

# Tema 4 (II) El procesador



Grupo ARCOS

Estructura de Computadores  
Grado en Ingeniería Informática  
Universidad Carlos III de Madrid

# Contenidos

---

## 1. Técnicas de control

1. Lógica cableada
2. Lógica almacenada

## 2. Otros aspectos de funcionamiento

1. Interrupciones y excepciones
2. Estado del procesador
3. Niveles de ejecución
  - ▶ Modo privilegiado/usuario
4. Arranque del computador
5. Paralelismo

# ¡ATENCIÓN!

---

- ❑ Estas transparencias son un guión para la clase
- ❑ Los libros dados en la bibliografía junto con lo explicado en clase representa el material de estudio para el temario de la asignatura
- ❑ Para la preparación de los exámenes se ha de utilizar todo el material de estudios

# Contenidos

---

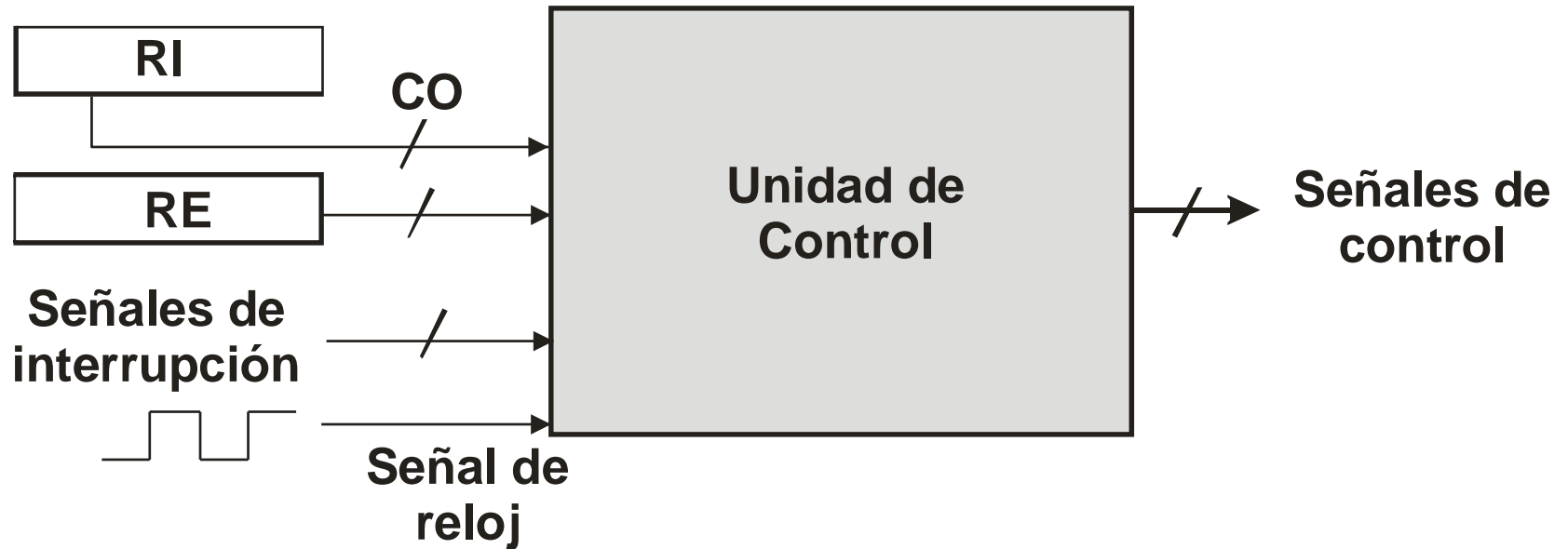
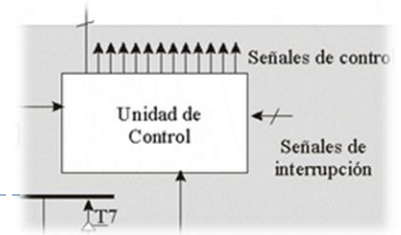
## 1. **Técnicas de control**

1. **Lógica cableada**
2. **Lógica almacenada**

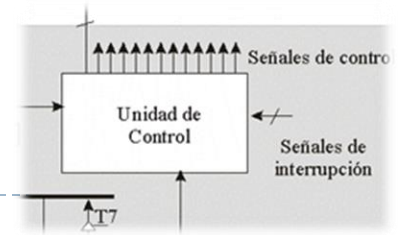
## 2. **Otros aspectos de funcionamiento**

1. **Interrupciones y excepciones**
2. **Estado del procesador**
3. **Niveles de ejecución**
  - ▶ **Modo privilegiado/usuario**
4. **Arranque del computador**
5. **Paralelismo**

# Unidad de control



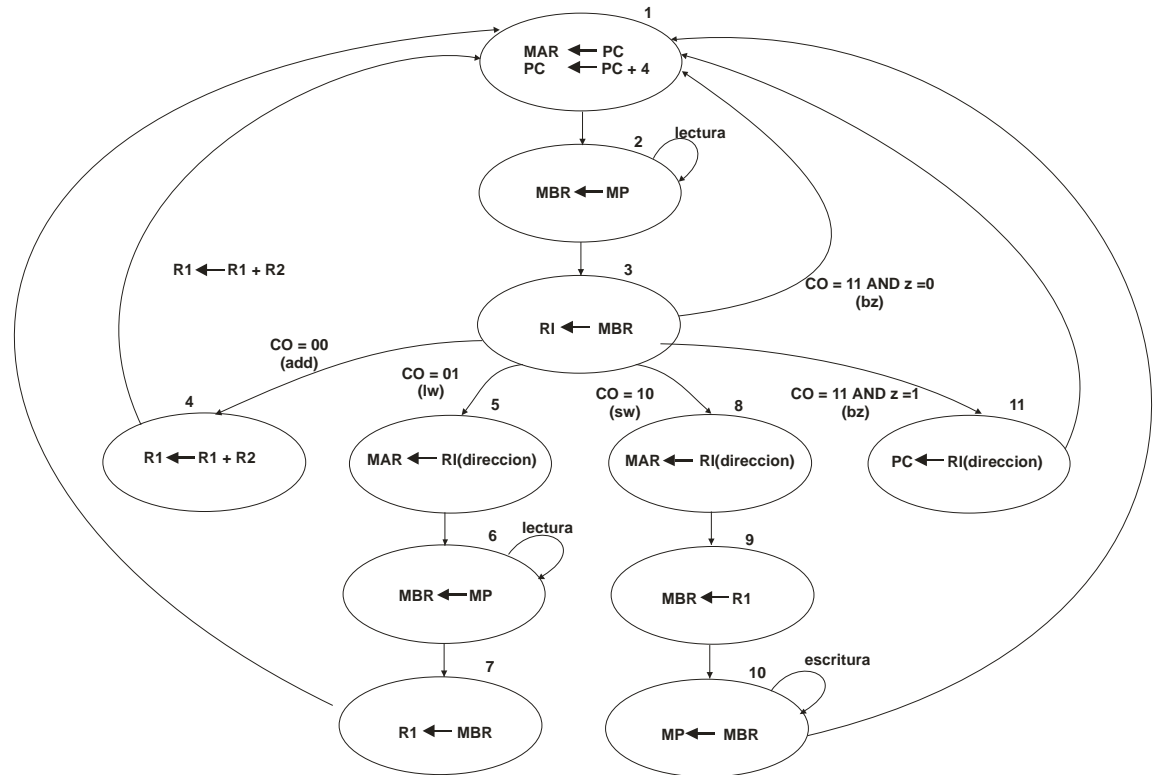
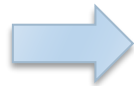
# Técnicas de control



