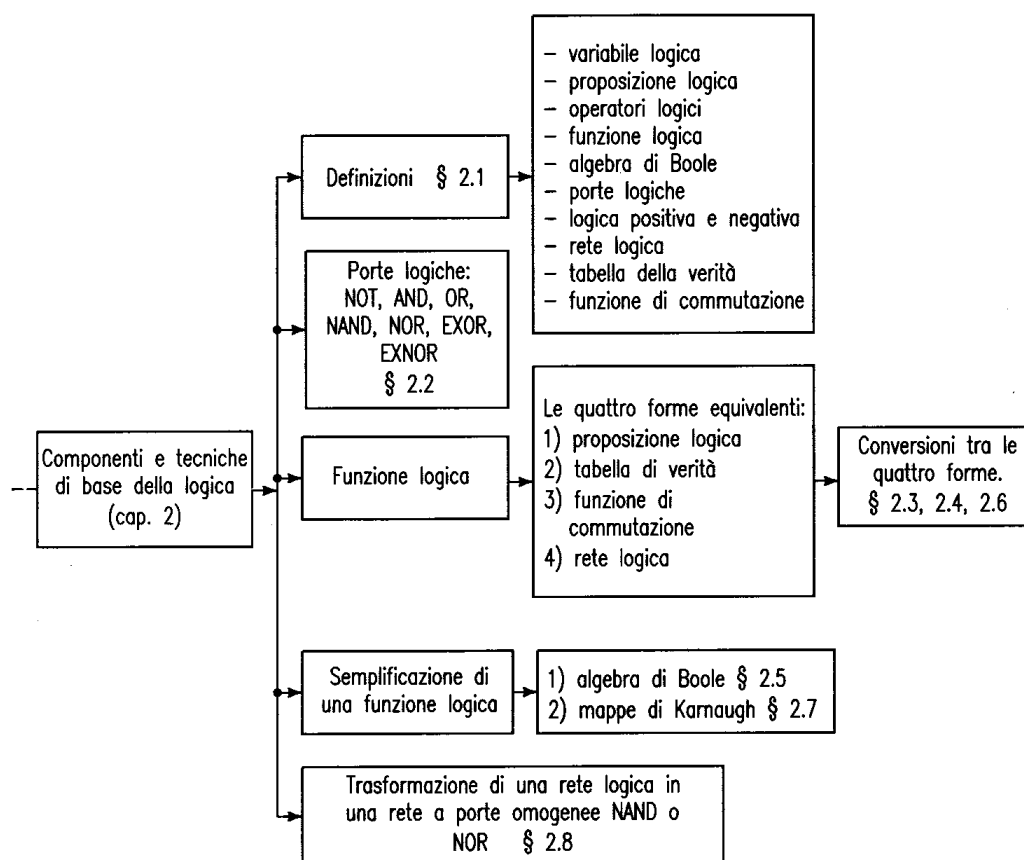


I COMPONENTI E LE TECNICHE DI BASE DELLA LOGICA



Struttura del cap. 2. Fig. 2.0

I componenti (porte logiche) e le tecniche descritte nel presente capitolo rappresentano i fondamenti dell'elettronica digitale e risultano propedeutici per l'analisi e la progettazione dei circuiti combinatori (capp. 3 e 5) e sequenziali (cap. 7).

Si analizzano le quattro forme equivalenti utilizzate per descrivere la relazione esistente tra gli ingressi e l'uscita (funzione di commutazione) di un circuito logico combinatorio, e le conversioni tra le quattro forme. Poiché le reti che possono svolgere una certa funzione sono infinite conviene scegliere la più semplice ed economica; le tecniche di semplificazione sfruttano l'algebra di Boole e le mappe di Karnaugh.

2.1 Alcune definizioni ed indicazioni generali

Def. ■ Variabile logica (o *binaria*): è una variabile che può assumere solo due valori, che normalmente vengono denominati 1 e 0.

Sinonimi del valore 1 sono: V (Vero), T ($True = Vero$).

Sinonimo del valore 0 è: F ($False = Falso$)

Una variabile logica può essere utilizzata, ad esempio, per rappresentare:

- lo stato di un interruttore (aperto/chiuso);
- lo stato di una lampada (accesa/spenta);
- il passaggio o meno di una corrente elettrica;
- il valore (alto/basso) di una tensione, ecc.

Def. ■ Proposizione logica: è una frase che ammette solo di essere vera o falsa.

Esempi di proposizioni logiche sono:

“la finestra è aperta”; “la lampada è accesa”; “la chitarra non è accordata”; ecc.

Def. ■ Operatori logici: sono quegli elementi che permettono di collegare le proposizioni logiche per costruire proposizioni più complesse; gli operatori logici fondamentali sono il NOT (negazione, complementazione), l'AND (prodotto logico) e l'OR (somma logica).

Ecco alcuni esempi di proposizioni che utilizzano operatori logici:

“la finestra non (NOT) è aperta” (assume valore vero quando è falsa la frase senza il “non”);

“la finestra è aperta e (AND) la lampada è accesa”; (assume valore vero quando sono vere entrambe le proposizioni);

“è aperta la finestra o (OR) la porta” (assume valore vero quando almeno una delle due proposizioni è vera).

Def. ■ Funzione logica: è il legame esistente tra una variabile binaria, detta dipendente, ed altre variabili binarie dette indipendenti. È importante saper individuare, nelle proposizioni logiche, le variabili dipendenti e quelle indipendenti.

Un esempio di funzione logica espressa attraverso una proposizione è:

“la finestra rimane aperta quando è giorno e non piove”. Lo ‘stato della finestra’ rappresenta la variabile dipendente, mentre le due variabili ‘giorno/notte’ e ‘pioggia/non pioggia’ sono quelle indipendenti.

Def. ■ Algebra di Boole: è lo strumento matematico che consente di rappresentare le funzioni logiche sotto forma di espressione algebrica ed elaborarle (semplificarle), attraverso opportune regole (descritte nel par. 2.5).

Nell'algebra di Boole le variabili logiche sono rappresentate da lettere maiuscole (A, B, C), l'operatore NOT da un trattino posto sopra la lettera (\bar{A} : negazione), l'operatore AND dal prodotto tra le variabili ($A \cdot B$: prodotto logico), l'operatore OR dalla somma tra le variabili ($A + B$: somma logica).

Come si vede nell'esempio 2.1, è possibile descrivere, attraverso un'espressione algebrica, la funzione logica individuata da una proposizione.

Ricavare l'espressione algebrica corrispondente alla seguente proposizione: "utilizzo l'automobile quando piove o quando sono in ritardo".

Soluzione

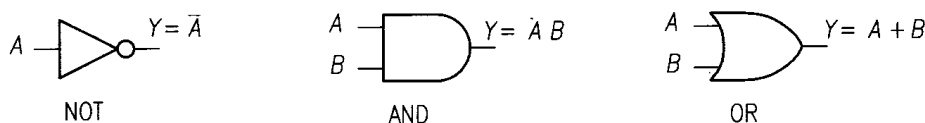
La variabile dipendente (la chiamiamo Y) è costituita dall'utilizzo o meno dell'auto. Le variabili indipendenti sono: la presenza o meno della pioggia (A) e il fatto di essere o no in ritardo (B).

Il legame logico tra le variabili indipendenti è costituito da o (operatore OR \rightarrow somma logica $\rightarrow +$).

L'espressione algebrica risulta quindi: $Y = A + B$ (si pronuncia: ipsilon uguale A or B).

Il passaggio diretto dalla proposizione all'espressione algebrica è possibile solo in casi molto semplici, come quello visto nell'esempio 2.1. Le tecniche che consentono la soluzione di casi più complessi verranno illustrate nel corso del presente capitolo.

Def. **Porte logiche** (*logic gates*): rappresentano la realizzazione elettronica degli operatori dell'algebra di Boole; esse ricevono in ingresso segnali elettrici digitali, in genere tensioni alte o basse, e producono in uscita il valore di tensione corrispondente al risultato dell'operazione. In fig. 2.1 sono mostrati i simboli grafici delle porte logiche fondamentali (NOT, AND, OR).



Simboli grafici delle porte logiche fondamentali (NOT, AND, OR). **Fig. 2.1**

Come si vedrà in seguito, vengono posti in commercio circuiti integrati contenenti le porte fondamentali (2, 3, 4, 6, ecc. ogni integrato) o altre porte logiche (NAND, NOR, EXOR, EXNOR), che possono essere ricavate come combinazione di quelle fondamentali. Si definiscono:

Def. **logica positiva** (*positive logic*): l'associazione dell'1 logico con una tensione ALTA (H) e dello 0 logico con una tensione BASSA (L);
logica negativa (*negative logic*): l'associazione inversa ($1 \rightarrow L$, $0 \rightarrow H$).

Def. **Rete logica**: è un circuito costituito da porte logiche.

In fig. 2.2 si vede un esempio di semplice rete logica.

Generalmente gli ingressi vengono disegnati a sinistra mentre le uscite a destra della rete.

Def. **Funzione di commutazione**: è l'espressione algebrica che lega una variabile d'uscita con gli ingressi di una rete logica.

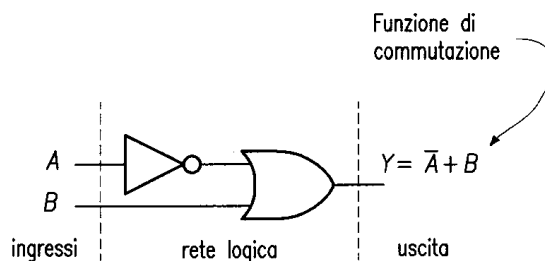


Fig. 2.2 Esempio di semplice rete logica.

Ogni rete logica può essere descritta attraverso funzioni di commutazione (una per ogni uscita), così come ad ogni funzione di commutazione corrisponde una rete logica.

Def. **Tabella della verità (truth table):** è un modo alternativo alla funzione di commutazione, per esprimere la relazione tra ingressi e uscite di una rete. La tabella fa corrispondere ad ogni possibile combinazione delle variabili di ingresso il relativo valore della variabile d'uscita.

Se la rete possiede n ingressi le combinazioni possibili sono 2^n e la tabella è quindi costituita da altrettante righe. Per ottenere un'impostazione ordinata della tabella, conviene ordinare le combinazioni dei valori delle variabili d'ingresso secondo la numerazione binaria.

Nella fig. 2.3 sono riportate le impostazioni delle tabelle per 1, 2, 3 e 4 variabili d'ingresso; nella colonna Y verranno inseriti i valori della variabile d'uscita, corrispondenti ad ogni combinazione degli ingressi.

Regola pratica per compilare la parte sinistra di una tabella della verità (fig. 2.3): osservando in verticale i valori delle variabili d'ingresso si nota che la prima a destra cambia valore ad ogni riga (0101...), la seconda cambia valore ogni due righe (00110011...), la terza ogni quattro righe (000011110000...), la quarta ogni otto e così via.

A	Y
0	
1	

A	B	Y
0	0	
0	1	
1	0	
1	1	

A	B	C	Y
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

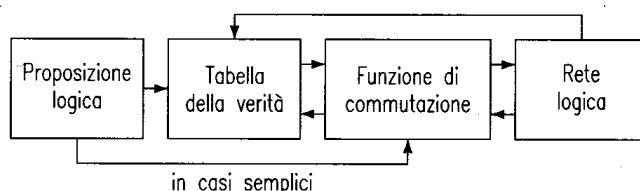
A	B	C	D	Y
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

Fig. 2.3 Tabelle della verità per reti a 1, 2, 3 e 4 variabili d'ingresso.



Osservazione importante:

proposizione logica, tabella della verità, funzione di commutazione e rete logica sono quattro forme diverse per rappresentare la funzione logica, cioè il legame esistente tra le variabili d'ingresso e quella d'uscita. Nel corso del presente capitolo si analizzeranno le tecniche che consentono di passare da una rappresentazione all'altra, secondo lo schema di fig. 2.4, e di semplificare tali rappresentazioni quando è possibile.

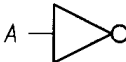
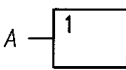
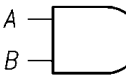
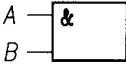
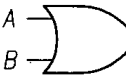
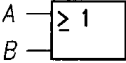


Passaggi tra le quattro rappresentazioni di una funzione logica. **Fig. 2.4**

2.2 Le porte logiche

Nella tab. 2.1 sono raffigurati i simboli grafici delle tre porte logiche fondamentali con le corrispondenti tabelle della verità e descrizioni a parole. È utile memorizzare le descrizioni a parole delle porte logiche, perché rimangono valide anche per le porte con più di due ingressi.

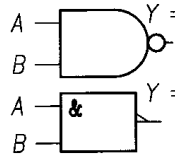
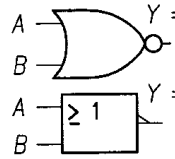
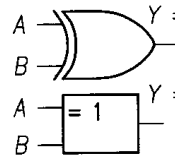
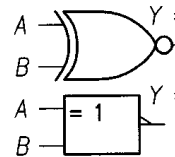
Tab. 2.1 - Simboli grafici, tabelle della verità e descrizioni a parole delle porte logiche fondamentali.

