Lezione 8

Circuiti sincroni

Circuiti sequenziali: i bistabili

Proff. A. Borghese, F. Pedersini

Dipartimento di Scienze dell'Informazione
Università degli Studi di Milano

Sommario

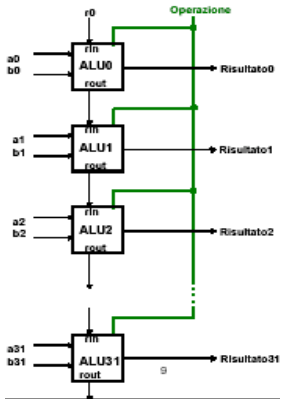
- Il clock. Circuiti sincroni.
- Circuiti sequenziali. I bistabili.
  - Latch asincroni
  - Latch sincroni: tipo SR, tipo D
  - Flip-flop

Perchè esiste il "clock" ?

- *Per quanto tempo dobbiamo mantenere uno stesso ingresso sul circuito?*
- *Quando possiamo leggere l'uscita?*
  - Quando le uscite sono divenute stabili!
- *Quando possiamo presentare nuovi ingressi?*
  - Dobbiamo aspettare che il risultato sia:

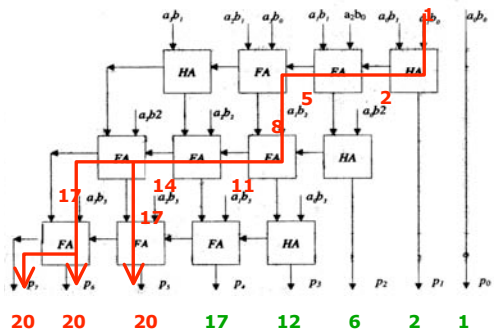
  Calcolato → tempo di calcolo

  Utilizzato → tempo di uscita stabile



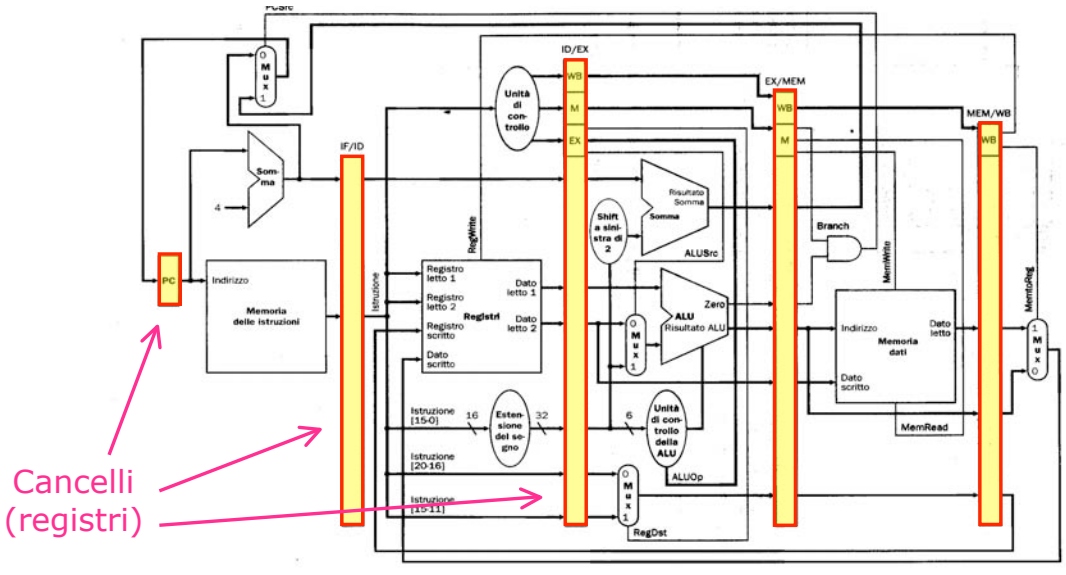
- Necessità:
  sincronizzazione tra uscite e ingressi

- Soluzione: "Cancello" tra le uscite e gli ingressi successivi



"Cancelli" nelle architetture

- Esempio: CPU multi-ciclo
  - Un "cancello" dopo ogni stadio di elaborazione dell'istruzione



Perchè esiste il "clock" ?

- Dove il "cancello" viene inserito, si ***sincronizza*** l'attività di elaborazione
  - Indispensabile per sincronizzare il funzionamento delle varie componenti nelle architetture complesse.
  - Altrimenti i cammini critici renderebbero il funzionamento del circuito sempre più casuale.

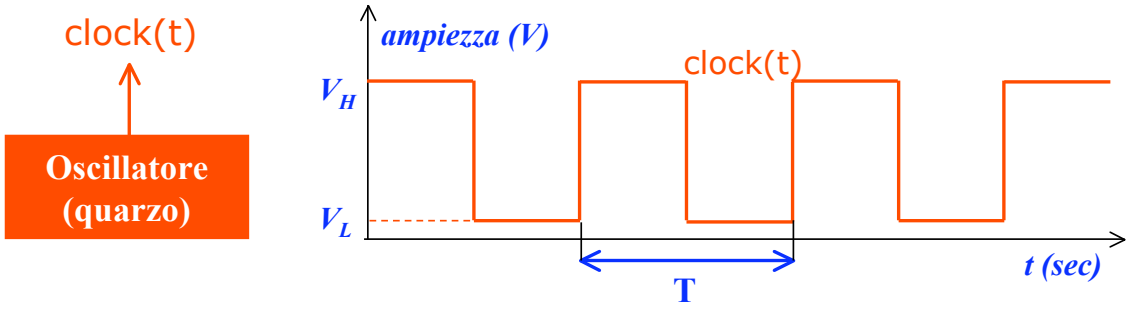
- Cancelli = Barriere di sincronizzazione tra circuiti
  - Memorizzano l'ingresso ad un certo istante
    → registri
  - Lo mantengono stabile in uscita, per un certo periodo
    → orologio in comune

- Presenza di registri → **circuiti sequenziali** (con memoria)
- Orologio in comune → **circuiti sincroni**

