



ESTRUCTURA i TECNOLOGIA de COMPUTADORS

Transparències del Tema 7
La Màquina Senzilla (MS1)
Modificacions i Microprogramació

X. Cufí, M. Fàbregas i J. Ferrer
Departament d'Electrònica,
Informàtica i Automàtica

ETC - Esquema del Tema 7:

La Màquina Senzilla MS1 – Modificacions i Microprogramació
Alternatives per la modificació d'un Computador

- Software
- Hardware
- Firmware
- Valoració

Modificació de la MS1

- Nou repertori d'instruccions
- Modes d'adreçament

Microprogramació

- Valoració de la **UC** de la MS1
- UC Microprogramada
- Implementació

Registres

Cicles

- Fetch
- Lectura
- Escriptura

Camins de Dades

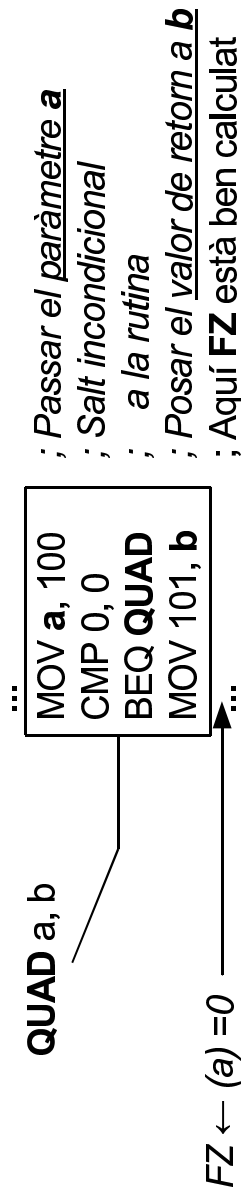
ETC – LA MÀQUINA SENZILLA – Modificacions Software (II)

AFEGIR QUAD F, D PER SOFTWARE (1^{er} Refinement)

- Convertir el programa en una rutina que es pugui parametritzar i reutilitzar.
- La única forma de fer-ho amb la Màquina Senzilla és usant adreces fixes.
 - Definim dues adreces fixes:
 - Una adreça (paràmetre) per passar (a) a QUAD.
 - Una adreça (retorn) pel resultat (b) de QUAD.
 - Per exemple, es pot reservar les adreces 100 i 101 respectivament.

QUAD: MOV 100, 101 ; (101) = (100)
 ADD 101, 101 ; (101) = (100) * 2
 ADD 101, 101 ; (101) = ((100) * 2) * 2

- Mecanisme de crida a QUAD:



ETC – LA MÀQUINA SENZILLA – Modificacions Software (III)

AFEGIR QUAD F, D PER SOFTWARE (2^{on} Refinement)

PROBLEMES

- A la versió anterior, **NO** es pot retornar de la crida.

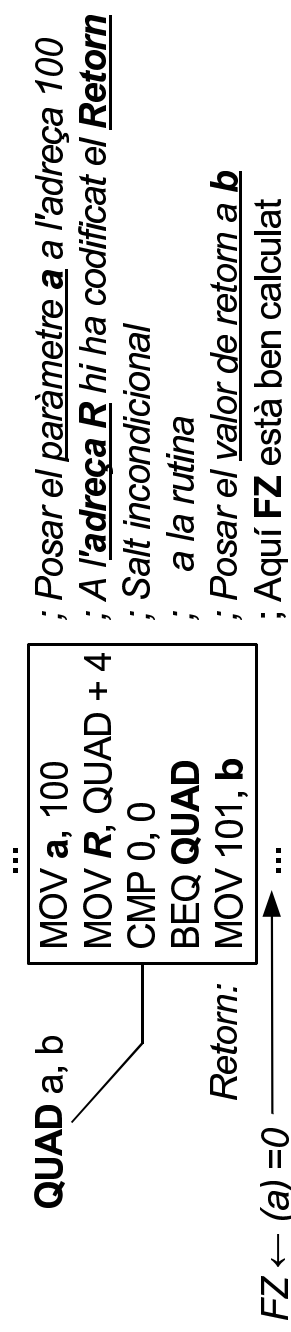
SOLUCIÓ

- Definir el retorn abans de fer la crida.

QUAD : MOV 100, 101 ; (101) = (100)
 ADD 101, 101 ; (101) = (100) * 2
 ADD 101, 101 ; (101) = ((100) * 2) * 2
 CMP 0, 0 ; Salt a l'adreça

Mecanisme
de *Return* → QUAD + 4 : **“BEQ Return”** ; de *return*

- Mecanisme de crida a QUAD:



ETC – LA MÀQUINA SENZILLA – Modificacions Software (IV)

AFEGIR QUAD F, D PER SOFTWARE (Exemple: $b = a * 16$)

000:	MOV a, 100	; Paràmetre = a
001:	MOV Retb, 054	; Retorn
002:	CMP 0, 0	
003:	BEQ QUAD	; (101) = a * 4
004:	MOV 101, 100	; Paràmetre = a * 4
005:	MOV Retb, 054	; Retorn
006:	CMP 0, 0	
007:	BEQ QUAD	; (101) = a * 16
008:	MOV 101, b	; b = a * 16
009:	BEQ NoZero	; Consultar si
...		; a * 16 és 0
021:	NoZero:	
...		
050:	QUAD: MOV 100, 101	; (101) = (100)
051:	ADD 101, 101	; (101) = (100)*2
052:	ADD 101, 101	; (101) = ((100)*2)*2
053:	CMP 0, 0	; Salt a l'adreça
054:	"BEQ XXX"	; de retorn
...		

Paràmetre = 100: ?
Resultat = 101: ?
a = 102: 1673
b = 103: ? ←
Retb = 104: BEQ 004
Retb = 105: BEQ 008
106:
107:
108:
119:

***BEQ 004** = 11 00000000 0000100_b
= 49156_d = C004_h
***BEQ 008** = 11 00000000 0001000_b
= 49160_d = C008_h

ETC – LA MÀQUINA SENZILLA – Modificacions Software (V)

AFEGIR QUAD F, D PER SOFTWARE (V) - VALORACIÓ

S'HA POGUT AFEGIR LA INSTRUCCIÓ QUAD F, D

- Es troba una sola vegada a memòria.
- Es pot cridar tantes vegades com es vulgui.

AVANTATGES

- No hi ha hagut **cap canvi del Hardware** de la **MS1** (ni a la **UP** ni a la **UC**).

INCONVENIENTS

- Per implementar la **rutina** cal:
 - 2 adreces fixes (**paràmetre** i el retorn del **resultat**)
 - **Cost:** 3*4 (ADD/CMP) + 1*3 (MOV) + 1*2 (BEQ) = 17 Cicles
- Per cada crida a la rutina:
 - 1 adreça per guardar la constant amb el **"BEQ Return"**
 - **Cost:** 1*4 (CMP) + 3*3 (MOV) + 1*2 (BEQ) = 15 Cicles

Cost total per poder multiplicar per 4 per software: **32 Cicles**

ETC – LA MÀQUINA SENZILLA – Modificacions Hardware (I)

AFEGIR QUAD F, D PER HARDWARE (I)

OBJECTIU

- En aquest cas, es tracta de **modificar la UP i la UC** per realitzar la nova operació:
- **QUAD F, D** és una **operació aritmètica** (un producte).
- A la **MS1**, les **operacions aritmètiques** es realitzen a l'**ALU**.

MODIFICACIONS A LA UP

- Es pot afegir una nova operació modificant l'**ALU**:
 - Hi ha una entrada lliure al multiplexor de selecció d'operació.
 - Caldrà algun bloc per realitzar l'operació (multiplicar per 4).

MODIFICACIONS A LA UC

- Caldrà implementar la part de l'autòmata d'estats de la **UC** per *Carregar, Descodificar, Cercar l'operand i Executar* la nova operació.

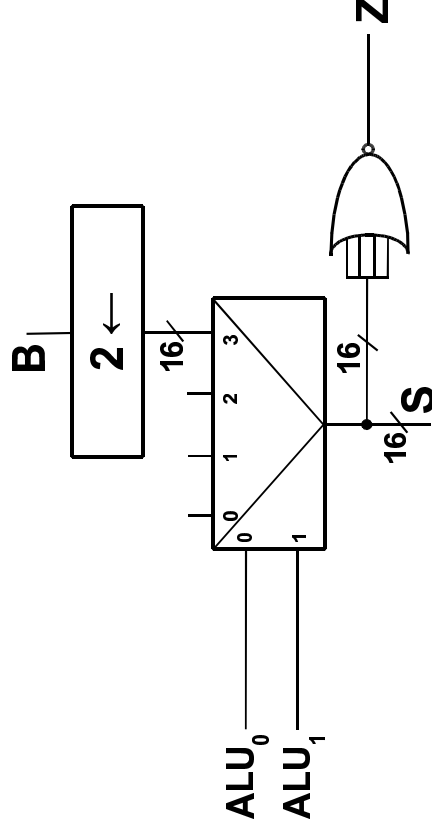
ETC – LA MÀQUINA SENZILLA – Modificacions Hardware (II)

AFEGIR QUAD F, D PER HARDWARE (Modificacions a la UP)

NOU DISSENY DE L'ALU

ALU ₁	ALU ₀	OPERACIÓ
0	0	A + B
0	1	A xor B (bit a bit)
1	0	Transferència de B
1	1	4 * B

