Capitolo 6

Alcuni moduli sequenziali

Come abbiamo già avuto occasione di discutere, le reti logiche vengono frequentemente costruite per realizzare funzioni standard. Nel campo delle reti sequenziali abbiamo studiato in particolare i registri come specifici moduli utilizzati per la memorizzazione di parole. L'impiego di moduli sequenziali non si limita però a funzioni così semplici: studieremo in questo capitolo diverse strutture di largo impiego, che costituiscono blocchi costruttivi elementari dei sistemi.

6.1 Reti autonome e contatori

Diamo il nome di reti sequenziali *autonome* a quelle reti che, pur rispondendo al modello generale di rete sequenziale sincrona (fig. 4.9) non hanno ingressi x. L'unico segnale in ingresso è dunque l'impulso c: la rete evolve tra i suoi stati all'arrivo dell'impulso, cosicché il prossimo stato e l'uscita sono funzione unicamente dello stato presente.[1]

Nel diagramma degli stati di una rete autonoma vi è una sola freccia uscente da ogni nodo (ovvero la tabella di flusso ha una sola colonna). Ne segue che la sequenza di stati che la rete assume, e la sequenza di uscite generate, devono essere periodiche, salvo una eventuale sequenza iniziale fuori periodo. Un esempio di rete autonoma è rappresentato nel diagramma degli stati della figura 6.1 ove, partendo dallo stato B, si genera la sequenza di uscita z : 1$\underbrace{0010}$ $\underbrace{0010}$ ...

[1] Le reti sequenziali nelle quali l'uscita dipende unicamente dallo stato interno (cioè non dall'ingresso) sono dette *reti di Moore*, in contrapposizione alle *reti di Mealy* che rispondono ai legami funzionali del modello generale. Le reti autonome sono ovviamente reti di Moore.

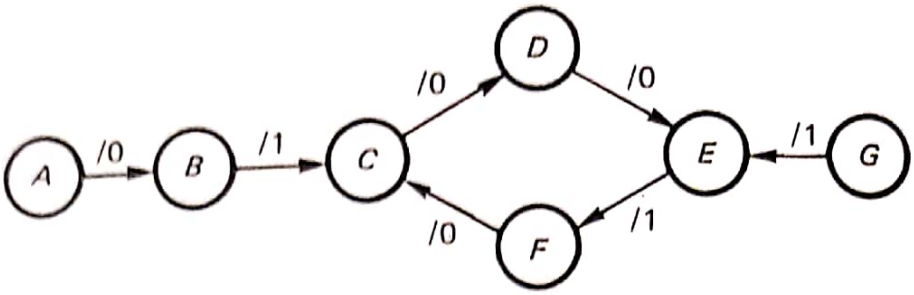


Figura 6.1
Diagramma degli stati di una rete autonoma.

Esempi importanti di reti sequenziali autonome sono i *contatori*, cioè i dispositivi atti a contare, in numerazione binaria, gli impulsi che giungono su una determinata linea, e a rendere accessibile all'esterno il valore raggiunto in ogni istante. Vi è, per esempio, evidente analogia con il contachilometri dell'automobile, che in ogni istante mostra il numero dei chilometri percorsi. Analogia completa perché, come il contachilometri ha una capacità limitata, poniamo cinque cifre decimali, e dopo aver contato fino a 99999 torna a 00000 al chilometro successivo, così il contatore logico ha capacità di n bit, assume le 2^n diverse configurazioni a indicare altrettanti valori, e torna quindi alla configurazione iniziale all'arrivo di un successivo impulso.

Un contatore si costruisce spontaneamente come rete sequenziale attorno a un registro di n bit (ed è per questo spesso indicato come *registro contatore*). Il registro memorizza il valore raggiunto dal conteggio, che si interpreta come stato interno; l'arrivo dell'impulso da contare provoca la transizione al prossimo stato, cioè altera il numero binario contenuto nel registro, trasformandolo nel numero successivo secondo il codice scelto. La rete presenta un'ulteriore semplificazione rispetto al modello generale di rete autonoma, poiché le uscite z_i coincidono con i segnali di anello y_i, in quanto interessa leggere il contenuto del registro. Lo schema è quello della figura 6.2.

Consideriamo per esempio un contatore binario che impiega il codice naturale di tre bit (numeri positivi senza segno): cioè esso conta ciclicamente da 000 a 111. Per progettarlo si segue il procedimento generale di sintesi di una rete sequenziale visto nel capitolo precedente, partendo dal diagramma degli stati (fig. 6.3). Poiché in questo caso non vi è rete interposta tra le y_i e le z_i, i segnali z_i assumono istantaneamente i nuovi valori, e sono associati direttamente agli stati del diagramma. Di conseguenza la scelta delle configurazioni delle y_i per

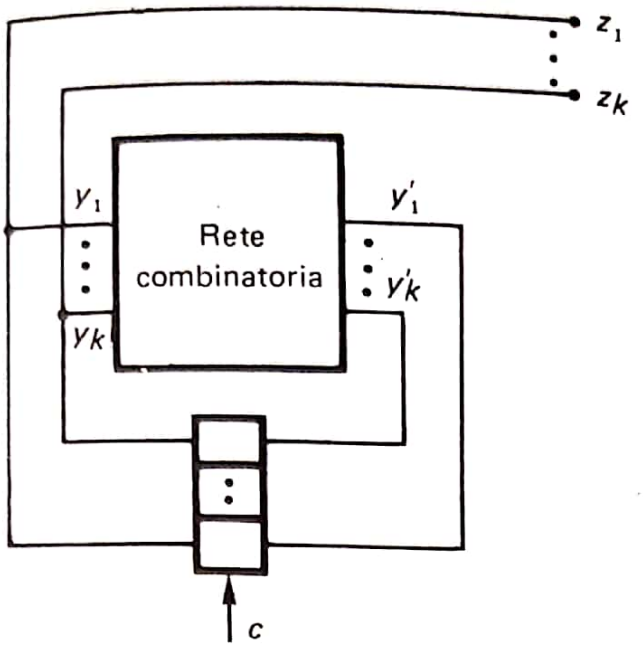


Figura 6.2
Schema di rete sequenziale autonoma di tipo contatore, in cui le variabili z_i coincidono con le y_i.

