LABORATORIO DI ARCHITETTURA DEI CALCOLATORI

lezione n° 9

Prof. Rosario Cerbone

rosario.cerbone@uniparthenope.it

http://digilander.libero.it/rosario.cerbone

a.a. 2007-2008

Sintesi di Reti Sequenziali Sincrone

- In questa lezione vengono riassunti i concetti fondamentali per la modellazione e la minimizzazione dei circuiti sequenziali mediante SIS.

Macchina Sequenziale

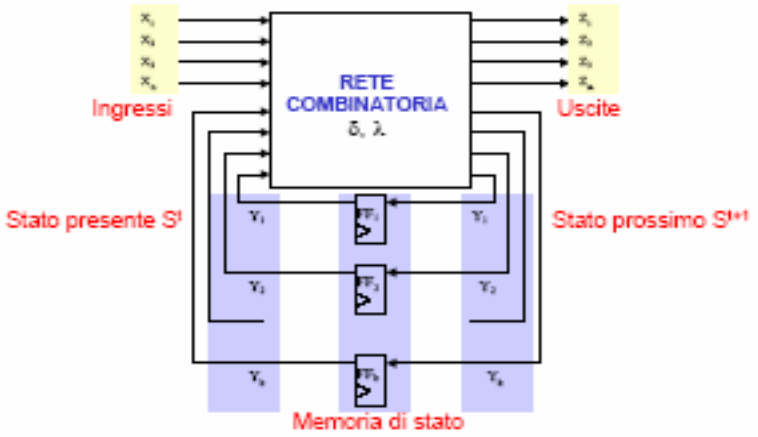
- Una ***macchina sequenziale*** è definita dalla quintupla (I,U,S,δ,λ) dove:
  - **I** è l'insieme finito dei simboli d'ingresso
  - **U** è l'insieme finito dei simboli d'uscita
  - **S** è l'insieme finito e non vuoto degli stati
  - δ è la funzione stato prossimo
  - λ è la funzione d'uscita

La funzione stato prossimo δ:S×I→S associa, ad ogni stato presente e per ogni simbolo di ingresso, uno stato futuro.

La funzione d'uscita λ genera un simbolo d'uscita

- Macchina di Mealy: L'uscita dipende sia dallo stato sia dall'ingresso (λ:S×I→U)
- Macchina di Moore: L'uscita dipende solamente dallo stato (λ:S→U)

Modello di Huffman



Sintesi di una Rete Sequenziale

- Il problema della sintesi di una rete sequenziale consiste nella
  - Identificazione delle funzioni δ e λ
  - Sintesi della rete combinatoria che le realizza.
- Gli elementi di memoria sono costituiti da Flip-Flop.
  - I flip-flop di tipo D sono quelli usati più comunemente
- La funzione stato prossimo dipende dal tipo di bistabile utilizzato
- La funzione d'uscita non dipende dal tipo di bistabile utilizzato

Tabella degli Stati

- Una Macchina a Stati Finiti (FSM) può essere descritta mediante la Tabella degli stati in cui
  - Gli indici di colonna sono i simboli di ingresso $i_\alpha \in I$
  - Gli indici di riga sono i simboli dello stato presente $s_j \in S$
- Gli elementi della tabella sono
  - La coppia $\{u_\beta, s_k\}$ con $u_\beta = \lambda(i_\alpha, s_j)$ e $s_k = \delta(i_\alpha, s_j)$ (Macchine di Mealy)
  - Il simbolo stato prossimo $s_k = \delta(i_\alpha, s_j)$ (Macchine di Moore)
- Nelle macchine di Moore i simboli d'uscita sono associati allo stato presente

Tabella degli stati

- Macchina di Mealy

Stato attuale	Ingresso i_1	Ingresso i_2	Ingresso ...
s_1^t	s_j^{t+1}/u_j	s_k^{t+1}/u_k	...
s_2^t	s_m^{t+1}/u_m	s_n^{t+1}/u_n	...
...	...	...	...

- Macchina di Moore

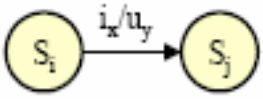
Stato attuale	Ingresso i_1	Ingresso i_2	Ingresso ---	Uscita
s_1^t	s_j^{t+1}	s_k^{t+1}	---	u_1
s_2^t	s_m^{t+1}	s_n^{t+1}	---	u_2
...	---	---	---	---

Diagramma degli Stati

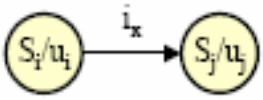
- Spesso la stesura della tabella degli stati è preceduta dalla costruzione di una rappresentazione grafica equivalente denominata *Diagramma degli stati*
- Il diagramma degli stati è un grafo orientato *G*(V,E,L)
- **V**: Insieme dei nodi
  - Ogni nodo rappresenta uno stato
  - Ad ogni nodo è associato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

