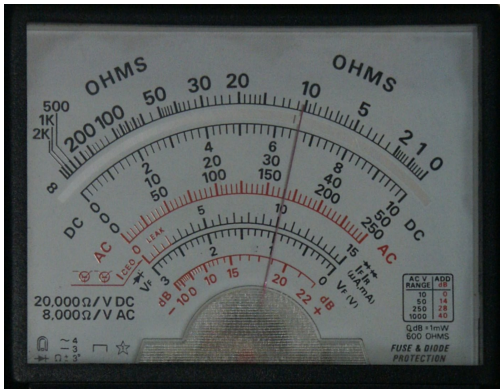
Analogico e digitale



Lettura analogica di una tensione.

L'informazione analogica (tensione) viene riportata sul quadrante sotto forma di un angolo proporzionale al valore della tensione

Lettura digitale di una tensione.

L'informazione analogica iniziale (tensione) viene trasformata in un codice in cui *N* circuiti digitali elementari (bit) comandano lo stato acceso/spento di ciascun segmento del display.

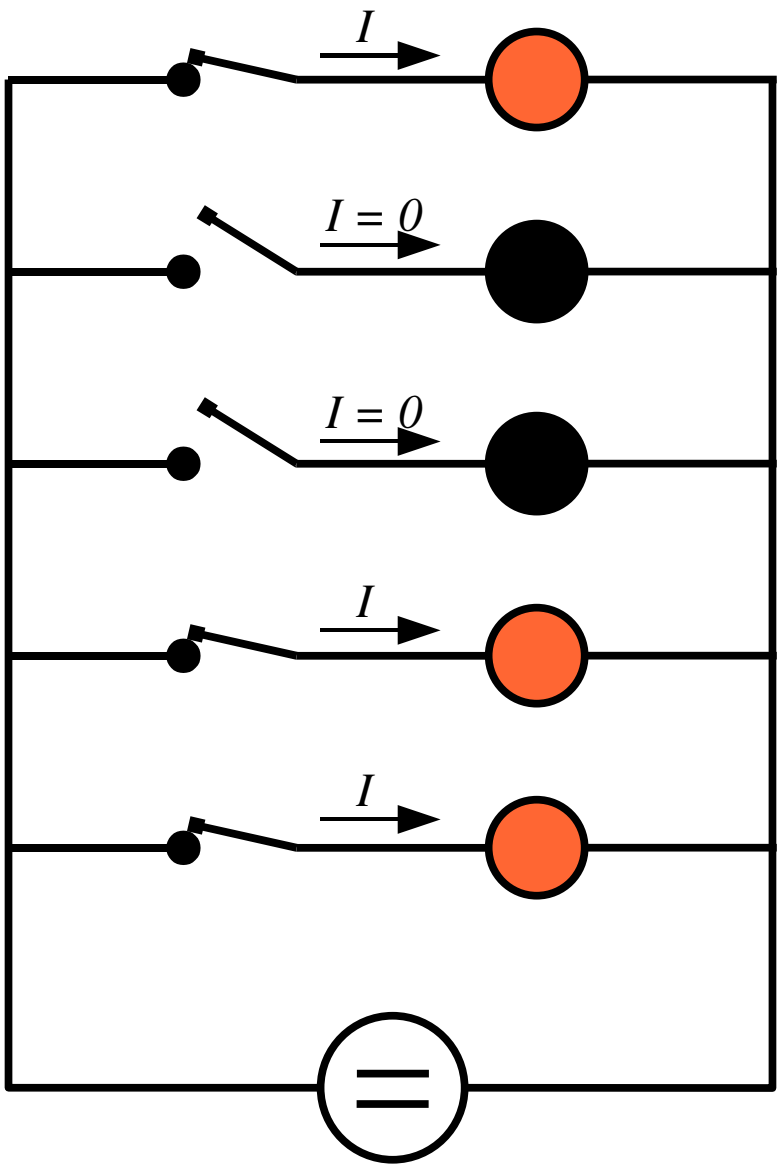
Analogico

Un solo conduttore la cui tensione e' la grandezza fisica di interesse.

Digitale

N conduttori ciascuno dei quali puo' avere solo due valori possibili di tensione, indicati con 0 e 1. La combinazione di 0 e 1 rappresenta un numero binario, che contiene il valore della grandezza fisica di interesse.

Numeri binari



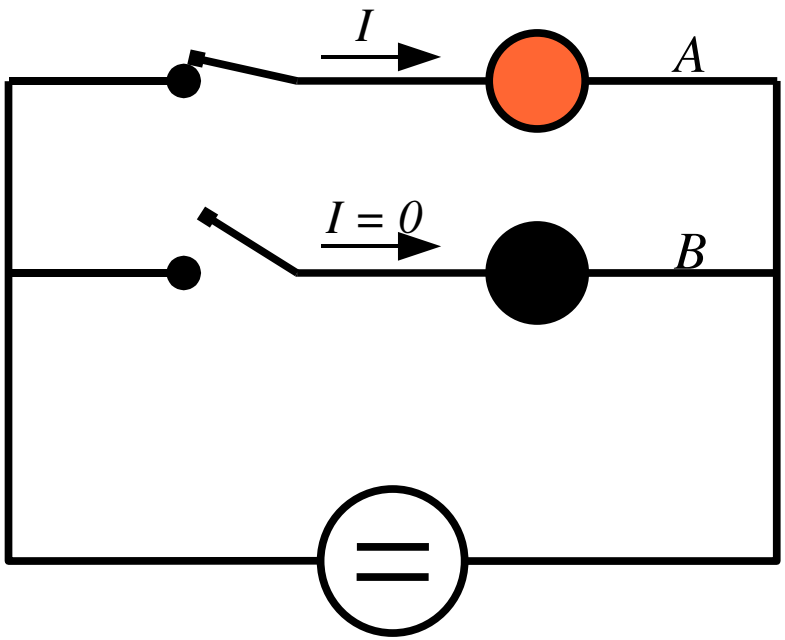
Conversione binario ↔ decimale

b_4	b_3	b_2	b_1	b_0	cifra
1	**0**	**0**	**1**	**1**	**numero binario**
2^4	2^3	2^2	2^1	2^0	valore della cifra
16	**+ 0**	**+ 0**	**+ 2**	**+ 1**	**= 19** valore decimale

Il numero decimale 19 scritto in codice binario

Il numero decimale 19 rappresentato mediante cinque circuiti elettrici binari.

Variabili logiche



Interruttore chiuso: nel circuito circola corrente.
Valore logico: 1 (oppure vero, +, acceso, rosso ...)

Interruttore aperto: nel circuito non circola corrente.
Valore logico: 0 (oppure falso, -, spento, nero ...)

A e B sono due *variabili logiche.*

$A = 1$

$B = 0$

Un *bit* e' una variabile logica binaria che puo' assumere uno di due valori (stati):

0	**1**
vero	**falso**
rosso	**nero**

I due circuiti elettrici dello schema contengono ciascuno il valore di un *bit.*

Algebra di Boole

Operazioni tra variabili logiche

Le tre operazioni logiche elementari tra variabili logiche sono:

AND ***OR*** ***NOT***

C = A ***AND*** B La variabile C e' 1 se A *e* B sono entrambe 1; altrimenti e' 0.

C = ***NOT*** A La variabile C e' 1 se A e' 0; la variabile C e' 0 se A e' 1.

C = A ***OR*** B La variabile C e' 1 se A *oppure* B (o entrambi) sono 1; altrimenti e' 0.

Tavole di *verita'*

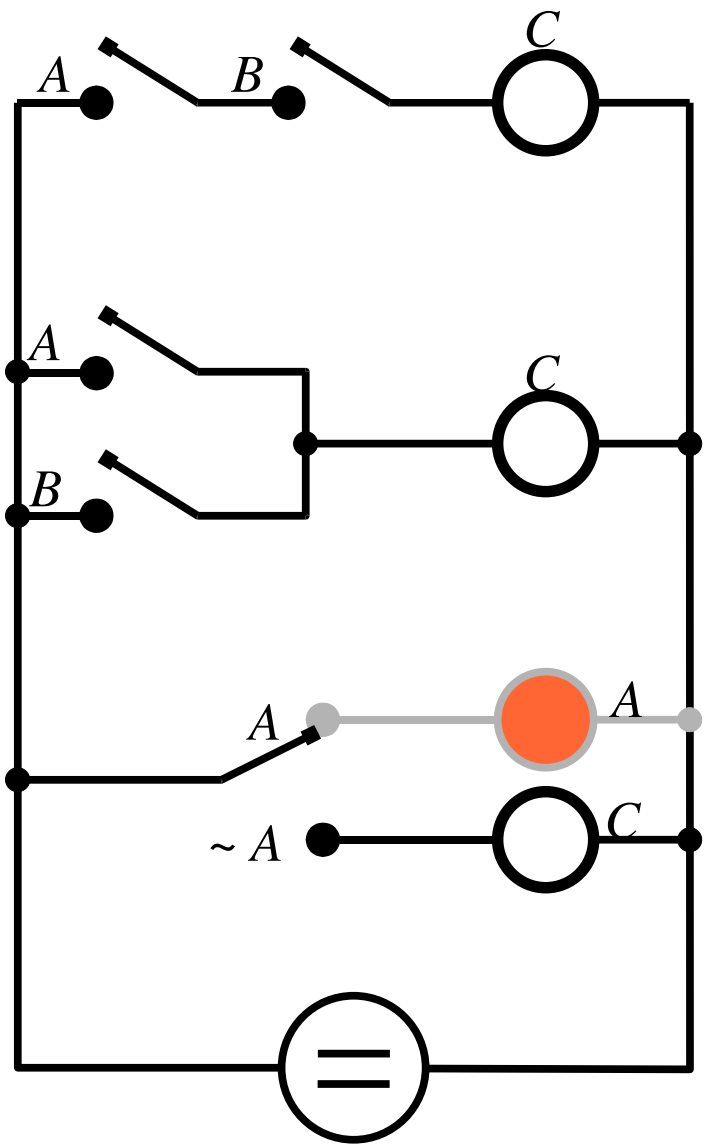
		$A \wedge B$
A	B	A ***AND*** B
0	0	0
0	1	0
1	0	0
1	1	1

		$A \vee B$
A	B	A ***OR*** B
0	0	0
0	1	1
1	0	1
1	1	1

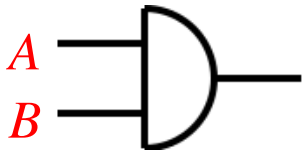
	$\sim A$
A	***NOT*** A
0	1
1	0

Algebra di Boole

Operazioni logiche mediante i circuiti elettrici

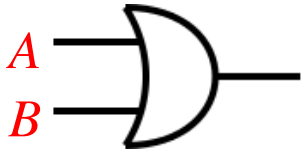


C (lampada accesa) = interruttore A chiuso e interruttore B chiuso



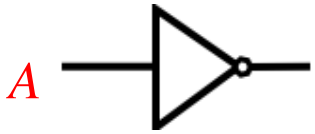
$C = A \wedge B$

C (lampada accesa) = interruttore A chiuso o interruttore B chiuso



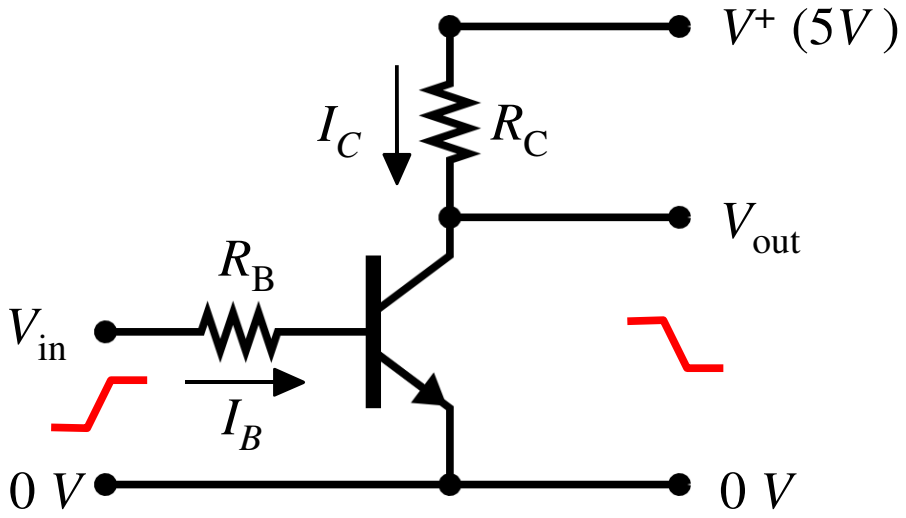
$C = A \vee B$

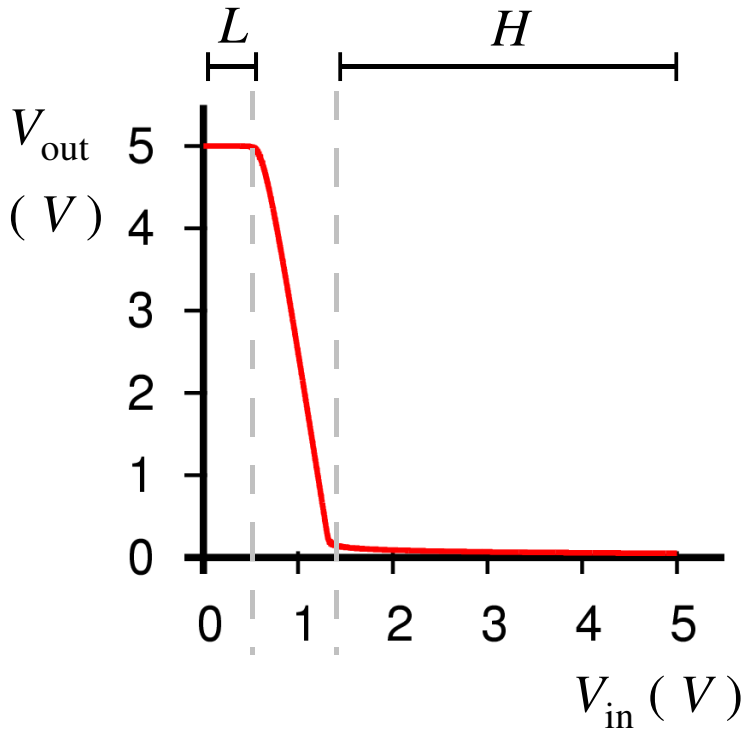
C (lampada accesa) = interruttore A aperto

