####### ############
########

####### ############ ########

########: 43126

2013#11#07 ## #####.

########

####### ## ### ####### ############ ########. #### ###### ## # *work in progress* ### ## ### #### ## #### ###########. #### ######## ## ### ### ##### ### #### ## ##### #### ## ##### #### ## ## #######. ## ### ### ########## ## ####### #### #### #######, #### ##### ## ### ####### ############# ####### ####### ####.

######. ## ### #### ## ########## ## # ####### ## ####### ### ########### ####### #### ### ####### ### ####### ## ### ## ### ######## ###– ### #####.

####### ## # ########## ######### ## ### ####### ##########.

#########.

#####, #######, ########, ###, #########, ### ##, #########, ### ######## ### ########## ## ##### ########, ###., ########## ## ### ###### ###### ### ##### #########.

#########, ###########, ######, #######, #######, ####### ##### ### ####### ## ### ###### ###– ####### ########## ## ########## ## ######### ########### ## ### ###### ###### ###/## ##### #########.

##########. ##### ##### ############ ###### ## #### ########, ### ### ####### ####### ### ##### ## ### ######### #####, ### ############ #### #### ######## ## ### ### ## ### ### ######.

#########

############## ### ### ## ###### (### #######) ### ########## ##### (###, ####, ###, ##########, ### ### ## #####) #### ## ####### ###########, ### ######### ######## #### ### ######### ###– ####### ### ###:

1. ############### ## ###### #### (### #######) #### ###### ### ##### ######### ######, #### #### ## ########## ### ### ######### ########## ## ### ##### ##### ## #### #### ##########.
2. ############### ## ######## #### (########### ## ##### ####, ######### ## ###, ##########, ### ### ##### #######) #### ######### ### ##### ######### ######, #### #### ## ########## ### ### ######### ########## ## ### ############# ###/## ##### ######### ######## #### ### ############.

##

#########

##########, #########, ### ### ####### ##, ### ####### ########## ## ############### ### ####### ### # ########## ####### ### ##########. ## ## ##### ##### ### ####### ############# ####### ## ###### ### ### ######, ########, ##########, #######, #########, ## ############# ####### (#########, ### ### ####### ##, ########### ## ########## ##### ## ########; #### ## ###, ####, ## #######; ## ######## ############) ####### ###### ### ## ### ###### ## #########, ####### ## ########, ###### #########, ## #### (######### ########## ## #########) ####### ## ### ### ### ## ### ### ## #### ############, #### ## ####### ## ### ########### ## #### ######.

#########

########

#######

#########

#.

#########

1. ############## ### #######

Contributed by Sergey Lyubka.

1.1.

#######, ######## #### ### #### (########) ####, ## ### ##### #### ####### ########. ##### ### ####### ##### ## ###### ####### ### #### ############ #########, ### ###32 ############ ## #### ## ## #######.

1.2.

#######, ######## ### #### ###### ######, ####### #### # ######## ##### ### ## ## #########, #### ## ######### ####. ### #### ####### ## ####### #### ####### #####:

- #### ####
- boot0 #####
- boot2 #####
- ###### #####
- ###### ##############

boot0 ### boot2 ###### ### #### ######## ## ## *bootstrap stages 1 and 2* ## ####(8) ## ### ##### ##### ## ######### 3###### ############# #########. ####### ##–######### ## ####### ## ### ###### ## #### #####, ## ### ### ######## ######### #### ##### ### ##### #### #######. ###### #### #### ### ###### #### ### ###### #### ####### ## #######:

###### (### ####)	#### (########) ########
F1 FreeBSD F2 BSD F5 Disk 2	boot0
>>FreeBSD/i386 BOOT Default: 1:ad(1,a)/boot/loader	boot2[#]

boot:	
BTX loader 1.0 BTX version is 1.01 BIOS drive A: is disk0 BIOS drive C: is disk1 BIOS 639kB/64512kB available memory FreeBSD/i386 bootstrap loader, Revision 0.8 Console internal video/keyboard (jkh@bento.freebsd.org, Mon Nov 20 11:41:23 - GMT 2000) /kernel text=0x1234 data=0x2345 - syms=[0x4+0x3456] Hit [Enter] to boot immediately, or any other - key for command prompt Booting [kernel] in 9 seconds..._	######
Copyright (c) 1992-2002 The FreeBSD Project. Copyright (c) 1979, 1980, 1983, 1986, 1988, - 1989, 1991, 1992, 1993, 1994 The Regents of the University of - California. All rights reserved. FreeBSD 4.6-RC #0: Sat May 4 22:49:02 GMT - 2002 devnull@kukas:/usr/obj/usr/src/sys/DEVNULL Timecounter -"i8254" frequency 1193182 Hz	######

[#]#### ###### #### ###### ## ### #### ####### # ### #### ##### ######### ## ## ## #### ## ### boot0 #####.

1.3. ####

##, ### ############ ######### ### ### ## #### ########## ###–###. ### ## ### ######### ## ### *instruction pointer* ########, ### ### ##### ##### # ##### ## ## #### #######: ## ## # 32#### ##### ## 0########0. ### ########### ####### ######## ###### ## #### ## ## ######## ## ### #########. ### ## ### ###–###### ## ### cr0 32#### ####### ########, ### ### ##### #### ##### ### ###### ## 0. ### ## ### ##0## ####, ### ### ## (########## #######) ######### ####### ### ######### ## ####### ## ######### ## #### ####. ##### ## #### #### #### ### ## #######, ### ######### ##### ## #### ####. #### #### #####, ##### ##### ######, #### ###### ### ######## ######### ### #########.

0########0 ## ######## #### #### 4##, ## ###### ### ####### ### 4## ######## ######, ## ###### ##### ## # ##### ###### #######. ### ########## ######## ########## #### ####### ## #### ## ###### ## # #### ###### #####.

Basic Input Output System, ### ## ## # #### ## ### ########### #### ### # ########## ##### ###### ## ######### ###### (###). #### ###### ######## ####### ######### ######## #### ### ######## ## ### ######## ######## #### ### ###########. ##, ### ######### #### ##### #### ## ### ####### 0########0,

######. ####### #### ####### ######## # #### ##–######### ## ### ###### #### ########.

Power On Self Test. #### ## # ### ## ######## ######### ### ###### #####, ###### ### ##### ### ##### ######### ##### ## #### ### ### ### ########## ### ######## ########. ### ######### #### ## #### ##### ## ########### ### #### ######. ### ###### ###### ##### ### #### ###### ## ## ### ########, ## ### ### #### #### # ######, ######, ######## ###.

INT 0x19 ###########. #### ########### ##### 512 ##### #### ### ##### ###### ## #### ###### #### ### ###### ## ####### 0#7#00. ### #### *first sector* ########## #### ######### ############, ##### ### ######–## ##### ## ####### ## # ###### ## ########### ######. ###### ### ########, ### ##### ##### ## ####### ## # ###### (####### 64) #######. ##### ###### 0 ## ### ###–###### ## ### ######## #####, ### ###### 1, ### ##### ###### (######, ##, #####–####, ### ######## ######## #### 0, ### ####### # ######## #### 1), ### # ####### #######. ## ## #### ###### ###### #### ######, ## ###. ### ######### ####### ## ### ##### ##### ### ##### #### [1].

1.4. boot0

/boot/boot0. #### ## # ##### 512##### ####, ### ## ## ####### #### ######### ############ ######### ##### ## #### ########### ### ## ### ##### ### ############# ###### ## ############ ####.

##########, ### INT 0x19 ########### ##### ## ###, #.#., ### boot0 ###–####, #### ### ###### ## ####### 0#7#00. ###### # #### ## ### #### sys/boot/i386/boot0/boot0.S ### #### # ##### ## #### ## ######### ##### # #### ## ### #### #######, ##### ## ## ####### ##### ## #### ####### ## ###### #######.

###, ##, boot0, ### # ####### ######### ######## #### ###### 0#1##, ###### ### *partition table*. ## ### 4 ####### ## 16 ##### ####, ###### *partition records*, ##### ###–###### ### ### ########(#) ### ###########, ##, ## ######### ###########, ######. ### #### ## ##### 16 #### ####### # ######### (#####) ## ######## ## ###. ####### ### ###### #### #### #### #### ###, ######### boot0## #### #### ###### ## #######.

######:

- ### 1##### ########## ####
- ### 1##### ######## ####

[1] #### ######### #### ## #########(8) ### ##### ### ########### ## #### ####, ###### ## ### ###### ######.

- ### 6 #### ########## ## ### ######
- ### 8 #### ########## ## ### ######

###-###### ####### ## ### #####. #### ###########, ### ### ###, ######## ### ##### ##-#########, ### ## ######### ####: ### (####### ##### ###########) ### ### ######## ###### ### ### ######### ### ### ########### ######, ##### ### (######## #### ######) ### ########### ### ### ##### ### #### ####### ## ### #########.

####. ## ######## ## ########### ###, boot0 ######## ### ######### #######:

- ######## ### ######### #### ### ### ######## ######### ## #### ## ########, ### ###### ### ########
- ##### ###### ## #### ## ######## #### ########## (#####) ### #### ######## ## ## ### ## ## ### ####### ## ### #### ####
- ##### ### ##### ###### ## ### ######## ######### (#####) #### ###### ### ##### #####

(#####), ## ### ####, # ####### #####? ## ### ### #### ####### #######, ## ## boot2.

1.5. boot2

######, ### boot2 ##### ##### boot0, ### ### ####1. ########, ##### ## # 512##### #### ###### boot1 ## ### ######### /boot ## ####. ## ## #### ### ####### #### # ######. #### ####### #### # ######, boot1 ##### ### #### #### ## boot0 ### # ########: ## ####### boot2 ### #### ##.

/boot/mbr ###### ## ####. ## ## # ########## ###-#### ## boot0. ### #### ## mbr #### ### ####### # #### ### ### ####, ## #### ####### ##### ### ######### ###### ######.

boot2 ####### ## sys/boot/i386/boot2/, ### ### ########## ###### ## ## /boot. ### ##### boot0 ### boot2 #### ### ## /boot ### ### ##### ## ### #########, ### ## ######### #### ## ####0###. ### ###### ######## ### boot0 ## ## ### ###. ### boot2 ## ## ### ######### ## # ######## ####### #####. ##### ######### ### ### ##### ### ############ #######, ## #### ### ######### ## ######## #### ##.

boot2 ## ## #### ### #### /boot/loader, ##### ## ### ##### ##### ## ### ############# #########. ### #### ## boot2 ###### ### ### ######## #### open()

####### 1. ############# ### ######
##############

read(), ##### ### ###### ## ### ### ######. ## #### #### ### ########, ####### ##### ### ########## #########, #### ### #### /boot/loader, #### ## #### ###### ##### # #### #######, ### #### #### ### ######### ## ### ######## ##### #####.

####### ####, boot2 ####### ### #### ##### ## ### ###### ### ## ###### #### ###–###### ####, ####, ##### ### #########.

boot2 ###### ## ####### ## ####### ###:

```
sys/boot/i386/boot2/Makefile:
boot2.ld: boot2.ldr boot2.bin ${BTXKERN}
 btxld --v --E ${ORG2} --f bin --b ${BTXKERN} --l boot2.ldr \
    --o ${.TARGET} --P 1 boot2.bin
```

#####(8) ## #### ## #### ### ######. ###, ##### ###### ### #### ########, ## # ##### ## #### #### ######## # ######### #### ##–######### ### ### #######, ###### ### ######, #### ## ## ###### ####. ## boot2 ## # ### ######, #.#., ## #### ### ####### ######## ## ###.

######. ## ##### ### ######## ########. ### ########## ####### #####(8) ### ##(1) ## #### ## ####### ##### ###### ##### #### # ###### ###### ## ##########, ##### ##### ##### ## ###### #### #### ### ###, ######### ### ###### #### ######## ## ## ### ## ### ######### ## ### ######### ### ### ###### ####.

boot0 ###### ### ######### ## ##### ##### #####. ### #### ######## ### ######### ## ######### ####, ### ######## # ###### ########### ###### ####### ### ######. #### ########:

- ####### #86 ####. #### #####, ### ### ## # #86 #######. #### #### ############ #### #####, ####, ###, ###, ## ###### ## ### ######, #### ####.
- ######### ########## ##### (###) ## ### ## ## ### ######## ########## ### ###### ## ### ####### ###### ########, ### ######### 0#30 ## ### ## ## ## ### ####### ####.
- ### ###### #####: exec ### exit, ### #######:

```
sys/boot/i386/btx/lib/btxsys.s:
  .set INT_SYS,0x30  # Interrupt number
#
# System call: exit
#
__exit:  xorl %eax,%eax   # BTX system
  int $INT_SYS   #  call 0x0
#
# System call: exec
#
```

```
__exec:  movl $0x1,%eax   # BTX system
  int $INT_SYS   #  call 0x1
```

(###):

```
sys/boot/i386/btx/btx/btx.s:
gdt:  .word 0x0,0x0,0x0,0x0  # Null entry
  .word 0xffff,0x0,0x9a00,0xcf # SEL_SCODE
  .word 0xffff,0x0,0x9200,0xcf # SEL_SDATA
  .word 0xffff,0x0,0x9a00,0x0 # SEL_RCODE
  .word 0xffff,0x0,0x9200,0x0 # SEL_RDATA
  .word 0xffff,MEM_USR,0xfa00,0xcf# SEL_UCODE
  .word 0xffff,MEM_USR,0xf200,0xcf# SEL_UDATA
  .word _TSSLM,MEM_TSS,0x8900,0x0 # SEL_TSS
```

###_### (0##000), ### # ######## (###_#####) ###### ## ### ######## #### #######. ### ###_##### ########## ### ########## ######### ##### (###) 3, ##### ## ### ###### ######### #####. ### ### INT 0x30 ########### ####### ####### ## # ####### ####### ## ## ### ###_##### (########## ####) ########, ## ##### #### ### #### #### ####### ## ###:

```
  mov $SEL_SCODE,%dh  # Segment selector
init.2:  shr %bx    # Handle this int?
  jnc init.3   # No
  mov %ax,(%di)   # Set handler offset
  mov %dh,0x2(%di)  #  and selector
  mov %dl,0x5(%di)  # Set P:DPL:type
  add $0x4,%ax   # Next handler
```

##, #### ### ###### ##### __exec(), ### #### #### ## ######## #### ### ####### ####– ######. #### ###### ### ###### ## ###### ### ######### #### #### ##########, #### ## #### ######, ###, ###, ### #####, ## ######.

boot2 ####### ## ######### #########, struct bootinfo. #### ######### ## ########### ## boot2 ### ###### ## ### ######, ### #### ####### ## ### ######. #### ##### ## #### ########## ### ### ## boot2, ### #### ## ### ######. #### #########, ##### ##### ###########, ######## ### ###### ########, #### ######## ########, #### ##### ###### ### #### ######, ######## ###### #########, envp ####### ###. ### ########## ### ## ##:

```
/usr/include/machine/bootinfo.h:
struct bootinfo {
 u_int32_t bi_version;
 u_int32_t bi_kernelname;  /* represents a char * */
 u_int32_t bi_nfs_diskless; /* struct nfs_diskless * */
    /* End of fields that are always present. */
#define bi_endcommon bi_n_bios_used
 u_int32_t bi_n_bios_used;
 u_int32_t bi_bios_geom[N_BIOS_GEOM];
 u_int32_t bi_size;
```

```
 u_int8_t bi_memsizes_valid;
 u_int8_t bi_bios_dev;  /* bootdev BIOS unit number */
 u_int8_t bi_pad[2];
 u_int32_t bi_basemem;
 u_int32_t bi_extmem;
 u_int32_t bi_symtab;  /* struct symtab * */
 u_int32_t bi_esymtab;  /* struct symtab * */
   /* Items below only from advanced bootloader */
 u_int32_t bi_kernend;  /* end of kernel space */
 u_int32_t bi_envp;  /* environment */
 u_int32_t bi_modulep;  /* preloaded modules */
};
```

boot2 ###### #### ## ######### #### ####### ### #### #####, #### ##### load(). ## ### #### #### ### ##### ########, ### #### ###### ## # #######, ## load() #### #### ### ####### #### (/boot/loader). ######### ino_t lookup(char *filename) ### int xfsread(ino_t inode, void *buf, size_t nbyte) ### #### ## #### ### ####### ## # #### #### ######. /boot/loader ## ## ### ######, ### ##### ### ### ###### ## ######### #### #.##### struct exec #########. load() ##### ### ######## ### ######, ####### ### ####### ## /boot/loader #### ######, ### ####### ### ######### ## ### ######## #####:

```
sys/boot/i386/boot2/boot2.c:
   __exec((caddr_t)addr, RB_BOOTINFO -| (opts & RBX_MASK),
   MAKEBOOTDEV(dev_maj[dsk.type], 0, dsk.slice, dsk.unit, dsk.part),
   0, 0, 0, VTOP(&bootinfo));
```

1.6. ######

####. # #### ### ######## ## #### ## ######, ##### ## # ############# ####### ####### ## #### #####, ######(8). ### ########## #####–##### ### ### #### ######### #####.

######. #### ### ###### ## ###### #### ######, ## ## ##### ###### ## ### ######:

```
sys/boot/common/boot.c:
   -/* Call the exec handler from the loader matching the kernel */
   module_formats[km->m_loader]->l_exec(km);
```

1.7. ######

######. #### #### #### ######## ### ##### ######## ##### ### ###### ###### ######### ## ### ######. #### ####–#### ## ### ######## ###### ##### #####.

```
sys/conf/Makefile.i386:
ld --elf --Bdynamic --T -/usr/src/sys/conf/ldscript.i386  --export-dynamic \
```

```
-dynamic-linker -/red/herring --o kernel --X locore.o \
<lots of kernel -.o files>
```

####. #####, ### ###### ## ## ### ########### ###### ######, ### ### ####### ###### ### ###### ## /red/herring, ##### ## ########## # ##### ####. ######, ###### # #### ## ### #### sys/conf/ldscript.i386 ##### ## #### ##### #### ## ####### ### #### #### ######### # ######. ####### ####### ### ##### ### #####, ###

```
sys/conf/ldscript.i386:
ENTRY(btext)
```

#######. #### ###### ## ####### ## locore.s:

```
sys/i386/i386/locore.s:
 .text
/**********************************************************************
 *
 * This is where the bootblocks start us, set the ball rolling...
 *
 */
NON_GPROF_ENTRY(btext)
```

#####, ### ######## ###### ## ### ## # ########## ##### ## 0#00000002. #### ### ### ####### ######### ### ###########:

```
sys/i386/i386/locore.s:
/* Don't trust what the BIOS gives for eflags. */
 pushl $PSL_KERNEL
 popfl

/*
 * Don't trust what the BIOS gives for %fs and %gs.  Trust the bootstrap
 * to set %cs, %ds, %es and %ss.
 */
 mov %ds, %ax
 mov %ax, %fs
 mov %ax, %gs
```

recover_bootinfo(), identify_cpu(), create_pagetables(), ##### ### #### ####### ## locore.s. #### ## # ########### ## #### #### ##:

recover_bootinfo	#### ####### ###### ### ########## ## ### ###### ###### #### ### ####–#####. ### ###### ### #### #### ####–## ## 3 ####: ## ### ######, ######### #####, ## ### ### #### #### ######, ##

####### 1. ############# ### ######
###############

	## ### ### ######## #### #########. #### ######## ########## ### ####– ### ######, ### ###### ### struct bootinfo ######### #### ### ###### ######.
identify_cpu	#### ######### ##### ## #### ### #### ### ## ## ####### ##, ####### ### ##### ##### ## # ######## _cpu.
create_pagetables	#### ######## ######### ### ##### ### # #### ##### ######### ## ### ### ## ### ###### ###### ####.

###, ## ### ### ######## ##:

```
 testl $CPUID_VME, R(_cpu_feature)
 jz 1f
 movl %cr4, %eax
 orl $CR4_VME, %eax
 movl %eax, %cr4
```

####, ######## ######:

```
/* Now enable paging */
 movl R(_IdlePTD), %eax
 movl %eax,%cr3   /* load ptd addr into mmu */
 movl %cr0,%eax   /* get control word */
 orl $CR0_PE|CR0_PG,%eax  /* enable paging */
 movl %eax,%cr0   /* and let's page NOW! */
```

###, ## ### #### ## ###### ## ######## ### ######### ## ########### ####### #####:

```
 pushl $begin    /* jump to high virtualized address */
 ret

/* now running relocated at KERNBASE where the system is linked to run */
begin:
```

init386() ## ###### #### # ####### ## ### ##### #### ######## ####, ##### #### mi_startup(). init386 ## ## ############ ######### ############## ########, ### mi_startup() ## ## ############ ########### ### (### ###_# ###### ###### ### ####### ###########). ### ###### ##### ####### #### mi_startup(), ### ## ####### ##, ### ###### ######## #######:

```
sys/i386/i386/locore.s:
 movl physfree, %esi
 pushl %esi    /* value of first for init386(first) */
 call _init386   /* wire 386 chip for unix operation */
 call _mi_startup   /* autoconfiguration, mountroot etc */
 hlt  /* never returns to here */
```

1.7.1. **init386()**

init386() ## ####### ## sys/i386/i386/machdep.c ### ######## ######### ############## ######## ## ### #386 ####. ### ###### ## ######### #### ### ######### ## ### ######. ### ###### ### ####### ### #### ##### ####, ## ##### ### ###### ######### ## #######. ###### ####### ## ### ####, ######## ### ##### ### ######### #### ######## ## ########## ######### #### #########:

- ########## ### ###### ####### ##########, ###### #### ### ############# ###–####.
- ####### ### ###.
- ####### ### ###.
- ########## ### ###### #######.
- ########## ### ###, ## ## ## ######## #### ######.
- ########## ### ###.
- ####### ### ###.
- ### ## ####0## ###.

init386() ########### ### ####### ########## ###### #### ######### ## ####### ### ########### ####### (####) ### ####### init_param1(). ### #### ####### ### #### ###### #### ###### ## ### bootinfo #########:

```
sys/i386/i386/machdep.c:
  kern_envp = (caddr_t)bootinfo.bi_envp + KERNBASE;

 /* Init basic tunables, hz etc */
 init_param1();
```

init_param1() ## ####### ## sys/kern/subr_param.c. #### #### ### # ###### ## #######, ### ### #########, init_param1() ### init_param2(), #### ### ###### #### init386():

```
sys/kern/subr_param.c:
 hz = HZ;
 TUNABLE_INT_FETCH("kern.hz", &hz);
```

#######_<########>_##### ## #### ## ##### ### ##### #### ### ###########:

```
/usr/src/sys/sys/kernel.h:
#define TUNABLE_INT_FETCH(path, var) getenv_int((path), (var))
```

kern.hz ## ### ###### ##### ####. ############, ##### ####### ### ### ## init_param1(): kern.maxswzone, kern.maxbcache, kern.maxtsiz, kern.dfldsiz, kern.maxdsiz, kern.dflssiz, kern.maxssiz, kern.sgrowsiz.

init386() ######## ### ###### ########### ##### (###). ##### #### ## ## #86 ## ####### ## ### ### ####### ####### #####, ### #### ##### ## ######### ## # #######:###### ####. ###, ### ########, ### ####### ########### ## ## ######## ## ### ######### #### ## ##:###, #### ### ###### ####### ####### ### #### #######–#### ##### ## #### ####### ####### ## #### ####### ### + ###. ### ###########, ######## ##### ## ####### ####### 0 ### ### ## # 4## ########. #########, ### ############# ###### ####### ####### ### #### ####### ##### #### ## ### ##### ## ###. ####### ######### #### ## ##, ## ### ### ### #########, #.#., #######, #### ### (## ## #### #######, ## ##### ## ### # ######## ######, ### ### ##### ##### ## # ########). ######### ### ##### ########### ### 15 ######### ### ###:

```
sys/i386/i386/machdep.c:
union descriptor gdt[NGDT * MAXCPU]; /* global descriptor table */

sys/i386/include/segments.h:
/*
 * Entries in the Global Descriptor Table (GDT)
 */
#define GNULL_SEL 0 /* Null Descriptor */
#define GCODE_SEL 1 /* Kernel Code Descriptor */
#define GDATA_SEL 2 /* Kernel Data Descriptor */
#define GPRIV_SEL 3 /* SMP Per-Processor Private Data */
#define GPROC0_SEL 4 /* Task state process slot zero and up */
#define GLDT_SEL 5 /* LDT -- eventually one per process */
#define GUSERLDT_SEL 6 /* User LDT */
#define GTGATE_SEL 7 /* Process task switch gate */
#define GBIOSLOWMEM_SEL 8 /* BIOS low memory access (must be entry 8) */
#define GPANIC_SEL 9 /* Task state to consider panic from */
#define GBIOSCODE32_SEL 10 /* BIOS interface (32bit Code) */
#define GBIOSCODE16_SEL 11 /* BIOS interface (16bit Code) */
#define GBIOSDATA_SEL 12 /* BIOS interface (Data) */
#define GBIOSUTIL_SEL 13 /* BIOS interface (Utility) */
#define GBIOSARGS_SEL 14 /* BIOS interface (Arguments) */
```

##########, ### #### # ##### ##### ## # ########, ## #### ### ####### ### ####### ## ### ###. ### #######, ## ###### ######## ### ### ###### #### (#####_###) ### ### ##### 0#08.

(###). #### ##### ## ########## ## ### ######### #### # ######## ## ######## ######### ######. ### #######, ## #### # ###### ####, #### ########### ###### ### INT 0x80 ###########. #### ## # ######## #########, ## ### ########### ######## ##### ## # ###### #### ##### 0#80 ## ### ###. #### ###### ###### ## ### ####### #### ####### #### ##–#######, ## #### ########## ####, #### #### ## ### ######## ####### ####. ### ### ### #### # ####### ## 256 (0#100) #######. ### ###### ######### #### ####### ### ### ###, ##### #### ## ### ####### (256):

```
sys/i386/i386/machdep.c:
```

```
static struct gate_descriptor idt0[NIDT];
struct gate_descriptor *idt = &idt0[0]; /* interrupt descriptor table */
```

#########, ## ########### ####### ## ###. ### ####### #### ### INT 0x80 ## ### ## ####:

```
sys/i386/i386/machdep.c:
  setidt(0x80, &IDTVEC(int0x80_syscall),
   SDT_SYS386TGT, SEL_UPL, GSEL(GCODE_SEL, SEL_KPL));
```

INT 0x80 ###########, ####### #### ######## ## ### ######## _Xint0x80_syscall, ##### ## ## ### ###### #### ####### ### #### ## ######## #### ########## ##########.

####### ### ### ### #### ###########:

```
sys/i386/i386/machdep.c:
 cninit();
/* skipped */
#ifdef DDB
 kdb_init();
 if (boothowto & RB_KDB)
  Debugger("Boot flags requested debugger");
#endif
```

#86 ######### #### #########, ### ### ## #### ## ### ######## ## ##### #### ########### #### # #### ###### ######.

####. ####### ######### ### ####### ## ##### ## ### ###, #### ### ### ###### #### ##### ### ### #### #### ### #### #########:

```
/usr/include/machine/segments.h:
#define LSYS5CALLS_SEL 0 /* forced by intel BCS */
#define LSYS5SIGR_SEL 1
#define L43BSDCALLS_SEL 2 /* notyet */
#define LUCODE_SEL 3
#define LSOL26CALLS_SEL 4 /* Solaris >= 2.6 system call gate */
#define LUDATA_SEL 5
/* separate stack, es,fs,gs sels -? */
/* #define LPOSIXCALLS_SEL 5*/ /* notyet */
#define LBSDICALLS_SEL 16 /* BSDI system call gate */
#define NLDT  (LBSDICALLS_SEL + 1)
```

####, ####0## ####### ####### ##### (struct pcb) ######### ## ###########. ####0 ## # struct proc ######### #### ######### # ###### #######. ## ## ###### ####### ##### ### ###### ## #######, ######### ## ## ######## ## ######:

```
sys/kern/kern_init.c:
    struct proc proc0;
```

struct pcb ## # #### ## # #### #########. ## ## ####### ## /usr/include/machine/pcb.h ### ### # ######### ########### ######## ## ### #386 ############, #### ## ######### ######.

1.7.2. mi_startup()

###:

```
sys/kern/init_main.c:
 for (sipp = sysinit; *sipp; sipp++) {

  /* -... skipped -... */

  /* Call function */
  (*((*sipp)->func))((*sipp)->udata);
  /* -... skipped -... */
 }
```

######## ### ####### ######### ## ######### ## ### ########### ########, # #### ####### ### ######### ## ##.

(####### ######) ## ####### ## ####### # #######() #####. ### ## #### ## ####### ## announce ####### ######. #### ###### ###### ### ######### #######:

```
sys/kern/init_main.c:
static void
print_caddr_t(void *data __unused)
{
 printf("%s", (char *)data);
}
SYSINIT(announce, SI_SUB_COPYRIGHT, SI_ORDER_FIRST, print_caddr_t, copyright)
```

##_###_######### (0#0800001), ##### ##### ##### ##### ### ##_###_####### (0#0800000). ##, ### ######### ####### #### ## ####### ### #####, #### ##### ### ####### ##############.

SYSINIT() ####. ## ####### ## # C_SYSINIT() #####. ### C_SYSINIT() ##### #### ####### ## # ###### struct sysinit ######### ########### #### ####### DATA_SET ##### ####:

```
/usr/include/sys/kernel.h:
     #define C_SYSINIT(uniquifier, subsystem, order, func, ident) \
     static struct sysinit uniquifier ## _sys_init = { \ subsystem, \
     order, \ func, \ ident \ -}; \ DATA_SET(sysinit_set,uniquifier ##
     _sys_init);

#define SYSINIT(uniquifier, subsystem, order, func, ident) \
 C_SYSINIT(uniquifier, subsystem, order,   \
```

```
 (sysinit_cfunc_t)(sysinit_nfunc_t)func, (void *)ident)
```

DATA_SET() ##### ####### ## # MAKE_SET(), ### #### ##### ## ### ##### ##### ### ### ####### ##### ## ######:

```
/usr/include/linker_set.h:
#define MAKE_SET(set, sym)      \
 static void const * const __set_##set##_sym_##sym = &sym; \
 __asm(".section -.set." #set -",\"aw\"");    \
 __asm(".long -" #sym);      \
 __asm(".previous")
#endif
#define TEXT_SET(set, sym) MAKE_SET(set, sym)
#define DATA_SET(set, sym) MAKE_SET(set, sym)
```

####, ### ######### ########### #### #####:

```
static struct sysinit announce_sys_init = {
 SI_SUB_COPYRIGHT,
 SI_ORDER_FIRST,
 (sysinit_cfunc_t)(sysinit_nfunc_t)  print_caddr_t,
 (void *) copyright
};

static void const *const __set_sysinit_set_sym_announce_sys_init =
    &announce_sys_init;
__asm(".section -.set.sysinit_set" -",\"aw\"");
__asm(".long -" -"announce_sys_init");
__asm(".previous");
```

__asm ########### #### ###### ## ### ####### ###### ### ######## ##########. #### #### ###### ## ###### #### ####. ### ####### #### #### ### #### .set.sysinit_set. ### ####### ## #### ####### ## ### 32#### #####, ### ####### ## ########_###_#### #########, ### #### ## #### ### ###### __asm ##. ### ##### __asm ########### ##### ### ### ## # #######. ## # ######### #### ### #### ####### #### ######## ######, ### #######, #.#., ### 32#### #####, #### ## ######## ## ### ##–###### #######, ## ####### ## ##### ## 32#### ########.

####### ####### ## # ###### ######, ### ### ###### ### ######## ## #### ##### ########:

```
% objdump --h -/kernel
  7 -.set.cons_set 00000014  c03164c0  c03164c0  002154c0  2**2
               CONTENTS, ALLOC, LOAD, DATA
  8 -.set.kbddriver_set 00000010  c03164d4  c03164d4  002154d4  2**2
               CONTENTS, ALLOC, LOAD, DATA
  9 -.set.scrndr_set 00000024  c03164e4  c03164e4  002154e4  2**2
               CONTENTS, ALLOC, LOAD, DATA
 10 -.set.scterm_set 0000000c  c0316508  c0316508  00215508  2**2
               CONTENTS, ALLOC, LOAD, DATA
 11 -.set.sysctl_set 0000097c  c0316514  c0316514  00215514  2**2
```

```
              CONTENTS, ALLOC, LOAD, DATA
 12 -.set.sysinit_set 00000664  c0316e90  c0316e90  00215e90  2**2
              CONTENTS, ALLOC, LOAD, DATA
```

.###.#######_### ####### ## 0#664 #####, ## 0x664/sizeof(void *) ####### ####### ### ######## #### ### ######. ### ##### ###–##### #### ## .set.sysctl_set ######### ##### ###### ####.

struct linker_set ### ####### ## .set.sysinit_set ####### #### ## ########### #### #### ########:

```
sys/kern/init_main.c:
     extern struct linker_set sysinit_set; -/* XXX */
```

struct linker_set ## ####### ## #######:

```
/usr/include/linker_set.h:
  struct linker_set {
 int ls_length;
 void *ls_items[1];  /* really ls_length of them, trailing NULL */
};
```

#######, ### ### ###### #### #### ## # ############### ##### ## ######## ## ####.

######### ## ### mi_startup() ##########, ## ## #### ## ##### ###, ### ### ####### ####### ### ##### #########. ### mi_startup() ######## ##### #### ### ##### ####. ### #### #### ###### ## ### ###### #########:

```
/usr/include/sys/kernel.h:
enum sysinit_sub_id {
 SI_SUB_DUMMY  = 0x0000000, /* not executed; for linker*/
 SI_SUB_DONE  = 0x0000001, /* processed*/
 SI_SUB_CONSOLE  = 0x0800000, /* console*/
 SI_SUB_COPYRIGHT = 0x0800001, /* first use of console*/
...
 SI_SUB_RUN_SCHEDULER = 0xfffffff /* scheduler: no return*/
};
```

sys/vm/vm_glue.c, ### ### ##### ##### ### #### ###### ## scheduler(). #### ######## ## ######## ## ####–#### ####, ### ## ########## # ####### #### ### 0, ### ####### #######. ### ####0 #########, ######### ######, ## #### ## ######## ##.

#######, ###### *init*, ## ####### ## ### ####### ###### init:

```
sys/kern/init_main.c:
static void
create_init(const void *udata __unused)
{
 int error;
```

```
 int s;

 s = splhigh();
 error = fork1(&proc0, RFFDG -| RFPROC, &initproc);
 if (error)
  panic("cannot fork init: %d\n", error);
 initproc->p_flag -|= P_INMEM -| P_SYSTEM;
 cpu_set_fork_handler(initproc, start_init, NULL);
 remrunqueue(initproc);
 splx(s);
}
SYSINIT(init,SI_SUB_CREATE_INIT, SI_ORDER_FIRST, create_init, NULL)
```

create_init() ######### # ### ####### ## ####### fork1(), ### #### ### #### ## ########. #### #### ### ####### ## ######### ### ######### ## ### #########, ### start_init() #### ## ######. #### ######## ## ####### ## init_main.c. ## ##### ## #### ### #### ### init ######, ####### /sbin/init #####, #### /sbin/oinit, /sbin/init.bak, ### ####### /stand/sysinstall:

```
sys/kern/init_main.c:
static char init_path[MAXPATHLEN] =
#ifdef INIT_PATH
    __XSTRING(INIT_PATH);
#else
    -"/sbin/init:/sbin/oinit:/sbin/init.bak:/stand/sysinstall";
#endif
```

2. ########

This chapter is maintained by the FreeBSD SMP Next Generation Project.

#####–#### ################ ###### ### ######. ####### ### ## ######## ### ####### #####. #### ########### ### ## ########## ## ######## ## #######(9) #####. # ### ######### ### ######### ###### ## ###### ##### ###### ########## ## ###### ####.

2.1.

#########. ############, # ##–### ### #### ## ##### ## ### ###### ## # ####. ## ####### ###### ###### ## ###### # ##### #### ## ####### #####, ## #### #### ##### ### ##### ## ########. ## ### ####### ######, ####### ### ##### ## #########.

####### ### ## ############ ########, ### #### ### ######## ## ## #### ### # ##### ###### ## ####. #############, ### ### ### ##### ##### ####### # #####. ## ### #### ## #### # #### ###### # #####, ### # #######(9) ####.

########:

######## ####

######.

####### ####

mtx_init. #### #### ## ######### ## ### ##### ######## ### ####### ###### ### ######## ### ## #### ## ########### ####### ## ### ####### ####.

####

MTX_* #####. ### ####### ### #### #### ## ####### ## ### ####### ## ########### ## #####(9).

MTX_DEF

#####

MTX_SPIN

#####

MTX_RECURSE

#######.

##########

###–#####. ### #### ######### #######, ### #### #### ## ## ### #### ## structure name.member name.

######### #########

######### #### ### #### ## ###### ## #### ##### ## ####.

2.1. #####

######## ####	####### ####	####	##########	######### #########
#####_####	###### #####	MTX_SPIN | MTX_RECURSE	_gmonparam, cnt.v_swtch, cp_time, curpriority, mtx.mtx_blocked, mtx.mtx_contested, proc.p_procq, proc.p_slpq, proc.p_sflag, proc.p_stat, proc.p_estcpu, proc.p_cpticks proc.p_pctcpu, proc.p_wchan, proc.p_wmesg, proc.p_swtime, proc.p_slptime, proc.p_runtime, proc.p_uu, proc.p_su, proc.p_iu, proc.p_uticks, proc.p_sticks, proc.p_iticks, proc.p_oncpu, proc.p_lastcpu, proc.p_rqindex, proc.p_heldmtx, proc.p_blocked, proc.p_mtxname, proc.p_contested, proc.p_priority, proc.p_usrpri, proc.p_nativepri, proc.p_nice,	setrunqueue, remrunqueue, mi_switch, chooseproc, schedclock, resetpriority, updatepri, maybe_resched, cpu_switch, cpu_throw, need_resched, resched_wanted, clear_resched, aston, astoff, astpending, calcru, proc_compare

######## ####	####### ####	####	##########	######### #########
			proc.p_rtprio, pscnt, slpque, itqueuebits, itqueues, rtqueuebits, rtqueues, queuebits, queues, idqueuebits, idqueues, switchtime, switchticks	
##86###_####	###86### #####	MTX_DEF	vm86pcb	vm86_bioscall
#####	#######	MTX_DEF | MTX_RECURSE	###### #####–#####	####
#######_####	######## #####	MTX_SPIN | MTX_RECURSE	callfree, callwheel, nextsoftcheck, proc.p_itcallout, proc.p_slpcallout, softticks, ticks	

2.2. ###### #########

#######. ######### #### ### ###### ## #######(9).

2.2. ###### ######### ####

######## ####	##########
allproc_lock	allproc zombproc pidhashtbl proc.p_list proc.p_hash nextpid
proctree_lock	proc.p_children proc.p_sibling

2.3. ########## #########

####. #######, ### #### ######## ## ### ######### ### ####### ###### ########## ## ######### ## ######(9). #### ### ######### ### ####### #### ###, ######## ##### ############### ########## #### ## ####### ### ########### #### ########## ######### #########.

- mtx.mtx_lock

3. ######

#######, ## *Kobj* ######## ## ############### # ########### ###### ### ### ######. ## #### ### #### ##### ######## ## ####### ### ########### ## ### ## ####### ## ##. #### ###### ########## ## ## ##### ### ####### #### ## ######### ## ### #### ### ####### ######## ###### #############.

3.1.

######
: # ### ## #### # #### ######### # #### ##########.

######
: ## ######### # ########.

#####
: ### ## #### #######.

#########
: # ######## ### ## ### ## #### #######.

3.2. ####

#######. #### ########### ##### # ###### ## ## #### ## # ####### ########. ### ############# ####### ## #### ## ######## ######## ### ###### ###### # ###### ###### #####.

############. ###### ### ### ##### ## ########. ### ########### ##–####### # ##### ### ########## ## ##### ### #####. # ###### ## ## ######## ## #### ###### ########### ###### ### ###### ##### ## ### ##### ## ### ####### #### ## ## ####### ########### ##### ###########. ### ##### ###### ## ## #### # ####–#### ## ######### ## ###### ## ####### ######### ### ############# ######### ### ###### ########### ### # ######. ### ######### ######## ##### ## ### ###### ## ##### ### ###### ## ########## #### ### ###### ########### ## # #### #### ### ##### ########## #### ### ######## #####. ## ### ###### ## ### ###### ### #####–#### ######## ######## ## ### ### ###### ##### ## ##### ### ######. ## ### ###### ## ##### #### ### ########## ######## ###### ### ##### ## ####; #########, ### ####### ######## ########## #### ### ###### ########### ## ####.

#########:

```
object->cache<->class
```

3.3. #####

3.3.1.

```
struct kobj_method
```

3.3.2.

```
void kobj_class_compile(kobj_class_t cls);
void kobj_class_compile_static(kobj_class_t cls, kobj_ops_t ops);
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);
```

3.3.3.

```
KOBJ_CLASS_FIELDS
KOBJ_FIELDS
DEFINE_CLASS(name, methods, size)
KOBJMETHOD(NAME, FUNC)
```

3.3.4.

```
<sys/param.h>
<sys/kobj.h>
```

3.3.5. ######## ## #########

#########. ######## ### ######### ######## ######## # ######## #### ### ###### src/sys/kern/makeobjops.pl ### ### ## ###–##### ### ###### ### #### ### ### ###### ############ ### ###### ###### ####–#####.

####: #include, INTERFACE, CODE, METHOD, STATICMETHOD, ### DEFAULT.

#include ######### ### #### ####### ## ## ###### ######## ## ### #### ## ### ######### #### ####.

#######:

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

INTERFACE ####### ## #### ## ###### ### ######### ####. #### #### ## ###–######### #### #### ###### #### ## [######### ####]_[###### ####]. ### ###### ## ######### [######### ####];.

#######:

```
INTERFACE foo;
```

CODE ####### ###### ### ######### ######## #### ### #### ####. ### ###### ## CODE { [whatever] };

#######:

```
CODE {
 struct foo * foo_alloc_null(struct bar *)
 {
  return NULL;
}
};
```

METHOD ####### ######### # ######. ### ###### ## METHOD [return type] [method name] { [object [, arguments]] };

#######:

```
METHOD int bar {
 struct object *;
 struct foo *;
 struct bar;
};
```

DEFAULT ####### ### ###### ### METHOD #######. ## ####### ### METHOD ### #### ## ####### ### ####### ######## ### ######. ### ######## ###### ## METHOD [return type] [method name] { [object; [other arguments]] }DEFAULT [default function];

#######:

```
METHOD int bar {
 struct object *;
 struct foo *;
 int bar;
} DEFAULT foo_hack;
```

STATICMETHOD ####### ## #### #### ### METHOD ####### ###### ### #### #### ## ### ## ### #### ## ### ###### ######### ## ####### ## ####_# ##### ## #####–####. ####### STATICMETHOD ###### ## ### #### #### ##### ########## ## #####. #### ## #### ###### ### ####### ####### ######## ### ## # ####### ###### #####.

########:

```
src/sys/kern/bus_if.m
src/sys/kern/device_if.m
```

3.3.6. ######## #

#####. # ##### ######## ## # ####, # ##### ## #######, ### ### #### ## ####### ## ###### ###### ########

####. ## ###### ### ##### ### ### ##### DEFINE_CLASS(). ## ###### ### ###### ##### ###### ## ##### ## ####_######_# ########## ## # #### #####. #### ######## ##### ### ## ####### ##### ### ##### KOBJMETHOD().

#######:

```
DEFINE_CLASS(fooclass, foomethods, sizeof(struct foodata));

kobj_method_t foomethods[] = {
 KOBJMETHOD(bar_doo, foo_doo),
 KOBJMETHOD(bar_foo, foo_foo),
 { NULL, NULL}
};
```

##########. ######### ## ### ##### ## ### ###### ## ### #### #### ### ##### ## ## ## ########### # ########## ######### #####, #### ###### #### ## ## ####. #### ### ## ############ ## ######### # struct kobj_ops ### ##### kobj_class_compile_static(); #########, kobj_class_compile() ###### ## ####.

3.3.7. ######## ##

######. #### ###### ###–##### ######## ###### #### #### #### ## ## ### #### ## ## ######. ## #### ## ### ########### ### #### #### ## ######## ### ###### ######## ### #### ### kobj_init() ## ### #### ####### ## ##; #########, ### ### ### kobj_create() ## ######## ### ###–####### ### #### ####### ## ### ###### #############. kobj_init() ### #### ## #### ## ###### ### ##### #### ## ###### ####.

####_######.

#######

```
struct foo_data {
 KOBJ_FIELDS;
 foo_foo;
 foo_bar;
};
```

3.3.8. #######

#####. #### ## ## ###### ## ##### ### #####–#### #### ### ### ###### #### #### # ### ##############. ### ######### #### ###### ## ############ #### ### ###### #### ##### # #_# ####### ####, ### ## ##### ####.

#######, ## ### ######### #### ### ### ### ### ###### ### ### #### ### #### ##### ##:

```
[return value = -] FOO_BAR(object [, other parameters]);
```

3.3.9. ########

kobj_create() ## ## ###### ###### kobj_delete() ### ## ###### ## ##, ### #### # ##### ## ## ###### ##### #### kobj_class_free() ### ## ###### ## ##.

4. ### ####

Evan Sarmiento

<*evms@cs.bu.edu*>

####® #######, root ### ########## #####. #### ######## ##########. ## ## ######## ###### root ## # ######, ## ##### #### ##### ######## ## ### ##########. ## ####### ##### ### ####### ##### ###### ### ##### ## root, ## ##### ## ######## ### ###### ###### ## ## ########. ############, ### ## ##### ######### ## ###### secure levels. #########, ####### ######## ##### ## ####### #### ####### 4.0 ### ##–####, ## # ####### ###### ####(8). #### ####### ## ########### ### #### ####### ############ ## ######### ##### ### ###### ###### ### ####. ### #######, # ###### ####### ###### ###### ######### ####### ### ####, ####### ####### ###### #####, ## ####### ### ###### ## ### #### ###########.

#####. ###### ### ####### ########### ######–#### ####### #### ## ######, ####, ### ######## ###### #####, ## #### ## ## ##–###### ##### root ###### ### ####, ## ## #### ## #########, ### ### # ###########. #### ####### ###### ####### ## ### ######### (###### ####) ## ####. ### #######–#### ## ### ## ### ## # #### ### ### ######## ##### ## #####.

4.1.

######: ### ######## #######, ####(8), ### ### #### #####–###### ###### ### #######: ### ####(2) ###### #### ### ########## ############. # #### ## ########## ### ######## ####### ### #### ### #### ## ########### ###### ### ######.

4.1.1. ########

/usr/src/usr.sbin/jail, ########## ## ### ####, jail.c. ### ####### ##### ##### #########: ### #### ## ### ####, ########, ## #######, ### ### ####### ## ## ########.

4.1.1.1. ####

jail.c, ### ##### ##### # ##### #### ## ### ########### ## ## ######### ######### struct jail j; ##### ### ######## #### /usr/include/sys/jail.h.

jail ######### ##:

```
/usr/include/sys/jail.h:
```

```
struct jail {
        u_int32_t       version;
        char            *path;
        char            *hostname;
        u_int32_t       ip_number;
};
```

###, ##### ## ## ##### ### #### ## ### ######### ###### ## ### ####(8) #######, ### ######, #### ### ### ###### ### #########.

```
/usr/src/usr.sbin/jail/jail.c
char path[PATH_MAX];
...
if (realpath(argv[0], path) == NULL)
    err(1, -"realpath: %s", argv[0]);
if (chdir(path) -!= 0)
    err(1, -"chdir: %s", path);
memset(&j, 0, sizeof(j));
j.version = 0;
j.path = path;
j.hostname = argv[1];
```

4.1.1.2.

####(8) ####### ## ## ## ####### #### ##### ### #### ### ## ######## #### ### #######. ####(8) ########## ### ## ####### ##### #### #### #### ##### ### #### ###### ## ## j (### jail #########).

```
/usr/src/usr.sbin/jail/jail.c:
struct in_addr in;
...
if (inet_aton(argv[2], &in) == 0)
    errx(1, -"Could not make sense of ip-number: %s", argv[2]);
j.ip_number = ntohl(in.s_addr);
```

####_####(3) ######## ########### ### ######### ######### ###### ## ## ##–###### #######, ####### ### ####### #### ### ######### ########.# ### ip_number ###### ## ### jail ######### ## ### #### #### ### ## ####### ###### #### ### in ######### ## ####_####(3) ## ########## #### #### #### ##### ## #####(3).

4.1.1.3. ####### ###

#######, ### ######## ####### ##### ### #######. #### ### ####### ## ########## ####### ###### ### #### ######## ### ####### ##### ##### #####(3).

```
/usr/src/usr.sbin/jail/jail.c
i = jail(&j);
...
if (execv(argv[3], argv + 3) -!= 0)
    err(1, -"execv: %s", argv[3]);
```

###, ### jail() ######## ## ######, ### ### ######## ## ### jail ######### ##### ### #### ###### #### ### ######### ##### ## ### #######. #######, ### ###–#### ### ####### ## ########. # #### ### ####### ### #### ## ########### ###### ### ######.

4.1.2. ######

/usr/src/sys/kern/kern_jail.c. #### ## ### #### ##### ### ####(2) ###### ####, ########### #######, ### ########## ######### ### #######.

4.1.2.1.

kern_jail.c, ### ######### ####### ### #######:

```
/usr/src/sys/kern/kern_jail.c:

int     jail_set_hostname_allowed = 1;
SYSCTL_INT(_security_jail, OID_AUTO, set_hostname_allowed, CTLFLAG_RW,
    &jail_set_hostname_allowed, 0,
    -"Processes in jail can set their hostnames");

int     jail_socket_unixiproute_only = 1;
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");

int     jail_sysvipc_allowed = 0;
SYSCTL_INT(_security_jail, OID_AUTO, sysvipc_allowed, CTLFLAG_RW,
    &jail_sysvipc_allowed, 0,
    -"Processes in jail can use System V IPC primitives");

static int jail_enforce_statfs = 2;
SYSCTL_INT(_security_jail, OID_AUTO, enforce_statfs, CTLFLAG_RW,
    &jail_enforce_statfs, 0,
    -"Processes in jail cannot see all mounted file systems");

int     jail_allow_raw_sockets = 0;
SYSCTL_INT(_security_jail, OID_AUTO, allow_raw_sockets, CTLFLAG_RW,
    &jail_allow_raw_sockets, 0,
    -"Prison root can create raw sockets");

int     jail_chflags_allowed = 0;
SYSCTL_INT(_security_jail, OID_AUTO, chflags_allowed, CTLFLAG_RW,
    &jail_chflags_allowed, 0,
    -"Processes in jail can alter system file flags");

int     jail_mount_allowed = 0;
SYSCTL_INT(_security_jail, OID_AUTO, mount_allowed, CTLFLAG_RW,
    &jail_mount_allowed, 0,
    -"Processes in jail can mount/unmount jail-friendly file systems");
```

######(8) #######. ########## ### ######, ##### ######### ####### ### ########## ## ##### ####. ### #######, ### #### ## ### ##### ###### ## security.jail.set_hostname_allowed.

4.1.2.2. ####(2) ######

#####, ### ####(2) ###### #### ##### ### #########, struct thread *td ### struct jail_args *uap. td ## # ####### ## ### thread ######### ##### ######### ### ####–### ######. ## #### #######, uap ## # ####### ## ### ########## ## ##### # ####### ## ### jail ######### ###### ## ### ######## jail.c ## #########. #### # ######### ### ######## ####### ######, ### ### #### ### ####(2) ###### #### ### ##### # jail ######### ## ### ### ########.

```
/usr/src/sys/kern/kern_jail.c:
/*
 * struct jail_args {
 *  struct jail *jail;
 * -};
 */
int
jail(struct thread *td, struct jail_args *uap)
```

#########, uap->jail ### ## #### ## ###### ### jail ######### ##### ### ###### ## ### ###### ####. ####, ### ###### #### ###### ### jail ######### #### ###### ##### ##–### ### ######(9) ########. ######(9) ##### ##### #########: ### ####### ## ### #### ##### ## ## ## ###### #### ###### #####, uap->jail, ##### ## ##### ##, j ### ### #### ## ### #######. ### jail ######### ####### ## uap->jail ## ###### #### ###### ##### ### ## ###### ## ####### jail #########, j.

```
/usr/src/sys/kern/kern_jail.c:
error = copyin(uap->jail, &j, sizeof(j));
```

jail.h. ## ## ### prison #########. ### prison ######### ## #### ########### ###### ###### #####. #### ## ### ######–#### ## ### prison #########.

```
/usr/include/sys/jail.h:
struct prison {
        LIST_ENTRY(prison) pr_list;                    -/* (a) all prisons */
        int             pr_id;                         -/* (c) prison id */
        int             pr_ref;                        -/* (p) refcount */
        char            pr_path[MAXPATHLEN];           -/* (c) chroot path */
        struct vnode    *pr_root;                      -/* (c) vnode to rdir */
        char            pr_host[MAXHOSTNAMELEN];       -/* (p) jail hostname */
        u_int32_t       pr_ip;                         -/* (c) ip addr host */
        void            *pr_linux;                     -/* (p) linux abi */
        int             pr_securelevel;                -/* (p) securelevel */
        struct task     pr_task;                       -/* (d) destroy task */
        struct mtx      pr_mtx;
      void            **pr_slots;                      -/* (p) additional data */
```

```
};
```

####(2) ###### #### #### ######### ###### ### # prison ######### ### ###### #### ####### ### jail ### prison #########.

```
/usr/src/sys/kern/kern_jail.c:
MALLOC(pr, struct prison *, sizeof(*pr), M_PRISON, M_WAITOK -| M_ZERO);
...
error = copyinstr(j.path, &pr->pr_path, sizeof(pr->pr_path), 0);
if (error)
   goto e_killmtx;
...
error = copyinstr(j.hostname, &pr->pr_host, sizeof(pr->pr_host), 0);
if (error)
    goto e_dropvnref;
pr->pr_ip = j.ip_number;
```

####, ## #### ####### ####### ######### ###### #### ####_######(2), ##### ##–######## ### ######## ## ### # ####### #### ### ####.

```
/usr/src/sys/kern/kern_jail.c:
/*
 * struct jail_attach_args {
 *      int jid;
 * -};
 */
int
jail_attach(struct thread *td, struct jail_attach_args *uap)
```

####. ## ########## #### ####_######(2) #### ### ##, ####### ####–###### ########### ## ######.

#######, #### ###### ####### ###### ## ########## ## ### thread #########, ##### ### ######### ### ######### ## ##### proc ##########. ### ### #### ### ######–##### ## ### thread ### proc ######### ## /usr/include/sys/proc.h. ### #######, ### td ####–#### ## ### ###### #### ## ######## # ####### ## ### ####### ######## thread #####–####, ## ###### ######. ### td_proc ###### ## ### thread ######### ####### ## td ## # ####### ## ### proc ######### ##### ########## ### ####### #### ######## ### ###### ########### ## td. ### proc ######### ######## ####### ##### ### ######## ### ####### ########(p_ucred), ### ####### ######## ######(p_limit), ### ## ##. ## ### ucred ######### ####### ## p_ucred ###### ## ### proc #########, ##### ## # ####### ## ### prison #########(cr_prison).

```
/usr/include/sys/proc.h:
struct thread {
   -...
   struct proc *td_proc;
   -...
};
```

```
struct proc {
   -...
   struct ucred *p_ucred;
   -...
};
/usr/include/sys/ucred.h
struct ucred {
   -...
   struct prison *cr_prison;
   -...
};
```

kern_jail.c, ### ######## jail() #### ##### ######## jail_attach() #### # ##### jid. ### jail_attach() ##### ######## change_root() ## ###### ### #### ######### ## ### ####### #######. ### jail_attach() #### ####### # ### ucred #########, ### ######## ### ###–## ####### ucred ######### ## ### ####### ####### ##### ## ### ############ ##–###### ### prison ######### ## ### ucred #########. #### #### ##, ### ####### ####### ## ########## ## ######. #### ### ###### ####### jailed() ## ###### ## ### ###### #### ### ##### ####### ucred ######### ## ### ########, ## ####### 1 ## #### #### ### ########## ## ######### #### # ####. ### ###### ######## ####### ## ### ### ####### ###### ###### ### ####, ## ### ####### ##### #### ####(8), ## ## ##### ### ####(2) ###### ####. #### # ####### ## ######## ####### ######(2), ## ######## ### ###### ######## ## ### ######## ucred #########, ######### ## ### # ###### ucred #########.

```
/usr/src/sys/kern/kern_jail.c
int
jail(struct thread *td, struct jail_args *uap)
{
...
   struct jail_attach_args jaa;
...
   error = jail_attach(td, &jaa);
   if (error)
      goto e_dropprref;
...
}

int
jail_attach(struct thread *td, struct jail_attach_args *uap)
{
   struct proc *p;
   struct ucred *newcred, *oldcred;
   struct prison *pr;
...
   p = td->td_proc;
...
   pr = prison_find(uap->jid);
...
   change_root(pr->pr_root, td);
...
```

```
    newcred->cr_prison = pr;
    p->p_ucred = newcred;
...
}
```

#######, ### ####(2) ###### #### ##–## crhold() ## ######## ### ########## ### ### ##### ###### #######. ## ########–## #### ### ##### ###### ####### ########## ########## #### ### ######, ## ### ##### ####### ## #### ######.

```
/usr/src/sys/kern/kern_fork.c:
p2->p_ucred = crhold(td->td_ucred);
...
td2->td_ucred = crhold(p2->p_ucred);
```

4.2.

