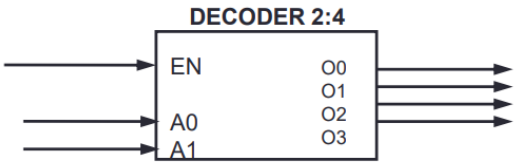
DEMULTIPLEXER: smista un singolo input in una delle possibili n uscite. È una rete logica con 1 ingresso dato, n segnali di controllo e 2^n. Si dice anche decoder in quanto viene usato per decodificare un segnale binario (con EN a 1).

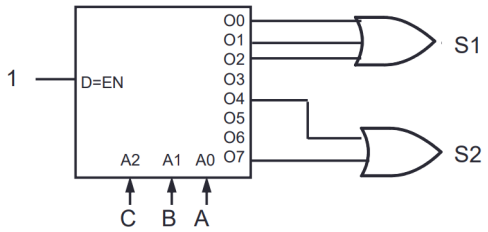


Può essere usato come generatore di mintermini.

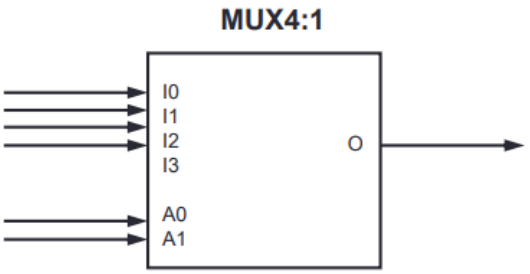
Esempio

S1 = A'BC' + A'B'C + A'B'C'

S2 = AB'C' + ABC



MULTIPLEXER: è quel blocco logico che permette di deviare su un'unica uscita un segnale proveniente da uno tra n possibili ingressi. È una rete logica avente 2^n ingressi di tipo dati e n ingressi di tipo segnali di controllo ed 1 uscita: in ogni istante il dato presente all'ingresso selezionato, mediante la configurazione dei segnali di controllo, viene riportato in uscita.



Sintesi attraverso la tabella di verità

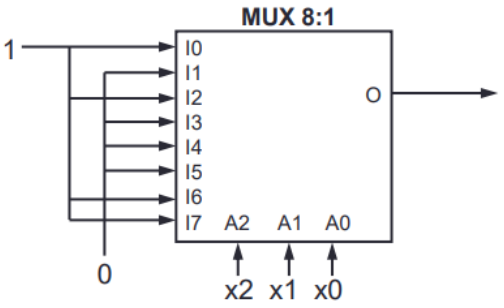
A_1	A_0	I_3	I_2	I_1	I_0	O
0	0	×	×	×	0	0
0	0	×	×	×	1	1
0	1	×	×	0	×	0
0	1	×	×	1	×	1
1	0	×	0	×	×	0
1	0	×	1	×	×	1
1	1	0	×	×	×	0
1	1	1	×	×	×	1

O = I3A1A0 + I2A1A0' + I1A1'A0 + I0A1'A0'

Può inoltre essere usato come generatore di tabelle di verità.