Porta	Simboli circuitali: - tradizionale - IEEE/IEC	Tabella di verità	Descrizione															
NOT	<div><div>$Y = \bar{A}$</div><div>$Y = \bar{A}$</div></div>	<table><tr><th>A</th><th>Y</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	Y	0	1	1	0	L'uscita assume valore complementare a quello dell'ingresso									
A	Y																	
0	1																	
1	0																	
AND	<div><div>$Y = AB$</div><div>$Y = AB$</div></div>	<table><tr><th>A</th><th>B</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	Y	0	0	0	0	1	0	1	0	0	1	1	1	L'uscita è uguale a 1 solo se tutti gli ingressi sono uguali a 1
A	B	Y																
0	0	0																
0	1	0																
1	0	0																
1	1	1																
OR	<div><div>$Y = A+B$</div><div>$Y = A+B$</div></div>	<table><tr><th>A</th><th>B</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	Y	0	0	0	0	1	1	1	0	1	1	1	1	L'uscita è uguale a 1 se almeno uno degli ingressi è uguale a 1
A	B	Y																
0	0	0																
0	1	1																
1	0	1																
1	1	1																

Nella tab. 2.2 sono rappresentate le porte logiche derivate dalla combinazione di quelle fondamentali.

Un circoletto sull'ingresso di una porta, così come sull'uscita, equivale alla presenza di un NOT come si vede in fig. 2.5.

Tab. 2.2 - Simboli grafici, tabelle della verità e descrizioni a parole delle porte logiche derivate.

Porta	Simboli circuitali: - tradizionale - IEEE/IEC	Tabella di verità	Descrizione															
NAND	 $Y = \overline{A \cdot B}$ $Y = \overline{A \cdot B}$	<table><tr><th>A</th><th>B</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	Y	0	0	1	0	1	1	1	0	1	1	1	0	L'uscita è uguale a 0 solo se tutti gli ingressi sono uguali a 1
A	B	Y																
0	0	1																
0	1	1																
1	0	1																
1	1	0																
NOR	 $Y = \overline{A + B}$ $Y = \overline{A + B}$	<table><tr><th>A</th><th>B</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	Y	0	0	1	0	1	0	1	0	0	1	1	0	L'uscita è uguale a 0 se almeno uno degli ingressi è uguale a 1
A	B	Y																
0	0	1																
0	1	0																
1	0	0																
1	1	0																
EXOR	 $Y = A \oplus B$ $Y = A \oplus B$	<table><tr><th>A</th><th>B</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	Y	0	0	0	0	1	1	1	0	1	1	1	0	L'uscita vale 0 se gli ingressi sono uguali tra loro; vale 1 se sono diversi
A	B	Y																
0	0	0																
0	1	1																
1	0	1																
1	1	0																
EXNOR	 $Y = \overline{A \oplus B}$ $Y = \overline{A \oplus B}$	<table><tr><th>A</th><th>B</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	Y	0	0	1	0	1	0	1	0	0	1	1	1	L'uscita vale 1 se gli ingressi sono uguali tra loro; vale 0 se sono diversi
A	B	Y																
0	0	1																
0	1	0																
1	0	0																
1	1	1																

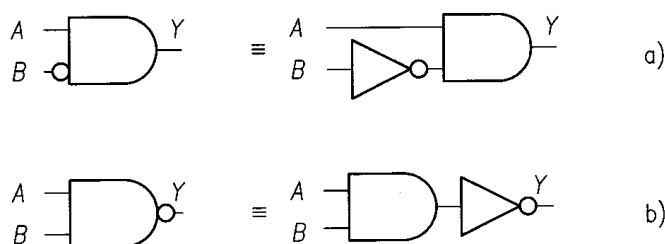


Fig. 2.5 Un circoletto in ingresso o in uscita di una porta logica rappresenta la funzione NOT.

Esistono in commercio porte AND, OR, NAND e NOR con *più di due ingressi*; il loro funzionamento è definito dalla stessa descrizione a parole delle corrispondenti porte a due ingressi.

Le porte EXOR (OR esclusivo) ed EXNOR (OR esclusivo negato) esistono in commercio solo con due ingressi.

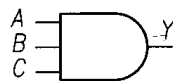
Si noti che le porte EXOR ed EXNOR permettono di distinguere quando due variabili hanno valore uguale o complementare, possono quindi essere utilizzate come comparatori.

ESEMPIO 2.2

Compilare le tabelle della verità delle porte AND e OR a tre ingressi in figura.

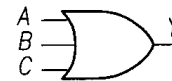
Soluzione

Dalla descrizione a parole dell'AND si deduce che l'unica combinazione di ingressi che produce 1 in uscita è 111. Per l'OR l'unica combinazione che produce 0 in uscita è 000. Di conseguenza le tabelle della verità sono quelle in figura.



A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

a)



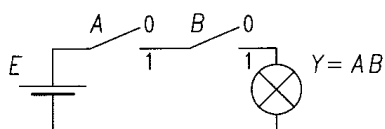
A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

b)

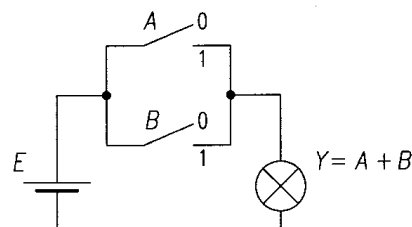
Fig. 2.6

ESEMPIO 2.3

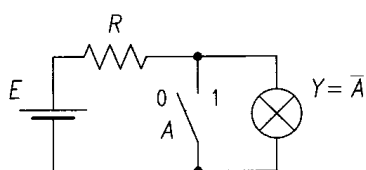
Individuare la funzione di commutazione che lega lo stato degli interruttori con quello della lampada nei due casi in figura. (Si considerino gli stati di interruttore chiuso e di lampada accesa corrispondenti all'uno logico).



a)



b)



c)

Fig. 2.7

Soluzione

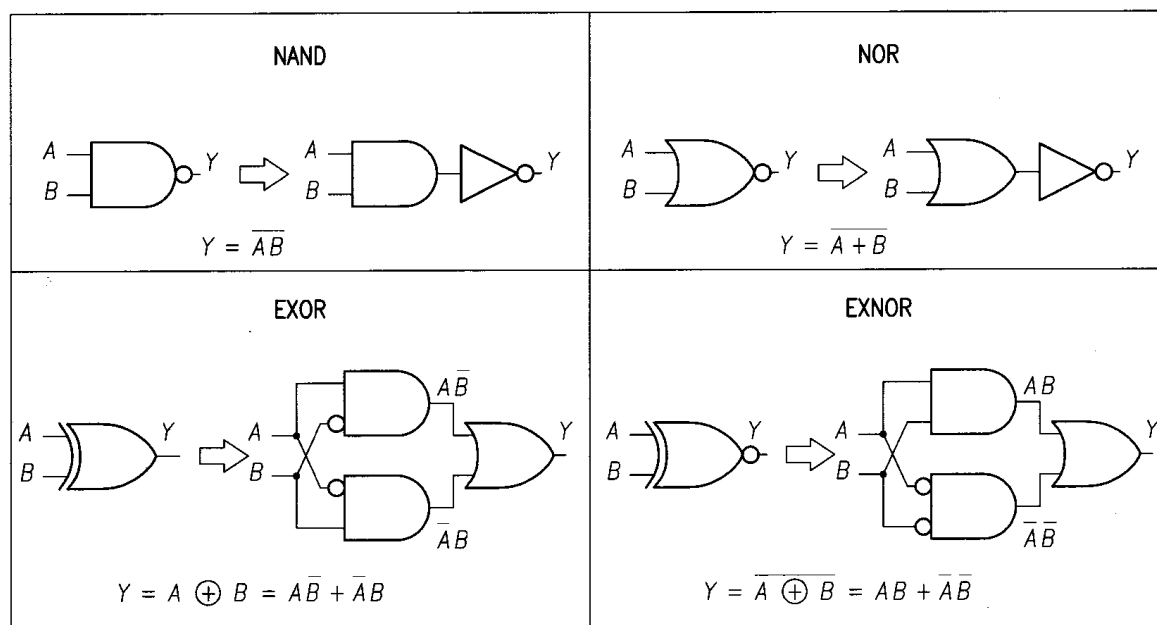
Le variabili d'ingresso sono rappresentate dalla posizione degli interruttori (A,B) mentre quella d'uscita corrisponde allo stato della lampada (Y).

- a) la lampada si accende ($Y=1$) solo quando entrambi gli interruttori sono chiusi ($A=B=1$), quindi $Y=AB$;

- b) la lampada si accende quando almeno un interruttore è chiuso; quindi $Y = A + B$;
 c) la lampada si accende quando l'interruttore è aperto, quindi $Y = \bar{A}$. In questo caso il resistore ha la funzione di limitare la corrente fornita dal generatore, quando l'interruttore è chiuso.

La tab. 2.3 illustra come sia possibile realizzare le funzioni delle porte derivate (NAND, NOR, EXOR e EXNOR), utilizzando solamente porte fondamentali (AND, OR, NOT).

Tab. 2.3 - Realizzazione delle funzioni derivate (NAND, NOR, EXOR, EXNOR) con porte fondamentali.



Def. Le porte logiche NAND e NOR vengono dette anche **porte universali**, in quanto è possibile realizzare qualunque funzione di commutazione utilizzando solo NAND o solo NOR. La tab. 2.4 mostra come realizzare le funzioni fondamentali con sole porte NAND o sole porte NOR.

Si verifichi per esercizio l'identità tra le tabelle della verità delle reti a NAND o a NOR e quelle delle corrispondenti porte fondamentali.

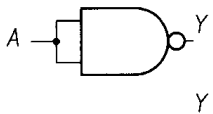
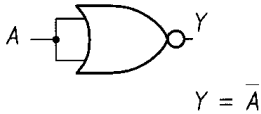
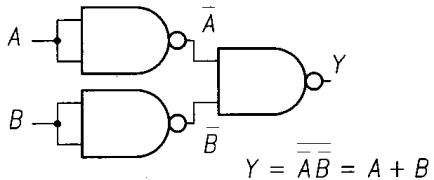
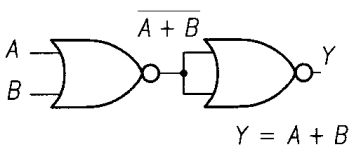
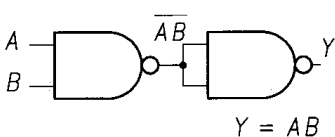
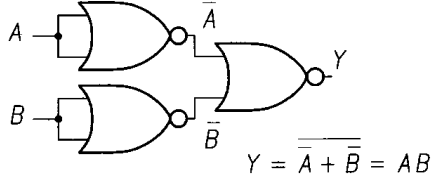
Qualunque rete logica può quindi essere trasformata in una rete di sole porte NAND o NOR; nel par. 2.8 si analizzano la tecnica per effettuare rapidamente questa trasformazione ed i vantaggi che ne derivano.

Nell'appendice C è riportato l'elenco dei circuiti integrati in commercio che contengono porte logiche e le indicazioni fondamentali per il loro utilizzo. I circuiti integrati contenenti fino a una decina di porte logiche sono definiti di tipo SSI (*Small Scale Integration*).

2.2.1 Controllo del flusso di un segnale tramite porte logiche

Se agli ingressi di una porta logica si presentano segnali variabili nel tempo, l'uscita assume il valore specificato dalla tabella della verità in base al valore, istante per istante, dei segnali d'ingresso. Si vedano a proposito i diagrammi temporali nell'esempio 2.4.

Tab. 2.4 - Realizzazione delle funzioni fondamentali con sole porte NAND o sole porte NOR.

Porta	con operatori NAND	con operatori NOR
NOT	 $Y = \bar{A}$	 $Y = \bar{A}$
OR	 $Y = \overline{\bar{A}\bar{B}} = A + B$	 $Y = A + B$
AND	 $Y = AB$	 $Y = \overline{\bar{A} + \bar{B}} = AB$

ESEMPIO 2.4

Ricavare i diagrammi temporali delle uscite delle porte AND e OR quando all'ingresso sono presenti i segnali variabili nel tempo A e B , rappresentati in fig. 2.8b.

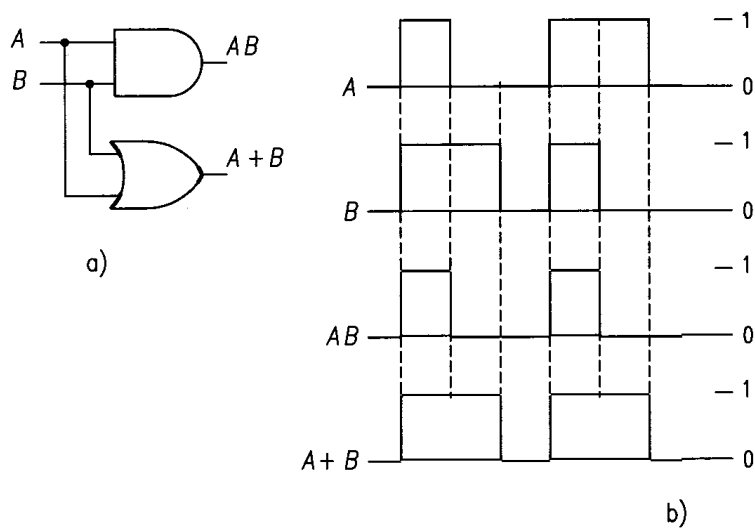


Fig. 2.8

Soluzione

Il valore dell'uscita di ogni porta dipende, istante per istante, dalla combinazione dei valori in ingresso, come specificato dalla tabella della verità. L'andamento nel tempo delle uscite è illustrato nella fig. 2.8b.

