Grupo ARCOS

uc3m | Universidad Carlos III de Madrid

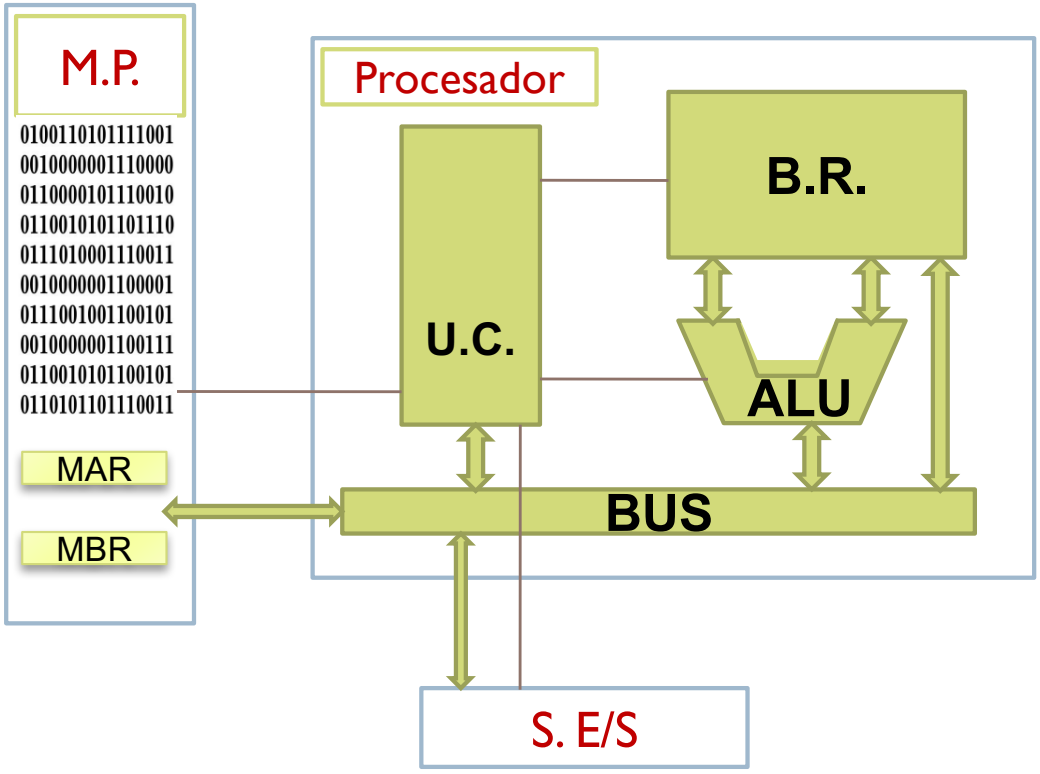
Tema 4 (I)
El procesador

Estructura de Computadores
Grado en Ingeniería Informática

Contenido

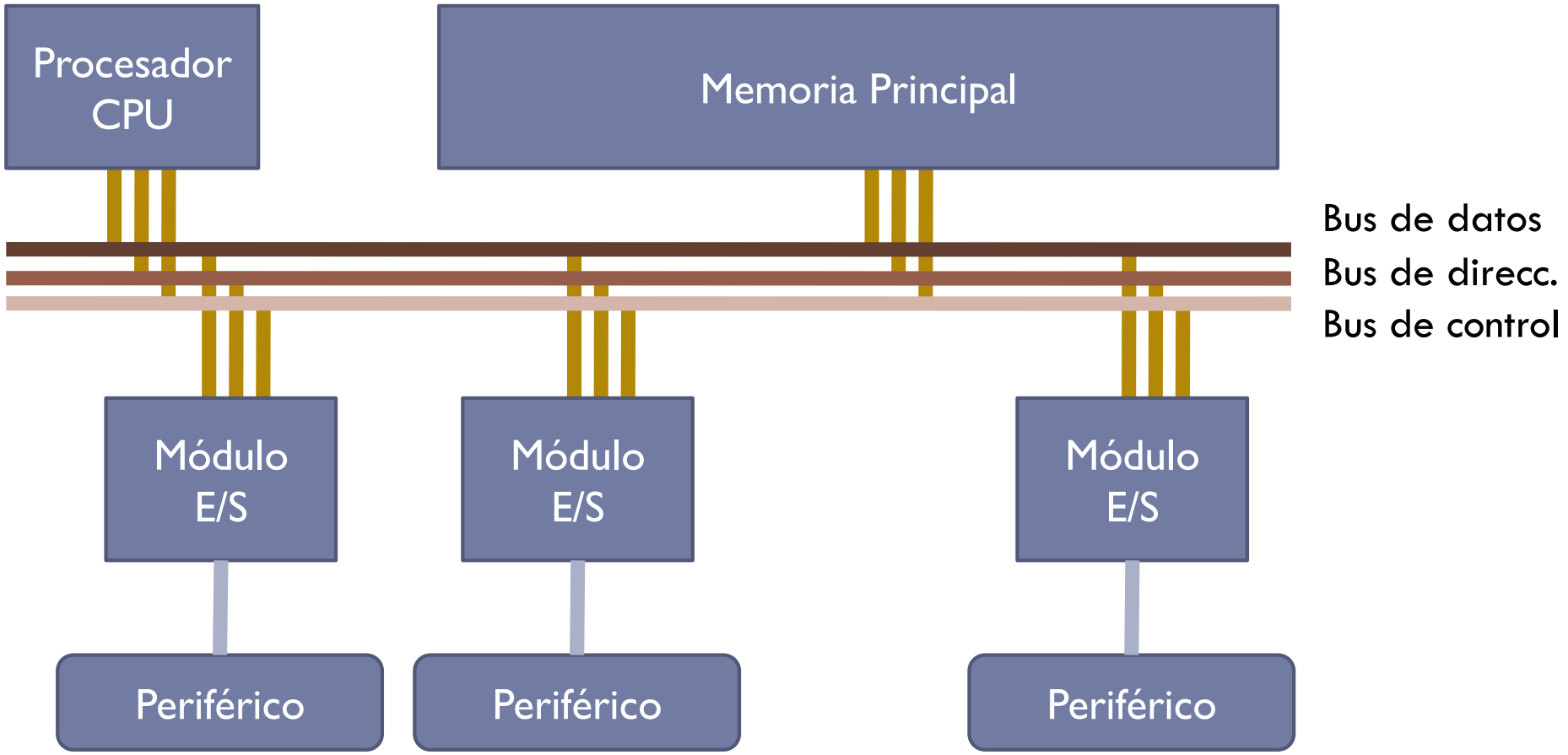
1. Elementos de un computador
2. Organización del procesador
3. La unidad de control
4. Ejecución de instrucciones
5. Modos de ejecución
6. Arranque de un computador
7. Interrupciones
8. Diseño de la unidad de control
9. Arranque de un computador
10. Prestaciones y paralelismo

Motivación

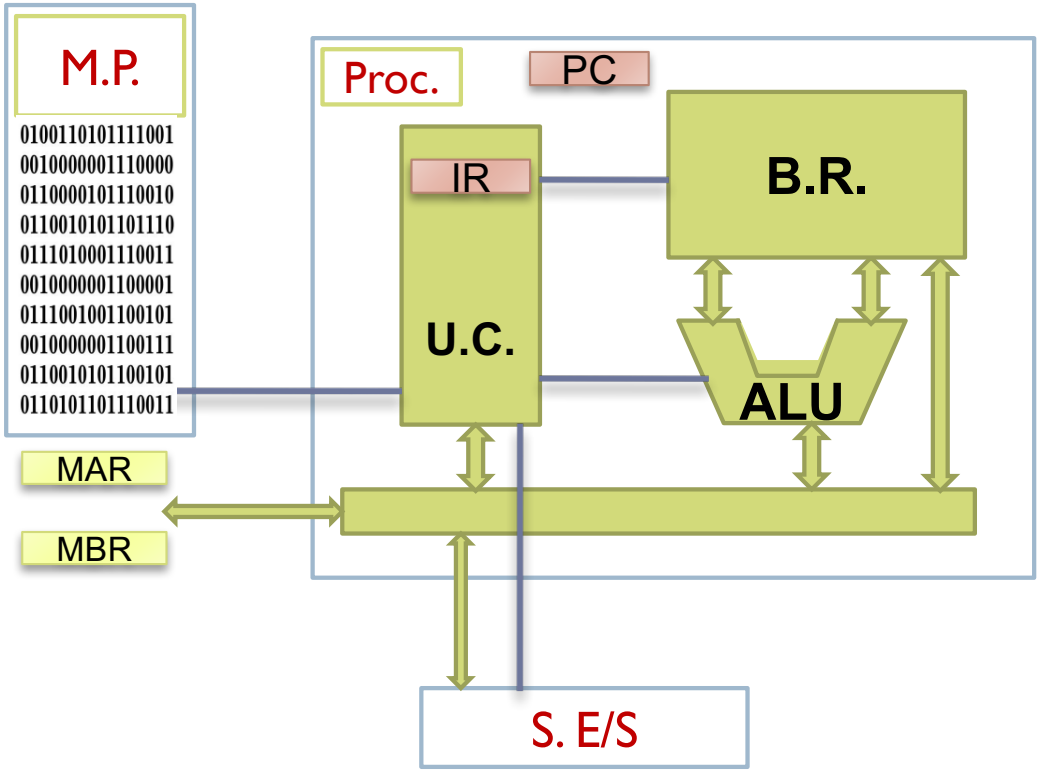


- En el tema 3 se estudian las instrucciones máquina
- En el tema 4 se estudia cómo se ejecutan las instrucciones en el computador

Componentes de un computador

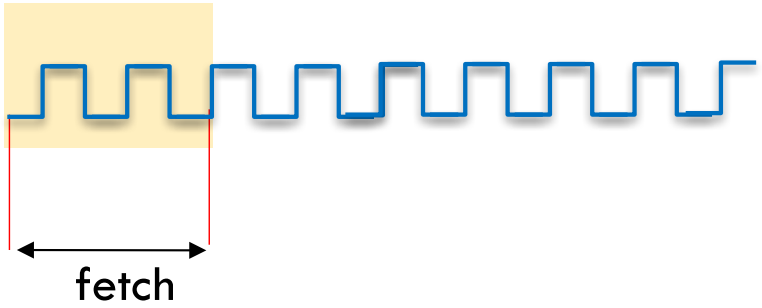


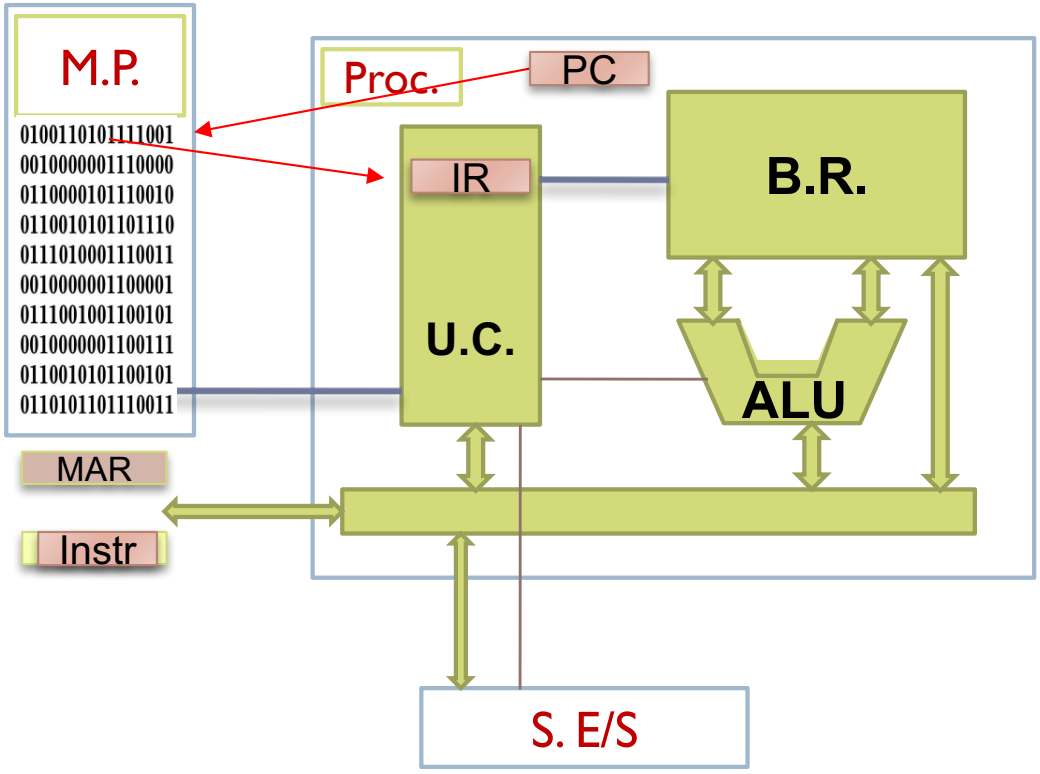
Funcionamiento básico de la UC



- Ejecutar instrucciones máquina

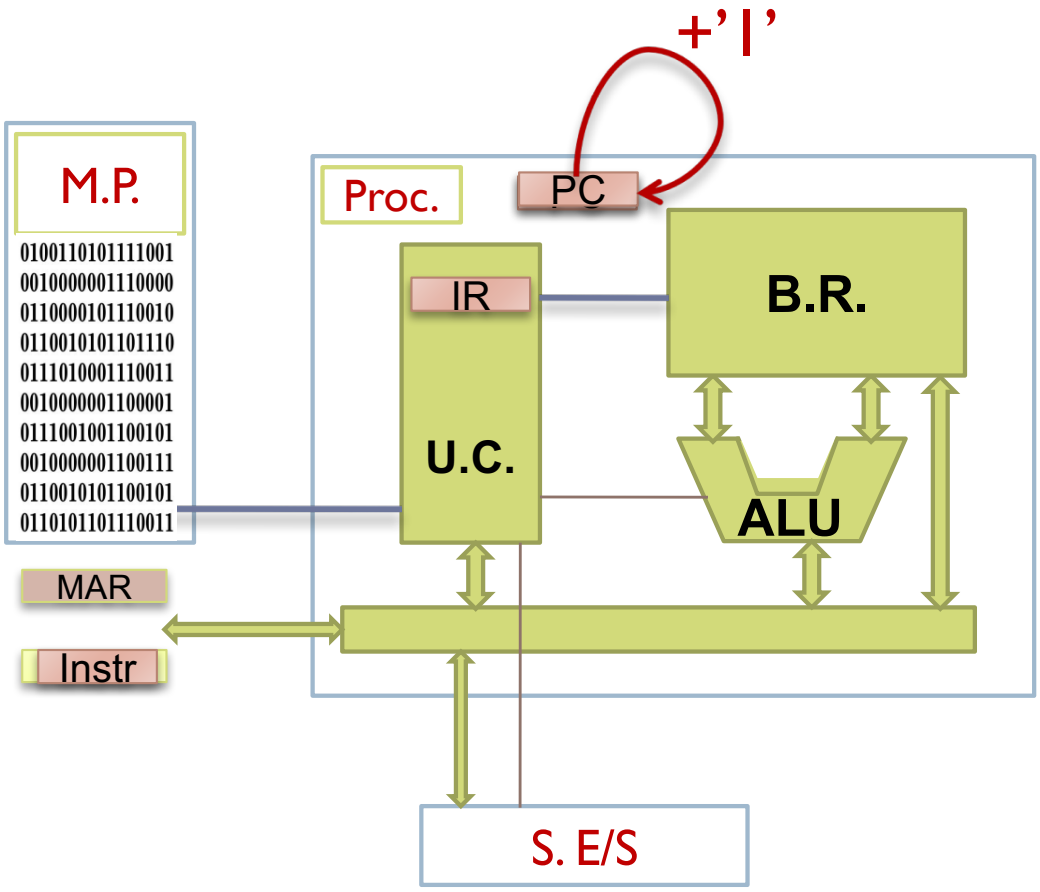
Funcionamiento básico de la UC

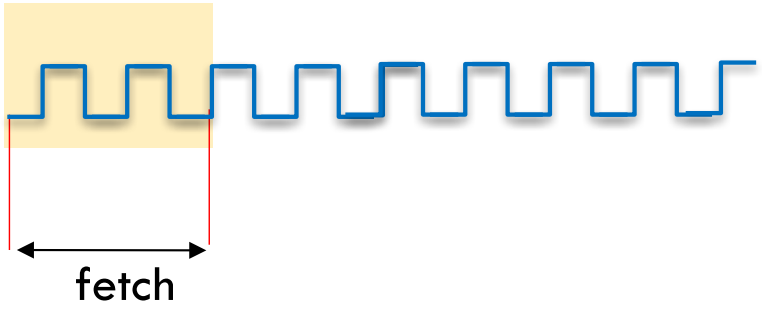




- Leer de memoria principal la instrucción apuntada por el PC

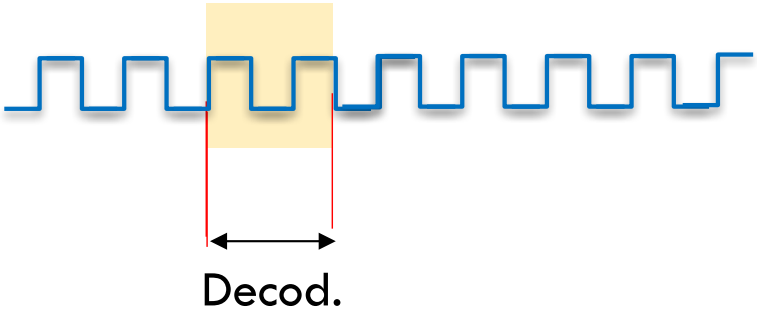
Funcionamiento básico de la UC

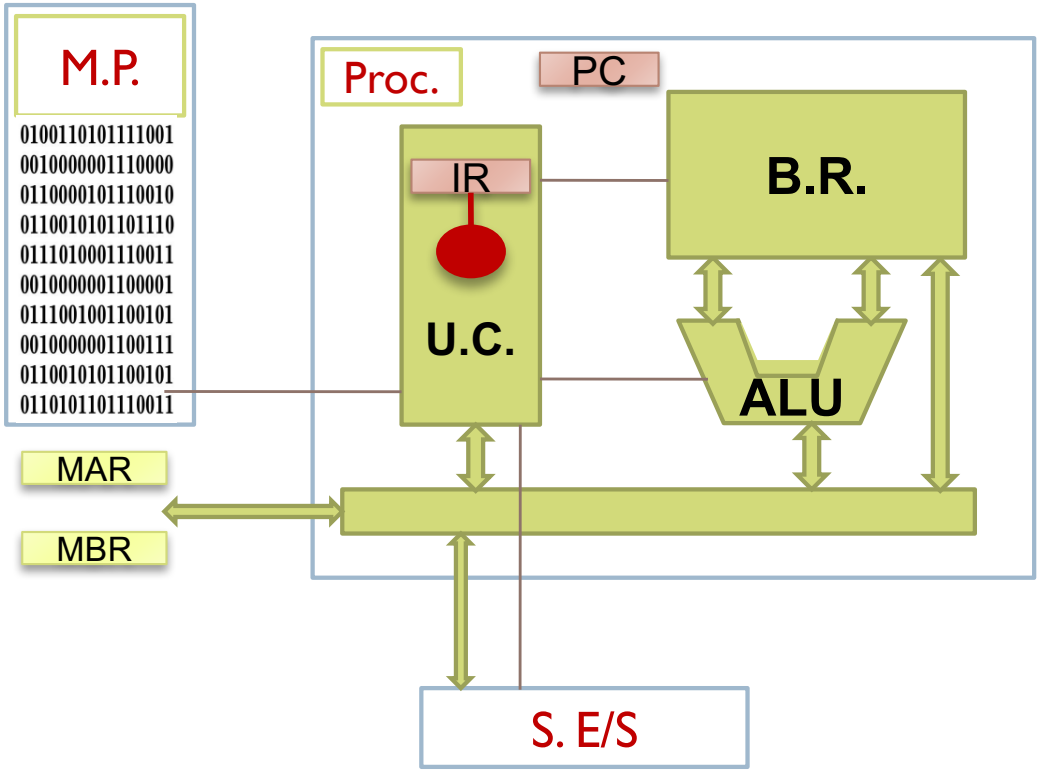




- Leer de memoria principal la instrucción apuntada por el PC
- Incrementar el PC
  - Siguiente dirección

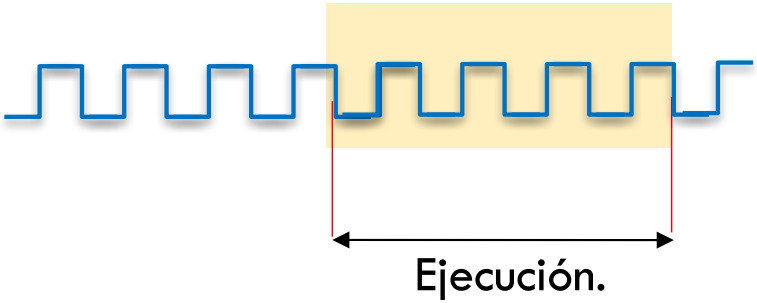
Funcionamiento básico de la UC

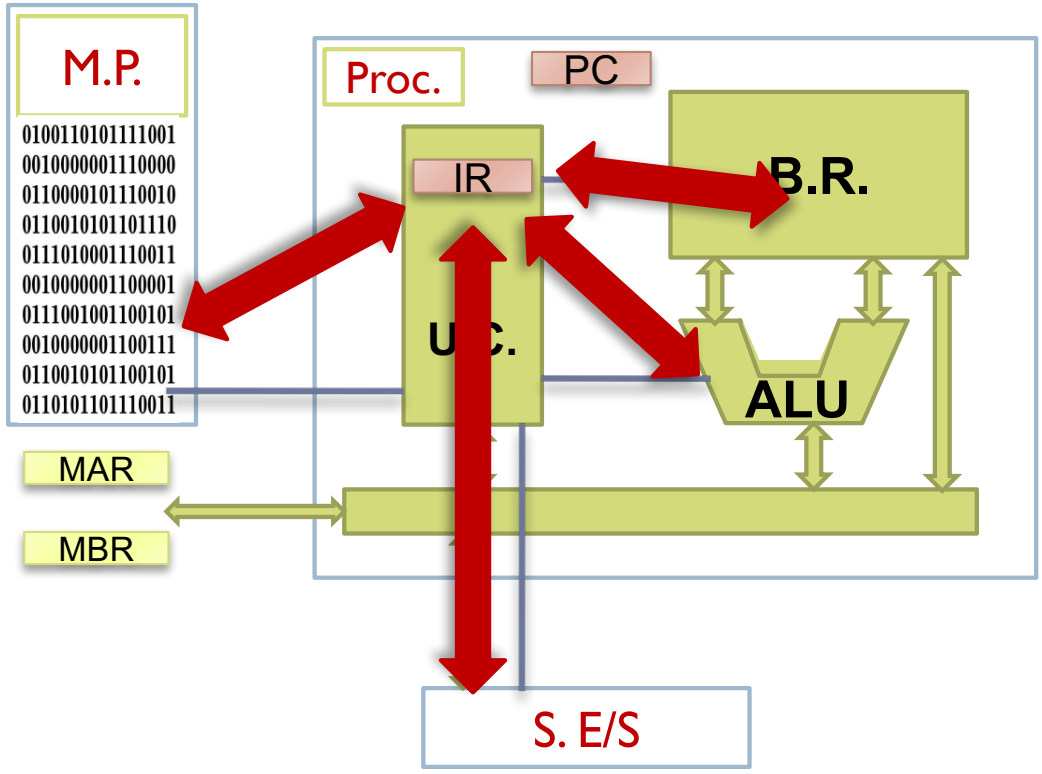




- Leer de memoria principal la instrucción apuntada por el PC
- Incrementar el PC
- Decodificación

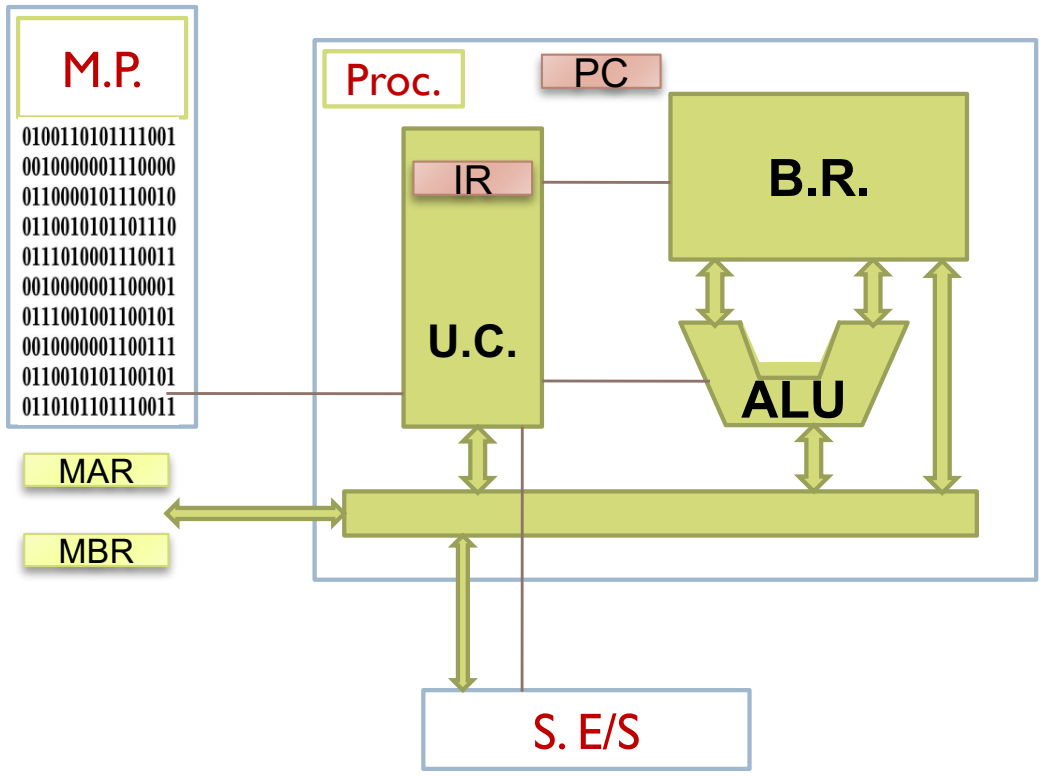
Funcionamiento básico de la UC





- Leer de memoria principal la instrucción apuntada por el PC
- Incrementar PC
- Decodificar instrucción
- Ejecución

Otras funciones de la UC

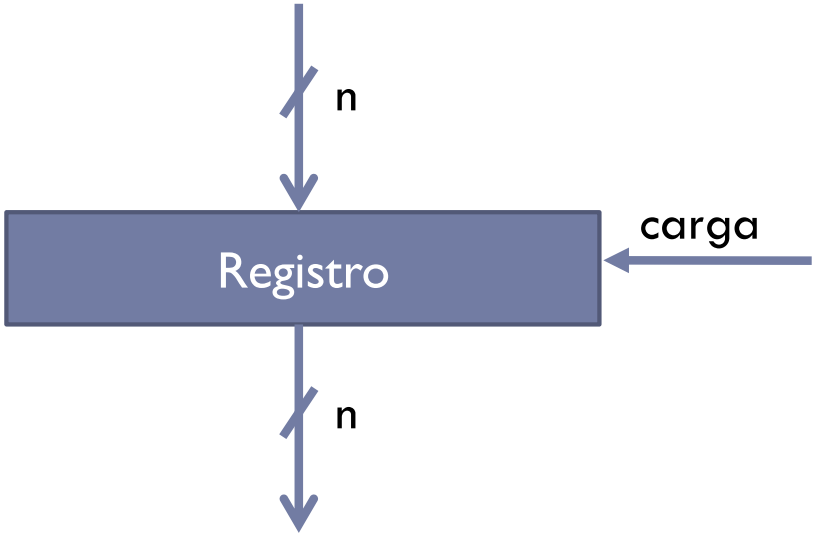


- Resolver situaciones anómalas
  - Instrucciones ilegales
  - Accesos a memoria ilegales
  - …
- Atender las interrupciones
- Controlar la comunicación con los periféricos

Componentes del procesador

- Banco de registros
- Unidad aritmético-lógica
- Unidad de control
- Memoria caché

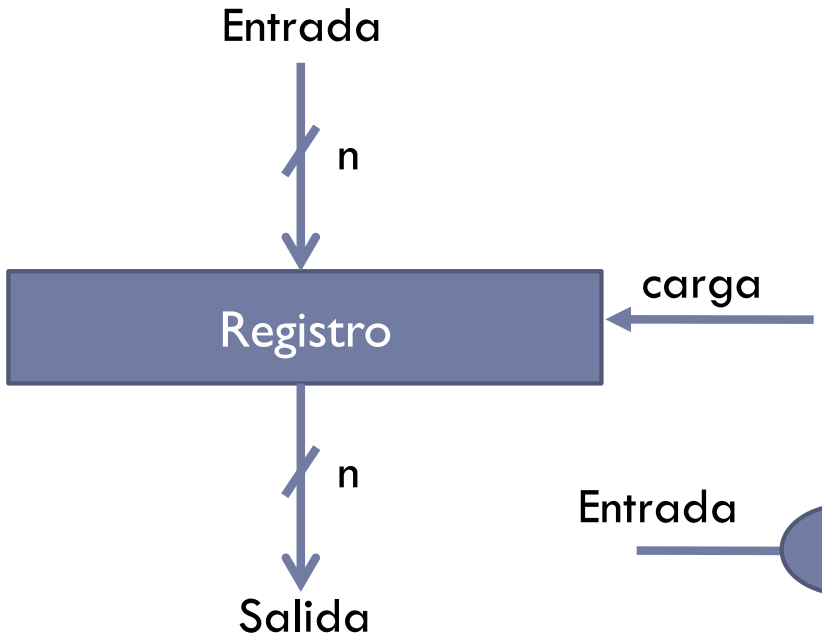
Registros



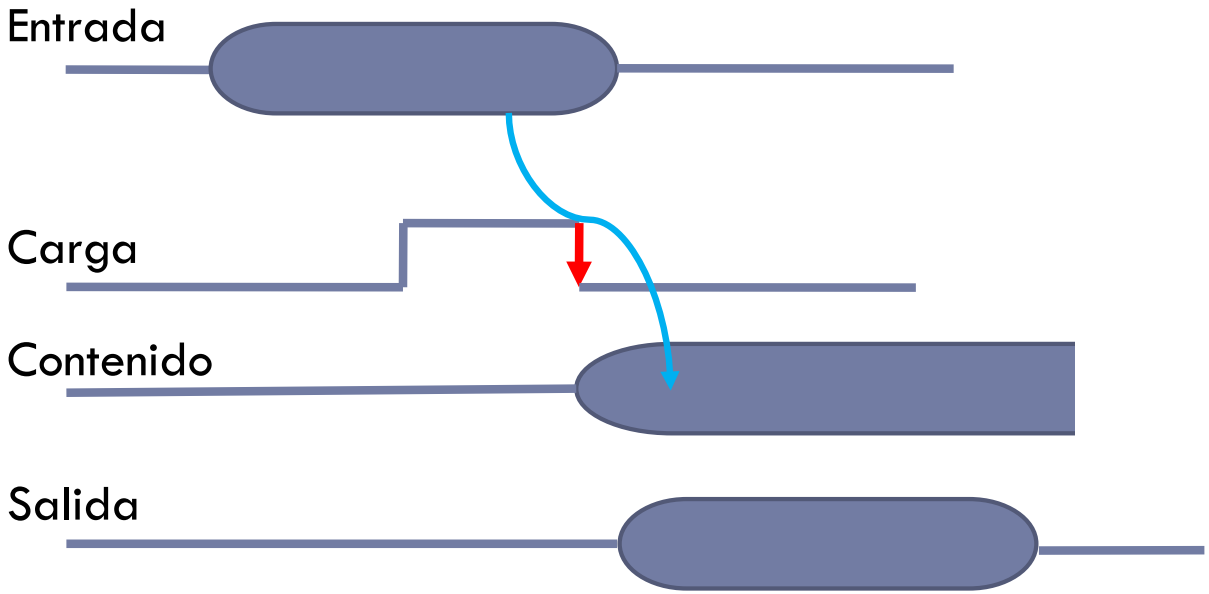
Elemento que almacena un conjunto de bits