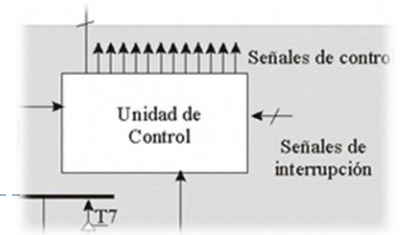
- ▶ El **diseño de la unidad de control** exige haber definido previamente las señales que hay que activar en cada una de las instrucciones máquina que es capaz de interpretar:

- ▶ Cronogramas.
- ▶ Operaciones elementales.
- ▶ Lenguaje simbólico.

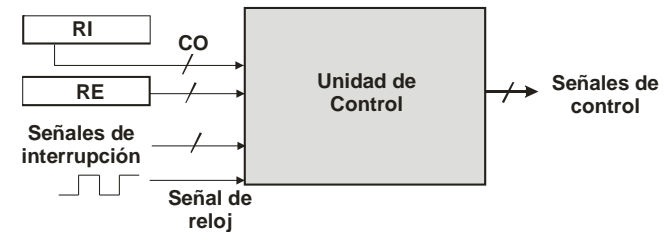
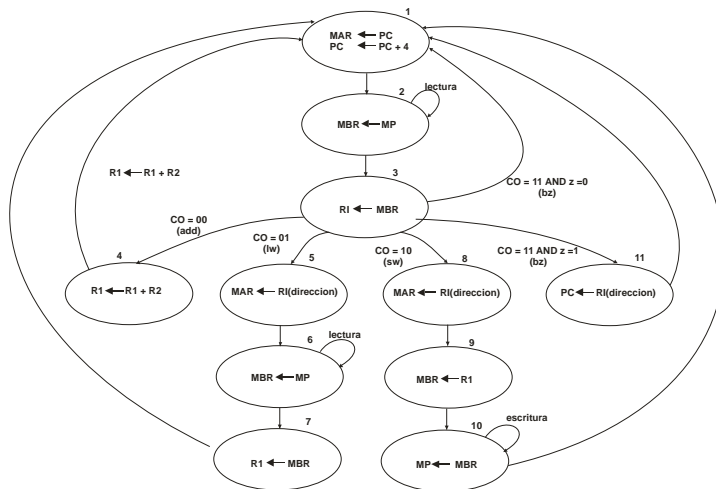
- `add r1, r2`
- `lw r1, dir`
- `bz dir`
- `sw r1, dir`



# Técnicas de control

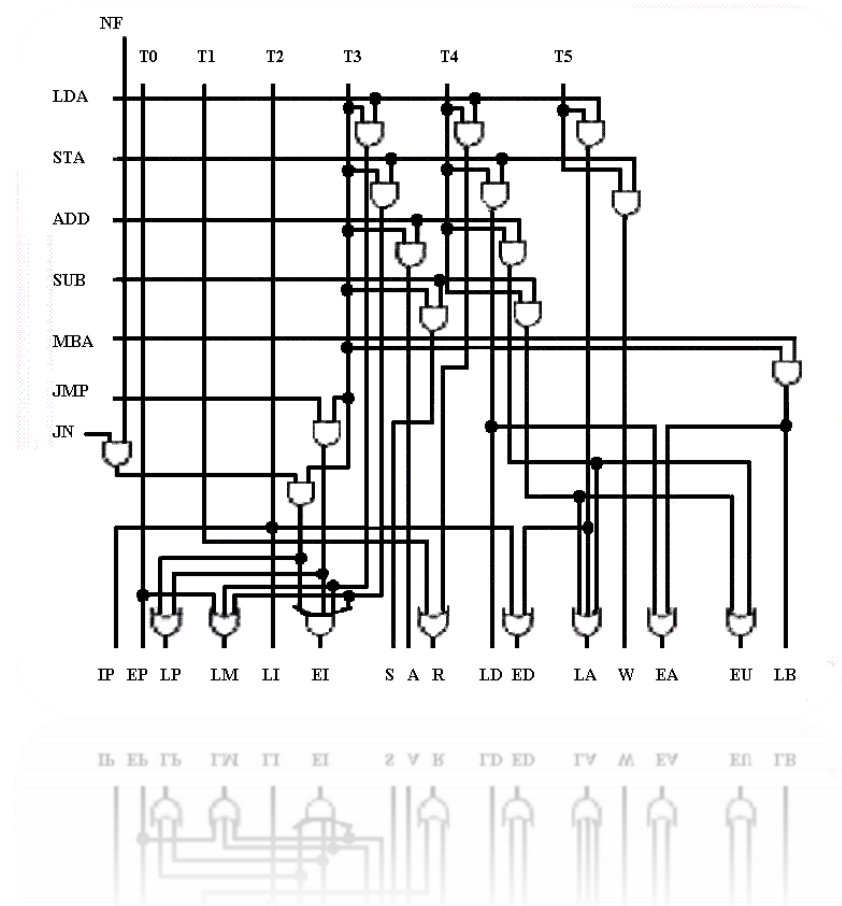


- ▶ **Dos técnicas de diseñar y construir una unidad de control:**
  - ▶ **Lógica cableada**
  - ▶ **Lógica almacenada (microprogramación)**



# A) Unidad de control cableada

- ▶ Construcción mediante puertas lógicas, siguiendo los métodos de diseño lógico.
- ▶ Características:
  - ▶ Laborioso y costoso el diseño y puesta a punto del circuito
  - ▶ Difícil de modificar:
    - ▶ rediseño completo.
  - ▶ Muy rápida (usado en computadores RISC)





# B) Unidad de control almacenada. Microprogramación

## ▶ Idea básica:

Emplear una memoria donde almacenar las señales de cada período de cada instrucción.

## ▶ Características:

### ▶ Fácil modificación

- ▶ Actualización, ampliación, etc..
- ▶ Ej.: Ciertas consolas, *routers*, etc.

