Funzioni e porte logiche

- Creatore originale: @LucaCaffa
  - @Giacomo Dandolo (14/04/2025): aggiunta nomenclatura per le funzioni logiche e per l'algebra di Boole.

Grandezze elettriche

Utilizziamo gli stati logici 1 e 0 per rappresentare il comportamento delle grandezze elettriche. La convenzione maggiormente usata è la seguente:

- Tensione alta → 1 logico;
- Tensione bassa → 0 logico.

Threshold

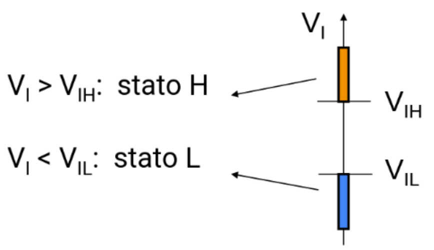
In un modello reale, una tensione non è mai costante, ma spesso presenta dei rumori (ha diverse oscillazioni), quindi non basta considerare tensione bassa e alta.

Si introduce il concetto di threshold (soglia), definito attraverso una tensione di soglia chiamata V_T. Ipotizzando di avere una generica tensione V, si ha che:

- se $V > V_\text{T}$ → 1 logico;
- se $V < V_\text{T}$ → 0 logico.

Lo stato logico viene trovato confrontando la tensione di ingresso V_I con quella di soglia V_T. La soglia non è un valore fisso, ma un range $V_\text T = [V_\text{IL}, V_\text{IH}]$, dove:

- se $V_\text{I} > V_\text{IH}$ → 1 logico;
- se $V_\text{I} < V_\text{IL}$ → 0 logico;
- la tensione tra V_IL e tra V_IH non è definita.



Rappresentazione del concetto di threshold

Funzioni logiche

Funzione logica NOT

La funzione logica NOT indica la negazione di un ingresso, quindi se in ingresso abbiamo 0, in uscita avremo 1 e viceversa.

$$y = \text{NOT } x = \overline{x}$$

La dicitura descritta indica che y è l'inverso di x.

Le varie notazioni sono:

xxx	yyy
0	1
1	0

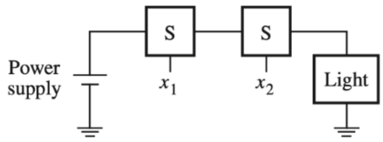
NOT x=x‾=x′= !x= ~x=x*\text{NOT } x = \overline x = x' = \ !x = \ \sim x = x^*NOT x = x = x′ = !x = ~ x = x*

> □□
> E' importante notare che la funzione logica NOT può essere applicata a ogni funzione logica, e che ogni funzione logica negata ha la tabella di verità con valori di uscita opposti alla funzione non negata.

Funzione logica AND

Nella figura:

- "Power Supply" è l'alimentatore;
- x1x_1x1 e x2x_2x2 sono due interruttori (Switch);
- Light è la lampadina, che rappresenta la nostra uscita yyy.



(a) The logical AND function (series connection)

Rappresentazione di un circuito che implementa la funzione logica AND

Per far accendere la lampadina abbiamo bisogno che gli switch siano entrambi chiusi, in modo da far arrivare la tensione alla lampadina.

Questo concetto si traduce con la seguente funzione logica AND:

f(x1,x2)=y=x1 AND x2=x1⋅x2=x1x2f(x_1, x_2) = y = x_1 \text{ AND } x_2 = x_1 \cdot x_2 = x_1x_2f(x1,x2) = y = x1 AND x2 = x1 ⋅ x2 = x1x2

Questa funzione dice che y=1y=1y = 1 solo se x1x_1x1 e x2x_2x2 sono entrambi 1.

x1x_1x1	x2x_2x2	yyy
0	0	0
0	1	0
1	0	0
1	1	1

Funzione logica NAND

La funzione logica AND negata si chiama NAND, cioè "NOT AND".

f(x1,x2)=y=x1 NAND x2=x1⋅x2‾f(x_1, x_2) = y = x_1 \text{ NAND } x_2 = \overline{x_1 \cdot x_2}f(x1,x2) = y = x1 NAND x2 = x1 ⋅ x2

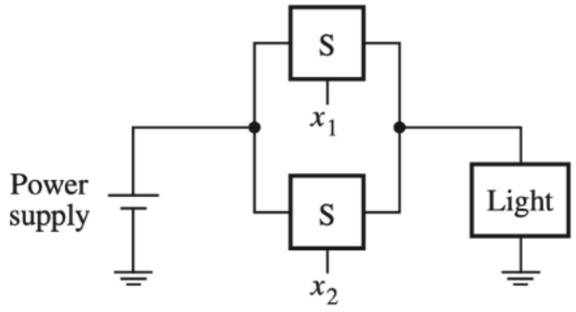
x1x_1x1	x2x_2x2	yyy
0	0	1
0	1	1
1	0	1

x1x_1x1	x2x_2x2	yyy
1	1	0

Funzione logica OR

Nella figura:

- “Power Supply” è l’alimentatore;
- x1x_1x1 e x2x_2x2 sono due interruttori (Switch);
- Light è la lampadina, che rappresenta la nostra uscita yyy.



