UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Departament d'Arquitectura de Computadors

Conceptes Avançats de Sistemes Operatius

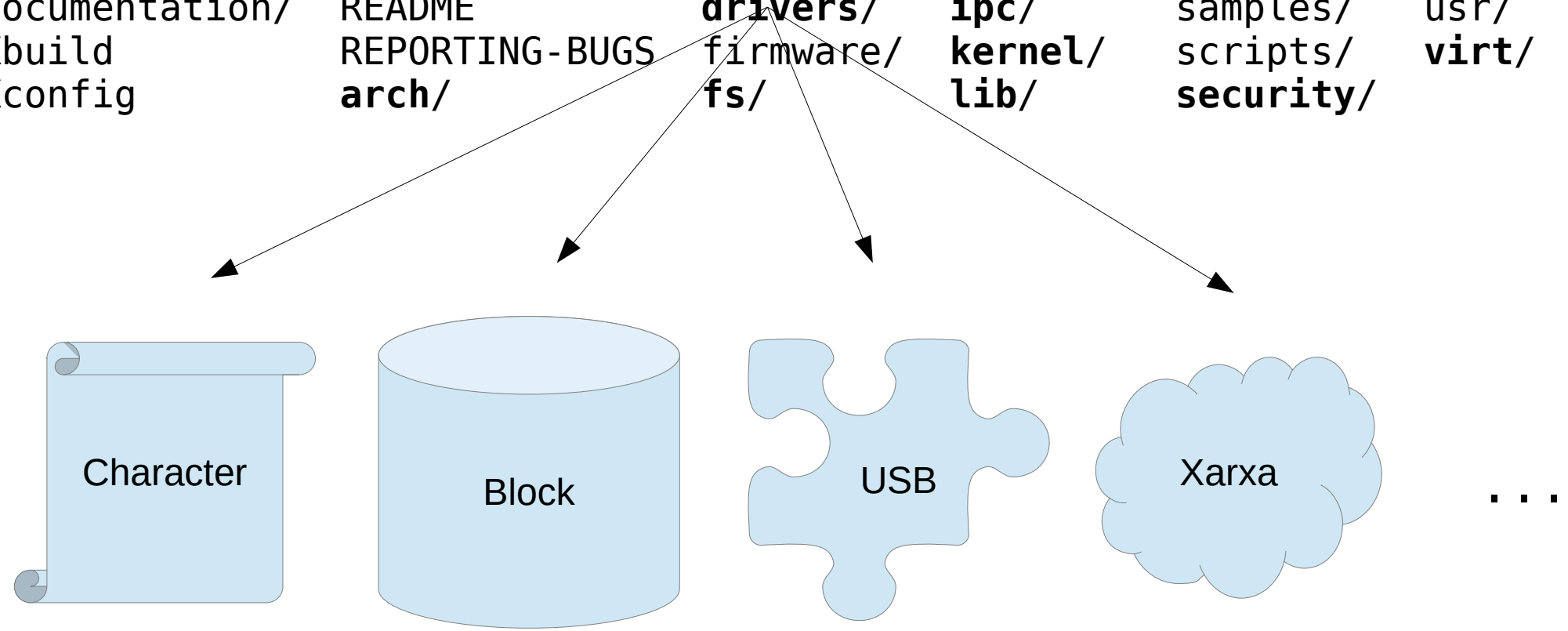
Curs 2023/24 Q2

Gestió de dispositius

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

Estructura del kernel

```
COPYING             MAINTAINERS     block/      include/    mm/         sound/
CREDITS             Makefile        crypto/     init/       net/        tools/
Documentation/      README          drivers/    ipc/        samples/    usr/
Kbuild              REPORTING-BUGS  firmware/   kernel/     scripts/    virt/
Kconfig             arch/           fs/         lib/        security/
```



Motivació

- Varietat de dispositius

Acks: ASUSTek Computer Inc.
Xilinx Inc., Seagate
Dell, Logitech

Índex

- **Dispositius de caràcter**
- Dispositius de block
- Dispositius USB
- Dispositius PCIe
- Dispositius de xarxa

https://www.kernel.org/doc/Documentation/rtc.txt

Dispositius de caràcter

- Permeten l'accés a la informació caràcter a caràcter

```
crw-r----- 1 root kmem      1,   1 Oct 19 07:05 mem
crw-r----- 1 root kmem      1,   2 Oct 19 07:05 kmem
crw-rw-rw- 1 root root      1,   3 Oct 19 07:05 null
crw-r----- 1 root kmem      1,   4 Oct 19 07:05 port
crw-rw-rw- 1 root root      1,   5 Oct 19 07:05 zero
crw-rw-rw- 1 root root      1,   7 Oct 19 07:05 full
crw-rw-rw- 1 root root      1,   8 Oct 19 07:05 random
crw-rw-rw- 1 root root      1,   9 Oct 19 07:05 urandom
crw------- 1 root root      1,  11 Oct 19 07:05 kmsg

crw------- 1 root root     10,   1 Oct 19 07:05 psaux
crw-rw-rw- 1 root root     10, 229 Oct 11 07:46 fuse
crw-r----- 1 root root     13, 33 Oct 19 19:36 mouse1
crw-r----- 1 root root     13, 34 Oct 20 22:26 mouse2

crw-rw---- 1 root video    81,   0 Oct 19 07:05 video0
crw-rw---- 1 root video    29,   0 Mar 29 09:16 fb0
crw-r--r-- 1 root root    254,   0 Oct 19 07:05 rtc0
```

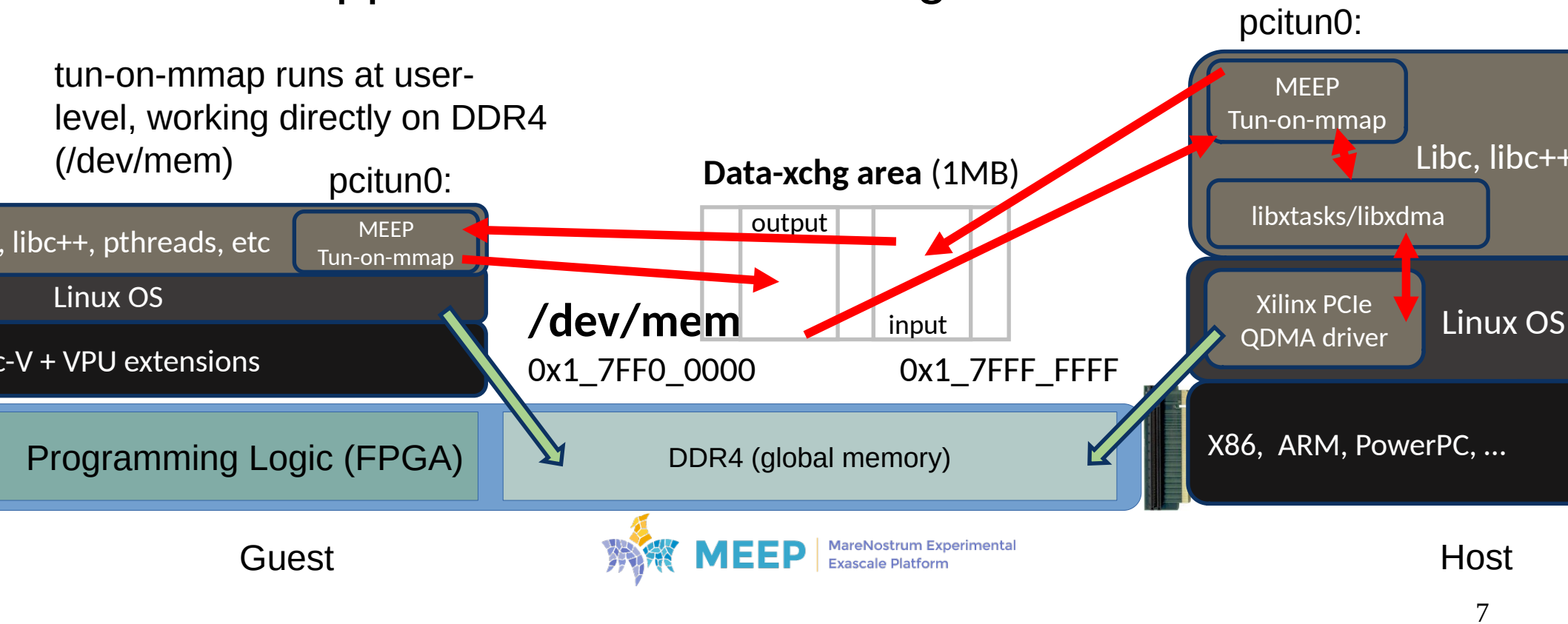
Dispositius de caràcter

- /dev/mem
  - Imatge de la memòria principal de l'ordinador
    - Adreces físiques!
  - Utilitat?

```
0051100    0000    0205    005b    0000    5441    2041    2020    2020
         \0  \0 005 002   [  \0  \0  \0   A   T   A
0051120    5453    3035    4c30    304d    3030    532d    4853    2d44
         S   T   5   0   0   L   M   0   0   0   -   S   S   H   D   -
0051140    494c    3556    0000    0000    0000    0000    0000    0000
         L   I   V   5  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0
*
0051240    0000    0000    0000    0000    0c08    d887    8800    ffff
         \0  \0  \0  \0  \0  \0  \0  \0  \b  \f 207 330  \0 210 377 377
0051260    8008    d88d    8800    ffff    4508    d530    8800    ffff
         \b 200 215 330  \0 210 377 377  \b   E   0 325  \0 210 377 377
0051300    0000    0000    0000    0000    196a    0013    0716    6354
         \0  \0  \0  \0  \0  \0  \0  \0   j 031 023  \0 026  \a   T   c
0051320    5f6c    6547    4974    746e    7246    6d6f    624f    2e6a
         l   _   G   e   t   I   n   t   F   r   o   m   O   b   j   .
0051340    2e33    7a67    0000    0000    0000    0000    0000    0000
         3   .   g   z  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0
```

Dispositius de caràcter

- Utilitat /dev/mem
  - Networking through PCIe
  - Input/output circular buffers between host and FPGA
  - Mmapped from /dev/mem in guest Risc-V



Dispositius de caràcter

- port
  - Accés físic als ports d'entrada/sortida
    - 96 → teclat
    - ...
- kmem
  - Memòria del kernel
    - Com podem trobar l'adreça on comença?

```
// Una mica de màgia
$ sudo dd if=/dev/port skip=96 count=1 bs=1 status=none | od -t x1
0000000 1c
0000001
```

```
root@pcxavim5:# od -x /dev/kmem
od: /dev/kmem: read error: Bad address

root@pcxavim5:# read_kmem 0xffff_ffff_c000_0000
0f 1f 44 00 00 8b 07 55 48 89 e5 83 f8 13 77 20...
```

Killing off /dev/kmem (by Jonathan Corbet - April 5, 2021)
https://lwn.net/Articles/851531/