Esempio

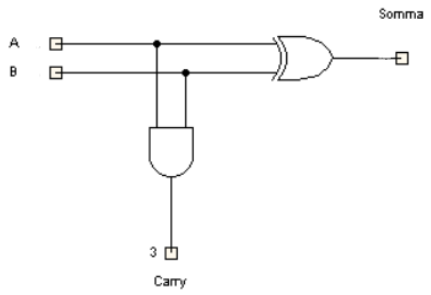
F(x0,x1,x2) = m0+m2+m6+m7



HALF ADDER

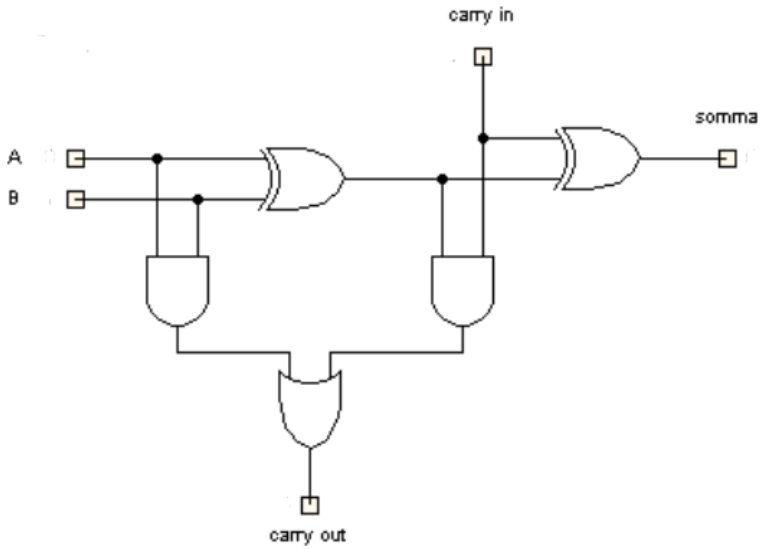
Somma due bit in input restituendo somma ed eventuale riporto.

A	B	S	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



FULL ADDER

È possibile realizzare un sommatore che tenga conto del riporto in ingresso mediante due half-adder. Il riporto dell'operazione precedente viene sommato ai due bit attuali.

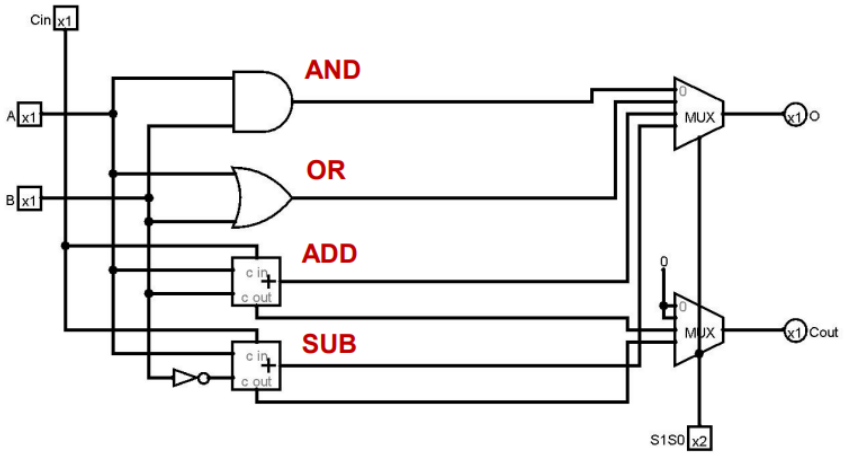


Il riporto di un bit si usa come *carry in* del sommatore completo alla sua sinistra (cifra più significativa).

Si può realizzare un sommatore a n-bit realizzato collegando n full-adder.
Vantaggi: chiara, facilmente replicabile, ottimizzata per il numero di porte complessivo.
Svantaggi: propaga il carry dal bit meno significativo al bit più significativo.

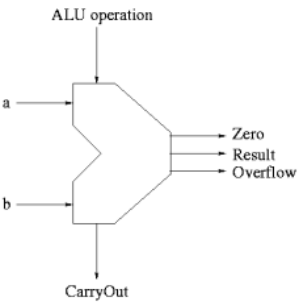
ALU

L'ALU (arithmetic logic unit o unità aritmetico logica) è un circuito combinatorio in grado di svolgere operazioni aritmetiche e/o logiche su due operandi. L'elenco delle operazioni possibili viene definito in fase di progetto e, mediante opportuni segnali, si seleziona quale fare. Normalmente come output fornisce il risultato e alcuni FLAG (es. negativo, overflow…)



Combinando un sommatore e alcune porte logiche è possibili realizzare un ALU che effettua operazioni su singoli bit. Le 4 operazioni, selezionabili tramite multiplexer, considerate sono:

- AND logico
- OR logico
- Somma (con/senza carry in ingresso)
- Sottrazione (con/senza carry in ingresso)



Combinando opportunamente più ALU a 1 bit è possibile realizzare una ALU a più bit.

