Introducció als Ordinadors:
Capítol 2:
REGISTRES DE LA CPU

Registres

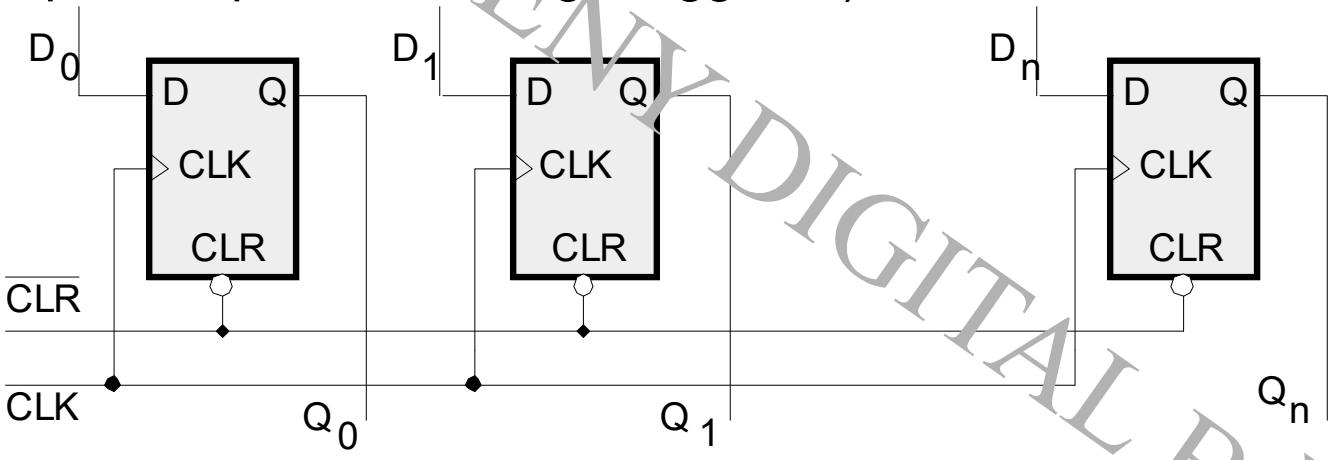
- Conjunt d'elements biestables governats per el mateix clock.
- Un registre és una memòria d'alta velocitat i poca capacitat integrada en el processador que permet guardar temporalment i accedir a dades

•Sincronisme:

•Latch: actiu per nivells (no pot utilitzar-se per operacions del tipus

Rj <=f(Rj, Ri)

•Flip-Flop: actiu per flancs (edge-triggered)



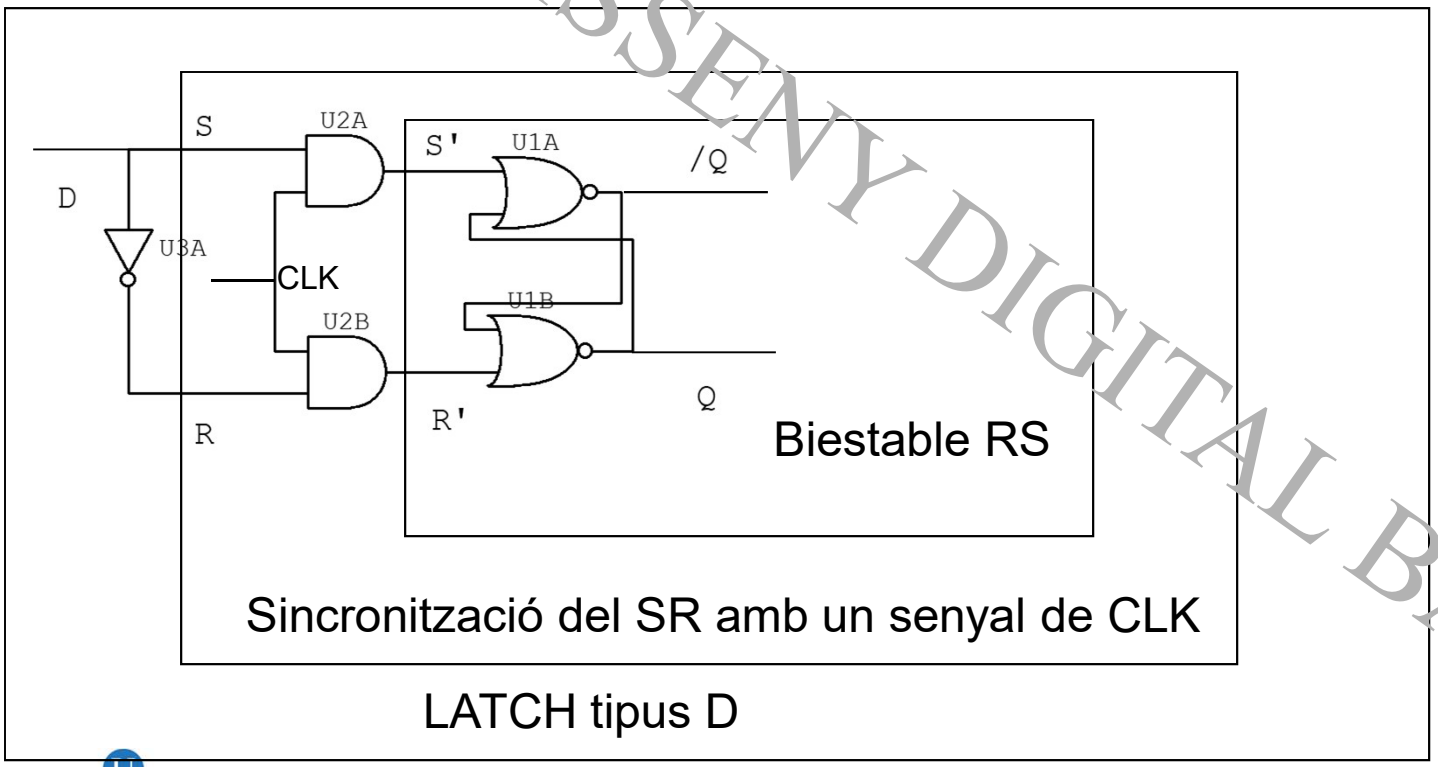
n elements = n bits

Registres

LATCH

-Quan el senyal de rellotge està habilitat, l'entrada D passa a la sortida Q

-Es fonamenta en un biestable SR, sincronitzat amb un senyal de rellotge



Clk	D	Q
0	0	Q
0	1	Q
1	0	0
1	1	1

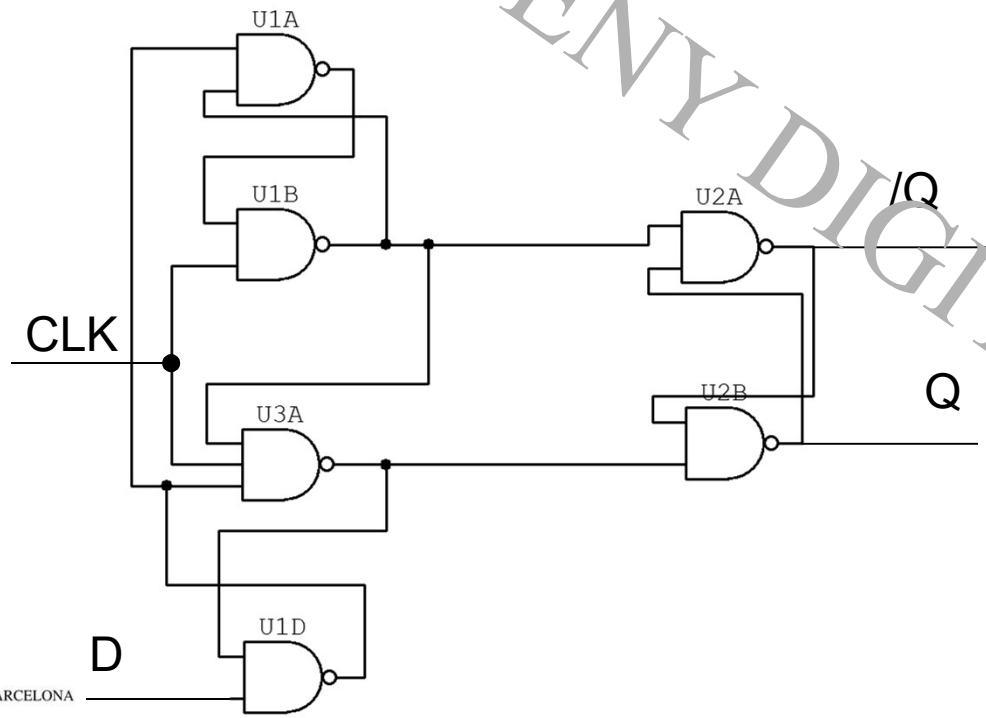
Registres

Flip-flop actiu per flanc (edge-triggered)

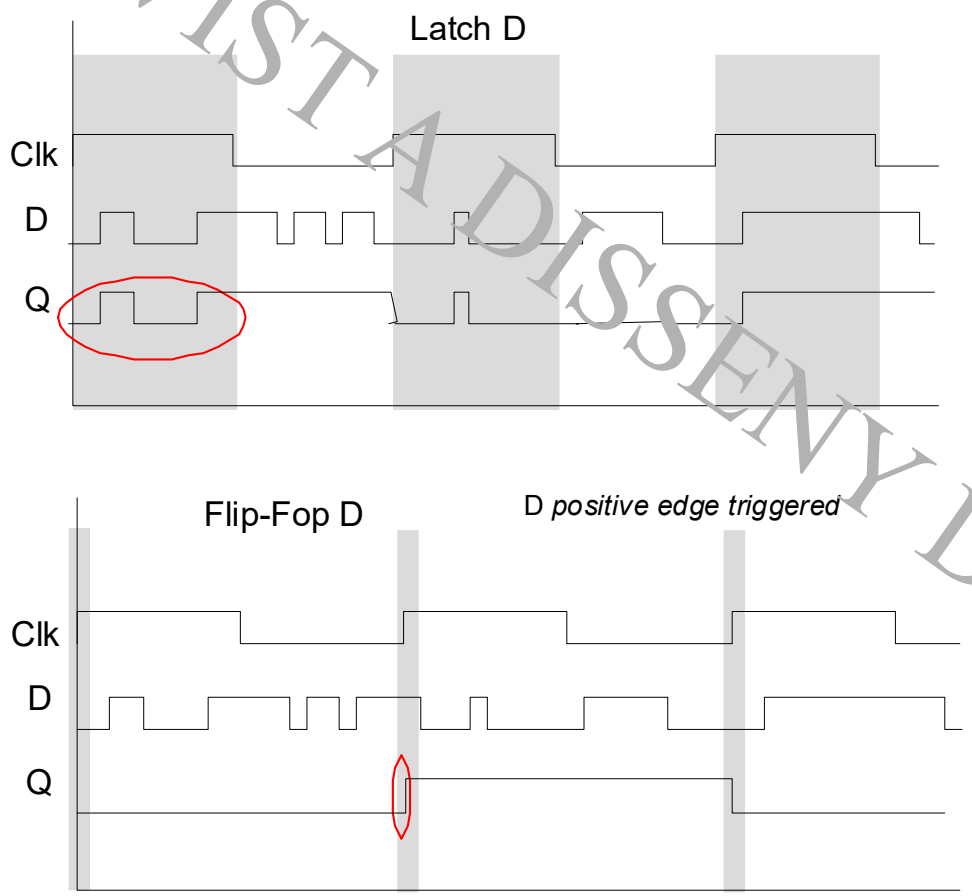
Només permet els canvis durant el temps de transició del senyal de clk
-elimina sorolls
-elimina problemes de transicions en els flip-flops

⇒

El canvi es produeix en la transició 0-1 (rise time) o 1-0 (fall time)



