

Introducció als Ordinadors

Capítol 1

Característiques genèriques

Manel López

Introducció als computadors

DEFINICIONS PRÈVIES

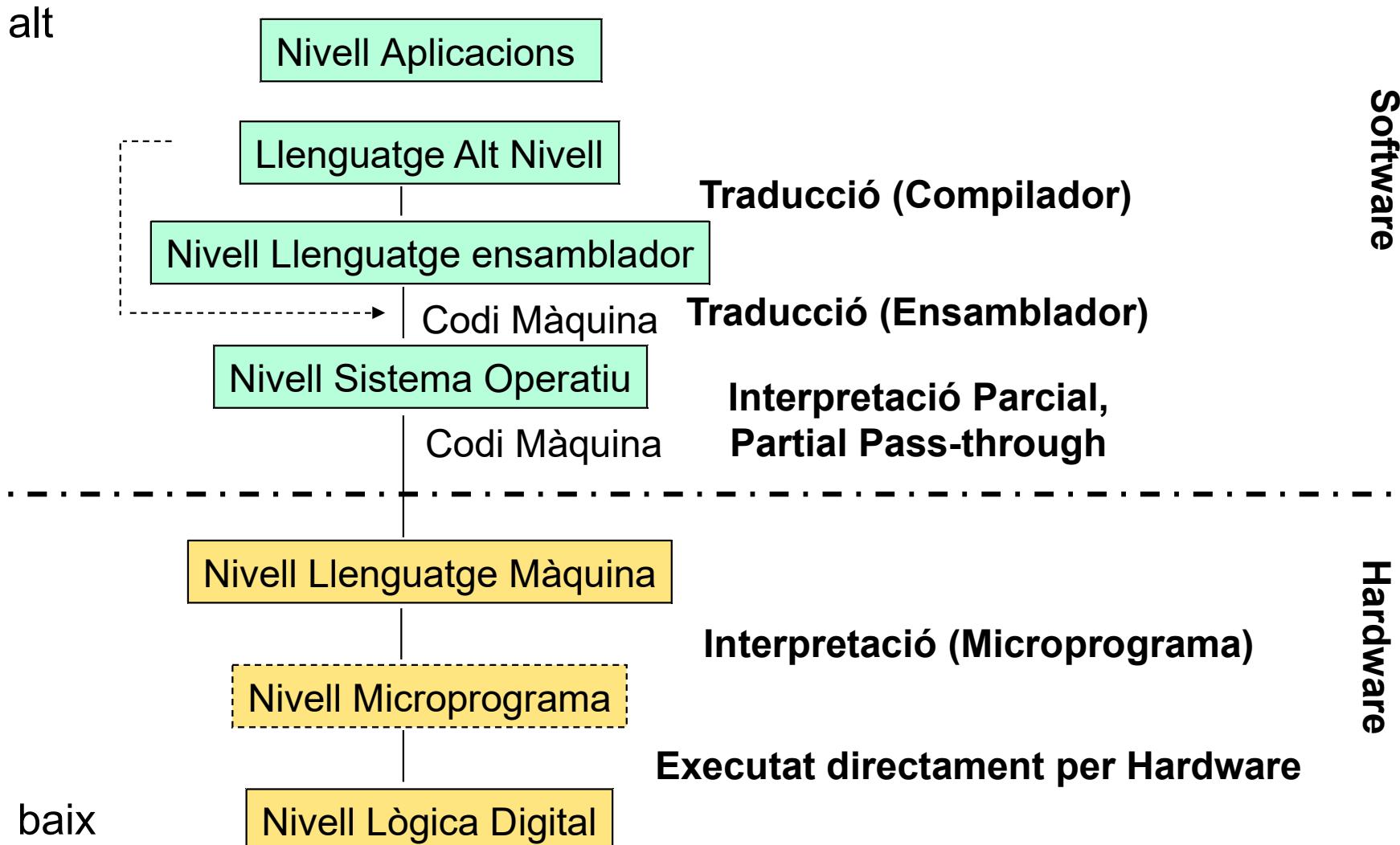
- En un sentit genèric, un ORDINADOR és aquell aparell dissenyat per processar informació.
- Un procès es correspon a les fases o transformacions que suporta la informació per tal de resoldre un problema determinat.
- L'arquitectura d'un computador defineix el seu comportament funcional.

Arquitectura i Organització

- **Arquitectura:** consisteix en aquells atributs que són visibles al programador
 - Conjunt d'instruccions
 - Nombre de bits usat per representació de dades
 - Mecanismes Entrada/Sortida
 - Tècniques d'adreça
 - P.ex.: Hi ha instrucció de multiplicar?
- **Organització:** consisteix en com s'han implementat, típicament amagats al programador
 - Senyals de control, interfícies, tecnologia de memòria
 - P.Ex.: Hi ha unitat de multiplicació per HW o es fa per addició repetida (algoritme)?

- Tota la família Intel x86 comparteixen la mateixa arquitectura bàsica
- La família System/370 comparteixen la mateixa arquitectura bàsica
- Això proporciona **compatibilitat** de codi
 - Almenys cap enrere
 - Però la complexitat augmenta a cada generació.
Potser seria més eficient començar nova arquitectura a cada nova tecnologia e.g. RISC vs. CISC
- Organització difereix entre versions diferents

Computador com a màquina multi-nivell



Nivells

Nivell Lògica Digital

- Unitats Funcionals: (ALU; Registres,...).
- Nivell portes lògiques; Computacions primitives reduïdes a operacions Booleanes (AND, OR, NOT)
- A aquest nivell no hi ha concepte de programa. Simplement seqüència d'operacions a ser processades

Nivell Microprograma

- Instruccions de llenguatge màquina. Interpretades. Provoca una sèrie d'instruccions simples a ser executades en el nivell inferior Lògica-Digital
- *Microprograma* per cada instrucció, guardat permanentment a la memòria interna del Microprocessador

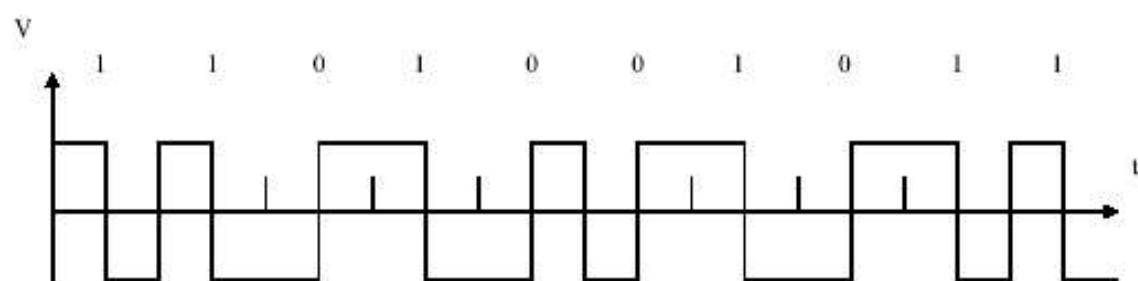
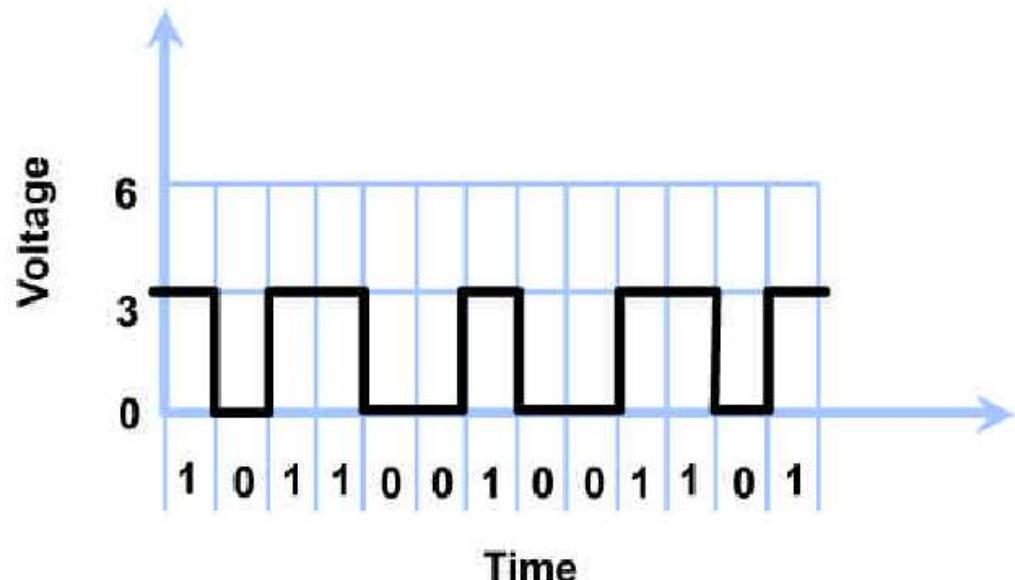
Codificació en Binari

Representació de la informació:

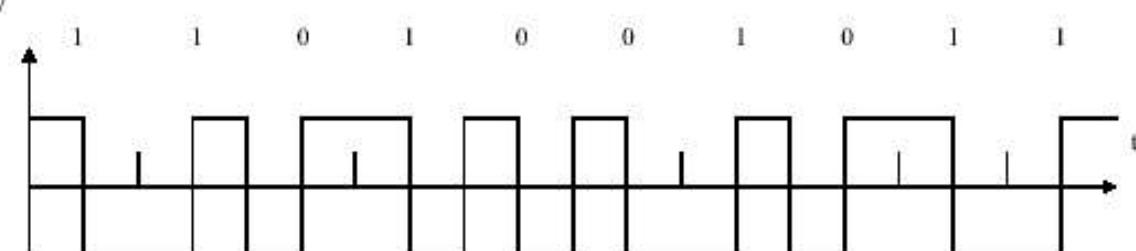
Representació de la informació típica...

...Però no única!!

Entre d'altres tenim per exemple...



Codificació Manchester



Codificació bifase diferencial

Codificació en Binari

Representem la informació. Ex: Codificació ASCII

Dec	Hx	Oct	Char		Dec	Hx	Oct	Html	Chr		Dec	Hx	Oct	Html	Chr		Dec	Hx	Oct	Html	Chr
0	0	000	NUL	(null)	32	20	040	#32;	Space		64	40	100	#64;	B		96	60	140	#96;	'
1	1	001	SOH	(start of heading)	33	21	041	#33;	!		65	41	101	#65;	A		97	61	141	#97;	s
2	2	002	STX	(start of text)	34	22	042	#34;	:		66	42	102	#66;	B		98	62	142	#98;	b
3	3	003	ETX	(end of text)	35	23	043	#35;	#		67	43	103	#67;	C		99	63	143	#99;	c
4	4	004	ETB	(end of transmission)	36	24	044	#36;	:		68	44	104	#68;	D		100	64	144	#100;	d
5	5	005	ENQ	(enquiry)	37	25	045	#37;	:		69	45	105	#69;	E		101	65	145	#101;	e
6	6	006	ACK	(acknowledge)	38	26	046	#38;	:		70	46	106	#70;	F		102	66	146	#102;	f
7	7	007	BEL	(bell)	39	27	047	#39;	:		71	47	107	#71;	G		103	67	147	#103;	g
8	8	010	BS	(backspace)	40	28	050	#40;	 		72	48	110	#72;	H		104	68	150	#104;	h
9	9	011	TAB	(horizontal tab)	41	29	051	#41;	:		73	49	111	#73;	I		105	69	151	#105;	i
10	A	012	LF	(NL line feed, new line)	42	2A	052	#42;	:		74	4A	112	#74;	J		106	6A	152	#106;	j
11	B	013	VT	(vertical tab)	43	2B	053	#43;	+		75	4B	113	#75;	K		107	6B	153	#107;	k
12	C	014	FF	(NP form feed, new page)	44	2C	054	#44;	-		76	4C	114	#76;	L		108	6C	154	#108;	l
13	D	015	CR	(carriage return)	45	2D	055	#45;	-		77	4D	115	#77;	M		109	6D	155	#109;	m
14	E	016	SO	(shift out)	46	2E	056	#46;	.		78	4E	116	#78;	N		110	6E	156	#110;	n
15	F	017	SI	(shift in)	47	2F	057	#47;	/		79	4F	117	#79;	O		111	6F	157	#111;	o
16	10	020	DLE	(data link escape)	48	30	060	#48;	0		80	50	120	#80;	P		112	70	160	#112;	p
17	11	021	DC1	(device control 1)	49	31	061	#49;	1		81	51	121	#81;	Q		113	71	161	#113;	q
18	12	022	DC2	(device control 2)	50	32	062	#50;	2		82	52	122	#82;	R		114	72	162	#114;	r
19	13	023	DC3	(device control 3)	51	33	063	#51;	3		83	53	123	#83;	S		115	73	163	#115;	s
20	14	024	DC4	(device control 4)	52	34	064	#52;	4		84	54	124	#84;	T		116	74	164	#116;	t
21	15	025	NAK	(negative acknowledge)	53	35	065	#53;	5		85	55	125	#85;	U		117	75	165	#117;	u
22	16	026	SYN	(synchronous idle)	54	36	066	#54;	6		86	56	126	#86;	V		118	76	166	#118;	v
23	17	027	ETB	(end of trans. block)	55	37	067	#55;	7		87	57	127	#87;	W		119	77	167	#119;	w
24	18	030	CAN	(cancel)	56	38	070	#56;	0		88	58	130	#88;	X		120	78	170	#120;	x
25	19	031	EM	(end of medium)	57	39	071	#57;	9		89	59	131	#89;	Y		121	79	171	#121;	y
26	1A	032	SUB	(substitute)	58	3A	072	#58;	:		90	5A	132	#90;	Z		122	7A	172	#122;	z
27	1B	033	ESC	(escape)	59	3B	073	#59;	;		91	5B	133	#91;	[123	7B	173	#123;	[
28	1C	034	FS	(file separator)	60	3C	074	#60;	<		92	5C	134	#92;	\		124	7C	174	#124;	\
29	1D	035	GS	(group separator)	61	3D	075	#61;	=		93	5D	135	#93;	:		125	7D	175	#125;	:
30	1E	036	RS	(record separator)	62	3E	076	#62;	>		94	5E	136	#94;	^		126	7E	176	#126;	^
31	1F	037	US	(unit separator)	63	3F	077	#63;	,		95	5F	137	#95;	;		127	7F	177	#127;	;