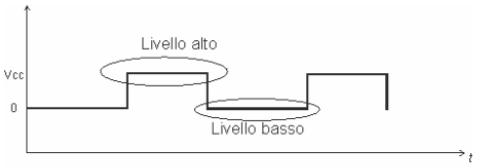
RETI SEQUENZIALI: reti logiche in cui in ogni istante le uscite dipendono non solo dalla configurazione degli ingressi in quell'istante, ma anche dalle configurazioni degli ingressi negli istanti precedenti. Il comportamento quindi si basa su una **memoria**, il circuito deve ricordare lo stato passato.

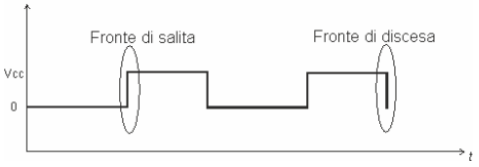
- **Stato presente**: stato attuale
- **Stato futuro**: in seguito alla variazione degli ingressi il sistema può calcolare in ogni istante lo stato futuro.

Le reti sequenziali possono essere asincrone o sincrone:

- **asincrone**: le variazioni delle configurazioni di ingresso vengono sentite e modificano lo stato e le uscite in qualsiasi istante
- **sincrone**: le variazioni delle configurazioni di ingresso vengono sentite e modificano lo stato e le uscite solo in presenza di un opportuno evento di sincronizzazione.

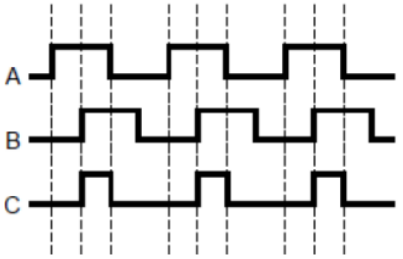
Clock: segnale attivo a cui si associa l'evento di sincronizzazione. È generato da un circuito che emette un segnale impulsivo periodico con una precisa durata (pulse width). È un segnale **free-running** (continua finchè il sistema è alimentato), di tipo **periodico** (tempo di clock) e con una certa **frequenza** (inverso del periodo).





Logica a livello: l'evento si verifica mentre il clock è attivo
Logica a fronte (edge-triggered): l'evento si verifica al cambiamento del clock. Di solito si usa il fronte di salita.

Il segnale di clock sequenzializza tutti gli eventi. Oltre al clock primario esistono clock secondari per eseguire più azioni nello stesso clock. Per questo si parla del clock della CPU, del clock di sistema o di clock multipli.

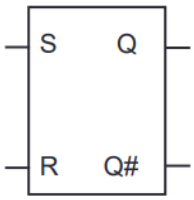


Gli elementi di base delle reti sequenziali sono gli elementi di memoria chiamati **bistabili**, che mantengono al loro interno il valore **0 (R - Reset)** o il valore **1 (S - Set)**. Esistono blocchi elementari per memorizzare lo stato attuale.

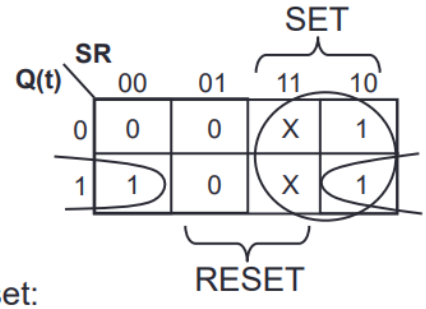
Memoria binaria (**bistabile asincrono**): elemento capace di memorizzare il valore di una variabile di stato binaria e di commutare alla presenza di un'opportuna configurazione di ingresso.

