

Sintesi Sequenziale Sincrona

Sintesi comportamentale di reti sequenziali sincrone

Reti Logiche A

Sommario

- Introduzione
- Sintesi comportamentale
- Architettura generale
- Tabella degli stati
- Diagramma degli stati
- Riduzione del numero degli stati
- Codifica degli stati

Reti Logiche A

Introduzione

- Progetto di reti combinatorie
 - I metodi sono noti e ben assestati
 - Si ricercano nuove soluzioni che aumentino l'efficienza:
 - Computazionale degli strumenti automatici
 - Nell'uso delle risorse (es. BDD)
 - Metodologica (es. *Signature Cube* - '92)
- Progetto di reti sequenziali
 - L'ottimizzazione di circuiti sequenziali è in costante evoluzione
 - Esistono buoni metodi ma non di uso generale
 - Lo sviluppo di software efficienti necessita di ulteriori sforzi

Reti Logiche A

Introduzione

- Il modello di un circuito sincrono può essere:
- Comportamentale
 - La transizione degli stati è descritta in termini di tabelle o diagrammi
 - Le informazioni sugli stati sono esplicite
 - Le informazioni sull'area e sui ritardi sono implicite
- Strutturale
 - Il modello del circuito è una *netlist* ovvero un insieme di componenti, registri e logica combinatoria, collegati tra loro
 - Le informazioni sugli stati sono implicite
 - Le informazioni sull'area e ritardi sono esplicite

Reti Logiche A

Sintesi comportamentale

- Il valore delle uscite all'istante t dipende dalla successione degli ingressi che precedono l'istante t
- Ciò implica il concetto di *stato*
- Una macchina sequenziale è definita dalla quintupla $(I, U, S, \delta, \lambda)$
 - I - Alfabeto di Ingresso
 - E' costituito dall'insieme *finito* dei *simboli* di ingresso
 - Con n linee di ingresso si hanno 2^n simboli
 - U - Alfabeto d'Uscita
 - E' costituito dall'insieme *finito* dei *simboli* d'uscita
 - Con m linee d'uscita si hanno 2^m simboli.
 - S - Insieme degli Stati
 - Insieme *finito* e *non vuoto* degli *stati*
 - δ - Funzione stato prossimo
 - λ - Funzione d'uscita

Reti Logiche A

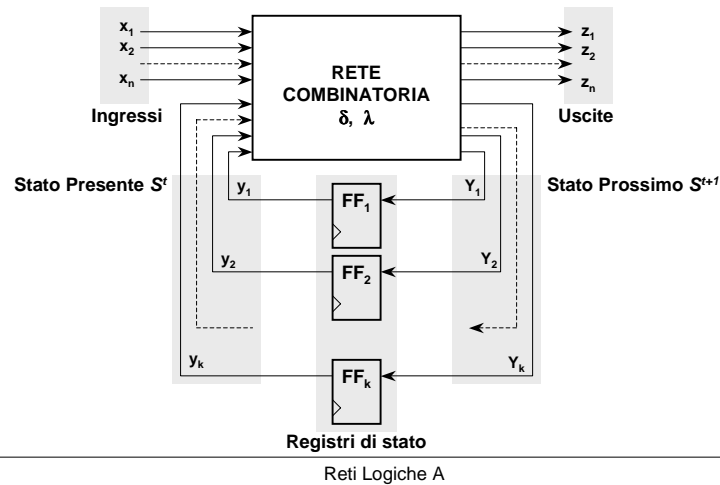
Sintesi comportamentale

- Funzione stato prossimo δ
 - Ad ogni stato presente e per ogni simbolo di ingresso la funzione δ associa uno stato futuro:
$$\delta : S \times I \rightarrow S$$
 - Ad ogni coppia $\{\text{stato}, \text{simbolo di ingresso}\}$ è associato, se specificato, uno ed uno solo stato futuro.
- Funzione d'uscita λ
 - Genera il simbolo d'uscita
 - Macchine di Mealy. L'uscita dipende sia dallo stato sia dall'ingresso:
$$\lambda : S \times I \rightarrow U$$
 - Macchine di Moore. L'uscita dipende solamente dallo stato:
$$\lambda : S \rightarrow U$$

