Elettronica Applicata

Gaetano Insinna

Politecnico di Torino a.a. 2021/22

Le seguenti dispense non sono da considerare una completa sostituzione delle slides, videolezioni o appunti dei professori, né un riassunto perfetto della materia (specialmente nel capitolo (4) dove sono state omesse molte parti a mio avviso non presenti con regolarità negli esami), ma piuttosto una raccolta di concetti chiave che ho ritrovato diverse volte nelle prove d'esame e negli esercizi proposti dai professori, guidati dalle slides e ampliati tramite diverse fonti (tra cui siti inerenti all'elettronica gestiti da professori e appunti di altri atenei).

Se si dovessero incontrare degli errori vi prego di scrivere in privato al sottoscritto e mi occuperò di aggiornare il file.

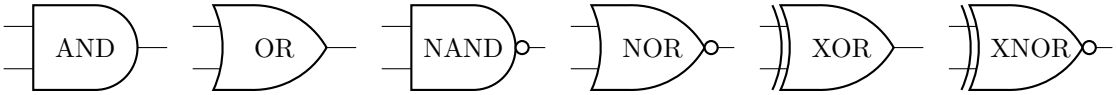
Questo lavoro, per essere come volevo, mi ha portato via tanto tempo, nonostante ciò credo che sia giusto condividerlo con voi. Chiunque voglia ringraziarmi con una donazione simbolica vi lascio il mio paypal: *gaetano.insinna@gmail.com*

Indice

1 Circuiti Logici e Digitali

1.1 Circuiti Logici

Un **circuito logico** è un sistema di porte logiche formato da N input e M output. Le porte logiche, a loro volta, sono la traduzione delle tavole della verità che forniscono un'uscita tramite degli stati logici (0 e 1). Tra le principali *porte logiche* troviamo:



A livello elettronico una porta logica è un **modulo digitale** che riconosce gli stati logici tramite a dei segnali elettrici. Ogni modulo digitale ha un'**alimentazione** che è descritta tramite una tensione positiva. A valle di ciò, anche gli stati logici sono rappresentati tramite valori di tensioni e per convenzione assumiamo che la tensione di tipo alto V_H rappresenta lo stato 1, mentre la tensione di tipo basso V_L rappresenta lo stato 0. Si crea così una *correlazione* tra gli stati logici e le grandezze elettriche e in una prima approssimazione possiamo affermare che:

$$\begin{cases} V_H \approx V_{AL} \\ V_L \approx 0V(GND) \end{cases}$$

Ciò porta a chiedersi quale sia il valore di tensione in cui si passa da stato logico basso ad alto. Esso è chiamato **valore di soglia** ed è rappresentato come V_T (valore di threshold). Tuttavia, essendo i valori in ingresso e in uscita affetti da disturbi, la soglia non può essere rappresentata solamente da un valore, in quanto non può essere assicurato che il valore non sia stato perturbato. Per ovviare a questo problema la soglia deve essere rappresentata da un certo intervallo di valori e gli stati alti e bassi, per essere considerati come tali, devono rispettare certe condizioni.

$$\begin{cases} V_{OL} < V_{IL} \\ V_{OH} > V_{IH} \end{cases} \tag{1}$$

Questo sistema prende il nome di **condizioni di compatibilità** e V_O rappresenta la tensione di uscita mentre V_I quella di ingresso. Il modulo della differenza $V_O - V_I$ prende il nome di **noise margin** e in base al livello elettrico preso in considerazione può essere NM_H oppure NM_L.

$$NM_X = |V_{OX} - V_{IX}| \tag{2}$$

Il rumore non porta solamente a dei disturbi che influenzano il valore di soglia, ad ogni elaborazione di un segnale esso viene perturbato fino a cambiare totalmente le informazioni, per ovviare a questo problema si sceglie di utilizzare dei segnali **digitali** sui quali si può effettuare un recupero dei dati in modo da non perdere delle informazioni tramite un **comparatore**. Ogni modulo digitale opera da comparatore trasformando quindi un segnale analogico in digitale tramite il valore di soglia.

1.1.1 Transistor

Per creare delle porte logiche si usano i **transistor** che possono essere semplificati trattandoli come un interruttore non ideale in serie ad una resistenza. Ci sono due tipi di transistor **pMOS** e **nMOS**. La principale differenza tra i due MOSFET è la seguente: un pMOS è ON con un ingresso logico basso mentre un nMOS è ON con un ingresso logico alto.

Input	pMOS	nMOS
0	ON	OFF
1	OFF	ON

Table 1: pMOS e nMOS

Una delle porte logiche basilari è l'**inverter** il quale è formato da **due transistor** uno pMOS chiamato di **pull-up** e uno nMOS chiamato di **pull-down** in quanto il primo opera come un interruttore verso V_{DD} mentre il secondo come un interruttore verso GND (la combinazione tra un nMOS e un pMOS prende il nome di **cMOS** (combined MOS)).

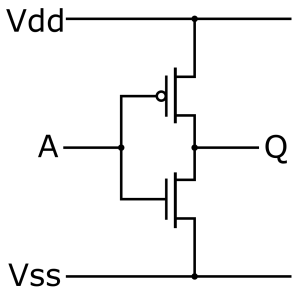


Figure 1: cMOS inverter

Il funzionamento dell'inverter cMOS è legato alle reti di pull-up e pull-down in quanto, qualora l'ingresso sia V_{DD} allora la rete di pull-up è spenta è quella di pull-down portando l'uscita verso GND e viceversa.

Input	Pull-up	Pull-down	Output
0	ON	OFF	V_{DD}
1	OFF	ON	GND

Table 2: cMOS inverter

Ampliando la trattazione del inverter e inserendo delle intere reti logiche al posto dei singoli transistor, si possono creare delle reti cMOS più complesse, come ad esempio delle porte logiche. Di seguito la realizzazione di una porta *NAND* tramite cMOS.

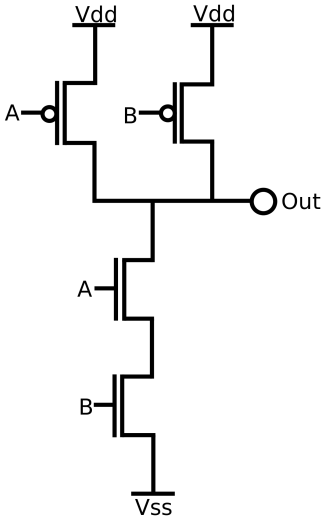


Figure 2: cMOS NAND gate

Ricordiamo la tavola della verità della porta NAND.

A	B	$\overline{A+B}$
0	0	1
0	1	1
1	0	1
1	1	0

Table 3: tabella della verità della porta NAND

Il funzionamento di questa porta è simile a quello dell'inverter con la differenza che le reti di pull-up e pull-down hanno diversi MOSFET. Quando l'ingresso ha uno stato logico basso significa che la rete di

pull-u**p** (formata dai **p**MOS) è attiva mentre quella di pull-dow**n** (formata dai **n**MOS) è spenta. Per far sì che la rete di pull-up sia accesa, essendo A e B in parallelo, basta che un MOS tra A e B abbia ingresso 0, mentre per far sia che la rete di pull-down sia accesa è necessario che entrambi i MOS abbiano ingresso alto in quanto sono in serie.

1.1.2 Parametri elettrici

I MOS di uscita hanno una resistenza equivalente R_O che può essere diversa tra le due reti (R_{OL} se diretta verso il GND mentre R_{OH} se diretta verso V_{AL}). In base a che stato si sta analizzando bisogna rispettare delle certe condizioni inerenti alla corrente di uscita I_O in quanto la tensione di uscita V_O dipende da essa.

$$\begin{cases} V_O > V_{OH} \Leftrightarrow |I_O| < |I_{OH}| \\ V_O < V_{OL} \Leftrightarrow |I_O| < |I_{OL}| \\ V_{OL} < V_{IL} \\ V_{OH} > V_{IH} \end{cases} \tag{3}$$

Queste sono le condizioni elettriche che devono essere rispettate affinché venga garantito il riconoscimento corretto degli stati logici.

