Los archivos de periféricos

1. Introducción

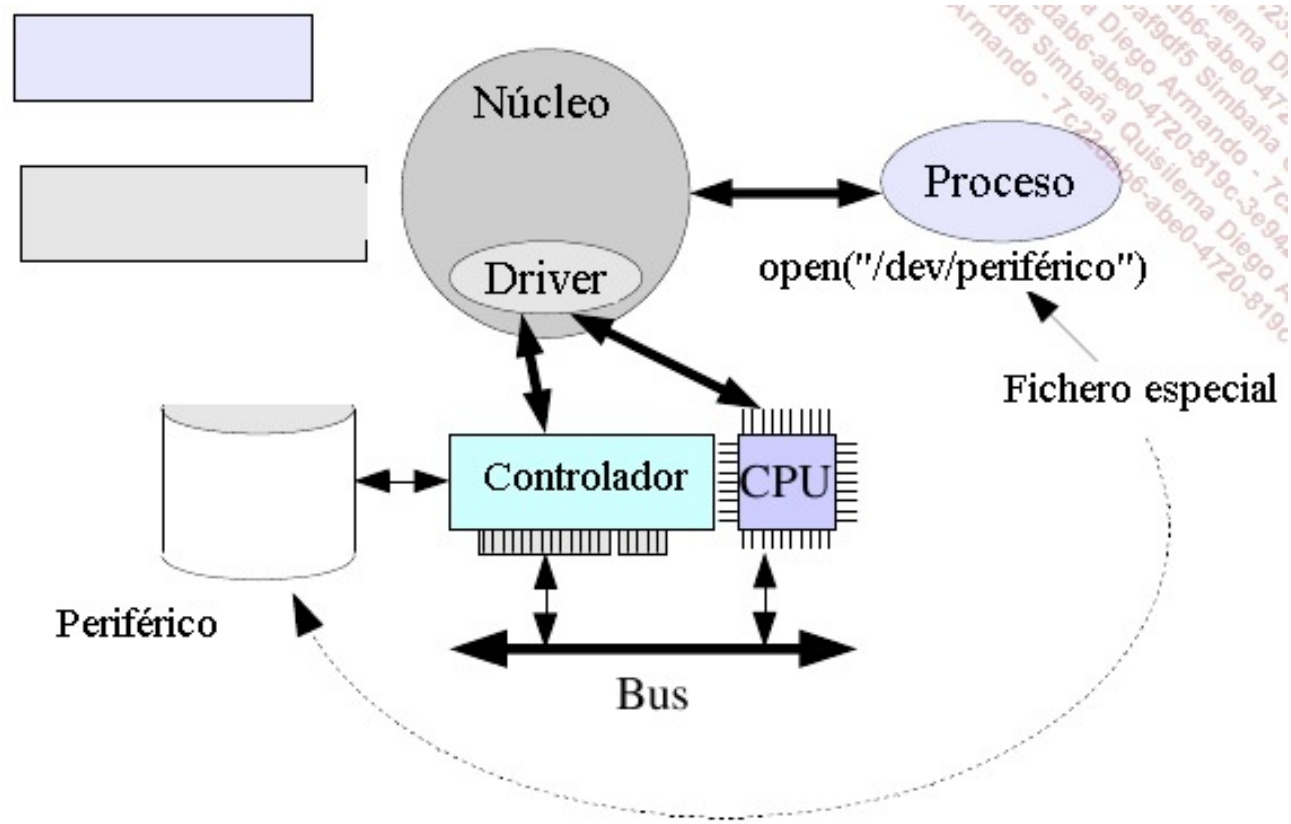
Regresamos al funcionamiento de los periféricos dentro de un ordenador. El principio suele ser el mismo en todos los ordenadores.

Los periféricos están vinculados a un controlador, por ejemplo IDE o SATA para los discos IDE, un controlador SCSI para los discos, lectores y otros escáneres SCSI, o incluso un controlador USB. Un controlador debe saber controlar varios periféricos vinculados a él.

El controlador se comunica con el microprocesador y la memoria con la ayuda del bus (bus de comandos y datos).

En cuanto a Linux, el controlador y sus periféricos se gestionan con la ayuda de drivers (un driver para el controlador, y uno o varios drivers para los periféricos relacionados con él, por ejemplo, un driver para el controlador SCSI, luego un driver para los discos, otro para los escáneres y otro para un CD-Rom). El driver suele ser un módulo complementario del núcleo, entregado por el fabricante o ya presente.

Los periféricos son vistos como archivos. En consecuencia, los procesos acceden a los periféricos mediante estos archivos con la ayuda de las funciones en lenguaje C cuyo código está en el núcleo, llamadas API (*Application Programming Interface*). El proceso debe abrir primero el archivo especial del periférico (primitive open), luego leer (read) o escribir (write) datos desde el periférico o hacia él, como lo haría un archivo normal. Luego el driver del periférico interpreta estas operaciones de lectura/escritura.



