Fundamentos de los Sistemas Operativos

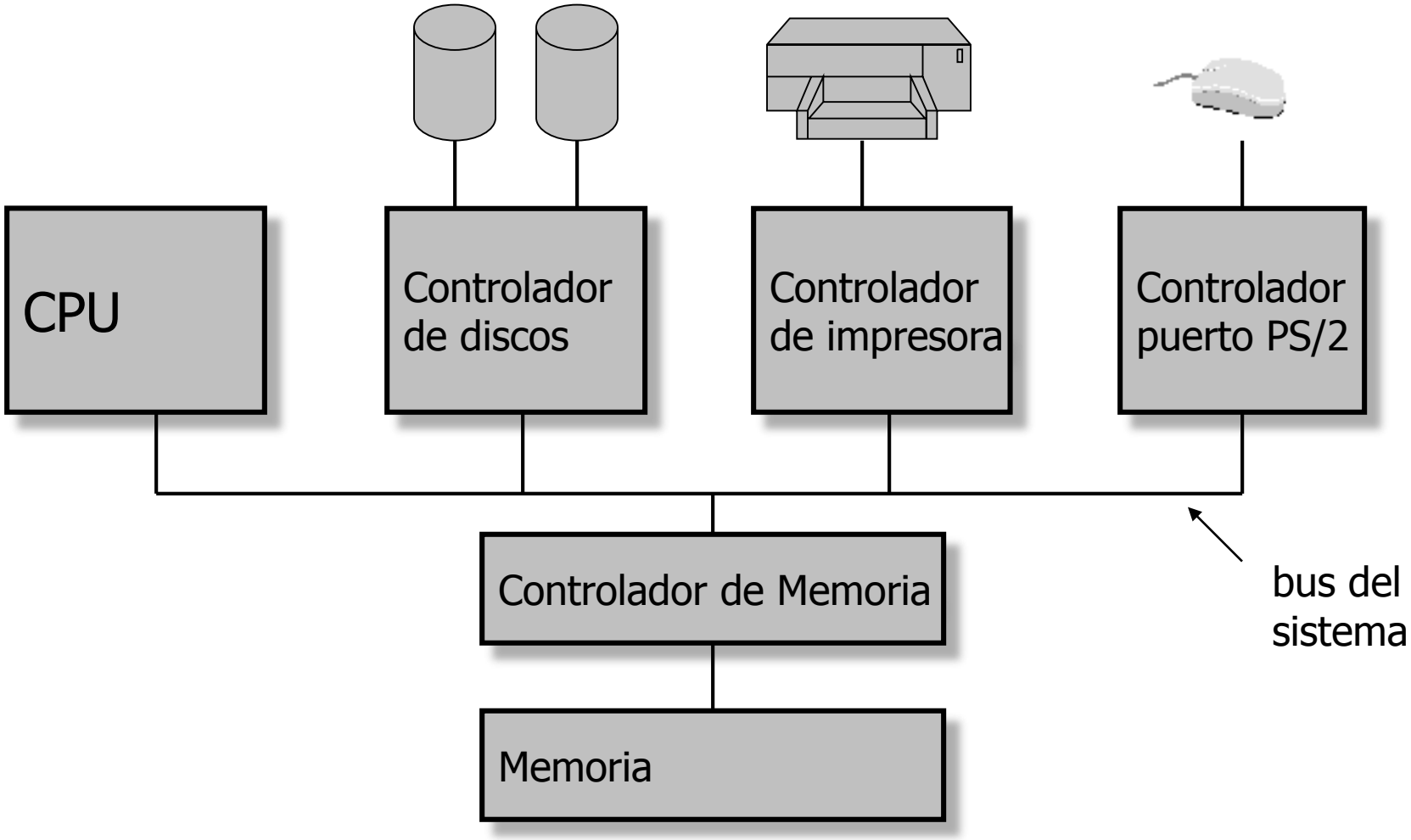
Tema 1. Conceptos generales
Estructura del computador y el SO

UNIVERSIDAD DE LAS PALMAS
DE GRAN CANARIA

Contenidos

- Estructura de la E/S
- Sistema de interrupciones
- Jerarquía de memorias
- Protección del hardware

Estructura de la E/S



Estructura de la E/S

- Los dispositivos se conectan al bus a través de **controladores de E/S**.
- La CPU se comunica con los controladores a través de instrucciones especiales o de direcciones de memoria concretas.
- Cada controlador tiene un búfer local. La CPU envía y recoge datos del búfer.
- El controlador notifica a la CPU la finalización de una operación o la llegada de nuevos datos mediante una **interrupción**.

