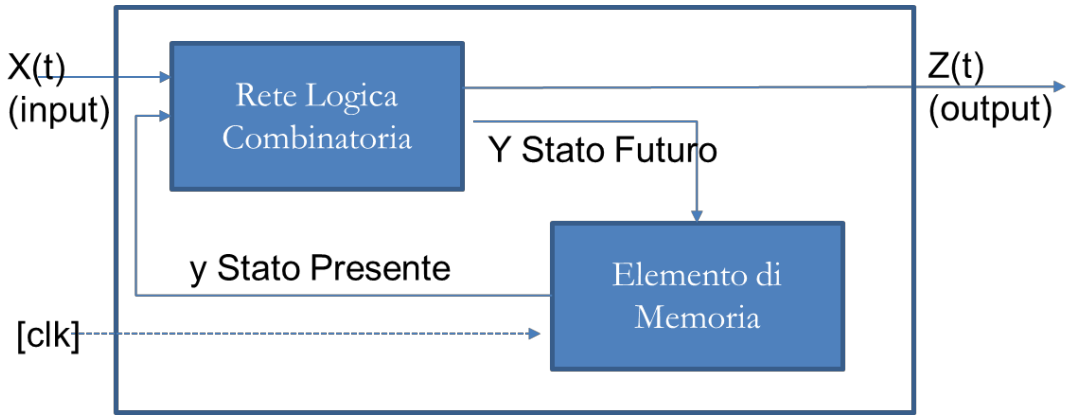
LE RETI LOGICHE SEQUENZIALI

Le **reti sequenziali** sono reti logiche in cui *le uscite dipendono non solo dalla configurazione degli ingressi in quell'istante, ma anche dalle configurazioni degli ingressi negli istanti precedenti*. Le reti sequenziali quindi devono conservare **memoria** degli eventi passati nel proprio **stato interno.**

$$z(t)=f(x_1(t), \ldots, x_n(t),t)$$

Variazioni delle configurazioni di ingresso modificano oltre che le uscite anche lo stato interno.

Lo stato interno attuale si dice **stato presente** mentre lo stato della rete logica in seguito alla variazione degli ingressi si definisce **stato futuro**.



La rete sequenziale è una rete logica che è composta da un blocco di reti combinatorie e da un blocco contenete elementi di memoria capaci di tenere traccia nel tempo dello stato.

Dalla rete logica viene creato lo stato futuro che dopo essere memorizzato diventa lo stato presente. Lo stato presente è per la rete combinatoria un input come gli ingressi X. ***La rete combinatoria è composta da due reti***:

- Una per generare le uscite
- Una per generate lo stato futuro

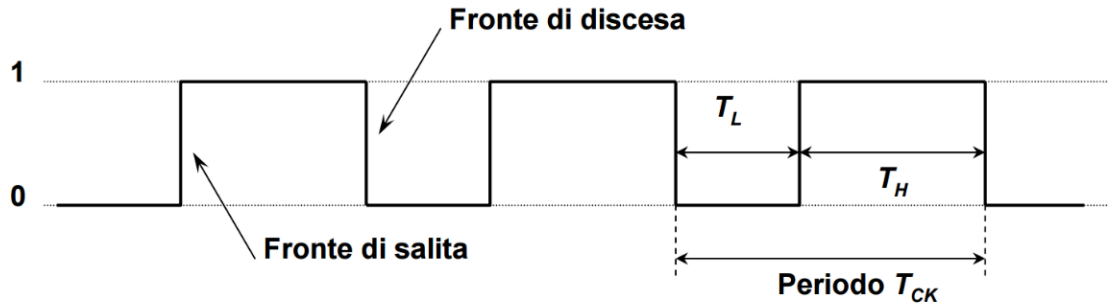
→ Ha retroazione interna

Le reti sequenziali possono essere:

1) **Asincrone:** *se le variazioni delle configurazioni di ingresso vengono sentite e modificano lo stato e le uscite in ogni istante → l'elemento di memoria non appena vede una variazione dello stato futuro lo memorizza e cambia lo stato presente*

2) **Sincrone:** *se le variazioni delle configurazioni di ingresso vengono sentite e modificano lo stato e le uscite **SOLO** in presenza di un opportuno segnale di sincronizzazione* = **segnale di clock**