Reti Logiche A

Architettura generale

- La struttura generale di una macchina sequenziale è la seguente:



Architettura generale

- Il problema della sintesi comportamentale di una rete sequenziale consiste nella:
 - Identificazione delle le funzioni δ e λ
 - Sintesi della rete combinatoria che le realizza
- Gli elementi di memoria sono costituiti da bistabili
- I flip-flop di tipo D sono quelli usati più comunemente
- La funzione di stato prossimo dipende dal tipo di bistabili utilizzati
- La funzione di uscita è indipendente dal tipo di bistabili utilizzati

Tabella degli stati

- Una FSM può essere descritta mediante la *Tabella degli stati*
- Indici di colonna sono i simboli di ingresso $i_\alpha \in I$
- Indici di riga sono i simboli di stato $s_j \in S$ che indicano lo stato presente
- Elementi sono:
 - Macchine di Mealy: La coppia $\{u_\beta, s_j\}$:
 - $u_\beta = \lambda(i_\alpha, s_j)$ è il simbolo di uscita
 - $s_j = \delta(i_\alpha, s_j)$ è il simbolo stato prossimo
 - Macchine di Moore: Il simbolo stato prossimo s_j :
 - $s_j = \delta(i_\alpha, s_j)$ è il simbolo stato prossimo
- Nelle macchine di Moore i simboli d'uscita sono associati allo stato presente

Reti Logiche A

Tabella degli stati

- Macchine di Mealy

	i_1	i_2	..
s_1^t	s_1^{t+1} / u_j	s_k^{t+1} / u_k
s_2^t	s_m^{t+1} / u_m	s_l^{t+1} / u_l
..

- Macchine di Moore

	i_1	i_2	..	
s_1^t	s_1^{t+1}	s_k^{t+1}	u_1
s_2^t	s_m^{t+1}	s_l^{t+1}	u_2
..

Reti Logiche A

Diagramma degli stati

- Spesso, la stesura della *Tabella degli stati* è preceduta da una rappresentazione grafica ad essa equivalente, denominata *Diagramma degli stati*
- Il Diagramma degli stati è un *grafo orientato* $G(V,E,L)$
 - V - Insieme dei *nodi*
 - Ogni nodo rappresenta uno stato
 - Ad ogni nodo è associato un simbolo d'uscita (macchine di Moore)
 - E - Insieme degli *archi*
 - Ogni arco rappresenta le transizioni di stato
 - L - Insieme degli:
 - Ingressi e Uscite (macchine di Mealy)
 - Ingressi (macchine di Moore)

Reti Logiche A

Esempio – Macchina di Mealy

- Questo esempio mostra l'equivalenza delle due rappresentazioni nel caso di una macchina di Mealy

Diagramma degli stati

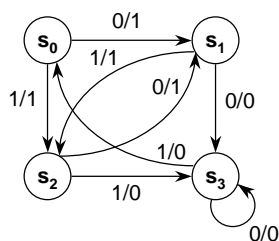


Tabella degli stati

	0	1
S ₀	S ₁ /1	S ₂ /1
S ₁	S ₃ /0	S ₂ /1
S ₂	S ₁ /1	S ₃ /0
S ₃	S ₃ /1	S ₀ /0

Reti Logiche A

Esempio – Macchina di Moore

- Questo esempio mostra l'equivalenza delle due rappresentazioni nel caso di una macchina di Moore

Diagramma degli stati

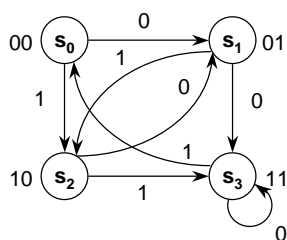


Tabella degli stati

	0	1	U
s₀	s ₁	s ₂	00
s₁	s ₃	s ₂	01
s₂	s ₁	s ₃	10
s₃	s ₃	s ₀	11

Reti Logiche A

Sintesi: metodo completo

- La sintesi si svolge nei seguenti passi:
 - Realizzazione del *diagramma degli stati* a partire dalle specifiche informali del problema
 - Costruzione della *tabella degli stati*
 - Riduzione del numero degli stati: *ottimizzazione*
 - Assegnamento degli stati: *codifica*
 - Costruzione della *tabella delle eccitazioni*
 - Sintesi della rete combinatoria che realizza la funzione stato prossimo
 - Sintesi della rete combinatoria che realizza la funzione d'uscita

