Macchine Sequenziali

Corso di Calcolatori Elettronici A

2007/2008

Sito Web:http://prometeo.ing.unibs.it/quarella

Prof. G. Quarella

prof@quarella.net

Limiti delle reti combinatorie

- Ogni funzione di n variabili binarie, con n finito grande a piacere può essere realizzata da una rete combinatoria.
- In una rete combinatoria il segnale di uscita da una porta non può mai riapparire come segnale di ingresso della stessa porta
- Per realizzare **funzioni complesse** sarebbe quindi necessario usare reti con un numero molto elevato di porte (problemi di **costo** e di **tempo**) [es. nella moltiplicazione/divisione il numero di porte cresce con n^2]
- Se gli n input non arrivano simultaneamente, occorre memorizzarli per fare il calcolo con una rete combinatoria.

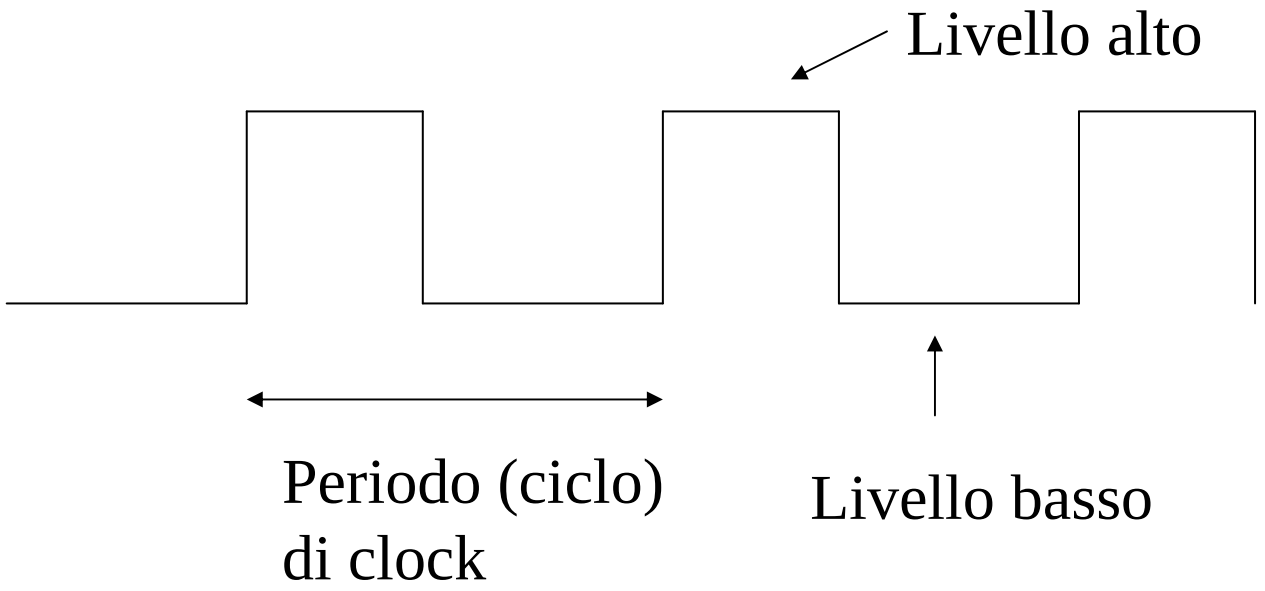
Il ruolo del tempo

- I circuiti dotati di memoria prevedono l'uso di *controreazione:* diventa cruciale controllare i diversi ***istanti*** in cui i segnali si presentano all'ingresso di una porta
- Le reti sequenziali usano un altro circuito elettronico detto ***temporizzatore*** (clock pulse generator)
- I segnali emessi dal temporizzatore servono a ***sincronizzare*** la rete

Segnali di temporizzazione (clock)

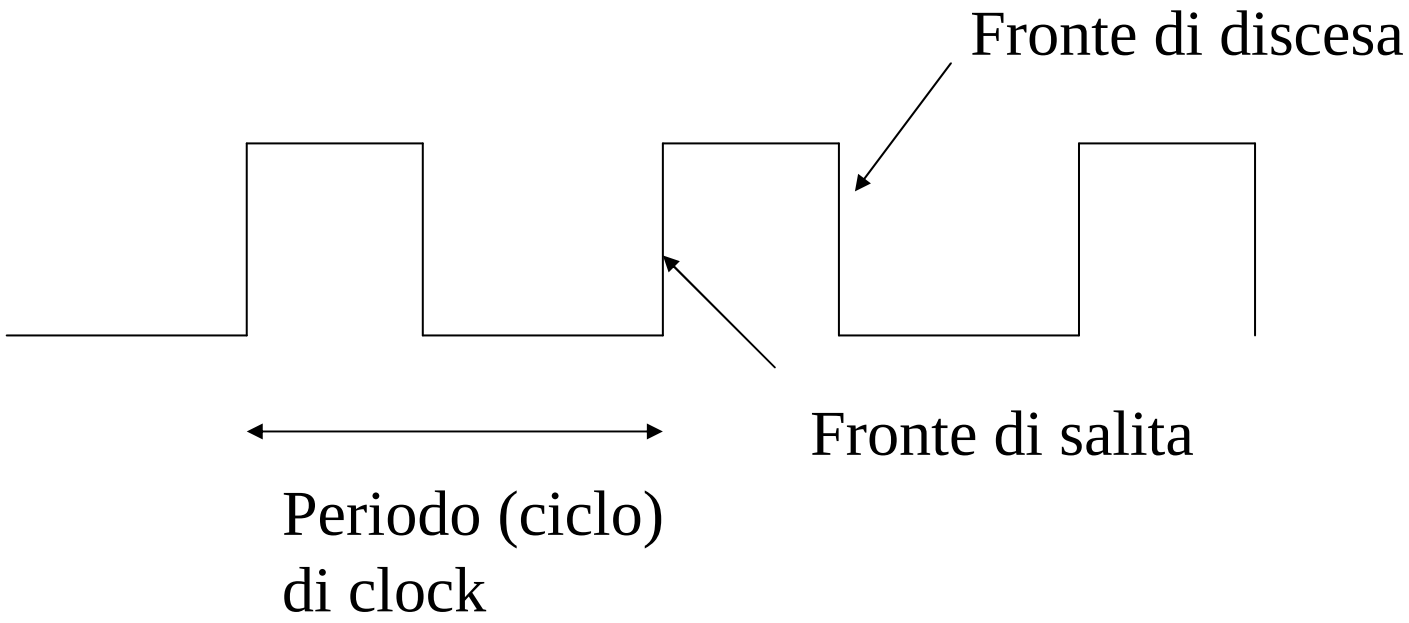
- Un segnale di clock è un segnale che evolve con un *periodo* (tempo di ciclo) predeterminato e costante (intervallo di tempo fra 2 segnali di clock consecutivi)
- La *frequenza di clock* è l'inverso del periodo (numero di cicli di clock al secondo)
- Esempio: frequenza 500 MHz, periodo 2 ns
- Esistono diverse metodologie di temporizzazione

Metodologia di temporizzazione sensibile ai livelli



In questa metodologia le **variazioni** degli elementi di stato avvengono in corrispondenza dei ***livelli alti*** o di quelli ***bassi***

Metodologia di temporizzazione sensibile ai fronti

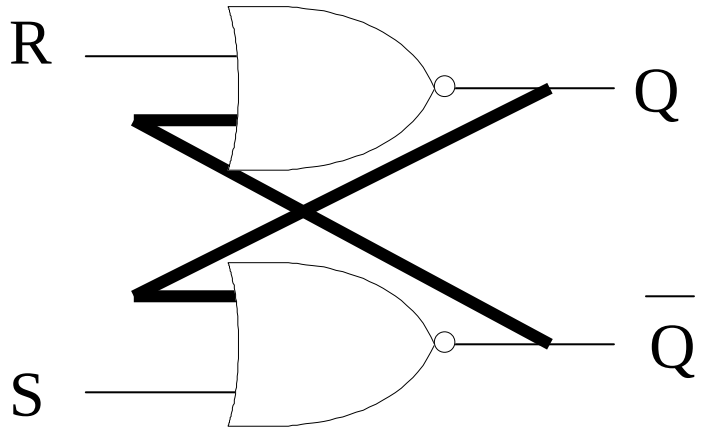


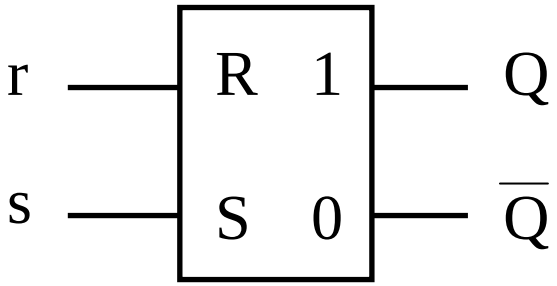
In questa metodologia uno dei due fronti è detto ***attivo*** e causa le ***variazioni di stato***, cioè il contenuto degli elementi di stato può cambiare **solo** in corrispondenza del fronte attivo del clock

Elementi di memoria

- Gli elementi di memoria memorizzano **informazioni di stato**
- L’uscita di un elemento di memoria dipende sia dai suoi ingressi che dal valore memorizzato all’interno dell’elemento stesso (lo stato)
- I circuiti che contengono elementi di memoria sono detti **sequenziali**

Una rete non combinatoria: il Latch SR





Latch SR (set-reset)

- Elemento di memoria per la memorizzazione di 1 bit (deve ricordare due situazioni)

- Ha 2 ingressi, R ed S, e si comporta secondo le regole:
  - R=1, S=0 memorizza 0
  - R=0, S=1 memorizza 1
  - R=S=0 ricorda la situazione corrente
  - R=S=1 non è permesso

Tabella di transizione

R	S	Q(t)	Q(t+1)
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	-
1	1	1	-
		t	$t+\Delta t$

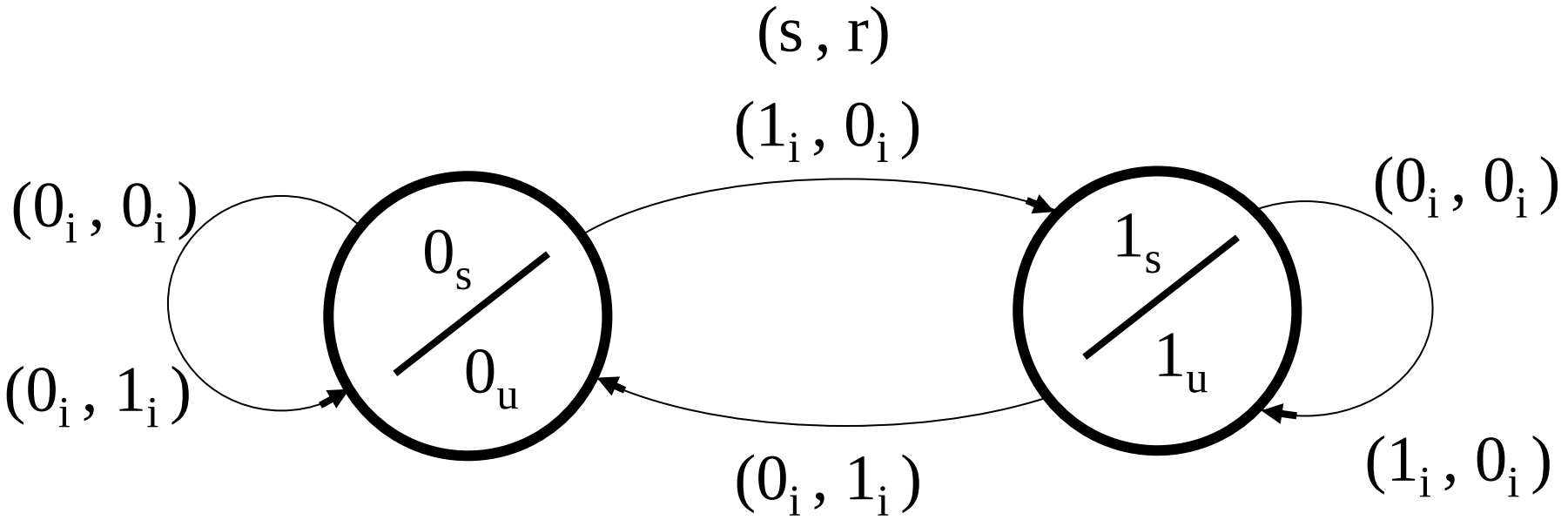
Per descriverlo…

Devo specificare:

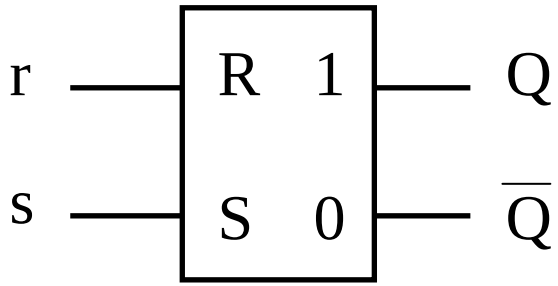
- **I** = {0,1} insieme dei simboli di ingresso
- **U** = {0,1} insieme dei simboli di uscita
- **S** = $\{S_0, S_1\}$ insieme degli stati (situazioni da ricordare)
- δ funzione di transizione che specifica il nuovo stato in base allo stato corrente e all'ingresso (la tabella)
- ω funzione di uscita che specifica il valore di uscita in base allo stato corrente e all'ingresso (in questo caso l'identità)
- Automa $A = \langle I, U, S, \delta, \omega \rangle$

È comodo utilizzare la seguente rappresentazione grafica:

Una descrizione grafica

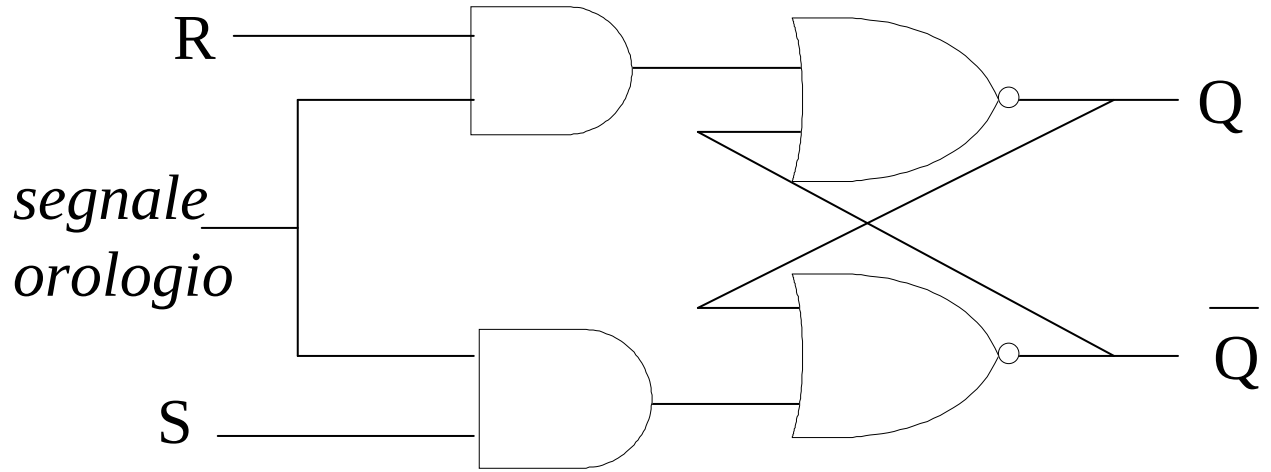


(1,1) proibito



Latch SR sincrono

Ha in ingresso un segnale di clock e la variazione di stato viene attivata dal clock



Latch di tipo D(elay) sincrono

- Memorizza il valore del dato D presente in ingresso
- Permette di risolvere l'ambiguità del latch RS: infatti R=S=1 non si può verificare.