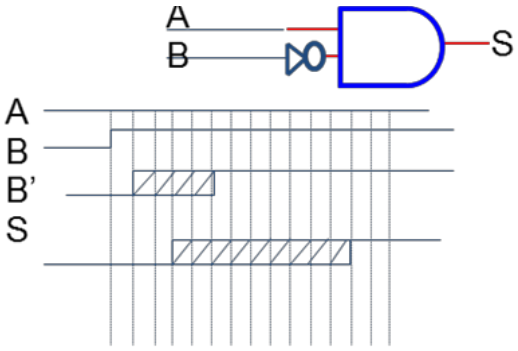
→ L'evento di sincronizzazione è normalmente associato ad un segnale attivo (il **clock**) o al cambiamento dello stato del segnale di sincronizzazione (fronte del clock). **Il segnale di clock è quindi un ingresso aggiuntivo sempre presente nelle reti sequenziali.**



Nelle reti logiche ogni evento elementare si verifica in un ciclo di clock:

- ***mentre il clock è attivo*** (di solito alto) si dice che la logica lavora "a **livello**",
- ***al cambiamento del clock*** si dice che l'evento è "**edge- triggered**" o "a **fronte**".

Tempo di propagazione: *tempo impiegato dal gate a propagare in uscita il cambiamento del segnale di ingresso.*



→ Il tempo di propagazione è il caso PEGGIORE nel cammino più lungo: in questo caso è 7+5= 12ns. Esiste invece un periodo transitorio in cui l'uscita ha comportamenti non stabili. Quello che succede durante il tempo di propagazione è un comportamento transitorio.

Per questo nelle reti logiche si descrivono le forme d'onda reali con anche la descrizione del tempo di ritardo. Durante la fase transitoria le reti logiche possono dare uscite non desiderate.

- **Glitch:** *un impulso indesiderato dell'uscita in fase transitoria* → *se una rete combinatoria può potenzialmente presentare un glitch*, allora **si dice che è caratterizzata da un'ALEA.**
- **Alea:** *difetto di una rete combinatoria tale per cui le uscite in transitorio possono avere un valore diverso da quello che possiedono a regime.* In alcuni casi sono eliminabili con una sintesi opportuna, in altri casi non sono eliminabili.

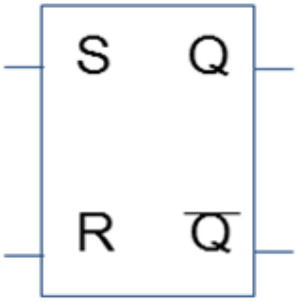
  1) **ALEE DINAMICHE:** *se in seguito ad un cambiamento degli ingressi, l'uscita prima di cambiare valore definitivamente a regime, cambia più volte durante il transitorio.*
  2) **ALEE STATICHE:** *se in seguito ad un cambiamento di ingresso, l'uscita nel transitorio ha cambiamento di valore mentre avrebbe dovuto rimanere costante.*

ELEMENTI BASE RETI SEQUENZIALI:

BISTABILI: *elementi capaci di mantenere al loro interno il valore 0 o il valore 1. Esistono diversi* ***tipi di bistabili****:*

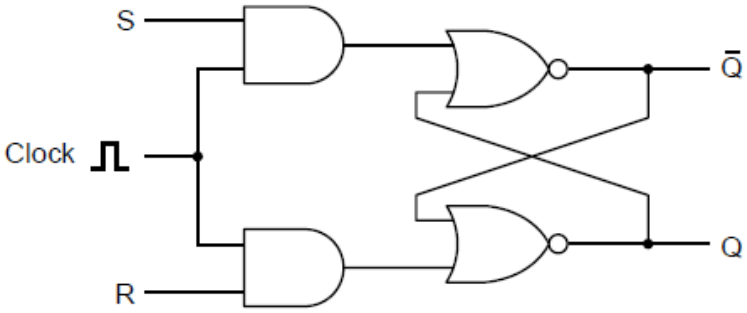
1) ***SET-RESET di tipo ASINCRONO E SINCRONO***
2) ***LATCH:*** bistabile sincrono trasparente, capace di memorizzare o meno segnali di ingresso in funzione di un segnale di abilitazione (enable). La transizione di stato avviene per tutto il tempo in cui il clock è attivo (alto). Quando l'enable non è attivo, il latch mantiene in uscita l'ultima configurazione presente sugli ingressi.
3) ***FLIP-FLOP:*** un dispositivo bistabile privo della proprietà di trasparenza. Nel flip-flop il cambiamento della uscita non è conseguenza del cambiamento dell'ingresso di dato, ma è conseguenza del cambiamento di un ingresso di controllo sincrono (il clock) o asincrono (preset o clear) → ***si definisce flip-flop un bistabile sincrono a commutazione sul fronte perché:*** *la transizione di stato avviene nell'istante in cui si ha l'evento significativo del clock (fronte di salita o di discesa) in base agli ingressi in quel momento.*