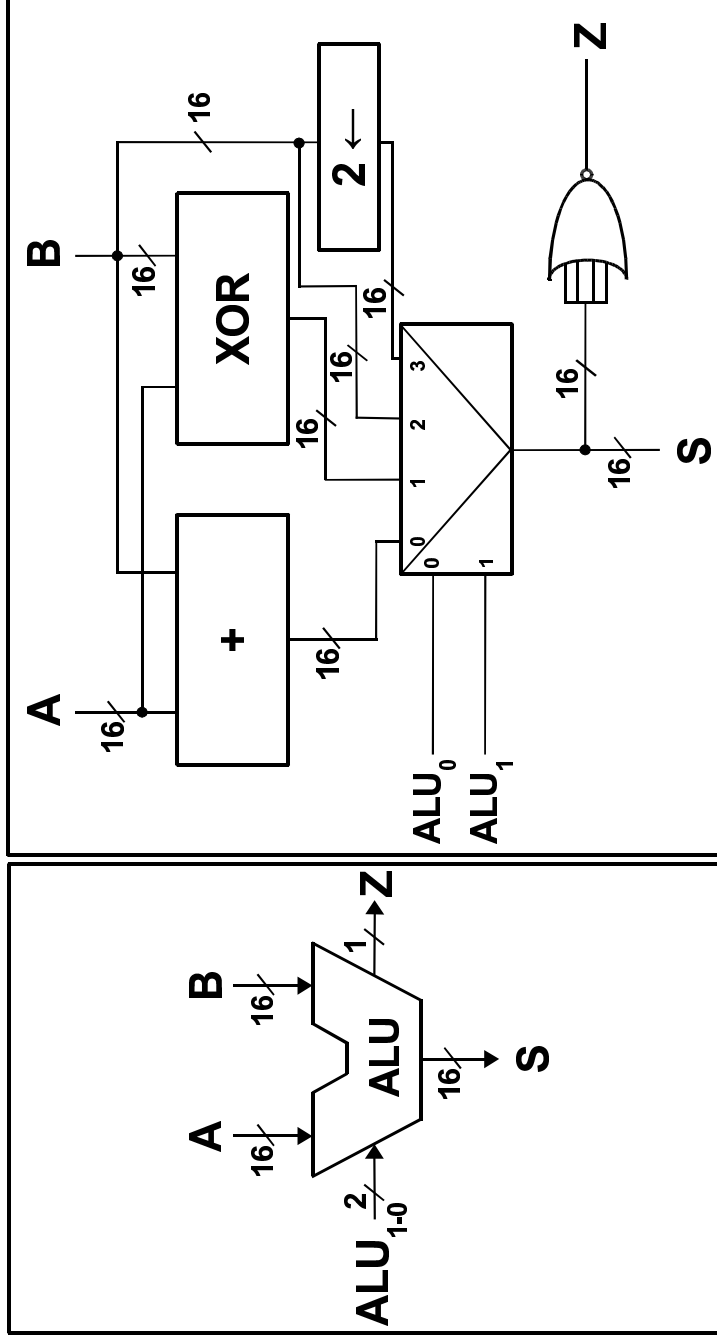
MODIFICACIÓ DE LA IMPLEMENTACIÓ DE L'ALU



ETC – LA MÀQUINA SENZILLA – Modificacions Hardware (III)

AFEGIR QUAD F, D PER HARDWARE (Modificacions a la UP)

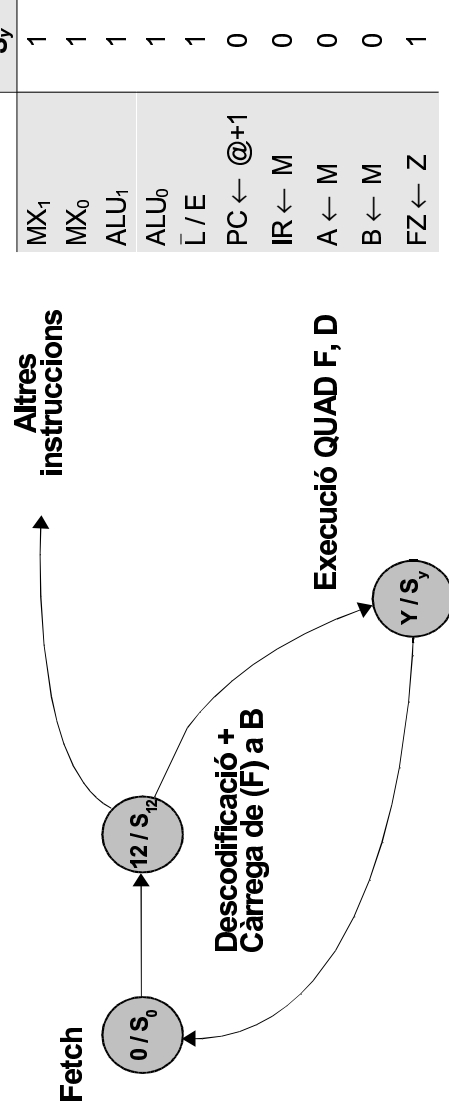
NOVA IMPLEMENTACIÓ DE L'ALU



ETC – LA MÀQUINA SENZILLA – Modificacions Hardware (IV)

AFEGIR QUAD F, D PER HARDWARE (Modificacions a la UC)

- Al graf d'estats de la MS1 simplificat ja existeix:
 - Fetch
 - Descodificació + Càrrega de l'operand Font (al registre B)
- Cal afegir:
 - Descodificació de la nova instrucció.
 - Execució de QUAD F, D



ETC – LA MÀQUINA SENZILLA – Modificacions Hardware (V)

AFEGIR QUAD F, D PER HARDWARE (VALORACIÓ)

S'HA POGUT AFEGIR LA INSTRUCCIÓ QUAD F, D

- S'han realitzat modificacions a la UP i a la UC.
- Encara faltaria modificar el format d'instrucció (codificar la instrucció QUAD F, D).

AVANTATGES

- La instrucció és molt eficient:
 - **Costa** només **3 cicles**.
 - Realitza **3 accessos a memòria** (*Fetch, Operand, Resultat*).

INCONVENIENTS

- És més car ja que ha calgut **modificar el Hardware** de la **MS1** original.

ETC – LA MÀQUINA SENZILLA – Modificacions Firmware (I)

AFEGIR QUAD F, D PER FIRMWARE (I)

OBJECTIU

- Solució intermitja a les dues anteriors (**Software i Hardware**).
- Aquest cop, s'intentarà afegir la nova instrucció només modificant la **Unitat de Control** i utilitzant els recursos que proporciona la **UP**.

FASES D'EXECUCIÓ DE LA NOVA INSTRUCCIÓ

QUAD F, D

Definició dels estats

Estat S_0 (Fase 1):	$IR \leftarrow (PC)$	$PC = PC + 1$
Estat S_a (Fase 2/3):	Avaluació de CO_1 i CO_0	$A \leftarrow (F)$ $B \leftarrow (F)$
Estat S_b (Fase 4):	$(D) \leftarrow A + B$	
Estat S_c (Fase 3):	$A \leftarrow (D)$	$B \leftarrow (D)$
Estat S_d (Fase 4):	$(D) \leftarrow A + B$	$FZ \leftarrow Z$

ETC – LA MÀQUINA SENZILLA – Modificacions Firmware (II)

AFEGIR QUAD F, D PER FIRMWARE (VALORACIÓ)

S'HA POGUT AFEGIR LA INSTRUCCIÓ QUAD F, D

- S'han realitzat modificacions només a la UC.
- Encara faltaria modificar el format d'instrucció (codificar la instrucció QUAD F, D).

AVANTATGES

- La instrucció és **mitjanament** eficient:
 - **Costa 5 cicles** (les 5 fases per dur-la a terme).
 - Realitza **5 accessos a memòria** (*Fetch, Operands, Resultat*).
 - Té un cost intermig.

INCONVENIENTS

- No és la solució més eficient.

ETC – LA MÀQUINA SENZILLA – Modificacions - Valoració

VALORACIÓ D' AFEGIR QUAD F, D SEGONS L'ESTRATÈGIA

SOFTWARE (1^{era} Aproximació)

Modificacions a la MS1:

- Cost:
- No s'ha realitzat cap modificació ni a la UP ni a la UC.
 - Necessita **11 cicles** i realitza **11 accessos a memòria**.

Valoració final: **Poc eficient i Molt barata.**

HARDWARE

Modificacions a la MS1:

- Cost:
- S'ha modificat la UC, la UP i el format d'instrucció.
 - Necessita **3 cicles** i realitza **3 accessos a memòria**.

Valoració final: **Molt eficient i Molt cara.**

FIRMWARE

Modificacions a la MS1:

- Cost:
- S'ha modificat la UC i el format d'instrucció.
 - Necessita **5 cicles** i realitza **5 accessos a memòria**.

Valoració final: **Mitjanament eficient i relativament cara.**

Quina és la BONA?

ETC – LA MÀQUINA SENZILLA – Modificació de la MS1

OBJECTIU

AMPLIAR EL REPERTORI D'INSTRUCCIONS DE LA MS1

- Realitzar totes les modificacions necessàries perquè la **MS1** pugui executar les noves instruccions.
- Reconèixer alguns **modes d'adreçament**:
 - Directe
 - Immediat
 - Implícit
- Modificar el format d'instrucció per poder codificar les noves instruccions.

