**Informática Electrónica**

# *Manejadores de Dispositivos (Device Drivers)*

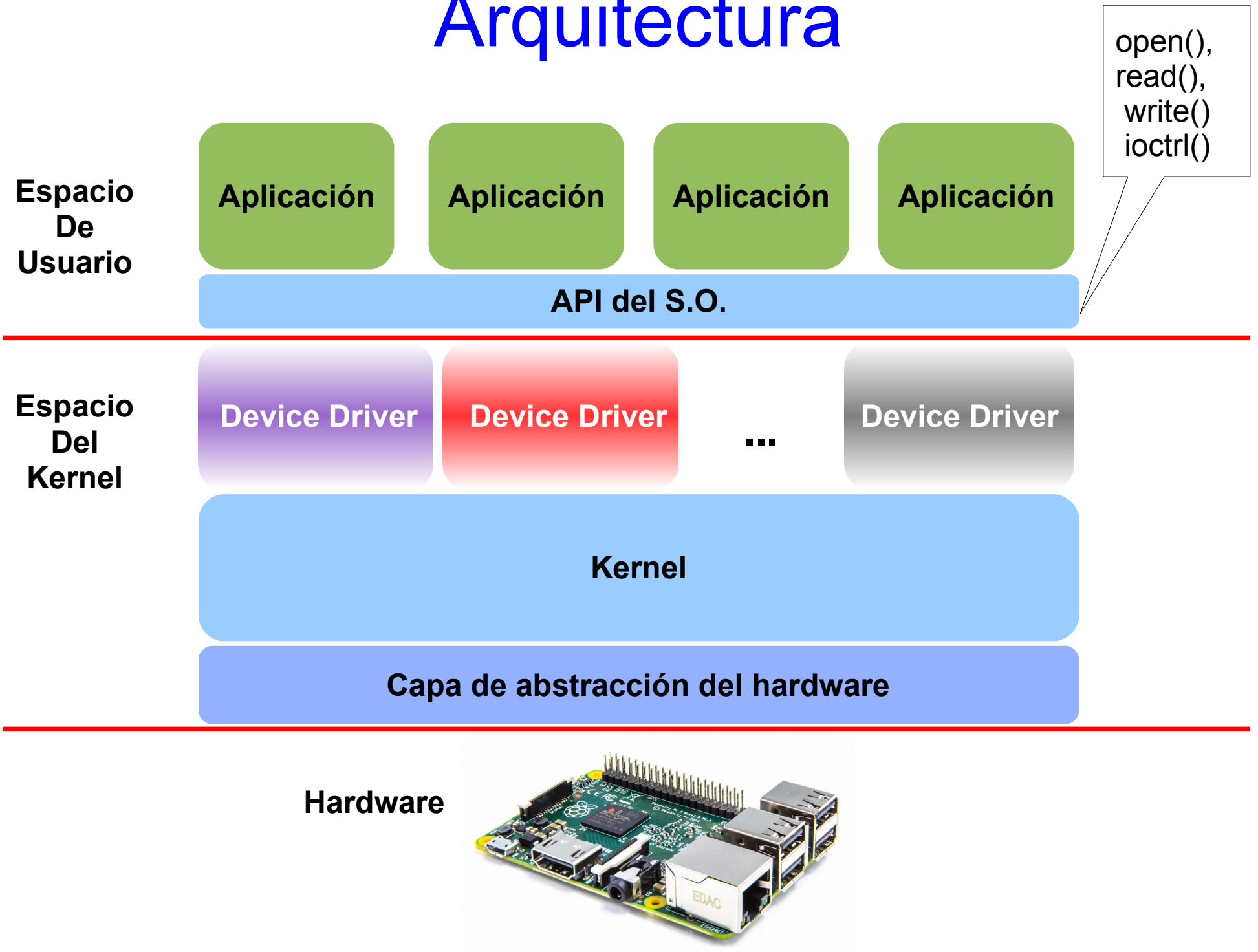
DSI-EIE-FCEIA

*2015*

# ¿Que es un DD?

- Es una pieza de software que interactúa con (entre) el sistema operativo y con uno o mas dispositivos físicos
- Mientras que las aplicaciones tradicionales ejecutan una o varias tareas desde su arranque hasta el fin de ejecución, un DD se "*inicializa*" y queda en cargado en memoria a la espera de "*peticiones de servicio*"
- En algunos casos un DD puede ser "*cargado*" y "*descargado*" de memoria dinámicamente, es decir, cuando se lo necesita. Ej.: cuando conectamos una cámara de fotos a la PC

# Arquitectura



# Niveles de Privilegio (i)

- Todos los procesadores modernos permiten distintos “niveles de privilegio” para la ejecución de software
- Cada nivel permite ejecutar un subconjunto específico del set de instrucciones del micro
- En procesadores Intel x86 el nivel 0 permite todas las instrucciones del set, mientras que los siguientes (1, 2...) restringen operaciones como `out`
- El núcleo del sistema operativo (kernel) corre al nivel 0, es decir, no tiene restricciones de ejecución. Se dice que corre en “modo privilegiado (kernel)”

