Sistemes digitals (Resum)

lloc:	Cursos IOC - Batxillerat	Imprès per:	Invitado
Curs:	Electrotècnia (autoformació IOC)	Data:	divendres, 11 febrer 2022, 00:24
Llibre:	Sistemes digitals (Resum)		

Taula de continguts

1. Sistemes analògics i digitals

Els sistemes electrònics es poden classificar en dues grans categories: digitals i analògics. Aquesta classificació es fa atenent la naturalesa dels valors que poden assignar-se als diferents senyals que intervenen en el sistema.
Mentre que l'electrònica digital considera valors discrets de tensió (per exemple, un interruptor de llum només pot estar obert o tancat), l'electrònica analògica considera i treballa amb rangs de valors variables.

Sistemes analògics

Un senyal analògic (figura 1) és aquell que presenta una variació contínua en el temps. El predomini de senyals analògics en el nostre entorn (variacions del corrent, la tensió, força, etc.) en fa necessari l'estudi. Els senyals analògics es transformen en senyals elèctrics per al seu tractament.

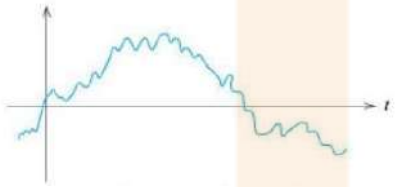


Figura 1. Senyal analògic

Els dispositius que constitueixen aquest tipus de sistemes treballen en la zona lineal, és a dir, els senyals de sortida estan relacionats per una constant amb els de l'entrada. Poden prendre qualsevol valor dins d'un rang delimitat pel mateix sistema.

1.2. Sistemes digitals

Un senyal digital és aquell que presenta una variació discontínua en el temps i que només pot prendre certs valors concrets:

Figura 2. Senyal digital

Un sistema digital típic parteix del fet que tots els seus senyals són binaris, i per tant tots els dispositius que constitueixen un sistema digital només adopten dos nivells o estats. Aquests estats hauran d'estar perfectament diferenciats i a cadascun li correspondrà el valor d'una de les magnituds del sistema.

2. Sistemes de numeració

El **sistema decimal** és el sistema de numeració que utilitzem habitualment, i és un sistema de numeració en el qual les quantitats es representen utilitzant com a base el número deu. Ho expressem de la manera següent: $_{(10}$. No obstant això, en contextos com el de la informàtica s'utilitzen sistemes de numeració més específics com el binari o l'hexadecimal.

Sistema binari

El sistema binari és un sistema de numeració utilitzat en electrònica digital en el qual només poden donar-se dos nivells o estats possibles. És un sistema en base dos: 0 ó 1, nivell alt o nivell baix.

En informàtica s'utilitzen els codis binaris per emmagatzemar informació, fer operacions aritmètiques, reparar errors, etc.

La conversió d'un nombre **decimal a binari** consisteix a dividir successivament el nombre decimal per dos, per a la part sencera. L'últim quocient i les restes formen el nombre en base 2 que es representa: $_{(2}$. A continuació, es pot observar de manera detallada un exemple:

Exemples

1. Conversió de decimal a binari del nombre 19.

```
19  |2
 1   9  |2
        1   4  |2
               0   2  |2
                      0   1
```

$19_{(10} = 10011_{(2}$

El nombre binari s'obté de l'últim quocient i totes les restes col·locats en l'ordre invers. És a dir, que l'últim quocient serà el bit més significatiu i anirà seguit de l'última resta. La segona resta anirà en tercera posició i així successivament fins a arribar al bit menys significatiu, que es correspon amb la primera resta obtinguda.

Conversió dels nombres decimals següents a binari: 54, 125 i 563:

$54_{(10} = 110110_{(2}$ $125_{(10} = 1111101_{(2}$

$563_{(10}=1\,0\,0\,0\,1\,1\,0\,0\,1\,1_{(2}$

Per a convertir un nombre **binari a decimal**, s'utilitza l'expressió polinòmica:

$a_n \cdot 2^n + \ldots + a_2 \cdot 2^2 + a_1 \cdot 2^1 + a_0 \cdot 2^0$

El número 2 correspon a la base numèrica de la que es bol fer el canvi per passar ho a decimal

Exemple

Per passar a decimal el nombre binari: $10001_{(2}$, es fa de la següent forma:

$$1\cdot 2^4+0\cdot 2^3+0\cdot 2^2+0\cdot 2^1+1\cdot 2^0=1\cdot 2^4+1=16+1=17_{(10}$$

Observeu que és el mateix nombre que havíem convertit anteriorment de manera inversa.

Decimal	Binari	Decimal	Binari
0	0000	6	0110
1	0001	7	0111
2	0010	8	1000
3	0011	9	1001
4	0100	10	1010
5	0101	11	1011

Taula 1. Numeració decimal i binària

Sistema de numeració binari BCD

Consisteix en transformar cada dígit decimal a un grup de 4 dígits binaris. Amb aquest codi expressem cada un dels dígits decimals en binari per separat. Per tant, necessitem representar del 0 al 9, i per a fer-ho necessitem 4 bits, doncs 2^3 només ens deixa representar 8 valors, de 0 a 7. Amb 2^4 (des de 0000 fins a 1111) podem representar fins a 16 números diferents.

Exemple

1.- Volem representar en el sistema BCD el número decimal 23:

farem la composició del 2 i del 3 de la forma següent:

2 -> 0010 ; 3 -> 0011 Per tant la solució serà el número BCD: **0010 0011**

2.- El numero 179 expressat en codi BCD seria:

1 -> 0001 ; 7 -> 0111 ; 9 -> 1001, la solució seria**: 0001 0111 1001**

Sistema de numeració Hexadecimal

El sistema hexadecimal és un sistema numèric amb base 16. Es representa normalment utilitzant els símbols 0–9 i A–F o a–f. Per exemple, el nombre decimal 79, la representació del qual en sistema binari és 01001111, es pot escriure com 4F en hexadecimal (4 = 0100, F = 1111).

