Bistabili e Flip-Flop

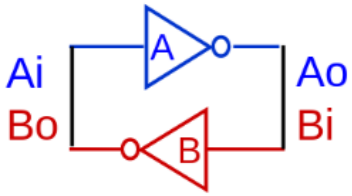
▼ Creatore originale: @LucaCaffa

Bistabile

Consideriamo un anello di inverter, in cui l'ingresso di una porta NOT (inverter A) è collegato all'uscita di un'altra porta NOT (inverter B).

Per ogni inverter, utilizzeremo una notazione del tipo:

- Xi: ingresso dell'inverter X (es. Ai, ingresso di inverter A);
- Xo: uscita dell'inverter X (es. Ao, uscita dell'inverter A).



Visualizzazione dell'anello di inverter

Nell'elemento descritto, l'entrata di un inverter è uguale all'uscita dell'altro: Ao=Bi, Ai = Bo.

Stati stabili

In questo circuito, abbiamo 2 stati stabili:

- stato 0 (S0)(S_0)(S0): se l'uscita di A è 1, allora l'uscita di B è 0;
  - Ao=H;
  - Bo=L.
- stato 1 (S1)(S_1)(S1): se l'uscita di A è 0, allora l'uscita di B è 1.
  - Ao=L;
  - Bo=H.

Dati questi due stati, si parla di circuito bistabile.

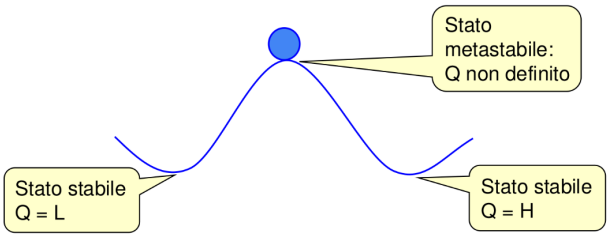
Metastabilità

Nella realtà esiste un terzo stato non stabile, chiamato stato metastabile.

Risulta poco intuitivo, ma possiamo immaginare di avere un circuito dove gli ingressi e le uscite non sono definite, e quindi non hanno un valore che possiamo determinare.

Per descrivere il concetto, possiamo utilizzare un'analogia meccanica (pallina su una collina):

- se la pallina è sul lato sinistro della collina, siamo nello stato stabile S0S_0S0;
- se la pallina è sul lato destro della collina, siamo nello stato stabile S1S_1S1;



Visualizzazione dell'analogia meccanica

- se la pallina è sulla cima della collina, siamo nello stato metastabile SxS_xSx, dove una minima influenza esterna modifica la posizione della pallina, portandola ad una delle due posizioni stabili descritte nei punti precedenti.

Lo stato metastabile è uno stato che non fornisce alcuna informazione importante, poiché non possiamo osservare alcun valore. Non vogliamo MAI trovarci in questo stato, poiché il minimo rumore può portare il circuito in uno dei due stati, portando un fattore definito dalla probabilità al circuito.

Flip-Flop

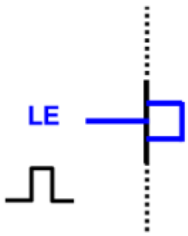
I Flip-Flop (FF) sono dei componenti bistabili, ovvero che presentano due stati stabili.

Sincronizzazione dei FF

Latch

Il comando LE\text{LE}LE è un segnale di tipo asincrono. Esso è identificato attraverso un quadrato nei circuiti.

- LE=1\text{LE} = 1LE = 1: stato di trasparenza;
- LE=0\text{LE} = 0LE = 0: stato di memoria;
- LE=(1→0)\text{LE} = (1 \rightarrow 0)LE = (1 → 0): memorizza l'ingresso.



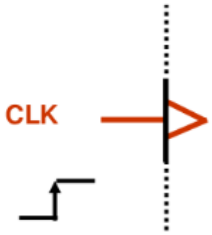
Segnale di Enable (Latch, LE)

Clock

Il comando CLK\text{CLK}CLK è un segnale di tipo sincrono. Esso è identificato attraverso un triangolo nei circuiti.

