

Reti sequenziali

Alessandro Pellegrini
a.pellegrini@ing.uniroma2.it

Limiti dei circuiti di commutazione

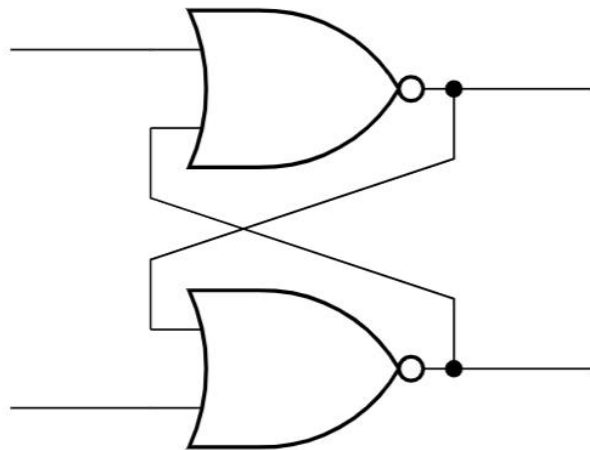
- I circuiti visti fino ad ora possono essere definiti *combinatori*
- Dato un input $\mathbf{x} = \langle x_1, \dots, x_n \rangle$, essi calcolano un output $\mathbf{y} = \langle y_1, \dots, y_n \rangle$ secondo la relazione:

$$\mathbf{y} = f(\mathbf{x})$$

- Tali circuiti quindi hanno un'uscita che dipende *esclusivamente* dall'input
- Se avessimo a disposizione solo questo tipo di circuito, non potremmo implementare *elementi di memoria*

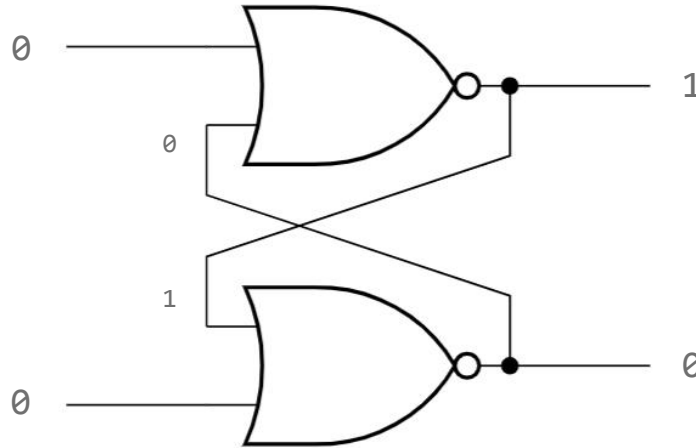
Circuito Latch

- Un circuito latch (*lucchetto*) “cattura” il valore di input utilizzando degli anelli a *retroazione*
- È un circuito analogico che consente di immagazzinare un'informazione digitale



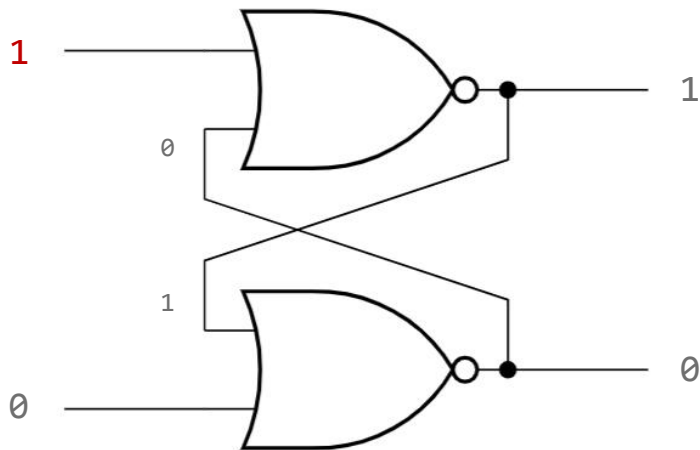
Circuito Latch

- Un circuito latch (*lucchetto*) “cattura” il valore di input utilizzando degli anelli a *retroazione*
- È un circuito analogico che consente di immagazzinare un'informazione digitale



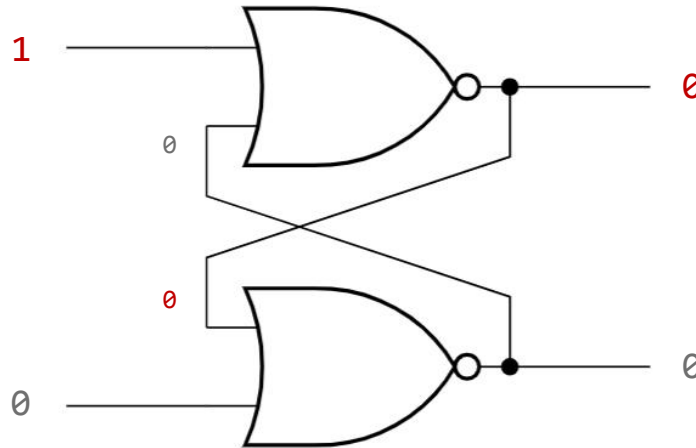
Circuito Latch

- Un circuito latch (*lucchetto*) “cattura” il valore di input utilizzando degli anelli a *retroazione*
- È un circuito analogico che consente di immagazzinare un'informazione digitale