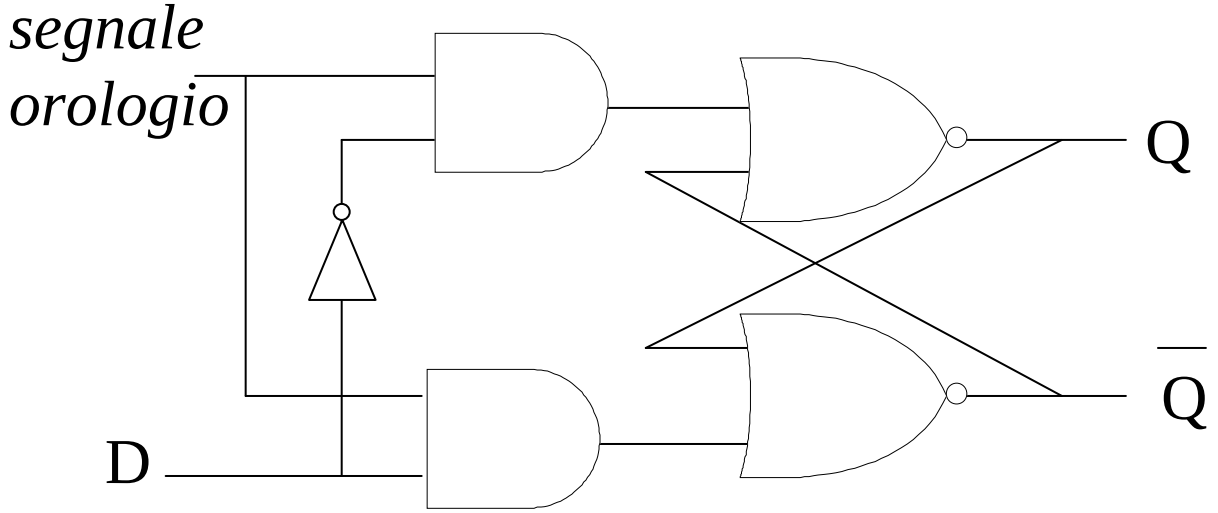
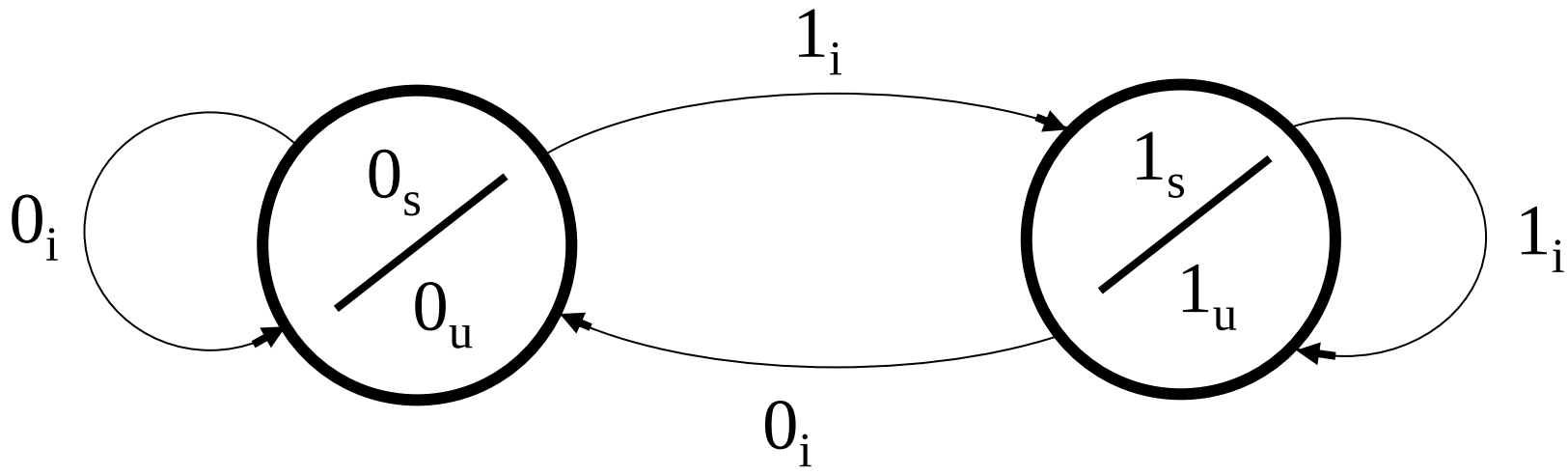
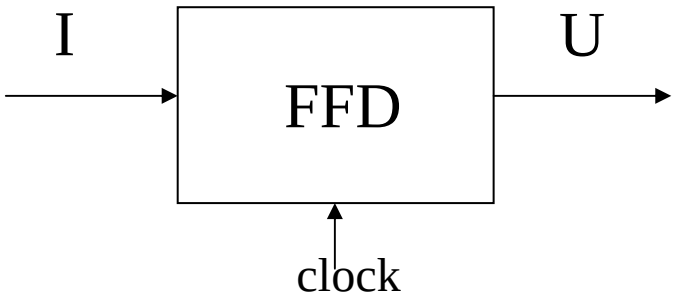


Tabella di transizione

D	Q(t)	Q(t+1)
0	x	0
1	x	1

Una descrizione grafica

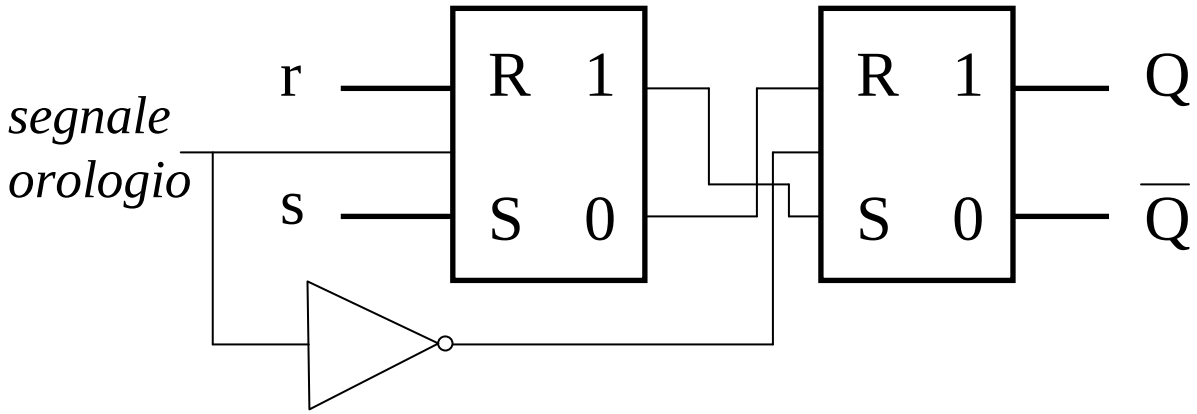




Latch e Flip-flop

- Il latch non permette, *durante lo stesso ciclo di clock*, di leggere il segnale in uscita e di modificare quello in ingresso
- Infatti, se il segnale d'ingresso è applicato *prima* che sia stato letto quello di uscita, il latch potrebbe commutare ed il segnale letto potrebbe non essere quello desiderato
- Il **flip-flop** (circuito flip-flop master slave) permette la scrittura e la lettura simultanea
- Nel seguito useremo sempre flip-flop come elementi di stato

Flip-flop master-slave



- Collegamento di 2 latch in serie: il latch master riceve gli ingressi e il latch slave produce le uscite
- Si sfruttano i 2 livelli del ciclo di clock per effettuare in tempi diversi i cambiamenti di stato dei due latch
- Durante il livello attivo è possibile la lettura, dato che è nel livello passivo che il latch slave cambia stato.

Macchine sequenziali

- Consideriamo sistemi il cui numero di stati interni distinti è **finito** e corrisponde a tutti i possibili valori degli elementi di memoria interna
- Con n bit di memoria si hanno 2^n stati
- La **funzione di stato futuro** è una funzione combinatoria che calcola il valore del prossimo stato assunto dal sistema in base agli ingressi ed allo stato presente
- Allo stesso modo, la **funzione di uscita** calcola le uscite
- Le macchine considerate sono di tipo **sincrono**

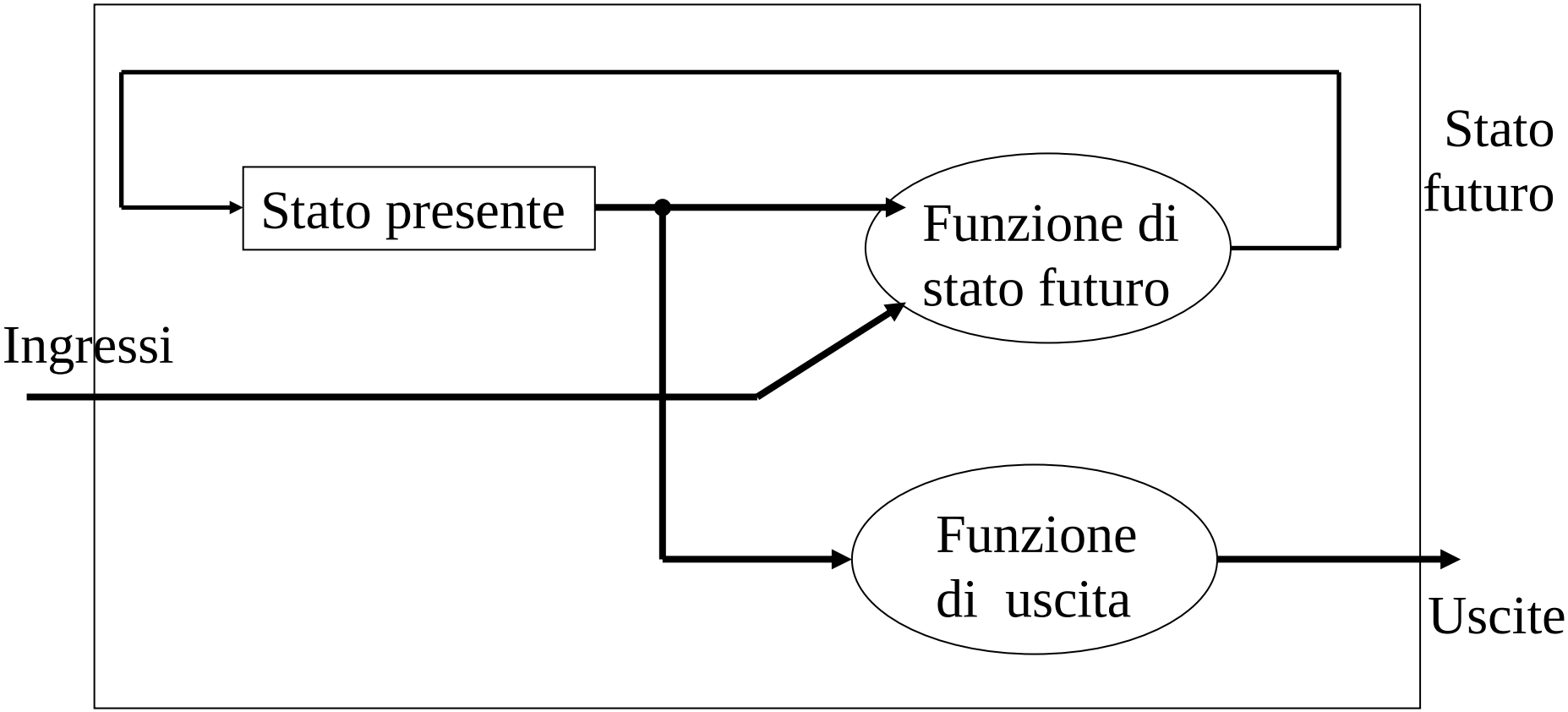
Macchine a stati finiti

Le macchine (automi) a stati finiti sono descrivibili mediante:

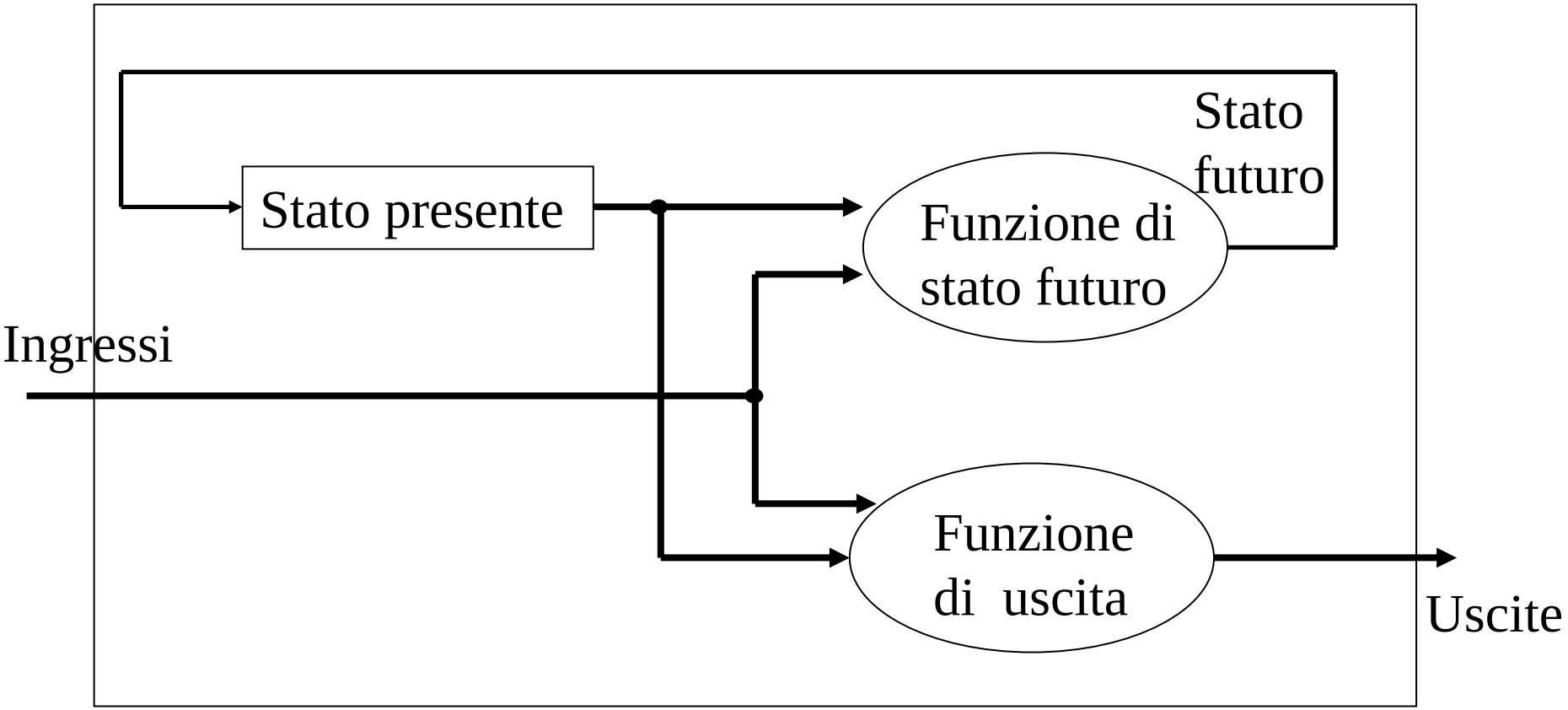
- **I** = $\{i_1, i_2, \ldots, i_p\}$ insieme dei simboli di ingresso
- **U** = $\{u_1, u_2, \ldots, u_r\}$ insieme dei simboli di uscita
- **S** = $\{s_1, s_2, \ldots, s_n\}$ insieme degli stati
- δ funzione di transizione che specifica il nuovo stato in base allo stato corrente e all'ingresso
- ω funzione di uscita che specifica il valore di uscita in base allo stato corrente e all'ingresso
- Automa A = <I, U, S, δ, ω>