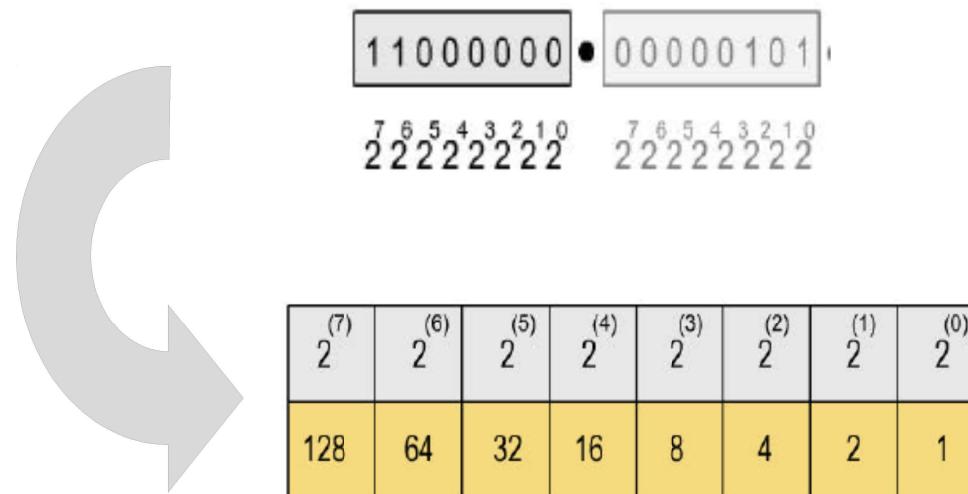
Source: www.LookupTables.com

128	Ҫ	144	Ӯ	160	Ӯ	176	Ԭ	192	Ӷ	208	Ӯ	224	Ӯ	240
129	Ӱ	145	Ӯ	161	Ӯ	177	Ԭ	193	ӷ	209	Ӯ	225	Ӯ	241
130	Ӳ	146	Ӯ	162	Ӳ	178	Ԭ	194	ӷ	210	Ӯ	226	Ӯ	242
131	Ӵ	147	Ӯ	163	Ӵ	179	Ӆ	195	ӷ	211	Ӯ	227	Ӯ	243
132	ӵ	148	Ӯ	164	ӵ	180	ӷ	196	ӷ	212	Ӯ	228	Ӯ	244
133	Ӷ	149	Ӯ	165	Ӷ	181	ӷ	197	ӷ	213	Ӯ	229	Ӯ	245
134	Ӹ	150	Ӯ	166	Ӹ	182	ӷ	198	ӷ	214	Ӯ	230	Ӯ	246
135	ӹ	151	Ӯ	167	ӹ	183	ӷ	199	ӷ	215	Ӯ	231	Ӯ	247
136	ӻ	152	Ӯ	168	ӻ	184	ӷ	200	ӷ	216	ӷ	232	ӷ	248
137	Ӽ	153	Ӯ	169	Ӽ	185	ӷ	201	ӷ	217	ӷ	233	ӷ	249
138	ӽ	154	Ӯ	170	ӽ	186	ӷ	202	ӷ	218	ӷ	234	ӷ	250
139	Ӿ	155	Ӯ	171	Ӿ	187	ӷ	203	ӷ	219	ӷ	235	ӷ	251
140	ӿ	156	Ӯ	172	ӿ	188	ӷ	204	ӷ	220	ӷ	236	ӷ	252
141	ӱ	157	Ӯ	173	ӱ	189	ӷ	205	ӷ	221	ӷ	237	ӷ	253
142	Ӳ	158	Ӯ	174	Ӳ	190	ӷ	206	ӷ	222	ӷ	238	ӷ	254
143	ӳ	159	Ӯ	175	ӳ	191	ӷ	207	ӷ	223	ӷ	239	ӷ	255

Source: www.LookupTables.com

Codificació en Binari

Normalment estem acostumats a fer servir codificació decimal. Podem passar fàcilment de decimal a binari de la següent forma:



L'agrupació de bits en bytes fa que els nombres binaris i els nombres hexadecimals tinguin una relació directa: així 11001000 és pot expressar com:

<table border="1"><tr><td>1100</td><td>1000</td></tr></table>	1100	1000	}	0xC8	nomenclatura llenguatge C
1100	1000				
0x C 8	C8h	nomenclatura ASM			

Codificació en Binari

Per operar els conjunts de bits utilitzem l'àlgebra de Boole

ÀLGEBRA DE BOOLE

Sistema matemàtic consistent de B elements amb cardinalitat > 2 i 2 operacions matemàtiques que denominarem \cdot i $+$ que acompleix una sèrie de postulats.

- Si $x, y \in B$
 - (a) $x + y \in B$
 - (b) $x \cdot y \in B$
- $x + 0 = x$
- $x \cdot 1 = 1 \cdot x = x$
- $\forall x, y \in B$
 - (a) $x+y = y+x$
 - (b) $x \cdot y = y \cdot x$

- $\forall x, y, z \in B$
 - (a) $x \cdot (y+z) = x \cdot y + x \cdot z$
 - (b) $x+(y \cdot z) = (x+y) \cdot (x+z)$
- $\forall x \in B \exists x' \in B$
 - (a) $x + x' = 1$
 - (b) $x \cdot x' = 0$

Codificació en Binari

Funcions lògiques:
AND, OR, Inversor

x	y	$x+y$ (operación OR)	$x \cdot y$ (operación AND)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	1	1

x	x'
0	1
1	0

Sense oblidar-nos de...

Nomenclatura de programació

!= not
^= xor
&= and
|= or

NAND: AND complementada

NOR: OR complementada

XOR y NXOR:

x	y	$x \oplus y$
00	0	0
01	1	1
10	1	1
11	0	0

x	y	$(x \oplus y)'$
00	0	1
01	1	0
10	1	0
11	0	1

Codificació en Binari

- Representació dels números en binari
 - Binari sense signe
 - Binari amb signe
 - Complement a 1
 - Complement a 2 (mètode utilitzat pels ordinadors)

Binari sense signe

Representació numèrica de dígits. Traducció directa de decimal a binari

0 → 0 1 → 1 2 → 10 3 → 11 4 → 100
5 → 101 6 → 110

Codificació en Binari

- Binari amb signe
 - Es reserva el bit més significatiu per guardar el signe
 - Bit més significatiu a 1 implica nº negatiu
 - Bit més significatiu a 0 implica nº positiu
 - L'operació no és immediata
 - Ex: $7 - 3$

$$0111 = 7$$

$$1011 = -3$$

$$0111 + 1011 = 0010 + \text{bit de carry}$$

Codificació en Binari

- Què és el CARRY??
 - El CARRY és un bit que s'activa quan la operació entre dos nº's binaris sobrepassa l'espai (en bits) reservat per aquesta operació
 - Ex:
 $11001111 + 10000011 = 01010010 + \text{bit de carry activat}$

Codificació en Binari

- Complement a 1
 - El complement a 1 d'un determinat n° s'obté canviant els díigits del n° pel seu complementari. ie. On hi ha un 1 posem un 0 i on hi ha un 0 posem un 1
 - Matemàticament s'expressa com
$$Ca1(Nº) = 2^{n-1} + (2^{n-1} - 1) - Nº$$
 - La representació del Ca1 d'un n° depen del n° de bits amb el que representem aquest n°.
$$Ca1(6) \text{ amb 6 bits} = 111001; (2^{6-1}) + (2^{6-1} - 1) - 6 = 57d$$
$$Ca1(6) \text{ amb 8 bits} = 11111001$$

Codificació en Binari

- Operacions en Ca1

- Es fa la suma dels dos nº's i al resultat se li suma el carry si hi ha.

Ex:

$$9 - 4 = 5$$

$$00001001 - \text{Ca1}(00000100) = 00001001 + 11111011$$

00001001

11111011

$$100000100 + \text{el bit de CARRY} \Rightarrow 00000101$$

bit de CARRY

Codificació en Binari

- Complement a 2
 - El complement a 2 d'un determinat nº es calcula fent primer el Ca1 i després sumant-li 1
 - El complement a 2 d'un determinat nº format per n dígits binaris és
$$Ca2(N) = 2^n - N$$
 - Exemple: Calculem el complement a 2 de 12 amb 8 bits
$$12_d = 00001100_b; Ca2(12) = 2^8 - 12 = 244 = 11110100$$

Codificació en Binari

- Operacions en Ca2
 - Es fa la suma dels dos nº's i el carry no es té en consideració

Exemple

$$9 - 4 = 5$$

$$\begin{aligned}9 - 4 &= 00001001 + \text{Ca2}(00000100) = \\&= 00001001 + 11111100 = \cancel{100000101}\end{aligned}$$

Operacions amb nombres reals

- Segueix l'especificació ANSI/IEEE 754
- Considera dos nivells de precisió:
 - Simple (32 bits)
 - Doble (64 bits)
- Es divideix en Signe, Exponent i Mantissa
- $N = (-1)^S \cdot 2^e \cdot M$ on $e = \text{Exponent} - 127$
- En precisió simple Exponent = 8 bits, Mantissa = 23 bits

Nivells

Codi Màquina

- Conjunt d'instruccions fonamentals que la màquina pot executar
- Expressada en un conjunt de 0's i 1's

Llenguatge Ensamblador

- Equivalents alfanumèrics del llenguatge màquina
- Mnemònics més intel·ligibles

Ensamblador

- Programa que tradueix (un-a-un) el llenguatge ensamblador a codi màquina
- El llenguatge "nadiu" del computador és el codi màquina

MC68000 Assembly Language	Machine Language
MOVE.W D4, D5	0011 101 000 000 100
ADDI.W #9, D2	00000110 01 000 010 0000 0000 0000 1001

Op code Data reg. #5 Data reg. #4

Two Motorola MC68000 Instructions

Nivells

Format de les **instruccions** en ensamblador: S'ajusta a les característiques de la màquina.

Exemple

El format d'instruccions d'un processador consta de **3 camps**. El primer correspon al **OP code**, el segon al operand font i el tercer a l'altre operand font, que, a més es fa servir per depositar el resultat de l'operació

El repertori d'instruccions del procesador és de 50

Si els operands es determinen donant les direccions que ocupen en la memòria principal i aquesta pot tenir un mida màxima de 64KBytes, quin serà el format de la instrucció i la longitud dels seus camps en bits.

Nivells

Resposta

El Op. Code serà de 6 bits ja que amb aquests podem arribar a tenir un nº de 64 combinacions, que és superior a les 50 necessàries.

El camp dels dos operands serà de 16 bits cada un, ja que per adreçar una memòria de 64KBytes precisem de 16 bits ($2^{16} = 65.536 = 64 \text{ KB}$)

Per tant el format de la instrucció serà de 38 bits

Hi ha altres possibilitats, que es veuen al capítol 3

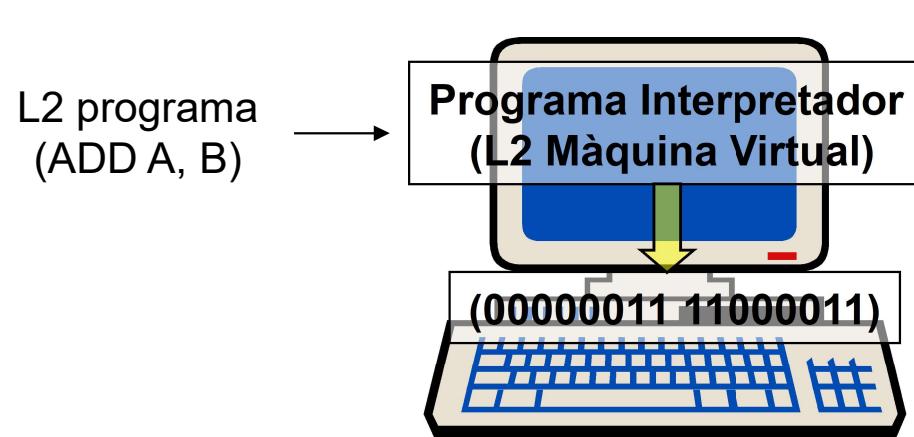
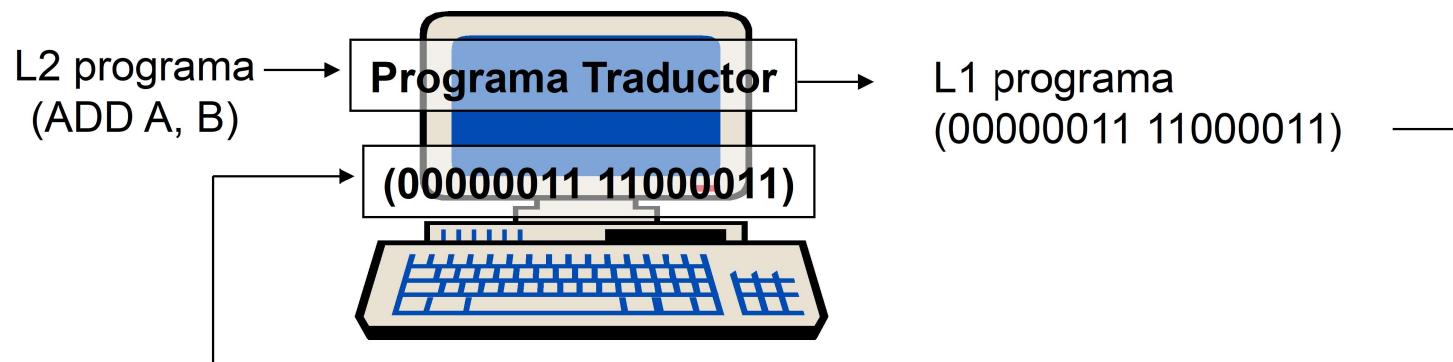
Traducció - Interpretació

