Capitolo 6

Elettronica digitale

In *elettronica analogica* le variabili elettriche, tensioni e correnti, in un circuito sono funzioni continue, non necessariamente lineari, delle grandezze fisiche che rappresentano; nei circuiti elettronici digitali invece le variabili elettriche assumono solo due valori discreti, indicati convenzionalmente con una coppia di simboli come 0/1 oppure *Falso/Vero* (*False/True*), *Basso/Alto* (*Low/High*), *Spento/Acceso* (*Off/On*), ecc. (fig. 6.1). Una variabile digitale elementare, con il suo valore 0/1, prende il nome di variabile logica o variabile binaria o *bit* (*bit* = **bi***nary digi***t**).

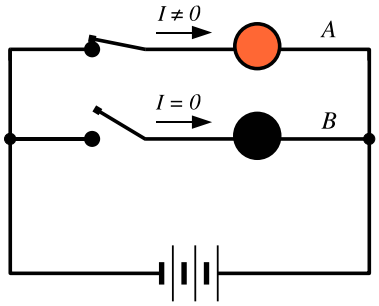


Figura 6.1: Circuiti digitali: il circuito A è *acceso* (circola corrente), il circuito B è *spento* (non circola corrente).

In elettronica digitale un segnale è rappresentato da un certo numero di bit che ne esprimono il valore secondo un opportuno codice. Nella tabella 6.1 sono riportati i primi 16 elementi del codice di numerazione binaria mediante il quale è possibile rappresentare, utilizzando N bit, 2^N valori diversi, ad esempio i primi 2^N numeri interi.

Al confine tra il mondo dell'elettronica analogica e quello digitale vi sono i *Convertitori Analogico-Digitale (ADC)* e *Digitale-Analogico (DAC)*. Gli *ADC* hanno come ingresso un singolo segnale analogico e come uscita un certo numero di bit che, dopo la conversione, esprimono il valore numerico del segnale di ingresso; i convertitori *DAC* svolgono la funzione inversa: impostando sui bit che costituiscono le variabili digitali di ingresso un codice numerico, il *DAC* sintetizza la tensione (o corrente) di valore corrispondente (fig. 6.2).

La conversione da analogico a digitale è il destino finale di ogni misura di una

valore decimale	codice binario 2^3	2^2	2^1	2^0	valore decimale	codice binario 2^3	2^2	2^1	2^0
0	0	0	0	0	8	1	0	0	0
1	0	0	0	1	9	1	0	0	1
2	0	0	1	0	10	1	0	1	0
3	0	0	1	1	11	1	0	1	1
4	0	1	0	0	12	1	1	0	0
5	0	1	0	1	13	1	1	0	1
6	0	1	1	0	14	1	1	1	0
7	0	1	1	1	15	1	1	1	1

Tabella 6.1: Numeri interi da 0 a 15 e corrispondente codice binario.

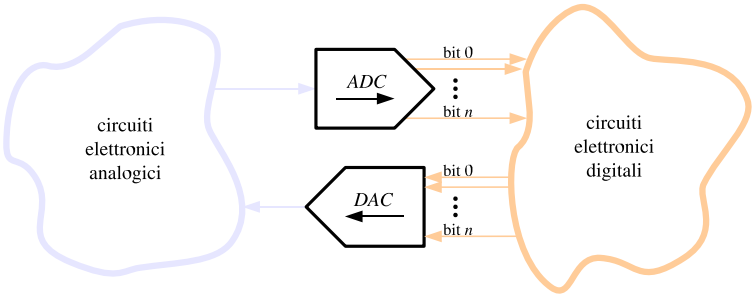


Figura 6.2: I convertitori da *Analogico a Digitale* (*ADC*) e da *Digitale ad Analogico* (*DAC*) costituiscono il legame tra le variabili continue delle reti analogiche ed i valori *acceso*/*spento* dei bit delle reti digitali.

grandezza fisica analogica. Questo può avvenire tardi, alla fine del processo di misura, quando un operatore legge sulla scala numerata di un quadrante un valore e lo riporta in forma numerica su un quaderno di appunti, oppure può avvenire precocemente all'interno degli stessi strumenti di misura, ad opera di un *ADC* (fig. 6.3).

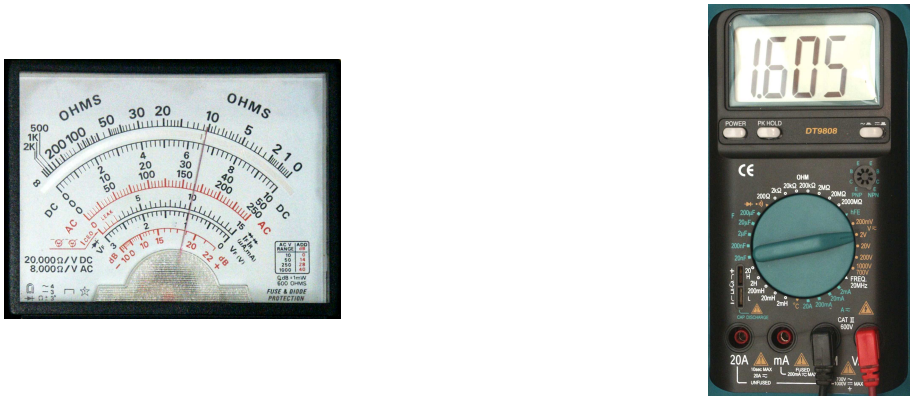


Figura 6.3: Strumenti di misura con lettura analogica e digitale.

Una volta che un segnale è disponibile in forma digitale è possibile elaborarlo tramite circuiti elettronici digitali (reti logiche) in grado di eseguire algoritmi logico-

matematici.

Gli elaboratori elettronici (*computer*) sono oggi le più complesse e versatili reti di elaborazione digitale che siano state sviluppate.

6.1 Algebra di Boole

L'Algebra di Boole definisce tre operazioni *logiche* elementari che possono essere eseguite sulle variabili binarie. Tramite queste tre operazioni (in realtà si vedrà che una è addirittura ridondante) è possibile realizzare algoritmi logico-matematici come la somma, il confronto tra due numeri binari, ecc. Combinando in maniera opportuna molte operazioni elementari è possibile arrivare allo sviluppo di algoritmi sempre più complessi. In fin dei conti, il calcolo numerico insegna che la moltiplicazione si può ridurre ad una serie di addizioni, che un integrale può essere calcolato con una serie di somme e moltiplicazioni sino al grado voluto di precisione, ecc.

Quindi se si è in grado di costruire circuiti elettronici in grado di effettuare le operazioni logiche elementari, allora è possibile costruire reti digitali contenenti tanti di questi circuiti elementari (spesso veramente tanti!) in grado di elaborare la soluzione di qualsiasi problema che possa essere espresso tramite un algoritmo logico-matematico.

6.1.1 Operazioni logiche elementari

Le tre operazioni logiche elementari sono convenzionalmente identificate dai termini inglesi AND, OR e NOT ed indicate nelle espressioni logiche con i simboli, rispettivamente, $\wedge, \vee, \sim$. Le operazioni AND e OR agiscono su due variabili elementari (due bit), dando un terzo bit come risultato, secondo le definizioni seguenti:

$C = A \wedge B$: la variabile C è 1 se A **e** B sono entrambe 1.
$C = A \vee B$: la variabile C è 1 se A **oppure** B è 1.

La terza operazione, NOT, ha un solo bit come operando e dà un bit come risultato:

$C = \sim A$: la variabile C è 1 se A è 0; la variabile C è zero se A è 1.

Un altro modo di descrivere le operazioni AND, OR, NOT è tramite le *tavole di verità*, riportate in fig. 6.4. Sotto ogni operazione è riportato anche il simbolo utilizzato negli schemi delle reti digitali per indicare la *porta*, cioè il circuito logico elementare, che effettua l'operazione.

6.1.2 Circuiti logici elementari (*porte*)

I circuiti logici elementari possono essere realizzati utilizzando diverse *tecnologie*. Per esempio, in fig. 6.5 le operazioni AND, OR, NOT sono state realizzate mediante interruttori meccanici.

Nella descrizione delle operazioni contenuta nella figura è stata implicitamente utilizzata la corrispondenza 0 = *spento* e 1 = *acceso*, indicata convenzionalmente come *logica positiva*. Anche se siamo portati istintivamente ad utilizzare questa

A	B	$A \wedge B$
0	0	0
0	1	0
1	0	0
1	1	1

A	B	$A \vee B$
0	0	0
0	1	1
1	0	1
1	1	1

A	$\sim A$
0	1
1	0

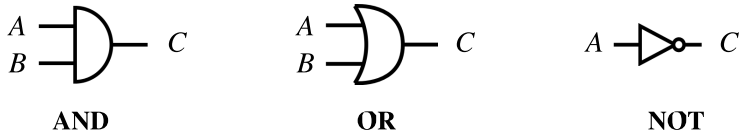
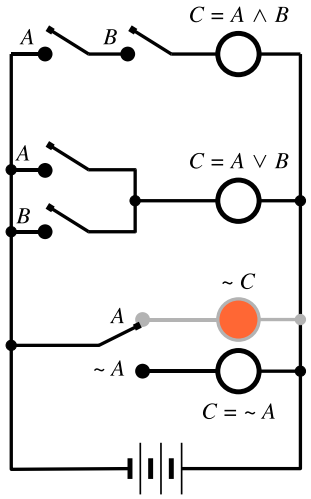


Figura 6.4: Descrizione delle operazioni logiche elementari tramite le *tavole di verità* e simboli con cui le operazioni vengono rappresentate negli schemi delle reti digitali.



AND: La lampadina *C* è accesa se l'interruttore *A* è chiuso **e** interruttore *B* è chiuso.

OR: La lampadina *C* è accesa se l'interruttore *A* è chiuso **oppure** interruttore *B* è chiuso.

NOT: La lampadina *C* è accesa se l'interruttore *A* **non** è chiuso.

Figura 6.5: Le operazioni logiche elementari effettuate tramite interruttori elettrici.

associazione, la *logica negativa* opposta (0 = *acceso* e 1 = *spento*) è altrettanto legittima.

6.1.3 Le leggi di De Morgan

Le tavole di verità delle operazioni AND e OR interpretate secondo le due logiche sono riportate nella tabella 6.2.

Si vede che una operazione AND secondo la logica positiva corrisponde ad una operazione OR quando i valori alle variabili vengono assegnati secondo la logica negativa e viceversa.

