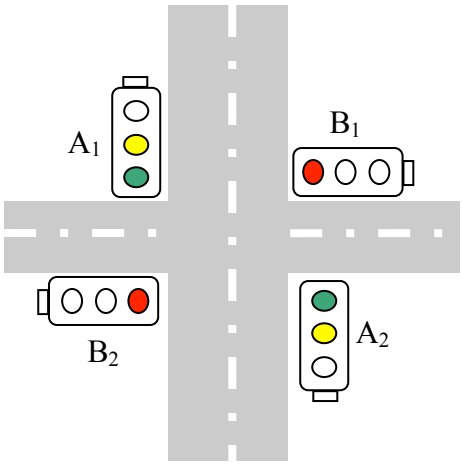
# Automi a stati finiti

## 1. Automi a stati finiti: introduzione

Supponiamo di avere un sistema che si può trovare in uno stato appartenente ad un insieme finito di stati possibili.

*Es: Immaginiamo un incrocio tra due strade regolate tramite semafori.*




*Un semaforo può trovarsi in uno dei seguenti stati (consideriamo per semplicità una sola coppia di semafori, ex. $A_1$ e $A_2$):*

$$S = \{rosso, verde , giallo, lampeggiante\}$$

Supponiamo che il nostro sistema sia dotato di ingressi che condizionano l'evoluzione del sistema stesso. Supponiamo che l'insieme dei possibili valori di ingresso sia anch'esso finito e che il sistema, a determinati istanti di tempo, cambi di stato in funzione degli ingressi e dello stato corrente.

*Es:Il semaforo dell'esempio precedente potrebbe avere in ingresso un sensore che lo informa se è giorno o notte in modo da cambiare il suo stato in lampeggiante quando è notte (si presuppone che il traffico sia così esiguo da rendere il controllo semaforico superfluo). In ingresso al sistema quindi possiamo avere due segnali possibili:*

$$I=\{giorno, notte\}$$

*Durante il giorno il semaforo segue la usuale sequenza di stati mentre durante la notte il semaforo resta lampeggiante. La transizione giorno-notte, cioè la transizione tra sequenza normale e lampeggiante, deve avvenire la prima volta che il semaforo transita per il rosso durante la notte mentre la transizione notte-giorno deve avvenire con la transizione dallo stato lampeggiante allo stato verde.*

*L'insieme delle possibili transizioni sarà una funzione tra tutti i possibili ingressi e tutti possibili stati verso l'insieme degli stati:*

```
giorno e verde → giallo
giorno e giallo → rosso
giorno e rosso → verde
giorno e lampeggiante → verde        # questa transizione avviene solo la
                                     #  prima volta che il sensore rileva
                                     #  il giorno
notte e verde →  giallo              # Quando il sensore rileva che
notte e giallo → rosso               # è giunta la notte, attende il rosso
notte e rosso →  lampeggiente        # quindi  passa a lampeggiante
notte e lampeggiante → lampeggiante  # e rimane lampeggiante fino alla
                                     # alla prossima transizione
                                     # notte-giorno
```

Supponiamo infine che il sistema, in funzione dello stato corrente, generi uno di un insieme finito di possibili segnali di uscita.

*Es: L'insieme dei possibili segnali in uscita del sistema semaforico sono i seguenti:*

$$O=\{luce\_rossa, luce\_verde, luce\_gialla, luce\_lampeggiante\}$$

*La funzione che associa gli stati ai segnali di uscita è banalmente la seguente:*

```
verde →  luce_verde
rosso →  luce_rossa
giallo →  luce_gialla
lampeggiante →  luce_lampeggiante
```

Un sistema di questo tipo è detto **Automa a Stati Finiti** (**F**inite **S**tate **A**utomata), e può essere descritto formalmente con una 5-pla:

$$\mathbf{M = < S , I , O , \delta , \lambda >}$$

dove:

- **S** è l'insieme finito degli stati possibili
- **I** è l'insieme possibile dei simboli in ingresso
- **O** è l'insieme dei possibili simboli in uscita
- **δ** è la funzione che lega ingressi e stato corrente con lo stato prossimo
- **λ** è la funzione che lega lo stato e l'ingresso correnti alle uscite

