Estructura de Computadores

Tema 4

Procesador

Dpto. Arquitectura y Tecnología de Sistemas Informáticos.
Universidad Politécnica de Madrid

Índice

- Índice
  - Introducción
  - Operaciones elementales
  - Estructura de un computador elemental y sus señales de control
  - Cronogramas
  - Diseño de la Unidad de Control (UC)
  - Gestión de excepciones

Objetivos

- Visión **dinámica** del computador:
  - mostrar cómo se **ejecutan** las instrucciones
  - describir el órgano encargado de que esto se lleve a cabo (la **unidad de control, UC**)
  - entender esta ejecución de instrucciones mediante su representación en el tiempo por **cronograma**s
- Comprender el **diseño** de la UC
  - ser capaz de descomponer las instrucciones en **operaciones básicas** según la estructura del computador
  - diseño de la unidad de control **microprogramada**
- Comprender y analizar aspectos de diseño que pueden mejorar el **rendimiento** a través de mejoras en la estructura del computador o de la UC

Bibliografía

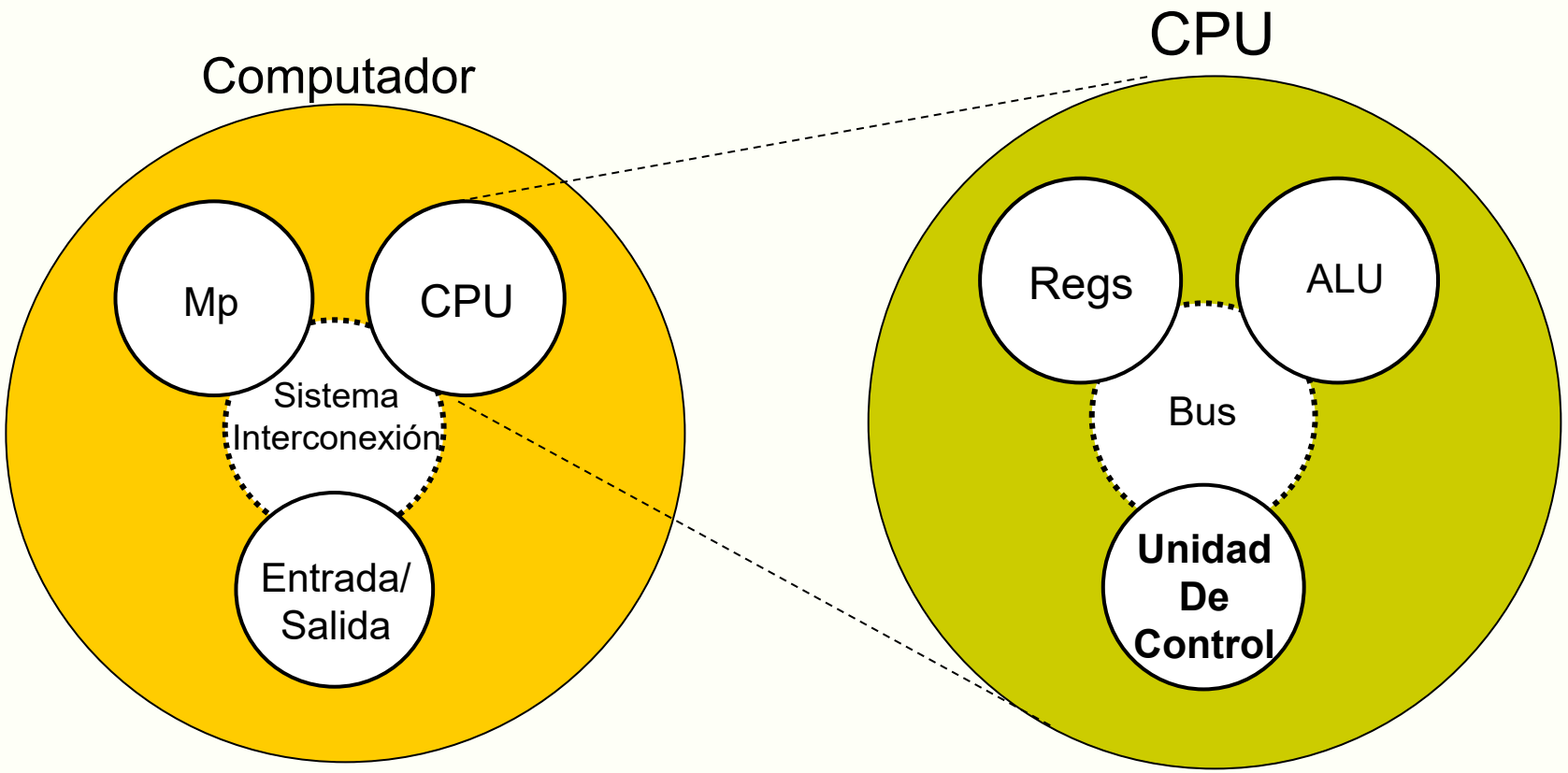
- de Miguel, P. *"Fundamentos de los computadores",* Paraninfo, 2004. 9ª Edición
- Patterson, D. A.; Hennessy, J. L. *Estructura y diseño de Computadores.* Ed. Reverté 2011. 4ª Edición
- Stallings, W. *"Organización y arquitectura de computadores",* Prentice Hall, 2006, 7ª Edición (hay 8ª ed., 2010, sólo en inglés)

Introducción

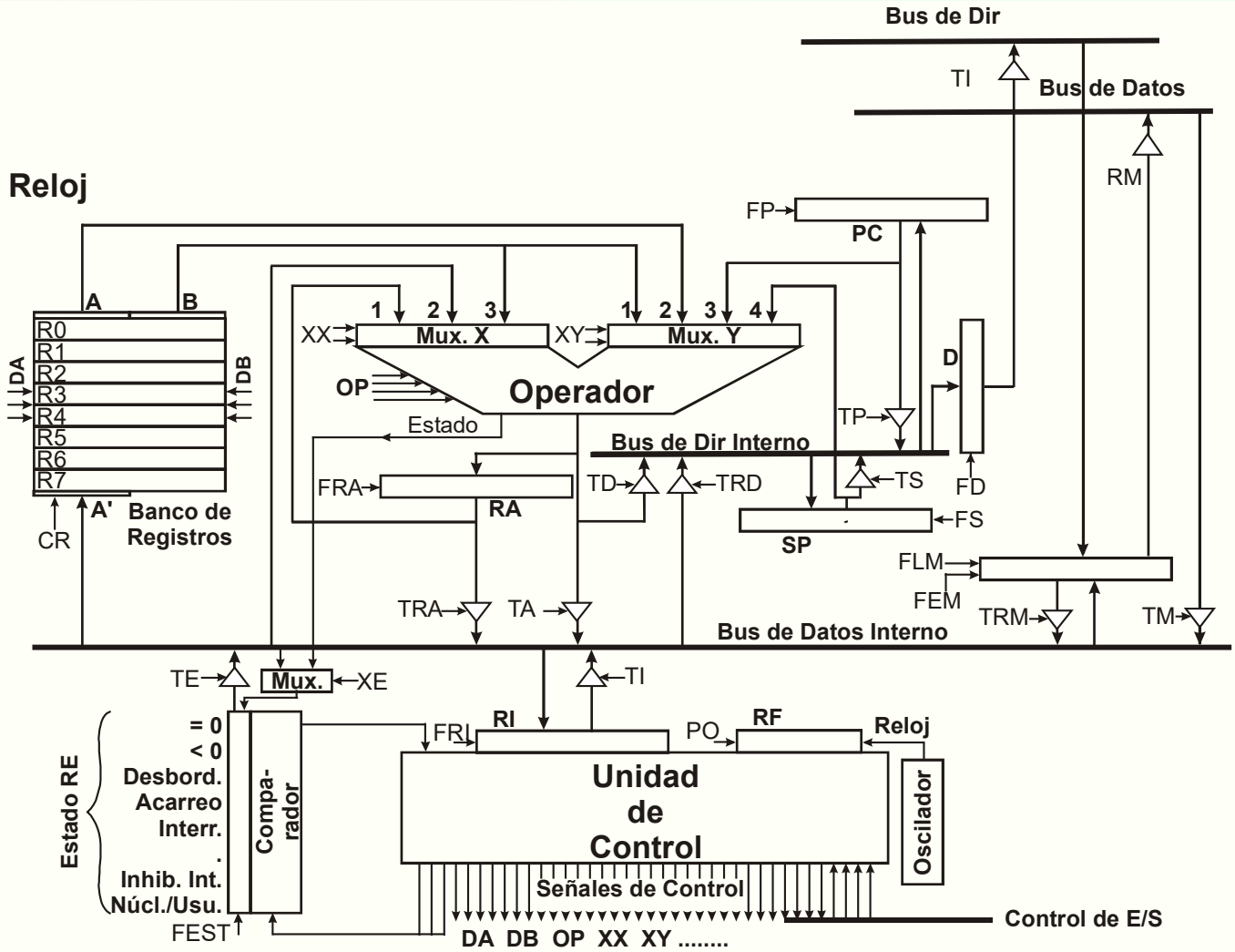
- Estudiaremos:
  - **Unidad de control**:
    - Encargada de interpretar las instrucciones del programa y gobernar la ejecución de las mismas
  - **Camino de datos (o *datapath)***:
    - Elementos internos de la CPU donde se transfieren los datos procedentes de la memoria o registros internos para obtener los resultados
    - Debe soportar el conjunto de operaciones que precisa el repertorio de instrucciones
- Organización de procesadores: ha evolucionado
  - desarrollo tecnológico
  - la necesidad de obtener altas prestaciones

Introducción

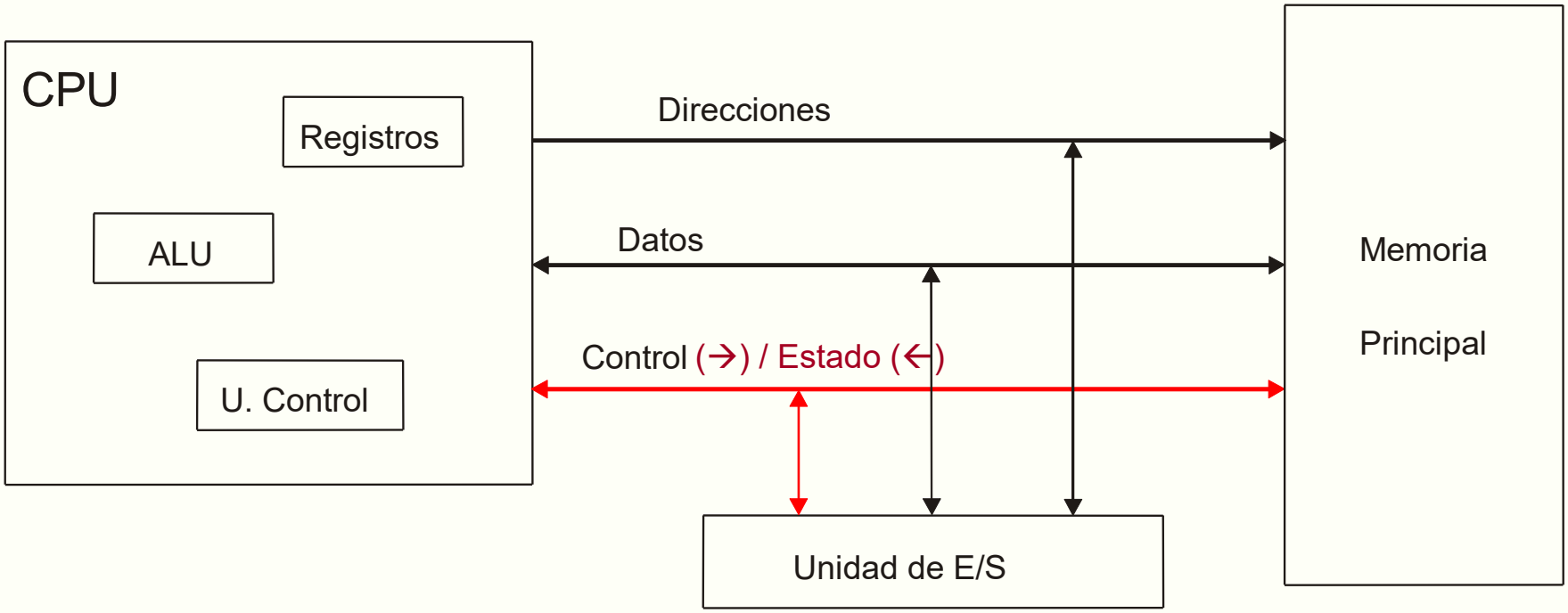
Visión global del computador y de la CPU



Estructura de Computador Elemental y sus señales de control



Esquema básico del computador Von Neumann



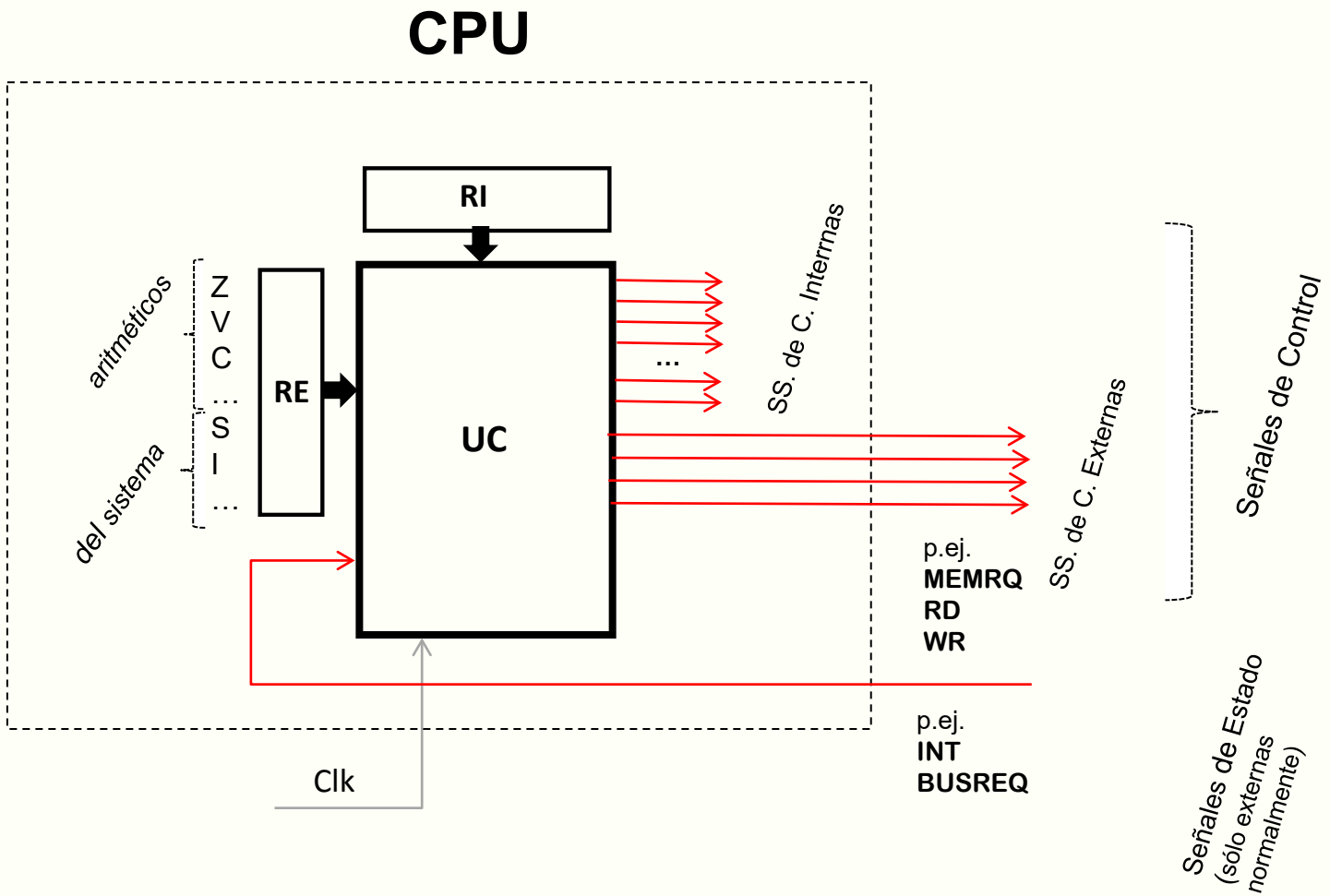
Introducción

- **Funciones de la CPU**:
  1. **Ejecuta** instrucciones (función básica)
     - Lectura (*fetch*) y decodificación de las instrucciones
     - Generación de órdenes para la ejecución
     - Secuenciamiento de las instrucciones:
       decidir cuál es la siguiente que se ha de ejecutar)
  2. **Gestiona** situaciones anómalas
     (desbordamiento, operación no válida, violación de privilegio, etc)
  3. Controla la **comunicación** con periféricos

Introducción

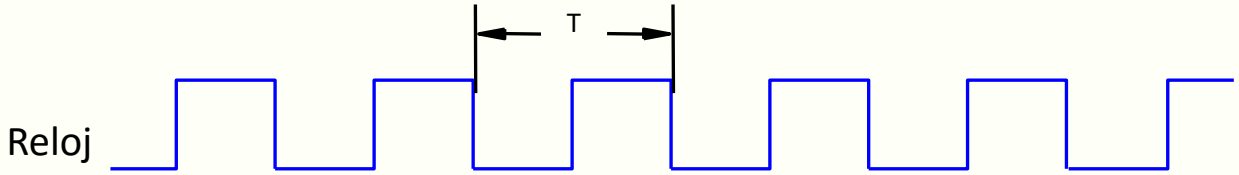
- Entradas y salidas de la UC:
  - **Entradas**
    - Registro de Instrucción (**RI o IR**): CO, MDs
    - **Reloj**: registro contador de fases
    - Registro de estado (**RE o SR**)
    - **Señales de estado** externas (de E/S y Mem)
  - **Salidas**
    - Todas las **señales de control** que permiten realizar cada una de las instrucciones máquina: (casi todas) internas y (algunas) externas (E/S y Mem)

Introducción



Introducción

- **Reloj**: tren de pulsos caracterizado por su periodo



  - ***señales de control***: siempre sincronizadas con el reloj
  - Define el tiempo de cada operación.
    - Memoria: tiempo de lectura y escritura
    - ALU: tiempo de operación
  - **CAMINO CRÍTICO**: camino de máximo retardo entre un origen y un destino. Depende de los dispositivos que tengan que atravesar las señales.