Circuiti sincroni e asincroni

- **Architettura sincrona:**
  l'elaborazione e propagazione dei segnali è scandita da un **orologio comune a tutto il circuito (clock)**
  - Il Clock regola l'attività dei "cancelli"
  - Il circuito ha il tempo di stabilizzarsi (transitori critici) fino al successivo impulso di Clock
- **Architettura asincrona:**
  l'elaborazione e propagazione dei segnali avviene **in modo incontrollato**, secondo le velocità di reazione dei circuiti
  - Non ci sono cancelli
  - Non devo mai aspettare l'impulso di clock → massima velocità
- **Progettazione sincrona**
  - il controllo dei transitori/cammini critici è limitato alla parte di circuito tra due cancelli (porte di sincronizzazione)
- **Progettazione asincrona**
  - Devo progettare il circuito in modo che nessun transitorio/cammino critico causi problemi → analisi di tutti i transitori critici possibili.
  - Eccezioni: TRANSPUTER (INMOS, UK)
  - Architettura di calcolatore con CPU asincrona
  - Complessità enorme, sia di progettazione che di programmazione → progetto abbandonato.

Clock

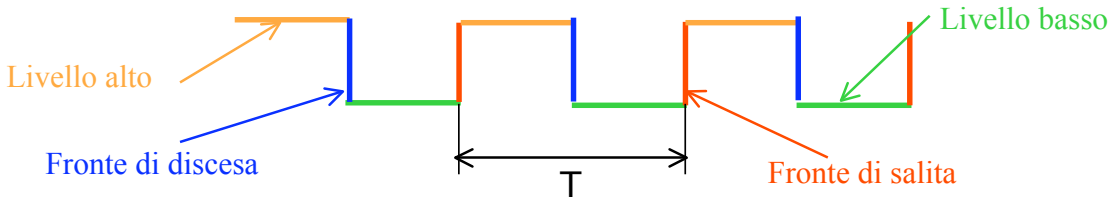


- Periodo: **T** [s] durata di 1 ciclo (secondi)
- Frequenza: ***f*** [Hz]=[s^{-1}] numero di cicli al secondo

T = 1/*f*

- Tempo di salita e discesa trascurabile, rispetto al periodo T.

Utilizzo del clock



- **Metodologia sensibile ai livelli:**
  - Le variazioni di stato avvengono quando il clock è alto (basso). La parte bassa (alta) del ciclo di clock permette la propagazione tra sottocircuiti, così che i segnali si siano stabilizzati quando il clock cambia livello.
- **Metodologia sensibile ai fronti:**
  - Le variazioni di stato avvengono in corrispondenza di un fronte di clock (salita o discesa).

Sommario

- Il clock. Circuiti sincroni.
- Circuiti sequenziali. Bistabili.
  - Latch asincroni
  - Latch sincroni: tipo SR, tipo D
  - Flip-flop

Circuiti sequenziali

- **Circuiti combinatori** = circuiti senza memoria
  - Gli output al tempo t dipendono unicamente dagli input al tempo t

$$Uscita = f\ (\ Ingresso\ )$$

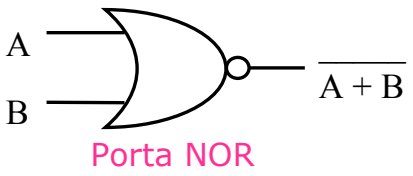
  - Per consentire ad un dispositivo di mantenere le informazioni, sono necessari circuiti con memoria
  - Esempio: distributore automatico
    - Deve ricordare quante e quali monete sono state inserite
    - Deve comportarsi tenendo conto non solo delle monete inserite attualmente (ingresso) ma anche di quelle inserite in precedenza (storia passata)
- **Circuiti sequenziali** = circuiti con memoria (stato)
  - La memoria contiene lo **stato** del sistema:

$$Uscita = f\ (\ Ingresso\ ,\ Stato\ )$$

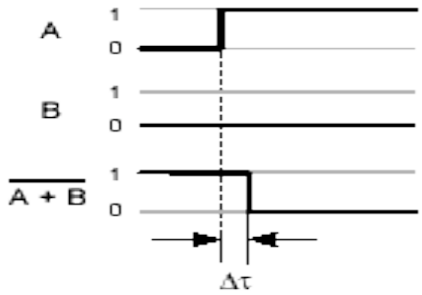
Latch & Bistabili

- Elemento cardine dei circuiti sequenziali è lo **stato**
  - Lo stato riassume il funzionamento negli istanti precedenti e deve essere immagazzinato (memorizzato)
  - Necessità della memoria (bistabili → registri → memorie)
- Elemento base della **memoria** è il **bistabile**
  - dispositivo in grado di **mantenere *indefinitamente* il valore di input**
  - Il suo valore di uscita coincide con lo stato
  - Memoria: insieme di bistabili (bistabili → registri → memorie)
- **Tipologie di bistabile**
  - Bistabili non temporizzati (asincroni) / temporizzati (sincroni).
  - Bistabili (sincroni) che commutano sul livello (**latch**) o sul fronte (**flip-flop**)

Latch Set-Clear (SR) asincrono

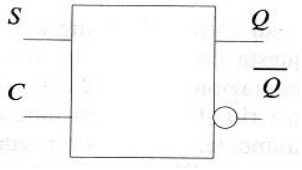


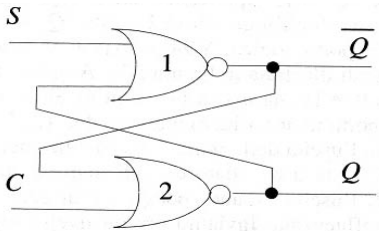
Porta NOR



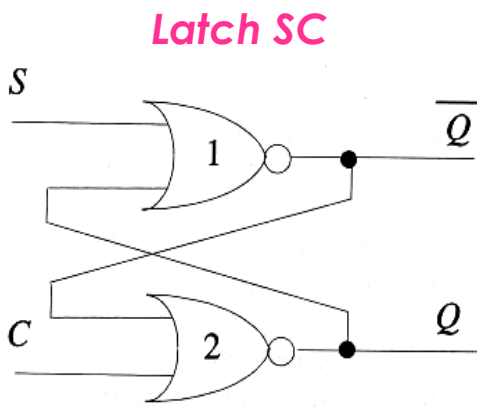
- Il **ritardo Δτ** introdotto dalla porta NOR è alla base del funzionamento di un bistabile.
- Una coppia di porte NOR retroazionate può memorizzare un bit!