1.- Per a convertir un nombre decimal en hexadecimal manualment, cal dividir el nombre decimal entre 16; el quocient enter d'aquesta divisió es torna a dividir per 16 i així successivament. Quan el darrer quocient sigui inferior a 16, s'escriuen, un darrere de l'altre, el darrer quocient obtingut i tots els residus en ordre invers al de la seva obtenció, substituint aquells nombres que siguin més

grans de 9 per la seva lletra corresponent (A=10, B=11, C=12, D=13, E=14 i F=15).

Exemple: convertir el nombre 41716 en hexadecimal:

41716 | 16
4 2607 | 16
15 162 | 16
2 10 (= A)

considerant que 15 = F i 10 = A
Resultat: A2F4

El procés invers es realitza utilitza l'expressió polinòmica amb base 16

$a_n \cdot 16^n + \ldots + a_2 \cdot 16^2 + a_1 \cdot 16^1 + a_0 \cdot 16^0$

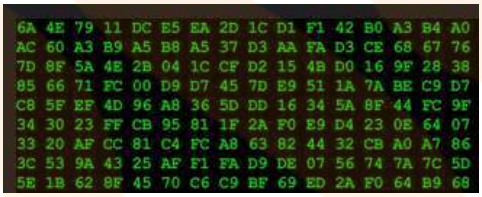
Exemple: convertir el nombre A2F4 en decimal:

$A \cdot 16^3 + 2 \cdot 16^2 + F \cdot 16^1 + 4 \cdot 16^0$

A → 10 x 16^3 = 40960
2 → 2 x 16^2 = 512
F → 15 x 16^1 = 240
4 → 4 x 16^0 = 4

TOTAL = 41716

Resultat: 41716



Codi màquina visualitzat en un tub de fosfor verd

3. Àlgebra de Boole

L'àlgebra de Boole opera amb **relacions lògiques** i les **variables** només poden prendre dos valors diferents –**vertader** o **fals**–, que es representen simbòlicament amb un **1** ó un **0** respectivament.

L'àlgebra de Boole, en operar només amb dos estats, s'utilitzarà com a eina matemàtica en aquells circuits que presenten dos estats estables de funcionament ben diferenciats: components electrònics elementals (díodes, transistors, etc.) i com a conseqüència, blocs lògics o circuits digitals, elements elèctrics. És per això que l'àlgebra de Boole es presenta com el suport matemàtic ideal per al disseny i l'anàlisi de circuits electrònics digitals. Així, per exemple, l'emmagatzematge de dades en una memòria es realitza mitjançant zeros i uns, la qual cosa implica que els elements bàsics que formen la memòria es troben físicament en un dels dos estats possibles de funcionament. El mateix passa amb els microprocessadors i la resta de dispositius digitals.

3.1. Operacions lògiques

Les operacions que es fan amb les variables digitals són la **suma lògica** o operació O (OR en anglès), el **producte lògic** o operació I (AND en anglès) o la **negació** o **inversió** (NOT en anglès). La següent figura representa el resultat d'aquestes operacions en funció dels 2 possibles valors (0 ó 1) que poden tenir unes variables digitals a i b:

suma			producte			negació	
a	**b**	**a+b**	**a**	**b**	**a·b**	**a**	**ā**
0	0	0+0=0	0	0	0·0=0	0	1
0	1	0+1=1	0	1	0·1=0	1	0
1	0	1+0=1	1	0	1·0=0		
1	1	1+1=1	1	1	1·1=1		

Respecte a les matemàtiques tradicionals, hem de fixar-nos en el cas diferenciat que es presenta 1+1=1 i com a concepte en aquesta operació digital estem sumant una cosa verdadera a una altre que també ho és i, per tant, el resultat també serà verdader.

La negació d'una variable digital és l'altre valor possible, és a dir, el valor digital s'inverteix, negar un 1 és un 0, i negar un 0 és un 1. L'operació negació es representa mitjançant un guió a sobre de la variable digital que es nega. La funció **a** negada és **ā**.

Propietats

a.- **Commutativa**

$$a \cdot b = b \cdot a$$

b.- **Associativa**

Respecte la suma

$$a + (b + c) = (a + b) + c$$

Respecte el producte

$$a \cdot (b \cdot c) = (a \cdot b) \cdot c$$

c.- **Distributiva**

Respecte la suma

$$a \cdot (b + c) = (a \cdot b) + (a \cdot c)$$

Respecte el producte

$$a + (b \cdot c) = (a + b) \cdot (a + c)$$

Teoremes o operacions lògiques

a.- Sumar o multiplicar una variable digital per ella mateixa dona ella mateixa:

$a+a=a \qquad a\cdot a=a$

b.- Qualsevol cosa sumada a 1 dona 1 i qualsevol cosa multiplicada per 0 és 0:

$a+1=1 \quad a\cdot 0=0$

c.- La negació és l'operació complementaria respecte a la suma i el producte:

$a+\overline{a}=1 \quad a\cdot\overline{a}=0$

d.- Lleis de l'absorció, es verifiquen les següents igualtats:

1.- $a+a\cdot b=a$ 2.- $a+\overline{a}\cdot b=a+b$ 3.- $a\cdot(\overline{a}+b)=a\cdot b$

Veiem les demostracions:

1.- $a+a\cdot b=a\cdot(1+b)=a\cdot 1=a$

2.- $a+\overline{a}\cdot b=(a+\overline{a})\cdot(a+b)=1\cdot(a+b)=a+b$

3.- $a\cdot(\overline{a}+b)=(a\cdot\overline{a})+(a\cdot b)=0+(a\cdot b)=a\cdot b$

e.- *Primera llei de De Morgan:* sumar 2 valors i negar-los és igual a negar-los i multiplicar-los

$\overline{(a+b)}=\overline{a}\cdot\overline{b}$

f.- *Segona llei de De Morgan:* multiplicar 2 valors i negar-los és negar-los i sumar-los

$\overline{(a\cdot b)}=\overline{a}+\overline{b}$

4. Taula de la veritat

La taula de la veritat és un quadre format per tantes columnes com variables d'entrada més la corresponent a la mateixa funció de sortida, i per tantes files com combinacions binàries és possible construir.
La fórmula que indica en nombre de files o combinacions possibles de les variables d'entrada és: 2^n, on n és el nombre de variables.

