Realizzazione di FSM sincrone

Architettura degli elaboratori

M. Favalli

Engineering Department in Ferrara

Sommario

Progettazione del STG

Realizzazione (sincrona) del STG

Il problema della codifica dello stato

Temporizzazioni nelle reti sincrone

Sommario

Progettazione del STG

Realizzazione (sincrona) del STG

Il problema della codifica dello stato

Temporizzazioni nelle reti sincrone

Introduzione

- **Passi nel progetto di un sistema digitale basato su macchine a stati finiti di tipo sincrono:**
  1. **traduzione delle specifiche informali in specifiche formali (STG o tabella)**
  2. **ottimizzazione della descrizione formale**
  3. **scelta della codifica dello stato**
  4. **realizzazione della rete combinatoria che realizza le funzioni di uscita e stato futuro, dimensionamento del segnale di clock**
- **Questi passi sono tipicamente seguiti da un passo di verifica**

Realizzazione del grafo di transizione dello stato

- **É la parte piú difficile in quanto, partendo da un modello informale, non esiste un approccio sistematico**
- **Si forniscono quindi solo una serie di suggerimenti per aiutare il progettista, il piú importante dei quali riguarda la corretta identificazione delle informazioni che lo stato deve contenere al fine di poter calcolare l'uscita**
- **Questo passo consente di evitare l'errore piú diffuso ovvero il considerare le diverse possibilitá per la sequenza di ingresso aggiungendo tipicamente stati inutili**

Scelta dello stato di partenza

- **In alcuni casi si puó identificare una condizione di funzionamento di riposo indipendente dalla storia precedente del sistema**
- **In altri casi possono esistere piú condizioni di funzionamento che dipendono in maniera molto semplice dalla storia del sistema**
- **Se ci si trova in difficoltá, si puó cercare di identificare una possibile storia del sistema che porta univocamente a una qualsiasi condizione di funzionamento**

Sequenza principale

- **In molti casi, come quello degli automi che riconoscono sequenze di simboli, si puó identificare una sequenza principale di stati che portano al riconoscimento di tale simbolo**
- **In questi casi si puó focalizzare l'attenzione su tale sequenza completando poi in seguito l'automa**
- **In alcuni casi si arriva alla sequenza principale tramite una sequenza di attivazione**

Macchine in cui lo stato dipende da un numero finito di valori dell'ingresso

- **In generale si puó esplicitare lo stato futuro in funzione di una sequenza di j ingressi e di uno stato iniziale**
- **Per fare questo si parte dalla relazione di stato futuro $s_{k+1} = \delta(s_k, x_k)$ e si esplicitano recursivamente gli stati presenti: $s_k = \delta(s_{k-1}, x_k - 1)...$:, da cui $s_{k+1} = \sigma(s_{k-j}, x_k, x_{k-1}, ..., x_{k-j})$**
- **Se esiste un j per cui scompare la dipendenza dallo stato s_{k-j}, posso utilizzare 2^j stati ciascuno per ogni configurazione di $x_{k-1}, ..., x_{k-j}$**
- **Questo approccio puó portare a un numero di stati ridondanti, ma é utile in presenza di specifiche che chiedono di analizzare gli ultimi $j + 1$ simboli di ingresso (Mealy)**

Esempi

Regole

Realizzazione di reti sincrone correttamente funzionanti

- **Non devono comparire stati non raggiungibili**

  lo stato D é non raggiungibile

	0	**1**
A	$A, 0$	$B, 0$
B	$C, 0$	$C, 1$
C	$A, 0$	$A, 1$
D	$D, 0$	$D, 1$

- **Non devono comparire stati assorbenti**

  non si puó uscire dallo stato D

o	**0**	**1**
A	$A, 0$	$B, 0$
B	$D, 0$	$C, 0$
C	$D, 0$	$A, 1$
D	$D, 0$	$D, 0$

Automa minimo

- Nel progetto di una FSM possono comparire alcuni stati ridondanti
- Alcuni sottoinsiemi di stati possono riassumere la stessa storia passata del sistema
- Per ogni sequenza di ingresso, l'evoluzione dello stato e le uscite della FSM a partire da tali stati sono le stesse
- Poiché il numero di stati si riflette sul numero di FF da utilizzare e sulla complessitá della rete combinatoria che realizza λ e δ, conviene sostituire ciascuno di tali sottoinsiemi di stati con un un singolo stato
- Si puó arrivare a un automa minimo
- Non vedremo i metodi utilizzati per ottenere questo risultato, ma solo un esempio