SET-RESET ASINCRONO: *elemento più semplice capace di memorizzare all'interno il valore di una variabile di stato binaria e di commutare alla presenza di un'opportuna configurazione di ingresso.*



Funzione di eccitazione : $Q(t+1)=S+R'Q(t)$

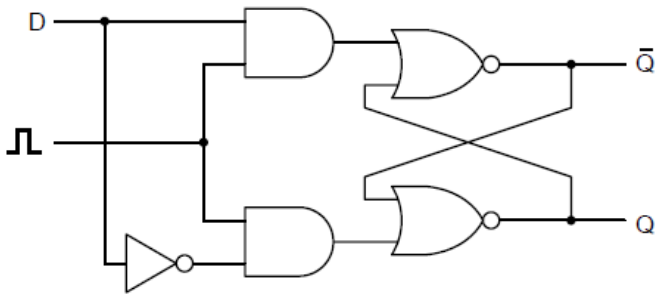
SET-RESET SINCRONO → LATCH: *memorizzo quello che c'è agli ingressi nel periodo quando EN=1*



- **Se il segnale di clock vale 0 (non è attivo):** *non mi interessa cosa valgano S, R → non c'è cambiamento di stato:* **stato di hold.**
- **Se il segnale di clock vale 1:** *la rete è sensibile al cambiamento degli ingressi CHE:*

  a) Se sono **entrambi 0:** la rete **rimane in hold**
  b) Se sono **entrambi 1:** si ha una **configurazione vietata**
  c) **Altrimenti cambia stato**

D-LATCH: *è una memoria capace di* ***mantenere l'uscita se il segnale di clock NON è attivo*** *e di cambiare l'uscita campionando l'ingresso quando il clock è attivo*



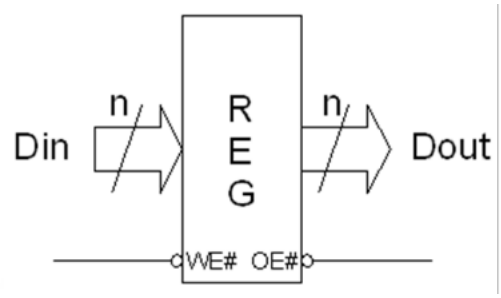
- **Tempo di set-up:** *periodo in cui gli ingressi devono rimanere stabili* ***prima del fronte del clock*** *per poter essere campionati correttamente*
- **Tempo di hold:** *periodo in cui gli ingressi devono rimanere* ***stabili dopo l'evento del clock***

METASTABILITA':
effetto per cui l'uscita rimane indefinitivamente in uno stato instabile tra 0 e 1 → serve per fare sì che le reti non cambino a caso