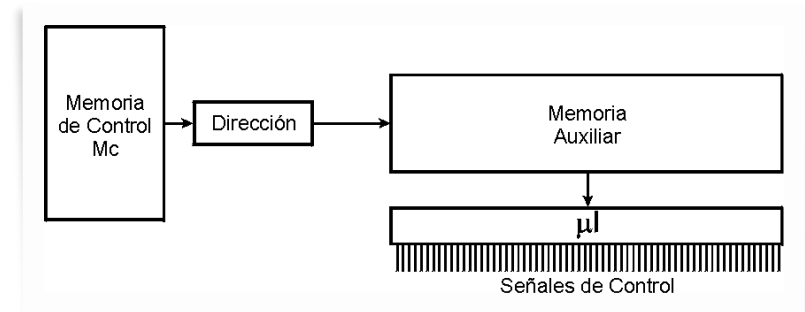
### ▶ Fácil tener instrucciones complejas

- ▶ Ej.: Rutinas de diagnóstico, etc.

### ▶ Fácil tener varios juegos de instrucciones

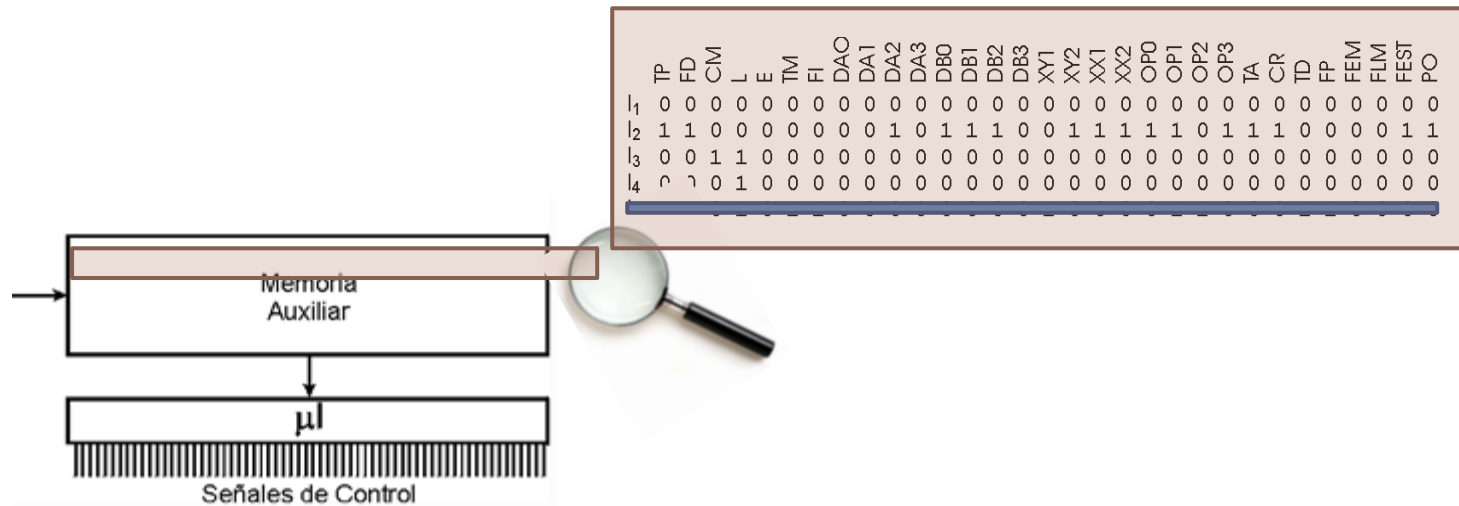
- ▶ Se pueden emular otros computadores.

### ▶ Hw simple $\Rightarrow$ difícil microcódigo



# Unidad de control almacenada.

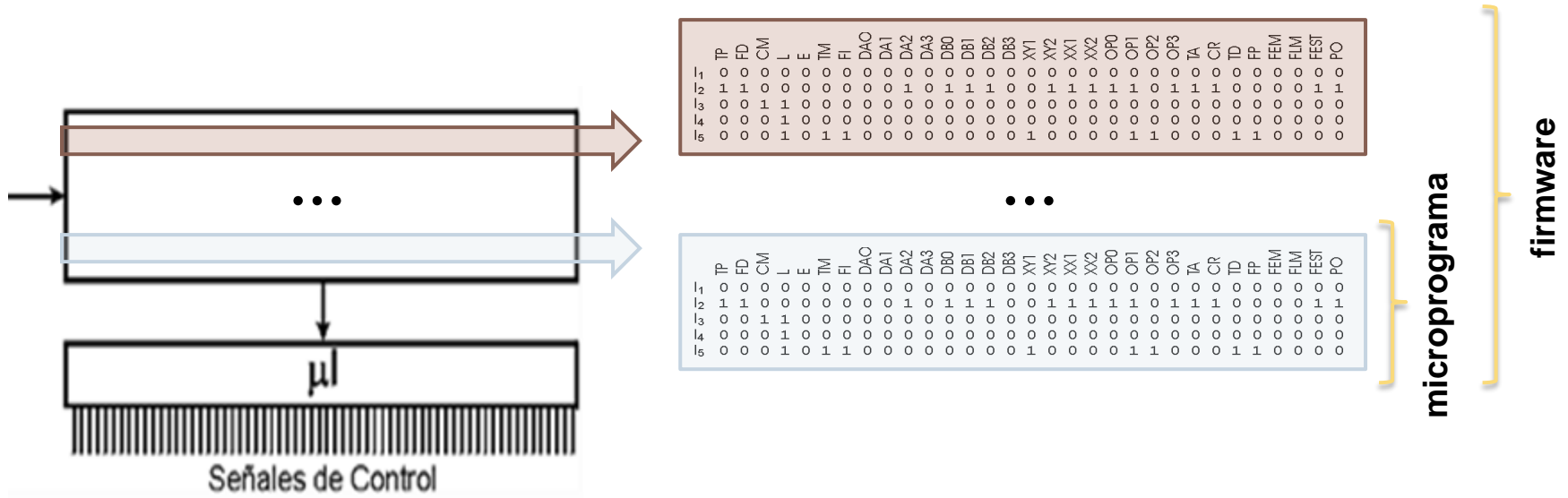
## Microprogramación (1)



- ▶ **Microinstrucción**: A cada palabra que define un período de una instrucción
- ▶ Las **microinstrucciones**
  - ▶ tienen un bit por cada señal de control.
  - ▶ cadena de 1's y 0's que representa el estado de cada señal de control durante un período de una instrucción.

# Unidad de control almacenada.

## Microprogramación (2)



- ▶ **Microprograma:** conjunto ordenado de microinstrucciones, que representan el cronograma de una instrucción.
- ▶ **Firmware** (microcódigo): conjunto de los microprogramas de una máquina.