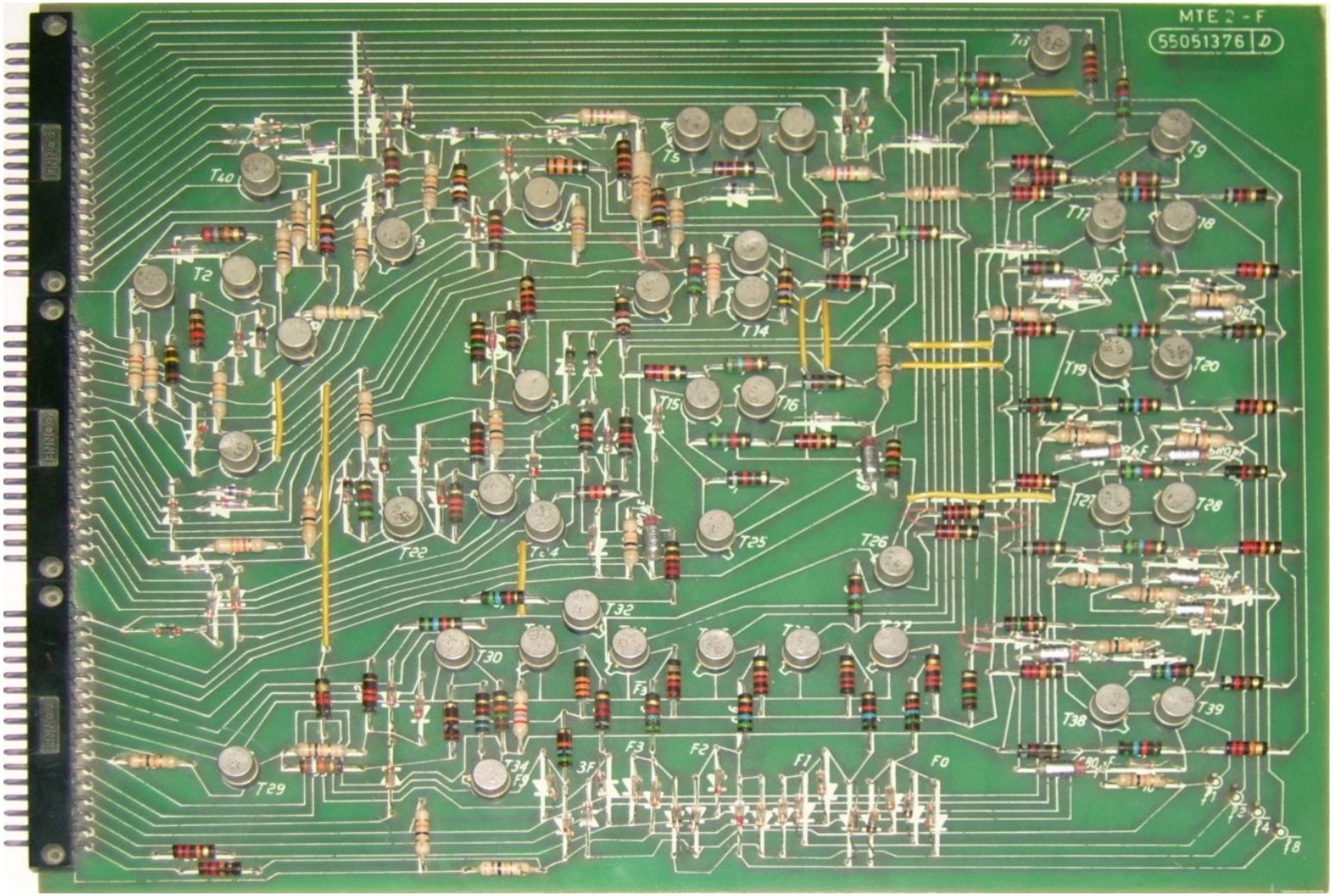


$C = \sim A$

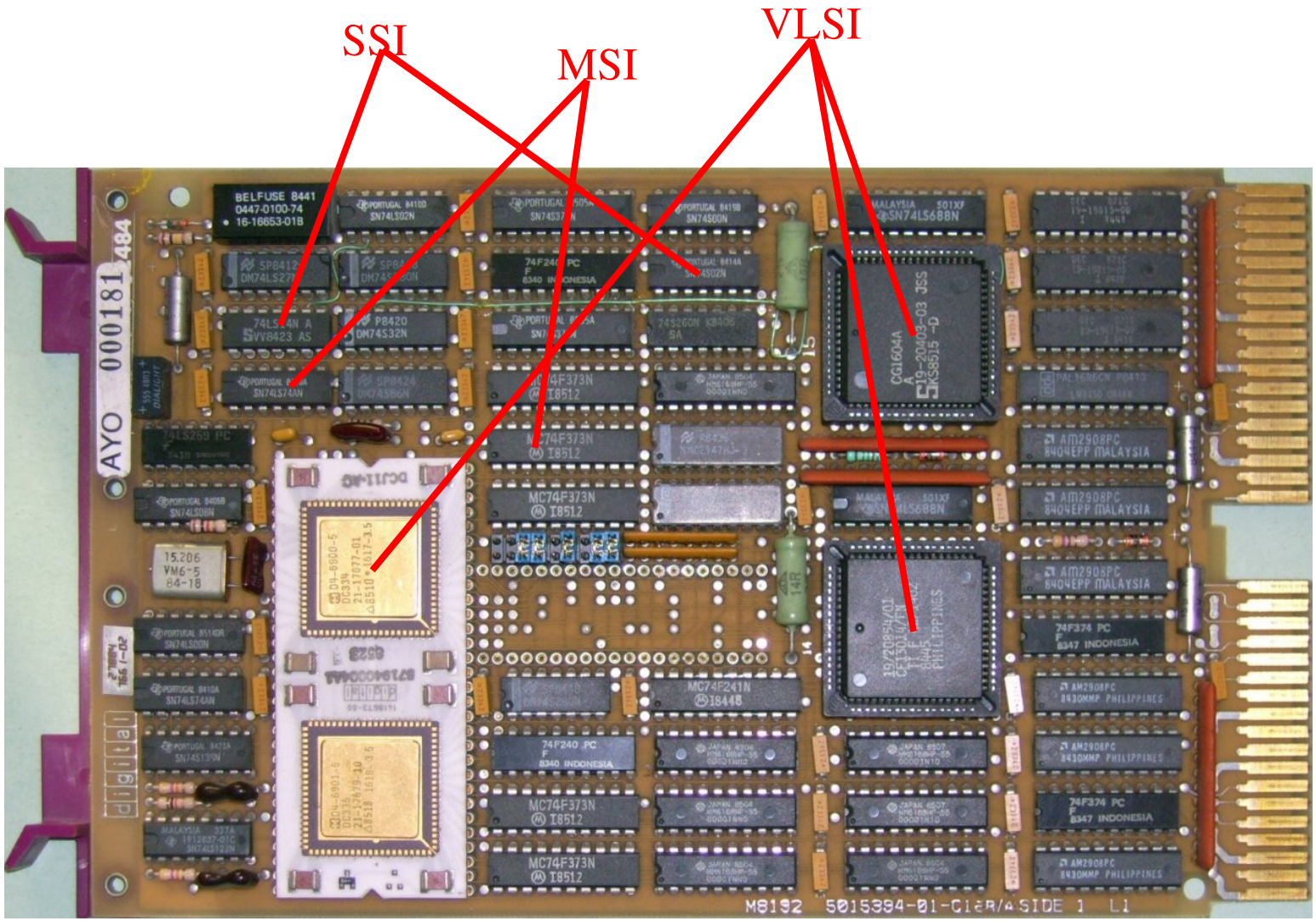
Il transistor come interruttore







Parte di unita' di calcolo realizzata a componenti discreti

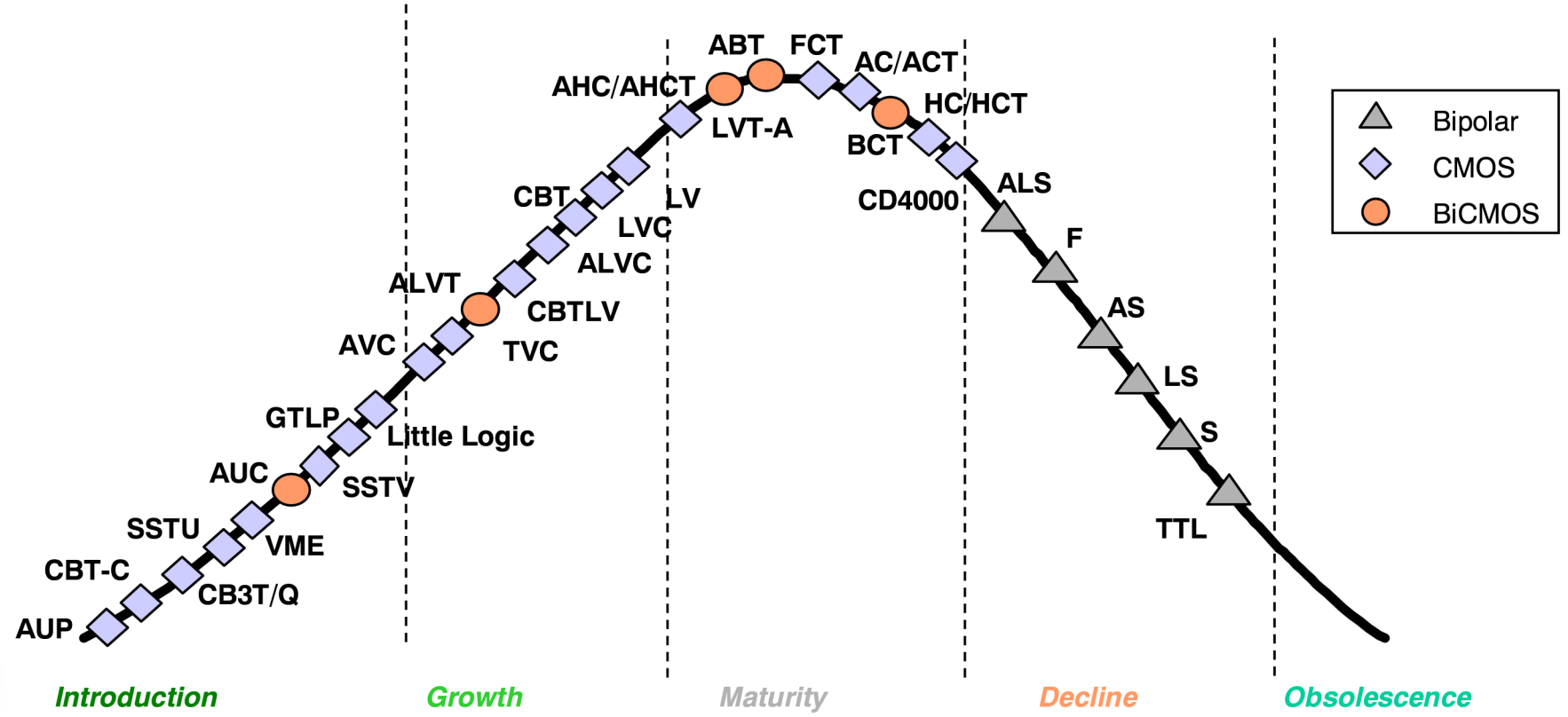


Unita' di calcolo di un computer PDP11-23 contenente circuiti integrati ad alta e a bassa integrazione

Famiglie logiche attualmente in uso

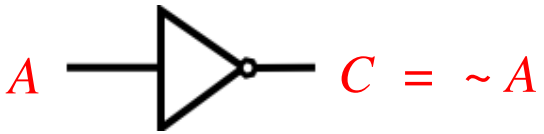
BiPolar	TTL	Transistor-Transistor Logic	
	ECL	Emitter coupled logic	
	S	Schottky Logic	
	LS	Low-Power Schottky Logic	
	AS	Advanced Schottky Logic	
	F	Fast Logic	
	ALS	Advanced Low-Power Schottky Logic	
CMOS	C	CMOS Logic	
	CD4000	CMOS Logic	
	HC	High-Speed CMOS Logic	
	HCT	High-Speed CMOS Logic	
	AC	Advanced CMOS Logic	
	ACT	Advanced CMOS Logic	
	FCT	Fast CMOS Technology	
	AHC	Advanced High-Speed CMOS	
	AHCT	Advanced High-Speed CMOS	
	ALVC	Advanced Low-Voltage CMOS Technology	
	AVC	Advanced Very-Low-Voltage CMOS Logic	
	AUC	Advanced Ultra-Low-Voltage CMOS Logic	
	AUP	Advanced Ultra-Low-Power CMOS Logic	
BiCMOS	BCT	BCT	BiCMOS Technology
	ABT	ABT	Advanced BiCMOS Technology
	LVT	LVT	Low-Voltage BiCMOS Technology
	ALVT	ALVT	Advanced Low-Voltage CMOS Technology