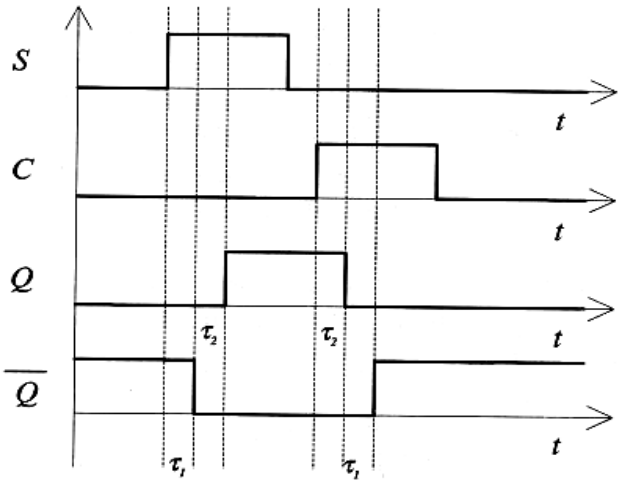
Latch SC





Funzionamento del Latch SC (SR)





- Funzionamento:
  - Set: C = 0, S → 1 Q → 1 (~Q → 0)
  - Reset: S = 0, C → 1 Q → 0 (~Q → 1)
  - Comportamenti anomali:
  - S = 1 , C: 0→1 Q=~Q=0 (anomalia)

Tabella delle transizioni

- Tabella delle transizioni: **Q* = f(Q,I) = *f*(Q,S,C)**
  - Q: valore dell'uscita attuale: stato corrente
  - Q*: uscita al tempo successivo: stato prossimo

	Q*			
Q	**SC = 00**	**SC = 01**	**SC = 11**	**SC = 10**
0	0	0	X	1
1	1	0	X	1
	Q* = Q	**Clear / Reset**		**SET**

Tabella delle transizioni

- Tabella delle transizioni ***f*** :

$$Q^* = f(I, Q)$$

SC / Q	SC=00	SC=01	SC=11	SC=10
Q=0	**0**	**0**	**X**	**1**
Q=1	**1**	**0**	**X**	**1**

S	C	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	**X**
1	1	1	**X**

- Considerando lo stato Q come ingresso ottengo la tabella delle verità di Q*:

Tabella delle eccitazioni

- *Data la transizione: Q → Q*, qual'è la configurazione di valori di ingresso che la determina?*

- Tabella delle eccitazioni ***h***:

$$Q \rightarrow Q^* = h(I)$$

Q	Q*	S	C
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

- *Per il Latch SR:*

$$Q \rightarrow Q^* = h(S,C)$$

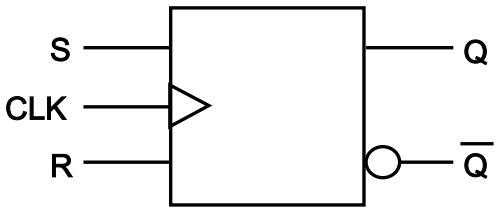
Sommario

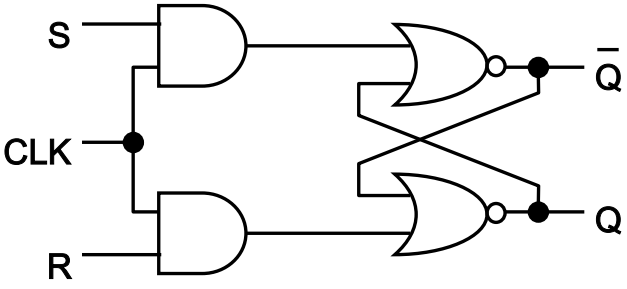
- Il clock. Circuiti sincroni.
- Circuiti sequenziali. Bistabili.
  - Latch asincroni
  - Latch sincroni: tipo SR, tipo D
  - Flip-flop

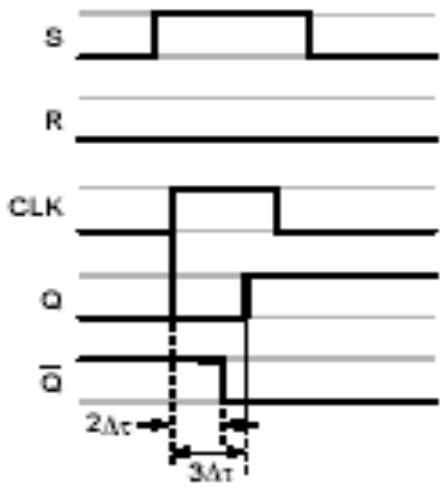
Latch SR sincrono

Struttura:

- Latch SR + Porte AND tra il clock e gli ingressi.
  - Solo quando il clock è alto i "cancelli" (porte AND) fanno passare gli input → LATCH







Sintesi della funzione logica

T	Q	S	R	Q*
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	X
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	X
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	X
1	1	0	0	1
1	1	0	1	0
1	1	1	0	1
1	1	1	1	X

Tabella delle transizioni: Q*= f(S,R,T,Q)

TQ	SR = 00	SR = 01	SR = 11	SR = 10
00	0	0	X	0
01	1	1	X	1
11	1	0	X	1
10	0	0	X	1

$$Q^* = \bar{T}Q\bar{S}\bar{R} + \bar{T}Q\bar{S}R + \bar{T}QS\bar{R} + T\bar{Q}S\bar{R} + TQ\bar{S}\bar{R} + TQS\bar{R} +$$
$$\left(+\bar{T}QSR + T\bar{Q}SR\right) =$$
$$= \bar{T}Q\bar{R} + TQ\bar{R} + \bar{T}QR + T\bar{Q}S =$$
$$= \bar{T}Q + T\left(Q\bar{R} + \bar{Q}S\right)$$