Si possono definire due tipi di clock:

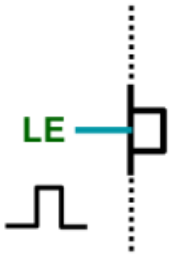
- l'uscita del componente commuta sulle transizioni del Clock 0→10 \rightarrow 10 → 1, 1→01 \rightarrow 01 → 0, e viene chiamato Ec Triggered;
- l'uscita del componente commuta solo sulla transizione del Clock 0→10 \rightarrow 10 → 1.



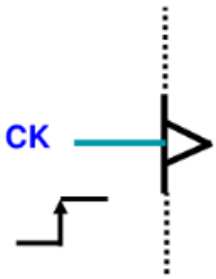
Segnale di clock (CLK)

Inverter all'ingresso

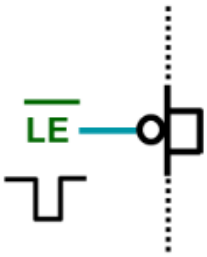
Ogni tipo di segnale, sia esso LE\text{LE}LE o CLK\text{CLK}CLK, può avere un'inverter all'ingresso, portando all'attivazione sul valore opposto.



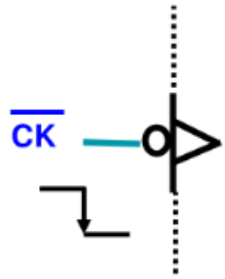
LE\text{LE}LE con valore diretto



CLK\text{CLK}CLK con valore diretto



LE\text{LE}LE con valore inverso



CLK\text{CLK}CLK con valore inverso

Flip-Flop Set-Reset

Il Flip-Flop Set-Reset (FF-SR) prende il nome dai suoi due ingressi: SET (SSS) e RESET (RRR).

Esso presenta due uscite, QaQ_aQa e QbQ_bQb, le quali sono complementari (ovvero una è il negato dell'altro). Data la complementarità delle uscite, possiamo anche chiamarle rispettivamente QQQ e Q*Q^*Q*. Questo componente ha dei ritardi di propagazione, classici delle porte NOR.

Condizioni di comando

Descriviamo i due stati stabili S0S_0S0 e S1S_1 S1(condizioni di comando) del FF-SR:

- S0(S=1,R=0)S_0 (S=1, R=0)S0(S = 1, R = 0): si chiama stato di SET, poiché S=1S=1S = 1;
- S1(S=0,R=1)S_1 (S=0, R=1)S1(S = 0, R = 1): si chiama stato di RESET, poiché R=1R=1R = 1.

Stato di memoria e stato proibito

Oltre ai due stati stabili, esistono le restanti combinazioni dei valori QaQ_aQa e QbQ_bQb, che permettono di definire altri due stati S2S_2S2 e S3S_3S3:

- S2S_2S2: si chiama stato di memoria, poiché permette di conservare un valore.

$Qa(t) = Qa(t-1) Q\_a(t) = Q\_a(t-1) Qa(t) = Qa(t-1)$

$Qb(t) = Qb(t-1) Q\_b(t) = Q\_b(t-1) Qb(t) = Qb(t-1)$

Come si nota dalle equazioni che definiscono $Qa(t) Q\_a(t) Qa(t)$ e $Qb(t) Q\_b(t) Qb(t)$, il valore delle uscite è definito da quello dell'istante precedente, definendo la condizione di memoria. Grazie a questo stato, il FF-SR è uno dei componenti fondamentali per le memorie;

- S3S_3S3: si chiama stato proibito, poiché il valore delle uscite non è definito.

  $Qa(t) = ? Q\_a(t) = \backslash ? Qa(t) = ?$

  $Qb(t) = ? Q\_b(t) = \backslash ? Qb(t) = ?$

  Come si nota dalle equazioni che definiscono $Qa(t) Q\_a(t) Qa(t)$ e $Qb(t) Q\_b(t) Qb(t)$, il valore delle uscite non è definito, definendo la condizione non permessa (o proibita). Questo stato è definito dalla metastabilità.

