



UdG

Universitat de Girona

ESTRUCTURA i TECNOLOGIA de COMPUTADORS

Transparències del Tema 7
La Màquina Senzilla (MS1)

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

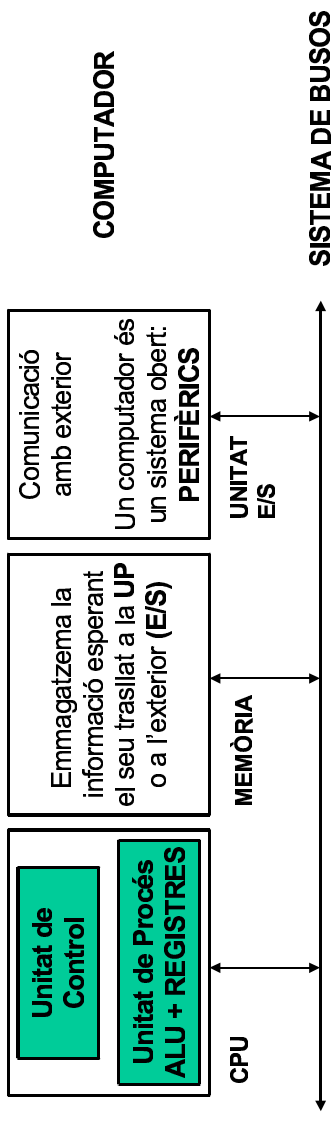
ETC - Esquema del Tema 7:

*Estructura bàsica d'un computador
La Màquina Senzilla MS1*

- Objectiu
- Programació
- Memòria
- Format d'Instrucció
- Dades
- Estructura General
- Disseny de la Unitat de Procés (UP)
 - ALU
 - Registres
 - Flags
 - Adreçament a Memòria
 - Esquema General
- Disseny de la Unitat de Control (UC)
 - Tipus d'UCs
 - Fases d'Execució
 - Graf d'Estats
 - Simplificació del Graf d'Estats
 - Implementació de la UC

ETC - Estructura d'un Computador (I)

COMPUTADOR: Sistema seqüencial capaç de realitzar un conjunt d'operacions bàsiques que s'anomenen **INSTRUCCIONS**.



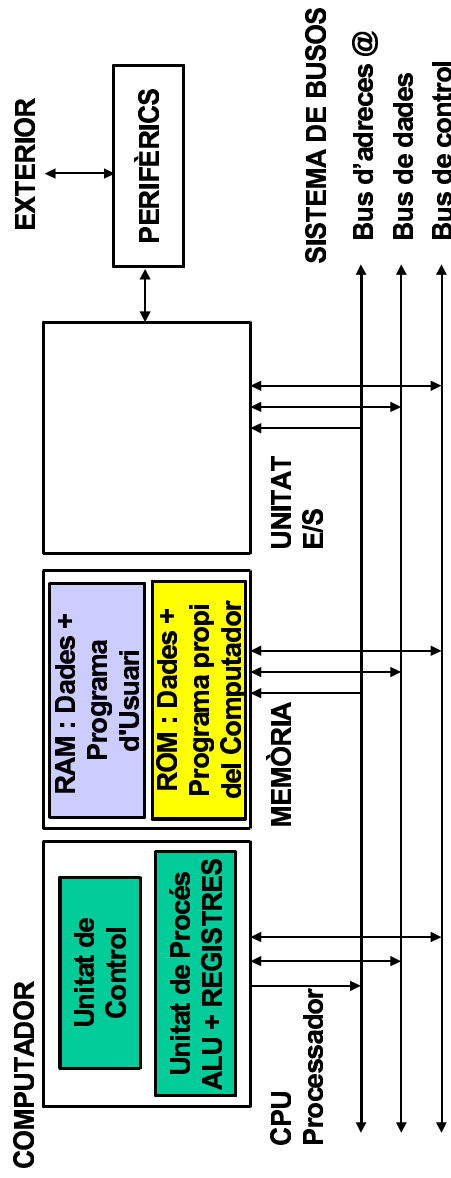
PROGRAMA: Seqüència d'**INSTRUCCIONS** que resol un problema determinat.

UNITAT DE CONTROL (UC): Conjunt de blocs que governen el funcionament de la **UP**. Per a cada instrucció del programa, la **UC** decideix les accions que s'ha de realitzar sobre la **UP** i la resta de blocs del computador.

UNITAT DE PROCÉS (UP): Conjunt de blocs destinats a fer operacions sobre les dades.

- **UNITAT ARITMÈTICO-LÒGICA (ALU):** Conjunt de blocs destinat a fer operacions aritmètiques i lògiques.
- **REGISTRES:** Emmagatzemen dades dels operands i resultats de les operacions. Emmagatzemen tota la informació necessària per executar les instruccions.
- **DATAPATH:** Conjunt dels cables i dispositius de selecció i encaminament (com ara **MUX**, **DEMUX**, ...) que calguin per connectar els diferents elements de la **UP** i la **UC**.