Instrucció L1 :

00000011 11000011

Instrucció Equivalent L2 :

ADD A, B

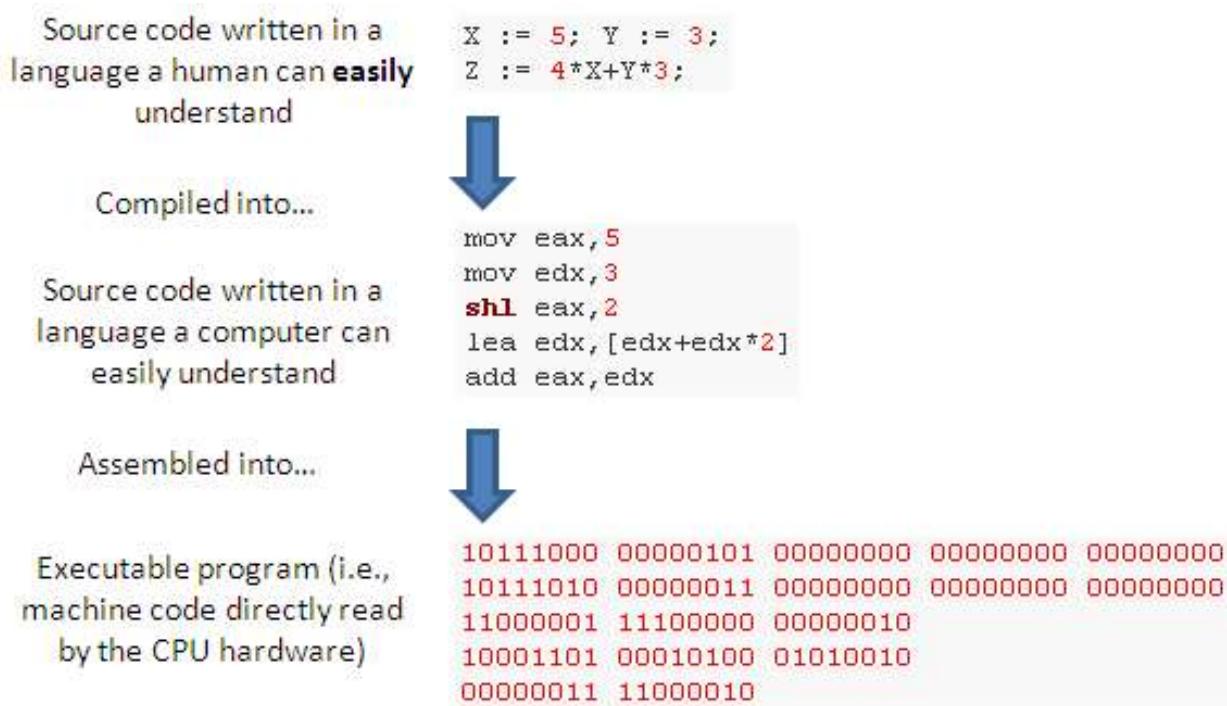


Nivell Sistema Operatiu

- Un Sistema Operatiu és un *programa* que proporciona un entorn en el que l'usuari pot executar altres programes de manera convenient i eficient.
- És responsable de controlar els recursos del sistema (en temps i espai), assignant recursos als programes d'usuari i monitoritzant la integritat del sistema
- Proporciona serveis a programes per facilitar la tasca del programador. Aquests serveis poden ser invocats per el programa d'usuari a través de crides al sistema (*system calls*)
- La majoria del codi generat per nivells superiors és passat directament al nivell llenguatge màquina

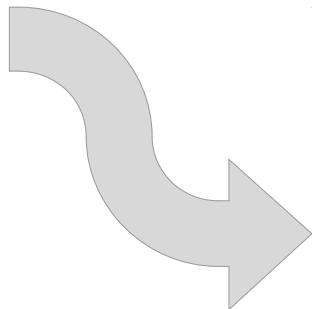
Llenguatges Alt-Nivell

- Llenguatges Compilats: Pascal, C, C++, Fortran, Java
- Llenguatges Interpretats: Perl, Python, JavaScript, Java
- Amaguen al programador els detalls arquitecturals de baix nivell
- Els llenguatges d'alt nivell són generalment independents de la màquina



Llenguatges Alt-Nivell

```
for (int i=0;i<N_MAX;i++){  
    double b = i+CONSTANT;  
    ...  
}
```



```
SUB R1,R1,R1  
SUB R2,R2,R2  
SUB R3,R3,R3  
SUB R4,R4,R4  
ADDI R4,CONSTANT,R4  
ADDI R1, N_MAX, R1  
loop:  
    ADD R4,R2,R2  
    ADD R2,R3,R2  
    ADDI R3,1,R3  
    SUBI R1,1,R1  
    BR loop
```

Estructura i Funció

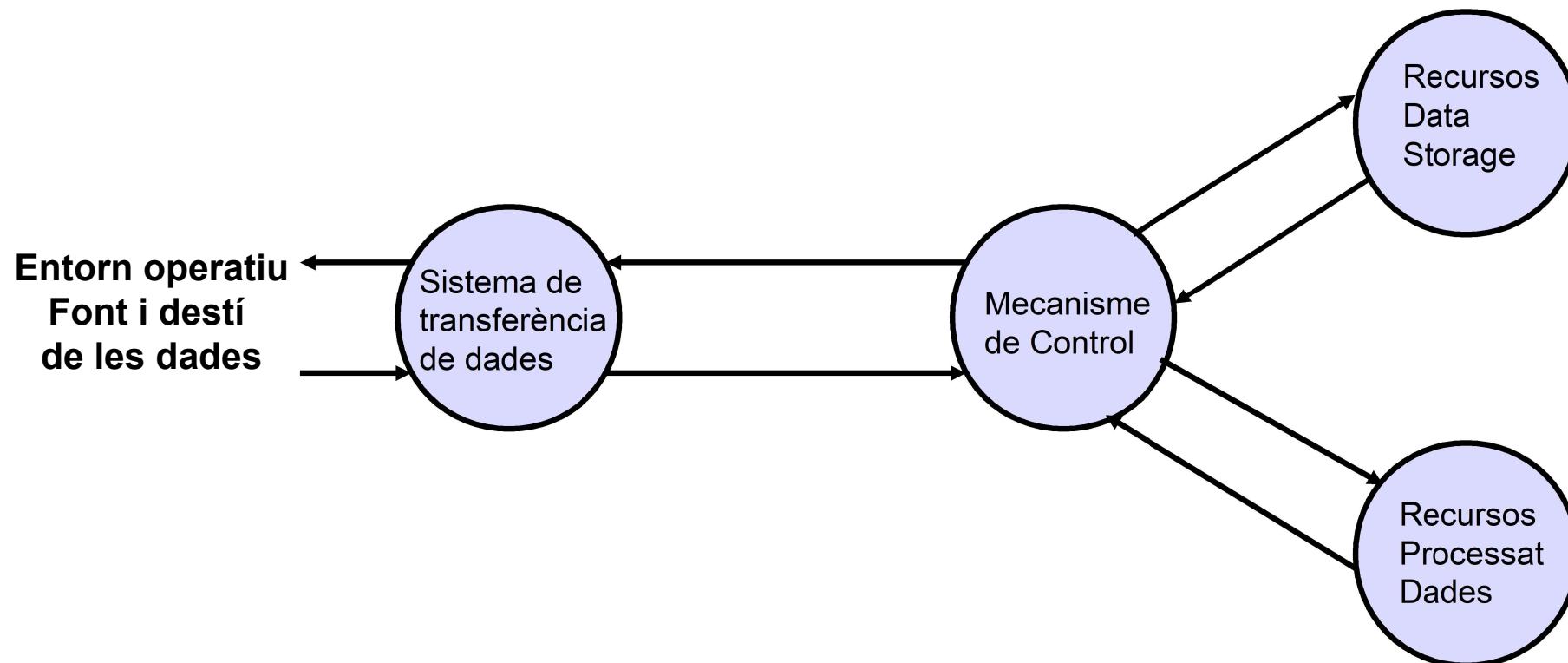
- A cada nivell el dissenyador ha de considerar
 - **Estructura:** La manera amb que es relacionen uns components amb els altres.
 - **Funció:** L'operació dels components individuals com a part de l'estructura

Funció

- Les funcions d'un computador són:
 - Processat de dades
 - Emmagatzematge de dades
 - Moviment de dades
 - Control

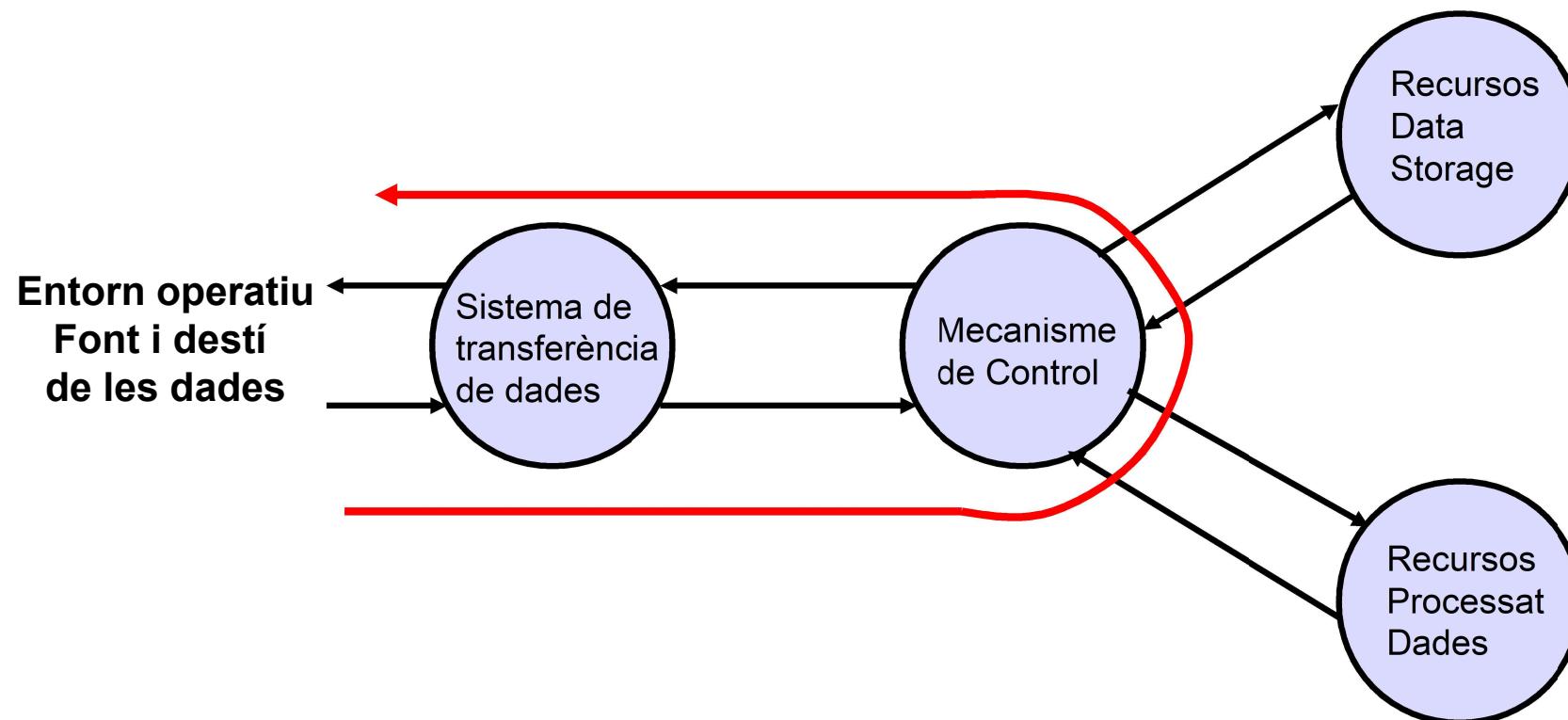
Visió Funcional

- Visió funcional d'un computador



Operacions (1)