Implementazione con porte NOR

L'implementazione con porte NOR di un FF-SR si costruisce in maniera analoga all'anello di inverter visto prima, sostituendo gli inverter con delle porte NOR.

Gli stati in questa implementazione sono definiti come segue:

- condizione di comando SET S0S_0S0;

  $S=1, R=0 S=1, R=0 S = 1, R = 0$

- condizione di comando RESET S1S_1S1;

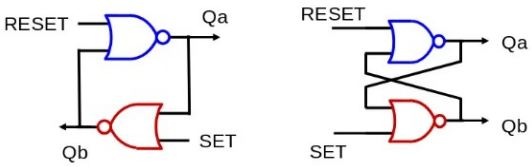
  $S=0, R=1 S=0, R=1 S = 0, R = 1$

- condizione di memoria S2S_2S2;

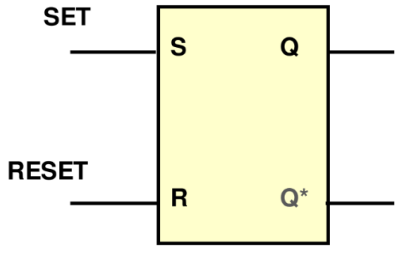
  $S=0, R=0 S=0, R=0 S = 0, R = 0$

- condizione proibita S3S_3S3.

  $S=1, R=1 S=1, R=1 S = 1, R = 1$



Implementazione di FF-SR attraverso due anelli di porte NOR



Simbolo del Flip-Flop Set-Reset con porte NOR

La tavola di verità del FF-SR implementato con le porte NOR è definita a lato, utilizzando questa legenda:

- condizioni di comando;
- condizione di memoria;
- condizione proibita.

SSS	RRR	QQQ	Q*Q^*Q*
0	0	Q−1Q_{-1}Q−1	Q−1*Q^*_{-1}Q−1*
0	1	0	1
1	0	1	0
1	1	?	?

Implementazione con porte NAND

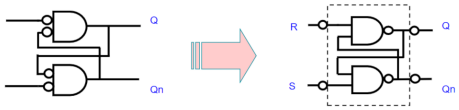
L'implementazione con porte NAND di un FF-SR si costruisce in maniera analoga all'implementazione con porte NOR, ma la differenza è nella tavola di verità, in cui tutte le condizioni sono invertite.

Per capire il motivo del comportamento identico e invertito, ricordiamo la legge di De Morgan:

$$x+y^{-}=x^{-} \cdot y^{-}\color{red} \overline {x + y}$$

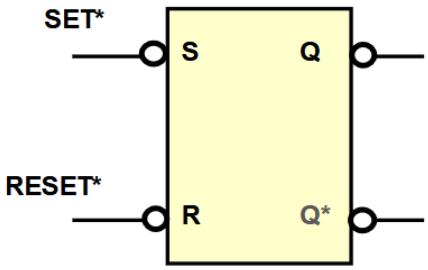
$$\color{white} =$$

$$\color{orange} \overline x \cdot \overline yx + y \quad x \cdot y$$



Implementazione di FF-SR attraverso due anelli di porte NAND

Si ha, quindi, una corrispondenza tra la porta NOR e la porta NAND, e ciò permette di realizzare l'anello di inverter con le porte NAND al posto degli inverter, come definito nella visualizzazione.



Simbolo del Flip-Flop Set-Reset con porte NAND

Gli stati in questa implementazione sono definiti come segue:

- condizione di comando SET S0S_0S0;

  S=1,R=0S=1, R=0S = 1, R = 0

- condizione di comando RESET S1S_1S1;

  S=0,R=1S=0, R=1S = 0, R = 1

- condizione di memoria S2S_2S2;

  S=1,R=1S=1, R=1S = 1, R = 1

- condizione proibita S3S_3S3.