L'esempio precedente descrive un sistema in cui l'uscita dipende solo dallo stato corrente; un automa di questo tipo è detto **Automa di Moore**. Se l'uscita dipende sia dallo stato corrente che dagli ingressi, come enunciato nella definizione precedente, allora è detto **Automa di Mealy**. Nell'automa di *Moore* le uscite sono collegate allo stato corrente del sistema, nell'automa di *Mealy* sono collegate alle transizioni da uno stato ad un altro. I due automi sono equivalenti, cioè un sistema che si può realizzare con uno dei due modelli può essere sempre realizzato anche con l'altro modello. In generale, gli

automi di Mealy hanno meno stati del corrispondente automa di Moore, anche se sono più complicati da sintetizzare.

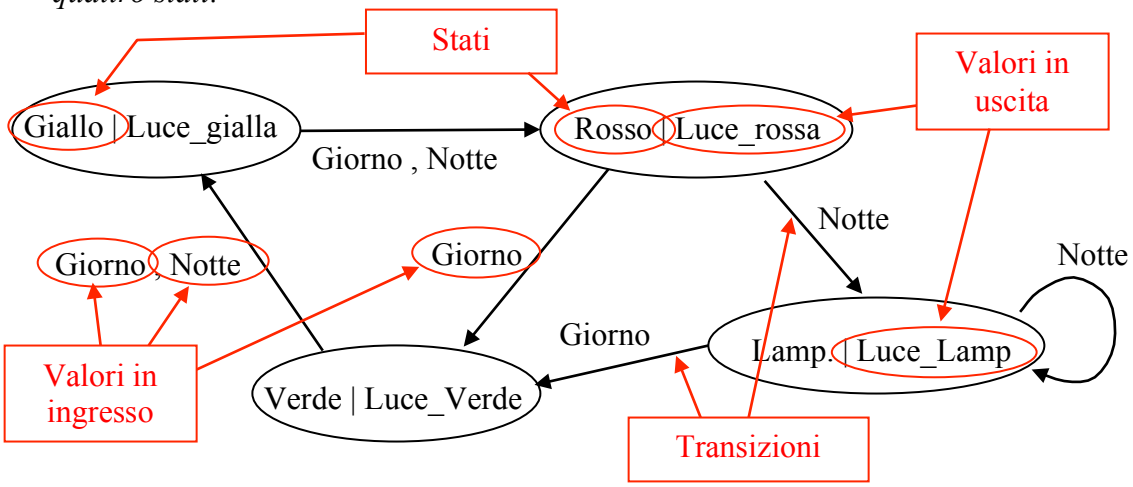
A volte può essere utile indicare lo stato iniziale $\mathbf{s_0}$ dell'automa (ad esempio lo stato che assume il sistema all'atto dell'accensione o in risposta ad un segnale di reset). In questo caso il modello teorico diventa una 6-pla:

$$\mathbf{M = < S , I , O , \delta , \lambda, s_0 >}$$

## 2. Grafo di transizione degli stati (STG -State Transition Graph)

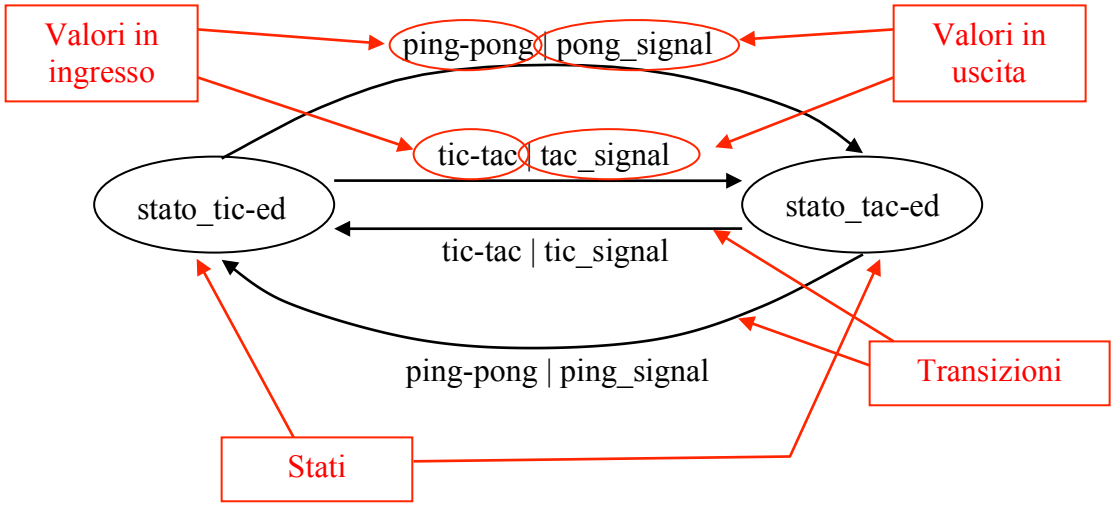
Il **Grafo di Transizione degli Stati** permette di descrivere completamente un automa a stati finiti attraverso un grafo orientato ed etichettato. Ad ogni stato corrisponde un nodo del grafo. Se è possibile una transizione da uno stato *A* ad uno stato *B* in relazione ad un ingresso *i* allora sul grafo esisterà un arco orientato dal nodo corrispondente ad *A* al nodo corrispondente a *B*, etichettato con *i.* Nel caso di automi di *Moore* in cui l'uscita è funzione del solo stato corrente, l'indicazione dell'uscita *u* corrispondente ad uno stato *A* è riportata all'interno del nodo che identifica lo stato *A*.

*Es: L'esempio precedente può essere descritto tramite un automa di Moore a quattro stati:*




Nel caso di un automa di *Mealy* in cui l'uscita è generata in corrispondenza delle transizioni da uno stato ad un altro, ogni arco (e quindi ogni transizione) viene etichettato, oltre che con l'ingresso che guida la transizione, anche con l'uscita corrispondente.

*Ex: Supponiamo di avere un metronomo che genera un segnale bistabile e supponiamo che sia possibile scegliere, attraverso un selettore, il tipo di segnale in uscita: tic-tac oppure ping-pong. Un sistema del genere è descrivibile con un automa di Mealy a due stati e un ingresso binario. L' STG corrispondente è:*




### 3. Tabella di transizione degli stati (STT - State Transition Table)

Un altro modo per descrivere completamente un automa a stati finiti è dato dalla **Tabella di Transizione degli Stati** che riporta in forma tabellare la funzione di transizione **δ** e la funzione di uscita **λ.** Nel caso di un automa di *Moore* per ogni stato corrente possibile e per ogni combinazione degli ingressi viene indicato lo stato prossimo raggiunto dall'automa. Un'ulteriore colonna della tabella esprime infine la relazione tra stato del sistema e uscita corrispondente.

Ex: STT relativa al semaforo degli esempi precedenti:

| **Ingresso** / **Stato** | **δ** giorno | notte | **λ** **Uscita** |
|---|---|---|---|
| rosso | verde | lampeggiante | luce_rossa |
| verde | giallo | giallo | luce_verde |
| giallo | rosso | rosso | luce_gialla |
| lampeggiante | verde | lampeggiante | luce_lamp. |