Puede existir otra señal para poner el registro a cero

Registros



Puede existir otra señal para poner el registro a cero



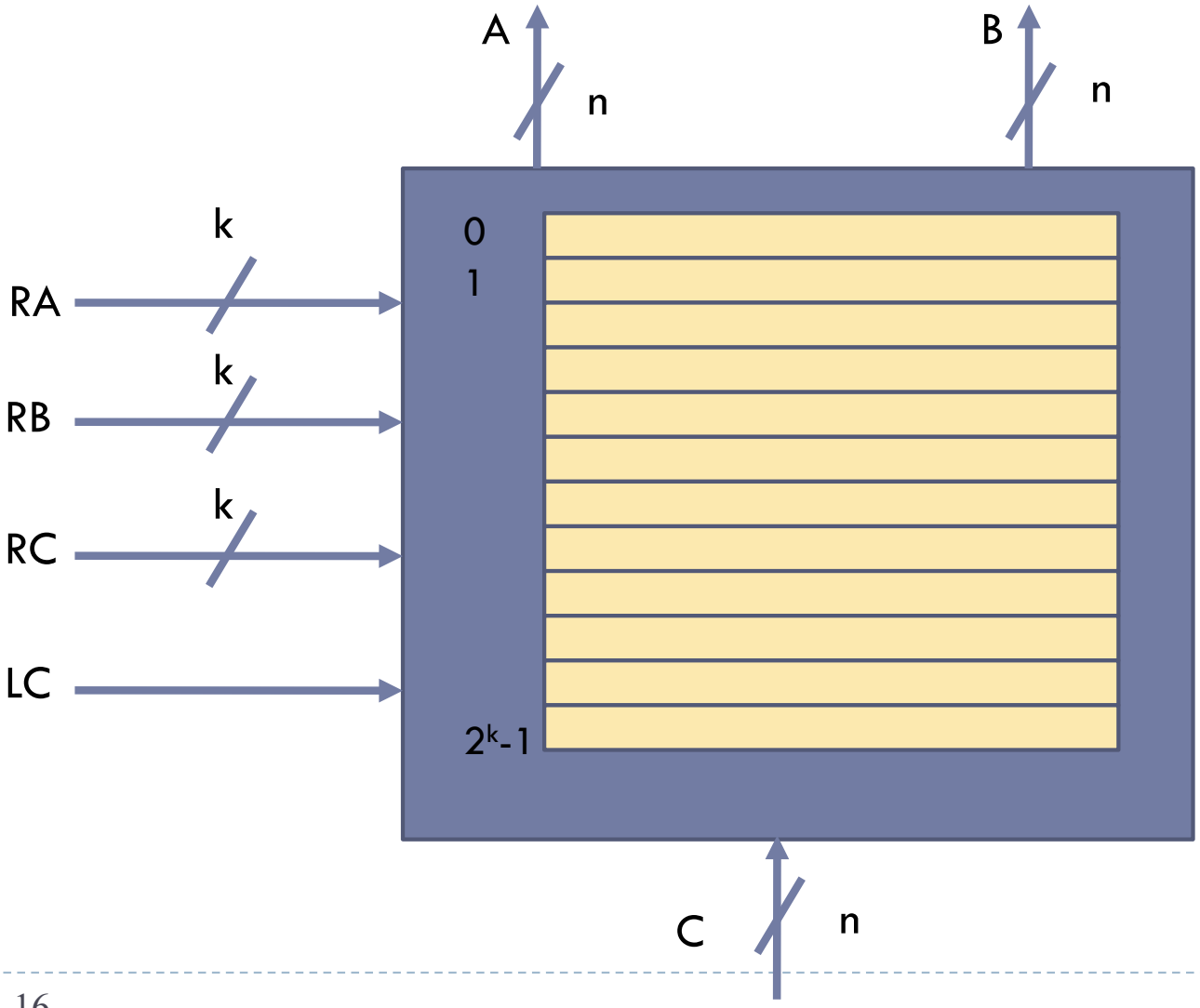
Tipos de registros

- Visibles al programador
  - En el MIPS: $t0, $t1, ….
- Registros no visibles
  - Registros temporales
- Registros de control y estado
  - Contador de programa, PC (*program counter*)
  - Registro de instrucción, IR (*instruction register*)
  - Registro de direcciones de memoria, MAR (*memory address register*)
  - Registro de datos de memoria, MBR (*memory buffer register*)
  - Registro de estado, SR (*status register*)

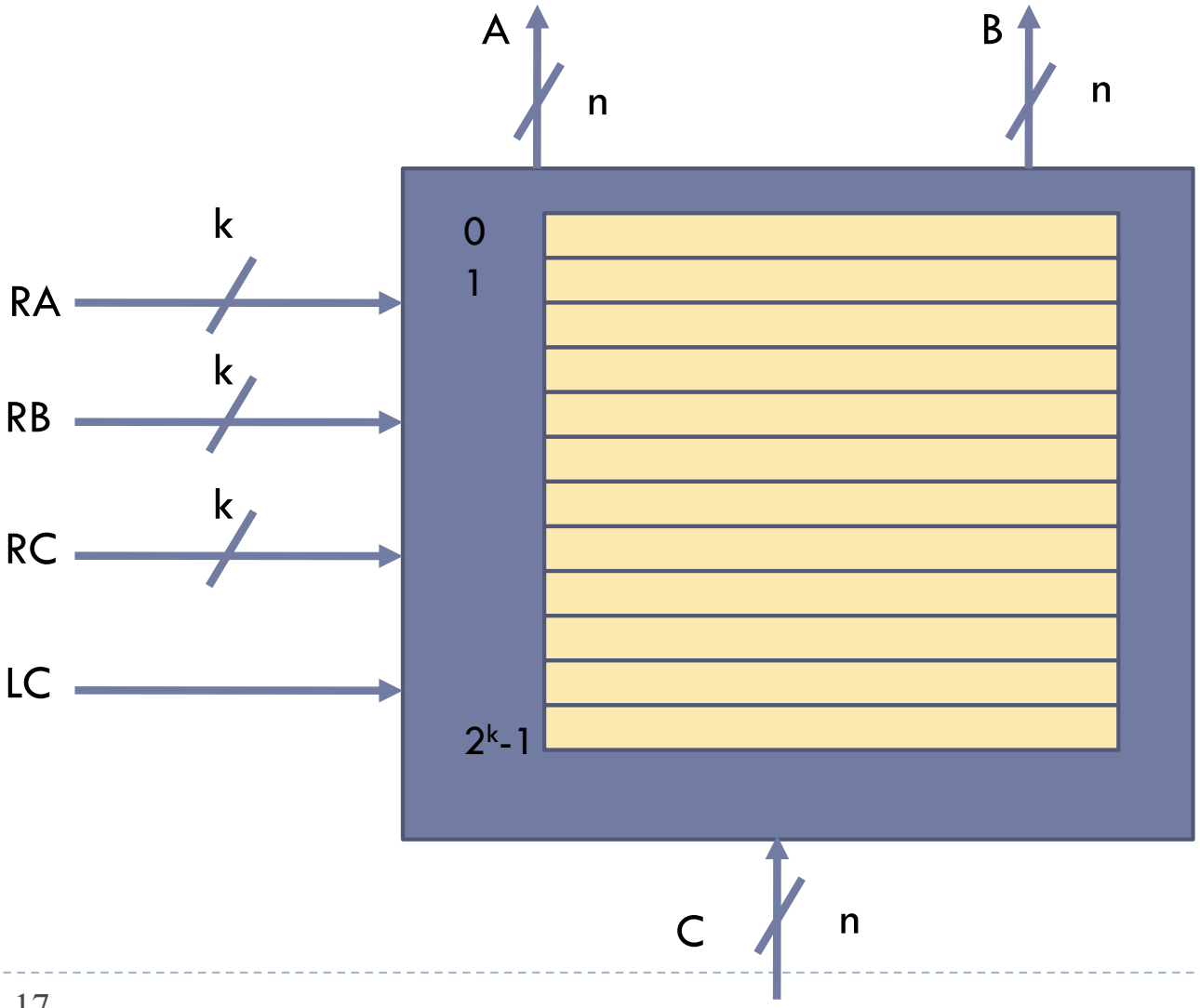
Banco de registros

- Agrupación de registros.
- Típicamente un número de registros potencia de 2.
  - n registros → $\log_2 n$ bits para seleccionar cada registro
  - k bits de selección → 2^k registros
- Elemento fundamental de almacenamiento.
  - Acceso muy rápido.

Banco de registros



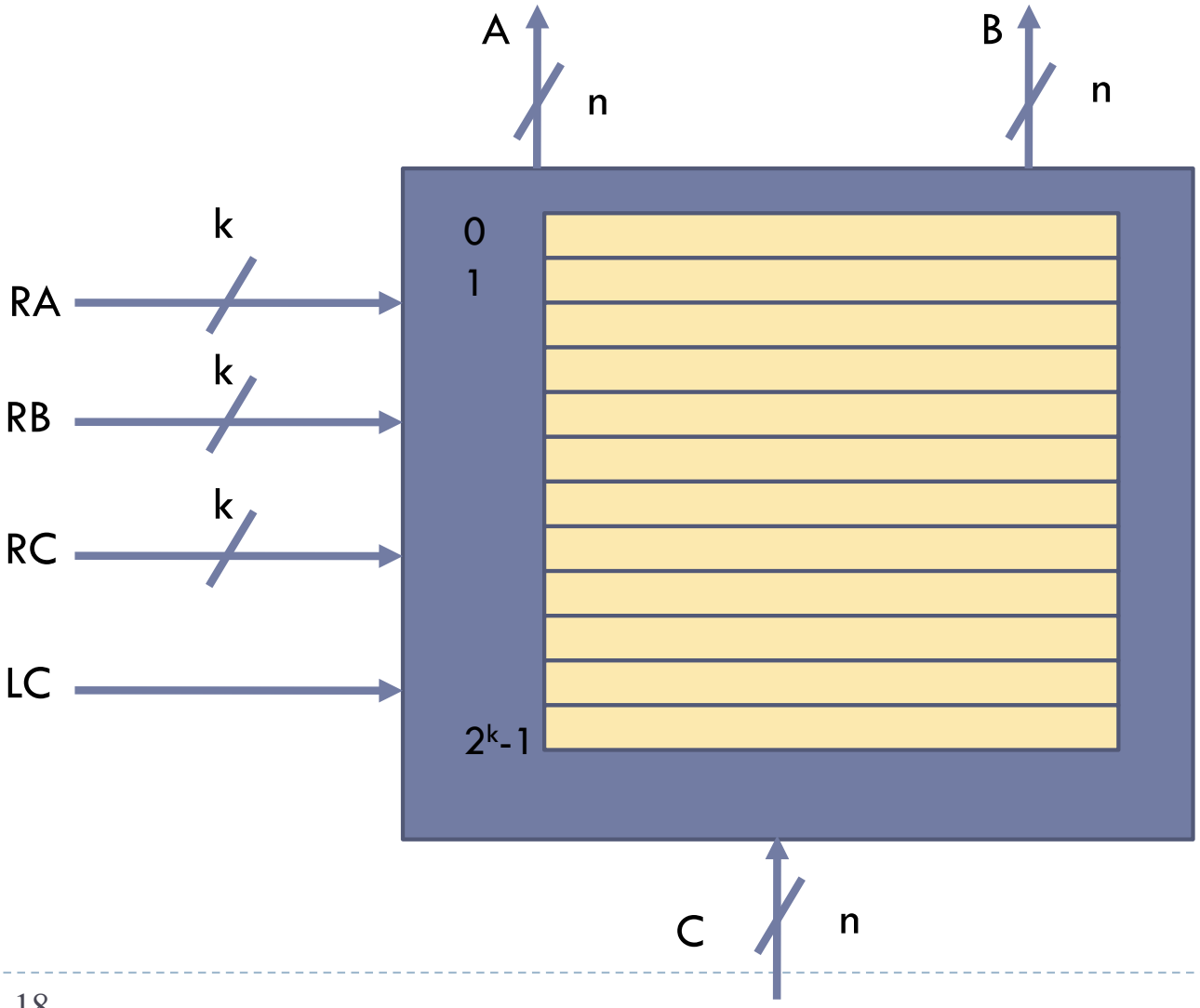
Banco de registros



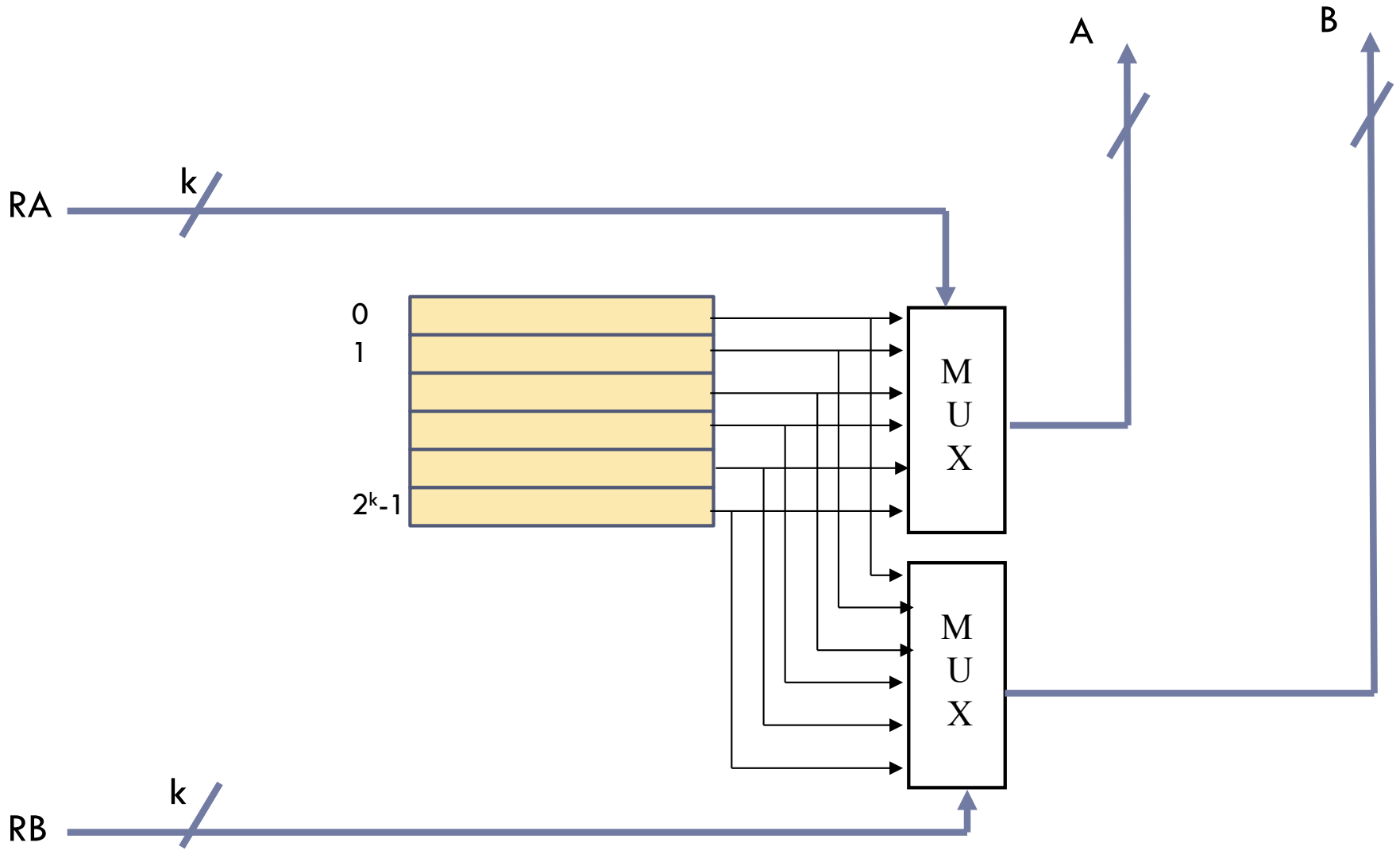
Con 32 registros, k=5

Banco de registros

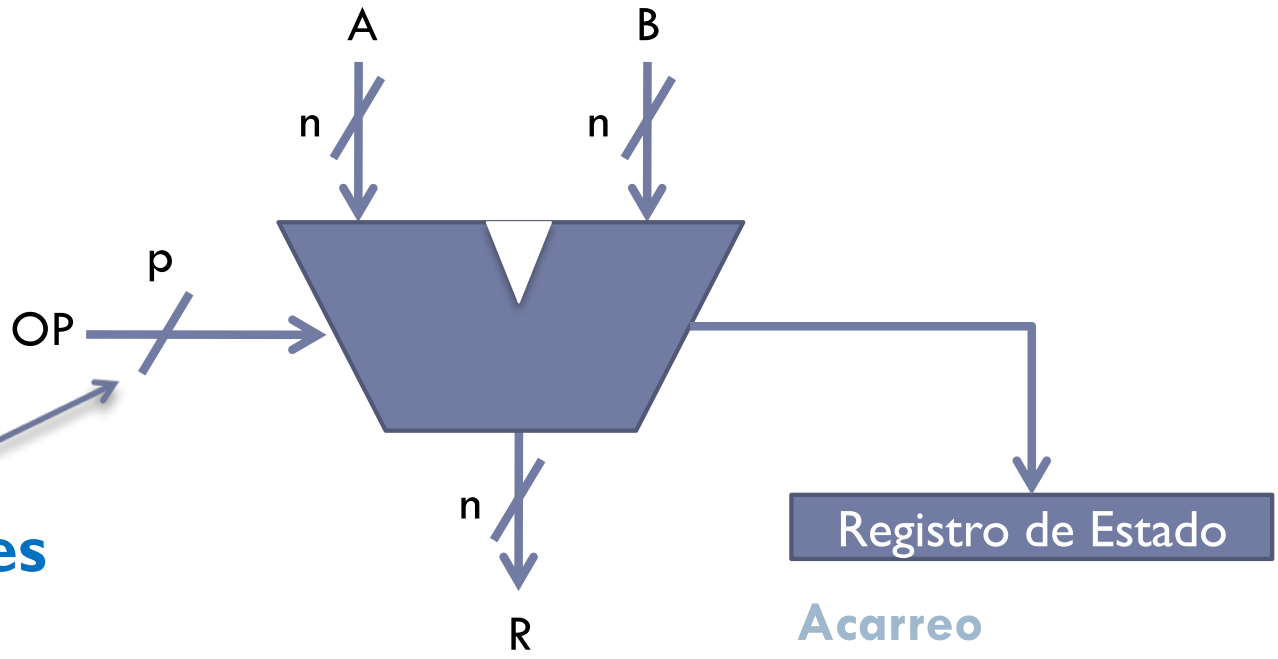
¿Qué valor tiene que tener RA para sacar por A el contenido del registro 14?



Esquema para lectura



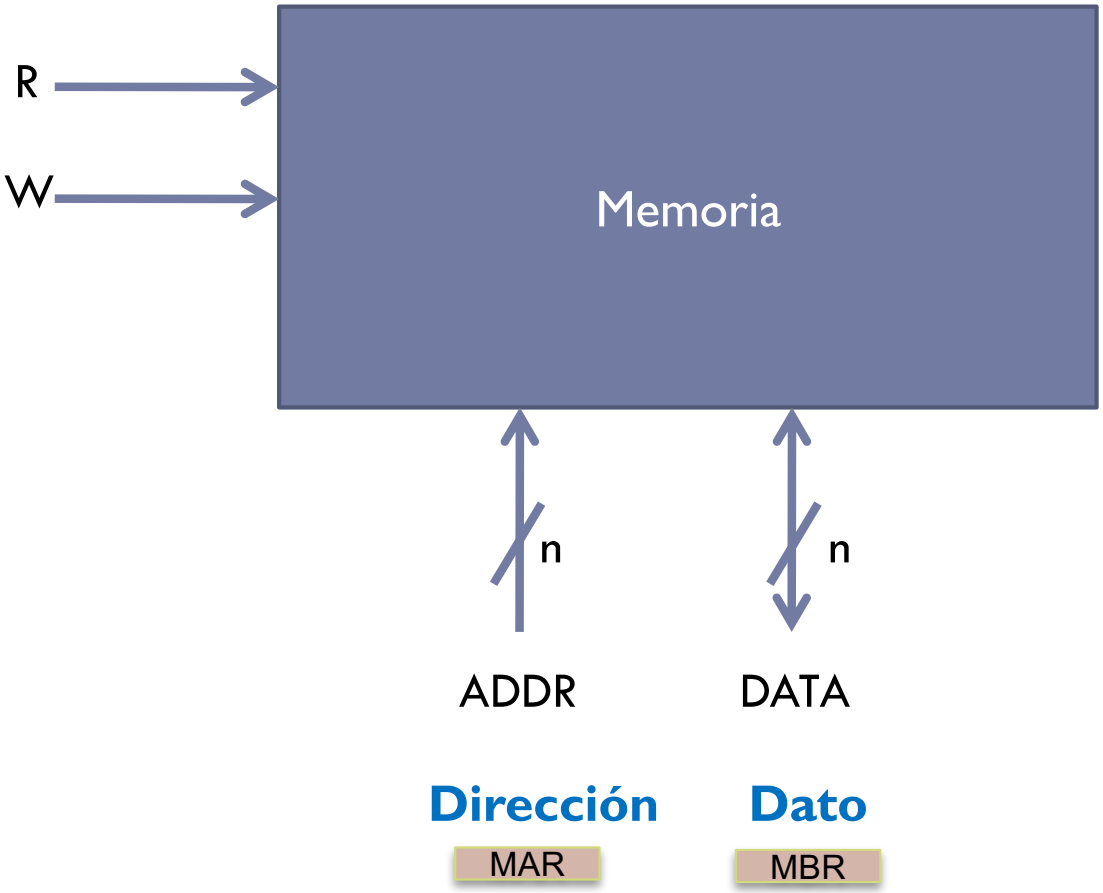
Unidad aritmético lógica



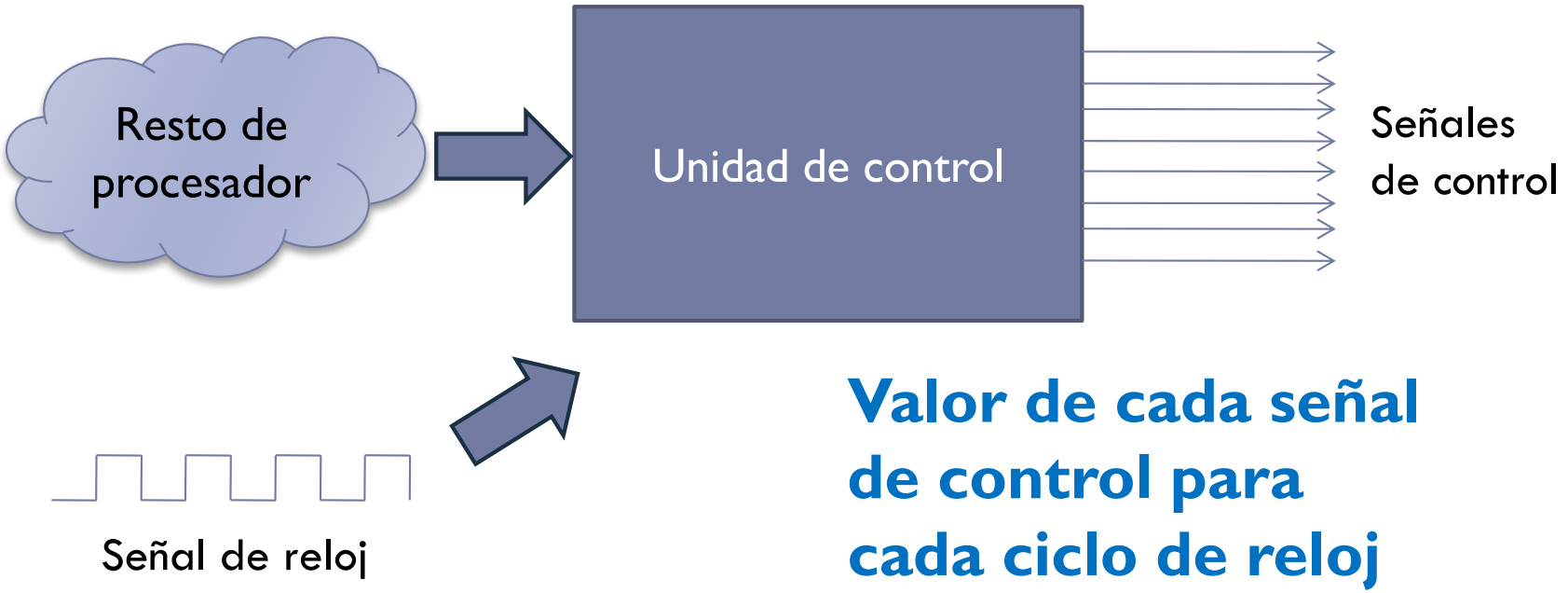
2^p operaciones aritméticas y lógicas distintas

- Acarreo
- Desbordamiento
- Cero
- Positivo
- Negativo
- …

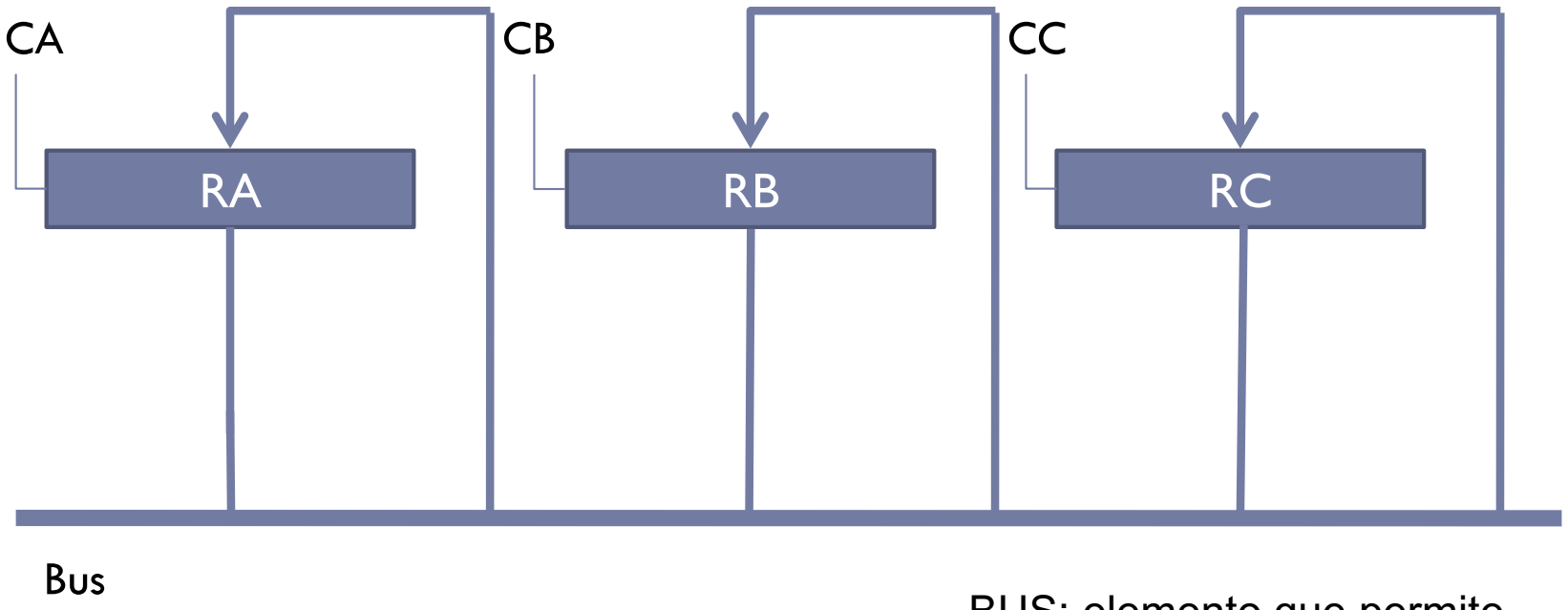
Acceso a la memoria



Unidad de control

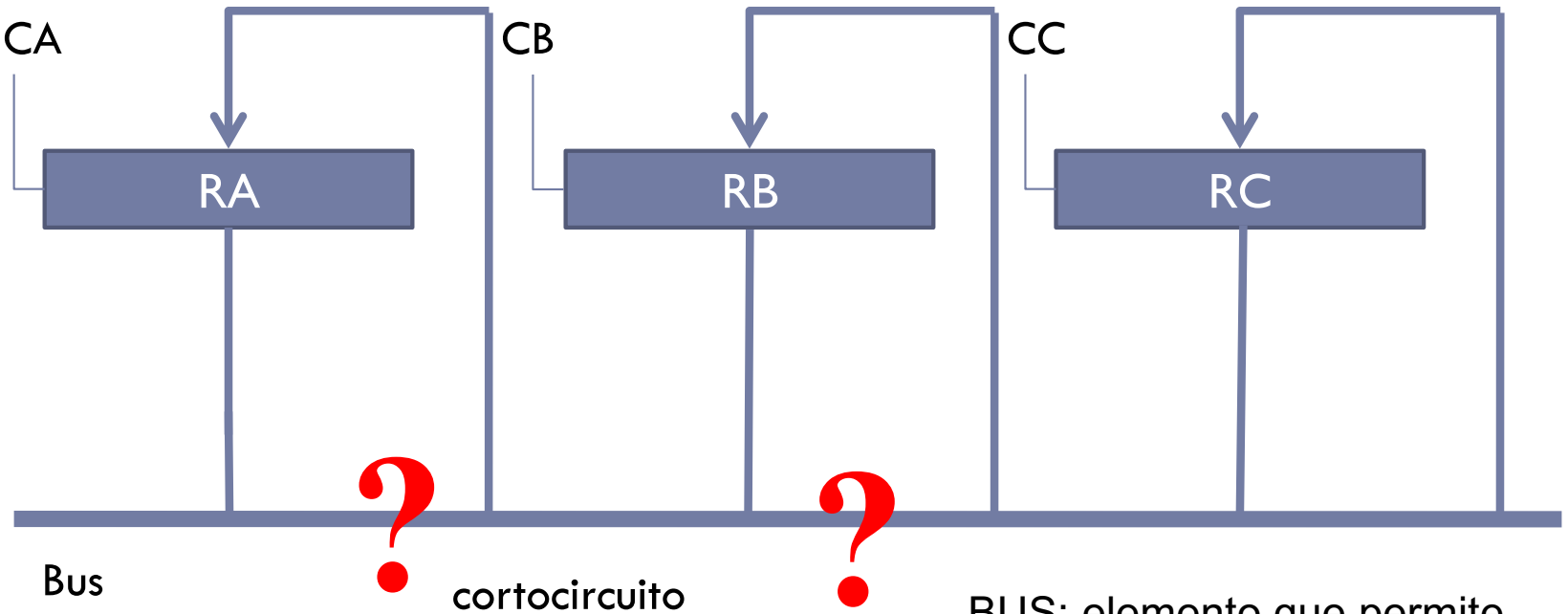


Conexión de registros a un bus



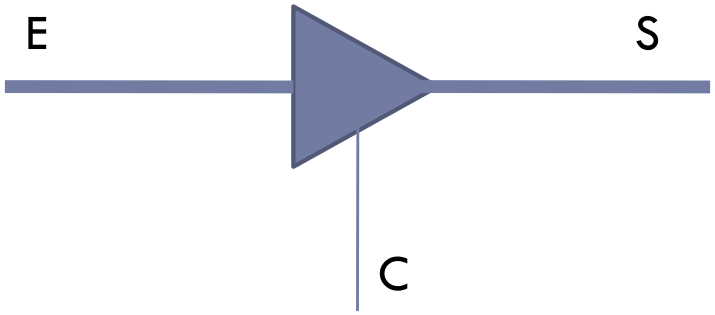
BUS: elemento que permite
Transmitir varios bits entre
Elementos de almacenamiento