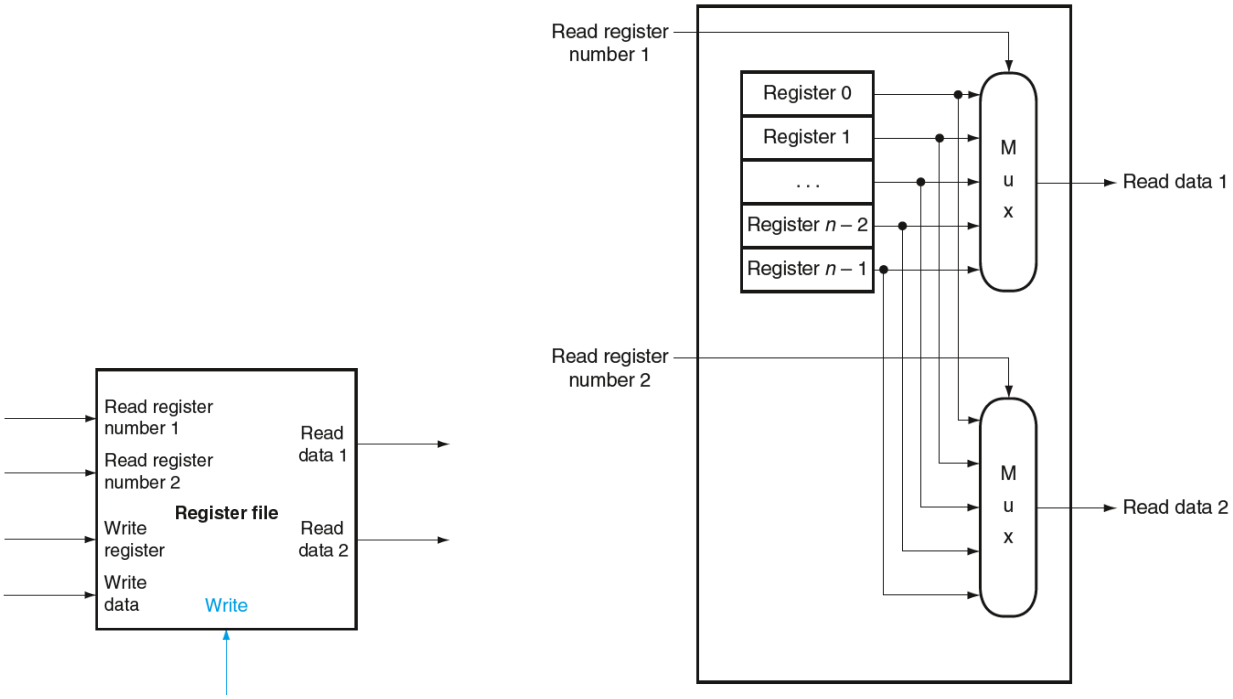
I REGISTRI:

I FF e i Latch D sono molto usati e se impiegati **in parallelo** (uno per ogni bit) **possono registrale parole di *n* bit** → formano i registri nella forma più semplice:

- Si sono *n FF* con segnale di WE#
- Segnale di OU# che da l'uscita collegato con dei tristate così da fornire in uscita tutti o nessuno



Register file (memoria a doppia porta): all'interno di ogni CPU. È un insieme di *registri, normalmente contenenti i dati degli operandi, che possono essere letti e scritti, in base ad un numero* (o un nome come eax, ebx etc) *che viene usato come indirizzo.*



Lettura da registro: scrittura su memoria o voglio fare una ALU. I registri che vengono letti hanno un comportamento analogo a quello di reti logiche combinatorie. Il register file ha 2 uscite ed io seleziono quale registro voglio con MUX (ha 2 segnali di controllo) presente prima delle uscite.

Come si progetta una rete sequenziale?

AUTOMA A STATI FINITI: applicazione → riconoscitore di sequenze: riconoscere se dagli ingressi entrano sequenze esatte (esercizio sul quaderno)

È una quintupla **Ms = {X, Z, S, F, G}** dove:

- **X: insieme simboli di ingresso**
- **Z: insieme simboli di uscita**
- **S: insieme finito degli stati interni**
- **F: funzione di uscita**
- **G: funzione di stato**

Ognuno dei simboli usati dagli alfabeti corrisponde ad una variabile binaria, impiegata per rappresentare il valore assunto dal corrispondente segnale binario.