Interrupciones

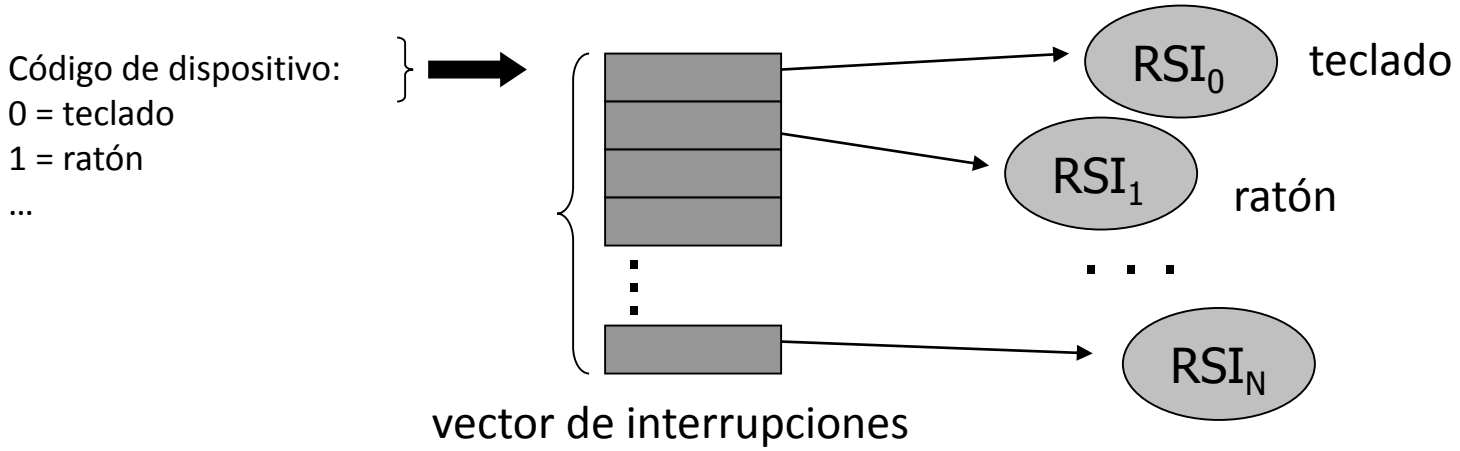
- Cuando llega una señal de interrupción a la CPU, ésta suspende lo que está haciendo y ejecuta una **rutina de servicio de interrupción (RSI)**.
- Antes de ejecutar la RSI, hay que guardar el estado de la CPU, para que pueda reanudar lo que estaba haciendo después de completar la RSI.

Interrupciones

- ¿Cómo sabe la CPU qué dispositivo ha interrumpido?
  - Método primitivo: preguntando a todos los dispositivos
    → ***polling***
  - Método avanzado: el dispositivo envía un número por el bus
    → **interrupciones vectorizadas**

Interrupciones vectorizadas

- **Vector de interrupciones**. Una zona de la memoria principal contiene las direcciones de todas las RSI.
- El dispositivo que interrumpe envía un número por el bus de datos. El número sirve de índice en el vector de interrupciones. La CPU ejecuta la RSI correspondiente.