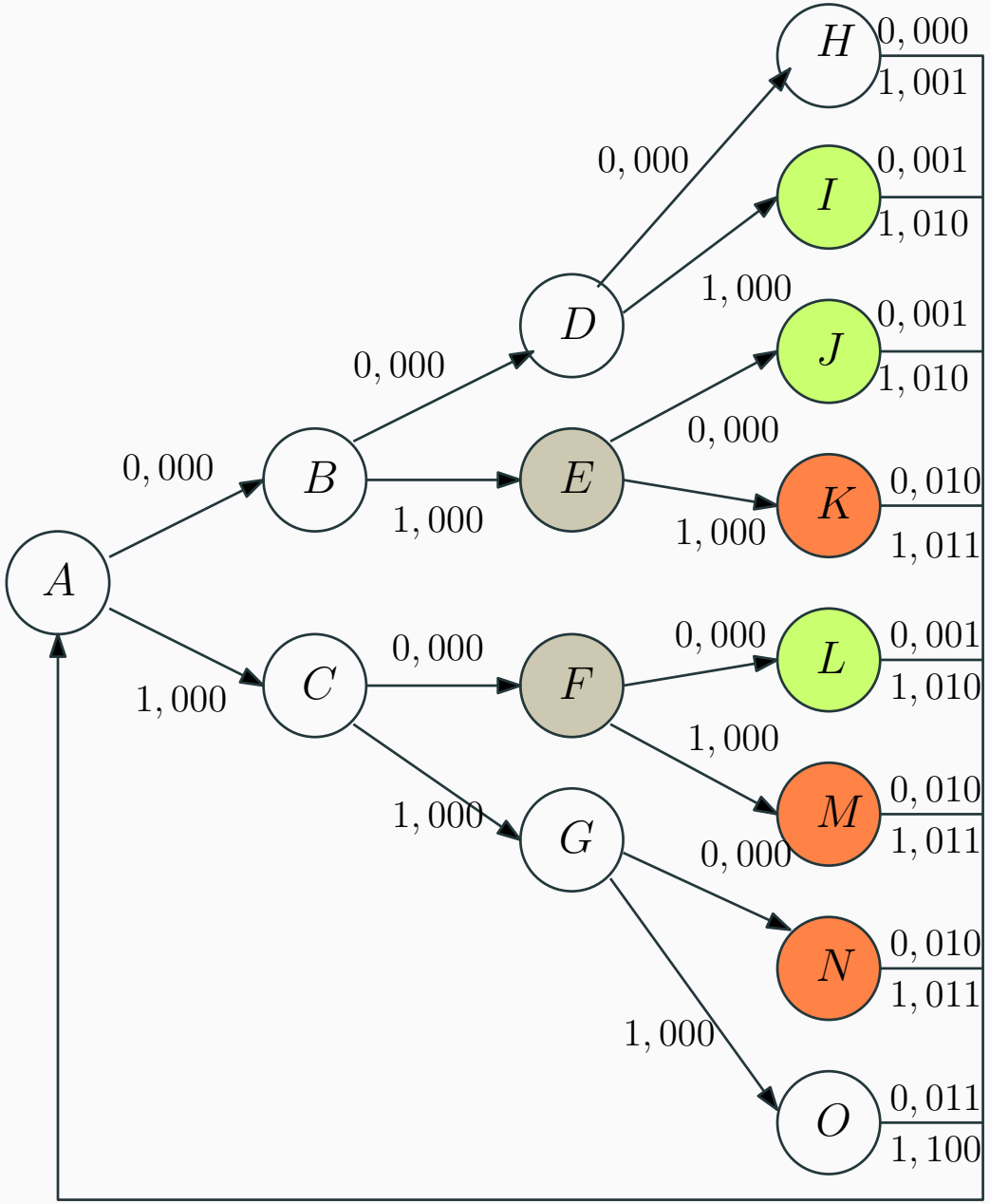
Esempio

Si disegni una FSM che riceve serialmente su un ingresso x parole di 4 bit, compito della rete é produrre sulle sue 3 uscite $z_2 z_1 z_0$ la codifica binaria del numero di 1 ricevuti in una parola. tale codifica deve essere prodotta sul quarto bit ricevuto (Mealy), mentre la riceve i 3 bit precedenti le uscite valgono 0.

x	....	0	1	0	0	1	0	1	0 ....
z_2	....	0	0	0	0	0	0	0	0 ....
z_1	....	0	0	0	0	0	0	0	1 ....
z_0	....	0	0	0	1	0	0	0	0 ....

