Circuiti combinatori

Andrea Canale

December 23, 2024

Contents

1 Porte logiche

2 Algebre di Boole

Dato un insieme S contenente elementi distinti 0 e 1 e due operazioni binarie + e ·, questo insieme è un algebra di Boole se valgono le seguenti proprietà:

- Associatività: $(x + y) + z = x + (y + z)$ e $(x \cdot y) \cdot z = x \cdot (y \cdot z)$
- Commutatività: $x + y = y + x$ e $x \cdot y = y \cdot x$
- Distributività: $x \cdot (y+z) = (x \cdot y)+(x \cdot z)$ e $x+(y \cdot z) = (x+y) \cdot (x+z)$ e $x+(y \cdot z) = (x+y) \cdot (x+z)$
- Esistenza di elementi neutri: $x + 0 = x$ e $x \cdot 1 = x$
- Esistenza di un inverso: $x + x^{'} = 1$ e $x \cdot x^{'} = 0$

L'inverso in un algebra di Boole è univoco tale che:

$$x + y = 1 \text{ e } xy = 0 \text{ allora } y = x^{'}$$

Inoltre soddisfa altre 6 proprietà:

- Idempotenza: $x + x = x$ e $xx = x$
- Legge del limite/dominanza: $x + 1 = 1$ e $x \cdot 0 = 0$
- Legge di assorbimento: $x + xy = x$ e $x(x + y) = x$
- Legge d'involuzione: $(x^{'})^{'} = x$
- Legge 0 e 1: $0^{'} = 1$ e $1^{'} = 0$
- Leggi di De Morgan per le algebre di Boole: $(x + y)^{'} = x^{'} \cdot y^{'}$ e $(xy)^{'} = x^{'} + y^{'}$

3 Dualità

Il duale di un espressione booleana è l'espressione ottenuta invertendo tutte le operazioni che incontriamo: Rimpiazziamo 0 con 1 e 1 con 0 e Rimpiazziamo + con · e · con +

Notiamo che il duale di un teorema su un algebra di Boole è sempre un teorema.

4 Funzioni booleane

Una funzione booleana è una funzione di n argomenti su un algebra di Boole. è una funzione della forma:

$$f(x_1, ..., x_n) = X(x_1, ..., x_n)$$

Ad esempio: $f(x_1, x_2) = x_1 \wedge \overline{x_2}$

4.1 Ricavare funzioni da tavole di verità

Partendo da una tavola della verità del tipo:

x_1	x_2	x_3	$f(x_1, x_2, x_3)$
1	1	1	1
1	1	0	0
1	0	1	0
1	0	0	1
0	1	1	0
0	1	0	1
0	0	1	0
0	0	0	0

Procediamo così:

- Scriviamo le formule che danno come output 1. Queste formule sono dette mintermini
- Disgiungiamole tra di loro

In questo caso abbiamo:

- $x_1x_2x_3$

- $x_1\overline{x_2x_3}$
- $\overline{x_1}x_2\overline{x_3}$

Otteniamo che la seguente tavola di verità si ottiene dalla funzione:

$$f(x_1, x_2, x_3) = (x_1x_2x_3) \vee (x_1\overline{x_2x_3}) \vee \overline{x_1}x_2\overline{x_3}$$

Questa forma per una funzione booleana è dette **forma disgiuntiva normale**.

5 Porte funzionalmente complete

Qualsiasi funzione booleana si può scrivere come risultato di combinazioni di $\{AND, OR, NOT\}$.

Ne deduciamo che queste porte sono funzionalmente complete.

In particolare $\{AND, NOT\}$ e $\{OR, NOT\}$ sono funzionalmente complete.

Anche la porta NAND è funzionalmente completa.