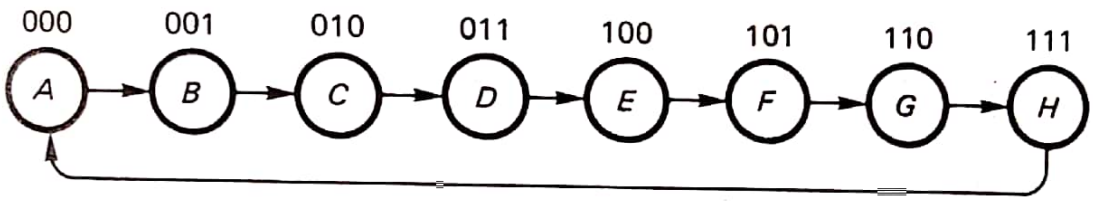


Figura 6.3
Diagramma degli stati di un contatore binario di 3 bit.

codificare gli stati è ora obbligata, poiché queste devono coincidere con le z_i (per esempio lo stato A è codificato con $y_3y_2y_1 = 000$ ecc.).

La tabella delle transizioni, riordinata in forma di mappa di Karnaugh, e le relative funzioni, sono:

y_2y_1 \ y_3	0	1
00	001	101
01	010	110
11	100	000
10	011	111

$y'_3y'_2y'_1$

$$y'_3 = \overline{y}_3 y_2 y_1 + y_3 \overline{y}_1 + y_3 \overline{y}_2 ,$$

$$y'_2 = \overline{y}_2 y_1 + y_2 \overline{y}_1 ,$$

$$y'_1 = \overline{y}_1 ,$$

$$z_i = y_i , \quad i = 1, 2, 3.$$

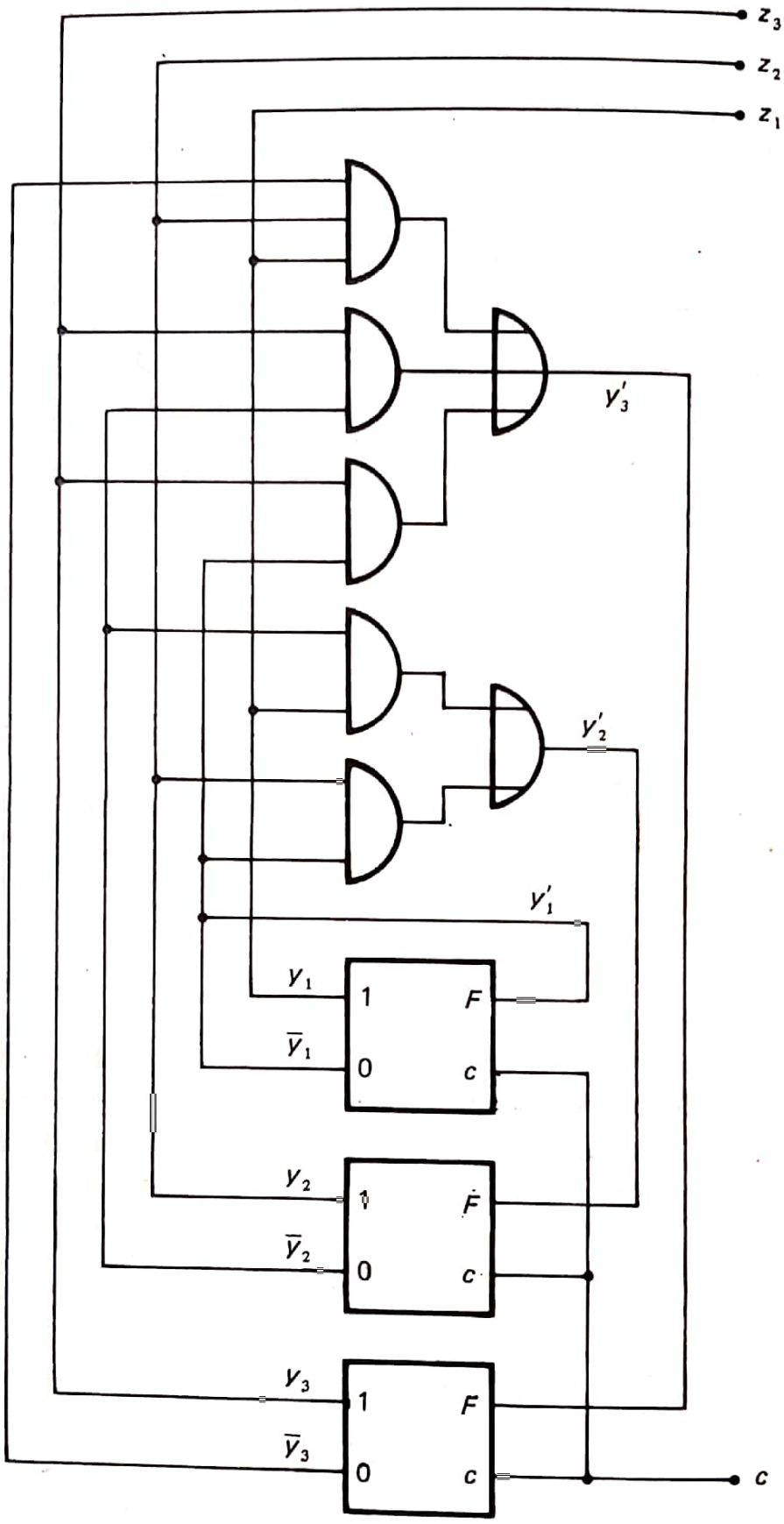


Figura 6.4
Rete logica del contatore a tre bit.

Queste funzioni specificano le interconnessioni tra i flip-flop del contatore: usando flip-flop di tipo *Fc* si ottiene il disegno complessivo di figura 6.4.

