Per ottenere tempi di esecuzione ancora più veloci, possono essere utilizzate delle memorie, che contengono la tabella dei risultati per ciascuna combinazione degli ingressi; in questo caso non esiste algoritmo di calcolo e quindi una successione di operazioni elementari, ma una semplice lettura della tabella stessa (*look-up table*).

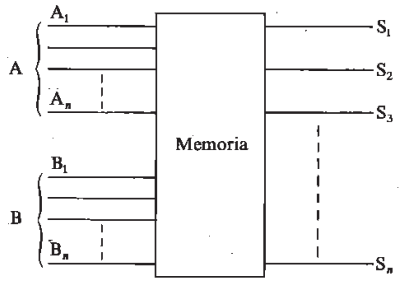

Fig. 2.3.1.

Questo sistema è praticamente utilizzabile, quando il numero di bit di ciascuno dei fattori non è molto elevato.
Con fattori di n bit ciascuno, devono essere infatti memorizzate 2^{2n} possibili combinazioni.
Con $n=8$ ad esempio occorrerebbe una memoria di $2^{16} = 65536$ celle.
Un caso particolare di moltiplicazione, che risulta molto semplice da eseguire, è quello in cui uno dei due fattori è una potenza di due; in questo caso è sufficiente far slittare a sinistra i bit dell'altro fattore di tante volte, quanto è l'esponente di due.
Ad esempio

$$\begin{array}{cccc} 010111 & \times & 10 & = & 0101110 \\ (23) & & (2) & & (45) \end{array}$$

$$\begin{array}{cccc} 001010 & \times & 1000 & = & 01010000 \\ (10) & & (8) & & (80) \end{array}$$

2.4. Divisione di numeri binari

Anche la divisione tra numeri binari può essere effettuata come successione di sottrazioni del dividendo dal divisore, fino ad arrivare ad un valore minore del dividendo.
Come per la moltiplicazione questo procedimento può risultare molto lungo a seconda dei numeri in gioco.
Esistono anche in questo caso algoritmi opportuni, che in un numero di cicli limitato eseguono l'operazione, tramite successione di shift e somma.

3. LE STRUTTURE PROGRAMMATE

3.1. Introduzione

Il grande sviluppo dell'elettronica nel campo dei computer, dei controlli di processo e industriali e nei settori più svariati, verificatosi in questi ultimi anni è sicuramente dovuto all'evolversi della tecnologia dei circuiti integrati e in particolare alla diffusione dei microprocessori.
I primi microprocessori apparsi sul mercato negli anni '70, racchiudevano in un unico circuito integrato i blocchi funzionali caratteristici della struttura di un calcolatore e, più in particolare, delle strutture microprogrammate (già utilizzate per sistemi di controllo industriale), e gettarono le basi dell'*architettura dei sistemi* a microprocessore attualmente usata.
Come infatti sarà visto nel seguito, i sistemi a microprocessori hanno un'architettura standard, che risponde a regole ben precise, legate alla struttura del microprocessore; d'altra parte il microprocessore, a seconda del tipo, ha a sua volta un'architettura diversa e cioè internamente contiene blocchi funzionali differenti.
La grande innovazione portata dai microprocessori consiste nel passaggio dalla cosidetta *logica cablata* alla *logica programmata*.
Per logica cablata si intende la realizzazione di una prefissata rete combinatoria o sequenziale realizzata mediante interconnessione di circuiti digitali di complessità più o meno grande (al limite tramite gli elementi più semplici e cioè le porte logiche, come visto nella teoria delle sintesi delle reti combinatorie e sequenziali).

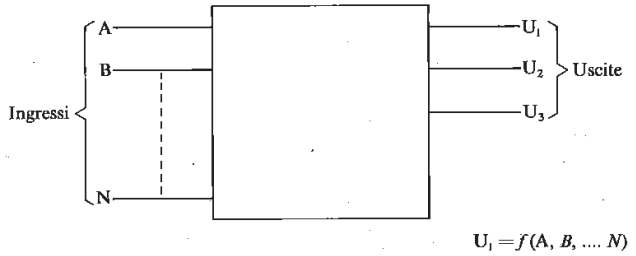

Fig. 3.1.1. Rete combinatoria

La realizzazione dei circuiti a logica cablata, prevede la connessione di circuiti logici, in numero più o meno grande a seconda della complessità della