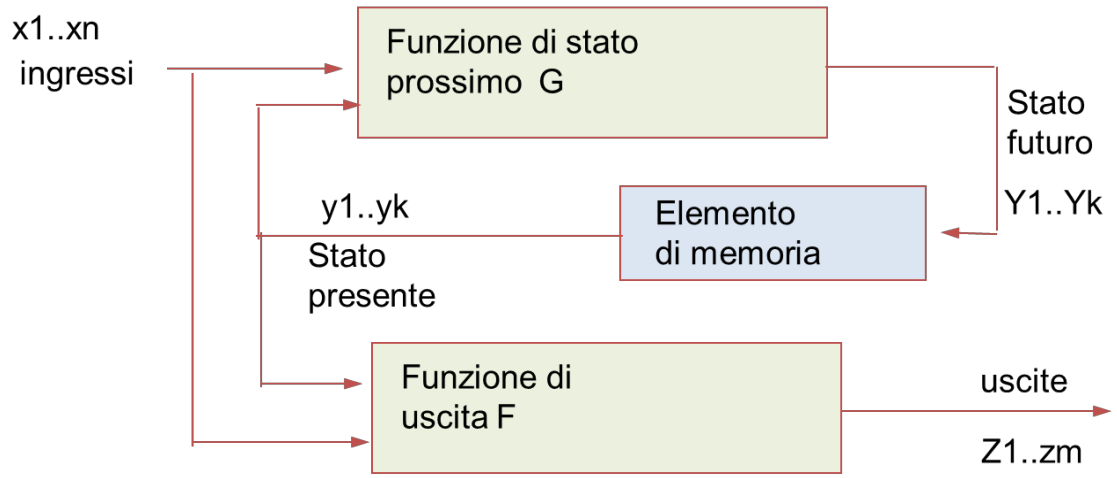
- **X={*x1,...xn*} n variabili di ingresso**
- **Z={z1,...zm} m variabili di uscita**
- **S={s1,..sk} k variabili di stato interno** s1..sk dove k è finito e non vuoto

G:S × X →S è la **funzione di stato prossimo:** G è la funzione combinatoria che ad ogni configurazione di **stato presente e ad ogni configurazione di ingresso associa** la configurazione dello **stato futuro.**

F:S × X →Z è la **funzione di uscita:** F è la funzione combinatoria che determina per ogni stato interno ed ingresso, la configurazione delle uscite.

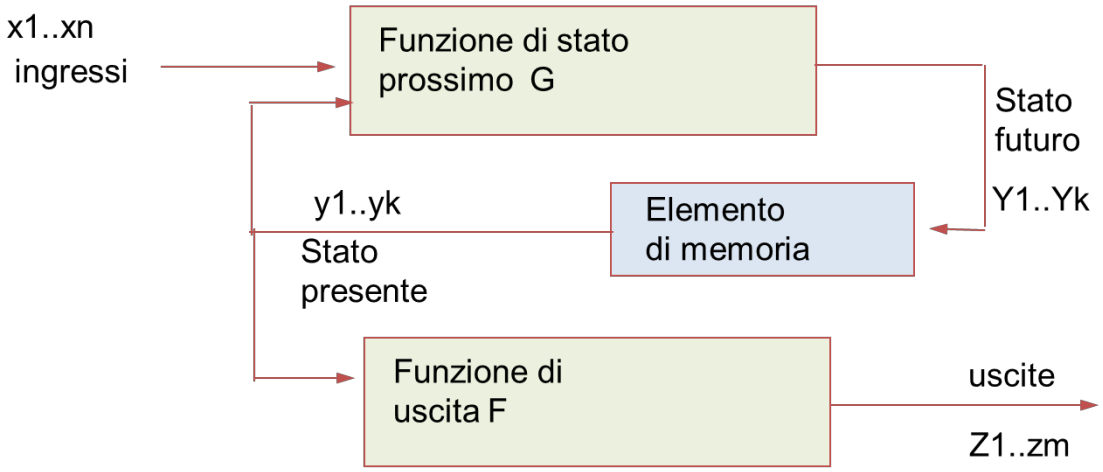
A) AUTOMA DI MEALY:

Il modello più generale è l'**automa di Mealy** in cui *la configurazione delle uscite dipende dallo stato presente e dagli ingressi in quell'istante come indicato dalla funzione F.*



B) AUTOMA DI MOORE:

Modello in cui la configurazione delle uscite dipende SOLO dallo stato presente e non dagli ingressi in quell'istante. F: S→Z è la **funzione di uscita secondo l'automa di Moore.**



SINTESI DI RETI SEQUENZIALI COME AUTOMI A STATI FINITI

Una rete sequenziale non memorizza tutte le configurazioni di ingresso passate ma memorizza una loro elaborazione rappresentata dagli **stati interni** e in particolari dei valori assunti da variabili di stato, nella ipotesi che tali stati siano in numero finito (FSM).