Queste equivalenze costituiscono le *Leggi di De Morgan*:

$$\sim A \wedge \sim B = \sim (A \vee B) \tag{6.1}$$

$$\sim A \vee \sim B = \sim (A \wedge B) \tag{6.2}$$

Operazione AND

stato del circuito			logica positiva			logica negativa		
A	*B*	*C*	*A*	*B*	*C*	*A*	*B*	*C*
spento	*spento*	*spento*	0	0	0	1	1	1
spento	*acceso*	*spento*	0	1	0	1	0	1
acceso	*spento*	*spento*	1	0	0	0	1	1
acceso	*acceso*	*acceso*	1	1	1	0	0	0

Operazione OR

stato del circuito			logica positiva			logica negativa		
A	*B*	*C*	*A*	*B*	*C*	*A*	*B*	*C*
spento	*spento*	*spento*	0	0	0	1	1	1
spento	*acceso*	*acceso*	0	1	1	1	0	0
acceso	*spento*	*acceso*	1	0	1	0	1	0
acceso	*acceso*	*acceso*	1	1	1	0	0	0

Tabella 6.2: Tavole di verità delle operazioni AND e OR interpretate secondo la logica positiva e negativa.

oppure

$$A \wedge B = \sim (\sim A \vee \sim B) \tag{6.3}$$

$$A \vee B = \sim (\sim A \wedge \sim B) \tag{6.4}$$

Quindi le due operazioni AND e OR non sono indipendenti. Qualunque funzione logica potrebbe essere ottenuta combinando soltanto AND e NOT oppure OR e NOT (fig. 6.6).

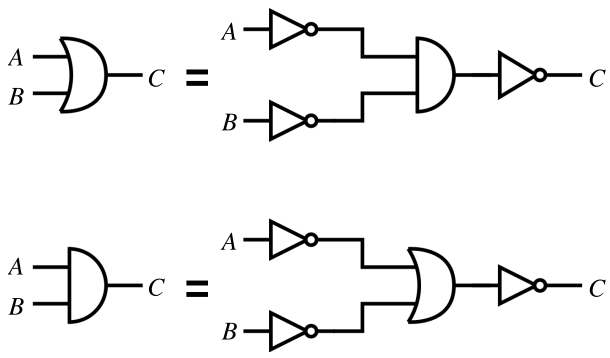


Figura 6.6: Leggi di De Morgan: l'operazione OR può essere ottenuta combinando AND e NOT e viceversa.

6.1.4 *NAND* e *NOR*

Le combinazioni delle due operazioni *AND-NOT* e *OR-NOT* che compaiono in fig. 6.6 sulla destra sono, come vedremo, le vere funzioni elementari delle reti digitali elettroniche, tanto da avere un proprio nome (*NAND* e *NOR*) ed un simbolo (fig. 6.7). In genere l'operazione *NOT* associata ad un'altra operazione viene riportata

Figura 6.7: Le operazioni elementari realizzate mediante circuiti elettronici digitali sono *NAND* e *NOR*.

negli schemi con il solo cerchietto invece che con il simbolo completo.

6.2 Il transistor come interruttore

Nei circuiti elettronici digitali sono i transistor, sia bjt che fet, che svolgono la funzione degli interruttori di fig. 6.5.

In fig. 6.8 è riportato uno schema di principio di un circuito *NOT* realizzato con un transistor bjt; a destra è riportata la funzione di trasferimento $V_{out}(V_{in})$.

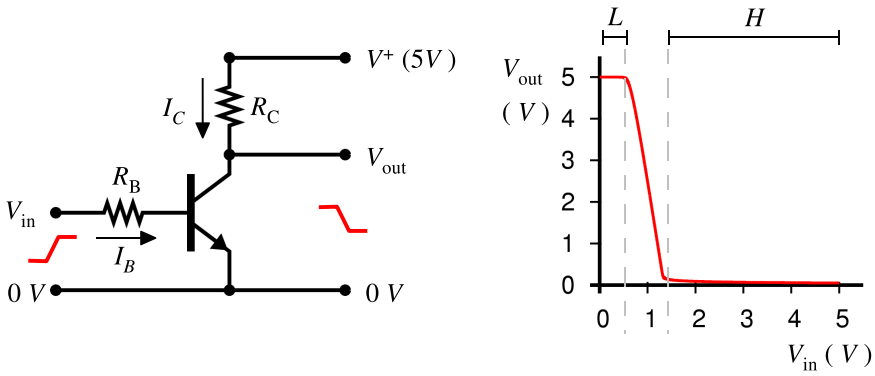


Figura 6.8: Circuito *NOT* realizzato con un transistor bjt.

Sino a che la tensione di ingresso V_{in} rimane nell'intervallo di valori indicato con L (*Low*, $0 \ldots V_\gamma$), si ha corrente I_B (e quindi I_C) praticamente nulla; la tensione di uscita V_{out} rimane al valore V^+ ed il transistor è spento. Quando V_{in} supera V_γ, comincia a circolare una corrente di base $I_B = (V_{in} - V_\gamma)/R_B$ e una corrente di collettore $I_C = I_B \cdot h_{FE}$; la tensione $V_{out} = V^+ - I_C \cdot R_C$ scende rapidamente, il transistor attraversa la regione attiva ed infine entra in saturazione. Per tutti i valori di V_{in} compresi nell'intervallo H (*High*), V_{out} rimane al di sotto di 200 mV.

In definitiva il circuito *NOT* non è altro che un amplificatore fatto lavorare al di fuori della sua caratteristica lineare.

Dal grafico si vede bene come i due stati digitali 0 e 1 (L e H) non sono due valori di tensione ben definiti, ma piuttosto due intervalli, separati da una zona di transizione che viene solo attraversata rapidamente nel passaggio da uno stato logico all'altro.

È importante che tutti i valori possibili della tensione di uscita nei due stati logici siano riconosciuti in maniera non ambigua dal circuito di ingresso di un'altra porta logica con le stesse caratteristiche. Questo permette di realizzare *famiglie* di dispositivi logici che possono essere direttamente interconnessi tra di loro per ottenere funzioni logiche più complesse.

Ogni famiglia è caratterizzata da un proprio insieme di valori delle caratteristiche elettriche corrispondenti ai due stati logici.

6.3 Le famiglie logiche

Nella figura 6.9 è riportata la foto di una scheda, risalente agli anni '60 del secolo scorso, contenente una parte dell'unità di calcolo di un computer e realizzata a *componenti discreti*, cioè singoli transistor, resistenze, condensatori e diodi.

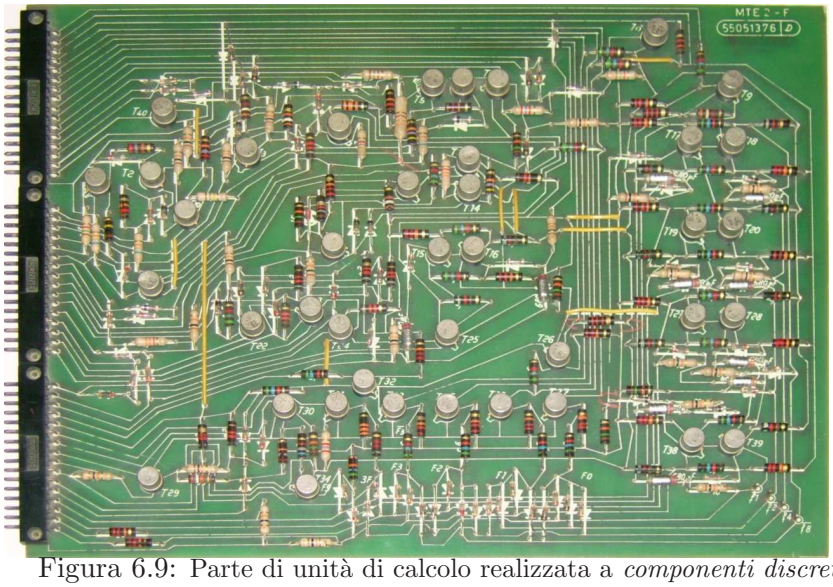


Figura 6.9: Parte di unità di calcolo realizzata a *componenti discreti*.

La scheda contiene complessivamente qualche decina di porte logiche elementari.

Con l'introduzione dei circuiti integrati è stato possibile costruire all'interno dello stesso *chip* di Silicio intere porte logiche e poi reti digitali via via più complesse, fino ad arrivare alla densità attuale di $2^6 \ldots 2^8$ porte, cioè un intero computer, in un singolo circuito integrato.

Nella tabella 6.3 è riportata l'evoluzione della complessità dei circuiti integrati digitali nel corso degli anni.

La foto in figura 6.10 riproduce l'intera unità di calcolo di un computer della serie PDP11, in produzione nei primi anni '80, realizzata con circuiti integrati sia ad alta che a bassa integrazione.

anno di riferimento	livello di integrazione		circuiti elementari per *chip*
1950		circuiti a componenti discreti	
1960	SSI	Small Scale integration	< 10
1966	MSI	Medium Scale integration	$10 \div 10^2$
1969	LSI	Large Scale integration	$10^2 \div 10^3$
1975	VLSI	Very Large Scale integration	$10^3 \div 10^6$
1990	ULSI	Ultra Large scale integration	$> 10^6$

Tabella 6.3: Evoluzione della complessità dei circuiti integrati digitali a partire dalla introduzione del transistor.

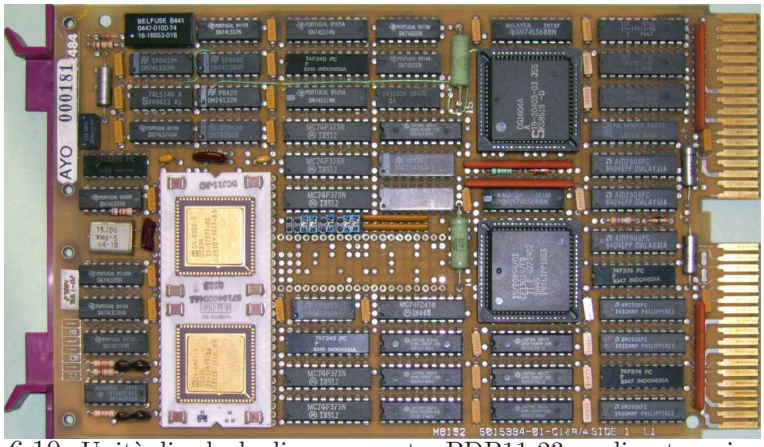


Figura 6.10: Unità di calcolo di un computer PDP11-23, realizzata a circuiti integrati sia a bassa che ad alta integrazione.

L'evoluzione dei circuiti integrati ha portato nel corso degli anni allo sviluppo di diverse soluzioni circuitali e tecnologiche per la realizzazione delle porte logiche elementari *AND*, *OR*, *NOT* (o meglio *NAND*, *NOR*, *NOT*) ed una varietà di funzioni più o meno complesse ottenute dalla combinazione di più porte elementari.