Dispositius de caràcter

- rtc0

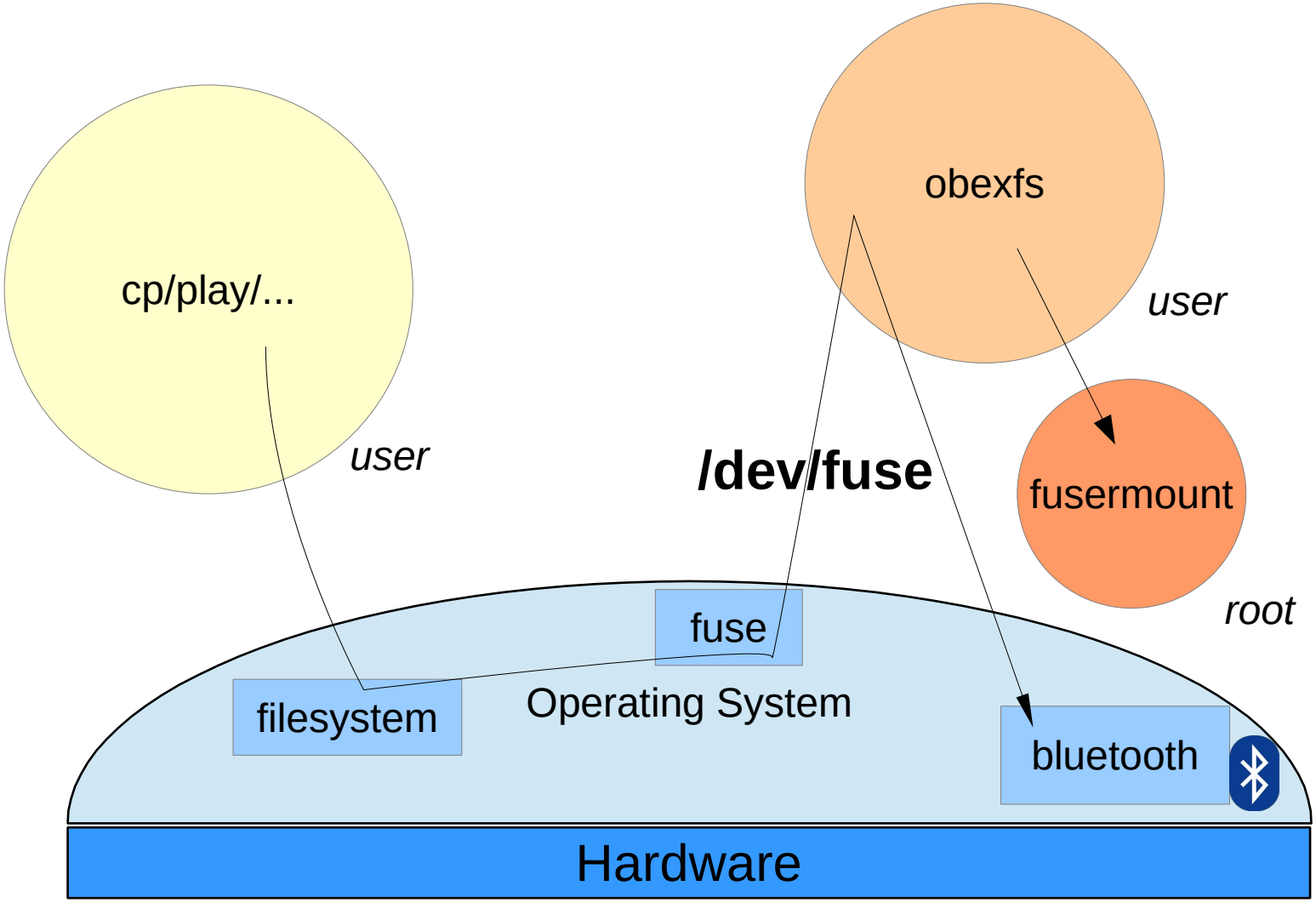
```
bash-4.3$ ./rtctest

                RTC Driver Test Example.

Counting 5 update (1/sec) interrupts from reading /dev/rtc0: 1 2 3 4 5
Again, from using select(2) on /dev/rtc: 1 2 3 4 5

Current RTC date/time is 9-4-2015, 20:00:02.
Alarm time now set to 20:00:07.
Waiting 5 seconds for alarm... okay. Alarm rang.

Periodic IRQ rate is 128Hz.
        /* The frequencies 128Hz, 256Hz, ... 8192Hz are only allowed for root. */
Counting 20 interrupts at:
2Hz:     1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
4Hz:     1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
8Hz:     1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
16Hz:    1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
32Hz:    1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
64Hz:    1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20RTC_IRQP_SET ioctl: Permission
denied
```

Dispositius de caràcter

- psaux, mouse, mice
- fuse
  - Per comunicar la llibreria de FUSE amb el driver del kernel
  - El descriptor de fitxer obtingut s'usa per relacionar
    - Punt de muntatge
    - Aplicació que s'ocupa del sistema de fitxers en espai d'usuari

OBEX + FUSE

- Comunicació servidor - driver



Dispositius de caràcter

- També s'usen pels terminals

```
crw-rw---- 1 root dialout   4,  64 Oct 19 07:06 ttyS0
crw-rw---- 1 root dialout   4,  65 Oct 19 07:05 ttyS1
crw-rw---- 1 root dialout   4,  67 Oct 19 07:05 ttyS3
crw-rw---- 1 root dialout   4,  66 Oct 19 07:05 ttyS2

crw--w---- 1 root tty       4,   0 Oct 19 07:05 tty0
crw-rw---- 1 root tty       4,   1 Oct 19 07:05 tty1
crw-rw-rw- 1 root tty       5,   0 Oct 21 11:04 tty
crw------- 1 root root      5,   1 Oct 19 07:05 console
crw-rw-rw- 1 root tty       5,   2 Oct 21 11:49 ptmx

crw--w---- 1 xavim tty  136,  0 Oct 19 07:06 0
crw------- 1 xavim tty  136,  1 Oct 21 11:15 1
crw------- 1 xavim tty  136,  2 Oct 21 11:48 2
...
```

Per llegir:

Terminals, Gettys and Display Managers: http://moi.vonos.net/linux/ttys/

Special Devices and File Systems http://www.cs.rutgers.edu/~pxk/416/notes/14-specialfs.html

Virtual terminal

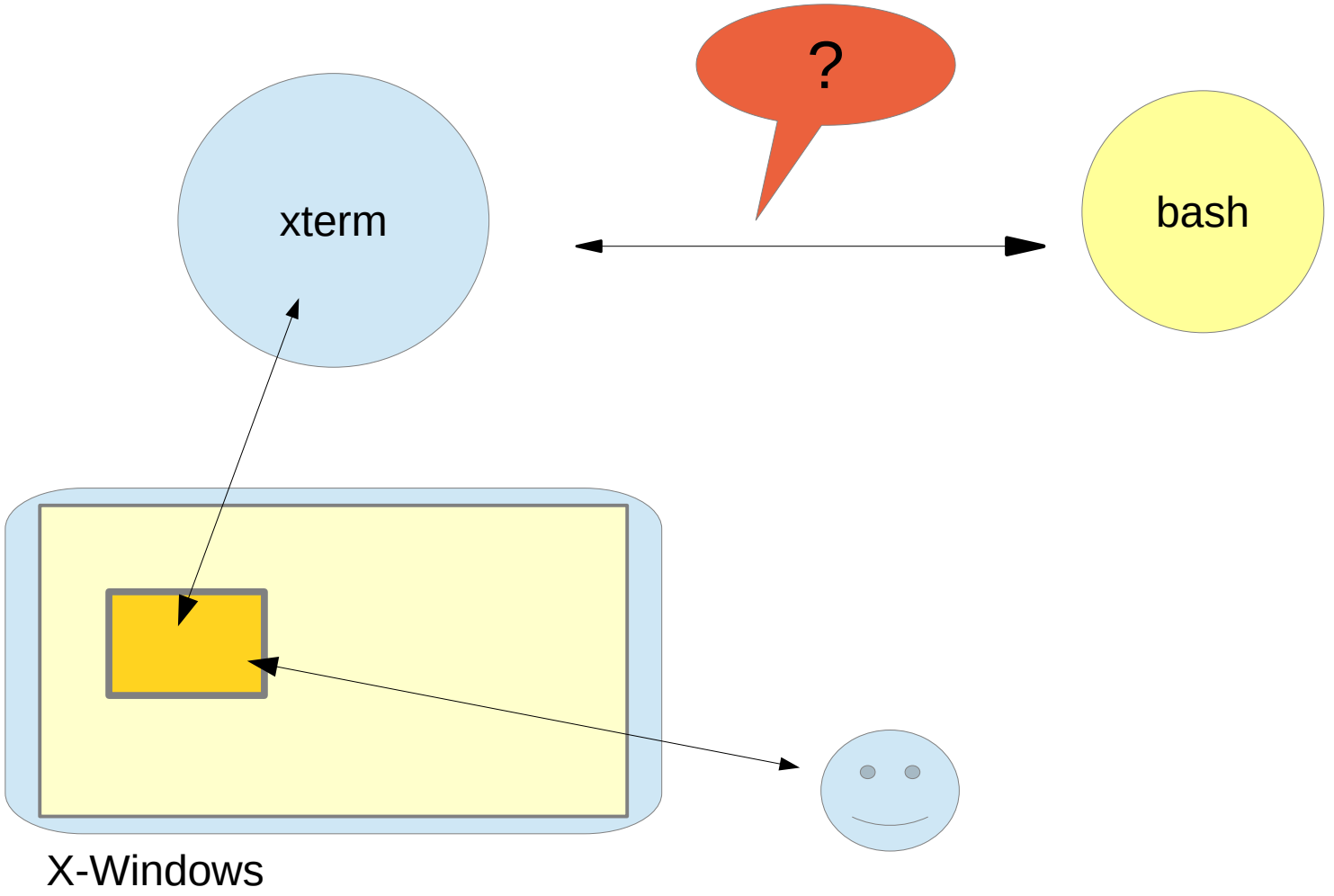
- /dev/vcs[0-63]
  - Conté la memòria amb els caràcters i atributs de les pantalles de text
- /dev/vcsa[0-63]
  - Igual que vcs, incloent a més, la mida de la pantalla
- Permeten fer cut&paste

Graphics

- A través de
  - /dev/drm
  - /dev/dri/card0
  - /dev/dri/control
  - /dev/dri/render
    - De vegades necessaris per executar CUDA i OpenCL a la GPU
- Amb accés directe a memòria a través de
  - mmap sobre /dev/mem