########## ### ###### ##### ### ###### ############ ######## ## ###### #########. #######, ##### ############ #### ##### ####### ### ####### ## ######, ### ## ##, ####### ## #####. ### #######:

```
if (jailed(td->td_ucred))
    return (EPERM);
```

4.2.1. ####

########. ######### ### #### #### ##### ##### ######## ##### #### #### ### ## ###. ### ######### ##### #### #### ######## ###: ######(3), ######(3), ######(3) ### ######(3). #######, # ######### #### ##### #### ####### ####### ### ##### #### ## ## ### ## ##### ## ###### ### ######## ## ####. ### ## ##### ####### ### security.jail.sysvipc_allowed. ## #######, #### ###### ## ### ## 0. ## ## #### ### ## 1, ## ##### ###### ### ##### ####### ## ###### # ####; ########## ##### #### ### #### ##### ## #### ## ###### ######### ####### ### ###### ###########. ### ########## ####### # ####### ### # ###### ## #### ### ####### #### ######## ## ### ###### ######.

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

- msgget(key, msgflg): msgget ####### (### ######## #######) # ####### ########## #### ########## # ####### ##### ### ### ## ##### #########.
- msgctl(msgid, cmd, buf): ##### #### ########, # ####### ### ##### ### ###### ## # ###–#### ##########.
- msgsnd(msgid, msgp, msgsz, msgflg): msgsnd ##### # ####### ## # #######.

- msgrcv(msgid, msgp, msgsz, msgtyp, msgflg): # ####### ######## ######## ##### #### ####–####

#########, ##### ## #### ###–########:

```
/usr/src/sys/kern/sysv_msg.c:
if (!jail_sysvipc_allowed && jailed(td->td_ucred))
    return (ENOSYS);
```

######### ###### ##### ##### ######### ## ########### ######### ## ##### # ### ## ########## ########## ## # ### ## ##########. ######### ########## ####### ####### ### ### ######### #### #########. #######, ####### ####### ## # ####–#####, #### ## ##### ####, #### ##### ##### ### ######### ### ############. ### ######### ######### ###### ##### ### ####### ###### # ####: ######(2), ######(2) ### #####(2).

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

- semctl(semid, semnum, cmd, ...): semctl #### ### ######### cmd ## ### ######### ##### ##–####### ## semid.
- semget(key, nsems, flag): semget ####### ## ##### ## ##########, ############# ## key.

  key and flag take on the same meaning as they do in msgget.
- semop(semid, array, nops): semop ######## # ##### ## ########## ######### ## array, ## ### ### ## ########## ########## ## semid.

######. ######### ### ########### ######## #### #### ##### ## ####### ##### ## ##### ####### ####### ##### ### #### ####### ### ####### #### ###### ## ### ###### ######. ##### ###### ##### ### ####### ###### # ###### ###########: #####(2), #####(2), ######(2) ### ######(2).

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

- shmctl(shmid, cmd, buf): shmctl #### ####### ####### ########## ## ### ###### ###### ###### ########## ## shmid.
- shmget(key, size, flag): shmget ######## ## ####### # ###### ###### ###### ## size #####.
- shmat(shmid, addr, flag): shmat ######## # ###### ###### ###### ########## ## shmid ## ### ####### ##### ## # #######.
- shmdt(addr): shmdt ######## ### ###### ###### ###### ########## ######## ## addr.

4.2.2.

######(2) ###### #### ### ####### ########### ###### ######### ## # ####### ######. ## ##### ## ######### ####### # ####### ###### ## ####### ## ## #######, ## ##### ###### ## ### ## ### ###### security.jail.socket_unixiproute_only ## ###. ## ###, ####### ### #### ####### ## ## ####### ## ### ###### ######### ## ###### PF_LOCAL, PF_INET ## PF_ROUTE. #########, ## ####### ## #####.

```
/usr/src/sys/kern/uipc_socket.c:
int
socreate(int dom, struct socket **aso, int type, int proto,
   struct ucred *cred, struct thread *td)
{
   struct protosw *prp;
...
   if (jailed(cred) && jail_socket_unixiproute_only &&
      prp->pr_domain->dom_family -!= PF_LOCAL &&
      prp->pr_domain->dom_family -!= PF_INET &&
      prp->pr_domain->dom_family -!= PF_ROUTE) {
      return (EPROTONOSUPPORT);
   -}
...
}
```

4.2.3. ######## ######

#######. ### ## ### ########## ## ### #####(8) ####### ## ### ## #### ## # ###### ###########.

4.2.4.

######, #### ## ###, ###, ## ### ####. ## ### #### ### ## ### #### #####: ### ####### ##### 2. ##### ### ####### ######–##### ##### ### ##### ## ##### ## ####### # ###### ####### #### ####### # ###–##### ## # ####### ####### #### ## ### nam ######### ## ###. nam ## # ####### ## # sockaddr #########, ##### ######### ### ####### ## ##### ## #### ### #######. # #### ##### ########## ## #### sockaddr #### ## #### ## # ######## ### ######### ## ### ########### ### ### ###### ## #### ########. ## ### ######## in_pcbbind_setup(), sin ## # ####### ## # sockaddr_in #########, ##### ######## ### ####, #######, ###### ### ###### ###### ## ### ###### ##### ## ## ## #####. #########, #### ######### ### ######### #### #### ## ## #### ## ####### ### ####### #### #### ### ###### ## ### #### ## ##### ### ####### ####### ######.

```
/usr/src/sys/netinet/in_pcb.c:
int
in_pcbbind_setup(struct inpcb *inp, struct sockaddr *nam, in_addr_t *laddrp,
   u_short *lportp, struct ucred *cred)
{
   -...
```

```
    struct sockaddr_in *sin;
    -...
    if (nam) {
        sin = (struct sockaddr_in *)nam;
        -...
        if (sin->sin_addr.s_addr -!= INADDR_ANY)
            if (prison_ip(cred, 0, &sin->sin_addr.s_addr))
                return(EINVAL);
        -...
        if (lport) {
            -...
            if (prison && prison_ip(cred, 0, &sin->sin_addr.s_addr))
                return (EADDRNOTAVAIL);
            -...
        -}
    -}
    if (lport == 0) {
        -...
        if (laddr.s_addr -!= INADDR_ANY)
            if (prison_ip(cred, 0, &laddr.s_addr))
                return (EINVAL);
        -...
    -}
...
    if (prison_ip(cred, 0, &laddr.s_addr))
        return (EINVAL);
...
}
```

prison_ip() ####. prison_ip() ## ##### ##### ####–#####, # ####### ## ### ##########(########### ## cred), ### #####, ### ## ## ##–#####. ## ####### 1 ## ### ## ####### #### ### ###### ## ### #### ## 0 #########. ## ### ### ### ##### ### ####, ## ## ## ###### ## ## ####### ### ######### ## ### ####, ### ######## ## ### ####### ## #### ## #### #######.

```
/usr/src/sys/kern/kern_jail.c:
int
prison_ip(struct ucred *cred, int flag, u_int32_t *ip)
{
    u_int32_t tmp;

    if (!jailed(cred))
        return (0);
    if (flag)
        tmp = *ip;
    else
        tmp = ntohl(*ip);
    if (tmp == INADDR_ANY) {
        if (flag)
            *ip = cred->cr_prison->pr_ip;
        else
            *ip = htonl(cred->cr_prison->pr_ip);
```

```
        return (0);
    -}
    if (tmp == INADDR_LOOPBACK) {
        if (flag)
            *ip = cred->cr_prison->pr_ip;
        else
            *ip = htonl(cred->cr_prison->pr_ip);
        return (0);
    -}
    if (cred->cr_prison->pr_ip -!= tmp)
        return (1);
    return (0);
}
```

4.2.5.

root ##### ###### ### #### ### ### ####### ## ##### ## ###### ### #### #####, #### ## #########, ###########, ### ############ #####, ## ### ########### ## ####### #### 0.

```
/usr/src/sys/ufs/ufs/ufs_vnops.c:
static int
ufs_setattr(ap)
    -...
{
    -...
        if (!priv_check_cred(cred, PRIV_VFS_SYSFLAGS, 0)) {
            if (ip->i_flags
                & (SF_NOUNLINK -| SF_IMMUTABLE -| SF_APPEND)) {
                    error = securelevel_gt(cred, 0);
                    if (error)
                        return (error);
            -}
            -...
        -}
}
/usr/src/sys/kern/kern_priv.c
int
priv_check_cred(struct ucred *cred, int priv, int flags)
{
    -...
    error = prison_priv_check(cred, priv);
    if (error)
        return (error);
    -...
}
/usr/src/sys/kern/kern_jail.c
int
prison_priv_check(struct ucred *cred, int priv)
{
    -...
    switch (priv) {
```

```
    -...
    case PRIV_VFS_SYSFLAGS:
        if (jail_chflags_allowed)
            return (0);
        else
            return (EPERM);
    -...
    -}
    -...
}
```

5. ### ########

####### ## ### ######### ### # ####### #### #### ### ######## #########. ####### ######### #### ## ### ### ####### ############## ## ### ######. ####### ###### ######### ###### ########## ## ## #########, ### #####, #######, ### ######## ## ###### #### #### #### ### ###### ## ### ## ### ####### ## ###### ####### ###### ## #### # ########## ####### ############## ####### ### ######### ### ######. #### ###### #### ###### ###### #######, ######### ###### *KLD's*, ## ## ########## ########, ######, ### ########### ## #### #### ### ###### #### ##### ##### ### ###### ## ####### #######. #### ## ############ ##### ### ####### ####### ### ####### #####.

5.1.

###

> # ###### ######### ## ##### ### ###### ####### ########## ######## #### ########## # ######### ###### ##### #### # ###### ############ ########### #### ## ####.

5.2. #######

####### ###### ## ### ####### ## ### ###### ## #### ###### #### ######## ## ###–##### ######### ########## # ######### ###### ### ##### ## ######## ## # ###### ########## ##### ## ####. #### ###### ######### ## ###### # ####### ####. ####### #### ### ###### #### ## ######## ### #### #### ########## #### ########## #### #####, ########, ### # ####### ## ### #### ## #### ## #### ########.

####### #### ### ########## #### ######## ### ######### ### #########. ### ##### ######## ## # ######### ## ###### ## ####### ##### ### ######### ###–##### ## #########. ####### ########### #### ### ## <sys/kernel.h> ## ### #### #### sysinit_sub_id. ### ###### ######## #### ## ## ####### ##### ###### ### #########. ####### ########### ######### ####### ###### ### ## <sys/kernel.h> ## ### #### #### sysinit_elem_order.

#######. ######## ######## ## ###### #####–## ### ###### ###### #####, ### ######## ######## ## ###### ######## ### ###–### ###### ######. ###### ########## ##### ### ###### ####### ######### ## ###–####### #### ##########, ### ####### ### ####### ########## ######### #### # ####### ## ########## ### ### ##### #### #########. ###### ####### ######

SYSINIT() ########. ####### ####### ### #### ####### #### ### #### ## # ### ######### ###### ### DRIVER_MODULE() ## ####### # ######## #### ####### ### ##–#### ###, ## ## ## #######, ########### ### ######. ## #### ## # ### ###### ###–##### ## ####### ### #### #### SYSINIT() ######. ### ##############, ##### ### ### #### ## # ### #########, ### DEV_MODULE().

5.3. #####

5.3.1.

5.3.1.1.

```
<sys/kernel.h>
```

5.3.1.2.

```
SYSINIT(uniquifier, subsystem, order, func, ident)
SYSUNINIT(uniquifier, subsystem, order, func, ident)
```

5.3.2.

SYSINIT() ##### ####### ### ######### ####### #### ## ######### ####### #### ### ### ####### ## #### ### ######## # ######## ## ###### ####### ### ###### ####. SYSINIT() ##### # ########## #### ####### #### ## ######## ### ########## ######## ######## ####, ### ######### #####, ### ######### ####### #####, ### ######## ## ####, ### ### #### ## #### ### ########. ### ######### #### #### # ######## ####### ########.

####### 5.1. ####### ## # SYSINIT()

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

void foo_null(void *unused)
{
        foo_doo();
}
SYSINIT(foo, SI_SUB_FOO, SI_ORDER_FOO, foo_null, NULL);

struct foo foo_voodoo = {
        FOO_VOODOO;
}

void foo_arg(void *vdata)
{
        struct foo *foo = (struct foo *)vdata;
        foo_data(foo);
```

```
}
SYSINIT(bar, SI_SUB_FOO, SI_ORDER_FOO, foo_arg, &foo_voodoo);
```

SI_SUB_FOO ### SI_ORDER_FOO #### ## ## ## ### sysinit_sub_id ### sysinit_elem_order ###### ## ######### #####. ###### ### ######## #### ## ### #### ### ## ### ######. ### ### #### ### #### ### ########### ### ##### # ####### #### ### ##. #### ####### ##### # ####### #### ##### ## ## ### #### ###### ###### ### ######### #### ###### ###### ###### ##########.

5.2. ####### ## ######### SYSINIT()

```
static void
mptable_register(void *dummy __unused)
{

 apic_register_enumerator(&mptable_enumerator);
}

SYSINIT(mptable_register, SI_SUB_TUNABLES -- 1, SI_ORDER_FIRST,
    mptable_register, NULL);
```

5.3.3.

SYSUNINIT() ##### ####### ######### ## ### SYSINIT() ##### ###### #### ## #### ### ####### #### ## ######### ######## #### ###.

5.3. ####### ## # SYSUNINIT()

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

void foo_cleanup(void *unused)
{
        foo_kill();
}
SYSUNINIT(foobar, SI_SUB_FOO, SI_ORDER_FOO, foo_cleanup, NULL);

struct foo_stack foo_stack = {
        FOO_STACK_VOODOO;
}
```

```
void foo_flush(void *vdata)
{
}
SYSUNINIT(barfoo, SI_SUB_FOO, SI_ORDER_FOO, foo_flush, &foo_stack);
```

6. ### ########### ###

Chris Costello and Robert Watson.

6.1. ### #############

############, ### ######–## ######## ######## ## ####### ##########, ###. ##### #####/###### ######## #66001#01###8035 (#######), ## #### ## ### ##### ##### ######## #######.

############# ### ### ## ###### (#### #######) ### ########## ##### (####, ####, ###, ##########, ### ### ## #####) #### ## ####### ############, ### ######### ######## #### ### ######### ########## ### ###:

1. ############### ## ###### #### (#### #######) #### ###### ### ##### ######### ######, #### #### ## ########## ### ### ######### ########## ## ### ##### ##### ## #### #### ##########.

2. ############### ## ######## #### (########### ## ##### ####, ######### ## ###, ##########, ### ### ##### #######) #### ######### ### ##### ######### ######, #### #### ## ########## ### ### ######### ########## ## ### #######–###### ###/## ##### ######### ######## #### ### ############.

#########

##########, ### ### ### ### ### ####### ## ####### ######–####, #########, ### ### ####### ##, ### ####### ########## ## ############### ### ####### ### # ########## ####### ### ##########. ## ## ##### ##### ######## ########## ####–######, ### ## ###### ### ### ######, ########, ##########, #######, #########, ## ############# ####### (#########, ### ### ####### ##, ########### ## ########## ##### ## ########; #### ## ###, ####, ## #######; ## ######## ############) ###–#### ###### ### ## ### ###### ## #########, ####### ## ###–#####, ###### #########, ## #### (######### ##########

#########) ####### ## ### ### ### ## ### ### ## #### ############# ######, #### ## ####### ## ### ########### ## #### ######.

6.2.

####### ######## ############ ####### ### ####### ######### ###### ####### ########, ## #### ## # ######### ### ###### ######## #############, ### ########## ### #########. ### ### ######### ## # ######### ###### ####### #########, ########## ### ######## ######## ## ## ###### ###### #### ### ######, ###### ## ####, ## ###### ########### ## ########. ### ######### ######## # ####### ## ######## ## #### ## ###### ## ######### ### ######## ########, ######### ### ####### ## ###### ### ######## ###### (#### ## ############### ###########) #### ###### #######.

######.

6.3.

#####. ### ###### ######## ### ## ########### ## ###### ####### ### ###### ## ### ######; ## ######## ###### ####### ### #######, ##### ####### #### ## ########. ### ### ######### ######## # ####### ## ###### ####### ############### ######## ## ###### ###### #######, ######### ####### ### ######### ### ########## ############### ###### ######## ######. #### ####### ## ######### ########## ############.

#######, ## #### ## ######### ######### ## ########### ############, ## ##### ##### ### ### ### ######### ######## ###### ####### ######### ## ### ######.

6.4. ######

######### ###### ####### (###), ###### ## # ### ## ###### ####### ######## #### ### ########### ######## ## ##### ## #### ######### ######. ### ######## ### ## ########## #### ############# ###### ####### (###) ###########, ## ##### #### ############## ##### ### (## ##### ##########) ####### #######. ## ########### #### #######, ### ########### ####### #### ########### ### ###### ####### #####; ### ########### ####### ####### ######## ########## ########## ######### ### #########. # ####### ## ### ######## #### #### ########## ## ######### ###### ######### ### ######## ##########, ######### ### ########### ########

(###) ############### ######, ### #### ######### ######, ########## ###### ###–#### (####), ###### ### #### ########### (###), ### #### ########### (##). #### ##### ##### ######### ## # ####### ## #######, ######### #### ########, ####, ### ######## #########, ## #### ## ######## ###### ## ####### ########### ######## #### ## #### ########### ### #########.

##–####### ### ## ##### ########, ## #### ## # ##### ##### ## ###### ######### ####–####, ##### ### ### ######## ######## ##########, #### ## #### ### ##### ###, ## #### ## ######## ########## ## #####, ### ##### ###### ##########. ## ########, ####### ### ####, ### ### ######### ### #### ## #### ## ######### ###### ######–####### ########, ## ###### ####### ### ##### ########### ########### ## ### #### ######### ###########.

6.5. ### ######### ######

######, ## #### ## ######### ############## ############ ##–###### ## #### ###### ####### #######. ## ######## ######## ### ############## ######, ### ### ######### #### ######## (### #### ########## ## ######) ####### ### ####### ## ### ########.

6.5.1. ######

########:

- ######### ########## ##########
- ########### ### ############### ##########.
- ###### ############
- ########## ######## ##### ### ###### #######
- ###### ##### ##### ########### #########
- ##### ########## ##########
- ##### ##### ### ####### ## ###### ########
- ##### ##### ### ## ###### #######
- ##### ###### ############### (###### #### #####, ###### #### #####/##### ###–#######, ###### ####### ######).
- ############### ################ ###### #####

- mac_syscall() ######### ###### ####
- ####### ######## ######## ########### ## ### ###### #######

6.5.2. ######### ##########

########, ###### ########, ### ###### #####.

#####, ######## ### ###### ######## ## ### #### #### ###### ### #### ##–########, ### ####### ######## #### ## ########### ## ########### ######## ## ####### ###### ##########. ## ########, ## ### ######### ####### ## ######## #### ### ######, ####### ######## #### ## ########## ######## ##### ##########. ## ## ######### ######### #### ############# ########### ######## ### ## #### ## ####### ###### ######## ## ########## ############, ## #### ####### ###### ### ######### ## ### ###### ########. #######, ########## ######## ###### ## #########, ## #### ####### ####### ########### ### ####### ########### ######–##### ### ###### #######.

####### ### ######### ## ###### ####### ## ######### ##### ### ###### ###–### ########## ###### ##### ### ##### ###### ##########, ######### #### ###### #########; ###### ####### #### #### ### ########### ## ######### #### ### ##–#### ######, ######### ########## ######### ######### ## ### ######.

6.5.3. ###### #### ########### ###

########, ### ### ########## ## ##–### ###### ## ##########, ############### ## ######## ## ####### ####### ## ##–####### ## ######## ##### ## ##### ##### ########## ## ## ########, ######## ### ### ## ###### ######## ### ### ########. #### ## ############ ## ##### ## # ######### #### #####: ######## ## ##### ##### ## #######, ### #### ##### ## ##–#########; ######## ## ## ######, ### #### ##### ## ###########. ##### ### #### ##### ## ########, ###### #### ####### ### ### #########, ### ####### ########## ## ###### ### ###### #### #### ##### ##### ### #### ## ### ####. ### #### ##### ## ######### ## # #####, ### # ######### ######## ## #### ## #### ## ######## ####### ## ###### #### #############. ### #### ###### ## #### ################ ##### ## #### ######### #### ### ### ######### #### ###### # ###### ###### ## #########, ######## ### ######### ####.

####### ############ ### #### ## ###### ### ######## ## ### #### #####, ######–### ######## ### #### #### ## ############ ### ############ ## ### #### ## ###–## ## ######## #### ###### ####### (######## #### ### ###### ###### ### ###### ######, ### ###### ## ########). # ############ ###### ## #### ######## ##### ######## ### ###### ## ### ### ## ###### ######## ## ########, ##### ########## ### ##### ####### ##### ########## #### ########## ########### ###### ### ##–###### ######## ## ############### ## ## ###### ########.

######, # ###### ##### #### ###### ## ####; ## # ######, ## ## ######## ### ### #### ## ###### ####### ## ##### ### # ########### ###### ## #### ####### ### ### ######### ## ###### ####.

6.5.4. #####

####, ### ############ ##### ## #### #### ### ###### #### ### ### ######### ## #########, ######## ######### ####### ########## ## ### ### #####–#### ## ######### ############. ## #######, ######## ###### ################ ## ###### ###### #### ## #### ## ####### ### ######### ######## ###### ## ### ##–####: ### #######, ### ###### ## ####### ### ######### ##### ### ######## ####–## #####. ########, ######### ### ########## ###### ### ######### ######### ## ##### ## ### ######### #######: ### ###########, ############# ######### ### ########## ### ##### ######## ## #### ### ######### ## ### ########## #####–####. ### ### ######### ####### ######### ##### ## ####### #### ####### #### ## ###### #########. ###### ####### #### ## ##### ## ##### ################ #####–####, ## #### #### ######### ##### ### ##### ## ######## ######### ## ######: ### #######, #### # ######### ######### ## # ########## ## ###### ## # ###### ### ## ##### #####, #### #### ########## ### ######### ## ### ##### ##### ######## ## ### ##########.

6.5.5. ###### ################ ###

#######–####### ##### ### #### ###### ##### ###### ### ## ### ######## ### ########## ###### ## ### ####### ######. ## ### ###### ###### ##### ### ## ####### #####, #### ### ####### ### ### ## ################ ########## ###### ### ###### ## ####### ############ ##### ## #### ##### ######### ## ######### ######### ## ### ######. ######## #### ######### ## #### ## #### ### ## ######## ####### ###–############ ########## ### #### #######, ######### #######, ##### #####, ###–###### #########, ### ######## ##########. #######, ######## ###### ## ####### ## ###### ##### ########## #########, ########## ######## ###### #### ######, ### ########### #### #### ##### ###### ### ### ######### ## #####, ######## ### ### ## ########## ## ##### ##### ###### ## ####### ### ###### ##########.

##########, #### #### ######### #### ## ####### ### ######### ##### ## ##### ## ##### ######### ### ###### #### ##### ## ####### #### #########. #### #### ######## ###### ##### ## #### ##### ## ### ###### #### #####, ####### ## ##### ########.

6.5.6. ######

########: # ###### ####, ### # ####### ####. ### ##### ###### #### #### ####### ## ##### ####### #########:

######## ######### ##### ## ### ######## ## #### ### ## ### ###### ####. #### ###### ####### ########## ### ######### ######## ### ######, ### ###### ###### #### ### SYSINIT ## ###### # ############ ########; #### # ###### ###### ## ##–######, SYSINIT #### ######## ###### # ############### ########. ############ ### #### ## # ###### ###### ## ###### #### #### ####, ## ############ ######### ### ######### ### ### ############# (### #######, ### ###### ##### ####### #####–### ### ############ ######## ##### ##### ## #########), ## ##### ###### ###–########## ##### ### ## ### (#### ######## ### #### ## ###### ##### ## ####). ########, ############### ### #### ## # ###### ## ####### ## ### ##########.

6.5.7. #####

####: #### ###### # ##–#### ## #### ## ###### ### ######### ## ######## ######, ### #### ####### # ###–############ ##### ######### ####### ## ################# #######. ### ##### ####### ## ########## ## ### ### ######### ### ##### ########## ##### ######, ### ####### ### ######### ## ##### # ######## ####### ## ###### ####### ####### ########## ############ ####### ## ### ###### ######### ########### ### ##–####. ### #######, ##### ######## #### #### ##### ## #########, ####### ######–#####, #######, #####, ######, #####, ####### ##########, ## ########## ######, ### # ####### ## ##### ################# ##########. ###### ######## #### ##–#### ### ### ######### #### #### ####### ######### ######## #########, ######–#### ###### ####### ## ####### ##### ######### ##### ## ##### ### ######## (###–##### ######### #### ###### ## ######## ######). #### ## ##### ######## #####–### ######### #### ## ######## ###### ####### ######; #######, #### ###### #### ####### ######## ######### #### ## ###### ######## ### ####### ### ##### ####–###### ## ####### #########.

6.5.8. ######

####, ### ##–##### ## ### ###### ####### #### ## ######## ## ### ######### ##### # #######–#### ########. #### ######## ## ######### ##########, ### ######## #### ### ##–#### ######## #### ####### # ####### ### ## ## ###### #######. ## ######## ### ###### # ####### ## ##### ########## (#######, ###### ######, ###### #### ### #####, ...), # ########## ######## ####### ### ######### ##### #### ### ### ## ##–#### ######## ## ########. ## #######, ###### ########## #### ## ###### #### ### ##### #### ## ######### ## ###### ########## #### ###### ## ## ###### ## ######. ##### ## ## ### ########## #### ### ######### ########### #### ## ###### ## ##–####, ## #### ##### #### ## ## ### #### ###### ########## ## ########. ### ####–###, ########### ####### ####### ##### #### #### ### ## #### ######## ##### # ####### ###########.

6.5.9. ########

##–
#####, ### ### ######### ######## # ### ## ############### ##### ##########
#######. ###### #####
####### ######### ###########, ########### ######, ######### ######, #######–
#####, #######, #####, ### #####. ## ###### ########, ## #### #### ###### #######
##–
####. ###### ####### ########### ## ### ############### ### ############### ##
############ ###### ######## ## #### ############, ### ### ###### ######## ##–
#######.

######### ###### ### ###### ## ############### struct label, ##### ######## ## #
############ ##### ## ######, #### ####### # void * ####### ### # long. ########
########### ### ##### ####### #### ## ######## # ###### ##########, ##### ###
#######. ### ######### ## ### ####### ###
######## ## ## ### ###### ######: ####### ### ######## #### # ####### ## #####
#####, ######### ##############,
###########/#########, ### ############. ##### ##### ###########, ## ## ######## ##
######### ######### ######## ### ##### ####### ######. ###### ###### ## ### ##–
#####,
######
####### ## ### ######### ##### #### ##### ######. ### ####### ######### ## ####
##########, ##### #### ## ######## ############ ##
#####. #### ### ###### ## ###### ######### ## ### ### #########.

############## ##### ###### ########## ####### # ######## ########### #### ##–
######## ####### ## ### ## ############## ## ######### ## #####; ## ########
#########, # ####### ### ## ######## ## ###### ########## ## ###
(### ##### ######). #### ### #####, ### #######, ## ### ####### ##### ###### ##–
####### ########, ##### ######## ## ### #########, ## ##### ### ###### ##### #
#####. ### ## ### ########### #### ## ########### ###### ## ######### #######
####### (#####), ######## #### ############ ####### # ########### #### #### ##–
#########. ########### ###### ######## ###### ### ## ###### #### ##
######,
####### ### ####### ##### #### ### ###### ## ######. ### ### ######### ######–
0 ## #### #####, #####
######. #######, ## ########## ## #### ######
###########, ######## #### #### ## #### ## ###### # #### ##### ####### ### #####
###########.

######, ####### ####### ## ######## ### ###
####### ## ######## ###### ## ######## ##########. ##### #########, ########
######### ############ ### #### ## ###### ########## ######## ####### ## ####–
###2

######. ###### ####### ### ###### ## ######### ########## #### ###### ###### ###### ##### ### (## ####) ######## ##########. ### ########## #######, ### ##–### ##### (v_label) ## # ##### ## ### ####### #####; ######## ### #### ## #### ###### #### ### ##### #### ### ##### ## ############, ### ###### ### ##### ## ######. ## # ######, ### ######## ######### #### ### ## ######## ######## #### ##### ###### ####### #####.

> **####**
>
> #########, ## # ####### ###### ####### ####### #########, ### ##### #### ###### ## #########, ##### ###### # ###### (### ########## ###) ##### ## ### ###### ## ############# ########## ### ####### ########.

6.5.10. ######

#####: #### ## ##### ##### ###–#### ### ############### ##### ######### ### ############ #### ####### ## #### ############.

#########, struct mac, ##### ######## # ###### ## ### ##### ########. #### ####### ######## # ######### ###### ####, ### ######### ###### #####. #### ###### #### ## ##### ### ###### ## ##### # ########## ####### ####, ########## ######## ## ###### ######## ####–####### ######## ## #######. ###### ####### ####### ### ############### ### ##–############# ####### ###### ###### ### ############# ###### ### ##### ######, ########## # ####### ## #########. ##### ########## ###### ##### ### ######### ####### ## #### ####### ######### ## ####### ###### ########## ### ##### ###–#####, ########### #### ############ #### #### ###### ######.

######:

- mac_get_proc() ### ## #### ## ######## ### ##### ## ### ####### #######.
- mac_set_proc() ### ## #### ## ####### # ###### ## ### ##### ## ### ####### #######.
- mac_get_fd() ### ## #### ## ######## ### ##### ## ## ###### (####, ######, ####, ...) ########## ## # #### ##########.
- mac_get_file() ### ## #### ## ######## ### ##### ## ## ###### ########## ## # #### ###### ####.
- mac_set_fd() ### ## #### ## ####### # ###### ## ### ##### ## ## ###### (####, ######, ####, ...) ########## ## # #### ##########.

- mac_set_file() ### ## #### ## ####### # ###### ## ### ##### ## ## ###### ########## ## # #### ###### ####.
- mac_syscall() ####### ###### ####### ## ###### ### ###### ##### ####### ######### ### ###### #### #####; ## ####### # ###### ###### ####, ######### ######, ### ###### ######## ### ### ## ### ######.
- mac_get_pid() ### ## #### ## ####### ### ##### ## ####### ####### ## ####### ##.
- mac_get_link() ## ######### ## mac_get_file(), #### ## #### ### ###### # ######## #### ## ## ## ### ##### ##### ## ### ####, ## ### ## #### ## ######## ### ##### ## # #######.
- mac_set_link() ## ######### ## mac_set_file(), #### ## #### ### ###### # ######## #### ## ## ## ### ##### ##### ## # ####, ## ### ## #### ## ########## ### ##### ## # #######.
- mac_execve() ## ######### ## ### execve() ###### ####, #### ## #### ####### # ##–####### ##### ## ### ### ####### ##### ## #### ######### ######### ## # ### #######. #### ###### ## ##### ## ######### ## ######## ## ## # ############.
- mac_get_peer(), ######## ########### ### # ###### ######, ######### ### ##### ## # ###### #### ## # ######, ## #########.

#####, ### SIOCSIGMAC ### SIOCSIFMAC ####### #####–#### ###### ###### ### ###### ## ####### ########## ## ## ######### ### ###.

6.6. ### ######

######## ######## ### ###### ###### ######## #### ### ######, ## ######## #### ######## ###### ####### #### ### ## ###### ## ####, ## ########### ##### ### ###### ####### ###### ##### ## #######. ###### ####### ######## #### ### ###### ####### # ### ## ######## ##### ######, ######### ###### ## # ###### ## ###### ###### ### ########## ### ###### ## ######### ###### ####### #########. #### ###### ######## # ###### ## ########:

- ######## ############# ########## ### ######.
- ########### ############## ## ### ###### ##### ### ##########.
- ######## ############## ## ###### #### ##### ######, #### ## ############## ### ###########.
- ######## ####### ### ############, ###########, ### ########## ###### ## ##–###### ###### #######.

- ######## ####### ### #### ####### ########## ### ############ ## ###### ## ######## #######.
- ############## ## ######## ###### ####### ##### ###### #### ### ## ######## ## ### ######.
- ########### ## ###### ########, ###### ##### ######, ### ###### ##########.

6.6.1. ######

####### ### ## ######## ##### ### MAC_POLICY_SET() #####, ##### ##### ### ######, ######## # ######### ## ### ### ##### ##### ######, ######## ######### ##### ########### ### ### ###### ######### ###### ###### ### ######, ### ########## ######## ### ########## ## ##### ##### ## ### #########.

```
static struct mac_policy_ops mac_policy_ops =
{
	-.mpo_destroy = mac_policy_destroy,
	-.mpo_init = mac_policy_init,
	-.mpo_init_bpfdesc_label = mac_policy_init_bpfdesc_label,
	-.mpo_init_cred_label = mac_policy_init_label,
/* -... */
	-.mpo_check_vnode_setutimes = mac_policy_check_vnode_setutimes,
	-.mpo_check_vnode_stat = mac_policy_check_vnode_stat,
	-.mpo_check_vnode_write = mac_policy_check_vnode_write,
};
```

######, mac_*policy*_ops ## #### #######, ########## ####–##### ####### ## ### ###### #### ######## ##### ######. # ######## ####### ## ######### ##### ###### ### ##### ########## ### ## ##### ## ### ### ##### ##### ######### #######. ## ######## ######## ###### ###### ############ ### ### .###_####### ### .###_#### ##### ######. .###_#### #### ## ####### #### # ###### ## ############ ########## #### ### ####### ######### ### ##### ## ### ##### ##### ###### ######## ######. #### ####### ### ###### ## ####### ### ###–############ ########## ### ##############, #### ## ############## ## ### #### ## #####. .###_####### #### ## ####### #### # ###### ###### ## ######## ## ###–### ######### ## ### ######### ###### ### ########### ## #####. #########, ##### ### ##### ###### ### ####### #### ### ### ###### #### ##### #### ## ####### ### ##### ##### ###### #### ##### #######: #### #### ## #######, ### ## ### #### ####, ######## ###### ## ####### ##### #### ###### ########## #### ###### ## ## ## ##### #### ######## ## ######## ########.

############. ## ########### ###### ###### ## ########. ### #### ###### #### ## ### ###### ## ######### ## ### #### ### ### ###### ### ###### #### ### ###### ######, ### #### ######## #### ######### ###### ########### ## ######## #########.

6.6.2. ######

######. ###–######, ##### ##### ### #######:

###_########_####_########
: #### #### ######### #### ### ###### ###### ### ## ########. ## #### #### ## ### ########, #### ### ###### ######### #### ###### ######## ## ###### ### ######. #### #### ##### ## #### ## ####### #### ######## ##### ##### ### ### ###### ## #### #### ##### ## #######.

###_########_####_#######
: #### #### ######### #### ### ###### ###### #### ## ###### ### ########### ##### ## ### #### #######. ## ### #### ## #########, ######## ## ######## ### ###### ######### #### #### ## ########. ### #### ### ## #### ## ######## #### ####### ######### ######## ## ### ###### #######, ### ###### ###### ####### #### #### ### #### ######## ########### ## ### ######.

###_########_####_##########
: #### #### ######### #### ### ###### ###### ######## ######## ## #####, ### #### ###### ###### ###### ## ######### ### ### ####### ## #### ######. ## #######, ### ### ######### #### ### ######## ##### ####### ### ##### ###### ## ##### ### ###### ###### ### #### #### ###. #### ########## ######## ###–#### ########### #### ######## ## ### ####### #### ########. # ###### ##–####, MAC_ALWAYS_LABEL_MBUF, ###### ## ##### ### ### ######### ## ####–#### #### ##### ####### ########## ## ### ####### ## #### ####, ### ### ## ###### ## #### ############.

> ### ####
>
> ######## ##### ### MPC_LOADTIME_FLAG_LABELMBUFS ####### ### MPC_LOADTIME_FLAG_NOTLATE #### ### #### ## #### ## ###–###### ###### NULL #### ##### ######## ###### #### ##### ######. #### ## ######### ## ######### ##### ####### ##### ####### ### ####### ##### # ###### ######## #### ######## ### #### ######. ## # ###### ## ###### ###### ### ####### ###–###### ## ###### (#.#., ### ###### ## ### ##### ###### ####), #### ### ##### ### ########## ## #### ##### #######.

6.6.3. ###### #####

#####–####: ##### ###### ########## #### ### ############ ### ########## ## ########,

##############, ########, ###########, ### ##### #### ##– ### ###### ### ###### #######, ###### ########## #### ###### ####### ######### #### ### ###### ###### ### #########, ### ##### ########## #### ### ########## ## ###### ## #######. ## ########, # mac_syscall() ##### ##### ## ######## ## #### ######## ### ###### ### ###### ######### ####### ########### ### ###### #####.

########, ## #### ## #### ###### ##### ### ######### ###### ##### ##### ######. ####### ###### ####### ## ##### ######## ######### ## ######## ######## ######## ##### ###### ## ##### ######, ### #### ###### ### ####### ######## ### ###### ###### ### ###– #########. ## ##########, ####### ###### ## ##### #### ##### ######### ##### ## ###### ####### ### ##### ###### ### ######### ####, ########## ##### ## ###### ## ###### ## ### ##### ### ### ## ####### ### ### ##### ######. ####### #######– #### ### ######### ## ########## ## ### ### ######### ##### ##### ########.

######. #### ####### ####### ######## ## ## ####### ## ### ######### ## ### ###### ### ##### #### ######### ##### ## ### #####. ### ######### ## #### ## ### ####### ##########, ##### ## ####### ## ## ########## ## ######## ## # ##### ##### ######## ###### ## ### ######.

6.7. ### ###### ##### #####

6.7.1. ############### ###### #####

6.7.1.1. **mpo_init**

void **mpo_init**(*conf*);

struct mac_policy_conf **conf*;

####–
#####–
####
###

conf ###
###–
#–
##
##–
#–
#–
#–
#–
####

#####. ### ###### #### ##### ## ####, ## ##### ########## ###### ## #########, ### ##### ### ## ##### ###### ########## #### ## #### #### #######. ## ########### ######## ###### ########### ### ######## ###### ###### #######–#######, #### ###### ## #### ##### # ######## ###### #######().

6.7.1.2. **mpo_destroy**

void **mpo_destroy**(*conf*);

struct mac_policy_conf **conf*;

####–
#####–
####
###

conf
###–
#–
##
##–
#–
#–
#–
#–
####

#####. ### ###### #### ##### ## ####, ## ####### ###### ## #######.

6.7.1.3. **mpo_syscall**

int **mpo_syscall**(*td, call, arg*);

struct thread **td*;
int *call*;
void **arg*;

####–
#####–
####
###

td ####–
###
######

####– #####–	####	###
call ###– #– ######– ###–	## #######	###– ###
arg ###– ##	## #######	##– ##– #####

######–## ###### #####. ### ###### #### ######## ###### ############ ## #### ## ##### ##### #### ########, ### ### ######### #### ## ######### ## #### ##### #####. #### ############ ### ########, ######## ####### ###### ## #### ## ###### ##–######### ###### ####### ###### #### ### ### ######### ## ######. ### #######, ## # ###### ########## ## ######### ###### #############, ## ###### #### ### ######### ###### ###### ####### ###### ## ###### ### ### ######### ### ##### ########## ########.

> ####
>
> ####### #### ######### ####### ### copyin() ## ### ####### #### ## ##### ###.

6.7.1.4. mpo_thread_userret

void **mpo_thread_userret**(*td*);

struct thread **td*;

####- #####- #### ###	##- ####- ### ######

#####, ### # ###### #### ######, #### ######, ## #########. #### ## ######## ### ######## #### #### ######## ####### ######, ## ## ## ### ##-#### ######## ## ####### ### ####### #### ## ######### ###### ## ### ##### ###-### ###### #### ###########; ####### ###### ##### ######### ########### ######-######## ####, ####### ####### ###########, ## ##### ####. ## ###### #### #####-####, ######## ####### ## ### ####### ########## ##### ### ## ###### ## ### p_label ######### ## # ########## #### ####, ### #### ### ### ########## TDF_ASTPENDING #### ### ########### PS_MACPENDM #### ## ######## # #### ## ### ####### ##### #####. #### #### ##### #####, ### ###### ### ###### # ########### ########## #### #### ####### ##### ### ####### #######. ###### ####### ### ######### #### ##### ######## ######## ## ########## ## ### ### ### ### ##### ######### ### ## ###-#### ### ########## ## ############# ############.

6.7.2. #####

6.7.2.1. mpo_init_bpfdesc_label

void **mpo_init_bpfdesc_label**(*label*);

struct label **label*;

####- #####- #### ###	### ##- ### ## ##- ###

########## ### ##### ## # ##### ############ ####### (### ##########). ######## ## #########.

6.7.2.2. **mpo_init_cred_label**

void **mpo_init_cred_label**(*label*);

struct label **label*;

####– #####– #### ###	la## bel– ### ## ###– ####– ###

########## ### ##### ### # ##### ############ #### ##########. ######## ## ###–######.

6.7.2.3. **mpo_init_devfsdirent_label**

void **mpo_init_devfsdirent_label**(*label*);

struct label **label*;

####– #####– #### ###	la## bel– ### ## ##– ###

########## ### ##### ## # ##### ############ ##### #####. ######## ## #########.

6.7.2.4. **mpo_init_ifnet_label**

void **mpo_init_ifnet_label**(*label*);

struct label **label*;

####–
#####–
####
###

label ###
##–
###
##
##–
###

########## ### ##### ## # ##### ############ ####### #########. ######## ## #########.

6.7.2.5. mpo_init_ipq_label

void **mpo_init_ipq_label**(*label*, *flag*);

struct label **label*;
int *flag*;

####–
#####–
####
###

label ###
##–
###
##
##–
###

flag #####–
###/####
#####–
###
###–
###(9);
###
##–
###

########## ### ##### ## # ##### ############ ## ######## ########## #####. ### *flag* ##### ### ## ### ## #_###### ### #_######, ### ###### ## ######## ## ##### ########## # ######## ######(9) ###### #### ############## ####. ## ######## ########## ##### ########## ########## ###### ## ########### ######### ####–########, ### ### ############## ###### ## ####### ## ##### ######## ## ##### ##### ##########. #### ##### ##### ## ######### ## #### ######### ## ### ####### ## ######## ### ## ######## ########## #####.

6.7.2.6. mpo_init_mbuf_label

void **mpo_init_mbuf_label**(*flag*, *label*);

int *flag*;
struct label **label*;

####–#####–####	###	
flag	###–###/####–#####–###	###–###(9); ### ##–###
label	##–####	## ##–### ## ###–####–###

########## ### ##### ## # ##### ############ #### ###### ###### (*mbuf*). ### *flag* ##### ### ## ### ## #_###### ### #_######, ### ###### ## ######## ## ##### ###–####### # ######## ######(9) ###### #### ############## ####. #### ########## ########## ###### ## ########### ######### ############, ### ### ##########–#### ###### ## ####### ## ##### ######## ## ########## ##########. #### ##### ##### ## ######### ## #### ######### ## ### ####### ## ######## ### #### ######.

6.7.2.7. mpo_init_mount_label

void **mpo_init_mount_label**(*mntlabel*, *fslabel*);

struct label **mntlabel*;
struct label **fslabel*;

####– #####– #### ###	
mnt– *la*– *bel*	###– ##– ### ## ## ###– ####– #### ### ### ##### ##– ####
fs– *la*– *bel*	###– ##– ### ## ## ###– ####– #### ### ### #### ###– ###

########## ### ###### ## # ##### ############ ##### #####. ######## ## ######–###.

6.7.2.8. **mpo_init_mount_fs_label**

void **mpo_init_mount_fs_label**(*label*);

struct label **label*;

####– #####– #### ###	
la##– *bel*## ## ## ###– ####– ####	

########## ### ##### ## # ##### ####### #### ######. ######## ## #########

6.7.2.9. **mpo_init_pipe_label**

void **mpo_init_pipe_label**(*label*);

struct label**label*;

####– #####– #### ###	
la##– *bel*## ## ## ###### ##	

########## # ##### ### # ##### ############ ####. ######## ## #########.

6.7.2.10. **mpo_init_socket_label**

void **mpo_init_socket_label**(*label*, *flag*);

struct label **label*;
int *flag*;

####–
#####–
####
###

label ###
###–
###
##
###–
####–
###

flag ###–
###(9)
#####

########## # ##### ### # ##### ############ ######. ### *flag* ##### ### ## ### ## #_###### ### #_######, ### ###### ## ######## ## ##### ########## # ######## ######(9) ###### #### ############## ####.

6.7.2.11. **mpo_init_socket_peer_label**

void **mpo_init_socket_peer_label**(*label*, *flag*);

struct label **label*;
int *flag*;

####–
#####–
####
###

label ###
###–
###
##
###–
####–
###

flag ###–
###(9)
#####

########## ### #### ##### ### # ##### ############ ######. ### *flag* ##### ### ## ### ## #_###### ### #_######, ### ###### ## ######## ## ##### ########## # #####–### ######(9) ###### #### ############## ####.

6.7.2.12. mpo_init_proc_label

void **mpo_init_proc_label**(*label*);

struct label **label*;

####–
#####–
####
###

ta##
bel–
###
##
###–
####–
###

########## ### ##### ### # ##### ############ #######. ######## ## #########.

6.7.2.13. mpo_init_vnode_label

void **mpo_init_vnode_label**(*label*);

struct label **label*;

####–
#####–
####
###

ta##
bel–
###
##
###–
####–
###

########## ### ##### ## # ##### ############ #####. ######## ## #########.

6.7.2.14. mpo_destroy_bpfdesc_label

void **mpo_destroy_bpfdesc_label**(*label*);

struct label **label*;

####–
#####–
####
###

####–
##
##–
###

####### ### ##### ## # ### ##########. ## #### ##### ##### # ###### ###### #### ### ######## ####### ########## #### *label* ## #### ## ### ## #########.

6.7.2.15. mpo_destroy_cred_label

void **mpo_destroy_cred_label**(*label*);

struct label **label*;

####–
#####–
####
###

##–
###
##–
###
##–
#######

####### ### ##### ## # ##########. ## #### ##### #####, # ###### ###### ###### #### ### ######## ####### ########## #### *label* ## #### ## ### ## #########.

6.7.2.16. mpo_destroy_devfsdirent_label

void **mpo_destroy_devfsdirent_label**(*label*);

struct label **label*;

####–
#####–
####
###

##–
###
##–
###

####–
#####–
####
###
##–
#######

####### ### ##### ## # ##### #####. ## #### ##### #####, # ###### ###### ###### #### ### ######## ####### ########## #### *label* ## #### ## ### ## #########.

6.7.2.17. mpo_destroy_ifnet_label

void **mpo_destroy_ifnet_label**(*label*);

struct label **label*;

####–
#####–
####
###
la##–
bel###
##–
###
##–
#######

####### ### ##### ## # ####### #########. ## #### ##### #####, # ###### ###### ###### #### ### ######## ####### ########## #### *label* ## #### ## ### ## #########.

6.7.2.18. mpo_destroy_ipq_label

void **mpo_destroy_ipq_label**(*label*);

struct label **label*;

####–
#####–
####
###
la##–
bel###
##–
###
##–
#######

####### ### ##### ## ## ## ######## #####. ## #### ##### #####, # ###### ###### ###### #### ### ######## ####### ########## #### *label* ## #### ## ### ## #########.

6.7.2.19. mpo_destroy_mbuf_label

void **mpo_destroy_mbuf_label**(*label*);

struct label **label*;

####–
#####–
####
###

la#–
bel#
##–
###
##–
#######

####### ### ##### ## ## #### ######. ## #### ##### #####, # ###### ###### ###### #### ### ######## ####### ########## #### *label* ## #### ## ### ## #########.

6.7.2.20. mpo_destroy_mount_label

void **mpo_destroy_mount_label**(*label*);

struct label **label*;

####–
#####–
####
###

la####
bel####
##–
###
##–
###
##–
#######

####### ### ###### ## # ##### #####. ## #### ##### #####, # ###### ###### ###### #### ### ######## ####### ########## #### *mntlabel* ## #### #### ### ## #########.

6.7.2.21. mpo_destroy_mount_label

void **mpo_destroy_mount_label**(*mntlabel*, *fslabel*);

struct label *mntlabel*;
struct label *fslabel*;

####– #####– #### ###	*mnt*##### *la*##### *bel*– ### ##– ### ##– #######	*fs*#### *la*##– *bel*## ##– ### ##– ### ##– #######>

####### ### ###### ## # ##### #####. ## #### ##### #####, # ###### ###### ###### #### ### ######## ####### ########## #### *mntlabel* ### *fslabel* ## #### #### ### ## #########.

6.7.2.22. mpo_destroy_socket_label

void **mpo_destroy_socket_label**(*label*);

struct label *label*;

####– #####– #### ###	*la*####– *bel* ##– ### ##– ###

####–
#####–
####
###
##–
#######

####### ### ##### ## # ######. ## #### ##### #####, # ###### ###### ###### #### ### ######## ####### ########## #### *label* ## #### ## ### ## #########.

6.7.2.23. **mpo_destroy_socket_peer_label**

void **mpo_destroy_socket_peer_label**(*peerlabel*);

struct label **peerlabel*;

####–
#####–
####
###
peer##–
la#
bel###
##–
###
##–
###
##–
#######

####### ### #### ##### ## # ######. ## #### ##### #####, # ###### ###### ###### #### ### ######## ####### ########## #### *label* ## #### ## ### ## #########.

6.7.2.24. **mpo_destroy_pipe_label**

void **mpo_destroy_pipe_label**(*label*);

struct label **label*;

####–
#####–
####
###
la###
bel#–
###

####### ### ##### ## # ####. ## #### ##### #####, # ###### ###### ###### #### ### ######## ####### ########## #### *label* ## #### ## ### ## #########.

6.7.2.25. mpo_destroy_proc_label

void **mpo_destroy_proc_label**(*label*);

struct label **label*;

####–
#####–
####
###
la######
##–
###

####### ### ##### ## # #######. ## #### ##### #####, # ###### ###### ###### #### ### ######## ####### ########## #### *label* ## #### ## ### ## #########.

6.7.2.26. mpo_destroy_vnode_label

void **mpo_destroy_vnode_label**(*label*);

struct label **label*;

####–
#####–
####
###
la######
##–
###

####### ### ##### ## # #####. ## #### ##### #####, # ###### ###### ###### #### ### ######## ####### ########## #### *label* ## #### ## ### ## #########.

6.7.2.27. mpo_copy_mbuf_label

void **mpo_copy_mbuf_label**(*src*, *dest*);

struct label **src*;
struct label **dest*;

####–
#####–
####
###

src ######
##–
###

dest ###–
##–
##–
####
##–
###

src #### *dest*.

6.7.2.28. mpo_copy_pipe_label

void **mpo_copy_pipe_label**(*src*, *dest*);

struct label **src*;
struct label **dest*;

####–
#####–
####
###

src ######
##–
###

dest ###–
##–
##–
####
##–
###

src #### *dest*.

6.7.2.29. mpo_copy_vnode_label

void **mpo_copy_vnode_label**(*src*, *dest*);

struct label **src*;
struct label **dest*;

####–
#####–
####
###

src######
##–
###

dest###–
##–
##–
####
##–
###

src #### *dest*.

6.7.2.30. **mpo_externalize_cred_label**

int **mpo_externalize_cred_label**(*label*, *element_name*, *sb*, *claimed*);

struct label **label*;
char **element_name*;
struct sbuf **sb*;
int **claimed*;

####–
#####–
####
###

la##–
sb###
##
##
##–
###–
###–
####

element_name####
##
###
###–
#–
##
#####

####–
#####–
####
###
##–
###
######
##
##–
###–
###–
####
sb######
######
##
##
######
####
#
####
###–
##–
###–
##–
####
##
##–
###
claimed#####
##
##–
###–
####–
##
####
element_data
###
##
######
##.

####### ## ############ ##### ##### ## ### ##### ######### ######. ## ########–
##

####. #########, ### ######### externalize ##### ###### #### ## ######, ## ### ############## ###### ##### ### ###–##### ## *element_name* ###### ########## ## #### ## *sb*. ## *element_name* #### ### ##### ### #### ## #### ######, ###### ###### 0. #### ###### ####### ## ## ##### ###### ##### ############# ### ##### ####. #### ### ###### ##### ## *element_data*, *claimed ###### ## ###########.

6.7.2.31. mpo_externalize_ifnet_label

int **mpo_externalize_ifnet_label**(*label*, *element_name*, *sb*, **claimed*);

struct label **label*;
char **element_name*;
struct sbuf **sb*;
int **claimed*;

####–
#####–
####
###

label ##–
###
##
##
##–
###–
###–
####

element_name ####
##
###
###–
#–
##
#####
##–
###
######
##
##–
###–
###–
####

####–
#####–
####
###
sb ######
######
##
##
######
####
#
####
###–
##–
###–
##–
####
##
##–
###
claimed ######
##
##–
###–
####–
##
####
element_data
###
##
######
##.

####### ## ############ ##### ##### ## ### ##### ######### ######. ## ########–#### ##### ######## ## # #### ############## ## ### ##### ######## #### ### ## #### #### ######## ############ ### #### ## ### ####. #########, ### ######### externalize ##### ###### #### ## ######, ## ### ############## ###### ##### ### ###–##### ## *element_name* ###### ########## ## #### ## *sb*. ## *element_name* #### ### ##### ### #### ## #### ######, ###### ###### 0. #### ###### ####### ## ## ##### ###### ##### ############# ### ##### ####. #### ### ###### ##### ## *element_data*, *claimed ###### ## ###########.

6.7.2.32. mpo_externalize_pipe_label

int **mpo_externalize_pipe_label**(*label*, *element_name*, *sb*, **claimed*);

struct label **label*;
char **element_name*;
struct sbuf **sb*;
int **claimed*;

####–
#####–
####
###

label
##–
###
##
##
##–
###–
###–
####

element_name
####
##
###
###–
#–
##
#####
##–
###
######
##
##–
###–
###–
####

sb
######
######
##
##
######
####
#

####–
#####–
####
###
####
###–
##–
###–
##–
####
##
##–
###
claimed #####
##
##–
###–
####–
##
####
element_data
###
##
######
##.

####### ## ############ ##### ##### ## ### ##### ######### ######. ## ########–#### ##### ######## ## # #### ############## ## ### ##### ######## #### ### ## #### #### ######## ############ ### #### ## ### ####. #########, ### ######### externalize ##### ###### #### ## ######, ## ### ############## ###### ##### ### ###–##### ## *element_name* ###### ########## ## #### ## *sb*. ## *element_name* #### ### ##### ### #### ## #### ######, ###### ###### 0. #### ###### ####### ## ## ##### ###### ##### ############# ### ##### ####. #### ### ###### ##### ## *element_data*, *claimed ###### ## ###########.

6.7.2.33. **mpo_externalize_socket_label**

int **mpo_externalize_socket_label**(*label, element_name, sb, *claimed*);

struct label **label*;
char **element_name*;
struct sbuf **sb*;
int **claimed*;

####–
#####–
####
###

##–
###
##
##
##–
###–
###–
####

*element_name*
##
###
###–
#–
##
#####
##–
###
######
##
##–
###–
###–
####

sb######
######
##
##
######
####
#
####
###–
##–
###–
##–
####
##
##–
###

####–
#####–
####
###

claimed #
##
##–
###–
####–
##
####
element_data
###
##
######
##.

####### ## ############ ##### ##### ## ### ##### ######### ######. ## ########–#### ##### ######## ## # #### ############## ## ### ##### ######## #### ### ## #### #### ######## ############ ### #### ## ### ####. #########, ### ######### externalize ##### ###### #### ## ######, ## ### ############## ###### ##### ### ###–##### ## *element_name* ###### ########## ## #### ## *sb*. ## *element_name* #### ### ##### ### #### ## #### ######, ###### ###### 0. #### ###### ####### ## ## ##### ###### ##### ############# ### ##### ####. #### ### ###### ##### ## *element_data*, *claimed ###### ## ###########.

6.7.2.34. mpo_externalize_socket_peer_label

int **mpo_externalize_socket_peer_label**(*label*, *element_name*, *sb*, *claimed*);

struct label *label*;
char *element_name*;
struct sbuf *sb*;
int *claimed*;

####–
#####–
####
###

label #–
##
##
##
##–
###–

####–
#####–
####
###
###–
####
####
element_name
##
###
###–
#–
##
#####
##–
###
######
##
##–
###–
###–
####
sb######
######
##
##
######
####
#
####
###–
##–
###–
##–
####
##
##–
###
######
claimed
##
##–
###–
####–
##

####– #####– #### ###	#### *element_data* ### ## ###### ##.