Questi 5 insiemi possono essere descritti anche graficamente

Macchina sequenziale di Moore



Macchina sequenziale di Mealy



Macchine di Moore e di Mealy

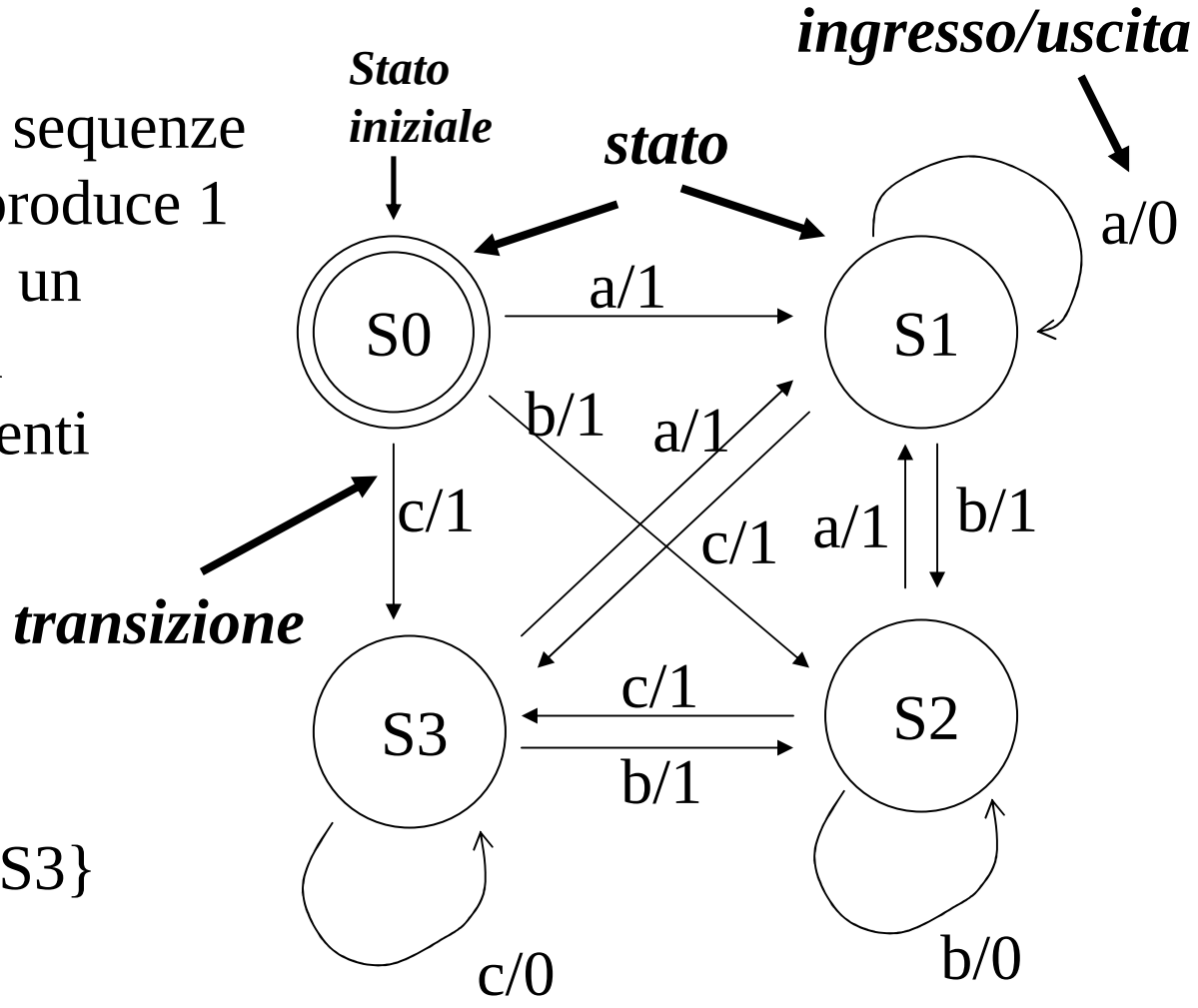
- Le **macchine di Mealy** hanno, per ogni arco, un simbolo di entrata e uno di uscita.
- Nelle **macchine di Moore** l'uscita è invece già codificata nel valore dello **stato** in cui si trova la macchina, ovvero la funzione di uscita dipende solo da S anziché da $I \times S$.
- È possibile trasformare una macchina di Mealy in una di Moore e viceversa, solitamente quelle di Moore hanno più stati.

Macchine di Moore e di Mealy

- In una macchina di Mealy si ha una **uscita *durante ciascuna transizione.***
- In una macchina di Moore si ha una **uscita *dopo che è avvenuta la transizione*** e questa uscita dipende solo dal nuovo stato della macchina.
- Moore produce quindi un primo output, associato allo stato iniziale, non presente in un automa di Mealy.

Esempio di macchina a stati finiti

Macchina che legge sequenze di simboli *a*, *b*, *c* e produce 1 ogni volta che legge un carattere diverso dal precedente, 0 altrimenti



I = {a, b, c}

U = {0, 1}

S = {S0, S1, S2, S3}

Esempio di funzionamento

Tabella degli stati

s	a	b	c
S0	S1, 1	S2,1	S3,1
S1	S1, 0	S2,1	S3,1
S2	S1, 1	S2,0	S3,1
S3	S1, 1	S2,1	S3,0

stato prossimo, uscita

L'automa si trova inizialmente nello stato S0 e riceve in ingresso la sequenza *abccbaabb*. Cosa succede?

seq. ingr. *a b c c b a a b b*

seq. stati $S_1 S_2 S_3 S_3 S_2 S_1 S_1 S_2 S_2$

seq. usc. 1 1 1 0 1 1 0 1 0

Esempio
Doppia lampada con 3 modalità di accensione

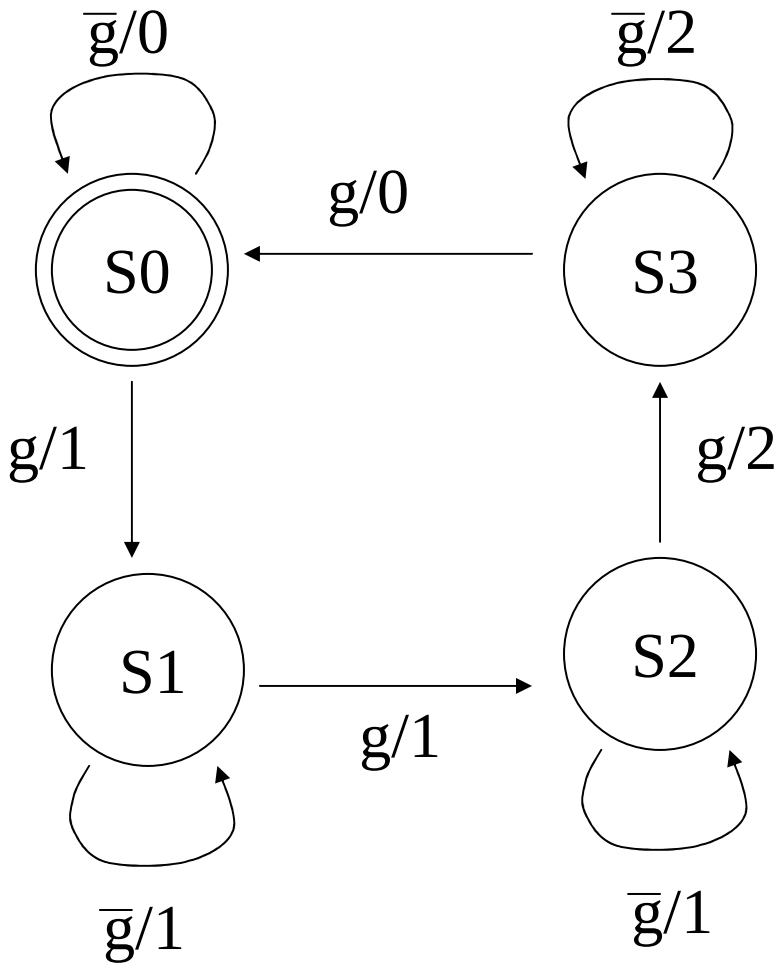
Quando la lampada è spenta e si gira l’interruttore si accende la lampadina sinistra; se si gira di nuovo l’interruttore si accende la lampadina destra e si spegne quella sinistra; se si gira l’interruttore ancora una volta le due lampadine si accendono entrambe; infine, con un ultimo giro, le lampadine si spengono entrambe.

Macchina per la doppia lampada

Stati = {S0, S1, S2, S3}
S0 = nessuna lampadina accesa
S1 = lampadina sx accesa
S2 = lampadina dx accesa
S3 = tutte e due accese
Ingressi = {g, $\overline{g}$}
Uscite = {0, 1, 2}

	g	$\overline{g}$
S0	S1, 1	S0, 0
S1	S2, 1	S1, 1
S2	S3, 2	S2, 1
S3	S0, 0	S3, 2

Tabella degli stati



Contatori

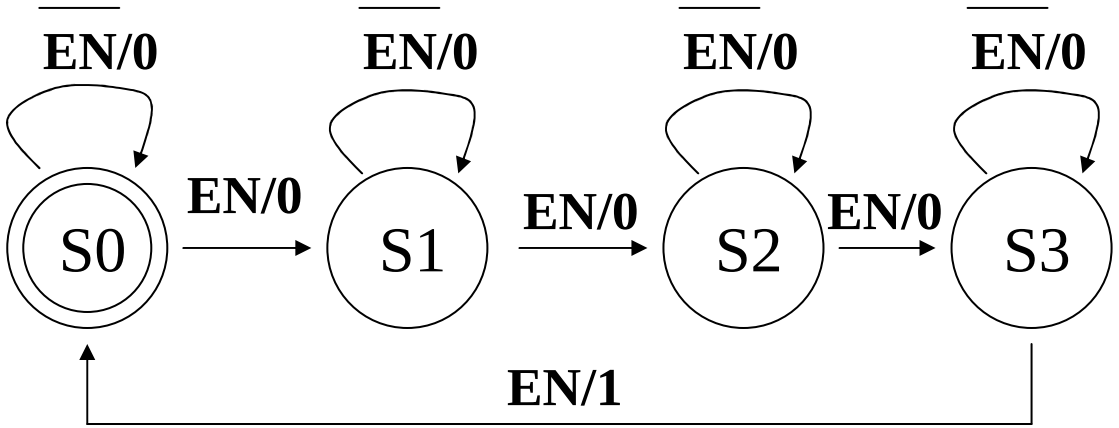
- Un contatore è un dispositivo sequenziale che aggiorna periodicamente il suo **stato** secondo una regola che riprende il susseguirsi dei numeri naturali, ad esempio:

  0000 – 0001 – 0010 – 0011 – 0100 - …

- Prendiamo qui in considerazione solo contatori sincroni, ovvero che aggiornano il loro stato **ad ogni ciclo di clock**
- **Contatore modulo k**: conta da 0 a k-1
- Un contatore modulo k può essere usato come un “demoltiplicatore” del segnale orologio: produce un segnale di uscita pari a 1 ogni k cicli di clock

Esempio: Contatore modulo 4

(Usato come demoltiplicatore del segnale orologio)

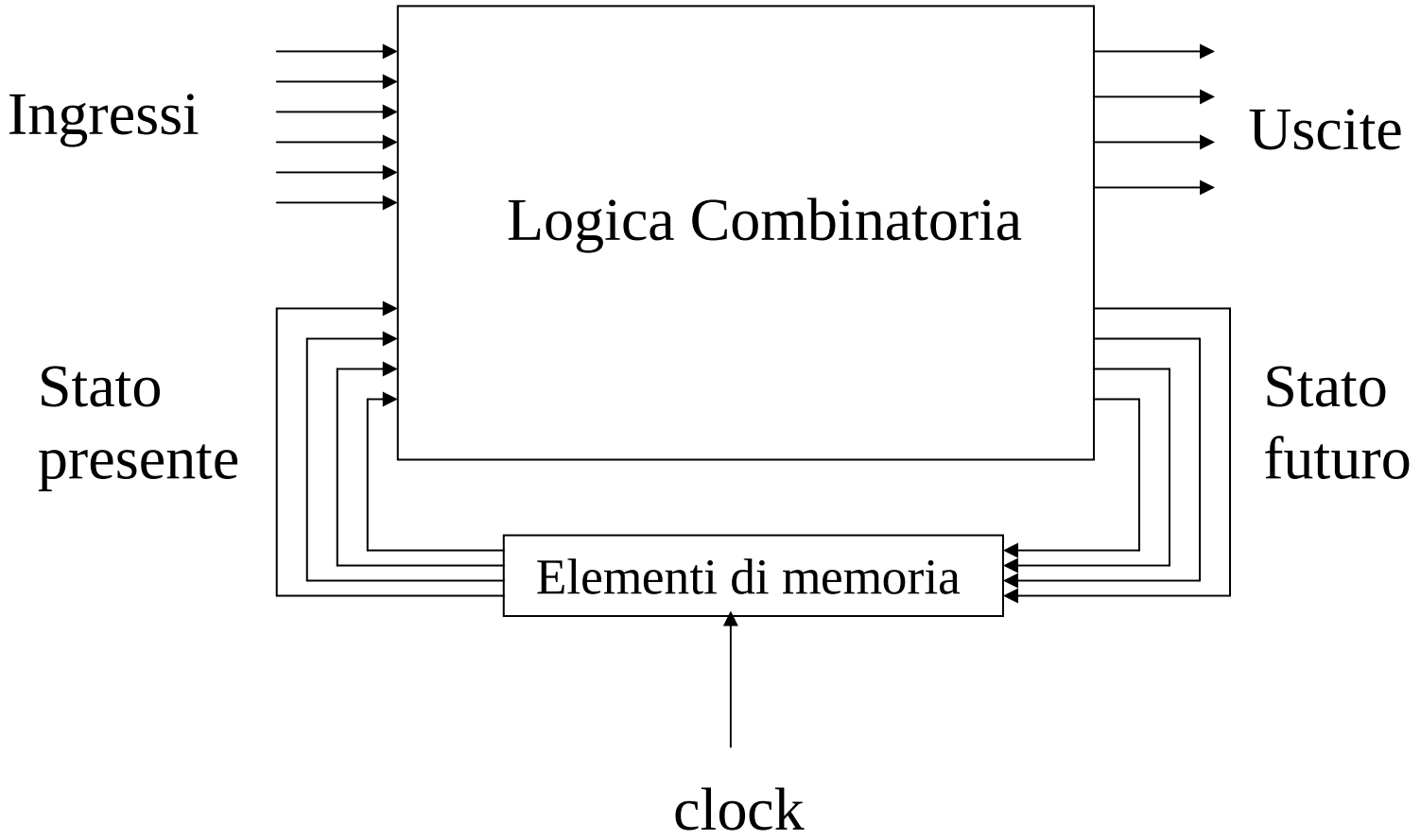


Stati = {S0, S1, S2, S3}
S0 : conteggio = 0
S1 : conteggio = 1
S2 : conteggio = 2
S3 : conteggio = 3
Ingressi = {EN} segnale di abilitazione
Uscite = {0, 1}

Sono necessari 2 bit per codificare gli stati e le combinazioni corrispondono esattamente a 0, 1, 2, 3 in binario

S0 $\Rightarrow$ 00
S1 $\Rightarrow$ 01
S2 $\Rightarrow$ 10
S3 $\Rightarrow$ 11

Sintesi di una macchina sequenziale

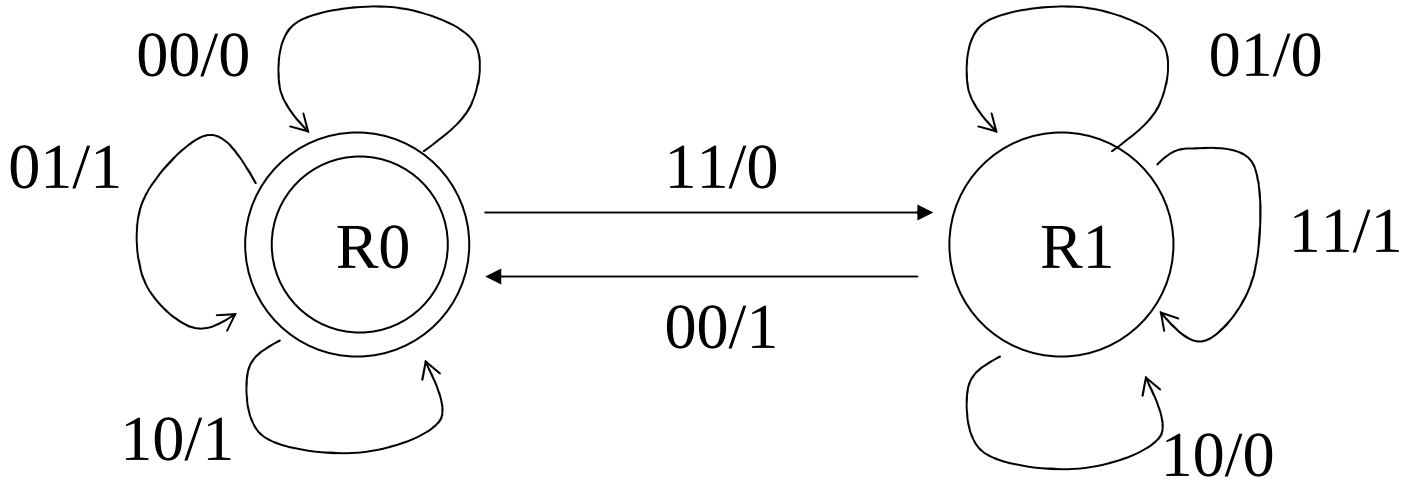


Sintesi di una macchina sequenziale
Procedura

1. Specificare la macchina a stati finiti o la **tabella degli stati**
2. Definire una **codifica per i simboli di entrata e per i simboli di uscita**
3. Determinare il numero dei flip-flop necessari in base al numero degli stati e i valori da memorizzare nei flip-flop (**assegnamento degli stati**)
4. Derivare la **tabella delle transizioni e delle uscite**
5. Ricavare le formule per le variabili di stato a seconda del tipo di flip-flop scelto (**funzione di stato prossimo**)
6. Ricavare le formule per le variabili di uscita (**funzione di uscita**)
7. Realizzare la rete combinatoria che sintetizza le funzioni suddette.

