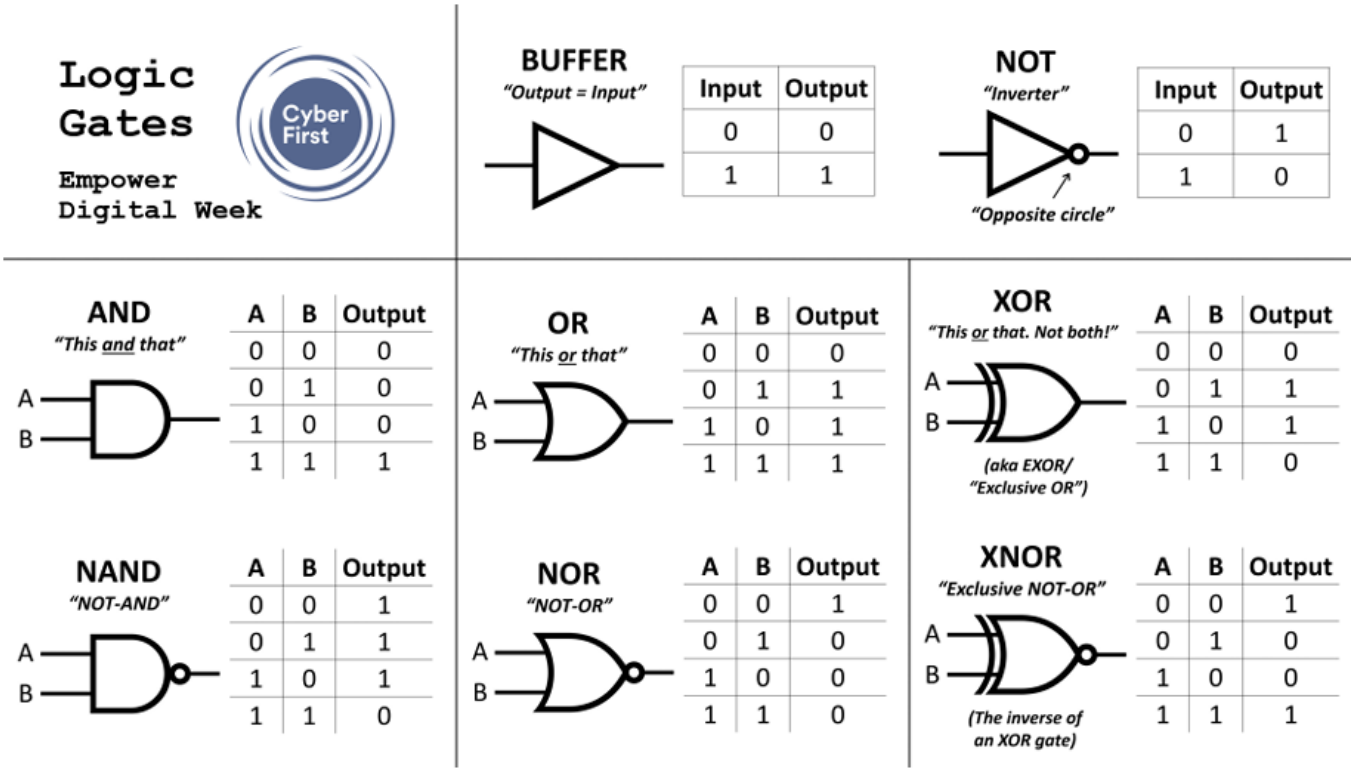
Porte logiche



Circuiti logici notevoli

- OR
- NOT
- AND

Negate:

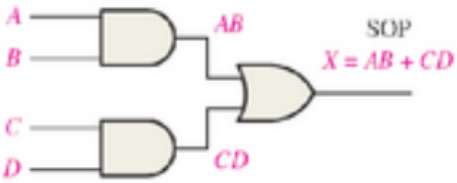
- NAND
- NOR
- NOT

Esclusive:

- XOR
- XNOR

AND-OR

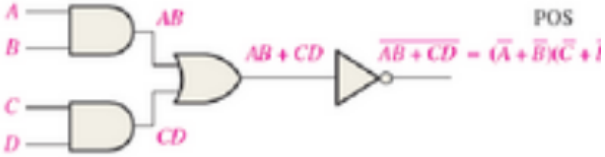
An AND-OR circuit directly implements an SOP expression



AND-OR-INVERT LOGIC

$$X = (\overline{A} + \overline{B})(\overline{C} + \overline{D}) = (\overline{AB})(\overline{CD}) = \overline{(\overline{AB})(\overline{CD})} = \overline{\overline{AB} + \overline{CD}} = \overline{AB + CD}$$

When the output of an AND-OR circuit is complemented (inverted), it results in an AND-OR Invert circuit.

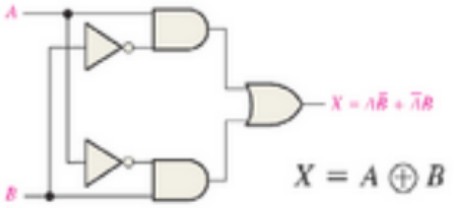


EXCLUSIVE-OR

Although this circuit is considered a type of logic gate with its own unique symbol, it is actually a combination of two AND gates, one OR gate, and two

Truth table for an exclusive-OR.

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

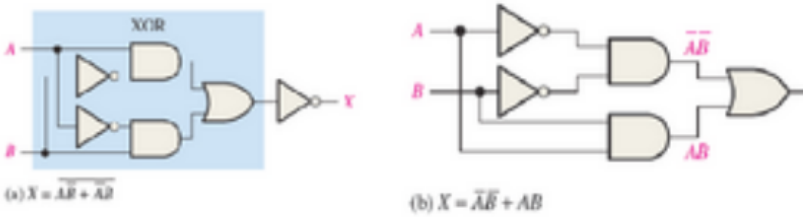


EXCLUSIVE-NOR LOGIC

$$X = \overline{A\overline{B} + \overline{A}B} = \overline{(A\overline{B})}\,\overline{(\overline{A}B)} = (\overline{A} + B)(A + \overline{B}) = \overline{A}\,\overline{B} + AB$$

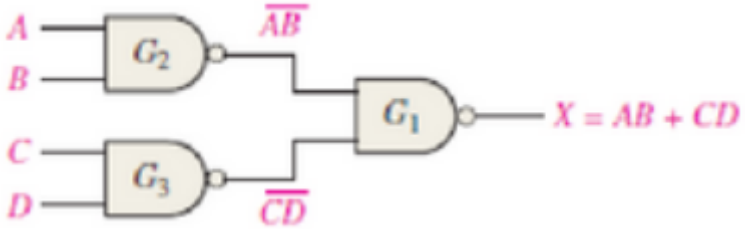
Notice that the output X is HIGH only when the two inputs, A and B, are at the same level.

The exclusive-NOR can be implemented by simply inverting the output of an exclusive-OR, as shown in Figure 5–6(a), or by directly implementing the expression A B + AB, as shown in part (b).



Two equivalent ways of implementing the exclusive-NOR.

NAND



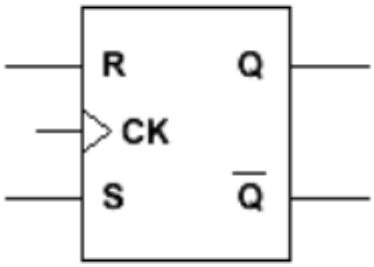
Circuiti logici sequenziali

Riferimento: [Come funziona un flip flop (elettronica digitale) - Andrea Minini]

Flip flop

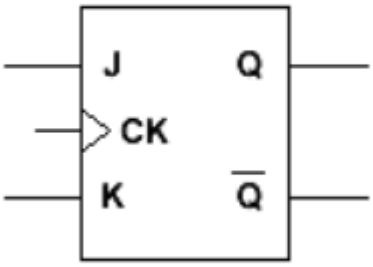
I flip flop sono circuiti sequenziali usati come memoria elementare. Esistono diverse tipologie di flip-flop: SR (Set Reset), JK, T (Toggle), D (Delay)

FLIP FLOP RS



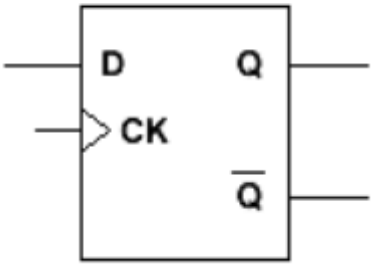
R	S	Q	$\overline{Q}$
0	0	nc	nc
1	0	0	1
0	1	1	0
1	1	–	–

FLIP FLOP JK

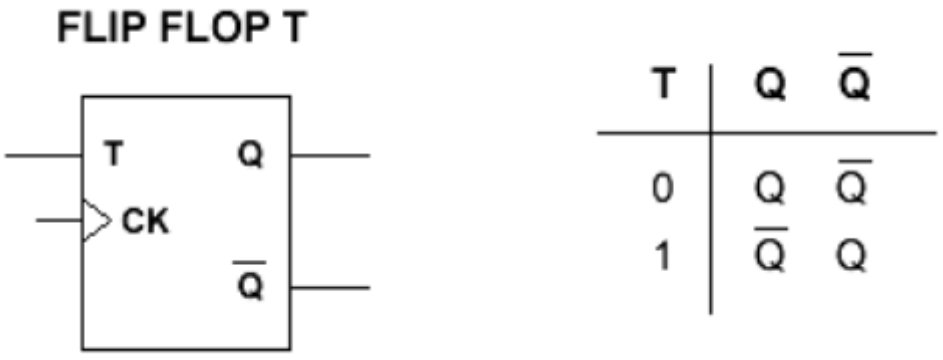


J	K	Q	$\overline{Q}$
0	0	nc	nc
1	0	0	1
0	1	1	0
1	1	$\overline{Q}$	Q

FLIP FLOP D

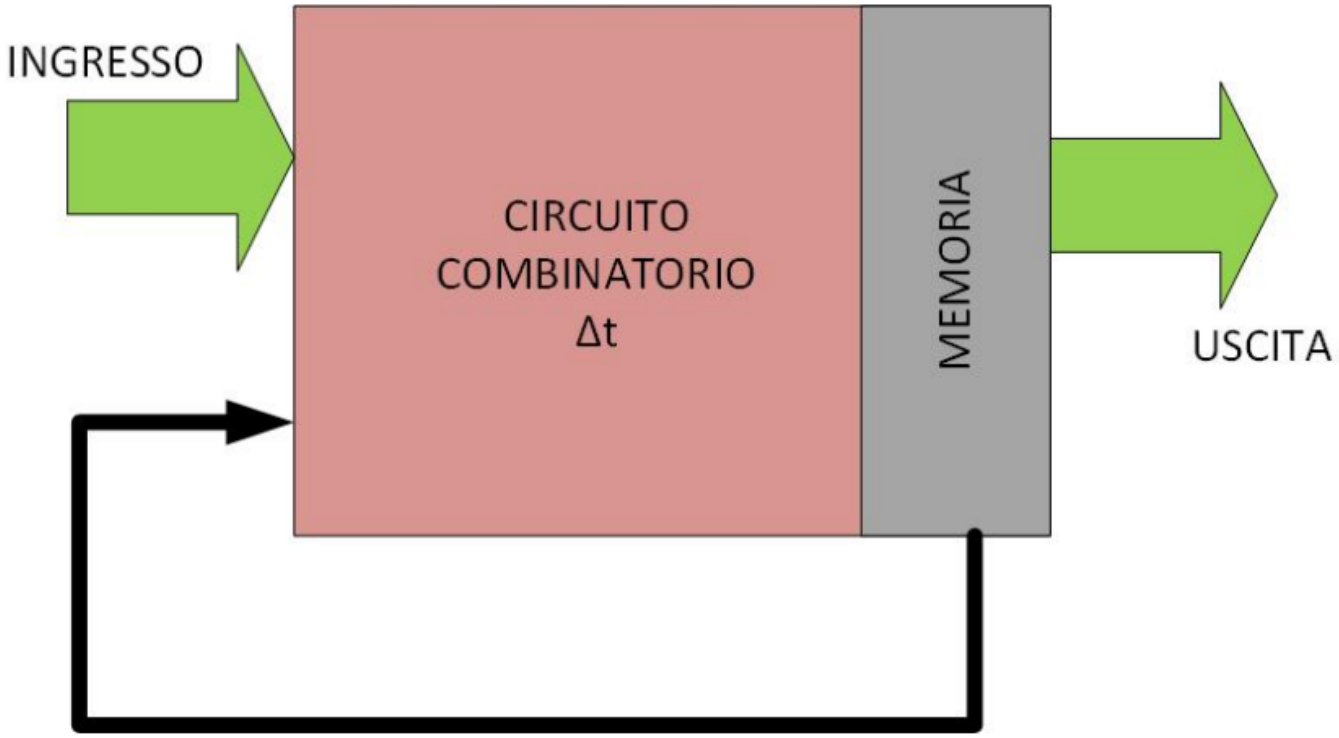


D	Q	$\overline{Q}$
0	0	1
1	1	0



Latch

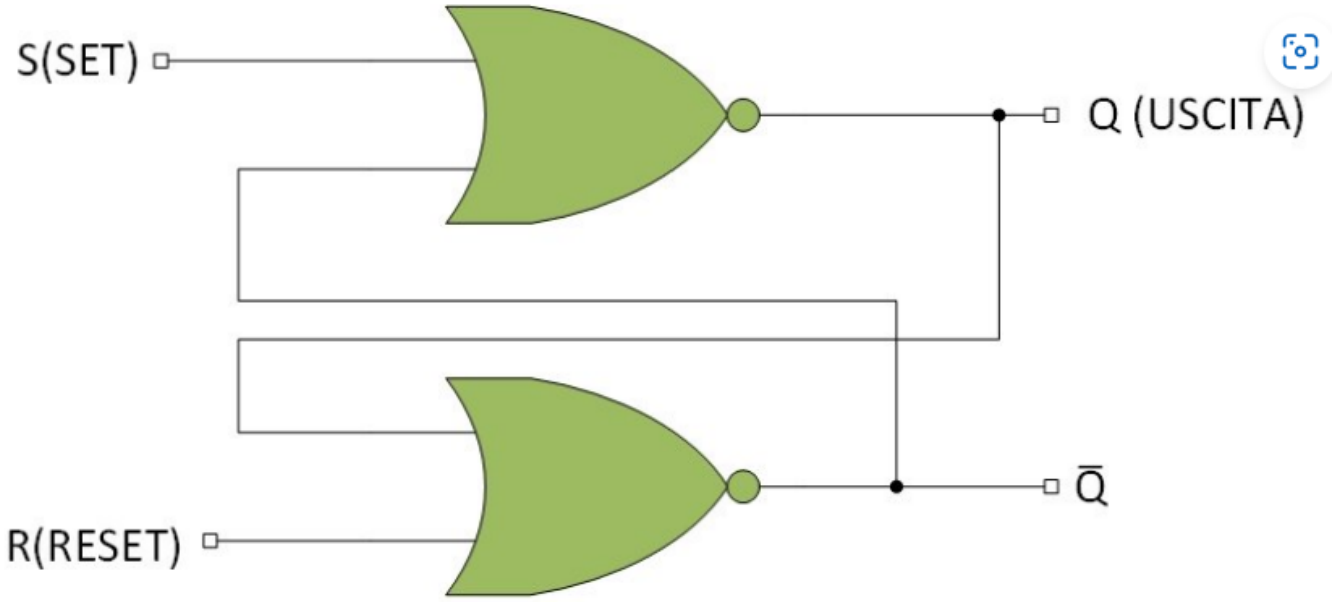
Basate su reti sequenziali:



Latch = Mantiene gli stati

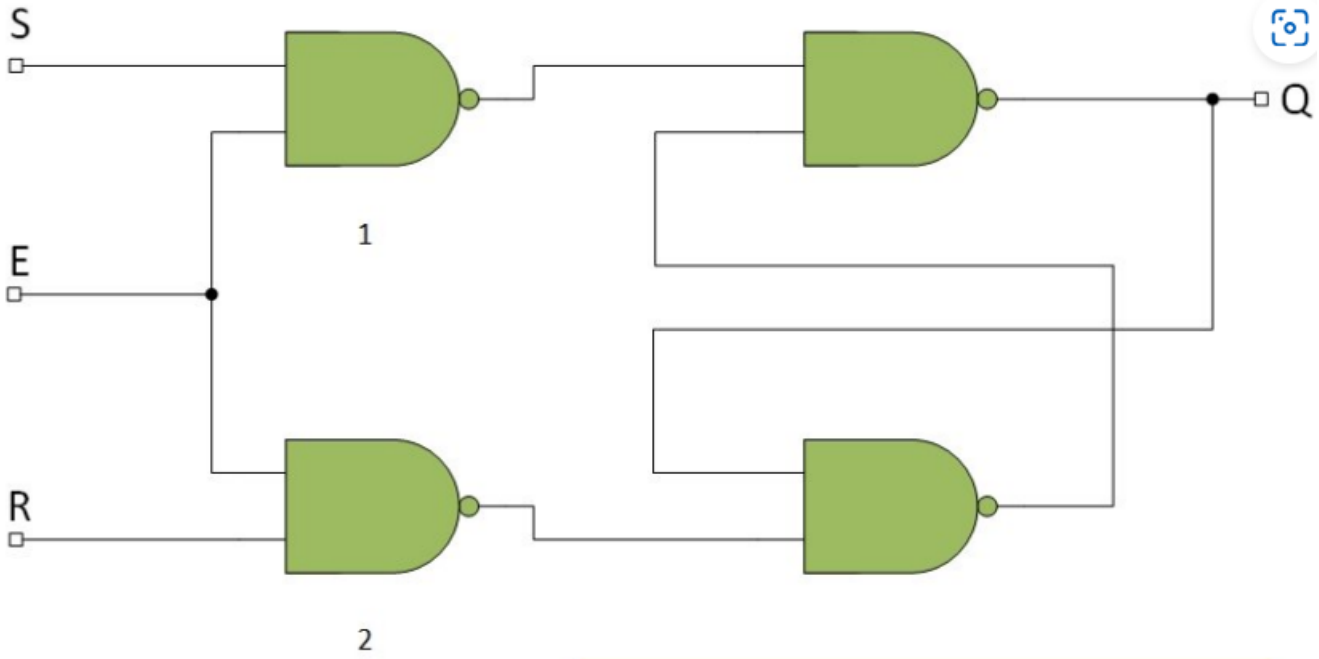
Vari tipi:

- Sequenziale (SR)
  - Flip-flop = Basati su segnale di clock
  - Pulsanti di SET e RESET usando le porte NAND/NOR



S	R	Q
1	0	1
0	1	0
0	0	MEMORIZZAZIONE
1	1	NON AMMESSA

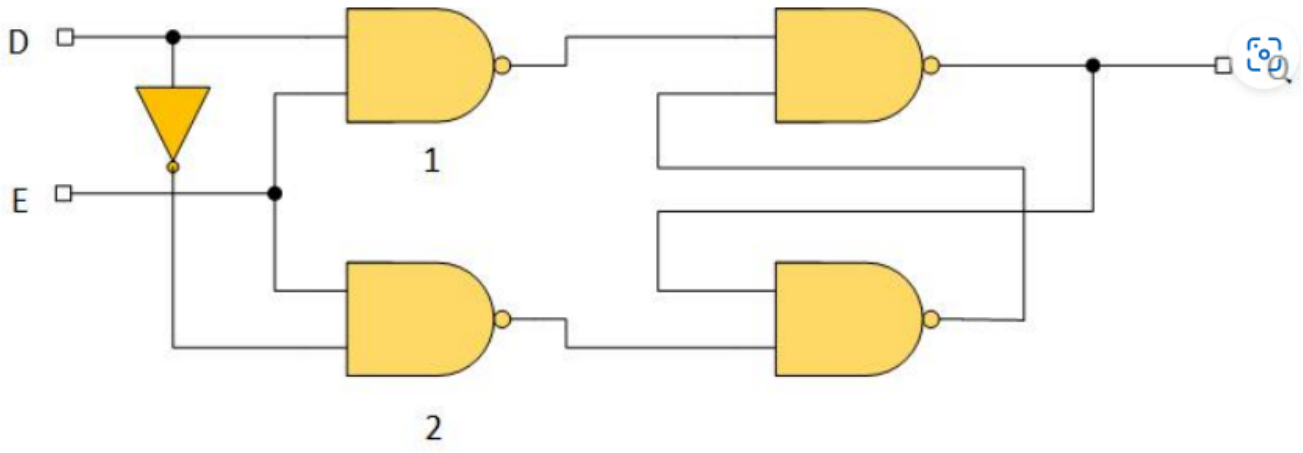
- Latch con enable (NAND)



E	S	R	Q
0	x	x	ULTIMO DATO
1	1	0	1
1	0	1	0
1	0	0	MEMORIA
1	1	1	NON AMMESSO

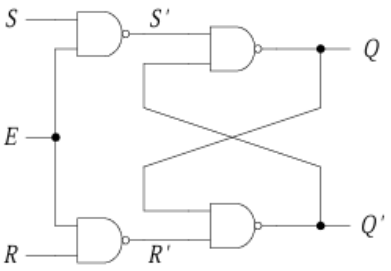
D-Latch = SET/RESET entrambi ad 1

D-LATCH



E	D	Q
0	X	Ultimo Dato
1	1	1
1	0	0

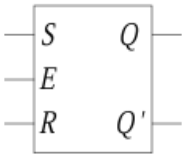
Esempio grafico completo:



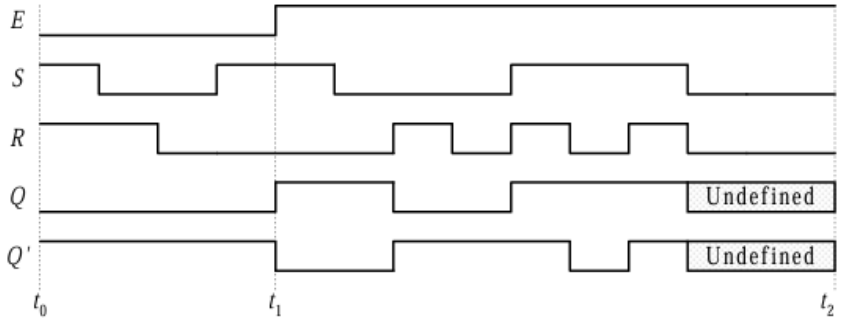
(a)

E	S	R	Q	Q_{next}	Q_{next}'
0	×	×	0	0	1
0	×	×	1	1	0
1	0	0	0	0	1
1	0	0	1	1	0
1	0	1	×	0	1
1	1	0	×	1	0
1	1	1	×	1	1

(b)



(c)



(d)

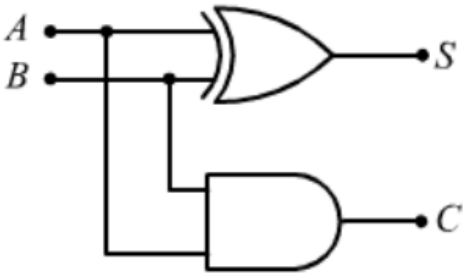
Sommatori binari

Riferimento: 10_Sommatori e sottrattori binari.pdf (unipd.it)

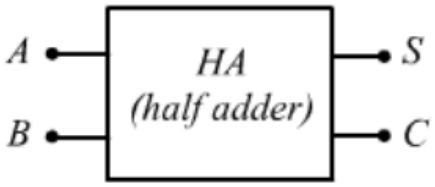
Reti combinatorie che ricevono in ingresso n bit degli addendi da sommare e generano in uscita i bit della somma binaria con il relativo riporto.

Half-adder

Somma con bit di riporto



A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



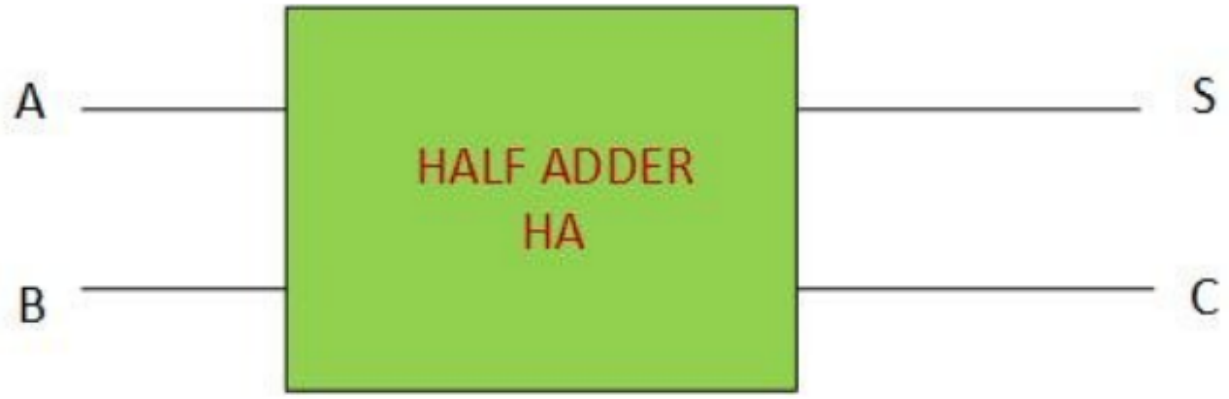
$S = A \oplus B$

$C = A \cdot B$

Dove A e B sono i bit da sommare, S è il bit della somma (sum) e C è il bit del riporto (carry).

Abbiamo così costruito il semisommatore o Half-Adder.

SCHEMA A BLOCCHI DI UN HALF ADDER



$S = A \oplus B$

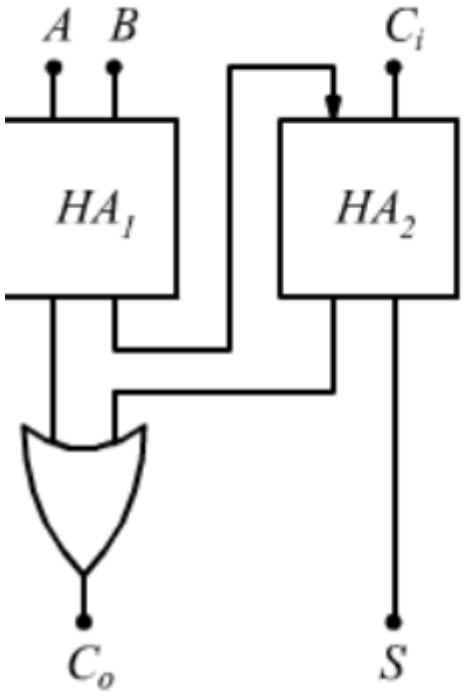
C=AB

S=SOMMA

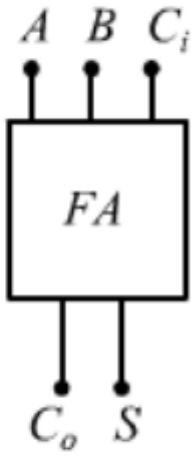
C=CURRY (RIPORTO)

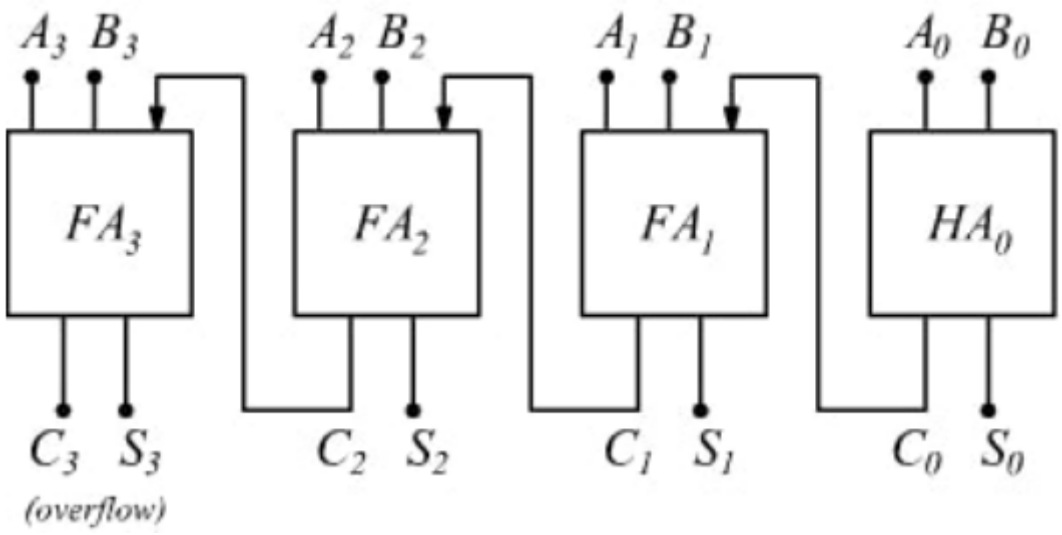
Full adder

Il circuito full-adder si presenta con tre ingressi e due uscite:

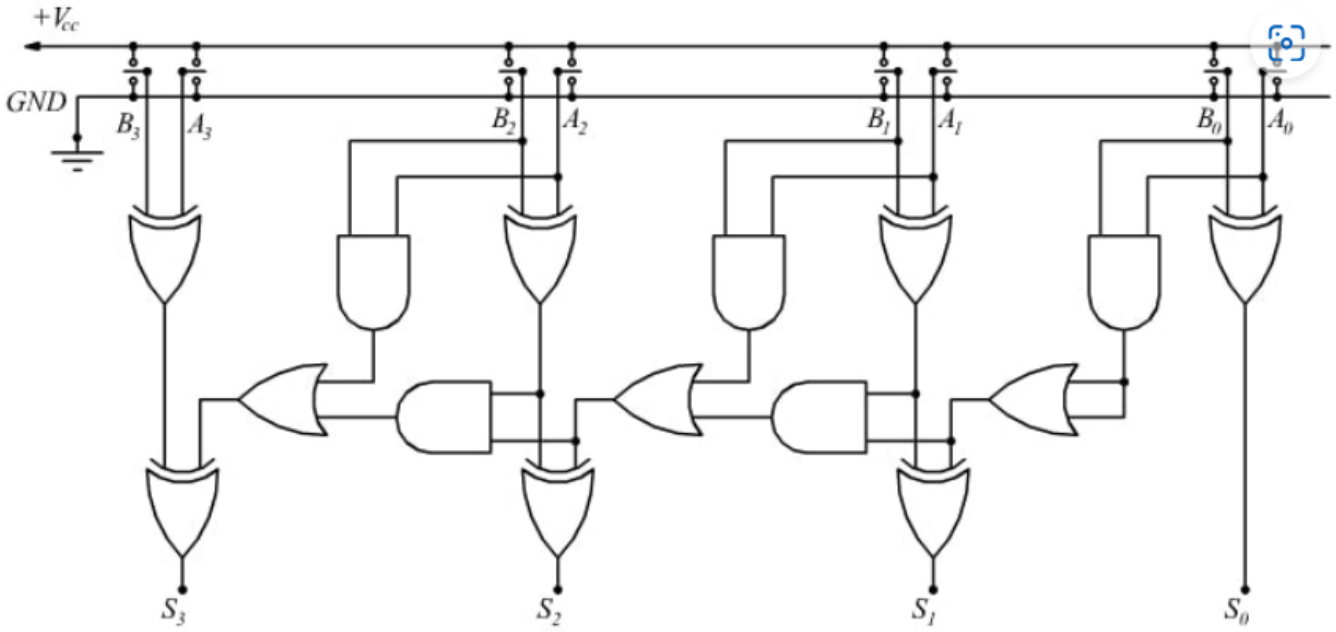


ingressi			uscite	
A	B	C_i	S	C_o
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1





Esempio con parole a 4 bit:



Trasparenza

Un latch può essere “trasparente” o “in memorizzazione” (hold), e si troverà nell'uno o nell'altro stato a seconda del livello (alto o basso) del segnale di clock.

- Quando un latch è “trasparente” ogni variazione dell'ingresso I comporta immediatamente una variazione dell'uscita U.

Differenza tra latch e flipflop

- Latch = Attivo con segnale di enable
- Flip-flop = Dipende dal clock