

Grupo ARCOS

uc3m | Universidad **Carlos III** de Madrid

Tema 4 (III) El procesador

Estructura de Computadores
Grado en Ingeniería Informática

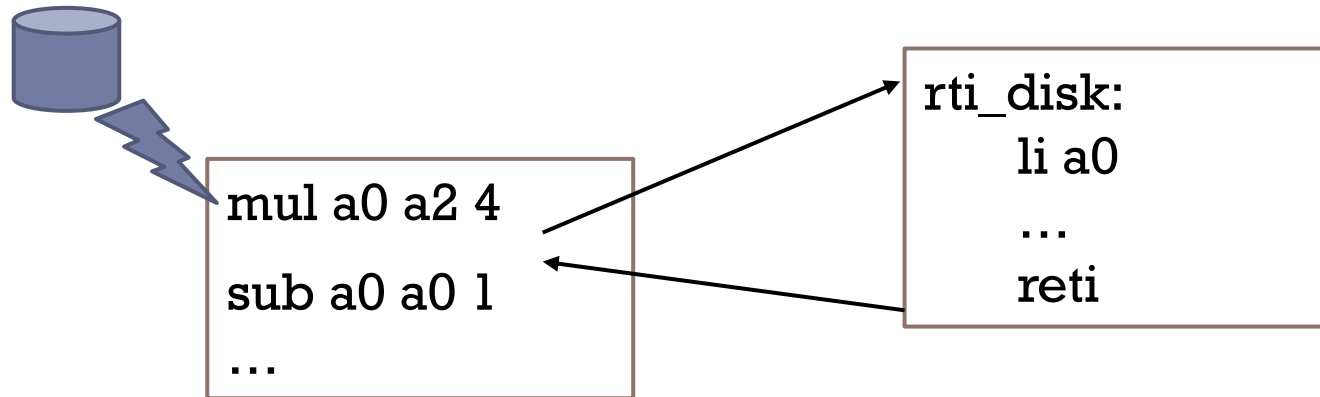
Contenido

1. Elementos de un computador
2. Organización del procesador
3. La unidad de control
4. Ejecución de instrucciones
5. Diseño de la unidad de control
6. Modos de ejecución
7. Interrupciones
8. Arranque de un computador
9. Prestaciones y paralelismo

Modos de ejecución

- ▶ Se indica con un bit situado en el **registro de estado (U)**
- ▶ Al menos 2 modos:
 - ▶ **Modo usuario**
 - ▶ El procesador no puede **ejecutar instrucciones privilegiadas** (ejemplo: instrucciones de E/S, de habilitación de interrupciones, ...)
 - ▶ Si un proceso de usuario ejecuta una instrucción privilegiada se produce una interrupción
 - ▶ **Modo núcleo**
 - ▶ Reservado al sistema operativo
 - ▶ El procesador puede ejecutar todo el repertorio de instrucciones

Idea de interrupción

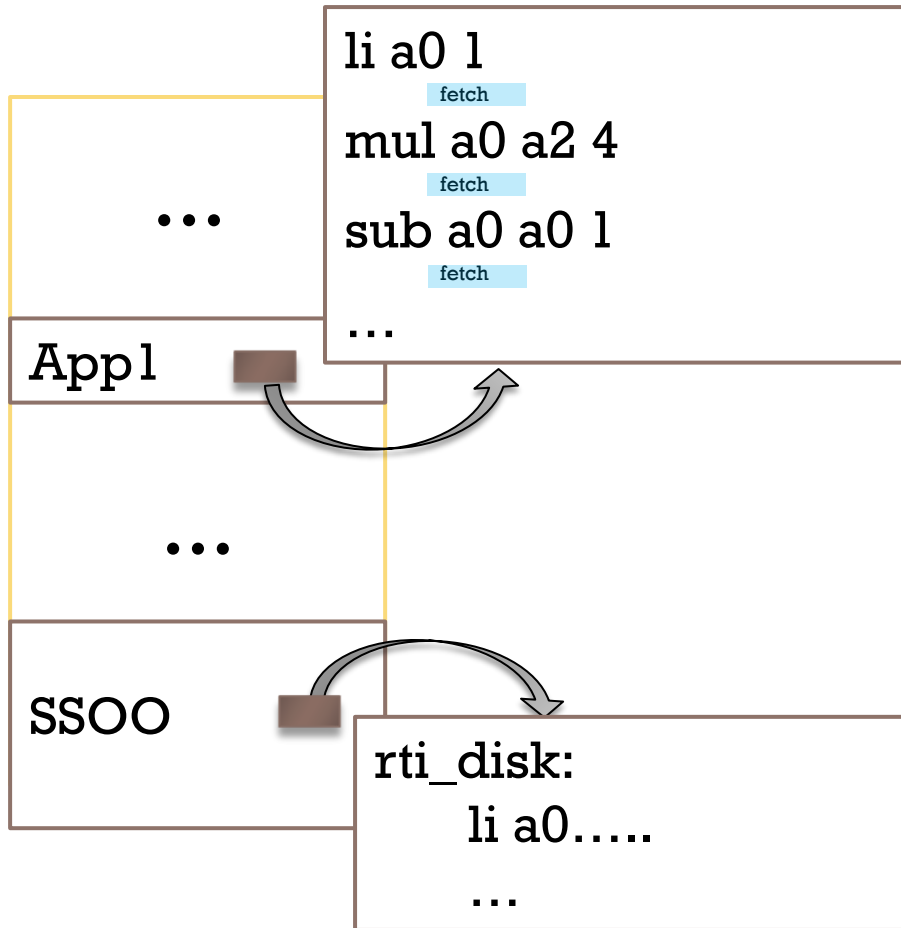


- ▶ Señal que llega a la U.C. y que rompe la secuencia normal de ejecución:
 - ▶ Se pausa la ejecución del programa actual y se transfiere la ejecución a otro programa que atiende la interrupción (ISR).
 - ▶ Al terminar el ISR la ejecución del programa se reanuda.
- ▶ Ejemplo de causas:
 - ▶ Cuando un periférico solicita la atención del procesador,
 - ▶ Cuando ocurre un error en la ejecución de la instrucción,
 - ▶ Etc.

Clasificación de las interrupciones

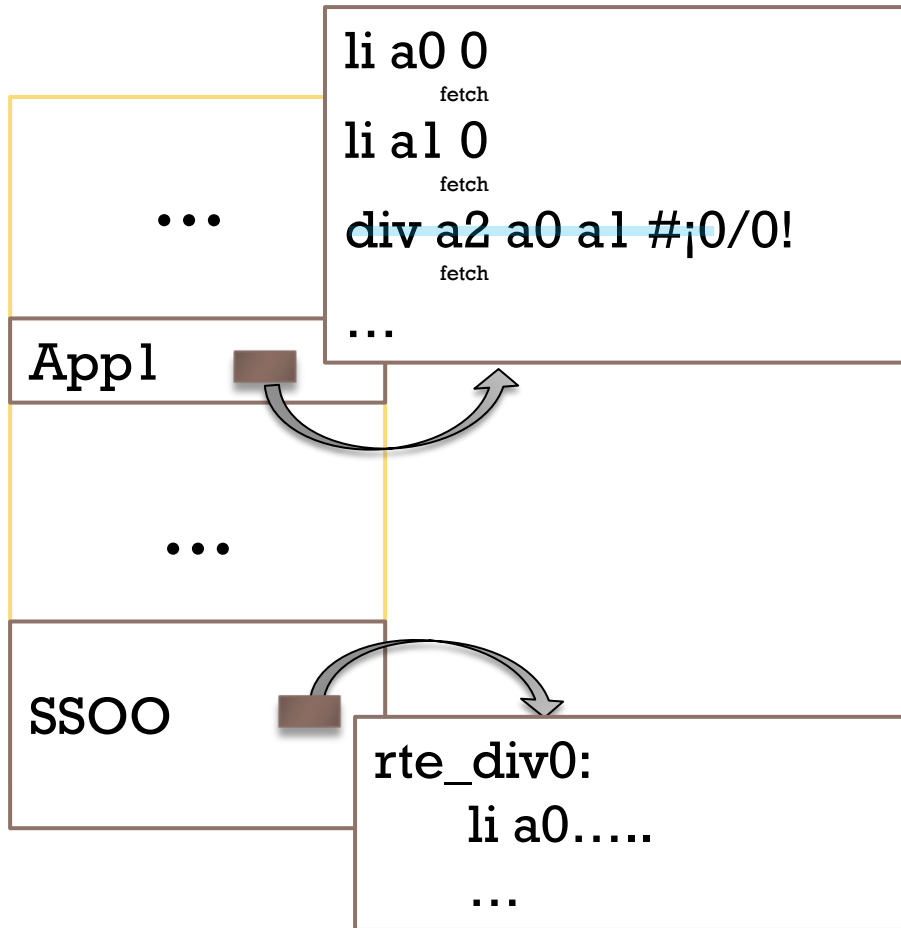
- ▶ Excepciones hardware **síncronas**
 - ▶ Cuando un **error** ocurre **en la ejecución** de la **instrucción en curso**: División por cero, acceso a una posición de memoria ilegal, etc.
- ▶ Excepciones hardware **asíncronas**
 - ▶ Fallos o **errores** en el hardware **no relacionados** con la **instrucción en curso**: impresora sin papel, etc.
- ▶ Interrupciones externas
 - ▶ Cuando un periférico precisa de atención por parte de la CPU: periféricos, interrupción del reloj
- ▶ Llamadas al sistema
 - ▶ Petición de servicio del sistema operativo
 - ▶ Instrucciones máquina especiales que generan una interrupción para activar al sistema operativo