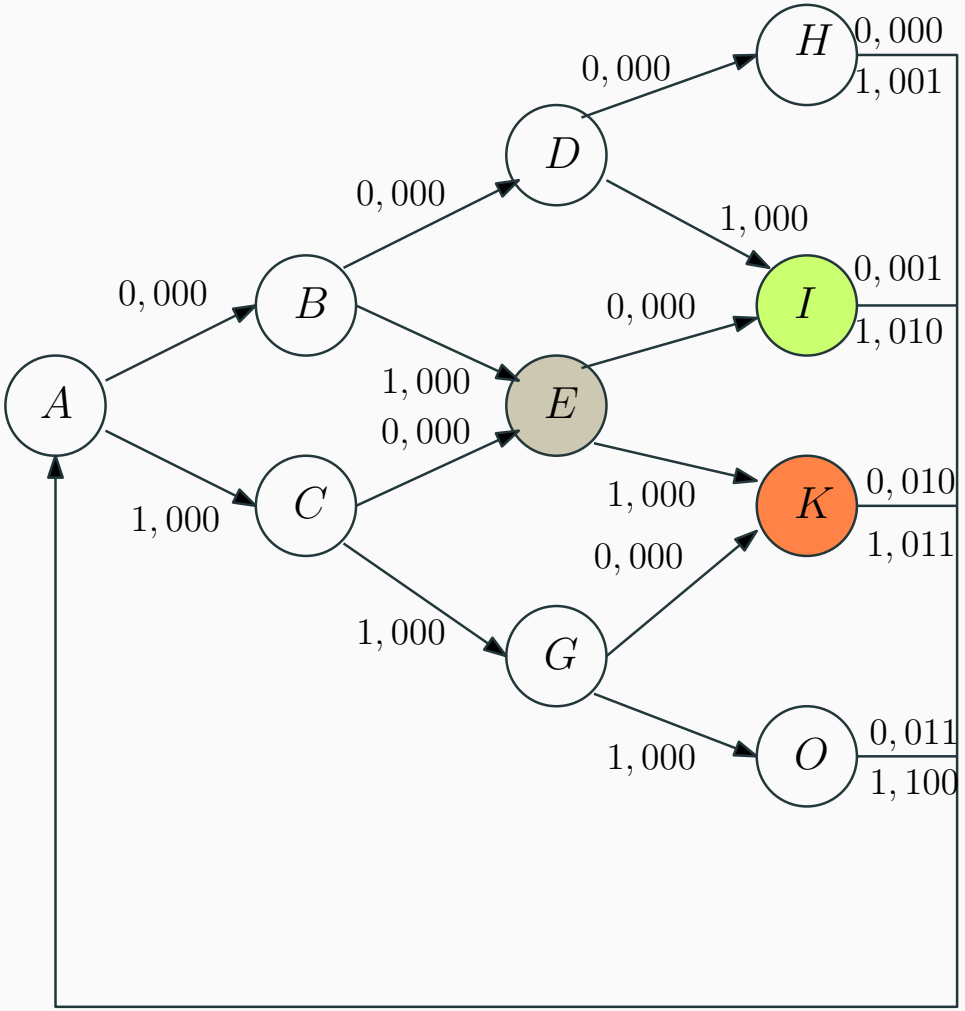
Esempio

- Gli stati con lo stesso colore riassumono la stessa storia passata del sistema
- A partire da essi, per ogni sequenza di ingresso l'uscita Íla stessa
- Possono essere accorpati in un unico stato



Esempio

FSM ridotta sulla base di tali considerazioni, la FSM ha il minimo numero di stati



Sommario

Introduzione

- **Una volta ottenuto l'automa minimo o comunque un automa soddisfacente si passa alla realizzazione dell'automa come rete sequenziale sincrona**
- **I simboli di ingresso ($\mathcal{X}$), di uscita ($\mathcal{Z}$) e gli stati ($\mathcal{S}$) devono essere codificati con un codice binario**
- **Le funzioni λ e δ diventano funzioni dell'algebra di commutazione (Λ e Δ) che vengono poi realizzate tramite reti combinatorie**
- **La memoria in retroazione che garantisce la trasformazione dello stato futuro in stato presente a ogni istante di sincronizzazione viene realizzata tramite flip-flop o latch controllati dal segnale di clock**

Obbiettivi della codifica

- **Costo (costo della rete combinatoria e dei flip-flop o latch)**
- **Prestazioni**
- **Consumo di potenza**
- **Affidabilitá**

Questi obbiettivi possono essere raggiunti mediante opportune tecniche euristiche

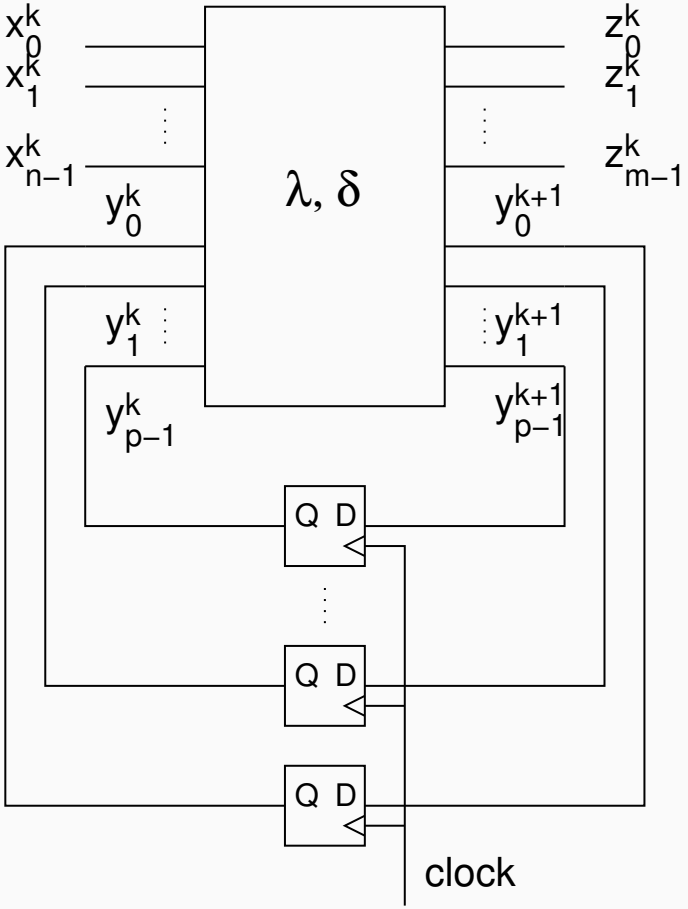
Ipotesi

- **Supponiamo che la codifica binaria dei simboli di ingresso e di uscita sia assegnata**
- **Per motivi di tempo supponiamo di utilizzare una codifica casuale per lo stato**
- **Per la retroazione si utilizzano flip-flop di tipo D**

Codifica

- **Codifica dei simboli di ingresso:** $f_{in} \ : \ \mathcal{X} \rightarrow \{0,1\}^n, \ 2^n \geq |\mathcal{X}|$
- **Codifica dei simboli di uscita:** $f_{out} \ : \ \mathcal{Z} \rightarrow \{0,1\}^m, \ 2^m \geq |\mathcal{Z}|$
- **Codifica degli stati:** $f_{state} \ : \ \mathcal{S} \rightarrow \{0,1\}^p, \ 2^p \geq |\mathcal{S}|$
- **Variabili di ingresso:** $\{x_0, x_1, ..., x_{n-1}\}$
- **Variabili di uscita:** $\{z_0, z_1, ..., z_{m-1}\}$
- **Variabili di stato:** $\{y_0, y_1, ..., y_{p-1}\}$