Le variabili di stato sono memorizzate in elementi di retroazione e nelle macchine a stati finiti gli elementi di retroazione sono Flip Flop con un unico segnale di clock.

L'insieme dei FF sincronizzati è detto **registro di stato** o memoria e riceve in ingresso lo stato presente che una volta memorizzato diventa dopo il segnale di sincronismo lo stato futuro.

→ Algoritmo per la sintesi sequenziale:

1) Si prepara una **descrizione comportamentale a parole** o con un linguaggio di descrizione dell'hardware (*specifiche di progetto*).

2) Si definisce il **diagramma degli stati** per definire le transizioni.

3) Si traduce nella **tabella di flusso**: è la fase che corrisponde in software alla creazione dell'algoritmo perchè si definiscono gli stati interni e le transizioni.

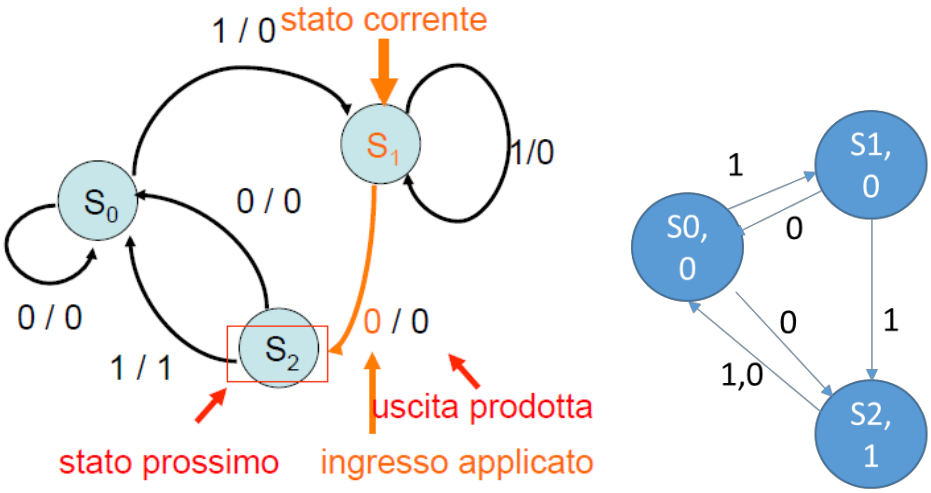
4) Si impiegano metodi manuali o automatici per la **minimizzazione degli stati.**

5) Dalla tabella di flusso che contiene sia gli stati futuri che le uscite si creano due tabelle: **la tabella delle transizioni e la tabella delle uscite** attraverso il processo di codifica o assegnamento degli stati.

6) Si ottiene la sintesi o **implementazione.**

DIAGRAMMA DEGLI STATI: è un grafo G=(V, E) con tanti nodi V quanti gli stati e tanti archi E quante le transazioni da uno stato all'altro dovute a cambiamenti in ingresso. Nel diagramma vengono rappresentate anche le uscite:

- ***Modello Mealy (sx): archi*** perché l'uscita dipende dagli ingressi e dallo stato
- ***Modello Moore (dx): nodi***



Le tabelle delle sintesi:

La tabella di flusso è una tabella (mappa) che memorizza tutte le transizioni di stato. Essa ha:

- ha tante colonne quante le configurazioni di ingresso
- tante righe quanti gli stati presenti e contiene gli stati futuri e le uscite