Reti Logiche A

Riduzione del numero degli stati

- Il numero minimo di elementi di memoria (flip-flop) necessari a memorizzare tutti gli stati dell'insieme S è:

$$N_{FF,min} = \lceil \log_2 |S| \rceil$$

- Nel modello di una macchina a stati possono esistere stati ridondanti
- L'identificazione ed eliminazione di tali stati comporta:
 - Reti combinatorie meno costose
 - Riduzione del numero di bit necessari per codificare gli stati
 - Aumento dei gradi di libertà nella sintesi combinatoria (don't cares)
 - Numero minore di elementi di memoria
 - Riduzione del numero degli elementi di memoria

Reti Logiche A

Riduzione del numero degli stati

- Lo scopo della riduzione del numero degli stati consiste nella individuazione della *macchina minima equivalente*
- La *macchina minima equivalente* è quella macchina:
 - Funzionalmente equivalente alla macchina data
 - Avente il minimo numero di stati
- Il problema della minimizzazione è distinto in due classi:
 - Macchine completamente specificate
 - Macchine non completamente specificate

Reti Logiche A

Riduzione del numero degli stati

- Si consideri una **macchina completamente specificata**
- Siano:
 - I_α - generica sequenza di ingresso $i_p \dots i_k$
 - U_α - sequenza d'uscita ad essa associata ottenuta attraverso λ .
 - s_p, s_j - due generici stati
- I due stati s_i e s_j appartenenti ad S sono *indistinguibili* se:

$$U_{\alpha,i} = \lambda(s_i, I_\alpha) = \lambda(s_j, I_\alpha) = U_{\alpha,j} \quad \forall I_\alpha$$

- L'indistinguibilità tra s_i e s_j si indica con:

$$s_i \sim s_j$$

Reti Logiche A

Riduzione del numero degli stati

- La relazione di *indistinguibilità* gode di tre proprietà:
 - *Riflessiva*: $s_i \sim s_i$
 - *Simmetrica*: $s_i \sim s_j \leftrightarrow s_j \sim s_i$
 - *Transitiva*: $s_i \sim s_j \wedge s_j \sim s_k \rightarrow s_i \sim s_k$
- La relazione di indistinguibilità è una quindi una *relazione d'equivalenza*
- Una *relazione di equivalenza* induce sull'insieme degli stati una *partizione* Π_e
- L'insieme S si dice partizionato nelle m classi C_1, \dots, C_m se:

$$C_1 \cup C_2 \cup \dots \cup C_m = S$$

$$C_i \cap C_j = \emptyset \quad \forall i, j : i \neq j$$

- Il nuovo insieme degli stati è formato dalle classi della partizione

Reti Logiche A

Riduzione del numero degli stati

- La definizione di indistinguibilità è di difficile applicabilità poiché richiederebbe di considerare *tutte* le sequenze di ingresso
- Si ricorre ad una regola introdotta da *Paull – Unger*
- Due stati s_i e s_j appartenenti ad S sono equivalenti se e solo se per ogni simbolo di ingresso i_α :

$$\lambda(s_i, i_\alpha) = \lambda(s_j, i_\alpha)$$

- le uscite sono uguali per ogni simbolo di ingresso

$$\delta(s_i, i_\alpha) \sim \delta(s_j, i_\alpha)$$

- Gli ststi prossimi sono indistinguibili
- La regola di *Paull – Unger* è ricorsiva