Esempio
Sommatore sequenziale a 2 stati

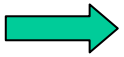
- Vengono sommate coppie di bit a_i e b_i di due operandi
  I = {00, 01, 10, 11}
- L'uscita c_i appartiene a U = {0, 1}
- Si distinguono 2 stati: "riporto presente" (R1) e "riporto assente" (R0)



Esempio

Sommatore sequenziale a 2 stati

Tabella degli stati corrispondente all'automa



stato \ $a_i b_i$	00	01	10	11
R0	R0,0	R0,1	R0,1	R1,0
R1	R0,1	R1,0	R1,0	R1,1

Assegnamento degli stati

Stato	*y*
R0	0
R1	1

Usiamo un **flip-flop di tipo D** che memorizza il valore della variabile di stato *y* nell'intervallo compreso fra due impulsi di clock successivi

Tabella delle transizioni e delle uscite

a_i	b_i	y	y'	c_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Formula per il flip-flop D

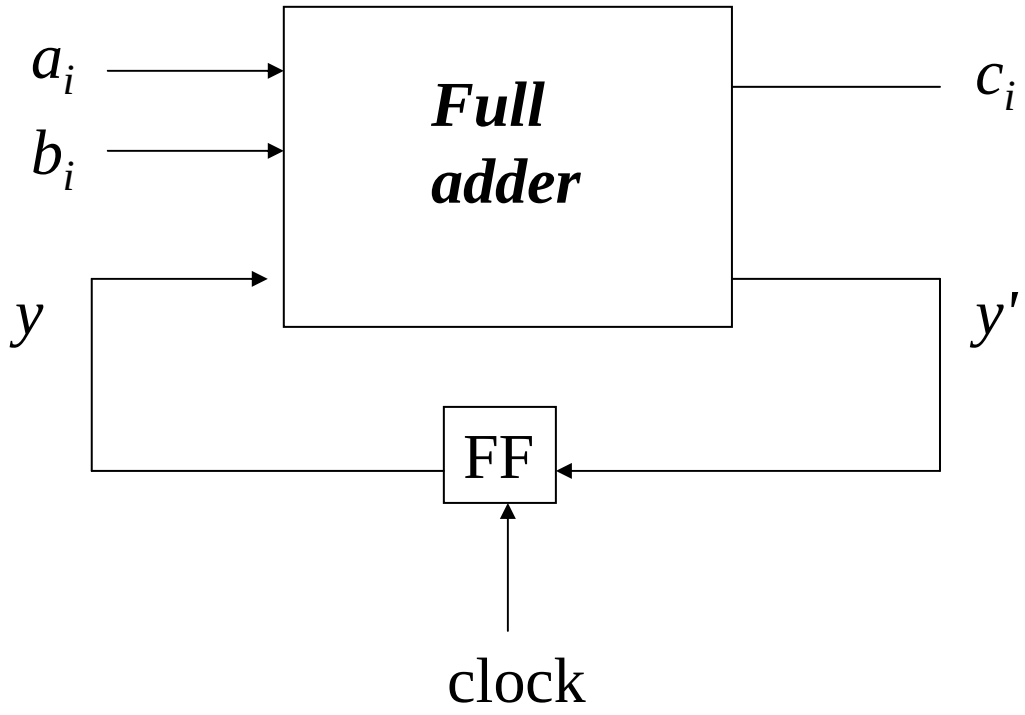
$$y' = b_i y + a_i y + a_i b_i$$

Formula per la variabile di uscita

$$c_i = \bar{a}_i \bar{b}_i y + a_i \bar{b}_i \bar{y} + a_i \bar{b}_i y + a_i b_i y$$

Sono le espressioni delle uscite di un full adder

Macchina sequenziale per un sommatore a 2 stati

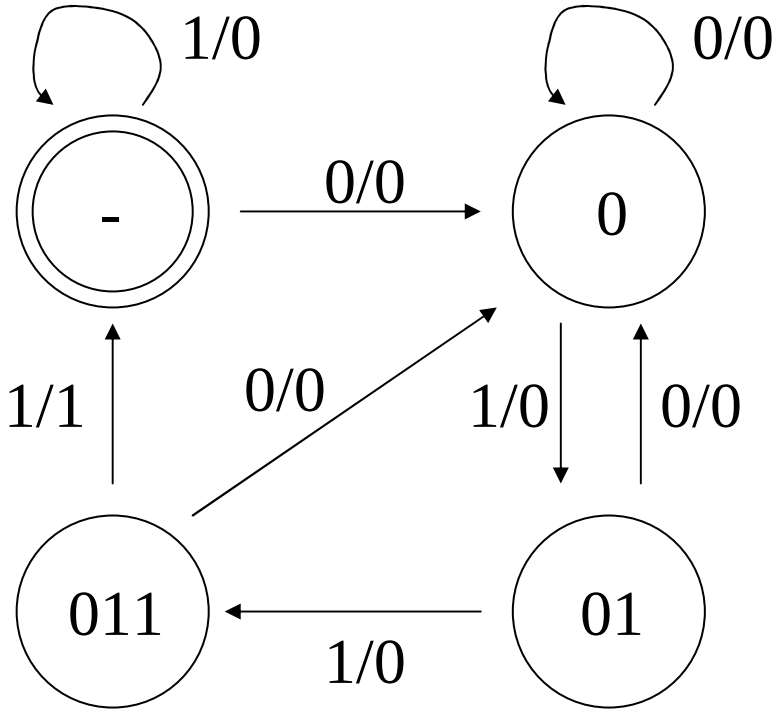


Quali sono le differenze rispetto alla macchina combinatoria per la somma a *n* bit?

Riconoscitore di sequenze

- Macchina che legge sequenze di 0 e 1 di lunghezza arbitraria (I={0,1})
- Produce in uscita 1 ogni volta che ha riconosciuto la sequenza **0111** (U= {0,1})
- Identifichiamo **4 stati** che indichiamo con “-”, “0”, “01” e “011”: i nomi rappresentano la parte della sottosequenza finora riconosciuta
- Lo stato “-” è lo stato iniziale: nessun carattere della sottosequenza cercata è stato riconosciuto

Automa riconoscitore delle sequenze 0111



Per 4 stati bastano 2 flip-flop (tipo D)

Facciamo il seguente assegnamento degli stati

Stato	y_1	y_0
"-"	0	0
"0"	0	1
"01"	1	0
"011"	1	1

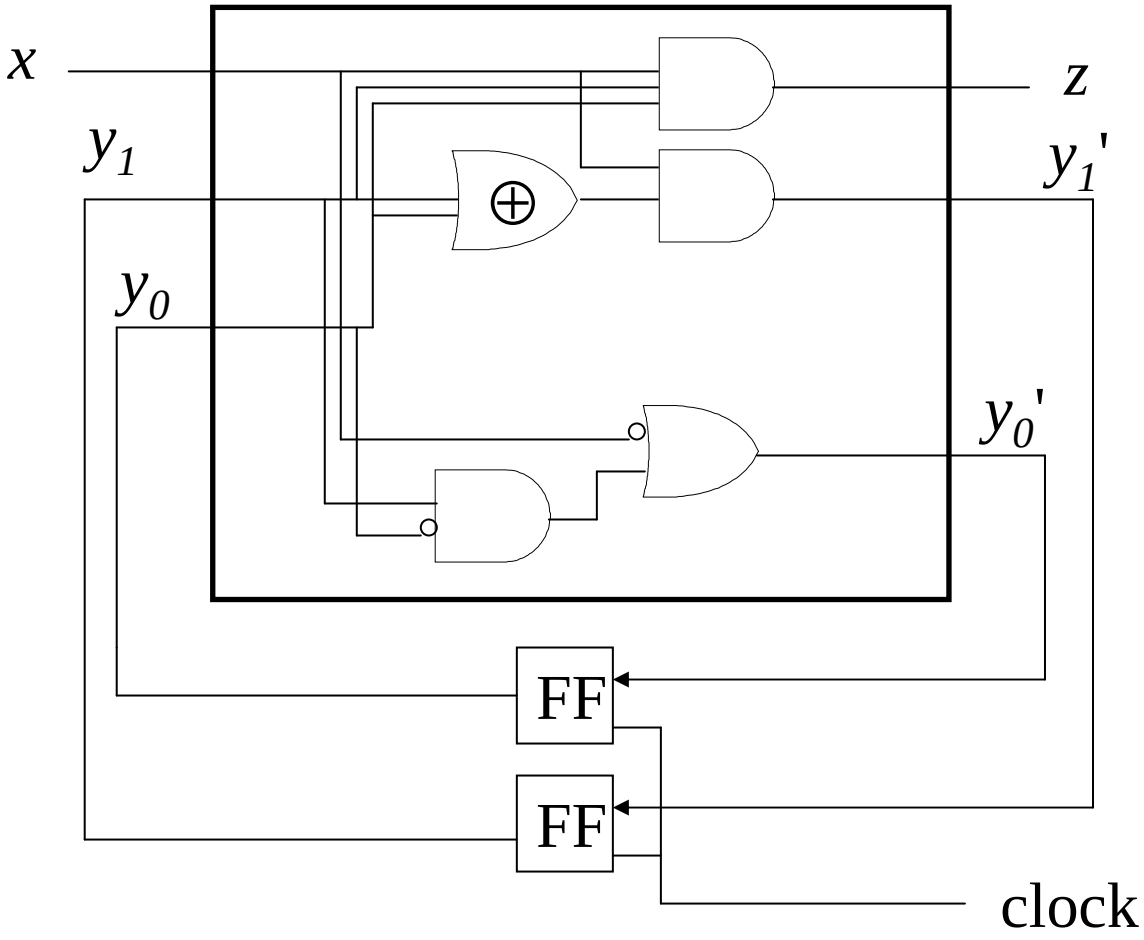
Tabella delle transizioni e delle uscite

Stato presente		***Ingresso***	***Stato futuro***		***Uscita***
y_1	y_0	x	y_1'	y_0'	z
0	0	0	0	1	0
0	0	1	0	0	0
0	1	0	0	1	0
0	1	1	1	0	0
1	0	0	0	1	0
1	0	1	1	1	0
1	1	0	0	1	0
1	1	1	0	0	1

$$y_1' = \bar{y}_1 y_0 x + y_1 \bar{y}_0 x = x(y_1 \oplus y_0) \qquad z = y_1 y_0 x$$

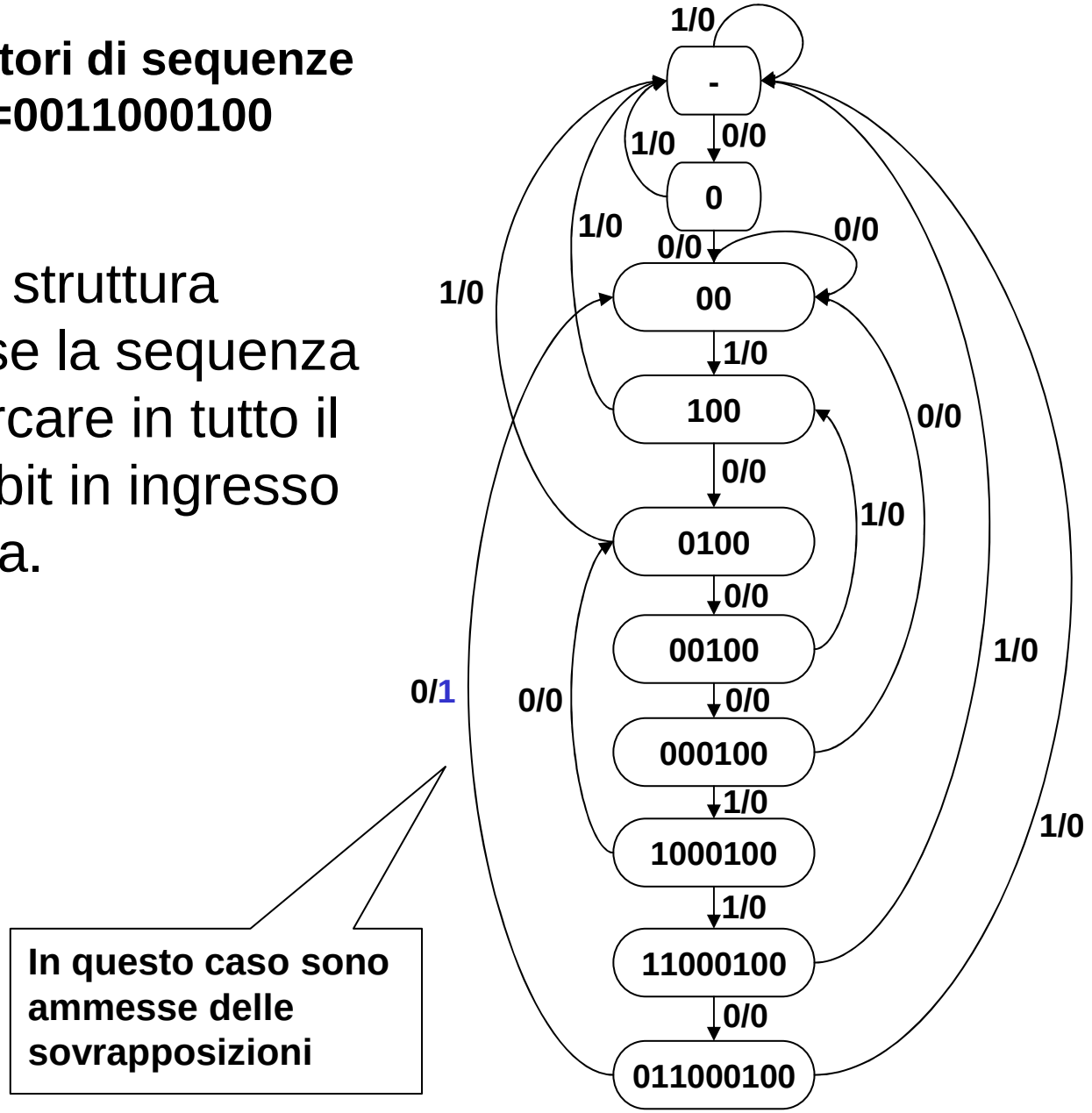
$$y_0' = \bar{y}_1 \bar{y}_0 \bar{x} + \bar{y}_1 y_0 \bar{x} + y_1 \bar{y}_0 \bar{x} + y_1 \bar{y}_0 x + y_1 y_0 \bar{x} = \bar{y}_1 \bar{x} + y_1 \bar{y}_0 + y_1 \bar{x} = \bar{x} + y_1 \bar{y}_0$$