- Moviment o transferència de dades
 - P.ex. Teclat a pantalla



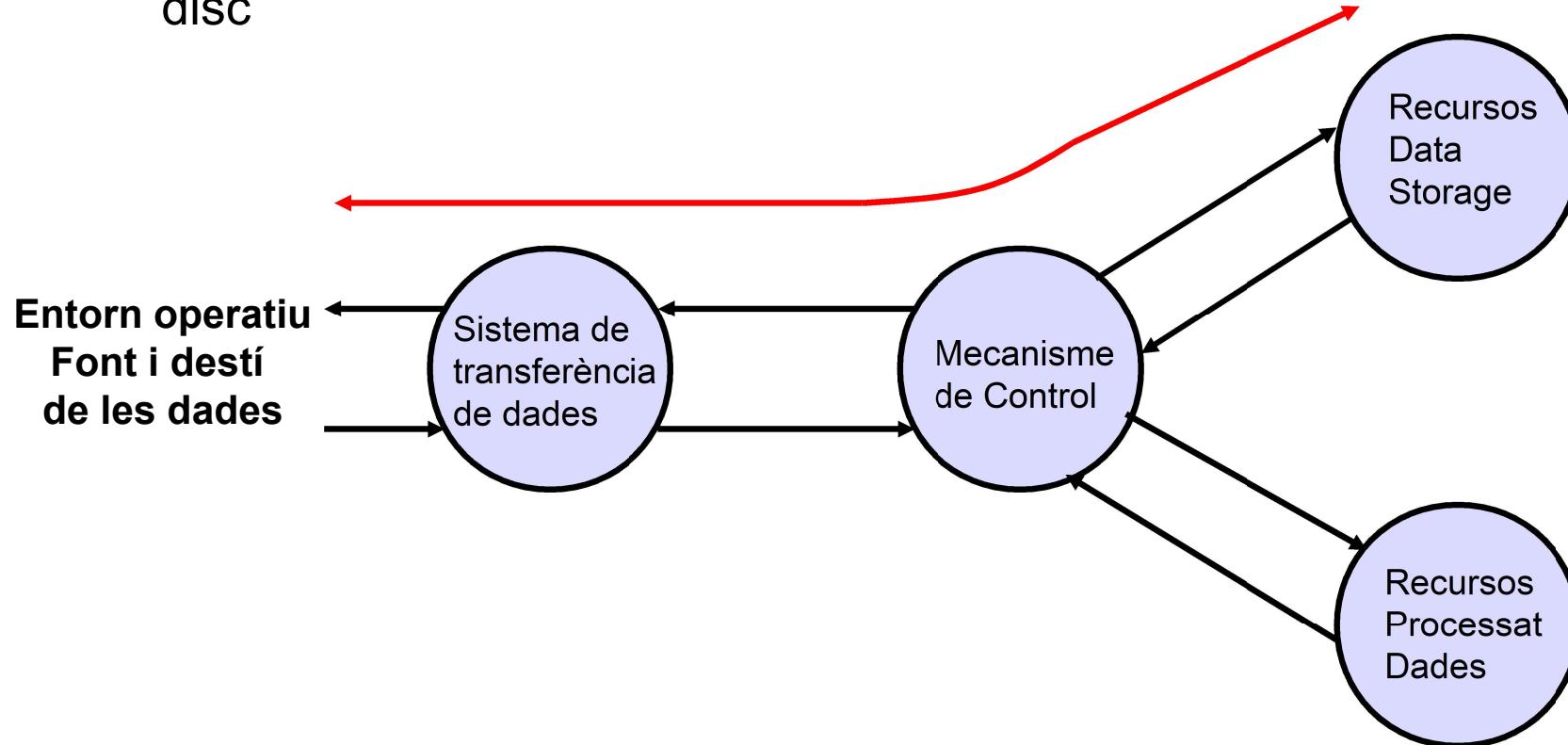
Moviment o transferència de dades

- Exemple de moviment de dades
 - 1) De teclat a memòria i de memòria a pantalla
(Llenguatge ensamblador i8085)
 - IN 04h (captura de dades DES DE l'adreça 04h)
 - OUT 05h (sortida de dades CAP A l'adreça 05h)

Operacions (2)

- Emmagatzematge (Data Storage)

- P.ex. Dades transferides des d'un entorn extern (internet) a disc

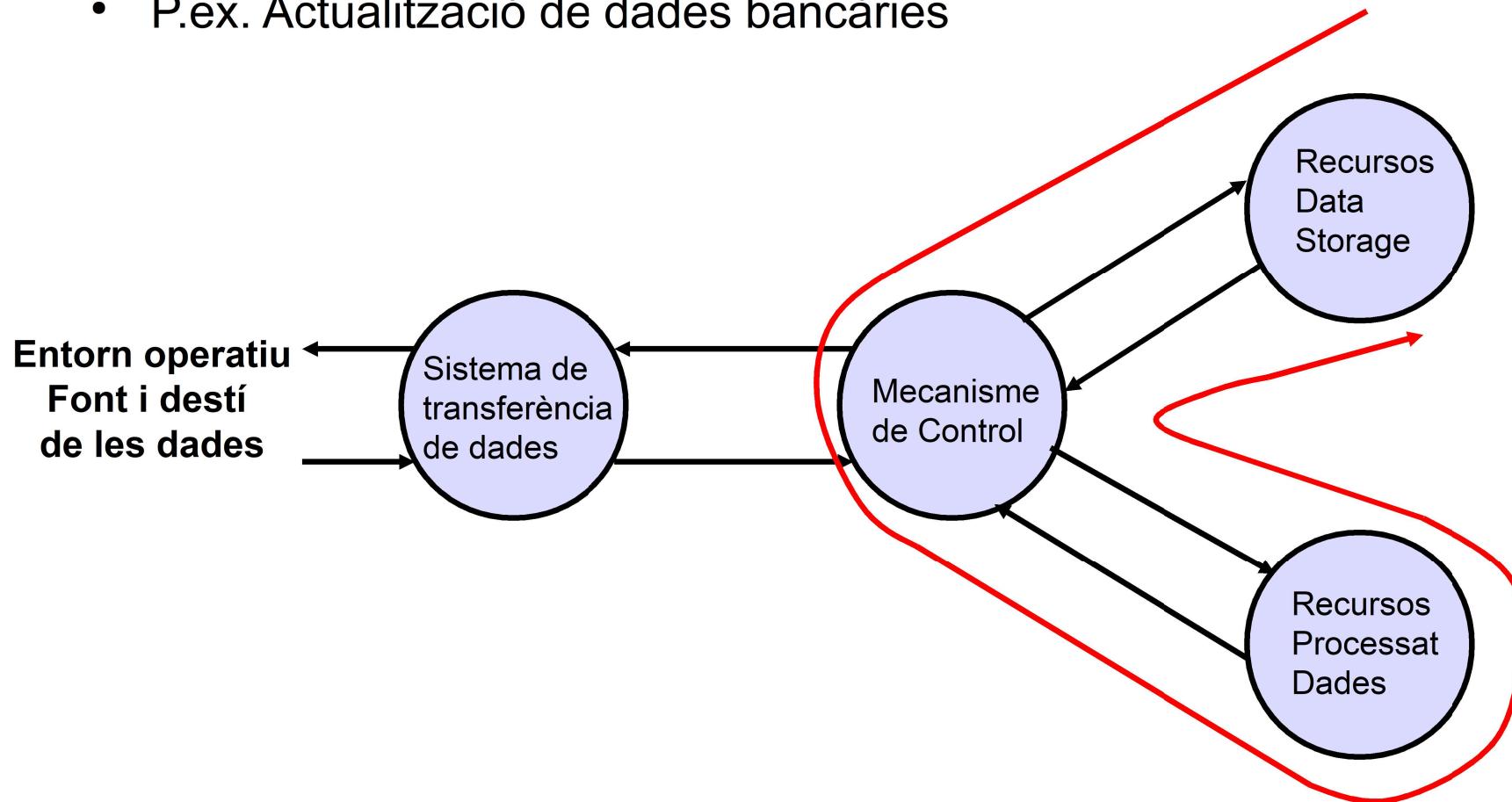


Emmagatzematge (Data Storage)

- 1) De memòria a CPU
 - LOAD offset(R1), R2
- 2) De CPU a memòria
 - STORE R1, offset(R2)

Operacions (3)

- Processat des de/a emmagatzematge
 - P.ex. Actualització de dades bancàries



Processat i emmagatzematge

- Exemple

LOAD 3(R0), R1

LOAD 4(R0), R2

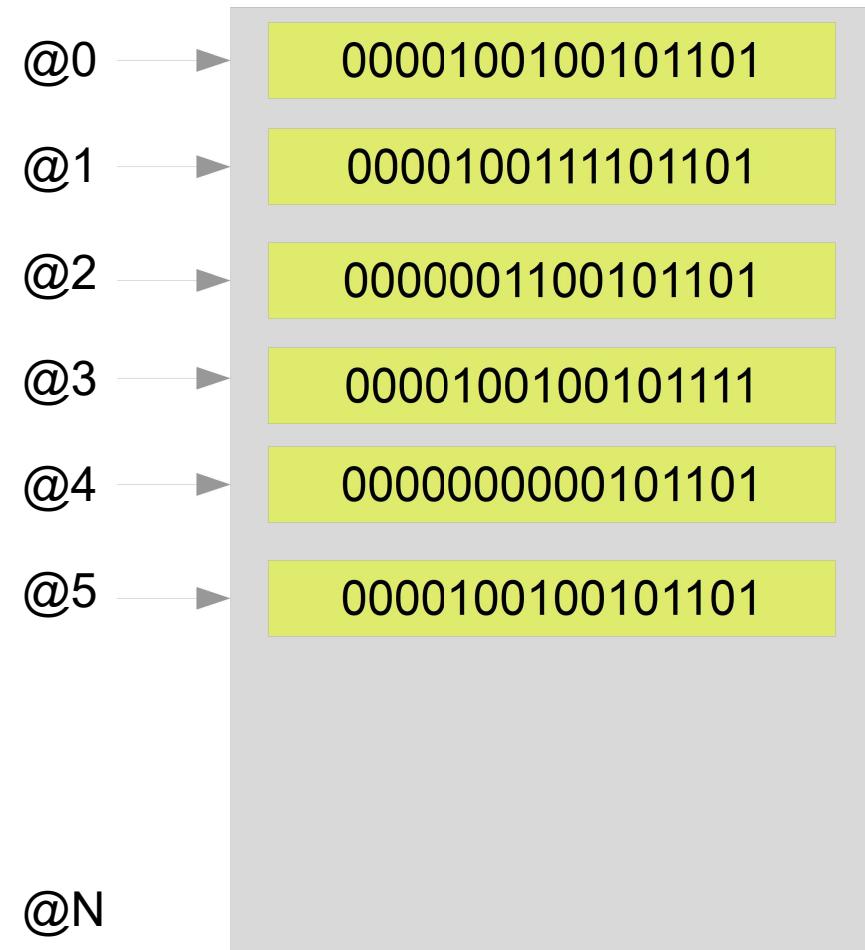
SUB R3,R3,R3

loop: ADD R1,R3,R3

SUBI R2, 1, R2

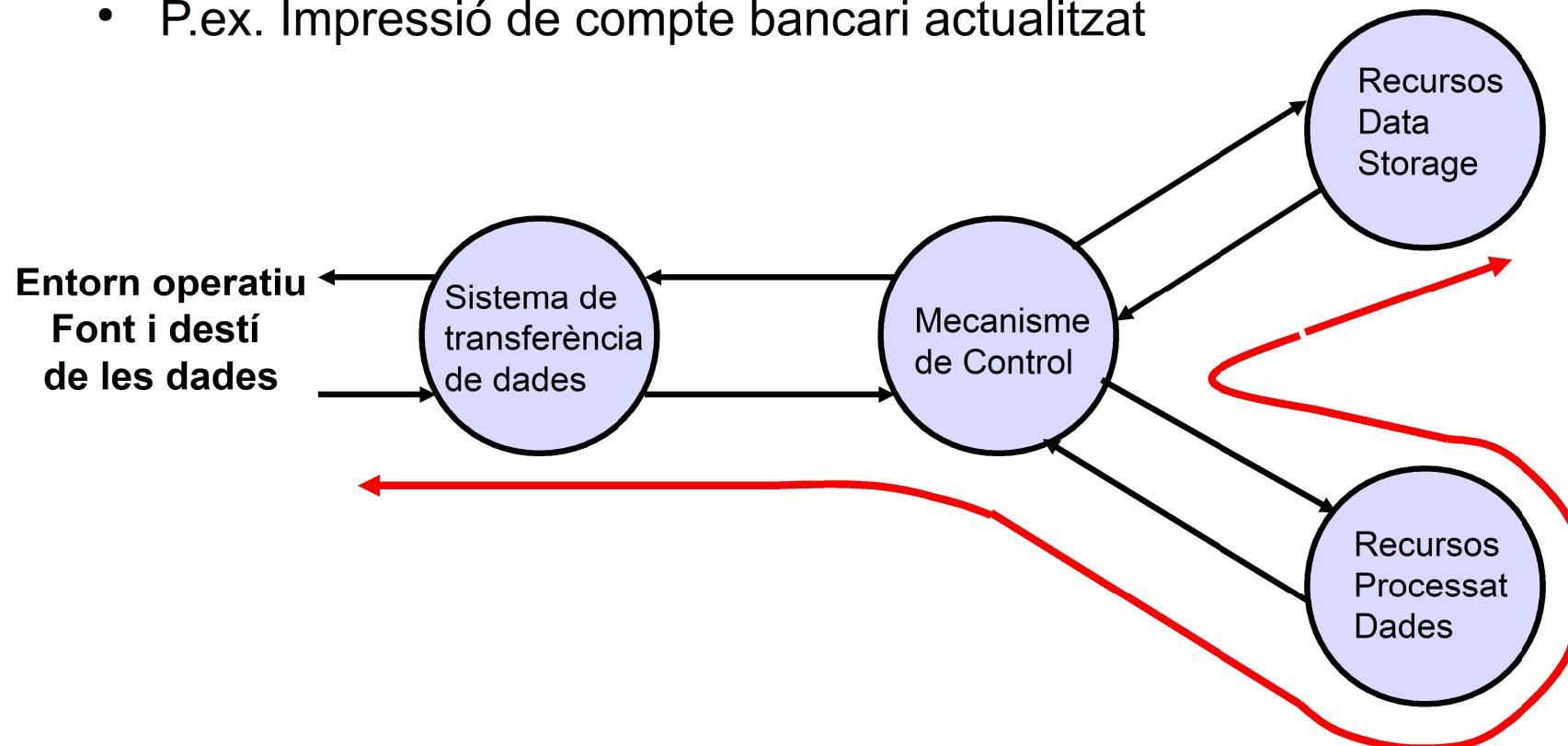
BG loop

STORE R3, 5(R0)