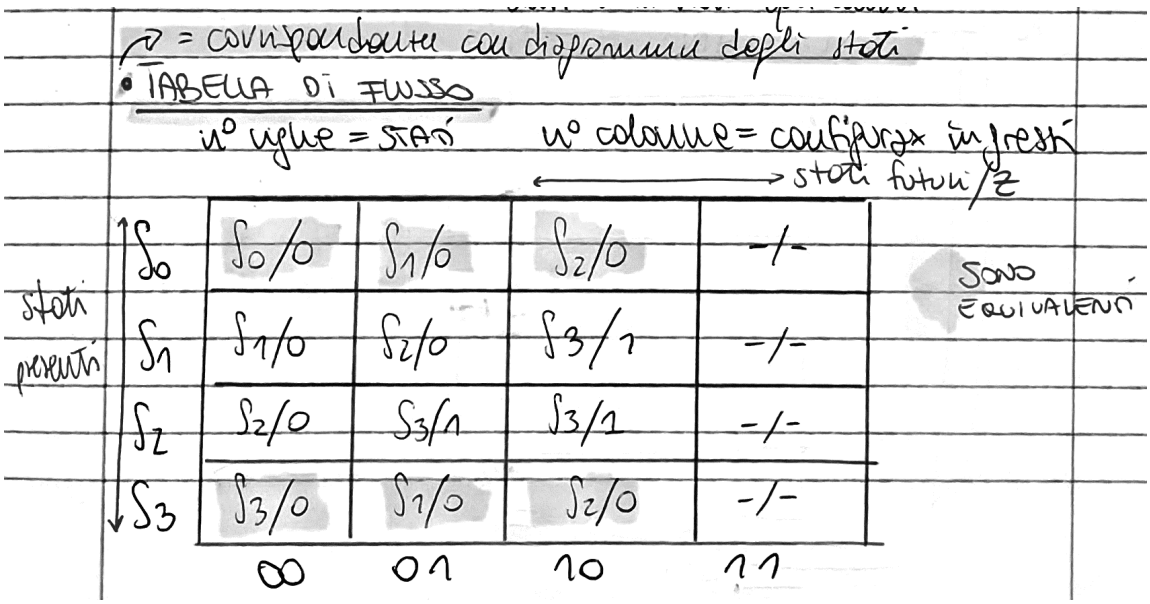
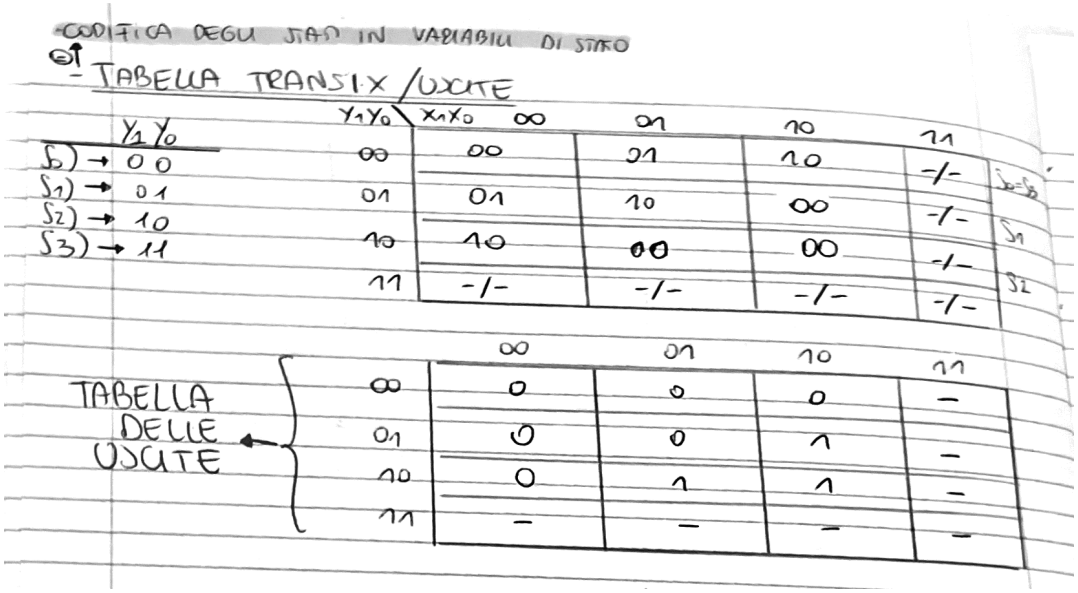


Tabella delle transizioni è la tabella di flusso in cui gli ingressi sono rappresentati da variabili binare e gli stati sono rappresentati in codifica binaria attraverso le **variabili di stato.**

Tabella delle uscite è la tabella che codifica la o le uscite in dipendenza dagli stati presenti, codificati con le variabili di stato e, in caso di modello di Mealy anche degli ingressi.



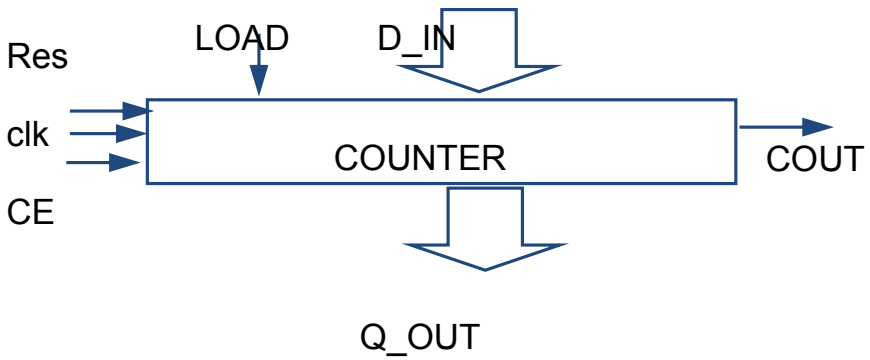
PROCEDURA PER SEGUIRE LA SINTESI:

Dal **diagramma** si passa con corrispondenza 1:1 alla **tabella di flusso** di Mealy o di Moore a seconda del tipo di diagramma degli stati.

Si impiegano se servono algoritmi di **minimizzazione** degli stati (che NON vengono trattati in questo corso); si esegue l'assegnamento o **codifica** degli stati e si passa alla **tabella di transizione** e delle uscite; si **realizza** lo schema finale scegliendo i gate elementari e i FF da usare come registro di stato. In base alla funzione di eccitazione dei FF cambia la rete logica combinatoria di creazione degli ingressi dei FF come stato futuro.

Attenzione: ***la codifica è una fase importante che mette in corrispondenza gli stati con le variabili di stato ordinate***, che possono essere ordinate con ordine sequenziale ma anche seguendo le adiacenze come nelle mappe di Karnaough. Esistono algoritmi specifici per trovare la miglior codifica. ***Qui si impiegherà il semplice ordine sequenziale.***

I CONTATORI:



I contatori ***sono reti sequenziali*** molto comuni che si ***usano nei calcolatori (***timer, nelle CPU, nelle parti di controllo della memoria). I contatori ***possono essere all'avanti o all'indietro e hanno di solito una sola uscita che vale 1 quando l'automa transita nello stato 0.*** Di solito ci sono anche uscite aggiuntive che indicano la configurazione dello stato interno e quindi indicano lo stato del conteggio. Possono non avere ingressi, e contano indefinitamente, o possono avere un ingresso di Reset, per ricominciare da zero.

Il contatore in avanti: ***è una rete sequenziale sincrona che ad ogni clock incrementa il suo valore,*** all'indietro viceversa.

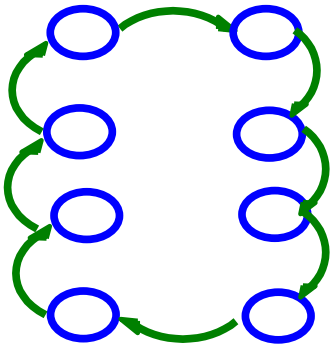


Fig. 26: Contatore per 8 all'avanti.

In generale i **contatori** ***sono automi il cui stato interno è destinato ad evolversi seguendo ordinatamente e ciclicamente una determinata sequenza di valori***. Se la rete contiene *n* stati si dice che il contatore conta per *n*. Un contatore ha:

- una base di conteggio (per gli n stati del ciclo)
- una codifica degli stati

Contatore binario: codifica binaria e una base potenza del 2

I contatori si usano anche come blocchi logici per creare altre funzioni in modo RTL. Il blocco generico è composto oltre che dall'ingresso di clock da

- 1 ingresso CE (clock enable) che se 1 abilita il conteggio
- 1 ingresso Res che se 1 riporta allo stato 0
- 1 ingresso di Load che se 1 fa ripartire dallo stato specificato come D_In
- Dalle uscite COUNT (1 se passa per lo 0) e dalla uscita Q_Out che indica lo stato interno.