Alcune delle principali *famiglie* di porte e funzioni logiche oggi in uso sono riportate nella tabella 6.4; si possono tutte raggruppare intorno a tre tecnologie principali, che fanno uso di transistor bipolari (famiglia *TTL* e sue discendenti e famiglia *ECL*), di transistor mosfet complementari (famiglia *CMOS* e sue discendenti) o misti bipolari e mosfet (famiglia *BiCmos* e discendenti).

Per ogni famiglia è riportato anche lo *stato evolutivo*, cioè la situazione riguardo allo sviluppo, affermazione nelle applicazioni e declino.

Tra una famiglia e l'altra di una stessa soluzione tecnologica vi sono differenze nelle prestazioni, come maggiore o minore velocità o consumi, differenti tensioni di alimentazione, diversa capacità di pilotaggio di altri circuiti, ecc. Ma le caratteristiche circuitali di base di tutte le famiglie di una stessa tecnologia sono essenzialmente le stesse.

Esistono altre famiglie non riportate nella tabella ma comunque in uso ed anche molte famiglie ormai completamente scomparse dalla produzione e dall'uso corrente e di cui si può trovare traccia solo in apparecchiature e manuali sufficientemente vecchi.

TTL - Famiglie logiche a transistor bipolari		
TTL	Transistor-Transistor Logic	obsoleta
S	Schottky Logic	estinta
LS	Low-Power Schottky Logic	in declino
AS	Advanced Schottky Logic	in declino
F	Fast Logic	in declino
ALS	Advanced Low-Power Schottky Logic	in declino
ECL	Emitter Coupled Logic	in declino
CMOS - Famiglie logiche a transistor mosfet complementari		
C	CMOS Logic	estinta
CD4000	CMOS Logic	in declino
HC	High-Speed CMOS Logic	di uso corrente
HCT	High-Speed CMOS Logic	di uso corrente
AC	Advanced CMOS Logic	di uso corrente
ACT	Advanced CMOS Logic	di uso corrente
FCT	Fast CMOS Technology	di uso corrente
AHC	Advanced High-Speed CMOS	di uso corrente
AHCT	Advanced High-Speed CMOS	di uso corrente
ALVC	Advanced Low-Voltage CMOS Technology	in crescita
BiCMOS - Famiglie logiche a tecnologia mista bipolari+mosfet		
BCT	BiCMOS Technology	di uso corrente
ABT	Advanced BiCMOS Technology	di uso corrente
LVT	Low-Voltage BiCMOS Technology	di uso corrente

Tabella 6.4: Alcune delle principali famiglie logiche e loro *stato evolutivo*.

6.3.1 Famiglia *CMOS*

Anche se storicamente la famiglia *CMOS* è nata diversi anni dopo l'affermazione della famiglia *TTL*, la sua enorme importanza attuale e la notevole semplicità di principio dei suoi circuiti suggeriscono di analizzarla per prima.

Il circuito base della tecnologia *CMOS* è l'*inverter* (porta *NOT*) riportato in fig. 6.11. Invece di avere un transistor ed una resistenza di carico, come nell'esempio di

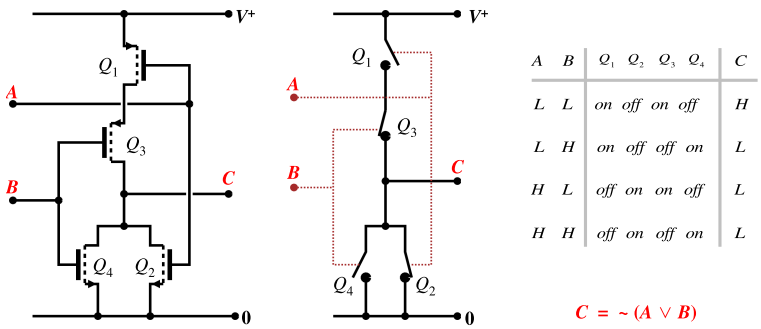


Figura 6.11: Porta *NOT* in tecnologia *CMOS*.

fig. 6.8, il circuito è composto da due transistor mosfet complementari (Q_1 è un *pmos*, Q_2 è un *nmos*) che lavorano in maniera simmetrica, in modo che ciascuno faccia da carico all'altro. Da questo il nome dato alla famiglia: CMOS = *Complementary MOSfet*.

Normalmente i due transistor sono realizzati in modo da avere caratteristiche elettriche eguali, a parte gli opportuni segni. Quindi, ad esempio, $V_{th_1} = -V_{th_2}$ ecc. In queste condizioni è facile vedere che quando Q_1 è spento Q_2 è acceso e viceversa. I due transistor si comportano quindi come i due interruttori accoppiati e contrapposti in fig. 6.11 a destra, per cui l'uscita C si trova collegata alternativamente con la massa attraverso Q_2, quando l'ingresso A è *Alto* ($V_{in} = V^+$) o con l'alimentazione V^+ attraverso Q_1 quando l'ingresso A è *Basso* ($V_{in} = 0$).

Questo circuito ha diversi vantaggi rispetto all'esempio di principio di fig. 6.8, che farebbe uso di un transistor e di una resistenza di carico:

- In condizioni statiche la dissipazione di energia è sempre nulla, sia con l'uscita *Alta* che con l'uscita *Bassa*, in quanto uno dei due transistor è spento e quindi non si ha passaggio di corrente da alimentazione V^+ a massa attraverso i due transistor. Nell'altro circuito solo nello stato di uscita *Alta* (transistor spento) si ha dissipazione nulla; nell'altro stato viene dissipata una potenza $P \simeq {V^+}^2/R_C$.

- L'uscita è sempre collegata ad un potenziale di riferimento (V^+ o massa) attraverso la bassa impedenza del transistor acceso.

- I livelli elettrici relativi agli stati *Alto* e *Basso* sono simmetrici.

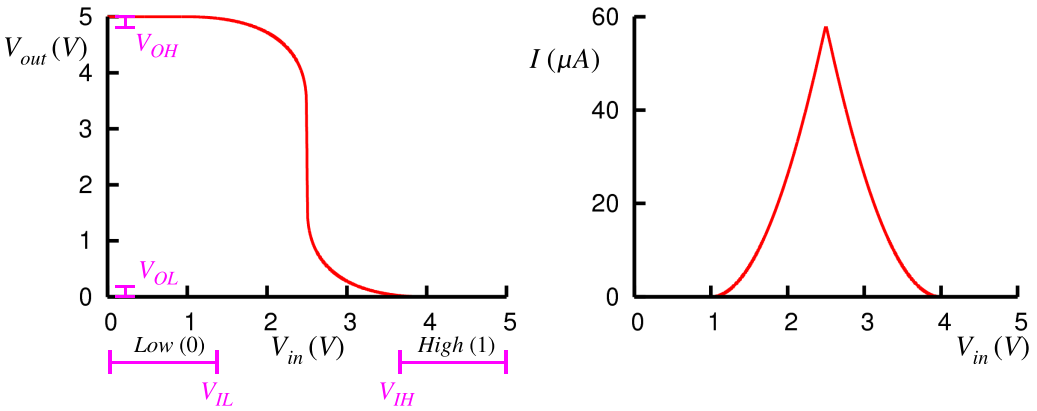


Figura 6.12: A sinistra: relazione tra tensione di ingresso V_{in} e tensione di uscita V_{out} nella porta *NOT* in tecnologia *CMOS*. A destra: corrente attraverso i transistor durante la commutazione.

La figura 6.12 riporta la relazione tra tensione V_{in} all'ingresso A e tensione V_{out} all'uscita C per una porta con due transistor complementari con $V_{th} = \pm 1.0\ V$, $K_n = 50\ \mu A/V^2$, $\lambda = 0.01\ V^{-1}$ e tensione di alimentazione $V^+ = 5\ V$.

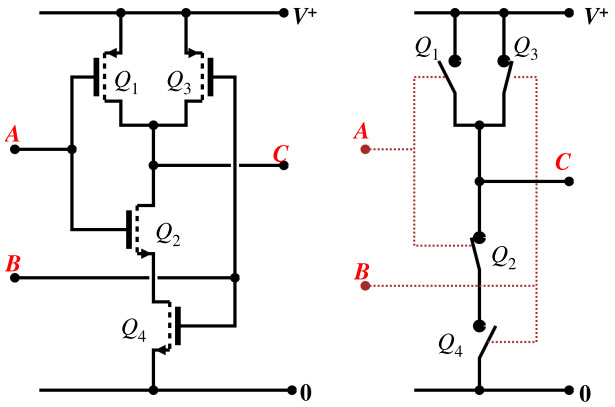
I valori di tensione indicati in viola sono:

V_{IL} = massima tensione di ingresso considerata come 0 (*Low*)
V_{IH} = minima tensione di ingresso considerata come 1 (*High*)
V_{OL} = massima tensione di uscita quando l'uscita è 0 (*Low*)
V_{OH} = minima tensione di uscita quando l'uscita è 1 (*High*)

Gli intervalli di tensione $0 \ldots V_{IL}$ e $V_{IH} \ldots V^+$ corrispondono ai valori di ingresso riconosciuti rispettivamente come 0 e 1, mentre gli intervalli $0 \ldots V_{OL}$ e $V_{OH} \ldots V^+$ rappresentano i valori *garantiti* per la tensione di uscita negli stati 0 e 1.

6.3.2 Porta $NAND$ e NOR in tecnologia CMOS

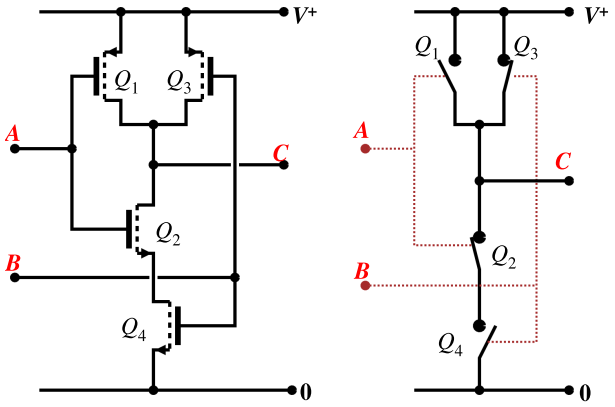
Raddoppiando i transistor-interruttori della fig. 6.11 come riportato negli schemi di fig. 6.13 e 6.14 dalla porta NOT si ottiene una porta $NAND$ oppure una porta NOR.