Conexión de registros a un bus



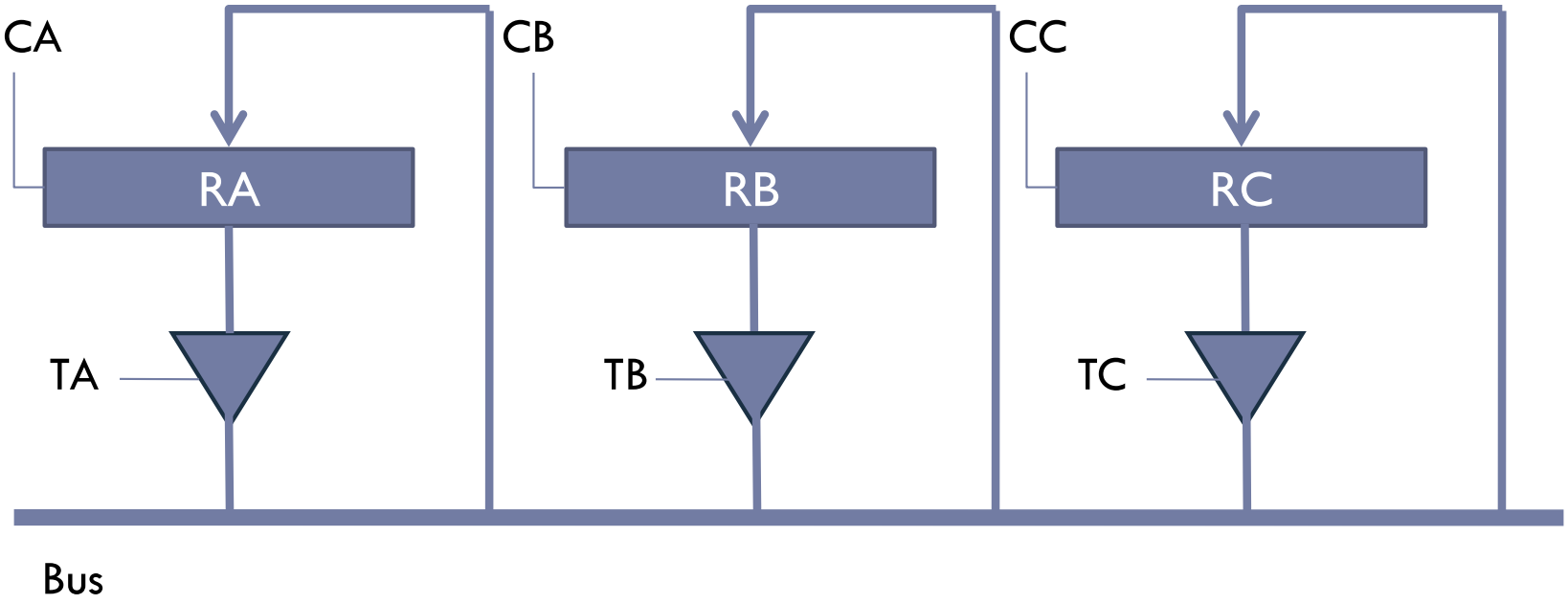
Búfer triestado

- Tipo especial de puerta lógica que puede poner su salida en alta impedancia (Z)
- Útil para permitir múltiples conexiones a un mismo punto

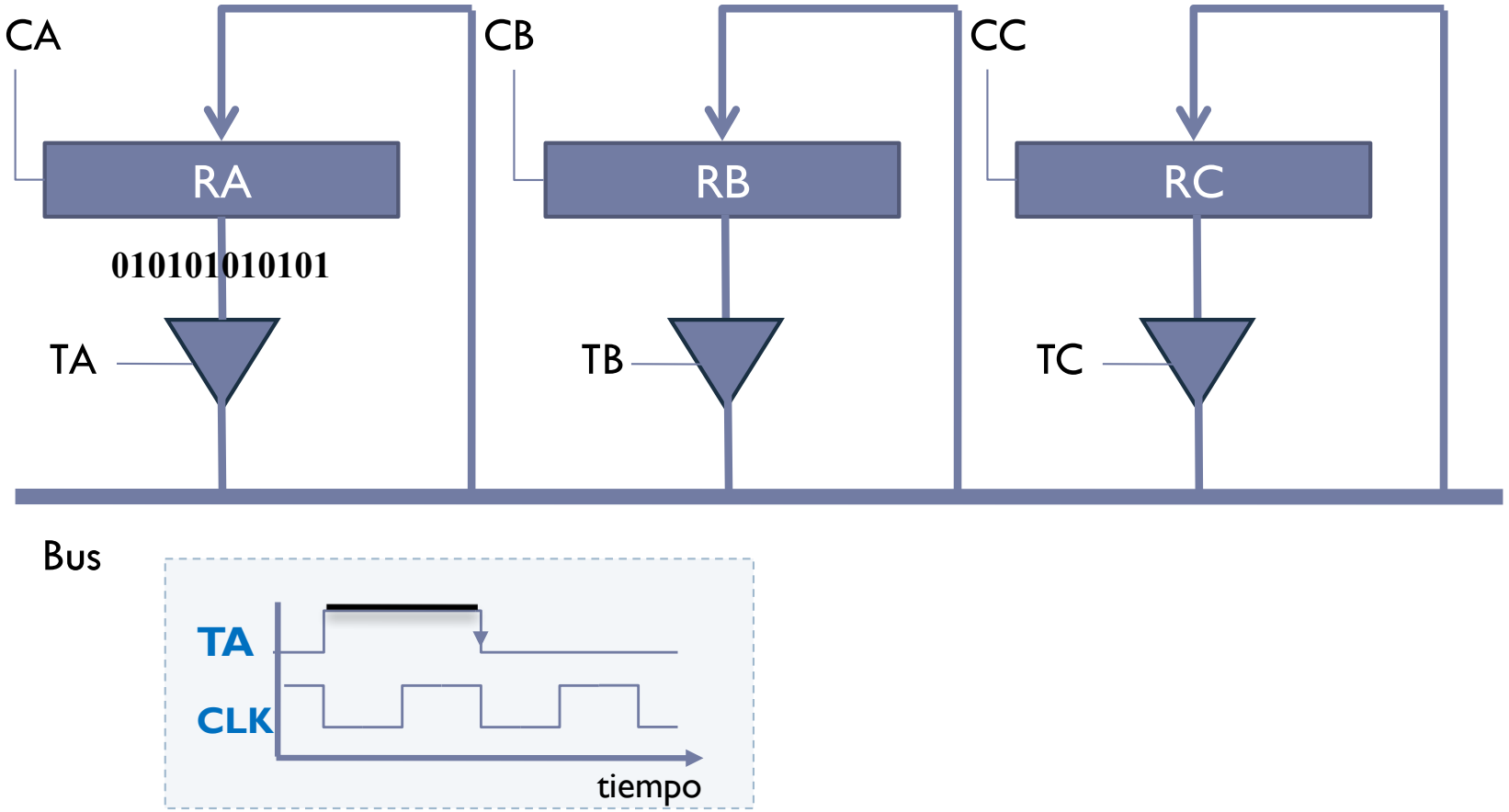


E	C	S
0	0	Z
1	0	Z
0	1	0
1	1	1

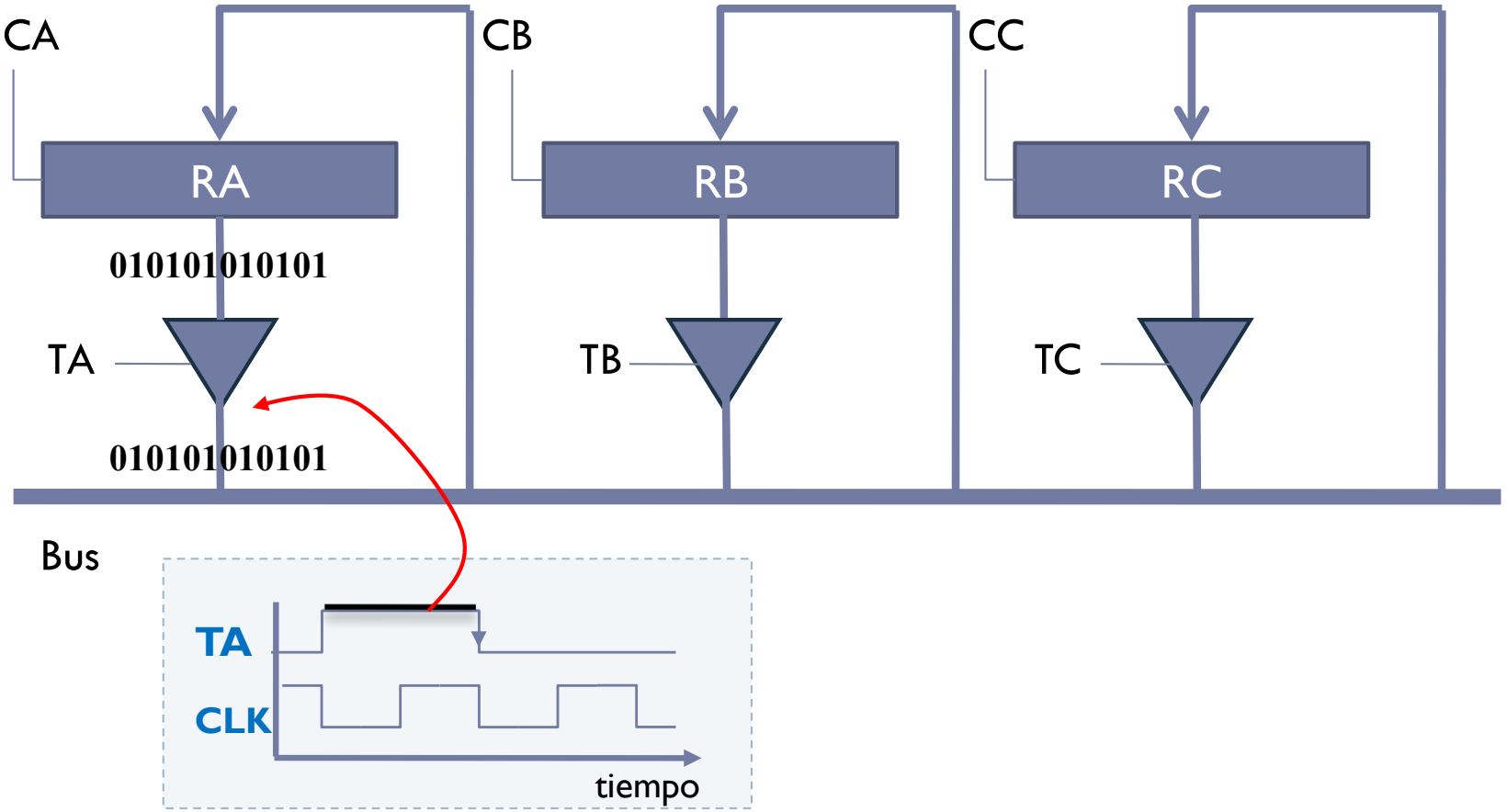
Acceso a un bus



Acceso a un bus

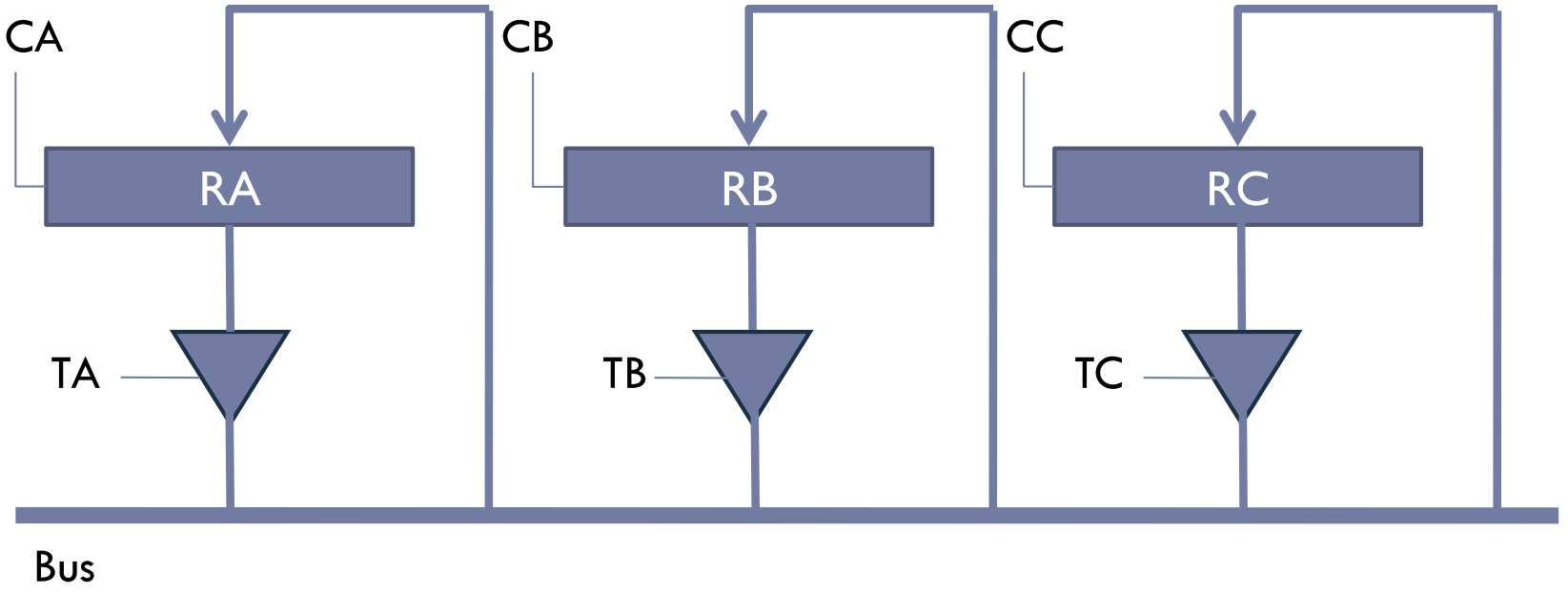


Acceso a un bus

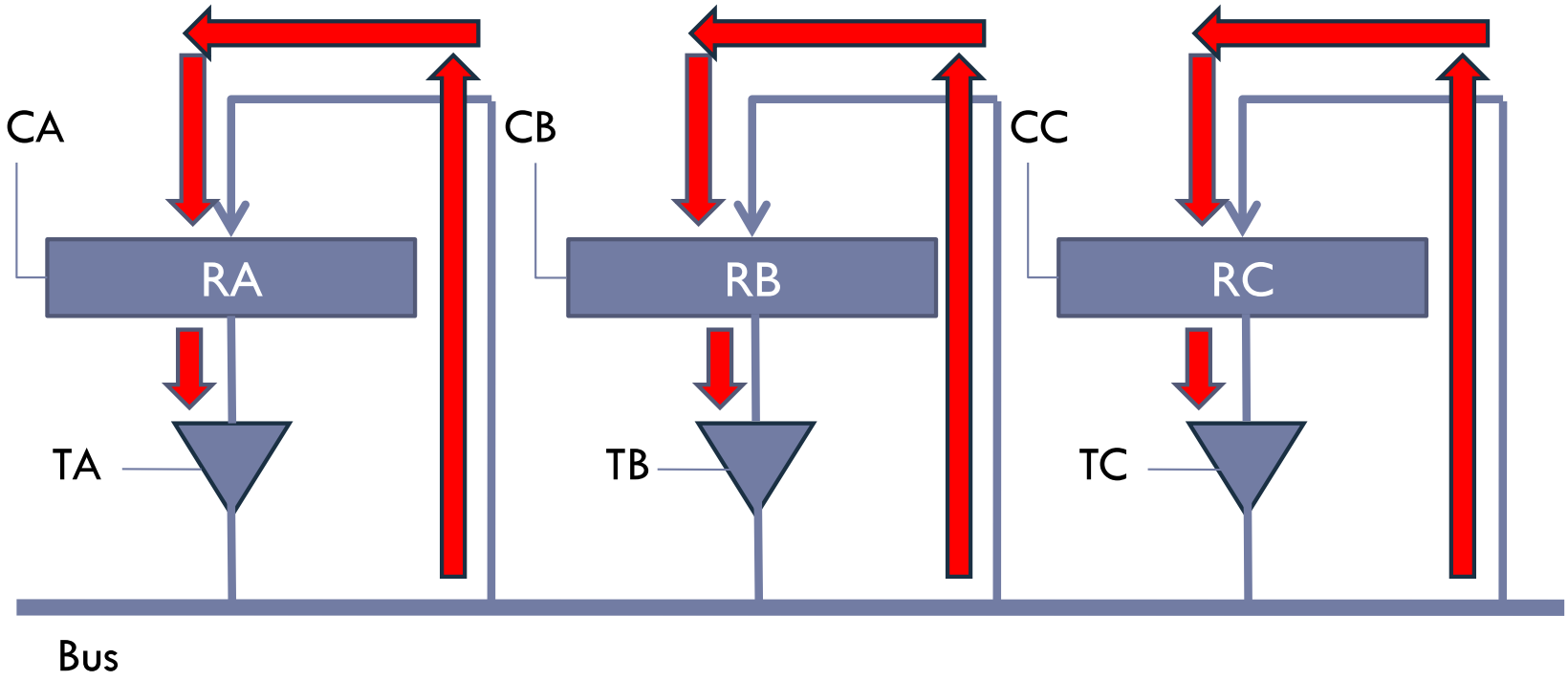


Ejemplo

- ¿Qué señales de control hay que activar para cargar el contenido de RA en RB?



Camino de datos (RB ← RA)

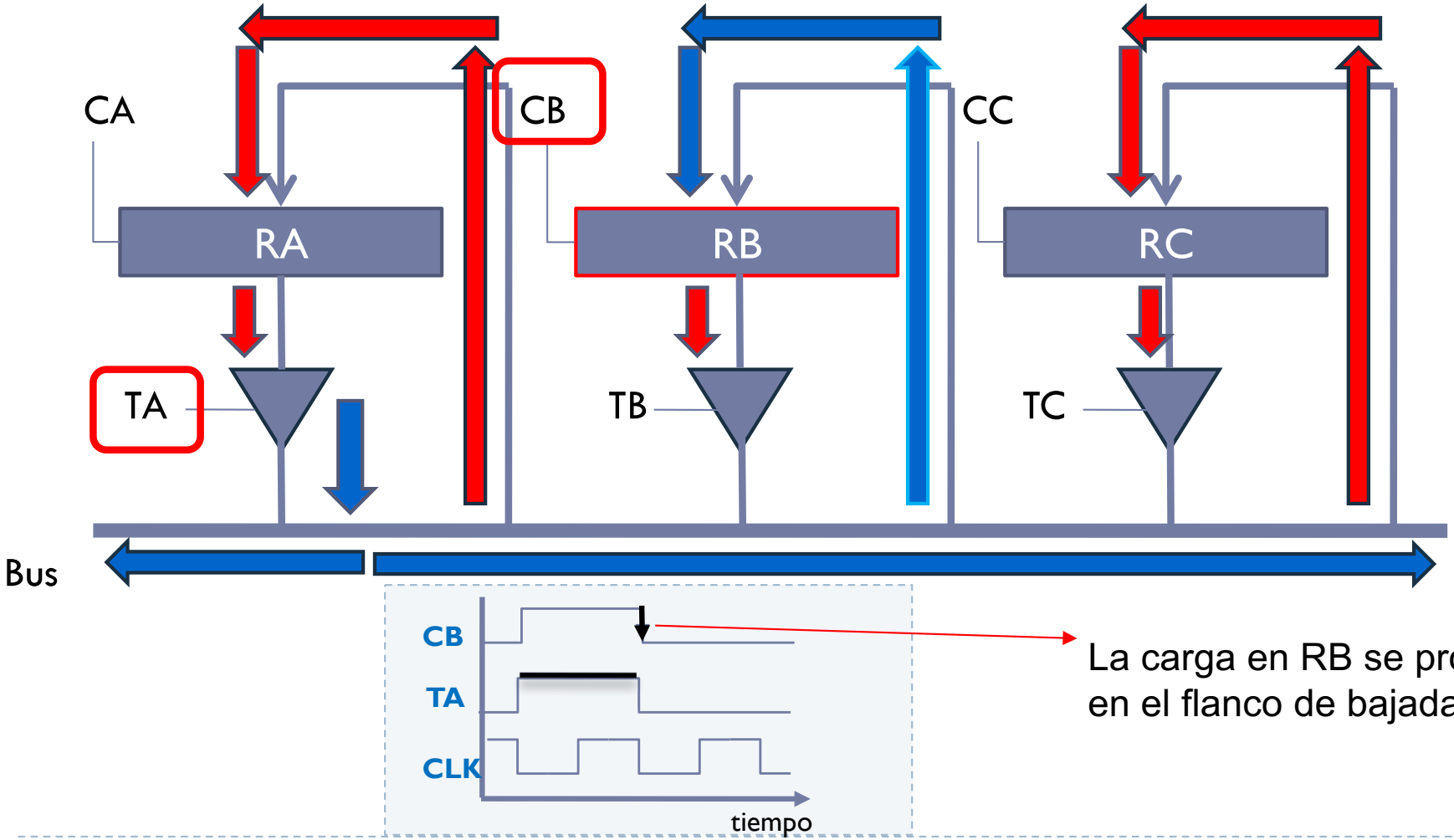


Situación con todas las señales desactivadas

Camino de datos (RB ← RA)

IMPORTANTE

No se puede activar dos o más triestados al mismo bus y al mismo tiempo



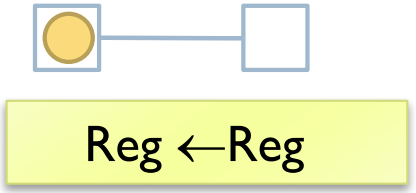
Lenguaje nivel RT

- Lenguaje de nivel de transferencia de registros.
  - Registro1 ← Registro2

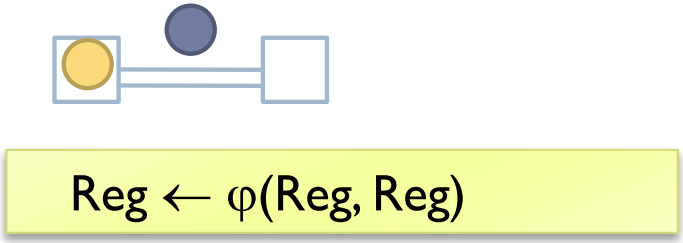
- Especifica lo que ocurre en el computador mediante transferencias de datos entre registros.

Operaciones elementales

- Operaciones de transferencia
  - MAR ← PC

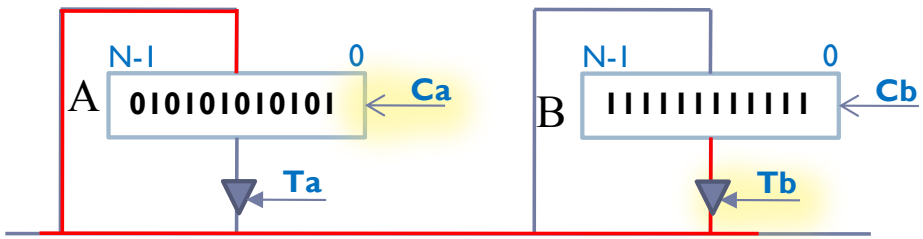


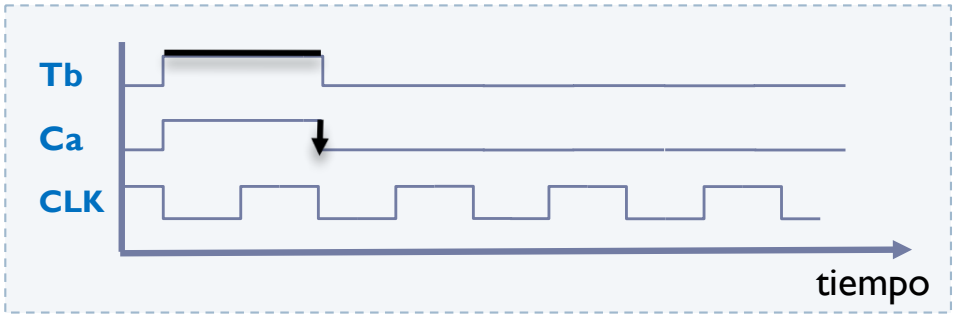
- Operaciones de proceso
  - R1 ← R2 + RT2



- Lenguaje RT
  - Lenguaje de nivel de transferencia de registros.
  - Especifica lo que ocurre en el computador mediante transferencias de datos entre registros.

Ejemplo de operación elemental de transferencia



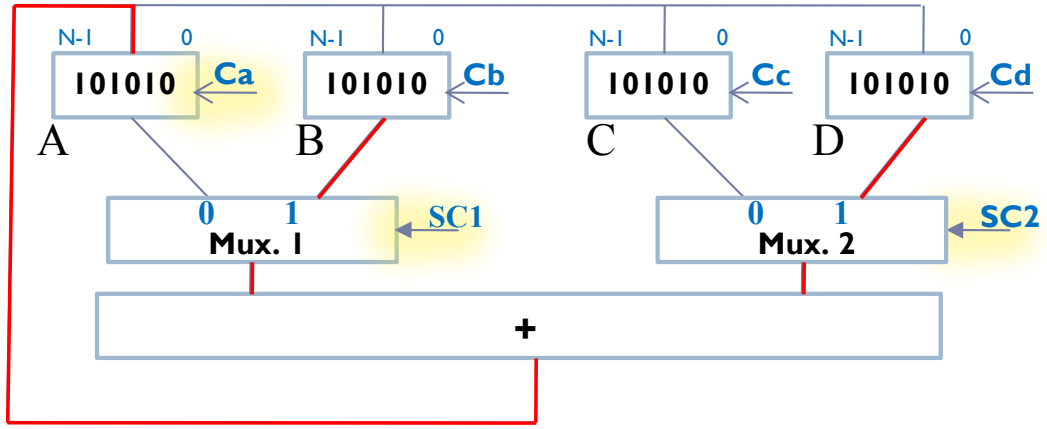


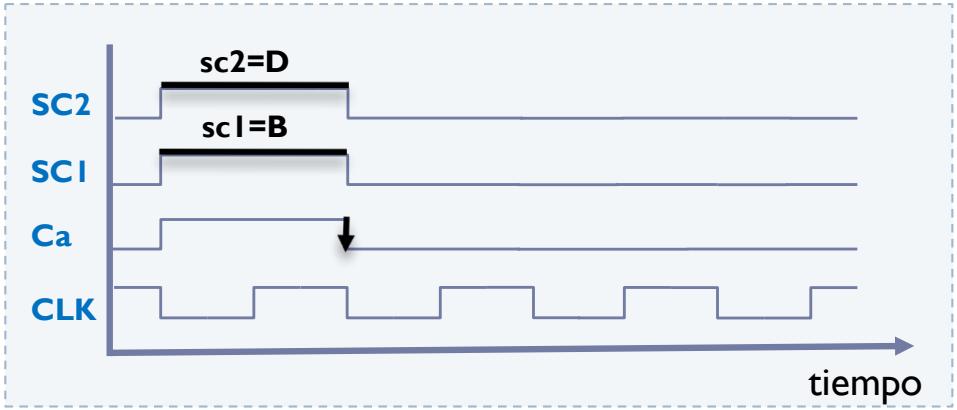
- **Operación elemental** de **transferencia**:
  - Elemento de almacenamiento origen
  - Elemento de almacenamiento destino
  - Se establece un camino

xx: A ← B [Tb, Ca]

- IMPORTANTE
  - Establecer el camino entre origen y destino en un mismo ciclo
  - En un mismo ciclo NO:
    - se puede atravesar un registro

Ejemplo de operación elemental de procesamiento





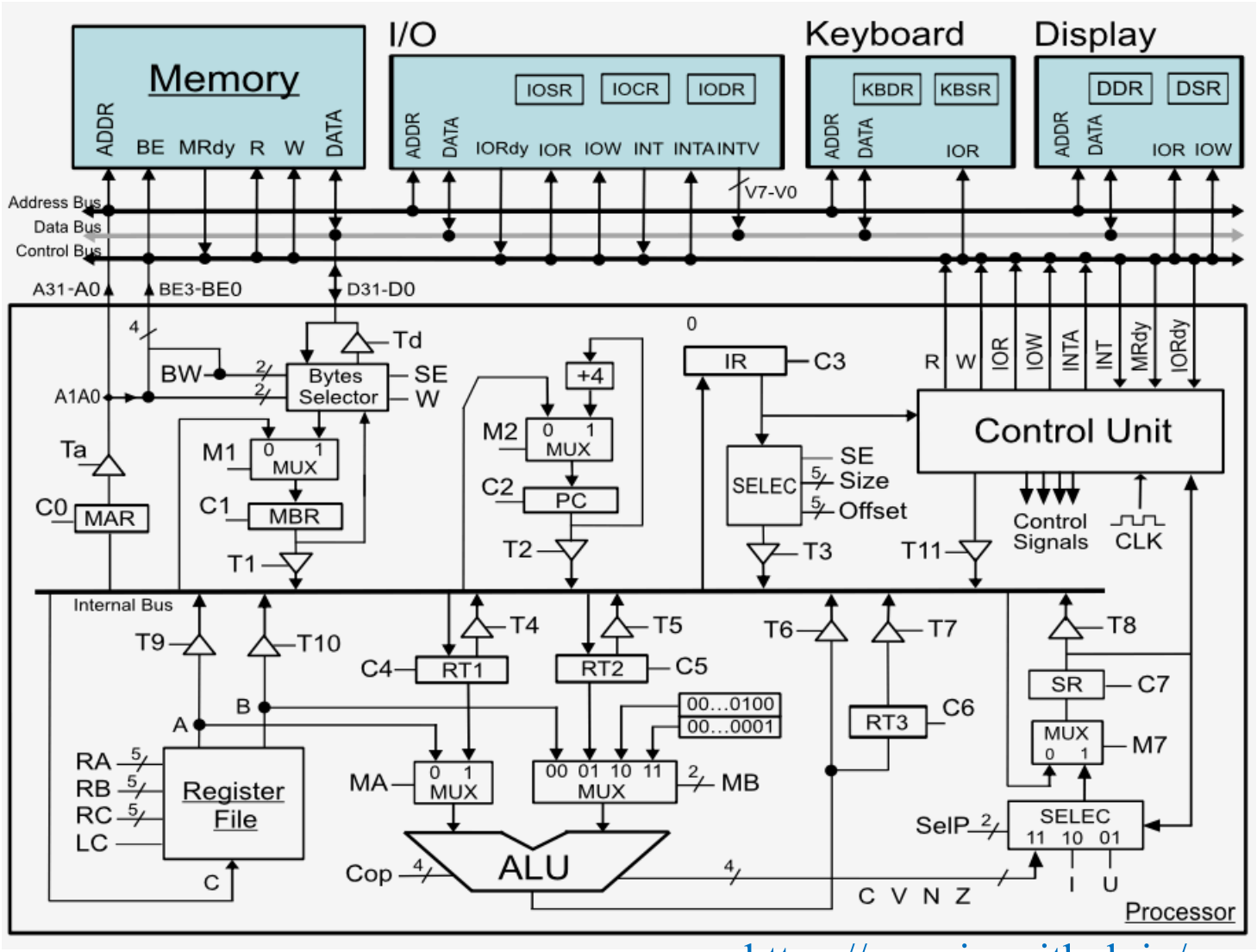
- **Operación elemental de procesamiento**:
  - Elemento(s) de origen
  - Elemento destino
  - Operación de transformación en el camino

yy: A ← B+D [SC1=b,SC2=d, Ca]

- IMPORTANTE
  - Establecer el camino entre origen y destino en un mismo ciclo
  - En un mismo ciclo NO se puede atravesar un registro