Excepciones hardware asíncronas e Interrupciones externas



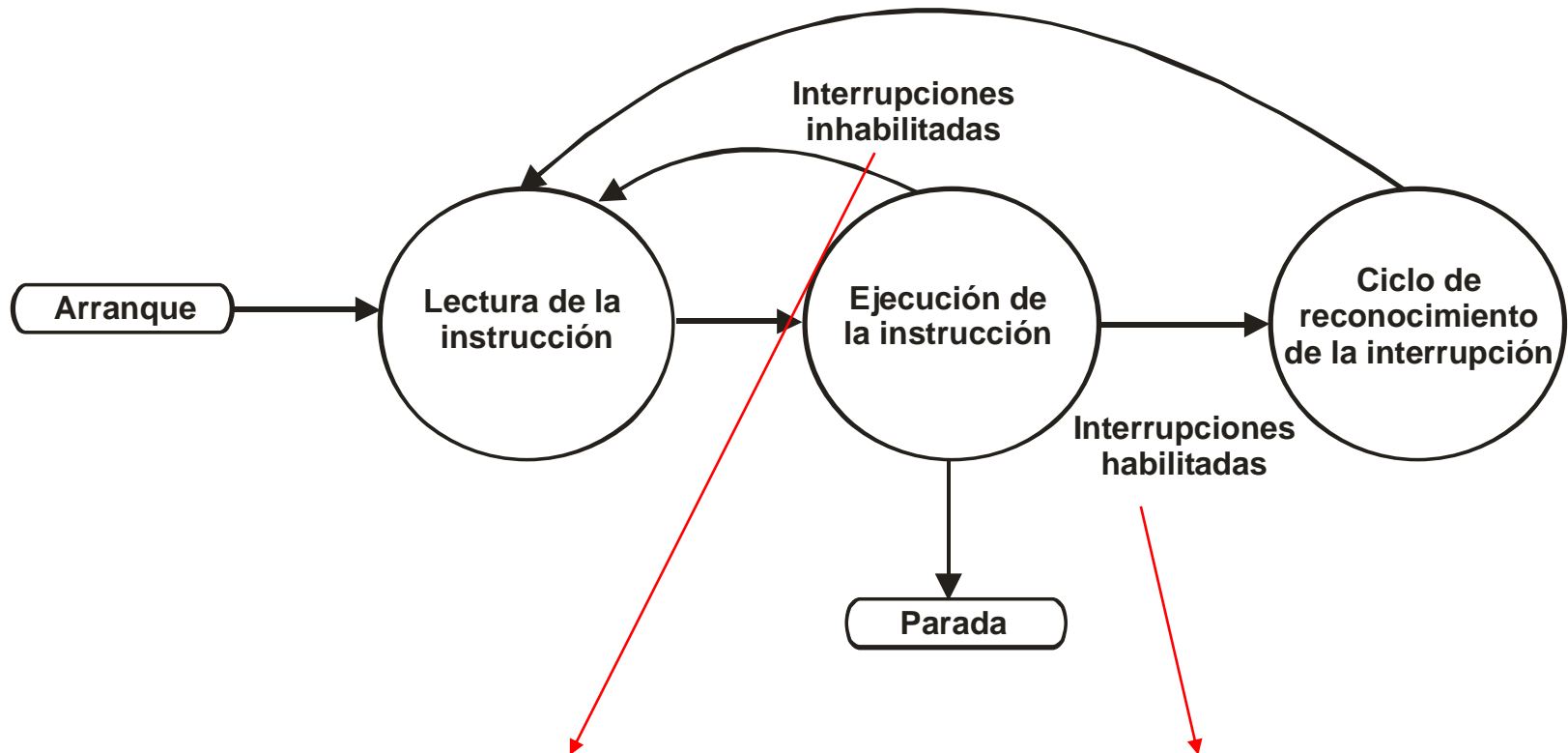
- ▶ Originan una ruptura de secuencia no programada
 - ▶ **Antes del ciclo de fetch de la siguiente instrucción, ver si hay interrupción pendiente, y si la hay...**
 - ▶ ...Bifurcación a subrutina del S.O. que la trata
- ▶ Posteriormente, restituye el estado y devuelve el control al programa interrumpido.
- **Causa asíncrona a la ejecución del programa en curso**
 - ▶ Atención a periférico
 - ▶ Etc.

Excepciones hardware síncronas



- ▶ Originan una ruptura de secuencia no programada
 - ▶ **Dentro del microprograma de la instrucción en curso...**
 - ▶ ...Bifurcación a subrutina del S.O. que la trata
- ▶ Posteriormente, restituye el estado y devuelve el control al programa interrumpido **o finaliza su ejecución**
- **Causa síncrona a la ejecución del programa en curso**
 - ▶ División entre cero
 - ▶ Etc.

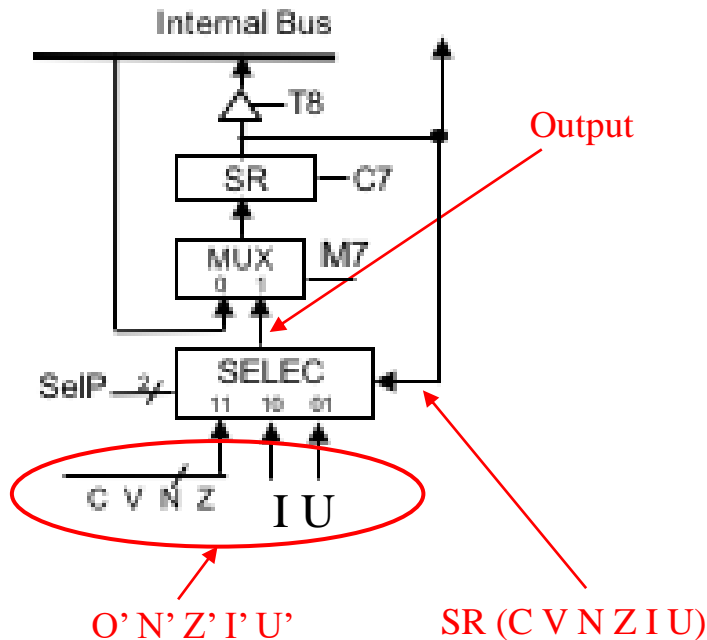
Activación de interrupción en el registro de estado