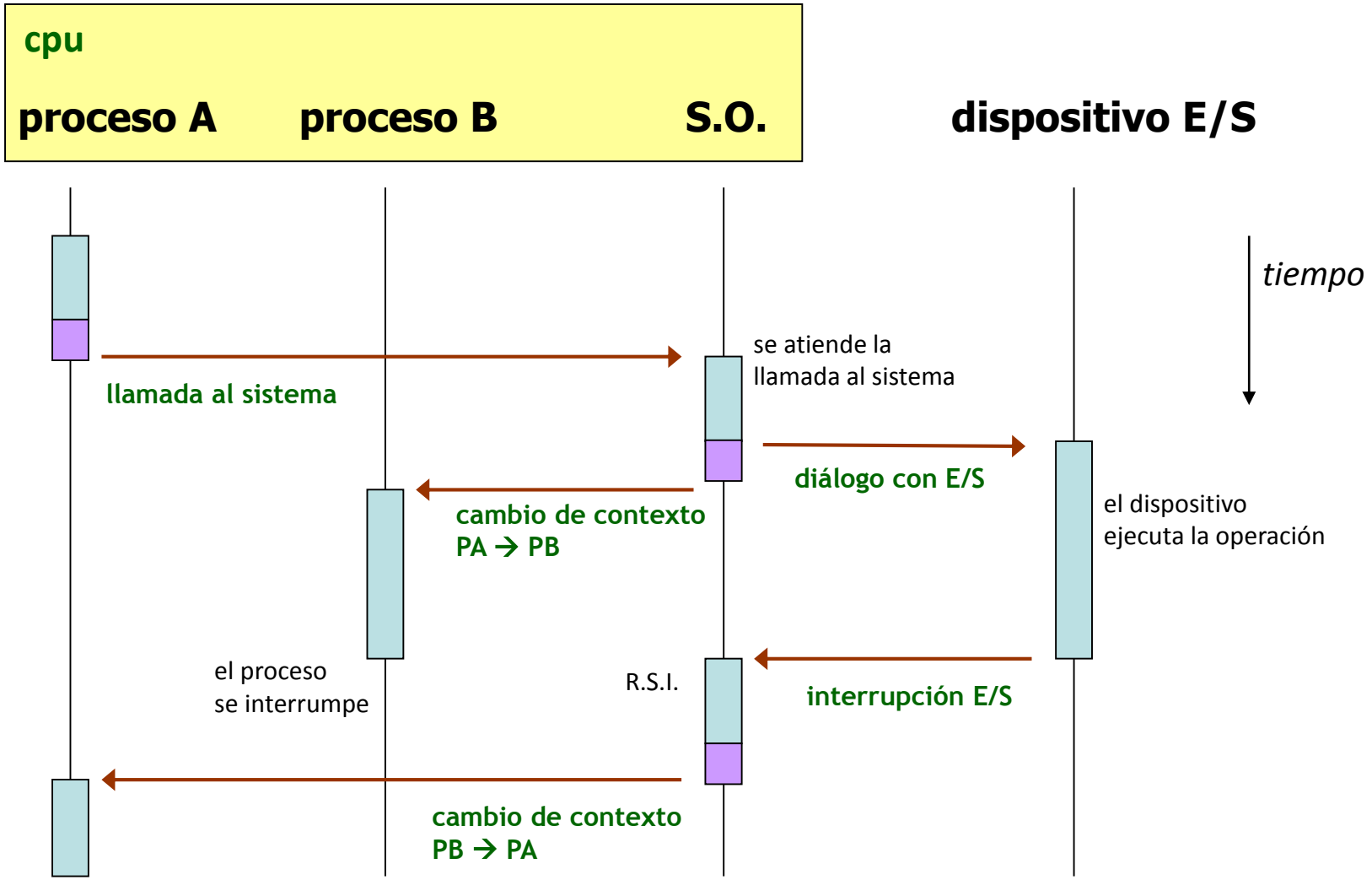
El SO es un software reactivo

- El SO no se activa por sí solo. Se activa cuando ocurre un evento que tiene que atender → comportamiento **reactivo**
- Tipos de eventos:
  - **Interrupciones del hardware**
  - **Llamadas al sistema**
  - **Excepciones**
- Cada tipo de evento activa una RSI diferente, indexada a través del vector de interrupciones

Sincronización entre E/S y SO

- Cuando el SO solicita a un periférico una operación de E/S, ¿cómo espera a que la operación finalice?
  - De forma **síncrona**: el SO deja la CPU en espera hasta que llega la interrupción de la E/S
  - De forma **asíncrona**: el SO cede la CPU a otros procesos mientras la E/S va trabajando.
- Si trabajamos de forma asíncrona, podemos sacar más rendimiento al sistema.