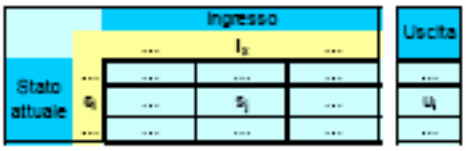
Da Diagramma a Tabella degli Stati

Macchine di Mealy

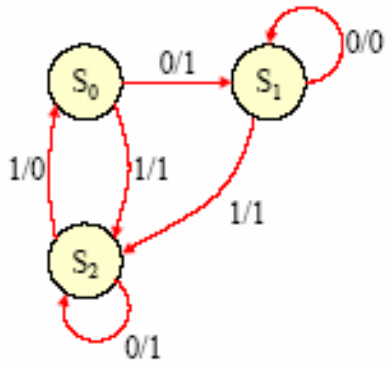


Macchine di Moore



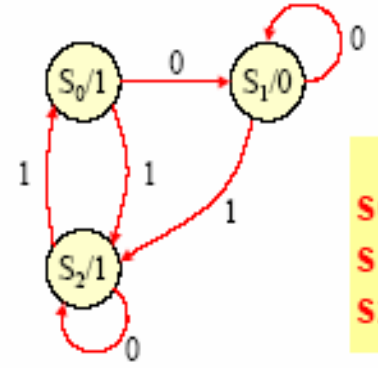


Macchina di Mealy
Esempio



	0	1
S0	$S_1/1$	$S_2/1$
S1	$S_1/0$	S2/1
S2	$S_2/1$	$S_0/0$

Macchina di Moore
Esempio

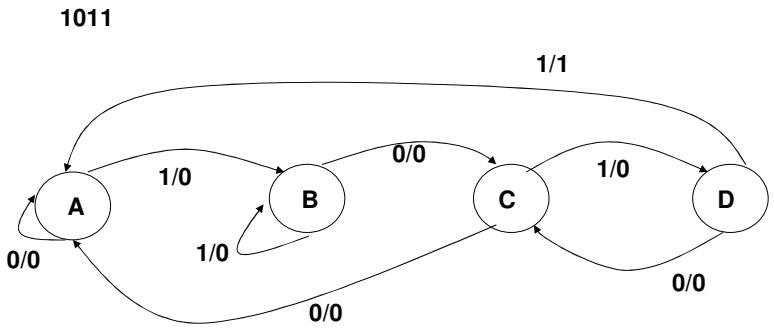


	0	1	
S0	S_1	S_2	1
S1	S_1	S2	0
S2	S_2	S_0	1

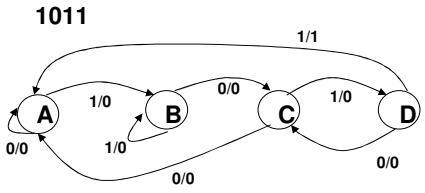
Esercizio 9

- Si ricavi il diagramma e la tabella degli stati di un sistema sequenziale la cui uscita vada ad 1 una volta riconosciuta la sequenza 1011.
- Le sequenze riconosciute non devono essere sovrapposte, ovvero: una volta riconosciuta una sequenza 1011, e prodotto un 1, il sistema ritorna nello stato di partenza. (In altre parole: l'ultimo 1 di una sequenza non può sovrapporsi col primo 1 di una sequenza successiva).

Esercizio 9 - Svolgimento



Esercizio 9 - La tabella degli stati

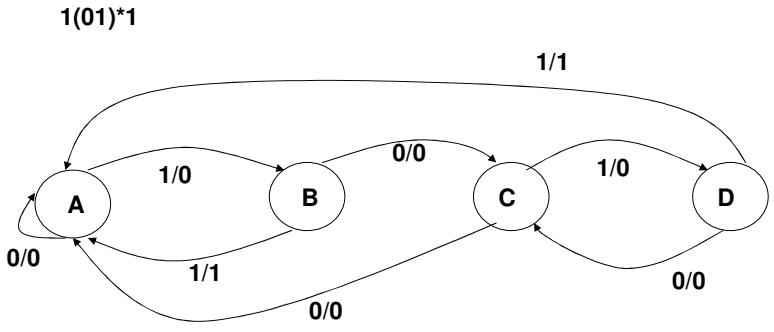


	0	*1*
A	A/0	B/0
B	C/0	**B/0**
C	A/0	D/0
D	C/0	A/1

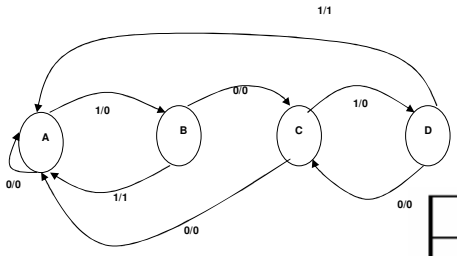
Esercizio 9.a

- Si ricavi il diagramma e la tabella degli stati di un sistema sequenziale la cui uscita vada ad 1 una volta riconosciuta la sequenza 1(01)*1.
- (01)* significa da 0 a infinite presenze della coppia 01.
- Le sequenze riconosciute non devono essere sovrapposte, ovvero: una volta riconosciuta una sequenza 1(01)*1, e prodotto un 1, il sistema ritorna nello stato di partenza. (In altre parole: l ultimo 1 di una sequenza non può sovrapporsi col primo 1 di una sequenza successiva).

Esercizio 9.a - Svolgimento



Esercizio 9.a - La tabella degli stati

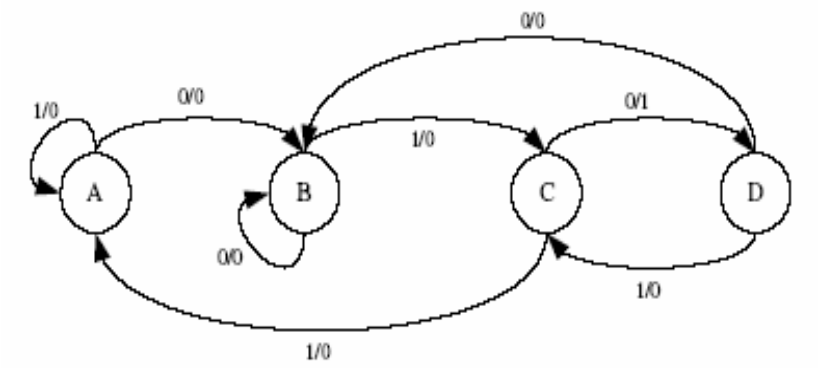


	0	*1*
A	A/0	B/0
B	C/0	A/1
C	A/0	D/0
D	C/0	A/1

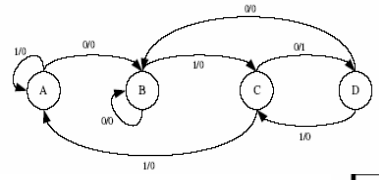
Esercizio 9.b

- Sia data la seguente specifica funzionale di una macchina sequenziale sincrona con un ingresso x ed un uscita z:
- L'uscita al tempo t vale 1 se in ingresso si è presentata la sequenza 010. Le sequenze possono essere anche sovrapposte (es., se la sequenza di ingresso è 00101010, la sequenza d'uscita sarà 00010101).
- Tracciare il diagramma degli stati e ricavare la tabella degli stati

Esercizio 9.b - Il diagramma degli stati



Esercizio 9.b - La tabella degli stati



	x=0	*x=1*
A	B/0	A/0
B	B/0	C/0
C	D/1	A/0
D	B/0	C/0

Sintesi di Reti Sequenziali Sincrone

- **Il procedimento generale di sintesi si svolge nei seguenti passi:**

1. Realizzazione del diagramma degli stati a partire dalle specifiche del problema
2. Costruzione della tabella degli stati
3. ***Minimizzazione del numero degli stati***
4. Codifica degli stati interni
5. Costruzione della tabella delle transizioni
6. Scelta degli elementi di memoria
7. Costruzione della tabella delle eccitazioni
8. Sintesi sia della rete combinatoria che realizza la funzione stato prossimo sia di quella che realizza la funzione d'uscita