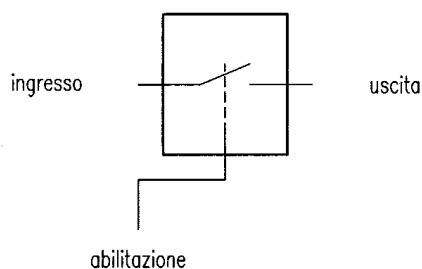


Fig. 2.9 Schematizzazione di una porta logica per il controllo del flusso di un segnale digitale.

Agendo opportunamente sugli ingressi di una porta logica è possibile controllare il flusso di un segnale digitale, cioè permettere o impedire il passaggio del segnale attraverso la porta.

Il segnale digitale viene inserito in un ingresso della porta mentre l'altro ingresso funge da abilitazione (enable); l'uscita riprodurrà il segnale d'ingresso, oppure si manterrà ad un livello costante a seconda dello stato dell'ingresso di abilitazione, come schematizzato in fig. 2.9.

La tab. 2.5 illustra le modalità di utilizzazione delle varie porte logiche, per controllare il flusso di un segnale.

Tab. 2.5 - Modalità di utilizzazione delle porte logiche per il controllo del flusso di segnali digitali.

Controllo con porta	Ingresso di abilitazione	Uscita
AND	0	0
	1	Riproduce il segnale d'ingresso
OR	0	Riproduce il segnale d'ingresso
	1	1
NAND	0	1
	1	Riproduce il segnale d'ingresso negato
NOR	0	Riproduce il segnale d'ingresso negato
	1	0

La denominazione 'porte logiche' (*logic gates*) di questi componenti, deriva dalla loro caratteristica di poter funzionare come interruttori nei confronti di un segnale digitale. La tecnica di controllo del flusso dei segnali digitali tramite porte logiche è detta *gating*.

ESEMPIO 2.5

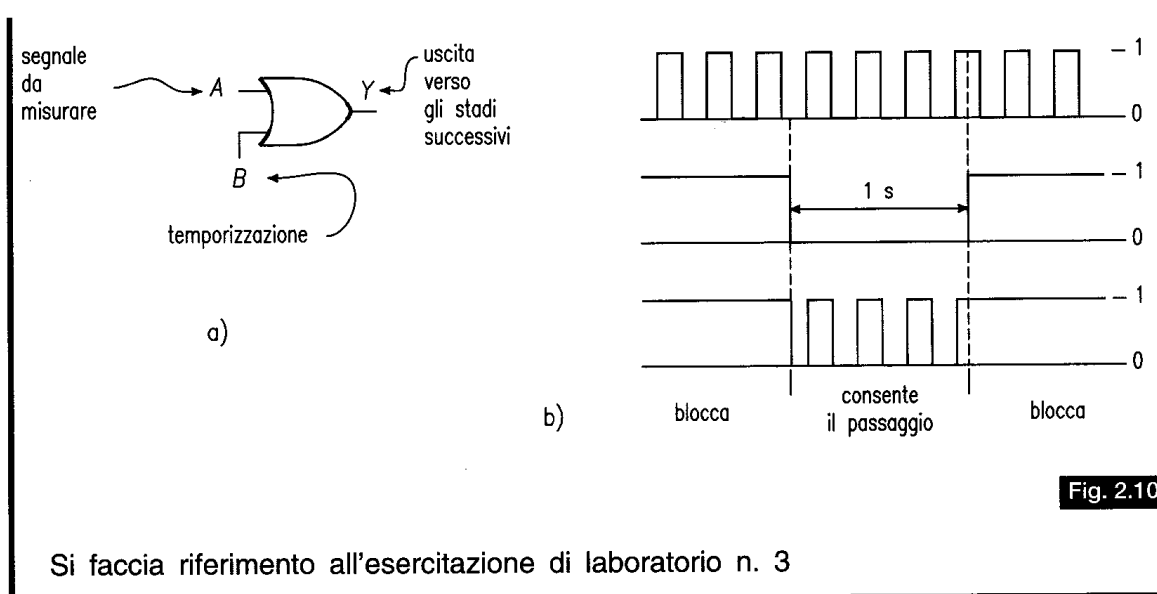
Lo stadio d'ingresso di un frequenzimetro deve abilitare il passaggio del segnale digitale da misurare, verso i successivi stadi di conteggio e visualizzazione, per un intervallo di tempo di un secondo. Il segnale di temporizzazione va a livello 0 per un secondo per segnalare l'intervallo di conteggio e poi ritorna a livello 1.

Individuare la porta logica necessaria per realizzare lo stadio d'ingresso.

Soluzione

La porta necessaria è l'OR (fig. 2.10a) perché lascia transitare il segnale quando il terminale di abilitazione vale 0. La forma d'onda in uscita è rappresentata nella fig. 2.10b.

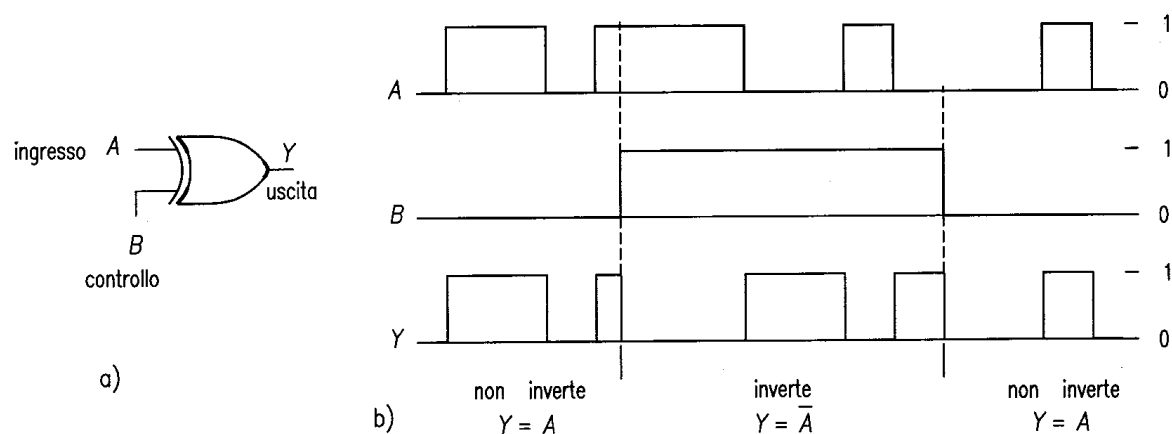
È possibile utilizzare anche una porta NOR; in questo caso, quando il terminale di abilitazione vale 0, in uscita viene riprodotto l'ingresso negato.



In maniera analoga, utilizzando porte EXOR o EXNOR si può negare o meno il segnale d'ingresso, a seconda del livello presente sull'ingresso di controllo, come illustrato in tab. 2.6 ed in fig. 2.11.

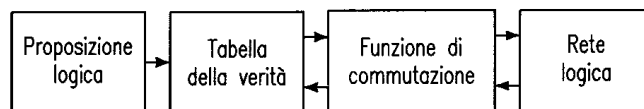
Tab. 2.6

Controllo con porta	Ingresso di controllo	Uscita
EXOR	0	Riproduce il segnale d'ingresso
	1	Riproduce il segnale d'ingresso negato
EXNOR	0	Riproduce il segnale d'ingresso negato
	1	Riproduce il segnale d'ingresso



2.3 Rete logica e funzione di commutazione

Fig. 2.12



Collegando più porte logiche si realizza una rete logica, caratterizzata da un certo numero d'ingressi e di uscite (fig. 2.13). Poiché per descrivere il funzionamento della rete è necessario esprimere il legame esistente tra ognuna delle uscite e tutti gli ingressi, si devono ricavare tante funzioni di commutazione quante sono le uscite della rete.

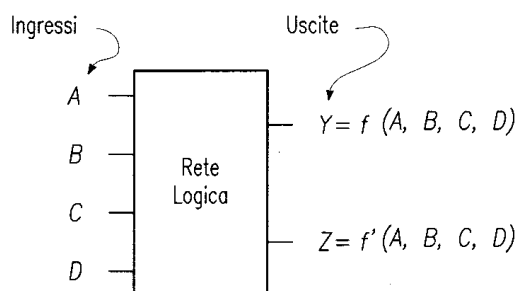


Fig. 2.13 Schema a blocchi di una rete logica.

Def. **Numero dei livelli di una rete** è il massimo numero di porte logiche attraversate dai segnali che vanno dagli ingressi all'uscita della rete.

Def. Come si vede in fig. 2.14, vengono dette **porte di 1° livello** tutte quelle che ricevono solamente segnali d'ingresso. Seguendo i percorsi più lunghi, dall'ingresso all'uscita, le porte successive verranno dette di **2° livello**, di **3° livello**, ecc.; nei percorsi rimanenti potranno mancare porte relative a qualche livello. Si noti, ad esempio, che nella fig. 2.14 l'ingresso D è collegato direttamente ad una porta di 2° livello.

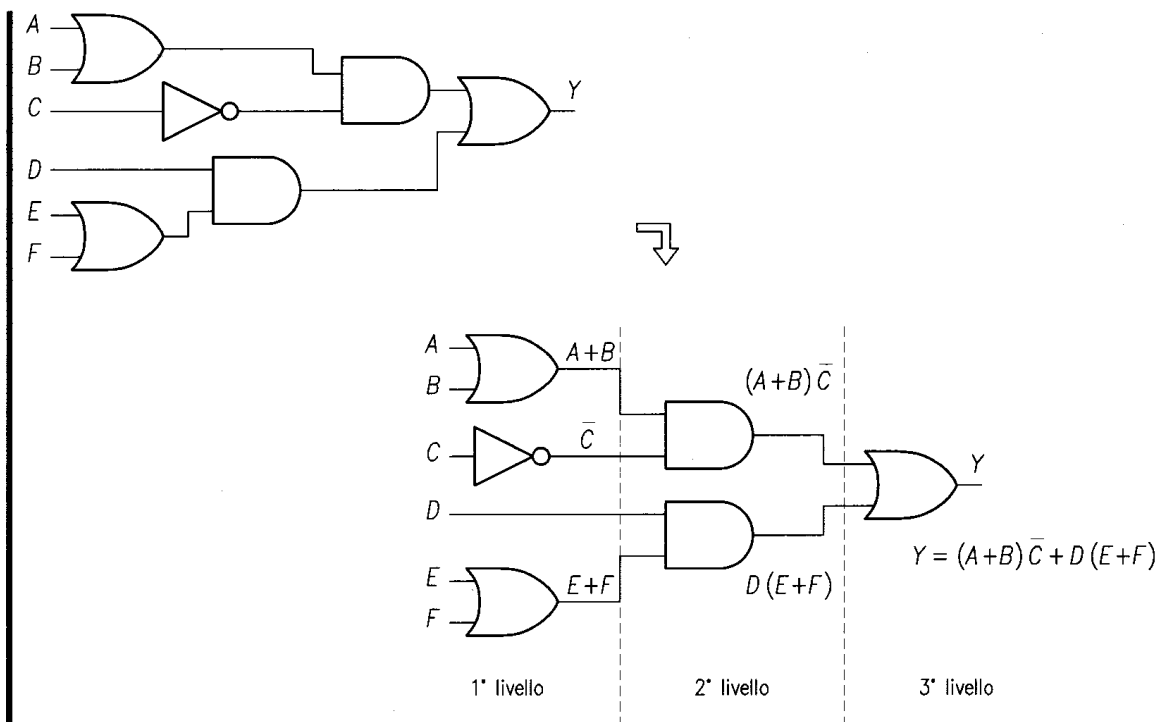
2.3.1 Dalla rete logica alla funzione di commutazione



Per ricavare la funzione di commutazione di una data rete logica, si scrive l'espressione logica all'uscita di tutte le porte di 1° livello e poi si passa al 2° livello, procedendo così fino all'uscita. Si veda l'esempio 2.6.

ESEMPIO 2.6

Suddividere per livelli le porte della rete logica in fig. 2.14a e ricavare la funzione di commutazione corrispondente.