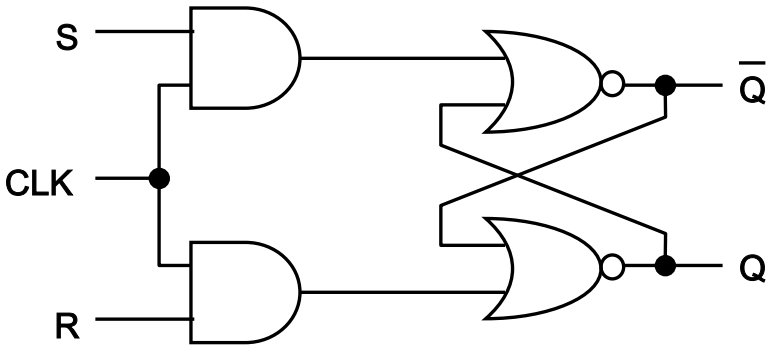
clock alto → Q* = Q~R + ~QS

clock basso → Q* = Q (*status quo*)

Analisi della funzione logica sintetizzata

$$Q^* = \bar{T}Q + T\left(Q\bar{R} + \bar{Q}S\right)$$

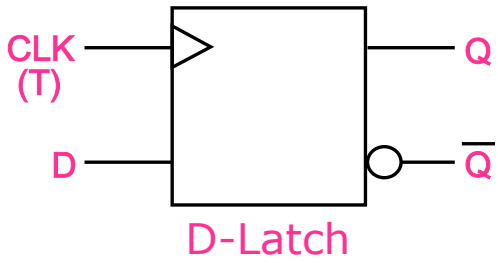
$$T = 1 \rightarrow \quad Q^* = Q\bar{R} + \bar{Q}S : \quad \begin{cases} \text{Set}: S = 1, R = 0 & Q^* = Q + \bar{Q} = 1 \\ \text{Reset}: S = 0, R = 1 & Q^* = 0 + 0 = 0 \end{cases}$$



Latch D sincrono

- LATCH D sincrono:
  - Memorizza il valore presente all'ingresso dati quando il clock è alto

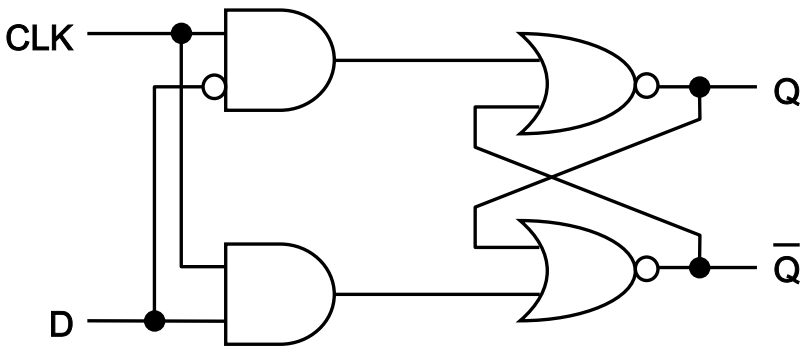
if CLK = 1 **then** Q*=D **else** Q*=Q



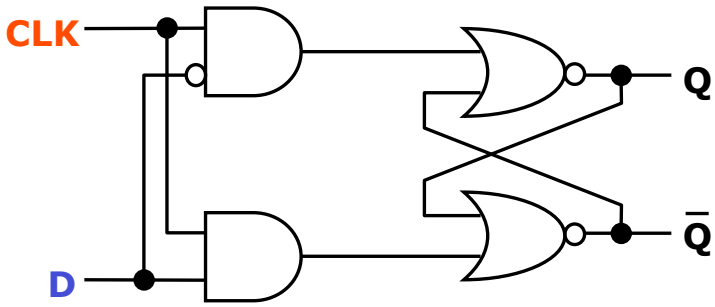
D-Latch

Comportamento del Latch D

- **Clock BASSO:**
  - L'uscita Q rimane bloccata sull'ultimo valore assunto
- **Clock ALTO:**
  - L'uscita Q insegue l'ingresso D (con $3\Delta\tau$ di ritardo)



La struttura del latch D



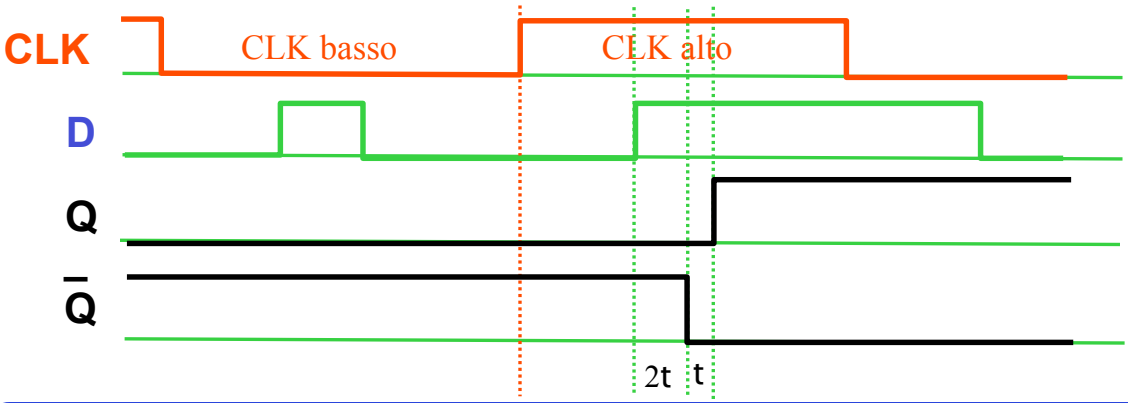


Tabella delle transizioni

- Dalla tabella delle transizioni **Q* = f(T,Q,D)** calcolo la corrispondente funzione logica:

Tabella delle transizioni

T	Q	D	Q*
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Funzione logica:

$$Q^* = \bar{T}Q\bar{D} + \bar{T}QD + T\bar{Q}D + TQD =$$

$$= \bar{T}Q + TD$$

Clock T alto:
→ Q* = D (input)