# Unidad de control almacenada. Microprogramación (3)

Contenido M.Control



- ▶ **RESET: valores por defecto**
  - ▶ **PC <- 0x00, SP <- 0xFF, ...**
- ▶ **FETCH: traer sig. instrucc.**
  - ▶ **I <- Mem[PC], PC++, salto-a-C.O.**
- ▶ **Microprograma: uno por instrucción de ensamblador**
  - ▶ **Traer resto de operandos (si hay)**
    - ▶ actualizar PC en caso de más operandos
  - ▶ **Realizar la instrucción**
  - ▶ **Guardar los datos**
  - ▶ **Salto a FETCH**

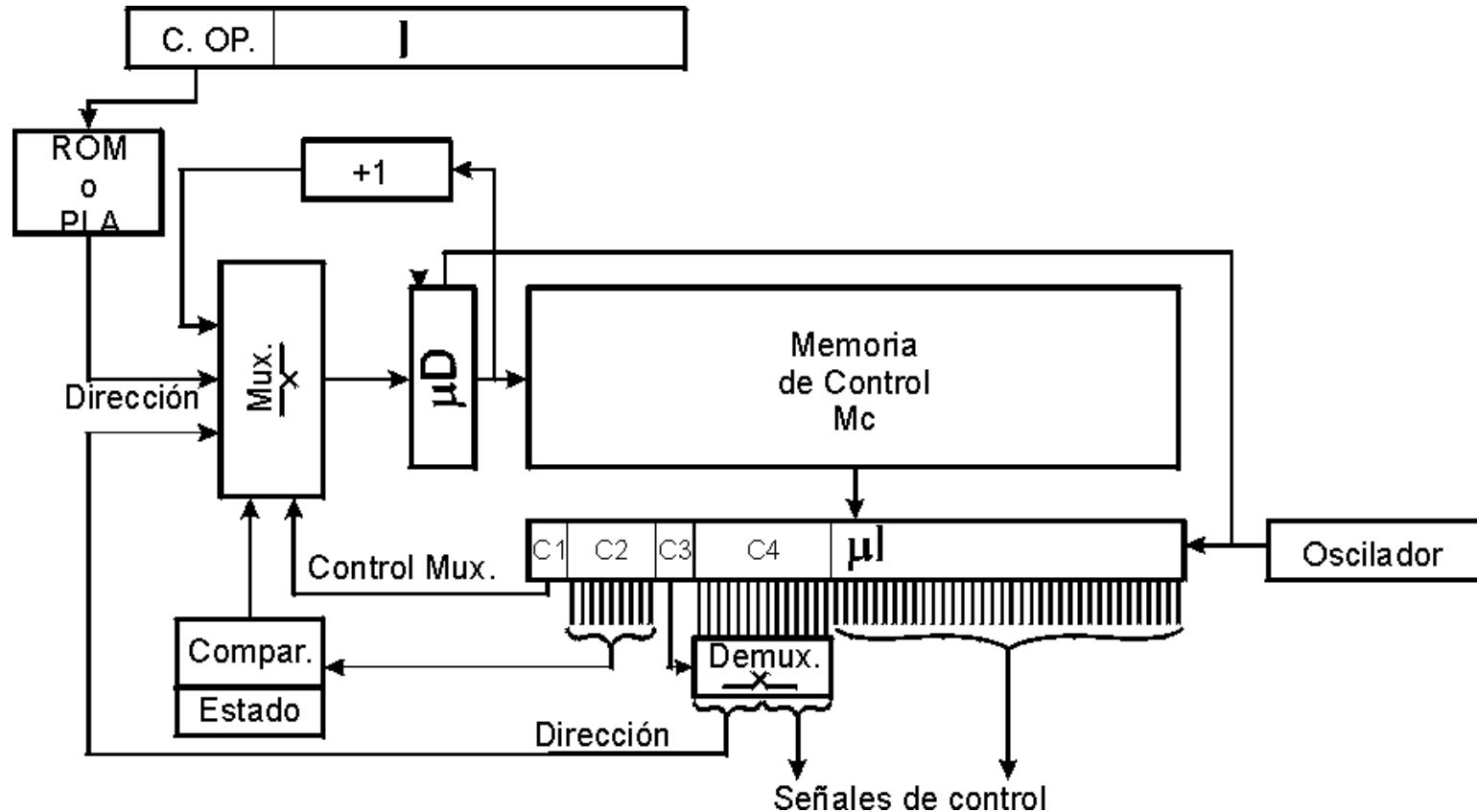
# Estructura de la unidad de control microprogramada (1)

---

## ► Tres condiciones básicas:

1. Memoria de control suficiente para almacenar todos los microprogramas correspondientes a todas las instrucciones.
2. Procedimiento para asociar a cada instrucción su microprograma
  - Procedimiento que convierta el código de operación de la instrucción en la dirección de la memoria de control donde empieza su microprograma.
3. Mecanismo para ir leyendo las sucesivas microinstrucciones, y para bifurcar a otro microprograma cuando termina el que se está ejecutando.

# Estructura de la unidad de control microprogramada (1)



- ▶ Ejemplo de estructura de unidad de control
- ▶ Permite microbifurcaciones condicionales, microbucles y microsubrutinas

# Formato de las microinstrucciones

---

- ▶ **Formato de la microinstrucción:**  
especifica el n° de bits y el significado de cada uno de ellos.



- ▶ Las señales se agrupan por **campos**:
  - ▶ Señales triestado de acceso a bus
  - ▶ Señales de gobierno de la ALU
  - ▶ Señales de gobierno del banco de registros
  - ▶ Señales de gobierno de la memoria

# Contenidos

---

## 1. Técnicas de control

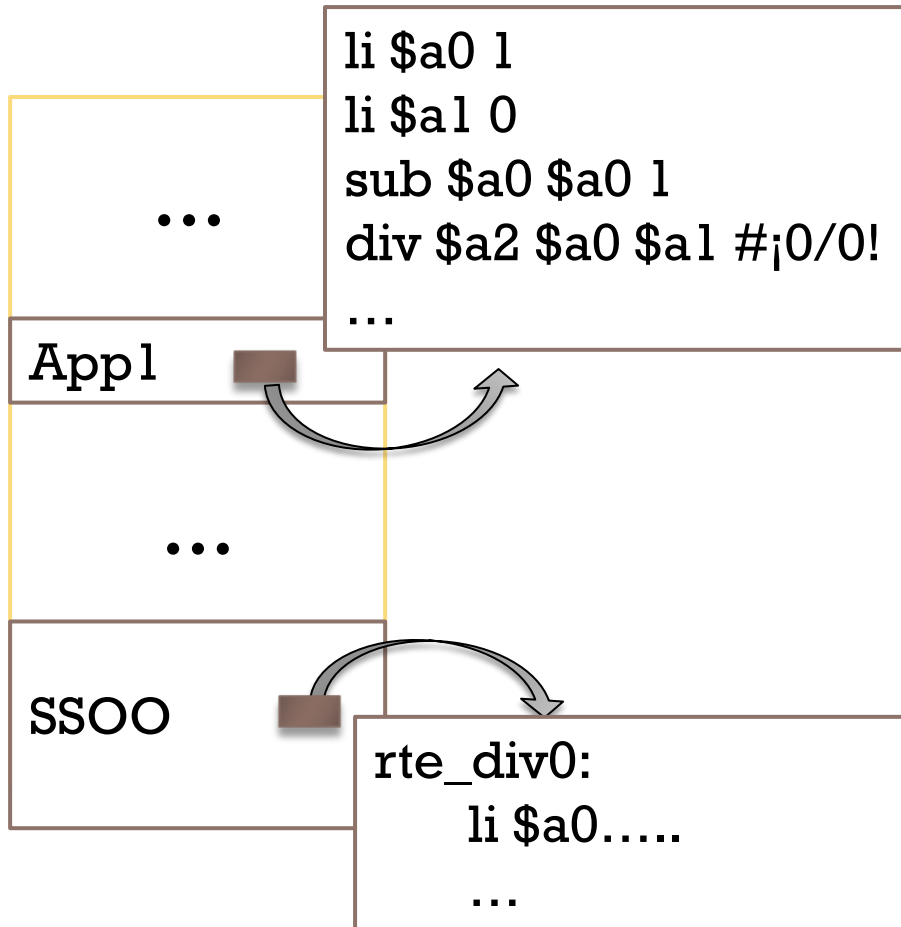
1. Lógica cableada
2. Lógica almacenada

## 2. Otros aspectos de funcionamiento

1. Interrupciones y excepciones
2. Estado del procesador
3. Niveles de ejecución
  - ▶ Modo privilegiado/usuario
4. Arranque del computador
5. Paralelismo