Per a cada combinació possible de les entrades, la taula de veritat ens dona el valor digital de la sortida.

Exemples habituals de nombre de combinacions possibles són:

- 2 variables suposaran 4 combinacions (2^2=4)

- 3 variables suposaran 8 combinacions (2^3 = 8)

- 4 variables suposaran 16 combinacions (2^4 = 16)

Veieu totes les possibilitats per 2 i 3 variables o entrades (que seran les úniques que farem servir):

Taula de veritat per a 2 entrades a,b

a	**b**	***S*** (sortida)
0	0	
0	1	
1	0	
1	1	

Són 4 casos possibles

Per a obtenir la taula de veritat a partir d'una funció lògica, substituïm cada combinació d'entrades en la funció i apuntem el valor de la sortida en el seu lloc corresponent.

Taula de veritat per a 3 entrades a,b,c

a	**b**	**c**	***S*** (sortida)
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Són 8 casos possibles

La taula de la veritat apareix com una de les dades característiques més significatives en els catàlegs dels dispositius digitals integrats. El seu coneixement és fonamental per analitzar el funcionament i l'aplicació de cada bloc o funció lògica.

5. Funció lògica o booleana

Es defineix com a funció lògica o booleana tota variable binària ***S*** (funció de sortida) en què el seu valor depèn d'una expressió algebraica formada per altres variables binàries relacionades mitjançant els signes que representen les operacions suma, multiplicació i negació.

Primer exemple:

La següent funció lògica: $S=a+\overline{b}$ ens diu que la sortida ***S*** s'obté de totes les combinacions possibles de les variables a i b, recordeu que eren $2=2^2=4$.

a	**b**	$\overline{b}$	$S=a+\overline{b}$
0	0	1	1
0	1	0	0
1	0	1	1
1	1	0	1

Segon exemple:

La següent funció lògica: $S=a\cdot\overline{b}+c$ ens diu que la sortida ***S*** s'obté de totes les combinacions possibles de les variables a, b i c, recordeu que eren $2=2^3=8$.

Si ens diuen que en un determinat moment a = 1, b = 0 i c = 1, substituïm aquests valors en la funció lògica, de forma que ara la sortida serà: $S=1\cdot\overline{0}+1$. Substituïm els valors negats per la seva inversa: $S=1\cdot1+1$ i operem els productes i les sumes: $S=1\cdot1+1=1+1=1$. El resultat final és que la sortida serà 1 quan les entrades siguin 1, 0, 1.

a	**b**	**c**	$\overline{b}$	$a\cdot\overline{b}$	$S=a\cdot\overline{b}+c$
0	0	0	1	0	0
0	0	1	1	0	1
0	1	0	0	0	0
0	1	1	0	0	1
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	0	0	0
1	1	1	0	0	1

6. Circuits digitals

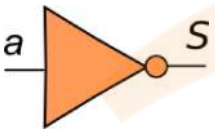
Una altra forma de representar circuits digitals és mitjançant la combinació elèctrica de circuits digitals comercials que realitzen les operacions digitals més bàsiques. Aquests circuits simples s'anomenen "portes lògiques" i són implementacions físiques de les operacions suma, producte i negació.
Les portes lògiques es representen mitjançant una simbologia normalitzada. Existeixen dos simbologies utilitzades per a representar les portes lògiques: la simbologia MIL (MILitary standard graphics symbol for logic diagrams) i la simbologia IEC (International Electrotechnical Commission) . És obligatori utilitzar la normativa IEC normalitzada i la simbologia MIL és cada vegada menys utilitzada, però encara no es pot eliminar perquè encara hi ha moltes indústries que la fan servir. A l'examen sempre us posaran la IEC, que és l'adoptada per les normes DIN.

Les portes lògiques

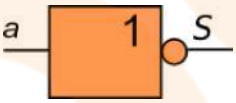
a) **Porta lògica NOT:** realitza la negació o inversió

A la sortida hi haurà la negada de la entrada. (Per no fer tant farragosos els circuits digitals, després veureu com per fer la negada d'una funció simplement afegim un rodona petita, com la que podeu veure al costat de la lletra S del dibuix). $S=\bar{a}$

Simbologia antiga MIL (militar)



Simbologia IEC



Taula de la veritat funció NOT

a	S
0	1
1	0

En un circuit elèctric convencional equivaldria a tenir un contacte normalment tancat:

b) **Porta lògica OR**: realitza l'addició (suma) lògica

- Si tenim dues variables *a* i *b*, la funció resultant quedarà definida per l'expressió $S=a+b$

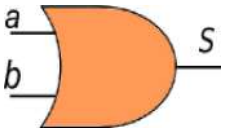
Simbologia MIL (militar)

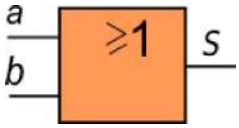
OR

Simbologia IEC

OR

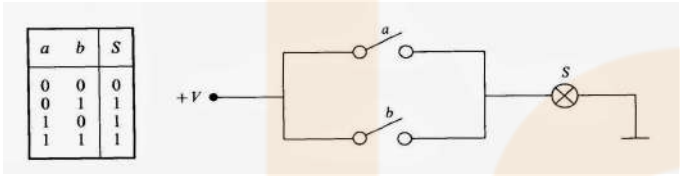
Taula de la veritat de la funció OR





a	b	$S=a+b$
0	0	0
0	1	1
1	0	1
1	1	1

El circuit elèctric equivalent en aquest cas correspondria a tenir les dues entrades en paral·lel. Tant si activem una com l'altra, com les dues, la bombeta s'encendrà:

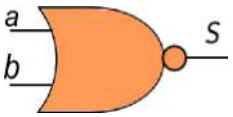


c) **Porta lògica NOR**: realitza l'addició (suma) lògica i una posterior negació.