Reti Logiche A

Riduzione del numero degli stati

- Poiché gli insiemi S ed I hanno cardinalità finita, dopo un certo numero di passi ci si troverà in una delle due condizioni:
- $s_i \not\sim s_j$
 - Se i simboli d'uscita sono diversi
 - Se gli stati prossimi sono già stati verificati come distinguibili
- $s_i \sim s_j$
 - Se i simboli di uscita sono uguali
 - Se gli stati prossimi sono già stati verificati come indistinguibili
 - Gli stati prossimi si considerano indistinguibili se sono già stati parte della sequenza di controllo

Reti Logiche A

Riduzione del numero degli stati

- Le relazioni di indistinguibilità o *equivalenze* possono essere identificate attraverso l'uso della *Tabella delle Implicazioni*
- Tale tabella ha le seguenti caratteristiche:
 - Mette in relazione ogni coppia di stati
 - E' triangolare per la proprietà simmetrica
 - E' priva della diagonale principale per la proprietà riflessiva
- Ogni elemento della tabella contiene:
 - Il simbolo di equivalenza, se gli stati corrispondenti sono equivalenti
 - Le coppie di stati a cui si rimanda la verifica, se non è possibile pronunciarsi sulla equivalenza degli stati corrispondenti
- Sulla tabella così ottenuta si procede ad una analisi di tutte le coppie di stati

Reti Logiche A

Riduzione del numero degli stati

- In particolare, per ogni coppia di stati:
- Se si trova il simbolo di equivalenza gli stati sono equivalenti e non è necessaria alcuna ulteriore verifica
- Se si trova un rimando ad un'altra coppia:
 - Se gli stati della coppia cui si rimanda sono equivalenti anche gli stati della coppia in esame sono equivalenti
 - Se gli stati della coppia cui si rimanda non sono equivalenti anche gli stati della coppia in esame non sono equivalenti
 - Se gli stati della coppia cui si rimanda dipendono da una ulteriore coppia di stati si ripete il procedimento in modo ricorsivo fino a quando ci si riconduce ad uno dei due casi precedenti
- L'algoritmo termina quando non sono più possibili eliminazioni
- Le coppie rimaste sono equivalenti

Reti Logiche A

Esempio - Riduzione del numero degli stati

Tabella degli stati

	0	1
a	h/0	g/1
b	c/0	e/0
c	b/0	a/0
d	e/1	c/0
e	h/0	d/1
f	e/1	h/0
g	a/1	c/0
h	d/0	f/1

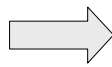


Tabella delle implicazioni

b	x						
c	x	ae					
d	x	x	x				
e	dg	x	x	x			
f	x	x	x	ch	x		
g	x	x	x	ae	x	ae ch	
h	dh fg	x	x	x	dh dg	x	x
	a	b	c	d	e	f	g

Reti Logiche A

Esempio - Riduzione del numero degli stati

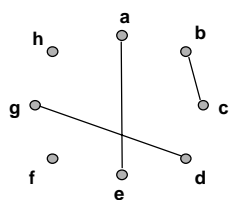
b	x						
c	x	ae					
d	x	x	x				
e	dg	x	x	x			
f	x	x	x	ch	x		
g	x	x	x	ae	x	ae ch	
h	dh fg	x	x	dh dg	x	x	x
	a	b	c	d	e	f	g

1. $a \sim e$ se $d \sim g$, $d \sim g$ se $a \sim e$
si ha circolarità di un vincolo per cui gli stati (a, e) e (d, g) sono equivalenti
- $c \sim b$ se $a \sim e$
a è già stato dimostrato equivalente ad e, quindi (c, b) sono equivalenti
3. $a \sim h$ e $e \sim h$ se $d \sim h$
d non è equivalente a h per cui gli stati (a, h) e (e, h) non sono equivalenti
4. $f \sim d$ e $f \sim g$ se $c \sim h$
c non è equivalente a h per cui gli stati (f, d) e (f, g) non sono equivalenti

Reti Logiche A

Esempio - Riduzione del numero degli stati

- Le relazioni di equivalenza individuate inducono una partizione sull'insieme degli stati
- Le relazioni d'equivalenza sono rappresentabili su un grafo:
 - Vertice: rappresenta uno stato
 - Lato: due vertici sono uniti da un lato se e solo se sono equivalenti
- Le classi d'equivalenza sono i poligoni completi del grafo:



$$\Pi_e = \{ \{a, e\}, \{b, c\}, \{d, g\}, h, f \} = \{ \alpha, \beta, \delta, h, f \}$$