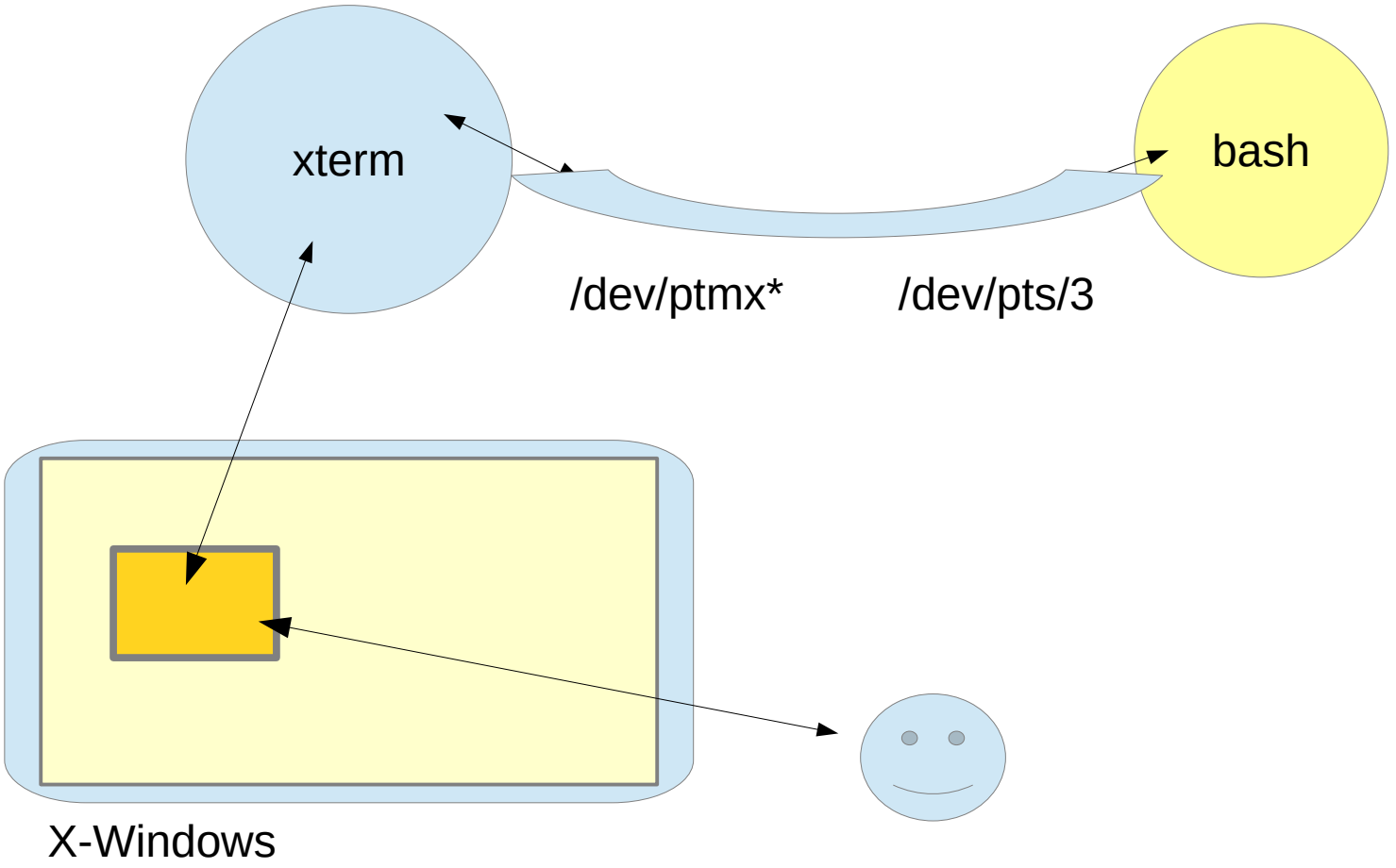
Pseudo-terminals

- Exemple: xterm



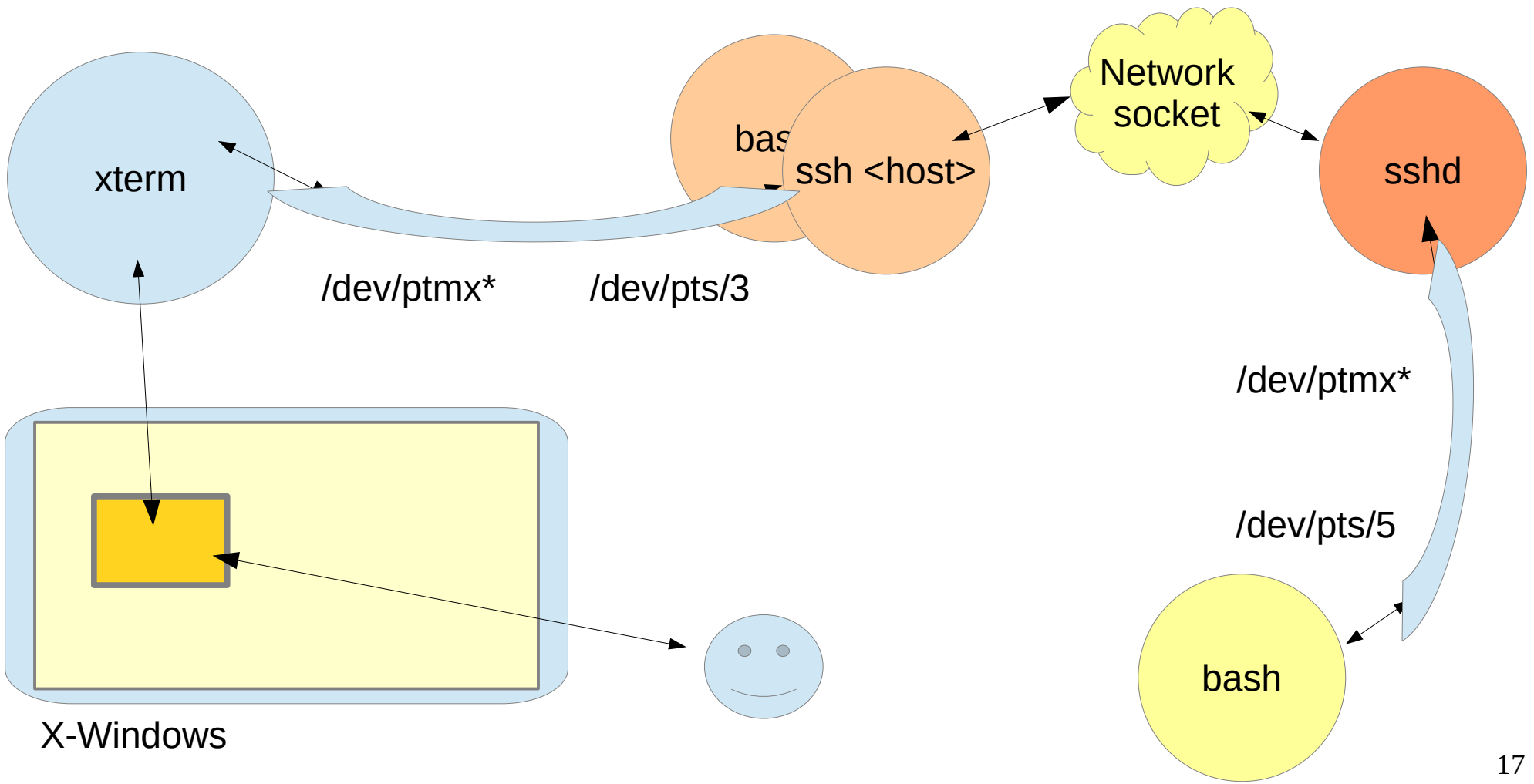
Pseudo-terminals

- Exemple: xterm

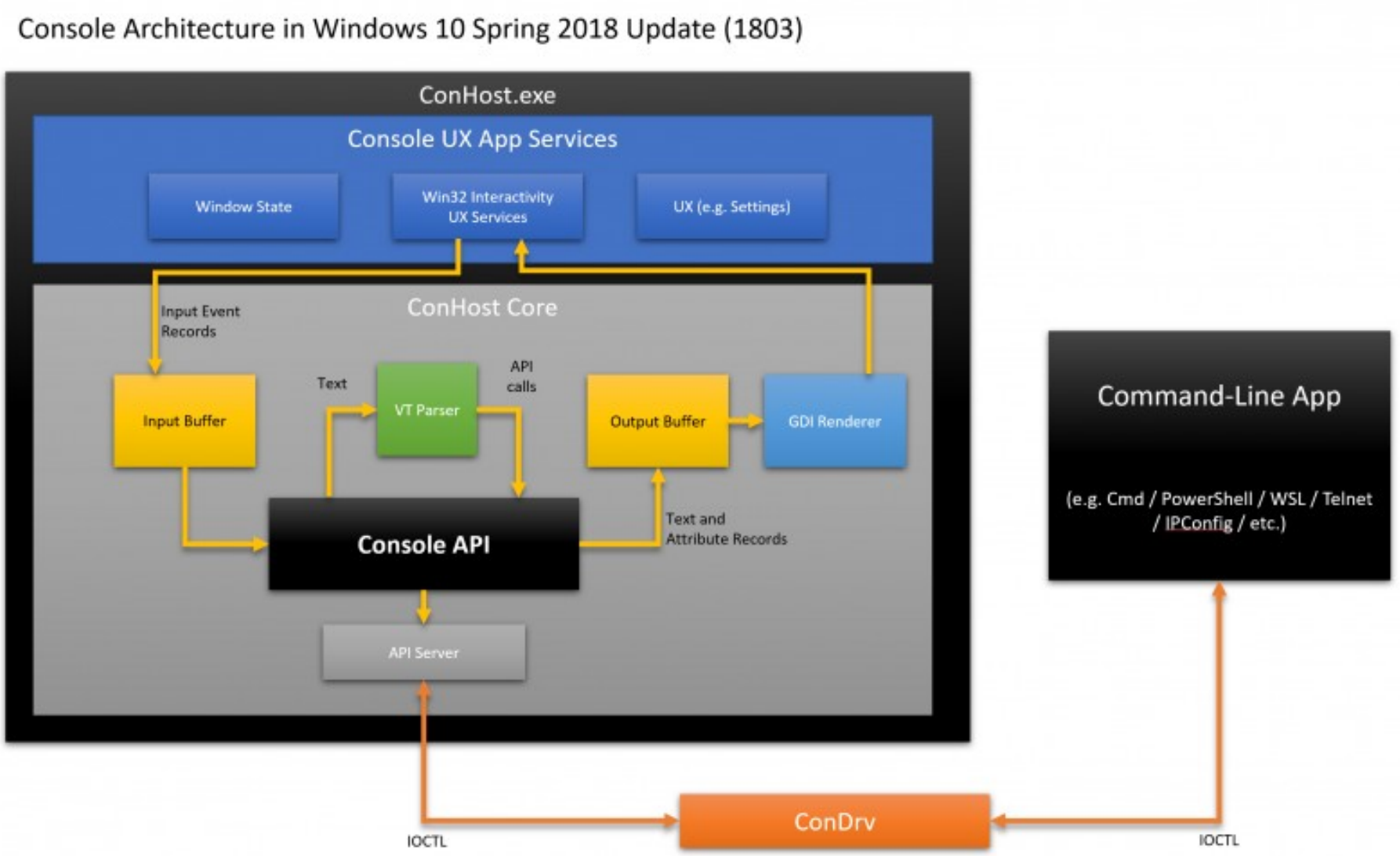


Pseudo-terminals

- Exemple: ssh



Windows 10



Windows Command-Line: Inside the Windows Console
https://devblogs.microsoft.com/commandline/windows-command-line-inside-the-windows-console/

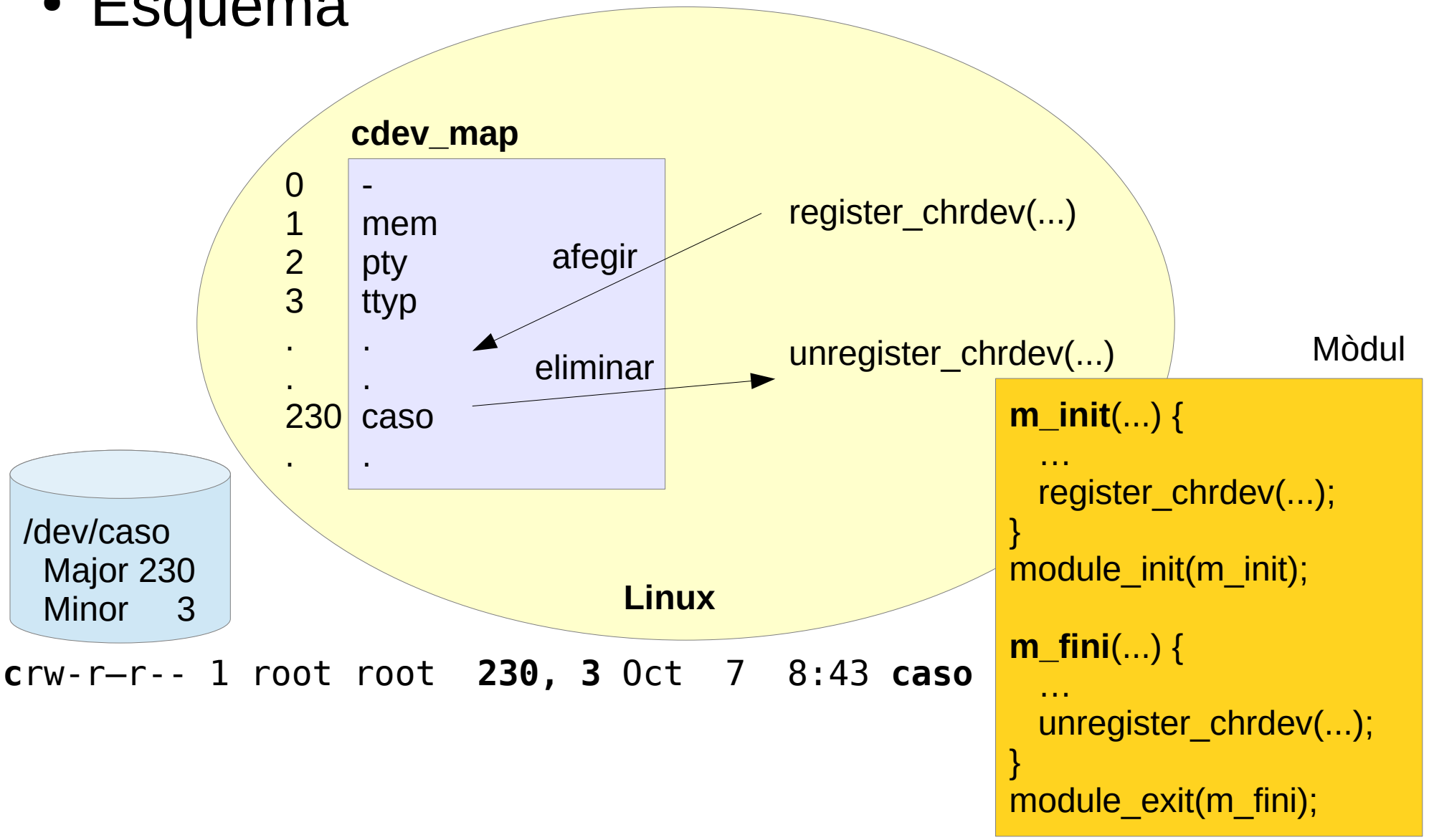
Drivers de tipus caràcter

- Preparació
  - Seleccionar major/minor del dispositiu
    - Exemple: 200, 1
  - Programar el controlador en un mòdul extern
    - Inicialització
      - Enregistrar el dispositiu
        - register_chrdev(...)
    - Finalització
      - Desenregistrar el dispositiu
        - unregister_chrdev(...)