- Si tenim dues variables *a* i *b*, la funció resultant quedarà definida per l'expressió $S=\overline{(a+b)}$

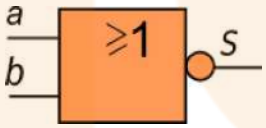
Simbologia MIL

NOR



Simbologia IEC

NOR



Taula de la veritat de la funció NOR

a	b	$S=\overline{a+b}$
0	0	1
0	1	0
1	0	0
1	1	0

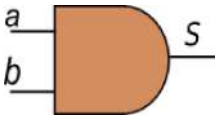
Fixeu-vos com per fer el símbol de la negació s'ha afegit una rodoneta al final

d) **La porta lògica AND**: realitza el producte lògic

- Si tenim dues variables *a* i *b*, la funció resultant quedarà definida per l'expressió $S=a\cdot b$

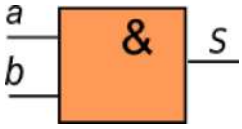
Simbologia MIL (militar)

AND



Simbologia IEC

AND



Taula de la veritat de la funció AND

a	b	$S=a\cdot b$
0	0	0
0	1	0
1	0	0
1	1	1

El circuit elèctric equivalent en aquest cas correspondria a tenir les dues entrades en sèrie. Si activem només una no s'encendrà, només si activem les dues, la bombeta s'encendrà:

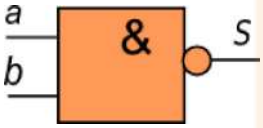
a	b	S
0	0	0
0	1	0
1	0	0
1	1	1

e) **Porta lògica NAND**: realitza el producte lògic i una posterior negació.

- Si tenim dues variables *a* i *b*, la funció resultant quedarà definida per l'expressió .

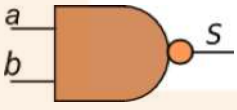
Simbologia MIL (militar)

NAND



Simbologia IEC

NAND



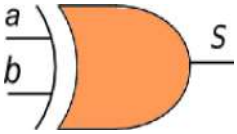
Taula de la veritat de la funció NAND

a	b	$S=\overline{a\cdot b}$
0	0	1
0	1	1
1	0	1
1	1	0

f) **Porta lògica XOR**: implementa la disjunció lògica exclusiva, és a dir, es comporta segons la taula de la veritat següent: surt un 1 si una i només una de les entrades és un 1. Si les dues entrades són 1 o les dues són 0, el resultat és un 0.

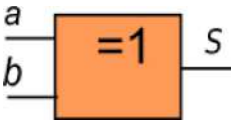
Simbologia antiga MIL (militar)

XOR



Simbologia moderna IEC

XOR



Taula de la veritat de la funció XOR

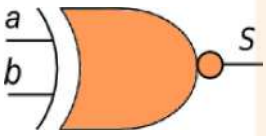
a	b	$S=a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0

Donada la taula de la veritat anterior, la funció XOR serà: $S=\overline{a} \cdot b+a \cdot \overline{b}$

g) **Porta lògica XNOR**: implementa l'equivalència o comparació. La sortida és ALTA (1) si les dos entrades són iguals, i si són diferents la sortida és zero.

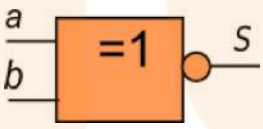
Simbologia MIL (militar)

XNOR



Simbologia IEC

XNOR



Taula de la veritat de la funció XNOR

a	b	$S=\overline{a \oplus b}$
0	0	1
0	1	0
1	0	0
1	1	1

Donada la taula de la veritat anterior, la funció XNOR serà: $S=\overline{a} \cdot \overline{b}+a \cdot b$

6.1. Exemples

1.- *Obtenir la taula de veritat i dibuixar l'esquema que correspon a la funció lògica:*

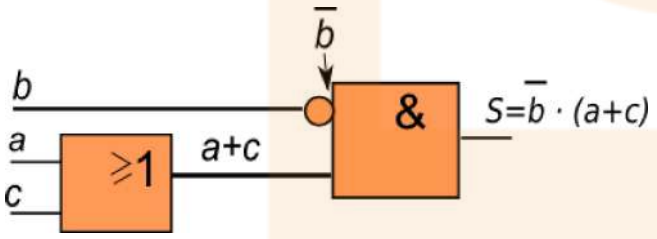
$S=\overline{b}\cdot(a+c)$

Per a obtenir la taula de veritat, substituïm cada combinació d'entrades en la funció:

a	b	c	Sortida $S=\overline{b}\cdot(a+c)$
0	0	0	$S=\overline{0}\cdot(0+0)=1\cdot(0+0)=1\cdot0$ = **0**
0	0	1	$S=\overline{0}\cdot(0+1)=1\cdot(0+1)=1\cdot1$ = **1**
0	1	0	$S=\overline{1}\cdot(0+0)=0\cdot(0+0)=0\cdot0$ = **0**
0	1	1	$S=\overline{1}\cdot(0+1)=0\cdot(0+1)=0\cdot1$ = **0**
1	0	0	$S=\overline{0}\cdot(1+0)=1\cdot(1+0)=1\cdot1$ = **1**
1	0	1	$S=\overline{0}\cdot(1+1)=1\cdot(1+1)=1\cdot1$ = **1**
1	1	0	$S=\overline{1}\cdot(1+0)=0\cdot(1+0)=0\cdot1$ = **0**
1	1	1	$S=\overline{1}\cdot(1+1)=0\cdot(1+1)=0\cdot1$ = **0**

Veiem que la sortida serà 1 quan b sigui 0 i a o c siguin 1.