6.2 Operazioni sui registri

Esaminiamo ora alcune operazioni elementari che si possono compiere sui registri, mediante trasferimenti di informazione tra i vari flip-flop. Come nel caso dei contatori, ciò permette di estendere la struttura del registro a quella di una rete sequenziale atta a svolgere speciali funzioni sulla parola memorizzata.

Lo *spostamento* (*shift*) di una parola B di n bit $b_n, ..., b_2, b_1$, contenuta in un registro, è la traslazione di un posto di tutti i bit di B. Poiché si suppone che la lunghezza della parola risultante sia uguale a quella della parola originale, lo spostamento sinistro (o destro) di B causerà la perdita del bit b_n (o b_1), e l'azzeramento di b_1 (o b_n), come mostra l'esempio seguente:

b_6	b_5	b_4	b_3	b_2	b_1	
1	0	1	0	0	1	parola B
0	1	0	0	1	0	spostamento sinistro
0	1	0	1	0	0	spostamento destro

Registri dotati dei collegamenti necessari a eseguire lo spostamento della parola si dicono *registri a spostamento* (sinistro o destro). Essi sono reti sequenziali in cui la semplicissima parte combinatoria è costituita da ovvie connessioni tra i flip-flop, che possono essere progettate senza ricorrere al procedimento generale di sintesi.

Un registro a spostamento sinistro è mostrato nella figura 6.5: i flip-flop sono di tipo *FAc* e lo spostamento ha luogo all'arrivo dell'impulso c, se $A=1$. Il flip-flop b_1 è semplicemente azzerato dall'impulso c, che carica il valore costante 0.

La rete di figura 6.5 funziona correttamente solo se sono osservate le regole di temporizzazione delle reti sequenziali sincrone. In particolare è indispensabile che i flip-flop siano di tipo a impulsi, e che la durata dell'impulso c sia inferiore al ritardo proprio dei singoli flip-flop.

Lo schema di un registro a spostamento destro si può immediatamente ricavare dal precedente invertendo l'ordine dei collegamenti. L'operazione di spostamento sinistro o destro può anche essere eseguita in modo *circolare*. In questo caso i bit b_n e b_1 sono considerati adia-

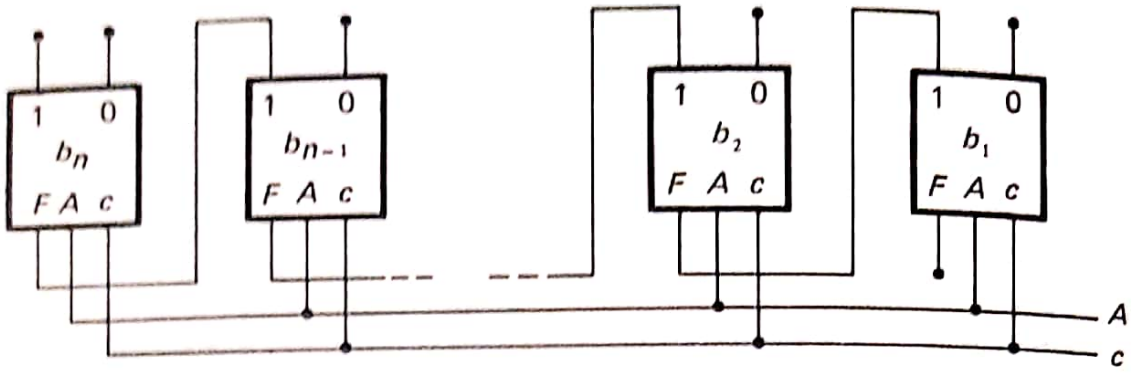


Figura 6.5
Registro a spostamento sinistro formato da flip-flop *FA c*.

centi e il bit che nello spostamento semplice sarebbe perduto è inserito all'estremo opposto della parola. La struttura del registro è ovvia.

Un'altra operazione eseguita comunemente in un registro è la *complementazione*, che consiste nell'invertire il valore logico in ciascuno dei suoi flip-flop. Ciò può essere ottenuto in un tempo elementare secondo lo schema della figura 6.6, che richiede un collegamento tra uscita 0 e ingresso F per ogni flip-flop: la complementazione ha luogo all'arrivo dell'impulso c, se $A=1$.

Seguendo strettamente gli schemi indicati per contatori, registri a spostamento e a complementazione, ciascun registro potrebbe effettuare la sola operazione per cui è stato definito. Per questo gli schemi logici sono in genere arricchiti di circuiti atti a consentire nuove operazioni, tra le quali ha grande importanza il caricamento del registro dall'esterno.

Per esempio il registro a spostamento sinistro della figura 6.5 può

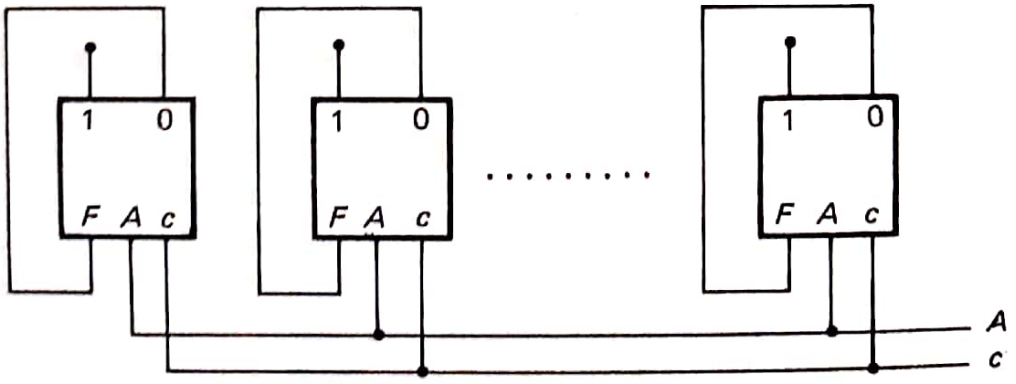


Figura 6.6
Registro a complementazione.

essere trasformato in quello della figura 6.7, dotato della rete necessaria a consentire l'accesso ai suoi flip-flop anche ai segnali E provenienti dall'esterno. Rispetto allo schema precedente, l'ingresso F di ogni flip-flop è pilotato da una rete logica che riceve i segnali S_s, E e l'uscita del flip flop precedente (o la costante 0 per il flip-flop più a destra). Se $S_s = 1$ il registro esegue lo spostamento sinistro; se $S_s = 0$ il registro carica l'informazione proveniente dagli ingressi E. Si noti che la nuova rete logica è di fatto un selettore di ingresso con variabile di controllo S_s. Similmente si estendono gli schemi dei registri contatori, complementatori ecc. per consentirne il caricamento dall'esterno (esercizio 6.1).