	0	1
α	h,0	δ ,1
β	β ,0	α ,0
δ	α ,1	h,0
f	α ,1	β ,0
h	δ ,0	f,1

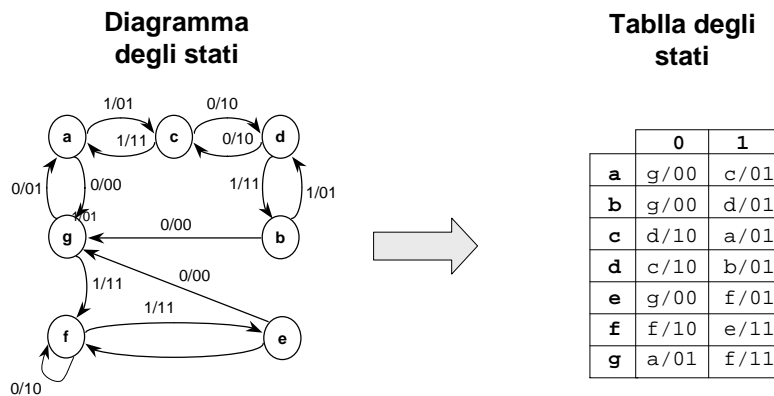
Reti Logiche A

Riduzione del numero degli stati

- La macchina a stati finiti ottenuta dopo la minimizzazione del numero degli stati ha alcune proprietà
- Equivalente alla precedente
 - Per costruzione
- Minima
 - Due stati indistinguibili appartengono ad una sola classe della partizione
 - Due stati distinguibili appartengono a classi distinte della partizione
- Unica
 - La partizione d'equivalenza è unica