A	B	Q_1	Q_2	Q_3	Q_4	C
L	L	on	off	on	off	H
L	H	on	off	off	on	H
H	L	off	on	on	off	H
H	H	off	on	off	on	L

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

Figura 6.13: Porta $NAND$ della famiglia CMOS.



A	B	Q_1	Q_2	Q_3	Q_4	C
L	L	on	off	on	off	H
L	H	on	off	off	on	H
H	L	off	on	on	off	H
H	H	off	on	off	on	L

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

Figura 6.14: Porta NOR della famiglia CMOS.

Le tabelle a fianco degli schemi elettrici descrivono il funzionamento dei circuiti, riportando oltre alla tavola di verità della porta anche lo stato acceso/spento (on/off) di tutti i transistor in corrispondenza degli stati alto/basso (H/L o $1/0$) degli ingressi. Per *alto* in questa tabella ed in tutti i paragrafi che seguono si intende un valore di tensione positivo (nominalmente V^+), per basso si intende il valore nominale $0\ V$.

6.3.3 Famiglia *TTL*

In figura 6.15 è riportato lo schema della porta *NAND*, elemento base dei dispositivi logici realizzati con la tecnologia *TTL*.

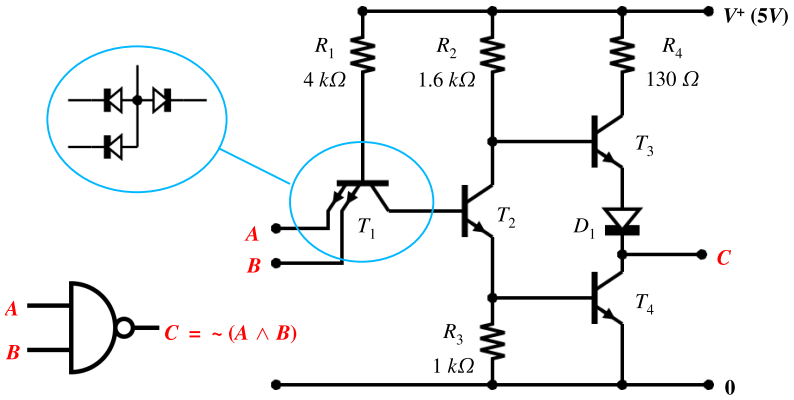


Figura 6.15: Porta *NAND* in tecnologia *TTL*.

L'aspetto del circuito è reso intricato dalla presenza di un transistor con doppio emettitore sugli ingressi A e B, in una configurazione che apparentemente non corrisponde a nessuna di quelle viste nel cap. 4. Conviene inizialmente considerare questo transistor come una rete di tre diodi connessi come indicato in alto a sinistra. Il transistor bjt è formato da due diodi contrapposti, come visto nel par. 4.1; realizzando la regione di emettitore come due regioni separate a contatto con un'unica regione di base si ottiene la configurazione di fig. 6.15.

Supponiamo inizialmente che i due ingressi A e B si trovino entrambi nello stato alto ($V_A = V_B = +5\ V$). Osservando il circuito è facile vedere che le due giunzioni base-emettitore di T_1 si troveranno contropolarizzate, mentre la giunzione base collettore di T_1 e le giunzioni base-emettitore di T_2 e T_4 si troveranno tutte e tre collegate in serie e direttamente polarizzate dalla tensione di alimentazione V^+ tramite la resistenza R_1. La corrente attraverso R_1, tra base e collettore di T_1 e nella base di T_2 è quindi $I_{R_1} = (V^+ - 3V_\gamma)/R_1$. Quindi i transistor T_2 e T_4 saranno entrambi accesi. I componenti del circuito, in particolare la resistenza R_2, sono dimensionati in modo che in queste condizioni il transistor T_2 si trovi non solo acceso, ma anche in saturazione. Quindi si avrà:

$$V_{E_2} = V_{BE_4} = V_\gamma \simeq 0.65\ V \tag{6.5}$$

$$V_{C_2} = V_\gamma + V_{CE_{sat}} = 0.7 \ldots 0.9\ V \tag{6.6}$$

Il transistor T_3 invece sarà spento. Infatti, a causa del diodo D_1 presente sul suo emettitore, per arrivare all'accensione è necessaria una tensione $V_{B_3} = V_{C_2}$ almeno superiore a $2V_\gamma \simeq 1.3\ V$, mentre si è visto che V_{C_2} non arriva ad $1\ V$. Il transistor T_3 spento si comporta come una resistenza di carico di valore infinito per T_4, che risulta quindi in saturazione. In definitiva all'uscita si avrà una tensione $V_C = V_{CE_{SAT}} \leq 0.4\ V$.

Collegando a massa, o ad un livello di tensione sufficientemente basso, uno o tutti e due gli ingressi A e B, la giunzione base emettitore di T_1 si viene a trovare in polarizzazione diretta e quindi $V_{B_1} \simeq V_\gamma$. Questa tensione non è più sufficiente a tenere in conduzione le tre giunzioni in serie V_{BC_1}, V_{BE_2} e V_{BE_4} per cui T_2 e T_4 si troveranno spenti. In queste condizioni il transistor T_3 si trova in configurazione *emitter follower* con la base collegata a V^+ tramite R_2. La tensione all'uscita sarà $V_C = V^+ - R_2 \cdot I_{B_3} - 2V_\gamma \simeq 3.7\ V$.

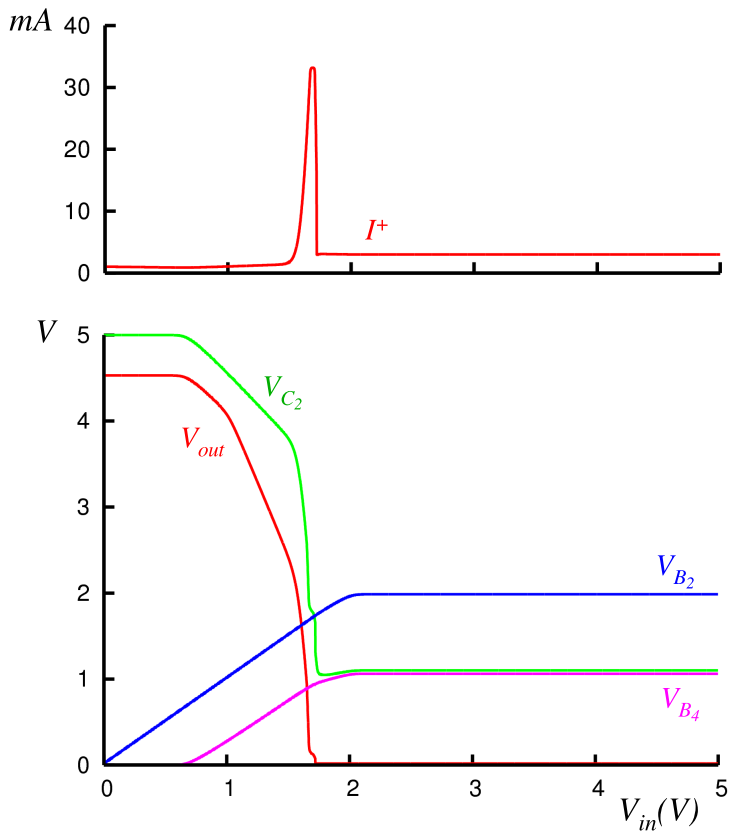


Figura 6.16: Tensioni in alcuni punti del circuito della porta *TTL* di fig. 6.15 e corrente di alimentazione.

Nella figura 6.16 è riportata la relazione tra tensione di ingresso e tensione di uscita per una porta *NAND* della famiglia *TTL*, insieme ai valori di tensione in alcuni nodi interni del circuito. La curva in rosso in alto rappresenta la corrente di alimentazione fornita dal generatore V^+ durante la commutazione. Per un breve intervallo della tensione di ingresso V_{in} intorno a $\simeq 1.6\ V$ i due transistor T_3 e T_4 dello stadio di uscita si trovano entrambi in forte conduzione; la resistenza R_4 ha lo scopo di limitare la corrente durante questo transitorio.

Anche questo circuito è dotato di uno stadio di uscita contenente due transistor che lavorano in maniera complementare, come visto per le porte *CMOS*. In questo caso, però, i due transistor sono della stessa polarità, *npn* e non complementari. Questa configurazione è denominata *totem pole*.

Negli istanti immediatamente successivi alla commutazione degli ingressi A e B

da 1 a 0, a causa della capacità di diffusione della base di T_2 che rimane carica per un breve istante ad una tensione positiva, il transistor T_1 si viene a trovare in regione attiva e funziona effettivamente come transistor invece che semplicemente come rete di diodi. Questo rende più veloce la scarica della capacità di base di T_2 e quindi riduce i tempi di commutazione.

6.3.4 Caratteristiche elettriche di famiglie *CMOS* e *TTL*

Nella tabella 6.5 sono riportati i valori caratteristici dei parametri elettrici per alcune famiglie di dispositivi *CMOS* e *TTL*.

	TTL	*TTL-LS*	*ACT*	*AHC*		
V^+	5	5	5	5	V	tensione di alimentazione
V_{IL}	0.8	0.8	0.8	1.5	V	massima tensione di ingresso considerata *low*
V_{IH}	2.0	2.0	2.0	3.5	V	minima tensione di ingresso considerata *high*
V_{OL}	0.4	0.5	0.4	0.5	V	massima tensione di uscita nello stato *low*
V_{OH}	2.4	2.7	3.7	4.5	V	minima tensione di uscita nello stato *high*
I_{IL}	-1.6	-0.4	-0.001		mA	corrente di ingresso nello stato *low*
I_{IH}	0.04	0.02	0.001		mA	corrente di ingresso nello stato *high*
I_{OL}	16	8	24	8	mA	corrente di uscita massima nello stato *low*
I_{OH}	-0.4	-0.4	-24	-8	mA	corrente di uscita massima nello stato *high*
fan-out	10	20	40	40		ingressi equivalenti collegabili ad una uscita
NM_L	0.4	0.3	0.4	1	V	margine di rumore a livello *low*
NM_H	0.4	0.7	1.7	1	V	margine di rumore a livello *high*

Tabella 6.5: Principali caratteristiche elettriche di alcune famiglie *CMOS* e *TTL*. La famiglia *ACT*, di tipo *CMOS*, ha livelli elettrici compatibili con le famiglie *TTL*.