####### ## ############ ##### ##### ## ### ##### ######### ######. ## ########–#### ##### ######## ## # #### ############## ## ### ##### ######## #### ### ## #### #### ######## ############ ### #### ## ### ####. #########, ### ######### externalize ##### ###### #### ## ######, ## ### ############## ###### ##### ### ###–##### ## *element_name* ###### ########## ## #### ## *sb*. ## *element_name* #### ### ##### ### #### ## #### ######, ###### ###### 0. #### ###### ####### ## ## ##### ###### ##### ############# ### ##### ####. #### ### ###### ##### ## *element_data*, *claimed ###### ## ###########.

6.7.2.35. mpo_externalize_vnode_label

int **mpo_externalize_vnode_label**(*label, element_name, sb, *claimed*);

struct label **label*;
char **element_name*;
struct sbuf **sb*;
int **claimed*;

####– #####– #### ###	*label* ##– ### ## ## ##– ###– ###– ####
element_name ####	## ### ###–

####–
#####–
####
###
#–
##
#####
##–
###
######
##
##–
###–
###–
####
sb######
######
##
##
######
####
#
####
###–
##–
###–
##–
####
##
##–
###
claimed######
##
##–
###–
####–
##
####
element_data
###
##
######
##.

####### ## ############ ##### ##### ## ### ##### ######### ######. ## ########–#### ##### ######## ## # #### ############## ## ### ##### ######## #### ### ## #### #### ######## ############ ### #### ## ### ####. #########, ### ######### externalize ##### ###### #### ## ######, ## ### ############## ###### ##### ### ###–##### ## *element_name* ###### ########## ## #### ## *sb*. ## *element_name* #### ### ##### ### #### ## #### ######, ###### ###### 0. #### ###### ####### ## ## ##### ###### ##### ############# ### ##### ####. #### ### ###### ##### ## *element_data*, *claimed ###### ## ###########.

6.7.2.36. mpo_internalize_cred_label

int **mpo_internalize_cred_label**(*label*, *element_name*, *element_data*, *claimed*);

struct label **label*;
char **element_name*;
char **element_data*;
int **claimed*;

####–
#####–
####
###

label
##–
###
##
##
######
##

element_name
#####
##
###
###–
#–
##
#####
##–
###
######
##
##–
###–
###–
####

####– #####– #### ###	
element_data	##– ## ## ## ##– ###– ###– ####
claimed	## ##– ###– ####– ## #### ##– ## ### ## ###– ####– ###– ## ##– ###– ###– ####.

####### ## ######## ##### ######### ##### ## ############ ##### #### ## #### ######. #########, ### ######### internalize ##### ###### ### ###### #### #####–########## ## #########, ## ### ############## ###### ####### ### ######## ## *element_name* ## ### ### #### ## ##### ## ## #### ## ###### ## ############# ### #### ## *element_data*. #### ## ## ### externalize ##### ######, ### ##### ##### ###### ###### 0 ## *element_name* #### ### ##### ### ### ####, ## #### #### ### ############ ## ##–##########, ## ##### #### *claimed ###### ## ###########.

6.7.2.37. mpo_internalize_ifnet_label

int **mpo_internalize_ifnet_label**(*label*, *element_name*, *element_data*, *claimed*);

struct label **label*;
char **element_name*;
char **element_data*;
int **claimed*;

####–
#####–
####
###

label#–
##
##
##
######
##

element_name
##
###
###–
#–
##
#####
##–
###
######
##
##–
###–
###–
####

element_data
##–
##
##
##
##–
###–
###–
####

claimed#
##
##–
###–

####–
#####–
####
###
####–
##
####
##–
##
###
##
###–
####–
###–
##
##–
###–
###–
####.

####### ## ######## ##### ######### ##### ## ############ ##### #### ## #### ######. #########, ### ######### internalize ##### ###### ### ###### #### #####–########## ## #########, ## ### ############## ###### ####### ### ######## ## *element_name* ## ### ### #### ## ##### ## ## #### ## ###### ## ############# ### #### ## *element_data*. #### ## ## ### externalize ##### ######, ### ##### ##### ###### ###### 0 ## *element_name* #### ### ##### ### ### ####, ## #### #### ### ############ ## ##–##########, ## ##### #### *claimed ###### ## ###########.

6.7.2.38. mpo_internalize_pipe_label

int **mpo_internalize_pipe_label**(*label*, *element_name*, *element_data*, *claimed*);

struct label **label*;
char **element_name*;
char **element_data*;
int **claimed*;

####–
#####–
####
###
la##–
bel##
##
##

####–
#####–
####
###
######
##
element_name
##
###
###–
#–
##
#####
##–
###
######
##
##–
###–
###–
####
element_data
##–
##
##
##
##–
###–
###–
####
claimed
##
##–
###–
####–
##
####
##–
##
###
##
###–
####–

####–
#####–
####
###
###–
##
##–
###–
###–
####.

####### ## ######## ##### ######### ##### ## ############ ##### #### ## #### ######. #########, ### ######### internalize ##### ###### ### ###### #### #####–########## ## #########, ## ### ############## ###### ####### ### ######## ## *element_name* ## ### ### #### ## ##### ## ## #### ## ###### ## ############# ### #### ## *element_data*. #### ## ## ### externalize ##### ######, ### ##### ##### ###### ###### 0 ## *element_name* #### ### ##### ### ### ####, ## #### #### ### ############ ## ##–##########, ## ##### #### *claimed ###### ## ###########.

6.7.2.39. mpo_internalize_socket_label

int **mpo_internalize_socket_label**(*label*, *element_name*, *element_data*, *claimed*);

struct label **label*;
char **element_name*;
char **element_data*;
int **claimed*;

####–
#####–
####
###
label##–
label###
##
##
######
##
element_name####
##
###
###–
#–
##
#####

####–
#####–
####
###
##–
###
######
##
##–
###–
###–
####
element_data
##–
##
##
##
##–
###–
###–
####
claimed#
##
##–
###–
####–
##
####
##–
##
###
##
###–
####–
###–
##
##–
###–
###–
####.

####### ## ######## ##### ######### ##### ## ############ ##### #### ## #### ######. #########, ### ######### internalize ##### ###### ### ###### #### #####–

########## ## #########, ## ### ############## ###### ####### ### ######## ## *element_name* ## ### ### #### ## ##### ## ## #### ## ###### ## ############# ### #### ## *element_data*. #### ## ## ### externalize ##### ######, ### ##### ##### ###### ###### 0 ## *element_name* #### ### ##### ### ### ####, ## #### #### ### ############ ## ##–##########, ## ##### #### *claimed ###### ## ###########.

6.7.2.40. mpo_internalize_vnode_label

int **mpo_internalize_vnode_label**(*label*, *element_name*, *element_data*, *claimed*);

struct label **label*;
char **element_name*;
char **element_data*;
int **claimed*;

####–
#####–
####
###

label
##–
###
##
##
######
##

element_name
####
##
###
###–
#–
##
#####
##–
###
######
##
##–
###–
###–
####

element_data
####
##–
##
##

####–
#####–
####
###

##
##–
###–
###–
####

claimed##
##
##–
###–
####–
##
####
##–
##
###
##
###–
####–
###–
##
##–
###–
###–
####.

####### ## ######## ##### ######### ##### ## ############ ##### #### ## #### ######. #########, ### ######### internalize ##### ###### ### ###### #### #####–########## ## #########, ## ### ############## ###### ####### ### ######## ## *element_name* ## ### ### #### ## ##### ## ## #### ## ###### ## ############# ### #### ## *element_data*. #### ## ## ### externalize ##### ######, ### ##### ##### ###### ###### 0 ## *element_name* #### ### ##### ### ### ####, ## #### #### ### ############ ## ##–##########, ## ##### #### *claimed ###### ## ###########.

6.7.3. #####

#######. ### #### ####### ###### ###### ## ######## ## # ### ######, ##### ###### ### ## ########## ### ######## #### ##### ######. ### ####### ######### ##############, ########, ### ########### #####. #### ####### #### #### ######### ##########, ######## #### #########

#######. #### ####### #### #### ######### ############### ######, #### ## ##### ###### ########## #### ## ##########. # ####### ####### ###### #### #### ### ######### #### ##### ## ##### ######:

```
Label initialization        o
(object-specific wait)       \
Label creation               o
                        \
Relabel events,               o--<--.
Various object-specific,         -|    -|
Access control events           ~-->--o
                              \
Label destruction                    o
```

###–### ### ###### ####### ####### ### ### ### ## ### ######. ### ##### #### #######–## ## # ###### #### ## ###### ## #######, ## #### ######## ### ### #### ## ####### ##############.

######. ### #######, ##### ### ## ######### ### ###### ###### ## # #### ##### #### ### ########. #### ########## ###### ##### ############## ## ### #### ## #### #####, ### #### ######## ###### #### ### #### ## ########## #### # ####–####. #########, ####### #### ## ######## ### # ######## #####, ######### ### ############# ## ### ########, ### ###### ## ##### ######## ####### ## ### ###–##### #######. ### #######, #### ## #### ## ####### #### # ######, ### ###### ### ### ##### #### ## ######### ## ########### ######## ## ######## ## ### ### #### ### ### #####. ###### ########## ## ######## ###### ## ###########, ## ## ### ##### ## ########### ######### ##### ## ### ######; ## ########, ######## ##### ### ### ######### ## #### ## # ####### ## ######## ###### ###### ## ########.

######, ### #### ######### ####### ## ########### ## ###### ## ###### ### ##### ## ## ###### ##### ## ########## #######. ### #######, ### ##### ## ## ## ######## ########## ##### ### ## ####### ###### ### ###_######_### ##### ##### ## # ###### ## ### ########### ## ## ########## #### ## #### #####.

#######.

########.

#######, ## ########## ##### ## ###### ######: ######### ######. ##### ###### ### #### ## ##### ###### ########### ######### ## #### #########. ##### ###### ### ########### ### ##–####### ## #### ##### ##### #####, ### ### ######## ##### ## ###_###########, ##### ####### # #### ##### ## ## ######### ## ## ######### ##############.

6.7.3.1. #### ###### ###### ######## #####

6.7.3.1.1. mpo_associate_vnode_devfs

void **mpo_associate_vnode_devfs**(*mp*, *fslabel*, *de*, *delabel*, *vp*, *vlabel*);

struct mount **mp*;
struct label **fslabel*;
struct devfs_dirent **de*;
struct label **delabel*;
struct vnode **vp*;
struct label **vlabel*;

####– #####– #### ###	
mp##– ### ##### #####	
fs#– *la*## *bel*### ###– ### ##– ### (mp- >mnt_fslabel)	
de#– ### ##– ###– ##– ## ##– ###	
de##– *la*– *bel*# ##– ###	

####–
#####–
####
###
##–
##–
##–
##–
##
####
de
##–
vp
###
##–
##–
##–
##–
##
####
de
###–
vla
bel
##
##–
###
##–
##–
##–
##–
##
####
vp

(*vlabel*) ### # ##### ####### ##### ##### ##### ## ### ##### ##–####### ##### ###### ## *de* ### ### #####.

6.7.3.1.2. mpo_associate_vnode_extattr

int **mpo_associate_vnode_extattr**(*mp*, *fslabel*, *vp*, *vlabel*);

struct mount **mp*;
struct label **fslabel*;
struct vnode **vp*;
struct label **vlabel*;

####–
#####–
####
###

mp###
###–
###
#####
#####

fs###
la##–
bel##
##–
###

vp#–
###
##
##–
###

vla##–
bel
##
##–
###
##–
##–
##–
##–
##
####
vp

####### ## ######## ### ##### ### *vp* #### ### #### ###### ######## ##########. #### #######, ### ##### 0 ## ########. ###### ######## ######### ######### ### ## #########, ## ######## ######## ## ## #### *fslabel* #### *vlabel*. ## ### ##### ## ## #####, ## ########### ##### ### errno ###### ## ########.

6.7.3.1.3. mpo_associate_vnode_singlelabel

void **mpo_associate_vnode_singlelabel**(*mp*, *fslabel*, *vp*, *vlabel*);

struct mount **mp*;
struct label **fslabel*;
struct vnode **vp*;

struct label **vlabel*;

	####– #####– #### ###
mp	### ###– ### ##### #####
fslabel	#### ##– # ##– ###
vp	##– ### ## ##– ###
vlabel	###– ## ##– ### ##– ##– ##– ##– ## #### *vp*

#######, #### ##### ##### ## ###### ## ### ### ###### ##### ### *vp* ##### ## ### #### ###### #####, *fslabel*.

6.7.3.1.4. mpo_create_devfs_device

void **mpo_create_devfs_device**(*dev*, *devfs_dirent*, *label*);

dev_t *dev*;
struct devfs_dirent **devfs_dirent*;
struct label **label*;

####–
#####–
####
###

##–
####
###–
##–
#####–
###
####
devfs_dirent

devfs_dirent
###
##–
###–
##–
##
##–
###
##
##
##–
#####.

##–
###
###
devfs_dirent
##
##
######
##.

#####_###### ##### ####### ### ### ###### ######. #### #### #### ## #### #### ### ###### #### ###### ## #######, ###########, ## # ### ###### ## #### #########.

6.7.3.1.5. mpo_create_devfs_directory

void **mpo_create_devfs_directory**(*dirname*, *dirnamelen*, *devfs_dirent*, *label*);

char **dirname*;
int *dirnamelen*;
struct devfs_dirent **devfs_dirent*;

struct label *label;

####–
#####–
####
###
#####
dirname
##
##–
###–
##–
##
##–
###
###–
##–
##
######
name
##
len
######
dirname
##
devfs_dirent
###
##–
###–
##–
##
##–
###
###
##–
###–
##–
##
##–
###
###–
##–
##.

#####_###### ##### ####### ### ### ###### #########. #### #### #### ## #### #### ### ###### #### ###### ## #######, ###########, ## # ### ###### ######### # ######## ######### ######### ## #### #########.

6.7.3.1.6. mpo_create_devfs_symlink

void **mpo_create_devfs_symlink**(*cred*, *mp*, *dd*, *ddlabel*, *de*, *delabel*);

struct ucred **cred*;
struct mount **mp*;
struct devfs_dirent **dd*;
struct label **ddlabel*;
struct devfs_dirent **de*;
struct label **delabel*;

####–
#####–
####
###

cred ###–
####
###–
###–
####

mp ##–
###
#####
#####

dd ###
###–
##–
##–
####

dd ##–
la ##
bel ##–
##–
##–
##–
##
####
dd

de ###–
####
##–
###

####–
#####–
####
###

de–
la##
bel–
##–
##–
##–
##
####
de

(*delabel*) ### # ##### ####### #####(5) ######## #### #####.

6.7.3.1.7. mpo_create_vnode_extattr

int **mpo_create_vnode_extattr**(*cred*, *mp*, *fslabel*, *dvp*, *dlabel*, *vp*, *vlabel*, *cnp*);

struct ucred **cred*;
struct mount **mp*;
struct label **fslabel*;
struct vnode **dvp*;
struct label **dlabel*;
struct vnode **vp*;
struct label **vlabel*;
struct componentname **cnp*;

####–
#####–
####
###

cred–
####
###–
###–
####

mount
###–
###
#####
#####

####–
#####–
####
###

####
###–
###
##–
###

###–
###
##–
###–
##–
##
##–
###

###–
###
##–
##–
##–
##–
##
####
dvp

###–
##
###–
##–
##
##–
###

###–
##–
##
##–
###
##–
##–
##–
##–

####–
#####–
####
###
##
####
vp
####–
##–
####
####
###
vp

vp ## ### ########### ######## #########. ## ### ##### ########, #### ## *vlabel* #### ### #####, ### ###### 0. #########, ###### ## #####– ###### #####.

6.7.3.1.8. mpo_create_mount

void **mpo_create_mount**(*cred*, *mp*, *mnt*, *fslabel*);

struct ucred **cred*;
struct mount **mp*;
struct label **mnt*;
struct label **fslabel*;

####–
#####–
####
###
###–
####
###–
###–
####
##–
####;
####
###–
###
##–
###

####–
#####–
####
###

#####–
##

mnt###–
la##–
bel##
##–
###
##
##
######
##
###
mp

fs###–
la##–
bel##
##–
###
###
###
####
###–
###
mp
######.

###–#######. #### #### #### ## #### #### # ### #### ###### ## #######.

6.7.3.1.9. mpo_create_root_mount

void **mpo_create_root_mount**(*cred*, *mp*, *mntlabel*, *fslabel*);

struct ucred **cred*;
struct mount **mp*;
struct label **mntlabel*;
struct label **fslabel*;

####–
#####–
####
###

###
###–
6.7.3.1.8,####
#mpo_create_mount#.

##########. #### #### #### ## #### #### ### #### #### ###### ## #######, ##### ###_######_#####;.

6.7.3.1.10. **mpo_relabel_vnode**

void **mpo_relabel_vnode**(*cred*, *vp*, *vnodelabel*, *newlabel*);

struct ucred **cred*;
struct vnode **vp*;
struct label **vnodelabel*;
struct label **newlabel*;

####–
#####–
####
###

cred ###–
####
###–
###–
####

vp ##–
###
##
##–
##–
###

vn-ode-la-bel ##–
###–
###
###–
#–
##
##–

####– #####– #### ###
vp
newla- ###– *bel* ##– ### ###– #### ##– ### ## ##– ##### *vn-* *ode-* *la-* *bel*

##########.

6.7.3.1.11. **mpo_setlabel_vnode_extattr**

int **mpo_setlabel_vnode_extattr**(*cred*, *vp*, *vlabel*, *intlabel*);

struct ucred **cred*;
struct vnode **vp*;
struct label **vlabel*;
struct label **intlabel*;

####– #####– #### ###
cred ###– #### ###– ###– ####

####–
#####–
####
###
vp##–
###
###
#####
###
##–
###
##
##–
###
####–
###
vlabel##–
##
##–
###
##–
##–
##–
##–
##
####
vp
intlabel##–
###
#####
###

intlabel ## ## ######## #########. #### ## ###### #### vop_stdcreatevnode_ea.

6.7.3.1.12. mpo_update_devfsdirent

void **mpo_update_devfsdirent**(*devfs_dirent*, *direntlabel*, *vp*, *vnodelabel*);

struct devfs_dirent **devfs_dirent*;
struct label **direntlabel*;
struct vnode **vp*;

struct label *vnodelabel;

####-
#####-
####
###

devfs_dirent
####;
##-
###
##-
###-
##-
##
##-
###

dirent-la-bel
##-
###
###
devfs_dirent
##
##
##-
###-
##.

vp #####
###
##-
###

vnode-la-bel ###-
###
###
vp

devfs_dirent ##### #### ### ###### ##### ##### #####. #### #### #### ## #### #### # ##### ##### ### #### ############ ######### ## ###### ### ##### ###### #### #### ## ##### #### ## ### ##### ## ########. ## #### #### ## ####

#####, ######### # #### ## mac_vnode_create_from_vnode ## ###-####### ### ##### #####.

6.7.3.2. ### ###### ######## #####

6.7.3.2.1. mpo_create_mbuf_from_socket

void **mpo_create_mbuf_from_socket**(*so*, *socketlabel*, *m*, *mbuflabel*);

struct socket **so*;
struct label **socketlabel*;
struct mbuf **m*;
struct label **mbuflabel*;

####- #####- #### ###
socket ####- ## ####- ### ###
socketlabel ###- ##- ### ###- ### ### *socket*
m ##- ####; ####
mbuflabel ###- ##- ### ##- ### ## #### ## ### *m*

#####. #### #### ## #### #### # ### ######## ## ####### ## ######### ## ### ###### ### ###### ## ### ###### ####.

6.7.3.2.2. mpo_create_pipe

void **mpo_create_pipe**(*cred*, *pipe*, *pipelabel*);

struct ucred **cred*;
struct pipe **pipe*;
struct label **pipelabel*;

####–
#####–
####
###

cred ###–
####
###–
###–
####

pipe ####

pipelabel ####–
##
##
##–
###
##–
##–
##–
##–
##
####
pipe

##########. #### #### ## #### #### # ### #### ## #######.

6.7.3.2.3. mpo_create_socket

void **mpo_create_socket**(*cred*, *so*, *socketlabel*);

struct ucred **cred*;
struct socket **so*;
struct label **socketlabel*;

####–
#####–
####
###

###–
#######
###–
###–
####

##–
####;
####–
##
##
##–
###

###-
###
##
####
##
###
so

##########. #### #### ## #### #### # ###### ## #######.

6.7.3.2.4. mpo_create_socket_from_socket

void **mpo_create_socket_from_socket**(*oldsocket*, *oldsocketlabel*, *newsocket*, *newsocketlabel*);

struct socket ***oldsocket*;
struct label ***oldsocketlabel*;
struct socket ***newsocket*;
struct label ***newsocketlabel*;

####–
#####–
####
###

###–
###–
###

####-
#####-
####
###

####-
##

old#-
#sock-
et#
la#-
bel#
##-
##-
##-
##-
##
####
old-
sock-
et

new#
sock##-
et#

new#-
#sock-
et#
la#-
bel#
##-
##-
##-
##-
##
####
new-
sock-
et-
la-
bel

######, *newsocket*, ##### ######(2)##, ##### ## ### ######(2) ######, *oldsocket*.

6.7.3.2.5. mpo_relabel_pipe

void **mpo_relabel_pipe**(*cred*, *pipe*, *oldlabel*, *newlabel*);

struct ucred **cred*;
struct pipe **pipe*;
struct label **oldlabel*;
struct label **newlabel*;

####–
#####–
####
###

cred ###–
####
###–
###–
####

pipe ####

old-label ###–
####
###–
#–
##
##–
###
##–
##–
##–
##–
##
####
pipe

newla-bel ###
##
##–
###
##–
####
##
##–
###

####– #####– #### ###	## *pipe*

#####, *newlabel*, ## *pipe*.

6.7.3.2.6. mpo_relabel_socket

void **mpo_relabel_socket**(*cred*, *so*, *oldlabel*, *newlabel*);

struct ucred **cred*;
struct socket **so*;
struct label **oldlabel*;
struct label **newlabel*;

####– #####– #### ###	
cred	####### ###– ###– ####
so	####; ####– ##
oldlabel	### ### *so*
newlabel	##– #### ### *so*

######.

6.7.3.2.7. mpo_set_socket_peer_from_mbuf

void **mpo_set_socket_peer_from_mbuf**(*mbuf*, *mbuflabel*, *oldlabel*, *newlabel*);

struct mbuf **mbuf*;
struct label **mbuflabel*;
struct label **oldlabel*;
struct label **newlabel*;

####– #####– #### ###
mbuf ## ####– #### ##– ###### #### ####– ##
mbuflabel ## ## ## *mbuf*
oldlabel ##– ### – ### ### ### ####– ##
newlabel ## ##– ### ## ## ###### ###

####–
#####–
####
###

###
###
####–
##

#####. #### #### #### ## #### #### ### ##### ######## ## ######## ## ### ###### ######, #### ### ######### ## #### ###### #######.

6.7.3.2.8. mpo_set_socket_peer_from_socket

void **mpo_set_socket_peer_from_socket**(*oldsocket*, *oldsocketlabel*, *newsocket*, *newsocketpeerlabel*);

struct socket **oldsocket*;
struct label **oldsocketlabel*;
struct socket **newsocket*;
struct label **newsocketpeerlabel*;

####–
#####–
####
###

old-sock-et ####–
##

old-sock-et-label ###
###
old-sock-et

new-sock-et ####–
##

####– #####– #### ###	###– new- # sock- et ## ### peer- la-## ## bel #### ## ### *new-* *sock-* *et*

########. #### #### #### ## #### #### ### ###### #### ## #########, ### #### ## #### ### #### #########.

6.7.3.3. ####### ###### ######## #####

6.7.3.3.1. mpo_create_bpfdesc

void **mpo_create_bpfdesc**(*cred*, *bpf_d*, *bpflabel*);

struct ucred **cred*;
struct bpf_d **bpf_d*;
struct label **bpflabel*;

####– #####– #### ###	###– *cred* ####### ###– ###– ####
	### *bpf_d* ####; ### ##–

####–
#####–
####
###

#####–
###

bpf ##–
#–
##
##–
###
##
##
######
##
###
bpf_d

######–####. #### #### #### ## #### #### # ### ###### #### ## ###### ## # ####### #### ### ###### ####### ##########.

6.7.3.3.2. mpo_create_ifnet

void **mpo_create_ifnet**(*ifnet*, *ifnetlabel*);

struct ifnet **ifnet*;
struct label **ifnetlabel*;

####–
#####–
####
###

ifnet ###–
####
##–
###–
####

ifnet###–
la#–
bel##
##–
###

####–
#####–
####
###
##
####
##
###
ifnet

#########. #### #### ### ## #### #### # ### ####– #### ######### ####### ######### ## ### ######, ## #### # ################ ## ############ ###### ### #### ## ## # ###### ## # #### ######.

6.7.3.3.3. **mpo_create_ipq**

void **mpo_create_ipq**(*fragment*, *fragmentlabel*, *ipq*, *ipqlabel*);

struct mbuf **fragment*;
struct label **fragmentlabel*;
struct ipq **ipq*;
struct label **ipqlabel*;

####–
#####–
####
###
frag####
ment##
######
##
####–
####
frag##–
ment-#
la#
bel#–
###
###
frag-
ment
ipq##
##–

####–
#####–
####
###
#####–
###
#####
##
##
##–
#####
ipqla-
bel
##
##–
###
##
##
######
##
###
ipq

####–
########.

6.7.3.3.4. **mpo_create_datagram_from_ipq**

void **mpo_create_create_datagram_from_ipq**(*ipq*, *ipqlabel*, *datagram*, *datagramlabel*);

struct ipq **ipq*;
struct label **ipqlabel*;
struct mbuf **datagram*;
struct label **datagramlabel*;

####–
#####–
####
###
ipq
##–
#####–
###
#####

####–
#####–
####
###
ipqla-
bel
##
##–
###
###
ipq
data-
gram
##
##
##–
#####
data-
gram-
la-
bel–
###
##
##
######
##
###
data-
gram-
la-
bel

#########.

6.7.3.3.5. mpo_create_fragment

void **mpo_create_fragment**(*datagram*, *datagramlabel*, *fragment*, *fragmentlabel*);

struct mbuf **datagram*;
struct label **datagramlabel*;
struct mbuf **fragment*;
struct label **fragmentlabel*;

####– #####– #### ###
data####– *gram*####
data###– *gram*-# *la*# *bel*#– ### ### *data-gram*
frag####– *ment*#### ## ## ##– #####
frag###– *ment*-# *la*# *bel*#– ### ## ## ###### ## ### *data-gram*

####.

6.7.3.3.6. **mpo_create_mbuf_from_mbuf**

void **mpo_create_mbuf_from_mbuf**(*oldmbuf*, *oldmbuflabel*, *newmbuf*, *newmbuflabel*);

struct mbuf **oldmbuf*;
struct label **oldmbuflabel*;
struct mbuf **newmbuf*;
struct label **newmbuflabel*;

####–
#####–
####
###

*oldm-
buf*#–
###
(######)
####

*oldm-
bu-
fla-
bel*–
###
###
*oldm-
buf*

*newm-
buf*##
##
##
##–
#####

*newm-
bu-
fla-
bel*–
###
##
##
######
##
###
*newm-
buf*

########. #### #### ### ## #### ## # ###### ## ##########, ######–### #### ## #### ## ############ ### ######### ########.

6.7.3.3.7. mpo_create_mbuf_linklayer

void **mpo_create_mbuf_linklayer**(*ifnet*, *ifnetlabel*, *mbuf*, *mbuflabel*);

struct ifnet **ifnet*;
struct label **ifnetlabel*;
struct mbuf **mbuf*;
struct label **mbuflabel*;

####–#####–#### ###
ifnet ####–####–##–###–####
ifnet-label ####–##–### ### *ifnet*
mbuf #### ####–## ### ### ####–####
mbuf-label ####–##–### ## ## ###### ## ### *mbuf*

#########. #### #### ### ##

##########, ######### ### ### ## ##6 ######### ## ### ###4 ### ###6 ######.

6.7.3.3.8. mpo_create_mbuf_from_bpfdesc

void **mpo_create_mbuf_from_bpfdesc**(*bpf_d*, *bpflabel*, *mbuf*, *mbuflabel*);

struct bpf_d ***bpf_d*;
struct label ***bpflabel*;
struct mbuf ***mbuf*;
struct label ***mbuflabel*;

####–
#####–
####
###

bpf_d
##–
#####–
###

bpfla-
bel
##
##–
###
###
bpfla-
bel

mbuf
####
##
##
##–
#####

mbuf-
fla-
bel
##–
###
##
####
##
###
mbuf

##########. #### #### ## #### #### # ##### ## ######### ## ### ### ###### ########## #### ### ###### ### ##########.

6.7.3.3.9. mpo_create_mbuf_from_ifnet

void **mpo_create_mbuf_from_ifnet**(*ifnet*, *ifnetlabel*, *mbuf*, *mbuflabel*);

struct ifnet **ifnet*;
struct label **ifnetlabel*;
struct mbuf **mbuf*;
struct label **mbuflabel*;

####– #####– #### ###
ifnet #### ##– ###– ####
ifnet-label ##– ### ### *ifnet-la-bel*
mbuf ####– ## ### ### ####– ####
mbuf-la-bel ##– ### ##

####–
#####–
####
###
##
######
##
###
mbuf

#########.

6.7.3.3.10. mpo_create_mbuf_multicast_encap

void **mpo_create_mbuf_multicast_encap**(*oldmbuf*, *oldmbuflabel*, *ifnet*, *ifnetlabel*, *newmbuf*, *newmbuflabel*);

struct mbuf **oldmbuf*;
struct label **oldmbuflabel*;
struct ifnet **ifnet*;
struct label **ifnetlabel*;
struct mbuf **newmbuf*;
struct label **newmbuflabel*;

####–
#####–
####
###
oldm- ####
buf ##–
##
###
##–
###–
###
####–
####
oldm- ###–
bu- ##
fla- ##
bel ##–
###
###

####–
#####–
####
###
oldm-
buf
ifnet–
####
##–
###–
####
ifnet–
la-
bel
##–
###
###
ifnet
newm-
buf##–
##
##
##
##–
#####
###
###
####–
####
newm-
bu-
fla-
bel–
###
##
##
######
##
###
newm-
buf

#######– ###### #########. #### #### ## #### #### ## #### ## ## ## ######### ##### ### ####### #########.

6.7.3.3.11. mpo_create_mbuf_netlayer

void **mpo_create_mbuf_netlayer**(*oldmbuf*, *oldmbuflabel*, *newmbuf*, *newmbuflabel*);

struct mbuf **oldmbuf*;
struct label **oldmbuflabel*;
struct mbuf **newmbuf*;
struct label **newmbuflabel*;

####–#####–#### ###	
oldmbuf #### ####–####	
oldmbuflabel–### ### *oldmbuf*	
newmbuf ###–##–## ####–####	
newmbuflabel–### ###	

####–
#####–
####
###

newm-
buf

(*oldmbuf*). #### #### ### ## #### ## # ###### ## ##########, ######### #### ########## ## #### ####### #########.

6.7.3.3.12. mpo_fragment_match

int **mpo_fragment_match**(*fragment*, *fragmentlabel*, *ipq*, *ipqlabel*);

struct mbuf **fragment*;
struct label **fragmentlabel*;
struct ipq **ipq*;
struct label **ipqlabel*;

####–
#####–
####
###

frag- ##
ment ####–
####
####–
####

frag- ###–
ment- #
la ##
bel ##–
###
###
frag-
ment

ipq ##
####–
####
##–
#####–
###
#####

####–
#####–
####
###

ipqla-
bel
##
##–
###
###
ipq

######### ####### ## #### ###### ########## ## ## ######## (*fragment*) ######## ####### ### ##### ## ### ###### ## ######## ########## ##### (*ipq*). ###### (1) ### # ########## #####, ## (0) ### ## #####. #### #### ## #### #### ### ## ##### ######## ## #### ## ######## ######## ########## ##### ### # ##### ######## ########; ## #### #####, # ### ######## ########## ##### ### ## ############ ### ### ####–####. ######## ### ### #### ##### ##### ## ####### ### ########## ## ######### ######## ## ######### ## ###### #### ### ###### #### ## ## ########### ##### ## ### ##### ## ##### ###########.

6.7.3.3.13. mpo_relabel_ifnet

void **mpo_relabel_ifnet**(*cred*, *ifnet*, *ifnetlabel*, *newlabel*);

struct ucred **cred*;
struct ifnet **ifnet*;
struct label **ifnetlabel*;
struct label **newlabel*;

####–
#####–
####
###

cred
####
###–
###–
####

ifnet
####;
###–
####

####– #####– #### ###	##– ###– ####	
ifnet ###– la- bel ##	##– ### ### *ifnet*	
newla- bel ###	##– #### ##	##– ### ## *ifnet*

#########, *ifnet*, ##### ## ### ###### ###### #####, *newlabel*, ### ### ###### ####### ##########, *cred*.

6.7.3.3.14. mpo_update_ipq

void **mpo_update_ipq**(*fragment*, *fragmentlabel*, *ipq*, *ipqlabel*);

struct mbuf ***fragment*;
struct label ***fragmentlabel*;
struct ipq ***ipq*;
struct label ***ipqlabel*;

####– #####– #### ###	## *mbuf* ####– ####	

####– #####– #### ###
###– *mbuf* ##– ### ##– ### ### *mbuf*
ipq ####– #### ##– #####– ###
ipq la- ## *bel* ## ##– ### ## ## ##– ###– ## ### *ipq*

(*ipq*) ##### ## ### ########## ## ### ###### ## ######## #### ###### (*mbuf*).

6.7.3.4. ####### ######## #####

6.7.3.4.1. mpo_create_cred

void **mpo_create_cred**(*parent_cred*, *child_cred*);

struct ucred **parent_cred*;
struct ucred **child_cred*;

####–
#####–
####
###

parent_cred
###
###–
####
###–
###–
####

child_cred
###–
####
###–
###–
####

##########. #### #### #### ## #### #### ######(9) ## ####### ## # ##### ####### ###### #####. #### #### ###### ### ## ######## #### # ####### ####### ## ######## #####.

6.7.3.4.2. mpo_execve_transition

void **mpo_execve_transition**(*old*, *new*, *vp*, *vnodelabel*);

struct ucred **old*;
struct ucred **new*;
struct vnode **vp*;
struct label **vnodelabel*;

####–
#####–
####
###

old
#######
###
###–
####
###–
###–
####

####– #####– #### ###	
### *new* ###– #### ###– ###– #### ## ## ##– #####	
vp ###### ## ##– #– ####	
###– *vnodelabel* ###– ### ### *vp*	

(*new*) #### ### ###### ##–###### ####### ########## (*old*) ##### ## # ##### ########## ###### ## ######### ### ###### ##### (*vp*). #### #### ###### #### # ####### ######## ### ###### ##–### ### ### ## ### ######## ####### # ####### #### ### mpo_execve_will_transition ##### #####. ######## ### ###### ## ######### #### #### ###### ## ######## mpo_create_cred ### ####### ### ### ####### ########### ## ## ### ## ######### # ############# #####. ######## ###### ### ##### #### ##### ##### ############ ## #### #####–#### mpo_create_cred, #### ## #### ## ### ######### mpo_execve_will_transition.

6.7.3.4.3. mpo_execve_will_transition

int **mpo_execve_will_transition**(*old*, *vp*, *vnodelabel*);

struct ucred **old*;
struct vnode **vp*;
struct label **vnodelabel*;

####– #####– #### ###	cred ####### ###– ###– #### ###– ## ## ##– ####(2)	vp ## ##– #– ####	vnode label ###– ### ### vp

######### ####### ### ###### #### #### ## ####### # ########## ##### ## # ###### ## ### ######### ## ### ###### ##### ## ### ###### ####### ##########. ###### 1 ## # ########## ## ########, 0 ## ###. #### ## # ###### ####### 0, ## ###### ###### ######### ## ### ######## ## ## ########## ########## ## mpo_execve_transition, ## #### #### ### ###### ## # ###### ## ####### ###### ########## # ##########.

6.7.3.4.4. mpo_create_proc0

void **mpo_create_proc0**(*cred*);

struct ucred **cred*;

####–
#####–
####
###
###–
####
###–
###–
####
##
##
######
##

0, ### ###### ## ### ###### #########.

6.7.3.4.5. mpo_create_proc1

void **mpo_create_proc1**(*cred*);

struct ucred **cred*;

####–
#####–
####
###
###–
####
###–
###–
####
##
##
######
##

1, ### ###### ## ### #### #########.

6.7.3.4.6. mpo_relabel_cred

void **mpo_relabel_cred**(*cred*, *newlabel*);

struct ucred **cred*;

struct label **newlabel*;

####–
#####–
####
###

cred ###–
####
###–
###–
####

newla- ##
bel ###
##–
####
##
##–
###
##
cred

#####.

6.7.4. ###### #######

######. #########, ######## ### ######, ######### ## ## ###### ####### ##### ##### #### ####### ### ## #### ########### ###########, ########### (######## ######### # #####) ### ### ##### ####### ######## ## ### #########. ## ###### ####### ##### ##### ### ###### 0 ## ###### ### #########, ## ## #####(2) ##### #####. ### ####### ## ######## ### ##### ##### ###### ####### ########## ###### ####### #### ## ######## ## #######: ## ### ####### ###### ### ######### ## #######, ####### #### ## ########. ## ### ## ####### ####### # #######, # ####### #### ## ########. ## #### #### ### ###### ####### # ####–###, ### ##### ##### ## ###### ## ### #### #### ## ######## ##### ### ######### ##########, ########### ## ### error_select() ######## ## kern_mac.c:

#### ##########	#######
	######
	#####
	######
##### ##########	#####

#####-##### ##### #### ## ########### ######## ##### #### ### ### #### ## ########. ## #######, ### ##### ####### ########## ## ###### ## ### ######### #####: ###### ########, ####### #########, ###### ### #######, ###### ### #########, #####.

6.7.4.1. mpo_check_bpfdesc_receive

int **mpo_check_bpfdesc_receive**(*bpf_d*, *bpflabel*, *ifnet*, *ifnetlabel*);

struct bpf_d **bpf_d*;
struct label **bpflabel*;
struct ifnet **ifnet*;
struct label **ifnetlabel*;

####-#####-####	###	
bpf_d	####; ### ##-#####-###	
bpflabel	## ##-### ### *bpf_d*	
ifnet	####; ###-#### ##-###-####	
ifnetlabel	##-###	

####–
#####–
####
###
###
ifnet

######### ####### ### ### ######### ###### ###### ######### #### ### ###### ######### ## ## ######### ## ### ####### ## ### ###### ### ##########. ###### (0) ### #######, ## ## errno ##### ### ####### ######### #######: ###### ### ##### ##########, ##### ### #### ## #########.

6.7.4.2. **mpo_check_kenv_dump**

int **mpo_check_kenv_dump**(*cred*);

struct ucred **cred*;

####–
#####–
####
###
cred ###–
####
###–
###–
####

######### ####### ### ####### ###### ## ####### ## ######## ### ###### ####– ####### (### ####(2)).

6.7.4.3. **mpo_check_kenv_get**

int **mpo_check_kenv_get**(*cred*, *name*);

struct ucred **cred*;
char **name*;

####–
#####–
####
###
cred ###–
####

####– #####– #### ###	###– ###– ####	*name* ###– ### ##– ##– ###– #### ####– #### ####

######### ####### ### ####### ###### ## ####### ## ######## ### ##### ## ### ######### ###### ########### ########.

6.7.4.4. **mpo_check_kenv_set**

int **mpo_check_kenv_set**(*cred*, *name*);

struct ucred **cred*;
char **name*;

####– #####– #### ###	*cred* ###– #### ###– ###– ####	*name* ###– ### ##– ##– ###– #### ####–

####–
#####–
####
###

####
####

######### ####### ### ####### ###### ## ####### ## ### ### ######### ###### ########### ########.

6.7.4.5. mpo_check_kenv_unset

int **mpo_check_kenv_unset**(*cred*, *name*);

struct ucred **cred*;
char **name*;

####–
#####–
####
###

cred ###–
####
###–
###–
####

name ###–
###
##–
##–
###–
####
####–
####
####

######### ####### ### ####### ###### ## ####### ## ##### ### ######### ###### ########### ########.

6.7.4.6. mpo_check_kld_load

int **mpo_check_kld_load**(*cred*, *vp*, *vlabel*);

struct ucred **cred*;
struct vnode **vp*;

struct label **vlabel*;

####– #####– #### ###
cred ###– #### ###– ###– ####
vp ###– ### ###– ### ##– ###
vla- ##– *bel* ## ##– ##– ##– ##– ## #### *vp*

######### ####### ### ####### ###### ## ####### ## #### ### ######### ###### ####.

6.7.4.7. mpo_check_kld_stat

int **mpo_check_kld_stat**(*cred*);

struct ucred **cred*;

####– #####– #### ###
cred ###– #### ###–

####–
#####–
####
###
###–
####

######### ####### ### ####### ###### ## ####### ## ######## # #### ## ###### ###### ###### ##### ### ########## ##########.

6.7.4.8. mpo_check_kld_unload

int **mpo_check_kld_unload**(*cred*);

struct ucred **cred*;

####–
#####–
####
###
cred ###–
####
###–
###–
####

######### ####### ### ####### ###### ## ####### ## ###### # ###### ######.

6.7.4.9. mpo_check_pipe_ioctl

int **mpo_check_pipe_ioctl**(*cred*, *pipe*, *pipelabel*, *cmd*, *data*);

struct ucred **cred*;
struct pipe **pipe*;
struct label **pipelabel*;
unsigned long *cmd*;
void **data*;

####–
#####–
####
###
cred ###–
####
###–

####–
#####–
####
###

###–
####

pipe ###

pipela-bel ####–
##
##
##–
###
##–
##–
##–
##–
##
####

pipe

cred #####(2)
###–
####

da-ta #####(2)
###–
##

######### ####### ### ####### ###### ## ####### ## #### ### ######### #####(2) ####.

6.7.4.10. mpo_check_pipe_poll

int **mpo_check_pipe_poll**(*cred*, *pipe*, *pipelabel*);

struct ucred **cred*;
struct pipe **pipe*;
struct label **pipelabel*;

####–
#####–
####
###

cred ###–
####

####–
#####–
####
###

###–
###–
####

pipe ####

pipela- ####
bel

##
##–
###
##–
##–
##–
##–
##
####
pipe

######### ####### ### ####### ###### ## ####### ## #### *pipe*.

6.7.4.11. mpo_check_pipe_read

int **mpo_check_pipe_read**(*cred*, *pipe*, *pipelabel*);

struct ucred **cred*;
struct pipe **pipe*;
struct label **pipelabel*;

####–
#####–
####
###

cred ###–
####
###–
###–
####

pipe ####

pipela- ####
bel

####–
#####–
####
###
##
##–
###
##–
##–
##–
##–
##
####
pipe

######### ####### ### ####### ###### ## ####### #### ###### ## *pipe*.

6.7.4.12. mpo_check_pipe_relabel

int **mpo_check_pipe_relabel**(*cred*, *pipe*, *pipelabel*, *newlabel*);

struct ucred **cred*;
struct pipe **pipe*;
struct label **pipelabel*;
struct label **newlabel*;

####–
#####–
####
###
cred ###–
####
###–
###–
####
pipe ####
pipelabel ####
####
###–
#–
##
##–
###
##–

####– #####– #### ###	##– ##– ##– ## #### *pipe*	*newla-bel* ## ### ##– #### ## *pipela-bel*

######### ####### ### ####### ###### ## ####### ## ####### *pipe*.

6.7.4.13. mpo_check_pipe_stat

int **mpo_check_pipe_stat**(*cred*, *pipe*, *pipelabel*);

struct ucred **cred*;
struct pipe **pipe*;
struct label **pipelabel*;

####– #####– #### ###	*cred* ###– #### ###– ###– ####	*pipe* ####
	pipela-bel #### ## ##– ### ##–	

####–
#####–
####
###
##–
##–
##–
##
####
pipe

######### ####### ### ####### ###### ## ####### ## ######## ########## ####### ## *pipe*.

6.7.4.14. mpo_check_pipe_write

int **mpo_check_pipe_write**(*cred*, *pipe*, *pipelabel*);

struct ucred **cred*;
struct pipe **pipe*;
struct label **pipelabel*;

####–
#####–
####
###
###– *cred*
####
###–
###–
####
pipe
####– *pipelabel*
##
##–
###
##–
##–
##–
##–
##
####
pipe

######### ####### ### ####### ###### ## ####### ## ##### ## *pipe*.

6.7.4.15. **mpo_check_socket_bind**

int **mpo_check_socket_bind**(*cred*, *socket*, *socketlabel*, *sockaddr*);

struct ucred *cred*;
struct socket *socket*;
struct label *socketlabel*;
struct sockaddr *sockaddr*;

####–#####–#### ###
cred #### ###–###–####
socket ## ## #####
socketlabel ### ### *socket*
sockaddr #### ## *socket*

6.7.4.16. **mpo_check_socket_connect**

int **mpo_check_socket_connect**(*cred*, *socket*, *socketlabel*, *sockaddr*);

struct ucred *cred*;

struct socket **socket*;
struct label **socketlabel*;
struct sockaddr **sockaddr*;

####– #####– #### ###	
cred	#### ###– ###– ####
sock-et	####– ## ## ## ###– ####– ##
sock-et-la-bel	##– ## ##– ### ### *sock-et*
sock-addr	##– #### ## *sock-et*

######### ####### ### ####### ########## (*cred*) ### ####### ### ###### ###### (*socket*) ## ### ###### ###### ####### (*sockaddr*). ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ##########, ##### ### #### ## #########.

6.7.4.17. mpo_check_socket_receive

int **mpo_check_socket_receive**(*cred, so, socketlabel*);

struct ucred **cred*;

struct socket **so*;
struct label **socketlabel*;

####– #####– #### ###	
cred	###– #### ###– ###– ####
so	####– ##
socketlabel	###– #– ## ##– ### ##– ##– ##– ##– ## #### *so*

######### ####### ### ####### ###### ## ####### ## ####### ########### #### ### ###### *so*.

6.7.4.18. mpo_check_socket_send

int **mpo_check_socket_send**(*cred*, *so*, *socketlabel*);

struct ucred **cred*;
struct socket **so*;
struct label **socketlabel*;

####– #####– #### ###	
cred	###– ####

####- #####- #### ###	###- ###- ####	*so* ###- ##
	sock ###- *et* #- *ta* ## *bel* ##- ### ##- ##- ##- ##- ## #### *so*	

######### ####### ### ####### ###### ## ####### ## #### ########### ###### ### ###### *so*.

6.7.4.19. mpo_check_cred_visible

int **mpo_check_cred_visible**(*u1*, *u2*);

struct ucred **u1*;
struct ucred **u2*;

####- #####- #### ###	*u1* ##- #### ###- ###- ####	*u2* ##- ####

####- #####- #### ###
###- ###- ####

######### ####### ### ####### ########## *u1* ### ##### ##### ######## #### ### ###### ####### ########## *u2*. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ##########, ##### ### #### ## #########, ## ##### ## #### ##########. #### #### ### ## #### ## # ###### ## ##########, ##-####### ############# ###### ######## #### ## ps, ### ## ###### #######.

6.7.4.20. **mpo_check_socket_visible**

int **mpo_check_socket_visible**(*cred*, *socket*, *socketlabel*);

struct ucred **cred*;
struct socket **socket*;
struct label **socketlabel*;

####- #####- #### ###
cred ###- #### ###- ###- ####
sock-et ###; ####- ##
sock-et-la-bel ###- ### ### *sock-* *et*

6.7.4.21. mpo_check_ifnet_relabel

int **mpo_check_ifnet_relabel**(*cred*, *ifnet*, *ifnetlabel*, *newlabel*);

struct ucred **cred*;
struct ifnet **ifnet*;
struct label **ifnetlabel*;
struct label **newlabel*;

####-
#####-
####
###

cred ###-
####
###-
###-
####

ifnet ###-
####;
###-
####
##-
###-
####

ifnet-
la- ##-
bel #
###-
#-
##
##-
###
###
ifnet

newla-
bel #
##
##-
###
##-
####
##
###-

####– #####– #### ###	## ## ##– ##### ## *ifnet*

######### ####### ### ####### ########## ### ####### ### ###### ####### #####–#### ## ### ###### ##### ######.

6.7.4.22. mpo_check_socket_relabel

int **mpo_check_socket_relabel**(*cred, socket*, *socketlabel*, *newlabel*);

struct ucred **cred*;
struct socket **socket*;
struct label **socketlabel*;
struct label **newlabel*;

####– #####– #### ###	
cred	###– #### ###– ###– ####
sock-et	####; ####– ##
sock-et-la-bel	###– ### ###– #– ## ##– ###

####-#####-#### ###	### *socket*
## *newlabel* ##	##-#### ## ###-## ## ##-##### ## *socketlabel*

######### ####### ### ####### ########## ### ####### ### ###### ###### ## ### ###### ##### ######.

6.7.4.23. mpo_check_cred_relabel

int **mpo_check_cred_relabel**(*cred*, *newlabel*);

struct ucred **cred*;
struct label **newlabel*;

####-#####-#### ###	### *cred* ##-#### ###-###-####
## *newlabel* ##	

####–
#####–
####
###

##–
####
##
###–
##
##
##–
#####
##
cred

######### ####### ### ####### ########## ### ####### ###### ## ### ###### ##### ######.

6.7.4.24. mpo_check_vnode_relabel

int **mpo_check_vnode_relabel**(*cred*, *vp*, *vnodelabel*, *newlabel*);

struct ucred **cred*;
struct vnode **vp*;
struct label **vnodelabel*;
struct label **newlabel*;

####–
#####–
####
###

cred ###–
#######
###–
###–
####

vp ######
####;
##–
###

vn ##–
ode ###–
la ###
bel ##–

####–
#####–
####
###
#–
##
##–
###
###
vp
newla-
bel
##
##–
###
##–
####
##
###–
##
##
##–
#####
##
vp

######### ####### ### ####### ########## ### ####### ### ###### ##### ## ### ###### ##### ######.

6.7.4.25. mpo_check_mount_stat

int **mpo_check_mount_stat**(*cred*, *mp*, *mountlabel*);

struct ucred *cred;
struct mount *mp;
struct label *mountlabel;

####–
#####–
####
###
cred
####
###–

####–
#####–
####
###

###–
####

mp–
####;
####
###–
###
#####

mount-
la-
bel
##–
###
###
mp

######### ####### ### ####### ########## ### ### ### ####### ## # ###### ###– ###### ## ### #### ######. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ########## ## ##### ### #### ## #########. #### #### ### ## #### ## # ###### ## ##########, ######### ###### ########### ## ######(2) ### ####### #####, ## #### ## ## ######### #### #### ####### ## ####### #### ######## ## #### #######, #### ## #### #########(2) ## #######.

6.7.4.26. mpo_check_proc_debug

int **mpo_check_proc_debug**(*cred*, *proc*);

struct ucred **cred*;
struct proc **proc*;

####–
#####–
####
###

cred–
#######
###–
###–
####

####–
#####–
####
###
probe
####;
#######

######### ####### ### ####### ########## ### ##### ### ###### #######. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ########, ##### ### #### ## #########, ## ##### ## #### ########## ## ### ######. #### #### ### ## #### ## # ###### ## ##########, ######### ### ## ### ######(2) ### ######(2) ####, ## #### ## ### #### ##### ## ###### ##########.

6.7.4.27. mpo_check_vnode_access

int **mpo_check_vnode_access**(*cred*, *vp*, *label*, *flags*);

struct ucred **cred*;
struct vnode **vp*;
struct label **label*;
int *flags*;

####–
#####–
####
###
cred–
####
###–
###–
####
vp–
####;
##–
###
label–
##
##–
###
###
vp

####–
#####–
####
###

~~flags~~ ##
####(2)
#####

######### ### ########### ## ######(2) ### ####### ##### ## ### ####### ######–#### ###### ###### #### ######### ## ### ###### ##### ##### ### ###### ###### #####. #### ###### ######### ## ########### ##### ### #### ######### #### ## mpo_check_vnode_open. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ########## ## ##### ### #### ## #########.

6.7.4.28. mpo_check_vnode_chdir

int **mpo_check_vnode_chdir**(*cred*, *dvp*, *dlabel*);

struct ucred **cred*;
struct vnode **dvp*;
struct label **dlabel*;

####–
#####–
####
###

~~cred~~ ###–
####
###–
###–
####

~~dvp~~ ##–
####;
##–
###
##
#####(2)
##–
##

~~dlabel~~ ###–
##
##
##–

####–
#####–
####
###
###
###
dvp

######### ####### ### ####### ########## ### ###### ### ####### ####### ##–####### ## ### ###### #####. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ########, ## ##### ### #### ## #########.

6.7.4.29. mpo_check_vnode_chroot

int **mpo_check_vnode_chroot**(*cred*, *dvp*, *dlabel*);

struct ucred **cred*;
struct vnode **dvp*;
struct label **dlabel*;

####–
#####–
####
###
cred ###–
####
###–
###–
####
dvp ##–
###–
##–
##
##–
###
dla ###–
bel #
##
##–
###
##–
##–
##–
##–

####–
#####–
####
###

##
####
dvp

######### ####### ### ####### ###### ## ####### ## ######(2) #### ### #########
######### (*dvp*).

6.7.4.30. **mpo_check_vnode_create**

int **mpo_check_vnode_create**(*cred*, *dvp*, *dlabel*, *cnp*, *vap*);

struct ucred **cred*;
struct vnode **dvp*;
struct label **dlabel*;
struct componentname **cnp*;
struct vattr **vap*;

####–
#####–
####
###

cred ###–
####
###–
###–
####

dvp ##–
####;
##–
###

dla ###–
bel #
##
##–
###
###
dvp

cnp ###–
##–

####–
#####–
####
###
####
####
###
dvp
vap ##–
###
##–
####–
##–
##
###
vap

######### ####### ### ####### ########## ### ###### # ##### #### ### ###### ###### #########, ###### #### ###########, ### ###### ######### ###########. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ########, ## ##### ### #### ## #########. #### #### ### ## #### ## # ###### ## ##########, ######### ## # ###### ## ##### ## ####(2) #### #_#####, ######(2), ### ######.

6.7.4.31. mpo_check_vnode_delete

int **mpo_check_vnode_delete**(*cred*, *dvp*, *dlabel*, *vp*, *label*, *cnp*);

struct ucred **cred*;
struct vnode **dvp*;
struct label **dlabel*;
struct vnode **vp*;
void **label*;
struct componentname **cnp*;

####–
#####–
####
###
cred ###–
####
###–
###–
####

####–
#####–
####
###

dvp##–
###
##–
###–
##–
##
##–
###

dta##–
tlel
##
##–
###
###
dvp

vp##–
####;
##–
###
##
######

ka##–
tlel
##
##–
###
###
vp

dnp##–
##–
####
####
###
vp

######### ####### ### ####### ########## ### ###### # ##### #### ### ###### ###### ######### ### ###### #### ###########. ###### 0 ### #######, ## ## ermo ##### ### #######. ######### #######: ###### ### ##### ########, ## ##### ###

#########. #### #### ### ## #### ## # ###### ## ##########, ######### ## # ###### ## ##### ## ######(2) ### #####(2). ######## ############ #### ##### ##### ###### #### ######### mpo_check_rename_to ## ######### ######## ## ####### ## # ###### ## ##### ### ###### ## # ######.

6.7.4.32. mpo_check_vnode_deleteacl

int **mpo_check_vnode_deleteacl**(*cred*, *vp*, *label*, *type*);

struct ucred **cred*;
struct vnode **vp*;
struct label **label*;
acl_type_t *type*;

####– #####– #### ###	
cred ###– ####### ###– ###– ####	
vp ###### ####; ##– ###	
label ###– ## ##– ### ### *vp*	
type ### ####	

######### ####### ### ####### ########## ### ###### ### ### ## ###### #### #### ### ###### #####. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ########, ## ##### ### #### ## #########.

6.7.4.33. mpo_check_vnode_exec

int **mpo_check_vnode_exec**(*cred*, *vp*, *label*);

struct ucred *cred;
struct vnode *vp;
struct label *label;

####– #####– #### ###	####– ###	
cred	###– #### ###– ###– ####	
vp	##– ####; ##– ### ## ##– #– ####	
label	###– ##– ## ##– ### ### *vp*	

######### ####### ### ####### ########## ### ####### ### ###### #####. #####–######## ## ####### ######### ## #### ########## #### ######### ##### ### ####–######### #####. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ########, ## ##### ### #### ## #########.

6.7.4.34. mpo_check_vnode_getacl

int **mpo_check_vnode_getacl**(*cred*, *vp*, *label*, *type*);

struct ucred *cred;
struct vnode *vp;
struct label *label;
acl_type_t *type*;

####– #####– #### ###	
cred ###– #### ###– ###– ####	
vp ##– ####; ##– ###	
label ##– ## ##– ### ### *vp*	
type ####	

######### ####### ### ####### ########## ### ######## ### ### ## ###### #### #### ### ###### #####. ###### 0 ### #######, ## ## errno ##### ### #######. #######– ## #######: ###### ### ##### ########, ## ##### ### #### ## #########.