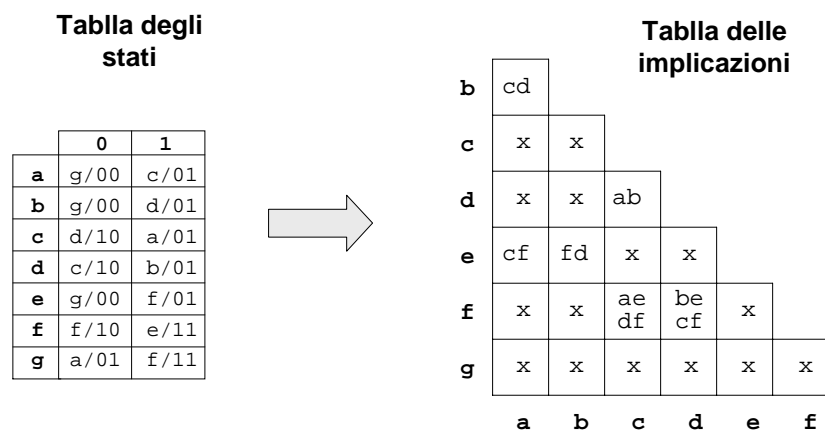
Reti Logiche A

Esempio - Riduzione del numero degli stati



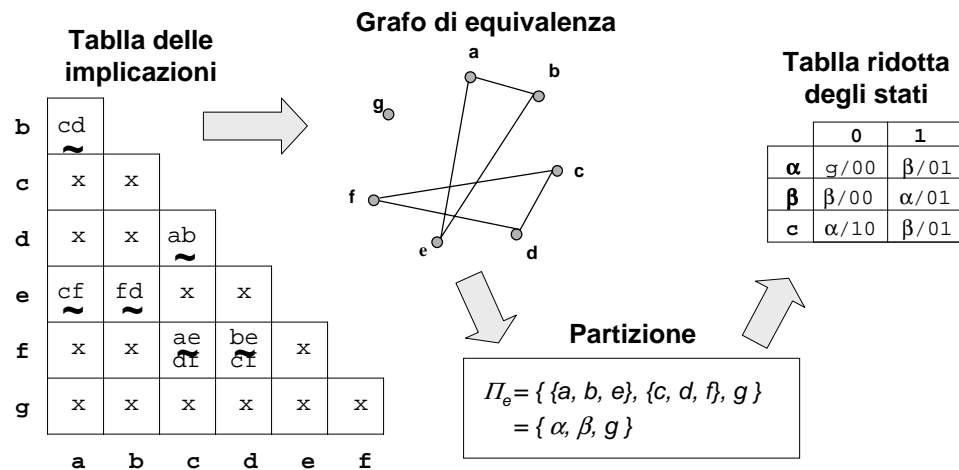
Reti Logiche A

Esempio - Riduzione del numero degli stati



Reti Logiche A

Esempio - Riduzione del numero degli stati



Reti Logiche A

Esempio - Sintesi

- Sintetizzare una macchina di Moore secondo le specifiche:
 - La FSM ha due ingressi A e B
 - La FSM ha una uscita Z, che assume valore iniziale 1
 - L'uscita assume il valore di B quando A=1
 - Tale valore viene mantenuto fino a che non si ripresenta la condizione specificata
 - Il ruolo assunto da A e B viene scambiato quando si presenta la condizione $A = B = Z = 1$
- Il primo passo consiste nel disegnare il diagramma delle transizioni e nel costruire la corrispondente tabella degli stati

Reti Logiche A

Esempio - Sintesi

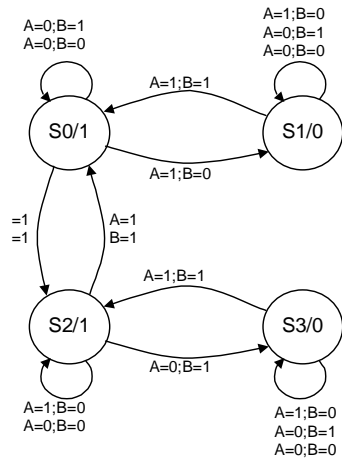


Tabella degli stati

	00	01	11	10	Z
s0	s0	s0	s2	s1	1
s1	s1	s1	s0	s1	0
s2	s2	s3	s0	s2	1
s3	s3	s3	s2	s3	0

Reti Logiche A

Esempio - Sintesi

Tabella degli stati

	00	01	11	10	Z
s0	s0	s0	s2	s1	1
s1	s1	s1	s0	s1	0
s2	s2	s3	s0	s2	1
s3	s3	s3	s2	s3	0

Tabella delle implicazioni

	s0	s1	s2
s1	x		
s2	s0, s3 s1, s2	x	
s3	x	s2, s0	x

Reti Logiche A

Esempio - Riduzione del numero degli stati

Tabella degli
stati

	00	01	11	10
s1	S2/0	S8/1	S6/0	S3/0
s2	S7/0	S1/1	S5/1	S8/1
s3	S4/0	S8/1	S7/0	S5/0
s4	S6/0	S3/1	S1/1	S8/1
s5	S2/0	S8/1	S7/0	S1/0
s6	S1/1	S6/0	S3/1	S7/1
s7	S3/1	S6/0	S5/1	S7/1
s8	S1/1	S2/1	S8/1	S7/1

Tabella delle
implicazioni

	s1	s2	s3	s4	s5	s6	s7
s2	x						
s3	S2/S4 S6/S7 S3,S5	x					
s4	x	S6,S7 S5,S1 S3,S1	x				
s5	S6,S7 S3,S1	x	S4,S2 S5,S1	x			
s6	x	x	x	x	x		
s7	x	x	x	x	x	S3,S1 S3,S5	
s8	x	x	x	x	x	x	x

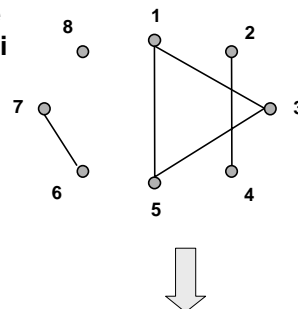
Reti Logiche A

Esempio - Riduzione del numero degli stati

Tabella delle
implicazioni

	s1	s2	s3	s4	s5	s6	s7
s2	x						
s3	S2/S4 S6/S7 S3,S5	x					
s4	x	S6,S7 S5,S1 S3,S1	x				
s5	S6,S7 S3,S1	x	S4,S2 S5,S1	x			
s6	x	x	x	x	x		
s7	x	x	x	x	x	S3,S1 S3,S5	
s8	x	x	x	x	x	x	x