ETC - Estructura d'un Computador (II)



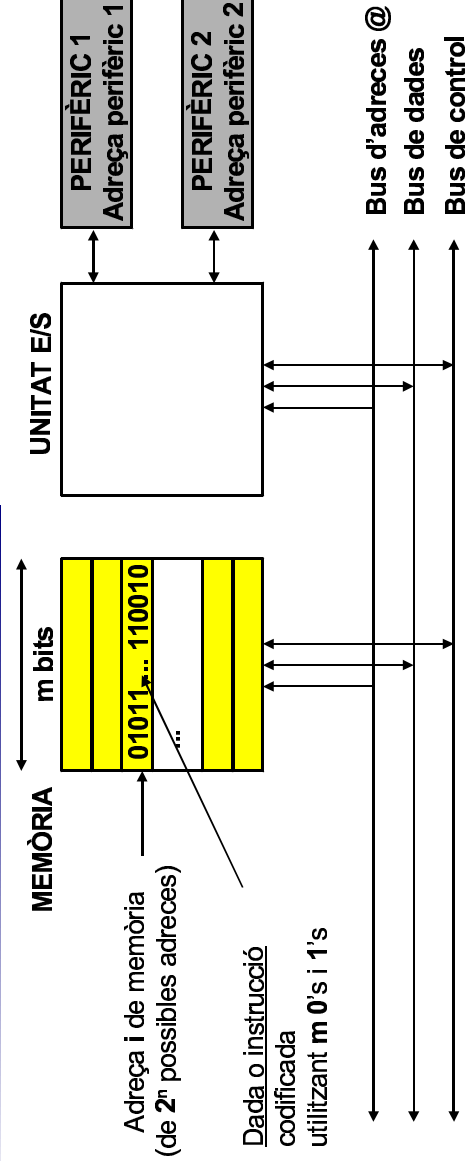
CPU: Per a l'usuari, es caracteritza pel seu **Repertori d'Instruccions** (conjunt d'operacions bàsiques que pot executar). Per cada instrucció la **UC** realitza una seqüència d'accions sobre la **UP** per a portar a terme l'operació indicada en la instrucció.

MEMÒRIA: Emmagatzema dos tipus d'informació.

- **DADES:** Variables i constants emprades pel programa.
- **PROGRAMES:** Seqüències d'instruccions.

MEMÒRIA ROM : Programes i dades propis del computador.

ETC - Estructura d'un Computador (III)



SISTEMA DE BUSOS

BUS D'ADRECES: És unidireccional. La CPU indica l'adreça de memòria on es troba la dada o instrucció a la qual vol accedir.

BUS DE DADES: És bidireccional (pot fer lectures o escriptures). S'utilitza per realitzar la transferència del contingut de l'adreça seleccionada.

BUS DE CONTROL: És bidireccional. Permet controlar la transferència de dades que s'ha de realitzar: la lectura o l'escriptura, a memòria o a perifèric, *Clk*, etc...

ETC – LA MÀQUINA SENZILLA MS1: OBJECTIU

Disseny de la CPU d'un computador senzill, definit pel conjunt d'instruccions que sigui capaç d'executar i per la memòria que sigui capaç d'adreçar.

REPERTORI D'INSTRUCCIONS: 4 Instruccions diferents

NOM	MNEMÒNIC	OPERACIÓ	COMENTARIS
1. SUMA	ADD F, D	(D) ← (D) + (F) FZ ← (D) + (F) = 0	El FZ és el Flag de Zero que indica que el resultat de la darrera operació és 0.
2. TRANSFERÈNCIA	MOV F, D	(D) ← (F) FZ ← (F) = 0	Copia el contingut de l'adreça font a l'adreça destí i a més actualitza FZ.
3. COMPARACIÓ	CMP F, D	(D) – (F) FZ ← (F) – (D) = 0	Actualitza FZ amb la comparació entre dos valors. No emmagatzema el resultat.
4. SALT	BEQ D	Si FZ = 1 Llavors PC ← D	Salt condicional. Permet trencar la seqüencialitat implícita del programa.

NECESSITATS

Les posicions de memòria D, F, ...

El Flag de zero FZ.

El registre comptador de programa PC.

ETC – LA MÀQUINA SENZILLA MS1: PROGRAMACIÓ

EXEMPLE PROGRAMACIÓ

Multiplicació de dues variables a i b, deixant el resultat en la variable c

RECORDEU

- EXECUTAR UN PROGRAMA:**
1. Carregar una instrucció de memòria a la CPU.
 2. Executar-la.
 3. Tornar al pas 1.

L'enguatge d'Alt Nivell		L'enguatge Assemblador		L'enguatge Màquina					
	Etiqueta		Comentari	@	Codi Op	@ Font	@ Destí	@	Hex
c=a*b									
	Begin								
	c := 0	MOV Zero, c	; c:=0	0	2	105	102	0	0xB4E6
	i := 0	MOV Zero, i	; i:=0	1	2	105	103	1	0xB4E7
	While i<b Do	CMP i, b	; Mentre	2	1	103	101	2	0x74E5
	c := c + a	BEQ End	; i<b	3	3	X	8	3	0xC008
	i := i + 1	ADD a, c	; c:=c+a	4	0	100	102	4	0x3266
	end	ADD Un, i	; i:=i+1	5	0	104	103	5	0x3467
	end	CMP x, x		6	1	105	105	6	0x74E9
		BEQ While		7	3	X	2	7	0xC002
	End:			8				8	

ETC – LA MÀQUINA SENZILLA MS1: MEMÒRIA

MEMÒRIA RAM

Les dades i el programa han de residir a memòria.

Utilitzarem una RAM amb una capacitat màxima de 128 paraules.

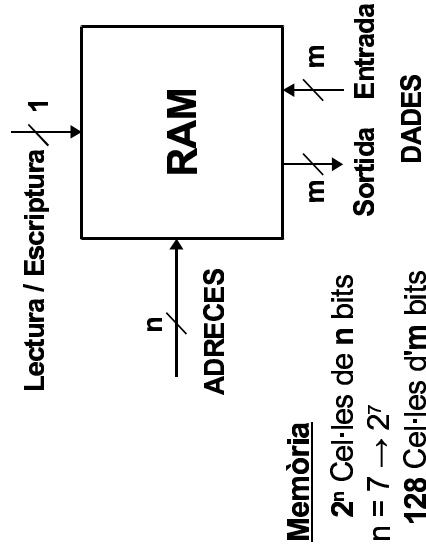
Cal emmagatzemar: **Dades: Variables i Constants i Programa: Instruccions.**

Suposant que les variables i constants s'emmagatzemen a partir de l'adreça 100 i el programa a partir de l'adreça 0, es pot definir la següent assignació a partir dels identificadors del programa anterior:

Identificador Adreça

Etiquetes: Begin: 0
While: 2
End: 8

Variables i Constants: a 100
b 101
c 102
i 103
Un 104
Zero 105



ETC – LA MÀQUINA SENZILLA MS1: FORMAT D'INSTRUCCIÓ

INSTRUCCIONS: Cal codificar-les a memòria en binari.