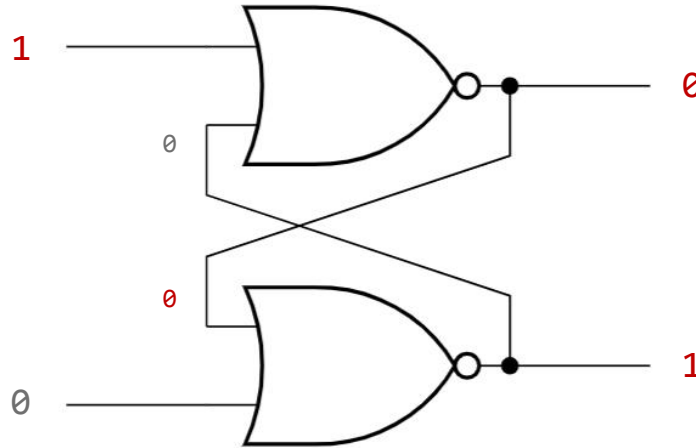
Circuito Latch

- Un circuito latch (*lucchetto*) “cattura” il valore di input utilizzando degli anelli a *retroazione*
- È un circuito analogico che consente di immagazzinare un'informazione digitale



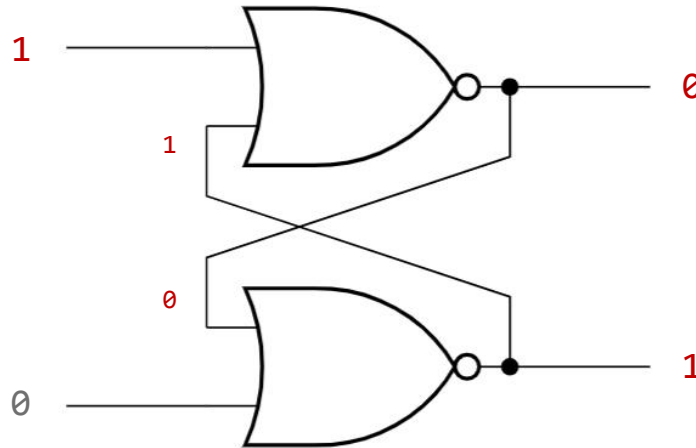
Circuito Latch

- Un circuito latch (*lucchetto*) “cattura” il valore di input utilizzando degli anelli a *retroazione*
- È un circuito analogico che consente di immagazzinare un'informazione digitale



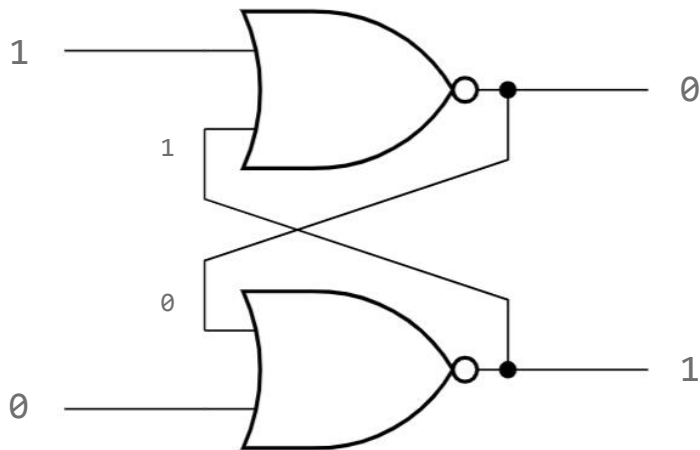
Circuito Latch

- Un circuito latch (*lucchetto*) “cattura” il valore di input utilizzando degli anelli a *retroazione*
- È un circuito analogico che consente di immagazzinare un'informazione digitale



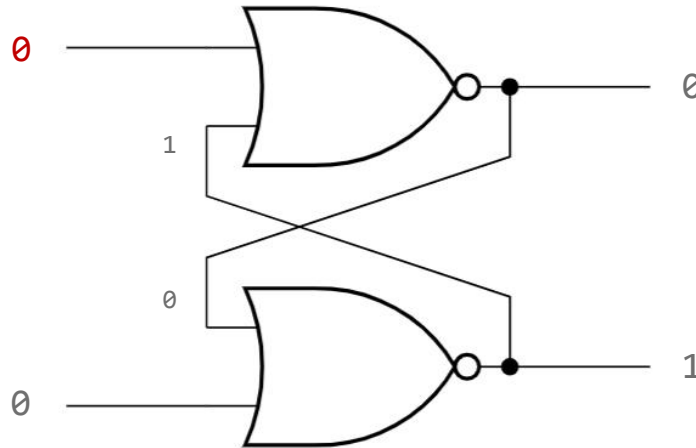
Circuito Latch

- Un circuito latch (*lucchetto*) “cattura” il valore di input utilizzando degli anelli a *retroazione*
- È un circuito analogico che consente di immagazzinare un'informazione digitale



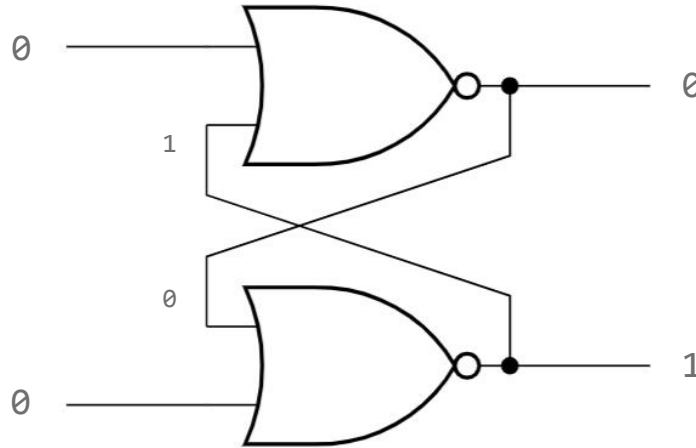
Circuito Latch

- Un circuito latch (*lucchetto*) “cattura” il valore di input utilizzando degli anelli a *retroazione*
- È un circuito analogico che consente di immagazzinare un'informazione digitale



