# Parte III:
# Analisi e sintesi di macchine sequenziali

## 3.1 Introduzione

Una macchina sequenziale  un sistema nel quale, detto I(t) l'insieme degli ingressi in t, O(t) l'insieme delle uscite in t, ed M(t) una funzione di I(t-1), I(t-2)...(i=1,..n) detta memoria, si ha:

$$o_i(t)=F(I(t),M(t)) \ , \quad o_i \in O$$

ovvero, le uscite in ogni istante dipendono non solo dai valori degli ingressi nello stesso istante, ma anche dalla "storia" del sistema, rappresentata dalla memoria M.

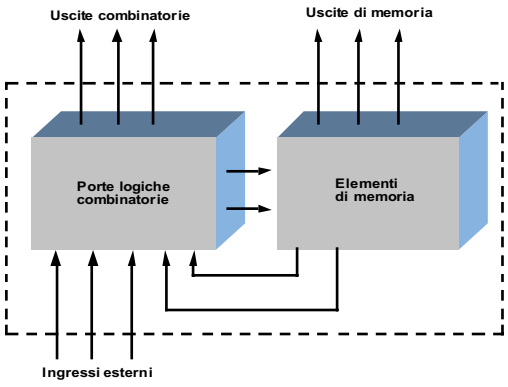



Fig. 3.1 *Schema generico di una macchina sequenziale*

Un esempio di sistema sequenziale  il seguente: si supponga di dover progettare un circuito che, se riceve in ingresso uno 0, produce in uscita uno 0, ma se riceve un 1, produce in uscita un 1 solo se l'input precedentemente ricevuto era un 1.
Il funzionamento della macchina pu essere rappresentato come segue:

| I(t) | I(t-1) | O(t) |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

Si vede immediatamente dall'esempio che O(t)  una funzione combinatoria AND fra l'ingresso I(t) e la memoria dell'ingresso precedente, M(t)=I(t-1).
Prima di affrontare aspetti di implementazione (ad esempio, come realizzare elementi di memoria), studieremo un modello di rappresentazione pi efficace delle relazioni ingresso-memoria-uscita in una macchina sequenziale.
Questo modello, detto degli **automi a stati finiti**, consente di progettare agevolmente macchine alquanto complesse.

Nel seguito, la teoria degli automi  descritta in maniera alquanto parziale, limitatamente alla necessit di acquisire strumenti necessari per gli obiettivi di questo corso. Lo studente avr modo di approfondire l argomento in altri corsi.

## 3.2 Automi a Stati Finiti

Un automa e' il modello matematico di un sistema con input ed output discreti. Il sistema puo' trovarsi in una fra n configurazioni diverse, dette **STATI**.
Lo stato rappresenta una condizione in cui il sistema si trova, e sommarizza la storia precedente del sistema in termini di input ricevuti. Ad esempio, per un sistema che deve riconoscere stringhe contenenti la sequenza '001'una condizione possibile e' quella in cui gli ultimi due bit ricevuti sono uguali a zero. Un tale sistema ha bisogno di almeno due bit di memoria.
Nell'esempio del paragrafo precedente, gli stati sono due: quello in cui M=I(t-1)=0, e quello in cui M=I(t-1)=0.
In un automa a stati finiti, la conoscenza dello stato in cui si trova il sistema e' necessaria per determinare il comportamento del sistema a fronte di input successivi.
A fronte di un nuovo input, il sistema transita in un nuovo stato, e questo e' rappresentato graficamente nel modello con un arco orientato dallo stato di partenza a quello di arrivo, etichettato con il valore dell'input.

La definizione formale di automa a stati finiti e' la seguente:
DEF Un *automa a stati finiti* una quintupla $(Q,\Sigma,\delta,q_0, F)$ dove Q e' un insieme finito di stati, $\Sigma$ e' un alfabeto finito di simboli, $q_0$ e' lo statoiniziale, $F \subseteq Q$ e' il set di stati finali, e d e' la funzione di transizione $Q\times\Sigma \dashrightarrow Q$ ( $Q\times\Sigma$ e' il prodotto cartesiano, ovvero l'insieme delle coppie q,a ); $\delta(q,a)$ rappresenta uno **stato** raggiunto dall'automa, per ogni ogni stato di partenza q e simbolo di ingresso a.

La definizione appena fornita in realt si riferisce ad un sottoinsieme di automi a stati finiti, gli automi *deterministici*. Per questa classe di automi, la funzione $\delta$ (q,a) ha un unico valore, cio , non ambigua.

**Simbologia adottata**:

1) se Q e' un insieme di **stati**, $q_i$ ($Q_i$)ed $s_j$ ($S_j$)indicano stati, $q_0$ (s0) indica uno stato iniziale.
2) $\Sigma$ e' l'alfabeto di ingresso, $\alpha$ e $\beta$ indicano simboli dell'alfabeto (nel caso di sistemi binari, $\Sigma=\{01\}$
3) $\delta$ indica una funzione di transizione $\delta : Q\times\Sigma\rightarrow Q$
4) F e' il set degli stati **finali**, o di **accettazione** $F \subset Q$
Il comportamento di un automa pu essere rappresentato mediante un **grafo** in cui gli archi rappresentano le transizioni fra stati, i cerchi rappresentano stati. Gli stati finali sono evidenziati da cerchi doppi. I simboli sugli archi sono gli input del sistema, ed appartengono all'alfabeto $\Sigma$.

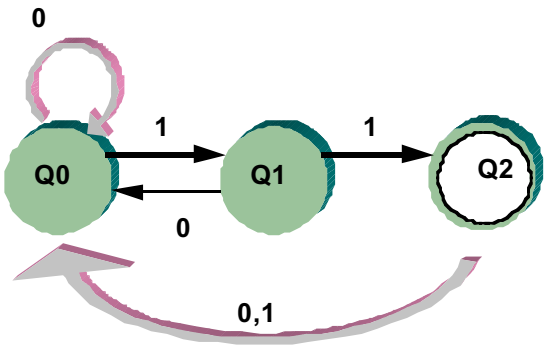



**Fig. 3.2** *Automa a stati finiti*

Alternativamente, un automa si pu rappresentare mediante una **tabella delle transizioni, o stati futuri:**

| **stato di partenza** | **stato di arrivo per input=0** | **stato di arrivo per input=1** |
|---|---|---|
| q0 | q0 | q1 |
| q1 | q0 | q3 |
| q2 | q0 | q0 |

## 3.3 Automi a stati finiti con output

Il comportamento di un circuito sequenziale pu essere modellato mediante un particolare tipo di automi a stati finiti : gli automi deterministici con output. Questi automi vengono chiamati macchine di *Moore* o di *Mealy.*, a seconda che l'output sia associato agli stati, o alle transizioni fra stati.

### 3.3.1 Macchine di Moore

*DEF* Una macchina di Moore una sestupla $(Q,\Sigma,\Delta,\delta,\lambda,q0)$ dove $\Delta$ un alfabeto di output, e $\lambda$ una funzione di transizione $\lambda : Q \rightarrow \Delta$, che associa un simbolo di output ad ogni stato. Per ogni stato, $\lambda(qi)=aj$, $aj \in \Delta$.
Un automa deterministico a stati finiti pu essere visto come un caso speciale di macchina di Moore dove $\Delta=(0,1)$ e $\lambda(qi)=1$ se $qi \in F$.
Notare che nelle macchine con output non occorre una distinzione fra stati di accettazione e non.

**Esempio**

Consideriamo un automa che produce in uscita il valore del resto di un intero i, in binario, modulo 4.
I possibili valori del resto sono 0,1,2 e 3. Osserviamo che se gli ultimi due bit ricevuti sono 00, allora il resto 0, 01->1, 10->2, 11->3.

Perci abbiamo 4 stati, che possiamo identificare con i valori dell'output: 0,1,2, e 3.
Abbiamo la seguente tabella degli stati futuri $\delta/\lambda$:

| stato\input | 0 | 1 |
|---|---|---|
| q0 | q0/0 | q1/1 |
| q1 | q2/2 | q3/3 |
| q2 | q0/0 | q1/1 |
| q3 | q2/2 | q3/3 |

In questo caso, ogni cella (i,j) della tabella fornisce *lo stato di arrivo* a partire dallo stato i e a fronte dell'input j, ed anche l'*output* prodotto.

### 3.3.2 Macchine di Mealy

*DEF* Una macchina di Mealy una sestupla $(Q,\Sigma,\Delta,\delta,\lambda,q0)$ , dove $\lambda$ un mapping da $Q\times\Sigma \to \Delta$, ovvero $\lambda(qi,bk)=aj$, $bk\in\Sigma$, $aj\in\Delta$.

**Equivalenza fra macchine di Moore e Mealy**

*Teorema*. Se M1= $(Q,\Sigma,\Delta,\delta,\lambda,q0)$ una macchina di Moore, allora esiste una macchina di Mealy equivalente M2.
*Dimostrazione*. sia M2=$(Q,\Sigma,\Delta,\delta,\lambda',q0)$ e definiamo:
$\lambda'(q,a)=\lambda(\delta(q,a))$

Allora, M2 equivalente a M1 e segue le stesse transizioni, emettendo ad ogni transizione l'output associato allo stato di arrivo in M1.

*Teorema*. Se M1= $(Q,\Sigma,\Delta,\delta,\lambda,q0)$ una macchina di Mealy, allora esiste una macchina di Moore equivalente M2.

*Dimostrazione.* sia M2=$(Q\times\Delta,\Sigma,\Delta,\delta',\lambda',(q0\ b0))$, dove b0 un qualsiasi carattere di $\Delta$. Gli stati M2 sono coppie rappresentate da stati di M1 e simboli di $\Delta$. Definiamo
$\delta'((q, b),a) = (\delta(q,a), \lambda(q,a))$ e $\lambda'((q,b))=b$
I due automi sono equivalenti, infatti le transizioni di M2 da uno stato all'altro sono determinate solo dal primo elemento della coppia che identifica lo stato, e dal valore dell'input. Ovvero, da uno stato (q, b), quando si riceve il simbolo **a,** si transita in uno stato (q', c) il cui primo elemento rappresenta lo stato in cui transita M1 quando da q riceve **a** ed il cui secondo elemento rappresenta l'output che , nella macchina di Mealy, avrebbe assunto l'output transitando in quello stato dallo stato q a fronte di un certo input **a.**

## 3.4 Minimizzazione di automi.

Dato un automa M, possiamo notare che alcuni stati si comportano nello stesso modo per tutte le stringhe di ingresso. Consideriamo per esempio l'automa in figura 3.3.a.

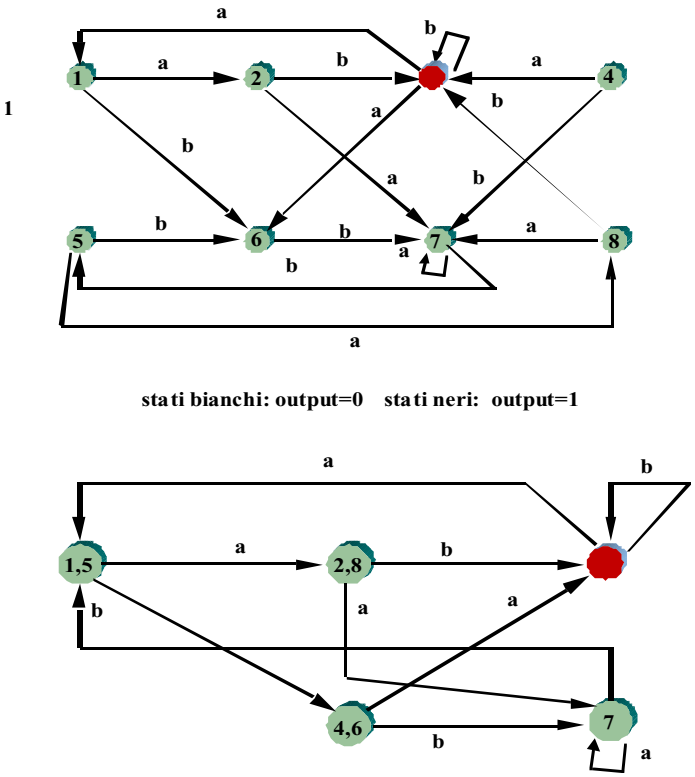



**Figure 3.3.a** *(automa non minimizzato)* **e 3.3.b** *(automa minimizzato)*

L'automa in figura 3.3.a una macchina di Moore. L'output di tutti gli stati si suppone uguale a zero, mentre l'output dello stato 3 (in rosso) uguale ad 1.
Osserviamo l'automa: se siamo negli stati 2 o 8, a fronte dell' input **a** si transita in 7, a fronte dell'input **b** si transita in 3. In un certo senso, potremmo dire che la presenza di 2 e 8 in M ridondante, perch i due stati effettuano le stesse transizioni a fronte degli stessi input.
Poich , come vedremo, un automa un modello astratto di una macchina sequenziale, intuitivo il fatto che sia conveniente minimizzare un automa, ovvero trovare un automa equivalente che abbia il minimo numero di stati. Ridurre il numero di stati infatti equivale a ridurre il numero di componenti di memoria nel circuito corrispondente.
In pratica, quello che si fa eliminare stati del tipo di quelli nell'esempio citato.

*DEF* Sia dato un automa di Moore M (Q,Σ,Δ,δ,λ,q0). Due stati p e q si dicono **distinguibili** in una macchina di Moore se gli output associati a p e q sono diversi, o se per qualche sequenza di simboli a1a2..an ricevuti a partire da p e q, si transita in due stati p' e q' caratterizzati da output diversi.

Se tutte le possibili coppie p q in M sono distinguibili, allora M non pu essere minimizzato, ma se viceversa M contiene collezioni di stati **non** distinguibili, allora possiamo eliminare le ridondanze prelevando un solo stato per ogni gruppo di stati non distinguibili, ed eliminando gli altri.

Il procedimento per l'individuazione delle **classi di indistinguibilit** di una macchina di Moore il seguente (gli esempi si riferiscono alla figura 3.3.a):

1) Si traccia, a partire dall'automa o dalla sua tabella degli stati futuri, una tabella triangolare che permetta, ai suoi incroci, di indicare il risultato del confronto di ogni possibile coppia di stati.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| q2 | | | | | | | |
| q3 | | | | | | | |
| q4 | | | | | | | |
| q5 | | | | | | | |
| q6 | | | | | | | |
| q7 | | | | | | | |
| q8 | | | | | | | |
| | q1 | q2 | q3 | q4 | q5 | q6 | q7 |

2) Si esaminano una dopo l'altra tutte le possibili coppie di righe della tabella degli stati futuri, inserendo nel corrispondente incrocio della tabella triangolare:

-una X se in almeno una colonna risultano specificate uscite diverse

- la denominazione della coppia di stati futuri individuati colonna per colonna se in tutte le colonne le uscite risultano uguali.