Linux accede a los periféricos mediante archivos especiales

En el archivo especial **`/dev/perifØrico`**, el sistema de gestión de archivos encuentra la información necesaria para dirigirse al controlador correspondiente a través del periférico abierto por un proceso.

2. Archivos especiales

Por convención, se colocan los archivos especiales periféricos en el directorio /dev; éstos disponen de un inodo único, como cualquier otro archivo. Por lo tanto, puede conocer sus atributos mediante el comando **ls -l**.

El primer carácter identifica el tipo de periférico:

- **c**: tipo de periférico en modo carácter;
- **b**: tipo de periférico en modo bloque.

Estos modos diferencian el tipo de intercambio de datos entre el módulo de gestión de archivos y el driver del periférico. En modo carácter, no se utilizan los buffers del sistema y el intercambio se hace byte por byte.

En modo bloque, el sistema accede al periférico mediante un índice que representa las coordenadas del bloque de datos en el soporte. Por lo tanto, es más rápido para periféricos del tipo discos duros.

Los otros dos atributos esenciales de un archivo periférico son el par de información que se encuentra en lugar del tamaño del archivo: el número **mayor** y el número **menor**.

- El número mayor identifica el driver y como consecuencia el controlador de periférico.
- El número menor suele identificar el periférico, pero también puede designar una particularidad del periférico, como la partición de un disco, una ubicación concreta, el número de tarjeta (en caso de que haya varias tarjetas de controladores idénticas, varias tarjetas de sonido, etc.).

Veamos algunos archivos especiales corrientes, según la distribución:

- **`/dev/mem`**: memoria física.
- **`/dev/kmem`**: memoria virtual del núcleo.
- **`/dev/console`**: consola maestra (/dev/syscon).
- **`/dev/tty`**: entrada/salida estándar del proceso en curso.
- **`/dev/tty*`**: los diferentes terminales.
- **`/dev/mouse`**: ratón, a menudo un atajo.
- **`/dev/swap`**: disco swap primario.
- **`/dev/null`**: basura UNIX. Se puede escribir en ella. La lectura provoca un EOF.
- **`/dev/zero`**: retorna cero siempre que se lee. Ideal para llenar archivos vacios para pruebas.
- **`/dev/root`**: sistema de archivos especial root.
- **`/dev/dump`**: disco donde el núcleo hace su dump en caso de “panic system”.
- **`/dev/rmt0`**: lector de banda magnética o de cartucho en modo carácter.

- **`/dev/fd0`**: lector de disquetes en modo bloque.
- **`/dev/pts/1`**: ídem, pero para Unix SYSTEM V (y Linux).
- **`/dev/lp0`**: impresora paralela.
- **`/dev/ttyS0`**: puerto COM1.
- **`/dev/ttyS1`**: puerto COM2.
- **`/dev/psaux`**: puerto PS2 para el ratón.
- **`/dev/sound`**: tarjeta de sonido.
- **`/dev/dsp`**: controlador DSP de la tarjeta de sonido.
- **`/dev/sequencer`** : secuenciador MIDI de la tarjeta de sonido.
- **`/dev/ide/*`**: periféricos IDE.
- **`/dev/scsi/*`**: periféricos SCSI.
- **`/dev/disk/*`**: los dispositivos de tipo disco, por diferentes métodos.
- **`/dev/block/*`** : los dispositivos de tipo bloque, en la forma de enlaces simbólicos.
- **`/dev/usb/*`**: periféricos USB.
- **`/dev/hdX`**: discos IDE excluyendo libata.
- **`/dev/sdX`**: discos SATA o SCSI.
- **`/dev/sg*`**: los dispositivos SCSI genéricos.
- etc.

3. Crear un archivo especial

El comando **mknod** permite crear un archivo especial. Aunque Linux disponga de métodos particulares que a menudo se encargan automáticamente de la creación de los archivos periféricos (udev, systemd), en ocasiones la documentación del periférico especifica cómo configurarlo manualmente. En este caso, ¡cuidado! El sistema de archivos **/dev** no es por lo general un simple directorio lleno de archivos especiales, sino un sistema de archivos virtual de tipo **udev** o **devpts**, ¡y totalmente dinámico! Habrá que encontrar otra solución.

```
mknod /dev/periférico tipo mayor menor
```

O bien se integran los drivers en el núcleo durante la compilación de éste, o se compilan estos drivers en forma de módulos complementarios cargados de manera dinámica. Según las distribuciones, el directorio /dev resulta ser a veces un sistema de archivos dinámico (devfs, udev) cuyo contenido cambia en función de la presencia o no de los periféricos. Así, el driver del periférico y un demonio particular **devfsd** o **udevd** se encargan de la creación del archivo periférico. Esto facilita, por ejemplo, la conexión en caliente, como la conexión de periféricos bajo demanda: el núcleo detecta el dispositivo, carga el driver correcto y este driver crea dinámicamente el archivo periférico.