Clock T basso
→ Q* = Q (status quo)

Sommario

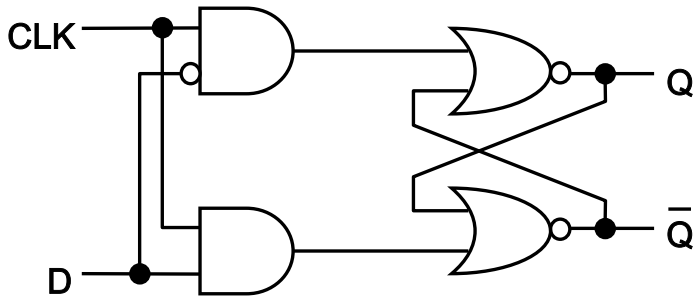
- Il clock. Circuiti sincroni.
- Circuiti sequenziali. Bistabili.
  - Latch asincroni
  - Latch sincroni: tipo SR, tipo D
  - Flip-Flop

Latch: Bistabili level-sensitive

- I latch sono dispositivi **trasparenti:**
  - Per tutto il tempo in cui il clock è attivo (alto), il valore di D viene riportato in uscita:

Q = D : uscita collegata all'ingresso

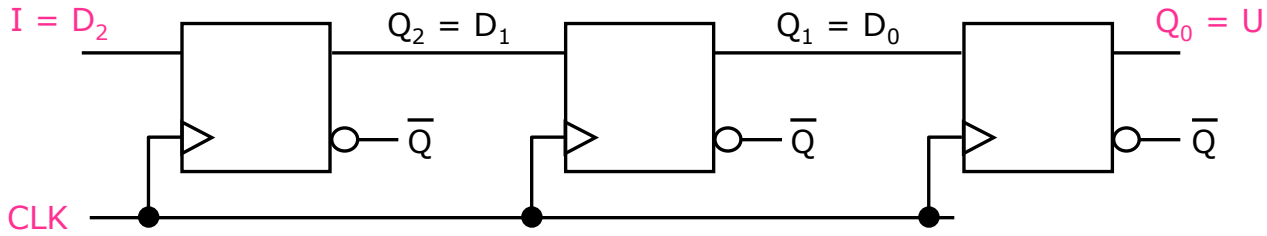
- A noi interessa memorizzare l'informazione in un determinato istante



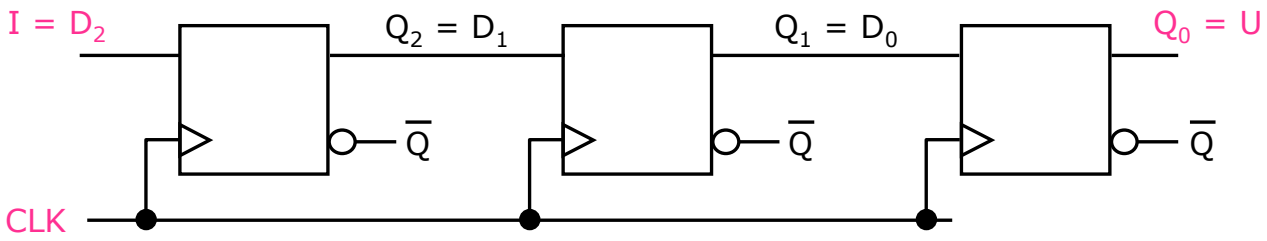
if	**CLK = 1**
then	**Q*=D**
else	Q*=Q

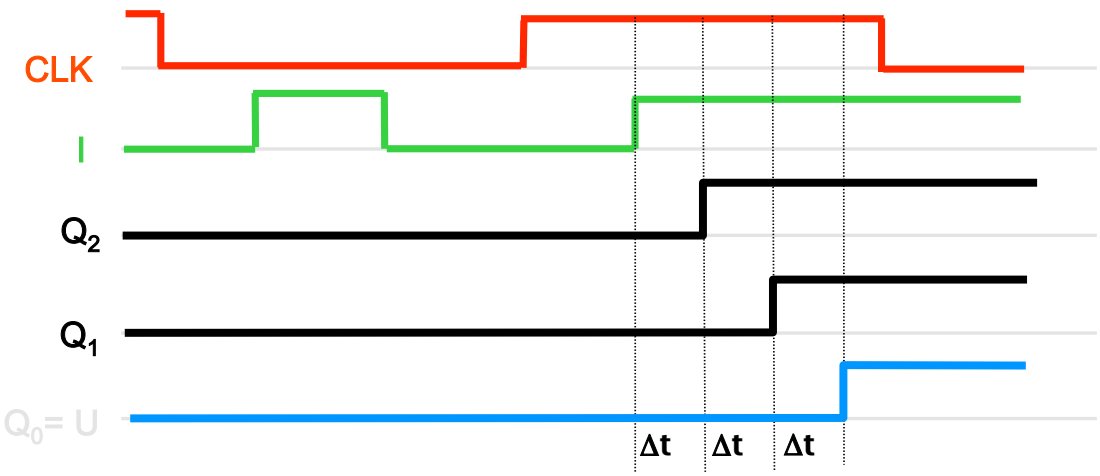
Problemi con i latch sincroni

- Registro a scorrimento (shift register)
  - Un unico ingresso **I** e un'unica uscita **U**
  - In presenza di segnale attivo (clock alto), il contenuto deve essere spostato verso destra di una posizione
- Realizzazione mediante bistabili **LATCH**:
  - Funziona?