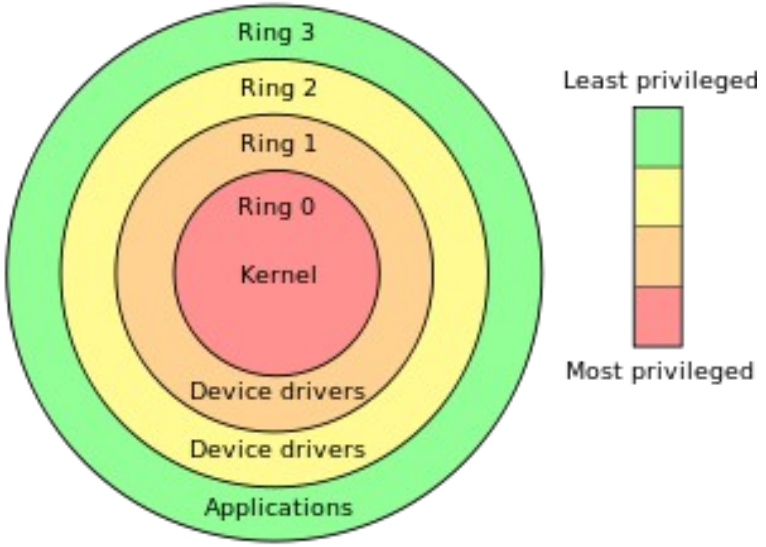
# Niveles de Privilegio (ii)

- La mayoría de las aplicaciones que usamos en un sistema operativo, como por ejemplo Linux corre en un modo no privilegiado llamado "***modo usuario*** *(user)*"
- En el "modo usuario" una aplicación no tiene acceso directo a la memoria física, por ejemplo
- Esta política permite arbitrar la **concurrencia** de aplicaciones

Fuentes: Intel® 64 and IA-32 Architectures Software Developer's Manual
The ARM Architecture With a focus on v7A and Cortex-A8

# Niveles de Privilegio (iii)

- Los niveles de privilegio se representan como anillos concéntricos
- Hacia el centro ⇒ mayor privilegio



# Espacios del kernel y del usuario

- Un DD se ejecuta en el nivel de privilegio del kernel y una aplicación lo hace en el nivel de privilegio del usuario
- Esta separación es garantizada por el kernel, y permite proteger a cada aplicación de las demás ejecutándose concurrentemente, tanto el área de memoria asignada como el estado del procesador (contexto)
- Todas las operaciones críticas son ejecutadas en el nivel de privilegio del kernel del S.O.