Introducción

Ciclo de lectura en M

1. dirección → **AR**
2. **M(AR)** → **Reg** (*), lectura (supone 3 ciclos)

(*) normalmente, el Reg. de Datos de la CPU, DR

Señales de Control:
carga en AR: **FAR**
lectura: **MEMRQ** y **RD**
carga en Reg o (DR): **FReg**

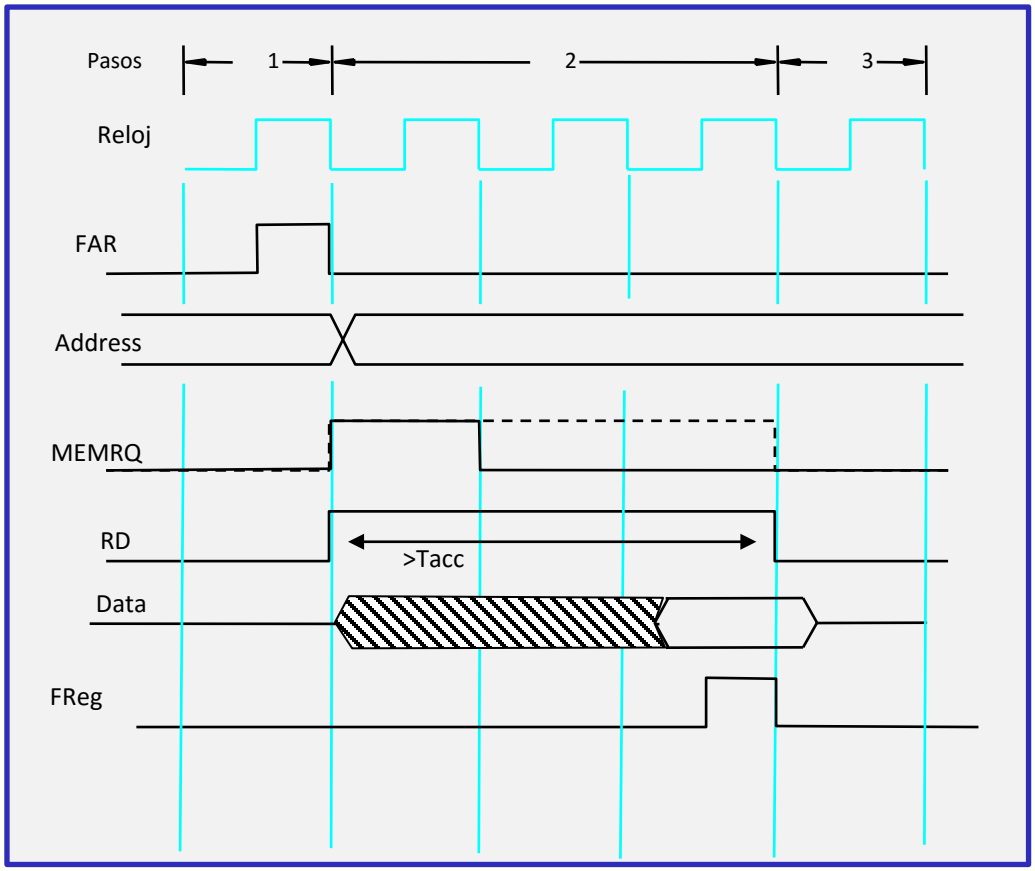


Figura . Temporización de la lectura en memoria

Introducción

Ciclo de escritura en M

1. dirección → **AR**

   dato → **DR** (**)
2. **DR** → **M(AR)**, escritura (supone 3 ciclos)

(**) supuesto que se pueda realizar simultáneamente la carga en ambos registros

Señales de Control:
- carga en AR: **FAR**
- carga en DR: **FDR**
- escritura: **MEMRQ** y **WR**

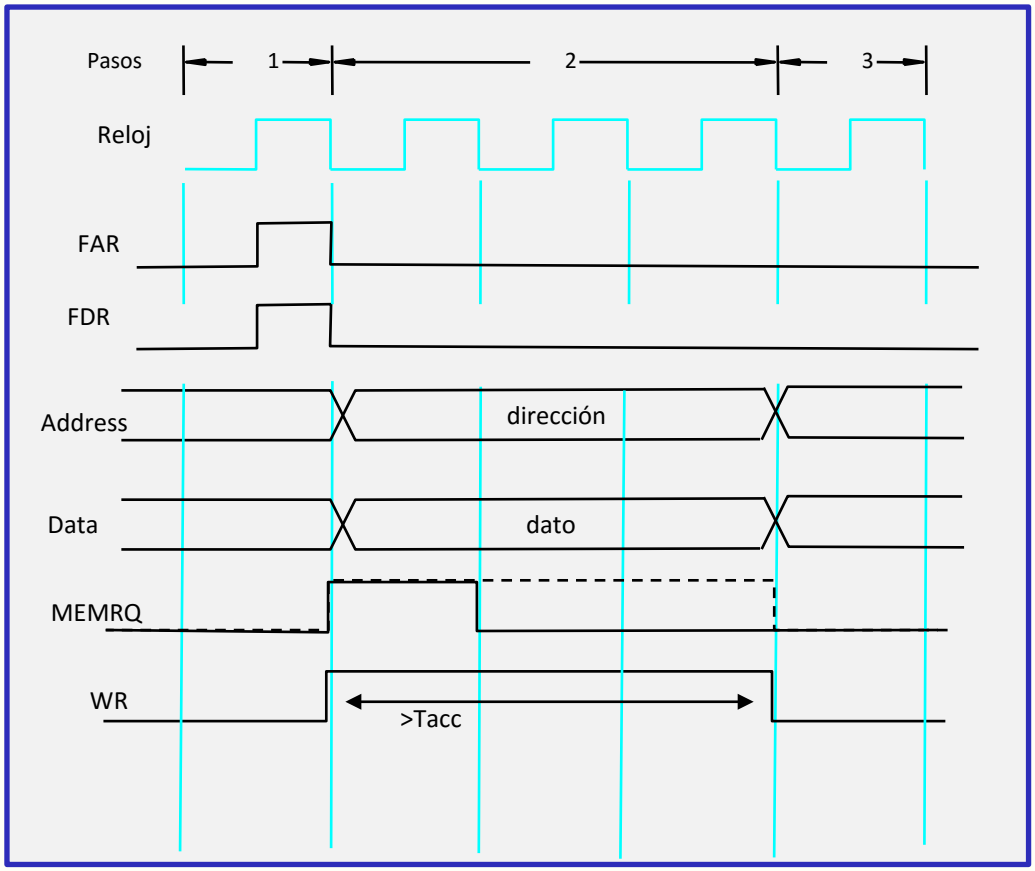


Figura . Temporización de la escritura en memoria

Introducción

Ciclos de lectura y escritura en **M** o en los **Módulos de E/S** (en sus registros): **ciclo de BUS**

- implica un acceso al **exterior** a la CPU
  - durante ese tiempo sólo la CPU puede acceder a los buses
  - (ningún periférico)
- suele durar más de 1 ciclo de reloj, p.ej., 3 ciclos en la figura

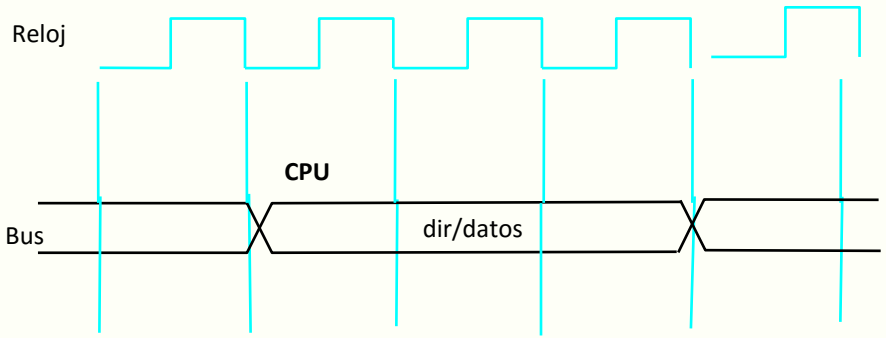


Figura . Ciclo de bus

Índice

- Índice
  - Introducción
  - Operaciones elementales
  - Estructura de un computador elemental y sus señales de control
  - Cronogramas
  - Diseño de la Unidad de Control (UC)
  - Gestión de excepciones

Operaciones Elementales

- **Ciclo de instrucción**: conjunto de acciones que requiere la ejecución de una instrucción
  1. Leer la instrucción en memoria (*fetch*)
  2. Decodificar la instrucción
  3. Ejecución: como máximo
     - Búsqueda de operandos, que puede conllevar
       - Cálculo de operando
       - Leer operando en memoria
     - Operación
     - Almacenar resultado
  4. Preparar la siguiente instrucción (ir a paso 1)

Operaciones Elementales

- Las **fases de ejecución** ayudan a simplificar el diseño de la UC.

  Ej: *fetch* común a todas las instrucciones

- Funcionamiento del computador durante la ejecución de un programa consiste en una **sucesión de ciclos de instrucción**

Operaciones Elementales

- **OPERACIONES ELEMENTALES (microoperaciones)**: operaciones **realizables directamente por el hardware** en que la UC divide cada una de las fases de ejecución de una instrucción: p.ej.

`SP → AR`

- Las operaciones elementales se realizan por la UC mediante la **activación de señales de control**
- Cada OE **dura un ciclo de reloj** (excepto las de M)
- Se pueden **simultanear** en el tiempo, cuidando:
  - Orden preestablecido
  - Conflictos en los elementos Hw

Operaciones Elementales

- Tipos:
  - **De transferencia**

    Llevan información de un origen a un destino
  - **De proceso**

    Llevan información de un origen a un destino, pero ésta pasa por un **operador**

Operaciones Elementales

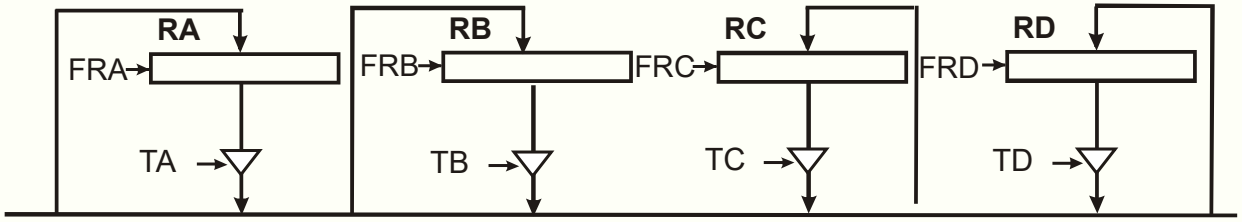
- **OE de transferencia**
  - Llevan información de un ORIGEN (reg. o M) a un DESTINO (reg. o M)
    1. Establecer **camino físico** entre salida de origen y entrada de destino (abrir triestados, multiplexores, etc)
    2. Activar la señal de carga en destino para que grabe el valor presente a su entrada.

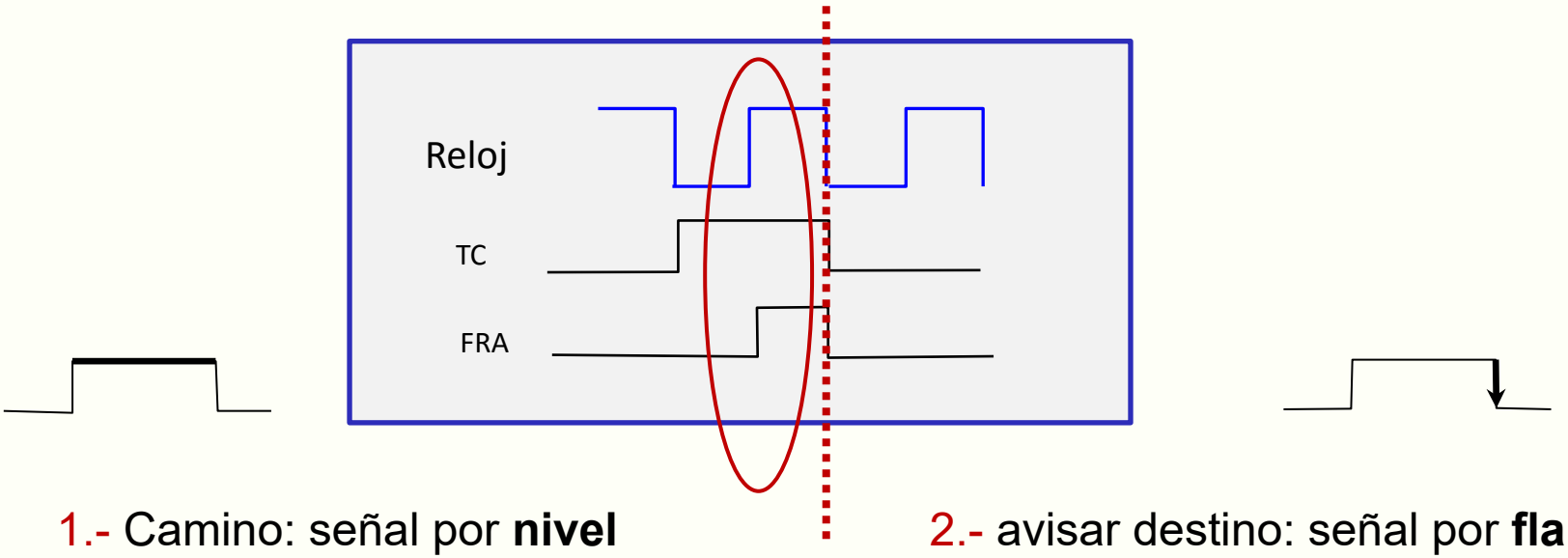
Esencial:

1) Los buses **no almacenan** información
2) En cada ciclo sólo puede haber un acceso abierto hacia un bus

