CAPITOLO 1

CIRCUITI COMBINATORI

Con questo capitolo iniziamo lo studio dell'*elettronica digitale*, partendo dalle *porte logiche* che costituiscono i circuiti digitali più elementari. In altre parole, un circuito digitale, per quanto complesso, può essere ricondotto ad un insieme di porte logiche elementari variamente connesse tra loro. L'insieme di tali porte logiche e delle loro connessioni prende il nome di *circuito logico*.

Connettendo tra loro più porte logiche si realizzano circuiti più complessi che prendono il nome di *circuiti combinatori*. Introducendo degli elementi di *memoria* (e aumentando la complessità circuitale) si realizzano dei *circuiti sequenziali*.

Prima di affrontare l'analisi dei circuiti combinatori ed intraprendere lo studio dell'elettronica digitale, sarebbe bene fare un rapido esame di coscienza in modo da chiedersi cosa è rimasto nei propri neuroni del corso di Reti Logiche… Con tutta l'indulgenza possibile, si darà per scontato la conoscenza delle nozioni fondamentali dell'*aritmetica binaria* e dell'*algebra di Boole*. Infatti, l'aritmetica binaria costituisce il "linguaggio" dell'elettronica digitale, mentre l'algebra di Boole, rappresenta la "sintassi" di tale linguaggio, poiché fornisce uno strumento matematico per l'analisi del funzionamento dei circuiti digitali. Se vi trasferite negli Stati Uniti senza conoscere né la lingua, né la sua sintassi, non potrete neanche andare a lavare i piatti da McDonald's! Capita la metafora?...

Si definiscono *circuiti combinatori* una classe di circuiti – normalmente integrati – in cui lo stato delle uscite ad un certo istante dipende solo e sempre dallo stato presente degli ingressi nello stesso istante. In altri termini, tali circuiti sono *privi di memoria*, in quanto conservano lo stato delle uscite sino a quando permane quella particolare configurazione d'ingressi che vi ha dato origine.

Questa classe di circuiti comprende tutte le porte logiche (AND, OR, NOT, ecc.) già esaminate nel corso di reti logiche, oltre ad altri sistemi che consentono di realizzare funzioni tipiche frequentemente ricorrenti nei sistemi digitali, quali codifica e decodifica dei dati, multiplazione e demultiplazione, operazioni algebriche su numeri binari.

I circuiti digitali vengono realizzati con differenti tecniche che fanno riferimento ai due dispositivi elettronici fondamentali, di cui parleremo più avanti nei capitoli 7 e 8, cioè il MOSFET ed il BJT (talvolta chiamato semplicemente *transistor*). I circuiti digitali integrati a BJT

storicamente più utilizzati, fanno parte della famiglia TTL (*transistor-transistor logic*): essi vengono alimentati a 5 V e, in prima approssimazione, si può dire che il livello logico "0" corrisponde a 0 V, mentre il livello logico "1" corrisponde a +5 V. Le logiche a transistor costituiscono ormai una piccola fetta del mercato dei circuiti integrati digitali. Largamente utilizzate sono invece le logiche a MOSFET, ed in particolare quelle di tipo CMOS (*complementary MOS*): esse possono essere alimentate con differenti valori di tensione (sino a 18 V, anche se in genere i valori sono più bassi) e, conseguentemente, anche i valori di tensione corrispondenti ai livelli "0" e "1" possono variare. Molto utilizzate sono le logiche CMOS dette *TTL compatibili*, che vengono realizzate con dispositivi MOSFET, ma adoperano gli stessi livelli di tensione e la stessa alimentazione dei circuiti TTL. Studieremo più in dettaglio le famiglie logiche nel Cap. 11, quando i principi di funzionamento dei dispositivi elettronici saranno ormai noti.

1.1 Porte logiche

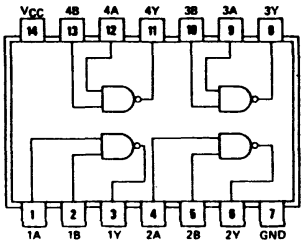
Nel corso di Reti Logiche è stato introdotto il concetto di porte logiche ed è stata data una breve descrizione delle principali funzioni svolte. Nel Cap. 11 vedremo che queste funzioni logiche possono essere realizzate da opportuni circuiti elettronici. Sebbene tali circuiti possano essere realizzati tramite uno o più BJT o MOSFET in commutazione connettendoli opportunamente, nella pratica si fa uso di *circuiti integrati*, i quali racchiudono al loro interno diverse porte logiche. Tali porte possono essere direttamente disponibili o essere assemblate in diversi modi in maniera tale da realizzare funzioni logiche sempre più complesse.

Limitandoci per ora soltanto alle semplici porte logiche integrate, bisogna notare che in commercio sono disponibili vari tipi di porte integrate, sia di tipo TTL che CMOS. Sono disponibili porte NOT, AND, OR, NAND, NOR, EXOR o talvolta combinazioni di esse. Esse possono essere a due o a più ingressi. Lo stesso tipo di porta è riprodotto più volte nello stesso integrato (disponibili tipicamente a 14 piedini), ad esempio ben 6 porte NOT trovano posto nell'integrato 7404.

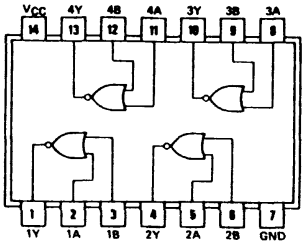
Una panoramica non esaustiva di porte logiche TTL e CMOS disponibili in commercio è riprodotta nelle seguenti Fig. 1.1 e Fig. 1.2.

1.2 Codificatori

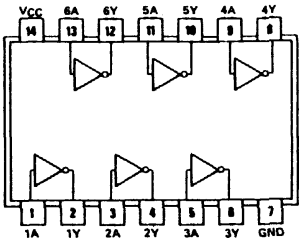
La funzione di un *codificatore (encoder) è* quella di rivelare la presenza di un livello attivo su una delle linee di ingresso fornendo sulle linee di uscita un determinato *codice binario* corrispondente alla linea attivata.



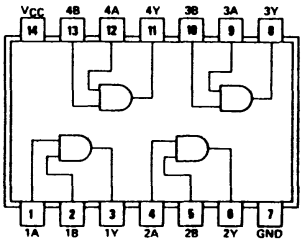
SN5400 (J) SN7400 (J, N)
SN54H00 (J) SN74H00 (J, N)
SN54L00 (J) SN74L00 (J, N)
SN54LS00 (J, W) SN74LS00 (J, N)
SN54S00 (J, W) SN74S00 (J, N)



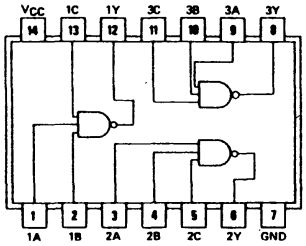
SN5402 (J) SN7402 (J, N)
SN54L02 (J) SN74L02 (J, N)
SN54LS02 (J, W) SN74LS02 (J, N)
SN54S02 (J, W) SN74S02 (J, N)



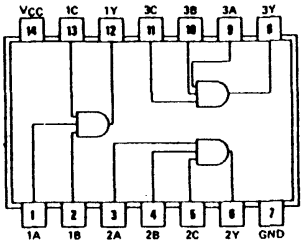
SN5404 (J) SN7404 (J, N)
SN54H04 (J) SN74H04 (J, N)
SN54L04 (J) SN74L04 (J, N)
SN54LS04 (J, W) SN74LS04 (J, N)
SN54S04 (J, W) SN74S04 (J, N)



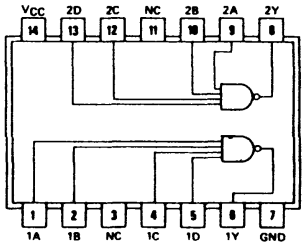
SN5408 (J, W) SN7408 (J, N)
SN54LS08 (J, W) SN74LS08 (J, N)
SN54S08 (J, W) SN74S08 (J, N)



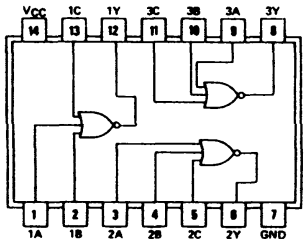
SN5410 (J) SN7410 (J, N)
SN54H10 (J) SN74H10 (J, N)
SN54L10 (J) SN74L10 (J, N)
SN54LS10 (J, W) SN74LS10 (J, N)
SN54S10 (J, W) SN74S10 (J, N)



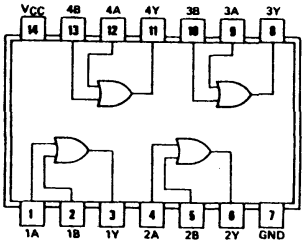
SN54H11 (J) SN74H11 (J, N)
SN54LS11 (J, W) SN74LS11 (J, N)
SN54S11 (J, W) SN74S11 (J, N)



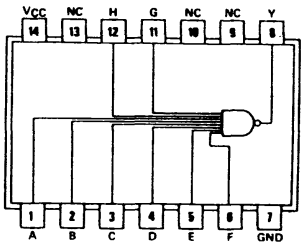
SN5420 (J) SN7420 (J, N)
SN54H20 (J) SN74H20 (J, N)
SN54L20 (J) SN74L20 (J, N)
SN54LS20 (J, W) SN74LS20 (J, N)
SN54S20 (J, W) SN74S20 (J, N)



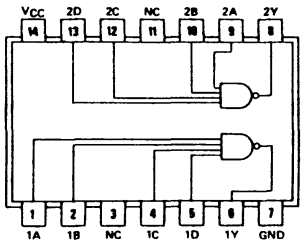
SN5427 (J, W) SN7427 (J, N)
SN54LS27 (J, W) SN74LS27 (J, N)



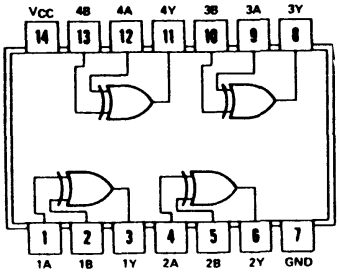
SN5432 (J, W) SN7432 (J, N)
SN54LS32 (J, W) SN74LS32 (J, N)
SN54S32 (J, W) SN74S32 (J, N)



SN5430 (J) SN7430 (J, N)
SN54H30 (J) SN74H30 (J, N)
SN54L30 (J) SN74L30 (J, N)
SN54LS30 (J, W) SN74LS30 (J, N)
SN54S30 (J, W) SN74S30 (J, N)



SN5440 (J) SN7440 (J, N)
SN54H40 (J) SN74H40 (J, N)
SN54LS40 (J,W) SN74LS40 (J, N)
SN54S40 (J, W) SN74S40 (J, N)

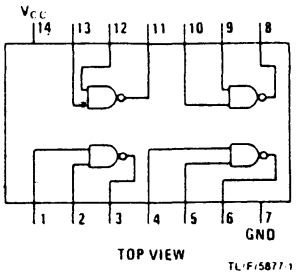


SN5486 (J, W) SN7486 (J, N)
SN54LS86 (J, W) SN74LS86 (J, N)
SN54S86 (J, W) SN74S86 (J, N)

Fig. 1.1 – *Porte logiche TTL*

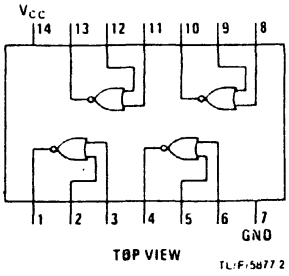
Dual-In-Line Packages

MM54C00/MM74C00



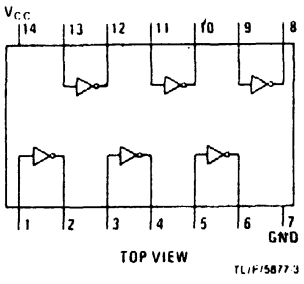
Order Number MM54C00J, MM74C00J, MM54C00N or MM74C00N
See NS Package J14A or N14A

MM54C02/MM74C02



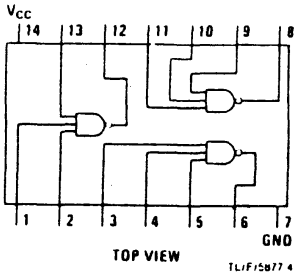
Order Number MM54C02J, MM74C02J, MM54C02N or MM74C02N
See NS Package J14A or N14A

MM54C04/MM74C04



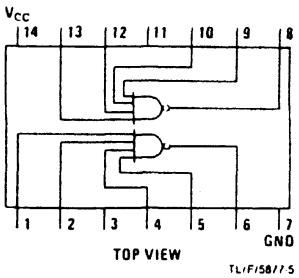
Order Number MM54C04J, MM74C04J, MM54C04N or MM74C04N
See NS Package J14A or N14A

MM54C10/MM74C10



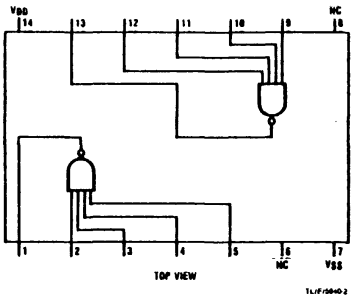
Order Number MM54C10J, MM74C10J, MM54C10N or MM74C10N
See NS Package J14A or N14A

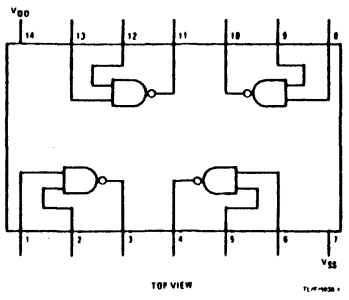
MM54C20/MM74C20



Order Number MM54C20J, MM74C20J, MM54C20N or MM74C20N
See NS Package J14A or N14A