Ejemplo de funcionamiento asíncrono

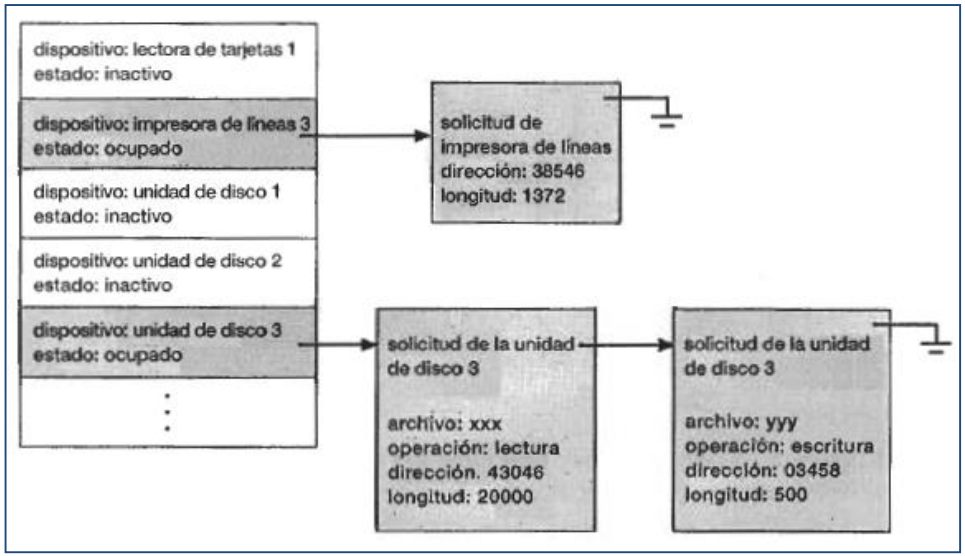


Explicación del diagrama anterior

- El proceso A invoca una operación de E/S a través de una llamada al sistema.
- El SO atiende la llamada y dialoga con la E/S.
- Como el proceso A queda en espera por la E/S, el SO decide traer a CPU al proceso B.
- Cuando la E/S finaliza, genera una interrupción.
- El SO atiende la interrupción y decide reanudar la ejecución del proceso A.

Colas de espera por E/S

- Cuando se solicita E/S, el dispositivo puede estar ya ocupado
- Hay que mantener en una **cola de espera** las peticiones pendientes
- A medida que vayan finalizando las operaciones, el SO va alimentando la E/S con peticiones encoladas