Estructura de un computador elemental

Simulador WepSIM

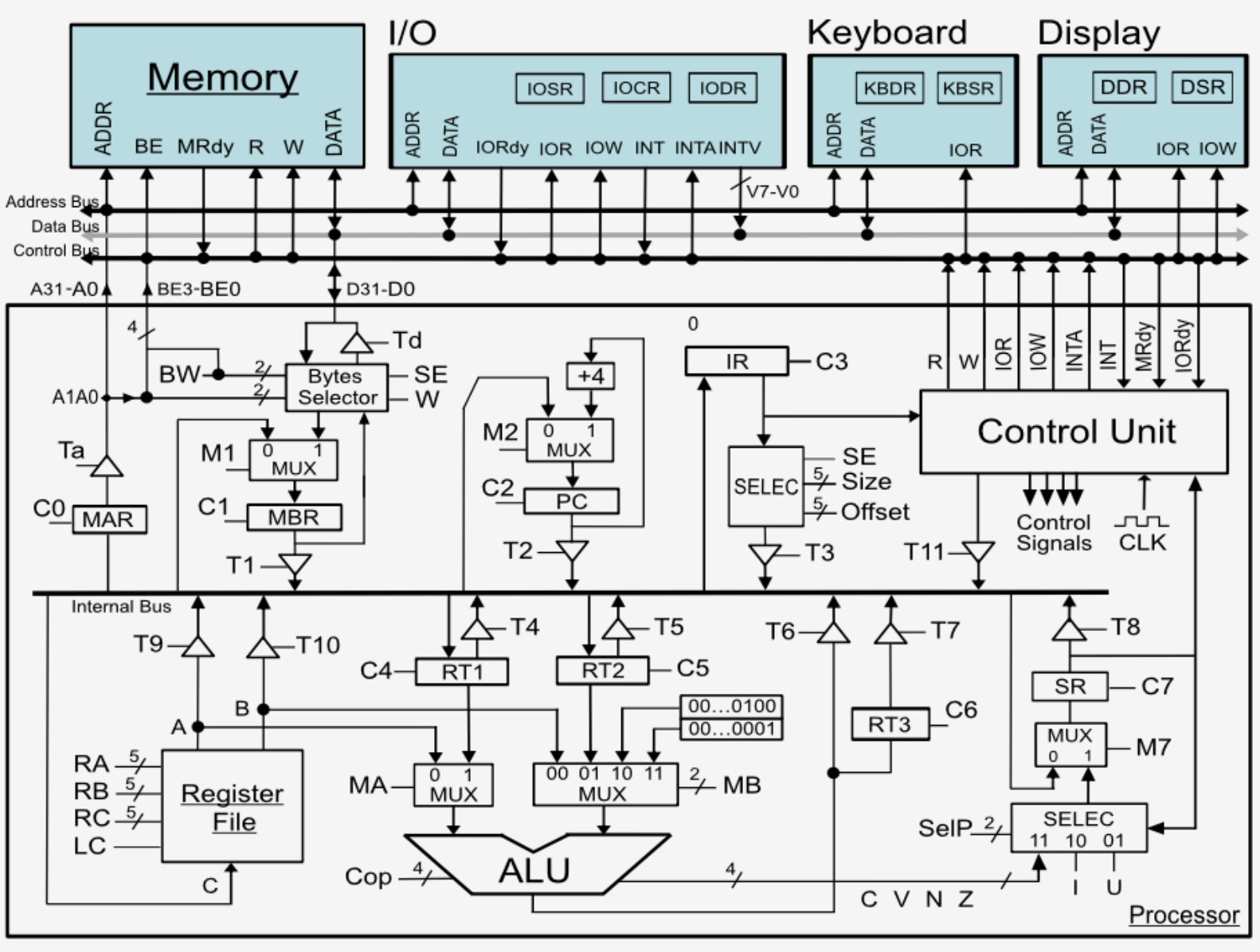


https://wepsim.github.io/wepsim/

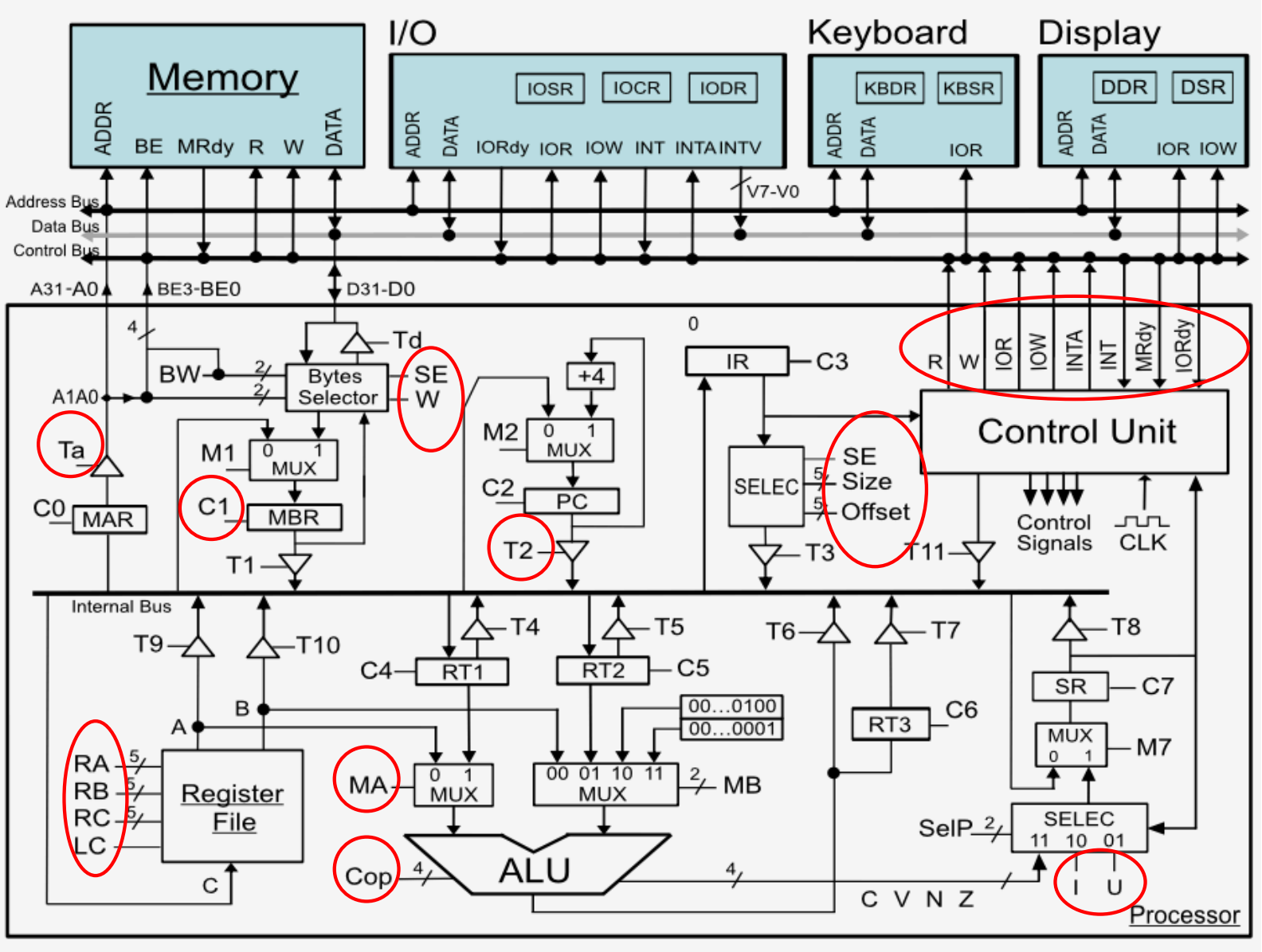
Características

- Computador de 32 bits
- La memoria se direcciona por por bytes
  - Un ciclo para las operaciones de lectura y escritura
- Banco de 32 registros visibles
  - R0..R31
  - Asumir como en el MIPS:
    - R0 = 0 y SP = R29
- Registros no visibles al usuario
  - RT1, RT2, RT3: no visibles
- Otros registros de control y estado
  - MAR, MBR, PC, SR, IR
- Simulador WepSIM
  - https://wepsim.github.io/wepsim/

Estructura de un computador elemental



Señales de control



Señales de control

- Señales de acceso a memoria
- Señales de carga en registros
- Señales de control de las puertas triestado
- Señales de selección de los MUX
- Señales de control del banco de registros (RA, RB, RC y LC)
- Otras señales de selección

Nomenclatura:

- Ry: Selección de registros del banco de registros
- Mx: Selección en multiplexor
- Tx: Señal de activación triestado
- Cx: Señal de carga de registro

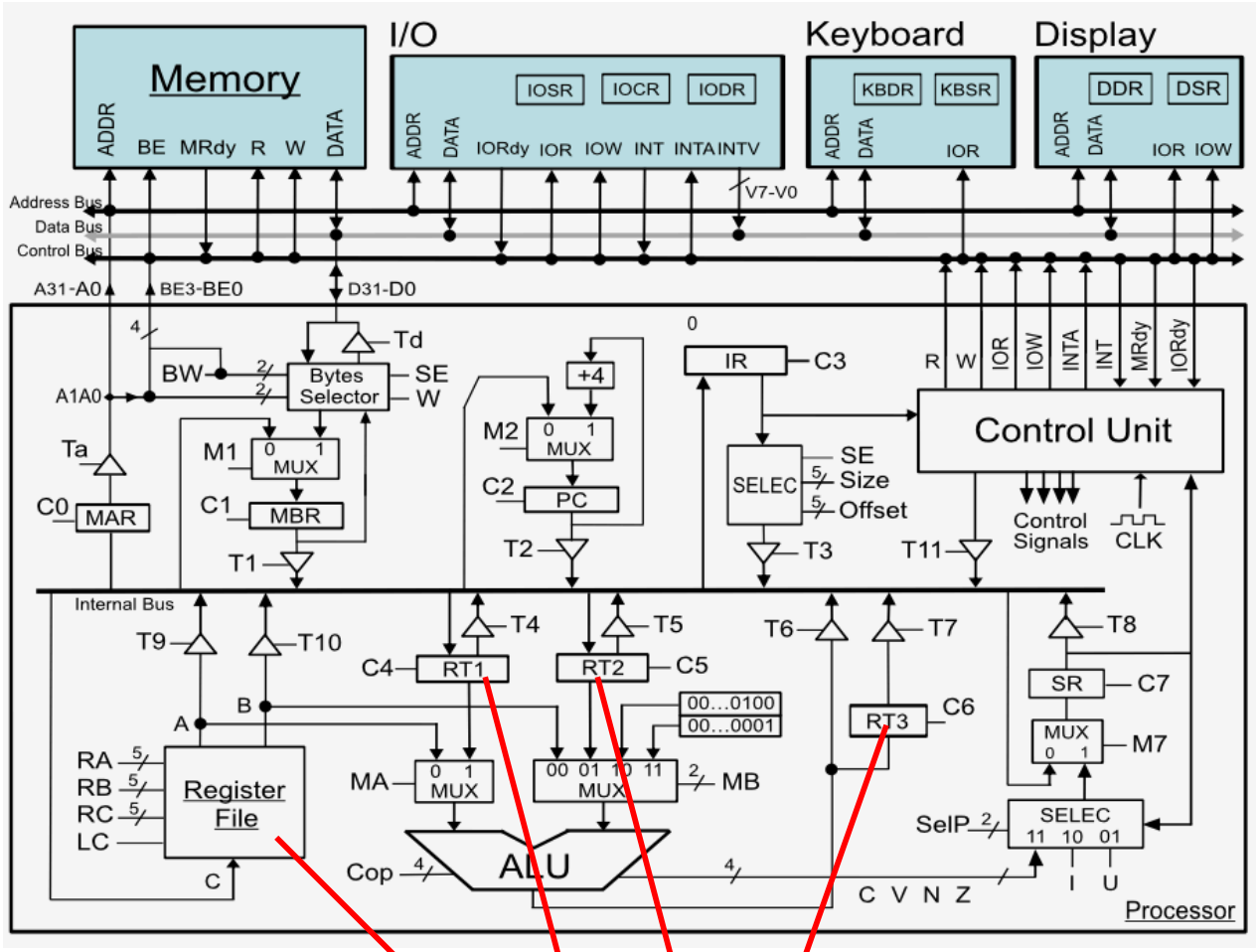
Registros

- Registros del banco de registros

- Registros de control y estado:
  - PC: contador de programa
  - IR: registro de instrucción
  - SP: puntero de pila (en el banco de registros)
  - MAR: registro de direcciones de memoria
  - MBR: registro de datos de memoria
  - SR: registro de estado

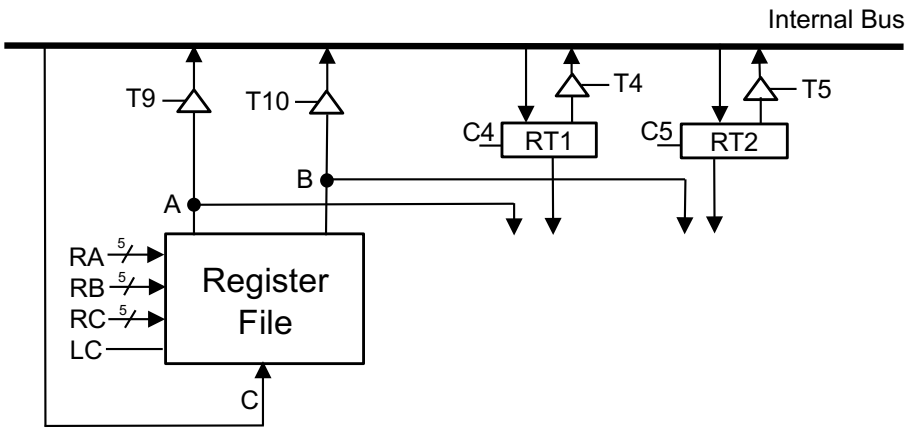
- Registros no visibles al usuario: RT1, RT2, RT3.

Estructura de un computador elemental



Banco de registros y
registros auxiliares (RT1, RT2 y RT3)

Señales de control



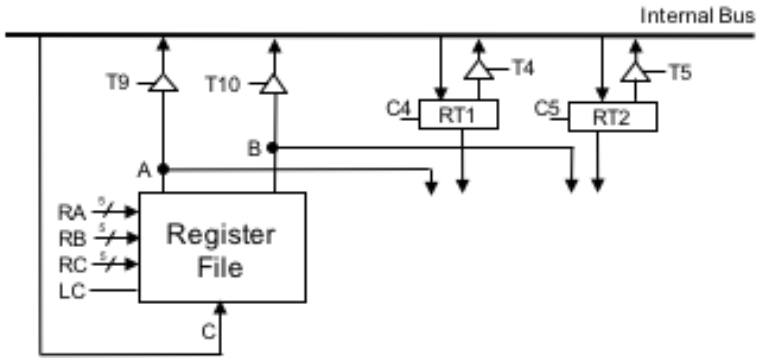
Nomenclatura:

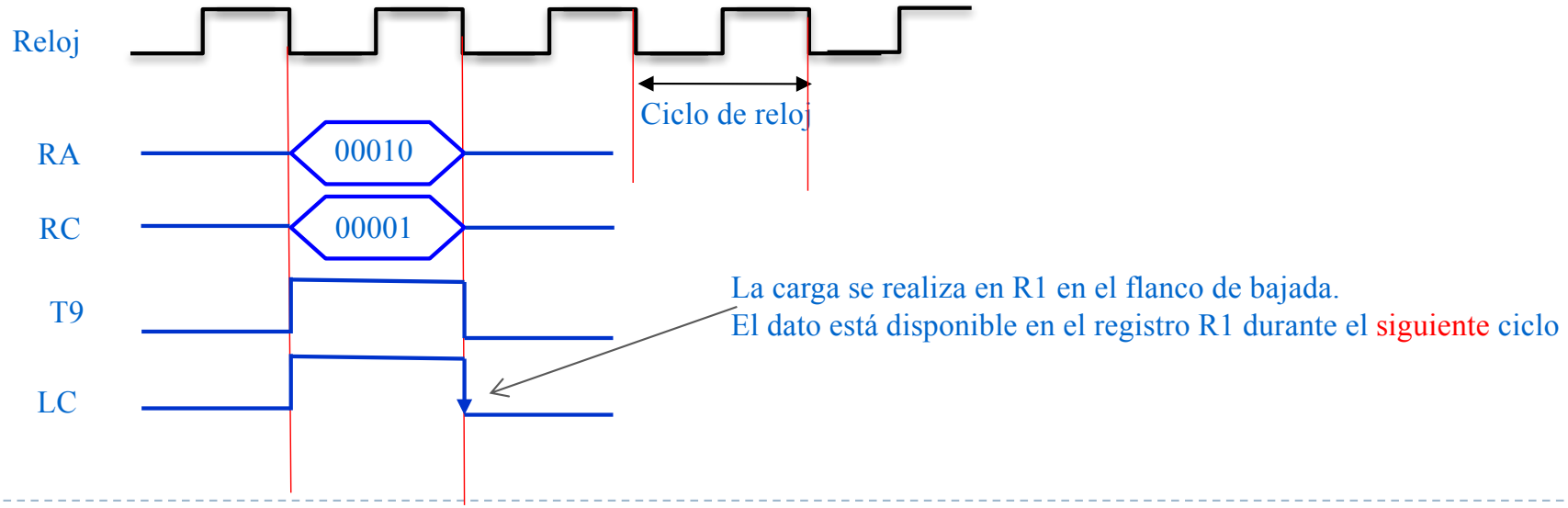
- Ry -> Identificador de registro para el punto y
- Mx -> Selección en multiplexor
- Tx -> Señal de activación triestado
- Cx -> Señal de carga de registro

- **Banco de registros y registros RT1 y RT2**
  - RA – salida de registro RA a A
  - RB – salida de registro RB a B
  - RC – salida de registro RC a E
  - LC – activa la escritura para RC
  - T9 – copia de A al bus interno
  - T10 – copia de B al bus interno
  - C4 – del bus interno al RT1
  - T4 – salida de RT1 al bus interno
  - C5 – del bus interno al RT2
  - T5 – salida de RT2 al bus interno

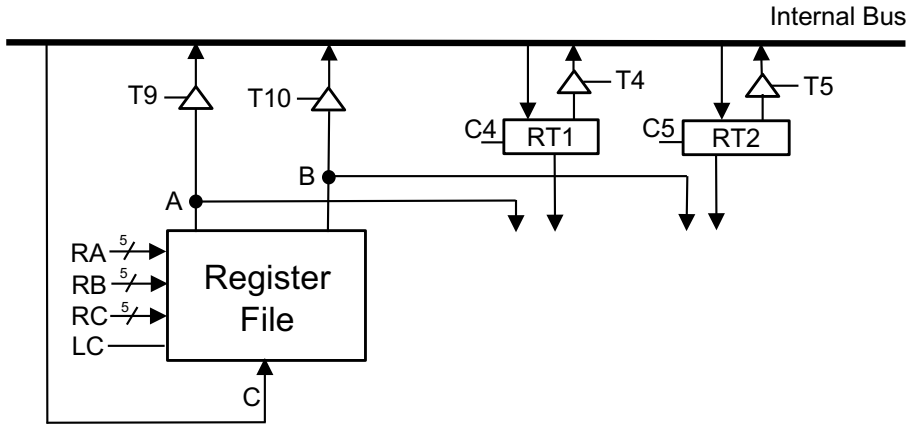
Uso del banco de registros

- Señales de control a activar en un ciclo para la operación elemental R1 ← R2, donde R1 es el registro 1 y R2 el registro 2 del Banco de registros:
  - RA = 00010
  - RC = 00001
  - T9 y LC
  - El resto a 0



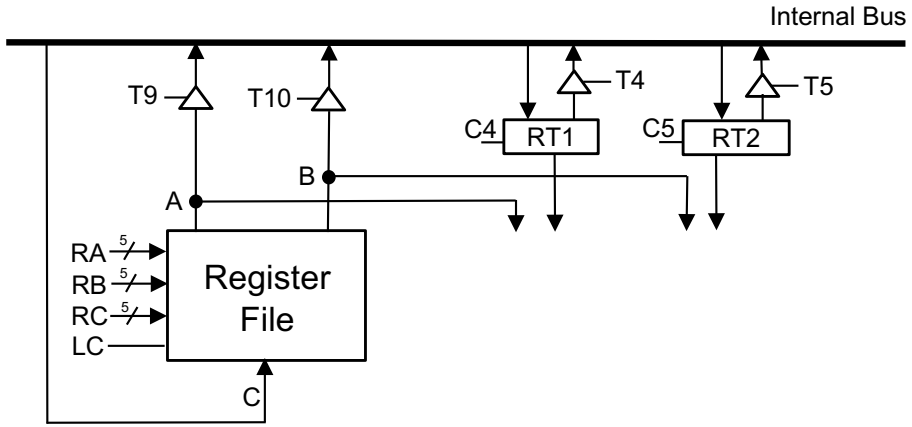


Ejemplo
operaciones elementales en registros



- **SWAP R1 R2**

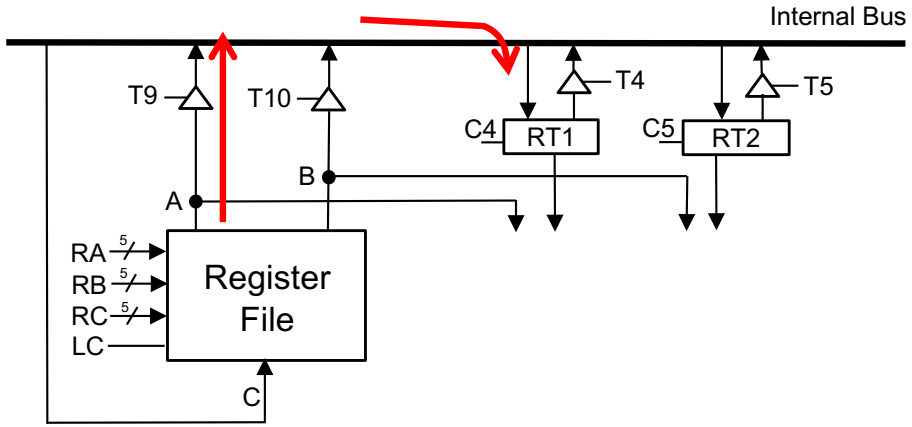
Ejemplo
operaciones elementales en registros



- **SWAP R1 R2**

O. Elemental	Señales

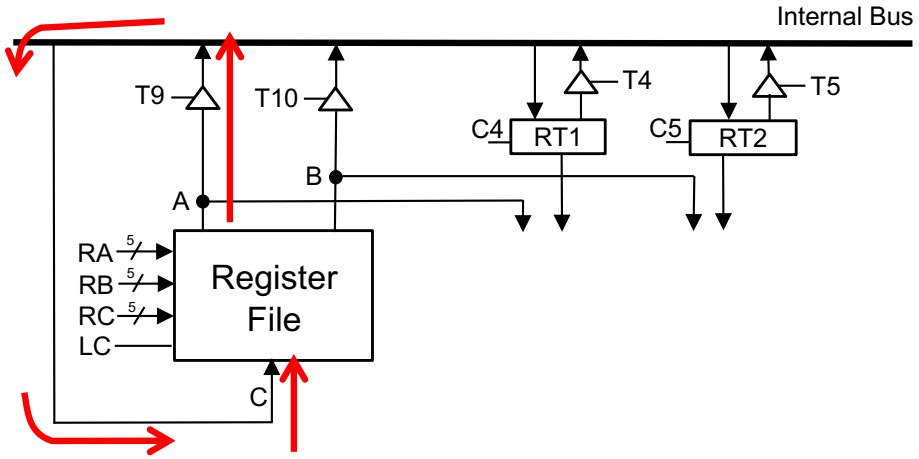
Ejemplo
operaciones elementales en registros



- **SWAP R1 R2**

O. Elemental	Señales
RT1← R1	RA=00001, T9, C4

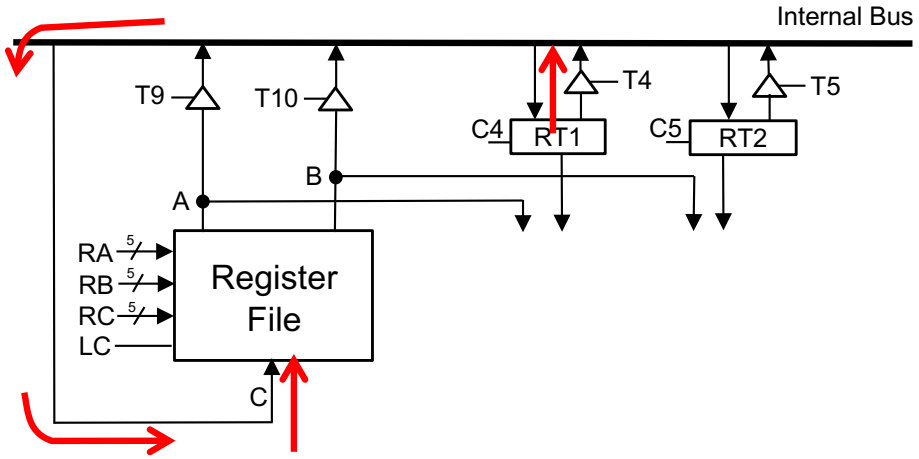
Ejemplo
operaciones elementales en registros



- **SWAP R1 R2**

O. Elemental	Señales
RT1← R1	RA=00001, T9, C4
R1 ← R2	RA=2 (00010), T9, RC=1, LC

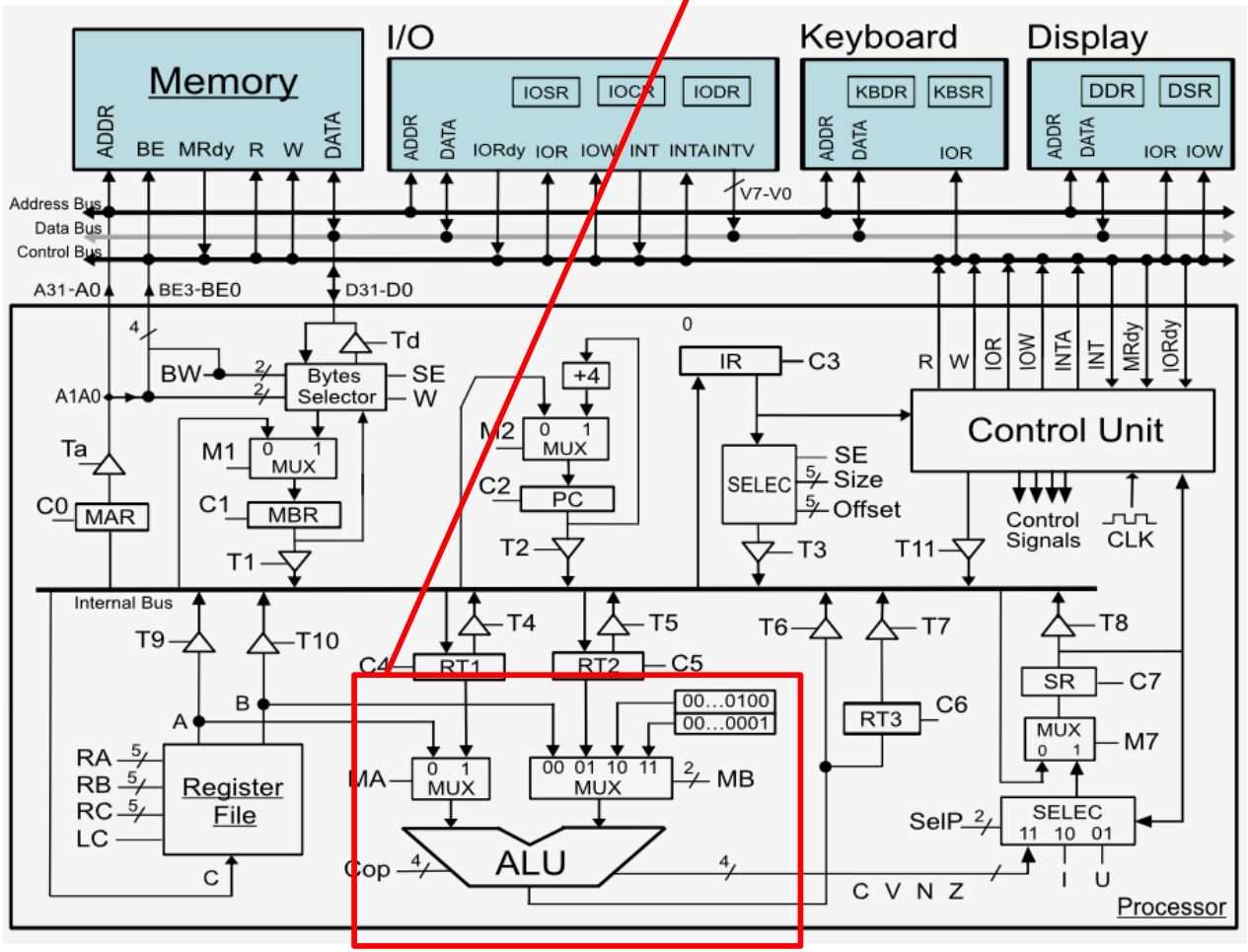
Ejemplo
operaciones elementales en registros



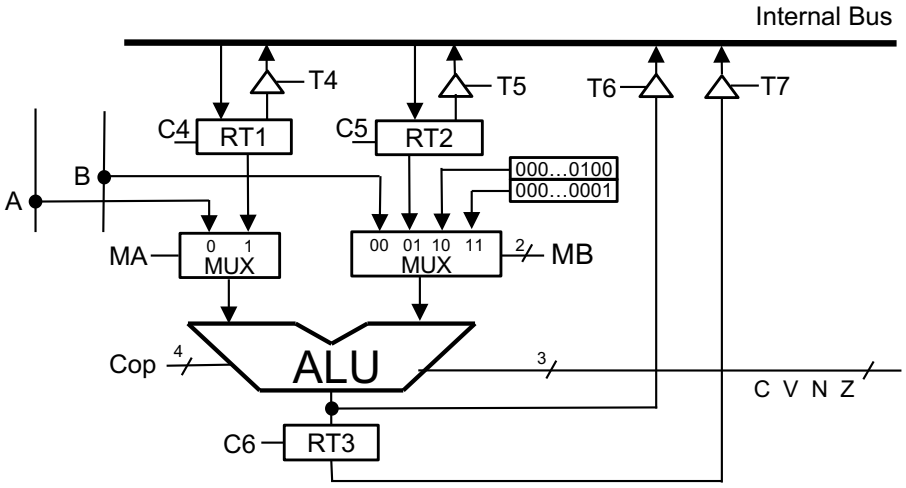
- **SWAP R1 R2**

O. Elemental	Señales
RT1← R1	RA=1, T9, C4
R1 ← R2	RA=2 (00010), T9, RC=1, LC
R2 ← RT1	T4, RC=2 (00010), LC

Estructura de un computador elemental



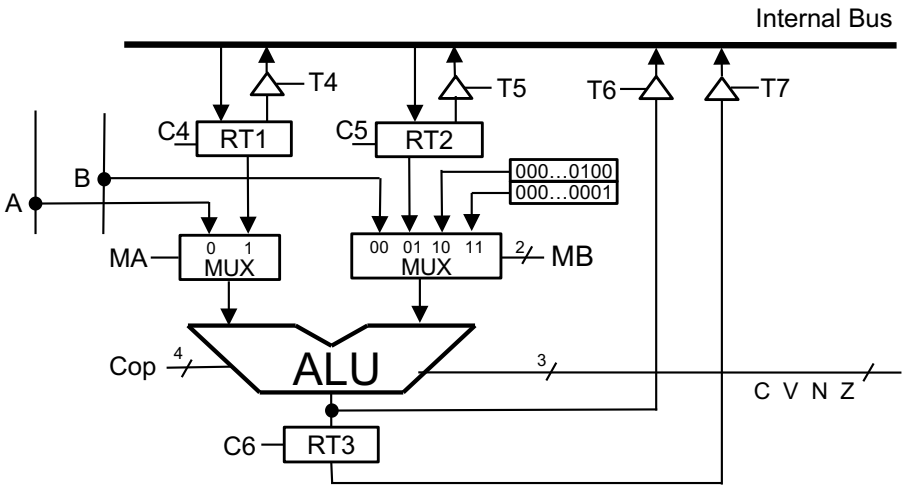
Señales de control

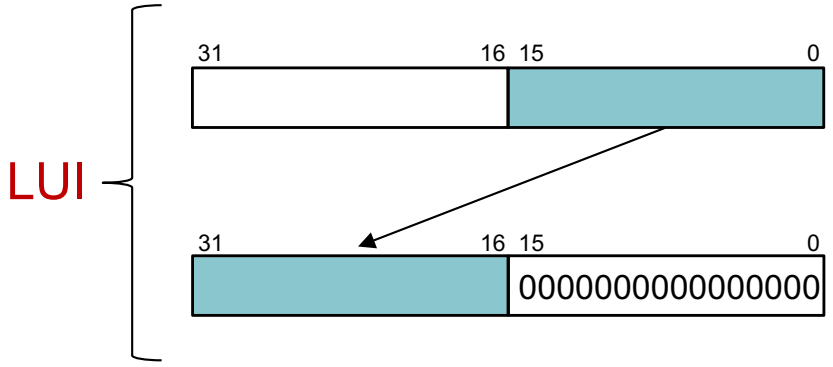


- **ALU**
  - MA – selección de operando A
  - MB – selección de operando B
  - Cop – código de operación

Cop (Cop_3-Cop_0)	Operación
0000	NOP
0001	A and B
0010	A or B
0011	not (A)
0100	A xor B
0101	Shift Right Logical (A) B= number of bits to shift
0110	Shift Right Arithmetic(A) B= number of bits to shift
0111	Shift left (A) B= number of bits to shift
1000	Rotate Right (A) B= number of bits to rotate
1001	Rotate Left (A) B= number of bits to rotate
1010	A + B
1011	A - B
1100	A * B (with overflow)
1101	A / B (integer division)
1110	A % B (integer division)
1111	LUI (A)