Totem Pole: rappresenta l'uscita di un circuito logico binario in cui la tensione di uscita dello stato alto rappresenta V_{AL} mentre la tensione di uscita dello stato basso rappresenta il GND.

3-State: stessa implementazione del totem pole con l'aggiunta di un'uscita Z ovvero lo stato di **alta impedenza** ovvero lo stato logico dipende dalle altre parti del circuito. Questo modello prevede due switch uno inerente allo stato logico di uscita e uno che abilita o disabilita l'uscita. Può essere utilizzato per creare un modulo di controllo (generare le abilitazioni in in modo esclusivo - e.g. multiplexer).

Open Drain (Open Collector): realizzato con un solo NMOS verso GND. Se NMOS è chiuso allora uscita verso GND, mentre se NMOS aperto uscita di alta impedenza che dipende dal circuito esterno. Inoltre, l'open drain può essere utilizzato anche per realizzare delle funzioni logiche cablate collegando più uscite OD in parallelo (da ciò possono derivare anche dii bus seriali a basso costo).

1.1.3 Parametri dinamici

Nei casi reali le tempistiche non sono mai istantanee, numerosi parametri influiscono sulla temporizzazione di moduli digitali, interconnessioni eccetera. Nei segnali reali il **riconoscimento di nuovi stati logici** (fronte di salita o discesa ad esempio) non avviene in maniera istantanea, ma avviene con del ritardo rispetto al momento di inizio. Inoltre, all'interno di ogni modulo, tra l'ingresso e l'uscita dei dati avvengo delle **variazioni di tensione e corrente** che determinano del ritardo all'interno dello stesso. La combinazione di questi due fattori induce dei **ritardi nella propagazione** dei segnali logici e dei **limiti alla velocità operativa**. Si hanno due tipi di temporizzazione: **tempi di transizione** e **ritardi di propagazione**. Nel primo caso si parla di tempo di salita/discesa oppure transizione da livello alto a basso e viceversa, mentre nel secondo caso si parla di propagazione tra ingresso alto ed uscita bassa e viceversa.

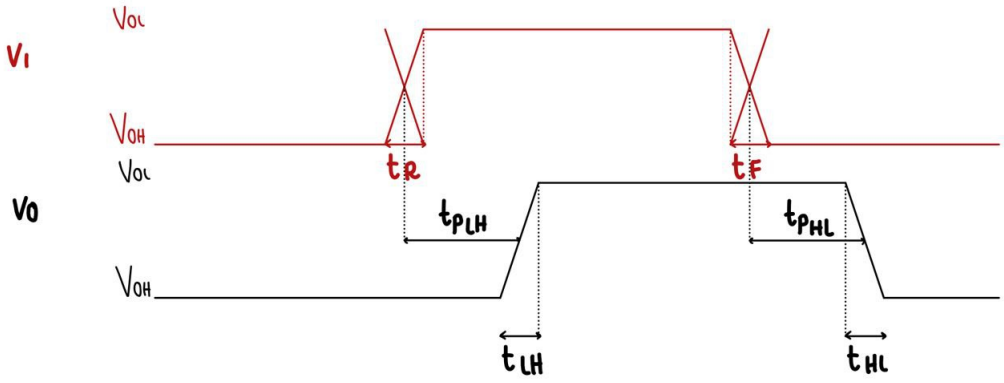


Figure 3: Tempi di transizione e di ritardo

1.1.4 Ritardi

Il **ritardo di propagazione** rappresenta l'intervallo di tempo che trascorre fra il momento in cui l'ingresso raggiunge il punto al 50% e l'uscita raggiunge anch'essa il 50%. Il punto al 50% è descritto nel modo seguente:

$$V_{50\%} = \frac{V_{OH} + V_{OL}}{2}$$

Il ritardo di propagazione da basso ad alto t_{pLH} e viceversa t_{pHL}, non sono necessariamente uguali e il ritardo complessivo si calcola tramite il valore medio:

$$t_p = \frac{t_{pLH} + t_{pHL}}{2}$$

Strutture simmetriche (CMOS e Totem Pole): $R_{OH} \approx R_{OL}$ quindi tempi di carico e scarico sono simili.

Strutture asimmetriche (Open Drain): transitorio di scarica (da stato alto a basso) veloce mentre quello di carica (da basso ad alto) lento in quanto cambiano le resistenze in gioco ($R_{LH} = R_{PU}$ mentre $R_{HL} = R_{OL}//R_{PU}$).

1.1.5 Effetti di carico

Oltre alla resistenza, un parametro che incide sulla costante di tempo τ è il carico. Aumentando il numero di carichi aumenta la capacità totale e questo porta a un aumento di ritardo e tempi di transizione che causano a loro colta delle **variazioni di stato multiple** per via del **rumore**.

Il massimo numero di ingressi collegabili a un'uscita è limitato dal **carico capacitivo**, questo numero prende il nome di **FANOUT**. Esso dipende generalmente dalla **compatibilità statica** (correnti di ingresso e uscita, carichi), ma anche dalla **compatibilità dinamica** (ritardi, tempi di salita e discesa). Nei circuiti CMOS (essendo la corrente di ingresso praticamente nulla) la compatibilità statica è trascurabile e quindi il FANOUT dipende esclusivamente dal carico capacitivo.

1.1.6 Segnali digitali differenziali

I segnali differenziali hanno la caratteristica di non essere *single-ended* ovvero un singolo circuito comune, il quale però è affetto ai disturbi. Un segnale differenziale è composto da una coppia di circuiti **complementari** che è meno affetto dai rumori esterni in quanto i disturbi si compensano grazie alla complementarietà.

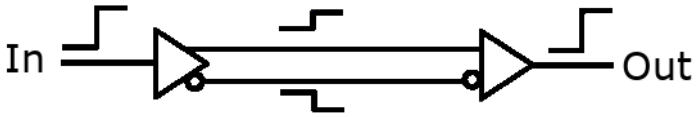


Figure 4: Segnali differenziali

1.1.7 Consumo di potenza

Il funzionamento di qualsiasi modulo richiede *energia*, che viene utilizzata per il funzionamento interno del modulo stesso, per i segnali esterni ed, infine, una parte viene trasformata in calore. L'energia necessaria viene fornita sotto forma di tensione, in questo caso **tensione di alimentazione** V_{AL}. Questa è solitamente costante (valori finiti di tensione) e per indicare il consumo del modulo si prende in considerazione la **corrente assorbita** dall'alimentazione.

Potenza statica: potenza assorbita in **assenza di commutazione**. Assorbimento pressoché costante, varia con la temperatura e la V_{AL}, modellabile come una **corrente continua** tra la tensione di alimentazione e la massa.

Potenza dinamica: potenza assorbita dall'alimentazione per eseguire una **commutazione** (da alto a basso e/o viceversa). Dipende dalla corrente che carica o scarica la capacità in uscita.