Shift register con i latch

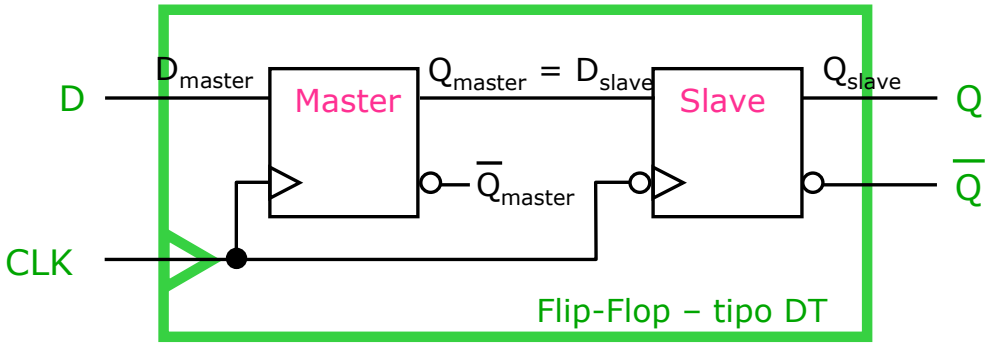




Bistabili "edge sensitive": i Flip-Flop

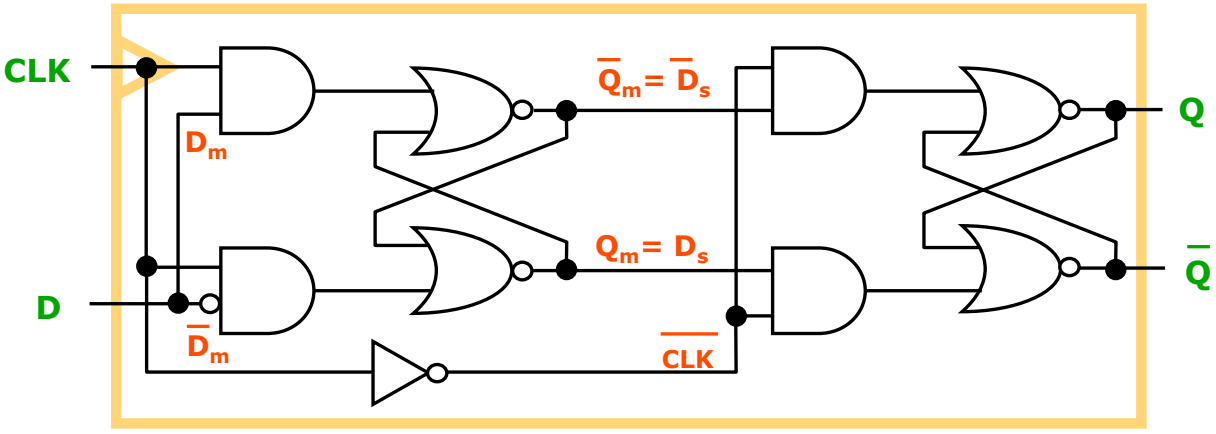
- Dispositivi attivi sul fronte del clock (*edge sensitive*):
  - il loro stato (uscita) può commutare solo in corrispondenza del fronte di salita o di discesa del clock.

Bistabile tipo DT – Configurazione "Master-Slave"