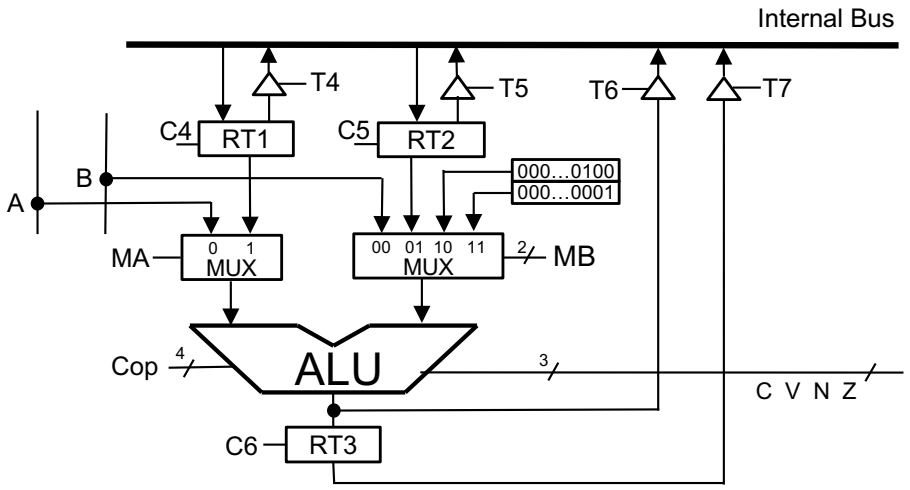
Señales de control





Cop (Cop_3-Cop_0)	Operación
0000	NOP
0001	A and B
0010	A or B
0011	not (A)
0100	A xor B
0101	Shift Right Logical (A) B= number of bits to shift
0110	Shift Right Arithmetic(A) B= number of bits to shift
0111	Shift left (A) B= number of bits to shift
1000	Rotate Right (A) B= number of bits to rotate
1001	Rotate Left (A) B= number of bits to rotate
1010	A + B
1011	A - B
1100	A * B (with overflow)
1101	A / B (integer division)
1110	A % B (integer division)
1111	LUI (A)

Señales de control



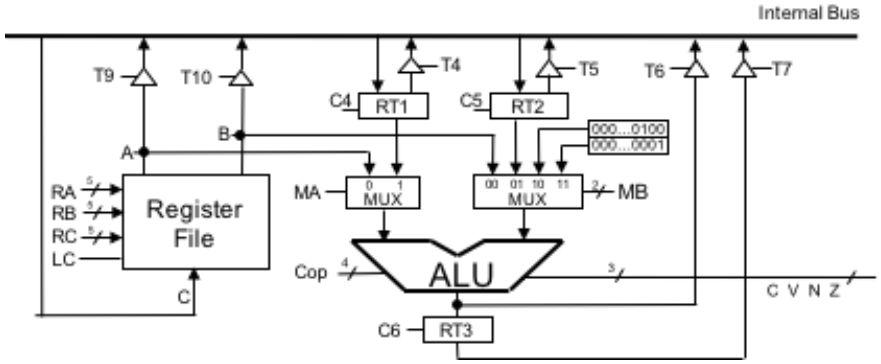
Resultado	C	V	N	Z
Resultado positivo (0 se considera +)	0	0	0	0
Resultado **== 0**	0	0	0	**1**
Resultado **negativo**	0	0	**1**	0
Desbordamiento de la operación	**0**	**1**	0	0
División por cero	0	**1**	0	**1**
Acarreo en el bit 32	**1**	0	0	0

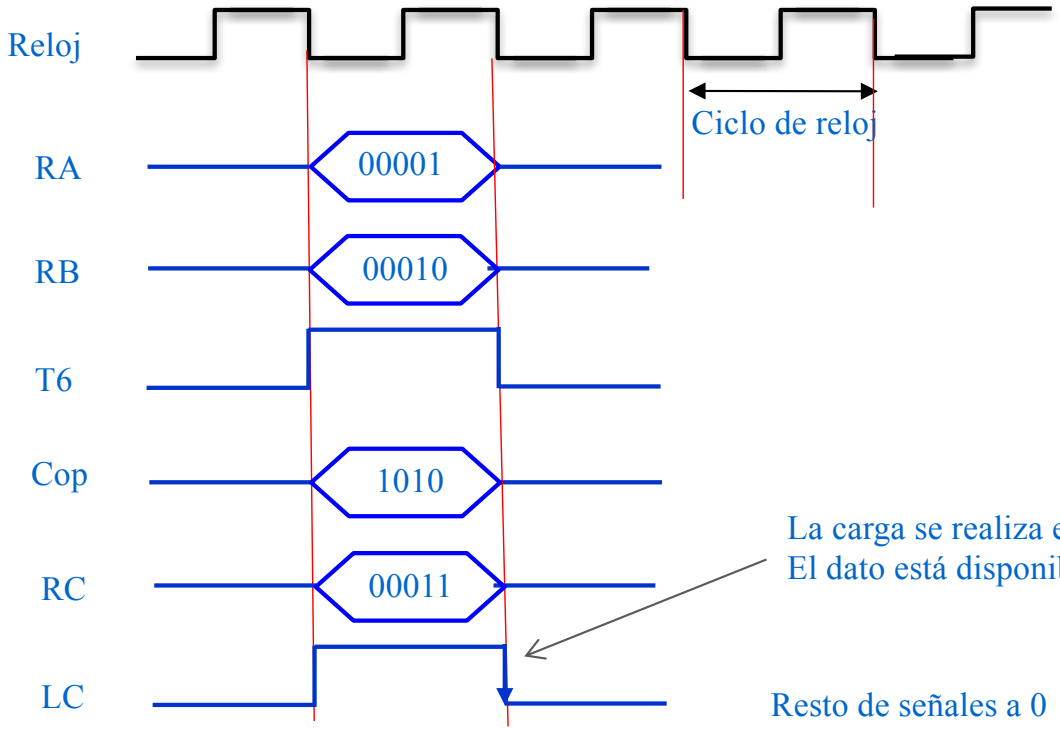
Cop (Cop_3-Cop_0)	Operación
0000	NOP
0001	A and B
0010	A or B
0011	not (A)
0100	A xor B
0101	Shift Right Logical (A) B= number of bits to shift
0110	Shift Right Arithmetic(A) B= number of bits to shift
0111	Shift left (A) B= number of bits to shift
1000	Rotate Right (A) B= number of bits to rotate
1001	Rotate Left (A) B= number of bits to rotate
1010	A + B
1011	A - B
1100	A * B (with overflow)
1101	A / B (integer division)
1110	A % B (integer division)
1111	LUI (A)

Ejemplo
operaciones elementales en ALU

- **ADD R3 R1 R2**

O. Elemental	Señales
R3← R1 + R2	RA=R1, RB=R2, Cop=+, T6, RC=R3, LC=1

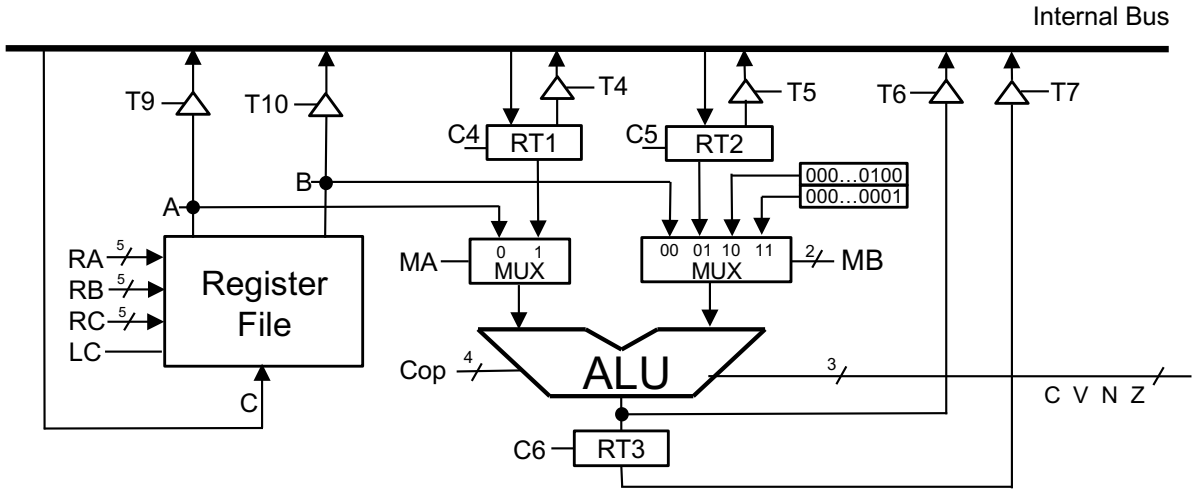




La carga se realiza en R3en el flanco de bajada.
El dato está disponible en el registro RT1 durante el siguiente ciclo

Resto de señales a 0

Ejemplo
operaciones elementales en ALU

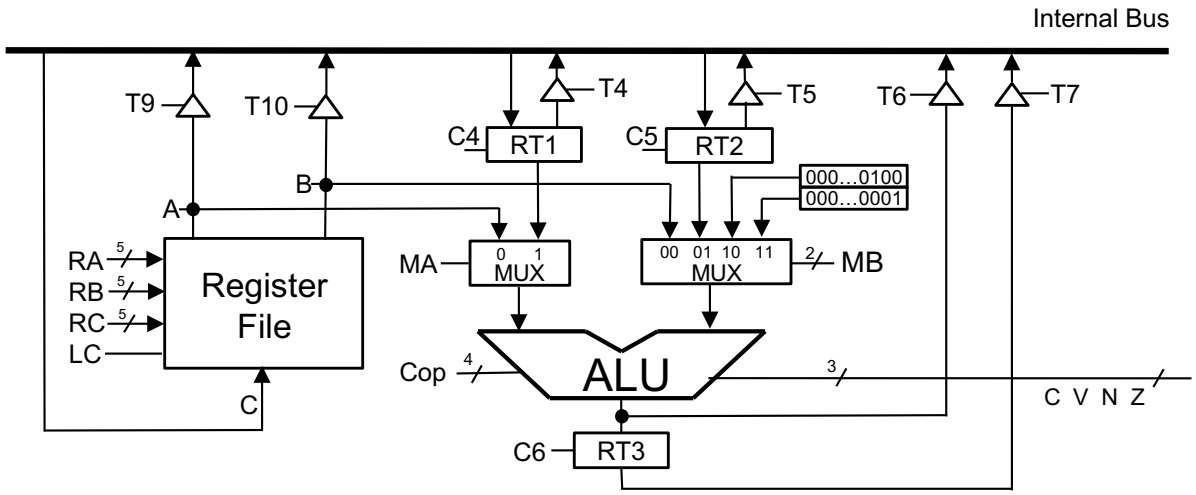


- **SWAP R1 R2**

O. Elemental	Señales
RT1← R1	RA=1, T9, C4
R1 ← R2	RA=2, T9, RC=1, LC
R2 ← RT1	T4, RC=2, LC

- **SWAP R1, R2 sin R_{tmp}**

Ejemplo
operaciones elementales en ALU



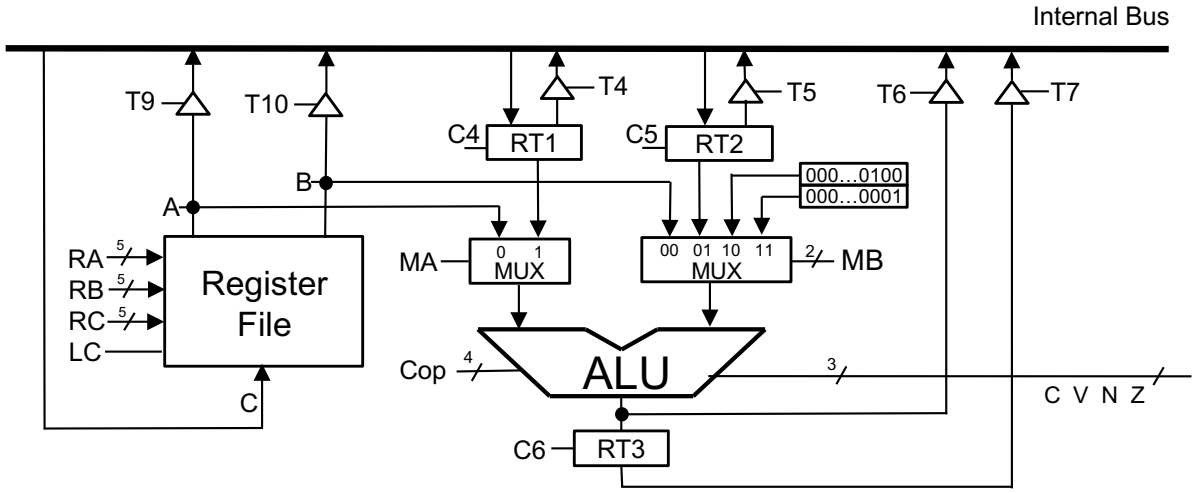
- **SWAP R1 R2**

O. Elemental	Señales
RT1← R1	RA=1, T9, C4
R1 ← R2	RA=2, T9, RC=1, LC
R2 ← RT1	T4, RC=2, LC

- **SWAP R1, R2 sin R_{tmp}**

O. Elemental	
R1←R1 ^ R2	R1 ← (R1 ^ R2)
R2←R1 ^ R2	R2 ← (R1 ^ R2) ^ R2
R1←R1 ^ R2	R1 ← (R1 ^ R2) ^ R1

Ejemplo
operaciones elementales en ALU



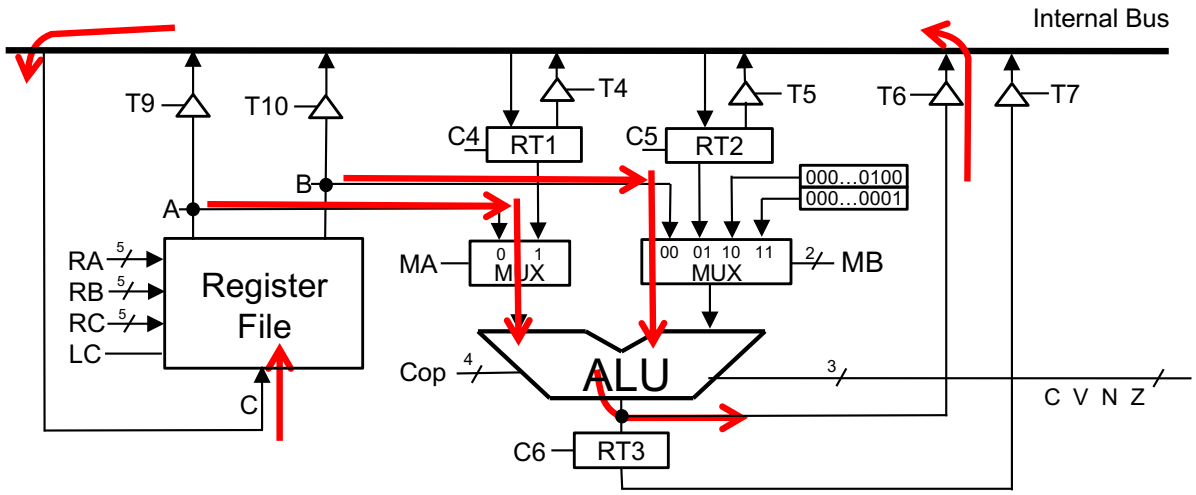
- **SWAP R1 R2**

O. Elemental	Señales
RT1← R1	RA=1, T9, C4
R1 ← R2	RA=2, T9, RC=1, LC
R2 ← RT1	T4, RC=2, LC

- **SWAP R1, R2 sin R_{tmp}**

O. Elemental	Señales
R1←R1 ^ R2	RA=1, RB=2, Cop=^, T6, RC=1
R2←R1 ^ R2	RA=1, RB=2, Cop=^, T6, RC=2
R1←R1 ^ R2	RA=1, RB=2, Cop=^, T6, RC=1

Ejemplo
operaciones elementales en ALU



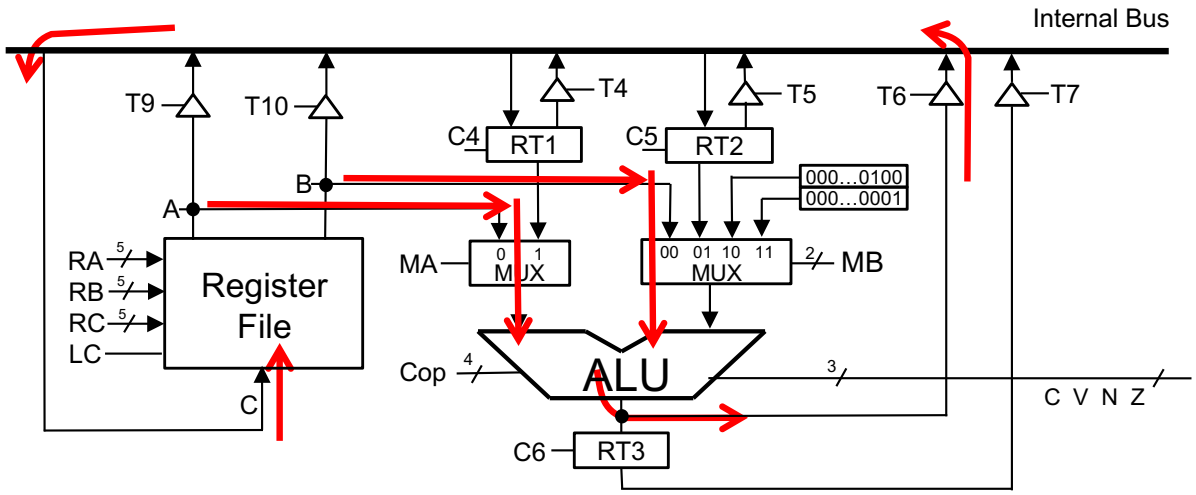
- **SWAP R1 R2**

O. Elemental	Señales
RT1← R1	RA=1, T9, C4
R1 ← R2	RA=2, T9, RC=1, LC
R2 ← RT1	T4, RC=2, LC

- **SWAP R1, R2 sin R_{tmp}**

O. Elemental	Señales
R1←R1 ^ R2	RA=1, RB=2, Cop=^, T6, RC=1
R2←R1 ^ R2	RA=1, RB=2, Cop=^, T6, RC=2
R1←R1 ^ R2	RA=1, RB=2, Cop=^, T6, RC=1

Ejemplo
operaciones elementales en ALU



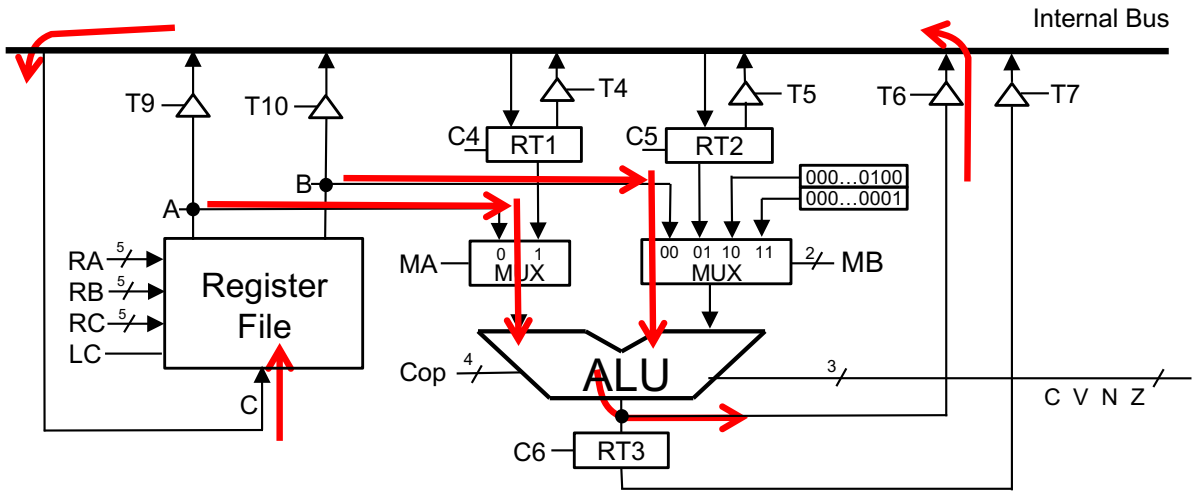
- **SWAP R1 R2**

O. Elemental	Señales
RT1← R1	RA=1, T9, C4
R1 ← R2	RA=2, T9, RC=1, LC
R2 ← RT1	T4, RC=2, LC

- **SWAP R1, R2 sin R_{tmp}**

O. Elemental	Señales
R1←R1 ^ R2	RA=1, RB=2, Cop=^, T6, RC=1
R2←R1 ^ R2	RA=1, RB=2, Cop=^, T6, RC=2
R1←R1 ^ R2	RA=1, RB=2, Cop=^, T6, RC=1

Ejemplo
operaciones elementales en ALU



- **SWAP R1 R2**

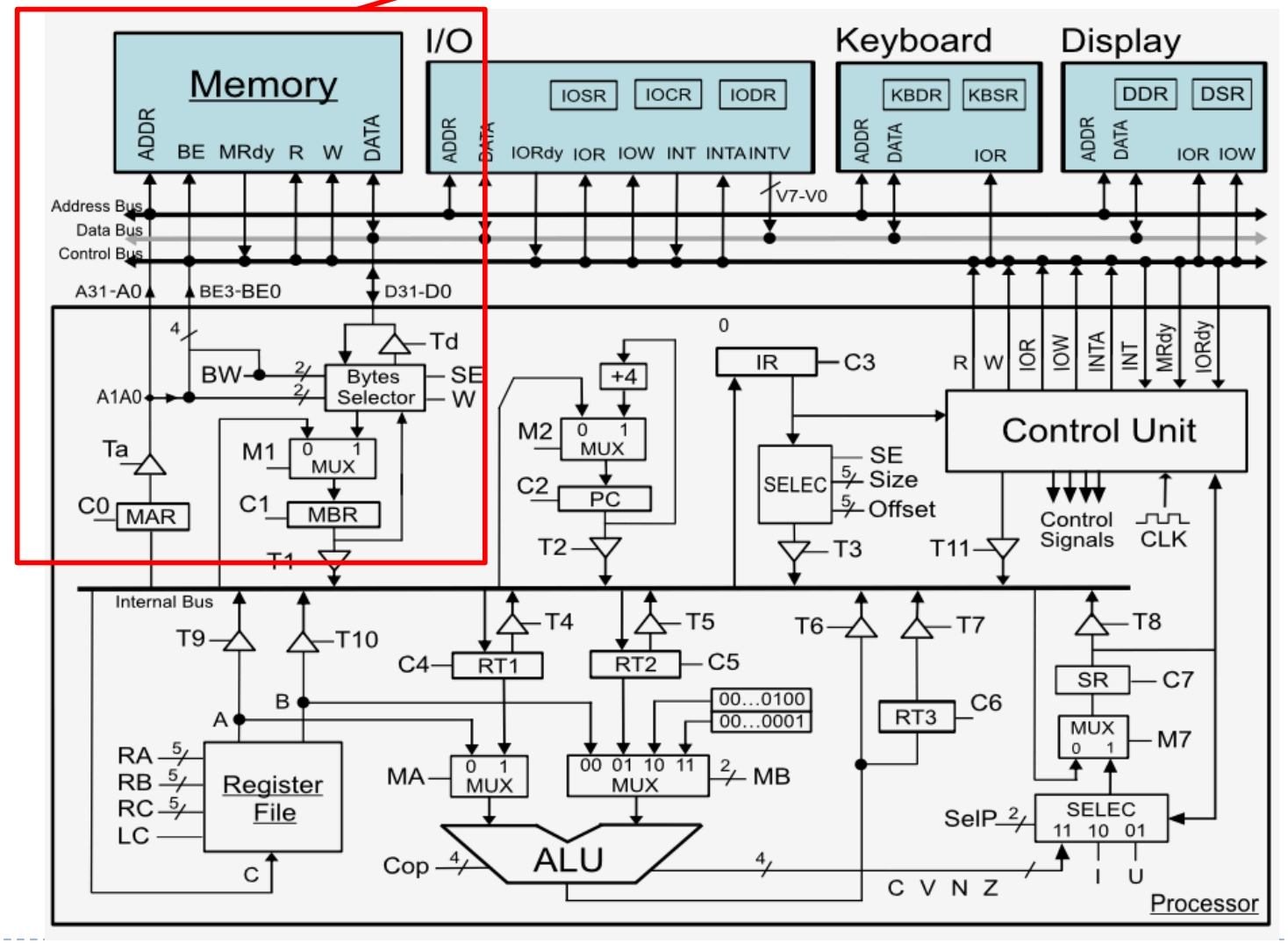
O. Elemental	Señales
RT1← R1	RA=1, T9, C4
R1 ← R2	RA=2, T9, RC=1, LC
R2 ← RT1	T4, RC=2, LC

- **SWAP R1, R2 sin R_{tmp}**

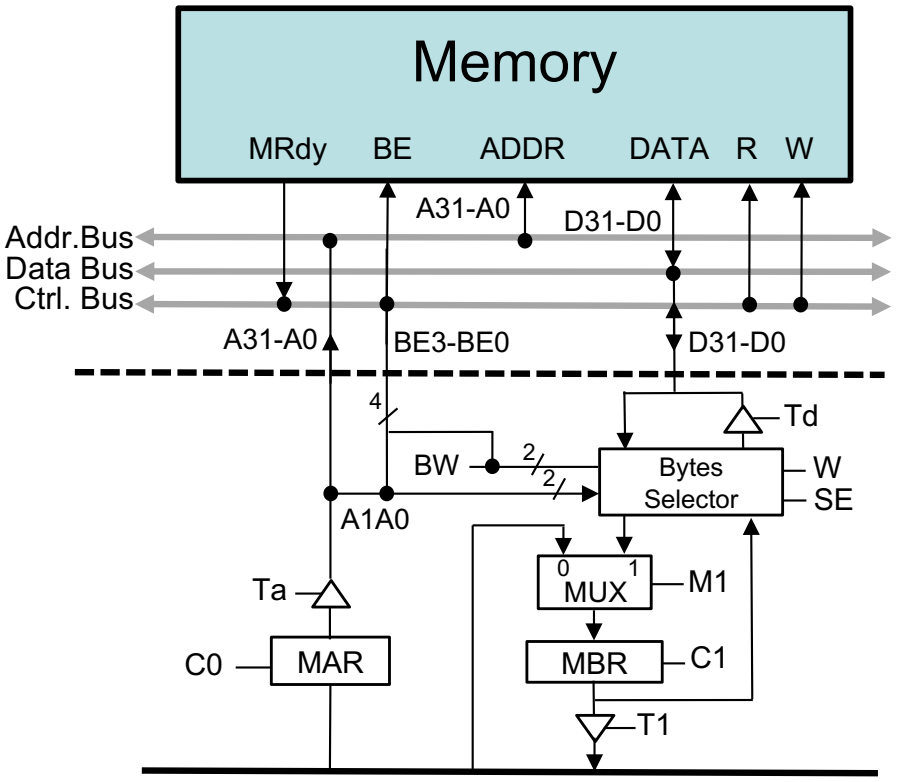
O. Elemental	Señales
R1←R1 ^ R2	RA=1, RB=2, Cop=^, T6, RC=1
R2←R1 ^ R2	RA=1, RB=2, Cop=^, T6, RC=2
R1←R1 ^ R2	RA=1, RB=2, Cop=^, T6, RC=1

Estructura de un computador elemental

Memoria principal,
registro de direcciones y de datos



Señales de control



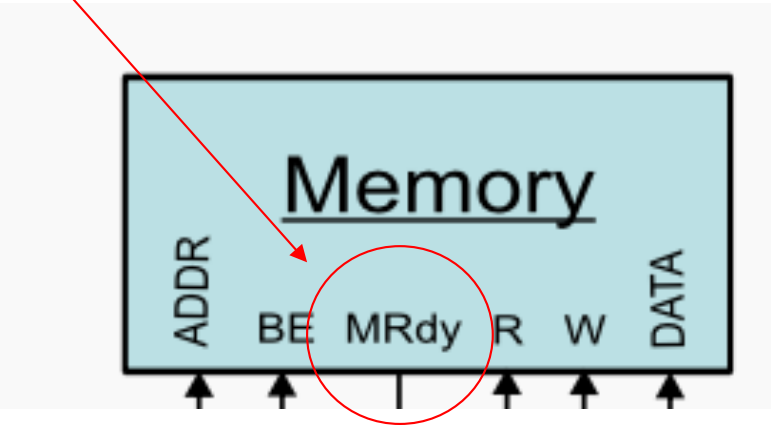
Nomenclatura:

- MAR -> registro de direcciones
- MBR -> registro de datos

- **Memoria principal**
  - R – lectura
  - W – escritura
  - BE3-BE0 = A1A0 + BW
    - Tamaño acceso (byte, palabra, media palabra)
  - C0 – del bus interno al MAR
  - C1 – del bus de datos al MBR
  - Ta – salida de MAR al bus de direcciones
  - Td – salida de MBR al bus de datos
  - T1 – salida de MBR al bus interno
  - M1 – selección para MBR: de memoria o bus interno

Acceso a Memoria

- Síncrono: la memoria requiere un número determinado de ciclos
- Asíncrono: la memoria indica cuándo finaliza la operación



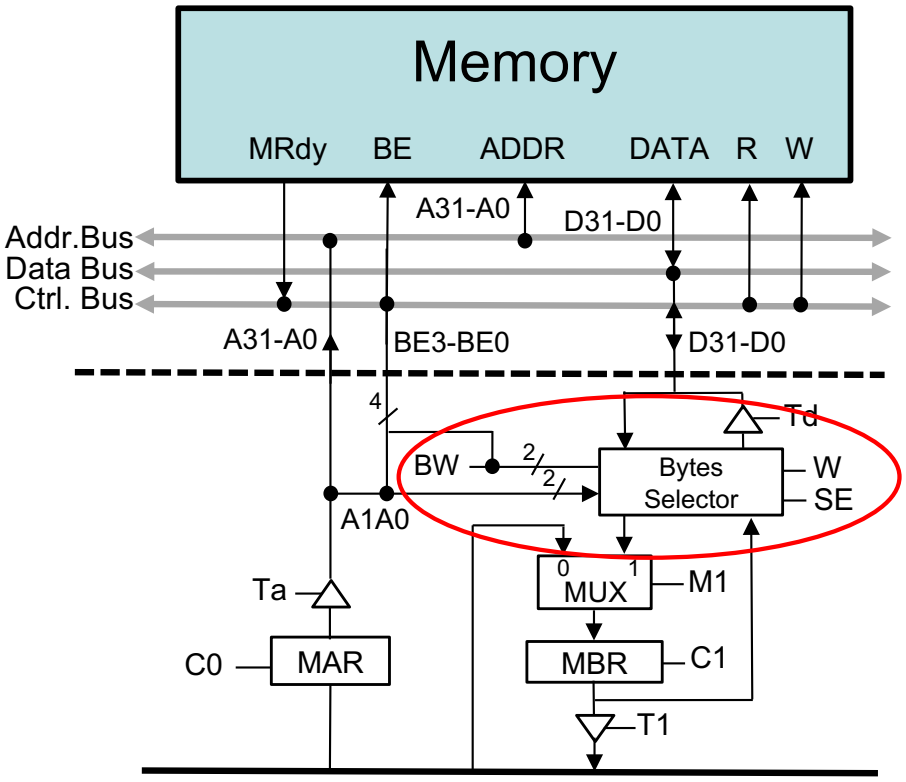
Señales BE (Byte-Enable) para lectura

Bytes en memoria				Selección de bytes				Salida al BUS			
D31-D24	D23-D16	D15-D8	D7-D0	BE3	BE2	BE1	BE0	D31-D24	D23-D16	D15-D8	D7-D0
Byte 3	Byte 2	Byte 1	Byte 0	0	0	0	0	---	---	---	Byte 0
Byte 3	Byte 2	Byte 1	Byte 0	0	0	0	1	---	---	Byte 1	---
Byte 3	Byte 2	Byte 1	Byte 0	0	0	1	0	--	Byte 2	---	---
Byte 3	Byte 2	Byte 1	Byte 0	0	0	1	1	Byte 3	---	---	---
Byte 3	Byte 2	Byte 1	Byte 0	0	1	0	X	---	---	Byte 1	Byte 0
Byte 3	Byte 2	Byte 1	Byte 0	0	1	1	X	Byte 3	Byte 2	---	---
Byte 3	Byte 2	Byte 1	Byte 0	1	1	X	X	Byte 3	Byte 2	Byte 1	Byte 0