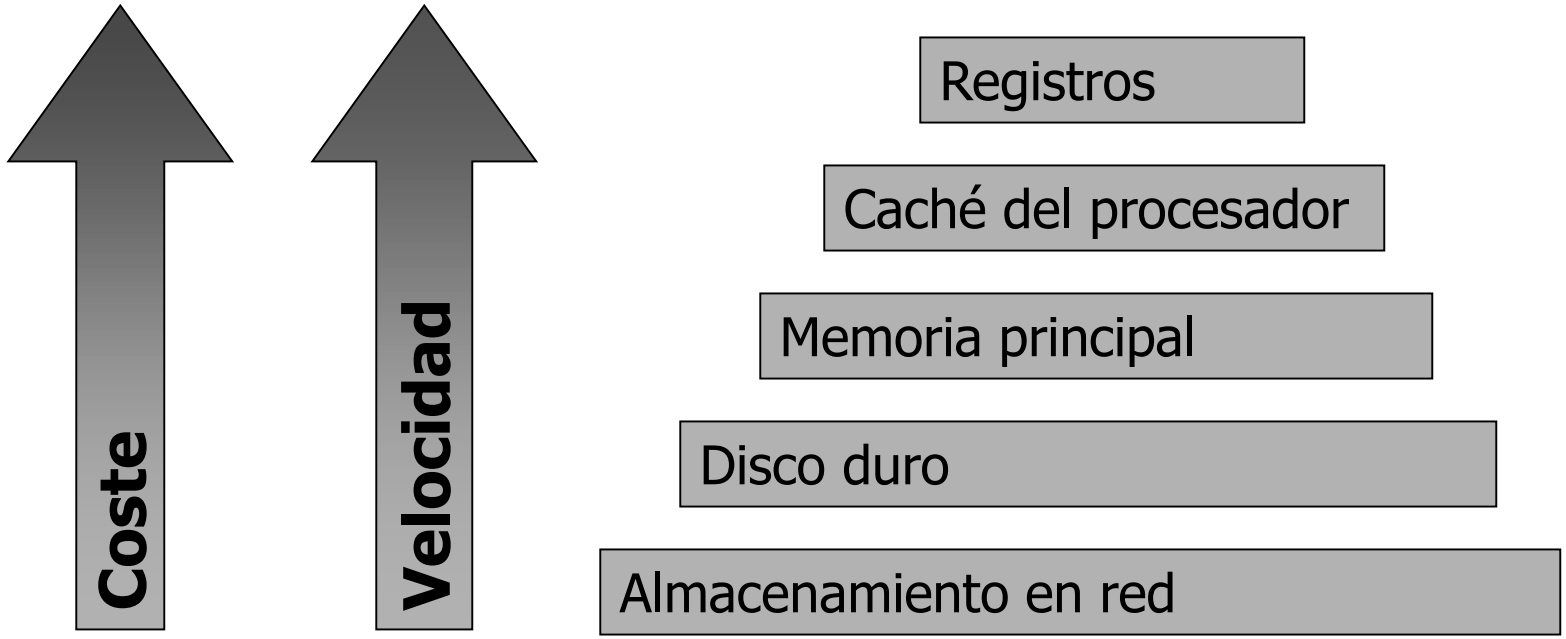


Interrupciones software

- Las llamadas al sistema y las excepciones funcionan como *interrupciones del software*: se gestionan igual que las interrs. del hardware, pero las provoca la propia CPU.
- Para provocar una interrupción software, existe una instrucción de máquina específica:
  - INT (Intel)
  - TRAP (Motorola)
  - SYSCALL (MIPS)

Jerarquía de memorias

- En un sistema informático, los medios de almacenamiento se pueden organizar en una jerarquía, según su coste y su velocidad.



Tecnologías de almacenamiento y su rendimiento

Level	1	2	3	4	5
Name	registers	cache	main memory	solid state disk	magnetic disk
Typical size	< 1 KB	< 16MB	< 64GB	< 1 TB	< 10 TB
Implementation technology	custom memory with multiple ports CMOS	on-chip or off-chip CMOS SRAM	CMOS SRAM	flash memory	magnetic disk
Access time (ns)	0.25 - 0.5	0.5 - 25	80 - 250	25,000 - 50,000	5,000,000
Bandwidth (MB/sec)	20,000 - 100,000	5,000 - 10,000	1,000 - 5,000	500	20 - 150
Managed by	compiler	hardware	operating system	operating system	operating system
Backed by	cache	main memory	disk	disk	disk or tape

(Silberschatz, Galvin, Gagne, 2013)

Repaso: unidades de medida de la información

- Bit = capaz de almacenar dos valores (ej. 0/1)
- Byte = 8 bits
  (ojo, en el pasado se usaron bytes de distintos tamaños)
- Palabra = 8..64 bits, dependiendo de la CPU

	Kilo	Mega	Giga	Tera	Peta
Sistema binario (ISO 80000-13)	1KiB = 2^{10}	1MiB = 2^{20}	1GiB = 2^{30}	1TiB = 2^{40}	1PiB = 2^{50}
Sistema decimal	1KB = 10^3 = 1000	1MB = 10^6 1.000.000	1GB = 10^9= 1 millardo	1TB = 10^{12} = 1 billón	1PB = 10^{15}= 1000 billones

¿Cómo gestionar la jerarquía de memorias? → memoria caché

- Aplicar el principio de **caché**: guardar en la memoria más rápida la información que se usa con más frecuencia.
  - Ejemplo: caché de disco
  - Ejemplo: caché de páginas web
- La caché se acaba llenando. En ese caso, hay que descartar bloques de información que ya no se necesitan → política de caché / política de reemplazo
  - Descartar el más antiguo (FIFO)
  - Descartar el que hace más tiempo que se usó (LRU)
  - Descartar el menos frecuentemente usado (LFU)
  - Al azar

Protección del hardware

- Modo dual de operación: operaciones privilegiadas
- Protección de la memoria
- Protección de la E/S
- Protección contra uso abusivo de la CPU