Codifica degli Stati Interni

- Il processo di codifica degli stati ha l'obiettivo di identificare per ogni rappresentazione simbolica dello stato una corrispondente rappresentazione binaria
- In seguito alla codifica la *Tabella degli stati* viene trasformata in *Tabella delle Transizioni*
- In questa fase è necessario affrontare i seguenti problemi
- Scelta del codice
  - A minimo numero di bit
  - One-Hot
  - Distanza Minima
- Identificazione della codifica di ogni stato

Codifica degli Stati Interni

- Una volta scelto il codice, la codifica degli stati influisce sia sull'area sia sulle prestazioni del dispositivo

Codifica degli Stati Interni

- **Binario Naturale**
- Il numero di bit è minimo
- Al primo stato corrisponde la configurazione di bit associata a 0, al secondo stato corrisponde la configurazione di bit associata a 1, …
- L'ordinamento degli stati è quello determinato in fase di realizzazione della tabella degli stati
- **One-Hot**
- Il numero di bit è pari al numero degli stati
- In ogni codifica un solo bit assume il valore 1, tutti gli altri assumono valore 0

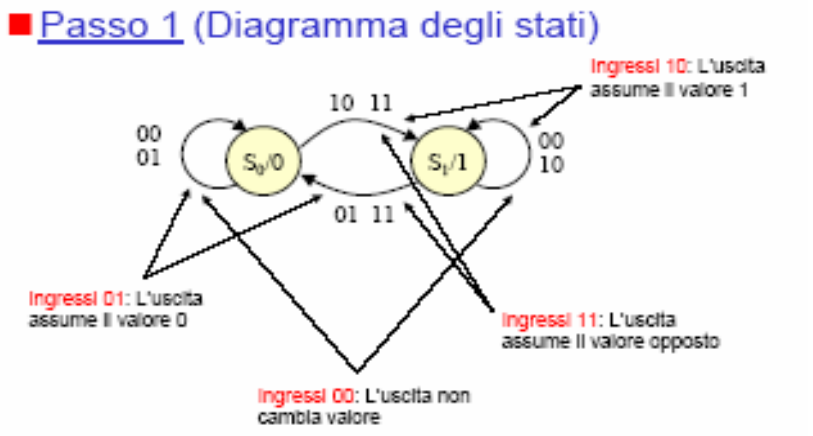
	Binario naturale	Hot-One
S0	00	0001
S1	01	0010
S2	10	0100
S3	11	1000

Esempio 9.c

- **Specifica**
- Realizzare la sintesi di un sistema con due ingressi ed una uscita che abbia il seguente comportamento:
- Ingressi 00: l'uscita non cambia valore
- Ingressi 01: l'uscita assume il valore 0
- Ingressi 10: l'uscita assume il valore 1
- Ingressi 11: l'uscita assume il valore opposto

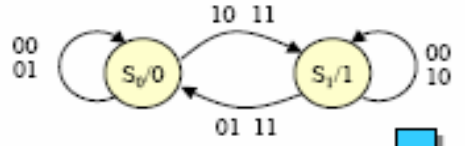
Esempio **9.c**

- Passo 1 (Diagramma degli stati)



Esempio 9.c

- Passo 2 (Tabella degli stati)

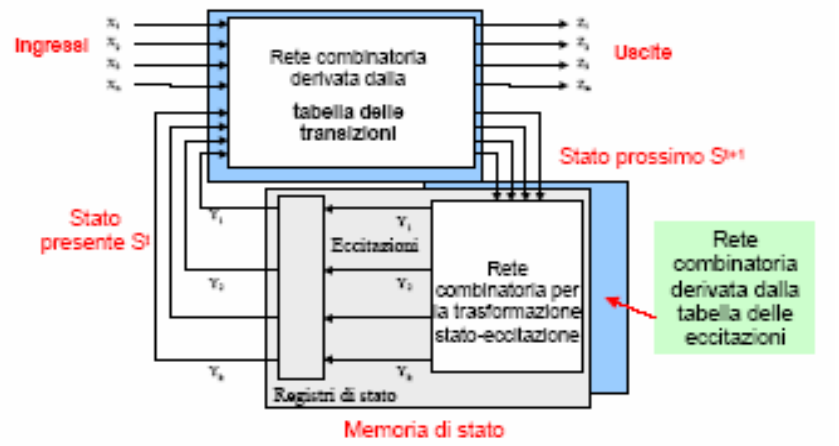


Stato	Ingresso 00	01	11	10	Uscita
S_0	S_0	S_0	S_1	S_1	0
S_1	S_1	S_0	S_0	S_1	1

Scelta degli Elementi di Memoria

- La tabella delle transizioni descrive la relazione tra i bit dello stato presente e quelli dello stato futuro
  - La configurazione in bit dello stato presente è in diretta corrispondenza con l'uscita degli elementi di memoria
  - La configurazione in bit dello stato futuro indica ciò che si vuole ottenere
- Cambiando il tipo dei bistabili variano i segnali che bisogna generare per realizzare la transizione stato presente-stato futuro
- I segnali di ingresso di un bistabile prendono il nome di eccitazioni
- La tabella delle eccitazione di un bistabile rappresenta lo strumento per passare dalla tabella delle transizioni alla tabella delle eccitazioni di una specifica macchina a stati