- non si scrive nulla nel caso in cui le indicazioni di stato futuro siano identiche o coincidano con la denominazione della coppia di stati presa in esame

Osservazione:

- ¥ Le caselle con X individuano stati <u>distinguibili</u> sulla base di un solo simbolo di ingresso, oppure del simbolo di uscita. Ad esempio, in figura 3.3.a, gli stati 4 ed 8 possono subito essere marcati con una X, poich a fronte del simbolo **a** transitano rispettivamente negli stati 3 e 7, che hanno output diversi. Inoltre, tutte le coppie $(q_i,q_3)$ i≠ 3 sono distinguibili sulla base del simbolo di uscita.
- Le caselle con indicazione di una o pi coppie di stati futuri indicano stati <u>indistinguibili per sequenze di ingresso di lunghezza 1</u>. Per questi stati, la decisione deve essere rimandata. Ad esempio, gli stati 1 e 5 a fronte del simbolo **b** transitano nello stesso stato 6, ma a fronte del simbolo **a**, transitano in 2 e 8, rispettivamente. Gli stati 2 e 8 hanno lo stesso output, ma in una prima fase non sappiamo se sono fra loro distinguibili o no, perci la decisione va rimandata.
- ¥ le caselle vuote viceversa individuano stati <u>indistinguibili</u> per sequenze di ingresso di lunghezza qualsiasi. Ad esempio, gli stati 4 e 6 a fronte del simbolo **a** transitano in 3, ed a fronte del simbolo **b**, transitano in 7, dunque si comportano esattamente nello stesso modo. Analogamente gli stati 2 ed 8.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| q2 | X | | | | | | |
| q3 | X | X | | | | | |
| q4 | X | X | X | | | | |
| q5 | (2,8) | X | X | X | | | |
| q6 | X | X | X | | X | | |
| q7 | (6,5)(2,7) | X | X | X | (8,7) (6,5) | X | |
| q8 | X | | X | X | X | X | X |
| | q1 | q2 | q3 | q4 | q5 | q6 | q7 |

3) Esaminando ordinatamente le celle della tabella, si continua ad operare finch possibile, sbarrando via via le caselle contenenti indicazioni di stati futuri che corrispondono a caselle gi precedentemente sbarrate, o cancellando caselle che corrispondono a caselle vuote. Ad esempio, una volta marcata come "bianca" la casella (2,8), possiamo "cancellare" anche la casella (1,5) che "puntava" alla coppia 2,8. Viceversa, una volta marcati come distinguibili gli stati 2 e 7 (a fronte del simbolo **b** transitano in stati con output diversi) si potr sbarrare anche, ad esempio, la cella (1,7) che conteneva un puntatore alla coppia 2,7. Il procedimento sia arresta quando tutte le caselle sono vuote o sbarrate.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| q2 | X | | | | | | |
| q3 | X | X | | | | | |
| q4 | X | X | X | | | | |
| q5 | | X | X | X | | | |
| q6 | X | X | X | | X | | |
| q7 | X | X | X | X | X | X | |
| q8 | X | | X | X | X | X | X |
| | q1 | q2 | q3 | q4 | q5 | q6 | q7 |

4) Procedendo da destra verso sinistra si esaminano una dopo l'altra le colonne della tabella triangolare contenente caselle **non sbarrate** e si costruisce un corrispondente sottoinsieme S con la denominazione della colonna stessa e delle righe relative. Si controllano via via i sottoinsiemi che risultano contenuti in sottoinsiemi individuati. Nell'esempio, i sottoinsiemi sono: colonna 1: (1,5), colonna 2: (2,8), colonna 4: (4,6). Questi sottoinsiemi prendono il nome di **classi di indistinguibilit** .
5) Si aggiungono tanti sottoinsiemi con un unico elemento quanti sono gli stati dell'automa che non compaiono in blocchi delle partizioni trovate al punto precedente. Nell'esempio, le ulteriori classi di indistinguibilit sono (3) e (7). Si costruisce la tabella degli stati futuri minima copiando solo le righe della tabella di partenza che corrispondono al primo stato di ciascuna classe di indistinguibilit , e correggendo di conseguenza le indicazioni dello stato futuro.

| **stato di partenza** | **stato futuro e output se i=a** | **stato futuro e output se i=b** |
|---|---|---|
| 1' = (1,5) | 2'/0 | 4' /0 |
| 2 '= (2,8) | 7'/0 | 3'/1 |
| 3 '= (3) | 1'/0 | 3'/1 |
| 4' = (4,6) | 3'/1 | 7'/0 |
| 7' = (7) | 7'/0 | 1'/0 |

La figura 3.2.b rappresenta graficamente l'automa minimizzato.
L'algoritmo di minimizzazione pu essere espresso sinteticamente, nel caso di output binario, come segue:

Sia A l'insieme degli stati con output=1 e Q-A l'insieme di stati con output =0.
**begin**

1) **for** p in A e q in Q-A, **marca** la cella (p,q)
2) **for** ogni cella (p,q) in AxA o (Q-A)x(Q-A) **do**
3) **if** per qualche simbolo di ingresso **a** (δ(q,**a**) , δ(p,**a**) ) sono marcati, **then**
**begin**
4) **marca** (p, q)
5) **marca** ricorsivamente tutte le celle non ancora marcate sulla lista delle coppie che puntano a (p, q)
**end**
**else** /*nessuna cella (δ(q,**a**) δ(p,**a**) ) risulta marcata
6) **for** tutti i simboli **a** di Σ **do**
7) scrivi (p , q) nella lista di celle che puntano a (δ(q,**a**) δ(p,**a**) ), tranne nel caso in cui δ(q,**a**)= δ(p,**a**)
**end**

## 3.5 Circuiti di memorizzazione elementari: i Flip Flop

*(nota: questa parte degli appunti va integrata con i par. 5.1 e 5.2 del Fummi et al.)*

Ora che abbiamo a disposizione un modello formale per l'analisi e sintesi di circuiti sequenziali, possiamo studiare circuiti elementari e complessi.
Il circuito sequenziale basilare il Flip Flop, un circuito in grado di memorizzare un bit di informazione. Il Flip Flop prende anche il nome di *bistabile*, poich pu trovarsi in due stati stabili: lo stato in cui ha memorizzato un 1 e quello in cui ha memorizzato uno 0.

Il primo tipo di FF che analizziamo il FF di tipo Set Reset.
In Figura 3.4 mostrato il simbolo grafico, lo schema circuitale, l'automa e la tabella di verit di un FF di tipo Set Reset realizzato con porte NAND.
La capacit del circuito di memorizzare informazioni data dall'elemento di controreazione delle porte NAND.
Innanzitutto osserviamo che, per descrivere la reazione del circuito ad una coppia di valori di input S,R occorre conoscere i valori delle uscite precedentemente memorizzati, Q(t-1) e $\overline{Q}$(t-1). Poich questi valori sono, per definizione, l'uno il negato dell'altro, il FF una macchina sequenziale a due stati: chiamiamo Q0 lo stato in cui memorizza uno 0 e Q1 lo stato in cui memorizza un 1 . L'informazione memorizzata la stessa disponibile in uscita sull'output Q.
Analizziamo ora i casi possibili, ricordando che le porte NAND transitano al valore 0 solo quando entrambi gli ingressi sono 1.

1) Q(t-1)=0 $\overline{Q}$(t-1)=1, S(t)=1 R(t)=1
   In questo caso la porta NAND 1 resta a 0 e la porta NAND 2 resta ad uno, dunque la situazione resta immutata (il FF non commuta)

2) Q(t-1)=1 $\overline{Q}$(t-1)=0, S(t)=1 R(t)=1
   In questo caso la porta NAND 1 resta a 1 e la porta NAND 2 resta ad 0, dunque ancora la situazione resta immutata.

*nota: L'input (1,1) lascia il FF nello stesso stato.*

3) Q(t-1)=0 $\overline{Q}$(t-1)=1, S(t)=0 R(t)=1
   In questo caso la porta NAND 1 passa a 1 e la porta NAND 2 va a 0, dunque il FF commuta

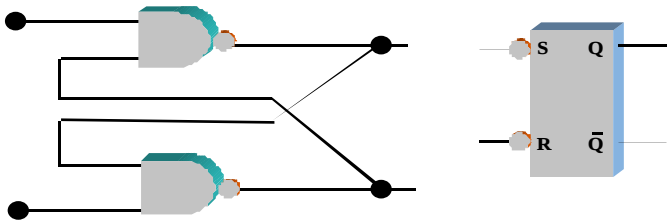
4) Q(t-1)=1 $\overline{Q}$(t-1)=0, S(t)=0 R(t)=1
   In questo caso la porta NAND 1 resta a 1 e la porta NAND 2 resta ad 0, dunque la situazione resta immutata.

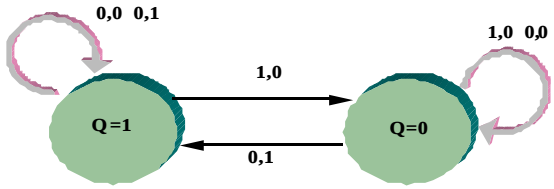
*nota: L'input (0,1), indipendentemente dallo stato di partenza, lascia in FF nello stato Q1.*

5) Q(t-1)=0 $\overline{Q}$(t-1)=1, S(t)=1 R(t)=0
   In questo caso la porta NAND 1 resta a 0 e la porta NAND 2 resta ad 1, dunque il FF non commuta

6) Q(t-1)=1 $\overline{Q}$(t-1)=0, S(t)=1 R(t)=0
   In questo caso la porta NAND 1 va a 0 e la porta NAND 2 ad 1, dunque il FF commuta.

*nota: L'input (1,0) lascia o fa transitare il FF nello stato Q0.*




| Set(t) | Reset(t) | Uscita Q(t) |
|---|---|---|
| 1 | 1 | Q(t-1) (immutata) |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 0 | non ammessa |

Fig. 3.4 *Schema di funzionamento di un Flip-Flop di tipo S-R realizzato con porte nand*

7) L'ultimo caso da analizzare quello agli ingressi di Set e Reset vengono applicati contemporaneamente due impulsi bassi.
In questo caso, quale che sia la situazione iniziale, entrambe le uscite andranno inizialmente ad uno, il che non corrispondente alle specifiche. Inoltre, poich non possibile garantire che S e R tornino contemporaneamente nella condizione di riposo (S=R=1 nel SR nand), il FF, anzich conservare l informazione memorizzata durante la fase attiva, inizier una serie di oscillazioni.

*Si vedano sul Fummi altri tipi di FF:*

*FF SR con porte NOR (Latch NOR)*
*FF JK*
*FF D (Delay)*

Un ulteriore tipo di FF il Toggle

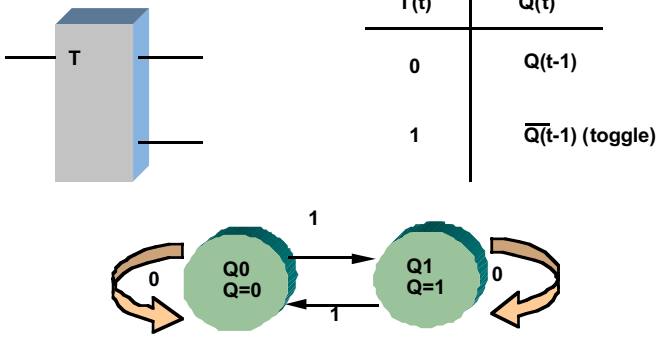



Fig. 3.5 *Schema di funzionamento del Flip-Flop di tipo T*

Come si vede, il Toggle commuta a fronte di un ingresso uguale ad 1, mentre resta nello stato di partenza a fronte di un input basso.

**Si noti che i FF di tipo D e T possono essere ottenuti da FF di tipo JK**, come mostrato in figura.

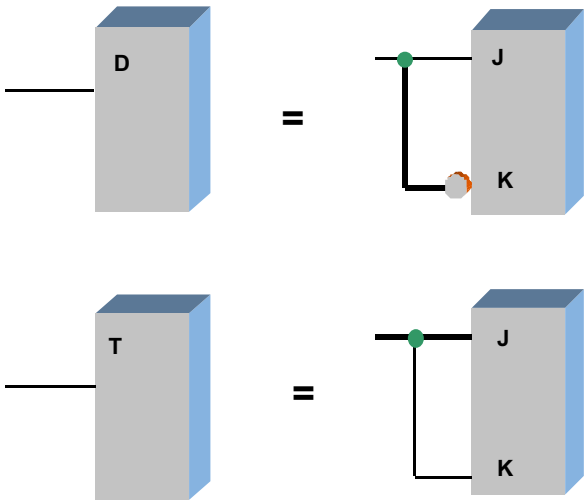



Fig. 3.6 *Schema di equivalenza tra Flip-Flop di tipo D, T e J-K*

### 3.5.1 Diagrammi temporali e sincronizzazione.

Per rappresentare un FF abbiamo a disposizione, come visto: lo schema circuitale (porte logiche con controreazione), il simbolo grafico, l'automa, la tabella degli stati. Esiste anche un'altra modalit di analisi, che il *diagramma temporale*.
Il diagramma temporale consente di rappresentare le commutazioni delle uscite di un FF, o di un circuito pi complesso, in funzione dell'andamento temporale ingressi.
Ad esempio, si osservi il diagramma della figura pi sotto. Le commutazioni dei segnali di ingresso non sempre hanno effetto sull'uscita, ad esempio se Q(t) gi "alto" una transizione di J da 0 a 1 non ha effetto su Q. Si noti inoltre che il diagramma temporale descrive un comportamento "ideale", in cui le transizioni sono istantanee (anzich , come nel caso reale, in pendenza) e l'effetto di un cambiamento di input si riflette istantaneamente sull'output (anzich con un certo ritardo).
In figura, sono indicate con delle frecce le commutazioni di J o K che hanno effetto su Q.