  S=0,R=0S=0, R=0S = 0, R = 0

La tavola di verità del FF-SR implementato con le porte NAND è definita a lato, utilizzando questa legenda:

- condizioni di comando;
- condizione di memoria;
- condizione proibita.

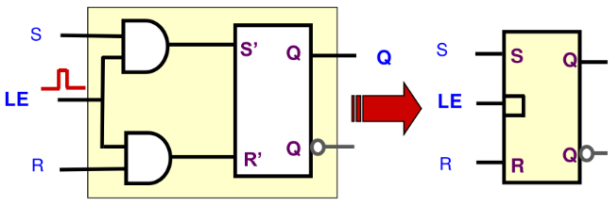
SSS	RRR	QQQ	Q*Q^*Q*
0	0	?	?
0	1	1	0
1	0	0	1
1	1	Q−1Q_{-1}Q−1	Q−1*Q^*_{-1}Q−1*

Si noti come le uscite delle condizioni di comando sono invertite rispetto all'implementazione con le porte NOR. Per esempio, se prima con S=1,R=0S=1, R=0 S = 1, R = 0 si avevano Q=1,Q*=0Q=1, Q^*=0Q = 1, Q* = 0, in questo caso l'output sarà opposto: Q=0,Q*=1Q=0, Q^*=1Q = 0, Q* = 1.

Flip-Flop Latch (FF Latch)

Partendo da un FF-SR e aggiungendo agli ingressi due porte AND, ricaviamo un componente chiamato FF Latch.

La caratteristica fondamentale di questo componente è quella di avere un segnale LE\text{LE}LE che attiva o disattiva gli ingressi, come segue:



Implementazione del Flip-Flop Latch a partire da un FF-SR

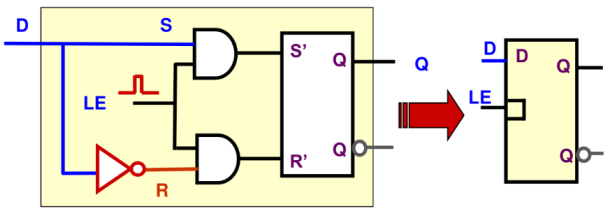
- LE=1\text{LE} = 1LE = 1 (stato trasparente): l'uscita può cambiare stato in base agli ingressi SSS e RRR;
- LE=0\text{LE} = 0LE = 0 (stato di memoria/latched mode): l'uscita non può cambiare stato.

Ciò significa che, anche se SSS e RRR assumono dei valori, il segnale entrerà nel FF solo se LE=1\text{LE}=1LE = 1, come conseguenza dell'aggiunta delle porte AND.

> □□
> Il comando LE\text{LE}LE è identificato attraverso un quadrato, che identifica un segnale di tipo asincrono.

Flip-Flop Latch D (D Latch)

Per migliorare il FF Latch, possiamo fare in modo che i due ingressi diventino un unico segnale, e che quindi SSS e RRR siano sostituiti da un unico segnale. Questo possiamo farlo perché SSS e RRR devono sempre essere uno il negato dell'altro.



Implementazione completa di un FF Latch D a partire da un FF-SR

L'unico caso in cui devono essere uguali è il comando di memoria, ma poiché qui abbiamo il segnale LE\text{LE}LE che ci permette di creare lo stato di memoria, possiamo rendere SSS e RRR sempre opposti.

Per permettere il funzionamento con un singolo ingresso, bisogna fare in modo che il ramo entrante in SSS abbia segnale diretto, mentre il ramo entrante in RRR abbia segnale negato.

Dopo questa aggiunta, si ottiene un Latch D, dove D è l'ingresso unico. Si noti come questa versione del Latch D è asincrona.

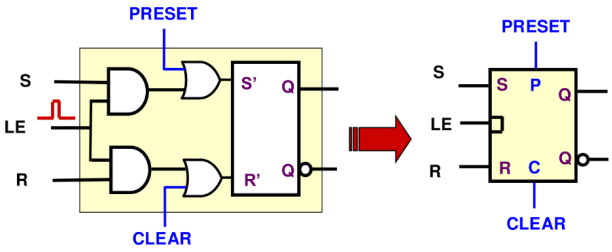
Il Latch D è di fondamentale importanza, poiché risolve il problema dello stato metastabile del FF-SR. Avendo un unico ingresso, in un caso negato e nell'altro diretto, è impossibile ricadere nella condizione di metastabilità.