A livello di blocco logico, i registri contatori, a spostamento e a complementazione, sono indicati con i simboli di figura 6.8, che includono la rete per il caricamento dall'esterno attraverso gli ingressi indicati sui flip-flop. E' implicita l'applicazione del segnale c di sincronismo.

Semplici connessioni tra diversi registri permettono di eseguire operazioni tra questi. La figura 6.9 indica le connessioni necessarie a eseguire la somma logica X OR Y e a trasferire il risultato in Y. L'operazione è eseguita per $A_y = 1$, mentre è ininfluente il valore di A_x. Si

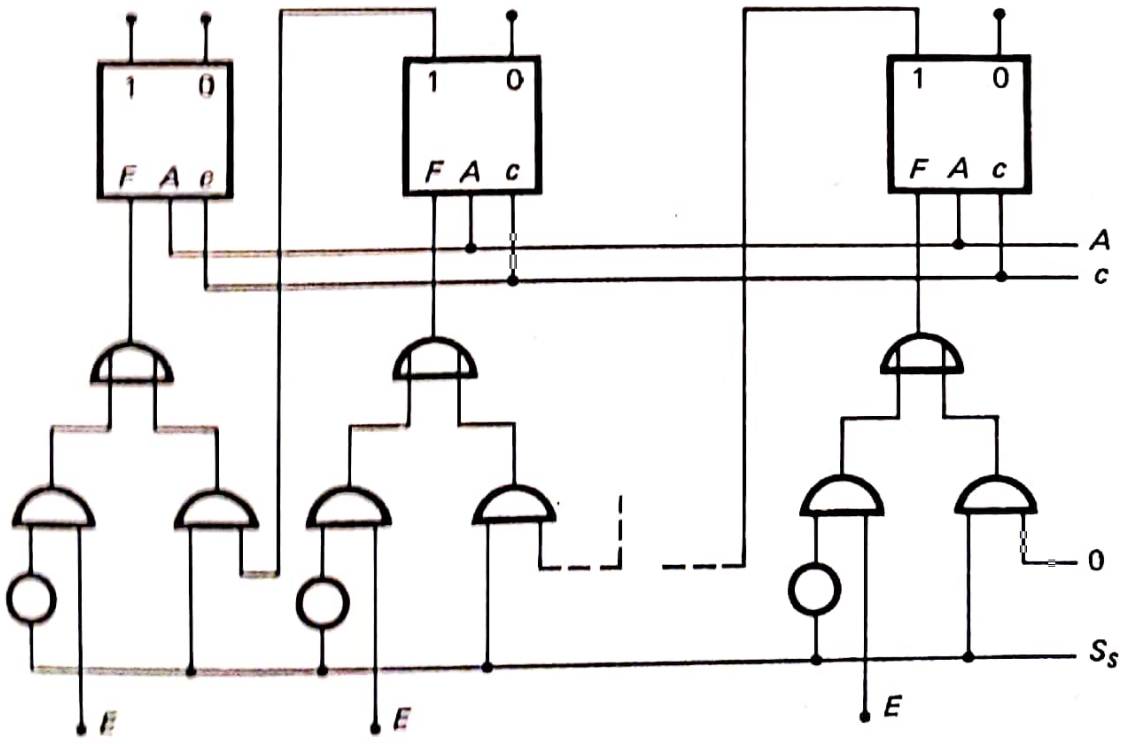
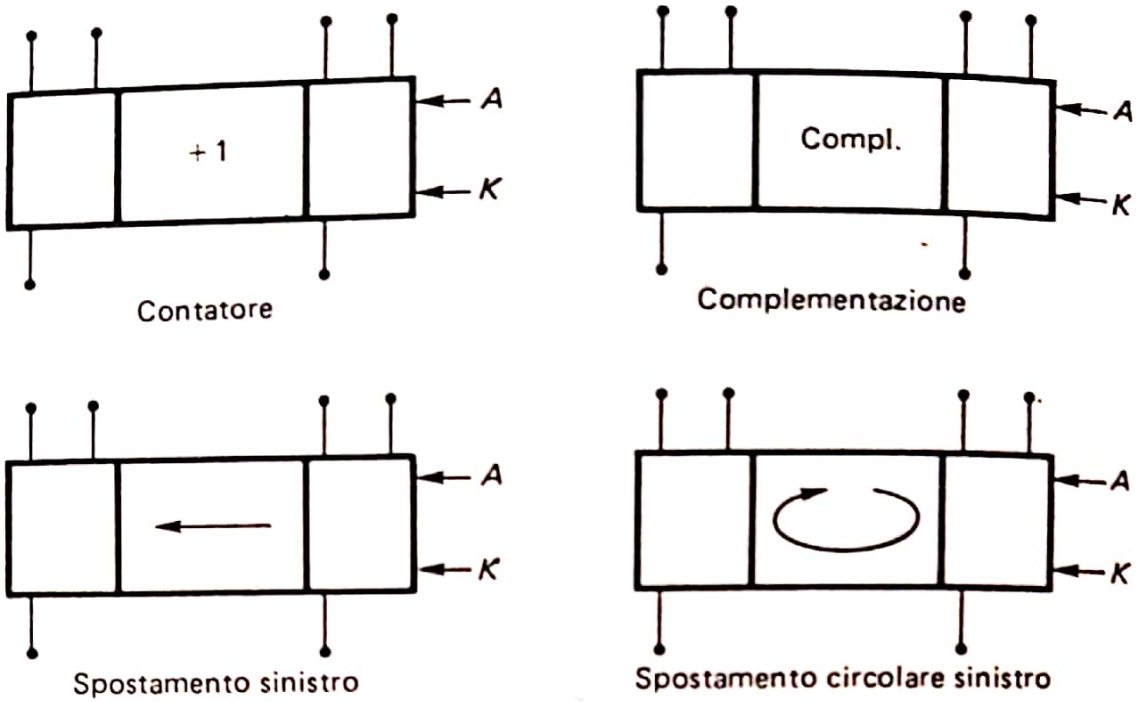


Figura 6.7
Registro a spostamento dotato di ingressi dall'esterno.