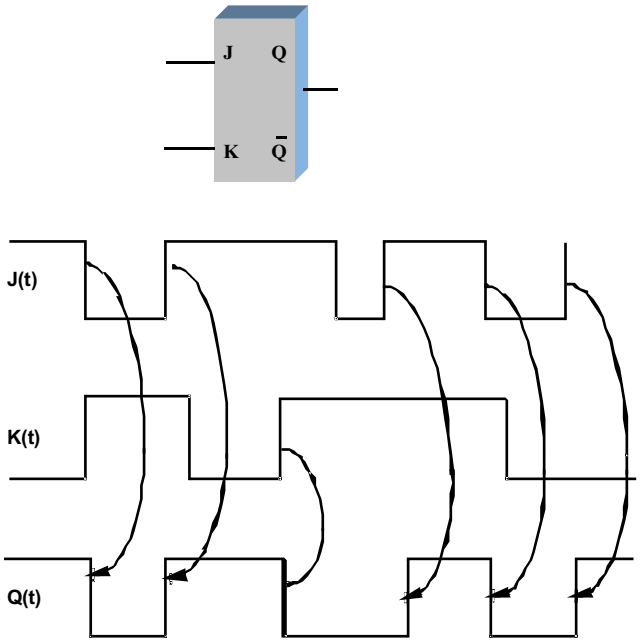



Fig. 3.7 *Schema di funzionamento di un Flip-Flop di tipo J-K*

I sistemi sequenziali possono operare in modo **sincrono** o **asincrono**.
Nei sistemi *asincroni* (come quello dell'esempio sopra) i circuiti cambiano il valore delle uscite ogni volta che uno o pi ingressi cambiano.
Nei sistemi *sincroni* (o *cadenzati)*, l'istante esatto in cui una qualsiasi uscita pu cambiare determinato da un segnale di "cadenza" detto **clock**. Un clock un treno di impulsi ad onda quadra. Un sistema sincrono pu essere sensibile alle transizioni (o fronti) positivi o di salita (0→1) del clock, oppure ai fronti negativi o di discesa (1→0).
Nei FF sincroni, oltre agli input gi descritti, esiste un input detto clock (CK). Un simbolo circolare (inverter) sull'input di clock indica che il FF sensibile ai fronti di discesa, altrimenti le commutazioni avvengono sui fronti di salita.
In figura mostrato un esempio di FF JK sincrono, o cadenzato. Si osservi che il comportamento del FF completamente diverso dall'esempio precedente bench i segnali J e K siano gli stessi dell'esempio precedente. In questo caso, i valori degli input vengono "letti" dal FF JK cadenzato solo in corrispondenza dei fronti di discesa del segnale di Clock. Una "X" sulla forma d'onda di J o K indica il punto di campionamento del segnale.

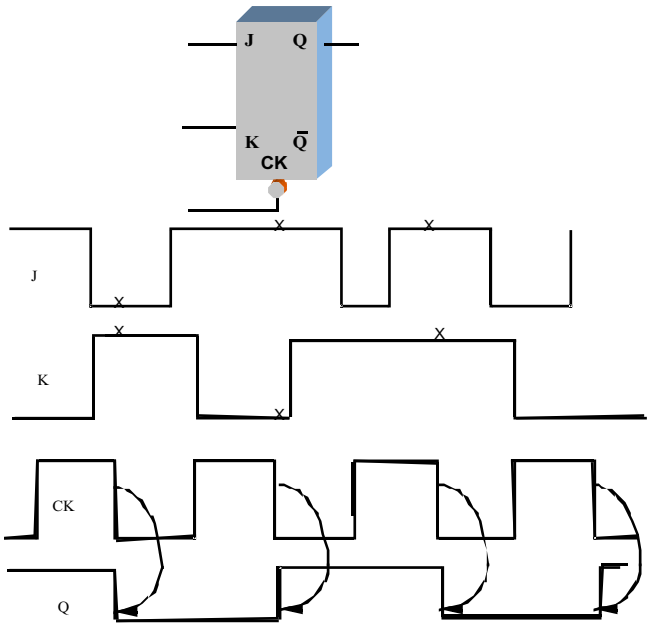



Fig. 3.8 *Schema di funzionamento di un Flip-Flop cadenzato*

In questo corso faremo riferimento solo a sistemi sincroni, o cadenzati.

## 3.6 Analisi di circuiti sequenziali

Nel paragrafo 1 di questo capitolo abbiamo descritto lo schema circuitale generale di un circuito sequenziale. In seguito, ci siamo dotati di uno strumento formale di analisi, gli automi a stati finiti. Quindi, abbiamo descritto gli elementi di memorizzazione che costituiscono la parte essenziale di un circuito sequenziale, i Flip Flop.
Siamo ora pronti per studiare il comportamento di circuiti sequenziali complessi.
Partiremo dal problema dell'analisi, ovvero:
*dato lo schema circuitale di circuito sequenziale, descriverne il funzionamento*.
Il funzionamento di un circuito sequenziale va descritto in termini di un automa a stati finiti.
Dato lo schema circuitale, dapprima dobbiamo identificare gli elementi di memoria che vi sono inclusi. In ogni istante, la memoria del sistema, ovvero il valore binario memorizzato nei FF, indica lo **stato** in cui il sistema si trova. Per ogni possibile stato e possibile combinazione degli input, da un esame della parte combinatoria del circuito possiamo determinare i valori delle uscite e il successivo stato in cui il sistema transiter .
Di seguito descriveremo la procedura generale, riferendoci per semplicit ad un esempio concreto, mostrato in figura.

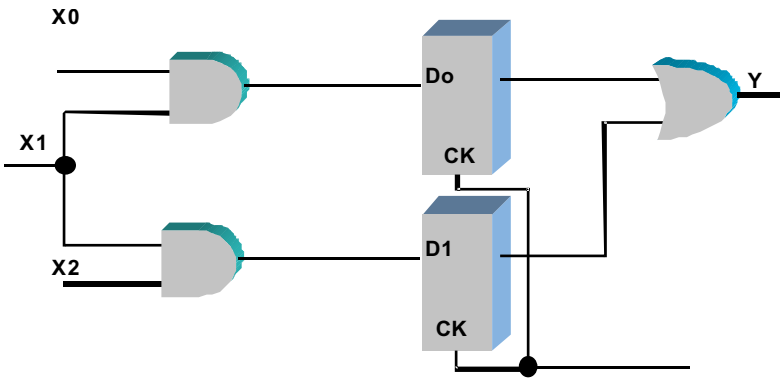



**Figura 3.9** *Esempio di circuito sequenziale*

### 3.6.1 Procedura di analisi di circuiti sequenziali

**1** Si esaminano gli elementi di memoria del circuito, ovvero i FF. I possibili **stati** del circuito sono rappresentati dalle possibili combinazioni di valori memorizzabili nei FF, e disponibili sulle uscite $Q_i$. Per n FF, avremo $2^n$ combinazioni, e $2^n$ stati.

**2** Si assegna un simbolo di stato ad ogni combinazione di memoria. Per l'esempio di figura 3.6.1, abbiamo 2 FF, e 4 possibili valori memorizzati su Q1 e Q0: 00,01,10,11. Possiamo ad esempio assegnare la seguente codifica:
S0→00, S1→ 01, S2 →10, S3 →11

**3** Si analizza la parte combinatoria del circuito e si ricavano le espressioni booleane per ciascun ingresso di ciascun FF contenuto nel circuito, dette anche funzioni di eccitazione dei FF, nonch le EB delle uscite.

Per l'esempio di figura 3.6.1 avremo le seguenti EB:
$D_0 = X_0X_1$
$D_1 = X_1X_2$
$Y = Q_0Q_1$

**4** Si traccia una tabella degli stati futuri, cos composta:

| possibili combinazioni degli input e degli stati **in t** | corrispondenti valori delle funzioni di eccitazione dei FF **in t** (nel caso di FF JK) | corrispondenti valori delle uscite **in t** | stati futuri, ovvero nuovi valori che saranno memorizzati **in t+1** |
|---|---|---|---|
| $Q_{n-1}...Q_0\ I_m...I_0$ | $J_{n-1}\ K_{n-1}.....J_0\ K_0$ | $Y_k..Y_0$ | $Q_{n-1}...Q_0$ |

La prima colonna contiene tante righe quante sono le possibili combinazioni degli ingressi (m) e dei FF (n), ovvero $2^{m+n}$ . La seconda colonna va riempita, per ogni ingresso di ogni FF (JK, D, SR, o T) tenendo conto delle EB ricavate al passo 3. La terza colonna va anch'essa riempita usando le EB ricavate al passo 3. La quarta colonna deve contenere i valori delle transizioni che ciascun FFi effettuer , sul successivo fronte di clock, in base al valore Qi precedentemente memorizzato ed ai valori dei propri input JiKi. La colonna va riempita, singolarmente per ogni Qi(t+1), in base ai corrispondenti valori degli input JiKi (t) (o Di, Ti, ecc) e Qi(t), letti sulla riga corrispondente. Ad esempio, se sulla riga j-esima della tabella si legge, per il FFk, Jk=0 Kk=0 (colonna 2) e Qk=1 (colonna 1), il futuro valore di Qk sar 0 e pertanto si inserisce uno 0 nell'elemento k-esimo della riga j-esima della quarta colonna.

Per quanto riguarda l'esempio di figura 3.6.1, la tabella sarebbe in teoria piuttosto laboriosa da scrivere, perch ci sono 3 input e 2 FF, per un totale di 5 variabili booleane nella prima colonna, che darebbero luogo a 32 combinazioni. Tuttavia possiamo osservare che D0 e D1 non dipendono da Q0 e Q1, e che la colonna degli stati futuri pu essere riempita solo in base ai valori di D0 e D1 (in un FF di tipo D le uscite Qi "seguono" gli ingressi applicati). Quindi possiamo descrivere la tabella in termini modulari.

Il modulo essenziale della tabella il seguente:

| X2X1X0(t) | D1D0(t) | Q1Q0(t+1) |
|---|---|---|
| 000 | 00 | 00 |
| 001 | 00 | 00 |
| 010 | 00 | 00 |
| 011 | 01 | 01 |
| 100 | 00 | 00 |
| 101 | 00 | 00 |
| 110 | 10 | 10 |
| 111 | 11 | 11 |

Indichiamo ora con le lettere **A**, **B** e **C** rispettivamente la prima, seconda e terza colonna della tabella sopra. La tabella completa sar la seguente:

| Q1Q0X2X1X0 (t) | D1D0(t) | Y(t) | Q1Q0 (t+1) |
|---|---|---|---|
| **00 A** | **B** | **0** | **C** |
| **01 A** | **B** | **1** | **C** |
| **10 A** | **B** | **1** | **C** |
| **11 A** | **B** | **1** | **C** |

Ogni cella della tabella sopra si intende composta di 8 righe. Ad esempio, la cella **00A** equivale alla sequenza di celle:

| |
|---|
| 00000 |
| 00001 |
| 00010 |
| 00011 |
| 00100 |
| 00101 |
| 00110 |
| 00111 |

**5** A partire dalle colonne 1, 3 e 4 della tabella degli stati futuri, si ricava l'automa, o la sua rappresentazione tabellare :

| | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
|---|---|---|---|---|---|---|---|---|
| S0 | S0/0 | S0/0 | S0/0 | S1/1 | S0/0 | S0/0 | S2/1 | S3/1 |
| S1 | S0/0 | S0/0 | S0/0 | S1/1 | S0/0 | S0/0 | S2/1 | S3/1 |
| S2 | S0/0 | S0/0 | S0/0 | S1/1 | S0/0 | S0/0 | S2/1 | S3/1 |
| S3 | S0/0 | S0/0 | S0/0 | S1/1 | S0/0 | S0/0 | S2/1 | S3/1 |

**6** Si procede (se applicabile) alla minimizzazione dell'automa
Gli stati possono essere suddivisi in due insiemi, {S0} con uscita =0, e {S1,S2,S3} con uscita =1. La tabella triangolare :

| | | | |
|---|---|---|---|
| S1 | X | | |
| S2 | X | | |
| S3 | X | | |
| | S0 | S1 | S2 |

Tutti gli stati nell'insieme {S1,S2,S3} sono indistinguibili, perch , come emerge dalla tabella delle transizioni, a fronte degli stessi input transitano negli stessi stati.

**7** Si disegna quindi l'automa minimizzato del circuito.

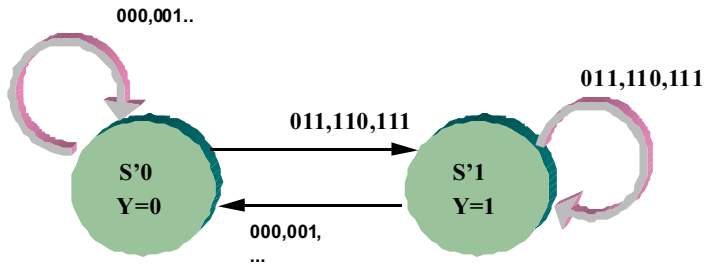



Fig. 3.10 *Automa minimizzato*

# Circuiti sequenziali notevoli

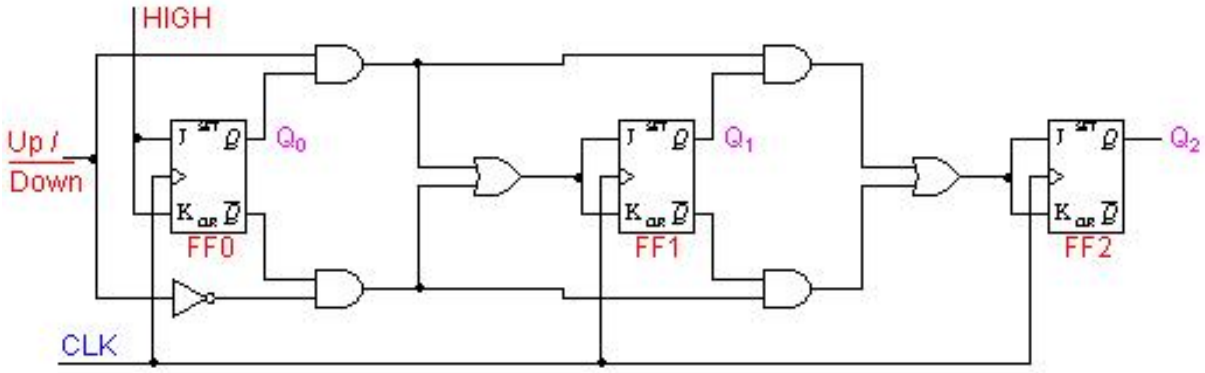
*I seguenti tipi di circuiti sequenziali notevoli verranno analizzati a lezione e esercitazione.*

**Applicazioni**:
*Riconoscimento di una sequenza di ingresso (vedere sul sito del corso i moltissimi esercizi di esame di questo tipo)*
*Memorizzazione e trasferimento di dati (vedere esercizi di esame)*

**Contatori** *(vedere Fummi et al. paragrafo 5.4)*

*Contatori Asincroni*
*Contatori con numeri di MOD>$2^N$*
*Contatore Asincroni alla rovescia*
*Contatori Sincroni*
*Contatori paralleli alla rovescia e bidirezionali UP-DOWN (vedere il sito* ***http://www.allaboutcircuits.com/vol_4/chpt_11/3.html*** ***e*** ***http://www.eelab.usyd.edu.au/digital_tutorial/part2/counter01.html****)*