https://www.kernel.org/doc/ => Linux Device Driver Tutorial (Youtube)

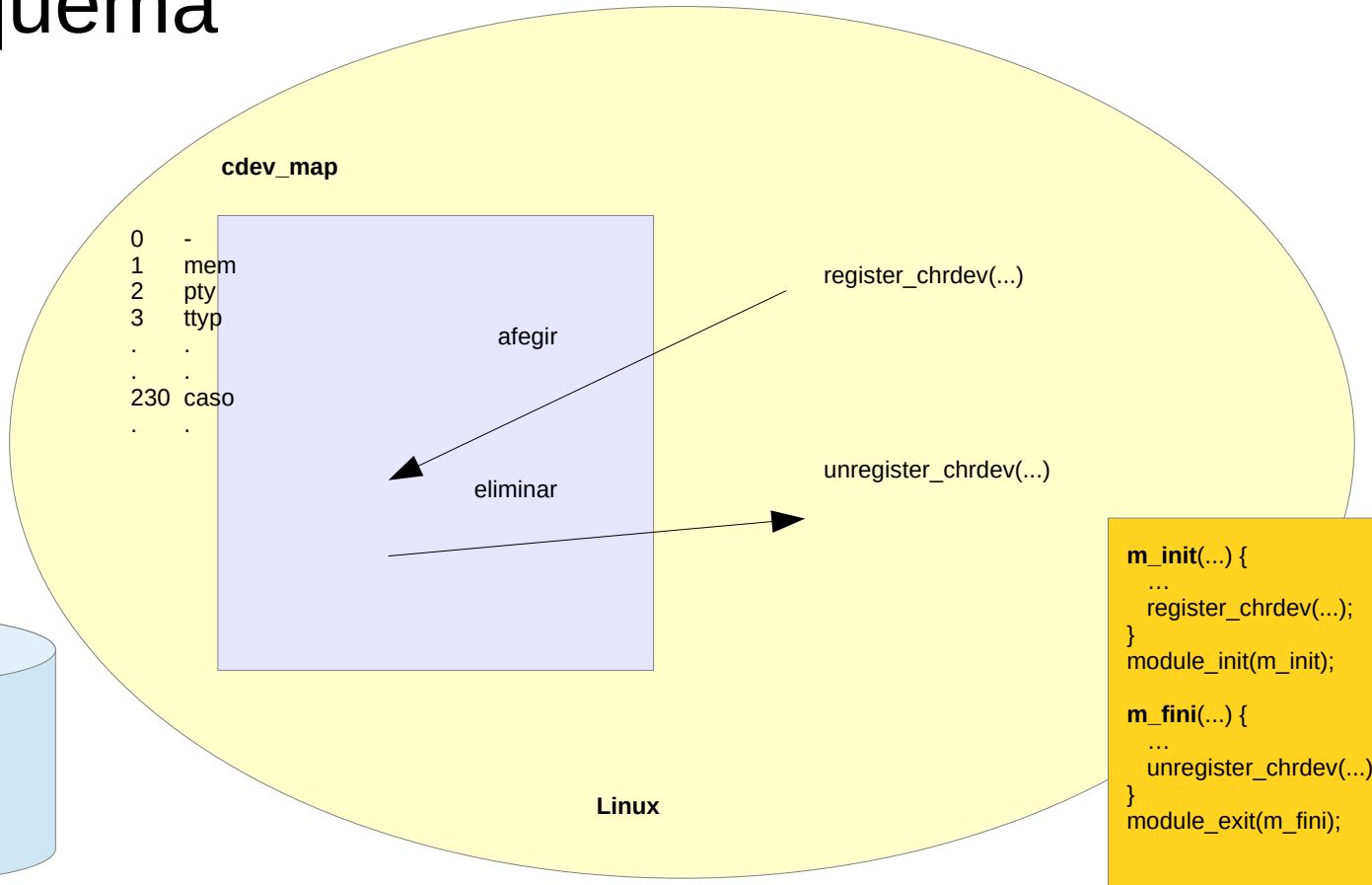
Drivers de tipus caràcter

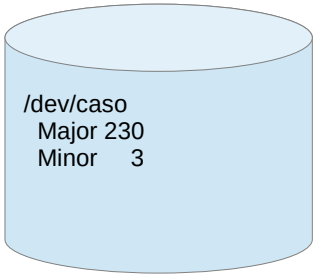
- Esquema



Drivers de tipus caràcter

- Esquema






```
crw-r—r-- 1 root root  230, 3 Oct  7  8:43 caso
```

Mòdul

```
m_init(...) {
   …
   register_chrdev(...);
}
module_init(m_init);

m_fini(...) {
   …
   unregister_chrdev(...);
}
module_exit(m_fini);
```

Drivers de tipus caràcter

- Inicialització / finalització

```
static int mychardrv_init(void) {
...
  res = register_chrdev(MY_MAJOR, "mychardrv", &mychardrv_fops);
  if (res < 0) {
    printk("unable to get major %d for mychardrv\n", MY_MAJOR);
    return res;
  }
...
}
```

```
static void mychardrv_exit(void) {
...
  unregister_chrdev(MY_MAJOR, "mychardrv");
...
}
```

Drivers de tipus caràcter

- Operacions

```
struct file_operations {
     struct module *owner;
     loff_t (*llseek) (struct file *, loff_t, int);
     ssize_t (*read) (struct file *, char __user *, size_t, loff_t *);
     ssize_t (*write) (struct file *, const char __user *, size_t, loff_t *);
     ssize_t (*aio_read) (struct kiocb *, const struct iovec *, unsigned long , loff_t);
     ssize_t (*aio_write) (struct kiocb *, const struct iovec *, unsigned long, loff_t);
     int (*readdir) (struct file *, void *, filldir_t);
     unsigned int (*poll) (struct file *, struct poll_table_struct *);
     long (*unlocked_ioctl) (struct file *, unsigned int, unsigned long);
     long (*compat_ioctl) (struct file *, unsigned int, unsigned long);
     int (*mmap) (struct file *, struct vm_area_struct *);
     int (*open) (struct inode *, struct file *);
     int (*flush) (struct file *, fl_owner_t id);
     int (*release) (struct inode *, struct file *);
     int (*fsync) (struct file *, loff_t, loff_t, int datasync);
     int (*aio_fsync) (struct kiocb *, int datasync);
     int (*fasync) (int, struct file *, int);
     int (*lock) (struct file *, int, struct file_lock *);
     ssize_t (*sendpage) (struct file *, struct page *, int, size_t, loff_t *, int);
     ...
```

include/linux/fs.h

Drivers de tipus caràcter

- Operacions

```
struct file_operations {
     ...
        unsigned long (*get_unmapped_area)(struct file *, unsigned long,
                              unsigned long, unsigned long, unsigned long);
        int (*check_flags)(int);
        int (*flock) (struct file *, int, struct file_lock *);
        ssize_t (*splice_write)(struct pipe_inode_info *, struct file *,
                              loff_t *, size_t, unsigned int);
        ssize_t (*splice_read)(struct file *, loff_t *, struct pipe_inode_info *,
                                   size_t, unsigned int);
        int (*setlease)(struct file *, long, struct file_lock **);
        long (*fallocate)(struct file *file, int mode, loff_t offset,
                            loff_t len);
};
```

linuxinclude/linux/fs.h

Drivers de tipus caràcter

- Exemple d'estructura amb les operacions
  - Rellotge d'alta resolució – HPET
    - High Precision Event Timer

```
static const struct file_operations hpet_fops = {
        .owner = THIS_MODULE,
        .llseek = no_llseek,
        .read = hpet_read,
        .poll = hpet_poll,
        .unlocked_ioctl = hpet_ioctl,
#ifdef CONFIG_COMPAT
        .compat_ioctl = hpet_compat_ioctl,
#endif
        .open = hpet_open,
        .release = hpet_release,

        .fasync = hpet_fasync,
        .mmap = hpet_mmap,
};
```

linux/drivers/char/hpet.c

Drivers de tipus caràcter

- Identificació del mòdul responsable de les operacions d'un dispositiu
  - Tots els mòduls defineixen una variable estàtica definida durant el procés de compilació del mòdul
    - struct module __this_module;
  - I la poden referenciar usant la macro
    - #define THIS_MODULE (&__this_module)
- Si un dispositiu és part del codi del kernel, llavors s'indica que no pertany a cap mòdul
    - #define THIS_MODULE ((struct module *)0)

linuxinclude/linux/export.h

Tractament d'errors

- Codis d'error definits a <linux/errno.h>
  - asm-generic/errno-base.h
  - asm-generic/errno.h
    - Son números positius
- A l'interior del kernel els errors es passen com a números negatius
  - I a usuari són els números positius

```
EPERM            1
ENOENT           2
ESRCH            3
…
EDEADLK         35
ENAMETOOLONG    36
…
```

```
res = -EPERM;
return res;
```

Còpia d'informació a/de nivell usuari

- Copiar cap a l'espai de l'usuari
  - Arguments: destí (u), font (s), quantitat en bytes

```
static inline long copy_to_user(       void __user * to,
                                const void         * from,
                                unsigned long        n );
```

- Obtenir informació de l'espai d'usuari
  - Arguments: destí (s), font (u), quantitat en bytes

```
static inline long copy_from_user(     void         * to,
                                const void __user * from,
                                unsigned long        n );
```

#include <linux/uaccess.h>

Índex

- Dispositius de caràcter
- **Dispositius de block**
- Dispositius USB
- Dispositius PCIe
- Dispositius de xarxa