# Clases de Device Drivers

- Caracter: la transferencia de datos se lleva a cabo byte a byte, como por ejemplo en una UART
- Bloque: la transferencia de datos se lleva a cabo en bloques de bytes de longitud fija, como por ejemplo en un disco
- Red: la transferencia tiene lugar en tramas o paquetes de bytes. Ejemplo: un adaptador Ethernet

# Política y Mecanismo

- Un DD provee un conjunto de características que las aplicaciones pueden usar, llamadas *“mecanismo”*
- La forma en la que cada aplicación decide usar estas características es privativa de la aplicación y se denomina *“política”*

# Ciclo de Vida de un DD

- Un DD, como toda pieza de software, tiene una fase de diseño, una de construcción y una de utilización
- Las dos primeras son similares a cualquier otra pieza de software
- La fase de utilización es específica:
  - Integrado al núcleo del sistema operativo
  - Cargado y descargado a requerimiento del usuario

# Fase de Utilización

- El DD del disco rígido o el de la placa de red están integrados al núcleo: son necesarios para el arranque del S.O.
- Los DD de dispositivos "*plug-an-play*" se cargan cuando el dispositivo se conecta físicamente a la computadora, por ejemplo, una cámara o un celular

# Desarrollo de DD

- Por la complejidad inherente al desarrollo de DD, los sistemas operativos modernos brindan entornos de programación que proveen funcionalidades de base y permiten al diseñador de DD centrarse en el manejo del dispositivo
- Consisten en especificaciones, librerías y herramientas a disposición del diseñador

# Utilización de un DD

- Según la plataforma, existen servicios del S.O que permiten acceder en forma controlada a los dispositivos.
- El modelo de acceso sigue la lógica de archivos: el dispositivo es mostrado por el S.O. como un archivo especial, sobre el cual pueden ejecutarse lecturas, escrituras y funciones de control.
- El programa que quiere acceder al DD debe obtener un “manejador” *(handler)*.

# Device Handlers

- Un "*device handler*" es un identificador que se obtiene mediante una llamada a una función (generalmente `open()`) y sirve para las subsecuentes operaciones sobre el dispositivo, tales como leer y/o escribir datos y controlar su funcionamiento.

# API Estándar Posix

```
int open(char *path, int oflag, ...);

int read(int handler, void *buf, int nbyte);

int write(int handler, void *buf, int nbyte);

int ioctl(int handler, long cmd, ...);

int close(int handler);
```