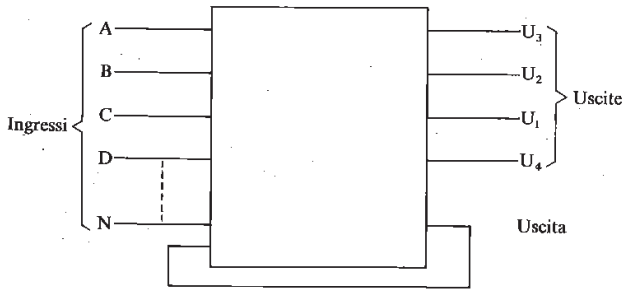

Fig. 3.1.2. Rete sequenziale

funzione da realizzare; nel progetto di tali reti inoltre dovrà essere considerato con attenzione il problema delle temporizzazioni (tenendo conto del ritardo non nullo introdotto da ciascun circuito).
Se la funzione logica da realizzare cambia, naturalmente cambia il circuito sia come componenti, che come interconnessioni.
In una realizzazione industriale, questo comporterebbe, anche per una variazione minima della funzionalità del circuito, la riprogettazione del circuito tenendo conto anche delle temporizzazioni, la sostituzione dell'intero circuito e di conseguenza il rifacimento del circuito stampato e la non utilizzazione dei circuiti stampati precedenti (per apparecchiature prodotte in gran numero ciò porterebbe a un costo elevato).
Per *logica programmata* si intende invece un circuito con configurazione standard (cioè con componenti e interconnessioni fissate) che si presta a realizzare le più svariate funzioni logiche (sia combinatorie che sequenziali); la funzionalità del circuito risiede nel **programma** e cioè in una sequenza di **istruzioni** contenute in una memoria non volatile.
È chiaro a questo punto il grande vantaggio della logica programmata dal punto di vista della modificabilità, della facilità di collaudo e del risparmio di componenti; infatti a parità di circuito (*hardware*) è possibile realizzare funzioni logiche diverse (e quindi apportare modifiche a sistemi già esistenti) cambiando soltanto il contenuto di una memoria (variando cioè il programma) senza alterare il circuito sia come componenti che come interconnessioni.
Chiaramente la convenienza di questa ultima soluzione è più marcata per funzioni logiche di una certa complessità, in cui cioè il circuito a logica cablata comporterebbe un gran numero di circuiti integrati, con tutti i problemi connessi sia per quanto riguarda l'affidabilità, la collaudabilità, e la manutenzione.
Un circuito a logica programmata con una certa configurazione hardware, si presta a risolvere funzioni logiche dalle più semplici alle più complesse con un limite superiore legato alla struttura del circuito stesso e alla capacità della memoria di programma.

La scelta, dunque, di un circuito a logica cablata o programmata deve tener conto di tutti i fattori precedentemente enunciati e ovviamente della complessità del problema da risolvere; per esemplificare questo concetto è chiaro che una struttura a logica programmata, che prevede un certo numero minimo di componenti deve avere un grado di sfruttamento adeguato (una funzione logica molto semplice, realizzata cioè a logica cablata, con un basso numero di componenti e con un circuito stampato semplice, e quindi di basso costo, se realizzato a logica programmata, comporterebbe viceversa il numero minimo di componenti per tale struttura, numero che potrebbe essere molto maggiore di quello precedente).
In questa scelta, bisogna naturalmente tener conto anche di altri fattori sia tecnici che economici: infatti ulteriori parametri per la progettazione di un circuito digitale sono l'ingombro, la velocità cui deve lavorare il circuito stesso, e un costo di investimento per le attrezzature del laboratorio.
Per apparecchiature a logica programmata, occorrono infatti per la progettazione del programma e per la sua verifica, speciali sistemi, detti sistemi di sviluppo, che consentono cioè di predisporre la memoria di programma da inserire nella struttura a logica programmata.

3.2. Strutture microprogrammate

Questo tipo di struttura ha costituito storicamente il passaggio dalla logica cablata a quella programmata e in particolare ai sistemi a microprocessore, anche se vengono ancora comunemente utilizzate per particolari applicazioni di controllo in cui cioè un sistema a microprocessore potrebbe essere eccessivo e, viceversa, uno a logica cablata troppo dispendioso.
Il concetto base è quello di utilizzare una memoria non volatile (ad esempio, una memoria a sola lettura ROM), inserendo in essa, in corrispondenza alle varie celle di memoria, delle informazioni che corrispondono alle funzionalità desiderate dall'apparecchiatura.
Come già visto, una memoria a sola lettura ROM si presenta ai suoi morsetti:

A. con un certo numero di indirizzi (*address*), numero che dipende dalla configurazione della memoria stessa e cioè dalle celle contenute secondo la relazione
$M = 2^N$ dove M è il numero di celle
N è il numero di bit degli indirizzi.