Latch vs. Flip-Flop



Latch

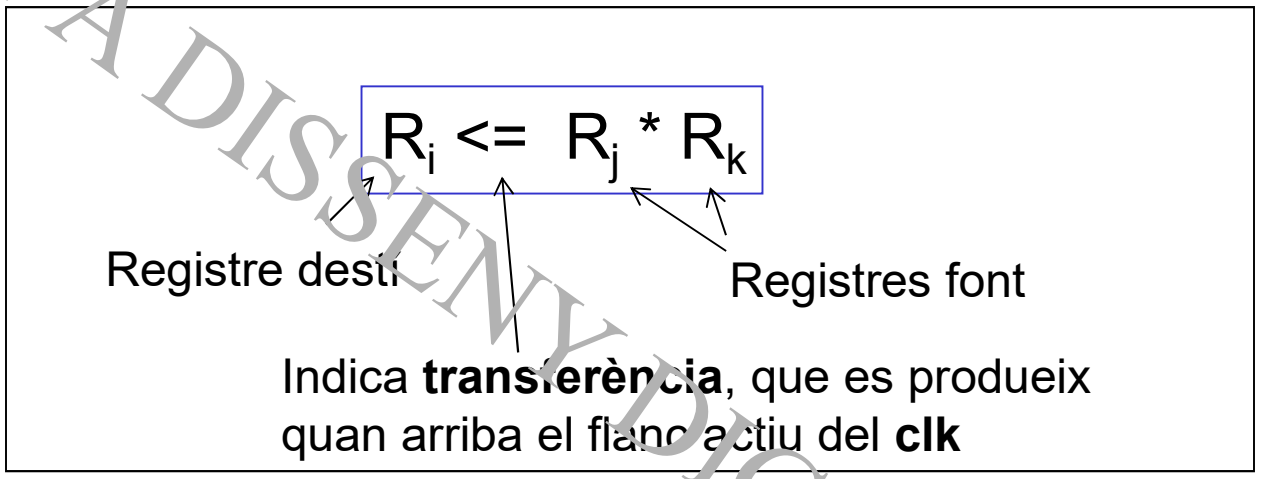
- Actiu per nivells
- Útil per guardar
- No útil per Reg.Desplaçament
- Asincrònic
- No útil per: font i destí alhora

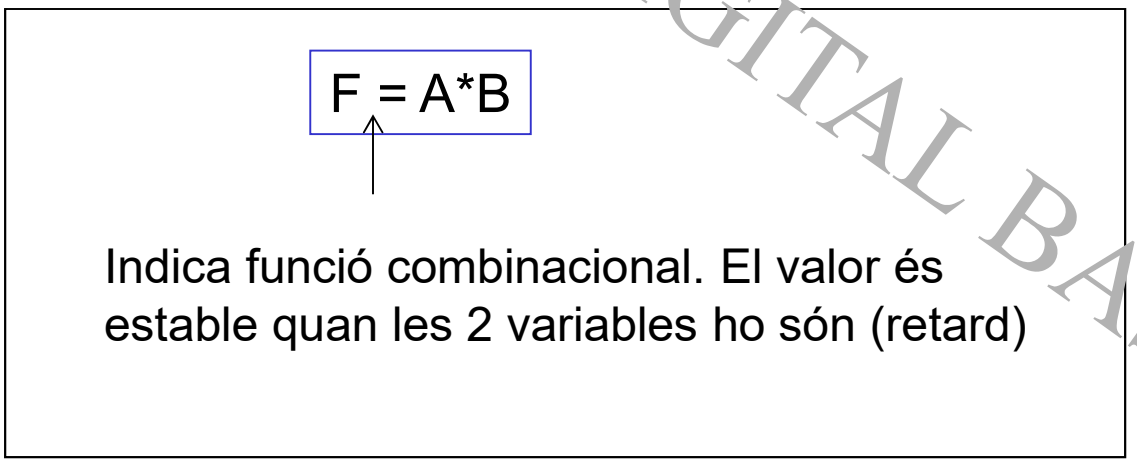
Flip-Flop

- Actiu per canvis de nivells
- Útil per guardar
- Útil per Reg.Desplaçament
- sincrònic

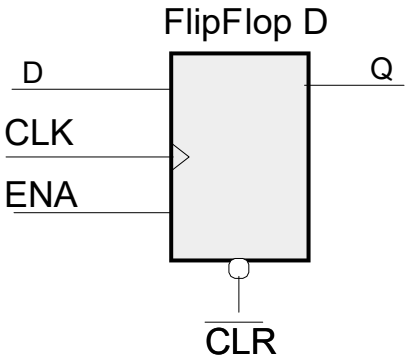
Nivell RTL (*Register Transfer Level*)

Disseny de sistemes digitals **sincrònics**
Les operacions són **RTL**





Senyals síncrons d'habilitació i RESET



Introduïm un senyal d'habilitació (ENABLE) active high
Introduïm un senyal de clear (RESET) active low

Hem de generar la funció que ens doni l'entrada D del flip flop en funció de les variables EN, /CLR i DI

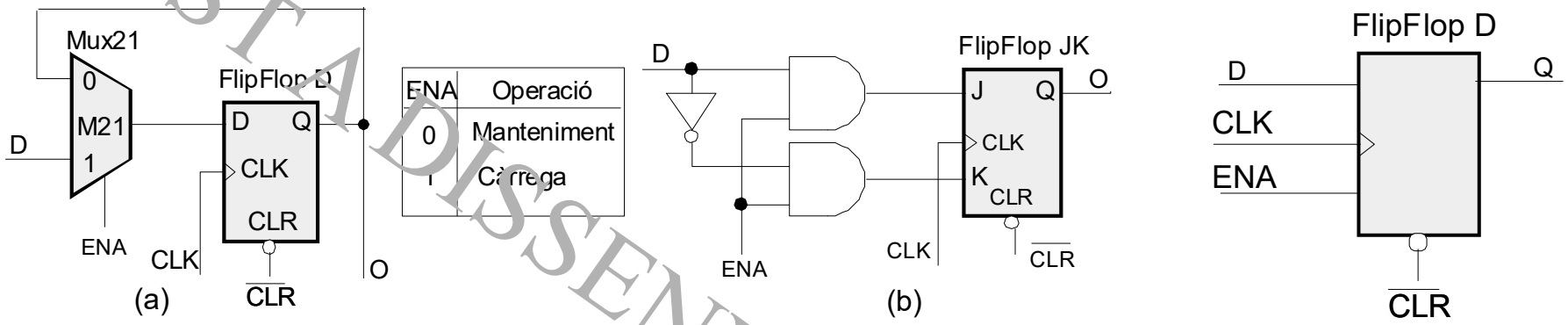
La funció és:

$$D = /EN \cdot /CLR \cdot Q + EN \cdot /CLR \cdot DI$$

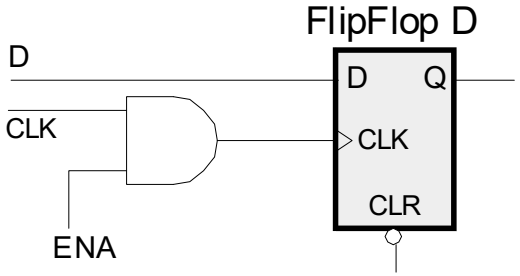
EN	/CLR	DI	D
0	0	0	0
0	0	1	0
0	1	0	Q
0	1	1	Q
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Estructura dels Registres

Funcionament correcte: governat per senyal d'habilitació



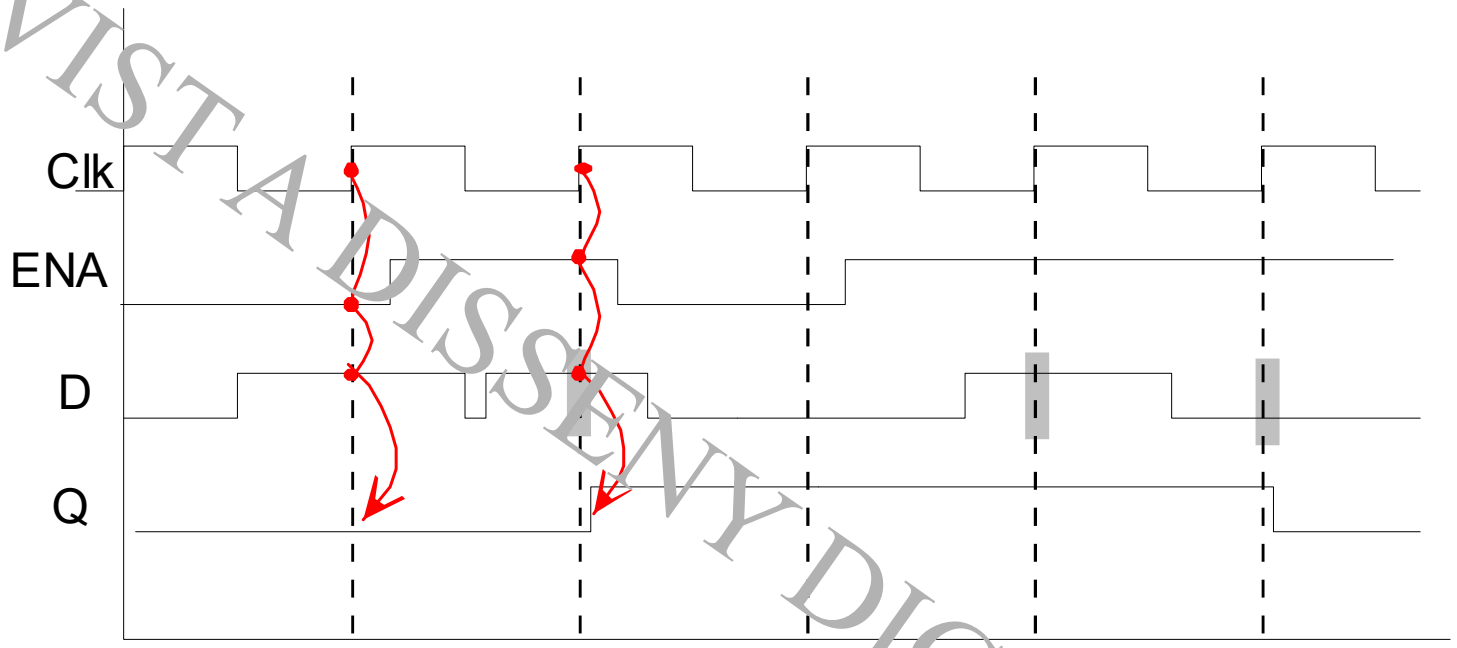
No aconsellable: habilitació actuant sobre el senyal de clk

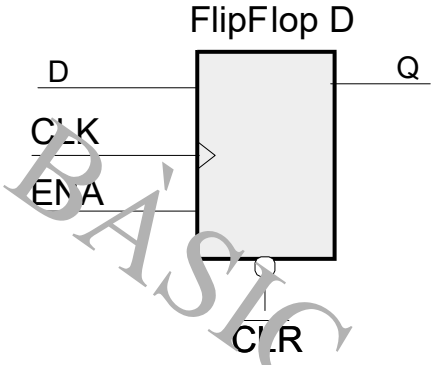


Gated clock

- Retard
- Pèrdua de sincronisme

Sincronisme en Registres

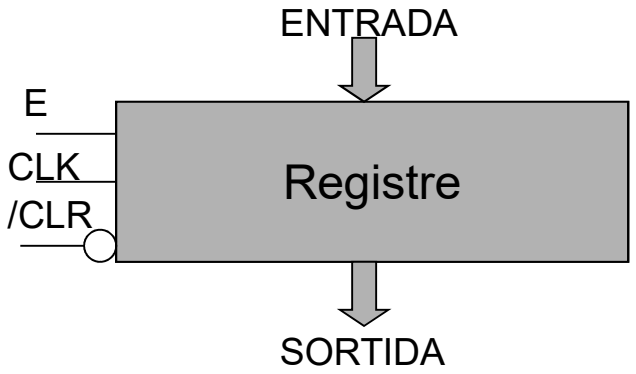


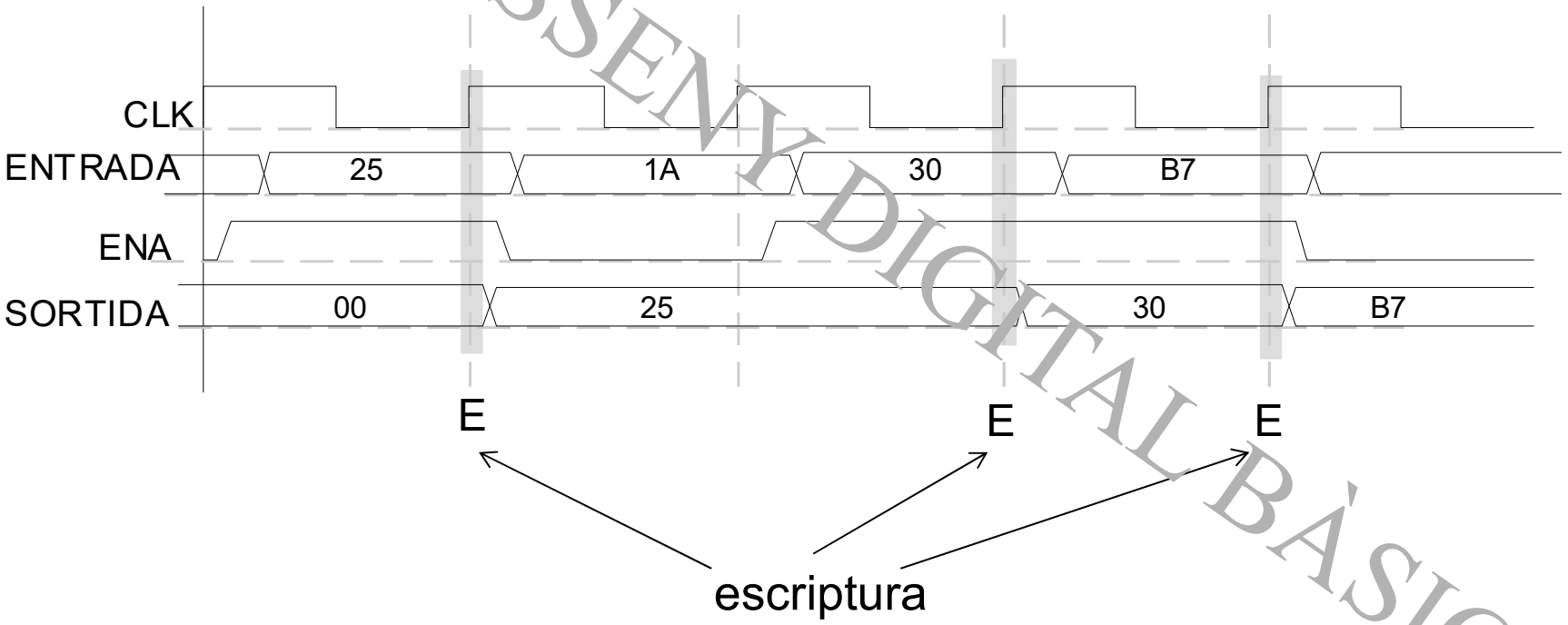


Registre governat per senyal d'habilitació (ENA)