Scelta degli Elementi di Memoria



Scelta degli Elementi di Memoria

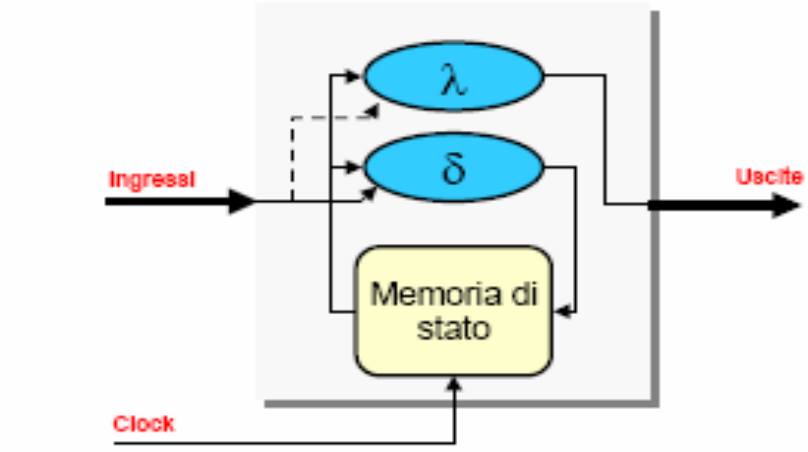
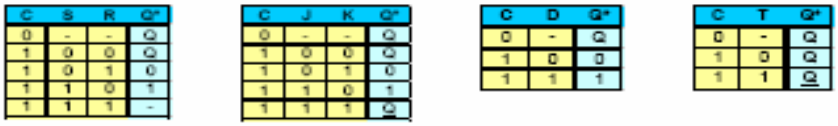
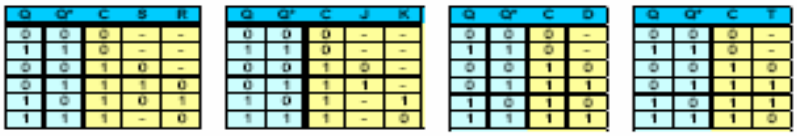


Tabelle delle Transizioni ed Eccitazioni

- Tabelle delle Transizioni



- Tabelle delle Eccitazioni



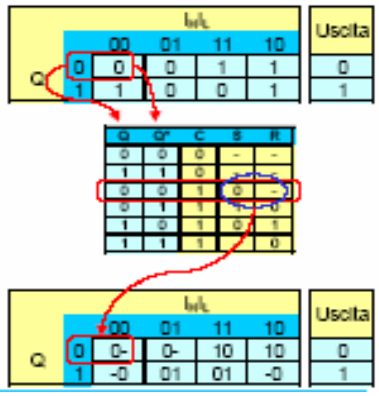
Scelta Bistabile e Costruzione Tabella delle Eccitazioni

- Bistabile SR
- Bistabile JK
- Bistabile D
- Bistabile T

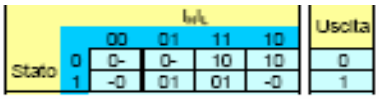
Scelta Bistabile e Costruzione Tabella delle Eccitazioni

- (Passo 3) Codifica Naturale
- (Passo 5) Tabella delle Transizioni
- (Passo 6) Scelta del Bistabile
- (Passo 7) Costruzione Tabella delle eccitazioni

S_0=0, S_1=1



Sintesi

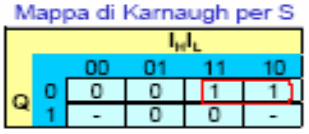


U = Q

Tabella delle eccitazioni

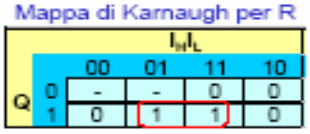
- (Passo 8) Sintesi

Mappa di Karnaugh per S

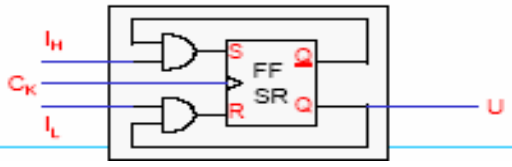


$S = I_H Q$

Mappa di Karnaugh per R



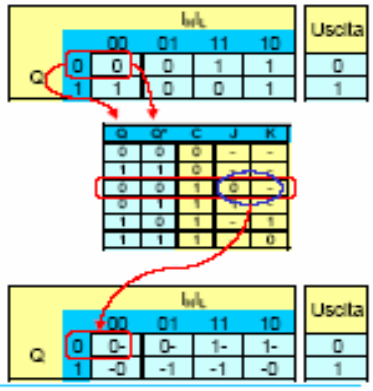
$R = I_L Q$



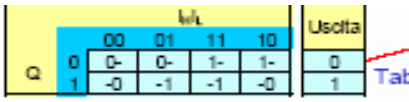
Scelta Bistabile e Costruzione Tabella delle Eccitazioni

- (Passo 3) Codifica Naturale
- (Passo 5) Tabella delle Transizioni
- (Passo 6) Scelta del Bistabile
- (Passo 7) Costruzione Tabella delle eccitazioni

S_0=0, S_1=1



Sintesi

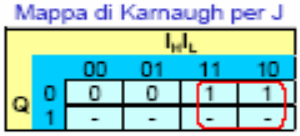


U = Q

Tabella delle eccitazioni

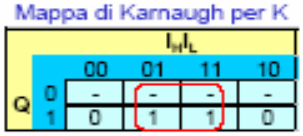
- (Passo 8) Sintesi

Mappa di Karnaugh per J

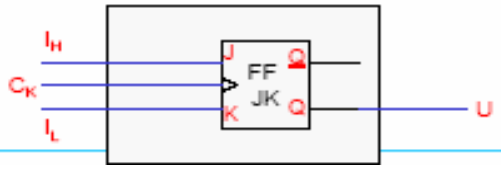


$J = I_H$

Mappa di Karnaugh per K



$R = I_L$



Scelta Bistabile e Costruzione Tabella delle Eccitazioni

- (Passo 3) Codifica Naturale — S_0=0, S_1=1
- (Passo 5) Tabella delle Transizioni