Comandi asincroni per il D Latch

Possiamo aggiungere al Latch D appena visto due comandi, che non dipendono dal segnale LE\text{LE}LE, e per questo sono definiti comandi asincroni del Latch D.

I comandi sono PRESET\text{PRESET}PRESET (P) e CLEAR\text{CLEAR}CLEAR (C):

- se CLEAR=1\text{CLEAR} = 1CLEAR = 1, si imposta Q=0Q = 0Q = 0, ovvero l'uscita QQQ va immediatamente a 0;
- se PRESET=1\text{PRESET} = 1PRESET = 1, si imposta Q=1Q=1Q = 1, ovvero l'uscita QQQ va immediatamente a 1.



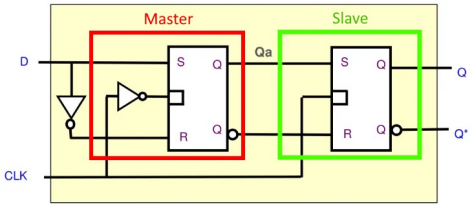
Costruzione Latch D con i comandi asincroni Clear (C) e Preset (P)

Se PRESET\text{PRESET}PRESET e CLEAR\text{CLEAR}CLEAR si attivano contemporaneamente, il FF si porta nella condizione proibita.

Flip-Flop Master-Slave (D-FF)

Il FF Master-Slave (D-FF) è un componente formato da una cascata di FF Latch con abilitazione complementare dei dati attraverso il segnale di clock.

Come si può notare dall'implementazione mostrata, si ha un solo segnale di ingresso D. Le uscite del primo FF Latch sono le entrate del secondo FF Latch.

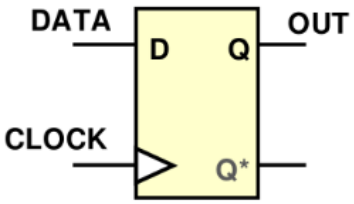


Implementazione completa di un FF Master-Slave a partire da due FF Latch

Funzionamento rispetto al Clock

In questo componente, il clock abilita uno dei due FF, attraverso il collegamento a Latch, durante la fase di transizione 0 → 1 come descritto in seguito:

- CLK=0\text{CLK} = 0CLK = 0: abilita il primo FF Latch (Master) a registrare l'ingresso D, mentre il secondo (Slave) rimane nello stato di memoria. In questo caso, si parla di Master trasparente e Slave in memoria;
- CLK=1\text{CLK} = 1CLK = 1: abilita il secondo FF Latch (Slave) ad ottenere il valore in memoria del Master, portandolo come valore in uscita al componente, mentre il secondo (Master) rimane nello stato di memoria. In questo caso, si parla di Master in memoria e Slave trasparente.



Simbolo del Flip-Flop Master-Slave

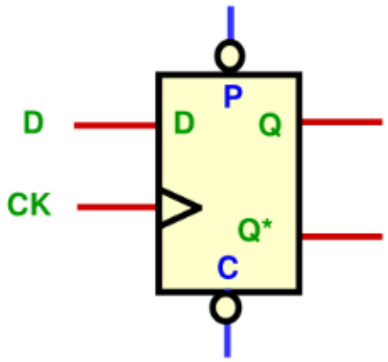
□□
Il comando CLK\text{CLK}CLK è identificato attraverso un triangolo, che identifica un segnale di tipo sincrono.

□□
E' il tipo di porta sequenziale più usato.

Comandi asincroni per il D-FF

Come per il D Latch, anche il D-FF può avere i due comandi asincroni PRESET e CLEAR.