CD4012

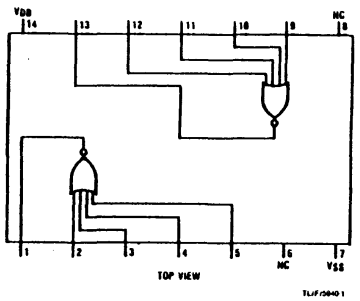




Order Number CD4001MJ, CD4001CJ, CD4011MJ or CD4011CJ
See NS Package J14A

Order Number CD4001MN, CD4001CN, CD4011MN or CD4011CN
See NS Package N14A

CD4002



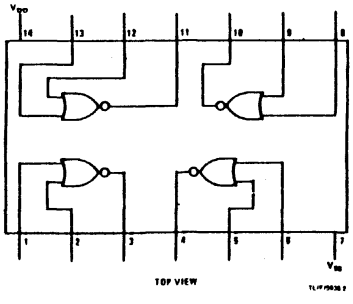
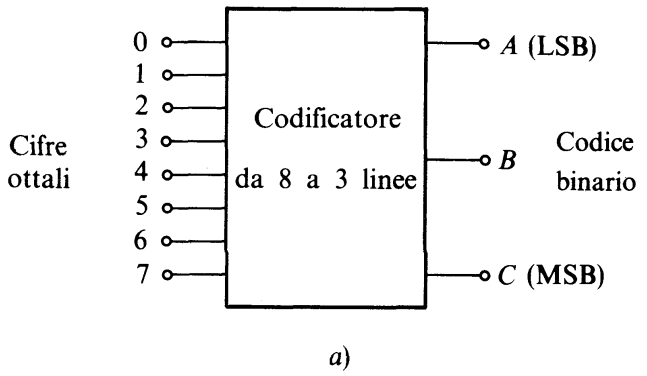


Fig. 1.2 – *Porte logiche CMOS*



a)

b)

Ingressi attivi	*Uscite Codice binario*		
	C	*B*	*A*
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

Fig. 1.3 – *a) Codificatore da 8 a 3 linee (ottale-binario). b) Tabella funzionale*

Gli ingressi sono spesso associati alle cifre del sistema decimale o di altri sistemi di numerazione, oppure ai caratteri alfabetici o ancora ad altri simboli speciali; le linee di uscita forniscono i bit della parola binaria corrispondente a ciascuna cifra, carattere o simbolo.

In Fig. 1.3*a* è proposto ad esempio un codificatore da 8 a 3 linee: le otto linee di ingresso possono essere associate alle cifre del sistema *ottale*, e quindi identificate con i numeri da 0 a 7, mentre le tre linee di uscita sono sufficienti a rappresentare i codici binari corrispondenti a ciascuna cifra; pertanto questo circuito può essere visto come un *codificatore ottale-binario.* Le relazioni fra ingressi attivi e uscite sono descritte sinteticamente dalla tabella di Fig. 1.3*b*.

Si nota inoltre che l'uscita più significativa *C* (MSB: *most significant bit*) è attiva (a livello logico 1) quando sono attivi gli ingressi 4, 5, 6 e 7; il secondo bit significativo *B* è attivo con gli ingressi 2, 3, 6 e 7, mentre quello meno significativo *A* (LSB: *least significant bit*) è attivo per tutti gli ingressi associati a cifre dispari. Il codificatore ottale-binario può quindi essere realizzato con una rete combinatoria esprimibile mediante le relazioni

$$\begin{cases} C = 4+5+6+7 \\ B = 2+3+6+7 \\ A = 1+3+5+7 \end{cases} . \qquad (1.1)$$

L'implementazione delle equazioni (1.1) conduce al circuito di Fig. 1.4, in cui gli ingressi delle porte, tenuti normalmente a livello basso tramite la resistenza (si considerano qui trascurabili le correnti di ingresso delle porte), passano al livello alto (V_{CC}) allorché vengono premuti i pulsanti.

Si deve inoltre rilevare che non è presente il pulsante relativo all'ingresso 0; ciò dipende dal fatto che lo stato delle uscite in condizione di riposo, ossia quando non viene premuto nessun pulsante, coincide proprio con il codice binario 000 corrispondente all'ingresso 0.

Un ultimo problema da esaminare è quello relativo alla situazione in cui due o più pulsanti vengano premuti contemporaneamente. In tale circostanza il codificatore di Fig. 1.4 non funziona più correttamente. Si può superare questo inconveniente assegnando a ciascuno degli ingressi un *livello di priorità.* Riferendosi ancora al codificatore ottale-binario, si può ad esempio assegnare priorità crescente dalla cifra 0 alla cifra 7; in tal modo, se vengono premuti contemporaneamente i tasti 3 e 6, il codice dì uscita sarà 110, ossia quello corrispondente alla cifra 6, con priorità più elevata. Ovviamente la rete logica diviene molto più complessa.

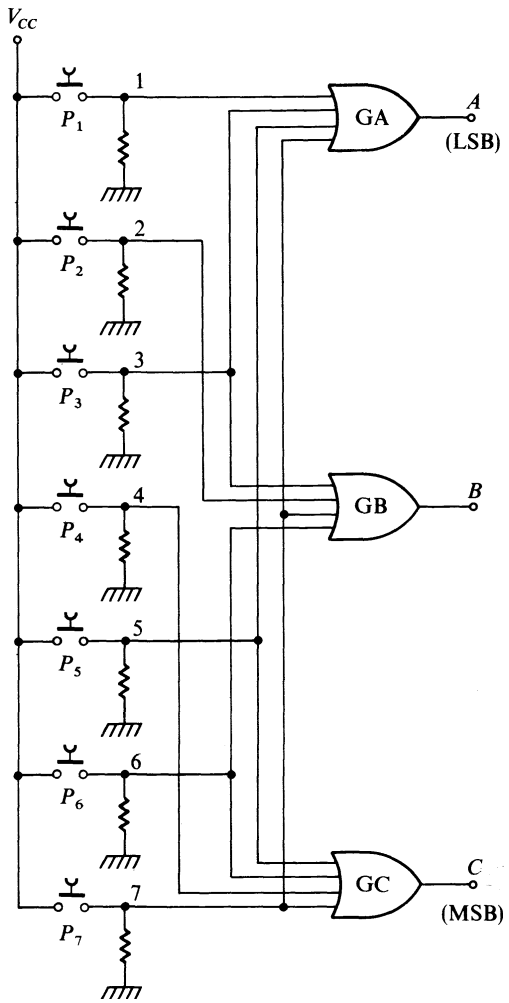


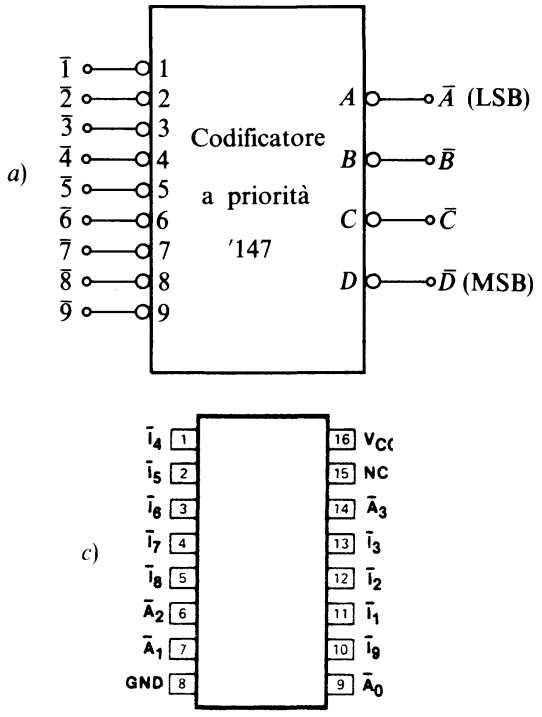
Fig. 1.4 – *Codificatore ottale-binario realizzato tramite porte logiche*

Quando si rende necessario l'uso di un codificatore, non è tuttavia necessario (almeno per la maggior parte dei casi) progettare una rete logica combinatoria come quella di Fig. 1.4. In

commercio esistono infatti svariati tipi di codificatori integrati sia di tipo TTL che CMOS. Vediamone alcuni.

1.2.1 Codificatori TTL

Fra i codificatori TTL si possono ricordare i tipi 74147 e il 74148. Il 74147, di cui sono riportati in Fig. 1.5 il simbolo logico, la tavola di verità e il pin-out (piedinatura), è un codificatore a priorità *decimale-BCD*.(♥) Esso presenta solo 9 ingressi in quanto, di fatto, il codice d'uscita corrispondente all'ingresso decimale 0 coincide con lo stato di riposo (nessuna linea di ingresso attiva) del codificatore. Osservando il simbolo (vedi i cerchietti di negazione presenti sugli ingressi e sulle uscite) si desume che sia gli ingressi che le uscite sono attivi bassi. Ciò è evidente anche considerando la tavola di verità: la prima riga rappresenta lo stato di riposo, in cui tutti gli ingressi sono alti e il codice di uscita è HHHH (in logica positiva $DCBA$ = 0000; ciò equivale a dire che le variabili $DCBA$ sono a livello logico 0, convenzionalmente lo stato inattivo). Si nota poi che l'attivazione della linea 9, quella a priorità maggiore, prevale sull'attivazione di qualsiasi altro ingresso; infatti, qualunque sia il livello, alto o basso (X), presente sugli ingressi da 1 a 8, lo stato delle uscite $\overline{D}\overline{C}\overline{B}\overline{A}$ è LHHL ($DCBA$ = 1001, che è proprio il codice BCD del numero 9).



Ingressi									*Uscite*			
$\overline{1}$	$\overline{2}$	$\overline{3}$	$\overline{4}$	$\overline{5}$	$\overline{6}$	$\overline{7}$	$\overline{8}$	$\overline{9}$	$\overline{D}$	$\overline{C}$	$\overline{B}$	$\overline{A}$
H	H	H	H	H	H	H	H	H	H	H	H	H
X	X	X	X	X	X	X	X	L	L	H	H	L
X	X	X	X	X	X	X	L	H	L	H	H	H
X	X	X	X	X	X	L	H	H	H	L	L	L
X	X	X	X	X	L	H	H	H	H	L	L	H
X	X	X	X	L	H	H	H	H	H	L	H	L
X	X	X	L	H	H	H	H	H	H	L	H	H
X	X	L	H	H	H	H	H	H	H	H	L	L
X	L	H	H	H	H	H	H	H	H	H	L	H
L	H	H	H	H	H	H	H	H	H	H	H	L

b)

Fig. 1.5 – *a) Simbolo logico del codificatore decimale-BCD TTL 74147, con relativa b) tabella della verità e c) pin-out*

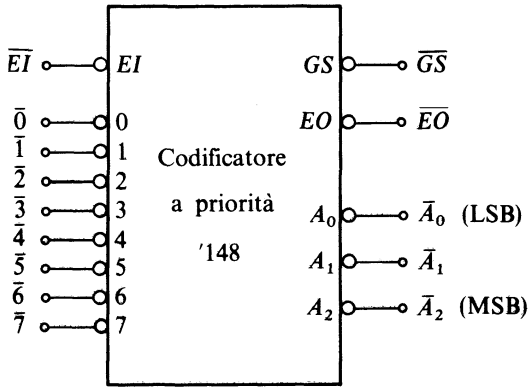
(♥) Si noti che i costruttori spesso preferiscono indicare i bit "0" e "1" con "L" (*low*, ossia *stato basso*) e "H" (*high*, ossia *stato alto*).

L'integrato 74148 è invece un codificatore a priorità da 8 a 3 linee, ossia un codificatore *ottale-binario*. In Fig. 1.6 sono illustrati il simbolo logico, la relativa tavola di verità e il pin-out. Anche in questo caso ingressi e uscite sono attivi bassi e l'ingresso a priorità più alta è il 7. Vi sono inoltre tre linee di controllo:

- EI (Enable Input). Se è al livello basso abilita il codificatore al funzionamento, altrimenti disabilita le linee di entrata dati ponendo tutte le uscite nello stato logico alto.
- EO (Enable Output). Si porta al livello logico basso per indicare che il chip è abilitato (EI = 0) ma non è stato attivato alcun ingresso.
- GS (Group Signal). Si porta al livello logico basso per indicare che il chip è stato abilitato (EI = 0) e contemporaneamente è stato attivato almeno un ingresso.