6.4 Logica combinatoria

Si definisce *logica combinatoria* quella realizzata con reti digitali in cui i valori delle variabili di uscita dipendono esclusivamente dai valori istantanei delle variabili di ingresso, a parte ovviamente i ritardi dovuti ai tempi di propagazione dei segnali ed ai tempi di risposta finiti dei circuiti. Per contro nella *logica sequenziale*, descritta nel paragrafo 6.6 e successivi, le variabili di uscita dipendono non solo dai valori istantanei delle variabili di ingresso, ma anche dalla storia precedente del circuito: la rete sequenziale è dotata di *memoria*.

6.4.1 Esempi di circuiti a logica combinatoria

Exclusive-*OR* (*XOR*)

La definizione di operazione *OR* stabilisce che l'uscita è 1 se uno dei due ingressi, o anche tutti e due, hanno valore 1. Esiste anche una definizione più stretta del-

l'operazione, che stabilisce che l'uscita è 1 quando uno o l'altro degli ingressi è 1, ma non tutti e due insieme. In questa versione l'operazione si chiama *exclusive-OR* (*OR esclusivo*) e viene indicata con il termine XOR. In fig. 6.17 è riportata una rete combinatoria che esegue l'operazione XOR, insieme con la tavola di verità ed il simbolo utilizzato negli schemi logici.

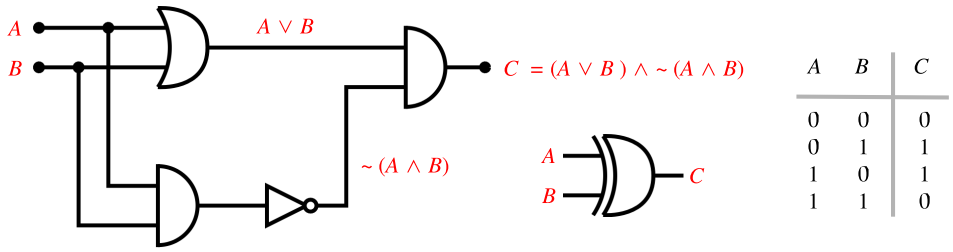


Figura 6.17: Operazione *exclusive-OR* (XOR). A sinistra: esempio di rete combinatoria per effettuare l'operazione; a destra: tavola di verità; al centro: simbolo con cui l'operazione viene rappresentata negli schemi.

Semiaddizionatore (*Half adder*)

L'operazione XOR, con una piccola modifica, permette di realizzare un addizionatore ad un bit, cioè una rete combinatoria che ha due ingressi e due uscite e fornisce sulle uscite in codice binario il valore della somma aritmetica dei valori dei due bit di ingresso. In pratica esegue il calcolo $0 + 0 = 0$, $0 + 1 = 1$, $1 + 0 = 1$, $1 + 1 = 0$ con riporto di uno.

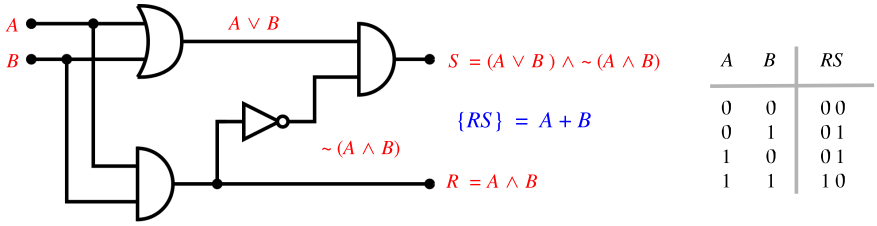
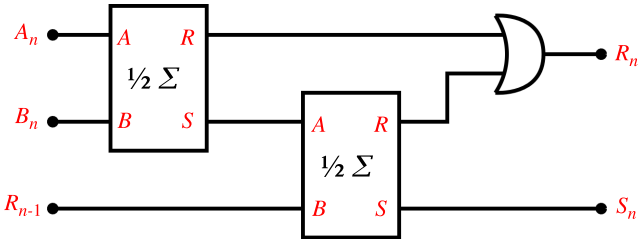


Figura 6.18: Circuito semi-addizionatore. Il codice binario formato dai due bit $\{RS\}$ rappresenta la somma aritmetica del valore dei due bit A e B.

In questa forma, il circuito prende il nome di semi-addizionatore (*half adder*). Infatti volendo effettuare la somma di numeri binari di lunghezza superiore ad un bit è necessario prevedere per ogni addizionatore un terzo ingresso per il riporto dall'addizionatore del bit immediatamente meno significativo.

Addizionatore (*Full adder*)

Un addizionatore completo può essere costruito mediante due semi-addizionatori ed una porta OR (fig 6.19). Ciascuno dei due rettangoli nella figura rappresenta un semi-addizionatore ($^1/_2\ \Sigma$) con i due ingressi A e B e le due uscite R e S come in fig. 6.18.



A_n	B_n	R_{n-1}	R_n	S_n
0	0	0	0	0
0	1	0	0	1
1	0	0	0	1
1	1	0	1	0
0	0	1	0	1
0	1	1	1	0
1	0	1	1	0
1	1	1	1	1

Figura 6.19: Circuito addizionatore (*full adder*). Con A_n e B_n si indicano i bit n-esimi di due numeri binari A e B; R_{n-1} è il riporto che viene dal circuito addizionatore del bit immediatamente meno significativo; S_n è il bit n-esimo della somma e R_n il riporto verso il bit immediatamente più significativo. Sulla destra la tavola di verità.

Osservando la tavola di verità si può vedere come i due bit R_n ed S_n, se letti come numero binario, contengono la somma aritmetica degli 1 presenti sui tre ingressi A, B ed R_{n-1}.

Addizionatore a 4 bit

Combinando insieme N circuiti addizionatori come quello di fig. 6.19 è possibile realizzare una rete combinatoria in grado di effettuare l'operazione di *somma aritmetica* di due variabili numeriche di N bit. Nella figura 6.20 è riportato un esempio per $N = 4$.

Il circuito è composto da una catena di quattro addizionatori *full adder* (Σ) da un bit. Ogni singolo addizionatore opera su un bit della variabile $A = \{A_3\ A_2\ A_1\ A_0\}$, sul bit corrispondente della variabile $B = \{B_3\ B_2\ B_1\ B_0\}$ e sul riporto che proviene dal bit alla sua destra, cioè immediatamente meno significativo. Ogni sezione del circuito genera un bit S_i della variabile somma $S = \{S_3\ S_2\ S_1\ S_0\}$ ed il riporto R_i per la sezione alla sua sinistra, cioè per il bit immediatamente più significativo.

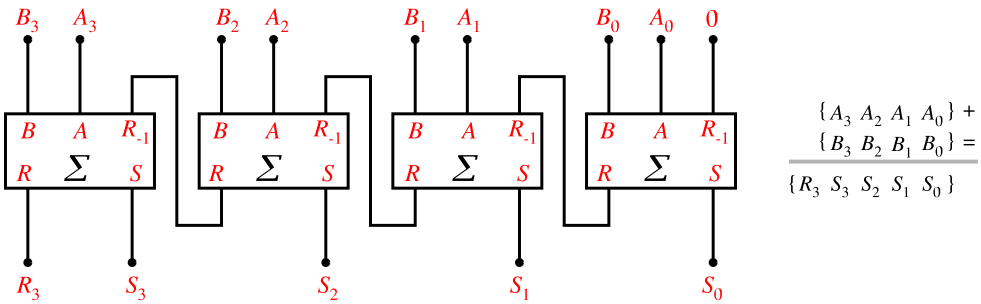


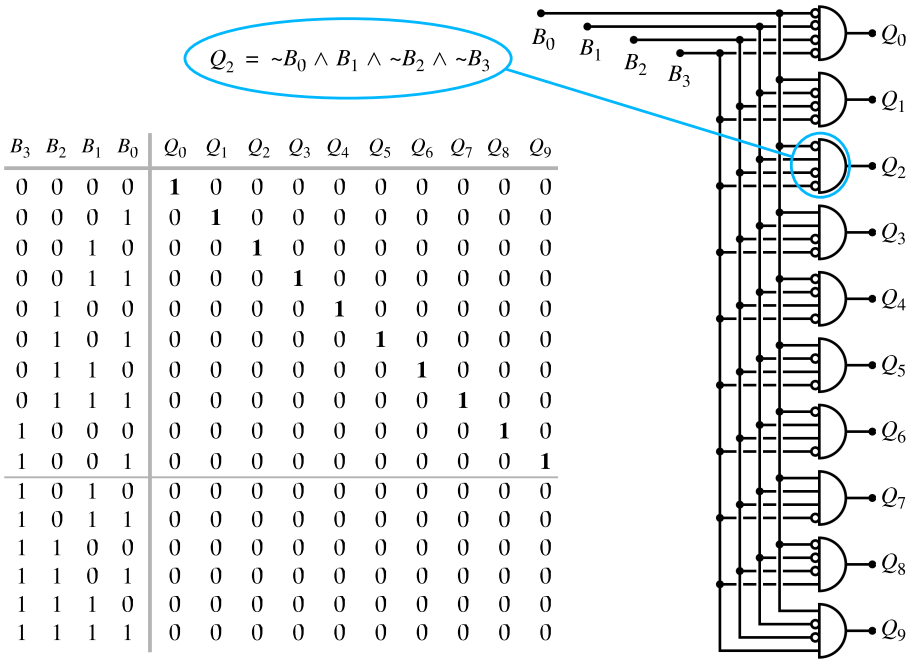
Figura 6.20: Circuito addizionatore a 4 bit.

L'ultimo riporto (R_3) costituisce il bit più significativo del risultato (in qualsiasi sistema di numerazione, la somma di due numeri di N cifre può dare un risultato fino a $N + 1$ cifre).

Convertitori di codice

Il circuito in fig. 6.21 è un convertitore da codice *BCD* a codice decimale. Il codice *BCD* (*Binary Coded Decimal*) è utilizzato per rappresentare in codice binario le cifre del sistema di numerazione decimale. Poichè con N bit è possibile avere 2^N combinazioni diverse, per codificare 10 elementi è necessario disporre di $N = \log_2 10 = 4$ bit (arrotondati per eccesso). Delle $2^4 = 16$ combinazioni possibili, dieci sono utilizzate per indicare le cifre da 0 a 9, le altre sei sono ridondanti.

Il circuito ha un ingresso a 4 bit ($B_3 \ldots B_0$) e dieci uscite. Solo una delle dieci uscite si può trovare nello stato 1, come indicato dalla tavola a sinistra nella figura, selezionando quella delle dieci cifre indicata dal codice binario di ingresso.