- se CLEAR=1\text{CLEAR} = 1CLEAR = 1, si imposta Q=0Q = 0Q = 0, ovvero l'uscita QQQ va immediatamente a 0;
- se PRESET=1\text{PRESET} = 1PRESET = 1, si imposta Q=1Q=1Q = 1, ovvero l'uscita QQQ va immediatamente a 1.



Simbolo del Flip-Flop Master-Slave con comandi asincroni Clear (C) e Preset (P)

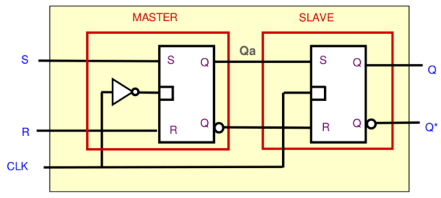
□□
Se PRESET\text{PRESET}PRESET e CLEAR\text{CLEAR}CLEAR si attivano contemporaneamente, il FF si porta nella condizione proibita.

□□
In questo esempio, i comandi PRESET e CLEAR sono attivabili se il loro valore è 1. Si può anche realizzare con i comandi attivabili se il valore è 0.

Flip-Flop Jack-Kilby (FF-JK)

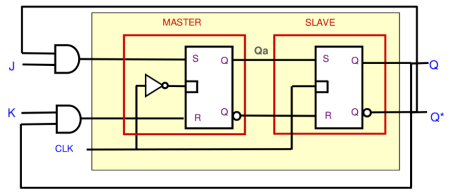
Costruiamo il FF-JK a partire dal D-FF seguendo i passi descritti:

1. a partire dal D-FF, si rimuove lo stato comune D, in modo da avere nuovamente due ingressi distinti sul componente;



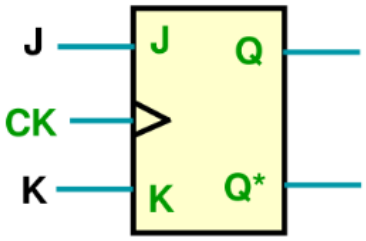
Primo passo della costruzione del FF-JK

2. creiamo una retroazione con 2 AND, definendo due nuovi segnali chiamati JJJ e KKK.



Secondo passo della costruzione del FF-JK

Attraverso questa costruzione otteniamo il Flip-Flop Jack-Kilby (FF-JK), ovvero un componente sequenziale, il cui funzionamento è molto simile a quello del FF-SR, ma senza il problema legato allo stato metastabile.



Simbolo del Flip-Flop Jack-Kilby (FF-JK)

Tavola di Verità

La tavola di verità del FF-JK è definita a lato, utilizzando questa legenda:

- condizione di memoria;
- condizioni di comando;
- condizione per complementare l'uscita.

JJJ	KKK	SSS	RRR	Q{\color{orange}{Q}}Q	Q*{\color{orange}{Q^*}}Q*
0	0	0	0	Q−1Q_{-1}Q−1	Q−1*Q^*_{-1}Q−1*
0	1	0	Q−1Q_{-1}Q−1	0	1
1	0	Q−1*Q^*_{-1}Q−1*	0	1	0
1	1	Q−1*Q^*_{-1}Q−1*	Q−1Q_{-1}Q−1	Q−1*Q^*_{-1}Q−1*	Q−1Q_{-1}Q−1

Dalla tavola di verità riusciamo a capire bene il comportamento di questo componente:

- le condizioni di comando evidenziano che le uscite dipendono dal valore dei vecchi ingressi;
- la condizione di memoria permette di memorizzare gli stati precedenti di Q\color{orange}QQ e Q*\color{orange}Q^*Q*;
- la vecchia condizione di metastabilità diventa la condizione per complementare l'uscita, ovvero un comando per scambiare i valori nelle uscite Q\color{orange}QQ e Q*\color{orange}Q^*Q*.

Si noti come gli ingressi del FF-JK siano strettamente legati alle uscite, quindi è possibile definire le equazioni dello stato futuro QQQ.

Q=JQ−1∗+Q−1K¯Q = JQ^*_{-1} + Q_{-1} \overline K
Q = JQ−1∗ + Q−1K