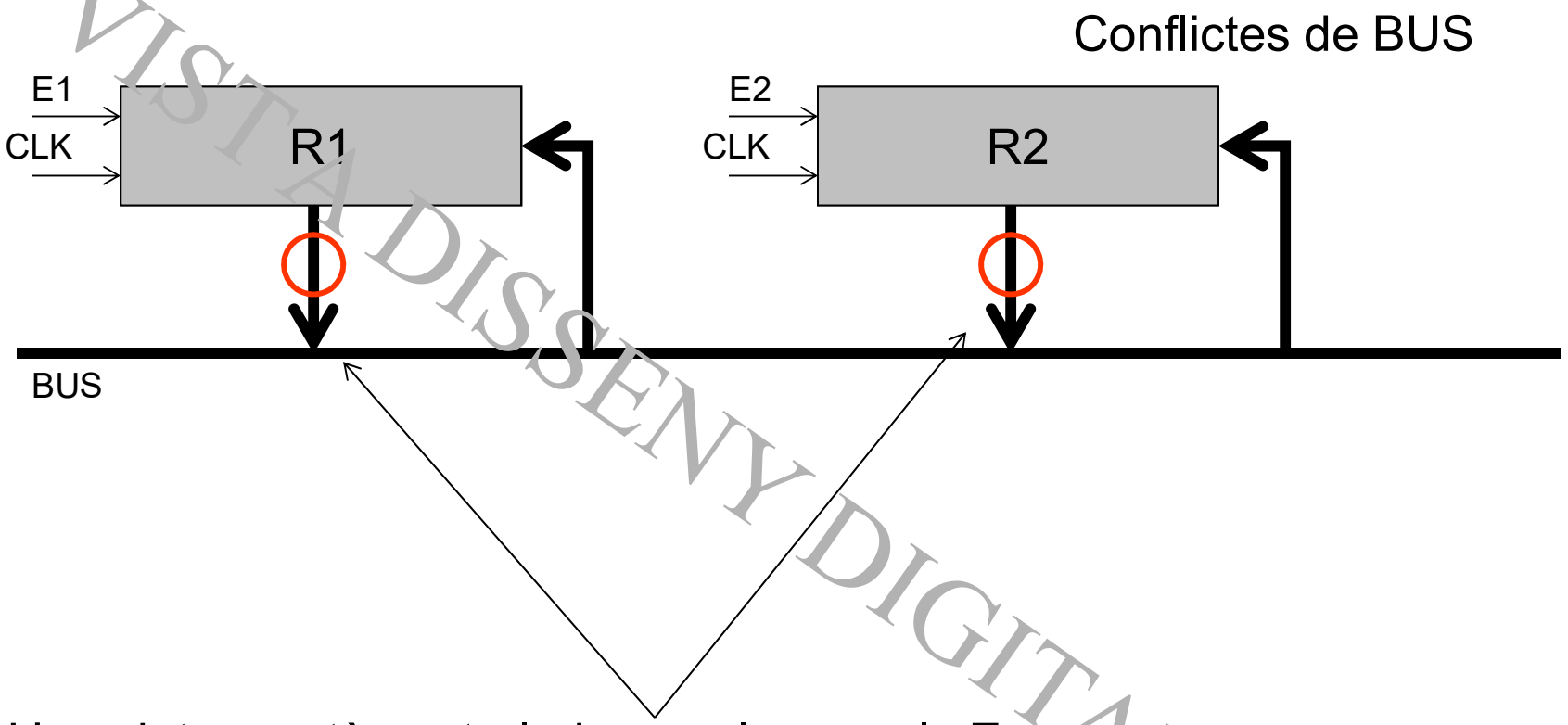
L'ESCRIPTURA és SÍNCRONA

La LECTURA és ASÍNCRONA



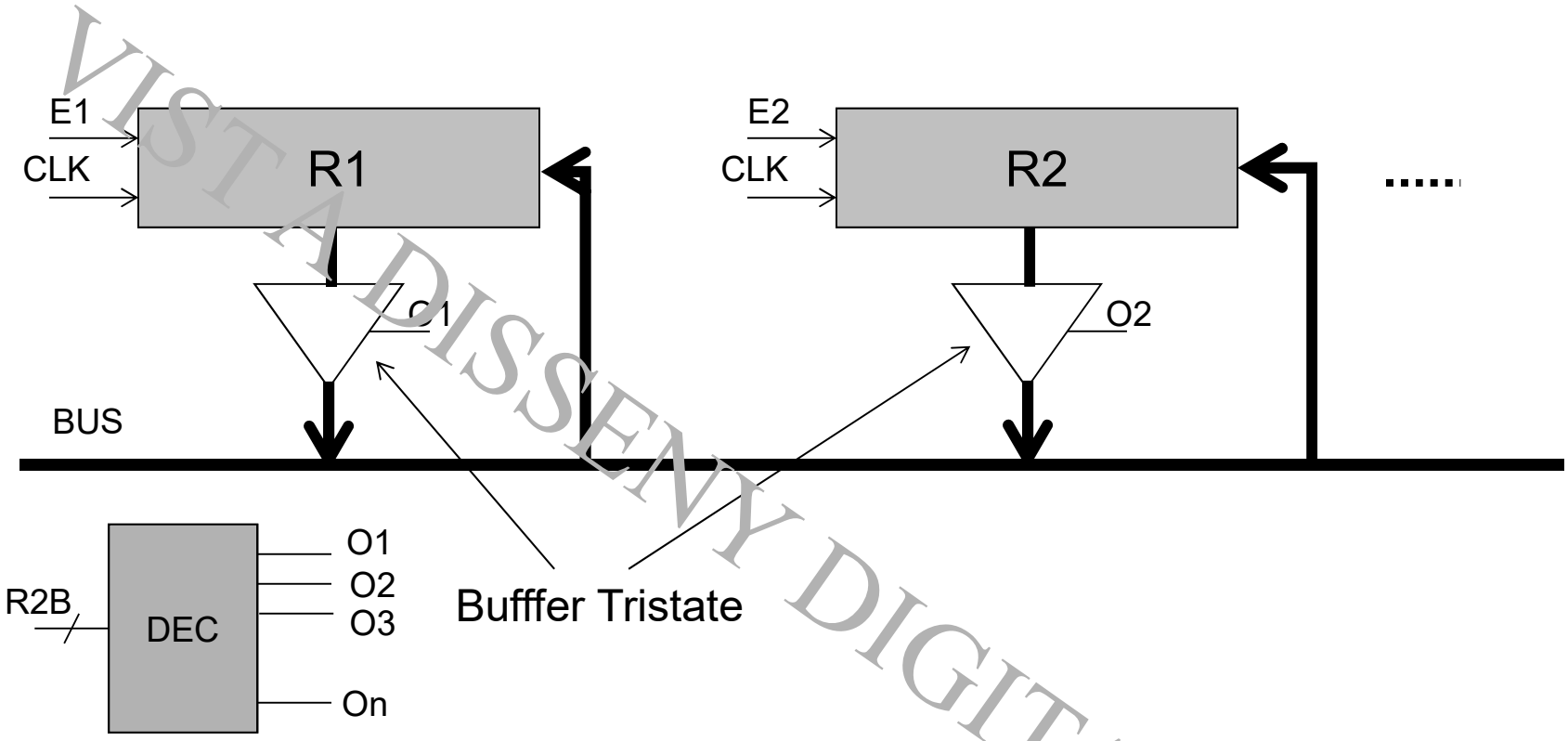


Lectura de REGISTRES

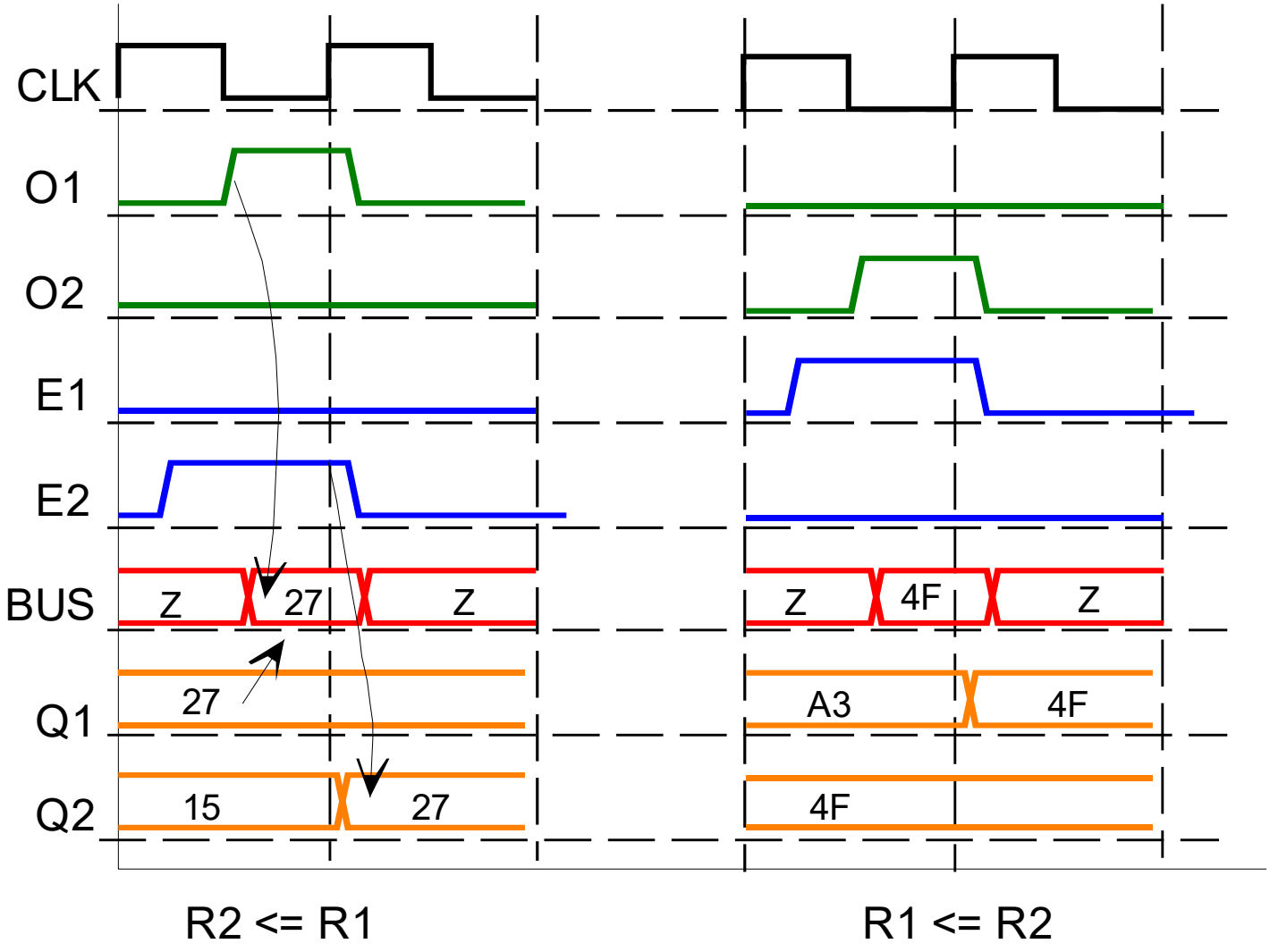


- L'escriptura està controlada per els senyals E_i.
- Però si es vol llegir de dos registres i passar les dades a BUS es produeix un **conflicte de BUS**
- Cal controlar l'accés a BUS