6 Semplificazione di circuiti combinatori

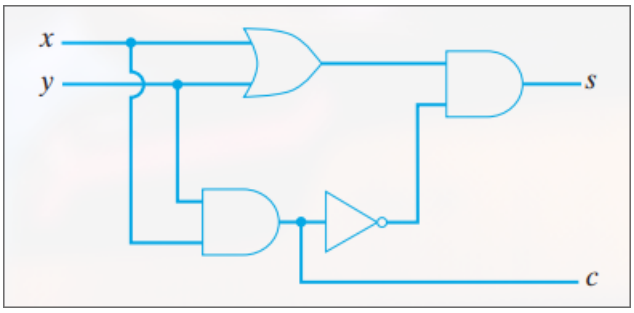
Possiamo semplificare circuiti combinatori attraverso le leggi di Quine-McCluskey:

- $Ea \vee E\overline{a} = E(a \vee \overline{a} = E1 = E$
- $E = E \vee Ea$

Dove E è una condizione booleana. Queste regole valgono per qualsiasi algebra di boole

7 Esempi di circuiti combinatori

7.1 Half-Adder

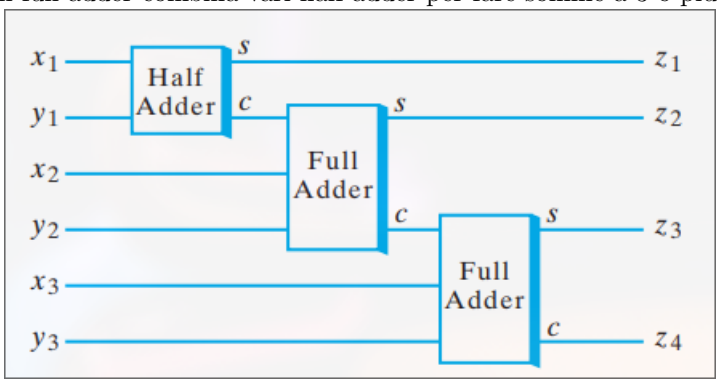


Un half-adder somma i due ingressi x, y e dà il risultato in uscita su s e l'eventuale riporto in c.

x	y	c	s
1	1	1	0
1	0	0	1
0	1	0	1
0	0	0	0

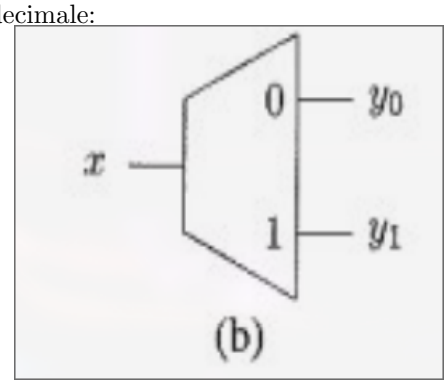
7.2 Full-Adder

Un full-adder combina vari half-adder per fare somme a 3 o più bit.



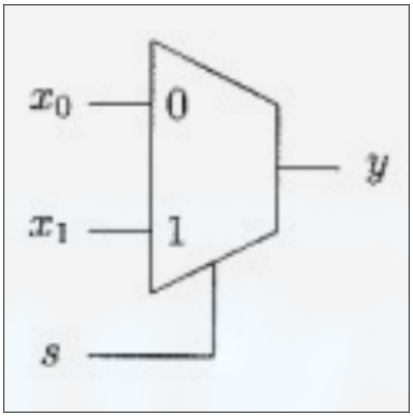
7.3 Decoder

Prende in ingresso n bit e attiva l'uscita sulla base dei bit in ingresso convertiti in numero decimale:



Ad esempio se in input c'è il numero 4, cioè 100 verrà attivata l'uscita 4.