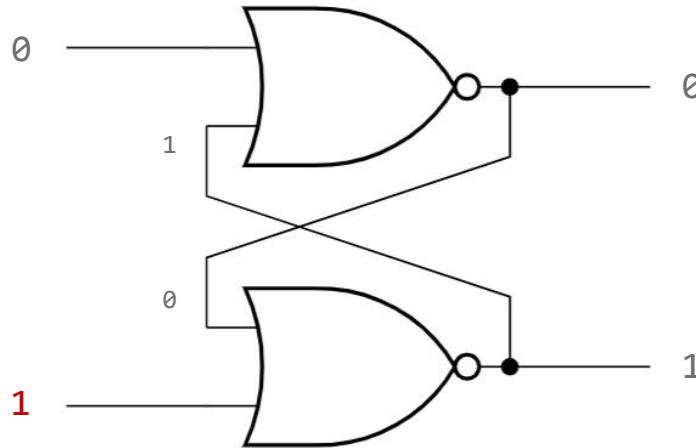
Circuito Latch

- Un circuito latch (*lucchetto*) “cattura” il valore di input utilizzando degli anelli a *retroazione*
- È un circuito analogico che consente di immagazzinare un'informazione digitale



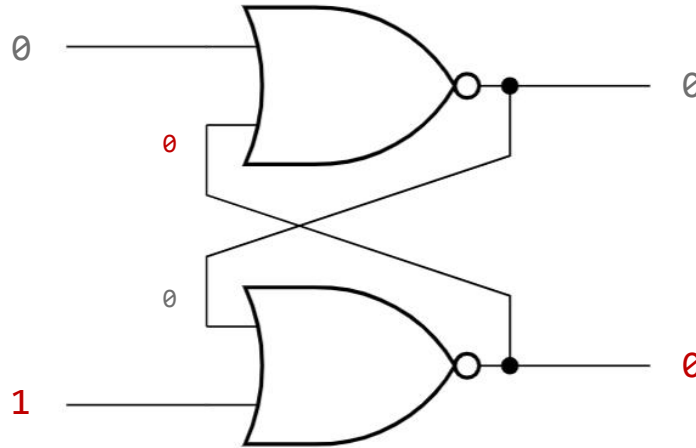
Circuito Latch

- Un circuito latch (*lucchetto*) “cattura” il valore di input utilizzando degli anelli a *retroazione*
- È un circuito analogico che consente di immagazzinare un'informazione digitale



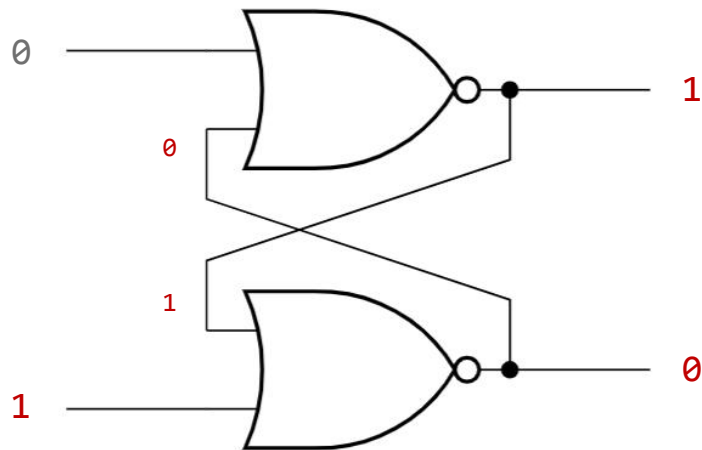
Circuito Latch

- Un circuito latch (*lucchetto*) “cattura” il valore di input utilizzando degli anelli a *retroazione*
- È un circuito analogico che consente di immagazzinare un'informazione digitale



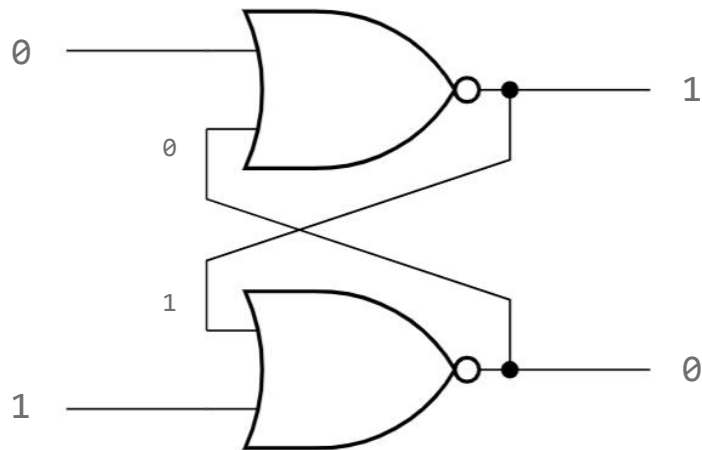
Circuito Latch

- Un circuito latch (*lucchetto*) “cattura” il valore di input utilizzando degli anelli a *retroazione*
- È un circuito analogico che consente di immagazzinare un'informazione digitale