Se indica con un bit situado en el **registro de estado (I)**

Activación de interrupción en el registro de estado

Operación de SELEC:



if (SelP1 = 1 AND SelP0 == 1)
Output = **C' V' N' Z' I U**

```
if (SelP1 == 1 AND SelP0 ==0)
    Output = C V N Z I' U
```

if (SelP1 == 0 AND SelP0 == 1)
Output = C V N Z I **U'**

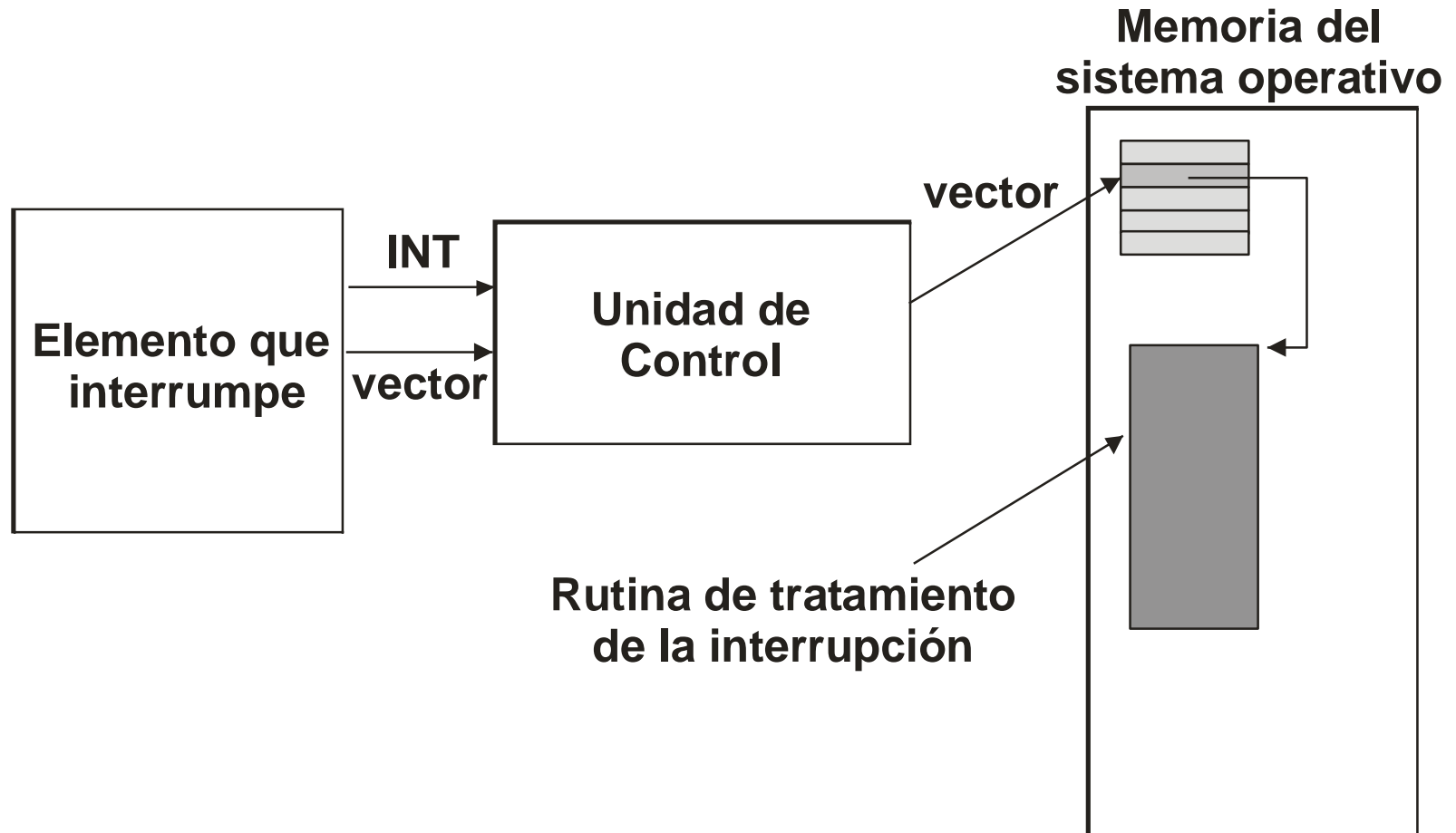
Ciclo de reconocimiento de la interrupción (CRI)

- ▶ Forma parte del microcódigo antes del ciclo de fetch
 - ▶ Trata especialmente las interrupciones asíncronas
- ▶ Estructura general del CRI:
 1. Comprueba se hay activada una señal de interrupción.
 2. Si está activada:
 1. Salva PC y RE (el contador de programa y el registro de estado)
 - Equivalent to “push pc, push sr”
 2. Pasa de modo usuario a modo núcleo
 - Equivalent to “SR.U = 0”
 3. Obtiene la dirección de la rutina de tratamiento de la interrupción
 - Equivalent to “isr_addr = Vector_interrupts[id_interrupt]”
 4. Almacena en el contador de programa la dirección obtenida (de esta forma la siguiente instrucción será la de la rutina de tratamiento)
 - Equivalent to “PC = isr_addr”

Rutina de tratamiento de la interrupción (RTI)

- ▶ Forma parte del código del sistema operativo
 - ▶ Hay una RTI por cada interrupción que pueda darse
- ▶ Estructura general de las RTI:
 1. Salva el resto de registros del procesador (que precise)
 2. Atiende la interrupción
 3. Restaura los registros del procesador guardados en (2)
 4. Ejecuta una instrucción máquina especial: RETI
 - ▶ Restaura el registro de estado del programa interrumpido (fijando de nuevo el modo del procesador a modo usuario)
 - ▶ Restaura el contador de programa (de forma que la siguiente instrucción es la del programa interrumpido).