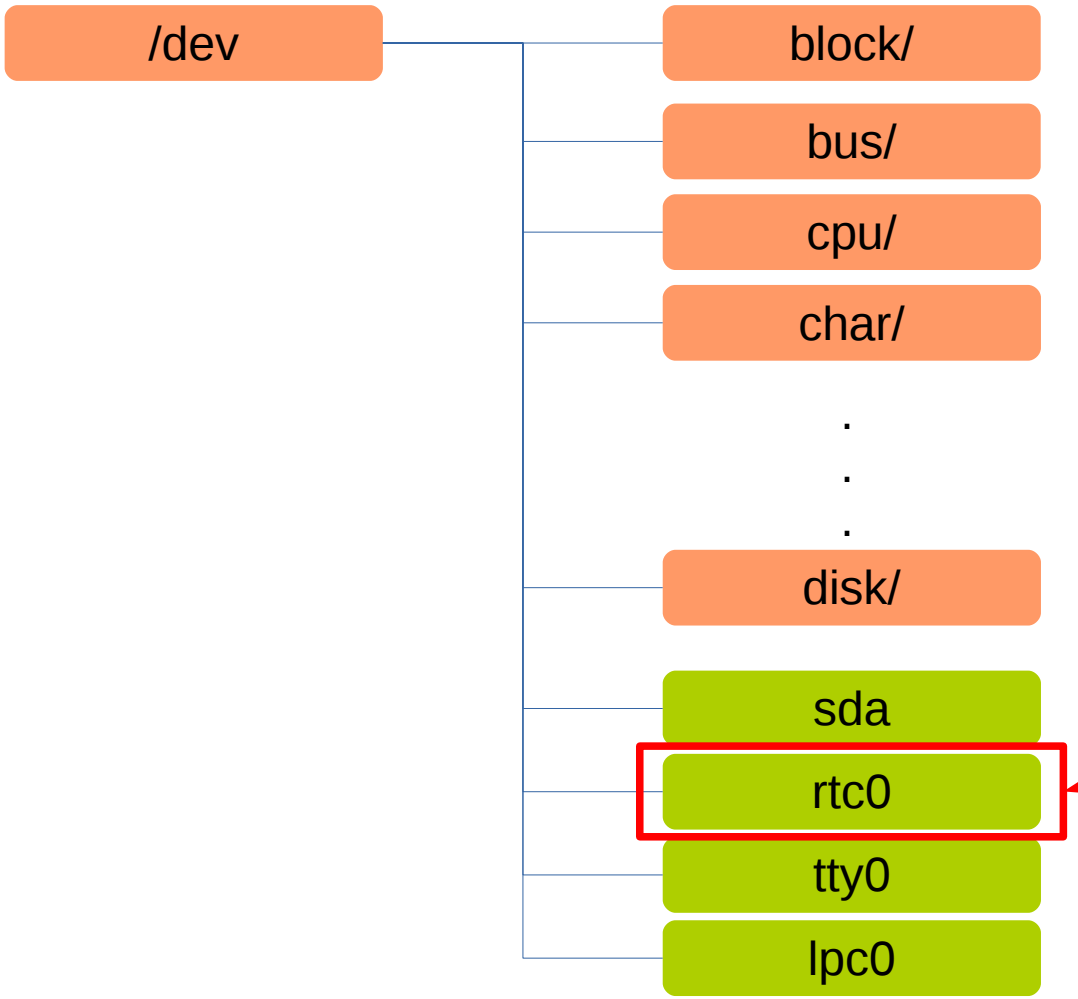
# Árbol de Dispositivos

- Es una descripción del hardware del sistema, que contiene:
  - El nombre de la(s) CPU(s)
  - La configuración de memoria
  - La enumeración de todos los dispositivos de entrada/salida
- Se representa como una jerarquía en forma de árbol, de allí su nombre (*Device Tree*)

# Árbol de Dispositivos

- Es una estructura de datos independiente del sistema operativo
- En el caso de estudio utilizaremos Linux: el DT está representado por una jerarquía de carpetas basada en **`/dev`**
- El DT es análogo a la organización de un disco con carpetas, subcarpetas y archivos

# Árbol de Dispositivos Linux




Para conectar una aplicación al reloj de tiempo real, por ejemplo, el primer argumento de `open()` es **`"/dev/rtc0"`**

# Caso de Estudio: RTC Linux

- Todas las computadoras tienen un “Reloj de Tiempo Real”, encargado de mantener la referencia temporal externa en forma permanente, aún cuando esté apagada
- Es frecuentemente utilizado el chip Motorola MC146818 (o derivados)
- En Linux este dispositivo es representado mediante el archivo especial “**`/dev/rtc`**”

# Lectura del RTC en Linux*

```
/* Leer los segundos de RTC */
unsigned char segundos;
int fd;
fd = open("/dev/rtc0", O_RDONLY);
ioctl(fd, 0, &segundos);
close(fd);

/* Leer fecha y hora del RTC */
char fecha_hora[256];
int fd;
fd = open("/dev/rtc0", O_RDONLY);
read(fd, fecha_hora, 255);
close(fd);
```

* Ubuntu

# Práctica 5: Uso de un DD

# Bibliografía


- Linux device drivers third edition, Jonathan Corbet, Alessandro Rubini, and Greg Kroah-Hartman, O'reilly, 2005
- Device Tree Org, http://devicetree.org/Main_Page
- Sistemas Operativos, 7ª edición, Galvin, Silverschatz y Gagne