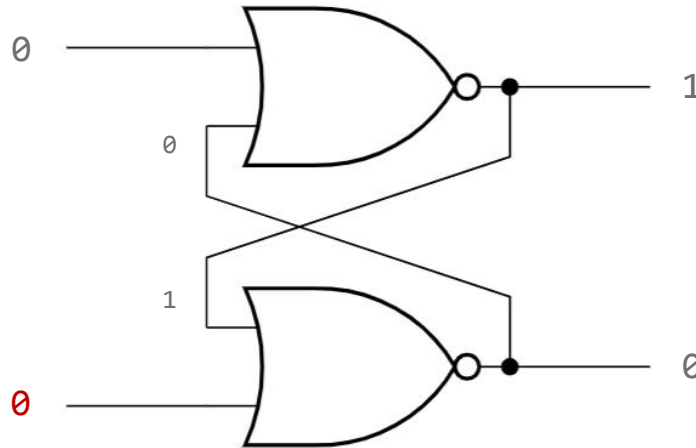
Circuito Latch

- Un circuito latch (*lucchetto*) “cattura” il valore di input utilizzando degli anelli a *retroazione*
- È un circuito analogico che consente di immagazzinare un'informazione digitale



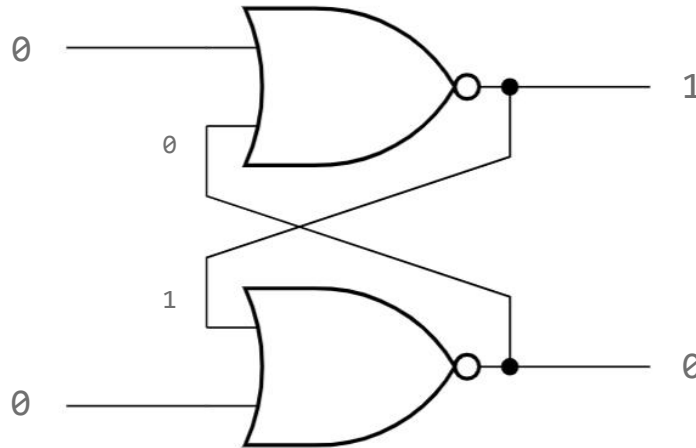
Circuito Latch

- Un circuito latch (*lucchetto*) “cattura” il valore di input utilizzando degli anelli a *retroazione*
- È un circuito analogico che consente di immagazzinare un'informazione digitale



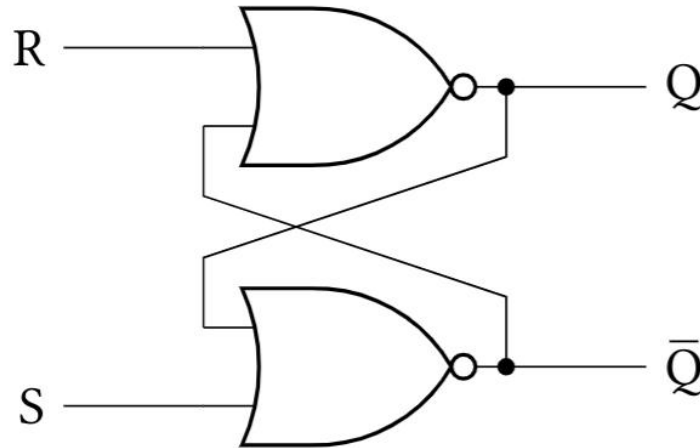
Circuito Latch

- Un circuito latch (*lucchetto*) “cattura” il valore di input utilizzando degli anelli a *retroazione*
- È un circuito analogico che consente di immagazzinare un'informazione digitale



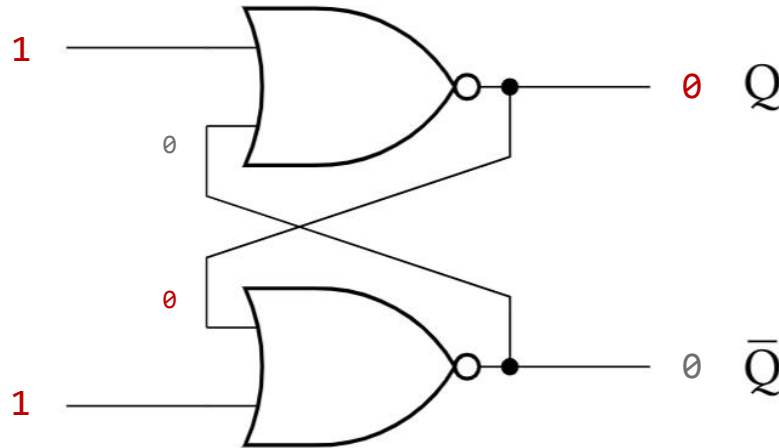
Circuito Latch

- Un circuito latch (*lucchetto*) “cattura” il valore di input utilizzando degli anelli a *retroazione*
- È un circuito analogico che consente di immagazzinare un'informazione digitale



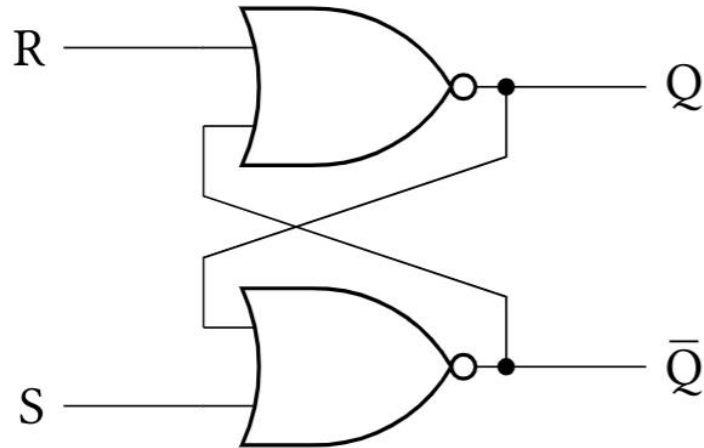
Circuito Latch

- Un circuito latch (*lucchetto*) “cattura” il valore di input utilizzando degli anelli a *retroazione*
- È un circuito analogico che consente di immagazzinare un'informazione digitale