6.7.4.35. mpo_check_vnode_getextattr

int **mpo_check_vnode_getextattr**(*cred*, *vp*, *label*, *attrnamespace*, *name*, *uio*);

struct ucred **cred*;
struct vnode **vp*;
struct label **label*;
int *attrnamespace*;
const char **name*;
struct uio **uio*;

####– #####– #### ###	
cred ###– ####	

####–
#####–
####
###

###–
###–
####

vp##–
####;
##–
###

la###–
bel
##
##–
###
###
vp

dt##–
tr####–
name-
space
#######
####–
#####

name
####–
##
##–
#######
####

lib
#
#####–
####
#####–
##;
###
###(9)

######### ####### ### ####### ########## ### ######## ### ######## #########
#####. ########

######## ##### ######## ########## ### ## ########## ## ####### ######## ## ########## ## ##### ######## ##########. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ########, ## ##### ### #### ## #########.

6.7.4.36. mpo_check_vnode_link

int **mpo_check_vnode_link**(*cred*, *dvp*, *dlabel*, *vp*, *label*, *cnp*);

struct ucred **cred*;
struct vnode **dvp*;
struct label **dlabel*;
struct vnode **vp*;
struct label **label*;
struct componentname **cnp*;

####–
#####–
####
###

cred ###–
####
###–
###–
####

dvp ##–
###–
##–
##
##–
###

dla ###–
bel
##
##–
###
##–
##–
##–
##–
##
####
dvp

####–
#####–
####
###

vp####
###–
##–
##–
####
##–
###

label##–
##
##–
###
##–
##–
##–
##–
##
####
vp

cnp##–
##–
####
####
###
###
####
##–
###
###–
##–
##

######### ####### ### ####### ###### ## ####### ## ###### # #### ## ### ##### *vp* #### ### #### ######### ## *cnp*.

6.7.4.37. mpo_check_vnode_mmap

int **mpo_check_vnode_mmap**(*cred*, *vp*, *label*, *prot*);

struct ucred **cred*;

struct vnode *vp;
struct label *$label$;
int $prot$;

####- #####- #### ###	
cred	#### ###- ###- ####
vp	##- ### ## ###
label	###- ## ##- ### ##- ##- ##- ##- ## #### *vp*
prot	#### ###- ###- ##### (### ####(2))

######### ####### ### ####### ###### ## ####### ## ### ### ##### *vp* #### ### ########### ######### ## *prot*.

6.7.4.38. mpo_check_vnode_mmap_downgrade

void **mpo_check_vnode_mmap_downgrade**(*cred*, *vp*, *label*, *prot*);

struct ucred *$cred$;

struct vnode **vp*;
struct label **label*;
int **prot*;

####– #####– #### ###	
cred	###
vp	###– 6.7.4.37, ####
label	#mpo_check_vnode_mmap#.
prot	#### ###– ###– ##### ## ## ####– ####– ##

######### ### #### ########### ##### ## ### ####### ### ###### ######.

6.7.4.39. mpo_check_vnode_mprotect

int **mpo_check_vnode_mprotect**(*cred*, *vp*, *label*, *prot*);

struct ucred **cred*;
struct vnode **vp*;
struct label **label*;
int *prot*;

####– #####– #### ###	
cred	###– #### ###– ###– ####

####–
#####–
####
###

vp######
##–
###

prot###–
#–
##
###–
###–
#####

######### ####### ### ####### ###### ## ####### ## ### ### ######### ###### ########### ## ###### ###### #### ### ##### *vp*.

6.7.4.40. mpo_check_vnode_poll

int **mpo_check_vnode_poll**(*active_cred*, *file_cred*, *vp*, *label*);

struct ucred **active_cred*;
struct ucred **file_cred*;
struct vnode **vp*;
struct label **label*;

####–
#####–
####
###

active_cred###–
####
###–
###–
####

file_cred###–
###–
####
##–
##–
##–
##–
##

####–
#####–
####
###

####
###
######
####

vp######
##–
###

la###–
bel
##
##–
###
##–
##–
##–
##–
##
####
vp

######### ####### ### ####### ###### ## ####### ## #### ### ##### *vp*.

6.7.4.41. mpo_check_vnode_rename_from

int **mpo_vnode_rename_from**(*cred*, *dvp*, *dlabel*, *vp*, *label*, *cnp*);

struct ucred **cred*;
struct vnode **dvp*;
struct label **dlabel*;
struct vnode **vp*;
struct label **label*;
struct componentname **cnp*;

####–
#####–
####
###

cred###–
####
###–

####–
#####–
####
###
###–
####
dvp–
###–
##–
##
##–
###
dla###–
tlel
##
##–
###
##–
##–
##–
##–
##
####
dvp
vp##–
###
##
##
##–
#####
la###–
tlel
##
##–
###
##–
##–
##–
##–
##
####
vp

####–
#####–
####
###

cnp ###–
##–
####
####
###
vp

######### ####### ### ####### ###### ## ####### ## ###### ### ##### *vp* ## ####–##### ####.

6.7.4.42. mpo_check_vnode_rename_to

int **mpo_check_vnode_rename_to**(*cred*, *dvp*, *dlabel*, *vp*, *label*, *samedir*, *cnp*);

struct ucred **cred*;
struct vnode **dvp*;
struct label **dlabel*;
struct vnode **vp*;
struct label **label*;
int *samedir*;
struct componentname **cnp*;

####–
#####–
####
###

cred ###–
####
###–
###–
####

dvp ##–
###–
##–
##
##–
###

dla ###–
bel ##
##

####–
#####–
####
###
##–
###
##–
##–
##–
##–
##
####
dvp
####–
####–
###
##–
###
##–
bel
##
##–
###
##–
##–
##–
##–
##
####
vp
#####;
1
##
###
######
###
###–
##–
##–
####
##–
###–
##–

####–
#####–
####
###

####
###
###
####

cnp###–
##–
##–
####
###–
##–
####
####

######### ####### ### ####### ###### ## ####### ## ###### ## ### ##### *vp*, #### ### ######### *dvp*, ## ## ### #### ########### ## *cnp*. ## ##### ## ## ######## #### ## #########, *vp* ### *label* #### ## ####.

6.7.4.43. mpo_check_socket_listen

int **mpo_check_socket_listen**(*cred*, *socket*, *socketlabel*);

struct ucred **cred*;
struct socket **socket*;
struct label **socketlabel*;

####–
#####–
####
###

cred###–
####
###–
###–
####

sock-###
et####;
####–
##

sock-###–
et#–

####-#####-#### ###
label ### ### *socket*

######### ####### ### ####### ########## ### ###### ## ### ###### ######. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ########, ## ##### ### #### ## #########.

6.7.4.44. mpo_check_vnode_lookup

int **mpo_check_vnode_lookup**(, , , *cnp*);

struct ucred *cred;
struct vnode *dvp;
struct label *dlabel;
struct componentname **cnp*;

####-#####-#### ###
cred #### ###-###-####
dvp ####; ##-###
dlabel ## ##-### ### *dvp*

####–
#####–
####
###

cmp ###–
##–
####
####
##–
###
######
##

######### ####### ### ####### ########## ### ####### # ###### ## ### ###### ##–####### ##### ### ### ###### ####. ###### 0 ### #######, ## ## errno ##### ### ####–###. ######### ########: ###### ### ##### ########, ## ##### ### #### ## #########.

6.7.4.45. mpo_check_vnode_open

int **mpo_check_vnode_open**(*cred*, *vp*, *label*, *acc_mode*);

struct ucred **cred*;
struct vnode **vp*;
struct label **label*;
int *acc_mode*;

####–
#####–
####
###

cred ###–
####
###–
###–
####

vp ##–
####;
##–
###

label ###–
##–
##
##–
###

####- #####- #### ###	### *vp*
acc_mode	[####(2)] ##- #### ####

######### ####### ### ####### ########## ### ####### ## #### ######### ## ### ###### ##### #### ### ###### ###### ####. ###### 0 ### #######, ## ## ##### ##### ### #######. ######### #######: ###### ### ##### ########, ## ##### ### #### ## #########.

6.7.4.46. mpo_check_vnode_readdir

int **mpo_check_vnode_readdir**(, ,);

struct ucred *cred;
struct vnode *dvp;
struct label *dlabel;

####- #####- #### ###	
cred	###- #### ###- ###- ####
dvp	##- ####; ##- ###- ##- ## ##- ###
dlabel	###- ## ##

####–
#####–
####
###
##–
###
###
dvp

######### ####### ### ####### ########## ### ####### # readdir ######### ## ### ###### ######### #####. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ########, ## ##### ### #### ## #########.

6.7.4.47. mpo_check_vnode_readlink

int **mpo_check_vnode_readlink**(*cred*, *vp*, *label*);

struct ucred **cred*;
struct vnode **vp*;
struct label **label*;

####–
#####–
####
###
cred ###–
####
###–
###–
####
vp ##–
####;
##–
###
label ###–
##
##–
###
###
vp

######### ####### ### ####### ########## ### ####### # readlink ######### ## ### ###### ####### #####. ###### 0 ### #######, ## ## errno ##### ### #######. #########

#######: ###### ### ##### ########, ## ##### ### #### ## #########. #### #### ### ## #### ## # ###### ## ##########, ######### ## ######## readlink #### ## ### #### #######, ## ## # ###### ## ## ######## readlink ###### # #### ###### ## ### #######.

6.7.4.48. mpo_check_vnode_revoke

int **mpo_check_vnode_revoke**(*cred*, *vp*, *label*);

struct ucred **cred*;
struct vnode **vp*;
struct label **label*;

####–
#####–
####
###

cred ###–
####
###–
###–
####

vp ##–
####;
##–
###

label ##–
##
##–
###
###
vp

######### ####### ### ####### ########## ### ###### ###### ## ### ###### #####. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ########, ## ##### ### #### ## #########.

6.7.4.49. mpo_check_vnode_setacl

int **mpo_check_vnode_setacl**(*cred*, *vp*, *label*, *type*, *acl*);

struct ucred **cred*;
struct vnode **vp*;

struct label **label*;
acl_type_t *type*;
struct acl **acl*;

####–
#####–
####
###

cred ###–
####
###–
###–
####

vp ##–
####;
##–
###

label ###–
##
##–
###
###
vp

type ###
####

acl ###

######### ####### ### ####### ########## ### ### ### ###### ### ## ###### #### ## ### ###### #####. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ########, ## ##### ### #### ## #########.

6.7.4.50. **mpo_check_vnode_setextattr**

int **mpo_check_vnode_setextattr**(*cred*, *vp*, *label*, *attrnamespace*, *name*, *uio*);

struct ucred **cred*;
struct vnode **vp*;
struct label **label*;
int *attrnamespace*;
const char **name*;
struct uio **uio*;

####–
#####–
####
###
####–
####
###–
###–
####
###–
####;
##–
###
####–
###–
##
##–
###
###
vp
###–
#####–
###me-
###ce
#######
####–
#####
###me
####–
##
##–
#######
####
###b
#
#####–
####
#####–
##;
###
###(9)

######### ####### ### ####### ########### ### ### ### ######## ######### ## ###### #### ### ###### ######### ## ### ###### #####. ######## ############ ##–###### ###### ###### #### ######## ########### ### #### ## ####### ########## ########### ### ##### ##########. ############, ######## ###### ##### ###### ######### ##### ## ### #### ########### #### *uio*, ## ##### ## # ######### #### ######### ####### #### ##### ### ### ###### #########. ### *uio* ### #### ## NULL ## # ###### ######### ## ##### #########. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ########, ## ##### ### #### ## #########.

6.7.4.51. mpo_check_vnode_setflags

int **mpo_check_vnode_setflags**(*cred*, *vp*, *label*, *flags*);

struct ucred **cred*;
struct vnode **vp*;
struct label **label*;
u_long *flags*;

####–#####–#### ###	
cred	###–#### ###–###–####
vp	##–####; ##–###
label	###–##; ##–### ### *vp*
flags	##### #####; ### #######(2)

######### ####### ### ####### ########## ### ### ### ###### ##### ## ### ###### #####. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###–### ### ##### ########, ## ##### ### #### ## #########.

6.7.4.52. mpo_check_vnode_setmode

int **mpo_check_vnode_setmode**(*cred*, *vp*, *label*, *mode*);

struct ucred **cred*;
struct vnode **vp*;
struct label **label*;
mode_t *mode*;

####–#####–####	###
cred	###–#### ###–###–####
vp	##–####; ##–###
label	## ##–### ### *vp*
mode	####; ### #####(2)

######### ####### ### ####### ########## ### ### ### ###### #### ## ### ###### #####. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###–### ### ##### ########, ## ##### ### #### ## #########.

6.7.4.53. mpo_check_vnode_setowner

int **mpo_check_vnode_setowner**(*cred*, *vp*, *label*, *uid*, *gid*);

struct ucred **cred*;
struct vnode **vp*;
struct label **label*;
uid_t *uid*;
gid_t *gid*;

####– #####– #### ###	
cred ###– #### ###– ###– ####	
vp ###– ####; ##– ###	
label ###– ## ##– ### ### *vp*	
uid ### ##	
gid ### ##	

######### ####### ### ####### ########## ### ### ### ###### ### ### ###### ### ## #### ### ### #### ### ## ### ###### #####. ### ### ### ## ### ## (-1) ## ####### ## ######. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ########, ## ##### ### #### ## #########.

6.7.4.54. mpo_check_vnode_setutimes

int **mpo_check_vnode_setutimes**(, , , ,);

struct ucred *cred;
struct vnode *vp;
struct label *label;
struct timespec atime;

struct timespec mtime;

####– #####– #### ###	*cred*	*vp*	*label*	*atime*	*mtime*
	#### ###– ###– ####	####; ##	## ##– ### ### *vp*	#### ####; ### ######(2)	#– ##– ##– #### ####; ### ######(2)

######### ####### ### ####### ########## ### ### ### ###### ###### ########## ## ### ###### #####. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ########, ## ##### ### #### ## #########.

6.7.4.55. **mpo_check_proc_sched**

int **mpo_check_proc_sched**(*ucred*, *proc*);

struct ucred **ucred*;

struct proc **proc*;

####-#####-#### ###	
cred	###-#### ###-###-####
proc	## ####; #######

######### ####### ### ####### ########## ### ###### ### ########## ########## ## ### ###### #######. ###### 0 ### #######, ## ## ermo ##### ### #######. #######-## #######: ###### ### ##### ########, ##### ### #### ## #########, ## ##### ## ##### ##########.

###########(2) ### #### ###########.

6.7.4.56. mpo_check_proc_signal

int **mpo_check_proc_signal**(*cred*, *proc*, *signal*);

struct ucred **cred*;
struct proc **proc*;
int *signal*;

####-#####-#### ###	
cred	###-#### ###-###-####
proc	## ####; #######
signal	###- ##;

####-
#####-
####
###

###
####(2)

######### ####### ### ####### ########## ### ####### ### ###### ###### ## ### ###### #######. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ########, ##### ### #### ## #########, ## ##### ## ###-## ##########.

6.7.4.57. mpo_check_vnode_stat

int **mpo_check_vnode_stat**(*cred*, *vp*, *label*);

struct ucred **cred*;
struct vnode **vp*;
struct label **label*;

####-
#####-
####
###

cred ###-
####
###-
###-
####

vp ##-
####;
##-
###

label ###-
##
##-
###
###
vp

######### ####### ### ####### ########## ### stat ### ###### #####. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ###-#####, ## ##### ### #### ## #########.

####(2) ### #### ###########.

6.7.4.58. mpo_check_ifnet_transmit

int **mpo_check_ifnet_transmit**(*cred*, *ifnet*, *ifnetlabel*, *mbuf*, *mbuflabel*);

struct ucred **cred*;
struct ifnet **ifnet*;
struct label **ifnetlabel*;
struct mbuf **mbuf*;
struct label **mbuflabel*;

**####–
#####–
####**
###

cred ###–
####
###–
###–
####

ifnet ###–
####
##–
###–
####

ifnet###–
la–
bel##
##–
###
###
ifnet

mbuf ##
####;
####
##
##
####

mbuf###–
fla–
bel##
##–
###

####–
#####–
####
###
###
mbuf

######### ####### ### ####### ######### ### ######## ### ###### ####. ###### 0 ### #######, ## ## errno ##### ### #######. ######### #######: ###### ### ##### ########, ## ##### ### #### ## #########.

6.7.4.59. mpo_check_socket_deliver

int **mpo_check_socket_deliver**(*cred*, *ifnet*, *ifnetlabel*, *mbuf*, *mbuflabel*);

struct ucred **cred*;
struct ifnet **ifnet*;
struct label **ifnetlabel*;
struct mbuf **mbuf*;
struct label **mbuflabel*;

####– #####– #### ###	
cred	####– #### ###– ###– ####
ifnet	#### ##– ###– ####
ifnetlabel	##– ### ### *ifnet*
mbuf	####;

####–#####–#### ###	#### ## ## ##–###–####	
mbuf label	##–### ### *mbuf*	

######### ####### ### ###### ### ####### ### ######## ###### ## ### ###### #### ######. ###### 0 ### #######, ## ## errno ##### ### #######. ######### ########: ###### ### ##### ########, ## ##### ### #### ## #########.

6.7.4.60. mpo_check_socket_visible

int **mpo_check_socket_visible**(*cred*, *so*, *socketlabel*);

struct ucred **cred*;
struct socket **so*;
struct label **socketlabel*;

####–#####–#### ###		
cred	####### ###–###–####	
so	####; ####–##	

####- #####- #### ###	
sock-et-la-bel ###- #- ## ###- ### ### *so*	

######### ####### ### ####### ########## #### ### ##### ### ###### ###### (*socket*) ##### ###### ########## #########, #### ## ##### ######## ## #######(8) ### ########(1). ###### 0 ### #######, ## ## errno ##### ### #######. ######### ####-###: ###### ### ##### ##########, ##### ### #### ## #########, ## ##### ## #### ##########.

6.7.4.61. mpo_check_system_acct

int **mpo_check_system_acct**(*ucred*, *vp*, *vlabel*);

struct ucred **ucred*;
struct vnode **vp*;
struct label **vlabel*;

####- #####- #### ###	
ucred ###- #### ###- ###- ####	
vp ##- #####- ### ####; ##- ##(5)	
vla-bel ##- ### ##-	

####–
#####–
####
###
##–
##–
##–
##
####
vp

######### ####### ### ####### ###### ## ####### ## ###### ##########, ##### ## ### ##### ### ### ##### ## ### ########## ### ####.

6.7.4.62. mpo_check_system_nfsd

int **mpo_check_system_nfsd**(*cred*);

struct ucred **cred*;

####–
#####–
####
###
cred ###–
####
###–
###–
####

######### ####### ### ####### ###### ## ####### ## #### ######(2).

6.7.4.63. mpo_check_system_reboot

int **mpo_check_system_reboot**(*cred*, *howto*);

struct ucred **cred*;
int *howto*;

####–
#####–
####
###
cred ###–
####

####–#####–####	###	###–###–####	*how-to* ##–##–##–### #### ##–####(2)

######### ####### ### ####### ###### ## ####### ## ###### ### ###### ## ### ######### ######.

6.7.4.64. mpo_check_system_settime

int **mpo_check_system_settime**(*cred*);

struct ucred **cred*;

####–#####–####	###	*cred* ###–####	###–###–####

######### ####### ### #### ###### ## ####### ## ### ### ###### #####.

6.7.4.65. mpo_check_system_swapon

int **mpo_check_system_swapon**(*cred, vp, vlabel*);

struct ucred **cred*;
struct vnode **vp*;
struct label **vlabel*;

####-#####-#### ###	
cred ###-#### ###-###-####	
vp #### ##-####	
vlabel ##-### ##-##-##-##-## #### vp	

######### ####### ### ####### ###### ## ####### ## ### *vp* ## # #### ######.

6.7.4.66. mpo_check_system_sysctl

int **mpo_check_system_sysctl**(*cred*, *name*, *namelen*, *old*, *oldlenp*, *inkernel*, *new*, *newlen*);

struct ucred **cred*;
int **name*;
u_int **namelen*;
void **old*;
size_t **oldlenp*;
int *inkernel*;
void **new*;
size_t *newlen*;

####-#####-#### ###	
cred ###-####	

####–
#####–
####
###
###–
###–
####
name
######(3)
name-len
old
oldlenp
######## *inherit*;
1 del
##
######
####
###–
###
new
######(3)
newlen

######### ####### ### ####### ###### ## ####### ## #### ### ######### ######(3) ###########.

6.7.5. ##### ##########

####### ###### ##### #### # #### ####### ### ######### #### ### ##### ## ## ##–#### ## ########. # ######### ###### ######: #####, ## ###### ####### ##### #### ## ######### ## ######### ## ### ###### ## #### ##### ### #########, ### #### ### ###### ###### ## ######### ### # ######## ##### #####. ####### ##### ###### ######### ###### ### ######, ###### ##### #########, ### ## ###### ##### ###–###### ## ### #######. ###### ########## ###### ####### ## ###########, ## ##–##### ##### ### ### ######### ## #### (####### ###### ## ######## ####### ## ### ####### #####).

6.8. ########

########, ##–####### ### ####### ########## ### ########## ######## ######, ############# ## ### ###### ########### ########## ### ##### ######### ## ####### ### ######–

#####, ### # ### ## ##### ## ####### ### ###### ###### ## #########, #####, ### ####### ##########. #### ####### ## ### #### ############ #### ## ##### ## #### ####### ## ### #### ######.

6.8.1. #### ### ############### #####

###–####### ############ ## ####### ### ####### ## ######## ##### # ############–### #########. #### ####### ############ ## ########## ###### ### # ####### ## ######## ####### ##### ####### ## ####### ######## ########. ##### ########### ### #### ## ################ ##### #### ## #########(8), ##(1) ### ##(1) ## #### ###### ## ####### ###########, #####, ### #########. ### #### #### ####### ### ########## ##### ######### #######(8), #######(8), #######(8), ########(8), ### #######(8). ### ### #### ### ########## ## ###(3).

############ ###### ### ###### ## ### #####: ## ############ #### #### ## ###### ### ### ###### ## ########## ### ######## (mac_t), ### ############ #### ##### ## # ####### ########### ### ######## ## ############# #####, ####### ## ### ####, ## ##### #### ### ####. #### ### ##### ######### # ####### ## #########, #### #######–### ## # #### ### ##### ####. ###### ####### ## ### ###### #### ## ######## ##### ### ########## ### ###### ## ############### ####. ## ### ############ ###### ####, ###### ### ############ ## # ################ #### ## #### ### ##### ##### ######### ## ### / ##########. ###### ### ## ######### ######### ## ### #### #### ##### ######## ####; #### ########### ###### #### ### ######, ############ ##–### ####### #### ##### ## ######### ### ### ####### ##### ####### ###. #########, #### ## #### ## ### ## ### ####: ##### ###_#######(3) ### ## ######### #### ## ####### ##### ########, ## ### ## ### ######### ## ### #### #### ##### # ####### ####### ### #### ### ###.####(5) ############# ####. ########## ######## ###### ########### ####### ## ######## ####### ###### ########## #### ####### ####–### ##### ##### ## ### ######### ####### ## ### ######.

> **####**
>
> #########, ###### ############ ## ##### ######## ##### #### ## ########### ## # #### ######, ###### #######, ### ######–#### #### ## ## ############ ##### ## ### ######### ## ### ### #######. #### ########### ### ## ##### ## ### ###### ## #### ##### ######### ### ########### #######.

6.8.2. ####### ## ###### ##

#########, #############(3), ### #### ######## ## ######### ### ###### ########## #### # ###### #####

#####.####(5). ##### ###### ### #### ### ##### #### ##### #### ####### #### ##–#### LOGIN_SETALL ## #########, ## #### LOGIN_SETMAC ## ########## #########.

> **####**
>
> ## ## ######## ####, ## # ###### ####### ## #######, ### ### ##### ######## #### ## ######### #### ### login.conf #### ##### ###########, ### ## ########## ## # ######## ########. ###–####, ### ##############(3) ### ###### ###### ### #### ###–###### #### # ######.

6.9.

##–###### ###### ## # ###### ########## ######. #### ### ## #### ##### ## ######## ###### ##########, ## ##### ## ##### #### #### ## ########## #### ### ########## ## ### ### #########. ### ######### ## ############ ######## ## ######### # ##–##### ## ###### #####, ######### ########### #### ######## ######## #### ## ### ### ####, ## #### ## ######## ##### ## ######## ### ########### ## #### ######–#####. ###### ####### ### #### ## ####### #### ############# ## #### ## ######## ######## ####### #### ############ # ### ######## #######.

7. ######## #######

Contributed by Matthew Dillon.

7.1. ########## ## ######### ########vm_page_t

######## ###### ## ####### ## # ############ ##### ####### ### vm_page_t #####-####. ##### ## ######## ###### ### ############ ####### ### ######### ## ##### ########## vm_page_t ########## ## ### ## ####### ###### ######.

#####, ######, ########, #####, ## #### #####. ###### ### ### ##### #####, ### #### ## ######### ###### ## # ###### #### #### ##### #########-### ### ##### #### ## ## ##. ##### ##### ### ### ###### ## ### #####.

####### ########## # #### ######## ###### ##### ### ####### ### #### ##### ## ##### ## ######### #### ########. #### ## ##### ###### ######## ######## ###### ######## ######### ## ### #### ## ### ########### #1 ### #2 ######. #### # ### #### ##### ## ## #########, ####### ######## ## ###### ### #### ## ########## #### ####### #### ### ##### ## ##### ## ### #1 ### #2 ###### ######## ## ### ## ###### ### #### ## ##### ######### ###.

############, # #### ### ## #### #### # ######### ##### ## ###### #### # #### #####. ### ## ###### #### ########## ## ######### ####### ##### ### # #### ##### ### ##_#### ### ## ### ###### #####.

#####, #### ## ### ###### ###### ######## ## # ### #######. # #### ## ######### ###### ## # ##### ## ###### ##### #########. #### #####, ### #### ## ####### ########## #### # #### ##### #########. ### ## ###### #### ### #### ## ######## ##### ## # #### ###### ###### ##### (###) ## ##### ## #### #### ## # ##### ###### ###### #####. ##### #### ### ##### #### ### ##### ### ##### ########## #### # ## ###### ### ### ########## ### ######### #####. ##### ## ### #### ##### ### ##### ####. ####### ######## ## ######## ### ###### ## ##### ## ### #### #####, ### # ####### ####### ###### ## ##### #### ##### #### ## ########## ## ##### ## ########### #### ########## ## ######### ####.

(#### ## ### ######## ## ##### ######), # ########## ########### #### ############ ##### ###### ##### ###### ### #### ## ## ###########. ## ### #### #### ### ##### ## ###### ###### ## ###, ### ####### #### ##### ##### ### #### ## ####### ## #### ####.

####### ########### ##### ### ###### ###### ### ######## ## ######## ########## ###### ## ##### ## ### ####### ###### ## #### ## ######## ## ######## # ########## ######### ## ##### ###### ##### #####. ### ###### ## ########### #### ###### ####### ## ### ######## ###### ####. #### ########### ## ########### ## ### ####### ###### ### ######## ########## ##### ##### (####### #### #### ##### ####### #####), ######## #### ##### ### ######## ########## (######### ##### ######## ## ### ### ###### ## ###### #### ####### ######), ######### ##### ####### ###### #### ### ###### ### ### ## #######, ### ## #####. ######### ## ###### ## ####### ## #### # ########## ###### ## ############ #### ###### ## ######### ### ###### ## ### #### # #### ######## ##. #### ##### ## ###### ######### ##### #### ## ## #### ## ####### ## ######## # ####.

7.2. ### ####### ###### ######vm_object_t

####### ########## ### #### ## # ####### ### #######. ## ####### ### ## ########## #### ####### ##### ## ####### ##############, ###########, ######## ############, ## ########### #######. ##### ### ########## #### ### #### ## ####### ## ###### ####### #### ######## ## #####, ### ###### ## # ####### ###### #####.

shadowed. #### ##, #### ### ## ####### ## ### ## #### #####. ### #######, ### ##### #### # ########### ## ###### ####### ## ### ## # ##### ###### ## ###### ## ##### ## ######### # ###_####### ####()###. #### ######## ## #### #### ## ######### ####### ####### ##########, ######### #############, ### ###### ####### ######.

vm_page_t ### #### ## ########## #### ### ## ###### ## # ####. ### ## ###### ######### ########## ### ######### ####### ## ### #### #### ###### ######## #########.

7.3. ########## #/##struct buf

############ ## #######, #### ## ########### #######, ######### #### ## ######## ##### ### #####/##### #### ########### #### ### ## ######## #### ## #####/#####. ### #######, #### ### ## ###### ####### ## ########### # ######## #### ## ### ####### #####, ### ## ###### ##### ## #### ### #### ##### ###### ### #### ## ######## ####### ## ### ####### #####. ############, ########### #### ## ## #### ## ### ######## ## # #### ## #### ######## #### ### ## ##### ## ####### ## ##.

#######, struct buf##, ## bp##. #### # ########## ##### ## ####### ## # ####### ## # ## ######, ##

###. ## ### #### ######, #### #/# ## ######### ###### ## ####### ######## ## ####### #### ###### ########## ### #### ####### ### #/# ## ### ###### #####–#####. ### ########## ##_####_### ### ######### ###### ### ### ######## ## ### #/#. ########## ####### #### #### ##### ### ####### ## ##### ####, ##### ## ###–### ## ########## ###### #### ##### ##### ###### ####### ## ########## ####### ####### ## #### ## #####.

####### ######## # ####### ###### ## ### ## #### ######## #### ###### ####, ### ## ###### ## #### ##### #### #### ### ## #### ###### ## #### ######## ### #### ### ##### ### ####### ## ##### ####. ######## #### ####### ## ######## # ######## ## vm_page_t##, ### ########## #######. #######, ##### ########## ####### ### #### ## ######### #/#, #### ## ########## ##### ### ###### ## ########## #/# ########. #######, ## ##### ### ####### # ### ######## ########## ####### #########, #### ## ### ####### # #######.

7.4. ####### #### #######vm_map_t, vm_entry_t

####### ######### ### ######## #### ##### ######## #### ### ## ######. ### #### ########### #### ###### ### ## ############# ## ### ### ### ### ####### ######–#### #########. ####### #### ####### #### ## ##### ######## ### ### ######### ########### ############. ##### #### ###### ### ### #########.

####### ########## ######## ## ##_####### #### ####### ###### ## ####### ####–## ####### vm_map_t ### vm_entry_t ##########. #### ###### ### ######## ########### #### ### vm_map_t/vm_entry_t/ vm_object_t #########. ###### #### # ######### #### ####–#### ##### ### #### ######## ########## #### # vm_object; #### ## ### ##### ####. vm_page_t## ### #### ###### #### #### ###### #### #### ### ######## ########## ####. ### vm_page_t ### ## ###### #### ####### *pmaps*, ## #### ###### ### ######. #######, ### ############ ########### #####, ## ### ########## ## ### #### #### ## ### #### ###### ######### ### #### vm_page_t ### #### #### ## ###### ##### ###–######## ###### ### #####.

7.5. ### ######

####### #### ### ## #### ####### ###### ##########. ### ###### ####### ###### #### ## ### ## ### ########## ###### #####. #### ##, ######## ######## ## struct buf ########.

#####, ####### #### *not* ### ### ## ######## ###### #### ###. #### ##### #### ####### ### ###### ###### ############## ## ## 4# ## 32 ### #########. ## ####, ## ### ### #### ####### ## ##, ####### ##### ############# ######

############# ## ## 8## ## # 32 ### ########. #######, ##### #### 32 ### ####–##### ### #### ####### ## ####### 4## ## ###, #### ## # #### #####.

##########. ### #### ######### #### ## ###### ### ## ### *zone allocator*. ### #### ######### ##### # ##### ## ### ### ###### ## ## #### ############## ###### ## ###### ## ##### ## ######## # ######## #### ## #########. ### ### ### vmstat -m ## ### ## ######## ## ####### ### ########### ###### #### ## ####.

7.6. ###### ### ####### ##

######. ######### ### ## ### #### ## #### #### ######## ###### ### maxusers ### NMBCLUSTERS ###### ###### #######. #### ##, ###### ########### ####### ####–##### ## (#########) /usr/src/sys/i386/conf/CONFIG_FILE. # ########### ## ### ######### ###### ############# ####### ### ## ##### ## /usr/src/sys/i386/conf/LINT.

maxusers. ###### #######–## ##### #### 10 ## 128. #### #### ####### maxusers ### #### ### ##### ### ###### ## ######## ######### ### ######### ## ############# #########. ## ## ###### ## ##### maxusers ## #### ########## ###### ### ### ##### #######, #### ## NMB-CLUSTERS, ## ######## ######## #########.

#######, ### ### #### ## ######## NM-BCLUSTERS. ####### ###### ##### #### 1024 ## 4096.

NBUF ######### ## #### ############# #### ## ##### ### ######. #### ######–### ########## ### ###### ## ### ### ###### ### ### ## ### ########## ####### ### #/#. #### #### #### ######### ### ####### ########## ## ## #### ### ####### ###### #####! #### ######### ## ########### ##### ## 3.0######## ### ##### ###–#### ### ###### ######### ### ## ######## ########. ## ######### #### ### *not* ### ## ####### ## NBUF #########. ### ### ###### #### ##. ### ##### # ##### ### ###### ## ######### ########### ########## ######### ##### ### ##### # ##### ### ###### ### #### ###### ## ####### ### #### ##### ## ###### ##### ####.

#######, ####### ####### ### ### #########. ### ### ### ######### ### ####–######## ##### #### ### makeoptions ######### ## ### ###### #############. #### #### ### ###### ### ### -g ###### ### ### ########### ### ##### (######### 7 ## +) ####### #### ######.

```
makeoptions     DEBUG="-g"
makeoptions     COPTFLAGS="-O --pipe"
```

########. ### ######### ## ### #### ## #### #### ### ## ### ###### #########, ########## ### ## ####### ####.

###############. #####, ### #### ####### ## #### ###/### ########### ######## ########. /usr/src/sys/ufs/ffs/README.softupdates ######## ############ (### ############) ## ### ## ######### ##.

######, ######### ########## ####. ### ###### #### # #### ######### ########## ## #### ######## ####, ## ## ####, #### ## #### ###### #####. ### ###### #### ## ##### 2# ### #### ##### ## ### #### #### ######, ### ######## #### #### ## ### ## ### #### # ### ## ######. ### ###### #### #### #### #### ######### ##### ## ### ####### ###### ############# ### #### ###### ## ### ## ### ####### ## ### ## ### #### ## ########### #### ##### ##### ##. ## ### #### ## ## #### ## ########### # ##### ####, #### ##### #### ######### #### ## ## ##### ## ##### ## #### ###### ### /var/crash #### #### ########## #### ##### ## #### ### ####.

######### #### ## ######### ########## ## 4.# ## ##### #######, ### ### #### ## ##### #### ### ### ###### #### #### ### ##### ## ### ###### ####.

8. ##### ######

Written by John Baldwin and Robert Watson.

8.1.

############. #####, ### ##### ########## ### ##### ### ##########. ####, # ####### ############ ### ### ####### ######## ############### ### ######### ##### ## #### ###. ####, ####### ########## ### ######## ########## ### #########, ########### ### ########### ##### ## ######### ############ ############### ### ########### ### #### #########. #######, ######## ############## ##### ### ######## ## ######## ###### #######, ### #### ### ###### ##### ## ######### ############ ######### ### ### ## ######## ##########.

################, ### #### ## ####### ## ####### ######## ###### ### ############### ########### ########## #### ### ##### #######. #### ######## ######### ##### #### ## ####### ####, ### #### ## ####### ### ## #### ########. ####### ## ########### ######### ### ######## ### ## ######## ## ### ######## #######.

######. ### ###### ## ######### ### ###### ##### ### ####### #######. ## #### ### ###### ############## ## ### #### ## ### #### ##### #### ## #### ##### ######## ##############. ##### ####### #######, ######/######### #####, ##########, ### ######### #########. ### ### ########### ## ##### ### ##### ########### #####, ###### ### ### ######## ####### ## #### #######.

8.2. ##### ##### ### #######

8.2.1. ###### ############ ### ######

############, ## #### ####### #### ### ####### # ### ## ######. ## ### ## ######, ### ### ### ## ###### ####### ######### ####### # #### ## # #### ## #### ## ####### ###### ## #### ########. #### ####### ####### #### ### ######## #### ###### ######## ###### ######### ######## ##### ## ###### ##########; #### ## ### #### ### ######### ##### ###### ## ###### ##########. #### ##, # ###### ####### #### ### ##### ### ######## ## # ##### ##### ##### ## ##### ###### ## #####. #######,

#######. ### ### ## #### ## #### #### #### ## ### ##### ## ##### ###### ## #### ## ## #####. #######, ## ####### ### ######## ## ###### ########### ## ### #### #####, ### ##### ### #### ######### #### ### ####### ##### ## #### ####### ## ### ###### ### ##### #### ### ##### ########## #### ###### ######## ### #######.

#######, ######## # ###### ##### ##### #### ## ########## ####### #### ## ## ## #### ###### (## # ###### #####), #### ## ###### ########### ## ######### ## ### ###, ##### ##### ##### ####### ### ###### #### ##### ### ###### ## #### #### ##### #### ##### #### ### ####### ########## ###### # ###### #######.

############. ### #######, ## ### ##### ##### ##############, ## #### ## ### ## atomic_cmpset ###### #### ## atomic_set ## #### ## ### MTX_CONTESTED ###. ### ###### ## #### ## #### ### ##### ## mtx_lock #### # ######## ### #### #### # ######## ##### ## #### ####. #######, ### ##### ## #### ### ## #####, ## ## ### ###### ##### ## ### ###### ### ########. ####, #### ### atomic_set ########, ## ### ### ## ####### ### ### ## ####### ##### #### ### ### ## #### ### ######## ##. ####, ## #### ## ### ## atomic_cmpset ## ### ### ##### #### ## ### ##### ## #### ### ######## ## ## ########## ### #####.

#######, ###### ############ #### ##### ### #### ## ## ####### ## ####. ## ### ##### ## ########## ###### ####### #####, #### # #### #### ## #### #######. ### #######, ## ### ######## #### ## #### ### #### ###### #### ### ########## ####–#### ## #### #####, #### ##### ######## #### ## ######### ## # #### ###### #### ## ######## ###### ############.

8.2.2. #### ##### ###### #####

#####. #### ##### ## ##### #### ## ###### #### ### #### #### ### ######### ## ### #####. #######, #### ##### ###### ######## ######## ######. ######## ####### ### ###### #### # #####. ##### ###–###### ##### ## ##### ### ##### ### ###### ### ## ########### ## # ###### ## ####.

#####, ## ##### ### ## #### ## #### ###### ## ##### ## ######### #### #### ####### ### # ###### #### #### #######. #### ###### ## ##### ###############.

#######. ### ### ####### # ##### #### ######## #####. #### ### ####### #### #### ### ###### #### ## #### # #### #### ## ### ## ### #####. #######, ## ##### ## ### ####, ### #### ## #### # ##### #### ## ### ## ### #####. #### ### #### ####### ###### ######### ### ### ## ###### #### #### ## ######## ## ####### ####. ### #######, ### ###### ####### ####### ## ######### ## #### ### proctree_lock ## #### ### ### ########### #####. ######### ### #### #### ## ###### ## ## ### #### ######## ## ### ### # ###### ## # ####### ## #### ## #######

######. #######, ##### ###### #### ## inferior #### ## #### ### #### ## ######### ### ###### ######## ### ####### #### ####### ##### ## ########### ## #### ## # #### ## ######### #### ### ######### ### ### ######## ####### ##### ### #### ### ##### ### ### ####### ##### ## # ###### ## ### #####.

8.2.3. ####### ########## ###

####, ### ### ### #### #### ### #### ##### ####### ### ######## ### #### #### ### #### ###### ### ### ## ### ##### ### ####. #### ### #### ### ####, ### ######## ### ###### ######### #### ######## #######. ####, ### #### #### ### #### #### ##### ####### ### ######## ### ##### ########## ### ###### ## # ###### ## ### ####.

8.3. ####### ############ ###

8.3.1. #########

######### ### ####### ## ####### ##### ############## ####® #######, ####### ##### #### ######### ######## ## ###### #### ##### ### ###### #######. ######–### # ####### ### ######### ######## ###### #### ## ##### ## #####. ## #### ##### #######, #######, ######### ####### ### ## ######### ###### ########. ####, ##–####### ######## ###### ### ####### ### #### #### ## ##### ######## ##### ###–### #######. ## ########, ##### ######## ######## ### ##### ## ######### ######, ######### ######## ###### ### ##### ## ### # ######### #### ## ##### ######## ####### ######### #######.

##–####### #######. #### ### ###### #### ####### ######### ## ## ######### ###### ######## # #### ####### ######. ## ### ####### ##############, ### ###### ### ### ########## ### #### ########## #### ########### # ###### ###### ##### #### ## #### ##### ### ###### ###### ####### ## ######## ## ######## ###### #### ##### #### ## ########### ## ###.

########, ### ###### ## ####### ### #### #### ######–####. #########, ## #### ####### # ###### ###### #### ## ####### # ##### ##### ## #### ## ######### ##### ##. #######, ### #### ## ## #### ### ####### ###### ##### ########## ## ######### #####.

#######. #######, #### ######## ####### ######## ## ####### ######### #######. ##### ######### ######## ### ######### ######## ###### ######### ######## ##### ### INTR_FAST #### #### ## ####### ######## ## ### ###### ## #### ## #### ##### ########. ### #### #####–##### ##### ######### ### ##### ##### ## ######### ######## ### ##### #####–

#/# ###### ##########. ##### ##### ######## ## ### #### ##### ### #######, #### ### ### ####### ######## ##### ### #### ### #### ### #### ##–#####.

#######, ##### ## ### ######## ############ #### ### ## ##### ## ## #### ###### ########### ####### ########. ##### ## ######### ###### ######## ## # ###### #######, ## ### ###### ### ####### ## ### #######. ####, ## # ########### ####### ######, ### ###### ## ### ######### ###### #### ### ###### ######## ### ####### ### ####### ## ### ########### ######. ## ##### ## ###### #### ### ####### ## ### ########### ###### #### ### ######### ### #### ##### ##, ### ########### ###### ## ### ####### ## ####### ##### ### ######### ###### ## ## ###### ######### ### #######. #### ### ###### #### ### ######### ###### ###### ###### ## ########. ## ## ######### ###### ######, #### ## #### ### ### ### ####### #### ## ## #### ######## #####. ####, ## ### ####### ### ########### ######.

###–#### ### #### #### ##### ### ###### ## ##### ## # ##### ########### ###########. ## #### ##### ## ## ######## ### ##### ## ####, ### ## ####, #### ######## #### ########### ## ###### ### ######### ######## #### ########### ##### ## ##### ### ####### # ###### ###### #### #### #####. ####, ## ########### ###### ## ##–####### ######## ### #### ######## ## #### ##### #### ##### #### ##:

1. #### ######### ####### ### ### #####: # ######### ##### #### ## ####### ##–####### ####### ### # ####### ##### #### ## ### ### ###### #######.

2. ## ## ######### ####### ### # #########, #### #### ## ######### ## #########, ### ######### ## ###. ## ### ######### ####### #### #### ### ######### ## ##–##### ## ## ##### ####### ### ### ###### ####### #### ### #########. ## ### ######### ####### ##### ## ##### ## ## #########, #### ### ######## ####### ## ######### ## ###.

####### ##### ###### ####### ######## #### #### ###### ##### ##### ###### ###–########. ##### ## ### #### ## ###### ## #### ###### ## #### ##### ## ### ######, ## ## ######## #### ## ##### #### ## ##### ###### ####### ######## ##### ## #### ####### ## ### ##### ######### ######## ############ ### ########## ### ##### ###### ####### ######## ##### ## ##### ### ### #### ##.

8.3.2. ###### ########## ### ########

8.3.2.1. ###### ########## ## #

######. ### ##### #### ## #### # ### ###### ###### ## ##### ### ####### ######## #### #########. ####, #### ## ### ##### ## #####. ##### ### # ###### ## ##### ##### ### ####### ## ######### ### ##### ## ### ##### ##### #######.

############ #### ####### ########### ## #### ################: #### ### ######## # ###### ## ## ######### ## ####### ## ## # ### #####, ### ##### ## ### ## ### ######## ## ###### #### ### ######### ######### ######. ## ##, ### ######## # ####### ###### ## #### ######.

###########, ### ### ## ### ###–### ## ########### ####. ### ########, ## # ###### ####### # #### ##### ######### ### ### ### ###### ######## ## ##### ### #### #### #####, ### ### ###### ### #### ####### ## ### ############ ###### ### ##### ### # ###### ## #######. ####, #### #### #### ## ### #### ## ####### ## ######## ##### ####### ### # ####### ###### exec ## ### ##### ##### ## ### ## ########## ## ## ######### ### ####### ######## ###### ####. ########## ## ######### ### ##### #### ######### ## ##### # ######### #######.

8.3.2.2. ########

#######. #### # ##### ########### ######, ##### setrunqueue ## # ###### ##### #### ### ######## ###### ## # ########### #####. ### ############### ## ### critical_enter ## ### # ########### #### #### ## ####### ## ### ############. ## setrunqueue ## ###### #### #### #### ###, ## #### ### ######## ########## ## ### ###–##### ## ### ### ###### ######### ## ### ####### ######. #######, ##### ######## ######### ### #### ## ##### ######## ## ####### ####### ######## ### ######## #### ######## ### ## #########, ### ######### ####### ### #### ####### #######. ### #### ###### ### ######## ############### #### # ####### ##### ######## ## # ###### #### ###### ####.

########, ############ ###### ## # ######### ####### ### ##–###### ###### #### ########. ## # ###### #### ##### ######### ## ########## ## ## #### ######### ##### ### ######## ###### ## ## # ######### ########, #### # #### ###### #### ## ### ## ######### #### ##### ## # ####### ##########. #### ### ###–###### ######## ####### ## #######, ### #### ## ########. ## ### #### ## ###, #### ### ######## ###### ## ########## ## ##### ### ###### ######### ###### ## ###.

########## #### # ####### #### ####### ## #### #######. ## # ######### #####–#### ####### ##### # ####, ## ##### ## ### ########## ### #### ####### #### #### ## ##### ######## #### ########## ###########. ##########, ######### #### #####–#### ## ############ #### ######### ####### ### ## ##### ## ### cpu_critical_enter ### cpu_critical_exit ##########. ######### #### ### ######### ### ########### ########## ## ### ## ######### ####### ##########. #### ######## ### ### ## ###### #######, ### ## ## ###### ## ########### ### ###### ## ### #####. ##############, #### ###### ### #### #### ## #### ### #### ####### #### ### #### ## ####### ######### ###–####. #######, ## ##### ### #### ########, ## ## #### ### ### #### ####### ### #### ### ######### #########. ## ### ## ########## ## ##### ### ### ## ### #### ### ## ### ### #### ### ## ## ############ #### ### ## ### ## ### #### ##### ###############. ## ##### ######## ## #####, #### ### ## ### ###### ##### #### # ###### ## #### #### ## ## # ######## ###.

8.3.2.3. ######

#######, # ###### ## ########## #### #### #### ## ######### ##### ##### ####### ########## ### ### ###### ####### ### ##### ###########.

#######. ####### ## #### # ### ##### # ### # #### ### ######## ## ### ###### ## 4 ### ### ######## ## ### ###### ## 2. ## ### # ##### # ###### #### ######## 1 ########, #### ## ######, ## #### ### # ## ###### ## ### ### ###### ## #### ## #### ## ####### ### ### ####### ######## ######## #######. #######, ### #### ## ########### ##### ### ## ####### # ########## ## ## #### ## ######## ######### #### ### ### ## ### ##### #### ### ############### #### ##### ## ######## ##### ## ########. ####, ### ####### #### ##### ####### ##### ### # ## ###### ## ### ###### ######## ######. #### #### #### ##### #### ### ###### ## # ###### ######## ## ### # ## ######### # ###### ## ######## 1 ###### #### # ###### ## ######## 2.

######–## ###### #######. ## ### ###### #### ## ########## ####### #####, # ###### ## ###### ######### ########### (## ## #### ## # ######## ####### ## ######) ## # ###### ######## ###### ## #### ########. #######, #### ####### ######### ## ### ###### #### ####### ## # ###### ####### ### # ##### #### ###### ###### ######## ## ######### ## ########. ####, ## ### ###### ######## ##### ####### ## ### ##–##### ########### ###### ######, ### ###### ### ###### ### ### ######### ###### #### ###### ## ## ##### ## ##### ## #######. ### ##### ## ### ### #### #### ###### ## ### ### ######. #### ### ###### ####### ## ### ######### ######, ## #### ###### ### ### ##### ########### #### ### ####. ## ########, ### ##### ####### ######## ### ######### #### ##### ## ####### ## ### ###### ######## ### ########## ##### ### ##### ###### ####### ## ######## ## ########. ####, ## #######, ### ######–#### #### #### #### ####### ########### ## ### ###### ######## ###### ## # ##–####### ######## ######.

####### ## #### ###### ########## ### ### ###### ####### ### ##### ## # #####–#### ### ##### ## ####### #### #### ##########. ## ## ########## ###### ## ## ####### #### #### ##### ### #### ## ######## #########. ####, ##### ## # ###### ###### FULL_PREEMPTION ## ###### ########## ### ### ###### ####### #### ### ## #### ### ######### ########.

8.3.3. ######

###, # ###### ######## #### ## ##### #### ### ### ## #######. ## # #######–####### ###### #### ### #### ###### ## ############ ###### #### ## #### ####### msleep ## ######### ## ########. #######, ## ### ########## ######, ## ######### ### ##### # ########## ### ######## ######### ## ### ####. #### ### #### ######## ####### ## ####### #### ##### #### ### ######### ## curthread ### curpcb ### #### ### ###### ######## ### #######. ##### ### ### ########### ####### ## ### ####

####### ########### ####### #### ###### ###### #######. #### ## ## ######### ## ## #### ## ####### ######### ### ######## ## #### #### #### ####### #### ## ## ######.

######## ######## ######### ####### ######### ##### #### ## ### ##### ####### ########. #######, #### ### ## ### ###### ## # ########### ## ####### ## #### ##### ##### # ######## ####### #### ########### ###### ######### ####### ## ### ####### #########. ## # ######, ####### ### ### #### ######## ## ##### ### ####### ###### ## ######## #### ## ## ######### ## ###### ### ####### ## ####### ###.

#########. ### ### ######## ## ### #########: sched_pin ### sched_unpin. ##### ######### ###### # #### ###### ####### ##### td_pinned. # ###### ## ###### #### ### ####### ##### ## ####### #### #### ### # ###### ###### ### ######## #### # ####### ##### ## ####. #### ######### ############## ## ######## ## ###### #### ###### ####### ### #### ######## ## ### ### #### #### #### ######### ## ##### ### sched_pin ### ##### ######. ##### ### ####### ##### ## #### ####### ## ## ### ###### ###### ### ## #### #### ## ##### ####### #### ### ###### ###### ## ### ######### ### ##### sched_lock ## ####, #### td_pinned #### ### ##### ### #######. ### sched_pin ######## ########## ### ####### ##### ### sched_unpin ########## ### ####### #####. #### #### ##### ####–##### #### ####### ## ### ####### ###### ### #### ### ####### ###### ## ### ### ## ## ######### ## ## ### ####. ## #### ## ######### ###### ## # ######## ###, ### sched_bind ### sched_unbind ######### ###### ## #### #######.

8.3.4.

timeout ###### ######## ####### ###### ######## ## ######## ######### ### ######### ## #### ## ### softclock ######## #########. ###### ### ######### ##### ## # ####### ###### ## ##### #####, ### ######### ## ### ################# ######## #### ##### ## ############# ### ##### ####.

#####, callout_lock; ### ###### ## ### ####### #### #### ## ######### #### #### ##### ####. #### softclock ## ##### ##, ## ##### ### #### ## ####### ######## ### ##### #### ###### ####. ## ##### ## ##### #### ##### ########, ### softclock ###### #### ####### ### callout_lock ##### #### ######## ### ######## timeout ######## ########. ## ### CALLOUT_MPSAFE #### ### ### ### ###### ############, #### ##### #### ## ####### ###### ######## ### #######, ### #### ######## ##########. ### callout_lock ##### #### ## ########## ###### ##########. ### softclock #### ## ####### ## ##### ### #### ## # ########## ##### ##### ######### ### #####. ## DIAGNOSTIC ## #######, #### ### #### ##### ## ####### #### ######## ## ########, ### # ####### ## ###–###### ## ## ####### # #########.

8.4. ######## #######

8.4.1.

struct ucred ## ### ######## ######## ########## #########, ### ## ######### #### ## ### ##### ### ############## ###### ####### ###### ### ######. ########### ####### ### # ############### ##### ### ########## ####: ######## ########## ### ##### ### # ########## #########, ### #### # ###### ##### ## ## ####, ### ######### ## ##########, ########, ### #### ### ######### ########. ### ## ##### ###### ####### ## ### ########## ## ######### ###### ####### ## ####, #### ##–##### ## ########### ###### #######. #### # #### ## ############ ###, #### ##### #### ##### ############# ## ####### ########## ## ######### #### ############ #### ##### ## ## ######## ##########, ######## ### #### ### ######## #######–######## #### ######### # ############ ##########.

########## ########## #### # ###### ######### ### ########## #######; ###### ########## ########## #### ### ## ######## ## # #### ######### ## ######. # ##–###, cr_mtxp ######## ### ######### ##### ## struct ucred ## ## ## ######## #########–##. ### ### ## ### ######### ######## # ##### ######### ### ### ######## ## ### ###, ## ### ######### ### ## ######## ### #### ##### ### ############ ########.

struct ucred ##### ## # #### ##### ### ## ########### ### # ##### #### ### ###–######## #######.

#######, ########### ### #### ## # ######### ###### ### ###### ####### ####–#####, ### ## #### #### td_ucred ## ######### ######### ####### ## ######## ## #######. #### # ######## ########## ## ####### ### proc #### #### ## #### ###### ### ##### ### ###### ########## #### ##### #####. ### ####### ########## p_ucred #### ## #### ### ##### ### ###### ########## ## ####### #############, ######## ### #####.

##########, ### ##### ## td_ucred #### #### ## ######### ## ### ####### ####### #####. #### #### ####### ### ## # ##### ########## ######### # ######. ### ###### ############# ######### ### td_ucred ####### ## ### ###### ######### #### ### ####### p_ucred ######## # ####### ###### ### ######, ########## ### ## # ##### ########## ### ###### ###### #######.

8.4.2. #### ########### ### #### ##########

####### ## ######.

8.4.3. ####

struct prison ###### ############## ####### ######### ## ### ########### ## ##### ####### ##### ### ####(2) ###. #### ######## ### ######## ########, ## #######,

########. #### ######### ## ################# ##### ######## ## ######### ## ### ######### ### ###### ## #### ########## ##########. # ###### #####, pr_mtx ######## #### ### ##### ###### ## ### ######### ##### ### ### ####–### ######### ###### ### ###### ####. #### ######### ### ### #### #### ### #### ## #######, ### # ##### ######### ## ### struct prison ## ########## ## #### ##### ######. ### ####### ####### ## #### ##### ## ########## ### ######## ## sys/jail.h.

8.4.4. ###

#######, ## ### #### ## struct label. ## #######, ###### ## ###### ####### ### ######### ## ### #### #### ## ### ######### ## ### ###### ######. ### #######, ### v_label ##### ## struct vnode ## ######### ## ### ##### #### ## ### #####.

#######, ### ### ######### #### ######### # #### ## ########## ### ###### ########. ### ###### #### ## ###–###### ## # ###### ##### (mac_policy_list_lock) ### # #### ##### (#### ######### ## ### #####). ##### #### ###### ####### ###### ### ##### ## ########, ##### ## ### ######### ### # ######### ###### ## ### ###### #### ######## ####### ### ##### ##### ############ (### ##### ############) ### #### #####. ### ##### #### ### ## #### ### ### ######## ## ### ### ##### ############### ##########, #### ## ##### ########## ## #### ###### ############ ##########. ## ###### ### ###### ####, #### ## ###### ###### ############ ### ###############, ### ##### #### ## #### ### ### ######### ##### #### ## ####, ## ####### ############ ## ### #### ##### ## ## ## ###.

########, mac_policy_list_not_busy, ## ######### ## ####### #### #### ## #### ### ### #### ## ###### ######, ### #### ######### ######## #### #### ## ###### ## ## ### ###### ## ####### ## ##### #####, ## # #### ##### ######### ### ## ########. ### #### #####, ## ######, #### ## # #### ## ######/######### #### #### ###### ## ### ##########: ### ########### ## ####, ###### #### ## ## ####, ######### ####### ### ### #### ## ###### ###### ### ## ########, ###### #### ########## #### ##### ######## #### ####### ## ### #### ##### ### ##### ##### #### ### ## #### ## ##### ## (## ######) ### ### #########.

