7 *MACCHINE SEQUENZIALI*

7.1 Schema generale

Nel capitolo 5 sono state introdotte le funzioni sequenziali ed è stato risolto in forma generale il problema delle funzioni di ingresso per un elemento di memoria.

Nel capitolo 6 sono stati presentati alcuni esempi riguardanti diversi tipi di contatore che, in forma generale, possono essere rappresentati da uno schema a blocchi del tipo di fig. 7.1.1, dove è messo in evidenza il registro degli elementi di memoria (R) e il blocco di logica combinatoria (LC) che realizza le funzioni degli ingressi.

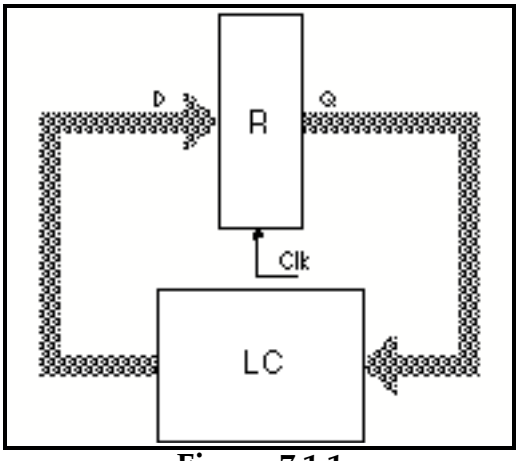

Figura 7.1.1

Nella figura in oggetto le variabili che vengono trattate dalla logica combinatoria provengono unicamente dal registro dei flip-flop, come nel caso dei contatori, ma questo tuttavia non è il caso più generale. Un'estensione della fig. 7.1.1 ci porta a considerare schemi come quello di fig. 7.1.2, dove le equazioni degli ingressi, e quindi la sequenza, sono funzioni anche dei segnali esterni C_k.

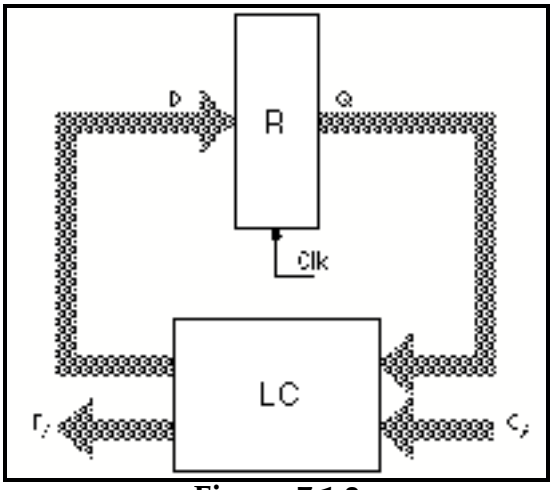

Figura 7.1.2

In questo schema ricadono, ad esempio, contatori che cambiano la sequenza di conteggio in funzione di C_k: conteggio avanti, conteggio indietro o cambio di codice. Più in generale si possono concepire macchine il cui stato all'istante n non dipende solo dallo stato all'istante (n-1), ma anche dallo stato delle linee di controllo o d'ingresso C_k all'istante (n-1). Possiamo dire che queste macchine percorrono sequenze che sono funzione di C_k e, allo stesso tempo, sono in grado di produrre sequenze di segnali di controllo P_l. Se i segnali di controllo P_l. sono funzione sia di C_k che delle uscite Q del registro la macchina sequenziale è detta di Mealey, se invece sono solo funzione di quest'ultime allora la macchina è detta di Moore. Queste sono da preferire perchè forniscono uscite P_l sincronizzate invece nelle macchine di Mealey le uscite possono cambiare, seguendo gli ingressi, in modo asincrono.

Il funzionamento di queste macchine è descritto in modo efficace dai diagrammi di Moore. Un esempio chiarirà l'uso di questi diagrammi.

7.2 Riconoscimento di una sequenza

Supponiamo di voler realizzare un dispositivo che fornisca un'uscita Z=**1** ogni volta che due linee di ingresso M e N compiono le transizioni **1** → **0** e **0** → **1** rispettivamente. Il diagramma che descrive la sequenza degli stati è dato in fig. 7.2.1.