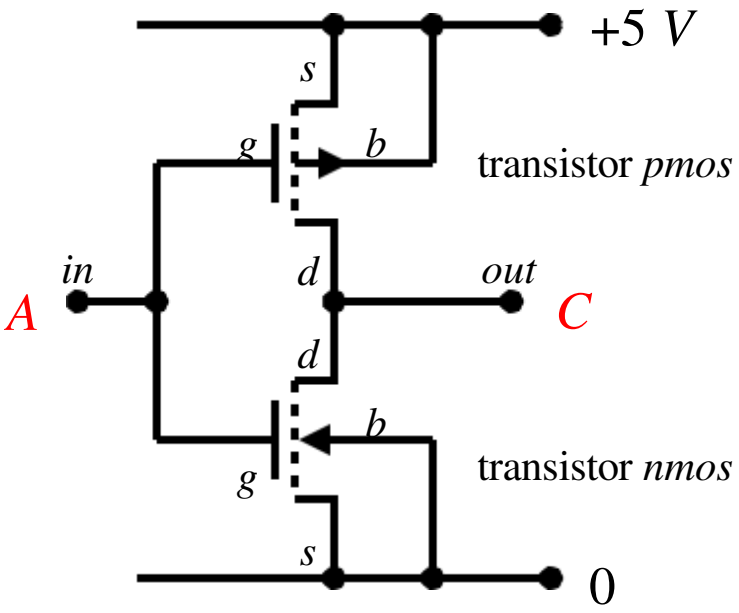
Product Life Cycle

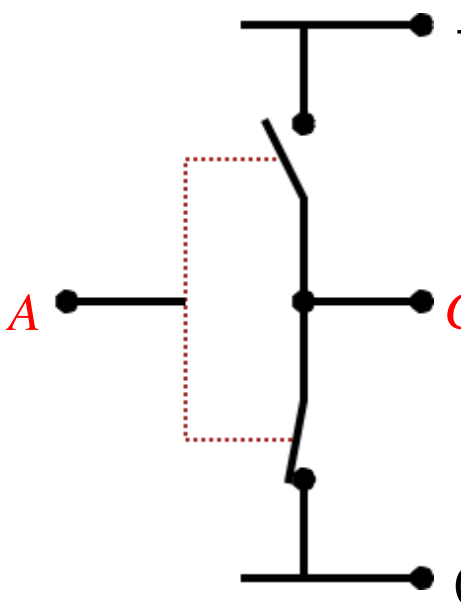


Il transistor come interruttore

Circuito *NOT* a transistor *CMOS* (*Complementary-MOS*)







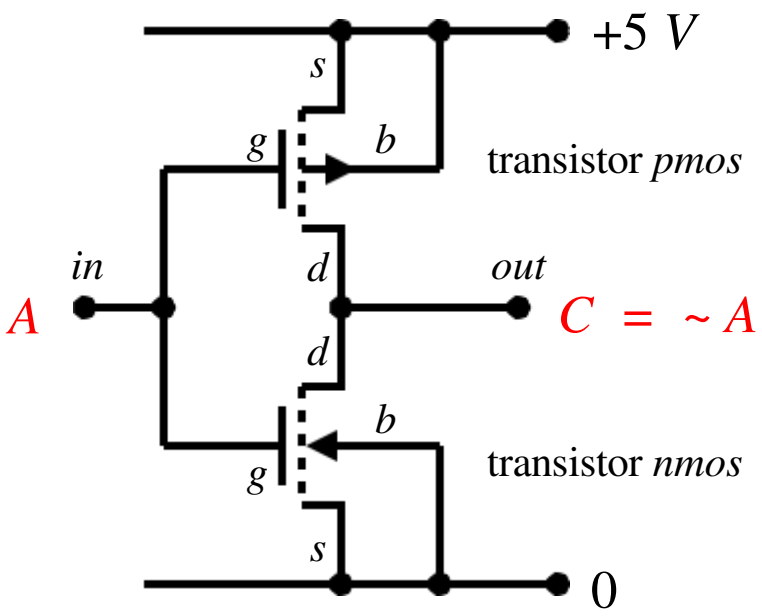
Il transistor *mos* spento ($|V_G| < |V_T|$) presenta tra *drain* e *source* una resistenza elevatissima.

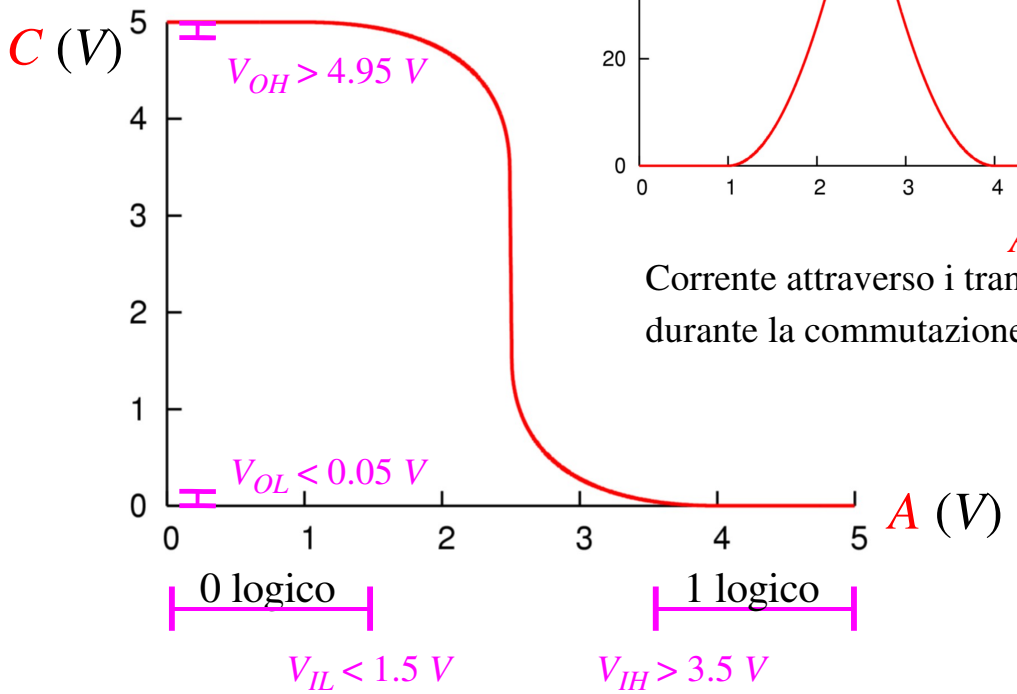
Il transistor *mos* acceso ($|V_G| > |V_T|$) funziona nella regione lineare e si comporta come una resistenza ohmica di piccolo valore.

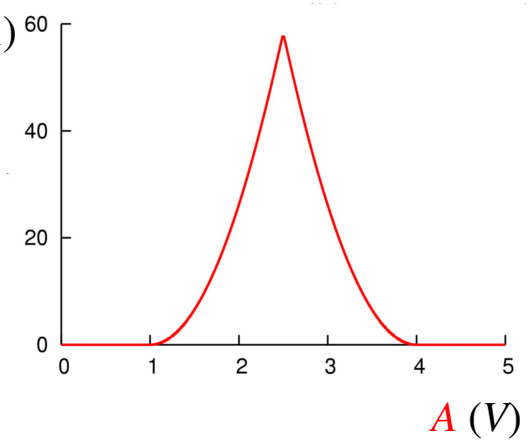
Circuito *NOT* (*inverter*) a transistor *CMOS*

I due transistor *mos* funzionano in maniera *complementare* come due interruttori contrapposti: quando la tensione di ingresso e' 0 *V* il transistor *nmos* e' spento ed il *pmos* e' acceso e l'uscita *C* e' collegata alla tensione +5 *V*; viceversa con tensione di ingresso +5 *V* il transistor *nmos* e' acceso ed il *pmos* spento e l'uscita e' collegata alla massa (0 *V*).

Circuito *NOT* a transistor *CMOS*







Corrente attraverso i transistor durante la commutazione.

Margine di rumore:

$V_{IL} - V_{OL} = 1.45\ V$

$V_{IH} - V_{OH} = -\ 1.45\ V$

Il *margine di rumore* misura l'ampiezza di un ipotetico disturbo in grado di far confondere un livello ***L*** con un ***H*** o viceversa.

0 = *Low* 1 = *High*

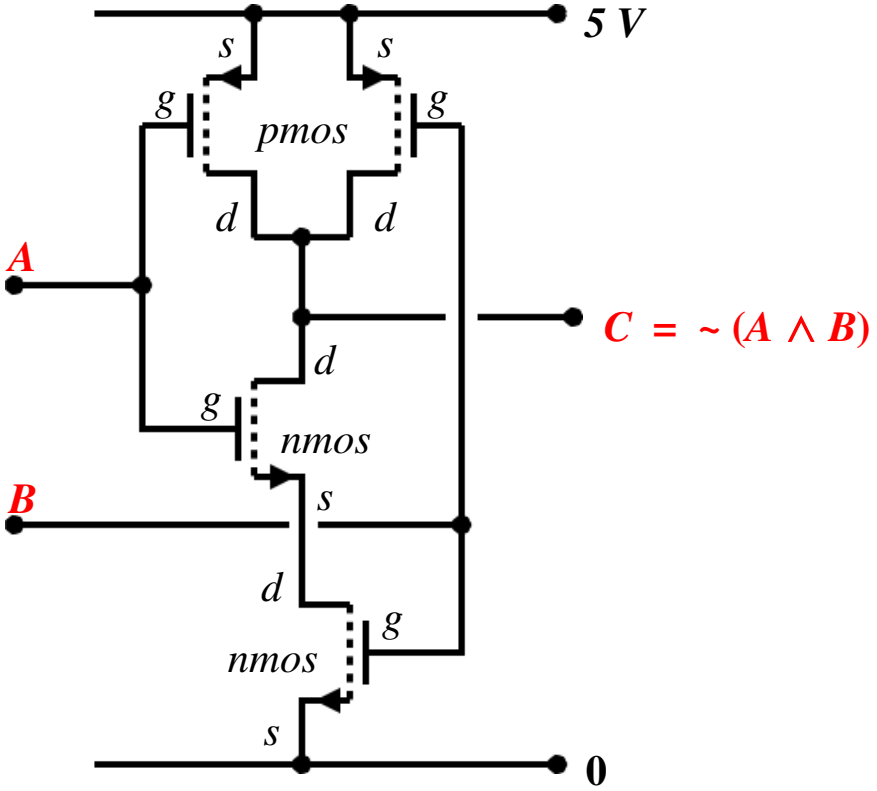
V_{IL} = massima tensione di ingresso con valore 0 (*Low*)

V_{IH} = minima tensione di ingresso con valore 1 (*High*)

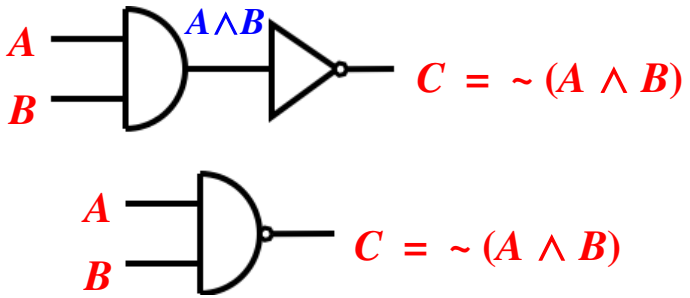
V_{OH} = minima tensione di uscita al valore 1 (*High*)

V_{OL} = massima tensione di uscita al valore 0 (*Low*)

Porte logiche *in tecnologia CMOS* – Porta *NAND*



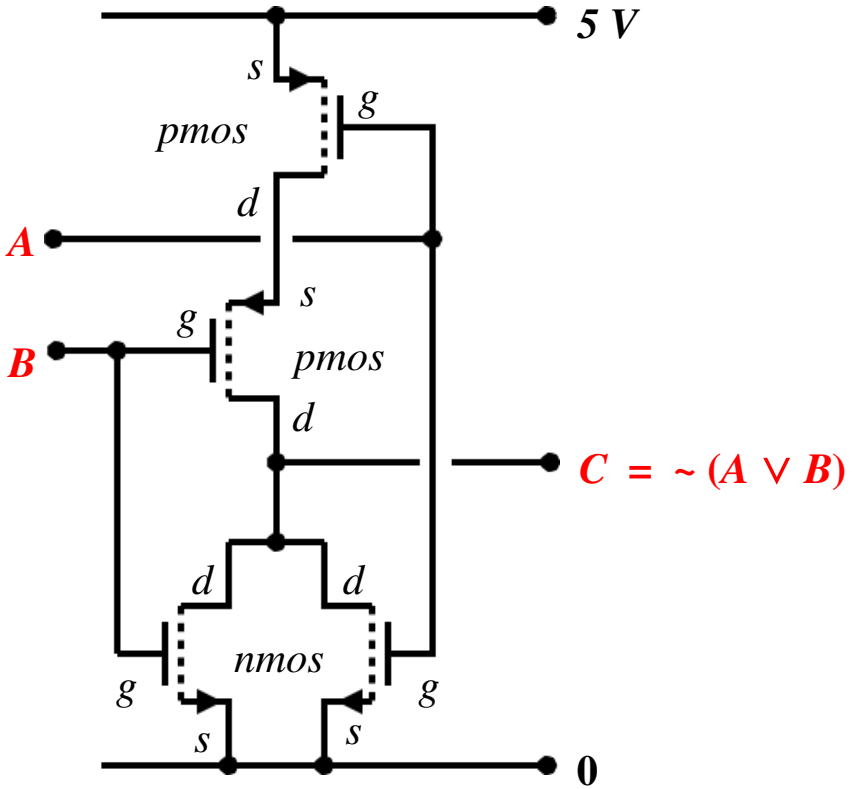
NAND (A,B) = NOT (AND(A,B))



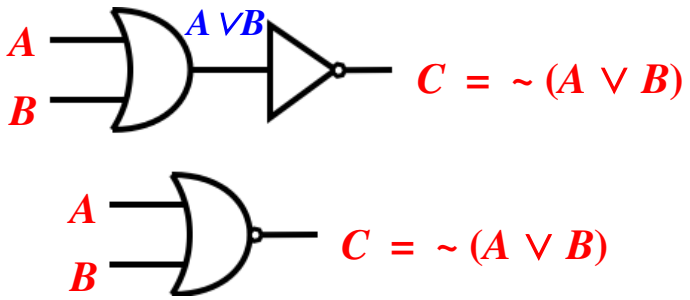
A	*B*	~ (*A* ∧ *B*)
0	0	1
0	1	1
1	0	1
1	1	0

Ogni operazione (*AND*, *OR*) effettuata con dispositivi attivi (transistor c*mos*, *bjt* ecc.) comporta sempre anche una *inversione* (*NOT*). Le operazioni base sono *NAND*, *NOR* e *NOT*.

Porte logiche *in tecnologia CMOS* – Porta *NOR*



NOR (A,B) = NOT (OR (A,B))



A	*B*	~ (*A* ∨ *B*)
0	0	1
0	1	0
1	0	0
1	1	0

Leggi di De Morgan

Le tre operazioni logiche elementari *AND*, *OR* e *NOT* non sono indipendenti:

$$\sim (A \vee B) = \sim A \wedge \sim B$$

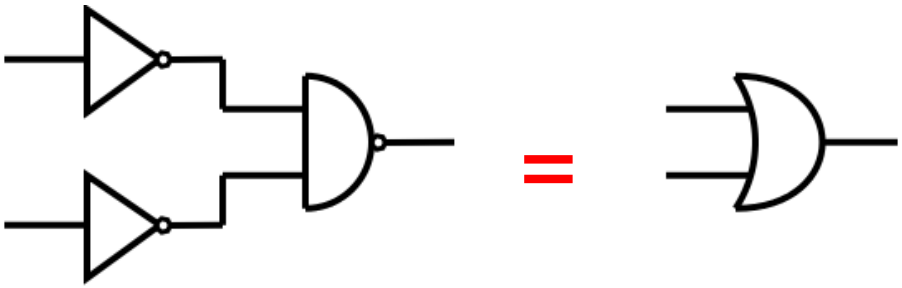
$$\sim (A \wedge B) = \sim A \vee \sim B$$

Leggi di De Morgan

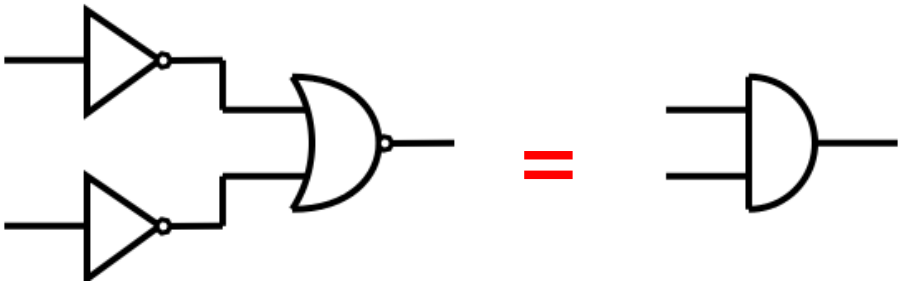
			$\sim A \wedge \sim B$		
A	B	$A \vee B$	$\sim (A \vee B)$	$\sim A$	$\sim B$
0	0	0	1	1	1
0	1	1	0	1	0
1	0	1	0	0	1
1	1	1	0	0	0

			$\sim A \vee \sim B$		
A	B	$A \wedge B$	$\sim (A \wedge B)$	$\sim A$	$\sim B$
0	0	0	1	1	1
0	1	0	1	1	0
1	0	0	1	0	1
1	1	1	0	0	0

I legge



II legge



Il transistor *bjt* come interruttore

Livelli elettrici delle porte *TTL*

$V_{IL} = 0.8\ V$ $I_{IL} = -\ 1.6\ mA$ (0.4)

$V_{IH} = 2$ V $I_{IH} < 40\ \mu$A (20)

$V_{OL} < 0.4\ V$ @ $I_{OL} = 16$ mA (8)

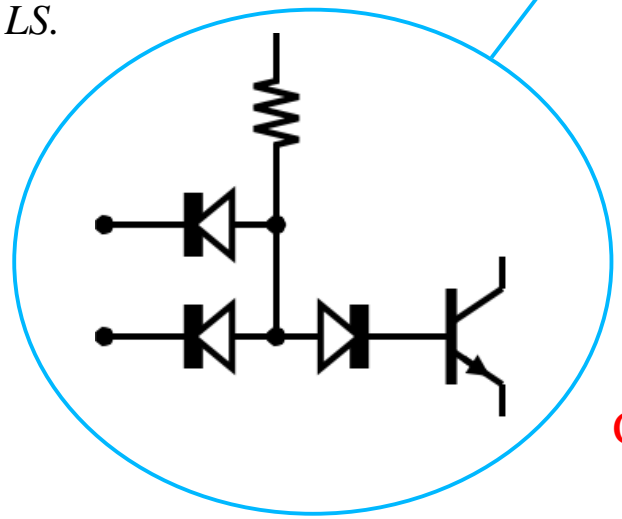
$V_{OH} > 2.4\ V$ @ $I_{OH} = -800\ \mu$A (400)

Margine di rumore:

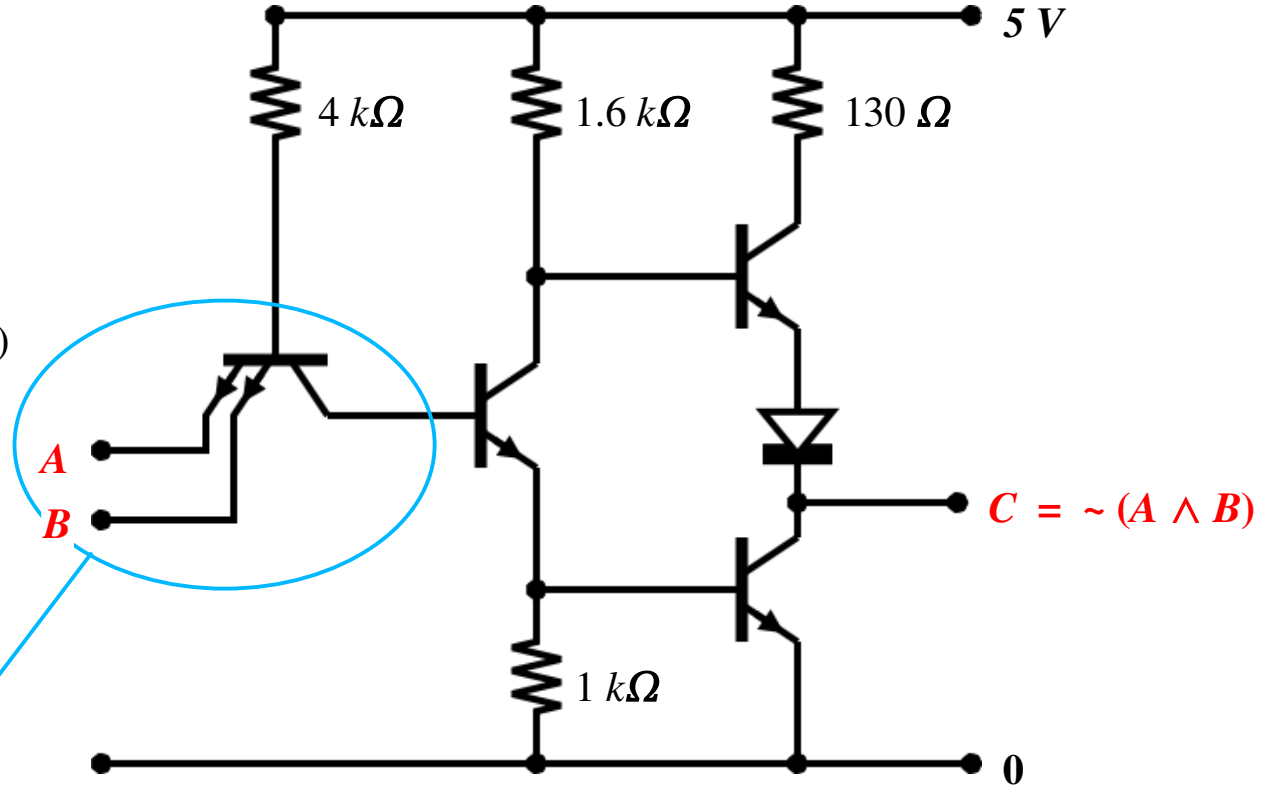
Low = 0.4 V High = 0.4 V

Fan-out = 10

I valori tra parentesi si riferiscono alla serie *LS*.



Circuito di ingresso equivalente in tecnologia *DTL*
(*Diode Transistor Logic*)



Porta *NAND* in tecnologia *TTL*
(*Transistor Transistor Logic*)

Numeri binari

Numerazione ottale:

- si usano le cifre da **0** a **7**
- ogni cifra rappresenta 3 bit

Numerazione esadecimale:

- si usano le cifre da **0** a **9** e i caratteri da ***A*** ad ***F***
- ogni cifra rappresenta 4 bit

Esempio:

$$\mathbf{1\,0\,1\,1\,0\,0\,1\,1_{2} = 2\,6\,3_{8} = B\,3_{16} = 179_{10}}$$

Codice *BCD*

Si codifica un numero decimale utilizzando 4 bit per ogni cifra:

$$\mathbf{179_{10} = 0\,0\,0\,1 \quad 0\,1\,1\,1 \quad 1\,0\,0\,1}$$

Numerazione ottale

0	0 0 0
1	0 0 1
2	0 1 0
3	0 1 1
4	1 0 0
5	1 0 1
6	1 1 0
7	1 1 1

Numerazione esadecimale

0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1
8	1 0 0 0
9	1 0 0 1
A	1 0 1 0
B	1 0 1 1
C	1 1 0 0
D	1 1 0 1
E	1 1 1 0
F	1 1 1 1

C o d i c i

Con 8 bit (2^8 = 256 combinazioni) si possono rappresentare:

- i numeri interi da 0 a 255
- i numeri interi da -128 a 127
- . . . . . .
- 256 simboli diversi

Il **codice ASCII** codifica in 8 bit caratteri dell'alfabeto e segni di interpunzione:

A = 0 1 0 0 0 0 0 1 = 1 0 1 $_8$ = 41 $_{16}$
B = 0 1 0 0 0 0 1 0 = 1 0 2 $_8$ = 42 $_{16}$
C = 0 1 0 0 0 0 3 1 = 1 0 3 $_8$ = 43 $_{16}$
. . . . . .
a = 0 1 1 0 0 0 0 1 = 1 4 1 $_8$ = 61 $_{16}$
a = 0 1 1 0 0 0 1 0 = 1 4 2 $_8$ = 62 $_{16}$
.
{ = 0 1 1 1 1 0 1 1 = 1 7 3 $_8$ = 7B $_{16}$
.
.

Codici

Con opportuni codici e' possibile rappresentare qualunque insieme (finito) di oggetti.

Alcuni esempi di colori codificati con un codice a 24 bit (6 cifre esadecimali):

Logica combinatoria

Una rete logica *combinatoria* fornisce alle proprie uscite valori logici che sono funzione esclusivamente dei valori delle variabili logiche di ingresso.

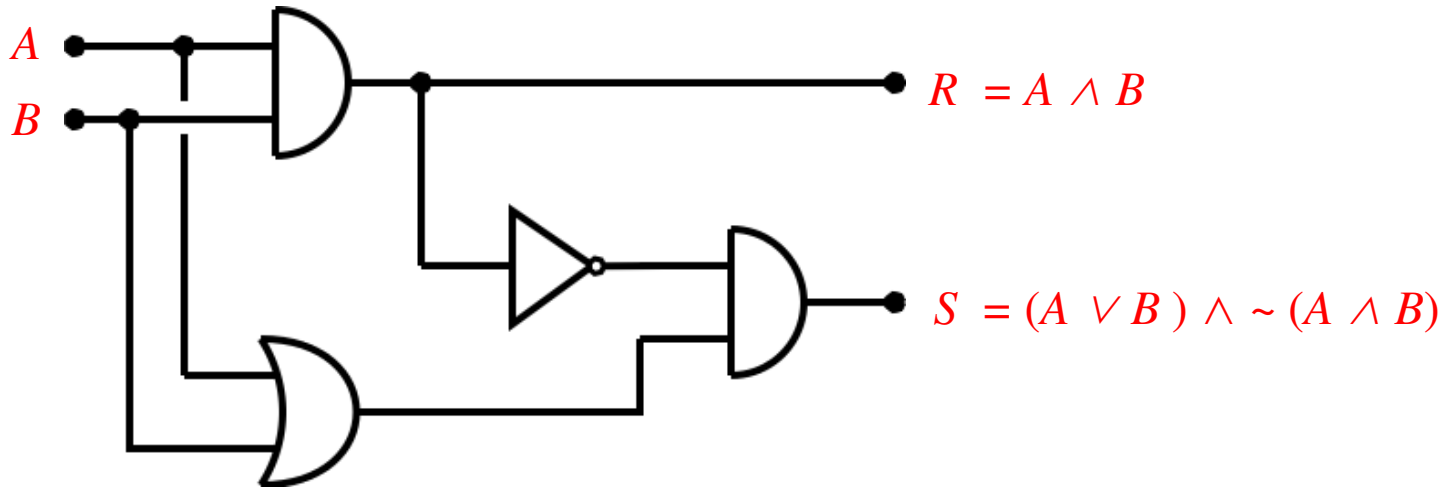
Circuito (semi) addizionatore (*half adder*)

Un addizionatore di due cifre binarie e' un esempio di logica combinatoria.

Ingressi: A, B (Rappresentano ciascuna una quantita' numerica 0/1)

Uscite: S, R (Contengono il valore numerico di $A+B$. S = somma; R = riporto)

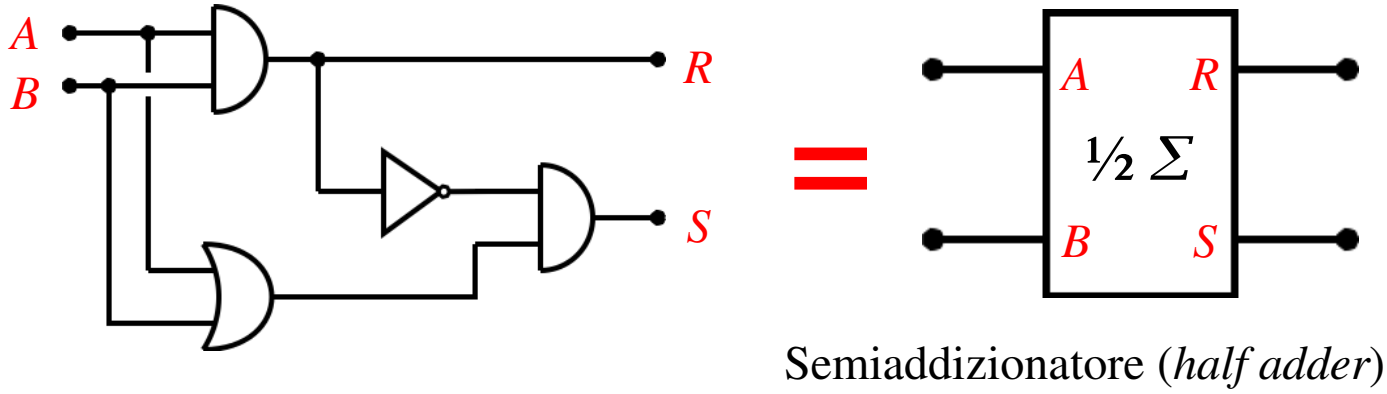
A	B	R	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



Circuito addizionatore (*full adder*)

Un addizionatore completo per numeri binari di N cifre deve prevedere per ogni cifra anche il riporto dalla cifra precedente.

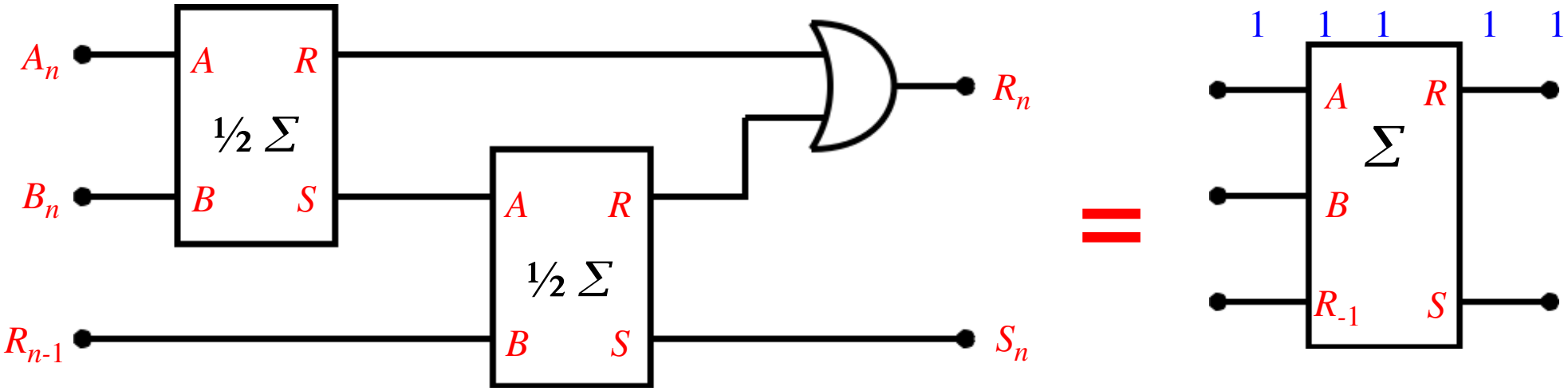
Un addizionatore completo puo' essere ottenuto combinando due semiadizionatori:



Semiaddizionatore (*half adder*)

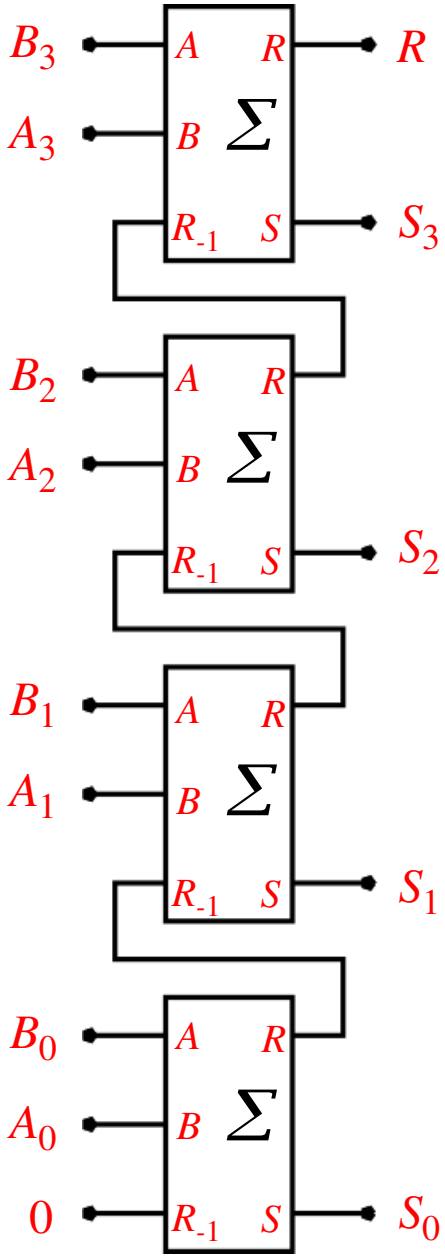
tavola di verita' dell'addizionatore completo (*full adder*)

A	B	R_{-1}	S	R
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1



Addizionatore completo ottenuto combinando due semiaddizionatori ed una porta *OR*

Addizionatore per due numeri binari di 4 bit.

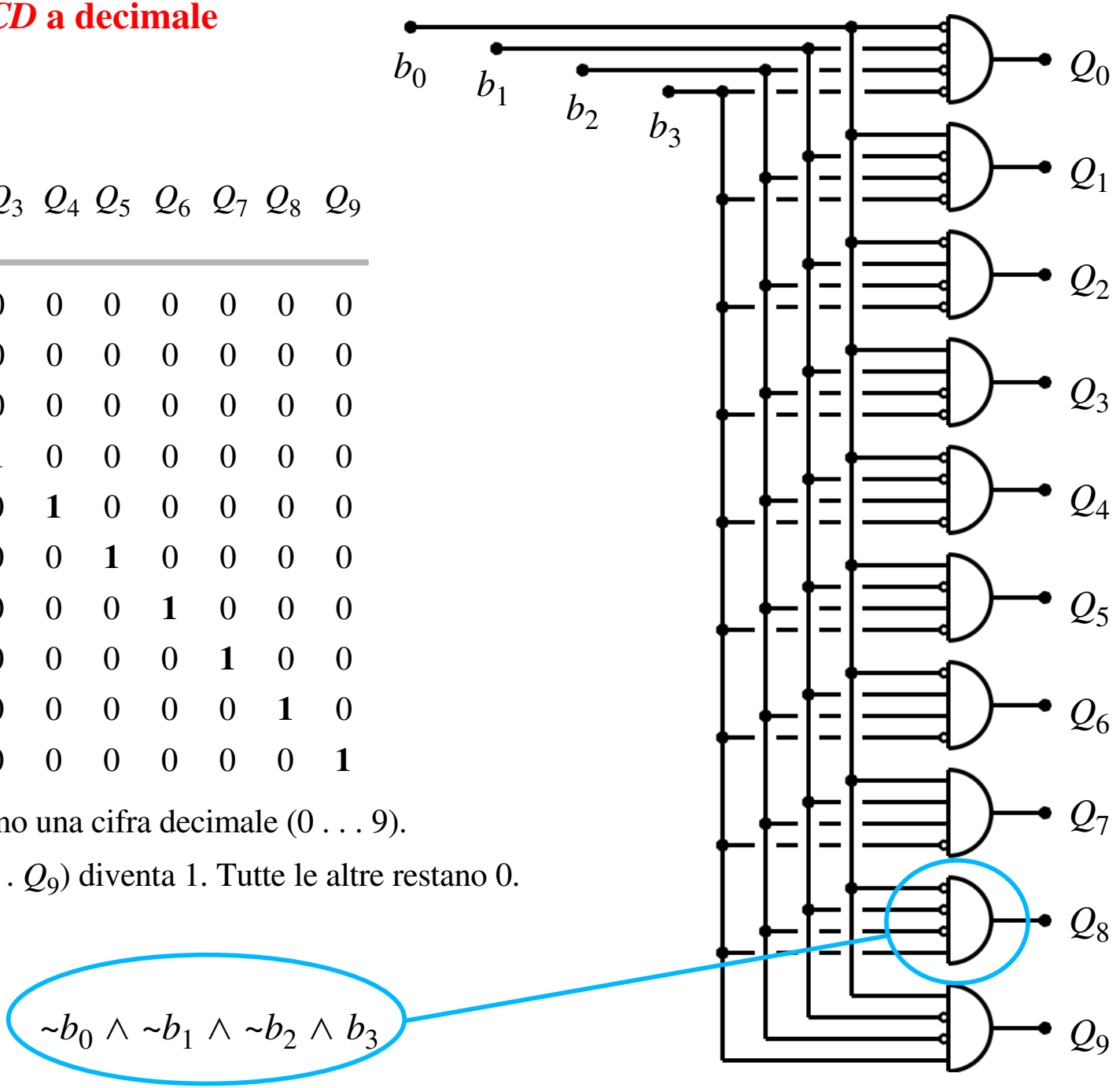


Convertitore da codice *BCD* a decimale

b_3	b_2	b_1	b_0	Q_0	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7	Q_8	Q_9
0	0	0	0	**1**	0	0	0	0	0	0	0	0	0
0	0	0	1	0	**1**	0	0	0	0	0	0	0	0
0	0	1	0	0	0	**1**	0	0	0	0	0	0	0
0	0	1	1	0	0	0	**1**	0	0	0	0	0	0
0	1	0	0	0	0	0	0	**1**	0	0	0	0	0
0	1	0	1	0	0	0	0	0	**1**	0	0	0	0
0	1	1	0	0	0	0	0	0	0	**1**	0	0	0
0	1	1	1	0	0	0	0	0	0	0	**1**	0	0
1	0	0	0	0	0	0	0	0	0	0	0	**1**	0
1	0	0	1	0	0	0	0	0	0	0	0	0	**1**

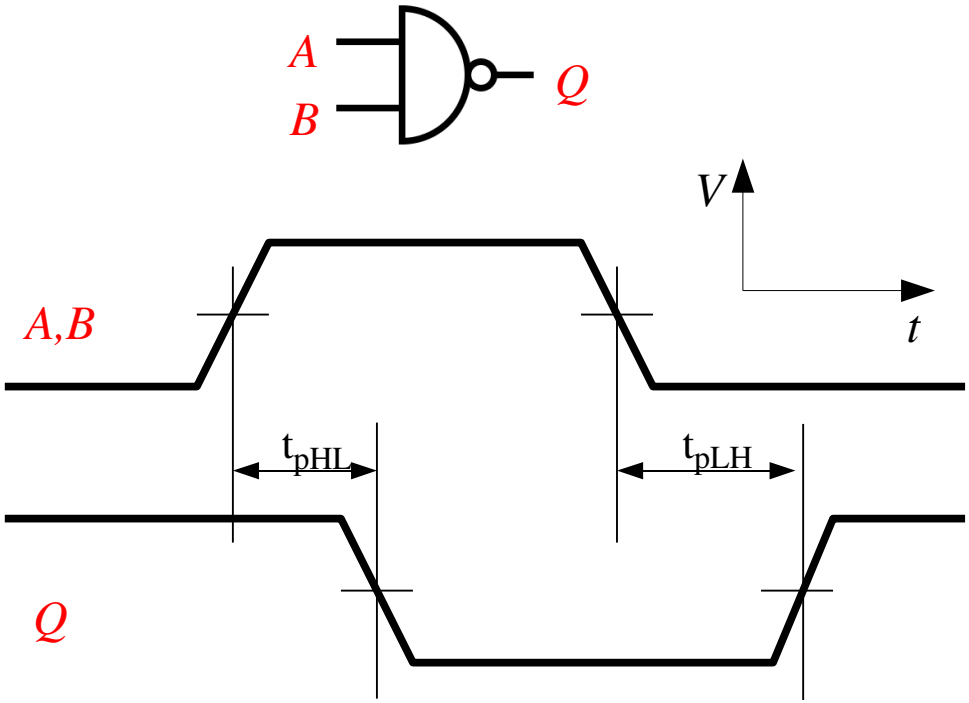
Gli ingressi $b_0 \ldots b_3$ codificano una cifra decimale (0 . . . 9).

L'uscita corrispondente ($Q_0 \ldots Q_9$) diventa 1. Tutte le altre restano 0.



Tempi di commutazione

La propagazione dei segnali tra ingressi e uscite di una rete combinatoria non e' istantanea, ma richiede un tempo finito.

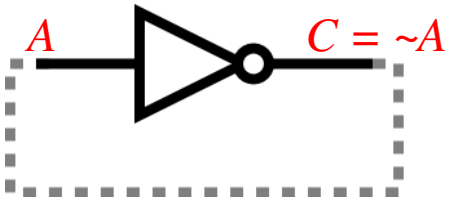


Propagazione dei segnali tra ingressi (A,B) e uscita (Q) di una porta *NAND*.

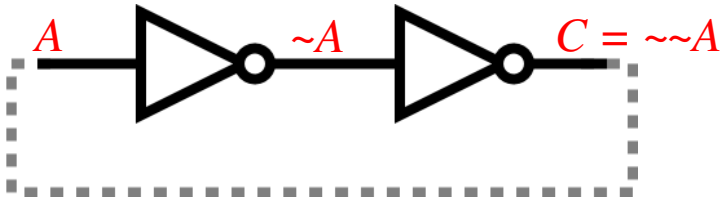
Tempi di propagazione in una porta *NAND* realizzata con diverse tecnologie (*nsec*)

	t_{pHL}		t_{pLH}	
	typ	max	typ	max
7400 (*TTL*)	7	15	11	22
74*LS*00 (*TTL-LS*)	3	10	3	10
74*F*00	3.2	4.3	3.7	5
74*ACT*00 (*CMOS*)	4	7	5.5	9

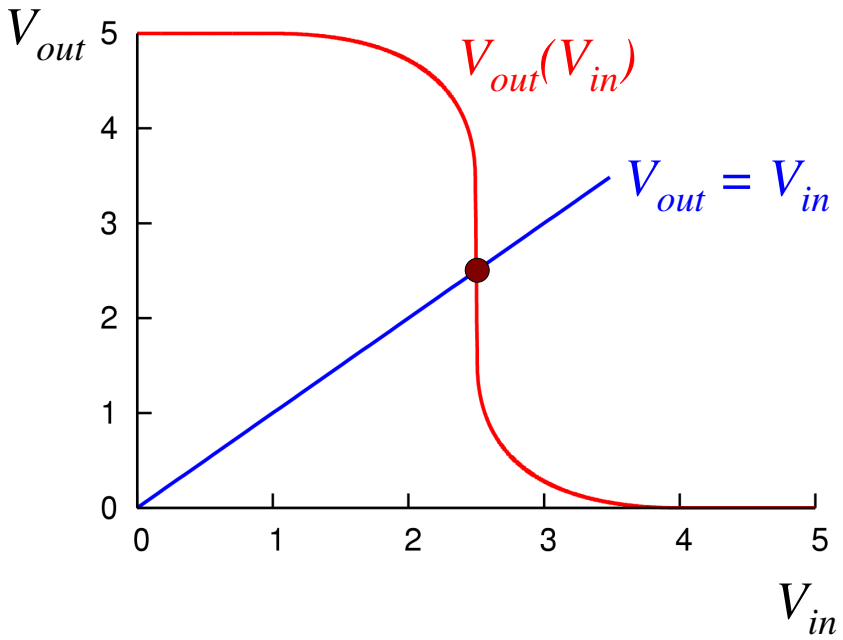
Logica sequenziale: *flip-flop*



Anello con reazione negativa



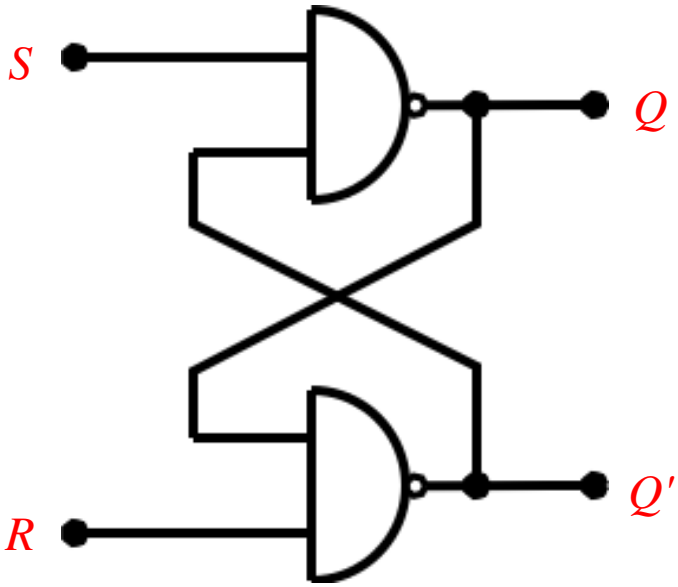
Anello con reazione positiva



Logica sequenziale

flip-flop

Una rete logica *sequenziale* fornisce alle proprie uscite valori logici che sono funzione dei valori delle variabili logiche di ingresso e della storia precedente del circuito.



circuito *bistabile*

flip-flop di tipo *set-reset*

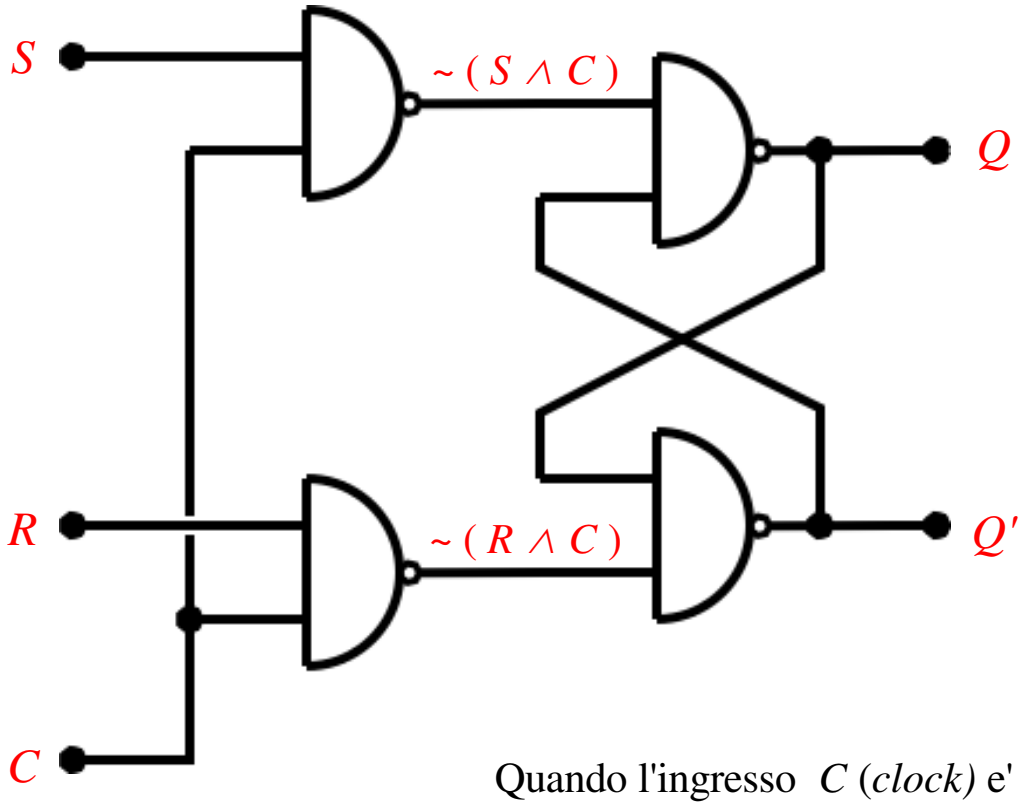
S	R	Q	Q'
0	0	1	1
0	1	1	0
1	0	0	1
1	1	0	1
1	1	1	0

Quando gli ingressi S ed R sono entrambi ad 1 ci sono due stati stabili possibili per le uscite (circuito *bi*-stabile):

$$Q = 0 \quad Q' = 1 \qquad\qquad Q = 1 \quad Q' = 0.$$

Lo stato in cui si trova effettivamente il circuito dipende dalla storia precedente: quale dei due ingressi S o R si e' trovato per ultimo nello stato 0. Il circuito e' dotato di memoria. Puo' *ricordare* 1 bit di informazione.

clocked set-reset flip-flop



S	R	C	Q	Q'
0	0	1	$^{0}/_{1}$	$^{1}/_{0}$
0	1	1	0	1
1	0	1	1	0
1	1	1	1	1
X	X	0	Q_	Q'_

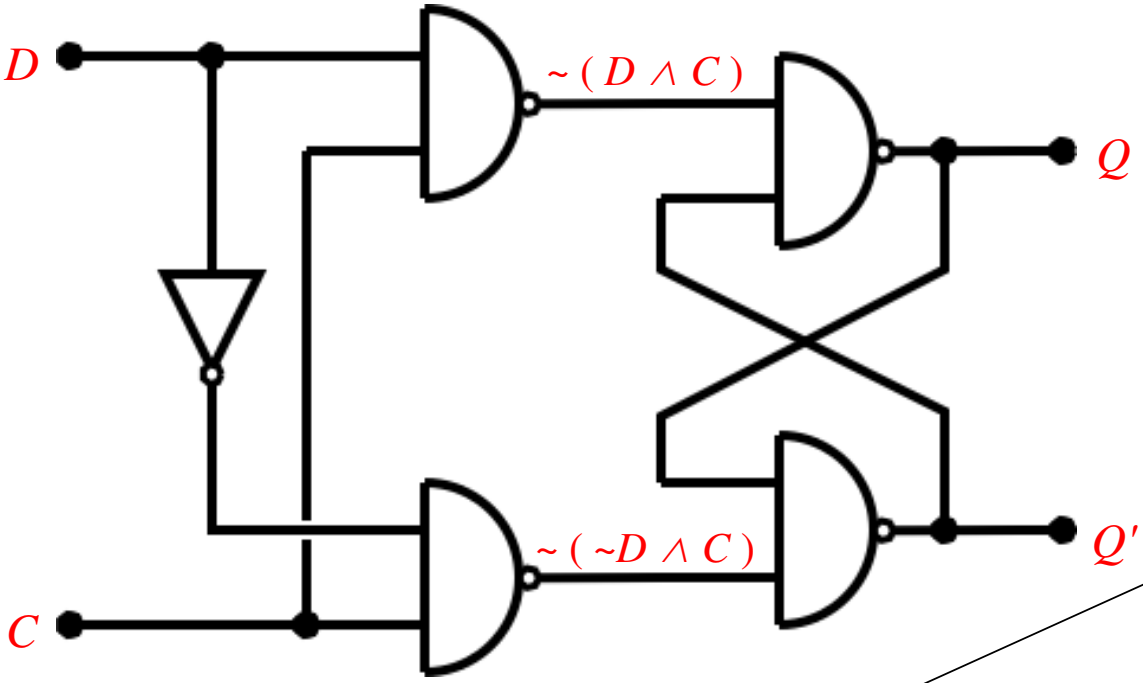
Quando l'ingresso *C* (*clock)* e' alto (1) i livelli degli ingressi *S* ed *R* vengono trasferiti al *flip-flop* ed il circuito si comporta come un *flip-flop* normale di tipo *set-reset*.

Quando l'ingresso *C* e' basso, lo stato delle uscite rimane *congelato* al valore *Q* / *Q'* presente prima della transizione 1 → 0 di *C*.

Se alla transizione 1 → 0 di *C* gli ingressi *S* ed *R* sono entrambi 1, lo stato 1-1 di *Q* e *Q'* decade in 1- 0 o 0 -1 in maniera imprevedibile (*race condition*).

Se il clock *C* diventa 1 con *S* ed *R* entrambi 0 lo stato delle uscite non viene modificato.

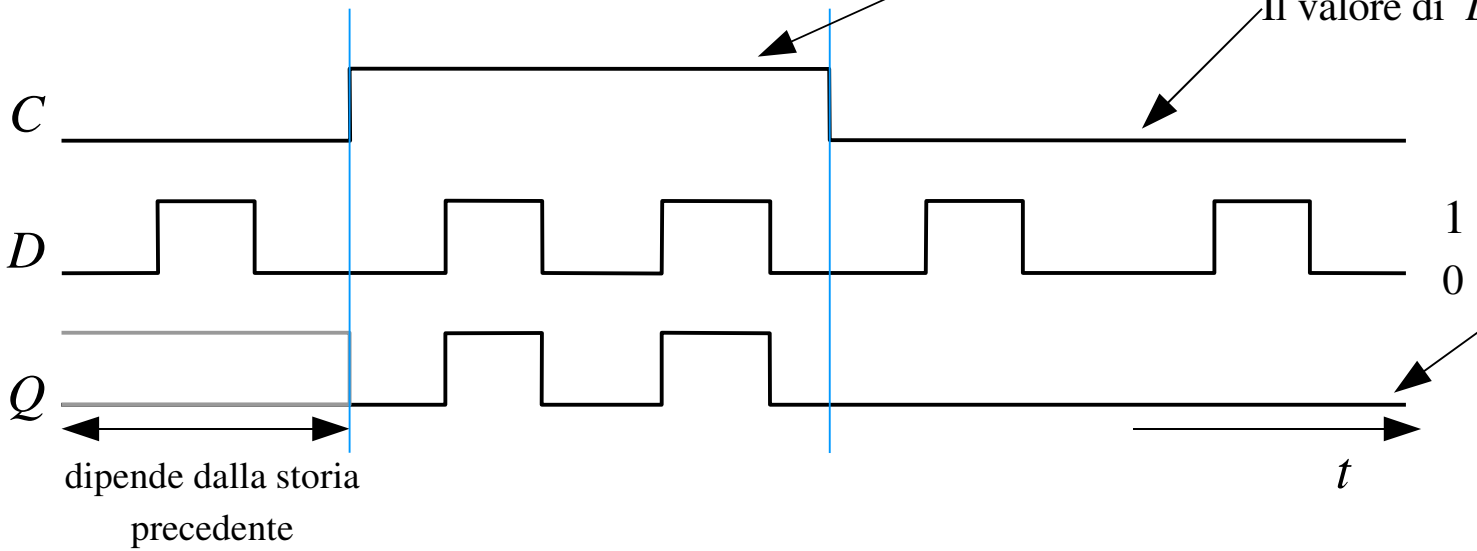
D-type flip-flop - latch



Il dato presente all'ingresso *D* viene trasferito all'uscita *Q* (e *Q'*) fino a che il *clock C* e' alto; quando il *clock C* diventa basso il valore delle uscite viene congelato ed ogni variazione su *D* non ha piu' alcun effetto.

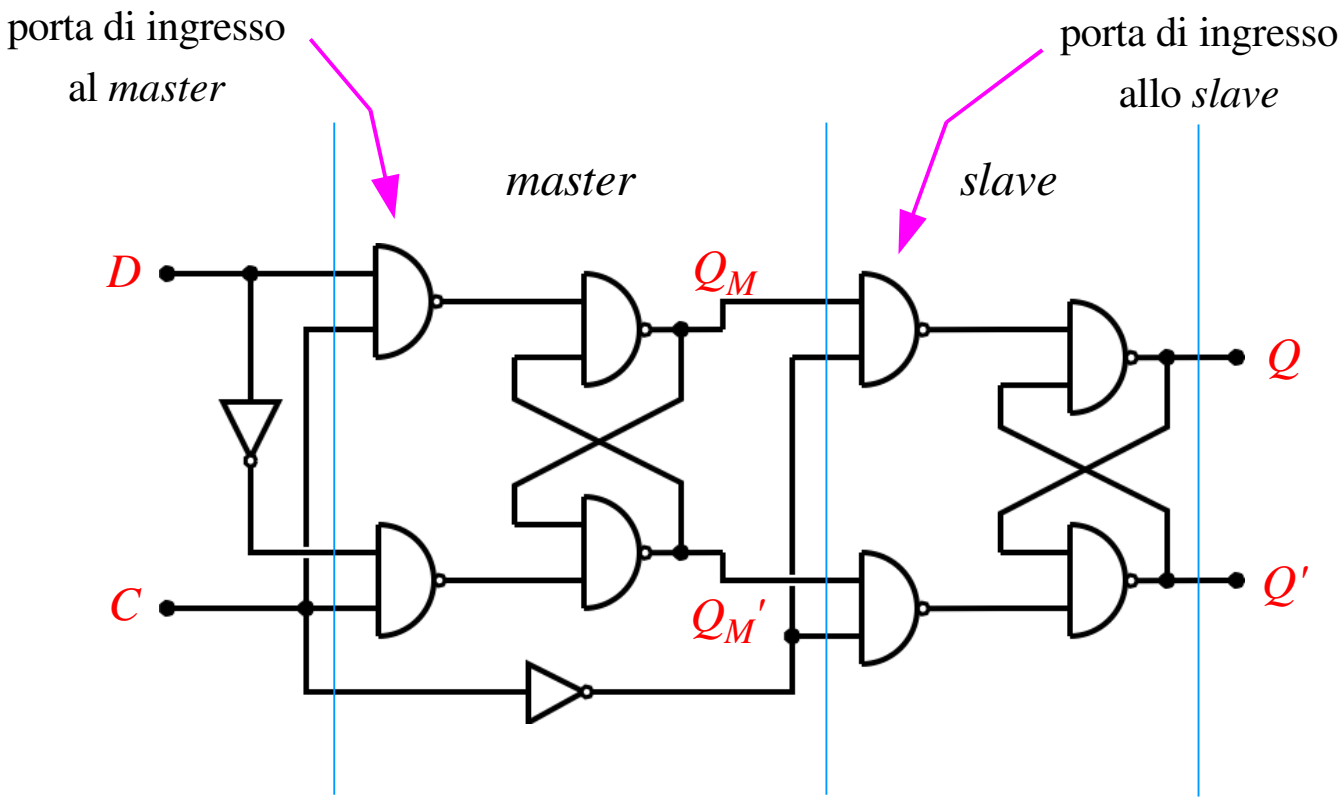
Porta di ingresso aperta.
L'ingresso *D* viene trasferito all'uscita *Q*.

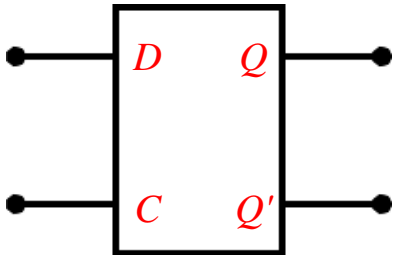
Porta di ingresso chiusa.
Il valore di *D* non ha alcun effetto su *Q*.



Il valore di Q alla chiusura della porta di ingresso rimane memorizzato.

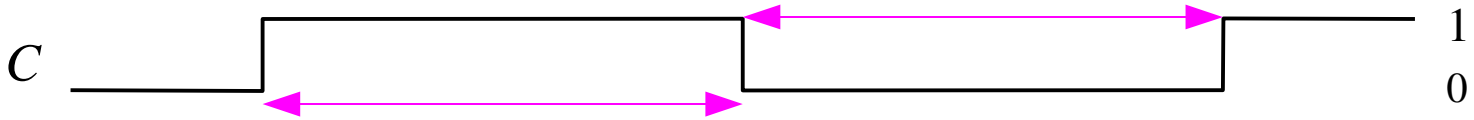
Master-Slave D-type flip-flop





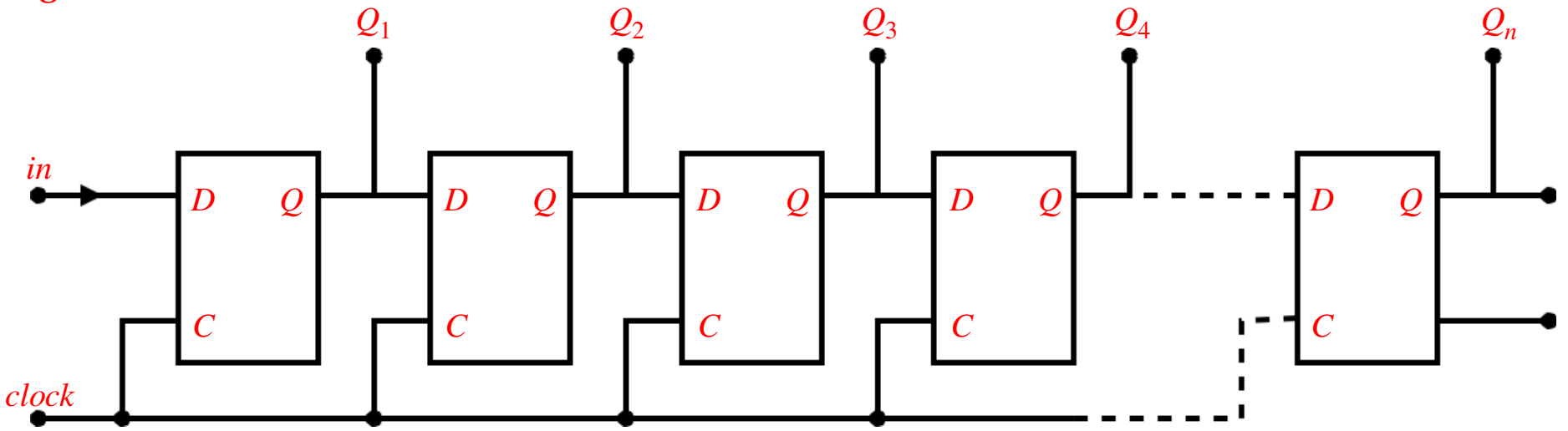
Simbolo del *flip-flop master-slave* di tipo *D*

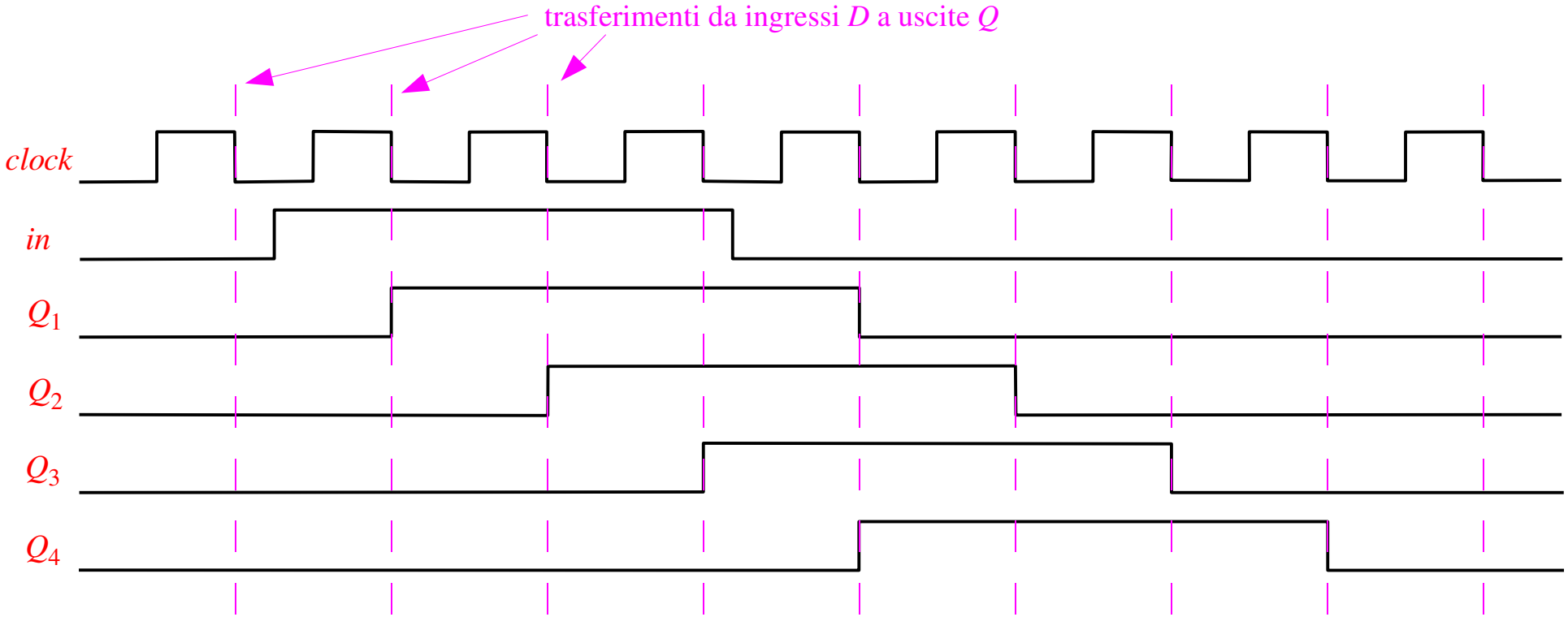
La porta di ingresso al *master* e' chiusa. I valori di Q_M e Q_M' vengono trasferiti sullo *slave*.



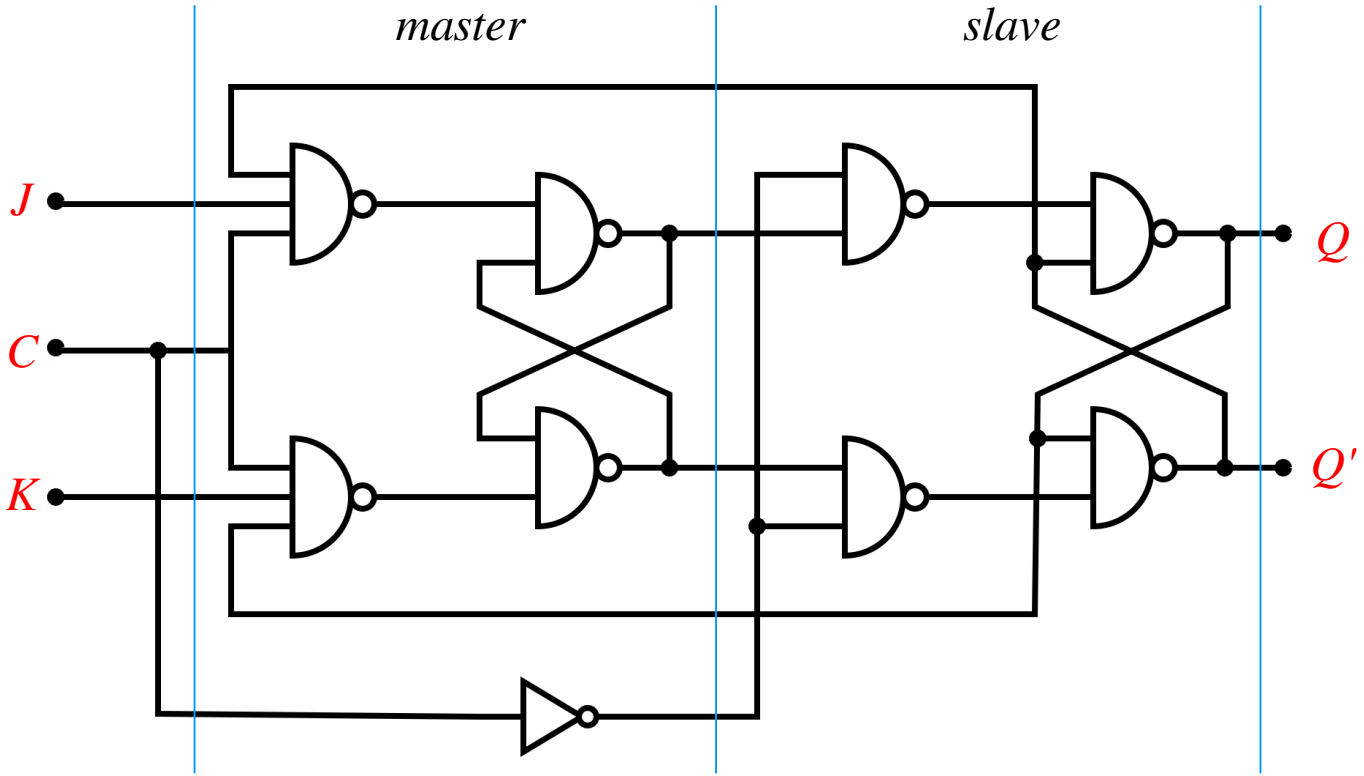
La porta di ingresso al *master* e' aperta. Le uscite Q_M e Q_M' seguono il dato *D*.

Shift register



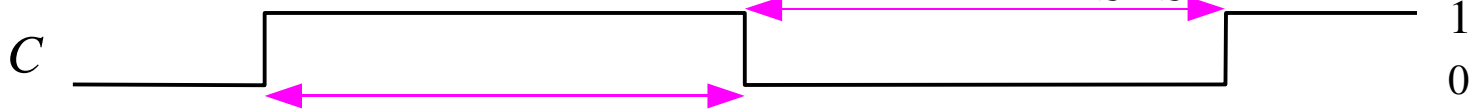


Master-Slave JK flip-flop



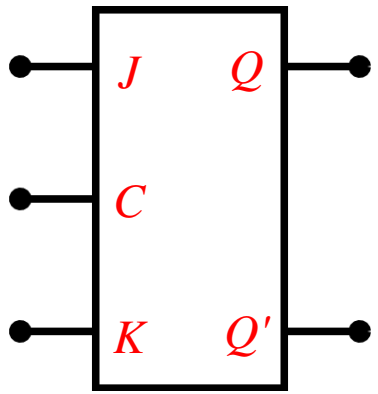
J	K	Q	Q'
0	0	Q_-	Q'_-
0	1	0	1
1	0	1	0
1	1	Q'_-	Q_-

Valori assunti dalle uscite Q e Q' dopo la transizione ↓ del *clock* C.



Il dato viene trasferito dal *master* allo *slave* e appare come nuova uscita su Q e Q'.

Il dato Q e Q' entra nel *master*, controllato dagli ingressi J e K.

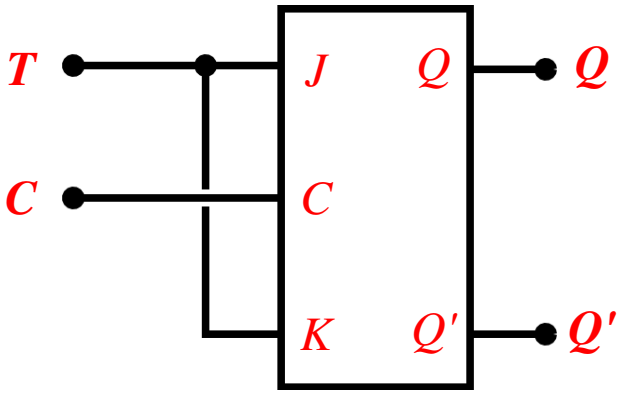


Rappresentazione schematica del *flip-flop* di tipo *JK*.

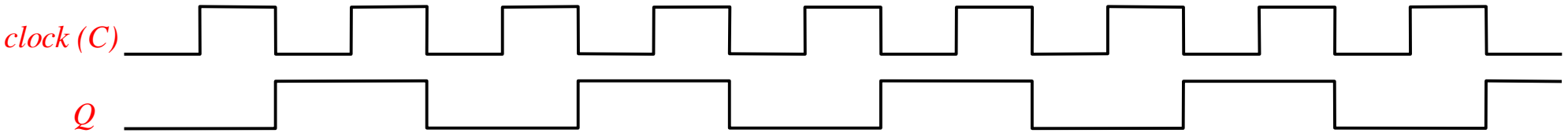
T type flip-flop - contatore

J	K	Q	Q'
0	0	Q_-	Q'_-
1	1	Q'_-	Q_-

Valori assunti dalle uscite Q e Q' dopo la transizione ↓ del *clock* C.



Il *flip-flop* di tipo JK con gli ingressi J e K collegati insieme diventa un *flip-flop* ti tipo T (*toggle*). Ad ogni transizione da 1 a 0 dell'ingresso C (*clock*) il valore delle uscite si inverte se T = 1, rimane invariato se T=0.

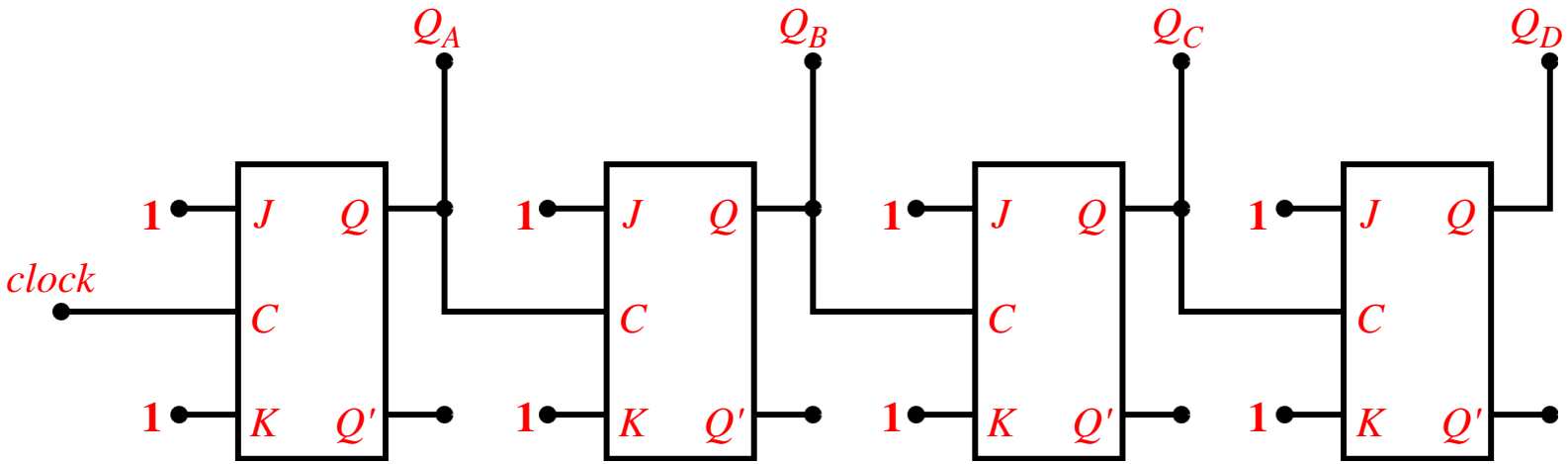


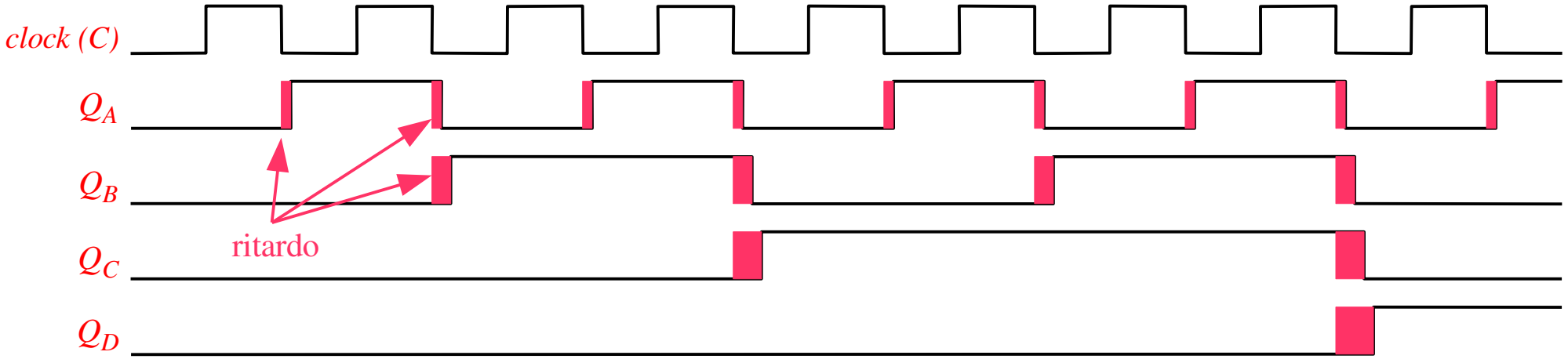
L'uscita Q compie un ciclo completo ogni due cicli dell'ingresso C. Il circuito e' un *divisore di frequenza* per 2.

Contatore binario (asincrono)

Collegando in cascata piu' *divisori* (*flip-flop* di tipo *T*) si ottiene un *contatore binario:* la configurazione delle uscite *Q* rappresenta il numero di impulsi giunti al primo ingresso (*clock*).

Q_A	Q_B	Q_C	Q_D
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	0
0	0	1	0
1	0	1	0
0	1	1	0
1	1	1	0
0	0	0	1





Tempi di propagazione – frequenza massima di conteggio

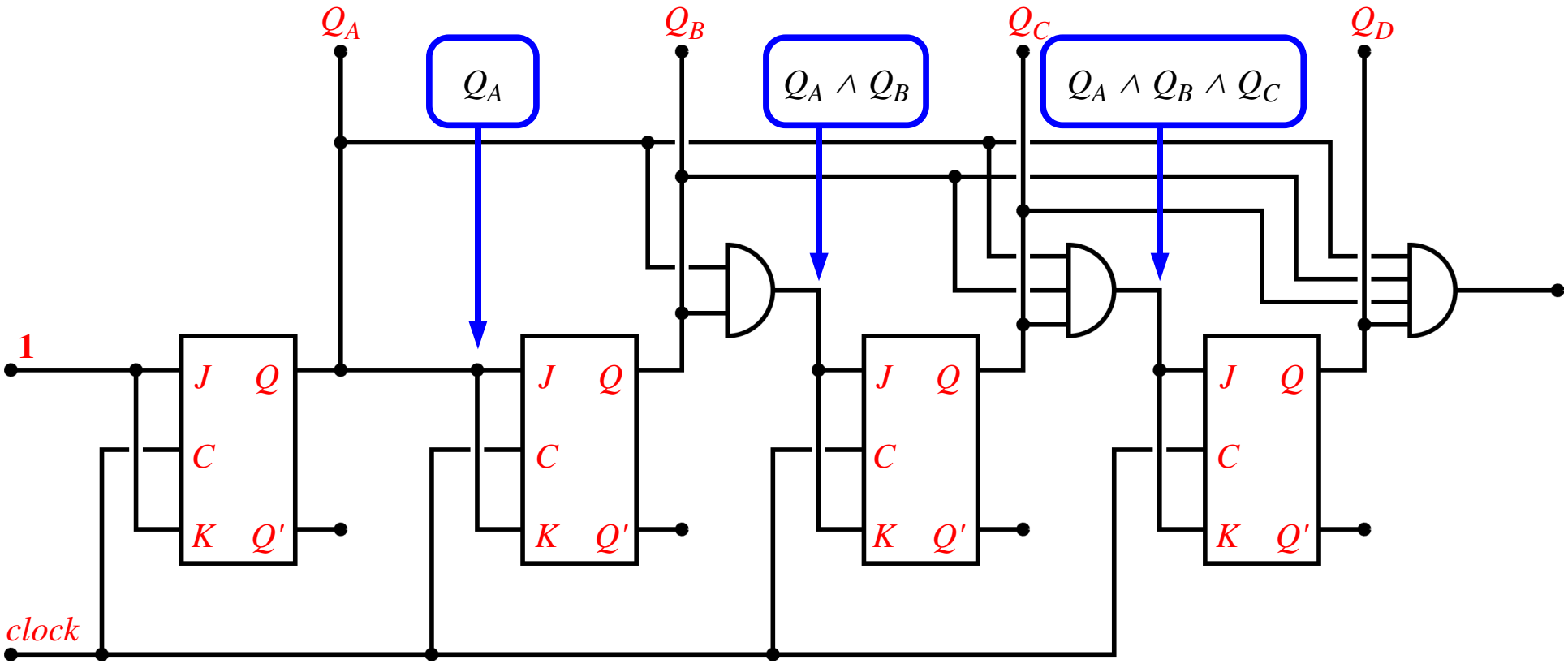
Nel contatore asincrono l'ingresso di ogni *flip-flop* e' comandato dall'uscita del precedente.
Il tempo di propagazione dei segnali dall'ingresso all'ultima uscita e' dato dalla somma dei tempi di propagazione attraverso ogni stadio.

Ad esempio, in un *flip-flop J-K* di tipo *TTL-LS*, il tempo di propagazione e' circa 18 *nsec*.
In una catena di 4 stadi tra il segnale all'ingresso *clock* e l'ultima uscita si ha un ritardo di 72 *nsec* e quindi una frequenza massima di conteggio minore di $(72\ nsec)^{-1} = 13.9\ Mhz$.

Si possono realizzare contatori sincroni in cui il segnale di ingresso (*clock*) e' applicato contemporaneamente ad ogni *flip-flop* della catena:

- ogni flip-flop esegue o meno il conteggio sulla base dei segnali presenti sugli ingressi *J*,*K* generati dalle uscite degli stadi precedenti;
- ogni *flip-flop* "*sa*" in anticipo se dovra' cambiare stato al successivo segnale di *clock* e di conseguenza quando questo arriva tutti gli stadi commutano contemporaneamente.

Contatore sincrono



Schema di principio di un contatore sincrono a quattro bit

Ogni flip-flop commuta al successivo passaggio del *clock* da 1 a 0 se il suo ingresso *JK* e' 1.

Il tempo di propagazione dal segnale di *clock* ad ogni uscita e' quello di commutazione del singolo *flip-flop*.