(b) The logical OR function (parallel connection)

Rappresentazione di un circuito che implementa la funzione logica OR

Per far accendere la lampadina basta che uno dei due interruttori sia chiuso, in modo da far passare la tensione.

Questo concetto si traduce con la seguente funzione logica OR:

f(x1,x2)=y=x1 OR x2=x1+x2f(x_1, x_2) = y = x_1 \text{ OR } x_2 = x_1 + x_2f(x1,x2) = y = x1 OR x2 = x1 + x2

Questa funzione dice che y=1y=1y = 1 se x1x_1x1 o x2x_2x2 valgono 1.

x1x_1x1	x2x_2x2	yyy
0	0	0
0	1	1
1	0	1
1	1	1

Funzione logica NOR

La funzione logica OR negata si chiama NOR, cioè “NOT OR”.

f(x1,x2)=y=x1 NOR x2=x1+x2‾f(x_1, x_2) = y = x_1 \text{ NOR } x_2 = \overline{x_1 + x_2}f(x1,x2) = y = x1 NOR x2 = x1 + x2

x1x_1x1	x2x_2x2	yyy
0	0	1
0	1	0
1	0	0
1	1	0

Funzione logica XOR

La funzione logica XOR corrisponde ad un OR esclusivo, in cui l'uscita è 1 se uno e uno solo degli ingressi vale 1.

f(x1,x2)=y=x1 XOR x2=x1⊕x2f(x_1, x_2) = y = x_1 \text{ XOR } x_2 = x_1 \oplus x_2f(x1, x2) = y = x1 XOR x2 = x1 ⊕ x2

x1x_1x1	x2x_2x2	yyy
0	0	0
0	1	1
1	0	1
1	1	0

Funzione logica XNOR

La funzione logica XNOR corrisponde ad uno XOR negato, in cui l'uscita è 1 se gli ingressi sono uguali.

f(x1,x2)=y=x1 XNOR x2=x1⊕x2¯f(x_1, x_2) = y = x_1 \text{ XNOR } x_2 = \overline{x_1 \oplus x_2}f(x1, x2) = y = x1 XNOR x2 = x1 ⊕ x2

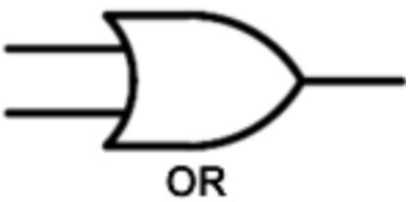
Lo XNOR può anche essere trovato, nella letteratura, come NXOR, EXNOR, ENOR o XAND.

x1x_1x1	x2x_2x2	yyy
0	0	1
0	1	0
1	0	0
1	1	1

Porte logiche

Le porte logiche sono dei circuiti che contengono transistori connessi in modo da comportarsi come una funzione logica. Le porte logiche più utilizzate sono quelle a due ingressi, ma ne esistono anche a più di due ingressi.

Porta logica AND



Porta logica OR

NOT

Porta logica NOT

NAND

Porta logica NAND

NOR

Porta logica NOR

XOR

Porta logica XOR

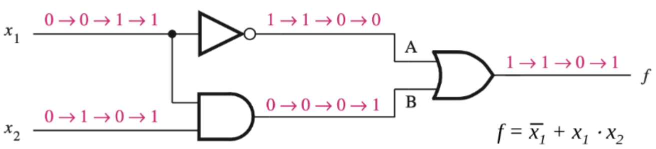
NXOR

Porta logica XNOR

Reti Logiche

Una combinazione di porte logiche crea una “rete logica”, anche chiamato “circuito logico”.

Possiamo analizzare il comportamento di una rete logica per determinarne la funzione e determinare l’uscita in base agli ingressi.