Q	$I_H I_L$ = 00	01	11	10	Uscita
0	0	0	1	1	0
1	1	0	0	1	1

- (Passo 6) Scelta del Bistabile

Q	Q'	C	D
0	0	0	-
1	1	0	-
0	0	1	0
0	1	1	1
1	0	1	0
1	1	1	1

- (Passo 7) Costruzione Tabella delle eccitazioni

Q	$I_H I_L$ = 00	01	11	10	Uscita
0	0	0	1	1	0
1	1	0	0	1	1

Sintesi

Q	$I_H I_L$ = 00	01	11	10	Uscita
0	0	0	1	1	0
1	1	0	0	1	1

U = Q

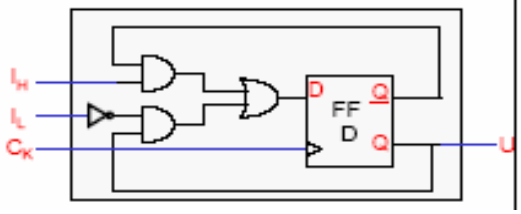
Tabella delle eccitazioni

- (Passo 8) Sintesi

Mappa di Karnaugh per D

Q	$I_H I_L$ = 00	01	11	10
0	0	0	1	1
1	1	0	0	1

$D = I_H Q + I_L Q$



Scelta Bistabile e Costruzione Tabella delle Eccitazioni

- (Passo 3) Codifica Naturale

S_0=0, S_1=1

- (Passo 5) Tabella delle Transizioni

Q	$I_H I_L$ = 00	01	11	10	Uscita
0	0	0	1	1	0
1	1	0	0	1	1

- (Passo 6) Scelta del Bistabile

Q	Q'	S	T
0	0	0	-
1	1	0	-
0	0	1	0
0	1	1	1
1	0	1	1
1	1	1	0

- (Passo 7) Costruzione Tabella delle eccitazioni

Q	$I_H I_L$ = 00	01	11	10	Uscita
0	0	0	1	1	0
1	0	1	1	0	1

Sintesi

Q	$I_H I_L$ = 00	01	11	10	Uscita
0	0	0	1	1	0
1	0	1	1	0	1

U = Q

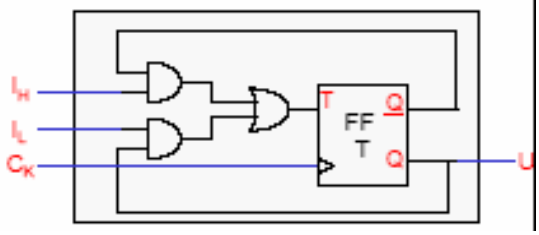
Tabella delle eccitazioni

- (Passo 8) Sintesi

Mappa di Karnaugh per T

Q	$I_H I_L$ = 00	01	11	10
0	0	0	1	1
1	0	1	1	0

$T = I_H\overline{Q} + I_L Q$



Minimizzazione degli Stati

- Il procedimento generale di sintesi si svolge nei seguenti
- passi:
- **1.** Realizzazione del diagramma degli stati a partire dalle specifiche
- del problema
- **2.** Costruzione della tabella degli stati
- **3. Minimizzazione del numero degli stati**
- **4.** Codifica degli stati interni
- **5.** Costruzione della tabella delle transizioni
- **6.** Scelta degli elementi di memoria
- **7.** Costruzione della tabella delle eccitazioni
- **8.** Sintesi sia della rete combinatoria che realizza la funzione stato
- prossimo sia di quella che realizza la funzione d'uscita

Motivazioni

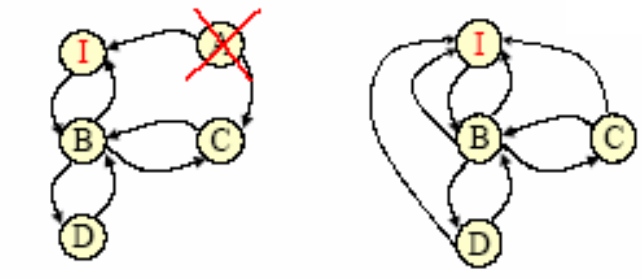
- Il numero minimo di elementi di memoria necessari a memorizzare gli stati dell'insieme S è Nmin=intsup($\log_2$ |S|)
- Nel modello di una macchina a stati possono esistere degli stati ridondanti
- L'identificazione ed eliminazione degli stati ridondanti comporta
  - Reti combinatorie meno costose
  - Minori elementi di memoria

Obiettivi

- Obiettivo della riduzione del numero degli stati è l'individuazione di una macchina minima equivalente, ovvero funzionalmente equivalente e con il minimo numero di stati
- La riduzione viene realizzata in due fasi
- Eliminazione degli stati non raggiungibili partendo dallo stato iniziale
- Identificazione degli stati
  - Equivalenti, per le macchine completamente specificate
  - Compatibili, per le macchine non completamente specificate

Stati Irraggiungibili

- Uno stato è irraggiungibile se non esiste alcuna sequenza di transizione di stato che porti dallo stato iniziale in tale stato



Minimizzazione di Macchine Completamente Specificate

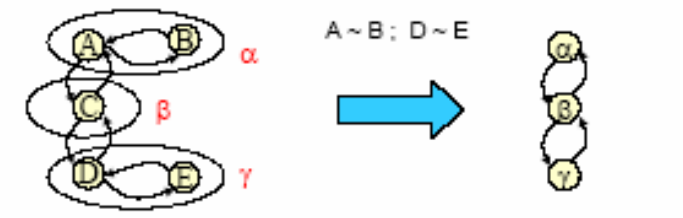
Definizioni

- Siano:
- **I**α una sequenza d' ingresso {ij, …, ik}
- **U**α, sequenza d'uscita ad essa associata ottenuta attraverso λ
- **si, sj** due generici stati
- Due stati si e sj appartenenti ad S sono *indistinguibili* se per qualsiasi sequenza di ingresso le uscite generate partendo da si o da sj sono le stesse
- L'indistinguibilità tra si e sj si indica con si ~ sj
- La relazione di indistinguibilità gode di tre proprietà
  - Riflessiva: si ~ si
  - Simmetrica: si ~ sj ↔ sj ~ si
  - Transitiva: si ~ sj ; sj ~ sk → si ~ sk

Minimizzazione di Macchine Completamente Specificate

Classi di Stati Equivalenti

- Due stati indistinguibili sono equivalenti e possono essere sostituiti da un solo stato
- Un gruppo di stati tra loro equivalenti può essere raggruppato in un'unica classe
- L'insieme di classi individuate determina l'insieme di stati della macchina minima equivalente



Minimizzazione di Macchine Completamente Specificate

Regola di Paull-Unger

- La definizione di indistinguibilità è di difficile applicabilità poiché richiederebbe di considerare tutte le sequenze di ingresso
- Regola di Paull-Unger
- Due stati si e sj sono indistinguibili se e solo se
  - le uscite sono uguali per tutti i simboli d'ingresso
  - gli stati prossimi sono indistinguibili per tutti i simboli d'ingresso
- La regola è iterativa

Minimizzazione di Macchine Completamente Specificate

Regola Paull-Unger - Esempio

	0	1
a	d/0	b/1
b	e/0	b/1
c	a/1	c/1
d	b/1	c/0
e	a/1	c/0

a e **b** hanno la stessa uscita
se gli stati futuri **d** ed **e** sono indistinguibili, **a** ~ **b**

d ed **e** hanno la stessa uscita
se gli stati futuri **a** ed **b** sono indistinguibili, **d** ~ **e**

a non è indistinguibile da **c**,**d** ed **e** poiché ha una differente uscita

Macchina minima equivalente

	0	1
α	γ /0	α /1
β	α /1	β /1
γ	α /1	β /0

Poiché l'indistinguibilità tra **a** e **b** dipende da quella tra **d** ed **e** e viceversa, possiamo concludere che **a** ~ **b, d** ~ **e**