Sui rami sono indicati i valori di M e N che definiscono il passaggio dallo stato n allo stato (n+1). Gli stati distinguibili sono 3

come è facile verificare e all'interno del cerchio che rappresenta lo stato, viene dato il valore di Z.

Dal diagramma di Moore si passa alla tabella della verità, avendo codificato i tre stati con due elementi di memoria A, B.

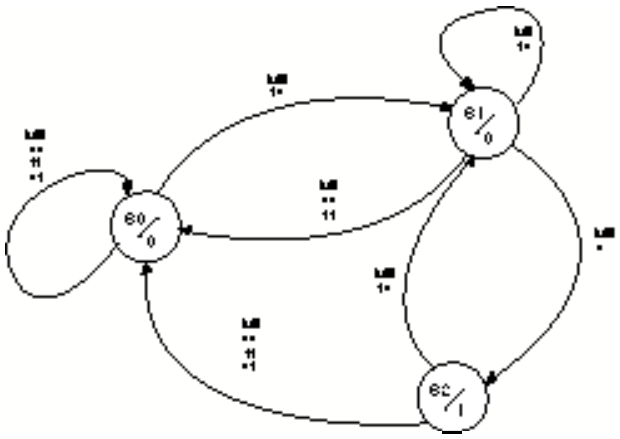

Figura 7.2.1

Tabella 7.2.1

M	N	A^n	B^n	A^{n+1}	B^{n+1}	Z
0	0	0	0	0	0	0
1	0	0	0	1	0	0
0	1	0	0	0	0	0
1	1	0	0	0	0	0
0	0	1	0	0	0	0
1	0	1	0	1	0	0
0	1	1	0	0	1	1
1	1	1	0	0	0	0
0	0	0	1	0	0	0
1	0	0	1	1	0	0
0	1	0	1	0	0	0
1	1	0	1	0	0	0

Dalla tabella si ricavano le equazioni applicative, che vengono semplificate facendo uso dei quattro termini minimi ridondanti corrispondenti ad $A = B = \mathbf{1}$

$$A^{n+1} = (M\overline{N})^n \tag{7.2.1}$$

$$B^{n+1} = (A\overline{M}N)^n \tag{7.2.2}$$

In questo caso il segnale cercato Z coincide con B.

Utilizzando flip-flop di tipo D il circuito che realizza Z è dato in fig. 7.2.2.

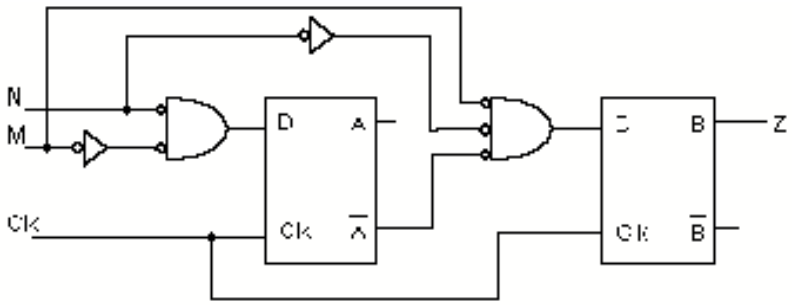

Figura 7.2.2

7.3 Riconoscimento di un codice seriale

Supponiamo di dover riconoscere una qualsiasi sequenza ..0 1 1 0 1 0.. che compaia su una linea *M*. Costruiamo il diagramma di Moore, fig. 7.3.1, partendo dallo stato iniziale S_0. Sui rami è riportato il valore della linea *M*. La tabella della verità 7.3.1, è costruita codificando i sette stati percorsi dal sistema con tre elementi di memoria *A*, *B*, *C*.