$A = 0$: mantiene inalterato il contenuto

$A = 1$: { $K = 0$ carica dall'esterno; $K = 1$ esegue la funzione (conta, complemento, spostamento)

Figura 6.8
Simboli di registri e significato dei comandi (similmente si indicano i contatori a passo diverso da + 1 e i registri a spostamento destro).

noti ancora una volta come il registro Y sia in grado di essere letto e caricato all'arrivo del medesimo impulso c.

I registri a spostamento permettono di effettuare operazioni di conversione parallelo/serie e serie/parallelo, che possono avere grande interesse se si vogliono collegare due parti di un sistema atte a elaborare rispettivamente in parallelo e in serie i bit di una parola. La figura 6.10 mostra il trasferimento in serie di una parola dal registro X al registro Y mediante un comando di spostamento sinistro e l'applicazione dei segnali $A_x = 1$, $A_y = 1$, mantenuti per la durata di n impulsi. E' questo un esempio di conversione parallelo/serie in quanto una parola può essere introdotta in parallelo nel registro X e poi estratta in serie attraverso l'uscita x_n. La conversione serie/parallelo si ottiene invece caricando in serie i bit di una parola nel registro X attraverso l'ingresso del flip-flop x_1, mediante n spostamenti successivi, e leggendola poi in parallelo alle uscite di X.

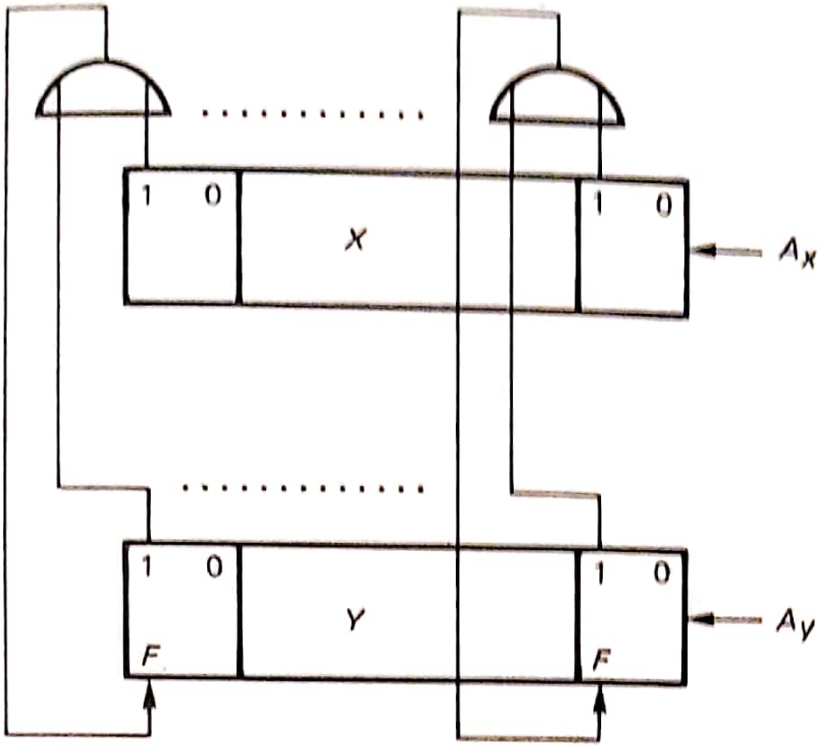


Figura 6.9
Connessioni per l'operazione logica X OR $Y \rightarrow Y$. Il caricamento di Y dall'esterno richiede un'estensione della rete logica agli ingressi di Y.

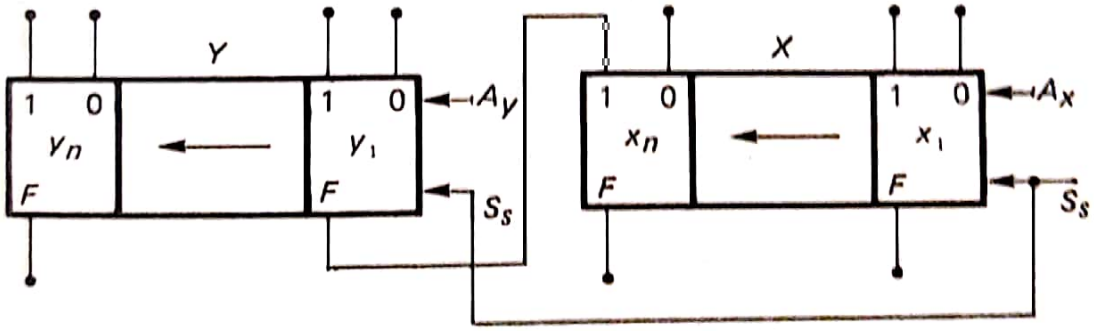


Figura 6.10
Trasferimento in serie: $X \rightarrow Y$.