Multiplexat de les sortides dels registres (buffers-tristate)



Una alternativa és un MUX-n. Diferències de cost depenent del nombre de bits


CLK
O1
O2
E1
E2
BUS
Z
27
Z
Q1
27
Q2
15
27
R2 <= R1
Z
4F
Z
A3
4F
4F
R1 <= R2

Depenent de l'arquitectura del processador, el ensamblador pot actuar sobre els registres. Per exemple...

```
SUB R1,a1,a1
ADDI a1,a1,8
ADD a1,zero,a2
…
On
```

L'OPERACIÓ DE **LECTURA** DE REGISTRES ÉS **ASÍNCRONA**

L'OPERACIÓ **D'ESCRIPTURA** ÉS **SÍNCRONA**

ADD Rd, Rs1, Rs2

ADDI Rd, Rs1, Immediat

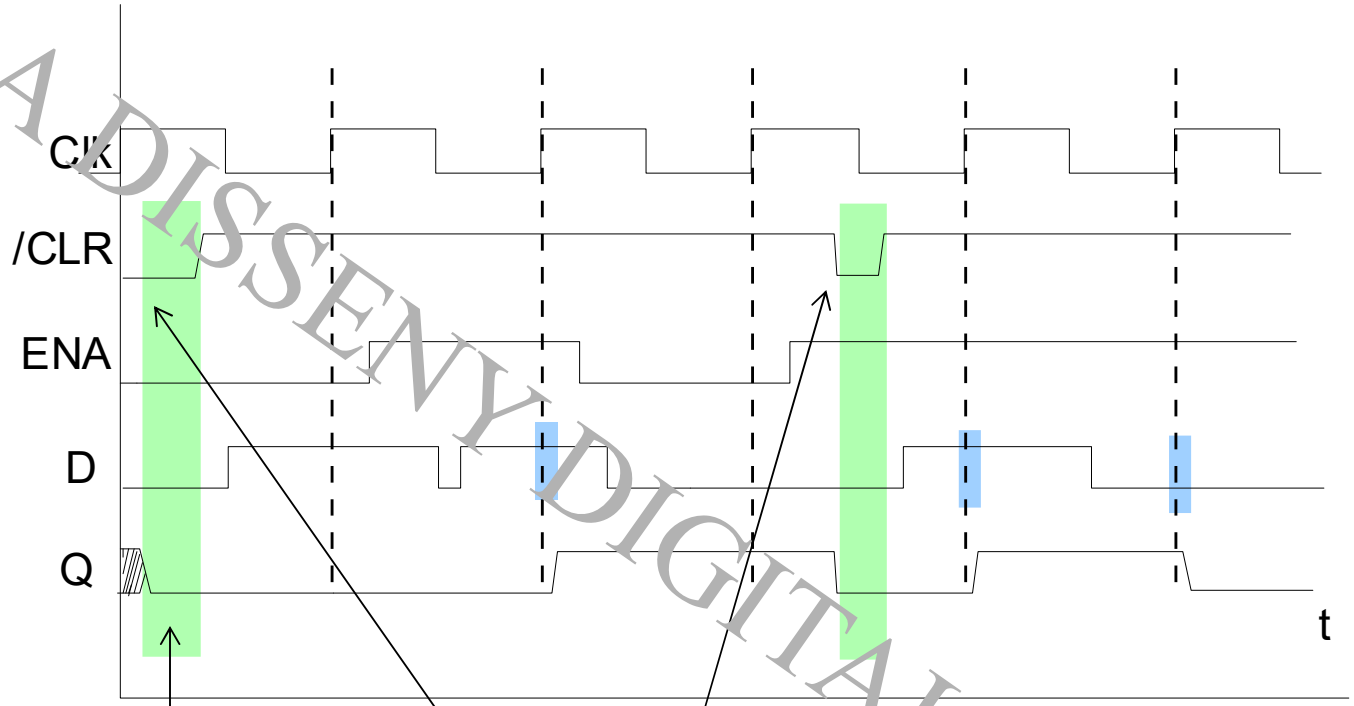
SUB Rd, Rs1, Rs2

El processador RISC-V el registre x0 sempre conté un 0
No es pot modificar

RESET/CLEAR en Registres

Generalment els flip-flops consten de RESET (CLEAR) asíncron

El CLEAR sol ser ***active low*** (/CLR)



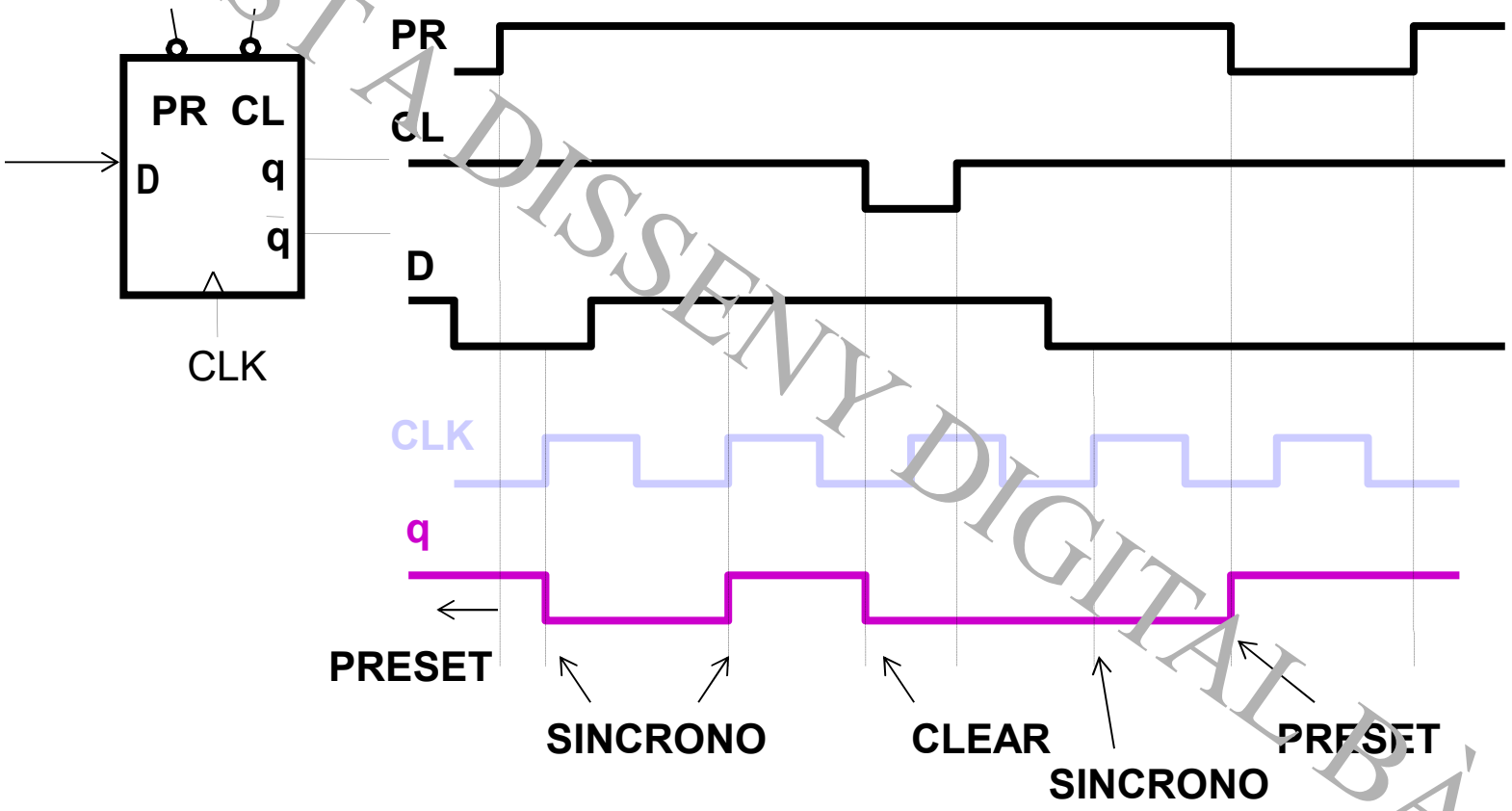
CLEAR asincrònic

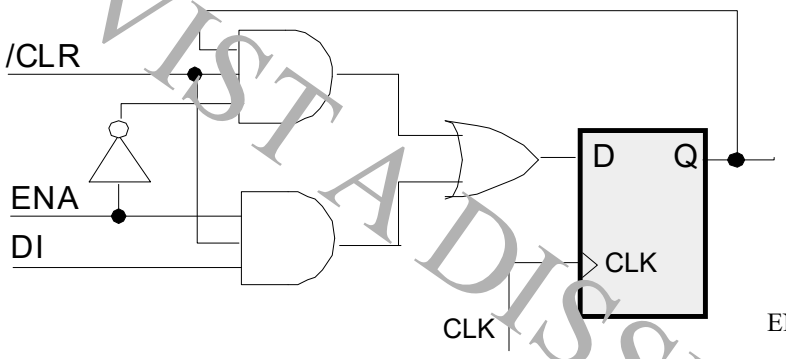
Aconsellable iniciar amb un CLEAR dels registres