Drivers de tipus block

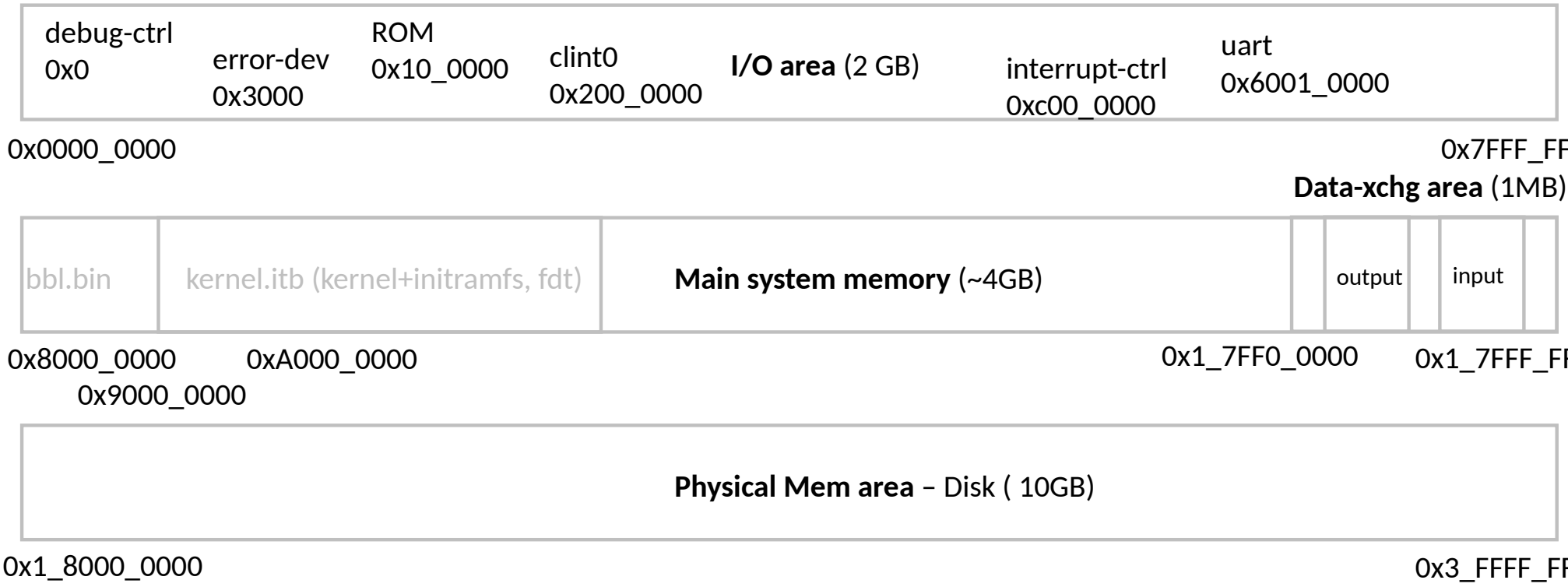
- Ram disks, SATA, SCSI, IDE…

```
brw-rw---- 1 root disk      1,   0 Oct 19 07:05 ram0
brw-rw---- 1 root disk      1,   1 Oct 19 07:05 ram1
…
brw-rw---- 1 root disk      X,   Y Apr 22 12:43 pmem0
…
brw-rw---- 1 root disk      1,  15 Oct 19 07:05 ram15

brw-rw---- 1 root disk      7,   0 Oct 20 12:21 loop0
brw-rw---- 1 root disk      7,   1 Oct 19 07:05 loop1
…
brw-rw---- 1 root disk      8,   0 Oct 19 07:05 sda
brw-rw---- 1 root disk      8,   1 Oct 19 07:05 sda1
...
```

- Si hi ha particions als ramdisks s'anomenen:
  - /dev/ram2p1
  - /dev/ram2p2
  - ...
  - /dev/pmem0p1
  - /dev/pmem0p2
  - ...

Drivers de tipus block

- Risc-V memory map
  - I/O area, Main memory, and Physical RAM area (disk)



Boot time only

System up and running

MEEP MareNostrum Experimental Exascale Platform

Drivers de tipus block

- Per donar suport a discos
  - Inicialització
    - res = register_blkdev(sd_major(i), "sd");
    - No rep les operacions...
  - struct block_device_operations
    - Cal obtenir un "struct gendisk" i omplir els camps

```
struct gendisk * disk = alloc_disk(max_partitions – 1);
disk->major           = DISK_MAJOR;
disk->fops            = disk_fops;
…
blk_register_region(MKDEV(DISK_MAJOR, 0), 1 << MINORBITS,
                    THIS_MODULE, brd_probe, NULL, NULL);
```

```
static struct kobject *xrd_probe(dev_t dev, int *part, void *data);
```

Drivers de tipus block

- blk_register_region
  - Reserva el rang de números minor [0:range-1] pel dispositiu
- request_manager (xdr_q, xdr_make_request)
  - relaciona el dispositiu amb la funció de tractament de les peticions de lectura/escriptura
- xrd_probe obté el dispositiu (get_disk) i allà hi troba la taula d'operacions (xrd_fops)

Drivers de tipus block

- Per donar suport a discos

```
struct block_device_operations {
    int (*open) (struct block_device *, fmode_t);
    int (*release) (struct gendisk *, fmode_t);
    int (*ioctl) (struct block_device *, fmode_t, unsigned, unsigned long);
    int (*compat_ioctl) (struct block_device *, fmode_t, unsigned, unsigned long);
    int (*direct_access) (struct block_device *, sector_t,
                                                 void **, unsigned long *);
    unsigned int (*check_events) (struct gendisk *disk,
                                  unsigned int clearing);
    /* ->media_changed() is DEPRECATED, use ->check_events() instead */
    int (*media_changed) (struct gendisk *);
    void (*unlock_native_capacity) (struct gendisk *);
    int (*revalidate_disk) (struct gendisk *);
    int (*getgeo)(struct block_device *, struct hd_geometry *);
    /* this callback is with swap_lock and sometimes page table lock held */
    void (*swap_slot_free_notify) (struct block_device *, unsigned long);
    struct module *owner;
};
```

include/linux/
blkdev.h

Altre suport per discos

- SCSI disks

```
err = scsi_register_driver(&sd_template.gendrv);
```

- SATA disks, Compaq SMART2 controllers

```
err = pci_register_driver(&carm_driver);
```

Suport per arguments als mòduls

- Un mòdul pot rebre arguments en ser carregat
  - insmod <modul.ko> argument=N
- Els arguments es defineixen dins el mòdul

```
static int max_part;    // màxim número de particions per disc
module_param(max_part, int, S_IRUGO);
MODULE_PARM_DESC(max_part, "Maximum number of partitions per RAM disk");
```

- Es poden veure amb
  - modinfo <modul.ko>

Suport per arguments als mòduls

```
$ modinfo /lib/modules/5.4.0-72-generic/kernel/drivers/block/floppy.ko
filename:       /lib/modules/5.4.0-72-generic/kernel/drivers/block/floppy.ko
alias:          block-major-2-*
license:        GPL
author:         Alain L. Knaff
srcversion:     74A29049FDCC95BC5F34852
alias:          acpi*:PNP0700:*
alias:          pnp:dPNP0700*
depends:
retpoline:      Y
intree:         Y
name:           floppy
vermagic:       5.4.0-72-generic SMP mod_unload modversions
signat:         PKCS#7
signer:
sig_key:
sig_hashalgo:   md4
parm:           floppy:charp
parm:           FLOPPY_IRQ:int
parm:           FLOPPY_DMA:int
```

Arguments

Suport per gestió de memòria

- Interna al kernel <linux/slob_def.h> <linux/slub_def.h>
  - Simple List Of Blocks (discontinuat) (simple, ràpid, actiu)
    - Per embedded, però pateix de fragmentació interna
    - static inline void ***kmalloc**(size_t size, gfp_t flags);
    - void **kfree**(const void *block);
  - Flags
    - Un d'entre: GFP_USER, GFP_KERNEL, …
    - Addicionals: GFP_ZERO | GFP_THISNODE | GFP_DMA …
  - Suport per màquines NUMA
    - static inline void ***kmalloc_node**(size_t size, gfp_t flags, int node);

https://www.kernel.org/doc/gorman/html/understand/understand025.html

https://events.static.linuxfound.org/sites/events/files/slides/slaballocators.pdf

Suport per gestió de memòria

- Interna al kernel <linux/slab.h> (cache friendly, actiu)
  - static inline void ***kzalloc**(size_t size, gfp_t flags)
  - static inline void ***kcalloc**(size_t n, size_t size, gfp_t flags)
  - Slab: quantitat per la qual una regió de memòria pot creixer o disminuir
    - Per objectes petits (<512 bytes), pàgines dedicades a una mida fixa (array)
      - Els objectes d'entre 16 i 32 bytes es situen a la mateixa pàgina
    - Per objectes més grans (512 bytes i més) es mante una llista
  - Sembla que la memòria sl?b és físicament contigua

Suport per gestió de memòria

- vmalloc - non-contiguous memory allocation
  - En pàgines físiques pot ser no contigua
  - En l'espai virtual és contigua (no hi ha més remei)
    - void * vmalloc(unsigned long size);
    - void * __vmalloc(unsigned long size,
      int gfp_mask, pgprot_t prot);
      - Tipus de memòria: GFP_USER, GFP_KERNEL, GFP_DMA...
      - Proteccions pel kernel
        - PAGE_KERNEL (lectura/escriptura)
        - PAGE_KERNEL_EXEC (+execució)
        - PAGE_KERNEL_RO (només lectura)
        - ...

https://www.kernel.org/doc/gorman/html/understand/understand010.html
https://www.kernel.org/doc/gorman/html/understand/understand024.html

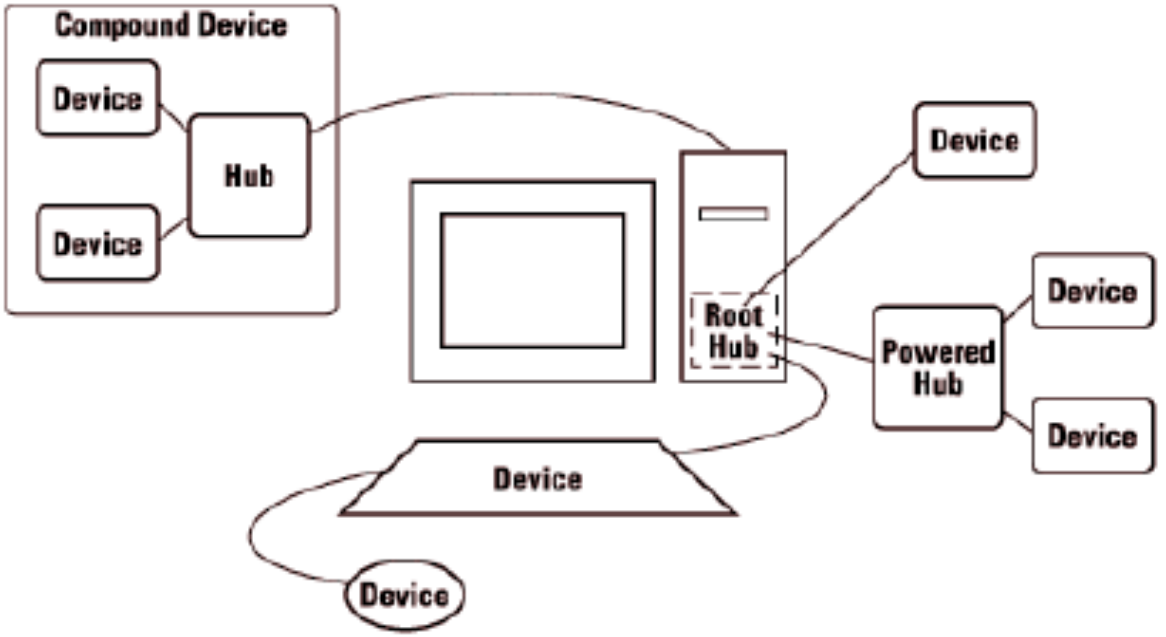
Índex

- Dispositius de caràcter
- Dispositius de block
- **Dispositius USB**
- Dispositius PCIe
- Dispositius de xarxa

http://www.coleparmer.com/TechLibraryArticle/944

Dispositius USB

- Connectats en un bus, cadascun atén a les peticions que se li dirigeixen
  - Identificats amb (vendor:product)



libusb

- Llibreria d'usuari multi-SO
  - Permet
    - Llistar
    - Accedir
    - Veure les característiques dels dispositius USB
      - Vendor / product, class Ids
      - MaxPacketSize
      - Endpoints
      - ...

http://libusb.info/

USB endpoints

- Buffer per a la transmissió de dades
  - Registre en el dispositiu, o
  - Regió mapejada en memòria
- Endpoint 0 sempre existeix i és el de control
  - Per demanar propietats del dispositiu i
  - Encarregar transmissió de dades
  - Bidireccional
- Poden haver-n'hi fins a 30
  - 1 – 15 i IN/OUT

Suport per USB

- Enregistrats amb una taula d'operacions particular
  - struct usb_serial_driver
  - struct usb_gadget_driver
  - struct usb_composite_driver
  - struct usb_driver

```
result = usb_register(&my_driver);
if (result)
        err("usb_register failed. Error number %d", result);
```

Operacions USB

- Suporten connexió, desconnexió, i suspend/resume

```
struct usb_driver {
        const char *name;

        int (*probe) (struct usb_interface *intf,
                      const struct usb_device_id *id);

        void (*disconnect) (struct usb_interface *intf);

        int (*unlocked_ioctl) (struct usb_interface *intf, unsigned int code,
                        void *buf);

        int (*suspend) (struct usb_interface *intf, pm_message_t message);
        int (*resume) (struct usb_interface *intf);
        int (*reset_resume)(struct usb_interface *intf);

        int (*pre_reset)(struct usb_interface *intf);
        int (*post_reset)(struct usb_interface *intf);
 ...
};
```