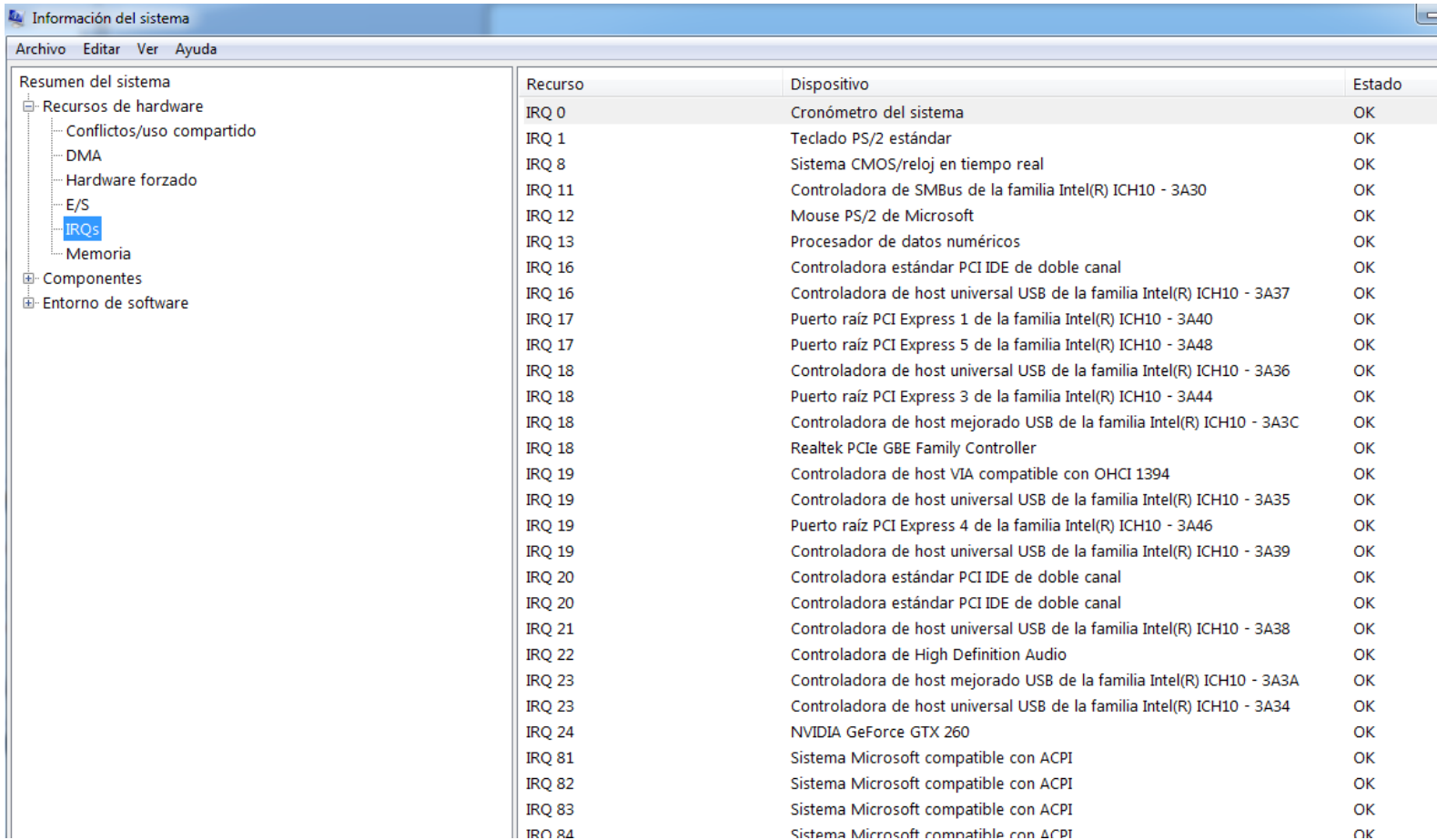
Interrupciones vectorizadas



Interrupciones vectorizadas

- ▶ El elemento que interrumpe suministra el **vector de interrupción**
- ▶ Este **vector** es un índice en una tabla que contiene la dirección de la rutina de tratamiento de la interrupción.
- ▶ La UC lee el contenido de esta entrada y carga el valor en el PC
- ▶ Cada sistema operativo rellena esta tabla con las direcciones de cada una de las rutinas de tratamiento, que son dependientes de cada sistema operativo.

Interrupciones en un PC con Windows



The screenshot shows the 'Información del sistema' window in Windows. The left sidebar has a tree view with 'Resumen del sistema' expanded, and 'IRQs' selected under 'Recursos de hardware'. The main pane displays a table of system resources and their interrupt request (IRQ) assignments.

Recurso	Dispositivo	Estado
IRQ 0	Cronómetro del sistema	OK
IRQ 1	Teclado PS/2 estándar	OK
IRQ 8	Sistema CMOS/reloj en tiempo real	OK
IRQ 11	Controladora de SMBus de la familia Intel(R) ICH10 - 3A30	OK
IRQ 12	Mouse PS/2 de Microsoft	OK
IRQ 13	Procesador de datos numéricos	OK
IRQ 16	Controladora estándar PCI IDE de doble canal	OK
IRQ 16	Controladora de host universal USB de la familia Intel(R) ICH10 - 3A37	OK
IRQ 17	Puerto raíz PCI Express 1 de la familia Intel(R) ICH10 - 3A40	OK
IRQ 17	Puerto raíz PCI Express 5 de la familia Intel(R) ICH10 - 3A48	OK
IRQ 18	Controladora de host universal USB de la familia Intel(R) ICH10 - 3A36	OK
IRQ 18	Puerto raíz PCI Express 3 de la familia Intel(R) ICH10 - 3A44	OK
IRQ 18	Controladora de host mejorado USB de la familia Intel(R) ICH10 - 3A3C	OK
IRQ 18	Realtek PCIe GBE Family Controller	OK
IRQ 19	Controladora de host VIA compatible con OHCI 1394	OK
IRQ 19	Controladora de host universal USB de la familia Intel(R) ICH10 - 3A35	OK
IRQ 19	Puerto raíz PCI Express 4 de la familia Intel(R) ICH10 - 3A46	OK
IRQ 19	Controladora de host universal USB de la familia Intel(R) ICH10 - 3A39	OK
IRQ 20	Controladora estándar PCI IDE de doble canal	OK
IRQ 20	Controladora estándar PCI IDE de doble canal	OK
IRQ 21	Controladora de host universal USB de la familia Intel(R) ICH10 - 3A38	OK
IRQ 22	Controladora de High Definition Audio	OK
IRQ 23	Controladora de host mejorado USB de la familia Intel(R) ICH10 - 3A3A	OK
IRQ 23	Controladora de host universal USB de la familia Intel(R) ICH10 - 3A34	OK
IRQ 24	NVIDIA GeForce GTX 260	OK
IRQ 81	Sistema Microsoft compatible con ACPI	OK
IRQ 82	Sistema Microsoft compatible con ACPI	OK
IRQ 83	Sistema Microsoft compatible con ACPI	OK
IRQ 84	Sistema Microsoft compatible con ACPI	OK

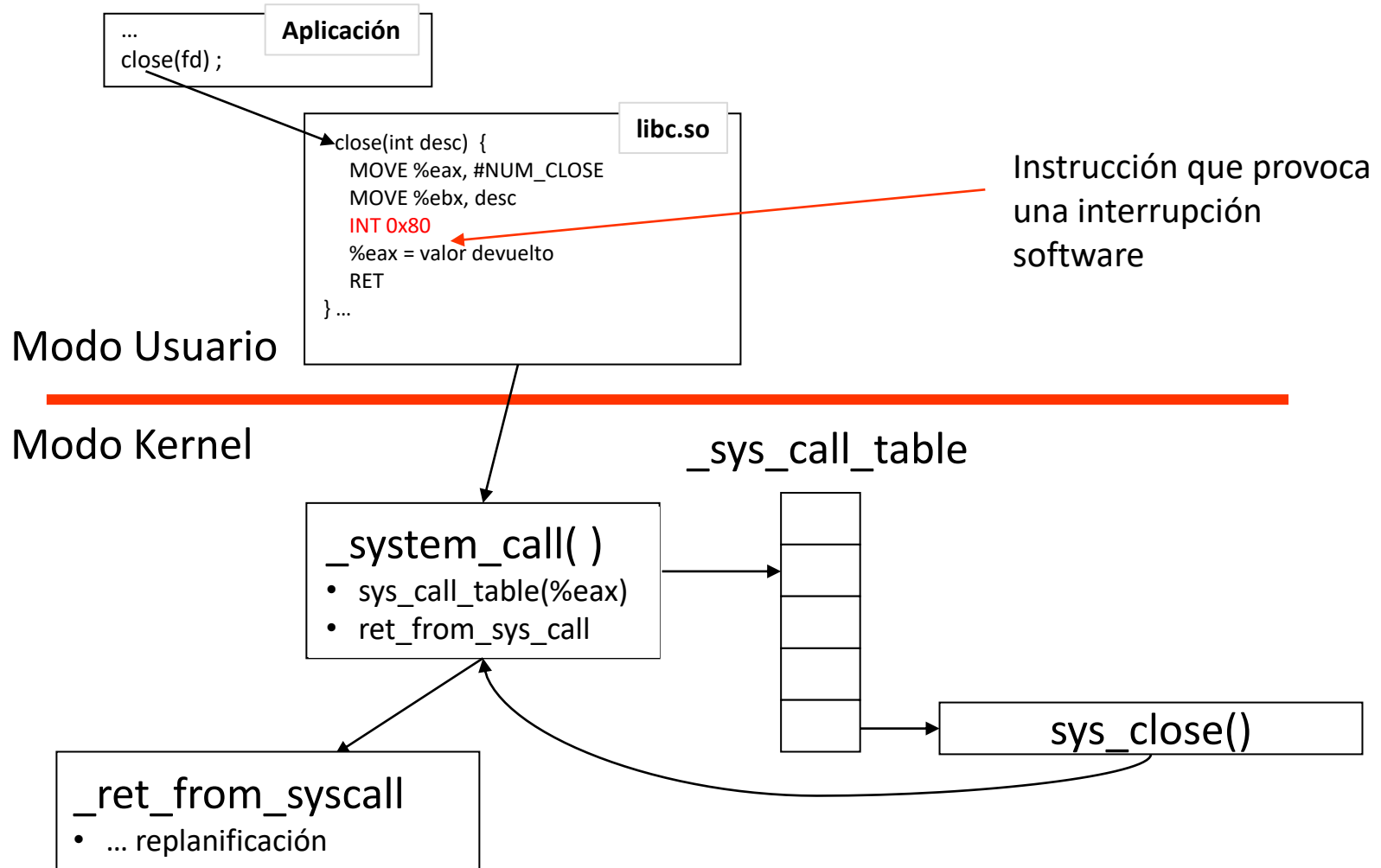
Interrupciones por software.