ETC – LA MÀQUINA SENZILLA – Modificació de la MS1

AMPLIACIÓ DEL REPERTORI D'INSTRUCCIONS (I)

Es vol ampliar el repertori d'instruccions de la **MS1** amb **3 instruccions** addicionals:

NOM	MNEMÒNIC	OPERACIÓ	COMENTARIS
5. CLEAR	CLEAR D	(D) ← 0	Posa un 0 a la posició de memòria D. NO modifica el Flag de Zero.
6. MOURE CONSTANT	MOVD K, D	(D) ← K [*] FZ ← K [*] = 0	Carrega la constant K directament a l'adreça de memòria D. Actualitza FZ .
7. SUMAR CONSTANT	ACUM K, D	(D) ← (D) + K [*] FZ ← ((D) + K [*]) = 0	Acumula la constant K a l'adreça D. A més a més, actualitza el Flag de Zero segons el resultat de l'acumulació.

***NOTA:**

La constant **K** és una paraula de 16 bits.

ETC – LA MÀQUINA SENZILLA – Modificació de la MS1

AMPLIACIÓ DEL REPERTORI D'INSTRUCCIONS (II)

PROBLEMES

- Actualment només es poden codificar 4 instruccions diferents (**ADD**, **CMP**, **MOV** i **BEQ**) ja que només hi ha 2 bits (CO_1 i CO_0) per definir el codi d'operació.
- Les noves instruccions porten associada una constant **K** de 16 bits.

SOLUCIÓ

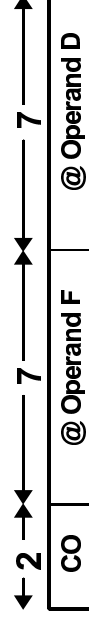
- **Redefinir el format d'instrucció** per poder codificar les noves operacions.
- Definir i utilitzar nous modes d'adreçament per poder utilitzar **K**.

ETC – LA MÀQUINA SENZILLA – Modificació de la MS1

NOU FORMAT D'INSTRUCCIÓ

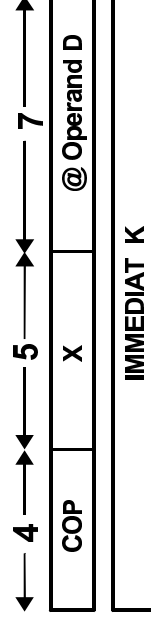
FORMAT D'INSTRUCCIÓ ACTUAL

*Instruccions de
2 operands*



NOU FORMAT D'INSTRUCCIÓ

*Instruccions d'1
i/o immediat*



Codificació Extesa

La combinació
11 diferència els dos
tipus d'instruccions

CO_1	CO_0	E_1	E_0	OPERACIÓ	OPERANDS
0	0			ADD	F, D
0	1			CMP	F, D
1	0			MOV	F, D
1	1	0	0	BEQ	D
1	1	0	1	CLEAR	D
1	1	1	0	MOVD	K, D
1	1	1	1	ACUM	K, D

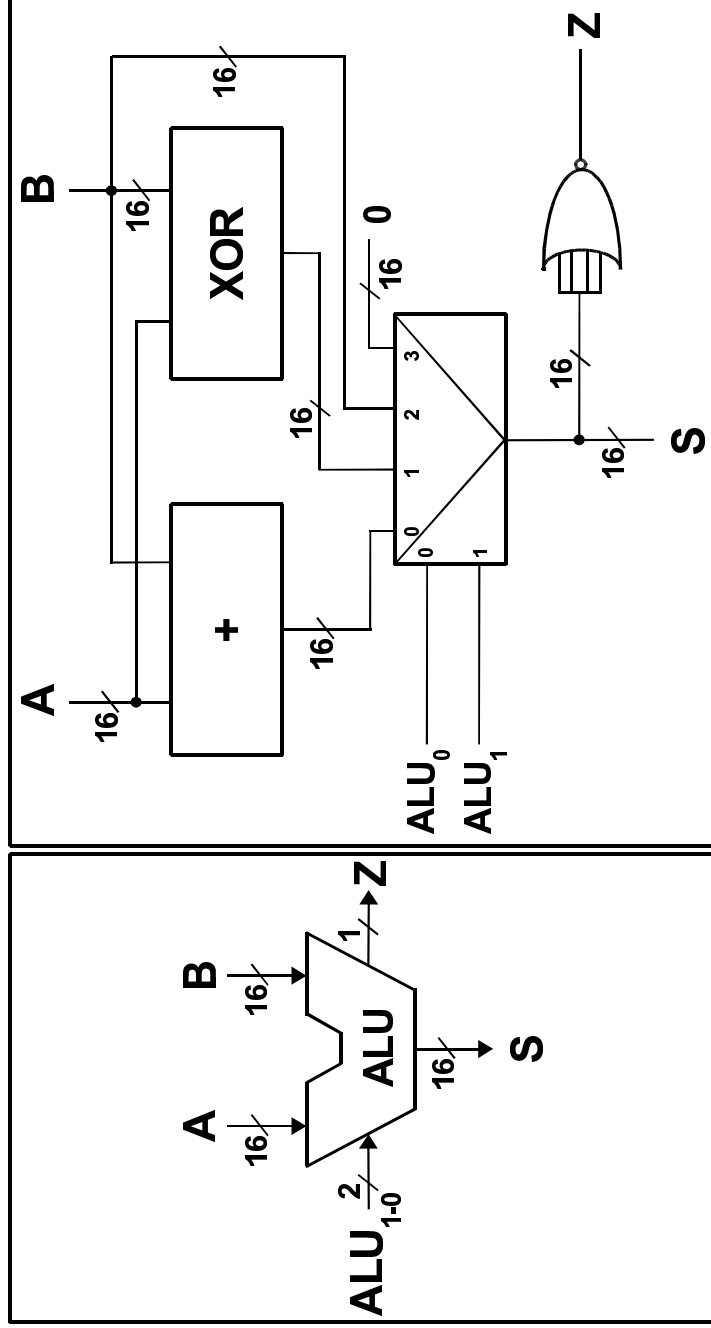
Codificació de les noves instruccions

- 4 Bits pel codi d'instrucció
- 5 Bits no utilitzats
- 7 Bits per l'adreça de l'operand **D**
- 16 Bits per **K** (**MOVD** i **ACUM**)

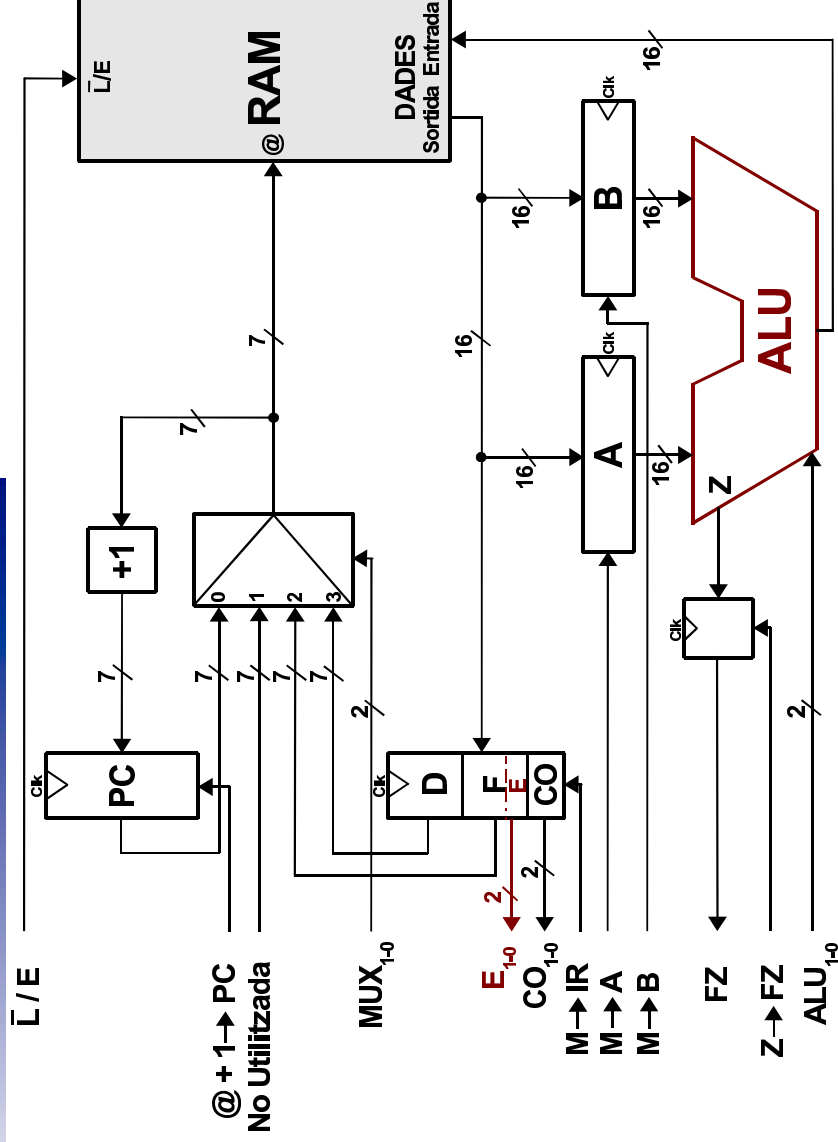
ETC – LA MÀQUINA SENZILLA – Modificació de la MS1

MODIFICACIONS A LA UNITAT DE PROCÉS (II)

NOVA IMPLEMENTACIÓ DE L'ALU



ETC – LA MÀQUINA SENZILLA – Modificació de la MS1

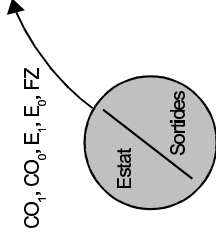


ETC – LA MÀQUINA SENZILLA – Modificació de la MS1

MODIFICACIONS A LA UNITAT DE CONTROL (I)

QUÈ HA CANVIAT A LA UC?

