Funzioni booleane

Vitoantonio Bevilacqua

bevilacqua@poliba.it

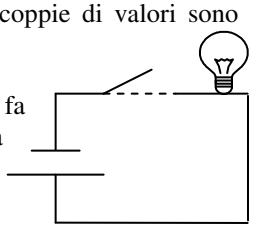
Sommario. Il presente paragrafo si riferisce alle lezioni del corso di Fondamenti di Informatica e Laboratorio di Informatica dei giorni 9 marzo e 10 marzo 2010.

Parole chiave: Variabile booleana, Tavola di Verità, Forma Canonica, Forma sintetizzata, Logic gate

1 Introduzione

Il matematico George Boole nel 1954 pubblicò un lavoro circa un algebra sulle relazioni logiche, nel quale investigava sulle proprietà dei sistemi binari. Egli postulò una tabella di associazione tra variabili logiche, tabella nella quale 4 possibili coppie di valori sono associate tra loro tramite una condizione logica (es: OR o AND)

Dato un circuito, il concetto di circuito aperto - chiuso che fa – non fa passare corrente ha a che fare con un concetto booleano di sistema binario: si presentano infatti due casi, vero o falso, passa o non passa corrente, c'è o non c'è tensione.



2 Funzione Booleana

Una funzione F=F(A,B,C,D) con A,B,C,D variabili indipendenti, F variabile dipendente, si dice booleana di variabili booleane se e solo se ciascuna delle variabili indipendenti può avere solo valore 0 o 1, quindi ha dominio {0;1} e la variabile dipendente può anch'essa avere solo valore 0 o 1. Quindi il codominio della funzione è {0;1}.

Poiché ogni variabile è definita nell'insieme {0;1}, le 4 variabili sono definite nel prodotto cartesiano di {0;1}×{0;1}×{0;1}×{0;1} (il prodotto cartesiano, date le 4 variabili booleane, rappresenta le 2^4=16 combinazioni che possono assumere le variabili); quindi il dominio è pensato come l'insieme delle 16 configurazioni che possono assumere le variabili A,B,C,D.

2.1 La tavola di verità

Tramite la tavola della verità si rappresenta il dominio e codominio della funzione booleana, esplicitando il prodotto cartesiano, scrivendo le 16 possibili combinazioni che possono assumere le variabili A,B,C,D (ciò equivale al conteggio che va da 0 a 15 in binario) e il valore della F che questa assume in corrispondenza di ogni combinazione.

Esempio di funzione booleana: F sarà vera se

- A,B,C,D sono false
- A,B,C,D sono vere
- A,C,D sono vere e B è falsa

A	B	C	D	**F**
0	0	0	0	**1**
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	**1**
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	**1**

Dunque scritta in questo modo, la F sarà una delle possibili 2^{16}=65536 funzioni che si possono implementare.

2.2 La forma canonica

Per ottenere una funzione analitica che dia la funzione descritta nella tavola della verità, si applica un criterio detto "sintesi di una funzione booleana": data una tavola della verità, la F è pensata come somma logica[1] di tanti termini quanti sono gli 1 della colonna F; ciascun contributo è dato dal prodotto logico di tutte le variabili e per far si che ogni prodotto logico valorizzi la sequenza corrispondente, le singole variabili fattori del prodotto logico devono essere "complementate" (ossia devono essere negate) nel caso assumano in quella sequenza il valore 0. Tale funzione si dice che è scritta in forma "canonica"

Esempio: F = ABCD + AB'CD + A'B'C'D' [2]

[1] la sintesi di una funzione booleana si può anche intendere come prodotto logico di somme logiche

[2] per indicare il complementare di una variabile, si utilizza la notazione "Ā"; tuttavia per comodità si intenderà la notazione "A' " equivalente alla complementazione

2.3 La logica AOI

Per definire le operazioni di somma logica, prodotto logico e complementazione, si definisce la Logica AOI (And, Or, Inverter):

- "+": per somma logica in algebra booleana si intende l'operatore OR; è una funzione duale (necessita di due variabili), definita sul prodotto cartesiano delle due variabili argomento, assegnata vera quando almeno una delle due variabili sono vere