Tabella 7.3.1

M	A^n	B^n	C^n	A^{n+1}	B^{n+1}	C^{n+1}
0	0	0	0	1	0	0
1	0	0	0	0	0	0
0	1	0	0	1	0	0
1	1	0	0	0	1	0
0	0	1	0	1	0	0
1	0	1	0	1	1	0
0	1	1	0	0	0	1
1	1	1	0	0	0	0
0	0	0	1	1	0	0
1	0	0	1	1	0	1
0	1	0	1	0	1	1
1	1	0	1	1	1	0
0	0	1	1	1	0	0
1	0	1	1	0	0	0

Tenendo presenti gli stati ridondanti, corrispondenti a S_7, si ottengono le equazioni applicative:

$$A^{n+1} = [\overline{A}(\overline{M} + B\overline{C} + \overline{B}C) + A(\overline{B}\,\overline{C}\,\overline{M} + CM)]^n \qquad (7.3.1)$$

$$B^{n+1} = [\overline{B}(AM + AC) + B(\overline{A}\,\overline{C}M)]^n \qquad (7.3.2)$$

$$C^{n+1} = [\overline{C}(AB\overline{M}) + C(A\overline{M} + \overline{A}\,\overline{B}M)]^n \qquad (7.3.3)$$

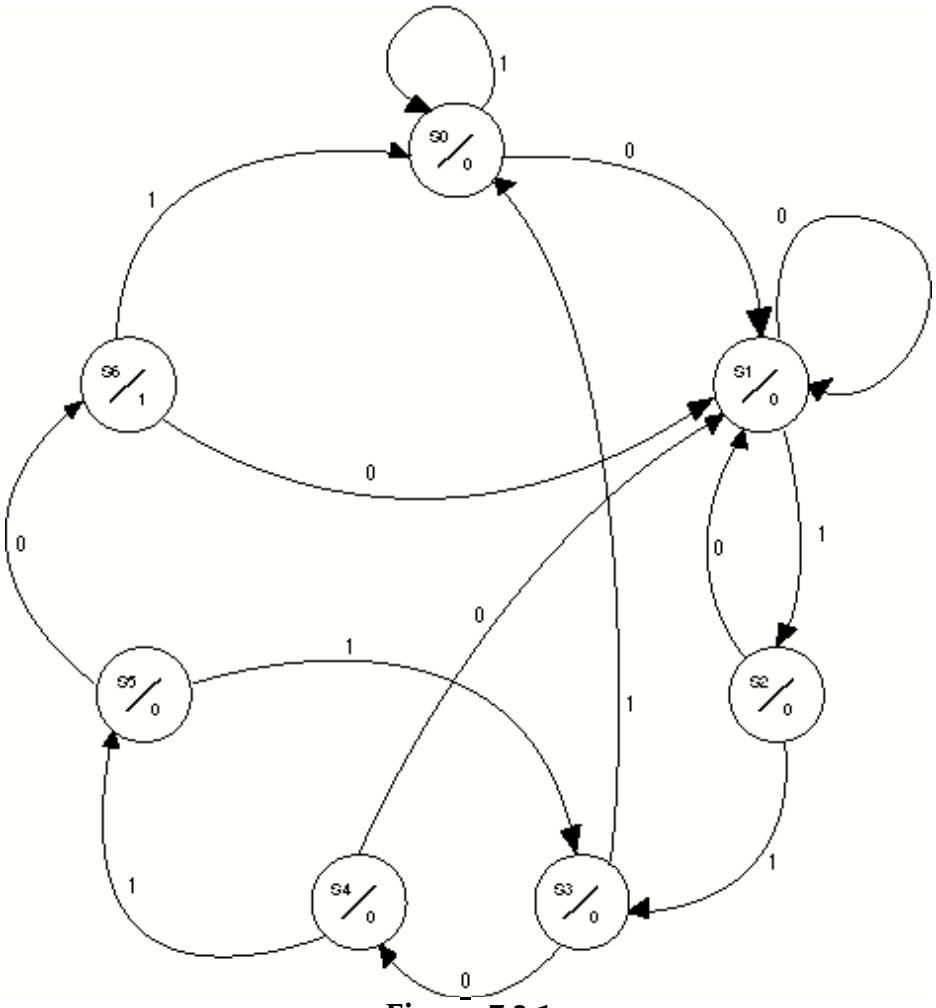

Figura 7.3.1

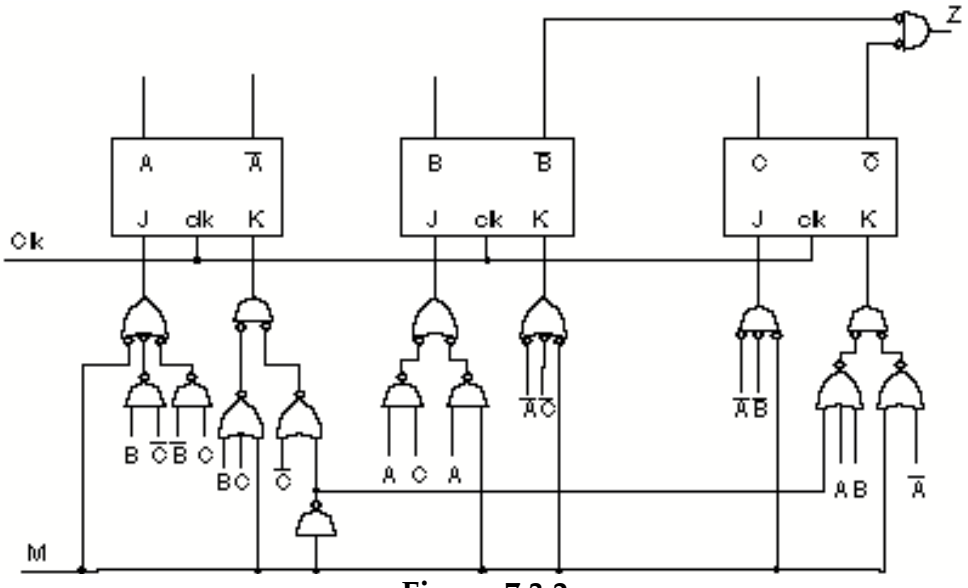

Figura 7.3.2

Realizzando il circuito con J-K e ricordando le soluzioni generali (5.4.7) e (5.4.8) si ottiene lo schema di fig. 7.3.2.

7.4 Sequenziatori a ROM