Operaciones Elementales

Ej: Transferencia a través de un bus de dos registros: **RC → RA**



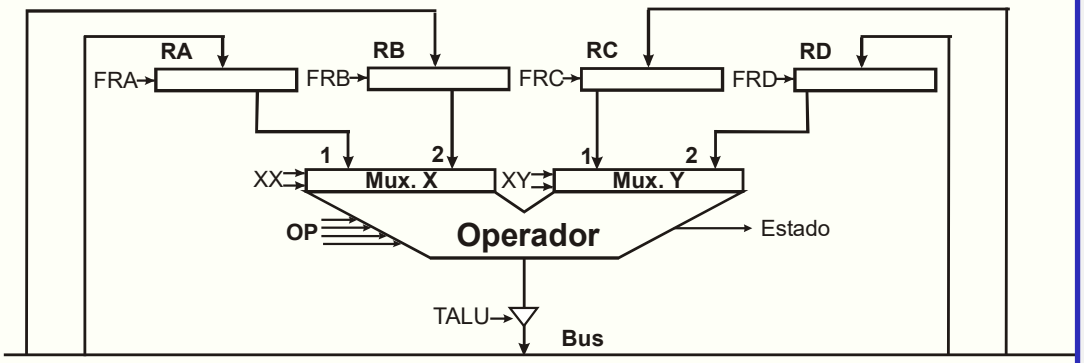


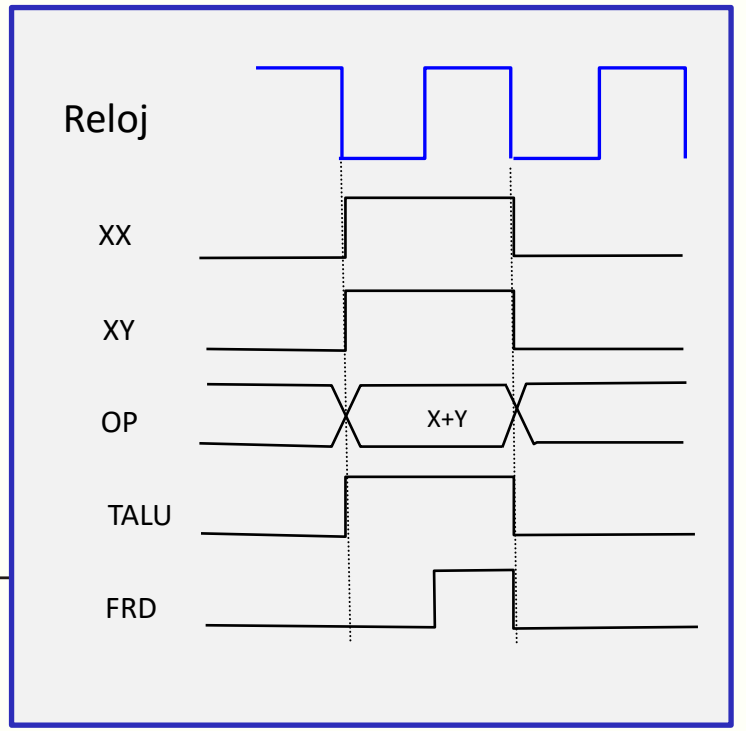
1.- Camino: señal por **nivel**

2.- avisar destino: señal por **flanco**

Operaciones Elementales

- **OE de proceso**
  - Funcionamiento similar a las transferencias, pero la información de ORIGEN se pasa por un operador (ALU) que la procesa en su camino al DESTINO
  - Ej: Operación en ALU con registros

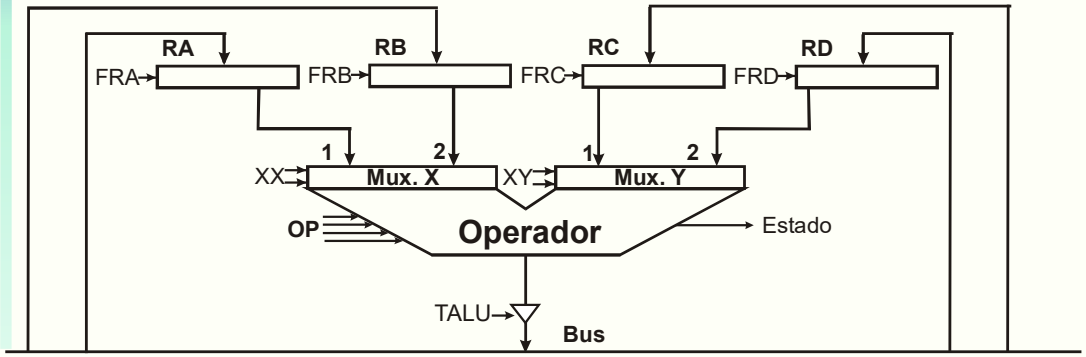


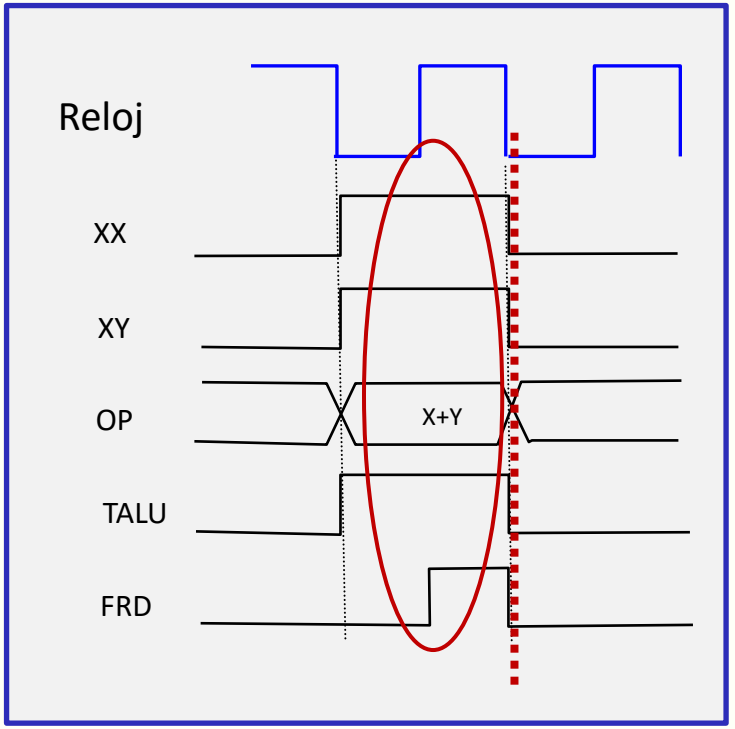


Operaciones Elementales

Ej: Operación en la ALU con registros:

RA + RC → RD





1.- Establecer camino:
señales por **nivel**
simultáneas

2.- avisar destino: señal por **flanco**

Operaciones Elementales

- **Reglas de agrupación de OEs**
  1. Respetar el **orden lógico** entre las acciones

     Ej: antes de leer, guardar dirección en AR
  2. **Evitar conflictos** en el mismo recurso físico

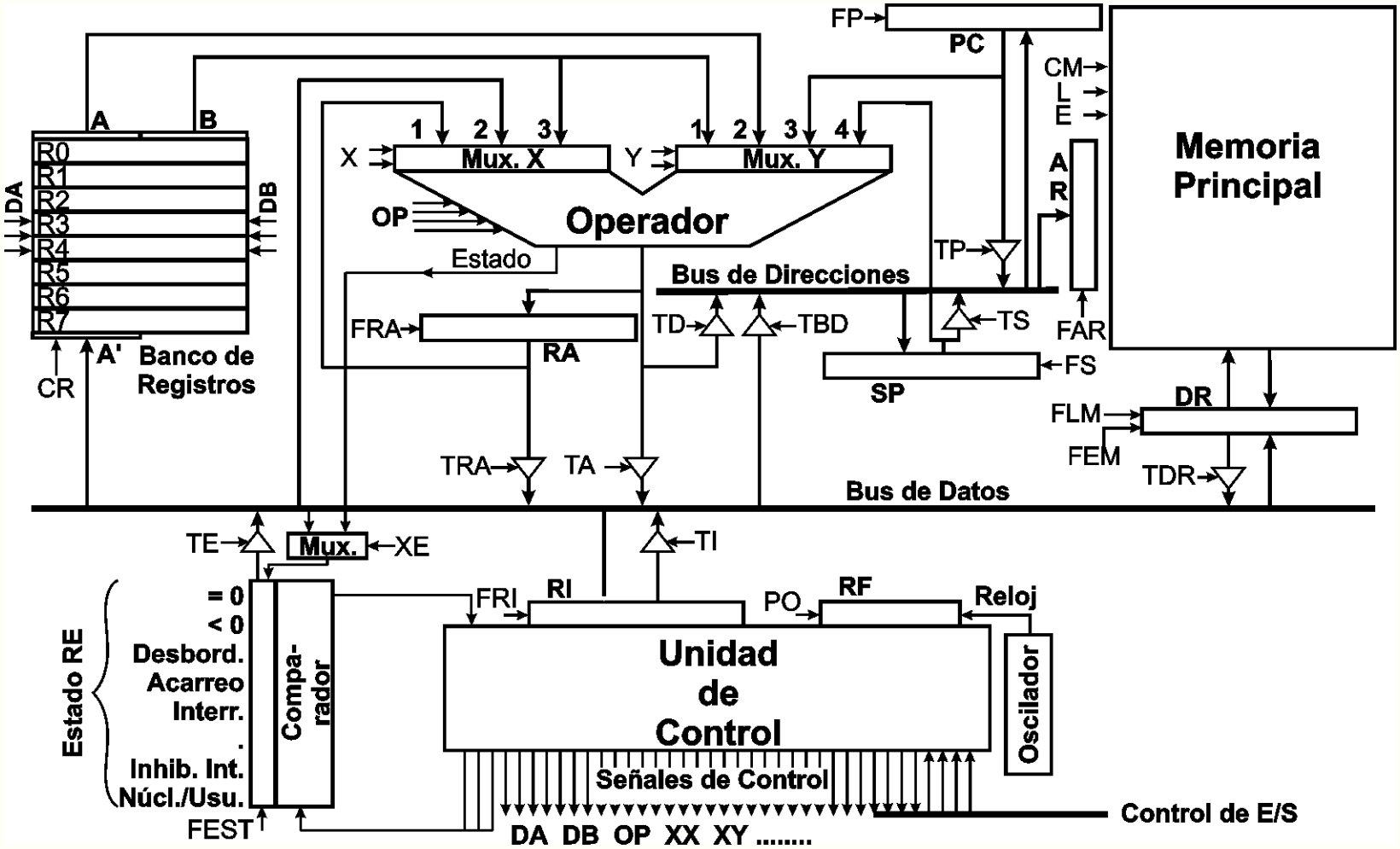
     un elemento físico no puede estar en dos estados diferentes al mismo tiempo

     Ej: conflicto en bus; memoria no puede leer y escribir al mismo tiempo, única operación en la ALU, etc.
  3. La información debe guardarse en algún sitio: ORIGEN y DESTINO deben poder **almacenar información (reg** o **mem)**

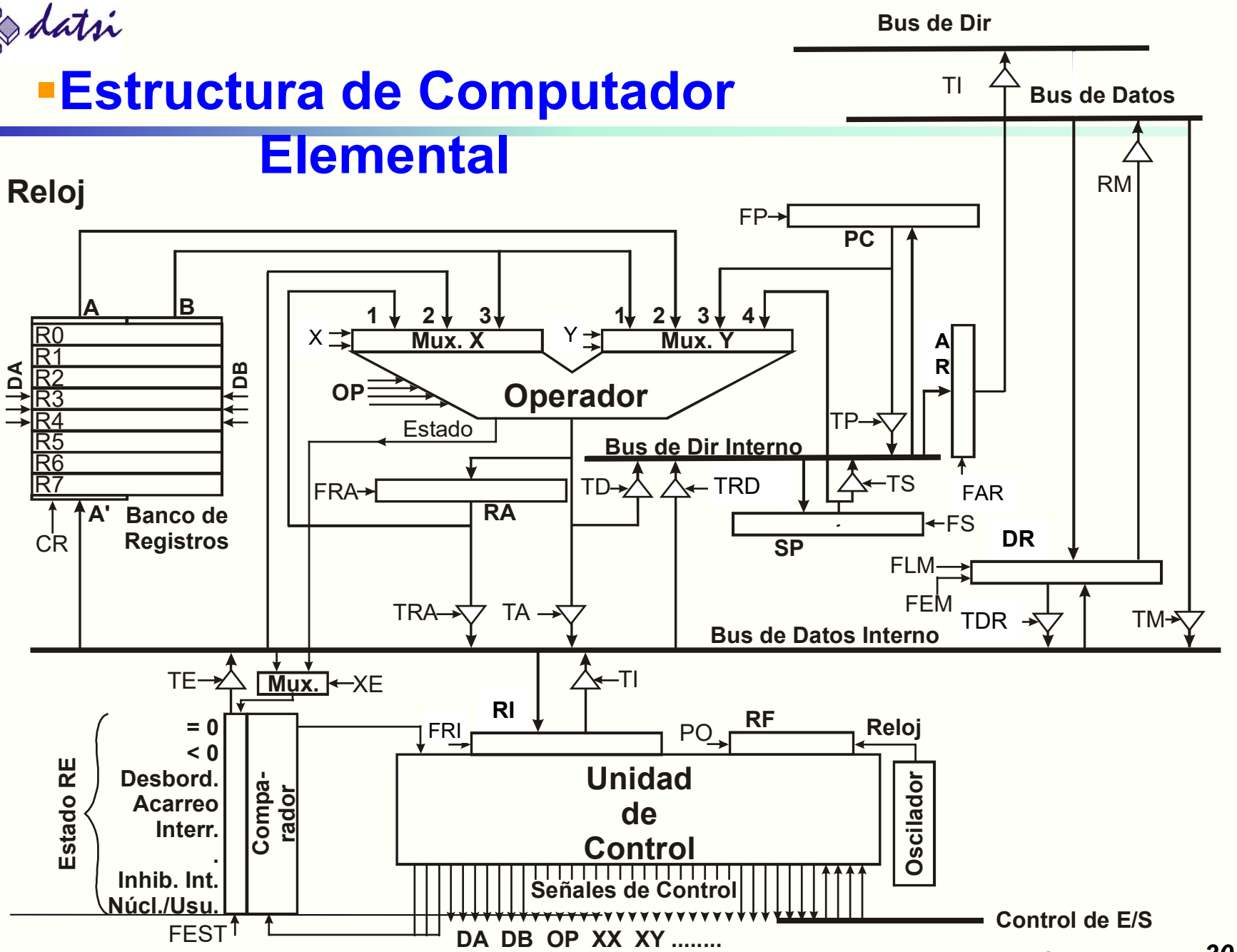
Índice

- Índice
  - Introducción
  - Operaciones elementales
  - Estructura de un computador elemental y sus señales de control
  - Cronogramas
  - Diseño de la Unidad de Control (UC)
  - Gestión de excepciones

Estructura de Computador Elemental
PDM v1.0

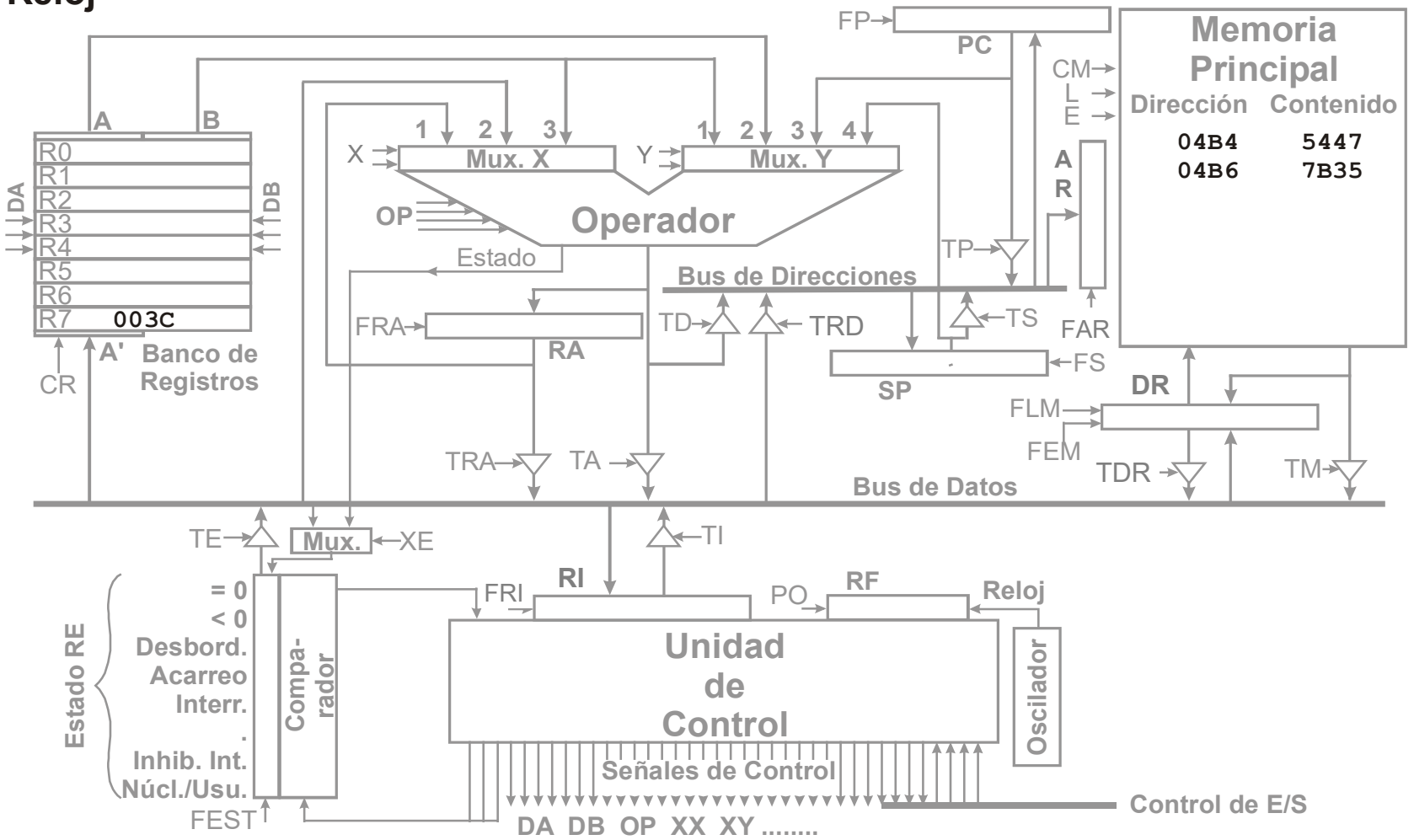


Estructura de Computador Elemental



ADD .R4,.R7

Reloj



Esta animación presenta la forma en que un computador sencillo ejecuta
la instrucción de suma de dos registros.