- Amb el nou format d'instrucció que s'ha incorporat, calen 2 bits més per poder descodificar totes les instruccions.
- Canvi en el vector de senyals entre **UP** i **UC**.



- Cal afegir estats per **acabar de descodificar, carregar els operands i executar** les noves instruccions (el *fetch* i part de la *descodificació* ja la fa l'autòmat actual).

ETC – LA MÀQUINA SENZILLA – Modificació de la MS1

MODIFICACIONS A LA UNITAT DE CONTROL (II)

FASES D'EXECUCIÓ DE LES NOVES INSTRUCCIONS (I)

CLEAR

Definició dels estats

Estat **S₀** (Fase 1):

Estat **S₁** (Fase 2):

Estat **S₁₃** (Fase 4):

IR ← (PC)

Avaluació de CO₁, CO₀, E₁ i E₀
(D) ← 0

PC = PC + 1

MOVD

Definició dels estats

Estat **S₀** (Fase 1):

Estat **S₁** (Fase 2):

Estat **S₁₄** (Fase 3):

Estat **S₁₅** (Fase 4):

IR ← (PC)

Avaluació de CO₁, CO₀, E₁ i E₀

B ← (PC)
PC = PC + 1

(D) ← B
FZ ← Z

PC = PC + 1

ETC – LA MÀQUINA SENZILLA – Modificació de la MS1

MODIFICACIONS A LA UNITAT DE CONTROL (III)

FASES D'EXECUCIÓ DE LES NOVES INSTRUCCIONS (II)

ACUM K, D

Definició dels estats

Estat S_0 (Fase 1):

Estat S_1 (Fase 2):

Estat S_{14} (Fase 3):

Estat S_{16} (Fase 3):

Estat S_{17} (Fase 4):

IR ← (PC)

Avaluació de CO_1, CO_0, E_1, E_0

B ← (PC)

A ← (D)

(D) ← A + B

PC = PC + 1

PC = PC_0, E_1, E_0

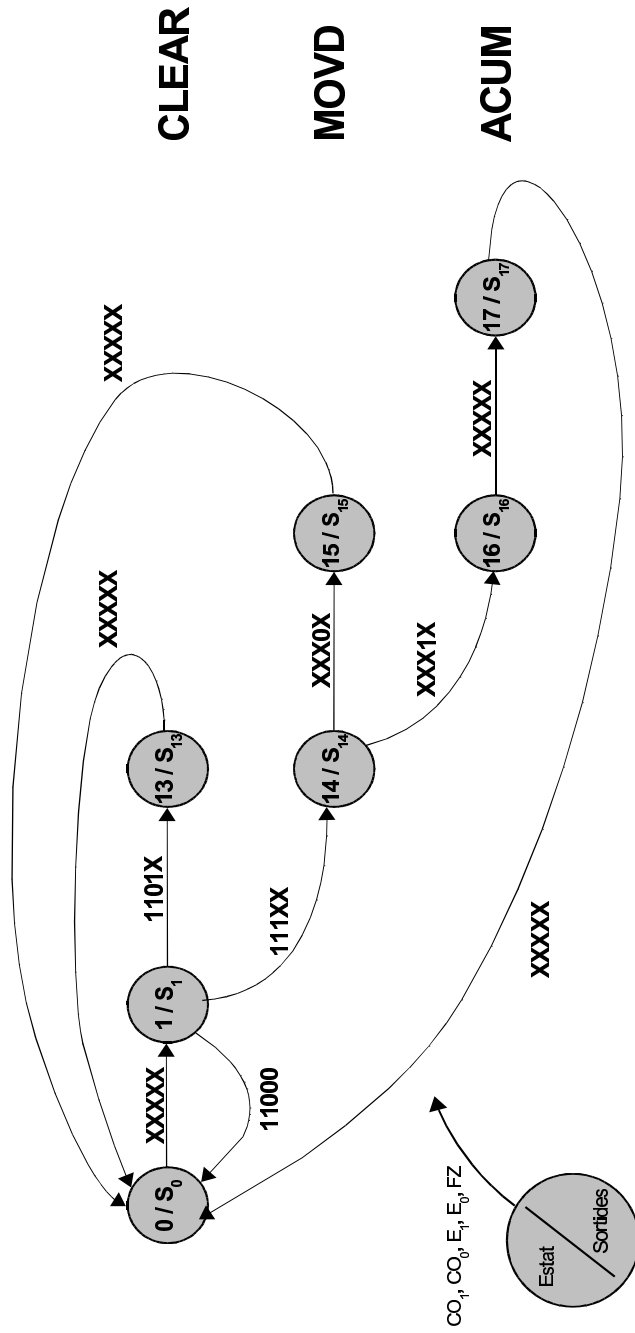
PC = PC + 1

FZ ← Z

ETC – LA MÀQUINA SENZILLA – Modificació de la MS1

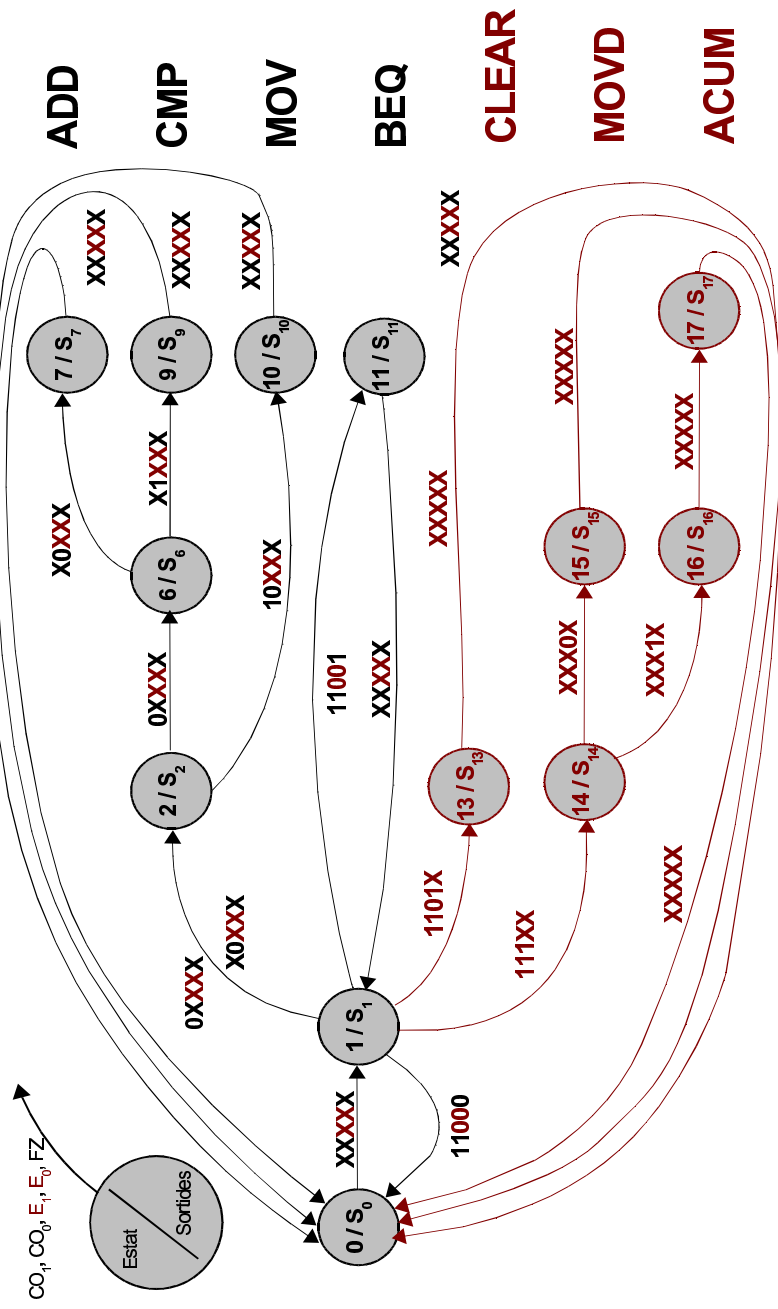
MODIFICACIONS A LA UNITAT DE CONTROL (IV)

NOU AUTÒMAT QUE CAL AFEGIR A L'ACTUAL



ETC – LA MÀQUINA SENZILLA – Modificació de la MS1

MODIFICACIONS A LA UNITAT DE CONTROL (V)



ETC – LA MÀQUINA SENZILLA – Modificació de la MS1

MODIFICACIONS A LA UNITAT DE CONTROL (VI)