Protección del hardware

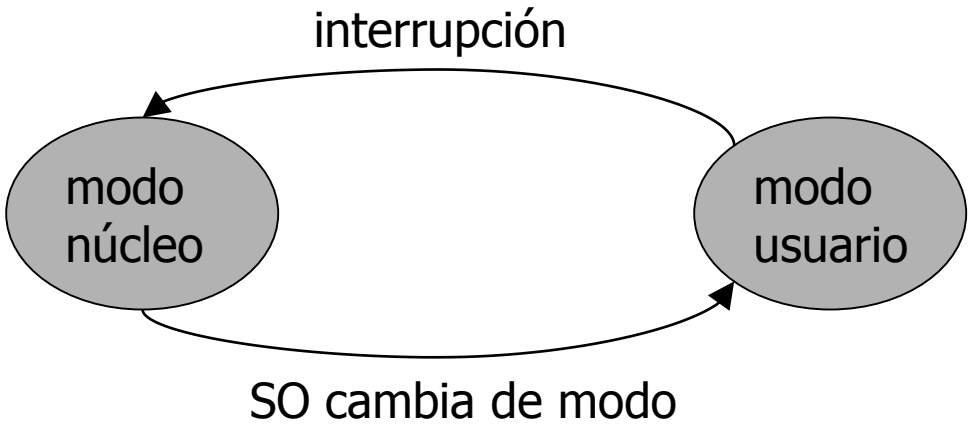
- Para que el S.O. funcione adecuadamente, hay que impedir que los programas de usuario puedan realizar libremente ciertas operaciones:
  - acceso a la memoria del S.O. y de otros programas
  - acceso directo a los dispositivos de E/S
  - utilizar la CPU todo el tiempo que quieran
- Solución: **modo dual de operación**

Modo dual de operación

- La CPU define un repertorio de **instrucciones privilegiadas**.
- Dos modo de operación del hardware:
  - **modo privilegiado/supervisor/sistema.** se pueden ejecutar todas las instrucciones.
  - **modo no privilegiado/usuario:** si se intenta ejecutar una instrucción privilegiada, la CPU interrumpe la ejecución y genera una **excepción**.

¿Cuándo y cómo se cambia de modo?

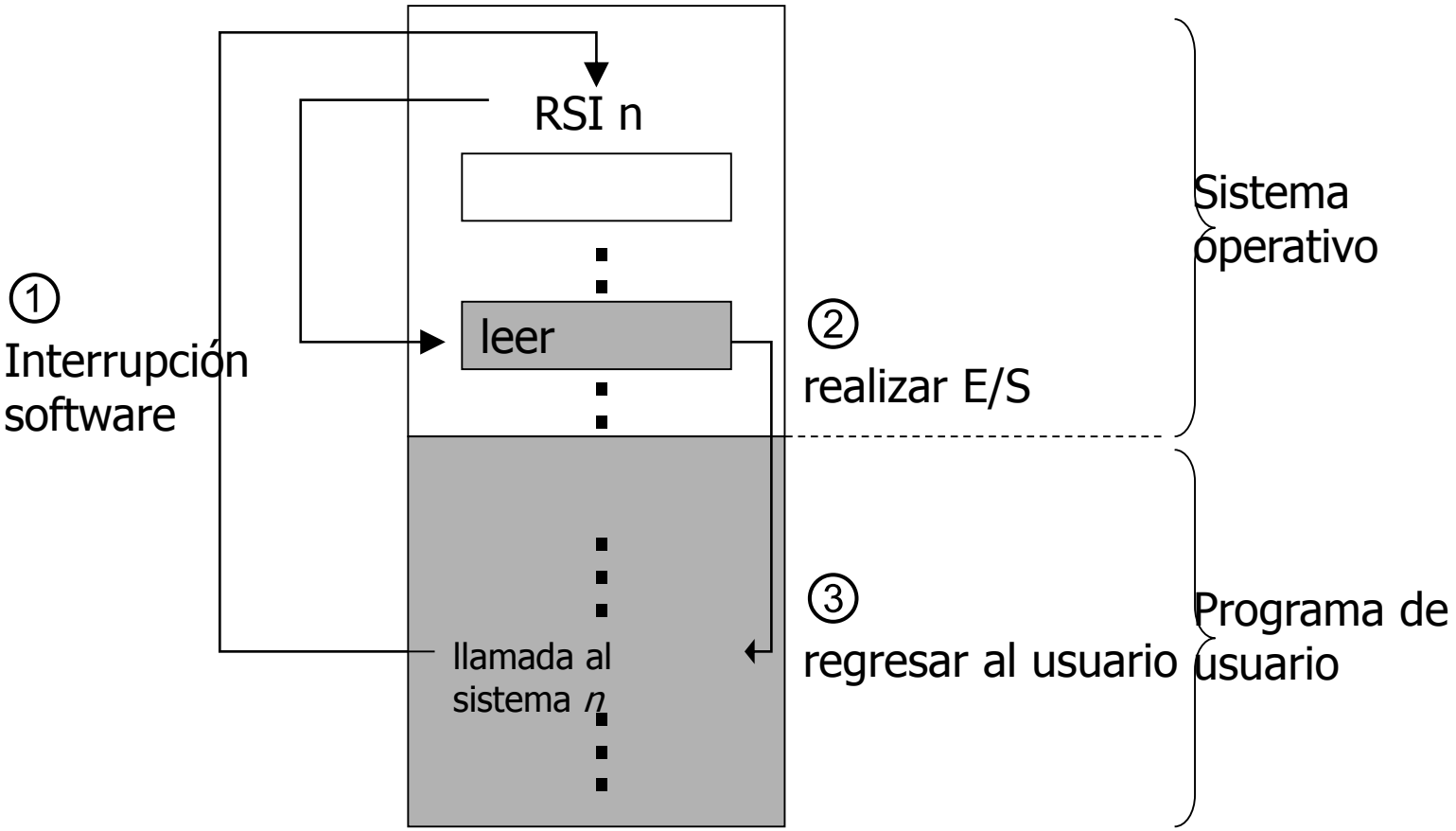
- La CPU arranca en *modo privilegiado*.
- Cuando el S.O. cede el control al usuario, conmuta previamente a *modo no privilegiado*.
- Sólo se vuelve a *modo privilegiado* cuando el S.O. recupera el control, es decir, cuando ocurre una interrupción, una llamada al sistema o una excepción.