Se supondrá que la arquitectura es de 16 bits y que la memoria se direcciona a nivel de byte. Además, el acceso a la memoria principal requiere dos ciclos.

La instrucción se encuentra almacenada en la posición de memoria H’4B4, ocupa 16 bits y se compone del código de operación (H’54) seguido de los descriptores de los registros, (4 y 7).

El resultado se dejará en el registro 4.

Los contenidos de los registros son H’4F2 y H’3C, por lo que el resultado es H’52E. Además, el único bit de estado que se activa es de NZ, puesto que el resultado no es cero.

La animación contempla dos imágenes por ciclo de reloj. La primera corresponde a las señales de control de tipo nivel, mientras que la segunda refleja las señales de carga.

Para ver el detalle de un ciclo se recomienda la animación “element”, que muestra el desarrollo de una operación elemental.

Solamente se han indicado los valores de los registros afectados por la instrucción.

Se ha considerado que la lectura de la instrucción empieza en el ciclo 3 de la instrucción anterior y que el contador de ciclos se pone a “0” en el momento de la decodificación de la instrucción leída.

Las operaciones elementales necesarias son las siguientes:

D ← PC
I ← M(D); esta lectura tarda dos ciclos
Se decodifica la instrucción
R4 ← R4 + R7

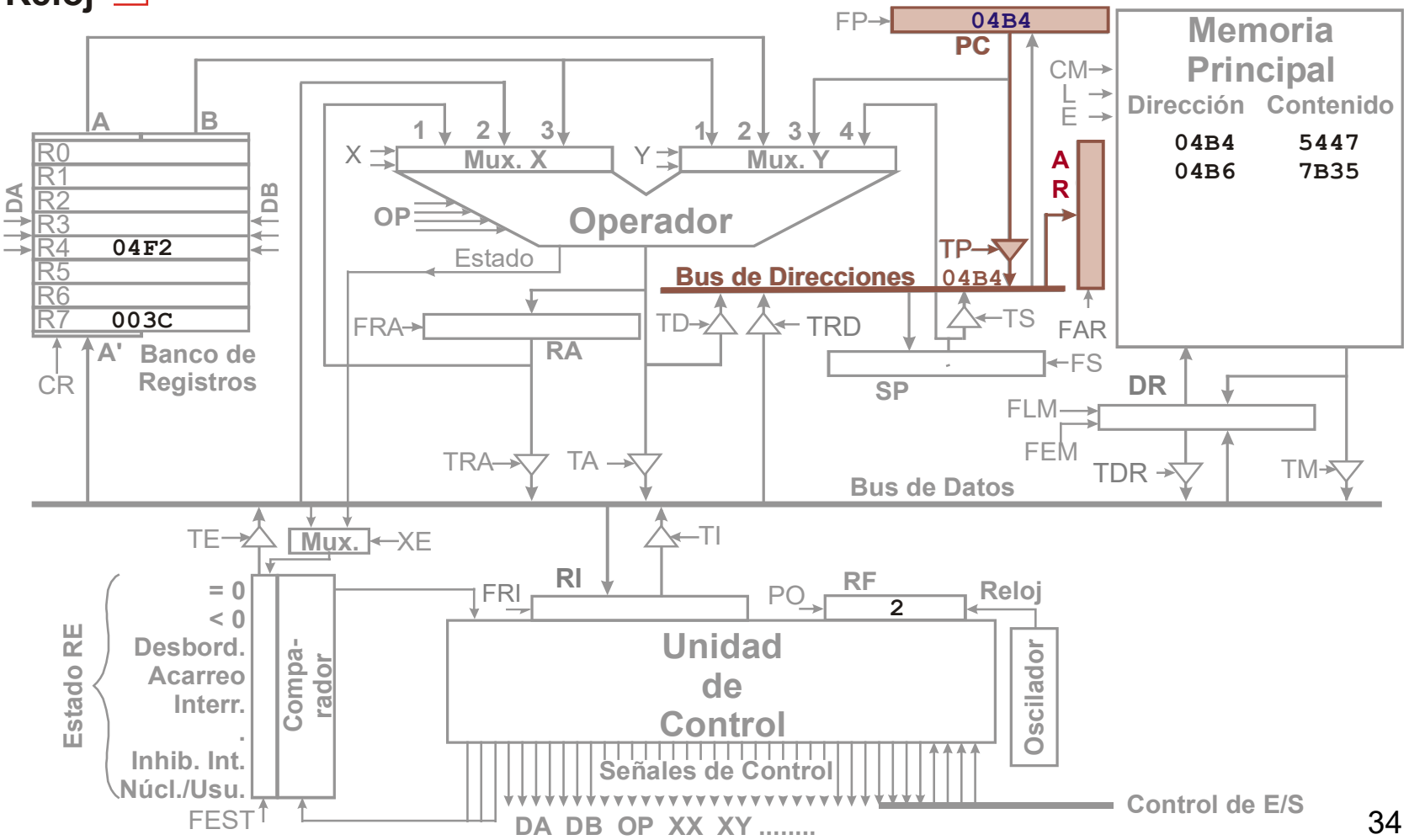
Además de estas operaciones, se incrementa el contador de programa, para ir preparando la lectura de la siguiente instrucción.