VECTOR BINARI DE SORTIDES DE CADA ESTAT DEL GRAF

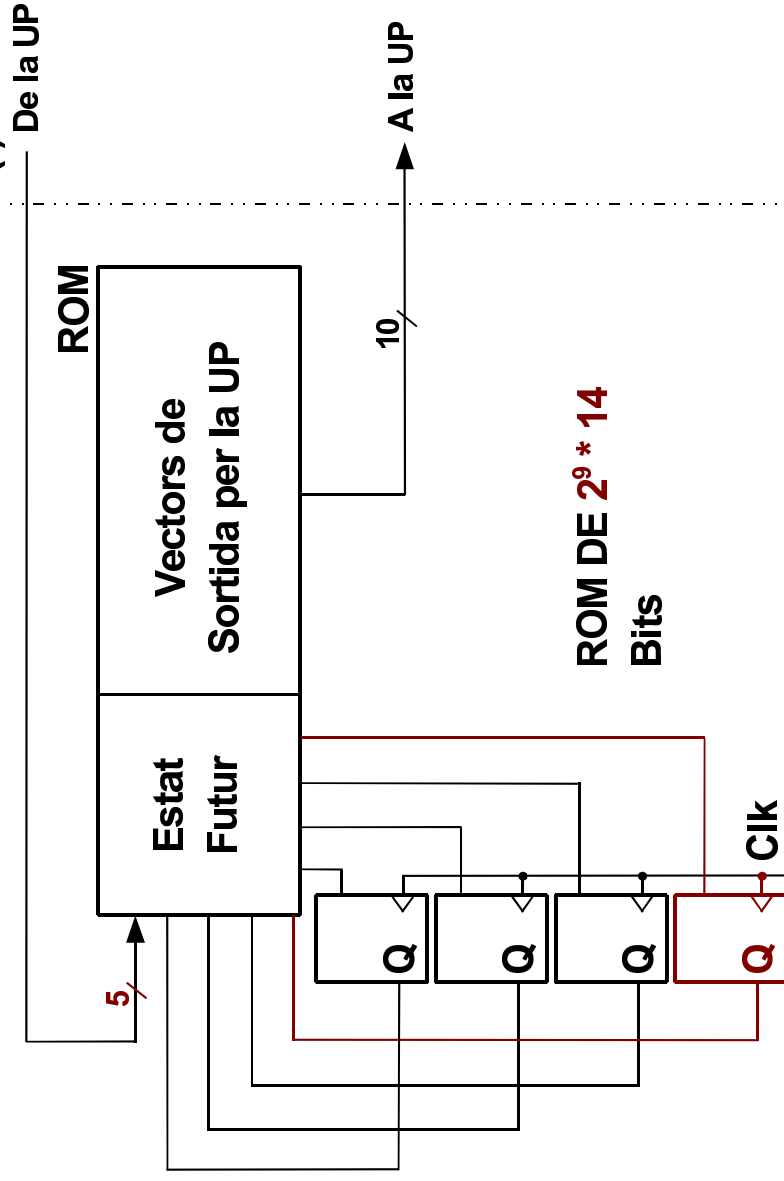
	S [*] ₀	S [*] ₁	S ₁₃	S ₁₄	S ₁₅	S ₁₆	S ₁₇
MX ₋₁	0	X	1	0	1	1	1
MX ₀	0	X	1	0	1	1	1
ALU ₁	X	X	1	X	1	X	0
ALU ₀	X	X	1	X	0	X	0
L / E	0	0	1	0	1	0	1
PC ← @+1	1	0	0	1	0	0	0
IR ← M	1	0	0	0	0	0	0
A ← M	0	0	0	0	0	1	0
B ← M	0	0	0	1	0	0	0
FZ ← Z	0	0	0	0	1	0	1

***NOTA:**

- S'ha partit de l'autòmata semi-simplificat de la MS1 sense realitzar la darrera modificació (fusionar els estats S₁ i S₂ generant l'estat S₁₂).

ETC – LA MÀQUINA SENZILLA – Modificació de la MS1

REIMPLEMENTACIÓ DE LA UC CABLEJADA DE LA MS1 (I)



ETC – LA MÀQUINA SENZILLA – Modificació de la MS1

VALORACIÓ DE LA MODIFICACIÓ

- S'han afegit 3 noves instruccions a la MS1 original.
- S'ha realitzat la modificació per **Hardware**.
- Les modificacions han produït un redisseny de:
 - El Format d'Instrucció.
 - La Unitat de Procés.
 - Afegir la nova operació (0) a la **ALU**.
 - La Unitat de Control.
 - Cal afegir un **nou biestable** (el graf té 12 estats).
 - Cal redefinir el contingut de la ROM.
 - La ROM hauria de ser de **9 línies d'adreces i 14 línies de dades**.

MILLORES POSSIBLES

- Implementar una unitat de control **Microprogramada**.
 - Permetrà modificar la **MS1** fent més simples els canvis (**UC**).
 - Més flexibilitat a l'hora de fer modificacions.
 - La memòria ROM utilitzada es reduirà dràsticament.

ETC – LA MÀQUINA SENZILLA – Microprogramació

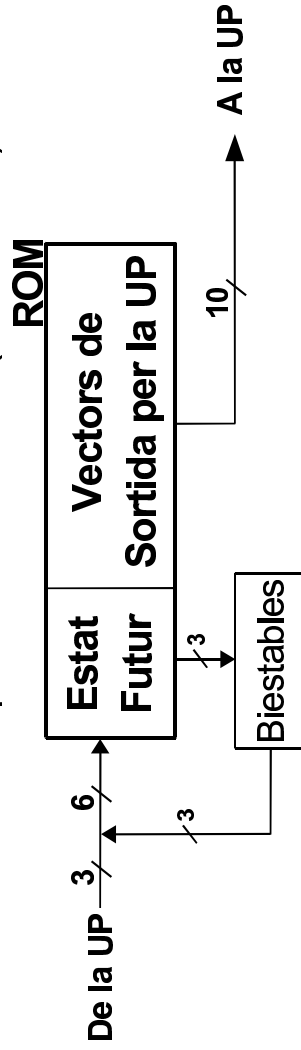
CPU – MS1

- Unitat de Procés (**UP**)
 - Blocs combinacionals.
 - Blocs seqüencials.
- Unitat de Control (**UC**)
 - Sistema seqüencial (implementació d'un autòmata d'estats).
 - Implementat amb qualsevol mètode de síntesi seqüencial:
 - Karnaugh, ROM, ...
 - En una **CPU Real** l'autòmata és molt més complex:
 - El vector de sortides de la UC pot ser molt gran.
 - Molts estats interns.

ETC – LA MÀQUINA SENZILLA – Microprogramació

MS1 – Valoració de la implementació de la UC

- Autòmata
 - 7 Estats
 - 3 Entrades
 - 10 Sortides
- Si s'implementa amb una ROM
 - ROM de 2⁶ paraules de 13 bits (64 * 13 Bits)



- És poc eficient utilitzar una ROM tan gran per representar només les 7 coses diferents que poden passar (7 Estats).
- Solució: **MICROPROGRAMACIÓ**

ETC – LA MÀQUINA SENZILLA – Microprogramació

SEQÜÈNCIES D'OPERACIONS (I)

- La MS1 original només executa 4 operacions diferents.
 - Execució d'una operació:
 - Seqüència de petites operacions:
 - Seqüència de senyals de control cap a la UP.

ADD F, D

Seqüència d'operacions

1	@ = PC	ALU = X	PC = @+1	IR = M	L/E = 0	; Fetch
2	@ = X	ALU = X				; Descodificació
3	@ = F	ALU = X	B = M	L/E = 0		; Cerca 1 ^{er} Op.
4	@ = D	ALU = X	A = M	L/E = 0		; Cerca 2 ^{on} Op.
5	@ = D	ALU = +	L/E = 1	FZ = Z		; Suma i guarda ; el resultat

ETC – LA MÀQUINA SENZILLA – Microprogramació

SEQÜÈNCIES D'OPERACIONS (II)

CMP F, D

Seqüència d'operacions

1	@ = PC	ALU = X	PC = @+1	IR = M	L/E = 0	; Fetch
2	@ = X	ALU = X				; Descodificació
3	@ = F	ALU = X	B = M	L/E = 0		; Cerca 1 ^{er} Op.
4	@ = D	ALU = X	A = M	L/E = 0		; Cerca 2 ^{on} Op.
5	@ = X	ALU = ⊕	FZ = Z			; Comparació

MOV F, D

Seqüència d'operacions

1	@ = PC	ALU = X	PC = @+1	IR = M	L/E = 0	; Fetch
2	@ = X	ALU = X				; Descodificació
3	@ = F	ALU = X	B = M	L/E = 0		; Cerca 1 ^{er} Op.
4	@ = D	ALU = B	L/E = 1	FZ = Z		; Moure

ETC – LA MÀQUINA SENZILLA – Microprogramació

SEQÜÈNCIES D'OPERACIONS (III)

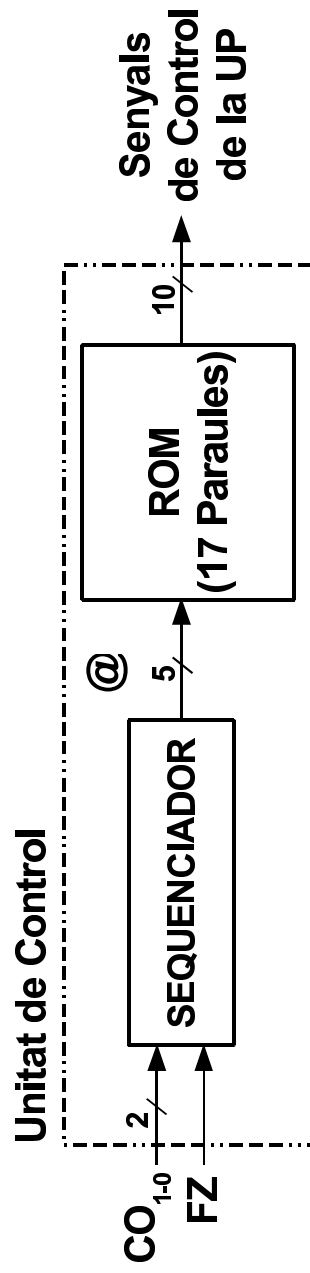
BEQ D

Seqüència d'operacions

- 1 @ = PC ALU = X PC = @+1 IR = M L/E = 0 ; Fetch
- 2 @ = X ALU = X ; Descodificació
- 3 @ = D ALU = X PC = @+1 IR = M L/E = 0 ; Fetch següent ; instrucció