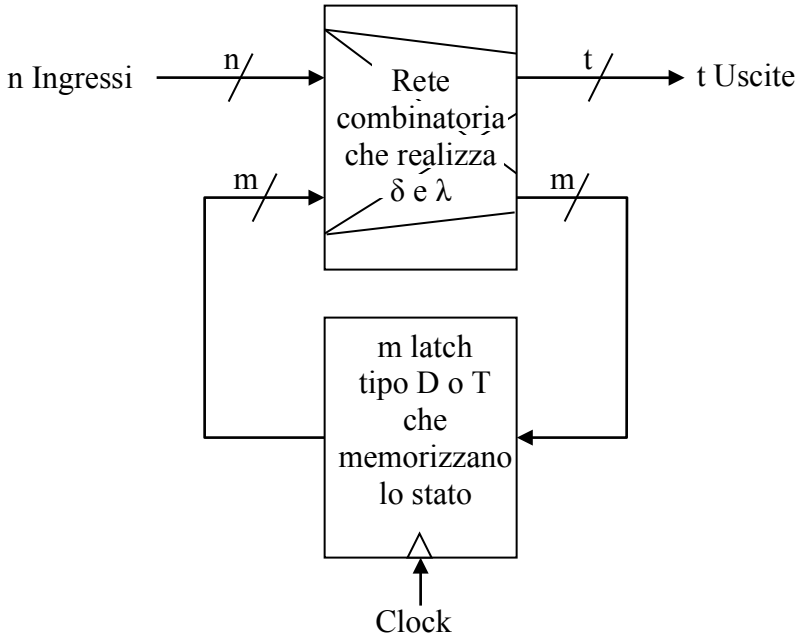
Nel caso di un automa di *Mealy* per ogni stato corrente possibile e per ogni combinazione degli ingressi viene indicato sia lo stato prossimo raggiunto dall'automa che l'uscita corrispondente.

Ex: STT relativa all'esempio del metronomo:

| **Ingresso** / **Stato** | **δ \| λ** ping-pong | tic-tac |
|---|---|---|
| stato_tic-ed | stato_tac-ed \| ping_signal | stato_tac-ed \| tac_signal |
| stato_tac-ed | stato_tic-ed \| pong_signal | stato_tic-ed \| pong_signal |

## 4. Reti sequenziali sincrone: modello di Huffman.

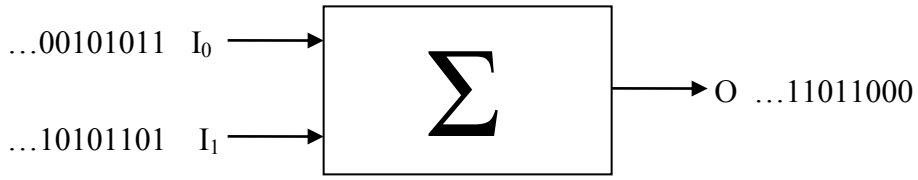
Una **rete combinatoria** è un sistema in cui uscita è univocamente determinata dallo stato degli ingressi, indipendentemente dai valori degli ingressi agli istanti precedenti (sistemi *stateless*). Se l'uscita del sistema al contrario dipende anche dall'evoluzione nel passato degli ingressi allora il sistema è detto **rete sequenziale** (sistemi *statefull*). Le reti sequenziali sono caratterizzate in ogni istante da uno stato corrente che descrive l'informazione immagazzinata dal sistema circa i segnali giunti in ingresso nel passato. L'uscita di una rete sequenziale così come lo stato prossimo della rete è una funzione dello stato e degli ingressi attuali. Se la rete evolve da uno stato all'altro in risposta ad segnale di sincronismo, o clock, allora è detta **rete sequenziale sincrona**. Il **modello di Huffman** descrive una rete sequenziale sincrona che può essere utilizzata per costruire fisicamente automi a stati finiti di Moore e Mealy.




*Modello di Huffman*

## 5. Sintesi di un automa di Moore: Sommatore binario sequenziale infinito.

Si vuole costruire una rete sequenziale che dati in ingresso due sequenze di bit, dal bit meno significato ai bit più significativi realizzi la somma binaria con l'usuale algoritmo aritmetico della somma bit a bit con riporto.