La presenza delle linee di controllo *EI*, *EO* e *GS* è prevista per rendere agevole il collegamento di più encoder in cascata in modo da poter codificare un numero superiore di linee di ingresso.

a)



Ingressi									*Uscite*				
$\overline{EI}$	$\overline{0}$	$\overline{1}$	$\overline{2}$	$\overline{3}$	$\overline{4}$	$\overline{5}$	$\overline{6}$	$\overline{7}$	$\overline{A}_2$	$\overline{A}_1$	$\overline{A}_0$	$\overline{GS}$	$\overline{EO}$
H	X	X	X	X	X	X	X	X	H	H	H	H	H
L	H	H	H	H	H	H	H	H	H	H	H	H	L
L	X	X	X	X	X	X	X	L	L	L	L	L	H
L	X	X	X	X	X	X	L	H	L	L	H	L	H
L	X	X	X	X	X	L	H	H	L	H	L	L	H
L	X	X	X	X	L	H	H	H	L	H	H	L	H
L	X	X	X	L	H	H	H	H	H	L	L	L	H
L	X	X	L	H	H	H	H	H	H	L	H	L	H
L	X	L	H	H	H	H	H	H	H	H	L	L	H
L	L	H	H	H	H	H	H	H	H	H	H	L	H

b)

c)

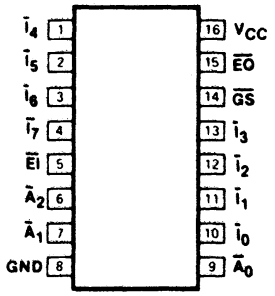


Fig. 1.6 – *a) Simbolo logico del codificatore decimale-BCD TTL 74148, con relativa b) tabella della verità e c) pin-out*

1.2.2 Codificatori CMOS

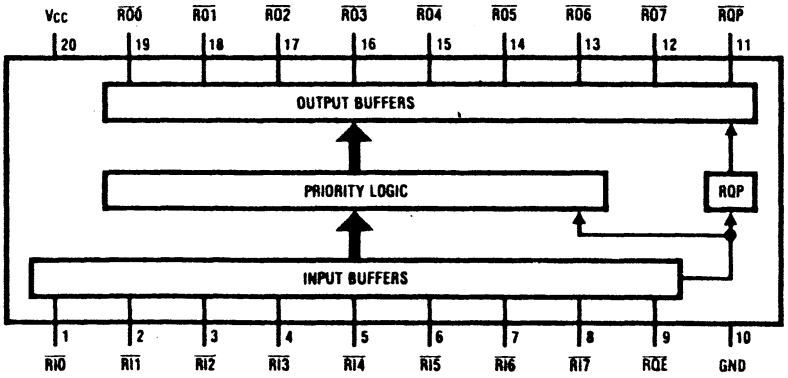
Fra i codificatori CMOS citiamo i tipi 74HC147 e il 74HC149. La sigla "HC" indica che il circuito è molto veloce (*High-speed* CMOS). Il 74HC147 è identico nella piedinatura e nel principio

di funzionamento al 74147 della TTL prima analizzato. Presenta un tempo di propagazione, ossia il tempo necessario per effettuare una commutazione da un livello di tensione all'altro(♣), t_p = 30 ns, con una potenza dissipata (a riposo) P_d = 40 μW, a differenza del suo omologo TTL che presenta t_p = 10 ns e P_d = 225 mW (ben quattro ordini di grandezza maggiore!).

Il tipo HC149 è un codificatore con priorità a 8 linee di ingresso *RI*7, *RI*6, ..., *RI*0 e 8 linee di uscita *RO*7, ..., *RO*0. È fornito di un ingresso di abilitazione $\overline{RQE}$ e di una linea di uscita supplementare $\overline{RQP}$ che si porta al livello basso se il chip é abilitato ($\overline{RQE}$ = 0) ed è attivata almeno una linea di ingresso.

Il dispositivo opera nel seguente modo: se si pone $\overline{RQE}$ = 0 si abilita il chip; attivando una o più linee di ingresso (livello basso) si porta al livello basso tra le uscite corrispondenti solo quella a più alta priorità.

In Fig. 1.7 si mostrano la piedinatura e la tabella della verità di tale integrato. Il chip in esame è disponibile anche nella versione 74HCT. La sigla "HCT" indica che il circuito è ad alta velocità ed è TTL compatibile (*High-speed CMOS TTL-compatible*).



a)

Inputs									Outputs								
0	1	2	3	4	5	6	7	$\overline{RQE}$	0	1	2	3	4	5	6	7	$\overline{RQP}$
X	X	X	X	X	X	X	X	H	H	H	H	H	H	H	H	H	H
H	H	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H
X	X	X	X	X	X	X	L	L	H	H	H	H	H	H	H	L	L
X	X	X	X	X	X	L	H	L	H	H	H	H	H	H	L	H	L
X	X	X	X	X	L	H	H	L	H	H	H	H	H	L	H	H	L
X	X	X	X	L	H	H	H	L	H	H	H	H	L	H	H	H	L
X	X	X	L	H	H	H	H	L	H	H	H	L	H	H	H	H	L
X	X	L	H	H	H	H	H	L	H	H	L	H	H	H	H	H	L
X	L	H	H	H	H	H	H	L	H	L	H	H	H	H	H	H	L
L	H	H	H	H	H	H	H	L	L	H	H	H	H	H	H	H	L

b)

Fig. 1.7 – *a) Piedinatura e schema logico del codificatore CMOS 74HC149 e b) relativa tabella della verità*

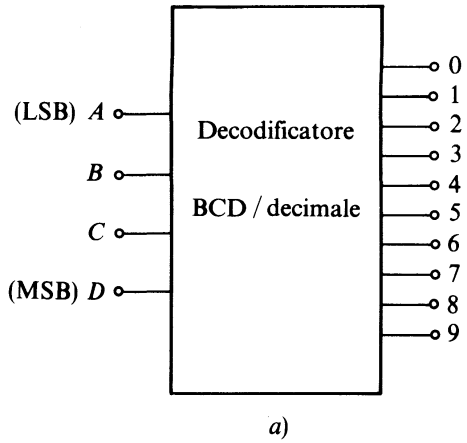
(♣) Una definizione più rigorosa del tempo di propagazione sarà data nel Cap. 12.

1.3 Decodificatori

La funzione di un decodificatore è di rivelare la presenza sui propri ingressi di particolari combinazioni di bit (*codici*) attivando, per ciascuna di esse, una determinata linea di uscita. In pratica, il decodificatore funziona in modo opposto al codificatore: in base al codice in ingresso si attiva la corrispondente uscita.

A titolo di esempio, in Fìg. 1.8 è illustrato il simbolo logico di un *decodificatore BCD-decimale* con la relativa tavola di verità. Gli stati logici presenti sulle quattro linee di ingresso rappresentano le cifre decimali in codice BCD; in corrispondenza di ciascuna combinazione di ingresso viene attivata una delle dieci linee di uscita. Dalla tavola di verità si desume facilmente che le varie uscite sono definite semplicemente da funzioni AND del tipo

$$\begin{array}{lllll} 0=\overline{D}\,\overline{C}\,\overline{B}\,\overline{A} & 1=\overline{D}\,\overline{C}\,\overline{B}A & 2=\overline{D}\,\overline{C}B\,\overline{A} & 3=\overline{D}\,\overline{C}BA & 4=\overline{D}C\,\overline{B}\,\overline{A} \\ 5=\overline{D}C\,\overline{B}A & 6=\overline{D}CB\,\overline{A} & 7=\overline{D}CBA & 8=D\,\overline{C}\,\overline{B}\,\overline{A} & 9=D\,\overline{C}\,\overline{B}A \end{array} \quad . \tag{1.2}$$



a)

b)

Ingressi BDC				*Uscite decimali attive*
D	*C*	*B*	*A*	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9

Fig. 1.8 – *a) Decodificatore BCD-decimale e b) relativa tabella della verità*

II procedimento descritto può essere utilizzato per progettare decodificatori di qualsiasi tipo: *binario-ottale* (da 3 linee di ingresso a 8 linee di uscita), *binario-esadecimale* (da 4 a 16 linee), *codice Gray-decimale,* ecc.

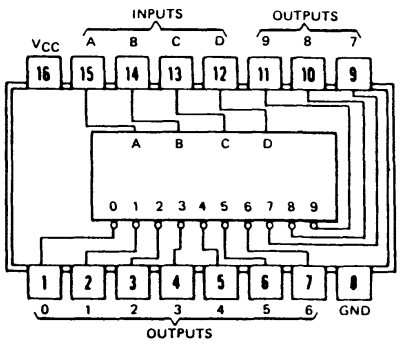
In commercio è disponibile una vasta gamma di decodificatori integrati realizzati nelle varie tecnologie con strutture e caratteristiche diverse.

1.3.1 Decodificatori TTL

Della famiglia TTL analizziamo dapprima il decodificatore 7442. Esso è un decodificatore a 4 entrate indicate con *DCBA* e 10 uscite 0, 1, 2, 3, 4, 5, 6, 7, 8 e 9. Il codice di entrata è BCD e le

uscite sono in logica negativa; inoltre per le combinazioni di entrata comprese tra 10 e 15 le uscite sono tutte al livello alto. In Fig. 1.9 si mostra la piedinatura e la tavola della verità di tale decodificatore.

Altri interessanti decodificatori sono il 74184 e il 74185: il primo converte il codice BCD in binario naturale mentre il secondo esegue la conversione opposta. Entrambi gli integrati, compatibili nella piedinatura, presentano una linea di abilitazione indicata con *G* ed operante in logica negativa.



a)

NO.	'42A, 'L42, 'LS42 BCD INPUT				ALL TYPES DECIMAL OUTPUT									
	D	C	B	A	0	1	2	3	4	5	6	7	8	9
0	L	L	L	L	L	H	H	H	H	H	H	H	H	H
1	L	L	L	H	H	L	H	H	H	H	H	H	H	H
2	L	L	H	L	H	H	L	H	H	H	H	H	H	H
3	L	L	H	H	H	H	H	L	H	H	H	H	H	H
4	L	H	L	L	H	H	H	H	L	H	H	H	H	H
5	L	H	L	H	H	H	H	H	H	L	H	H	H	H
6	L	H	H	L	H	H	H	H	H	H	L	H	H	H
7	L	H	H	H	H	H	H	H	H	H	H	L	H	H
8	H	L	L	L	H	H	H	H	H	H	H	H	L	H
9	H	L	L	H	H	H	H	H	H	H	H	H	H	L
INVALID	H	L	H	L	H	H	H	H	H	H	H	H	H	H
	H	L	H	H	H	H	H	H	H	H	H	H	H	H
	H	H	L	L	H	H	H	H	H	H	H	H	H	H
	H	H	L	H	H	H	H	H	H	H	H	H	H	H
	H	H	H	L	H	H	H	H	H	H	H	H	H	H
	H	H	H	H	H	H	H	H	H	H	H	H	H	H

b)

Fig. 1.9 – *a) Piedinatura del decodificatore TTL 7442 e b) relativa tabella della verità*

Analizziamo infine i tipi 74138, 74139, e 74154. Questi decodificatori sono noti anche con il nome di *decoder/demultiplexer* poiché oltre alla funzione di decodifica svolgono anche quella di *distributore* (*demultiplexer*) che sarà analizzata nel par. 1.5. Il 74139 è un doppio decodificatore contenente 2 decoder indipendenti ciascuno a 2 ingressi e 4 uscite forniti di una linea di ENABLE indicata con *CS* (chip select). Il 74138 è un decoder a 3 ingressi e 8 uscite con tre linee di abilitazione indipendenti indicate con *CS*1, *CS*2 e *CS*3. Il 74154 presenta 4 ingressi, 16 uscite e 2 linee di ENABLE indicate con *CS*1 e *CS*2. Sia le linee di ENABLE che quelle di uscita sono in logica negativa. In Fig. 1.10 si mostra la piedinatura e la logica del 74154. Si noti che quest'ultimo possiede 24 piedini.

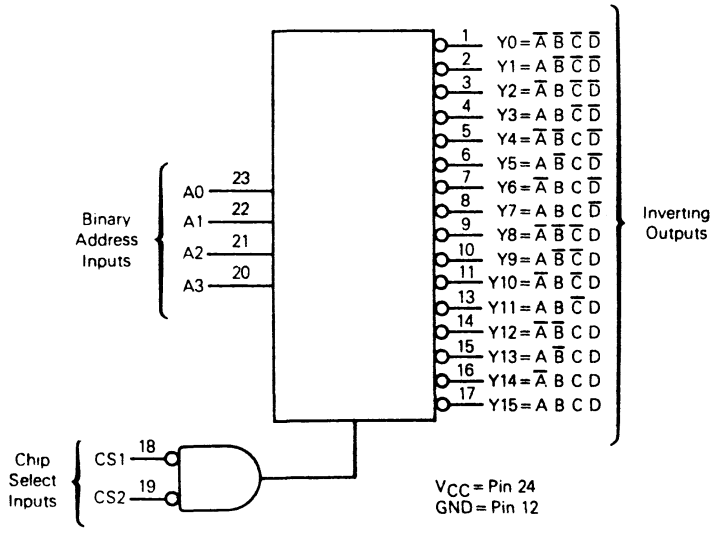


Fig. 1.10 – *a) Piedinatura e logica del decodificatore/demultiplexer TTL 74154*

1.3.2 Decodificatori CMOS

Nella famiglia CMOS esistono numerosi decodificatori che svolgono le stesse funzioni di quelli corrispondenti della TTL e spesso compatibili nella piedinatura (serie 74 C). I due decodificatori mostrati precedentemente in Fig. 1.9 e Fig. 1.10 esistono anche della serie 74 HC.

Per quanto concerne la serie 4000 si vuole ricordare solo il CD4028 che ha 4 ingressi nel codice BCD e possiede 10 uscite indicate con 0, 1, 2, 3, 4, 5, 6, 7, 8 e 9. Dal punto di vista logico l'unica differenza rispetto al 7442 TTL è che le uscite sono in logica positiva, anziché negativa.