Le classi di indistinguibilità sono:
α={a, b}, β={c}, γ={d, e}

Minimizzazione di Macchine Completamente Specificate

Regola di Paull-Unger

- Poiché gli insiemi I ed S hanno cardinalità finita, dopo un numero finito di passi si verifica una delle due condizioni:
- si≠~sj se i simboli d'uscita sono diversi o gli stati prossimi sono distinguibili
- si~sj se i simboli d'uscita sono uguali e gli stati prossimi sono indistinguibili

Minimizzazione di Macchine Completamente Specificate

Regola Paull-Unger - Tabella delle Implicazioni

- Le relazioni di indistinguibilità possono essere identificate mediante la **Tabella delle Implicazioni**
  - Mette in relazione ogni coppia di stati
  - È triangolare (proprietà simmetrica) e priva di diagonale principale
- Ogni elemento della tabella contiene
  - Il simbolo di non equivalenza (X) o di equivalenza (~)
  - La coppia di stati a cui si rimanda la verifica, se non è possibile pronunciarsi sulla equivalenza degli stati corrispondenti

S1	x		
S2	x	~	
S3	S1,S2	x	x
	S0	S1	S2

Minimizzazione di Macchine Completamente Specificate
Regola Paull-Unger - Tabella delle Implicazioni

- Per ogni coppia di stati
  - Se è marcata come equivalente non è richiesta una ulteriore verifica
  - Se si rimanda ad un'altra coppia
    - Se questi stati sono equivalenti anche gli stati della coppia in esame sono equivalenti
    - Se questi sono non equivalenti anche gli stati della coppia in esame sono non equivalenti
    - Se gli stati della coppia cui si rimanda dipendono da una coppia ulteriore si ripete il procedimento in modo iterativo
- L'analisi termina quando non sono più possibili eliminazioni
- Le coppie rimaste sono equivalenti

Minimizzazione di Macchine Completamente Specificate
Regola di Paull-Unger - Tabella delle Implicazioni

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

	a	b	c	d	e	f
b	cg ae					
c	x	x				
d	bg	bc ae	x			
e	~	cg	x	bg ae		
f	x	x	de fg	x	x	
g	x	x	ae	x	x	ad

Minimizzazione di Macchine Completamente Specificate
Regola Paull-Unger - Tabella delle Implicazioni

	a	b	c	d	e	f
b	cg ae					
c	x	x				
d	bg	bc ae	x			
e	~	cg	x	bg ae		
f	x	x	de fg	x	x	
g	x	x	ae	x	x	ad

Analisi delle coppie degli stati

b-a: c-g è indistinguibile se lo è a-e → b~a
d-a: b-g è distinguibile → d~a
d-b: b-c è distinguibile → d~b
e-b: c-g è indistinguibile se lo è a-e → b~e
e-d: b-g è distinguibile → e~d
f-c: d-e è indistinguibile se lo è b-g → f~c
g-c: a-e è indistinguibile → g~c
g-f: a-d è indistinguibile se lo è b-g → g~f

Minimizzazione di Macchine Completamente Specificate
Regola Paull-Unger - Tabella delle Implicazioni

	a	b	c	d	e	f
b	~					
c	x	x				
d	x	x	x			
e	~	~	x	x		
f	x	x	x	x	x	
g	x	x	~	x	x	x

Classi di indistinguibilità

α = {a,b,e}
β = {c,g}
γ = {d}
δ = {f}

Tabella degli stati minima equivalente

	0	1
α	β /0	α /1
β	α /1	β /0
γ	α /0	α /1
δ	γ /1	δ /0

Minimizzazione di Macchine Completamente Specificate

Regola di Paull-Unger - Osservazioni

- Per le FSM completamente specificate l'algoritmo di Paull-Unger
  - Consente di identificare in maniera esatta la FSM minima equivalente
    - La partizione di equivalenza è unica (ogni stato appartiene ad una ed una sola classe)
  - Ha una complessità esponenziale con il numero di stati

Macchine non completamente specificate

Definizioni

- Sono macchine in cui per alcune configurazioni degli ingressi e stati correnti non sono specificati gli stati futuri e/o le configurazioni d'uscita
- Due stati si e sj si dicono compatibili ($si \approx sj$)
  - Se, assunti come stati iniziali, per ogni possibile sequenza di ingresso (grande a piacere) danno luogo a sequenze di simboli d'uscita identici a meno di condizioni di indifferenza

Macchine non completamente specificate
Regola di Paull-Unger Estesa

- La compatibilità è una relazione meno forte di quella di indistinguibilità, non vale la proprietà transitiva

	0	1
A	C/1	A/-
B	C/-	A/1
C	C/0	A/-

A≈B; B≈C ma A ≉ C

La regola di Paull-Unger è stata estesa per trattare il caso di macchine non completamente specificate

Due stati si e sj sono compatibili se e solo se

- λ(si,i) = λ(sj,i) □ i □ I ovunque sono entrambi specificati
- δ(si,i) ≈ δ(sj,i) □ i □ I ovunque sono entrambi specificati

La suddetta definizione è ricorsiva

Modellazione e minimizzazione di circuiti sequenziali in SIS

- Per modellare un circuito sequenziale in SIS è necessario definire la tabella delle transizioni ed eventualmente la codifica degli stati (esiste un comando che permette di eseguire la codifica in modo automatico).

Modellazione e minimizzazione di circuiti sequenziali in SIS

- La tabella delle transizioni deve essere descritta all'interno delle keyword **.start_kiss** e **.end_kiss** dopo aver definito
- **.model**
- **.inputs**
- **.outputs**.
- Le transizioni devono essere specificate come un insieme di righe che riportano in ordine: valore degli ingressi, stato presente, stato prossimo, valore delle uscite.