Macchina sequenziale per il riconoscitore di sequenze



Riconoscitori di sequenze
Es.: S=0011000100

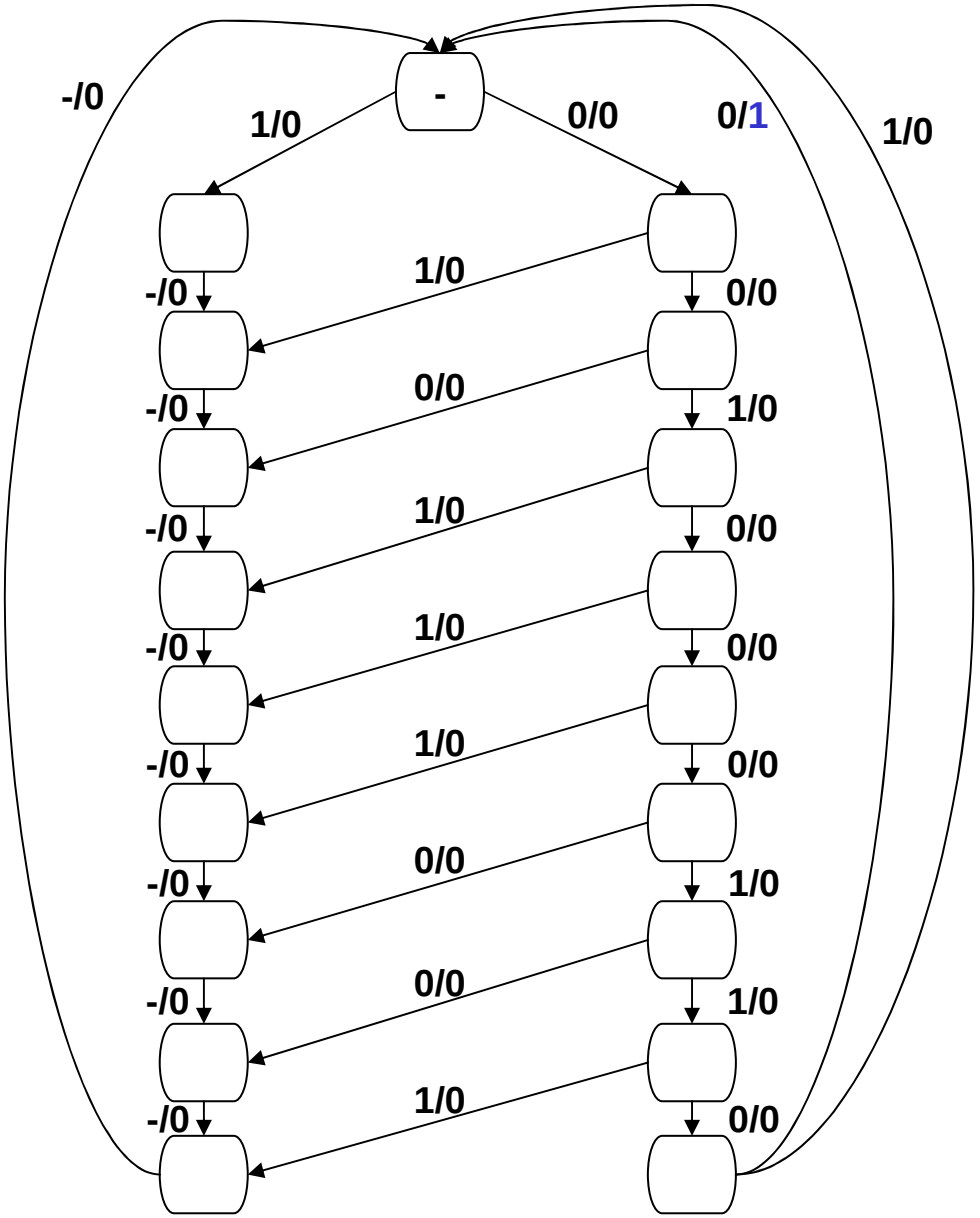
Hanno la struttura indicata se la sequenza è da ricercare in tutto il flusso di bit in ingresso all'automa.



Riconoscitori di sequenze
Es.: S=0011000100

Hanno la struttura indicata se la sequenza è da ricercare in blocchi consecutivi di n bit

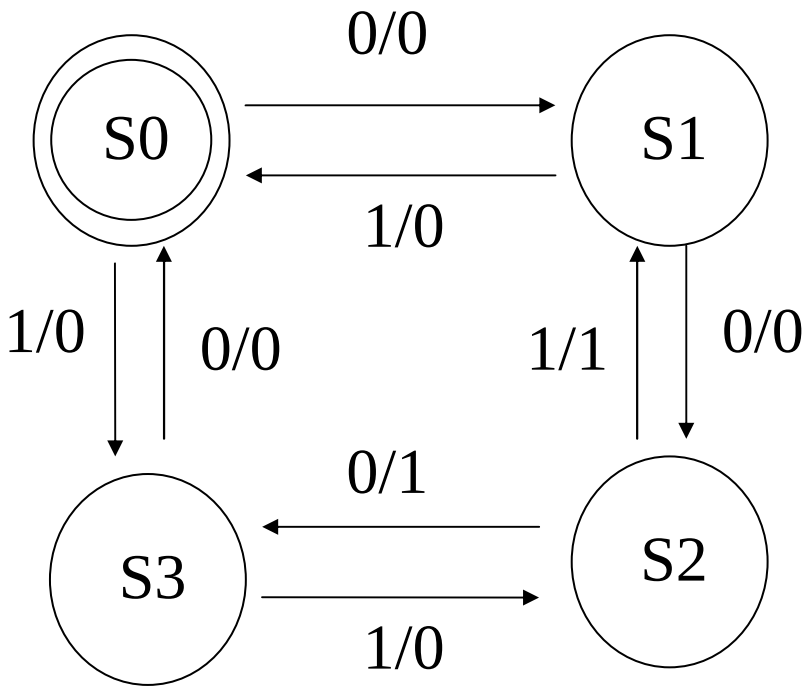
(es.: numeri trasmessi in maniera sequenziale).



Contatore bidirezionale

- Progettiamo un contatore modulo 4 che conta sia in avanti sia all'indietro, a seconda del valore della variabile di ingresso *x*
- Se *x* = 0 il circuito conta in avanti, se *x* = 1 il circuito conta all'indietro
- Se il valore del conteggio è 2 la variabile di uscita *z* viene posta a 1 (quando viene lasciato lo stato corrispondente a 2) altrimenti viene posta a 0
- Ad ogni impulso di clock il circuito cambia stato in base al valore di *x* e al valore attuale del conteggio

Macchina a stati finiti per il contatore bidirezionale



4 stati $\rightarrow$ 2 flip-flop

Stato	y_1	y_0
S0	0	0
S1	0	1
S2	1	0
S3	1	1

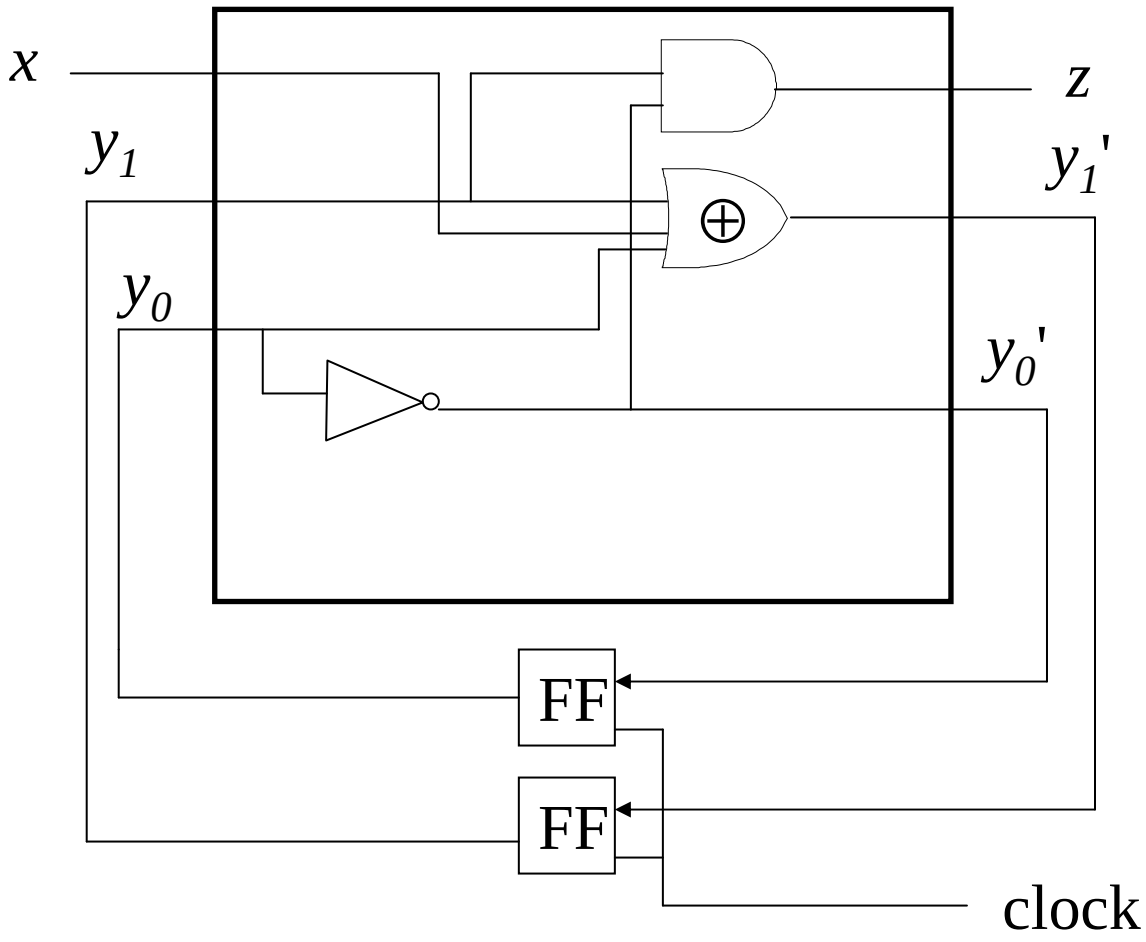
Tabella delle transizioni e delle uscite

Stato presente		***Ingresso***	***Stato futuro***		***Uscita***
y_1	y_0	x	y_1'	y_0'	z
0	0	0	0	1	0
0	0	1	1	1	0
0	1	0	1	0	0
0	1	1	0	0	0
1	0	0	1	1	1
1	0	1	0	1	1
1	1	0	0	0	0
1	1	1	1	0	0

$$z = y_1\overline{y_0} \qquad y_0' = \overline{y}_1\overline{y}_0 + y_1\overline{y}_0 = \overline{y}_0$$

$$y_1' = \overline{y}_1\overline{y}_0 x + \overline{y}_1 y_0 \overline{x} + y_1\overline{y}_0\overline{x} + y_1 y_0 x = y_1 \oplus y_0 \oplus x$$

Macchina sequenziale per il contatore bidirezionale



Automa per la doppia lampada

Stati = {S0, S1, S2, S3}
S0 = nessuna lampadina accesa
S1 = lampadina sx accesa
S2 = lampadina dx accesa
S3 = tutte e due accese

Ingressi = {g, $\overline{g}$}

Uscite = {0, 1, 2}

Codifica degli stati
S0 ⟶ 00
S1 ⟶ 01
S2 ⟶ 10
S3 ⟶ 11

Per 4 stati sono necessari 2 flip-flop:
per gli stati usiamo le variabili y_1 e y_0

Codifica degli ingressi = {0, 1}
per l'ingresso usiamo la variabile *x*

Codifica delle uscite = {00, 01, 10}
Per le uscite usiamo le variabili z_1 e z_0

Automa per la doppia lampada

	x 0	1
S0	S0, 00	S1, 01
S1	S1, 01	S2, 01
S2	S2, 01	S3, 10
S3	S3, 10	S0, 00

Tabella degli stati

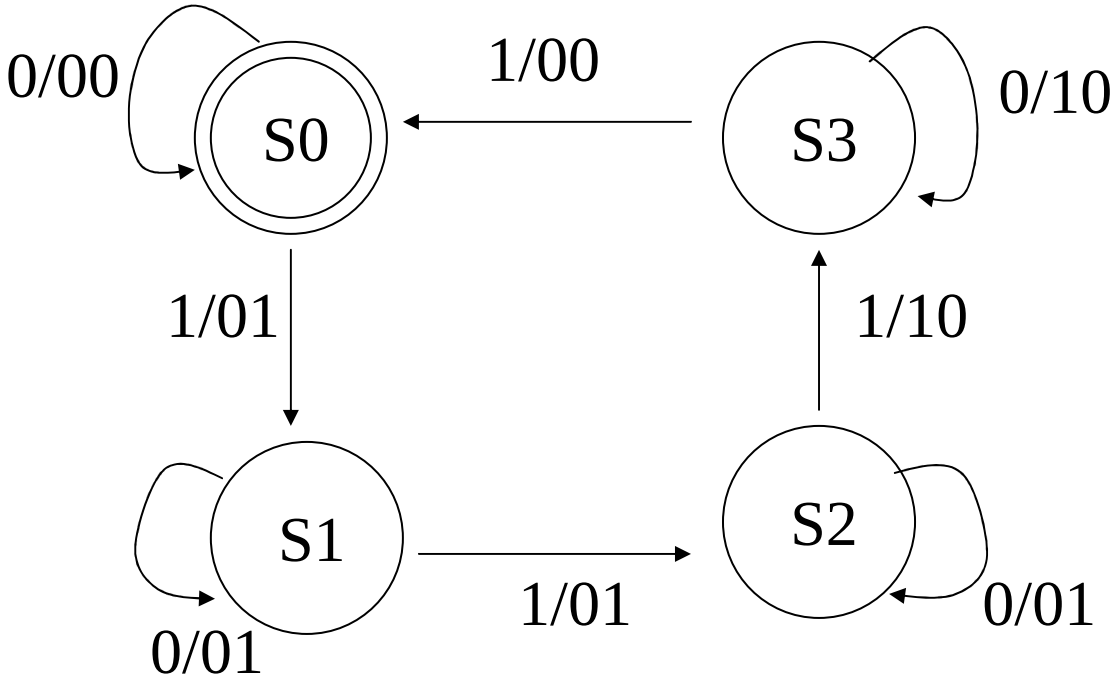


Tabella delle transizioni e delle uscite per la doppia lampada

Stato presente		*Ingressi*	*Stato prossimo*		*Uscite*	
y_1	y_0	x	y_1'	y_0'	z_1	z_0
0	0	0	0	0	0	0
0	0	1	0	1	0	1
0	1	0	0	1	0	1
0	1	1	1	0	0	1
1	0	0	1	0	0	1
1	0	1	1	1	1	0
1	1	0	1	1	1	0
1	1	1	0	0	0	0

$y_0' = \overline{y}_1\overline{y}_0x + \overline{y}_1y_0\overline{x} + y_1\overline{y}_0x + y_1y_0\overline{x}$