Per a obtenir l'esquema de connexió, hem de utilitzar una porta lògica NOT per a negar l'entrada b, una porta OR per a sumar les entrades a+c, i després utilitzarem una porta lògica AND per a multiplicar les sortides de les altres dos portes, b negat per la suma de a,c:



OR AND

2.- *Obtenir la taula de veritat i dibuixar l'esquema que correspon a la funció lògica*

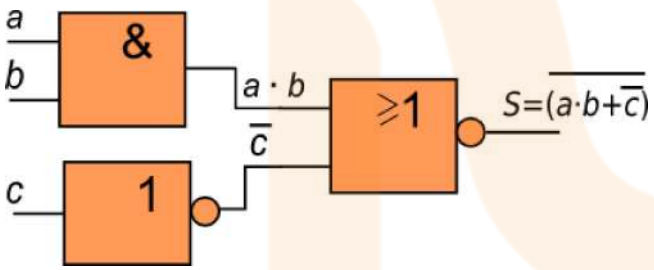
$S=(\overline{a\cdot b+\overline{c}})$

Per a obtenir la taula de veritat, substituïm cada combinació d'entrades en la funció. Anirem realitzant primer les operacions més internes dins del parèntesi: primer c negat, després a·b, sumarem aquestes operacions i acabarem fent la negació final.

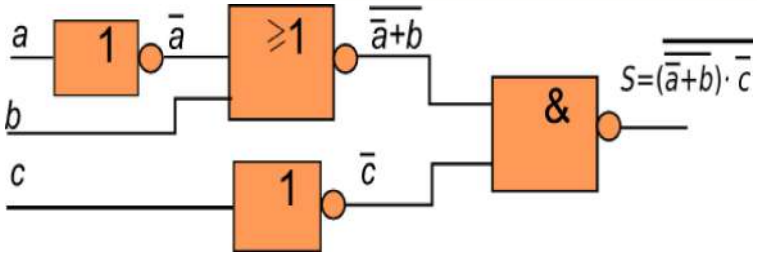
a	b	c	Sortida $S=(\overline{a\cdot b+\overline{c}})$
0	0	0	$S=\overline{(0\cdot 0+\overline{0})}=\overline{(0\cdot 0+1)}=\overline{(0+1)}=\overline{1}$ =**0**
0	0	1	$S=\overline{(0\cdot 0+\overline{1})}=\overline{(0\cdot 0+0)}=\overline{(0+0)}=\overline{0}$ =**1**
0	1	0	$S=\overline{(0\cdot 1+\overline{0})}=\overline{(0\cdot 1+1)}=\overline{(0+1)}=\overline{1}$ =**0**
0	1	1	$S=\overline{(0\cdot 1+\overline{1})}=\overline{(0\cdot 1+0)}=\overline{(0+0)}=\overline{0}$ =**1**
1	0	0	$S=\overline{(1\cdot 0+\overline{0})}=\overline{(1\cdot 0+1)}=\overline{(0+1)}=\overline{1}$ =**0**
1	0	1	$S=\overline{(1\cdot 0+\overline{1})}=\overline{(1\cdot 0+0)}=\overline{(0+0)}=\overline{0}$ =**1**
1	1	0	$S=\overline{(1\cdot 1+\overline{0})}=\overline{(1\cdot 1+1)}=\overline{(1+0)}=\overline{1}$ =**0**
1	1	1	$S=\overline{(1\cdot 1+\overline{1})}=\overline{(1\cdot 1+0)}=\overline{(1+0)}=\overline{1}$ =**0**

La sortida serà 0 quan **c** sigui 0 o quan el producte **a** per **b** sigui 1.

Per a obtenir l'esquema de connexió, hem d'utilitzar una porta lògica NOT per a negar l'entrada c, una porta AND per a multiplicar les entrades a · b, i després utilitzarem una porta lògica NOR per a sumar les sortides de les altres dos portes amb un posterior negat:



3.- *Obtenir la funció lògica i la taula de veritat a partir del següent esquema donat:*



Per a trobar la funció lògica hem de seguir l'esquema, escriurem de forma seqüencial la sortida que hi haurà en cada porta lògica i que serà una de les entrades de la següent porta lògica. En la sortida de la primera NOT tenim el negat de l'entrada a, per tant la sortida de la porta NOR serà la suma de les 2 entrades (el negat més b) amb un posterior negació $\overline{(\overline{a}+b)}$.

La sortida de la segona porta NOT nega l'entrada c, que és una entrada de la porta NAND final. L'altra entrada de la porta NAND és la sortida de la porta NOR que era $\overline{(a+b)}$ i per tant la sortida final de l'esquema serà la multiplicació de les 2 entrades anteriors i una posterior negació de tot:

$$S=\overline{(\overline{(a+b)}\cdot c)}$$

Per a obtenir la taula de veritat, substituïm cada combinació d'entrades en la funció. Anirem realitzant primer les operacions més internes dins del parèntesi: primer negar a i c, després fer la suma que hi ha dins del parèntesi i negar-la, fer la multiplicació interior, i acabarem fent la negació final.