Circuito Latch

- Un circuito latch (*lucchetto*) “cattura” il valore di input utilizzando degli anelli a *retroazione*
- È un circuito analogico che consente di immagazzinare un'informazione digitale



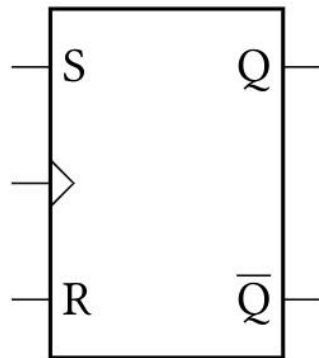
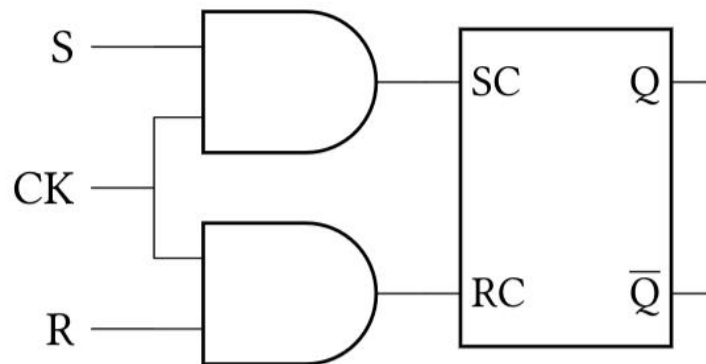
S	R	Q	Q'
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	indeterminato
1	1	1	indeterminato

Problema del Latch

- Una configurazione di ingresso in cui $S = 1$ e $R = 1$ manda il circuito in uno stato indeterminato
- I valori di S ed R possono essere calcolati da altre reti combinatorie
- Non è possibile garantire che, a causa dei ritardi di propagazione, non si osservi mai una configurazione transiente pari a $S = 1$ e $R = 1$
- Soluzione: *campionare* il valore di S ed R quando siamo certi che gli input si sono stabilizzati
- In questo caso il circuito assume il nome di *flip flop*

Flip Flop SR

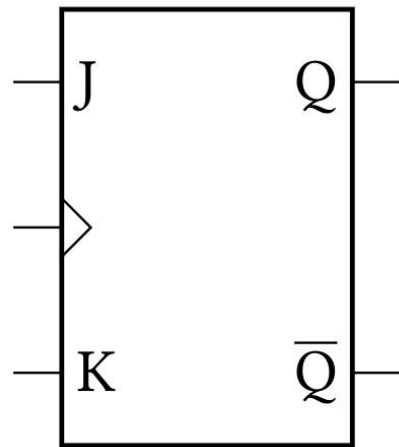
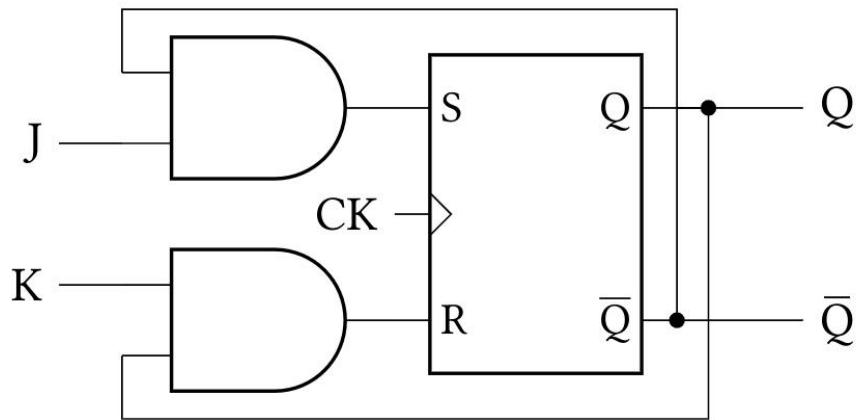
- Si basa sull'aggiunta di un segnale di *clock* al latch



- È ancora possibile che gli input vengano impostati a $S = 1$ e $R = 1$
- Possiamo costruire un circuito che *prevenga* l'insorgere di configurazioni oscillanti

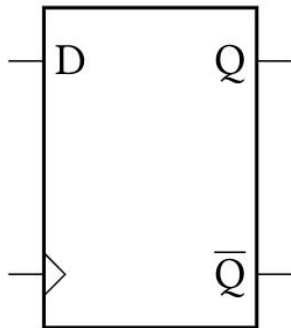
Flip Flop JK

- Il flip flop JK (da Jack Kilby, il nome dell'inventore) aggiunge una rete di controllo ai segnali in ingresso
- Tale rete rende *impossibile* che si verifichi la condizione $S = 1$ e $R = 1$ in input al flip flop SR interno



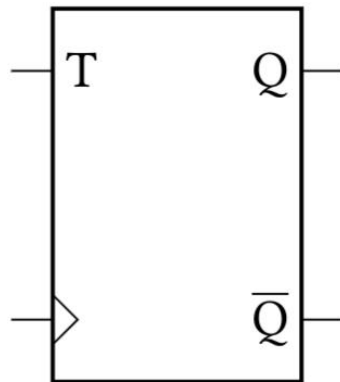
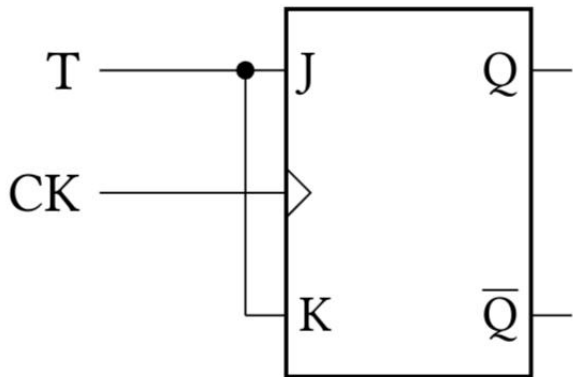
Flip Flop D