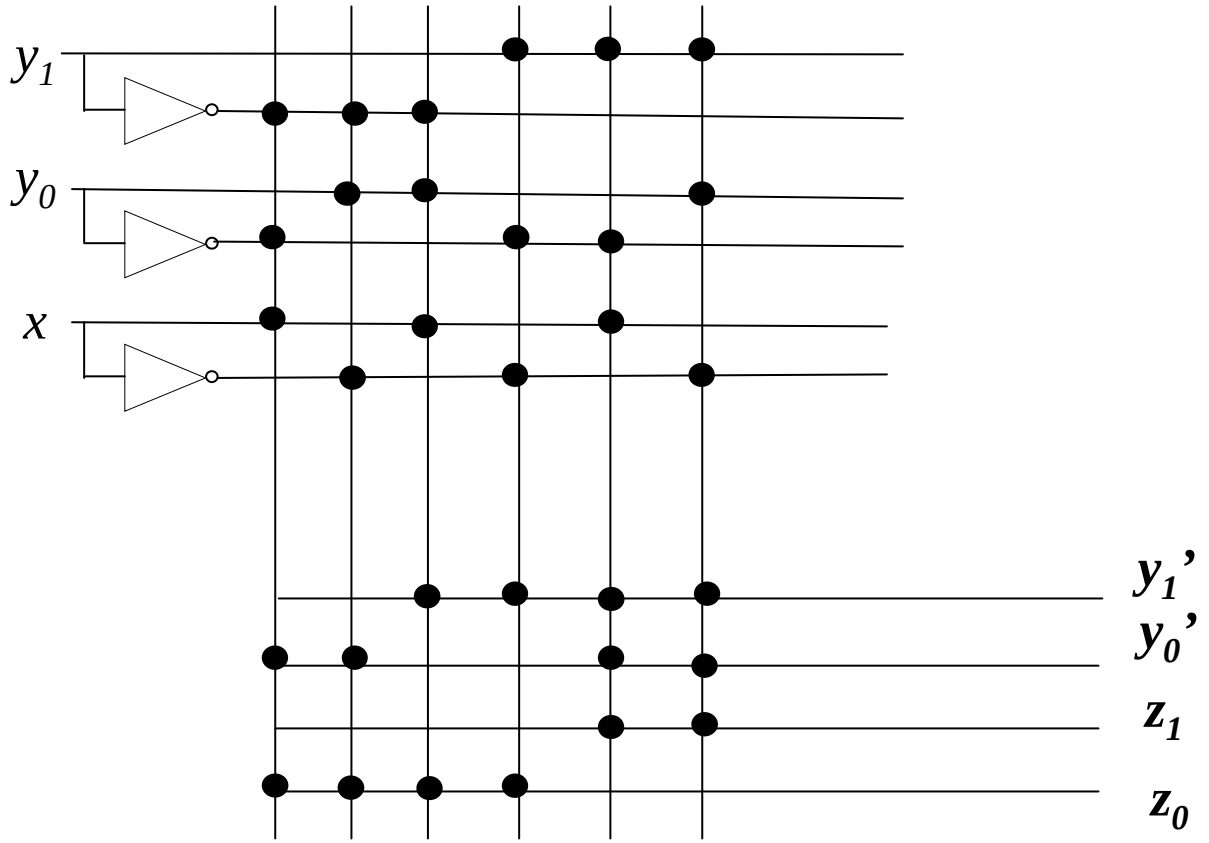
6 mintermini

$y_1' = \overline{y}_1y_0x + y_1\overline{y}_0\overline{x} + y_1\overline{y}_0x + y_1y_0\overline{x}$

$z_0 = \overline{y}_1\overline{y}_0x + \overline{y}_1y_0\overline{x} + \overline{y}_1y_0x + y_1\overline{y}_0\overline{x}$

$z_1 = y_1\overline{y}_0x + y_1y_0\overline{x}$

PLA che realizza la logica combinatoria



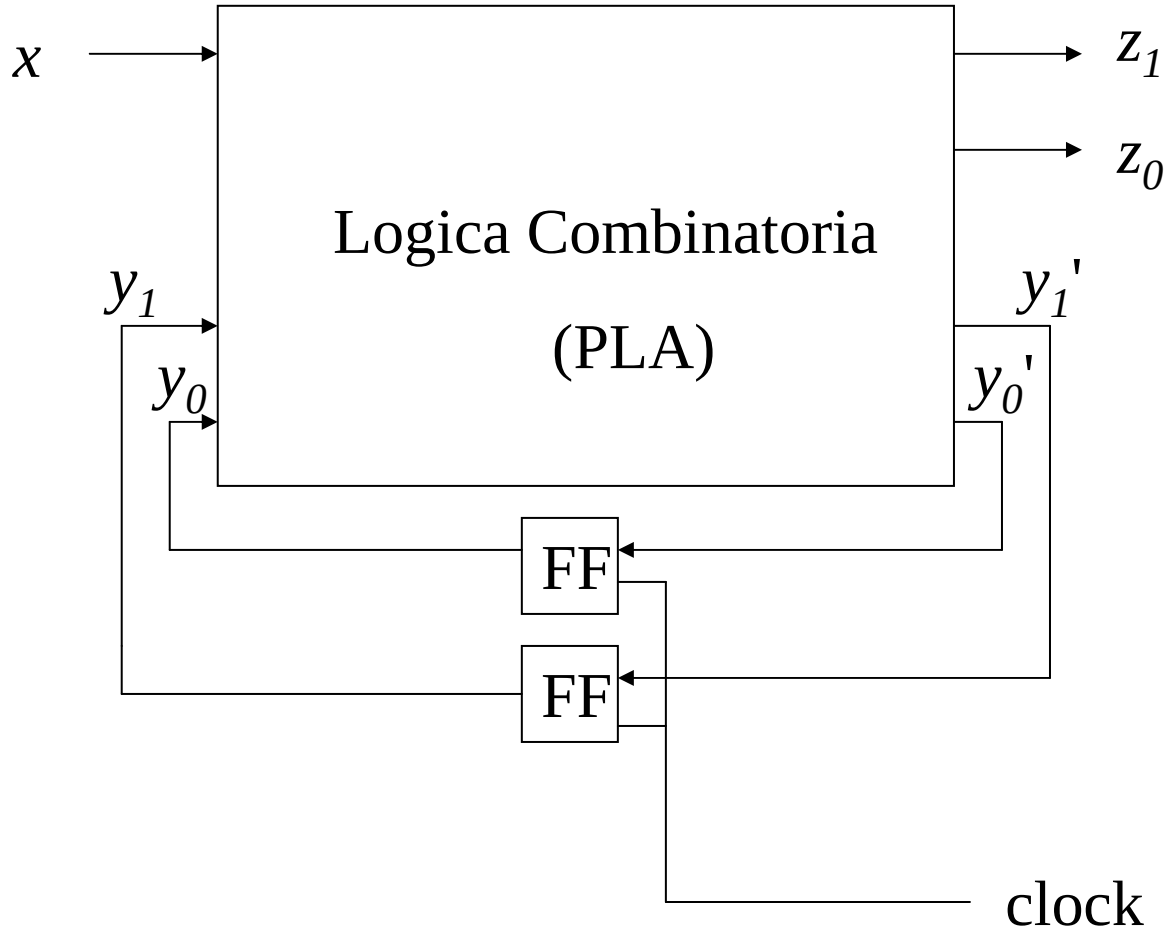
$$y_0' = \overline{y}_1\overline{y}_0x + \overline{y}_1y_0\overline{x} + y_1\overline{y}_0x + y_1y_0\overline{x}$$

$$y_1' = \overline{y}_1y_0x + y_1\overline{y}_0\overline{x} + y_1\overline{y}_0x + y_1y_0\overline{x}$$

$$z_0 = \overline{y}_1\overline{y}_0x + \overline{y}_1y_0\overline{x} + \overline{y}_1y_0x + y_1\overline{y}_0\overline{x}$$

$$z_1 = y_1\overline{y}_0x + y_1y_0\overline{x}$$

Macchina sequenziale per la doppia lampada



Macchina distributrice

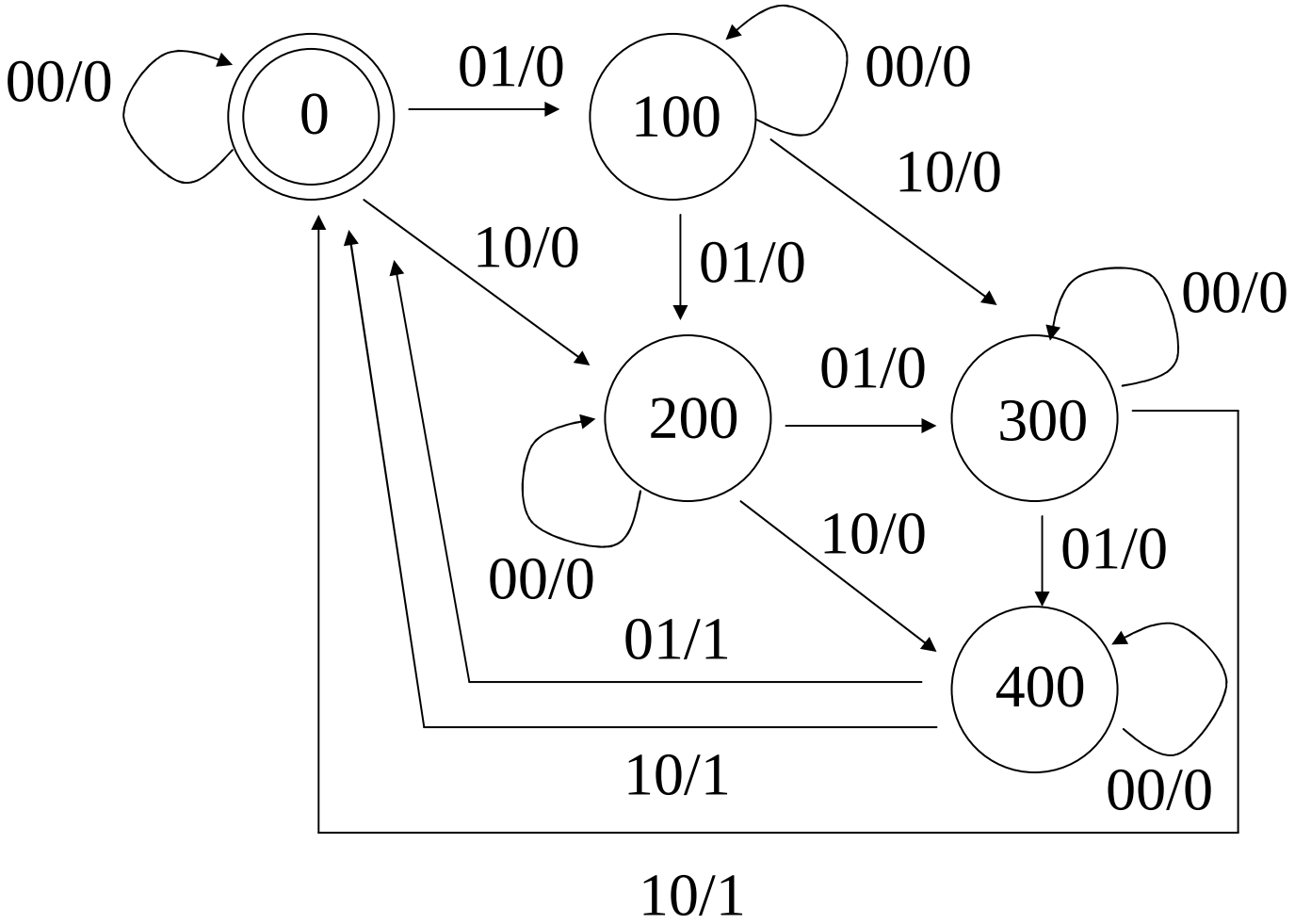
- La macchina accetta monete da 100 e 200
- **Codifichiamo gli ingressi** con le variabili x_1 e x_2

$x_2 x_1$	
0 0	nessuna moneta
0 1	moneta da 100
1 0	moneta da 200

I = {00, 01, 10}

- Fornisce il prodotto quando viene raggiunto l'importo di 500 (uscita 1)
- Convenzione: chiamiamo ogni stato della macchina con l'importo raggiunto

Automa per macchina distributrice



Assegnamento degli stati

- Per 5 stati sono necessari 3 flip-flop (tipo D): y_2, y_1, y_0
- Facciamo il seguente **assegnamento degli stati**

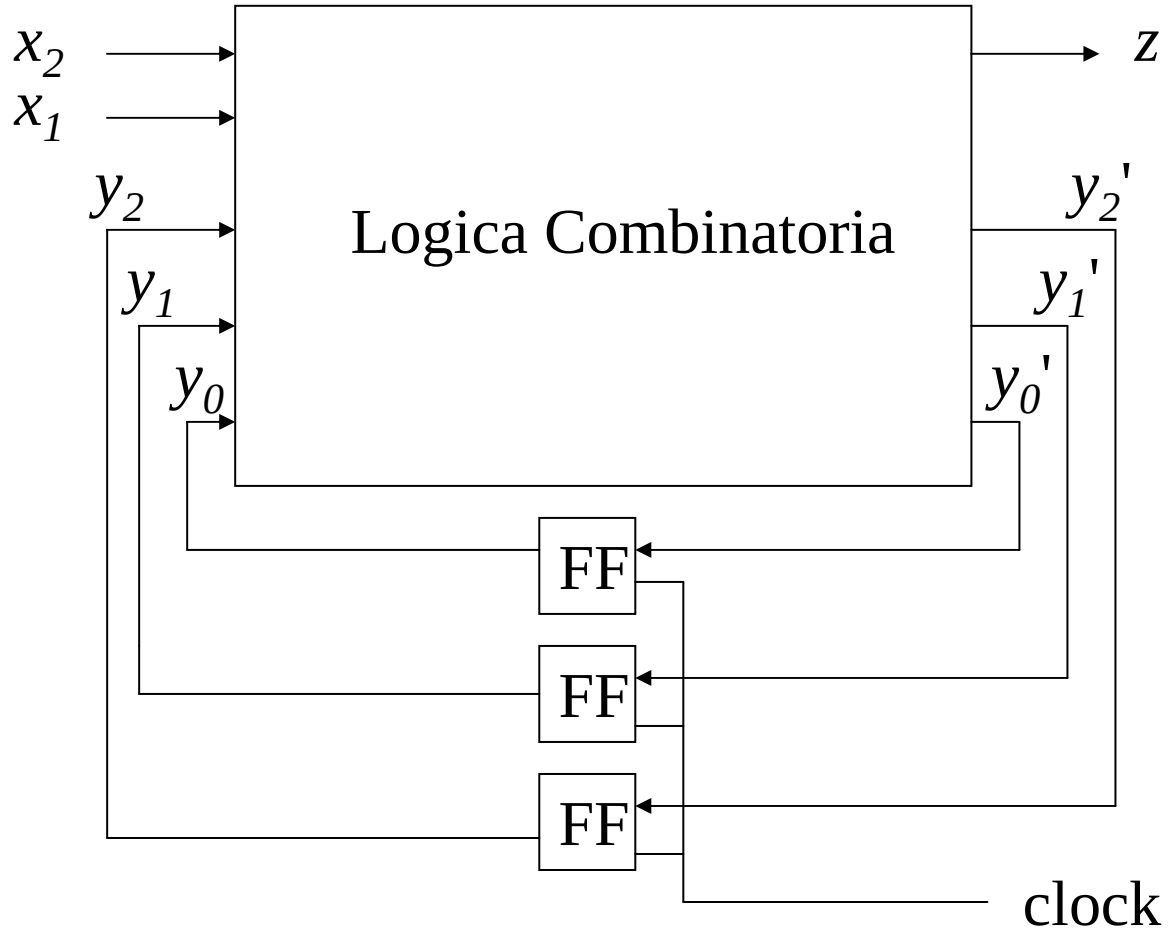
Stato	y_2	y_1	y_0
“0”	0	0	0
“100”	0	0	1
“200”	0	1	0
“300”	0	1	1
“400”	1	0	0

- Deriviamo dall’automa la tabella delle transizioni e delle uscite

Tabella delle transizioni e delle uscite

Stato presente			Ingressi		Stato prossimo			Uscite
y_2	y_1	y_0	x_2	x_1	y_0'	y_1'	y_2'	z
0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	1	0
0	0	0	1	0	0	1	0	0
0	0	1	0	0	0	0	1	0
0	0	1	0	1	0	1	0	0
0	0	1	1	0	0	1	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	1	0	0	0
0	1	1	0	0	0	1	1	0
0	1	1	0	1	1	0	0	0
0	1	1	1	0	0	0	0	1
1	0	0	0	0	1	0	0	0
1	0	0	0	1	0	0	0	1
1	0	0	1	0	0	0	0	1