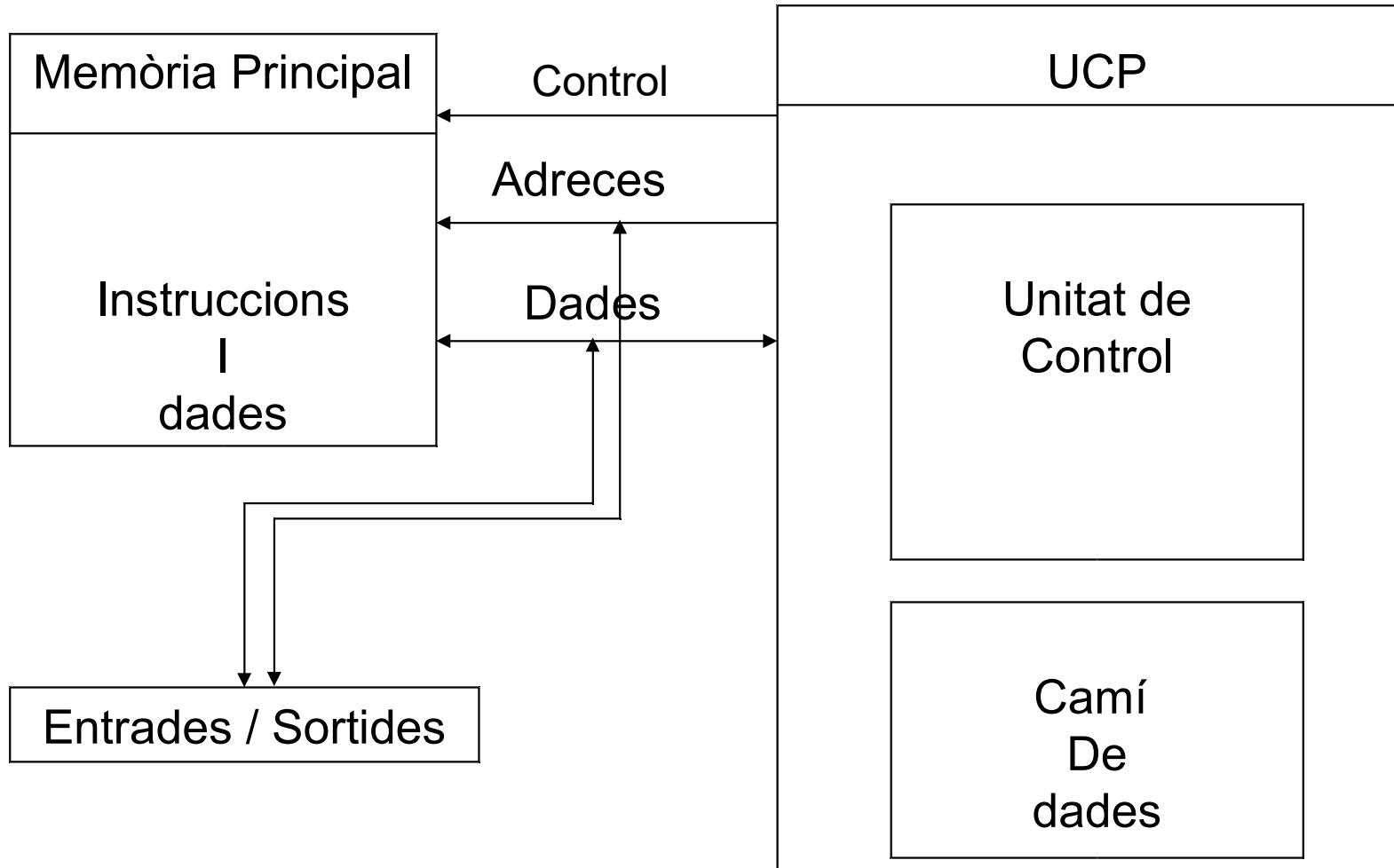
Operacions (4)

- Processat des d'emmagatzematge a Entrada/Sortida
 - P.ex. Impressió de compte bancari actualitzat



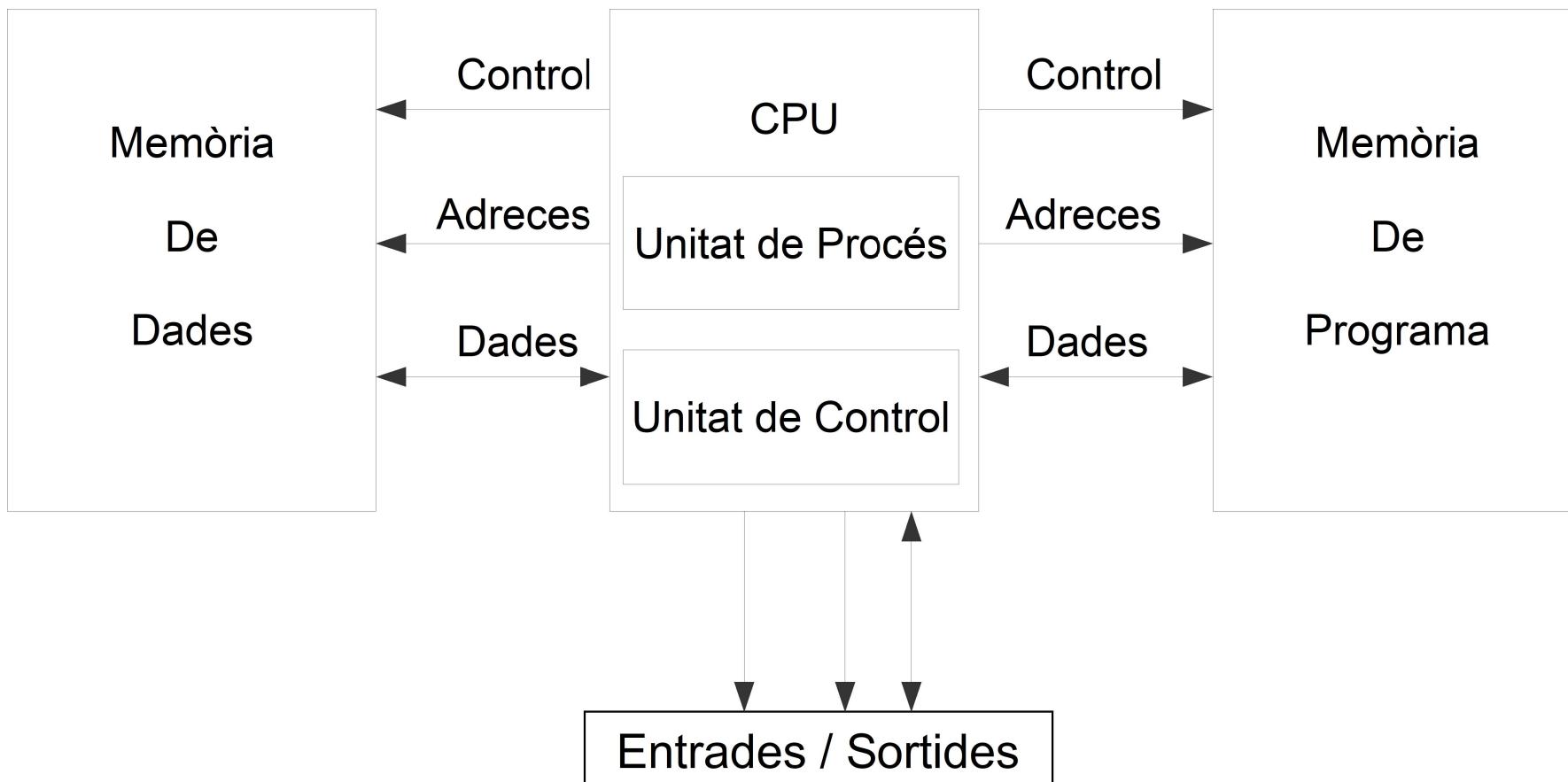
Nivells

Estructura simplificada del computador proposada per Von Neumann



Nivells

Estructura Harvard. Usualment utilitzada en microcontrol·ladors i sistemes empotrats



Estructura

- Principals Components d'un Computador

- **Unitat Central de Procés (CPU)**

Controla l'operació del computador I fa el processat de dades

- **Memòria primària**

Guarda dades

- **Entrada / Sortida E/S**

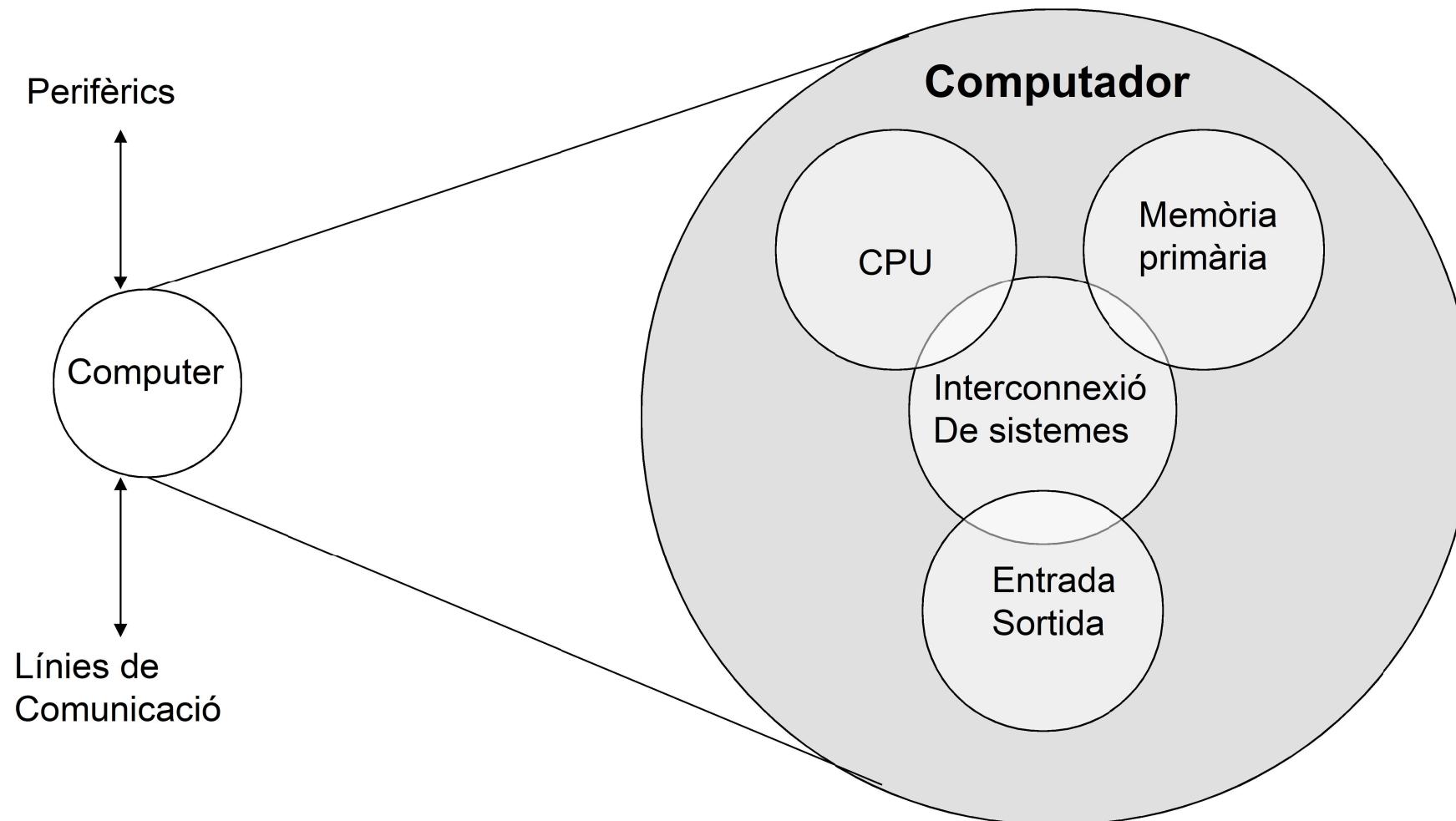
Mou dades entre el computador I l'entorn extern