Studiamo infine due esempi di progetto per registri che eseguono più funzioni. Il primo è un registro con un ingresso S, che deve realizzare lo spostamento destro per $S=0$ o sinistro per $S=1$. Affrontando il problema come sintesi di una rete sequenziale, limitata inizialmente al caso di tre bit, si ottiene immediatamente la seguente tabella delle

transizioni:

$y_3y_2y_1$ \ S	0	1
000	000	000
001	000	010
011	001	110
010	001	100
100	010	000
101	010	010
111	011	110
110	011	100

Dalla tabella, riordinata in forma di una mappa di Karnaugh, si costruiscono le espressioni algebriche per y'_3, y'_2, y'_1:

Sy_3 \ y_2y_1	00	01	11	10
00	000	000	001	001
01	010	010	011	011
11	000	010	110	100
10	000	010	110	100

$$y'_3 = y_2 S,$$
$$y'_2 = y_3 \overline{S} + y_1 S,$$
$$y'_1 = y_2 \overline{S}.$$

E' facile ora estendere le funzioni trovate al caso di n bit, per cui risulta:

$$y'_n = y_{n-1} S,$$
$$y'_i = y_{i+1} \overline{S} + y_{i-1} S, \qquad \text{con } 2 \leqslant i \leqslant n-1,$$
$$y'_1 = y_2 \overline{S}.$$

Nella figura 6.11 sono mostrati i collegamenti di ingresso dell'i-esimo flip-flop. Notiamo nuovamente che la rete logica all'ingresso dei

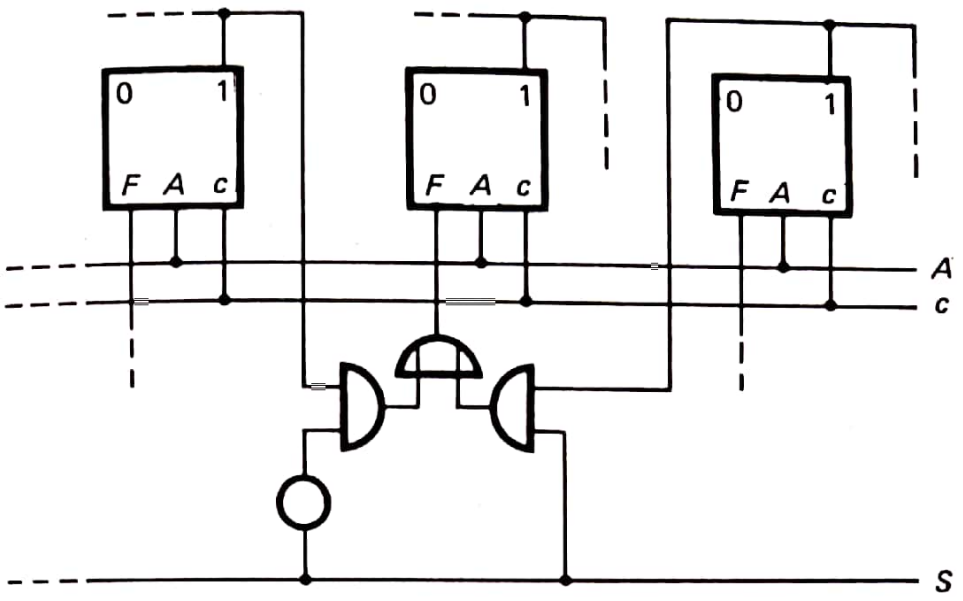


Figura 6.11
Particolare dei collegamenti del registro a spostamento destro e sinistro (non sono mostrati i collegamenti per il caricamento esterno).

flip-flop costituisce un selettore controllato da S, che fa passare y_{i+1} (per $S=0$) o y_{i-1} (per $S=1$).

Come secondo esempio costruiamo un contatore con un ingresso K, che conta da 0 a 5, "in avanti" se $K=0$, "indietro" se $K=1$. Il diagramma degli stati è indicato in figura 6.12: come abbiamo già visto nel paragrafo 6.1, i valori delle variabili di uscita, coincidenti con quelli dei segnali di anello $y_3y_2y_1$, sono associati agli stati e ne rappresentano una codifica.

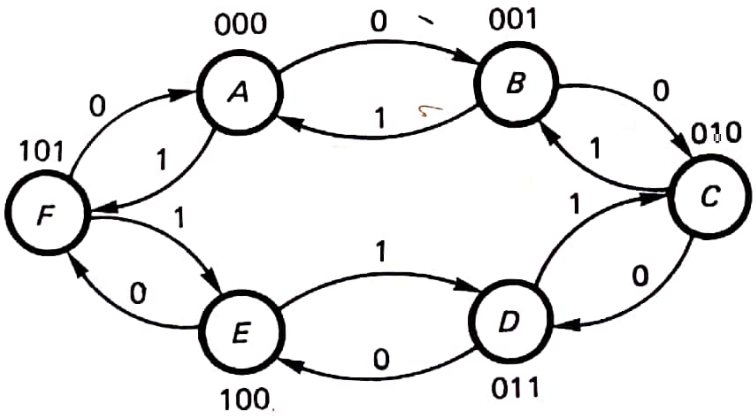


Figura 6.12
Diagramma degli stati per contare tra 0 e 5 in due direzioni.

Dalla tabella delle transizioni, tenendo conto delle condizioni di non specificazione, si ricavano le seguenti espressioni per y'_3, y'_2 e y'_1:

Ky_3 \ y_2y_1	00	01	11	10
00	001	010	100	011
01	101	000	–	–
11	011	100	–	–
10	101	000	010	001

$$y'_3 = \overline{K}y_3\overline{y}_1 + \overline{K}y_2y_1 + Ky_3y_1 + K\overline{y}_3\overline{y}_2\overline{y}_1,$$
$$y'_2 = \overline{K}y_2\overline{y}_1 + Ky_2y_1 + \overline{K}\overline{y}_3\overline{y}_2y_1 + Ky_3\overline{y}_1,$$
$$y'_1 = \overline{y}_1.$$

Il disegno della rete è lasciato al lettore.

6.3 La memoria ad accesso diretto

Il termine *memoria ad accesso diretto* (o semplicemente *memoria* se non vi sono ambiguità, o anche RAM, in gergo tecnico, per *Random Access Memory*) indica un'unità che memorizza un grande numero di parole in un insieme di flip-flop, opportunamente connessi, mediante un sistema di indirizzamento e trasferimento di parole.