1.3.3 Decoder-driver

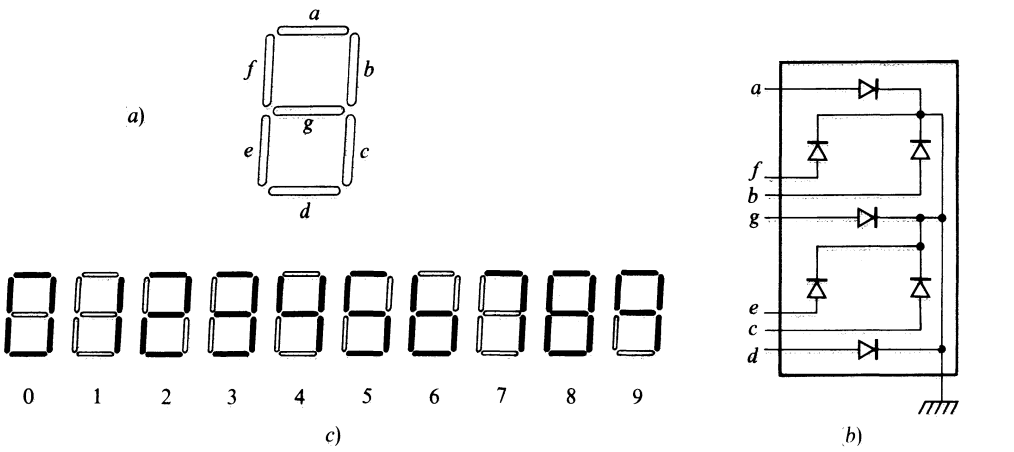
I *decoder-driver* sono una particolare categoria di decodificatori utilizzati quasi esclusivamente per pilotare *display a sette segmenti* o *LCD* (*Liquid-Crystal Displays*). I display a sette segmenti sono costituiti da sette *LED* (*Light-Emitting Diodes*) a forma di segmenti, contrassegnati dalle lettere *a*, *b*, ..., *f*, *g* (cfr. Fig. 1.11*a*), che possono essere resi luminosi individualmente o secondo combinazioni tali da rappresentare le cifre decimali o esadecimali.

I *diodi LED* sono dispositivi a semiconduttore provvisti di due elettrodi: l'anodo e il catodo. Essi sono in grado di emettere luce se "*polarizzati direttamente*", ossia se all'anodo viene applicata una tensione maggiore rispetto a quella del catodo di qualche volt. Questo concetto risulterà più chiaro quando verrà affrontato lo studio dei diodi e dei diodi LED nei capitoli 6 e 9.

In Fig. 1.11*b* è illustrata la struttura interna di principio di un display a LED *a catodo comune*. I catodi, connessi insieme, vengono collegati a massa; applicando ad uno dei terminali di ingresso una tensione tale da polarizzare direttamente il diodo, il corrispondente segmento risulta acceso. Tralasciando per ora le considerazioni relative ai livelli di tensione e corrente necessari, si conclude

che, attivando opportunamente gli ingressi, si possono rappresentare con chiarezza le cifre decimali come si vede in Fig. 1.11*c*.

Supponendo di disporre, come spesso accade, di cifre decimali espresse in codice BCD e di volerle visualizzare con il display a sette segmenti, occorrerà convertire il codice secondo la tabella di Fig. 1.11*d*. Così, ad esempio, al codice d'ingresso 0001 corrisponderà il codice di uscita 0110000, ossia saranno accesi i segmenti *b* e *c*, e verrà visualizzata la cifra decimale 1. Un tal tipo di decoder-driver prende anche il nome di *convertitore BCD-sette segmenti*.



d)

Cifra decimale	Ingressi BCD				Uscite 7-segmenti						
	D	*C*	*B*	*A*	***a***	***b***	***c***	***d***	***e***	***f***	***g***
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	0	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	0	0	1	1

Fig. 1.11 – *a) Display a sette segmenti. b) Struttura interna. c) Visualizzazione delle cifre decimali. d) Tabella di verità del decoder-driver BCD-sette segmenti*

Il progetto della rete logica del convertitore di codice è piuttosto laborioso, ma in commercio sono comunque disponibili dispositivi integrati che risolvono completamente questo problema. I decoder-driver (o più semplicemente *decoder*) integrati generalmente forniscono i livelli elettrici necessari per pilotare i display, richiedendo talvolta l'aggiunta di resistori di limitazione esterni.

I decoder presentano alcuni terminali supplementari, che rendono più agevole la realizzazione di visualizzatori a più cifre e il controllo degli stessi. In molti casi è presente un ingresso indipendente

(*LT*: *lamp test*) che consente di provare se i segmenti sono tutti efficienti; inoltre è in genere disponibile un ingresso che permette di spegnere il display indipendentemente dal codice BCD presente in ingresso.

Esempi di decoder TTL per display LED a 7 segmenti sono i tipi 7446, 7447, 7448 e 7449. Presentano 4 ingressi in logica positiva *DCBA* e 7 uscite indicate con *a*, *b*, *c*, *d*, *e*, *f*, *g*, che devono essere collegate ai corrispondenti segmenti del display. Impostando una configurazione di ingresso si attivano simultaneamente le uscite corrispondenti alla cifra da visualizzare. Il 7446 e il 7447 hanno le uscite in logica negativa per cui possono pilotare display a 7 segmenti ad anodo comune. Le uscite possono sopportare una corrente di 40 mA con tensione di 30 V per il 7446 e 15 V per il 7447.

Il 7448 e il 7449 presentano le uscite in logica positiva ed entrambi possono pilotare display a catodo comune. Tutti i modelli citati, ad esclusione del 7449, presentano:

– la linea *LT* di lamp test, che se posta al livello basso consente l'accensione simultanea di tutti i segmenti;
– la linea *BI/RBO* (*blanking input/ripple blanking output*) che è usata sia come ingresso che come uscita nel collegamento in cascata di più visualizzatori; se si applica su tale linea un livello basso, indipendentemente da tutte le altre condizioni, le uscite risultano disabilitate per cui il display è spento;
– la linea *RBI* (*ripple blanking input*) se posta al livello basso disabilita le uscite (display spento) per la combinazione di entrata corrispondenti al numero 0. Ciò consente di tener "spenti" gli zeri non significativi nel caso di collegamento in cascata di più cifre da visualizzare.

In Fig. 1.12 si mostra il tipico collegamento tra decoder/driver e display a 7 segmenti.

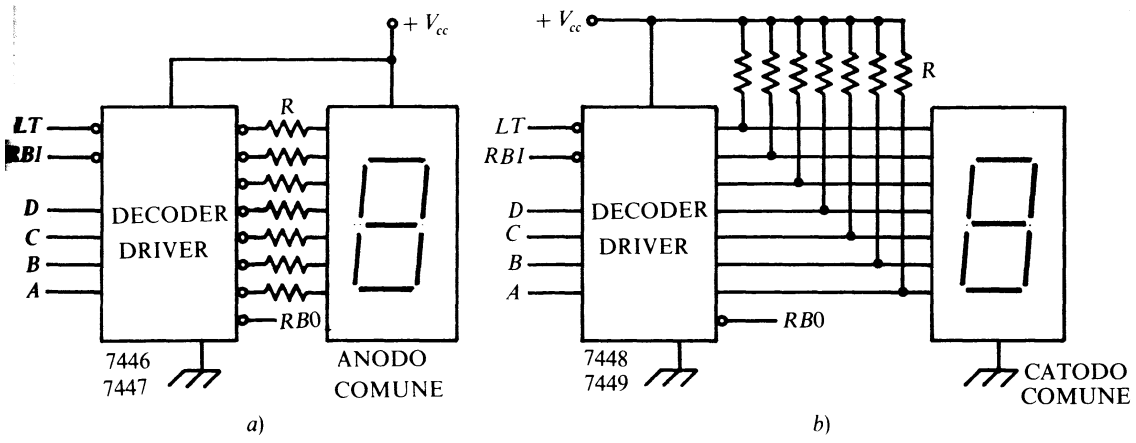


Fig. 1.12 – *Collegamento di un display: a) ad anodo comune pilotato da un decoder in logica negativa; b) a catodo comune pilotato da un decoder a logica positiva*

Le resistenze *R* definiscono la luminosità dei segmenti del display, che infatti dipende dalla corrente che circola nei LED; normalmente il loro valore è compreso tra 150 e 470 Ω. Si tenga presente che per l'accensione dei LED la corrente minima è dell'ordine di 5-8 mA, mentre è bene che questa non superi 40-50 mA per evitare la distruzione del dispositivo.

Il pin-out e la tabella della verità dei quattro tipi sopra riportati sono illustrati in Fig. 1.13. Per i modelli con uscita in logica negativa (7446 e 7447), ON indica lo stato logico basso e OFF quello alto; viceversa per il 7448 e il 7449, le cui uscite sono in logica positiva.

Un altro interessante decoder TTL è il 9368 corrispondente nella piedinatura e nella logica al 7448 rispetto al quale, però, presenta le seguenti differenze:

a) può erogare una corrente di 19 mA per cui è possibile il collegamento diretto al display senza necessità di inserire resistenze esterne (di protezione);

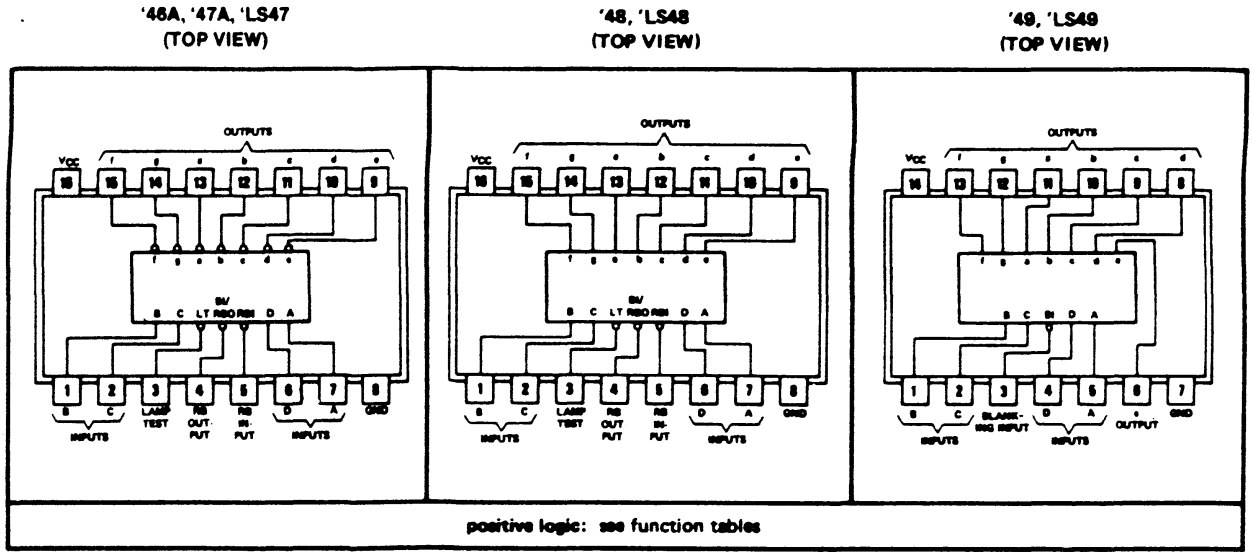
b) è in grado di visualizzare il codice esadecimale;

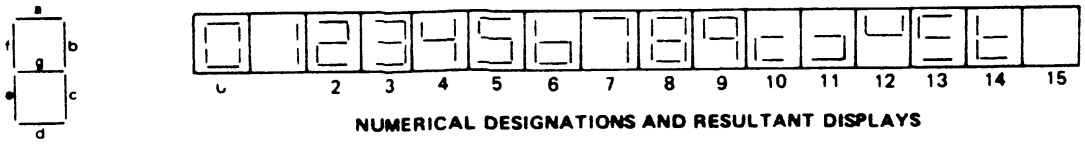
c) ha una memoria interna abilitata dalla linea *LE* (*Latch Enable*) che gli consente di tenere memorizzato lo stato logico di uscita, ponendo *LE* = 1, indipendentemente dal valore delle entrate.

Per quanto concerne le tecniche di pilotaggio dei display LCD giova brevemente ricordare che la tecnologia a cristalli liquidi ha assunto un grande sviluppo per il fatto che consente un consumo di potenza ridottissimo (10-15 μW per display). Ciò perché il display a cristalli liquidi non genera nessuna forma di energia luminosa ma sfrutta una sorgente luminosa esterna posta anteriormente al visualizzatore (LCD *a riflessione* di luce) oppure posteriormente (LCD *a trasmissione* di luce).

1.4 Multiplexer (o selettore)

La funzione di un *multiplexer* (o *selettore*), il cui schema a blocchi è mostrato in Fig. 1.14 (indicato con MUX), è quella di trasferire all'uscita i dati digitali presenti ai suoi *n* ingressi, convogliandoli su un'unica linea; la selezione del singolo dato che deve essere trasferito avviene attraverso ingressi di controllo o *selezione* (*data select*; selezione dati).