Nova implementació de la UC

17 tasques de 10 bits → ROM $2^5 * 10$ Bits



ETC – LA MÀQUINA SENZILLA – Microprogramació

MICROINSTRUCCIONS (I) - Definició

- Seqüenciador: Ha de generar la seqüència d'adreces per executar les instruccions.
- ROM: Hi ha cinc grups de paraules → **Microinstruccions**

Contingut de la ROM

- Totes les instruccions comencen pel Fetch i la Descodificació:

PARAULA	MICROINSTRUCCIÓ				
0	@ = PC	ALU = X	PC = @ + 1	IR = M	L/E = 0 ; Fetch
1	@ = X	ALU = X			; Descodificació
2	@ = F	ALU = X	B = M	L/E = 0	; ADD
3	@ = D	ALU = X	A = M	L/E = 0	
4	@ = D	ALU = +	L/E = 1	FZ = Z	
5	@ = F	ALU = X	B = M	L/E = 0	; CMP
6	@ = D	ALU = X	A = M	L/E = 0	
7	@ = D	ALU = ⊕	FZ = Z		
8	@ = F	ALU = X	B = M	L/E = 0	; MOV
9	@ = D	ALU = B	L/E = 1	FZ = Z	
10	@ = D	ALU = X	PC = @ + 1	IR = M	L/E = 0 ; BEQ

ETC – LA MÀQUINA SENZILLA – Microprogramació

MICROINSTRUCCIONS (II) - Execució

- Existeixen 11 microinstruccions.
- Accions del Seqüenciador:
 - *Fetch* i *Descodificació*.
 - Segons el codi d'operació:
 - CO = ADD → Llegir la microinstrucció 2.
 - CO = CMP → Llegir la microinstrucció 5.
 - CO = MOV → Llegir la microinstrucció 8.
 - CO = BEQ → FZ = 1 → Llegir la microinstrucció 10.
FZ = 0 → Llegir la microinstrucció 0.
- L'execució és seqüencial fins que s'acaba la seqüència de microinstruccions d'una operació (4, 7 i 9), que cal tornar a l'adreça 0.
- Després de la microinstrucció 10 cal anar a la 1 (si FZ = 1).
- En general cal **executar seqüencialment** i en determinats casos **saltar** segons **CO₁**, **CO₀** i **FZ**. En altres casos cal saltar **incondicionalment**.

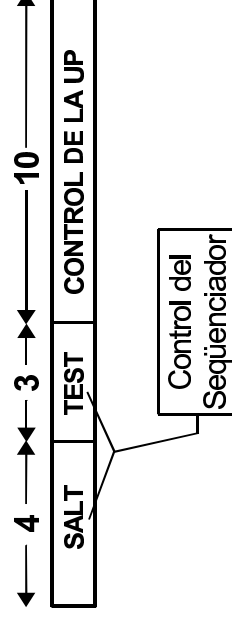
ETC – LA MÀQUINA SENZILLA – Microprogramació

MICROINSTRUCCIONS (III) - Format

Microinstrucció

- Camp de 10 bits de sortides cap a la UP.
- Camps per codificar els “salts” en el *microprograma*:
 - Què cal consultar per saltar.
 - Adreça on cal saltar.

Format



Codificació dels senyals

SENYAL	CODI
CO ₀	00
CO ₁	01
FZ	10
ZERO	11

; ZERO és un senyal que sempre val 0

ETC – LA MÀQUINA SENZILLA – Microprogramació

MICROINSTRUCCIONS (III) - Funcionament

Camp TEST (3 Bits):

- 2 Bits codifiquen un dels senyals (CO₁, CO₀, FZ, ZERO)
- El bit de **més pes** serveix per comparar amb el senyal codificat.

Funcionament lògic del seqüenciador:

S_i Valor_Senyal (TEST₁, TEST₀) == TEST₃ **Llavors**

Executar la microinstrucció del camp SALT.

Altrament

Executar la microinstrucció següent.

Fisi

ETC – LA MÀQUINA SENZILLA – Microprogramació

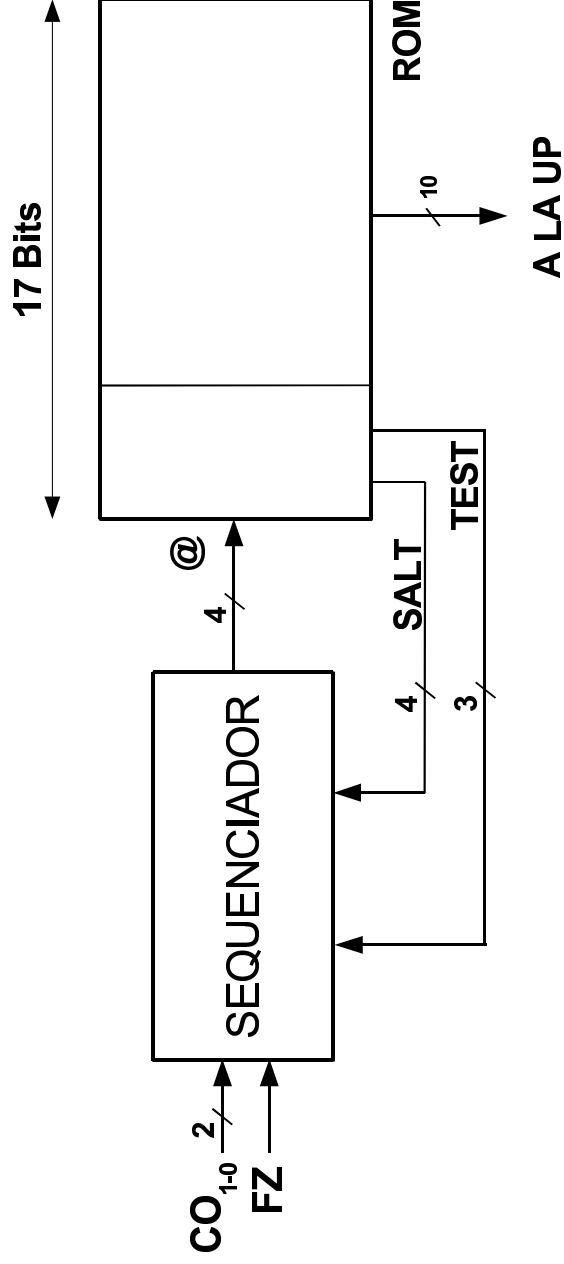
MICROINSTRUCCIONS (IV) – Contingut de la ROM

ROM de 2⁴ * 17 Bits

Paraula	SALT	TEST	MX ₁	MX ₀	ALU ₁	ALL ₀	L/E	PC	IR	A	B	FZ	COMENTARI
0	XXXX	111	0	0	X	X	0	1	1	0	0	0	Fetch
1	1000	100	X	X	X	X	0	0	0	0	0	0	Si CO ₀ = 1 anar a 8
2	0110	101	X	X	X	X	0	0	0	0	0	0	Si CO ₁ = 1 anar a 6
3	XXXX	111	1	0	X	X	0	0	0	0	1	0	CO = 00: ADD
4	XXXX	111	1	1	X	X	0	0	0	1	0	0	Cercar Operands
5	0000	011	1	1	0	0	1	0	0	0	0	1	Suma i anar a 0
6	XXXX	111	1	0	X	X	0	0	0	0	1	0	CO = 10: MOV
7	0000	011	1	1	1	0	1	0	0	0	0	1	Moure, flag FZ, anar a 0
8	1100	101	X	X	X	X	0	0	0	0	0	0	Si CO ₁ = 1 anar a 12
9	XXXX	111	1	0	X	X	0	0	0	0	1	0	CO = 01: CMP
10	XXXX	111	1	1	X	X	0	0	0	1	0	0	Cercar Operands
11	0000	011	X	X	0	1	0	0	0	0	0	1	⊕, flag FZ i anar a 0
12	0000	010	X	X	X	X	0	0	0	0	0	0	CO = 11: BEQ, FZ == 0? anar a 0
13	0001	011	1	1	X	X	0	1	1	0	0	0	FZ == 1? Fetch, anar a 1