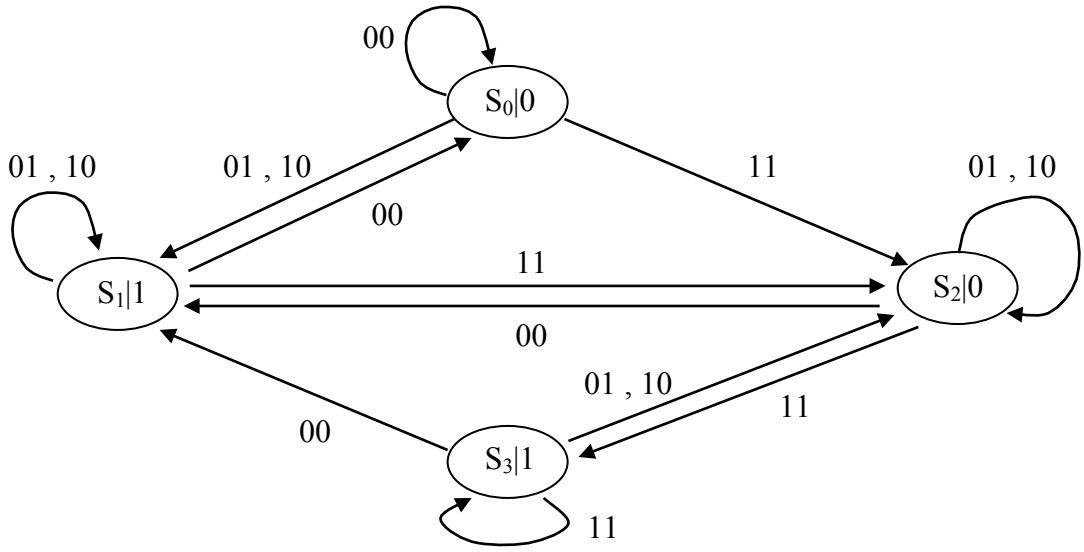

Il sistema in ogni passo di calcolo deve ricordare il risultato della somma dei due bit in ingresso e l'eventuale riporto che andrà propagato ai due bit seguenti. Il numero di stati del sistema è quindi pari a quattro:

| **Stato** | **Descrizione** |
|---|---|
| $S_0$ | S= 0 , R=0 |
| $S_1$ | S= 1 , R=0 |
| $S_2$ | S= 0 , R=1 |
| $S_3$ | S= 1 , R=1 |

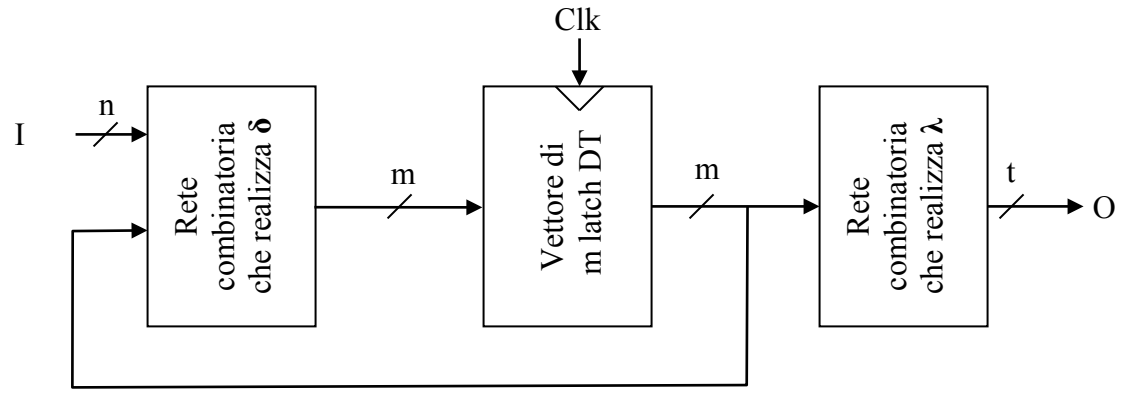
L'ingresso del sistema è costituito dai due bit da sommare. Le configurazioni in ingresso possibili (corrispondenti all'alfabeto di ingresso) sono quindi quattro: **00**, **01**, **10**, **11**. L'uscita è costituita da un solo bit che può assumere sono due configurazioni, **0** oppure **1**, corrispondente alla somma dei due bit in ingresso più l'eventuale riporto precedente.
Definiamo l'automa di Moore corrispondente attraverso la STT:

| | δ | | | | λ |
|---|---|---|---|---|---|
| **Stato** | $I_0I_1$=00 | $I_0I_1$=01 | $I_0I_1$=10 | $I_0I_1$=11 | **O** |
| $S_0$ | $S_0$ | $S_1$ | $S_1$ | $S_2$ | 0 |
| $S_1$ | $S_0$ | $S_1$ | $S_1$ | $S_2$ | 1 |
| $S_2$ | $S_1$ | $S_2$ | $S_2$ | $S_3$ | 0 |
| $S_3$ | $S_1$ | $S_2$ | $S_2$ | $S_3$ | 1 |