A veces, sólo una parte del directorio /dev es dinámico, como el soporte de los terminales (devpts) o del USB con el sistema de archivos **usbdevfs**.

4. Conocer su hardware

a. Bus PCI

El comando **lspci** proporciona información detallada sobre las tarjetas y los adaptadores relacionados con el bus PCI. Los adaptadores pueden ser los conectados a los puertos de extensión de la placa base pero también los integrados a la placa base (controladores IDE/SATA, tarjetas de red, etc.). Los AGP son considerados bus PCI.

```
# lspci
00:00.0 Host bridge: ATI Technologies Inc RS480 Host Bridge (rev 10)
00:01.0 PCI bridge: ATI Technologies Inc RS480 PCI Bridge
00:11.0 IDE interface: ATI Technologies Inc 437A Serial ATA Controller
00:12.0 IDE interface: ATI Technologies Inc 4379 Serial ATA Controller
00:13.0 USB Controller: ATI Technologies Inc IXP SB400 USB Host Controller
00:13.1 USB Controller: ATI Technologies Inc IXP SB400 USB Host Controller
00:130,2 USB Controller: ATI Technologies Inc IXP SB400 USB2 Host Controller
00:14.0 SMBus: ATI Technologies Inc IXP SB400 SMBus Controller (rev 10)
00:14.1 IDE interface: ATI Technologies Inc Standard Dual Channel PCI IDE
Controller
```

```
00:14.3 ISA bridge: ATI Technologies Inc IXP SB400 PCI-ISA Bridge
00:14.4 PCI bridge: ATI Technologies Inc IXP SB400 PCI-PCI Bridge
00:14.5 Multimedia audio controller: ATI Technologies Inc IXP SB400
AC'97 Audio Controller (rev 01)
00:18.0 Host bridge: Advanced Micro Devices [AMD] K8 [Athlon64/Opteron]
HyperTransport Technology Configuration
00:18.1 Host bridge: Advanced Micro Devices [AMD] K8 [Athlon64/Opteron]
Address Map
00:18.2 Host bridge: Advanced Micro Devices [AMD] K8 [Athlon64/Opteron]
DRAM Controller
00:18.3 Host bridge: Advanced Micro Devices [AMD] K8 [Athlon64/Opteron]
Miscellaneous Control
01:05.0 VGA compatible controller: ATI Technologies Inc RS480 [Radeon
Xpress 200G Series]
02:03.0 Ethernet controller: Realtek Semiconductor Co., Ltd. RTL-8169
Gigabit Ethernet (rev 10)
```

Puede ir aún más lejos en los detalles con la opción **-v** y especificar un adaptador con sus identificadores. Para obtener la información detallada sobre el controlador Ethernet (02:03.0), haga síguelo siguiente:

```
# lspci -v -s 02:03.0
02:03.0 Ethernet controller: Realtek Semiconductor Co., Ltd. RTL-8169
Gigabit Ethernet (rev 10)
    Subsystem: Unknown device 1631:d008
    Flags: bus master, 66MHz, medium devsel, latency 64, IRQ 20
    I/O ports at 8800 [size=256]
    Memory at ff3ffc00 (32-bit, non-prefetchable) [size=256]
    Expansion ROM at ff300000 [disabled] [size=128K]
    Capabilities: [dc] Power Management version 2
```

Cuando es posible, la línea Subsystem indica el fabricante y el modelo exacto del periférico. Se identifica el periférico con un par de valores hexadecimales. El primero es el identificador único del fabricante (vendor). El segundo es el identificador del modelo. Algunos modelos tienen a veces los mismos identificadores, aunque el dispositivo difiera (cambio de chipset). La correspondencia es estática (se inscriben las asociaciones dentro del código compilado). Sirve, entre otras cosas, para cargar los módulos correctos. Si la información no es suficiente, especifique la opción **-vv**.

b. Bus USB

El comando **lsusb** hace lo mismo que lspci, pero para el bus USB:

```
# lsusb
Bus 021 Device 002: ID 05e3:0616 Genesys Logic, Inc. USB3.0 Hub
Bus 020 Device 012: ID 05e3:0723 Genesys Logic, Inc. USB Storage
Bus 020 Device 006: ID 05e3:0610 Genesys Logic, Inc. USB2.0 Hub
Bus 020 Device 009: ID 044f:b315 ThrustMaster, Inc. Thrustmaster dual analog 3.2
Bus 020 Device 008: ID 0a12:0001 Cambridge Silicon Radio Ltd. CSR8510 A10
Bus 020 Device 005: ID 05ac:1006 Apple Inc. Keyboard Hub Serial: 000000000000
Bus 020 Device 010: ID 05ac:0250 Apple Inc. Apple Keyboard
Bus 020 Device 004: ID 046d:0843 Logitech Inc. Logitech Webcam C930e Serial: 73EB166E
Bus 020 Device 003: ID 046d:c52f Logitech Inc. USB Receiver
Bus 026 Device 001: ID 8087:8008 Intel Corporation Hub
Bus 029 Device 001: ID 8087:8000 Intel Corporation Hub
Bus 000 Device 001: ID 1d6b:LPTH Linux Foundation USB 3.0 Bus
Bus 000 Device 001: ID 1d6b:IPCI Linux Foundation USB 2.0 Bus
Bus 000 Device 001: ID 1d6b:IPCI Linux Foundation USB 2.0 Bus
```