Modellazione e minimizzazione di circuiti sequenziali in SIS

- La tabella delle transizioni deve essere preceduta da 5 righe che specificano il numero di input, il numero di output, il numero di transizioni (opzionale), il numero di stati e lo stato di reset.
- Dopo la tabella delle transizioni (dopo **.end_kiss**) possono essere riportate le istruzioni necessarie per definire la codifica degli stati qualora non si decida di farla definire a SIS in modo automatico. La keyword da utilizzare per definire la codifica è **.code** seguita dal nome dello stato e dalla sua codifica binaria.

Modellazione e minimizzazione di circuiti sequenziali in SIS

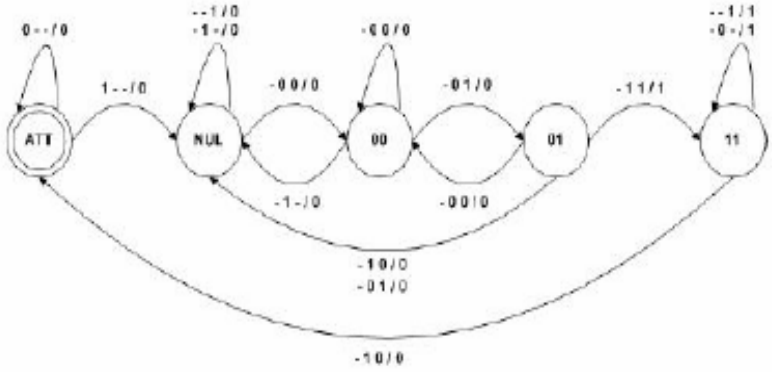
- Dopo aver modellato il circuito, è possibile procedere con la minimizzazione degli stati.
- Una volta caricato il file .blif con il comando read_blif, la minimizzazione degli stati si esegue con il comando **state_minimize stamina**. Quindi è possibile eseguire la minimizzazione della logica combinatoria, ad esempio lanciando lo script script.rugged come visto per minimizzazione dei circuiti combinatori.

Esempio

- Si consideri il circuito sequenziale che è in grado di riconoscere la sequenza di ingresso 00 01 11, (ingresso IN a 2 bit).
- Il circuito è attivo ed inizia ad analizzare i valori dell'ingresso IN quando l'ingresso START passa da 0 a 1.
- Nello stesso ciclo di clock in cui viene riconosciuta la sequenza 00 01 11, l'uscita OUT passa da 0 a 1.
- OUT rimane a 1 fino a quando gli ingressi assumeranno il valore 10; momento in cui il circuito viene nuovamente posto in attesa che il segnale START passi da 0 a 1.

Esempio

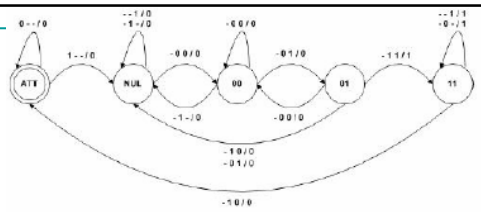
- La seguente figura illustra il grafo di transizione degli stati.



Esempio

- Nello stato ATT, il circuito è in attesa che il segnale di start commuti da 0 a 1. Gli altri 4 stati indicano che una porzione della sequenza è stata riconosciuta. Si noti che se nello stato 01 viene applicato l'input 00, l'automa anziché portarsi nello stato NUL si sposta nello stato 00 riconoscendo l'inizio di una sequenza valida.
- Scegliendo di codificare gli stati come ATT = 000, NUL = 001, 00 = 010, 01 = 011 e 11 = 100, si riporta di seguito la tabella di verità della funzione dello stato prossimo e dell'uscita OUT. Lo stato attuale è codificato mediante i bit a2 a1 a0, mentre lo stato prossimo è codificato con i bit s2 s1 s0.

Esempio



a2	a1	a0	START	IN1	IN0	s2	s1	s0	OUT
0	0	0	0	-	-	0	0	0	0
0	0	0	1	-	-	0	0	1	0
0	0	1	-	0	0	0	1	0	0
0	0	1	-	-	1	0	0	1	0
0	0	1	-	1	-	0	0	1	0
0	1	0	-	0	0	0	1	0	0
0	1	0	-	0	1	0	1	1	0
0	1	0	-	1	-	0	0	1	0
0	1	1	-	1	1	1	0	0	1
0	1	1	-	0	0	0	1	0	0
0	1	1	-	1	0	0	0	1	0
0	1	1	-	0	1	0	0	1	0
1	0	0	-	1	0	0	0	0	0
1	0	0	-	0	-	1	0	0	0
1	0	0	-	-	1	1	0	0	1
1	0	0	-	0	-	1	0	0	1

Esempio

- La rappresentazione nel formato blif è la seguente:
- .model automa
- .inputs START IN1 IN0
- .outputs OUT
- .start_kiss
- .i 3 *#numero di segnali di ingresso*
- .o 1 *#numero di segnali di uscita*
- .s 5 *#numero di stati*
- .p 15 *#numero di transizioni*
- .r ATT *#stato di reset*

Esempio

- *#tabella delle transizioni*
- *#(ingressi, stato presente, stato prossimo, uscita)*
- 0-- ATT ATT 0
- 1-- ATT NUL 0
- --1 NUL NUL 0
- -1- NUL NUL 0
- -00 NUL 00 0
- -00 00 00 0
- -1- 00 NUL 0
- -01 00 01 0
- -00 01 00 0
- -10 01 NUL 0
- -01 01 NUL 0
- -11 01 11 1
- --1 11 11 1
- -0- 11 11 1
- -10 11 ATT 0
- .end_kiss

a2	a1	a0	START	IN1	IN0	s2	s1	s0	OUT
0	0	0	0	-	-	0	0	0	0
0	0	0	1	-	-	0	0	1	0
0	0	1	-	0	0	0	1	0	0
0	0	1	-	-	1	0	0	1	0
0	0	1	-	1	-	0	0	1	0
0	1	0	-	0	0	0	1	0	0
0	1	0	-	0	1	0	1	1	0
0	1	0	-	1	-	0	0	1	0
0	1	1	-	1	1	1	0	0	1
0	1	1	-	0	0	0	1	0	0
0	1	1	-	1	0	0	0	1	0
0	1	1	-	0	1	0	0	1	0
1	0	0	-	1	0	0	0	0	0
1	0	0	-	0	-	1	0	0	0
1	0	0	-	-	1	1	0	0	1
1	0	0	-	0	-	1	0	0	1