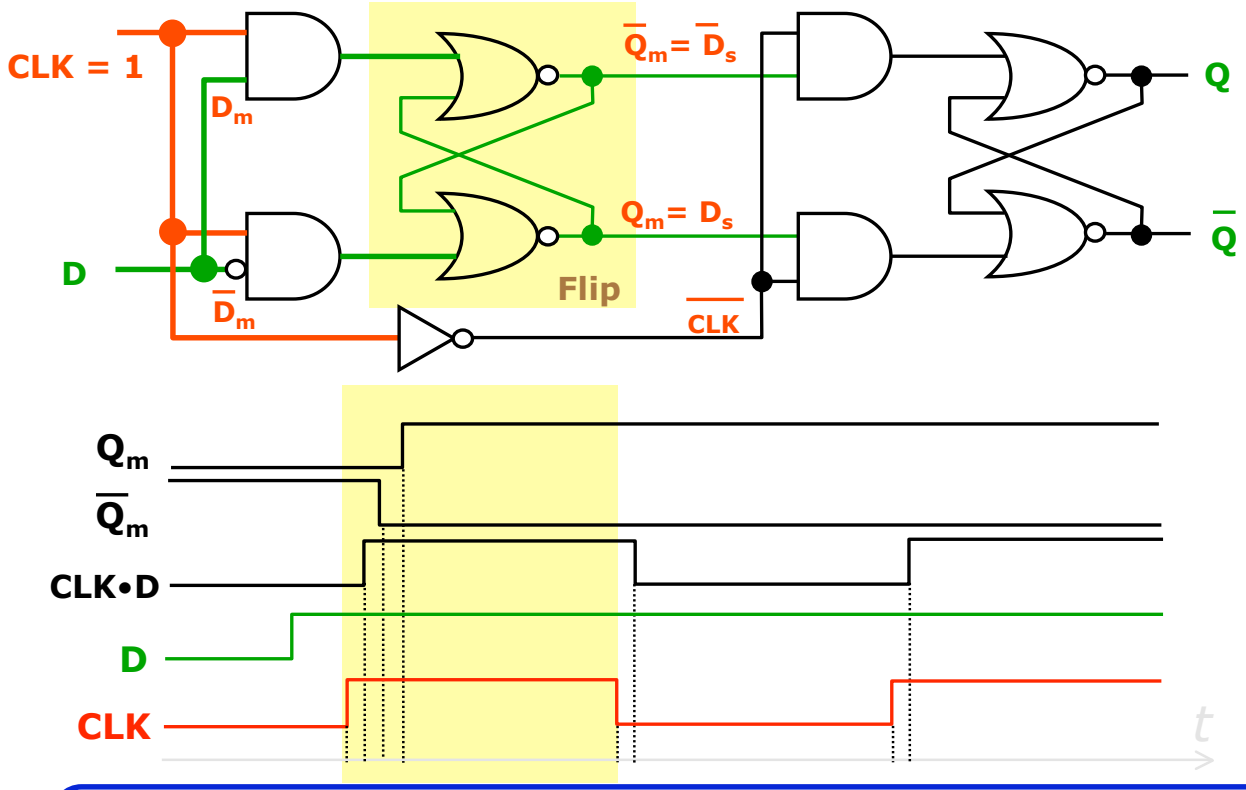


Flip-flop: struttura master–slave

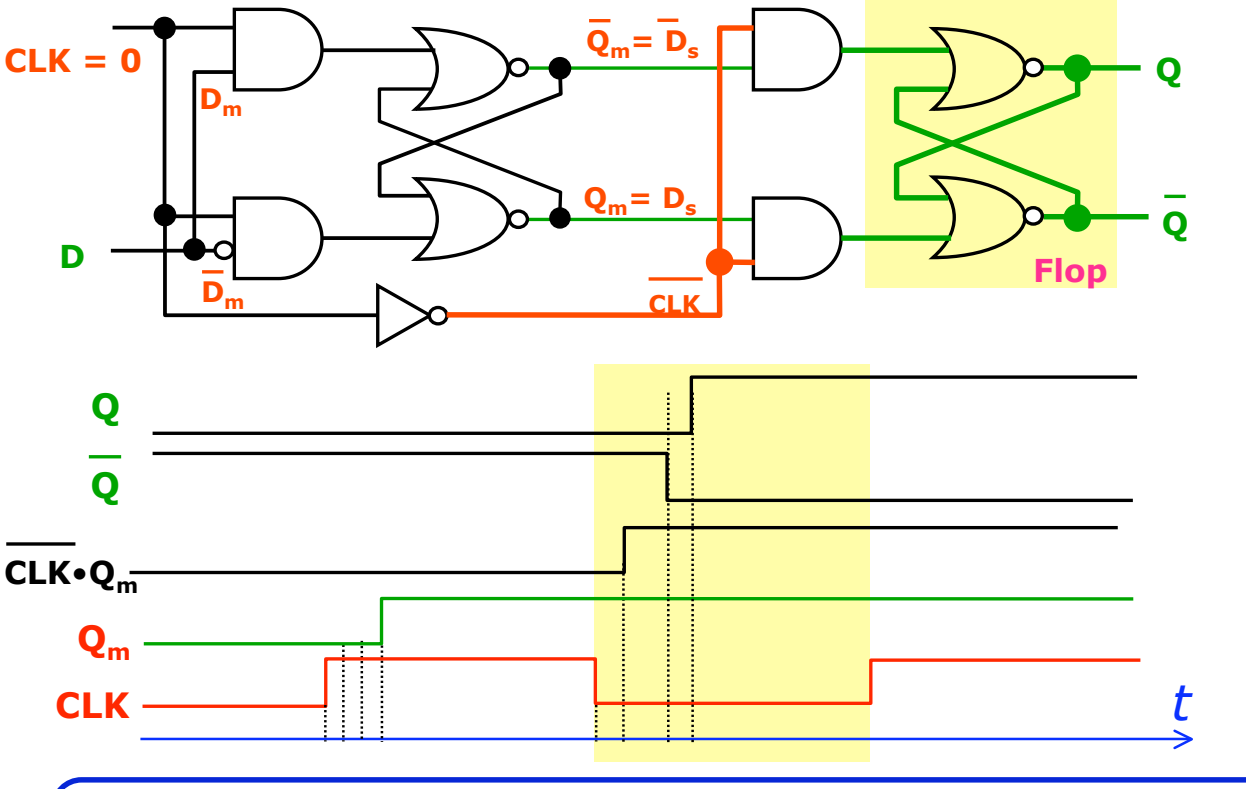
- **FLIP (clock ALTO)**: L'ingresso D viene memorizzato nel latch MASTER
  - L'ingresso è aperto, ma l'uscita è bloccata
- **FLOP (clock BASSO)**: l'uscita stabile del latch MASTER viene propagato al latch SLAVE
  - L'ingresso è bloccato, l'uscita è stabile.



Funzionamento: **FLIP**



Funzionamento: **FLOP**



Funzionamento dei Flip-Flop

- Fronte di **SALITA – FLIP**
  - Attivato lo stadio **MASTER**
  - Memorizzato il dato sull'ingresso: ***D → STATO***
  - Uscita invariata

  Cancello ingresso aperto, cancello uscita chiuso

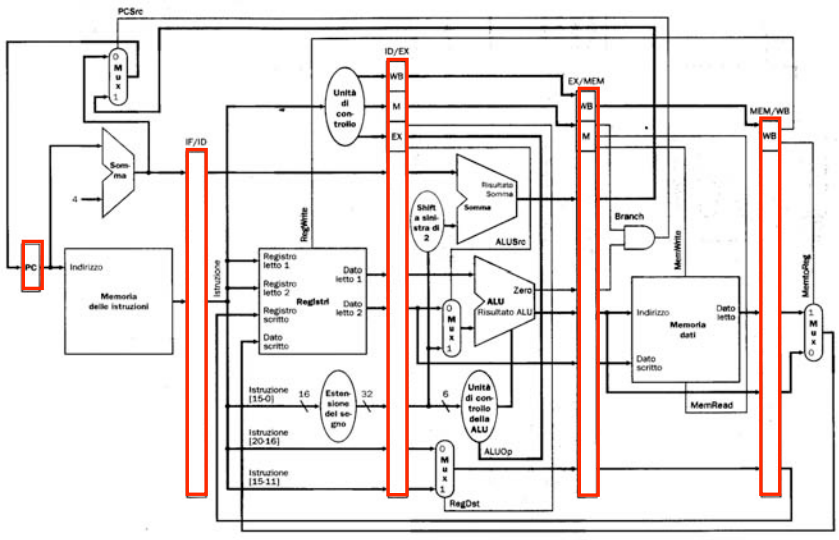
- Fronte di **DISCESA – FLOP**
  - Attivato stadio **SLAVE**
  - Presenta il dato memorizzato in uscita: ***STATO → Q***
  - Ingresso isolato

  Cancello ingresso chiuso, cancello uscita aperto

Sincronizzazione mediante flip-flop

- I "cancelli" devono **disaccoppiare** i diversi sottosistemi logici
  - "raccogliere" i segnali, senza farli passare, e "rilanciarli" ad un determinato istante
  - **Cancello doppio**: **ingresso** e **uscita**
  - Mai aperti contemporaneamente