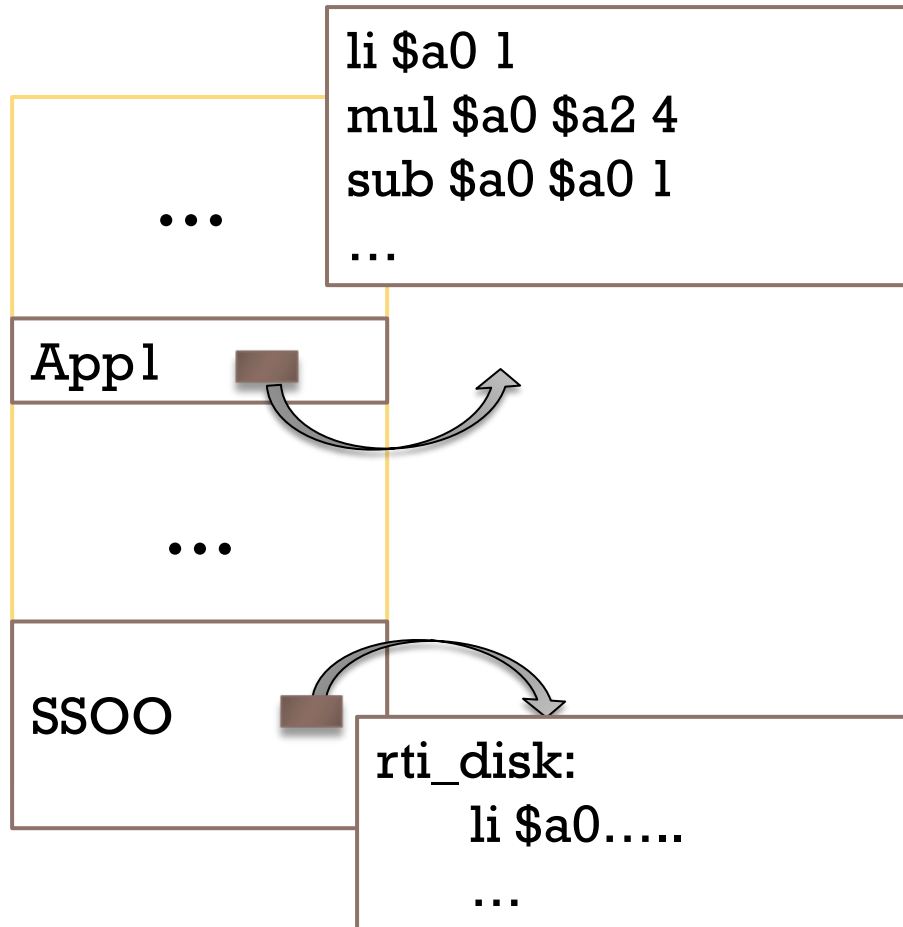


# Excepciones



- ▶ 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.

# Interrupción



- ▶ Originan una ruptura de secuencia no programada
  - ▶ **Al final microprograma de la instrucción en curso 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.

# Estado del procesador

---

- ▶ Las interrupciones rompen la secuencia normal de ejecución del programa.
- ▶ Una vez tratada la interrupción, el programa interrumpido debe seguir ejecutándose.
- ▶ **Estado del procesador:** contenido de los elementos de memoria internos necesarios para que un programa pueda seguir funcionando correctamente después de una interrupción.

# Elementos del estado del procesador

---

- ▶ PC
- ▶ Banco de registros
- ▶ Registros aritméticos auxiliares
- ▶ Biestables de estado aritméticos
- ▶ Biestables de estado de E/S
- ▶ Biestables o registros de modificación del mapa de memoria
- ▶ Registros de clave o de protección de memoria
- ▶ Biestables de control residual (modo usuario/privilegiado)

# Contenidos

---

## 1. Técnicas de control

1. Lógica cableada
2. Lógica almacenada

## 2. Otros aspectos de funcionamiento

1. Interrupciones y excepciones
2. Estado del procesador
3. **Niveles de ejecución**
  - ▶ **Modo privilegiado/usuario**
4. **Arranque del computador**
5. Paralelismo

# Niveles de ejecución.

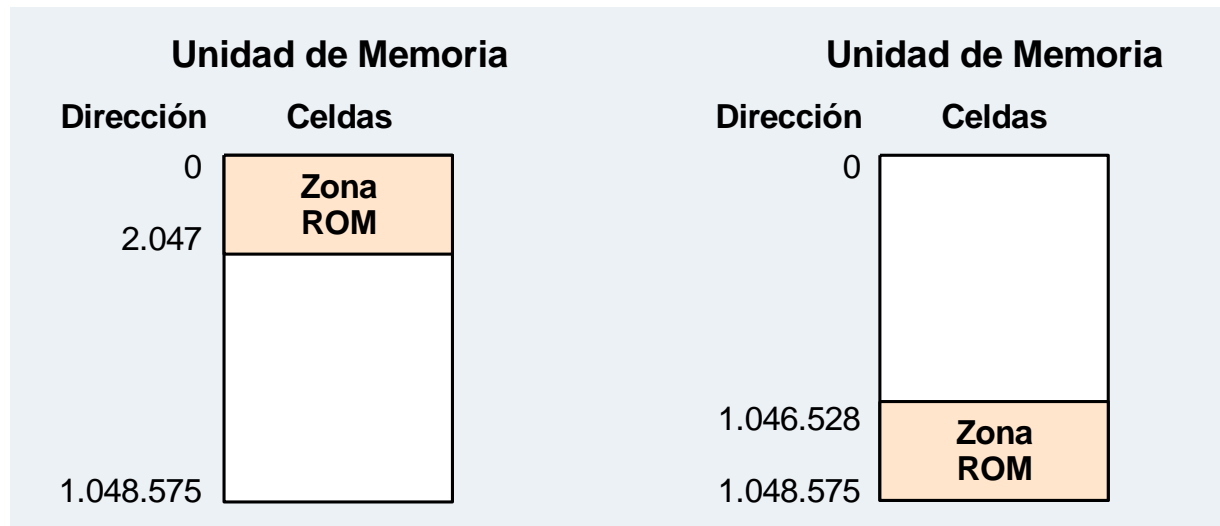
## Modo privilegiado/usuario

---

- ▶ En los computadores existen una serie de **instrucciones privilegiadas**, cuya ejecución debe reservarse al sistema operativo, y que el usuario normal no puede utilizar por razones de seguridad.
- ▶ Se añaden a la instrucción uno o varios bits (**bits de control residual**), procedentes de biestables, que distinguen entre:
  - Modo de ejecución privilegiado (S.O.)
  - Modo de ejecución de usuario
- ▶ El código de operación cambia su significado con los bits de control residual, impidiendo su ejecución si no se está en el nivel adecuado.