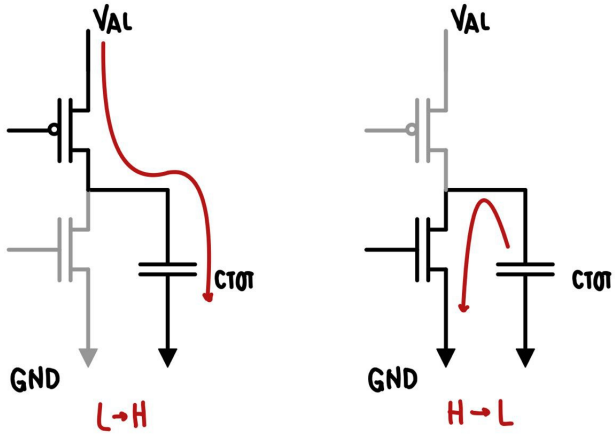


Figure 5: Corrente che circola nelle commutazioni

La potenza dinamica dipende da tensione e corrente, che a sua volta dipende dalla carica nel condensatore. Effettuando i calcoli troviamo che:

$$P_D = F \cdot C \cdot V^2$$

Da ciò possiamo dedurre che:

- la potenza statica P_S dipende dalla tecnologia del dispositivo (in quanto dipende da V_{TH})
- la potenza dinamica P_D dipende dalla tecnologia e soprattutto dal **carico capacitivo**

Per ridurre la potenza dinamica si può operare limitando la **frequenza di commutazione** (utilizzando algoritmi che richiedono un minor numero di commutazioni ad esempio), riducendo la **capacità** (migliorando la tecnologia) ed infine riducendo l'escursione di tensione $V_H - V_L$.

Un parametro che rappresenta la potenza in dipendenza alla tecnologia è il **prodotto potenza-ritardo**:

$$P_D \cdot T_P = k$$

che in un piano cartesiano altro non sono che delle iperboli che rappresentano la correlazione inversa che hanno potenza e ritardo (moduli con ritardi più bassi consumano più energia e viceversa).

Consumo nelle strutture CMOS: in condizioni statiche (senza carico) la potenza

$$P_S = I_{off} \cdot V_{AL}$$

mentre in condizioni dinamiche la potenza

$$P_D = C \cdot F \cdot V_{AL}^2$$

In prima approssimazione c'è solo un consumo dinamico, tuttavia nei circuiti recenti diventi significativo il consumo stativo.

1.2 Circuiti Bistabili (Flip-Flop)

I circuiti bistabili, detti anche **flip-flop**, sono dei circuiti sequenziali ovvero dei circuiti nei quali l'uscita dipende dalla **storia precedente** (quindi devono esserci degli **elementi di memoria**. I moduli che non prevedono degli elementi di memoria, quindi che dipendono solamente dall'ingresso, prendono il nome di **circuiti combinatori**. Ci sono due modalità con le quali un flip-flop può commutare lo stato, in maniera **sincrona**, ovvero con la presenza di un *clock* che regola la temporizzazione, oppure in maniera **asincrona**, regolati da singoli comandi.

L'elemento alla base dei flip-flop è l'**anello invertente**

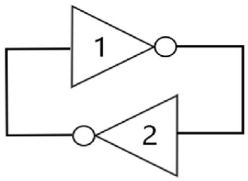


Figure 6: Anello di inverter

nel quale l'uscita dell'*inverter* 1 è uguale all'ingresso dell'*inverter* 2 e viceversa.

1.2.1 Flip-Flop Set Reset

È il flip-flop più semplice dal punto di vista funzionale. Esso ha due ingressi **S (set)** e **R (reset)**, chiamato a volte anche *clear*, due porte *NOR* (potrebbe essere realizzato con due porte NAND) e due uscite $\boldsymbol{Q}$ e $\overline{\boldsymbol{Q}}$, complementari tra loro. Il funzionamento è semplice, l'ingresso set pilota la modifica dello stato in memoria portando l'uscita a 1, l'ingresso reset, azzera la memoria portando l'uscita a 0. Per far sì che set funzioni reset deve essere basso, mentre se entrambi gli ingressi sono a 0 allora ci sarà condizione di memoria. Tuttavia, entrambi gli ingressi non possono trovarsi entrambi a 1 perchè porterebbero ad una situazione di instabilità, motivo per il quale si preferiscono altri flip-flop al SR.

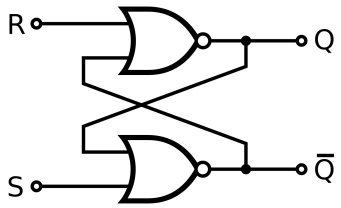


Figure 7: FF SR con porte NOR, attivo alto

S	R	$\boldsymbol{Q}$	
0	0	$\boldsymbol{Q_{n-1}}$	*memoria*
0	1	0	*reset*
1	0	1	*set*
1	1	-	—

Table 4: tabella della verità del FF SR

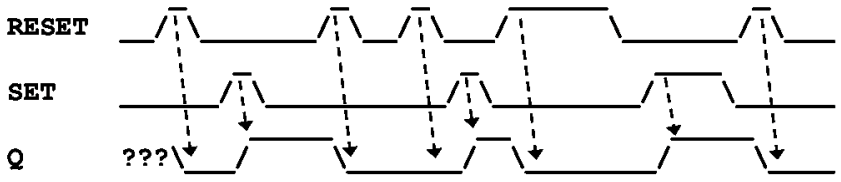


Figure 8: Diagramma temporale del FF SR

1.2.2 Latch D

È un dispositivo nel quale viene eliminata la condizione instabile del FF Set Reset. Per fare ciò S e R vengono cortocircuitati con un inverter tra gli ingressi così da non avere due ingressi uguali. Il comando avviene al di fuori del flip-flop e prende il nome **D**. Oltre al comando D (che quindi produrrà un set e reset opposti), il Latch D prevede un comando chiamato **latch enable (LE)**, che attiva o disattiva la memorizzazione dei dati in quanto, se LE è attivato allora c'è *trasparenza*, l'ingresso passa sull'uscita, mentre se è disattivato c'è **memoria**.

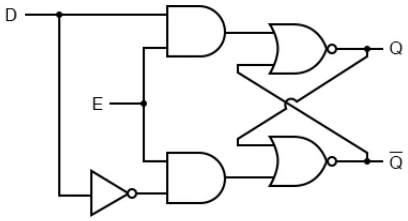


Figure 9: Latch D

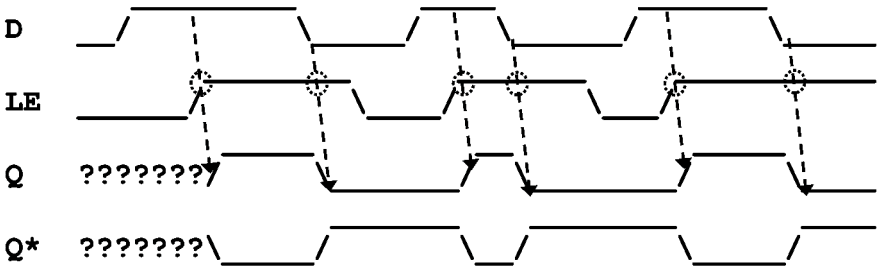


Figure 10: Diagramma temporale Latch D

Esiste una versione del Latch D con due segnali asincroni chiamati **PRESET** e **CLEAR** che agiscono in maniera indipendente dal *latch enable*.