**Esempio di contatore Up-Down**




**Registri**:
*vedere sul sito*: ***http://www.eelab.usyd.edu.au/digital_tutorial/part2/register01.html*** *e* ***su Fummi et al. par. 5.3***

*Ingresso Parallelo / Uscita Parallela*
*Ingresso Seriale/Uscita Parallela*
*Ingresso Parallelo/Uscita Seriale*
*Registri a scorrimento (Ingresso Seriale/Uscita Seriale)*

A titolo di esempio, il generico stato Si dell'automa di un contatore parallelo bidirezionale effettua le transizioni come nella figura:

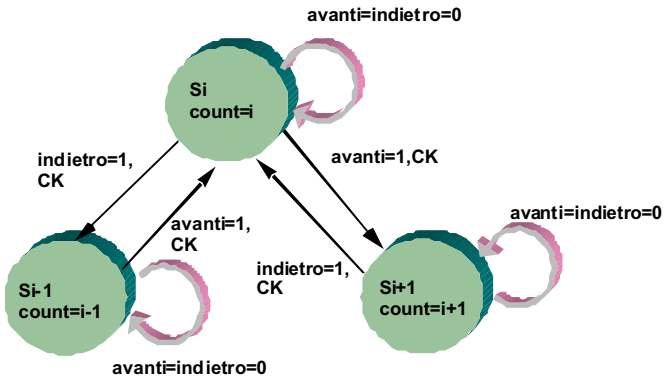
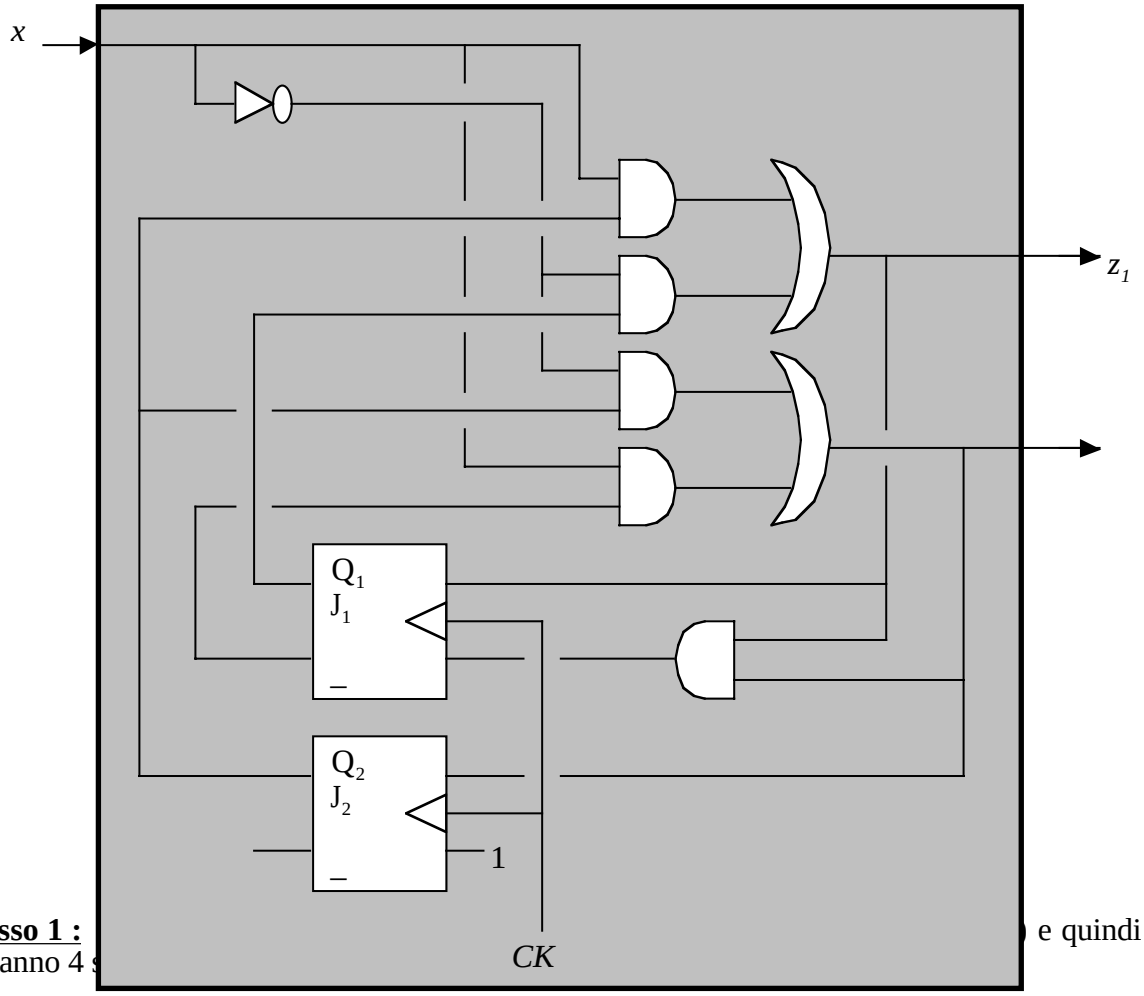



Fig. 3.11 *Transizioni dello stato i-esimo dell automa di un contatore parallelo bidirezionale*

# Esecizi di analisi

Dato un circuito sequenziale, stabilire la funzione che esso calcola. Sia dato il circuito e sia assuma che le uscite iniziali dei FF siano entrambe a 0.




**Passo 1 :** e quindi ci saranno 4 s

**Passo 2 :** in base ai valori di uscita dei FF avrò 4 possibili configurazioni:

- $Q_1\ Q_2 = 0\ 0 \Rightarrow$ associo lo stato $S_0$
- $Q_1\ Q_2 = 0\ 1 \Rightarrow$ associo lo stato $S_1$
- $Q_1\ Q_2 = 1\ 0 \Rightarrow$ associo lo stato $S_2$
- $Q_1\ Q_2 = 1\ 1 \Rightarrow$ associo lo stato $S_3$

**Passo 3 :** per ogni ingresso dei FF e per ogni uscita del circuito calcolo l'EB associata

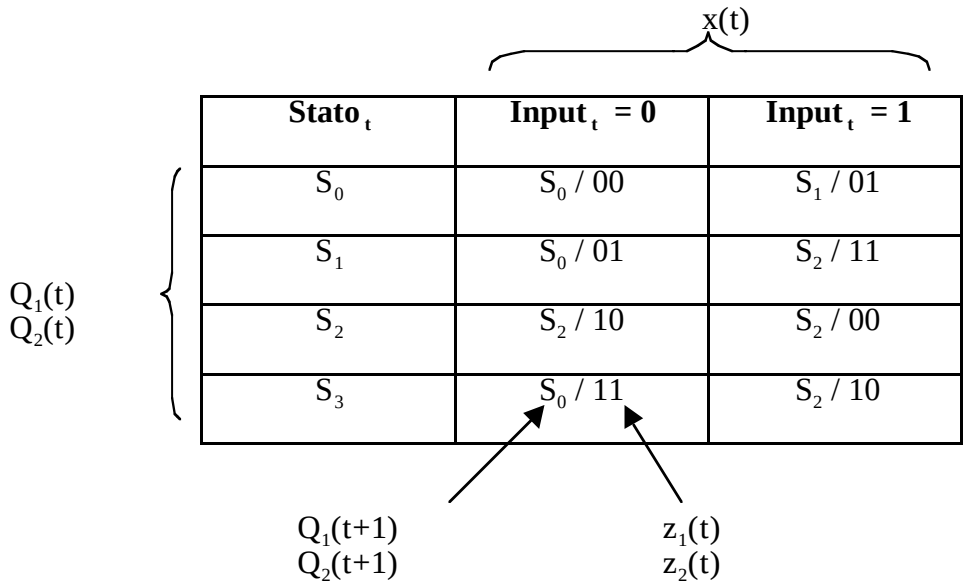
- $K_1 = z_1 \cdot z_2$
- $J_1 = z_1 = x \cdot Q_2 + \bar{x} \cdot Q_1$
- $K_2 = 1$
- $J_2 = z_2 = \bar{x} \cdot Q_2 + x \cdot \bar{Q}_1$

**Passo 4 :** tabella degli stati futuri

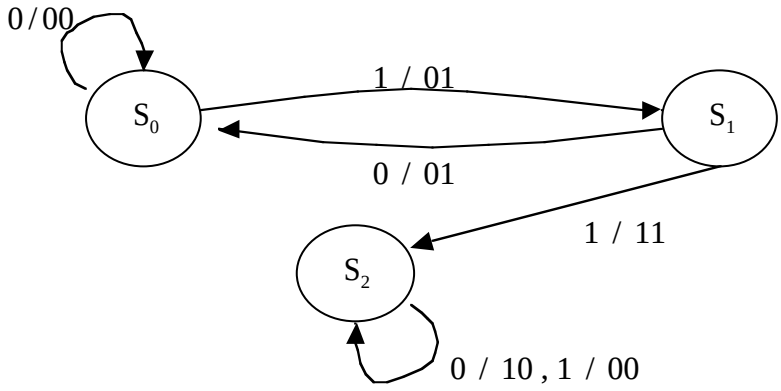
| Stato$_t$ | | Input | EntrateFF$_t$ | | | | UsciteCirc$_t$ | | Stato | |
|---|---|---|---|---|---|---|---|---|---|---|
| $Q_1(t)$ | $Q_2(t)$ | $x(t)$ | $J_1(t)$ | $K_1(t)$ | $J_2(t)$ | $K_2(t)$ | $z_1(t)$ | $z_2(t)$ | $Q_1(t+1)$ | $Q_2(t+1)$ |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |

**Passo 5 :** dalla 1^, 3^ e 4^ colonna ricavo la tabella dell'**automa di Mealy** corrispondente




Osserviamo che, con stato iniziale $Q_1 = Q_2 = 0$ (cioè $S_0$), si ha che lo stato $S_3$ è irraggiungibile e quindi può essere eliminato dall'automa. A seguito di questa modifica è facile convincersi che l'automa ottenuto è minimo ed ha la seguente rappresentazione grafica




Se invece vogliamo l'**automa di Moore** corrispondente applichiamo l'algoritmo di passaggio

- $\forall$ stato$_{Me}$ $\forall$ output$_{Me}$ si ha stato$_{Mo}$ = < stato$_{Me}$ , output$_{Me}$ >

- $\forall$ stato$_{Mo}$ $\forall$ input$_{Mo}$ si ha $\delta_{Mo}(<q, b>, c) = <\delta_{Me}(q, c), \lambda_{Me}(q, c)>$

$$\lambda_{Mo}(<q, b>) = b$$

da cui (chiaramente consideriamo l'automa di Mealy semplificato, cioè senza $S_3$)

| **Stato$_t$** | | | **Input$_t$ = 0** | **Input$_t$ = 1** |
|---|---|---|---|---|
| Stato$_{Me}$ | Output$_{Me}$ | Stato/Output$_{Mo}$ | | |
| 0 0 | 0 0 | $T_0$ / 00 | $T_0$ | $T_5$ |
| 0 0 | 0 1 | $T_1$ / 01 | $T_0$ | $T_5$ |
| 0 0 | 1 0 | $T_2$ / 10 | $T_0$ | $T_5$ |
| 0 0 | 1 1 | $T_3$ / 11 | $T_0$ | $T_5$ |
| 0 1 | 0 0 | $T_4$ / 00 | $T_1$ | $T_{11}$ |
| 0 1 | 0 1 | $T_5$ / 01 | $T_1$ | $T_{11}$ |
| 0 1 | 1 0 | $T_6$ / 10 | $T_1$ | $T_{11}$ |
| 0 1 | 1 1 | $T_7$ / 11 | $T_1$ | $T_{11}$ |
| 1 0 | 0 0 | $T_8$ / 00 | $T_{10}$ | $T_8$ |
| 1 0 | 0 1 | $T_9$ / 01 | $T_{10}$ | $T_8$ |
| 1 0 | 1 0 | $T_{10}$ / 10 | $T_{10}$ | $T_8$ |
| 1 0 | 1 1 | $T_{11}$ / 11 | $T_{10}$ | $T_8$ |

Come stato iniziale scegliamo $T_0$ (dobbiamo scegliere un qualsiasi stato costruito da $S_0$ con un output arbitrario; possiamo scegliere indifferentemente $T_0$ o $T_1$ perché tale scelta consente di minimizzare il numero di stati raggiungibili).
Possiamo subito eliminare gli stati $T_2$ , $T_3$ , $T_4$ , $T_6$ , $T_7$ , $T_9$ poiché non saranno mai raggiunti dallo stato iniziale $T_0$. Pertanto la tabella dell'automa è

| **Stato$_t$** | **Input$_t$ = 0** | **Input$_t$ = 1** |
|---|---|---|
| $T_0$ | $T_0$ / 00 | $T_5$ / 01 |
| $T_1$ | $T_0$ / 00 | $T_5$ / 01 |
| $T_5$ | $T_1$ / 01 | $T_{11}$ / 11 |
| $T_8$ | $T_{10}$ / 10 | $T_8$ / 00 |
| $T_{10}$ | $T_{10}$ / 10 | $T_8$ / 00 |
| $T_{11}$ | $T_{10}$ / 10 | $T_8$ / 00 |

Una minimizzazione sembrerebbe possibile (per esempio raggruppando assieme $< T_8 , T_{10} , T_{11} >$ oppure $< T_0 , T_1 >$, che hanno righe uguali nella tabella appena vista) ma ciò non è possibile perché tutti gli stati con stessa funzione di transizione hanno output diverso, mentre quelli con stesso output hanno funzioni di transizione diverse. Ciò è evidenziato dalla procedura di minimizzazione:

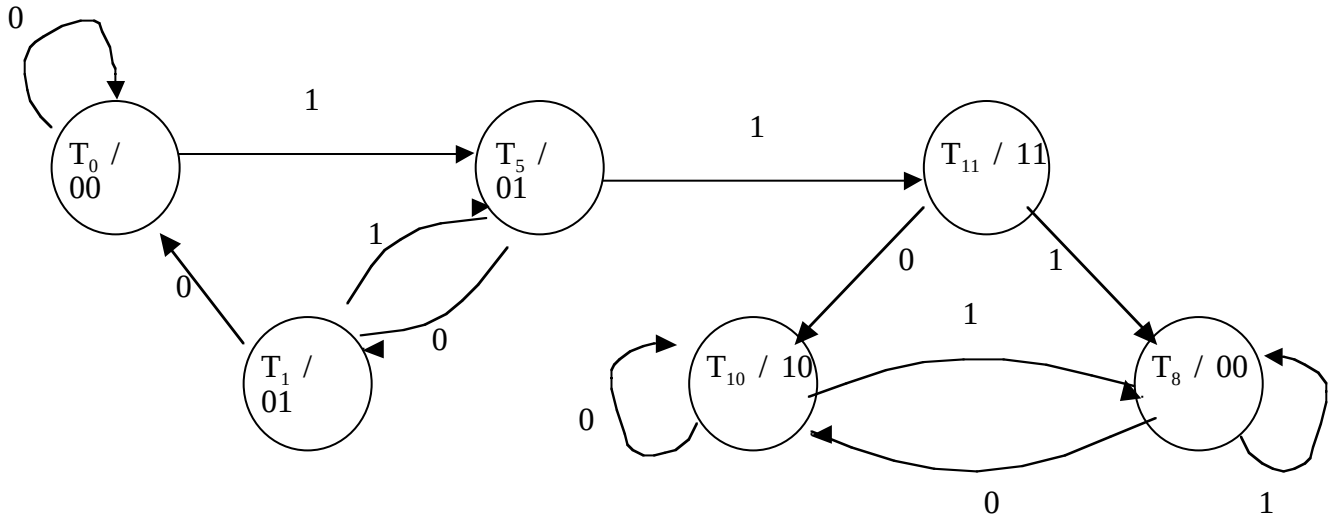
- raggruppa gli stati in base al loro output
- per ogni coppia di stati con output diversi, marca la casella ad essi corrispondente
- per ogni cella (p,q) non marcata
  *se* da p a q posso transire (con un certo output) su stati marcati
  *allora* marca (p,q) e tutte le celle da essa dipendenti

*altrim.* per ogni carattere c t.c. $\delta(p,c) \neq \delta(q,c)$
aggiungi in (p,q) la cella ( $\delta(p,c)$ , $\delta(q,c)$ )

In base al carattere emesso dallo stato possiamo distinguere le seguenti coppie:

| | | | | | |
|---|---|---|---|---|---|
| $T_1$ | X | | | | |
| $T_5$ | X | | | | |
| $T_8$ | | X | X | | |
| $T_{10}$ | X | X | X | X | |
| $T_{11}$ | X | X | X | X | X |
| | $T_0$ | $T_1$ | $T_5$ | $T_8$ | $T_{10}$ |

Osserviamo che : $(T_0,T_8)$ è da marcare poiché leggendo 0 si va nella cella marcata $(T_0,T_{10})$
$(T_1,T_5)$ è da marcare poiché leggendo 0 si va nella cella marcata $(T_0,T_1)$
Pertanto gli stati sono tutti distinguibili e l'automa ottenuto è minimo. La sua rappresentazione grafica è




## 3.7 Sintesi di circuiti sequenziali

La procedura di sintesi di un circuito sequenziale ricalca, in ordine inverso, la procedura di analisi. In questo caso, dobbiamo passare dalle specifiche (eventualmente verbali) all'automa e dall automa allo schema circuitale.

**Procedura di sintesi di un circuito sequenziale.**

**1** Dalle specifiche verbali, si ricava l'automa e/o la tabella delle transizioni del sistema sequenziale da progettare. Questo passo richiede semplicemente di riflettere sul problema e fornire una sua rappresentazione in termini di automi. Pu essere

necessario scomporre il problema in sottoproblemi, e modellare con un automa ciascun sottoproblema.

**2** Una volta ottenuto l'automa (o gli automi) che modellano il funzionamento del circuito, si verifica la possibilit di minimizzazione secondo il metodo visto.

**3** Si progetta la parte di memorizzazione del circuito sequenziale, per ciascun blocco-automa del sistema. Si procede nel seguente modo: se l'automa ha N stati, occorrono $n=\lceil \log_2 N \rceil$ flip flop. La scelta del tipo di FF arbitraria (il caso pi generale rappresentato da FF JK). Si associa ad ogni configurazione degli n FF il simbolo di uno degli N stati dell'automa A questo proposito, ricordiamo che lo schema generale di una macchina sequenziale il seguente:

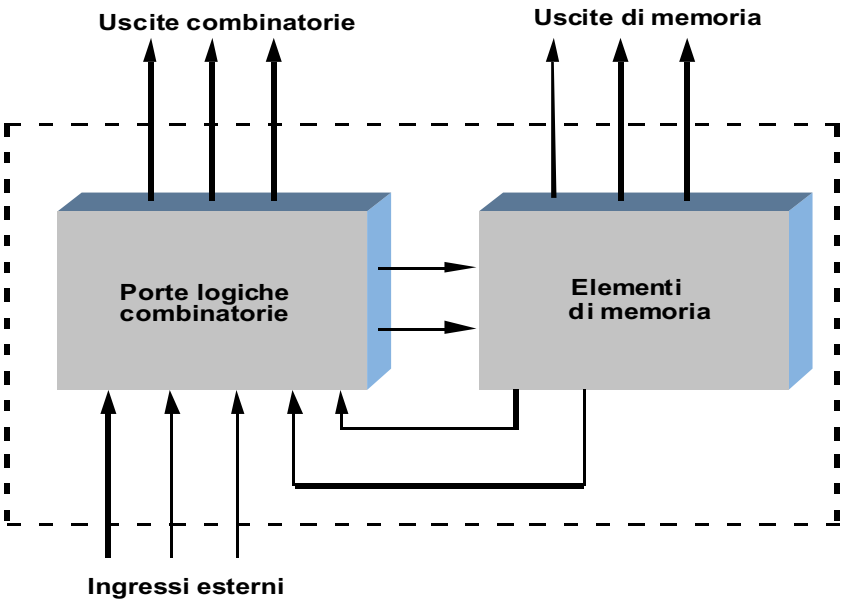



Fig. 3.14 *Schema generico di una macchina sequenziale*

Dunque, dobbiamo progettare dapprima la parte di memorizzazione del sistema, scegliendo il numero e tipo di FF, e quindi la parte combinatoria. la parte combinatoria riceve in ingresso gli input del sistema e i bit Qi della memoria (che indicano lo stato attuale), e produce in uscita le uscite richieste, nonch i segnali di eccitazione per i FF della memoria, necessari a far effettuare al sistema le transizioni desiderate.

**4** Si costruisce una tabella cos strutturata:

| possibili combinazioni degli input e degli stati **in t** | corrispondenti valori delle funzioni di eccitazione dei FF **in t** (nel caso di FF JK) | corrispondenti valori delle uscite **in t** | stati futuri, ovvero nuovi valori che saranno memorizzati **in t+1** |
|---|---|---|---|
| $Q_{n-1}...Q_0\ I_m...I_0$ | $J_{n-1}\ K_{n-1}.....J_0\ K_0$ | $Y_k..Y_0$ | $Q_{n-1}...Q_0$ |

La tabella analoga a quella utilizzata per il problema di analisi di circuiti sequenziali, ma in questo caso, possiamo immediatamente riempire le colonne 1, 3 e 4, semplicemente osservando l'automa. Per riempire la colonna due dobbiamo tener conto della tabella delle transizioni dei tipi di FF utilizzati. Supponiamo di utilizzare FF di tipo JK. Nella riga i-esima della tabella, colonne 1 e 4, osserver una certa transizione del FFj $Qj(t) \rightarrow Qj(t+1)$, ad esempio, da 0 a 1.

$Q_{n-1}..\mathbf{Qj}....Q_0 \rightarrow Q_{n-1}..\mathbf{Qj}....Q_0$

Nella corrispondente posizione della riga i-esima della colonna due:

$J_{n-1}$ $K_{n-1}$..**JjKj**....$J_0$ $K_0$

dovr assegnare agli input Jj e Kj del FFj i valori che consentono al FF di effettuare, sul successivo fronte di Clock, la transizione desiderata, nel caso in considerazione, la coppia 1X (sia J=1K=1 che J=1K=0 ottengono l'effetto di far commutare Q da 0 a 1).Per riempire la colonna 2 della tabella, si tengano presenti le seguenti tabelle di transizione dei vari tipi di FF (si ricordi X= *dont'care, o condizione di indifferenza*):

| **Q(t) →Q(t+1)** | **J(t) K(t)** | **S(t) R(t)** | **T(t)** | **D(t)** |
|---|---|---|---|---|
| 0 → 0 | 0 X | 0 X | 0 | 0 |
| 0 → 1 | 1 X | 1 0 | 1 | 1 |
| 1 → 0 | X 1 | 0 1 | 1 | 0 |
| 1 → 1 | X 0 | X 0 | 0 | 1 |

**5** In base alla tabella, possiamo progettare la parte combinatoria del circuito. Le colonne 1, 2 e 4 rappresentano infatti la tabella di verit della parte combinatoria del circuito da progettare, poich stabiliscono una relazione tra input del circuito e relativi output (si veda punto 3). Si applica il metodo di Karnaugh o uno qualsiaisi dei metodi di sintesi di circuiti combinatori per ottenere le funzioni di trasferimento di Jn-1Kn-1....J0K0 e di Yk...Y0. Si ricordi che le condizioni di indifferenza (indicate con X, _ o *d* nei vari libri di testo) possono essere utilizzate per ottimizzare la logica combinatoria del circuito (creazione di primi implicanti di dimensione maggiore possibile).

**Esempio:**

Si progetti una macchina sequenziale sincrona che legge una sequenza di bit e che emette 1 ogni volta che il numero di bit ricevuto pari e che, assumendo il primo bit letto come bit meno significativo, la sequenza letta codifichi un numero divisibile per 5.
Ad esempio, sull'input 010110101, la macchina deve produrre l'output 000100010.
Si utilizzino automi di Mealy e flip-flop di tipo T.

### 1 Ricavare l'automa

Per quanto concerne l'esempio, si osservi la ricorrenza dei resti della divisione fra il bit di peso $2^i$ e 5, per i=0,1,2, . La sequenza ricorrente 1, 2, 4, 3. Si pu quindi scomporre il problema nella progettazione di due automi: un automa che sappia contare ciclicamente fino a 4 e che produca in uscita una codifica del peso da attribuire al bit in lettura e un automa che tenga memoria del valore (modulo 5) del numero letto finora (ovvero del resto). Tale automa avr uno stato per ogni possibile resto, ovvero: 0, 1, 2, 3 e 4.
Se *n* il valore corrispondente alla sequenza di bit letta in un certo istante, ed Ri rappresenta il resto della divisione di *n* per 5, l'arrivo di un bit uguale a 0 non modifica lo stato del sistema (0¥n=n, ¥ = concatenazione), mentre l'arrivo di un bit=1 contribuisce per unvalore che dipende

dal peso $2^i$ (mod 4) del bit, ovvero 1,2, 4 o 3. Se indichiamo con Pk il "contributo" del bit $2^i$ e con Ri il resto attuale, il sistema transiter nello stato in cui il resto pari a $[Ri + Pk]_{mod\ 5}$

Lo schema a blocchi della macchina risulta il seguente:

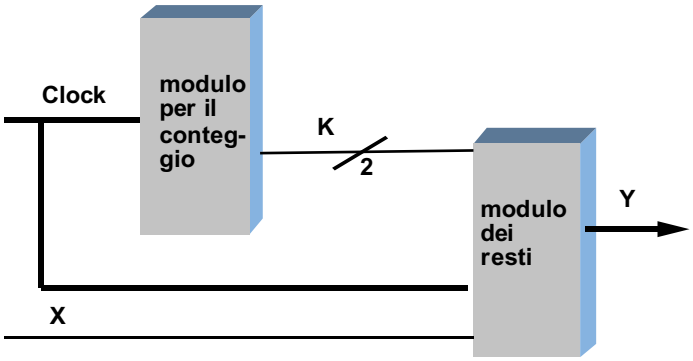



Fig. 3.12 *Schema di una macchina che riconosce sequenze modulo n*

Si consideri dapprima il blocco adibito al conteggio. Esso costituito da un automa in grado di contare ciclicamente da 1 a 4, dunque con 4 stati. In ogni stato, l'automa dovr produrre un output a due bit che indica al modulo dei resti il "peso" da attribuire all'i-esimo (mod 4) bit ricevuto in ingresso (input X).
Si codifichino i pesi in uscita, da 1 a 4, nel seguente modo: P1= 00, P2=01, P3=10, P4=11. Vale a dire : l'uscita P1 sar codificata ponendo entrambi i bit K di controllo a 0, P2 ponendo K0=1 e K1=0, ecc.
Queste uscite saranno portate in ingresso, insieme al segnale x, all'automa presente nel blocco dei resti. Si vedano gli automi di Mealy in figura, rispettivamente del contatore e del modulo dei resti. Il simbolo ↑ indica i fronti positivi del clock di sistema.

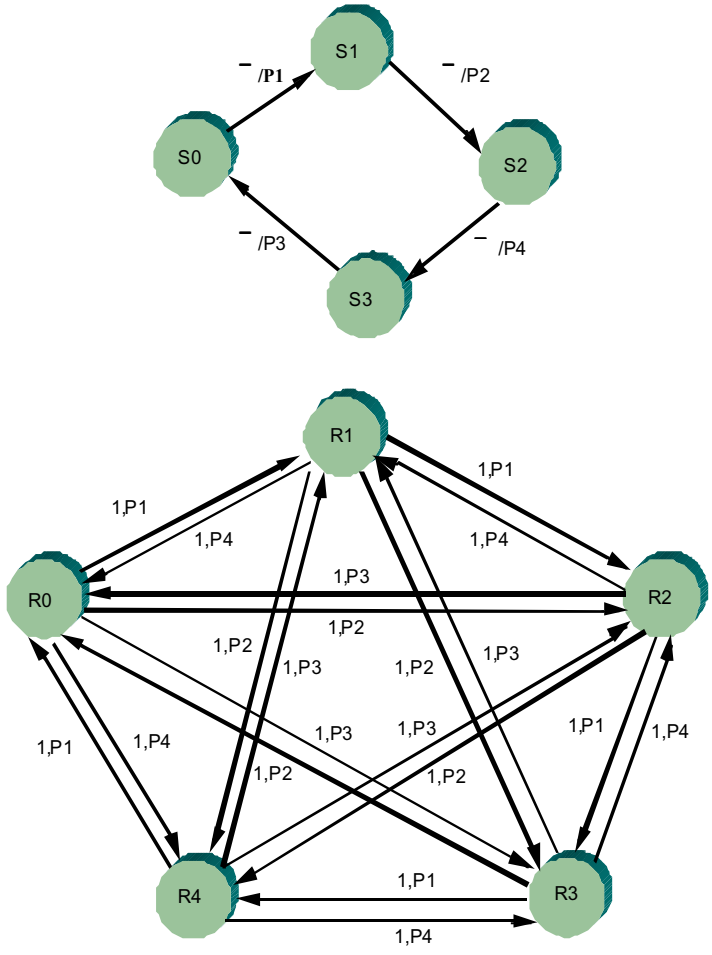



Fig. 3.13 *Automa a stati finiti per riconoscere numeri divisibili per 5*

Per non appesantire il grafo del secondo automa non sono state indicate le transizioni con ingresso x=0 (per qualsiasi valore di K0 e K1) e le uscite Z associate alle transizioni. Tutte le transizioni con ingresso 0 fanno rimanere l'automa nello stesso stato. Tutte le transizioni entranti (anche con ingresso 0) nello stato R0 producono un'uscita Z=1 (cio il numero divisibile per 5), mentre negli altri casi, Z=0. L'uscita Y dell'intero sistema sequenziale deve tener conto del valore di Y e della parit .