B. un numero di bit che corrisponde al dato relativo ad ogni cella (DATA): tipicamente 4 o 8 bit.

C. un segnale di abilitazione (*chip select* o *chip enable*) che in corrispondenza al livello attivo (alto o basso a seconda del componente) consente il trasferimento sui bit di dati delle informazioni relative alla cella selezionata.

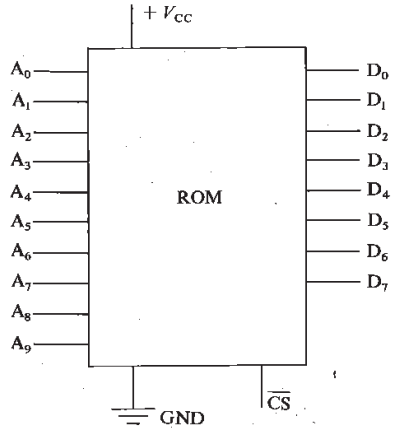

Fig. 3.2.1.

La ROM mostrata in figura 3.2.1 in base a quanto detto in precedenza ha configurazione 1024 × 8, cioè 1024 celle (individuate da 10 bit di indirizzo) ciascuna da 8 bit.
Naturalmente a seconda della tecnologia usata (TTL, MOS, CMOS, ecc.), saranno diversi i *tempi di accesso* cioè il tempo intercorrente tra l'aver fissato un certo indirizzo e quello in cui si presenta in uscita il dato stabile.
La scrittura della ROM, detta anche programmazione, viene fatta per mezzo

Tab. 3.2.I. *Tabella di programmazione**

A_N	....	A_2	A_1	A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
0	....	0	0	0	0	1	0	1	1	0	1	1	(5B)
0	....	0	0	1	0	0	1	0	0	1	0	0	(24)
0		0	1	0	0	1	1	1	1	1	1	1	(7F)
0		0	1	1	.								
0		1	0	0	.								
0		1	0	1	.								
0		1	1	0	.								
0		1	1	1	.								
.		.	.	.	.								
.		.	.	.	.								
.		.	.	.	.								
.		.	.	.	.								
.		.	.	.	.								
.		.	.	.	.								
.		.	.	.	.								
1	....	1	1	1	1	0	1	1	1	1	0	1	(BD)

* Normalmente per facilità di lettura della tabella, sia gli indirizzi che i dati vengono scritti in codice esadecimale.

di una speciale apparecchiatura con predisposizioni a seconda del tipo di ROM usata detta programmatore di ROM (questo è valido per le memorie ROM programmabili -PROM- mentre per le ROM in senso stretto la programmazione viene effettuata *a maschera* cioè direttamente quando vengono prodotte in fabbrica).
La programmazione viene effettuata sulla base di una tabella in cui in corrispondenza a ciascun indirizzo vengono specificati i valori dei dati in uscita. Un esempio potrebbe essere quello della tabella 3.2.I.

3.2.1. Reti combinatorie

Tornando alle strutture microprogrammate, il caso più semplice è quello che realizza una rete combinatoria.
Tale rete, come già visto, possiede un certo numero di ingressi e un certo numero di uscite: ciascuna uscita è legata agli ingressi tramite una espressione booleana come si può desumere dalla tabella di verità.

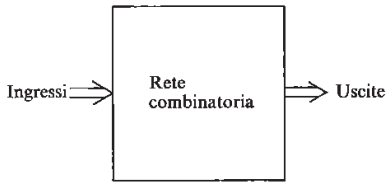

Fig. 3.2.2.

I metodi di sintesi e di minimizzazione sono stati esaminati in precedenza e portano a circuiti comprendenti porte logiche; va comunque osservato che la complessità del circuito dipende dal numero delle variabili di ingresso, come pure la complessità di effettuare la sintesi (sono stati esaminati infatti metodi di minimizzazione fino a 5 variabili di ingresso).