Modello di riferimento (Huffman)



Rete sequenziale sincrona

- **Sostituendo i simboli di ingresso, uscita e stato che compaiono nella tabella di transizione dello stato con le rispettive codifiche si ottiene la tabella delle transizioni codificata**
- **Tale tabella costituisce la tabella di veritá per le funzioni Λ e Δ (si considera un automa di Mealy):**
  - $\lambda \Rightarrow \Lambda : \{0,1\}^{n+p} \rightarrow \{0,1\}^{m}$**, che per la *i*-ma variabile di uscita all'istante *k* fornisce:**

$$z_i^k = \Lambda_i(x_0^k, x_1^k, ..., x_{n-1}^k, y_0^k, y_1^k, ..., y_{p-1}^{k+1})$$

  - $\delta \Rightarrow \Delta : \{0,1\}^{n+p} \rightarrow \{0,1\}^{p}$**, che per l'*i*-ma variabile di stato futuro all'istante *k* fornisce:**

$$y_i^{k+1} = \Delta_i(x_0^k, x_1^k, ..., x_{n-1}^k, y_0^k, y_1^k, ..., y_{p-1}^{k+1})$$

Retroazione

- **Per garantire il corretto funzionamento di una rete sincrona, bisogna che le variabili di stato rimangano stabili per un periodo di clock**
- **Lo stato corrente é memorizzato all'interno dei FF messi in retroazione**
- **Le uscite della rete combinatoria dipendono dal tipo di FF usati (ne esistono di diverso tipo oltre ai D)**
- **Nel caso di FF di tipo D, l'equazione caratteristica é: $Q^{k+1} = D^k$ e quindi Δ fornisce direttamente gli ingressi dei FF**
- **I FF possono sia essere visti come elementi che memorizzano lo stato, sia come elementi che garantiscono il corretto sequenziamento delle operazioni**