- **Sistema de connexió (BUS)**

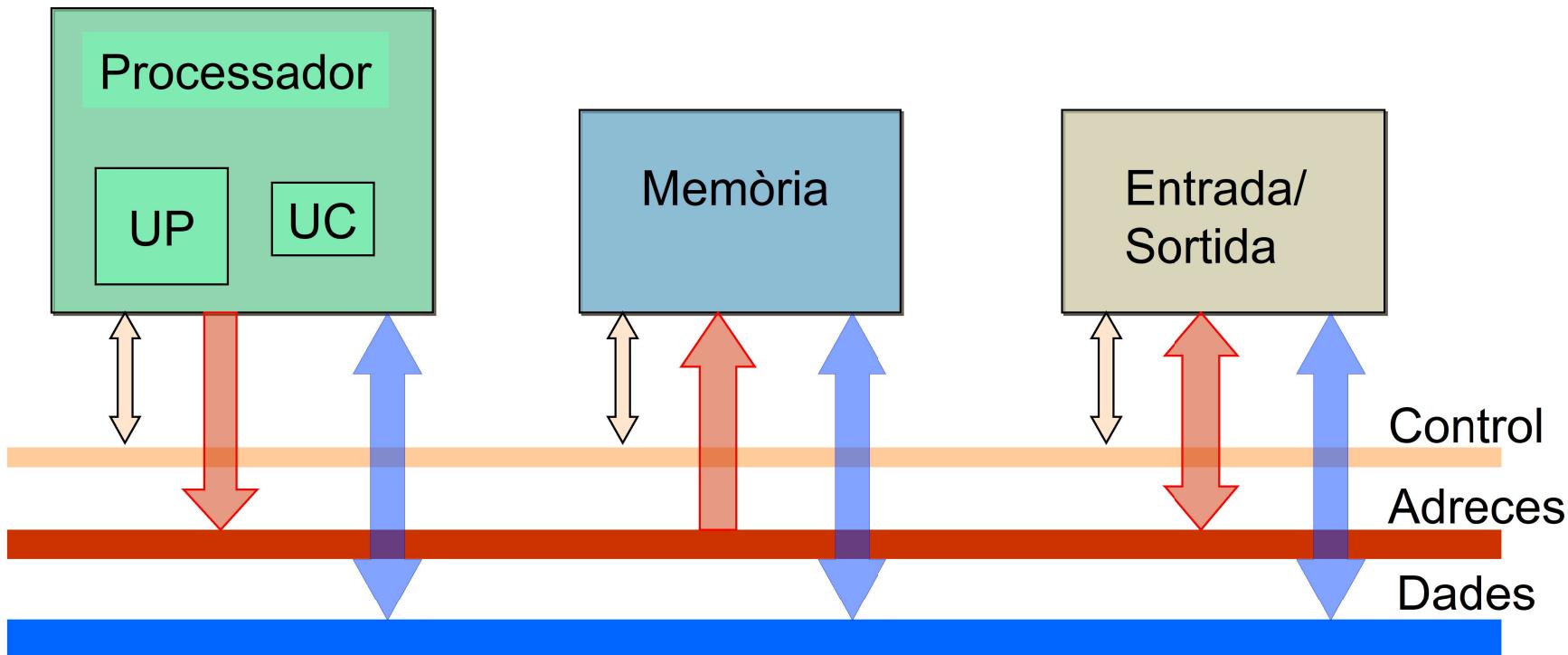
Mecanisme que proporciona comunicacions entre els components del sistema,

BUS= conjunt de línies agrupades que condueixen senyals relacionats

Estructura – Nivell superior



BUS Genèric de sistema



Bus de sistema = constituït per 3 busos:

Dades

Adreces

Control

BUS Genèric de sistema

Bus de sistema:

-Constituït per 50 – 100 línies. Cada una amb la seva funció particular.

- **Bus de dades:** Proporcionen un camí per transmetre dades entre els mòduls del sistema. Normalment consta de 8, 16 o 32 línies diferents (amplada del bus). Això determina el nº de bits a transmetre en un interval de temps.

1.- L'amplada del bus és clau per determinar les prestacions del conjunt del sistema

Ex: Amplada de bus de 8 bits, instruccions de 16 bits

BUS Genèric de sistema

-Bus d'Adreces: S'utilitza per designar la font o el destí de les dades.
P.E. Si la CPU desitja llegir una paraula (8, 16, o 32 bits) de dades de la memòria, posarà la direcció de la paraula al bus de direcció.

- i) L'amplada del bus d'adreces determina la màxima capacitat de memòria possible del sistema.
- ii) Les línies d'adreces també s'utilitzen per adreçar els ports d'E/S.
- iii) Els bits d'ordre més alt s'utilitzen per seleccionar una posició de memòria o un port E/S dintre del mòdul.

Ex: BUS de 8bits. La direcció 01111111 o inferiors fan referència a posicions dintre d'un mòdul de memòria. La direcció 10000000 i superiors designen dispositius connectats a un mòdul E/S,

BUS Genèric de sistema

-Bus de control. S'utilitza per control.lar l'accés i l'ús de les línies de dades i de adreces. Això es fa ja que les línies de dades i d' adreces són compartides per tots els components, i per tant ha d'existir una forma de control.lar el seu ús.

- i) Transmissió d'ordres: Especifiquen les operacions a realitzar
- ii) Transmissió d'informació de temporització entre mòduls del sistema, indicant la validesa de les dades i les adreces.

Ex. Línies de control típiques:

- . Escriptura en Memòria
- . Escriptura de E/S
- . Lectura de E/S
- . Petició de Bus
- . Cesió de Bus
-

BUS Genèric de sistema

Funcionament del BUS:

Si un mòdul desitja enviar una dada a un altre mòdul farà dues coses:

- 1.- Obtenir l'ús del bus
- 2.- Transferir la dada a través del bus

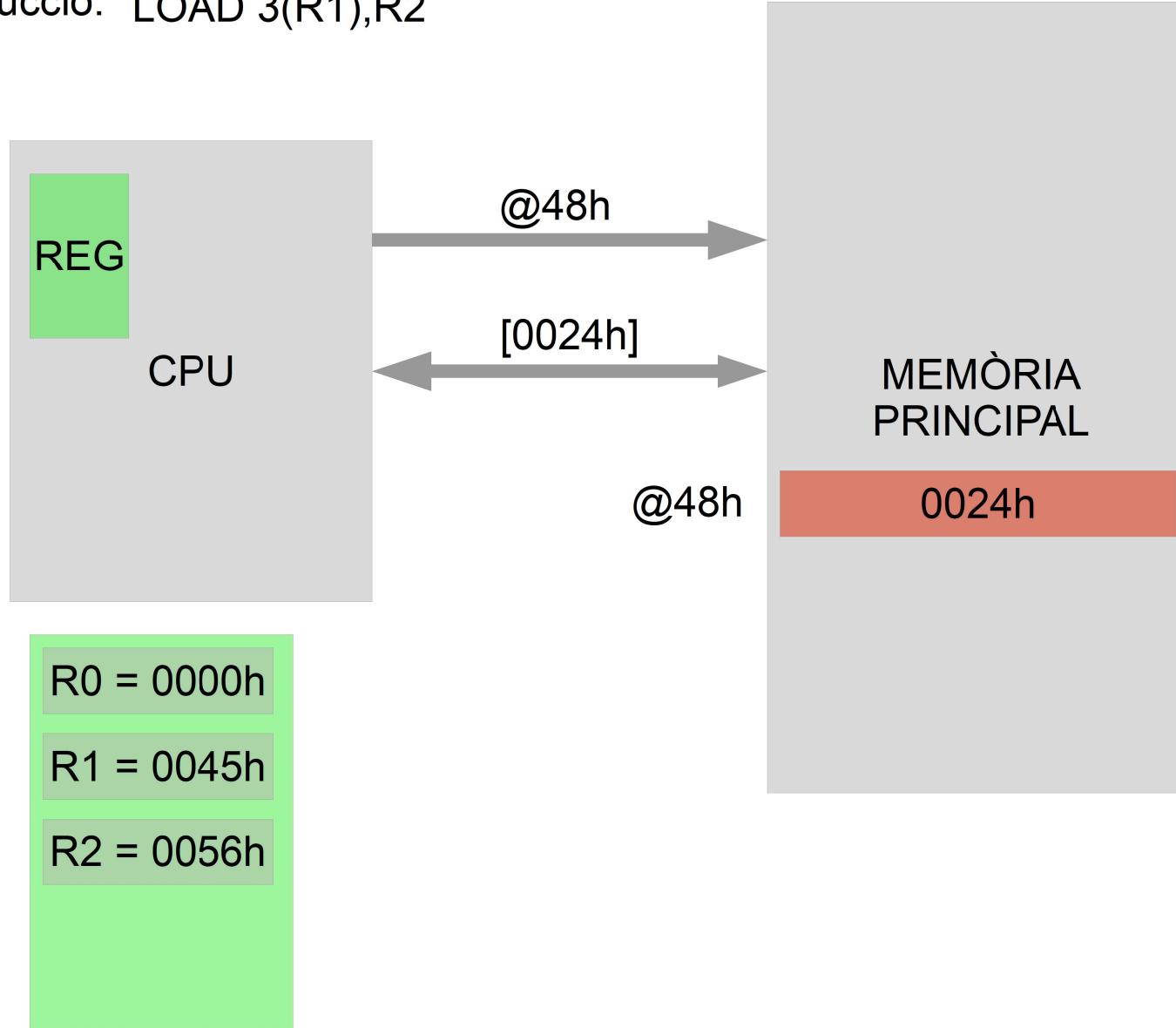
Si un mòdul desitja demanar una dada un altre mòdul haurà de:

- 1.- Obtenir l'ús del bus
- 2.- Transferir la petició a l'altre mòdul mitjançant les línies de control i adreces apropiades

BUS Genèric de sistema

Exemple d'ús dels busos:

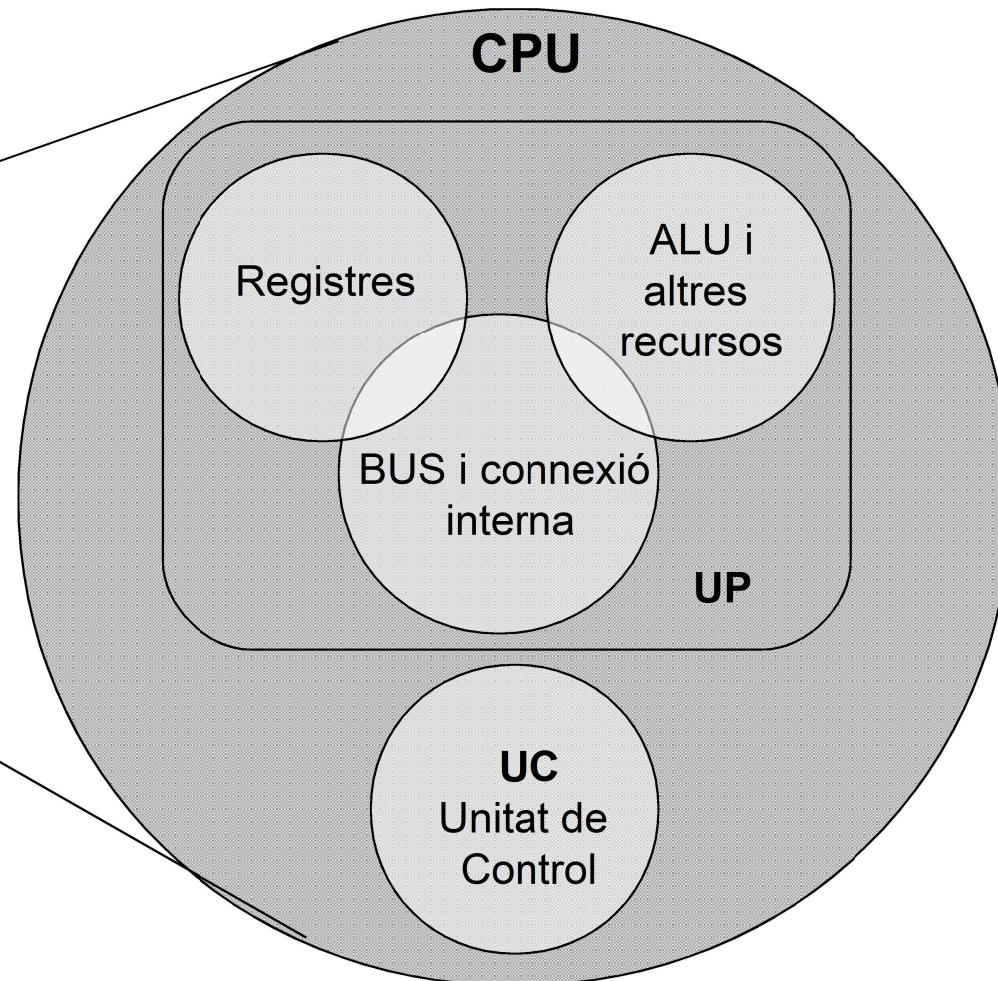
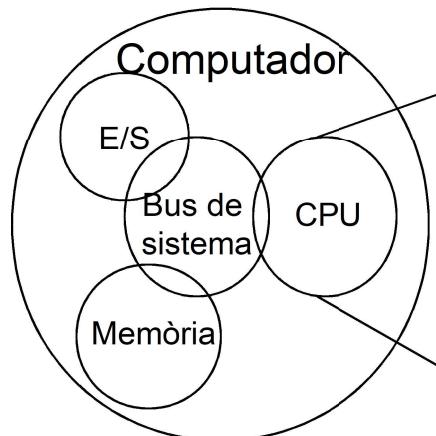
Instrucció: LOAD 3(R1),R2



Estructura - La CPU

Principals components de la CPU

- UNITAT DE CONTROL (UC)
- UNITAT DE PROCÉS (UP)



