buf *bp)
- void bawrite (struct buf *bp)
- void bwillwrite (void)
- int buf_dirty_count_severe (void)
- void brelse (struct buf *bp)
- void bqrelse (struct buf *bp)
- int vfs_bio_awrite (struct buf *bp)
- static struct buf * getnewbuf (int slpflag, int slptimeo, int size, int maxsize)
- SYSCTL_INT (_vfs, OID_AUTO, flushwithdeps, CTLFLAG_RW,&flushwithdeps, 0,"Number of buffers flushed with dependecies that require rollbacks")
- buf * incore (struct bufobj *bo, daddr_t blkno)
- buf * getblk (struct vnode *vp, daddr_t blkno, int size, int slpflag, int slptimeo, int flags)
- buf * geteblk (int size)
- int allocbuf (struct buf *bp, int size)
- void biodone (struct bio *bp)
- int biowait (struct bio *bp, const char *wchan)
- void biofinish (struct bio *bp, struct devstat *stat, int error)
- int bufwait (struct buf *bp)
- static void bufdonebio (struct bio *bip)
- void dev_strategy (struct cdev *dev, struct buf *bp)
- void bufdone (struct buf *bp)
- void bufdone_finish (struct buf *bp)
- void vfs_unbusy_pages (struct buf *bp)
- void vfs_busy_pages (struct buf *bp, int clear_modify)
- void vfs_bio_set_validclean (struct buf *bp, int base, int size)
- void vfs_bio_clrbuf (struct buf *bp)
- int vmapbuf (struct buf *bp)
- void vunmapbuf (struct buf *bp)
- void bdone (struct buf *bp)
- void bwait (struct buf *bp, u_char pri, const char *wchan)
- int bufsync (struct bufobj *bo, int waitfor, struct thread *td)
- void bufstrategy (struct bufobj *bo, struct buf *bp)
- void bufobj_wrefl (struct bufobj *bo)
- void bufobj_wref (struct bufobj *bo)
- void bufobj_wdrop (struct bufobj *bo)
- int bufobj_wwait (struct bufobj *bo, int slpflag, int timeo)
- void bpin (struct buf *bp)
- void bunpin (struct buf *bp)
- void bunpin_wait (struct buf *bp)

Variables

- bio_ops bioops
- buf_ops buf_ops_bio
- buf ∗ buf
- static struct proc ∗ bufdaemonproc
- int vmiodirenable = TRUE
- int runningbufspace
- static int bufspace
- static int maxbufspace
- static int bufmallocspace
- static int maxbufmallocspace
- static int lobufspace
- int hibufspace
- static int bufreusecnt
- static int buffreekvacnt
- static int bufdefragcnt
- static int lorunningspace
- static int hirunningspace
- int dirtybufferflushes
- int bdwriteskip
- int altbufferflushes
- static int recursiveflushes
- static int numdirtybuffers
- static int lodirtybuffers
- static int hidirtybuffers
- int dirtybufthresh
- static int numfreebuffers
- static int lofreebuffers
- static int hifreebuffers
- static int getnewbufcalls
- static int getnewbufrestarts
- static int bd_request
- static struct mtx bdlock
- vm_page_t bogus_page
- static int runningbufreq
- static struct mtx rbreqlock
- static int needsbuffer
- static struct mtx nblock
- static struct mtx bdonelock
- static struct mtx bpinlock
- static struct mtx bqlock
- const char ∗ buf_wmesg = BUF_WMESG
- static struct kproc_desc buf_kp
- static int flushwithdeps = 0

9.152.1 Define Documentation

9.152.1.1 #define BUFFER_QUEUES 6

Definition at line 252 of file vfs_bio.c.

Referenced by bufinit().

9.152.1.2 #define QUEUE_CLEAN 1

Definition at line 255 of file vfs_bio.c.

Referenced by bqrelse(), and getnewbuf().

9.152.1.3 #define QUEUE_DIRTY 2

Definition at line 256 of file vfs_bio.c.

Referenced by bqrelse(), and buf_daemon().

9.152.1.4 #define QUEUE_DIRTY_GIANT 3

Definition at line 257 of file vfs_bio.c.

Referenced by bqrelse(), and buf_daemon().

9.152.1.5 #define QUEUE_EMPTY 5

Definition at line 259 of file vfs_bio.c.

Referenced by bufinit(), and getnewbuf().

9.152.1.6 #define QUEUE_EMPTYKVA 4

Definition at line 258 of file vfs_bio.c.

Referenced by getnewbuf().

9.152.1.7 #define QUEUE_NONE 0

Definition at line 254 of file vfs_bio.c.

Referenced by bdirty(), bqrelse(), bremfree(), bremfreel(), and bundirty().

9.152.1.8 #define VFS_BIO_NEED_ANY 0x01

Definition at line 273 of file vfs_bio.c.

Referenced by bufcountwakeup(), getblk(), and getnewbuf().

9.152.1.9 #define VFS_BIO_NEED_BUFSPACE 0x08

Definition at line 276 of file vfs_bio.c.

Referenced by bufspacewakeup(), and getnewbuf().

9.152.1.10 #define VFS_BIO_NEED_DIRTYFLUSH 0x02

Definition at line 274 of file vfs_bio.c.

Referenced by bwillwrite(), and numdirtywakeup().

9.152.1.11 #define VFS_BIO_NEED_FREE 0x04

Definition at line 275 of file vfs_bio.c.

Referenced by bufcountwakeup().

9.152.2 Function Documentation

9.152.2.1 __FBSDID ("$FreeBSD: src/sys/kern/vfs_bio. *c*, v 1.517 2007/02/22 14:52:59 delphij Exp $")

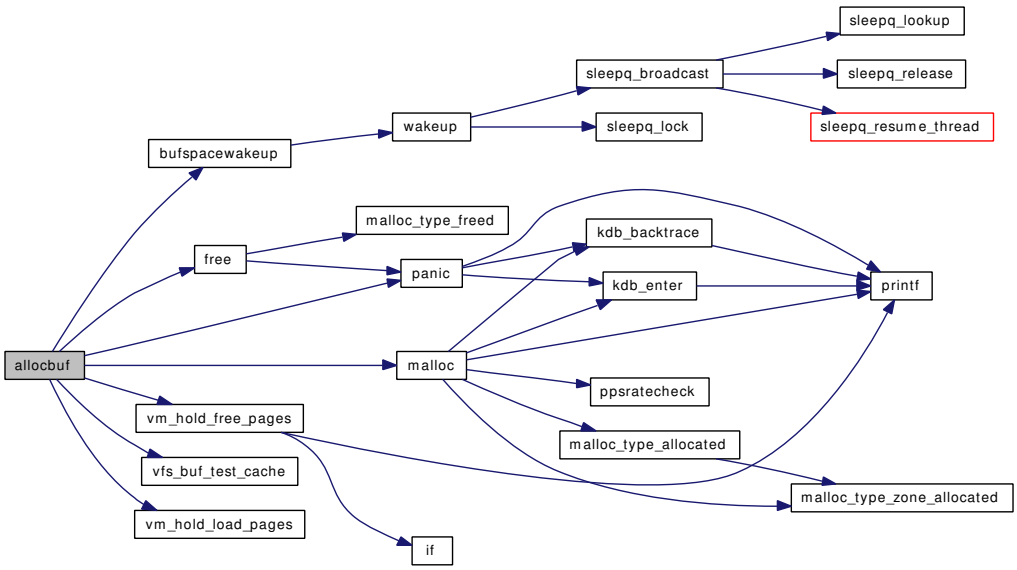
9.152.2.2 int allocbuf (struct buf * *bp*, int *size*)

Definition at line 2703 of file vfs_bio.c.

References bogus_page, bufmallocspace, bufspacewakeup(), free(), malloc(), maxbufmallocspace, panic(), vfs_buf_test_cache(), vm_hold_free_pages(), and vm_hold_load_pages().

Referenced by brelse(), getblk(), geteblk(), and getnewbuf().

Here is the call graph for this function:



9.152.2.3 void bawrite (struct buf * *bp*)

Definition at line 1103 of file vfs_bio.c.

Referenced by bufbdflush(), cluster_wbuild(), cluster_write(), vop_stdfsync(), and vtruncbuf().

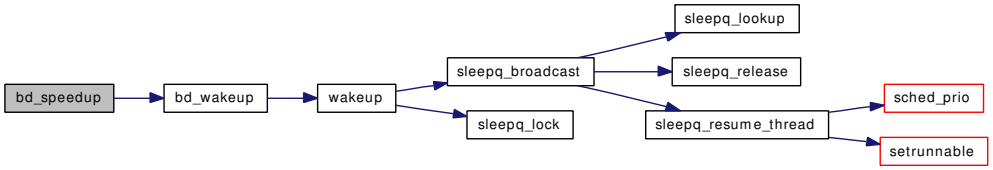
9.152.2.4 static __inline void bd_speedup (void) [static]

Definition at line 442 of file vfs_bio.c.

References bd_wakeup().

Referenced by getnewbuf().

Here is the call graph for this function:



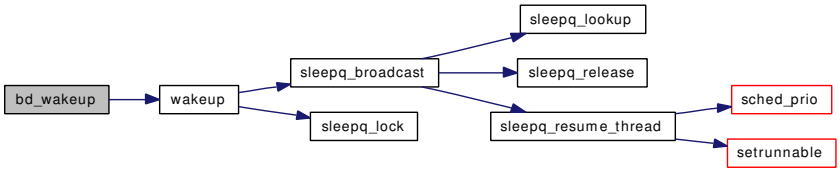
9.152.2.5 static __inline void bd_wakeup (int *dirtybuflevel*) [static]

Definition at line 425 of file vfs_bio.c.

References bd_request, bdlock, numdirtybuffers, and wakeup().

Referenced by bd_speedup(), bdirty(), bdwrite(), and bwillwrite().

Here is the call graph for this function:



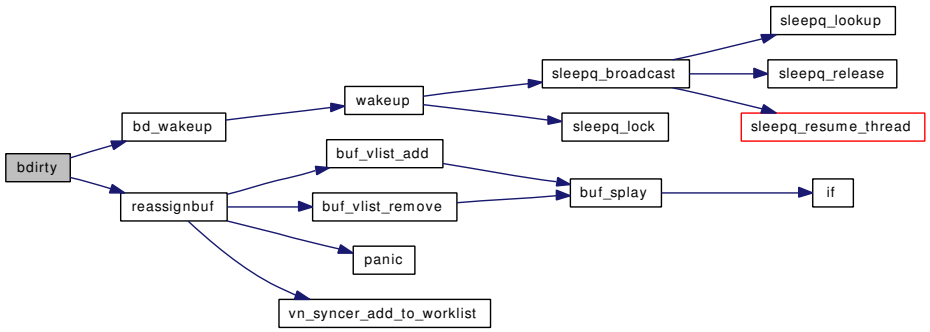
9.152.2.6 void bdirty (struct buf * *bp*)

Definition at line 1040 of file vfs_bio.c.

References bd_wakeup(), hidirtybuffers, lodirtybuffers, numdirtybuffers, QUEUE_NONE, and reassignbuf().

Referenced by bdwrite(), and brelse().

Here is the call graph for this function:



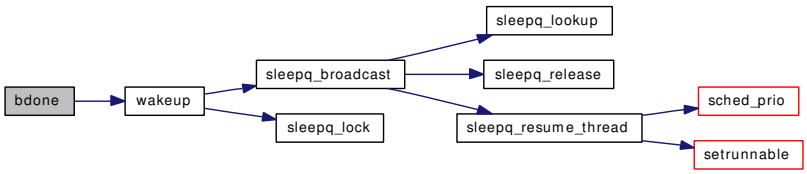
9.152.2.7 void bdone (struct buf * *bp*)

Definition at line 3788 of file vfs_bio.c.

References bdonelock, and wakeup().

Referenced by bufdone_finish(), and physio().

Here is the call graph for this function:



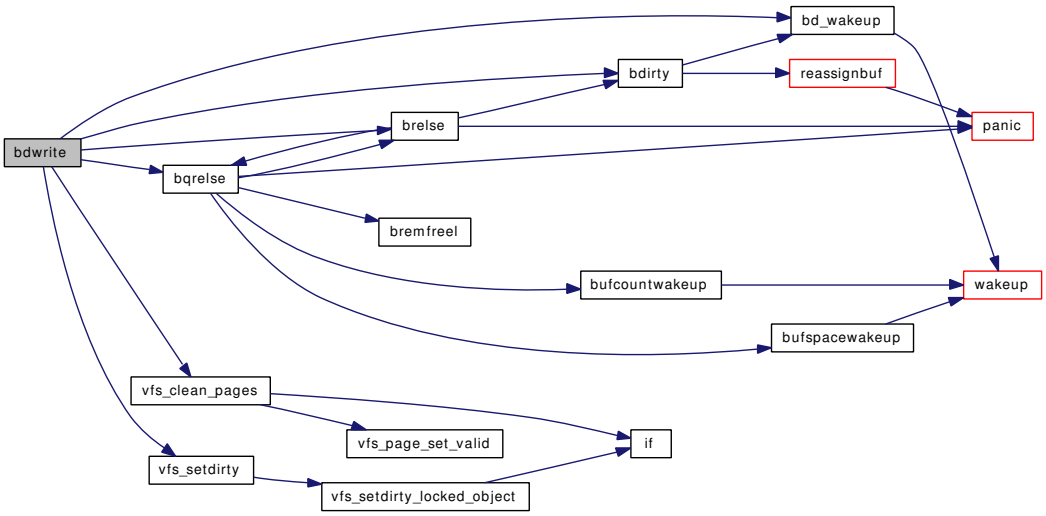
9.152.2.8 void bdwrite (struct buf * *bp*)

Definition at line 944 of file vfs_bio.c.

References bd_wakeup(), bdirty(), bqrelse(), brelse(), hidirtybuffers, lodirtybuffers, recursiveflushes, td, vfs_clean_pages(), and vfs_setdirty().

Referenced by cluster_write().

Here is the call graph for this function:



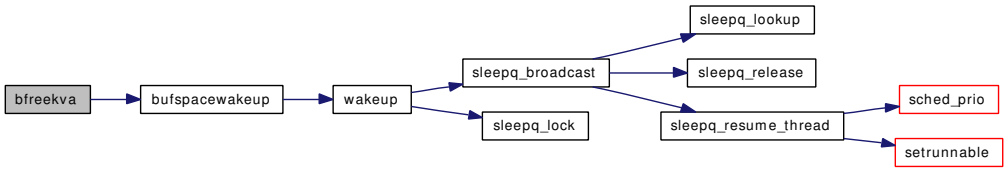
9.152.2.9 static void bfreekva (struct buf * *bp*) [static]

Definition at line 630 of file vfs_bio.c.

References buffreekvacnt, bufspace, and bufspacewakeup().

Referenced by getnewbuf().

Here is the call graph for this function:



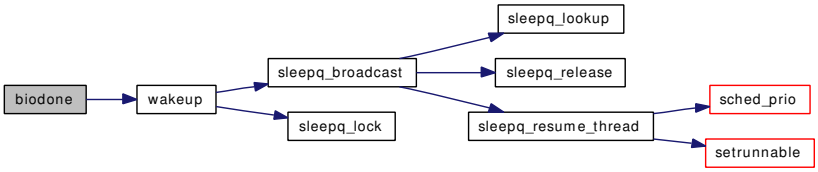
9.152.2.10 void biodone (struct bio * *bp*)

Definition at line 2999 of file vfs_bio.c.

References bdonelock, and wakeup().

Referenced by biofinish(), and bufdone().

Here is the call graph for this function:



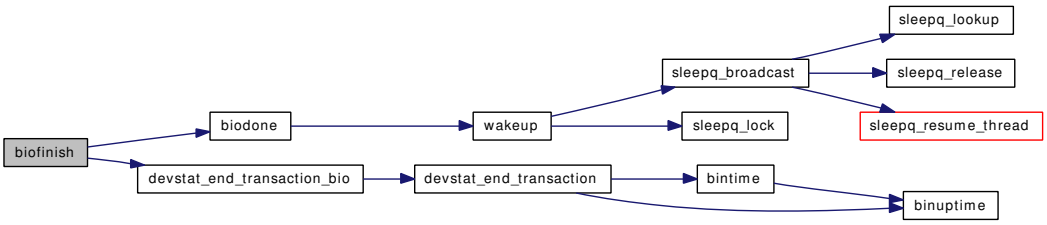
9.152.2.11 void biofinish (struct bio * *bp*, struct devstat * *stat*, int *error*)

Definition at line 3035 of file vfs_bio.c.

References biodone(), and devstat_end_transaction_bio().

Referenced by bioq_flush(), dead_strategy(), and no_strategy().

Here is the call graph for this function:

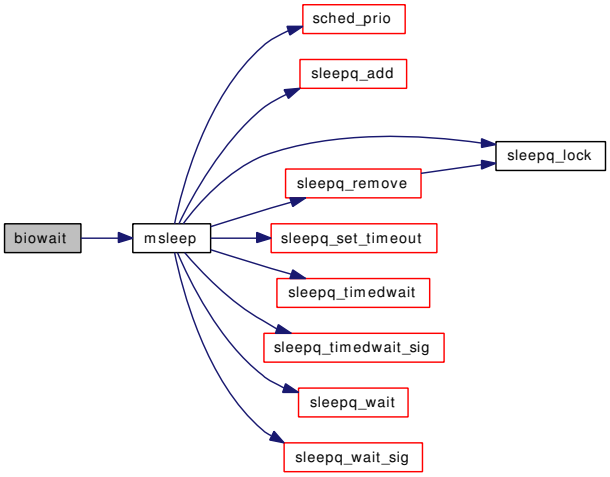


9.152.2.12 int biowait (struct bio * *bp*, const char * *wchan*)

Definition at line 3020 of file vfs_bio.c.

References bdonelock, hz, and msleep().

Here is the call graph for this function:



9.152.2.13 void bpin (struct buf * *bp*)

Definition at line 3880 of file vfs_bio.c.

References bpinlock.

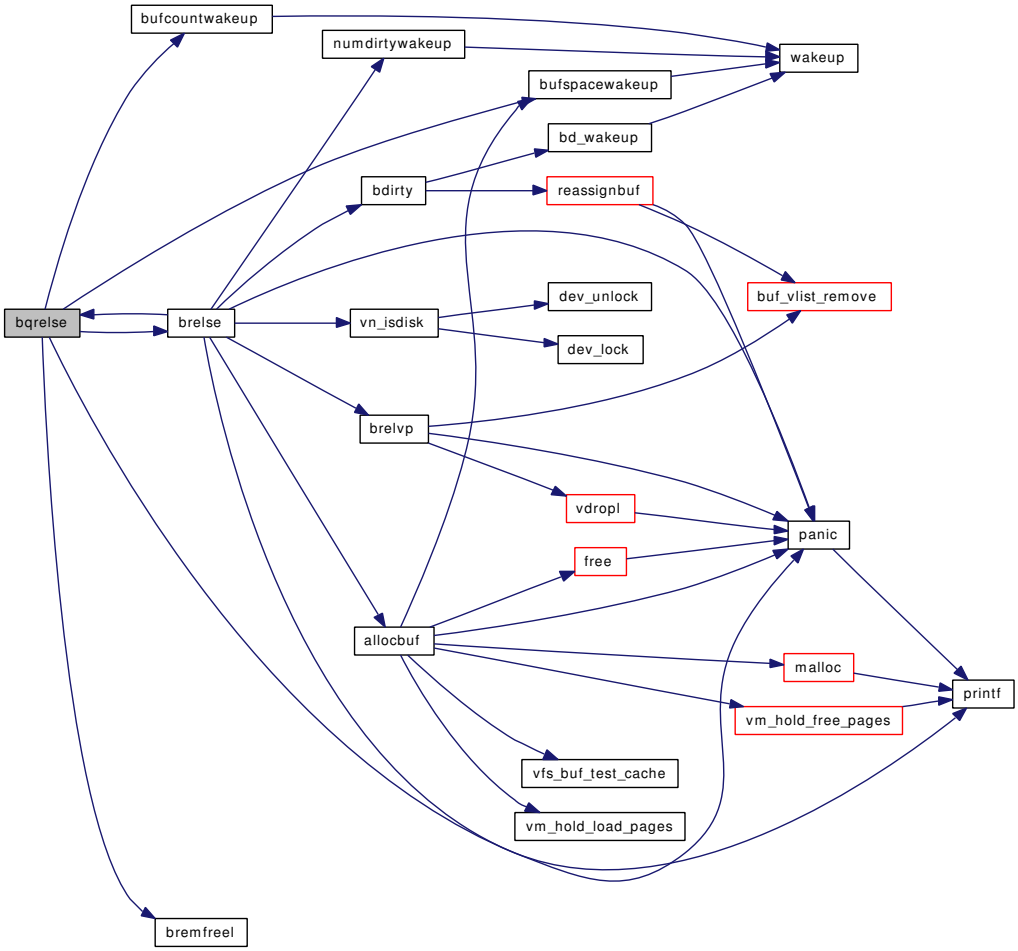
9.152.2.14 void bqrelse (struct buf * *bp*)

Definition at line 1440 of file vfs_bio.c.

References bqlock, brelse(), bremfreel(), bufcountwakeup(), bufspacewakeup(), panic(), QUEUE_CLEAN, QUEUE_DIRTY, QUEUE_DIRTY_GIANT, and QUEUE_NONE.

Referenced by bdwrite(), brelse(), bufdone_finish(), cluster_rbuild(), cluster_read(), and getblk().

Here is the call graph for this function:



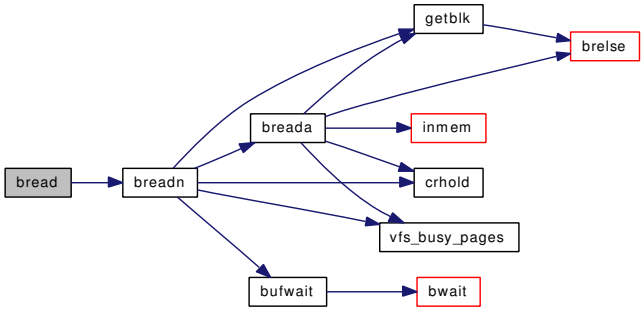
9.152.2.15 int bread (struct vnode * *vp*, daddr_t *blkno*, int *size*, struct ucred * *cred*, struct buf ** *bpp*)

Definition at line 727 of file vfs_bio.c.

References breadn().

Referenced by cluster_collectbufs().

Here is the call graph for this function:



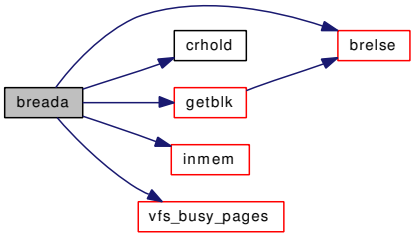
9.152.2.16 void breada (struct vnode * *vp*, daddr_t * *rablkno*, int * *rabsize*, int *cnt*, struct ucred * *cred*)

Definition at line 740 of file vfs_bio.c.

References brelse(), buf, crhold(), getblk(), inmem(), and vfs_busy_pages().

Referenced by breadn().

Here is the call graph for this function:



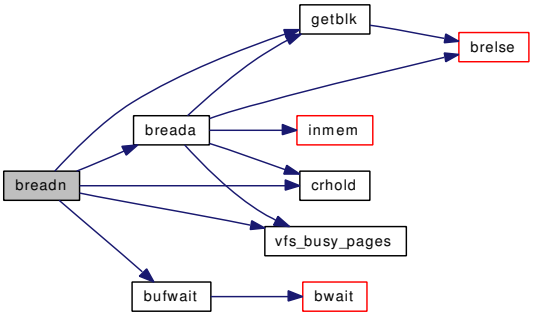
9.152.2.17 int breadn (struct vnode * *vp*, daddr_t *blkno*, int *size*, daddr_t * *rablkno*, int * *rabsize*, int *cnt*, struct ucred * *cred*, struct buf ** *bpp*)

Definition at line 775 of file vfs_bio.c.

References breada(), buf, bufwait(), crhold(), getblk(), and vfs_busy_pages().

Referenced by bread().

Here is the call graph for this function:



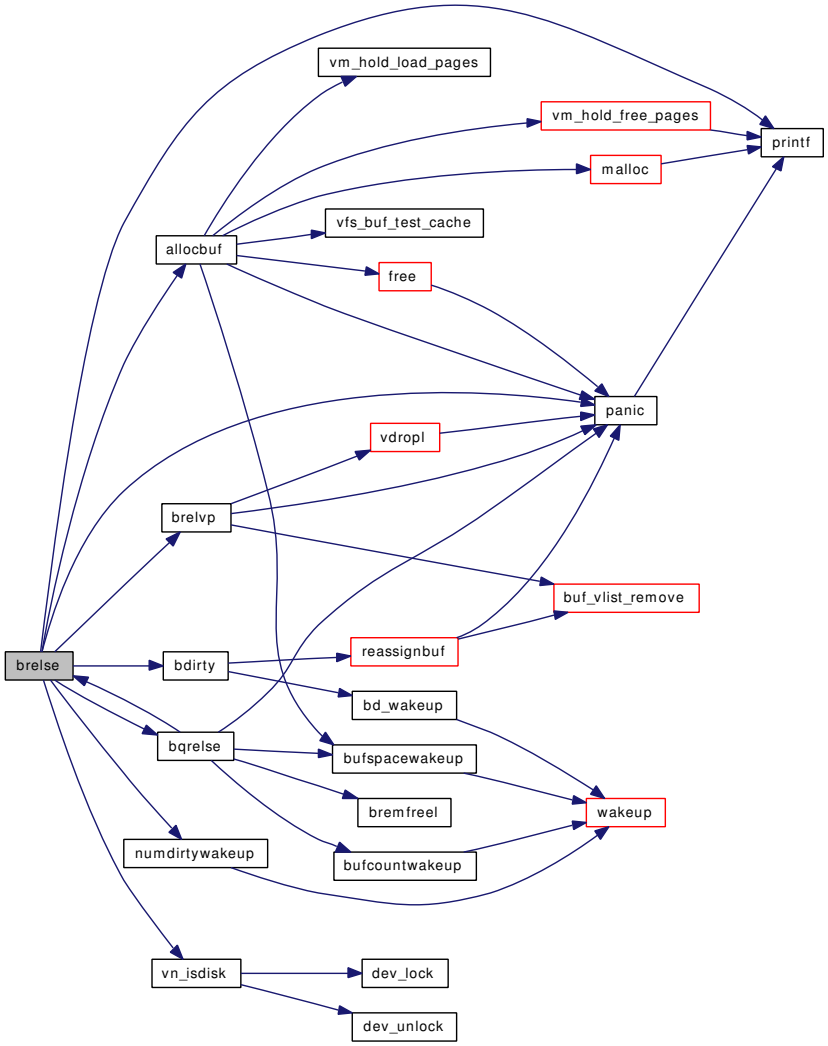
9.152.2.18 void brelse (struct buf * *bp*)

Definition at line 1154 of file vfs_bio.c.

References allocbuf(), bdirty(), bogus_page, bqrelse(), brelvp(), lodirtybuffers, numdirtybuffers, numdirtywakeup(), panic(), printf(), and vn_isdisk().

Referenced by bdwrite(), bqrelse(), breada(), bufdone_finish(), bufwrite(), cluster_rbuild(), cluster_write(), flushbuflist(), flushbufqueues(), getblk(), getnewbuf(), and vtruncbuf().

Here is the call graph for this function:



9.152.2.19 void bremfree (struct buf * *bp*)

Definition at line 654 of file vfs_bio.c.

References numfreebuffers, and QUEUE_NONE.

Referenced by bufbdflush(), cluster_wbuild(), flushbuflist(), getblk(), vfs_bio_awrite(), vop_stdfsync(), and vtruncbuf().

9.152.2.20 void bremfreef (struct buf * *bp*)

Definition at line 677 of file vfs_bio.c.

References bqlock, and bremfreel().

Here is the call graph for this function:

9.152.2.21 static void bremfreel (struct buf * *bp*) [static]

Definition at line 691 of file vfs_bio.c.

References bqlock, numfreebuffers, and QUEUE_NONE.

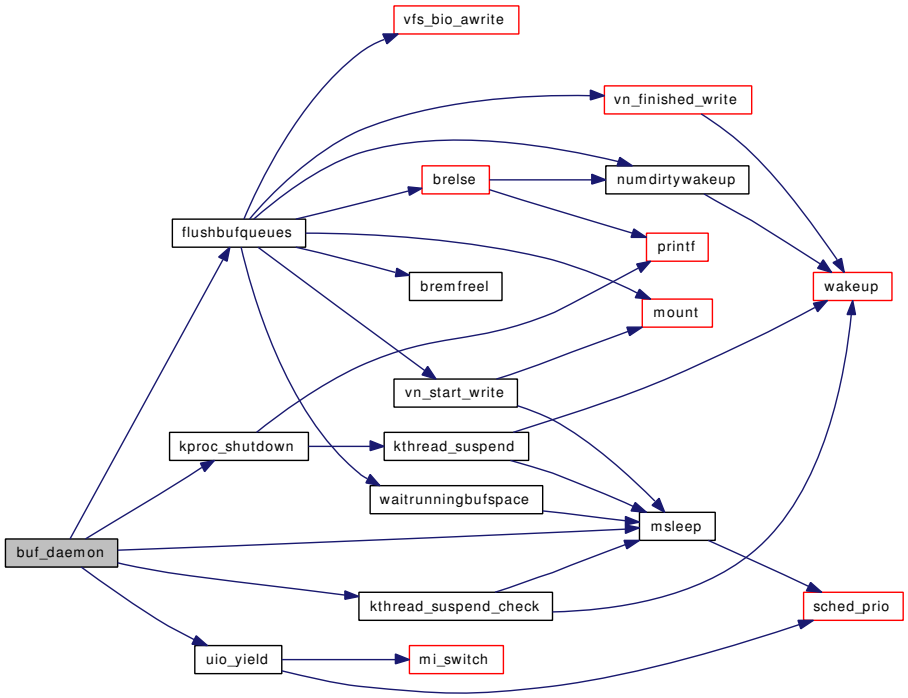
Referenced by bqrelse(), bremfreef(), flushbufqueues(), and getnewbuf().

9.152.2.22 static void buf_daemon (void) [static]

Definition at line 2038 of file vfs_bio.c.

References bd_request, bdlock, flushbufqueues(), Giant, hz, kproc_shutdown(), kthread_suspend_check(), lodirtybuffers, msleep(), numdirtybuffers, QUEUE_DIRTY, QUEUE_DIRTY_GIANT, and uio_yield().

Here is the call graph for this function:



9.152.2.23 int buf_dirty_count_severe (void)

Definition at line 1140 of file vfs_bio.c.

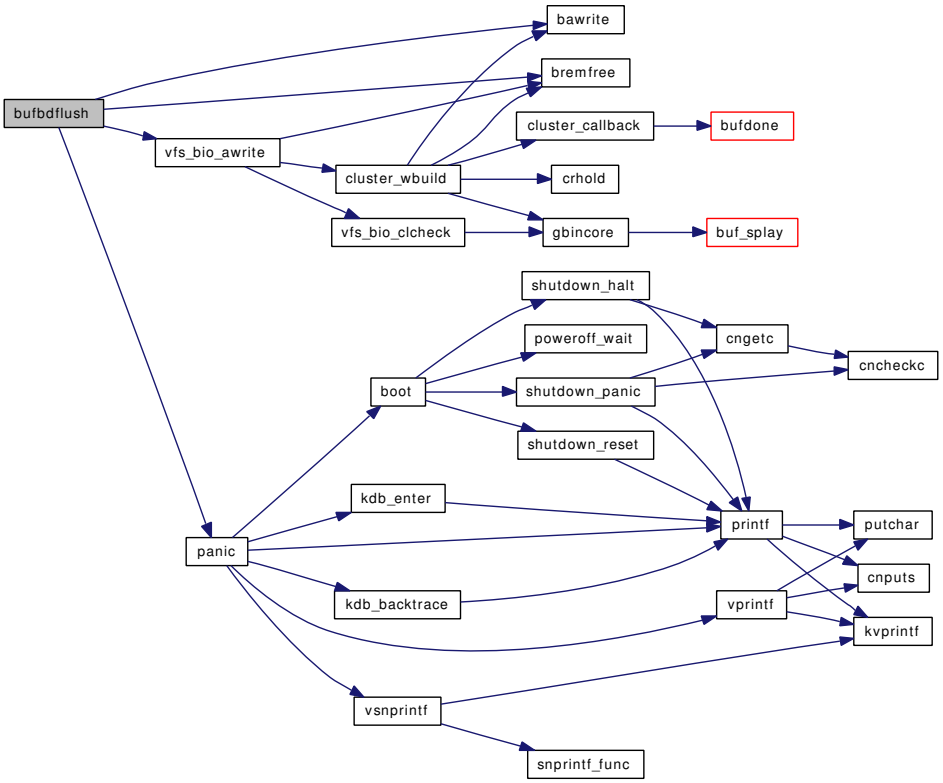
References hidirtybuffers, and numdirtybuffers.

9.152.2.24 void bufbdflush (struct bufobj * *bo*, struct buf * *bp*)

Definition at line 894 of file vfs_bio.c.

References altbufferflushes, bawrite(), bremfree(), buf, dirtybufferflushes, dirtybufthresh, panic(), and vfs_-bio_awrite().

Here is the call graph for this function:



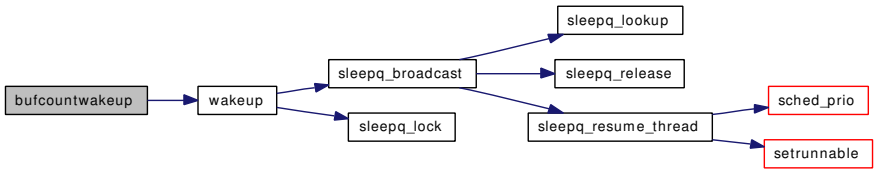
9.152.2.25 static __inline void bufcountwakeup (void) [static]

Definition at line 358 of file vfs_bio.c.

References hifreebuffers, nblock, needsbuffer, numfreebuffers, VFS_BIO_NEED_ANY, VFS_BIO_-NEED_FREE, and wakeup().

Referenced by bqrelse().

Here is the call graph for this function:



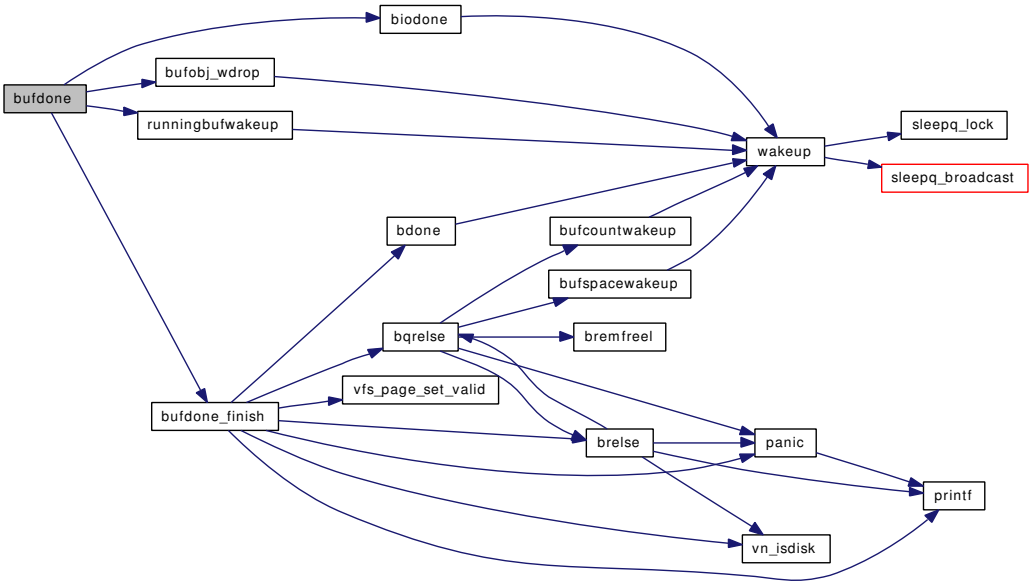
9.152.2.26 void bufdone (struct buf * *bp*)

Definition at line 3149 of file vfs_bio.c.

References biodone(), buf, bufdone_finish(), bufobj_wdrop(), and runningbufwakeup().

Referenced by bufdonebio(), cluster_callback(), dev_strategy(), and vop_nostrategy().

Here is the call graph for this function:



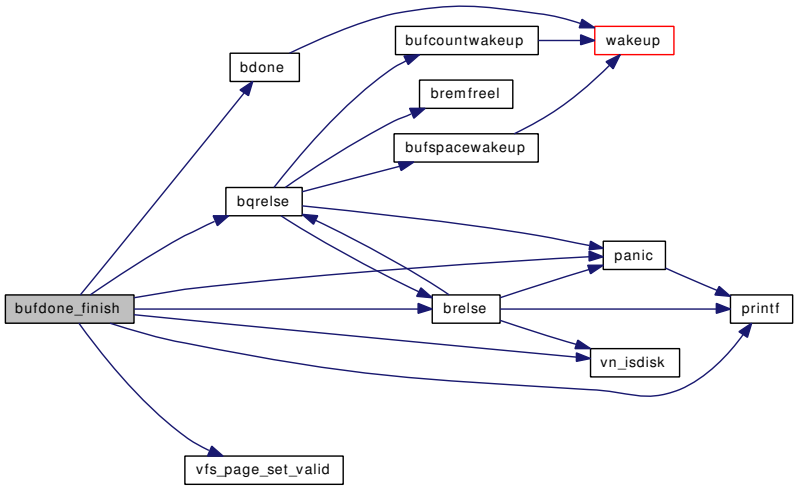
9.152.2.27 void bufdone_finish (struct buf * *bp*)

Definition at line 3181 of file vfs_bio.c.

References bdone(), bogus_page, bqrelse(), brelse(), panic(), printf(), vfs_page_set_valid(), and vn_-isdisk().

Referenced by bufdone().

Here is the call graph for this function:



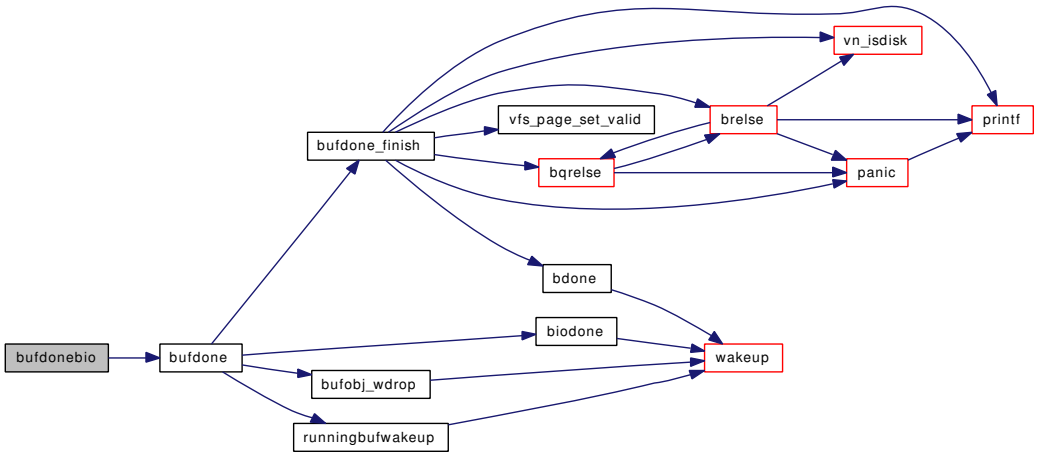
9.152.2.28 static void bufdonebio (struct bio * *bip*) [static]

Definition at line 3076 of file vfs_bio.c.

References buf, and bufdone().

Referenced by dev_strategy().

Here is the call graph for this function:



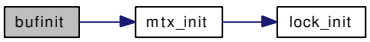
9.152.2.29 void bufinit (void)

Definition at line 527 of file vfs_bio.c.

References bdlock, bdonelock, bogus_page, bpinlock, bqlock, buf, BUFFER_QUEUES, dirtybufthresh, hibufspace, hidirtybuffers, hifreebuffers, hirunningspace, lobufspace, lodirtybuffers, lofreebuffers, lorun-

ningspace, maxbufmallocspace, maxbufspace, mtx_init(), nblock, nbuf, numdirtybuffers, numfreebuffers, QUEUE_EMPTY, and rbreqlock.

Here is the call graph for this function:



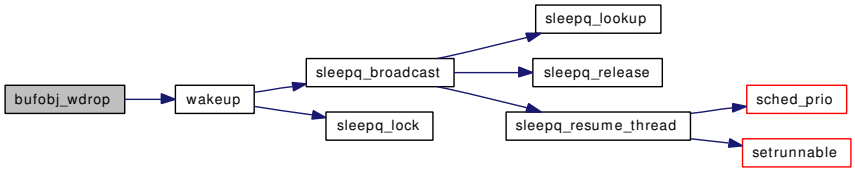
9.152.2.30 void bufobj_wdrop (struct bufobj * *bo*)

Definition at line 3848 of file vfs_bio.c.

References wakeup().

Referenced by bufdone().

Here is the call graph for this function:



9.152.2.31 void bufobj_wref (struct bufobj * *bo*)

Definition at line 3838 of file vfs_bio.c.

Referenced by bufwrite().

9.152.2.32 void bufobj_wrefl (struct bufobj * *bo*)

Definition at line 3829 of file vfs_bio.c.

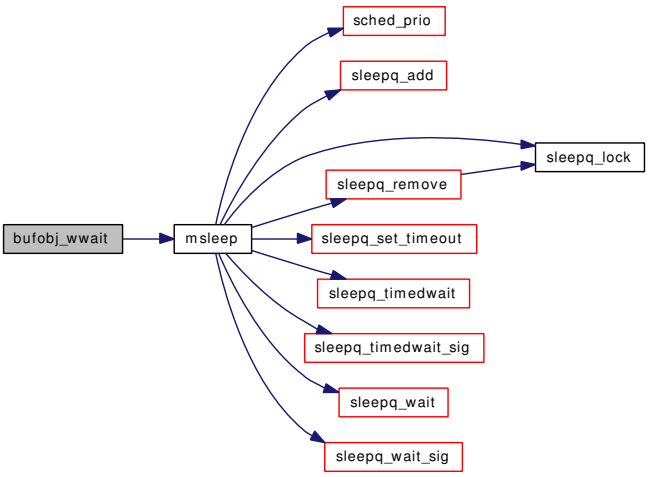
9.152.2.33 int bufobj_wwait (struct bufobj * *bo*, int *slpflag*, int *timeo*)

Definition at line 3862 of file vfs_bio.c.

References msleep().

Referenced by bufobj_invalbuf(), vop_stdfsync(), and vtruncbuf().

Here is the call graph for this function:



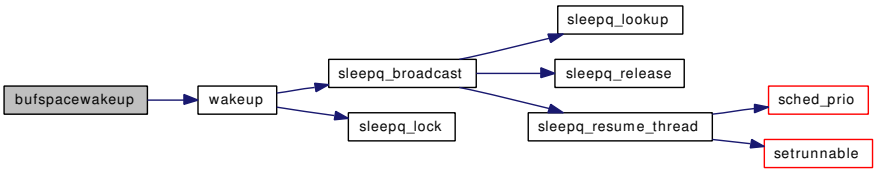
9.152.2.34 static __inline void bufspacewakeup (void) [static]

Definition at line 312 of file vfs_bio.c.

References nblock, needsbuffer, VFS_BIO_NEED_BUFSPACE, and wakeup().

Referenced by allocbuf(), bfreekva(), and bqrelse().

Here is the call graph for this function:



9.152.2.35 void bufstrategy (struct bufobj * *bo*, struct buf * *bp*)

Definition at line 3815 of file vfs_bio.c.

9.152.2.36 int bufsync (struct bufobj * *bo*, int *waitfor*, struct thread * *td*)

Definition at line 3808 of file vfs_bio.c.

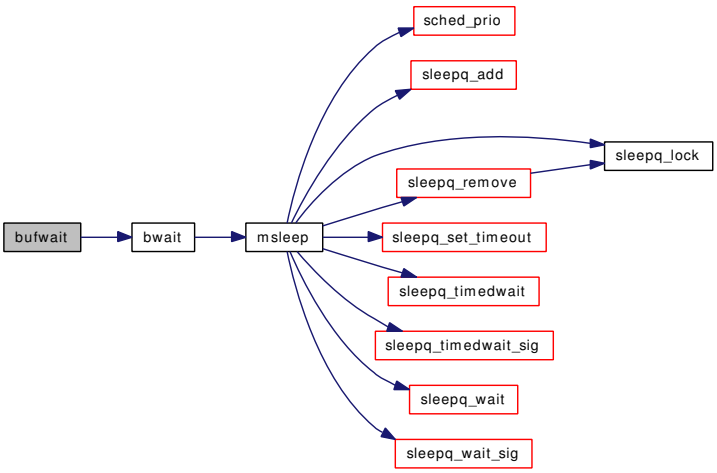
9.152.2.37 int bufwait (struct buf * *bp*)

Definition at line 3055 of file vfs_bio.c.

References bwait().

Referenced by breadn(), bufwrite(), and cluster_read().

Here is the call graph for this function:

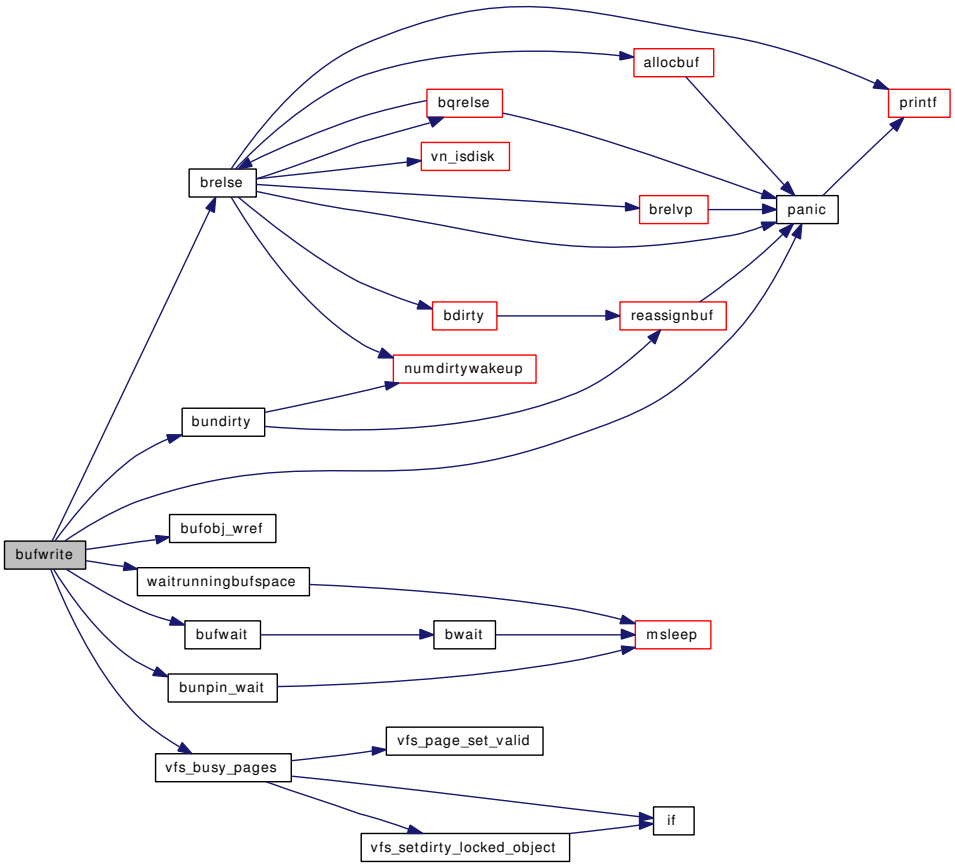


9.152.2.38 int bufwrite (struct buf * *bp*)

Definition at line 820 of file vfs_bio.c.

References brelse(), bufobj_wref(), bufwait(), bundirty(), bunpin_wait(), panic(), runningbufspace, vfs_busy_pages(), and waitrunningbufspace().

Here is the call graph for this function:



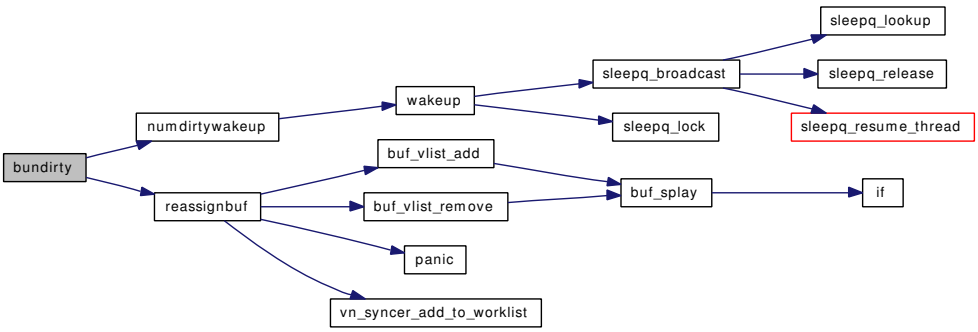
9.152.2.39 void bundirty (struct buf * *bp*)

Definition at line 1072 of file vfs_bio.c.

References lodirtybuffers, numdirtybuffers, numdirtywakeup(), QUEUE_NONE, and reassignbuf().

Referenced by bufwrite().

Here is the call graph for this function:

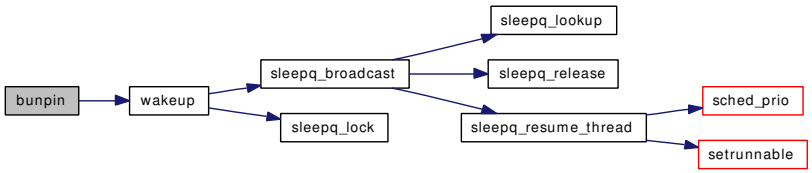


9.152.2.40 void bunpin (struct buf * *bp*)

Definition at line 3888 of file vfs_bio.c.

References bpinlock, and wakeup().

Here is the call graph for this function:



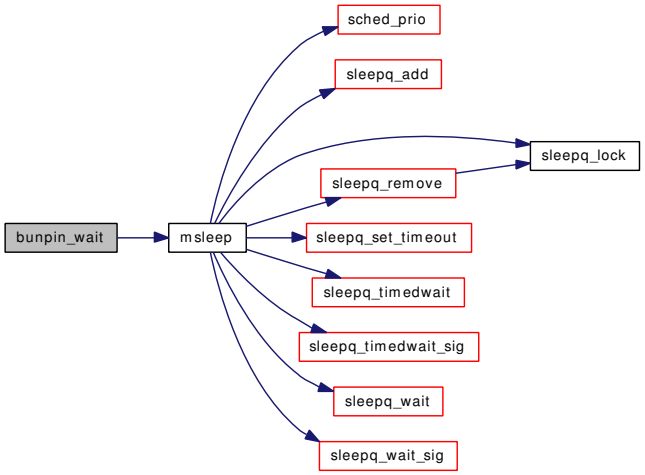
9.152.2.41 void bunpin_wait (struct buf * *bp*)

Definition at line 3897 of file vfs_bio.c.

References bpinlock, and msleep().

Referenced by bufwrite(), and getblk().

Here is the call graph for this function:



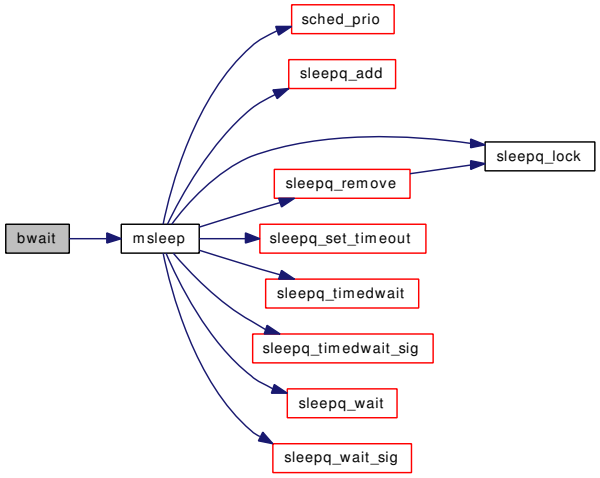
9.152.2.42 void bwait (struct buf * *bp*, u_char *pri*, const char * *wchan*)

Definition at line 3798 of file vfs_bio.c.

References bdonelock, and msleep().

Referenced by bufwait(), and physio().

Here is the call graph for this function:



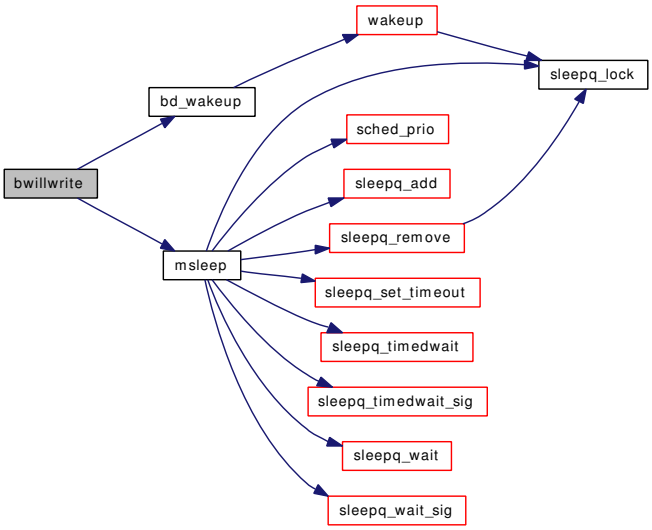
9.152.2.43 void bwillwrite (void)

Definition at line 1121 of file vfs_bio.c.

References bd_wakeup(), hidirtybuffers, msleep(), nblock, needsbuffer, numdirtybuffers, and VFS_BIO_-NEED_DIRTYFLUSH.

Referenced by aio_process(), dofilewrite(), kern_link(), kern_mkdir(), kern_mkfifo(), kern_mknod(), kern_rename(), kern_rmdir(), kern_symlink(), kern_unlink(), undelete(), vn_open_cred(), vn_rdwr_-inchunks(), and vn_write().

Here is the call graph for this function:



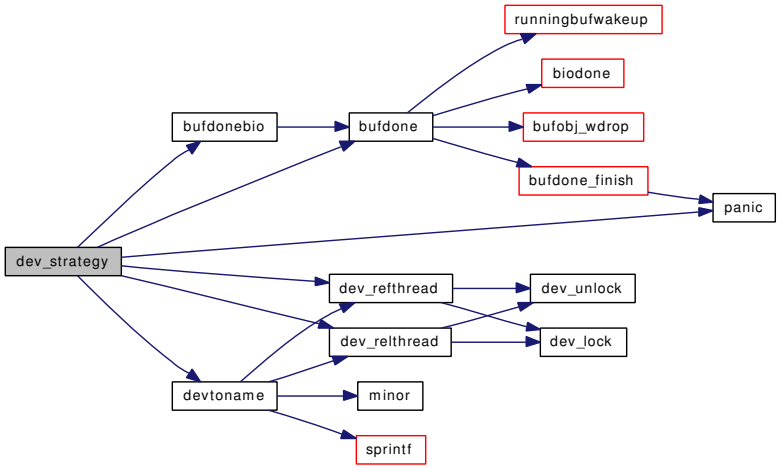
9.152.2.44 void dev_strategy (struct cdev * *dev*, struct buf * *bp*)

Definition at line 3092 of file vfs_bio.c.

References bufdone(), bufdonebio(), dev_refthread(), dev_relthread(), devtoname(), hz, and panic().

Referenced by aio_qphysio(), and physio().

Here is the call graph for this function:



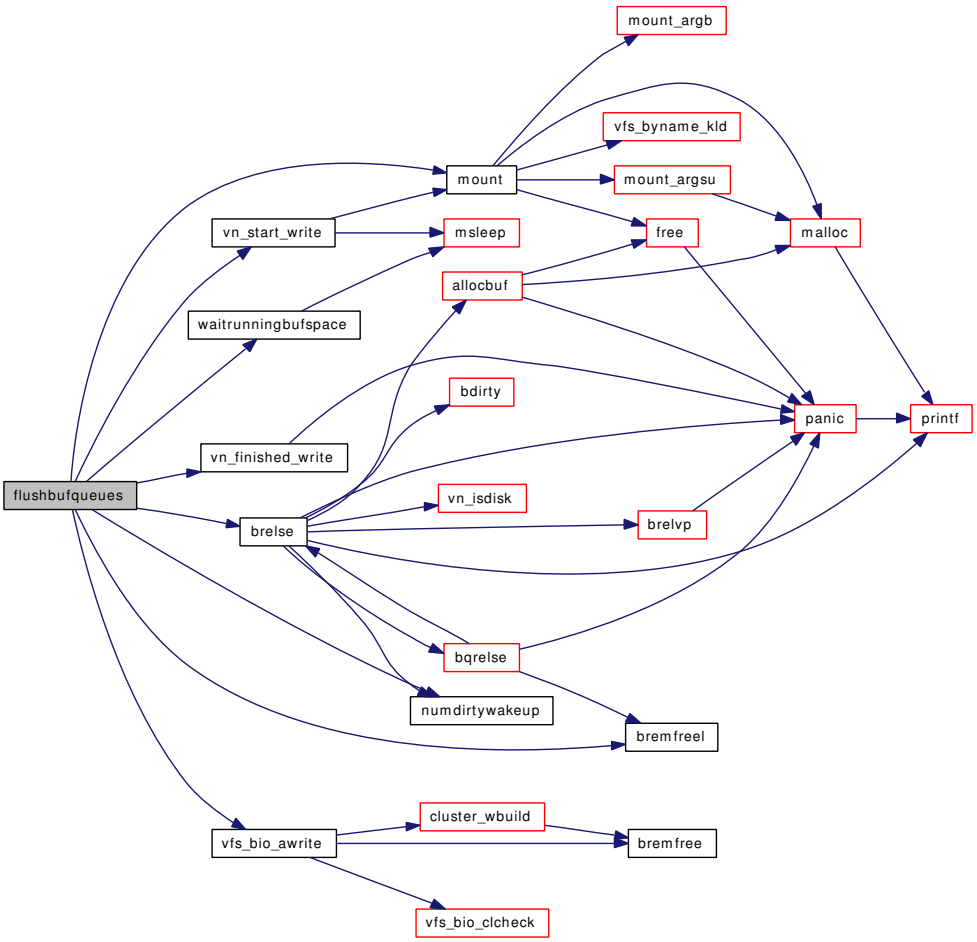
9.152.2.45 static int flushbufqueues (int, int) [static]

Definition at line 2134 of file vfs_bio.c.

References bqlock, brelse(), bremfreel(), buf, hidirtybuffers, lodirtybuffers, mount(), numdirtybuffers, numdirtywakeup(), td, vfs_bio_awrite(), vn_finished_write(), vn_start_write(), and waitrunningbufspace().

Referenced by buf_daemon().

Here is the call graph for this function:



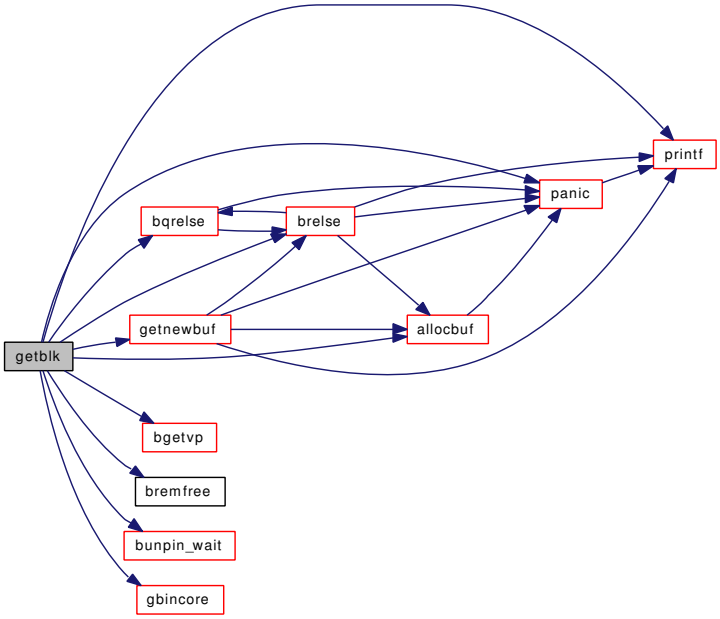
9.152.2.46 struct buf* getblk (struct vnode * *vp*, daddr_t *blkno*, int *size*, int *slpflag*, int *slptimeo*, int *flags*)

Definition at line 2423 of file vfs_bio.c.

References allocbuf(), bgetvp(), bqrelse(), brelse(), bremfree(), buf, bunpin_wait(), gbincore(), getnewbuf(), nblock, needsbuffer, numfreebuffers, panic(), printf(), and VFS_BIO_NEED_ANY.

Referenced by breada(), breadn(), cluster_rbuild(), and cluster_read().

Here is the call graph for this function:

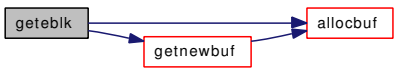


9.152.2.47 struct buf* geteblk (int *size*)

Definition at line 2672 of file vfs_bio.c.

References allocbuf(), buf, and getnewbuf().

Here is the call graph for this function:



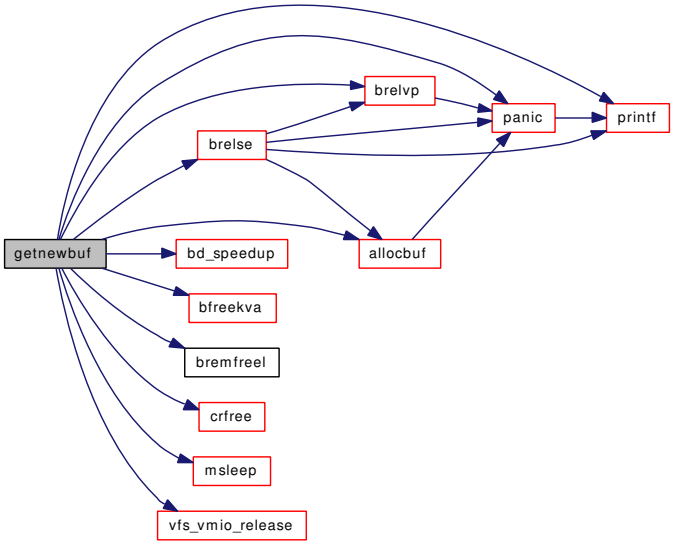
9.152.2.48 static struct buf* getnewbuf (int *slpflag*, int *slptimeo*, int *size*, int *maxsize*) [static]

Definition at line 1701 of file vfs_bio.c.

References allocbuf(), bd_speedup(), bfreekva(), bqlock, brelse(), brelvp(), bremfreel(), buf, bufdefragcnt, bufreusecnt, bufspace, crfree(), getnewbufcalls, getnewbufrestarts, hibufspace, lobufspace, msleep(), nblock, needsbuffer, panic(), printf(), QUEUE_CLEAN, QUEUE_EMPTY, QUEUE_EMPTYKVA, VFS_BIO_NEED_ANY, VFS_BIO_NEED_BUFSPACE, and vfs_vmio_release().

Referenced by getblk(), and geteblk().

Here is the call graph for this function:



9.152.2.49 struct buf* incore (struct bufobj * *bo*, daddr_t *blkno*)

Definition at line 2232 of file vfs_bio.c.

References buf, and gbincore().

Referenced by inmem().

Here is the call graph for this function:

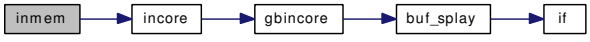
9.152.2.50 static int inmem (struct vnode * *vp*, daddr_t *blkno*) [static]

Definition at line 2249 of file vfs_bio.c.

References incore().

Referenced by breada().

Here is the call graph for this function:

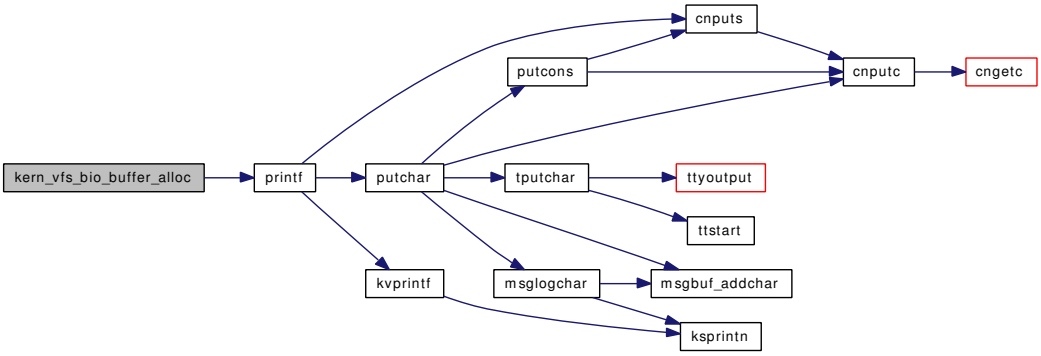


9.152.2.51 caddr_t kern_vfs_bio_buffer_alloc (caddr_t *v*, long *physmem_est*)

Definition at line 455 of file vfs_bio.c.

References buf, maxbcache, nbuf, nswbuf, printf(), and swbuf.

Here is the call graph for this function:



9.152.2.52 static MALLOC_DEFINE (M_BIOBUF, "biobuf", "BIO buffer") [static]

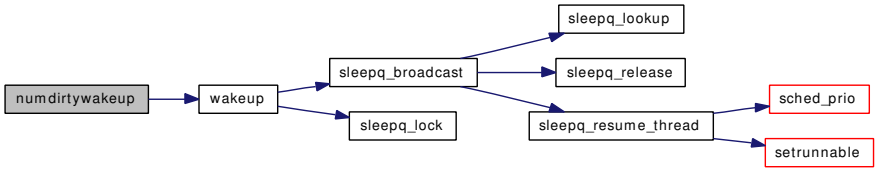
9.152.2.53 static __inline void numdirtywakeup (int *level*) [static]

Definition at line 289 of file vfs_bio.c.

References nblock, needsbuffer, numdirtybuffers, VFS_BIO_NEED_DIRTYFLUSH, and wakeup().

Referenced by brelse(), bundirty(), and flushbufqueues().

Here is the call graph for this function:



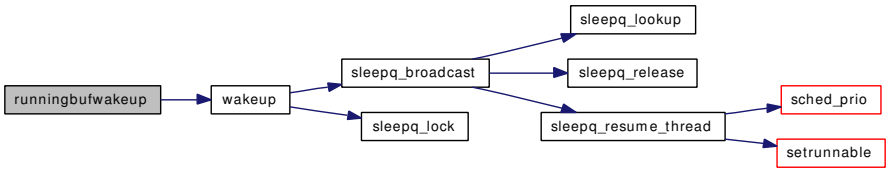
9.152.2.54 void runningbufwakeup (struct buf * *bp*)

Definition at line 333 of file vfs_bio.c.

References lorunningspace, rbreqlock, runningbufreq, runningbufspace, and wakeup().

Referenced by bufdone(), and vfs_unbusy_pages().

Here is the call graph for this function:



9.152.2.55 SYSCTL_INT (_vfs, OID_AUTO, flushwithdeps, CTLFLAG_RW, & *flushwithdeps*, 0, "Number of buffers flushed with dependecies that require rollbacks")

9.152.2.56 SYSCTL_INT (_vfs, OID_AUTO, getnewbufrestarts, CTLFLAG_RW, & *getnewbufrestarts*, 0, "Number of times getnewbuf has had to restart a buffer aquisition")

9.152.2.57 SYSCTL_INT (_vfs, OID_AUTO, getnewbufcalls, CTLFLAG_RW, & *getnewbufcalls*, 0, "Number of calls to getnewbuf")

9.152.2.58 SYSCTL_INT (_vfs, OID_AUTO, hifreebuffers, CTLFLAG_RW, & *hifreebuffers*, 0, "XXX Complicatedly unused")

9.152.2.59 SYSCTL_INT (_vfs, OID_AUTO, lofreebuffers, CTLFLAG_RW, & *lofreebuffers*, 0, "XXX Unused")

9.152.2.60 SYSCTL_INT (_vfs, OID_AUTO, numfreebuffers, CTLFLAG_RD, & *numfreebuffers*, 0, "Number of free buffers")

9.152.2.61 SYSCTL_INT (_vfs, OID_AUTO, dirtybufthresh, CTLFLAG_RW, & *dirtybufthresh*, 0, "Number of bdwrite to bawrite conversions to clear dirty buffers")

9.152.2.62 SYSCTL_INT (_vfs, OID_AUTO, hidirtybuffers, CTLFLAG_RW, & *hidirtybuffers*, 0, "When the number of dirty buffers is considered severe")

9.152.2.63 SYSCTL_INT (_vfs, OID_AUTO, lodirtybuffers, CTLFLAG_RW, & *lodirtybuffers*, 0, "How many buffers we want to have free before bufdaemon can sleep")

9.152.2.64 SYSCTL_INT (_vfs, OID_AUTO, numdirtybuffers, CTLFLAG_RD, & *numdirtybuffers*, 0, "Number of buffers that are dirty (has unwritten changes) at the moment")

9.152.2.65 SYSCTL_INT (_vfs, OID_AUTO, recursiveflushes, CTLFLAG_RW, & *recursiveflushes*, 0, "Number of flushes skipped due to being recursive")

9.152.2.66 SYSCTL_INT (_vfs, OID_AUTO, altbufferflushes, CTLFLAG_RW, & *altbufferflushes*, 0, "Number of fsync flushes to limit dirty buffers")

9.152.2.67 SYSCTL_INT (_vfs, OID_AUTO, bdwriteskip, CTLFLAG_RW, & *bdwriteskip*, 0, "Number of buffers supplied to bdwrite with snapshot deadlock risk")

9.152.2.68 SYSCTL_INT (_vfs, OID_AUTO, dirtybufferflushes, CTLFLAG_RW, & *dirtybufferflushes*, 0, "Number of bdwrite to bawrite conversions to limit dirty buffers")

9.152.2.69 SYSCTL_INT (_vfs, OID_AUTO, hirunningspace, CTLFLAG_RW, & *hirunningspace*, 0, "Maximum amount of space to use for in-progress I/O")

9.152.2.70 SYSCTL_INT (_vfs, OID_AUTO, lorunningspace, CTLFLAG_RW, & *lorunningspace*, 0, "Minimum preferred space used for in-progress I/O")

9.152.2.71 SYSCTL_INT (_vfs, OID_AUTO, bufdefragcnt, CTLFLAG_RW, & *bufdefragcnt*, 0, "Number of times we have had to repeat buffer allocation to defragment")

9.152.2.72 SYSCTL_INT (_vfs, OID_AUTO, buffreekvacnt, CTLFLAG_RW, & *buffreekvacnt*, 0, "Number of times we have freed the KVA space from some buffer")

9.152.2.73 SYSCTL_INT (_vfs, OID_AUTO, bufreusecnt, CTLFLAG_RW, & *bufreusecnt*, 0, "Number of times we have reused a buffer")

9.152.2.74 SYSCTL_INT (_vfs, OID_AUTO, hibufspace, CTLFLAG_RD, & *hibufspace*, 0, "Maximum allowed value of bufspace (excluding buf_daemon)")

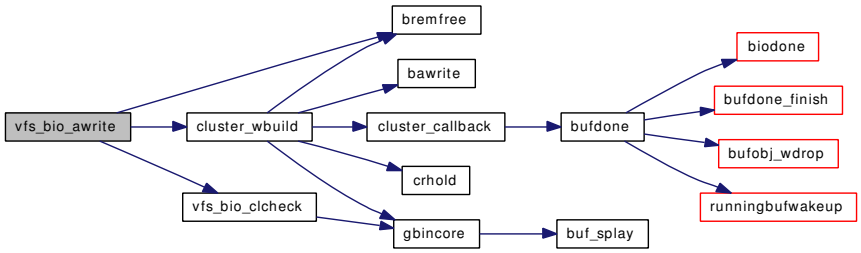
9.152.2.83 int vfs_bio_awrite (struct buf * *bp*)

Definition at line 1621 of file vfs_bio.c.

References bremfree(), cluster_wbuild(), and vfs_bio_clcheck().

Referenced by bufbdflush(), flushbufqueues(), and vop_stdfsync().

Here is the call graph for this function:



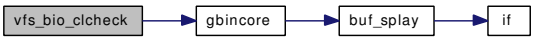
9.152.2.84 static int vfs_bio_clcheck (struct vnode * *vp*, int *size*, daddr_t *lblkno*, daddr_t *blkno*) [static]

Definition at line 1578 of file vfs_bio.c.

References buf, and gbincore().

Referenced by vfs_bio_awrite().

Here is the call graph for this function:



9.152.2.85 void vfs_bio_clrbuf (struct buf * *bp*)

Definition at line 3568 of file vfs_bio.c.

References bogus_page, if(), and mask.

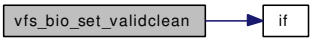
Here is the call graph for this function:

9.152.2.86 void vfs_bio_set_validclean (struct buf * *bp*, int *base*, int *size*)

Definition at line 3526 of file vfs_bio.c.

References if().

Here is the call graph for this function:



9.152.2.87 static __inline void vfs_buf_test_cache (struct buf * *bp*, vm_ooffset_t *foff*, vm_offset_t *off*, vm_offset_t *size*, vm_page_t *m*) [static]

Definition at line 409 of file vfs_bio.c.

Referenced by allocbuf().

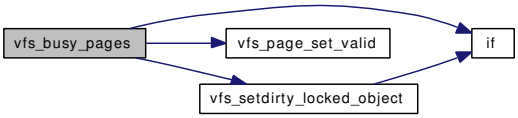
9.152.2.88 void vfs_busy_pages (struct buf * *bp*, int *clear_modify*)

Definition at line 3413 of file vfs_bio.c.

References bogus_page, if(), vfs_page_set_valid(), and vfs_setdirty_locked_object().

Referenced by breada(), breadn(), bufwrite(), and cluster_read().

Here is the call graph for this function:



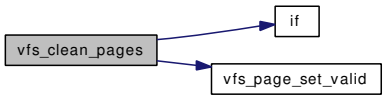
9.152.2.89 static void vfs_clean_pages (struct buf * *bp*) [static]

Definition at line 3487 of file vfs_bio.c.

References if(), and vfs_page_set_valid().

Referenced by bdwrite().

Here is the call graph for this function:



9.152.2.90 static void vfs_page_set_valid (struct buf * *bp*, vm_ooffset_t *off*, int *pageno*, vm_page_t *m*) [static]

Definition at line 3371 of file vfs_bio.c.

Referenced by bufdone_finish(), vfs_busy_pages(), and vfs_clean_pages().

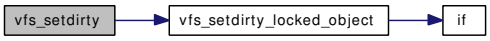
9.152.2.91 static void vfs_setdirty (struct buf * *bp*) [static]

Definition at line 2303 of file vfs_bio.c.

References vfs_setdirty_locked_object().

Referenced by bdwrite().

Here is the call graph for this function:



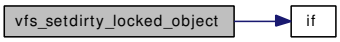
9.152.2.92 static void vfs_setdirty_locked_object (struct buf * *bp*) [static]

Definition at line 2327 of file vfs_bio.c.

References if().

