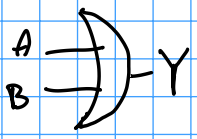
Porte Logiche

- AND

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

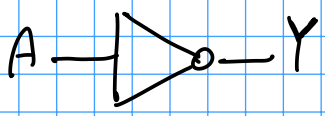
- OR

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1



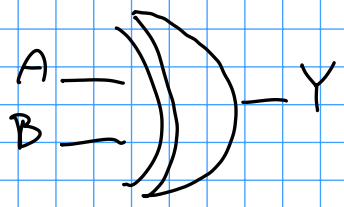
- Not

A	Y
0	1
1	0



- XOR

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0



Logica combinatoria

Requisiti

- Uno o più input
- Uno o più output
- Specifiche funzionali che descrivono la relazione tra input ed output

- Specifiche temporali che descrivono il delay tra il cambiamento degli input e l'adattamento degli output

I circuiti digitali sono classificati come

- Combinatori
- Sequenziali

Nei circuiti combinatori l'output dipende solo dal valore corrente degli input

In quelli sequenziali, invece, il valore dell'output dipende dal valore corrente e da quello precedente degli input.

I circuiti combinatori non hanno memoria, a differenza di quelli sequenziali.

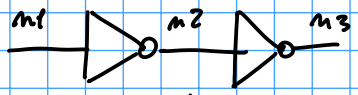
Regole della logica combinatoria

- Ogni elemento del circuito deve essere combinatorio

- Ogni nodo può essere connesso soltanto come input o ad un solo output di un elemento del circuito
- Il circuito non può contenere percorsi ciclici

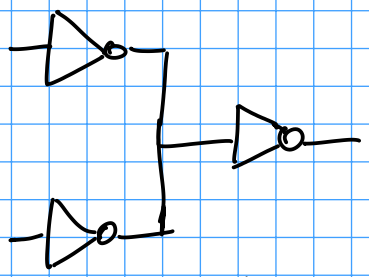
Esempio:

a)



combinatorio

b)



Non combinatorio

Logica Booleana

- Complemento di $A = \bar{A}$ ($A=0 \Rightarrow \bar{A}=1$)
- Variabili e complementi sono detti letterali
- L'AND è il prodotto, o implicante, tra letterali.
- Un mintermine è un AND contenente tutti gli input della funzione.
- L'OR è la somma tra letterali.

- Un maxtermine è un OR contenente tutti gli input della funzione.

Nella espressioni Booleane l'ordine è: Not, And e Or.

Es:

$Y = A + BC \Rightarrow Y = A \text{ OR } (B \text{ AND } C)$

Forma Sop (Sum of products)

Una tabella di verità contiene 2^N righe dove N = # input.

A	B	Y	mintermine
0	0	0	$\bar{A}\bar{B}$
0	1	1	$\bar{A}B$
1	0	0	$A\bar{B}$
1	1	1	AB

Nella forma Sop prendiamo solo i mintermini di valore 1. Quindi

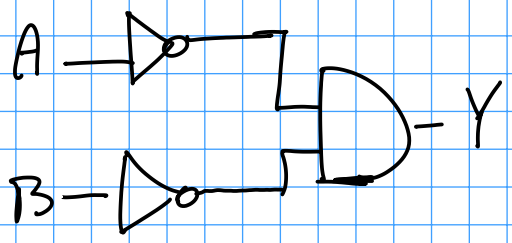
$Y = \bar{A}B + AB$

Esempio:

Costruire un circuito dato la seguente tavola

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

$$Y = \bar{A}\bar{B}$$



Forma POS (product of sums)

Funziona in maniera opposta alla forma SOP.
Si prendono dunque i maxtermini uguali a 0.
(In questa forma gli "1" sono negati)

A	B	Y
0	0	0
0	1	1
1	0	0
1	1	1

$$Y = (A+B)(\bar{A}+B)$$

Algebra Booleana

Utile per semplificare ulteriormente le forme precedenti (se possibile)

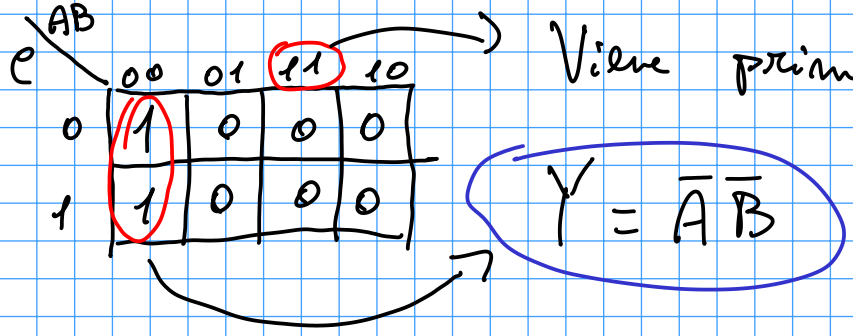
Guardare sul quaderno perché non ho voglia di

Scrivere

Mappe di Karnaugh

Sono un metodo grafico e più sicuro per minimizzare

Funzionano da tre variabili in su.