# Arranque del computador (1)

- ▶ El *Reset* carga valores predefinidos en registros
- ▶  $PC \leftarrow$  dirección de arranque del cargador ROM
- ▶ Se ejecuta el cargador ROM del sistema
  - Test del sistema
  - Trae a memoria el boot del SO
- ▶ La ROM del PC contiene además el soft. de E/S (BIOS)



# Arranque del computador (2)

---

- ▶ El arranque del SO carga el SO residente y da control al programa de arranque del SO, que:
  - ▶ Comprueba el hardware
  - ▶ Comprueba el sistema de ficheros
  - ▶ Establece las tablas del SO
  - ▶ Crea procesos (según el tipo de SO)
    - ▶ Proceso INIT
    - ▶ Procesos o tareas del sistema (demonios)
    - ▶ Procesos de *login* (uno por terminal)
  - ▶ Tras la autenticación, el proceso *login* se convierte en shell



# Contenidos

---

## 1. Técnicas de control

1. Lógica cableada
2. Lógica almacenada

## 2. **Otros aspectos de funcionamiento**

1. Interrupciones y excepciones
2. Estado del procesador
3. Niveles de ejecución
  - ▶ Modo privilegiado/usuario
4. Arranque del computador
5. **Paralelismo**

# 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
- ▶ superescalar
- ▶ multicore

# Segmentación de instrucciones

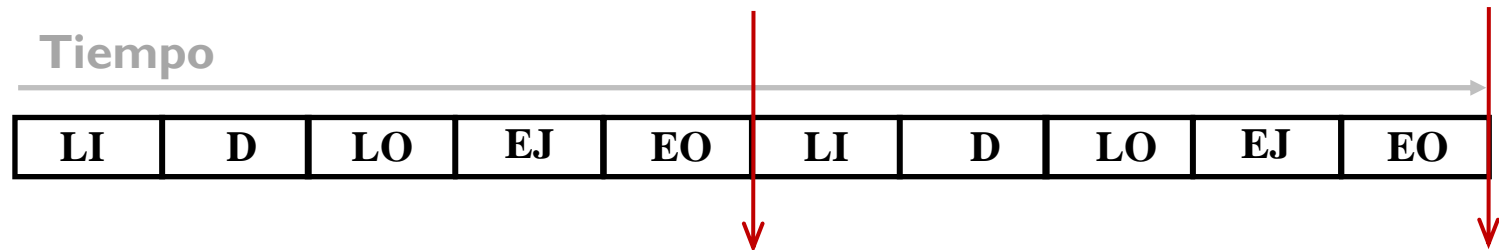


- ▶ Etapas de ejecución de una instrucción:
  - ▶ **LI**: Lectura de la instrucción e incremento del PC
  - ▶ **D**: Decodificación
  - ▶ **LO**: Lectura de Operandos
  - ▶ **Ej**: Ejecución de la instrucción
  - ▶ **EO**: Escritura de Operandos

# Segmentación de instrucciones

## sin pipeline

- ▶ segmentación
- ▶ superescalar
- ▶ multicore

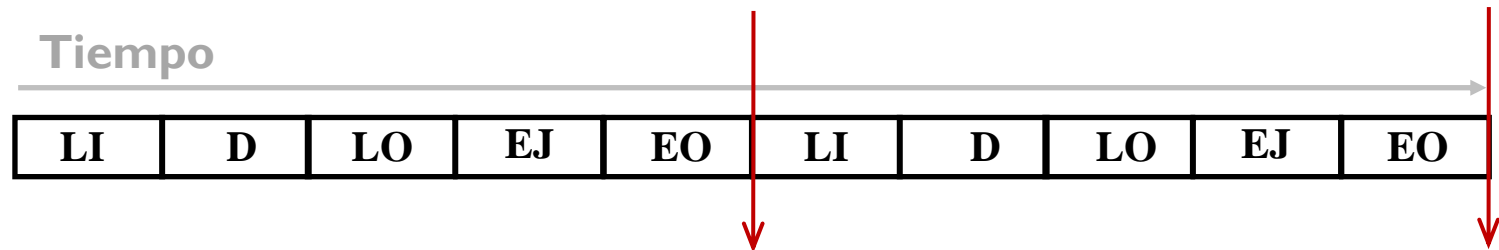


- ▶ Etapas de ejecución de una instrucción:
  - ▶ **LI**: Lectura de la instrucción e incremento del PC
  - ▶ **D**: Decodificación
  - ▶ **LO**: Lectura de Operandos
  - ▶ **EJ**: Ejecución de la instrucción
  - ▶ **EO**: Escritura de Operandos

# Segmentación de instrucciones

## sin pipeline

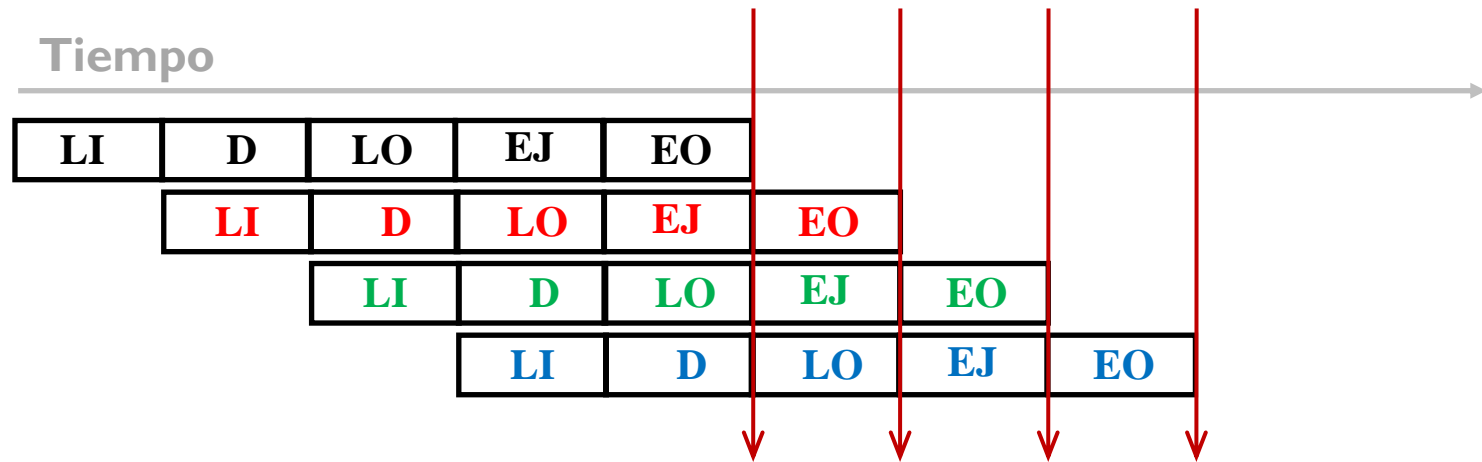
- ▶ segmentación
- ▶ superescalar
- ▶ multicore