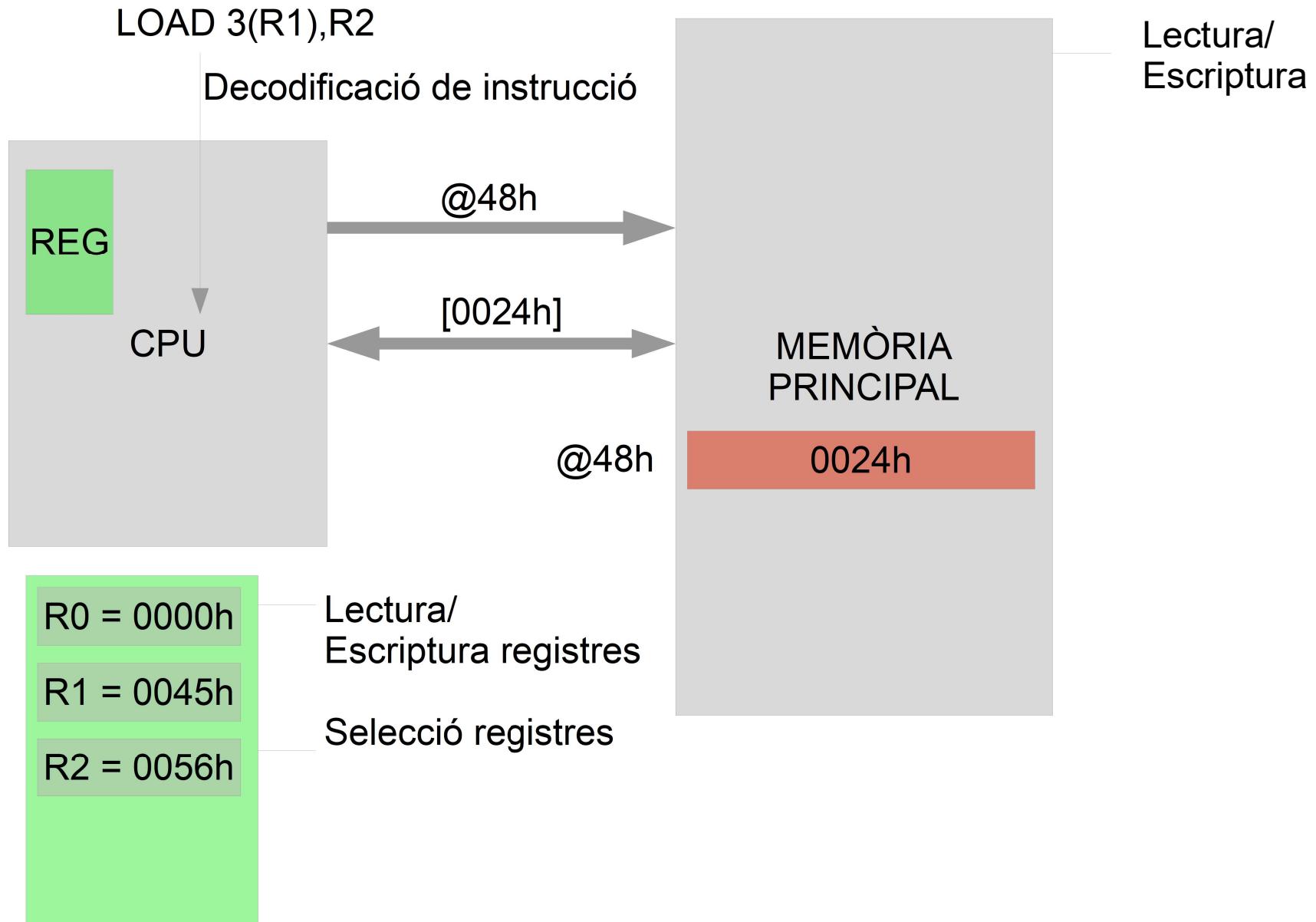
Estructura - Unitat de procés (UP)

- ALU i altres recursos:
 - Unitat aritmètico-lògica
 - Basada en sumador
 - Suma, resta, complementació, $\times 2$, $/2$
 - AND, OR, XOR,.. (bit-a-bit)
 - Multiplicador, Desplaçador, LUT
- REGISTRES
 - Propòsit específic:
 - IR, PC, SP, Status, AR, DR, AC
 - Propòsit general
 - Conjunt de registres (dual port)
- BUS i connexions internes
 - DADES; ADRECES; CONTROL
 - Diferents estructures de BUS dades

Estructura – Unitat de Control (UC)

- Unitat de Control cablejada (*hardwired*)
 - Màquina d'estats fixa.
Diferents estratègies de disseny
- Unitat de Control Microprogramada
 - La unitat de control és en si un petit computador.
 - Una instrucció del processador és implementada amb un microprograma amb determinat nombre de microinstruccions
 - Lògica seqüencial – Control de l'ordre dels events
 - Microprograma
 - Memòria del microprograma

Estructura busos. Accés MP-CPU



Estructura busos. Accés MP-CPU

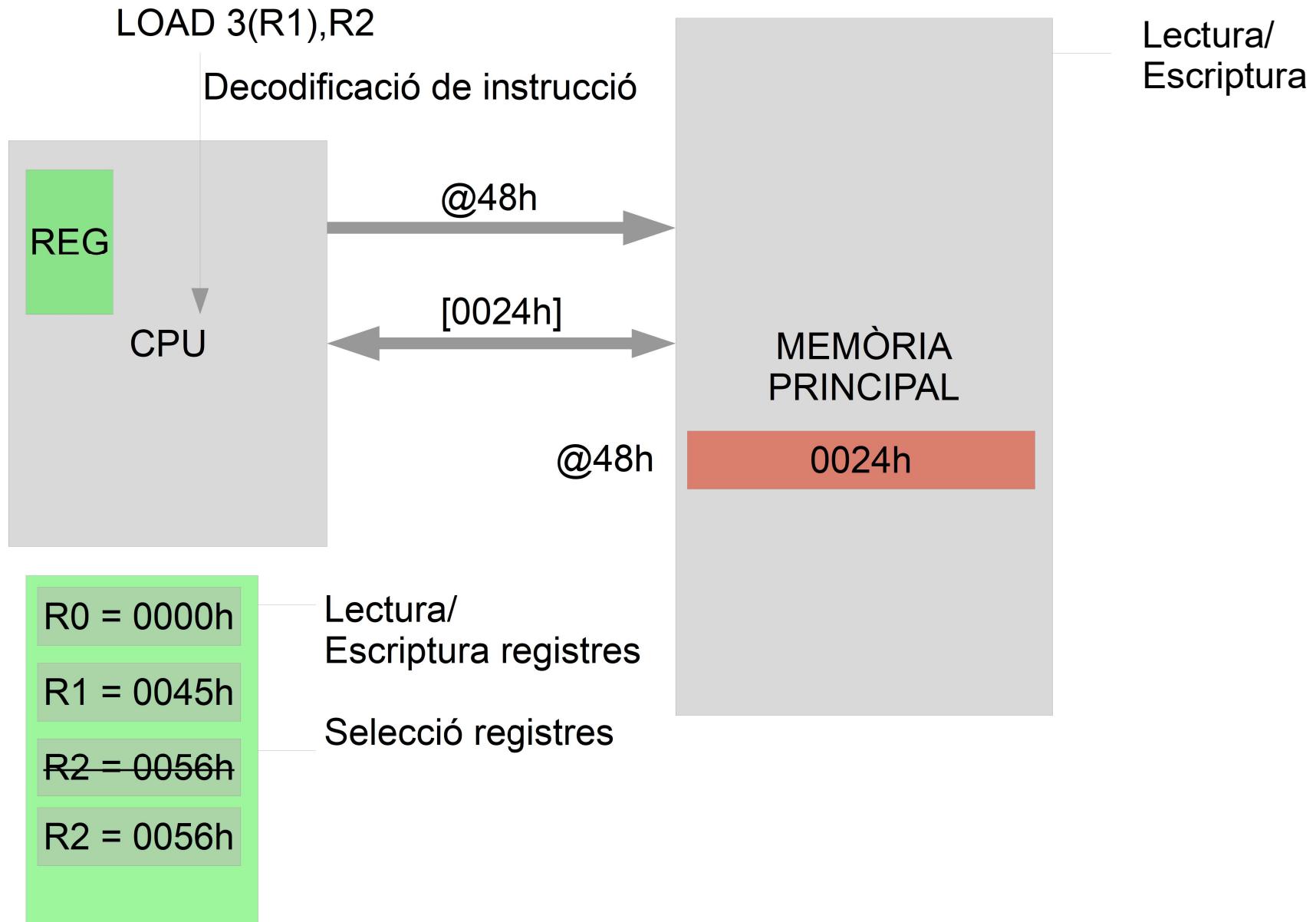
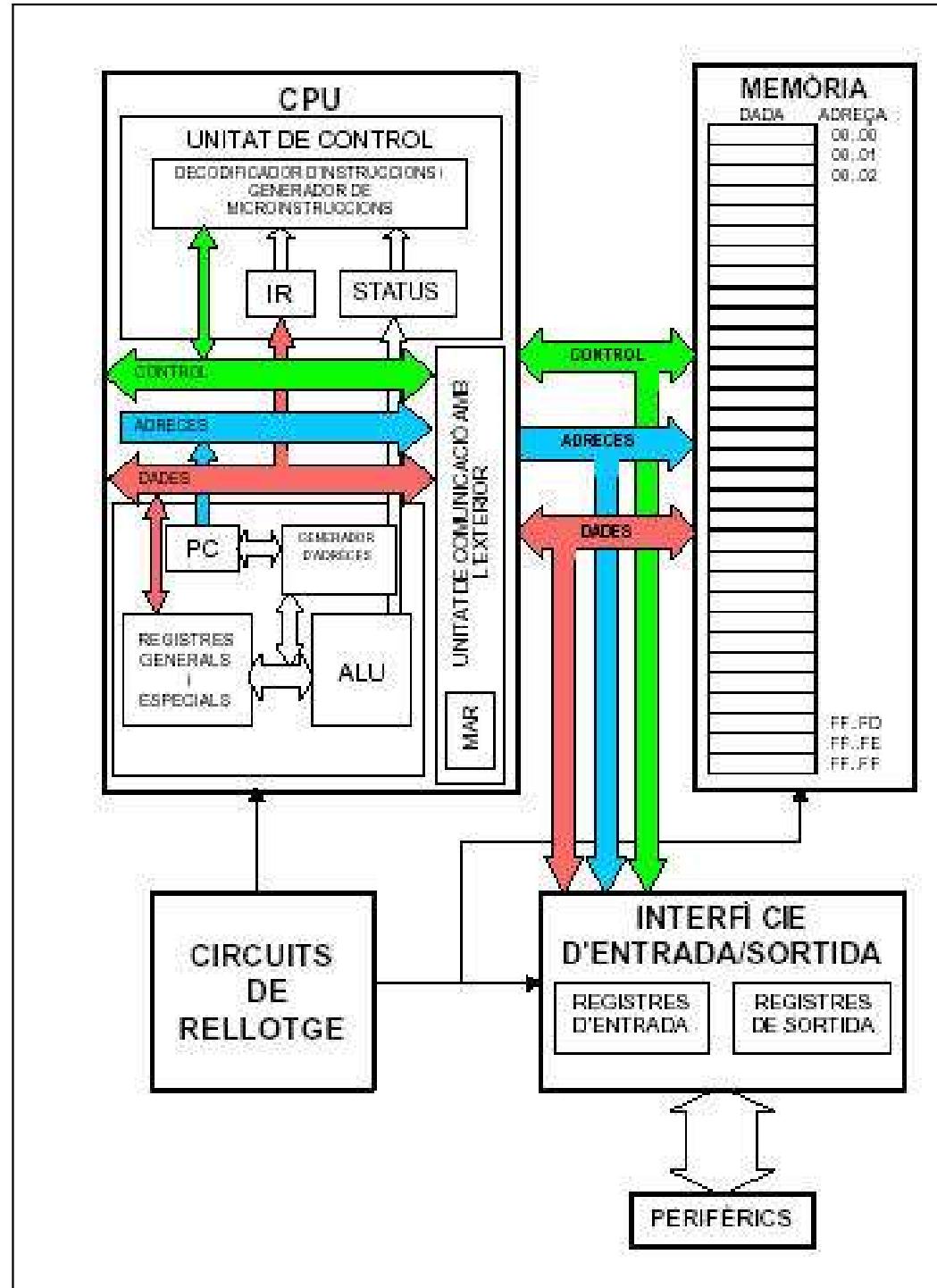