a	b	c	Sortida $S=\overline{(\overline{(a+b)}\cdot c)}$
0	0	0	$S=\overline{(\overline{(0+0)}\cdot\overline{0})}=\overline{(\overline{(1+0)}\cdot 1)}=\overline{(\overline{(1)}\cdot 1)}=\overline{(0\cdot 1)}=\overline{0}=1$
0	0	1	$S=\overline{(\overline{(0+0)}\cdot\overline{1})}=\overline{(\overline{(1+0)}\cdot 0)}=\overline{(\overline{(1)}\cdot 0)}=\overline{(0\cdot 0)}=\overline{0}=1$
0	1	0	$S=\overline{(\overline{(0+1)}\cdot\overline{0})}=\overline{(\overline{(1+1)}\cdot 1)}=\overline{(\overline{(1)}\cdot 1)}=\overline{(0\cdot 1)}=\overline{0}=1$
0	1	1	$S=\overline{(\overline{(0+1)}\cdot\overline{1})}=\overline{(\overline{(1+1)}\cdot 0)}=\overline{(\overline{(1)}\cdot 0)}=\overline{(0\cdot 0)}=\overline{0}=1$
1	0	0	$S=\overline{(\overline{(1+0)}\cdot\overline{0})}=\overline{(\overline{(0+0)}\cdot 1)}=\overline{(\overline{(0)}\cdot 1)}=\overline{(1\cdot 1)}=\overline{1}=0$
1	0	1	$S=\overline{(\overline{(1+0)}\cdot\overline{1})}=\overline{(\overline{(0+0)}\cdot 0)}=\overline{(\overline{(0)}\cdot 0)}=\overline{(1\cdot 0)}=\overline{0}=1$
1	1	0	$S=\overline{(\overline{(1+1)}\cdot\overline{0})}=\overline{(\overline{(0+1)}\cdot 1)}=\overline{(\overline{(1)}\cdot 1)}=\overline{(0\cdot 1)}=\overline{0}=1$
1	1	1	$S=\overline{(\overline{(1+1)}\cdot\overline{1})}=\overline{(\overline{(0+1)}\cdot 0)}=\overline{(\overline{(1)}\cdot 0)}=\overline{(0\cdot 0)}=\overline{0}=1$

7. Obtenció de funcions

Per obtenir el circuit amb portes lògiques a partir d'una expressió matemàtica (funció lògica), només cal utilitzar la porta corresponent a l'operació lògica que cal efectuar.
Per obtenir la funció lògica a partir de l'esquema d'un circuit es fa l'operació inversa. Es parteix de les variables d'entrada i es col·loca a la sortida de cada porta l'equació que la resol, d'acord amb les seves variables d'entrada. Les sortides de les portes es tracten com a entrades de les portes següents a les quals estan connectades, i així successivament fins a arribar a la sortida final del circuit.

Ens falta aprendre un mètode per a trobar una funció lògica vàlida que realitzi una taula de veritat dintre de les moltes possibilitats possibles. Aquesta solució s'ha anomenat *funció lògica canònica*.

Per a resoldre circuits lògics combinacionals (que són aquells que la seva sortida depèn únicament del valor de les entrades en el moment en que es mira la sortida) amb portes lògiques, es segueix el següent procés de disseny:

1. Donat l'enunciat, es confecciona la taula de veritat on s'estableix, per a cada combinació possible de les entrades, l'estat de la sortida.

2. Un cop creada la taula de veritat podem obtenir la funció lògica canònica de dues maneres:

com a *addició de productes o minterms* o com a *multiplicació de sumes o maxterms*. Escollirem minterms o maxterms en funció del nombre de 1 o 0 que hagi a la taula.

Expressió lògica minterm

Aprofitem la propietat que per tenir un 1 lògic dins d'una taula de veritat totes les variables multiplicades han de ser 1. La suma de totes les multiplicacions ens donarà la taula de la veritat

Aquesta expressió lògica *minterm* s'obté de fer:

a) Ens fixem només en les combinacions d'entrades que en la taula de veritat donen un resultat 1.

b) Per a cadascuna d'aquestes combinacions escrivim les entrades multiplicades de la següent manera: Si la entrada val 1 escriurem directament l'entrada corresponent i si val 0 escriurem l'entrada negada

c) Sumar tots els productes de les diferents combinacions

Expressió lògica maxterm

Aprofitem la propietat que per tenir un 0 lògic dins d'una taula de veritat totes les variables sumades han de ser 0. La multiplicació de totes les sumes ens donarà la taula de veritat.

Aquesta expressió lògica *maxterm* s'obté de fer:

a) Ens fixem només en les combinacions d'entrades que en la taula de veritat donen un resultat 0.

b) Per a cadascuna d'aquestes combinacions escrivim les entrades sumades de la següent manera: Si l'entrada val 0 escriurem directament l'entrada corresponent, i si val 1 escriurem l'entrada negada.

c) multiplicar totes les sumes de les diferents combinacions anteriors.

El *maxterm* al nostre nivell és poc utilitzat, ja que normalment tindrem menys uns que zeros.

Entrades			Sortida	Minterms	Maxterms
a	**b**	**c**	*S*		
0	0	0	0		$a+b+c$

0	0	1	1	$\overline{a}\cdot\overline{b}\cdot c$	
0	1	0	0		$a+\overline{b}+c$
0	1	1	0		$a+\overline{b}+\overline{c}$
1	0	0	1	$a\cdot\overline{b}\cdot\overline{c}$	
1	0	1	1	$a\cdot\overline{b}\cdot c$	
1	1	0	0		$\overline{a}+\overline{b}+c$
1	1	1	0		$\overline{a}+\overline{b}+\overline{c}$

L'expressió lògica canònica la podem expressar llavors com:

Per minterms : $S=\overline{a}\cdot\overline{b}\cdot c+a\cdot\overline{b}\cdot\overline{c}+a\cdot\overline{b}\cdot c$

Per maxterms:

És evident que en aquest cas la funció per maxterms és més llarga que la obtinguda per minterms ja que en la taula hi ha 5 sortides que són "0", mentre que les sortides que donen "1" són 3. En cas de plantejar-nos en aquest cas concret quin mètode utilitzar per trobar una solució vàlida, hauríem d'escollir el mètode de minterms des del principi, ja que hi ha menys uns.

7.1. Simplificació de funcions

Les funcions canòniques (per minterms o maxterms) són fàcils de obtenir, però no són les funcions més simples possibles que poden donar la taula de veritat indicada. Podeu comprovar que una funció com $S=\overline{b}\cdot(a+c)$ realitza la mateixa taula de veritat de l'exemple anterior d'una forma més simplificada.
Existeixen llavors mètodes per a obtenir les funcions més simples possible, la qual cosa és necessària per implementar circuits digitals amb el mínim nombre de portes lògiques possibles i abaratir costos i espai.

