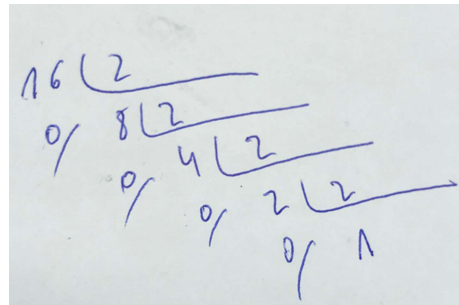


Classe Teòrico-Pràctica 1

El mon binari

- Codificació binària – decimal – Hexadecimal

binari	decimal	hexa
0	0	0
1	1	1
10	2	2
11	3	3
100	4	4
101	5	5
110	6	6
111	7	7
1000	8	8
1001	9	9
1010	10	A
1011	11	B
1100	12	C
1101	13	D
1110	14	E
1111	15	F
10000	16	10



16 = 10000b



11000000 • 00000101
_{7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0}
_{2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2}



	1	0	0	0	0
x	2 ⁴	2 ³	2 ²	2 ¹	2 ⁰
	16	+ 0	+ 0	+ 0	+ 0

2 ⁽⁷⁾	2 ⁽⁶⁾	2 ⁽⁵⁾	2 ⁽⁴⁾	2 ⁽³⁾	2 ⁽²⁾	2 ⁽¹⁾	2 ⁽⁰⁾
128	64	32	16	8	4	2	1

El món binari

- Quan treballem en binari, normalment agrupem els bits en conjunt de 8. A això l'anomenem Byte

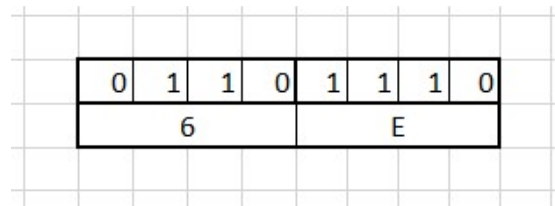


Figura 1

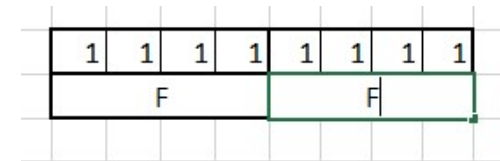


Figura 2

- La traducció binari – hexadecimal és exacta. Que vol dir amb això? Mirem la figura 2. Correspon a un Byte en binari amb tot uns. En hexa és FF. Incrementar en una unitat qualsevol dels dos implica passar a un nou byte. El funcionament és igual en tots dos casos. Per això es fa servir tant l'hexadecimal en l'àrea d'arquitectura i estructura de computadors

El mon binari

- Operacions aritmètiques
 - Suma
 - Resta
 - Multiplicació
 - Divisió
 - Desplaçament aritmètic
- Operacions lògiques
 - AND
 - OR
 - XOR
 - NOT
 - NAND
 - NOR
 - Desplaçament lògic

El mon binari

- Exercicis:

A partir del següents nombres binaris: $A = 1001\ 0011$ i $B = 0011\ 1100$

Feu les següents operacions

$A \cdot B$

$A \text{ AND } B$

$A + B$

$A \text{ OR } B$

Comenteu els resultats

El mon binari

Decimal <u>term</u>	Abbreviation	value	Binary term	Abbreviation	value	% larger
kilobyte	kB	10^3	kibibyte	kiB	2^{10}	2%
megabyte	MB	10^6	mebibyte	MiB	2^{20}	5%
gigabyte	GB	10^9	gibibyte	GiB	2^{30}	7%
terabyte	TB	10^{12}	tebibyte	TiB	2^{40}	10%
petabyte	PB	10^{15}	pebibyte	PiB	2^{50}	13%
exabyte	EB	10^{18}	exbibyte	EiB	2^{60}	15%
zettabyte	ZB	10^{21}	zebibyte	ZiB	2^{70}	18%
yottabyte	YB	10^{24}	yobibyte	YiB	2^{80}	21%

Ambigüitat de 2^x vs 10^x es va resoldre afegint una notació binària a tots els termes associats a l'amplada dels nombres

Representació dels nombres

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

En C:

unsigned int: 32 bits sense signe

Int: 32 bits amb signe representat en Ca2

Long: 64 bits amb signe representat en Ca2

Representació dels nombres

- Binari amb signe
 - Es reserva el bit més significatiu per guardar el signe
 - Bit més significatiu a 1 implica n^o negatiu
 - Bit més significatiu a 0 implica n^o positiu
 - L'operació no és immediata
 - Ex: 7 -3
0111 = 7
1011 = -3 0111+1011 = 0010 + bit de carry

Representació dels nombres

- Complement a 1

- El complement a 1 d'un determinat n° s'obté canviant els dígit 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'expresa com

$$Ca1(N^{\circ}) = 2^{n-1} + (2^{n-1} - 1) - N^{\circ}$$

- La representació del Ca1 d'un n° depen del n° de bits amb el que representem aquest n° .