8.4.5.

####. #### #### ## # ######/######### (##) #### ### ### # #### ###### ## ####### ## ## ######## (###### ## ###########), ######### ##### ### # ### ###### #### #### #### ##### ## #### ###### ## ### #### #### ####. ##### ###### ### ## ####### ## sys/module.h ### ### ##### ##### ## ##### ## #####. ### #### ########## ######### ##### #### #### ### ### module_t ########## (#### ######) ### ### ###### modulelist_t #########, #######. ### ###### ###### ### ####### ###### #### ## kern/kern_module.c ## ####### ########## ### ####### ########.

8.4.6. ###### ######

####. ####### #### #### # ###### (####) #### (##_#####(9)) ### ####### #### #### ## ######### (#####) #### (##_#####(9)). ##–###### ######### #### ### ## ####### ## ###. ########## ####### #### #### #### ## ######. ##### ##### #### ## ### ###### ## ### #### ## ### ## #### #### ### ## ######, ##### #### #### ## ## #### ### #### ### ##### (#.#., ######_###_#####(9)). ##### #### ## ########## ### ####### ## ### ###### #### ##########, ## # ###### #### ###### ## ########## ### ### ###### # ########### #######.

8.4.7.

...

8.4.8. ######### ###

#########

#####,

#####, ######–### ##_#####

##########

########

8.4.9.

sched_lock ### ##### ######## ## ######## ########## ### ####### ##### ######### ## ### ########.

8.4.10. ###### ###

select ### poll ######### ###### ####### ## ##### ####### ## ###### ## #### ##–############### ##########, ####### ## ### ### #### ########### ### ######## ## ########.

...

8.4.11.

####/##### ###### ## ######### #### ########### #######. ## #### ### ####### ## ####### ##### ## ######### ## ######## ### ##### #### ### ##### ###### ######, ### #### ####### ## ##### ## ######## ## ##

#####. #### ###### ########### ##### ############# ######### ######## ##### #### ## NULL ## ### ###### ## ### ###########, ## ###### ## # struct sigio ########## ### ############. #### ##### ## ######### ## # ###### #####, sigio_lock. ####### ## ##### ########### ######### #### #### ## #### ##### ### ########## ## #### ##### ###–##### ###### ## ### ##### ### ### #### #### ########### ## ### ####.

struct sigio ## ######### ### #### ########### ###### ########## #### ### ####### ## ####### #####, ### ######### ############# ## ### ######, #####, ###### ##–#########, # ###########, ### ### ####### ########### ## ### ############. #### ####### ## ######### ##### ######### # #### ## ########## struct sigio ##########, p_sigiolst ### #########, ### pg_sigiolst ### ####### ######. ##### ##### ### #######–## ## ### ####### ## ####### ##### ##### ############. #### ###### ## #### struct sigio ### ######### ### ### ######### ## ### #############, #### ### ######### ## ### sio_pgsigio ##### ##### ##### ### struct sigio #### ### ####### ## ####### ##### ####. ########## ############ ### ###### ####### ########## ##### ####, ## #######, #### ## ##### ####### ######### ##### ##### ######## ##### ########## ####–#####, #### ## fsetown ## funsetown ## ##### ######### # #### ##### ####### ######### ##### ### ### ###### ##### ####. #### ## ######### ######## ####### ### ## ## ######## ######### ##### ## ### ##########, #### ## ######## ## # #### ########## ######### ## # #### ###### # #### #########.

8.4.12.

sysctl ### ####### ## ####### #### #### ###### ### ###### ### #### ######## ##–########## ##### # ###### ####. ## ##### ### ###### ### ###### ## #######: #####, ### ########## ## ### ########### ########### ### #########, ### ######, #####–####### #### ###### ######### ### ######### #### ### ######## ## ### ###### #########. ##### ###### ####### ### ###### ###### (### ############) ## ###–### ########## ### ############## ##########, ### ###### ######### #### ###### ##### ## ########### ####### ######### ### ##### #########. #########, ###### ##### ### ## # ###### ###### ## #### ## ########## ### ## sysctl; #######, ## ## ##–##### ## ####### ##### ##### ### ##### ########### ### ### ########. ### ######–### ## #### ####### ########## ## ####### ### ######## ####### ## ######.

###, ###### ### #######, ##### ### ## ######, ####, #### ### ### ##### ###, ######, #######. ###### ######## #### #### ####### ### ### ##### ### ### # ### ##### #### #### ###### ### ##### ## #### ## ###### ### ### #####. #######, ## ### ## ####### ## ### ### ###### ##### ### ### ## ### ###### ######## ## ##### #### ##########.

####, # ###### ##### ##### # ####### ## # ##### ## ### ######_### ###### ### ## ### ######. #### ##### #### ### #### ########. ### ###–### ######### ## ## #####, #### #######, ## ##### ####### ########## ####### # ###### ##### #####, ######_#### ##### ##### ## #### ## ### ### ####### #####.

8.4.13.

####. ### taskqueue_queues_mutex ## ##### ## ##### ## # #### ## ####### ### taskqueue_queues #####. ### ##### ##### #### ########## #### #### ###### ## ### ### ## ### struct taskqueue #### #########. ### ### ## ### ###########– #### ######### #### ## ## ####### ### ######### ## ### #### ## ### struct taskqueue. ## ###### ## ##### #### ##### ### ## ######## ###### ## ###### ### #### ## ####– ### #### ###/### ### #### ##### ##### ##### ### #### ###### ### ##### ## ## #### ####### ## kern/subr_taskqueue.c.

8.5. ##############

8.5.1. #####

#######. #### ###### #### ## ### ###### ## # #### ####### ####### # ##### ##### ######### ###### #### ##. #### # ###### ###### ## # #### #######, ## ####### ### ##### ##### ######### ## #### #### #######. ##### ###### ########## #### # #### ####### ### ###### ## # #### #####.

######. #### ##### ## ### #### ##### ## ###### # ########## #####. ### ##### ######## # ###### #### ## ##### ###### ### # #### #####. ### #### ##### ######## ### #### ## ##### ###### ## #### ## ### ######## ## ### ##### ##### ########## ## ### ####. #### ### ##### ##### ## ########## #### # ##### #### #######. ## ######## ####### ##### ## # #### ####### #### ### ##### ###### ########## #### ### ### ### ##### ###### ### ###### ## # #### ## #### ##### ###### ## ### ###### ##### #####. #### # ###### ## ####### #### ### ##### ##### ## ## ##### ### ## ### ##### ##### ########## #### ### ###### ####### #### #### ## ## ## ### ### #### ###### ###### ## ### #####. ### #### ###### ## ##### ### ###### ##### ##### #### ## ## #######. ##### ####### ### ## ####### #### ### ##### ##### ## # ######### ##### #### #### ### #####, # ###### ### ###### #### # ##### ##### #### # ######### ##### ##### ######### #### ### ### ## ####### ####.

sleepq_lock ######## ##### ### #### ##### ## ### ##### ##### ##### #### #### ## # ######## #### #######. ### sleepq_lookup ######## ##### ## ### #### ##### ### ### ###### ##### ##### ########## #### # ##### #### #######. ## ## ###### ##### ##### ## #####, ## ####### NULL. ### sleepq_release ######## ####### ### #### ##### ########## #### # ##### #### #######.

sleepq_add. #### ######## ####### ### #### #######, # ####### ## ### ##### #### ######## ### #### #######, # #### ####### ##– ######### ######, ### # #### ## #####. ### ##### ##### ##### ###### ## ######

sleepq_lock ###### #### ######## ## ######. ## ## ##### ######## ### #### ####### (## ## ## ######### ## #####), #### ### ##### ####### ######## ###### ## NULL. ### ##### ######## ######## # #### ##### #### ######### ### #### ## ##### ##### #### ### ###### ## ##### ##### ## ### # #### ## ######## ## ### ##### ## ############# (SLEEPQ_INTERRUPTIBLE). ######### ##### ### #### ### ##### ## ##### ######: ###–######## ##### ###### ####### ### ### msleep ### wakeup ######### (SLEEPQ_MSLEEP) ### ######### ######## ##### ###### (SLEEPQ_CONDVAR). ### ##### ##### #### ### #### ####### ######## ### #### ###### ### ######## ######### ########. #### #### ##### sleepq_add ###### ########## ###### ### ######### ########## ### #### ####–### ##### ### ########## ########## ##### ### #### ###### ### sleepq_lock ### ##–#### ######## ## ### ##### ##### ### ### ## ### ####### #########.

sleepq_set_timeout. ### ######### ####### ### #### ####### ### ### ####### #### ## # ######## #### ##### ## ### #########. ## # ##### ###### ## ########### ## ######### #######, ### sleepq_catch_signals ######## ###### ## ###### ## ####. #### ######### ####### ### #### ####### ## ### #### ##–#######. ## ##### ## ####### # ###### ####### ### #### ######, #### sleepq_catch_signals #### ###### # ###### ######; #########, ## #### ###### 0.

#####, ## ###### ##### ### ## ### sleepq_wait #########. ##### ### #### #### ######### ######### ## ####### ## ### ### ###### ###### ## ### # ####### ## #### ### ##### ####### ## ###### ####### ## ## #####–#### #### ### ######## ###### #########. ### sleepq_wait ######## ###### ##### ##–### ### ####### ###### ## ########### ####### ## ### ## ### ###### #########. ### sleepq_timedwait ######## ##### ##### ###### ### ###### ## ########## ####### ## ### ####### ### ## ## ####### #### ## sleepq_set_timeout #######. ### sleepq_wait_sig ######## ##### ##### ###### ### ###### ## ########## ####### ## ### ##### ## #######. ### sleepq_timedwait_sig ######## ##### ##### ###### ### ###### ## ########## #######, ### ####### ### ## ## ####### #### ## sleepq_set_timeout #######, ## ### ######## ##### ## #######. ### ## ### #### ######### ###### ### #### ####### ## ##### ##### ##–#######. ## ########, ### sleepq_timedwait_sig ######## ####### # ###### ####### ##–####### ## ######## ## ### ####### #### ## sleepq_catch_signals ##### # ####### ######.

######, #### # ##### ## #### ## ######## ## ### #### ######## ## ######## # ########## #####. ## ### ###### ## ####### ## ###### # ####### ## ## ######### #### ### ######## ###### ######### #### ## ########### ##### ##### ## ######## #######. #### #### ##### sleepq_wait ### #### ###### 0 ## #### ### ###### ######## ### ### ###### ###### ###### # ###–####### #####. ####, ## # ######## ##### ##### ### ### ## ####### ############# #### sleepq_timedwait_sig #### ###### ## ##### ########## #### # ####### ########. ## ## ##### ##### ## 0 ## ######## ### ###### sleepq_wait_sig ## sleepq_timedwait_sig ### #### ## #####, #### ### ######## sleepq_calc_signal_retval ###### ## ###### ## ##### ### ### ####### ####### ### ######### ## ########### ###### ##### ## ### ### #####. ### ###### ###### ######## ## ### ####### #### ## sleepq_catch_signals ###### ## ###### ## ### #### ######## ## sleepq_calc_signal_retval.

####### ###### ## # #### ####### ### ########## ####### ## ### sleepq_broadcast ### sleepq_signal #########. #### ######### ###### ### #### ####### #### ##### ## ##–#### #######, # ######## ## ##### ####### ####### ##, ### # ##### ######## ## ######## ##### #### ## ##### ##### ## ##### #######. ### ######## ######## ## ####### ## # ####### ########. ## # ###### ##### ####### ####### ### # ###### ######## (########### #####) #### ### ######## ######## #### ### ######## ## ### ########. ### ##### ######## ## #### ### ######## ########## ## ###### #### ##### ###### ### ### ##### ####### ## ### ##### ####. ### #######, ### ######### ######## ######### ###### ### ###### ####### ## # ########### ##### #####. ### sleepq_broadcast ######## ####### ### ####### #### ### ####### ## ### ######### #### ####### ##### sleepq_signal #### ####### ### ####### ######## ###### ####### ## ### #### #######. ### ##### ##### ##### ###### ##### ## ###### ### ### sleepq_lock ####–#### ###### ####### ##### #########.

sleepq_abort ####–####. #### ######## #### ## ###### #### sched_lock #### ### ### ###### #### ## ###### ## # ##### #####. # ###### ### #### ## ####### #### # ######## ##### ##### ### ### sleepq_remove ########. #### ######## ####### #### # ###### ### # #### ####### ## ## ######## ### #### ####### ### ###### ## ## ## ## ### ##### ##### ### ### ######### #### #######. ## ### ###### ## ### ## # ##### ##### ## ## ## ## # ##### ##### ### # ######### #### #######, #### #### ######## #### #######.

8.5.2.

#######/######## #### ##### ######.

######/####/#######. # ######## ###_######### ####.

###########.

8.5.3. ####### ## ### #####

###_#######() ##### ## ### ### ###### ###### #### #### ### ####### ## ###### #### #########?

8.5.3.1. ####

#######...

8.5.3.2. #####

#######

###_#### ## # ######### ##### #### ####### ### ######### ##### ####.

8.5.4.

##

####

8.6. #############

8.6.1. ######### ###### ### ###

####

#######

8.6.2. ##### ###### #########/######

sema_wait?

#####?

######.

########

######	## ######### ## ###### ## ### ## ### ####### ### ####### ## ##### #### ######## #### ### ###### ###### ######## ## ########. ## ### ##–######## #### ### ###### ############ ###–##### ######## ## ####### #############. ## # ###### #####, ## ####### ####### ## # #####–#### ### ######### ## # ####, #### # ### ## ##–######## ### ###### ## #### ### ### ######### ##### ####### ### #### ####### ######### ### #### ## ####### ### ## ### ##########. ### #### #########.
#####	# ###### ## ####### #### ## ## ####### ## # ####, ########, ## #########. ############# #### #### ## # ### ########## ## # ######. ### #### #####.
######## #######	# ####### ## #### #### ## ### ####### ## ## ###–######. # ######## ####### ## ####### ### ##–#### ##### ### ########_#####(9) ###.

##	####### #########. ### #### ##.
###### #########	# ###### ######### ##### ###/## ###### ## # ###### ########.
##	####### ###########. ### #### ##.
#########	### ###### #########.
####### ######### #######	####### ######### ####### ###### ## ### #### #### #### #### ## ######### ######. #### #### ### ###### ### ## ######### ####### ######## ## ######## ## ############ ######### ###### ## ####### ### ######### ######## ### # ##### ######### ######.
######## ###### ######	# #### ######## ###### ######. #########, ### #### ######## ######## ###### ####### ### ##– ####### #######. ### #### ######.
#####	# ###### ## ###### #### ## ## ####### ## # ###– ###### ######## ## # ##### ##### ### msleep ## tsleep. ### #### #####.
######### ####	# ######### #### ## # #### #### ### ## #### ## # ###### ##### ## ######. ####### ##### ### ## ##### ### ######### ### #### ######### ##### ## #######. ##########, #### ## ##### #### ## ### ####### ### ######## ##### ### ###### #### ######### #####. ### #### #####.
######	# ###### ###### ########### ## # ###### ######. ####### ### ##### ### #### # ###### ######### #######.
#### #######	# ###### ####### ####### #### ####### ### ##### ##.

##. #######

########

9. ########
########

Written by Murray Stokely.
Based on intro(4) manual page by Jörg Wunsch.

9.1.

#######. # ###### ## #### ####### ## # #### #### ###### ### ################ ##### #### ####### ## ### ######, #### #####, ########, ## # ######## ####### #### ### ########. # ###### ###### ## ### ######## ######### ## ### ######### ###### #### ######## # ######## ######. ##### ### #### ######### ############## ##### # ####### ###### ######## ### ######## ## # ###### ## ######## ####### ### ########## ########## ########. ###### ####### ### ## ######## #### ### ###### ########### ## ###### ## ###### ####### ### ####### ###### ###### ######## #####.

####®##### ######### ###### ### ######## ####### #############, ######### #### ###### ####### #####. ##### ##### ### ####### ####### ##### ### ######### /dev ## ### ########## #########.

##########; ######### ### ####### ###### #######.

9.2. ####### ###### ###### ######## #

######. #### ###### ###### ###### ####### ## #### ##### ### ####### #### # ####### ###### ####### ########## ######### ## #### #######.

#######:

- kldload # ##### # ### ###### ######
- kldunload # ####### # ###### ######
- kldstat # ##### ###### #######

######## ###### ## # ###### ######

```
/*
 * KLD Skeleton
 * Inspired by Andrew Reiter's Daemonnews article
 */

#include <sys/types.h>
#include <sys/module.h>
#include <sys/systm.h>  -/* uprintf */
#include <sys/errno.h>
#include <sys/param.h>  -/* defines used in kernel.h */
#include <sys/kernel.h> -/* types used in module initialization */

/*
 * Load handler that deals with the loading and unloading of a KLD.
 */

static int
skel_loader(struct module *m, int what, void *arg)
{
  int err = 0;

  switch (what) {
  case MOD_LOAD:                -/* kldload */
    uprintf("Skeleton KLD loaded.\n");
    break;
  case MOD_UNLOAD:
    uprintf("Skeleton KLD unloaded.\n");
    break;
  default:
    err = EOPNOTSUPP;
    break;
  -}
  return(err);
}

/* Declare this module to the rest of the kernel */

static moduledata_t skel_mod = {
  -"skel",
  skel_loader,
  NULL
};

DECLARE_MODULE(skeleton, skel_mod, SI_SUB_KLD, SI_ORDER_ANY);
```

9.2.1.

####### ######## # ###### ######## ## ######## ######### # ###### ######.

```
SRCS=skeleton.c
KMOD=skeleton

.include <bsd.kmod.mk>
```

####### make #### #### ######## #### ###### # #### skeleton.ko #### ### ## ###### #### ### ###### ## ######:

```
# kldload --v -./skeleton.ko
```

9.3. #########

#######. #### ## ### #### ###### #### ## ###### ###### ### ##### ### ###### ## ###### ######## ## ### ###### ####.

####.

9.1. ####### ## # ###### #### ####### ###### ###### ### ####### 10.#

```
/*
 * Simple Echo pseudo-device KLD
 *
 * Murray Stokely
 * Søren (Xride) Straarup
 * Eitan Adler
 */

#include <sys/types.h>
#include <sys/module.h>
#include <sys/systm.h>  -/* uprintf */
#include <sys/param.h>  -/* defines used in kernel.h */
#include <sys/kernel.h> -/* types used in module initialization */
#include <sys/conf.h>   -/* cdevsw struct */
#include <sys/uio.h>    -/* uio struct */
#include <sys/malloc.h>

#define BUFFERSIZE 255

/* Function prototypes */
static d_open_t      echo_open;
static d_close_t     echo_close;
static d_read_t      echo_read;
static d_write_t     echo_write;

/* Character device entry points */
static struct cdevsw echo_cdevsw = {
 .d_version = D_VERSION,
 .d_open = echo_open,
 .d_close = echo_close,
```

```
 .d_read = echo_read,
 .d_write = echo_write,
 .d_name = -"echo",
};

struct s_echo {
 char msg[BUFFERSIZE + 1];
 int len;
};

/* vars */
static struct cdev *echo_dev;
static struct s_echo *echomsg;

MALLOC_DECLARE(M_ECHOBUF);
MALLOC_DEFINE(M_ECHOBUF, -"echobuffer", -"buffer for echo module");

/*
 * This function is called by the kld[un]load(2) system calls to
 * determine what actions to take when a module is loaded or unloaded.
 */
static int
echo_loader(struct module *m __unused, int what, void *arg __unused)
{
 int error = 0;

 switch (what) {
 case MOD_LOAD:                -/* kldload */
  error = make_dev_p(MAKEDEV_CHECKNAME -| MAKEDEV_WAITOK,
      &echo_dev,
      &echo_cdevsw,
      0,
      UID_ROOT,
      GID_WHEEL,
      0600,
      -"echo");
  if (error -!= 0)
   break;

  echomsg = malloc(sizeof(*echomsg), M_ECHOBUF, M_WAITOK -|
      M_ZERO);
  printf("Echo device loaded.\n");
  break;
 case MOD_UNLOAD:
  destroy_dev(echo_dev);
  free(echomsg, M_ECHOBUF);
  printf("Echo device unloaded.\n");
  break;
 default:
  error = EOPNOTSUPP;
  break;
 }
 return (error);
```

```
}

static int
echo_open(struct cdev *dev __unused, int oflags __unused, int devtype __unused,
    struct thread *td __unused)
{
 int error = 0;

 uprintf("Opened device \"echo\" successfully.\n");
 return (error);
}

static int
echo_close(struct cdev *dev __unused, int fflag __unused, int devtype __unused,
    struct thread *td __unused)
{

 uprintf("Closing device \"echo\".\n");
 return (0);
}

/*
 * The read function just takes the buf that was saved via
 * echo_write() and returns it to userland for accessing.
 * uio(9)
 */
static int
echo_read(struct cdev *dev __unused, struct uio *uio, int ioflag __unused)
{
 size_t amt;
 int error;

 /*
  * How big is this read operation?  Either as big as the user wants,
  * or as big as the remaining data.  Note that the -'len' does not
  * include the trailing null character.
  */
 amt = MIN(uio->uio_resid, uio->uio_offset >= echomsg->len + 1 -? 0 -:
     echomsg->len + 1 -- uio->uio_offset);

 if ((error = uiomove(echomsg->msg, amt, uio)) -!= 0)
  uprintf("uiomove failed!\n");

 return (error);
}

/*
 * echo_write takes in a character string and saves it
 * to buf for later accessing.
 */
static int
echo_write(struct cdev *dev __unused, struct uio *uio, int ioflag __unused)
{
```

```
 size_t amt;
 int error;

 /*
  * We either write from the beginning or are appending --- do
  * not allow random access.
  */
 if (uio->uio_offset -!= 0 && (uio->uio_offset -!= echomsg->len))
  return (EINVAL);

 /* This is a new message, reset length */
 if (uio->uio_offset == 0)
  echomsg->len = 0;

 /* Copy the string in from user memory to kernel memory */
 amt = MIN(uio->uio_resid, (BUFFERSIZE -- echomsg->len));

 error = uiomove(echomsg->msg + uio->uio_offset, amt, uio);

 /* Now we need to null terminate and record the length */
 echomsg->len = uio->uio_offset;
 echomsg->msg[echomsg->len] = 0;

 if (error -!= 0)
  uprintf("Write failed: bad address!\n");
 return (error);
}

DEV_MODULE(echo, echo_loader, NULL);
```

###:

```
# echo --n -"Test Data" > -/dev/echo
# cat -/dev/echo
Opened device -"echo" successfully.
Test Data
Closing device -"echo".
```

#######.

9.4. ##### ####### (### ####)

####® ####### ### ####### # ###### #### ## #### ###### ##### ## ##### #######. ##### ####### ### #### ####### ### ##### ### ###### ######## #######. #### ####### ##### ############# ###### ########, ## ## ##### ########### ##–########. ### ####### #### ####### ### ######## ## ##### ##########, ######### ### ########### ## ### ####### ## #### ### ##### #### ######## ## ### ### ##–

####. #### ##### ########### ### ######## ##### ######## ## ####### #### ########## (###########, ######### ###.) ##########. ##### ###### ### ## #######, ##### ## ## ### ### ###### ### ###### ## ### ########### ##### ###– ####### ##### ######### ########### # ##### #####, #### ####### ######### ### ########### #######. ### #### ######, ## ####### ############ #### ## ##### ##– #####, ### ## ####, ###### ### ############ ##### ###### ##### ######## #### ##### ##### ## ####### #### ########## (## #####) ####### ###### ###### ## ####. ####### ### ############### ## ### ######## ## #### #### (#########) ## ### ##– ##### #### ######### ######### ############# ########### ### ######## ###### #### ####### ####### ####### ### ###### #### ####### ## #### ## ### #########– #### ## ### #### #/# ###############.

9.5. #######

####### ### ####### ####### ## ### ### ###### ##### ## ##### ## ## ########. ##### ######### ## ##### ## ##### ######### #### ###### ### ###### ### ####### ## ####### ####(), ### ## # ####### ###### ## ######### ########## ## ##### ### ###### #### ######(2).

#####(9), ### ###### ## ### ######## ######, ### #### ###### ####### #######.

10. ### #######

Written by Sergey Babkin.
Modifications for Handbook made by Murray Stokely, Valentino Vaschetto and Wylie Stilwell.

10.1.

##–####. ### ########### ######### #### ## ###### ######## ### ########### ## ### #### #### ### ## ##### #### ###########. ## ###### ### ####### ########## ## ### ####### ## ### ##########. ### ######### ######## ### ## ##### ## ### ###### #### ## #### #######. ## ########## ### ####### ep ### aha ### #### ####### ## #####–######.

10.2. #####

#####:

```
#include <sys/module.h>
#include <sys/bus.h>
#include <machine/bus.h>
#include <machine/resource.h>
#include <sys/rman.h>

#include <isa/isavar.h>
#include <isa/pnpvar.h>
```

#########.

#######, ### #### #####–##### ### ######## ## ########## ###### #########.

###. ### # ############ ###### ##### ##### #### ##### ##:

- static void xxx_isa_identify (driver_t *, device_t); ######## #### ### ### #######, ### ###### #######. ### ### ### ####### #### ###### ### #### ####### ###: ## ### ###### ######## #### ############### (#######) ### ## ########### ####### #### ###–#### ### ######### ##.
- static int xxx_isa_probe (device_t dev); ##### ### # ###### ## # ##### (## ###) ########. #### ####### ### #### ########### ############### ############## ## ########## ### ######### ########## #######.

- static int xxx_isa_attach (device_t dev); ###### ### ########## ######.
- static int xxx_isa_detach (device_t dev); ###### ###### ###### ######### ### ###### ######.
- static int xxx_isa_shutdown (device_t dev); ####### ######## ## ### ###### ###### ###### ########.
- static int xxx_isa_suspend (device_t dev); ####### ### ###### ###### ### ###### #### ## ### ########## #####. ### #### ##### ########## ## ### ########## #####.
- static int xxx_isa_resume (device_t dev); ###### ### ###### ######## ##### ###### #### ###–####### #####.

xxx_isa_probe() ### xxx_isa_attach() ### #########, ### #### ## ### ######## ### ########, ######### ## ### ######## #####.

############.

```
-/* table of supported bus methods */
static device_method_t xxx_isa_methods[] = {
   -/* list all the bus method functions supported by the driver */
   -/* omit the unsupported methods */
   DEVMETHOD(device_identify,  xxx_isa_identify),
   DEVMETHOD(device_probe,     xxx_isa_probe),
   DEVMETHOD(device_attach,    xxx_isa_attach),
   DEVMETHOD(device_detach,    xxx_isa_detach),
   DEVMETHOD(device_shutdown,  xxx_isa_shutdown),
   DEVMETHOD(device_suspend,   xxx_isa_suspend),
   DEVMETHOD(device_resume,    xxx_isa_resume),

DEVMETHOD_END
-};

static driver_t xxx_isa_driver = {
   -"xxx",
   xxx_isa_methods,
   sizeof(struct xxx_softc),
-};


static devclass_t xxx_devclass;

DRIVER_MODULE(xxx, isa, xxx_isa_driver, xxx_devclass,
   load_function, load_argument);
```

xxx_softc ## # ############### ######### #### ######## ####### ###–### #### ### ########### ### ### ######## #########. ### ### #### ############# ######### ### ##### ########## ### ###### ## ######.

load_function() ## ###### ## ## ############### ############## ## ######## #### ### ###### ## ###### ## ##–

####_######## ## ###### ## ### ## ### #########. ## ### ###### #### ### ####### ####### ####### (## ##### ##### ## #### ###### ## ###### #### ### ######) #### ##### ###### ###### ## ### ## 0 ### ### #### ########## ##### #### ####:

```
DRIVER_MODULE(xxx, isa, xxx_isa_driver,
    xxx_devclass, 0, 0);
```

#######. ### ##### ######## ## # #### ## ### ### ######### ## #### ###### ### ############## ########### ## ### ######## ##### ### ###### ###### ##### ###. ## ##### ####:

```
static struct isa_pnp_id xxx_pnp_ids[] = {
    -/* a line for each supported PnP ID */
    { 0x12345678,   -"Our device model 1234A" -},
    { 0x12345679,   -"Our device model 1234B" -},
    { 0,        NULL -}, -/* end of table */
-};
```

#####, ####:

```
static struct isa_pnp_id xxx_pnp_ids[] = {
    { 0,        NULL -}, -/* end of table */
-};
```

10.3. **device_t**

device_t ## ### ####### #### ### ### ###### #########. #### ## ######## #### ### ####### ########### #### ### ###### ###### ######## ##########. ### ####### ## ########## ###### ## ### ###### ######### ###:

- device_t device_get_parent(dev) ### ### ###### ### ## # ######.
- driver_t device_get_driver(dev) ### ####### ## ### ###### #########.
- char *device_get_name(dev) ### ### ###### ####, #### ## "xxx" ### ### #######.
- int device_get_unit(dev) ### ### #### ###### (##### ### ######## #### 0 ### ### ####### ########## #### #### ######).
- char *device_get_nameunit(dev) ### ### ###### #### ######### ### #### ######, #### ## ####0#, ####1# ### ## ##.
- char *device_get_desc(dev) ### ### ###### ###########. ######## ## ######### ### ##– ### ##### ## ###### ## ############## ####.

############# #### ### ### ##### ##
########### ### ####### ###### ##–
#################

- device_set_desc(dev, desc) ### ### ###########. #### ##### ### ###### ########### ##### ## ### ###### #### ##### ### ### ## ########### ## ####### ##### ####.

- device_set_desc_copy(dev, desc) ### ### ###########. ### ########### ## ###### #### ## ######## ########### ######### ######, ## ### ###### #### ### ## ####### ########## ####### ####### #######.

- void *device_get_softc(dev) ### ####### ## ### ###### ########## (###### xxx_softc) ##–######## #### #### ######.

- u_int32_t device_get_flags(dev) ### ### ##### ######### ### ### ###### ## ### #######–###### ####.

device_printf(dev, fmt, ...) ### ## #### ## ##### ### ######## #### ### ###### ######. ## ############# ######## ### ######## ### ##### ## ### ###–####.

######_# ####### ### ########### ## ### #### kern/bus_subr.c.

10.4. ############# #### ### ### ##### ## ########### ### ####### ######

####:

```
device xxx0 at isa? port 0x300 irq 10 drq 5
      iomem 0xd0000 flags 0x1 sensitive
```

####, ### ### ## ## ### ######### ## ### ######## ###### ########## #### ### ######. #### ### ########, ######### ## ### ######## ##### ### #####–#### ### #################. ### #######, #### ####### ## ### #### ### ## ### ### #### ##### ### ####### ## #### ### ### ####### #### ### ###### ############# #####. ## # ####### ### ######## ### ##### ### ##### ### ### ## ######### ## ### ############# ####, #### isa0 ## isa1, ######### ### ###### ##### ## ######## ### ## ### ### ### #####.

sensitive ## # ######## ########## #### #### ###### #### ## ###### ###### ### #### ######### #######. ## ## ######### ### #### ### #### ## ## #### ## ### ####### ######.

##########. ### #### ###### ## ## ########## ## ### ###### #### #### # ###### ####. ## ### ####### ## #### #### ### ######### ## ### ###### ### ##### ## #### ### ############# #### ### ### ############# ######, ##:

```
device xxx0 at isa?
```

##########.

######, #### ### ### ##### ## #### ######:

```
device xxx at isa?
```

####. ### ############### ####### #### ## ##### ########–#####.

#######:

(######### ### ### ######## ####### ##### ########## ### ### ### #######) ### ###### ## ###### #####. ## #### ######## ### ####### #### ### #### ## ### #### ## ### ### ###. ######## ### ######## ######## ######## ######### ##### ####### #### ### ### #######. ### ### ######## ####### #### ### #### ##### ### ##### ####### ### ## ## #### ### ######### ### #### ### ### ####### ## ####.

#######.

sensitive ### ######. ## ##### ### # ###### #### ############, ### ###### ####### ## ###### ### ##.

########.

#######: #/# ### ###### ####### ######, #### ### ####, ### ## #### ### ###–######## #### ### ######## ###### #######.

######. ### ##### ### #### ###### ### ###### #### ########. ## ## ######## #### ######## ####### ##### ##### ### ###### #### ######### ########; ## #### ####, ### ################ ###### ####. ### ##### ######## #### #### ISA_PNP_PROBE() ## ####### ### ###### ### ## #### ### #### ## ### ### ######### ## ### ###### ### ## ### ## ## ### ## ### ##### ###### #######. #### ##### #### ########## ##### ######, #### ### #### ### ########## ### ### ####### #### #### ISA_PNP_PROBE(), ## ##### #### ## ##### ### ## ##### ## ###### ####### ## ####### ### #######.

(### ##### ####) ## #####, #### ## ######## ##### ## #######.

#####-#####. ### #######, ## ##### ############# ######### ### # ##### ######## ##-####### ##### ### ######### ## ######### #######. #### #### ####### ##### ##-#### ### ######. ### ###### ##### ####### # ###### ##### ## ### ##### ####### ##### ########## (## ##### #####, ### ###### ######### 0 ### ####### ##########, ######### #1 ## ####, ######### #2 ## ##### ## ### ## ##). ## ###### ### ####### ##### ####### #### ### ### ######### #### ## ####### ## ### ### ###### (##### ###### ###### #1 #### ### ##### #######) ##### ### ####### ########## ### ### ######### ## #### #### ## ####### ## ### ### ###### (##### ###### ###### 0 #### ### ##### #######). ## ######## ####### ###### ### #### ##### #### ### ### ###### ##### ####. ## ## # ###### ####### ##### 0 ## ### ## #### #### ## ### ### ######## ###########.

######. #### ### ### ####### ## ### ##### ### ###### ### #### ######, #### ### #### #### ###. #### ####### ## ### ########### ## ### ######## ###### ### ## ### ########## ####### ## #### ########.

####### ### ### ####### ### ######## #### ####### ### ###### ####### #### #### ### ## ######## ##### (#### ## ###### ### #### ## ###). ### ## #### ## #########-####### ######## ######## ## ## ### ############## ## ### ###### ## #### #### #### ### #### ###### #### ### ## ######## ## ### ###### #####: #### ## ###### ############### ### #### ## ###############.

####### ######### ########### ### ### ############### ####### (#### ### ### ################) ## #### ### ##### ### ### ## ###### ## #### #### ### ###### ############# ####. ## #### #### ###### ### ### ### ##### ## ### ## ### ### ##### #### ### ###### #### 0 ### ### ### ############### ####### ## ### ### ###### ######### ####### ## #####.

###-########## ######### ######## #### ######### ## ######## resource_query_*() ### resource_*_value(). ##### ############### ### ####### ## kern/subr_bus.c. ### ### ### #### ###### i386/isa/wd.c ######## ######## ## #### ###. ### ### ######## ##### ## ###-########## #### ###### ## #########. ##### ####### ### ############# ######### ## ### ### ############# ####.

10.5.

#########. #### #######-#### ## ###### ## ### ### ############# #### ### ########### #### # ##### ## ######### ######_# ### ### ### ######### ########## #### ##. ### ####### ### ###### ### ############# ######### ######## ##### ######### resource_* ###

####### ##### ## ##############. #######, ######### #### ## ####### ###### ### ###########, ## #### ##### ## ### ######### ####### ####.

######. #### ### ########## ## #### ### ###### ###### ### ####. ### ### ### ### ### ######### ##### ### #######:

- *SYS_RES_IRQ* # ######### ######
- *SYS_RES_DRQ* # ### ### ####### ######
- *SYS_RES_MEMORY* # ##### ## ###### ###### ###### #### ### ###### ###### #####
- *SYS_RES_IOPORT* # ##### ## ###### #/# #########

0, ## ## # ###### ### ### ###### ####### ## ##### #### ######### ## #### SYS_RES_MEMORY ######## 0 ### 1. ### ######## #### ### ####### ## ## #### ### # ######## ####, ### ### ######## ###### #### ### # ######## #### unsigned long ### #### ## #### ## #########. ### ######## ####### ## ### #### ## ## ##########, ######## ### ### #### ######## ##### ##. ### ######### ######## ####### ### ### ####### ###:

```
IRQ: 0-1
DRQ: 0-1
MEMORY: 0-3
IOPORT: 0-7
```

######, #### # ##### ##### ### #####. ### ### ### ### ######### ### ##### ##### ######## ## ##### ## 1. ### ###### ### ###### ##### ## ### ######## #########.

#########:

- ###/###
- ########/#######
- ########/##########

####### #### ### ##### #### ## ### ########. ########## ######## ### ######### ##### #### ## ##### ###### ##### ## #### ## ####### ## (### ######## #### ## ##### ###### ######## #### ##### #######). ########## ##### ### ######## ########## ## ### ###### ## ##### ######## ## ######### ### #### (### #######, ### ###### ## ##### ## ####### #### ### ###### ####### ####### #####).

###:

- int bus_set_resource(device_t dev, int type, int rid, u_long start, u_long count)

########. ####### 0 ## ##########, ##### #### #########. ###–#####, #### ######## #### ###### ## ##### #### ## ### ## type, rid, start ## count ### # ##### #### ##### ### ## ### ######### #####.

- ### # ######## ######
- #### # #### ## ########, ###_###_#
- ### # ######## ###### (##) ###### ####
- #####, ##### # ######## #####

- int bus_get_resource(device_t dev, int type, int rid, u_long *startp, u_long *countp)

  ### ### ##### ## ########. ####### 0 ## ##########, ##### #### ## ### ######## ## ### ####### ###.

- u_long bus_get_resource_start(device_t dev, int type, int rid) u_long bus_get_resource_count (device_t dev, int type, int rid)

  ########### ######### ## ### #### ### ##### ## #####. ###### 0 ## #### ## #####, ## ## ### ######## ##### ### 0 ##### ### ########## ###### ## ##### ## #####–##### ## #### ## ### ##### ## 0 ## ## ##### ########. #######, ## ### ######### ### ###### ####### ### #### # ##### ##### ##### ## 0.

- void bus_delete_resource(device_t dev, int type, int rid)

  ###### # ########, #### ## #########.

- struct resource * bus_alloc_resource(device_t dev, int type, int *rid, u_long start, u_long end, u_long count, u_int flags)

  ######## # ######## ## # ##### ## ##### ###### ### ######### ## ###### ####, ######### ####### ##### ### ###. ####, ######### ## ### #########. ## ### ######## ### ### ### ### ## ## ############# #######. ### ####### ###–### ## ##### 0 ### ### #0 (### ####) ##### #### ### ##### ###### ########## ### ## bus_set_resource() #### ## #### #######: ##### ### ##### ## ########## ### ###=(#####+#####), ## #### #### ## ### ######## ### ### ####### ###### #### ## ##### ## ########. ######## ### ## ###### ## ######### ## ## ### ### ######## ## ### ######## ########## #### ## ### ### ###. (### ##### ##### ### ### # ###–###### ######## ### ###### ##).

######, ### ##### ########### ### ### ###### ###:

- *RF_ACTIVE* # ###### ### ######## ## ## ############# ######### ##### ####–######.

- *RF_SHAREABLE* # ######### ### ## ###### ## ### #### #### ## ######## #######.
- *RF_TIMESHARE* # ######### ### ## ########### ## ######## #######, #.#., ####–##### ## ### #### #### ## #### ### ######### #### ## ### ## ### ##### ###### ## ####.
- ####### 0 ## #####. ### ######### ###### ### ## ######## #### ### ######## ###### ##### ####### rhand_*().
- int bus_release_resource(device_t dev, int type, int rid, struct resource *r)
- ####### ### ########, # ## ### ###### ######## ## bus_alloc_resource(). ####### 0 ## #######, ##### #### #########.
- int bus_activate_resource(device_t dev, int type, int rid, struct resource *r) int bus_deactivate_resource(device_t dev, int type, int rid, struct resource *r)
- ######## ## ########### ########. ###### 0 ## #######, ##### #### #########. ## ### ######## ## ########### ### ######### ######### ## ####### ###### #### EBUSY ## ########.
- int bus_setup_intr(device_t dev, struct resource *r, int flags, driver_intr_t *handler, void *arg, void **cookiep) int bus_teardown_intr(device_t dev, struct resource *r, void *cookie)
- ######### ## ############ ### ######### ####### #### # ######. ###### 0 ## #######, ##### #### #########.
- # # ### ######### ######## ####### ########## ### ###

  ##### # ### ######### ######## #####, ### ##:

  - INTR_TYPE_TTY # ######### ### ##### ######## ############## #######. ## #### #### ### spltty().
  - (INTR_TYPE_TTY | INTR_TYPE_FAST) # ######## #### ####### #### ##### ##### ######, ######## ## ### #### #### ## ##### (#### ## ### ############# ###### #####). ## #### #### ### spltty().
  - INTR_TYPE_BIO # ########## #######, ###### ##### ## ### ### ###########. ## #### #### ### splbio().
  - INTR_TYPE_CAM # ### (###### ###### ######) ### ###########. ## #### #### ### splcam().
  - INTR_TYPE_NET # ####### ######### ###########. ## #### #### ### splimp().
  - INTR_TYPE_MISC # ############# #######. ##### ## ## ##### ### ## #### #### #### ## splhigh() ##### ##### ### ##########.

###–##### ##### #### ## ######. ### #### ######### ## ### #### ##### ### ##### ## ##### ########## ### ###### ### ##### ## ### ###### ## ### ##### #########.

- *handler* # ####### ## ### ####### ########, ### #### ######_####_# ## ####### ## void driver_intr_t(void *)
- *arg* # ### ######## ###### ## ### ####### ## ######## #### ########## ######. ## ## #### #### ##### ## ### #### #### ## ### #######. ### ### ########## ### ### ### ######### ######## ### ## ### ### #### ###### ## ########, ### ### (###########) ########## ## ##### # ####### ## ### ###### ##### #########.
- *cookie[p]* # ### ##### ######## #### setup() ## #### ## ######## ### ####### #### ###### ## teardown()

(###### ##–###### #). ##### ## ######## ## ### ###### ###### ####### ###:

- u_long rman_get_start(r) u_long rman_get_end(r) ### ### ##### ### ### ## ######### ######## #####.
- void *rman_get_virtual(r) ### ### ####### ####### ## ######### ###### ########.

10.6. ### ######

######. ### ######## ### ########:

(#) ###### ## ####### ## ### ###### ####

(#) ###### ## ### #### ###### ## ### ########

(#) ### ###### ###### ###### ### #### #### ### ##### ####### ### ####### ###### ### ### #### ###### ## #########. ## ### ### ####### ###### #### ### ###### ####### ####### ##### ### ######## ####### ### ###### ## ### ####### ###### #### ## ####### ## # SYS_RES_MEMORY ########. #### ######## ### #### ## ######### ### #########, ### ### ####### ####### ######## ##### rman_get_virtual(). ### ##### ####### #### ### ######## pmap_mapdev() ### #### #######, ##### ###### ### ## #### ######## ### ####. ### ## ## ### ## ### ######## ##### ## ######## ##########.

640###1##. #### ## ### ### ##### ####### ###### ###### ###### ##### ###### ## ###### ######### ##### 16## (####### ## ### 24#### ####### ###–####### ## ### ### ###). ## #### #### ## ### ####### ### #### ###### #### ### ##### ####### ## ### ###### ###### (## ##### #####, #### #######) # ###### ####

#######. #### ###### ##### #######–###### ## # ###### #### ## 1## ######## ## 14## ## 15##. ####### ### ###### ### ###### ##### ######## ## ### #### ####### #### ######## (#### ####### ### ## ###### ## ### ######).

(#) #### ### ####### ## ### #### ## #### ## ### ######, ### ### ###### #### ### ## ######## ###### ### #### ## ### #### ######. ### ########### ### #######: #####, ### ##### ### #### ###### ###### ##### 16##. ######, ### ########## ##### ## ####### ####### ##### ### ### ## ########## ## ######## ####### #####, ## ### ###### ### #### ## ## #######/###### ##########. ### ### ######### ######## ##### ######### ### #### ## ##### ########, ### #### ### ## ## #### ## ### ####### ##########.

##########, bus_dma_tag_t ### bus_dmamap_t. ### ######### ### ########## ######## ### ### ### ######. ### ########## # ###### ##### ######### ######### ## ##### ##########. ######## #### ### ## ########## #### ### #### ###.

######–####. # ##### ### ######## ### ### ############ ## ### ###### ###, ### ### #### #### #### ###### ### ##### #### #####.

######## ### ######### ### (#### ## ######) ## ####### ### #### ###### ####. ## ######## ###### ##### #### ######### ############ ### ###### ### #### ###### #### # ### ### #### ## #### ### ## ####### ## # ##### ## ### ###### ###.

####.

#####, # ##### ## ########### ###### ########## #### ### ### ############ ### ## ######### (### ##### ### ## #####). #### ## ######## #### ## ######## ########## ########### ##### ## ###### ### ############# #### ### ######. ####### ## #### ###### #### # ### ## #######: ## ## ###### ########## ## ### ##### ## ### #####–###### ######## ###### #####.

######, ## ######### #### ## ####### ###### ### ## ###### #### # ###. #### #### ## #### ###### #### ## ####### ### ########### ## ### ### ###########. ## ## ######## #### ## ## #### ## ### ######## ########. ## ## ## ### #### # ##### ########## ####### ##### ## ######### ### #### ## ############ #######. #### ####### ### #### #### ### ############## ######## ##### #### #### ## ###### ## ##### ###### ##### ##### ### #### ########### #### ### ###### ##### ## ### ##–####. #### ####### ### #### ##### ## ##### ### ###### ## ### ###### ##### ### #### ###### ## ##### ############## ######## #####. ### ####### ## ####### ####### ### ######## ### ###### ##### ## ###### ###############. #### ## ######## #### ## # ############ #####: ###### ### #### ######## ##### ## ######, ######## #### ### ###### ########.

###:

- int bus_dma_tag_create(bus_dma_tag_t parent, bus_size_t alignment, bus_size_t boundary, bus_addr_t lowaddr, bus_addr_t highaddr, bus_dma_filter_t *filter, void *filterarg, bus_size_t maxsize, int nsegments, bus_size_t maxsegsz, int flags, bus_dma_tag_t *dmat)

  ###### # ### ###. ####### 0 ## #######, ### ##### #### #########.

  - *parent* # ###### ###, ## #### ## ###### # ######### ###.

  - *alignment* # ######## ######## ######### ## ### ###### #### ## ## ######### ### #### ###. ### ##### 1 ### ### ######## ##########. ####### #### ## ### ###### bus_dmamem_alloc() ### ### bus_dmamap_create() #####.

  - *boundary* # ######## ####### ######## #### #### ### ## ####### #### ####–###### ### ######. ### ##### 0 ### ### #########. ####### #### ## ### ###### bus_dmamem_alloc() ### ### bus_dmamap_create() #####. #### ## ##### ## 2. ## ### ###–### ## ####### ## ## #### ## ############ ### #### (#.#., ### ### ######### #### ## ######## ### ## ### ###### ###### ### ## ### ### ### ##########) #### ### ######## #### ## ## ###### #### 64## (64#1024) ### ## ### ########### ## ### ### ########.

  - *lowaddr, highaddr* # ### ##### ### ######## ##########; ##### ###### ### #### ## ##### ### ######### ##### ## ######## ######### #### ## ######## ### ######. ### ##### ####### ###### ######### ## ### ####### ###### ###:

    - ### bus_dmamem_alloc() ### ### ######### #### 0 ## ########1 ### ######### #########, ### ###### #### ### #########.

    - ### bus_dmamap_create() ### ### ######### ####### ### ######### ##### [#####–##; ########] ### ######### ##########. ### ######### ## ##### ###### ### ##### ### ###### ## ### ###### ######## ##### ####### ## #### ### #####–#####. ## ## ###### ######## ## ######## #### ### ### ##### ## ######### ############.

    - ### ### ### ####### ### ###### ###### (#### ## ###### ########) ###:

      ####### = ###_#####_#######_24###

      ######## = ###_#####_#######

  - *filter, filterarg* # ### ###### ######## ### ### ########. ## #### ## ###### ### ###### #### ### ##### ##### [#######, ########] ## ########## #########–### #### ##### bus_dmamap_create(). ######### ### ######## ####### ## #### ##–####### #### ## ##### [#######; ########] ## ###### ## ### ###### ######## ##### ####### ## ## ## ##########. ### ######### ## ### ###### ######## ##:

int filterfunc(void *arg, bus_addr_t paddr). ## #### ###### 0 ## ### #### ## ##########, #### #### #########.

- *maxsize* # ### ####### #### ## ###### (## #####) #### ### ## ######### ####### #### ###. ## #### ## ## ######### ## ######## ## ##### ## ########### ###, ### ##### ### ### ####### ##### ## BUS_SPACE_MAXSIZE_24BIT.

- *nsegments* # ####### ###### ## ############## ######## ######### ## ### ##–####. ## ############ #### ### ##### BUS_SPACE_UNRESTRICTED ###### ## ####. #### ##### ## ########### ### ### ###### ####, ### ###### ############ ##### #### ## ######### ### ### ########## ####. #### #### ######### ##### ## BUS_SPACE_UNRESTRICTED ### ### ## #### ## ######## #### ####, #### ### ## #### #### ## ###### ####. ### ######### ##### ### ######### ##### ## ## ##### 250#300, ###### ###### #### ##### ###### ##### ######## (### ######## ### ### ######## ####### #### #### ############## ####### ######).

- *maxsegsz* # ####### #### ## # ############## ####### ######### ## ### ######. ### ####### ##### ### ### ###### ##### ## BUS_SPACE_MAXSIZE_24BIT.

- *flags* # # ###### ## #####. ### #### ########### ##### ###:

  - *BUS_DMA_ALLOCNOW* # ######## ## ######## ### ### ########### ###### ###### ##### #### ######## ### ###.

  - *BUS_DMA_ISA* # ########## #### #### #### ## ##### ########. ## ## ### ####### ### ### #386 ########. ######## ## ###### ## #### ## ### ### ### ####### ### ##### ######## ### ## ##### #### ##### ### ## #### ####### ###.

- *dmat* # ####### ## ### ####### ### ### ### ### ## ## ########.

- int bus_dma_tag_destroy(bus_dma_tag_t dmat)

  ####### # ###. ####### 0 ## #######, ### ##### #### #########.

  #### # ### ### ## ## #########.

- int bus_dmamem_alloc(bus_dma_tag_t dmat, void** vaddr, int flags, bus_dmamap_t *mapp)

  ######## ## #### ## ########## ###### ######### ## ### ###. ### #### ## ###### ## ## ######### ## ##### #######. ####### 0 ## #######, ### ##### #### #####–####. ### ###### ##### ### ## ## ###### ## bus_dmamap_load() ###### ##### #### ## ### ### ######## ####### ## ### ######.

  - *dmat* # ### ###

  - *vaddr* # ####### ## ### ####### ### ### ###### ####### ####### ## ### ####–##### #### ## ## ########.

- ##### # # ###### ## #####. ### #### ########### #### ##:
  - *BUS_DMA_NOWAIT* # ## ### ###### ## ### ########### ######### ###### ### #####. ## #### #### ## ### ### #### ### ####### ## ####### ## ##### ##### ### ###### ####### #########.
- *mapp* # ####### ## ### ####### ### ### ### ### ## ## ########.

- void bus_dmamem_free(bus_dma_tag_t dmat, void *vaddr, bus_dmamap_t map)

  #### ### ###### ######### ## bus_dmamem_alloc(). ## #######, ####### ## ### ###–### ######### #### ### ############ ## ### ############. ####### ## #### ### ########### ##### ## ### ## ## #### ### ###### ### ######### ##### ### ## #### ## ########. ## ### ####### #### #### #### ### #### ####### ######## ## #####. #### #### ### #### #### bus_dmamem_free() ###### ### ## #### ## ###: ######### ## #### ## ######## ########### ####.

  - *dmat* # ### ###
  - *vaddr* # ### ###### ####### ####### ## ### ######
  - *map* # ### ### ## ### ###### (## ######### #### bus_dmamem_alloc())

- int bus_dmamap_create(bus_dma_tag_t dmat, int flags, bus_dmamap_t *mapp)

  ###### # ### ### ### ###, ## ## #### ## bus_dmamap_load() #####. ####### 0 ## ###–####, ### ##### #### #########.

  - *dmat* # ### ###
  - *flags* # #############, # ### ### ## #####. ### ## ##### ### ####### ###, ## ## ####### ## #### ## ###### 0.
  - *mapp* # ####### ## ### ####### ### ### ### ### ## ## ########

- int bus_dmamap_destroy(bus_dma_tag_t dmat, bus_dmamap_t map)

  ####### # ###. ####### 0 ## #######, ### ##### #### #########.

  - #### # ### ### ## ##### ### ### ## ##########
  - ### # ### ### ## ## #########

- int bus_dmamap_load(bus_dma_tag_t dmat, bus_dmamap_t map, void *buf, bus_size_t buflen, bus_dmamap_callback_t *callback, void *callback_arg, int flags)

  #### # ###### #### ### ### (### ### #### ## ########## ####### ## bus_dmamap_create() ## bus_dmamem_alloc()). ### ### ##### ## ### ###### ### ####### ###

########### ## ### ### ############# ### ### ##### ### ########### ### ###### ##### ### ##########. ## ##### ## ######## ####### ########### ## ##### ### ###### ## ### ######## #######. #### ######## ####### ## #### ######## ## ###–### ## ## #### ###. ### ###### ## ###### ####### ## ### ###### ## #######, ## ## ### ###### ####### ### ###### ### ### ############ ######### ### ####### #### ## ###### ### ### ######## #### ## ###### #### ### ###### ####### #### ###### #########. ####### 0 ## ### ######## ### ######## ########### ## ###–######## ## ### ####### ### ###### ### ###### #########. ## ### ###### #### ### ############### #### ###### ######## ####### ## ### ############## ## ### ######.

- *dmat* # ### ###
- *map* # ### ###
- *buf* # ###### ####### ####### ## ### ######
- *buflen* # ###### ## ### ######
- *callback*, callback_arg # ### ######## ######## ### ### ########

##:

void callback(void *arg, bus_dma_segment_t *seg, int nseg, int error)

- *arg* # ### #### ## ########_### ###### ## bus_dmamap_load()
- *seg* # ##### ## ### ####### ###########
- *nseg* # ###### ## ########### ## #####
- *error* # ########## ## ### ####### ###### ########: ## ## ## ### ## ##### #### ### ###### ### ### ### #### ### ####### ###### ## ######## ######### ## ### ###. ## #### #### #### ### ######### ###### ## ########### #### ## ## ### #####. ######## ## #### ######### ## ## ## ### ######: ######### ## ### ##–##### ######### ## ### ###### ######## #### ## ##### ## ##### ### ###### ## ### ### ###### ### ###### #### ##########

######:

- *ds_addr* # ######## ### ####### ## ### #######
- *ds_len* # ###### ## ### #######

- void bus_dmamap_unload(bus_dma_tag_t dmat, bus_dmamap_t map)

  ###### ### ###.

- *dmat* # ###
- *map* # ###### ###

- void bus_dmamap_sync (bus_dma_tag_t dmat, bus_dmamap_t map, bus_dmasync_op_t op)

  ########### # ###### ###### #### ### ###### ##### ###### ### ##### #####–### ######## ## ## #### ######. #### ## ### ######## #### #### ### ### #####–#### ####### ## #### ####### ### ######## ###### ### ### ###### #######. ### ####### #### ## ############ #### ####### ### ##### ##### ### ########.

  - *dmat* # ###
  - *map* # ###### ###
  - *op* # #### ## ############### ######### ## #######:
  - BUS_DMASYNC_PREREAD # ###### ####### #### ###### #### ######
  - BUS_DMASYNC_POSTREAD # ##### ####### #### ###### #### ######
  - BUS_DMASYNC_PREWRITE # ###### ####### ### ###### ## ######
  - BUS_DMASYNC_POSTWRITE # ##### ####### ### ###### ## ######

######, ## #### #### ### ## ####### ## ### ######. ############### ## ### ###### ### ### ###### ######## #### bus_dmamem_alloc().

bus_dmamap_load() ### ####### ##### ## ###### ## ### #####. ### ## #### ############# ### ### ####### ###### ## ######## ####### ## ### ###. ####### ## ##### ### ######### ##### ### ### ###### ## ######## ## #386 ############ ## ##### 250#300 (### ###### ##### ## 4## ##### ### #### ## ### #### #########, #### ## # ####### ##### ##### ## 8 #####, ### #### ##### #### ## ####). ####### ### ##### ## ######### ##### ## ### ####### ###### #### ##### #### ### ## ### ###### #### ###### ######. ###########, ### #### ## ######## ### ####### ######### ###### ## ######## ## #### #####. ### ## ### ###### ##### ## ###### ####### #### # #### ##### ###### ## ############## ######## ## ###### ## #### ## ########: #### #### ## ### ######, ######## ## ## ### ######, #### #### #### ## ### ######, ### ## ##.