Negli esempi del presente paragrafo e del precedente non abbiamo considerato la sincronizzazione, col clock del sistema, delle linee C_k d'ingresso. Qualora questo fosse necessario lo schema generale di fig. 7.1.2 diventerebbe quello di fig. 7.4.1.

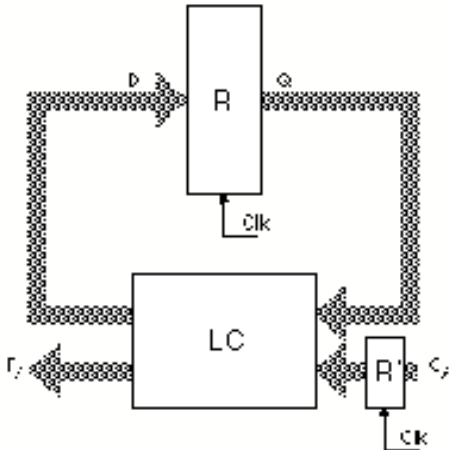

Figura 7.4.1

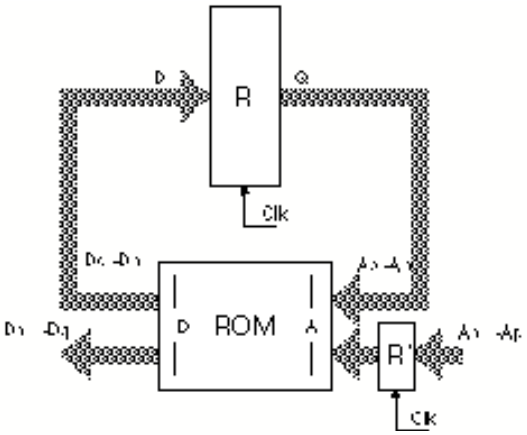

Figura 7.4.2

Ricordando inoltre che le logiche sequenziali si possono realizzare con ROM, lo schema di fig. 7.4.1 si può ridisegnare come in fig. 7.4.2 dove $(D_0,.....,D_q)$ sono i dati in uscita della ROM e $(A_0,.....,A_p)$ sono i bit di indirizzo.

E' facile verificare che con queste strutture si possono realizzare sequenziatori estremamente versatili. Ad esempio, disponendo di una ROM da 256 bytes, raggruppando i bit come in fig. 7.4.3, si possono programmare fino a 16 sequenze diverse, definite da $C_0 \div C_3$, ognuna con un massimo di 16 passi, che forniscono 4 segnali programmati $P_0 \div P_3$.