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

- "·": per prodotto logico in algebra booleana si intende l'operatore AND; è una funzione duale definita sul prodotto cartesiano delle due variabili argomento, assegnata vera solo quando entrambe le variabili sono vere

A	B	AB
0	0	0
0	1	0
1	0	0
1	1	1

- " ¯ ": per complementazione si intende l'operatore NOT; è una funzione che ha bisogno di una sola variabile, definita sulle due possibili configurazioni di quella variabile. È vera quando la variabile è falsa, è falsa quando la variabile è vera

A	$\bar{A}$
0	1
1	0

Esempio: verificare che la sequenza 0110 dia, sostituita nella forma canonica, il valore 0 assunto dalla funzione nella tavola di verità in corrispondenza della sequenza:
0'1'1'0'+01'10+0110=1001+0010+0110=0+0+0=0

2.4 Proprietà dell'algebra booleana

Data la variabile A:

1. A''=A
2. A+A'=1
3. AA'=0
4. A+1=1
5. A1=A

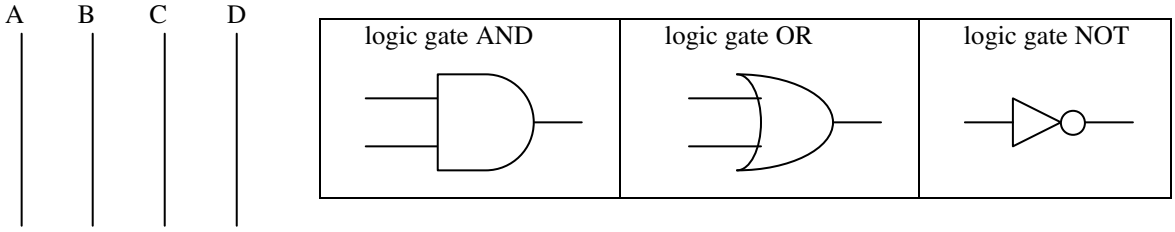
Date le variabili A e B:

6. Proprietà di "assorbimento": A·(A+B)=A; A+A·B=A
7. Identità di De Morgan:
   a. la somma dei negati è equivalente al negato del prodotto
      A'+B'=(AB)'; (A'+B')'=AB
   b. il prodotto dei negati è equivalente al negato della somma
      A'B'=(A+B)'; (A'B')'=A+B

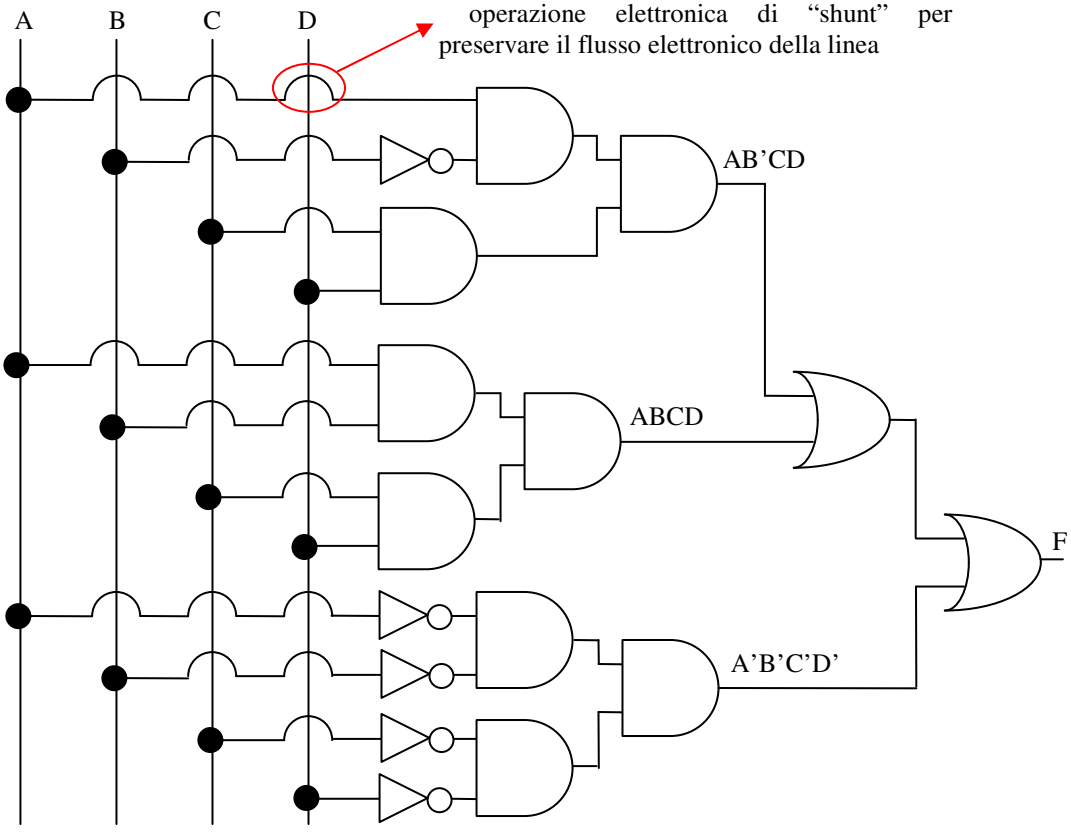
3 Implementazione di un circuito logico booleano