Grafo delle
implicazioni



Partizione

$$\Pi_e = \{ \{s1, s3, s5\}, \{s2, s4\}, \{s6, s7\}, s8 \} = \{a, b, c, s8\}$$

Reti Logiche A

Esempio - Riduzione del numero degli stati

Grafo delle implicazioni

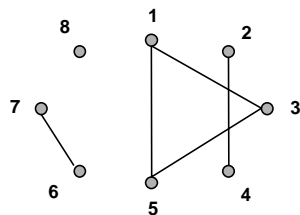


Tabella ridotta degli stati

	00	01	11	10
a	b/0	S8/1	c/0	a/0
b	c/0	a/1	a/1	S8/1
c	a/1	c/0	a/1	c/1
S8	a/1	b/1	S8/1	c/1

Partizione

$\Pi_e = \{ \{S1, S3, S5\}, \{S2, S4\}, \{S6, S7\}, S8 \} =$
 $= \{a, b, c, S8\}$

Reti Logiche A

Codifica degli stati

- Consiste nel determinare la rappresentazione binaria degli stati.
- La codifica degli stati influisce su:
 - Area
 - Prestazioni
- Il numero possibili codifiche è:

$$\frac{(2^{\lceil \log_2 |S| \rceil} - 1)!}{(2^{\lceil \log_2 |S| \rceil} - |S|)! \cdot \lceil \log_2 |S| \rceil!}$$

- Ad esempio, con $|S| = 8$ si hanno 840 possibili codifiche

Reti Logiche A

Codifica degli stati

- Spesso si usano metodi euristici
- Uno dei metodi utilizzabili manualmente, su piccole macchine, si basa sulle seguenti considerazioni (ordinate per importanza):
 1. Se due stati s_i e s_j hanno gli stessi stati futuri è opportuno che abbiano codifiche adiacenti; in modo da avere coppie di 1 o di 0 adiacenti sulle colonne delle funzioni stato prossimo
 2. Se due stati s_i e s_j sono stati prossimi dello stesso stato e corrispondono a ingressi adiacenti, è opportuno che abbiano codifiche adiacenti; in modo da avere coppie di 1 o di 0 adiacenti sulle righe delle funzioni di stato prossimo
 3. Se due stati s_i e s_j hanno la stessa uscita è opportuno dare loro assegnamenti adiacenti; in questo modo si semplifica la funzione di uscita

Reti Logiche A

Codifica degli stati

- É difficile soddisfare queste tre regole contemporaneamente
 - Si cercano soluzioni che sono efficienti da un punto di vista probabilistico
 - A volte si usano codifiche *non minime*, come la one-hot, per motivi di testing
- Procedimento
 - Si raggruppano gli stati che dovrebbero essere adiacenti
 - Si costruisce una mappa di Karnaugh con le variabili di stato necessarie per rappresentare i vari stati
 - Si inseriscono nella tabella i nomi degli stati, cercando di rispettare il maggior numero di vincoli (in particolare i primi due)

Reti Logiche A

Codifica degli stati: esempio

- Data la seguente tabella degli stati (es. Moore)

- Regola 1: adiacenze (a,h) e (b,g)
- Regola 2: adiacenze (b,a) e (a,e)
- Regola 3: conferma (a,h) e (b,g)