Operacions USB

- La funció USB de "probe" enregistra la interfície

```
static int my_probe(struct usb_interface *interface,
                    const struct usb_device_id *id)
{
     /* allocate memory for our device state and initialize it */
     dev = kzalloc(sizeof(*dev), GFP_KERNEL);
     ...
     /* we can register the device now, as it is ready */
     retval = usb_register_dev(interface, &my_usb_class);

     /* let the user know what node this device is now attached to */
     dev_info(&interface->dev,
              "USB device now attached to USB-%d",
              interface->minor);
     return 0;
}
                         static struct usb_class_driver my_usb_class = {
                                 .name =          "usbdev%d",
                                 .fops =            &my_usb_fops,
                                 .minor_base =   MY_USB_MINOR_BASE,
                         };
```

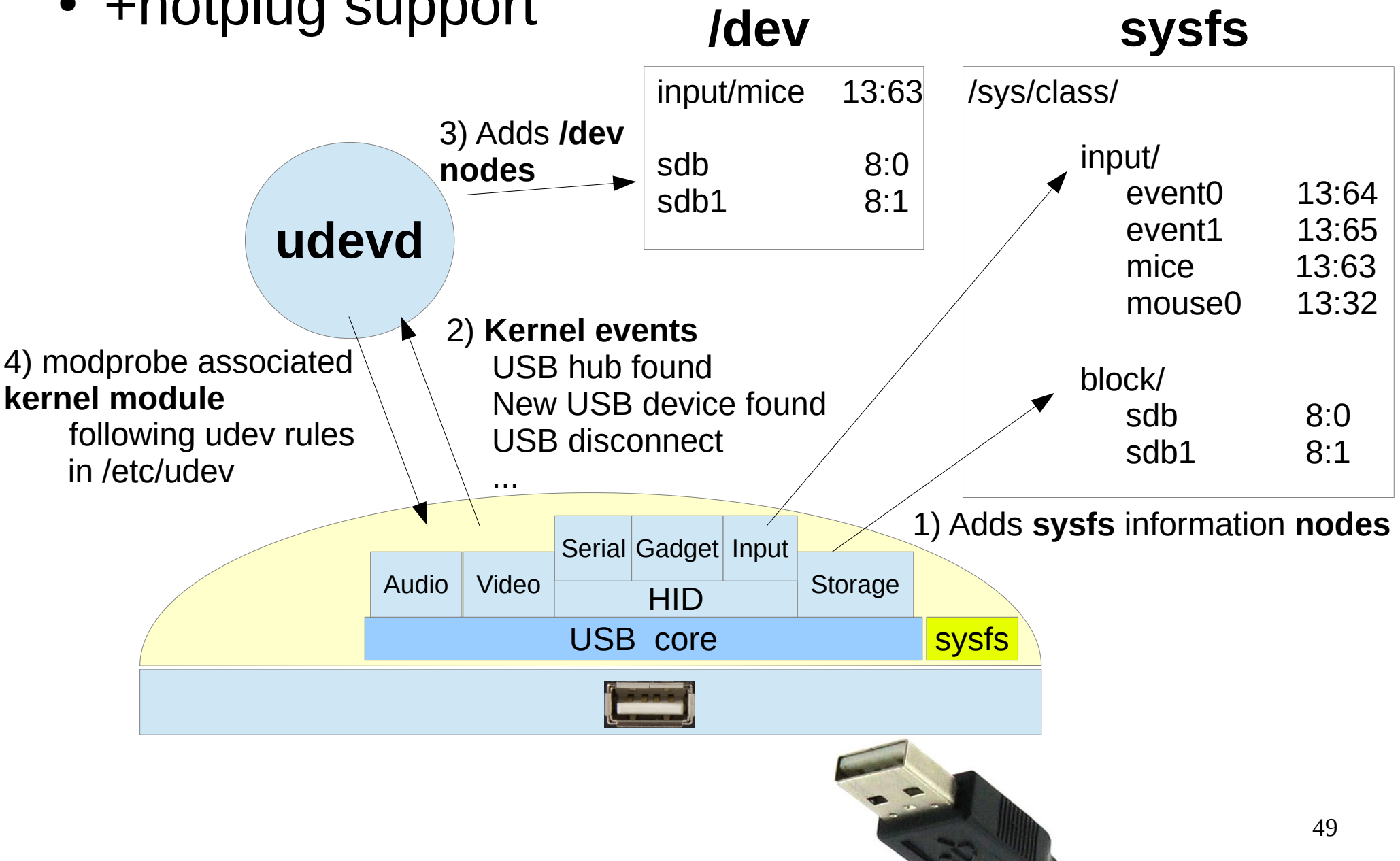
Operacions USB

- Les operacions del dispositiu són conegudes: file_operations

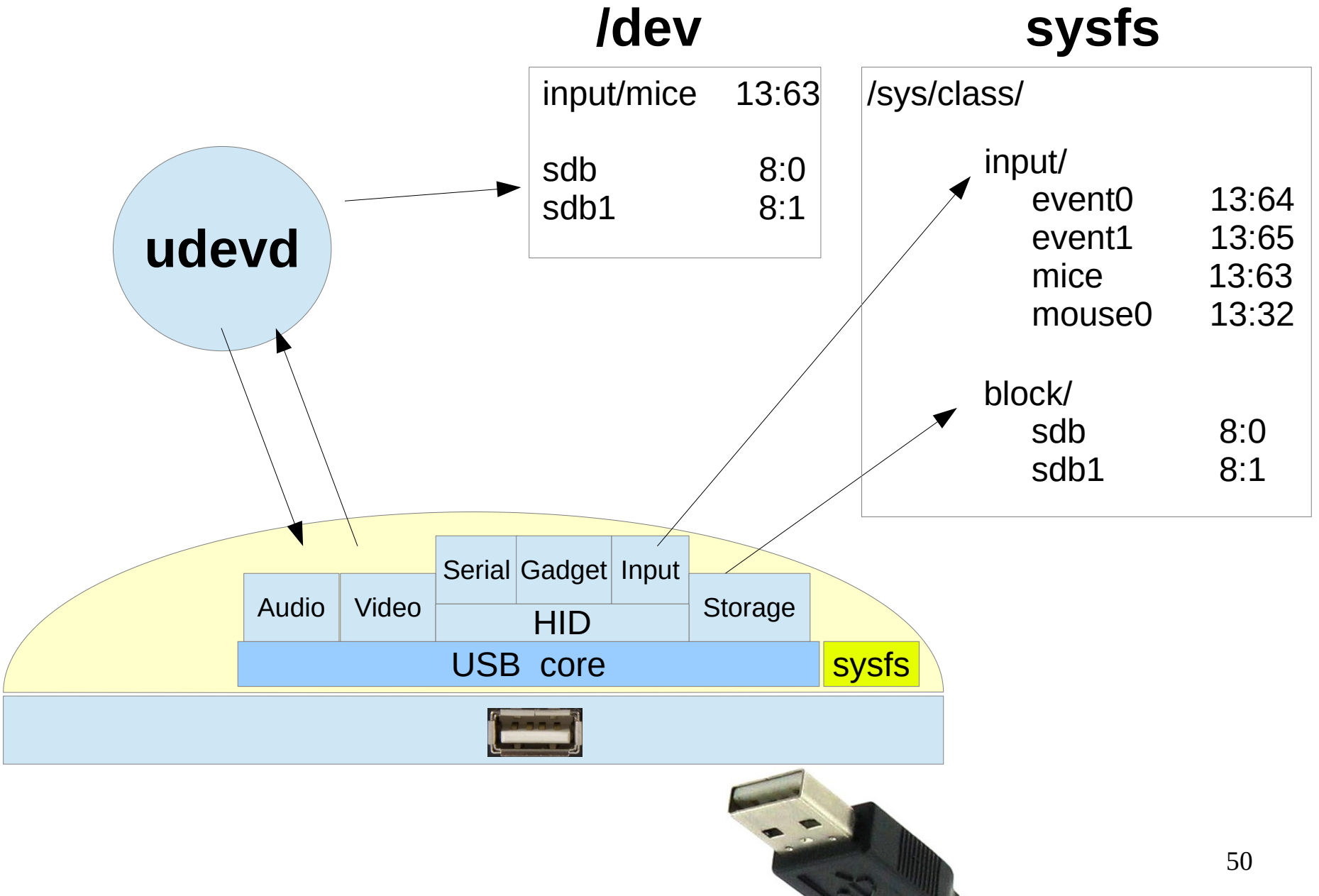
```
static const struct file_operations my_usb_fops = {
        .owner =        THIS_MODULE,
        .read =         my_read,
        .write =        my_write,
        .open =         my_open,
        .release =      my_close,
        .llseek =       noop_llseek,
};
```

Connexió USB - driver

- +hotplug support



Connexió USB - driver



udev / sysfs

- Procés d'usuari, més fàcil de depurar
- Crea fitxers dinàmicament a /dev
  - Administració automàtica de major/minor
- Permet un espai de noms consistent
  - Linux Standard Base - LSB
- Ofereix una interfície als usuaris, permetent veure la jerarquia de dispositius

http://en.wikipedia.org/wiki/Linux_Standard_Base

joint project by several Linux distributions under the organizational structure of the Linux Foundation to standardize the software system structure, including the filesystem hierarchy

Jerarquia del hardware

- Llista de busos (/sys/bus)

```
acpi          hid           memory  pci_express  scsi   usb
clocksource   i2c           mmc     platform     sdio   virtio
cpu           machinecheck  node    pnp          serio  xen
event_source  mdio_bus      pci     rapidio      spi    xen-backend
```

- Dispositius en cada bus
  - /sys/bus/memory/devices/ `memory0 memory1 memory2 memory3`
  - /sys/bus/pci/devices/

```
0000:00:00.0  0000:00:01.1
0000:00:01.3  0000:00:03.0
0000:00:01.0  0000:00:01.2
0000:00:02.0
```

- Mòduls associats

```
/sys/devices/pci0000:00/0000:00:01.1/ata1/ \
                                  host0/target0:0:0/0:0:0:0/modalias
scsi:t-0x00
/sys/devices/pci0000:00/0000:00:01.1/ata2/ \
                                  host1/target1:0:0/1:0:0:0/modalias
scsi:t-0x05
```

Jerarquia (I)

```
/sys/devices/pci0000:00/0000:00:02.0/drm/controlD64/dev
/sys/devices/pci0000:00/0000:00:02.0/drm/card0/dev
/sys/devices/pci0000:00/0000:00:1b.0/sound/card0/pcmC0D0p/dev
/sys/devices/pci0000:00/0000:00:1b.0/sound/card0/pcmC0D0c/dev
/sys/devices/pci0000:00/0000:00:1b.0/sound/card0/hwC0D0/dev
/sys/devices/pci0000:00/0000:00:1b.0/sound/card0/controlC0/dev
/sys/devices/pci0000:00/0000:00:1b.0/sound/card0/mixer/dev
/sys/devices/pci0000:00/0000:00:1b.0/sound/card0/dsp/dev
/sys/devices/pci0000:00/0000:00:1b.0/sound/card0/audio/dev
/sys/devices/pci0000:00/0000:00:1d.7/usb2/2-2/dev
/sys/devices/pci0000:00/0000:00:1d.7/usb2/2-2/2-2:1.0/host10/target10:0:0/ \
                                                    10:0:0:0/bsg/10:0:0:0/dev
/sys/devices/pci0000:00/0000:00:1d.7/usb2/2-2/2-2:1.0/host10/target10:0:0/ \
                                                    10:0:0:0/block/sdb/dev
/sys/devices/pci0000:00/0000:00:1d.7/usb2/2-2/2-2:1.0/host10/target10:0:0/ \
                                                    10:0:0:0/block/sdb/sdb1/dev
/sys/devices/pci0000:00/0000:00:1f.2/host0/target0:0:0/0:0:0:0/bsg/0:0:0:0/dev
/sys/devices/pci0000:00/0000:00:1f.2/host0/target0:0:0/0:0:0:0/block/sda/dev
/sys/devices/pci0000:00/0000:00:1f.2/host0/target0:0:0/0:0:0:0/block/sda/sda1/dev
/sys/devices/pci0000:00/0000:00:1f.2/host0/target0:0:0/0:0:0:0/block/sda/sda2/dev
/sys/devices/pci0000:00/0000:00:1f.2/host0/target0:0:0/0:0:0:0/block/sda/sda3/dev
/sys/devices/pci0000:00/0000:00:1f.2/host0/target0:0:0/0:0:0:0/block/sda/sda4/dev

/sys/devices/pci0000:00/0000:00:1f.2/host1/target1:0:0/1:0:0:0/block/sr0/dev
/sys/devices/pci0000:00/0000:00:1f.2/host1/target1:0:0/1:0:0:0/bsg/1:0:0:0/dev
/sys/devices/pnp0/00:07/rtc/rtc0/dev
```