FORMAT D'INSTRUCCIÓ: Organització de la informació en camps de bits que es necessita per executar les instruccions. Cal especificar:

Quina és la instrucció: **CODI D'OPERACIÓ (CO)**.

Com s'especificuen els operands: **on estan i on cal emmagatzemar el resultat**.

La **MS1 té 4 instruccions diferents**: el **CO** precisa de **2 bits**:

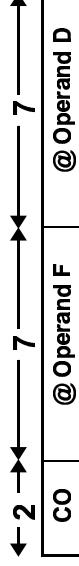
CO ₁	CO ₀	OPERACIÓ	OPERANDS
0	0	ADD	F, D (Adreces)
0	1	CMP	F, D (Adreces)
1	0	MOV	F, D (Adreces)
1	1	BEQ	D (Adreça)

Tres instruccions tenen 2 operands:

Necessitem, **com a mínim, dos** camps per especificar on estan els operands.

La **MS1** utilitza el **mode d'adreçament directe o simple** on F i D són adreces o etíquetes que representen posicions de memòria. En aquest adreçament l'**amplada dels camps** depèn de la **grandària** de la **memòria disponible (n = 7)**.

Per tant, les instruccions tenen un format de **16 Bits = 2 + 7 + 7**



ETC – LA MÀQUINA SENZILLA MS1: DADES

La **MS1** operarà amb dades (variables i constants) **enteres positives de 16 bits**.

La **MEMÒRIA RAM** de la MS1: 128 paraules de 16 bits per

Instruccions

Dades: Variables i Constants

El programa anterior:

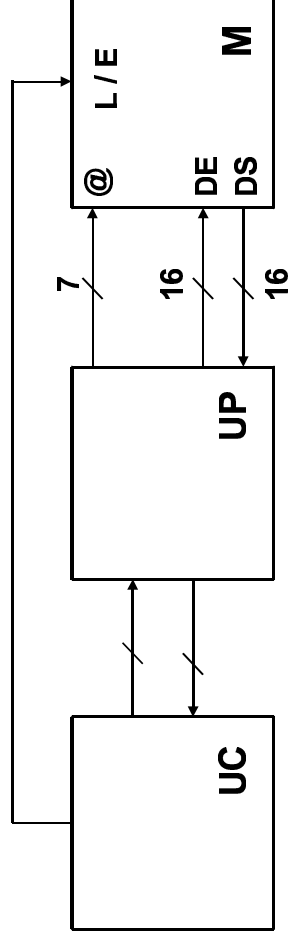
PROGRAMA

0	2	105	102
1	2	105	103
2	1	103	101
3	3	X	8
4	0	100	102
5	0	104	103
6	1	105	105
7	3	X	2
8

VARIABLES I CONSTANTS

a = 100	...
b = 101	
c = 102	
i = 103	
Un = 104	1
Zero = 105	0
	...

ETC – LA MÀQUINA SENZILLA MS1: ESTRUCTURA GENERAL



(NOTA: No hi ha bloc d'E/S)

Ara cal dissenyar la CPU del computador:

- Unitat de Procés: ALU + Registres Interns
- Unitat de Control

ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE PROCÉS (I)

DISSENY DE L'ALU

Ha de permetre realitzar les operacions que fan falta per realitzar les instruccions:

SUMA: Cal un sumador de números positius de 16 bits.

COMPARACIÓ: Es pot resoldre de dues formes diferents:

1. Amb un restador i un detector de zero.
2. Fent una comparació bit a bit utilitzant portes XOR.
S'ha escullit la segona opció ja que és molt més senzilla d'implementar.

TRANSFERÈNCIA: Deixar passar una dada de l'entrada fins a la sortida (per la instrucció **MOV**).

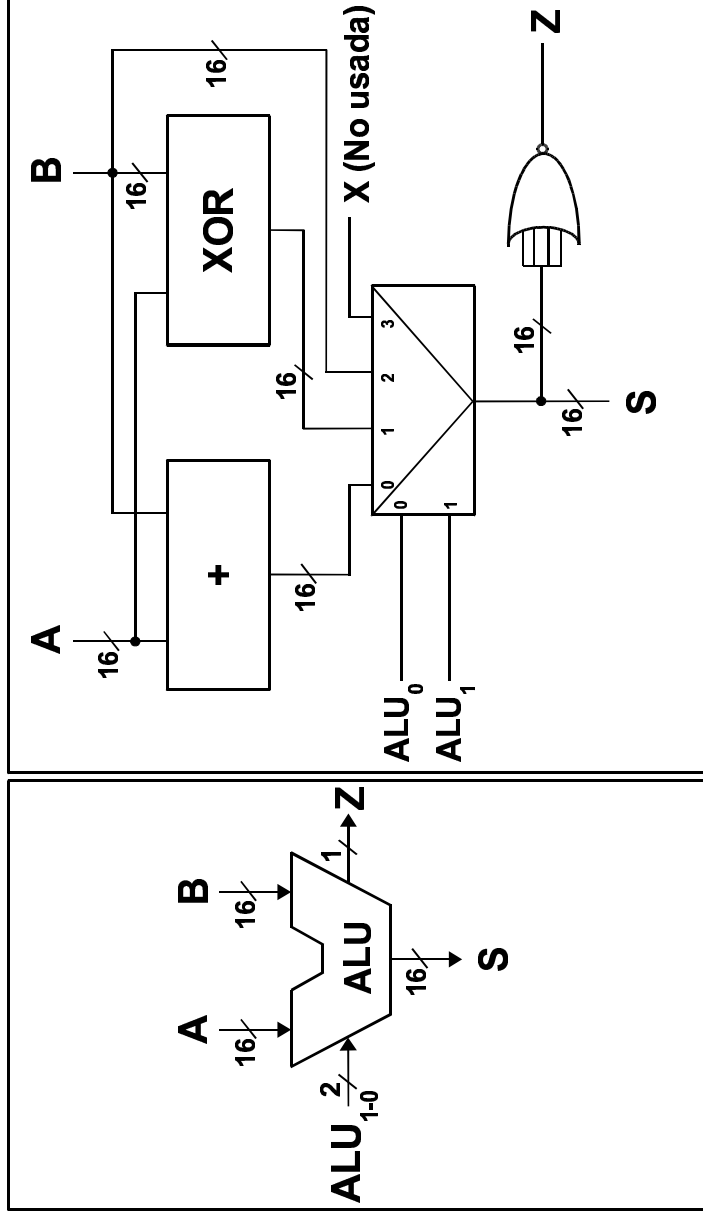
DETECCIÓ DE ZERO: Cal poder detectar si la darrera operació ha donat zero.