- Riassumendo le adiacenze sono

- a: e,h b: c,g

↓

y2 y3 \ y1	00	01	11	10
0	A	E	C	B
1	H	D	F	G



Codifica finale

A: 000 B: 010

C: 011 D: 101

E: 001 F: 111

G: 110 H: 100

In \ S'	0	1	Out
A	B	C	10
B	A	E	01
C	D	D	00
D	G	G	00
E	F	F	00
F	H	H	00
G	A	A	01
H	B	B	10

Reti Logiche A

Tabella delle eccitazioni

- Tutte le informazioni necessarie alla sintesi di una FSM sono fornite da:
 - Tabella, eventualmente ridotta, degli stati
 - Codifica degli stati
- Il problema è quello di sintetizzare le due funzioni:
 - Stato prossimo
 - Uscita
- La funzione di stato prossimo dipende dai bistabili utilizzati:
 - La tabella degli stati e la loro codifica indica quali sono le transizioni che i bistabili devono realizzare
 - Per forzare una data transizione su un bstable si devono applicare ingressi specifici

Reti Logiche A

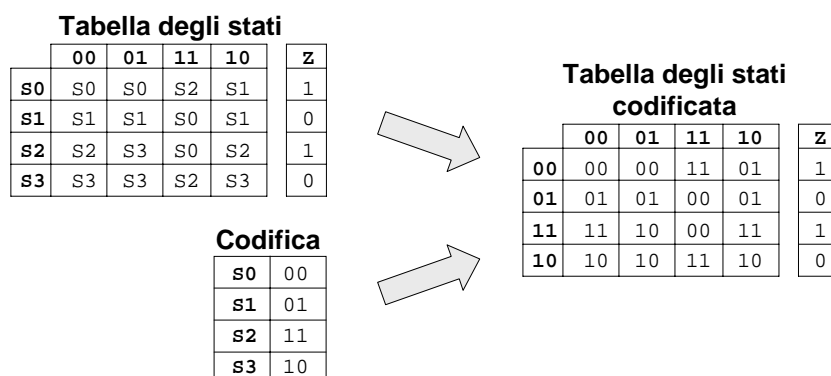
Tabella delle eccitazioni

- Se si utilizzano bistabili di tipo D:
 - L'ingresso del bistabile è uguale al corrispondente bit della codifica dello stato prossimo
- Se si utilizzano bistabili di tipo SR o JK:
 - Si devono determinare gli ingressi S e R o J e K in modo da ottenere la transizione voluta
- Gli ingressi da imporre ad un bistabile in modo che si produca una specifica transizione di stato sono detti *eccitazioni*
- Per determinare le eccitazioni necessarie si ricorre alla *tabella di eccitazione* del tipo di bistabile prescelto

Reti Logiche A

Esempio - Tabella delle eccitazioni

- Si sintetizzi la funzione di stato prossimo della seguente FSM nell'ipotesi di utilizzare bistabili di tipo SR



Reti Logiche A

Esempio - Tabella delle eccitazioni

Tabella delle eccitazioni
di un bistabile di tipo SR

Q	Q'	SR
0	0	0-
0	1	10
1	0	01
1	1	1-

Tabella degli stati
codificata

	00	01	11	10
00	00	00	11	01
01	01	01	00	01
11	11	10	00	11
10	10	10	11	10

Funzioni di
eccitazione S/R

	00	01	11	10
00	0- 0-	0- 0-	10 10	0- 10
01	0- 1-	0- 1-	0- 01	0- 01
11	1- 1-	1- 01	01 01	1- 1-
10	1- 0-	1- 0-	1- 10	1- 0-

Reti Logiche A

Esempio - Tabella delle eccitazioni

- Le quattro mappe di Karnaugh che si ottengono sono quindi:

ab	00	01	11	10
cd				
00	0	0	1	0
01	0	0	0	0
11	1	1	0	1
10	1	1	1	1

$$S_0 = c/d + c/a + ca/b + dab$$

ab	00	01	11	10
cd				
00	-	-	0	-
01	-	-	-	-
11	-	-	1	-
10	-	-	-	-

$$R_0 = c$$

ab	00	01	11	10
cd				
00	0	0	1	1
01	1	1	0	0
11	1	0	0	1
10	0	0	1	0

$$S_1 = c/d + a + d/a/b + dab + c/da + cd/b$$

ab	00	01	11	10
cd				
00	-	-	0	0
01	-	-	1	1
11	-	1	1	-
10	-	-	0	-

$$R_1 = d$$

Reti Logiche A