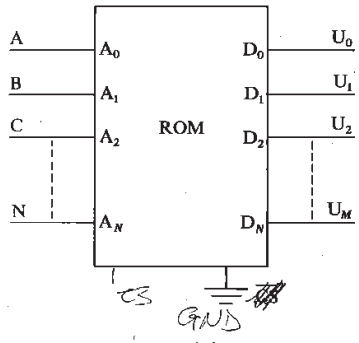

Fig. 3.2.3

Utilizzando una ROM (fig. 3.2.3), viceversa, nota la tabella di verità della rete combinatoria, si può realizzare la funzione logica, collegando gli ingressi agli indirizzi della memoria e le uscite della rete ai dati in uscita; naturalmente la ROM deve essere programmata secondo le tabelle della verità.
Appare evidente il vantaggio di quest'ultima soluzione rispetto a quella a logica cablata, dal punto di vista del numero di componenti impiegati e quindi della economicità: facendo ad esempio riferimento a una memoria ROM di configurazione 256 × 8 è possibile realizzare con un unico integrato 8 reti combinatorie indipendenti, ciascuna con 8 ingressi.
Per completare questo confronto, si pensi alla complessità, in termini di porte logiche impiegate di una rete combinatoria con 8 ingressi!

Esempio

Si voglia realizzare un circuito che trasformi una parola di 4 bit in codice binario naturale, in codice BCD.
Dato che il numero massimo rappresentabile in binario con 4 bit è 15, e che ciascuna cifra in BCD è rappresentata con 4 bit, il numero di uscite della rete dovrebbe essere 8; dato però che la cifra più significativa è o 0 o 1, i 3 bit più significativi di tale cifra sono sempre nulli, e quindi non dipendono dalle variabili di ingresso.
Pertanto le uscite da generare sono 5.
La tavola di verità è mostrata in tabella 3.2.II.

Tab. 3.2.II. *Tavola di verità delle reti combinatorie*

Ingressi				*Uscite*				
D	C	B	A	U4	U3	U2	U1	U0
0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	1
0	0	1	0	0	0	0	1	0
0	0	1	1	0	0	0	1	1
0	1	0	0	0	0	1	0	0
0	1	0	1	0	0	1	0	1
0	1	1	0	0	0	1	1	0
0	1	1	1	0	0	1	1	1
1	0	0	0	0	1	0	0	0
1	0	0	1	0	1	0	0	1
1	0	1	0	1	0	0	0	0
1	0	1	1	1	0	0	0	1
1	1	0	0	1	0	0	1	0
1	1	0	1	1	0	0	1	1
1	1	1	0	1	0	1	0	0
1	1	1	1	1	0	1	0	1

La sintesi di questa rete combinatoria, condotta con i metodi esaminati in precedenza, porta alla costruzione di 5 mappe di Karnaugh (una per uscita). I circuiti corrispondenti, come può facilmente verificare il lettore, sono mostrati in figura 3.2.4 e rispondono alle seguenti espressioni booleane:

$U4 = DC + D\overline{B}$
$U3 = D\overline{C}\overline{B}$
$U2 = \overline{D}C + C\overline{B}$
$U1 = \overline{D}B + DC\overline{B}$
$U0 = A$

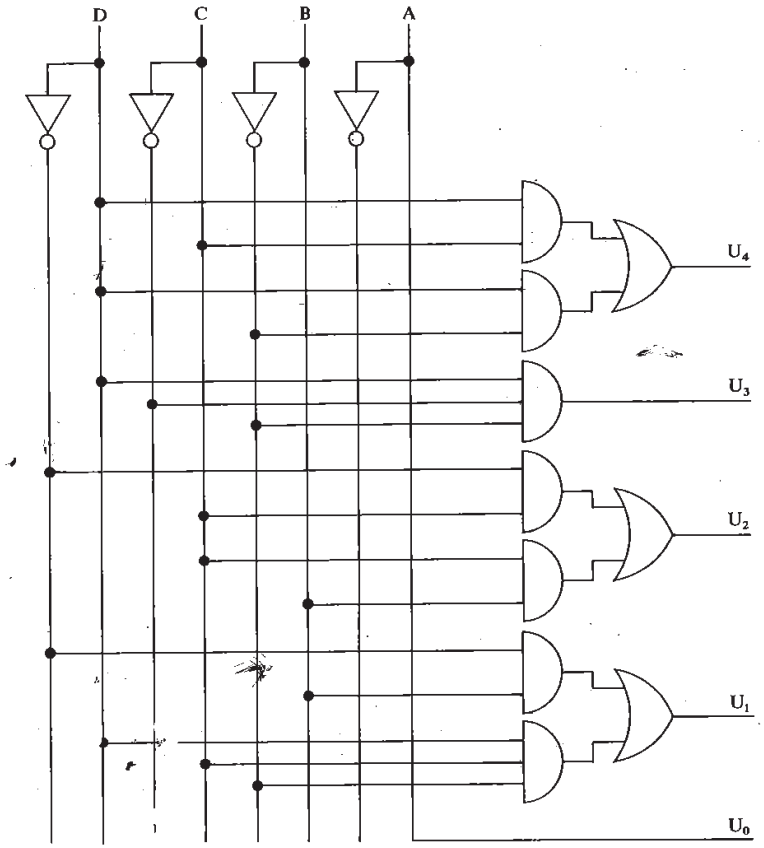

Fig. 3.2.4.