NUMERICAL DESIGNATIONS AND RESULTANT DISPLAYS

SEGMENT IDENTIFICATION

DECIMAL OR FUNCTION	INPUTS						BI/RBO†	OUTPUTS							NOTE
	LT	RBI	D	C	B	A		a	b	c	d	e	f	g	
0	H	H	L	L	L	L	H	ON	ON	ON	ON	ON	ON	OFF	1
1	H	X	L	L	L	H	H	OFF	ON	ON	OFF	OFF	OFF	OFF	
2	H	X	L	L	H	L	H	ON	ON	OFF	ON	ON	OFF	ON	
3	H	X	L	L	H	H	H	ON	ON	ON	ON	OFF	OFF	ON	
4	H	X	L	H	L	L	H	OFF	ON	ON	OFF	OFF	ON	ON	
5	H	X	L	H	L	H	H	ON	OFF	ON	ON	OFF	ON	ON	
6	H	X	L	H	H	L	H	OFF	OFF	ON	ON	ON	ON	ON	
7	H	X	L	H	H	H	H	ON	ON	ON	OFF	OFF	OFF	OFF	
8	H	X	H	L	L	L	H	ON	ON	ON	ON	ON	ON	ON	
9	H	X	H	L	L	H	H	ON	ON	ON	OFF	OFF	ON	ON	
10	H	X	H	L	H	L	H	OFF	OFF	OFF	ON	ON	OFF	ON	
11	H	X	H	L	H	H	H	OFF	OFF	ON	ON	OFF	OFF	ON	
12	H	X	H	H	L	L	H	OFF	ON	OFF	OFF	OFF	ON	ON	
13	H	X	H	H	L	H	H	ON	OFF	OFF	ON	OFF	ON	ON	
14	H	X	H	H	H	L	H	OFF	OFF	OFF	ON	ON	ON	ON	
15	H	X	H	H	H	H	H	OFF	OFF	OFF	OFF	OFF	OFF	OFF	
BI	X	X	X	X	X	X	L	OFF	OFF	OFF	OFF	OFF	OFF	OFF	2
RBI	H	L	L	L	L	L	L	OFF	OFF	OFF	OFF	OFF	OFF	OFF	3
LT	L	X	X	X	X	X	H	ON	ON	ON	ON	ON	ON	ON	4

H = high level, L = low level, X = irrelevant

NOTES: 1. The blanking input (BI) must be open or held at a high logic level when output functions 0 through 15 are desired. The ripple-blanking input (RBI) must be open or high if blanking of a decimal zero is not desired.
2. When a low logic level is applied directly to the blanking input (BI), all segment outputs are off regardless of the level of any other input.
3. When ripple-blanking input (RBI) and inputs A, B, C, and D are at a low level with the lamp test input high, all segment outputs go off and the ripple-blanking output (RBO) goes to a low level (response condition).
4. When the blanking input/ripple blanking output (BI/RBO) is open or held high and a low is applied to the lamp-test input, all segment outputs are on.

†BI/RBO is wire-AND logic serving as blanking input (BI) and/or ripple-blanking output (RBO).

Fig. 1.13 – *Piedinatura e tabella della verità dei decoder TTL 7446, 7447, 7448 e 7449*

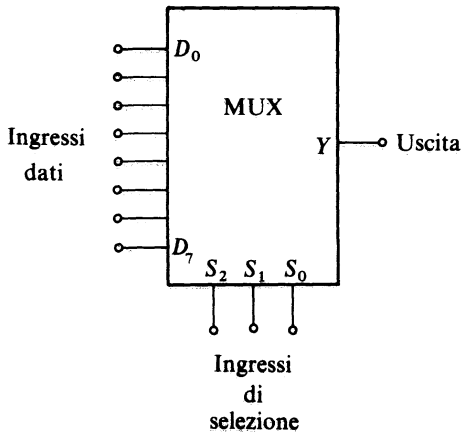
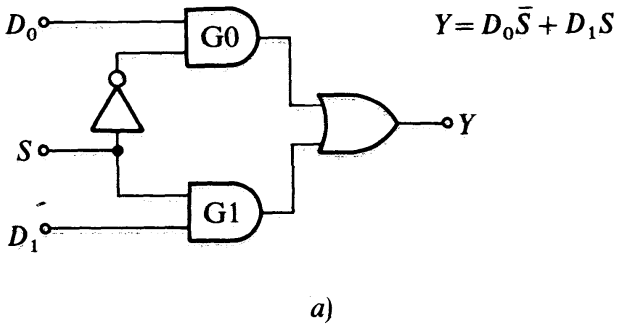


Fig. 1.14 – *Schema a blocchi di un multiplexer a otto ingressi*

In Fig. 1.15 è illustrato il circuito logico del più semplice multiplexer accanto alla sua tabella di verità. Quando l'ingresso di selezione è $S = 0$, risulta abilitata la porta AND superiore G0 e il dato digitale (0 o 1) presente sull'ingresso dati D_0 viene trasferito all'uscita. Quando viceversa $S = 1$, è abilitata la porta AND inferiore G1 e all'uscita viene trasferito il valore digitale presente su D_1. In altri termini, l'ingresso dati D_0 è selezionato da $S = 0$, mentre l'ingresso dati D_1 è selezionato da $S = 1$. La funzione logica che esprime l'uscita è

$$Y = D_0\overline{S} + D_1 S\,. \tag{1.3}$$



a)

b)

Ingressi			*Uscita*
S	D_1	D_0	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Fig. 1.15 – *a) Realizzazione circuitale di un multiplexer a due ingressi e b) relativa tabella di verità*

Utilizzando più linee di selezione si possono realizzare multiplexer con numerosi ingressi dati: con 2 linee di selezione si possono multiplare 4 linee dati, con 3 si arriva a 8 linee dati e così via. Pertanto con m linee di selezione si possono multiplare $n = 2^m$ linee di ingresso.

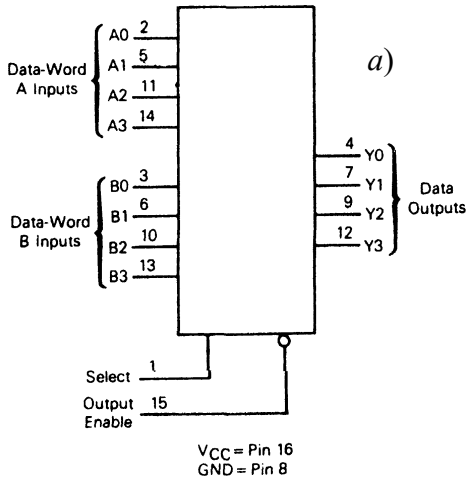
I multiplexer sono impiegati in numerosi campi applicativi che vanno dal controllo dei dispositivi di visualizzazione e delle memorie, ai sistemi di acquisizione dati, ai sistemi di trasmissione, alla generazione di funzioni logiche, ecc.

In commercio sono disponibili integrati, realizzati in tutte le tecnologie, che contengono multiplexer di vario tipo: con 2, 4, 8, 16 linee di ingresso dati, con uscite complementate e non, con stadio di uscita a più stati. In tutti questi dispositivi sono sempre presenti una o più linee di *strobe* o di abilitazione, normalmente attive basse, che consentono il controllo del dispositivo stesso.

1.4.1 Multiplexer TTL

Della famiglia TTL analizziamo i multiplexer:

- *74157* e *74158*. Entrambi contengono 4 multiplexer a 2 ingressi dati e una linea di selezione e sono compatibili tra loro nella piedinatura (Fig. 1.16*a*). Come appare nella tabella della verità di Fig. 1.16*b*, il primo presenta l'uscita coincidente con il valore del dato selezionato, nel secondo invece l'uscita è invertita Il 74158 è disponibile solo della serie S con tempo di propagazione t_p = 4 ns e della serie LS con t_p = 7 ns, mentre il 74157 è disponibile anche nella serie standard con t_p = 9 ns. La potenza dissipata varia tra 50 mW e 250 mW a seconda della serie di appartenenza. Entrambi i dispositivi hanno un ingresso di *output enable*, *OE*, in logica negativa che ha il compito di abilitare il chip.
- *74257* e *74258*. Sono analoghi ai modelli 157 e 158, compatibili anche nella piedinatura; la differenza è che l'ingresso *OE* denominato *output control* ha la funzione cosiddetta di *three-state*, vale a dire se è posto al livello alto le uscite si portano in uno stato di *alta impedenza* mentre al livello basso il chip è abilitato al funzionamento. Il three-state è pertanto un "terzo stato" che può assumere il circuito, né alto, né basso ma di inibizione (alta impedenza) che permette di escludere l'uscita del circuito stesso. Esso verrà esaminato in dettaglio nel Cap. 11.



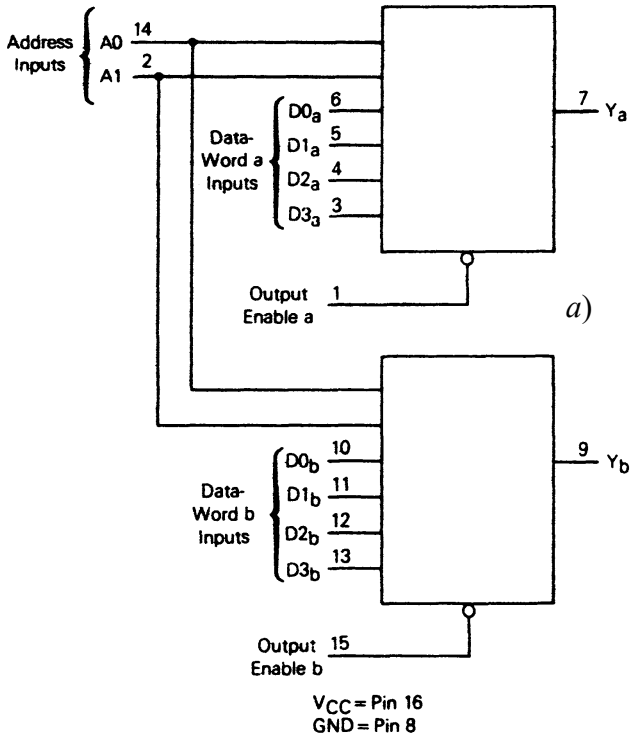
a)

OE	S	A	B	Y (157)	Y (158)
1	X	X	X	0	1
0	0	0	X	0	1
0	0	1	X	1	0
0	1	X	0	0	1
0	1	X	1	1	0

b)

Fig. 1.16 – *a) Schema funzionale e b) tabella di verità dei multiplexer TTL 74157 e 74158*

- *74153*. È un doppio multiplexer a 4 ingressi dati. Ogni multiplexer presenta l'ingresso di output enable *OE* in logica negativa e un'uscita *Y* di valore coincidente a quello del dato di ingresso D_0, D_1, D_2, D_3 a seconda degli ingressi di selezione A_1 e A_0, comuni ad entrambi i multiplexer. In Fig. 1.17 si mostra lo schema funzionale insieme alla tabella della verità.
- *74352*. È compatibile nella piedinatura con il 153, ha le uscite negate rispetto al 153 ma è disponibile solo nella serie S e LS con tempo di propagazione di 6 ns e 12 ns rispettivamente. Il modello 253 è la versione three-state, mentre il 353 è il three-state con uscite invertite.
- *74151*. È un multiplexer con 8 ingressi dati da D_0 a D_7, 3 bit di selezione A_2, A_1, A_0 e una linea di output enable *OE* in logica negativa. Il dispositivo presenta, inoltre, 2 uscite *Y* e $\overline{Y}$ tra loro complementari. Il modello 251 è la versione three-state.



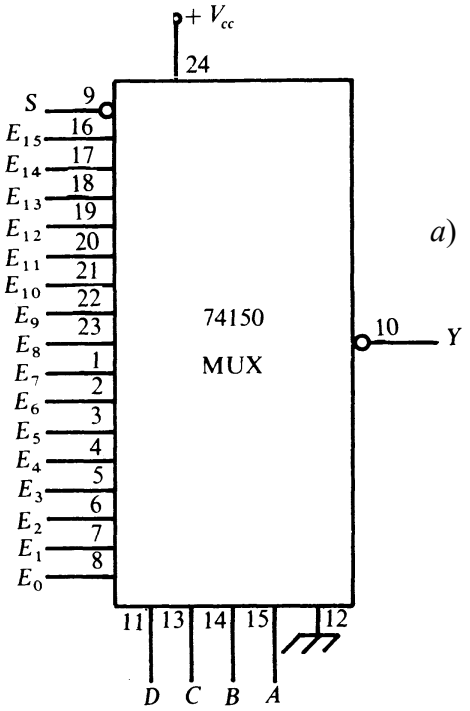
Inputs			Output
A1	A0	Output Enable	Y
X	X	H	L
L	L	L	D0
L	H	L	D1
H	L	L	D2
H	H	L	D3

D0, D1, D2, and D3 = the level of the respective Data Input

b)

Fig. 1.17 – *a) Schema funzionale e b) tabella di verità del multiplexer TTL 74153*

- *74150*. È un multiplexer a 16 ingressi. In Fig. 1.18 si mostra lo schema funzionale e la tabella della verità.



S	D	C	B	A	Y
1	X	X	X	X	1
0	0	0	0	0	$\overline{E}_0$
0	0	0	0	1	$\overline{E}_1$
0	0	0	1	0	$\overline{E}_2$
0	0	0	1	1	$\overline{E}_3$
0	0	1	0	0	$\overline{E}_4$
0	0	1	0	1	$\overline{E}_5$
0	0	1	1	0	$\overline{E}_6$
0	0	1	1	1	$\overline{E}_7$
0	1	0	0	0	$\overline{E}_8$
0	1	0	0	1	$\overline{E}_9$
0	1	0	1	0	$\overline{E}_{10}$
0	1	0	1	1	$\overline{E}_{11}$
0	1	1	0	0	$\overline{E}_{12}$
0	1	1	0	1	$\overline{E}_{13}$
0	1	1	1	0	$\overline{E}_{14}$
0	1	1	1	1	$\overline{E}_{15}$

b)

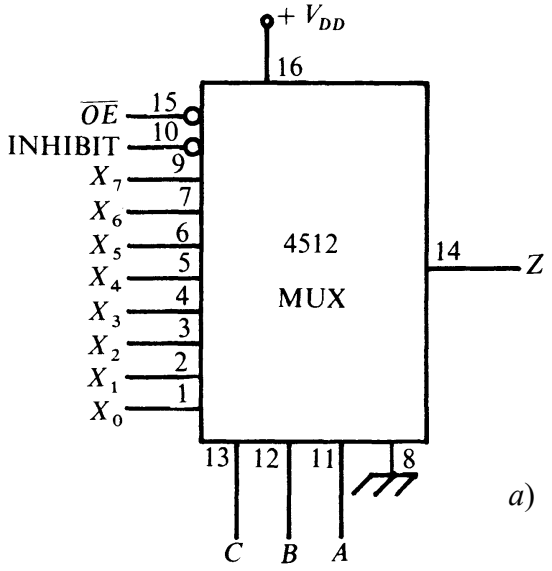
Fig. 1.18 – *a) Schema funzionale e b) tabella di verità del multiplexer TTL 74150*

1.4.2 Multiplexer CMOS

Dei modelli TTL analizzati sinora, esistono anche i corrispondenti CMOS della serie 74C, equivalenti nella piedinatura ai corrispondenti TTL.