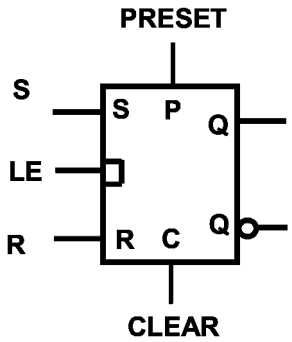


Figure 11: Latch D con PRESET e CLEAR

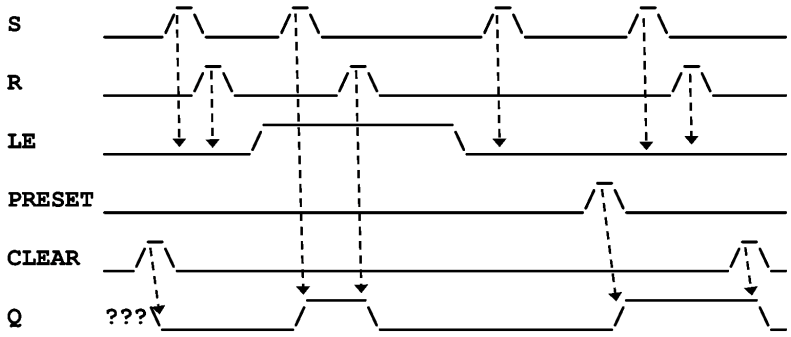


Figure 12: Diagramma temporale Latch D con PRESET e CLEAR

1.2.3 Flip-flop Master-Slave

I FF **Master-Slave** sono dei particolari tipi circuiti formati da una cascata di FF latch (ovvero le uscite del primo FF sono gli ingressi del secondo) aventi il clock in opposizione di fase. Il dato in ingresso viene memorizzato quando CK passa da **basso ad alto** infatti viene bloccato il primo flip-flop, chiamato **master** in quanto prende il controllo dei dati, che immagazzina il dato e il secondo flip-flop, **slave**, è nello stato trasparente. Essendo master e slave con il clock in opposizione di fase non potranno mai essere entrambi nello stesso stato.

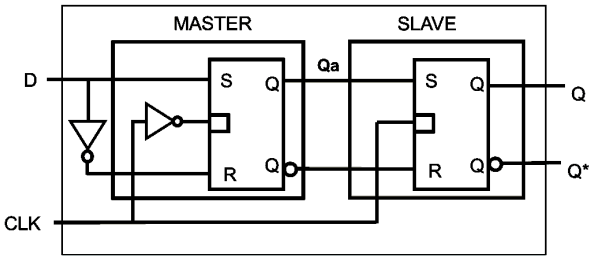


Figure 13: Circuito FF con latch D master-slave

Per quanto riguarda il master: è trasparente (**porta in uscita i dati ricevuti in ingresso**) quando CK è basso, mentre memorizza quando CK è alto. Di seguito $\boldsymbol{Q_a}$ si riferisce all'uscita del master. Per quanto riguarda lo slave, invece, ogni transizione $0 \rightarrow 1$ viene portato D in Q.

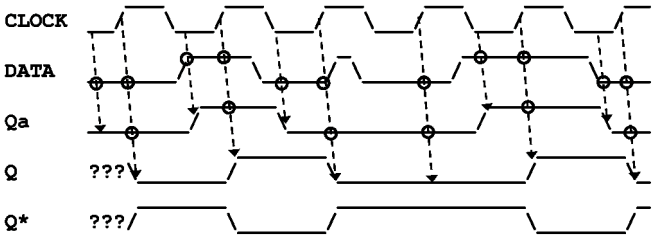


Figure 14: Diagramma temporale D master-slave

Esiste, inoltre, un secondo tipo di D Master-Slave che si differenzia da quello appena descritto per la condizione di memoria. Si chiama **D dual edge DDR** e come si intuisce dal nome la memorizzazione avviene la doppio della frequenza infatti, il flip flop non memorizza solamente per valori di clock alti, ma ad ogni fronte di clock ($0 \rightarrow 1, 1 \rightarrow 0$).

1.2.4 Flip-Flop JK

Il flip-flop JK altro non è che un FF SR master-slave con reazione incrociata, infatti le uscite sono le medesime con l'unica differenza di avere al posto dell'uscita proibita la *commutazione* ovvero, ogniqualvolta **J = K = 1 l'uscita commuta ad ogni colpo di clock**.

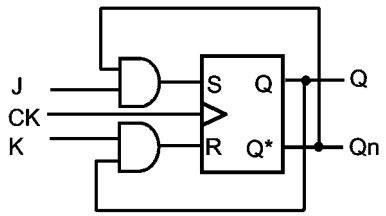


Figure 15: Circuito Flip-Flop JK

J	K	Q	
0	0	Q_{n-1}	*memoria*
0	1	0	*reset*
1	0	1	*set*
1	1	$\overline{Q_{n-1}}$	*complemento*

Table 5: tabella della verità del FF JK

1.2.5 Temporizzazione dei Flip-Flop

Le porte logiche introducono dei ritardi: l'uscita commuta **sempre dopo** il comando all'ingresso di un tempo t_p (*tempo di propagazione*). Oltre a ciò bisogna rispettare dei parametri affinché il dato in ingresso riesca ad arrivare in uscita senza errori di sorta. I parametri principali sono:

- t_p tempo di **propagazione**: quanto tempo occorre al dato per giungere dall'ingresso all'uscita
- t_{SU} tempo di **setup**: tempo **prima** del fronte di salita del clock in cui il dato deve rimanere stabile
- t_H tempo di **hold**: tempo **dopo** il fronte di salita del clock in cui il dato deve rimanere stabile
- t_R tempo di **salita** del clock
- t_F tempo di **discesa** del clock

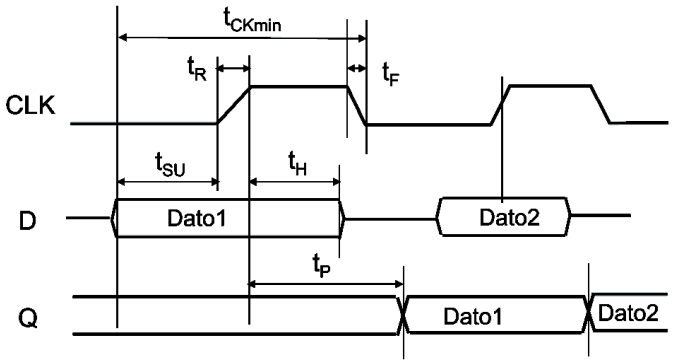


Figure 16: Temporizzazioni da garantire

Il clock deve restare per un tempo minimo negli stati H e L secondo la formula:

$$T_{CKmin} = t_{SU} + t_R + t_H + t_F$$

ricordando anche la relazione:

$$F_{CKmax} = \frac{1}{T_{CKmin}}$$

1.3 Circuiti Sequenziali

1.3.1 Segnali seriali e paralleli

I segnali numerici possono essere trattati in forma **seriale** o **parallela**. Il primo modo prevede dei bit in tempi successivi su un unico bus, mentre il secondo modo prevede bit nello stesso istante di tempo, ma in diversi bus. Entrambe le tipologie di segnali sono cadenzate da un segnale di clock, che nel primo caso trasferisce un bit per ogni ciclo di clock, mentre nel secondo caso N bit contemporaneamente. Una connessione seriale è più lenta, ma ha un minor consumo e costo (più usata su lunghe distanze), mentre la connessione parallela è più veloce, ma ha un maggior consumo e costo.

1.3.2 Registri

Un registro è un **insieme di flip-flop** con comandi comuni. Possono avere ingresso ed uscita parallela o seriale (PIPO, PISO, SIPO, SISO: parallel/serial input, parallel/serial output).

Shift-Register: insieme di FF(D) in cascata con un clock in comune che hanno un meccanismo a *scorrimento* ovvero il dato viene scalato nella catena.

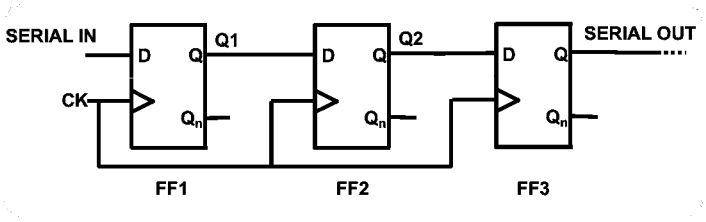


Figure 17: Circuito di uno Shift-Register SISO

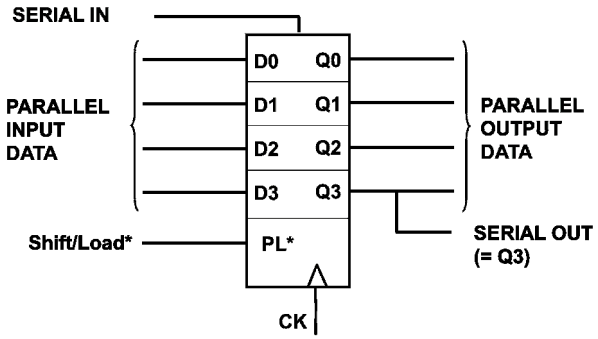


Figure 18: Shift-Register completo

Ogni FF ha in ingresso un *multiplexer* che seleziona uno fra due dati in ingresso (parallelo o seriale).