Soluzione

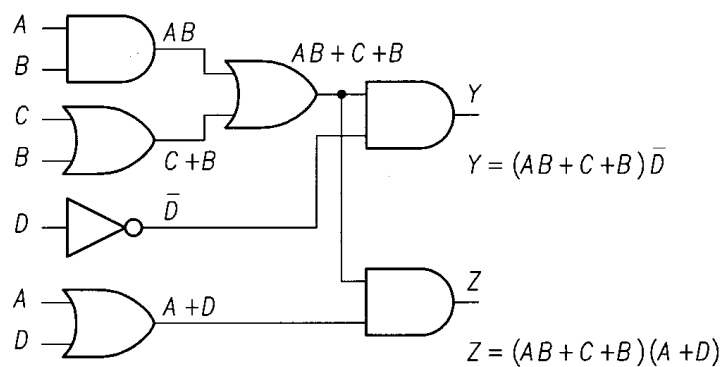
La suddivisione in livelli è rappresentata nella fig. 2.14b. La funzione di commutazione risulta:

$$Y(A+B)\bar{C} + D(E+F)$$

Se la rete possiede più di un'uscita, bisogna ricavare una funzione di commutazione per ognuna delle uscite, come illustrato nell'esempio 2.7.

ESEMPIO 2.7

Ricavare le funzioni di commutazione che descrivono la rete in figura.



Soluzione

$$Y = (AB + C + B)\bar{D}$$

$$Y = (AB + C + B)(A + B)$$

2.3.2 Dalla funzione di commutazione alla rete logica

Il procedimento per disegnare la rete logica, data la funzione, è un po' più complesso, in quanto richiede un'attenta analisi della funzione per individuare il livello corrispondente alle varie operazioni.

Per **individuare il livello** di ogni operazione, nella funzione di commutazione, si seguono queste regole:

- le operazioni di primo livello sono contenute nelle parentesi più interne, mentre quelle tra le parentesi più esterne sono di livello massimo.
- l'AND ha la priorità sull'OR (cioè deve essere eseguito prima) all'interno della stessa parentesi; ad es. $(AB + C)$;
- se un termine di un prodotto o di una somma è negato, la negazione del termine ha la priorità; ad es. $A + \overline{B}$;
- se sopra ad un'espressione è posta una linea continua di negazione, le operazioni specificate dall'espressione hanno la precedenza sulla complementazione; ad es. $\overline{AB + C}$

Volendo calcolare il valore di Y , noti i valori delle variabili indipendenti, bisogna eseguire le operazioni in ordine dal primo all'ultimo livello.

ESEMPIO 2.8

Individuare i livelli delle operazioni contenute nella seguente espressione:
 $Y = (\overline{A}B + C)(A + B)$ e calcolare il valore di Y per $A = 0$, $B = 1$, $C = 0$.

Soluzione

I livelli delle operazioni sono indicati in figura. Eseguendo le operazioni dal 1° al 4° livello, con i dati forniti, si ottiene $Y = 1$

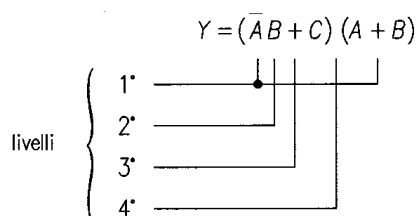


Fig. 2.16



Data una funzione di commutazione si determina la rete logica corrispondente, mediante il seguente algoritmo:

- individuare il livello di ogni operazione;
- disegnare le porte relative alle operazioni di primo livello (ricevono esclusivamente segnali d'ingresso);
- combinare le uscite di queste porte ed eventuali segnali d'ingresso, come specificato dalle operazioni di secondo livello;
- procedere analogamente per i livelli successivi, fino all'uscita del circuito.

Disegnare la rete logica corrispondente alla seguente funzione di commutazione:

$$Y = [(AB + C) + \overline{BD}] + (A + E)B + D$$

Soluzione

S'individuano i livelli delle operazioni come indicato in figura e si disegnano le porte logiche a partire da quelle di primo livello, procedendo da sinistra verso destra fino all'uscita Y (fig. 2.17a).

Scrivendo all'uscita di ogni porta la relativa espressione logica si può verificare la correttezza del risultato ottenuto (fig. 2.17b).

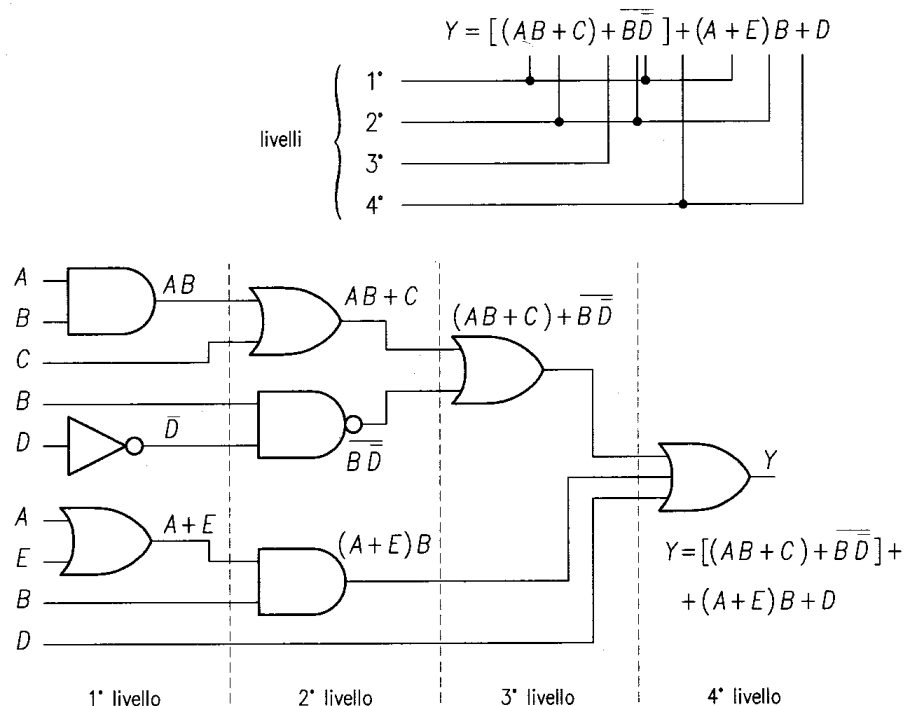


Fig. 2.17

Nel corso del capitolo sono descritte le tecniche per semplificare, laddove sia possibile, funzioni di commutazione complesse ed ottenere reti con un minor numero di componenti.

Nei capitoli seguenti si analizzano circuiti integrati che contengono funzioni equivalenti a decine o centinaia di porte logiche (integrati MSI e LSI); l'utilizzo di tali circuiti può ridurre notevolmente la complessità della rete.

2.4 Dalla proposizione logica alla tabella della verità

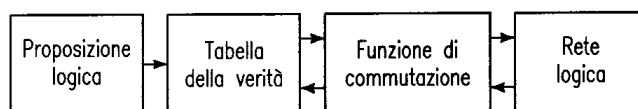


Fig. 2.18



Per ricavare la tabella della verità corrispondente ad una data proposizione logica, è necessario compiere i seguenti quattro passi:

- 1) individuare nella proposizione le variabili indipendenti (ingressi) e quella dipen-

- dente (uscita), associandole a lettere dell'alfabeto;
- 2) associare i valori 0 e 1 ai due stati di ogni variabile d'ingresso e d'uscita (qualunque associazione è valida, se non è specificata nella proposizione);
 - 3) impostare la tabella, elencando nella parte sinistra tutte le possibili combinazioni delle variabili d'ingresso (seguire la numerazione binaria crescente);
 - 4) riempire la colonna di destra con i valori della variabile d'uscita corrispondenti alle varie combinazioni degli ingressi, come specificato dalla proposizione data;

ESEMPIO 2.10

Ricavare la tabella della verità corrispondente alla seguente proposizione: "l'allarme suona se l'interruttore dell'impianto è attivato e almeno uno dei due sensori è eccitato"

Soluzione

Si seguono i quattro passi:

A	B	C	Y	1)	Variabili d'ingresso:	A = stato dell'interruttore dell'impianto B e C = stato dei sensori.
0	0	0	0			
0	0	1	0			
0	1	0	0			
0	1	1	0	2)	Variabile d'uscita:	Y = stato dell'allarme.
1	0	0	0		A = 1 impianto acceso;	A = 0 impianto spento;
1	0	1	1		B, C = 1 sensore eccitato;	B, C = 0 sensore a riposo;
1	1	0	1		Y = 1 l'allarme suona;	Y = 0 l'allarme non suona.
1	1	1	1	3)4)	La tabella della verità è rappresentata in fig. 2.19.	

Fig. 2.19



Se le variabili d'uscita sono più di una, altrettante dovranno essere le tabelle della verità. È tuttavia possibile, se ogni variabile d'uscita dipende da tutte le variabili d'ingresso, condensare le tabelle in un'unica tabella con tante colonne nella porzione di destra quante sono le variabili d'uscita.

A volte si trovano tabelle della verità che presentano delle **X nella porzione sinistra**, cioè in corrispondenza dei valori delle variabili d'ingresso. Il significato di queste **condizioni d'indifferenza sugli ingressi** è che l'uscita assume il valore specificato nella riga in cui compare la X, qualunque valore assumano gli ingressi contrassegnati con X, come illustrato nell'esempio 2.11. Le condizioni d'indifferenza sugli ingressi vengono utilizzate con lo scopo di compattare la tabella, riducendo il numero di righe necessarie.

ESEMPIO 2.11

A	B	C	Y
0	X	X	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

a)

A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

b)

Compilare la tabella della verità completa, corrispondente a quella compatta di fig. 2.20a.

Soluzione

Il significato della prima riga della tabella è: l'uscita assume valore 0 quando $A = 0$, qualunque sia il valore di B e C .

Di conseguenza la tabella completa sarà quella di fig. 2.20b. 2.5 Le regole dell'algebra di Boole

Fig. 2.20

2.5 Le regole dell'algebra di Boole

L'algebra sviluppata a metà del XIX secolo dal matematico inglese G. Boole, con lo scopo di analizzare ed elaborare le proposizioni logiche, rimase una teoria senza risvolti pratici fino agli anni '40, quando E. A. Shannon l'applicò allo studio di reti con interruttori e relais, per risolvere problemi di commutazione telefonica. Attualmente costituisce il fondamento per l'analisi ed il progetto delle reti elettroniche digitali.

L'algebra di Boole è definita da:

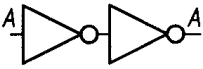
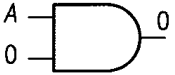
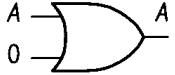

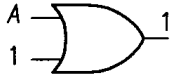
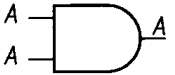
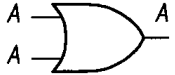
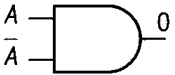
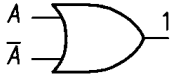
- un insieme di simboli: $\{0,1\}$;
- un insieme di operatori: $\{\neg, \times, +\}$ (complementazione o NOT, prodotto logico o AND, somma logica o OR);
- un insieme di assiomi che forniscono i risultati degli operatori sull'insieme dei simboli: si vedano le tabelle della verità delle porte NOT, AND e OR in tab. 2.1.

Dagli assiomi si ricavano le regole dell'algebra.

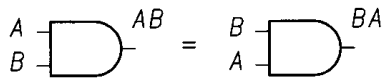
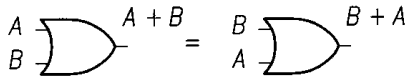
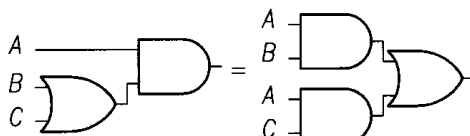
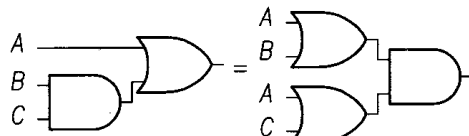
La memorizzazione delle regole sarà più semplice se si tiene presente il seguente **principio di dualità**: le proprietà ed i teoremi della somma logica si possono ricavare da quelle del prodotto logico (e viceversa) scambiando, ove presenti, il segno di AND (\cdot) con quello di OR ($+$) e lo 0 logico con l'1 logico.

Nelle tabb. 2.7, 2.8 e 2.9 sono riportate le regole dell'algebra di Boole, affiancate secondo il principio di dualità.

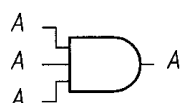
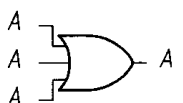

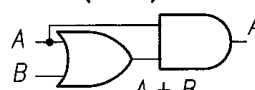
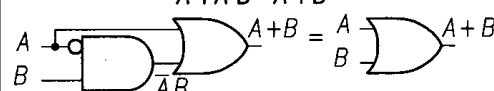
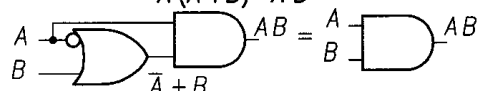
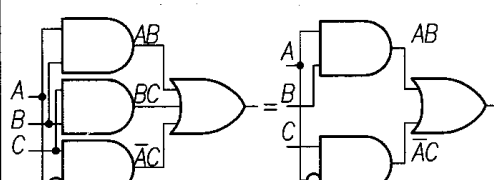
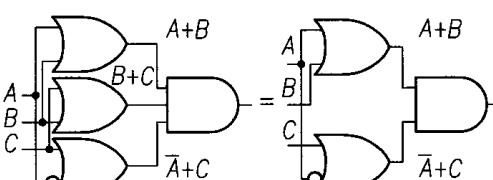
Tab. 2.7 - Proprietà delle funzioni logiche ad una variabile.