Sommario

Progettazione del STG

Realizzazione (sincrona) del STG

Il problema della codifica dello stato

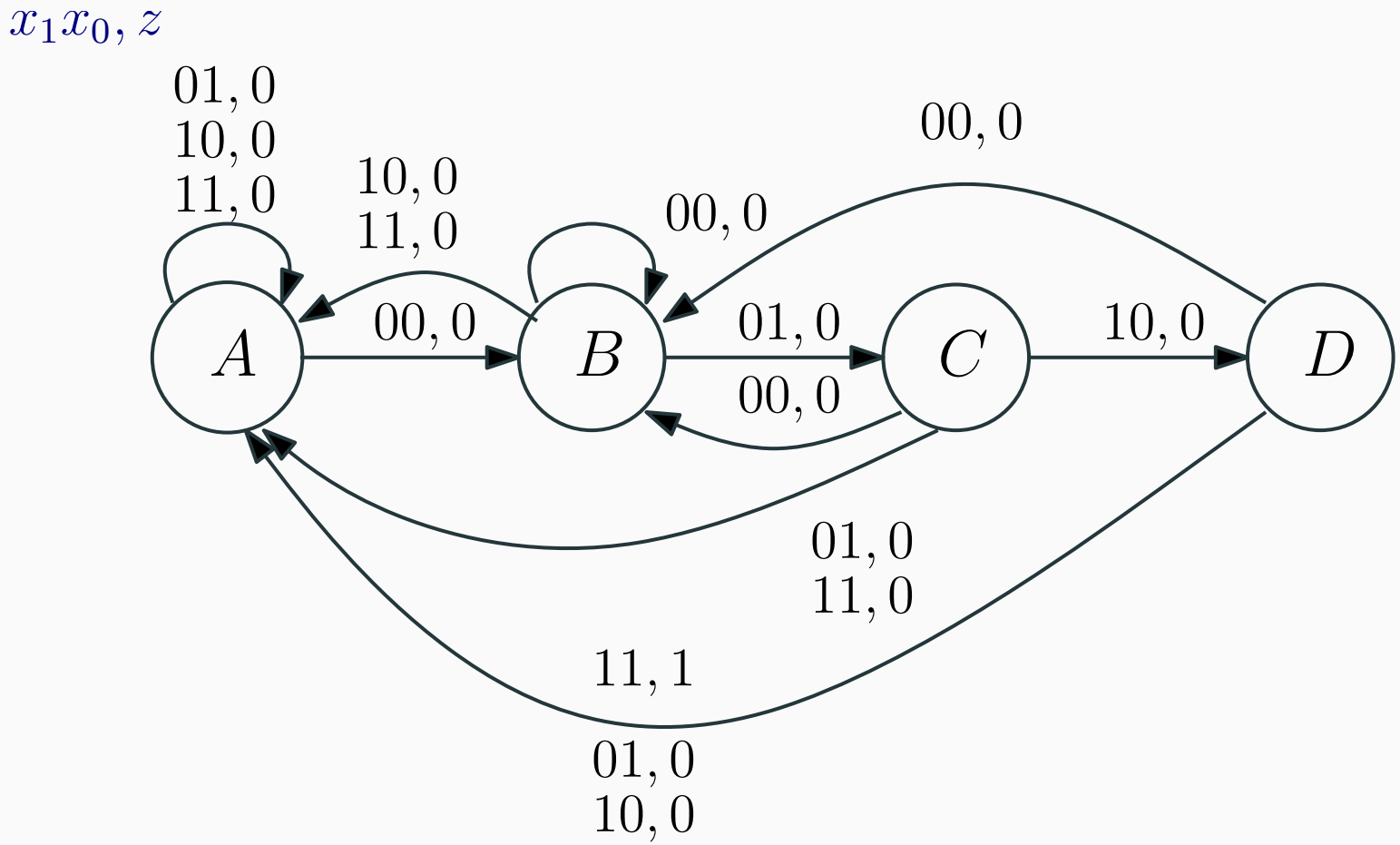
Temporizzazioni nelle reti sincrone

Codifica casuale

- **Se si fa l'ipotesi che il costo della rete sia proporzionale al numero di variabili di stato (questa ipotesi in diversi casi non é verificata), conviene usarne il minimo numero:** $p = \lceil log_2 |\mathcal{S}|$
- **Nella codifica casuale si puó semplicemente assegnare una configurazione binaria scelta a caso a ciascuno degli stati**
- **Esistono codifiche ottimizzate e codifiche in cui si usa un numero di variabili di stato maggiore di quello minimo**
- **Un esempio é dato dalla codifica 1-out-of-n in cui si usa un FF per ogni stato**

Esempio

STG di una FSM che ha due ingressi $x_1 x_0$ che codificano un numero naurale e che produce 1 in uscita (z) se riconosce la sequenza 0, 1, 2, 3



Esempio

s^k	$(x_1 x_0)^k$ **00**	**01**	**11**	**10**
A	$B, 0$	$A, 0$	$A, 0$	$A, 0$
B	$B, 0$	$C, 0$	$A, 0$	$A, 0$
C	$B, 0$	$A, 0$	$A, 0$	$D, 0$
D	$B, 0$	$A, 0$	$A, 1$	$A, 0$

Tabella di transizione dello stato

stato	$y_1 y_0$
A	**00**
B	**01**
C	**11**
D	**10**

Codifica

$(y_1 y_0)^k$	$(x_1 x_0)^k$ **00**	**01**	**11**	**10**
00	$01, 0$	$00, 0$	$00, 0$	$00, 0$
01	$01, 0$	$11, 0$	$00, 0$	$00, 0$
11	$01, 0$	$00, 0$	$00, 0$	$10, 0$
10	$01, 0$	$00, 0$	$00, 1$	$00, 0$

Tabella delle transizioni

Esempio

Dalla tabella delle transizioni si ottengono quindi le funzioni eccitazione per le 2 variabili di stato e la funzione per l'uscita

y_1^{k+1}

$(y_1 y_0)^k$	$(x_1 x_0)^k$ **00**	**01**	**11**	**10**
00	0	0	0	0
01	0	1	0	0
11	0	0	0	1
10	0	0	0	0

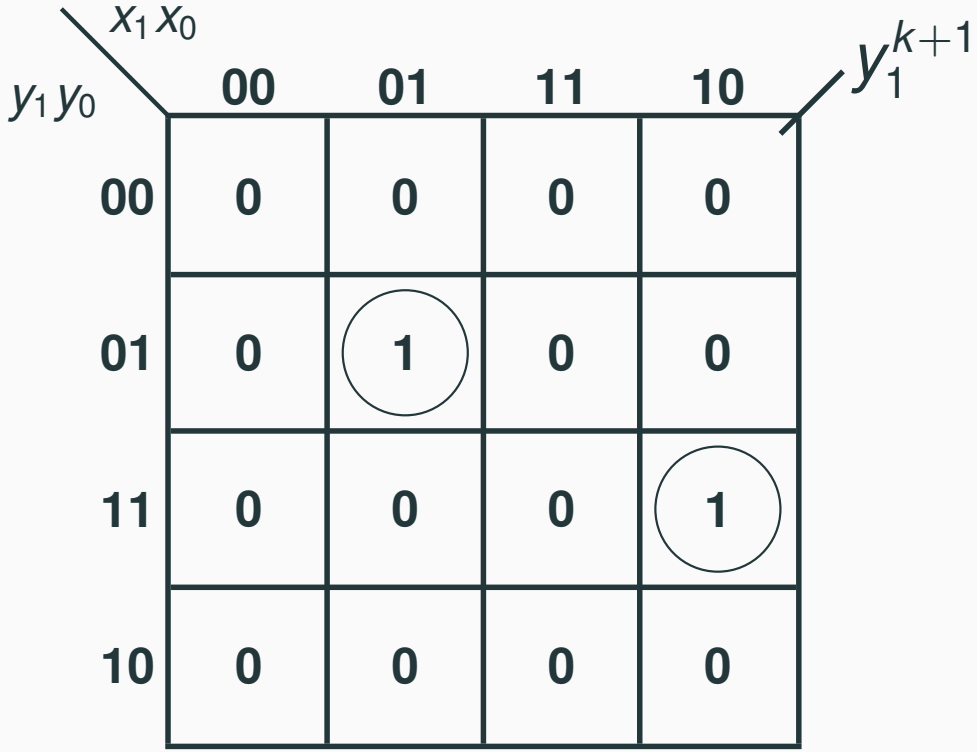
y_0^{k+1}

$(y_1 y_0)^k$	$(x_1 x_0)^k$ **00**	**01**	**11**	**10**
00	1	0	0	0
01	1	1	0	0
11	1	0	0	0
10	1	0	0	0