- I flip flop visti fino ad ora hanno la necessità di due segnali di controllo *opposti*
- Spesso, si vuole utilizzare un flip flop per immagazzinare un bit generato da una funzione di commutazione
- Per non dover negare esplicitamente il bit, si può usare un flip flop D
- Tale circuito si comporta da *ritardo* (delay)
 - L'input viene propagato in output dopo un periodo di clock



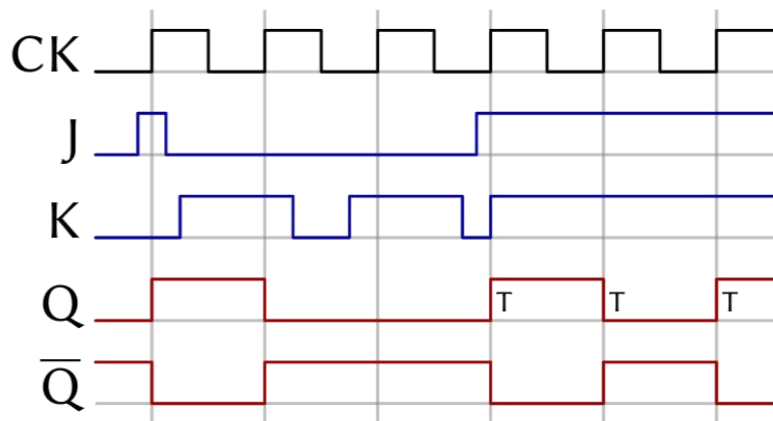
Flip Flop T

- A volte può essere utile avere a disposizione un flip flop che si comporta come switch
 - Al primo segnale, commuta da 0 a 1
 - Al secondo segnale, commuta da 1 a 0
- Il flip flop T (toggle) implementa questo comportamento



Fronti di commutazione

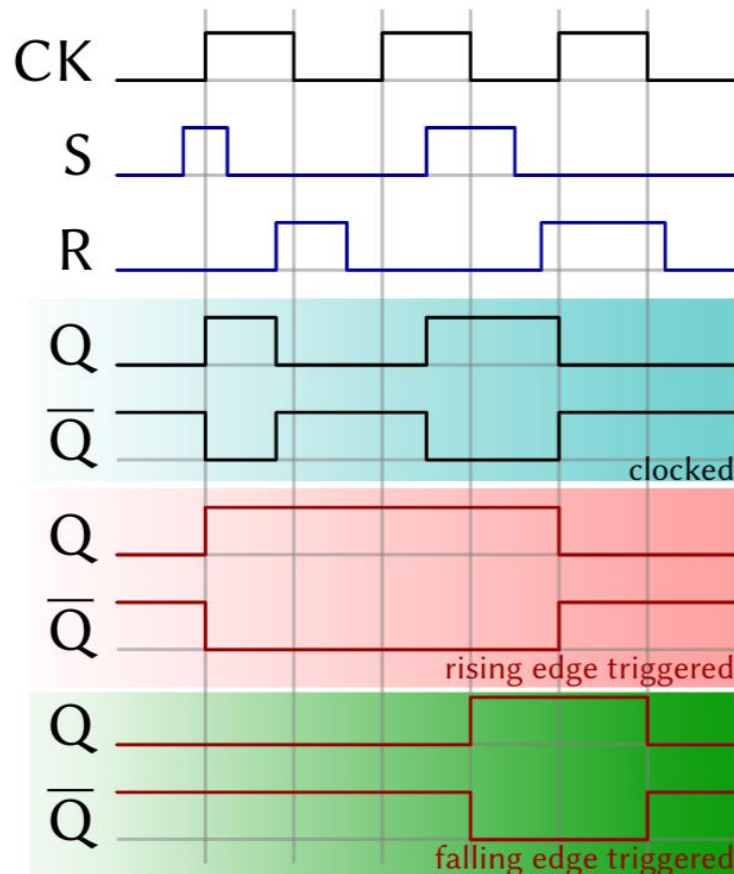
- Per funzionare correttamente, questi flip flop devono avere i segnali di controllo stabili per tutta la durata del periodo di clock
- È utile prevedere circuiti che effettuino la commutazione in finestre temporali precise e di durata più breve
- *Edge-triggered* flip flop: commutano sul fronte di salita o discesa



Rising Edge-Triggered JK Flip-Flop.

Fronti di commutazione

- Il comportamento dello stesso flip flop può essere estremamente differente
- In alcuni casi, alcuni segnali potrebbero essere ignorati
- Nella figura, sono riportati i differenti comportamenti per un flip flop SR