Cuando es posible, Linux indica cuáles son los nombres de los periféricos mediante una base de identificadores, de la misma manera que para las tarjetas PCI. Estos identificadores también sirven para determinar qué driver USB cargar.

Como en el caso de **lspci**, obtendrá más información con `-v` y con `-d`:

```
# lsusb -d 044f:b315
Bus 020 Device 009: ID 044f:b315 ThrustMaster, Inc. Firestorm Dual Analog 3
# lsusb -v -d 044f:b315
Bus 020 Device 009: ID 044f:b315 ThrustMaster, Inc. Firestorm Dual Analog 3
Device Descriptor:
  bLength                18
  bDescriptorType         1
  bcdUSB               1.10
  bDeviceClass            0
  bDeviceSubClass         0
  bDeviceProtocol         0
  bMaxPacketSize0         8
```

```
  idVendor           0x044f ThrustMaster, Inc.
  idProduct          0xb315 Firestorm Dual Analog 3
  bcdDevice            1.01
  iManufacturer           1
  iProduct                2
  iSerial                 0
  bNumConfigurations      1
...
```

c. Recursos físicos

El sistema de archivos virtual /proc está lleno de información sobre su hardware. Veamos una lista no exhaustiva.

Interrupciones

```
# cat /proc/interrupts
           CPU0       CPU1
  0:         30   IO-APIC   2-edge      timer
  1:       3098   IO-APIC   1-edge      i8042
  6:          3   IO-APIC   6-edge      floppy
  8:          1   IO-APIC   8-edge      rtc0
  9:          0   IO-APIC   9-fasteoi   acpi
 10:     112472   IO-APIC  10-fasteoi   virtio0
 11:         32   IO-APIC  11-fasteoi   uhci_hcd:usb1
 12:         15   IO-APIC  12-edge      i8042
 14:          0   IO-APIC  14-edge      ata_piix
 15:     223311   IO-APIC  15-edge      ata_piix
 24:          0   PCI-MSI 294912-edge   virtio2-config
 25:     509316   PCI-MSI 294913-edge   virtio2-input.0
 26:      41896   PCI-MSI 294914-edge   virtio2-output.0
 27:          0   PCI-MSI 81920-edge    virtio1-config
 28:          0   PCI-MSI 81921-edge    virtio1-control
 29:          0   PCI-MSI 81922-edge    virtio1-event
 30:      14982   PCI-MSI 81923-edge    virtio1-request
NMI:          0   Non-maskable interrupts
LOC:    3932265   Local timer interrupts
SPU:          0   Spurious interrupts
```

```
PMI:       0  Performance monitoring interrupts
IWI:       0  IRQ work interrupts
RTR:       0  APIC ICR read retries
RES:       0  Rescheduling interrupts
CAL:       0  Function call interrupts
TLB:       0  TLB shootdowns
TRM:       0  Thermal event interrupts
THR:       0  Threshold APIC interrupts
DFR:       0  Deferred Error APIC interrupts
MCE:       0  Machine check exceptions
MCP:     723  Machine check polls
HYP:       0  Hypervisor callback interrupts
HRE:       0  Hyper-V reenlightenment interrupts
HVS:       0  Hyper-V stimer0 interrupts
ERR:       0
MIS:       0
PIN:       0  Posted-interrupt notification event
NPI:       0  Nested posted-interrupt event
PIW:       0  Posted-interrupt wakeup event
```

Canales DMA

```
# cat /proc/dma
4: cascada
```

Intervalos de direcciones de entradas-salidas

```
# cat /proc/ioports
0000-001f: dma1
0020-0021: pic1
0040-0043: timer0
0050-0053: timer1
0060-006f: keyboard
0070-0077: rtc
0080-008f: dma page reg
00a0-00a1: pic2
00c0-00df: dma2
```

```
00f0-00ff: fpu
0290-0297: pnp 00:06
 0295-0296: w83627ehf
03c0-03df: vesafb
03f8-03ff: serial
0400-041f: 0000:00:1f.3
 0400-041f: i801_smbus
0480-04bf: 0000:00:1f.0
0800-087f: 0000:00:1f.0
 0800-0803: ACPI PM1a_EVT_BLK
 0804-0805: ACPI PM1a_CNT_BLK
 0808-080b: ACPI PM_TMR
 0810-0815: ACPI CPU throttle
 0820-082f: ACPI GPE0_BLK
0cf8-0cff: PCI conf1
9400-940f: 0000:00:1f.2
9480-948f: 0000:00:1f.2
 9480-948f: libata
...
```

Periféricos (bloque, carácter)

```
# cat /proc/devices
Character devices:
  1 mem
  4 /dev/vc/0
  4 tty
  4 ttyS
...
  5 /dev/tty
  5 /dev/console
  5 /dev/ptmx
  5 ttyprintk
  6 lp
  7 vcs
 10 misc
...
Block devices:
  7 loop
```

```
8 sd
 9 md
11 sr
65 sd

67 sd
...
```

Particiones

```
# cat /proc/partitions
major minor  #blocks name

   8     0   83886080 sda
   8     1       1024 sda1
   8     2   50329600 sda2
   8     3   31456256 sda3
   8     4    2095104 sda4
```

Procesadores

```
# cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 60
model name      : Intel(R) Core(TM) i5-4570 CPU @ 3.20GHz
stepping        : 3
cpu MHz         : 3192.616
cache size      : 6144 KB
physical id     : 0
siblings        : 2
core id         : 0
cpu cores       : 2
apicid          : 0
initial apicid  : 0
fpu             : yes
```

```
fpu_exception   : yes
cpuid level     : 13
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca
cmov pat pse36 clflush mmx fxsr sse sse2 ht syscall nx rdtscp lm constant_tsc
rep_good nopl xtopology nonstop_tsc eagerfpu pni pclmulqdq ssse3 cx16 sse4_1
sse4_2 movbe popcnt aes xsave avx rdrand hypervisor lahf_lm abm
bugs            : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf
mds swapgs itlb_multihit
bogomips        : 6385.23
clflush size    : 64
cache_alignment : 64
address sizes   : 39 bits physical, 48 bits virtual
power management:
...
```

Fíjese en la aparición de la línea **bugs**: esta nos indica que el procesador presenta fallos de tipo Spectre, Meltdown, etc. Se trata de la lista de los bugs conocidos en el momento de la publicación del núcleo y no presagia fallos futuros.

d. Otras herramientas

Las distintas distribuciones entregan juegos de herramientas complementarias, como **hwinfo** o **dmidecode** más estándar.

hwinfo

La herramienta **hwinfo** detecta su hardware y le da una lista (de manera corta con la opción `--short`). Este comando realiza un rastreo por todo su hardware, que devuelve la información:

```
# hwinfo --short
cpu:
                  Intel(R) Core(TM) i5-4570 CPU @ 3.20GHz, 3192 MHz
                  Intel(R) Core(TM) i5-4570 CPU @ 3.20GHz, 3192 MHz
keyboard:
 /dev/input/event2    AT Translated Set 2 keyboard
```

```
mouse:
  /dev/input/mice     Mouse
joystick:
  /dev/input/event7   ThrustMaster Firestorm Dual Analog 3
graphics card:
                VMware VMWARE0405
sound:
                Intel 82801AA AC'97 Audio Controller
storage:
                Intel 82371AB/EB/MB PIIX4 IDE
                Intel 82801HM/HEM (ICH8M/ICH8M-E) SATA Controller [AHCI mode]
network:
  enp0s3          Intel PRO/1000 MT Desktop Adapter
  enp0s8          Intel PRO/1000 MT Desktop Adapter
...
```

Para obtener más detalles, suprima la opción **`--short`** y especifique, si es preciso, qué componente quiere detallar. Por ejemplo, **`--cpu`** para el procesador, **`--memory`** para la memoria, etc. La lista está en el manual del comando.

```
# hwinfo --cpu
01: None 00.0: 10103 CPU
  [Created at cpu.460]
  Unique ID: rdCR.j8NaKXDZtZ6
  Hardware Class: cpu
  Arch: X86-64
  Vendor: "GenuineIntel"
  Model: 6.60.3 "Intel(R) Core(TM) i5-4570 CPU @ 3.20GHz"
  Features:
fpu,vme,de,pse,tsc,msr,pae,mce,cx8,apic,sep,mtrr,pge,mca,cmov,pat,pse36,
clflush,mmx,fxsr,sse,sse2,ht,syscall,nx,rdtscp,lm,constant_tsc,rep_good,nopl,
xtopology,nonstop_tsc, cpuid,tsc_known_freq,pni,pclmulqdq,ssse3,cx16,pcid,
sse4_1,sse4_2, x2apic,movbe, popcnt,aes,xsave,avx,rdrand,hypervisor,
lahf_lm,abm,invpcid_single,pti,fsgsbase,avx2,invpcid,md_clear,flush_l1d
  Clock: 3192 MHz
  BogoMips: 6385.21
  Cache: 6144 kb
```

```
  Units/Processor: 2
  Config Status: cfg=new, avail=yes, need=no, active=unknown
...

# hwinfo --memory
01: None 00.0: 10102 Main Memory
  [Created at memory.74]
  Unique ID: rdCR.CxwsZFjVASF
  Hardware Class: memory
  Model: "Main Memory"
  Memory Range: 0x00000000-0xf6f8dfff (rw)
  Memory Size: 3 GB + 768 MB
  Config Status: cfg=new, avail=yes, need=no, active=unknown
```

dmidecode

La herramienta **dmidecode** no interroga a los dispositivos directamente, sino que lee e interpreta la tabla **DMI** (*Desktop Management Interface*) del ordenador, a veces llamada también **SMBIOS** (*System Management BIOS*). Esta funciona también con UEFI. No sólo proporciona información sobre el estado físico actual de la máquina, sino también sobre sus extensiones posibles (por ejemplo, la velocidad máxima del procesador, la cantidad de memoria posible, etc.). A diferencia de **hwinfo**, que interroga a un componente, por ejemplo la CPU, **dmidecode** lee la información tal como la detecta la BIOS, UEFI y la placa base. Es rápido, a veces más concreto que **hwinfo**, pero en ocasiones da errores, por lo que se recomienda comprobarlo. A veces ocurre que dmidecode no tiene ningún retorno, o nos da poca información. Esto es debido al uso, o no, de la tabla DMI en algunos materiales, o algunos hipervisores.

Como la salida es mucho más larga que con **hwinfo**, especifique qué información desea con `-s` o `-t` (consulte el manual):

```
# dmidecode -t processor
# dmidecode 2.9
SMBIOS 2.4 present.

Handle 0x0004, DMI type 4, 35 bytes
Processor Information
```

Socket Designation: LGA775
Type: Central Processor
Family: Pentium 4
Manufacturer: Intel
ID: FB 06 00 00 FF FB EB BF
Signature: Type 0, Family 6, Model 15, Stepping 11
Flags:
FPU (Floating-point unit on-chip)
VME (Virtual mode extension)
DE (Debugging extension)
PSE (Page size extension)
TSC (Time stamp counter)
MSR (Model specific registers)
PAE (Physical address extension)
MCE (Machine check exception)
CX8 (CMPXCHG8 instruction supported)
APIC (On-chip APIC hardware supported)
SEP (Fast system call)
MTRR (Memory type range registers)
PGE (Page global enable)
MCA (Machine check architecture)
CMOV (Conditional move instruction supported)
PAT (Page attribute table)
PSE-36 (36-bit page size extension)
CLFSH (CLFLUSH instruction supported)
DS (Debug store) ACPI (ACPI supported)
MMX (MMX technology supported)
FXSR (Fast floating-point save and restore)
SSE (Streaming SIMD extensions)
SSE2 (Streaming SIMD extensions 2)
SS (Self-snoop)
HTT (Hyper-threading technology)
TM (Thermal monitor supported)
PBE (Pending break enabled)
Version: Intel(R) Core(TM)2 Duo CPU E6750 @ 2.66GHz
Voltage: 1.3 V
External Clock: 427 MHz
Max Speed: 3800 MHz
Current Speed: 3416 MHz
Status: Populated, Enabled

```
Upgrade: Socket LGA775
L1 Cache Handle: 0x0005
L2 Cache Handle: 0x0006
L3 Cache Handle: 0x0007
Serial Number: To Be Filled By O.E.M.
Asset Tag: To Be Filled By O.E.M.
Part Number: To Be Filled By O.E.M.
```

Observe que los valores en negrita no son los mismos que los devueltos por los otros comandos. Aquí no hay error: el procesador está «overclockeado» a 3,4 GHz...

```
# dmidecode -s processor-frequency
3416 MHz
```

5. El soporte del USB y del hotplug

a. Los módulos

El **USB** (*Universal Serial Bus*) es un bus de datos en modo serie y *plug and play*. El principio consiste en que, una vez conectado el adaptador USB, el sistema carga el posible driver correspondiente y el equipo funciona en seguida. En USB 2.0, las tasas de transferencia pueden alcanzar 480 Mbits/s. El USB 3 permite una tasa de transferencia de 4,8 Gbits/s.

Para las versiones USB 1.0 y USB 1.1, se reparten el mercado dos tipos de controladores: **UHCI** y **OHCI**.

- *Universal Controller Host Interface*: desarrollado por Intel.
- *Open Controller Host Interface*: los demás.

En USB 2.0, el controlador se llama **EHCI**: *Enhanced Host Controller Interface*.

Por último para USB 3.0 el controlador se llama **XHCI**: *eXtended Host Controller Interface*.

Linux gestiona los tres tipos de controladores.