####### ######### ########### ## #### ### ###### ## ######## ### ##### ### #### ## ### ######. ## ### ### ##### ## ### ###### ###### ## ## ########## #######–####### #### ### ####### ######### ###### #### ### #### ########## #### ##### ## (######### # ####_####). ### #######, ## # ####### ###### ## 10 ######## ## ######### #### ## #386 ####### ########## ######### ###### #### ##### ## 40#. ## # ###### #### ## ####### #### ####### ###### ###### ## #### ## ### ######.

###.

###. ### ########## #> ### #### ## #### ### #### ## ####.

######– #### ### ########## ## # ######:

###_######_##### #> ###_######_#### #> ...### ######... #> #> ###_######_###### #> ###_######_####

######:

```
bus_dmamap_create -->
--> bus_dmamap_load --> bus_dmamap_sync(PRE...) --> do transfer -->
--> bus_dmamap_sync(POST...) --> bus_dmamap_unload -->
-...
--> bus_dmamap_load --> bus_dmamap_sync(PRE...) --> do transfer -->
--> bus_dmamap_sync(POST...) --> bus_dmamap_unload -->
--> bus_dmamap_destroy
```

bus_dmamem_alloc() ### ###### ####### ### #### ## ### ###### #### ## ### #### ## #### ## bus_dmamem_alloc(). ## #### #### ## ## ########## #### ### ##### ###### #### ## ###### ## ### ####### (## ### ######## ### ## ##### ## #### ##########) ### ### ####### #### ## ######## ########### (########### #### ##### ## ########). ### ### ######## ##### ## ## ## #### #### ## ## #### ### ######## #######.

##:

```
 static void
alloc_callback(void *arg, bus_dma_segment_t *seg, int nseg, int error)
{
 *(bus_addr_t *)arg = seg[0].ds_addr;
-}

 -...
 int error;
 struct somedata {
   -....
 -};
 struct somedata *vsomedata; -/* virtual address */
 bus_addr_t psomedata; -/* physical bus-relative address */
 bus_dma_tag_t tag_somedata;
 bus_dmamap_t map_somedata;
 -...

 error=bus_dma_tag_create(parent_tag, alignment,
```

```
 boundary, lowaddr, highaddr, -/*filter*/ NULL, -/*filterarg*/ NULL,
 -/*maxsize*/ sizeof(struct somedata), -/*nsegments*/ 1,
 -/*maxsegsz*/ sizeof(struct somedata), -/*flags*/ 0,
 &tag_somedata);
if(error)
return error;

error = bus_dmamem_alloc(tag_somedata, &vsomedata, -/* flags*/ 0,
   &map_somedata);
if(error)
   return error;

bus_dmamap_load(tag_somedata, map_somedata, (void *)vsomedata,
   sizeof (struct somedata), alloc_callback,
   (void *) &psomedata, -/*flags*/0);
```

##. ### ######### ###–######## ##: ## ######## ###### ##### ### ######### ###### ######## ## ##### ## # ###### #### #### ## ####### #### ### #### ### ######### ### ######## ## ### (## ### ######### ### ######## ########### ######).

bus_dmamap_create() ####### ######## #### ## ##### ## ########### #### ### ######## ## #### ## ##### ## ##–#####. ### #### ##### #### ####:

```
{
 int s;
 int error;

 s = splsoftvm();
 error = bus_dmamap_load(
     dmat,
     dmamap,
     buffer_ptr,
     buffer_len,
     callback,
     -/*callback_arg*/ buffer_descriptor,
     -/*flags*/0);
 if (error == EINPROGRESS) {
     -/*
      * Do whatever is needed to ensure synchronization
      * with callback. Callback is guaranteed not to be started
      * until we do splx() or tsleep().
      */
    -}
 splx(s);
-}
```

###:

1. ## ######## ### ######### ## ####### #### ########## ## #### (#### ## ### ### ########) #### ## ##### ## ####### ## ### ### ### ####### ########## #### ###

######## ###### ##### ##### #### ### ####### #### ## ## ####. #### ### #### ##### ############### ## ######. ### ### #### ####### ####### ## ### ## # #### #### ## ###### ### ####### ##### ##### #### ####### #### #########.

2. ## ######## ### ######### #### ### ######## ####### (#### ## ####### #### ## ##### ######## ## ######### #######) #### # ############### #### ###### ## ### ## ### ###### ########## ### tsleep() ######. ##### #### ### ######## #### ###### ## #### ## ### ########## ### ##### #### ############### ####. ## ## ## ### #### ### ######## ###### ##### # ######. ## #### ######## ### ######## ######## ##### ###### ## ### ### ###### ########## (#### #### ### ######## ####) ## ###### #### ### ######## ##### ## ### ###### ##########. #### ##### ######## ######### ### ####### ######## ##### ### #### ##### ######## ##### ### ## ### ### ##########.

10.7.

(###) ## ########### ## ### ### ### ####### ### ### ###–####### (########, ### ## #### ### #### ## ## ########## ######). ## #### ### ##### ### ####### ###### ### ##### ### ##### ## ### ### ####### ### ####### ########## ### ############ ## ### ### ###########. ###########, ####### ######## # ### ## ######### #### ###### #### ### ######## ####### ## ### ### ########## #### ### ###### #######.

#######. #### ### ### ###### ##–##### ## ### #### ### ######## ### ### ###### ### ###### ######### ### ## ####–######. ### #### ##### #### ###### #### #### ### ### ########## ## ### #######–####. ## ### #### ####### #### ####### ## ## ######## ##### ### ###########. ### ### #### ##### ###### #### ### ###### ### ########## ## ## ###### ### ######–## #### ## # ### ####### ## ####### ### ######### ######## #### ######### ### ######:

void isa_dmacascade(int channel_number)

######. #### ######### ### ###### ## ########### ######### #### ## ## ######.

###########. ### ######### #### ###:

- int isa_dma_acquire(int chanel_number)

  ####### # ### #######. ####### 0 ## ####### ## ##### ## ### ####### ### ####### ######## ## #### ## # ######### ######. #### ## ### ### ####### ### ### #### ## ##### ### ######## ######, ## ######## #### ######## ## ###### #### ######### # ######. #### ########### ### #### ######### ## ### ###### ######### ## ### ######### ### ##### #### ## #### ## ######## ## ### ######. ## ### #### #### #####, ##### ### ######## #### #####.

- int isa_dma_release(int chanel_number)

  ####### # ########## ######## ### #######. ## ######### #### ## ## ######## #### ### ####### ## ######## (## ######## ### ###### #### ### ### ## ######## ######## ##### ### ####### ## ########).

- void isa_dmainit(int chan, u_int bouncebufsize)

  ######## # ###### ###### ### ### #### ### ######### #######. ### ######### #### ## ### ###### ### ### ###### 64##. #### ###### ###### #### ## ########–##### #### ##### ## # ######## ###### ####### ## ## ### ########## #######–### ## ####### ## ### ###### ########## ## ### ### ### ## ######## ### 64## ########. ## ### ######### #### ## ###### #### #### ####### ##### ####### ## ##### ########## (#### ## ##### ######### ## bus_dmamem_alloc() #### ###### ###–########) #### isa_dmainit() #### ### #### ## ## ######. ### ## ## ##### ########## ## ######## ######### #### ##### ### ### ##########. ### ###### ###### #### ############# #### ## ### ############## ######.

  - *chan* # ####### ######
  - *bouncebufsize* # #### ## ### ###### ###### ## #####

- void isa_dmastart(int flags, caddr_t addr, u_int nbytes, int chan)

  ####### ## ##### # ### #########. #### ######## #### ## ###### ## ### ## ### ### ########## ###### ######## ######## ######## ## ### ######. ## ###### #### ### ###### ## ########## ### ##### #### ### ### ###### #####, ## ### #### ### ###### ###### ## ############# ####. ## ###### ###### ## ######## ### ### ### ## ## isa_dmainit() ## ### ##### ### ### ######### ######## #### #### ### ###### #### #####. ## #### ## # ##### ####### #### ###### ###### ### #### #### ## ##–########### ###### ## ### ###### ######.

- ##### # # ####### ########### ### #### ## ######### ## ## ####. ### ######### #### #_#### ### #_##### ### ######## #########.

  - #_#### # #### #### ### ### ### #### ######
  - #_##### # ##### #### ### ###### ## ### ### ###
  - #_### # ## ### #### ### ### ########## #### ######## ### ###### ### ##### ### ### ## ######## #### ############# ############# ###### ## ###### ######## ## ### #### ###### ##### (## ######, ### ###### ### ###### ### #### ## ### ###### ###### ########## ####### ######## ## ### ######). ## ### ### #### ### ########## #### #### #### ### ### ########, ### isa_dmastart() #### #### ## ## ###### ##### ###### ########## ### #### ########. ##### #_### ##### ##### #### ## ### ###### ###### ## ### ####.

- #### # ####### ####### ## ### ######
- ###### # ###### ## ### ######. #### ## #### ## ##### ## 64##. ###### ## 0 ## ### #######: ### ### ########## #### ########## ## ## 64## ##### ### ###### #### #### ########## ## ## 0 ### #### ##### ##### ############# #######. ### ######## ###### 4 ### ###### ### ###### #### ## #### ####### ##### ######## ######## 2 ##### ## # ####. ## #### ## ## ### ###### ### #### #### #### ### ## ###########.
- #### # ####### ######
- void isa_dmadone(int flags, caddr_t addr, int nbytes, int chan)

  ########### ### ###### ##### ###### ####### #### ######## ## ####. ## #### ### # #### ######### #### # ###### ###### #### ### #### #### ## ###### #### ### ###### ###### ## ### ######## ######. ######### ### ### #### ## ### isa_dmastart(). #### #_### ## ######### ### ## #### ### ###### isa_dmadone() ## ### ###.

- int isa_dmastatus(int channel_number)

  ####### ### ###### ## ##### #### ## ### ####### ######## ## ## ###########. ## #### ### #### #_#### ### ### ## isa_dmastart() ### ###### ######## #### ##### ## ##### ## ####. ## ### ### ## ######## ## #### ## ############# ##### #### ## ### ###### ## ######. ### ###### ### ## ## ##### ### ###### ## ##### #### ##### ### ###### ####### #### ### ######## ## #########. ## ### ###### ## ##### ## ### 0 #### ######### ######## #### ##### #### #### ########.

- int isa_dmastop(int channel_number)

  ###### ### ####### ######## ### ####### ### ###### ## ##### #### #######–######.

10.8. ###_###_#####

#######. ## ### ###### ######## ##########–#### ## #### #### ## ###### ############# (#### ## ######### ###### ## ###### #######) #### ############## #### ## #### ## #### #######.

###, ## ### ###### ###### ## ######## ## ## ######## ### ###### ### ######### ## #### ##### ####### ######## #### ## ####### # ######## ##### ## #####. ### ##### ##### #### ## ######## ## ### ###### ## ### #######. ##### ##### ###### ### #### ##### ##########. #### ## ######## ###### #### #######. #### ## ### ####### ###### #### ## #######.

#####–#####. ### #######, ## ##### ############# ######### ### # ##### ######## ##–

####### ##### ### ######### ## ######### #######. #### #### ####### ##### ##–#### ### ######. ### ###### ##### ####### # ###### ##### ## ### ##### ####### ##### ########## (## ##### #####, ### ###### ######### 0 ### ####### ##########, ### ######### #1 ## ####, ### ######### #2 ## ##### ## ### ## ##). ## ###### ### ####### ##### ####### #### ### ### ######### #### ## ####### ## ### ### ###### (##### ###### ###### #1 #### ### ##### #######) ##### ### ####### ########## ### ### ######### ## #### #### ## ####### ## ### ### ###### (##### ###### ###### 0 #### ### ##### #######).

###_##### ## ######### ## ### ###### ###### ####–### ### ##### #######. ## ### ##### ####### ####### ## ##### ### ########## #### ## ############# ########### ## ### ######. ## ## # ####### ##### ###### ### ###### #### #### #### #### ### ### ######### ## #### ###### ##### ### ######–##### ### #### ####### ##### ### ########## #### ##### ###### ###########. ## ### ##### ######### ############ ### ########## #### ## ######### ## ### ###–### ### ##### ###### ## ### ####### xxx_isa_attach(). ## # ###### ####### # ######## ##### ## ### ### ## #### #### ## #### #### ### ####### ######## ### ### ###### ####### #### ## ######. ## ## #### #### ## #### #### ####### ### ### ######### ###### ######### ### ## ######### ######## #### ##### ## ### ###### #######. #### xxx_isa_probe() ####### 0 ######### ### ######### ###### ######### ## #### # #### #### ### # ############ ###### ###### ## ##. ### ## ##### ##### ##### ## #### ####### #### ######### ### ######### ### ###### ## ####### ## #### ######### ####### ######### 0 #### ### ##### ####### ### ######### ## ### ###### #######.

####:

```
struct xxx_softc *sc = device_get_softc(dev);
 int unit = device_get_unit(dev);
 int pnperror;
 int error = 0;

 sc->dev = dev; -/* link it back */
 sc->unit = unit;
```

#######. ### ##### ## ####### ### ## # ##### ########## ### #### ## ### ### ######### ## #### ###### ### ############## ############ ## ### ###### ###### ############# ## ##### ###.

```
pnperror=ISA_PNP_PROBE(device_get_parent(dev), dev,
xxx_pnp_ids); if(pnperror == ENXIO) return ENXIO;
```

###_###_##### ## ### #########: ## #### #### (###### ####) ### ### ######## ## ### #### ###### #### ## ########. ## ## ### ######## ## ### ### ### ######## ## #### ### ##### ### ## ### ### ## ### ##### #### ##### ## ########. #######, ## ## ### ### ####### ### ## ####### ## ## ### ### ## ### #####, 0 ## ######## ### ### ########### ########### #### ### ##### ## ### ## device_set_desc().

####:

```
if(pnperror -!= 0)
   return pnperror;
```

##–##### #### #### ## ##### ### ## ##### ### #### ###### ### ##### ## ###### ## # ### ####.

#########, #### ## #/# #### ###### ## #### ### #### ### ##### ##. ######### ## ### ######## ### ###### ### ## #### ## ######## ### ##### ######### ######### #############. ### ### ####### #### ### ### ######### ####### ## ### ### #########, ## ### ###### #### ### #### ## ######## #### ## ######.

######### ### ####### ########### ######## ## ### ###### ## ### ###### ## ### #/# #### ######. #### #### ####### ##### ## ### ### #### ## ########### #### ### ###### ############# ######### (###### ### ### ####### ## ####). ## ##### ## ### ## ### ### #### ##### #####:

```
sc->port0 = bus_get_resource_start(dev,
      SYS_RES_IOPORT, 0 -/*rid*/); if(sc->port0 == 0) return ENXIO;
```

###. ## ## #### ## #### #### ##### #### ####### ### ######## ######## #### #### ##### ## ######–####### ####. ## ## ## ### ### # #### ## #### ###### ## #####. #### ###### ####### ### ####### ## ###### ### ### ## ##### ### ### ######## #####, #### ####:

```
-/* table of all possible base I/O port addresses for this device */
static struct xxx_allports {
   u_short port; -/* port address */
   short used; -/* flag: if this port is already used by some unit */
-} xxx_allports = {
   { 0x300, 0 -},
   { 0x320, 0 -},
   { 0x340, 0 -},
   { 0, 0 -} -/* end of table */
-};

-...
int port, i;
-...

port =  bus_get_resource_start(dev, SYS_RES_IOPORT, 0 -/*rid*/);
if(port -!=0 -) {
   for(i=0; xxx_allports[i].port!=0; i++) {
      if(xxx_allports[i].used -|| xxx_allports[i].port -!= port)
         continue;
```

```
        -/* found it */
        xxx_allports[i].used = 1;
        -/* do probe on a known port */
        return xxx_really_probe(dev, port);
    -}
    return ENXIO; -/* port is unknown or already used */
-}

-/* we get here only if we need to guess the port */
for(i=0; xxx_allports[i].port!=0; i++) {
    if(xxx_allports[i].used)
        continue;

    -/* mark as used -- even if we find nothing at this port
     * at least we won't probe it in future
     */
     xxx_allports[i].used = 1;

    error = xxx_really_probe(dev, xxx_allports[i].port);
    if(error == 0) -/* found a device at that port */
        return 0;
-}
-/* probed all possible addresses, none worked */
return ENXIO;
```

######, ######## ### ######## identify() ####### ###### ## #### ### #### ######. ### ##### ### ## ### ##### ###### ### ## ### ## ###### ## ## #### ## probe(): ## #### ##### ##### ##### #### ##### ######### ###### #####. ### ##### ######## ### ####### #### ############# ## ### sensitive ####: ### ######### ####### ### ###### ##### ### ### #### ## ### ####### #####. ### ### identify() ######## ### ###### ###### ### ######, ## #### #### ## ####### ## ### ######### ####### ### ### ##### ####.

###, ##### ## ### ### ######## #### ## #### ## ### ### #### ##### (###### ### ### #######) ####### ### ###### #### ### #### #### ########### ## ### ############# ####.

```
if(pnperror -/* only for non-PnP devices */
&& bus_set_resource(dev, SYS_RES_IOPORT, 0, sc->port0,
XXX_PORT_COUNT)<0)
    return ENXIO;
```

####### ######## ### ######## # ##### ## #### ####### ##### (####### ###### ## ##### ### ### #### #### ##### ## ### ## bus_set_resource()#):

```
sc->port0_rid = 0;
sc->port0_r = bus_alloc_resource(dev, SYS_RES_IOPORT,
&sc->port0_rid,
    -/*start*/ 0, -/*end*/ ~0, -/*count*/ 0, RF_ACTIVE);

if(sc->port0_r == NULL)
```

```
return ENXIO;
```

##. ## ## #### ### #### ##### ## ######## #### ##### ###### ## ## ###### ## ### ## #### #######.

######## ####### ## ### ### ## ### ######### ######## ##### ### ###### #######. ####### #### ## ###### ## ### ####### #### ##### ### DELAY() ######## ### ####–###. ### ##### ####### #### ##### #### #######, ### ### ##### ### ### ###### #### ## #### #### ########. ## ### ###### #### ### ####### ###### ### #### ## ## ####–#### ###### ## ############# ### ### ###### #### ###### #####. #### ########### ### ####### ######## #### ### ###### #### ##### #### ## ## ## ### #### ####: ##–###### DELAY() ## ######## ## ##### ### ### #### ###### ## #### ## ### ####### ## ### #### ###### ## #####, ######### ## ### ##### ###.

###. ### #### ## ### ###########.

```
-/* implemented in some very device-specific way */
if(error = xxx_probe_ports(sc))
   goto bad; -/* will deallocate the resources before returning */
```

xxx_probe_ports() ### #### ### ### ###### ########### ######### ## ### ##### ##### ## ###### ## #########. ### ## ##### ## #### ### ######### ###### ##### #### ### ## ## #### #### ## # ######### ###. ## ######, ### ### ### ####### ### ### ####### #### ### ########### #### ### ##### #############.

```
if(pnperror)
   device_set_desc(dev, -"Our device model 1234");
```

##–######## ## ### ####. ## #### ######## ## #### ## ####### ## ######## ######. ### ##### ####### ###### ## ## ############# ## ########, ## ########## ### ##### ## ############# ## ### #### ## ######### (####### ### #####) ##### ## ###### #### ## ### ###### #######.

#########. ## #### ####### ### ######### ### #########, ##### ### ###### ## ####? ######## ## ### #### ######## ## ### ### ####### ########## ## ### ###### ############# #### #### #### #### #### ### ##### ### #### ### ###### #### ##########. ## ####### ## ############## ##### ##:

```
-/* try to find out the config address first */
```

```
	sc->mem0_p = bus_get_resource_start(dev, SYS_RES_MEMORY, 0 -/*rid*/);
	if(sc->mem0_p == 0) { -/* nope, not specified by user */
	    sc->mem0_p = xxx_read_mem0_from_device_config(sc);


	if(sc->mem0_p == 0)
	        -/* can't get it from device config registers either */
	        goto bad;
	-} else {
	    if(xxx_set_mem0_address_on_device(sc) < 0)
	        goto bad; -/* device does not support that address */
	-}

	-/* just like the port, set the memory size,
	 * for some devices the memory size would not be constant
	 * but should be read from the device configuration registers instead
	 * to accommodate different models of devices. Another option would
	 * be to let the user set the memory size as -"msize" configuration
	 * resource which will be automatically handled by the ISA bus.
	 */
	 if(pnperror) { -/* only for non-PnP devices */
	    sc->mem0_size = bus_get_resource_count(dev, SYS_RES_MEMORY, 0 -/*rid*/);
	    if(sc->mem0_size == 0) -/* not specified by user */
	        sc->mem0_size = xxx_read_mem0_size_from_device_config(sc);

	    if(sc->mem0_size == 0) {
	        -/* suppose this is a very old model of device without
	         * auto-configuration features and the user gave no preference,
	         * so assume the minimalistic case
	         * (of course, the real value will vary with the driver)
	         */
	        sc->mem0_size = 8*1024;
	    -}

	    if(xxx_set_mem0_size_on_device(sc) < 0)
	        goto bad; -/* device does not support that size */

	    if(bus_set_resource(dev, SYS_RES_MEMORY, -/*rid*/0,
	            sc->mem0_p, sc->mem0_size)<0)
	        goto bad;
	-} else {
	    sc->mem0_size = bus_get_resource_count(dev, SYS_RES_MEMORY, 0 -/*rid*/);
	-}
```

######### ### ### ### ### ### #### ## ##### ## #######.

#######.

```
	xxx_free_resources(sc);
	return 0;
```

#######, ###### ### ########### ##########. ### ### ######### ###### ## ######– ##### ###### #########. ## #### ### ## ### #### #### ###### ### ######### #####

###, ## ## ### #### ### ## #### ######## ### ####–#####: #### ### ########## ## ########.

```
bad:

xxx_free_resources(sc);
if(error)
    return error;
else -/* exact error is unknown */
   return ENXIO;
```

#######. ####### ## ######### ## #### #### ###–##### ######, ## ## ## ##### ## # ######## ##### ### #### ####:

```
static void
 xxx_free_resources(sc)
   struct xxx_softc *sc;
{
   -/* check every resource and free if not zero */

   -/* interrupt handler */
   if(sc->intr_r) {
      bus_teardown_intr(sc->dev, sc->intr_r, sc->intr_cookie);
      bus_release_resource(sc->dev, SYS_RES_IRQ, sc->intr_rid,
         sc->intr_r);
      sc->intr_r = 0;
   -}

   -/* all kinds of memory maps we could have allocated */
   if(sc->data_p) {
      bus_dmamap_unload(sc->data_tag, sc->data_map);
      sc->data_p = 0;
   -}
    if(sc->data) { -/* sc->data_map may be legitimately equal to 0 */
      -/* the map will also be freed */
      bus_dmamem_free(sc->data_tag, sc->data, sc->data_map);
      sc->data = 0;
   -}
   if(sc->data_tag) {
      bus_dma_tag_destroy(sc->data_tag);
      sc->data_tag = 0;
   -}

   -... free other maps and tags if we have them -...

   if(sc->parent_tag) {
      bus_dma_tag_destroy(sc->parent_tag);
      sc->parent_tag = 0;
   -}

   -/* release all the bus resources */
   if(sc->mem0_r) {
      bus_release_resource(sc->dev, SYS_RES_MEMORY, sc->mem0_rid,
```

```
            sc->mem0_r);
        sc->mem0_r = 0;
    -}
    -...
    if(sc->port0_r) {
        bus_release_resource(sc->dev, SYS_RES_IOPORT, sc->port0_rid,
            sc->port0_r);
        sc->port0_r = 0;
    -}
-}
```

10.9. ###_###_######

######. ## ### ##### ####### ######## 0 #### ### ###### ####### ### ###### ## ####### ### ###### ######### ##### ######, ## ## ### ### ## ### ##### #######. #### ## ### ##### ####### ####### 0 ## ### ###### #### ### ###### ####### ### #### ###### ##### ## ###### ## #### ##### ## ### ######. ## ### ##### ####### ####### # ######## ##### #### ### ###### ### #### #### ## ##### ###########.

0 ## ## ######### ############ ## ##### #### #########.

#######, #### ####### #### ########## #### #### #### #### ########## #########.

```
struct xxx_softc *sc = device_get_softc(dev);
int unit = device_get_unit(dev);
int error = 0;
```

#########. ####### ######## ### #### ##### #### ## ######## ###### ######### #### #####, ## ### ## ## ######### #####. ## ###### #### ### ##### ####### ### ######## ### ### ### ######## ######, ## #### ## ##### #### ## ### ######### #####. ## ### ##### ####### ### #### #### ######## ######### #### ## #### ### #### ## ## ######### ##### (##### ##### ## ########## ## #####).

```
sc->port0_rid = 0;
sc->port0_r = bus_alloc_resource(dev, SYS_RES_IOPORT,  &sc->port0_rid,
    -/*start*/ 0, -/*end*/ ~0, -/*count*/ 0, RF_ACTIVE);

if(sc->port0_r == NULL)
     return ENXIO;

-/* on-board memory */
sc->mem0_rid = 0;
sc->mem0_r = bus_alloc_resource(dev, SYS_RES_MEMORY,  &sc->mem0_rid,
    -/*start*/ 0, -/*end*/ ~0, -/*count*/ 0, RF_ACTIVE);
```

```
if(sc->mem0_r == NULL)
     goto bad;

-/* get its virtual address */
sc->mem0_v = rman_get_virtual(sc->mem0_r);
```

(###) ## ######### ########. ## ########## ## ### ####–##### ## ### isa_dma*() ######. ### #######:

isa_dmacascade(sc->drq0);

(###) ## # ### #######. ####### ########## ### ######## ######### ####### ###### ## ########## #### ##. ############ ## ### ### ### ####### ### ######## ###### ## ### ###### ## ### ######### ####### ### ### ###### #### ######. ### ## ###### ####### ### ########## ######## ####### ### ####### ## ######### #####. ### ######### ###### ## #### #### ### ######–##### ##### ### ######### ########### #### ####### ### #### ###### #### ##### ## #### ##### ####### ##### #### ### #### ###### ## #########. #### #### ######–#### ##### ### ####### ### ######### ##### #### #### ####### ### ###### #### ## ##### ### ####: #### ### #### ### ### #####, ######, ###### ### ##### ##########–## ######## ##### ### #### ## ### ###### #### ### ## ###### ##### ####.

```
sc->intr_rid = 0;
sc->intr_r = bus_alloc_resource(dev, SYS_RES_MEMORY,  &sc->intr_rid,
     -/*start*/ 0, -/*end*/ ~0, -/*count*/ 0, RF_ACTIVE);

if(sc->intr_r == NULL)
   goto bad;

-/*
 * XXX_INTR_TYPE is supposed to be defined depending on the type of
 * the driver, for example as INTR_TYPE_CAM for a CAM driver
 */
error = bus_setup_intr(dev, sc->intr_r, XXX_INTR_TYPE,
   (driver_intr_t *) xxx_intr, (void *) sc, &sc->intr_cookie);
if(error)
   goto bad;
```

######:

```
error=bus_dma_tag_create(NULL, -/*alignment*/ 4,
   -/*boundary*/ 0, -/*lowaddr*/ BUS_SPACE_MAXADDR_24BIT,
   -/*highaddr*/ BUS_SPACE_MAXADDR, -/*filter*/ NULL, -/*filterarg*/ NULL,
   -/*maxsize*/ BUS_SPACE_MAXSIZE_24BIT,
   -/*nsegments*/ BUS_SPACE_UNRESTRICTED,
   -/*maxsegsz*/ BUS_SPACE_MAXSIZE_24BIT, -/*flags*/ 0,
```

```
        &sc->parent_tag);
    if(error)
        goto bad;

    -/* many things get inherited from the parent tag
     * sc->data is supposed to point to the structure with the shared data,
     * for example for a ring buffer it could be:
     * struct {
     *   u_short rd_pos;
     *   u_short wr_pos;
     *   char    bf[XXX_RING_BUFFER_SIZE]
     * -} *data;
     */
    error=bus_dma_tag_create(sc->parent_tag, 1,
        0, BUS_SPACE_MAXADDR, 0, -/*filter*/ NULL, -/*filterarg*/ NULL,
        -/*maxsize*/ sizeof(* sc->data), -/*nsegments*/ 1,
        -/*maxsegsz*/ sizeof(* sc->data), -/*flags*/ 0,
        &sc->data_tag);
    if(error)
        goto bad;

    error = bus_dmamem_alloc(sc->data_tag, &sc->data, -/* flags*/ 0,
        &sc->data_map);
    if(error)
         goto bad;

    -/* xxx_alloc_callback() just saves the physical address at
     * the pointer passed as its argument, in this case &sc->data_p.
     * See details in the section on bus memory mapping.
     * It can be implemented like:
     *
     * static void
     * xxx_alloc_callback(void *arg, bus_dma_segment_t *seg,
     *     int nseg, int error)
     * {
     *    *(bus_addr_t *)arg = seg[0].ds_addr;
     * -}
     */
    bus_dmamap_load(sc->data_tag, sc->data_map, (void *)sc->data,
        sizeof (* sc->data), xxx_alloc_callback, (void *) &sc->data_p,
        -/*flags*/0);
```

#######–####. ### ############## ### ####### ####### #### ### ### ######## ######## ### ##########.

```
    if(xxx_initialize(sc) < 0)
         goto bad;
```

#######–#### ### ## #####. ### ## ### ###### ##### ## ##### #### ##### ########### ##### ### ###### ## ### ## ##, ### #######:

```
device_printf(dev, -"has on-card FIFO buffer of %d bytes\n", sc->fifosize);
```

###########.

######. ### ##### ### ## ## ## ####### ## ### #### ## ### ######: # ######### ######, # ##### ######, # ####### ######, # ### #### ### ###### ### ## ##.

#######.

```
error = xxx_attach_subsystem(sc);
if(error)
    goto bad;

return 0;
```

#######, ###### ### ########### ##########. ### ### ######### ###### ## ######–##### ###### ######### ## #####. ## #### ### ## ### #### #### ###### ### #####–#### ##### ## ###### ## ## ## #### ###### ###, ## ## ### #### ### ## #### ######## ### #########: #### ### ########## ## ########.

```
bad:

xxx_free_resources(sc);
if(error)
    return error;
else -/* exact error is unknown */
    return ENXIO;
```

#######.

10.10. ###_###_######

####–#### ###### #### ### ###### #### ### ####### ## ## ########. #### ## ## #####–#### ####### ## ### ######## ######## ### ####. ### ### ### ### #### ### ####### ### ####, ## #### ####### ## ### ############ ######### ### ### ### #######. ### ####### ## ###### # ###### ### ## ###### #### ######### ##, ### ## #### ##### ############ ## ### ### ####### ## ### ###### ##### ## ######## #### ##### ### ### ####### ####### ###### ### ###### ### # ###### #### ## ###### ######, ## ### ####### ##### ## ####### ### ###### ####### ### ### ## ##### ##. ####### ######## #### ######### ##### ##### ######### ### ####### ## # ########## ####### ##### ## ## ###### ###########. ########## # ### ####### ## # ###### ## #

######### ##### ###### ##### ## ######### ## # ########## ####### (### ##### ## ### ######### #### ### ###### ## ####### ## ###### ####). #####, ### ###### ####### ### ## ######## ### ### #### ## ############.

0 ## ### ###### ### ############ ######## ## ### ##–### #### #########.

######. ### ##### ##### ## ## ## ## ###### ### ###### #### ### ####### #########. ## ### ###### ## ######### #### ##### ### ###### ### ### #######: ###### ## ## ######## ## ######## ##### ### ####### #### ######. ### ###### #### ####### ## ### ####### ## ### ########## ###### ######–### ## ## # ###### ##### ### ## ### ########### ## ### ######## ######. ######### ### ###### ##### ##### ## ## ### ######### ###########.

```
struct xxx_softc *sc = device_get_softc(dev);
int error;

error = xxx_detach_subsystem(sc);
if(error)
    return error;
```

#####. #### ######## ######## ### ####### #########, ######### ### ### ######## ### #####–##### ## ##### ###### ########## ## ### ######. ### #### ## ### ####### #### ## ####### #### ### ######## ####### ####, ## ## ## ## ######## ## ### ###### ## ### #### #### ##.

xxx_isa_shutdown(dev);

#######.

```
xxx_free_resources(sc);
return 0;
```

10.11. ###_###_########

####. ## ## ######## ## ##### ### ######## ## ##### ########## #####. ### #### ## ### ### ####### ## ####### ###### ## ########, ## ### ######## ## ### ###### ######### ####### ### ###### #### ## ############## ## ###### ######. ### #### ####### #### ## ## #### #### #### # ####### #########, ## #### #### #### #### #### ## ######## ######## ##### #### ###### (#### ## ########## #### ### #### ####### #### ########### ############# #########). ## ### #### ######### ### ### ########## ## ### ##–#### ######### ### ######## ### ####### ######### ## # #### ####. ### ##### ##–#### ####### ## ### ########, ## ## ## ### ######## ## #### ## ### ######.

10.12. ###_####

######. ### ### ### #### ### ####### ######### ####### (###### ## #### ####### #####) ## ## ######## ## ### ######### ####### ## ###### #### ### ######### ###### ### #### #### #### ### ######. #####, ### ######### ####### #### #### ### ###### ######### ### #### #### #### ### ######### ### ######### ## ### ######. ## ### ## ###### #### ######.

########. #### ## ########, ### ### ### ####### ####### ######## ######## ### ######### ### #### ## ### ###### ####### #### ####### bus_setup_intr(). ## ### ### ########## ## ###### ## ### ####### ## ### ######### #####. ## ### ######### ####### ######## ###### ##:

```
static void
xxx_intr(struct xxx_softc *sc)
{
```

####– ##### ## bus_setup_intr(). #### ##### #### ### ### ##### ########## ## ### #### #### ## #### ## ### ### ######## ########## ### ########.

####:

```
while(xxx_interrupt_pending(sc)) {
   xxx_process_interrupt(sc);
   xxx_acknowledge_interrupt(sc);
-}
```

##########, ### ###### ##### #### ## ### ######.

11. ###

###.

11.1. ##### ###

########### #### ##### ### ### ### ### #### ######## ####### ### ########## ####### ### ### ## # ##### ###### ### #### ###### ## ### ## ####.

11.1.1. ###### ###### ###### (**mypci.c**)

```
/*
 * Simple KLD to play with the PCI functions.
 *
 * Murray Stokely
 */

#include <sys/param.h>  /* defines used in kernel.h */
#include <sys/module.h>
#include <sys/systm.h>
#include <sys/errno.h>
#include <sys/kernel.h>  /* types used in module initialization */
#include <sys/conf.h>  /* cdevsw struct */
#include <sys/uio.h>  /* uio struct */
#include <sys/malloc.h>
#include <sys/bus.h>  /* structs, prototypes for pci bus stuff and DEVMETHOD macros! */

#include <machine/bus.h>
#include <sys/rman.h>
#include <machine/resource.h>

#include <dev/pci/pcivar.h> /* For pci_get macros! */
#include <dev/pci/pcireg.h>

/* The softc holds our per-instance data. */
struct mypci_softc {
 device_t my_dev;
 struct cdev *my_cdev;
};

/* Function prototypes */
static d_open_t  mypci_open;
static d_close_t mypci_close;
static d_read_t  mypci_read;
static d_write_t mypci_write;

/* Character device entry points */
```

```
static struct cdevsw mypci_cdevsw = {
 .d_version = D_VERSION,
 .d_open = mypci_open,
 .d_close = mypci_close,
 .d_read = mypci_read,
 .d_write = mypci_write,
 .d_name = "mypci",
};

/*
 * In the cdevsw routines, we find our softc by using the si_drv1 member
 * of struct cdev.  We set this variable to point to our softc in our
 * attach routine when we create the -/dev entry.
 */

int
mypci_open(struct cdev *dev, int oflags, int devtype, d_thread_t *td)
{
 struct mypci_softc *sc;

 /* Look up our softc. */
 sc = dev->si_drv1;
 device_printf(sc->my_dev, -"Opened successfully.\n");
 return (0);
}

int
mypci_close(struct cdev *dev, int fflag, int devtype, d_thread_t *td)
{
 struct mypci_softc *sc;

 /* Look up our softc. */
 sc = dev->si_drv1;
 device_printf(sc->my_dev, -"Closed.\n");
 return (0);
}

int
mypci_read(struct cdev *dev, struct uio *uio, int ioflag)
{
 struct mypci_softc *sc;

 /* Look up our softc. */
 sc = dev->si_drv1;
 device_printf(sc->my_dev, -"Asked to read %d bytes.\n", uio->uio_resid);
 return (0);
}

int
mypci_write(struct cdev *dev, struct uio *uio, int ioflag)
{
 struct mypci_softc *sc;
```

```
 /* Look up our softc. */
 sc = dev->si_drv1;
 device_printf(sc->my_dev, -"Asked to write %d bytes.\n", uio->uio_resid);
 return (0);
}

/* PCI Support Functions */

/*
 * Compare the device ID of this device against the IDs that this driver
 * supports.  If there is a match, set the description and return success.
 */
static int
mypci_probe(device_t dev)
{

 device_printf(dev, -"MyPCI Probe\nVendor ID -: 0x%x\nDevice ID -: 0x%x\n",
     pci_get_vendor(dev), pci_get_device(dev));

 if (pci_get_vendor(dev) == 0x11c1) {
  printf("We've got the Winmodem, probe successful!\n");
  device_set_desc(dev, -"WinModem");
  return (BUS_PROBE_DEFAULT);
 }
 return (ENXIO);
}

/* Attach function is only called if the probe is successful. */

static int
mypci_attach(device_t dev)
{
 struct mypci_softc *sc;

 printf("MyPCI Attach for -: deviceID -: 0x%x\n", pci_get_devid(dev));

 /* Look up our softc and initialize its fields. */
 sc = device_get_softc(dev);
 sc->my_dev = dev;

 /*
  * Create a -/dev entry for this device.  The kernel will assign us
  * a major number automatically.  We use the unit number of this
  * device as the minor number and name the character device
  * -"mypci<unit>".
  */
 sc->my_cdev = make_dev(&mypci_cdevsw, device_get_unit(dev),
     UID_ROOT, GID_WHEEL, 0600, -"mypci%u", device_get_unit(dev));
 sc->my_cdev->si_drv1 = sc;
 printf("Mypci device loaded.\n");
 return (0);
}
```

```
/* Detach device. */

static int
mypci_detach(device_t dev)
{
 struct mypci_softc *sc;

 /* Teardown the state in our softc created in our attach routine. */
 sc = device_get_softc(dev);
 destroy_dev(sc->my_cdev);
 printf("Mypci detach!\n");
 return (0);
}

/* Called during system shutdown after sync. */

static int
mypci_shutdown(device_t dev)
{

 printf("Mypci shutdown!\n");
 return (0);
}

/*
 * Device suspend routine.
 */
static int
mypci_suspend(device_t dev)
{

 printf("Mypci suspend!\n");
 return (0);
}

/*
 * Device resume routine.
 */
static int
mypci_resume(device_t dev)
{

 printf("Mypci resume!\n");
 return (0);
}

static device_method_t mypci_methods[] = {
 /* Device interface */
 DEVMETHOD(device_probe,  mypci_probe),
 DEVMETHOD(device_attach, mypci_attach),
 DEVMETHOD(device_detach, mypci_detach),
 DEVMETHOD(device_shutdown, mypci_shutdown),
 DEVMETHOD(device_suspend, mypci_suspend),
```

```
    DEVMETHOD(device_resume, mypci_resume),

    DEVMETHOD_END
};

static devclass_t mypci_devclass;

DEFINE_CLASS_0(mypci, mypci_driver, mypci_methods, sizeof(struct mypci_softc));
DRIVER_MODULE(mypci, pci, mypci_driver, mypci_devclass, 0, 0);
```

11.1.2. **Makefile** ### ######

```
# Makefile for mypci driver

KMOD= mypci
SRCS= mypci.c
SRCS+= device_if.h bus_if.h pci_if.h

.include <bsd.kmod.mk>
```

Makefile #### # #########, ### ### ### make ## ####### ### ###### ######. ############, ### ### ### make load ## #### ### ###### #### ### ######### ####### ###### ### make unload ## ###### ### ###### ##### ## ## ######.

11.1.3. ##########

- ### ####### ######## #####
- ### ###### ############, ###### ####### ## ### #######, ## ##.

11.2. ###

####### ######## ## ############### ######### ### ########## ######### #### # ###### ###. ###### ### ####### #### ## # ##### ###### ## #### #### ## ### (###, ###, ###, ####, ###) ### ##### ####### #### ## ####### ######### #### ##### ###### ### (#### ## ###### ########, ######### #####, ## ### ########).

11.2.1. #### #######

Base Address Registers (####) #### ### ### ############# #####. ### ############ ####### ## ######### ### ### ### ########## ## ### bus_alloc_resource() ########.

#######, # ####### ###### ##### #### ######### ####### ## #### ## ### attach() ########:

```
    sc->bar0id = PCIR_BAR(0);
    sc->bar0res = bus_alloc_resource(dev, SYS_RES_MEMORY, &sc->bar0id,
      0, ~0, 1, RF_ACTIVE);
    if (sc->bar0res == NULL) {
        printf("Memory allocation of PCI base register 0 failed!\n");
        error = ENXIO;
        goto fail1;
    -}

    sc->bar1id = PCIR_BAR(1);
    sc->bar1res = bus_alloc_resource(dev, SYS_RES_MEMORY, &sc->bar1id,
      0, ~0, 1, RF_ACTIVE);
    if (sc->bar1res == NULL) {
        printf("Memory allocation of PCI base register 1 failed!\n");
        error =  ENXIO;
        goto fail2;
    -}
    sc->bar0_bt = rman_get_bustag(sc->bar0res);
    sc->bar0_bh = rman_get_bushandle(sc->bar0res);
    sc->bar1_bt = rman_get_bustag(sc->bar1res);
    sc->bar1_bh = rman_get_bushandle(sc->bar1res);
```

####### ### #### #### ####### ######## ### #### ## ### softc ######### ## #### #### ### ## #### ## ##### ## ### ###### #####.

bus_space_* #########. ### #######, # ###### ##### ####### # ######### ######## ## #### #### # ##### ######## ######## #### ####:

```
uint16_t
board_read(struct ni_softc *sc, uint16_t address)
{
    return bus_space_read_2(sc->bar1_bt, sc->bar1_bh, address);
}
```

#########, ### ##### ##### ## ### ######### ####:

```
void
board_write(struct ni_softc *sc, uint16_t address, uint16_t value)
{
    bus_space_write_2(sc->bar1_bt, sc->bar1_bh, address, value);
}
```

8###, 16###, ### 32### ######## ### ### ###### ### bus_space_{read|write}_{1|2|4} ###########.

> ####
>
> ## ####### 7.0 ### #####, ### ### ### ### bus_* ######### ##–##### ## bus_space_*. ### bus_* ######### #### # ###### ########

######. ####, ### ##### #### ### ### ### ### ### ###### ####### #### ### softc ### ####### ### board_read() ######## ##:

```
uint16_t
board_read(struct ni_softc *sc, uint16_t address)
{
 return (bus_read(sc->bar1res, address));
}
```

11.2.2.

########## ### ######### #### ### ############### ### #### ## # ### ####### ## ### ###### #########. ##### ## ### ######## #### ## ######### #### ### ###### ###, ### #### ### ######### ####### #### ## ### ## ## #### #### #### ###.

#####, # ###### #### # ###### attach() ######## #### #### #### #####.

```
/* Get the IRQ resource */

   sc->irqid = 0x0;
   sc->irqres = bus_alloc_resource(dev, SYS_RES_IRQ, &(sc->irqid),
     0, ~0, 1, RF_SHAREABLE -| RF_ACTIVE);
   if (sc->irqres == NULL) {
 printf("IRQ allocation failed!\n");
 error = ENXIO;
 goto fail3;
   -}

   -/* Now we should set up the interrupt handler */

   error = bus_setup_intr(dev, sc->irqres, INTR_TYPE_MISC,
     my_handler, sc, &(sc->handler));
   if (error) {
 printf("Couldn't set up irq\n");
 goto fail4;
   -}
```

######. ### #### ####### ### ######## ######### ######, ### ###### ### ######### #######. #### bus_teardown_intr() ### ########, ### #### #### #### ######### ####### #### ## ###### ## ###### ### #### ### ####### #### ##### #### #### ######### #### ######### ####### #### ##–######. ##### #### ######## ### #####, ### #### ### #### ### ####### #### ####### #### ########.

11.2.3.

#########, ### ####### #### ### ########## #######. ### ###### ####### ### ####### #### ##### ###### ## ## ### ### bus_space_dma*() ######### ##–#####. #### ######### ### ## ####### #### #### ####### ## ####### ## ####### #### #####. #######, ## ### ######, ### ### ## ## # ### ## ####, ## #### #### ####### ####, ## ##### ## #### ## ###### #### ####### ## ####### ####.

##, ########### #### #### ## ## ############# ### #### #### #### ######## #########. #### ## # ####### ##### ####### #### ####### ###### ### ##### ###### ########### #### ####### #########. ###########, ##### ## # ########, vtophys() ## ####.

```
#include <vm/vm.h>
#include <vm/pmap.h>

#define vtophys(virtual_address) (...)
```

#######, ### #### ## ###### #### ## # ######## ###### vtobus().

```
#if defined(__alpha__)
#define vtobus(va)      alpha_XXX_dmamap((vm_offset_t)va)
#else
#define vtobus(va)      vtophys(va)
#endif
```

11.2.4. ############

###–### attach(). #### #### ## ##### ## ########### ### ####### ##### #### ## # ####### ######### ## #### ### ###### #### ####### ###### ##### #### ###### ####.

12. ####### ####### ####### ####

Written by Sergey Babkin.
Modifications for Handbook made by Murray Stokely.

12.1.

########. #### ## ### ########### ## #### ######## ### ######### #### ### #######:

- ### (/sys/pci/ncr.c) ## ######## ########### ### ###### #####
- ### (/sys/dev/sym/sym_hipd.c) ## ###### #######
- ###7### (/sys/dev/aic7xxx/aic7xxx.c) ## ###### #. #####

(## ###### #. #####, ### /sys/cam/*). #### #### ######## ###### ### #### ####### ### ### ########### ######## ######### #### ### #### ## ###### #. #####, # ###### ## ## #############.

###########. ######## ######### ### ######## #### #### ####### ### #### #### #### ####, ## ## ##### ###########. ## ### ####### ## ########### ### ######## ## ## ############## ###. ### # #### ###### ##### ########## ### ## #### ####### ### #########. ## #### ######### #### ### ######## #######, ## #### ## ###### #### ##### ##### ### ############ ######## ## #### ### ######## ## ## ######### ## ### ##### ######## ## ### ##–######## ########. #### ####### ### ######## ##### ## ##### ## ######### #### ########### #####, ######## ## #################. ########### #### #### ##### #### ######## #### ### ### ####### ### ## ##### ## ### #### #######.

12.2. #######

######. ## ## # ####### ### ## ####### ### #/# ##### ## # ######### ###. #### ###### # ########## ## ### ####### ###### ####### #### ### ####### ########### ### #/# ###: ### ####### ### #### ###### ####### #### ## ####### ##### ## #### ####, ###, ###/## ### ##### ### ## ### #### ###### ####### #### ### #### ## ## ######### (## ###### ### ########) ### ##### ### #/# ###. #### ### ### #### ######### ###### ######## ###:

- *Peripheral Modules* # # ###### ### ########## ####### (####, ####, ######, ###.)
- *SCSI Interface Modules* (###) # # #### ### ####### ####### ### ########## ## ## #/# ### #### ## #### ## ###.

##, ######## #### ## # ######## ## #### ######## ### ###### ##### #### ######## ## # #### ######### ######. ### #### ######### ###### ## ########### ### ####### ##### ######## ## ### ###### ######## (## ## ### ###### ######## ## ### #### ###, ### #######, ### #### #### ########## ### #### ######## ## ### ###### ######## ## ### ########).

####### ## ### ########## ## ####### # #### ####### ###### ####, #### #### ##### ## ## #### ######## ########## #### ### ### ##########.

#####:

```
#include <cam/cam.h>
#include <cam/cam_ccb.h>
#include <cam/cam_sim.h>
#include <cam/cam_xpt_sim.h>
#include <cam/cam_debug.h>
#include <cam/scsi/scsi_all.h>
```

#########. #### ## #### ###### ### ######### xxx_attach() ######## (#### ### ####### ###_ ## #### ## ###### ### ###### ###### #### ######). ### xxx_attach() ######## ###### ## ###### ## ### ###### ### ################## #### ##### ## ## ### ######## ####.

#####: ##### ## ## ######### ## ######## ### ##### ## ######## ########## #### #### ###:

```
    struct cam_devq *devq;

    if(( devq = cam_simq_alloc(SIZE) -)==NULL) {
        error; -/* some code to handle the error */
    -}
```

SIZE ## ### #### ## ### ##### ## ## #########, ####### ###### ## ######## ## ##### #######. ## ## ### ###### ## ######## #### ### ### ###### ### ###### ## ######## ## ### #### ####. ######## ## ### ## ########## ##:

```
SIZE = NUMBER_OF_SUPPORTED_TARGETS * -
MAX_SIMULTANEOUS_COMMANDS_PER_TARGET
```

###:

```
    struct cam_sim *sim;

    if(( sim = cam_sim_alloc(action_func, poll_func, driver_name,
            softc, unit, max_dev_transactions,
```

```
        max_tagged_dev_transactions, devq) -)==NULL) {
    cam_simq_free(devq);
    error; -/* some code to handle the error */
  -}
```

devq #### ##–##### ## ### ## ####### #### #### ## ### ## #### ## ######## ######.

cam_sim #########.

###, ## ## #### ### devq ######### ### #### ## ### #### ###? ### ###### ##### ## ### ######## ## ### ### #### ##: ###### ###, ## ### ######## ###### #######.

###:

- action_func # ####### ## ### ######## xxx_action ########.

  static void **xxx_action** (*struct cam_sim *sim*, *union ccb *ccb*);

  *struct cam_sim *sim,* , *union ccb *ccb* ;

- poll_func # ####### ## ### ######## xxx_poll()

  static void **xxx_poll** (*struct cam_sim *sim*);

  *struct cam_sim *sim* ;

- ######_#### # ### #### ## ### ###### ######, #### ## ##### ## #####.
- softc # ####### ## ### ######## ######## ########## ### #### #### ####. #### ####### #### ## #### ## ### ###### ## ###### ## ### ####### ####.
- #### # ### ########## #### ######, ### ####### ### ########## ####0# #### ###–### #### ## 0
- ###_###_########### # ####### ###### ## ############ ########### ### #### ###### ## ### ########## ####. #### ##### #### ## ###### ########### ##### ## 1, #### ######## ########## #### ### ### ######## #####. #### ### ####### #### #### ## #### ######### ## ######### ### ########### ##### ####### ### ## ######## ### ### ## ## 2 ### #### #### ### #### ## ## ##### ### ##########.
- ###_######_###_########### # ### #### #####, ### ## ### ###### ####. #### ### ### #### ### ## ######## ######## ############ ## # ######: #### ########### ## ######## # ###### ### ### ### ########### ## #### ## ### ######. #### ### ###### ######### #### ########### ## ##### #### ### ###### ######## #### ### ### ## #### ### #### ####### (### ### ######) ### #### ##### ########### ###

#########. #### ######## ## #### ##### ## ### ####### ### #####. ## ####### ## ### ######### ## ### #### #######.

####### ## ######## ### #### ##### ########## #### ### #### #######:

```
if(xpt_bus_register(sim, bus_number) -!= CAM_SUCCESS) {
   cam_sim_free(sim, -/*free_devq*/ TRUE);
   error; -/* some code to handle the error */
-}
```

devq ######### ### #### ### (#.#., ## ######## # #### #### ######## ##### ## ######## ##### #### ### ### ####) #### ### ### ###### #### ###### ## 0, ######## #### ### ## ### #### #### ###### ## ### # ######## ######. #### ### ##### ### ### ######## ######### ###_###.

######. ### ##### ## devq ### ## ######### ###: ### #### ## ###### ## ## ######## ## ### ####### ##### #### ### ### #### ### ## ####### #### ##.

######. #### ###### ######### #### ### ##### ###### (### ### #######), #### ###### ######### #### ### ######–#### #######, #### ###### ######### #### ### ### ######### ######. ### ###### ### ######## ######### ### #### ##### ## ### ############ ######, ## #### ## ##### ## ######## ## ##### ###### #####.

#####. #### ########### ### ##### ########## ### ####### ## ##### ## ####### ## ### ##### ## ######. ### ###–############ ###### ######## ##### ###### # ########### #### #### ######. ## ### #### #### #### ############ ### ## ####### ## ###### #### ##### (#### ## #### ####### ### ##### #### ## ###### ## ### ##### ######### ####### ### ### ######–##### ### ###### ######## #######). #### ## ## #### ## ######## ##### #########–## ## ### #### ######### ### ######### ########, ######## #### ###### ####### ########, ### # ######## ####### #### #### ######## ## #### ##### ### ## ## #### ###### ## #### ####. ### # ### ##### ##### ## #### ## ###### # ######## #### ##–####### ### ####### ## ### ###. ## ## ### ###### ### #### ### ### ###### ### ##### ###### ## ####### ### ##### ######## #### ### ###### ###### ########:

```
struct cam_path *path;

if(xpt_create_path(&path, -/*periph*/NULL,
         cam_sim_path(sim), CAM_TARGET_WILDCARD,
         CAM_LUN_WILDCARD) -!= CAM_REQ_CMP) {
   xpt_bus_deregister(cam_sim_path(sim));
   cam_sim_free(sim, -/*free_devq*/TRUE);
   error; -/* some code to handle the error */
-}

softc->wpath = path;
softc->sim = sim;
```

########:

- ## ## ### ########## ###### (#### #### ####### ## #### ####)
- ## ## ### ### ###### (cam_sim_path(sim))
- #### ###### ###### ## ### ###### (###_######_######## ##### #### ########)
- #### ### ###### ## ### ######### (###_###_######## ##### #### #####)

########, ## ## #### #### ## ######### #### #### ###.

softc ######### ### ###### ###. ##### #### ## #### ### ##### ## ### (## ## ### #### ####### ## ## ### #### #### xxx_probe() ## ## ####).

##############. ## ## ###### ##### ##### ## ### #### ##### ####.

#####: #### # ###### ##– #### ## ########## ####. ## #### #### ######### ### #### ############ #### #### ###### ### ## # #### ####. ## ## ######## # ######## ### #### ##### #### ###. ### ####### ## ###### ## ### ## ########## ### ###### ## # ### ####### ##### ### #### #### ## #######:

```
struct ccb_setasync csa;

xpt_setup_ccb(&csa.ccb_h, path, -/*priority*/5);
csa.ccb_h.func_code = XPT_SASYNC_CB;
csa.event_enable = AC_LOST_DEVICE;
csa.callback = xxx_async;
csa.callback_arg = sim;
xpt_action((union ccb *)&csa);
```

xxx_action() ### xxx_poll() ###### ##### ######.

static void **xxx_action** (*struct cam_sim *sim*, *union ccb *ccb*);

*struct cam_sim *sim*, , *union ccb *ccb* ;

#########. ### ######### ### ### ### ### #######, ### ## ### ####### ######. ### ###### ### #### ####### ######. ## ## # ##### ## #### ######## #########, #### ########## ######### ### #### #### ## ############. ### ## ##### ######### ##### ### ### ###### ##### ### ###### #### ## ######### ## ######.

(########) #### ### ###### (########## # #### ######) ####. #### ## #### ######## ### #### ######## ## ### ######### ####.

(## ##### #####, #######) ####### ## ###### ### ###### #### ## ### ###### ###:

- cam_sim_path(sim) # ### #### ## (### #####)
- cam_sim_name(sim) # ### #### ## ### ###
- cam_sim_softc(sim) # ### ####### ## ### ##### (###### ####### ####) #########
- cam_sim_unit(sim) # ### #### ######
- cam_sim_bus(sim) # ### ### ##

######, xxx_action() ### ### ### #### ###### ### ####### ## ### #####–#### ##### ##### ##### #########.

ccb->ccb_h.func_code. ## ######### xxx_action() ######## ## # ### ######:

```
struct xxx_softc *softc = (struct xxx_softc *) cam_sim_softc(sim);
struct ccb_hdr *ccb_h = &ccb->ccb_h;
int unit = cam_sim_unit(sim);
int bus = cam_sim_bus(sim);

switch(ccb_h->func_code) {
case -...:
   -...
default:
   ccb_h->status = CAM_REQ_INVALID;
   xpt_done(ccb);
   break;
-}
```

(## ## ####### ####### ### ########) ### ###### #### ## ### ####### ## ### #### ccb->ccb_h.status ### ### ######### ### ## ##–###### #### ## ### ## ####### xpt_done(ccb).

xpt_done() #### ### #### ## ## ###### #### xxx_action(): ### ####### ## #/# ####### ### ## ######## ###### ### ### ###### ###/## ### #### ##########. #### #### ### ##–#### ##### #### ## ######### ######### #### ### ########## ## #### ####### ## ######## xpt_done() ### ## ###### #### ### ######### ######## #######.