Llamadas al sistema y sistemas operativos

- ▶ El mecanismo de llamadas al sistema es el que permite que los programas de usuario puedan solicitar los servicios que ofrece el sistema operativo
 - ▶ Cargar programas en memoria para su ejecución
 - ▶ Acceso a los dispositivos periféricos
 - ▶ Etc.
- ▶ Similar a las llamadas al sistema que ofrece el simulador CREATOR
 - ▶ Hay ejemplos en WepSIM que muestran cómo internamente se puede implementar las llamadas al sistema

Interrupciones software

Llamadas al sistema (ejemplo: Linux)



Interrupciones del reloj y sistemas operativos

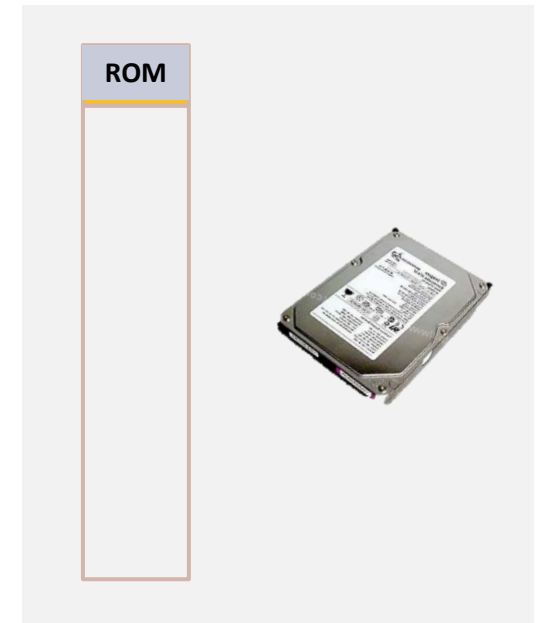
- ▶ La señal que gobierna la ejecución de las instrucciones máquina se divide mediante un divisor de frecuencia para generar una interrupción externa cada cierto intervalo de tiempo (pocos milisegundos)
- ▶ Estas **interrupciones de reloj** o tics son interrupciones periódicas que permite que el sistema operativo entre a ejecutar de forma periódica evitando que un programa de usuario monopolice la CPU
 - ▶ Permite alternar la ejecución de diversos programas en un sistema dado la apariencia de ejecución simultánea
 - ▶ Cada vez que llega una interrupción de reloj se suspende al programa y se salta al sistema operativo que ejecuta el **planificador** para decidir el **siguiente** programa a ejecutar

Contenido

1. Elementos de un computador
2. Organización del procesador
3. La unidad de control
4. Ejecución de instrucciones
5. Diseño de la unidad de control
6. Modos de ejecución
7. Interrupciones
8. Arranque de un computador
9. Prestaciones y paralelismo

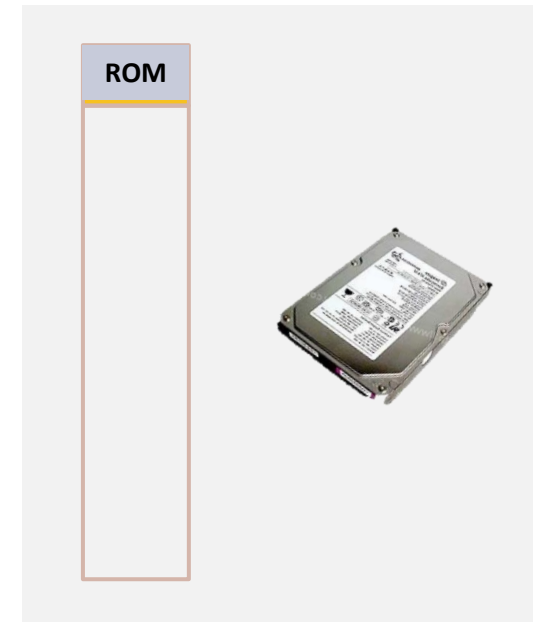
Arranque del computador

- ▶ El *Reset* carga en los registros sus valores predefinidos
 - ▶ $PC \leftarrow$ dirección de arranque del programa iniciador (en memoria ROM)



Arranque del computador

- ▶ El Reset carga en los registros sus valores predefinidos
 - ▶ PC ← dirección de arranque del programa iniciador (en memoria ROM)
- ▶ Se ejecuta el programa iniciador
 - ▶ Test del sistema (POST)



```
Award Modular BIOS v6.00PG, An Energy Star Ally
Copyright (C) 1984-2007, Award Software, Inc.

Intel X38 BIOS for X38-DQ6 F4

Main Processor : Intel(R) Core(TM)2 Extreme CPU X9650 @ 4.00GHz(333x12)
<CPUID:0676 Patch ID:0000>
Memory Testing : 2096064K OK

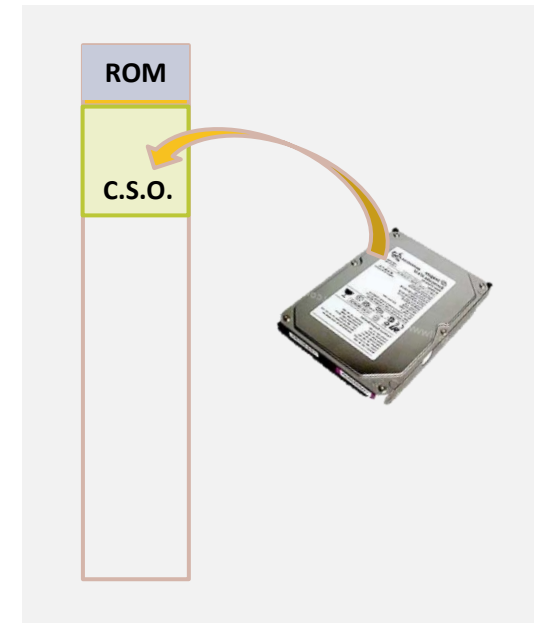
Memory Runs at Dual Channel Interleaved
IDE Channel 0 Slave : WDC WD3200AAJS-00RYA0 12.01B01
IDE Channel 1 Slave : WDC WD3200AAJS-00RYA0 12.01B01

Detecting IDE drives ...
IDE Channel 4 Master : None
IDE Channel 4 Slave : None
IDE Channel 5 Master : None
IDE Channel 5 Slave : None

<DEL>:BIOS Setup <F9>:XpressRecoveryZ <F12>:Boot Menu <End>:Quit
09/19/2007-X38-ICH9-6A79060QC-00
```

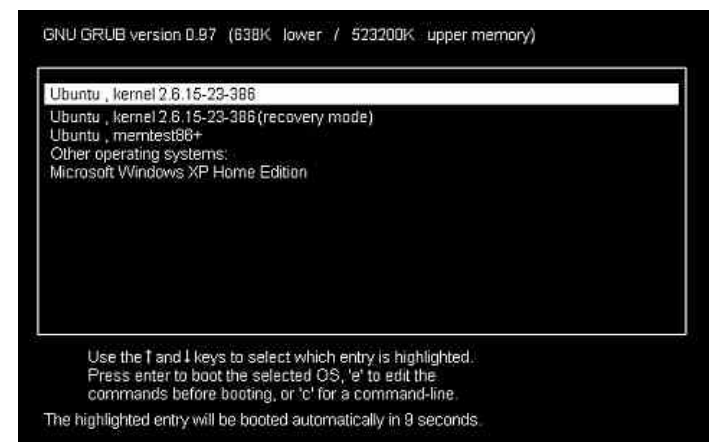
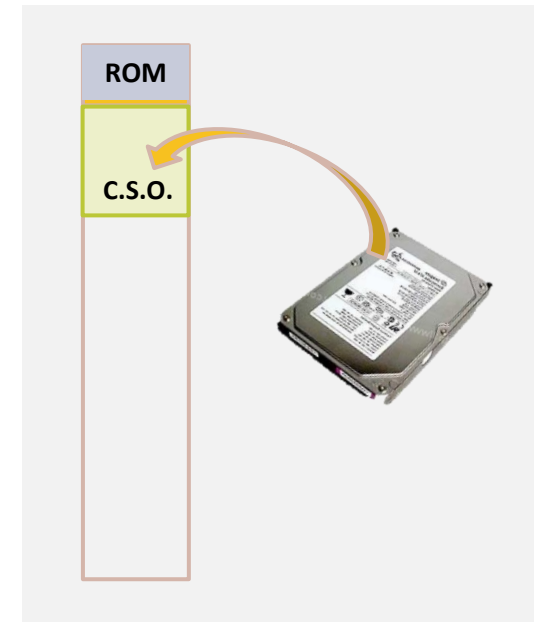
Arranque del computador