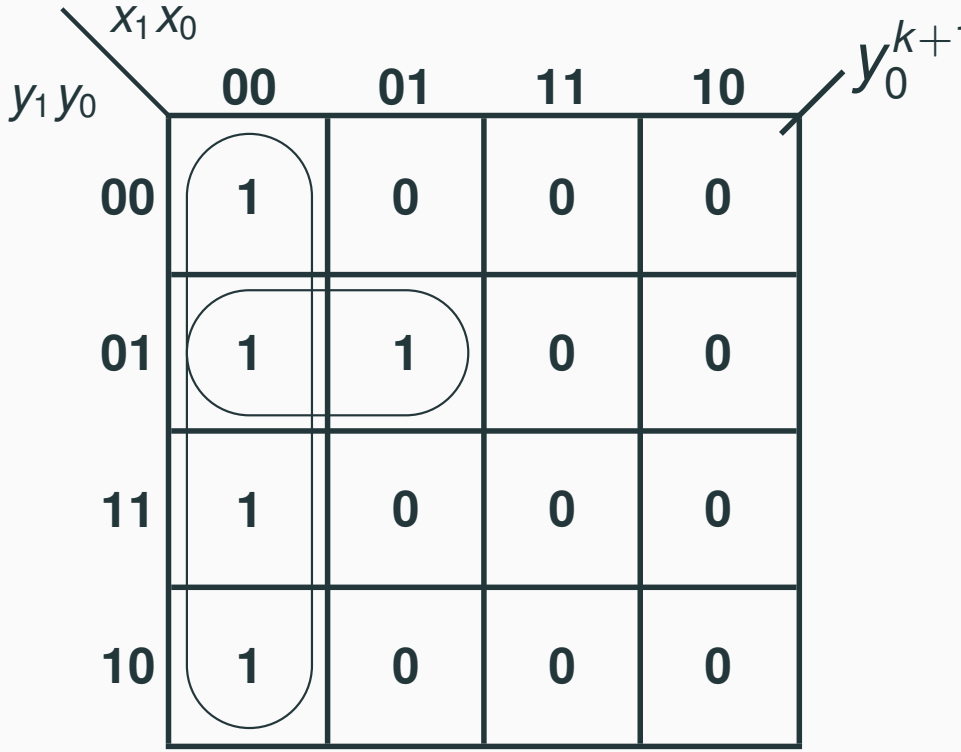
z_0^k

$(y_1 y_0)^k$	$(x_1 x_0)^k$ **00**	**01**	**11**	**10**
00	0	0	0	0
01	0	0	0	0
11	0	0	0	0
10	0	0	1	0

Funzioni di stato futuro (δ)