########, ### ### ###### ## ### #### ######## ## # ###### #### ### # ### ### #### ###### ### ### ####. ###### ### ## ###### ## ### xxx_action() ####### ## #### ### ###### ###_###_###### ####### #### ## ## ## ########. ##### ### # ########## ###### ## ###### ###### ####### ## /sys/cam/cam.h ##### ###### ## #### ## ######### ### ###### ## # ####### ## ##### ######. #### ########### ###, ### ###### ## ## #### # ######## ### ## ## ########## ###### ##### (### ##### 6 ####) ### #####–### ########## ######### #### (### ##### ####). ### ########## ###### #### ##

######### ##### ## #### ######. ### ####### ## #### ### ## ##### ## ### ###### ####### #######. ### ######## ###### ##### ###:

- *CAM_DEV_QFRZN* # ## ### ### ###### #### # ####### ##### (### #######, ### ###### #### ### ####### ## ### ######### ## ###### ### #### ########) #### ########## # ### ## ###### ###### ### ####### ##### ## ####### xpt_freeze_simq(), ###### ### ##### ######## ### ### ######### ### #### ### #### ###### #### ## ### ### #####, #### ### #### #### ### ### ########### ### ### #### xpt_done(). #### #### ###### ### ### ######### ## ######## ### ##### ##### ## ####### ### #####.
- *CAM_AUTOSNS_VALID* # ## ### ###### ######## ## ##### ######### ### ### #### ###_###_######### ## ### ### ## ### ### ### ###### #### ####### ### ##–##### ##### ####### ############# ## ####### ### ##### (######## ##### ##–#########) #### #### ### ######. ## #### ####### ### ########## ### ##### #### ###### ## ##### ## ### ### ### #### #### ###.
- *CAM_RELEASE_SIMQ* # #### ###_###_##### ### #### ## #### ##### ## #### ####–### (## ######## ########) #### ### #### ########## ######. #### ### ### ###### ######## ## ### ########## ###### ## ####### ## xpt_freeze_simq(). ### ########## ##### #### ## ######### ##### ### ### ###### ######### ### ######## ### ##–##### ### ## ######### #### ### #### #### #### ###.
- *CAM_SIM_QUEUED* # #### ### #### # ### #### ### ####### ##### #### #### ###### ## ### (### ####### #### #### ### #### ######## ###### ##### ######## #### ## ###). #### #### ## ### #### ######## ## ### ### #### ###, ## ### ####### ## ###### ##########.

xxx_action() ## ### ####### ## #####, ## ### ### ############### ### ######## ###### #### ## #### ##### ### ## ###### ##### ########. ####### ### ############# ##### ### ### ######### ######## ######### xpt_release_simq() ### xpt_release_devq() ## ######## ### ###### ########, ####### ####### # ### ## ###.

######:

- *path* # #### ## ### ### #######
- *target_id* # ###### ###### ## ### ### #######
- *target_lun* # ### ## ## ### ###### ######
- *timeout* # ####### ######## ### #### #######, ## ############
- *timeout_ch* # # ########### ##### ### ### ### ###### ## ##### ### ####### ###### (### ### ######### ###### #### ### #### ### ########### ##### ##)
- *flags* # ####### #### ## ########### ##### ### ####### #####_###0, #####_###1 # ###### ######## ### ####### ### ## ### ### ###### (#### ## ####### ## ### ### ###### ## ### ####### ####### ######); ########, #### ##### ##

######: #####_###0 ### #####_###1 #### ### #### (#### #), #####_#####0 ### #####_#####1 #### ### #### ######## ####, ###_####.#######[0].##### ### ###_####.#######[1].##### ### #### ###### ## ### #### ########## #### ### ###–## ############ ## ### ##### ### ###_####.##### ## ### #####, ##### ######.

######, ####:

```
#define ccb_some_meaningful_name    sim_priv.entries[0].bytes
#define ccb_hcb spriv_ptr1 -/* for hardware control block */
```

###:

- *XPT_SCSI_IO* # ####### ## #/# ###########

  ### ######## ####### ###_###### ##### ## ### ##### ### ## #### ## ######## ### #########. #### ###:

  - *cdb_io* # ####### ## ### #### ####### ###### ## ### ###### ######
  - *cdb_len* # #### ####### ######
  - *data_ptr* # ####### ## ### #### ###### (#### # ### ########### ## ####–###/###### ## ####)
  - *dxfer_len* # ###### ## ### #### ## ########
  - *sglist_cnt* # ####### ## ### #######/###### ########
  - *scsi_status* # ##### ## ###### ### #### ######
  - *sense_data* # ###### ### ### #### ##### ########### ## ### ####### ####### ## ##### (### ### ###### ## ######## ## ### ### ####### ##### ####### #######–###### ## #### #### ## ### ### #### ###_###_######### ## ### ###)
  - *sense_len* # ### ###### ## #### ###### (## ## ####### ## ## ###### #### #### ## #####_#### ### ### ###### #### ######## ###### ### ####### #####) #####, #####_##### # ## ### ######## ## #### ## #### ##### ######## ## ##### ##### ### ### ######## ######## ## ### ######## (### ###########) ####. #### ## ### #### ## ## ########## ##########, ## ## # #### #### #### ### ######### ## ####### (###, ######## ##### ## ### #### ############ #### ######) ## ##–######### ##### #### ## ## ####. ### # ############ ######### ######## #### #### ## ### ## ####.
  - *tag_action* # ### #### ## ### ## ###:

    - ###_###_######_#### # ## ### ### #### ### #### ###########

- ###_######_#_###, ###_####_##_#_###, ###_#######_#_### # ##### ##### ## ### ########### ### ####### (### /###/###/####/####_#######.#); #### ##### #### ### ### ####, ### ### ###### #### ###### ### ### ##### ######

#########:

#####, ## #### #### #### ### ####### ### ### ### ####### #### ## ### ####### ## ### #####:

```
struct ccb_scsiio *csio = &ccb->csio;

if ((ccb_h->status & CAM_STATUS_MASK) -!= CAM_REQ_INPROG) {
    xpt_done(ccb);
    return;
-}
```

##########:

```
if(ccb_h->target_id > OUR_MAX_SUPPORTED_TARGET_ID
-|| cch_h->target_id == OUR_SCSI_CONTROLLERS_OWN_ID) {
    ccb_h->status = CAM_TID_INVALID;
    xpt_done(ccb);
    return;
-}
if(ccb_h->target_lun > OUR_MAX_SUPPORTED_LUN) {
    ccb_h->status = CAM_LUN_INVALID;
    xpt_done(ccb);
    return;
-}
```

(#### ## ############## ######## ###–#### #####) ## #### ## ####### #### #######. ## ## ### ### #### ###### ### ### ##### ### ######## #### ## #### # ####### #########, ###### ### ### #### ### ### ### ## ######## ##. ##### #### ### ######### ###### ######### ### ### ##### #### ## ######## ## ######### # ### #### ### CAM_SIMQ_RELEASE ### ### ## ### ######. #########, ## ### #### ####, #### ### ### #### ### ######## ####### ##### (###) ### #### ## ## ######.

```
struct xxx_hcb *hcb = allocate_hcb(softc, unit, bus);

if(hcb == NULL) {
    softc->flags -|= RESOURCE_SHORTAGE;
    xpt_freeze_simq(sim, -/*count*/1);
    ccb_h->status = CAM_REQUEUE_REQ;
    xpt_done(ccb);
    return;
-}

hcb->ccb = ccb; ccb_h->ccb_hcb = (void *)hcb;
```

```
    ccb_h->status -|= CAM_SIM_QUEUED;
```

####### ### ###### #### #### ### #### ### ######## ####### #####. ##### ## ## ### ##### ## ###### # ### ### ## ### #### ######## ## ###### ### ### ##### ### #### ### ########. ### ### ###### ## #### ########### ### ############ #### ### ####### ## ### ### ####### ######## ######### ### #####, ########### #### ### ######.

```
    hcb->target = ccb_h->target_id; hcb->lun = ccb_h->target_lun;
    generate_identify_message(hcb);
    if( ccb_h->tag_action -!= CAM_TAG_ACTION_NONE -)
        generate_unique_tag_message(hcb, ccb_h->tag_action);
    if( -!target_negotiated(hcb) -)
        generate_negotiation_messages(hcb);
```

#######. ### ####### ####### ### ## ######### ## ### ### ## #### ########### ####, ######### ## ### ### #####. ### ####### ###### ### ## ######### ## ### ## ####### ##, ## ### ###### #### ### ####### ### ## ####–#### ## #######. ##### ### ######## ######## ##### ######## ####### ## ###### ####### ### ####### ## ### ######## ###.

####: ######## #### ## #### ## # #### ## vtophys(), ### ### ### ### ###### (##### ####### ### #### ## ### #### ########### ###) ######## ########### ## ### ##### ############ ### ########## #### ## #### ## vtobus() ####### ### ## ####### ##### ######. [#### ## ##### ## #### ###### ## #### ### ######## #########, vtop() ### ptobus() #### vtobus() ##### ## # ###### ############# ## ####.] ## #### ## # ######## ####### ## ######### ## ## ## ## ###### ### ### #### ### ###### ###_###_#######, ### ####### ####### ## ####. ### ## ## #### ######## ## ####### ### ############## ##### ## ####, ## ## #### ####### (##### ###### ## # #### ###### ### ## ## ####, ####### ########### ########### ## ### #######). ## ######### # ######## ####### ### ## #### ######### ## ###### #### ## # ####### ####### ### #### ### ####, ## ## ## ### ## ####.

```
    if(ccb_h->flags & CAM_CDB_POINTER) {
        -/* CDB is a pointer */
        if(!(ccb_h->flags & CAM_CDB_PHYS)) {
            -/* CDB pointer is virtual */
            hcb->cmd = vtobus(csio->cdb_io.cdb_ptr);
        -} else {
            -/* CDB pointer is physical */
#if defined(__alpha__)
            hcb->cmd = csio->cdb_io.cdb_ptr -| alpha_XXX_dmamap_or -;
#else
            hcb->cmd = csio->cdb_io.cdb_ptr -;
#endif
        -}
    -} else {
        -/* CDB is in the ccb (buffer) */
        hcb->cmd = vtobus(csio->cdb_io.cdb_bytes);
```

```
-}
hcb->cmdlen = csio->cdb_len;
```

####. #####, ### #### ####### ### ## ######### ## ### ### ## #### ########### ####, ######### ## ### ### #####. ##### ## ### ### ######### ## ### #### ########. ### ######## #### ## ## ##### ## ## #### ## ########:

```
int dir = (ccb_h->flags & CAM_DIR_MASK);

if (dir == CAM_DIR_NONE)
    goto end_data;
```

####, ### ### ######### ### ######## ## #######. ### #### ########## ### ## #### ## ###–### #### # ####### ###### ## ###### ## ####### ######. ## ### ####### #### #### ########## ## ###### ## #####. ## ### # ####### ######## ## ###### ### ### ## ###### ## ### ##### ### ### ######## #########. ### ######### ## ### ###### ### ################, ### #### ## ##### #### ####### ######## #####–#########. ### ########### ## ### #### ####### (###) ######## ### ### ####### ## ### ################### ######. ## #### ######### ## ### ######### ## ##–######## ## ######### #### # ########## #### ## ## ## ## ###### ### ###### ###_###_#######. ########, ## ##### #### ### ############## ####### ## ### #### ######## ## ### ### #### ###. ### ## ##### ### #### ### # ###### ########–##### ####### ###### #### ## ###########, ## ## ######## #### ## ###.

```
int rv;

initialize_hcb_for_data(hcb);

if((!(ccb_h->flags & CAM_SCATTER_VALID)) {
    -/* single buffer */
    if(!(ccb_h->flags & CAM_DATA_PHYS)) {
        rv = add_virtual_chunk(hcb, csio->data_ptr, csio->dxfer_len, dir);
        -}
    -} else {
        rv = add_physical_chunk(hcb, csio->data_ptr, csio->dxfer_len, dir);
    -}
-} else {
    int i;
    struct bus_dma_segment *segs;
    segs = (struct bus_dma_segment *)csio->data_ptr;

    if ((ccb_h->flags & CAM_SG_LIST_PHYS) -!= 0) {
        -/* The SG list pointer is physical */
        rv = setup_hcb_for_physical_sg_list(hcb, segs, csio->sglist_cnt);
    -} else if (!(ccb_h->flags & CAM_DATA_PHYS)) {
        -/* SG buffer pointers are virtual */
        for (i = 0; i < csio->sglist_cnt; i++) {
            rv = add_virtual_chunk(hcb, segs[i].ds_addr,
```

```
                segs[i].ds_len, dir);
            if (rv -!= CAM_REQ_CMP)
                break;
        -}
    -} else {
        -/* SG buffer pointers are physical */
        for (i = 0; i < csio->sglist_cnt; i++) {
            rv = add_physical_chunk(hcb, segs[i].ds_addr,
                segs[i].ds_len, dir);
            if (rv -!= CAM_REQ_CMP)
                break;
        -}
    -}
-}
if(rv -!= CAM_REQ_CMP) {
    -/* we expect that add_*_chunk() functions return CAM_REQ_CMP
     * if they added a chunk successfully, CAM_REQ_TOO_BIG if
     * the request is too big (too many bytes or too many chunks),
     * CAM_REQ_INVALID in case of other troubles
     */
    free_hcb_and_ccb_done(hcb, ccb, rv);
    return;
-}
end_data:
```

###:

```
if(ccb_h->flags & CAM_DIS_DISCONNECT)
    hcb_disable_disconnect(hcb);
```

###_###_######### ###### #### ## ###### ## ##, ## ####### ######### ####### ##### ## ### ### ######### #### ### #### ##.

#######, #### ### ### ## ### ######## ### ######, ### #### #### ## #### ## ### ######### ####### (## ####### #######).

```
ccb_h->timeout_ch = timeout(xxx_timeout, (caddr_t) hcb,
    (ccb_h->timeout * hz) -/ 1000); -/* convert milliseconds to ticks */
put_hcb_into_hardware_queue(hcb);
return;
```

###:

```
static void
free_hcb_and_ccb_done(struct xxx_hcb *hcb, union ccb *ccb, u_int32_t status)
{
    struct xxx_softc *softc = hcb->softc;

    ccb->ccb_h.ccb_hcb = 0;
    if(hcb -!= NULL) {
        untimeout(xxx_timeout, (caddr_t) hcb, ccb->ccb_h.timeout_ch);
```

```
        -/* we're about to free a hcb, so the shortage has ended */
        if(softc->flags & RESOURCE_SHORTAGE)  {
            softc->flags &= ~RESOURCE_SHORTAGE;
            status -|= CAM_RELEASE_SIMQ;
        -}
        free_hcb(hcb); -/* also removes hcb from any internal lists */
    -}
    ccb->ccb_h.status = status -|
        (ccb->ccb_h.status & ~(CAM_STATUS_MASK|CAM_SIM_QUEUED));
    xpt_done(ccb);
-}
```

- *XPT_RESET_DEV* # #### ### #### #### ###### ###### ####### ## # ######

  ##### ## ## #### ########### ## ### ###### ### ###### ### ### #### ########–### ######## ## ## ## ######_##. ######### ## ### ########## ######## # ####–#### ####### ##### #### #### ### ### ###_####_## ####### ### ## #########–## (### ###_####_## ####### ###########) ### #### ## ### ########## ## ### #### ########## ### ## ########### ########## ## #### #### ##### ####### ## ### ###### ## #### ####### ### ## #### ### ######### (### ###### ### ###–### ###_###_#######). #### ## ########## ## ### ####### ### ### ############ ############ ### #### ###### #### ## ####### (######## ## ### ######### ###–####).

  #### ### ### ####### ############ ### ### ###### ### #### ## #####, ## #### ##### ## ####### ###. ## #### ######## ### ## ########, ####### ###### ### ###–### ##### ####### ############## ## ### #### ###########.

- *XPT_RESET_BUS* # #### ### ##### ###### ## ### #### ###

  ## ######### ### ###### ## ### ###, ### #### ########### ######## ## ### #### ### ######### ## ### ###### ### #######.

  # ############ ############## ##### ###### ### #### ############ ### ### ### ####### ## ### ### ### ###### ### ###### ###_###_###.

  ### ###### ############## ##### ## ######## ######## ##### ### #### ### (###–##### #### ##### ### #### ##########) ### #### ### ### #### ##### #########, #### ##### ## ### ######## ##### ### ##### ##### ###########, ## #### #### ### ###### ###_####_###_#####. ####:

```
int targ, lun;
struct xxx_hcb *h, *hh;
struct ccb_trans_settings neg;
struct cam_path *path;

-/* The SCSI bus reset may take a long time, in this case its completion
 * should be checked by interrupt or timeout. But for simplicity
 * we assume here that it is really fast.
```

```
 */
reset_scsi_bus(softc);

-/* drop all enqueued CCBs */
for(h = softc->first_queued_hcb; h -!= NULL; h = hh) {
   hh = h->next;
   free_hcb_and_ccb_done(h, h->ccb, CAM_SCSI_BUS_RESET);
-}

-/* the clean values of negotiations to report */
neg.bus_width = 8;
neg.sync_period = neg.sync_offset = 0;
neg.valid = (CCB_TRANS_BUS_WIDTH_VALID
   -| CCB_TRANS_SYNC_RATE_VALID -| CCB_TRANS_SYNC_OFFSET_VALID);

-/* drop all disconnected CCBs and clean negotiations  */
for(targ=0; targ <= OUR_MAX_SUPPORTED_TARGET; targ++) {
   clean_negotiations(softc, targ);

   -/* report the event if possible */
   if(xpt_create_path(&path, -/*periph*/NULL,
         cam_sim_path(sim), targ,
         CAM_LUN_WILDCARD) == CAM_REQ_CMP) {
      xpt_async(AC_TRANSFER_NEG, path, &neg);
      xpt_free_path(path);
   -}

   for(lun=0; lun <= OUR_MAX_SUPPORTED_LUN; lun++)
      for(h = softc->first_discon_hcb[targ][lun]; h -!= NULL; h = hh) {
         hh=h->next;
         free_hcb_and_ccb_done(h, h->ccb, CAM_SCSI_BUS_RESET);
      -}
-}

ccb->ccb_h.status = CAM_REQ_CMP;
xpt_done(ccb);

-/* report the event */
xpt_async(AC_BUS_RESET, softc->wpath, NULL);
return;
```

############ ### #### ### ##### ## # ######## ### ## # #### #### ####### ## ##### ## ####### ## ### ####### ######## ## # #### ###### ## ### ###### ## #####.

- *XPT_ABORT* # ##### ### ######### ###

###_##### #### ## ### ##### ###. ### #### ######## ##### ## ## ##:

abort_ccb # ####### ## ### ### ## ## #######

###_##_#####. #### ## #### ### #### ### ## ######### ######### #### ####, ###### ###_##_##### ## ### ####.

########. ##### ##### #### ##### ####### ## # #### ###########:

```
struct ccb *abort_ccb;
abort_ccb = ccb->cab.abort_ccb;

if(abort_ccb->ccb_h.func_code -!= XPT_SCSI_IO) {
    ccb->ccb_h.status = CAM_UA_ABORT;
    xpt_done(ccb);
    return;
-}
```

#####. #### ### ## #### ## ####### ### #### ## ### ### ######## ####### ###### ## ###### ### ### ########## #### #### ###:

```
struct xxx_hcb *hcb, *h;

hcb = NULL;

-/* We assume that softc->first_hcb is the head of the list of all
 * HCBs associated with this bus, including those enqueued for
 * processing, being processed by hardware and disconnected ones.
 */
for(h = softc->first_hcb; h -!= NULL; h = h->next) {
    if(h->ccb == abort_ccb) {
        hcb = h;
        break;
    -}
-}

if(hcb == NULL) {
    -/* no such CCB in our queue */
    ccb->ccb_h.status = CAM_PATH_INVALID;
    xpt_done(ccb);
    return;
-}

hcb=found_hcb;
```

###. ## ### ## ###### ###–#### ## ### ##### ####### ## ## #### ## ### #### ###, ##### ########### ##### ###, ## ############ ### ####### ### ### ###### ## ### #######, ## ######## ######### ## ######## ### ### ### ###### ## #### ## ########. ## #### #### #### ## ## ### ### ## ### ##### #### ######## ## #### ### ### ## ##### #######,

##.

```
int hstatus;

-/* shown as a function, in case special action is needed to make
 * this flag visible to hardware
 */
set_hcb_flags(hcb, HCB_BEING_ABORTED);

abort_again:

hstatus = get_hcb_status(hcb);
switch(hstatus) {
case HCB_SITTING_IN_QUEUE:
    remove_hcb_from_hardware_queue(hcb);
    -/* FALLTHROUGH */
case HCB_COMPLETED:
    -/* this is an easy case */
    free_hcb_and_ccb_done(hcb, abort_ccb, CAM_REQ_ABORTED);
    break;
```

########. ### #### ########## ##### ### ### #### ######### ###### ### #### ### ###### ######## ## ## #### ## ##### #######. ## #### ##### ### ####### ## #### #### #### ### ###### ## ### ######## #######. ## ### ####### ##### ### ### ####### ## #### ########## #### #### 10 ####### ### ####### ####### ##### ## ##### ### ##### ### ##### #### ###. ####### ### ####### #### ## ####### ## #### ########## #### ## ### #### ###### ### ##### ####### ### ## ############ #########, ### #### ### ####### ### ## ####### (### ### #### ## ## #### ###).

```
case HCB_BEING_TRANSFERRED:
    untimeout(xxx_timeout, (caddr_t) hcb, abort_ccb->ccb_h.timeout_ch);
    abort_ccb->ccb_h.timeout_ch =
        timeout(xxx_timeout, (caddr_t) hcb, 10 * hz);
    abort_ccb->ccb_h.status = CAM_REQ_ABORTED;
    -/* ask the controller to abort that HCB, then generate
     * an interrupt and stop
     */
    if(signal_hardware_to_abort_hcb_and_stop(hcb) < 0) {
        -/* oops, we missed the race with hardware, this transaction
         * got off the bus before we aborted it, try again */
        goto abort_again;
    -}

    break;
```

#####. ##### ### ####### ### ###### ### ##### ####### ## ## #########.

```
case HCB_DISCONNECTED:
    untimeout(xxx_timeout, (caddr_t) hcb, abort_ccb->ccb_h.timeout_ch);
    abort_ccb->ccb_h.timeout_ch =
        timeout(xxx_timeout, (caddr_t) hcb, 10 * hz);
    put_abort_message_into_hcb(hcb);
    put_hcb_at_the_front_of_hardware_queue(hcb);
    break;
-}
ccb->ccb_h.status = CAM_REQ_CMP;
xpt_done(ccb);
return;
```

#######, ######## ##### ## ### #### #####. ####### ### ##### ####### ###### ### ### ####### ############ ## # ### ## #### ## #### ### ### ##### ###### ############ ## #### ### ## #######. #### ###### ## #### ## ### ######### #######, ##### ### ########### #### #######.

############ ### ### ##### ## # ######## ### ## ##### # #### ####, #### ######## ### ## ####### ## ## #/# ########### ##### ###. ### #### ########## ##### ## #### ### ##### ### ########### ##### ###### ### ###### ###_###_####### ### ### ##### ### #######. #### ### #### ###_##### ##### ## #####, #### ####:

```
case XPT_ABORT:
    struct ccb *abort_ccb;
    abort_ccb = ccb->cab.abort_ccb;

    if(abort_ccb->ccb_h.func_code -!= XPT_SCSI_IO) {
        ccb->ccb_h.status = CAM_UA_ABORT;
        xpt_done(ccb);
        return;
    -}
    if(xxx_abort_ccb(abort_ccb, CAM_REQ_ABORTED) < 0)
        -/* no such CCB in our queue */
        ccb->ccb_h.status = CAM_PATH_INVALID;
    else
        ccb->ccb_h.status = CAM_REQ_CMP;
    xpt_done(ccb);
    return;
```

- *XPT_SET_TRAN_SETTINGS* # ########## ### ###### ## #### ######## ########

  ### ######### ### ########### ## ### ######## ####### ###_#####_####### #### ## ### ##### ###:

  - *valid* # # ####### ####### ##### ######## ###### ## #######:

- *CCB_TRANS_SYNC_RATE_VALID* # ########### ######## ####
- *CCB_TRANS_SYNC_OFFSET_VALID* # ########### ######
- *CCB_TRANS_BUS_WIDTH_VALID* # ### #####
- *CCB_TRANS_DISC_VALID* # ### ######/####### #############
- *CCB_TRANS_TQ_VALID* # ### ######/####### ###### #######
- *flags* # ######## ## ### #####, ###### ######### ### ############## ## #### ##########. ### ###### ######### ###:
  - *CCB_TRANS_DISC_ENB* # ###### #############
  - *CCB_TRANS_TAG_ENB* # ###### ###### #######
- ### ############## ###:
  - *CCB_TRANS_CURRENT_SETTINGS* # ###### ### ####### ############
  - *CCB_TRANS_USER_SETTINGS* # ######## ### ####### #### ###### ####_######, ####_###### # ################, ## ####_######==0 #### ### ############ #### ## ######### ###_##### # ### #####, ## #### (### #####)

#########, ### #### ######## ### ### ####### ########. ### #### ######## ### ### ###### #### #### ## ### ### ###–####, #### ## ###### #### # ##### ## ###### ##### ### ##### ###### ### ##### (### ##### ######) ### ##### ##### ### ##########. ####### ### #### ########## #### ### ##### ############## ## ### ######## #####. ### ##### ### ##### ###–####### #### # ########### ## #### ##### ### ### ###### ###### #### ### #### ##########, ## ## ## ########### ### ### ########.

###, ## ### #### ####, #######. ######## #### ##### #### ### ########## #### ## ############# ## ### #### ########. #####, ##### #### ####### ######### ### ### ######## ## ## ###### ## ### ######, #### #### ### #### ## ### ####### #### ## ############. #### #### #### ## ####### ## ###### ############ ## ### #### ##########: ### #######, ## ### #### ########## ### 8#### ### ### ### ####### #### ## ### 16#### #### ######### #### ######### #### ## ######## ######### ## 8#### ######### ###### ####### ## ## ### ######.

###.

3 #### ## ########## (### ##### ### ########### ########) ##########:

- *user* # ### #### ###, ## #####
- *current* # ##### ######## ## ######
- *goal* # ##### ######### ## ####### ## ### ######### ##########

####:

```
struct ccb_trans_settings *cts;
int targ, lun;
int flags;

cts = &ccb->cts;
targ = ccb_h->target_id;
lun = ccb_h->target_lun;
flags = cts->flags;
if(flags & CCB_TRANS_USER_SETTINGS) {
    if(flags & CCB_TRANS_SYNC_RATE_VALID)
        softc->user_sync_period[targ] = cts->sync_period;
    if(flags & CCB_TRANS_SYNC_OFFSET_VALID)
        softc->user_sync_offset[targ] = cts->sync_offset;
    if(flags & CCB_TRANS_BUS_WIDTH_VALID)
        softc->user_bus_width[targ] = cts->bus_width;

    if(flags & CCB_TRANS_DISC_VALID) {
        softc->user_tflags[targ][lun] &= ~CCB_TRANS_DISC_ENB;
        softc->user_tflags[targ][lun] -|= flags & CCB_TRANS_DISC_ENB;
    -}
    if(flags & CCB_TRANS_TQ_VALID) {
        softc->user_tflags[targ][lun] &= ~CCB_TRANS_TQ_ENB;
        softc->user_tflags[targ][lun] -|= flags & CCB_TRANS_TQ_ENB;
    -}
-}
if(flags & CCB_TRANS_CURRENT_SETTINGS) {
    if(flags & CCB_TRANS_SYNC_RATE_VALID)
        softc->goal_sync_period[targ] =
            max(cts->sync_period, OUR_MIN_SUPPORTED_PERIOD);
    if(flags & CCB_TRANS_SYNC_OFFSET_VALID)
        softc->goal_sync_offset[targ] =
            min(cts->sync_offset, OUR_MAX_SUPPORTED_OFFSET);
    if(flags & CCB_TRANS_BUS_WIDTH_VALID)
        softc->goal_bus_width[targ] = min(cts->bus_width, OUR_BUS_WIDTH);

    if(flags & CCB_TRANS_DISC_VALID) {
        softc->current_tflags[targ][lun] &= ~CCB_TRANS_DISC_ENB;
        softc->current_tflags[targ][lun] -|= flags & CCB_TRANS_DISC_ENB;
    -}
    if(flags & CCB_TRANS_TQ_VALID) {
        softc->current_tflags[targ][lun] &= ~CCB_TRANS_TQ_ENB;
        softc->current_tflags[targ][lun] -|= flags & CCB_TRANS_TQ_ENB;
    -}
```

```
-}
ccb->ccb_h.status = CAM_REQ_CMP;
xpt_done(ccb);
return;
```

#/# ####### #### ## ######### ## #### ##### ## ## ### ## ### #########, ### ####### ## ####### ### ######## ######_##########(###). ## ### ## ########### #### ####:

```
int
target_negotiated(struct xxx_hcb *hcb)
{
   struct softc *softc = hcb->softc;
   int targ = hcb->targ;

   if( softc->current_sync_period[targ] -!= softc->goal_sync_period[targ]
   -|| softc->current_sync_offset[targ] -!= softc->goal_sync_offset[targ]
   -|| softc->current_bus_width[targ] -!= softc->goal_bus_width[targ] -)
      return 0; -/* FALSE */
   else
      return 1; -/* TRUE */
-}
```

##########, ## ### ###### #/# ############ ### ####### ### #### ########## ##### ## ### #### ### target_negotiated() ##### ###### ####. #### ### #### ## ########### (## xxx_attach()) ### ####### ########### ###### #### ## ########### ## ###### ############ ####, ### #### ### ####### ###### #### ## ########### ## ### ####### ###### ######### ## ##########.

XPT_GET_TRAN_SETTINGS # ### ###### ## #### ######## ########

###_###_####_########. #### ## ### ### ##– ###### ####### ###_#####_####### #### #### #### ## ######### ## ### ##### ###_#####_#######_######## ## ###_#####_####_######## (## #### ### ### #### ### ######## ####### ###### ### ####### ########). ### ### ### #### ## ### ##### #####.

XPT_CALC_GEOMETRY # ######### ####### (####) ######## ## ### ####

###_####_######## #### ## ### ##### ###:

- *block_size* # #####, ##### (#.#.# ######) #### ## #####
- *volume_size* # #####, ###### #### ## #####
- *cylinders* # ######, ####### #########

- *heads* # ######, ####### #####

- *secs_per_track* # ######, ####### ####### ### #####

####. ### ####### ########### ####### ##### #### ### ###7### ###### ##:

```
struct    ccb_calc_geometry *ccg;
u_int32_t size_mb;
u_int32_t secs_per_cylinder;
int   extended;

ccg = &ccb->ccg;
size_mb = ccg->volume_size
    -/ ((1024L * 1024L) -/ ccg->block_size);
extended = check_cards_EEPROM_for_extended_geometry(softc);

if (size_mb > 1024 && extended) {
    ccg->heads = 255;
    ccg->secs_per_track = 63;
-} else {
    ccg->heads = 64;
    ccg->secs_per_track = 32;
-}
secs_per_cylinder = ccg->heads * ccg->secs_per_track;
ccg->cylinders = ccg->volume_size -/ secs_per_cylinder;
ccb->ccb_h.status = CAM_REQ_CMP;
xpt_done(ccb);
return;
```

####, ### ##### ########### ####### ## ### ###### ## ### ########## ####. ## #### ######## ## ### ### ### ######### ############ #### ## ###### #### #### ###### ######## ## ####### ##### ## 1. ##### ####### ########## ###:

```
128 heads, 63 sectors -- Symbios controllers
16 heads, 63 sectors -- old controllers
```

###–####, ### ####### # ########### ## ####### 875/895 #### ### ####### #### ### #### ######## 128/63 ##### ##### ## ### 255/63 ##### # #### ##### ## #### ##–####.

- *XPT_PATH_INQ* # #### #######, ## ##### ##### ### ### ### ###### ### #### ###–####### (#### ##### ## ### # #### ### #######) ##########

###_####### #### ## ### ##### ###:

- #######_### # ### ### ###### ####### ######, ### ### ####### ### 1
- ###_####### # ####### ## ######## ######### ## ### ##########:
- ##_###_#### # ######### ### ####### (######### #### ####3?)
- ##_####_32 # ######## 32 ### #### ####
- ##_####_16 # ######## 16 ### #### ####
- ##_####_#### # ### ######### ########### ######## ####
- ##_######_### # ######## ###### ########
- ##_###_#### # ######## ###### ########
- ##_####_### # ######## #### ##### ########### (#### ##### ### #### ##### ### ######## ######### ###### # #### ###)
- ######_#### # ##### ### ###### #### #######, 0 ## ###########
- ###_#### # ############# ########## ########:
- ###_######## # ### ##### #### #### ## ## ### ##
- ###_######## # ######### ####### ### ######## ## ####
- ###_########### # ######### #### ### #########
- ###_########## # #### ### ######## ####### ### #####
- ###_###_### # ########## ### ###### #####, ######### ####### ## #######–####, ### ## ###### ### ## 0
- #####_##### # ############ #####, ###### ###
- ###_###### # ####### ######### ###### ## (7 ### 8#### ###, 15 ### 16#### ###, 127 ### ##### #######)
- ###_### # ####### ######### ### ## (7 ### ##### #### ###########, 63 ### ###–## ####)
- #####_##### # ####### ## ######### ##### #######, ###### ###
- #####_## # ####### #### ## ## ### #########, ###### ###

- ####_###### # ### ########## #### ######, ###_###_####(###)
- ###_## # ### ### ######, ###_###_###(###)
- #########_## # ### #### ## ## ### ########## ######
- ####_########_##### # ####### ######## ##### ## ##/# ### ############ ###–### #########, ###### ## 3300 ### ####
- ###_### # ### ######## ###### ##, # ############### ###### ## ####### ###### ###_##### ######### ### ########### ####
- ###_### # #### ############ ###### ##, # ############### ###### ## ####### ###### ###_##### ######### ### ########### ####
- ###_#### # ###### ###### ####, # ############### ###### ## ####### ###### ###_##### ######### ### ########### ####, ##### ## ###_###_####(###)

#######, ####:

```
strncpy(cpi->dev_name, cam_sim_name(sim), DEV_IDLEN);
```

###_###_### ### #### ### ### ## ####.

12.3.

static void **xxx_poll** (*struct cam_sim *sim*);

*struct cam_sim *sim* ;

######–### ## ### ########### (### #######, #### ### ###### ### ####### ### ## ######## ### ###### ####). ### ### ######### #### ### ###### ######### ##### ###### ####–### ### #### #######. ## ### ## ##### ## ## ## ## #### ### ######### ####### (## ### ##### ### ######, ### #### ####### ### ## ##### ### #### ###### ### ### #####–#### ####### ##### #### #### ### #### #######). ### ###### ##### # ######## ####–#### ####? ####### ## ######### ####### ###########. ### xxx_poll ####### #### ### ###### ###_### ####### ## ### ######## #### ### ### ######### ####### ## ###### ########## #### ####### ## ### ###### xxx_softc ### ### ### ######### ####### #### #### ### ###### #### ######. ## ### #### ####### ##### ######## #### ##:

```
static void
xxx_poll(struct cam_sim *sim)
{
    xxx_intr((struct xxx_softc *)cam_sim_softc(sim)); -/* for PCI device */
}
```

##

```
static void
xxx_poll(struct cam_sim *sim)
{
    xxx_intr(cam_sim_unit(sim)); -/* for ISA device */
}
```

12.4. ############

#######.

```
static void
ahc_async(void *callback_arg, u_int32_t code, struct cam_path *path, void *arg)
```

- ########_### # ### ##### ######## #### ########### ### ########
- #### # ########## ### #### ## #####
- #### # ########## ### ####### ## ##### ### ##### #######
- ### # ############## ########

############# ### # ###### #### ## #####, ##_####_######, ##### ####:

```
    struct xxx_softc *softc;
    struct cam_sim *sim;
    int targ;
    struct ccb_trans_settings neg;

    sim = (struct cam_sim *)callback_arg;
    softc = (struct xxx_softc *)cam_sim_softc(sim);
    switch (code) {
    case AC_LOST_DEVICE:
        targ = xpt_path_target_id(path);
        if(targ <= OUR_MAX_SUPPORTED_TARGET) {
            clean_negotiations(softc, targ);
            -/* send indication to CAM */
            neg.bus_width = 8;
            neg.sync_period = neg.sync_offset = 0;
            neg.valid = (CCB_TRANS_BUS_WIDTH_VALID
                -| CCB_TRANS_SYNC_RATE_VALID -| CCB_TRANS_SYNC_OFFSET_VALID);
            xpt_async(AC_TRANSFER_NEG, path, &neg);
        -}
        break;
    default:
        break;
    -}
```

12.5.

(###, ### ### ## ##) ## ##### ### #### ########## ## #########.

######. ## splcam() ###### ## #### ## ### ###### ## ########### ######## ####### ### ##–####### ####### ### ### #### ## ### ###### (### # #################### ###### ###### ### ### #### ########### ### ## ###### #### #### ####). ### ########### ## #### ######## ####### ####### ### ######## ## ###############. ### #### #### #### ### ###### ####. # ############# ######## ## ## ### splcam() ## ### ##### ## ### ##### ######## ### ##### ## ## ###### #### ########## #### ## ### ### ####–#### #######. ## #### #### #### ### ######### ##### #### ## ###### ######## # ####### ######## ### ## #######, ####:

```
static void
xxx_action(struct cam_sim *sim, union ccb *ccb)
{
   int s;
   s = splcam();
   xxx_action1(sim, ccb);
   splx(s);
-}

static void
xxx_action1(struct cam_sim *sim, union ccb *ccb)
{
   -... process the request -...
-}
```

###########. ## ### ##### #### ### ######### ## ### spl() ###### #### ###### #### ########, ## #### ###### ## #### ######## ######## ### ### ## #### ######.

########. ## ######## ### ### ## ######### ##########.

#####, ## ##### ## # #### ##### ### ########### ## ### ### (######## ###### ## ####### #### ########## ## ### #### #### ###). ## ## ## #### ### ### ######## ### ############ ########, ###### ### ###### ### ############# ### #### ##########. ## ## ######### #### ###### #### ############## ### ########## #### ### ##### ####### ##### ## #### ### ########### ## ### #### #### ### ##### ######### ##–#### #######. ### #### ## ##### ########## #####/#### ##### ## ####### ## ### #### #####, ### ## #### ######## #### #### ####### ##### ###### ## ### #### ### ## ##### ### ###### ## ### ########### #### ### #### #######.

```
int fatal=0;
struct ccb_trans_settings neg;
struct cam_path *path;

if( detected_scsi_reset(softc)
-|| (fatal = detected_fatal_controller_error(softc)) -) {
  int targ, lun;
  struct xxx_hcb *h, *hh;

  -/* drop all enqueued CCBs */
  for(h = softc->first_queued_hcb; h -!= NULL; h = hh) {
    hh = h->next;
    free_hcb_and_ccb_done(h, h->ccb, CAM_SCSI_BUS_RESET);
  -}

  -/* the clean values of negotiations to report */
  neg.bus_width = 8;
  neg.sync_period = neg.sync_offset = 0;
  neg.valid = (CCB_TRANS_BUS_WIDTH_VALID
    -| CCB_TRANS_SYNC_RATE_VALID -| CCB_TRANS_SYNC_OFFSET_VALID);

  -/* drop all disconnected CCBs and clean negotiations  */
  for(targ=0; targ <= OUR_MAX_SUPPORTED_TARGET; targ++) {
    clean_negotiations(softc, targ);

    -/* report the event if possible */
    if(xpt_create_path(&path, -/*periph*/NULL,
        cam_sim_path(sim), targ,
        CAM_LUN_WILDCARD) == CAM_REQ_CMP) {
      xpt_async(AC_TRANSFER_NEG, path, &neg);
      xpt_free_path(path);
    -}

    for(lun=0; lun <= OUR_MAX_SUPPORTED_LUN; lun++)
      for(h = softc->first_discon_hcb[targ][lun]; h -!= NULL; h = hh) {
        hh=h->next;
        if(fatal)
          free_hcb_and_ccb_done(h, h->ccb, CAM_UNREC_HBA_ERROR);
        else
          free_hcb_and_ccb_done(h, h->ccb, CAM_SCSI_BUS_RESET);
      -}
  -}

  -/* report the event */
  xpt_async(AC_BUS_RESET, softc->wpath, NULL);

  -/* re-initialization may take a lot of time, in such case
   * its completion should be signaled by another interrupt or
   * checked on timeout -- but for simplicity we assume here that
   * it is really fast
   */
  if(!fatal) {
    reinitialize_controller_without_scsi_reset(softc);
```

```
    -} else {
        reinitialize_controller_with_scsi_reset(softc);
    -}
    schedule_next_hcb(softc);
    return;
-}
```

####–##### ### ######## ## ### ####### ######## ####### #####. ######### ## ### ####–#### ##### ### ## ##### ############### ######, ## #### ## ### ######## #### ####. #### ## ####### #### ######## ## #### ###:

```
struct xxx_hcb *hcb, *h, *hh;
int hcb_status, scsi_status;
int ccb_status;
int targ;
int lun_to_freeze;

hcb = get_current_hcb(softc);
if(hcb == NULL) {
    -/* either stray interrupt or something went very wrong
     * or this is something hardware-dependent
     */
    handle as necessary;
    return;
-}

targ = hcb->target;
hcb_status = get_status_of_current_hcb(softc);
```

######.

```
if(hcb_status == COMPLETED) {
    scsi_status = get_completion_status(hcb);
```

###.

```
    if(hcb->flags & DOING_AUTOSENSE) {
        if(scsi_status == GOOD) { -/* autosense was successful */
            hcb->ccb->ccb_h.status -|= CAM_AUTOSNS_VALID;
            free_hcb_and_ccb_done(hcb, hcb->ccb, CAM_SCSI_STATUS_ERROR);
        -} else {
    autosense_failed:
            free_hcb_and_ccb_done(hcb, hcb->ccb, CAM_AUTOSENSE_FAIL);
        -}
        schedule_next_hcb(softc);
        return;
    -}
```

#########, ### #### ######### ## #######. ## ##–######## ## ### ######## ### #### ### ### ### ####### ### ###### #### ##### #### #### ### ####### ##### ####### ## ####### #### ####.

```
    hcb->ccb->csio.scsi_status = scsi_status;
    calculate_residue(hcb);

    if( (hcb->ccb->ccb_h.flags & CAM_DIS_AUTOSENSE)==0
    && ( scsi_status == CHECK_CONDITION
            -|| scsi_status == COMMAND_TERMINATED) -) {
        -/* start auto-SENSE */
        hcb->flags -|= DOING_AUTOSENSE;
        setup_autosense_command_in_hcb(hcb);
        restart_current_hcb(softc);
        return;
    -}
    if(scsi_status == GOOD)
        free_hcb_and_ccb_done(hcb, hcb->ccb, CAM_REQ_CMP);
    else
        free_hcb_and_ccb_done(hcb, hcb->ccb, CAM_SCSI_STATUS_ERROR);
    schedule_next_hcb(softc);
    return;
-}
```

######: ########### ######## ######## #### # #### ###### (## ###### ## ### ########### ####### ## ## ######## ######–####) ## ### ###### ## ###### ## ######### (####### ### ########### ######## ## #### ### ###### ####).

```
switch(hcb_status) {
case TARGET_REJECTED_WIDE_NEG:
    -/* revert to 8-bit bus */
    softc->current_bus_width[targ] = softc->goal_bus_width[targ] = 8;
    -/* report the event */
    neg.bus_width = 8;
    neg.valid = CCB_TRANS_BUS_WIDTH_VALID;
    xpt_async(AC_TRANSFER_NEG, hcb->ccb.ccb_h.path_id, &neg);
    continue_current_hcb(softc);
    return;
case TARGET_ANSWERED_WIDE_NEG:
    {
        int wd;

        wd = get_target_bus_width_request(softc);
        if(wd <= softc->goal_bus_width[targ]) {
            -/* answer is acceptable */
            softc->current_bus_width[targ] =
            softc->goal_bus_width[targ] = neg.bus_width = wd;

            -/* report the event */
            neg.valid = CCB_TRANS_BUS_WIDTH_VALID;
            xpt_async(AC_TRANSFER_NEG, hcb->ccb.ccb_h.path_id, &neg);
```

```
        -} else {
            prepare_reject_message(hcb);
        -}
    -}
    continue_current_hcb(softc);
    return;
case TARGET_REQUESTED_WIDE_NEG:
    {
        int wd;

        wd = get_target_bus_width_request(softc);
        wd = min (wd, OUR_BUS_WIDTH);
        wd = min (wd, softc->user_bus_width[targ]);

        if(wd -!= softc->current_bus_width[targ]) {
            -/* the bus width has changed */
            softc->current_bus_width[targ] =
            softc->goal_bus_width[targ] = neg.bus_width = wd;

            -/* report the event */
            neg.valid = CCB_TRANS_BUS_WIDTH_VALID;
            xpt_async(AC_TRANSFER_NEG, hcb->ccb.ccb_h.path_id, &neg);
        -}
        prepare_width_nego_rsponse(hcb, wd);
    -}
    continue_current_hcb(softc);
    return;
-}
```

######. ######### ## #### ###### ## ### ####### #####.

```
if(hcb->flags & DOING_AUTOSENSE)
    goto autosense_failed;

switch(hcb_status) {
```

##########. ##### ## ########## ###### ##### ## ##### ## ### ###### ##### ####### ### ######## ## ##### #####.

```
case UNEXPECTED_DISCONNECT:
    if(requested_abort(hcb)) {
        -/* abort affects all commands on that target+LUN, so
         * mark all disconnected HCBs on that target+LUN as aborted too
         */
        for(h = softc->first_discon_hcb[hcb->target][hcb->lun];
                h -!= NULL; h = hh) {
            hh=h->next;
            free_hcb_and_ccb_done(h, h->ccb, CAM_REQ_ABORTED);
        -}
        ccb_status = CAM_REQ_ABORTED;
    -} else if(requested_bus_device_reset(hcb)) {
        int lun;
```

```
    -/* reset affects all commands on that target, so
     * mark all disconnected HCBs on that target+LUN as reset
     */

    for(lun=0; lun <= OUR_MAX_SUPPORTED_LUN; lun++)
        for(h = softc->first_discon_hcb[hcb->target][lun];
                h -!= NULL; h = hh) {
            hh=h->next;
            free_hcb_and_ccb_done(h, h->ccb, CAM_SCSI_BUS_RESET);
        -}

    -/* send event */
    xpt_async(AC_SENT_BDR, hcb->ccb->ccb_h.path_id, NULL);

    -/* this was the CAM_RESET_DEV request itself, it is completed */
    ccb_status = CAM_REQ_CMP;
-} else {
    calculate_residue(hcb);
    ccb_status = CAM_UNEXP_BUSFREE;
    -/* request the further code to freeze the queue */
    hcb->ccb->ccb_h.status -|= CAM_DEV_QFRZN;
    lun_to_freeze = hcb->lun;
-}
break;
```

###:

```
case TAGS_REJECTED:
    -/* report the event */
    neg.flags = 0 & ~CCB_TRANS_TAG_ENB;
    neg.valid = CCB_TRANS_TQ_VALID;
    xpt_async(AC_TRANSFER_NEG, hcb->ccb.ccb_h.path_id, &neg);

    ccb_status = CAM_MSG_REJECT_REC;
    -/* request the further code to freeze the queue */
    hcb->ccb->ccb_h.status -|= CAM_DEV_QFRZN;
    lun_to_freeze = hcb->lun;
    break;
```

##########, #### ########## ######### ####### ## ####### ### ### ######:

```
case SELECTION_TIMEOUT:
    ccb_status = CAM_SEL_TIMEOUT;
    -/* request the further code to freeze the queue */
    hcb->ccb->ccb_h.status -|= CAM_DEV_QFRZN;
    lun_to_freeze = CAM_LUN_WILDCARD;
    break;
case PARITY_ERROR:
    ccb_status = CAM_UNCOR_PARITY;
    break;
```

```
case DATA_OVERRUN:
case ODD_WIDE_TRANSFER:
    ccb_status = CAM_DATA_RUN_ERR;
    break;
default:
    -/* all other errors are handled in a generic way */
    ccb_status = CAM_REQ_CMP_ERR;
    -/* request the further code to freeze the queue */
    hcb->ccb->ccb_h.status -|= CAM_DEV_QFRZN;
    lun_to_freeze = CAM_LUN_WILDCARD;
    break;
-}
```

##:

```
if(hcb->ccb->ccb_h.status & CAM_DEV_QFRZN) {
    -/* freeze the queue */
    xpt_freeze_devq(ccb->ccb_h.path, -/*count*/1);

    -/* re-queue all commands for this target/LUN back to CAM */

    for(h = softc->first_queued_hcb; h -!= NULL; h = hh) {
        hh = h->next;

        if(targ == h->targ
        && (lun_to_freeze == CAM_LUN_WILDCARD -|| lun_to_freeze == h->lun) -)
            free_hcb_and_ccb_done(h, h->ccb, CAM_REQUEUE_REQ);
    -}
-}
free_hcb_and_ccb_done(hcb, hcb->ccb, ccb_status);
schedule_next_hcb(softc);
return;
```

#########.

12.6. ######

#/# ####### #### ###### ### ## #####. ### ###### ## ##### ### ## ######## ## ### ### ###### #### ##### ######. ######## ## ### ### ###### #######–### #### ########. ### ############ #### ## ### ####### ## ########### #######–## ### ### ####### ##### ##########:

- *CAM_RESRC_UNAVAIL* # #### ######## ## ########### ########### ### ### ### ###### ###### ######## ## ##### #### ## #### ###### #########. ## ####### ## #### ######## ##### ## #### ################ ######## ######## ### ##### ### ########## #### ### ######## ## ######### #### ## ####### #########.

- *CAM_UNCOR_PARITY* # ########### ###### ##### ########
- *CAM_DATA_RUN_ERR* # #### ####### ## ########## #### ##### (##### ## ###–## ######### #### ######### ## ###_###_####) ## ### ######## ###### ### #### ########
- *CAM_SEL_TIMEOUT* # ######### ####### ######## (###### #### ### #######)
- *CAM_CMD_TIMEOUT* # ####### ####### ######## (### ####### ######## ###)
- *CAM_SCSI_STATUS_ERROR* # ### ###### ######## #####
- *CAM_AUTOSENSE_FAIL* # ### ###### ######## ##### ### ### ####### ##### ###–#### ######
- *CAM_MSG_REJECT_REC* # ####### ###### ####### ### ########
- *CAM_SCSI_BUS_RESET* # ######## #### ### #####
- *CAM_REQ_CMP_ERR* # ############ #### ##### ######## ## ######### #### ## ##### ## #### # ####### ##### ## ####### ###### ## ### #########
- *CAM_UNEXP_BUSFREE* # ########## ########## ########
- *CAM_BDR_SENT* # ### ###### ##### ####### ### #### ## ### ######
- *CAM_UNREC_HBA_ERROR* # ############# #### ### ####### #####
- *CAM_REQ_TOO_BIG* # ### ####### ### ### ##### ### #### ##########
- *CAM_REQUEUE_REQ* # #### ####### ###### ## ######### ## ######## #####–###### ########. #### ######### ###### #### ### ### ########## ## ##### #### ###### ###### ### ##### ### #### ##### ##### ###### ######## ### ### ###### ## ### ### ##### #### #### ### ### #####. ####### ##### ## #### ###### ### ##–####### ########, ####### ######## ### ##### #### ##########. ## #### ##### ### ########### ####### ####### ### ###### ########## ### #####, ### ### ### ##### ######## ##### #### ### ## #### ## ### ### ### ### #########.
- *CAM_LUN_INVALID* # ### ### ## ## ### ####### ## ### ######### ## ### #### ##########
- *CAM_TID_INVALID* # ### ###### ## ## ### ####### ## ### ######### ## ### #### ##########

12.7. #######

#######, #### #### #### ## ###_##### #######. ### #### ########## ## #### ### ######## ######

####### ####### ###### ## ###_###_####### ####### ## ###_###_####### (#### ## ### ############## ## ### ##### ###### ## #### ## # ########). ### ##### ## ### #### ######## #######: #### ## ### ##### ####### ###### #### ### #####? ## #### #### ### #### ### ###### ## #####, #### #### #### ## ###_#####_### ####### (### ### #### ##### ############ ## ## # ######## ###### #### #### ###### ####### #### ###). #### ## ###### ##### ### ##### #### ### ## # ###### ##### ####### ### #####. ## ##### ### ### ####### ######## ##### #### ####:

```
static void
xxx_timeout(void *arg)
{
   struct xxx_hcb *hcb = (struct xxx_hcb *)arg;
   struct xxx_softc *softc;
   struct ccb_hdr *ccb_h;

   softc = hcb->softc;
   ccb_h = &hcb->ccb->ccb_h;

   if(hcb->flags & HCB_BEING_ABORTED
   -|| ccb_h->func_code == XPT_RESET_DEV) {
      xxx_reset_bus(softc);
   -} else {
      xxx_abort_ccb(hcb->ccb, CAM_CMD_TIMEOUT);
   -}
}
```

######/### ### ####### ###. ## ##### ####### # ########, ###### ## ###### #### #### ###### ###_###_####### ## ###_###_#######? ### ####### ####### ### ###_###_#######. #### ##### ####### ####### ## ### ####### ### ##### ### #### ######## ######### ###### ### ## ######### ## ### ######, ## ## #### ##### ### ## ######### #### ##### #### ### ## ##########.

13. ###

Written by Nick Hibma.
Modifications for Handbook made by Murray Stokely.

13.1.

(###) ## # ### ### ## ######### ####### ## ######## ###–######. ### ### ############ ######## ####### ############# ### ### #### ##–####### ## # ######## ## ####### ######## ####### ### ######### #### #######–#### #### ### ####. ### ####### ## ######## ## ### ####### ## ######## ### ## ######### ######### ## ### ######## ##### ###. ####### ############ ## ### #### #### #### ### #### # ##### ######### ### ######## ############# ## ####### ### ######## ## ##### #######. ### ###### ## ############## ####### #### ### ###### ########## ## ### ###### ## ######## ## ### ## #### ### ##########, ######### ####### #### ### #### ############. ####### ### ### ######## ### ######### ## # #### ##### ##### ## ###### ### ### ######### ## ####### ########## ### ### ###–### #######. ### #### ### #### ###### ## ####### ### ## ### ######### ########–### # ###### #### ####. ### ### ############## ## ### ### ######### # ###### ## ######## ## ### ### #########.

Lennart Augustsson has done most of the implementation of the USB support for the NetBSD project. Many thanks for this incredible amount of work. Many thanks also to Ardy and Dirk for their comments and proofreading of this paper.

- ####### ####### ## ##### ## ### ######## ######## ## ## ####### ###### ####, ####### # ######## ###### #########.
- ### ####### ### ## ######### ### ############ ## ### ####.
- ####### ### ####### ########## ### ####### ####### ## ### #### ######
- ## ### ####### ### ## ####### #### ### ###, ### #### ######## ### ## #### ##### ## ##### ####### ### #### ###.
- ######### ####### ## ####### ############ ## ### ######### ###### ##### ##–###### #### ### ####### ## 126 ####### #### ### ## ######### ## ### #### ###, ####### ###### ########## ## ######### ## ### ###### ### ## #### #### #####–#### ## ### 12#### ######### #########. (#### 400#### #### ### 2.0)
- ####### ### ########### ### ####### ###### ########## ########### ##### ##########

#########.

#####. ##### ### #### #### ###### ## ####### ### ######### ##### #######, ## ###### ####### ### ### ####### ####### ######### ## ##### ######### ###### ##### ### ############ ### ### ## ######## ####### ### #### ### ######. #### ####### ##### ## ####### ######### ########### ### # ##### ############# ## ### ####### ############## ## ### ### ##### ## #######/######. ## ## ########### ####### ## #### ## ######## #### ### ######## ############## ######### ## ### ########## #####.

13.1.1. ######### ## ### ###

######. ### ###### ##### ###–##### ### #### ########## ######, ######### # ####### ######### ## ### ####–#### ### ### ########## ##########. ## ######## ############## ## ### ########, ########## ## ######### ### ######## ## ######### ###/## ###### #########. #### #### ########## ###### ########### # ####### ### ######### ######## ########### ###### ## ### ######### ########### ### #### ##### ## ### #### ## ### #######.

#############, ##### ###–########### ## ### ######, ###### #########, ### ############# ######## (#####) ### #### ######## ##########. #### ######## ##### #### ######## ### ####### ##### ### ### ###### ######## ########### #### ####.

(####### ##) #######. ##### ####### ######### ### ######## #### ## #### #### ### ##### ##### #### ### ####### ####. #### #### ######### ########## ############# ## #### ### ###### ######### ## ##### ##### ## ### ###### ## #########. #### ### ### ### ###### ######### (#####) ####### ## ### ######## #####.

13.2. ####

(##) ######## ### ############ ## ####### ## ### ###. ###### ## 1 ########### ### ####. ## ### ##### ## #### ##### ### #### ########## #####–#### # ##### ## ##### (###) ######.