- ▶ El *Reset* carga en los registros sus valores predefinidos
 - ▶ PC ← dirección de arranque del **programa iniciador** (en memoria ROM)
- ▶ Se ejecuta el **programa iniciador**
 - ▶ Test del sistema (POST)
 - ▶ Carga en memoria el **cargador del sistema operativo (MBR)**



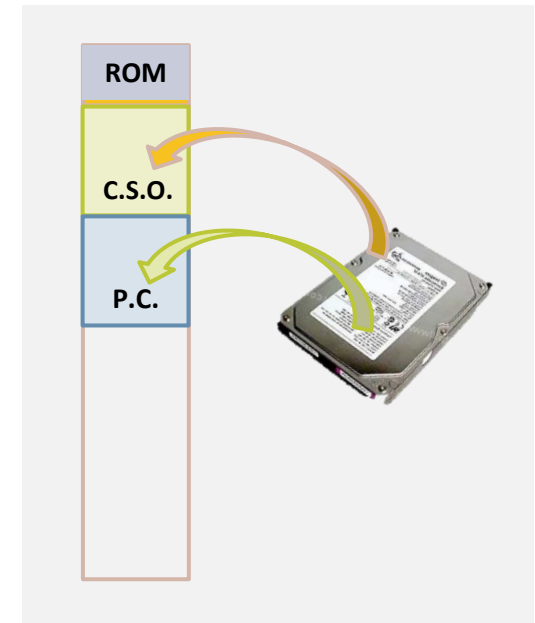
Arranque del computador

- ▶ El *Reset* carga en los registros sus valores predefinidos
 - ▶ PC ← dirección de arranque del **programa iniciador** (en memoria ROM)
- ▶ Se ejecuta el **programa iniciador**
 - ▶ Test del sistema (POST)
 - ▶ Carga en memoria el **cargador del sistema operativo (MBR)**
- ▶ Se ejecuta el **cargador del sistema operativo**
 - ▶ Establece opciones de arranque



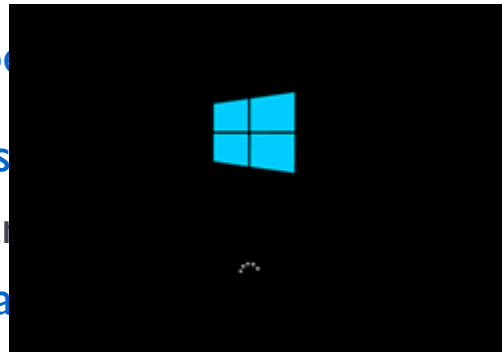
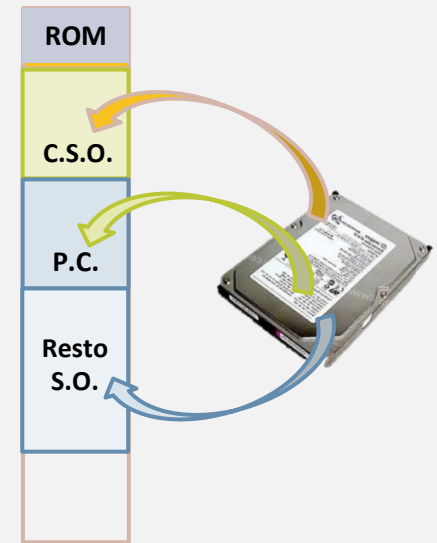
Arranque del computador

- ▶ El *Reset* carga en los registros sus valores predefinidos
 - ▶ PC ← dirección de arranque del **programa iniciador** (en memoria ROM)
- ▶ Se ejecuta el **programa iniciador**
 - ▶ Test del sistema (POST)
 - ▶ Carga en memoria el **cargador del sistema operativo (MBR)**
- ▶ Se ejecuta el **cargador del sistema operativo**
 - ▶ Establece opciones de arranque
 - ▶ Carga el **programa de carga**



Arranque del computador

- ▶ El **Reset** carga en los registros sus valores predefinidos
 - ▶ PC ← dirección de arranque del **programa iniciador** (en memoria ROM)
- ▶ Se ejecuta el **programa iniciador**
 - ▶ Test del sistema (POST)
 - ▶ Carga en memoria el **cargador del sistema operativo**
- ▶ Se ejecuta el **cargador del sistema operativo**
 - ▶ Establece opciones de arranque
 - ▶ Carga el **programa de carga**
- ▶ Se ejecuta el **programa de carga**
 - ▶ Establece estado inicial para el S.O.
 - ▶ Carga el sistema operativo y lo ejecuta

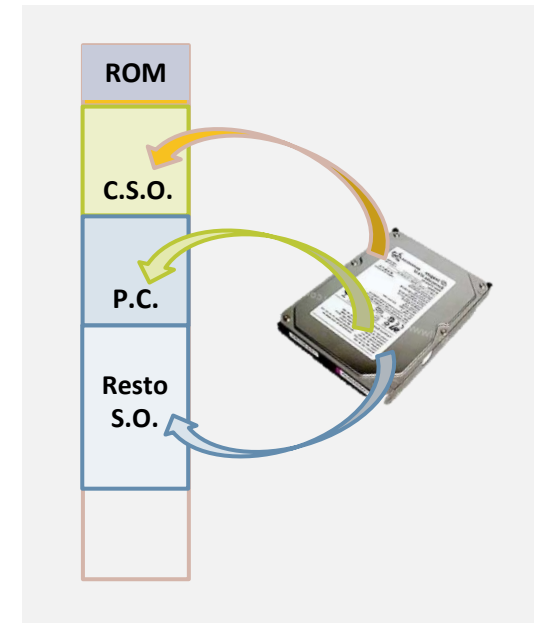


Una imagen de la pantalla de arranque de Linux, mostrando un pinguino Tux en la esquina superior izquierda y un texto de configuración en blanco sobre fondo negro. El texto incluye: 'Configuring ISA PNP', 'Setting system time from the hardware clock (localtime).', 'Using /etc/random-seed to initialize /dev/urandom.', 'Initializing basic system settings ...', 'Updating shared libraries', 'Setting hostname: engpc23.murdoch.edu.au', 'INIT: Entering runlevel: 4', 'rc.M ==> Going multiuser ...', 'Starting system logger ...', 'Initialising advanced hardware', 'Setting up modules ...', 'Initialising network', 'Setting up localhost ...', 'Setting up inet1 ...', 'Setting up route ...', 'Setting up fancy console and GUI', 'Loading fc-cache ...', 'rc.v1init ==> Going to runlevel 4', 'Starting services of runlevel 4', 'Starting dnsmasq ...', '==> rc.X Going to multiuser GUI mode ...', 'XFree86 Display Manager', 'Framebuffer /dev/fb0 is 307200 bytes.', 'Grabbing 640x480 ...'. Hay indicadores de estado como '[OK]' al final de algunas líneas.