El módulo básico se llama **usbcore**. Propone el conjunto de las API necesarias para otros módulos:

- **ohci_hcd**: soporte OHCI.
- **uhci_hcd**: soporte UHCI.
- **ehci_hcd**: soporte EHCI.
- **xhci_hcd** : soporte XHCI.
- **usb_storage**: capa de acceso a los soportes masivos: discos externos, pendrives, etc.
- **usbhid**: capa de acceso al soporte de los periféricos **HID** (*Human Interface Device*) de tipo teclados, ratones, joystick, etc.
- **snd-usb-audio**: soporte de tarjetas de sonido USB.
- **usbvideo**: soporte de tarjetas de vídeo y adquisición de USB.
- **irda-usb**: soporte de los puertos de infrarrojo USB.
- **usbnet**: soporte de las tarjetas de red USB.
- etc.

Si estos módulos parecen ausentes, pueden que estén integrados directamente en el núcleo. En este caso, una búsqueda en el archivo de configuración del núcleo o con dmesg le dará la información.

```
grep -i xhci_hcd /boot/config-5.3.0-29-generic
CONFIG_USB_XHCI_HCD=y
```

b. Carga

Se cargan los módulos USB (y otros relacionados con el hardware) de distintas formas:

- Con **ramdisk initial**: actualmente la mayoría de los teclados y ratones son de tipo USB. Por eso, se suele cargar el soporte básico del USB y el módulo usbhid mediante el initrd (Initial Ramdisk). De ahí que a menudo haya un pequeño desfase entre el momento en el cual GRUB carga el núcleo y el momento en el que se puede utilizar el teclado, entre el final del soporte del USB por la BIOS (paso del núcleo al modo protegido) y el momento en que el núcleo se encarga

del soporte USB.

- Con **init**: un servicio encargado de la detección del hardware y de la carga de los drivers carga la lista de los módulos correspondiente al hardware.
- Con **kmod**: carga automática de los módulos del núcleo. Cuando el núcleo detecta la presencia de un nuevo periférico USB, puede cargar el módulo correspondiente. Recuerde que un módulo proporciona los identificadores del hardware del cual se encarga. El archivo **modules.usbmap** provee esta correspondencia. Entonces el núcleo ejecuta **modprobe**.
- Con **udev** o **hotplug**: unas reglas permiten especificar acciones cuando llegan nuevos periféricos, entre las cuales se encuentra, por ejemplo, la carga de módulos complementarios.
- Con **systemd**: systemd y udev están vinculados completamente. Un mensaje del bus udev puede desencadenar una acción de systemd para la creación de un dispositivo y la carga de un módulo.
- Manualmente.

c. hotplug, usbmgr

Antes de la llegada de udev y HAL, se utilizaba principalmente el proyecto **Linux Hotplug Project**. hotplug no gestionaba únicamente el USB, sino cualquier hardware conectado, en caliente o no (coldplug), al PC.

Ya no se utiliza hotplug en ninguna de las distribuciones recientes basadas en el núcleo 2.6. Ha dejado su sitio al tándem udev/HAL.

usbmgr era un sistema de hotplug previsto para la gestión del USB. Ya no se utiliza.

d. udev

udev es un sistema de archivos dinámico que sustituye al antiguo **devfs** de los núcleos 2.4. **udev** gestiona todo el árbol /dev. Va a crear y modificar los archivos periféricos presentes en este directorio.

A partir de 2013 udev es un componente de systemd.

A diferencia de hotplug, que se ejecutaba en modo kernel, se inicia **udev** como un servicio

clásico, asociado a un sistema de archivos particular. Se sitúa en el espacio del usuario.

```
# mount |grep udev
udev on /dev type devtmpfs
(rw,nosuid,relatime,size=1817832k,nr_inodes=454458,mode=755)
```

El núcleo gestiona eventos y mensajes. **udev** lee los mensajes emitidos por el núcleo y los interpreta. Dispone de reglas que aplica según el tipo de mensaje. Por ejemplo, veamos tres reglas sencillas:

```
KERNEL=="raw1394*",          GROUP="video"
KERNEL=="dv1394*",           SYMLINK+="dv1394/%n", GROUP="video"
KERNEL=="video1394*",        SYMLINK+="video1394/%n", GROUP="video"
```

- **KERNEL**: nombre del evento del núcleo. Aquí aparece un directorio Firewire.
- **GROUP**: el archivo periférico pertenecerá al grupo indicado.
- **SYMLINK**: udev va a crear un vínculo simbólico del archivo periférico hacia la ubicación indicada. Aquí el %n representa el número en el orden de detección.

Es sólo un ejemplo. La formación LPI no requiere escribir reglas elaboradas, sino entender el mecanismo asociado. Con las reglas anteriores, si conecta una videocámara digital gracias al enchufe DV a su tarjeta Firewire, el núcleo va a generar un evento cuyo nombre empieza por "vídeo1394". Se ejecutará la regla correspondiente:

- Aparición de /dev/video1394.
- Modificación de su grupo en vídeo.
- Creación de un vínculo simbólico entre /dev/video1394 y /dev/video1394/0 (el primero).