L' STG del sistema è il seguente:




Un automa di Moore è fisicamente realizzabile tramite un circuito sequenziale così formato (è in pratica un modello di Huffman):

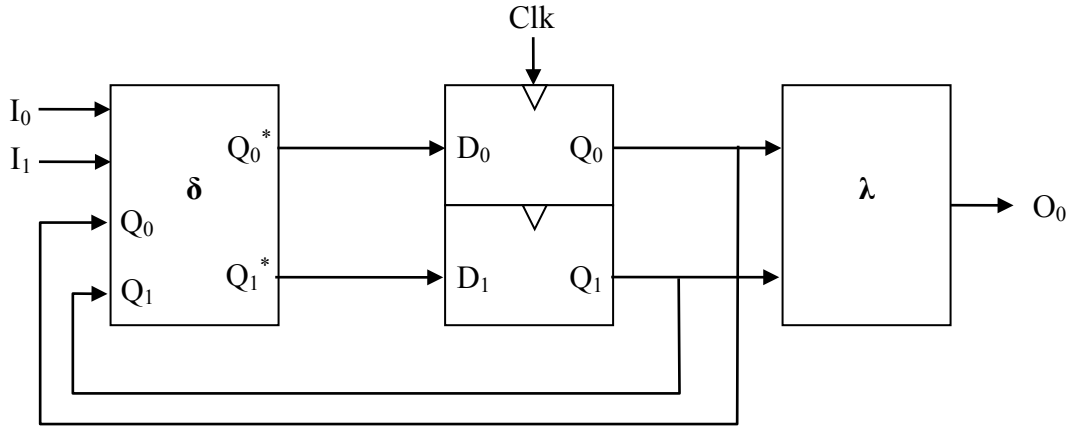



Il numero di latch occorrenti per memorizzare n stati è pari all'intero più piccolo non minore del logaritmo in base 2 di n. Poiché il sommatore richiede 4 stati i numero di latch da utilizzare sarà[1]:

$$\textbf{n.latch} = \textbf{ceil}( \log_2 4 ) = \textbf{2}.$$

[1] ceil(x) = l'intero più piccolo non inferiore a x. Es. ceil(3.1)=4 , ceil(4)=4.

Il circuito che realizza il sommatore dell'esempio è quindi:




Per sintetizzare le funzioni stato prossimo e di uscita occorre definire una corrispondenza tra gli stati del sistema e le configurazioni possibili dei latch cosi come occorre dare una codifica binaria per i simboli di ingresso ed uscita (nel caso del sommatore la mappatura degli ingressi e delle uscite è definibile naturalmente considerando i numeri binari in ingresso ed uscita stessi). La mappatura può essere fatta in qualsiasi modo tenendo però presente che influenza la complessità finale delle due funzioni δ ed λ. *Buone* mappature generano, il più delle volte, funzioni semplici mentre *cattive* mappature generano funzioni più complesse. Non esiste una regola generale per definire questa mappatura.
Nel caso del sommatore conviene assegnare le configurazioni di bit in modo che $Q_0$ indichi la somma dei bit e $Q_1$ indichi l'eventuale riporto da propagare alla transizione successiva:

| **Stato** | **Descrizione** | **$Q_0Q_1$** |
|---|---|---|
| $S_0$ | S= 0 , R=0 | 00 |
| $S_1$ | S= 1 , R=0 | 10 |
| $S_2$ | S= 0 , R=1 | 01 |
| $S_3$ | S= 1 , R=1 | 11 |