Della serie 4500 analizziamo i seguenti multiplexer:

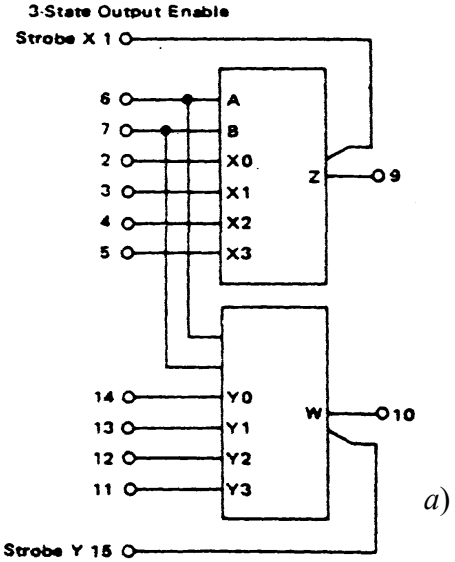
- *4512*. È un multiplexer a 8 ingressi dati e 3 selezioni con ingresso di abilitazione output enable *OE* funzionante in logica negativa che comanda il three-state di uscita. Presenta, inoltre, un ulteriore ingresso di INHIBIT che, se posto al livello alto, forza l'uscita nello stato basso. In Fig. 1.19 si mostra lo schema funzionale e la tabella della verità di tale dispositivo.
- *4529*. È un multiplexer che può operare sia come doppio multiplexer a 4 ingressi dati che come singolo a 8 ingressi. Il componente, mostrato in Fig. 1.20*a*, è adatto sia per segnali analogici che digitali. Il funzionamento da doppio multiplexer a 4 ingressi o da singolo a 8 ingressi dati si ottiene selezionando opportunamente le linee STX e STY come mostrato nella tabella della verità di Fig. 1.20*b*.



INHIBIT	$\overline{OE}$	C	B	A	Z
1	0	X	X	X	0
0	1	X	X	X	$HI-Z$
0	0	0	0	0	X_0
0	0	0	0	1	X_1
0	0	0	1	0	X_2
0	0	0	1	1	X_3
0	0	1	0	0	X_4
0	0	1	0	1	X_5
0	0	1	1	0	X_6
0	0	1	1	1	X_7

b)

Fig. 1.19 – *a) Schema funzionale e b) tabella di verità del multiplexer CMOS 4512*



ST_X	ST_Y	B	A	Z	W	
1	1	0	0	X0	Y0	Dual 4-Channel Mode 2 Outputs
1	1	0	1	X1	Y1	
1	1	1	0	X2	YZ	
1	1	1	1	X3	Y3	
1	0	0	0	X0		Single 8-Channel Mode 1 Output (Z and W tied together)
1	0	0	1	X1		
1	0	1	0	X2		
1	0	1	1	X3		
0	1	0	0	Y0		
0	1	0	1	Y1		
0	1	1	0	Y2		
0	1	1	1	Y3		
0	0	X	X	High Impedance		

x = Don't Care

b)

Fig. 1.20 – *a) Schema funzionale e b) tabella di verità del multiplexer CMOS 4529*

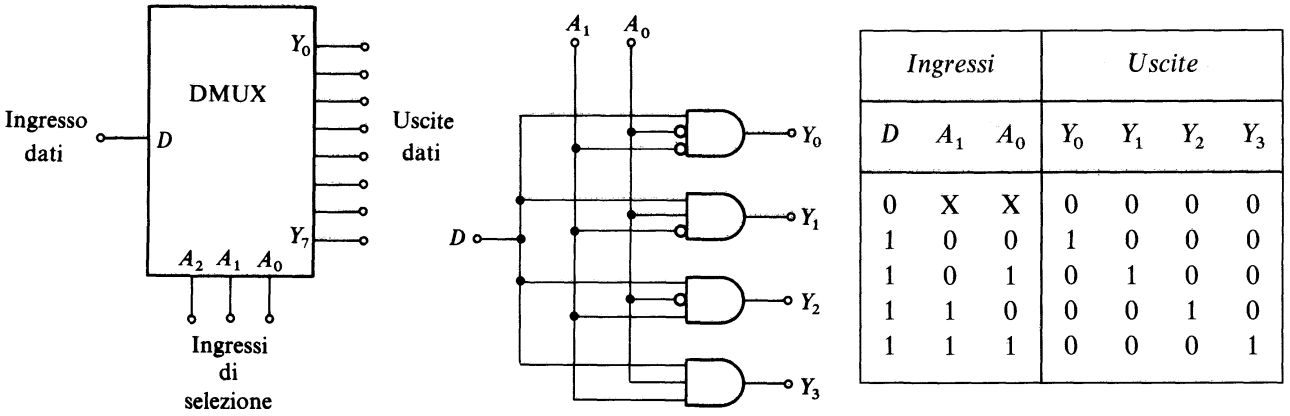
1.5 Demultiplexer (o distributore)

Un *demultiplexer* (o *distributore*) è un dispositivo il cui funzionamento è sostanzialmente invertito rispetto a quello del multiplexer. Esso è, infatti, un circuito logico che presenta essenzialmente una sola linea di ingresso dati, m linee di ingresso di selezione ed n di linee di uscita, dove solitamente $n = 2^m$. Lo schema a blocchi di Fig. 1.21*a* rappresenta appunto un demultiplexer (DMUX) da una ad otto linee dati che necessita di tre ingressi di selezione. La sua

funzione è quella di distribuire il dato digitale di ingresso ad una o all'altra delle linee di uscita, in relazione allo stato logico presente sugli ingressi di selezione. Il codice binario presente su questi ultimi viene cosi a rappresentare l'indirizzo della linea di uscita su cui viene trasferito il dato di ingresso.

I demultiplexer sono frequentemente utilizzati nei sistemi di distribuzione dati e nella trasmissione con tecniche digitali.

In Fig. 1.21*b* è illustrato lo schema logico di un demultiplexer da una a quattro linee accanto alla relativa tabella della verità (Fig. 1.21*c*). Per ciascuna combinazione dei *bit di selezione* o *indirizzo* A_1 A_0, una sola delle porte risulta abilitata e il dato D viene quindi trasferito all'uscita corrispondente. Si può notare che la struttura circuitale del demultiplexer è sostanzialmente quella di un decodificatore con l'unica differenza che ciascuna porta ha un ingresso in più: quello della linea di ingresso dati comune. In altre parole, il demultiplexer è un decodificatore in cui l'ingresso di *enable* del decodificatore coincide con quello dei dati del demultiplexer e le linee d'ingresso sono le linee di controllo del demultiplexer. Per questo motivo le case costruttrici classificano questi componenti come *decodificatori-demultiplexer*.

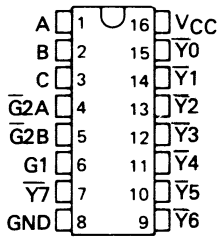


Ingressi			*Uscite*			
D	A_1	A_0	Y_0	Y_1	Y_2	Y_3
0	X	X	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

a) *b*) *c*)

Fig. 1.21 – *a) Schema a blocchi, b) schema logico e c) tabella di verità di un demultiplexer*

In Fig. 1.22 è illustrato il pin-out e la tabella della verità di un decodificatore-demultiplexer classico, il tipo 74138, (disponibile sia nella famiglia TTL che in quella CMOS) che accetta codici a 3 bit (C, B, A) e fornisce 8 linee di uscita ($\overline{Y_i}$)) attive basse. Il decoder presenta anche tre ingressi di abilitazione, uno attivo alto ($G1$) e due attivi bassi ($\overline{G2A}$ e $\overline{G2B}$); perché venga abilitato è quindi necessario che l'ingresso $G1$ sia alto e contemporaneamente $\overline{G2A}$ e $\overline{G2B}$ siano bassi.



a)

Ingressi					*Uscite*							
Abilitazione		*Selezione*										
G1	$\overline{G2}$	*C*	*B*	*A*	$\overline{Y0}$	$\overline{Y1}$	$\overline{Y2}$	$\overline{Y3}$	$\overline{Y4}$	$\overline{Y5}$	$\overline{Y6}$	$\overline{Y7}$
X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	H	L	H	H
H	L	H	H	L	H	H	H	H	H	H	L	H
H	L	H	H	H	H	H	H	H	H	H	H	L

$\overline{G2} = \overline{G2A} + \overline{G2B}$

b)

Fig. 1.22 – *a) Piedinatura e b) tabella di verità del decodificatore-demultiplexer 74138*

Per utilizzare questo dispositivo come demultiplexer basta adibire una delle linee di abilitazione, ad esempio $G1$, a linea di ingresso dati; gli altri due terminali di abilitazione mantengono invece la loro funzione. Dalla tabella di verità si desume che se $\overline{G2} = \overline{G2A} + \overline{G2B}$ è a livello basso, un livello logico alto sull'ingresso dati $G1$ abilita il dispositivo e pertanto l'uscita selezionata, corrispondente al codice presente negli ingressi C, B, A, va a livello basso. Viceversa se $G1$ è basso, il dispositivo rimane disabilitato e sull'uscita selezionata (come anche sulle altre) è presente un livello alto. In altri termini, il dato presente sull'ingresso dati $G1$ viene trasferito, in forma complementata, all'uscita selezionata.

1.6 Comparatori

La funzione fondamentale di un comparatore è quella di confrontare tra loro due numeri, o più genericamente due parole digitali, A e B, segnalando se sono uguali.

Se le due parole sono costituite da un solo bit, il circuito in grado di realizzare questa funzione è una semplice porta EXOR a due ingressi; essa, infatti, presenta in uscita uno 0 logico quando i bit di ingresso sono uguali. Si tratta pertanto di un comparatore di uguaglianza con uscita attiva bassa. Ovviamente utilizzando un circuito EXNOR, si ottiene un comparatore con uscita attiva alta, che segnala cioè con un 1 logico l'uguaglianza di due bit.

Per realizzare un comparatore per parole a più bit, è necessario confrontare i bit di uguale peso delle due parole; l'uscita dovrà segnalare l'uguaglianza solo se tutti i bit corrispondenti risultano

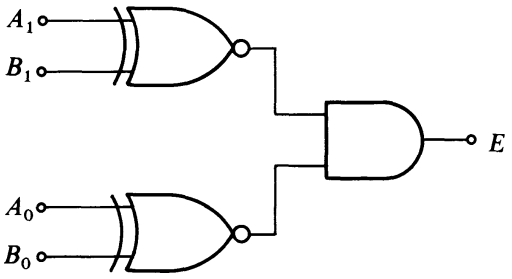
uguali. Si consideri ad esempio la tabella di verità mostrata in Fig. 1.23*a* relativa ad un comparatore che confronta due parole A e B a 2 bit, indicate rispettivamente come $A_1\,A_0$ e $B_1\,B_0$; l'uscita vale 1 solo quando tutti i bit corrispondenti, ossia di uguale peso, sono uguali. Com'era prevedibile, si ricava che l'uscita E è espressa dalla relazione

$$E = \left(\overline{A_1 \oplus B_1}\right)\left(\overline{A_0 \oplus B_0}\right). \tag{1.4}$$

cioè E vale 1 solo se $A_1 = B_1$ e contemporaneamente $A_0 = B_0$; pertanto il circuito risulta quello illustrato in Fig. 1.23*b*. Il procedimento illustrato può essere facilmente esteso ad un numero qualsiasi di bit.

Ingressi				*Uscita*
A_1	A_0	B_1	B_0	E
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	1
0	1	1	0	0
0	1	1	1	0
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	0
1	1	1	1	1

a)



b)

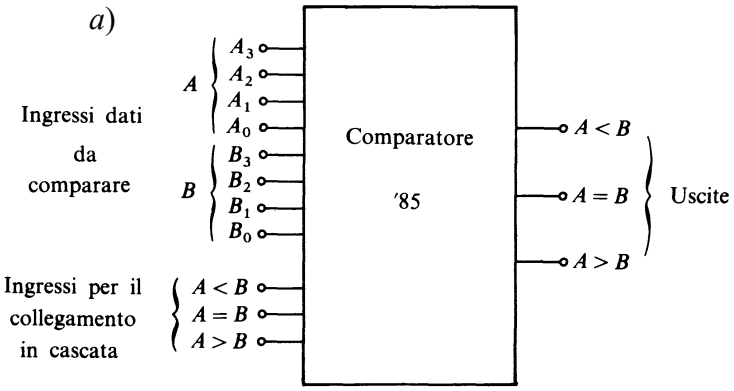
Fig. 1.23 – *a) Tabella di verità di un comparatore per parole a 2 bit. b) Schema logico relativo*

Nei sistemi digitali si presenta spesso la necessità di disporre, oltre che della segnalazione di uguaglianza ($A = B$), di un'indicazione di maggioranza; si richiede cioè la presenza di un'uscita supplementare ($A > B$), che diventi attiva quando il dato A è maggiore del dato B, accompagnata eventualmente da un'altra uscita che segnali quando $A < B$.

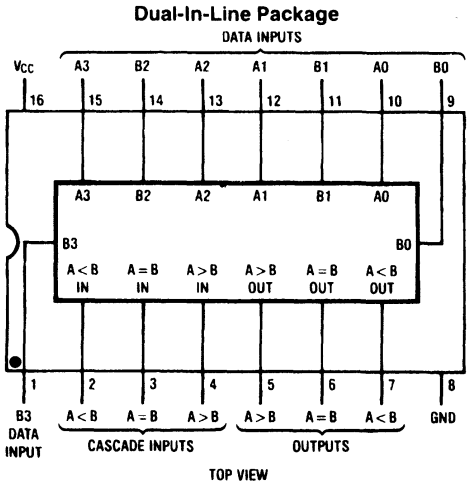
In forma integrata sono disponibili comparatori per parole a 4 e a 8 bit che forniscono tutte o alcune delle indicazioni $A = B$, $A > B$, $A < B$. In Fig. 1.24 sono illustrati il simbolo (Fig. 1.24*a*) e il pin-out (Fig. 1.24*b*) del comparatore di grandezze a 4 bit, tipo 7485, disponibile sia in tecnologia TTL sia in tecnologia CMOS. Si può notare che questo integrato, analogamente ad altri della stessa

classe di funzionamento, è dotato, oltre che dei terminali di ingresso dati e dei terminali di uscita ($A = B$, $A > B$, $A < B$), di ingressi supplementari ($A = B$, $A > B$, $A < B$); questi ultimi (*cascade inputs*) consentono il collegamento in cascata di più integrati dello stesso tipo per realizzare comparatori in grado di confrontare parole ad n bit (con n = 8, 12, 16, ecc.).

a)



b)



c)

Ingressi dei dati da comparare				*Ingressi per il collegamento in cascata*			*Uscite*		
A_3, B_3	A_2, B_2	A_1, B_1	A_0, B_0	$A > B$	$A < B$	$A = B$	$A > B$	$A < B$	$A = B$
$A_3 > B_3$	X	X	X	X	X	X	H	L	L
$A_3 < B_3$	X	X	X	X	X	X	L	H	L
$A_3 = B_3$	$A_2 > B_2$	X	X	X	X	X	H	L	L
$A_3 = B_3$	$A_2 < B_2$	X	X	X	X	X	L	H	L
$A_3 = B_3$	$A_2 = B_2$	$A_1 > B_1$	X	X	X	X	H	L	L
$A_3 = B_3$	$A_2 = B_2$	$A_1 < B_1$	X	X	X	X	L	H	L
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 > B_0$	X	X	X	H	L	L
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 < B_0$	X	X	X	L	H	L
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	H	L	L	H	L	L
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	L	H	L	L	H	L
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	L	L	H	L	L	H
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B$	$A_0 = B_0$	X	X	H	L	L	H
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	H	H	L	L	L	L
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	L	L	L	H	H	L

Fig. 1.24 – *a) Simbolo, b) pin-out e c) tabella di verità del comparatore 7485*

Si considerino innanzitutto le prime otto righe della tabella di verità di Fig. 1.24*c*, in cui, indipendentemente dal valore degli ingressi $A = B$, $A > B$, $A < B$, l'uscita è determinata

esclusivamente dal confronto dei bit dei dati di ingresso A_3 e B_3, A_2 e B_2, A_1 e B_1, A_0 e B_0. Se A_3 e B_3 sono i bit più significativi delle due parole di ingresso e se $A_3 > B_3$, viene attivata l'uscita $A > B$ senza che siano necessari altri controlli; così se $A_3 < B_3$ risulta attivata l'uscita $A < B$. Se viceversa $A_3 = B_3$, le uscite dipendono da A_2 e B_2; se poi $A_2 = B_2$, lo stato delle uscite è determinato dal confronto fra A_1 e B_1 e così via.

Le successive righe della tabella si riferiscono al caso in cui i 4 bit di A sono uguali ai corrispondenti 4 bit di B e pertanto lo stato delle uscite dipende dagli ingressi supplementari $A = B$, $A > B$, $A < B$. Questa situazione trova effettivo riscontro quando le due parole da confrontare sono costituite da un numero di bit superiore a 4 ed occorre usare due o più comparatori in cascata.

1.7 Sommatori

Supponiamo di dover sommare due numeri binari e progettiamo il circuito che realizza tale operazione. Nel caso più semplice i due numeri sono costituiti da un solo bit ciascuno, per cui abbiamo solo quattro casi possibili: 0 + 0 = 0, 0 + 1 = 1, 1 + 0 = 1, 1 + 1 = 0, quest'ultimo caso con un *riporto* di 1. Tale circuito deve avere allora due ingressi, costituiti dai due numeri e da due uscite costruite dal bit somma e dall'eventuale riporto. La sua tabella della verità è rappresentata di seguito in Fig. 1.25:

A	B	R	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Fig. 1.25 – *Tabella di verità per un semi-sommatore*

avendo indicato con R il bit di riporto, detto anche *carry*, e con S il bit somma. Da essa si ricavano le due funzioni canoniche

$$\begin{cases} R = A \cdot B \\ S = \overline{A} \cdot B + A \cdot \overline{B} \end{cases} \tag{1.5}$$

tramite le quali si può disegnare il circuito di Fig. 1.26.

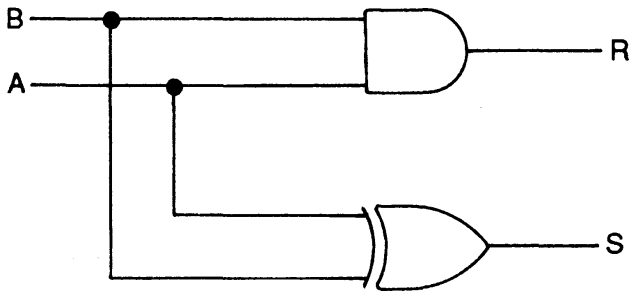
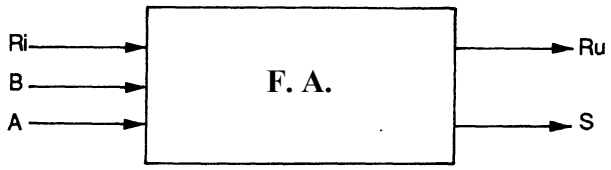


Fig. 1.26 – *Circuito semi-sommatore, o half adder"*

Tale circuito prende il nome di *semi-sommatore* o *half adder*. Cosa gli manca per divenire un *sommatore completo*, ossia un *full adder*? Evidentemente se i numeri da sommare sono costituiti da più di una cifra si deve considerare l'eventuale riporto della somma delle due cifre meno significative. In altri termini, quando si sommano due numeri in colonna, cifra per cifra, si può avere sia un riporto per la colonna successiva (cifre più significative), ma si può anche dovere sommare il riporto dovuto alla somma della colonna precedente (cifre meno significative).

Indicando con R_i il riporto della somma precedente e con R_u il riporto alla colonna successiva, il full adder si può rappresentare secondo lo schema di Fig. 1.27*a* con la tabella di verità riportata in Fig. 1.27*b*.



a)

b)

A	B	Ri	Ru	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Fig. 1.27 – *Full adder: a) schema a blocchi e b) tabella della verità*

Sviluppando la funzione S si ottiene:

$$S = \overline{A}\overline{B}R_i + \overline{A}B\overline{R_i} + A\overline{B}\overline{R_i} + ABR_i = \left(\overline{A}B + A\overline{B}\right)\overline{R_i} + \left(\overline{A}\overline{B} + AB\right)R_i \tag{1.6}$$

e poiché

$$\begin{cases} \overline{A}\cdot B + A\cdot\overline{B} = A\oplus B \\ \overline{A}\cdot\overline{B} + A\cdot B = \overline{A\oplus B} \end{cases}, \tag{1.7}$$

si può infine scrivere

$$S = (A \oplus B) \cdot \overline{R_i} + \overline{(A \oplus B)} \cdot R_i = (A \oplus B) \oplus R_i \ . \tag{1.8}$$

Il circuito per ricavare il bit somma è allora quello di Fig. 1.28

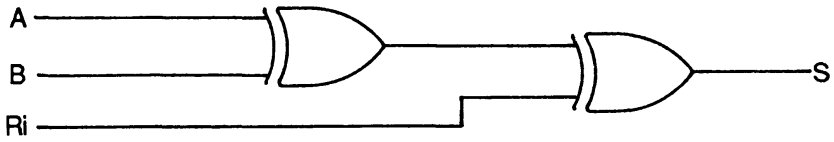


Fig. 1.28 – *Circuito per la generazione del bit somma S*

Per il riporto R_u conviene ricavare la mappa di Karnaugh per la funzione R_u; facilmente si ricava la mappa riportata in Fig. 1.29, in cui sono evidenziati i termini contigui.

Ri \ AB	00	01	11	10
0	0	0	1	0
1	0	1	1	1

Fig. 1.29 – *Mappa di Karnaugh relativa a R_u*

Si ottiene così

$$R_u = AB + \overline{A}BR_i + A\overline{B}R_i = AB + \left(\overline{A}B + A\overline{B}\right)R_i = AB + (A \oplus B)R_i \,, \tag{1.9}$$

che equivale al circuito di Fig. 1.30.

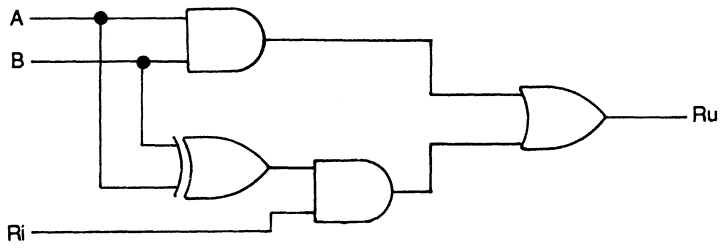


Fig. 1.30 – *Circuito per la generazione del bit di carry R_u*

Combinando i due circuiti si ricava il full adder di Fig. 1.31 che, come si vede, può essere realizzato mediante due half adder (evidenziati in figura) più una OR.

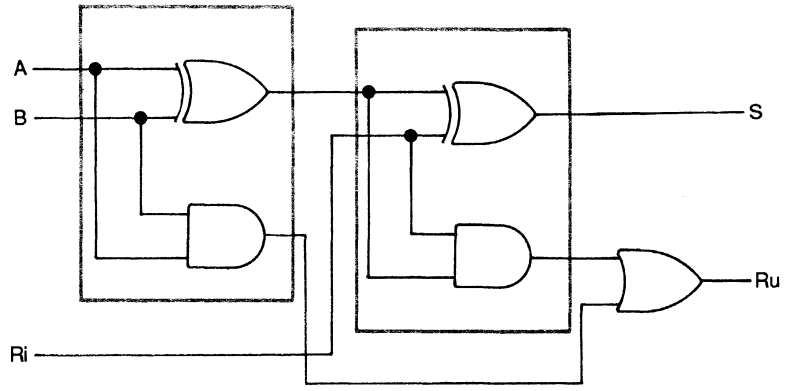


Fig. 1.31 – *Circuito full adder*

Ovviamente non è questo l'unico modo per realizzare un sommatore. I sommatori integrati utilizzano varie tecniche che si differenziano per il tipo di circuiti combinatori impiegati.

Prima di andare ad analizzare il funzionamento di qualche sommatore integrato, osserviamo che per sommare numeri di più cifre occorre utilizzare più circuiti sommatori opportunamente connessi insieme. Per due numeri A e B formati da n cifre ($A_1\ A_2\ \ldots\ A_n$ e $B_1\ B_2\ \ldots\ B_n$, dove con il pedice "1" ci si riferisce al MSB e con "n" al LSB) il circuito sommatore è composto da $(n - 1)$ full adder ed un half adder secondo lo schema di Fig. 1.32. A causa del particolare modo di collegamento dei riporti, il circuito è noto con il nome di *ripple adder* (*sommatore "ondulato"*).

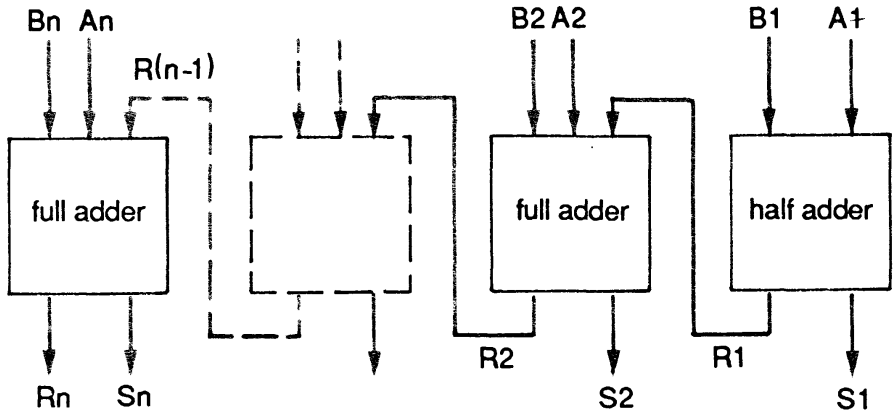
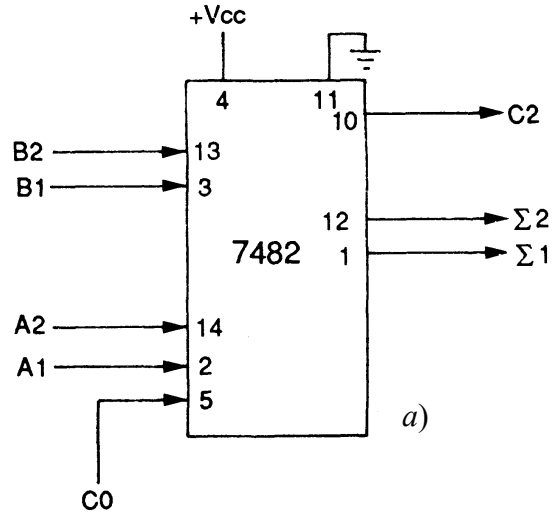


Fig. 1.32 – *Circuito sommatore di numeri a n cifre (ripple adder)*

Un tipico sommatore TTL è il 7482: esso è un sommatore binario completo a 2 bit, il cui schema a blocchi e la relativa tabella di verità sono riportati in Fig. 1.33.