De la mateixa manera pot haver-hi senyal de PRESET

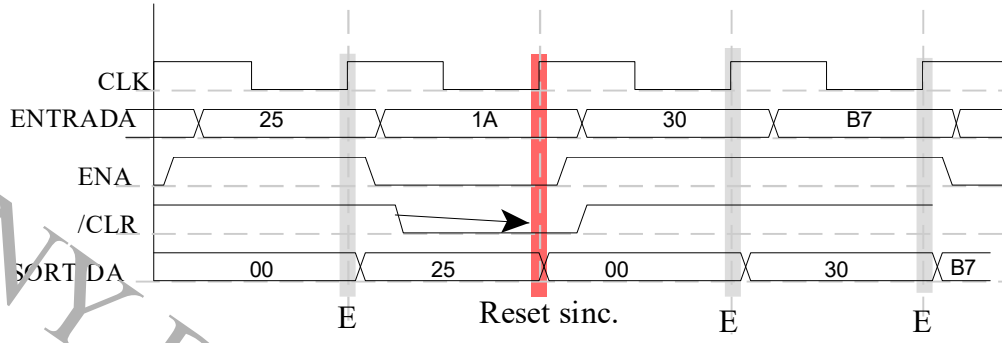
Elementos de memoria

Entradas asíncronas de los biestables



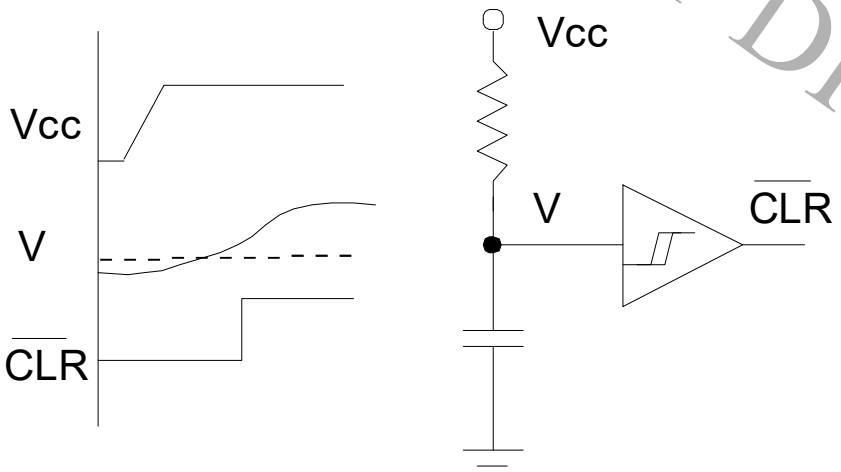


Alguns cops és aconsellable usar un senyal de CLEAR sincrònic



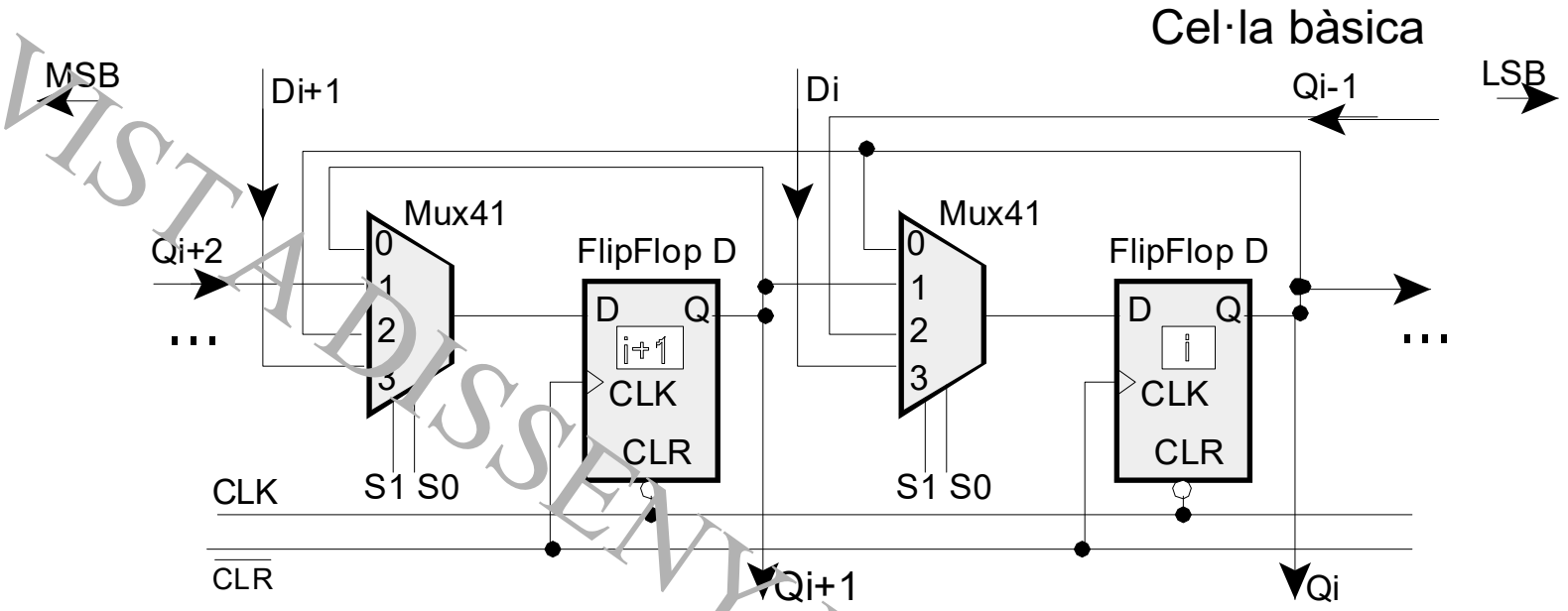
Power-On-Reset inicial

POWER-ON-RESET



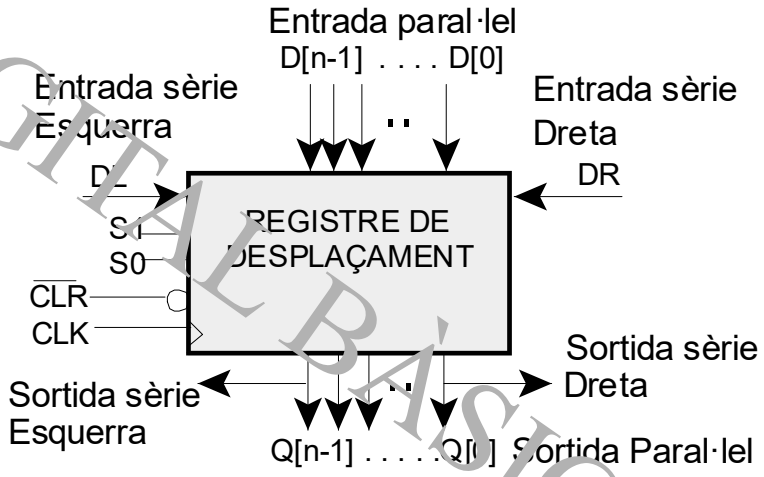
Molts Microprocessadors porten incorporat el Power-On-Reset

Registres de desplaçament



S0 i S1 són l'enable actuant al multiplexor

Selecció S1 S0	Operació	Senyals de Sortida Qn-1 Qn-2 ..... Q1 Q0
0 0	Mantenir HOLD	Qn-1 Qn-2 ..... Q1 Q0
0 1	Despl. Dreta SHR	DL Qn-1 ... Q2 Q1
1 0	Despl. Esquerra SHL	Qn-2 Qn-3 ..... Q0 DR
1 1	Càrrega LOAD	Dn-1 Dn-2 ..... D1 D0

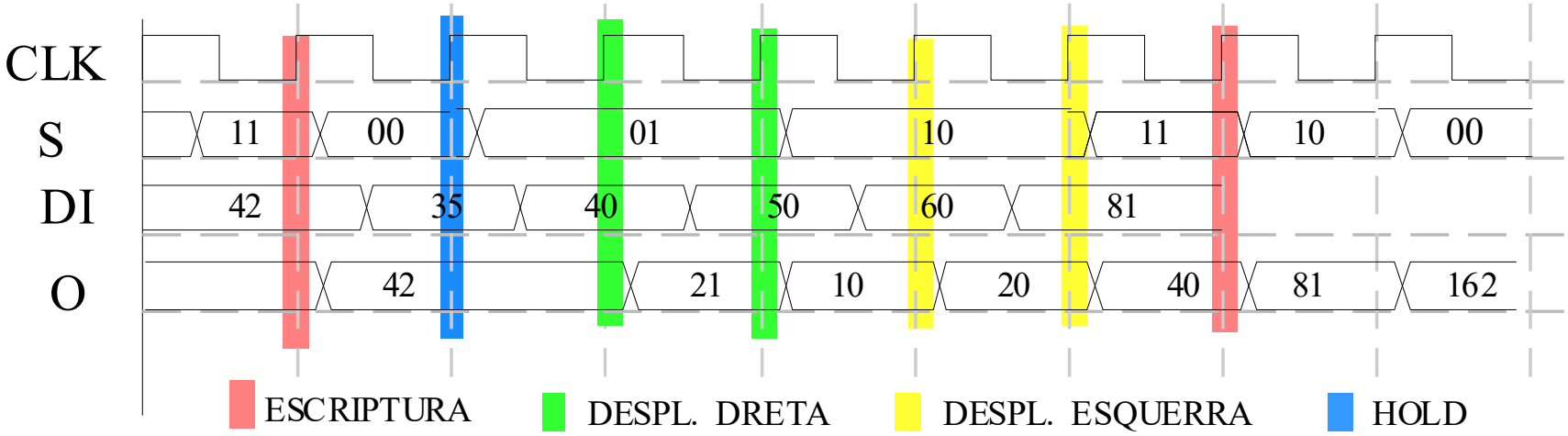


Aplicació directa dels registres de desplaçament en ASM

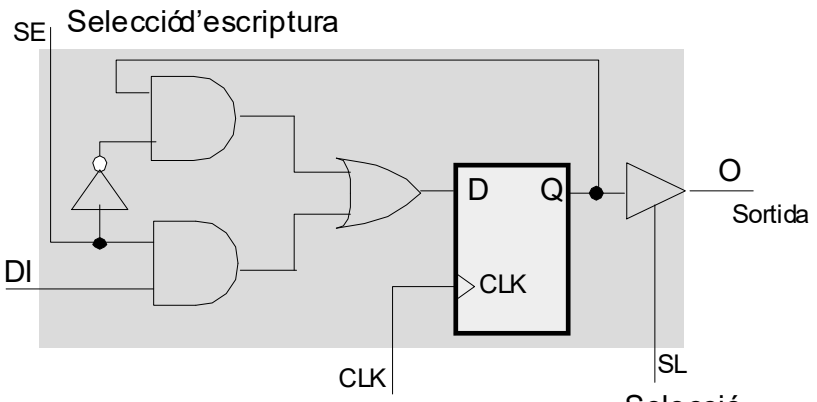
Instruccions de l'estil:

En RISC-V tenim per exemple
SLL Rd, Rs1, Rs2 => Desplaçament lògic de Rs1 [Rs2] bits
i es guarda en Rd.
SLLI Rd, Rs1, Imm => deplaçament lògic de Rs1 Imm bits

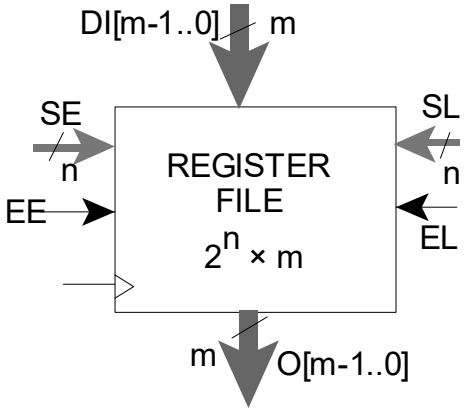
Funcionament del Registre de Desplaçament



Conjunt de Registres (*Register File*)

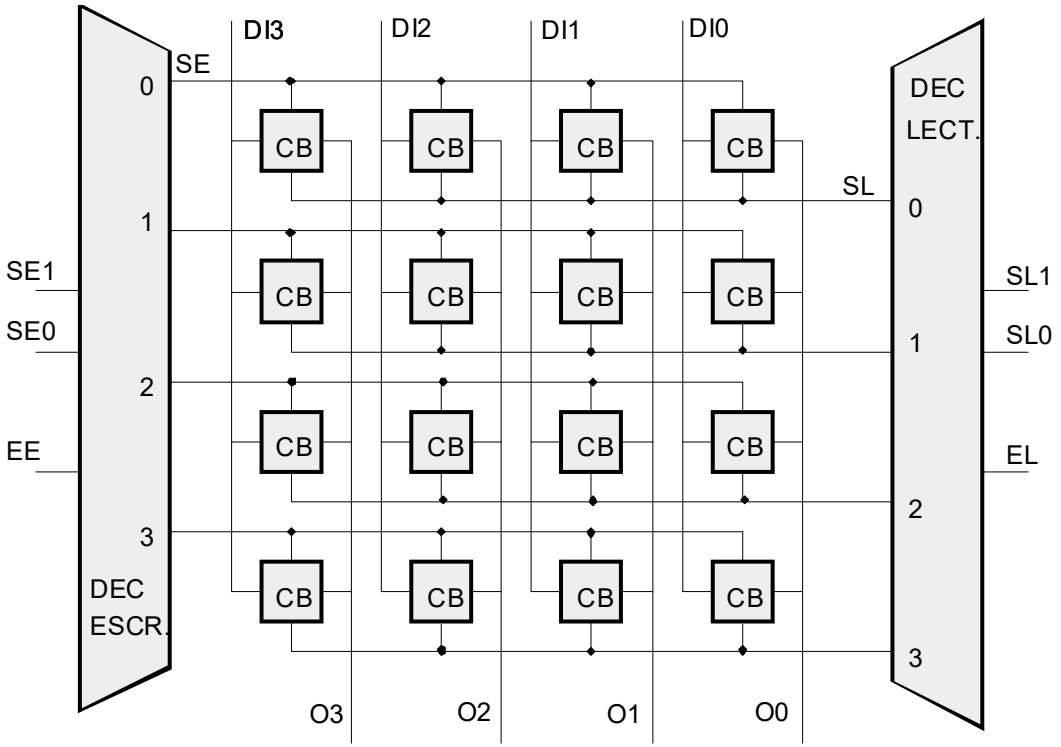


(a) Cel·la bàsica (CB)