Ora è possibile trascrivere la STT sostituendo alle etichette degli stati la corrispondente configurazione dei latch:

| | **$\delta = Q_0^* Q_1^*$** | | | | **λ** |
|---|---|---|---|---|---|
| **$Q_0Q_1$** | $I_0I_1$=00 | $I_0I_1$=01 | $I_0I_1$=10 | $I_0I_1$=11 | **O** |
| 00 | 00 | 10 | 10 | 01 | 0 |
| 10 | 00 | 10 | 10 | 01 | 1 |
| 01 | 10 | 01 | 01 | 11 | 0 |
| 11 | 10 | 01 | 01 | 11 | 1 |

La sintesi delle funzioni **δ** e **λ** può essere fatta in vari modi. Nei casi più semplici si può tentare di dedurre direttamente dalla tabella la formula che descrive la funzione. Negli altri casi occorre usare metodi più sistematici come le forme SOP.

Dalla STT si deduce immediatamente che l'uscita per ogni stato del sistema è banalmente uguale al valore di $Q_0$:

$$\lambda(Q_0Q_1) = Q_0$$

Questa semplificazione deriva da come è stata scelta la mappatura tra stati e configurazioni della memoria.

Sempre dalla STT possiamo sintetizzare le formule stato prossimo per $Q_0^*$ e $Q_1^*$. Riscrivo la tabella di verità per le due variabili:

| $I_0I_1Q_0Q_1$ | $Q_0^*$ | $Q_1^*$ |
|---|---|---|
| 0000 | 0 | 0 |
| 0001 | 1 | 0 |
| 0010 | 0 | 0 |
| 0011 | 1 | 0 |
| 0100 | 1 | 0 |
| 0101 | 0 | 1 |
| 0110 | 1 | 0 |
| 0111 | 0 | 1 |
| 1000 | 1 | 0 |
| 1001 | 0 | 1 |
| 1010 | 1 | 0 |
| 1011 | 0 | 1 |
| 1100 | 0 | 1 |
| 1101 | 1 | 1 |
| 1110 | 0 | 1 |
| 1111 | 1 | 1 |

Esprimo la forma SOP per $\mathbf{Q_0^*}$ e $\mathbf{Q_1^*}$ (per comodità uso ABCD al posto di $I_0I_1Q_0Q_1$):

$Q_0^*$ = ~A~B~CD + ~A~BCD + ~AB~C~D + ~ABC~D + A~B~C~D + A~BC~D
+ AB~CD + ABCD

Riordino ed applico la regola (7° e 2b): **xy+x~y = x**

= (~ABC~D + ~AB~C~D) + (A~BC~D + A~B~C~D )
+ (AB~CD + ABCD) + (~A~BCD + ~A~B~CD )

= ~AB~D + A~B~D + ABD + ~A~BD = (~AB + A~B) ~D + (AB + ~A~B) D

Ricordando che **x~y + ~xy = x ⊕ y** e che **xy + ~x~y = ~ ( x ⊕ y )** segue:

= (A ⊕ B) ~D + ~(A ⊕ B) D = (A ⊕ B) ⊕ D = ( $I_0$ ⊕ $I_1$ ) ⊕ $Q_1$

Analogamente:

$Q_1^* = \sim AB\sim CD + \sim ABCD + A\sim B\sim CD + A\sim BCD$
$+ AB\sim C\sim D + AB\sim CD + ABC\sim D + ABCD$

Gli ultimi quattro termini sono ottenuti da AB moltiplicandolo per tutte le possibili configurazioni di CD. ne segue che possono essere semplificati (con 7a e 2b) in AB:

$= ( \sim AB\sim CD + \sim ABCD ) + ( A\sim B\sim CD + A\sim BCD ) + AB$

$= ( \sim ABD + A\sim BD ) + AB = (\sim AB + A\sim B)\, D + AB = ( A \oplus B )\, D + AB$

$= ( I_0 \oplus I_1 )\, Q_1 + I_0\, I_1$

Le formule derivate per $Q_0^*$ e $Q_1^*$ non sono altro che le formule della somma e del riporto di un Full-Adder. Ciò deriva dalla scelta fatta per la mappatura degli stati nelle configurazioni dei latch. Una altra mappatura avrebbe generato altre formule (probabilmente più complesse).

Il curcuito logico finale che realizza l'automa a stati finiti del sommatore binario sequenziale è: