Algebra di Boole

Le reti logiche

- **Tutte le informazioni trattate finora sono codificate tramite stringhe di bit**
- **Le elaborazioni da compiere su tali informazioni consistono nel costruire, a partire da determinate configurazioni di bit, altre configurazioni che, nella codifica prefissata, rappresentano i risultati richiesti**
- **I circuiti elettronici che realizzano tali operazioni sono detti *circuiti di commutazione* (*switching circuits*) o *reti logiche***

Progetto di reti logiche

- **Il progetto delle reti logiche si svolge in primo luogo tenendo conto delle funzionalità del circuito, indipendentemente dalla realizzazione fisica (progetto logico)**
- **Ciò consente:**
  - di prescindere dai particolari realizzativi
  - di risolvere a livello logico eventuali problemi implementativi
- **Strumento fondamentale: l'algebra di Boole**

L'algebra di Boole

- **Consente di descrivere in forma algebrica le funzioni dei circuiti**
- **Fornisce dei metodi per l'analisi e la sintesi (a livello logico) dei circuiti**
- **Tramite l'algebra di Boole si stabilisce una corrispondenza biunivoca tra**
  - operazioni dell'algebra e componenti elementari
  - espressioni algebriche e circuiti

L’algebra di Boole

- **Nel progetto delle reti logiche si impiega un sistema algebrico in cui ogni variabile può assumere solo uno tra due valori: 0 e 1**
- **Sulle variabili si applicano le operazioni:**
  - prodotto logico (*) o AND
  - somma logica (+) o OR
  - negazione (!) o NOT

Operazioni binarie (AND, OR)

Operazioni unaria (NOT)

AND	OR	NOT
0*0=0	0+0=0	!0=1
0*1=0	0+1=1	!1=0
1*0=0	1+0=1	
1*1=1	1+1=1	

Proprietà dell'algebra di Boole

Commutativa:	a+b=b+a	a*b=b*a
Associativa:	(a+b)+c=a+(b+c)	(a*b)*c=a*(b*c)
Idempotenza:	(a+a)=a	(a*a)=a
Assorbimento:	a+(a*b)=a	a*(a+b)=a
Distributiva:	a*(b+c)=a*b+a*c	a+(b*c)=(a+b)*(a+c)
Min e max:	a*0=0	a+1=1
Elem.to neutro:	a+0=a	a*1=a
Complemento:	a*(!a)=0	a+(!a)=1
De Morgan:	!(a+b)=!a*!b	!(a*b)=!a+!b

Funzioni logiche

- **Una variabile può essere definita come funzione di altre variabili:**
  w=f(x,y,z)
- **Si dicono *funzioni logiche elementari* le funzioni:**
  z=x*y (funzione AND)
  z=x+y (funzione OR)
  y=!x (funzione NOT)
- **Quante sono le possibili funzioni in 2 variabili ?**

x	y	f(x,y)															
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

AND XOR OR EQU

Funzioni ed espressioni

Una funzione logica può essere definita, oltre che in forma tabellare (tabella di verità), tramite espressioni algebriche

Esempio:

f = x+y*!z+!y*z

f = x*!z+x*y+y*!z+!y*z

Espressioni equivalenti

Come passare dall'una all'altra ?

x	y	z	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Letterali, mintermini, maxtermini

Letterale: variabile affermata o negata

Termine: prodotto o somma di letterali

Mintermine: prodotto di letterali di tutte le variabili di una certa funzione

Maxtermine: somma di letterali di tutte le variabili di una certa funzione

Esempio

Mintermine: x!yz

Maxtermine: !x+y+z

x	y	z	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Forme canoniche

Una funzione definita tramite tabella di verità può essere espressa algebricamente in due diverse forme canoniche:

Somma di mintermini

f = !x!yz+!xy!z+x!y!z+x!yz+xy!z+xyz

Prodotto di maxtermini

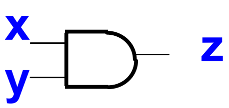
f =(x+y+z)(!x+y+z)

x	y	z	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

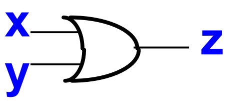
Equivalenza con i circuiti logici

Esiste una equivalenza tra le funzioni logiche e le porte elementari delle reti logiche (Shannon)

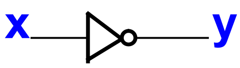
z=x*y ⟷



z=x+y ⟷



y=!x ⟷

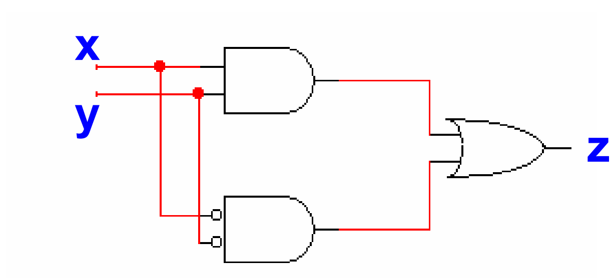


Equivalenza con i circuiti logici

L'equivalenza si estende alle espressioni ed ai circuiti

x	y	z
0	0	1
0	1	0
1	0	0
1	1	1

z = xy+!x!y



Minimizzazione delle funzioni logiche

- **Ad una funzione descritta tramite tabella di verità possono essere associate più espressioni algebriche. Quale scegliere ?**
- **Vista l'equivalenza con i circuiti, conviene scegliere l'espressione corrispondente al circuito a minimo costo (→ minimizzazione)**
- **Il costo può esprimersi in base a:**
  - numero di porte
  - numero di ingressi
  - eterogeneità delle porte

Minimizzazione delle funzioni logiche

- **I metodi per la minimizzazione si basano sulle proprietà dell'algebra di Boole.**
  Esempio:

f = !x!yz+!xy!z+x!y!z+x!yz+xy!z+xyz

x!y!z+x!yz = x!y(!z+z) = x!y
xy!z+xyz = xy(!z+z) = xy
x!yz+xyz = xz(!y+y) = xz
x!y!z+xy!z = x!z(!y+y) = x!z

x!y+xy = x(!y+y) = x

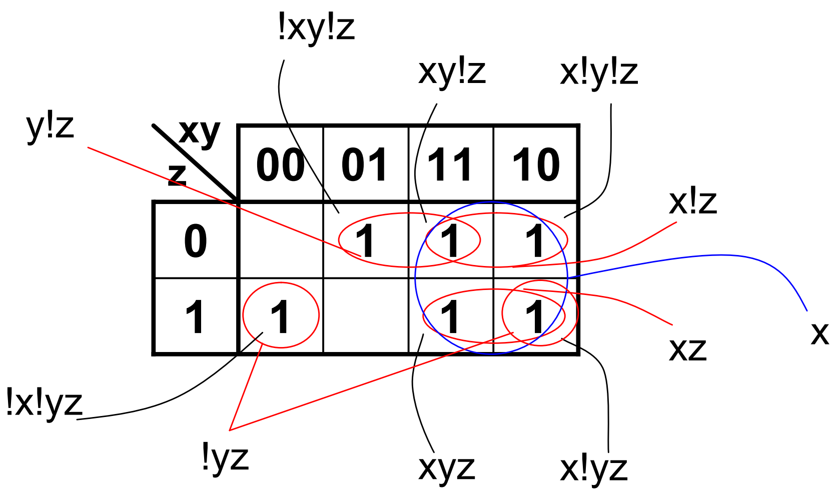
xz+x!z = x(z+!z) = x

!x!yz+x!yz = !yz(!x+x) = !yz
!xy!z+xy!z = y!z(!x+x) = y!z

Forma minima:
f = x+!yz+y!z

Le mappe di Karnaugh

- Due mintermini si dicono adiacenti se differiscono in un solo letterale.
- Le mappe di Karnaugh sono una rappresentazione grafica che evidenzia l'adiacenza tra mintermini



x	y	z	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Forma minima:
f = x+!yz+y!z

Funzioni non completamente specificate

Si verificano quando ci sono combinazioni delle variabili di ingresso che non sono possibili o, in corrispondenza delle quali, il valore di uscita non è influente.

z \ xy	00	01	11	10
0	1	1	x	1
1	1	x		x

x = "don't care"

x	y	z	f
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	x
1	0	0	1
1	0	1	x
1	1	0	x
1	1	1	0

Ai fini del progetto, i valori don't care possono essere specificati in modo da minimizzare l'espressione della funzione

Funzioni non completamente specificate

Le soluzioni ottenibili sono diverse. La scelta va fatta sulla base delle specifiche del progetto e sulla convenienza complessiva

z \ xy	00	01	11	10
0	1	1	x	1
1	1	x		x

⇨ **f = !x+!y**

z \ xy	00	01	11	10
0	1	1	x	1
1	1	x		x

⇨ **f = !y+!z**

Fasi del progetto di una rete logica

1. **Definizione delle specifiche**
   - Identificazione delle variabili in ingresso e in uscita
2. **Definizione della tabella di verità della funzione**
3. **Minimizzazione**
4. **Definizione del circuito**