Se colocan las reglas en `/etc/udev/rules.d` o en `/lib/udev/rules.d`.

```
# ls -l
total 324
-rw-r--r-- 1 root root    69 nov.  18 2014  40-crda.rules
```

```
-rw-r--r-- 1 root root 35692 ago   8 15:51 40-usb_modeswitch.rules
-rw-r--r-- 1 root root   613 nov.  24 13:21 40-vm-hotadd.rules
-rw-r--r-- 1 root root   210 nov.  24 13:21 50-firmware.rules
-rw-r--r-- 1 root root  3310 nov.  24 13:21 50-udev-default.rules
-rw-r--r-- 1 root root  6919 jul. 11  2016 55-dm.rules
-rw-r--r-- 1 root root   612 nov.  24 13:21 60-block.rules
-rw-r--r-- 1 root root   910 nov.  24 13:21 60-cdrom_id.rules
-rw-r--r-- 1 root root   153 nov.  24 13:21 60-drm.rules
-rw-r--r-- 1 root root   738 nov.  24 13:21 60-evdev.rules
-rw-r--r-- 1 root root   616 nov.  24 13:21 60-persistent-alsa.rules
-rw-r--r-- 1 root root  2626 nov.  24 13:21 60-persistent-input.rules
-rw-r--r-- 1 root root  1495 jul. 11  2016 60-persistent-storage-dm.rules
-rw-r--r-- 1 root root  5827 nov.  24 13:21 60-persistent-storage.rules
-rw-r--r-- 1 root root  1509 nov.  24 13:21 60-persistent-storage-tape.rules
-rw-r--r-- 1 root root   769 nov.  24 13:21 60-persistent-v4l.rules
```

Observe el orden numérico que proporciona el orden de ejecución de las reglas. Puede modificar las reglas y crear las suyas propias. En este caso, cree un archivo llamado **99-local.rules** o **99.zz.rules** (por ejemplo) y coloque dentro sus reglas.

A modo de ejemplo, las diferentes rutas posibles de acceso a los discos y particiones, presentadas en el capítulo Los discos y el sistema de archivos - Acceder a los sistemas de archivos, proceden de reglas udev. La regla siguiente (que depende de otras reglas en amont, lo que se puede deducir por el test de una variable **udev** DETYPE ya ubicada) importa el resultado de un comando **vol_id** que ya conoce para crear vínculos con UUID y con label.

```
# by-label/by-uuid (filesystem properties)
ENV{DEVTYPE}=="partition", IMPORT{program}="vol_id --export $tempnode"
```

Puede conocer qué periféricos están controlados por **udev** con **udevinfo**, así como el caso de un periférico dado, de la manera siguiente:

```
# udevinfo --query=all -n /dev/sda
P: /block/sda
N: sda
```

```
S: disk/by-id/scsi-SATA_ST380011A_5JVTH798
S: disk/by-id/ata-ST380011A_5JVTH798
S: disk/by-path/pci-0000:00:14.1-scsi-0:0:0:0
S: disk/by-id/edd-int13_dev80
E: DEVTYPE=disk
E: ID_VENDOR=ATA
E: ID_MODEL=ST380011A
E: ID_REVISION=8.01
E: ID_SERIAL=SATA_ST380011A_5JVTH798
E: ID_SERIAL_SHORT=5JVTH798
E: ID_TYPE=disk
E: ID_BUS=scsi
E: ID_ATA_COMPAT=ST380011A_5JVTH798
E: ID_PATH=pci-0000:00:14.1-scsi-0:0:0:0
E: ID_EDD=int13_dev80
```

El comando **udevinfo** ha sido sustituido por **udevadm info** en las últimas versiones de udev:

```
# udevadm info --query=all -n /dev/sdb
P: /devices/pci0000:00/0000:00:1f.2/host1/target1:0:0/1:0:0:0/block/sdb
N: sdb
S: disk/by-id/ata-WDC_WD5000AACS-00G8B1_WD-WCAUK0742110
S: disk/by-id/scsi-SATA_WDC_WD5000AACS-_WD-WCAUK0742110
S: disk/by-path/pci-0000:00:1f.2-scsi-1:0:0:0
S: disk/by-id/wwn-0x50014ee257f93cf8
E: UDEV_LOG=3
E: DEVPATH=/devices/pci0000:00/0000:00:1f.2/host1/target1:0:0
/1:0:0:0/block/sdb
E: MAJOR=8
E: MINOR=16
E: DEVNAME=/dev/sdb
E: DEVTYPE=disk
E: SUBSYSTEM=block
E: ID_ATA=1
E: ID_TYPE=disk
E: ID_BUS=ata
E: ID_MODEL=WDC_WD5000AACS-00G8B1
```