Simplificació algebraica de funcions

Per simplificar funcions podem utilitzar el sistema algebraic que consisteix en aplicar les lleis i teoremes de l'àlgebra de Boole ja explicats:

Exemples

1.- *Simplifica aplicant l'àlgebra de Boole:*

$$F=\overline{a}\cdot\overline{b}\cdot c+a\cdot\overline{b}\cdot\overline{c}+a\cdot\overline{b}\cdot c=\overline{b}\cdot(\overline{a}\cdot c+a\cdot\overline{c}+a\cdot c)=\overline{b}\cdot(\overline{a}\cdot c+a)=\overline{b}\cdot(c+a)$$

2.- *Simplifica la següent equació, , aplicant el teorema de De Morgan.*

$$F=\overline{a}+\overline{b}+\overline{c}+\overline{(a\cdot b\cdot c)}=\overline{(a\cdot b\cdot c)}+\overline{(a\cdot b\cdot c)}=\overline{(a\cdot b\cdot c)}$$

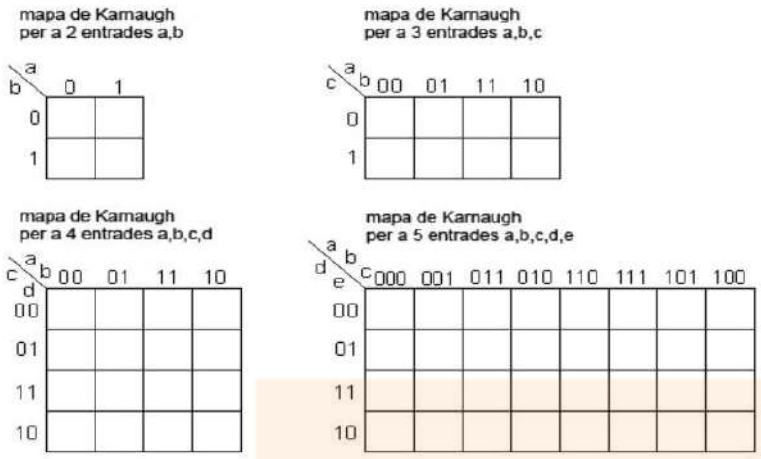
3.- *Simplifica la següent equació:*

$$F=\overline{a}\cdot\overline{b}+\overline{c}+\overline{a}\cdot(b+c)=\overline{a}\cdot\overline{b}+\overline{c}+\overline{a}\cdot b+\overline{a}\cdot c=\overline{a}\cdot(\overline{b}+b)+\overline{c}+\overline{a}\cdot c=\overline{a}+\overline{c}+\overline{a}\cdot c=\overline{a}\cdot(1+\overline{c})+\overline{c}=\overline{a}+\overline{c}$$

Simplificació de funcions pel mètode de Karnaugh

El mètode de simplificació de Karnaugh consisteix en escriure les taules de la veritat en un format de taula determinat, també anomenat mapes de Karnaugh, segons el nombre de entrades. Aquesta disposició especial permet identificar ràpidament els termes que es diferencien en una sola variable i poder així eliminar-la, obtenint com a resultat una simplificació de la funció.

Les taules que s'utilitzen depenen del nombre de variables de la funció, i són les següents:



Nosaltres utilitzarem només les de **2** i **3** variables d'entrada.

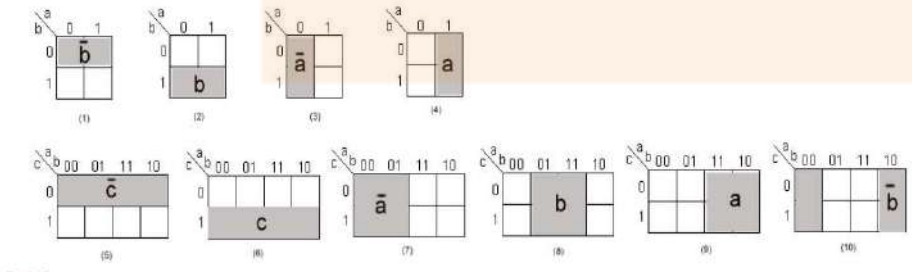
Els passos a seguir per a la simplificació són els següents:

1- El primer pas consistirà en moure els <1> de la taula de veritat a cada coordenada del mapa de Karnaugh corresponent a la combinació d'entrades que correspongui.

2- El segon pas serà agrupar tots els "1" en grups de 1, 2, 4, 8, 16,.. 2^n <1> contigus, amb les següents consideracions:

- Els "1" agrupats han de ser contigus en horitzontal o vertical, mai en diagonal.
- Quan més gran sigui el grup en que podem agrupar els "1" , més simplificada serà la funció resultant. Per tant hem de fer agrupacions sempre el més grans possible.
- Un "1" pot estar compartit per diferents grups, però no tots els "1" d'un grup poden pertànyer a altres grups.
- Els mapes de karnaugh son en realitat esferes per la qual cosa las caselles dels extrems també poden formar grups amb els uns de l'altre extrem.

3- El tercer pas és escriure les entrades que permeten obtenir cada grup dins de la taula. Per a cada grup escrivim les combinacions de les entrades que siguin comunes a totes les caselles del grup multiplicades de la següent manera: Si la entrada val 1 escriurem directament l'entrada corresponent i si val 0 escriurem l'entrada negada.



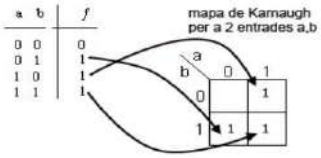
Taula 2

4- El quart i últim pas serà sumar les coordenades que identifiquen cada grup.

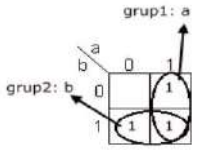
Tot això ho veurem millor a partir d'exemples concrets...