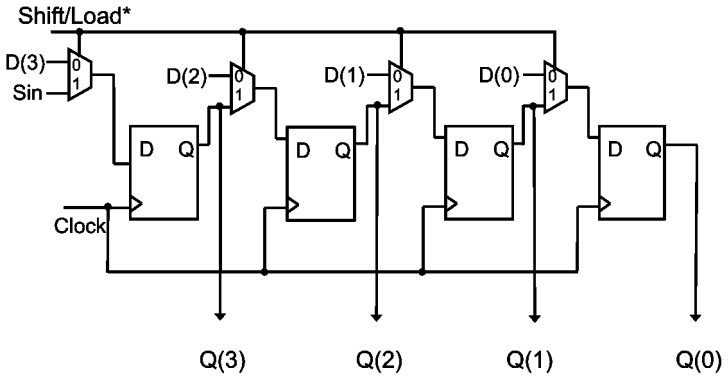


Figure 19: Circuito di uno Shift-Register completo

Il comando **shift/load** carica l'ingresso in parallelo ($D(3) \rightarrow D(0)$) se basso, mentre esegue lo scorrimento se alto.

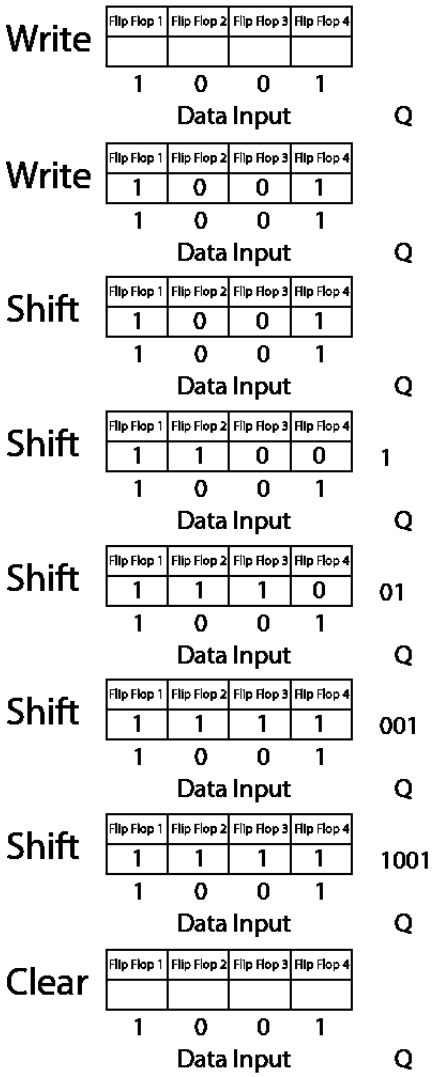


Figure 20: Funzionamento di uno Shift-Register

1.3.3 Contatori e Divisori

È un componente elettronico costituito da un circuito integrato in cui sono implementate le funzioni di contatore digitale.

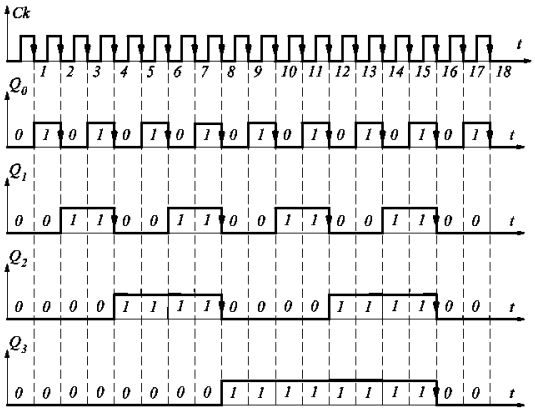


Figure 21: Diagramma temporale di un contatore

Contatore: circuito logico che genera sulle uscite una sequenza di conteggio binario incrementata a ogni ciclo di clock.

Divisore: contatore di cui viene utilizzata solo una uscita Q_n. Il suo campo di applicazione è la **divisione di frequenza**. In base a quanti FF ha il contatore avverrà una divisione di frequenza di ordine 2^n (esempio clock con frequenza 50 MHz contatore con 1 FF, quindi ad un bit, uscita avrà frequenza 25 MHz).

Contatore asincrono: un contatore si dice **asincrono** quando il clock è collagato solamente al primo flip-flop della serie. I successivi clock sono collegati a catena (**ripple**) costituendo così un *ritardo di commutazione* cumulativo.

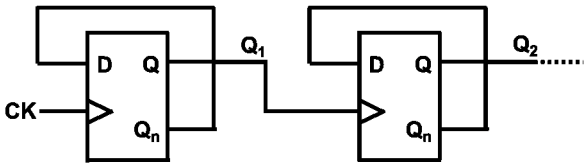


Figure 22: Contatore asincrono

In questa figura notiamo che, se in un singolo FF D, il tempo di propagazione vale T_p, per N FF ci sarà un ritardo totale di $N \cdot T_p$.

Contatore JK-FF: tramite dei FF di tipo JK si possono realizzare dei contatori asincroni in quanto, il flip-flop cambia stato ogniqualvolta che J = K = 1.

Contatore sincrono:un contatore è sincrono quando tutti i FF ricevono lo **stesso clock** e di conseguenza, tutte le uscite commutano con lo stesso ritardo (ritardo non varia da stadio a stadio come nei asincroni).

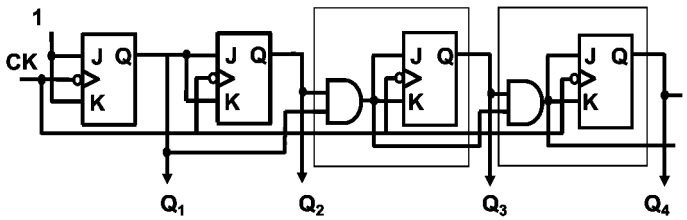


Figure 23: Circuito di un contatore sincrono, dal 3 stadio in poi sono tutti i medesimi

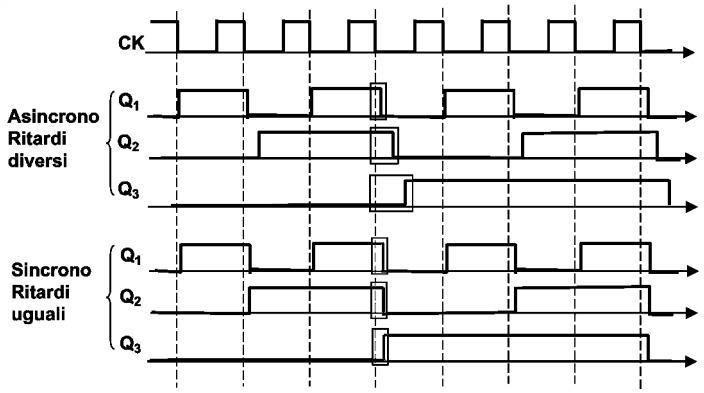


Figure 24: Differenza ritardi contatori sincroni e asincroni

1.3.4 Temporizzazioni con logica combinatoria

Come descritto nel paragrafo 1.2.5 la **frequenza massima di funzionamento** è legata al soddisfacimento del tempo di *setup* dei flip-flop, tuttavia, in quel paragrafo, non si è presa in considerazione i ritardi legati alla logica combinatoria dei circuiti. Introduciamo quindi un nuovo ritardo da garantire:

- t_{CK-q} ritardo **clock-uscita**: l'uscita si stabilizza dopo questo ritardo dopo il fronte del clock
- t_{LCMIN},t_{LCMAX} minimo e massimo **ritardo di propagazione** della logica combinatoria

Unendo le informazioni in nostro possesso sappiamo che le condizioni da rispettare per calcolare il minimo periodo di clock sono le seguenti:

$$\begin{cases} T_{CK} > t_{CK-q} + t_{LCMAX} + t_{SU} \\ t_{CK_q} + t_{LCMIN} > t_H \end{cases} \tag{4}$$

Riassumendo la **frequenza massima di funzionamento** è legata al ritardo della transizione tra **CLK e Q** del flip-flop, ritardo della **logica combinatoria** e dal **tempo di setup** richiesto dal flip-flop. Strutture **ripple** (catene di porte), inducono ritardi, nonostante siano le più facili da implementare. Una

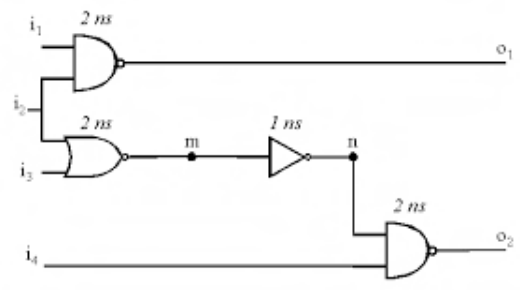


Figure 25: Ritardo logica combinatoria. In questo caso il $t_{LCMIN} = 2ns$ mentre il $t_{LCMAX} = 5ns$

soluzione per ovviare a questo problema, nonostante siano più complicate, è quella delle strutture **look ahead**, ovvero delle strutture dirette con un solo livello di porte.

1.4 Comparatori e Oscillatori

1.4.1 Comparatori

Il comparatore è un circuito in grado di tradurre un ingresso **analogico** in un'uscita **binaria** (L/H, 0/1). Se il valore in ingresso è maggiore di una soglia allora in uscita uno stato alto (comparatore non invertente, altrimenti sarebbe un comparatore invertente).

Essendo il segnale, tuttavia, affetto da rumore la soglia potrebbe non essere affidabile. Per ovviare a questo problema si utilizza una comparatore con **isteresi**, ovvero un comparatore con **due soglie**. L'isteresi è dovuta dalla presenza della **retroazione** in quanto, il principio che sta alla base è quello di ritardare l'effetto dello stato precedente (retroazione è cortocircuito tra uscita e ingresso).

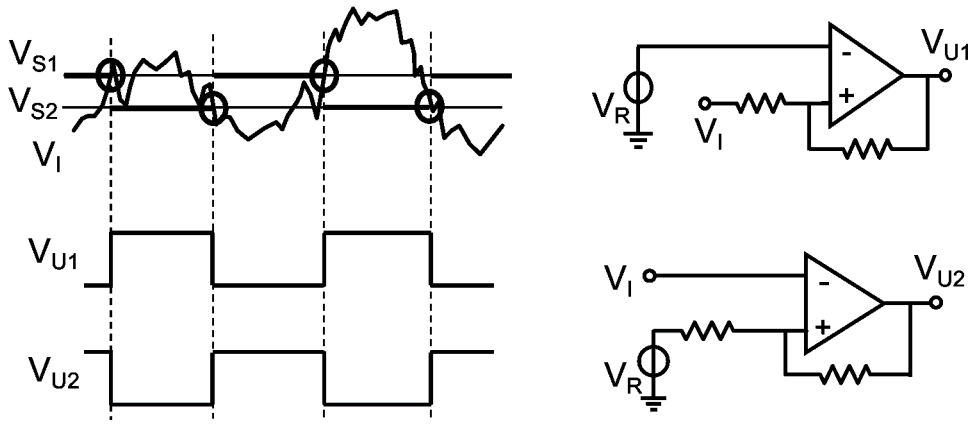


Figure 26: Comparatore con isteresi *invertente* e *non invertente*

Il comparatore con isteresi è anche chiamato **trigger di Schmitt**. Esso può essere utilizzato come **generatore di onda quadra** se messo in collegamento con una rete RC.

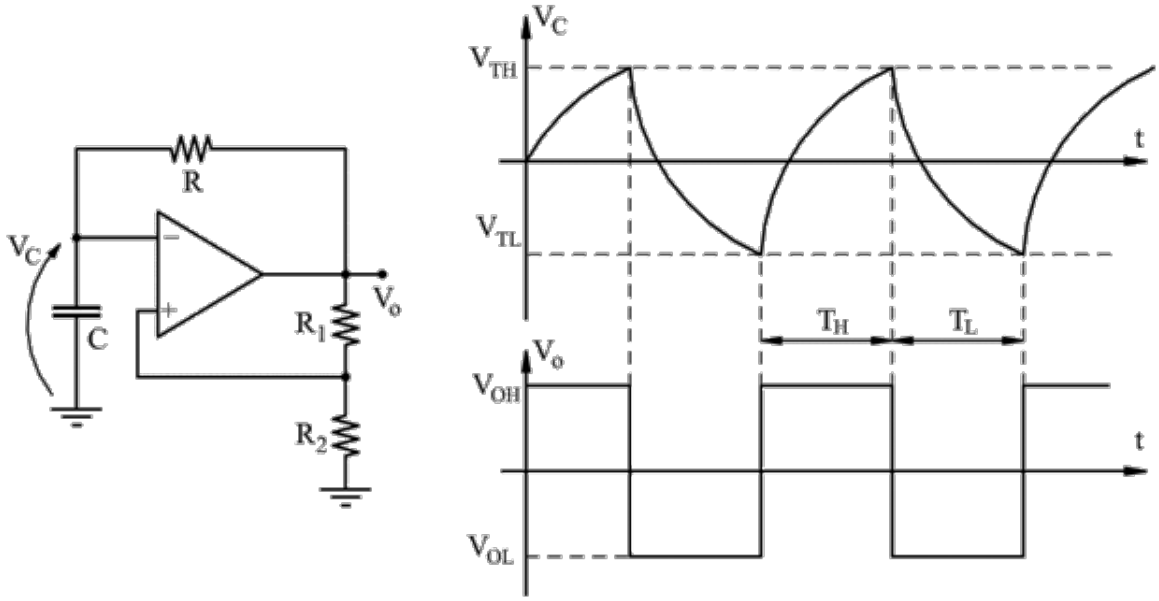


Figure 27: Generatore di onda quadra con trigger di Schmitt

1.4.2 Oscillatori

Sono dei circuiti che generano onde di varia frequenza, ampiezza e forma senza alcun segnale in ingresso.

Oscillatore ad anello: è formato da un numero **dispari di inverter** ed è privo di uno stato stabile. Inoltre il suo periodo non è preciso in quanto dipende dai ritardi di propagazione degli inverter secondo la formula $T = N(t_{pdLH} + t_{pdHL})$.

Sistemi con reazione: l'uscita U può oscillare anche senza alcun ingresso specifico se $A(f) \cdot \beta(f) = -1$. A prescindere dall'ingresso, le oscillazioni avvengono per $A(f) \cdot \beta(f) < -1$.