(b) Register File

CR

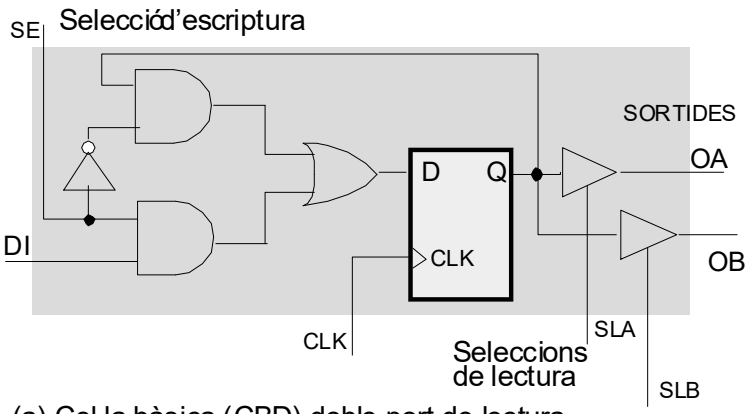


Conjunt de Registres de doble port

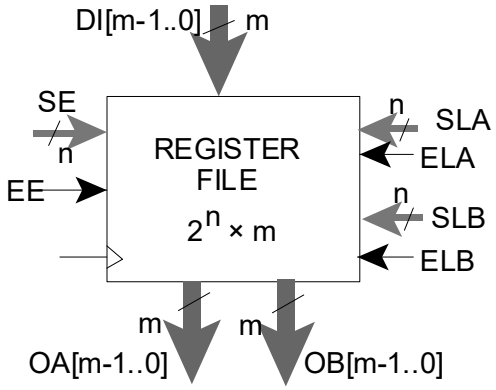
Dual-Port RF

Operacions

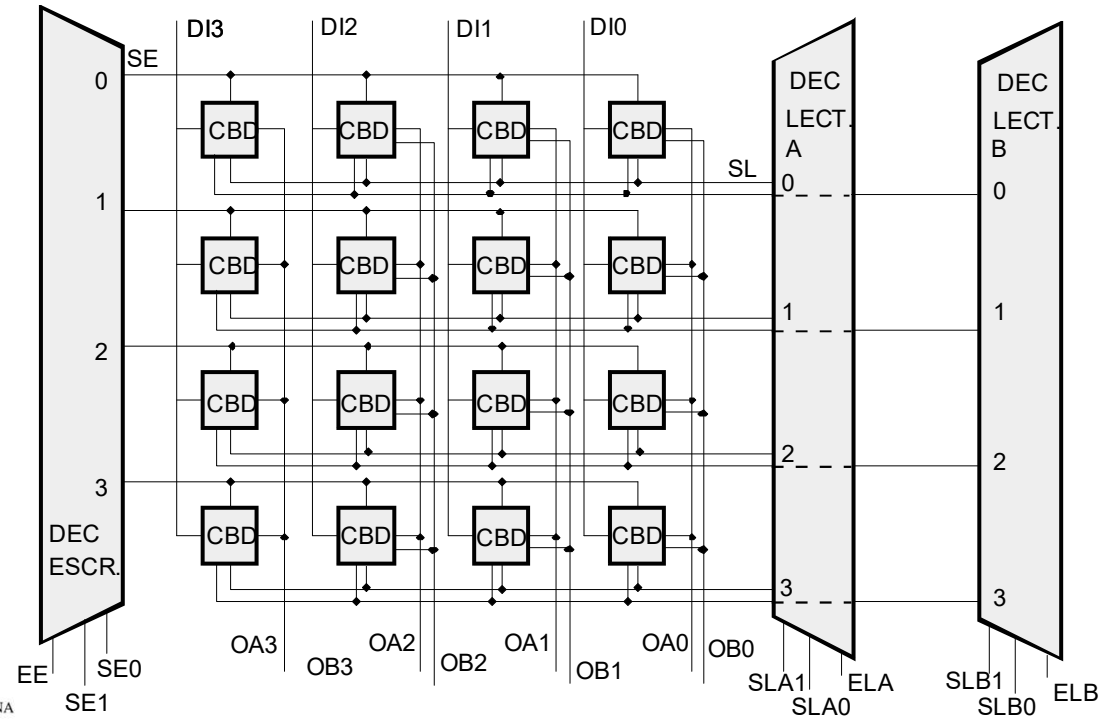
$R_i <= R_j * R_k$



(a) Cel·la bàsica (CBD) doble port de lectura



(b) *Register File* de doble port de lectura



Límits:

Àrea del CR creix aprox com el #ports^2

Temps d'accés creix lineal. amb el #ports

Registres de la CPU

•Tenim dos tipus de registres:

–Registres de propòsit general:

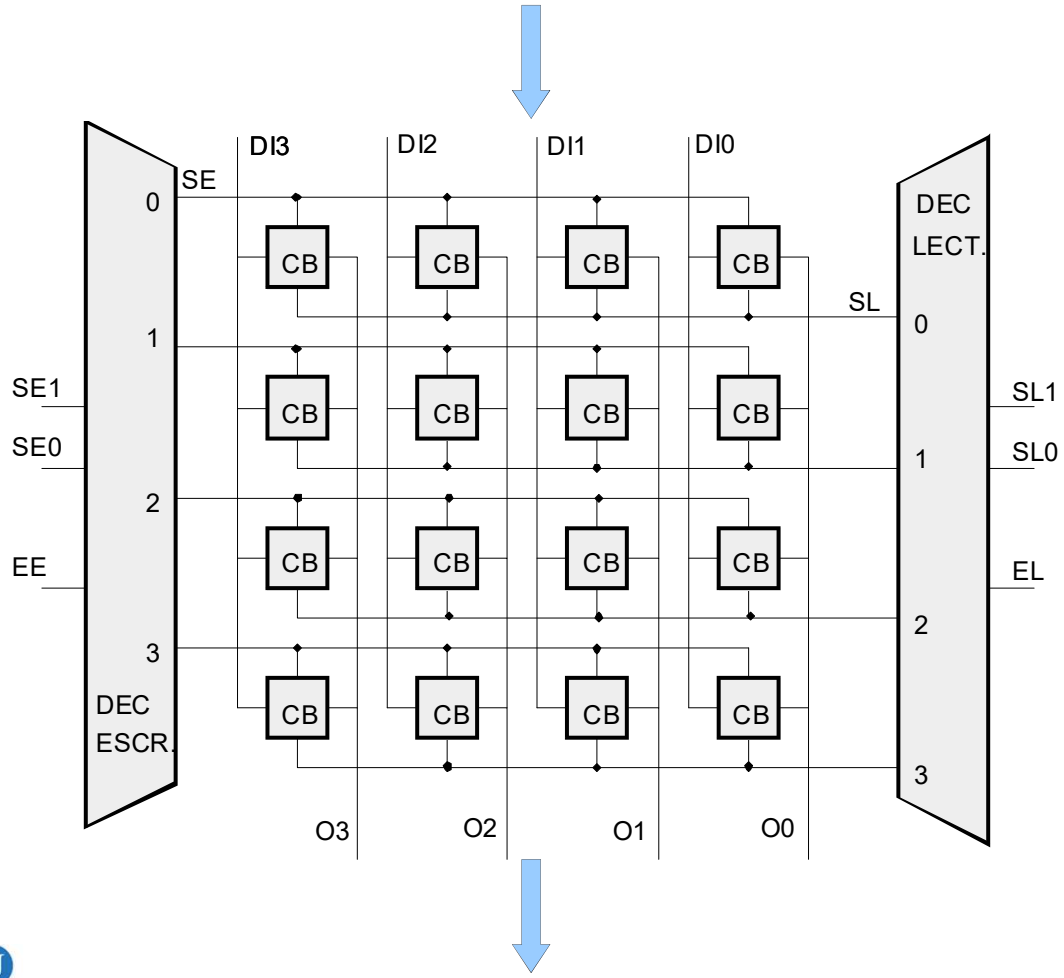
–GPR (General Purpose Registers). Poden guardar qualsevol tipus de dada o adreça de memòria. Fonamentals en l'arquitectura Von Neumann, poden ser accessibles al programador o no.

–Registres de propòsit específic:

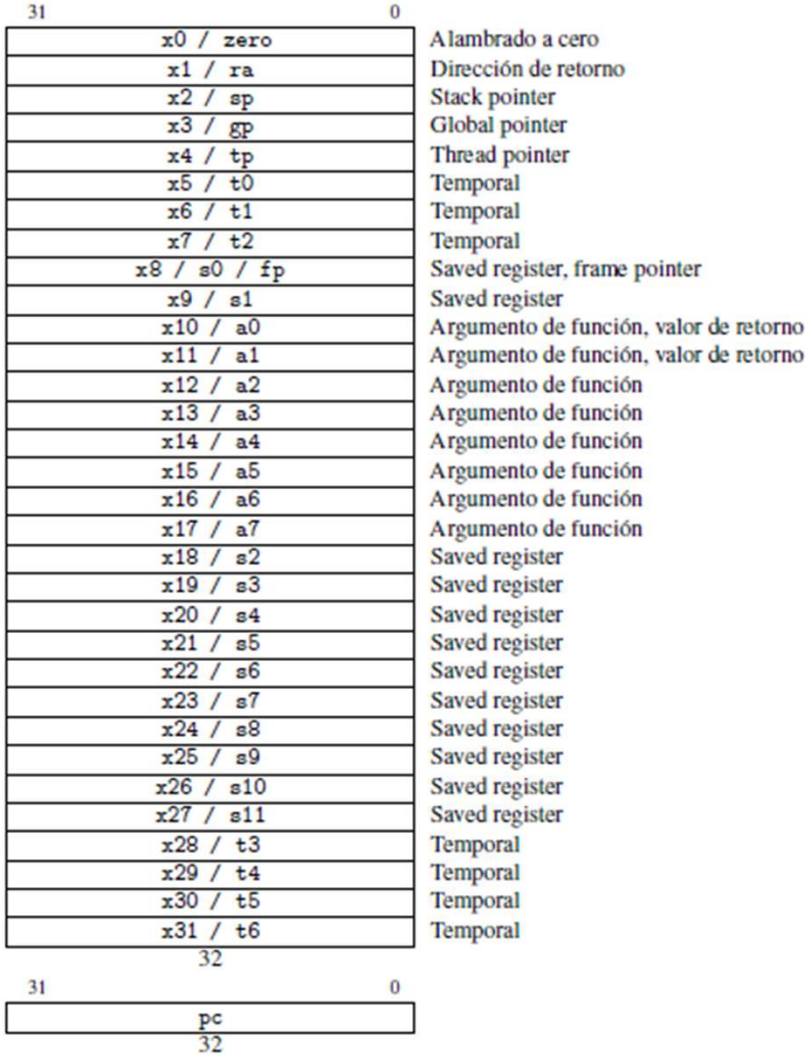
–Guarden informació específica de l'estat del sistema. No són accessibles al programador (a nivell d'escriptura, però si a nivell de lectura)

Registres de la CPU

•Registres de propòsit general (ja vist)



•Conjunt de registres RISC-V

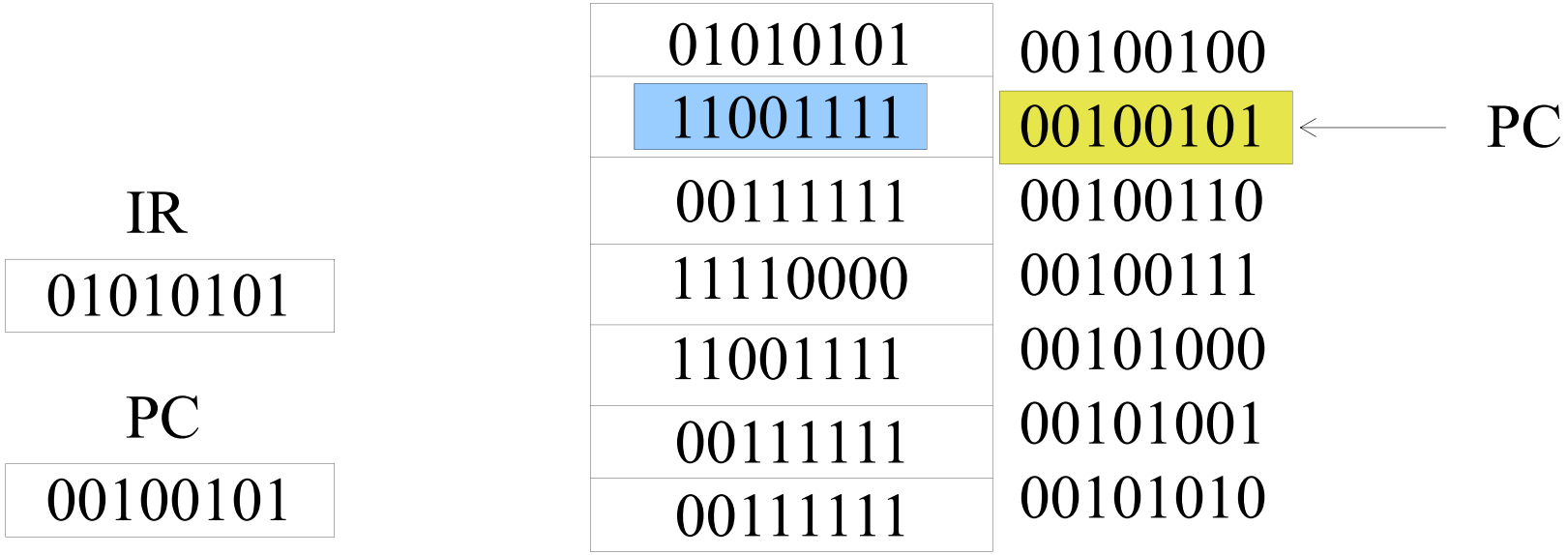


Registres de la CPU

•Registres específics

–PC (Program Counter). Ens indica la posició de memòria on es troba la seguent instrucció a executar. És un punter que ens indica l'adreça de memòria de la instrucció, NO EL CONTINGUT