Referenced by vfs_busy_pages(), and vfs_setdirty().

Here is the call graph for this function:

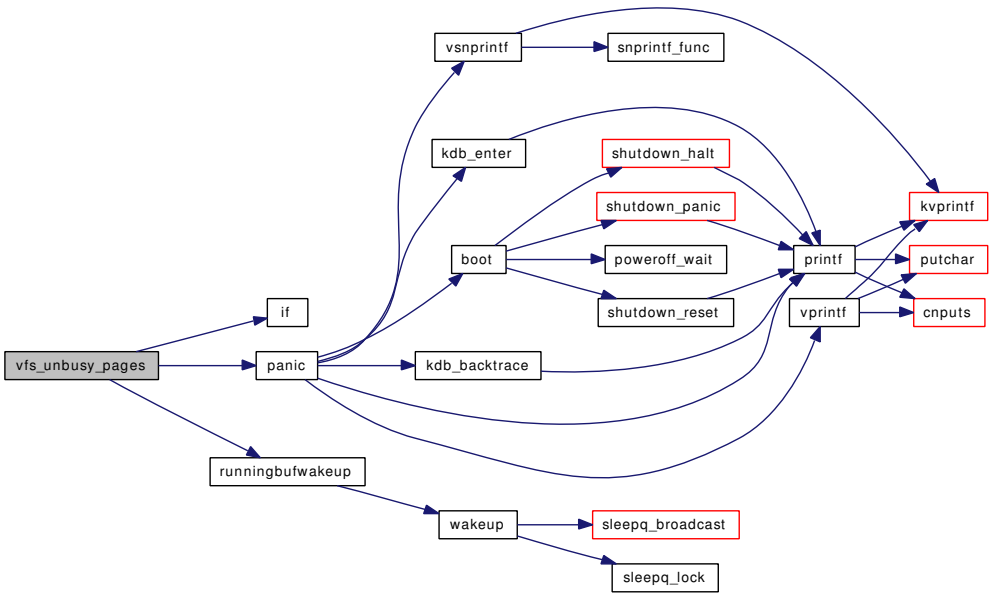


9.152.2.93 void vfs_unbusy_pages (struct buf * *bp*)

Definition at line 3333 of file vfs_bio.c.

References bogus_page, if(), panic(), and runningbufwakeup().

Here is the call graph for this function:



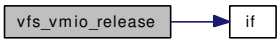
9.152.2.94 static void vfs_vmio_release (struct buf * *bp*) [static]

Definition at line 1520 of file vfs_bio.c.

References if().

Referenced by getnewbuf().

Here is the call graph for this function:



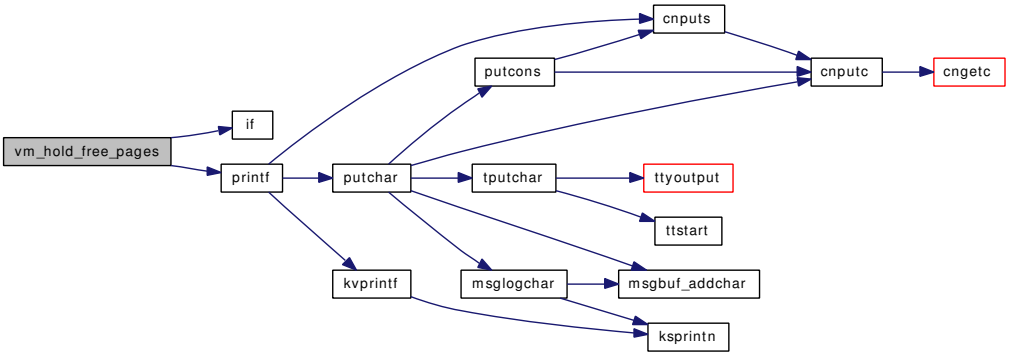
9.152.2.95 static void vm_hold_free_pages (struct buf * *bp*, vm_offset_t *from*, vm_offset_t *to*) [static]

Definition at line 3671 of file vfs_bio.c.

References if(), and printf().

Referenced by allocbuf().

Here is the call graph for this function:



9.152.2.96 static void vm_hold_load_pages (struct buf * *bp*, vm_offset_t *from*, vm_offset_t *to*) [static]

Definition at line 3632 of file vfs_bio.c.

Referenced by allocbuf().

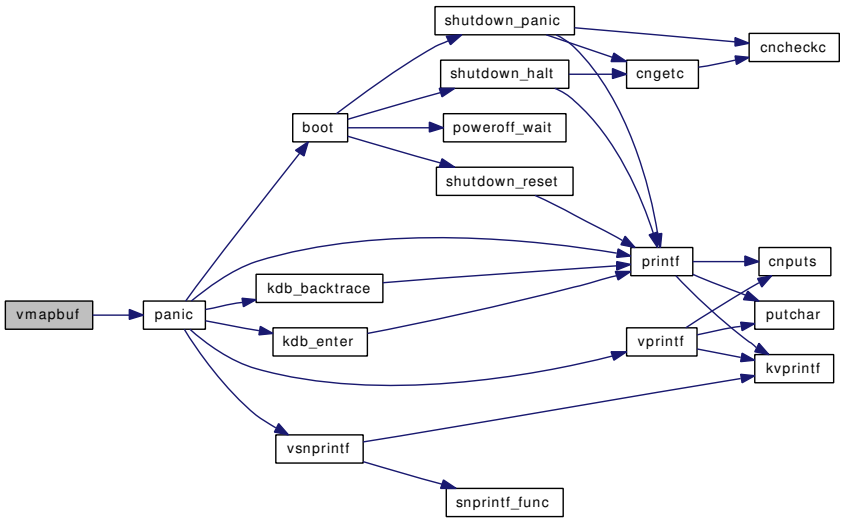
9.152.2.97 int vmapbuf (struct buf * *bp*)

Definition at line 3716 of file vfs_bio.c.

References panic().

Referenced by aio_qphysio(), and physio().

Here is the call graph for this function:



9.152.2.98 void vunmapbuf (struct buf * *bp*)

Definition at line 3772 of file vfs_bio.c.

Referenced by biohelper(), and physio().

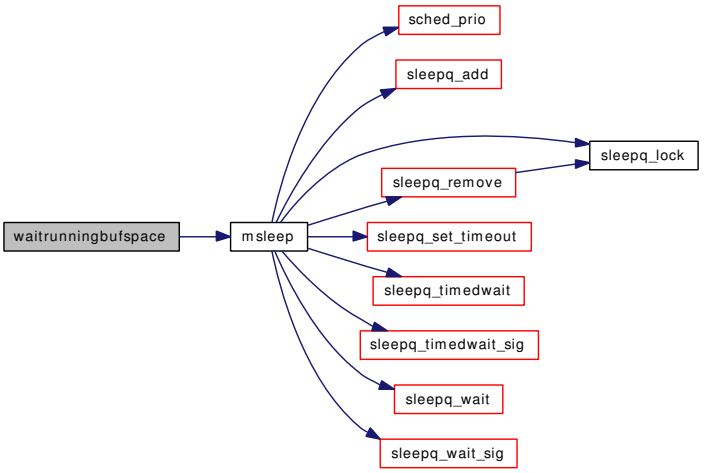
9.152.2.99 void waitrunningbufspace (void)

Definition at line 388 of file vfs_bio.c.

References hirunningspace, msleep(), rbreqlock, runningbufreq, and runningbufspace.

Referenced by bufwrite(), and flushbufqueues().

Here is the call graph for this function:



9.152.3 Variable Documentation

9.152.3.1 int altbufferflushes

Definition at line 156 of file vfs_bio.c.

Referenced by bufbdflush().

9.152.3.2 int bd_request [static]

Definition at line 195 of file vfs_bio.c.

Referenced by bd_wakeup(), and buf_daemon().

9.152.3.3 struct mtx bdlock [static]

Definition at line 200 of file vfs_bio.c.

Referenced by bd_wakeup(), buf_daemon(), and bufinit().

9.152.3.4 struct mtx bdonelock [static]

Definition at line 242 of file vfs_bio.c.

Referenced by bdone(), biodone(), biowait(), bufinit(), and bwait().

9.152.3.5 int bdwriteskip

Definition at line 153 of file vfs_bio.c.

9.152.3.6 struct bio_ops bioops

Definition at line 76 of file vfs_bio.c.

9.152.3.7 vm_page_t bogus_page

Definition at line 209 of file vfs_bio.c.

Referenced by allocbuf(), brelse(), bufdone_finish(), bufinit(), cluster_rbuild(), vfs_bio_clrbuf(), vfs_busy_pages(), and vfs_unbusy_pages().

9.152.3.8 struct mtx bpinlock [static]

Definition at line 247 of file vfs_bio.c.

Referenced by bpin(), bufinit(), bunpin(), and bunpin_wait().

9.152.3.9 struct mtx bqlock [static]

Definition at line 265 of file vfs_bio.c.

Referenced by bqrelse(), bremfreef(), bremfreel(), bufinit(), flushbufqueues(), and getnewbuf().

9.152.3.10 struct buf* buf

Definition at line 90 of file vfs_bio.c.

Referenced by aio_qphysio(), biohelper(), breada(), breadn(), buf_splay(), buf_vlist_add(), buf_vlist_remove(), bufbdflush(), bufdone(), bufdonebio(), bufinit(), cluster_callback(), cluster_collectbufs(), cluster_rbuild(), cluster_read(), cluster_wbuild(), cluster_write(), device_sysctl_handler(), expand_name(), flushbuflist(), flushbufqueues(), gbincore(), getblk(), geteblk(), getenv(), getnewbuf(), incore(), kdb_sysctl_current(), kern_alternate_path(), kern_ptrace(), kern_vfs_bio_buffer_alloc(), logread(), mqfs_read(), panic(), physio(), printf_uuid(), ptcread(), sbuf_printf_uuid(), setenv(), sysctl_kern_msgbuf(), sysctl_kern_timecounter_choice(), sysctl_sysctl_name(), uipc_bind(), unp_connect(), vfs_bio_clcheck(), vn_fullpath(), vn_printf(), vop_stdfsync(), vop_strategy_pre(), vsscanf(), and vtruncbuf().

9.152.3.11 struct kproc_desc buf_kp [static]

Initial value:

```
{
        "bufdaemon",
        buf_daemon,
        &bufdaemonproc
}
```

Definition at line 2030 of file vfs_bio.c.

9.152.3.12 struct buf_ops buf_ops_bio

Initial value:

```
{
        .bop_name       =       "buf_ops_bio",
        .bop_write      =       bufwrite,
        .bop_strategy   =       bufstrategy,
        .bop_sync       =       bufsync,
        .bop_bdflush    =       bufbdflush,
}
```

Definition at line 78 of file vfs_bio.c.

Referenced by getnewvnode().

9.152.3.13 const char* buf_wmesg = BUF_WMESG

Definition at line 271 of file vfs_bio.c.

9.152.3.14 struct proc* bufdaemonproc [static]

Definition at line 92 of file vfs_bio.c.

9.152.3.15 int bufdefragcnt [static]

Definition at line 141 of file vfs_bio.c.

Referenced by getnewbuf().

9.152.3.16 int buffreekvacnt [static]

Definition at line 138 of file vfs_bio.c.

Referenced by bfreekva().

9.152.3.17 int bufmallocspace [static]

Definition at line 123 of file vfs_bio.c.

Referenced by allocbuf().

9.152.3.18 int bufreusecnt [static]

Definition at line 135 of file vfs_bio.c.

Referenced by getnewbuf().

9.152.3.19 int bufspace [static]

Definition at line 117 of file vfs_bio.c.

Referenced by bfreekva(), and getnewbuf().

9.152.3.20 int dirtybufferflushes

Definition at line 150 of file vfs_bio.c.

Referenced by bufbdflush().

9.152.3.21 int dirtybufthresh

Definition at line 171 of file vfs_bio.c.

Referenced by bufbdflush(), and bufinit().

9.152.3.22 int flushwithdeps = 0 [static]

Definition at line 2129 of file vfs_bio.c.

9.152.3.23 int getnewbufcalls [static]

Definition at line 183 of file vfs_bio.c.

Referenced by getnewbuf().

9.152.3.24 int getnewbufrestarts [static]

Definition at line 186 of file vfs_bio.c.

Referenced by getnewbuf().

9.152.3.25 int hibufspace

Definition at line 132 of file vfs_bio.c.

Referenced by bufinit(), and getnewbuf().

9.152.3.26 int hidirtybuffers [static]

Definition at line 168 of file vfs_bio.c.

Referenced by bdirty(), bdwrite(), buf_dirty_count_severe(), bufinit(), bwillwrite(), and flushbufqueues().

9.152.3.27 int hifreebuffers [static]

Definition at line 180 of file vfs_bio.c.

Referenced by bufcountwakeup(), and bufinit().

9.152.3.28 int hirunningspace [static]

Definition at line 147 of file vfs_bio.c.

Referenced by bufinit(), and waitrunningbufspace().

9.152.3.29 int lobufspace [static]

Definition at line 129 of file vfs_bio.c.

Referenced by bufinit(), and getnewbuf().

9.152.3.30 int lodirtybuffers [static]

Definition at line 165 of file vfs_bio.c.

Referenced by bdirty(), bdwrite(), brelse(), buf_daemon(), bufinit(), bundirty(), and flushbufqueues().

9.152.3.31 int lofreebuffers [static]

Definition at line 177 of file vfs_bio.c.

Referenced by bufinit().

9.152.3.32 int lorunningspace [static]

Definition at line 144 of file vfs_bio.c.

Referenced by bufinit(), and runningbufwakeup().

9.152.3.33 int maxbufmallocspace [static]

Definition at line 126 of file vfs_bio.c.

Referenced by allocbuf(), and bufinit().

9.152.3.34 int maxbufspace [static]

Definition at line 120 of file vfs_bio.c.

Referenced by bufinit().

9.152.3.35 struct mtx nblock [static]

Definition at line 236 of file vfs_bio.c.

Referenced by bufcountwakeup(), bufinit(), bufspacewakeup(), bwillwrite(), getblk(), getnewbuf(), and numdirtywakeup().

9.152.3.36 int needsbuffer [static]

Definition at line 231 of file vfs_bio.c.

Referenced by bufcountwakeup(), bufspacewakeup(), bwillwrite(), getblk(), getnewbuf(), and numdirtywakeup().

9.152.3.37 int numdirtybuffers [static]

Definition at line 162 of file vfs_bio.c.

Referenced by bd_wakeup(), bdirty(), brelse(), buf_daemon(), buf_dirty_count_severe(), bufinit(), bundirty(), bwillwrite(), flushbufqueues(), and numdirtywakeup().

9.152.3.38 int numfreebuffers [static]

Definition at line 174 of file vfs_bio.c.

Referenced by bremfree(), bremfreel(), bufcountwakeup(), bufinit(), and getblk().

9.152.3.39 struct mtx rbreqlock [static]

Definition at line 222 of file vfs_bio.c.

Referenced by bufinit(), runningbufwakeup(), and waitrunningbufspace().

9.152.3.40 int recursiveflushes [static]

Definition at line 159 of file vfs_bio.c.

Referenced by bdwrite().

9.152.3.41 int runningbufreq [static]

Definition at line 216 of file vfs_bio.c.

Referenced by runningbufwakeup(), and waitrunningbufspace().

9.152.3.42 int runningbufspace

Definition at line 114 of file vfs_bio.c.

Referenced by bufwrite(), runningbufwakeup(), and waitrunningbufspace().

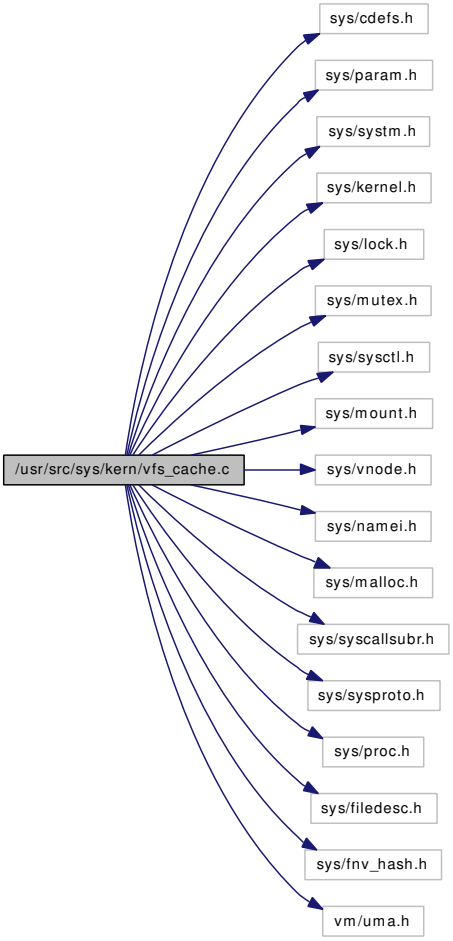
9.152.3.43 int vmiodirenable = TRUE

Definition at line 111 of file vfs_bio.c.

9.153 /usr/src/sys/kern/vfs_cache.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/sysctl.h>
#include <sys/mount.h>
#include <sys/vnode.h>
#include <sys/namei.h>
#include <sys/malloc.h>
#include <sys/syscallsubr.h>
#include <sys/sysproto.h>
#include <sys/proc.h>
#include <sys/filedesc.h>
#include <sys/fnv_hash.h>
#include <vm/uma.h>
```

Include dependency graph for vfs_cache.c:



Data Structures

- struct namecache
- struct __getcwd_args

Defines

- #define NCHHASH(hash) (&nchashtbl[(hash) & nchash])
- #define CACHE_LOCK() mtx_lock(&cache_lock)
- #define CACHE_UNLOCK() mtx_unlock(&cache_lock)
- #define CACHE_PATH_CUTOFF 32
- #define CACHE_ZONE_SMALL (sizeof(struct namecache) + CACHE_PATH_CUTOFF)
- #define CACHE_ZONE_LARGE (sizeof(struct namecache) + NAME_MAX)
- #define cache_alloc(len)
- #define cache_free(ncp)
- #define STATNODE(mode, name, var) SYSCTL_ULONG(_vfs_cache, OID_AUTO, name, mode, var, 0, "");
- #define NCF_WHITE 1
- #define STATNODE(name)

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/vfs_cache.c,v 1.106 2006/06/16 05:09:28 csjp Exp $")
- static LIST_HEAD (nchashhead, SYSCTL_INT(namecache)
- SYSCTL_PROC (_debug_hashstat, OID_AUTO, rawnchash, CTLTYPE_INT|CTLFLAG_RD, 0, 0, sysctl_debug_hashstat_rawnchash,"S,int","nchash chain lengths")
- static int sysctl_debug_hashstat_nchash (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_debug_hashstat, OID_AUTO, nchash, CTLTYPE_INT|CTLFLAG_RD, 0, 0, sysctl_debug_hashstat_nchash,"I","nchash chain lengths")
- static void cache_zap (struct namecache *ncp)
- int cache_leaf_test (struct vnode *vp)
- int cache_lookup (struct vnode *dvp, struct vnode **vpp, struct componentname *cnp)
- void cache_enter (struct vnode *dvp, struct vnode *vp, struct componentname *cnp)
- static void nchinit (void *dummy __unused)
- void cache_purge (struct vnode *vp)
- void cache_purgevfs (struct mount *mp)
- int vfs_cache_lookup (struct vop_lookup_args *ap)
- SYSCTL_INT (_debug, OID_AUTO, disablecwd, CTLFLAG_RW,&disablecwd, 0,"Disable the getcwd syscall")
- int __getcwd (struct thread *td, struct __getcwd_args *uap)
- int kern___getcwd (struct thread *td, u_char *buf, enum uio_seg bufseg, u_int buflen)
- SYSCTL_INT (_debug, OID_AUTO, disablefullpath, CTLFLAG_RW,&disablefullpath, 0,"Disable the vn_fullpath function")
- STATNODE (numfullpathcalls)
- STATNODE (numfullpathfail1)
- STATNODE (numfullpathfail2)
- STATNODE (numfullpathfail4)
- STATNODE (numfullpathfound)
- int vn_fullpath (struct thread *td, struct vnode *vn, char **retbuf, char **freebuf)
- static int vn_fullpath1 (struct thread *td, struct vnode *vp, struct vnode *rdir, char *buf, char **retbuf, u_int buflen)

Variables

- static int disablecwd
- static int disablefullpath

9.153.1 Define Documentation

9.153.1.1 #define cache_alloc(len)

Value:

```
uma_zalloc(((len) <= CACHE_PATH_CUTOFF) ? \
        cache_zone_small : cache_zone_large, M_WAITOK)
```

Referenced by cache_enter().

9.153.1.2 #define cache_free(ncp)

Value:

```
do { \
        if (ncp != NULL) \
                uma_zfree(((ncp)->nc_nlen <= CACHE_PATH_CUTOFF) ? \
                    cache_zone_small : cache_zone_large, (ncp)); \
} while (0)
```

Referenced by cache_zap().

9.153.1.3 #define CACHE_LOCK() mtx_lock(&cache_lock)

Referenced by cache_enter(), cache_leaf_test(), cache_lookup(), cache_purge(), cache_purgevfs(), and vn_fullpath1().

9.153.1.4 #define CACHE_PATH_CUTOFF 32

9.153.1.5 #define CACHE_UNLOCK() mtx_unlock(&cache_lock)

Referenced by cache_enter(), cache_leaf_test(), cache_lookup(), cache_purge(), and vn_fullpath1().

9.153.1.6 #define CACHE_ZONE_LARGE (sizeof(struct namecache) + NAME_MAX)

Referenced by nchinit().

9.153.1.7 #define CACHE_ZONE_SMALL (sizeof(struct namecache) + CACHE_PATH_CUTOFF)

Referenced by nchinit().

9.153.1.8 #define NCF_WHITE 1

Referenced by cache_enter(), and cache_lookup().

9.153.1.9 #define NCHHASH(hash) (&nchashtbl[(hash) & nchash])

Definition at line 92 of file vfs_cache.c.

Referenced by cache_enter(), and cache_lookup().

9.153.1.10 #define STATNODE(name)

Value:

```
static u_int name;                                              \
        SYSCTL_UINT(_vfs_cache, OID_AUTO, name, CTLFLAG_RD, &name, 0, "")
```

Definition at line 741 of file vfs_cache.c.

9.153.1.11 #define STATNODE(mode, name, var) SYSCTL_ULONG(_vfs_cache, OID_AUTO, name, mode, var, 0, "");

Definition at line 741 of file vfs_cache.c.

9.153.2 Function Documentation

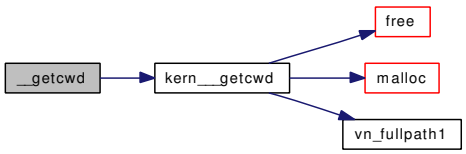
9.153.2.1 __FBSDID ("$FreeBSD: src/sys/kern/vfs_cache. *c*, v 1.106 2006/06/16 05:09:28 csjp Exp $")

9.153.2.2 int __getcwd (struct thread * *td*, struct __getcwd_args * *uap*)

Definition at line 695 of file vfs_cache.c.

References __getcwd_args::buf, __getcwd_args::buflen, and kern___getcwd().

Here is the call graph for this function:



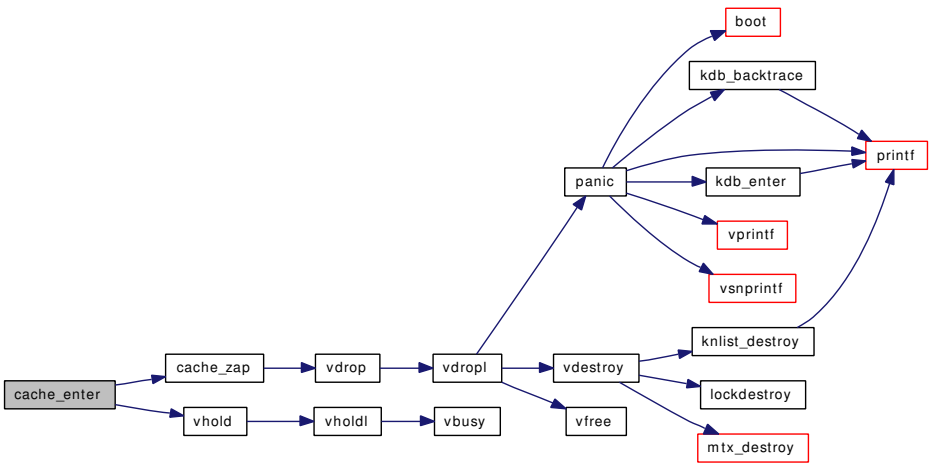
9.153.2.3 void cache_enter (struct vnode * *dvp*, struct vnode * *vp*, struct componentname * *cnp*)

Definition at line 473 of file vfs_cache.c.

References cache_alloc, CACHE_LOCK, CACHE_UNLOCK, cache_zap(), NCF_WHITE, NCHHASH, and vhold().

Referenced by mqfs_lookupx().

Here is the call graph for this function:



9.153.2.4 int cache_leaf_test (struct vnode * *vp*)

Definition at line 306 of file vfs_cache.c.

References CACHE_LOCK, and CACHE_UNLOCK.

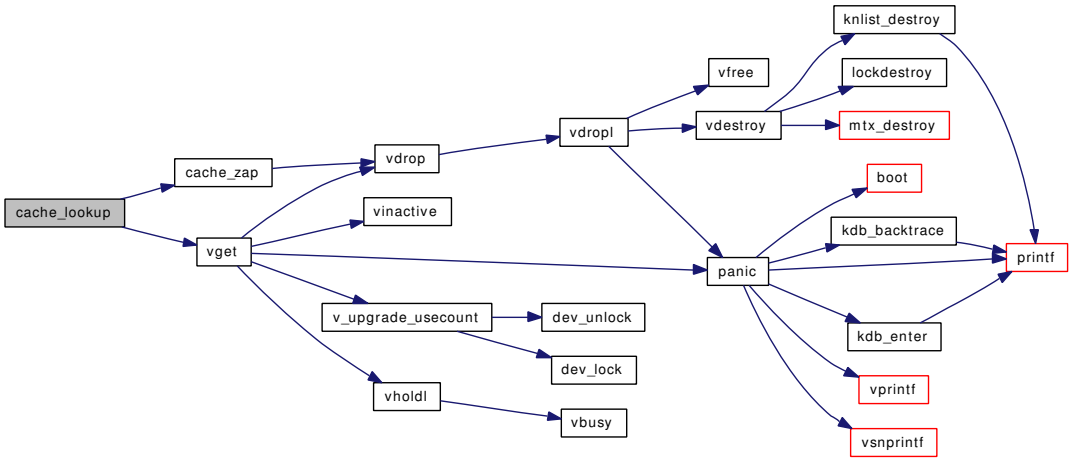
9.153.2.5 int cache_lookup (struct vnode * *dvp*, struct vnode ** *vpp*, struct componentname * *cnp*)

Definition at line 342 of file vfs_cache.c.

References CACHE_LOCK, CACHE_UNLOCK, cache_zap(), NCF_WHITE, NCHHASH, and vget().

Referenced by vfs_cache_lookup().

Here is the call graph for this function:



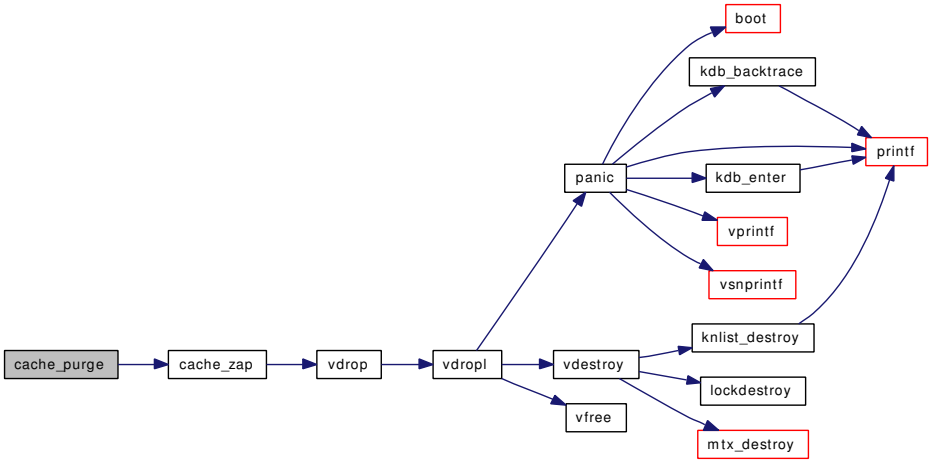
9.153.2.6 void cache_purge (struct vnode * *vp*)

Definition at line 588 of file vfs_cache.c.

References CACHE_LOCK, CACHE_UNLOCK, and cache_zap().

Referenced by devfs_fixup(), do_unlink(), vfs_domount(), and vgonel().

Here is the call graph for this function:



9.153.2.7 void cache_purgevfs (struct mount * *mp*)

Definition at line 609 of file vfs_cache.c.

References CACHE_LOCK.

Referenced by devfs_fixup(), and dounmount().

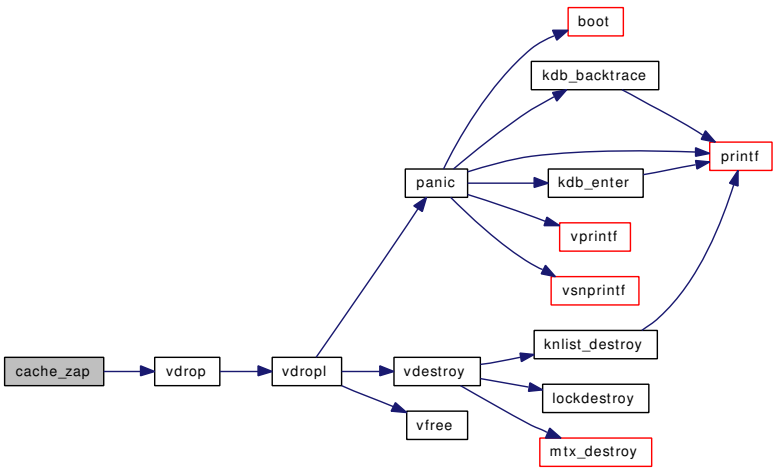
9.153.2.8 static void cache_zap (struct namecache * *ncp*) [static]

Definition at line 268 of file vfs_cache.c.

References cache_free, and vdrop().

Referenced by cache_enter(), cache_lookup(), and cache_purge().

Here is the call graph for this function:



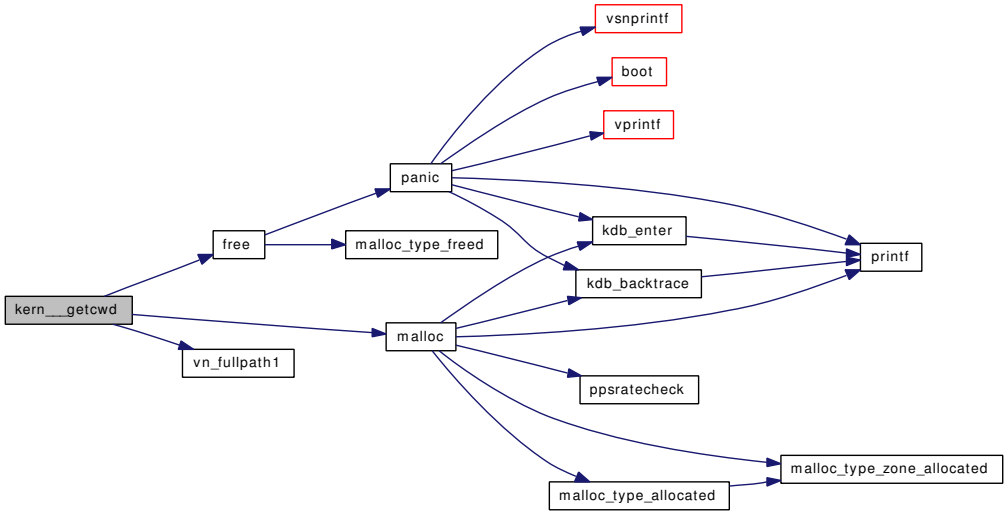
9.153.2.9 int kern___getcwd (struct thread * *td*, u_char * *buf*, enum uio_seg *bufseg*, u_int *buflen*)

Definition at line 704 of file vfs_cache.c.

References free(), Giant, malloc(), and vn_fullpath1().

Referenced by __getcwd().

Here is the call graph for this function:



9.153.2.10 static LIST_HEAD (nchashhead, SYSCTL_INT(*namecache*) [static]

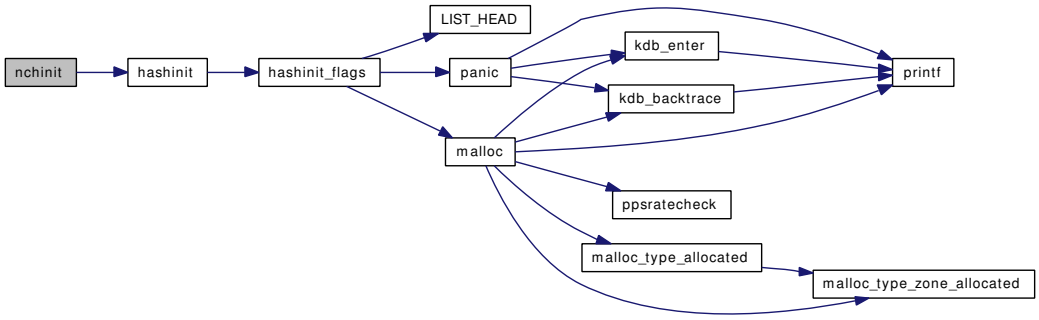
Definition at line 94 of file vfs_cache.c.

9.153.2.11 static void nchinit (void *dummy *__unused*) [static]

Definition at line 569 of file vfs_cache.c.

References CACHE_ZONE_LARGE, CACHE_ZONE_SMALL, desiredvnodes, and hashinit().

Here is the call graph for this function:



9.153.2.12 STATNODE (numfullpathfound)

9.153.2.13 STATNODE (numfullpathfail4)

9.153.2.14 STATNODE (numfullpathfail2)

9.153.2.15 STATNODE (numfullpathfail1)

9.153.2.16 STATNODE (numfullpathcalls)

9.153.2.17 static int sysctl_debug_hashstat_nchash (SYSCTL_HANDLER_ARGS) [static]

Definition at line 216 of file vfs_cache.c.

9.153.2.18 SYSCTL_INT (_debug, OID_AUTO, disablefullpath, CTLFLAG_RW, & *disablefullpath*, 0, "Disable the vn_fullpath function")

9.153.2.19 SYSCTL_INT (_debug, OID_AUTO, disablecwd, CTLFLAG_RW, & *disablecwd*, 0, "Disable the getcwd syscall")

9.153.2.20 SYSCTL_PROC (_debug_hashstat, OID_AUTO, nchash, CTLTYPE_INT| *CTLFLAG_RD*, 0, 0, sysctl_debug_hashstat_nchash, "I", "nchash chain lengths")

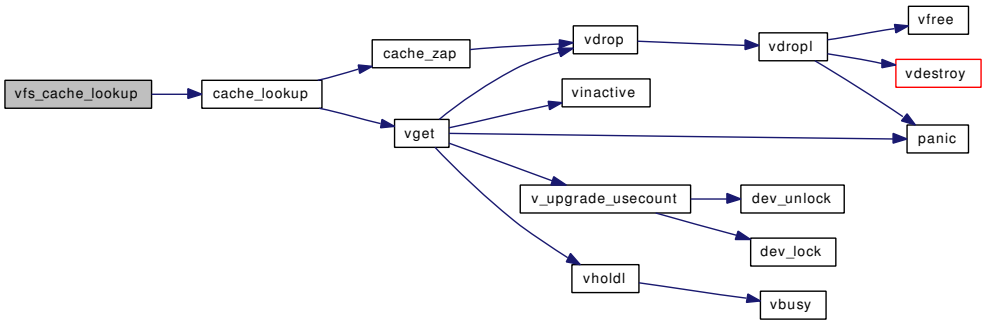
9.153.2.21 SYSCTL_PROC (_debug_hashstat, OID_AUTO, rawnchash, CTLTYPE_INT| *CTLFLAG_RD*, 0, 0, sysctl_debug_hashstat_rawnchash, " *S*, int", "nchash chain lengths")

9.153.2.22 int vfs_cache_lookup (struct vop_lookup_args * *ap*)

Definition at line 641 of file vfs_cache.c.

References cache_lookup(), and td.

Here is the call graph for this function:



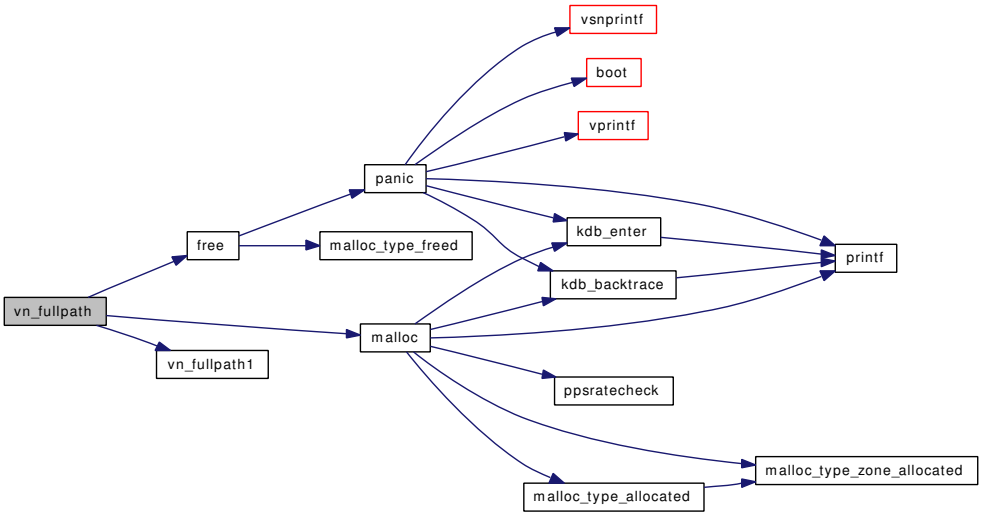
9.153.2.23 int vn_fullpath (struct thread * *td*, struct vnode * *vn*, char ** *retbuf*, char ** *freebuf*)

Definition at line 761 of file vfs_cache.c.

References buf, free(), malloc(), and vn_fullpath1().

Referenced by sysctl_kern_proc_pathname().

Here is the call graph for this function:



9.153.2.24 static int vn_fullpath1 (struct thread * *td*, struct vnode * *vp*, struct vnode * *rdir*, char * *buf*, char ** *retbuf*, u_int *buflen*) [static]

Definition at line 789 of file vfs_cache.c.

References CACHE_LOCK, CACHE_UNLOCK, and rootvnode.

Referenced by kern___getcwd(), and vn_fullpath().

9.153.3 Variable Documentation

9.153.3.1 int disablecwd [static]

Definition at line 689 of file vfs_cache.c.

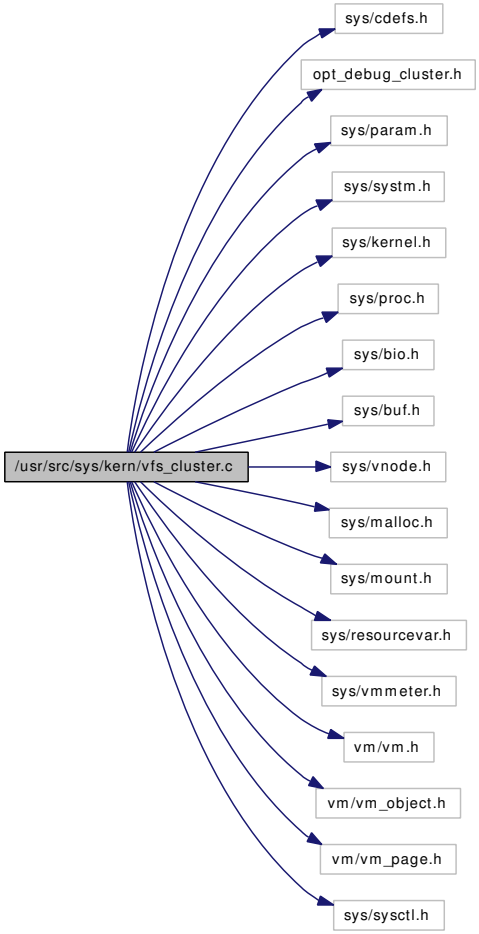
9.153.3.2 int disablefullpath [static]

Definition at line 745 of file vfs_cache.c.

9.154 /usr/src/sys/kern/vfs_cluster.c File Reference

```
#include <sys/cdefs.h>
#include "opt_debug_cluster.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/proc.h>
#include <sys/bio.h>
#include <sys/buf.h>
#include <sys/vnode.h>
#include <sys/malloc.h>
#include <sys/mount.h>
#include <sys/resourcevar.h>
#include <sys/vmmeter.h>
#include <vm/vm.h>
#include <vm/vm_object.h>
#include <vm/vm_page.h>
#include <sys/sysctl.h>
```

Include dependency graph for vfs_cluster.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/vfs_cluster.c,v 1.174 2006/10/22 04:28:13 alc Exp $")
- static MALLOC_DEFINE (M_SEGMENT,"cluster_save_buffer","cluster_save buffer")
- static struct cluster_save * cluster_collectbufs (struct vnode *vp, struct buf *last_bp)
- static struct buf * cluster_rbuild (struct vnode *vp, u_quad_t filesize, daddr_t lbn, daddr_t blkno, long size, int run, struct buf *fbp)
- static void cluster_callback (struct buf *)
- SYSCTL_INT (_vfs, OID_AUTO, write_behind, CTLFLAG_RW,&write_behind, 0,"Cluster write-behind; 0: disable, 1: enable, 2: backed off")
- SYSCTL_INT (_vfs, OID_AUTO, read_max, CTLFLAG_RW,&read_max, 0,"Cluster read-ahead max block count")
- int cluster_read (struct vnode *vp, u_quad_t filesize, daddr_t lblkno, long size, struct ucred *cred, long totread, int seqcount, struct buf **bpp)
- static __inline int cluster_wbuild_wb (struct vnode *vp, long size, daddr_t start_lbn, int len)
- void cluster_write (struct vnode *vp, struct buf *bp, u_quad_t filesize, int seqcount)
- int cluster_wbuild (struct vnode *vp, long size, daddr_t start_lbn, int len)

Variables

- static int write_behind = 1

- static int read_max = 8
- vm_page_t bogus_page

9.154.1 Function Documentation

9.154.1.1 __FBSDID ("$FreeBSD: src/sys/kern/vfs_cluster. *c*, v 1.174 2006/10/22 04:28:13 alc Exp $")

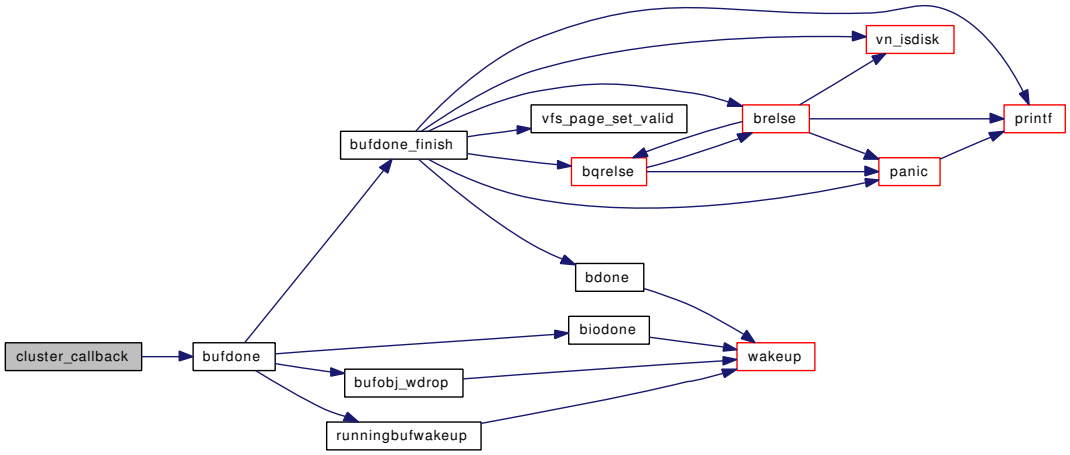
9.154.1.2 static void cluster_callback (struct buf *) [static]

Definition at line 505 of file vfs_cluster.c.

References buf, and bufdone().

Referenced by cluster_rbuild(), and cluster_wbuild().

Here is the call graph for this function:



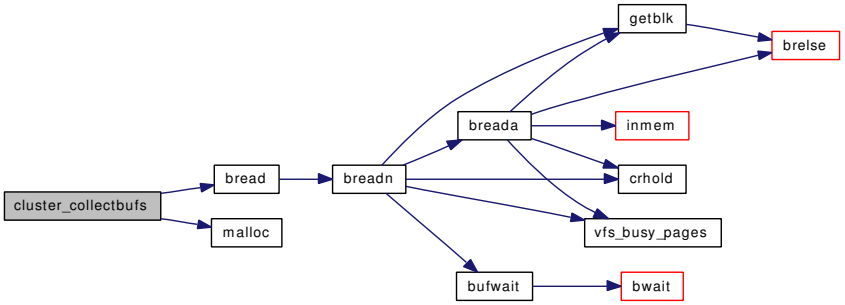
9.154.1.3 static struct cluster_save * cluster_collectbufs (struct vnode * *vp*, struct buf * *last_bp*) [static]

Definition at line 970 of file vfs_cluster.c.

References bread(), buf, and malloc().

Referenced by cluster_write().

Here is the call graph for this function:



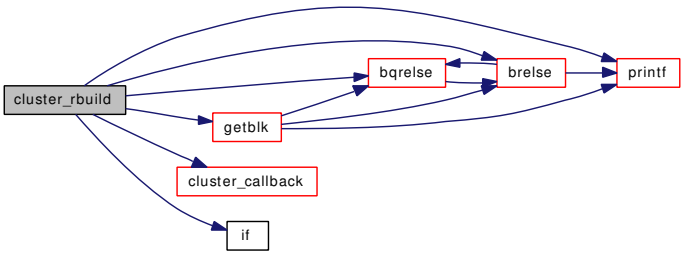
9.154.1.4 static struct buf * cluster_rbuild (struct vnode * *vp*, u_quad_t *filesize*, daddr_t *lbn*, daddr_t *blkno*, long *size*, int *run*, struct buf * *fbp*) [static]

Definition at line 299 of file vfs_cluster.c.

References bogus_page, bqrelse(), brelse(), buf, cluster_callback(), getblk(), if(), and printf().

Referenced by cluster_read().

Here is the call graph for this function:

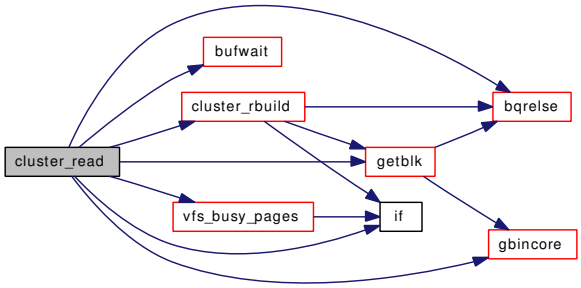


9.154.1.5 int cluster_read (struct vnode * *vp*, u_quad_t *filesize*, daddr_t *lblkno*, long *size*, struct ucred * *cred*, long *totread*, int *seqcount*, struct buf ** *bpp*)

Definition at line 86 of file vfs_cluster.c.

References bqrelse(), buf, bufwait(), cluster_rbuild(), gbincore(), getblk(), if(), nbuf, read_max, and vfs_busy_pages().

Here is the call graph for this function:



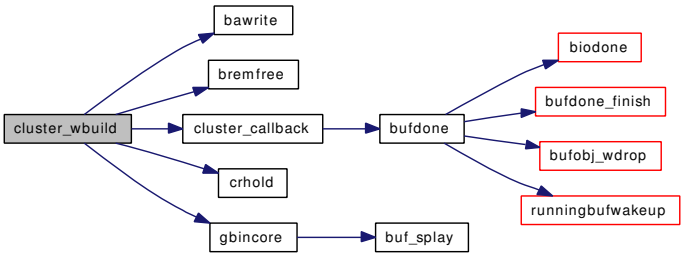
9.154.1.6 int cluster_wbuild (struct vnode * *vp*, long *size*, daddr_t *start_lbn*, int *len*)

Definition at line 736 of file vfs_cluster.c.

References bawrite(), bremfree(), buf, cluster_callback(), crhold(), and gbincore().

Referenced by cluster_wbuild_wb(), and vfs_bio_awrite().

Here is the call graph for this function:



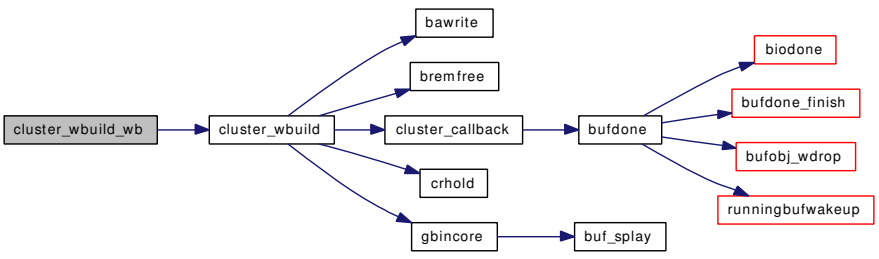
9.154.1.7 static __inline int cluster_wbuild_wb (struct vnode * *vp*, long *size*, daddr_t *start_lbn*, int *len*) [static]

Definition at line 559 of file vfs_cluster.c.

References cluster_wbuild(), and write_behind.

Referenced by cluster_write().

Here is the call graph for this function:

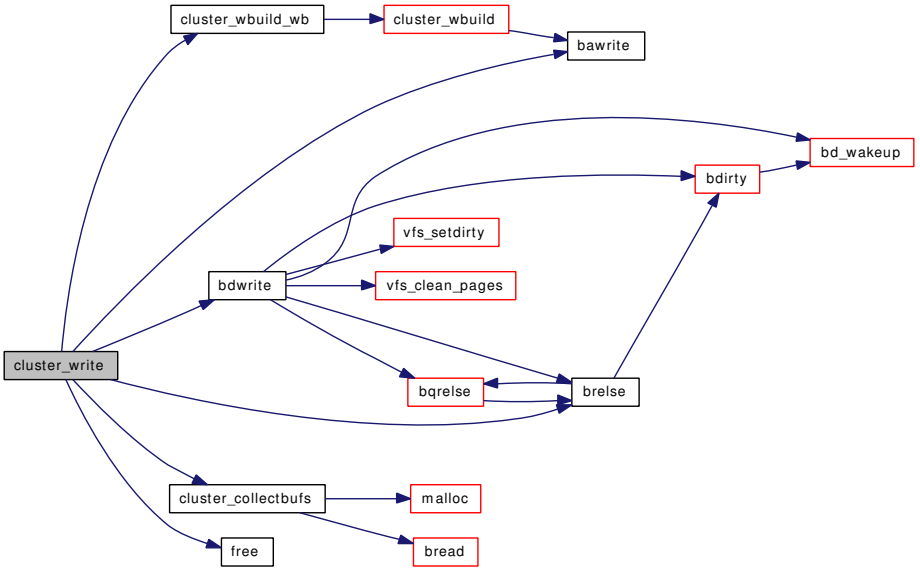


9.154.1.8 void cluster_write (struct vnode * *vp*, struct buf * *bp*, u_quad_t *filesize*, int *seqcount*)

Definition at line 590 of file vfs_cluster.c.

References bawrite(), bdwrite(), brelse(), buf, cluster_collectbufs(), cluster_wbuild_wb(), and free().

Here is the call graph for this function:



9.154.1.9 static MALLOC_DEFINE (M_SEGMENT, "cluster_save_buffer", "cluster_save buffer") [static]

9.154.1.10 SYSCTL_INT (_vfs, OID_AUTO, read_max, CTLFLAG_RW, & *read_max*, 0, "Cluster read-ahead max block count")

9.154.1.11 SYSCTL_INT (_vfs, OID_AUTO, write_behind, CTLFLAG_RW, & *write_behind*, 0, "Cluster write-behind; 0: *disable*, 1:enable, 2:backed off")

9.154.2 Variable Documentation

9.154.2.1 vm_page_t bogus_page

Definition at line 209 of file vfs_bio.c.

Referenced by allocbuf(), brelse(), bufdone_finish(), bufinit(), cluster_rbuild(), vfs_bio_clrbuf(), vfs_busy_pages(), and vfs_unbusy_pages().

9.154.2.2 int read_max = 8 [static]

Definition at line 74 of file vfs_cluster.c.

Referenced by cluster_read().

9.154.2.3 int write_behind = 1 [static]

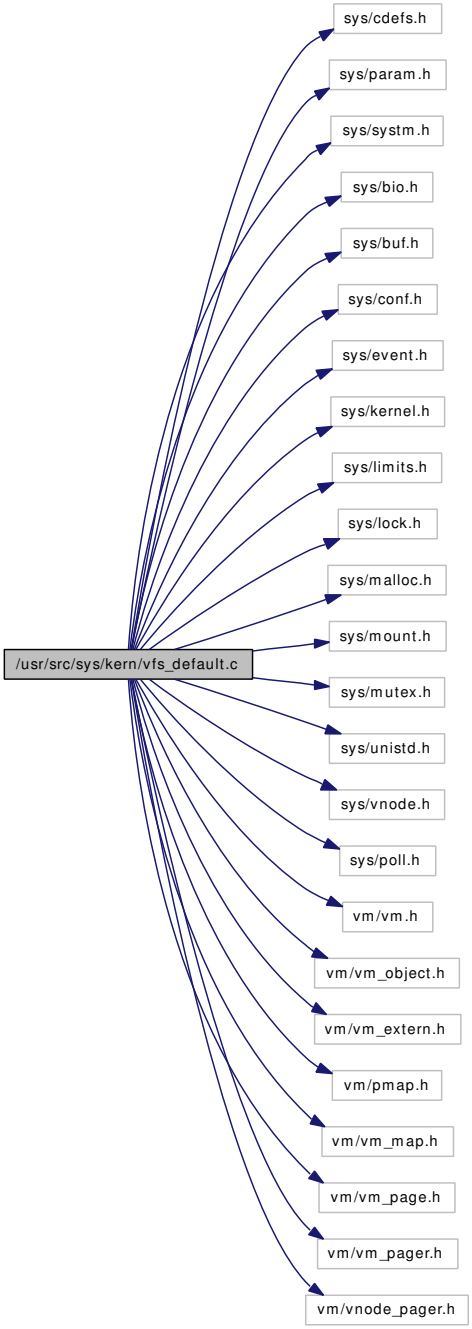
Definition at line 70 of file vfs_cluster.c.

Referenced by cluster_wbuild_wb().

9.155 /usr/src/sys/kern/vfs_default.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/bio.h>
#include <sys/buf.h>
#include <sys/conf.h>
#include <sys/event.h>
#include <sys/kernel.h>
#include <sys/limits.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mount.h>
#include <sys/mutex.h>
#include <sys/unistd.h>
#include <sys/vnode.h>
#include <sys/poll.h>
#include <vm/vm.h>
#include <vm/vm_object.h>
#include <vm/vm_extern.h>
#include <vm/pmap.h>
#include <vm/vm_map.h>
#include <vm/vm_page.h>
#include <vm/vm_pager.h>
#include <vm/vnode_pager.h>
```

Include dependency graph for vfs_default.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/vfs_default.c,v 1.137 2007/02/16 17:32:41 pjd Exp $")
- static int vop_nolookup (struct vop_lookup_args *)
- static int vop_nostrategy (struct vop_strategy_args *)
- int vop_eopnotsupp (struct vop_generic_args *ap)
- int vop_ebadf (struct vop_generic_args *ap)
- int vop_enotty (struct vop_generic_args *ap)
- int vop_einval (struct vop_generic_args *ap)

- int vop_null (struct vop_generic_args *ap)
- int vop_panic (struct vop_generic_args *ap)
- int vop_stdpathconf (struct vop_pathconf_args *ap)
- int vop_stdlock (struct _vop_lock_args *ap)
- int vop_stdunlock (struct vop_unlock_args *ap)
- int vop_stdislocked (struct vop_islocked_args *ap)
- int vop_nopoll (struct vop_poll_args *ap)
- int vop_stdpoll (struct vop_poll_args *ap)
- int vop_stdgetwritemount (struct vop_getwritemount_args *ap)
- int vop_stdbmap (struct vop_bmap_args *ap)
- int vop_stdfsync (struct vop_fsync_args *ap)
- int vop_stdgetpages (struct vop_getpages_args *ap)
- int vop_stdkqfilter (struct vop_kqfilter_args *ap)
- int vop_stdputpages (struct vop_putpages_args *ap)
- int vop_stdvptofh (struct vop_vptofh_args *ap)
- int vfs_stdroot (struct mount *mp, int flags, struct vnode **vpp, struct thread *td)
- int vfs_stdstatfs (struct mount *mp, struct statfs *sbp, struct thread *td)
- int vfs_stdquotactl (struct mount *mp, int cmds, uid_t uid, void *arg, struct thread *td)
- int vfs_stdsync (struct mount *mp, int waitfor, struct thread *td)
- int vfs_stdnosync (struct mount *mp, int waitfor, struct thread *td)
- int vfs_stdvget (struct mount *mp, ino_t ino, int flags, struct vnode **vpp)
- int vfs_stdfhtovp (struct mount *mp, struct fid *fhp, struct vnode **vpp)
- int vfs_stdinit (struct vfsconf *vfsp)
- int vfs_stduninit (struct vfsconf *vfsp)
- int vfs_stdextattrctl (struct mount *mp, int cmd, struct vnode *filename_vp, int attrnamespace, const char *attrname, struct thread *td)
- int vfs_stdsysctl (struct mount *mp, fsctlop_t op, struct sysctl_req *req)

Variables

- vop_vector default_vnodeops

9.155.1 Function Documentation

9.155.1.1 __FBSDID ("$FreeBSD: src/sys/kern/vfs_default. *c*, v 1.137 2007/02/16 17:32:41 pjd Exp $")

9.155.1.2 int vfs_stdextattrctl (struct mount * *mp*, int *cmd*, struct vnode * *filename_vp*, int *attrnamespace*, const char * *attrname*, struct thread * *td*)

Definition at line 653 of file vfs_default.c.

Referenced by vfs_register().

9.155.1.3 int vfs_stdfhtovp (struct mount * *mp*, struct fid * *fhp*, struct vnode ** *vpp*)

Definition at line 627 of file vfs_default.c.

Referenced by vfs_register().

9.155.1.4 int vfs_stdinit (struct vfsconf * *vfsp*)

Definition at line 637 of file vfs_default.c.

Referenced by vfs_register().

9.155.1.5 int vfs_stdnosync (struct mount * *mp*, int *waitfor*, struct thread * *td*)

Definition at line 606 of file vfs_default.c.

Referenced by vfs_register().

9.155.1.6 int vfs_stdquotactl (struct mount * *mp*, int *cmds*, uid_t *uid*, void * *arg*, struct thread * *td*)

Definition at line 547 of file vfs_default.c.

Referenced by vfs_register().

9.155.1.7 int vfs_stdroot (struct mount * *mp*, int *flags*, struct vnode ** *vpp*, struct thread * *td*)

Definition at line 526 of file vfs_default.c.

Referenced by vfs_register().

9.155.1.8 int vfs_stdstatfs (struct mount * *mp*, struct statfs * *sbp*, struct thread * *td*)

Definition at line 537 of file vfs_default.c.

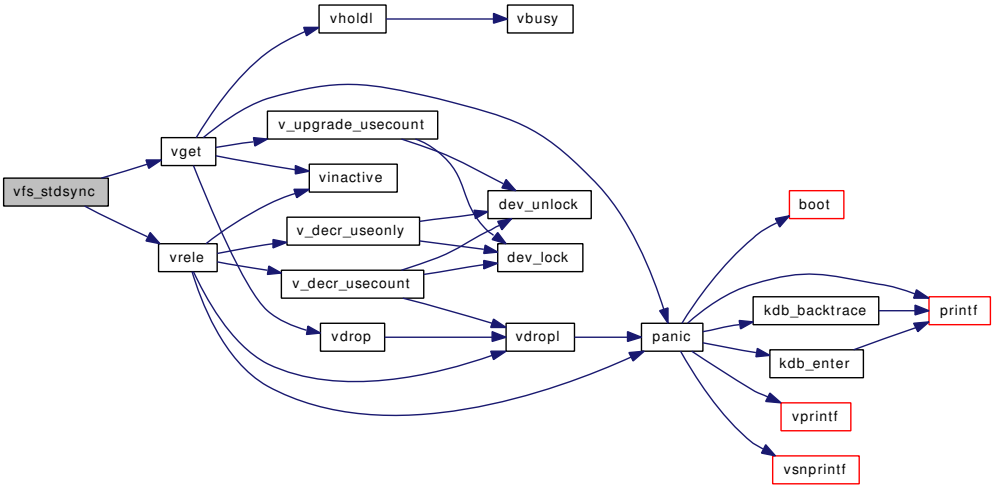
Referenced by vfs_register().

9.155.1.9 int vfs_stdsync (struct mount * *mp*, int *waitfor*, struct thread * *td*)

Definition at line 559 of file vfs_default.c.

References vget(), and vrele().

Here is the call graph for this function:



9.155.1.10 int vfs_stdsysctl (struct mount * *mp*, fsctlop_t *op*, struct sysctl_req * *req*)

Definition at line 668 of file vfs_default.c.

Referenced by vfs_register().

9.155.1.11 int vfs_stduninit (struct vfsconf * *vfsp*)

Definition at line 645 of file vfs_default.c.

Referenced by vfs_register().

9.155.1.12 int vfs_stdvget (struct mount * *mp*, ino_t *ino*, int *flags*, struct vnode ** *vpp*)

Definition at line 616 of file vfs_default.c.

Referenced by vfs_register().

9.155.1.13 int vop_ebadf (struct vop_generic_args * *ap*)

Definition at line 118 of file vfs_default.c.

9.155.1.14 int vop_einval (struct vop_generic_args * *ap*)

Definition at line 132 of file vfs_default.c.

9.155.1.15 int vop_enotty (struct vop_generic_args * *ap*)

Definition at line 125 of file vfs_default.c.

9.155.1.16 int vop_eopnotsupp (struct vop_generic_args * *ap*)

Definition at line 108 of file vfs_default.c.

9.155.1.17 static int vop_nolookup (struct vop_lookup_args *) [static]

Definition at line 168 of file vfs_default.c.

9.155.1.18 int vop_nopoll (struct vop_poll_args * *ap*)

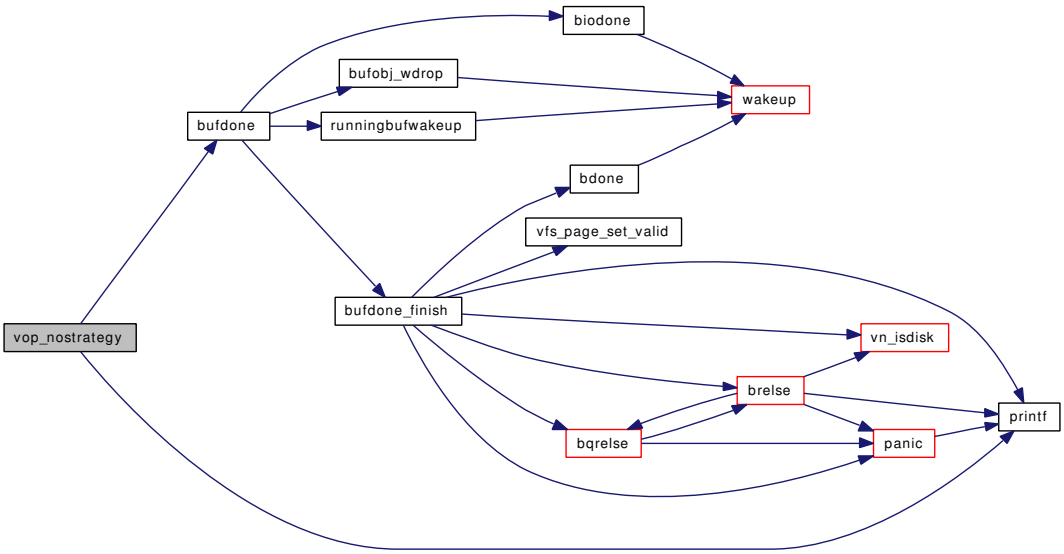
Definition at line 300 of file vfs_default.c.

9.155.1.19 static int vop_nostrategy (struct vop_strategy_args *) [static]

Definition at line 193 of file vfs_default.c.

References bufdone(), and printf().

Here is the call graph for this function:



9.155.1.20 int vop_null (struct vop_generic_args * *ap*)

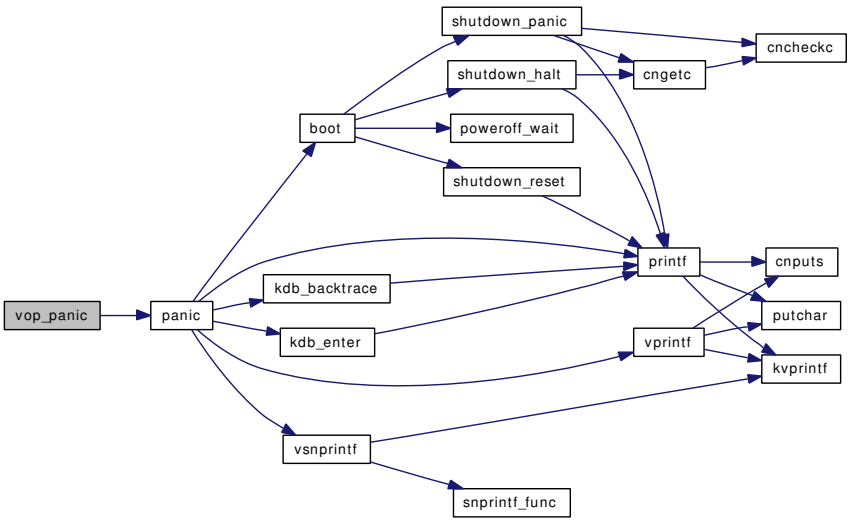
Definition at line 139 of file vfs_default.c.

9.155.1.21 int vop_panic (struct vop_generic_args * *ap*)

Definition at line 149 of file vfs_default.c.

References panic().

Here is the call graph for this function:



9.155.1.22 int vop_stdbmap (struct vop_bmap_args * *ap*)

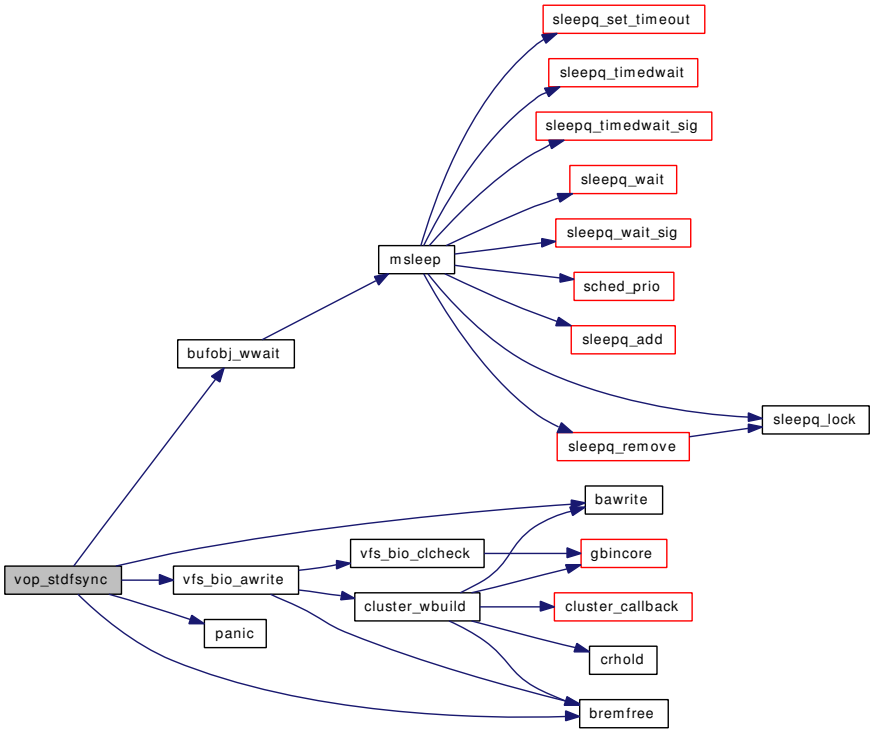
Definition at line 372 of file vfs_default.c.

9.155.1.23 int vop_stdfsync (struct vop_fsync_args * *ap*)

Definition at line 395 of file vfs_default.c.

References bawrite(), bremfree(), buf, bufobj_wwait(), panic(), and vfs_bio_awrite().

Here is the call graph for this function:



9.155.1.24 int vop_stdgetpages (struct vop_getpages_args * *ap*)

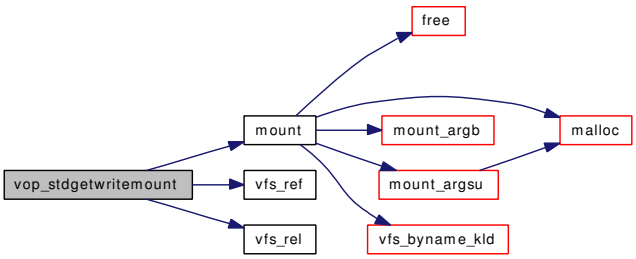
Definition at line 478 of file vfs_default.c.

9.155.1.25 int vop_stdgetwritemount (struct vop_getwritemount_args * *ap*)

Definition at line 343 of file vfs_default.c.

References mount(), vfs_ref(), and vfs_rel().

Here is the call graph for this function:

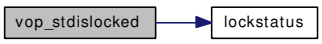


9.155.1.26 int vop_stdislocked (struct vop_islocked_args * *ap*)

Definition at line 286 of file vfs_default.c.

References lockstatus().

Here is the call graph for this function:

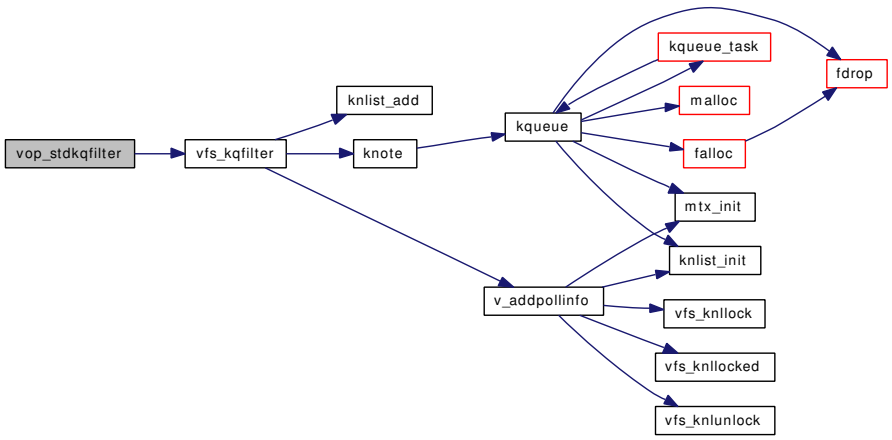


9.155.1.27 int vop_stdkqfilter (struct vop_kqfilter_args * *ap*)

Definition at line 493 of file vfs_default.c.

References vfs_kqfilter().

Here is the call graph for this function:

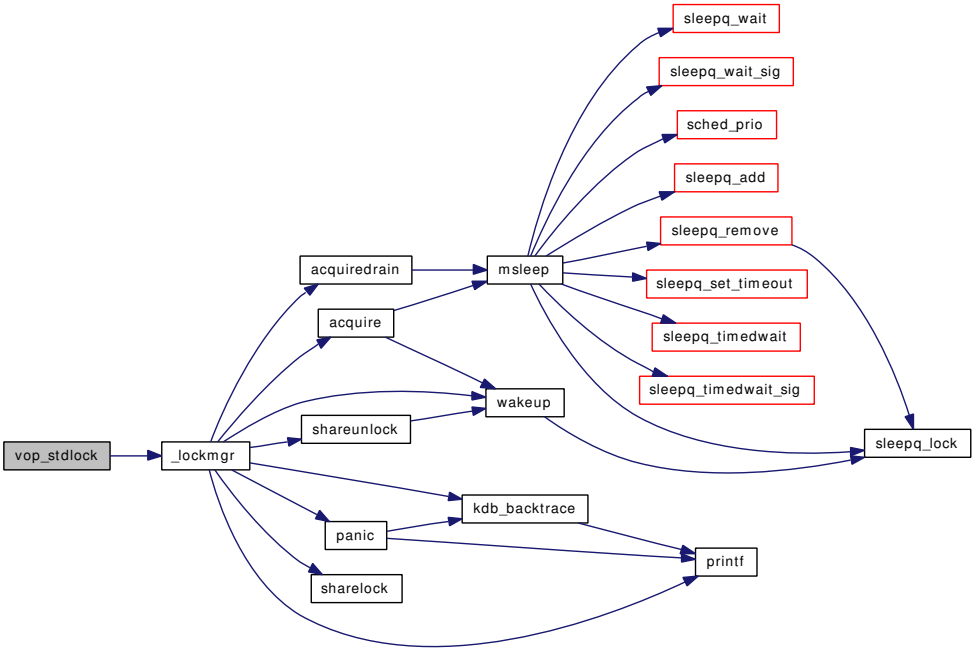


9.155.1.28 int vop_stdlock (struct _vop_lock_args * *ap*)

Definition at line 255 of file vfs_default.c.

References _lockmgr().

Here is the call graph for this function:



9.155.1.29 int vop_stdpathconf (struct vop_pathconf_args * *ap*)

Definition at line 212 of file vfs_default.c.

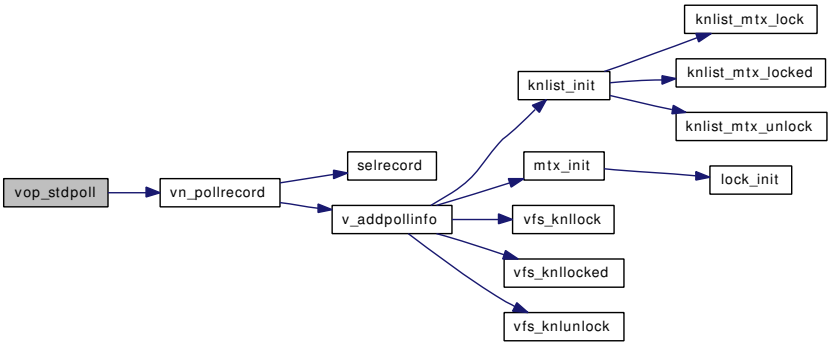
References MAX_INPUT.

9.155.1.30 int vop_stdpoll (struct vop_poll_args * *ap*)

Definition at line 326 of file vfs_default.c.

References vn_pollrecord().

Here is the call graph for this function:



9.155.1.31 int vop_stdputpages (struct vop_putpages_args * *ap*)

Definition at line 500 of file vfs_default.c.

9.155.1.32 int vop_stdunlock (struct vop_unlock_args * *ap*)

Definition at line 271 of file vfs_default.c.

9.155.1.33 int vop_stdvptofh (struct vop_vptofh_args * *ap*)

Definition at line 516 of file vfs_default.c.

9.155.2 Variable Documentation

9.155.2.1 struct vop_vector default_vnodeops

Initial value:

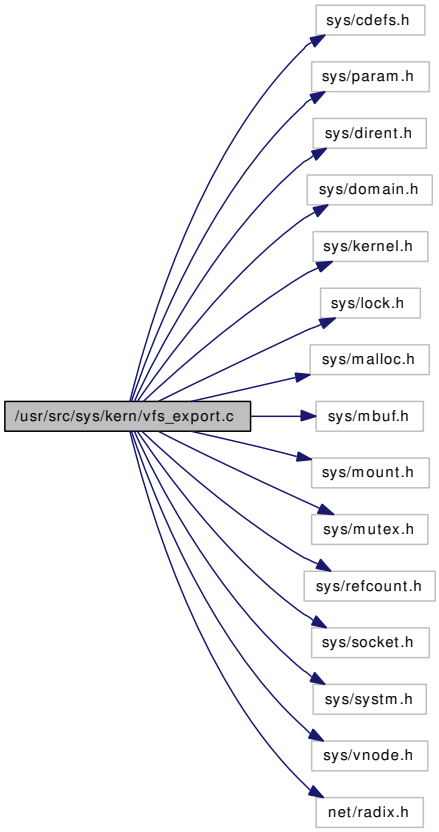
```
{
        .vop_default =          NULL,
        .vop_bypass =           VOP_EOPNOTSUPP,

        .vop_advlock =          VOP_EINVAL,
        .vop_bmap =             vop_stdbmap,
        .vop_close =            VOP_NULL,
        .vop_fsync =            VOP_NULL,
        .vop_getpages =         vop_stdgetpages,
        .vop_getwritemount =    vop_stdgetwritemount,
        .vop_inactive =         VOP_NULL,
        .vop_ioctl =            VOP_ENOTTY,
        .vop_kqfilter =         vop_stdkqfilter,
        .vop_islocked =         vop_stdislocked,
        .vop_lease =            VOP_NULL,
        ._vop_lock =            vop_stdlock,
        .vop_lookup =           vop_nolookup,
        .vop_open =             VOP_NULL,
        .vop_pathconf =         VOP_EINVAL,
        .vop_poll =             vop_nopoll,
        .vop_putpages =         vop_stdputpages,
        .vop_readlink =         VOP_EINVAL,
        .vop_revoke =           VOP_PANIC,
        .vop_strategy =         vop_nostrategy,
        .vop_unlock =           vop_stdunlock,
        .vop_vptofh =           vop_stdvptofh,
}
```

Definition at line 74 of file vfs_default.c.

9.156 /usr/src/sys/kern/vfs_export.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/dirent.h>
#include <sys/domain.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/malloc.h>
#include <sys/mbuf.h>
#include <sys/mount.h>
#include <sys/mutex.h>
#include <sys/refcount.h>
#include <sys/socket.h>
#include <sys/systm.h>
#include <sys/vnode.h>
#include <net/radix.h>
```

Include dependency graph for vfs_export.c:



Data Structures

- struct netcred
- struct netexport

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/vfs_export.c,v 1.341 2007/02/15 22:08:35 pjd Exp $")
- static MALLOC_DEFINE (M_NETADDR,"export_host","Export host address structure")
- static void vfs_free_addrlist (struct netexport *nep)
- static int vfs_free_netcred (struct radix_node *rn, void *w)
- static int vfs_hang_addrlist (struct mount *mp, struct netexport *nep, struct export_args *argp)
- static struct netcred * vfs_export_lookup (struct mount *, struct sockaddr *)
- int vfs_export (struct mount *mp, struct export_args *argp)
- int vfs_setpublicfs (struct mount *mp, struct netexport *nep, struct export_args *argp)
- int vfs_stdcheckexp (struct mount *mp, struct sockaddr *nam, int *extflagsp, struct ucred **credanonp)

9.156.1 Function Documentation

9.156.1.1 __FBSDID ("$FreeBSD: src/sys/kern/vfs_export. *c*, v 1.341 2007/02/15 22:08:35 pjd Exp $")

9.156.1.2 static MALLOC_DEFINE (M_NETADDR, "export_host", "Export host address structure") `[static]`

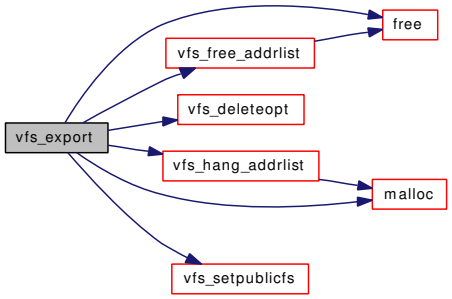
9.156.1.3 int vfs_export (struct mount * *mp*, struct export_args * *argp*)

Definition at line 238 of file vfs_export.c.

References free(), malloc(), vfs_deleteopt(), vfs_free_addrlist(), vfs_hang_addrlist(), and vfs_setpublicfs().

Referenced by vfs_domount().

Here is the call graph for this function:



9.156.1.4 static struct netcred * vfs_export_lookup (struct mount *, struct sockaddr *) [static]

Definition at line 382 of file vfs_export.c.

References netexport::ne_defexported, netexport::ne_rtable, and netcred::netc_rnodes.

Referenced by vfs_stdcheckexp().

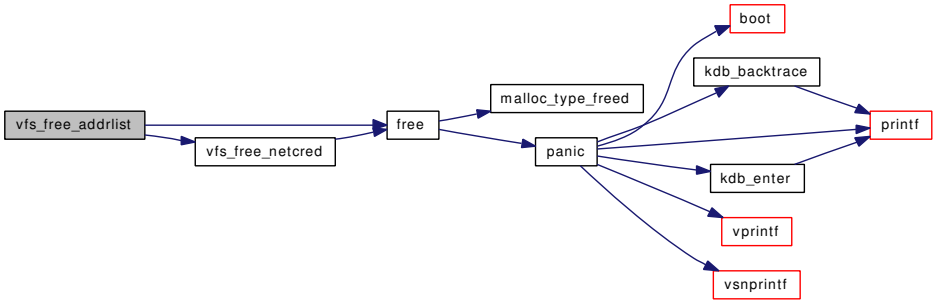
9.156.1.5 static void vfs_free_addrlist (struct netexport * *nep*) [static]

Definition at line 216 of file vfs_export.c.

References free(), netexport::ne_rtable, and vfs_free_netcred().

Referenced by vfs_export().

Here is the call graph for this function:



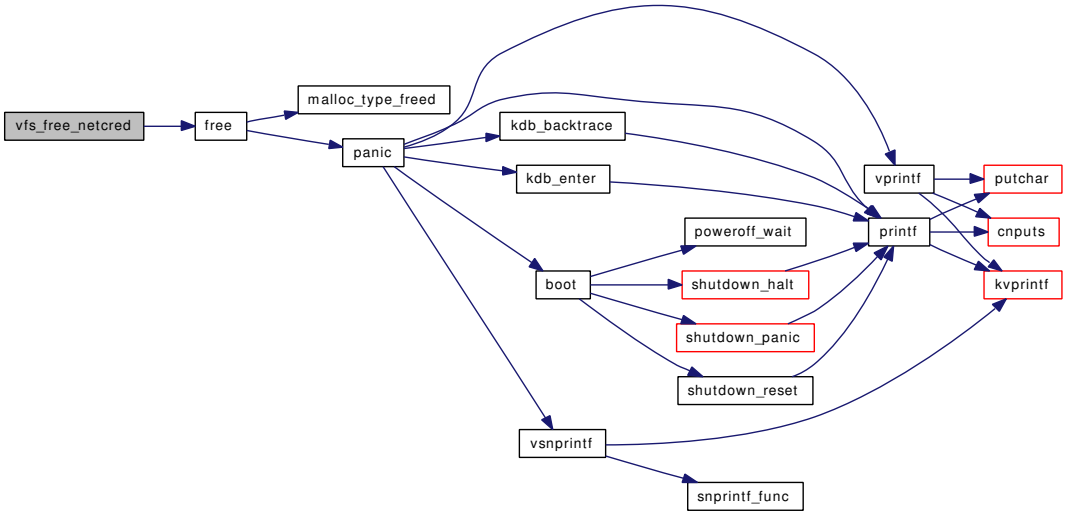
9.156.1.6 static int vfs_free_netcred (struct radix_node * *rn*, void * *w*) [static]

Definition at line 203 of file vfs_export.c.

References free().

Referenced by vfs_free_addrlist().

Here is the call graph for this function:



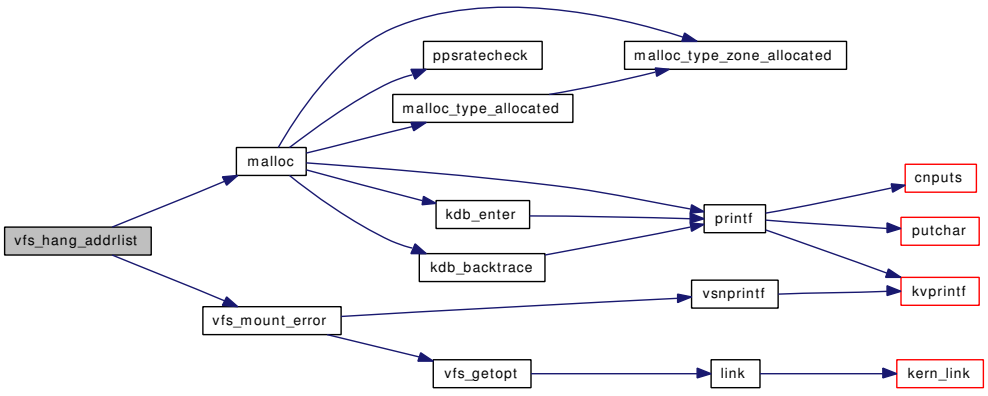
9.156.1.7 static int vfs_hang_addrlist (struct mount * *mp*, struct netexport * *nep*, struct export_args * *argp*) [static]

Definition at line 86 of file vfs_export.c.

References malloc(), netexport::ne_defexported, netexport::ne_rtable, netcred::netc_anon, netcred::netc_-exflags, and vfs_mount_error().

Referenced by vfs_export().

Here is the call graph for this function:



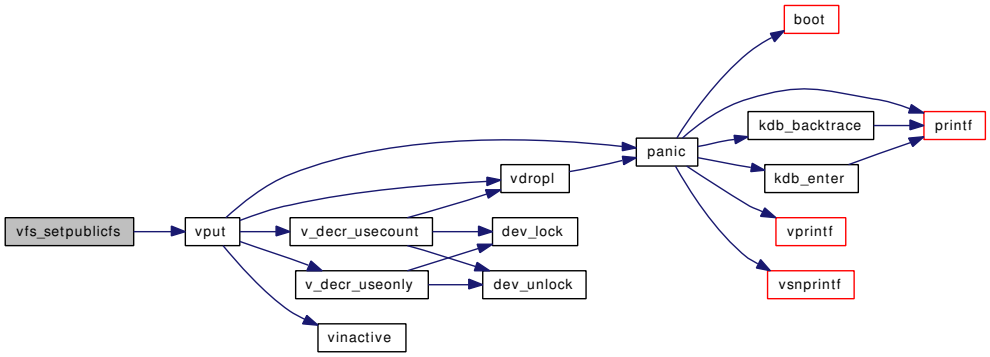
9.156.1.8 int vfs_setpublicfs (struct mount * *mp*, struct netexport * *nep*, struct export_args * *argp*)

Definition at line 302 of file vfs_export.c.

References vput().

Referenced by dounmount(), and vfs_export().

Here is the call graph for this function:



9.156.1.9 int vfs_stdcheckexp (struct mount * *mp*, struct sockaddr * *nam*, int * *extflagsp*, struct ucred ** *credanonp*)

Definition at line 428 of file vfs_export.c.

References netcred::netc_anon, netcred::netc_exflags, and vfs_export_lookup().

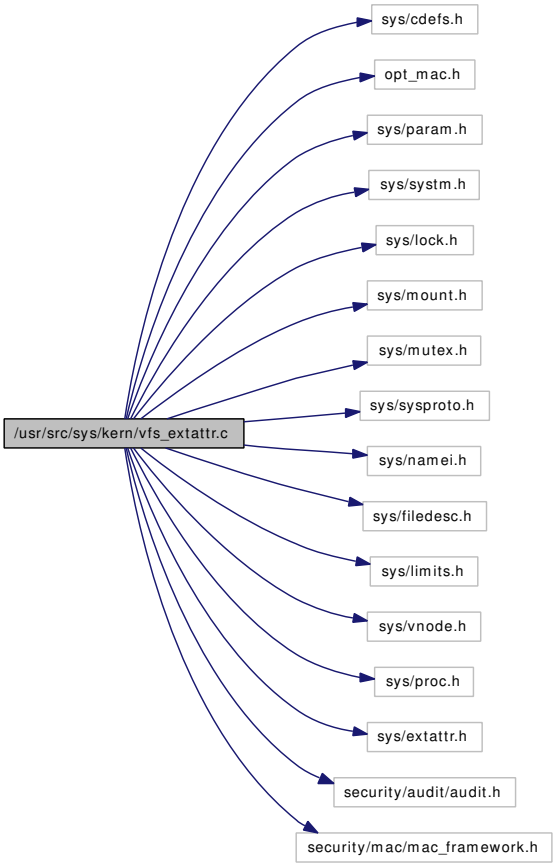
Referenced by vfs_register().

Here is the call graph for this function:

9.157 /usr/src/sys/kern/vfs_extattr.c File Reference

```
#include <sys/cdefs.h>
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/lock.h>
#include <sys/mount.h>
#include <sys/mutex.h>
#include <sys/sysproto.h>
#include <sys/namei.h>
#include <sys/filedesc.h>
#include <sys/limits.h>
#include <sys/vnode.h>
#include <sys/proc.h>
#include <sys/extattr.h>
#include <security/audit/audit.h>
#include <security/mac/mac_framework.h>
```

Include dependency graph for vfs_extattr.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/vfs_extattr.c,v 1.431 2006/12/23 00:30:03 rwatson Exp $")
- int extattrctl (struct thread *td, struct extattrctl_args *uap)
- static int extattr_set_vp (struct vnode *vp, int attrnamespace, const char *attrname, void *data, size_t nbytes, struct thread *td)
- int extattr_set_fd (struct thread *td, struct extattr_set_fd_args *uap)
- int extattr_set_file (struct thread *td, struct extattr_set_file_args *uap)
- int extattr_set_link (struct thread *td, struct extattr_set_link_args *uap)
- static int extattr_get_vp (struct vnode *vp, int attrnamespace, const char *attrname, void *data, size_t nbytes, struct thread *td)
- int extattr_get_fd (struct thread *td, struct extattr_get_fd_args *uap)
- int extattr_get_file (struct thread *td, struct extattr_get_file_args *uap)
- int extattr_get_link (struct thread *td, struct extattr_get_link_args *uap)
- static int extattr_delete_vp (struct vnode *vp, int attrnamespace, const char *attrname, struct thread *td)
- int extattr_delete_fd (struct thread *td, struct extattr_delete_fd_args *uap)
- int extattr_delete_file (struct thread *td, struct extattr_delete_file_args *uap)
- int extattr_delete_link (struct thread *td, struct extattr_delete_link_args *uap)
- static int extattr_list_vp (struct vnode *vp, int attrnamespace, void *data, size_t nbytes, struct thread *td)
- int extattr_list_fd (struct thread *td, struct extattr_list_fd_args *uap)
- int extattr_list_file (struct thread *td, struct extattr_list_file_args *uap)

- int extattr_list_link (struct thread *td, struct extattr_list_link_args *uap)

9.157.1 Function Documentation

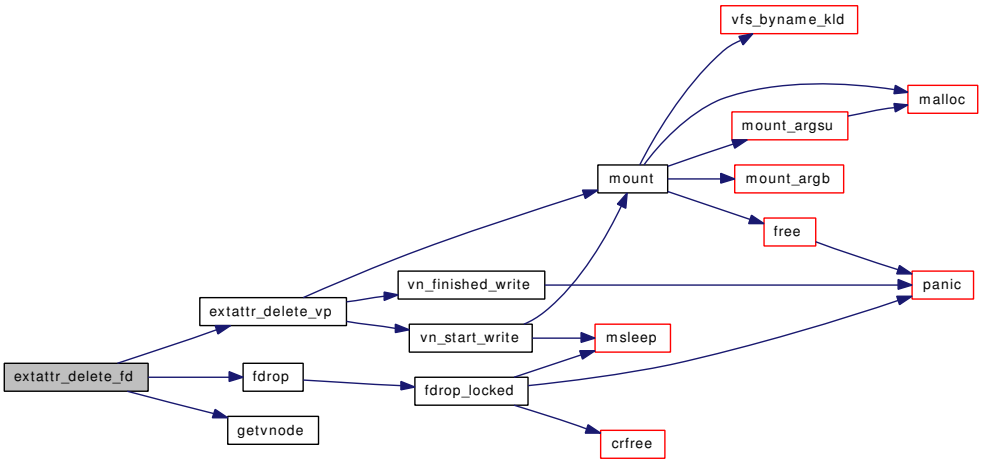
9.157.1.1 __FBSDID ("$FreeBSD: src/sys/kern/vfs_extattr. *c*, v 1.431 2006/12/23 00:30:03 rwatson Exp $")

9.157.1.2 int extattr_delete_fd (struct thread * *td*, struct extattr_delete_fd_args * *uap*)

Definition at line 535 of file vfs_extattr.c.

References extattr_delete_vp(), fdrop(), and getvnode().

Here is the call graph for this function:

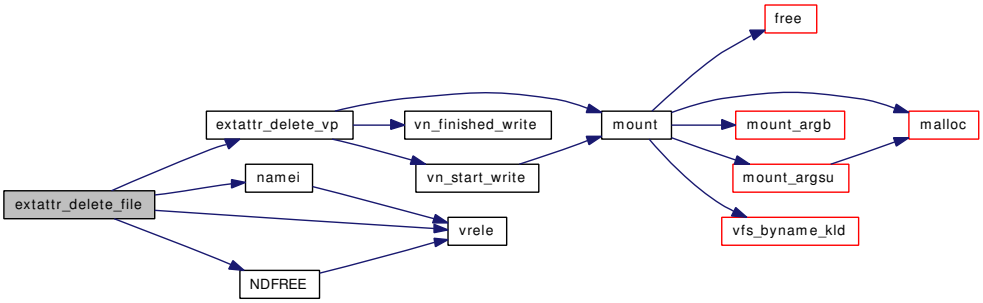


9.157.1.3 int extattr_delete_file (struct thread * *td*, struct extattr_delete_file_args * *uap*)

Definition at line 567 of file vfs_extattr.c.

References extattr_delete_vp(), namei(), NDFREE(), and vrele().

Here is the call graph for this function:

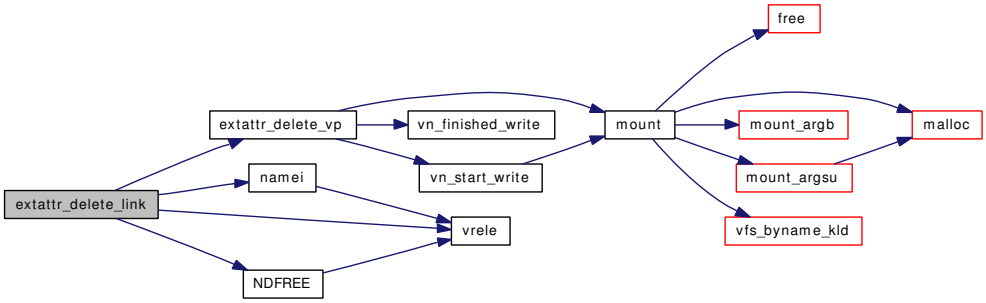


9.157.1.4 int extattr_delete_link (struct thread * *td*, struct extattr_delete_link_args * *uap*)

Definition at line 600 of file vfs_extattr.c.

References extattr_delete_vp(), namei(), NDFREE(), and vrele().

Here is the call graph for this function:



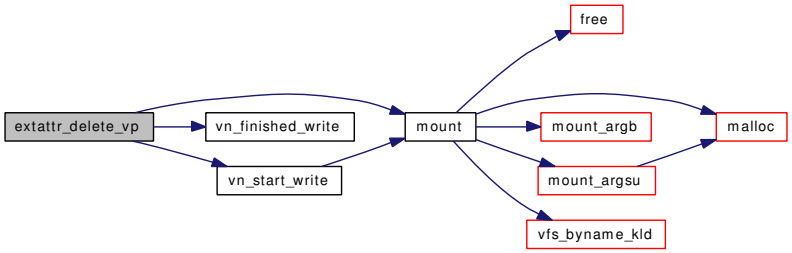
9.157.1.5 static int extattr_delete_vp (struct vnode * *vp*, int *attrnamespace*, const char * *attrname*, struct thread * *td*) [static]

Definition at line 501 of file vfs_extattr.c.

References mount(), vn_finished_write(), and vn_start_write().

Referenced by extattr_delete_fd(), extattr_delete_file(), and extattr_delete_link().

Here is the call graph for this function:

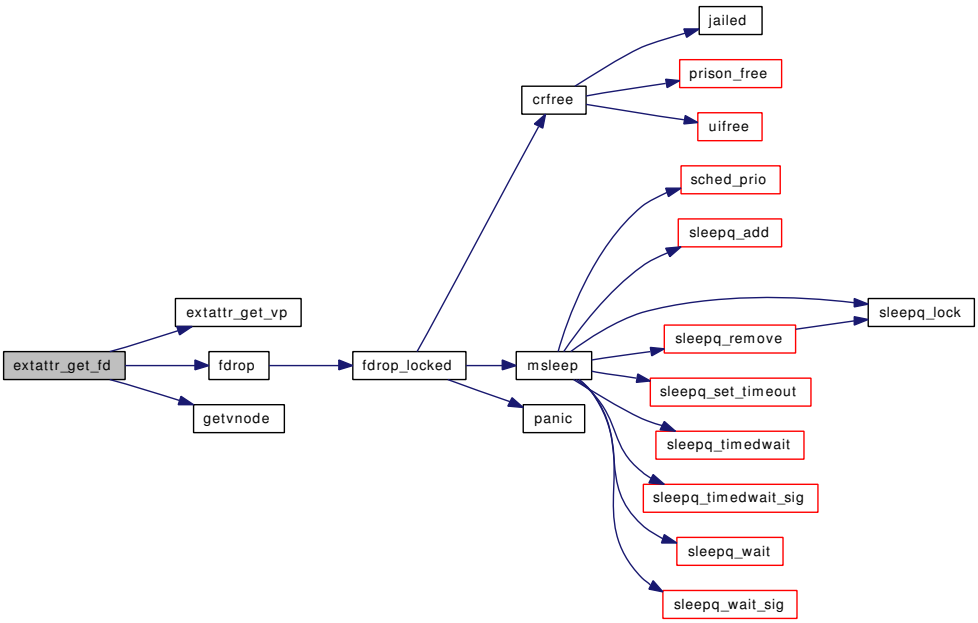


9.157.1.6 int extattr_get_fd (struct thread * *td*, struct extattr_get_fd_args * *uap*)

Definition at line 382 of file vfs_extattr.c.

References extattr_get_vp(), fdrop(), and getvnode().

Here is the call graph for this function:

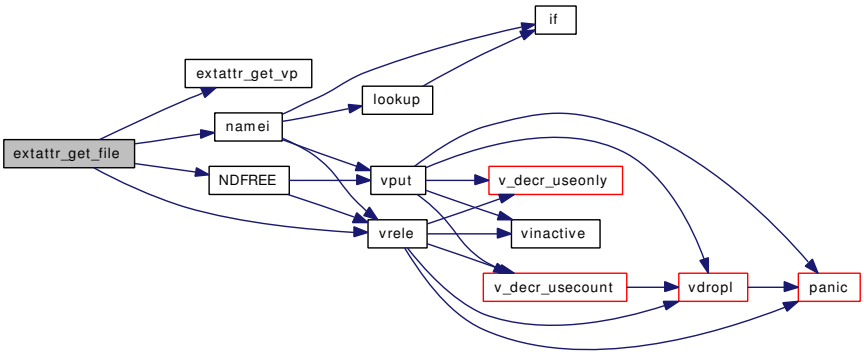


9.157.1.7 int extattr_get_file (struct thread * *td*, struct extattr_get_file_args * *uap*)

Definition at line 417 of file vfs_extattr.c.

References extattr_get_vp(), namei(), NDFREE(), and vrele().

Here is the call graph for this function:

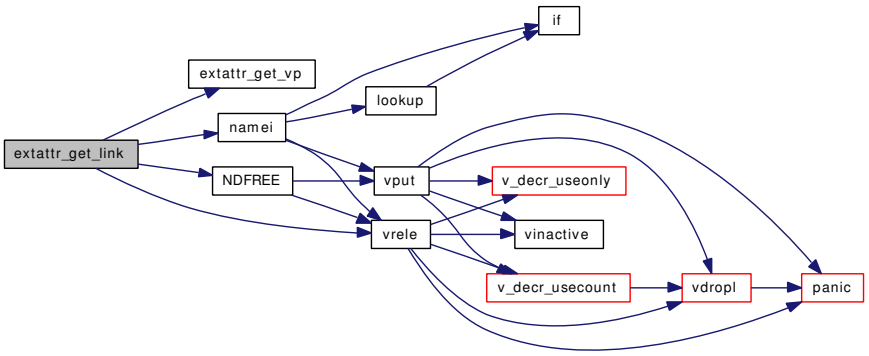


9.157.1.8 int extattr_get_link (struct thread * *td*, struct extattr_get_link_args * *uap*)

Definition at line 454 of file vfs_extattr.c.

References extattr_get_vp(), namei(), NDFREE(), and vrele().

Here is the call graph for this function:



9.157.1.9 static int extattr_get_vp (struct vnode * *vp*, int *attrnamespace*, const char * *attrname*, void * *data*, size_t *nbytes*, struct thread * *td*) [static]

Definition at line 320 of file vfs_extattr.c.

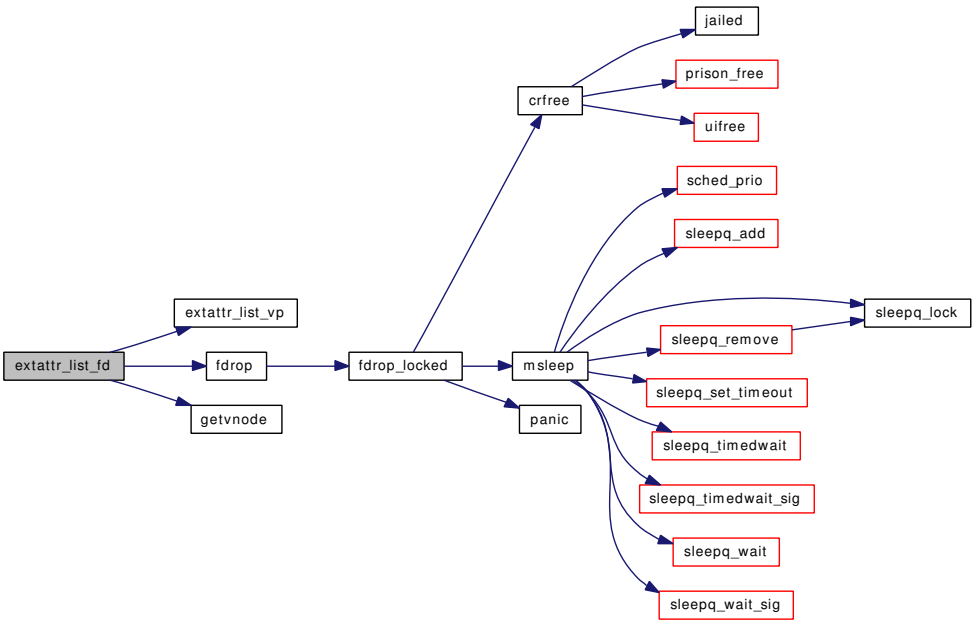
Referenced by extattr_get_fd(), extattr_get_file(), and extattr_get_link().

9.157.1.10 int extattr_list_fd (struct thread * *td*, struct extattr_list_fd_args * *uap*)

Definition at line 700 of file vfs_extattr.c.

References extattr_list_vp(), fdrop(), and getvnode().

Here is the call graph for this function:

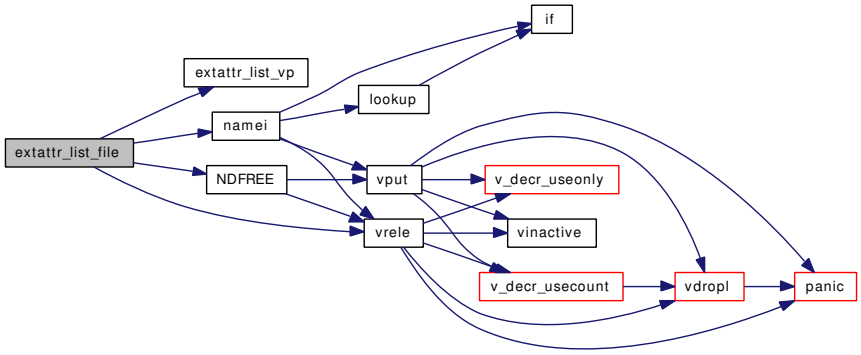


9.157.1.11 int extattr_list_file (struct thread* *td*, struct extattr_list_file_args * *uap*)

Definition at line 728 of file vfs_extattr.c.

References extattr_list_vp(), namei(), NDFREE(), and vrele().

Here is the call graph for this function:

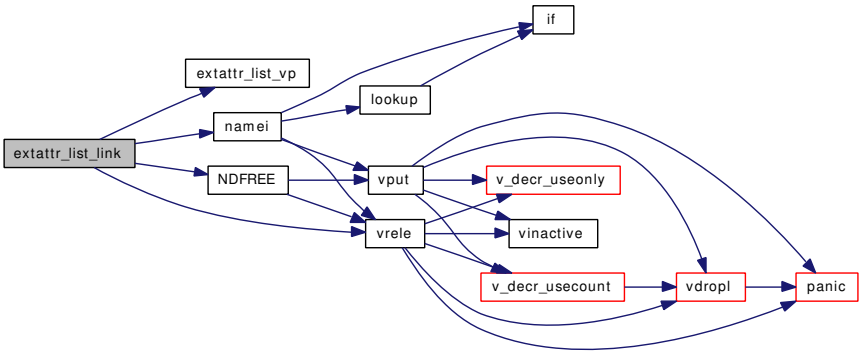


9.157.1.12 int extattr_list_link (struct thread* *td*, struct extattr_list_link_args * *uap*)

Definition at line 758 of file vfs_extattr.c.

References extattr_list_vp(), namei(), NDFREE(), and vrele().

Here is the call graph for this function:



9.157.1.13 static int extattr_list_vp (struct vnode * *vp*, int *attrnamespace*, void * *data*, size_t *nbytes*, struct thread * *td*) [static]

Definition at line 643 of file vfs_extattr.c.

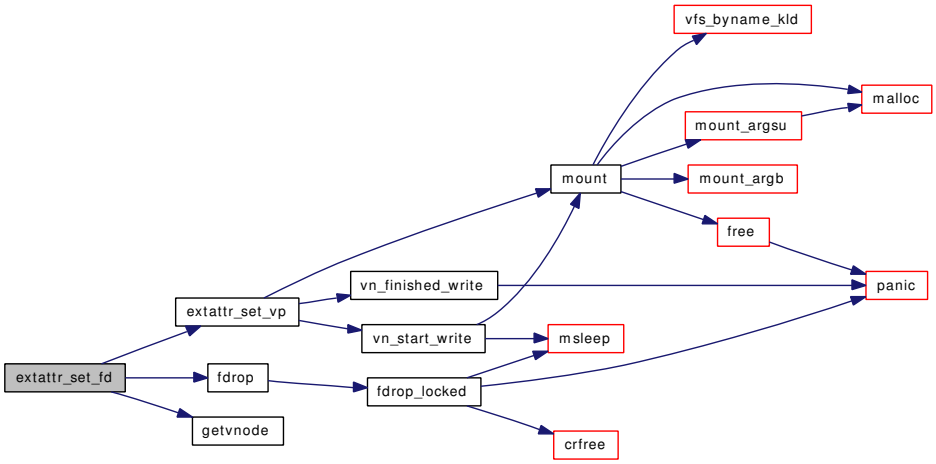
Referenced by extattr_list_fd(), extattr_list_file(), and extattr_list_link().

9.157.1.14 int extattr_set_fd (struct thread * *td*, struct extattr_set_fd_args * *uap*)

Definition at line 201 of file vfs_extattr.c.

References extattr_set_vp(), fdrop(), and getvnode().

Here is the call graph for this function:

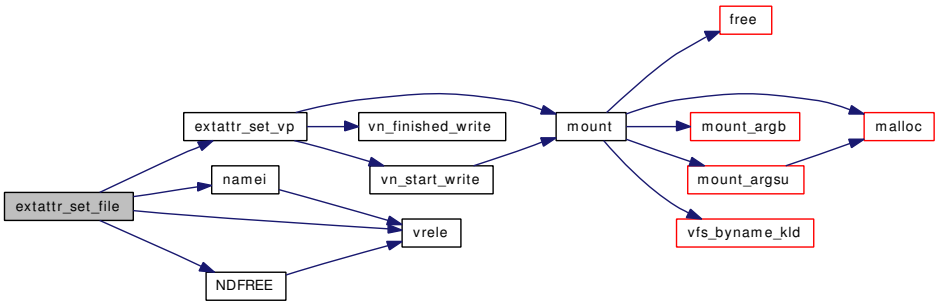


9.157.1.15 int extattr_set_file (struct thread * *td*, struct extattr_set_file_args * *uap*)

Definition at line 236 of file vfs_extattr.c.

References extattr_set_vp(), namei(), NDFREE(), and vrele().

Here is the call graph for this function:

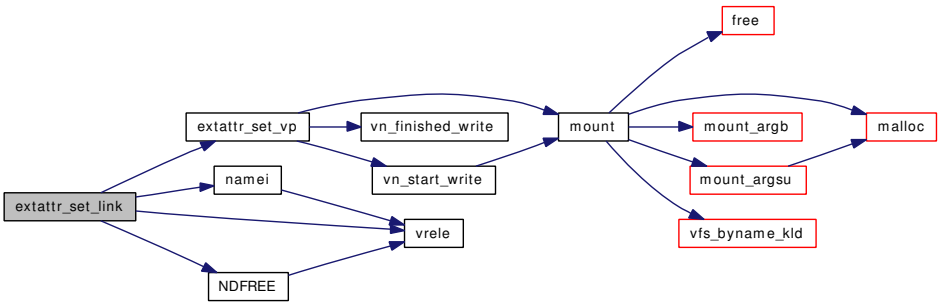


9.157.1.16 int extattr_set_link (struct thread * *td*, struct extattr_set_link_args * *uap*)

Definition at line 273 of file vfs_extattr.c.

References extattr_set_vp(), namei(), NDFREE(), and vrele().

Here is the call graph for this function:



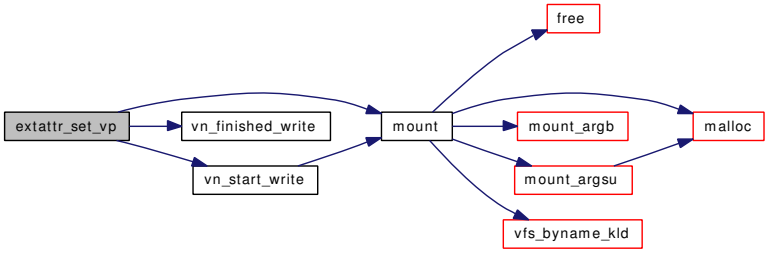
9.157.1.17 static int extattr_set_vp (struct vnode * *vp*, int *attrnamespace*, const char * *attrname*, void * *data*, size_t *nbytes*, struct thread * *td*) [static]

Definition at line 151 of file vfs_extattr.c.

References mount(), vn_finished_write(), and vn_start_write().

Referenced by extattr_set_fd(), extattr_set_file(), and extattr_set_link().

Here is the call graph for this function:

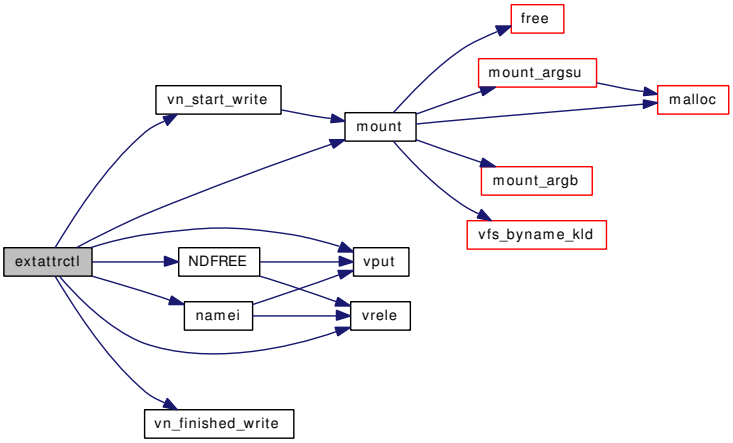


9.157.1.18 int extattrctl (struct thread * *td*, struct extattrctl_args * *uap*)

Definition at line 58 of file vfs_extattr.c.

References mount(), namei(), NDFREE(), vn_finished_write(), vn_start_write(), vput(), and vrele().

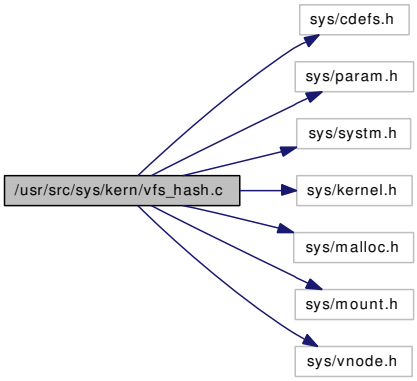
Here is the call graph for this function:


free
mount_argsu
malloc
vn_start_write
mount
mount_argb
vfs_byname_kld
extattrctl
NDFREE
vput
namei
vrele
vn_finished_write

9.158 /usr/src/sys/kern/vfs_hash.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/malloc.h>
#include <sys/mount.h>
#include <sys/vnode.h>
```

Include dependency graph for vfs_hash.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/vfs_hash.c,v 1.12 2006/04/18 08:05:08 delphij Exp $")
- static MALLOC_DEFINE (M_VFS_HASH,"vfs_hash","VFS hash table")
- static LIST_HEAD (vfs_hash_head, vnode)
- static struct vfs_hash_head * vfs_hash_index (const struct mount *mp, u_int hash)
- int vfs_hash_get (const struct mount *mp, u_int hash, int flags, struct thread *td, struct vnode **vpp, vfs_hash_cmp_t *fn, void *arg)
- void vfs_hash_remove (struct vnode *vp)
- int vfs_hash_insert (struct vnode *vp, u_int hash, int flags, struct thread *td, struct vnode **vpp, vfs_hash_cmp_t *fn, void *arg)
- void vfs_hash_rehash (struct vnode *vp, u_int hash)

9.158.1 Function Documentation

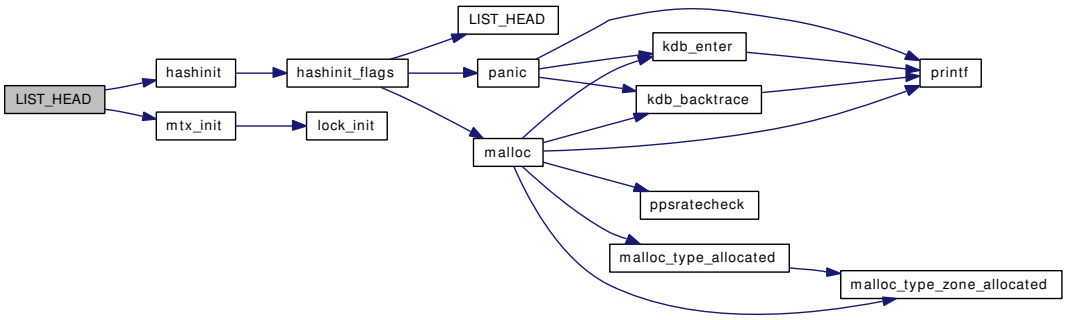
9.158.1.1 __FBSDID ("$FreeBSD: src/sys/kern/vfs_hash. *c*, v 1.12 2006/04/18 08:05:08 delphij Exp $")

9.158.1.2 static LIST_HEAD (vfs_hash_head, vnode) [static]

Definition at line 40 of file vfs_hash.c.

References desiredvnodes, hashinit(), and mtx_init().

Here is the call graph for this function:



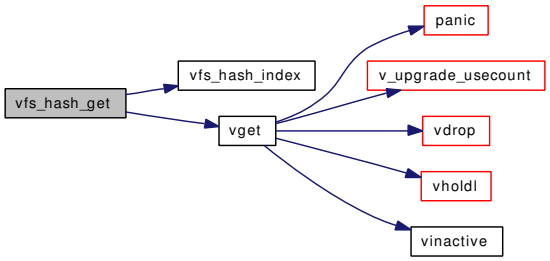
9.158.1.3 static MALLOC_DEFINE (M_VFS_HASH, "vfs_hash", "VFS hash table") [static]

9.158.1.4 int vfs_hash_get (const struct mount * *mp*, u_int *hash*, int *flags*, struct thread * *td*, struct vnode ** *vpp*, vfs_hash_cmp_t * *fn*, void * *arg*)

Definition at line 65 of file vfs_hash.c.

References vfs_hash_index(), and vget().

Here is the call graph for this function:



9.158.1.5 static struct vfs_hash_head* vfs_hash_index (const struct mount * *mp*, u_int *hash*) [static]

Definition at line 58 of file vfs_hash.c.

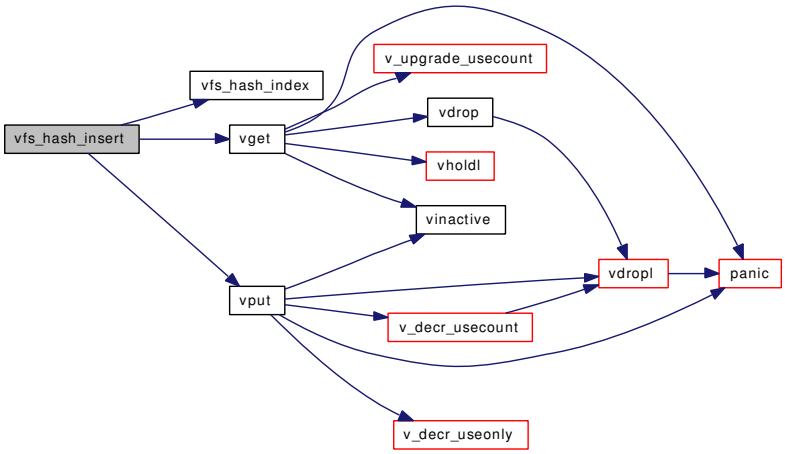
Referenced by vfs_hash_get(), vfs_hash_insert(), and vfs_hash_rehash().

9.158.1.6 int vfs_hash_insert (struct vnode * *vp*, u_int *hash*, int *flags*, struct thread * *td*, struct vnode ** *vpp*, vfs_hash_cmp_t * *fn*, void * *arg*)

Definition at line 107 of file vfs_hash.c.

References vfs_hash_index(), vget(), and vput().

Here is the call graph for this function:



9.158.1.7 void vfs_hash_rehash (struct vnode * *vp*, u_int *hash*)

Definition at line 148 of file vfs_hash.c.

References vfs_hash_index().

Here is the call graph for this function:

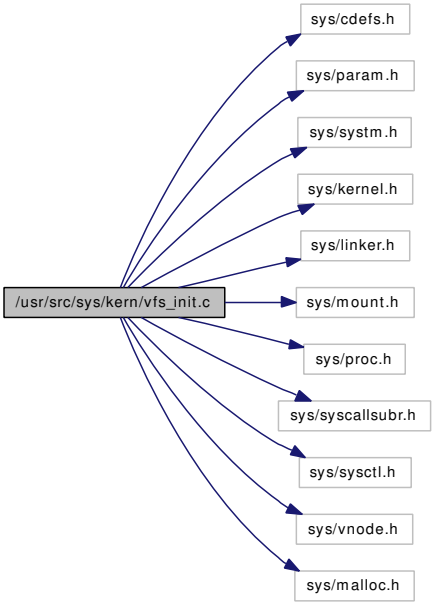
9.158.1.8 void vfs_hash_remove (struct vnode * *vp*)

Definition at line 98 of file vfs_hash.c.

9.159 /usr/src/sys/kern/vfs_init.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/linker.h>
#include <sys/mount.h>
#include <sys/proc.h>
#include <sys/syscallsubr.h>
#include <sys/sysctl.h>
#include <sys/vnode.h>
#include <sys/malloc.h>
```

Include dependency graph for vfs_init.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/vfs_init.c,v 1.85 2007/02/16 17:32:41 pjd Exp $")
- static int vfs_register (struct vfsconf *)
- static int vfs_unregister (struct vfsconf *)
- MALLOC_DEFINE (M_VNODE,"vnodes","Dynamically allocated vnodes")
- vfsconf * vfs_byname (const char *name)
- vfsconf * vfs_byname_kld (const char *fstype, struct thread *td, int *error)
- int vfs_modevent (module_t mod, int type, void *data)

Variables

- int maxvfsconf = VFS_GENERIC + 1
- vfsconfhead vfsconf = TAILQ_HEAD_INITIALIZER(vfsconf)
- vattr va_null

9.159.1 Function Documentation

9.159.1.1 __FBSDID ("$FreeBSD: src/sys/kern/vfs_init. *c*, v 1.85 2007/02/16 17:32:41 pjd Exp $")

9.159.1.2 MALLOC_DEFINE (M_VNODE, "vnodes", "Dynamically allocated vnodes")

9.159.1.3 struct vfsconf* vfs_byname (const char * *name*)

Definition at line 96 of file vfs_init.c.

References vfsconf.

Referenced by devfs_first(), vfs_byname_kld(), vfs_domount(), vfs_register(), and vfs_unregister().

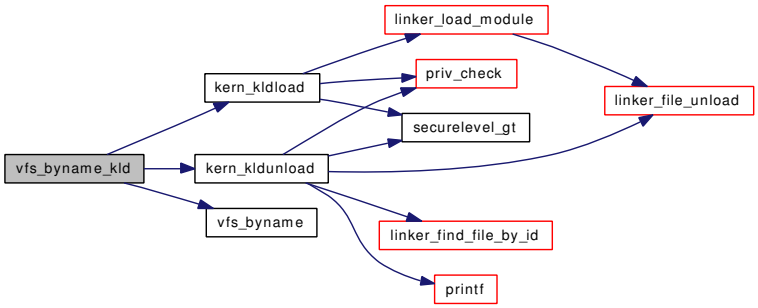
9.159.1.4 struct vfsconf* vfs_byname_kld (const char * *fstype*, struct thread * *td*, int * *error*)

Definition at line 109 of file vfs_init.c.

References kern_kldload(), kern_kldunload(), vfs_byname(), and vfsconf.

Referenced by mount(), and vfs_domount().

Here is the call graph for this function:

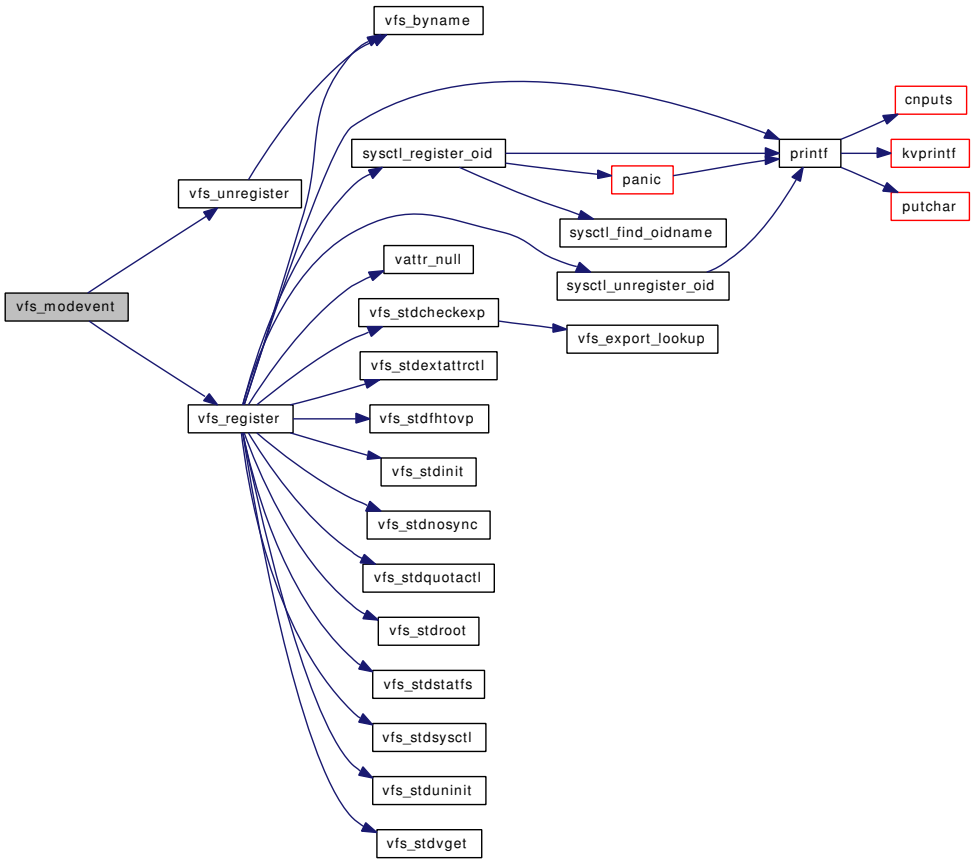


9.159.1.5 int vfs_modevent (module_t *mod*, int *type*, void * *data*)

Definition at line 272 of file vfs_init.c.

References vfs_register(), vfs_unregister(), and vfsconf.

Here is the call graph for this function:



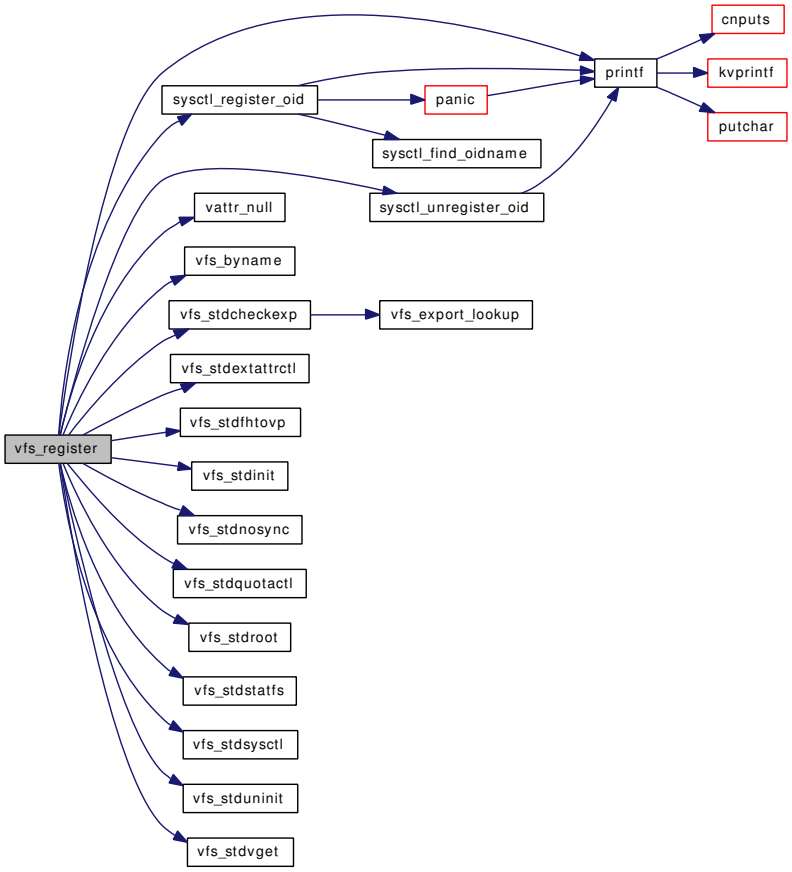
9.159.1.6 static int vfs_register (struct vfsconf *) [static]

Definition at line 136 of file vfs_init.c.

References maxvfsconf, printf(), sysctl_register_oid(), sysctl_unregister_oid(), va_null, vattr_null(), vfs_byname(), vfs_stdcheckexp(), vfs_stdextattrctl(), vfs_stdfhtovp(), vfs_stdinit(), vfs_stdnosync(), vfs_stdquotactl(), vfs_stdroot(), vfs_stdstatfs(), vfs_stdsysctl(), vfs_stduninit(), vfs_stdvget(), and vfsconf.

Referenced by vfs_modevent().

Here is the call graph for this function:



9.159.1.7 static int vfs_unregister (struct vfsconf *) [static]

Definition at line 241 of file vfs_init.c.

References maxvfsconf, vfs_byname(), and vfsconf.

Referenced by vfs_modevent().

Here is the call graph for this function:

9.159.2 Variable Documentation

9.159.2.1 int maxvfsconf = VFS_GENERIC + 1

Definition at line 59 of file vfs_init.c.

Referenced by vfs_register(), vfs_sysctl(), and vfs_unregister().

9.159.2.2 struct vattr va_null

Definition at line 72 of file vfs_init.c.

Referenced by vfs_register().

9.159.2.3 struct vfsconfhead vfsconf = TAILQ_HEAD_INITIALIZER(vfsconf)

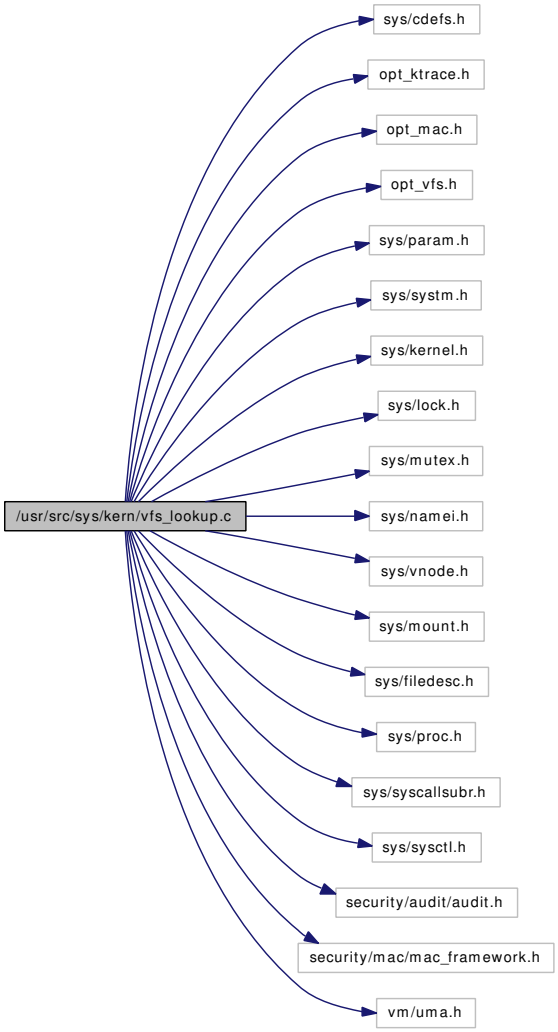
Definition at line 65 of file vfs_init.c.

Referenced by devfs_first(), mount(), sysctl_ovfs_conf(), sysctl_vfs_conflist(), vfs_byname(), vfs_byname_kld(), vfs_domount(), vfs_modevent(), vfs_register(), vfs_sysctl(), and vfs_unregister().

9.160 /usr/src/sys/kern/vfs_lookup.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ktrace.h"
#include "opt_mac.h"
#include "opt_vfs.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/kernel.h>
#include <sys/lock.h>
#include <sys/mutex.h>
#include <sys/namei.h>
#include <sys/vnode.h>
#include <sys/mount.h>
#include <sys/filedesc.h>
#include <sys/proc.h>
#include <sys/syscallsubr.h>
#include <sys/sysctl.h>
#include <security/audit/audit.h>
#include <security/mac/mac_framework.h>
#include <vm/uma.h>
```

Include dependency graph for vfs_lookup.c:



Defines

- #define NAMEI_DIAGNOSTIC 1

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/vfs_lookup.c,v 1.98 2007/02/15 09:53:49 kib Exp $")
- static void nameiinit (void *dummy __unused)
- SYSCTL_INT (_vfs, OID_AUTO, lookup_shared, CTLFLAG_RW,&lookup_shared, 0,"Enables/Disables shared locks for path name translation")
- int namei (struct nameidata *ndp)
- static int compute_cn_lkflags (struct mount *mp, int lkflags)
- int lookup (struct nameidata *ndp)
- int relookup (struct vnode *dvp, struct vnode **vpp, struct componentname *cnp)
- void NDFREE (struct nameidata *ndp, const u_int flags)
- int kern_alternate_path (struct thread *td, const char *prefix, char *path, enum uio_seg pathseg, char **pathbuf, int create)

Variables

- uma_zone_t namei_zone
- static struct vnode * vp_crossmp
- static int lookup_shared = 0

9.160.1 Define Documentation

9.160.1.1 #define NAMEI_DIAGNOSTIC 1

Definition at line 65 of file vfs_lookup.c.

9.160.2 Function Documentation

9.160.2.1 __FBSDID ("$FreeBSD: src/sys/kern/vfs_lookup. *c*, v 1.98 2007/02/15 09:53:49 kib Exp $")

9.160.2.2 static int compute_cn_lkflags (struct mount * *mp*, int *lkflags*) [static]

Definition at line 312 of file vfs_lookup.c.

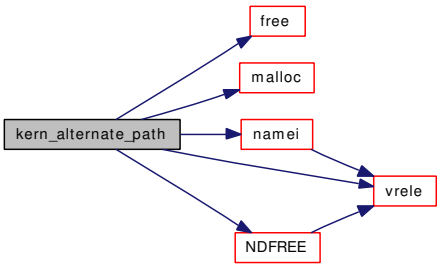
Referenced by lookup().

9.160.2.3 int kern_alternate_path (struct thread * *td*, const char * *prefix*, char * *path*, enum uio_seg *pathseg*, char ** *pathbuf*, int *create*)

Definition at line 999 of file vfs_lookup.c.

References buf, free(), malloc(), namei(), NDFREE(), and vrele().

Here is the call graph for this function:

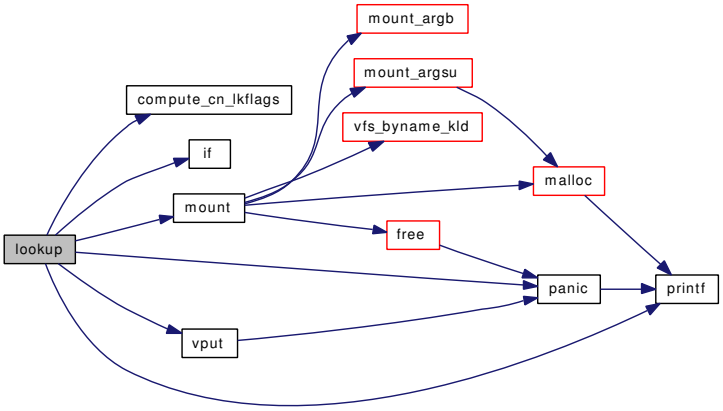


9.160.2.4 int lookup (struct nameidata * *ndp*)

Definition at line 361 of file vfs_lookup.c.

References compute_cn_lkflags(), if(), mount(), panic(), printf(), rootvnode, td, and vput().

Here is the call graph for this function:



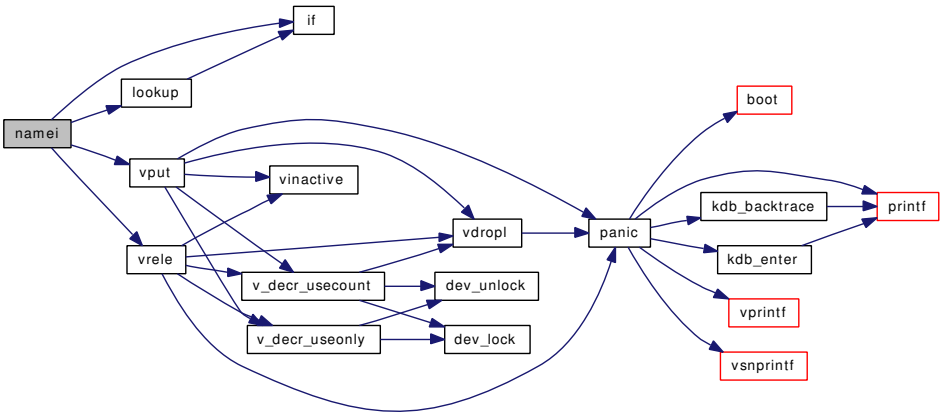
9.160.2.5 int namei (struct nameidata * *ndp*)

Definition at line 116 of file vfs_lookup.c.

References Giant, if(), lookup(), namei_zone, td, vput(), and vrele().

Referenced by __acl_aclcheck_file(), __acl_aclcheck_link(), __acl_delete_file(), __acl_delete_link(), __acl_get_file(), __acl_get_link(), __acl_set_file(), __acl_set_link(), chflags(), chroot(), devfs_fixup(), do_execve(), extattr_delete_file(), extattr_delete_link(), extattr_get_file(), extattr_get_link(), extattr_list_file(), extattr_list_link(), extattr_set_file(), extattr_set_link(), extattrctl(), getfh(), jail(), kern_access(), kern_alternate_path(), kern_chdir(), kern_chmod(), kern_chown(), kern_eaccess(), kern_lchown(), kern_link(), kern_lstat(), kern_lutimes(), kern_mkdir(), kern_mkfifo(), kern_mknod(), kern_pathconf(), kern_readlink(), kern_rename(), kern_rmdir(), kern_stat(), kern_statfs(), kern_symlink(), kern_truncate(), kern_unlink(), kern_utimes(), lchflags(), lchmod(), lgetfh(), quotactl(), revoke(), ttioctl(), uipc_bind(), undelete(), unp_connect(), vfs_domount(), and vn_open_cred().

Here is the call graph for this function:

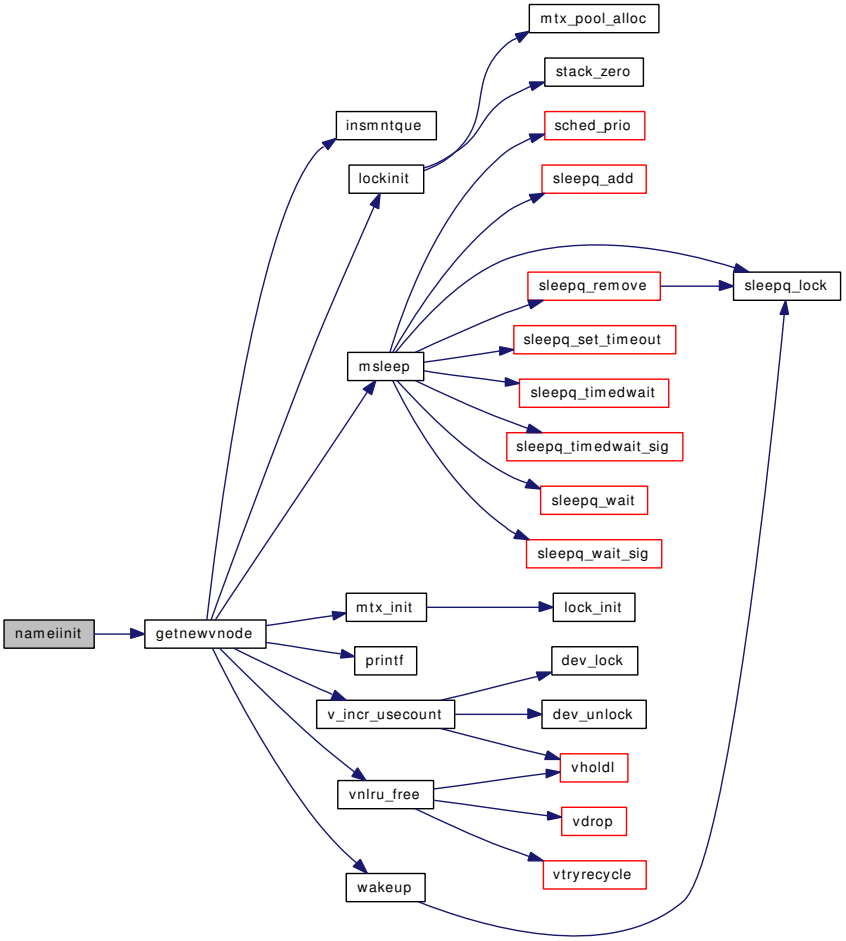


9.160.2.6 static void nameiinit (void *dummy *__unused*) [static]

Definition at line 78 of file vfs_lookup.c.

References getnewvnode(), namei_zone, and vp_crossmp.

Here is the call graph for this function:



9.160.2.7 void NDFREE (struct nameidata * *ndp*, const u_int *flags*)

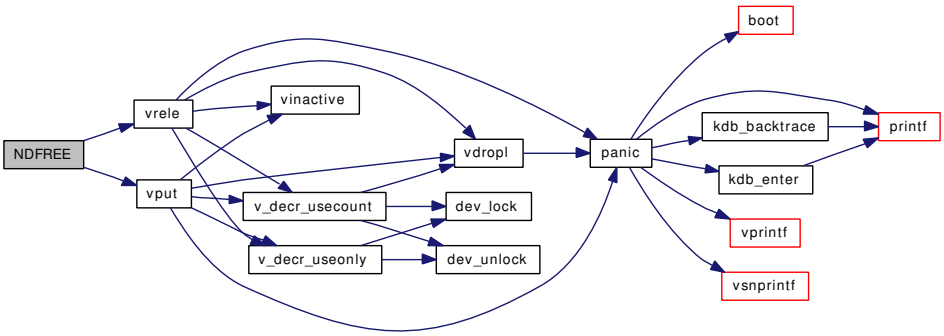
Definition at line 940 of file vfs_lookup.c.

References namei_zone, vput(), and vrele().

Referenced by __acl_aclcheck_file(), __acl_aclcheck_link(), __acl_delete_file(), __acl_delete_link(), __acl_get_file(), __acl_get_link(), __acl_set_file(), __acl_set_link(), acct(), alq_open(), chflags(), chroot(), cn_devopen(), coredump(), devfs_fixup(), do_execve(), extattr_delete_file(), extattr_delete_link(), extattr_get_file(), extattr_get_link(), extattr_list_file(), extattr_list_link(), extattr_set_file(), extattr_set_link(), extattrctl(), fdcheckstd(), getfh(), jail(), kern_access(), kern_alternate_path(), kern_chdir(), kern_chmod(), kern_chown(), kern_eaccess(), kern_lchown(), kern_link(), kern_lstat(), kern_lutimes(), kern_mkdir(), kern_mkfifo(), kern_mknod(), kern_open(), kern_pathconf(), kern_readlink(), kern_rename(), kern_rmdir(), kern_stat(), kern_statfs(), kern_symlink(), kern_truncate(), kern_unlink(), kern_utimes(),

ktrace(), lchflags(), lchmod(), lgetfh(), link_elf_load_file(), linker_hints_lookup(), linker_lookup_file(), quotactl(), revoke(), ttioctl(), uipc_bind(), undelete(), unp_connect(), vfs_domount(), and vn_open_cred().

Here is the call graph for this function:

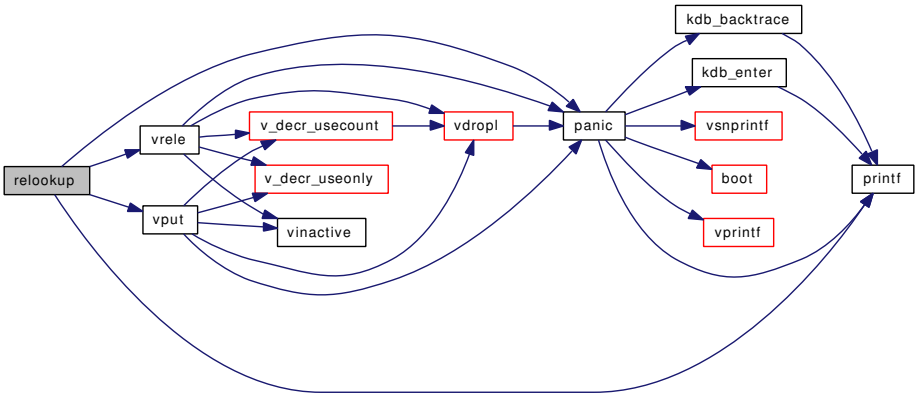


9.160.2.8 int relookup (struct vnode * *dvp*, struct vnode ** *vpp*, struct componentname * *cnp*)

Definition at line 797 of file vfs_lookup.c.

References panic(), printf(), td, vput(), and vrele().

Here is the call graph for this function:



9.160.2.9 SYSCTL_INT (_vfs, OID_AUTO, lookup_shared, CTLFLAG_RW, & *lookup_shared*, 0, "Enables/Disables shared locks for path name translation")

9.160.3 Variable Documentation

9.160.3.1 int lookup_shared = 0 [static]

Definition at line 90 of file vfs_lookup.c.

9.160.3.2 uma_zone_t namei_zone

Definition at line 71 of file vfs_lookup.c.

Referenced by kern_symlink(), namei(), nameiinit(), and NDFREE().

9.160.3.3 struct vnode* vp_crossmp [static]

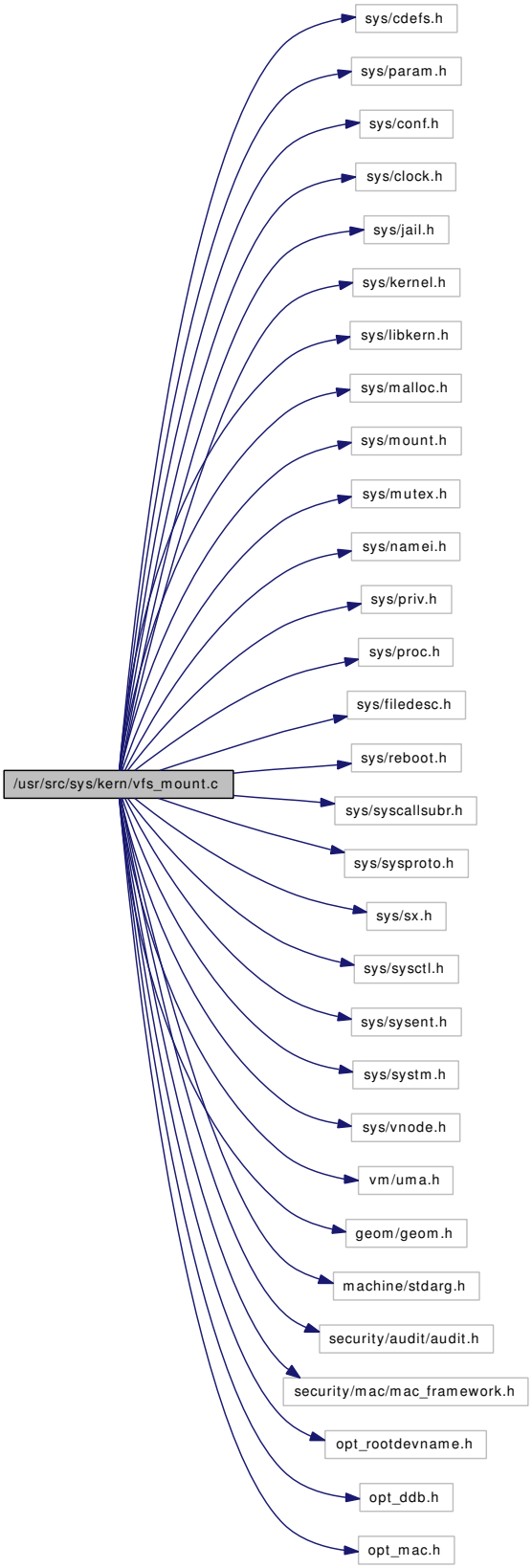
Definition at line 75 of file vfs_lookup.c.

Referenced by nameiinit().

9.161 /usr/src/sys/kern/vfs_mount.c File Reference

```
#include <sys/cdefs.h>
#include <sys/param.h>
#include <sys/conf.h>
#include <sys/clock.h>
#include <sys/jail.h>
#include <sys/kernel.h>
#include <sys/libkern.h>
#include <sys/malloc.h>
#include <sys/mount.h>
#include <sys/mutex.h>
#include <sys/namei.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/filedesc.h>
#include <sys/reboot.h>
#include <sys/syscallsubr.h>
#include <sys/sysproto.h>
#include <sys/sx.h>
#include <sys/sysctl.h>
#include <sys/sysent.h>
#include <sys/systm.h>
#include <sys/vnode.h>
#include <vm/uma.h>
#include <geom/geom.h>
#include <machine/stdarg.h>
#include <security/audit/audit.h>
#include <security/mac/mac_framework.h>
#include "opt_rootdevname.h"
#include "opt_ddb.h"
#include "opt_mac.h"
```

Include dependency graph for vfs_mount.c:


/usr/src/sys/kern/vfs_mount.c
sys/cdefs.h
sys/param.h
sys/conf.h
sys/clock.h
sys/jail.h
sys/kernel.h
sys/libkern.h
sys/malloc.h
sys/mount.h
sys/mutex.h
sys/namei.h
sys/priv.h
sys/proc.h
sys/filedesc.h
sys/reboot.h
sys/syscallsubr.h
sys/sysproto.h
sys/sx.h
sys/sysctl.h
sys/sysent.h
sys/systm.h
sys/vnode.h
vm/uma.h
geom/geom.h
machine/stdarg.h
security/audit/audit.h
security/mac/mac_framework.h
opt_rootdevname.h
opt_ddb.h
opt_mac.h

Data Structures

- struct vfsopt
- struct mount_args
- struct unmount_args
- struct root_hold_token
- struct mntaarg
- struct mntarg

Defines

- #define ROOTNAME "root_device"
- #define VFS_MOUNTARG_SIZE_MAX (1024 * 64)
- #define ROOTDEVNAME NULL

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/vfs_mount.c,v 1.244 2007/02/13 01:28:48 cognet Exp $")
- static int vfs_domount (struct thread *td, const char *fstype, char *fspath, int fsflags, void *fsdata)
- static struct mount * vfs_mount_alloc (struct vnode *dvp, struct vfsconf *vfsp, const char *fspath, struct thread *td)
- static int vfs_mountroot_ask (void)
- static int vfs_mountroot_try (const char *mountfrom)
- static int vfs_donmount (struct thread *td, int fsflags, struct uio *fsoptions)
- static void free_mntarg (struct mntarg *ma)
- static void vfs_mount_destroy (struct mount *)
- static int vfs_getopt_pos (struct vfsoptlist *opts, const char *name)
- SYSCTL_INT (_vfs, OID_AUTO, usermount, CTLFLAG_RW,&usermount, 0,"Unprivileged users may mount and unmount file systems")
- MALLOC_DEFINE (M_MOUNT,"mount","vfs mount structure")
- MALLOC_DEFINE (M_VNODE_MARKER,"vnodemarker","vnode marker")
- MTX_SYSINIT (mountlist,&mountlist_mtx,"mountlist", MTX_DEF)
- TAILQ_HEAD (vfsoptlist, vfsopt)
- static void vfs_freeopt (struct vfsoptlist *opts, struct vfsopt *opt)
- static void vfs_freeopts (struct vfsoptlist *opts)
- void vfs_deleteopt (struct vfsoptlist *opts, const char *name)
- static int vfs_equalopts (const char *opt1, const char *opt2)
- static void vfs_sanitizeopts (struct vfsoptlist *opts)
- static int vfs_buildopts (struct uio *auio, struct vfsoptlist **options)
- static void vfs_mergeopts (struct vfsoptlist *toopts, struct vfsoptlist *opts)
- int nmount (struct thread *td, struct nmount_args *uap)
- void vfs_ref (struct mount *mp)
- void vfs_rel (struct mount *mp)
- static int mount_init (void *mem, int size, int flags)
- static void mount_fini (void *mem, int size)
- int mount (struct thread *td, struct mount_args *uap)
- int unmount (struct thread *td, struct unmount_args *uap)
- int dounmount (struct mount *mp, int flags, struct thread *td)
- static LIST_HEAD (root_hold_token)
- void root_mount_rel (struct root_hold_token *h)

- static void root_mount_wait (void)
- static void set_rootvnode (struct thread *td)
- static void devfs_first (void)
- static void devfs_fixup (struct thread *td)
- void vfs_mount_error (struct mount *mp, const char *fmt,...)
- void vfs_mountroot (void)
- int vfs_filteropt (struct vfsoptlist *opts, const char **legal)
- int vfs_getopt (struct vfsoptlist *opts, const char *name, void **buf, int *len)
- char * vfs_getopts (struct vfsoptlist *opts, const char *name, int *error)
- int vfs_flagopt (struct vfsoptlist *opts, const char *name, u_int *w, u_int val)
- int vfs_scanopt (struct vfsoptlist *opts, const char *name, const char *fmt,...)
- int vfs_copyopt (struct vfsoptlist *opts, const char *name, void *dest, int len)
- vnode * __mnt_vnode_next (struct vnode **mvp, struct mount *mp)
- vnode * __mnt_vnode_first (struct vnode **mvp, struct mount *mp)
- void __mnt_vnode_markerfree (struct vnode **mvp, struct mount *mp)
- int __vfs_statfs (struct mount *mp, struct statfs *sbp, struct thread *td)
- void vfs_mountedfrom (struct mount *mp, const char *from)
- mntarg * mount_argb (struct mntarg *ma, int flag, const char *name)
- mntarg * mount_argf (struct mntarg *ma, const char *name, const char *fmt,...)
- mntarg * mount_argsu (struct mntarg *ma, const char *name, const void *val, int len)
- mntarg * mount_arg (struct mntarg *ma, const char *name, const void *val, int len)
- int kernel_mount (struct mntarg *ma, int flags)
- int kernel_vmount (int flags,...)

Variables

- static int usermount = 0
- static uma_zone_t mount_zone
- mntlist mountlist = TAILQ_HEAD_INITIALIZER(mountlist)
- mtx mountlist_mtx
- vnode * rootvnode
- static const char * global_opts [ ]
- static char * cdrom_rootdevnames [ ]
- char * rootdevnames [2] = {NULL, NULL}
- static const char * ctrootdevname = ROOTDEVNAME

9.161.1 Define Documentation

9.161.1.1 #define ROOTDEVNAME NULL

Definition at line 161 of file vfs_mount.c.

9.161.1.2 #define ROOTNAME "root_device"

Definition at line 78 of file vfs_mount.c.

Referenced by vfs_mountroot_try().

9.161.1.3 #define VFS_MOUNTARG_SIZE_MAX (1024 ∗ 64)

Definition at line 79 of file vfs_mount.c.

Referenced by vfs_buildopts().

9.161.2 Function Documentation

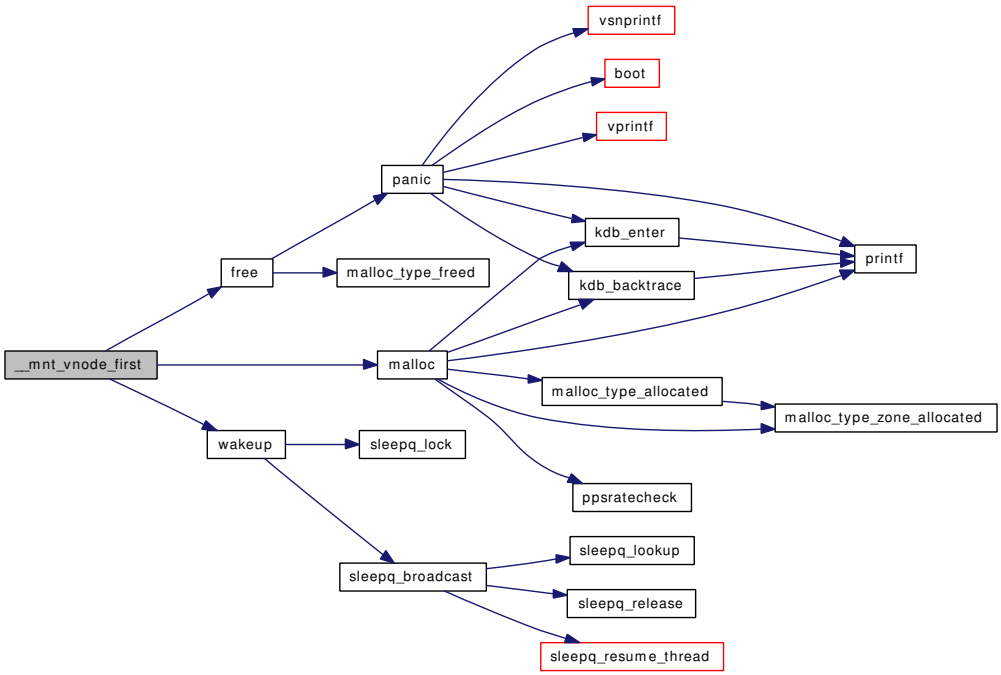
9.161.2.1 __FBSDID ("$FreeBSD: src/sys/kern/vfs_mount. *c*, v 1.244 2007/02/13 01:28:48 cognet Exp $")

9.161.2.2 struct vnode∗ __mnt_vnode_first (struct vnode ∗∗ *mvp*, struct mount ∗ *mp*)

Definition at line 1902 of file vfs_mount.c.

References free(), malloc(), and wakeup().

Here is the call graph for this function:



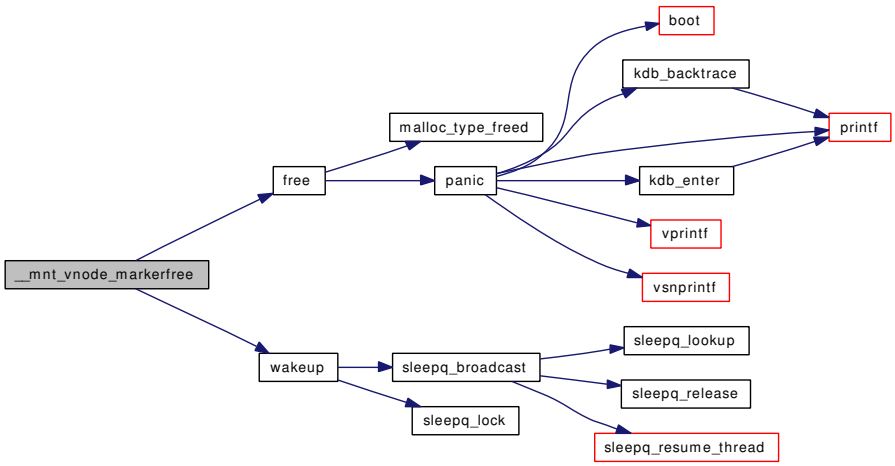
9.161.2.3 void __mnt_vnode_markerfree (struct vnode ∗∗ *mvp*, struct mount ∗ *mp*)

Definition at line 1948 of file vfs_mount.c.

References free(), and wakeup().

Referenced by __mnt_vnode_next().

Here is the call graph for this function:

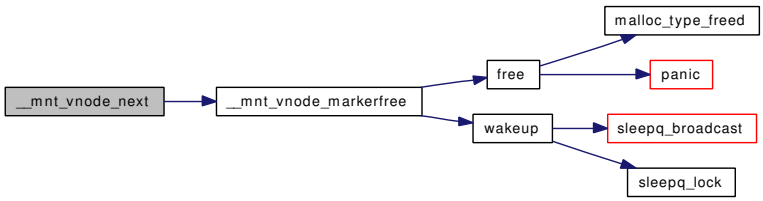


9.161.2.4 struct vnode* __mnt_vnode_next (struct vnode ** *mvp*, struct mount * *mp*)

Definition at line 1880 of file vfs_mount.c.

References __mnt_vnode_markerfree().

Here is the call graph for this function:



9.161.2.5 int __vfs_statfs (struct mount * *mp*, struct statfs * *sbp*, struct thread * *td*)

Definition at line 1971 of file vfs_mount.c.

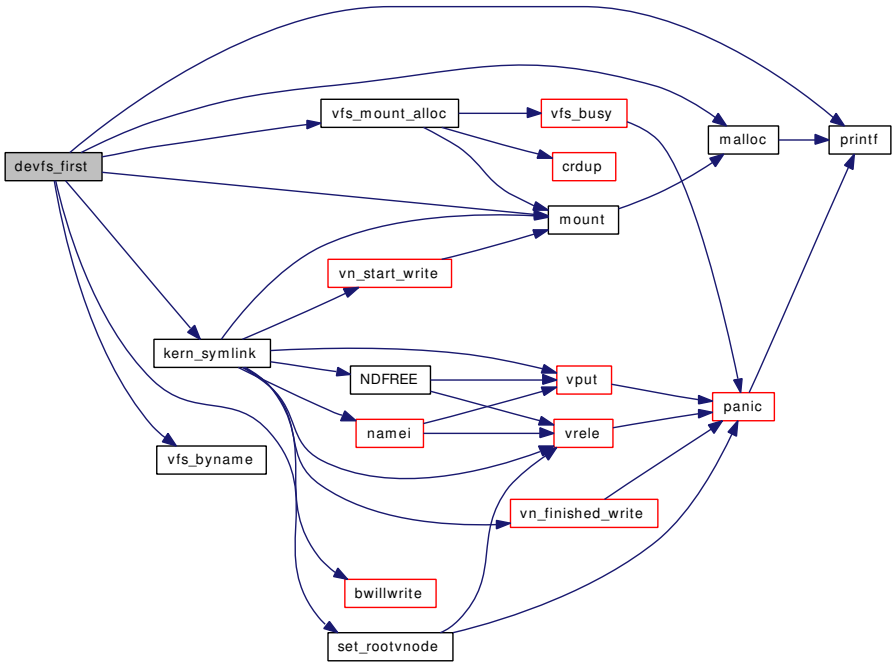
9.161.2.6 static void devfs_first (void) [static]

Definition at line 1399 of file vfs_mount.c.

References kern_symlink(), malloc(), mount(), mountlist, mountlist_mtx, printf(), set_rootvnode(), td, vfs_byname(), vfs_mount_alloc(), and vfsconf.

Referenced by vfs_mountroot().

Here is the call graph for this function:



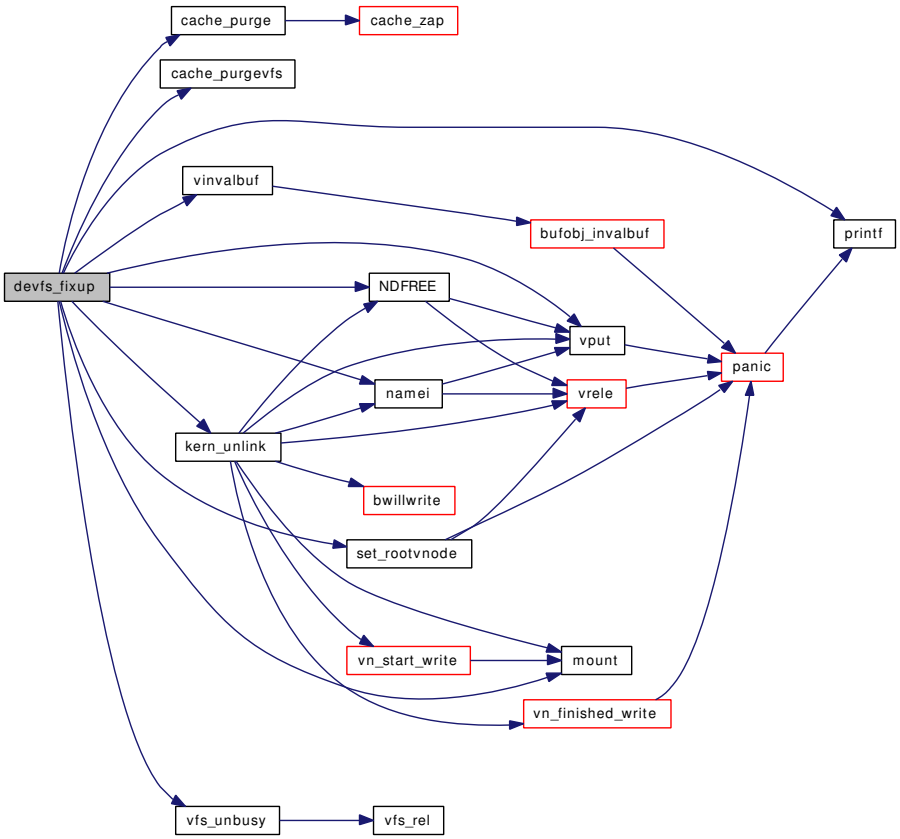
9.161.2.7 static void devfs_fixup (struct thread * *td*) [static]

Definition at line 1439 of file vfs_mount.c.

References cache_purge(), cache_purgevfs(), kern_unlink(), mount(), mountlist, mountlist_mtx, namei(), NDFREE(), printf(), rootvnode, set_rootvnode(), vfs_unbusy(), vinvalbuf(), and vput().

Referenced by vfs_mountroot_try().

Here is the call graph for this function:



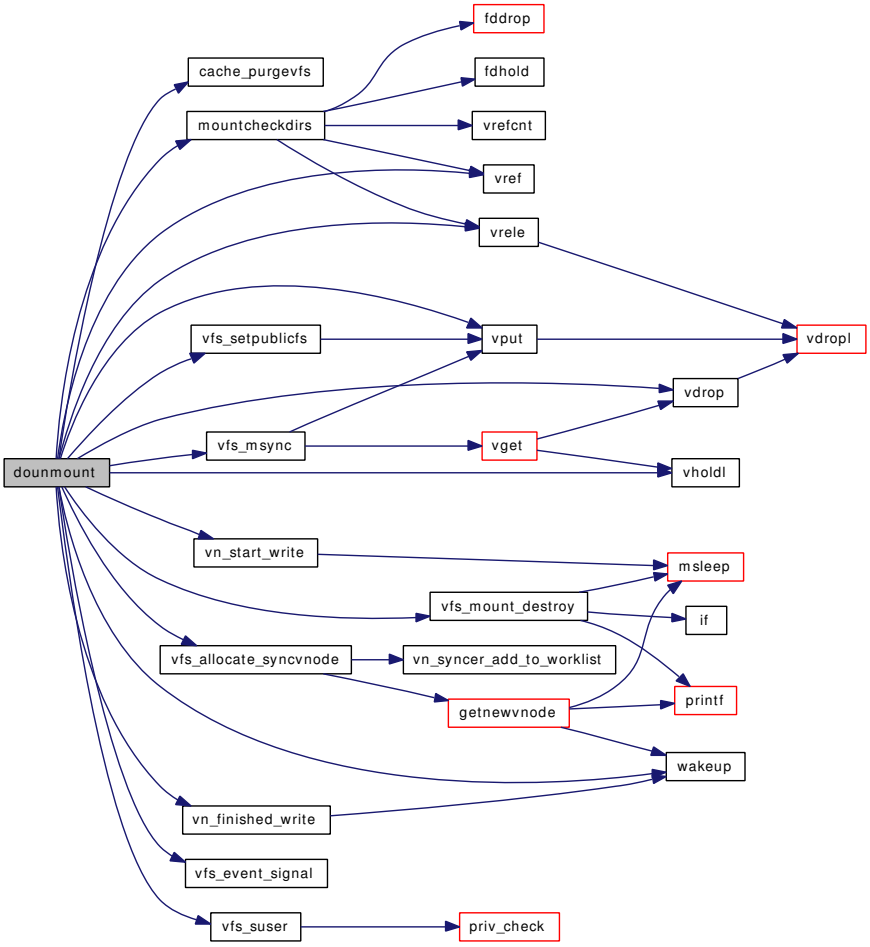
9.161.2.8 int dounmount (struct mount * *mp*, int *flags*, struct thread * *td*)

Definition at line 1167 of file vfs_mount.c.

References cache_purgevfs(), Giant, mountcheckdirs(), mountlist, mountlist_mtx, rootvnode, vdrop(), vfs_allocate_syncvnode(), vfs_event_signal(), vfs_mount_destroy(), vfs_msync(), vfs_setpublicfs(), vfs_-suser(), vholdl(), vn_finished_write(), vn_start_write(), vput(), vref(), vrele(), and wakeup().

Referenced by unmount(), and vfs_unmountall().

Here is the call graph for this function:



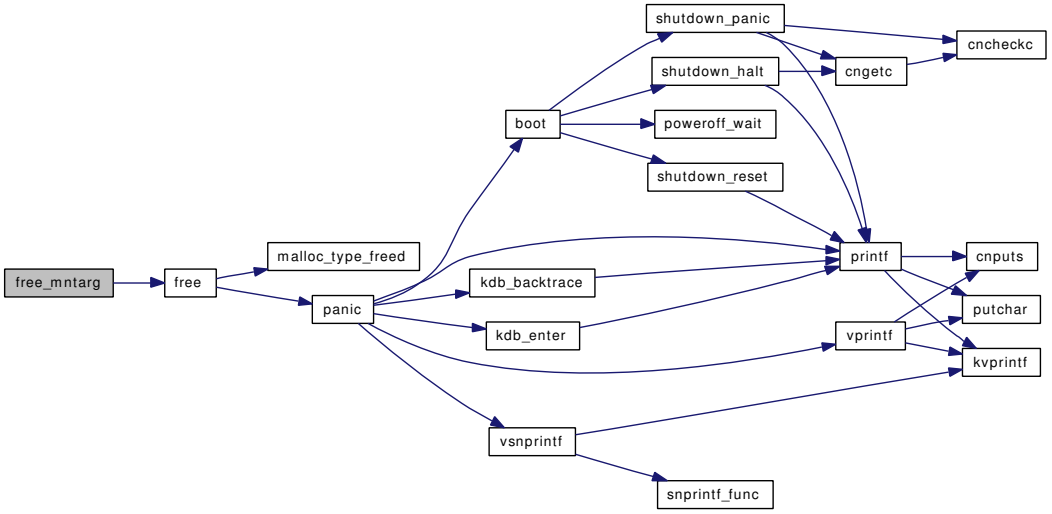
9.161.2.9 static void free_mntarg (struct mntarg * *ma*) [static]

Definition at line 2131 of file vfs_mount.c.

References free(), and mntarg::v.

Referenced by kernel_mount().

Here is the call graph for this function:



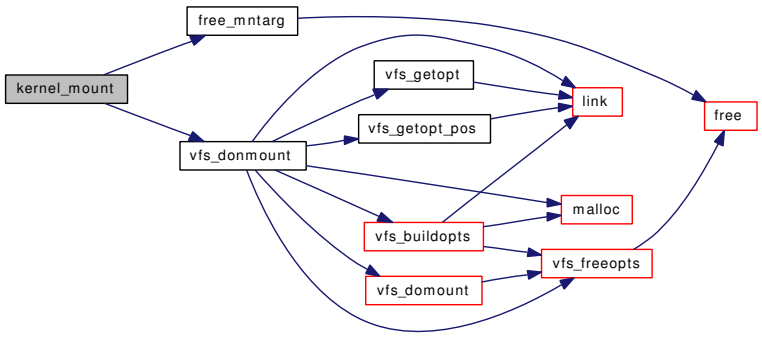
9.161.2.10 int kernel_mount (struct mntarg * *ma*, int *flags*)

Definition at line 2148 of file vfs_mount.c.

References mntarg::error, free_mntarg(), mntarg::len, mntarg::v, and vfs_donmount().

Referenced by kernel_vmount().

Here is the call graph for this function:



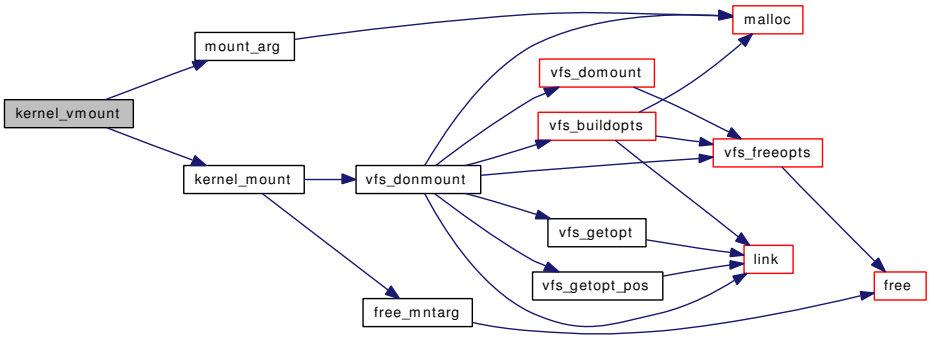
9.161.2.11 int kernel_vmount (int *flags*, ...)

Definition at line 2172 of file vfs_mount.c.

References mntarg::error, kernel_mount(), and mount_arg().

Referenced by vfs_mountroot_try().

Here is the call graph for this function:

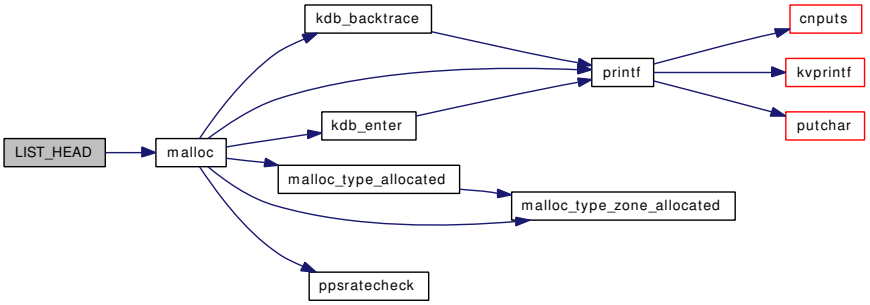


9.161.2.12 static LIST_HEAD (root_hold_token) [static]

Definition at line 1317 of file vfs_mount.c.

References malloc(), and mountlist_mtx.

Here is the call graph for this function:



9.161.2.13 MALLOC_DEFINE (M_VNODE_MARKER, "vnodemarker", "vnode marker")

9.161.2.14 MALLOC_DEFINE (M_MOUNT, "mount", "vfs mount structure")

9.161.2.15 int mount (struct thread * *td*, struct mount_args * *uap*)

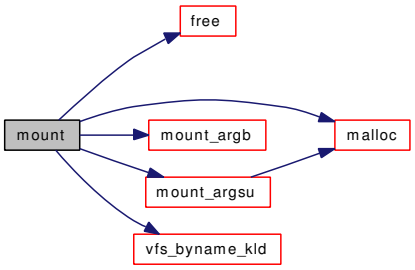
Definition at line 740 of file vfs_mount.c.

References mount_args::data, mntarg::error, mount_args::flags, free(), Giant, malloc(), mount_argb(), mount_argsu(), mount_args::path, mount_args::type, vfs_byname_kld(), and vfsconf.

Referenced by aio_fsync_vnode(), alq_doio(), coredump(), delmntque(), devfs_first(), devfs_fixup(), extattr_delete_vp(), extattr_set_vp(), extattrctl(), fchdir(), fhopen(), fhstat(), flushbufqueues(), fsync(), ftruncate(), kern_fhstatfs(), kern_fstatfs(), kern_getfsstat(), kern_link(), kern_mkdir(), kern_mkfifo(), kern_mknod(), kern_open(), kern_rename(), kern_rmdir(), kern_statfs(), kern_symlink(), kern_truncate(),

kern_unlink(), lookup(), mount_fini(), mount_init(), quotactl(), setfflags(), setfmode(), setfown(), setutimes(), sync(), sync_fsync(), sync_vnode(), sysctl_vfs_ctl(), uipc_bind(), undelete(), unmount(), vacl_delete(), vacl_set_acl(), vfs_domount(), vfs_getnewfsid(), vfs_getvfs(), vfs_mount_alloc(), vfs_mountroot(), vfs_mountroot_try(), vfs_unmountall(), vgonel(), vn_close(), vn_extattr_rm(), vn_extattr_set(), vn_open_cred(), vn_rdwr(), vn_start_secondary_write(), vn_start_write(), vn_write(), vnlru_proc(), vop_stdgetwritemount(), and vtryrecycle().

Here is the call graph for this function:



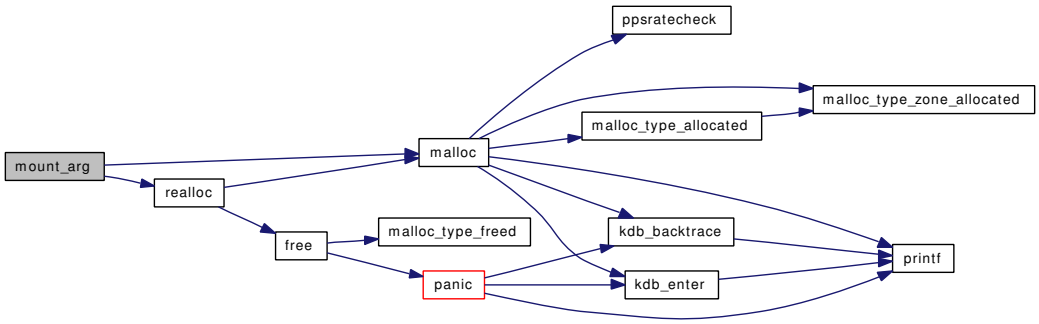
9.161.2.16 struct mntarg* mount_arg (struct mntarg * *ma*, const char * *name*, const void * *val*, int *len*)

Definition at line 2102 of file vfs_mount.c.

References mntarg::error, mntarg::len, malloc(), realloc(), and mntarg::v.

Referenced by kernel_vmount(), mount_argb(), and mount_argsu().

Here is the call graph for this function:



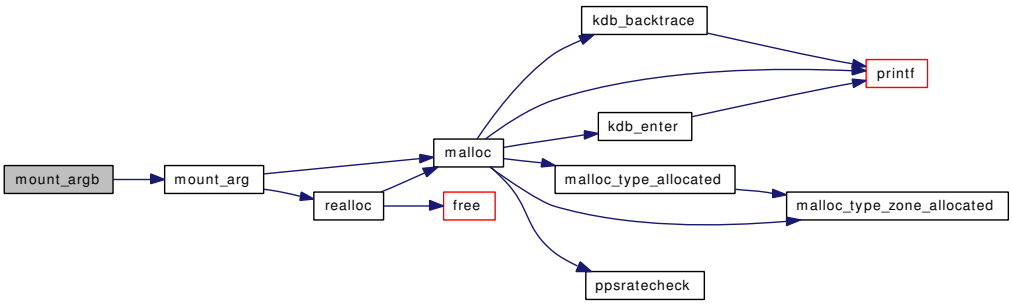
9.161.2.17 struct mntarg* mount_argb (struct mntarg * *ma*, int *flag*, const char * *name*)

Definition at line 2021 of file vfs_mount.c.

References mount_arg().

Referenced by mount().

Here is the call graph for this function:

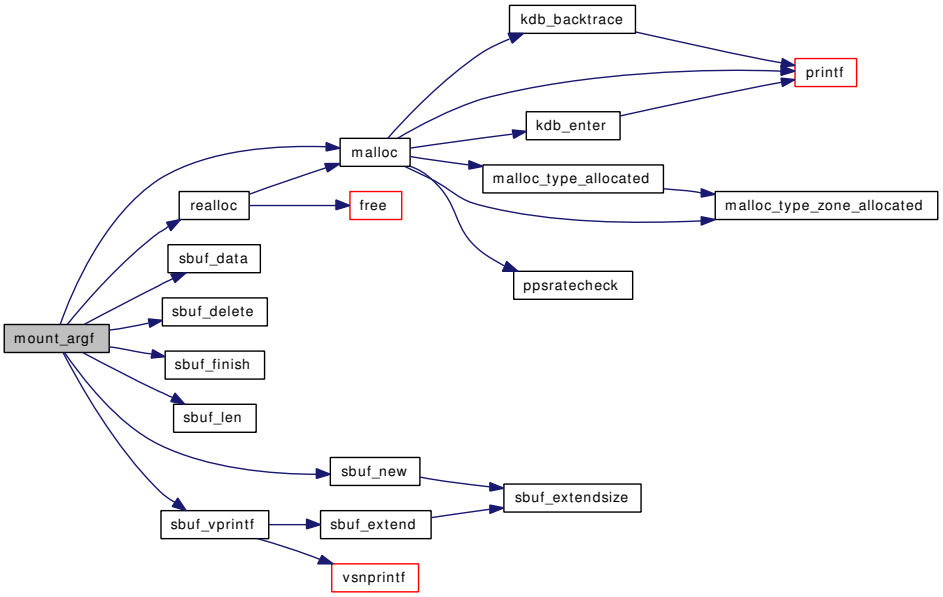


9.161.2.18 struct mntarg* mount_argf (struct mntarg * *ma*, const char * *name*, const char * *fmt*, ...)

Definition at line 2034 of file vfs_mount.c.

References mntarg::error, mntarg::len, malloc(), realloc(), sbuf_data(), sbuf_delete(), sbuf_finish(), sbuf_-len(), sbuf_new(), sbuf_vprintf(), and mntarg::v.

Here is the call graph for this function:



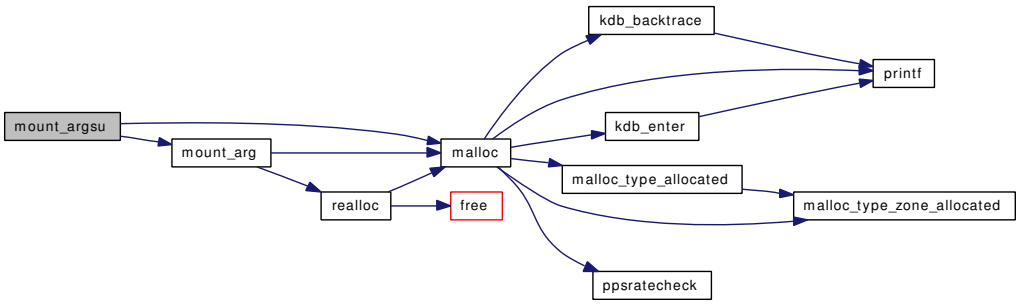
9.161.2.19 struct mntarg* mount_argsu (struct mntarg * *ma*, const char * *name*, const void * *val*, int *len*)

Definition at line 2076 of file vfs_mount.c.

References malloc(), and mount_arg().

Referenced by mount().

Here is the call graph for this function:



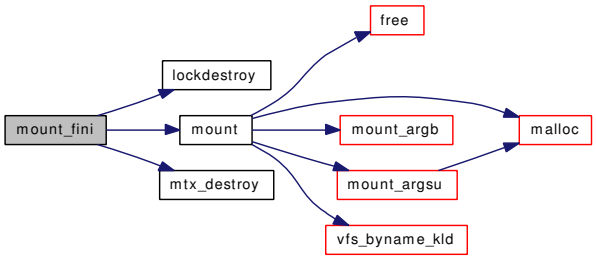
9.161.2.20 static void mount_fini (void * *mem*, int *size*) [static]

Definition at line 458 of file vfs_mount.c.

References lockdestroy(), mount(), and mtx_destroy().

Referenced by vfs_mountroot().

Here is the call graph for this function:



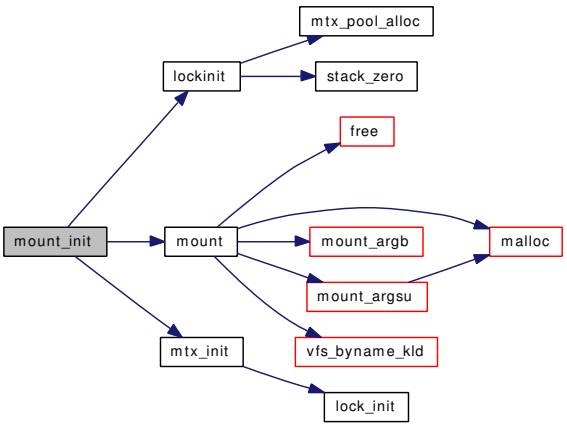
9.161.2.21 static int mount_init (void * *mem*, int *size*, int *flags*) [static]

Definition at line 447 of file vfs_mount.c.

References lockinit(), mount(), and mtx_init().

Referenced by vfs_mountroot().

Here is the call graph for this function:



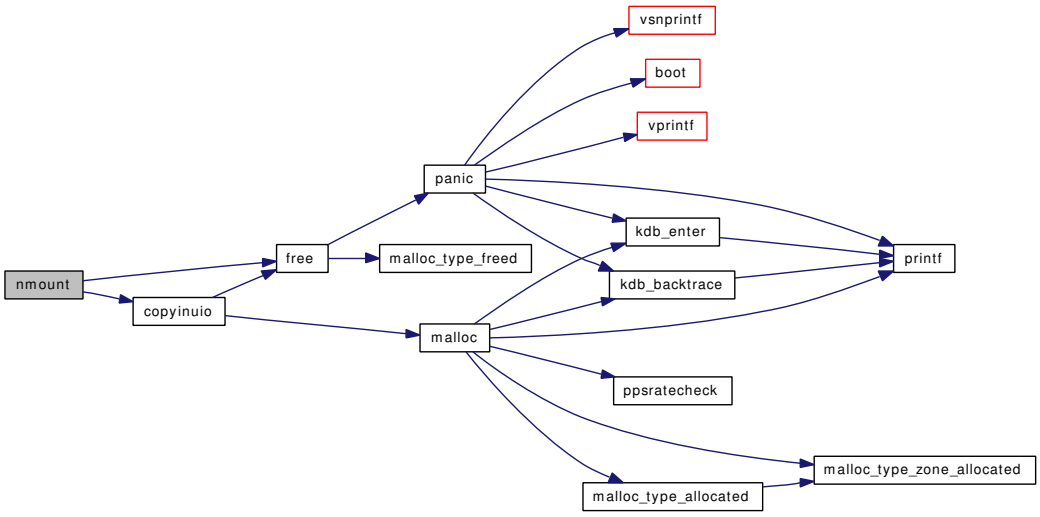
9.161.2.22 MTX_SYSINIT (mountlist, & *mountlist_mtx*, "mountlist", MTX_DEF)

9.161.2.23 int nmount (struct thread * *td*, struct nmount_args * *uap*)

Definition at line 378 of file vfs_mount.c.

References copyinuio(), and free().

Here is the call graph for this function:

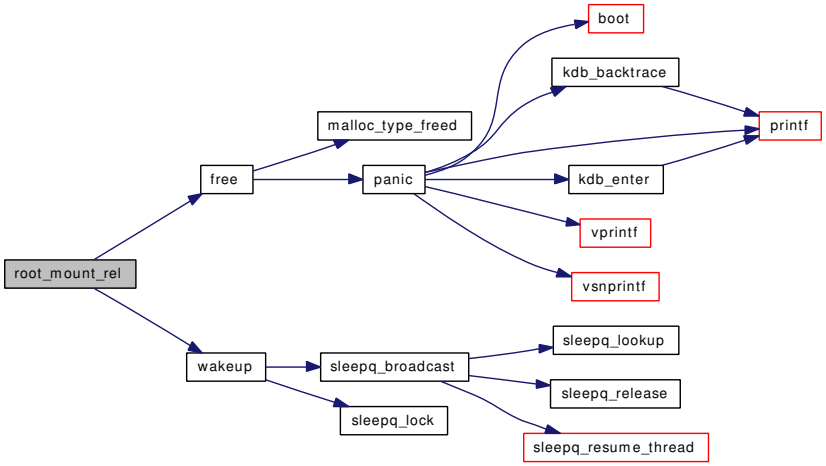


9.161.2.24 void root_mount_rel (struct root_hold_token * *h*)

Definition at line 1334 of file vfs_mount.c.

References free(), mountlist_mtx, and wakeup().

Here is the call graph for this function:



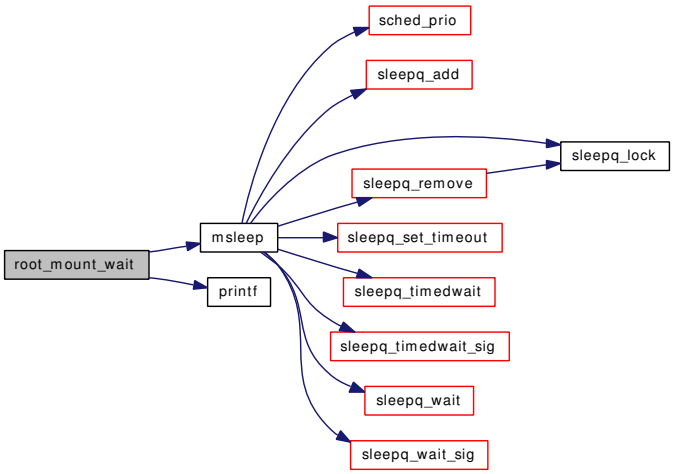
9.161.2.25 static void root_mount_wait (void) [static]

Definition at line 1345 of file vfs_mount.c.

References hz, mountlist_mtx, msleep(), printf(), and root_hold_token::who.

Referenced by vfs_mountroot().

Here is the call graph for this function:



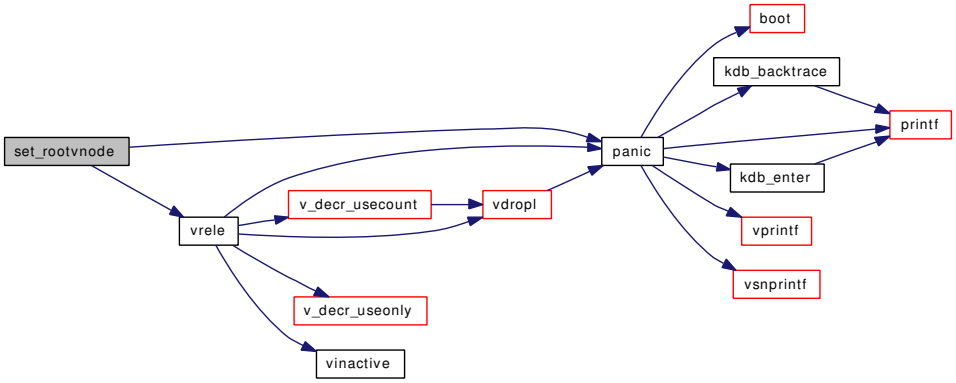
9.161.2.26 static void set_rootvnode (struct thread * *td*) [static]

Definition at line 1368 of file vfs_mount.c.

References mountlist, panic(), rootvnode, and vrele().

Referenced by devfs_first(), and devfs_fixup().

Here is the call graph for this function:



9.161.2.27 SYSCTL_INT (_vfs, OID_AUTO, usermount, CTLFLAG_RW, & *usermount*, 0, "Unprivileged users may mount and unmount file systems")

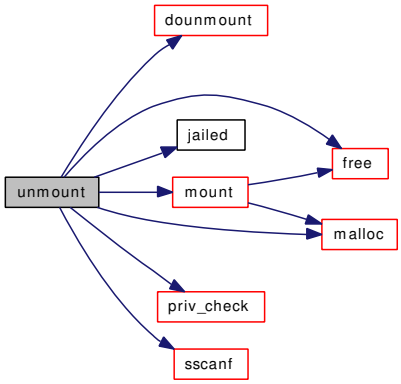
9.161.2.28 TAILQ_HEAD (vfsoptlist, vfsopt)

9.161.2.29 int unmount (struct thread * *td*, struct unmount_args * *uap*)

Definition at line 1089 of file vfs_mount.c.

References dounmount(), free(), Giant, jailed(), malloc(), mount(), mountlist, mountlist_mtx, priv_check(), sscanf(), and usermount.

Here is the call graph for this function:



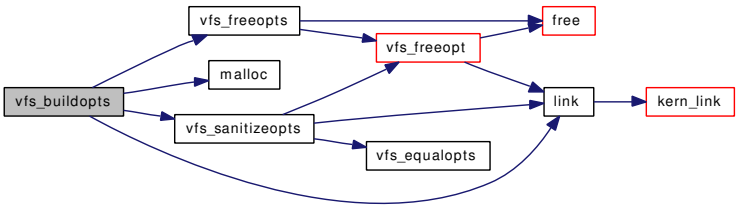
9.161.2.30 static int vfs_buildopts (struct uio * *auio*, struct vfsoptlist ** *options*) [static]

Definition at line 258 of file vfs_mount.c.

References link(), malloc(), vfs_freeopts(), VFS_MOUNTARG_SIZE_MAX, and vfs_sanitizeopts().

Referenced by vfs_donmount().

Here is the call graph for this function:



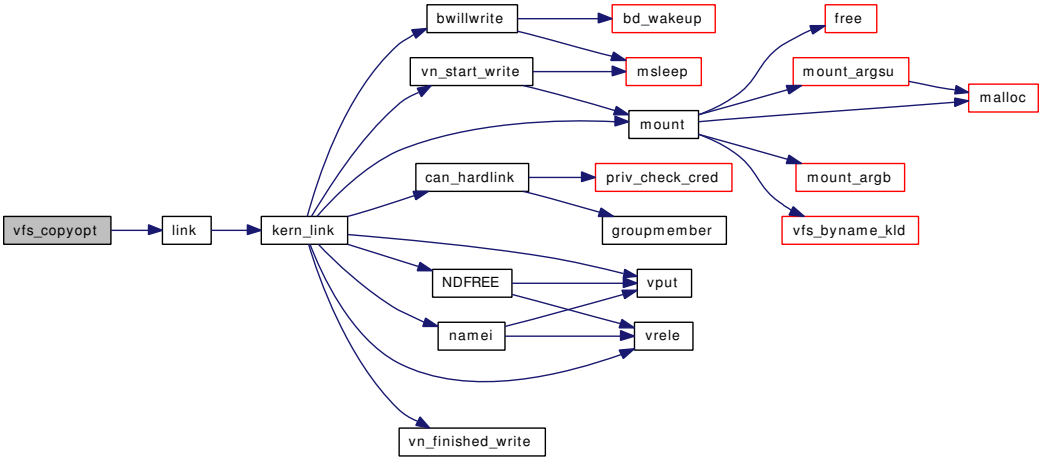
9.161.2.31 int vfs_copyopt (struct vfsoptlist * *opts*, const char * *name*, void * *dest*, int *len*)

Definition at line 1853 of file vfs_mount.c.

References link().

Referenced by vfs_domount().

Here is the call graph for this function:



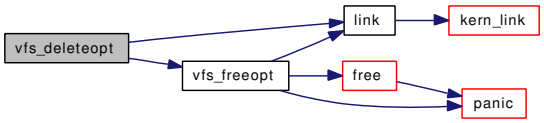
9.161.2.32 void vfs_deleteopt (struct vfsoptlist * *opts*, const char * *name*)

Definition at line 201 of file vfs_mount.c.

References link(), and vfs_freeopt().

Referenced by vfs_export().

Here is the call graph for this function:



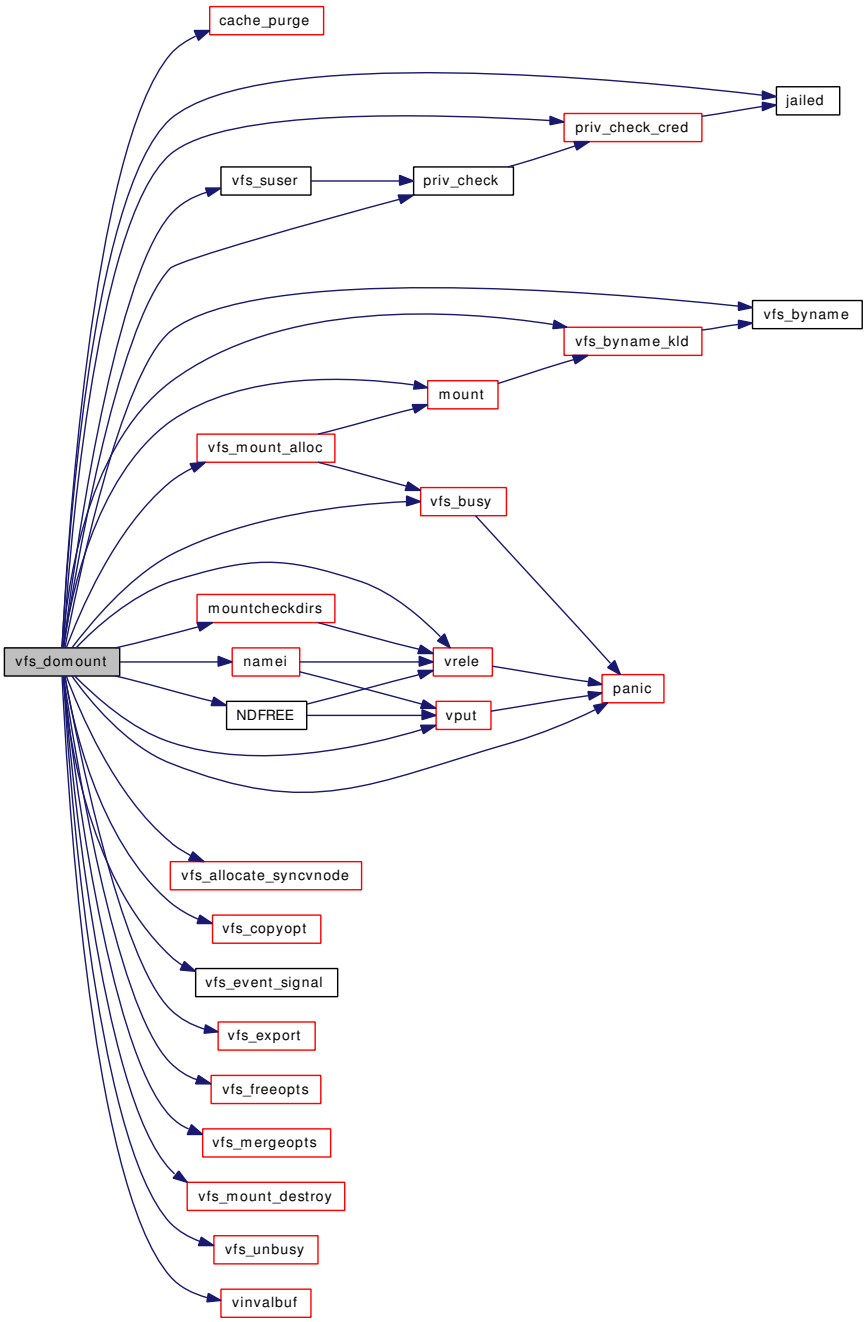
9.161.2.33 static int vfs_domount (struct thread * *td*, const char * *fstype*, char * *fspath*, int *fsflags*, void * *fsdata*) [static]

Definition at line 795 of file vfs_mount.c.

References cache_purge(), Giant, jailed(), mount(), mountcheckdirs(), mountlist, mountlist_mtx, namei(), NDFREE(), panic(), priv_check(), priv_check_cred(), usermount, vfs_allocate_syncvnode(), vfs_busy(), vfs_byname(), vfs_byname_kld(), vfs_copyopt(), vfs_event_signal(), vfs_export(), vfs_freeopts(), vfs_mergeopts(), vfs_mount_alloc(), vfs_mount_destroy(), vfs_suser(), vfs_unbusy(), vfsconf, vinvalbuf(), vput(), and vrele().

Referenced by vfs_donmount().

Here is the call graph for this function:



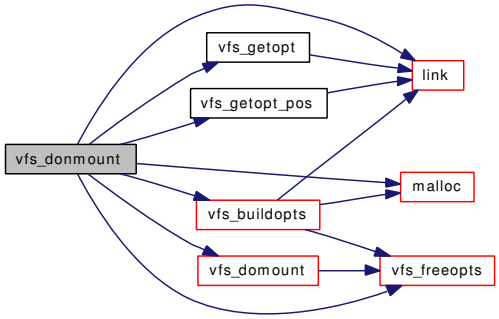
9.161.2.34 static int vfs_donmount (struct thread * *td*, int *fsflags*, struct uio * *fsoptions*) [static]

Definition at line 586 of file vfs_mount.c.

References Giant, link(), malloc(), vfs_buildopts(), vfs_domount(), vfs_freeopts(), vfs_getopt(), and vfs_-getopt_pos().

Referenced by kernel_mount().

Here is the call graph for this function:



9.161.2.35 static int vfs_equalopts (const char * *opt1*, const char * *opt2*) [static]

Definition at line 215 of file vfs_mount.c.

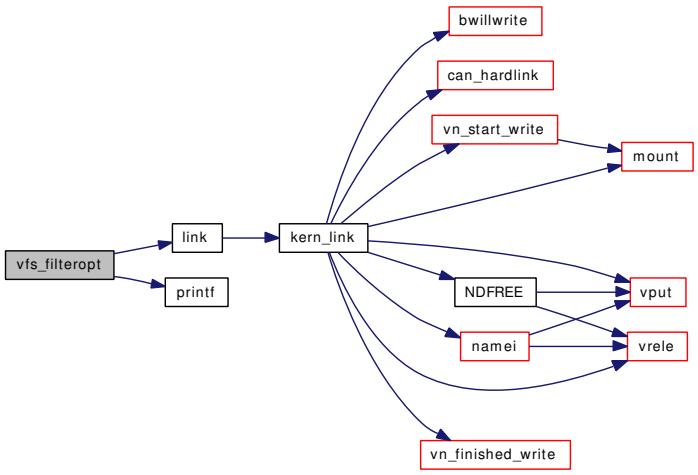
Referenced by vfs_sanitizeopts().

9.161.2.36 int vfs_filteropt (struct vfsoptlist * *opts*, const char ** *legal*)

Definition at line 1711 of file vfs_mount.c.

References global_opts, link(), and printf().

Here is the call graph for this function:

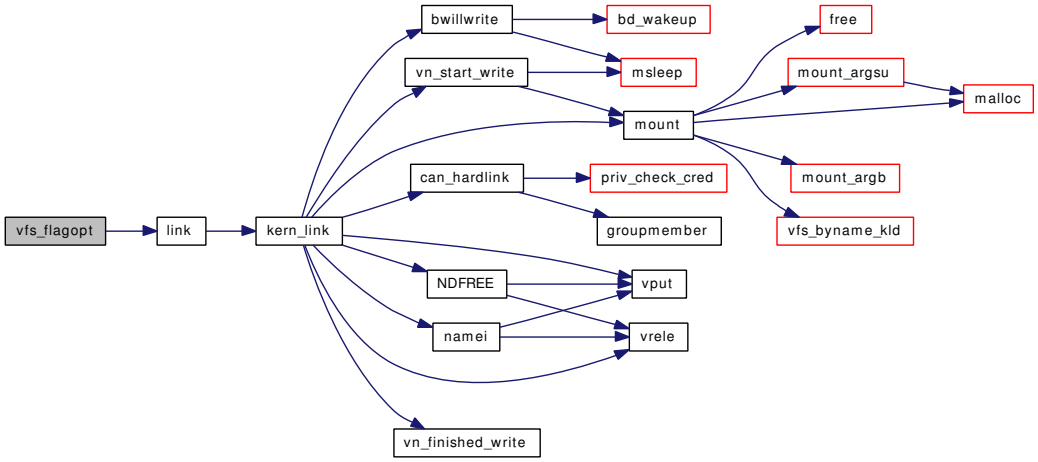


9.161.2.37 int vfs_flagopt (struct vfsoptlist * *opts*, const char * *name*, u_int * *w*, u_int *val*)

Definition at line 1806 of file vfs_mount.c.

References link().

Here is the call graph for this function:



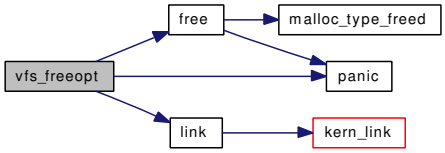
9.161.2.38 static void vfs_freeopt (struct vfsoptlist * *opts*, struct vfsopt * *opt*) [static]

Definition at line 172 of file vfs_mount.c.

References free(), link(), and panic().

Referenced by vfs_deleteopt(), vfs_freeopts(), vfs_mergeopts(), and vfs_sanitizeopts().

Here is the call graph for this function:



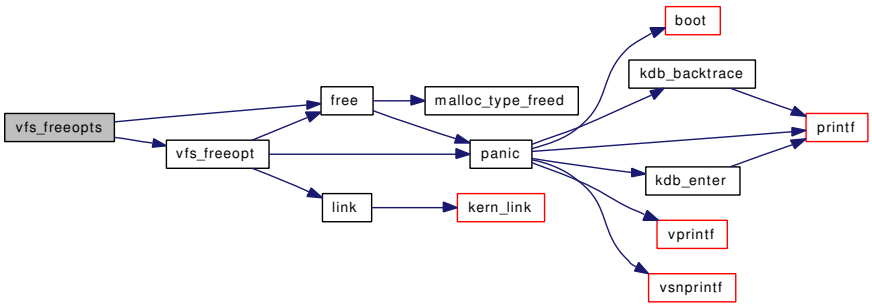
9.161.2.39 static void vfs_freeopts (struct vfsoptlist * *opts*) [static]

Definition at line 189 of file vfs_mount.c.

References free(), and vfs_freeopt().

Referenced by vfs_buildopts(), vfs_domount(), and vfs_donmount().

Here is the call graph for this function:



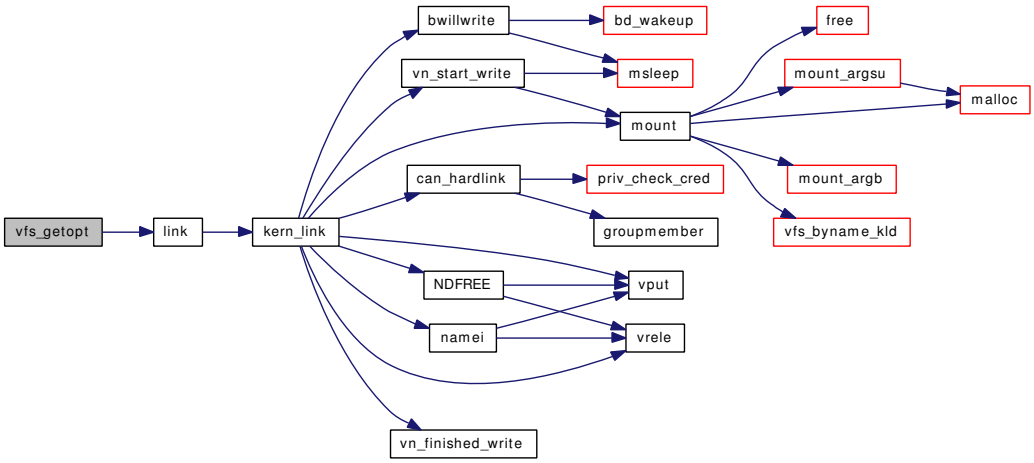
9.161.2.40 int vfs_getopt (struct vfsoptlist * *opts*, const char * *name*, void ** *buf*, int * *len*)

Definition at line 1746 of file vfs_mount.c.

References link().

Referenced by vfs_donmount(), and vfs_mount_error().

Here is the call graph for this function:



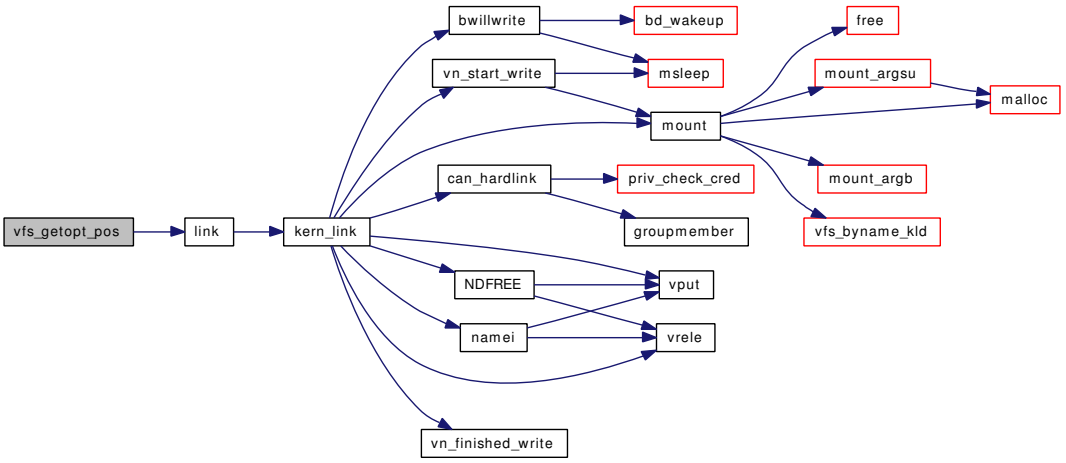
9.161.2.41 static int vfs_getopt_pos (struct vfsoptlist * *opts*, const char * *name*) [static]

Definition at line 1769 of file vfs_mount.c.

References link().

Referenced by vfs_donmount().

Here is the call graph for this function:

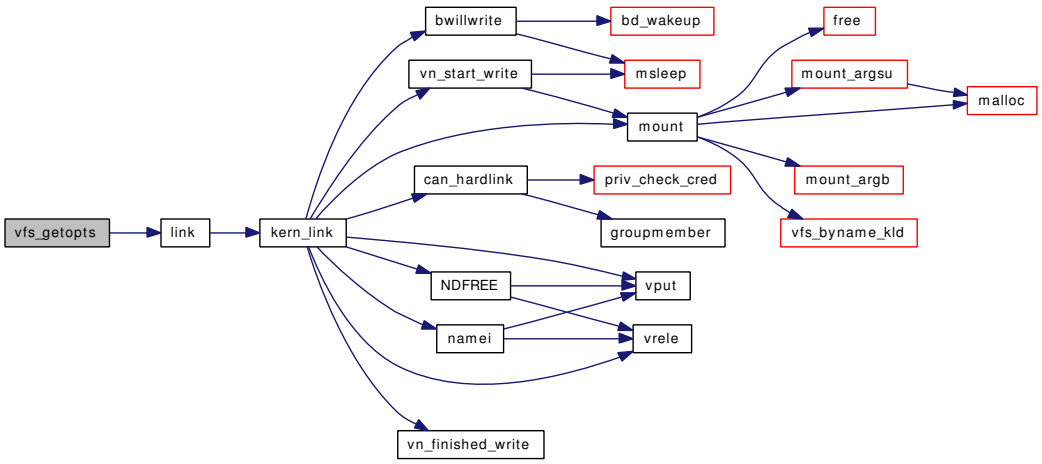


9.161.2.42 char* vfs_getopts (struct vfsoptlist * *opts*, const char * *name*, int * *error*)

Definition at line 1787 of file vfs_mount.c.

References link().

Here is the call graph for this function:



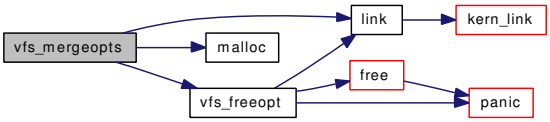
9.161.2.43 static void vfs_mergeopts (struct vfsoptlist * *toopts*, struct vfsoptlist * *opts*) [static]

Definition at line 336 of file vfs_mount.c.

References link(), malloc(), and vfs_freeopt().

Referenced by vfs_domount().

Here is the call graph for this function:



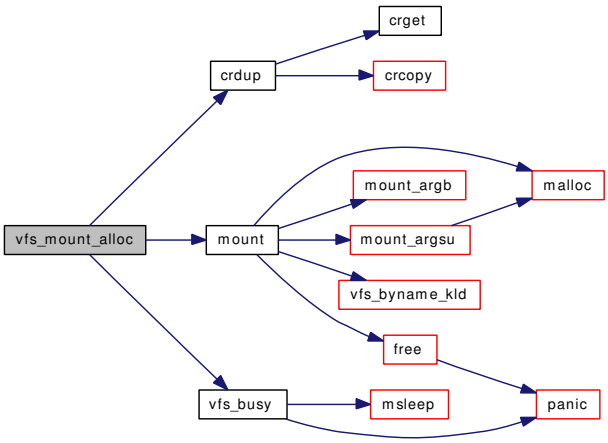
9.161.2.44 static struct mount * vfs_mount_alloc (struct vnode * *dvp*, struct vfsconf * *vfsp*, const char * *fspath*, struct thread * *td*) [static]

Definition at line 471 of file vfs_mount.c.

References crdup(), mount(), mount_zone, and vfs_busy().

Referenced by devfs_first(), and vfs_domount().

Here is the call graph for this function:



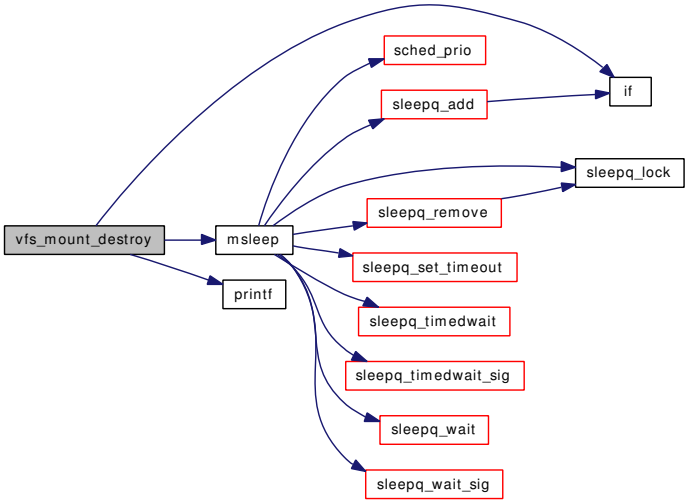
9.161.2.45 static void vfs_mount_destroy (struct mount *) [static]

Definition at line 509 of file vfs_mount.c.

References bootverbose, hz, if(), msleep(), printf(), and rebooting.

Referenced by dounmount(), and vfs_domount().

Here is the call graph for this function:



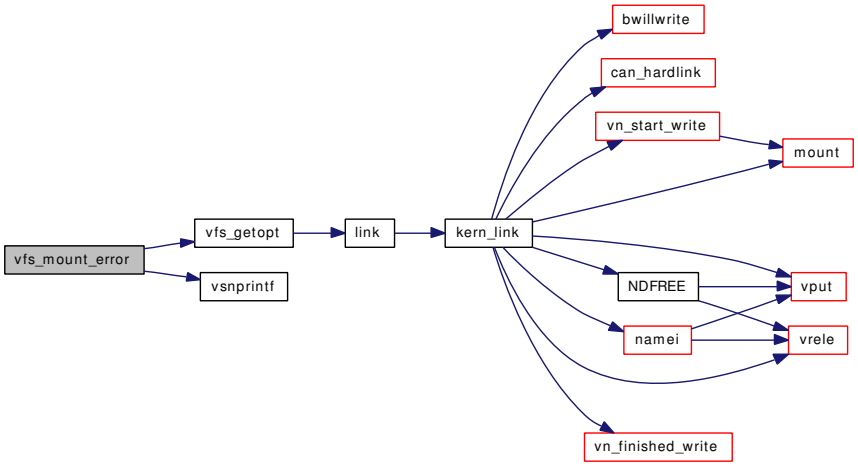
9.161.2.46 void vfs_mount_error (struct mount * *mp*, const char * *fmt*, ...)

Definition at line 1497 of file vfs_mount.c.

References vfs_getopt(), and vsnprintf().

Referenced by vfs_hang_addrlist().

Here is the call graph for this function:



9.161.2.47 void vfs_mountedfrom (struct mount * *mp*, const char * *from*)

Definition at line 1982 of file vfs_mount.c.

Referenced by mqfs_mount().

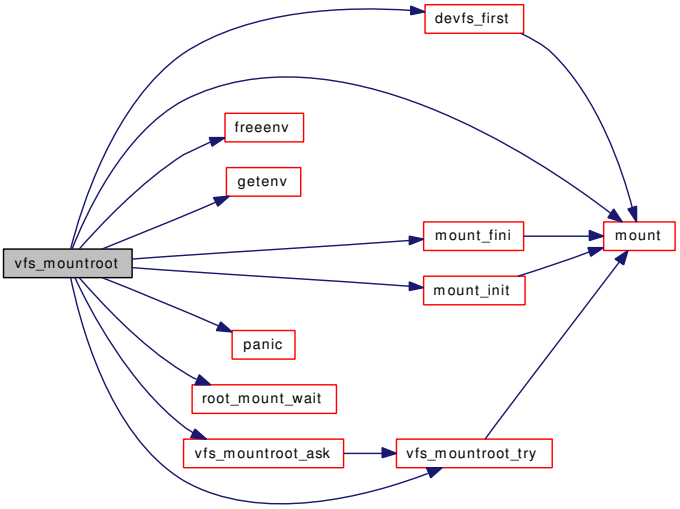
9.161.2.48 void vfs_mountroot (void)

Definition at line 1517 of file vfs_mount.c.

References boothowto, cdrom_rootdevnames, ctrootdevname, devfs_first(), freeenv(), getenv(), mount(), mount_fini(), mount_init(), mount_zone, panic(), root_mount_wait(), rootdevnames, vfs_mountroot_ask(), and vfs_mountroot_try().

Referenced by start_init().

Here is the call graph for this function:



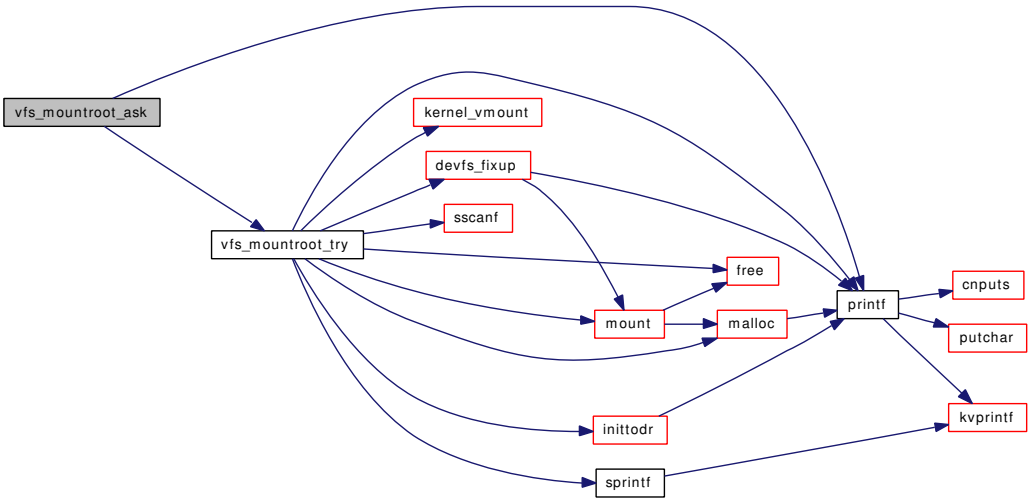
9.161.2.49 static int vfs_mountroot_ask (void) [static]

Definition at line 1674 of file vfs_mount.c.

References printf(), and vfs_mountroot_try().

Referenced by vfs_mountroot().

Here is the call graph for this function:



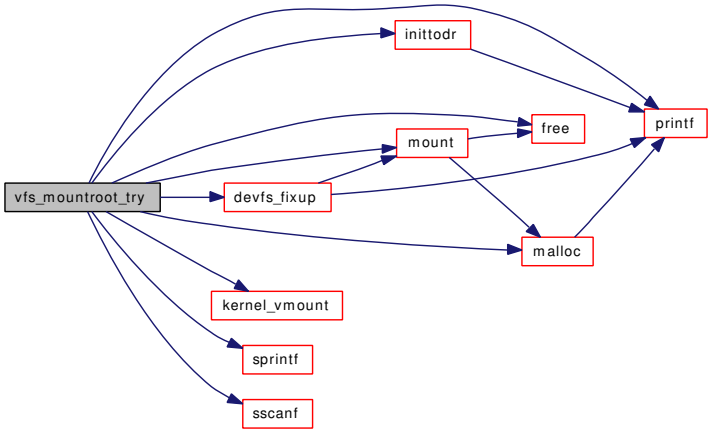
9.161.2.50 static int vfs_mountroot_try (const char * *mountfrom*) [static]

Definition at line 1602 of file vfs_mount.c.

References devfs_fixup(), free(), inittodr(), kernel_vmount(), malloc(), mount(), mountlist, printf(), ROOTNAME, sprintf(), and sscanf().

Referenced by vfs_mountroot(), and vfs_mountroot_ask().

Here is the call graph for this function:



9.161.2.51 void vfs_ref (struct mount * *mp*)

Definition at line 429 of file vfs_mount.c.

Referenced by kern_fstatfs(), kern_statfs(), vfs_getvfs(), and vop_stdgetwritemount().

9.161.2.52 void vfs_rel (struct mount * *mp*)

Definition at line 438 of file vfs_mount.c.

Referenced by fhopen(), fhstat(), kern_fhstatfs(), kern_fstatfs(), kern_statfs(), sysctl_vfs_ctl(), vfs_getnewfsid(), vfs_unbusy(), vn_start_secondary_write(), vn_write_suspend_wait(), and vop_stdgetwritemount().

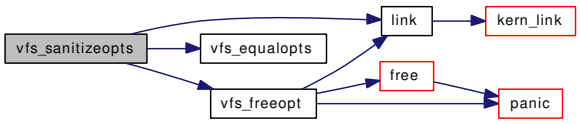
9.161.2.53 static void vfs_sanitizeopts (struct vfsoptlist * *opts*) [static]

Definition at line 236 of file vfs_mount.c.

References link(), vfs_equalopts(), and vfs_freeopt().

Referenced by vfs_buildopts().

Here is the call graph for this function:

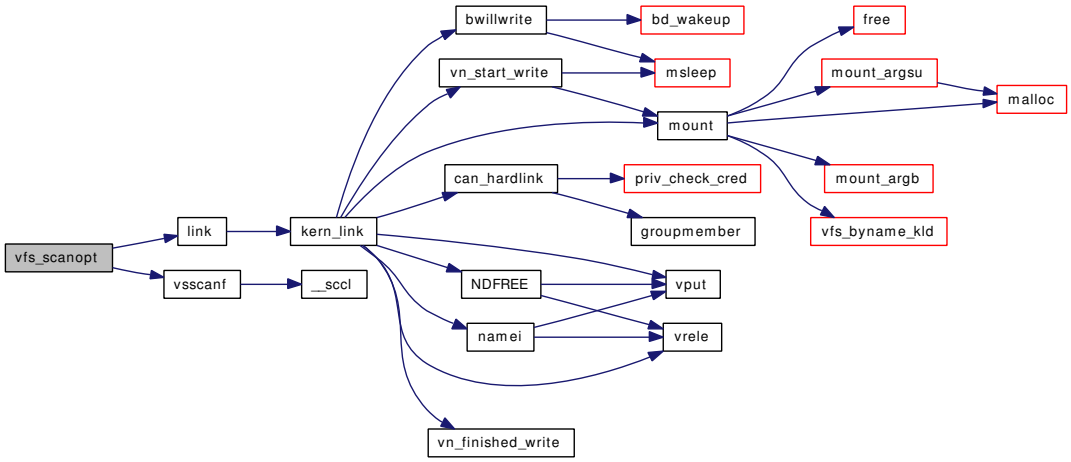


9.161.2.54 int vfs_scanopt (struct vfsoptlist * *opts*, const char * *name*, const char * *fmt*, ...)

Definition at line 1823 of file vfs_mount.c.

References link(), ret, and vsscanf().

Here is the call graph for this function:



9.161.3 Variable Documentation

9.161.3.1 char* cdrom_rootdevnames[ ] [static]

Initial value:

```
{
        "cd9660:cd0",
        "cd9660:acd0",
        NULL
}
```

Definition at line 152 of file vfs_mount.c.

Referenced by vfs_mountroot().

9.161.3.2 const char* ctrootdevname = ROOTDEVNAME [static]

Definition at line 163 of file vfs_mount.c.

Referenced by vfs_mountroot().

9.161.3.3 const char* global_opts[ ] [static]

Initial value:

```
{
        "errmsg",
        "fstype",
        "fspath",
        "rdonly",
        "ro",
        "rw",
        "suid",
        "exec",
        "update",
        NULL
}
```

Definition at line 136 of file vfs_mount.c.

Referenced by vfs_filteropt().

9.161.3.4 uma_zone_t mount_zone [static]

Definition at line 99 of file vfs_mount.c.

Referenced by vfs_mount_alloc(), and vfs_mountroot().

9.161.3.5 struct mntlist mountlist = TAILQ_HEAD_INITIALIZER(mountlist)

Definition at line 102 of file vfs_mount.c.

Referenced by devfs_first(), devfs_fixup(), dounmount(), kern_getfsstat(), set_rootvnode(), sync(), unmount(), vfs_domount(), vfs_getvfs(), vfs_mountroot_try(), vfs_unmountall(), and vnlru_proc().

9.161.3.6 struct mtx mountlist_mtx

Definition at line 105 of file vfs_mount.c.

Referenced by devfs_first(), devfs_fixup(), dounmount(), kern_getfsstat(), LIST_HEAD(), root_mount_rel(), root_mount_wait(), sync(), sync_fsync(), unmount(), vfs_domount(), vfs_getvfs(), and vnlru_proc().

9.161.3.7 char* rootdevnames[2] = {NULL, NULL}

Definition at line 159 of file vfs_mount.c.

Referenced by vfs_mountroot().

9.161.3.8 struct vnode* rootvnode

Definition at line 120 of file vfs_mount.c.

Referenced by change_root(), devfs_fixup(), dounmount(), linker_load_module(), lookup(), mountcheckdirs(), set_rootvnode(), and vn_fullpath1().

9.161.3.9 int usermount = 0 [static]

Definition at line 93 of file vfs_mount.c.

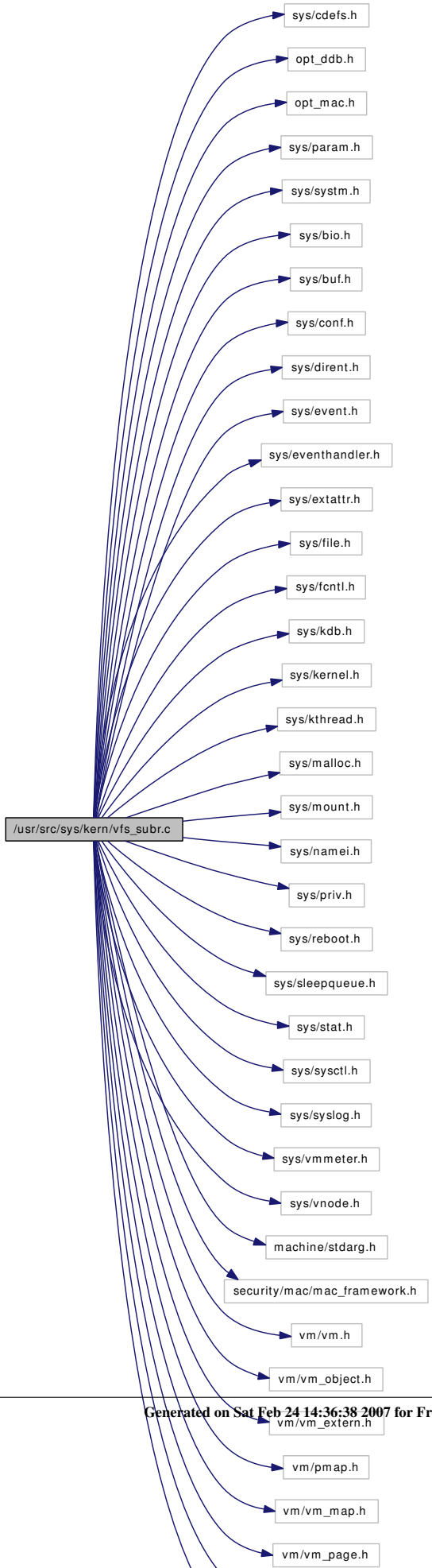
Referenced by unmount(), and vfs_domount().

9.162 /usr/src/sys/kern/vfs_subr.c File Reference

```
#include <sys/cdefs.h>
#include "opt_ddb.h"
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/bio.h>
#include <sys/buf.h>
#include <sys/conf.h>
#include <sys/dirent.h>
#include <sys/event.h>
#include <sys/eventhandler.h>
#include <sys/extattr.h>
#include <sys/file.h>
#include <sys/fcntl.h>
#include <sys/kdb.h>
#include <sys/kernel.h>
#include <sys/kthread.h>
#include <sys/malloc.h>
#include <sys/mount.h>
#include <sys/namei.h>
#include <sys/priv.h>
#include <sys/reboot.h>
#include <sys/sleepqueue.h>
#include <sys/stat.h>
#include <sys/sysctl.h>
#include <sys/syslog.h>
#include <sys/vmmeter.h>
#include <sys/vnode.h>
#include <machine/stdarg.h>
#include <security/mac/mac_framework.h>
#include <vm/vm.h>
#include <vm/vm_object.h>
#include <vm/vm_extern.h>
#include <vm/pmap.h>
#include <vm/vm_map.h>
```

```
#include <vm/vm_page.h>
#include <vm/vm_kern.h>
#include <vm/uma.h>
```

Include dependency graph for vfs_subr.c:


/usr/src/sys/kern/vfs_subr.c
sys/cdefs.h
opt_ddb.h
opt_mac.h
sys/param.h
sys/systm.h
sys/bio.h
sys/buf.h
sys/conf.h
sys/dirent.h
sys/event.h
sys/eventhandler.h
sys/extattr.h
sys/file.h
sys/fcntl.h
sys/kdb.h
sys/kernel.h
sys/kthread.h
sys/malloc.h
sys/mount.h
sys/namei.h
sys/priv.h
sys/reboot.h
sys/sleepqueue.h
sys/stat.h
sys/sysctl.h
sys/syslog.h
sys/vmmeter.h
sys/vnode.h
machine/stdarg.h
security/mac/mac_framework.h
vm/vm.h
vm/vm_object.h
vm/vm_extern.h
vm/pmap.h
vm/vm_map.h
vm/vm_page.h

Defines

- #define SYNCER_MAXDELAY 32
- #define SYNCER_SHUTDOWN_SPEEDUP 4
- #define VCANRECYCLE(vp) (((vp) → v_iflag & VI_FREE) && !(vp) → v_holdcnt)
- #define VSHOULDFREE(vp) (!((vp) → v_iflag & VI_FREE) && !(vp) → v_holdcnt)
- #define VSHOULDBUSY(vp) (((vp) → v_iflag & VI_FREE) && (vp) → v_holdcnt)
- #define MAXVNODES_MAX 100000
- #define KINFO_VNODESLOP 10
- #define sync_close ((int (∗)(struct vop_close_args ∗))nullop)

Enumerations

- enum { TSP_SEC, TSP_HZ, TSP_USEC, TSP_NSEC }

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/vfs_subr.c,v 1.692 2006/11/13 05:51:22 kmacy Exp $")
- static MALLOC_DEFINE (M_NETADDR,"subr_export_host","Export host address structure")
- static void delmntque (struct vnode ∗vp)
- static void insmntque (struct vnode ∗vp, struct mount ∗mp)
- static int flushbuflist (struct bufv ∗bufv, int flags, struct bufobj ∗bo, int slpflag, int slptimeo)
- static void syncer_shutdown (void ∗arg, int howto)
- static int vtryrecycle (struct vnode ∗vp)
- static void vbusy (struct vnode ∗vp)
- static void vdropl (struct vnode ∗vp)
- static void vinactive (struct vnode ∗, struct thread ∗)
- static void v_incr_usecount (struct vnode ∗)
- static void v_decr_usecount (struct vnode ∗)
- static void v_decr_useonly (struct vnode ∗)
- static void v_upgrade_usecount (struct vnode ∗)
- static void vfree (struct vnode ∗)
- static void vnlru_free (int)
- static void vdestroy (struct vnode ∗)
- static void vgonel (struct vnode ∗)
- static void vfs_knllock (void ∗arg)
- static void vfs_knlunlock (void ∗arg)
- static int vfs_knllocked (void ∗arg)
- TUNABLE_INT ("debug.mpsafevfs",&mpsafe_vfs)
- SYSCTL_INT (_debug, OID_AUTO, mpsafevfs, CTLFLAG_RD,&mpsafe_vfs, 0,"MPSAFE VFS")
- SYSCTL_LONG (_vfs, OID_AUTO, numvnodes, CTLFLAG_RD,&numvnodes, 0,"")
- static TAILQ_HEAD (freelst, vnode)
- SYSCTL_INT (_kern, KERN_MAXVNODES, maxvnodes, CTLFLAG_RW,&desiredvnodes, 0,"Maximum number of vnodes")
- SYSCTL_INT (_kern, OID_AUTO, minvnodes, CTLFLAG_RW,&wantfreevnodes, 0,"Minimum number of vnodes (legacy)")
- SYSCTL_INT (_debug, OID_AUTO, vnlru_nowhere, CTLFLAG_RW,&vnlru_nowhere, 0,"Number of times the vnlru process ran without success")
- static void vntblinit (void ∗dummy __unused)

- int vfs_busy (struct mount *mp, int flags, struct mtx *interlkp, struct thread *td)
- void vfs_unbusy (struct mount *mp, struct thread *td)
- mount * vfs_getvfs (fsid_t *fsid)
- int vfs_suser (struct mount *mp, struct thread *td)
- void vfs_getnewfsid (struct mount *mp)
- SYSCTL_INT (_vfs, OID_AUTO, timestamp_precision, CTLFLAG_RW,&timestamp_precision, 0,"")
- void vfs_timestamp (struct timespec *tsp)
- void vattr_null (struct vattr *vap)
- static int vlrureclaim (struct mount *mp)
- static void vnlru_proc (void)
- int getnewvnode (const char *tag, struct mount *mp, struct vop_vector *vops, struct vnode **vpp)
- int bufobj_invalbuf (struct bufobj *bo, int flags, struct thread *td, int slpflag, int slptimeo)
- int vinvalbuf (struct vnode *vp, int flags, struct thread *td, int slpflag, int slptimeo)
- int vtruncbuf (struct vnode *vp, struct ucred *cred, struct thread *td, off_t length, int blksize)
- static struct buf * buf_splay (daddr_t lblkno, b_xflags_t xflags, struct buf *root)
- static void buf_vlist_remove (struct buf *bp)
- static void buf_vlist_add (struct buf *bp, struct bufobj *bo, b_xflags_t xflags)
- buf * gbincore (struct bufobj *bo, daddr_t lblkno)
- void bgetvp (struct vnode *vp, struct buf *bp)
- void brelvp (struct buf *bp)
- static void vn_syncer_add_to_worklist (struct bufobj *bo, int delay)
- static int sysctl_vfs_worklist_len (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_vfs, OID_AUTO, worklist_len, CTLTYPE_INT|CTLFLAG_RD, NULL, 0, sysctl_vfs_worklist_len,"I","Syncer thread worklist length")
- static void sched_sync (void)
- static int sync_vnode (struct bufobj *bo, struct thread *td)
- int speedup_syncer (void)
- void reassignbuf (struct buf *bp)
- int vget (struct vnode *vp, int flags, struct thread *td)
- void vref (struct vnode *vp)
- int vrefcnt (struct vnode *vp)
- void vrele (struct vnode *vp)
- void vput (struct vnode *vp)
- void vhold (struct vnode *vp)
- void vholdl (struct vnode *vp)
- void vdrop (struct vnode *vp)
- int vflush (struct mount *mp, int rootrefs, int flags, struct thread *td)
- int vrecycle (struct vnode *vp, struct thread *td)
- void vgone (struct vnode *vp)
- int vcount (struct vnode *vp)
- int count_dev (struct cdev *dev)
- void vn_printf (struct vnode *vp, const char *fmt,...)
- static void vfsconf2x (struct vfsconf *vfsp, struct xvfsconf *xvfsp)
- static int sysctl_vfs_conflist (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_vfs, OID_AUTO, conflist, CTLFLAG_RD, NULL, 0, sysctl_vfs_conflist,"S,xvfsconf","List of all configured filesystems")
- static int sysctl_ovfs_conf (SYSCTL_HANDLER_ARGS)
- static int vfs_sysctl (SYSCTL_HANDLER_ARGS)

- static SYSCTL_NODE (_vfs, VFS_GENERIC, generic, CTLFLAG_RD|CTLFLAG_SKIP, vfs_-sysctl,"Generic filesystem")
- void vfs_unmountall (void)
- void vfs_msync (struct mount *mp, int flags)
- void v_addpollinfo (struct vnode *vp)
- int vn_pollrecord (struct vnode *vp, struct thread *td, int events)
- static int sync_fsync (struct vop_fsync_args *)
- static int sync_inactive (struct vop_inactive_args *)
- static int sync_reclaim (struct vop_reclaim_args *)
- int vfs_allocate_syncvnode (struct mount *mp)
- int vn_isdisk (struct vnode *vp, int *errp)
- int vaccess (enum vtype type, mode_t file_mode, uid_t file_uid, gid_t file_gid, mode_t acc_mode, struct ucred *cred, int *privused)
- int extattr_check_cred (struct vnode *vp, int attrnamespace, struct ucred *cred, struct thread *td, int access)
- void vop_rename_pre (void *ap)
- void vop_strategy_pre (void *ap)
- void vop_lookup_pre (void *ap)
- void vop_lookup_post (void *ap, int rc)
- void vop_lock_pre (void *ap)
- void vop_lock_post (void *ap, int rc)
- void vop_unlock_pre (void *ap)
- void vop_unlock_post (void *ap, int rc)
- void vop_create_post (void *ap, int rc)
- void vop_link_post (void *ap, int rc)
- void vop_mkdir_post (void *ap, int rc)
- void vop_mknod_post (void *ap, int rc)
- void vop_remove_post (void *ap, int rc)
- void vop_rename_post (void *ap, int rc)
- void vop_rmdir_post (void *ap, int rc)
- void vop_setattr_post (void *ap, int rc)
- void vop_symlink_post (void *ap, int rc)
- static void vfs_event_init (void *arg)
- SYSINIT (vfs_knlist, SI_SUB_VFS, SI_ORDER_ANY, vfs_event_init, NULL)
- void vfs_event_signal (fsid_t *fsid, u_int32_t event, intptr_t data __unused)
- static int filt_fsattach (struct knote *kn)
- static void filt_fsdetach (struct knote *kn)
- static int filt_fsevent (struct knote *kn, long hint)
- static int sysctl_vfs_ctl (SYSCTL_HANDLER_ARGS)
- SYSCTL_PROC (_vfs, OID_AUTO, ctl, CTLFLAG_WR, NULL, 0, sysctl_vfs_ctl,"","Sysctl by fsid")
- u_quad_t init_va_filerev (void)
- static int filt_vfsread (struct knote *kn, long hint)
- static int filt_vfswrite (struct knote *kn, long hint)
- static int filt_vfsvnode (struct knote *kn, long hint)
- static void filt_vfsdetach (struct knote *kn)
- int vfs_kqfilter (struct vop_kqfilter_args *ap)
- int vfs_read_dirent (struct vop_readdir_args *ap, struct dirent *dp, off_t off)
- void vfs_mark_atime (struct vnode *vp, struct thread *td)

Variables

- int mpsafe_vfs = 1
- static unsigned long numvnodes
- enum vtype iftovt_tab [16]
- int vttoif_tab [10]
- syncer_state
- int desiredvnodes
- static int vnlru_nowhere
- static int timestamp_precision = TSP_SEC
- static struct proc * vnlruproc
- static int vnlruproc_sig
- static struct kproc_desc vnlru_kp
- static struct proc * updateproc
- static struct kproc_desc up_kp
- static char * typename [ ]
- static struct vop_vector sync_vnodeops
- static struct knlist fs_knlist
- filterops fs_filtops
- static struct filterops vfsread_filtops
- static struct filterops vfswrite_filtops
- static struct filterops vfsvnode_filtops

9.162.1 Define Documentation

9.162.1.1 #define KINFO_VNODESLOP 10

Definition at line 2703 of file vfs_subr.c.

9.162.1.2 #define MAXVNODES_MAX 100000

Definition at line 292 of file vfs_subr.c.

Referenced by vntblinit().

9.162.1.3 #define sync_close ((int (*)(struct vop_close_args *))nullop)

Definition at line 2986 of file vfs_subr.c.

9.162.1.4 #define SYNCER_MAXDELAY 32

Referenced by TAILQ_HEAD().

9.162.1.5 #define SYNCER_SHUTDOWN_SPEEDUP 4

Referenced by sched_sync().

9.162.1.6 #define VCANRECYCLE(vp) (((vp) → v_iflag & VI_FREE) && !(vp) → v_holdcnt)

Definition at line 283 of file vfs_subr.c.

Referenced by vnlru_free().

9.162.1.7 #define VSHOULDBUSY(vp) (((vp) → v_iflag & VI_FREE) && (vp) → v_holdcnt)

Definition at line 285 of file vfs_subr.c.

Referenced by vholdl().

9.162.1.8 #define VSHOULDFREE(vp) (!((vp) → v_iflag & VI_FREE) && !(vp) → v_holdcnt)

Definition at line 284 of file vfs_subr.c.

Referenced by vfree().

9.162.2 Enumeration Type Documentation

9.162.2.1 anonymous enum

Enumerator:

TSP_SEC

TSP_HZ

TSP_USEC

TSP_NSEC

Definition at line 467 of file vfs_subr.c.

9.162.3 Function Documentation

9.162.3.1 __FBSDID ("$FreeBSD: src/sys/kern/vfs_subr. *c*, v 1.692 2006/11/13 05:51:22 kmacy Exp $")

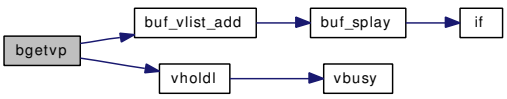
9.162.3.2 void bgetvp (struct vnode ∗ *vp*, struct buf ∗ *bp*)

Definition at line 1454 of file vfs_subr.c.

References buf_vlist_add(), and vholdl().

Referenced by getblk().

Here is the call graph for this function:



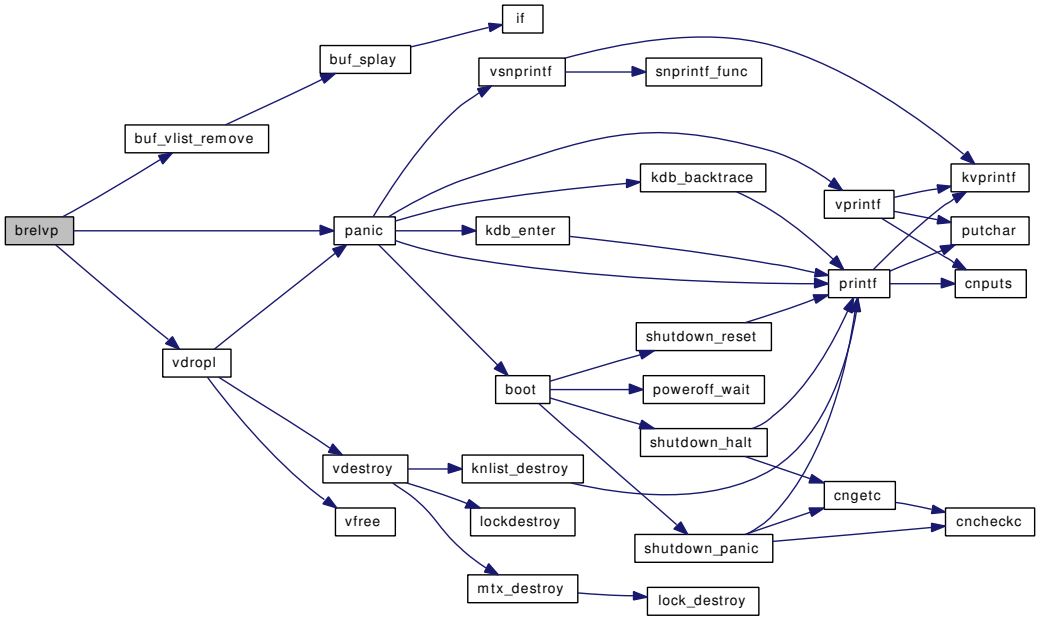
9.162.3.3 void brelvp (struct buf * *bp*)

Definition at line 1480 of file vfs_subr.c.

References buf_vlist_remove(), panic(), and vdropl().

Referenced by brelse(), and getnewbuf().

Here is the call graph for this function:



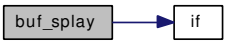
9.162.3.4 static struct buf* buf_splay (daddr_t *lblkno*, b_xflags_t *xflags*, struct buf * *root*) [static]

Definition at line 1285 of file vfs_subr.c.

References buf, and if().

Referenced by buf_vlist_add(), buf_vlist_remove(), and gbincore().

Here is the call graph for this function:



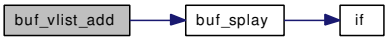
9.162.3.5 static void buf_vlist_add (struct buf * *bp*, struct bufobj * *bo*, b_xflags_t *xflags*) [static]

Definition at line 1377 of file vfs_subr.c.

References buf, and buf_splay().

Referenced by bgetvp(), and reassignbuf().

Here is the call graph for this function:



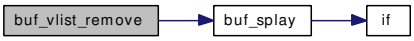
9.162.3.6 static void buf_vlist_remove (struct buf * *bp*) [static]

Definition at line 1340 of file vfs_subr.c.

References buf, and buf_splay().

Referenced by brelvp(), and reassignbuf().

Here is the call graph for this function:



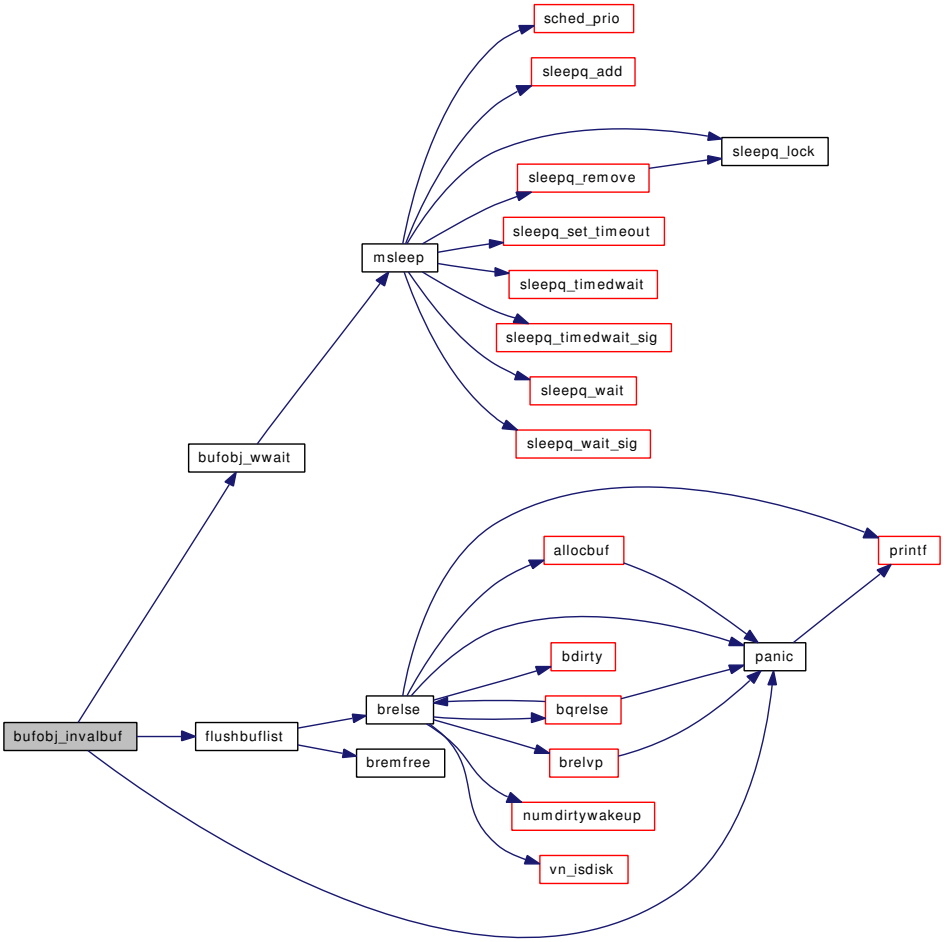
9.162.3.7 int bufobj_invalbuf (struct bufobj * *bo*, int *flags*, struct thread * *td*, int *slpflag*, int *slptimeo*)

Definition at line 1001 of file vfs_subr.c.

References bufobj_wwait(), flushbuflist(), and panic().

Referenced by vinvalbuf().

Here is the call graph for this function:

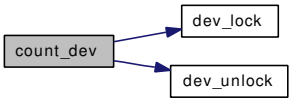


9.162.3.8 int count_dev (struct cdev * *dev*)

Definition at line 2492 of file vfs_subr.c.

References dev_lock(), and dev_unlock().

Here is the call graph for this function:



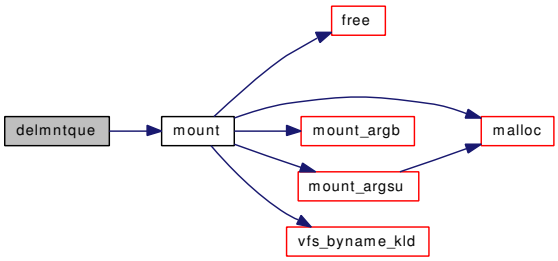
9.162.3.9 static void delmntque (struct vnode * *vp*) [static]

Definition at line 961 of file vfs_subr.c.

References mount().

Referenced by vgonel().

Here is the call graph for this function:

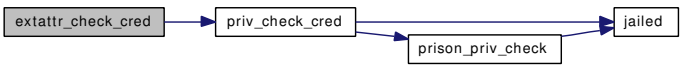


9.162.3.10 int extattr_check_cred (struct vnode * *vp*, int *attrnamespace*, struct ucred * *cred*, struct thread * *td*, int *access*)

Definition at line 3281 of file vfs_subr.c.

References priv_check_cred().

Here is the call graph for this function:



9.162.3.11 static int filt_fsattach (struct knote * *kn*) [static]

Definition at line 3669 of file vfs_subr.c.

References knlist_add().

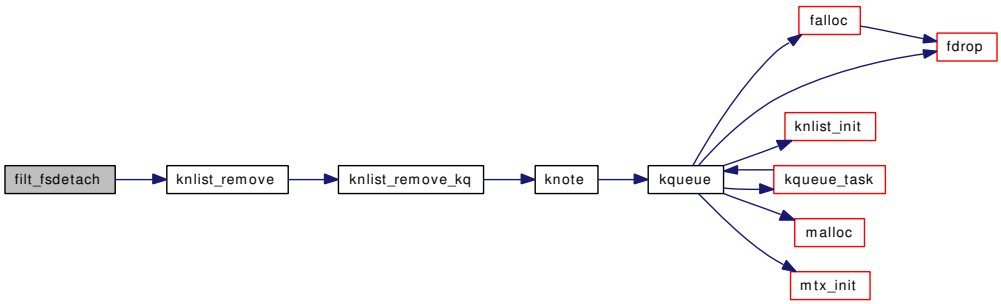
Here is the call graph for this function:

9.162.3.12 static void filt_fsdetach (struct knote * *kn*) [static]

Definition at line 3678 of file vfs_subr.c.

References knlist_remove().

Here is the call graph for this function:



9.162.3.13 static int filt_fsevent (struct knote * *kn*, long *hint*) [static]

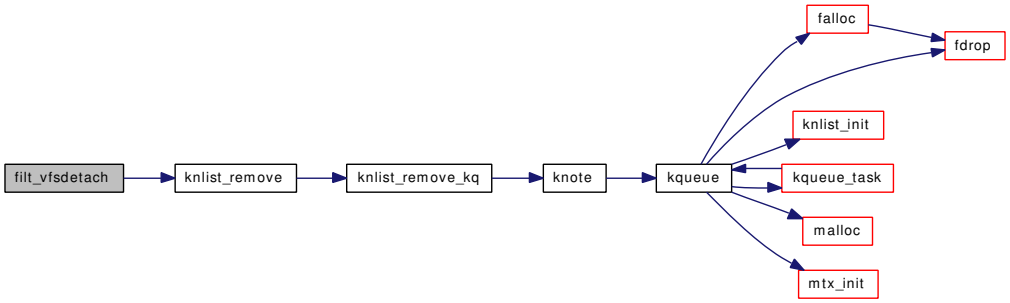
Definition at line 3685 of file vfs_subr.c.

9.162.3.14 static void filt_vfsdetach (struct knote * *kn*) [static]

Definition at line 3807 of file vfs_subr.c.

References knlist_remove().

Here is the call graph for this function:

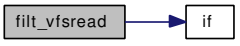


9.162.3.15 static int filt_vfsread (struct knote * *kn*, long *hint*) [static]

Definition at line 3817 of file vfs_subr.c.

References if().

Here is the call graph for this function:



9.162.3.16 static int filt_vfsvnode (struct knote * *kn*, long *hint*) [static]

Definition at line 3854 of file vfs_subr.c.

9.162.3.17 static int filt_vfswrite (struct knote * *kn*, long *hint*) [static]

Definition at line 3840 of file vfs_subr.c.

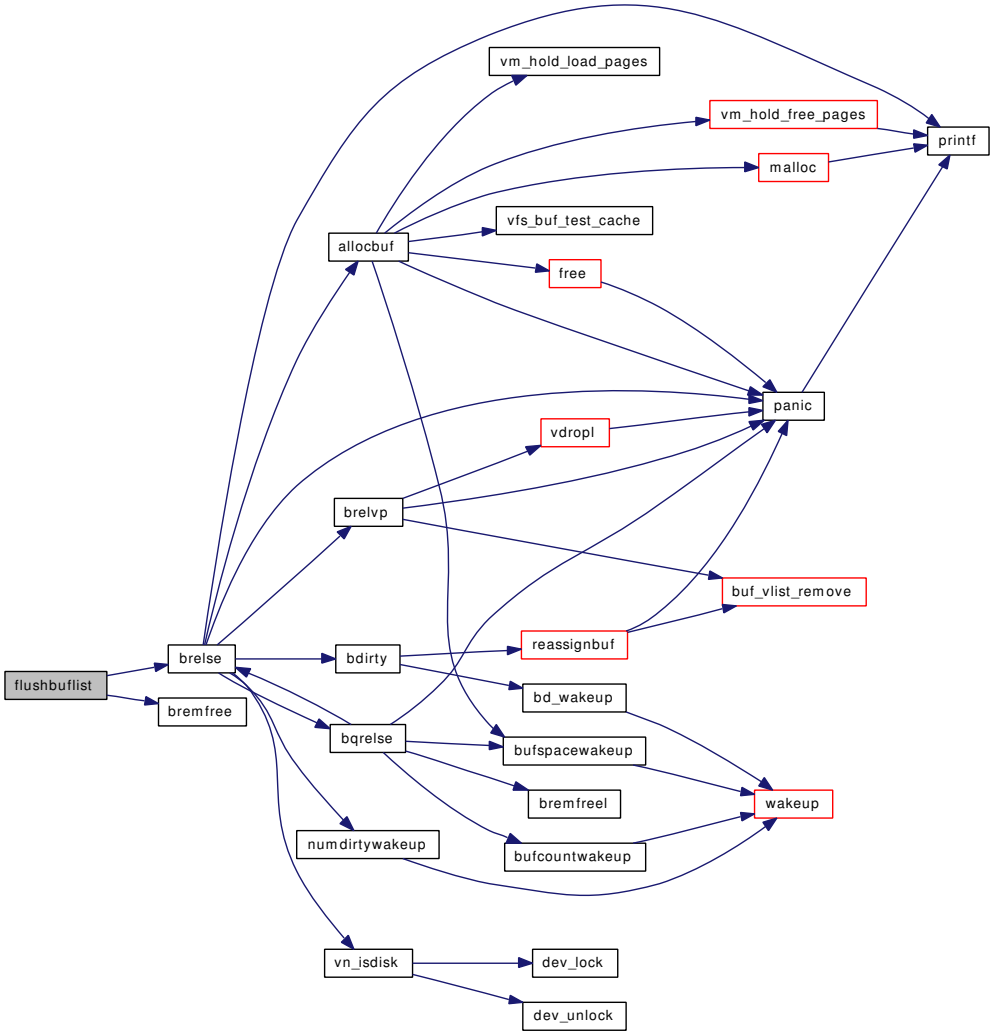
9.162.3.18 static int flushbuflist (struct bufv * *bufv*, int *flags*, struct bufobj * *bo*, int *slpflag*, int *slptimeo*) [static]

Definition at line 1099 of file vfs_subr.c.

References brelse(), bremfree(), and buf.

Referenced by bufobj_invalbuf().

Here is the call graph for this function:



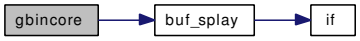
9.162.3.19 struct buf* gbincore (struct bufobj * *bo*, daddr_t *lblkno*)

Definition at line 1426 of file vfs_subr.c.

References buf, and buf_splay().

Referenced by cluster_read(), cluster_wbuild(), getblk(), incore(), and vfs_bio_clcheck().

Here is the call graph for this function:



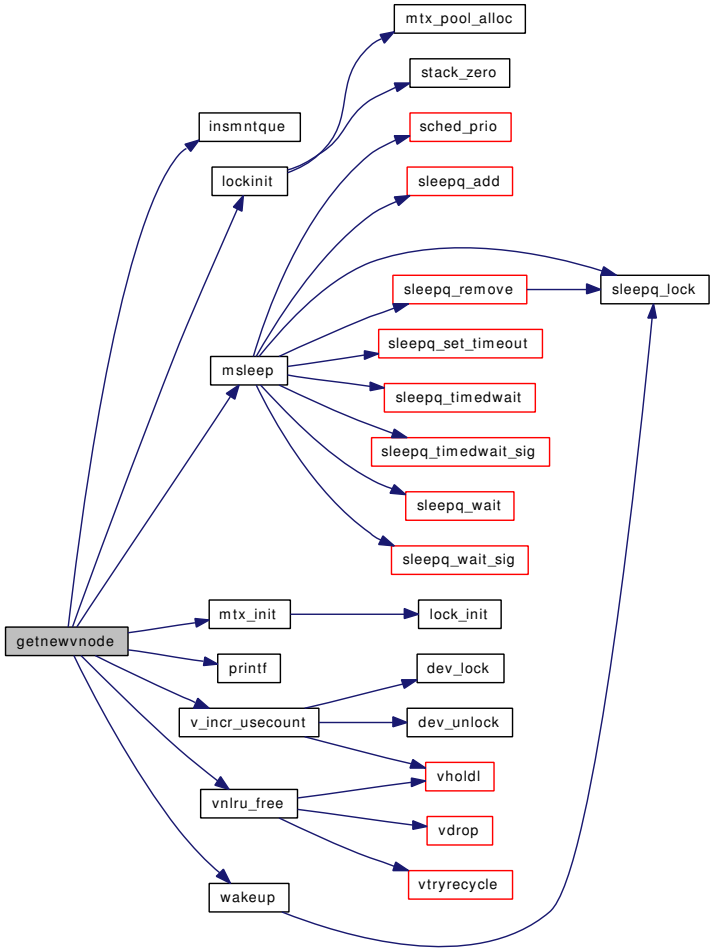
9.162.3.20 int getnewvnode (const char * *tag*, struct mount * *mp*, struct vop_vector * *vops*, struct vnode ** *vpp*)

Definition at line 863 of file vfs_subr.c.

References buf_ops_bio, hz, insmntque(), lockinit(), msleep(), mtx_init(), numvnodes, printf(), v_incr_usecount(), vnlru_free(), and wakeup().

Referenced by mqfs_allocv(), nameiinit(), and vfs_allocate_syncvnode().

Here is the call graph for this function:

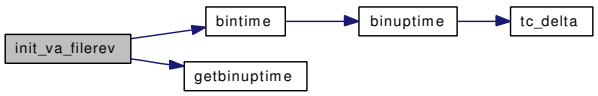


9.162.3.21 u_quad_t init_va_filerev (void)

Definition at line 3727 of file vfs_subr.c.

References bintime(), and getbinuptime().

Here is the call graph for this function:



9.162.3.22 static void insmntque (struct vnode * *vp*, struct mount * *mp*) [static]

Definition at line 982 of file vfs_subr.c.

Referenced by getnewvnode().

9.162.3.23 static MALLOC_DEFINE (M_NETADDR, "subr_export_host", "Export host address structure") [static]

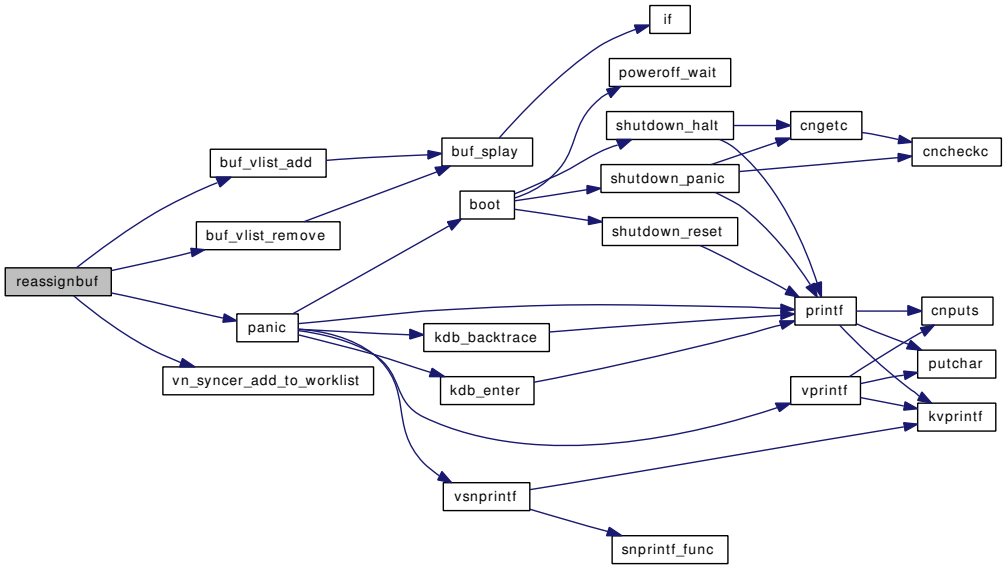
9.162.3.24 void reassignbuf (struct buf * *bp*)

Definition at line 1783 of file vfs_subr.c.

References buf_vlist_add(), buf_vlist_remove(), panic(), and vn_syncer_add_to_worklist().

Referenced by bdirty(), and bundirty().

Here is the call graph for this function:

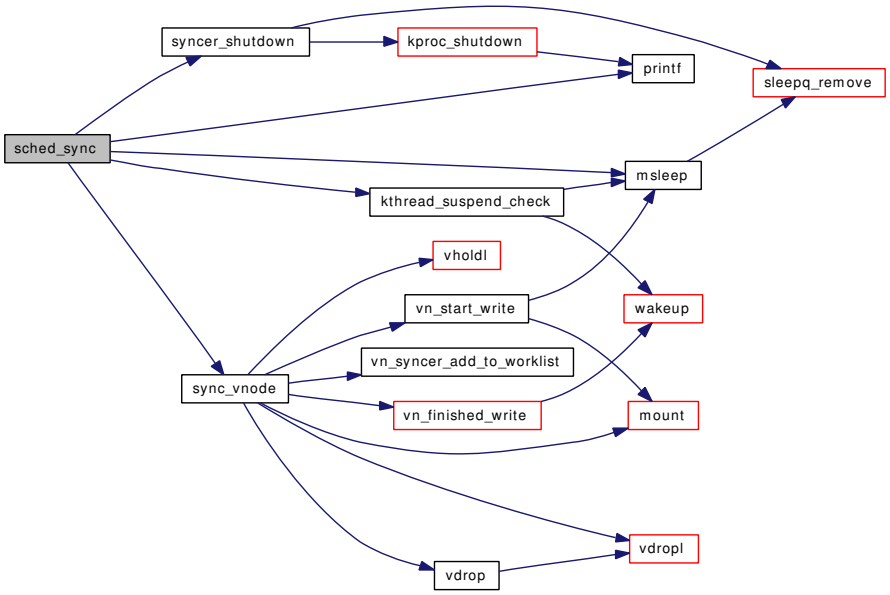


9.162.3.25 static void sched_sync (void) [static]

Definition at line 1609 of file vfs_subr.c.

References Giant, hz, kthread_suspend_check(), lbolt, msleep(), printf(), sync_vnode(), syncer_shutdown(), SYNCER_SHUTDOWN_SPEEDUP, syncer_state, td, time_uptime, and updateproc.

Here is the call graph for this function:

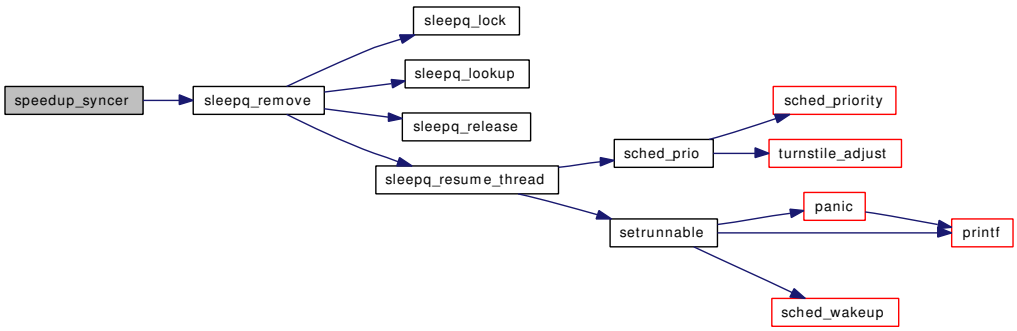


9.162.3.26 int speedup_syncer (void)

Definition at line 1740 of file vfs_subr.c.

References lbolt, ret, sleepq_remove(), td, and updateproc.

Here is the call graph for this function:

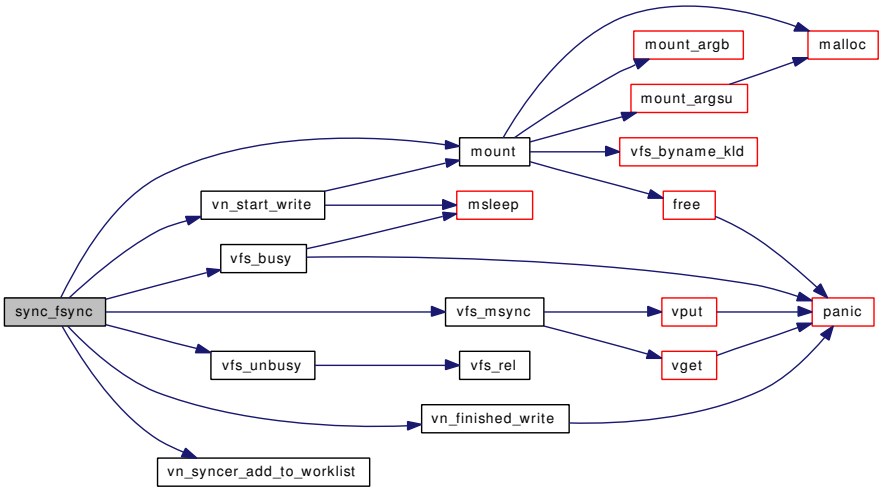


9.162.3.27 static int sync_fsync (struct vop_fsync_args *) [static]

Definition at line 3050 of file vfs_subr.c.

References mount(), mountlist_mtx, td, vfs_busy(), vfs_msync(), vfs_unbusy(), vn_finished_write(), vn_-start_write(), and vn_syncer_add_to_worklist().

Here is the call graph for this function:

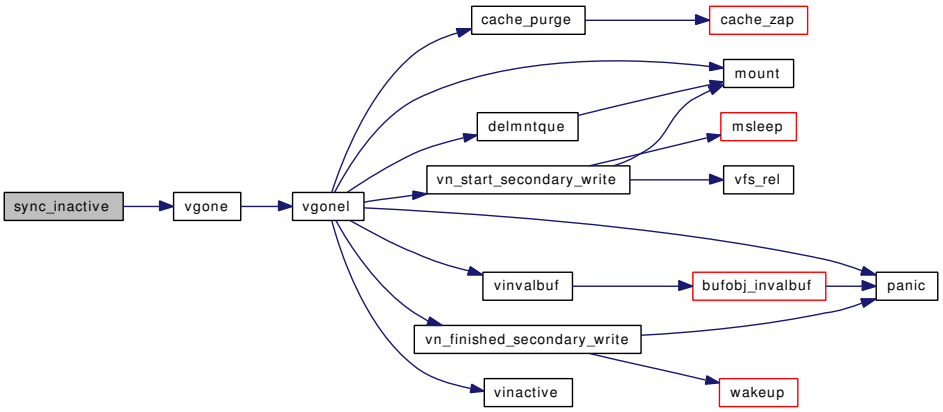


9.162.3.28 static int sync_inactive (struct vop_inactive_args *) [static]

Definition at line 3105 of file vfs_subr.c.

References vgone().

Here is the call graph for this function:



9.162.3.29 static int sync_reclaim (struct vop_reclaim_args *) [static]

Definition at line 3118 of file vfs_subr.c.

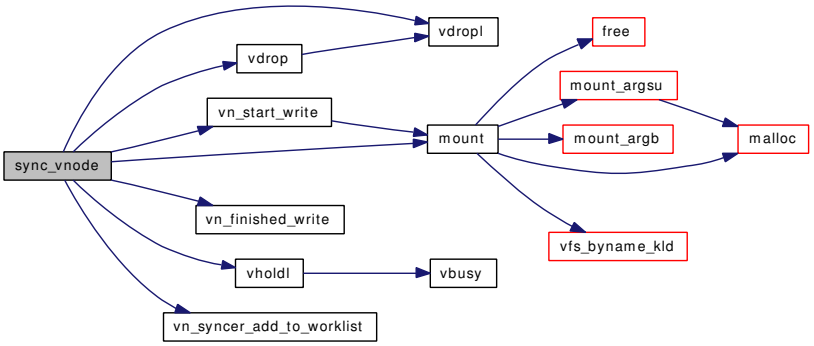
9.162.3.30 static int sync_vnode (struct bufobj * *bo*, struct thread * *td*) [static]

Definition at line 1562 of file vfs_subr.c.

References mount(), vdrop(), vdropl(), vholdl(), vn_finished_write(), vn_start_write(), and vn_syncer_-add_to_worklist().

Referenced by sched_sync().

Here is the call graph for this function:



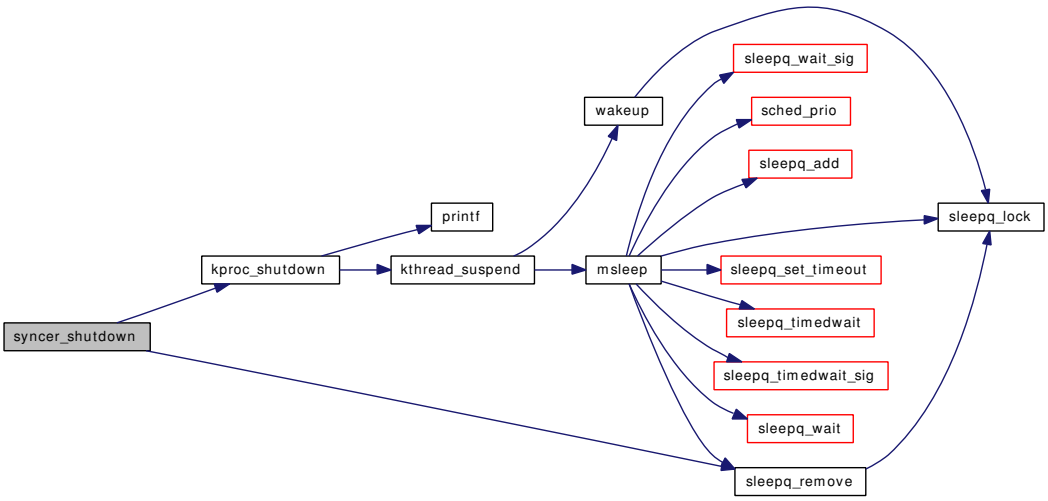
9.162.3.31 static void syncer_shutdown (void * *arg*, int *howto*) [static]

Definition at line 1762 of file vfs_subr.c.

References kproc_shutdown(), lbolt, sleepq_remove(), syncer_state, td, and updateproc.

Referenced by sched_sync().

Here is the call graph for this function:



9.162.3.32 SYSCTL_INT (_vfs, OID_AUTO, timestamp_precision, CTLFLAG_RW, & *timestamp_precision*, 0, "")

9.162.3.33 SYSCTL_INT (_debug, OID_AUTO, vnlru_nowhere, CTLFLAG_RW, & *vnlru_nowhere*, 0, "Number of times the vnlru process ran without success")

9.162.3.34 SYSCTL_INT (_kern, OID_AUTO, minvnodes, CTLFLAG_RW, & *wantfreevnodes*, 0, "Minimum number of vnodes (legacy)")

9.162.3.35 SYSCTL_INT (_kern, KERN_MAXVNODES, maxvnodes, CTLFLAG_RW, & *desiredvnodes*, 0, "Maximum number of vnodes")

9.162.3.36 SYSCTL_INT (_debug, OID_AUTO, mpsafevfs, CTLFLAG_RD, & *mpsafe_vfs*, 0, "MPSAFE VFS")

9.162.3.37 SYSCTL_LONG (_vfs, OID_AUTO, numvnodes, CTLFLAG_RD, & *numvnodes*, 0, "")

9.162.3.38 static SYSCTL_NODE (_vfs, VFS_GENERIC, generic, CTLFLAG_RD| *CTLFLAG_SKIP*, vfs_sysctl, "Generic filesystem") [static]

9.162.3.39 static int sysctl_ovfs_conf (SYSCTL_HANDLER_ARGS) [static]

Definition at line 2680 of file vfs_subr.c.

References vfsconf.

Referenced by vfs_sysctl().

9.162.3.40 SYSCTL_PROC (_vfs, OID_AUTO, ctl, CTLFLAG_WR, NULL, 0, sysctl_vfs_ctl, "", "Sysctl by fsid")

9.162.3.41 SYSCTL_PROC (_vfs, OID_AUTO, conflist, CTLFLAG_RD, NULL, 0, sysctl_vfs_conflist, " *S*, xvfsconf", "List of all configured filesystems")

9.162.3.42 SYSCTL_PROC (_vfs, OID_AUTO, worklist_len, CTLTYPE_INT| *CTLFLAG_RD*, NULL, 0, sysctl_vfs_worklist_len, "I", "Syncer thread worklist length")

9.162.3.43 static int sysctl_vfs_conflist (SYSCTL_HANDLER_ARGS) [static]

Definition at line 2614 of file vfs_subr.c.

References vfsconf, and vfsconf2x().

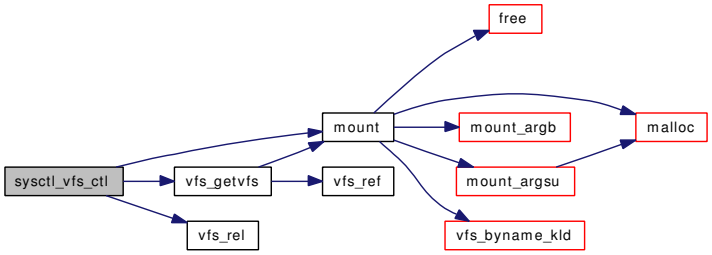
Here is the call graph for this function:

9.162.3.44 static int sysctl_vfs_ctl (SYSCTL_HANDLER_ARGS) [static]

Definition at line 3693 of file vfs_subr.c.

References mount(), vfs_getvfs(), and vfs_rel().

Here is the call graph for this function:



9.162.3.45 static int sysctl_vfs_worklist_len (SYSCTL_HANDLER_ARGS) [static]

Definition at line 1538 of file vfs_subr.c.

9.162.3.46 SYSINIT (vfs_knlist, SI_SUB_VFS, SI_ORDER_ANY, vfs_event_init, NULL)

9.162.3.47 static TAILQ_HEAD (freelst, vnode) [static]

Definition at line 152 of file vfs_subr.c.

References SYNCER_MAXDELAY.

9.162.3.48 TUNABLE_INT ("debug.mpsafevfs", & *mpsafe_vfs*)

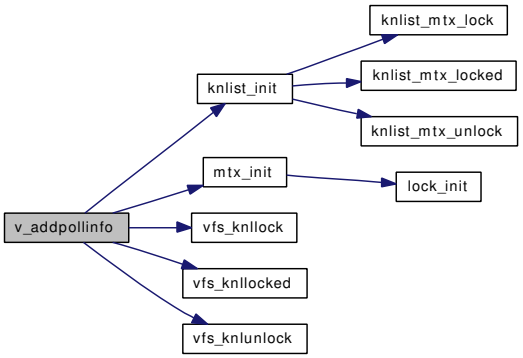
9.162.3.49 void v_addpollinfo (struct vnode * *vp*)

Definition at line 2933 of file vfs_subr.c.

References knlist_init(), mtx_init(), vfs_knllock(), vfs_knllocked(), and vfs_knlunlock().

Referenced by vfs_kqfilter(), and vn_pollrecord().

Here is the call graph for this function:



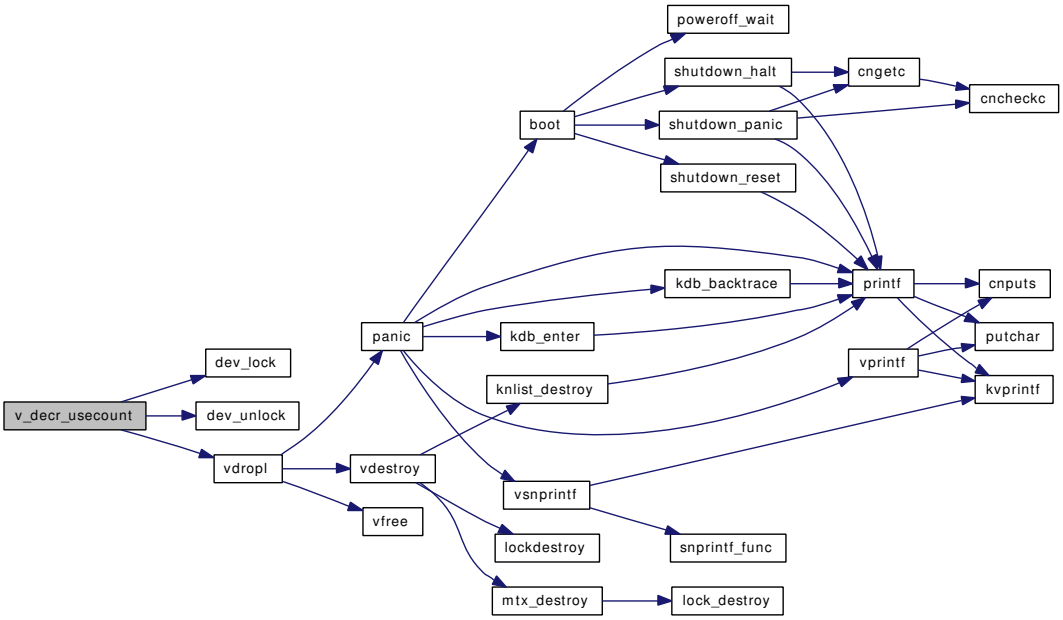
9.162.3.50 static void v_decr_usecount (struct vnode *) [static]

Definition at line 1907 of file vfs_subr.c.

References dev_lock(), dev_unlock(), and vdropl().

Referenced by vput(), and vrele().

Here is the call graph for this function:



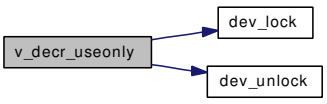
9.162.3.51 static void v_decr_useonly (struct vnode *) [static]

Definition at line 1931 of file vfs_subr.c.

References dev_lock(), and dev_unlock().

Referenced by vput(), and vrele().

Here is the call graph for this function:



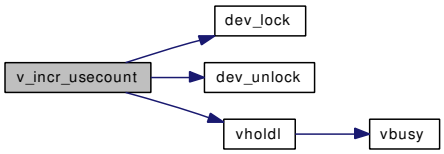
9.162.3.52 static void v_incr_usecount (struct vnode *) [static]

Definition at line 1869 of file vfs_subr.c.

References dev_lock(), dev_unlock(), and vholdl().

Referenced by getnewvnode(), and vref().

Here is the call graph for this function:



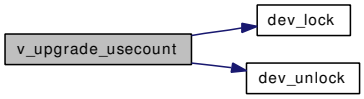
9.162.3.53 static void v_upgrade_usecount (struct vnode *) [static]

Definition at line 1888 of file vfs_subr.c.

References dev_lock(), and dev_unlock().

Referenced by vget().

Here is the call graph for this function:



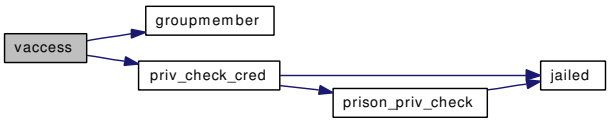
9.162.3.54 int vaccess (enum vtype *type*, mode_t *file_mode*, uid_t *file_uid*, gid_t *file_gid*, mode_t *acc_mode*, struct ucred * *cred*, int * *privused*)

Definition at line 3174 of file vfs_subr.c.

References groupmember(), and priv_check_cred().

Referenced by kmq_open(), and mqfs_access().

Here is the call graph for this function:



9.162.3.55 void vattr_null (struct vattr * *vap*)

Definition at line 504 of file vfs_subr.c.

Referenced by vfs_register().

9.162.3.56 static void vbusy (struct vnode * *vp*) [static]

Definition at line 2915 of file vfs_subr.c.

Referenced by vholdl().

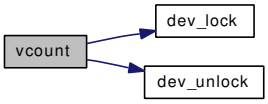
9.162.3.57 int vcount (struct vnode * *vp*)

Definition at line 2478 of file vfs_subr.c.

References dev_lock(), and dev_unlock().

Referenced by revoke().

Here is the call graph for this function:



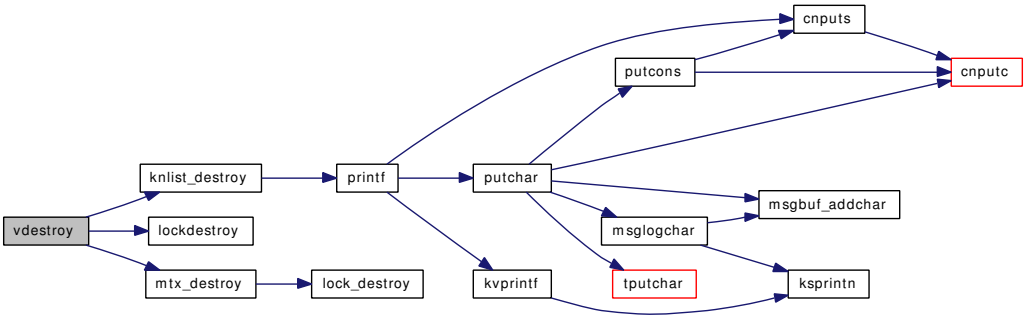
9.162.3.58 static void vdestroy (struct vnode *) [static]

Definition at line 771 of file vfs_subr.c.

References knlist_destroy(), lockdestroy(), mtx_destroy(), and numvnodes.

Referenced by vdropl().

Here is the call graph for this function:



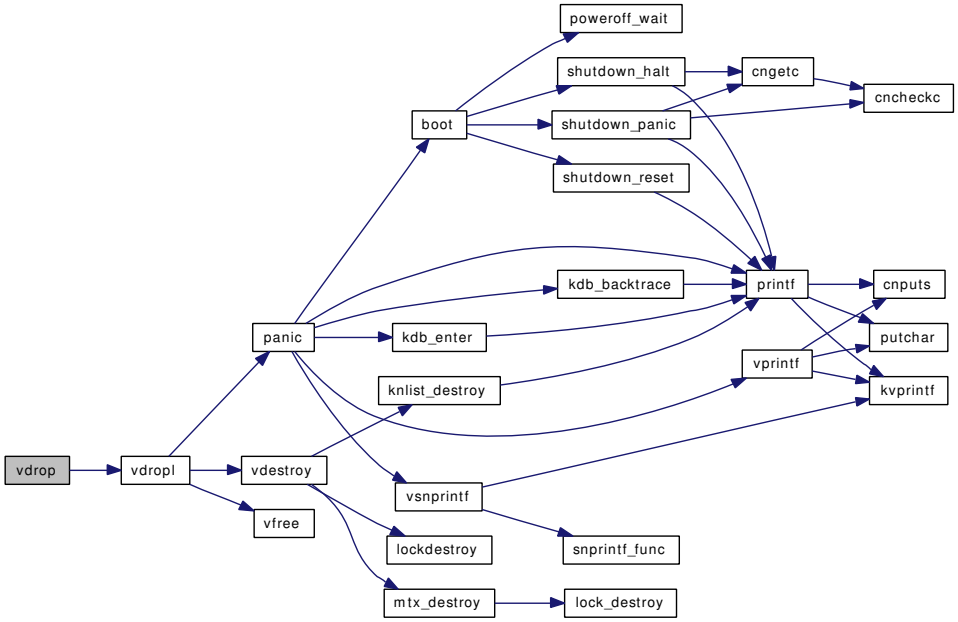
9.162.3.59 void vdrop (struct vnode * *vp*)

Definition at line 2177 of file vfs_subr.c.

References vdropl().

Referenced by cache_zap(), do_recycle(), dounmount(), sync_vnode(), vflush(), vget(), vlrureclaim(), vnlru_free(), and vop_rename_post().

Here is the call graph for this function:



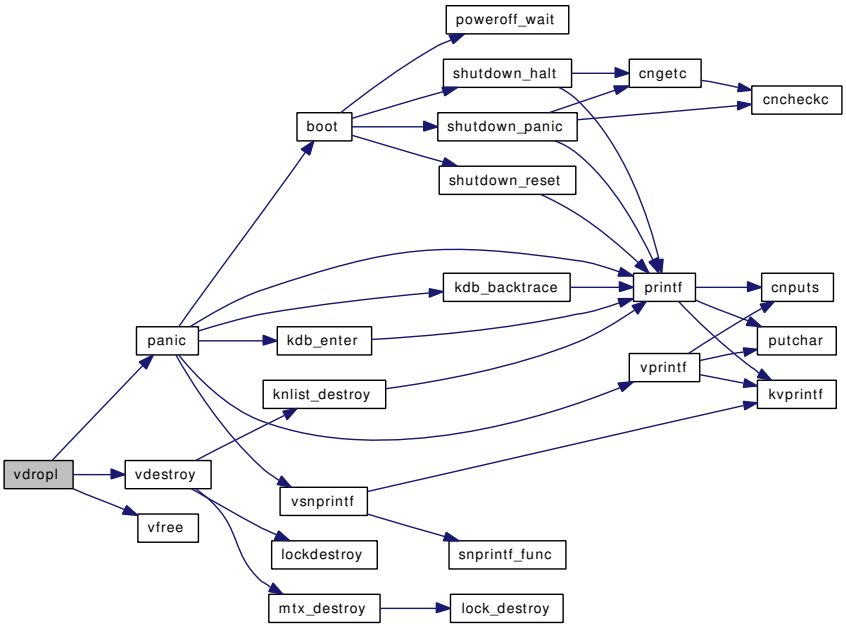
9.162.3.60 static void vdropl (struct vnode * *vp*) [static]

Definition at line 2190 of file vfs_subr.c.

References panic(), vdestroy(), and vfree().

Referenced by brelvp(), sync_vnode(), v_decr_usecount(), vdrop(), vflush(), vlrureclaim(), vput(), and vrele().

Here is the call graph for this function:



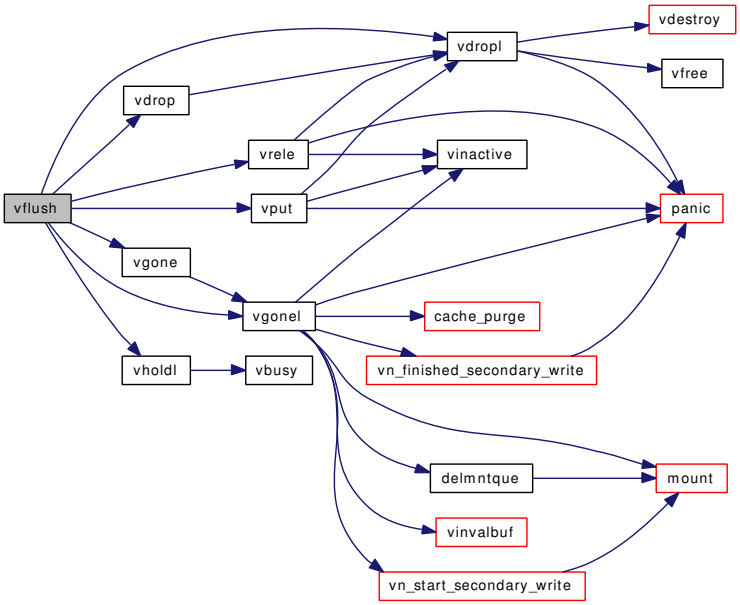
9.162.3.61 int vflush (struct mount * *mp*, int *rootrefs*, int *flags*, struct thread * *td*)

Definition at line 2256 of file vfs_subr.c.

References vdrop(), vdropl(), vgone(), vgonel(), vholdl(), vput(), and vrele().

Referenced by mqfs_unmount().

Here is the call graph for this function:



9.162.3.62 static void vfree (struct vnode *) [static]

Definition at line 2889 of file vfs_subr.c.

References VSHOULDFREE.

Referenced by vdropl().

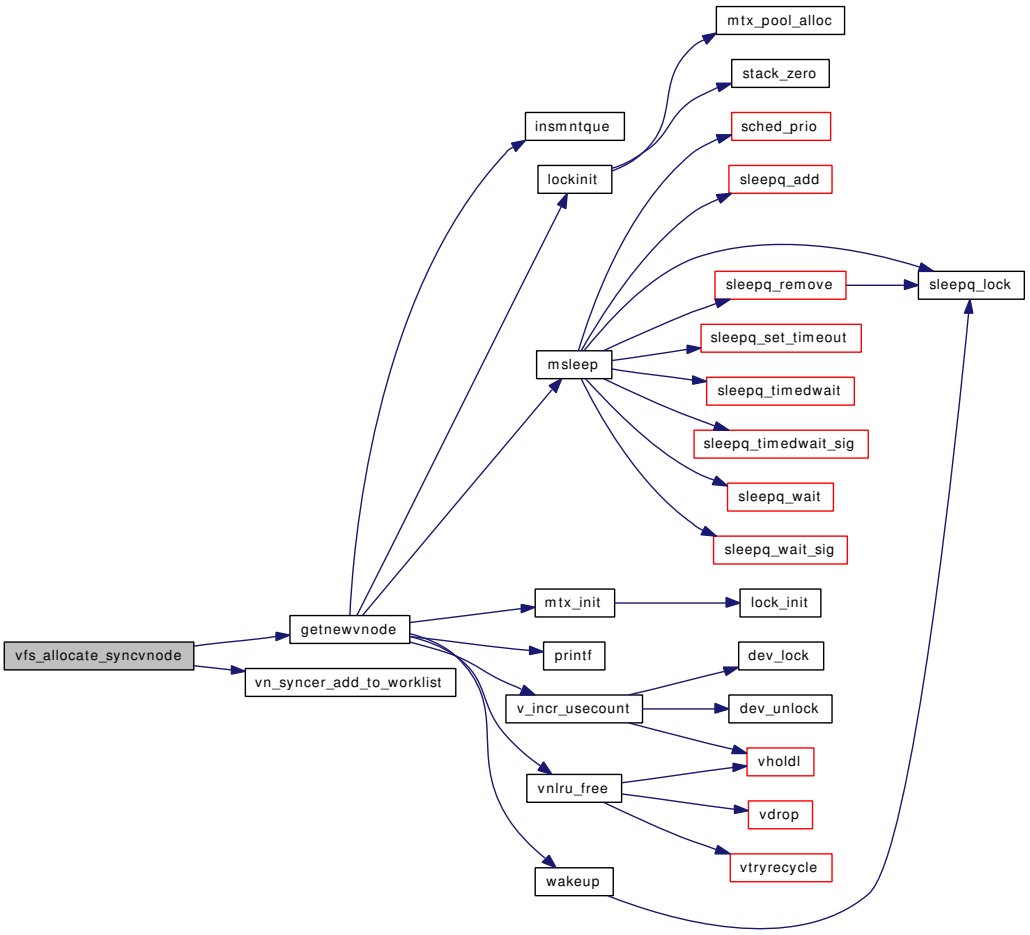
9.162.3.63 int vfs_allocate_syncvnode (struct mount * *mp*)

Definition at line 3006 of file vfs_subr.c.

References getnewvnode(), and vn_syncer_add_to_worklist().

Referenced by dounmount(), and vfs_domount().

Here is the call graph for this function:



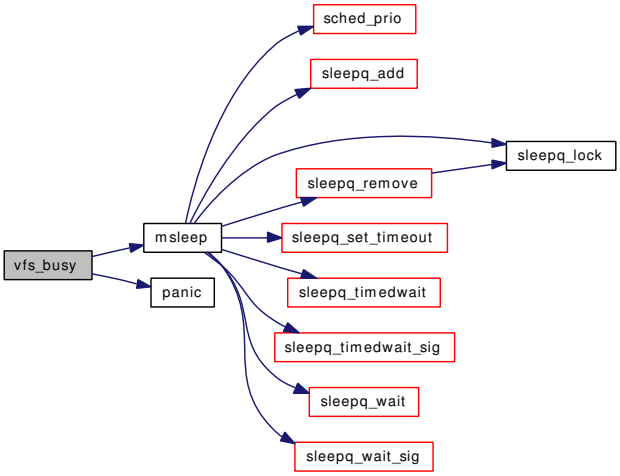
9.162.3.64 int vfs_busy (struct mount * *mp*, int *flags*, struct mtx * *interlkp*, struct thread * *td*)

Definition at line 337 of file vfs_subr.c.

References msleep(), and panic().

Referenced by fchdir(), sync(), sync_fsync(), vfs_domount(), vfs_mount_alloc(), and vnlru_proc().

Here is the call graph for this function:

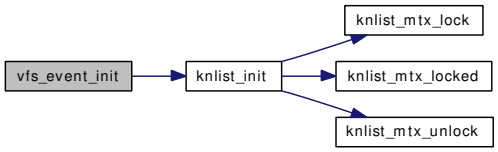


9.162.3.65 static void vfs_event_init (void * *arg*) [static]

Definition at line 3647 of file vfs_subr.c.

References knlist_init().

Here is the call graph for this function:



9.162.3.66 void vfs_event_signal (fsid_t * *fsid*, u_int32_t *event*, intptr_t data *__unused*)

Definition at line 3655 of file vfs_subr.c.

Referenced by dounmount(), and vfs_domount().

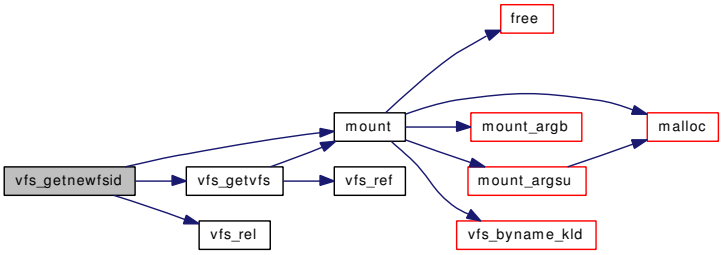
9.162.3.67 void vfs_getnewfsid (struct mount * *mp*)

Definition at line 435 of file vfs_subr.c.

References mount(), vfs_getvfs(), and vfs_rel().

Referenced by mqfs_mount().

Here is the call graph for this function:



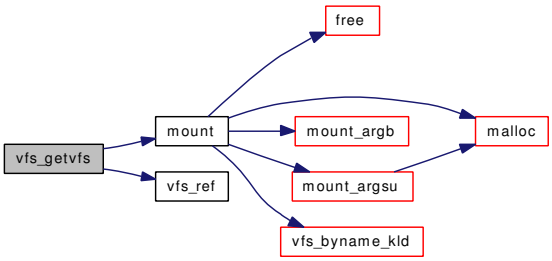
9.162.3.68 struct mount* vfs_getvfs (fsid_t * *fsid*)

Definition at line 389 of file vfs_subr.c.

References mount(), mountlist, mountlist_mtx, and vfs_ref().

Referenced by fhopen(), fhstat(), kern_fhstatfs(), sysctl_vfs_ctl(), and vfs_getnewfsid().

Here is the call graph for this function:



9.162.3.69 static void vfs_knllock (void * *arg*) [static]

Definition at line 3747 of file vfs_subr.c.

Referenced by v_addpollinfo().

9.162.3.70 static int vfs_knllocked (void * *arg*) [static]

Definition at line 3763 of file vfs_subr.c.

Referenced by v_addpollinfo().

9.162.3.71 static void vfs_knlunlock (void * *arg*) [static]

Definition at line 3755 of file vfs_subr.c.

Referenced by v_addpollinfo().

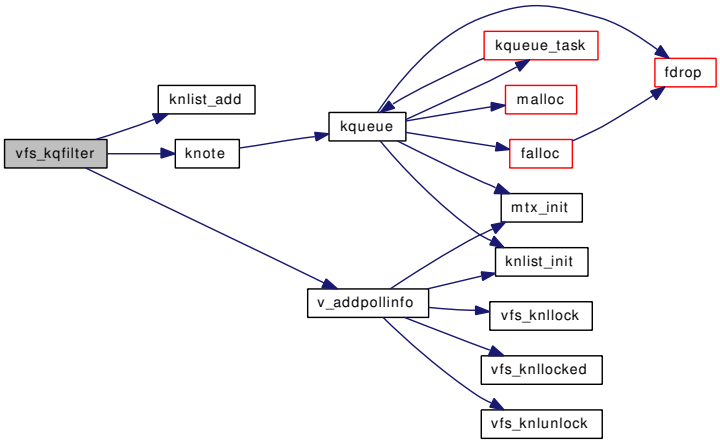
9.162.3.72 int vfs_kqfilter (struct vop_kqfilter_args * *ap*)

Definition at line 3771 of file vfs_subr.c.

References knlist_add(), knote(), and v_addpollinfo().

Referenced by vop_stdkqfilter().

Here is the call graph for this function:



9.162.3.73 void vfs_mark_atime (struct vnode * *vp*, struct thread * *td*)

Definition at line 3901 of file vfs_subr.c.

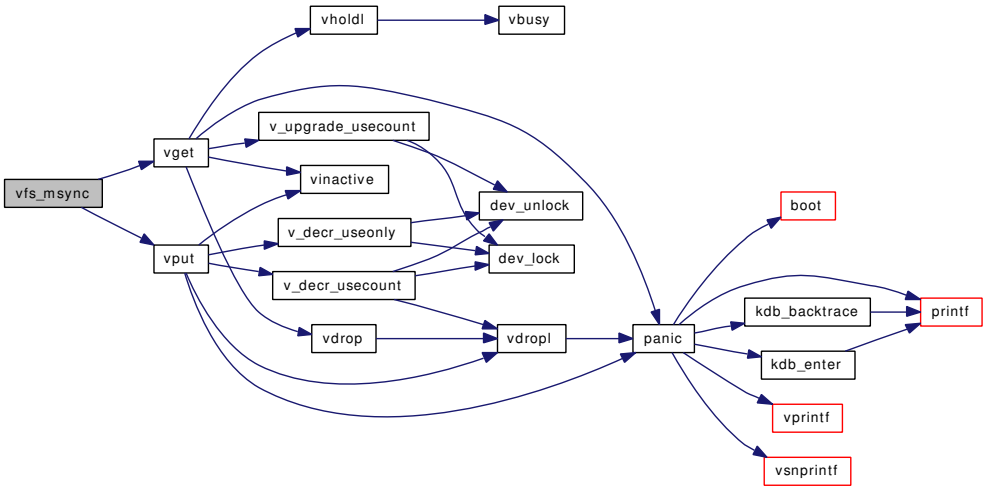
9.162.3.74 void vfs_msync (struct mount * *mp*, int *flags*)

Definition at line 2848 of file vfs_subr.c.

References vget(), and vput().

Referenced by dounmount(), sync(), and sync_fsync().

Here is the call graph for this function:



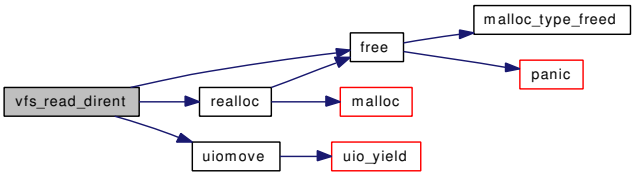
9.162.3.75 int vfs_read_dirent (struct vop_readdir_args * *ap*, struct dirent * *dp*, off_t *off*)

Definition at line 3866 of file vfs_subr.c.

References free(), realloc(), and uiomove().

Referenced by mqfs_readdir().

Here is the call graph for this function:



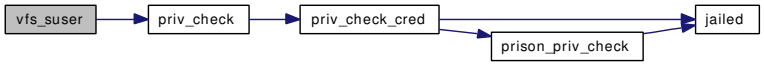
9.162.3.76 int vfs_suser (struct mount * *mp*, struct thread * *td*)

Definition at line 410 of file vfs_subr.c.

References priv_check().

Referenced by dounmount(), and vfs_domount().