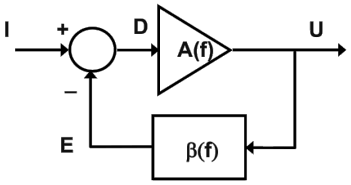


Figure 28: Oscillatore con retroazione

Solitamente $A(f)$ è costante in quanto un *amplificatore*, mentre il guadagno della rete di retroazione β dipende dalla frequenza che può essere implementata con: reti RC, circuito risonante LC oppure, con il **quarzo** (cristallo piezoelettrico). Esso assume una grande importanza in quanto:

- si deforma in presenza di un campo elettrico
- genera tensione se sottoposto a sollecitazione meccaniche
- elemento risonante con **basse perdite** e **alta precisione**

Un esempio di utilizzo del quarzo in elettronica è l'*oscillatore di Pierce* nel quale l'amplificatore può essere sostituito da una porta NOT, la resistenza in parallelo R_F forza l'inverter a lavorare intorno a $V_{in} = V_{out}$ e si ha una transcaratteristica con alto guadagno.

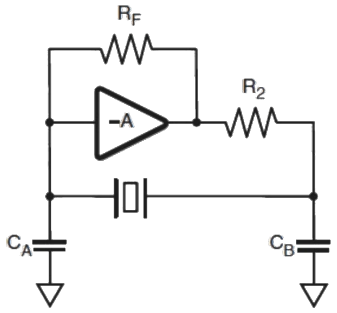


Figure 29: Oscillatore di Pierce

Phase Locked Loop: la massima frequenza di un oscillatore al quarzo è di ~ 200 MHz, per frequenze più alte nell'ordine dei GHz utilizzate quotidianamente nei processori attuali, serve il **PLL**. Esso è un sistema di controllo automatico che ha come scopo quello di **generare un segnale periodo** la cui fase è in relazione fissa con il segnale di ingresso di riferimento. Il suo utilizzo primario è quello sintetizzare una frequenza agganciando un **VCO** (*voltage controlled oscillator*), caratterizzato da un alta frequenza e una bassa precisione, con un oscillatore al quarzo, il quale ha le caratteristiche oppeste al primo, così da sintetizzare una oscillatore ad **alta frequenza e precisione**.

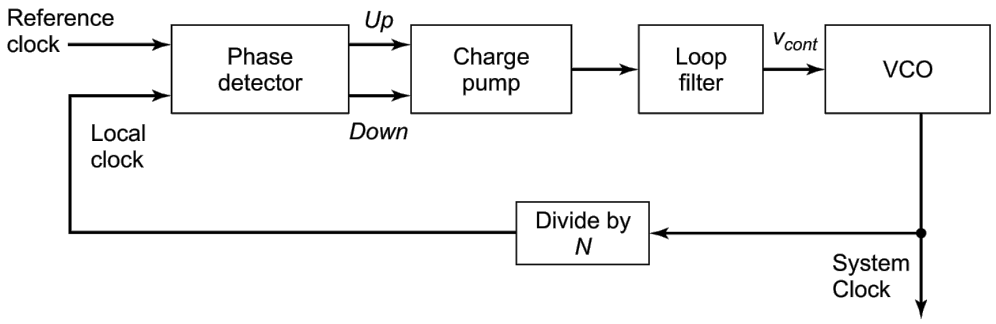


Figure 30: Phase locked loop

Il *PLL* è composto da:

1. **Phase Detector**: rilevatore di fase il quale identifica quando l'oscillatore è troppo *veloce* o *lento* rispetto alla frequenza di riferimento
2. **Charge Pump**: generatore di tensione controllato da due interruttori che vengono azionate se l'oscillatore è troppo lento (**UP**) o troppo veloce (**down**)

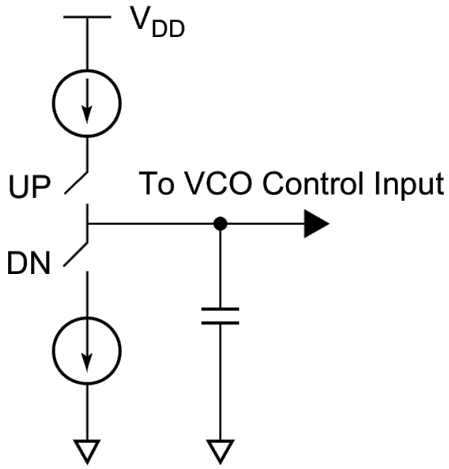


Figure 31: Circuito del *Charge Pump*

3. **Voltage Controlled Oscillator**: oscillatore controllato in tensione realizzato mediante un oscillatore ad anello che può raggiungere frequenze nell'ordine dei GHz, stabilizzato grazie all'anello di reazione che lo aggancia all'oscillatore al quarzo, di **riferimento**. Il singolo stadio ha un ritardo dipendente dalla tensione di controllo

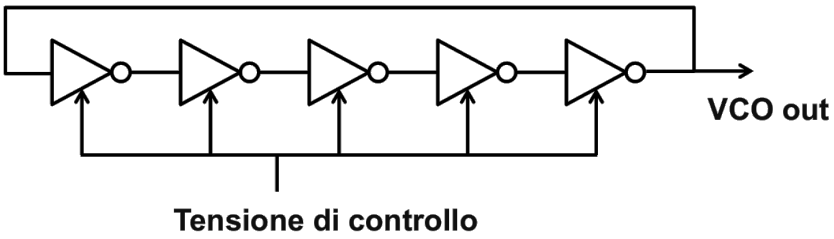


Figure 32: VCO con reazione

1.5 Logiche Programmabili

Un **dispositivo logico programmabile** è un circuito integrato programmabile utilizzato nei circuiti digitali. Al momento della fabbricazione non ha un funzione logica definita, ma tocca all'utente programmare il dispositivo prima di poterlo utilizzare.

1.5.1 Programmable Logic Array

Ogni funzione logica può essere espressa tramite somma logica, da ciò nasce il **PLA**, il quale prevede una matrice di porte **AND** programmabili collegata con una serie di porte **OR** programmabili.

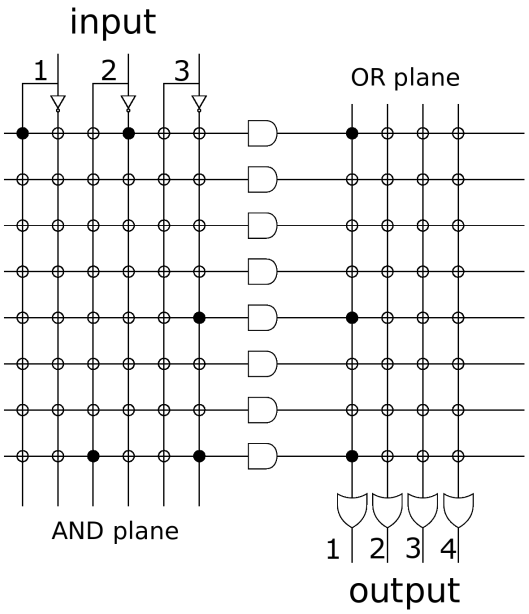


Figure 33: *Programmable Logic Array.* $O_1 = I_1 \cdot \overline{I_2} + \overline{I_3} + I_2 \cdot \overline{I_3}$

1.5.2 Field Programmable Gate Array

É un dispositivo hardware elettronico formato da un circuito integrato le cui funzionalità logiche di elaborazione sono appositamente programmabili e modificabili tramite opportuni linguaggi di descrizione hardware. Rispetto all'implementazione di un sistema digitale tramite CPU, l'implementazione tramite FPGA ha **maggiori costi e tempi di sviluppo** e **prestazioni superiori**.

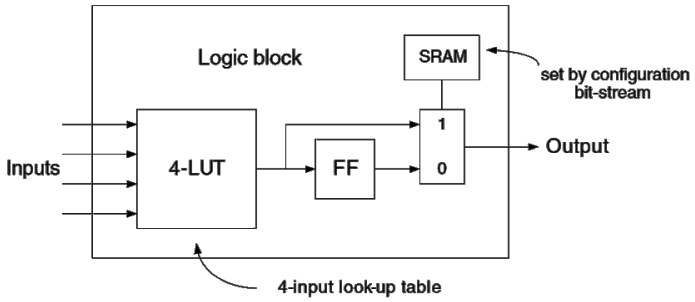


Figure 34: Blocco logico di una *FPGA*

Look-Up Table: struttura dati nella quale viene salvato il risultato di una tavola della verità. Permette di accedere alle soluzioni dei circuiti logici senza dover effettuare il calcolo in runtime.