x_1	x_2	$f(x_1, x_2)$	A	B
0	0	1	1	0
0	1	1	1	0
1	0	0	0	0
1	1	1	0	1

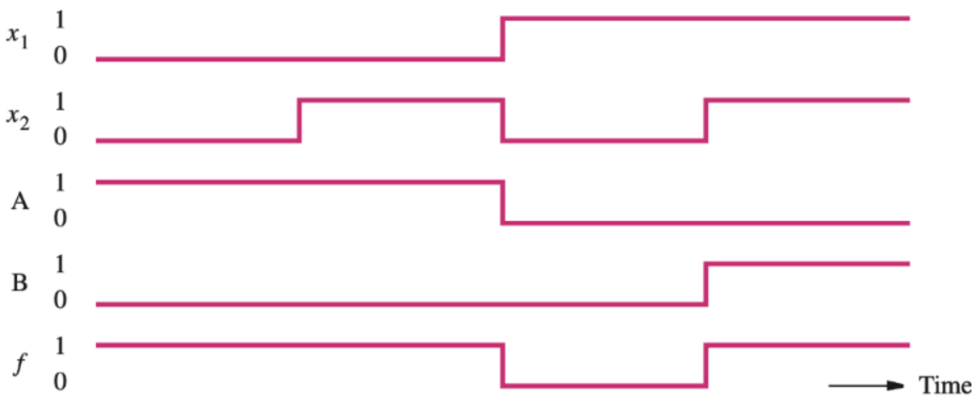
Esempio di rete logica con tabella di verità che descrive il comportamento del circuito logico

Diagramma temporale

Una volta analizzato il circuito logico, possiamo trovarne il diagramma temporale, disegnando le variazioni delle forme d'onda per ogni variabile, in cui gli ingressi sono definiti in maniera arbitraria o attraverso delle letture sul circuito.

□□

I diagrammi temporali sono molti importanti, poiché sono il modo in cui rappresentiamo i segnali elettrici. Oltre ad indicare i livelli logici (1 o 0), si possono indicare i tempi di transizione ed un eventuale presenza di rumore.



Esempio di diagramma temporale (comportamento ideale)

Algebra di Boole

L'algebra di Boole definisce assiomi, teoremi e proprietà legati alle funzioni logiche, permettendo la semplificazione di funzioni logiche definite.

Assiomi

Gli assiomi sono delle equivalenze che permettono di semplificare le funzioni logiche definite e di definire le dimostrazioni dei teoremi, che non verranno mostrate in queste dispense.

0·0=00 \cdot 0 = 00 · 0 = 0

0+0=00 + 0 = 00 + 0 = 0

1·1=11 \cdot 1 = 11 · 1 = 1

1+1=11 + 1 = 11 + 1 = 1

0·1=1·0=00 \cdot 1 = 1 \cdot 0 = 00 · 1 = 1 · 0 = 0

1+0=11 + 0 = 11 + 0 = 1

Teoremi

Definiamo i teoremi della funzione logica AND.

x·0=0x \cdot 0 = 0x · 0 = 0

x·1=xx \cdot 1 = xx · 1 = x

x·x=xx \cdot x = xx · x = x

Definiamo i teoremi della funzione logica OR.

x+0=xx + 0 = xx + 0 = x

x+1=1x + 1 = 1x + 1 = 1

x+x=xx + x = xx + x = x

Definiamo i teoremi di esistenza del complemento.

x⋅x¯=0x \cdot \overline x = 0x ⋅ x = 0

x+x¯=1x + \overline x = 1x + x = 1

Proprietà

Definiamo le proprietà commutative.

x+y=y+xx + y = y + xx + y = y + x

x⋅y=y⋅xx \cdot y = y \cdot xx ⋅ y = y ⋅ x

Definiamo le proprietà associative.

x+(y+z)=(x+y)+zx + (y+z) = (x+y) + zx + (y + z) = (x + y) + z

x⋅(y⋅x)=(x⋅y)⋅zx \cdot (y \cdot x) = (x \cdot y) \cdot
zx ⋅ (y ⋅ x) = (x ⋅ y) ⋅ z

Definiamo le proprietà di assorbimento.

x+(x⋅y)=xx + (x \cdot y) = xx + (x ⋅ y) = x

x+x¯⋅y=x+yx + \overline x \cdot y = x + yx + x ⋅ y = x + y

x⋅y+x⋅y¯=xx \cdot y + x \cdot \overline y = xx ⋅ y + x ⋅ y = x

x⋅(x+y)=xx \cdot (x + y) = xx ⋅ (x + y) = x

x⋅(x¯⋅y)=x⋅yx \cdot (\overline x \cdot y) = x \cdot
yx ⋅ (x ⋅ y) = x ⋅ y

(x+y)⋅(x+y¯)=x(x+y) \cdot (x + \overline y) =
x(x + y) ⋅ (x + y) = x

Definiamo le proprietà distributive.

x⋅(y+z)=x⋅y+x⋅zx \cdot (y + z) = x \cdot y + x \cdot
zx ⋅ (y + z) = x ⋅ y + x ⋅ z

x+(y⋅z)=(x+y)⋅(x+z)x + (y \cdot z) = (x+y) \cdot
(x+z)x + (y ⋅ z) = (x + y) ⋅ (x + z)

Definiamo le proprietà di idempotenza.

x+x=xx + x = xx + x = x

x⋅x=xx \cdot x = xx ⋅ x = x

Leggi di De Morgan

Le leggi di De Morgan forniscono delle equivalenze tra le funzioni logiche OR e AND.

$$\overline{x\cdot y} = \overline x + \overline y$$

$$\overline{x+y} = \overline x \cdot \overline y$$