Here is the call graph for this function:

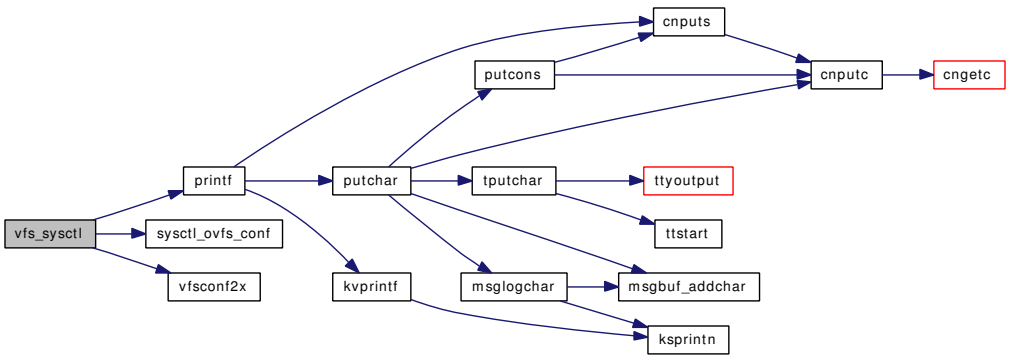


9.162.3.77 static int vfs_sysctl (SYSCTL_HANDLER_ARGS) [static]

Definition at line 2638 of file vfs_subr.c.

References maxvfsconf, printf(), sysctl_ovfs_conf(), vfsconf, and vfsconf2x().

Here is the call graph for this function:



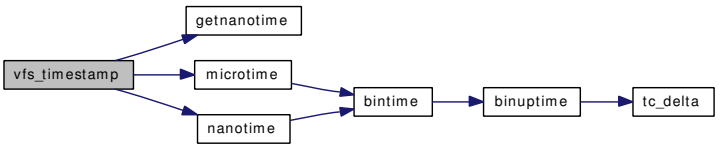
9.162.3.78 void vfs_timestamp (struct timespec * *tsp*)

Definition at line 477 of file vfs_subr.c.

References getnanotime(), microtime(), nanotime(), time_second, TSP_HZ, TSP_SEC, and TSP_USEC.

Referenced by mqfs_setattr(), pipe_read(), pipe_write(), and pipe_zone_ctor().

Here is the call graph for this function:



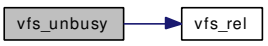
9.162.3.79 void vfs_unbusy (struct mount * *mp*, struct thread * *td*)

Definition at line 378 of file vfs_subr.c.

References vfs_rel().

Referenced by devfs_fixup(), fchdir(), sync(), sync_fsync(), vfs_domount(), and vnlru_proc().

Here is the call graph for this function:

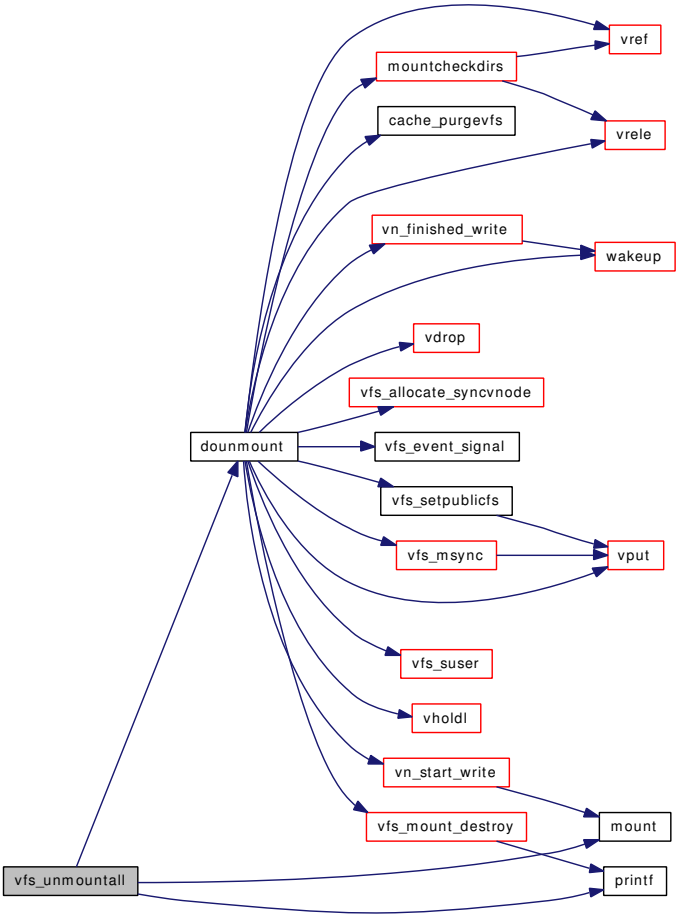


9.162.3.80 void vfs_unmountall (void)

Definition at line 2805 of file vfs_subr.c.

References dounmount(), mount(), mountlist, printf(), and td.

Here is the call graph for this function:



9.162.3.81 static void vfsconf2x (struct vfsconf * *vfsp*, struct xvfsconf * *xvfsp*) [static]

Definition at line 2595 of file vfs_subr.c.

Referenced by sysctl_vfs_conflist(), and vfs_sysctl().

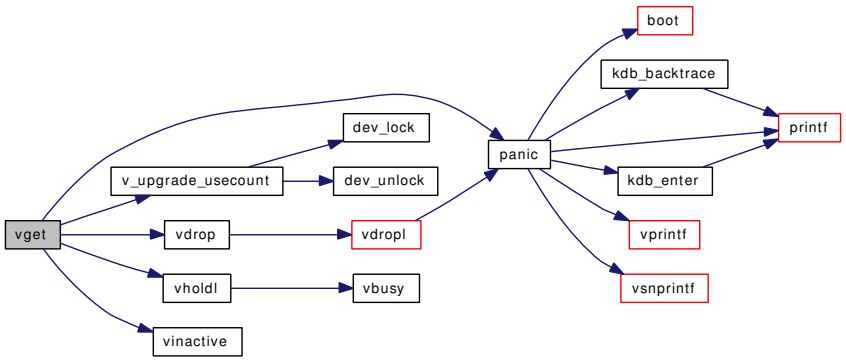
9.162.3.82 int vget (struct vnode * *vp*, int *flags*, struct thread * *td*)

Definition at line 1956 of file vfs_subr.c.

References panic(), v_upgrade_usecount(), vdrop(), vholdl(), and vinactive().

Referenced by cache_lookup(), mqfs_allocv(), vfs_hash_get(), vfs_hash_insert(), vfs_msync(), and vfs_-stdsync().

Here is the call graph for this function:



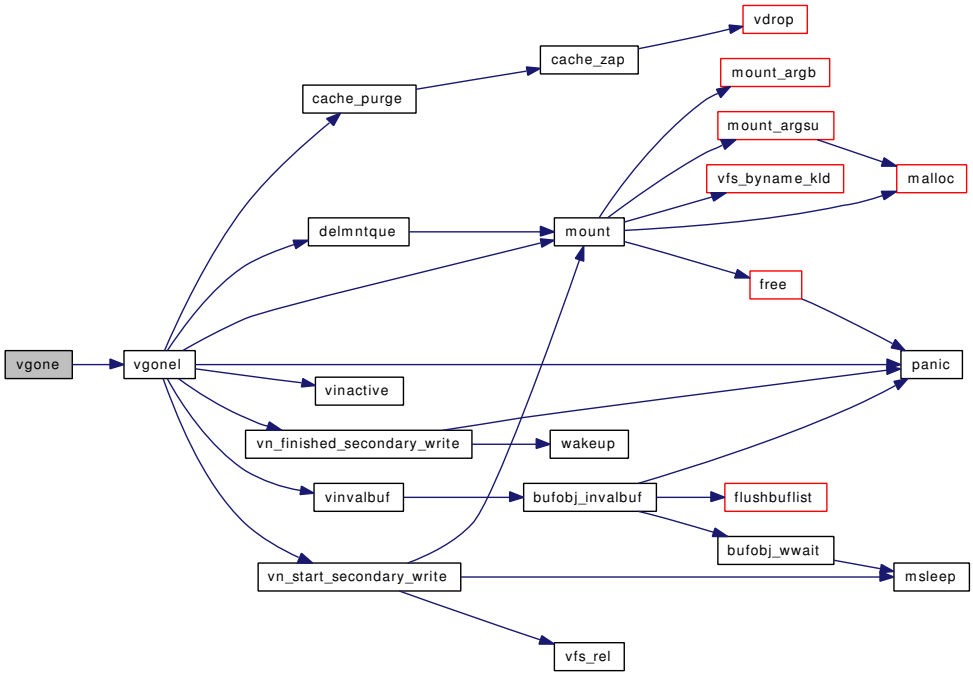
9.162.3.83 void vgone (struct vnode * *vp*)

Definition at line 2389 of file vfs_subr.c.

References vgonel().

Referenced by sync_inactive(), and vflush().

Here is the call graph for this function:



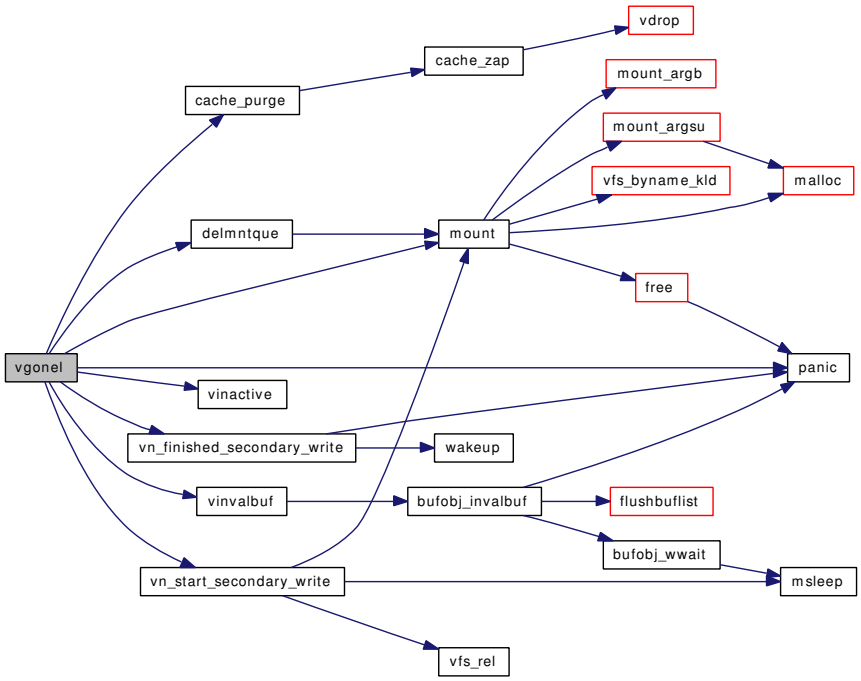
9.162.3.84 void vgonel (struct vnode *) [static]

Definition at line 2400 of file vfs_subr.c.

References cache_purge(), delmntque(), mount(), panic(), td, vinactive(), vinvalbuf(), vn_finished_-secondary_write(), and vn_start_secondary_write().

Referenced by vflush(), vgone(), vlrureclaim(), vrecycle(), and vtryrecycle().

Here is the call graph for this function:



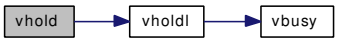
9.162.3.85 void vhold (struct vnode * *vp*)

Definition at line 2155 of file vfs_subr.c.

References vholdl().

Referenced by cache_enter(), do_unlink(), and vop_rename_pre().

Here is the call graph for this function:



9.162.3.86 void vholdl (struct vnode * *vp*)

Definition at line 2164 of file vfs_subr.c.

References vbusy(), and VSHOULDBUSY.

Referenced by bgetvp(), dounmount(), sync_vnode(), v_incr_usecount(), vflush(), vget(), vhold(), vlrureclaim(), and vnlru_free().

Here is the call graph for this function:

9.162.3.87 static void vinactive (struct vnode *, struct thread *) [static]

Definition at line 2213 of file vfs_subr.c.

Referenced by vget(), vgonel(), vput(), and vrele().

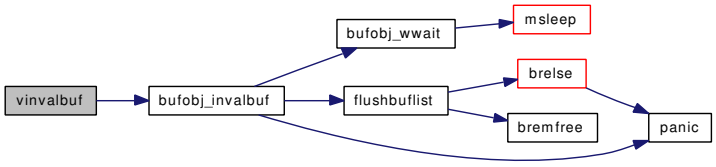
9.162.3.88 int vinvalbuf (struct vnode * *vp*, int *flags*, struct thread * *td*, int *slpflag*, int *slptimeo*)

Definition at line 1085 of file vfs_subr.c.

References bufobj_invalbuf().

Referenced by devfs_fixup(), vfs_domount(), and vgonel().

Here is the call graph for this function:



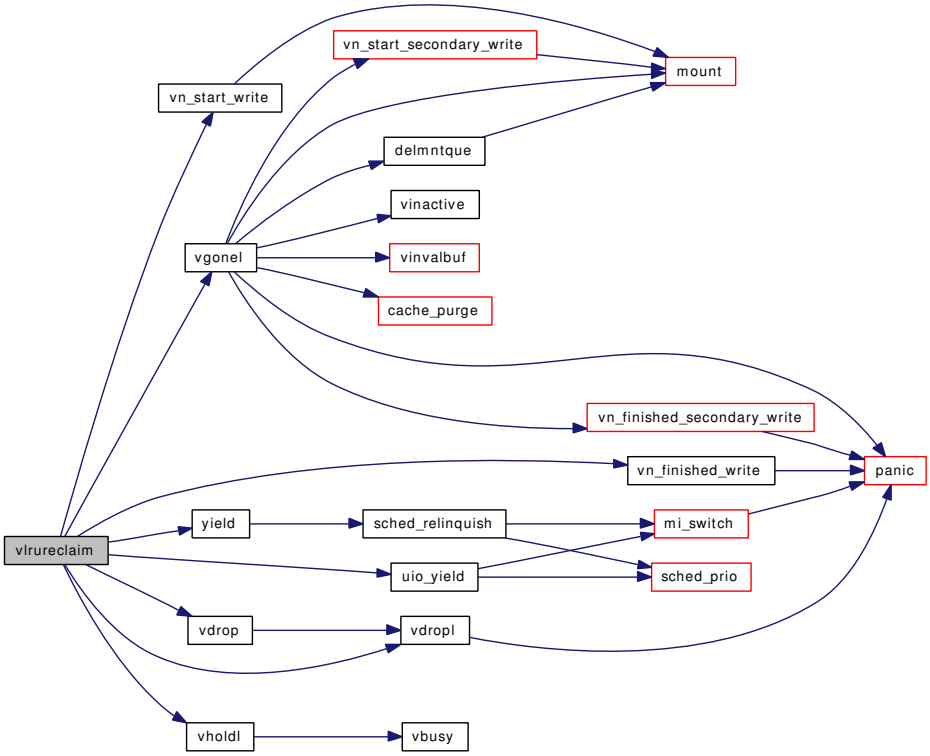
9.162.3.89 static int vlrureclaim (struct mount * *mp*) [static]

Definition at line 547 of file vfs_subr.c.

References td, uio_yield(), vdrop(), vdropl(), vgonel(), vholdl(), vn_finished_write(), vn_start_write(), and yield().

Referenced by vnlru_proc().

Here is the call graph for this function:



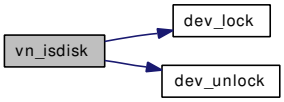
9.162.3.90 int vn_isdisk (struct vnode * *vp*, int * *errp*)

Definition at line 3143 of file vfs_subr.c.

References dev_lock(), and dev_unlock().

Referenced by aio_cancel(), aio_qphysio(), brelse(), and bufdone_finish().

Here is the call graph for this function:



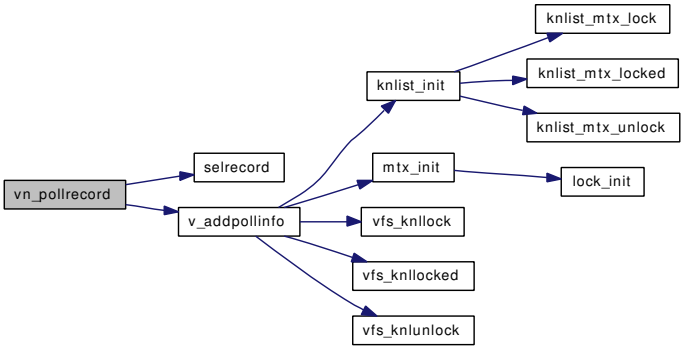
9.162.3.91 int vn_pollrecord (struct vnode * *vp*, struct thread * *td*, int *events*)

Definition at line 2957 of file vfs_subr.c.

References selrecord(), and v_addpollinfo().

Referenced by vop_stdpoll().

Here is the call graph for this function:

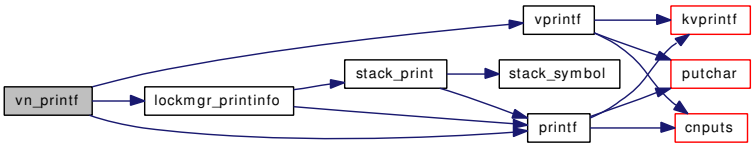


9.162.3.92 void vn_printf (struct vnode * *vp*, const char * *fmt*, ...)

Definition at line 2510 of file vfs_subr.c.

References buf, lockmgr_printinfo(), printf(), and vprintf().

Here is the call graph for this function:



9.162.3.93 static void vn_syncer_add_to_worklist (struct bufobj * *bo*, int *delay*) [static]

Definition at line 1515 of file vfs_subr.c.

Referenced by reassignbuf(), sync_fsync(), sync_vnode(), and vfs_allocate_syncvnode().

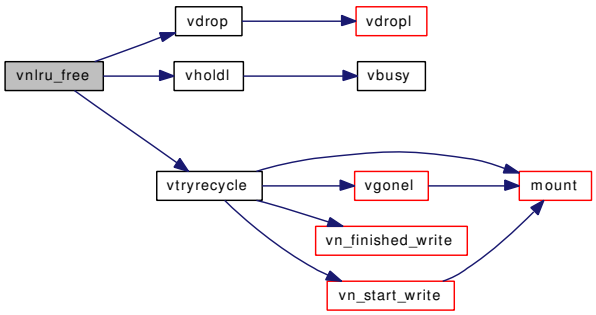
9.162.3.94 static void vnlru_free (int) [static]

Definition at line 646 of file vfs_subr.c.

References VCANRECYCLE, vdrop(), vholdl(), and vtryrecycle().

Referenced by getnewvnode(), and vnlru_proc().

Here is the call graph for this function:

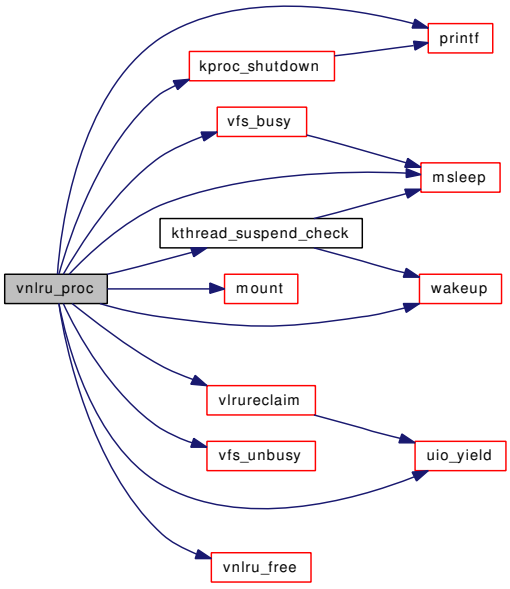


9.162.3.95 static void vnlru_proc (void) [static]

Definition at line 698 of file vfs_subr.c.

References Giant, hz, kproc_shutdown(), kthread_suspend_check(), mount(), mountlist, mountlist_mtx, msleep(), numvnodes, printf(), td, uio_yield(), vfs_busy(), vfs_unbusy(), vlrureclaim(), vnlru_free(), and wakeup().

Here is the call graph for this function:

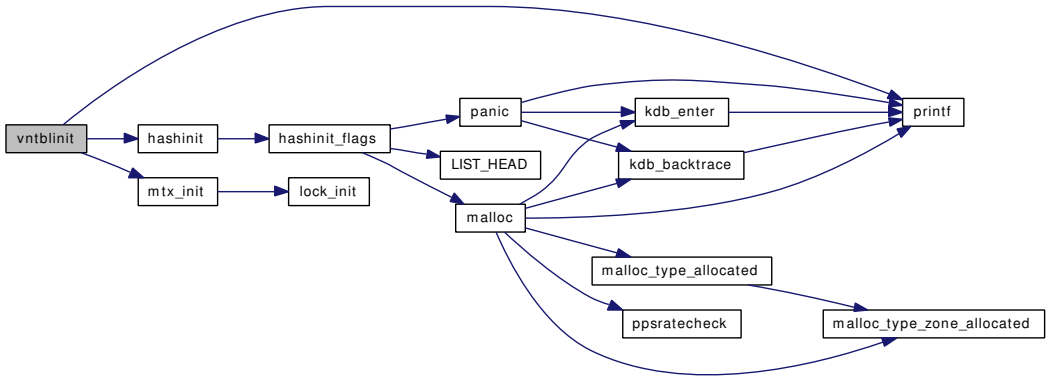


9.162.3.96 static void vntblinit (void *dummy *__unused*) [static]

Definition at line 295 of file vfs_subr.c.

References bootverbose, hashinit(), maxproc, MAXVNODES_MAX, mtx_init(), printf(), and vm_kmem_-size.

Here is the call graph for this function:



9.162.3.97 void vop_create_post (void * *ap*, int *rc*)

Definition at line 3546 of file vfs_subr.c.

9.162.3.98 void vop_link_post (void * *ap*, int *rc*)

Definition at line 3555 of file vfs_subr.c.

9.162.3.99 void vop_lock_post (void * *ap*, int *rc*)

Definition at line 3511 of file vfs_subr.c.

9.162.3.100 void vop_lock_pre (void * *ap*)

Definition at line 3498 of file vfs_subr.c.

9.162.3.101 void vop_lookup_post (void * *ap*, int *rc*)

Definition at line 3478 of file vfs_subr.c.

9.162.3.102 void vop_lookup_pre (void * *ap*)

Definition at line 3464 of file vfs_subr.c.

9.162.3.103 void vop_mkdir_post (void * *ap*, int *rc*)

Definition at line 3566 of file vfs_subr.c.

9.162.3.104 void vop_mknod_post (void * *ap*, int *rc*)

Definition at line 3575 of file vfs_subr.c.

9.162.3.105 void vop_remove_post (void * *ap*, int *rc*)

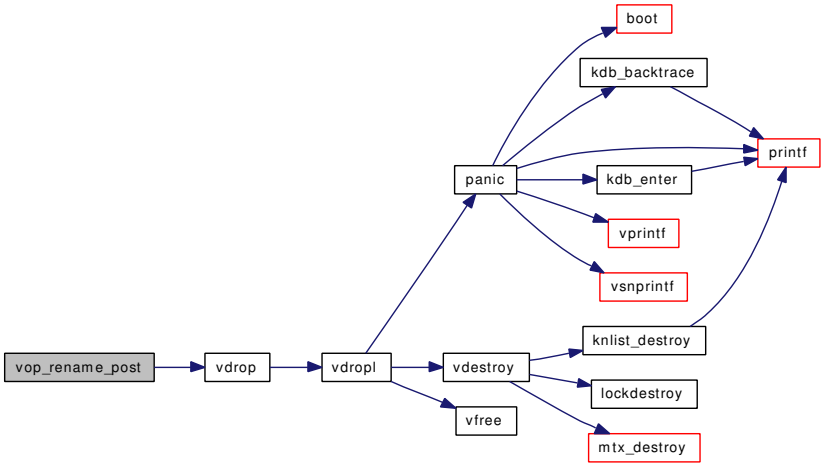
Definition at line 3584 of file vfs_subr.c.

9.162.3.106 void vop_rename_post (void * *ap*, int *rc*)

Definition at line 3595 of file vfs_subr.c.

References vdrop().

Here is the call graph for this function:

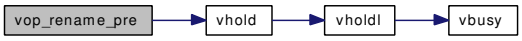


9.162.3.107 void vop_rename_pre (void * *ap*)

Definition at line 3406 of file vfs_subr.c.

References vhold().

Here is the call graph for this function:



9.162.3.108 void vop_rmdir_post (void * *ap*, int *rc*)

Definition at line 3616 of file vfs_subr.c.

9.162.3.109 void vop_setattr_post (void * *ap*, int *rc*)

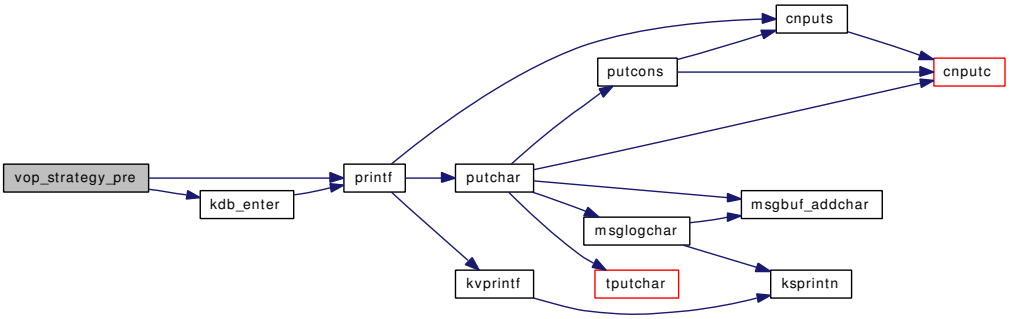
Definition at line 3627 of file vfs_subr.c.

9.162.3.110 void vop_strategy_pre (void * *ap*)

Definition at line 3438 of file vfs_subr.c.

References buf, kdb_enter(), and printf().

Here is the call graph for this function:



9.162.3.111 void vop_symlink_post (void * *ap*, int *rc*)

Definition at line 3636 of file vfs_subr.c.

9.162.3.112 void vop_unlock_post (void * *ap*, int *rc*)

Definition at line 3535 of file vfs_subr.c.

9.162.3.113 void vop_unlock_pre (void * *ap*)

Definition at line 3523 of file vfs_subr.c.

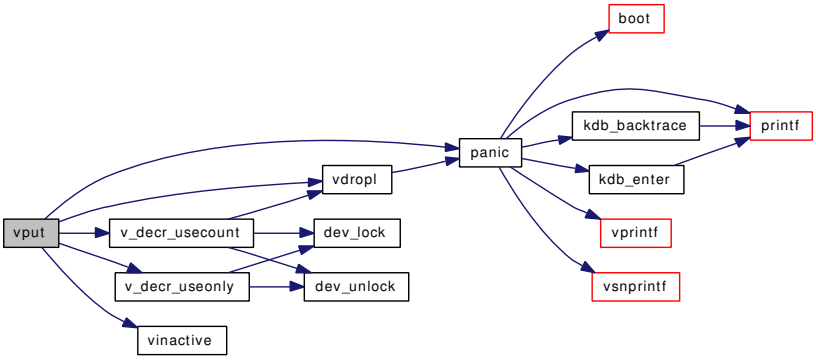
9.162.3.114 void vput (struct vnode * *vp*)

Definition at line 2099 of file vfs_subr.c.

References panic(), td, v_decr_usecount(), v_decr_useonly(), vdropl(), and vinactive().

Referenced by chroot(), devfs_fixup(), do_execve(), dounmount(), exit1(), extattrctl(), fchdir(), fhopen(), fhstat(), getdirentries(), getfh(), kern_access(), kern_chdir(), kern_eaccess(), kern_fhstatfs(), kern_link(), kern_lstat(), kern_mkdir(), kern_mkfifo(), kern_mknod(), kern_pathconf(), kern_readlink(), kern_rename(), kern_rmdir(), kern_stat(), kern_statfs(), kern_symlink(), kern_truncate(), kern_unlink(), lgetfh(), lookup(), namei(), NDFREE(), relookup(), revoke(), ttioctl(), uipc_bind(), undelete(), unp_connect(), vflush(), vfs_domount(), vfs_hash_insert(), vfs_msync(), vfs_setpublicfs(), vn_close(), and vn_open_cred().

Here is the call graph for this function:



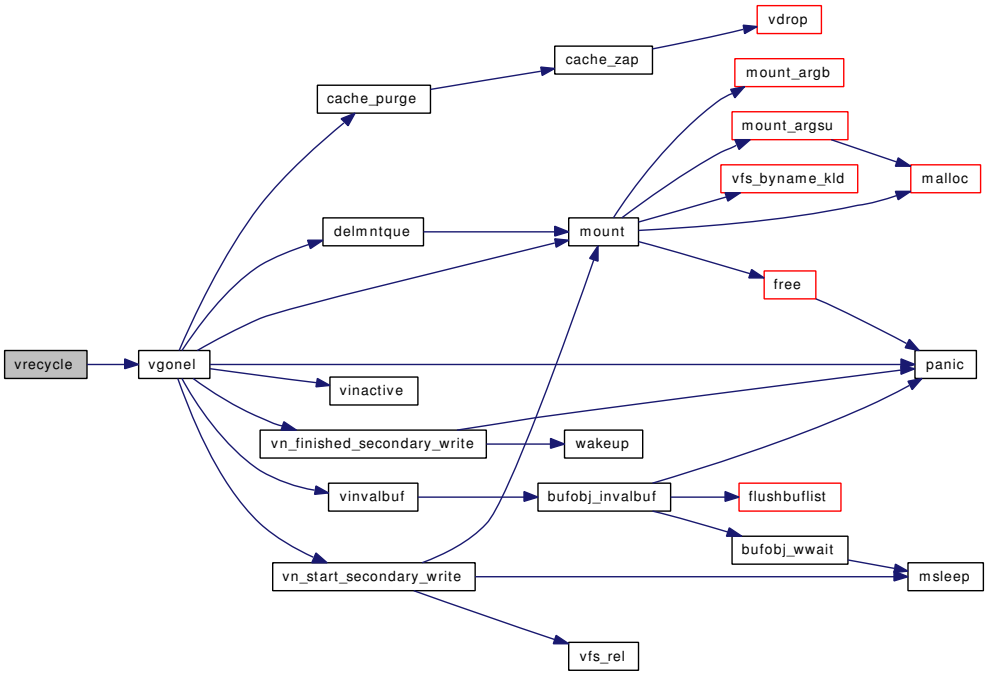
9.162.3.115 int vrecycle (struct vnode * *vp*, struct thread * *td*)

Definition at line 2369 of file vfs_subr.c.

References vgonel().

Referenced by do_recycle(), and mqfs_inactive().

Here is the call graph for this function:



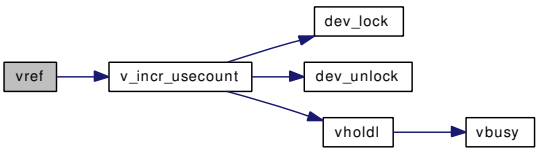
9.162.3.116 void vref (struct vnode * *vp*)

Definition at line 2007 of file vfs_subr.c.

References v_incr_usecount().

Referenced by _fgetvp(), dounmount(), fork1(), mountcheckdirs(), and sysctl_kern_proc_pathname().

Here is the call graph for this function:



9.162.3.117 int vrefcnt (struct vnode * *vp*)

Definition at line 2025 of file vfs_subr.c.

Referenced by mountcheckdirs().

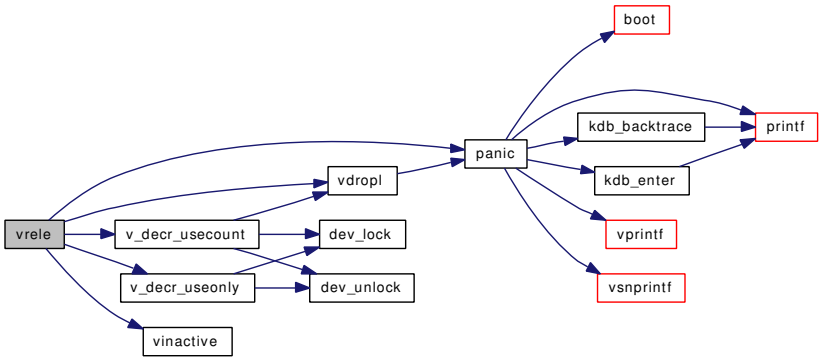
9.162.3.118 void vrele (struct vnode * *vp*)

Definition at line 2042 of file vfs_subr.c.

References panic(), td, v_decr_usecount(), v_decr_useonly(), vdropl(), and vinactive().

Referenced by change_root(), chflags(), chroot(), dounmount(), exit1(), extattr_delete_file(), extattr_delete_link(), extattr_get_file(), extattr_get_link(), extattr_list_file(), extattr_list_link(), extattr_set_file(), extattr_set_link(), extattrctl(), fchdir(), fdfree(), fhopen(), jail(), kern_alternate_path(), kern_chdir(), kern_chmod(), kern_chown(), kern_lchown(), kern_link(), kern_lutimes(), kern_mkdir(), kern_mkfifo(), kern_mknod(), kern_rename(), kern_rmdir(), kern_sendfile(), kern_symlink(), kern_truncate(), kern_unlink(), kern_utimes(), ktrace(), lchflags(), lchmod(), mountcheckdirs(), namei(), NDFREE(), prison_complete(), quotactl(), relookup(), set_rootvnode(), sysctl_kern_proc_pathname(), uipc_bind(), uipc_detach(), undelete(), vflush(), vfs_domount(), vfs_stdsync(), and vn_open_cred().

Here is the call graph for this function:

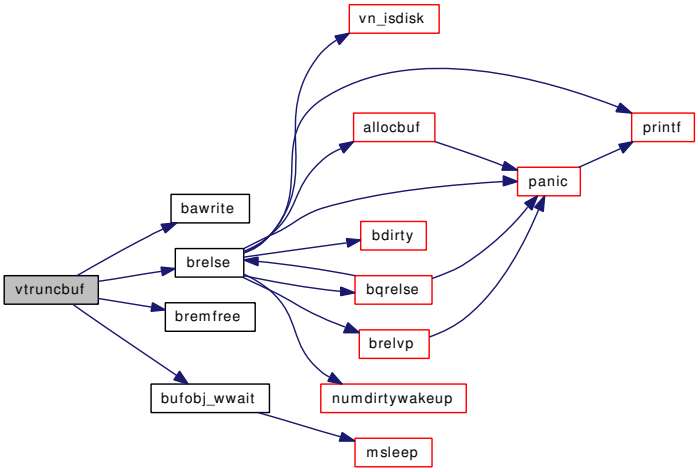


9.162.3.119 int vtruncbuf (struct vnode * *vp*, struct ucred * *cred*, struct thread * *td*, off_t *length*, int *blksize*)

Definition at line 1173 of file vfs_subr.c.

References bawrite(), brelse(), bremfree(), buf, and bufobj_wwait().

Here is the call graph for this function:



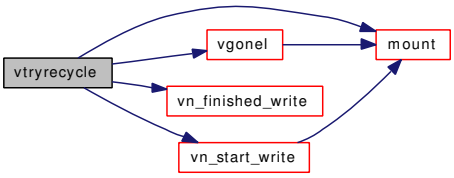
9.162.3.120 static int vtryrecycle (struct vnode * *vp*) [static]

Definition at line 818 of file vfs_subr.c.

References mount(), td, vgonel(), vn_finished_write(), and vn_start_write().

Referenced by vnlru_free().

Here is the call graph for this function:



9.162.4 Variable Documentation

9.162.4.1 int desiredvnodes

Definition at line 270 of file vfs_subr.c.

Referenced by LIST_HEAD(), and nchinit().

9.162.4.2 struct filterops fs_filtops

Initial value:

```
{ 0, filt_fsattach, filt_fsdetach, filt_fsevent }
```

Definition at line 3665 of file vfs_subr.c.

9.162.4.3 struct knlist fs_knlist [static]

Definition at line 3644 of file vfs_subr.c.

9.162.4.4 enum vtype iftovt_tab[16]

Initial value:

```
{
        VNON, VFIFO, VCHR, VNON, VDIR, VNON, VBLK, VNON,
        VREG, VNON, VLNK, VNON, VSOCK, VNON, VNON, VBAD,
}
```

Definition at line 140 of file vfs_subr.c.

9.162.4.5 int mpsafe_vfs = 1

Definition at line 119 of file vfs_subr.c.

9.162.4.6 unsigned long numvnodes [static]

Definition at line 132 of file vfs_subr.c.

Referenced by getnewvnode(), vdestroy(), and vnlru_proc().

9.162.4.7 struct vop_vector sync_vnodeops [static]

Initial value:

```
{
        .vop_bypass =   VOP_EOPNOTSUPP,
        .vop_close =    sync_close,
        .vop_fsync =    sync_fsync,
        .vop_inactive = sync_inactive,
        .vop_reclaim =  sync_reclaim,
        ._vop_lock =    vop_stdlock,
        .vop_unlock =   vop_stdunlock,
        .vop_islocked = vop_stdislocked,
}
```

Definition at line 2991 of file vfs_subr.c.

9.162.4.8 syncer_state

Definition at line 261 of file vfs_subr.c.

Referenced by sched_sync(), and syncer_shutdown().

9.162.4.9 int timestamp_precision = TSP_SEC [static]

Definition at line 469 of file vfs_subr.c.

9.162.4.10 char* typename[ ] [static]

Initial value:

```
{"VNON", "VREG", "VDIR", "VBLK", "VCHR", "VLNK", "VSOCK", "VFIFO", "VBAD",
 "VMARKER"}
```

Definition at line 2505 of file vfs_subr.c.

9.162.4.11 struct kproc_desc up_kp [static]

Initial value:

```
 {
        "syncer",
        sched_sync,
        &updateproc
}
```

Definition at line 1554 of file vfs_subr.c.

9.162.4.12 struct proc* updateproc [static]

Definition at line 1552 of file vfs_subr.c.

Referenced by sched_sync(), speedup_syncer(), and syncer_shutdown().

9.162.4.13 struct filterops vfsread_filtops [static]

Initial value:

```
        { 1, NULL, filt_vfsdetach, filt_vfsread }
```

Definition at line 3739 of file vfs_subr.c.

9.162.4.14 struct filterops vfsvnode_filtops [static]

Initial value:

```
        { 1, NULL, filt_vfsdetach, filt_vfsvnode }
```

Definition at line 3743 of file vfs_subr.c.

9.162.4.15 struct filterops vfswrite_filtops [static]

Initial value:

```
        { 1, NULL, filt_vfsdetach, filt_vfswrite }
```

Definition at line 3741 of file vfs_subr.c.

9.162.4.16 struct kproc_desc vnlru_kp [static]

Initial value:

```
{
        "vnlru",
        vnlru_proc,
        &vnlruproc
}
```

Definition at line 759 of file vfs_subr.c.

9.162.4.17 int vnlru_nowhere [static]

Definition at line 275 of file vfs_subr.c.

9.162.4.18 struct proc* vnlruproc [static]

Definition at line 694 of file vfs_subr.c.

9.162.4.19 int vnlruproc_sig [static]

Definition at line 695 of file vfs_subr.c.

9.162.4.20 int vttoif_tab[10]

Initial value:

```
{
        0, S_IFREG, S_IFDIR, S_IFBLK, S_IFCHR, S_IFLNK,
        S_IFSOCK, S_IFIFO, S_IFMT, S_IFMT
}
```

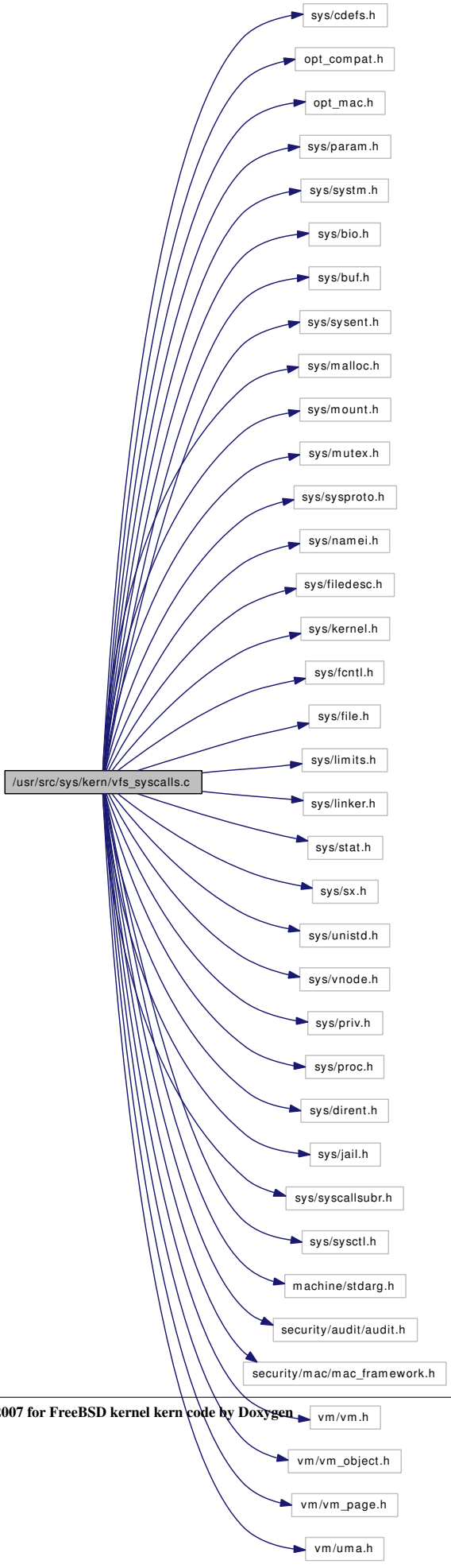
Definition at line 144 of file vfs_subr.c.

9.163 /usr/src/sys/kern/vfs_syscalls.c File Reference

```
#include <sys/cdefs.h>
#include "opt_compat.h"
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/bio.h>
#include <sys/buf.h>
#include <sys/sysent.h>
#include <sys/malloc.h>
#include <sys/mount.h>
#include <sys/mutex.h>
#include <sys/sysproto.h>
#include <sys/namei.h>
#include <sys/filedesc.h>
#include <sys/kernel.h>
#include <sys/fcntl.h>
#include <sys/file.h>
#include <sys/limits.h>
#include <sys/linker.h>
#include <sys/stat.h>
#include <sys/sx.h>
#include <sys/unistd.h>
#include <sys/vnode.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/dirent.h>
#include <sys/jail.h>
#include <sys/syscallsubr.h>
#include <sys/sysctl.h>
#include <machine/stdarg.h>
#include <security/audit/audit.h>
#include <security/mac/mac_framework.h>
#include <vm/vm.h>
#include <vm/vm_object.h>
#include <vm/vm_page.h>
```

```
#include <vm/uma.h>
```

Include dependency graph for vfs_syscalls.c:


/usr/src/sys/kern/vfs_syscalls.c
sys/cdefs.h
opt_compat.h
opt_mac.h
sys/param.h
sys/systm.h
sys/bio.h
sys/buf.h
sys/sysent.h
sys/malloc.h
sys/mount.h
sys/mutex.h
sys/sysproto.h
sys/namei.h
sys/filedesc.h
sys/kernel.h
sys/fcntl.h
sys/file.h
sys/limits.h
sys/linker.h
sys/stat.h
sys/sx.h
sys/unistd.h
sys/vnode.h
sys/priv.h
sys/proc.h
sys/dirent.h
sys/jail.h
sys/syscallsubr.h
sys/sysctl.h
machine/stdarg.h
security/audit/audit.h
security/mac/mac_framework.h
vm/vm.h
vm/vm_object.h
vm/vm_page.h
vm/uma.h

Data Structures

- struct sync_args
- struct quotactl_args
- struct statfs_args
- struct fstatfs_args
- struct getfsstat_args
- struct fchdir_args
- struct chdir_args
- struct chroot_args
- struct open_args
- struct mknod_args
- struct mkfifo_args
- struct link_args
- struct symlink_args
- struct unlink_args
- struct lseek_args
- struct access_args
- struct eaccess_args
- struct stat_args
- struct lstat_args
- struct nstat_args
- struct lstat_args
- struct pathconf_args
- struct readlink_args
- struct chflags_args
- struct fchflags_args
- struct chmod_args
- struct lchmod_args
- struct fchmod_args
- struct chown_args
- struct lchown_args
- struct fchown_args
- struct utimes_args
- struct lutimes_args
- struct futimes_args
- struct truncate_args
- struct ftruncate_args
- struct fsync_args
- struct rename_args
- struct mkdir_args
- struct rmdir_args
- struct getdirentries_args
- struct getdents_args
- struct umask_args
- struct revoke_args
- struct lgetfh_args
- struct getfh_args
- struct fhopen_args
- struct fhstat_args
- struct fhstatfs_args

Functions

- __FBSDID ("$FreeBSD: src/sys/kern/vfs_syscalls.c,v 1.431 2007/02/19 10:56:09 kib Exp $")
- static int chroot_refuse_vdir_fds (struct filedesc *fdp)
- static int getutimes (const struct timeval *, enum uio_seg, struct timespec *)
- static int setfown (struct thread *td, struct vnode *, uid_t, gid_t)
- static int setfmode (struct thread *td, struct vnode *, int)
- static int setfflags (struct thread *td, struct vnode *, int)
- static int setutimes (struct thread *td, struct vnode *, const struct timespec *, int, int)
- static int vn_access (struct vnode *vp, int user_flags, struct ucred *cred, struct thread *td)
- int sync (struct thread *td, struct sync_args *uap)
- int quotactl (struct thread *td, struct quotactl_args *uap)
- int statfs (struct thread *td, struct statfs_args *uap)
- int kern_statfs (struct thread *td, char *path, enum uio_seg pathseg, struct statfs *buf)
- int fstatfs (struct thread *td, struct fstatfs_args *uap)
- int kern_fstatfs (struct thread *td, int fd, struct statfs *buf)
- int getfsstat (struct thread *td, struct getfsstat_args *uap)
- int kern_getfsstat (struct thread *td, struct statfs **buf, size_t bufsize, enum uio_seg bufseg, int flags)
- int fchdir (struct thread *td, struct fchdir_args *uap)
- int chdir (struct thread *td, struct chdir_args *uap)
- int kern_chdir (struct thread *td, char *path, enum uio_seg pathseg)
- SYSCTL_INT (_kern, OID_AUTO, chroot_allow_open_directories, CTLFLAG_RW,&chroot_allow_open_directories, 0,"")
- int chroot (struct thread *td, struct chroot_args *uap)
- int change_dir (struct vnode *vp, struct thread *td)
- int change_root (struct vnode *vp, struct thread *td)
- int open (struct thread *td, struct open_args *uap)
- int kern_open (struct thread *td, char *path, enum uio_seg pathseg, int flags, int mode)
- int mknod (struct thread *td, struct mknod_args *uap)
- int kern_mknod (struct thread *td, char *path, enum uio_seg pathseg, int mode, int dev)
- int mkfifo (struct thread *td, struct mkfifo_args *uap)
- int kern_mkfifo (struct thread *td, char *path, enum uio_seg pathseg, int mode)
- int link (struct thread *td, struct link_args *uap)
- SYSCTL_INT (_security_bsd, OID_AUTO, hardlink_check_uid, CTLFLAG_RW,&hardlink_check_uid, 0,"Unprivileged processes cannot create hard links to files owned by other ""users")
- SYSCTL_INT (_security_bsd, OID_AUTO, hardlink_check_gid, CTLFLAG_RW,&hardlink_check_gid, 0,"Unprivileged processes cannot create hard links to files owned by other ""groups")
- static int can_hardlink (struct vnode *vp, struct thread *td, struct ucred *cred)
- int kern_link (struct thread *td, char *path, char *link, enum uio_seg segflg)
- int symlink (struct thread *td, struct symlink_args *uap)
- int kern_symlink (struct thread *td, char *path, char *link, enum uio_seg segflg)
- int undelete (struct thread *td, struct undelete_args *uap)
- int unlink (struct thread *td, struct unlink_args *uap)
- int kern_unlink (struct thread *td, char *path, enum uio_seg pathseg)
- int lseek (struct thread *td, struct lseek_args *uap)
- int access (struct thread *td, struct access_args *uap)
- int kern_access (struct thread *td, char *path, enum uio_seg pathseg, int flags)
- int eaccess (struct thread *td, struct eaccess_args *uap)
- int kern_eaccess (struct thread *td, char *path, enum uio_seg pathseg, int flags)
- int stat (struct thread *td, struct stat_args *uap)
- int kern_stat (struct thread *td, char *path, enum uio_seg pathseg, struct stat *sbp)

- int lstat (struct thread *td, struct lstat_args *uap)
- int kern_lstat (struct thread *td, char *path, enum uio_seg pathseg, struct stat *sbp)
- void cvtnstat (struct stat *sb, struct nstat *nsb)
- int nstat (struct thread *td, struct nstat_args *uap)
- int nlstat (struct thread *td, struct nlstat_args *uap)
- int pathconf (struct thread *td, struct pathconf_args *uap)
- int kern_pathconf (struct thread *td, char *path, enum uio_seg pathseg, int name)
- int readlink (struct thread *td, struct readlink_args *uap)
- int kern_readlink (struct thread *td, char *path, enum uio_seg pathseg, char *buf, enum uio_seg bufseg, int count)
- int chflags (struct thread *td, struct chflags_args *uap)
- int lchflags (struct thread *td, struct lchflags_args *uap)
- int fchflags (struct thread *td, struct fchflags_args *uap)
- int chmod (struct thread *td, struct chmod_args *uap)
- int kern_chmod (struct thread *td, char *path, enum uio_seg pathseg, int mode)
- int lchmod (struct thread *td, struct lchmod_args *uap)
- int fchmod (struct thread *td, struct fchmod_args *uap)
- int chown (struct thread *td, struct chown_args *uap)
- int kern_chown (struct thread *td, char *path, enum uio_seg pathseg, int uid, int gid)
- int lchown (struct thread *td, struct lchown_args *uap)
- int kern_lchown (struct thread *td, char *path, enum uio_seg pathseg, int uid, int gid)
- int fchown (struct thread *td, struct fchown_args *uap)
- static int getutimes (struct timeval *usrtvp, enum uio_seg tvpseg, struct timespec *tsp) const
- int utimes (struct thread *td, struct utimes_args *uap)
- int kern_utimes (struct thread *td, char *path, enum uio_seg pathseg, struct timeval *tptr, enum uio_seg tptrseg)
- int lutimes (struct thread *td, struct lutimes_args *uap)
- int kern_lutimes (struct thread *td, char *path, enum uio_seg pathseg, struct timeval *tptr, enum uio_seg tptrseg)
- int futimes (struct thread *td, struct futimes_args *uap)
- int kern_futimes (struct thread *td, int fd, struct timeval *tptr, enum uio_seg tptrseg)
- int truncate (struct thread *td, struct truncate_args *uap)
- int kern_truncate (struct thread *td, char *path, enum uio_seg pathseg, off_t length)
- int ftruncate (struct thread *td, struct ftruncate_args *uap)
- int fsync (struct thread *td, struct fsync_args *uap)
- int rename (struct thread *td, struct rename_args *uap)
- int kern_rename (struct thread *td, char *from, char *to, enum uio_seg pathseg)
- int mkdir (struct thread *td, struct mkdir_args *uap)
- int kern_mkdir (struct thread *td, char *path, enum uio_seg segflg, int mode)
- int rmdir (struct thread *td, struct rmdir_args *uap)
- int kern_rmdir (struct thread *td, char *path, enum uio_seg pathseg)
- int getdirentries (struct thread *td, struct getdirentries_args *uap)
- int getdents (struct thread *td, struct getdents_args *uap)
- int umask (struct thread *td, struct umask_args *uap)
- int revoke (struct thread *td, struct revoke_args *uap)
- int getvnode (struct filedesc *fdp, int fd, struct file **fpp)
- int lgetfh (struct thread *td, struct lgetfh_args *uap)
- int getfh (struct thread *td, struct getfh_args *uap)
- int fhopen (struct thread *td, struct fhopen_args *uap)
- int fhstat (struct thread *td, struct fhstat_args *uap)
- int fhstatfs (struct thread *td, struct fhstatfs_args *uap)
- int kern_fhstatfs (struct thread *td, fhandle_t fh, struct statfs *buf)

Variables

- int async_io_version
- static int prison_quotas
- static int chroot_allow_open_directories = 1
- static int hardlink_check_uid = 0
- static int hardlink_check_gid = 0

9.163.1 Function Documentation

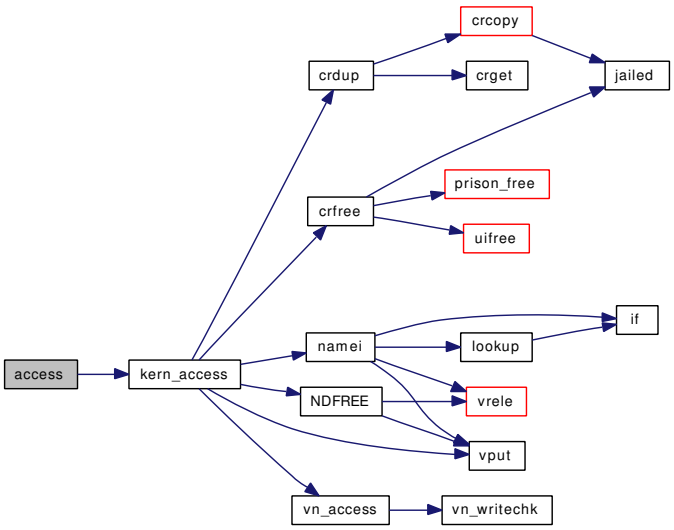
9.163.1.1 __FBSDID ("$FreeBSD: src/sys/kern/vfs_syscalls. *c*, v 1.431 2007/02/19 10:56:09 kib Exp $")

9.163.1.2 int access (struct thread * *td*, struct access_args * *uap*)

Definition at line 1870 of file vfs_syscalls.c.

References kern_access().

Here is the call graph for this function:



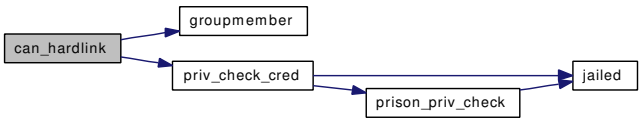
9.163.1.3 static int can_hardlink (struct vnode * *vp*, struct thread * *td*, struct ucred * *cred*) [static]

Definition at line 1389 of file vfs_syscalls.c.

References groupmember(), and priv_check_cred().

Referenced by kern_link().

Here is the call graph for this function:



9.163.1.4 int change_dir (struct vnode * *vp*, struct thread * *td*)

Definition at line 875 of file vfs_syscalls.c.

Referenced by chroot(), fchdir(), jail_attach(), and kern_chdir().

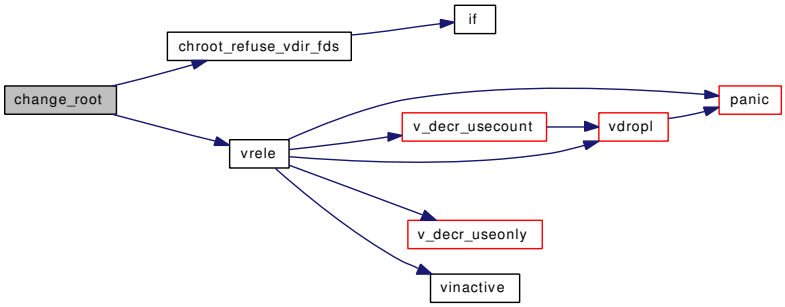
9.163.1.5 int change_root (struct vnode * *vp*, struct thread * *td*)

Definition at line 899 of file vfs_syscalls.c.

References chroot_refuse_vdir_fds(), rootvnode, and vrele().

Referenced by chroot(), and jail_attach().

Here is the call graph for this function:

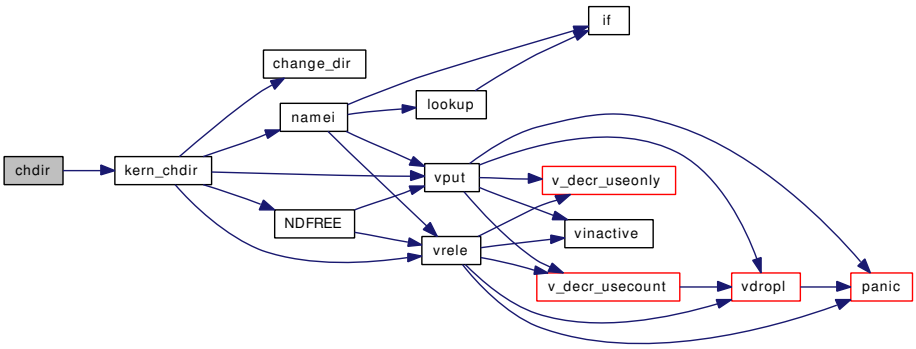


9.163.1.6 int chdir (struct thread * *td*, struct chdir_args * *uap*)

Definition at line 739 of file vfs_syscalls.c.

References kern_chdir(), and chdir_args::path.

Here is the call graph for this function:

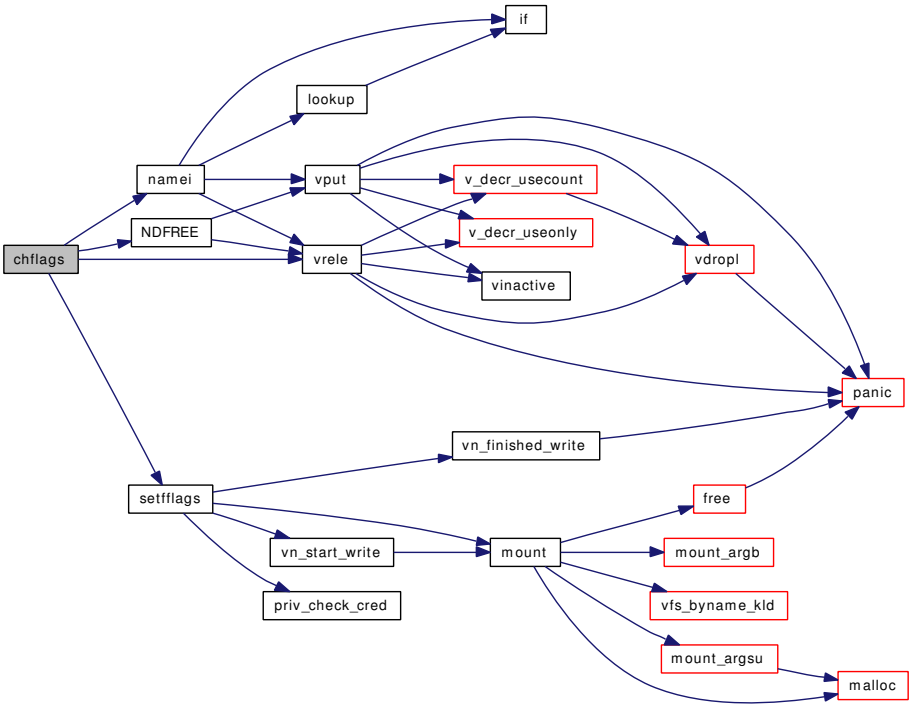


9.163.1.7 int chflags (struct thread * *td*, struct chflags_args * *uap*)

Definition at line 2398 of file vfs_syscalls.c.

References namei(), NDFREE(), setfflags(), and vrele().

Here is the call graph for this function:

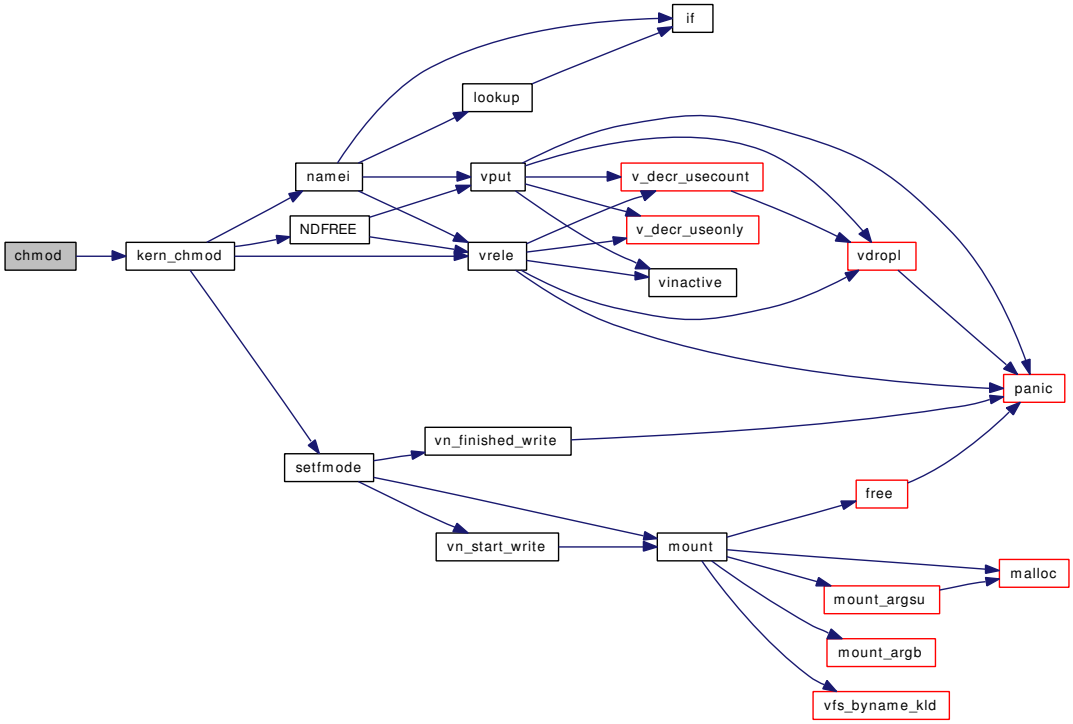


9.163.1.8 int chmod (struct thread * *td*, struct chmod_args * *uap*)

Definition at line 2526 of file vfs_syscalls.c.

References kern_chmod().

Here is the call graph for this function:

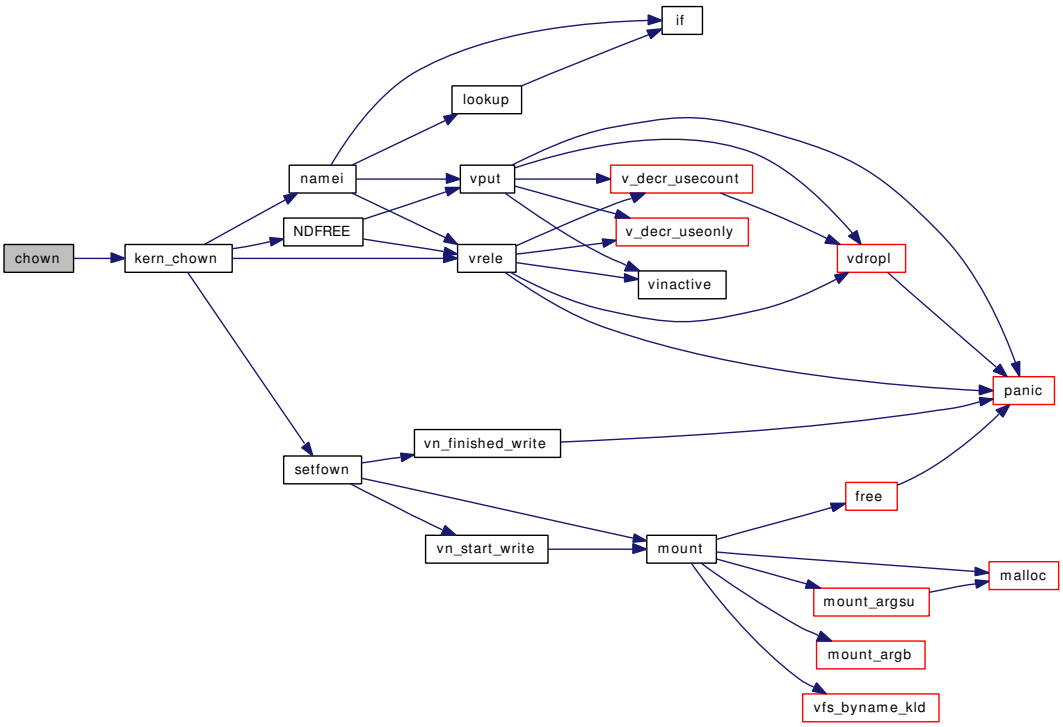


9.163.1.9 int chown (struct thread * *td*, struct chown_args * *uap*)

Definition at line 2670 of file vfs_syscalls.c.

References kern_chown().

Here is the call graph for this function:

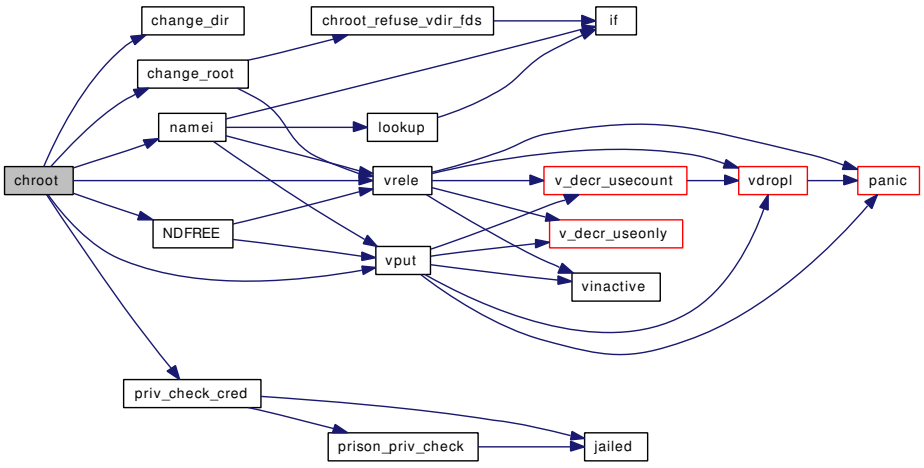


9.163.1.10 int chroot (struct thread * *td*, struct chroot_args * *uap*)

Definition at line 830 of file vfs_syscalls.c.

References change_dir(), change_root(), namei(), NDFREE(), chroot_args::path, priv_check_cred(), vput(), and vrele().

Here is the call graph for this function:



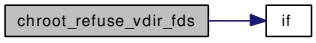
9.163.1.11 static int chroot_refuse_vdir_fds (struct filedesc * *fdp*) [static]

Definition at line 787 of file vfs_syscalls.c.

References if().

Referenced by change_root().

Here is the call graph for this function:



9.163.1.12 void cvtnstat (struct stat * *sb*, struct nstat * *nsb*)

Definition at line 2154 of file vfs_syscalls.c.

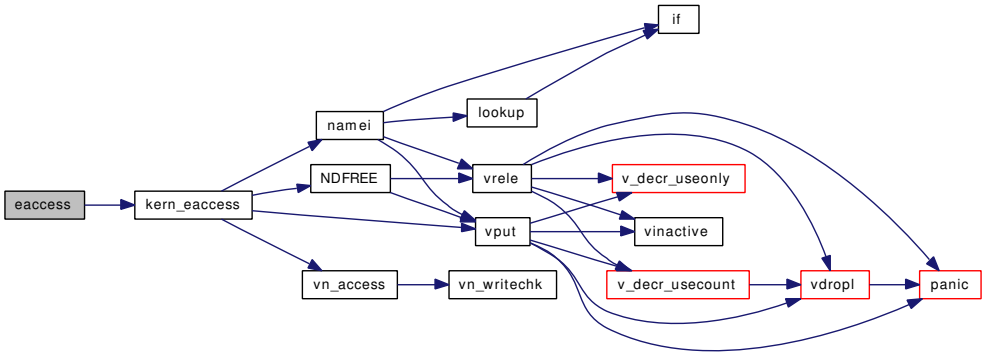
Referenced by nfstat(), nlstat(), and nstat().

9.163.1.13 int eaccess (struct thread * *td*, struct eaccess_args * *uap*)

Definition at line 1927 of file vfs_syscalls.c.

References kern_eaccess().

Here is the call graph for this function:

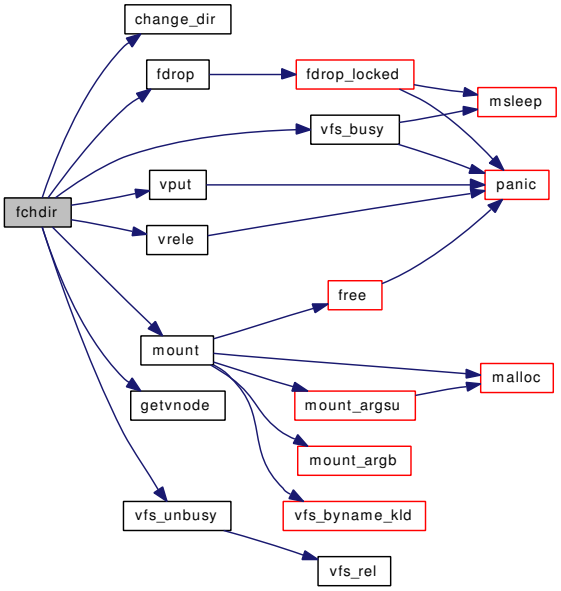


9.163.1.14 int fchdir (struct thread * *td*, struct fchdir_args * *uap*)

Definition at line 674 of file vfs_syscalls.c.

References change_dir(), fchdir_args::fd, fdrop(), getvnode(), mount(), vfs_busy(), vfs_unbusy(), vput(), and vrele().

Here is the call graph for this function:

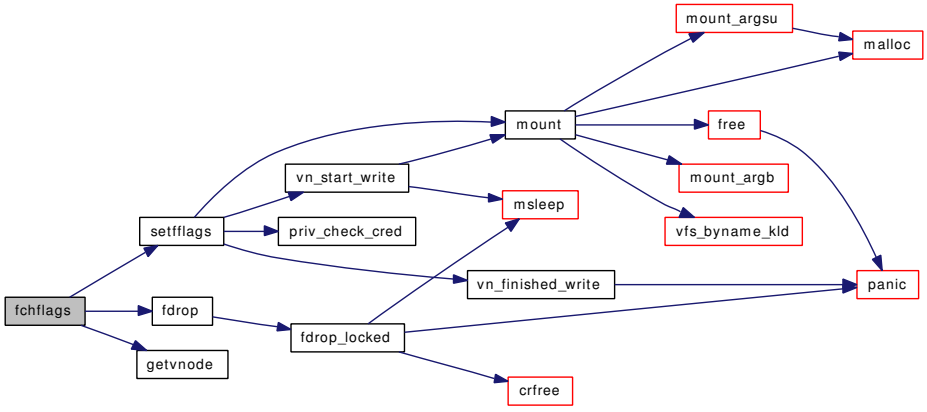


9.163.1.15 int fchflags (struct thread * *td*, struct fchflags_args * *uap*)

Definition at line 2460 of file vfs_syscalls.c.

References fdrop(), getvnode(), and setfflags().

Here is the call graph for this function:

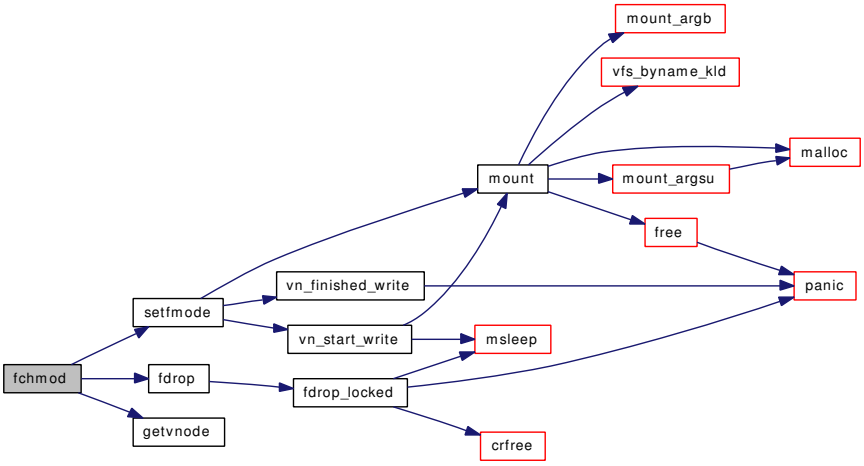


9.163.1.16 int fchmod (struct thread * *td*, struct fchmod_args * *uap*)

Definition at line 2600 of file vfs_syscalls.c.

References fdrop(), getvnode(), and setfmode().

Here is the call graph for this function:

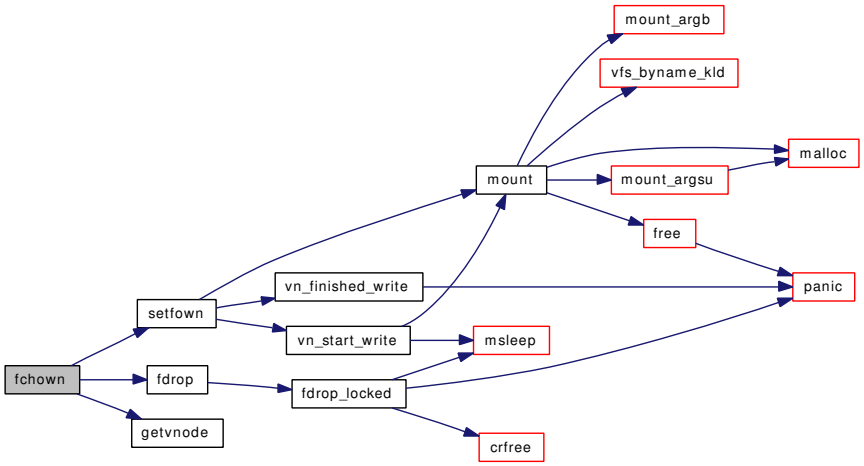


9.163.1.17 int fchown (struct thread * *td*, struct fchown_args * *uap*)

Definition at line 2756 of file vfs_syscalls.c.

References fdrop(), getvnode(), and setfown().

Here is the call graph for this function:

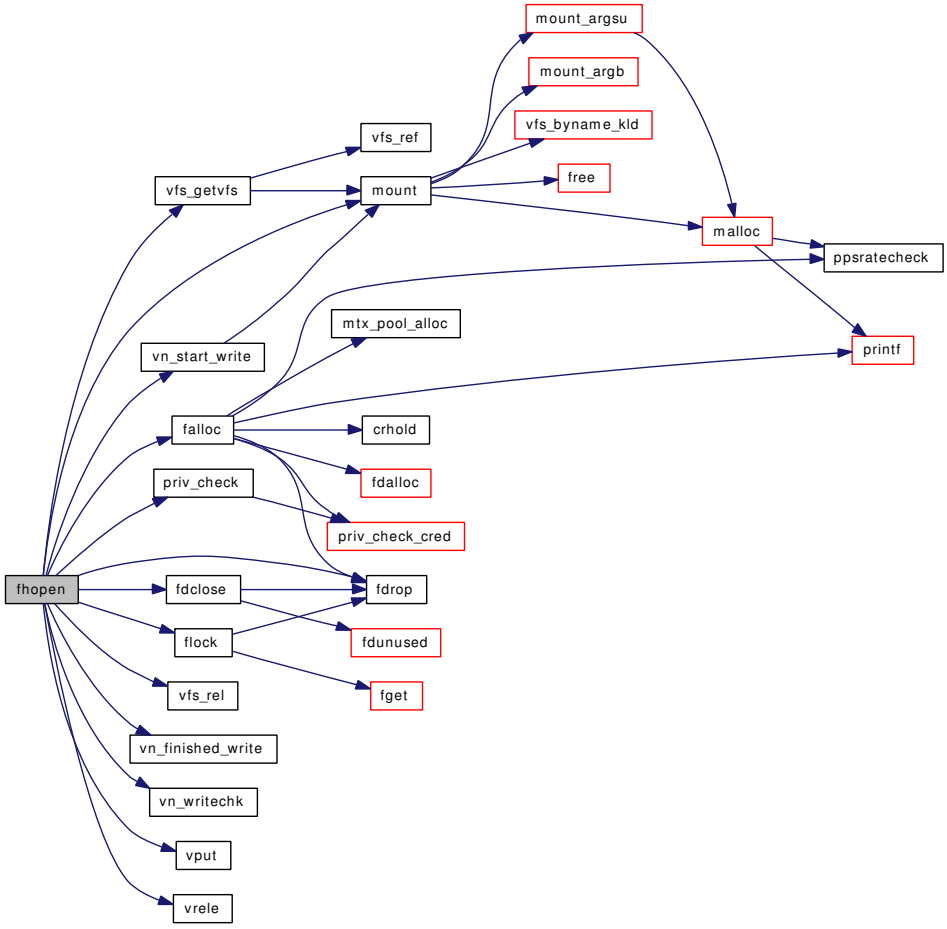


9.163.1.18 int fhopen (struct thread * *td*, struct fhopen_args * *uap*)

Definition at line 4014 of file vfs_syscalls.c.