######. ###### #### ##### ####### ### ###########, ###### #### #### ## ###### (###) ## #### ###### ## #### (##). ######### ### ###### ######### ## ### #### (###### #########). ######### ##### ### #### ## ### #### ### ### ###. #### ######## ## # ###### ### # ###### ##### ## ##### ### ######### ## ### #### ### ###### ###### ### (########### #########), ### (#####), ##### (##### #########) ## ####### (####### #### #####, ###### ### ######### ## ############). ####### 8.5 ## ### ### ############## ######## ### ####### ## ####### ## #### ######. #### ######### ##### ## ######### ### ##### ## # ### ###: #######, ####,

###########. ### ##### ## ######### ### ##### ############### ### ######### ##### (####### ##########).

######.

(####### #########) ## ### ####### ######### ### #######. #### ####### ## ### ## ##### ######: #####, #### (########) ### ######. ### ###### ###### ## #### ## ### ######. ## ##### ## # #### #####, ### ######### ## ### #### ######(#) ## ##### ## ### ###### ######. ### ######### ## ### ###### ##### ## ### ######## ## ### ######### ###### ### #### #####, ## ## ## ##### ### ## #### #####. ### #### ########## ######## #### ######## ######### #### ### ####### ###### ## ### #### ##### ### ### ####### #### #### ######## ##### ### #### ##### ## ### ###### ###### ########. ###### ## ##### ######### ## ######## ####### # #####–###### ### ## ######### ## ### ### ############# [2]. ### ####### ### #### ###### #### ### ### ###### ##### ##### ## ####### 11 ## #### #############. ## #### ###–#### # ####### #### ####### ##### ###### ######## ### ## #### ## ##. ## ####### ### ######## ###### ##### ######## ### ## ###########. ## ###### #### ####### ## ######### #### #### ####### ####### ######### ## ### #####. ##### ### ###–###### ### ############## ### #### ########### #########: ######### #### ###–####### ######### (####; #####) ### #### #### ########## ######### (####; ######, #########, ######## #############). ### #### ############# ### #### ######## ## ###### ######## ########### ## ######### ### #### ########## ###### ## ###### # ######## ######## ## ### ######### ### #### #####. #### #### ########### ### #### #### ########### ## ######### # #### ######## ######### ##### # ### ## #### ##########.

13.2.1.

1024 ######## ## ### ##### #### ##########. ## ########### ### ######### #### #####: ######## ########### (##) ### ##### ##### (##). #### ## ########## # ###### ## ## ############ ## ## #### # ###### ########. ### ### # ##### ## ######## (### ###) ########.

#######. ### #### ###### ###### ##### ######### #### ######## #######. ### ##### ########, ##### #### ########### #########, # ## ## #########. ### ##### #### ## ######## ##### ### ### ######### ## # ## ### #### ####. ########### ######### #### ## ## ######## ##### ####### ## ### ##### ####### ########### ### ### ######## ######## ## ## ### ####### ## ### #########. ### #### ########### ## ###### ## ### ## ### ######### ######### ### #### #####. ### ### ### ######### ######### ##### ## ### ## ### ####### #####–####, ##### ## #### ###### ## ### ## ### #### #########. ### ######### ####### ##### # ######### ######## ## ####:

#####. ##### ######## ### ####### ### ### ####### ##### #### ### ######### ### ########## ##### ###–##### ### ### ### ### ### ########### ####### ## #### #####. ### #### ## ##### ### ###### ## ### ## ### ### ######### ######### ### #########. ### #### ###–####### #### #### ####### #### #### ## ## ### ### ### ### ########## ######### #########. ##### ######### #### #####, ### ## ### ### ######### ######### #### ##### ### ########## ## ### ## ### ### ####### #########. ## #### ####### ### ### ######### ######### #####, ######## ## ### ### ######### ###### ## ### #### ##. ## ########## ### ######## ## ######## ## ###### ######### ######### ##### ## ########## ### ######### ## ### ######## ## ### ### ## #### #####. ## ### #### ## ### # ######## ### ############ ########## ### ## ### ## ### ## ###### ## #### ## ######### #### ### ######## ### #########. ## ##### ######### ## ####### ## # ## ####### ### ####### ##### #####. ## ### ##### ###### ###### ### ## ### ## # ## ### #### #### ### ### ###### ###### ## ########### #### ######### ## ####### ## ###### ### ########## ###### ## ### ######### ########. ## ## ### #### ###–####### ######## #### ## #### ### ##### ######## ### ######### ## ######## ## #####. #### ###### ### ######### ####### ####### #### ###### ### ### ######## ######### ### #### ##### #########.

#############.

13.2.2.

########### ## #### #### ########## ## #### #######. ### ########## ####### #### # ### ## ######### ## #########, ### ## ##### ## ########## ########## ### ### ######## ## ### ##### ## ######### ## # #####. ### #### #### ######### #### ## ### #### ########## ## ### ######## ########## (##) ## ##### # ##### ## ######## ########### (###) ## ########. ### ## ######## ### ####### ###### #### ####### ### ## ######## ### ### ########### ######## #### ### ######### #### #######. ### ######## ## ### #### ####### ### ####### ##### #### ######## ### ##### ### ##### ### ### ####### ### #####, ### ## ## ####### ## ### ##########. ### #### ##### ## ######### #### ##### ### ######. ####### ### #### ######### ### ###### #### ## ##### ### #####. ######### ### ### ###### ## # ####, #### ### ##### ## ### #### ######## ### ######### ## ##### #### ###.

######### #################### ####### ####

#######. ### ########## #### ##### ### ### ############ ####### ### #### ######, ## ## # #### ##### ### ## ### ## ######. #### ### ######### ######### ### #### ##### ###### ### ###, ## ##### ### ##### #### #### ## ### ##### ###### ## ## ##### #### ##### 0 ## ### #### ## ########## ###. ## ### ### ## #### #### ### ########### ### ### ######### ### ##### ### ######### ############. ### ########### ### ####### ### ##### ###### ## ### ##### ##### ### ######## ###### ## ### ##. ##### ### ### ##–###### ######### #### #### ###, ### ####### ### #### ###### ### ######### #####.

############ ### ######### ####### ####### ## ###### ## ####### ### #### ##### ### #### ### ######### ### #### ######## ### ########## ######### ### #######–#### #########.

#############. ######## ##### ### ###### ##### ######## ###### ## ### ###### ## # ########## ### ### ######### ######### ## ### ## ### ######### ####### ### ### ######## #####. ### ##### ##### #### ## ### ######### #######:

- ### ###### ############# ###########
- ### ##### ## ########### #### # ######
- ####### ### ######### ### ######### #### # ######.

13.3. ### ######

13.3.1. ###### #############

###########. #### ##–#### ### ### ## #### ##############, ## ##### ### ## ######## ###### #####/##–####. # ############# ######## ##### ### ######### ############. ###### #### ############# ##### ### ## ######## ##########. # ###### ######### ## # ######–#### ## #########. ### ####### ### ######## ### #### ## ######### ### ### ##### #### (##### #####) ### ## ######### ### ### #####, ##### ### ####### (### #####). ### ########## ## # ############# ### ###### ## ### #### #### ### ### ## ######## ## ## ######### #######. #### ######### ### #### ##########, ######### ######–### ####### ## ####### ##########. ## ### ####### ####### #### ## #### ## ###–#### ######### ##### ##### ### ####### ## ###### ### ######.

0 ## #### ######### ### ## #########. ######### ### ### ############## ###### ###### ### ############# #### # ######. #### ####### ####### ## ########### ##### ######## ## ######## #### #### ### ######. #### ######## ### # ###### ####### ###### # #############, ### ########## ###### #### ### #########. ### ####### ########, ######## 0, ## ### #### ## ### ######### ### ######### ## ### ##############. ## ## ####### ## ### ######## ##### ### ### ##–###### ######### ## ###### #######.

0 ##### 1 ##### 2 #### 0

3 #### 2 #### 1

(#### 4 ### ## 32 ##### #####)

(### ####### 9.6 ## ### ### ############# [2]).

#######. ### ######## ##### ###### ##### ########### ## ##### ########### ######### ## ### ### ###. ###### ## ### ########### ## ######## ####### ######## #####.

- ###### ###########: ####### ########### ##### ### ######, #### ######, ####–### ### ######## ##, ######### ###### #####, ######## ### ######## ## ######–####, ####### ###### #### ### ### ####### ########, ###.
- ############# ###########: ### ###### ## ########## ## #### #############, ####### ### ###### ############# ######### ### ##### ############.
- ######### ###########: ######### #####, ######## ### ######## ## ##########, ###### ## ######### ######## ### ### ######### ### ### ###### ## #########.
- ######## ###########: ######## #######, ######### ### ####, ####### ###### #### ######### ### ####### ######### ## ##### ## ######### ########. ##### ## ## ########## ### ### ####### ######## (######## 0) ### ## ## ##### ####### ## ## ######### ##########.
- ###### ###########: ## ### ##### ########### ###### ####### ### ######## ### #### ######.##### ### ## #### ## ######## ########### #######, ######## ## ######## #########.

#######.

#####. ####### ###### ######### ## ######### ## # #### ### ####### # ######## ## ## ###### ## ########## ## ####### ## ### ######## (########### #########) ## #### ### ########## (########### ########). ######### ## ## ######## ### ########## ## ### ####. # ######## ### ###### ########, #### ## ######## (## # ######## ### #### ###). ##### ### ### ##### ## ######### ### #########. ######### ### ###–### ### ## ######## ## ### ###### (############). ##### ######### ### #### ## ######## ### ### ## ### ## ############# ## ### ######. # ###### #### ## ##### ### ## ########### ## ######## ### ## ######### #### # ######## #### ### ###–##### ###### # ######### ###### ## #### (#######). ##### ### ###### ## # ###### ############# ########## (###) ### ########### #######. ### ##### ### #### ## ### ###### ### ##### ##### ## ####### ####, ###### ###### ## ####### ## ######## ######.

##########, ### #### ########## (####) ## ### ## ###### (####) #### ##### ### ######## #### ####### ## ####### ###### ####, #### ### #### ###### ######## ####### #### ### ####### ###### ####.

######### ## ## ### # ####### ### # ###### ## ###### #### #### #### #########. ### ####### ################ ## # ##### ##### ####### 200 ##### ## ####, ###

5 ##### ######### ## #### ####. ### ###### ### ### ###
(###) ####. ## ###### ### #### ########## ## ###### # ###### #### ##
#########. #### #### ## #### #####
############, ## ### ###### ## #### ## ## #### ## # ###### ## ###### ##### ##–
########. ## ## ############# ##### ###### ## # ###### ###### # ######## ### ####
#######. ###### ### #### #### ## ######## ## #### ### ###### ##### ##
###
####. ### #######
#####.

#####: #
/ ####### ####: ##### ## ### ####### #### ### ######, ######### ## ### ##–
(######## 0). ### #### ####### ### ###### ######## ### ########–
####. ### ########## ####### ######### #### ### ####### #### ### #####
#############
[2]. ##### ######## ### #### ## ##### ### ######### ### ######. # ##### ### ##
######## #### #### ## ######### ## #### ###### ## ######## ## ####### 9 ## ###
[2]. ### ######## ######### ## #### #### ### ## ######## ##
#############.

- #### ####: #### ## ### ### ########## ## # ### ############ ######.

- ######### ####: ### #### ##### # ####### ### #### ## ### ###### ### ## ### ##–
####, ## #### ### ### #### ######. ######### #########
######### ## # ######### ######### #### ######## ### ####.

- ########### ####: ##### ##### ### ######## ### ########### ####, ### ####–
#######, #### ##### #######, ### ## ########## ########.
##########–
####. ####### ## #######, #### ### ######### ######### ### ####### ## ## #####
########–
(###) ### ## ### ####### #### ## ###### ##### ## ##### ### ######## ####.
########### ####### ### ####### ### ####### ## #### ## ###### ######## ## ###
###########.

########
####. ######### ### ######### ###### ###### ## 1 ###########. ### ####–
#############,
5.6 [2]. ########### ### ######### ######### ### ####### ## ####### ## ## 90% ##
#####. ####### ### ####### ### #### ######### ### #####–
##–
####### #########.

##
5## ### ### ############# [2], ####### 1.3 ## ### #### #######–
[3] ### ####### 3.4.2 ## ### #### ############# [4].

13.4. ###### ##### ###

#########, ### ###–#### ##### ######## ## ### ####, ######### ### ###### #### 100 ## ## #######. ## #### ##### ### ###### ## ## ### ####### ##### ### ######### ## ###### ####### 0. ### ######## ##### #### ####### ## ######## ### ####### ########### ####### ### ####### ####. ##### #### ## #### #### # ### ####### ####### ## #### ### ##–#### #### #### ### ####### ###### ####### (####### 0). ######## ###### ####### ##### ## #### ## ####### ### #######. ### ####### # ##### ###### ##### ## #### ## ####### ## #### ## ####### ### ## ############# #########. # ###### ### #### ######## ##### ## ### #### ####### ##### ####### ## #### ## ####### #### ###### ####### ### #### ######. ## ####### #### ###########, ### ###### ###### #####–##### ########## ##### ##### ## #######. ####### ### # ######## ######## ## # ####### ##### ### ####### ### ### ####### ###### ### ###### ########. ## ##### #### ## #### ######## ####### ##### ###### ## ### ###### ## ##### ### ######## ########## ###### ### #############. #### ###### #### ##### ## ####### # ###### ## ### ##########.

#######. ## ### #####, ### ##### #### ######## #### ### ######### ###–########### ##### # ###### ######## ## # #############. ## ####### ####### #### ######## ####### ## ######### ##########, ### ##### ######## #### ### #####–##### ## # ############# #### #### ### ### #### ####### ## # ######. #########–##### #### ###### ### ##### ###### ### ### ### ### #######. ###### ###### ### ###### ###### ########## ### ###### ## ### ###### #####, ### ### ##### ##, ## #### #### #### ### ###### ########## ###### #### ### ### ### ####### ### ####### ####### ### ##. ## ##### ######### ########### ######### ###### ### ##### ### ######### #### ## ###### ####, ### ###### ######## ########## ### #### ##### ### #### ## ###### ### ### #### ## ######## ####. #### ### #### ## ###### ### #### ###### ## ###### #####, #### ###### ### ###### ##### ##### ## ########.

13.4.1. ###### ########## ###

######. ### ###### ## ### ######## ### ########## ### ############# ## ####### ## ### ##### ## ####. ####### ###### #### #### #### #### ## ### ##### ##### #### ### ###### ##########.

########### # ###### #### ### #### ############ ### ########### #### ### ## ########## ## ### #### ###### ########. #### ##### ###### ## ### ###### #### #### ####### ####### ###### ####### (### ### ###### ##########) ## ##### ##### ## ######## ## ######## ### # ###### #### #### #########.

##–####### ## ### ### ######. ### ###### ###### ########### ######### ##### #### ### #### # ########## ######. ### ###### ###### ###### ### ### ###### ####### ### ### ###### ######### ## #### #### ### ###### ### ### ########## ####### ##. ## ### #### ###### ######### #### ## ### #### #### # ###### ### #### ######### ## #### ####, ### ######### ### ####### ### ######### ### ###### #### ## #######. # ###### ##### #### ####### # ################## ######## ## ### ### ### #### ## ####### ## ######### #####.

13.5. ### ####### ########

#############. ########### ## #### ### ## ##### #### ####### #######. ### #### ######## ###### ## ### ########### ####### ## ### ### #### ##### [1]. #### ##### ##### # ####### ###### ## ########### ############# ### #########. ##### ############## ####### #### # ######### ###### ###### #### #### #### # ###### ###########, ##### ############# ## ##### ## ####### ### ### ######## #### ## ## ## #### #### ### ############# ########. ### ### ############# [2] ######## ### ########### ## ### ### #####. # ##### ############# ### ##### ######### ####### (###) ### #### ####### ## ##### ### #########, #######, ######## #######, #######, #####, ########, ###. # ##### ####### ## ### ##### ############# ### #### ####### #######. ### # #### #### ## ###### ####### ### ### ########## ####### ## ### ### #### ##### [1].

#######. ########### ## ### ######## ##### #### ##### ## ######### #### ### ####### ###–### ### ######. #### ######### #### ####### ### ## #### # ### ########### #####–#### (###) ###### ###### ### ### ##############. #### ## #### ##### ######### ###### ### ###### #### ######.

####### #### ###### ## ########### ## ### ##### ###### #######, ## # ###### ## ######### #### ####### ## ####### ####### ### ##### ### ##### #######. ## ## ###### # #### #### ## ####### ### ####### ## ##### ####### ### ##### ###### ## ###########.

#######: ##### ######### ####### ### ############# ### ### ##### ######### ##–##### #### #########, ####, #######, #######, #####,###. ## ######## ## ## ##### ###### ##### ############## ### ## #### ## #### #######.

###–####### ### ######## ## ##### #### ### # ###########. #### #### ####### # ### ###–##### ## # ######## ######### ### ### ####### ### ##### ## ### ##### ## ### ##–####. ### #### ## #### ### ### ####### ####### #####. ## ## ############## ## ##### ####### ### ##### ########## ####### ### ######### ###### ###

######### ######## #### ### ### ##### ###### ## ### ####### ######. ####### ##–
######
####.
######,
############# ######## ### ##### #### ######## #### ### ##/2 #### ## ### #### ##
##########
#############. ### ##### ### ######## ####### #### ###### ## ### ##–
######### ######### ### ######## ### ##### ## ### ### ########### #########.

#######: ######## ######## #### ####### #### #### #### ######### ### ##### ##
##.
###,
####### ####### ### ########### ## ### ######## ##### #### #### #### #########.

###############. ### ###### ##########
####### # ###### ### ####### ##. ### ##### ###### ###### ### ######## ## ## ###
######### ### ######## #### ##. ##### #### ### ###### #### ###### ###### ### ###
########. ##### # ##### ##### ### ###### ######### ### ######## ##
###. ### ###### #### #### ####### ### ######/#######/######## ## ## ####### ###
######
##.

#/# #####, ##### ## ###
####. ### #### #### # ####### ######## ########## ## #########. ###
########## #### ### ########## ##### ####### ### ######## ##. ## #### #### ###–
###
#########.

#######: #### ####### ####### ####### ### #### ####### ####### ## ###### #####
#########. ### ###### ### ######## ## ##### ## ### ####
#####. ### #### ######## ### ###### ######## ### ####### ## ######
########### #### ### #### ##### ## ### #### ### ######, ######### # #### ###–
####### #### ### ### ####. ##### ### ### ######## ### ######### ## # #######
#######.

2 ######### ##### ## ######## ##
####### #####.### ####### ####### ### ##### ## ####### ### ####### ### ######
####### ### ####### #### ### ##### #### ######### ### ### #### ## ## ##### ##–
######. ##### ## ########## # ###### ######## ###
####
########. ### ############# ######### ####### #### ###
###########.
#####
######## #### ### ######## ####### ### #### ####### #######. ### ######## #####
###. ##### ## #### ###### ##
###########.

########.

14.

Written by Jeroen Ruigrok van der Werven (asmodai) and Hiten Pandya.

Special thanks to Matthew N. Dodd, Warner Losh, Bill Paul, Doug Rabson, Mike Smith, Peter Wemm and Scott Long.

######.

14.1. ######

14.1.1. ####### ## # ######

(#.#., ####, ####### #######) ### ### ###### ############## ## ### ##########. ### *device driver interface (DDI)* ## ### ####### ######### ####### ### ###### ### ### ###### ###### #########.

14.1.2. ##### ## ######

####®, ### #### #######, ## ##### ##### #### #### ##### ## ####### #######:

- ##### ###### #######
- ######### ###### #######
- ####### ###### #######
- ############# #######

Block devices ######### ## # ### #### #### ##### #### ###### [## ####]. #### #### ## ###### ######## ## ### ######### *buffer cache*, ##### ### ###### ######## ###### ## #### ## # ######### #### ## ######. ##### #### ###### ##### ### ##### ## ###### ######, ##### ##### #### #### #### ### ######## ## ###### ## ### ###### ## #### ######## ### ###### ### ### ########## #### ########, #### ########## ######.

14.1.3. #########

#######, ## ### ######## ## ####### 4.0 ### ###### ### ########### ####### ##### ### ######### ####### ###### ############.

14.2. ######## ##

Newbus ## ### ############## ## # ### ### ############ ##### ## ########### ###### ##### ### ### ############ ## ####### 3.0 #### ### ##### #### ### ######## #### ### ###### ####. ## ### ### ##### 4.0 ###### ## ###### ### ####### ###### ## ### ### ###### #######. ### ##### ### ## ####### # #### ############### ##### ## ############### ### ####### ###### ### ####### ##### # #### ###### ######## ## ### *Operating System*.

######:

- ####### #########
- #### ############## ## #######
- #############

###–### ## # ###### #### ######.

“root” ###### ##### ## ### ###### ## #### ### ##### ####### ##. ### #### ############, ##### ## ######### # ###### ##### ## ###### ##### ### #### ###### ## *host-to-PCI bridges*, ###. ######## ## ##. ### #86, #### ###### ###### ## ### *“nexus”* ######. ### #####, ####### ######### ###### ## ##–### #### ######### ######### ####### ############# ## ### ######### ######## ########, ######### *lca*, *apecs*, *cia* ### *tsunami*.

######. ### ######## #### ### ###### ## ########### ## # ###### ######. ### ###### ## ### ###### ### #### ########; # ###### ##### ### ######## ## ##### ###### # *“bus”*. ######## ## ###### ###### ## ### ###### ### ### ### ###, ##### ###### ##### ## ####### ######## ## ### ### ### ###### ############.

#####, # ########## ####### ######### ##### ## ### ## ########### ## # *“bridge”* ######, ##### ######## ### ### ##### ### ### ######## ###. ## ####### ## #### ## # *PCI-to-PCI bridge* ##### ## ########### ## # ###### *pcibN* ## ### ###### ### ### ### ### # ##### *pciN* ### ### ######## ###. #### ###### ########## ### ########–###### ## ### ### ### ####, ######## ###### #### ## ## #### ### #### ######### ### ####### ######.

#########. ### ###### #### #### ### ### ###### ##### ### ##### ## #######. ##, ######### ### ##### ## ### #### #### ## ### ###### ###### ##### ##### ##### ### #########.

###

0x230, ## ## #### ### ######, ## #### #### ### ### ###. ### ### ### ##### ##

> ## ### ########## ###### ##### ## ### #### #### ### ### ###, ##### ####### ### ########### ###### ### ####### ### #####. ### ###### ## #### ########## ####### ## #### ##### ## #### ## ######### ### ########. ### #######, ### 0x230 ## #### ##–##### ##### ###### ############# ## 0xb0000230 ## # #### ### ## ### ### ######.

######## ########## ### ## ########## ## ### ##### ## ### ###### ####. ### ##–###### ## #### ##### #########, ### ########## ### ####### ########## #### ### ########## ### ######## ########### ### ### ########## ### ####### ## ### ####### ### ### ######. ## ###32, ### ### ### ########## ### #### ####### ## ### ######### ##### ######. ### #### #####, ###### ### #### ####### ##### ## ####### ## # ###### ###### # ##### ### ###32 ### ### ######## ####### ###### ## ##### (#.#., ### ## #######).

#########, ###### ##–######## ### bus_space #### #### ######. ##### ####### # ###### ### ## ####### ###/#### ### ###### ###### #####/######. ### ######### ## #### ## #### # ###### ###### ### ###### ### ###### ############# ######### ## ########### ######### (#### ######## ######## ####).

#########. #### # ######## ## #########, # ###### ### ######## ### ########## bus_space_tag_t ### bus_space_handle_t #### ### ########.

#######. ##### ### ### .m ##### #### ### ##### ##### ### src/sys #########.

#####. #### ###### ## ### ###### ### # ##### ## ####### ##### ## #########. ### ###### ### ##### ####### #### ##### ####### ## ####### ### ###### ### ####### ########. ### ######### ####### ######### ## # ###### ### ####### ## # ###### ## ############. ## ########### ## ###### # ##### ## ####### ####### ##### ### ## ########### ## # ######.

######, ### ####### ### # ###### ### ######## ## ### ####### ###### ####### ## ### ######. #### # ###### ## ######## ## # ###### ###### *auto-configuration*, ## #### ### ###### ##### ######## ## ### ######. # ###### ### ##### *detach* #### ### ###### ### *re-attach* ## # ### ###### #### # ### ###### #####. #### ###### ####### ########### ## ####### ##### ### ## ###### ### ###### ###########.

#######. ### ######### ##### ## ###### ## # ####### #### (##### ##### ######## ## ##### foo_if.m).

####### 14.1. ###### #######

```
   # Foo subsystem/driver (a comment...)

  INTERFACE foo

 METHOD int doit {
  device_t dev;
 };

 # DEFAULT is the method that will be used, if a method was not
 # provided via: DEVMETHOD()

 METHOD void doit_to_child {
  device_t dev;
  driver_t child;
 } DEFAULT doit_generic_to_child;
```

########, ## ######### # ###### #### #foo_if.h# ##### ###-##### ######## ############:

```
int FOO_DOIT(device_t dev);
int FOO_DOIT_TO_CHILD(device_t dev, device_t child);
```

####, #foo_if.c# ## #### ####### ## ######### ### ############# ######### ###### ####; ## ######## ################ ## ##### ######### ##### #### ## ### ######## ## ### ######## ######### ## ### ######## ###### ##### ### #### #### ########.

##########. ### ##### ########### ######### ## ###### *"device"* ### ######## ####### ##### ### ######## ## ### #######. ####### ## ### *"device"* ######### ####### *"probe"*, *"attach"* ### *"detach"* ## ####### ##-####### ## ######## ### *"shutdown"*, *"suspend"* ### *"resume"* ### ######## ##### ############.

######, #### ####### ######### ## *"bus"*. #### ######### ######## ####### ######## ### ####### ##### #### ########, ######### ####### ## ###### ### ###-##### ########## ###########[1], ##### ############ (*child_detached*, *driver_added*) ### ######## ########## (*alloc_resource*, *activate_resource*, *deactivate_resource*, *release_resource*).

######. ##### ####### ##### ######## ### ### ##### ### #########

[1] ###_#######_####_####(9) ### ###_#######_#####_####(9)

####### ### ### ########## ### ######## ### ### ##### ###### ##### ## ########### ### ########. ## ######### ###### ####, #### ## ##### ####### #### ######### ####–##### ##### ###### ### ###### ### #### # ###### ## ### ######. ### ######### ### ###### BUS_TEARDOWN_INTR(device_t dev, device_t child, ...) ### ## ###### ##### ### ####–#### bus_teardown_intr(device_t child, ...).

#############. ### #########, ### ### ### ###### ####### ### ##### ######### ##### ### ### ####### *read_config* ### *write_config* ### ########## ### ######–####### ######### ## # ### ######.

14.3. ######

####, #### ####### ##### #### ###### ## ########### ##. #### ########### ## ##### ## ### #### ######### ## ##### #########.

14.3.1. ######### ######### ## ### ######

src/sys/[arch]/[arch] # ###### #### ### # ######### ####### ############ ####### ## #### #########. ### #######, ### i386 ############, ## ### SPARC64 ############.

src/sys/dev/[bus] # ###### ####### ### # ######### [bus] ####### ## #### #########.

src/sys/dev/pci # ### ### ####### #### ####### ## ##### #########.

src/sys/[isa|pci] # ###/### ###### ####### ###### ## #### #########. ### ###/### ### ####### #### #### ## ##### ## #### ######### ## ####### ####### 4.0.

14.3.2. ######### ########## ### ####

devclass_t # #### ## # #### ########## ## # ####### ## # struct devclass.

device_method_t # #### ## ### #### ## kobj_method_t (### src/sys/kobj.h).

device_t # #### ## # #### ########## ## # ####### ## # struct device. device_t ########## # ###### ## ### ######. ## ## # ###### ######. ### src/sys/sys/bus_private.h ### ########–###### #######.

driver_t # #### ## # #### ########## ##### ########## struct driver. ### driver ###### ## # ##### ## ### device ###### ######; ## #### ##### #### ####### ## ### ######.

```
 struct driver {
KOBJ_CLASS_FIELDS;
void *priv;   /* driver private data */
 -};
```

14.1. *driver_t*

device_state_t ####, ##### ## ## ###########, device_state. ## ######## ### ######## ###### ## # ###### ###### ###### ### ##### ### ################# #######.

```
 -/*
  * src/sys/sys/bus.h
  */
 typedef enum device_state {
DS_NOTPRESENT, /* not probed or probe failed */
DS_ALIVE,  /* probe succeeded */
DS_ATTACHED, /* attach method called */
DS_BUSY   /* device is open */
 -} device_state_t;
```

14.2. ###### ###### *device_state_t*

15. #####

Contributed by Jean-Francois Dockes.

15.1.

####. #### ##### ## ###### ## ### ####### ### ### ########.

###(4) ######### ## ### ####### ##### ## ### ##### #########. ## ###### ##-######## ### ######### ########:

- # ###### #### ######### (####, #####, ######) ## ######### ##### ### ##### #########. ### ##### ####### ### ## ########## #### ### ###### *OSS* ## *Voxware* #########, ######## ###### ########## ########### ## ## ###### ####### ###-#########.
- ###### #### ### ########### ##### #### (###### ###########, ####### ########).
- # ####### ######## ######### ## ################# ##### ######### #######.
- ########## ####### ### #### ###### ######## ########## (##97), ## ###### ################# #### (##: ### ### ########).

###-####, ##### ####### ####### ### ##### ########## ## #### #### ### ####### pcm ####.

#######, ### #### pcm #### ##### ## ### #######, ###### #### ## ### ##### ######, ## ####### ## ### ################# #######.

###-### ### ### ### #### ## ### ######## #########. ###, ##### ### ##### #### ## #### ### #####, ## ## #### ###### ###### ## ########. #### ######## ##### ## #### ## ######## ## ### ######### ######### ### ####### #### ######### #### ### ##### ##### ######## ### ######## ####.

###########, ## ## ######### ## ######### #### # ####### #######, ### ### #### # ######### ###### ######## ## ####://######.#######.###/###/########.#

15.2.

/usr/src/sys/dev/sound/, ###### ### ### ###### ##### ##-####### ###########, ##### ## /usr/src/sys/sys/soundcard.h

/usr/src/sys/dev/sound/, ### pcm/ ######### ##### ### ####### ####, ##### ### pci/, isa/ ### usb/ ########### #### ### ####### ### ### ### ### ######, ### ### ### ##### #######.

15.3. #######, #########, ###.

######. ### ##### #### ## #### ## ### ### ## ### ######## ######## ## ### ######## ### #### ###########.

#######, ##### ####### ###### ## #### ####:

- #### ####### ########## ## pcm ##### #######, #### # struct snddev_info ###### ###–#### #########:

```
static driver_t xxx_driver = {
   -"pcm",
   xxx_methods,
   sizeof(struct snddev_info)
-};

DRIVER_MODULE(snd_xxxpci, pci, xxx_driver, pcm_devclass, 0, 0);
MODULE_DEPEND(snd_xxxpci, snd_pcm, PCM_MINVER, -
PCM_PREFVER,PCM_MAXVER);
```

  #### ##### ####### #### ## ##### ########## ####### ########### ##### ##### ######. # ####### #### ######### ## ####### ######### ## ### ###### #######. ### ####### ## ###### ## pcm ## ### ##### ## pcm_register() ### mixer_init(). pcm ##### ###### #### #### ####### ## # ######### ## ##### ## ### ##### ###### #####–#####.

- ### ##### ###### ###### ####### ###### ####### ### ##### ## ##97 ######### ## pcm ## ####### mixer_init(). ### # ##### #########, #### ###### ## #### # #### ## xxxmixer_init().

- ### ##### ###### ###### ####### ######## ### ####### ####### ############# ## pcm ## ####### pcm_register(dev, sc, nplay, nrec), ##### sc ## ### ####### ### ### ###### #### #########, #### ## ####### ##### #### pcm, ### nplay ### nrec ### ### ###### ## #### ### ###### ########.

- ### ##### ###### ###### ####### ######## #### ## ### ####### ####### ## ##### ## pcm_addchan(). #### #### ## ### ####### #### ## pcm ### ###### ## #### # #### ## xxxchannel_init().

- ### ##### ###### ###### ####### ###### #### pcm_unregister() ###### ######### ### #########.

#######:

- ### # device_identify() ###### (#######: sound/isa/es1888.c). ### device_identify() ###### ###### ### ### ######## ## ##### ######### ###, ## ## ##### # ######### ######, ###–#### # ### ### ###### ##### ## #### ###### ## #####/######.

- ### # ###### ###### ############# #### ########### ##### ### ### ####### (##–#####: sound/isa/mss.c).

pcm ####### ###### ######### device_suspend, device_resume ### device_shutdown ########, ## #### ##### ########## ### ###### ######### ######## #########.

15.4.

pcm #### ### ### ##### ####### ## ####### ## ##### ## ###### #######.

#######: *CHANNEL* ### ###### *MIXER* ## *AC97*.

AC97 ######### ## # #### ##### ######## ###### (######## ####/#####) #####–####, ########### ## ####### ### ######### #### ## ##97 #####. ## #### ####, ### ###### ##### ######### ## ######## ## ### ###### ##97 #### ## pcm.

15.4.1. ### #######

15.4.1.1. ###### ##### ### ########

######, ### ### ######### ### #### #### ### ###### #### ####### #### ## ########.

#########, ### ##### ######### ## ## ###### #######.

#########, ###### ### channel_init() ##### ### # ####### ## ### ####### ###### ######### (### ####### ### ####### ####### ### ####### ### ## pcm).

15.4.1.2. ######## ## #### ########

#########, ### pcm #### ### ### ##### ####### ########### ####### # ###### ###### ####, ######### ## # struct snd_dbuf.

struct snd_dbuf ## ####### ## pcm, ### ##### ####### ###### ###### ## ######## ## ##### ## ######## ######### (sndbuf_getxxx()).

sndbuf_getsize() ### ## ####### #### ##### #### ###### ## sndbuf_getblksz() #####.

#######, ### ####### ######## ######### ## ## ####### (####### ### #### ### #########):

- pcm ######### ##### ## ### ######, #### ##### ### ##### ######## xxxchannel_trigger() ######## #### # ######### ## #######_#####.
- ### ##### ###### #### ######## ## ########## ######## ### ##### ###### #### (sndbuf_getbuf(), sndbuf_getsize()) ## ### ######, ## ###### ## sndbuf_getblksz() #####. ## ##### #### ### chn_intr() pcm ######## ### #### ########### ##### (#### #### ###–###### ###### ## ######### ####).
- chn_intr() ######## ## #### ### #### ## ### #### #### ### ########### ## ### ###### (### ####), ### #### ########### ####### ## ### snd_dbuf #########.

15.4.1.3. #######_####

xxxchannel_init() ## ###### ## ########## #### ## ### #### ## ###### ########. ### ##### ### ######### #### ### ##### ###### ###### #######. (### ### ##### ### ##–#### #######).

```
static void *
xxxchannel_init(kobj_t obj, void *data,
  struct snd_dbuf *b, struct pcm_channel *c, int dir)❶
{
   struct xxx_info *sc = data;
   struct xxx_chinfo *ch;
    -...
   return ch;❷
 -}
```

❶ b ## ### ####### ### ### ####### struct snd_dbuf. ## ###### ## ########### ## ### ######## ## ####### sndbuf_alloc(). ### ###### #### ## ### ## ######## # ##### ######## ## ### ######### #### ######## #### ### #### ######.

c ## ### pcm ####### ####### ######### #######. #### ## ## ###### ######. ### ######## ###### ##### ## ## ### ##### ####### #########, ## ## #### ## ##### ##### ## pcm (##: chn_intr(c)).

dir ######### ### ####### ######### (PCMDIR_PLAY ## PCMDIR_REC).

❷ ### ######## ###### ###### # ####### ## ### ####### #### #### ## ####### #### #######. #### #### ## ###### ## # ######### ## ##### ####### ######### #####.

15.4.1.4. #######_#########

xxxchannel_setformat() ###### ### ## ### ######## ### ### ######### ####### ### ### ######### ##### ######.

```
static int
```

```
xxxchannel_setformat(kobj_t obj, void *data, u_int32_t format)❶
{
   struct xxx_chinfo *ch = data;
    -...
   return 0;
 -}
```

❶ format ## ######### ## ## AFMT_XXX value (soundcard.h).

15.4.1.5. #######_########

xxxchannel_setspeed() #### ## ### ####### ######## ### ### ######### ######## #####, ### ####### ### ######## ######## #####.

```
static int
xxxchannel_setspeed(kobj_t obj, void *data, u_int32_t speed)
{
   struct xxx_chinfo *ch = data;
    -...
   return speed;
 -}
```

15.4.1.6. #######_############

xxxchannel_setblocksize() #### ### ##### ####, ##### ## ### #### ## #### ############ ####### pcm ### ### ##### ######, ### ####### ### ##### ###### ### ### ######. #########, #### ##### ## ### ###### ## ##### ########### ###### ## ######### ######. ###### # ########, ### ##### ###### ###### #### pcm## chn_intr() ##### #### #### #### ### #### ###########.

####, ## ## #### #### ## ###### ######## #### ## #######.

```
static int
xxxchannel_setblocksize(kobj_t obj, void *data, u_int32_t blocksize)
{
   struct xxx_chinfo *ch = data;
     -...
   return blocksize;❶
 -}
```

❶ ### ######## ####### ### ######## ######## ##### ####. ## #### ### ##### #### ## ###### #######, sndbuf_resize() ###### ## ###### ## ###### ### ######.

15.4.1.7. #######_#######

xxxchannel_trigger() ## ###### ## pcm ## ####### #### ######## ########## ## ### ###–###.

```
static int
```

```
xxxchannel_trigger(kobj_t obj, void *data, int go)❶
{
    struct xxx_chinfo *ch = data;
    -...
    return 0;
-}
```

❶ go ####### ### ###### ### ### ####### ####. ### ######## ###### ###:

- PCMTRIG_START: ### ###### ###### ##### # #### ######## #### ## ## ### ####### ######. ## ######, ### ###### #### ### #### ### ## ######### ####### sndbuf_getbuf() ### sndbuf_getsize().
- PCMTRIG_EMLDMAWR / PCMTRIG_EMLDMARD: #### ##### ### ###### #### ### ##### ## ###### ###### ### #### #### #######. #### ####### #### ###### ##### #####.
- PCMTRIG_STOP / PCMTRIG_ABORT: ### ###### ###### #### ### ####### #####–###.

> ####
>
> ## ### ###### #### ### ###, sndbuf_isadma() ###### ## ###### ##–#### ########## ####### ## ### ######, ### #### #### #### ## ### ### #### #### ## ######.

15.4.1.8. #######_######

xxxchannel_getptr() ####### ### ####### ###### ## ### ######## ######. #### #### ###–###### ## ###### ## chn_intr(), ### #### ## ### pcm ##### ##### ## ### ######## ### ####.

15.4.1.9. #######_####

xxxchannel_free() ## ###### ## #### ## ####### #########, ### ####### #### ### ###–### ## ########, ### ###### ## ########### ## ### ####### #### ########## ### ########### ######### ## ## sndbuf_alloc() ### ### #### ### ###### ##########.

15.4.1.10. #######_#######

```
struct pcmchan_caps *
xxxchannel_getcaps(kobj_t obj, void *data)
{
    return &xxx_caps;❶
-}
```

❶ ### ####### ####### # ####### ## # (####### #################) pcmchan_caps ######### (####### ## sound/pcm/channel.h. ### ######### ##### ### ####### ### ####### ######## ###########, ### ### ######## ##### #######. #### ## ### ##### ###### ### ## #######.

15.4.1.11. ####

channel_reset(), channel_resetdone(), ### channel_notify() ### ### ####### ######## ### ###### ### ## ########### ## # ###### ####### ########## ## ## ### ####### ########## ####### ####.

channel_setdir() ## ##########.

15.4.2. ### #####

15.4.2.1. #####_####

xxxmixer_init() ########### ### ######## ### ##### pcm #### ##### ####### ### #####–#### ### ####### ### #########

```
static int
xxxmixer_init(struct snd_mixer *m)
{
    struct xxx_info   *sc = mix_getdevinfo(m);
    u_int32_t v;

    [Initialize hardware]

    [Set appropriate bits in v for play mixers]❶
    mix_setdevs(m, v);
    [Set appropriate bits in v for record mixers]
    mix_setrecdevs(m, v)

    return 0;
-}
```

❶ ### #### ## ## ####### ##### ### #### mix_setdevs() ### mix_setrecdevs() ## #### pcm #### ####### #####.

soundcard.h (SOUND_MASK_XXX ###### ### SOUND_MIXER_XXX ### ######).

15.4.2.2. #####_###

xxxmixer_set() #### ### ###### ##### ### ### ##### ######.

```
static int
xxxmixer_set(struct snd_mixer *m, unsigned dev,
             unsigned left, unsigned right)❶
{
```

```
    struct sc_info *sc = mix_getdevinfo(m);
    [set volume level]
    return left -| (right << 8);❷
-}
```

❶ ### ###### ## ######### ## # SOUND_MIXER_XXX #####

[0#100]. # ##### ## #### ###### #### ### ######.

❷ ## ### ######## ###### ######## #### ### ##### ### ##### #####, ### #### ######## #### #####, ### ####### ####### ### ###### ##### ###### (## ##### 0#100) ## #####.

15.4.2.3. #####_#########

xxxmixer_setrecsrc() #### ### ######### ###### ######.

```
static int
xxxmixer_setrecsrc(struct snd_mixer *m, u_int32_t src)❶
{
    struct xxx_info *sc = mix_getdevinfo(m);

    [look for non zero bit(s) in src, set up hardware]

    [update src to reflect actual action]
    return src;❷
 -}
```

❶ ### ####### ######### ####### ### ######### ## # ### #####

❷ ### ###### ####### ### ### ######### ### ########. #### ####### ### #### ### ### ###### ### #########. ### ######## ###### ###### #1 ## ## ##### ######.

15.4.2.4. #####_######, #####_######

xxxmixer_uninit() ###### ###### #### ### ##### ## ##### ### ## ######## ##### ####–#### ###### ## ####### ####

xxxmixer_reinit() ###### ###### #### ### ##### ######## ## ####### ## ### ### ######## ### ########## ## mixer_set() ## mixer_setrecsrc() ### ########.

15.4.3. ### ##97

AC97 ######### ## ########### ## ####### #### ## ##97 #####. ## #### ### ##### #######:

- xxxac97_init() ####### ### ###### ## ##97 ###### #####.
- ac97_read() ### ac97_write() #### ## ##### # ######### ########.

AC97 ######### ## #### ## ### ##97 #### ## pcm ## ####### ###### ##### ####–######. #### ## sound/pci/maestro3.c ## #### ###### ##### sound/pci/ ### ## #######.

16. ##

######. #######, ## ####### ## #### ######### ### ## ### # ### ###### ## ## ######## ###### ######.

16.1. ###### #

#######. ##### ## # ##### ## ######### ####### ## ### ###### #### ####### ### ## ###### ## # ######.

16.1.1.

####, #### ##### ## ### *Card Information Structure* (###) ###### ## ### ####. ### ##### ###### ## ## ### ####### ##########–## ### ####### #######. ### ###### ###### ## ## ### ### ##### ######## ####### #### ### #### ######### ## ### ###. ### ## #### ### #### # ########### ######## ### #### ###### ## ########## # ###### ####### ## #### ### ###### ###### ##### ####### ## ### ######.

######## ######### ############# (####) ##### ####### # ######### ###### ### # ## #### #######, #### #### #### ###### ## ##### ######### ## #######. ##### #####–#### ###### ### ######, ###### ### ####### ## ##### ###### ######## ## ########–## ####, ### ### ##### ### #### ##### #### ### ####. ### ########### ## ### ####–#### #### ### ######### #### #####, ## ### ####### ### #### ## ###. ## ########–## ##### #####, ##### ####### ##### ##### ####### #### ## ### ##### ######## ####### ## ### ### #####, ### ##### ### ############ ### ####### ### #########.

####### ## #### ########, ####### ####### ####### #### ## ####### ### ### ##–#### ##############. ##### ####### ### ### # ########### ######## ########### ###### ### ### ####### ### ##### ## ### ######. ### #### ######### ##### ## ### ### ###### #### ### ####, ########## #### ## ####### #### ### ###### ### #### ### #### ##### ####### #### # ######### ############ ## ###### ## ### ####### ########. #######, ######, ### ####### ### # ###### ## ## ############# ## ### ######## #### ##### #### ### #### ######. ##### #### ####### ### ## #### ## ##### ##### ##### #### ## ####### ### ######. #####, ##### ####### #### ### #### ### #### ############ ### ####### ###.

###########, ### ### ##### ###### ## ########## #### ####, ## /sys/dev/pccard/pccarddevs. ## #### ######## # ### ## ###### #### ##### ### ## ###### ######### ###### ####### ## ### ##### ### ###### #### ## ##### #######.

#######, #### ###### ### ### ####### ## ### ####### ############ ##########– #### ## ###. ##### ####### #### ## ######## ## ######## ### ##### ######## ### #######. ##### ## ##### ## #### ## ## ### ### #### #### ###### ## # ########, ## ## ######### ### #### #### ### ### ###### ####### ### ######## #####. #### ###### ###### ######### ## #######, ### # ###### ## ####### ### ###### ## #### ####### ####### #### #### ##### ##### ## ### ########### ## ### ### ###### ## ### ## #### ########. #### ###### ### #######, ###### ##### ### ####### ######.

16.1.2. ###### ## **pccarddevs**

pccarddevs #####. ### ##### ####### ##### ### ####– ######## ####### ### ####### #### ### ####. #### ####### ## ###### ## ######### #####. ### #### ####### ### ### ## ### ######## #### ### #### ## ##### #######, ##### #### ##### ####### ## ####### ### # ########### ######. ### ########### ###### ######### ## ### #### (####### ## ### ### ######## ########### ##### ## ### ##### ######## ###, #### ## ## ##### ## ### ####### #######). ##### ### ###– ##### ### #### ######## ### ####### #### ### ### ###### ######## ######. #####– ##, ####### ######## ######## ## /* ### */ ########## ### ####### ######## ## ### ####.

###. ###### #### #### #### ###### ## ####### #####. ####, ###### ########## ####### ## #### #### ####### ## ##### ## #### ###### ## #### ########## # ###### ######## ##### ### #### #####– ######. ### #######, #### ### ### ##### ### ###### ###:

```
vendor FUJITSU   0x0004  Fujitsu Corporation
vendor NETGEAR_2  0x000b  Netgear
vendor PANASONIC  0x0032 Matsushita Electric Industrial Co.
vendor SANDISK   0x0045 Sandisk Corporation
```

####### ### #### #### #### ### NETGEAR_2 ##### ## ###### ## ### #### ####### ######### ##### #### ### ### ###### ## ####### ### ##### ##### ### ####### ## ### #### #### ####### ### ##### ####### ###### ##. ##### ####### ### ###### ###############. ### ###### ####### ####### ### #### ## #### #### #### ##, ###– ##### ## ### #### ## ### ######. #### #### #### ## ######## ##### ## pccarddevs, ## #### ## #### ## ### ######## ##### ######, ## #### ## ##### ### # ##### # ######– ####. # ####### ## ## ### ########## ### ############. ## ### ### ### ## ### #### 0xffffffff ## 0xffff ####### ##### ### ######## ### (### ###### ## ### ## #### ##### ### ###### ## ######### #### ## ######### #### ####### ##### ## ### ## ######## ######). ####### ##### ## # ###### ########### ## ### ####### #### ##### ### ####. #### ###### ## ### #### ## ####### ### ######## ### ########## ########.

########. ## ##### ## #### #######, ### ###### ## ####### ## ### ###### #####:

```
/* Allied Telesis K.K. */
```

```
product ALLIEDTELESIS LA_PCM 0x0002 Allied Telesis LA-PCM

/* Archos */
product ARCHOS ARC_ATAPI 0x0043 MiniCD
```

product ####### ## ######## ## ### ###### ####, ######## #### #####. #### ## ######## ## ### ####### ####, ##### ## #### ## ### ###### ### ###### ## # ##### # ##########, ### ### #### ##### #### # ######. ## ##### ### #######, ### ### ####### ## ### #### #### ####### ### #### ########## ### 0xffffffff ### 0xffff. #######, ##### ## # ###### ########### ## ### ###### ######. #### ###### ######### ## ### #### ## #######, ##### ######### ###### ### ###### #### ######### # ###### #### ### ##–### ######## ### #######, ### ## ### ## #### ## ### #### ##### ##### #### ## ####–### ############. ### ######## ### ## ############ ##### ## ############, #### ######### ##### ## ####### ##. #### #### # # ####### ###### #### ############## ####### ### ##### ## # ##### #### ####### #######.

#######, ### #### ### ## ### ###–######### ####### ### ### ## -1, ####### ###### ######## ###### ## ### ####### ###### ### ####. ##### ##### ### # ###########, ##### ##### #### ## ######. ###–###### ### ###### ## ######### ## ### ##### ####### ## ### ####.

#######. #### ######### ###### ## # ###### ######### #### ### ####### #######:

```
product ADDTRON AWP100
 { -"Addtron", -"AWP-100&spWireless&spPCMCIA", -"Version&sp01.02", NULL -}
product ALLIEDTELESIS WR211PCM { -"Allied&spTelesis&spK.K.", -"WR211PCM", NULL, -
NULL -} Allied Telesis WR211PCM
```

product ####### ## ######## ## ### ###### #### ### ### #### ####, #### ## ## ### ###### ####### ## ### ####. #### ### ###### ######## #### #### #### ###–####. ##### ## # {} ########, ######## ## # ###### ## #######. ##### ####### #####–##### ## ### ######, #######, ### ##### ########### #### ## ####### ## # ###_#### #####. ##### ####### ### ######## ## ### ####### #### ######### pccarddevs.h ## ##–##### &## #### # #### #####. #### ####### #### #### ### ############# #### ## ### ##### ###### ## #######. ### ####### ##### #### ######## # ### #####. ## ###### ### ####### ### ####### ###### ###### #### ## ######## ### ### ######### ## ### ####. ######### ####### #### #### #### ######### ######## ## ### #### ## ### ##### #### ### ####, ## ##### #### #### ########### #### ##### ## ###### ### ####### #### # ####### #### ## ### ## #### #######. ######### ## ## ######### #### # ###### ###### ## #### #### ######### ##### ##### ### #### ##### ### ## ###### ############## (############, #####, ### ## #####). #### ## ### ## ####–#### ## ############## ### #### ## ##### #### ##### ##### ### ###### #### ### #### ############/####### ####. ####### ########## ######## ## ### ######### ## #### ####.

16.1.3. ###### #####

#######, ### #### ##–######## ### ##### ###/## ##### ######## #### #### ####### ####. ## ## ######–##### # ###### ## ####### 5.# ####### ##### ## # ############# ##### ### ####### ####### ## ####. ##### #### ### ############### ## #########, ## ######### ###–#### #### ## ######### ####.

```
static const struct pccard_product wi_pccard_products[] = {
 PCMCIA_CARD(3COM, 3CRWE737A, 0),
 PCMCIA_CARD(BUFFALO, WLI_PCM_S11, 0),
 PCMCIA_CARD(BUFFALO, WLI_CF_S11G, 0),
 PCMCIA_CARD(TDK, LAK_CD011WL, 0),
 { NULL -}
};

static int
wi_pccard_probe(dev)
 device_t dev;
{
 const struct pccard_product *pp;

 if ((pp = pccard_product_lookup(dev, wi_pccard_products,
     sizeof(wi_pccard_products[0]), NULL)) -!= NULL) {
  if (pp->pp_name -!= NULL)
   device_set_desc(dev, pp->pp_name);
  return (0);
 }
 return (ENXIO);
}
```

#######. ## ###### #####, ### #### ### #### (## ## ## ### foo_pccard_probe() ## #### ## foo_pccard_match()). ### ######## pccard_product_lookup() ## # ########### ######## #### ##### ### ##### ### ####### # ####### ## ### ##### ##### #### ## #######. #### ####### ### ### #### ######### ## ###### ########## ########### ##### #### ##### ## ### #### ## ### ######, ## ##### ### ## #### ######## ## ### #####. ### #### ########### ## #### #### ### ## ### ##### #### #### # struct pccard_product ## ### ##### #######.

####### ## ### ##### wi_pccard_products, ### ####### #### ### ### ####### ### ## ### #### PCMCIA_CARD(*foo*, *bar*, *baz*). ### *foo* #### ## ### ############ ## #### pccarddevs. ### *bar* #### ## ### ####### ##. *baz* ## ### ######## ######## ###### ### #### ####. #### ####### ### #### ######## #########, ### #### ### ## ############ ######## 1 #### ######## 0 ## ######. ### ### ### PCMCIA_CARD_D, ##### ######## ### ##–#### ########### #### pccarddevs. ### ### #### ### PCMCIA_CARD2 ### PCMCIA_CARD2_D ##### ### #### #### ### #### ## ##### #### ### ####### ### ############ #######, ## ### #### ### ####### ############ ### ##### ### ########### #### ########–### #######.

16.1.4. ####### ## ###

######, ### #### ##### ###### ### ############### ########### #### ### ######. ### ####### ### ## ## #### ## ## ###### ### ###### #### # ## #### ## ## #### ### ##### devinfo -v. ###### ######:

```
    cbb1 pnpinfo vendor=0x104c device=0xac51 subvendor=0x1265 subdevice=0x0300 -
class=0x060700 at slot=10 function=1
      cardbus1
      pccard1
        unknown pnpinfo manufacturer=0x026f product=0x030c cisvendor="BUFFALO" -
cisproduct="WLI2-CF-S11" function_type=6 at function=0
```

manufacturer ### product ### ### ####### ### ### #### #######, ##### cisvendor ### cisproduct ### ### ####### ########### ####### #### ### ###.

######, ##### ### ## ######### ## ##### ##### ## ####. ### ##### #### ### #### ######## ############# ### ### ####### ## #### #######. ### ###### ## #######, ##### ## ### ####### ### ## #####:

```
vendor BUFFALO   0x026f BUFFALO (Melco Corporation)
```

####. ####### ## ####:

```
/* BUFFALO */
product BUFFALO WLI_PCM_S11 0x0305 BUFFALO AirStation 11Mbps WLAN
product BUFFALO LPC_CF_CLT 0x0307 BUFFALO LPC-CF-CLT
product BUFFALO LPC3_CLT 0x030a BUFFALO LPC3-CLT Ethernet Adapter
product BUFFALO WLI_CF_S11G 0x030b BUFFALO AirStation 11Mbps CF WLAN
```

######, ## ### #### ### #### ##### ## pccarddevs:

```
product BUFFALO WLI2_CF_S11G 0x030c BUFFALO AirStation ultra 802.11b CF
```

########, ### #### ### ## ##### ## ### ######. #### ## # ###### ######### ## ###### ### ####:

```
static const struct pccard_product wi_pccard_products[] = {
 PCMCIA_CARD(3COM, 3CRWE737A, 0),
 PCMCIA_CARD(BUFFALO, WLI_PCM_S11, 0),
 PCMCIA_CARD(BUFFALO, WLI_CF_S11G, 0),
+ PCMCIA_CARD(BUFFALO, WLI_CF2_S11G, 0),
 PCMCIA_CARD(TDK, LAK_CD011WL, 0),
 { NULL -}
};
```

#+# ## ### #### ###### ### #### #### # #####, ### #### ## ###### ## ######### ### ####. ## ### ### ## ## ### ###### ######. #### ### #### ##### ### ####, ### ### ######### #### ###### ## ###### ### #### ##. ## ### ###### ## ########## ### #####, ###### ###### # #####. ## ## #### ### ####,

#####. ## ### ###### ## ### ########## ## ###, ### #### #### ######### ##### ### ###### ####### #### ####.

#########, ### ########## ####### ## ## #######, #### ### ### ###### ### ####### ## ### ####. #######, ##### ### #### ##### ###### ###### ## ## ##########. pccarddevs #### ## ######### ## ### #### #####. #### pccarddevs.h #### ## ########### ### ######### ## # ###### ####, ######## #### ### ##### ######### ### ## ## ### ###### ####. #######, ###### ### ######### ## ### ######.

16.1.5. ########## # ###

########. #######, ###–### #### ## # ## ### #### ### ###### ### ## ###### ### ### #######. #### ####### #### ####### ### ### ####. #### ########## # ##, ## ## ########### ## ####### ### pccardevs.h ##### ## ### #####, ##### ##### #### ## ###########. ## ## ######### ## ####### # ########### ## ### ######, ## #### ## ### ####### ## ### ###### ######. ## ### ## ### #### ### ####, ### ###99 ## ### ####, ### ### ###### #### ###### ###99 ########### ##### #############. ########## ###### ### ###### ###99, ### ####### #### ### ####### ### ##### ### ###### ### #### #### ####.

###.

#########

#############

[1] ######## #### ########, ##### ######, ####### # ######, ### #### # ####–######. ######### © 1996 ############## ########## #######, ###.. 0#201#54979#4. ############## ########## #######, ###.. *The Design and Implementation of the 4.4 BSD Operating System*. 1#2.

#####