Viene prima di 10 !

$Y = \overline{A}\,\overline{B}$

A	B	C	Y
0	0	0	1 ←
0	0	1	1 ←
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

- I mintermini si riportano nella relativa tabella

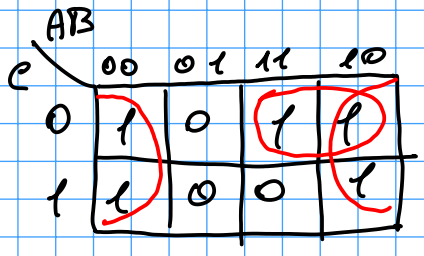
- Si cerchiano i mintermini adiacenti quando sono due o sui multipli, tenendo presente che le K-map sono circolari, e quindi che angoli e lati sono collegati

- Le variabili, in un cerchio (o ricoprimento) presenti sia come 1 che 0 vanno eliminate

Affinché la minimizzazione sia il più preciso possibile, bisogna fare cerchi il più grande possibile.

I mintermini possono anche essere cerchiati più volte

Es:



Le K-map utilizzano un sistema di numerazione detto Greg Code (11 prima di 10) così che gli elementi adiacenti differiscono per una sola variabile.

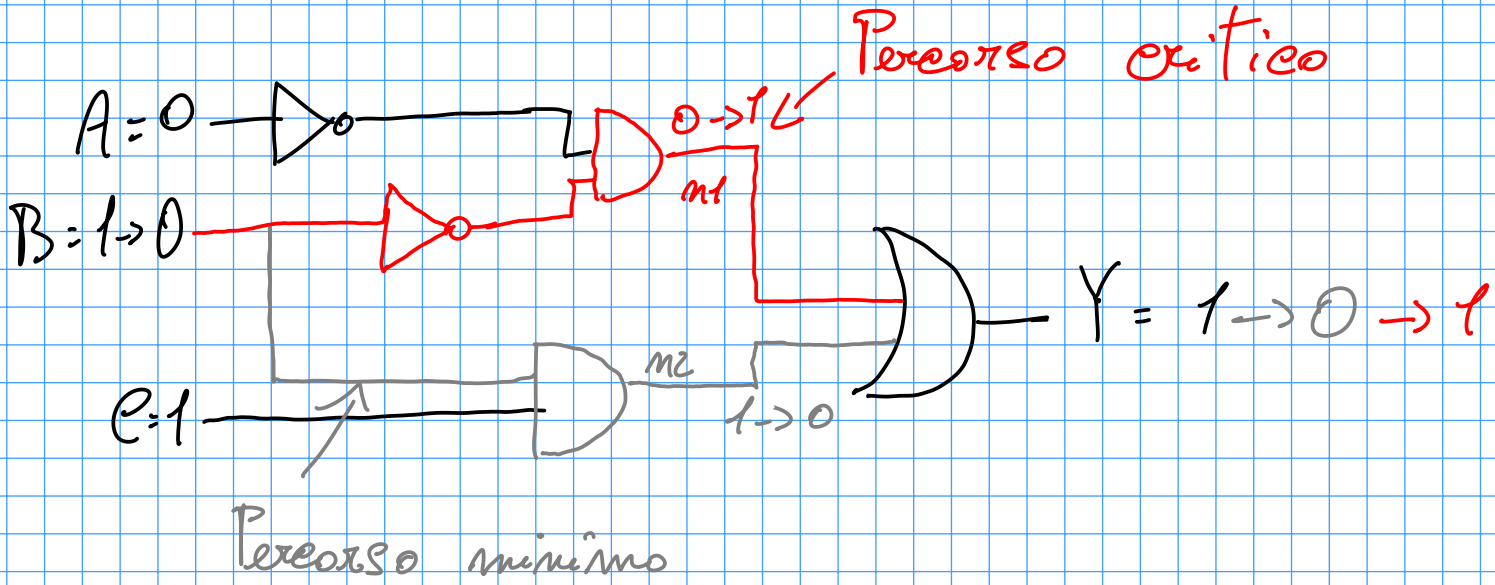
Don't Care

Sono valori che non hanno effetto sull'output, questi vengono indicati con una X e possono valere, a scelta, 1 oppure 0. Ovviamente se è possibile creare cerchi più grossi utilizzando i don't care come "1", questo andrà a favore della minimizzazione e del progettista.

Problema delle K-map

Si verifica quando un singolo cambiamento in entrata causa molteplici cambiamenti in uscita.

In un circuito come:



Quando B cambia da 0 a 1, attraverso (ovviamente) in maniera più rapida il percorso minimo, quindi per un certo lasso di tempo, ovvero finché non cambia l'output della porta AND sul percorso critico, sulla porta OR arriveranno 0 e 0, quindi Y sarà 0 fin quando non passeranno tutti i ritardi di propagazione. Questo problema è risolvibile

aggiungendo una porta al percorso minimo (e quindi un delay voluto).

Leggi di De Morgan

- Il complemento dell'AND tra due letterali A e B è uguale all'OR tra il complemento di A ed il complemento di B.

Se Alcisio rompe le palle sostituisce AND = "prodotto" OR = "somma"

$$\overline{AB} = \bar{A} + \bar{B}$$

- Il complemento dell'OR tra A e B è uguale all'AND tra il complemento di A e quello di B

$$\overline{A+B} = \bar{A}\,\bar{B}$$

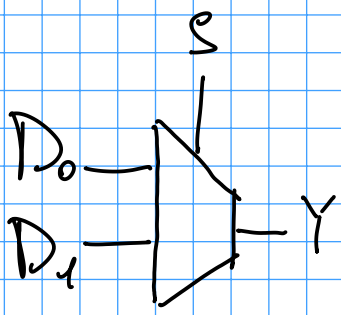
Per dimostrarli basta fare le tavole di verità e far vedere che

- $\overline{A+B} = \bar{A} \cdot \bar{B}$ ^ - $\overline{AB} = \bar{A} + \bar{B}$

Blocchi logici combinatori

- Multiplexer

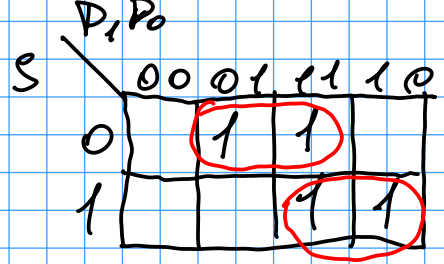
Sono i blocchi più usati. Selezionano un output da diversi input basandosi sul valore di selezione (o selettore)



S	D_1	D_0	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Quando
$S=0$, $Y=D_0$
$S=1$, $Y=D_1$

Possiamo scrivere la K-map

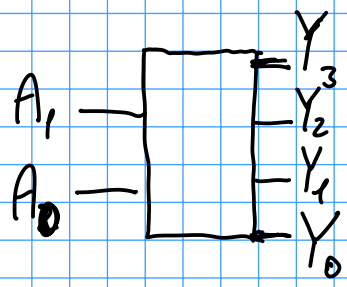


$$Y = \overline{S}D_0 + SD_1$$

Possiamo costruire mux più grossi (4:1, 16:1...) ma avremo bisogno di più selettori, in particolare di $\log_2 N$ dove $N = \#$ input

- Decoder

  Ha N input e 2^N output. Gli output sono tuttavia detti "one-hot" poiché uno solo di essi può valere 1



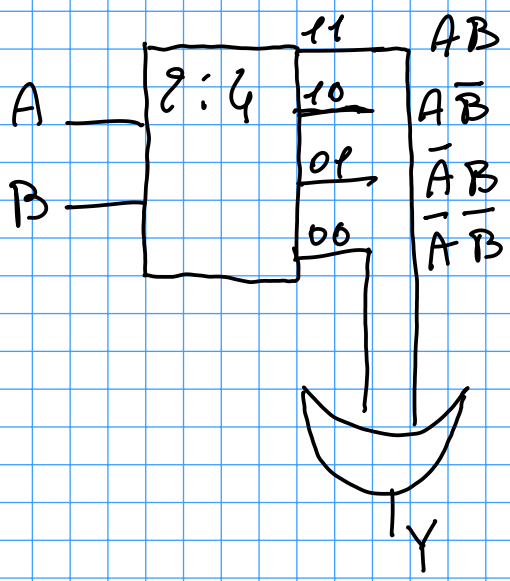
A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

Esempio di tabella di un decoder (uno solo Y a 1 per volta)

I decoder possono essere combinati con delle porte OR per creare funzioni logiche (SOP)

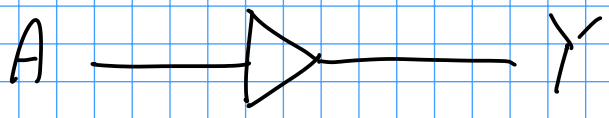
Ad esempio: $Y = \overline{A \oplus B}$ →

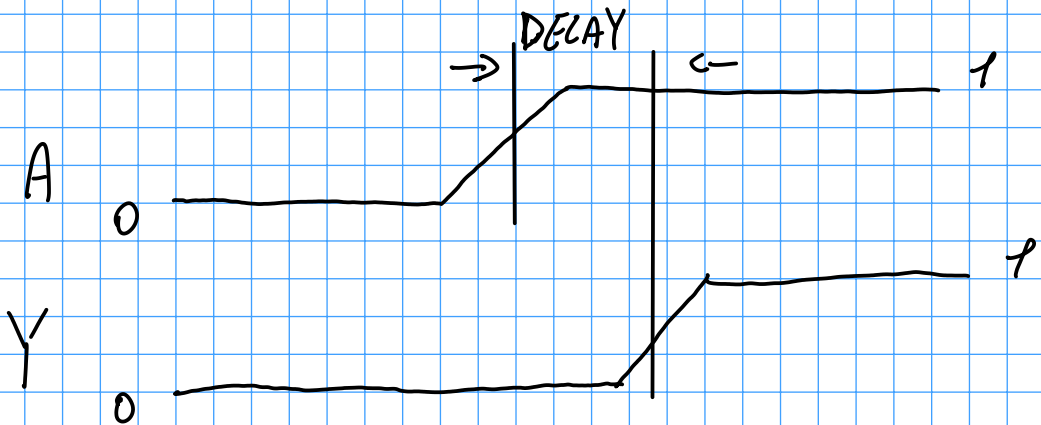
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1



Timing

Un output richiede diverso tempo a cambiare con il variare degli input.





Questo è un diagramma di tempo. La transizione da 0 -> 1 è detta fronte di salita.

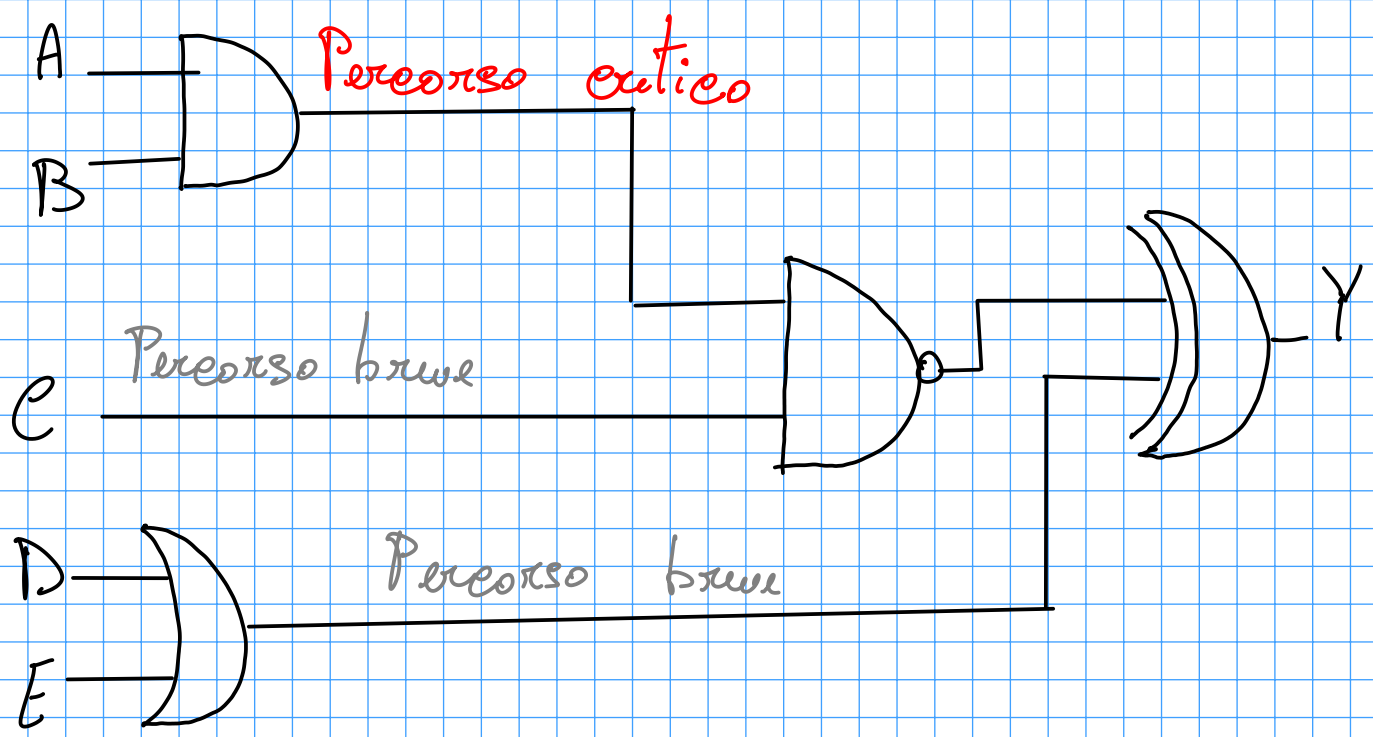
Tempo di propagazione e di contaminazione

La logica combinatoria è caratterizzata dal T_{pd} e dal T_{cd}.

- Il T_{pd} è il tempo massimo necessario affinché l'output cambi al variare dell'input
- Il T_{cd} è il tempo minimo da quando l'input cambia e quando l'output comincia a cambiare il suo valore

Di solito questi sono calcolati osservando il percorso di un segnale, percorso che può essere critico o breve (minimo).

- Percorso critico: Quello più lento in quanto l'input passa per un numero maggiore di porte
- Percorso breve: Tutti i percorsi che non sono critici



In generale:

T_{pd}: Somma dei ritardi di propagazione di ogni elemento del percorso critico

T_{cd}: Somma dei ritardi di contaminazione di ogni elemento del percorso breve

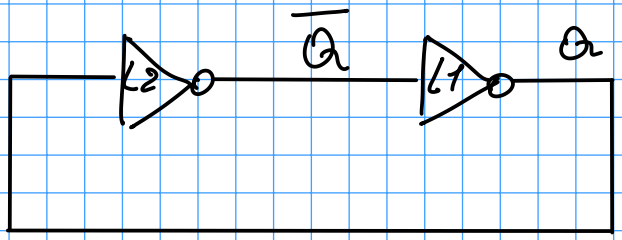
Logica Sequenziale

L'output dipende sia dallo stato attuale degli input che da quello precedente. Parlando di logica sequenziale

si fa riferimento a circuiti che possono memorizzare un bit

• Latches e Flip-flop

Il blocco fondamentale nella costruzione di memorie, è l'elemento bistabile



In questo circuito abbiamo due casi

$Q = 0$

L2 riceve FALSE e produce TRUE su $\overline{Q}$, che diventa ancora FALSE su Q per via di L1.

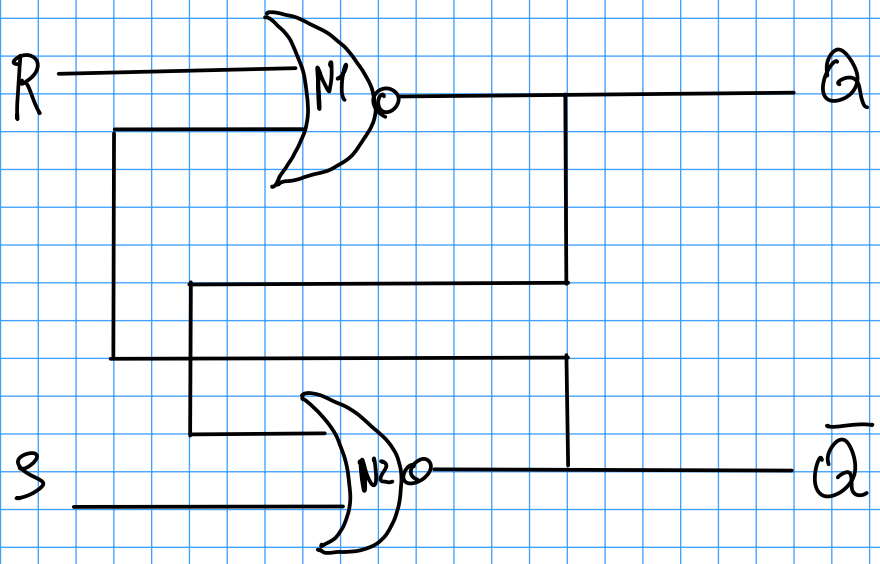
$Q = 1$

Stesso caso ma invertito. Anche in questo caso il valore di Q e $\overline{Q}$ sarà sempre uguale

Avendo 2 casi stabili è possibile memorizzare 1 bit ($\log_2 N \rightarrow N = \#$ casi stabili).

Questo circuito non è una soluzione molto pratica per memorizzare un bit siccome il valore di Q è imprevedibile ad ogni accensione.

Latch SR



Questo latch ha due input (set e reset) che controllano l'output Q.

- R = 1 S = 0

  N2 produce FALSE su $\bar{Q}$

  N1 produce TRUE su Q

- $R = 0 \quad S = 1$

  N2 produce FALSE su $\overline{Q}$

  N1 produce TRUE su Q

- $R = 1 \quad S = 1$

  N1 ed N2 ricevono entrambi almeno un TRUE, producendo FALSE su Q e $\overline{Q}$

- $R = 0 \quad S = 0$

  Indeterminabile, dobbiamo conoscere allora i precedenti valori di Q e $\overline{Q}$

  a) $Q = 0$

  N2 produce TRUE su $\overline{Q}$, N1 riceve $\overline{Q} = 1$ producendo FALSE su Q

  b) $Q = 1$

  N2 produce FALSE su $\overline{Q}$, N1 riceve $\overline{Q} = 0$ e produce TRUE su Q

Osservando meglio è possibile notare che $Q = Q_{prev}$ (idem $\bar{Q}$).

Scriviamo allora la tabella di verità

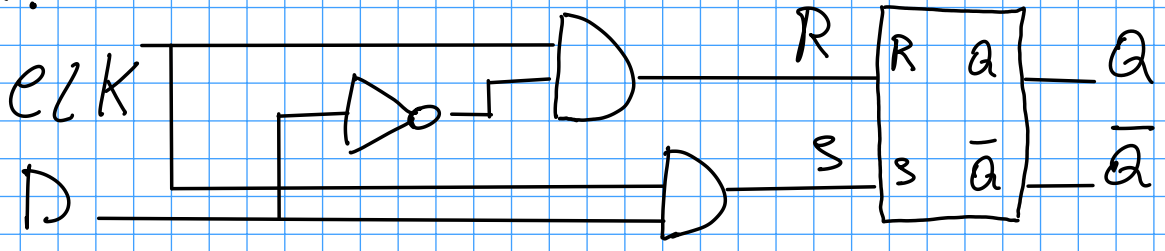
S	R	Q	$\bar{Q}$
0	0	Q_{prev}	$\bar{Q}_{prev}$
0	1	0	1
1	0	1	0
1	1	0	0

Il circuito è bistabile

Latch D

Il latch SR risulta scomodo per il suo comportamento $S = 1 \wedge R = 1$. Inoltre osservare un input determina sia "cosa" che "quando". Meglio disegnare circuiti che separano questi due aspetti.

Il Latch D risolve questo problema introducendo un clock, utile a determinare QUANDO lo stato deve cambiare.



- $CLK = 0$

Il circuito è spento, $Q = Q_{prec}$ e $\overline{Q} = \overline{Q}_{prec}$

- $CLK = 1 \quad D = 0$

Solo 1 porta AND produce 1

- $CLK = 1 \quad D = 1$

L'altro porta AND produce 1

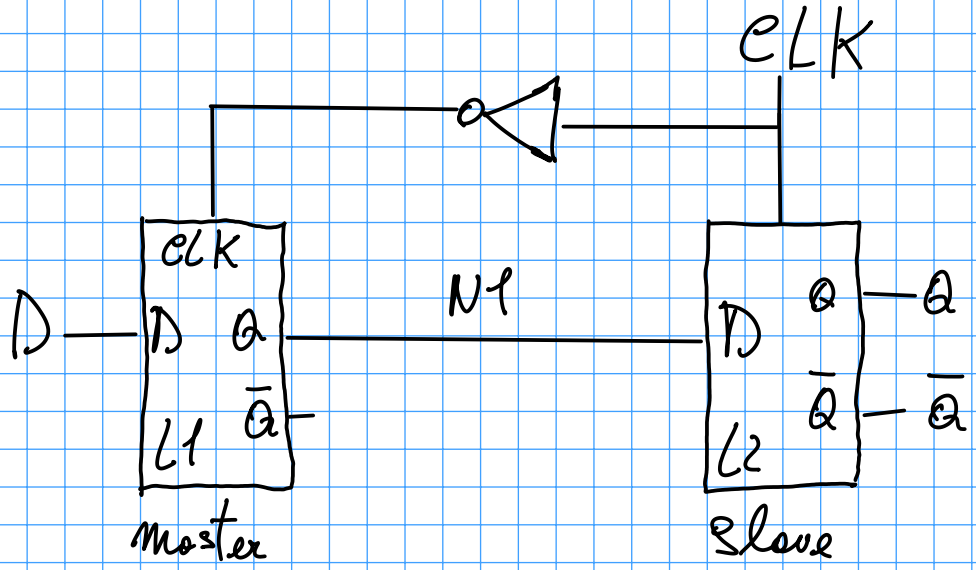
CLK	D	Q	$\overline{Q}$
0	x	Q_{prec}	$\overline{Q}_{prec}$
1	0	0	1
1	1	1	0

$CLK = 1$ il circuito è detto trasparente

$CLK = 0$ il circuito è detto opaco

Flip-Flop D (slave - master)

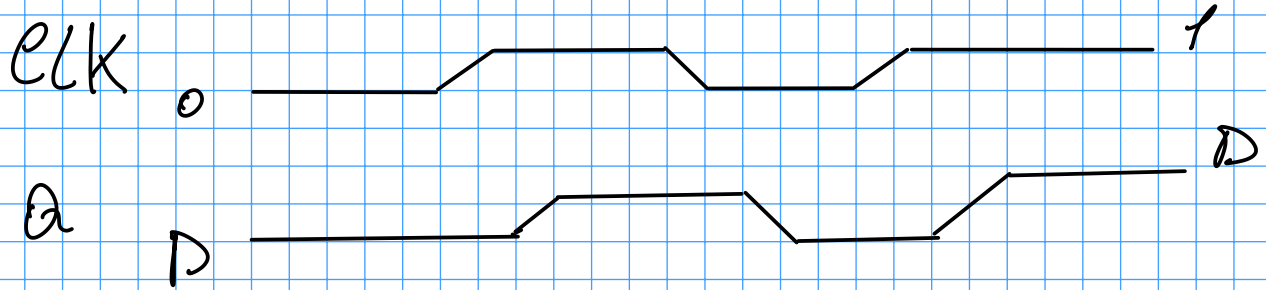
è composto da due latch D controllati da un clock complementare



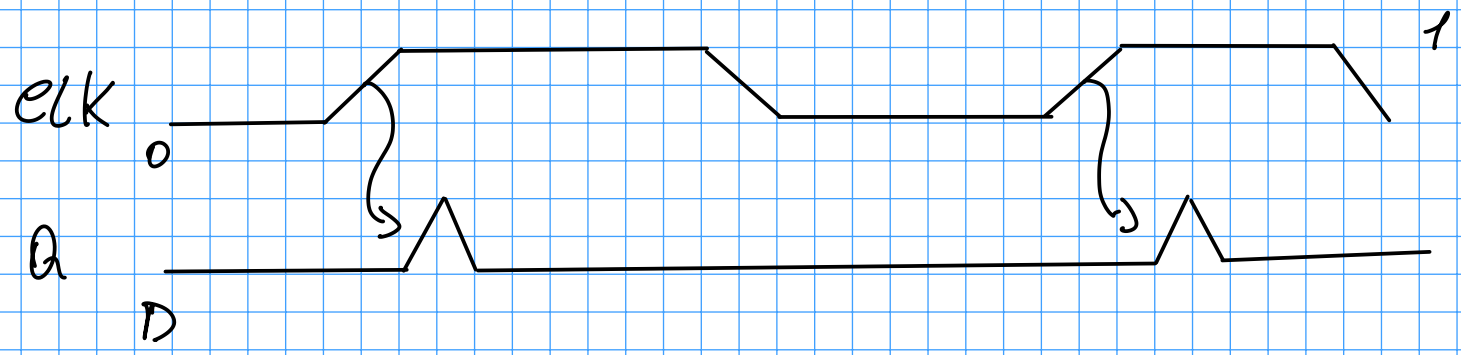
A differenza dei Latch, i flip-flop aggiornano il loro stato solo sul fronte di salita.

Quando CLK = 0, il master è trasparente e lo slave opaco. Qualunque sia il valore di D in N1, quando CLK = 1 lo slave diventa trasparente ed il master opaco.

Diagramma di tempo per Latch D:



Flip-Flop D



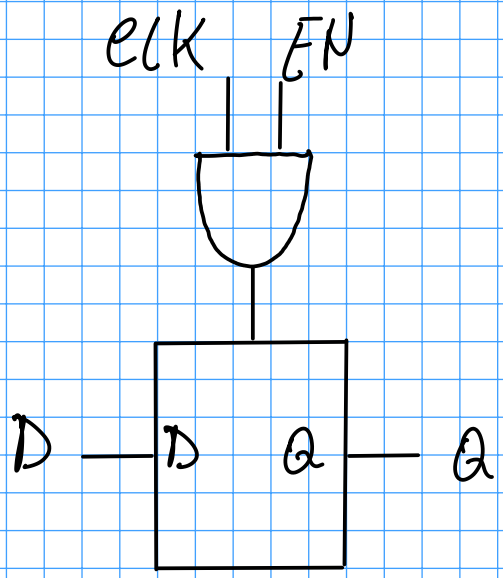
Registri

Un registro a N bit è un banco di N flip-flop che condividono un clk in comune così che tutti i bit vengono aggiornati allo stesso tempo

Flip-flop enabled

Aggiunge un ulteriore controllo per scegliere se aggiornare o no il bit memorizzato.

- En True, funziona come un normale Flip-flop D
- En False, il clock viene ignorato e si conserva lo stato precedente



Flip-flop resettabile

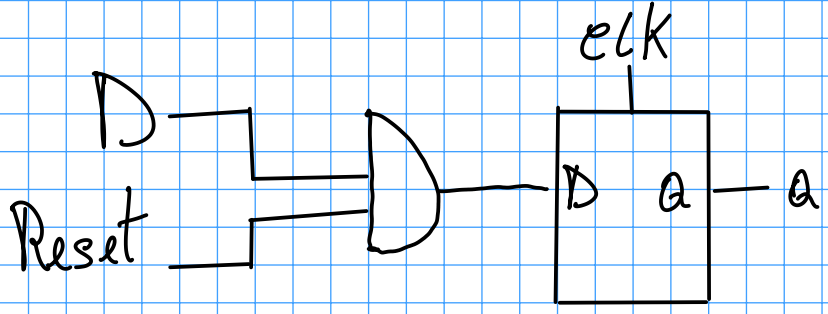
Aggiunge un input detto Reset, quando è FALSE funziona come un normale flip-flop D

Quando Reset è TRUE l'output viene impostato a 0, a prescindere dal resto.

È utile per quando si vogliono impostare a 0 gli output di un sistema da accendere per la prima volta.

Come altri flip-flop può essere <u>sincrono</u> (si resettano solo sul fronte di salita del clock)

oppure <u>osserveremo</u> (si resettano quando reset = 1 a prescindere dal clock).

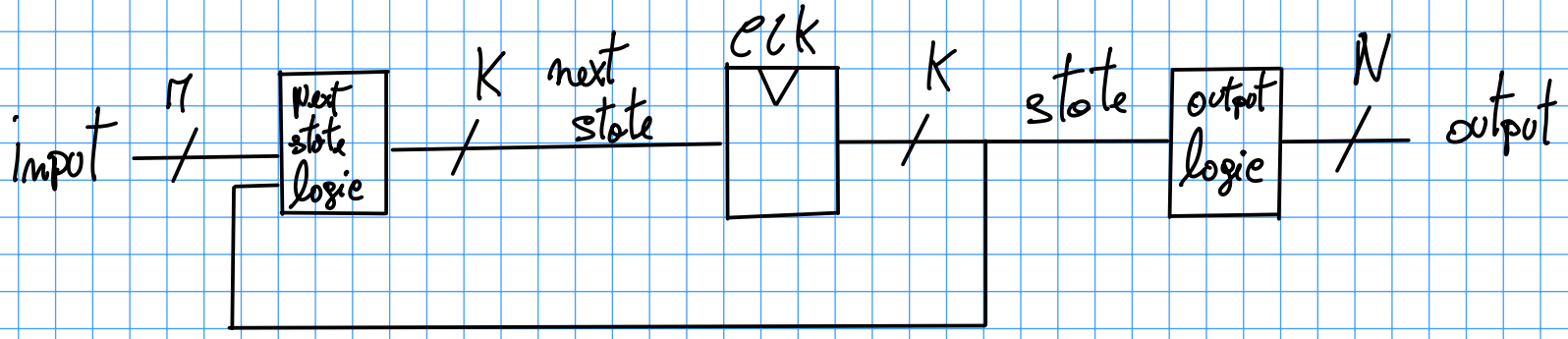


Circuiti sincroni (regole)

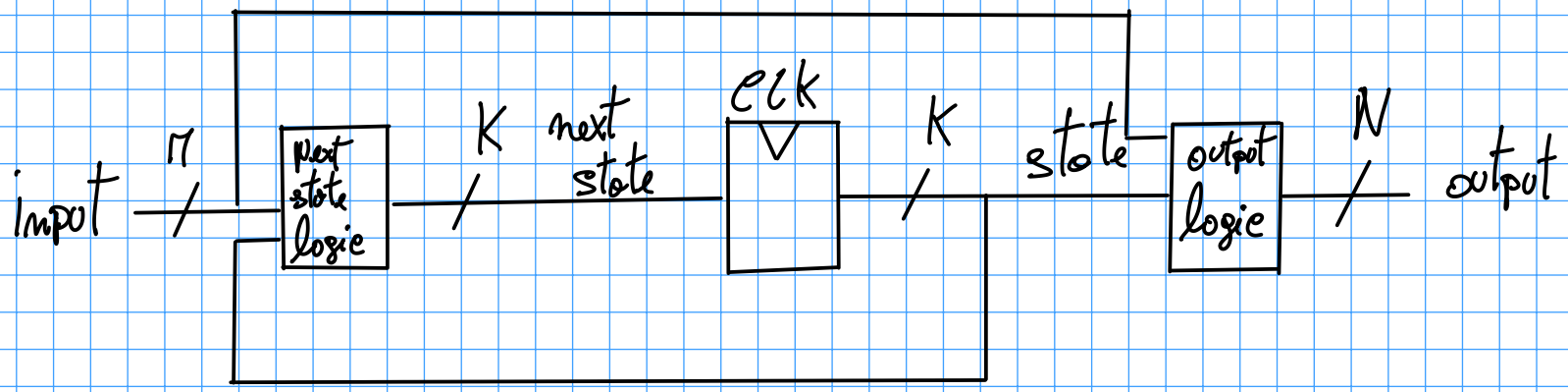
- Ogni elemento del circuito è un registro o un circuito combinatorio
- Almeno un elemento deve essere un registro
- Tutti i registri condividono lo stesso clock
- Ogni percorso ciclico contiene almeno un registro.

Macchine a stati finiti

Macchina di Moore



Macchina alla Mealy



I circuiti sequenziali possono essere disegnati come nelle precedenti figure.
Una macchina (o circuito) con K registri può trovarsi in un numero limitato di stati (2^K).
Una FSF ha M input, N output e K bit di stato oltre ad un clock ed, occasionalmente, un segnale di reset.
Consiste in due blocchi di logica combinatoria, next state logic ed output logic oltre ad un registro che conserva lo stato. Sul fronte di salita del clock le FSF avanzano al prossimo stato

computando sulla base di quello corrente e degli input.

Nella macchina alla Moore l'output dipende solo dallo stato corrente

Nella macchina alla Mealy l'output dipende dallo stato corrente e dagli input correnti.