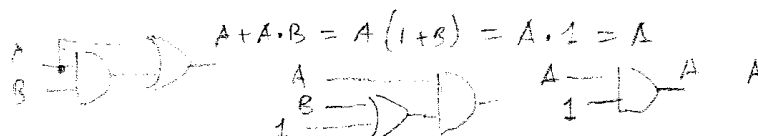
NOT	AND (diretta)	OR (duale)
funzione e schema	funzione e schema	funzione e schema
$\overline{\overline{A}} = A$ 	$A \cdot 0 = 0$ 	$A + 0 = A$ 
	$A \cdot 1 = A$ 	$A + 1 = 1$ 
	$A \cdot A = A$ 	$A + A = A$ 
	$A \cdot \overline{A} = 0$ 	$A + \overline{A} = 1$ 

Tab. 2.8 - Proprietà e teoremi delle funzioni logiche a due e più variabili.

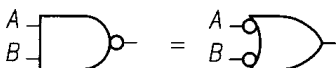
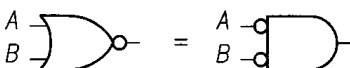
Proprietà	AND (diretta)	OR (duale)
commutativa	$A \cdot B = B \cdot A$ 	$A + B = B + A$ 
associativa	$A \cdot B \cdot C = (A \cdot B) \cdot C = A \cdot (B \cdot C) = (A \cdot C) \cdot B$	$A + B + C = (A + B) + C = A + (B + C) = (A + C) + B$
distributiva	$A \cdot (B + C) = A \cdot B + A \cdot C$ 	$A + B \cdot C = (A + B) \cdot (A + C)$ 



Teoremi	diretto	duale
idempotenza	$A \cdot A \cdot A = A$ 	$A + A + A = A$ 
1° assorbimento	$A + A \cdot B = A$ 	$A \cdot (A + B) = A$ 
2° assorbimento	$A + \bar{A} \cdot B = A + B$ 	$A \cdot (\bar{A} + B) = A \cdot B$ 
3° assorbimento	$A \cdot B + B \cdot C + \bar{A} \cdot C = A \cdot B + \bar{A} \cdot C$ 	$(A + B) \cdot (B + C) \cdot (\bar{A} + C) = (A + B) \cdot (\bar{A} + C)$ 



Tab. 2.9 - Teoremi di De Morgan.

	1° teorema (diretto)	2° teorema (duale)
con due variabili	$\overline{A \cdot B} = \overline{A} + \overline{B}$ 	$\overline{A + B} = \overline{A} \cdot \overline{B}$ 
con n variabili	$\overline{A \cdot B \cdot C \dots N} = \overline{A} + \overline{B} + \overline{C} + \dots + \overline{N}$	$\overline{A + B + C + \dots + N} = \overline{A} \cdot \overline{B} \cdot \overline{C} \dots \overline{N}$



Si faccia attenzione a non confondere espressioni del tipo $\overline{A + B}$ con $\overline{A} + \overline{B}$; nella prima la somma logica ha la priorità mentre nella seconda devono essere eseguite prima le negazioni.

ESEMPIO 2.12

Utilizzando le opportune regole dell'algebra di Boole dimostrare il duale del 2° teorema di assorbimento, e cioè:

$$A(\overline{A} + B) = AB$$

Soluzione

Applicando la proprietà distributiva del prodotto al primo membro si ottiene:

$$A\overline{A} + AB = AB$$

ma poiché $A\overline{A} = 0$ e $0 + AB = AB$ i due membri si equivalgono.

Un metodo alternativo per dimostrare l'uguaglianza è quello di compilare le tabelle della verità relative ai due membri e verificarne l'identità.

ESEMPIO 2.13

Semplificare la seguente funzione utilizzando le regole dell'algebra di Boole:

$$Y = ABC\overline{C} + AC + AB\overline{C}$$

Soluzione

Raccogliendo $AC\overline{C}$ dal 1° e dal 3° termine si ottiene

$$Y = AC\overline{C}(B + \overline{B}) + AC$$

ricordando che $B + \overline{B} = 1$ e che $AC\overline{C} \cdot 1 = AC\overline{C}$ si trova

$$Y = AC\overline{C} + AC$$

raccogliendo A ed eliminando la C come nel passo precedente si ottiene

$$Y = A$$

La funzione proposta equivale quindi a $Y = A$.

L'applicazione delle regole dell'algebra di Boole richiede una discreta pratica, oltre che una perfetta conoscenza dei principi e dei teoremi. Per funzioni di una certa complessità può risultare difficile individuare la successione delle operazioni necessarie per trasformare la funzione nella forma più semplice possibile (forma minima).

La tecnica delle mappe di Karnaugh, descritta nel par. 2.7, consente di effettuare la semplificazione della funzione in maniera quasi automatica.

Per semplificare funzioni che presentano la negazione di espressioni con due o più variabili si ricorre ai teoremi di De Morgan. Ad esempio nell'espressione $\overline{A+B}$ le negazioni non si possono semplificare direttamente, perché quella superiore è riferita a $(\overline{A+B})$ mentre quella inferiore alla sola variabile A . Per spezzare la linea continua di negazione bisogna applicare il teorema di De Morgan ottenendo così: $\overline{A+B} = \overline{A} \cdot \overline{B}$. Si veda a proposito l'esempio 2.14.

ESEMPIO 2.12

Semplificare la seguente espressione: $Y = \overline{\overline{A}BC} + \overline{\overline{A}C}$

Soluzione

Si applica il 1° teorema di De Morgan ad entrambi i termini:

$$Y = \overline{\overline{A}BC} + \overline{\overline{A}C}$$

e per la proprietà della negazione $\overline{\overline{A}} = A$ si trova:

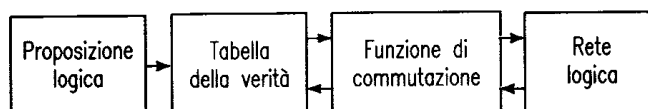
$$Y = \overline{\overline{A}BC} + \overline{\overline{A}C}$$

ma poiché $A + \overline{A} = 1$ e ogni somma logica con 1 dà sempre 1 si ottiene:

$$Y = 1$$

2.6 Tabella della verità e funzione di commutazione

Fig. 2.21



Def. Si definisce **mintermine** il prodotto logico di tutte le variabili d'ingresso vere o negate.
Si definisce **maxtermine** la somma logica di tutte le variabili d'ingresso vere o negate.

Riferendosi ad una rete con n ingressi, è possibile individuare 2^n mintermini ed altrettanti maxtermini.

ESEMPIO 2.15

Per una rete a tre ingressi (A, B, C), elencare tutti i possibili mintermini e maxtermini.

Soluzione

Mintermini: $ABC, ABC\bar{C}, \bar{A}BC, \bar{A}B\bar{C}, \bar{A}BC\bar{C}, \bar{A}\bar{B}C, \bar{A}\bar{B}\bar{C}$.

Maxtermini: $A+B+C, A+B+\bar{C}, A+\bar{B}+C, A+\bar{B}+\bar{C}, \bar{A}+B+C, \bar{A}+B+\bar{C},$
 $\bar{A}+\bar{B}+C, \bar{A}+\bar{B}+\bar{C}$

Come si è già visto nel paragrafo precedente, una funzione di commutazione può essere rappresentata con infinite espressioni logiche equivalenti; applicando le regole dell'algebra di Boole è possibile passare da una espressione all'altra.

Def. Due forme notevoli, per esprimere una funzione di commutazione sono:

- forma canonica SP** (*Somma di Prodotti*) o disgiuntiva, costituita dalla somma di alcuni mintermini;
- forma canonica PS** (*Prodotti di Somme*) o congiuntiva, costituita dal prodotto di alcuni maxtermini.

2.6.1 Dalla tabella della verità alla funzione di commutazione

Data una tabella della verità, si descrivono le tecniche con cui è possibile ricavare la corrispondente funzione di commutazione in forma canonica SP o PS.

ESEMPIO 2.16

Ricavare la funzione di commutazione, nelle forme canoniche SP e PS, corrispondente alla tabella della verità in figura.

A	B	C	Y	Mintermini	Maxtermini
0	0	0	1	$\bar{A}\bar{B}\bar{C}$	
0	0	1	0		$A+B+\bar{C}$
0	1	0	1	$\bar{A}B\bar{C}$	
0	1	1	0		$A+\bar{B}+\bar{C}$
1	0	0	1	$A\bar{B}\bar{C}$	
1	0	1	1	$A\bar{B}C$	
1	1	0	1	$AB\bar{C}$	
1	1	1	0		$\bar{A}+\bar{B}+\bar{C}$

Forma canonica PS

$$Y = (A+B+\bar{C})(A+\bar{B}+\bar{C})(\bar{A}+\bar{B}+\bar{C})$$

Forma canonica SP

$$Y = \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + A\bar{B}\bar{C} + A\bar{B}C + AB\bar{C}$$

Fig. 2.22

Soluzione

Come evidenziato in figura si ricava la funzione di commutazione in forma canonica SP:

$$Y = \overline{A}\overline{B}\overline{C} + \overline{A}B\overline{C} + A\overline{B}\overline{C} + A\overline{B}C + ABC$$

e in forma canonica PS:

$$Y = (A+B+\overline{C})(A+\overline{B}+\overline{C})(\overline{A}+\overline{B}+\overline{C})$$



Per esprimere una funzione in forma canonica SP: si scrive la somma dei mintermini corrispondenti agli 1 della variabile d'uscita; nei mintermini le variabili d'ingresso compariranno negate se il loro valore in tabella è 0 o vere se il valore è 1.



Per esprimere una funzione in forma canonica PS: si scrive il prodotto dei maxtermini corrispondenti agli 0 della variabile d'uscita; nei maxtermini le variabili d'ingresso compariranno negate se il loro valore in tabella è 1 o vere se il valore è 0.

Per dimostrare la validità delle due tecniche illustrate si osservi che:

- un mintermine assume valore 1 solo in corrispondenza della combinazione di variabili d'ingresso che l'ha generato, e quindi la forma canonica SP assume valore 1 quando uno dei mintermini sommati vale 1.
- un maxtermine assume valore 0 solo in corrispondenza della combinazione di variabili d'ingresso che l'ha generato, per cui la forma canonica PS assume valore 0 quando uno dei maxtermini messi a prodotto vale 0.

Le reti logiche derivate da forme SP presentano porte AND al primo livello ed una porta OR al secondo (reti AND-OR), quelle derivate da forme PS presentano porte OR al primo livello ed una porta AND al secondo (reti OR-AND).

Le forme canoniche generalmente possono essere semplificate applicando le regole dell'algebra di Boole.

Def. Una funzione di commutazione viene detta in **forma minima SP** se presenta il minor numero possibile di prodotti con, ognuno, il minor numero di variabili; di conseguenza darà origine ad una rete a due livelli con il minor numero possibile di porte logiche.

Def. La **forma minima PS** presenta il minor numero possibile di somme con, ognuna, il minor numero di variabili.

Le quantità di porte logiche necessarie a realizzare le due forme minime sono equivalenti.



Strategia di semplificazione delle forme canoniche SP con l'algebra di Boole

- 1) Individuare coppie di mintermini tra cui sia possibile raccogliere tutte le variabili a parte una, che sarà vera in un termine e negata nell'altro.

Es.: $ABC + \overline{A}BC = BC(A + \overline{A})$ (proprietà distributiva del prodotto).

- 2) È possibile impiegare più volte uno stesso mintermine (per la proprietà $A = A + A$, che vale anche con termini di più variabili; es.: $ABC = ABC + ABC$).

3) La variabile non raccolta viene eliminata per le regole:

$$A + \bar{A} = 1 \text{ e } A \cdot 1 = A$$

$$\text{Es.: } BC(A + \bar{A}) = BC$$

4) Tentare di semplificare ulteriormente ricominciando dal punto 1.

5) Applicare anche le altre regole, quando è possibile.

ESEMPIO 2.17

Semplificare la funzione dell'esempio 2.16, espressa in forma canonica SP, fino ad ottenere la forma minima SP.

Soluzione

$$\begin{aligned} Y &= \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + A\bar{B}\bar{C} + A\bar{B}C + A\bar{B}\bar{C} = \\ &= \bar{A}\bar{C} + \bar{A}B\bar{C} + A\bar{B}\bar{C} + A\bar{B}C + A\bar{B}\bar{C} = \\ &= \bar{C} + A\bar{B} \end{aligned}$$



Strategia di semplificazione delle forme canoniche PS con l'algebra di Boole

1) Individuare coppie di maxtermini tra cui sia possibile raccogliere tutte le variabili a parte una, che sarà vera in un termine e negata nell'altro.

$$\text{Es. } (A + \bar{B} + C)(A + B + C) = (A + C) + \bar{B}B \text{ (proprietà distributiva della somma).}$$

2) È possibile accoppiare uno stesso maxtermine più volte

3) La variabile non raccolta viene eliminata per le regole:

$$A\bar{A} = 0 \text{ e } A + 0 = A$$

$$\text{Es.: } A + C + \bar{B}B = A + C$$

4) Tentare di semplificare ulteriormente ricominciando dal punto 1.