–IR (instruccion Register). Registre de propòsit específic on es guarda el CONTINGUT de l'adreça de memòria on apuntava el PC



Registres de la CPU

•Registres especifics (II)

–ACC (Acumulador). És un registre on es guarda el resultat de qualsevol operació que realitza la ALU (Unitat Aritmètic-lògica)

–Pot ser explícit o implícit. En aquest darrer cas es fa servir com

–ACC un registre de propòsit general

–STAT (Status Reg.). És el registre on s'indiquen les incidències que es tenen al fer una determinada operación

Exemple:
El contingut del registre R1 és 010011
El contingut del registre R2 és 100110

Fem R1 + R2 → ACC <= R1+R2
ACC = 111001 = -7d
STAT => N =1, OV = 0, C = 0, Z = 0

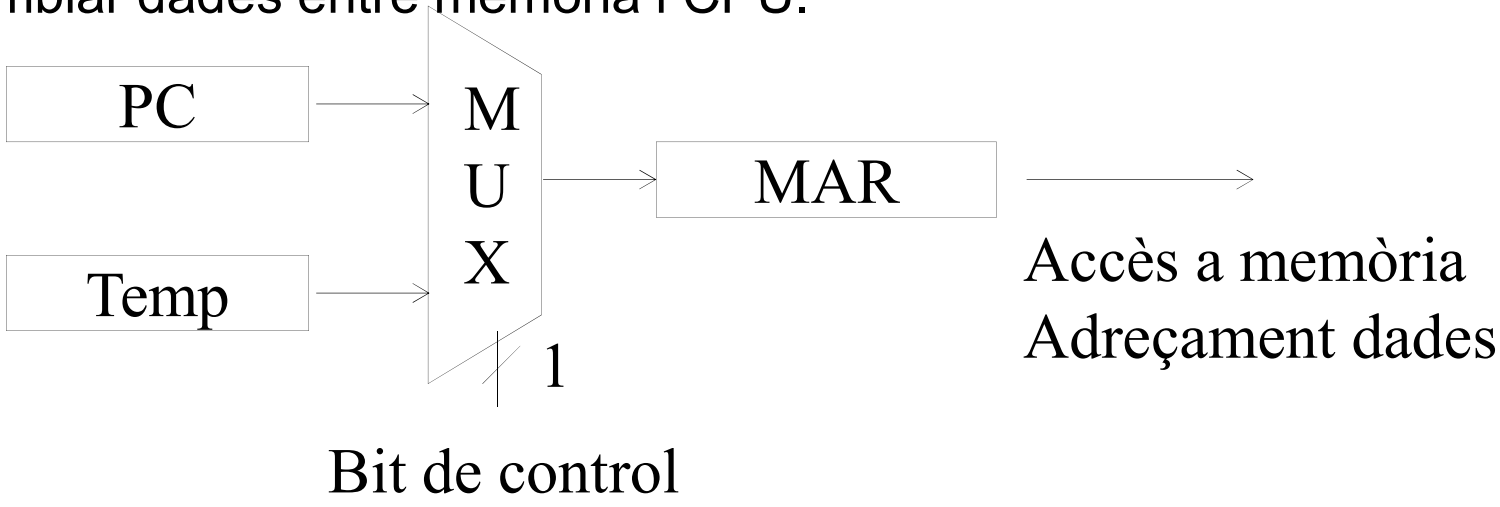
Registres de la CPU

•Registres específics

–SEQ (Sequenciador) S'encarrega de realitzar operacions de control (el veurem més endevant)

–MAR (Memory Address Register) Registre de adreces de memòria. Es fa servir de buffer per anar a una determinada posició de programa. Es el valor que es carrega al bus d'adreces

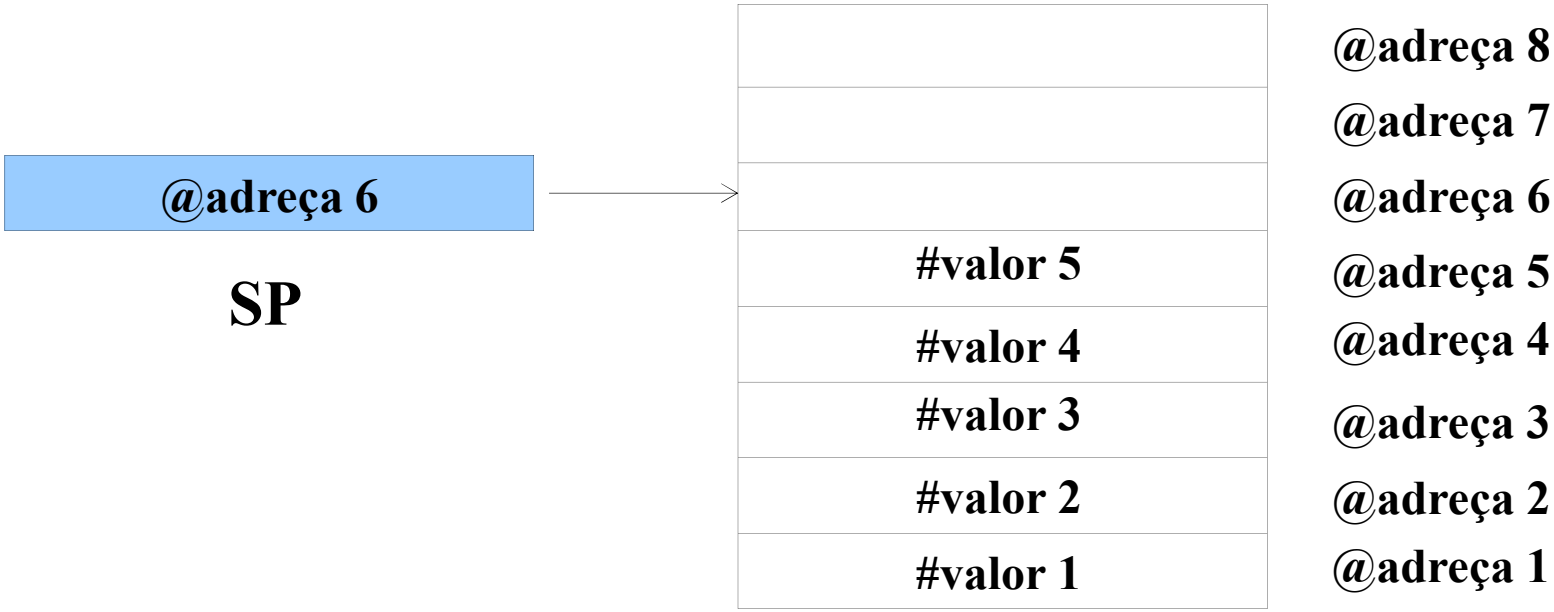
–MBR (Memory Buffer Register) Registre buffer de dades. Es un buffer de dades. Accedeix per tant al bus de dades i serveix per intercambiar dades entre memòria i CPU.



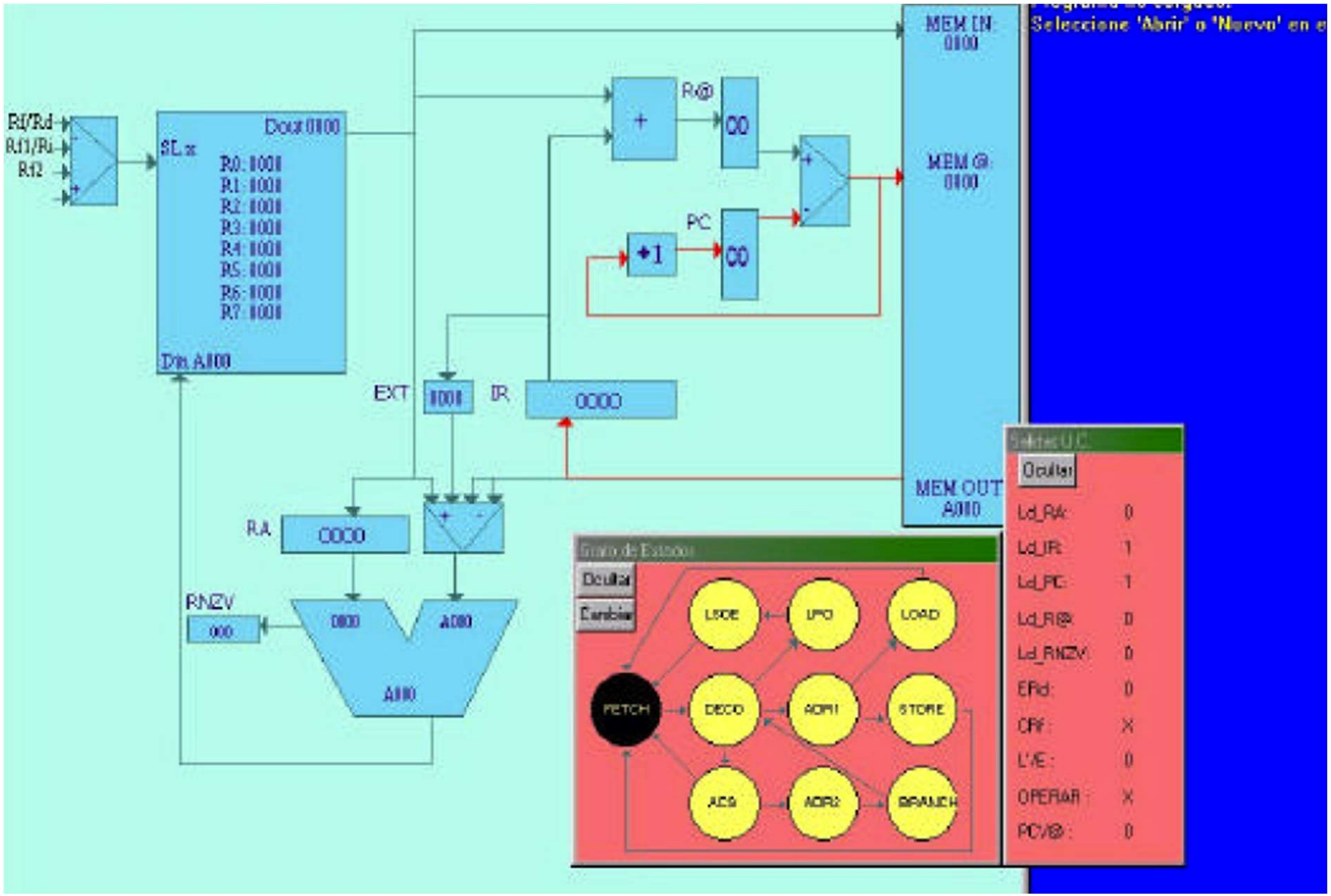
Registres de la CPU

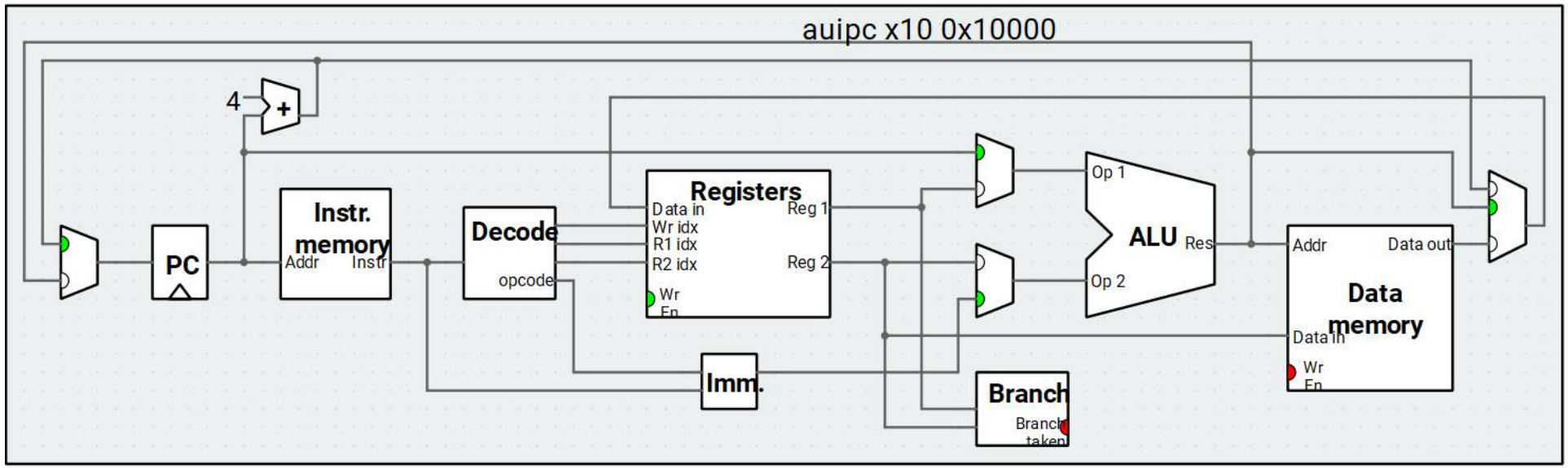
•Registres específics

–SP (Stack Pointer) Registre que ens indica la posició de l'últim valor guardat a la PILA.