- ▶ 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 se ejecuta  $1/5$  de instrucción

# Segmentación de instrucciones con pipeline

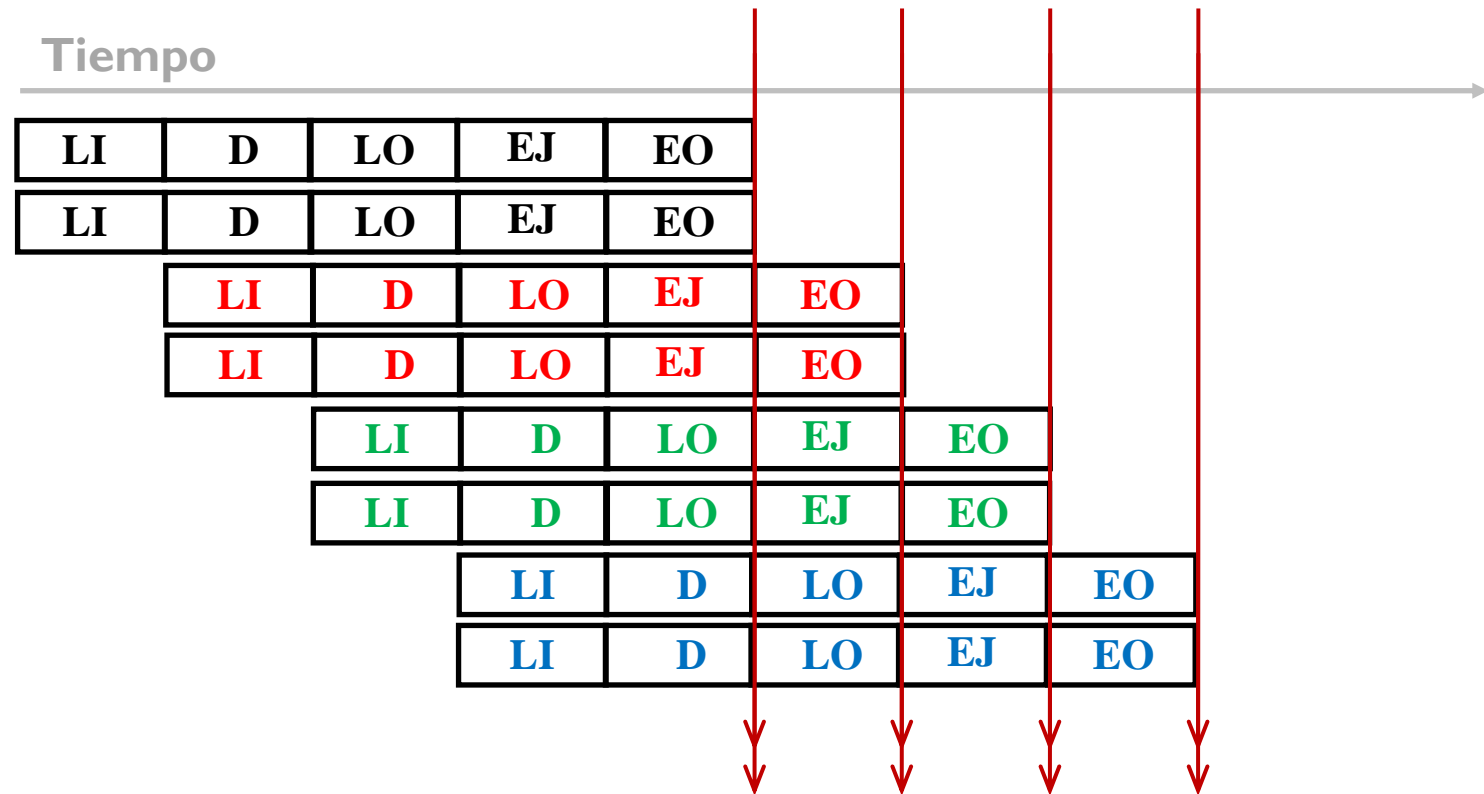
- ▶ segmentación
- ▶ superescalar
- ▶ multicore



- ▶ 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 se ejecuta 1 de instrucción

# Superescalar

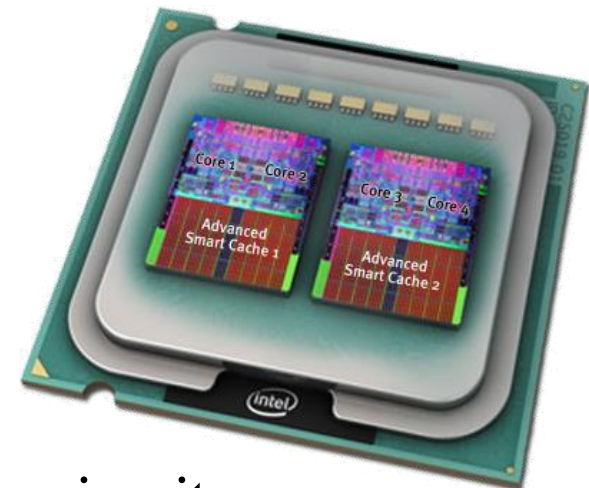
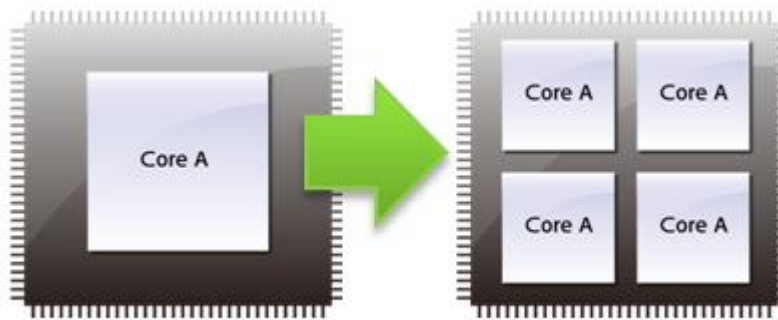
- segmentación
- **superescalar**
- multicore