**2 verificare la possibilit di minimizzazione**.

La tabella delle transizioni del secondo automa la seguente (considerando solo gli ingressi x=1 e riportando le configurazioni K0 K1):

| **Stato** | **00** | **01** | **10** | **11** |
|---|---|---|---|---|
| R0 | R1/0 | R2/0 | R3/0 | R4/0 |
| R1 | R2/0 | R3/0 | R4/0 | R0/1 |
| R2 | R3/0 | R4/0 | R0/1 | R1/0 |
| R3 | R4/0 | R0/1 | R1/0 | R2/0 |
| R4 | R0/1 | R1/0 | R2/0 | R3/0 |

Dall'esame della tabella appare chiaro che l'automanon e' minimizzabile (ogni stato, eccetto lo stato 1, produce una e una sola uscita 1 in corrispondenza di ingressi diversi).

**2 Progettare la parte di memorizzazione**.
Codifichiamo i 5 stati dell'automa dei resti con 3 flip flop di tipo T (toggle), le cui uscite chiamiamo Q2 Q1 Q0, nel seguente modo: R0 = 000, R1= 001, R2=010, R3= 011, R4=100.

**4 Costruire una tabella delle transizioni**:

La tabella dell'automa dei resti (considerando solo la parte relativa all'ingresso x=1) la seguente:

| | | | S(t) | | | T(t) | | | S(t+1) | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **X** | **K1** | **K0** | **Q2** | **Q1** | **Q0** | **T2** | **T1** | **T0** | **Q2** | **Q1** | **Q0** | **Z** |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 | x | x | x | x | x | x | x |
| 1 | 0 | 0 | 1 | 1 | 0 | x | x | x | x | x | x | x |
| 1 | 0 | 0 | 1 | 1 | 1 | x | x | x | x | x | x | x |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 | x | x | x | x | x | x | x |
| 1 | 0 | 1 | 1 | 1 | 0 | x | x | x | x | x | x | x |
| 1 | 0 | 1 | 1 | 1 | 1 | x | x | x | x | x | x | x |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 1 | x | x | x | x | x | x | x |
| 1 | 1 | 0 | 1 | 1 | 0 | x | x | x | x | x | x | x |
| 1 | 1 | 0 | 1 | 1 | 1 | x | x | x | x | x | x | x |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 | 0 | 1 | x | x | x | x | x | x | x |
| 1 | 1 | 1 | 1 | 1 | 0 | x | x | x | x | x | x | x |
| 1 | 1 | 1 | 1 | 1 | 1 | x | x | x | x | x | x | x |

**5. In base alla tabella, possiamo progettare la <u>parte combinatoria</u> del circuito.**
Le colonne 1, 2 e 4 rappresentano infatti la tabella di verit della parte combinatoria del circuito da progettare, poich stabiliscono una relazione tra input del circuito e relativi output (si veda punto 3). Si applica il metodo di Karnaugh o uno qualsiaisi dei metodi di sintesi di circuiti combinatori per ottenere le funzioni di trasferimento di Jn-1Kn-1....J0K0 e di Yk...Y0.

Avendo scelto flip- flop di tipo T, e rimanendo nello stesso stato sull'ingresso x=0, possiamo trascurare l'altra parte della tabella di verit nel determinare la rete combinatoria per i segnali T0, T1, T2.

Si ha quindi:

(¬=not)

T0 = ¬(K1)¬(K0)¬(Q2)¬(Q1)¬(Q0) + ¬(K1)¬(K0)¬(Q2)¬(Q1)Q0 + ¬(K1)¬(K0)¬(Q2)Q1¬(Q0) + ¬(K1)K0¬(Q2)Q1Q0 + ¬(K1)K0Q2¬(Q1)¬(Q0) + K1¬(K0)¬(Q2)¬(Q1)¬(Q0) + K1¬(K0)¬(Q2)¬(Q1)Q0 + K1K0¬(Q2)¬(Q1)Q0 + K1K0¬(Q2)Q1¬(Q0) + K1K0Q2Q1Q0

Minimizzando si ottiene:

T0 = ¬(K1)¬(K0)¬(Q2)(¬(Q1)+¬(Q0)) + ¬(K1)K0(¬(Q2)Q1Q0 + Q2¬(Q1)¬(Q0)) + K1¬(K0)¬(Q2)¬(Q1) + K1K0(¬(Q2)¬(Q1)Q0 + K1K0¬(Q2)Q1¬(Q0) + K1K0Q2Q1Q0)

T1 = ¬(K1)¬(K0)¬(Q2)¬(Q1)Q0 + ¬(K1)K0¬(Q2)¬(Q1)¬(Q0) + ¬(K1)K0¬(Q2)¬(Q1)Q0 + ¬(K1)K0¬(Q2)Q1¬(Q0) + ¬(K1)K0¬(Q2)Q1Q0 + K1¬(K0)¬(Q2)¬(Q1)¬(Q0) + K1¬(K0)¬(Q2)Q1¬(Q0) + K1¬(K0)¬(Q2)Q1Q0 + K1¬(K0)Q2¬(Q1)¬(Q0) + K1K0¬(Q2)Q1¬(Q0) + K1K0Q2¬(Q1)¬(Q0)

Minimizzando si ottiene:

T1 = ¬(K1)¬(K0)¬(Q2)¬(Q1)Q0 + ¬(K1)K0(¬(Q2) + K1¬(K0)(¬(Q2)¬(Q0) + ¬(Q2)Q1Q0 + Q2¬(Q1)¬(Q0)) + K1K0(¬(Q2)Q1¬(Q0) + Q2¬(Q1)¬(Q0))

T2 = ¬(K1)¬(K0)¬(Q2)Q1Q0 + ¬(K1)¬(K0)Q2¬(Q1)¬(Q0) + ¬(K1)K0¬(Q2)Q1¬(Q0) + ¬(K1)K0Q2¬(Q1)¬(Q0) + K1¬(K0)¬(Q2)¬(Q1)Q0 + K1¬(K0)Q2¬(Q1)¬(Q0) + K1K0¬(Q2)¬(Q1)¬(Q0) + K1K0Q2¬(Q1)¬(Q0)

Minimizzando si ottiene:

T2 = Q2¬(Q1)¬(Q0) + ¬(K1)¬(K0)¬(Q2)Q1Q0 + ¬(K1)K0¬(Q2)Q1¬(Q0) + K1¬(K0)¬(Q2)¬(Q1)Q0 + K1K0¬(Q2)¬(Q1)¬(Q0)

Tutte queste somme considerano l'ingresso x=1.
Per quanto riguarda Z si osservi che oltre alla somma di prodotti derivante dalla tabella per ingresso 1, si produce 1 anche in corrispondenza della configurazione ¬(Q2)¬(Q1)¬(Q0) per qualsiasi configurazione di K1 e K0, in corrispondenza dell'ingresso x=0.

Si ha quindi:
Y= ¬(x)¬(Q2)¬(Q1)¬(Q0) + x(¬(K1)¬(K0)Q2¬(Q1)¬(Q0) + ¬(K1)K0¬(Q2)Q1Q0 + K1¬(K0)¬(Q2)Q1¬(Q0) + K1K0¬(Q2)¬(Q1)Q0.

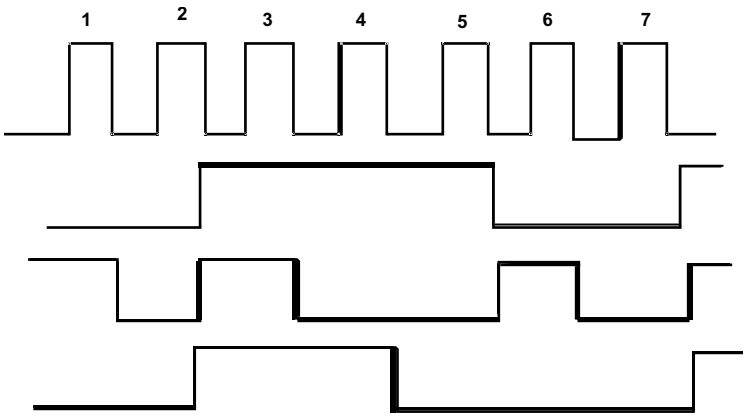
Per ottenere l'uscita Y si dovr comporre l'uscita Z dell'automa dei resticon un indicatore della parit . Tale indicatore costituito dall'ingresso dello automa del blocco Cont nello stato S0 o nello stato S2. Si possono quindi sfruttare i segnali K1 e K0. In particolare le configurazioni 01 e 10 indicano che siamo arrivati a leggere un numero di bit pari.
Si pu quindi definire la rete combinatoria di uscita definita dalla seguente tabella di verit :

| K1 | K0 | Z | Y |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |

Risulta quindi: Y = ¬(K1)K0z + K1¬(K0)z = z(¬(K1)K0 + K1¬(K0))

## Esempio 2

Sintetizzare un circuito sequenziale sincrono in base alle specifiche temporali riportate nel seguito. Il circuito riceve in input un segnale di temporizzazione (CK) e produce tre uscite, $O_0$ $O_1$ ed $O_2$.




**Soluzione**. Si tratta di un contatore di sequenze, le cui commutazioni avvengono sul fronte di discesa del clock, che rappresenta il solo input del sistema. L'output pu essere ricavato direttamente dalle uscite Qi dei FF. La sequenza ciclicamente contata :
000 →111→101 →110→ 010 → 000 .....

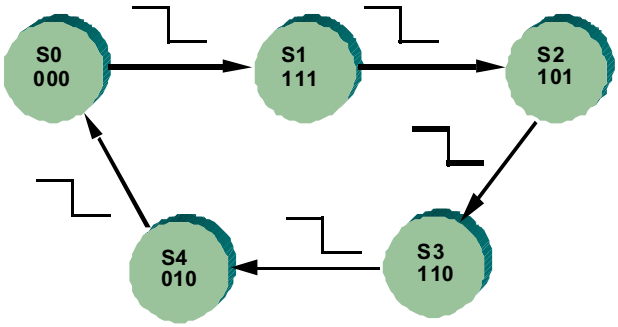



Fig. 3.15 *Automa di controllo del contatore di sequenze*

La soluzione pi semplice consiste nel far coincidere la codifica degli stati con il valore della sequenza contata (e quindi dell'output), utilizzando una macchina di Moore le cui transizioni avvengono in corrispondenza di ogni fronte ↓, come mostrato nell'automa in figura. Si costruisce la tabella degli stati futuri nel seguente modo (ad esempio utilizzando FF di tipo D). La prima e terza colonna della tabella rappresentano una tabella di verit da cui ricavare, col metodo di Karnaugh, le funzioni trasferimento per $D_2D_1D_0$ , e quindi il circuito.

Le configurazioni non incluse nella sequenza possono essere associate a transizioni in stati qualsiasi (X) ed in tal modo possibile ottenere espressioni semplificate per $D_2D_1D_0$ .

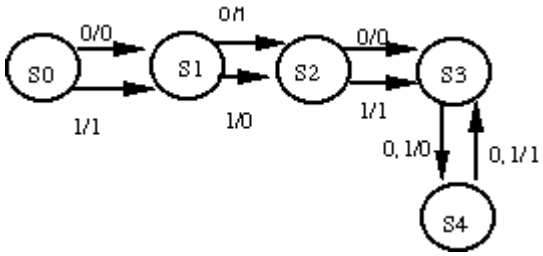
Alternativamente, si possono "forzare" transizioni dagli stati non consentiti ad uno stato della sequenza, ad esempio 000.

| S(t)<br>$Q_2Q_1Q_0$ | S(t+1)<br>$Q_2Q_1Q_0$ | D(t)<br>$D_2D_1D_0$ |
|---|---|---|
| 000 | 111 | 111 |
| 001 | XXX | XXX |
| 010 | 000 | 000 |
| 011 | XXX | XXX |
| 100 | 010 | 010 |
| 101 | 100 | 100 |
| 110 | XXX | XXX |
| 111 | 101 | 101 |

# Esempio 3:

Una macchina sequenziale, posta in uno stesso stato iniziale, riceve in ingresso una volta la sequenza 1 1 1 0 1 0 0 1 0 1 1 0, producendo in uscita la sequenza 1 0 1 0 1 0 1 0 1 0 1 0, e una volta la sequenza 0 0 0 0 0 0 1 1 1 1 1 0, fornendo in uscita e 0 1 0 0 1 0 1 0 1 0 1 0. Si mostri l'automa a stati finiti che descrive il comportamento della macchina, assumendo che per ogni continuazione delle sequenze, la macchina continui a produrre in uscita 0 e 1 alternati. (Non sono richiesti tabella di verità né schema circuitale, ma solo il diagramma degli stati. Si suggerisce di usare automi di Mealy).