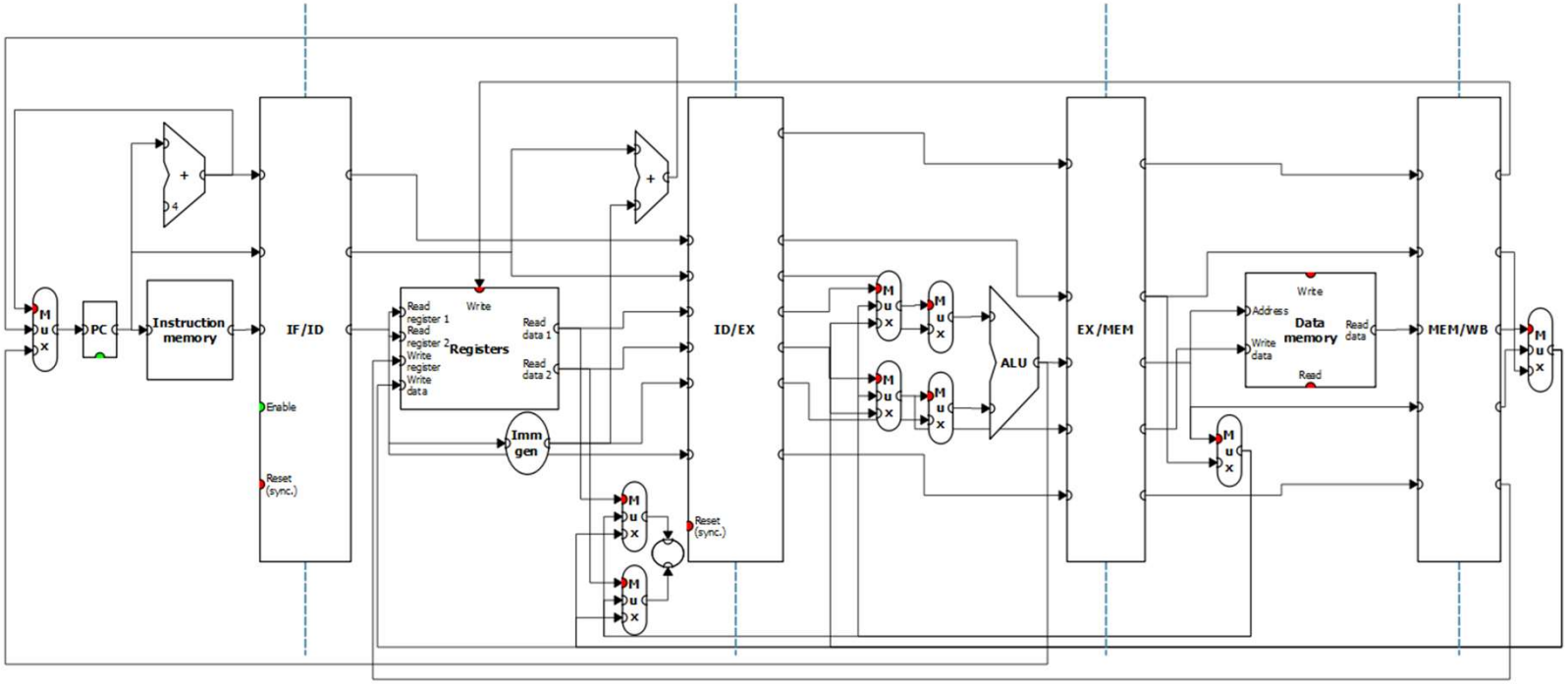


Registres de la CPU: Exemple SiMR

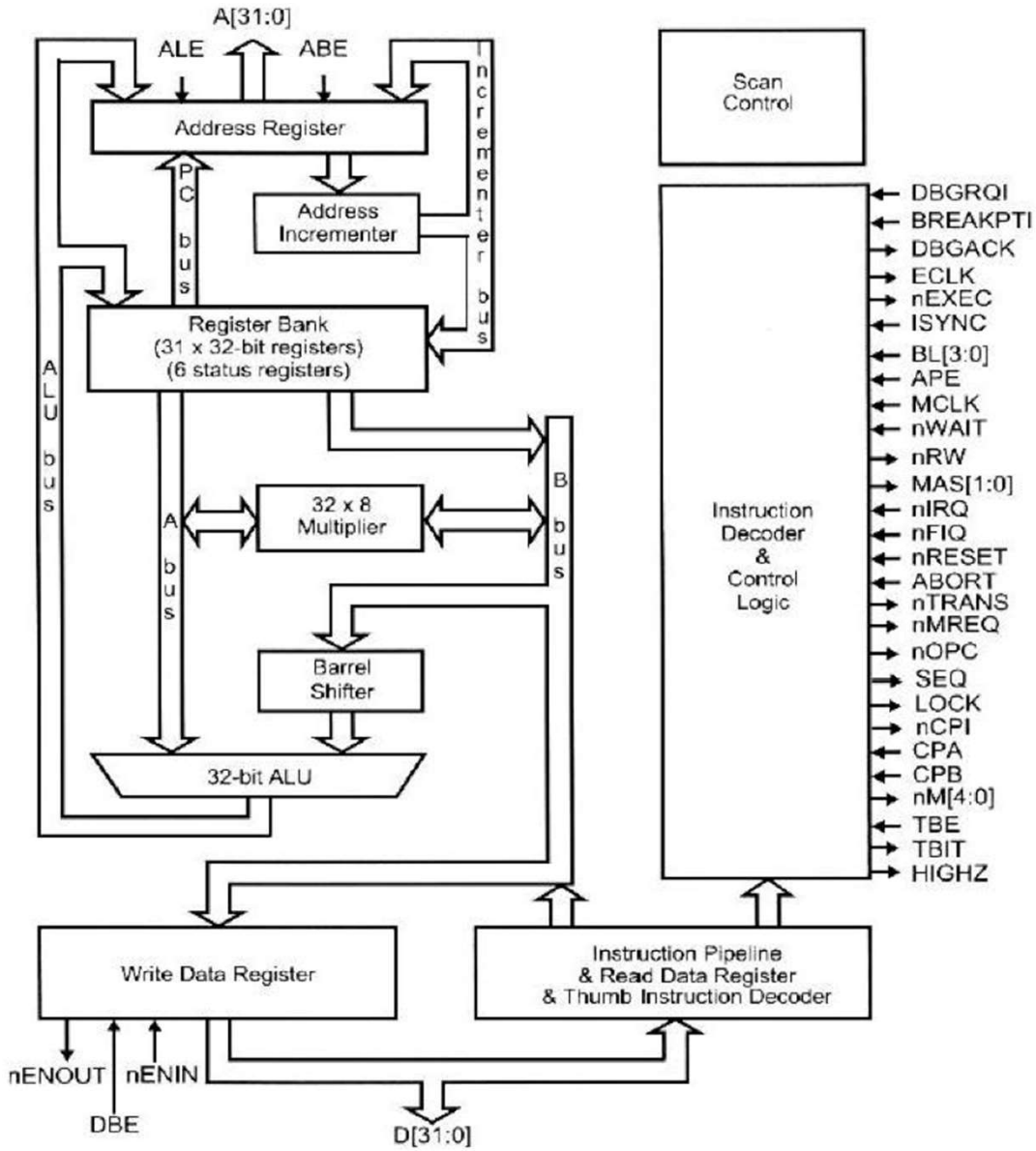



auipc x10 0x10000
4
+
PC
Instr.
memory
Addr
Instr
Decode
opcode
Registers
Data in
Wr idx
R1 idx
R2 idx
Wr
En
Reg 1
Reg 2
Imm.
Op 1
Op 2
ALU
Res
Branch
Branch
taken
Addr
Data out
Data
memory
Data In
Wr
En

Registres de la CPU: RISC-V



Registres de la CPU: Exemple Arquitectura ARM



Registres de la CPU: Arquitectura IA-32

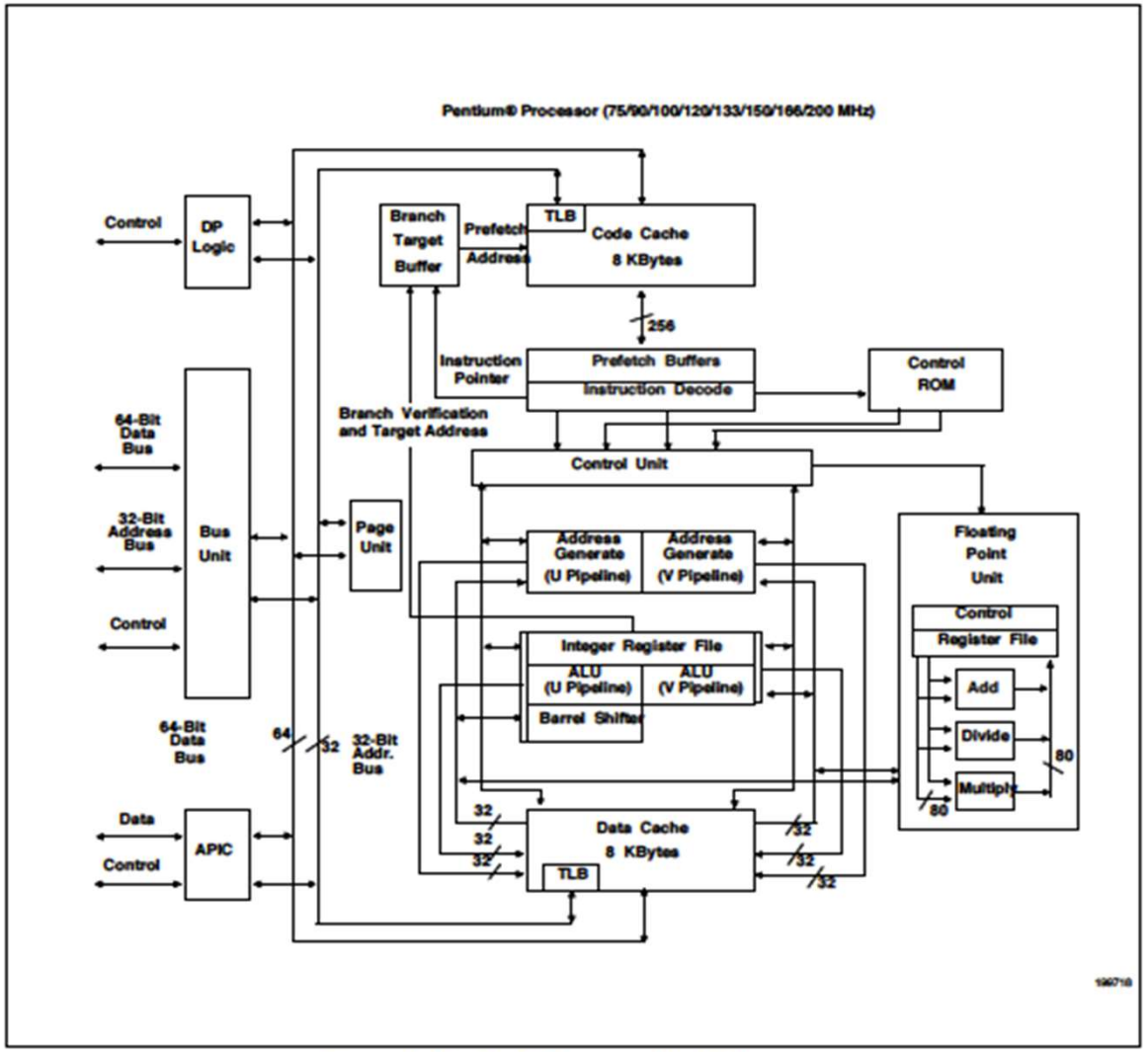


Figure 1. Pentium® Processor Block Diagram