$$Ca1(6) \text{ amb 6 bits} = 111001; (2^{6-1}) + (2^{6-1} - 1) - 6 = 57d$$

$$Ca1(6) \text{ amb 8 bits} = 11111001$$

Representació dels nombres

- Complement a 2

- El complement a 2 d'un determinat n° es calcula fent primer el Ca1 i després sumant-li 1
- El complement a 2 d'un determinat n° format per n dígits binaris és

$$\text{Ca2}(N) = 2^n - N$$

- Exemple: Calculem el complement a 2 de 12 amb 8 bits

$$12_{10} = 00001100_{10}; \text{Ca2}(12) = 2^8 - 12 = 244 = 11110100_{10}$$

7	0111b
6	0110b
5	0101b
4	0100b
3	0011b
2	0010b
1	0001b
0	0000b
-8	1000b
-7	1001b
-6	1010b
-5	1011b
-4	1100b
-3	1101b
-2	1110b
-1	1111b

Representació dels nombres

Binari	En binari sense signe...	En binari amb signe...	En complement a 1...	En complement a 2...
0000	0	0	0	0
0001	1	1	1	1
0010	2	2	2	2
0011	3	3	3	3
0100	4	4	4	4
0101	5	5	5	5
0110	6	6	6	6
0111	7	7	7	7
1000	8	-0	-7	-8
1001	9	-1	-6	-7
1010	10	-2	-5	-6
1011	11	-3	-4	-5
1100	12	-4	-3	-4
1101	13	-5	-2	-3
1110	14	-6	-1	-2
1111	15	-7	-0	-1

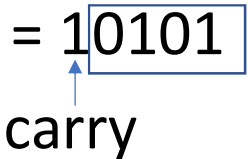
Representacions dels nombres

- Problemàtica de tenir una mida limitada:
- Overflow: Sumar dos nombres positius i que el resultat doni un valor negatiu. Exemple en una representació de 4 bits

$0101 + 0011 = 1000 \Rightarrow$ El bit més significatiu a 1, per tant valor negatiu

- Carry: La suma de dos nombres supera l'espai associat. Exemple en una representació de 4 bits i valors unsigned

$1001 + 1100 = 10101$



↑
carry

Coma flotant

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

Coma flotant

- $3,21 = 0 \underbrace{10000000}_{\text{exponent}} \underbrace{10011010111000010100100}_{\text{mantissa}}$
- $3,21 = S$ exponent mantissa
- Com?

1. Agafem la part entera: $3 = 11b$
2. Agafem la part decimal: $0,21$ i la convertim en binari:
3. Ho agrupem tot: $11.001101011100001...$
4. Passem a notació científica $1.1001101011100001 \times 2^1$
5. Valor positiu \Rightarrow Bit de signe = 0 ($-1^0 = 1$)
6. Exponent = $127 + \text{valor exponent} \Rightarrow 127 + 1 = 128$
7. La mantissa és el valor decimal calculat directament

0 10000000 1001101011100001...

op	resultat	part entera
0.21×2	0.42	0
0.42×2	0.84	0
0.84×2	1.68	1
0.68×2	1.36	1
0.36×2	0.72	0
0.72×2	1.44	1
0.44×2	0.88	0
0.88×2	1.76	1
0.76×2	1.52	1
0.52×2	1.04	1
0.04×2	0.08	0
0.08×2	0.16	0
0.16×2	0.32	0
0.32×2	0.64	0
0.64×2	1.28	1

Coma flotant

- L'operació inversa. Donat un num expressat en coma flotant:

0 10000110 1010100 0000 0000 0000 0000

1. El bit més significatiu correspon al bit de signe
2. 8 bits corresponents a l'exponent $\Rightarrow 10000110 = 134 \Rightarrow e = 134 - 127 = 7$
3. La mantissa : 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 $2^{-1} \ 2^{-2} \ 2^{-3} \ 2^{-4} \ 2^{-5} \dots$

$$m = 2^{-1} + 2^{-3} + 2^{-5} = 0,65625$$

- Per tant $x = -1^0 (1,65625) \cdot 2^7 = 212$

Coma flotant

- Transforma en representació de coma flotant el següent valor:
3,14

Un cop trobat el valor, feu el procés invers per recuperar el valor

Màscares

- Fem servir les màscares per veure el valor de un determinat bit. Per exemple, el bit de signe. Les operacions que fem servir són operacions lògiques

MASCARA = 80h

valor = XXh

if (valor && MASCARA == 0):

print('valor és positiu')

Màscares

- Feu un programa que ens indiqui si un determinat valor obtingut per una operació prèvia és positiu o negatiu fent servir màscares i operacions lògiques.
- Troba la màscara que has de fer servir per determinar que un número és negatiu. Considerant que el valor està donat en Ca2 , passa-ho al valor positiu que correspondria (ex. $-7 \Rightarrow 7$). Troba varies solucions al problema.