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**
1	0	1	0	0	0	0	0	0	0	0	0	0	0
1	0	1	1	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	1	0	0	0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0	0	0	0

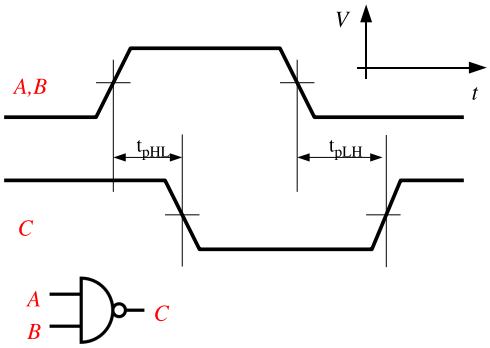
Figura 6.21: Convertitore da codice BCD a codice decimale.

6.5 Tempi di commutazione

La propagazione dei segnali tra ingressi e uscite di una rete combinatoria non è istantanea, ma richiede un tempo finito, dovuto principalmente alla velocità di risposta dei transistor delle porte che, per quanto elevata, è comunque sempre limitata. Per i dispositivi più veloci, anche il tempo di propagazione dei segnali lungo i collegamenti può diventare significativo (un collegamento della lunghezza di 2 *cm* introduce un ritardo di circa 0.1 *ns*).

Ogni porta di ogni famiglia logica ha i suoi tempi caratteristici di commutazione. Nella figura 6.22 sono riportati i tempi caratteristici della porta *NAND* per alcune famiglie *TTL* e *CMOS*. Il tempo di commutazione non è necessariamente lo stesso nelle transizioni da livello alto a livello basso e viceversa. Nella figura sono definiti due tempi: un tempo t_{pHL}, corrispondente alla commutazione $H \rightarrow L$ dell'uscita

in seguito al passaggio $L \rightarrow H$ dell'ingresso ed un tempo t_{pLH} corrispondente alla transizione opposta. Dalla tabella si vede che per alcune famiglie le differenze tra questi due tempi sono considerevoli.



Tempi di propagazione in *nsec*

	t_{pHL}		t_{pLH}	
	typ	max	typ	max
TTL	7	15	11	22
TTL-LS	10	9	9	15
TTL-F	3.2	4.3	3.7	5
ACT	4	7	5.5	9
AHC	3.8	5.5	3.8	5.5

Figura 6.22: Definizione dei tempi di propagazione dei segnali da ingressi A e B ad uscita C attraverso una porta *NAND* e valori numerici per alcune famiglie *TTL* e *CMOS*.

6.6 Logica sequenziale

Anche nelle reti logiche è possibile introdurre la reazione (feedback), come nelle reti analogiche. La figura 6.23 riporta un esempio di reazione negativa (a sinistra) e di reazione positiva (a destra).

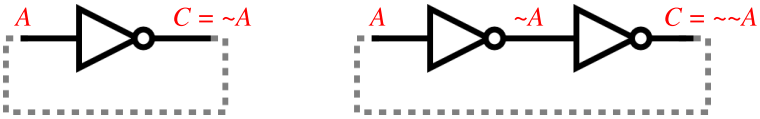


Figura 6.23: Due circuiti digitali con anello di reazione: negativa a sinistra, positiva a destra.

Da un punto di vista logico, chiudendo l'anello nel circuito a sinistra, si pone $A = C$; visto che per una porta *NOT* vale la relazione $C = \sim A$, si ottiene la relazione $A = \sim A$, che è una assurdità.

Il grafico in fig. 6.24 illustra quello che succede nel circuito. Nella figura è stata sovrapposta alla relazione di trasferimento tra ingresso e uscita della porta *NOT*, già vista in fig. 6.12, la relazione $V_{out} = V_{in}$ stabilita dal collegamento esterno. Il punto di incrocio tra le due curve corrisponde alla soluzione della equazione $V_{out}(V_{in}) = V_{in}$ ed indica i livelli assunti dalle tensioni, che non corrispondono a nessuno dei valori previsti per gli stati logici 0/1. Il circuito smette di funzionare come porta digitale e ritorna piuttosto ad un comportamento da circuito analogico.

È da notare che la condizione che si viene a stabilire corrisponde ad un punto di elevata dissipazione all'interno del dispositivo (in una porta CMOS, ad esempio, i transistor sono entrambi accesi ed al massimo della corrente che li attraversa). Per

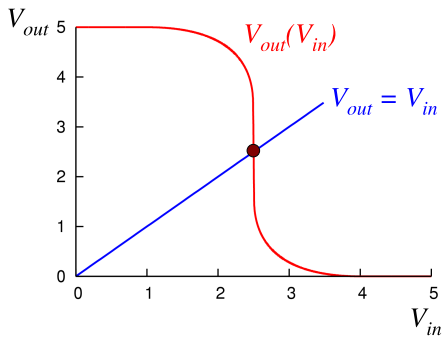


Figura 6.24: Una porta *NOT* con ingresso collegato all'uscita si comporta come un amplificatore con reazione negativa. Per la porta *NOT* CMOS si ha $V_{in} = V_{out} \simeq 2.5\ V$.

alcune famiglie questa può essere una condizione operativa non accettabile, in quanto produce riscaldamento eccessivo.

Molto più interessante dal punto di vista delle reti digitali è il circuito nella fig. 6.23, a destra.

Una volta chiuso l'anello il funzionamento del circuito è descritto dalla relazione $C = \sim\sim A = A$, relazione perfettamente valida. L'aspetto interessante è che la relazione è verificata sia per $A = 0$ sia per $A = 1$. In altre parole, uno stato 1 all'ingresso A genera una uscita $C = 1$, che genera, coerentemente, un ingresso $A = 1$. Risultato analogo si ottiene per $A = 0$.

Quindi sia che il circuito si trovi nello stato $A = C = 1$ che nello stato $A = C = 0$, vi rimane indefinitamente, o almeno fino a quando lo stato non viene modificato per effetto di una azione esterna.

Il circuito è quindi dotato di *memoria* ed è l'elemento base della logica *sequenziale*: i valori delle variabili di uscita di una rete sequenziale dipendono non solo dai valori attuali delle variabili di ingresso, ma anche dalla storia precedente della rete.

Per la sua caratteristica di avere due possibili stati stabili, il circuito è detto *bistabile*. Il termine corrispondente in lingua inglese è *flip-flop*.

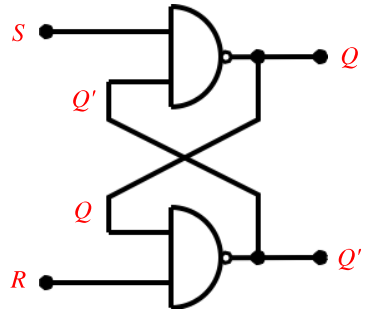
6.7 I *flip-flop*

6.7.1 *Set-Reset* (*SR*)

Il circuito bistabile decritto nell'introduzione precedente è solo di interesse illustrativo, in quanto non è provvisto di alcun terminale di ingresso tramite il quale sia possibile modificarne lo stato.

In figura 6.25 è riportato il circuito di uno dei più semplici flip-flop di interesse applicativo, dotato di due terminali di ingresso contrassegnati con le lettere S (*Set*) ed R (*Reset*), da cui il nome *set-reset* o *SR*.

La tabella di valori a destra illustra il funzionamento del circuito. Quando l'ingresso S oppure R hanno valore 0, l'uscita corrispondente, Q o Q' è forzata al valore 1, indipendentemente dal valore dell'altro ingresso e dell'altra uscita. Nello stato $S = 1$ ed $R = 1$, le due porte *NAND* si comportano come due porte *NOT* per i



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

Figura 6.25: Il flip-flop di tipo *set-reset*.

due segnali che provengono dalle uscite Q e Q' ed il circuito si comporta come quello dell'esempio di fig. 6.23, a destra. I due stati $\{Q = 0,\ Q' = 1\}$ e $\{Q = 1,\ Q' = 0\}$ sono entrambi due stati possibili e stabili.

Tramite i due ingressi S ed R è possibile modificare lo stato del flip-flop: portando S a 0 (e lasciando R ad 1), l'uscita assume il valore $\{Q = 1,\ Q' = 0\}$ e questo valore viene mantenuto anche quando S torna a 1. Alternativamente, un valore 0 applicato a R (lasciando S ad 1) porta l'uscita al valore $\{Q = 0,\ Q' = 1\}$.

Il circuito *ricorda* quale è stato l'ultimo dei due ingressi, S o R, a passare da 0 a 1.

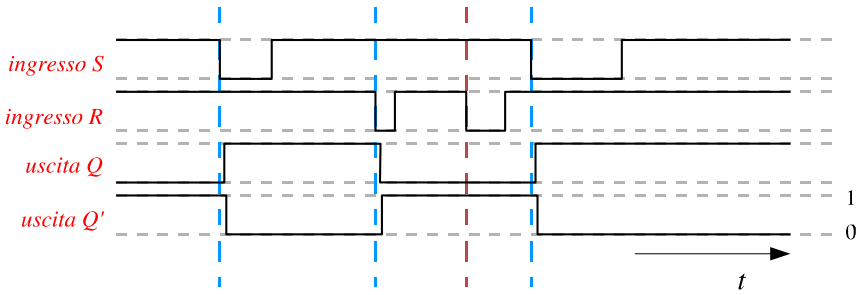


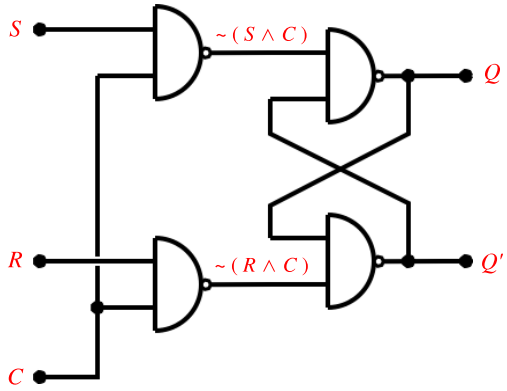
Figura 6.26: Diagrammi temporali per il flip-flop SR. Le uscite cambiano di stato in corrispondenza dei livelli 0 presenti sugli ingressi. In corrispondenza del terzo evento (linea verticale rossa) le due uscite sono già nello stato che dovrebbero raggiungere.

La figura 6.26 illustra il funzionamento del circuito tramite i diagrammi temporali. In corrispondenza delle transizioni da 1 a 0 degli ingressi S ed R, indicate dalle linee verticali tratteggiate, le uscite cambiano il loro stato di conseguenza, dopo un piccolo ritardo dovuto ai tempi di risposta finiti delle porte.