PC ← PC + 2

Prepara la lectura instrucción

Se prepara: D ← PC

Reloj

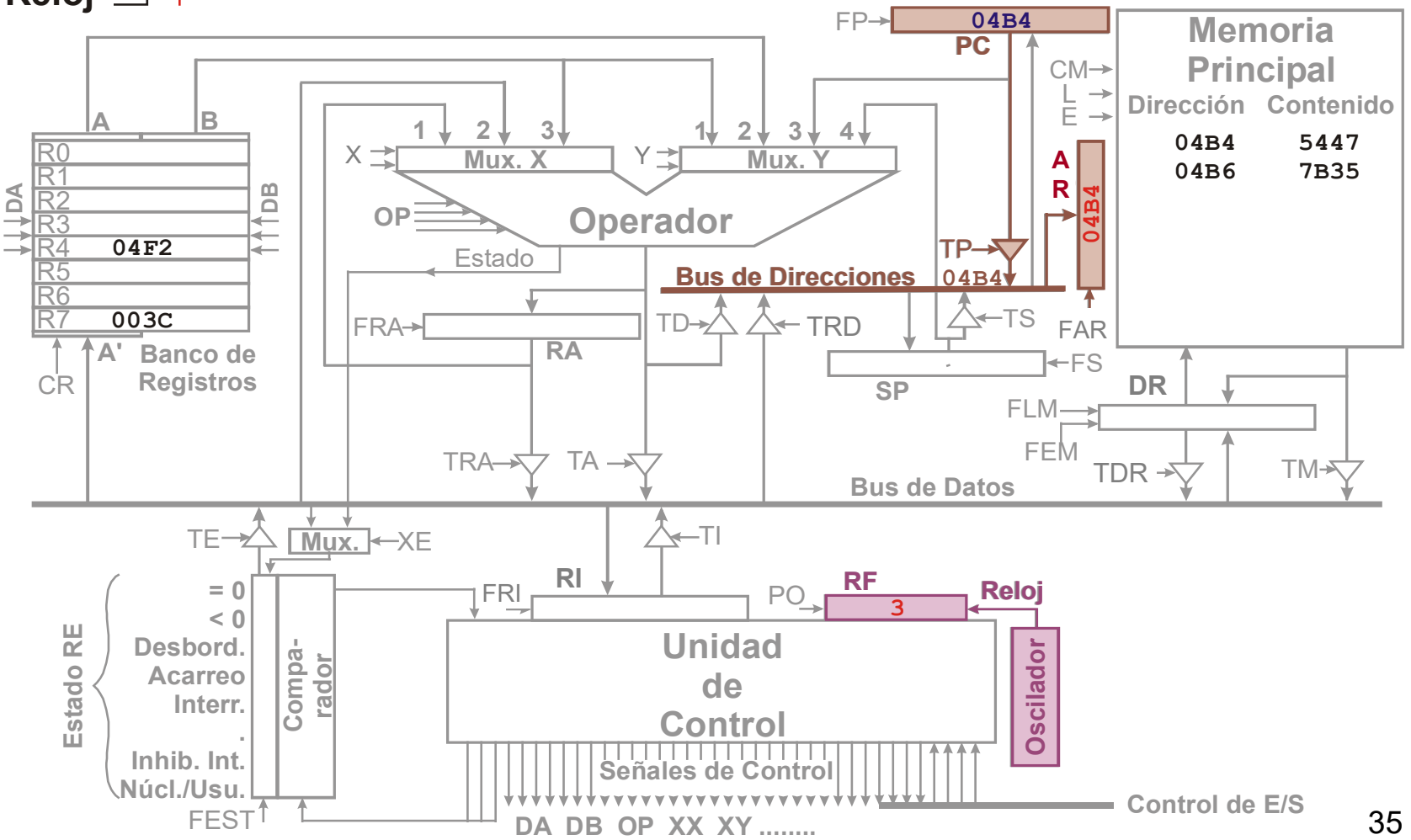


Prepara la lectura instrucción

Se realiza: D ← PC

Se incrementa RF: RF ← RF + 1

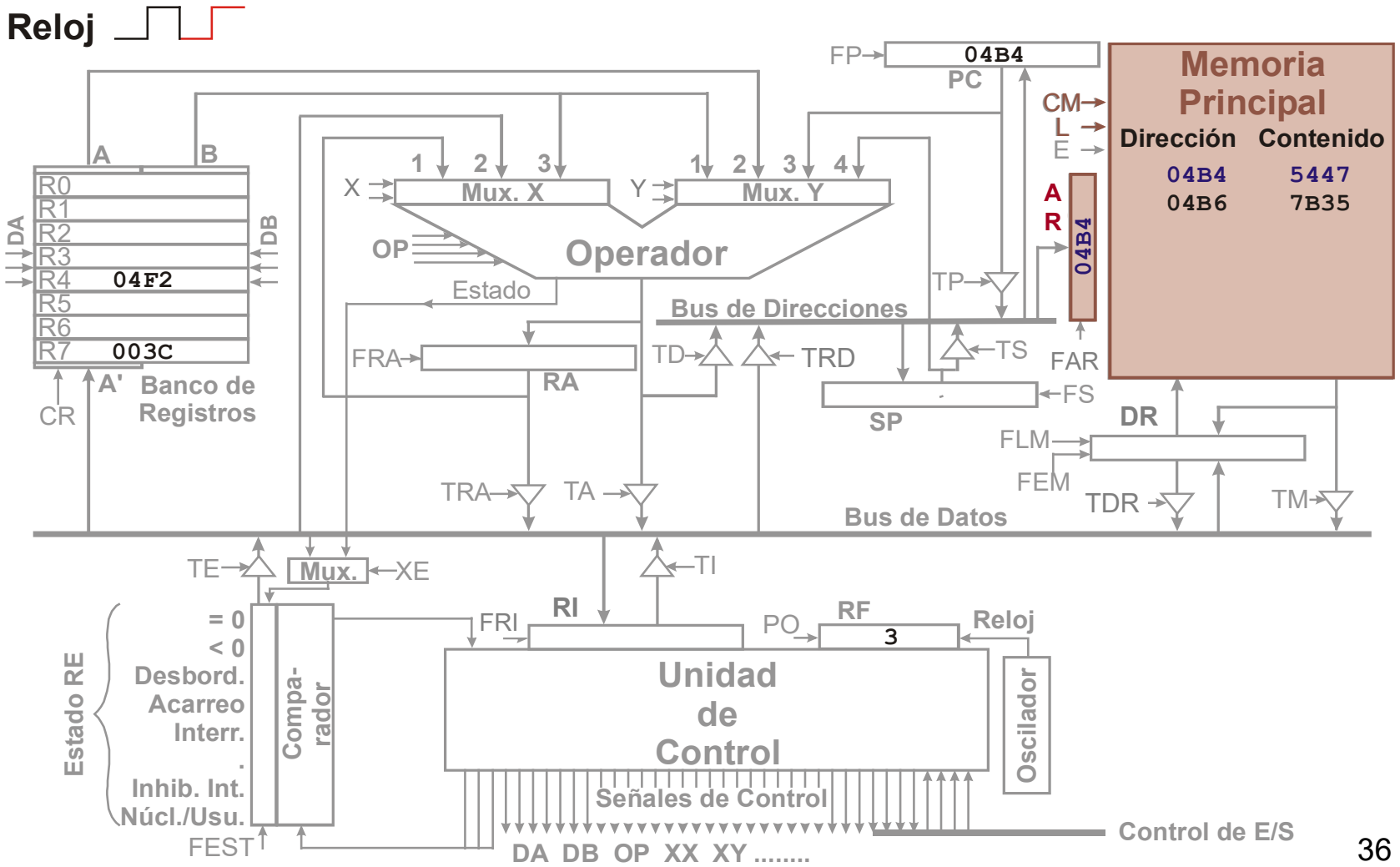
Reloj



ADD .R4,.R7

Lectura instrucción

Primer ciclo de memoria

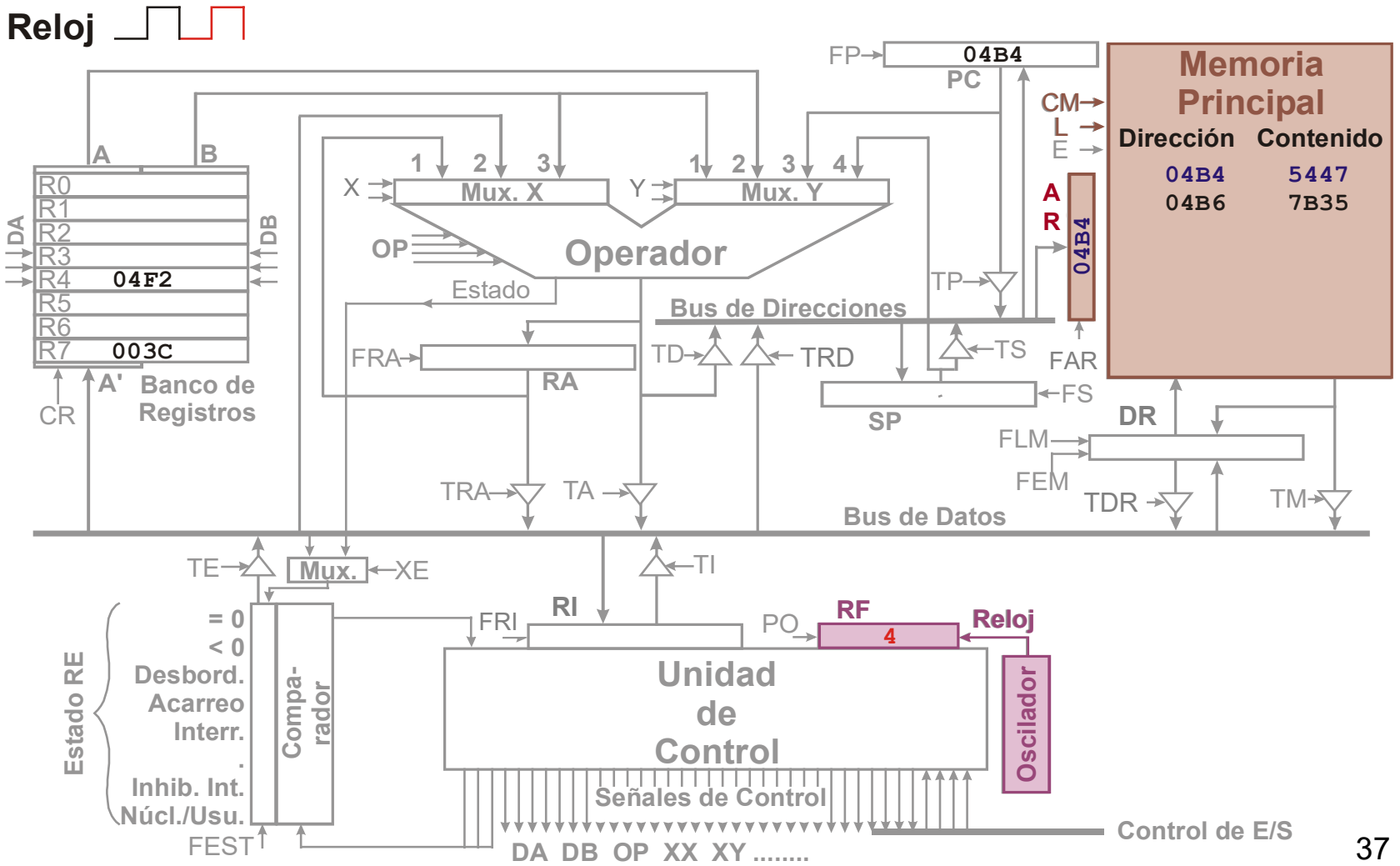


ADD .R4,.R7

Lectura instrucción

Primer ciclo de memoria

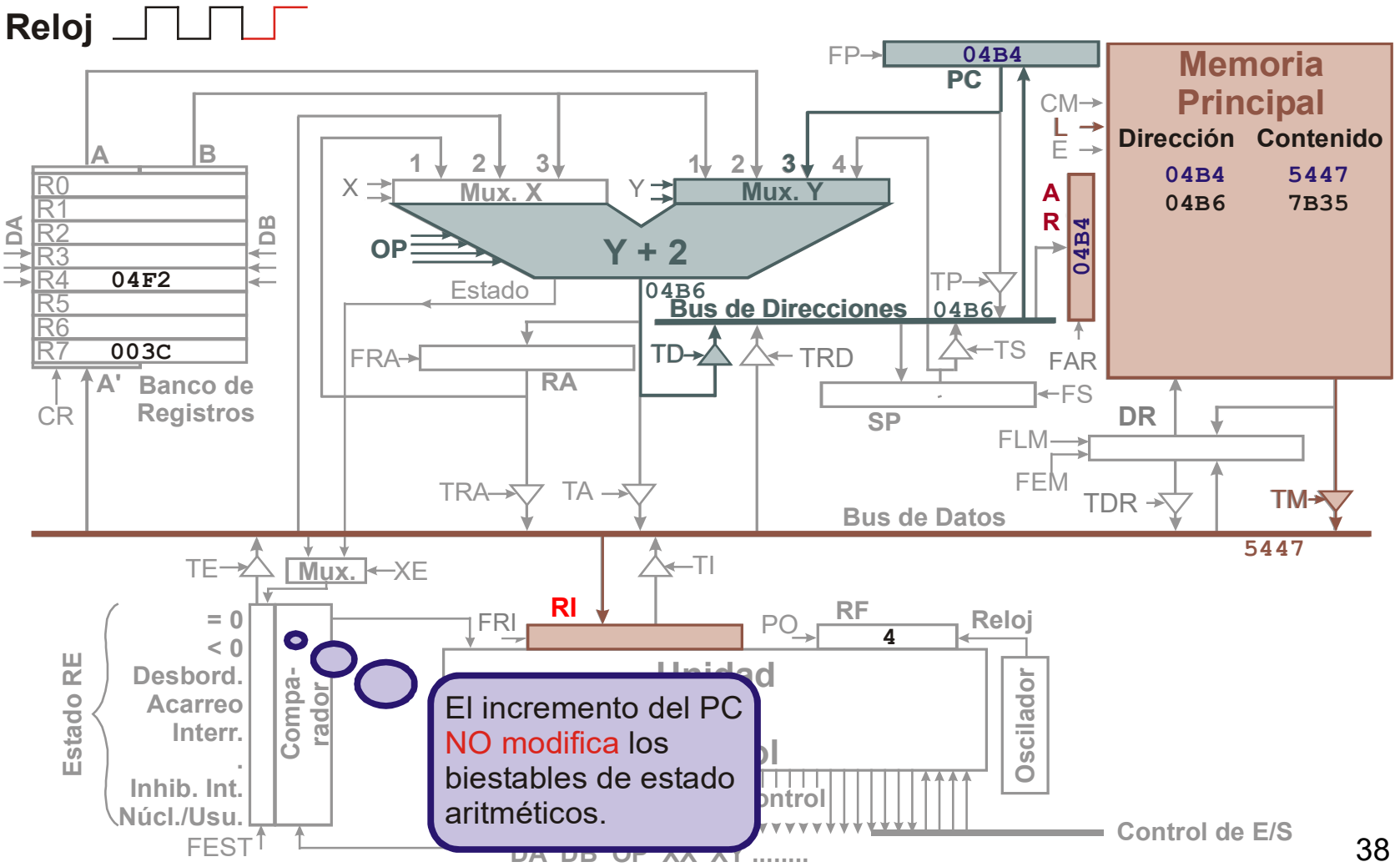
Se incrementa RF: RF ← RF + 1



Lectura instrucción

Segundo ciclo de memoria, se prepara: I ← M(D)

Se prepara incremento de PC: PC ← PC + 2



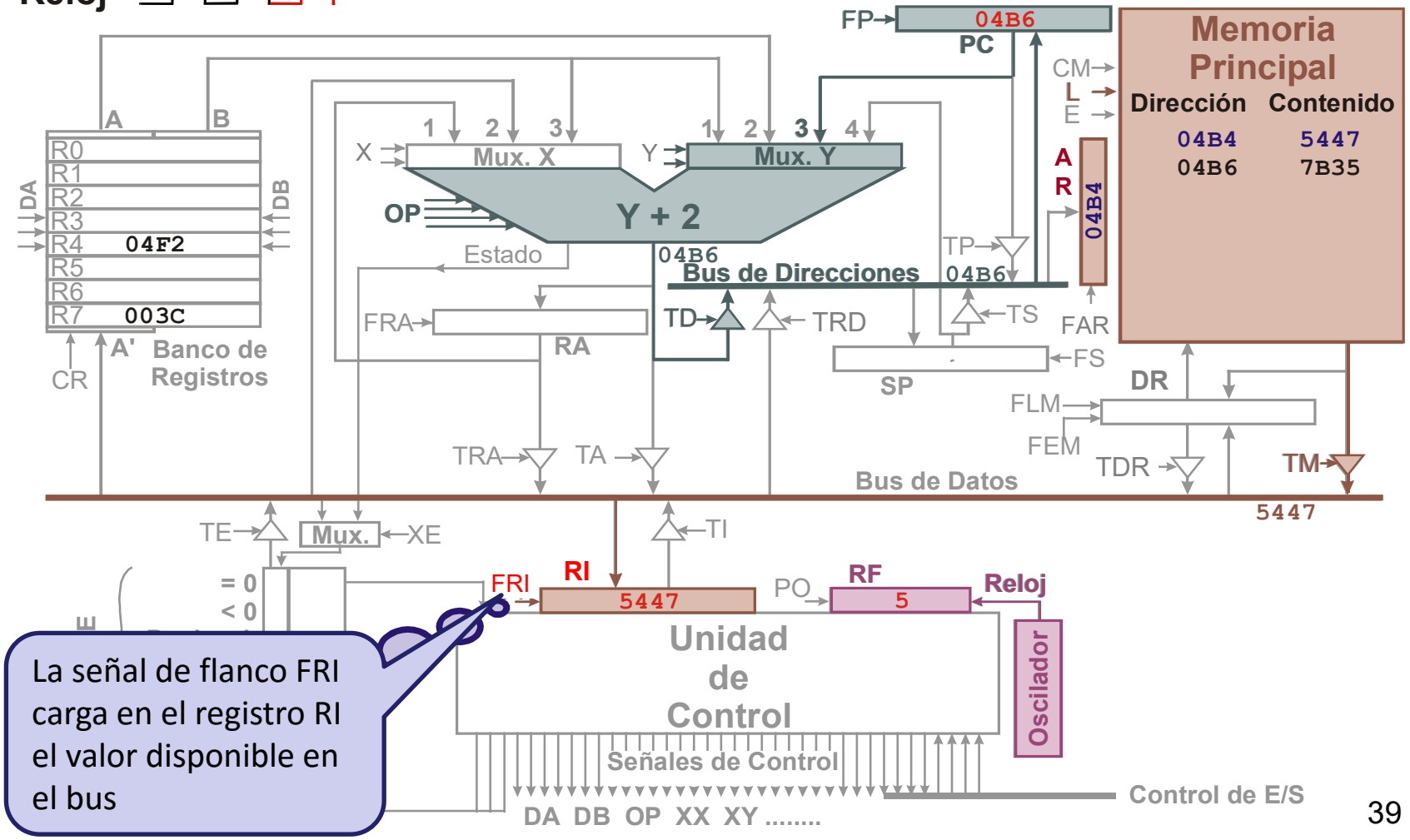
Lectura instrucción

Segundo ciclo de memoria, se realiza: I ← M(D)