Punters

- Un punter o apuntador és una variable manipulable la qual fa referència a una regió de la memòria; així, el programador, en comptes de manipular la variable en si, treballa amb l'adreça de memòria en la qual la dada es troba emmagatzemada.

[https://ca.wikipedia.org/wiki/Punter_\(programació\)](https://ca.wikipedia.org/wiki/Punter_(programació))

- Els accessos a memòria normalment van referits o s'associen a l'adreça. Cal diferenciar entre ADREÇA i CONTINGUT
- En C la sintàxi és:

```
int *ptr = NULL;  
int a = 5;  
print(&a); => 0x8130  
ptr = &a; => *ptr = 5
```

Address	Contents
0x8130	0x00000005
0x8134	0x00000000

Exercici

- La instrucció MOV A, M guarda en el registre A el contingut que hi ha a la posició de memòria M. Indiqueu el contingut del registre A en funció dels possibles valors de M

@ 6	1234
@ 5	65434
@ 4	5435
@ 3	33223
@ 2	67
@ 1	23457
@ 0	65432

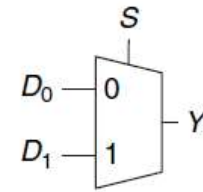
Blocs combinacionals

- Multiplexors

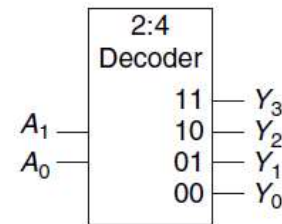
```
if (S == 0)
    Y = D0;
else if (S==1)
    Y = D1
```

- Descodificadors

```
If (A1 == 0){
    if (A0 == 0)
        OUT = Y0;
    else
        OUT = Y1;
}
else{
    if(A0 == 0)
        OUT = Y2;
    else
        OUT = Y3;
}
```



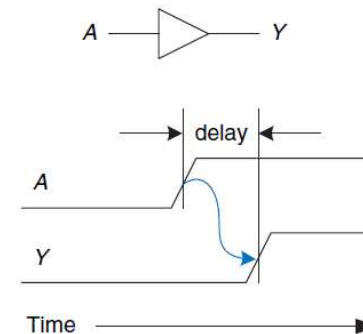
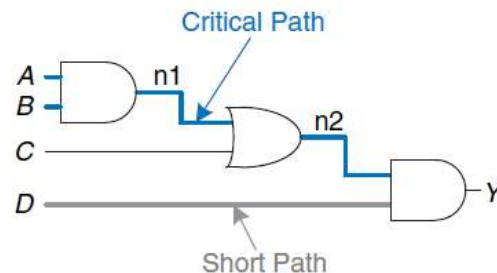
S	D ₁	D ₀	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1



A ₁	A ₀	Y ₃	Y ₂	Y ₁	Y ₀
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

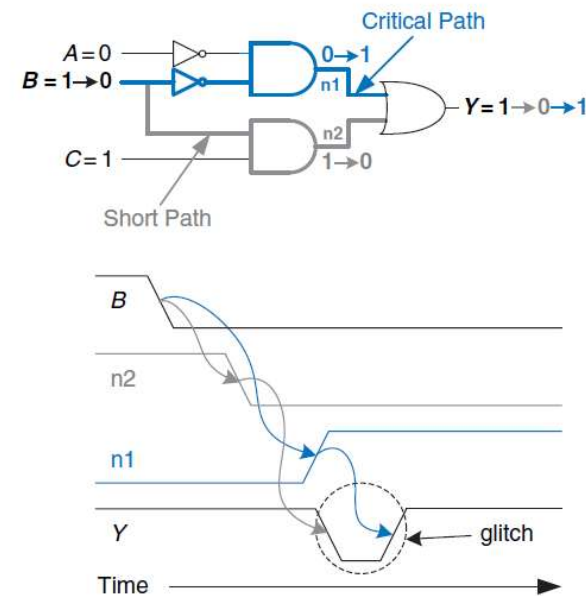
Timing

- És interessant conèixer el retard que hi ha des de que hi ha una determinada entrada en un circuit i quan la sortida associada és estable.
- Interessa conèixer en un determinat circuit quin és el camí que més triga en executar-se (camí crític)



Glitches

- És el problema que apareix quan un canvi en una entrada genera múltiples transicions en la sortida
- Implica una limitació temporal



Exercicis

1. Tenim una memòria de 4kBytes. La unitat de memòria és el word (32 bits). Quants bits necessitem per adreçar-la? Analitzeu i determineu la diferència entre adreça i contingut.
2. La instrucció LOAD R1, A carrega el contingut que hi ha a la posició de memòria A en el registre R1. Si la memòria és la de l'exercici 1, quants bits ens calen per identificar el valor de A? Si tenim 8 registres de propòsit general R0, R1, ..., R7. Quants bits calen per identificar el registre R1? Si el conjunt d'instruccions total que té aquest processador és de 100 instruccions, quants bits calen per identificar la instrucció LOAD?