Si noti che A_1 e B_1 sono i LSB mentre A_2 e B_2 i MSB. Quindi se, ad esempio, ci riferiamo alla quarta riga della tabella di verità di Fig. 1.33*b* (caso in cui $C_0 = 0$) è chiaro che Σ_1 (= $A_1 + B_1$) è uguale a 0 (L), dato che 1 + 1 = 0. Vi è tuttavia un riporto di 1 – che chiamiamo C_1 – che va sommato alla colonna successiva, cioè a Σ_2 (= $A_2 + B_2 + C_1$), per cui tale somma è uguale a 1 (H). Ovviamente non vi è riporto C_2.



a)

b)

INPUTS				OUTPUTS					
				WHEN C0 = L			WHEN C0 = H		
A1	B1	A2	B2	Σ1	Σ2	C2	Σ1	Σ2	C2
L	L	L	L	L	L	L	H	L	L
H	L	L	L	H	L	L	L	H	L
L	H	L	L	H	L	L	L	H	L
H	H	L	L	L	H	L	H	H	L
L	L	H	L	L	H	L	H	H	L
H	L	H	L	H	H	L	L	L	H
L	H	H	L	H	H	L	L	L	H
H	H	H	L	L	L	H	H	L	H
L	L	L	H	L	H	L	H	H	L
H	L	L	H	H	H	L	L	L	H
L	H	L	H	H	H	L	L	L	H
H	H	L	H	L	L	H	H	L	H
L	L	H	H	L	L	H	H	L	H
H	L	H	H	H	L	H	L	H	H
L	H	H	H	H	L	H	L	H	H
H	H	H	H	L	H	H	H	H	H

Fig. 1.33 – *a) Schema a blocchi e b) tabella della verità del sommatore TTL 7482*

Per sommare due numeri di quattro bit è necessario utilizzare due integrati 7482, uno – chiamiamolo "Basso" – per la somma dei due bit di peso minore, l'altro – chiamiamolo "Alto" – per la somma dei due di peso maggiore. Il riporto – uscita C_2 – del sommatore Basso deve entrare nel sommatore Alto. Che fare dell'ingresso C_0 di Basso? Non è bene lasciare dei piedini, specie se d'ingresso, scollegati, o, come si dice in gergo "appesi". Poiché sulle cifre meno significative non può esservi riporto da cifre precedenti, l'ingresso in questione dovrà essere sempre basso: dunque lo si collegherà a massa. Lo schema di tale circuito è riportato in Fig. 1.34.

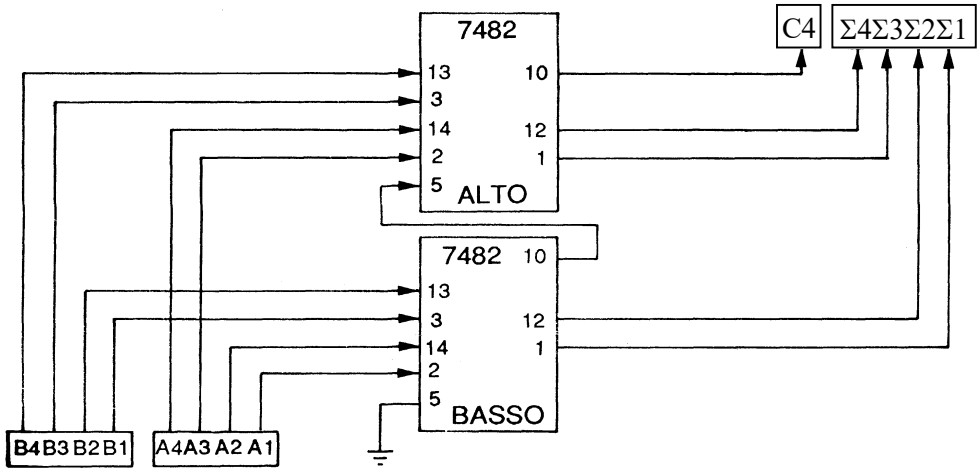
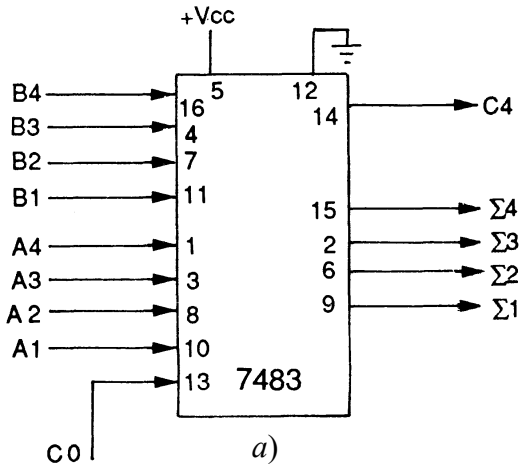


Fig. 1.34 – *Schema circuitale di un sommatore a 4 bit ottenuto tramite due integrati 7482*

Un altro tipico sommatore integrato TTL è il 7483: esso è un sommatore a 4 bit, il cui schema a blocchi e la relativa tabella di verità sono riportati in Fig. 1.35. In pratica tramite un solo 7483 è possibile ottenere il sommatore a 4 bit prima mostrato in Fig. 1.34.



a)

b)

INPUT				OUTPUT					
				WHEN C0 = L			WHEN C0 = H		
				WHEN C2 = L			WHEN C2 = H		
A1	B1	A2	B2	Σ1	Σ2	C2	Σ1	Σ2	C2
L	L	L	L	L	L	L	H	L	L
H	L	L	L	H	L	L	L	H	L
L	H	L	L	H	L	L	L	H	L
H	H	L	L	L	H	L	H	H	L
L	L	H	L	L	H	L	H	H	L
H	L	H	L	H	H	L	L	L	H
L	H	H	L	H	H	L	L	L	H
H	H	H	L	L	L	H	H	L	H
L	L	L	H	L	H	L	H	H	L
H	L	L	H	H	H	L	L	L	H
L	H	L	H	H	H	L	L	L	H
H	H	L	H	L	L	H	H	L	H
L	L	H	H	L	L	H	H	L	H
H	L	H	H	H	L	H	L	H	H
L	H	H	H	H	L	H	L	H	H
H	H	H	H	L	H	H	H	H	H

Fig. 1.35 – *a) Schema a blocchi e b) tabella della verita del sommatore TTL 7483*

Vi è tuttavia un'altra importante differenza tra i due sommatori: il 7483 – come indicato dal costruttore – è provvisto di *carry veloce* (*fast carry*), mentre nel 7482 si ha un *ripple carry*, cioè il carry si propaga da un full adder all'altro secondo lo schema del ripple adder di Fig. 1.32.

La tecnica del ripple carry risulta semplice nell'implementazione circuitale ma presenta lo svantaggio di una limitata *velocità* di calcolo. Infatti, il full adder di ordine più significativo fornirà un risultato corretto solo quando il relativo riporto avrà raggiunto il suo valore stabile. In altre parole la propagazione del riporto dalla cifra meno significativa a quella più significativa limita la velocità di calcolo complessiva del sommatore. Se T_p è il tempo necessario al singolo full adder per eseguire le proprie operazioni, il tempo di calcolo complessivo sarà $n \cdot T_p$, dove con n si è indicato il numero di full adder elementari.

Nel 7483 la velocità di elaborazione del sommatore viene aumentata mediante una particolare circuiteria atta a generare simultaneamente i riporti dei vari full adder. Ciò si realizza tramite una rete combinatoria definita *look-ahead carry generator* (*generatore simultaneo di riporto*), o semplicemente *generatore LAC*. Senza preoccuparci di come tale rete venga realizzata e di quale sia l'algoritmo di calcolo che essa sfrutta, è comunque di notevole interesse confrontare gli schemi logici circuitali (indicati dai costruttori come *functional diagram block*) dei due sommatori 7482 e 7483, Essi sono riportati in Fig. 1.36*a* (7482) e in Fig. 1.36*b* (7483).

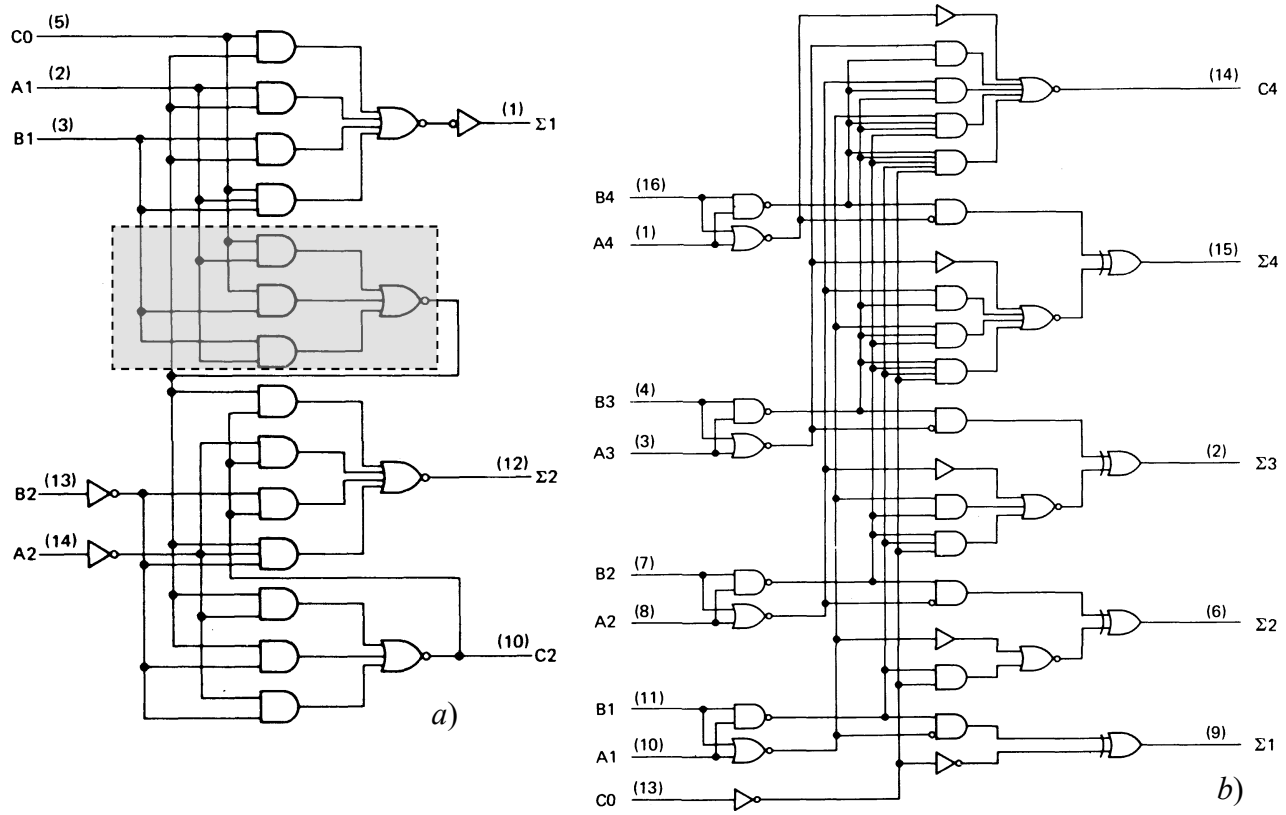


Fig. 1.36 – *Schema logico dei sommatori a) 7482 (ripple carry), b) 7483 (fast carry)*

Nello schema logico del 7482 è individuabile un'uscita di carry relativa alla somma delle due cifre meno significative A_1 B_1 che viene riportata nel blocco relativo alla somma delle due cifre successive A_2 B_2; la rete responsabile della generazione di tale carry è quella posta nel rettangolo grigio in tratteggio di Fig. 1.35*a*. Evidentemente, il blocco circuitale deputato a effettuare la somma tra A_2 e B_2 deve "aspettare" il risultato del blocco di carry. Nello schema del 7483 non si osserva niente di tutto ciò: qui non sono individuabili dei riporti C_1, C_2 o C_3, ma soltanto il riporto finale C_4 risultato combinatorio di tutti gli ingressi. Ovviamente in questo caso la complessità circuitale dei vari full adder aumenta, in quanto non è più possibile dire che la generica cifra d'uscita Σ_i dipenda soltanto da A_i e B_i (e dal riporto) poiché essa dipende da più cifre d'ingresso.

Il sommatore a 4 bit 7483 provvisto di generatore LAC ha una velocità di calcolo di 16 ns con una dissipazione di potenza di circa 76 mW per la serie standard. Di tale integrato esiste anche la versione HCMOS, sebbene il tempo di propagazione salga a 330 ns. Un altro sommatore veloce CMOS a 4 bit è il 4008 (tempo di propagazione: 400 ns).

Si noti infine che per ottenere sommatori a n bit (ad esempio a 8 cifre), è necessario porre in cascata – quindi con un collegamento di tipo *ripple* – più sommatori (ad esempio, due 7483) connettendo il riporto d'uscita C_4 del primo integrato all'ingresso C_0 del secondo. Per ottenere la massima velocità esistono degli integrati *LAC generator* per porre in cascata più sommatori veloci senza usare il ripple carry. Un tipico esempio è il LAC 74182.