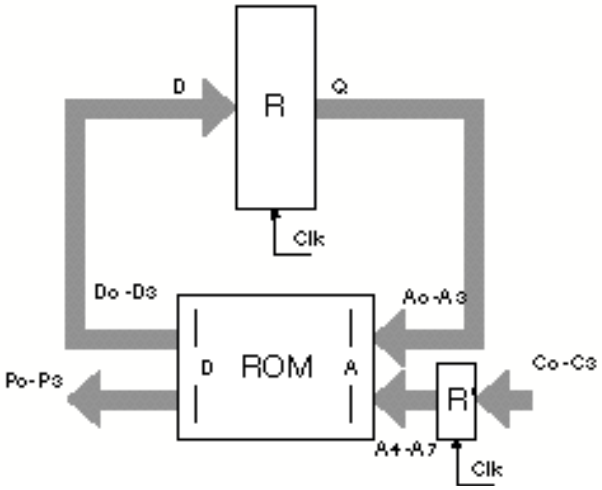

Figura 7.4.3

7.5 Ottimizzazione del circuito

Supponiamo di voler sincronizzare un microprocessore Z80 con una memoria lenta che richieda l'inserzione di un ciclo di "Wait" (TW), nelle istruzioni che fanno riferimento alla memoria, secondo lo schema temporale di fig. 7.5.1.

L'inserzione del ciclo di Wait è comandata dal segnale $\overline{WAIT}$ presente al fronte di discesa di *T*2. Se $\overline{WAIT}$ permanesse durante *TW* si inserirebbe un altro ciclo *TW*.

Il diagramma di Moore per un simile problema è dato in fig. 7.5.2 dove sono distinguibili tre stati.

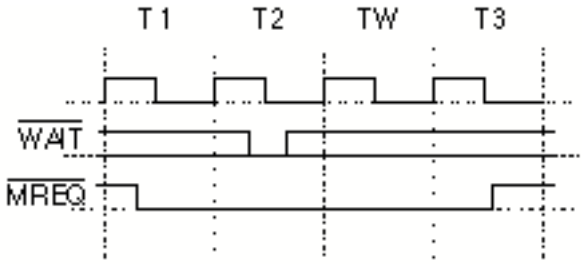

Figura 7.5.1

Si vede che da S_1 si passa a S_2 e da S_2 a S_0 non importa quale sia lo stato di *MREQ*. Il segnale di $\overline{WAIT}$ è generato da S_1.

La tabella 7.5.1 è relativa al diagramma di stato di fig. 7.5.2. Con l'asterisco sono indicati stati ridondanti.

Sono anche ridondanti tutti gli stati con $A = B = \mathbf{1}$. Si vede che le equazioni applicative sono:

$$A^{n+1} = (MREQ \bullet \overline{A}\overline{B})^n \quad (7.5.1)$$

$$B^{n+1} = A^n \quad (7.5.2)$$

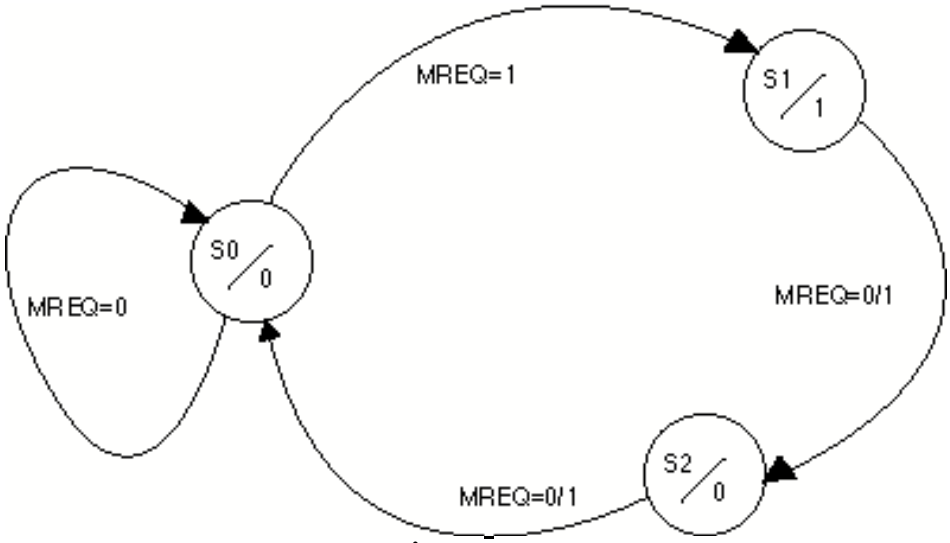

Figura 7.5.2

Tabella 7.5.1

$MREQ$	A^n	B^n	A^{n+1}	B^{n+1}
0	0	0	0	0
1	0	0	1	0
*0	1	0	X	X
1	1	0	0	1
*0	0	1	X	X
1	0	1	0	0