Se realiza incremento de PC: PC ← PC + 2

Se incrementa RF: RF ← RF + 1

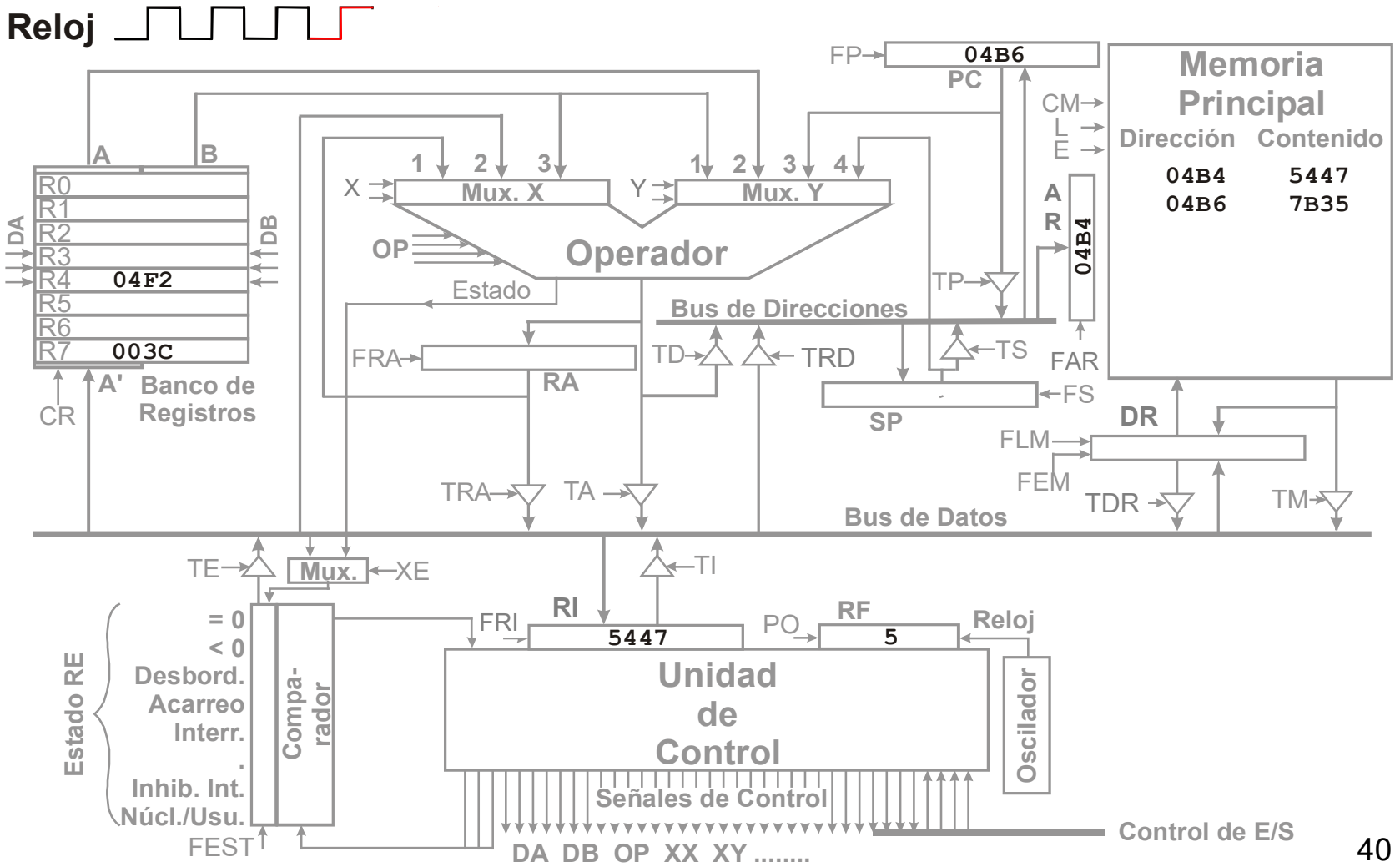
Reloj 



La señal de flanco FRI carga en el registro RI el valor disponible en el bus

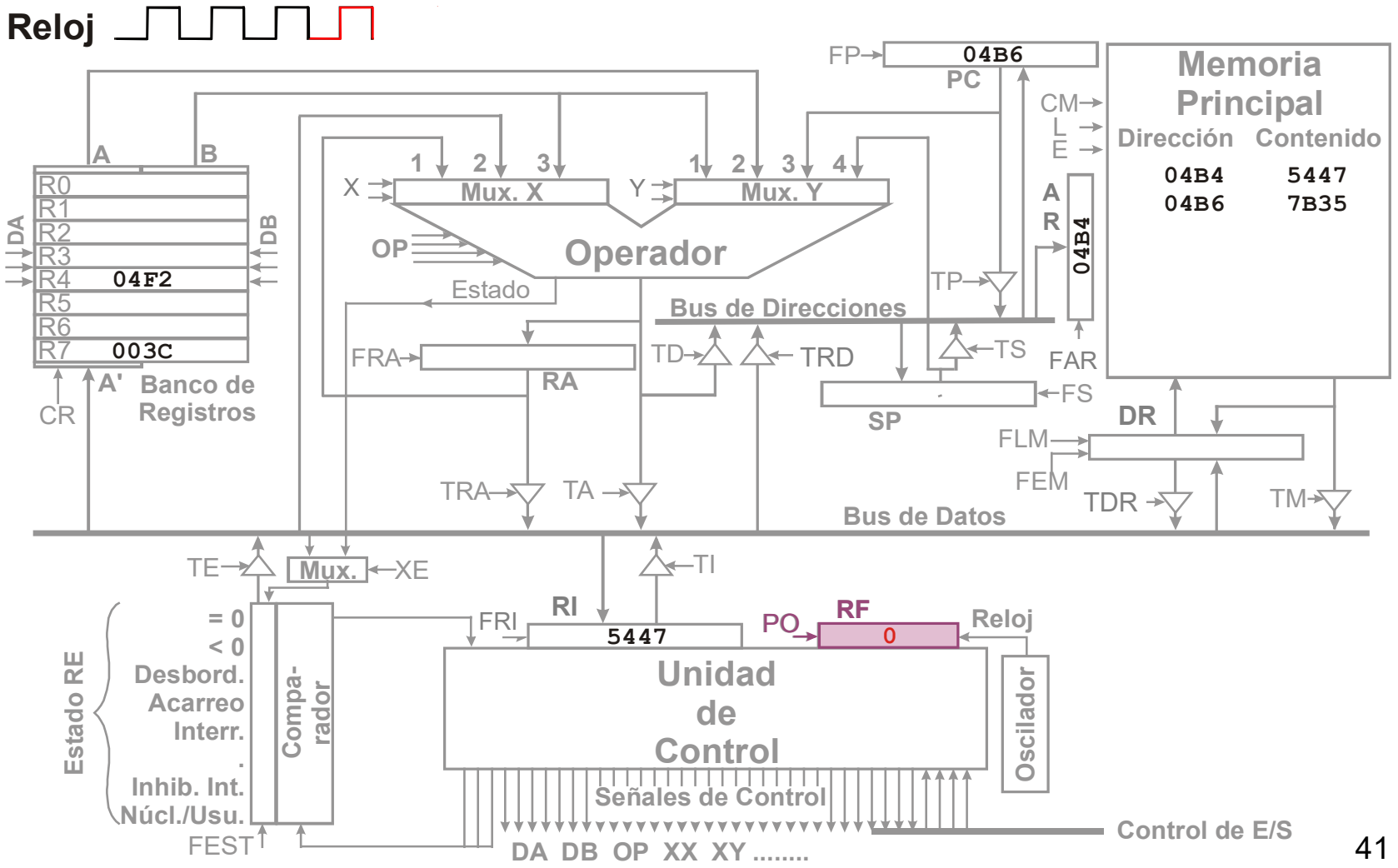
Decodificación instrucción

ADD .R4,.R7



Decodificación instrucción

Se pone RF a 0: RF ← 0

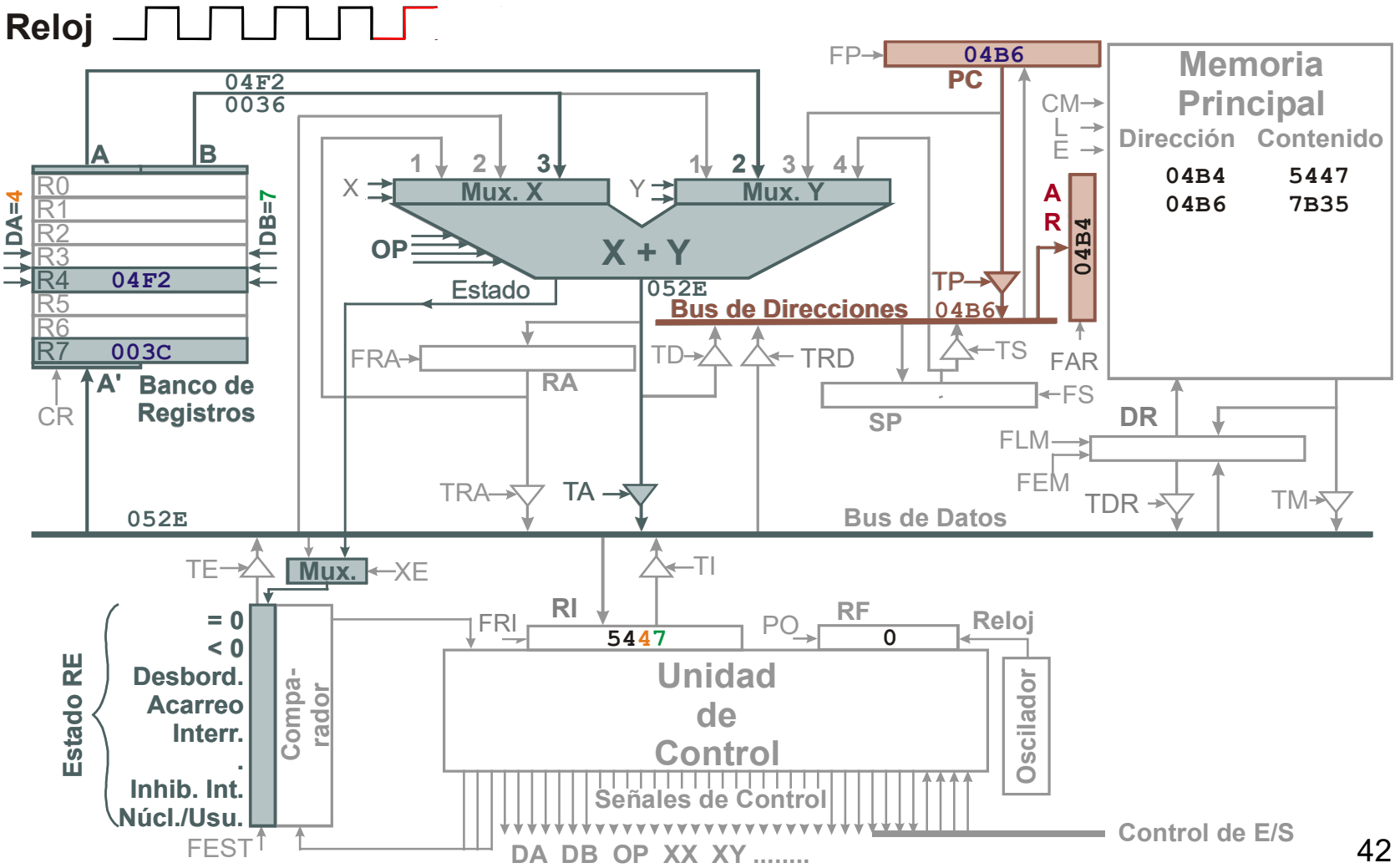


ADD .R4,.R7

Ejecuta instrucción y prepara lectura instrucción siguiente

Se prepara suma: R4 ← R4 + R7; RE ← Estado ALU

Se prepara: D ← PC

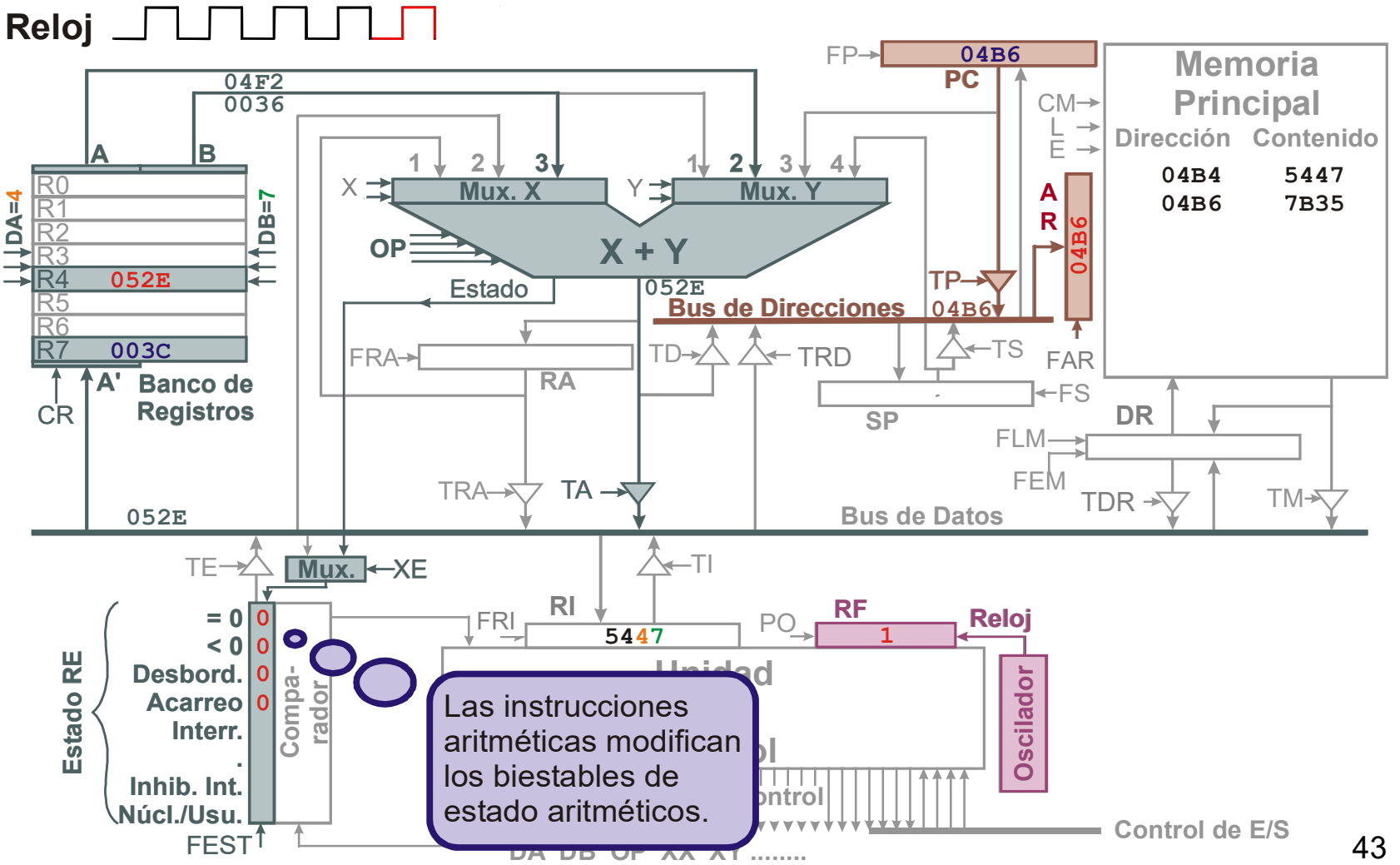


Ejecuta instrucción y prepara lectura instrucción siguiente

Se realiza suma: R4 ← R4 + R7; RE ← Estado ALU

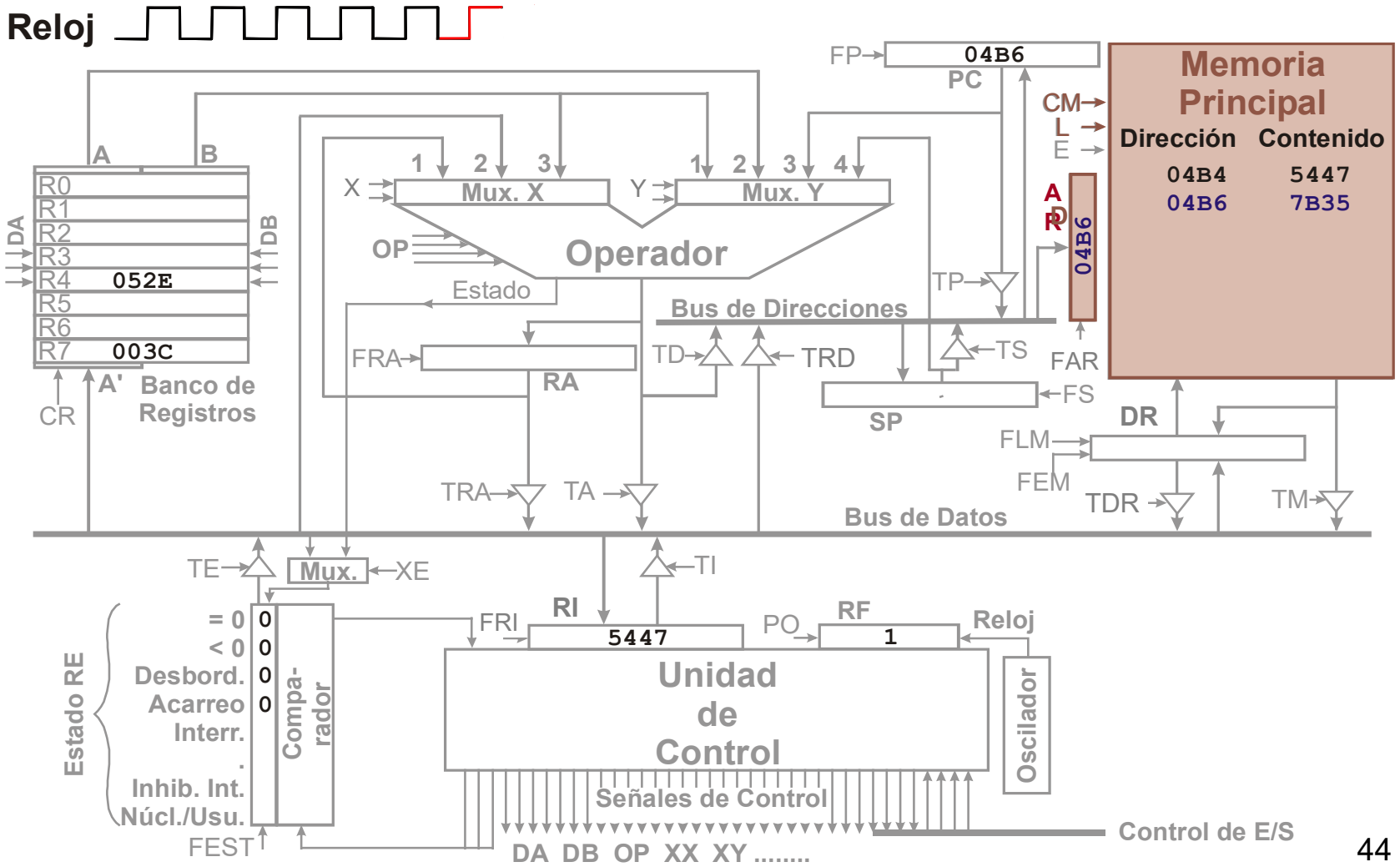
Se realiza: D ← PC

Se incrementa RF: RF ← RF + 1



Lectura instrucción siguiente

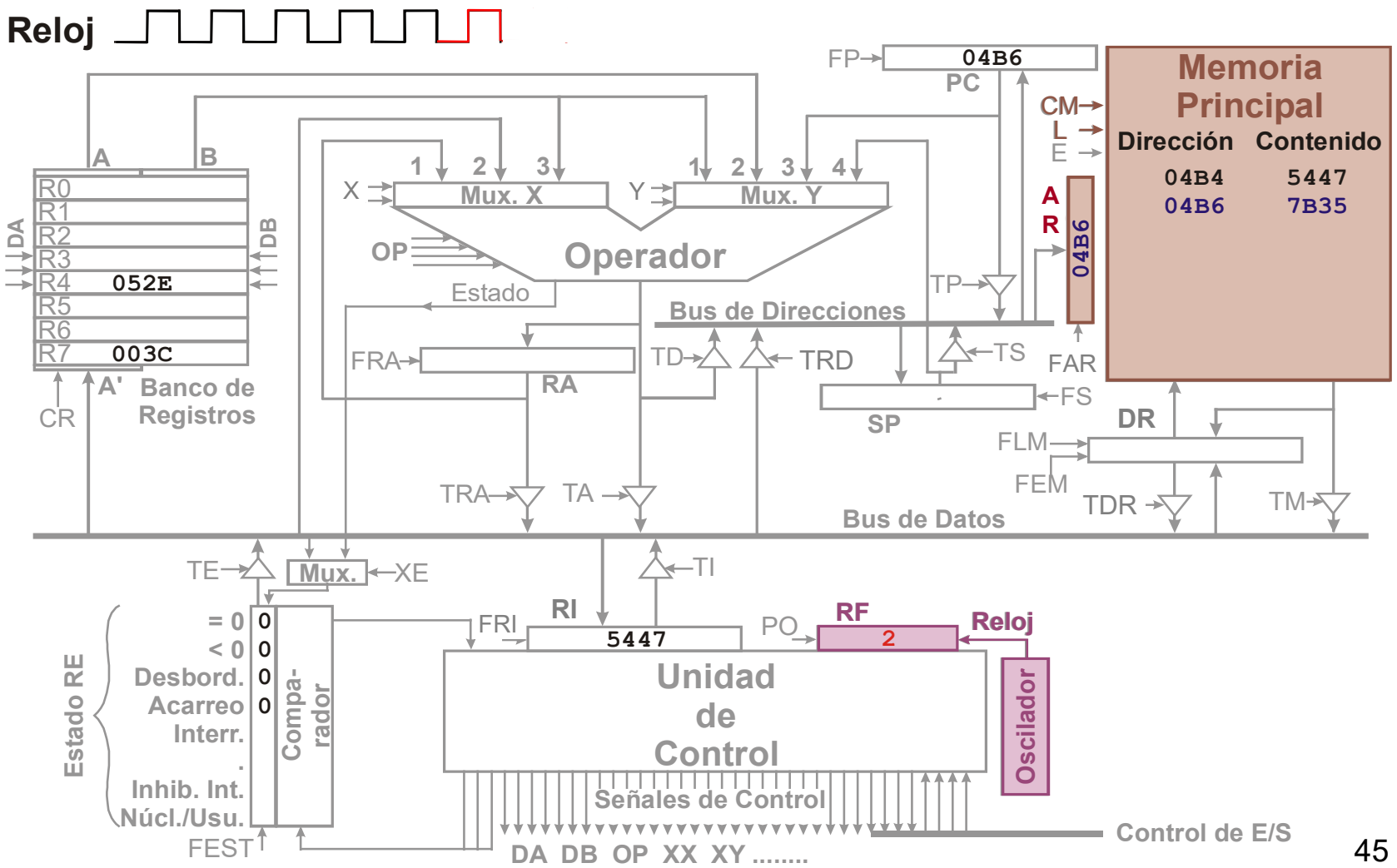
Primer ciclo de memoria



Lectura instrucción siguiente

Primer ciclo de memoria

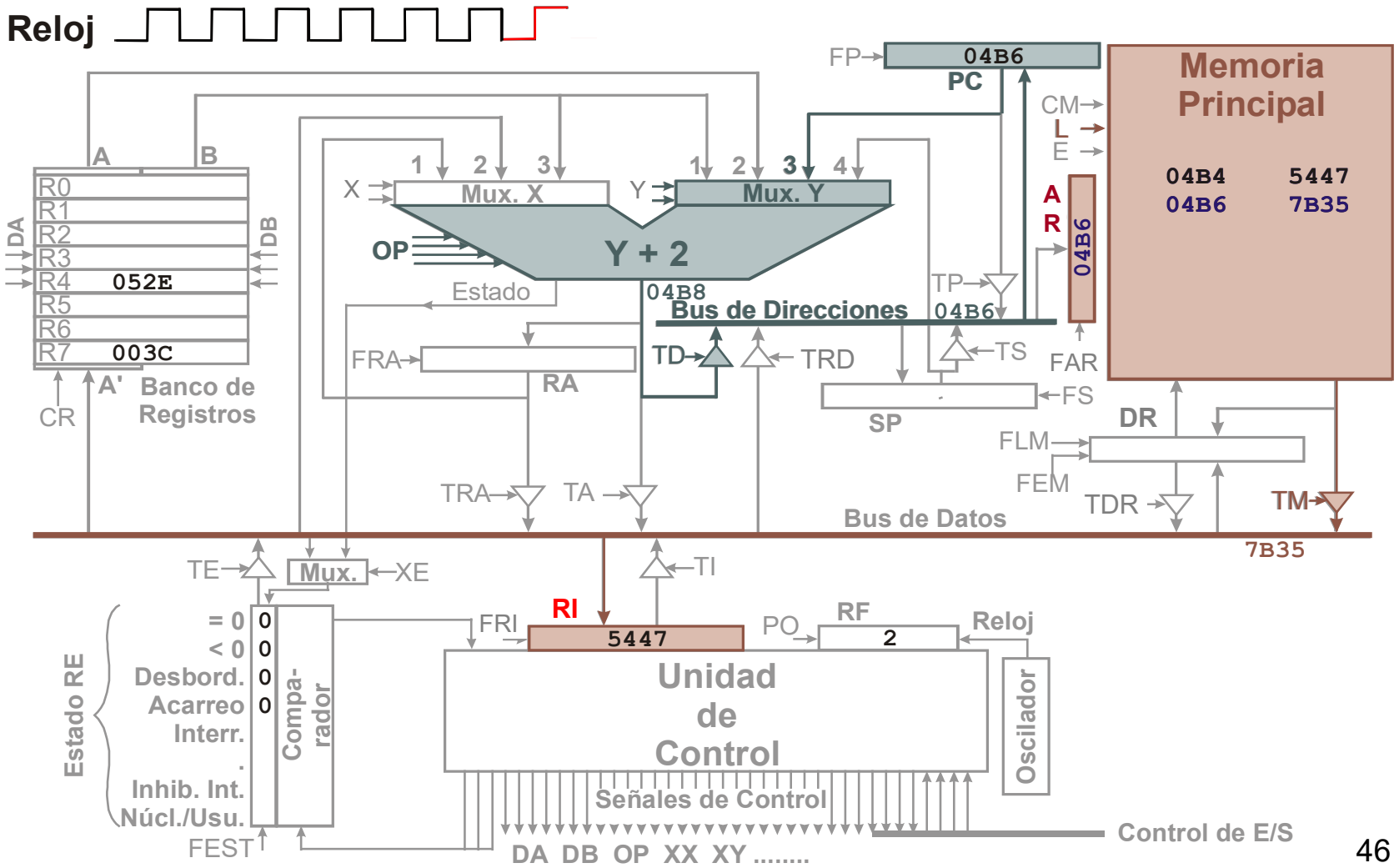
Se incrementa RF: RF ← RF + 1



Lectura instrucción

Segundo ciclo de memoria, se prepara: I ← M(D)

Se prepara incremento de PC: PC ← PC + 2

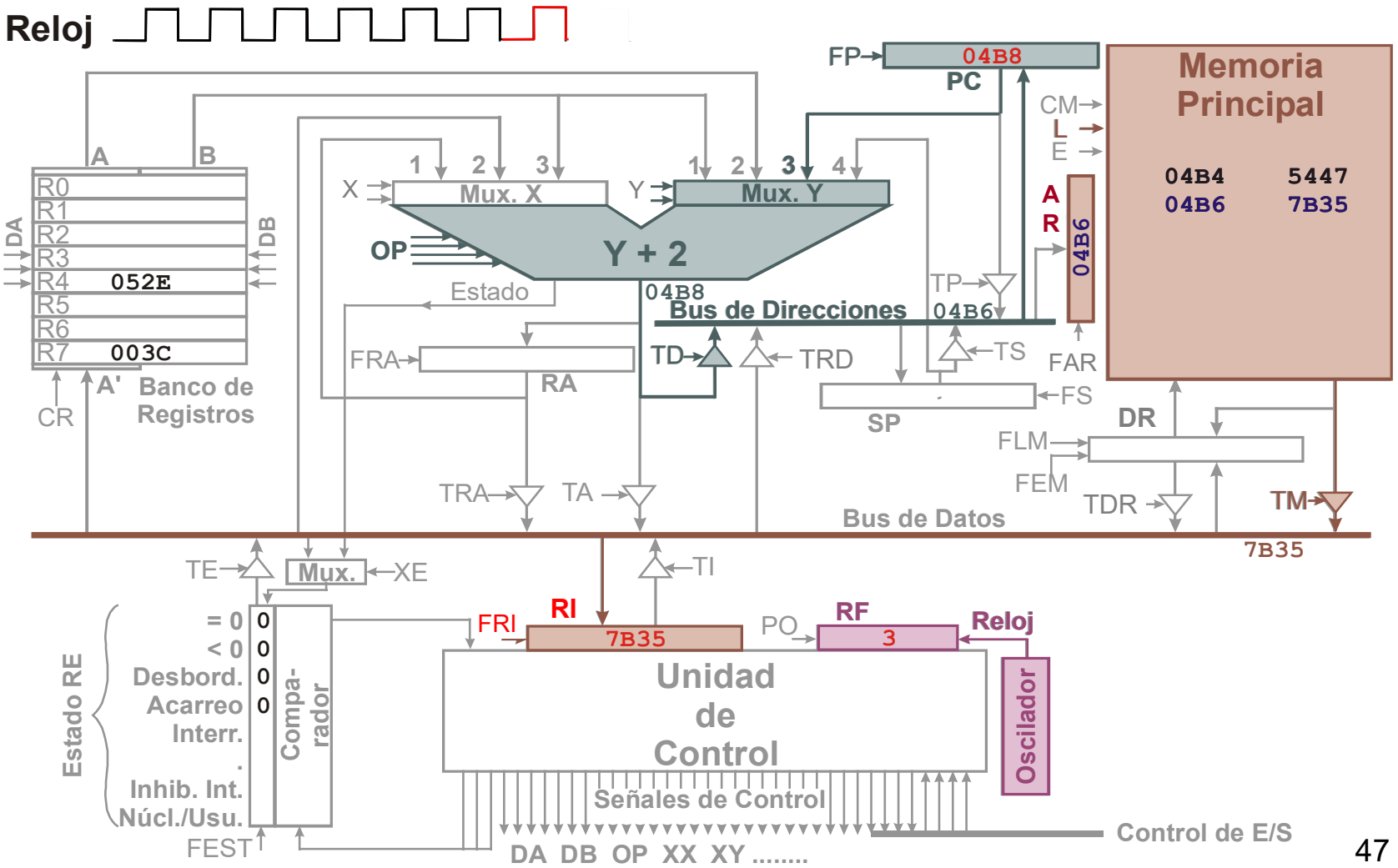


Lectura instrucción

Segundo ciclo de memoria, se realiza: I ← M(D)

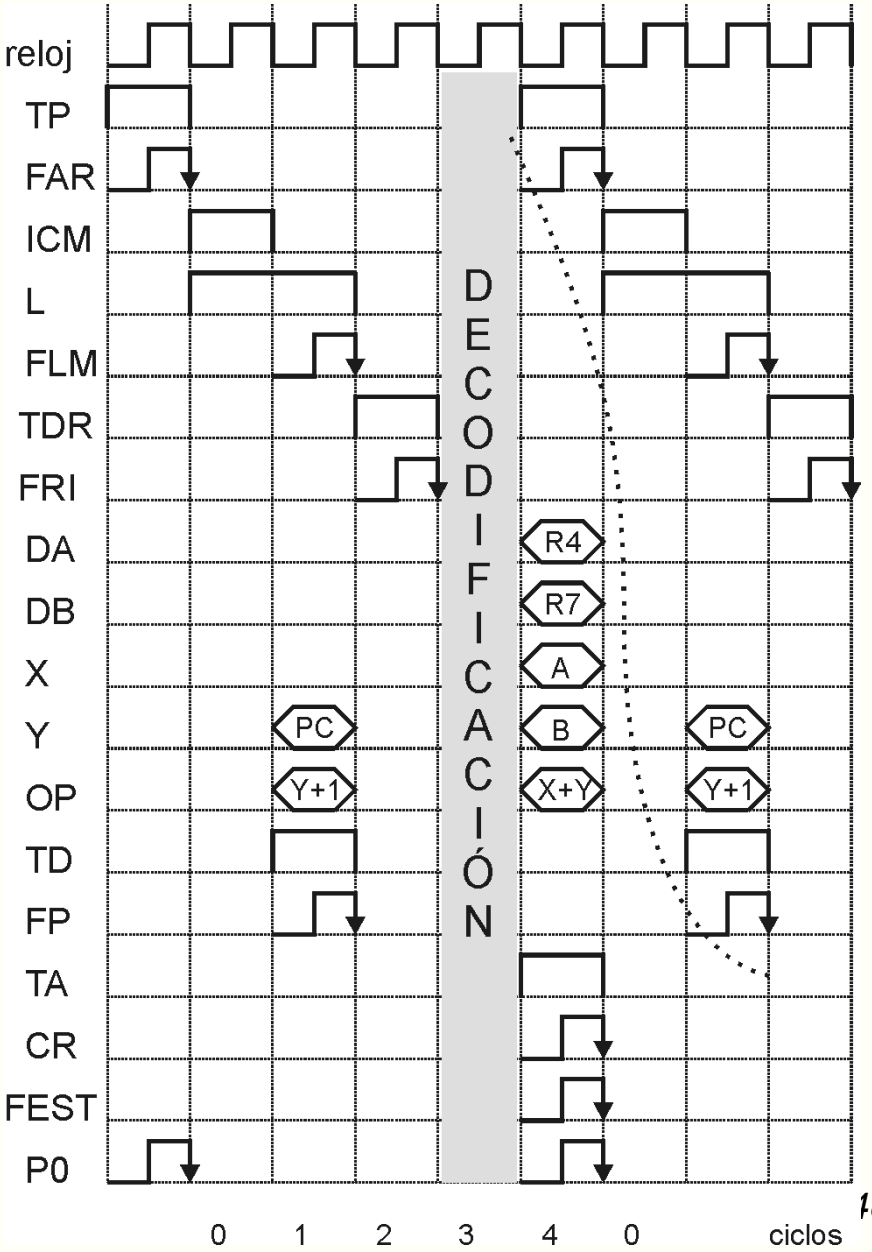
Se realiza incremento de PC: PC ← PC + 2

Se incrementa RF: RF ← RF + 1



Cronogramas

- *ADD .R4, .R7*
- *Lectura de la instrucción*
  - PC → AR (1 ciclo)
  - M(AR) → DR, PC + 1 → PC (2 ciclos)
  - DR → RI (1 ciclo)
- *Decodificación* (1 ciclo)
- *Ejecución y fetch sig. Instrucción*
  - R4+R7 → R4 y FEST;