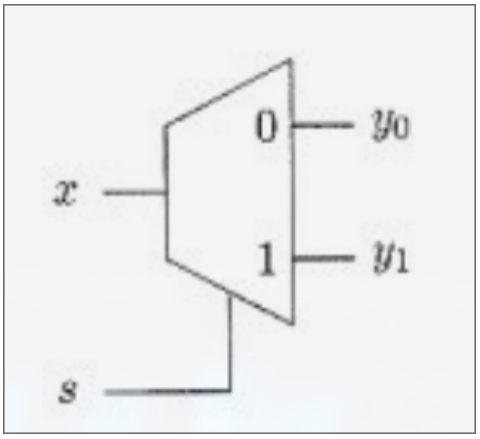
7.4 Multiplexer



Un multiplexer seleziona quale degli n ingressi viene mandato all'uscita attraverso uno o più ingressi di controllo s

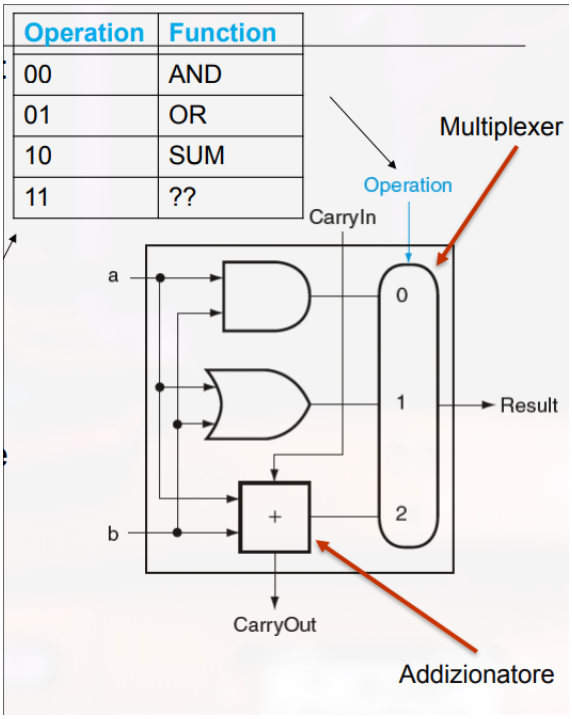
Ad esempio se i bit di controllo formano 11, verrà mandato in output l'ingresso 3.

7.5 Demultiplexer



7.6 ALU ad 1 bit

Operation	Function
00	AND
01	OR
10	SUM
11	??

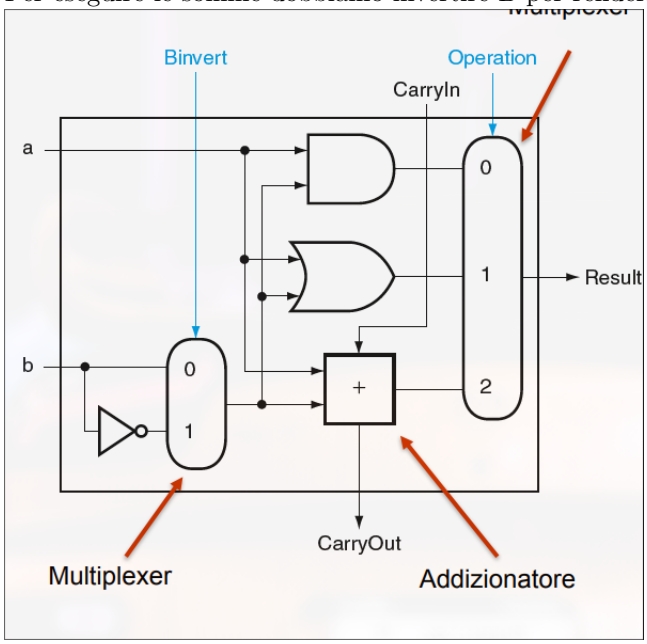


Un multiplexer attiva il circuito corrispondente all'operazione da eseguire, l'ingresso carryin permette di eseguire somme in complemento a due(sottrazioni) e dà in uscita un carryout in caso di riporto.

Il risultato viene fatto uscire nell'output result.

Per formare ALU a più bit, concateniamo questi circuiti.

Per eseguire le somme dobbiamo invertire B per renderlo negativo:



Usando la sottrazione possiamo anche implementare l'operazione $<$ in quanto se faccio $(a-b)$, verifico il bit più significativo, se è 1 vuol dire che la differenza è negativa e quindi $a < b$