I flip-flop della memoria possono dividersi in 2^k gruppi di h flip-flop, ciascuno dei quali memorizza una parola di h bit. Ogni gruppo è sostanzialmente un registro, qui detto *cella* di memoria, cui è logicamente associato un indirizzo di k bit nell'intervallo $0 \div 2^k - 1$. Il sistema di indirizzamento e trasferimento comprende un registro di k bit detto MAR (per *Memory Address Register*), e un registro di h bit detto MBR (per *Memory Buffer Register*), collegati con l'esterno; e un decodificatore a k ingressi e 2^k uscite (fig. 6.13).

Le operazioni di base della memoria sono la *lettura* e la *scrittura* di una parola. Esse si riferiscono alla cella di memoria il cui indirizzo è specificato in MAR, detta *cella selezionata*, e consistono nelle azioni seguenti:

1) lettura: al comando $L=1$ il contenuto della cella selezionata è trasferito nel registro MBR;

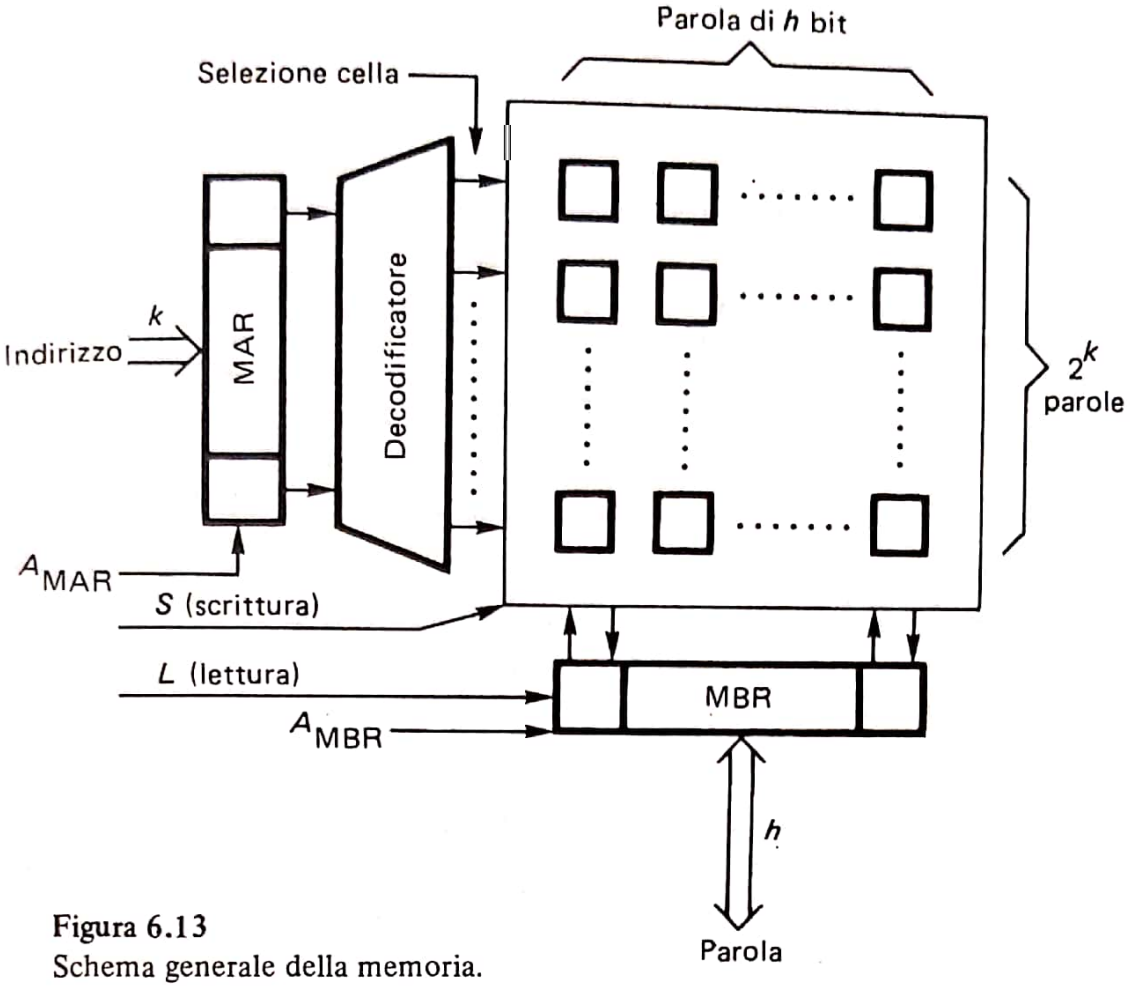


Figura 6.13
Schema generale della memoria.

2) scrittura: al comando $S=1$ il contenuto del registro **MBR** è trasferito nella cella selezionata.

L'accesso alla cella selezionata è ottenuto mediante il decodificatore, che attiva la corrispondente linea di selezione cella. Il trasferimento di una parola da una cella verso MBR (lettura), e da MBR verso una cella (scrittura), avviene in parallelo, secondo lo schema della figura 6.14, che mostra una possibile realizzazione della rete. Si noti la presenza del selettore, controllato dalla variabile L, all'ingresso di MBR, che consente di caricare tale registro con la parola letta dalla memoria ($L=1$), o con una parola E esterna ($L=0, A_{MBR}=1$).

I flip-flop della memoria non hanno ingresso impulsivo c (il loro schema logico si ottiene immediatamente da quello di fig. 4.6, sostituendo il segnale A a c), poiché il disaccoppiamento tra scrittura e lettura è garantito dal fatto che i segnali S e L non hanno mai contemporaneamente valore 1. Ammetteremo in genere che la scrittura in memoria avvenga in un intervallo di tempo inferiore alla distanza tra due impulsi.