Arranque del computador

resumen

- ▶ El *Reset* carga en los registros sus valores predefinidos
 - ▶ PC ← dirección de arranque del **programa iniciador** (en memoria ROM)
- ▶ Se ejecuta el **programa iniciador**
 - ▶ Test del sistema (POST)
 - ▶ Carga en memoria el **cargador del sistema operativo (MBR)**
- ▶ Se ejecuta el **cargador del sistema operativo**
 - ▶ Establece opciones de arranque
 - ▶ Carga el **programa de carga**
- ▶ Se ejecuta el **programa de carga**
 - ▶ Establece estado inicial para el S.O.
 - ▶ Carga el sistema operativo y lo ejecuta



Tiempo de ejecución de un programa

Iron law of processor performance

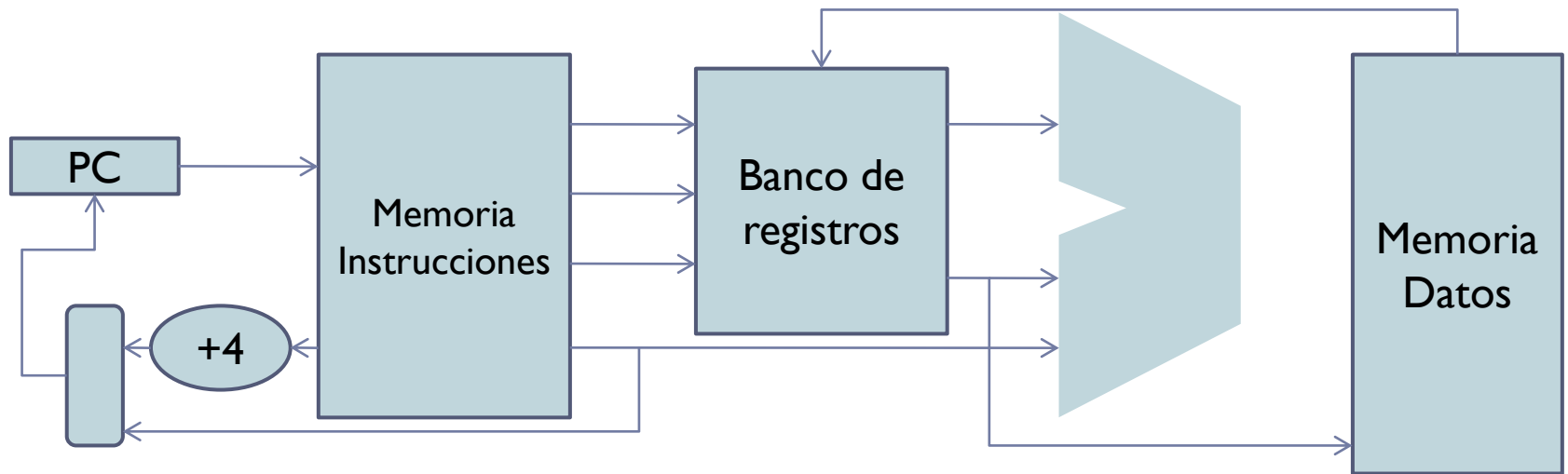
$$\text{Tiempo}_{\text{ejecución}} = \text{NI} \times \text{CPI} \times t_{\text{ciclo_CPU}} + \text{NI} \times \text{AMI} \times t_{\text{ciclo_mem}}$$

- ▶ **NI** es el número de instrucciones máquina del programa
- ▶ **CPI** es el número medio de ciclos de reloj necesario para ejecutar una instrucción
- ▶ **$t_{\text{ciclo_CPI}}$** es el tiempo que dura el ciclo de reloj del procesador
- ▶ **AMI** es el número medio de accesos a memoria por instrucción
- ▶ **$t_{\text{ciclo_mem}}$** es el tiempo de un acceso a memoria

Factores que afecta al tiempo de ejecución

	NI	CPI	$t_{\text{ciclo_CPI}}$	AMI	$t_{\text{ciclo_mem}}$
Programa	✓			✓	
Compilador	✓	✓		✓	
Juego de instrucciones (ISA)	✓	✓	✓	✓	
Organización		✓	✓		✓
Tecnología			✓		✓

Modelo de procesador basado en camino de datos (sin bus interno)



Paralelismo a nivel de instrucción

- ▶ Procesamiento concurrente de varias instrucciones
- ▶ Combinación de elementos que trabajan en paralelo:
 - ▶ **Procesadores segmentados**: utilizan técnicas de pipeline para procesar varias instrucciones simultáneamente
 - ▶ **Procesadores superescalares**: procesador segmentado que puede ejecutar varias instrucciones en paralelo cada una de ellas en una unidad segmentada diferente
 - ▶ **Procesadores multicore**: procesador que combina dos o más procesadores independientes en un solo empaquetado

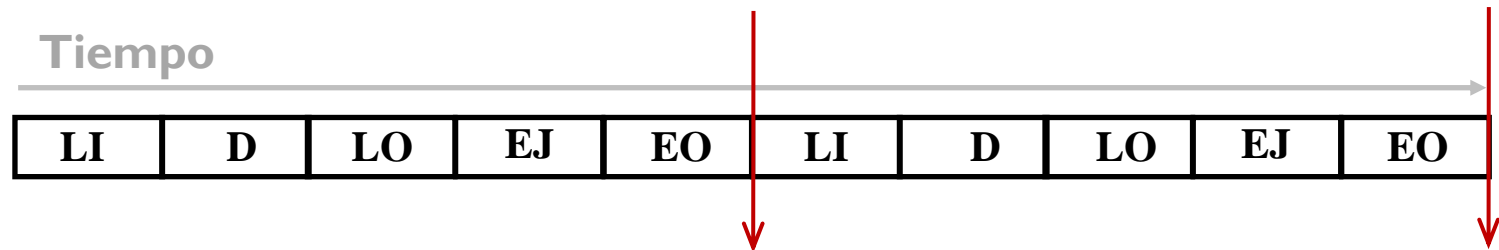
Segmentación de instrucciones



- ▶ Etapas de ejecución de una instrucción:
 - ▶ **LI**: Lectura de la instrucción e incremento del PC
 - ▶ **D**: Decodificación
 - ▶ **LO**: Lectura de Operandos
 - ▶ **Ej**: Ejecución de la instrucción
 - ▶ **EO**: Escritura de Operandos

Segmentación de instrucciones

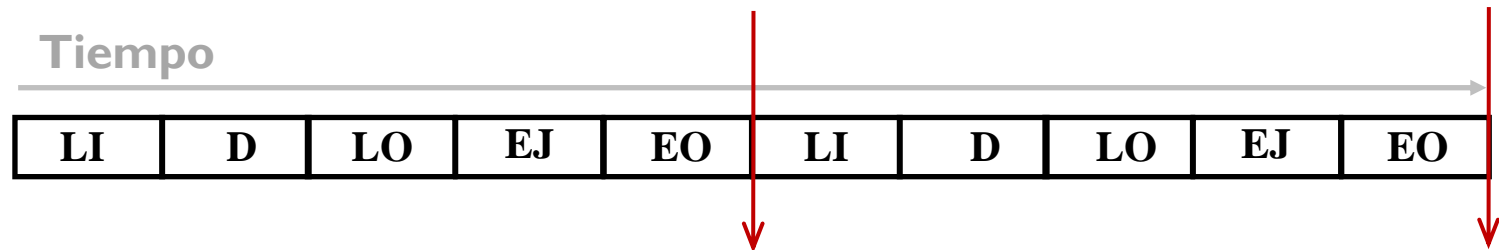
sin pipeline



- ▶ Etapas de ejecución de una instrucción:
 - ▶ **LI**: Lectura de la instrucción e incremento del PC
 - ▶ **D**: Decodificación
 - ▶ **LO**: Lectura de Operandos
 - ▶ **EJ**: Ejecución de la instrucción
 - ▶ **EO**: Escritura de Operandos