Llamadas al sistema y modo dual

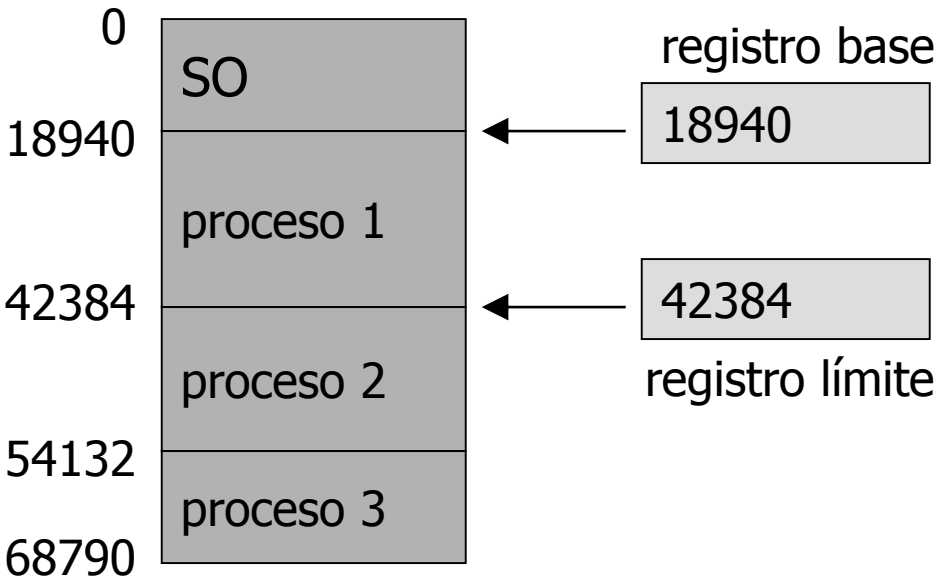
- La instrucción especial para llamada al sistema (syscall, trap, etc.) conmuta automáticamente a *modo privilegiado*.
- Por tanto, sirve para que el usuario cambie voluntariamente a *modo privilegiado*, pero ejecutando código del S.O. que no está bajo su control.