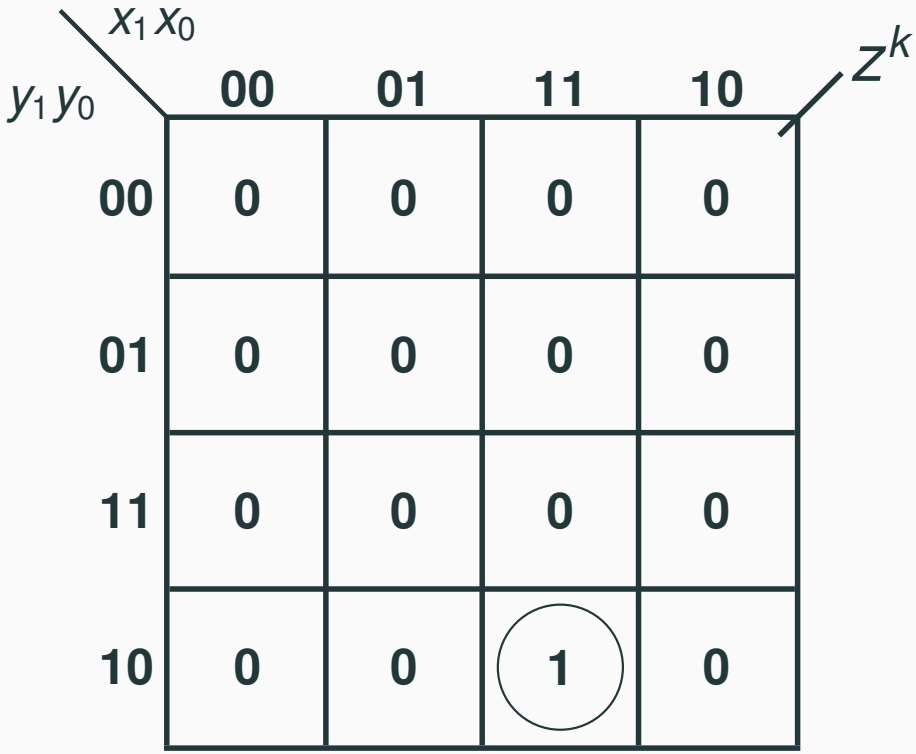


$$y_1^{k+1} = (x_1' x_0 y_1' y_0 + x_1 x_0' y_1 y_0)^k$$



$$y_0^{k+1} = (x_1' x_0' + x_1' y_1' y_0)^k$$

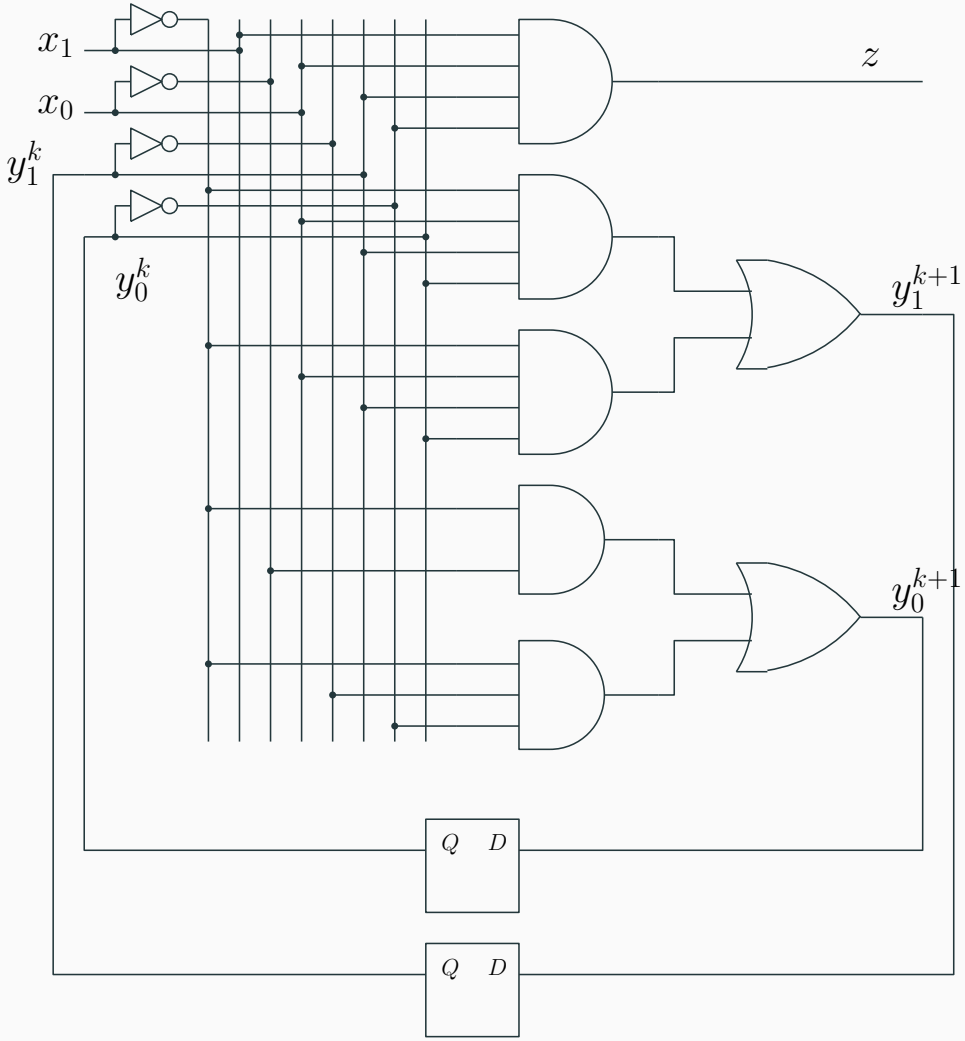
Funzione di uscita (λ)



$$z_0^k = (x_1 x_0 y_1 y_0')^k$$

Rete logic sincrona

Schema logico della rete che realizza la FSM



Ottimizzazione della codifica

- **Se nell'esempio precedente si fosse scelta una codifica diversa, probabilmente si sarebbe ottenuta una rete combinatoria con un costo diverso**
- **Nasce quindi il problema di selezionare la codifica che da luogo alla rete con il costo minore**
- **Dato il numero molto grande di possibili codifiche, esistono delle tecniche euristiche per la scelta della codifica (che non verranno esaminate)**

Problemi

- **Se $2^p > |\mathcal{S}|$ si avranno alcuni stati che esistono nell'automa codificato, ma non in quello di partenza**
- **Nella tabella di transizione dello stato saranno presenti delle indifferenze su stato futuro**
- **Tali indifferenze possono poi essere assegnate durante il processo di sintesi**
- **L'automa risultante risulterá diverso da quello di partenza**
- **É necessaria un operazione di analisi per capire se esistono problemi (stati irrangiugibili, componenti non connesse)**

Il problema dello stato iniziale

- Fino a questo momento non ci siamo posti il problema dello stato iniziale di una FSM
- Il problema é dato dal fatto che il dispositivo fisico a un certo punto viene acceso e i FF si portano in uno stato che puó essere incognito
- Ci sono FSM che funzionano correttamente per ogni possibile stato iniziale, altre che hanno sequenza di inizializzazione che porta a uno stato noto e altre che non hanno queste caratteristiche
- Il problema viene risolto aggiungendo un ingresso di reset ai FF in modo da portare la rete in uno stato iniziale noto
  - il reset puó essere aggiunto alla FSM a livello di STG oppure
  - puó essere aggiunto direttamente alla rete sincrona portando tale segnale in ingresso ai FF

Analisi di una rete sequenziale sincrona

- **Passaggio da una descrizione livello gate al STG**
- **Nel caso di reti molto semplici il problema puó essere risolto per via grafica**
  1. **determinare le espressioni SP di uscite e variabili di stato futuro**
  2. **tracciare tali espressioni su mappe di Karnaugh**
  3. **riunire tali mappe in unica mappa che rappresenta la tabella delle transizioni codificata**
  4. **associare un nome simbolico a ciascuno stato e ottenere quindi la tabella di transizione dello stato**
  5. **disegnare il STG**