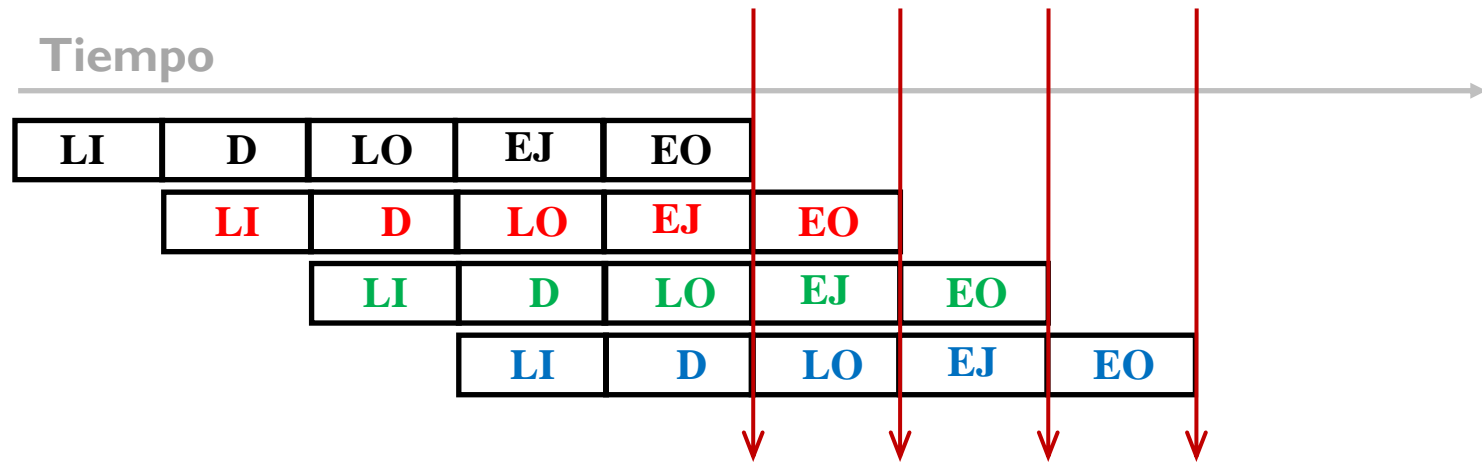
Segmentación de instrucciones

sin pipeline



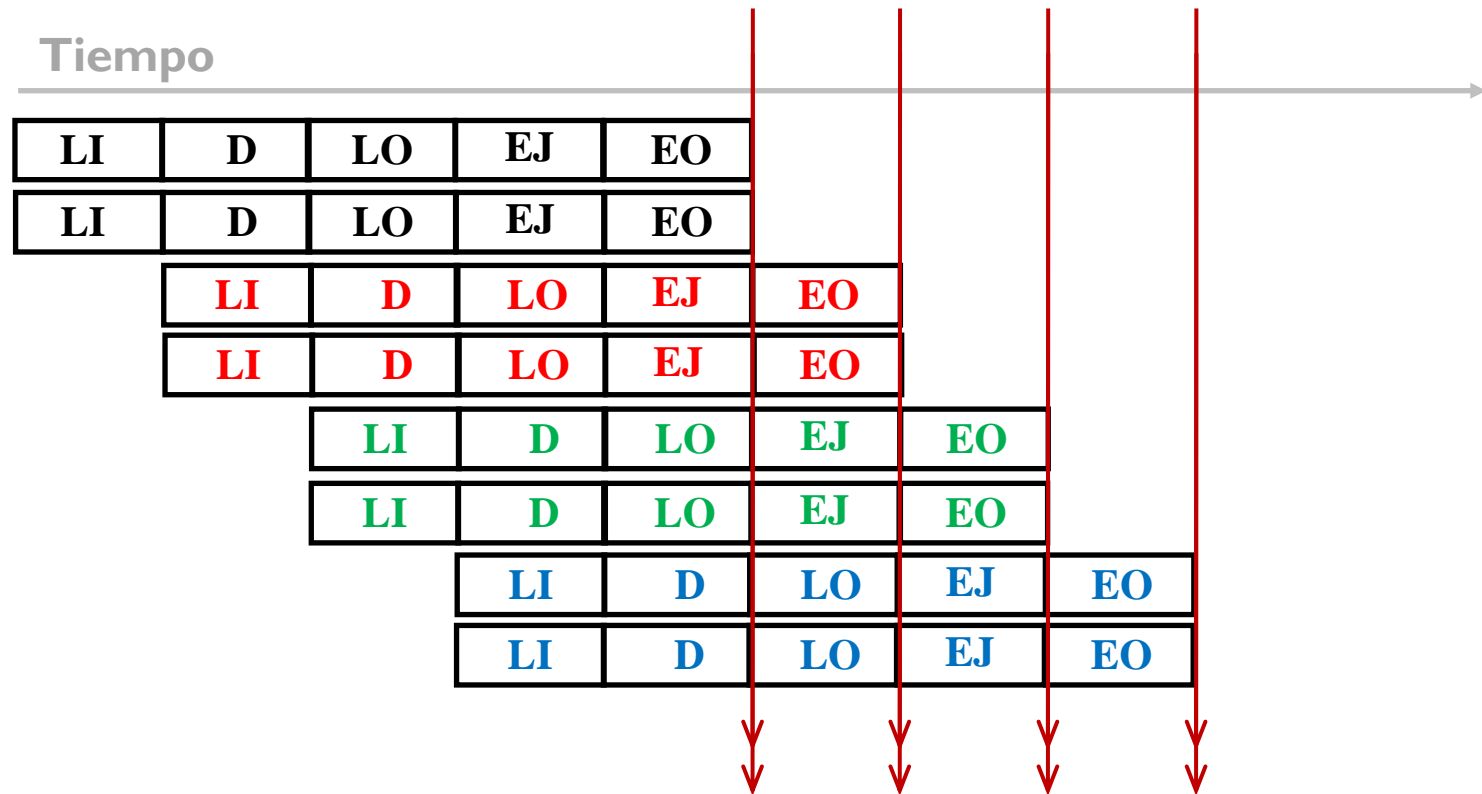
- ▶ Si cada fase dura N ciclos de reloj, entonces
 - ▶ Una instrucción se ejecuta en $5 \cdot N$ ciclos de reloj
 - ▶ Se ejecuta $1/5$ de instrucción cada N ciclos de reloj

Segmentación de instrucciones con pipeline



- ▶ Si cada fase dura N ciclos de reloj, entonces
 - ▶ Una instrucción se ejecuta en $5 \cdot N$ ciclos de reloj
 - ▶ Cada N ciclos de reloj termina 1 de instrucción

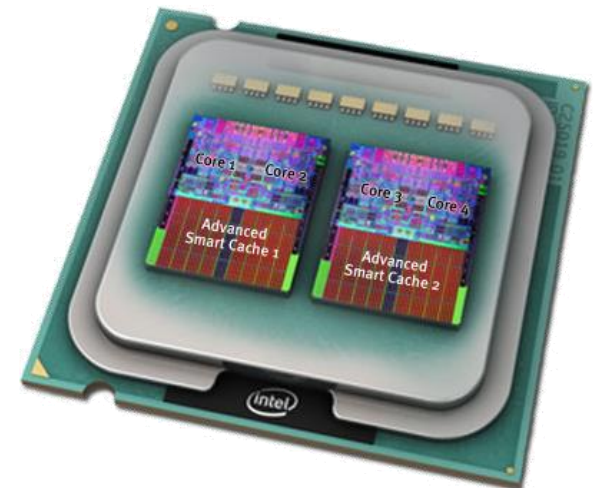
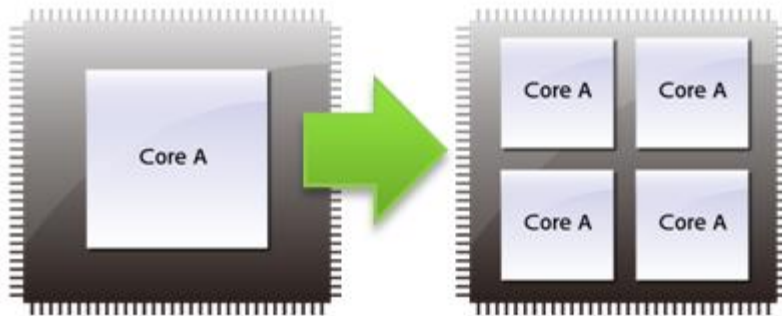
Superescalar



- Pipeline con varias unidades funcionales en paralelo

Multicore

- Múltiples procesadores en el mismo encapsulado



Multicore

- Múltiples procesadores en el mismo encapsulado



3 Knights Landing Products

A Paradigm Shift for Highly-Parallel

	Intel® 64 / AVX-512	Intel® 64 / AVX-512	Intel® 64 / AVX-512
Programming Model	PCIe	Fabric	Integrated Fabric
I/O	Baseline	>25% Better¹	>25% Better¹
Power Efficiency	Baseline	Intel server-class	Intel server-class
Resiliency	>3 TF¹	>3 TF¹	>3 TF¹
Performance	up to 16GB	up to 400GB¹	up to 400GB¹
Memory Capacity	>5x STREAM vs. DDR4¹	>5x STREAM vs. DDR4¹	>5x STREAM vs. DDR4¹
Memory Bandwidth			

<http://wccftech.com/intel-knights-landing-detailed-16-gb-highbandwidth-on-die-memory-384-gb-ddr4-system-memory-support-8-billion-transistors/>