**Soluzione**
Si può osservare che, a partire dal quarto istante in poi, l'automa, indipendentemente dall'ingresso, si alterna a produrre 0 o 1. Pertanto l'automa avrà le prime tre transizioni che emettono l'output in base all'input, mentre dalla quarta transizione in poi produrrà alternativamente 0 e 1 (e quindi avrà altri due stati con un ciclo di lunghezza due che, indipendentemente dall'input, daranno in output 0 e 1 in sequenza). Pertanto l'automa è




Se vogliamo sintetizzare tale automa in un circuito sequenziale, avremo bisogno di tre 3 FF, con la corrispondenza :

- $Q_2\ Q_1\ Q_0 = 000$ $S_0$
- $Q_2\ Q_1\ Q_0 = 001$ $S_1$
- $Q_2\ Q_1\ Q_0 = 010$ $S_2$
- $Q_2\ Q_1\ Q_0 = 011$ $S_3$
- $Q_2\ Q_1\ Q_0 = 100$ $S_4$

Dall'automa, otteniamo la seguente tabella degli stati futuri:

| x | $Q_2$ | $Q_1$ | $Q_0$ | $Q_2$' | $Q_1$' | $Q_0$' | y |
|---|---|---|---|---|---|---|---|
| *0* | *0* | *0* | *0* | 0 | 0 | 1 | 0 |
| *0* | *0* | *0* | *1* | 0 | 1 | 0 | 1 |
| *0* | *0* | *1* | *0* | 0 | 1 | 1 | 0 |
| *0* | *0* | *1* | *1* | 1 | 0 | 0 | 0 |
| *0* | *1* | *0* | *0* | 0 | 1 | 1 | 1 |
| *0* | *1* | *0* | *1* | X | X | X | X |
| *0* | *1* | *1* | *0* | X | X | X | X |
| *0* | *1* | *1* | *1* | X | X | X | X |
| *1* | *0* | *0* | *0* | 0 | 0 | 1 | 1 |
| *1* | *0* | *0* | *1* | 0 | 1 | 0 | 0 |
| *1* | *0* | *1* | *0* | 0 | 1 | 1 | 1 |
| *1* | *0* | *1* | *1* | 1 | 0 | 0 | 0 |
| *1* | *1* | *0* | *0* | 0 | 1 | 1 | 1 |
| *1* | *1* | *0* | *1* | X | X | X | X |
| *1* | *1* | *1* | *0* | X | X | X | X |
| *1* | *1* | *1* | *1* | X | X | X | X |

Utilizziamo FF di tipo D (per cui $D_i = Q_i'$ ); ricaviamo le EB minime per le entrate dei FF e per le uscite del circuito col metodo di Karnaugh:

| ***x*** ***$Q_2$*** \ ***$Q_1$*** ***$Q_0$*** | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 0 | 0 | 1 | 0 |
| 01 | 0 | X | X | X |
| 11 | 0 | X | X | X |
| 10 | 0 | 0 | 1 | 0 |

da cui $D_2 = Q_1 \cdot Q_0$

| ***x*** ***$Q_2$*** \ ***$Q_1$*** ***$Q_0$*** | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 0 | 1 | 0 | 1 |
| 01 | 1 | X | X | X |
| 11 | 1 | X | X | X |
| 10 | 0 | 1 | 0 | 1 |

da cui $D_1 = Q_2 + ( Q_1 \oplus Q_0 )$

| $x\ Q_2$ \ $Q_1\ Q_0$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 1 | 0 | 0 | 1 |
| 01 | 1 | X | X | X |
| 11 | 1 | X | X | X |
| 10 | 1 | 0 | 0 | 1 |

da cui $D_2 = \overline{Q_0}$

| $x\ Q_2$ \ $Q_1\ Q_0$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 0 | 1 | 0 | 0 |
| 01 | 1 | X | X | X |
| 11 | 1 | X | X | X |
| 10 | 1 | 0 | 0 | 1 |

da cui $y = Q_2 + x\cdot\overline{Q_0} + \overline{x}\cdot\overline{Q_1}\cdot Q_0$

Da tali espressioni è banale ricavarsi il circuito sequenziale.

Nota: si osservi in questo esercizio l'utilizzzo ottimale delle condizioni di indifferenza per l'ottimizzazione della rete combinatoria!

## 3.8 Bus di comunicazione fra registri

La funzione dei sistemi sequenziali di memorizzare ed elaborare l'informazione. Un'altra funzione importante quella di **trasferire** informazioni, tra registri e registri, registri e macchine sequenziali pi complesse Nella seguente tabella sono riportati i principali tipi di interconnessione, ed i relativi dispositivi.

| | destinazione prefissata | destinazione variabile |
|---|---|---|
| sorgente prefissata | punto-punto (porte logiche o tri-state) | 1-m con decodificatore |
| sorgente variabile | multiplexer | mesh |

Per semplicit , supporremo che sorgente e destinazione siano registri di tipo Parallel Input Parallel Output, ma in realt una informazione pu essere trasferita su dispositivi qualsiasi, purch dotati di capacit di memorizzazione (ad esempio, un circuito sommatore aritmetico).

### 3.8.1 Sorgente e destinazione prefissata

Consente di trasferire informazione da una sorgente ad una destinazione prefissata. Il trasferimento abilitato da un segnale di controllo che agisce su *porte logiche* o su *buffer tri-state*. Un buffer tri-state un interruttore elettronico, spesso integrato dentro il dispositivo di memorizzazione stesso (FF). Quando il segnale di controllo del buffer 0, l'impedenza fra ingresso e uscita del buffer molto alta, ossia come se il collegamento fra sorgente e destinazione fosse "tagliato". Quando il segnale di controllo un 1, l'informazione presente sulla sorgente viene trasferita a destinazione.

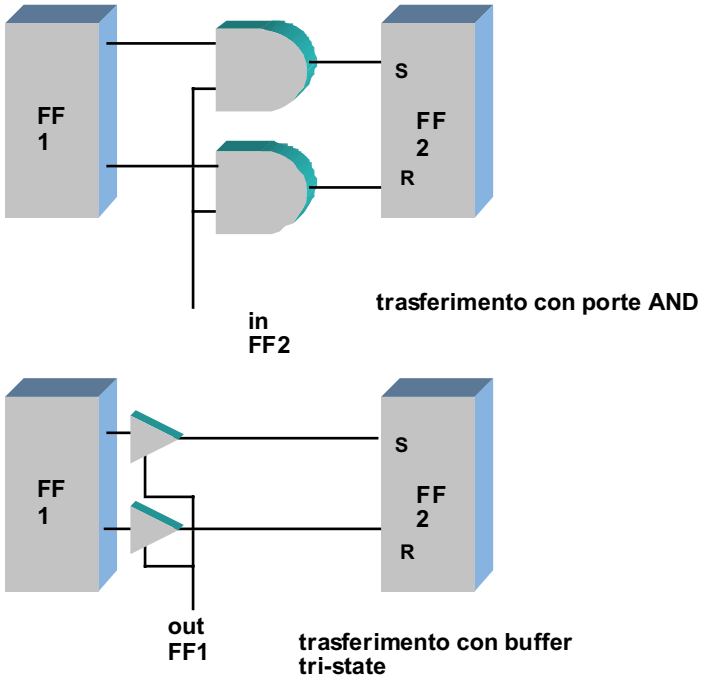



Fig. 3.16 *Possibili collegamenti tra due generici Flip-Flop*

In figura, viene illustrato solo il collegamento fra due generici FF della sorgente e della destinazione. L'informazione viene letta dal dispositivo destinazione sul successivo fronte di clock, non evidenziato in figura.
Dunque, il segnale di abilitazione (**inFF2** o **outFF1**) deve durare per un intervallo di tempo sufficientemente lungo perch l'informazione possa essere memorizzata sulla destinazione, ma non troppo lungo perch l'informazione sulla sorgente cambi. Schematicamente, un collegamento monodirezionale fra due registri viene illustrato come in figura:

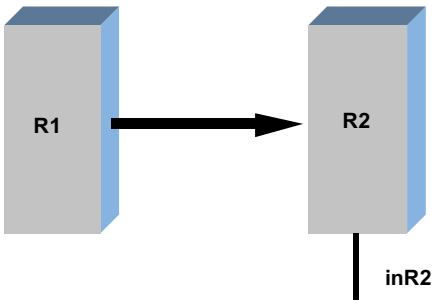



Fig. 3.17 *Schema di collegamento monodirezionale fra due registri*

### 3.8.2 Sorgente variabile destinazione prefissata

Per collegare varie sorgenti con una destinazione prefissata si utilizza un multiplexer. Per N sorgenti ed una destinazione, m=log2N linee di controllo selezionano la sorgente da collegare. Si noti che, se ogni registro memorizza M bit, servono in realt M multiplexers, controllati per dalle stesse m linee di controllo.

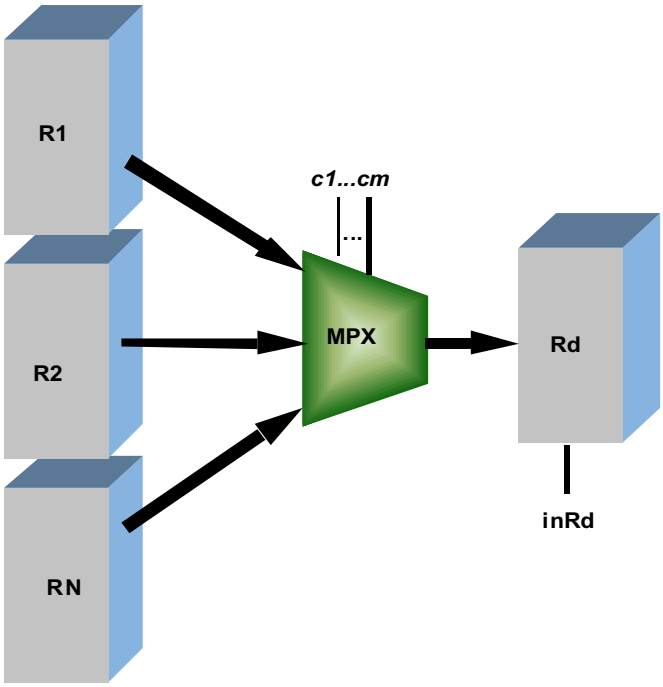



Fig. 3.18 *Schema di un bus di comunicazione molti a uno*

Si noti in figura che il segnale di controllo per l'abilitazione alla lettura del registro destinazione sempre necessario.

### 3.8.3 Sorgente prefissata destinazione variabile

In questo caso si pu utilizzare un demultiplexer (ovvero un decodificatore con porte logiche). Per N destinazioni, il demultiplexer sar controllato da m=log2N linee di controllo.

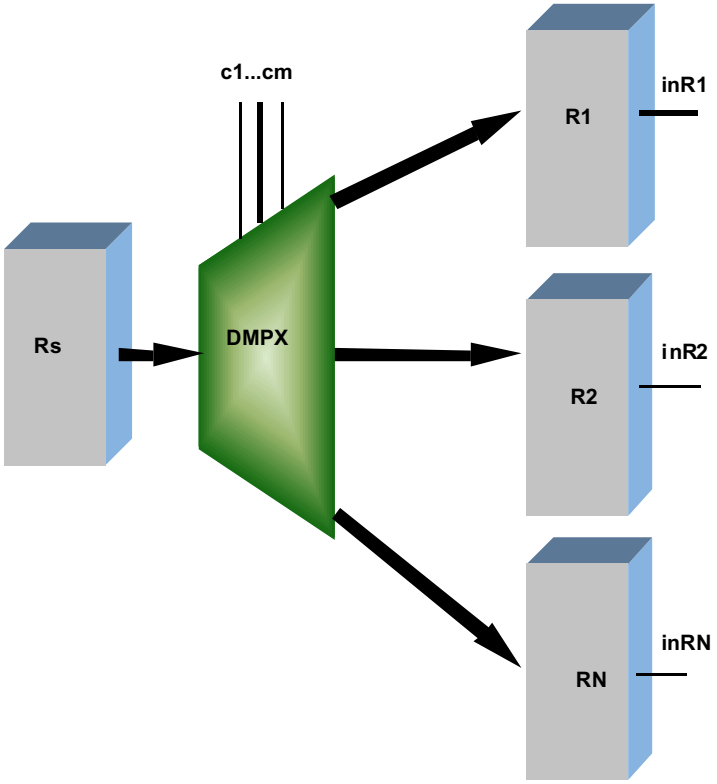



Fig. 3.19 *Schema di un bus di comunicazione uno a molti*

### 3.8.4 Sorgente variabile destinazione variabile

Per questo tipo di collegamento, con M sorgenti ed N destinazioni, occorrono N multiplexer, controllati da m=log2M segnali di controllo. Inoltre, occorrono i segnali di abilitazione alla lettura per ciascuno degli N registri destinazione.

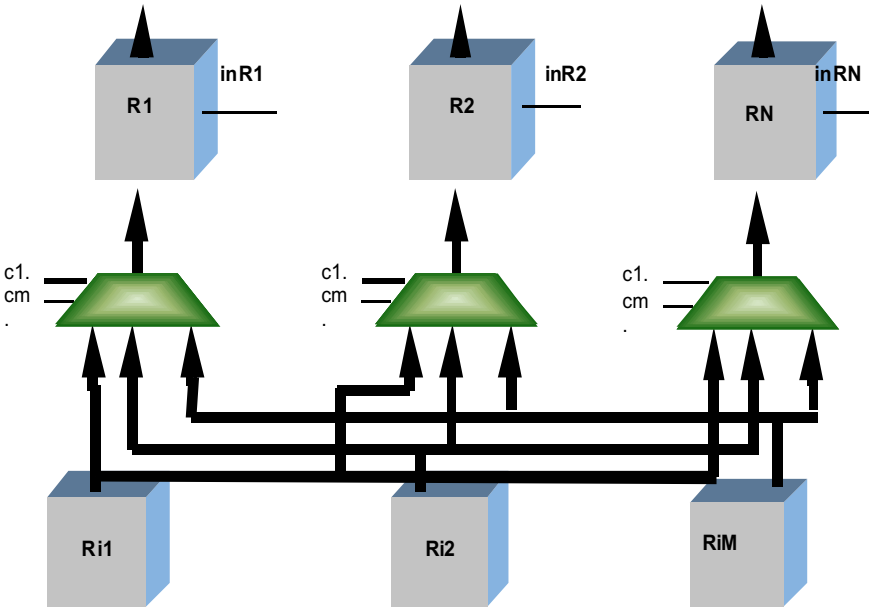


Fig. 3.20 *Schema di un bus di comunicazione molti a molti*

### 3.8.5 Progettare reti di interconnesione fra registri

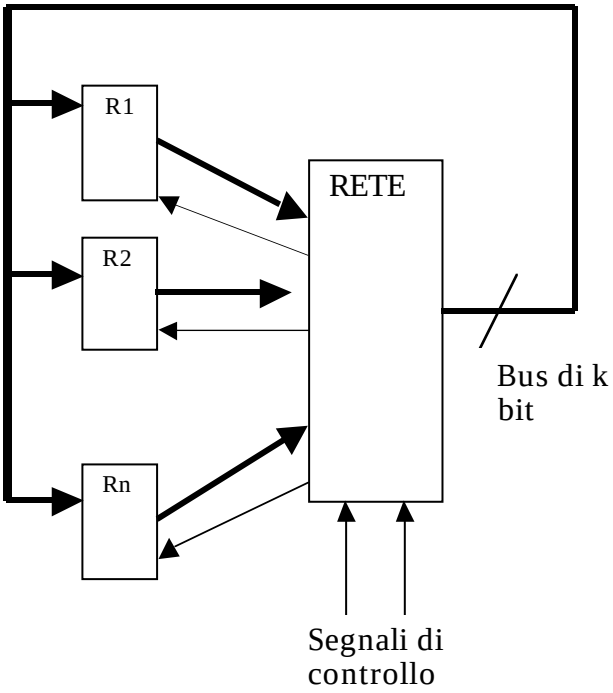



**Figura 1**

Questo schema illustra una rete di connessione fra n registri di k bit ciascuno. La linea in grassetto indica un bus di k linee. Le linee sottili sono segnali di controllo. In questo caso, ogni registro può essere sia sorgente che destinazione di una comunicazione.