    PC → AR (1 ciclo)
  - M(AR) → DR, PC + 1 → PC (2 ciclos)
  - DR → RI (1 ciclo)



Índice

- Índice
  - Introducción
  - Operaciones elementales
  - Estructura de un computador elemental y sus señales de control
  - Cronogramas
  - Diseño de la Unidad de Control (UC)
  - Gestión de excepciones

Diseño de la UC

- El diseño de la UC exige haber definido previamente
  - El repertorio de instrucciones (formatos y direccionamientos)
  - La estructura del computador

    ➔Conjunto de cronogramas
- La UC funciona como un TRADUCTOR

  RI(CO, MD), Reloj, Estado, señales ext. → señales de control

  Ej: CO: 8 bits, 1 instr=16 ciclos → RF: 5 bits, …

  traductor de unas 20-30 señales de entrada y 150 señales salida

Diseño de la UC

- Formas de diseño de la UC
  - UC CABLEADA
    - Utilizando exclusivamente puertas lógicas
  - UC MICROPROGRAMADA
    - Utilizando una memoria de control

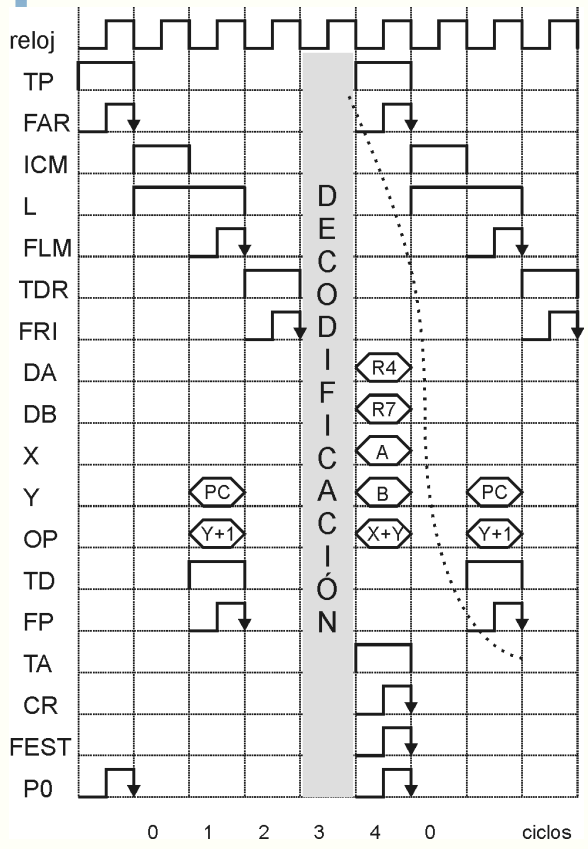
Diseño de la UC

- **UC CABLEADA**
  - Se construye mediante métodos generales del **diseño lógico** (circuito secuencial o combinacional)
  - Más **rápidas**
  - Diseño y construcción **complejo** y **costoso**
  - Las técnicas actuales (CAD, comp silicio) permiten que sea solución viable y válida cara a construir **máquinas rápidas**

Diseño de la UC

- **UC MICROPROGRAMADA**
  - Utiliza una **memoria** para almacenar el **estado** de las **señales de control** en cada periodo de la ejecución de cada instrucción.
  - Generar el **cronograma** de cada instrucción (ejecutar la Inst) es ir leyendo de esta memoria (Memoria de Control**,** **MC**).