Jerarquia (II)

```
/sys/devices/pci0000:00/0000:00:01.1/ata1/host0/target0:0:0/0:0:0:0/block/sda/dev
/sys/devices/pci0000:00/0000:00:01.1/ata1/host0/target0:0:0/0:0:0:0/block/sda/sda1/dev
/sys/devices/pci0000:00/0000:00:01.1/ata1/host0/target0:0:0/0:0:0:0/block/sda/sda2/dev
/sys/devices/pci0000:00/0000:00:01.1/ata2/host1/target1:0:0/1:0:0:0/bsg/1:0:0:0/dev
/sys/devices/pci0000:00/0000:00:01.1/ata2/host1/target1:0:0/1:0:0:0/block/sr0/dev
/sys/devices/pci0000:00/0000:00:01.2/usb1/1-1/dev
/sys/devices/pci0000:00/0000:00:01.2/usb1/1-1/1-1:1.0/0003:046D:C309.0001/ \
                                                              hidraw/hidraw0/dev
/sys/devices/pci0000:00/0000:00:01.2/usb1/1-1/1-1:1.0/input/input2/event2/dev
/sys/devices/pci0000:00/0000:00:01.2/usb1/1-1/1-1:1.1/0003:046D:C309.0002/ \
                                                              hidraw/hidraw1/dev
/sys/devices/pci0000:00/0000:00:01.2/usb1/1-1/1-1:1.1/input/input3/js0/dev
/sys/devices/pci0000:00/0000:00:01.2/usb1/1-1/1-1:1.1/input/input3/event3/dev
/sys/devices/pci0000:00/0000:00:01.2/usb1/1-1/1-1:1.1/input/input3/mouse0/dev
/sys/devices/pci0000:00/0000:00:01.2/usb1/1-2/dev
/sys/devices/pci0000:00/0000:00:01.2/usb1/1-2/1-2.2/dev
/sys/devices/pci0000:00/0000:00:01.2/usb1/1-2/1-2.2/1-2.2:1.0/input/input7/event4/dev
/sys/devices/pci0000:00/0000:00:01.2/usb1/1-2/1-2.2/1-2.2:1.0/0003:0D62:001C.0005/ \
                                                              hidraw/hidraw2/dev
/sys/devices/pci0000:00/0000:00:01.2/usb1/1-2/1-2.2/1-2.2:1.1/input/input8/event5/dev
/sys/devices/pci0000:00/0000:00:01.2/usb1/1-2/1-2.2/1-2.2:1.1/0003:0D62:001C.0006/ \
                                                              hidraw/hidraw3/dev
/sys/devices/pci0000:00/0000:00:01.2/usb1/1-2/1-2.2/1-2.2:1.1/usbmisc/hiddev0/dev
/sys/devices/pci0000:00/0000:00:02.0/drm/card0/dev
/sys/devices/pci0000:00/0000:00:02.0/drm/controlD64/dev
/sys/devices/pci0000:00/0000:00:02.0/graphics/fb0/dev
```

Events udev

USB

```
KERNEL[2684.967953] add      /devices/pci0000:00/0000:00:14.0/usb3/3-1 (usb)
KERNEL[2684.968755] add      /devices/pci0000:00/0000:00:14.0/usb3/3-1/3-1:1.0 (usb)
KERNEL[2684.969160] add      /devices/pci0000:00/0000:00:14.0/usb3/3-1/3-1:1.0/host13 (scsi)
KERNEL[2684.969189] add      /devices/pci0000:00/0000:00:14.0/ \
                                  usb3/3-1/3-1:1.0/host13/scsi_host/host13 (scsi_host)
UDEV  [2685.449086] add     /devices/pci0000:00/0000:00:14.0/usb3/3-1 (usb)
UDEV  [2685.450400] add     /devices/pci0000:00/0000:00:14.0/usb3/3-1/3-1:1.0 (usb)
UDEV  [2685.451150] add     /devices/pci0000:00/0000:00:14.0/usb3/3-1/3-1:1.0/host13 (scsi)
UDEV  [2685.451849] add     /devices/pci0000:00/0000:00:14.0/ \
                                  usb3/3-1/3-1:1.0/host13/scsi_host/host13 (scsi_host)
...
KERNEL[2685.973697] add      /devices/virtual/bdi/8:16 (bdi)
UDEV  [2685.974298] add     /devices/virtual/bdi/8:16 (bdi)
...
KERNEL[2687.987980] add      /devices/pci0000:00/0000:00:14.0/ \
                             usb3/3-1/3-1:1.0/host13/target13:0:0/13:0:0:0/block/sdb (block)
KERNEL[2687.988012] add      /devices/pci0000:00/0000:00:14.0/ \
                    usb3/3-1/3-1:1.0/host13/target13:0:0/13:0:0:0/block/sdb/sdb1 (block)
```

BDI, backing_dev_info

Particions

udevadm monitor

Events udev

Bateria

KERNEL[5248.875258] **change** /devices/LNXSYSTM:00/LNXSYBUS:00/PNP0A08:00/ \
device:08/PNP0C09:00/PNP0C0A:01/**power_supply/BAT1** (power_supply)
UDEV [5248.938060] **change** /devices/LNXSYSTM:00/LNXSYBUS:00/PNP0A08:00/ \
device:08/PNP0C09:00/PNP0C0A:01/**power_supply/BAT1** (power_supply)
KERNEL[5249.077989] **change** /devices/pci0000:00/0000:00:02.0/ \
drm/card0/card0-eDP-1/**intel_backlight** (backlight)
UDEV [5249.078408] **change** /devices/pci0000:00/0000:00:02.0/ \
drm/card0/card0-eDP-1/**intel_backlight** (backlight)

Il·luminació de la pantalla

KERNEL[6569.524200] **remove** /devices/pci0000:00/0000:00:14.0/usb3/3-1/3-1:1.0/ \
host13/target13:0:0/13:0:0:0/**block/sdb/sdb1** (block)
KERNEL[6569.524352] **remove** /devices/pci0000:00/0000:00:14.0/usb3/3-1/3-1:1.0/ \
host13/target13:0:0/13:0:0:0/**block/sdb** (block)
...

Desconnexió del disc

+ els events equivalents comunicats per l'Udev.

udevadm monitor

Exemple: usbmouse

```
static struct usb_driver usb_mouse_driver = {
        .name           = "usbmouse",
        .probe          = usb_mouse_probe,
        .disconnect     = usb_mouse_disconnect,
        .id_table       = usb_mouse_id_table,
};
```

```
static int usb_mouse_probe(struct usb_interface *intf,
                           const struct usb_device_id *id)
{
   ...
        mouse = kzalloc(sizeof(struct usb_mouse), GFP_KERNEL);
        input_dev = input_allocate_device();
        mouse->irq = usb_alloc_urb(0, GFP_KERNEL);
        input_set_drvdata(input_dev, mouse);
        input_dev->open = usb_mouse_open;
        input_dev->close = usb_mouse_close;
        usb_fill_int_urb(mouse->irq, dev, pipe, mouse->data,
                         (maxp > 8 ? 8 : maxp),
                         usb_mouse_irq, mouse, endpoint->bInterval);

        error = input_register_device(mouse->dev);
   ...
}
```

Exemple: usbmouse

```
static void usb_mouse_irq(struct urb *urb)
{
        struct usb_mouse *mouse = urb->context;
        signed char *data = mouse->data;

    ...
        input_report_key(dev, BTN_LEFT,   data[0] & 0x01);
        input_report_key(dev, BTN_RIGHT,  data[0] & 0x02);
        input_report_key(dev, BTN_MIDDLE, data[0] & 0x04);
        input_report_key(dev, BTN_SIDE,   data[0] & 0x08);
        input_report_key(dev, BTN_EXTRA,  data[0] & 0x10);

        input_report_rel(dev, REL_X,     data[1]);
        input_report_rel(dev, REL_Y,     data[2]);
        input_report_rel(dev, REL_WHEEL, data[3]);
        input_sync(dev);

        status = usb_submit_urb (urb, GFP_ATOMIC);

    ...
}
```

Exemple: codi genèric

- Observar la regió crítica

```
void input_event(struct input_dev *dev,
                 unsigned int type, unsigned int code, int value)
{
        unsigned long flags;

        if (is_event_supported(type, dev->evbit, EV_MAX)) {

                spin_lock_irqsave(&dev->event_lock, flags);
                input_handle_event(dev, type, code, value);
                spin_unlock_irqrestore(&dev->event_lock, flags);
        }
}
EXPORT_SYMBOL(input_event);
```

Exemple: codi genèric

- Dins la regió crítica

```
static void input_handle_event(struct input_dev *dev,
                               unsigned int type, unsigned int code, int value)
{
        int disposition;

        disposition = input_get_disposition(dev, type, code, value);

          // event going towards the device?
        if ((disposition & INPUT_PASS_TO_DEVICE) && dev->event)
                dev->event(dev, type, code, value);

        if (!dev->vals)
                return;

  ...
```

Exemple: codi genèric

- Passa la informació cap als gestors

```
...       // event going towards the user?
        if (disposition & INPUT_PASS_TO_HANDLERS) {
                struct input_value *v;

                if (disposition & INPUT_SLOT) {
                        v = &dev->vals[dev->num_vals++];
                        v->type = EV_ABS;
                        v->code = ABS_MT_SLOT;
                        v->value = dev->mt->slot;
                }

                v = &dev->vals[dev->num_vals++];
                v->type = type;
                v->code = code;
                v->value = value;
        }
...
```

Exemple: codi genèric

- Assegura que els valors arriben al dispositiu
  - flush
  - s'arriba a max_vals en el buffer