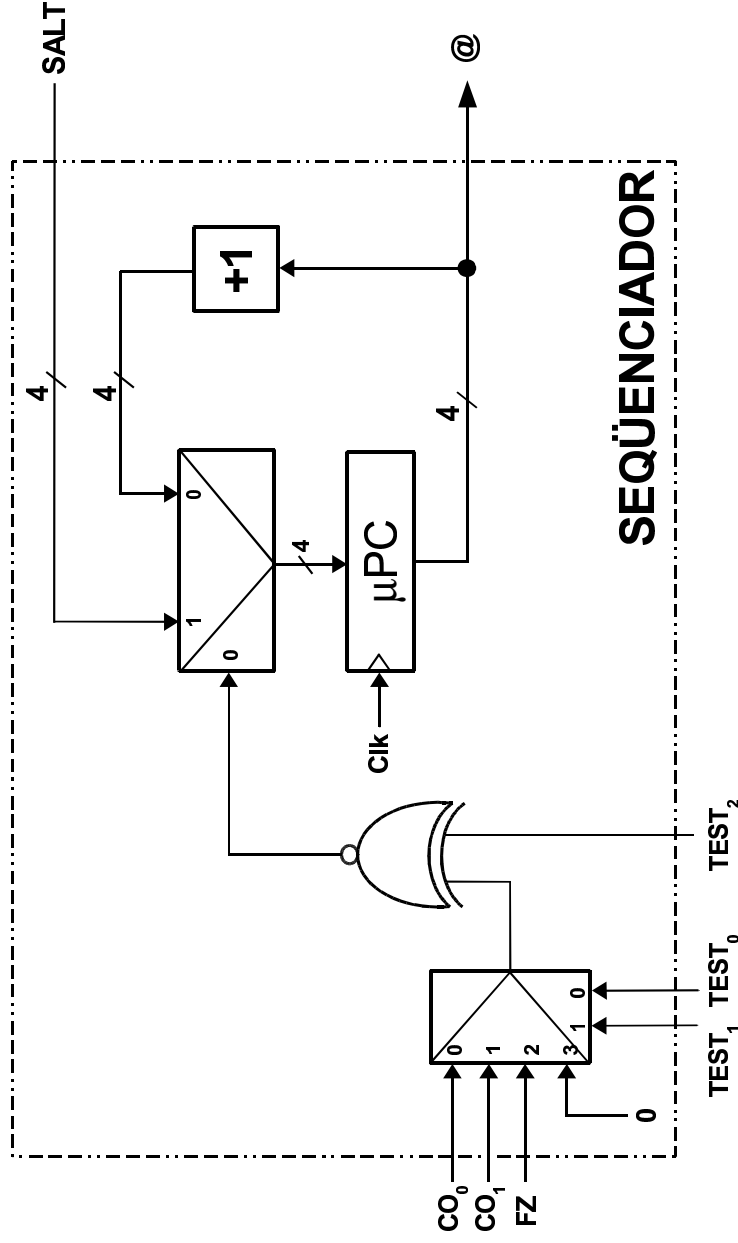
ETC – LA MÁQUINA SENZILLA – Microprogramació

MICROINSTRUCCIONS (V) – Implementació (I)



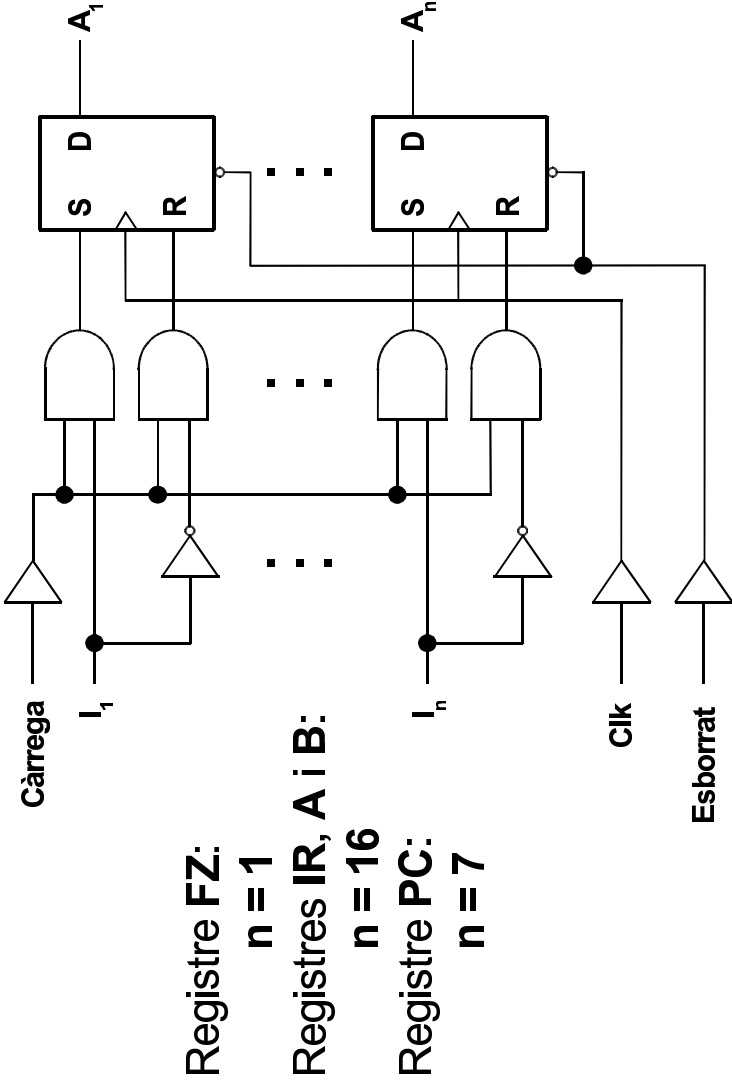
ETC – LA MÁQUINA SENZILLA – Microprogramació

MICROINSTRUCCIONS (VI) – Implementació (II)



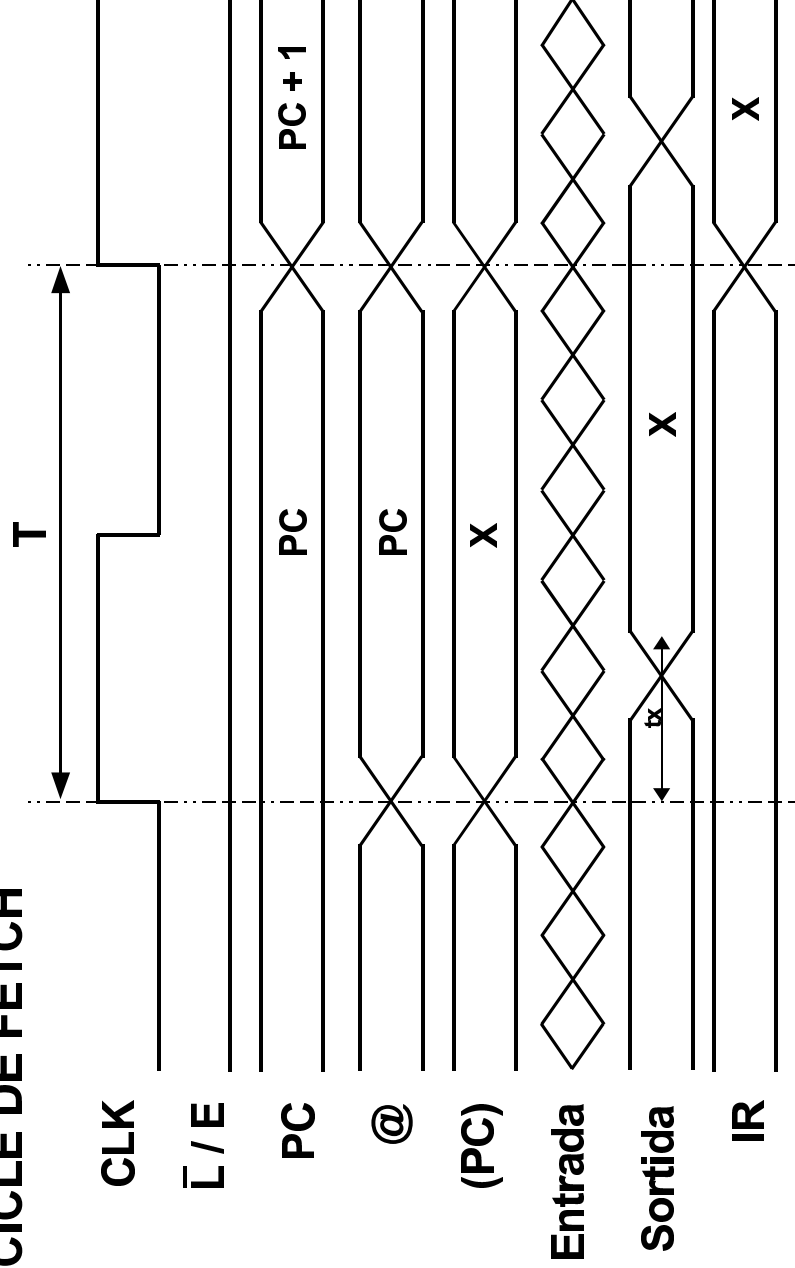
ETC – LA MÀQUINA SENZILLA – Apèndix (I)