SET-RESET: rete con due ingressi S e R e una uscita Q (ed una uscita complementata Q#). L'uscita Q assume il valore 1 quando S=1 e R=0 e il valore 0 quando S=0 e R=1. L'uscita rimane inalterata quando S=R=0. La combinazione di ingresso S=R=1 non si deve mai

verificare (configurazione proibita).



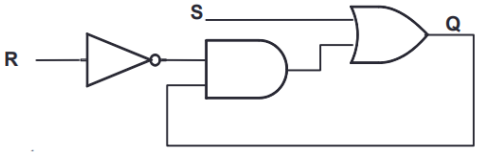
S R	Q(t)	Q'(t)
0 0	Q(t-1)	Q' (t-1)
0 1	0	1
1 0	1	0
1 1	-	-



- Funzione di eccitazione del Set-Reset:

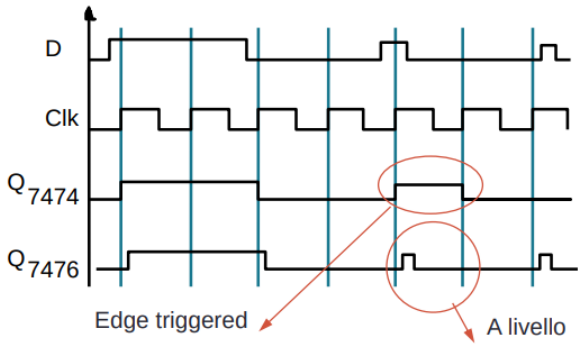
$$Q(t+1)=S+R'Q(t)$$

Esistono anche altri realizzazioni mediante NOR e NAND



LATCH (livello): bistabile sincrono trasparente capace di memorizzare segnali di ingresso in funzione del clock. Pertanto la transizione di stato avverrà per tutto il tempo in cui il clock è attivo (proprietà di trasparenza). Si dice quindi che il latch è trasparente agli ingressi quando l'enable è attivo.

Esempi di comportamenti di un Latch D (7476) e di un FF-D (7474)

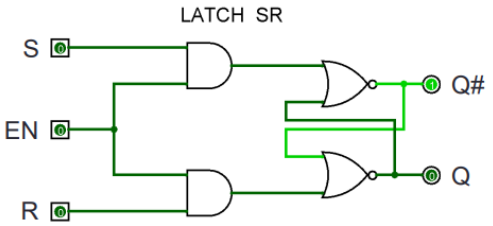


FLIP FLOP (fronte): dispositivo bistabile privo della proprietà di trasparenza. A differenza del latch dove i cambiamenti avvengono in funzione degli ingressi nel flip flop il cambiamento (edge-triggered) è conseguenza di un ingresso di controllo sincrono (clock) o asincrono (preset o clear). Si definiscono bistabili sincroni a commutazione sul fronte perché campionano il valore degli ingressi su un fronte.

Latch S-R

Viene aggiunto un segnale EN (enable):

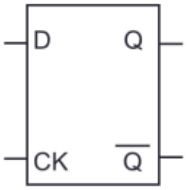
- se non è attivo (0) il secondo stadio ha la configurazione di ingresso (0,0) e rimane nello stato corrente (hold)
- se è attivo (1) la rete è sensibile al cambiamento degli ingressi: se sono entrambi a 0 la rete rimane in hold, altrimenti la rete cambia di stato (set o reset).



D Latch

Memoria capace di mantenere l'uscita costante se il segnale di clock (o enable) non è attivo e di cambiare l'uscita campionando l'ingresso quando il segnale di clock/enable è attivo. Il

latch D, a differenza del latch SR, risolve il problema della configurazione vietata.