Si può instaurare una corrispondenza tra la tavola di verità vista in precedenza e il comportamento di un circuito creato nel seguente modo:

- ad ogni variabile corrisponde una linea elettrica su cui può viaggiare un segnale elettrico
- ad ogni operatore logico corrisponde un "logic gate" = un dispositivo elettronico formato da una o due linee elettriche d'ingresso, se rispettivamente si tratta di AND e OR o di NOT (in realtà può anche essere formato da più di due linee d'ingresso ma in quel caso il comportamento rimane analogo) e da una linea elettrica d'uscita su cui può viaggiare un segnale elettrico a seconda del risultato dell'operazione booleana eseguita dal dispositivo

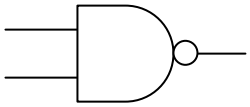


Esempio di implementazione di un circuito booleano secondo una funzione assegnata utilizzando i logic gate AND, OR e NOT:



L'intera funzione ha un costo di 2OR + 9AND + 5NOT = 16AOI. Tuttavia il numero di NOT si considera ininfluente nel calcolo del costo, che quindi diventa di 11AO.

3.1 La porta NAND



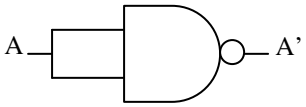
Per semplificare il tipo di porte logiche, si introduce la porta NAND: un logic gate che opera un AND e ne complementa l'output.

La funzione NAND è definita dalla seguente tavola di verità:

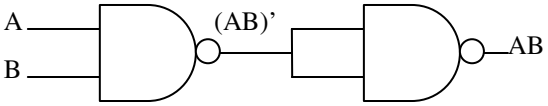
A	B	(AB)'
0	0	1
0	1	1
1	0	1
1	1	0

Utilizzando le porte NAND opportunamente, si possono ottenere comportamenti equivalenti alle altre tre porte:

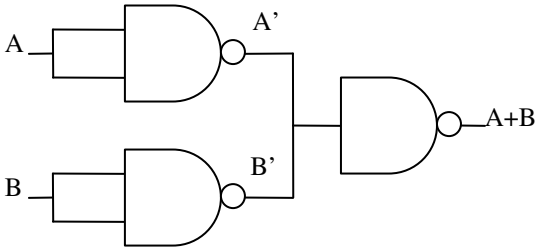
- per ottenere un comportamento equivalente alla NOT si usa una NAND i cui ingressi sono collegati alla stessa linea elettrica della variabile da complementare



- per ottenere una AND si utilizzano due NAND in successione



- per ottenere una OR si utilizza la II identità di De Morgan



4 Metodo delle mappe di Karnaugh

Il metodo delle mappe di Karnaugh è un procedimento utilizzato per minimizzare una forma canonica di una funzione booleana con un massimo di 6 variabili, in modo che costi meno.