Si può ipotizzare che i due ingressi S ed R compiano entrambi la transizione $0 \rightarrow 1$ nello stesso istante. È molto poco probabile che la transizione avvenga *esattamente* nello stesso istante; una differenza di tempo tra i due segnali, sia pur minima, c'e' praticamente sempre. Tuttavia, se anche la transizione avvenisse realmente nello stesso istante, lo stato $S = 1,\quad R = 1,\quad Q = 1,\quad Q' = 1$ non è uno stato stabile; in un tempo brevissimo il circuito ricadrebbe in uno dei due stati stabili, selezionato

magari in maniera casuale da qualche minima interferenza sempre presenti nei circuiti, o anche solo dall'agitazione termica delle cariche elettriche.

6.7.2 *Clocked-SR*



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'_-

Figura 6.27: Il flip-flop di tipo *clocked set-reset*.

Una variante del flip-flop di tipo SR è illustrata in fig. 6.27: vengono introdotte due ulteriori porte di tipo *NAND*, tramite le quali i due segnali S ed R vengono applicati al flip-flop vero e proprio. Le due porte sono controllate da un terzo ingresso C (*Clock*). Quando l'ingresso C è a livello 1 le due porte di ingresso sono *aperte* e i segnali presenti su S ed R (invertiti, perchè le porte sono *NAND*) sono applicati all'ingresso del flip-flop e modificano lo stato delle uscite. Quando le porte di ingresso si *chiudono* ($C = 0$), S ed R non possono più giungere agli ingressi del flip-flop e lo stato presente su Q e Q' al momento della chiusura delle porte rimane *congelato* fino a che le porte non vengono nuovamente aperte.

6.7.3 *D-type*

Nel flip-flop di tipo D (*Data*) si ha un ingresso D, che coincide con l'ingresso S del flip-flop SR; l'ingresso R viene ottenuto dalla negazione di D (fig. 6.28).

I quattro possibili stati dei due ingressi S ed R di fig. 6.27 si riducono a due $\{0, 1\}$ e $\{1, 0\}$.

Quando il *clock* C è alto (1), le due porte di ingresso sono aperte ed il dato D entra nel flip-flop e viene trasferito sull'uscita; quando il *clock* va basso (0), le porte si chiudono e, come nel caso precedente, rimane memorizzato il valore presente sulle uscite al momento della chiusura delle porte (fig. 6.29).

6.8 I flip-flop di tipo *Master-Slave*

Due flip-flop, ad esempio di tipo D o di tipo SR, possono essere combinati insieme in una disposizione detta *master-slave* (fig. 6.30). L'uscita del primo flip-flop, il *master*, costituisce l'ingresso del secondo.

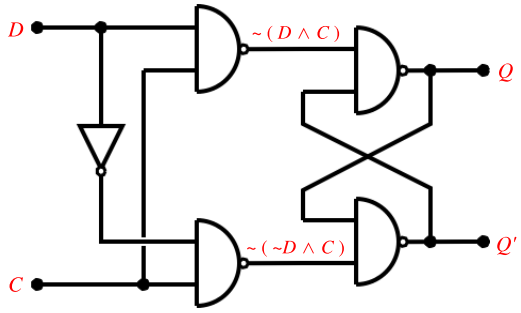


Figura 6.28: Il flip-flop di tipo D.

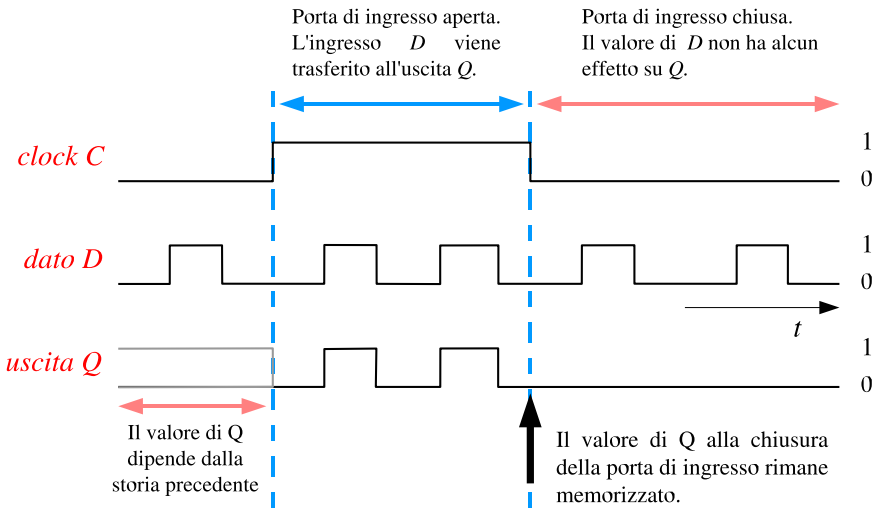


Figura 6.29: Il dato presente all'ingresso D viene trasferito all'uscita Q (e Q') fino a che il *clock* C e' alto; quando C diventa basso il valore delle uscite viene congelato ed ogni ulteriore variazione su D non ha piu' alcun effetto.

Un segnale di comando, il *clock*, è applicato sia al *master* che, invertito tramite una porta NOT, allo *slave*. Come nel flip-flop di tipo D del paragrafo precedente quando il *clock* è alto l'ingresso D e le sue variazioni entrano nel primo flip-flop della catena, il *master*, e sono presenti sulle uscite Q_M e Q'_M. L'ingresso al secondo flip-flop però è chiuso e quindi i segnali non possono giungere alle uscite Q e Q'. Solo quando il *clock* torna basso, si verifica l'apertura delle porte dello *slave*, con il trasferimento dei segnali dall'uscita del *master* all'uscita dello *slave*. Contestualmente, però, la porta del master si è chiusa, per cui le ulteriori variazioni presenti sull'ingresso D non possono più produrre alcun effetto. Al momento che il *clock* torna alto i valori di D possono tornare ad entrare nel *master*, ma non vengono più trasferiti nello *slave* a causa della chiusura della porta di ingresso di quest'ultimo.

In conclusione, mentre nel flip-flop del paragrafo precedente l'uscita seguiva l'ingresso per tutto il tempo che il segnale *clock* rimaneva alto, ora l'uscita cambia di stato solo ed esclusivamente in corrispondenza della transizione da alto a basso dell'ingresso *clock*: il dato presente all'uscita del master al momento della transizione vi

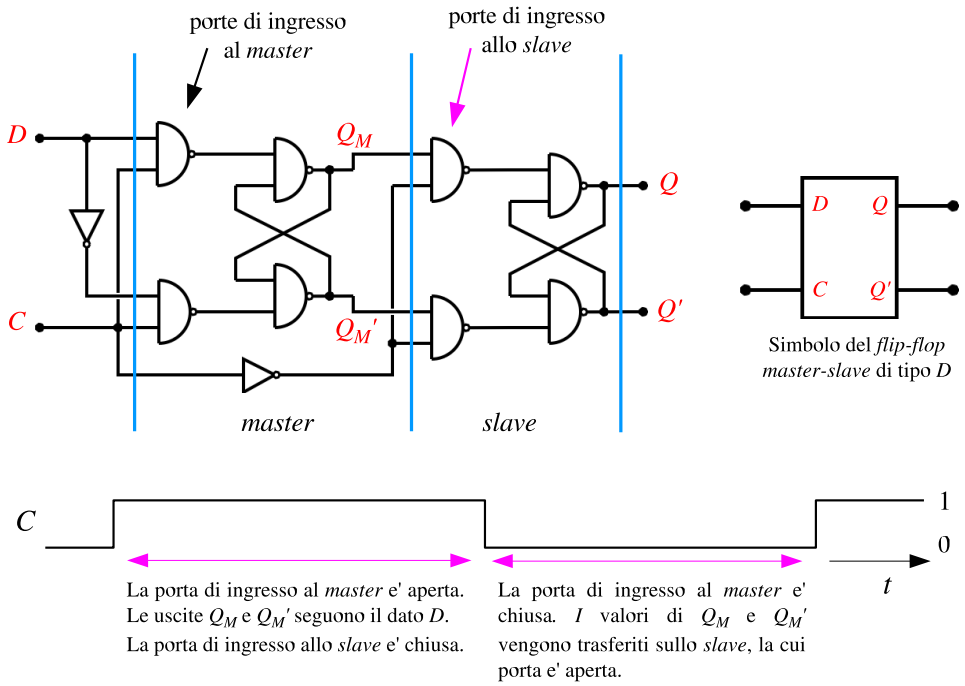


Figura 6.30: Flip-flop di tipo *master-slave* formato dalla combinazione di un flip-flop *D* e di un *clocked-SR*.

rimane congelato e trasferito sullo slave.

6.8.1 *Shift register* (Registro a scorrimento)

Una catena di flip-flop di tipo *master-slave* collegati come in fig. 6.31 costituisce un *registro a scorrimento* (*shift register*).

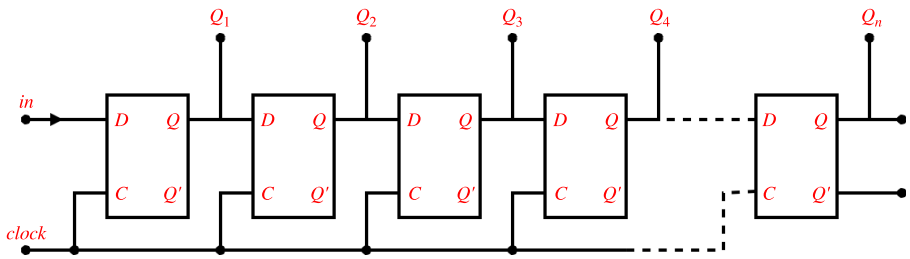


Figura 6.31: *Shift register*: il bit presente all'uscita di ogni flip-flop della catena viene inviato all'ingresso del flip-flop successivo; il segnale *clock* viene applicato contemporaneamente in parallelo a tutti i flip-flop.

Ogni rettangolo nello schema rappresenta un intero flip-flop del tipo descritto al paragrafo precedente, con i suoi ingressi D e C e le uscite Q e Q'. Dal secondo in poi, l'ingresso D di ogni flip-flop è collegato all'uscita Q del precedente. Il segnale *clock* è applicato in parallelo contemporaneamente a tutti gli elementi della catena.

L'ingresso del primo flip-flop è collegato al segnale *in* che il circuito riceve dall'esterno.