Reti sequenziali

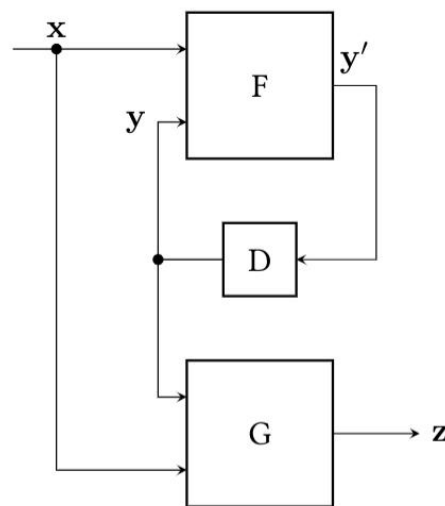
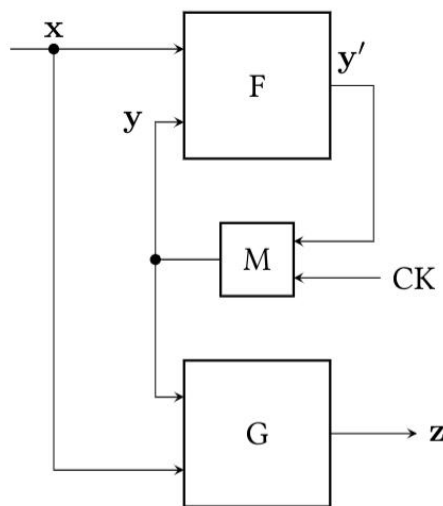
- I circuiti che implementano i flip flop sono semplici *reti sequenziali*
- Una rete sequenziale ha un funzionamento che *evolve nel tempo*
- È quindi necessario definire uno *stato interno* per descrivere l'evoluzione nel tempo

$$\begin{cases} \mathbf{y}' = f(\mathbf{x}, \mathbf{y}) \\ \mathbf{z} = g(\mathbf{x}, \mathbf{y}) \end{cases}$$

<i>S</i>	<i>R</i>	<i>Q</i>	<i>Q'</i>
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	indeterminato
1	1	1	indeterminato

Reti sequenziali

- Esistono due classi principali di reti sequenziali:
 - *sincrone*: se la transizione di stato avviene in istanti temporali controllabili dall'esterno
 - *asincrone*: se le transizioni di stato non sono controllate esternamente
- Ci concentreremo esclusivamente sulle prime



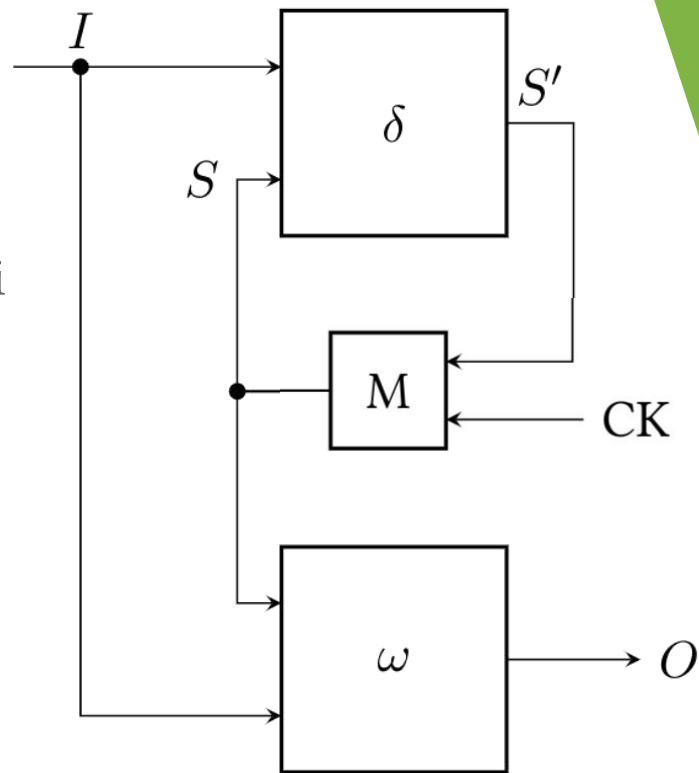
Macchine a stati finiti

- Una macchina a stati finiti è un modello matematico per la descrizione della computazione
- È una macchina astratta:
 - in un dato istante si trova in uno stato (tra un insieme finito di stati)
 - eventi esterni provocano una transizione da uno stato a un altro
 - la macchina può generare output verso l'esterno
- Esistono due varianti principali:
 - macchina di Mealy: l'output dipende dallo stato e dalla transizione innescata
 - macchina di Moore: l'output dipende unicamente dallo stato

Macchina di Mealy

$$\mathcal{M} = \langle I, S, s_0, O, \delta, \omega \rangle$$

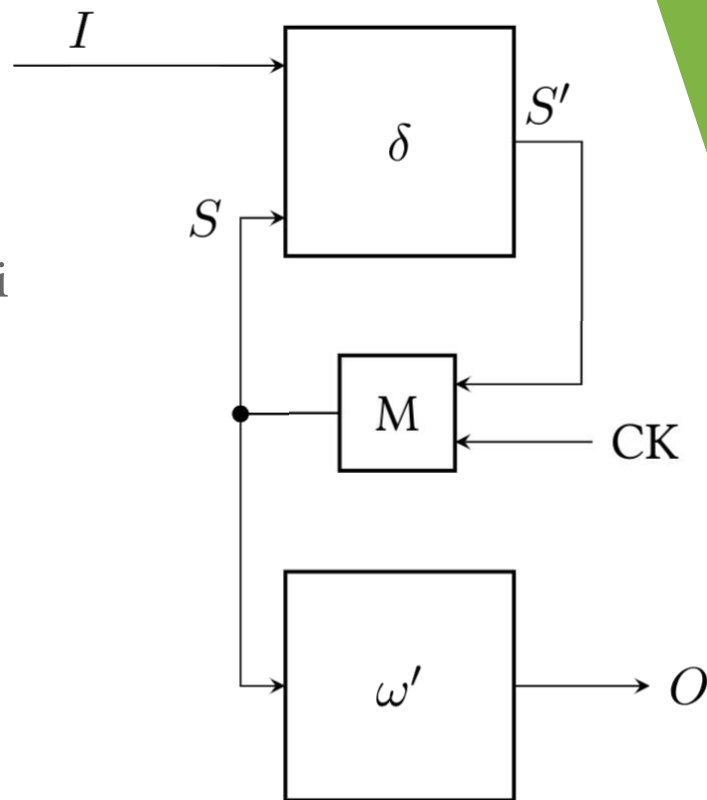
- $I = \{i_1, i_2, \dots, i_p\}$: alfabeto di ingresso
- $S = \{s_0, s_1, \dots, s_n\}$: insieme degli stati interni
- $s_0 \in S$: stato iniziale
- $O = \{o_1, o_2, \dots, o_r\}$: alfabeto di uscita
- $\delta: I \times S \mapsto S$: funzione di transizione
- $\omega: I \times S \mapsto O$: funzione che calcola l'output