Señales BE (Byte-Enable) para escritura

Dato en el bus				Selección de bytes				Bytes escritos en memoria			
D31-D24	D23-D16	D15-D8	D7-D0	BE3	BE2	BE1	BE0	D31-D24	D23-D16	D15-D8	D7-D0
Byte 3	Byte 2	Byte 1	Byte 0	0	0	0	0	---	---	---	Byte 0
Byte 3	Byte 2	Byte 1	Byte 0	0	0	0	1	---	---	Byte 1	---
Byte 3	Byte 2	Byte 1	Byte 0	0	0	1	0	--	Byte 2	---	---
Byte 3	Byte 2	Byte 1	Byte 0	0	0	1	1	Byte 3	---	---	---
Byte 3	Byte 2	Byte 1	Byte 0	0	1	0	X	---	---	Byte 1	Byte 0
Byte 3	Byte 2	Byte 1	Byte 0	0	1	1	X	Byte 3	Byte 2	---	---
Byte 3	Byte 2	Byte 1	Byte 0	1	1	X	X	Byte 3	Byte 2	Byte 1	Byte 0

Tamaño de acceso a memoria



Nomenclatura:

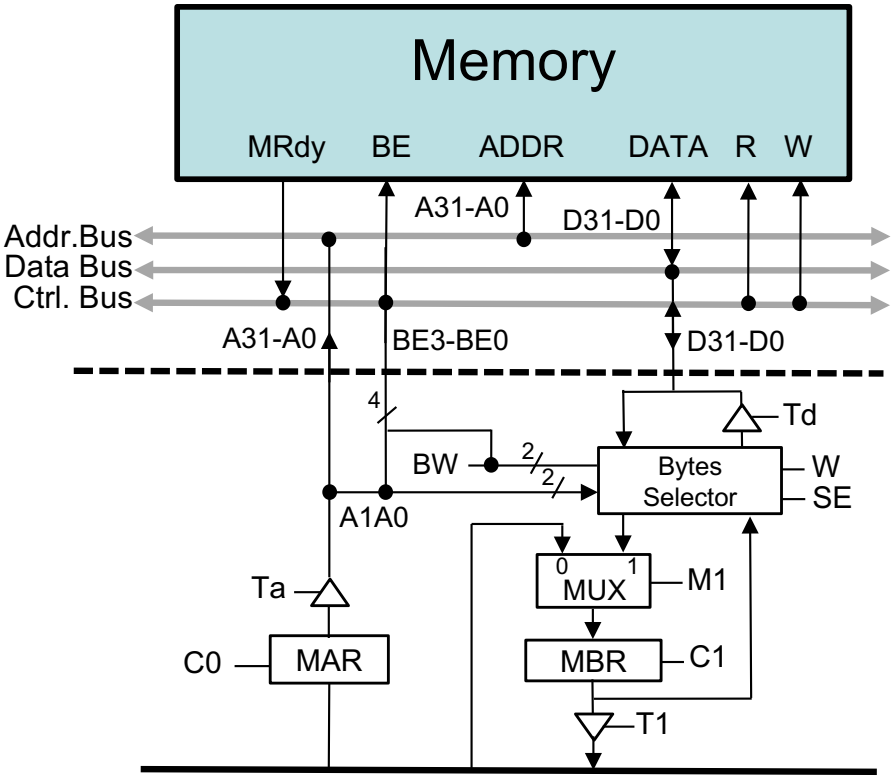
- MAR -> registro de direcciones
- MBR -> registro de datos

- Bytes Selector: selecciona qué bytes se almacenan en MBR en lectura y se vuelcan al bus en escritura

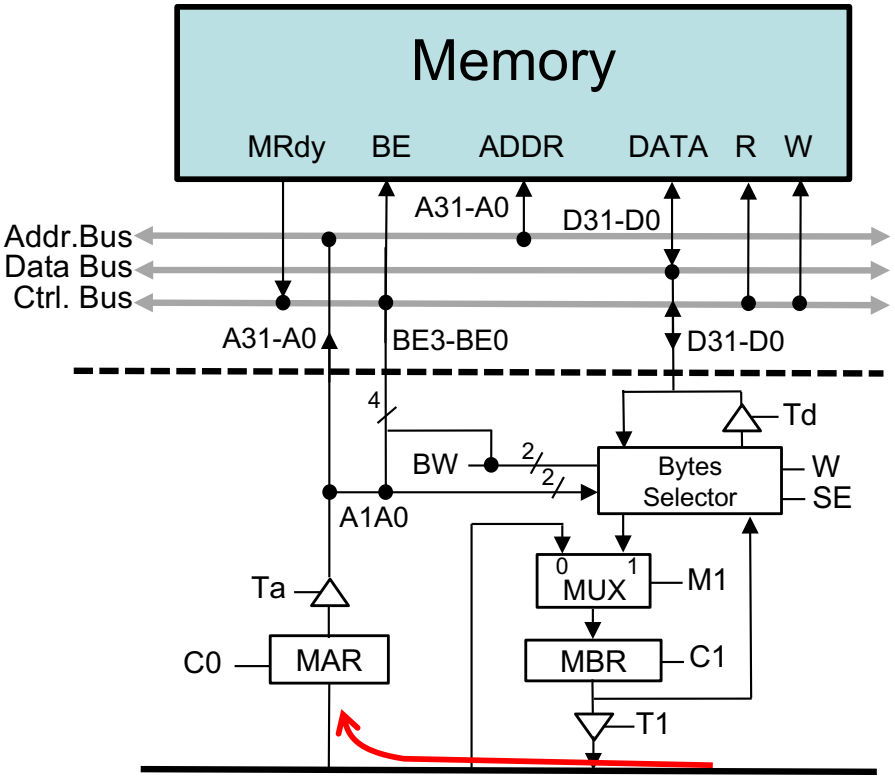
- Acceso a bytes: BW=0
- Acceso a media palabra: BW=01
- Acceso a palabra: BW =11
- SE: extensión de signo
  - 0: no extiende el signo en accesos más pequeños de una palabra
  - 1: extiende el signo en accesos más pequeños de una palabra

Ejemplo
operaciones elementales para usar la memoria

- **Lectura**



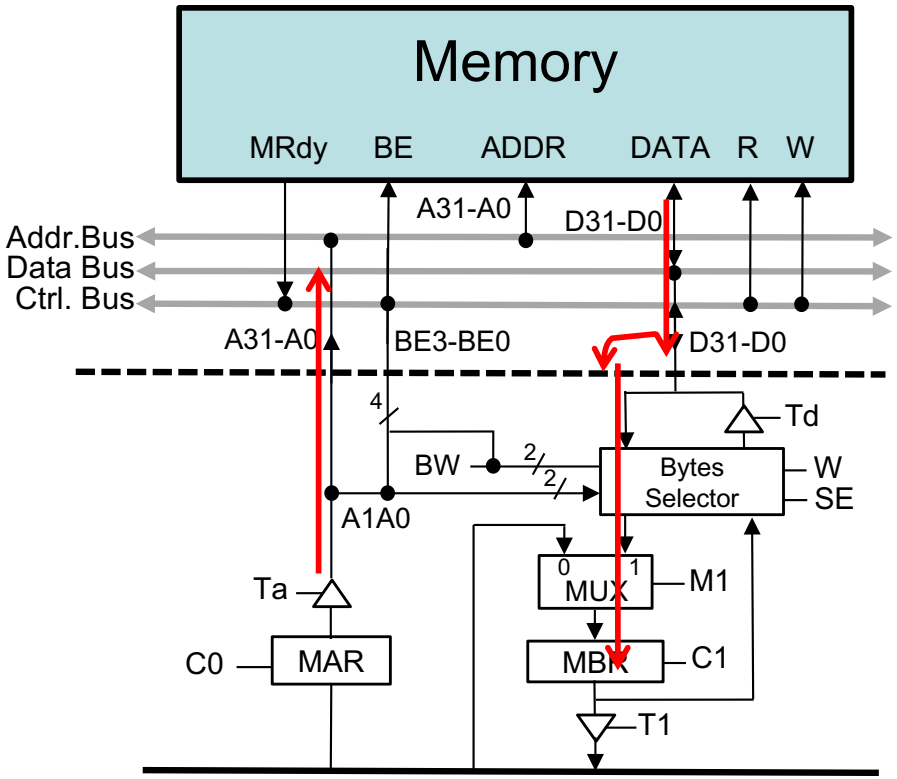
Ejemplo
Acceso a memoria síncrona de 1 ciclo



- **Lectura**

O. Elemental	Señales
MAR ← <dirección>	…, C0

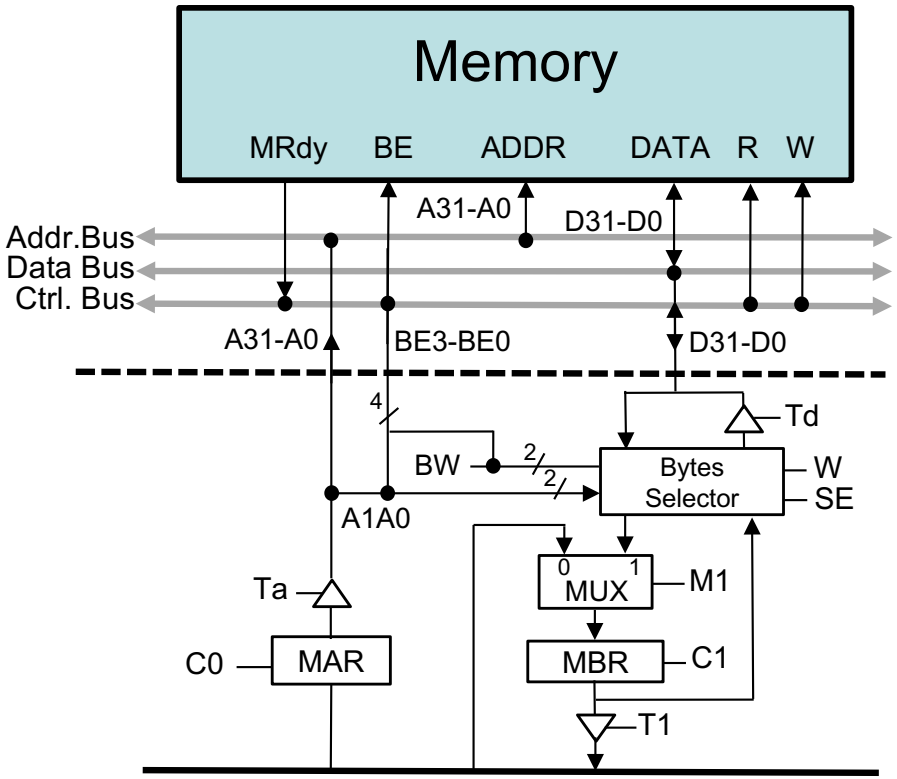
Ejemplo
Acceso a memoria síncrona de 1 ciclo



- **Lectura de una palabra**

O. Elemental	Señales
MAR ← <dirección>	…, C0
MBR ← MP[MAR]	Ta, R, M1, C1, BW=11

Ejemplo
Acceso a memoria síncrona de 1 ciclo

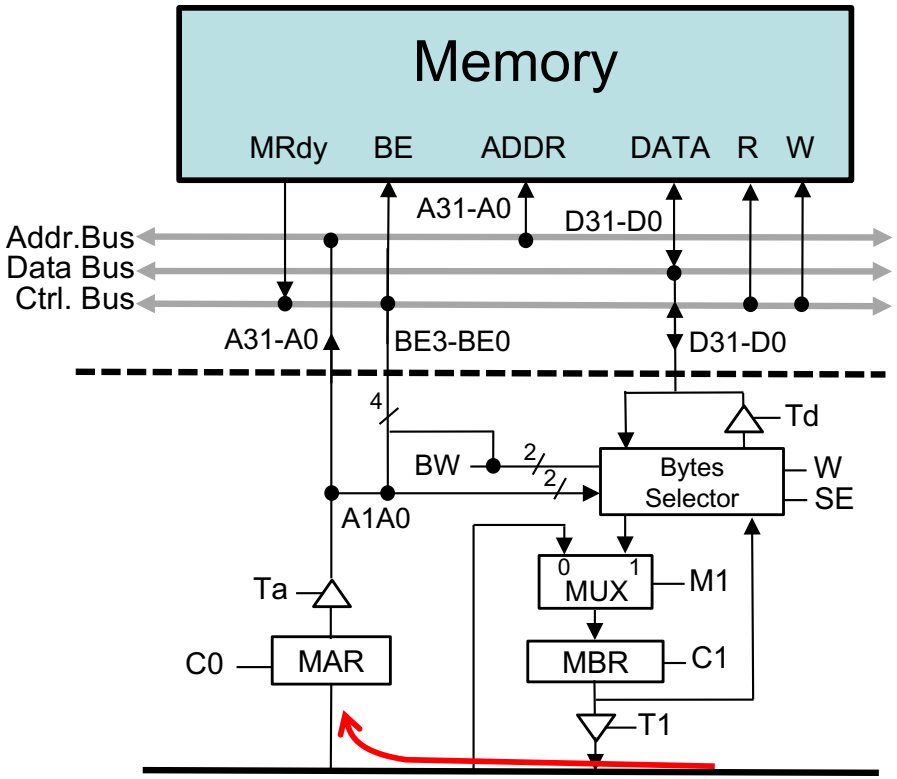


- **Lectura de una palabra**

O. Elemental	Señales
MAR ← <dirección>	…, C0
MBR ← MP[MAR]	Ta, R, M1, C1,BW=11

- **Escritura de una palabra**

Ejemplo
Acceso a memoria síncrona de 1 ciclo



- **Lectura**

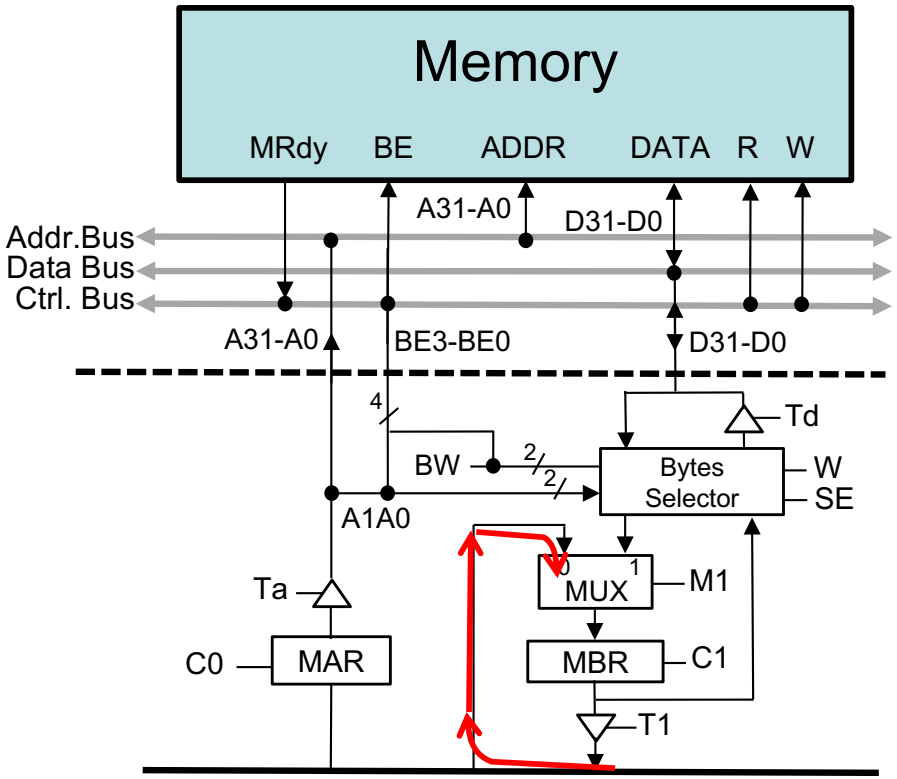
O. Elemental	Señales
MAR ← <dirección>	…, C0
MBR ← MP[MAR]	Ta, R, M1, C1

- **Escritura de una palabra**

O. Elemental	Señales
MAR ← <dirección>	…, C0

ARCOS @ UC3M
Félix García-Carballeira, Alejandro Calderón Mateos

Ejemplo
Acceso a memoria síncrona de 1 ciclo



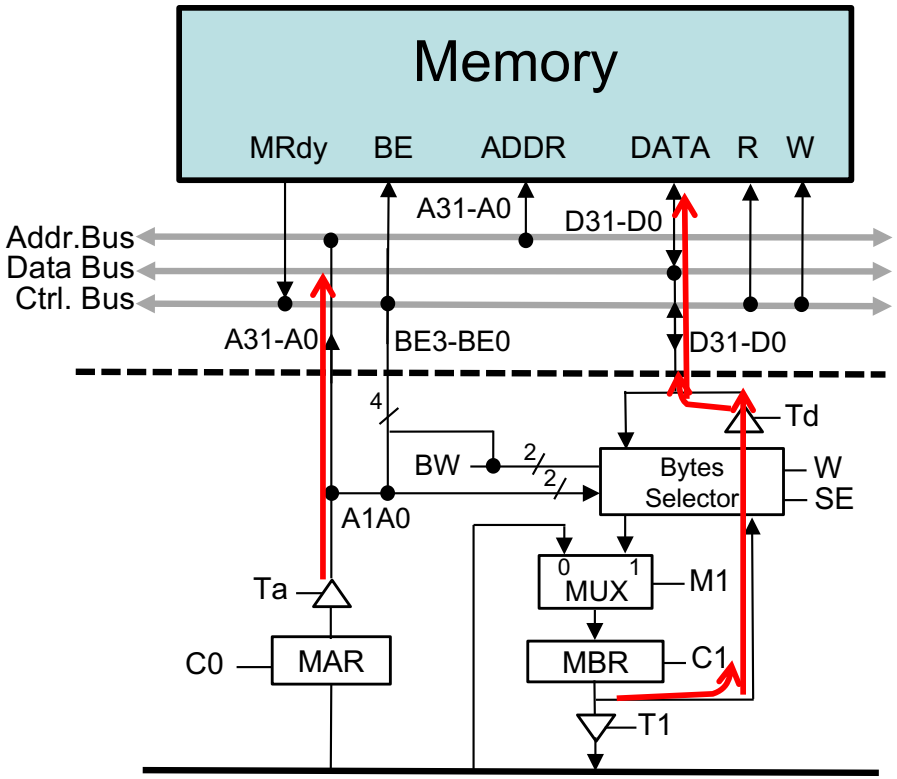
- **Lectura**

O. Elemental	Señales
MAR ← <dirección>	…, C0
MBR ← MP[MAR]	Ta, R, M1, C1

- **Escritura**

O. Elemental	Señales
MAR ← <dirección>	…, C0
MBR ← <dato>	…, C1

Ejemplo
Acceso a memoria síncrona de 1 ciclo



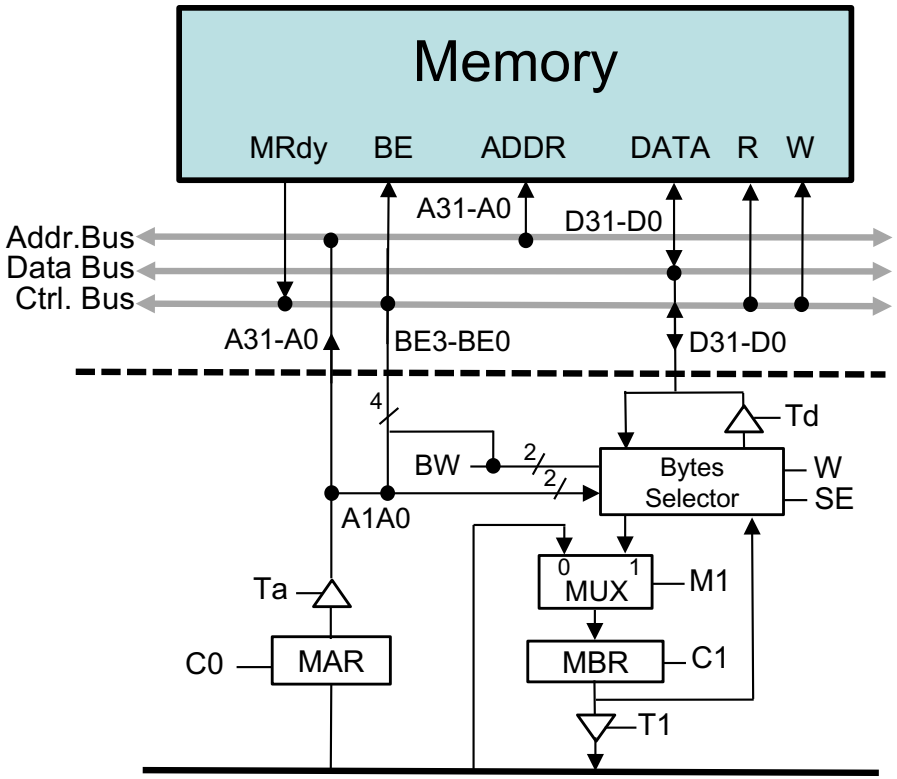
- **Lectura**

O. Elemental	Señales
MAR ← <dirección>	…, C0
MBR ← MP[MAR]	Ta, R, M1, C1

- **Escritura**

O. Elemental	Señales
MAR ← <dirección>	…, C0
MBR ← <dato>	…, C1
Ciclo de escritura	Ta, Td, W, BW=11

Ejemplo
Acceso a memoria síncrona de 1 ciclo



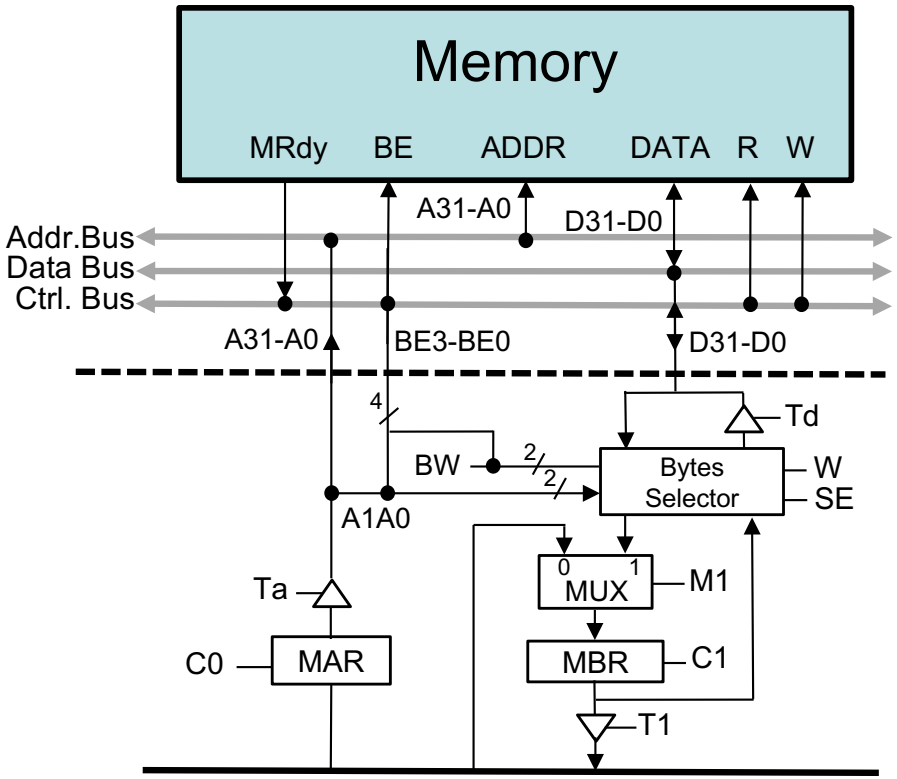
- **Lectura**

O. Elemental	Señales
MAR ← <dirección>	…, C0
MBR ← MP[MAR]	Ta, R, M1, C1

- **Escritura**

O. Elemental	Señales
MAR ← <dirección>	…, C0
MBR ← <dato>	…, C1
Ciclo de escritura	Ta, Td, W, BW=11

Ejemplo
Acceso a memoria síncrona de 2 ciclo

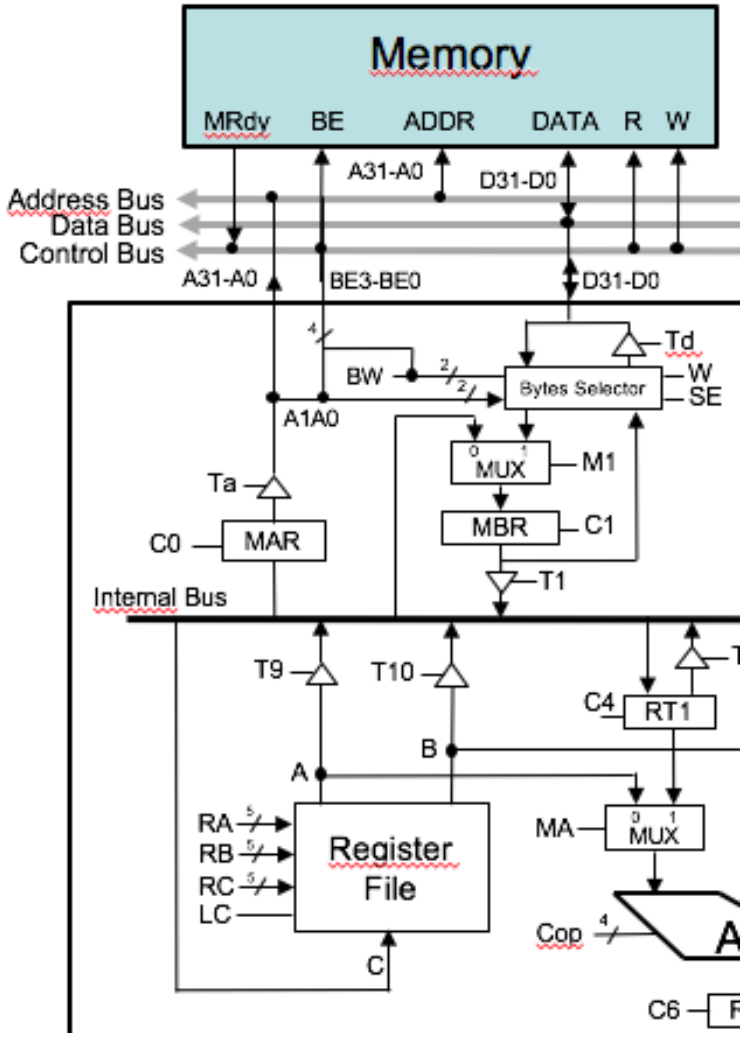


- **Lectura de una palabra**

O. Elemental	Señales
MAR ← <dirección>	…, C0
ciclo de lectura	Ta, R,
ciclo de lectura MBR ← MP[MAR]	Ta, R, M1, C1, BW=11

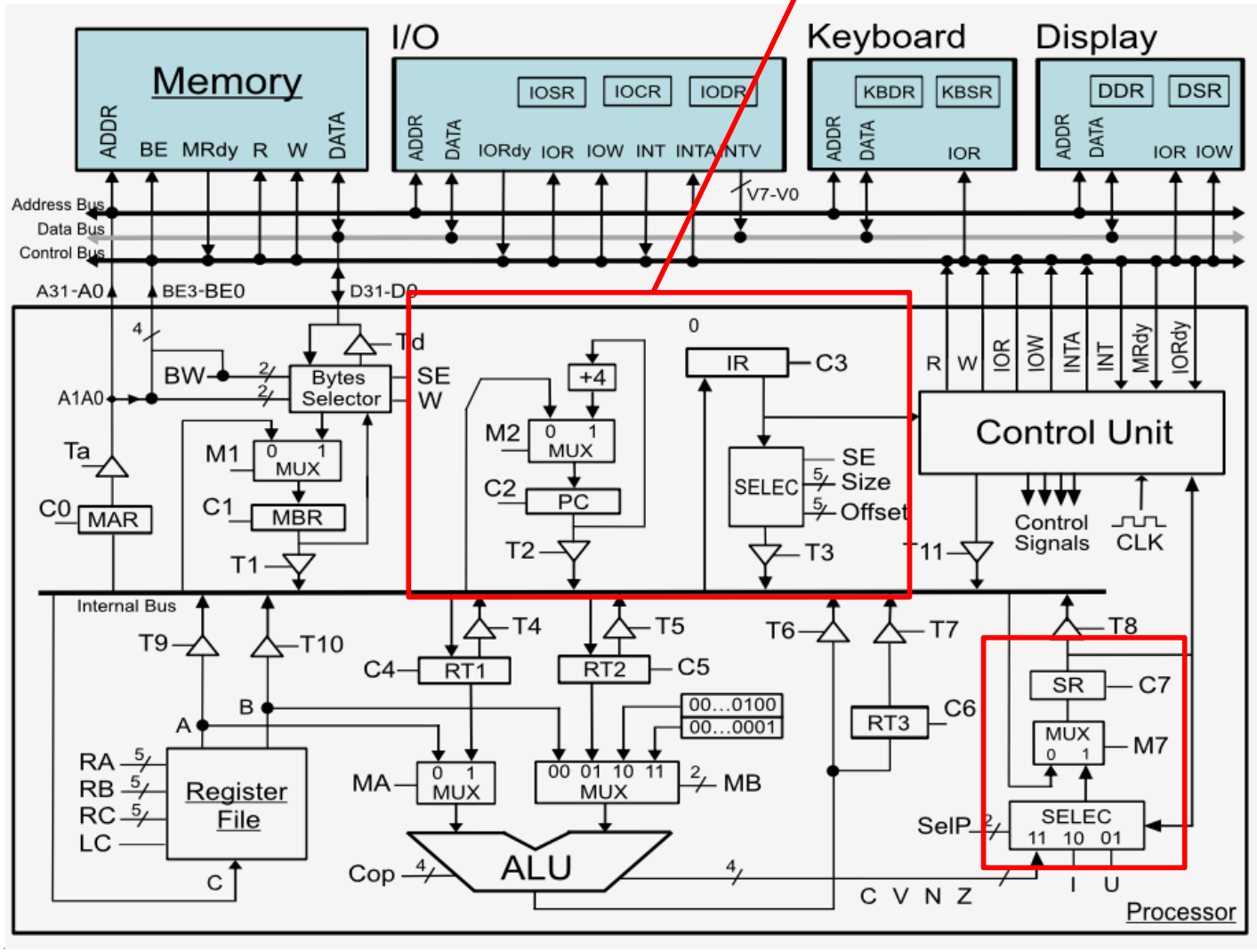
Ejemplo de acceso a memoria

- Señales de control a activar para realizar la operación de lectura (una palabra)
  R2 ← Memoria[R1]:
  - Primer ciclo: MAR ← R1
    - RA = 00001, T9, C0
  - Segundo ciclo: MBR ← Memoria
    - Ta, R, C1, M1, BW=11
  - Tercer ciclo: R2 ← MBR
    - T1, RC= 00010, LC



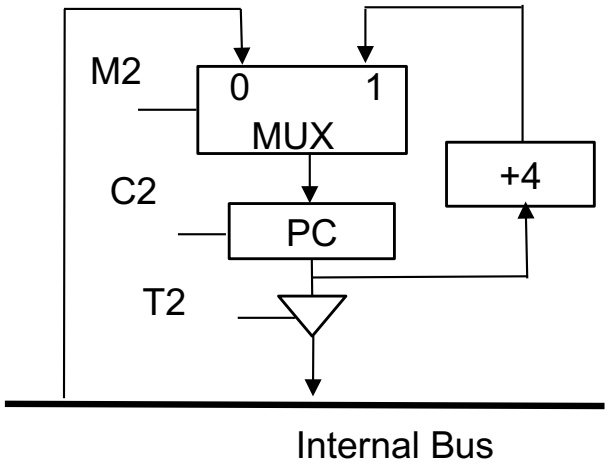
Estructura de un computador elemental

Registros SR, PC y IR

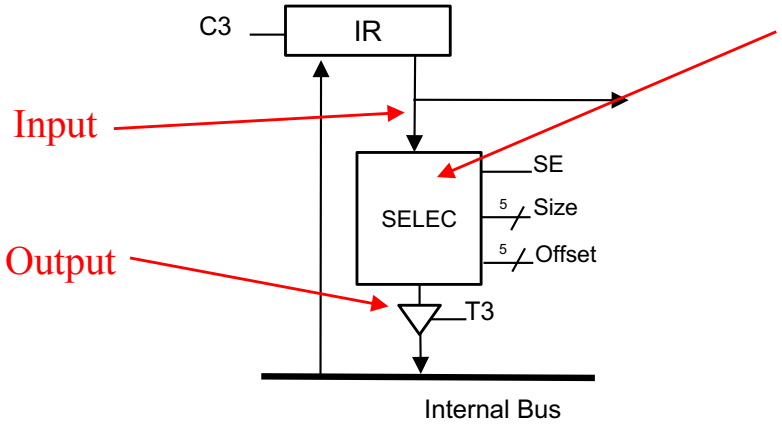


Contador de programa

- **Contador de programa PC**:
  - C2, M2
    - PC ← PC + 4
  - C2 – del bus interno al PC
  - T2 – de PC a bus interno