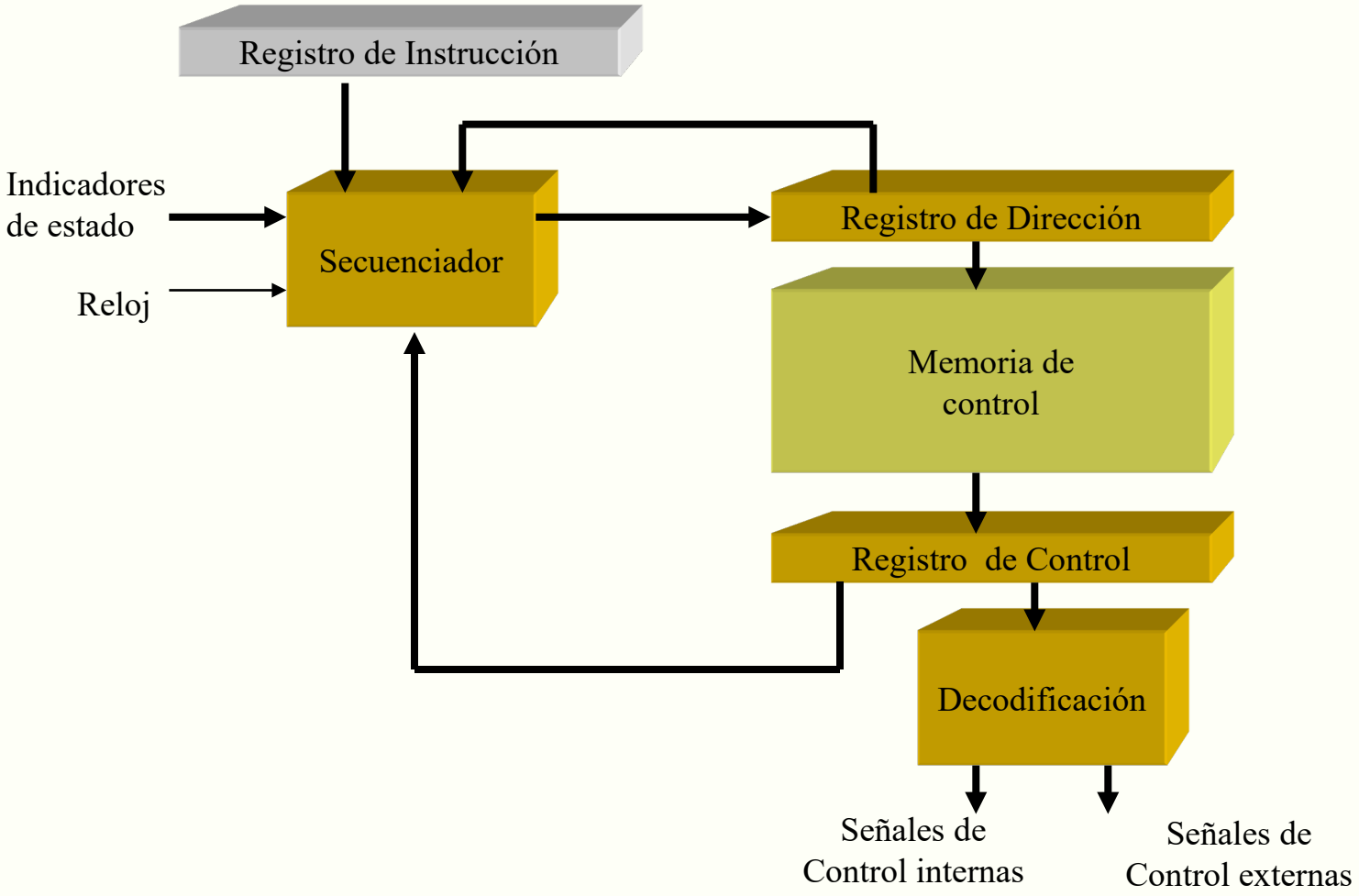
Diseño de la UC



Memoria de Control (MC)

TP	FAR	ICM	L	FLM	TDR	DA				DB				FRI	X		Y		OP				TD	FP	TA	CR	FEST	P0	...	...
x	x	x	x	x	x	0	0	0	0	0	0	0	1	1	0	1	0	1	0	0	0	0	1	1	0	0	1	0	1	0
1	0	0	0	1	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	1	0	1	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	1	0	0	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0
1	0	0	0	1	1	0	0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
1	0	1	0	1	1	0	0	1	0	0	1	0	1	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0
x	x	x	x	x	x	0	0	1	1	0	1	0	0	0	1	1	0	0	0	0	1	0	0	0	0	1	1	0	0	0
x	x	x	x	x	x	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1
x	x	x	x	x	x	0	1	0	1	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	1	0	1	0	0
x	x	x	x	x	x	0	1	1	0	0	1	1	1	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0
x	x	x	x	x	x	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
x	x	x	x	x	x	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
x	x	x	x	x	x	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Diseño de la UC



Diseño de la UC

UC CABLEADA	UC MICROPROGRAMADA
Ventajas • **Alta velocidad** de funcionamiento • Implementaciones más pequeñas (reducido número de componentes)	Ventajas • Son sistemáticas con un formato bien definido • Pueden ser fácilmente **modificables** durante el proceso de diseño
Inconvenientes • Difícil realizar modificaciones en el diseño: modificación → rediseño • No tienen estructura común	Inconvenientes • Requieren más componentes para su implementación • Tienden a ser **más lentas** que las unidades de control cableadas debido a que tienen que realizar operaciones de lectura de una memoria para obtener las señales de control
▪ Técnica de diseño favorita en las arquitecturas RISC	▪ Emulación • Se puede utilizar un microprograma para que la UC interprete otro lenguaje máquina distinto (una máquina a emular) sin necesidad de realizar modificaciones en el hardware de la unidad de control -> cambiando sólo el microprograma!

Diseño de la UC

- Conceptos básicos
  - MICROINSTRUCCIÓN
  - MICROPROGRAMA

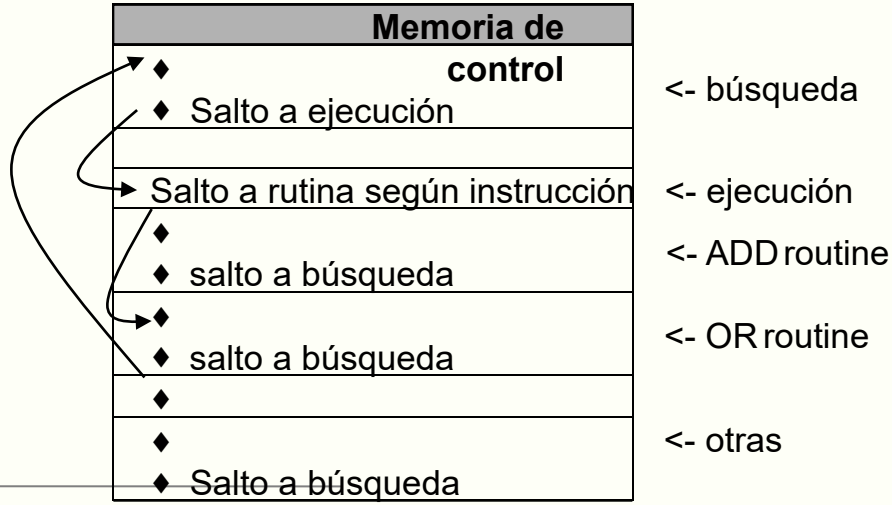
Conceptos básicos

- **Microinstrucción**
  - **Cadena de ceros y unos** que
    - representa la activación o no del conjunto de señales de control durante un ciclo de reloj
    - y la posibilidad de decidir condicionalmente qué microinstrucción se debe ejecutar a continuación
  - **Cada palabra** de la memoria de control es una **microinstrucción**.

Conceptos básicos

- **Microprograma**
  - Secuencia de μI´s cuya ejecución permite interpretar una instrucción
  - Ejecutar una microinstrucción ≈ Leer una μI de la MC
  - Una vez rellenada la memoria de μI´s:

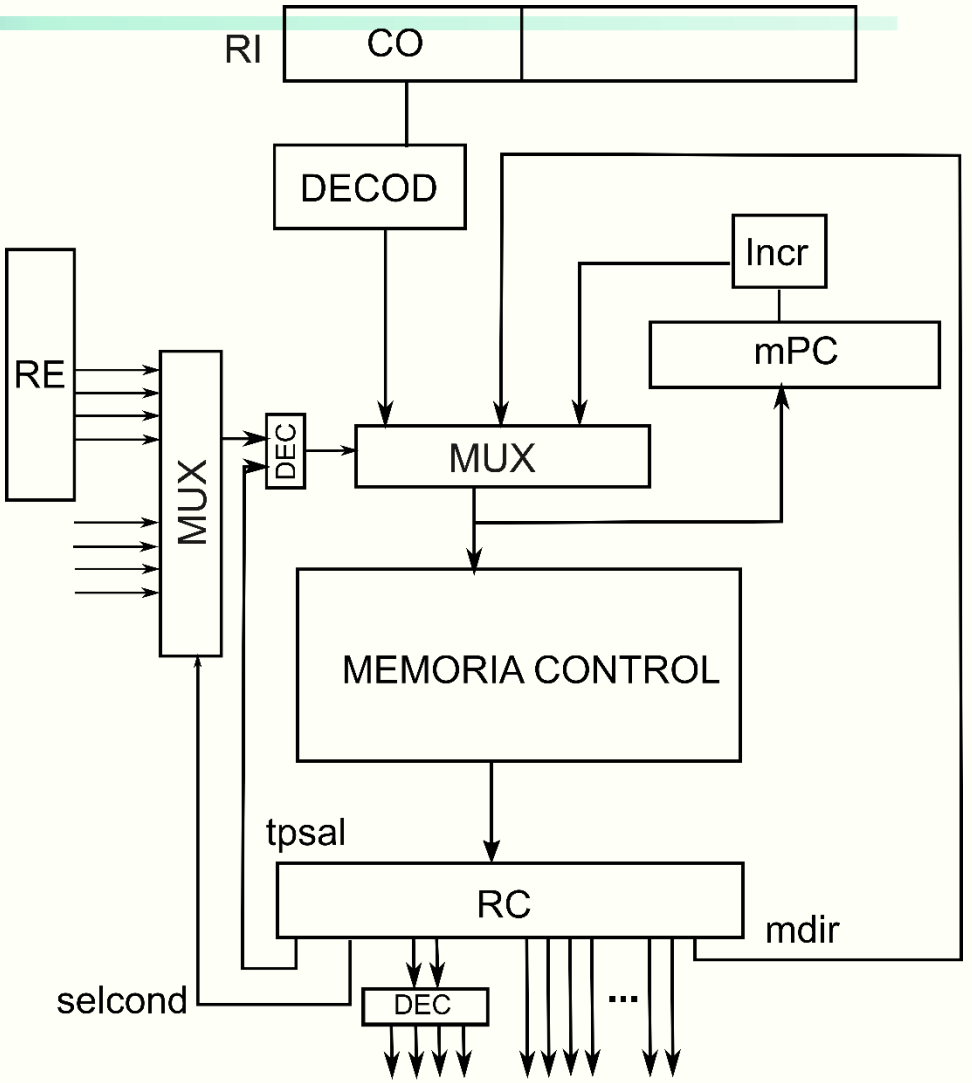
    el control es tarea de (micro)programación
  - Cambiando microprogramas se puede ejecutar otros repertorios de instrucc (simular otras arquit) **-> EMULACIÓN!**



Diseño de la UC

Estructura básica de la UC mprogramada

- microsaltos condicionales



Índice

- Índice
  - Introducción
  - Operaciones elementales
  - Estructura de un computador elemental y sus señales de control
  - Cronogramas
  - Diseño de la Unidad de Control (UC)
  - Gestión de excepciones

Gestión de excepciones

- Además de ejecutar la secuencia de instrucciones del programa de usuario, la UC debe controlar situaciones excepcionales

  - ¿Qué ocurre si el CO no es válido?
  - ¿Y si hay *overflow*?
  - ¿Y si el divisor es igual a cero?
  - ¿Y si se ejecuta una *instrucción privilegiada* en Modo Usuario?
  - ¿Cómo y cuándo comunicarse con los periféricos?
  - …

Gestión de excepciones

- **Excepción** [Overflow, división por cero, etc.]
  - Evento no planificado que detiene la ejecución de un programa: interno
    - Salvar el PC y Estado actuales en pila
    - Pasar a modo SO (modo supervisor)
    - Saltar a rutina de SO que maneje excepciones internas
    - SO retornará después al programa, en su anterior estado
- **Interrupción** [E/S]
  - Un evento externo inesperado rompe la ejecución del programa (Tema 7, E/S)

Gestión de excepciones

Excepciones

- Causadas por eventos internos
  - Condiciones de excepción (ej: overflow)
  - Errores (ej: error de alineamiento de memoria)
  - Fallos (ej: fallo de página)
  - Llamadas al sistema
- Síncronos a la ejecución del programa (se puede saber cuándo)
- Puede que el programa de usuario se retome tras tratar la excepción o bien que se “aborte” (¿’cancele’? ;-)

- **TRAP**: llamada al sistema

Gestión de excepciones

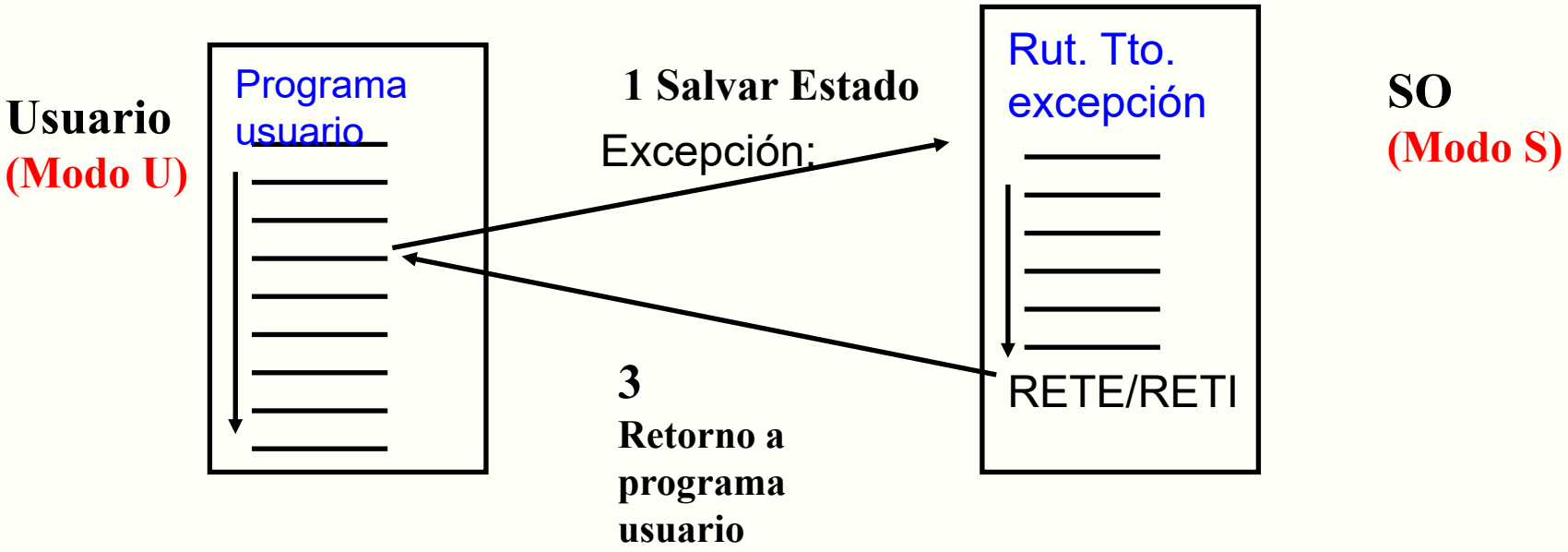
Interrupciones

- Causadas por eventos **externos** (peticiones de periféricos).
- Asíncronas a la ejecución del programa (en cualquier instante)
- Se atienden entre instrucciones (fetch)
- Obligan a suspender temporalmente la ejecución del programa de usuario

*FETCH**:* Si INT=1 e "INT_no_inhibidas" hacer
proceso_excepcion
Si no ir a FETCH

(Tema 7, E/S)

Gestión de excepciones



El sistema (**CPU + SO**) gestiona o trata la excepción: en general:

- Guarda la dir de la Instrucción en la que se produjo la excepción, PC
- Guarda RE del programa de usuario
- Pasa a **Modo Supervisor**
- Trata la excepción: ejecución de una **rutina de tratamiento**
- Restituye el estado del programa de usuario
- Retorna el control a usuario (salvo que se aborte), restituye PC