REGISTRES AMB SENYAL DE CÀRREGA I DE RESET



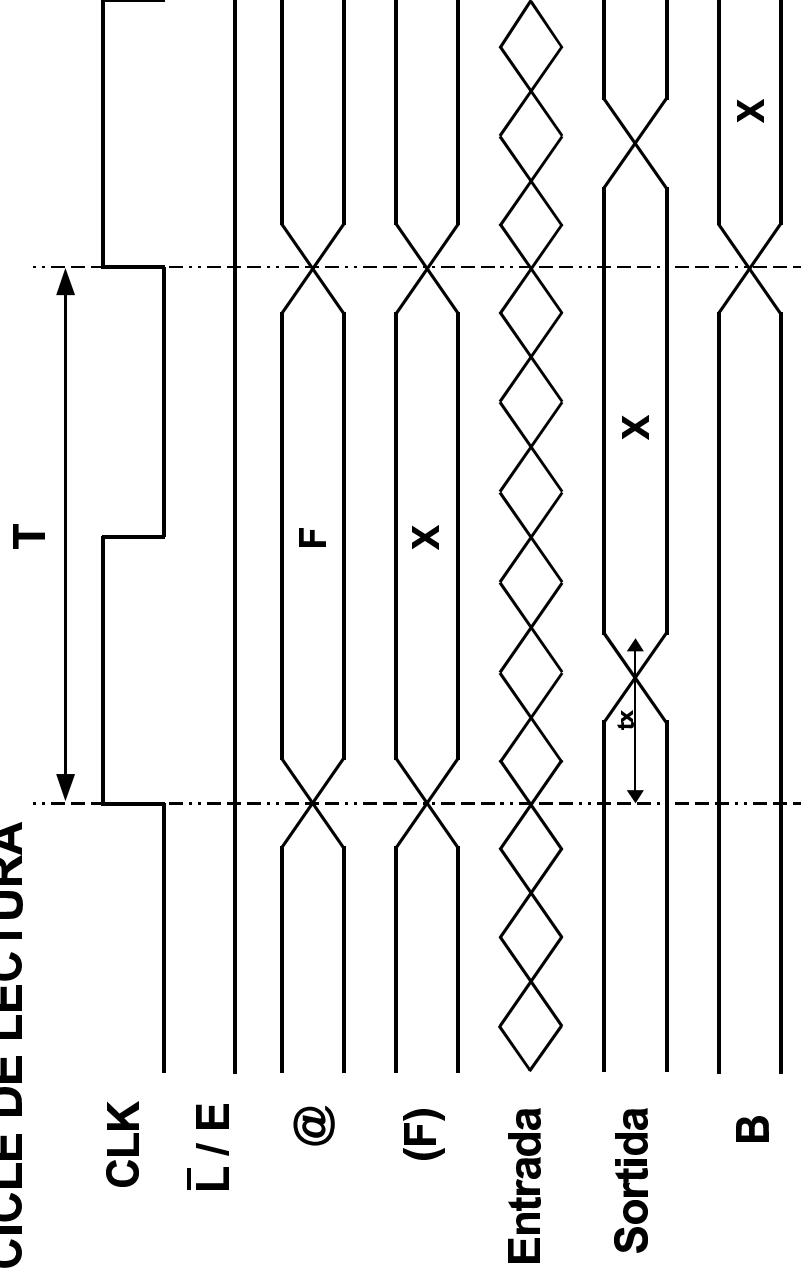
ETC – LA MÀQUINA SENZILLA – Apèndix (II)

CICLE DE FETCH



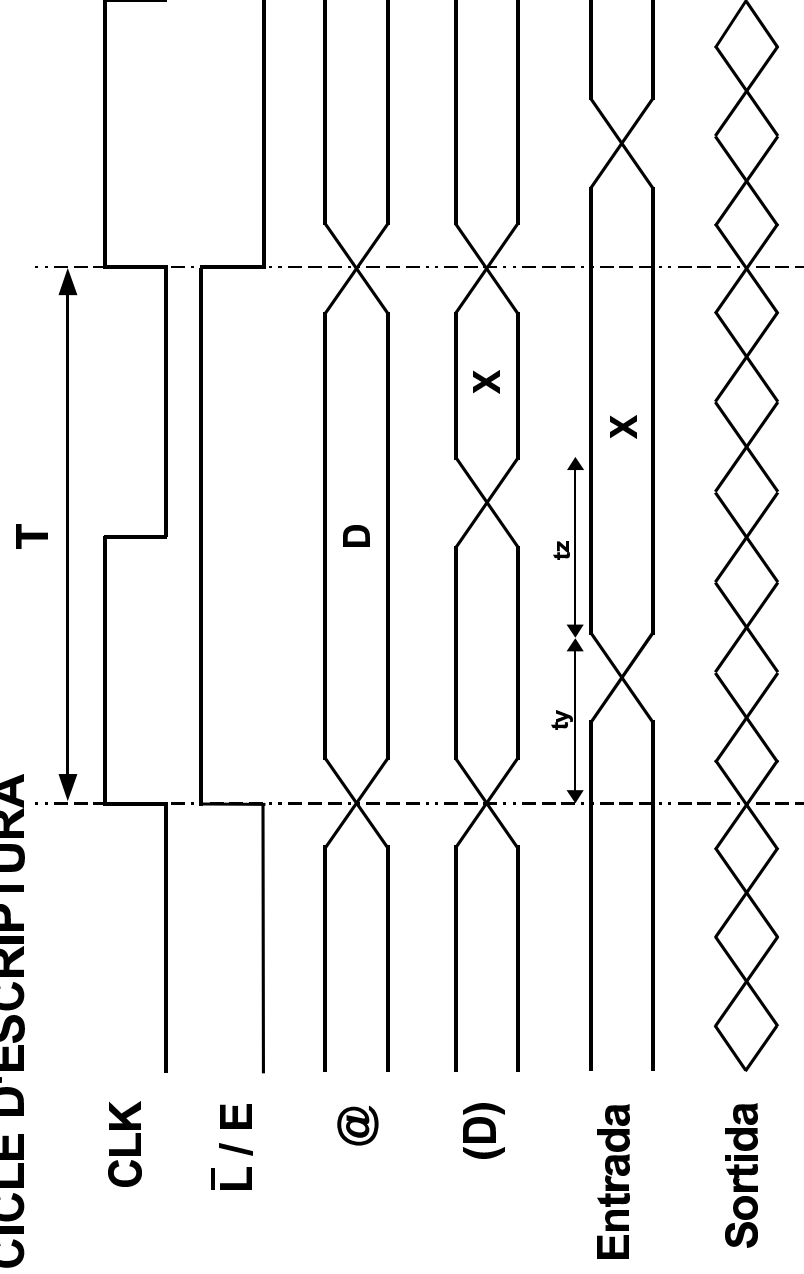
ETC – LA MÀQUINA SENZILLA – Apèndix (III)

CICLE DE LECTURA



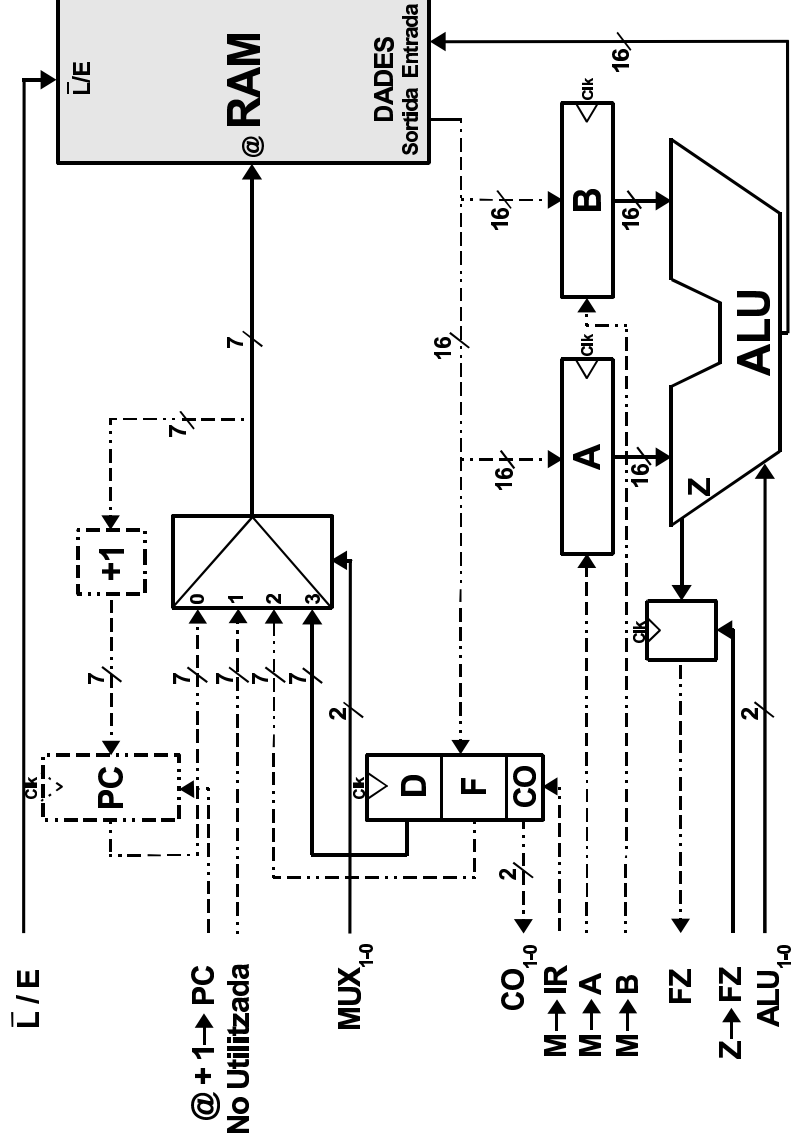
ETC – LA MÀQUINA SENZILLA – Apèndix (IV)

CICLE D'ESCRITURA



ETC – LA MÀQUINA SENZILLA MS1 - Apèndix (VII)

CAMINS DE DADES – EXECUCIÓ D'UN ADD



ETC – LA MÀQUINA SENZILLA MS1 - Apèndix (VIII)

CAMINS DE DADES – EXECUCIÓ D'UN BEQ