Macchina sequenziale risultante

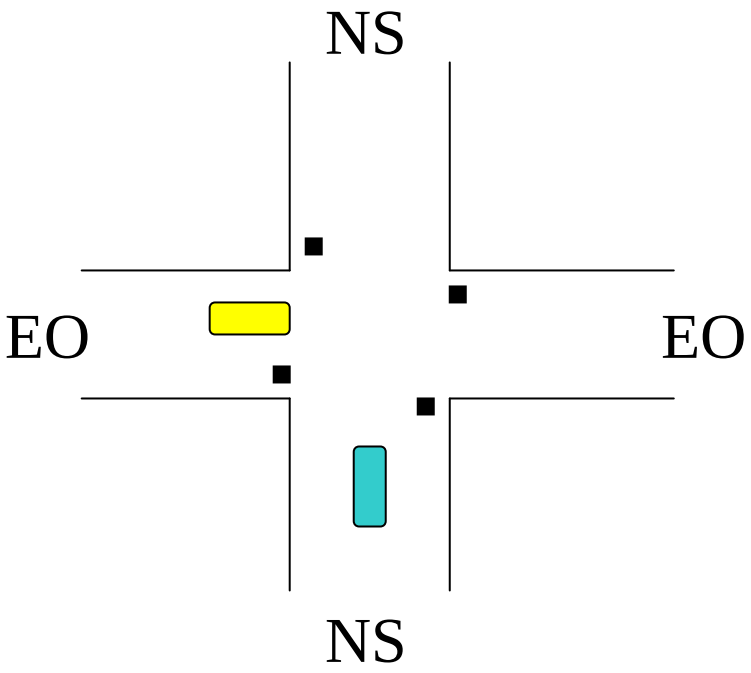


Esercizi

1. Realizzare la logica combinatoria per la macchina distributrice (ad es. con una PLA)

2. Realizzare la logica combinatoria della macchina sequenziale corrispondente alla doppia lampada con circuiti and/or e con una ROM

Macchina per il controllo di un semaforo



- Consideriamo solo le luci verde e rossa
- Ingressi = {AutoNS, AutoEO}
- Uscite = {LuceNS, LuceEO}
- Stati = {VerdeNS, VerdeEO}

Tabella degli stati

Stato \ *Ingresso*	00	01	10	11
VerdeNS	VerdeNS,10	VerdeEO,01	VerdeNS,10	VerdeEO,01
VerdeEO	VerdeEO,01	VerdeEO,01	VerdeNS,10	VerdeNS,10

Configurazioni di ingresso

AutoNS	AutoEO
0	0
0	1
1	0
1	1

Configurazioni di uscita

LuceNS	LuceEO
0	1
1	0

Assegnamento degli stati

	y
VerdeNS	0
VerdeEO	1

Macchina di Mealy per il controllo di un semaforo

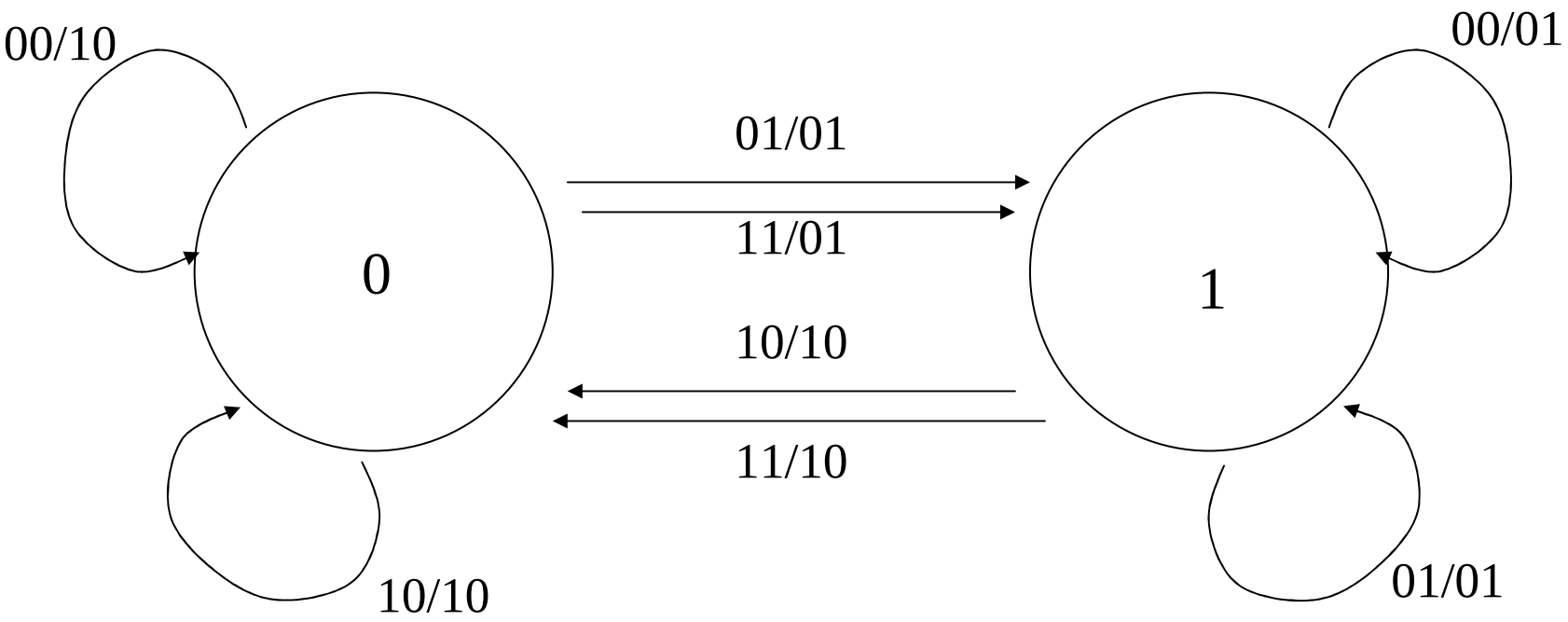


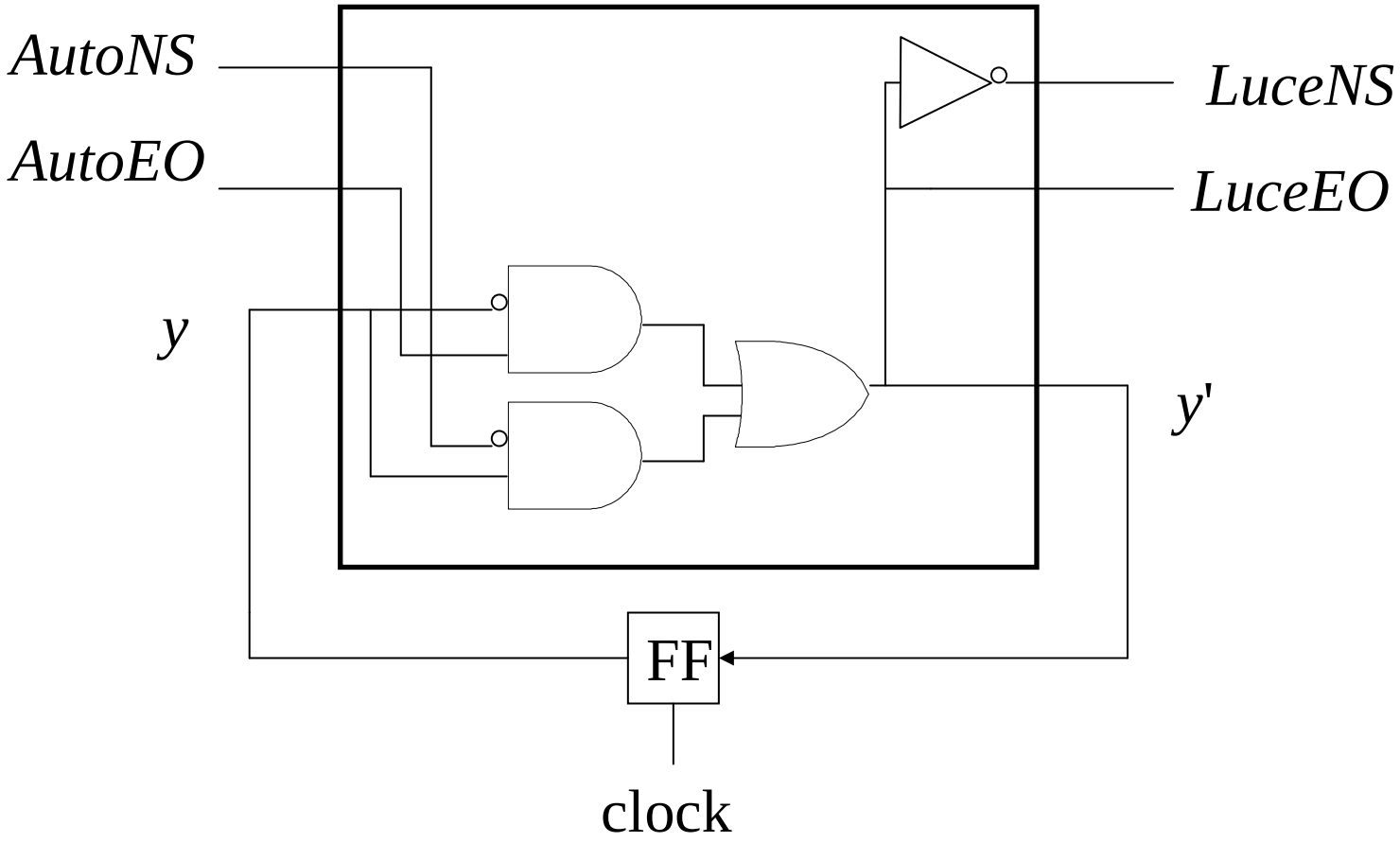
Tabella delle transizioni e delle uscite

Stato presente	***Ingressi***		***Stato prossimo***	***Uscite***	
y	***AutoNS***	***AutoEO***	***y'***	***LuceNS***	***LuceEO***
0	0	0	0	1	0
0	0	1	1	0	1
0	1	0	0	1	0
0	1	1	1	0	1
1	0	0	1	0	1
1	0	1	1	0	1
1	1	0	0	1	0
1	1	1	0	1	0

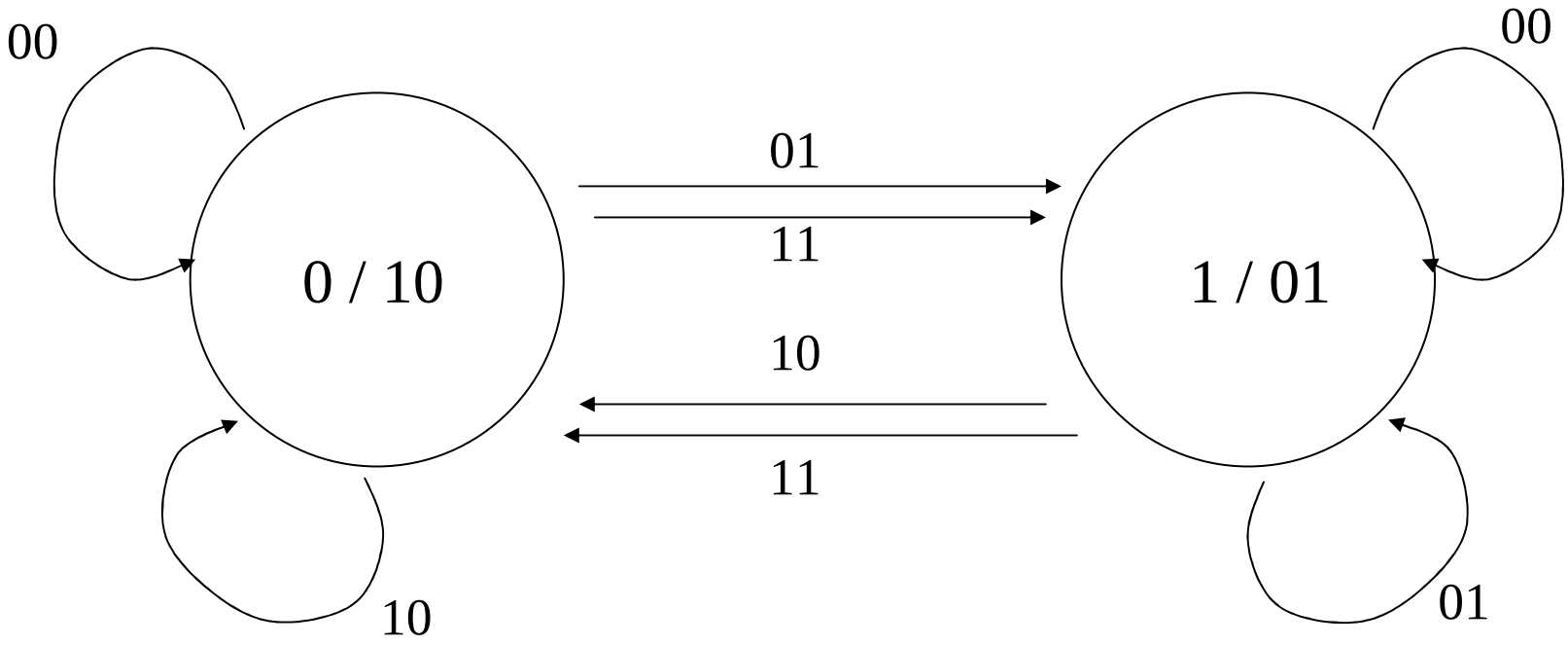
$$\boldsymbol{y'} = \overline{y}\,\overline{AutoNS}\,AutoEO + \overline{y}\,AutoNS\,AutoEO + y\,\overline{AutoNS}\,\overline{AutoEO} + y\,\overline{AutoNS}\,AutoEO = y\,\overline{AutoNS} + \overline{y}\,AutoEO$$

$$\boldsymbol{LuceEO} = y' \qquad \boldsymbol{LuceNS} = \overline{y'}$$

Macchina sequenziale per il controllo del semaforo relativa alla macchina di Mealy



Macchina di Moore per il controllo di un semaforo



Funzione di stato prossimo e funzione di uscita nel caso della macchina di Moore

Funzione di stato prossimo

Stato presente	***Ingressi***		***Stato prossimo***
y	***AutoNS***	***AutoEO***	***y'***
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