- Pipeline con varias unidades funcionales en paralelo

# Multicore

- segmentación
- superescalar
- **multicore**



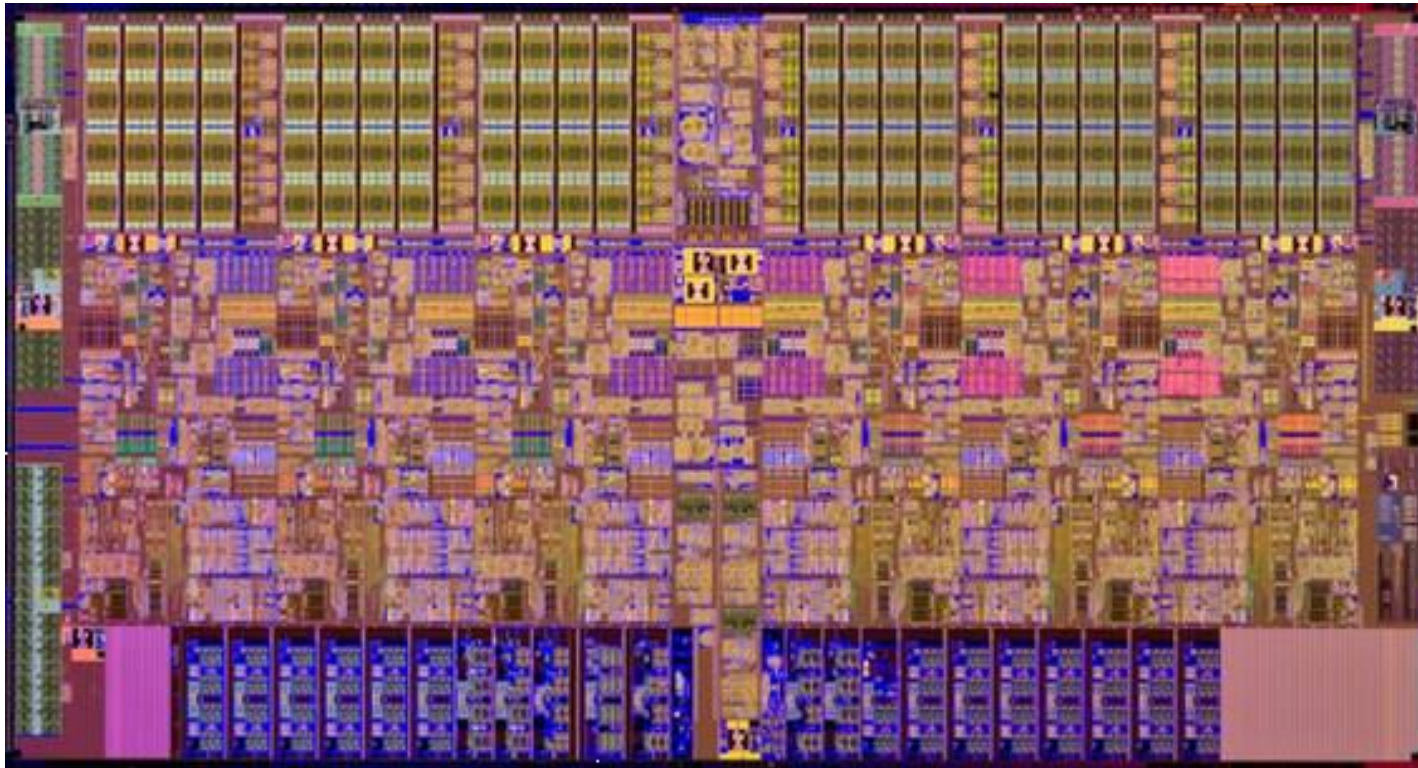
- Múltiples procesadores en el mismo circuito



# Multicore

---

- segmentación
- superescalar
- **multicore**

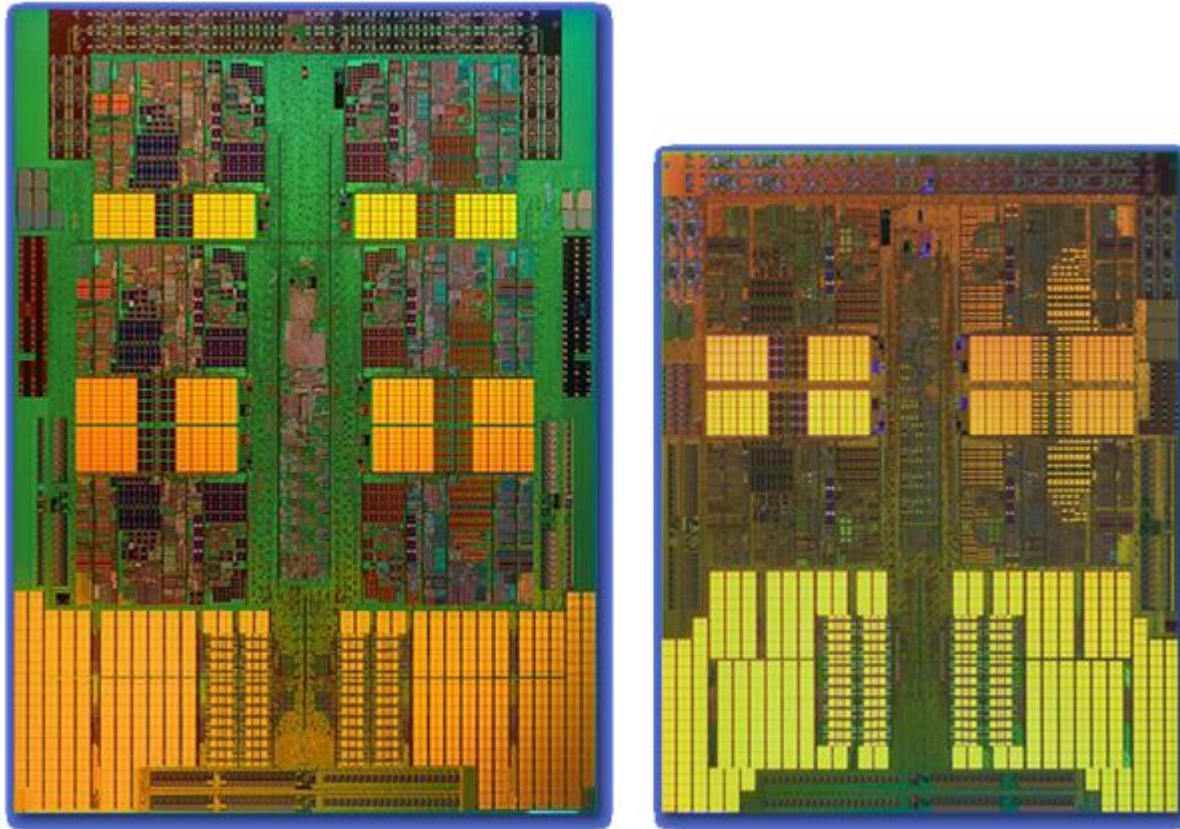


- Intel core i7-980x gulftown (6 cores)


# Multicore

---

- segmentación
- superescalar
- **multicore**



- AMD opteron Istanbul (6 cores) y shanghai (4 cores)



# Tema 4 (II) El procesador



Grupo ARCOS

Estructura de Computadores  
Grado en Ingeniería Informática  
Universidad Carlos III de Madrid