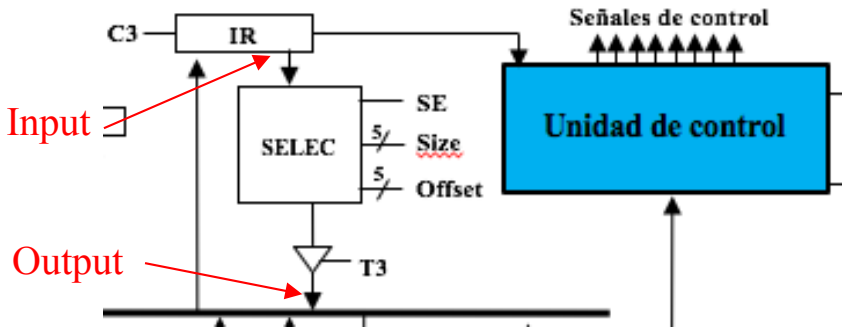


Registro de instrucción



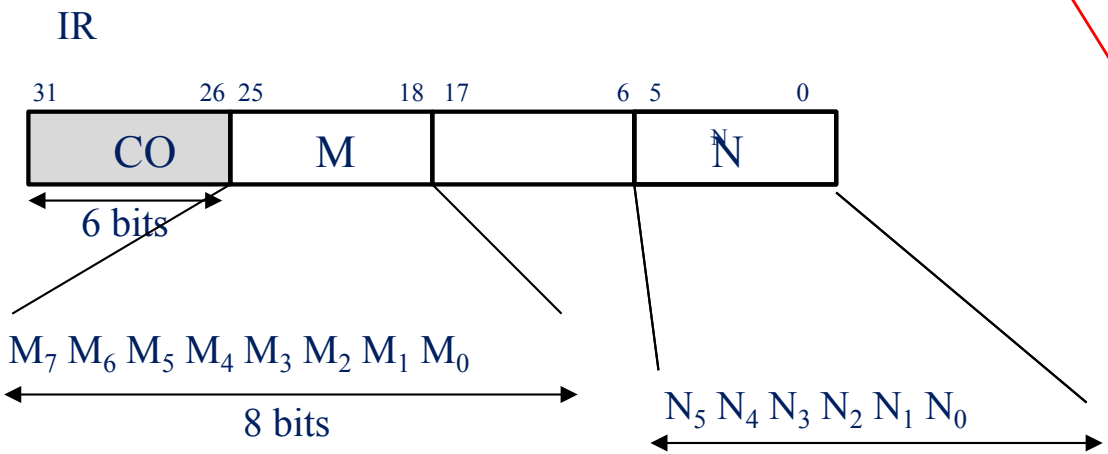
- C3 – del bus interno al IR
- SELEC: Transfiere el contenido de IR al bus
  - Size: Tamaño
  - Offset: desplazamiento
    - Bit de inicio (menos significativo)
  - SE: extensión de signo

Registro Selector



Selección sin extensión de signo (SE = 0)

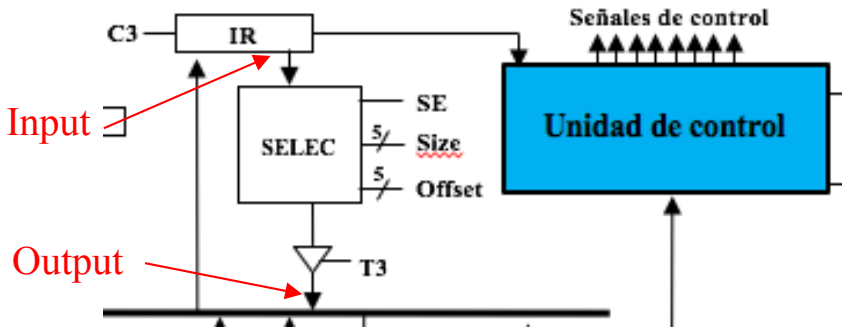
Size	Offset	Output
01000	10010	31 … 8 7 … 0: 0 0 0 $M_7..M_0$
00110	00000	31 … 8 7 … 0: 0 0 0 $00N_5…N_0$



Bit de inicio
(menos significativo)

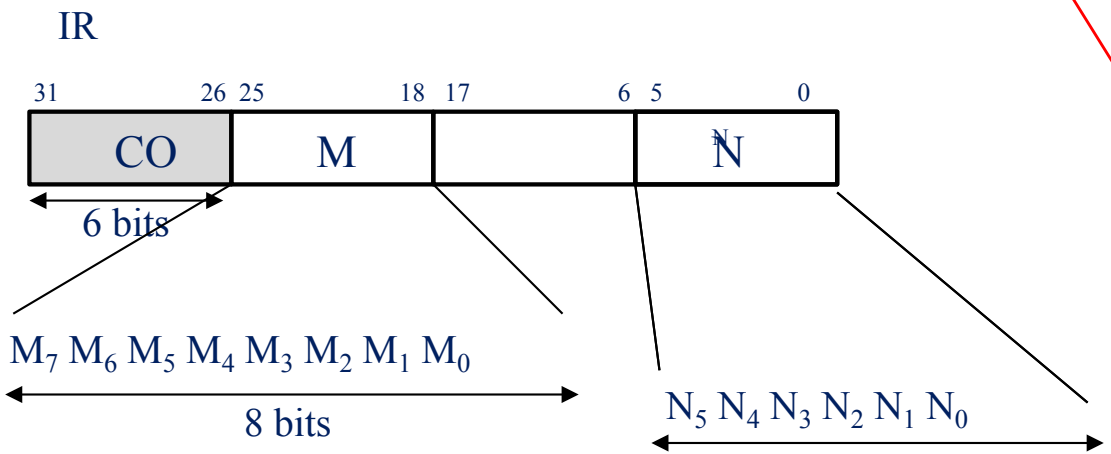
Tamaño en bits

Registro Selector



Selección con extensión de signo (SE = 1)

Size	Offset	Output
01000	10010	31 24 23 16 15 8 7 0 $M_7..M_7$ $M_7..M_7$ $M_7..M_7$ $M_7..M_0$
00110	00000	31 24 23 16 15 8 7 0 $N_5..N_5$ $N_5..N_5$ $N_5..N_5$ $N_5N_5N_5..N_0$

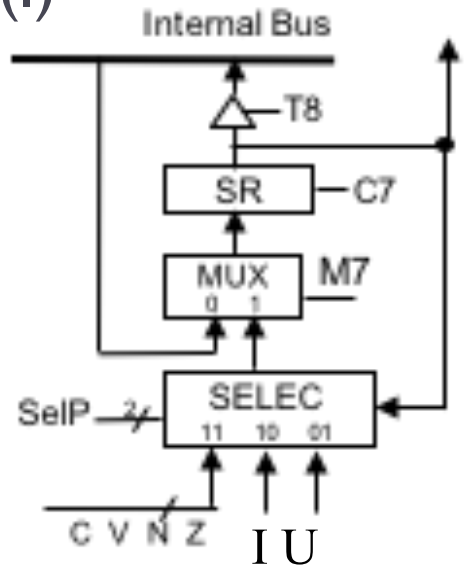


Bit de inicio
(menos significativo)

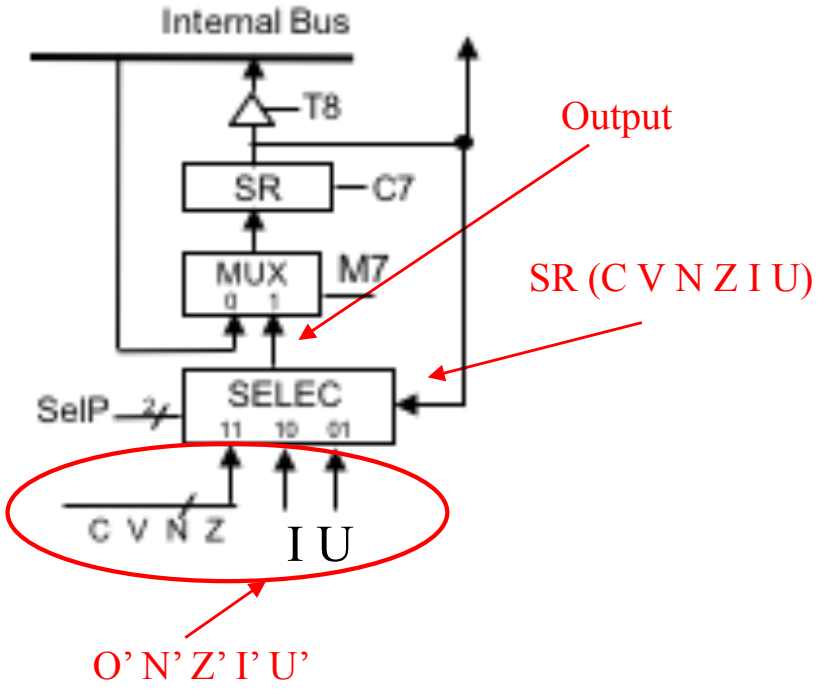
Tamaño en bits

Registro de estado

- Almacena información (bits de estado) sobre el estado del programa que se está ejecutando en el procesador:
  - Resultado de la última operación en la ALU: C, V, N, Z
  - Si el procesador está ejecutando en modo núcleo o modo usuario (U)
  - Si las interrupciones están habilitadas o no (I)
- Señales
  - C7 – de bus interno al SR
  - SelP, M7 – flags de ALU, I, o U a SR
  - T8 – del SR al bus interno



Registro de estado



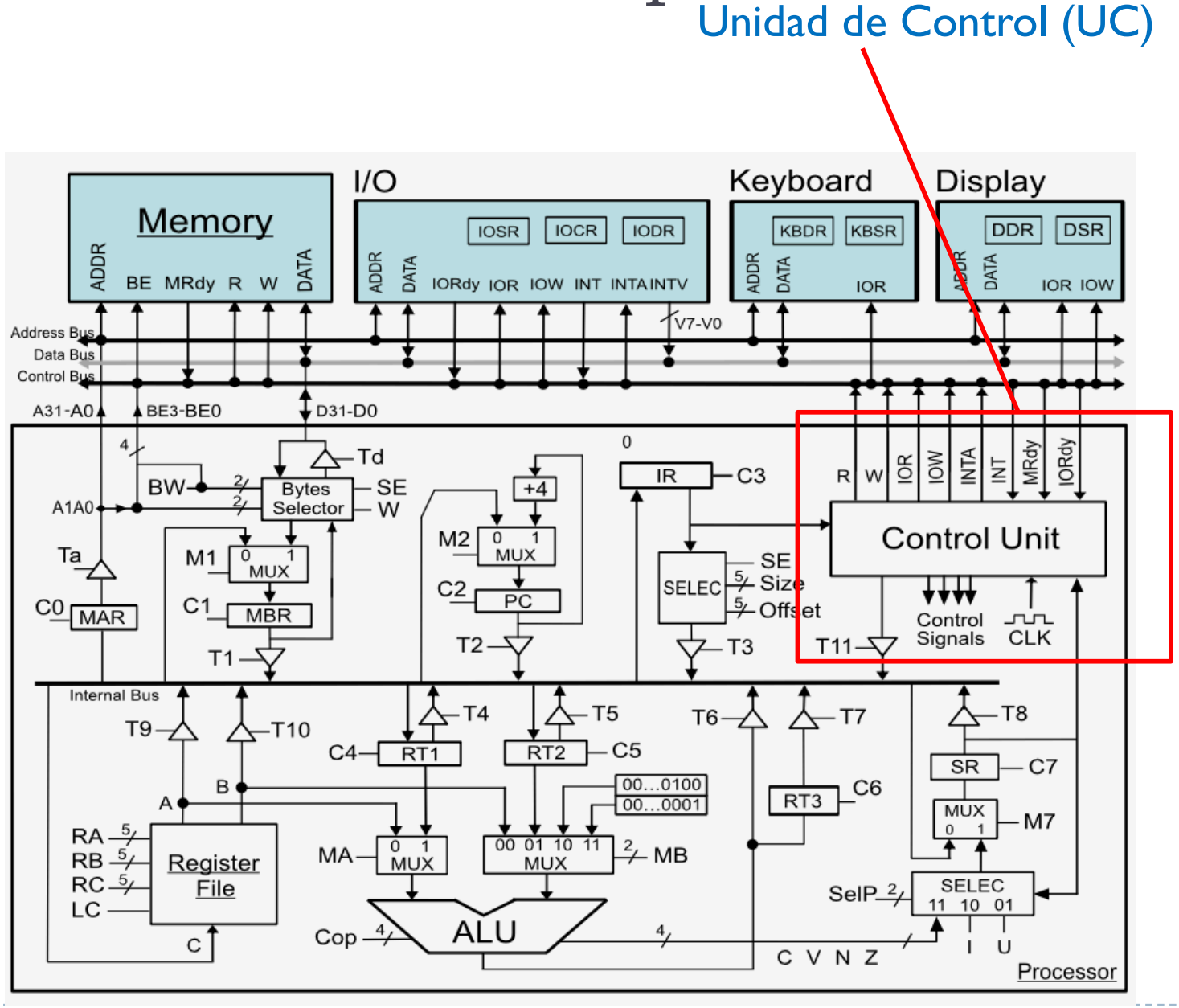
Operación de SELEC:

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

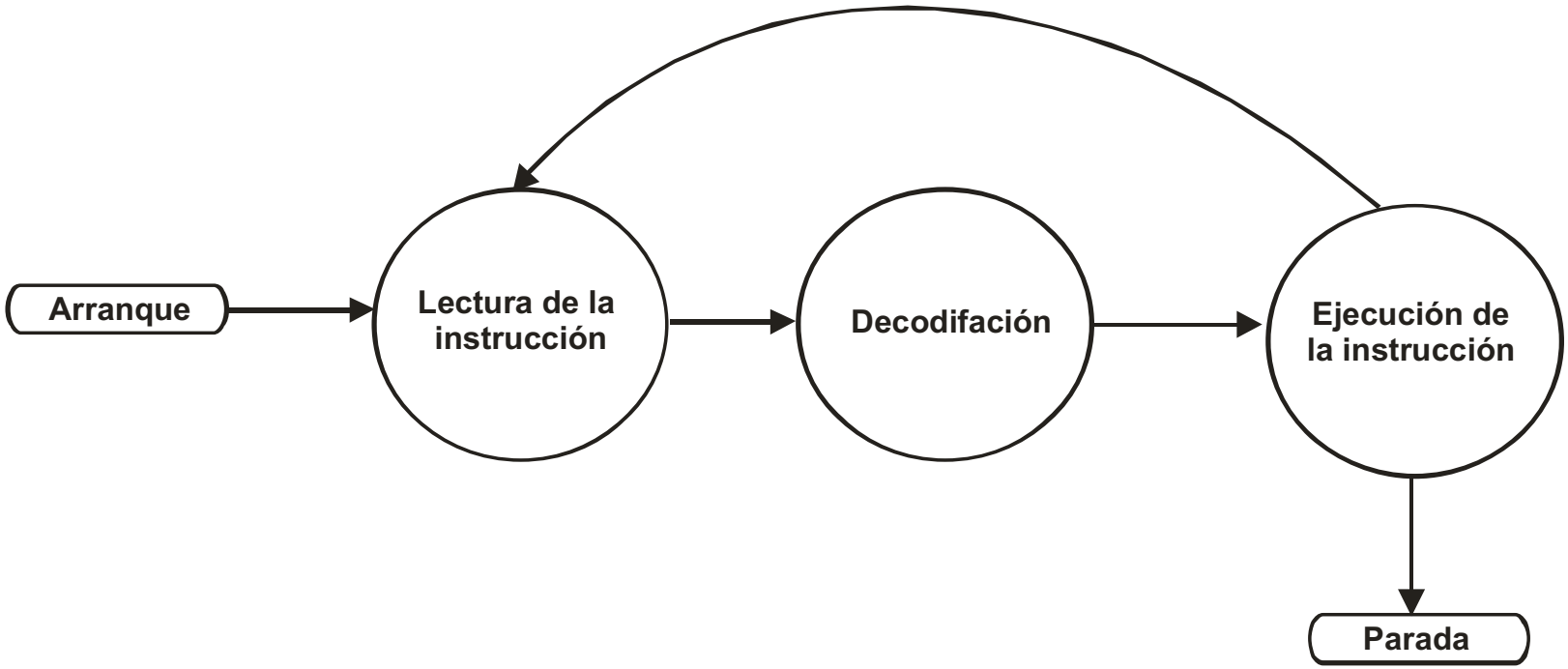
if (SelP1 == 0 AND SelP0 == 1)
Output = C V N Z I U'

Estructura de un computador elemental

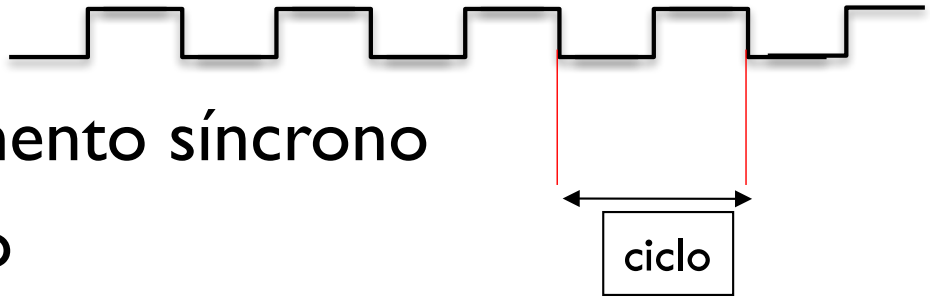


Unidad de control
Fases de ejecución de una instrucción

- Funciones básicas
  - Lectura de instrucciones de la memoria
  - Decodificación
  - Ejecución de instrucciones



Reloj



- Un computador es un elemento síncrono
- Controla el funcionamiento
- El reloj temporiza las operaciones
  - En un ciclo de reloj se ejecutan una o más operaciones elementales siempre que no haya conflicto
  - Durante el ciclo se mantienen activadas las señales de control necesarias
- En un mismo ciclo se puede realizar
  - MAR ← PC y RT3 ← RT2 + RT1
- En un mismo ciclo no se puede realizar
  - MAR ← PC y R1 ← RT3 ¿por qué?

Ejercicio

- ¿Cuál es la duración del ciclo de un computador con una frecuencia de reloj de 1 GHz?

Descripción de la actividad de la U.C.

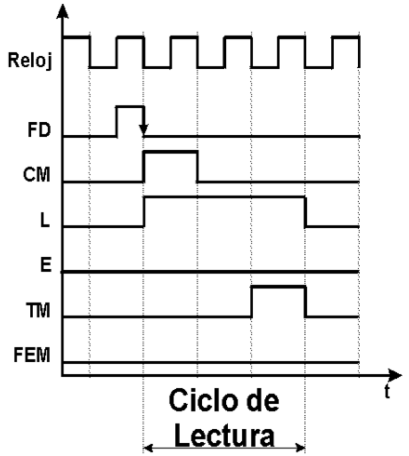
Instrucción

mv R0 R1

Secuencia de **operaciones elementales**

- RI <- [PC]
- PC++
- decodificación
- *R0 <- R1*

Secuencia de **señales de control** por cada operación elemental



+ nivel de detalle Hw.

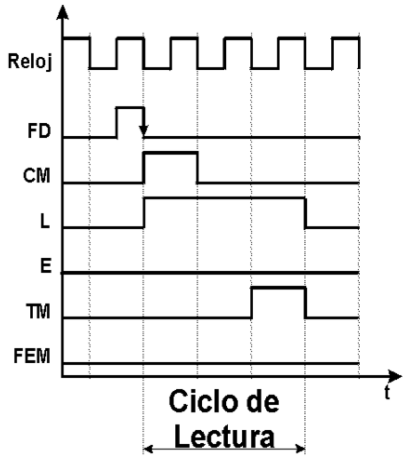
Descripción de la actividad de la U.C.

Instrucción *mv R0 R1*

Secuencia de **operaciones elementales**

- RI <- [PC]
- PC++
- decodificación
- *R0 <- R1*

Secuencia de **señales de control** por cada operación elemental



Fases de ejecución de una instrucción

- Lectura de la instrucción, captación o *fetch*
  - Leer la instrucción almacenada en la dirección de memoria indicada por PC y llevarla a RI.
  - Incremento del PC
- Decodificación
  - Análisis de la instrucción en RI para determinar:
    - La operación a realizar.
    - Direccionamiento a aplicar.
    - Señales de control a activar
- Ejecución
  - Generación de las señales de control en cada ciclo de reloj.

Lectura de una instrucción

Ciclo	Op. Elemental
C1	MAR ← PC
C2	PC ← PC + 4
C3	MBR ← MP
C4	IR← MBR

Ciclo	Op. Elemental
C1	MAR ← PC
C2	PC ← PC + 4, MBR ← MP
C3	IR← MBR

Posibilidad de operaciones simultáneas

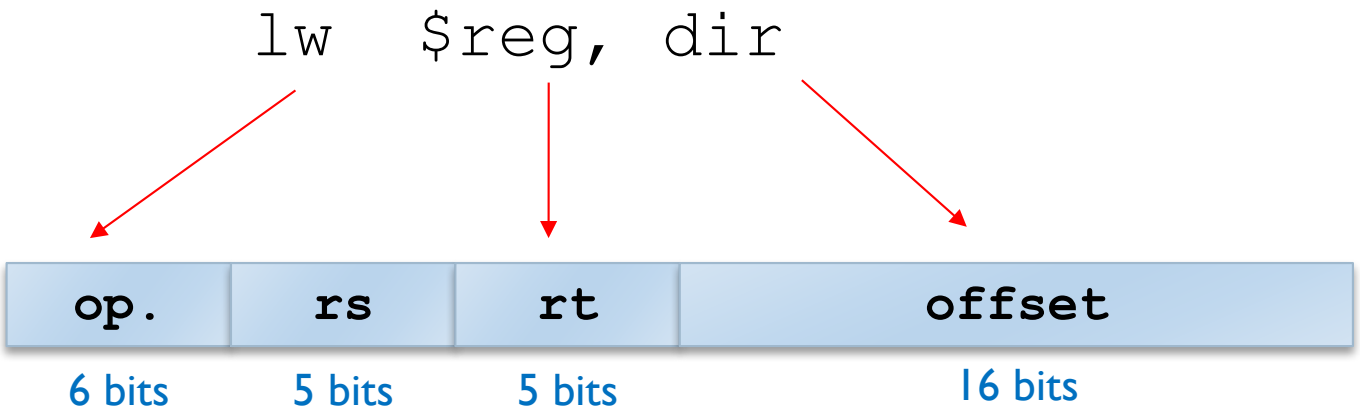
Señales de control del ciclo de fetch

- Especificación de las señales de control activas en cada ciclo de reloj.
  - Se puede generar a partir del nivel RT.

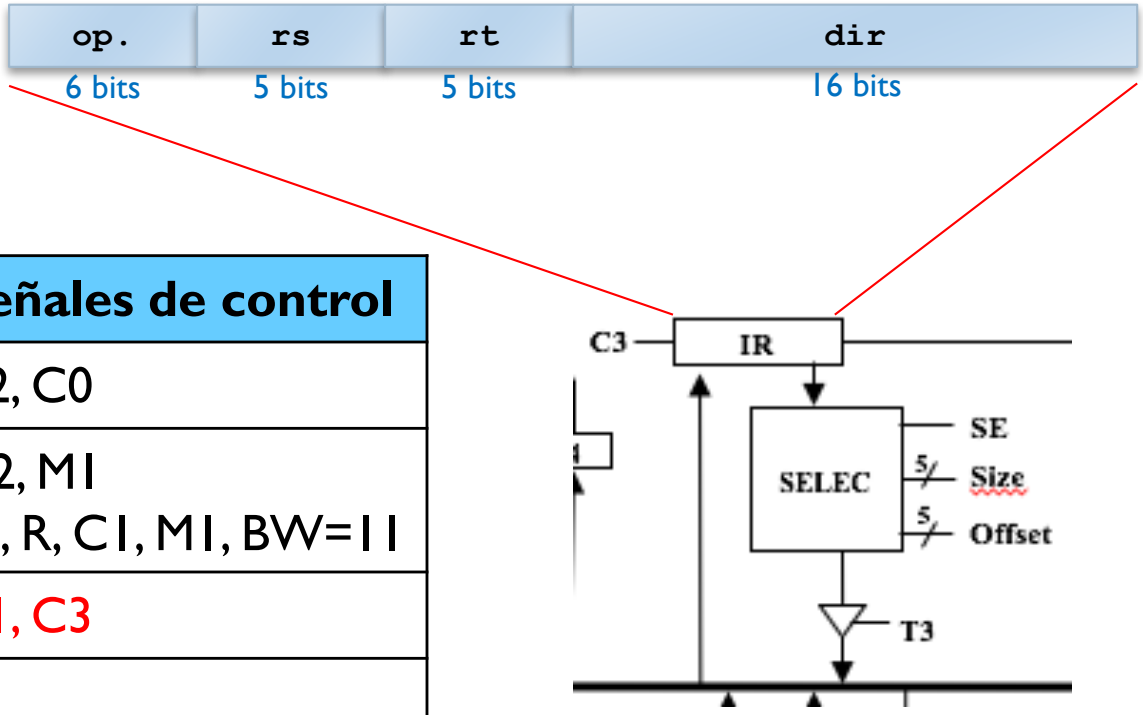
Ciclo	Op. Elemental	Señales de control activadas
C1	MAR ← PC	T2, C0
C2	PC ← PC + 4, MBR ← MP	C2, M1 Ta, R, C1, M1, BW=11
C3	IR← MBR	T1, C3

Ejecución de la instrucción de MIPS

- `lw  $reg, dir`



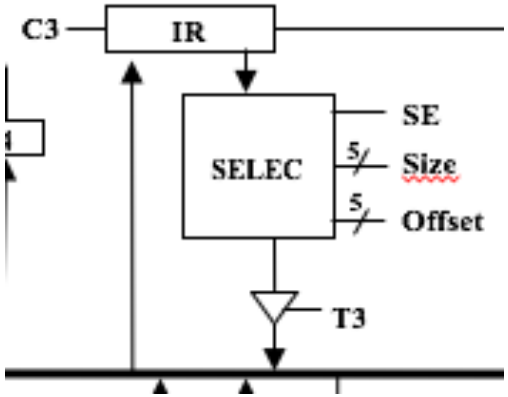
Ejecución de `lw $reg, dir`



Ciclo	Op. Elemental	Señales de control
C1	MAR ← PC	T2, C0
C2	PC ← PC + 4, MBR ← MP	C2, M1 Ta, R, C1, M1, BW=11
C3	IR← MBR	T1, C3
C4		
C5		
C6		
C7		

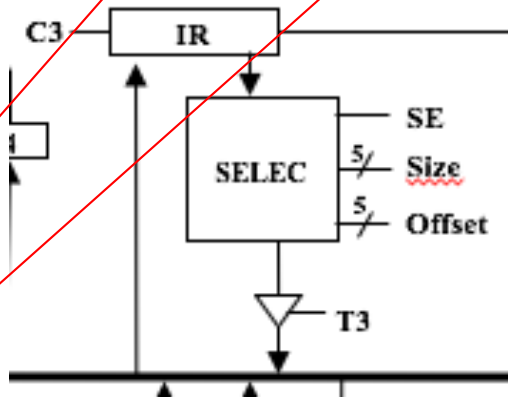
Ejecución de `lw $reg, dir`

Ciclo	Op. Elemental	Señales de control
C1	MAR ← PC	T2, C0
C2	PC ← PC + 4, MBR ← MP	C2, M1 Ta, R, C1, M1, BW=11
C3	IR← MBR	T1, C3
C4	Decodificación	
C5		
C6		
C7		



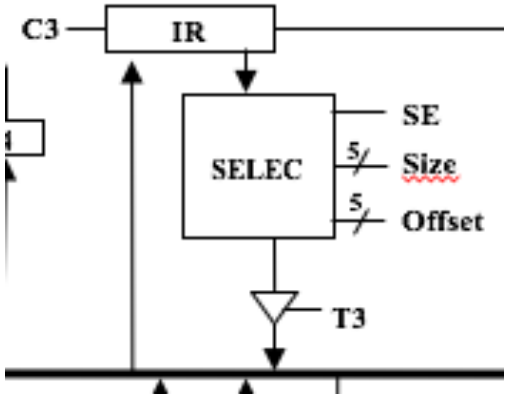
Ejecución de `lw $reg, dir`

Ciclo	Op. Elemental	Señales de control
C1	MAR ← PC	T2, C0
C2	PC ← PC + 4, MBR ← MP	C2, M1 Ta, R, C1, M1, BW=11
C3	IR← MBR	T1, C3
C4	Decodificación	
C5	MAR ← RI(dir)	C0, T3, Size =10000 Ofsset = 00000
C6		
C7		



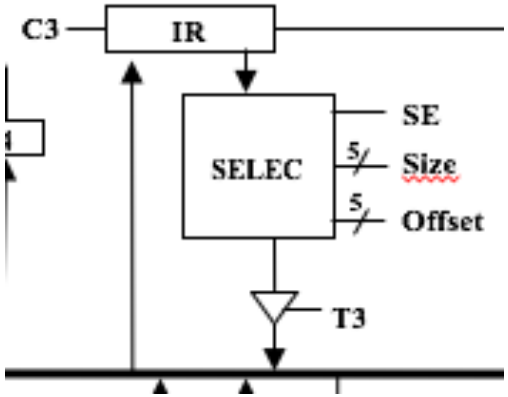
Ejecución de `lw $reg, dir`

Ciclo	Op. Elemental	Señales de control
C1	MAR ← PC	T2, C0
C2	PC ← PC + 4, MBR ← MP	C2, M1 Ta, R, C1, M1, BW=11
C3	IR← MBR	T1, C3
C4	Decodificación	
C5	MAR ← RI(dir)	C0, T3, Size =10000 Ofsset = 00000
C6	MBR ← MP	Ta, R, C1, M1, BW=11
C7		

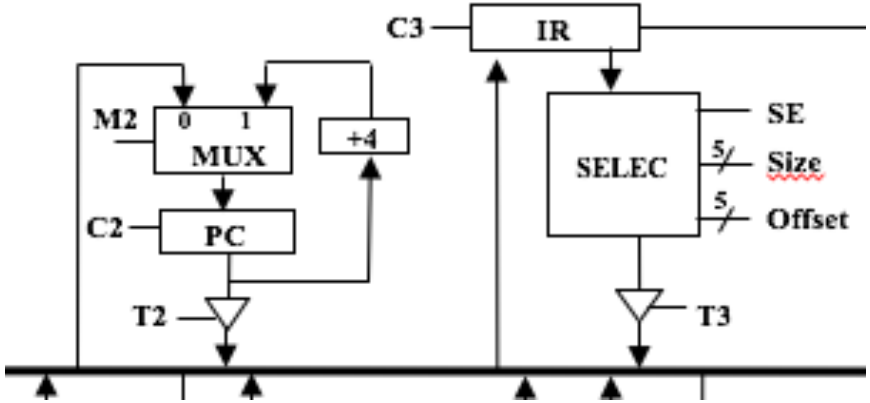


Ejecución de `lw $reg, dir`

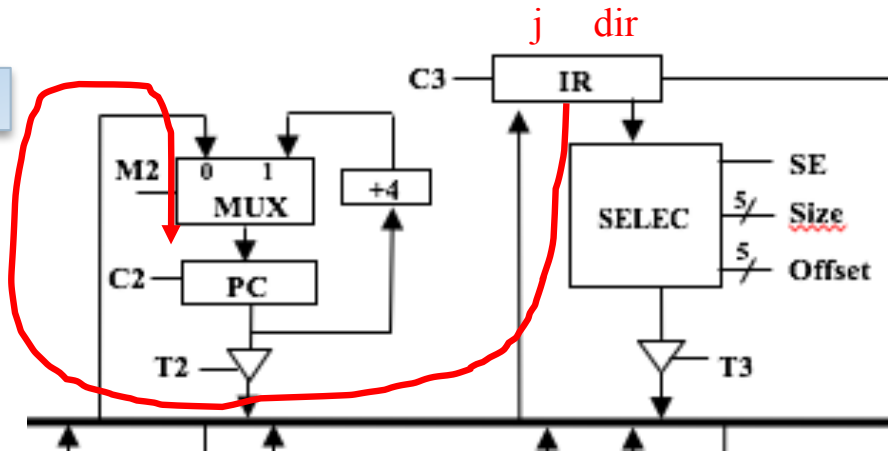
Ciclo	Op. Elemental	Señales de control
C1	MAR ← PC	T2, C0
C2	PC ← PC + 4, MBR ← MP	C2, M1 Ta, R, C1, M1, BW=11
C3	IR← MBR	T1, C3
C4	Decodificación	
C5	MAR ← RI(dir)	C0, T3, Size =10000 Ofsset = 00000
C6	MBR ← MP	Ta, R, C1, M1, BW=11
C7	$reg ←MBR	T1, RC=id de $reg LC



Ejecución de `j dir`



Ejecución de `j dir`



Ciclo	Op. Elemental	Señales de control
C1	MAR ← PC	T2, C0
C2	PC ← PC + 4, MBR ← MP	C2, M1 Ta, R, C1, M1, BW=11
C3	IR← MBR	T1, C3
C4	Decodificación	
C5	PC← RI(dir)	C2, T3, Size =11010 (26) Ofsset = 00000

Ejercicio

- Instrucciones que caben en una palabra:
  - `sw $reg, dir`
  - `add $rd, $ro1, $ro2`
  - `addi $rd, $ro1, inm`
  - `lw $reg1, desp($reg2)`
  - `j dir`
  - `jr $reg`
  - `beq $ro1, $ro2, desp`

beqz $reg, desplaz

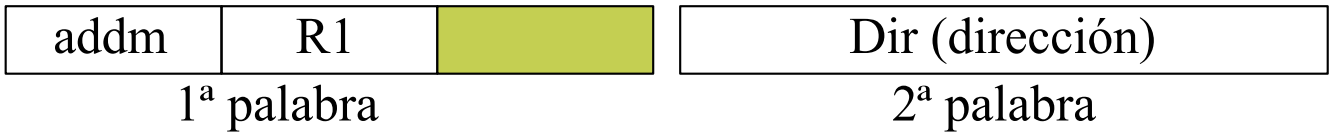
Ciclo	Op. Elemental
C1	MAR ← PC
C2	PC ← PC + 4, MBR ← MP
C3	IR←MBR
C4	Decodificación
C5	$reg + $0
C6	Si SR.Z == 0 salto a fetch
C7	RT2 ←PC
C8	RT1 ← IR(desplaz)
C9	RT1 ← RT1 * 4
C10	PC ← RT1 + RT2

Si $reg == 0
  PC ← PC + desp*4

Instrucciones que ocupan varias palabras

Ejemplo: `addm R1, dir` R1 ⟵ R1 + MP[dir]

Formato:



Ciclo	Op. Elemental
C1	MAR ← PC
C2	PC ← PC + 4, MBR ← MP
C3	IR← MBR
C4	Decodificación
C5	MAR← PC

Ciclo	Op. Elemental
C6	MBR← MP, PC ← PC + 4
C7	MAR ← MBR
C8	MBR ← MP
C9	RT1 ← MBR
C10	R1 ← R1 + RT1

Ejemplo

ADD (R_2) R_3 (R_4)

A. Fetch + Decodif.	1.- MAR ← PC 2.- RI ← Memoria(MAR) 3.- PC ← PC + "4" 4.- Decodificación de la instrucción
B. Traer operandos	5.- MAR ← R_4 6.- MBR← Memoria(MAR) 7.- RT1 ← MBR
C. Ejecutar	8.- MBR← R_3 + RT1
D. Guardar resultados	9.- MAR← R_2 10.- Memoria(MAR) ← MBR

Recordatorio

- **No** es **posible atravesar** un **registro en el ciclo** de reloj
- **No** es **posible llevar a un bus dos valores a la vez**.

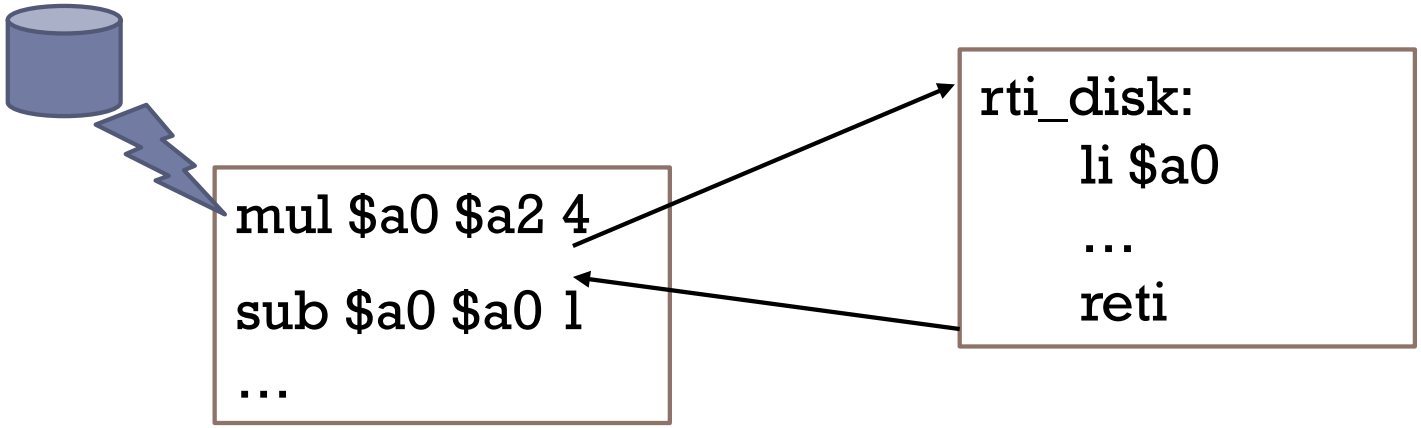
Modos de ejecución

- 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
- Se indica con un bit situado en el registro de estado (U)

Inerrupciones

- Señal que llega a la unidad de control y que rompe la secuencia normal de ejecución
- Causas:
  - Cuando ocurre un error en la ejecución de la instrucción (división por cero, …)
  - Ejecución de una instrucción ilegal
  - Acceso a una posición de memoria ilegal
  - Cuando un periférico solicita la atención del procesador
  - El reloj. Interrupciones de reloj
- Cuando se genera una interrupción se detiene el programa actual y se transfiere la ejecución a otro programa que atiende la interrupción

Idea de interrupción

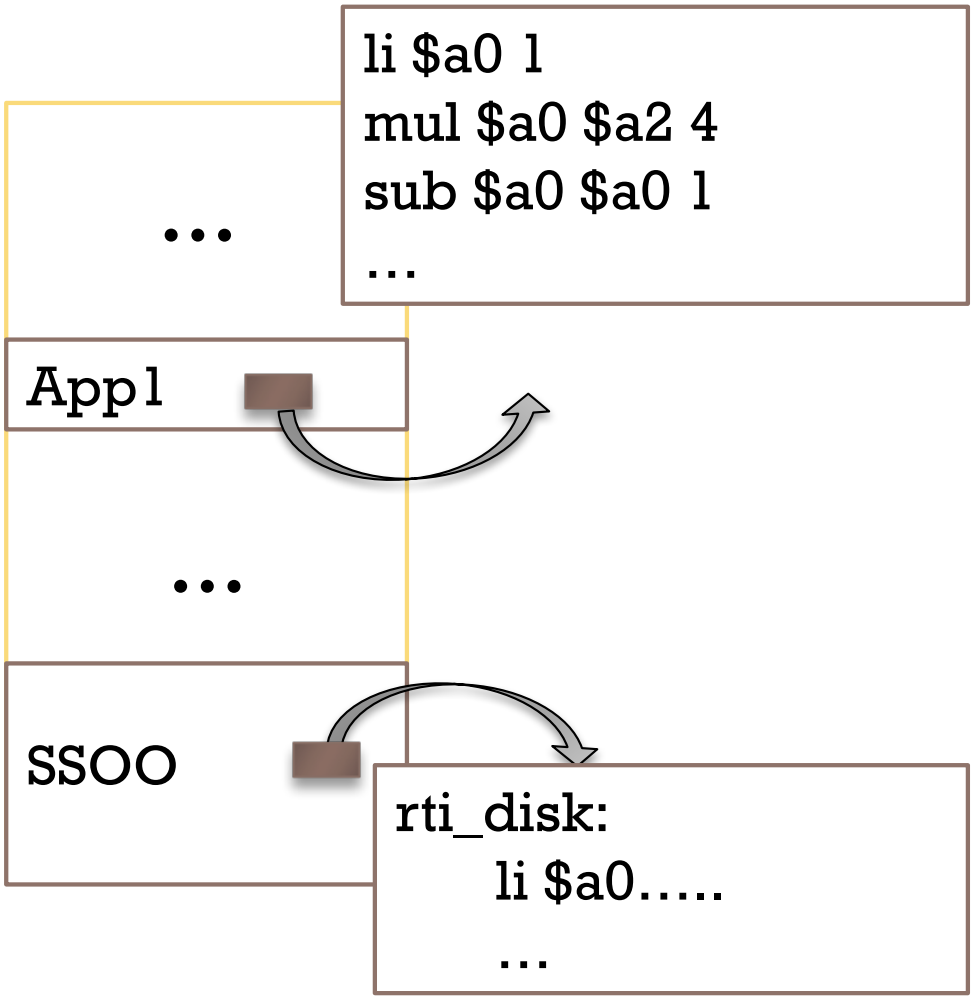


- Señal que llega a la U.C. y que rompe la secuencia normal de ejecución: se detiene el programa actual y se ejecuta otro que atiende la interrupción.
- Ejemplo de causas:
  - Cuando un periférico solicita la atención del procesador
  - Cuando ocurre un error en la ejecución de la instrucción, …

Clasificación de las interrupciones

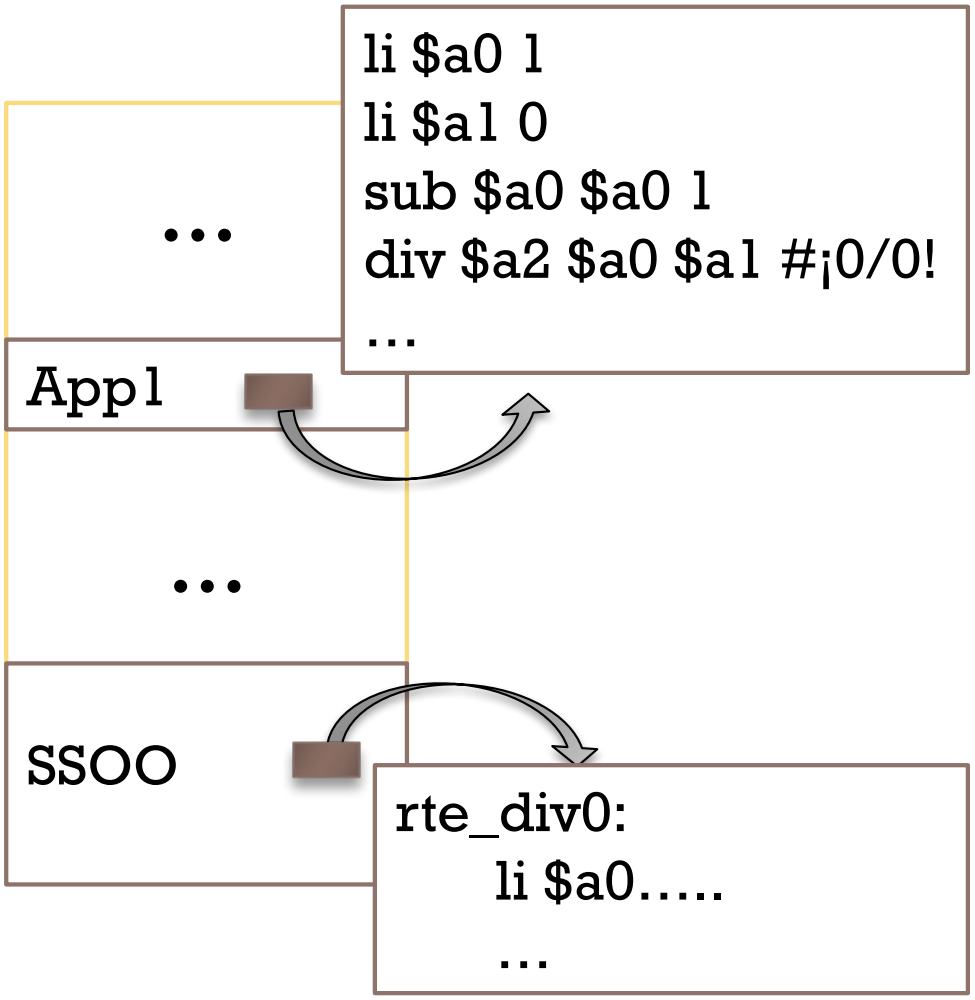
- Excepciones hardware síncronas
  - División por cero, acceso a una posición de memoria ilegal,
- Excepciones hardware asíncronas
  - Fallos o errores en el HW
- Interrupciones externas
  - Periféricos, interrupción del reloj
- Llamadas al sistema
  - Instrucciones máquina especiales que generan una interrupción para activar al sistema operativo

Excepciones hardware asíncronas e Interrupciones externas



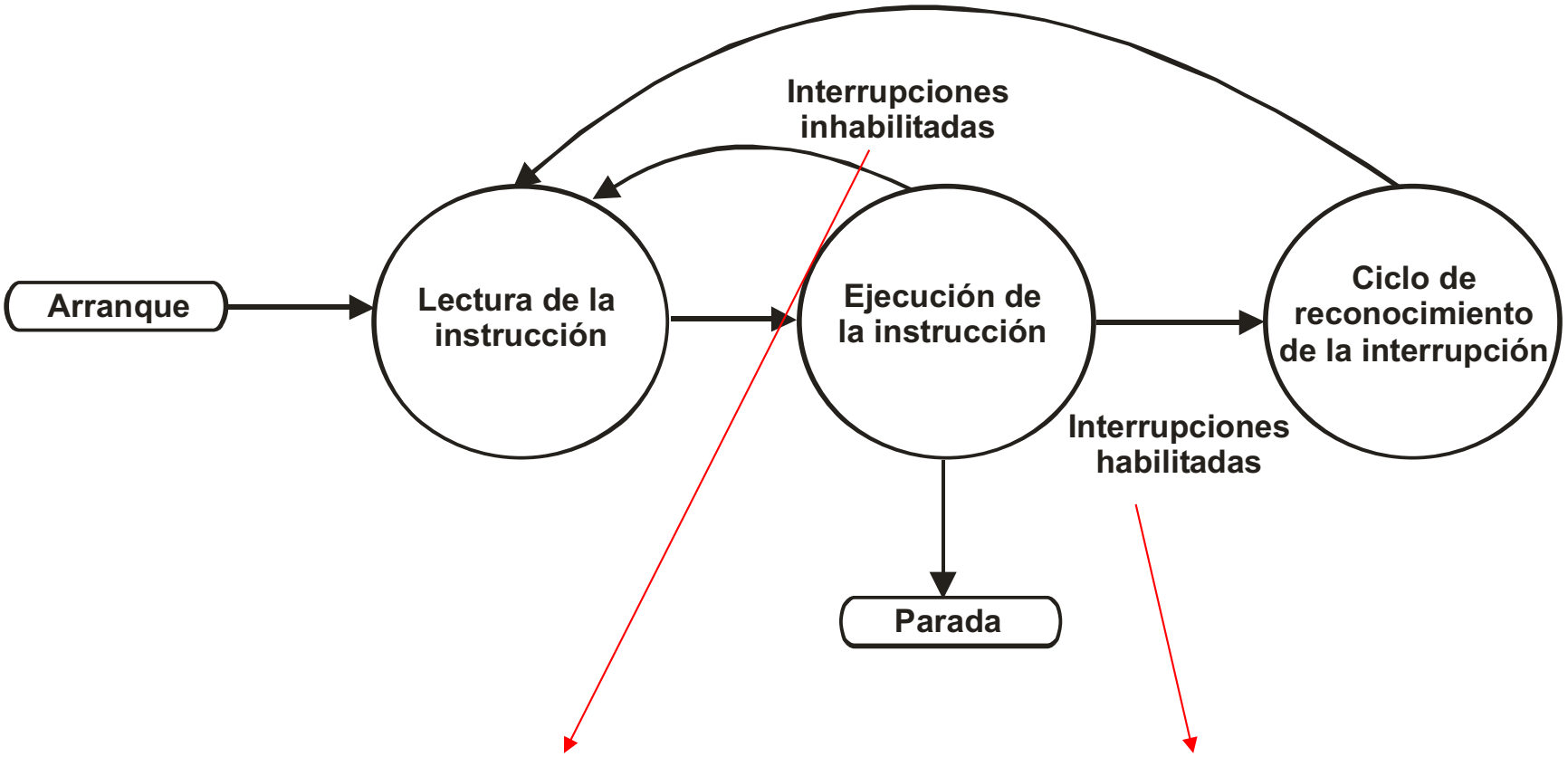
- Originan una ruptura de secuencia no programada
  - **Al final microprograma de la instrucción en curso ver si hay interrupción pendiente, y si la hay…**
  - …Bifurcación a subrutina del S.O. que la trata
- Posteriormente, restituye el estado y devuelve el control al programa interrumpido.
- **Causa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

Excepciones hardware síncronas



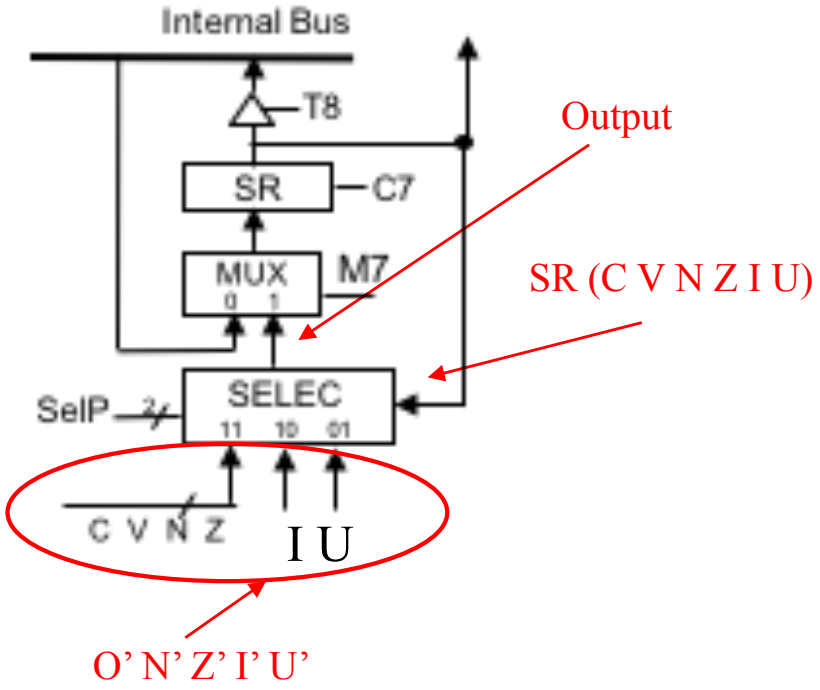
- Originan una ruptura de secuencia no programada
  - **Dentro del microprograma de la instrucción en curso…**
  - …Bifurcación a subrutina del S.O. que la trata
- Posteriormente, restituye el estado y devuelve el control al programa interrumpido **o finaliza su ejecución**
- **Causa síncrona a la ejecución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

Tratamiento de las interrupciones



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

Activación del 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'

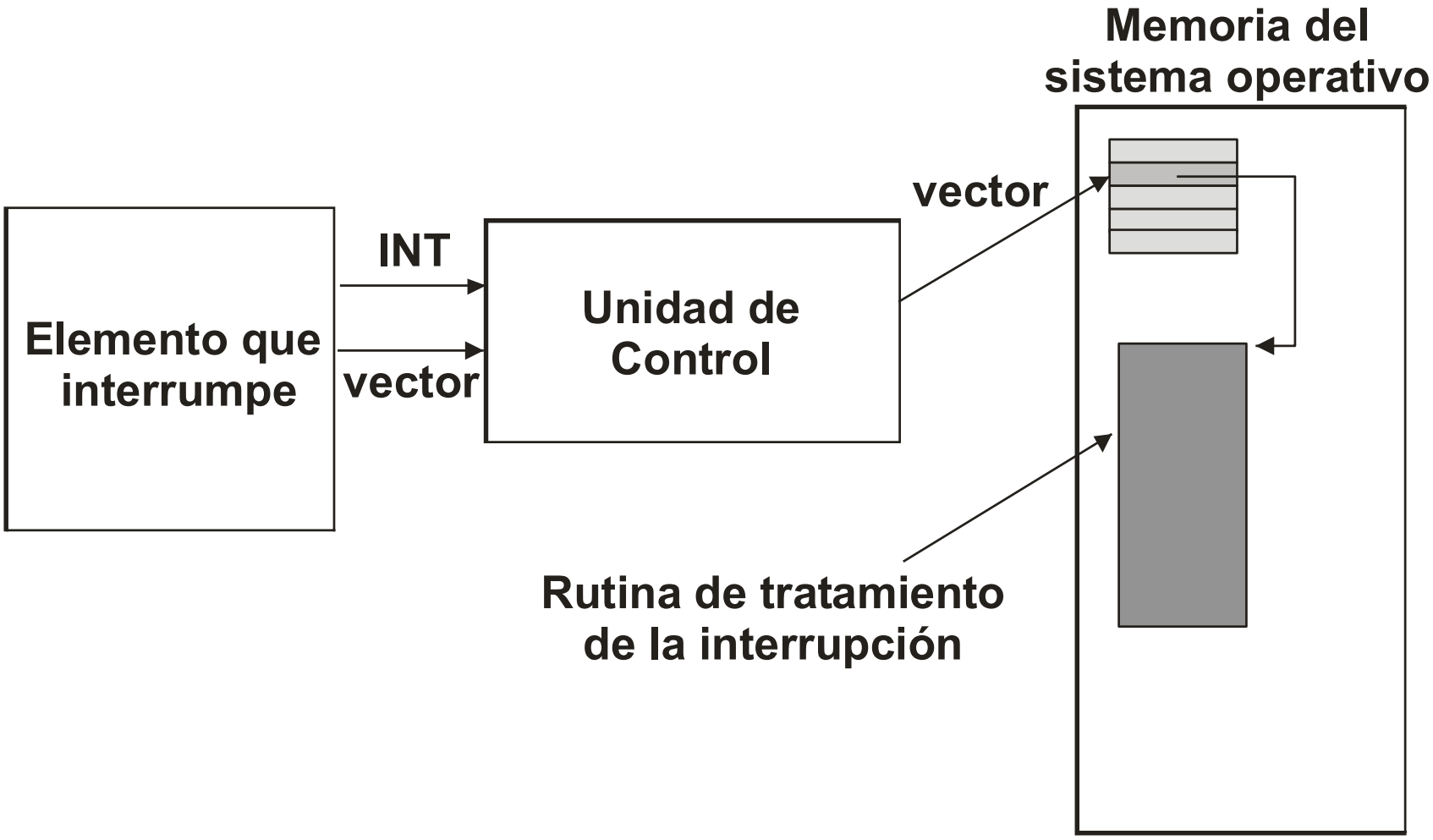
Ciclo de reconocimiento de la interrupción

- Durante este ciclo la Unidad de control realiza los siguientes pasos:
  - Comprueba se hay activada una señal de interrupción.
  - Si está activada:
    - Salva el contador de programa y el registro de estado
    - Pasa de modo usuario a modo núcleo
    - Obtiene la dirección de la rutina de tratamiento de la interrupción
    - Almacena en el contador de programa la dirección obtenida (de esta forma la siguiente instrucción será la de la rutina de tratamiento)

Rutina de tratamiento de la interrupción

- Forma parte del código del sistema operativo
- Salva el resto de registros del procesador
- Atiende la interrupción
- Restaura los registros del procesador utilizados por el programa interrumpido
- 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).

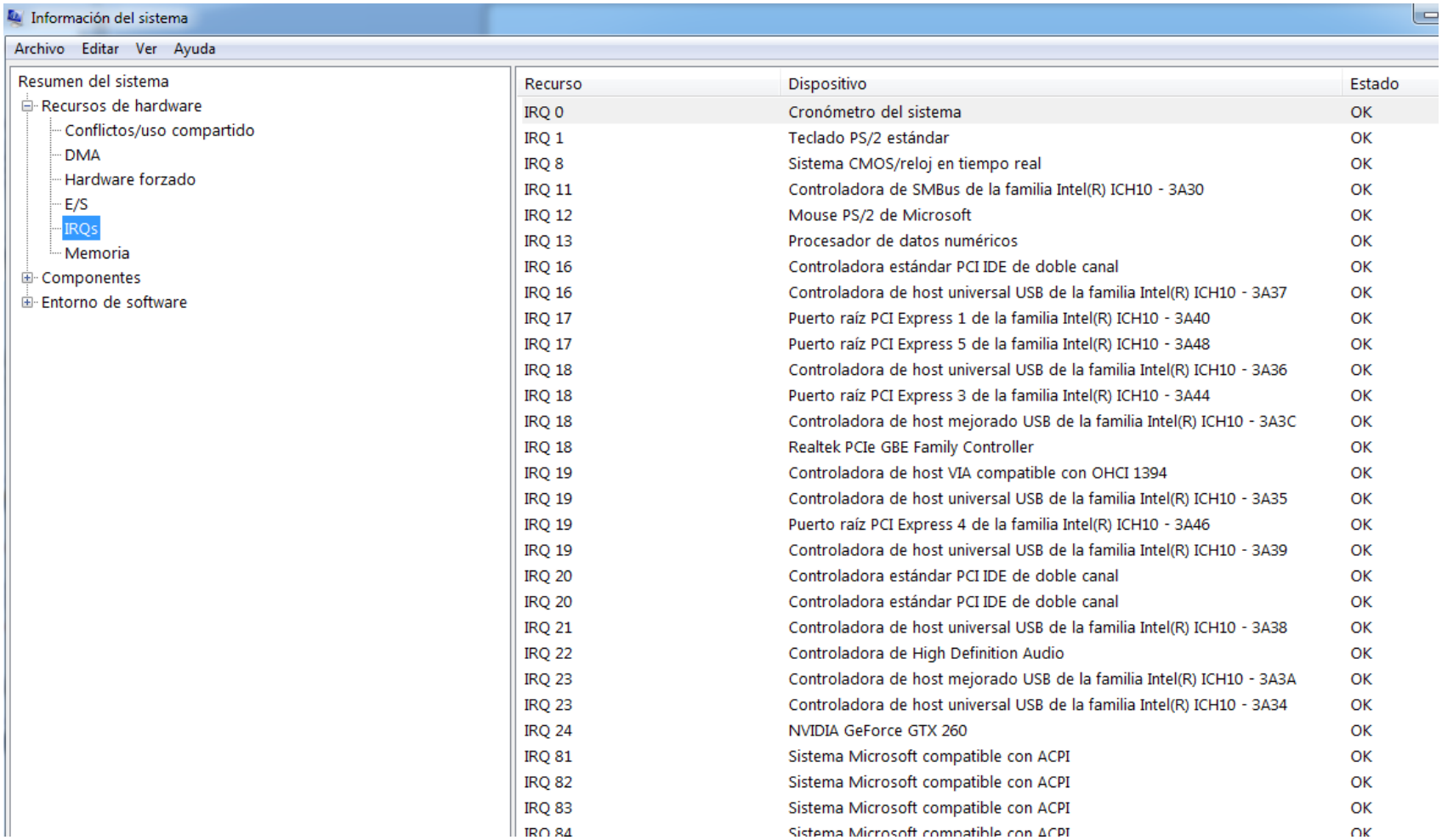
Interrupciones vectorizadas



Inerrupciones 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

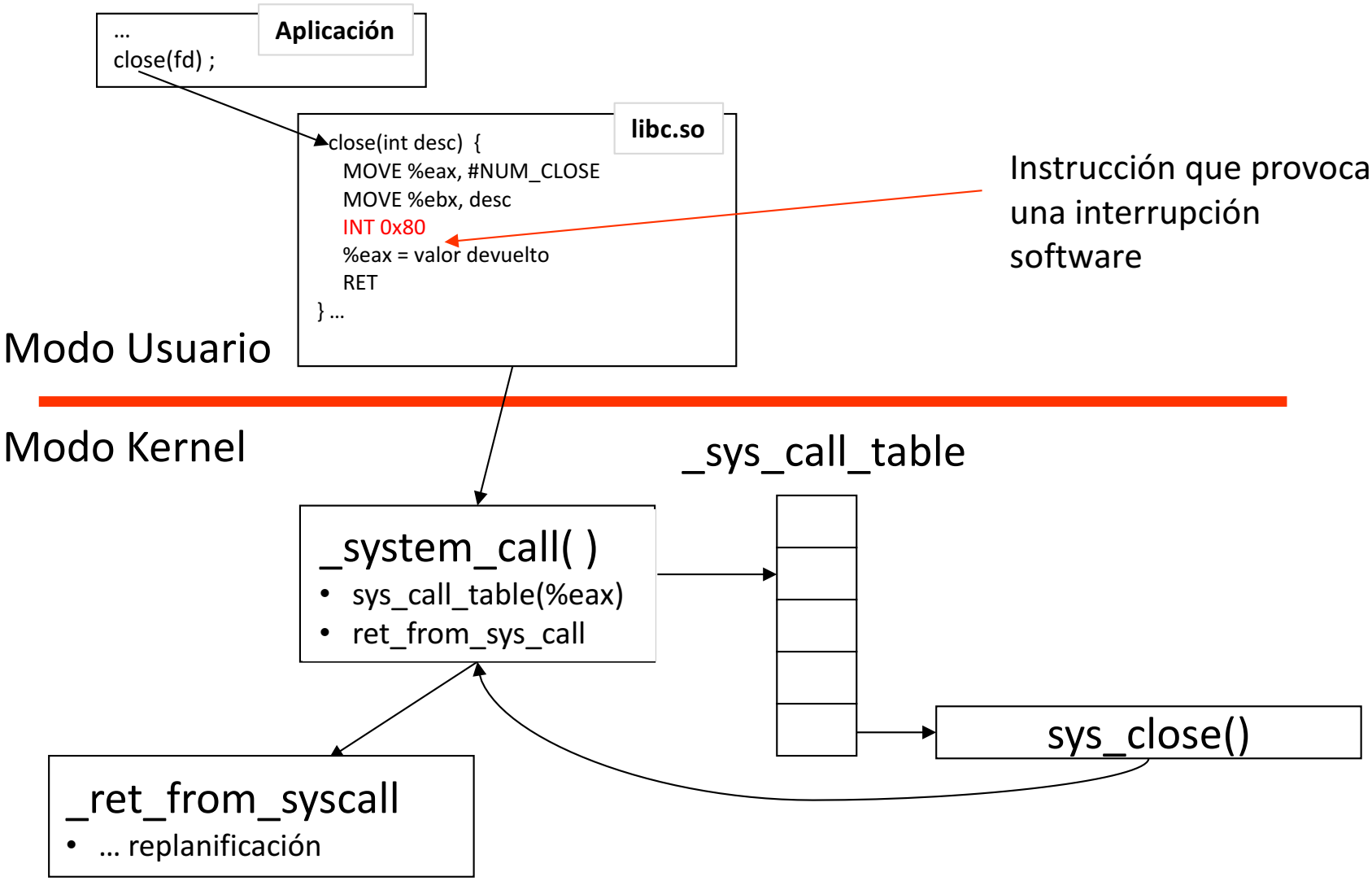


Recurso	Dispositivo	Estado
IRQ 0	Cronómetro del sistema	OK
IRQ 1	Teclado PS/2 estándar	OK
IRQ 8	Sistema CMOS/reloj en tiempo real	OK
IRQ 11	Controladora de SMBus de la familia Intel(R) ICH10 - 3A30	OK
IRQ 12	Mouse PS/2 de Microsoft	OK
IRQ 13	Procesador de datos numéricos	OK
IRQ 16	Controladora estándar 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 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
- Similar a las llamadas al sistema que ofrece el simulador QtSPIM

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

Ejercicio

- Señales de control a activar en caso de que se haya producido una interrupción
  - Se obtiene el vector de interrupción del bus de datos