Nella equazione (7.5.1) non si usano, per la semplificazione, stati ridondanti che invece sono usati per la (7.5.2). Il circuito che realizza le (7.5.1) e (7.5.2) è dato in fig. 7.5.3.

Notiamo tuttavia che non è assolutamente necessario che il ritorno allo stato S_0 avvenga da S_2 ma sarebbe consentito un passaggio attraverso un ulteriore stato S_3. In tal caso la tabella 7.5.1 diverrebbe la 7.5.2 dove nuovamente si sono indicati con asterischi gli stati ridondanti.

Le equazioni applicative diventano quindi le seguenti:

$$A^{n+1} = (\overline{A} \bullet MREQ)^n \quad (7.5.3)$$

$$B^{n+1} = [M(A + B)]^n \quad (7.5.4)$$

che evidentemente non semplificano la realizzazione circuitale tenendo presente che risulta:

$$\overline{WAIT} = \overline{A\overline{B}} = \overline{A} + B \quad (7.5.5)$$

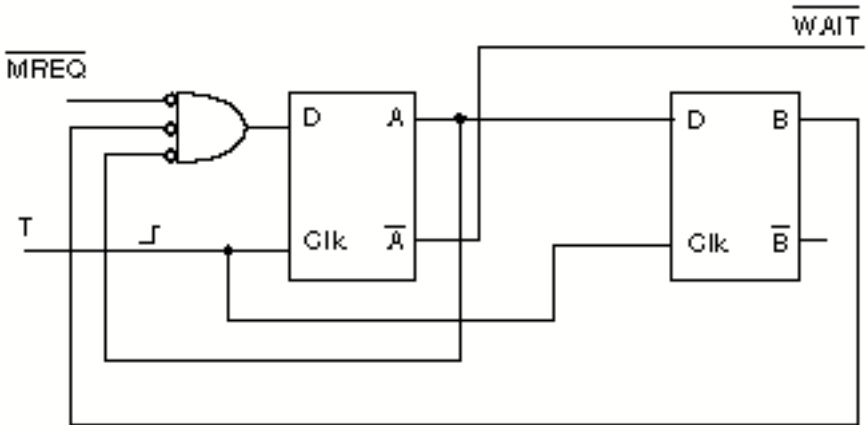

Figura 7.5.3

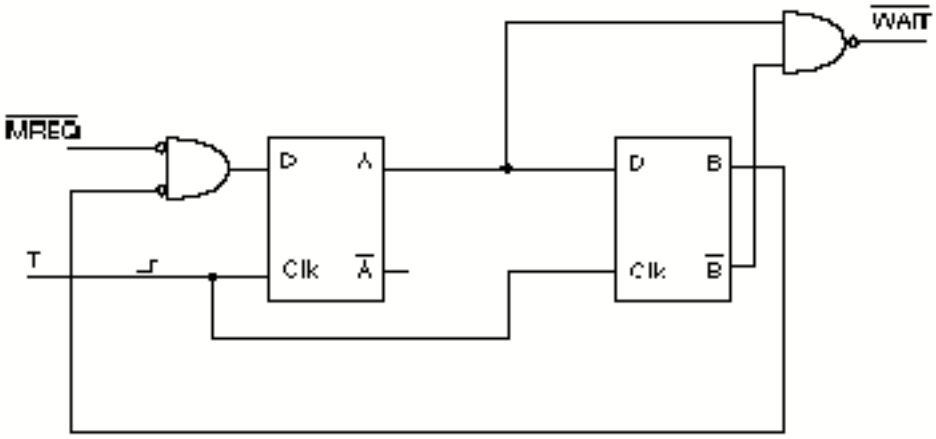

Figura 7.5.4

Tabella 7.5.2

$MREQ$	A^n	B^n	A^{n+1}	B^{n+1}
0	0	0	0	0
1	0	0	1	0
*0	1	0	X	X
1	1	0	0	1
*0	0	1	X	X
1	0	1	1	1
0	1	1	0	0
*1	1	1	X	X