Control de les 3 operacions que es poden realitzar: codificació amb 2 bits (**ALU₁** i **ALU₀**):

ALU ₁	ALU ₀	OPERACIÓ
0	0	A + B
0	1	A xor B (bit a bit)
1	0	Transferència de B
1	1	No definida

ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE PROCÉS (II)

IMPLEMENTACIÓ DE L'ALU



ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE PROCÉS (III)

ELS REGISTRES: EL PC

Sequencialitat implícita de l'execució de programes:

Un programa s'executa de **forma seqüencial**:

Després d'executar la instrucció que ocupa la posició i , s'executa la de la posició $i + 1$.

Per tal d'implementar aquest seqüenciament, es necessita un **registre**:

PC (Program Counter): S'anirà incrementant a mida que es van executant les instruccions.

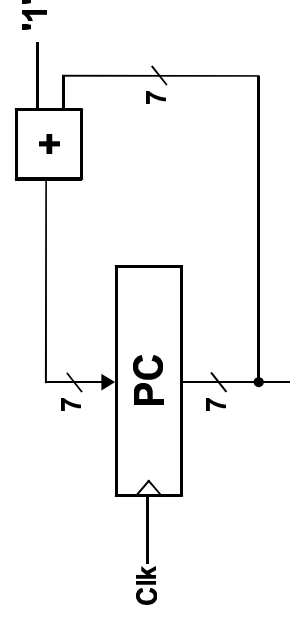
El PC indica, en cada moment, a quina adreça hi ha la **següent instrucció a executar**:

El PC **apunta** a la següent instrucció a executar.

En la MS1 serà, lògicament, un **registre de 7 bits**.

És necessari un mecanisme per **trencar la seqüencialitat** dels programes:

Les **instruccions de salt**.



@Següent Instrucció

ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE PROCÉS (IV)

ELS REGISTRES: IR, A i B (Disseny)

Per executar una instrucció cal realitzar varis accessos a memòria.

Per exemple:

ADD F, D

- 1^{er} accés: Anar a cercar la instrucció a la posició que apunta el PC.
- 2^{on} accés: Anar a cercar l'operand font que es troba a la posició de memòria F.
- 3^{er} accés: Anar a cercar l'operand destí a l'adreça D.
- 4^{rt} accés: Guardar el resultat a l'adreça D (accés d'escriptura).

On guardar aquesta informació? Calen 3 registres de 16 bits.

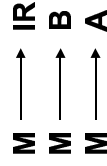
REGISTRE D'INSTRUCCIONS: Conté la instrucció en execució (**Instruction Register IR**).

REGISTRE B: Registre de dades per guardar el primer operand (el de la posició F).

REGISTRE A: Registre de dades pel segon operand (el de l'adreça D).

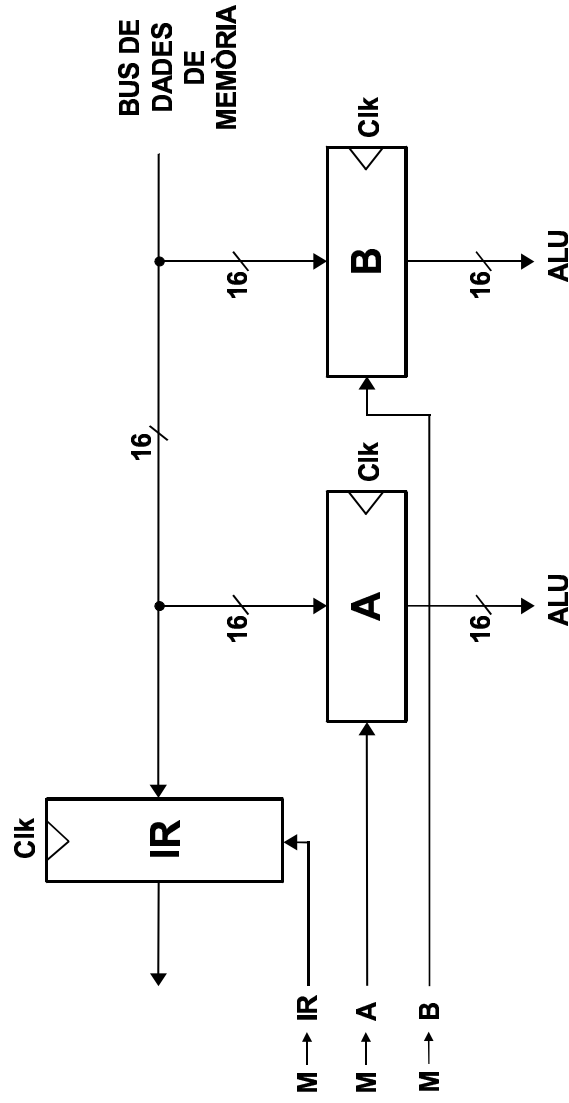
Línies de control d'accés als registres:

Senyals de càrrega dels registres (senyals d'Enable o Load).



ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE PROCÉS (V)

ELS REGISTRES: IR, A i B (Implementació)



ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE PROCÉS (VI)

ELS FLAGS: INDICADOR DE ZERO (FZ)

La instrucció **BEQ** utilitza l'indicador de zero:

S'anomena **Flag Zero** o simplement **FZ**.

Necessari perquè a l'execució de **BEQ D** es pugui decidir si cal o no cal saltar:

Si $FZ = 1$ **Llavors**

Saltar a l'adreça indicada per **D**

Altrament

Continuar a l'adreça que indica **PC**

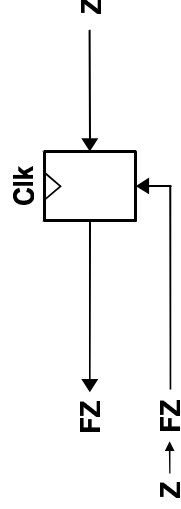
Fisi

Les instruccions **ADD**, **MOV** i **CMP** actualitzen aquest flag **FZ**.

Fa falta **un registre (1 bit) que guardi el bit Z** (resultat és zero) de sortida de l'**ALU**.

Cal guardar **Z** perquè es consultarà després de la seva generació.

Cal una línia de control per actualitzar aquest registre: $Z \rightarrow FZ$



ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE PROCÉS (VII)

ADREÇAMENT A MEMÒRIA

Possibles procedències de l'adreça de memòria:

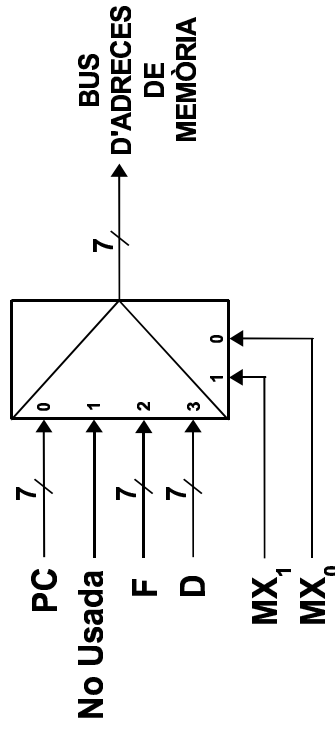
Del registre **PC**, en cas d'haver de llegir la **següent instrucció a executar**.

Del camp **F** de la instrucció en cas d'haver de llegir l'operand font.

Del camp **D** de la instrucció en cas d'haver de llegir l'operand destí o quan es guarda el resultat.

Per a la selecció d'una d'aquestes opcions fa falta un **MUX** de 4 entrades i 2 línies de control (MX_1 i MX_0):

Adreça	MX_1	MX_0
PC	0	0
No Utilitzada	0	1
F	1	0
D	1	1



ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE CONTROL (I)

Per dissenyar la **UNITAT de CONTROL** es pot utilitzar dues filosofies diferents:

- **UC CABLEJADA**: Disseny d'un **sistema seqüencial** definit per un **graf d'estats** associats a cada una de les fases d'execució de les instruccions.

Cada estat porta associat un **vector de sortides** que **controlen** la **UNITAT de PROCÉS** i la resta de **blocs del computador**.

Llavors **executar una instrucció** de **Llenguatge Màquina (LM)** és equivalent a **executar una seqüència d'estats**.

- **UC MICROPROGRAMADA**: autòmat controlat per un **microprograma resident en una memòria interna**.

Per cada instrucció del programa es té una **seqüència de microinstruccions** associades.

Llavors **executar una instrucció** de **Llenguatge Màquina (LM)** és equivalent a **executar un microprograma**.

EN PRIMER LLOC, S'UTILITZARÀ LA 1ª OPCIO:

Per tant, cal dissenyar un **sistema seqüencial** amb:

- **3 senyals d'entrada** procedents de la **UNITAT de PROCÉS**:
 CO_1, CO_0 (les 2 senyals del codi d'operació) i FZ
- **10 senyals de sortida** pel control dels blocs de la **UNITAT de PROCÉS**:
 $@+1 \rightarrow PC, \bar{L}/E, MUX_1, MUX_0, M \rightarrow IR, M \rightarrow A, M \rightarrow B, Z \rightarrow FZ, ALU_1, ALU_0$

ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE CONTROL (II)

FASES D'EXECUCIÓ D'UNA INSTRUCCIÓ

En general es pot parlar de l'existència de 4 fases d'execució d'una instrucció:

- **FASE 1.**
FETCH. Recerca a memòria de la instrucció apuntada pel registre **PC** per guardar-la al registre **IR**.
Increment del registre **PC** (seqüenciament implícit).
- **FASE 2.**
DESCODIFICACIÓ de la instrucció que hi ha a l'**IR**.
- **FASE 3.**
RECERCA DELS OPERANDS a memòria i/o avaluació del flag **FZ**.
- **FASE 4.**
EXECUCIÓ de la instrucció.
Emmagatzemament del resultats (a memòria i/o registres).

La **UNITAT de CONTROL** ha de controlar el **correcte seqüenciament** de les diferents fases de les instruccions i de l'activació dels blocs adequats de la **UP**, a partir del seguiment d'un **GRAF D'ESTATS** que cal dissenyar.

ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE CONTROL (III)

FASES D'EXECUCIÓ DE LES INSTRUCCIONS A LA MS1 (I)

ADD

Definició dels estats

Estat S_0	(Fase 1):	IR ← (PC)	PC = PC + 1
Estat S_1	(Fase 2):	Avaluació de CO_1 i CO_0	
Estat S_2	(Fase 3):	B ← (F)	
Estat S_3	(Fase 3):	A ← (D)	
Estat S_6	(Fase 3):	(D) ← A + B	
Estat S_7	(Fase 4):	FZ ← Z	

CMP

Definició dels estats

Estat S_0	(Fase 1):	IR ← (PC)	PC = PC + 1
Estat S_1	(Fase 2):	Avaluació de CO_1 i CO_0	
Estat S_3	(Fase 3):	B ← (F)	
Estat S_8	(Fase 3):	A ← (D)	
Estat S_9	(Fase 4):	$A \oplus B$	FZ ← Z

ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE CONTROL (IV)

FASES D'EXECUCIÓ DE LES INSTRUCCIONS A LA MS1 (II)

MOV

Definició dels estats

Estat S_0	(Fase 1):	IR ← (PC)	PC = PC + 1
Estat S_1	(Fase 2):	Avaluació de CO_1 i CO_0	
Estat S_4	(Fase 3):	B ← (F)	
Estat S_{10}	(Fase 4):	(D) ← B	FZ ← Z

BEQ

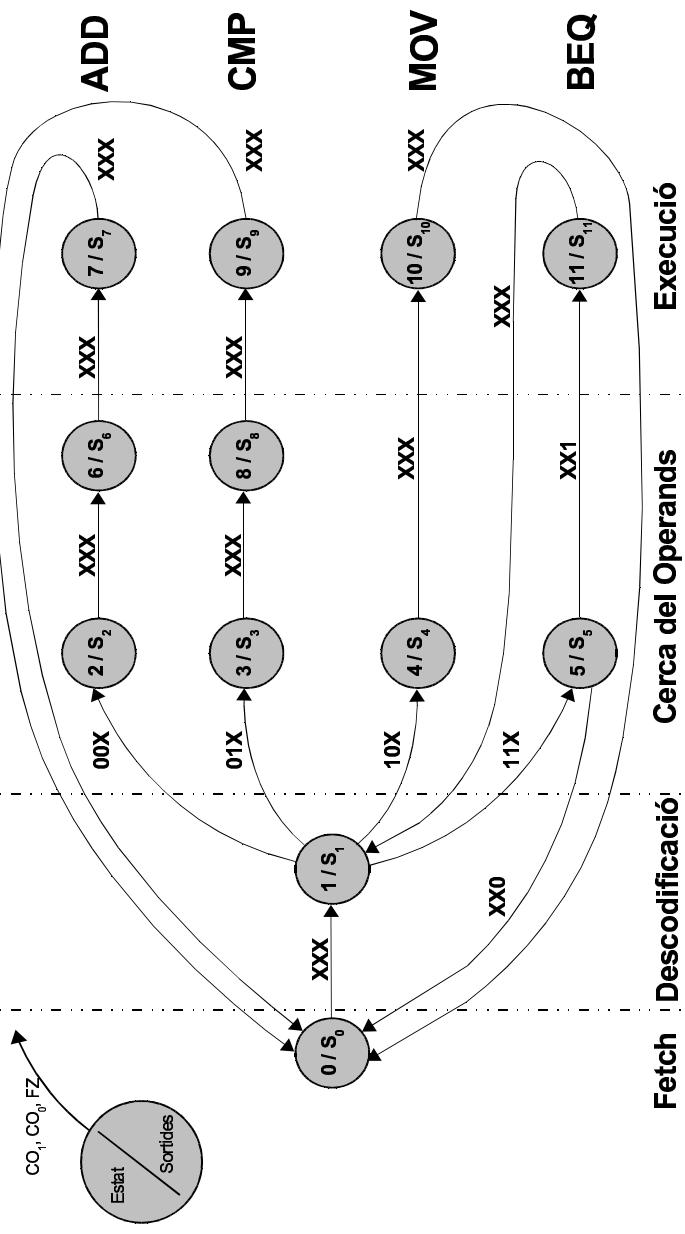
Definició dels estats

Estat S_0	(Fase 1):	IR ← (PC)	PC = PC + 1
Estat S_1	(Fase 2):	Avaluació de CO_1 i CO_0	
Estat S_5	(Fase 3):	Consulta de FZ	
Estat S_{11}	(Fase 1):	IR ← (D)	PC ← D + 1

*NOTA: L'execució d'un salt (BEQ) és el FETCH (Fase 1) de la instrucció que hi ha a l'adreça destí.

ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE CONTROL (V)

GRAF D'ESTATS DE LA UNITAT DE CONTROL



ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE CONTROL (VI)

VECTOR BINARI DE SORTIDES DE CADA ESTAT DEL GRAF

	S ₀	S ₁	S ₂	S ₃	S ₄	S ₅	S ₆	S ₇	S ₈	S ₉	S ₁₀	S ₁₁
MX ₁	0	X	1	1	1	X	1	1	1	X	1	1
MX ₀	0	X	0	0	0	X	1	1	1	X	1	1
ALU ₁	X	X	X	X	X	X	X	0	X	0	1	X
ALU ₀	X	X	X	X	X	X	X	0	X	1	0	X
L/E	0	0	0	0	0	0	0	1	0	0	1	0
PC ← @+1	1	0	0	0	0	0	0	0	0	0	0	1
IR ← M	1	0	0	0	0	0	0	0	0	0	0	1
A ← M	0	0	0	0	0	0	1	0	1	0	0	0
B ← M	0	0	1	1	1	0	0	0	0	0	0	0
FZ ← Z	0	0	0	0	0	0	0	1	0	1	1	0

EL GRAF INICAL DE 12 ESTATS es pot simplificar, hi ha estats que indiquen les mateixes operacions a la Unitat de Procés.

ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE CONTROL (VII)

SIMPLIFICACIÓ DEL GRAF D'ESTATS (I)

Les instruccions **ADD**, **MOV**, **CMP** tenen una fase comuna:

La cerca del primer operand

Es poden fusionar els tres estats **S₂**, **S₃** i **S₄** en un de sol.