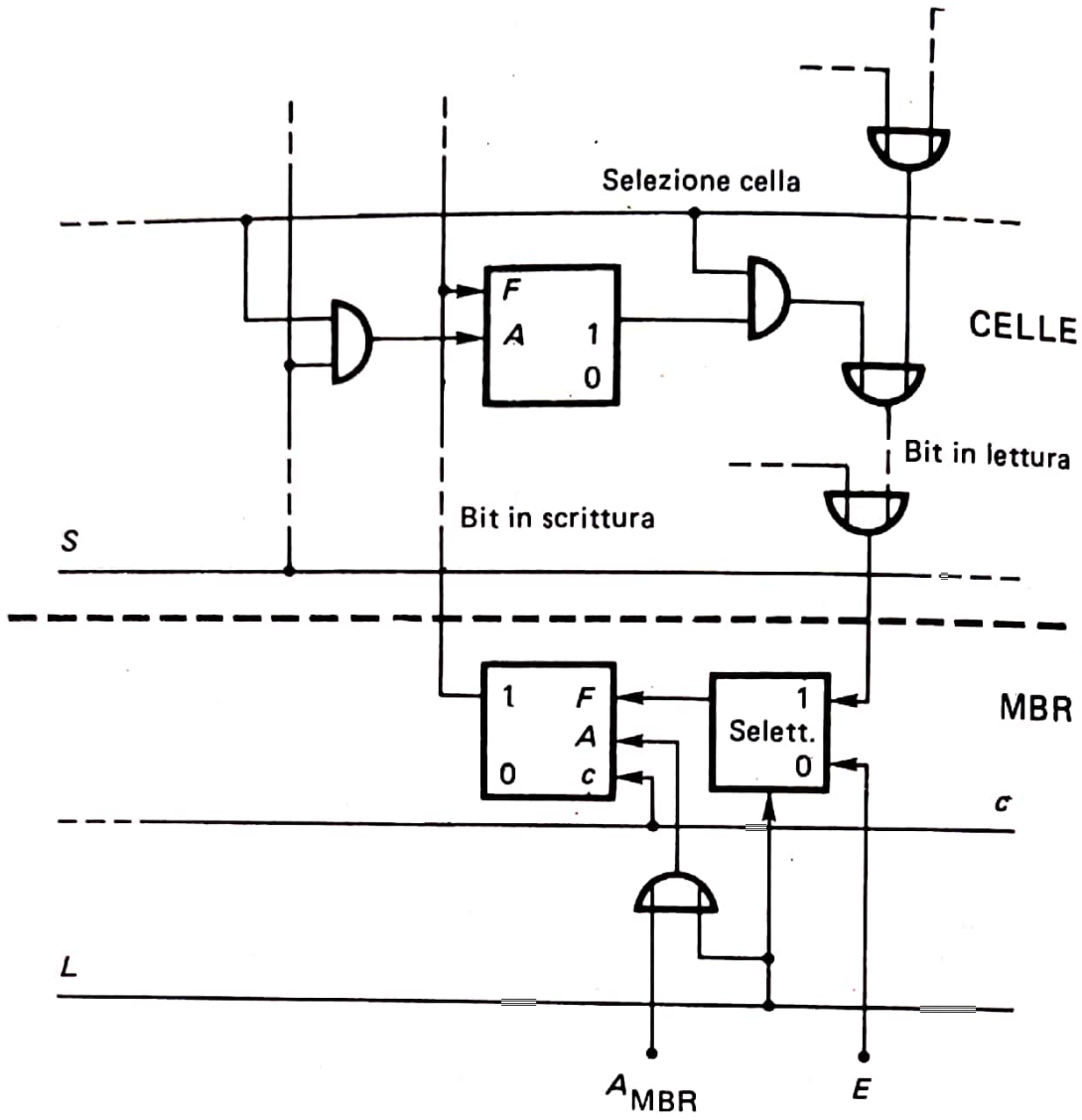


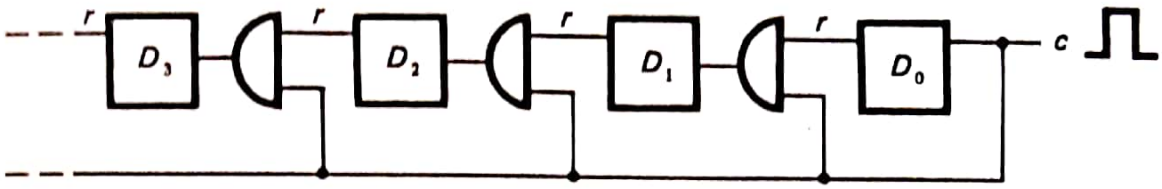
Figura 6.14
Dettaglio del collegamento tra MBR e celle di memoria. Il selettore comanda il trasferimento in MBR della parola di memoria (per $L=1$), o della parola esterna E (per $L=0, A_{MBR}=1$).

La figura 6.13 mostra in realtà lo schema di principio della memoria. Questa, per ragioni costruttive, è poi realizzata secondo una disposizione geometrica dei flip-flop lievemente diversa, come vedremo nel capitolo 8 discutendo di circuiti integrati.

Esercizi

6.1 Indicare lo schema dei collegamenti per un registro contatore e un registro a complementazione, dotati di caricamento esterno.

6.2 Un *contatore decimale* produce una conta in numerazione a base 10, rappresentando in codice binario naturale ogni singola cifra decimale. Tale contatore è diviso in *decadi* D_i connesse secondo lo schema seguente:



Ogni decade, composta di quattro flip-flop a_3, a_2, a_1, a_0, rappresenta in sequenza le cifre decimali da 0 a 9, e nella successiva transizione da 9 a 0 genera un riporto r che abilita il passaggio dell'impulso di conta alla decade successiva.

Progettare una decade, cioè trovare la rete sequenziale con registro di quattro flip-flop Fc che realizzi la conta e produca il riporto.

6.3 Un codice *Gray* ha la proprietà che per rappresentare numeri consecutivi usa configurazioni che differiscono per un solo bit. Progettare un contatore che utilizza il codice Gray:

conta	a_2	a_1	a_0
0	0	0	0
1	0	0	1
2	0	1	1
3	0	1	0
4	1	1	0
5	1	1	1
6	1	0	1
7	1	0	0
0	0	0	0

6.4 Progettare un registro a spostamento circolare sinistro con un ingresso X, che sposti il contenuto di ciascun flip-flop di una posizione (per $X = 0$) o di due posizioni (per $X = 1$).