References falloc(), fdclose(), fdrop(), fhopen_args::flags, flock(), mount(), priv_check(), fhopen_args::u_-fhp, vfs_getvfs(), vfs_rel(), vn_finished_write(), vn_start_write(), vn_writechk(), vnops, vput(), and vrele().

Here is the call graph for this function:

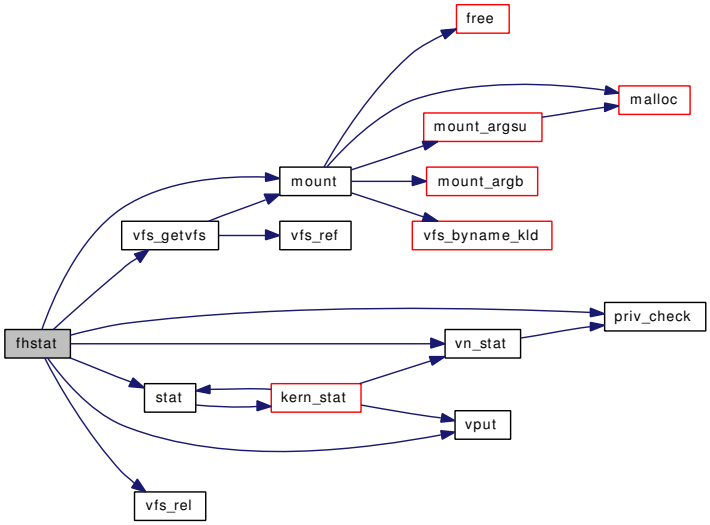


9.163.1.19 int fhstat (struct thread * *td*, struct fhstat_args * *uap*)

Definition at line 4205 of file vfs_syscalls.c.

References mount(), priv_check(), stat(), vfs_getvfs(), vfs_rel(), vn_stat(), and vput().

Here is the call graph for this function:

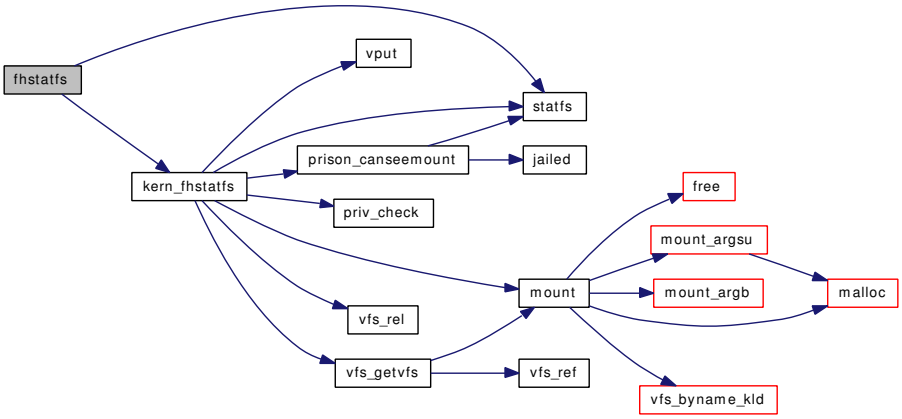


9.163.1.20 int fhstatfs (struct thread * *td*, struct fhstatfs_args * *uap*)

Definition at line 4255 of file vfs_syscalls.c.

References fhstatfs_args::buf, kern_fhstatfs(), statfs(), and fhstatfs_args::u_fhp.

Here is the call graph for this function:

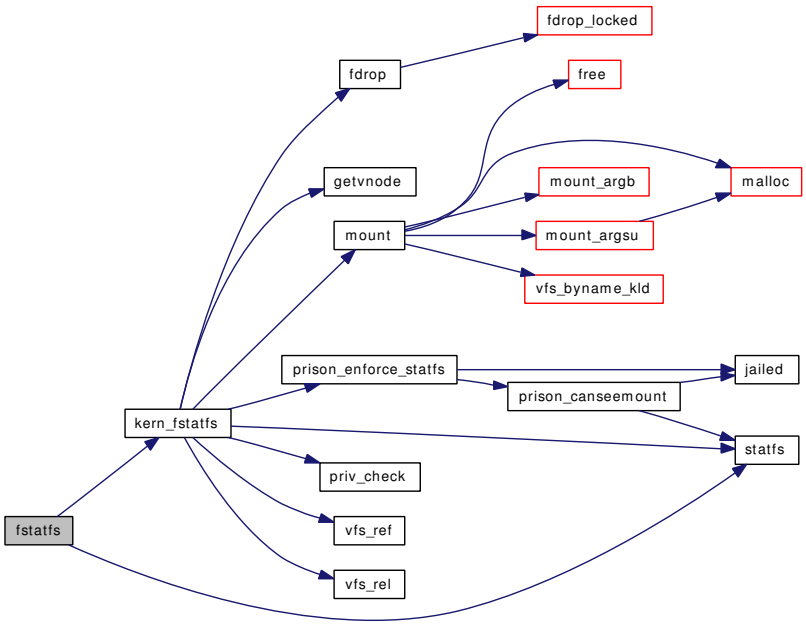


9.163.1.21 int fstatfs (struct thread * *td*, struct fstatfs_args * *uap*)

Definition at line 295 of file vfs_syscalls.c.

References kern_fstatfs(), and statfs().

Here is the call graph for this function:

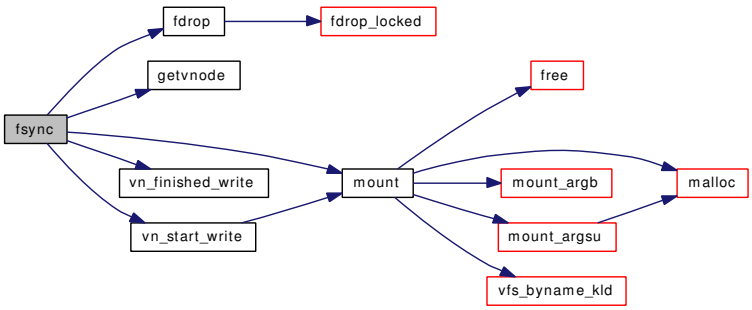


9.163.1.22 int fsync (struct thread * *td*, struct fsync_args * *uap*)

Definition at line 3192 of file vfs_syscalls.c.

References fsync_args::fd, fdrop(), getvnode(), mount(), vn_finished_write(), and vn_start_write().

Here is the call graph for this function:

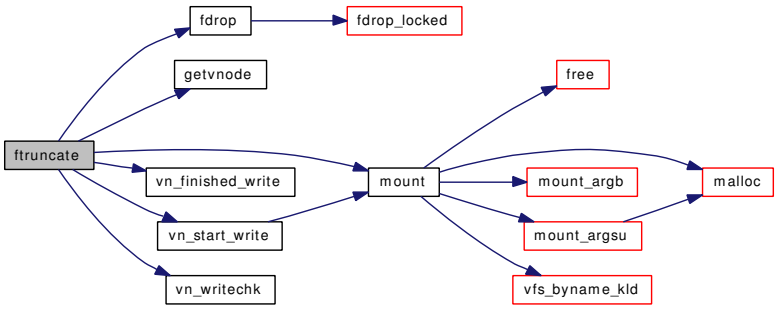


9.163.1.23 int ftruncate (struct thread * *td*, struct ftruncate_args * *uap*)

Definition at line 3074 of file vfs_syscalls.c.

References fdrop(), getvnode(), mount(), vn_finished_write(), vn_start_write(), and vn_writechk().

Here is the call graph for this function:

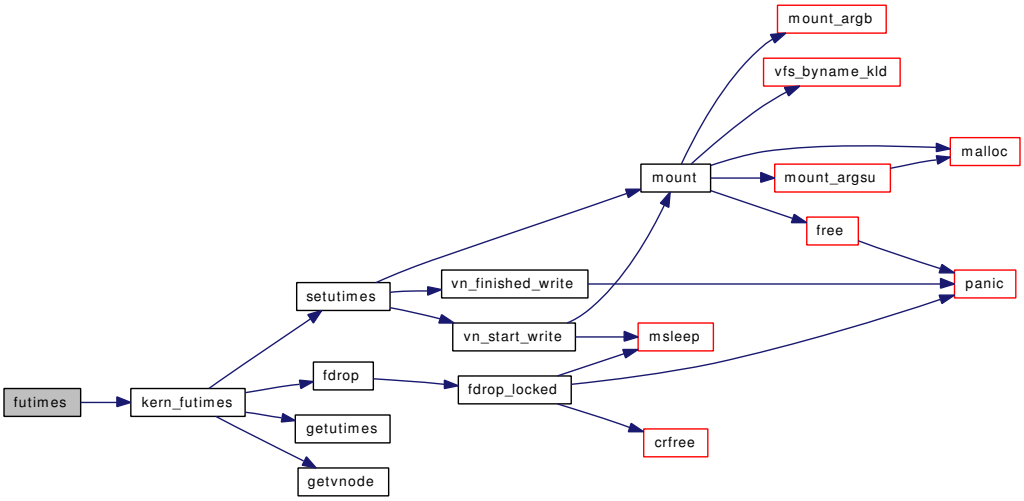


9.163.1.24 int futimes (struct thread * *td*, struct futimes_args * *uap*)

Definition at line 2960 of file vfs_syscalls.c.

References kern_futimes().

Here is the call graph for this function:

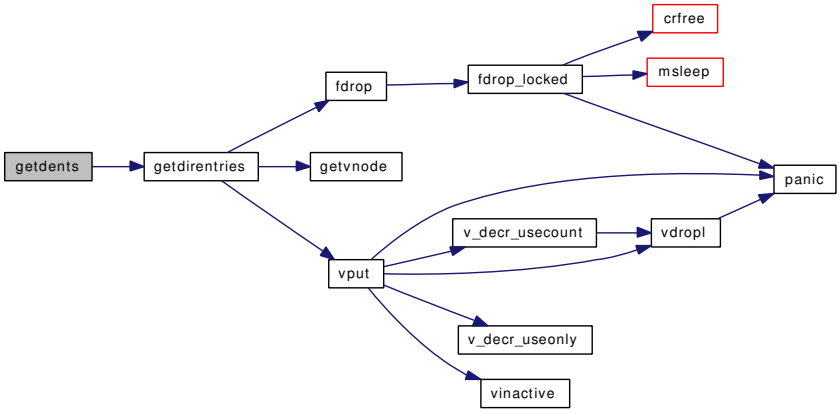


9.163.1.25 int getdents (struct thread * *td*, struct getdents_args * *uap*)

Definition at line 3785 of file vfs_syscalls.c.

References getdirentries_args::basep, getdirentries_args::buf, getdirentries_args::count, getdirentries_-args::fd, and getdirentries().

Here is the call graph for this function:



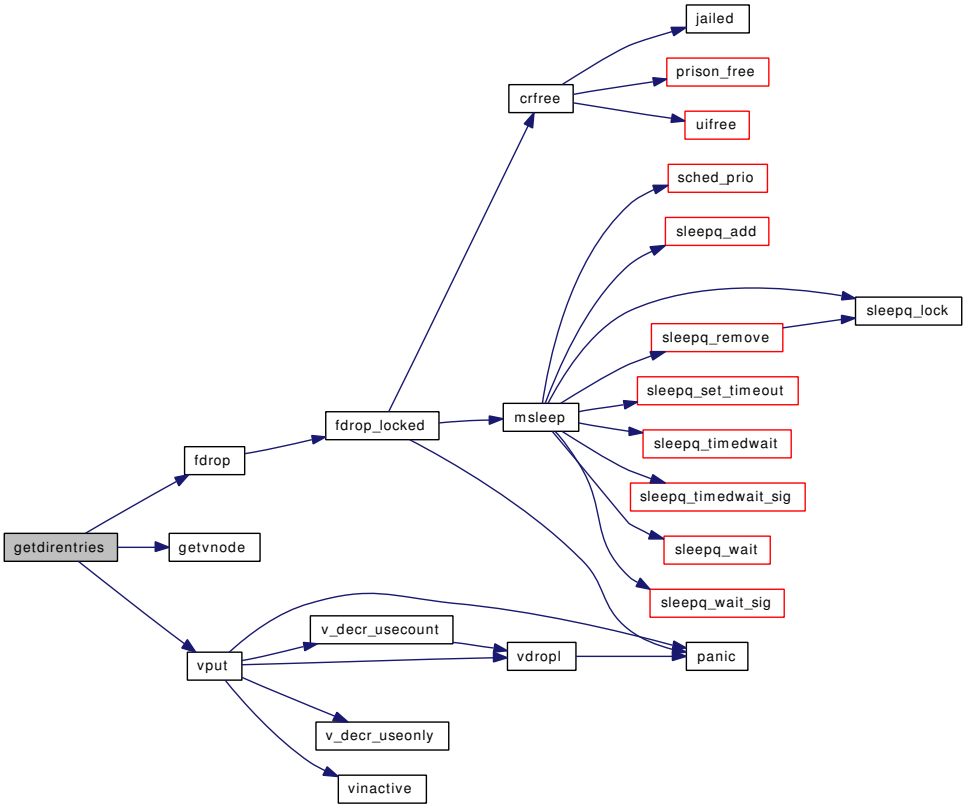
9.163.1.26 int getdirentries (struct thread * *td*, struct getdirentries_args * *uap*)

Definition at line 3698 of file vfs_syscalls.c.

References fdrop(), getvnode(), and vput().

Referenced by getdents().

Here is the call graph for this function:

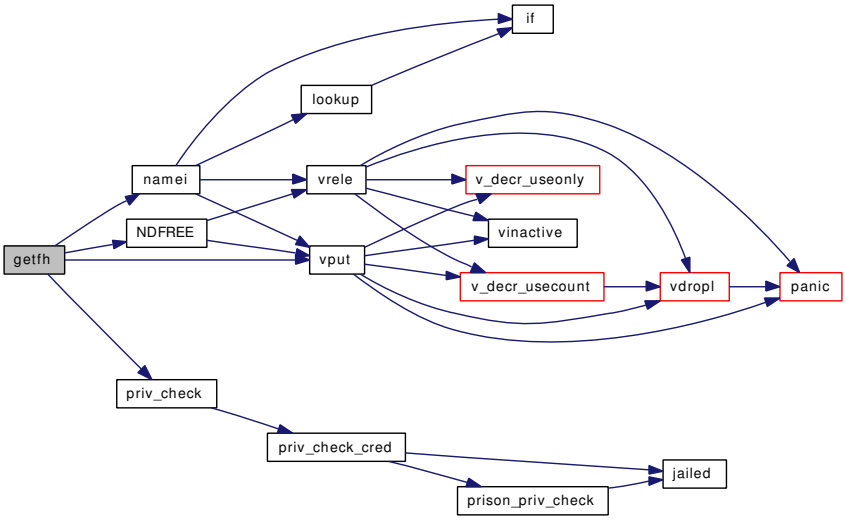


9.163.1.27 int getfh (struct thread * *td*, struct getfh_args * *uap*)

Definition at line 3966 of file vfs_syscalls.c.

References namei(), NDFREE(), priv_check(), and vput().

Here is the call graph for this function:

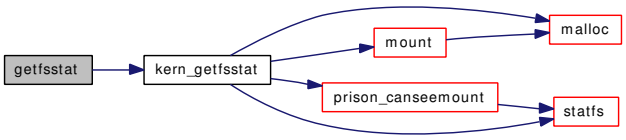


9.163.1.28 int getfsstat (struct thread * *td*, struct getfsstat_args * *uap*)

Definition at line 380 of file vfs_syscalls.c.

References kern_getfsstat().

Here is the call graph for this function:

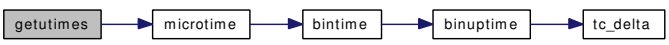


9.163.1.29 static int getutimes (struct timeval * *usrtvp*, enum uio_seg *tvpseg*, struct timespec * *tsp*) const [static]

Definition at line 2788 of file vfs_syscalls.c.

References microtime().

Here is the call graph for this function:



9.163.1.30 static int getutimes (const struct timeval *, enum *uio_seg*, struct timespec *) [static]

Referenced by kern_futimes(), kern_lutimes(), and kern_utimes().

9.163.1.31 int getvnode (struct filedesc * *fdp*, int *fd*, struct file ** *fpp*)

Definition at line 3888 of file vfs_syscalls.c.

Referenced by __acl_aclcheck_fd(), __acl_delete_fd(), __acl_get_fd(), __acl_set_fd(), extattr_delete_fd(), extattr_get_fd(), extattr_list_fd(), extattr_set_fd(), fchdir(), fchflags(), fchmod(), fchown(), fsync(), ftruncate(), getdirentries(), kern_fstatfs(), and kern_futimes().

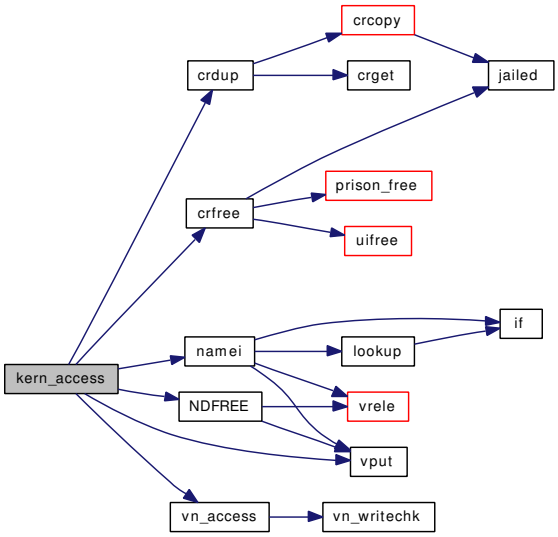
9.163.1.32 int kern_access (struct thread * *td*, char * *path*, enum uio_seg *pathseg*, int *flags*)

Definition at line 1882 of file vfs_syscalls.c.

References crdup(), crfree(), namei(), NDFREE(), vn_access(), and vput().

Referenced by access().

Here is the call graph for this function:



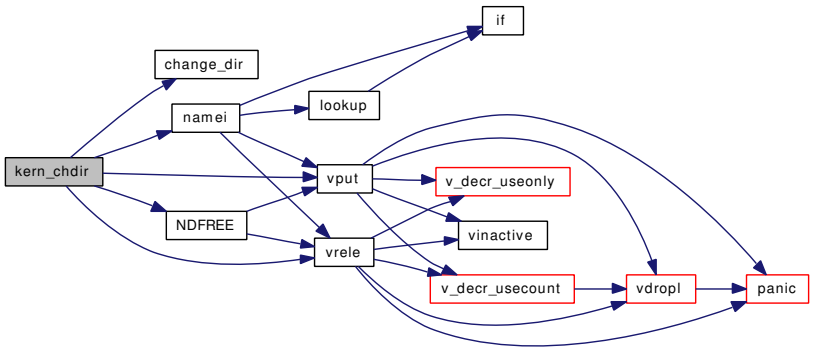
9.163.1.33 int kern_chdir (struct thread * *td*, char * *path*, enum uio_seg *pathseg*)

Definition at line 750 of file vfs_syscalls.c.

References change_dir(), namei(), NDFREE(), vput(), and vrele().

Referenced by chdir().

Here is the call graph for this function:



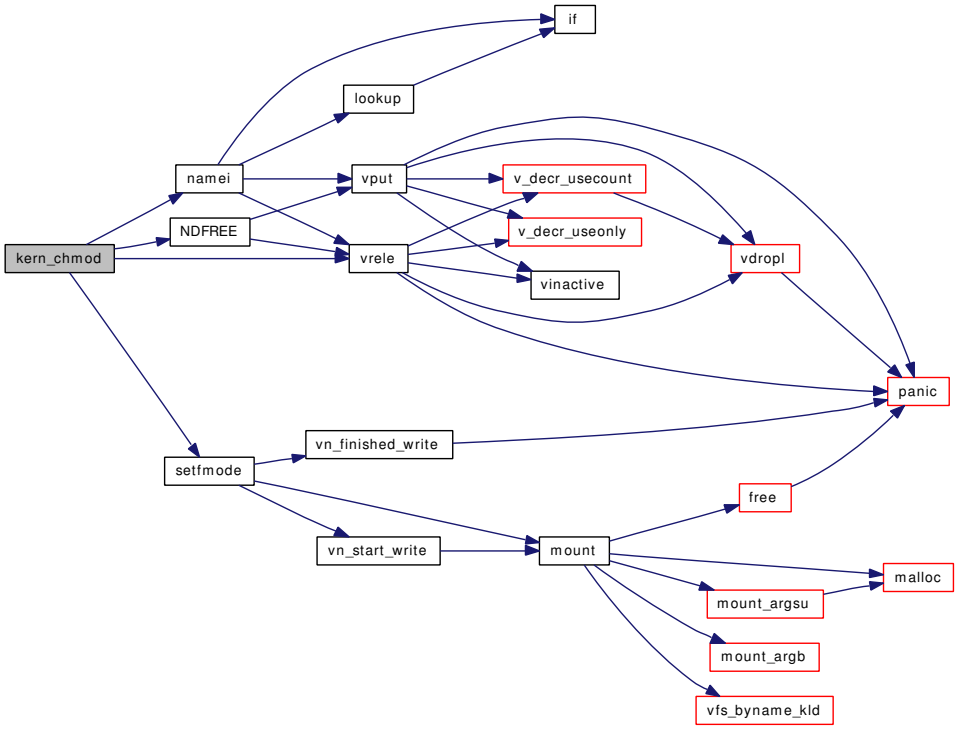
9.163.1.34 int kern_chmod (struct thread * *td*, char * *path*, enum uio_seg *pathseg*, int *mode*)

Definition at line 2538 of file vfs_syscalls.c.

References namei(), NDFREE(), setfmode(), and vrele().

Referenced by chmod().

Here is the call graph for this function:



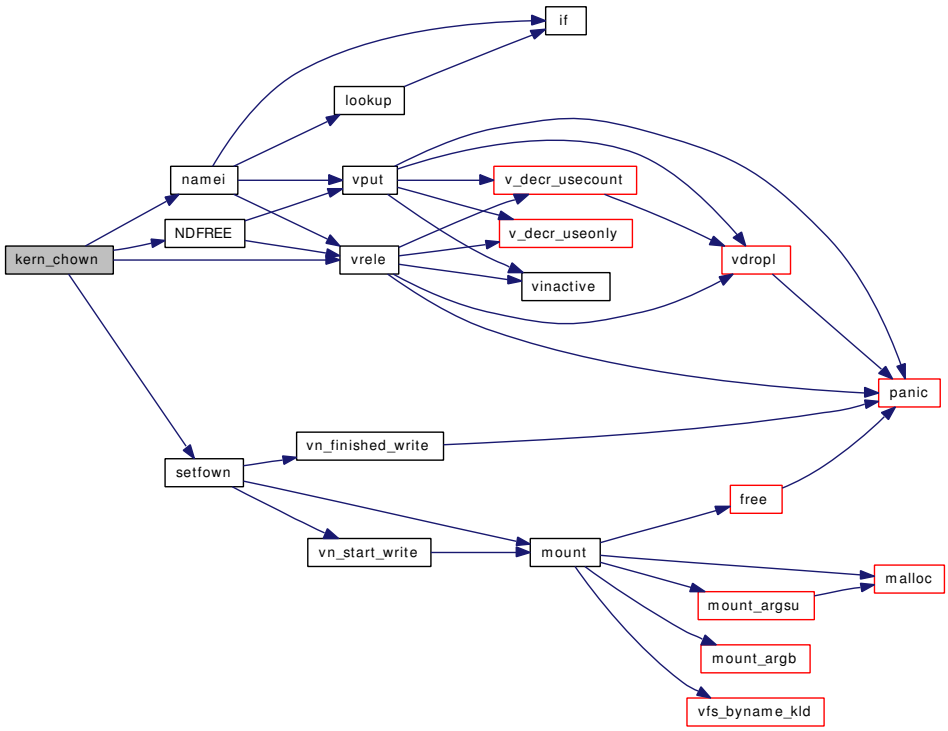
9.163.1.35 int kern_chown (struct thread * *td*, char * *path*, enum uio_seg *pathseg*, int *uid*, int *gid*)

Definition at line 2683 of file vfs_syscalls.c.

References namei(), NDFREE(), setfown(), and vrele().

Referenced by chown().

Here is the call graph for this function:



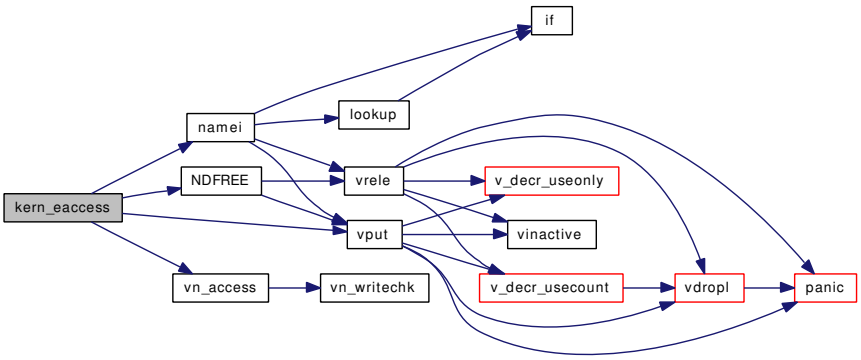
9.163.1.36 int kern_eaccess (struct thread * *td*, char * *path*, enum uio_seg *pathseg*, int *flags*)

Definition at line 1939 of file vfs_syscalls.c.

References namei(), NDFREE(), vn_access(), and vput().

Referenced by eaccess().

Here is the call graph for this function:



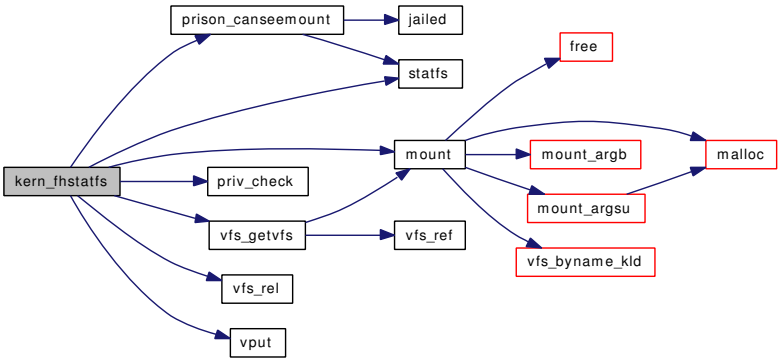
9.163.1.37 int kern_fhstatfs (struct thread * *td*, fhandle_t *fh*, struct statfs * *buf*)

Definition at line 4276 of file vfs_syscalls.c.

References mount(), prison_canseemount(), priv_check(), statfs(), vfs_getvfs(), vfs_rel(), and vput().

Referenced by fhstatfs().

Here is the call graph for this function:



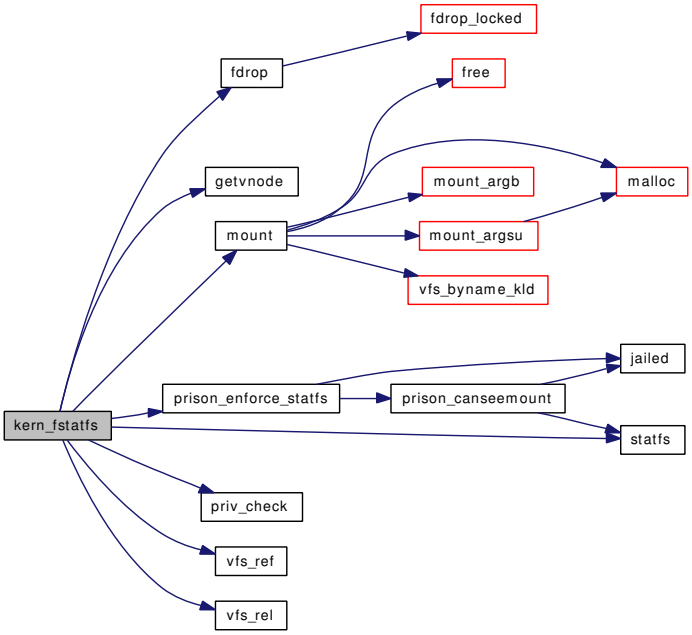
9.163.1.38 int kern_fstatfs (struct thread * *td*, int *fd*, struct statfs * *buf*)

Definition at line 312 of file vfs_syscalls.c.

References fdrop(), getvnode(), mount(), prison_enforce_statfs(), priv_check(), statfs(), vfs_ref(), and vfs_rel().

Referenced by fstatfs().

Here is the call graph for this function:



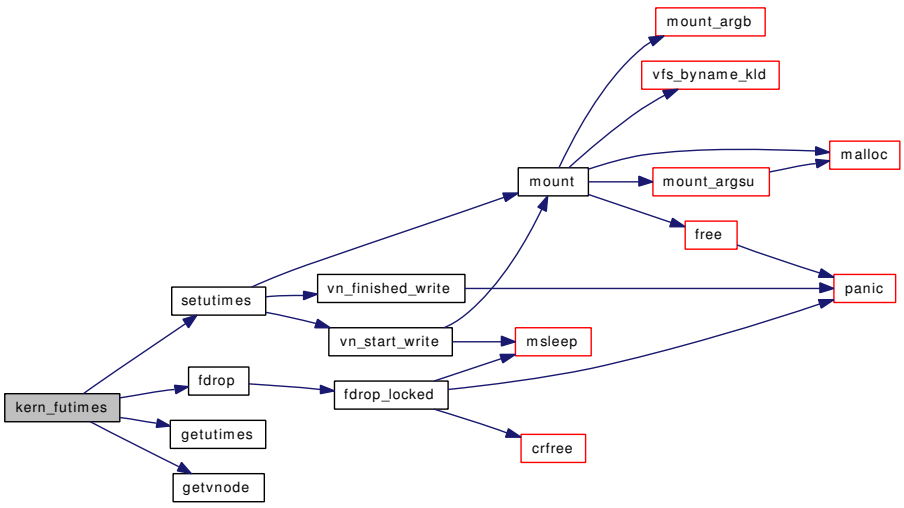
9.163.1.39 int kern_futimes (struct thread * *td*, int *fd*, struct timeval * *tptr*, enum uio_seg *tptrseg*)

Definition at line 2972 of file vfs_syscalls.c.

References fdrop(), getutimes(), getvnode(), and setutimes().

Referenced by futimes().

Here is the call graph for this function:



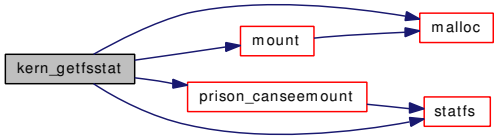
9.163.1.40 int kern_getfsstat (struct thread * *td*, struct statfs ** *buf*, size_t *bufsize*, enum uio_seg *bufseg*, int *flags*)

Definition at line 399 of file vfs_syscalls.c.

References malloc(), mount(), mountlist, mountlist_mtx, prison_canseemount(), and statfs().

Referenced by getfsstat().

Here is the call graph for this function:



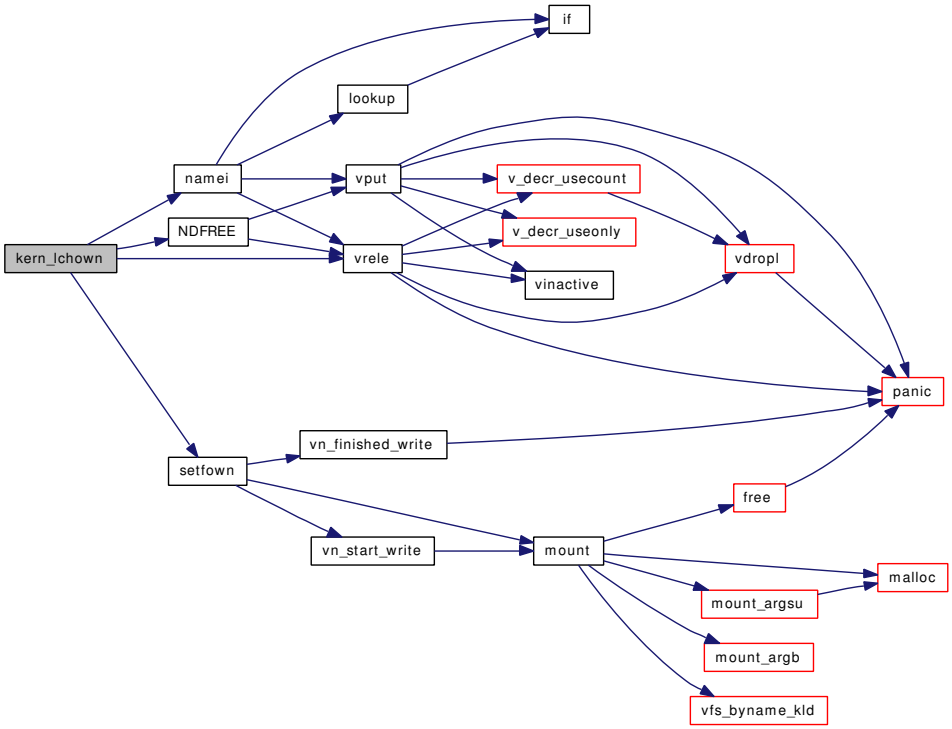
9.163.1.41 int kern_lchown (struct thread * *td*, char * *path*, enum uio_seg *pathseg*, int *uid*, int *gid*)

Definition at line 2726 of file vfs_syscalls.c.

References namei(), NDFREE(), setfown(), and vrele().

Referenced by lchown().

Here is the call graph for this function:



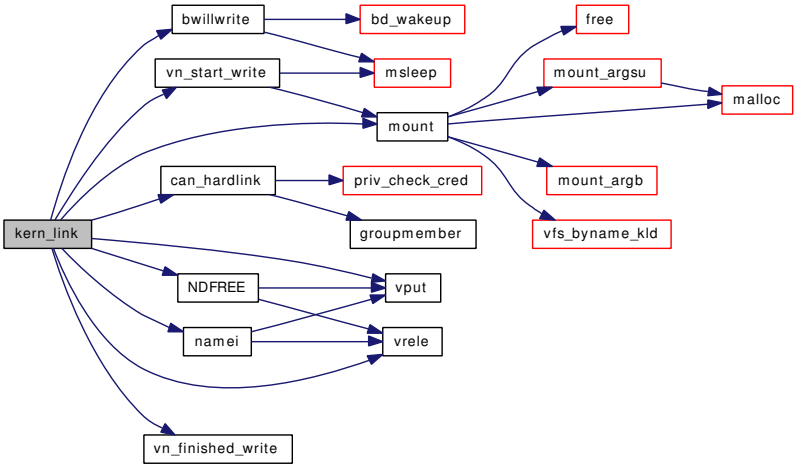
9.163.1.42 int kern_link (struct thread * *td*, char * *path*, char * *link*, enum uio_seg *segflg*)

Definition at line 1419 of file vfs_syscalls.c.

References bwillwrite(), can_hardlink(), mount(), namei(), NDFREE(), vn_finished_write(), vn_start_-write(), vput(), and vrele().

Referenced by link().

Here is the call graph for this function:



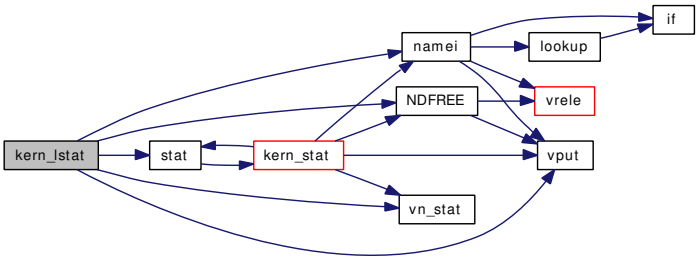
9.163.1.43 int kern_lstat (struct thread * *td*, char * *path*, enum uio_seg *pathseg*, struct stat * *sbp*)

Definition at line 2126 of file vfs_syscalls.c.

References namei(), NDFREE(), stat(), vn_stat(), and vput().

Referenced by lstat(), and nlstat().

Here is the call graph for this function:



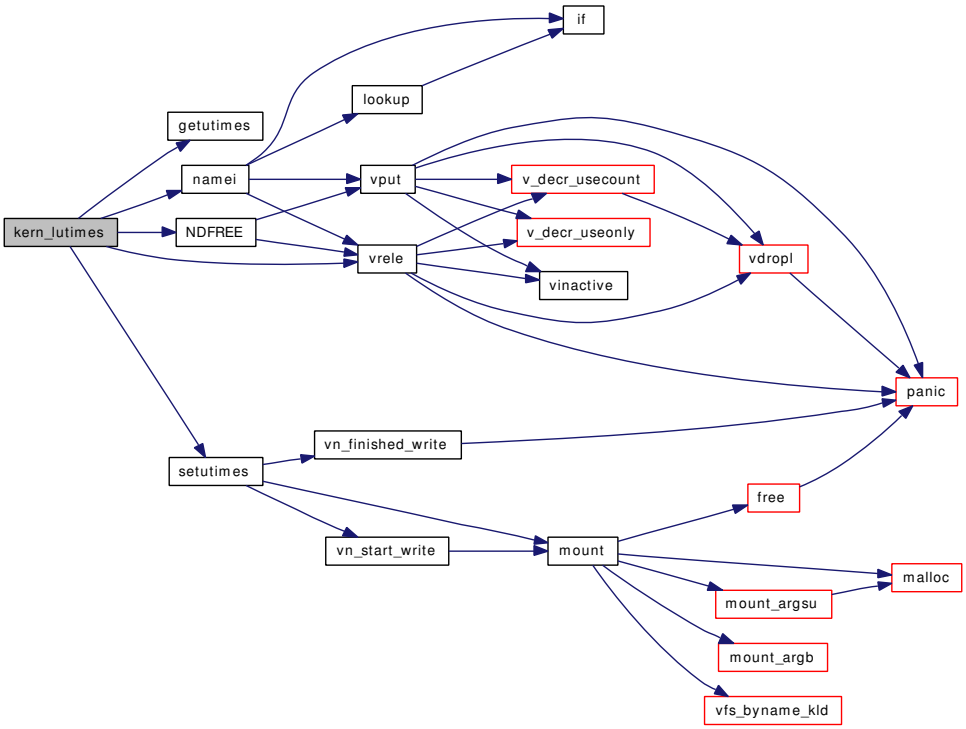
9.163.1.44 int kern_lutimes (struct thread * *td*, char * *path*, enum uio_seg *pathseg*, struct timeval * *tptr*, enum uio_seg *tptrseg*)

Definition at line 2929 of file vfs_syscalls.c.

References getutimes(), namei(), NDFREE(), setutimes(), and vrele().

Referenced by lutimes().

Here is the call graph for this function:



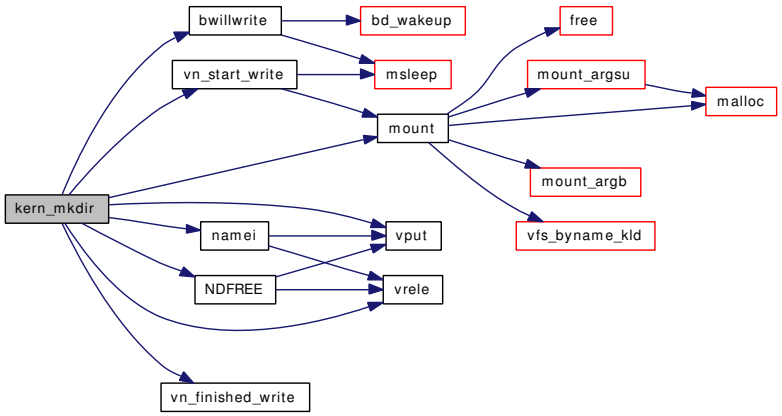
9.163.1.45 int kern_mkdir (struct thread * *td*, char * *path*, enum uio_seg *segflg*, int *mode*)

Definition at line 3386 of file vfs_syscalls.c.

References bwillwrite(), mount(), namei(), NDFREE(), vn_finished_write(), vn_start_write(), vput(), and vrele().

Referenced by mkdir().

Here is the call graph for this function:



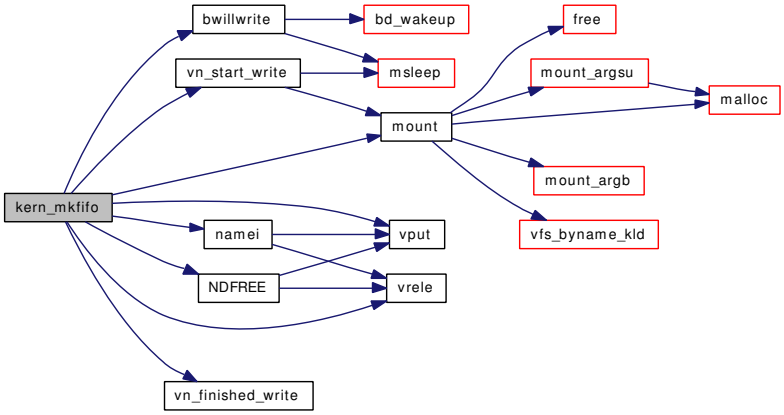
9.163.1.46 int kern_mkfifo (struct thread * *td*, char * *path*, enum uio_seg *pathseg*, int *mode*)

Definition at line 1295 of file vfs_syscalls.c.

References bwillwrite(), mount(), namei(), NDFREE(), vn_finished_write(), vn_start_write(), vput(), and vrele().

Referenced by mkfifo().

Here is the call graph for this function:



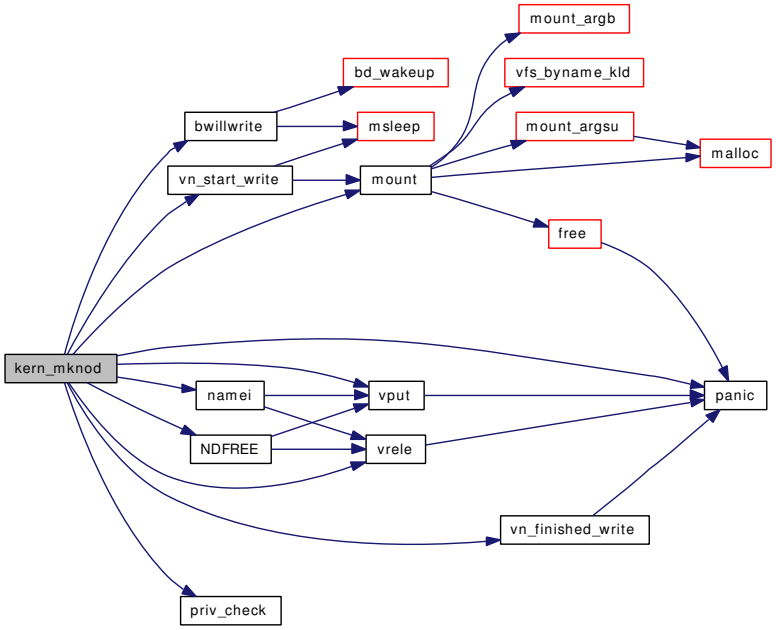
9.163.1.47 int kern_mknod (struct thread * *td*, char * *path*, enum uio_seg *pathseg*, int *mode*, int *dev*)

Definition at line 1169 of file vfs_syscalls.c.

References bwillwrite(), mount(), namei(), NDFREE(), panic(), priv_check(), vn_finished_write(), vn_-start_write(), vput(), and vrele().

Referenced by mknod().

Here is the call graph for this function:



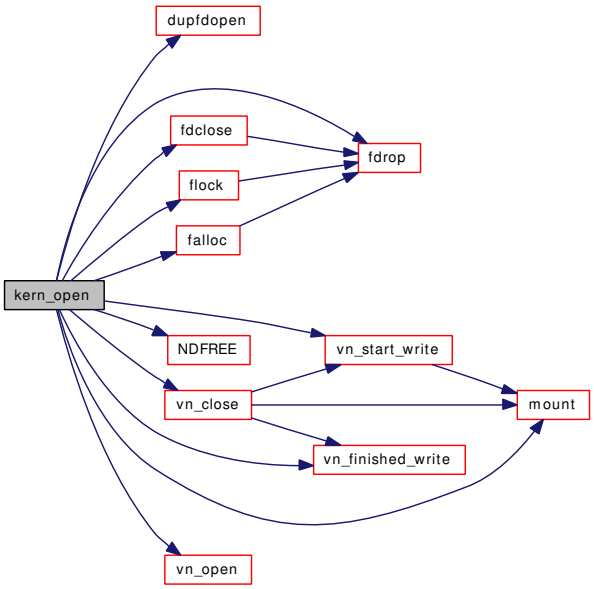
9.163.1.48 int kern_open (struct thread * *td*, char * *path*, enum uio_seg *pathseg*, int *flags*, int *mode*)

Definition at line 960 of file vfs_syscalls.c.

References badfileops, dupfdopen(), falloc(), fdclose(), fdrop(), flock(), mount(), NDFREE(), vn_close(), vn_finished_write(), vn_open(), vn_start_write(), and vnops.

Referenced by open().

Here is the call graph for this function:



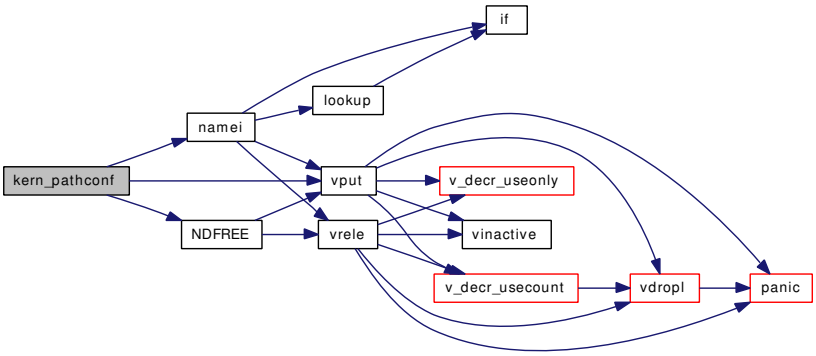
9.163.1.49 int kern_pathconf (struct thread * *td*, char * *path*, enum uio_seg *pathseg*, int *name*)

Definition at line 2254 of file vfs_syscalls.c.

References async_io_version, namei(), NDFREE(), and vput().

Referenced by pathconf().

Here is the call graph for this function:



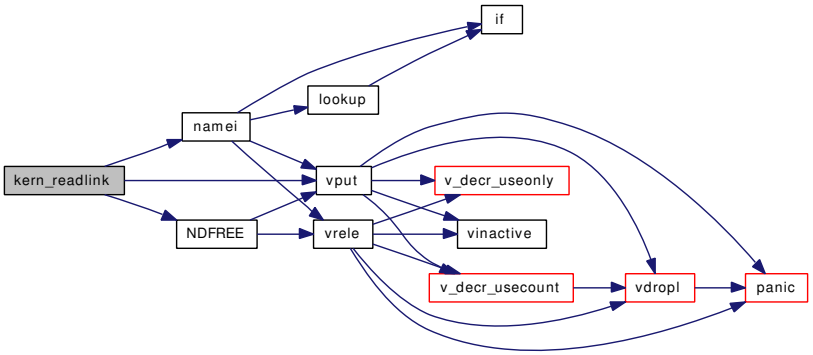
9.163.1.50 int kern_readlink (struct thread * *td*, char * *path*, enum uio_seg *pathseg*, char * *buf*, enum uio_seg *bufseg*, int *count*)

Definition at line 2301 of file vfs_syscalls.c.

References namei(), NDFREE(), and vput().

Referenced by readlink().

Here is the call graph for this function:



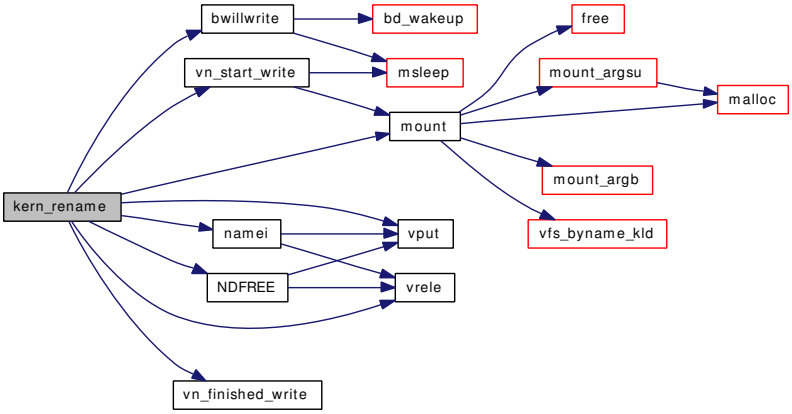
9.163.1.51 int kern_rename (struct thread * *td*, char * *from*, char * *to*, enum uio_seg *pathseg*)

Definition at line 3251 of file vfs_syscalls.c.

References bwillwrite(), mount(), namei(), NDFREE(), vn_finished_write(), vn_start_write(), vput(), and vrele().

Referenced by rename().

Here is the call graph for this function:



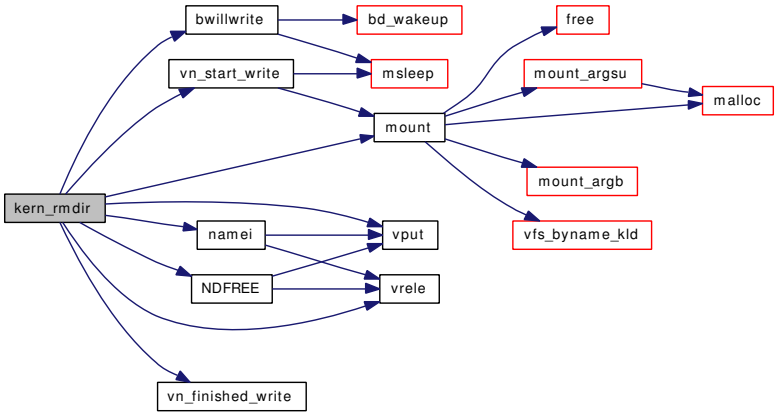
9.163.1.52 int kern_rmdir (struct thread * *td*, char * *path*, enum uio_seg *pathseg*)

Definition at line 3473 of file vfs_syscalls.c.

References bwillwrite(), mount(), namei(), NDFREE(), vn_finished_write(), vn_start_write(), vput(), and vrele().

Referenced by rmdir().

Here is the call graph for this function:



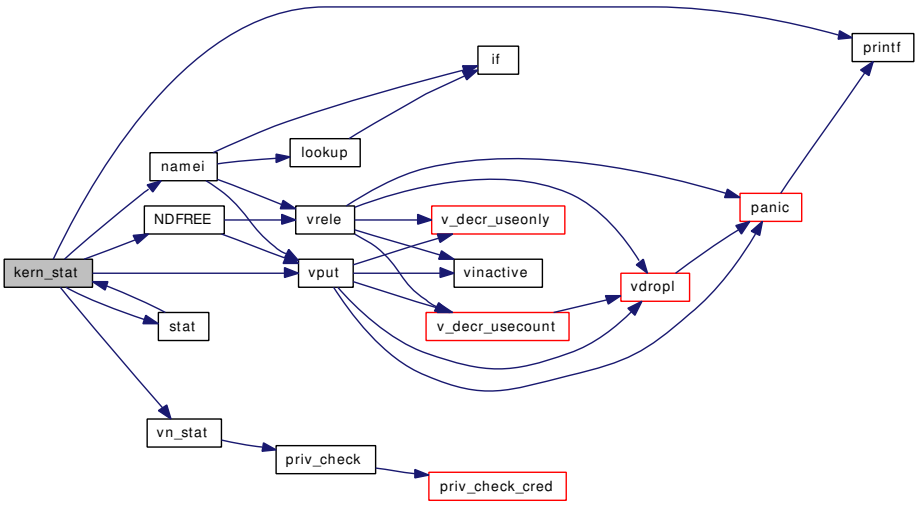
9.163.1.53 int kern_stat (struct thread * *td*, char * *path*, enum uio_seg *pathseg*, struct stat * *sbp*)

Definition at line 2075 of file vfs_syscalls.c.

References Giant, namei(), NDFREE(), printf(), stat(), vn_stat(), and vput().

Referenced by nstat(), and stat().

Here is the call graph for this function:



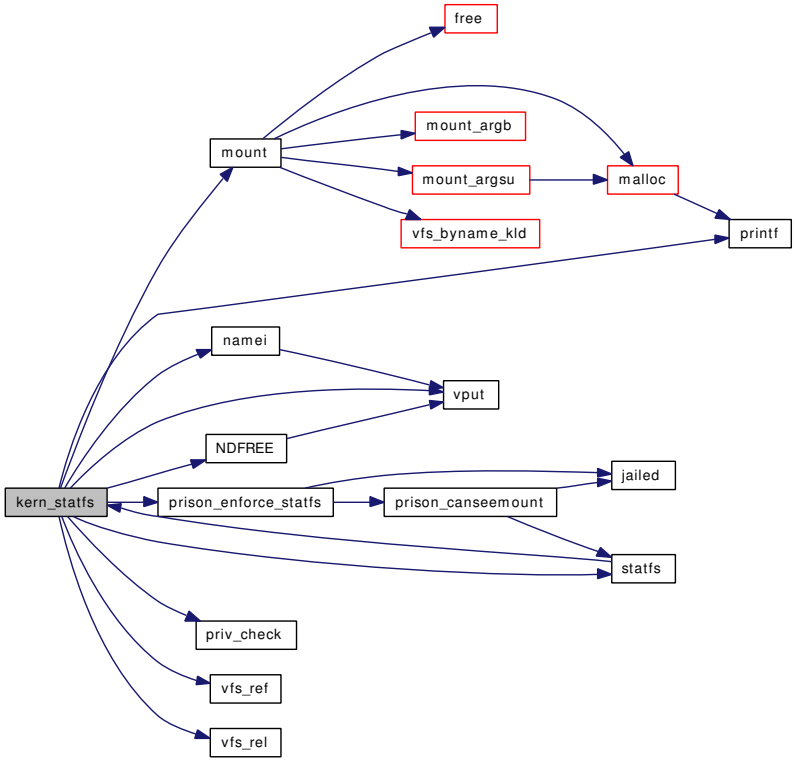
9.163.1.54 int kern_statfs (struct thread * *td*, char * *path*, enum uio_seg *pathseg*, struct statfs * *buf*)

Definition at line 236 of file vfs_syscalls.c.

References Giant, mount(), namei(), NDFREE(), printf(), prison_enforce_statfs(), priv_check(), statfs(), vfs_ref(), vfs_rel(), and vput().

Referenced by statfs().

Here is the call graph for this function:



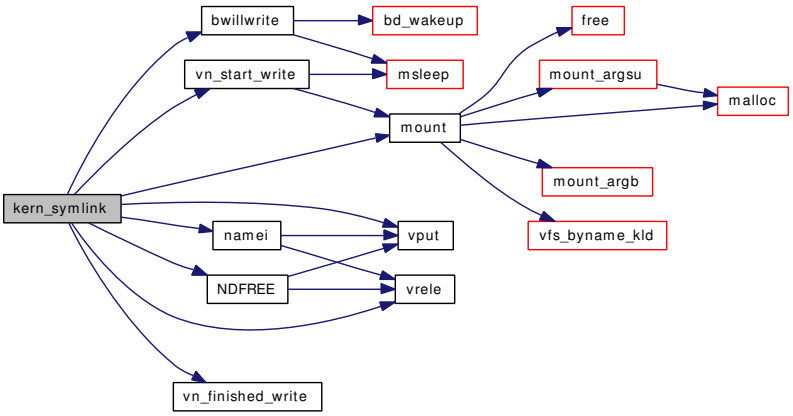
9.163.1.55 int kern_symlink (struct thread * *td*, char * *path*, char * *link*, enum uio_seg *segflg*)

Definition at line 1502 of file vfs_syscalls.c.

References bwillwrite(), mount(), namei(), namei_zone, NDFREE(), td, vn_finished_write(), vn_start_-write(), vput(), and vrele().

Referenced by devfs_first(), and symlink().

Here is the call graph for this function:



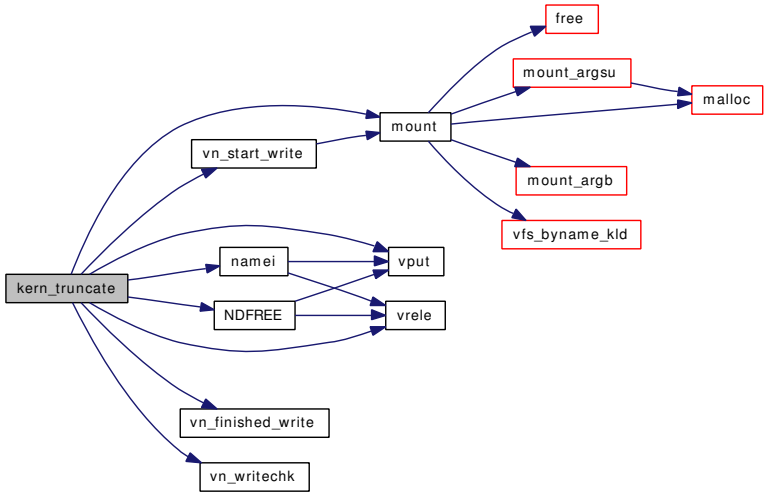
9.163.1.56 int kern_truncate (struct thread * *td*, char * *path*, enum uio_seg *pathseg*, off_t *length*)

Definition at line 3021 of file vfs_syscalls.c.

References mount(), namei(), NDFREE(), vn_finished_write(), vn_start_write(), vn_writechk(), vput(), and vrele().

Referenced by truncate().

Here is the call graph for this function:



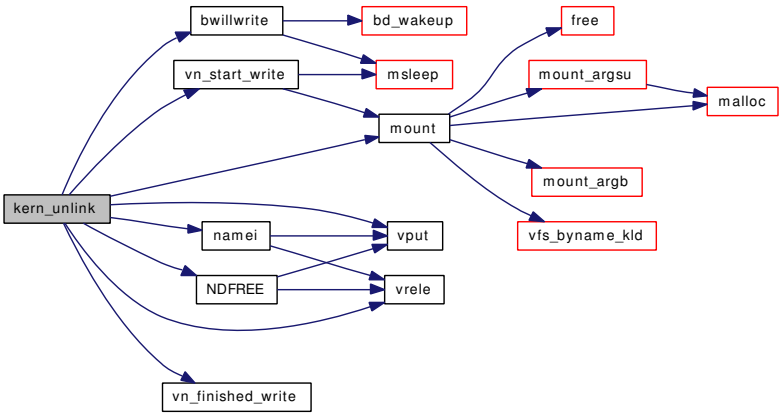
9.163.1.57 int kern_unlink (struct thread * *td*, char * *path*, enum uio_seg *pathseg*)

Definition at line 1647 of file vfs_syscalls.c.

References bwillwrite(), mount(), namei(), NDFREE(), vn_finished_write(), vn_start_write(), vput(), and vrele().

Referenced by devfs_fixup(), and unlink().

Here is the call graph for this function:



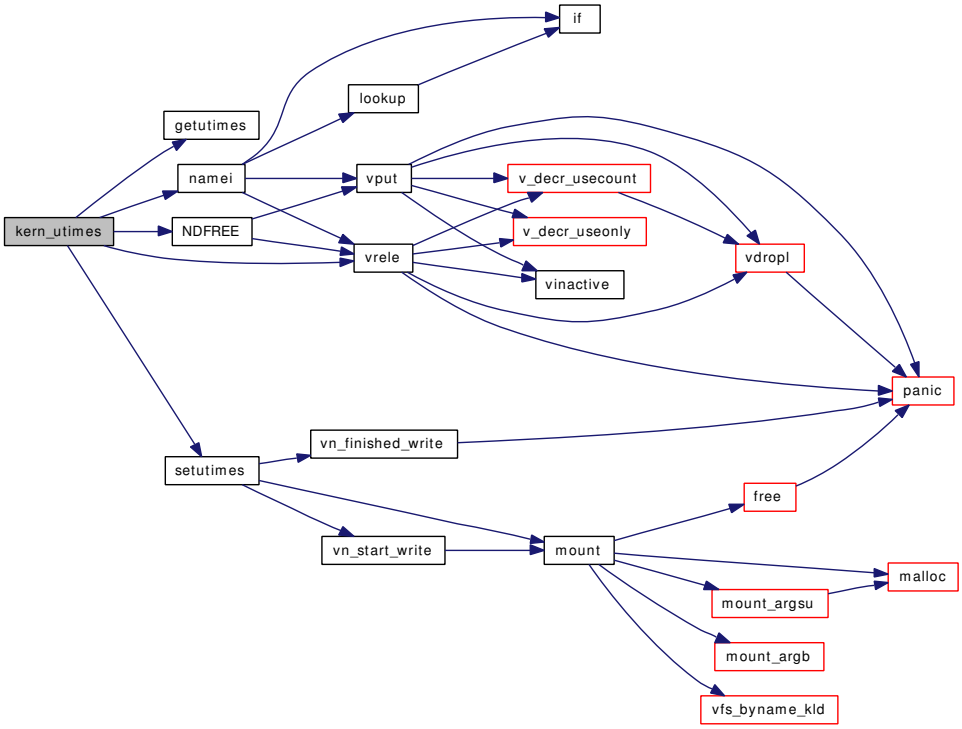
9.163.1.58 int kern_utimes (struct thread * *td*, char * *path*, enum uio_seg *pathseg*, struct timeval * *tptr*, enum uio_seg *tptrseg*)

Definition at line 2885 of file vfs_syscalls.c.

References getutimes(), namei(), NDFREE(), setutimes(), and vrele().

Referenced by utimes().

Here is the call graph for this function:

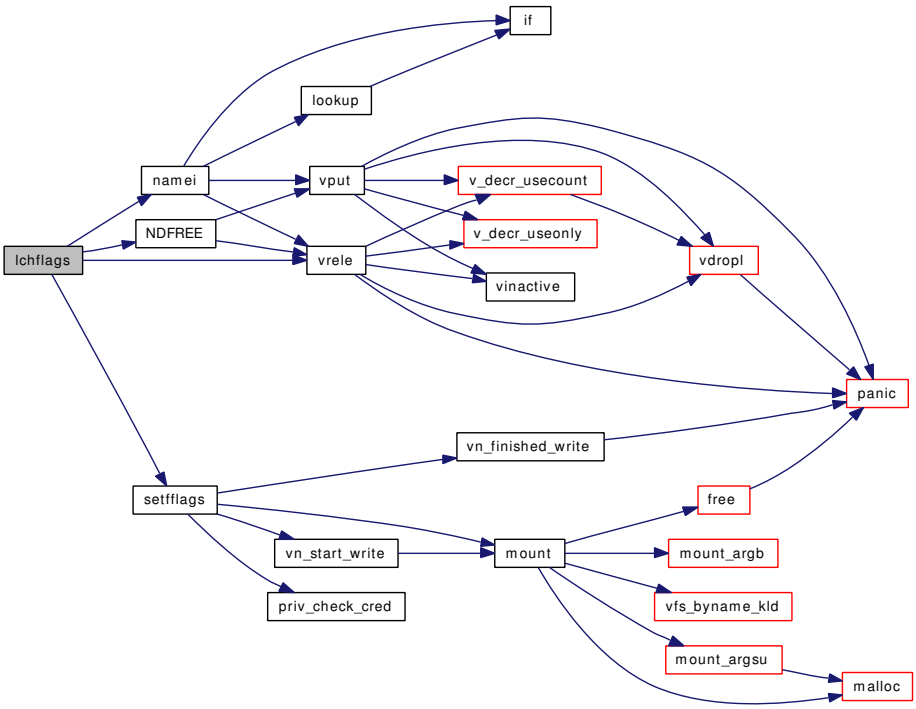


9.163.1.59 int lchflags (struct thread * *td*, struct lchflags_args * *uap*)

Definition at line 2426 of file vfs_syscalls.c.

References namei(), NDFREE(), setfflags(), and vrele().

Here is the call graph for this function:

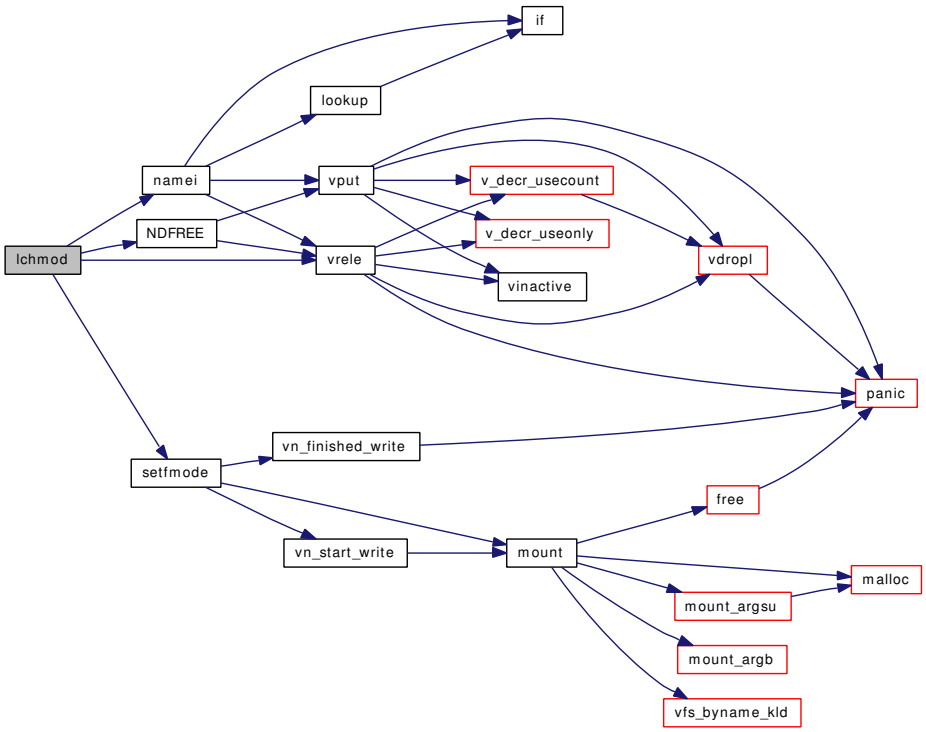


9.163.1.60 int lchmod (struct thread * *td*, struct lchmod_args * *uap*)

Definition at line 2566 of file vfs_syscalls.c.

References namei(), NDFREE(), setfmode(), and vrele().

Here is the call graph for this function:

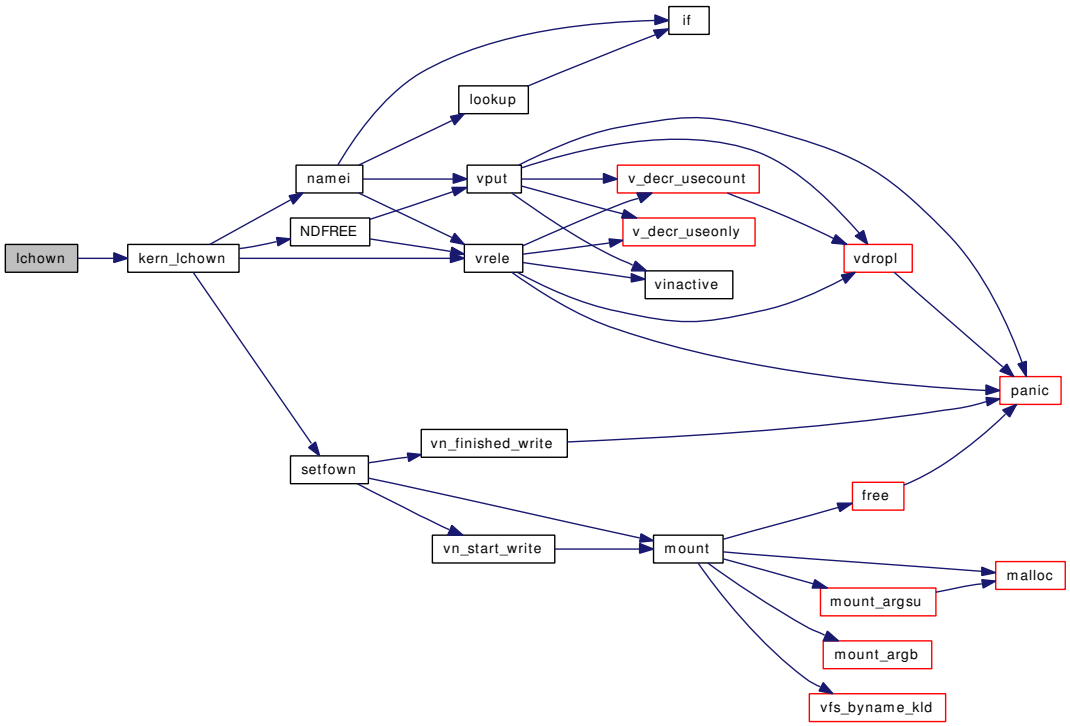


9.163.1.61 int lchown (struct thread * *td*, struct lchown_args * *uap*)

Definition at line 2713 of file vfs_syscalls.c.

References kern_lchown().

Here is the call graph for this function:

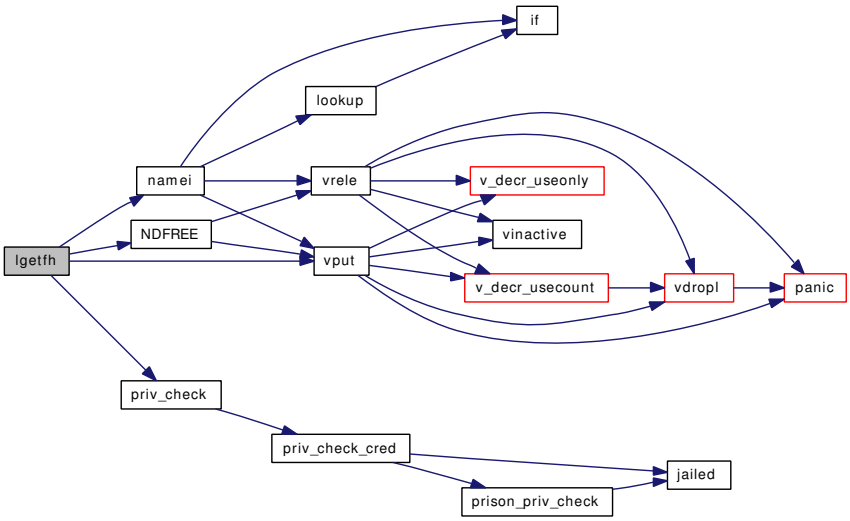


9.163.1.62 int lgetfh (struct thread * *td*, struct lgetfh_args * *uap*)

Definition at line 3927 of file vfs_syscalls.c.

References namei(), NDFREE(), priv_check(), and vput().

Here is the call graph for this function:



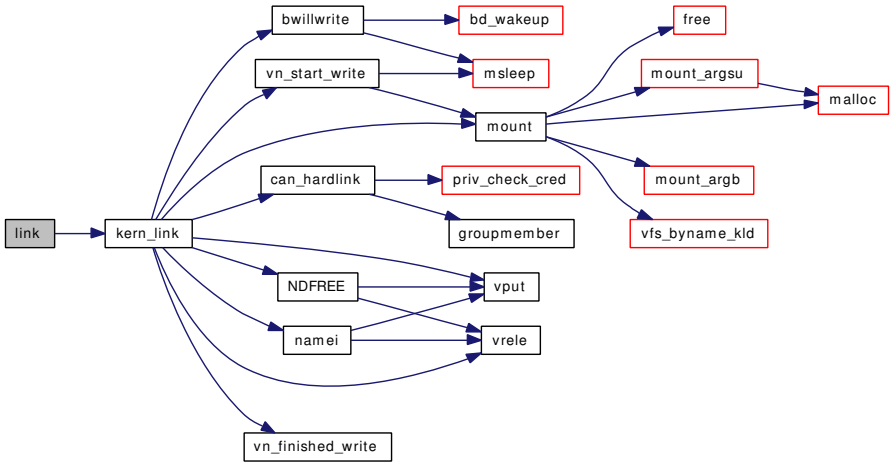
9.163.1.63 int link (struct thread * *td*, struct link_args * *uap*)

Definition at line 1364 of file vfs_syscalls.c.

References kern_link().

Referenced by bus_generic_attach(), bus_generic_detach(), bus_generic_driver_added(), bus_generic_probe(), bus_generic_resume(), bus_generic_shutdown(), bus_generic_suspend(), cf_levels_method(), cpufreq_detach(), cpufreq_dup_set(), cpufreq_expand_set(), cpufreq_insert_abs(), devclass_add_driver(), devclass_delete_driver(), devclass_find_driver_internal(), devclass_find_internal(), devclass_get_drivers(), devclass_quiesce_driver(), device_add_child_ordered(), device_delete_child(), device_get_children(), kldnext(), kldsym(), linker_add_class(), linker_file_foreach(), linker_file_lookup_symbol_internal(), linker_file_unload(), linker_find_file_by_id(), linker_find_file_by_name(), linker_load_file(), linker_make_file(), linker_preload(), mbp_alloc(), mbp_alloc_page(), mbp_card_free(), mbp_count(), mbp_free(), modlist_lookup(), modlist_lookup2(), modlist_newmodule(), modnext(), module_lookupbyid(), module_lookupbyname(), module_register(), module_release(), module_shutdown(), next_matching_driver(), resource_list_add(), resource_list_delete(), resource_list_find(), resource_list_free(), resource_list_print_type(), resource_list_purge(), root_bus_configure(), sysctl_ctx_entry_add(), sysctl_ctx_entry_del(), sysctl_ctx_entry_find(), sysctl_ctx_free(), sysctl_kern_function_list(), TAILQ_HEAD(), vfs_buildopts(), vfs_copyopt(), vfs_deleteopt(), vfs_donmount(), vfs_filteropt(), vfs_flagopt(), vfs_freeopt(), vfs_getopt(), vfs_getopt_pos(), vfs_getopts(), vfs_mergeopts(), vfs_sanitizeopts(), and vfs_scanopt().