Exemples

1.- *Extreu la funció simplificada de la taula de la veritat de la següent figura:*



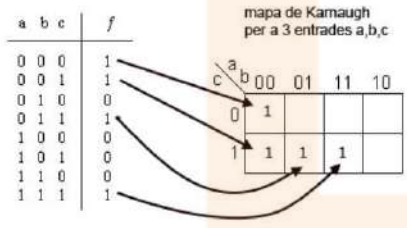
Traspassem els 3 "1" lògics de la taula al corresponent mapa de Karnaugh de 2 entrades. El primer "1" que correspon a a=0, b=1 l'escrivim a la primera columna i segona fila. El segon que correspon a a=1,b=0 l'escrivim a la segona columna i primera fila i el tercer "1" que correspon a a=1,b=1 l'escrivim a la segona columna i segona fila.



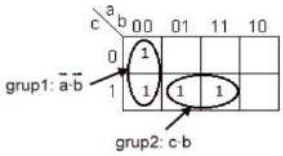
Una vegada passada la informació de la taula al mapa de Karnaugh, observem que podem agrupar els "1" en dos grups de 2 elements (un 1 queda encabit en els dos grups). El primer grup ocupa la segona columna, la qual queda identificada per a=1 (i que és la zona "a" de la taula2-fig 4) i el segon grup ocupa tota la segona fila, la qual queda identificada per b=1 (i que és la zona "b" de la taula 2 -fig 2).

Una vegada identificats els grups, la funció resultant serà la suma de les coordenades de tots els grups, en aquest cas: $F = grup1 + grup2 = a + b$

2.- *Extreu la funció simplificada de la taula de la veritat de la següent figura:*



Traspassem els 4 "1" lògics que hi ha a la taula a les coordenades corresponents del mapa de Karnaugh. El primer "1" el tenim quan les entrades són a=0 b=0 c=0 i el posem a la primera fila i columna (coordenada "00" de ab i "0" de c). El segon "1" lògic el tenim quan les entrades són a=0 b=0 c=1 i el posem a la segona fila i primera columna (coordenada "00" de ab i "1"de c). El tercer "1" lògic el tenim quan les entrades són a=0 b=1 c=1 i el posem a la segona fila i segona columna (coordenada "01" de ab i "1"de c) i el quart "1" lògic el tenim quan les entrades són a=1 b=1 c=1 i el posem a la segona fila i tercera columna (coordenada "11" de ab i "1"de c).



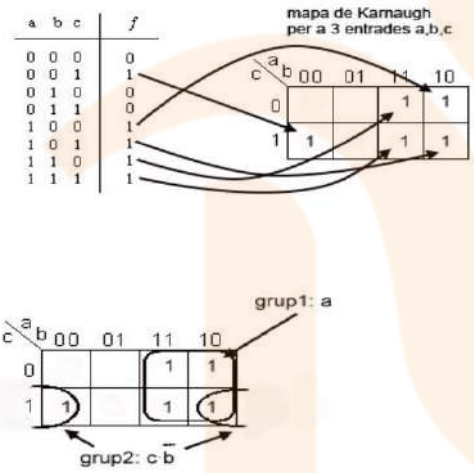
Seguidament agrupem els "1" en grups el més gran possible. Veiem que poder fer 2 grups de 2 "1" cadascun i que no podem fer cap grup de quatre (no es poden fer grups de 3!). El primer grup ocupa tota la primera columna i aquesta columna queda definida per a=0 b=0, i desapareix c. També podem observar que aquest grup és la intersecció de la zona $\bar{a}$ (Taula 2, fig 7) amb la zona $\bar{b}$ (Taula 2, fig 10). Per tant el grup queda definit per el producte $\bar{a}\cdot\bar{b}$.

El segon grup ocupa la segona fila, amb la intersecció de la segona i tercera columna. La segona fila queda definida per que c=1, mentre que la segona i tercera fila tenen en comú que en les dos el valor de b=1 , independentment de a. També podem observar que aquest grup és la intersecció de la zona c (Taula 2, fig 6) amb la zona b (Taula 2, fig 8). Per tant el grup queda definit per el producte $c\cdot b$.

Com que tenim les coordenades de cada grup, el resultat final de la funció lògica serà la suma de les coordenades de cada grup:

$S=\bar{a}\cdot\bar{b}+c\cdot b$

3.- *Extreu la funció simplificada de la taula de la veritat de la següent figura:*



Traspassem els "1" de la taula al mapa de Karnaugh de 3 entrades i agrupem els "1" en grups el més gran possibles.

Veiem que poder fer 2 grups, un primer grup de 4 elements i un segon grup de 2 elements (recordeu que les cel·les dels extrems es comuniquen entre elles com si fos un cilindre). Els quatre "1" del primer grup tenen en comú que a=1, independentment del valor de b i c i que corresponen a la zona "a" de la Taula2 Fig 9. El segon grup està en la segona fila identificada per c=1 i en la primera i quarta columna, les quals tenen en comú que en elles b=0. Aquest segon grup és la intersecció de la zona "c" (Taula 2 Fig 6) i la zona "b negada" (Taula 2 Fig 10), és a dir, que és $c\cdot\bar{b}$.

Com que tenim les coordenades de cada grup, el resultat final de la funció lògica serà la suma de les coordenades de cada grup:

$S=a+c\cdot\bar{b}$

7.2. Lleis de morgan

La negació d'una suma lògica de dos elements és igual al producte de les negacions dels elements.

$$\overline{a+b} = \overline{a} \cdot \overline{b}$$

La negació d'un producte lògic de dos elements és igual ala suma de les negacions dels elements.

$$\overline{a \cdot b} = \overline{a} + \overline{b}$$

Aquestes lleis s'utilitzen per poder canviar el tipus de les funcion.

Exemple
Transformació d'una funció(1) expressada com a 3 portes OR en portes NAND utilitzant les lleis de Morgan
(2) Neguem dos cops la 1a i la 3a. Al negar dos cops la la funció no varia

(3) Convertim els + en · dividint les negacions

(4) Trèiem les dobles negacions

(5) Neguem dos cops tota la funció.

(6) Convertim els + en · dividint la negació

(7) Trèiem les dobles negacions


Transformacios de Morgan

En aquest resultat no s'ha fet cap mena de simplificació