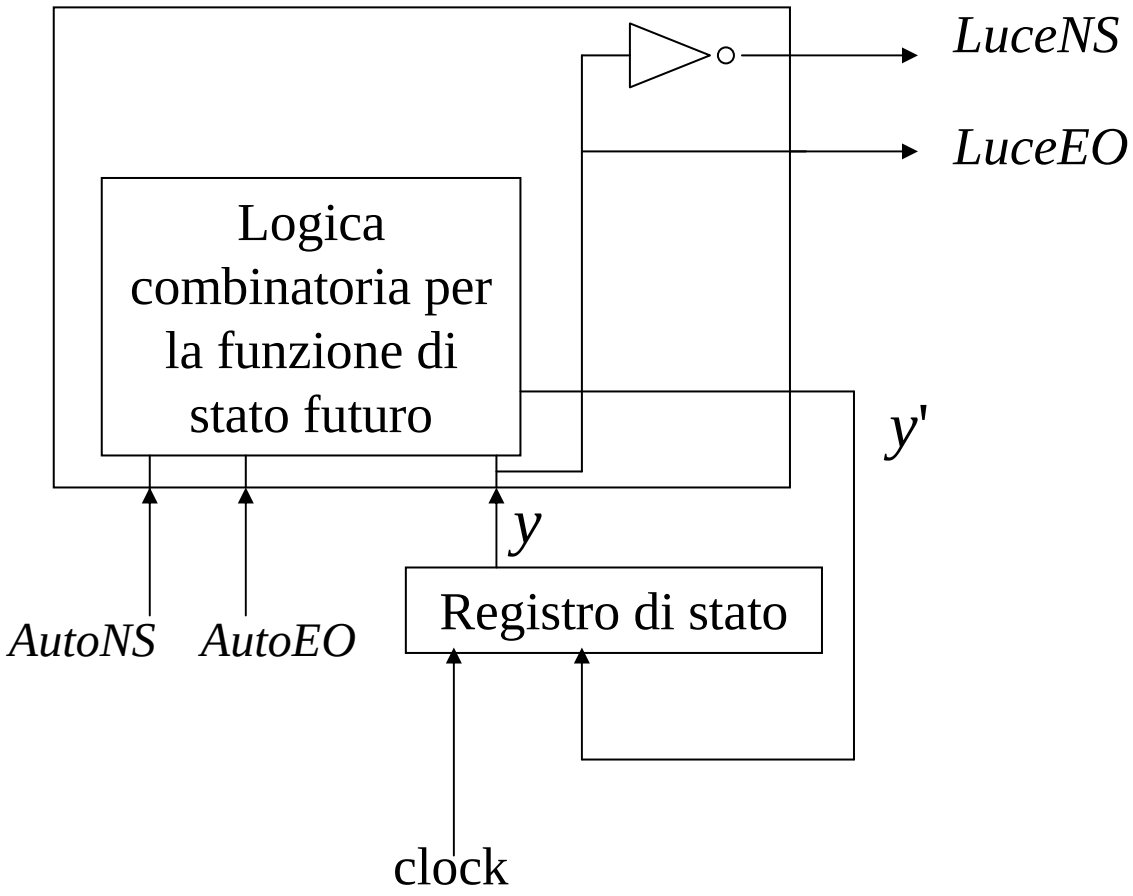
Funzione di uscita

Stato presente	***Uscite***	
y	***LuceNS***	***LuceEO***
0	1	0
1	0	1

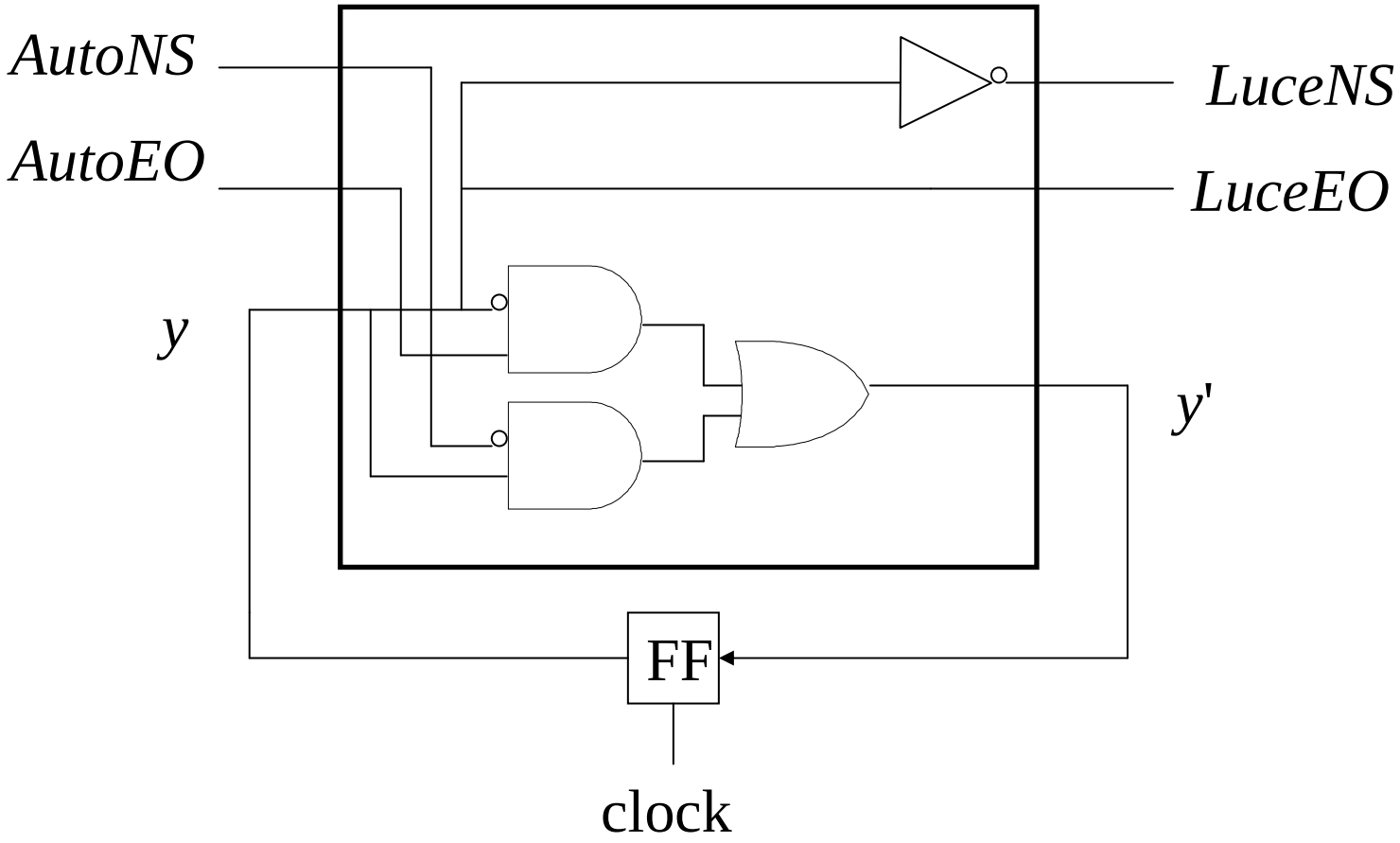
$$luceNS = \overline{y}$$
$$luceEO = y$$

$$y' = \overline{yAutoNS} + \overline{y}AutoEO$$

Macchina sequenziale per il controllo del semaforo relativa alla macchina di Moore



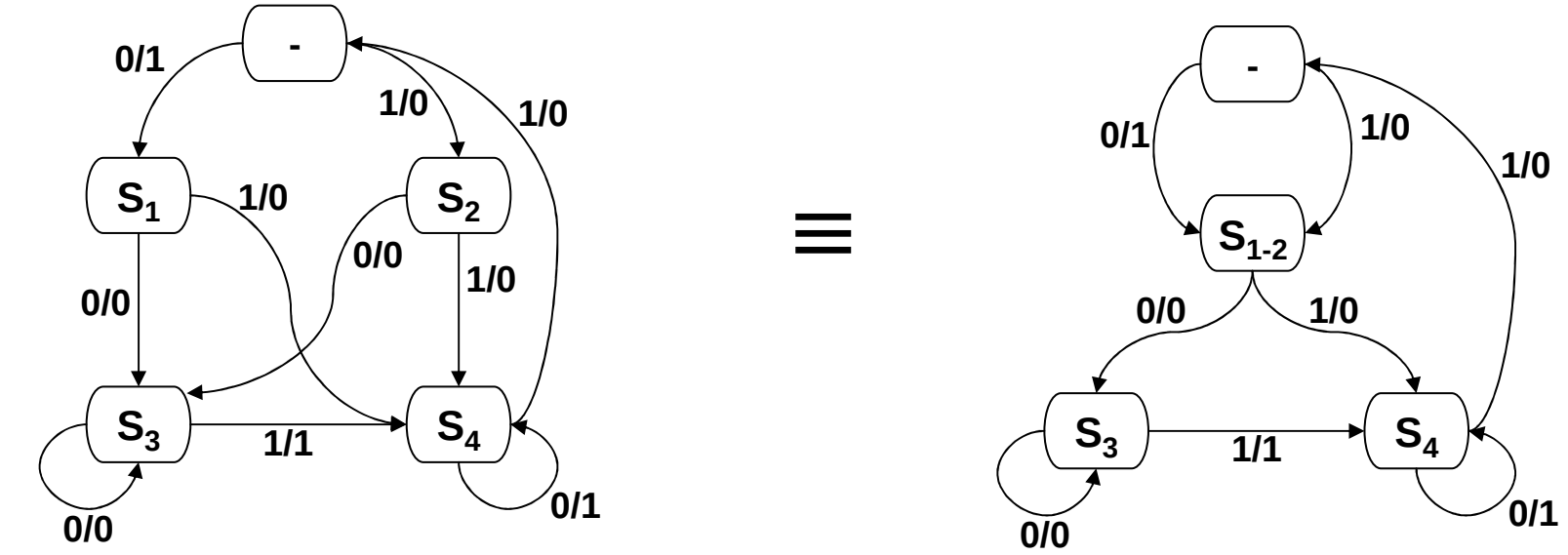
Macchina sequenziale per il controllo del semaforo relativa alla macchina di Moore

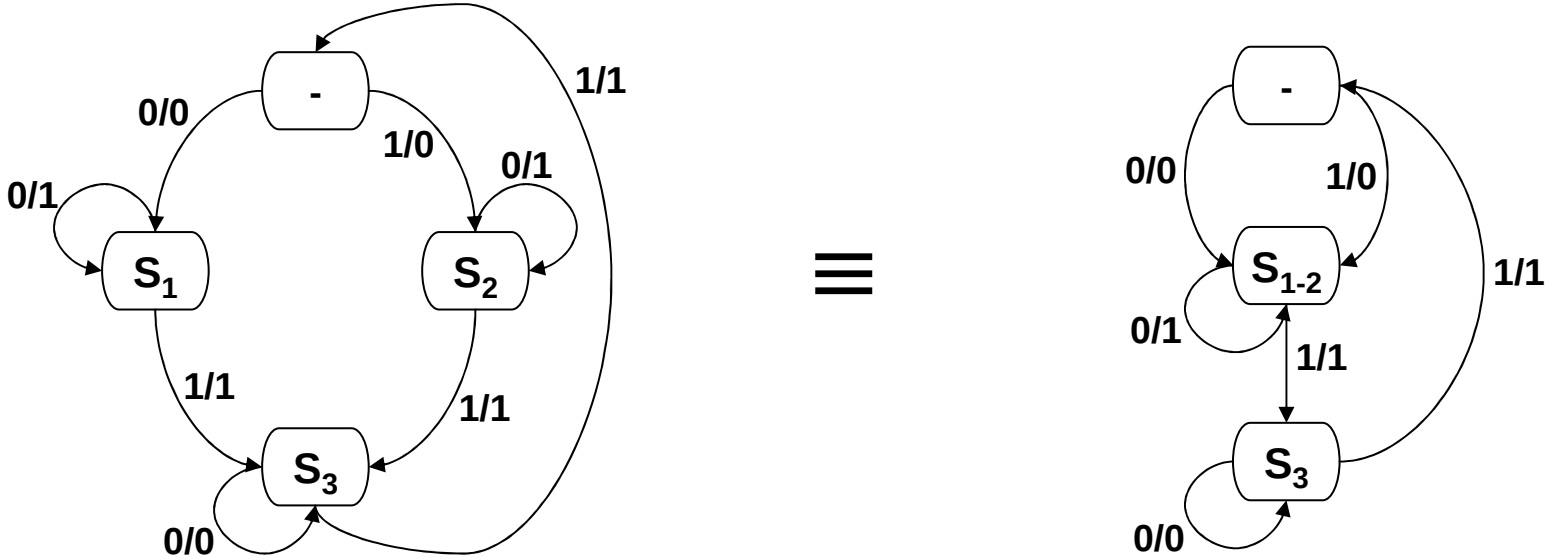


In conclusione…

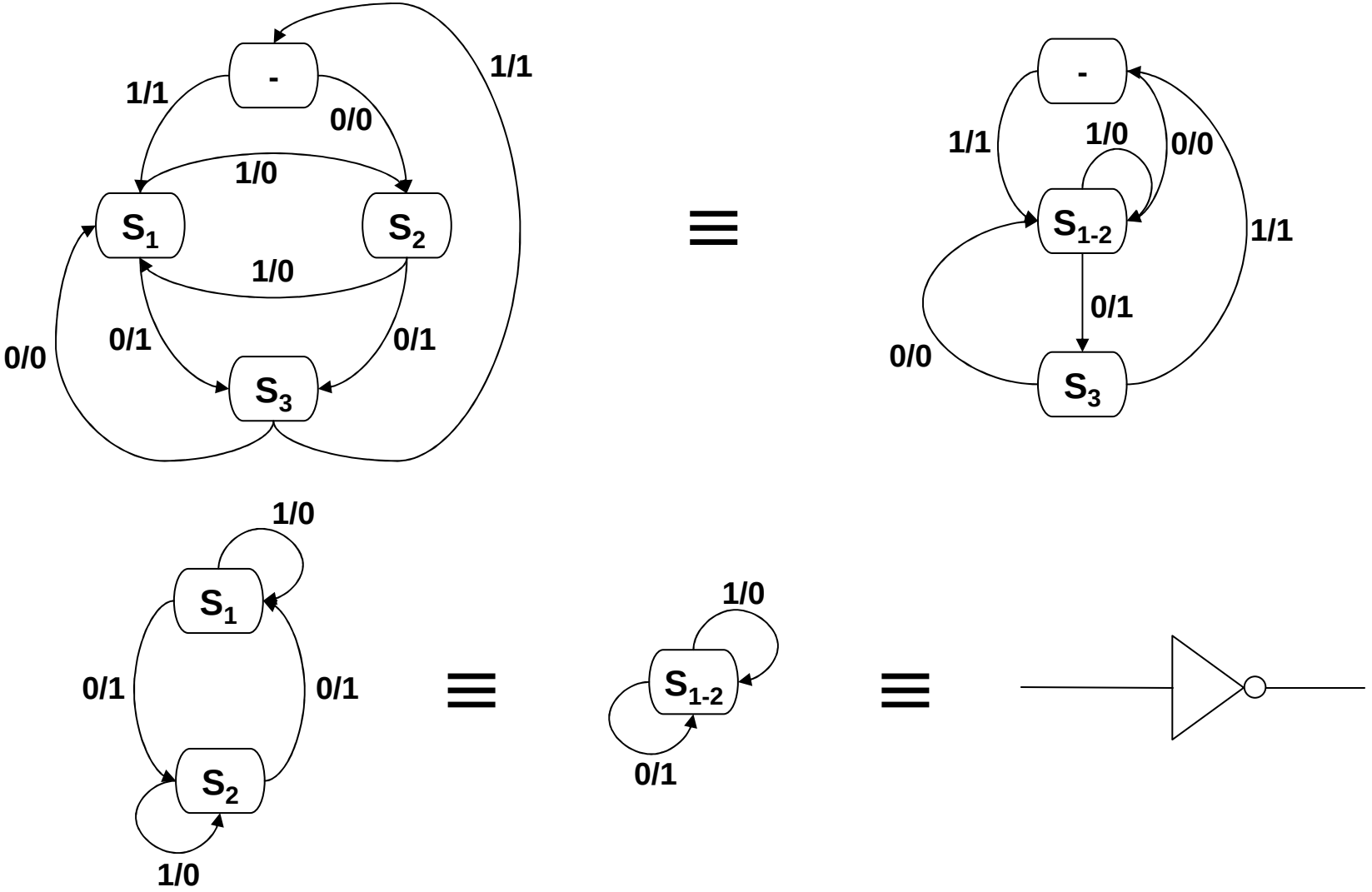
- Nella macchina di Moore le uscite dipendono solo dallo stato presente
- Occorre fare **2 tabelle di verità**:
  - una tabella rappresenta la ***funzione di stato futuro***, dove lo stato futuro è funzione degli ingressi e dello stato presente
  - l'altra tabella rappresenta ***la funzione di uscita***, dove l'uscita è funzione solo dello stato presente
- Nella macchina sequenziale la logica combinatoria può essere divisa in 2 parti: la prima determina le uscite in base allo stato presente e la seconda determina lo stato futuro in base agli ingressi e allo stato
- Vantaggio in termini di **velocità** (se le uscite devono essere disponibili il più presto possibile: si pensi al controllo del processore multi-ciclo) e in termini di **dimensioni**
- Svantaggio: il numero degli stati che può essere maggiore

Esempi di riduzione degli stati





Esempi di riduzione degli stati



Riferimenti

Computer Organization and Design

The Hardware/Software Interface 3rd Edition

David A. Patterson, John L. Hennessy

Appendice B

Versione italiana:

Struttura e Progetto dei Calcolatori

L'Interfaccia Hardware-Software

2ª edizione Zanichelli

http://en.wikipedia.org/ o http://it.wikipedia.org/