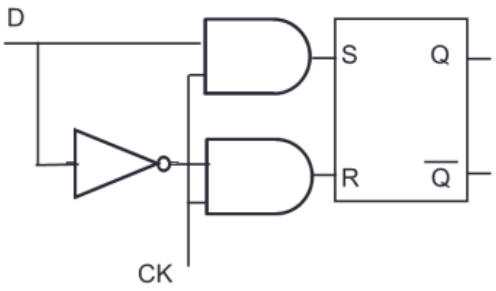


Tabella di verità

D CK	Q Q#	
- 0	Q Q#	
0 1	0 1	RESET
1 1	1 0	SET

Flip Flop D

Flip-flop D (derivato da questo latch) è il flip flop più usato per memorizzare segnali il cui valore è significativo solo in un preciso istante (sul fronte del clock).

Flip Flop JK

Risolve il problema dell'ingresso proibito (1 1) del FF-SR che svolgerà una funzione di toggle.

- **Q(t+1)=Q(t)K'+Q(t)'J**
- Vengono usati per il campionamento dei dati:
  - per memorizzare dato= 0 JK=01
  - per memorizzare dato=1 JK=10
- basta collegare il dato a J e collegare K a J'

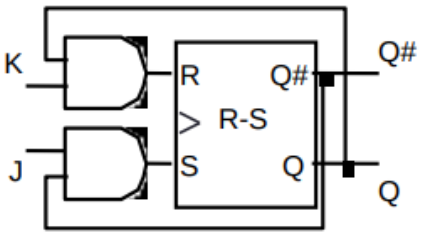
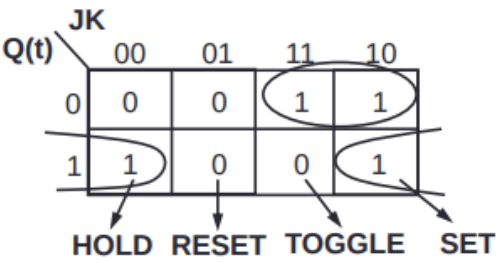


Tabella di verità

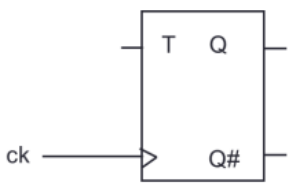
J	K	Q+	Qn+	Descrizione
0	0	Q	Qn	Memoria (nessun cambiamento)
0	1	0	0	Reset
1	0	1	1	Set
1	1	Qn	Q	Toggle (complemento)



Flip Flop T

Vengono usati per commutare lo stato di uscita.

- Proprietà: se T=1 l'uscita Q ha frequenza dimezzata rispetto al clock
- Applicazioni: è il componente base dei contatori, infatti collegando a cascata vari flip-flop T ad ogni uscita si ottiene un clock dimezzato rispetto al clock precedente.



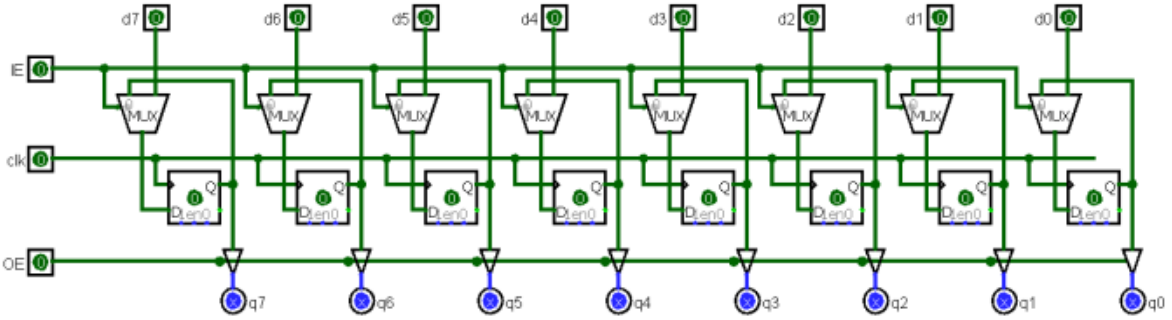
- **Q(t+1)=Q(t)T'+Q(t)'T**

T	Q+	Descrizione
0	Q	Memoria (nessun cambiamento)
1	Qn	Toggle (complemento)