5) Applicare anche le altre regole quando è possibile.

Un'ulteriore semplificazione sarà possibile se l'espressione ottenuta contiene coppie di termini del tipo: $AB + \bar{A}B$ che rappresentano l'espressione algebrica dell'EXOR, oppure coppie del tipo $\bar{A}B + AB$ che rappresentano l'espressione algebrica dell'EXNOR.

ESEMPIO 2.18

Semplificare la funzione dell'esempio 2.16, espressa in forma canonica PS, fino ad ottenere la forma minima PS.

Soluzione

$$\begin{aligned} Y &= (A + B + \bar{C})(A + \bar{B} + \bar{C})(\bar{A} + \bar{B} + C) = \\ &= (A + \bar{C})(\bar{B} + C) = \\ &= \bar{C} + A\bar{B} \text{ (raccolgendo } \bar{C} \text{)} \end{aligned}$$

Si noti l'identità con il risultato ottenuto nell'esempio 2.17, a partire dalla forma SP equivalente.

ESEMPIO 2.19

Semplificare l'espressione: $Y = A\bar{C} + AC + \bar{B}C + B\bar{C}$

Soluzione

$$Y = A + (B \oplus C)$$

Le strategie di semplificazione appena descritte, per ottenere la funzione di commutazione in forma minima, vengono applicate automaticamente utilizzando la tecnica delle mappe di Karnaugh, illustrata nel par. 2.7.

2.6.2 Dalla funzione di commutazione alla tabella della verità

Si propongono tre tecniche per la compilazione della tabella della verità, nota la funzione di commutazione.



1) Compilazione della tabella della verità, calcolando la funzione di commutazione per ogni combinazione delle variabili d'ingresso:

Si compila la colonna Y della tabella riga per riga, calcolando la funzione per ogni combinazione degli ingressi, facendo attenzione a rispettare le priorità corrette nell'esecuzione delle operazioni (si veda l'es. 2.20).



Per funzioni di una certa complessità il calcolo può divenire laborioso ed aumentare la probabilità di commettere errori.

ESEMPIO 2.20

Ricavare la tabella della verità della funzione:

$$Y = (\bar{A}B + \bar{D})C$$

Soluzione

Per ogni combinazione degli ingressi (riga della tabella), si calcola il corrispondente valore dell'uscita Y.

A	B	C	D	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	0

Fig. 2.23



2) Compilazione della tabella della verità espandendo la funzione nella forma canonica SP:

- a) Si ricava la forma canonica SP della funzione con i seguenti passaggi:
- a1) portare la funzione in forma SP utilizzando le regole dell'algebra di Boole;
 - a2) moltiplicare ogni termine prodotto che non contiene una delle variabili d'ingresso, supponiamo ad esempio la A, per un'espressione del tipo $(A + \bar{A})$; ad es. $\bar{B}\bar{C} \rightarrow \bar{B}\bar{C}(A + \bar{A})$.
- Naturalmente la funzione non viene alterata, perché $(A + \bar{A}) = 1$.
- a3) Riportare la funzione in forma SP applicando la proprietà distributiva del prodotto.
- Es. $\bar{B}\bar{C}(A + \bar{A}) = \bar{B}\bar{C}A + \bar{B}\bar{C}\bar{A}$
- a4) Se qualche termine prodotto non contiene ancora tutte le variabili d'ingresso, ripetere i passaggi a1) e a2) fino ad ottenere la forma canonica SP. Se qualche mintermine compare più volte, quelli doppi verranno eliminati (teorema dell'idempotenza: $A + A = A$)
- b) Si compila poi la colonna Y della tabella inserendo un 1 a fianco delle combinazioni delle variabili d'ingresso corrispondenti ai mintermini.

Si analizzi l'esempio 2.21.

ESEMPIO 2.21

Ricavare la tabella della verità della funzione:

$$Y = \overline{ABC} + ABC$$

Soluzione

- a) Si porta la funzione in forma SP utilizzando le regole dell'algebra di Boole:

$$Y = \bar{A} + \bar{B}\bar{C} + ABC = \bar{A} + BC + ABC =$$

e si ricava la forma canonica SP:

$$\begin{aligned} &= \bar{A}(B + \bar{B}) + BC(A + \bar{A}) + ABC = \\ &= \bar{A}B + \bar{A}\bar{B} + ABC + \bar{A}BC + ABC = \\ &= \bar{A}B(C + \bar{C}) + \bar{A}\bar{B}(C + \bar{C}) + ABC + \bar{A}BC + ABC = \\ &= \bar{A}BC + \bar{A}B\bar{C} + \bar{A}\bar{B}C + \bar{A}\bar{B}\bar{C} + ABC + ABC \end{aligned}$$

- b) Si compila la tabella della verità ponendo un 1, nella colonna Y, in corrispondenza ad ogni mintermine trovato:

A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Fig. 2.24



3) Compilazione della tabella della verità direttamente dalla funzione in forma SP (non canonica):

- Portare la funzione in forma SP utilizzando le regole dell'algebra di Boole.
- Per ognuno dei termini prodotto porre un 1 nella colonna dell'uscita, in corrispondenza di tutte le righe che contengono la corrispondente combinazione delle variabili d'ingresso.

Si veda a proposito l'esempio 2.22.

Questa tecnica è consigliata in quanto è la più rapida tra quelle proposte.

ESEMPIO 2.22

Compilare la tabella della verità relativa alla funzione:

$$Y = A(\bar{B} + B\bar{C}) + BC$$

Soluzione

- Si porta la funzione in forma SP:

$$Y = A\bar{B} + AB\bar{C} + BC$$

- Si compila la tabella della verità considerando che Y vale 1 quando un termine prodotto della forma SP vale 1; ad es. il termine $A\bar{B}$ vale 1 per $A=1$ e $B=0$, indipendentemente da C , e quindi la Y varrà 1 nelle righe 100 e 101:

A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Fig. 2.25

2.7 Minimizzazione tramite mappe di Karnaugh

La tecnica delle mappe di Karnaugh permette, a partire dalla tabella della verità, di ricavare la funzione di commutazione direttamente in forma minima, senza dover individuare ed applicare le regole dell'algebra di Boole.

2.7.1 Impostazione e compilazione delle mappe

Per realizzare la mappa corrispondente ad una tabella della verità, occorre applicare le seguenti regole:

- una mappa è costituita da tante caselle quante sono le righe della tabella della verità; quindi con n variabili d'ingresso si avranno 2^n caselle, disposte a rettangolo;
- ogni lato della mappa può essere di 2 o 4 caselle. Di conseguenza si avranno mappe 2×2 per due variabili d'ingresso, 2×4 per tre variabili e 4×4 per quattro variabili.

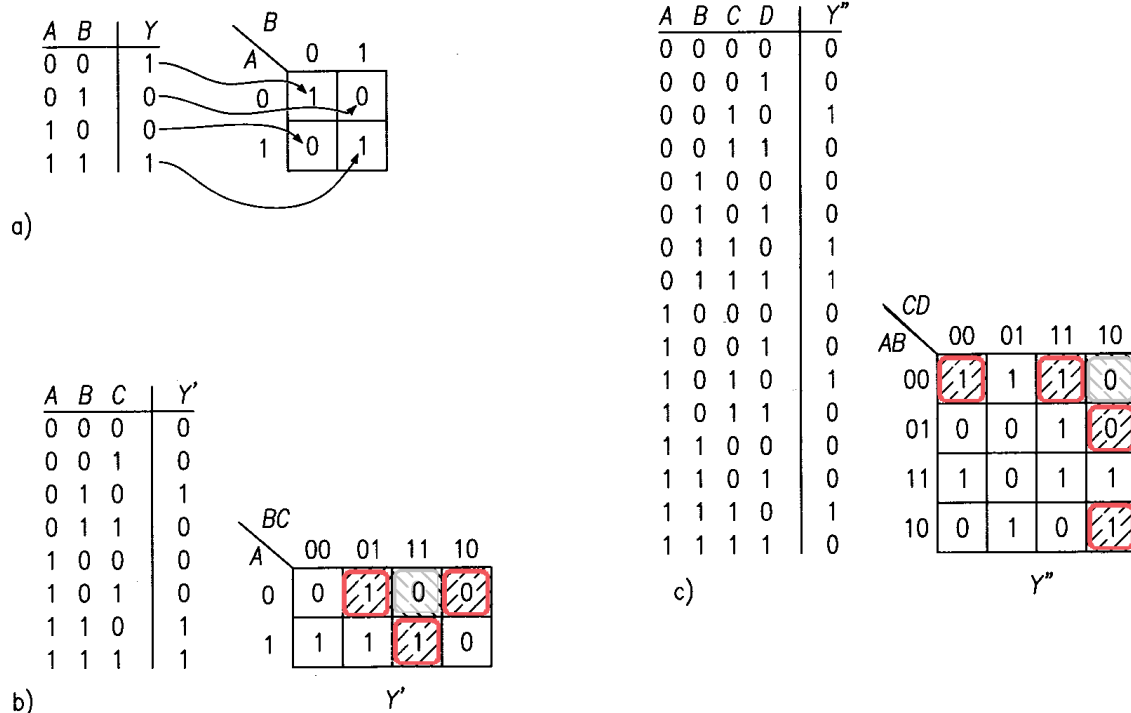
Per cinque variabili si scrivono due mappe 4×4 di cui una relativa al valore 0 e l'altra al valore 1 della quinta variabile. Per sei variabili si avranno quattro mappe 4×4 .

- c) Riportando le coordinate delle caselle sui lati della mappa è necessario che nel passaggio da una riga o da una colonna all'altra adiacente, cambi il valore di una sola variabile d'ingresso. La progressione, nel caso di lato composto da quattro celle, dovrà essere quindi:

00, 01, 11, 10

- d) All'interno delle caselle si scrivono i valori della variabile d'uscita, corrispondenti ai valori delle variabili d'ingresso che rappresentano le coordinate delle caselle.

Nella fig. 2.26 sono evidenziate le relazioni tra le mappe di funzioni a due, tre e quattro variabili e le corrispondenti tabelle della verità.



Corrispondenza tra tabelle della verità e mappe di Karnaugh con a) due, b) tre e c) quattro variabili d'ingresso.

Fig. 2.26

Def. Due caselle si dicono **adiacenti**, se le loro coordinate differiscono per il valore di una sola variabile.

Nelle mappe delle fig. 2.26b e 2.26c, per esempio, le caselle tratteggiate in colore sono adiacenti a quelle tratteggiate in nero.

Si noti che le caselle lungo un lato della tabella sono adiacenti alle corrispondenti caselle del lato opposto, poiché le loro coordinate differiscono per una sola variabile.

2.7.2 Come ricavare da una mappa la funzione di commutazione in forma minima



Per ottenere da una mappa la **funzione di commutazione in forma minima SP** e di conseguenza uno schema circuitale contenente il minimo numero di porte logiche, si segue il seguente procedimento:

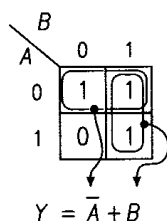
- occorre racchiudere tutti gli 1 della mappa in gruppi che possono essere costituiti da 1, 2, 4, 8 o 16 caselle adiacenti (le forme possibili per i gruppi sono rettangoli con lati di 1, 2 o 4 caselle);
- la scelta dei raggruppamenti deve essere tale da riuscire a racchiudere tutti gli 1 contenuti nella mappa, nel minor numero di gruppi, ognuno di dimensioni maggiori possibili;
- è possibile racchiudere lo stesso 1 in più gruppi, se ciò contribuisce ad aumentarne le dimensioni;
- la funzione di commutazione sarà costituita dalla somma di tanti termini prodotto quanti sono i gruppi ottenuti sulla mappa;
- in ogni termine prodotto compariranno solo le variabili d'ingresso che, come coordinate del gruppo corrispondente, non cambiano valore all'interno del gruppo. Compariranno negate se il loro valore è 0, oppure vere se il valore è 1.

ESEMPIO 2.23

Ricavare le funzioni di commutazione in forma minima SP relative alle tabelle della verità in figura, mediante le mappe di Karnaugh.

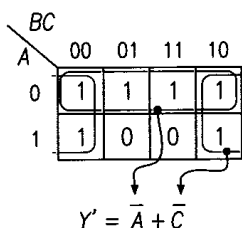
A	B	Y
0	0	1
0	1	1
1	0	0
1	1	1

a)



A	B	C	Y'
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

b)



A	B	C	D	Y''
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

c)

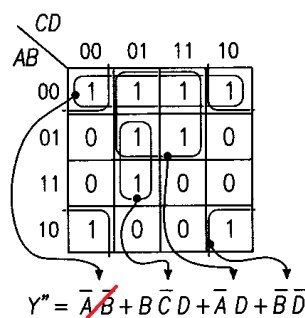


Fig. 2.27

Soluzione

Si compila la mappa relativa ad ogni tabella, si formano i gruppi di 1 e si ricavano le corrispondenti funzioni di commutazione in forma minima SP, indicate a fianco delle mappe.