I segnali di controllo in uscita dalla rete abilitano un' operazione di Write sui vari registri. Quelli in ingresso selezionano l'opportuna operazione di trasferimento.

Il problema in esame consiste nel progettare una rete che consenta il trasferimento del contenuto di **un registro sorgente** (Transmitter) su **uno o più registri destinazione** (Receivers). Il numero e la modalità dei trasferimenti viene specificato mediante istruzioni del tipo:

i. Rn→Rj,Rk,..Rm (i=0,n-1)

intendendo con ciò che, fra l'altro, la rete deve consentire di copiare il contenuto di Rn sui registri Rj,Rk,..Rm.

Come mostrato nella figura successiva, la rete deve includere k multiplexres n×1. I k MPX sono tutti controllati dagli stessi $\log_2 n$ segnali di controllo.
Il j-esimo MPX prende in ingresso i bit j-esimi di ciascuno degli n registri. L'input i-esimo (i=1..n) del j-esimo MPX è collegato al bit j-esimo (j=1..k) del registro Ri.
Quando i segnali di controllo in ingresso ai k MPX contengono il valore i (i=1..n), il contenuto dei k bit del registro Ri sarà disponibile sulle uscite Y1..Yk dei k MPX.

Il circuito combinatorio in basso a destra della figura 2 è un circuito che riceve in ingresso un codice indicante il tipo di trasferimento da effettuare, e produce in uscita:

- la selezione del registro Trasmitter (ottenuta agendo sui segnali di controllo dei MPX) e
- la selezione di uno o più Receivers, ottenuta portando al valore attivo i corrispondenti segnali di Write.

Il progetto di tale circuito dipende dal numero r tipo di trasferimenti che si vuole sia possibile effettuare.

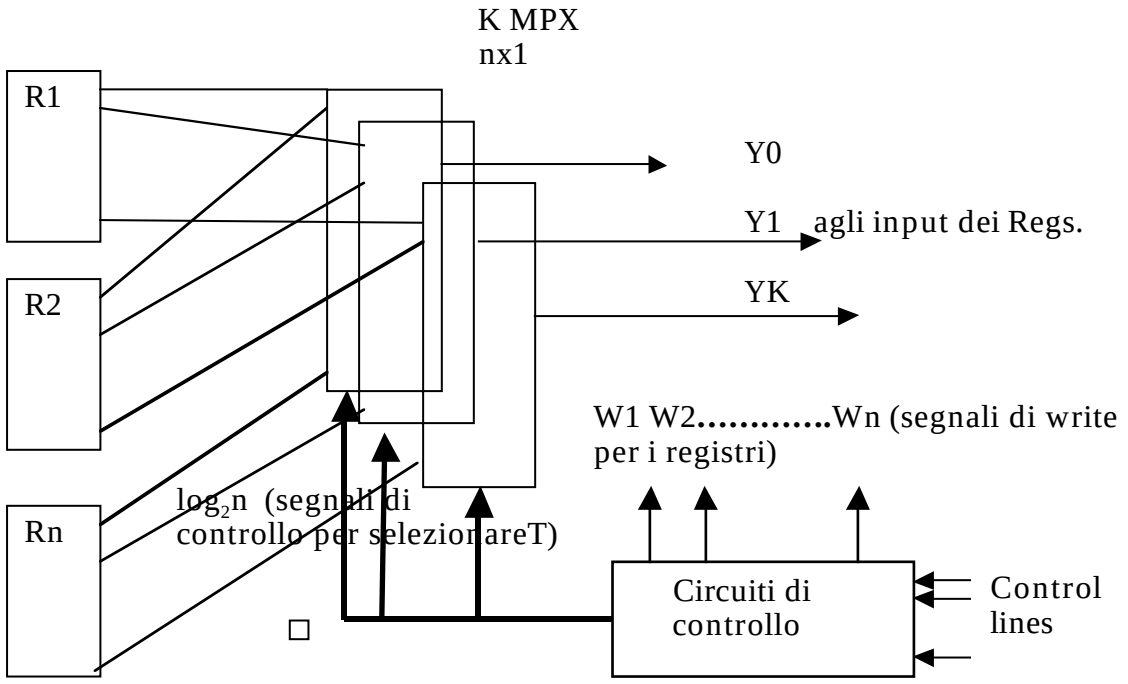



**Figura 2**

## Esempio 1

Supponiamo di avere 4 registri da 2 bit ciascuno, R1..R4. Supponiamo che si desideri rendere possibili i seguenti trasferimenti:

1. R1ŁR2
2. R4ŁR3,R1
3. R2ŁR3
4. R3ŁR4,R1

Per selezionare uno fra 4 possibili registri Transmitters occorrono 2 MPX a 4 ingressi. MPX1 riceverà in ingresso i bit $Q_0^i$ di ciascun registro (i=1..4), mentre MPX2 riceverà i bit $Q_1^i$.

Trattandosi di 4 possibilità di trasferimento, bastano due segnali di controllo, c0 e c1. Assegniamo la seguente codifica:

Trasferimento desiderato →codice identificativo

R1ŁR2 c0=0 c1=0
R4ŁR3,R1 c0=1 c1=1
R2ŁR3 c0=0 c1=1
R3ŁR4,R1 c0=1 c1=1

Possiamo progettare il circuito sulla base della seguente tabella di verità:

| Codice trasferimento c1c0 | Selezione del registro trasmettitore s1s0 (ai MPX) | Selezione dei registri riceventi W4W3W2W1 (segnali di write) |
|---|---|---|
| 00 | 00 (R1) | 0010 (write on R2) |
| 01 | 11 (R4) | 0101 (write on R3 and R1) |
| 10 | 01 (R2) | 0100 (write on R3) |
| 11 | 10 (R3) | 1001 (write on R4 and R1) |

Il passaggio dalla tabella di verità alle espresioni booleane (ed i circuiti) per s0,s1,W1,W2,W3 e W4 avviene secondo le modalità viste a lezione per il progetto di circuiti combinatori.

### 3.8.9. Trasferimenti paralleli

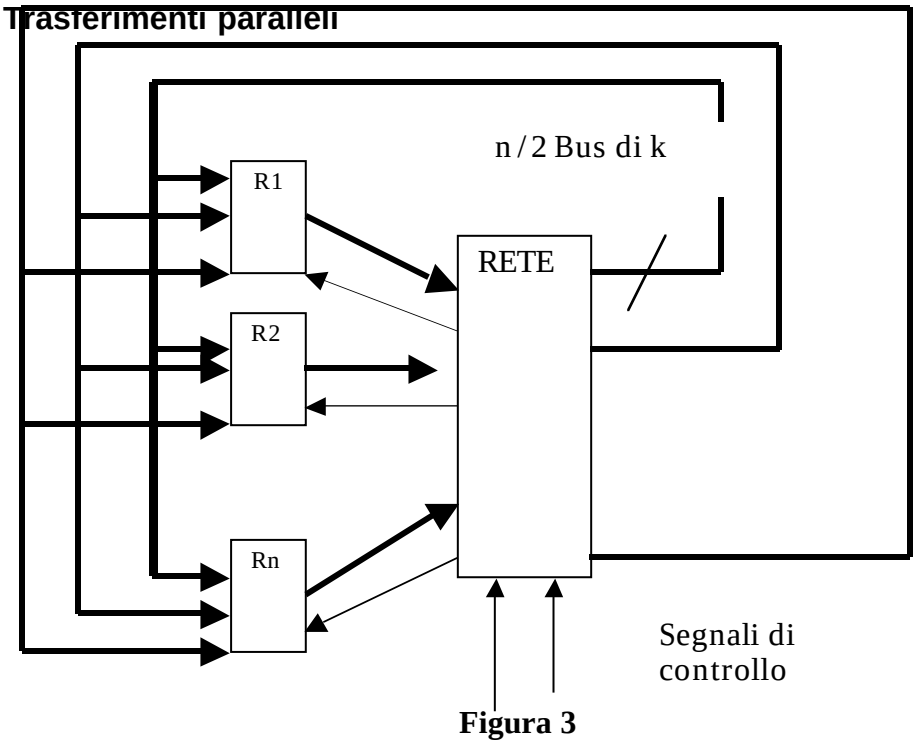



**Figura 3**

La rete in figura tre consente fino a n/2 trasferimenti in parallelo, ad esempio: R1ŁR2, R3ŁR4..Rn-1ŁRn.

In questo caso in uscita troviamo n/2 bus di k linee ciascuno (se i registri memorizzano k bit).
La rete sarà costituita da:

- n/2 gruppi Gj (j=1..n/2) di k MPX n×1.
- Ogni gruppo Gj è controllato dal medesimo insieme di $\log_2 n$ linee di controllo $(s_1^j s_2^j ....s_{log_2 n}^j)$, ma la selezione del trasmitter è indipendente per ogni gruppo Gj, ovvero in generale $(s_1^j s_2^j ....s_{log_2 n}^j) \neq (s_1^k s_2^k ....s_{log_2 n}^k)$ per i≠j

Ogni gruppo Gj è connesso come nella precedente figura 2: il MPX h-esimo del gruppo Gj riceve in ingresso gli h-esimi bit di tutti gli n registri.
Ogni FF h-esimo di ogni registro j-esimo avrà un ingresso siffatto:

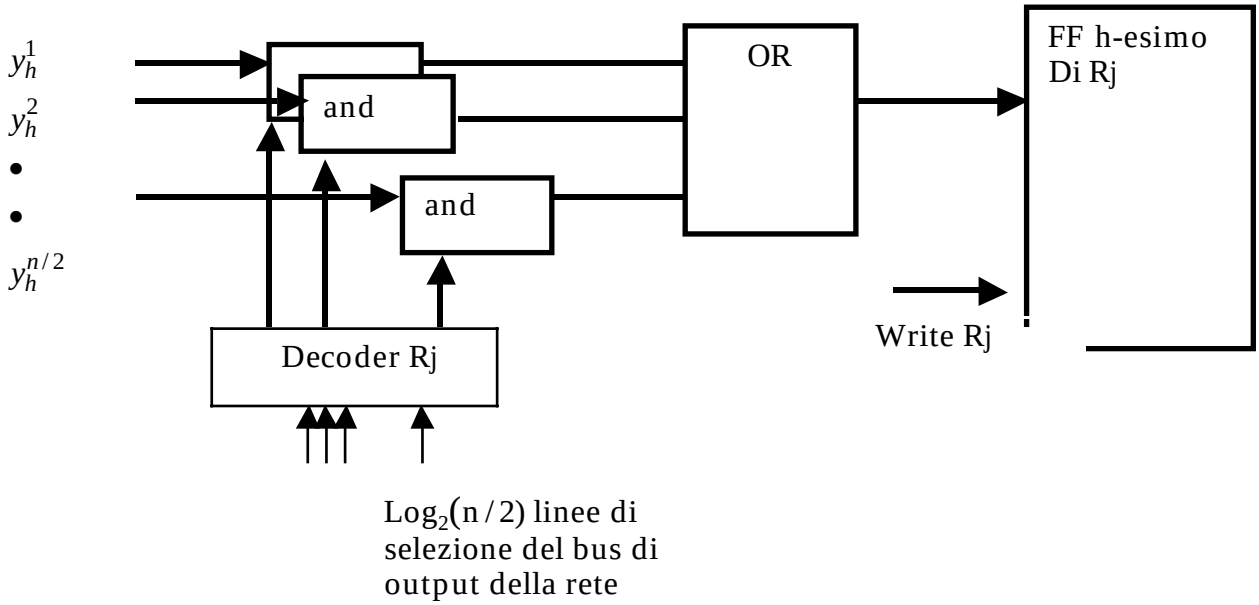



**Figura 4 Circuiti di selezione dell'input del FF h-esimo del registro j-esimo**

Come si vede il FF h-esimo potrà ricevere un input proveniente da una fra le uscite h-esime $y_h^k,\ k = 1..n/2$ di ciacuno degli n/2 bus di uscita della rete. Ovviamente, per ogni operazione di trasferimento, al più un input può essere selezionato, sulla base di segnali di controllo in ingresso al decoder, prodotti dal circuito combinatorio di controllo che, per ogni codifica di ingresso, deve selezionare al più n/2 coppie di Transmitters e Receivers.
In questo caso il progetto del circuito combinatorio è più complesso. Il circuito riceverà in ingresso un certo numero di segnali di controllo c1..cn, che codificano $2^n$ operazioni di trasferimento diverse. Ogni operazione può coinvolgere fino a n/2 trasferimenti paralleli. Le uscite prodotte dal circuito combinatorio devono selezionare, per ogni singolo trasferimento, l'opportuno trasmettitore e l'opportuno ricevitore.

**Esempio 2:**

Consideriamo ancora 4 registri da 2 bit ciascuno.
Supponiamo si debbano abilitare i seguenti trasferimenti:

- R1ŁR2, R3ŁR4
- R1ŁR4, R2ŁR3

La rete avrà due gruppi di MPX (n/2=2), ogni gruppo conterrà 2 MPX (k=2) avente 4 ingressi (n=4). Ogni gruppo di MPX sarà controllato da due linee di controllo, $\left\{s_1^1 s_2^1\right\},\left\{s_1^2 s_2^2\right\}$. Ci saranno due bus di uscita $\left\{y_0^1 y_1^1\right\}\left\{y_0^2 y_1^2\right\}$. Infine indichiamo con $\left\{r^1 r^2 r^3 r^4\right\}$, i segnali di selezione dell'input per ciascuno dei 4 registri. Occorre un solo segnale di controllo per ciascun registro, che ha valore zero se si vuole selezionare dal primo bus, valore 1 se si vuole selezionare dal secondo bus. Nel caso il registro non debba ricevere alcun input, il valore di $r^i$ non avrà importanza, poiché il segnale di Write per quel registro sarà posto a zero.
Dunque basta 1 bit di controllo, C, per codificare uno di questi due comandi.

| Trasferimenti da abilitare | Selezione Transmitters | Selezione input registri | Segnali di scrittura |
|---|---|---|---|
| C | $\left\{s_1^1 s_2^1\right\},\left\{s_1^2 s_2^2\right\}$ | $\left\{r^1 r^2 r^3 r^4\right\}$ | W1 W2 W3 W4 |
| C=0 (R1ŁR2, R3ŁR4) | 00 10 (R1 e R3) | X 0 X 1 | 0 1 0 1 |
| C=1 (R1ŁR4, R2ŁR3) | 00 01 (R1 e R2) | X X 1 0 | 0 0 1 1 |

Naturalmente, un progetto "ad-hoc" della rete di selezione degli input per i vari registri avrebbe semplificato molto il circuito. Infatti, R1 non è mai Receiver, R2 riceve solo da R1 se C=0, R3 solo da R2 se C=1, R4 da R3 se C=0 e da R1 se C=1.