Les instruccions **ADD** i **CMP** encara tenen una fase comuna:

La cerca del segon operand

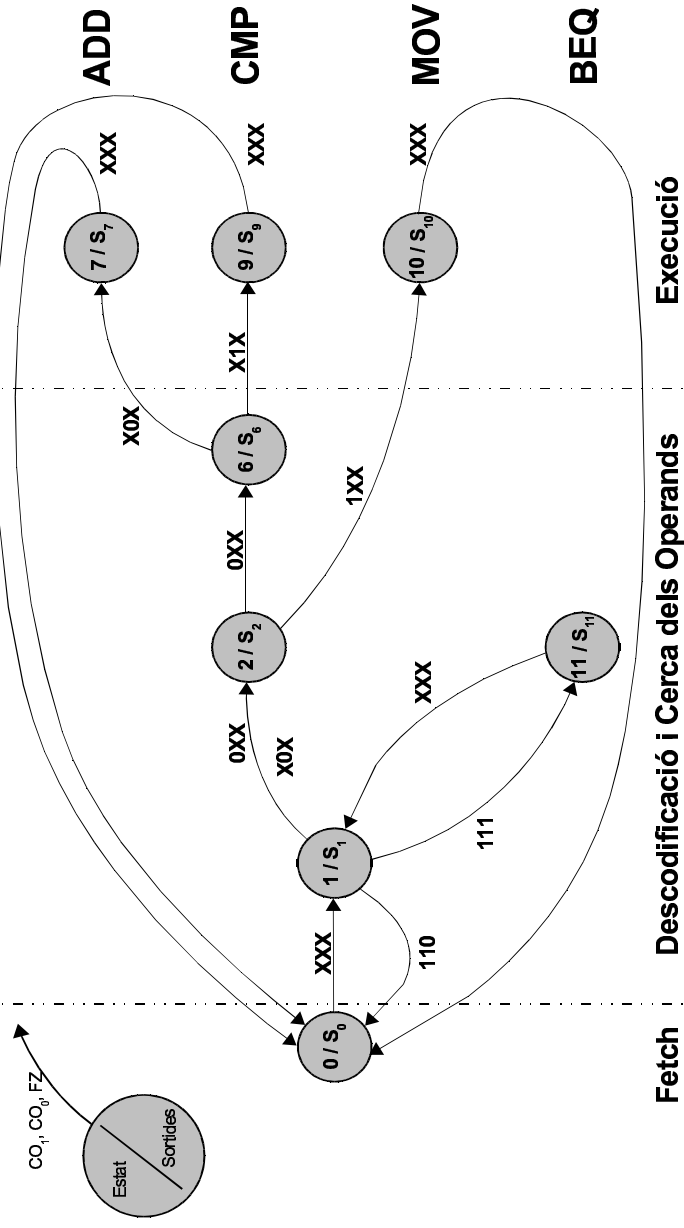
Es poden unificar els dos estats **S₆** i **S₈** en un de sol.

La consulta del flag **FZ** per decidir si saltar o no, es pot fer a l'estat **S₁**:

Es pot eliminar l'estat **S₅**

ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE CONTROL (VIII)

SIMPLIFICACIÓ DEL GRAF D'ESTATS (II)



ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE CONTROL (IX)

SIMPLIFICACIÓ DEL GRAF D'ESTATS (III)

Darrera optimització:

Unificar l'etapa S_1 de descodificació i S_2 de cerca del 1^{er} operand.

A l'etapa S_1 no s'accedeix a memòria.

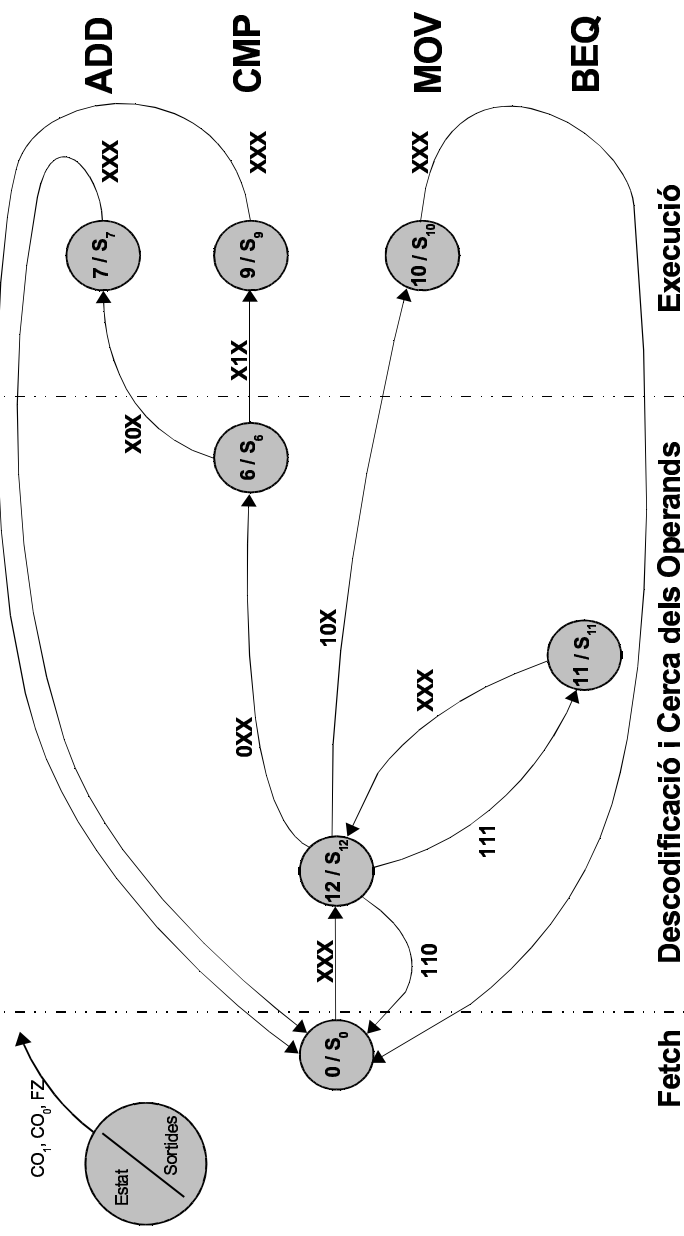
Si l'operació és **BEQ** s'ignora l'operand.