- Cancelli = registri costituiti da gruppi di flip-flop



Registri

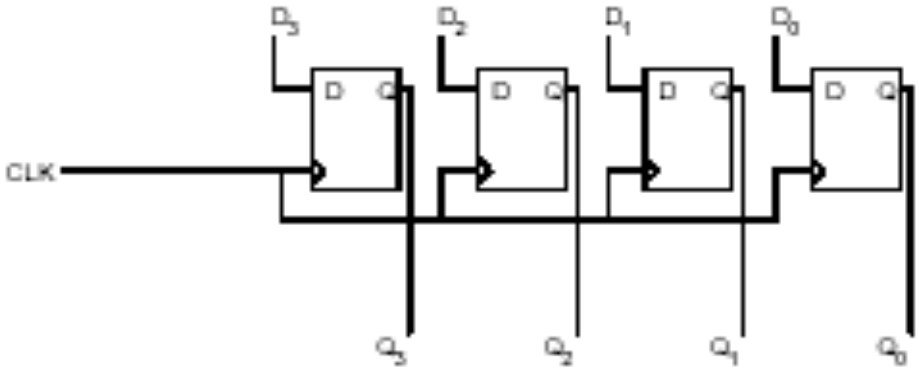
- Registro a **N** bit: N Flip-flop tipo DT

  SCRITTURA:

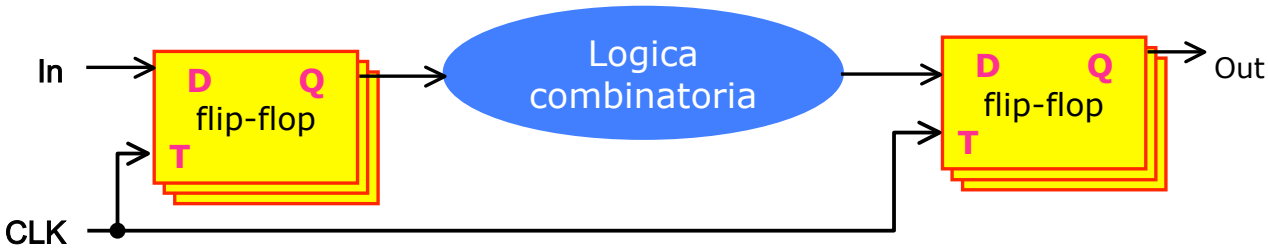
  - L'impulso di CLK memorizza i dati sugli ingressi D

  LETTURA:

  - I dati memorizzati sono presenti sulle uscite Q



Struttura di architetture sincrone

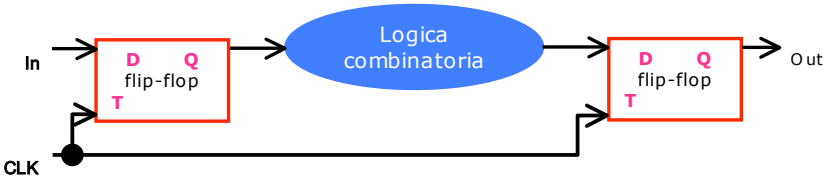


Cancello → Circuito combinatorio → Cancello

- Sincronizzazione: la logica combinatoria deve terminare la propria commutazione in tempo utile

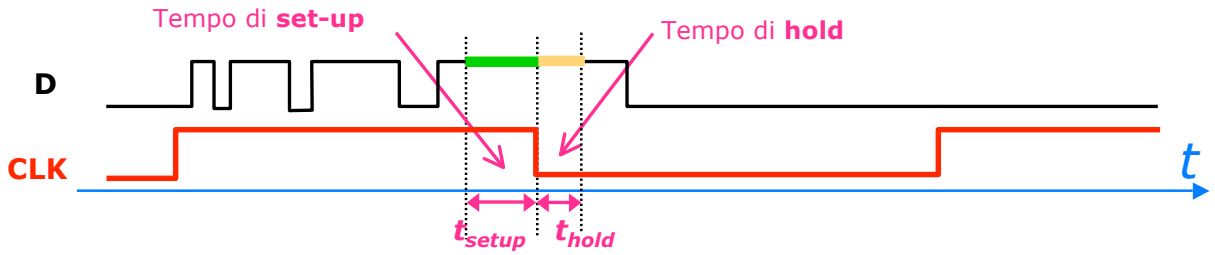
- Dimensionamento del periodo di clock **T**:
  - La commutazione del clock deve avvenire **dopo** che la logica combinatoria abbia terminato tutte le commutazioni
  - Il tempo necessario alla logica combinatoria per commutare dipende dal *cammino critico*

Dimensionamento clock: **set-up, hold**



- Tempo di **set-up**:
  tempo minimo per cui deve rimanere stabile l'input D prima del fronte di clock.

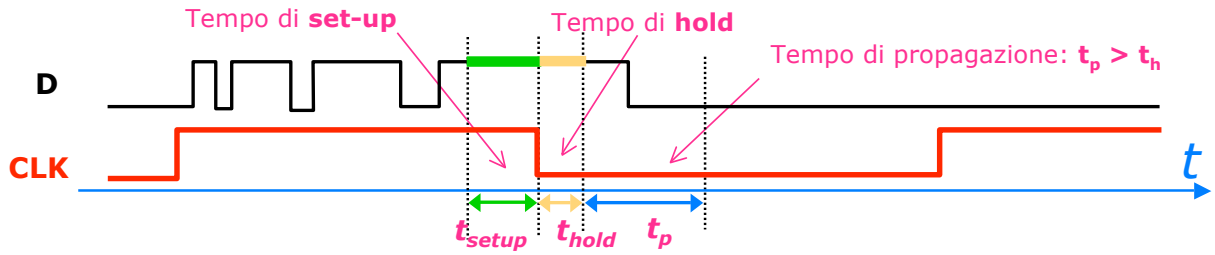
- Tempo di **hold**:
  tempo minimo per cui deve rimanere stabile l'input D dopo il fronte di clock
  - solitamente trascurabile



Dimensionamento clock: **propagazione, skew**

- Tempo di **propagazione**: è il tempo necessario per propagare il segnale dall'uscita slave alla logica combinatoria: t_{prop}
  - maggiore del tempo di hold t_h

- Tempo di **skew**: ritardo massimo del clock t_{sk}



$$T_{clock} > k * (t_h + t_{prop} + t_{sk} + t_{setup})$$