Esempi di problemi dovuti alla codifica dello stato

Sommario

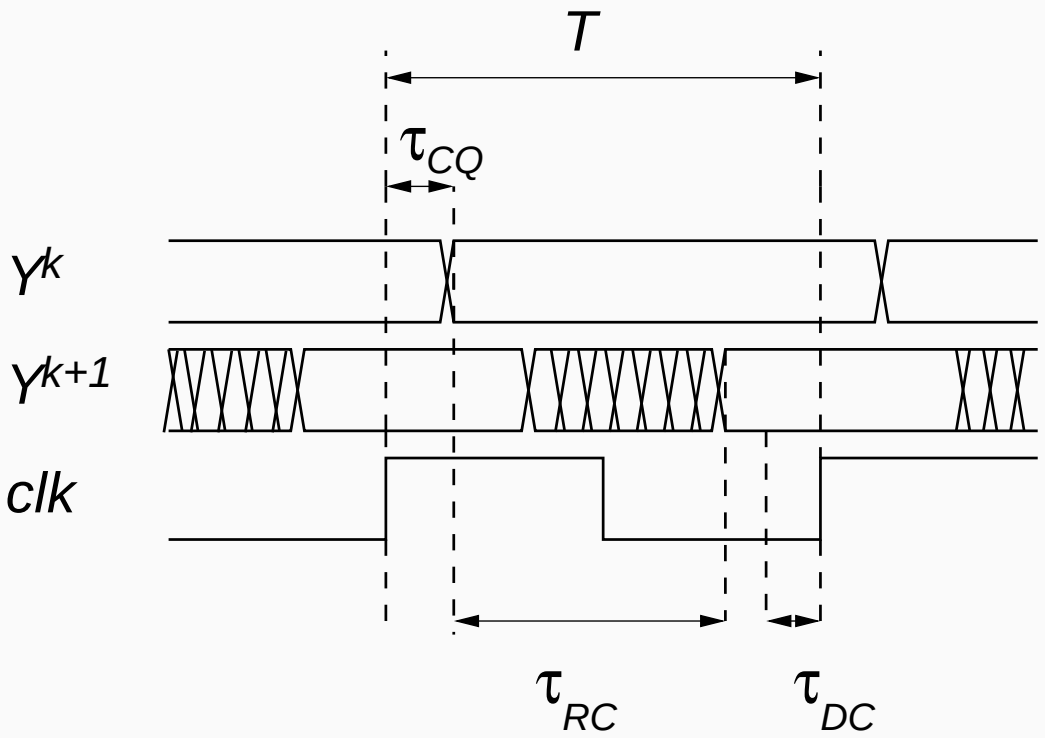
Condizioni per il campionamento

- **Una rete sequenziale (Huffman) si comporta come sincrona e in maniera aderente alle specifiche se le uscite della rete combinatoria vengono campionate quando tutti i transitori si sono esauriti**
- **Questo comportamento deve essere garantito dalla scelta di un valore opportuno del periodo di clock (T) che dipende dal ritardo della rete combinatoria e dai parametri dei FF**
- **Si utilizza l'ipotesi che il cambiamento degli ingressi sia sincrono con il periodo di clock e che le uscite siano anch'esse campionate dal segnale di clock**

Scelta del periodo di clock

T minimo

$T > \tau_{CQ} + \tau_{RC} + \tau_{DC}$

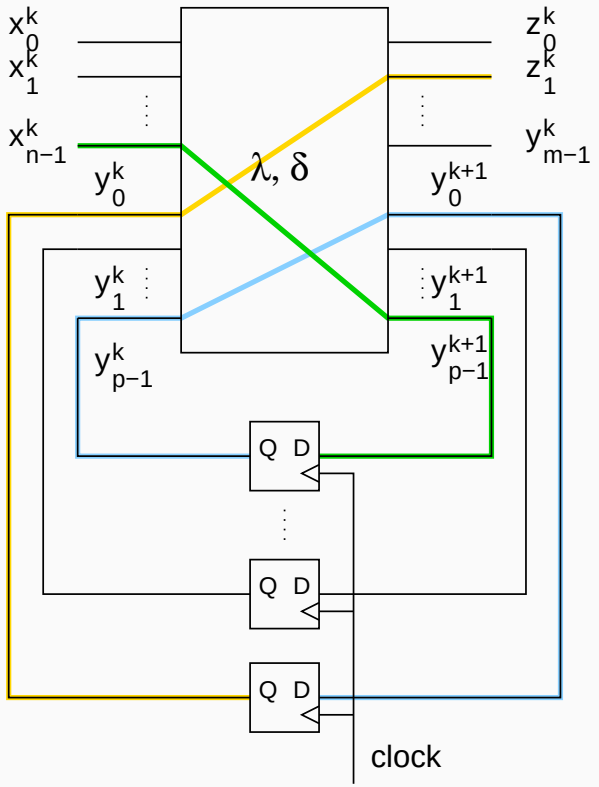


τ_{RC} é il ritardo massimo della rete combinatoria

τ_{CQ} e τ_{DC} sono il tempo di risposta e di setup dei FF

Ritardo massimo della rete combinatoria

- **Per determinare τ_{RC} si devono considerare i diversi possibili cammini per la propagazione di una transizione da un ingresso della RC (x_i^k o y_i^k) a una sua uscita (z_i^k o y_i^{k+1})**
- **Questi cammini si possono attivare dipendentemente dalle configurazioni in ingresso alla RC**
- **τ_{RC} deve essere determinato in condizioni di caso peggiore**



Sommario

Passi nella sintesi di una FSM

1. **Interpretazione delle specifiche**
2. **Progetto del STG**
3. **Ottimizzazione del numero degli stati**
4. **Codifica dello stato (con eventuali euristici per la scelta del codice)**
5. **Sintesi della rete combinatoria che realizza λ e δ**
6. **Calcolo della frequenza di clock**