S'obté un nou estat S_{12}

	S_{12}
MX_1	1
MX_0	0
ALU_1	X
ALU_0	X
\bar{L}/E	0
$PC \leftarrow @+1$	0
$IR \leftarrow M$	0
$A \leftarrow M$	0
$B \leftarrow M$	1
$FZ \leftarrow Z$	0

ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE CONTROL (X)

SIMPLIFICACIÓ DEL GRAF D'ESTATS (IV)



ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE CONTROL (XI)

EFICIÈNCIA DE LES INSTRUCCIONS DE LA MS1

Cicles de rellotge

ADD	4 Cicles.
CMP	4 Cicles.
MOV	3 Cicles.
BEQ	2 Cicles.

Si FZ = 1 l'estat S_{11} fa el Fetch de la instrucció destí.

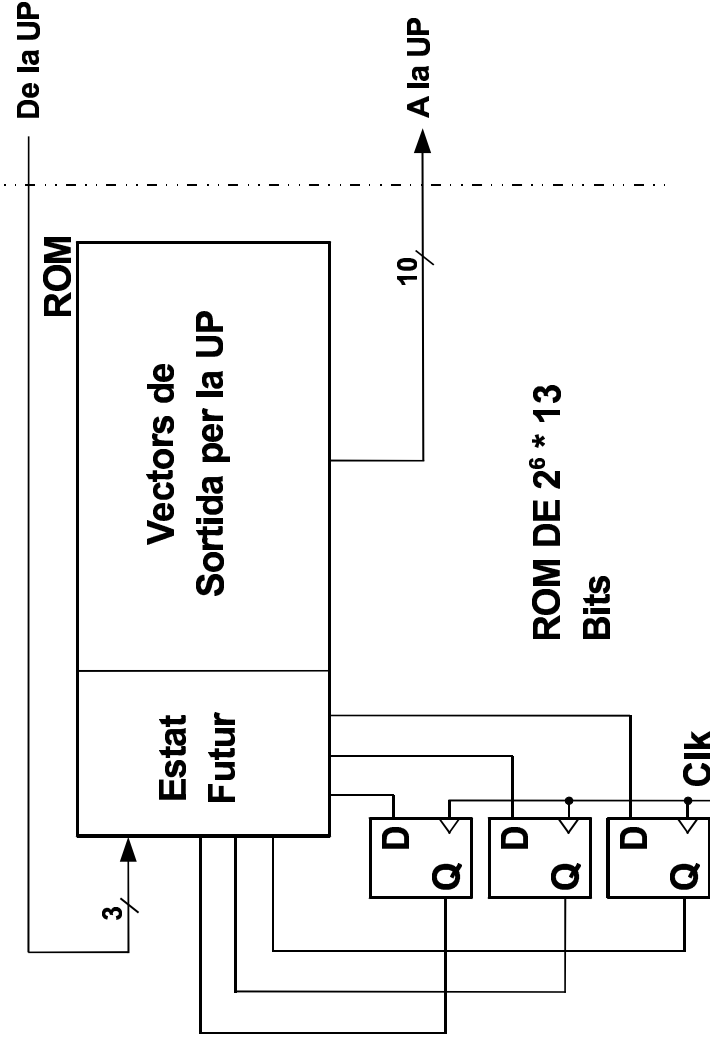
Accessos a memòria

ADD	4 Accessos.
CMP	3 Accessos.
MOV	3 Accessos.
BEQ	2 Accessos.

En la darrera versió optimitzada un accés és innecessari.

ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE CONTROL (XI)

IMPLEMENTACIÓ DE LA UC DE LA MS1 (I)



ETC – LA MÀQUINA SENZILLA MS1: UNITAT DE CONTROL (XIV)

IMPLEMENTACIÓ DE LA UC DE LA MS1. CONTINGUT DE LA ROM (III)

ADRECES		CONTINGUT DE LA ROM													
	*	*													
	*	*													
S ¹⁰	110000	000	1	1	1	0	1	0	0	0	0	0	0	1	
	110001	000	1	1	1	0	1	0	0	0	0	0	0	1	
	110010	000	1	1	1	0	1	0	0	0	0	0	0	1	
	110011	000	1	1	1	0	1	0	0	0	0	0	0	1	
	110100	000	1	1	1	0	1	0	0	0	0	0	0	1	
	110101	000	1	1	1	0	1	0	0	0	0	0	0	1	
	110110	000	1	1	1	0	1	0	0	0	0	0	0	1	
	110111	000	1	1	1	0	1	0	0	0	0	0	0	1	
	111000	XXX	X	X	X	X	X	X	X	X	X	X	X	X	X
	111001	XXX	X	X	X	X	X	X	X	X	X	X	X	X	X
111010	XXX	X	X	X	X	X	X	X	X	X	X	X	X	X	
111011	XXX	X	X	X	X	X	X	X	X	X	X	X	X	X	
111100	XXX	X	X	X	X	X	X	X	X	X	X	X	X	X	
111101	XXX	X	X	X	X	X	X	X	X	X	X	X	X	X	
111110	XXX	X	X	X	X	X	X	X	X	X	X	X	X	X	
111111	XXX	X	X	X	X	X	X	X	X	X	X	X	X	X	

NO DEFINIT