Per ottenere da una mappa la **funzione di commutazione in forma minima PS** si segue il seguente procedimento:

- occorre racchiudere tutti gli 0 della mappa in gruppi rettangolari che possono essere costituiti da 1, 2, 4, 8 o 16 caselle adiacenti;
- la scelta dei raggruppamenti deve essere tale da riuscire a racchiudere tutti gli 0 della mappa nel minor numero di gruppi, ognuno di dimensioni maggiori possibili;
- è possibile racchiudere lo stesso 0 in più di un gruppo, se ciò contribuisce ad aumentarne le dimensioni.
- la funzione di commutazione sarà costituita dal prodotto di tanti termini somma quanti sono i gruppi ottenuti sulla mappa;
- in un termine somma, le variabili d'ingresso che mantengono un valore costante pari a 1 all'interno del gruppo compariranno negate, mentre compariranno vere se il valore costante nel gruppo è 0.

ESEMPIO 2.24

Ricavare la funzione di commutazione in forma minima PS, relativa alla mappa di Karnaugh in figura.

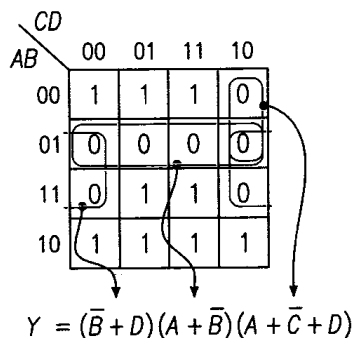


Fig. 2.28



Si noti che un gruppo formato da una sola casella origina un termine contenente tutte le variabili d'ingresso (mintermine o maxtermine). Ad ogni raddoppio del numero di caselle del gruppo, diminuisce di uno il numero delle variabili contenute nel termine prodotto o somma.

Le reti logiche corrispondenti alle due forme minime SP o PS, hanno un livello di complessità simile.

Un criterio di scelta tra le due forme minime può dipendere dalla componentistica che si intende utilizzare, ad esempio dalla necessità di avere porte AND al primo livello ed OR al secondo, o viceversa.

Per trovare la forma minima della funzione occorre comporre il minor numero di gruppi ognuno il più grande possibile; ad esempio il raggruppamento effettuato in fig. 2.29 non fornisce la funzione in forma minima.

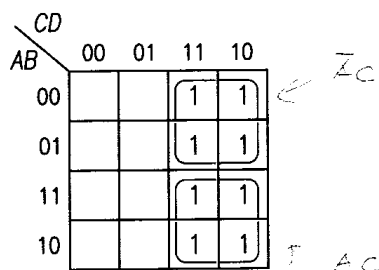


Fig. 2.29

2.7.3 Mappe per funzioni a 5 o 6 variabili d'ingresso

Nel caso di **5 variabili d'ingresso**, le 32 caselle saranno organizzate in due sottomappe quadrate da 16 caselle ognuna.

Una sottomappa sarà relativa al valore 0 della quinta variabile (E), l'altra al valore 1.

Le due sottomappe possono pensarsi come sovrapposte e risultano adiacenti le caselle che si trovano nella medesima posizione (infatti commuta solo la variabile E).

Il procedimento per ricavare la funzione minima è analogo a quello esposto, facendo attenzione che ora è possibile formare gruppi costituiti da caselle appartenenti a entrambe le sottomappe, come si vede nell'esempio 2.25.

ESEMPIO 2.25

Ricavare l'espressione in forma minima SP della funzione di cinque variabili rappresentata nella mappa in figura.

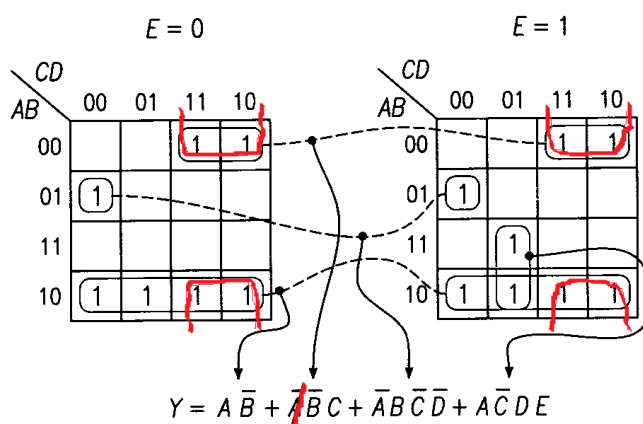


Fig. 2.30

Soluzione

Si formano i gruppi e si ricava la funzione in forma minima, come indicato in figura. Si noti che i primi tre termini, ottenuti da gruppi contenenti caselle di entrambe le sottomappe, non presentano la variabile E .

Nel caso di **6 variabili d'ingresso**, le 64 caselle saranno organizzate in quattro sottomappe quadrate da 16 caselle ognuna. Ogni sottomappa è individuata da una diversa combinazione di valori della quinta e della sesta variabile (E, F).

Valgono le considerazioni fatte per le funzioni di cinque variabili, tenendo presente che ora sono adiacenti le caselle corrispondenti di due sottomappe adiacenti (cioè tra cui varia solo una delle due variabili E ed F).

$F \oplus E \oplus A \oplus B \oplus C \oplus D$
 $\begin{matrix} 00 \\ 01 \\ 10 \\ 11 \end{matrix}$

ESEMPIO 2.26

Ricavare l'espressione in forma minima SP della funzione di sei variabili rappresentata nella mappa in figura.

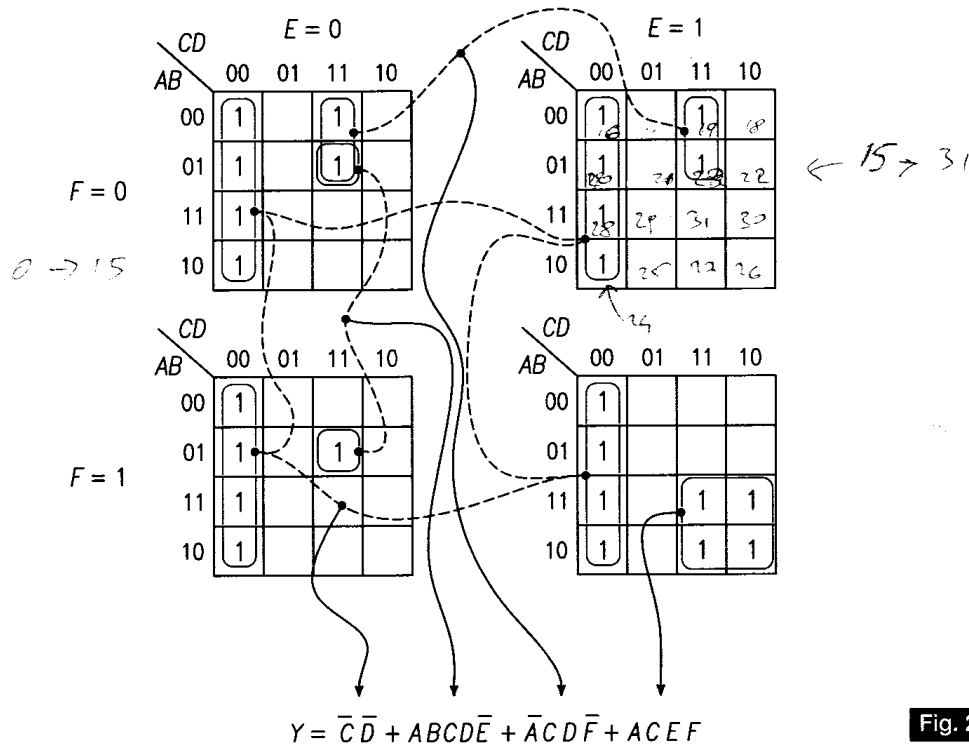


Fig. 2.31

Soluzione

Si formano i gruppi e si ricava la funzione in forma minima, come indicato in figura. Si noti che il gruppo relativo al primo termine contiene caselle di tutte le sottomappe e quindi le variabili E ed F non compaiono nel termine. Considerazioni analoghe valgono per gli altri termini.

2.7.4 Condizioni d'indifferenza

Nel compilare la tabella della verità e quindi la mappa relativa ad una proposizione logica, può capitare che in corrispondenza di alcune combinazioni delle variabili d'ingresso, il valore assunto dall'uscita non abbia importanza.

Ciò può accadere o per una effettiva impossibilità che nel caso in esame si presenti in ingresso quella combinazione di variabili, o perché, quando si verifica, il valore assunto dall'uscita è indifferente.

Def. Si definisce **condizione d'indifferenza** (*don't care*) una combinazione di variabili d'ingresso per cui il valore dell'uscita può essere indifferentemente 0 o 1; la corrispondente funzione viene detta parzialmente specificata.

In corrispondenza di una condizione d'indifferenza, la variabile d'uscita verrà contrassegnata con una X nella tabella e nella mappa.

In fase di formazione dei gruppi in una mappa, i *'don't care'* hanno la funzione di jolly

e si può assegnare loro valore 1 o 0 a seconda che le caselle corrispondenti contribuiscano o meno a formare gruppi di dimensioni maggiori e quindi termini più semplici nella funzione di commutazione.

In corrispondenza di una combinazione d'ingressi individuata come condizione d'indifferenza, l'uscita assumerà valore 1 o 0 a seconda che la X sia stata associata ad un gruppo di 1 o di 0.

Nel cap. 3 sono riportati esempi di proposizioni logiche che generano funzioni parzialmente specificate.

ESEMPIO 2.28

Ricavare la funzione di commutazione in forma minima relativa alla tabella della verità parzialmente specificata nella fig. 2.32a.

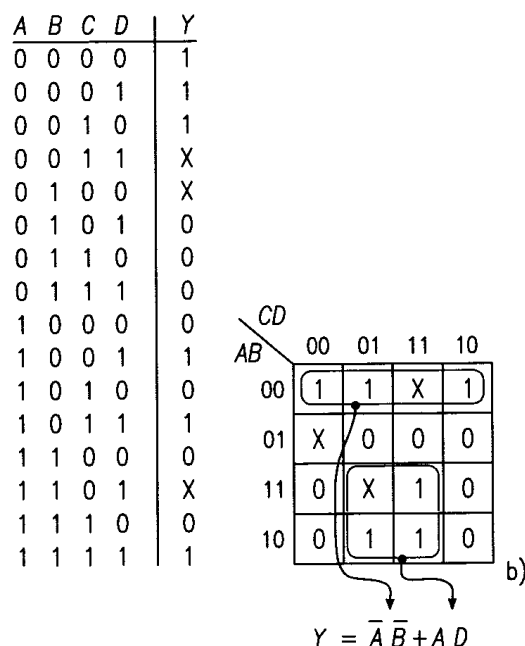


Fig. 2.32

Soluzione

Dalla tabella della verità si compila la mappa di Karnaugh corrispondente (fig. 2.32b) e si ricava la funzione:

$$Y = \bar{A}\bar{B} + AD$$

2.7.5 Utilizzo delle mappe per minimizzare una data funzione di commutazione



Le mappe di Karnaugh possono anche essere utilizzate per minimizzare una funzione di commutazione, con il seguente procedimento:

- 1) si porta la funzione in forma SP;
- 2) si compila la tabella della verità, come descritto nel par. 2.6;

3) si compila la mappa relativa e si ricava la funzione di commutazione in forma minima.

È possibile compilare direttamente la mappa a partire dalla funzione in forma SP, individuando i gruppi di 1 corrispondenti ai termini prodotto dell'espressione, come mostrato nell'esempio 2.28; si ricava poi la funzione minimizzata.

ESEMPIO 2.28

Semplificare la funzione: $Y = \bar{A} \bar{B} \bar{C} \bar{D} + \bar{A} \bar{B} \bar{C} + \bar{A} \bar{C} \bar{D} + \bar{A} \bar{D}$

Soluzione

- 1) Si compila la mappa individuando il gruppo di 1 corrispondente ad ogni termine della funzione (fig. 2.33a). Ad esempio al termine $\bar{A} \bar{D}$ corrisponderà il gruppo di 4 caselle di coordinate $A = 1$ e $D = 0$.