Macchina di Moore

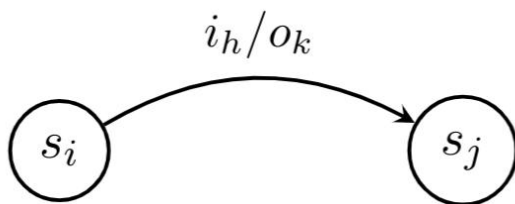
$$\mathcal{M} = \langle I, S, s_0, O, \delta, \omega \rangle$$

- $I = \{i_1, i_2, \dots, i_p\}$: alfabeto di ingresso
- $S = \{s_0, s_1, \dots, s_n\}$: insieme degli stati interni
- $s_0 \in S$: stato iniziale
- $O = \{o_1, o_2, \dots, o_r\}$: alfabeto di uscita
- $\delta: I \times S \mapsto S$: funzione di transizione
- $\omega': S \mapsto O$: funzione che calcola l'output

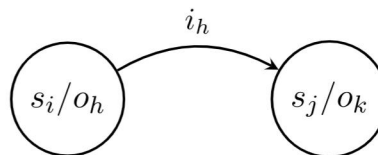


Rappresentazioni

- Per rappresentare una macchina a stati è possibile utilizzare due formalismi:
 - *Diagramma degli stati*: è un grafo che mostra graficamente le relazioni tra gli stati e le transizioni, identificando anche i caratteri di output
 - *Tabella degli stati e delle transizioni*: è una rappresentazione equivalente in forma tabellare



Modello di Mealy



Modello di Moore

Rappresentazioni

	i_1	i_2	...	i_j	...	i_p
s_1						
s_2						
\vdots						
s_i				$\delta(i_j, s_i)/\omega(i_j, s_i)$		
\vdots						
s_n						

Modello di Mealy

	i_1	i_2	...	i_j	...	i_p	ω'
s_1							
s_2							
\vdots							
s_i				$\delta(i_j, s_i)$			$\omega'(i_j, s_i)$
\vdots							
s_n							

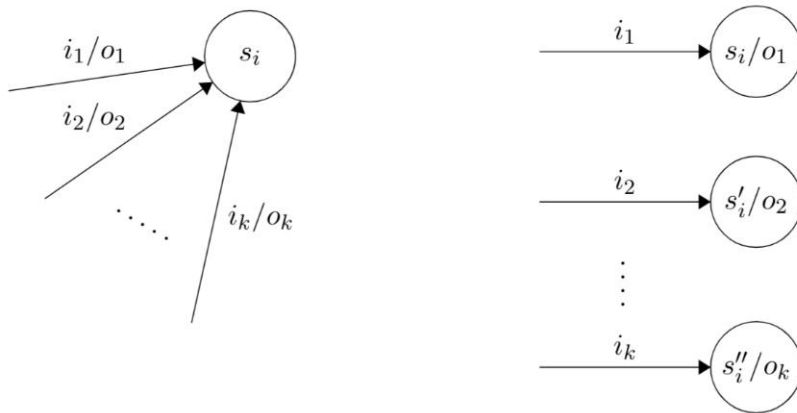
Modello di Moore

Equivalenza tra modelli

- Esiste sempre una trasformazione tra una macchina di Mealy e una macchina di Moore
 - I due modelli sono equivalenti
- Trasformazione da Moore a Mealy
 - Gli alfabeti di input ed output sono gli stessi
 - Gli stati sono gli stessi
 - Se in uno stato s_i raggiunto da una transizione causata da un carattere i_j viene generato un output o_k , quello stesso output viene generato durante la transizione i_j verso lo stato s_i

Equivalenza tra modelli

- La trasformazione da macchina di Mealy a macchina di Moore è meno immediata
- Possiamo avere più transizioni verso lo stesso stato che generano output differenti
- In questo caso, lo stato di destinazione deve essere decomposto in più stati differenti



Sintesi delle macchine

- Per realizzare circuitalmente una macchina è necessario:
 - Realizzare il blocco M: questo può essere fatto utilizzando un numero sufficiente di flip flop D
 - Realizzare la rete δ : è necessario realizzare un circuito di commutazione per aggiornare lo stato di ciascun flip flop D (*equazione di eccitazione di un flip flop*)
 - Realizzare la rete ω : è necessario realizzare un circuito di commutazione per generare ciascun bit dei caratteri di output
- Trattandosi di reti combinatorie, è possibile utilizzare le tecniche di sintesi e minimizzazione che abbiamo studiato per le funzioni booleane
- La sintesi può essere svolta a partire dalla tabella degli stati e transizioni

Esempio

- Sintesi della macchina che accetta la stringa “*mamma*” in input
- Variante di Mealy e Moore
- Con e senza stato pozzo
- Con e senza recupero dagli errori