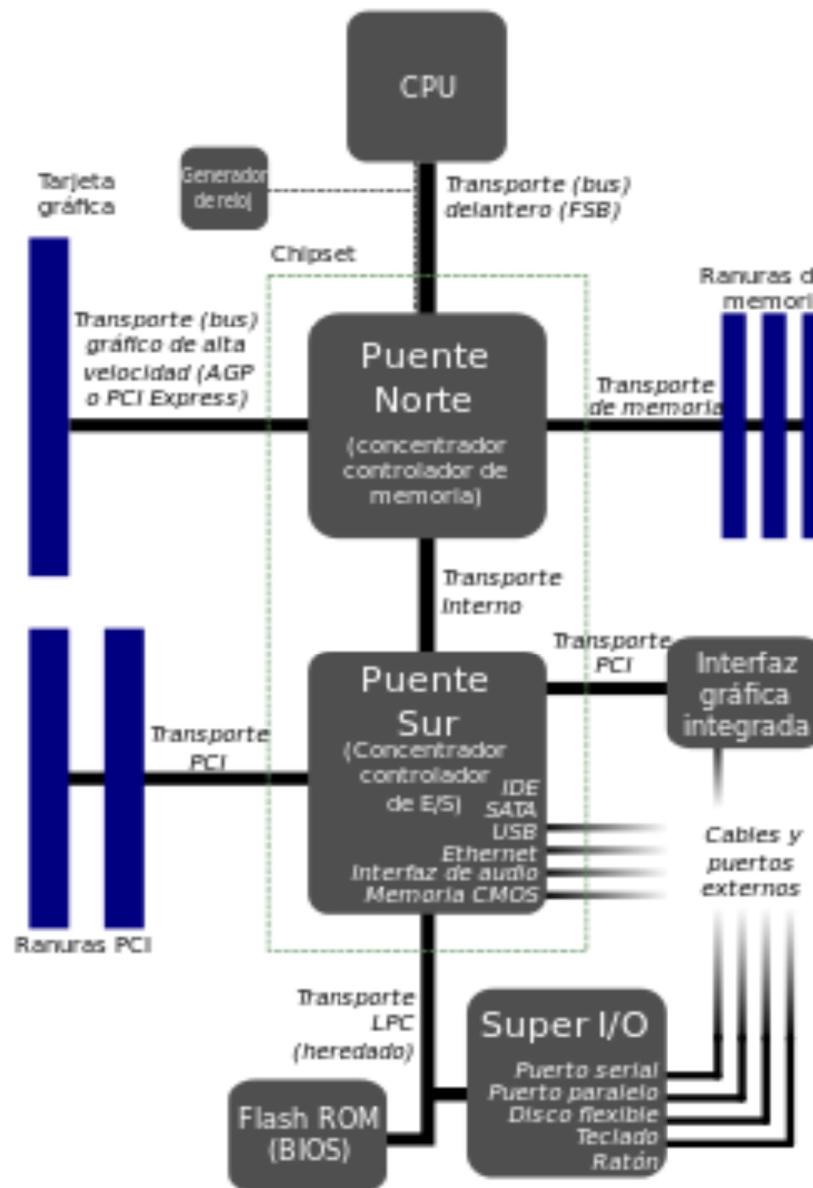
Diagrama de blocs General d'un Computador



Placa Base: Chipset

North Bridge: Controla l'accés a CPU, MP i AGP

- Conecta la CPU amb MP
- Conecta la CPU amb AGP o bé
- Conecta la CPU amb PCI Express



*Font: Puente Norte - Wikipedia

North Bridge

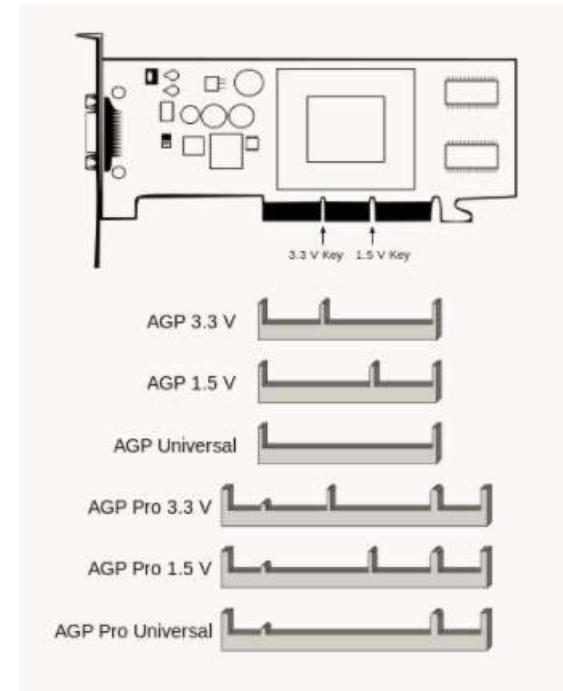
- PCI Express és un desenvolupament del bus PCI molt més ràpid
- Es basa en el bus PCI, està pensat per fer-se servir com a bus local.
- Cada ranura d'expansió porta 1, 2, 4, 8 ó 16 carrils de dades entre la placa base i les targetes connectades. Pot assolir amples de banda de 500MB/s per cada canal → en el cas de fer servir x16 podem arribar a 8GB/s en cada adreça èr a PCIE 2.x
- 8 carrils tenen un ample de banda comparable a la versió més ràpida de AGP



*Font: PCI Express - Wikipedia

North Bridge

- AGP (Accelerated Graphics Port) és una especificació de bus que proporciona una connexió directa entre l'adaptador de gràfics i la memòria. És un port → Només permet connectar un dispositiu
- Conté la majoria dels senyals del bus PCI + agregats
-

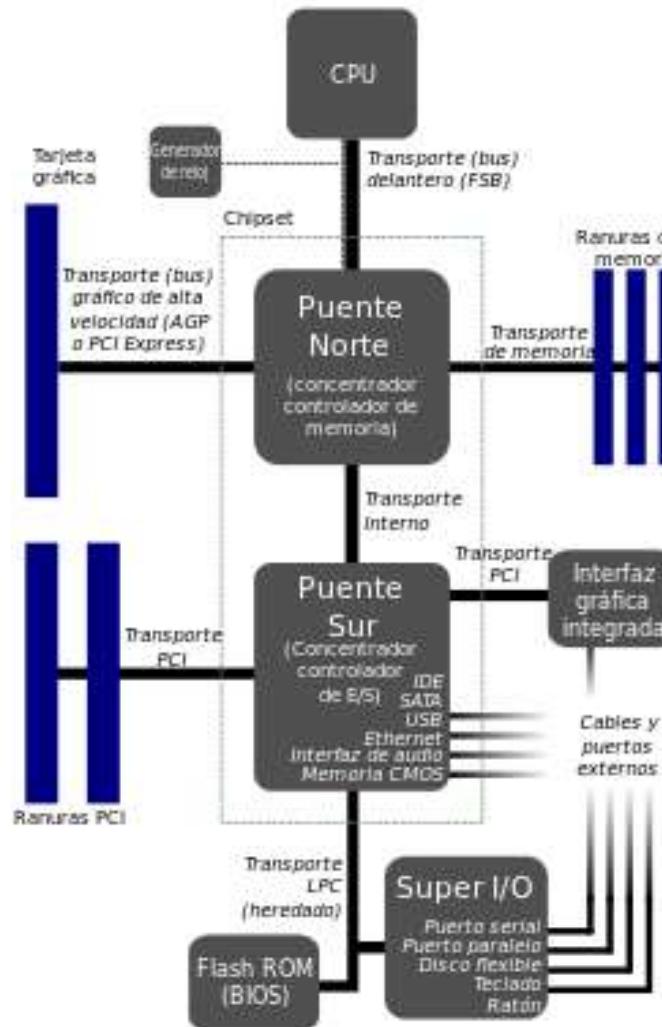


*Font: AGP - Wikipedia

Placa Base. Chipset

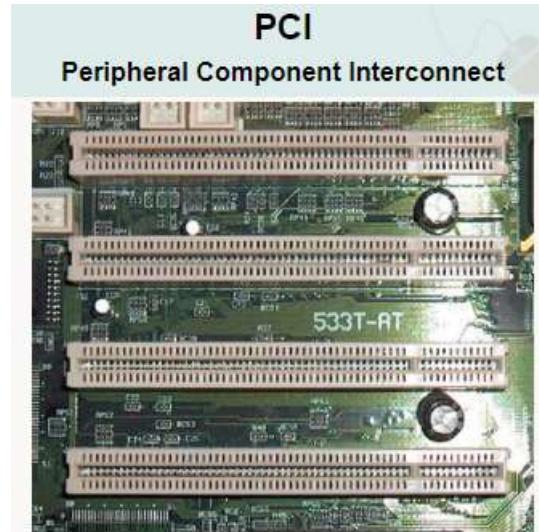
La funcionalitat que trobem als southbridges actuals inclou:

- Bus PCI
- Bus ISA
- Bus SPI
- System Management Bus (SMBus)
- Controlador DMA
- Controlador de Interrupcions
- Controlador IDE (SATA o PATA)
- Pont LPC
- Real Time Clock
- Administració de potència elèctrica APM i ACPI
- BIOS
- Interfaç de so AC97 o HD Audio.



South Bridge

- Peripheral Component Interconnect o PCI és un bus estàndard de computadores per connectar dispositius perifèrics directament a la placa base
- Molt comú en Pcs i Labtops on ha substituit al bus ISA



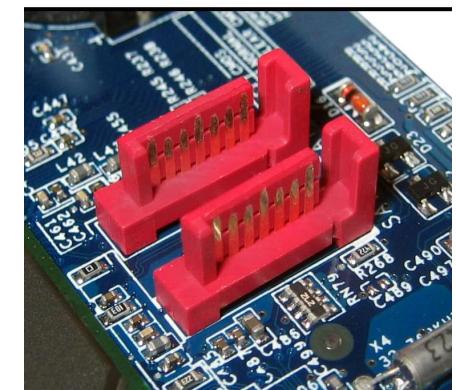
South Bridge

- Industry Standard Architecture (ISA). Dissenyat per connectar targetes d'ampliació a la placa base. Baixa velocitat. Substituït per PCI

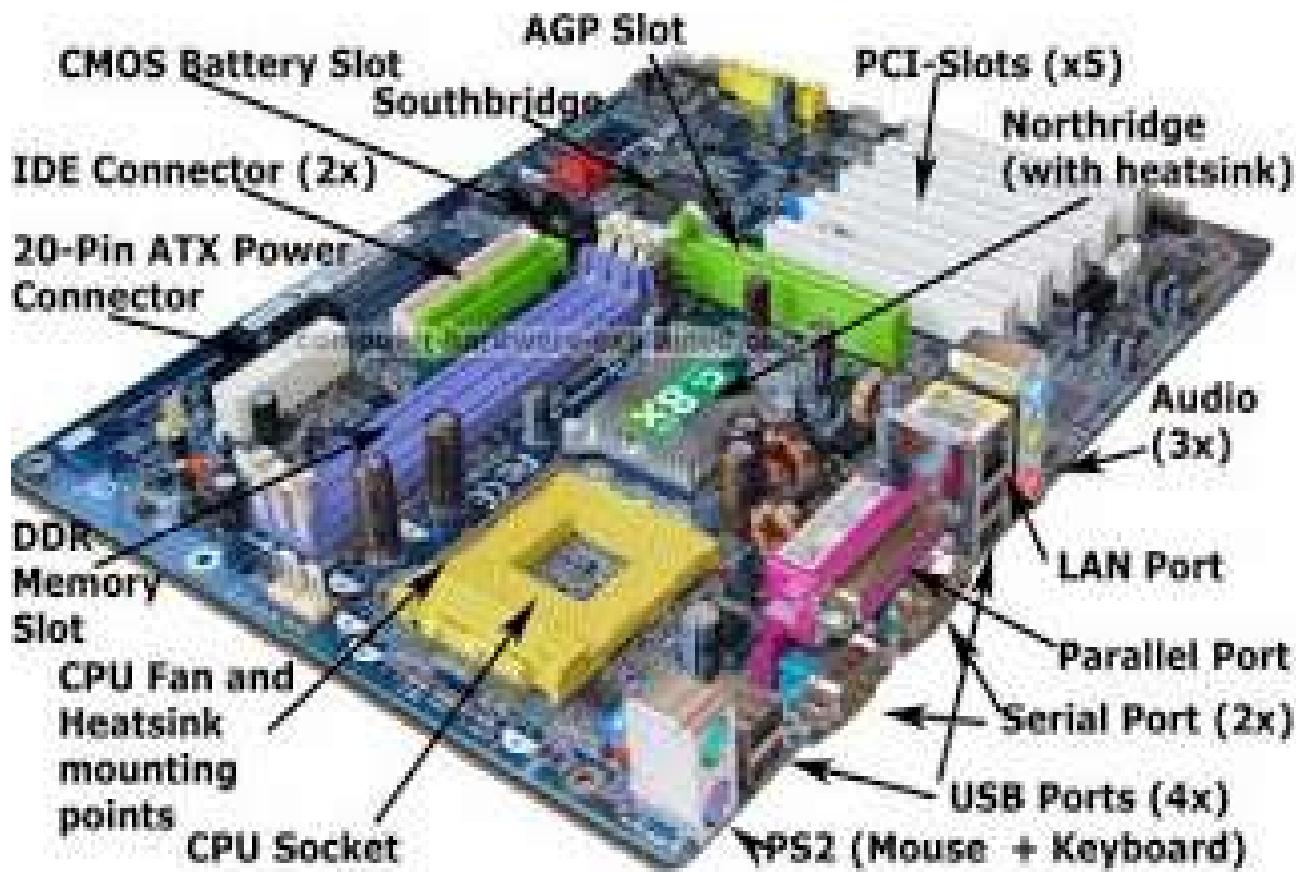


South Bridge

- Serial ATA o S-ATA (acrònim de Serial Advanced Technology Attachment) és una interfície de transferència de dades entre la placa mare i alguns dispositius d'emmagatzematge, com pot ésser el disc dur, o bé d'altres dispositius d'altres prestacions que encara s'estan desenvolupant. Serial ATA substitueix la tradicional Parallel ATA o P-ATA (estàndard que també és conegut com a IDE o ATA). El S-ATA proporciona velocitats més altes, més aprofitament quan hi ha diversos discos, més longitud de cable de transmissió de dades i capacitat per a connectar discos en calent (amb l'ordinador encès).
- Actualment és una interfície extensament acceptada i estandarditzada a les plaques mares de PC.



Placa Base: Exemplo



Placa Base: Exemplo