Nella realizzazione pratica di questo circuito sarebbero necessari, facendo riferimento a integrati TTL, 4 componenti e in particolare:

1 7404 (6 inverter)
1 7408 (4 AND a 2 ingressi)
1 7432 (4 OR a 2 ingressi)
1 7411 (3 AND a 3 ingressi)

Volendo realizzare lo stesso circuito a logica programmata sarebbe necessaria in questo caso una memoria ROM di configurazioni 16 × 5 (cioè con 4 indirizzi).
Utilizzando una PROM commerciale, esistente sul mercato, ad esempio una di configurazione 32 × 8 (Texas 74288 o AMD 27S19), il circuito sarebbe il seguente:

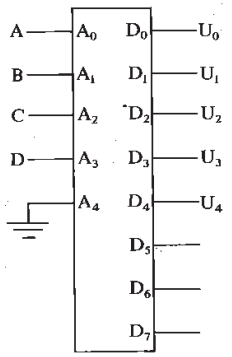

Fig. 3.2.5.

comprendente quindi un unico circuito integrato.
La tabella di programmazione della PROM è la tabella di verità della rete combinatoria 3.2.II.
Naturalmente con il tipo di PROM impiegata, la potenzialità è maggiore e si potrebbe realizzare una decodifica a 5 bit, e quindi per numeri binari di valore fino a 31 (utilizzando una sesta uscita).
In questo caso la variazione da 4 a 5 bit comporta la riprogammazione della PROM, mentre nella soluzione a logica cablata comporterebbe un aumento considerevole di componenti.

3.2.2. Reti sequenziali

Come già visto in precedenza, in una rete sequenziale le uscite dipendono oltre che dagli ingressi, dall'uscita nello *stato* precedente; il concetto di stato presuppone che vi sia una base di tempi e cioè una temporizzazione costituita da un clock che scandisce le varie sequenze del funzionamento della rete (contraddistinte da T1 nella figura 3.2.6).
Una rete sequenziale deve avere in qualche modo una *memoria* delle uscite in un determinato istante, per poter preparare l'uscita all'istante successivo (cioè al colpo di clock successivo).
Gli esempi più semplici di reti sequenziali sono i flip-flop, i contatori, ecc.
La sintesi di una rete sequenziale prevede, come già visto, la schematizzazione del problema in esame, in un **diagramma degli stati**.

Ogni stato elenca i valori delle uscite e l'indicazione dello stato successivo cui si perviene.

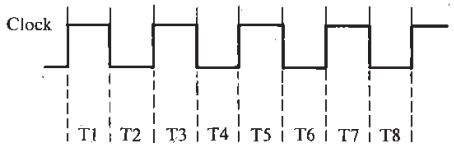

Fig. 3.2.6.

Si debba ad esempio schematizzare una rete sequenziale che, in corrispondenza di un clock, generi in uscita dei livelli bassi e alti secondo il disegno di figura 3.2.7, con periodicità di 8 cicli di clock.

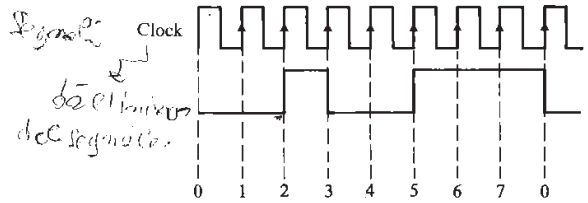

Fig. 3.2.7.

Il corrispondente diagramma degli stati è il seguente (fig. 3.2.8):

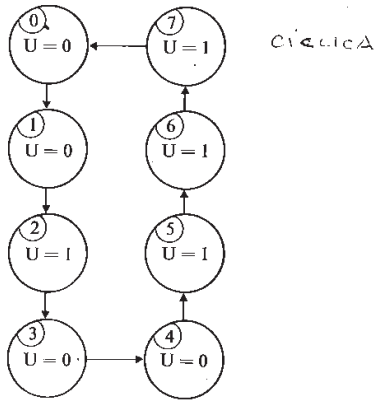

Fig. 3.2.8.

Si osservi come, in questo semplice esempio, da uno stato si passa al successivo, tranne che dallo stato 7 da cui si ritorna allo stato 0.
Se la periodicità della sequenza è multipla di 2, come nell'esempio illustrato,