Se invece si cambiano i pesi della sequenza si può ottenere qualche risultato più interessante. Supponiamo che la sequenza sia ora: S_0, S_1, S_3, S_2, S_0, descritta dalla tabella 7.5.3

Tabella 7.5.3

$MREQ$	A^n	B^n	A^{n+1}	B^{n+1}
0	0	0	0	0
1	0	0	1	0
*0	1	0	X	X

1	1	0	1	1
*0	1	1	X	X
1	1	1	0	1
0	0	1	0	0
*1	0	1	X	X

Le equazioni applicative sono ora le seguenti:

$$A^{n+1} = (\overline{B} \bullet MREQ)^n \tag{7.5.6}$$
$$B^{n+1} = A^n \tag{7.5.7}$$

e la funzione cercata è ancora descritta dalla (7.5.5). La realizzazione circuitale è in fig. 7.5.4 che è non molto diversa da quella di fig. 7.5.3.

Tuttavia se consideriamo che la sequenza può durare un ciclo di T in più rispetto a quelle considerate, otteniamo il diagramma di fig. 7.5.5, la cui sequenza è descritta dalla tabella 7.5.4, dove sono messe in evidenza le righe corrispondenti a $A^n = B^n = 1$.

Le equazioni applicative sono ora:

$$A^{n+1} = (MREQ)^n \tag{7.5.8}$$
$$B^{n+1} = A^n \tag{7.5.9}$$

e la funzione richiesta è ancora rappresentata dalla (7.5.5). La soluzione circuitale è data in fig. 7.5.6 mentre nella fig. 7.5.7 è dato il diagramma temporale di A e B nei quattro casi considerati.

Anche in questo caso notiamo che la semplicità del circuito dipende dal modo (peso) in cui sono stati codificati gli stati.

Qualora si fosse scelta la sequenza S_0, S_1, S_2, S_2, S_3, S_0, si sarebbe ottenuta la tabella 7.5.5 che porta alle soluzioni:

$$A^{n+1} = (MREQ \bullet \overline{A}\overline{B})^n \tag{7.5.10}$$
$$B^{n+1} = (MREQ \bullet A + \overline{A}B)^n \tag{7.5.11}$$

che richiedono più componenti circuitali.

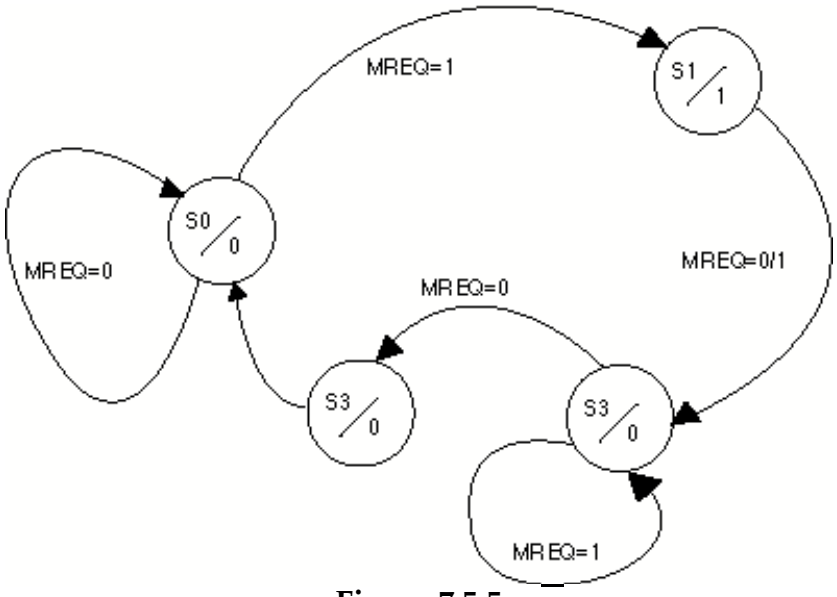

Figura 7.5.5

Tabella 7.5.4

MREQ	A^n	B^n	A^{n+1}	B^{n+1}
0	0	0	0	0
1	0	0	1	0
*0	1	0	X	X
1	1	0	1	1
.....				
0	1	1	0	1
1	1	1	1	1
.....				
0	0	1	0	0
*1	0	1	X	X