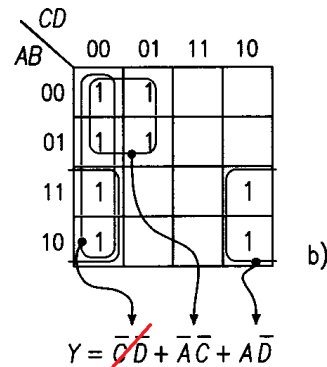
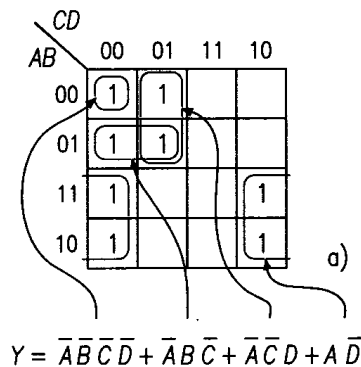


Fig. 2.33

- 2) Raggruppando poi in maniera più conveniente si ricava la funzione in forma minima (fig. 2.33b):

$$Y = \bar{C} \bar{D} + \bar{A} \bar{C} + \bar{A} \bar{D}$$

2.8 Realizzazione di una rete a sole porte NAND o NOR

Come già evidenziato nella tab. 2.4, è possibile sostituire a ognuna delle tre porte fondamentali (NOT, AND, OR), una rete equivalente a sole porte NAND o a sole porte NOR. Di conseguenza qualunque funzione di commutazione può essere realizzata con sole porte universali.

L'utilità di questa osservazione risiede nel fatto che, l'impiego di porte omogenee, può portare ad una riduzione del numero dei circuiti integrati necessari a realizzare il circuito, come si vede negli esempi che seguono.

Def. Questa operazione, detta *decomposizione funzionale*, può essere compiuta molto semplicemente a partire da forme SP e PS, e quindi dalle relative reti AND-OR e OR-AND, seguendo le seguenti regole pratiche.

2.8.1 Dalla rete AND-OR alla rete a soli NAND



- 1) Si sostituiscono tutte le porte (AND e OR) con porte NAND;
- 2) le variabili d'ingresso che entrano, vere o negate, direttamente nell'OR devono essere complementate nella rete a NAND;
- 3) gli eventuali NOT sono sostituiti da NAND con gli ingressi collegati insieme.

Dimostrazione:

la forma SP si può indicare così

$$Y = P_1 + P_2 + \dots + P_N$$

dove P_1, P_2, \dots, P_N rappresentano gli N prodotti (AND).

Per il teorema di De Morgan vale che:

$$P_1 + P_2 + \dots + P_N = \overline{\overline{P_1} \cdot \overline{P_2} \cdot \dots \cdot \overline{P_N}}$$

il che significa che la funzione può essere realizzata con N porte NAND al posto degli AND e da una porta NAND a N ingressi al posto dell'OR.

ESEMPIO 2.28

Trasformare la rete in AND-OR in figura in una rete a soli NAND.

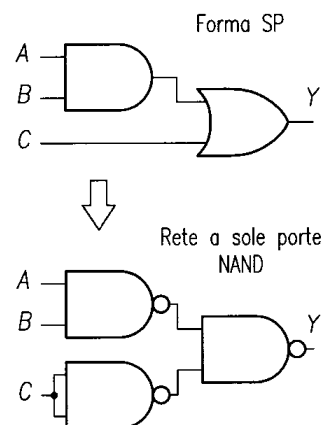


Fig. 2.34

Soluzione

Si sostituisce ad ogni porta AND e OR una porta NAND. Si noti che la variabile C , entrando vera nell'OR, deve essere complementata nella rete a soli NAND.

Si faccia riferimento all'esercitazione di laboratorio n. 4.

2.8.2 Dalla rete OR-AND alla rete a soli NOR



- 1) Si sostituiscono tutte le porte (OR e AND) con porte NOR;
- 2) le variabili d'ingresso che entrano, vere o negate, direttamente nell'AND devono essere complementate nella rete a NOR;
- 3) gli eventuali NOT sono sostituiti da NOR con gli ingressi collegati insieme.

Dimostrazione:

la forma PS si può indicare così $Y = S_1 \cdot S_2 \dots S_N$
dove S_1, S_2, \dots, S_N rappresentano le N somme (OR).

Per il teorema di De Morgan vale che:

$$S_1 \cdot S_2 \dots S_N = \overline{\overline{S_1} \cdot \overline{S_2} + \dots + \overline{S_N}}$$

il che significa che la funzione può essere realizzata con N porte NOR al posto degli OR e da una porta NOR a N ingressi al posto dell'AND.

Naturalmente, se la funzione che descrive la rete non si trova in forma SP o PS, è sempre possibile portarvela, applicando le regole dell'algebra di Boole.

ESEMPIO 2.30

Trasformare la rete OR-AND in figura in una rete a soli NOR.

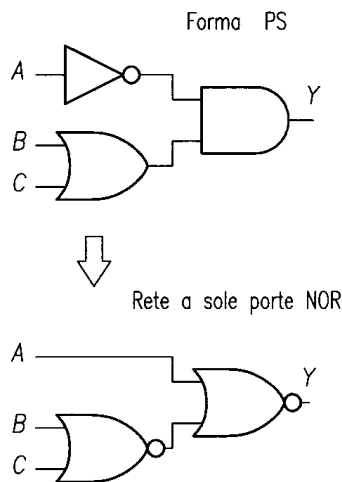


Fig. 2.35

Soluzione

Si sostituisce ad ogni porta OR e AND una porta NOR. Si noti che la variabile A , entrando negata nell'AND, viene complementata e quindi entra vera nel NOR.



Si noti che, poiché ogni integrato contiene 4 porte omogenee a due ingressi, il numero di circuiti integrati necessari nell'esempio 2.29 si è ridotto da 2 a 1, mentre nell'esempio 2.30 si è ridotto da 3 a 1.

2.9 Le porte logiche nei circuiti integrati

La teoria presentata in questo capitolo trova applicazione in diversi settori come l'elettromeccanica, la fluidodinamica e l'elettronica. Le porte logiche elettroniche sono contenute in circuiti integrati, vale a dire in circuiti, costituiti prevalentemente da transistor, tutti realizzati su di un'unica piastrina di silicio (*chip*).

Il chip viene poi incapsulato in un contenitore (*package*) plastico o ceramico che presenta generalmente piedini (*pin*) metallici disposti su due file. Sul contenitore viene stampato il codice che individua la funzione svolta dall'integrato.

Si riportano a titolo d'esempio, in fig. 2.36, i *pin-out* di alcuni circuiti integrati che contengono porte logiche.

Tutti gli integrati illustrati appartengono alla famiglia logica TTL, le cui caratteristiche elettriche sono esposte nel cap. 6.

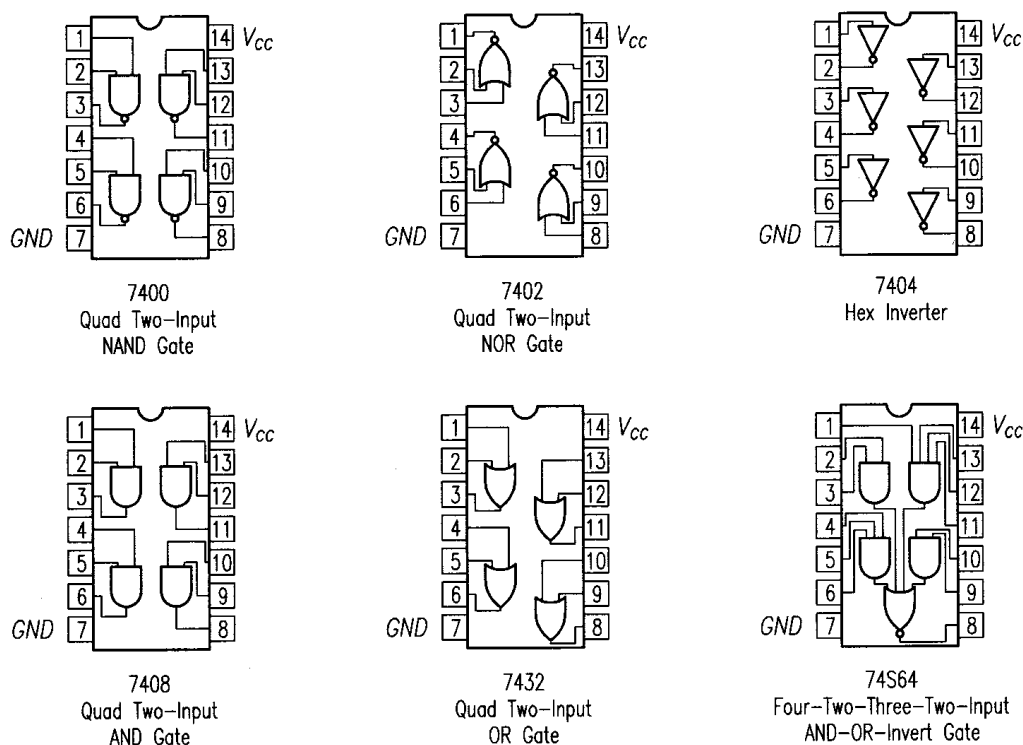


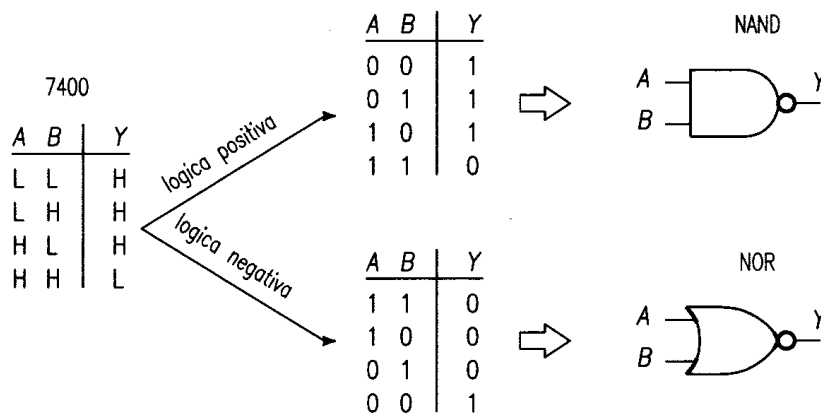
Fig. 2.36 Pin-out di alcuni circuiti integrati che contengono porte logiche.

Nell'appendice C del testo, sono riportate le sigle degli integrati che contengono porte logiche ed i pin-out di alcuni di questi. Nell'introduzione alla parte di laboratorio, vengono fornite alcune istruzioni per l'utilizzo dei circuiti integrati.

Utilizzando un computer connesso alla rete internet, è possibile consultare i data-book di alcune case costruttrici di circuiti integrati.



Generalmente i costruttori riportano, nelle tabelle della verità, i livelli di tensione L e H e non i valori logici 0 e 1. In questo modo la tabella risulta valida sia nel caso di utilizzo in *logica positiva* ($L \rightarrow 0$, $H \rightarrow 1$) che in quello di utilizzo in *logica negativa* ($L \rightarrow 1$, $H \rightarrow 0$), mentre la funzione logica svolta nei due casi è diversa. Si noti infatti (fig. 2.37) che le porte contenute nell'integrato 7400 si comportano



Le porte logiche contenute nell'integrato 7400 si comportano da NAND in logica positiva e da NOR in logica negativa.

Fig. 2.37

da NAND in logica positiva e da NOR in logica negativa. Per non creare ambiguità alcuni costruttori identificano l'integrato come "POSITIVE NAND GATES".

Come si è visto nel corso del capitolo, è molto frequente la necessità di realizzare reti di tipo AND-OR. Per ridurre il numero degli integrati necessari, i costruttori mettono a disposizione delle strutture AND-OR contenute in un unico circuito (**strutture AOI: And-Or-Invert**). Si veda a proposito l'esempio 2.31.

ESEMPIO 2.31

Realizzare con l'integrato 74S64 (pin out in fig. 2.36), contenente una struttura AOI, la funzione logica espressa dalla mappa in fig. 2.38a

CD \ AB	00 01 11 10			
	00	01	11	10
00	1	1	X	1
01	1	X	1	1
11	1	0	0	0
10	1	0	0	1

a)

Y

CD \ AB	00 01 11 10			
	00	01	11	10
00	0	0	X	0
01	0	X	0	0
11	0	1	1	1
10	0	1	1	0

b)

\bar{Y}

Fig. 2.38

Soluzione

Poiché la struttura contiene una rete AND-OR, si ricava dalla mappa la funzione espressa nella forma SP. Per compensare la negazione in uscita alla rete è necessario complementare tutta la mappa (fig. 2.38b), ricavando l'espressione: $\bar{Y} = AD + ABC$.

Nella fig. 2.39 sono riportati i collegamenti degli ingressi dell'integrato.

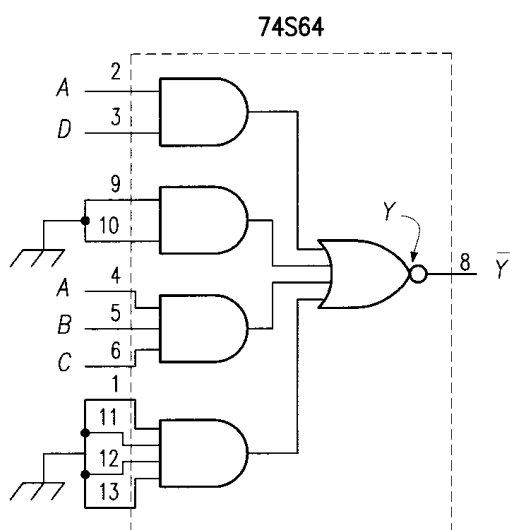


Fig. 2.39