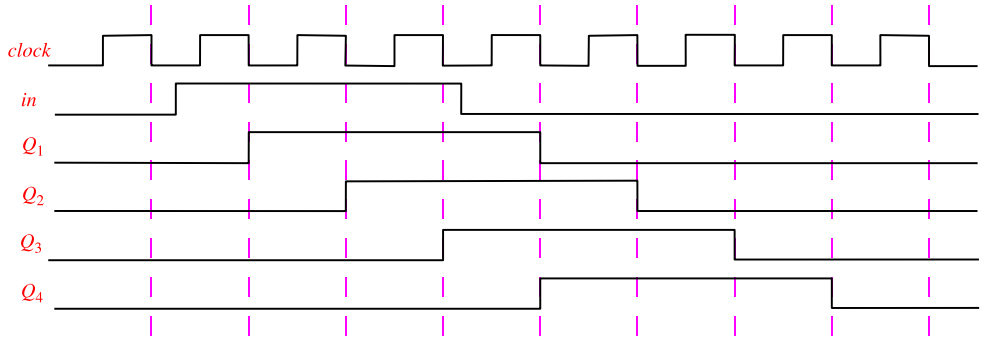
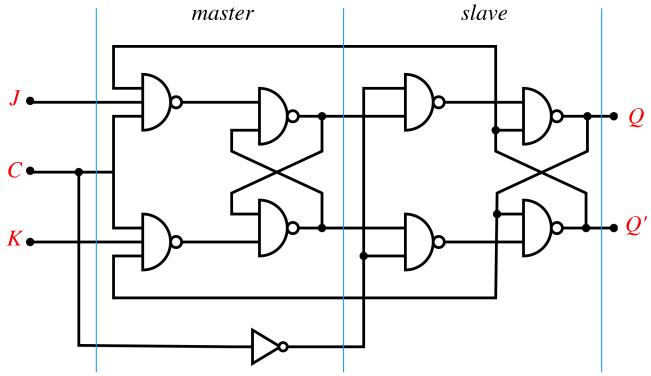


Figura 6.32: Ad ogni transizione da alto a basso del segnale *clock* (linee tratteggiate viola) il bit presente all'ingresso D di ogni flip-flop della catena viene memorizzato e trasferito alla corrispondente uscita Q: ad ogni ciclo del*clock* la sequenza di valori dei bit $\{Q_1, Q_2, \ldots Q_n\}$ scorre a destra di una posizione.

Ad ogni transizione del *clock* da alto a basso ogni flip-flop memorizza il dato che trova presente al proprio ingresso D e lo trasferisce alla propria uscita Q. Il primo flip-flop memorizza il dato presente all'ingresso *in* di tutta la catena. In questo modo, ad ogni ciclo del segnale *clock*, il valore della linea *in* viene campionato e fatto scorrere di una posizione nella catena dei flip-flop, come indicato dal diagramma temporale dei segnali in fig. 6.32.

6.8.2 Flip-flop *JK*

Uno dei flip-flop più versatili è quello indicato come JK (fig. 6.33).



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

Figura 6.33: Schema del flip-flop di tipo *JK*. La tabella a destra indica i valori assunti dalle uscite in corrispondenza della transizione $1 \rightarrow 0$ dell'ingresso C. Con l'indice $_{-1}$ si è indicato lo stato che le uscite avevano *prima* della transizione.

Si tratta di un flip-flop di tipo *master-slave*, formato da due flip-flop *clocked-SR*, con la differenza che l'ingresso al *master* è controllato da due porte *NAND* a tre

ingressi. Sul terzo ingresso di ciascuna porta è presente uno dei segnali di uscita, Q o Q', come indicato nella figura.

Come negli altri *master-slave* le uscite cambiano in corrispondenza della transizione $1 \rightarrow 0$ del *clock* C, in base al valore degli ingressi J e K, come descritto nella tabella a destra nella figura.

La condizione $J = K = 1$ è particolarmente interessante: quando entrambe le porte di ingresso sono aperte, ad ogni ciclo del segnale *clock* viene prelevato e trasferito l'inverso dello stato presente all'uscita. Quindi l'uscita Q (e Q') cambia di stato ad ogni ciclo del segnale *clock* e compie un ciclo completo ogni due cicli di quest'ultimo: il circuito è un *divisore di frequenza*. Se all'ingresso C viene applicata un'onda quadra di frequenza f, l'uscita Q fornisce un'onda quadra di frequenza $f/2$. Un flip-flop costruito appositamente per funzionare con questa modalità viene indicato come *Toggle* o *tipo T* (*toggle: interruttore a due posizioni*, fig. 6.34).

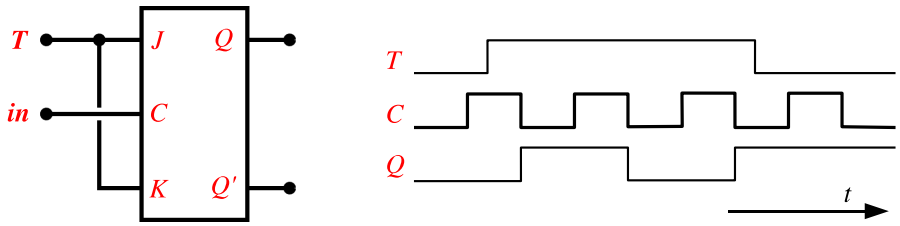


Figura 6.34: Il flip-flop JK con entrambi gli ingressi a 1 è un flip-flop di tipo T.

6.8.3 Contatore binario

In figura 6.35 è riportata una catena di flip-flop di tipo T.

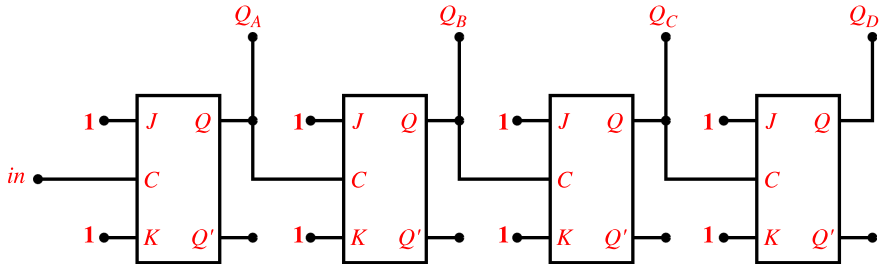


Figura 6.35: Un contatore binario può esere realizzato connettendo in cascata più flip-flop di tipo T (nello schema, flip-flop di tipo JK utilizzati in modalità T).

L'uscita Q_A cambia di stato ogni due cicli del segnale di ingresso *in*; Q_B ogni 4; Q_C ogni 8 ecc., come indicato nel diagramma temporale in fig. 6.36.

Dal grafico si vede facilmente come in ogni istante il valore dei bit $Q_D \ldots Q_A$ indichi, in codice binario, il numero di cicli effettuati dal segnale applicato all'ingresso *in* della catena. Il bit Q_D è il più significativo, il bit Q_A il meno significativo.

Dal grafico temporale si vede anche come la commutazione delle uscite Q avvenga con un ritardo sempre crescente via via che ci si sposta verso destra lungo la catena.

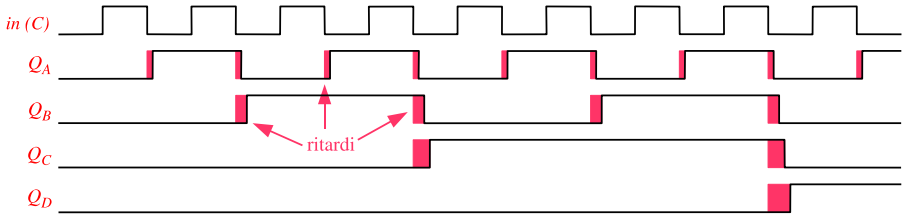


Figura 6.36: Diagramma temporale delle commutazioni in un contatore binario a 4 bit.

Il segnale nel percorso dall'ingresso *in* alle varie uscite deve attraversare un numero sempre maggiore di porte, ciascuna delle quali introduce il proprio ritardo.

Questo accumularsi dei ritardi pone dei limiti severi alla velocità massima di operazione di un contatore di questo tipo. Quando il ritardo accumulato dallo stadio *n-esimo* diventa eguale alla durata del ciclo del segnale di ingresso, non esiste più un intervallo di tempo durante il quale le uscite siano tutte quante valide e possano quindi essere lette correttamente.

6.8.4 Contatore sincrono

Il problema dell'aumento del ritardo visto al paragrafo precedente al crescere del numero degli stadi di un contatore binario può essere eliminato utilizzando una rete di conteggio *sincrona*.

Il principio base di ogni rete sincrona è di applicare il segnale che deve determinare la commutazione dei flip-flop contemporaneamente a tutti gli elementi della rete. Nel caso del contatore sincrono, il segnale *in* viene applicato a tutti gli ingressi C degli n flip-flop (fig. 6.37).

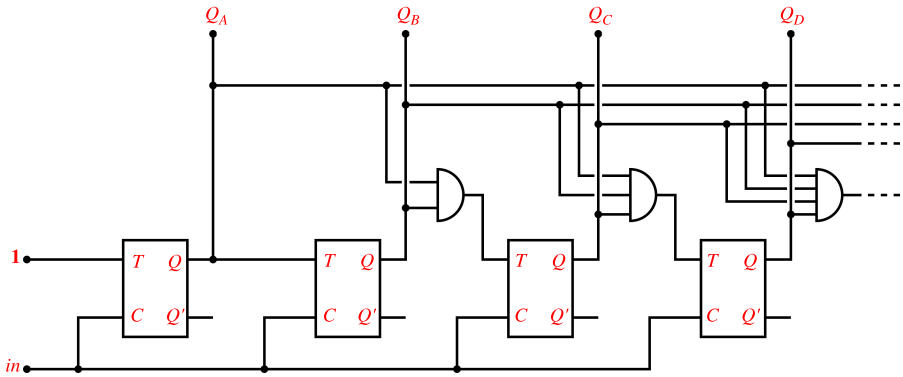


Figura 6.37: Schema di un contatore sincrono a 4bit.

Circuiti logici ausiliari predispongono ciascun flip-flop nella condizione di effettuare o no la commutazione in corrispondenza di ogni transizione del *clock*. Nel caso del contatore, la logica ausiliaria deve predisporre ciascun ingresso T a 1 (e quindi predisporre per la commutazione al successivo passaggio $1 \rightarrow 0$ di C) se e solo se

tutte le uscite Q dei flip-flop precedenti sono a 1, come richiesto dall'aritmetica del conteggio binario.

Nel circuito in figura 6.37 la frequenza massima di conteggio è limitata dal tempo necessario al segnale da ciascun ingresso C a giungere alla corrispondente uscita Q e propagarsi, in parallelo, attraverso ogni porta AND fino agli ingressi T.

La maggiore complessità circuitale, cioè l'aggiunta per ogni stadio del contatore di una porta AND con un numero di ingressi crescente fino al numero di stadi, è il prezzo da pagare per l'aumento di velocità che il contatore sincrono offre rispetto a quello *asincrono* di fig. 6.35.