Tabella 7.5.5

MREQ	A^n	B^n	A^{n+1}	B^{n+1}
0	0	0	0	0
1	0	0	1	0
*0	1	0	X	X
1	1	0	0	1
0	0	1	1	1
1	0	1	0	1
0	1	1	0	0
*1	1	1	X	X

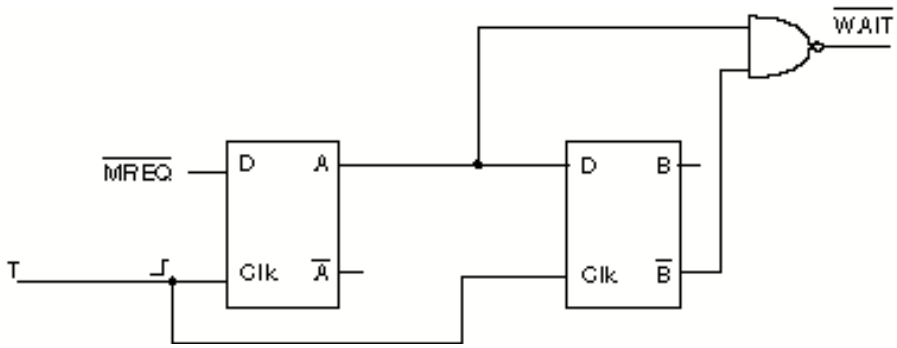

Figura 7.5.6

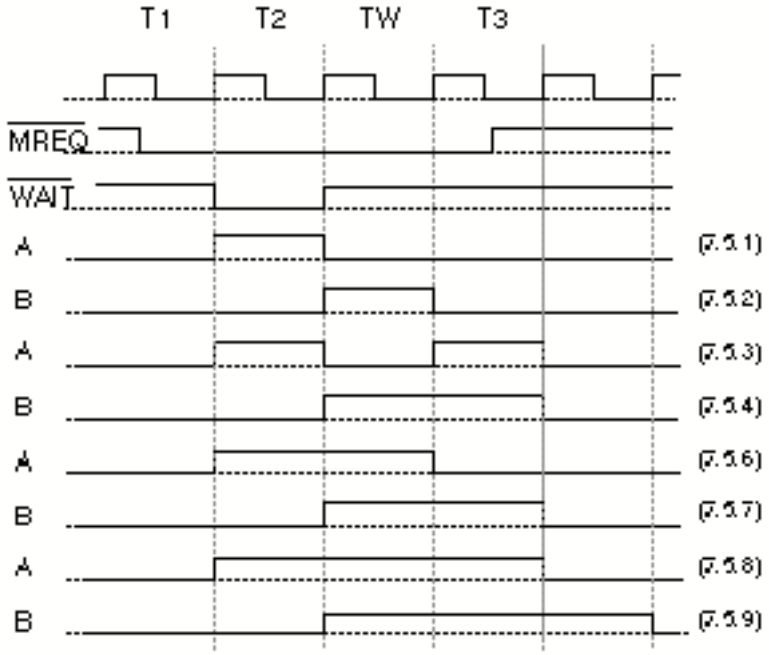

Figura 7.5.7

Possiamo concludere che può accadere che le soluzioni circuitali più semplici dipendano a volte dal modo in cui si codificano gli stati della macchina e dal numero degli stati che vengono consentiti. Naturalmente per macchine a stati molto complesse questo processo di ottimizzazione circuitale può risultare molto difficile.

7.6 Esercizi

a)-Realizzare un dispositivo che riconosca una qualsiasi sequnza di ..1100..che si presenti su una linea *S*. Disegnare il diagramma di Moore e il circuito con flip-flop di J-K e D.

b)-Disegnare il diagramma di Moore per il contatore BDC bidirezionale dell'esercizio b) del §6.8.

c)-Disegnare la macchina definita nel §8.3, fig.8.3.2.

d)-Disegnare una macchina che utilizzi una ROM e sia in grado di riconoscere una di *n* possibili sequenze di *m* stati. Definire le dimensioni della ROM.

7.14