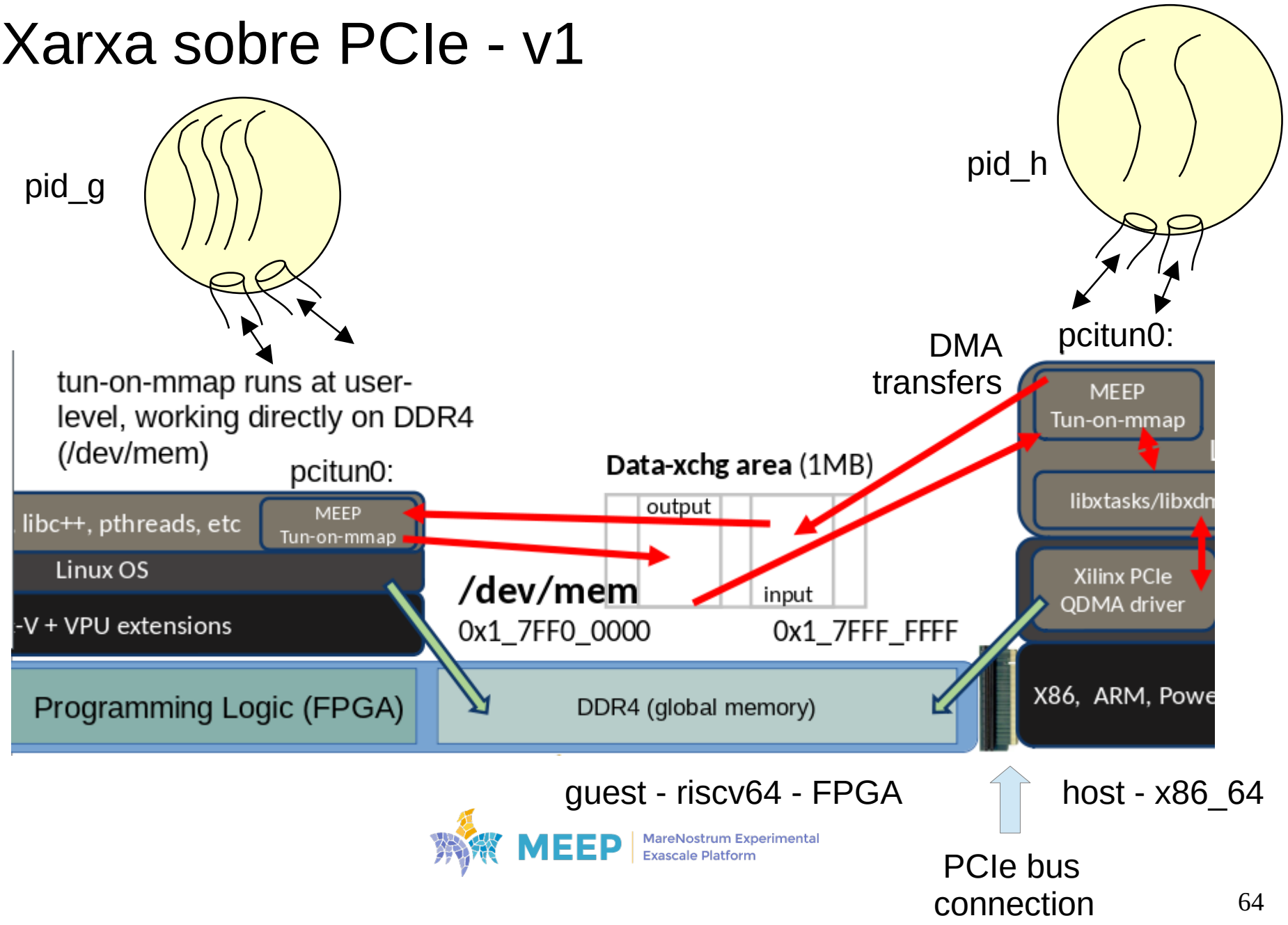
```
...
        if (disposition & INPUT_FLUSH) {
                if (dev->num_vals >= 2)
                        input_pass_values(dev, dev->vals, dev->num_vals);
                dev->num_vals = 0;
        } else if (dev->num_vals >= dev->max_vals - 2) {
                dev->vals[dev->num_vals++] = input_value_sync;
                input_pass_values(dev, dev->vals, dev->num_vals);
                dev->num_vals = 0;
        }
}
```

Índex

- Dispositius de caràcter
- Dispositius de block
- Dispositius USB
- **Dispositius PCIe**
- Dispositius de xarxa

Dispositius PCIe

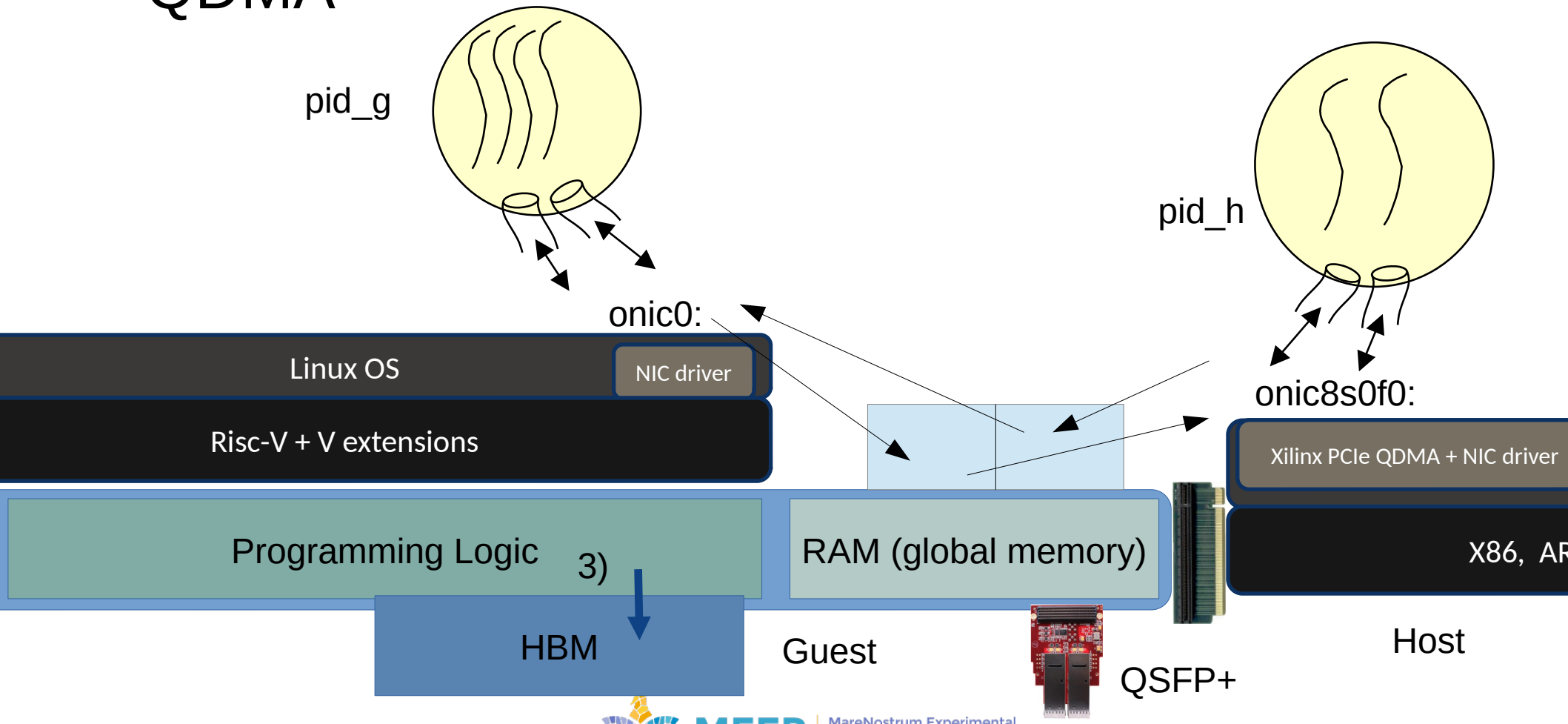
- Xarxa sobre PCIe - v1



MEEP MareNostrum Experimental Exascale Platform

Dispositius PCIe

- Xarxa sobre PCIe - v2: incorporar NIC al driver QDMA



MEEP MareNostrum Experimental Exascale Platform

Dispositius PCIe

- Reflexats a /sys physical/virtual functions & QDMA
  - /sys/bus/pci/devices/0000:08:00.[0-3]/qdma/
  - Transferències: 4.0 - 4.5 GB/s

```
$ sudo lspci -vd 10ee:
08:00.0 Memory controller: Xilinx Corporation Device 903f
        Subsystem: Xilinx Corporation Device 0007
        Flags: bus master, fast devsel, latency 0, IRQ 136
        Memory at b0200000 (64-bit, prefetchable) [size=256K]
        Memory at b0000000 (64-bit, prefetchable) [size=2M]
        Capabilities: [40] Power Management version 3
        Capabilities: [60] MSI-X: Enable+ Count=8 Masked-
        Capabilities: [70] Express Endpoint, MSI 00
        Capabilities: [100] Advanced Error Reporting
        Capabilities: [140] Single Root I/O Virtualization (SR-IOV)
        Capabilities: [180] Alternative Routing-ID Interpretation (ARI)
        Capabilities: [1c0] Secondary PCI Express
        Capabilities: [1f0] Virtual Channel
        Kernel driver in use: qdma-pf
        Kernel modules: qdma_pf
```

Índex

- Dispositius de caràcter
- Dispositius de block
- Dispositius USB
- Dispositius PCIe
- **Dispositius de xarxa**

Tunel sobre PCIe (v1)

- tun-on-mmap
  - Procés d'usuari
  - Intercanvia dades entre...
    - Xarxa - pcitun0:
    - Memòria - mmap(guest) - qdma(host)
  - Ha de llegir de xarxa i memòria
    - Com es bloqueja?

Tunel sobre PCIe (v1)

- Crea el dispositiu pcitun0: /dev/net/tun (master)
- Actualment,
  - No podem bloquejar les lectures de memòria
  - Fem polling, timeout al **select**
- Treball futur
  - Baixar la funcionalitat a dins del kernel - v2
  - Interrupcions host <=> guest

```
while (1) {
    fd_set readset;
    struct timeval tv = { 0, 20000 };

    FD_ZERO(&readset);
    FD_SET(tap_fd, &readset);
    res = select(tap_fd+1, &readset, NULL, NULL, &tv);
    if (res < 0) {
       perror ("select");
       close(tap_fd);
       return 1;
    }

    if (FD_ISSET(tap_fd, &readset)) {
          // read from pcitun0: to (output) memory
    }
    // poll on (input) memory
    // write packets to pcitun0:, if any
}
```

Drivers de xarxa

- Ethernet
  - Inicialització

```
ret = pci_register_driver(&e1000_driver);
```

  - Detecció, suspend/resume

```
static struct pci_driver e1000_driver = {
        .name     = e1000e_driver_name,
        .id_table = e1000_pci_tbl,
        .probe    = e1000_probe,
        .remove   = __devexit_p(e1000_remove),
#ifdef CONFIG_PM
        .driver.pm = &e1000_pm_ops,
#endif
        .shutdown = e1000_shutdown,
        .err_handler = &e1000_err_handler
};
```

Drivers de xarxa

- Operacions de xarxa

```
static const struct net_device_ops e1000e_netdev_ops = {
        .ndo_open               = e1000_open,
        .ndo_stop               = e1000_close,
        .ndo_start_xmit         = e1000_xmit_frame,
        .ndo_get_stats64        = e1000e_get_stats64,
        .ndo_set_rx_mode        = e1000_set_multi,
        .ndo_set_mac_address    = e1000_set_mac,
        .ndo_change_mtu         = e1000_change_mtu,
        .ndo_do_ioctl           = e1000_ioctl,
        .ndo_tx_timeout         = e1000_tx_timeout,
        .ndo_validate_addr      = eth_validate_addr,

        .ndo_vlan_rx_add_vid    = e1000_vlan_rx_add_vid,
        .ndo_vlan_rx_kill_vid   = e1000_vlan_rx_kill_vid,
#ifdef CONFIG_NET_POLL_CONTROLLER
        .ndo_poll_controller    = e1000_netpoll,
#endif
        .ndo_set_features = e1000_set_features,
};
```

Drivers de xarxa

- Gestió de les interrupcions

```
static int e1000_open(struct net_device *netdev)
{
 …
          /* allocate transmit descriptors */
        err = e1000e_setup_tx_resources(adapter);

        /* allocate receive descriptors */
        err = e1000e_setup_rx_resources(adapter);

        e1000_configure(adapter);

        err = e1000_request_irq(adapter, e1000_intr_msi);

        e1000_irq_enable(adapter);

}
```

Informació sobre la màquina

- Intel MultiProcessor Specification – 1993 - 1997
  - http://www.intel.com/design/archives/processors/pro/docs/242016.htm
  - http://www.uruk.org/mps/
- Advanced Configuration & Power Interface (ACPI) 1997 - 2013
  - Www.acpi.info
- Unified Extensible Firmware Interface
  - Www.uefi.org

Treball personal

- Examineu la informació que dóna el sistema
  - /proc/
    - modules
    - devices
    - partitions
    - interrupts
    - zoneinfo
    - bus/*
  - /sys
    - devices/system/cpu/*
    - class/block/*
    - class/usb/*
    - class/net/*
    - class/rtc/rtc0/*
    - bus/*
  - /dev
    - input/*
    - block/*
    - char/*
    - bus/*
- Quina informació que s'obté en llegir de cadascun dels dispositius de (od -x us pot anar bé):
  - /dev/input/ mouse0 mouse1 mice event0 event1 ...