Esempio

- *#codifica degli stati.*
- *#E' opzionale perché può essere calcolata automaticamente*
- *#tramite il comando state_assign jedi*
- .code ATT 000
- .code NUL 001
- .code 00 010
- .code 01 011
- .code 11 100
- .end

a2	a1	a0	START	IN1	IN0	s2	s1	s0	OUT
0	0	0	0	-	-	0	0	0	0
0	0	0	1	-	-	0	0	1	0
0	0	1	-	0	0	0	1	0	0
0	0	1	-	-	1	0	0	1	0
0	0	1	-	1	-	0	0	1	0
0	1	0	-	0	0	0	1	0	0
0	1	0	-	0	1	0	1	1	0
0	1	0	-	1	-	0	0	1	0
0	1	1	-	1	1	1	0	0	1
0	1	1	-	0	0	0	1	0	0
0	1	1	-	1	0	0	0	1	0
0	1	1	-	0	1	0	0	1	0
1	0	0	-	1	0	0	0	0	0
1	0	0	-	0	-	1	0	0	0
1	0	0	-	-	1	1	0	0	1
1	0	0	-	0	-	1	0	0	1

Esempio

- Una volta caricato il file blif con il comando read_blif, è possibile creare le funzioni per lo stato prossimo e per l'output con il comando **stg_to_network**.
- A questo punto, visualizzando il blif con il comando write_blif si ottiene:
- .model automa
- .inputs START IN1 IN0
- .outputs OUT
- # Sono stati creati tre elementi di memoria
- #(.latch, segnale di input, segnale di output, reset)
- .latch [2] LatchOut_v3 0
- .latch [3] LatchOut_v4 0
- .latch [4] LatchOut_v5 0
- .start_kiss
- .....
- .end_kiss

Esempio

- .latch_order LatchOut_v3 LatchOut_v4 LatchOut_v5
- .code ATT 000
- .code NUL 001
- .code 00 010
- .code 01 011
- .code 11 100
- #Queste sono le funzione di stato prossimo
- .names IN1 IN0 LatchOut_v3 LatchOut_v4 LatchOut_v5 [2]
- 0-1-- 1
- -11-- 1
- 11-11 1
- .names IN1 IN0 LatchOut_v4 LatchOut_v5 [3]
- 00-1 1
- 0-10 1
- .names START IN1 IN0 LatchOut_v3 LatchOut_v4 LatchOut_v5 [4]
- -01-1- 1
- -10--1 1
- --1-01 1
- -1--10 1
- 1--000 1

Esempio

- # questa è la funzione di l'uscita
- .names IN1 IN0 LatchOut_v3 LatchOut_v4 LatchOut_v5 OUT
- 0-1-- 1
- -11-- 1
- 11-11 1

Esempio

- # Non tutti le codifiche di stato sono utilizzate
- #Le configurazioni don't care sono le seguenti
- .exdc
- .inputs START IN1 IN0 LatchOut_v3 LatchOut_v4 LatchOut_v5
- .outputs [2] [3] [4] OUT
- .names LatchOut_v3 LatchOut_v4 LatchOut_v5 [2]
- 11- 1
- 1-1 1
- .names LatchOut_v3 LatchOut_v4 LatchOut_v5 [3]
- 11- 1
- 1-1 1
- .names LatchOut_v3 LatchOut_v4 LatchOut_v5 [4]
- 11- 1
- 1-1 1
- .names LatchOut_v3 LatchOut_v4 LatchOut_v5 OUT
- 11- 1
- 1-1 1
- .end

Esempio

- E' possibile:
- minimizzare gli stati tramite il comando **state_minimize stamina**,
- assegnare una codifica per gli stati minimizzati usando il comando **state_assign jedi**
- minimizzare la logica combinatoria tramite il comando source –x script.rugged.
- Notare che l'algoritmo jedi per l'assegnamento della codifica agli stati deve essere eseguito dopo la minimizzazione degli stati.

Comandi utili di SIS

read_blif	Carica la descrizione blif del circuito
simulate i0 i1 i2 …	Simula il circuito in base ai valori forniti per gli ingressi. Esecuzioni successive del comando considerano lo stato in cui il circuito si è portato dopo l'ultima esecuzione
print_stats	Visualizza informazioni sul circuito
write_blif	Visualizza la descrizione blif del circuito.
write_eqn	Visualizza le equazioni booleane corrispondenti ai nodi del circuito
stg_to_network	Costruisce automaticamente le funzioni di stato prossimo e di uscita a partire dalla tabella delle transizioni e dalla codifica degli stati
state_assign jedi	Usa l'algoritmo jedi per assegnare automaticamente una codifica degli stati che permetta una minimizzazione del circuito e costruisce le funzioni di stato prossimo e di uscita
state_minimize stamina	Usa l'algoritmo stamina per minimizzare gli stati della FSM
write kiss	Visualizza la tabella delle transizioni

Esercizio 9.1

- Tracciare il diagramma degli stati e ricavare la tabella degli stati di un circuito sequenziale in grado di riconoscere la sequenza

  00 11 00 01.

Eseguire la minimizzazione con il programma SIS.

Esercizio 9.2

- Tracciare il diagramma degli stati di un circuito sequenziale (2 ingressi, 2 uscite) corrispondente alla seguente tabella degli stati. Sia A lo stato di reset.
- Eseguire la minimizzazione con il programma SIS

	00	01	10	11
A	D/10	D/00	A/11	D/11
B	B/1-	A/--	B/-1	C/--
C	C/0-	H/-1	I/-0	A/-0
D	A/10	A/-0	D/11	A/11
E	G/10	G/--	C/-1	I/1-
F	H/--	H/-0	A/01	B/10
G	A/10	G/-0	H/00	D/11
H	D/1-	G/-0	H/00	A/-1
I	F/--	F/11	E/00	A/01

Esercizio 9.3

- Tracciare il diagramma degli stati di un circuito sequenziale (2 ingressi, 2 uscite) corrispondente alla seguente tabella degli stati. Sia A lo stato di reset.
- Eseguire la minimizzazione con il programma SIS

	00	01	10	11
A	D/00	D/00	A/11	D/11
B	B/1-	A/1-	B/-1	C/--
C	C/0-	H/-1	I/-0	A/-0
D	A/11	A/-0	D/11	A/11
E	G/10	D/--	D/-1	I/1-
F	H/--	A/-0	A/01	A/10
G	A/10	G/-0	H/00	H/11
H	D/1-	G/-0	D/00	A/-1
I	F/--	F/11	E/00	A/01