Here is the call graph for this function:

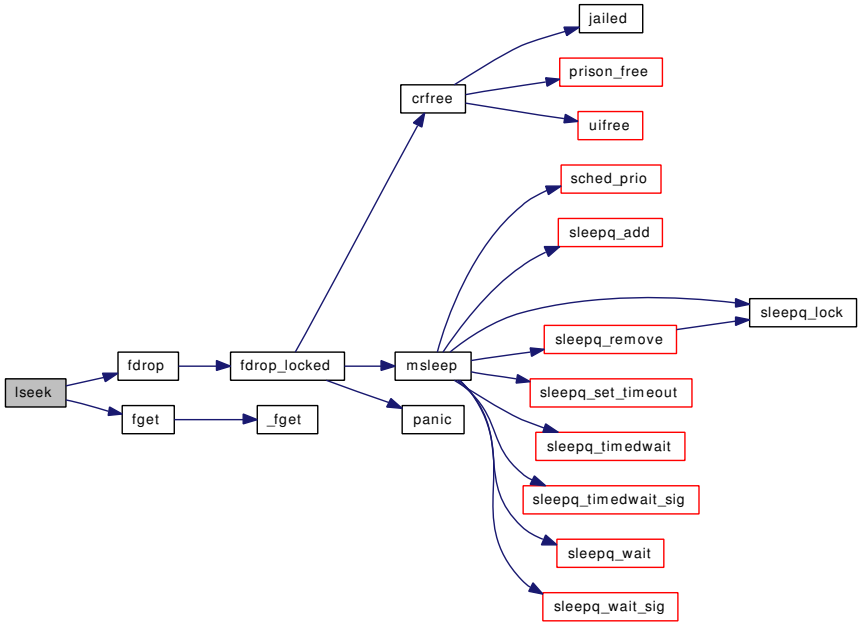


9.163.1.64 int lseek (struct thread * *td*, struct lseek_args * *uap*)

Definition at line 1723 of file vfs_syscalls.c.

References fdrop(), and fget().

Here is the call graph for this function:

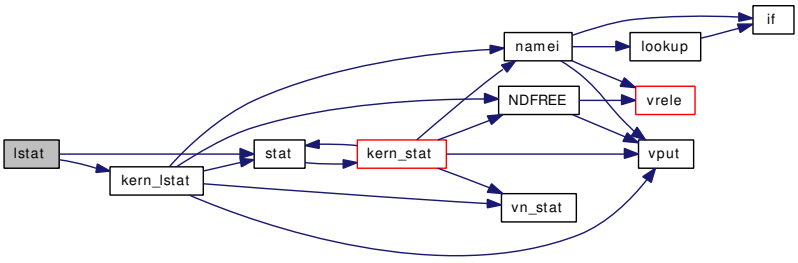


9.163.1.65 int lstat (struct thread * *td*, struct lstat_args * *uap*)

Definition at line 2109 of file vfs_syscalls.c.

References kern_lstat(), and stat().

Here is the call graph for this function:

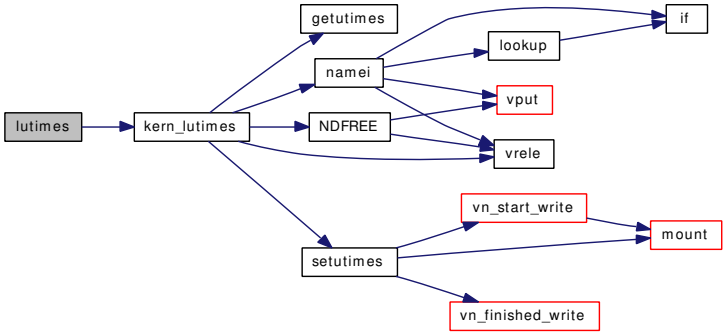


9.163.1.66 int lutimes (struct thread * *td*, struct lutimes_args * *uap*)

Definition at line 2916 of file vfs_syscalls.c.

References kern_lutimes().

Here is the call graph for this function:

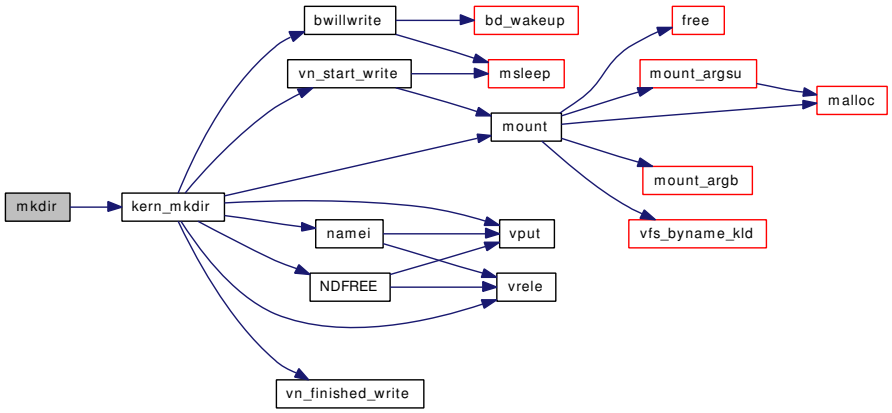


9.163.1.67 int mkdir (struct thread * *td*, struct mkdir_args * *uap*)

Definition at line 3374 of file vfs_syscalls.c.

References kern_mkdir().

Here is the call graph for this function:

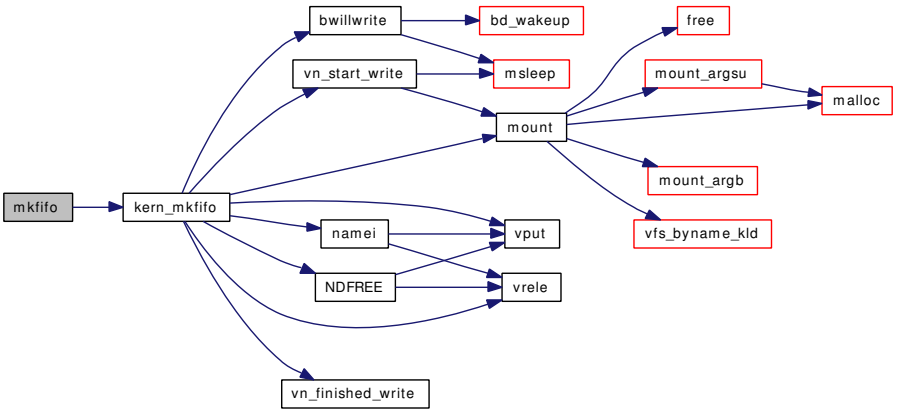


9.163.1.68 int mkfifo (struct thread * *td*, struct mkfifo_args * *uap*)

Definition at line 1283 of file vfs_syscalls.c.

References kern_mkfifo().

Here is the call graph for this function:

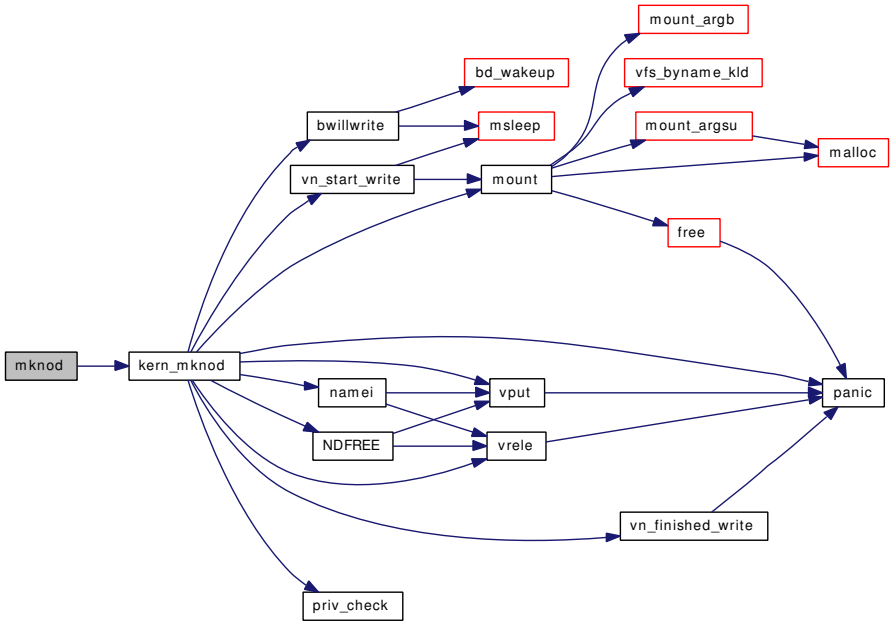


9.163.1.69 int mknod (struct thread * *td*, struct mknod_args * *uap*)

Definition at line 1156 of file vfs_syscalls.c.

References kern_mknod().

Here is the call graph for this function:

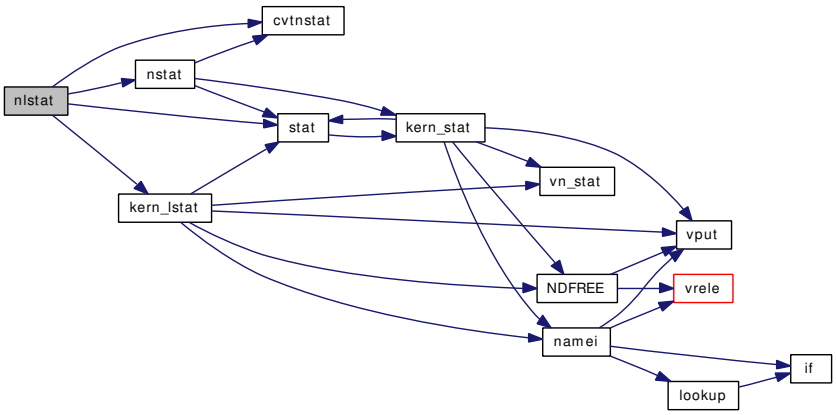


9.163.1.70 int nlstat (struct thread * *td*, struct nlstat_args * *uap*)

Definition at line 2213 of file vfs_syscalls.c.

References cvtnstat(), kern_lstat(), nstat(), and stat().

Here is the call graph for this function:



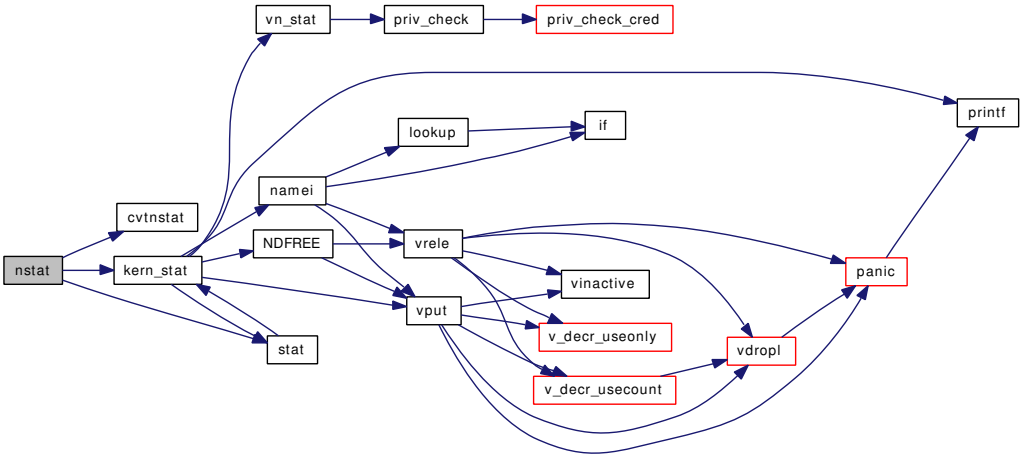
9.163.1.71 int nstat (struct thread * *td*, struct nstat_args * *uap*)

Definition at line 2184 of file vfs_syscalls.c.

References cvtnstat(), kern_stat(), and stat().

Referenced by nfstat(), and nlstat().

Here is the call graph for this function:

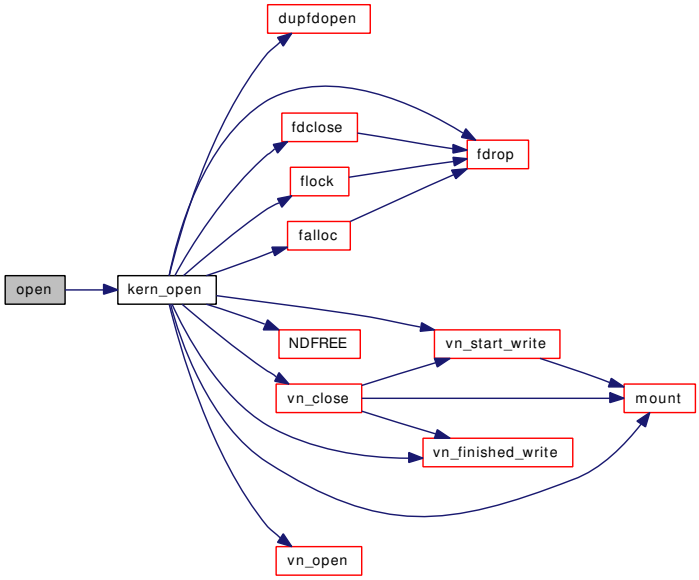


9.163.1.72 int open (struct thread * *td*, struct open_args * *uap*)

Definition at line 947 of file vfs_syscalls.c.

References kern_open().

Here is the call graph for this function:

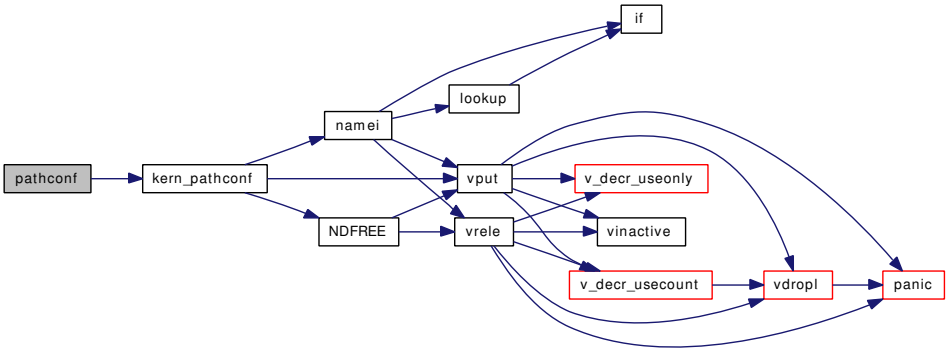


9.163.1.73 int pathconf (struct thread * *td*, struct pathconf_args * *uap*)

Definition at line 2242 of file vfs_syscalls.c.

References kern_pathconf().

Here is the call graph for this function:

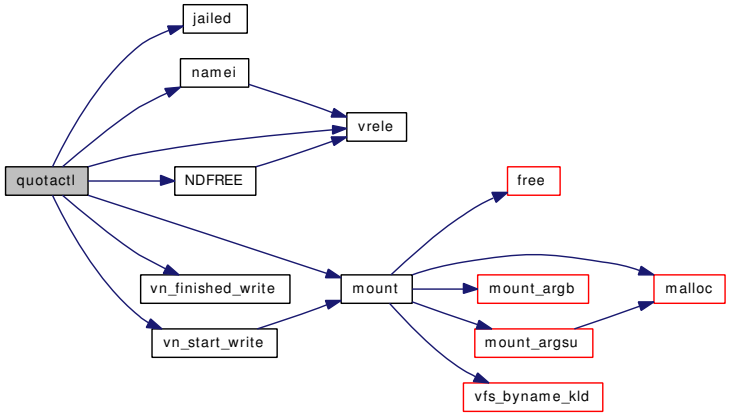


9.163.1.74 int quotactl (struct thread * *td*, struct quotactl_args * *uap*)

Definition at line 173 of file vfs_syscalls.c.

References jailed(), mount(), namei(), NDFREE(), prison_quotas, vn_finished_write(), vn_start_write(), and vrele().

Here is the call graph for this function:

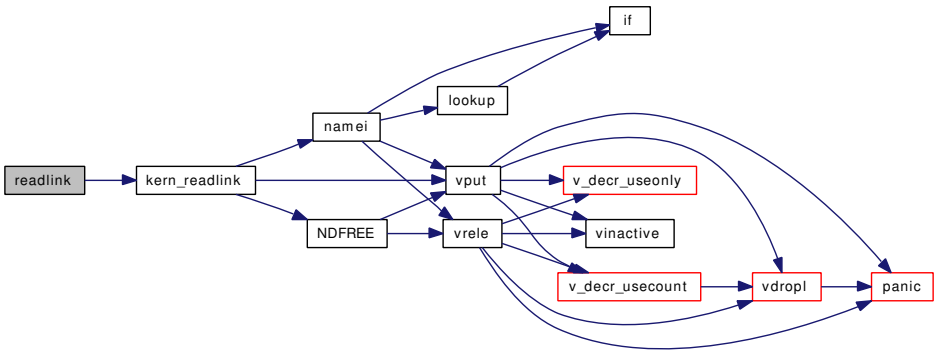


9.163.1.75 int readlink (struct thread * *td*, struct readlink_args * *uap*)

Definition at line 2287 of file vfs_syscalls.c.

References kern_readlink().

Here is the call graph for this function:

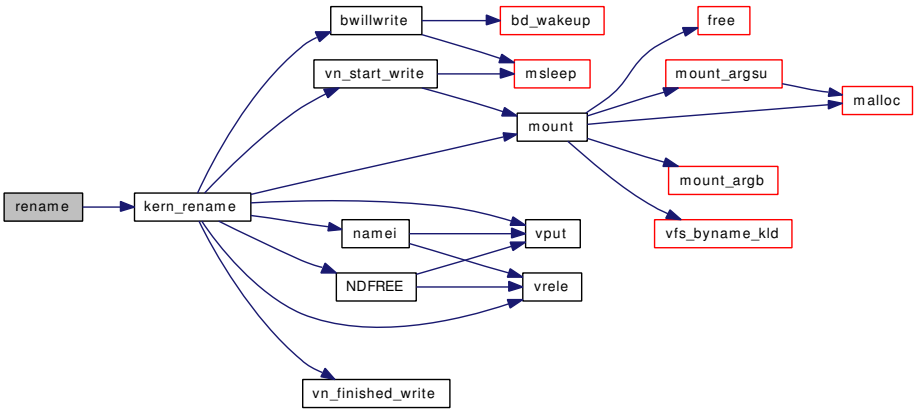


9.163.1.76 int rename (struct thread * *td*, struct rename_args * *uap*)

Definition at line 3239 of file vfs_syscalls.c.

References kern_rename().

Here is the call graph for this function:

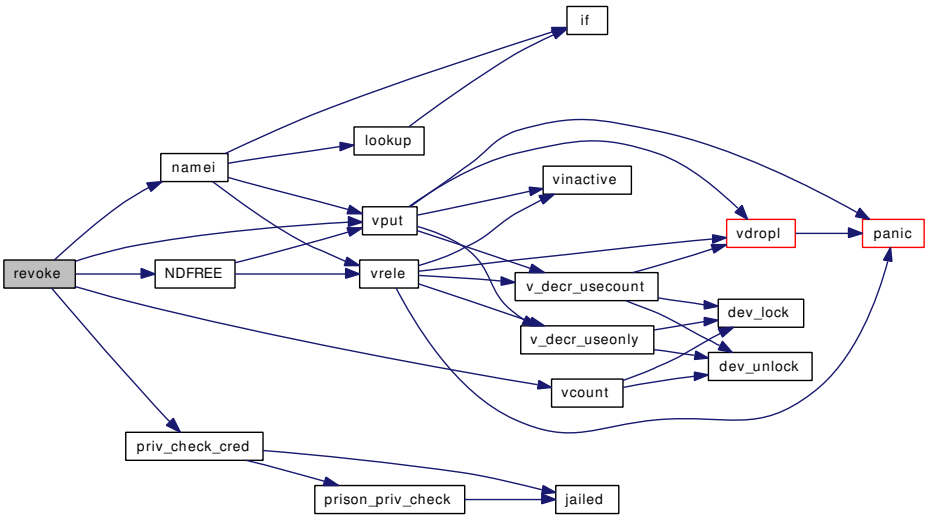


9.163.1.77 int revoke (struct thread * *td*, struct revoke_args * *uap*)

Definition at line 3838 of file vfs_syscalls.c.

References namei(), NDFREE(), priv_check_cred(), vcount(), and vput().

Here is the call graph for this function:

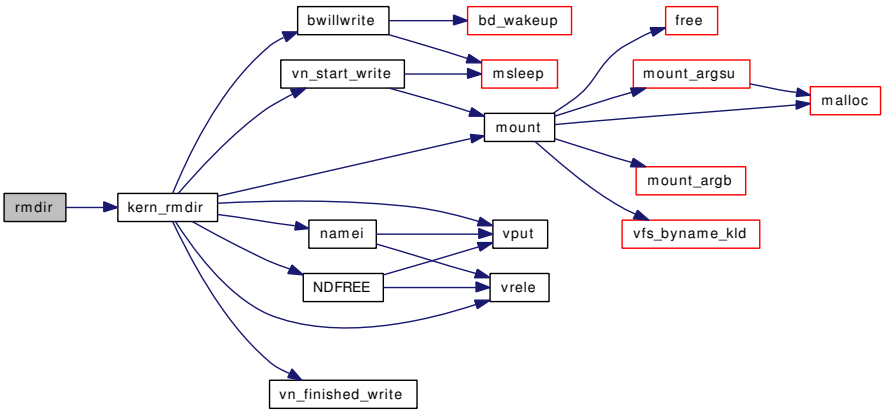


9.163.1.78 int rmdir (struct thread * *td*, struct rmdir_args * *uap*)

Definition at line 3462 of file vfs_syscalls.c.

References kern_rmdir(), and rmdir_args::path.

Here is the call graph for this function:



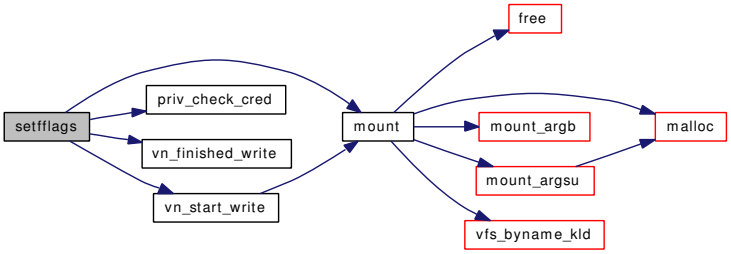
9.163.1.79 static int setfflags (struct thread * *td*, struct vnode *, int) [static]

Definition at line 2350 of file vfs_syscalls.c.

References mount(), priv_check_cred(), vn_finished_write(), and vn_start_write().

Referenced by chflags(), fchflags(), and lchflags().

Here is the call graph for this function:



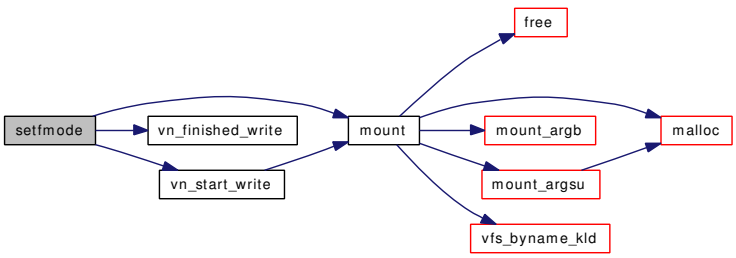
9.163.1.80 static int setfmode (struct thread * *td*, struct vnode *, int) [static]

Definition at line 2491 of file vfs_syscalls.c.

References mount(), vn_finished_write(), and vn_start_write().

Referenced by fchmod(), kern_chmod(), and lchmod().

Here is the call graph for this function:



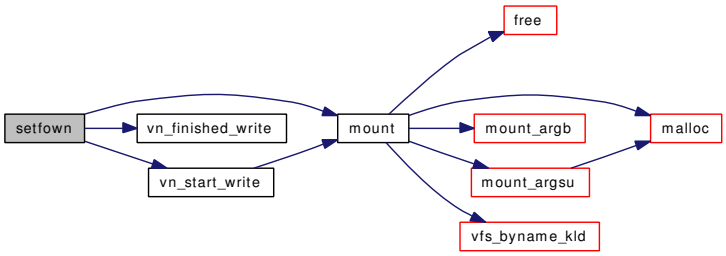
9.163.1.81 static int setfown (struct thread * *td*, struct vnode *, uid_t, gid_t) [static]

Definition at line 2631 of file vfs_syscalls.c.

References mount(), vn_finished_write(), and vn_start_write().

Referenced by fchown(), kern_chown(), and kern_lchown().

Here is the call graph for this function:



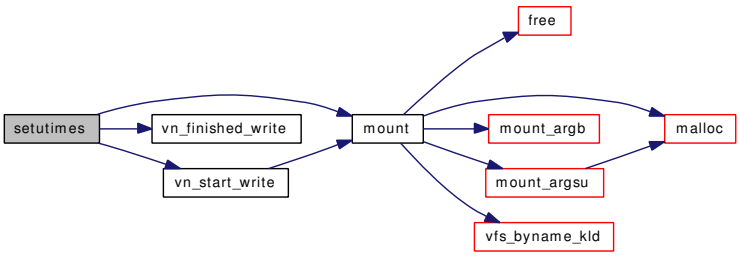
9.163.1.82 static int setutimes (struct thread * *td*, struct vnode *, const struct timespec *, int, int) [static]

Definition at line 2823 of file vfs_syscalls.c.

References mount(), vn_finished_write(), and vn_start_write().

Referenced by kern_futimes(), kern_lutimes(), and kern_utimes().

Here is the call graph for this function:



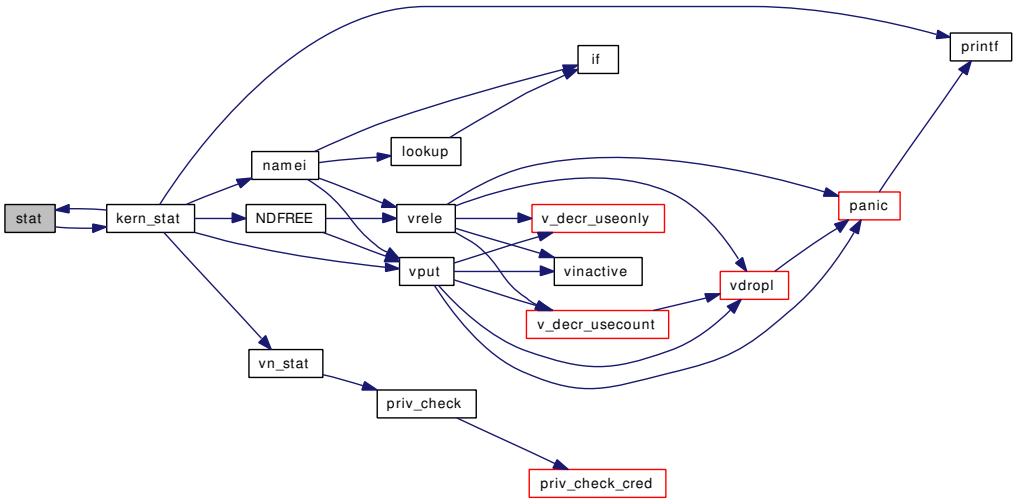
9.163.1.83 int stat (struct thread * *td*, struct stat_args * *uap*)

Definition at line 2058 of file vfs_syscalls.c.

References kern_stat().

Referenced by fhstat(), fstat(), kern_lstat(), kern_stat(), lstat(), modstat(), nfstat(), nlstat(), and nstat().

Here is the call graph for this function:



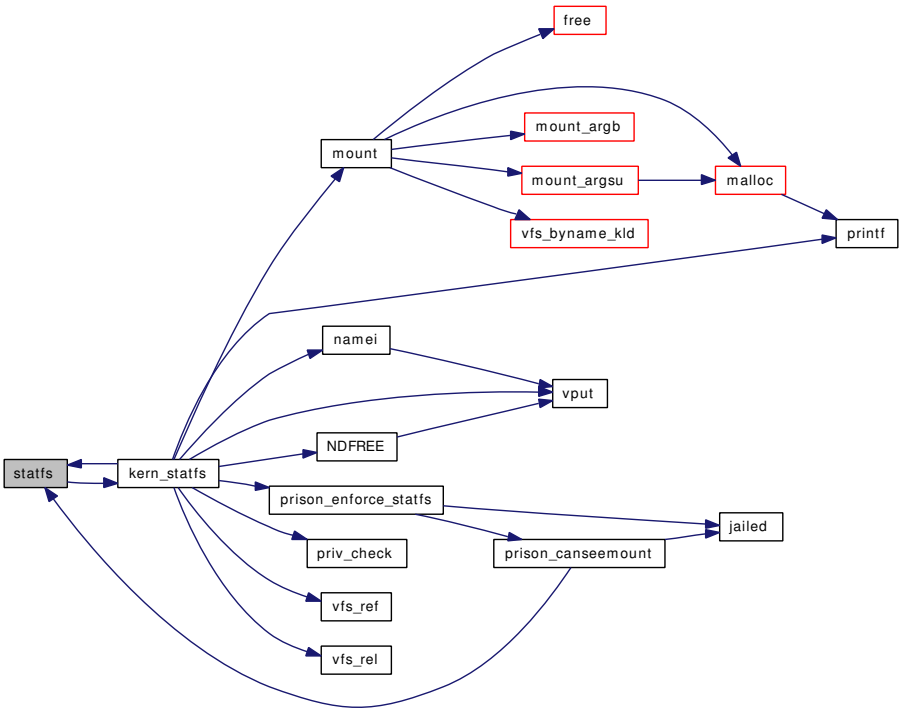
9.163.1.84 int statfs (struct thread * *td*, struct statfs_args * *uap*)

Definition at line 219 of file vfs_syscalls.c.

References kern_statfs().

Referenced by acctwatch(), fhstatfs(), fstatfs(), kern_fhstatfs(), kern_fstatfs(), kern_getfsstat(), kern_statfs(), mqfs_mount(), and prison_canseemount().

Here is the call graph for this function:

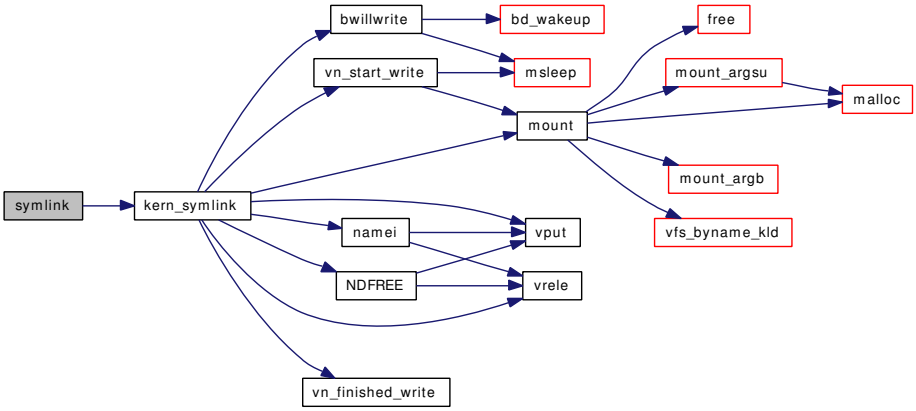


9.163.1.85 int symlink (struct thread * *td*, struct symlink_args * *uap*)

Definition at line 1490 of file vfs_syscalls.c.

References kern_symlink().

Here is the call graph for this function:

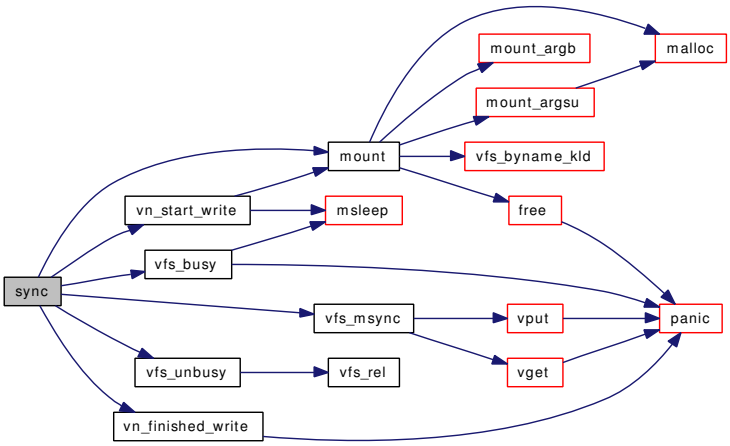


9.163.1.86 int sync (struct thread * *td*, struct sync_args * *uap*)

Definition at line 114 of file vfs_syscalls.c.

References mount(), mountlist, mountlist_mtx, vfs_busy(), vfs_msync(), vfs_unbusy(), vn_finished_-write(), and vn_start_write().

Here is the call graph for this function:



9.163.1.87 SYSCTL_INT (_security_bsd, OID_AUTO, hardlink_check_gid, CTLFLAG_RW, & *hardlink_check_gid*, 0, "Unprivileged processes cannot create hard links to files owned by other ""groups")

9.163.1.88 SYSCTL_INT (_security_bsd, OID_AUTO, hardlink_check_uid, CTLFLAG_RW, & *hardlink_check_uid*, 0, "Unprivileged processes cannot create hard links to files owned by other ""users")

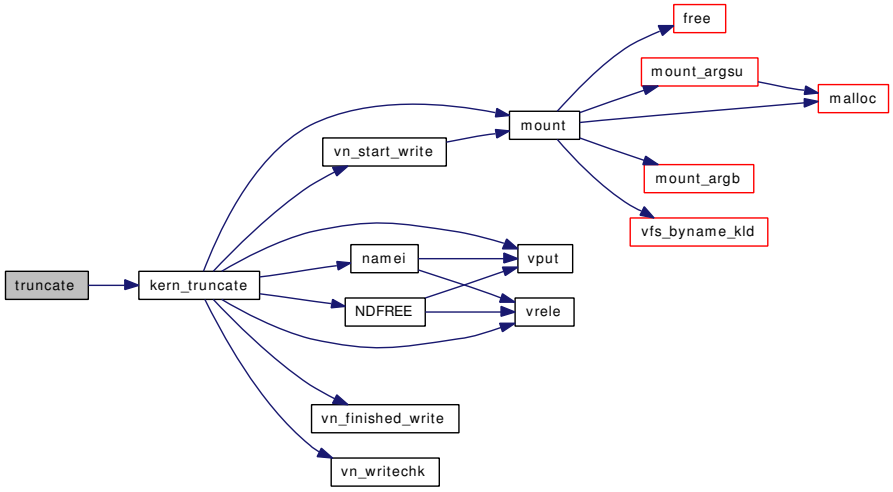
9.163.1.89 SYSCTL_INT (_kern, OID_AUTO, chroot_allow_open_directories, CTLFLAG_RW, & *chroot_allow_open_directories*, 0, "")

9.163.1.90 int truncate (struct thread * *td*, struct truncate_args * *uap*)

Definition at line 3008 of file vfs_syscalls.c.

References kern_truncate().

Here is the call graph for this function:



9.163.1.91 int umask (struct thread * *td*, struct umask_args * *uap*)

Definition at line 3812 of file vfs_syscalls.c.

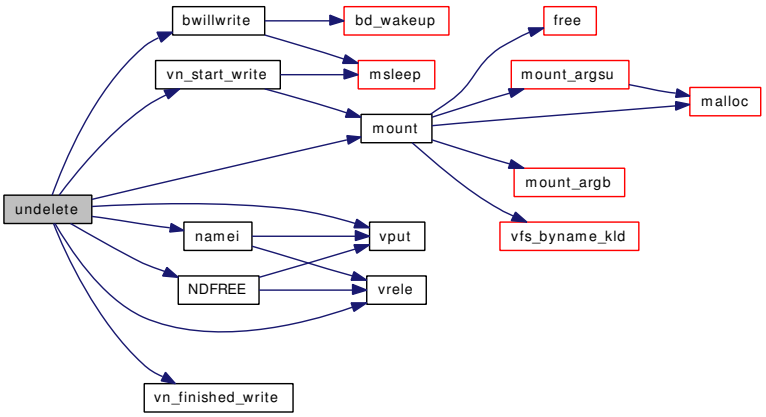
References umask_args::newmask.

9.163.1.92 int undelete (struct thread * *td*, struct undelete_args * *uap*)

Definition at line 1577 of file vfs_syscalls.c.

References bwillwrite(), mount(), namei(), NDFREE(), vn_finished_write(), vn_start_write(), vput(), and vrele().

Here is the call graph for this function:

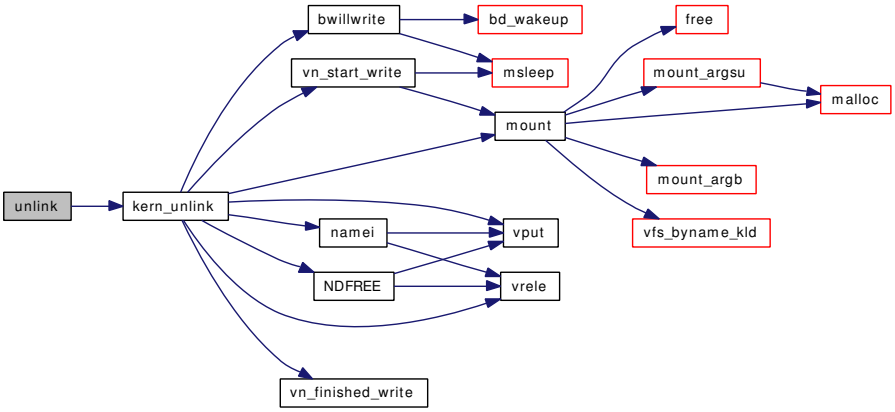


9.163.1.93 int unlink (struct thread * *td*, struct unlink_args * *uap*)

Definition at line 1634 of file vfs_syscalls.c.

References kern_unlink(), and unlink_args::path.

Here is the call graph for this function:

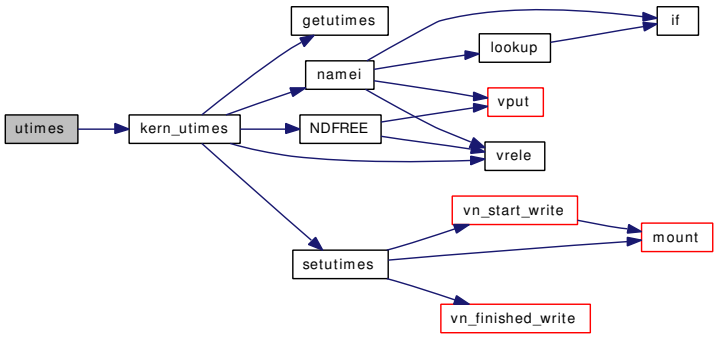


9.163.1.94 int utimes (struct thread * *td*, struct utimes_args * *uap*)

Definition at line 2872 of file vfs_syscalls.c.

References kern_utimes().

Here is the call graph for this function:



9.163.1.95 static int vn_access (struct vnode * *vp*, int *user_flags*, struct ucred * *cred*, struct thread * *td*) [static]

Definition at line 1831 of file vfs_syscalls.c.

References vn_writechk().

Referenced by kern_access(), and kern_eaccess().

Here is the call graph for this function:

9.163.2 Variable Documentation

9.163.2.1 int async_io_version

Definition at line 96 of file vfs_syscalls.c.

Referenced by aio_onceonly(), aio_unload(), fpathconf(), and kern_pathconf().

9.163.2.2 int chroot_allow_open_directories = 1 [static]

Definition at line 816 of file vfs_syscalls.c.

9.163.2.3 int hardlink_check_gid = 0 [static]

Definition at line 1382 of file vfs_syscalls.c.

9.163.2.4 int hardlink_check_uid = 0 [static]

Definition at line 1377 of file vfs_syscalls.c.

9.163.2.5 int prison_quotas [static]

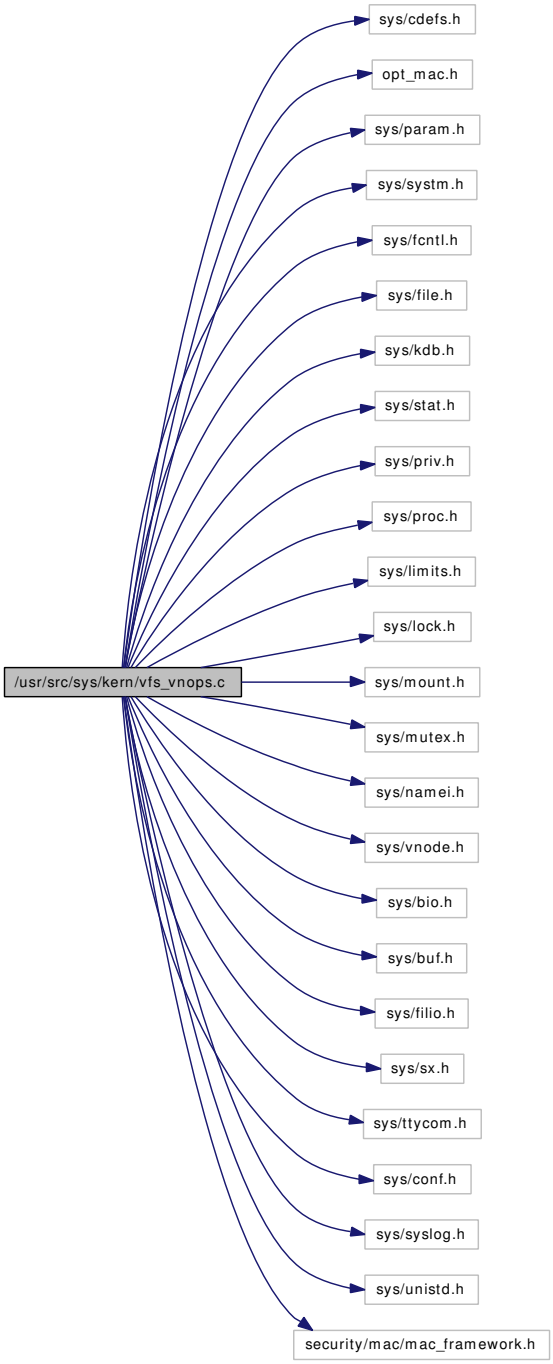
Definition at line 154 of file vfs_syscalls.c.

Referenced by quotactl().

9.164 /usr/src/sys/kern/vfs_vnops.c File Reference

```
#include <sys/cdefs.h>
#include "opt_mac.h"
#include <sys/param.h>
#include <sys/systm.h>
#include <sys/fcntl.h>
#include <sys/file.h>
#include <sys/kdb.h>
#include <sys/stat.h>
#include <sys/priv.h>
#include <sys/proc.h>
#include <sys/limits.h>
#include <sys/lock.h>
#include <sys/mount.h>
#include <sys/mutex.h>
#include <sys/namei.h>
#include <sys/vnode.h>
#include <sys/bio.h>
#include <sys/buf.h>
#include <sys/filio.h>
#include <sys/sx.h>
#include <sys/ttycom.h>
#include <sys/conf.h>
#include <sys/syslog.h>
#include <sys/unistd.h>
#include <security/mac/mac_framework.h>
```

Include dependency graph for vfs_vnops.c:



Functions

- __FBSDID ("$FreeBSD: src/sys/kern/vfs_vnops.c,v 1.248 2007/02/12 22:53:01 mpp Exp $")
- int vn_open (struct nameidata *ndp, int *flagp, int cmode, int fdidx)
- int vn_open_cred (struct nameidata *ndp, int *flagp, int cmode, struct ucred *cred, int fdidx)
- int vn_writechk (struct vnode *vp)
- int vn_close (struct vnode *vp, int flags, struct ucred *file_cred, struct thread *td)
- static __inline int sequential_heuristic (struct uio *uio, struct file *fp)

- int vn_rdwr (enum uio_rw rw, struct vnode *vp, void *base, int len, off_t offset, enum uio_seg segflg, int ioflg, struct ucred *active_cred, struct ucred *file_cred, int *aresid, struct thread *td)
- int vn_rdwr_inchunks (enum uio_rw rw, struct vnode *vp, void *base, size_t len, off_t offset, enum uio_seg segflg, int ioflg, struct ucred *active_cred, struct ucred *file_cred, size_t *aresid, struct thread *td)
- static int vn_read (struct file *fp, struct uio *uio, struct ucred *active_cred, int flags, struct thread *td)
- static int vn_write (struct file *fp, struct uio *uio, struct ucred *active_cred, int flags, struct thread *td)
- static int vn_statfile (struct file *fp, struct stat *sb, struct ucred *active_cred, struct thread *td)
- int vn_stat (struct vnode *vp, struct stat *sb, struct ucred *active_cred, struct ucred *file_cred, struct thread *td)
- static int vn_ioctl (struct file *fp, u_long com, void *data, struct ucred *active_cred, struct thread *td)
- static int vn_poll (struct file *fp, int events, struct ucred *active_cred, struct thread *td)
- int _vn_lock (struct vnode *vp, int flags, struct thread *td, char *file, int line)
- static int vn_closefile (struct file *fp, struct thread *td)
- int vn_start_write (struct vnode *vp, struct mount **mpp, int flags)
- int vn_write_suspend_wait (struct vnode *vp, struct mount *mp, int flags)
- int vn_start_secondary_write (struct vnode *vp, struct mount **mpp, int flags)
- void vn_finished_write (struct mount *mp)
- void vn_finished_secondary_write (struct mount *mp)
- int vfs_write_suspend (struct mount *mp)
- void vfs_write_resume (struct mount *mp)
- static int vn_kqfilter (struct file *fp, struct knote *kn)
- int vn_extattr_get (struct vnode *vp, int ioflg, int attrnamespace, const char *attrname, int *buflen, char *buf, struct thread *td)
- int vn_extattr_set (struct vnode *vp, int ioflg, int attrnamespace, const char *attrname, int buflen, char *buf, struct thread *td)
- int vn_extattr_rm (struct vnode *vp, int ioflg, int attrnamespace, const char *attrname, struct thread *td)

Variables

- static fo_rdwr_t vn_read
- static fo_rdwr_t vn_write
- static fo_ioctl_t vn_ioctl
- static fo_poll_t vn_poll
- static fo_kqfilter_t vn_kqfilter
- static fo_stat_t vn_statfile
- static fo_close_t vn_closefile
- fileops vnops

9.164.1 Function Documentation

9.164.1.1 __FBSDID ("$FreeBSD: src/sys/kern/vfs_vnops. *c*, v 1.248 2007/02/12 22:53:01 mpp Exp $")

9.164.1.2 int _vn_lock (struct vnode * *vp*, int *flags*, struct thread * *td*, char * *file*, int *line*)

Definition at line 805 of file vfs_vnops.c.

9.164.1.3 static __inline int sequential_heuristic (struct uio * *uio*, struct file * *fp*) [static]

Definition at line 304 of file vfs_vnops.c.

Referenced by vn_read(), and vn_write().

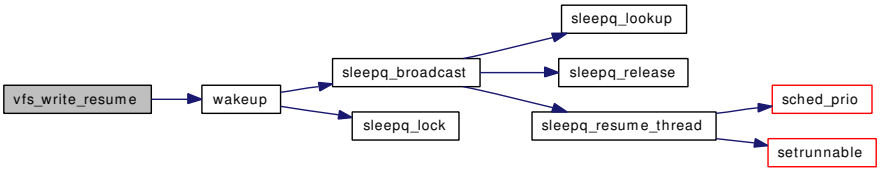
9.164.1.4 void vfs_write_resume (struct mount * *mp*)

Definition at line 1114 of file vfs_vnops.c.

References wakeup().

Referenced by vfs_write_suspend().

Here is the call graph for this function:

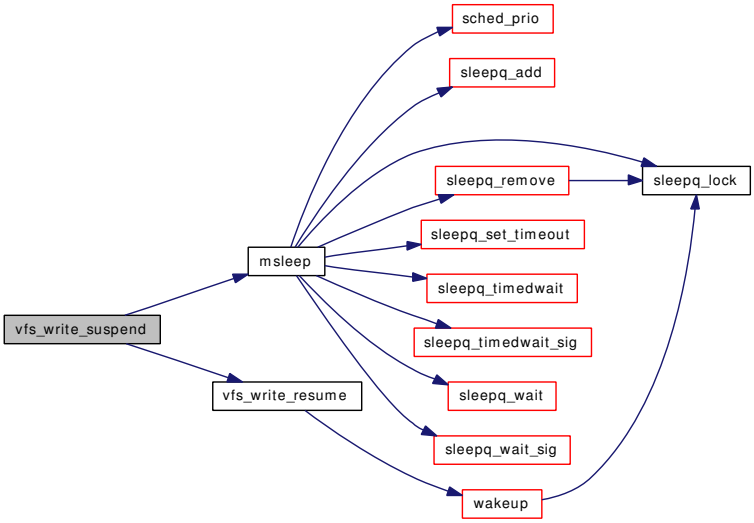


9.164.1.5 int vfs_write_suspend (struct mount * *mp*)

Definition at line 1088 of file vfs_vnops.c.

References msleep(), td, and vfs_write_resume().

Here is the call graph for this function:



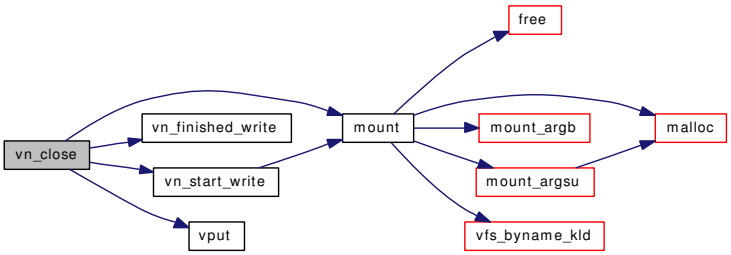
9.164.1.6 int vn_close (struct vnode * *vp*, int *flags*, struct ucred * *file_cred*, struct thread * *td*)

Definition at line 275 of file vfs_vnops.c.

References mount(), vn_finished_write(), vn_start_write(), and vput().

Referenced by acct(), acct_disable(), alq_shutdown(), cn_devopen(), cnclose(), cnremove(), coredump(), kern_open(), ktrace(), link_elf_load_file(), linker_hints_lookup(), linker_lookup_file(), and vn_closefile().

Here is the call graph for this function:

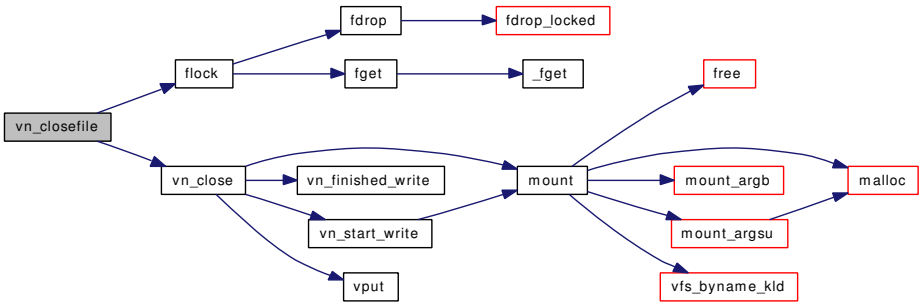


9.164.1.7 static int vn_closefile (struct file * *fp*, struct thread * *td*) [static]

Definition at line 850 of file vfs_vnops.c.

References badfileops, flock(), and vn_close().

Here is the call graph for this function:



9.164.1.8 int vn_extattr_get (struct vnode * *vp*, int *ioflg*, int *attrnamespace*, const char * *attrname*, int * *buflen*, char * *buf*, struct thread * *td*)

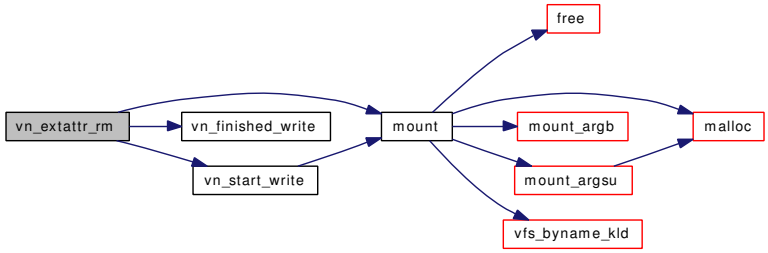
Definition at line 1150 of file vfs_vnops.c.

9.164.1.9 int vn_extattr_rm (struct vnode * *vp*, int *ioflg*, int *attrnamespace*, const char * *attrname*, struct thread * *td*)

Definition at line 1230 of file vfs_vnops.c.

References mount(), vn_finished_write(), and vn_start_write().

Here is the call graph for this function:

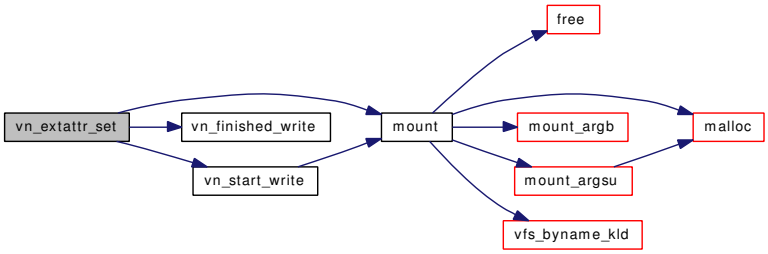


9.164.1.10 int vn_extattr_set (struct vnode * *vp*, int *ioflg*, int *attrnamespace*, const char * *attrname*, int *buflen*, char * *buf*, struct thread * *td*)

Definition at line 1191 of file vfs_vnops.c.

References mount(), vn_finished_write(), and vn_start_write().

Here is the call graph for this function:



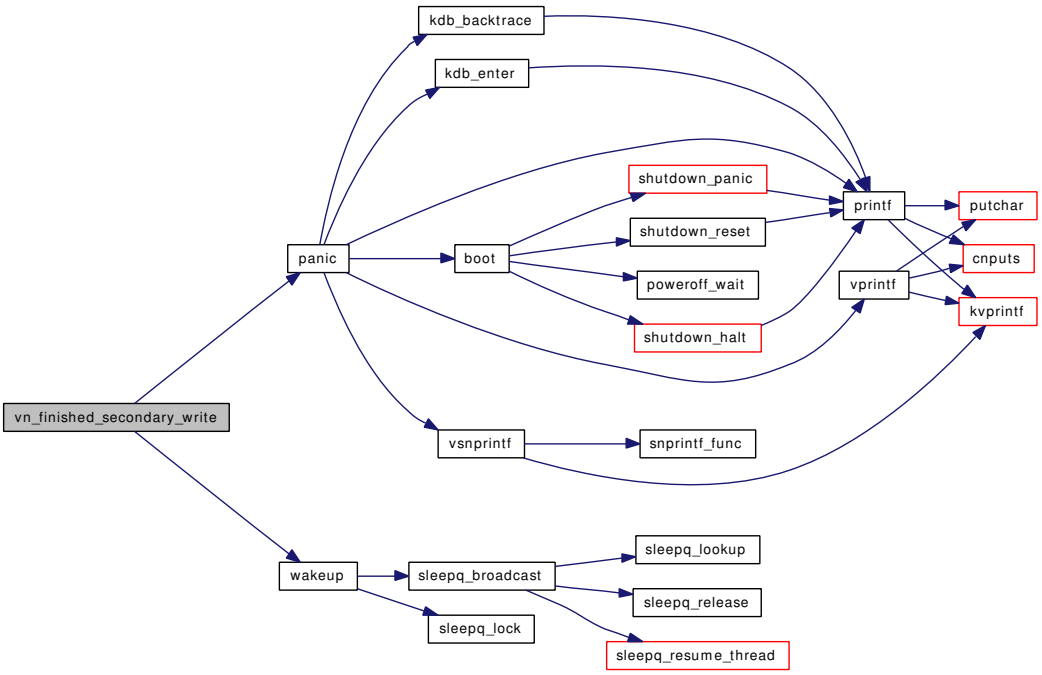
9.164.1.11 void vn_finished_secondary_write (struct mount * *mp*)

Definition at line 1067 of file vfs_vnops.c.

References panic(), and wakeup().

Referenced by vgonel().

Here is the call graph for this function:



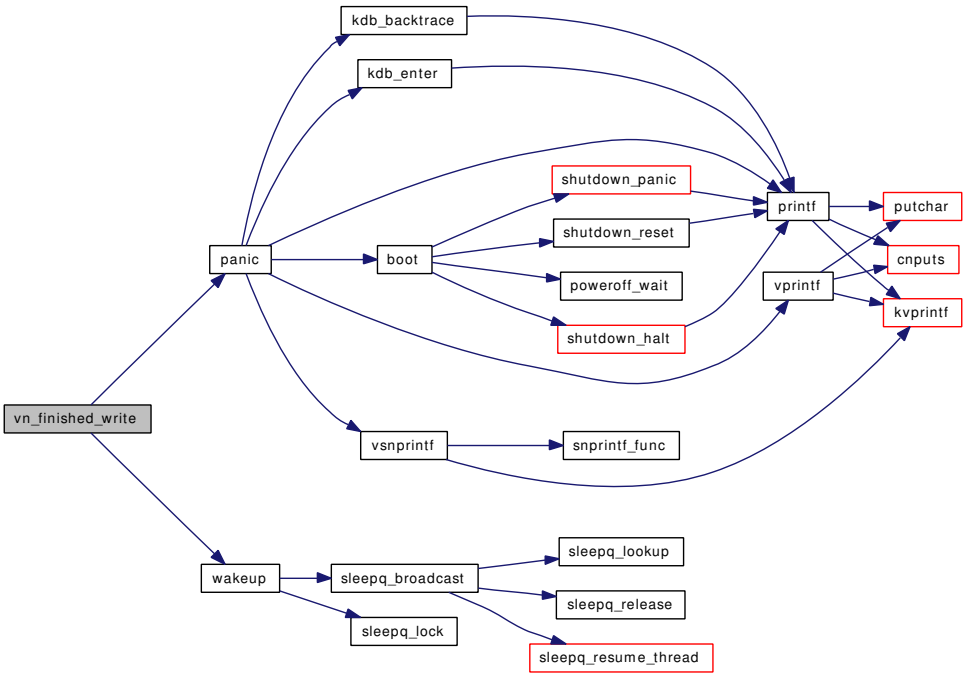
9.164.1.12 void vn_finished_write (struct mount * *mp*)

Definition at line 1045 of file vfs_vnops.c.

References panic(), and wakeup().

Referenced by aio_fsync_vnode(), alq_doio(), coredump(), dounmount(), extattr_delete_vp(), extattr_set_-vp(), extattrctl(), fhopen(), flushbufqueues(), fsync(), ftruncate(), kern_link(), kern_mkdir(), kern_mkfifo(), kern_mknod(), kern_open(), kern_rename(), kern_rmdir(), kern_symlink(), kern_truncate(), kern_unlink(), quotactl(), setfflags(), setfmode(), setfown(), setutimes(), sync(), sync_fsync(), sync_vnode(), uipc_bind(), undelete(), vacl_delete(), vacl_set_acl(), vlrureclaim(), vn_close(), vn_extattr_rm(), vn_extattr_set(), vn_-open_cred(), vn_rdwr(), vn_write(), and vtryrecycle().

Here is the call graph for this function:



9.164.1.13 static int vn_ioctl (struct file * *fp*, u_long *com*, void * *data*, struct ucred * *active_cred*, struct thread * *td*) [static]

Definition at line 734 of file vfs_vnops.c.

9.164.1.14 static int vn_kqfilter (struct file * *fp*, struct knote * *kn*) [static]

Definition at line 1132 of file vfs_vnops.c.

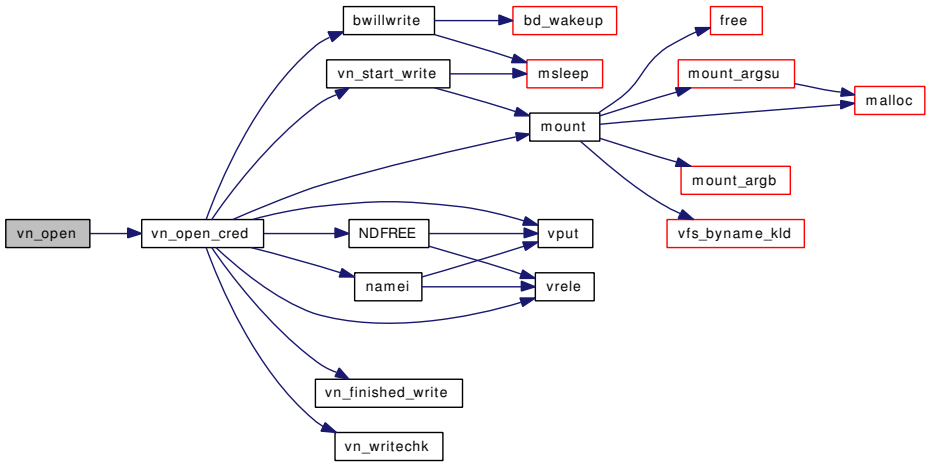
9.164.1.15 int vn_open (struct nameidata * *ndp*, int * *flagp*, int *cmode*, int *fdidx*)

Definition at line 87 of file vfs_vnops.c.

References td, and vn_open_cred().

Referenced by acct(), cn_devopen(), coredump(), fdcheckstd(), kern_open(), ktrace(), link_elf_load_file(), linker_hints_lookup(), and linker_lookup_file().

Here is the call graph for this function:



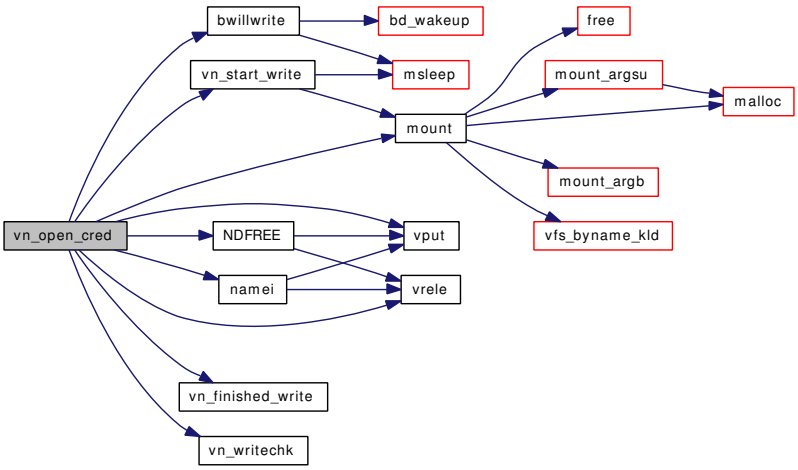
9.164.1.16 int vn_open_cred (struct nameidata * *ndp*, int * *flagp*, int *cmode*, struct ucred * *cred*, int *fdidx*)

Definition at line 104 of file vfs_vnops.c.

References bwillwrite(), mount(), namei(), NDFREE(), td, vn_finished_write(), vn_start_write(), vn_writechk(), vput(), and vrele().

Referenced by alq_open(), and vn_open().

Here is the call graph for this function:



9.164.1.17 static int vn_poll (struct file * *fp*, int *events*, struct ucred * *active_cred*, struct thread * *td*) [static]

Definition at line 776 of file vfs_vnops.c.

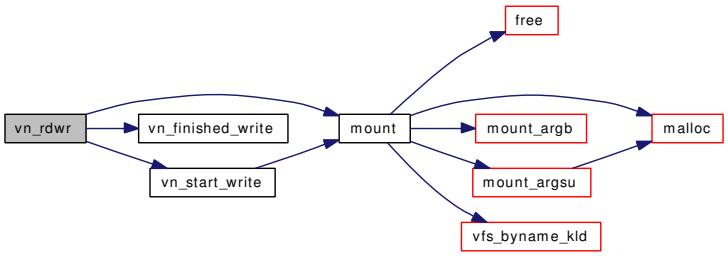
9.164.1.18 int vn_rdwr (enum uio_rw *rw*, struct vnode * *vp*, void * *base*, int *len*, off_t *offset*, enum uio_seg *segflg*, int *ioflg*, struct ucred * *active_cred*, struct ucred * *file_cred*, int * *aresid*, struct thread * *td*)

Definition at line 335 of file vfs_vnops.c.

References mount(), vn_finished_write(), and vn_start_write().

Referenced by acct_process(), kern_sendfile(), link_elf_load_file(), linker_hints_lookup(), and vn_rdwr_inchunks().

Here is the call graph for this function:



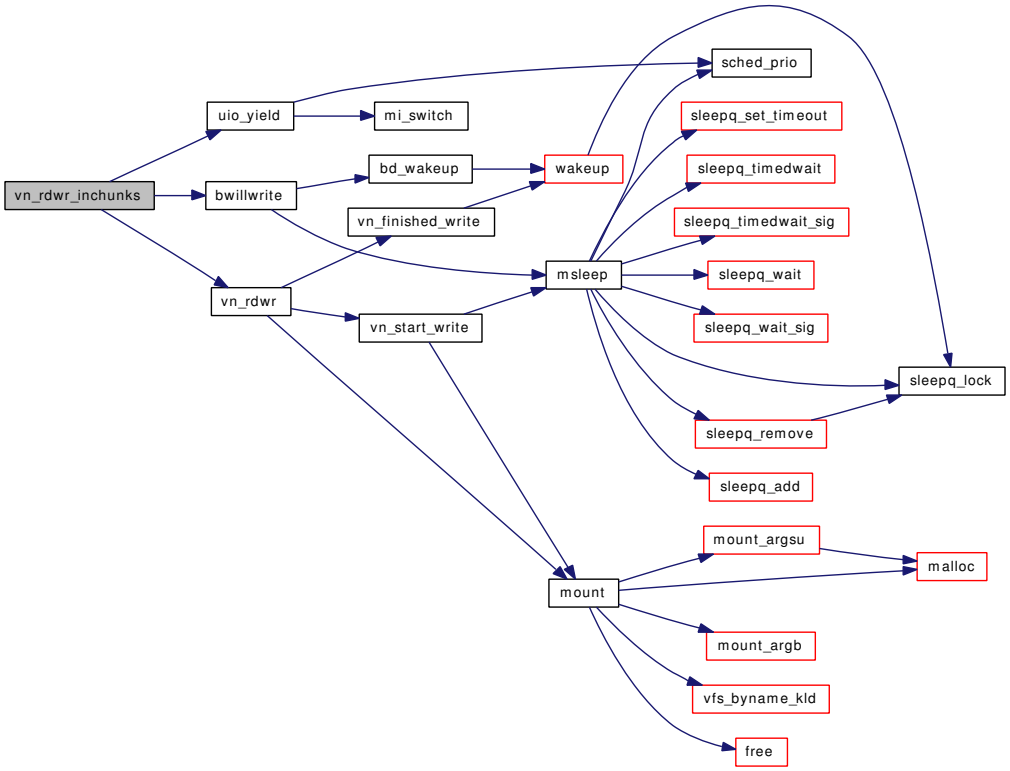
9.164.1.19 int vn_rdwr_inchunks (enum uio_rw *rw*, struct vnode * *vp*, void * *base*, size_t *len*, off_t *offset*, enum uio_seg *segflg*, int *ioflg*, struct ucred * *active_cred*, struct ucred * *file_cred*, size_t * *aresid*, struct thread * *td*)

Definition at line 428 of file vfs_vnops.c.

References bwillwrite(), uio_yield(), and vn_rdwr().

Referenced by coredump().

Here is the call graph for this function:

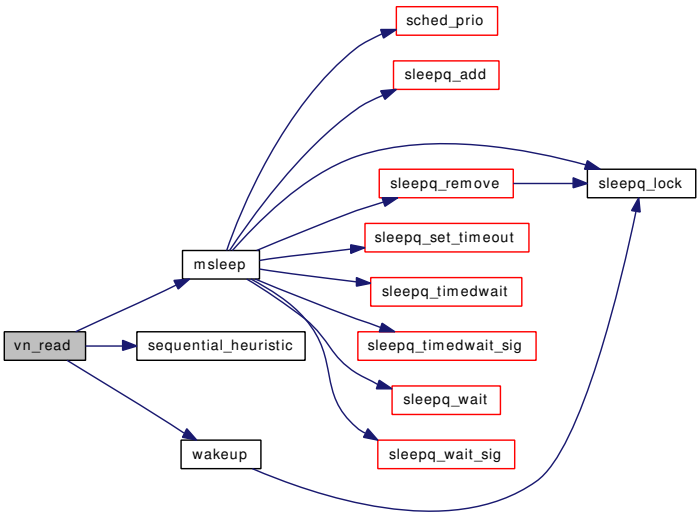


9.164.1.20 static int vn_read (struct file * *fp*, struct uio * *uio*, struct ucred * *active_cred*, int *flags*, struct thread * *td*) [static]

Definition at line 481 of file vfs_vnops.c.

References msleep(), sequential_heuristic(), and wakeup().

Here is the call graph for this function:



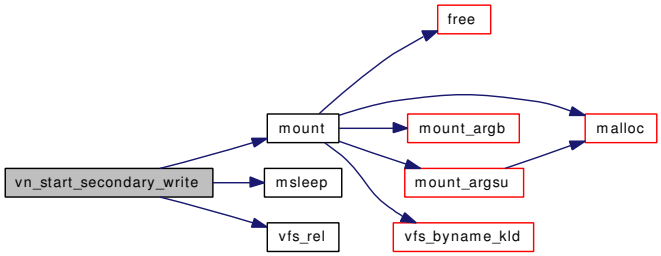
9.164.1.21 int vn_start_secondary_write (struct vnode * *vp*, struct mount ** *mpp*, int *flags*)

Definition at line 990 of file vfs_vnops.c.

References mount(), msleep(), and vfs_rel().

Referenced by vgonel().

Here is the call graph for this function:



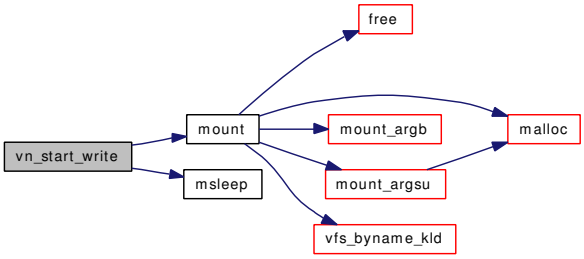
9.164.1.22 int vn_start_write (struct vnode * *vp*, struct mount ** *mpp*, int *flags*)

Definition at line 884 of file vfs_vnops.c.

References mount(), and msleep().

Referenced by aio_fsync_vnode(), alq_doio(), coredump(), dounmount(), extattr_delete_vp(), extattr_set_vp(), extattrctl(), fhopen(), flushbufqueues(), fsync(), ftruncate(), kern_link(), kern_mkdir(), kern_mkfifo(), kern_mknod(), kern_open(), kern_rename(), kern_rmdir(), kern_symlink(), kern_truncate(), kern_unlink(), quotactl(), setfflags(), setfmode(), setfown(), setutimes(), sync(), sync_fsync(), sync_vnode(), uipc_bind(), undelete(), vacl_delete(), vacl_set_acl(), vlrureclaim(), vn_close(), vn_extattr_rm(), vn_extattr_set(), vn_open_cred(), vn_rdwr(), vn_write(), and vtryrecycle().

Here is the call graph for this function:



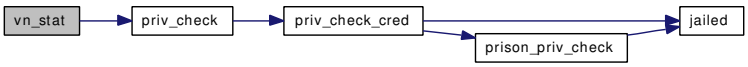
9.164.1.23 int vn_stat (struct vnode * *vp*, struct stat * *sb*, struct ucred * *active_cred*, struct ucred * *file_cred*, struct thread * *td*)

Definition at line 624 of file vfs_vnops.c.

References priv_check().

Referenced by fhstat(), kern_lstat(), kern_stat(), and vn_statfile().

Here is the call graph for this function:



9.164.1.24 static int vn_statfile (struct file * *fp*, struct stat * *sb*, struct ucred * *active_cred*, struct thread * *td*) [static]

Definition at line 601 of file vfs_vnops.c.

References vn_stat().

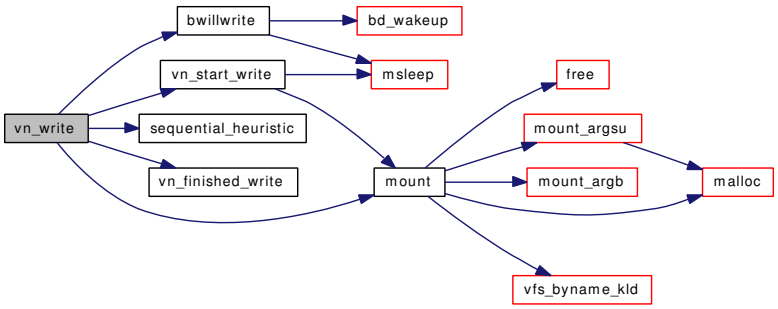
Here is the call graph for this function:

9.164.1.25 static int vn_write (struct file * *fp*, struct uio * *uio*, struct ucred * *active_cred*, int *flags*, struct thread * *td*) [static]

Definition at line 544 of file vfs_vnops.c.

References bwillwrite(), mount(), sequential_heuristic(), vn_finished_write(), and vn_start_write().

Here is the call graph for this function:

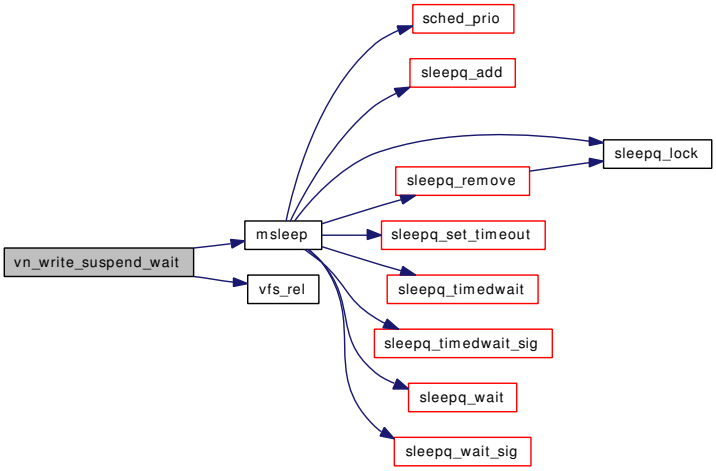


9.164.1.26 int vn_write_suspend_wait (struct vnode * *vp*, struct mount * *mp*, int *flags*)

Definition at line 940 of file vfs_vnops.c.

References msleep(), and vfs_rel().

Here is the call graph for this function:



9.164.1.27 int vn_writechk (struct vnode * *vp*)

Definition at line 255 of file vfs_vnops.c.

Referenced by fhopen(), ftruncate(), kern_truncate(), vn_access(), and vn_open_cred().

9.164.2 Variable Documentation

9.164.2.1 fo_close_t vn_closefile [static]

Definition at line 73 of file vfs_vnops.c.

9.164.2.2 fo_ioctl_t vn_ioctl [static]

Definition at line 69 of file vfs_vnops.c.

9.164.2.3 fo_kqfilter_t vn_kqfilter [static]

Definition at line 71 of file vfs_vnops.c.

9.164.2.4 fo_poll_t vn_poll [static]

Definition at line 70 of file vfs_vnops.c.

9.164.2.5 fo_rdwr_t vn_read [static]

Definition at line 67 of file vfs_vnops.c.

9.164.2.6 fo_stat_t vn_statfile [static]

Definition at line 72 of file vfs_vnops.c.

9.164.2.7 fo_rdwr_t vn_write [static]

Definition at line 68 of file vfs_vnops.c.

9.164.2.8 struct fileops vnops

Initial value:

```
 {
        .fo_read = vn_read,
        .fo_write = vn_write,
        .fo_ioctl = vn_ioctl,
        .fo_poll = vn_poll,
        .fo_kqfilter = vn_kqfilter,
        .fo_stat = vn_statfile,
        .fo_close = vn_closefile,
        .fo_flags = DFLAG_PASSABLE | DFLAG_SEEKABLE
}
```

Definition at line 75 of file vfs_vnops.c.

Referenced by fdcheckstd(), fhopen(), and kern_open().

Index