Ejemplo de llamada al sistema



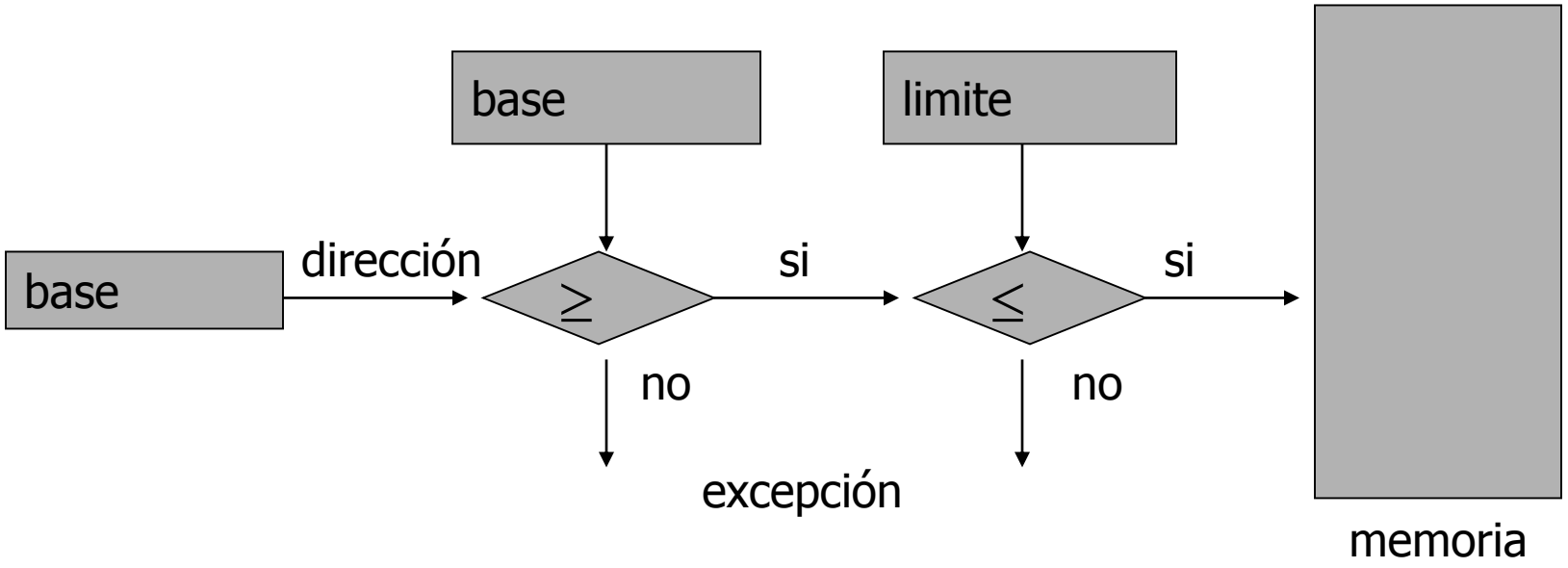
Protección de memoria

- Idea: pareja de **registros base y límite**, que delimitan la zona de memoria en la que el usuario está autorizado a trabajar.



Circuito de protección de memoria

- Cuando se está en *modo usuario*, se activa un circuito que verifica que la dirección de memoria es legal.
- Los accesos indebidos no salen al bus. En vez de ello, se produce una **excepción**.



Protección de E/S

- Las operaciones de E/S deben ser *privilegiadas*.
- Dos modelos de acceso a la E/S:
  - Con **instrucciones especiales** (in,out)
    → han de ser *privilegiadas* (sólo funcionan con bit 0).
  - A través de la **memoria** (memory mapped)
    → el acceso a las direcciones que usa la E/S debe estar prohibido en modo usuario.

Protección contra abusos de CPU

- Objetivo: evitar que un proceso acapare indefinidamente el tiempo de CPU (p.ej. al entrar en un bucle infinito)
- Solución: **temporizador**. Genera una interrupción tras un tiempo especificado (así el S.O. recupera el control):
    - contiene un contador inicializado al valor que se desee
    - el contador se decrementa con cada pulso de reloj del sistema
    - cuando el contador llega a cero, genera una interrupción

FIN
de la parte 2

Tema 1. Conceptos generales