SR:	Q(t+1)=S+R'Q(t)
D:	Q(t+1)=D
J-K:	Q(t+1)=JQ'(t)+K'Q(t)
T:	Q(t+1)=TQ'(t)+T'Q(t)

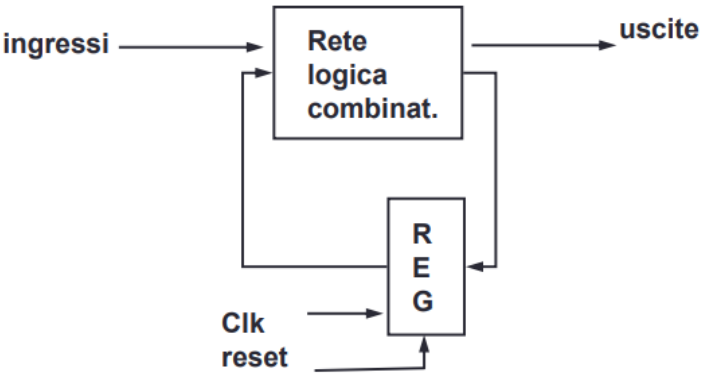
Q(t)	Q(t+1)	S	R	D	J	K	T
0	0	0	-	0	0	-	0
0	1	1	0	1	1	-	1
1	0	0	1	0	-	1	1
1	1	-	0	1	-	0	0

REGISTRI: elemento di memoria in cui n flip-flop vengono controllati dallo stesso clock formando un'unità in grado di memorizzare n bit.



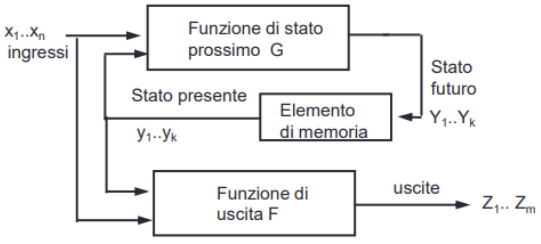
Ogni unità di memorizzazione viene detta cella di memoria. Si codificano in n bit il numero da utilizzare e, tramite un decoder, si producono segnali di abilitazione per la cella in questione.

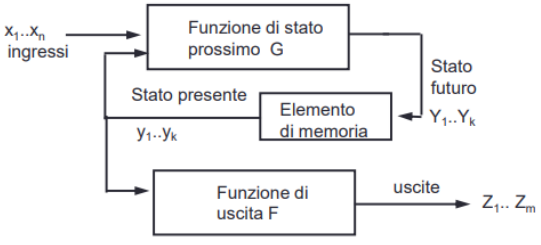
RETI SEQUENZIALI: rete che memorizza le informazioni sulle configurazioni di ingresso che si verificano nel tempo; la memorizzazione avviene in stati interni. Le variabili di stato definiscono lo stato attuale della rete.



FSM (Finite State Machine): macchine a stati finiti in cui gli elementi di memoria sono i FF con un unico segnale di clock. L'insieme dei FF è detto registro di stato.

- **Automa di Mealy**: il valore delle uscite dipende dallo stato presente e dagli ingressi in quell'istante
- **Automa di Moore**: il valore delle uscite dipende solo dallo stato presente e non dagli ingressi in quell'istante (codifico le uscite in base alle stato).





La maggior parte delle reti sequenziali sincrone sono descrivibili come FSM

Sintesi di reti sequenziali

1. Descrizione comportamentale a parle
2. Diagramma degli stati con tabella di verità
3. Minimizzazione degli stati
4. Tabella delle transizioni e delle uscite
5. Implementazione

Una rete sequenziale può essere rappresentata da un diagramma degli stati che corrisponde a un grafo con tanti nodi quanti gli stati e tanti archi quante le transizioni.

Nel diagramma vengono inoltre rappresentati i valori delle uscite per ogni stato:

- negli archi se il modello è di Mealy
- nei nodi se il modello è di Moore