4.1 Mappa di Karnaugh

Una mappa di Karnaugh esplicita il dominio della funzione booleana in forma matriciale, ponendo come riferimento orizzontale il valore della coppia delle prime due variabili nella tabella di verità (ed: A e B) e come verticale quello delle ultime due (es: C e D). Esempio:

AB/ CD	00	01	11	10
00	**1**	0	0	0
01	0	0	0	0
11	0	0	**1**	**1**
10	0	0	0	0

Concetto di adiacenza geometrica: si intende una relazione che lega le cifre immediatamente vicine orizzontalmente e verticalmente, tenendo anche conto della cilindricità orizzontale e verticale della mappa di Karnaugh, che rende adiacenti tra loro la cifra di un estremo con quella dell'estremo opposto.

Concetto di adiacenza logica: si intende una relazione che lega le cifre le cui coordinate differiscano per un solo valore; tale differenza si definisce "distanza di Hamming" e dovrà essere unitaria

I riferimenti orizzontali e verticali della matrice vanno dunque disposti nell'ordine 00 01 11 10 affinché l'adiacenza geometrica tra gli elementi della matrice coincida con l'adiacenza logica. Ciò sarà indispensabile per il passo successivo:

3.2 Algoritmo di Karnaugh

La minimizzazione della forma canonica richiede i seguenti passaggi:

1. si individuano nella mappa di Karnaugh gli 1 isolati (che non risultano avere altri 1 adiacenti). Ogni 1 isolato va sintetizzato senza semplificazioni
2. si individuano gli 1 adiacenti prendibili con un solo raggruppamento rettangolare di 2; si sintetizza ogni raggruppamento eliminando la variabile che cambia comportamento
3. si individuano gli 1 adiacenti prendibili con un solo raggruppamento rettangolare di 4; si sintetizza ogni raggruppamento eliminando le due variabili che cambiano comportamento
4. si individuano gli 1 adiacenti prendibili con un solo raggruppamento rettangolare da 8; si sintetizza ogni raggruppamento eliminando le tre variabili che cambiano comportamento [3]
5. si individuano gli 1 non raggruppati adiacenti ad altri 1 raggruppati; si considera per ciascuno un raggruppamento rettangolare più grande possibile (o uno a caso tra i raggruppamenti più grandi possibili, se ve ne è più di uno; questo perché la scelta avrebbe solo effetto sulla minimizzazione delle porte NOT che però non ci interessano); si sintetizza ogni raggruppamento eliminando le variabili che cambiano comportamento
6. si sommano i contributi derivanti da ciascun passaggio

Esempio: nella mappa di Karnaugh precedentemente scritta si nota che

1. è presente un **1** isolato; sintetizzandolo si ottiene A'B'C'D' che essendo isolato non può essere minimizzato
2. sono presenti due **1** adiacenti prendibili da un unico raggruppamento rettangolare; sintetizzando l'intero raggruppamento si ottiene ACD, in quanto la variabile B cambia comportamento e si elimina

Dunque la forma minimizzata della funzione F di partenza è F=A'B'C'D'+ACD con un costo di 1 OR + 5AND = 6AO

[3] se gli 1 adiacenti sono prendibili con un unico raggruppamento da 16, la mappa di Karnaugh sarà costituita da soli 1, cadono tutte le variabili e la funzione sarà costante e sempre vera, F=1

Infine per minimizzare il costo delle porte NOT, queste vengono rese disponibili sin dall'inizio per tutte le linee del circuito in modo da essere utilizzate quando lungo il circuito ne viene richiesto l'uso, anziché inserirne una ogni volta.

Ringraziamenti. Il presente capitolo è stato scritto anche grazie al prezioso contributo degli studenti Donato Mancuso, Giuseppe Ragno, Michele Masciavè, Michele Fioriello, Antonio Di Maio.

Riferimenti

1. Bevilacqua, V.: Dispense Teoria 2 e Karnaugh In: http://www.vitoantoniobevilacqua.it
2. http://it.wikipedia.org/wiki/Algebra_booleana
3. http://it.wikipedia.org/wiki/Mappa_di_Karnaugh