1.6 Memorie a Semiconduttore

Le **memorie** a semiconduttore sono le componenti, basate su semiconduttori (*transistor, chip, ...)*, del calcolatore in grado di mantenere per un certo lasso di tempo delle informazioni (*dati* o *istruzioni*). La principale differenza che caratterizza le memorie è la **volatilità** che indica la capacità di mantenere il dato anche in assenza di tensione di alimentazione.

In base al livello di astrazione l'unità base delle memorie è data da:

- **bit** → livello di *circuito*
- **byte** → livello di *chip*
- **word** (pagine) → livello di *sistema*

Inoltre le celle di memoria sono indirizzabili in diversi modi: *singolarmente* oppure in *gruppi di bits/bytes.*

1.6.1 Architettura della matrice di memoria

Generalmente le memorie sono costruite in maniera **matriciale** al fine di essere più scalabile e regolare possibile. É formata da 2^n word di 2^m bit ciascuna

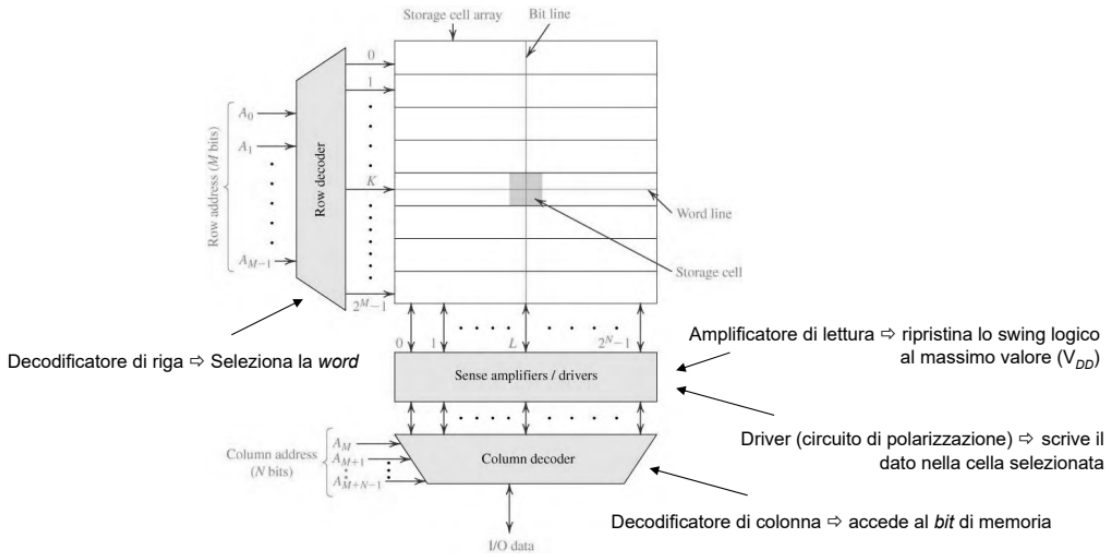


Figure 35: Rappresentazione di un blocco di una memoria

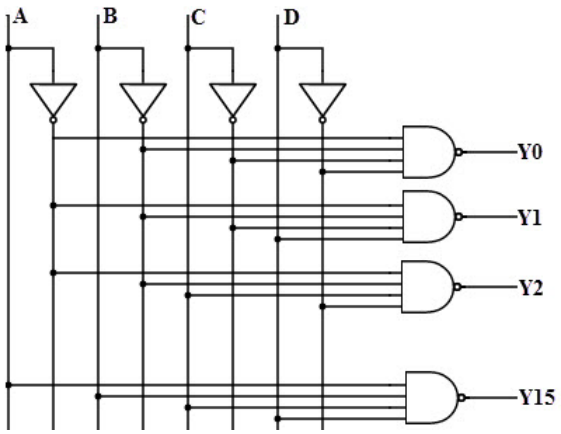


Figure 36: 4-bit decoder

Le operazioni che possono essere svolte su una memoria sono le seguenti:

- **Lettura**: il decoder di riga attiva una e una sola *wordline*. Successivamente tramite le *bitline* il decoder di colonna sceglie una word da leggere
- **Scrittura**: la parola da scrivere viene trasferita tramite le *bitline* e infine solo la parola selezionata dalla *wordline* viene scritta

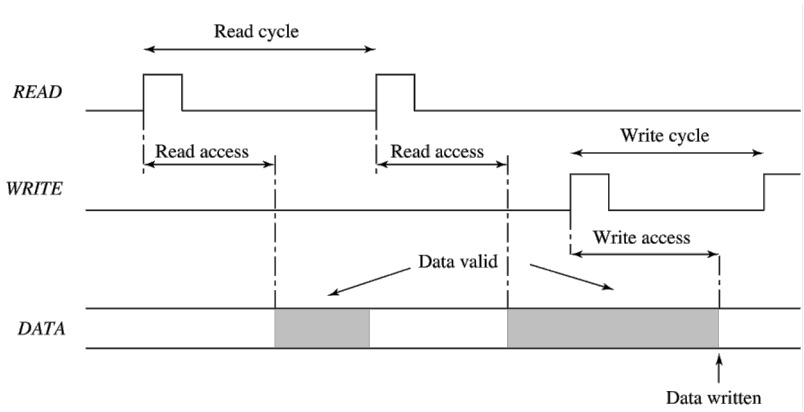


Figure 37: Temporizzazioni ciclo scrittura e lettura

1.6.2 DRAM

La **Dynamic RAM** è una memoria **volatile** che immagazzina i bit di memoria nel condensatore di *storage*. Si dice RAM dinamica in quanto il condensatore può perdere la carica e quindi ha bisogno di cicli di **refresh** per non perdere il dato. Essa è composta da un **transistor** e da un **condensatore**. Il transistor ha la funzione di caricare e scaricare (tramite comando della *wordline* WL) il condensatore, mentre quest'ultimo immagazzina il bit sotto forma di carica. La *bitline* BL ha una grande capacità parassita a causa delle numerosissime celle connesse ad essa ($\Rightarrow C_S << C_{BL}$ e la variazione di tensione sulla BL durante una lettura è $\sim$ 250mV).

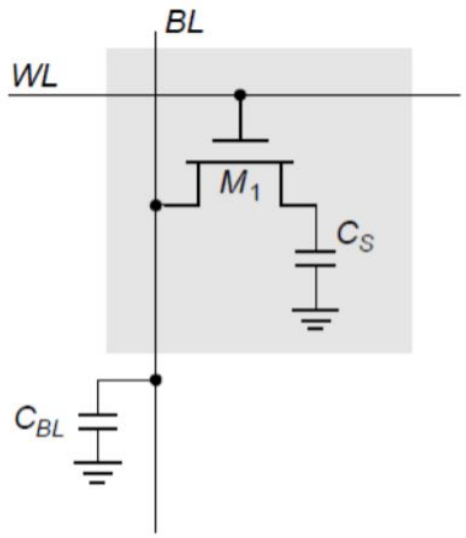


Figure 38: Cella DRAM a un transistor

Le operazioni che possono essere svolte su una memoria DRAM sono le seguenti:

- **Scrittura**: la tensione sulla BL è alta (1) o bassa (0), successivamente alla riga selezionata con la WL viene trasferita il valore di tensione sul condensatore della cella C_S
- **Lettura**: l'operazione di lettura avviene in più passaggi
  1. la BL portata a $\frac{V_{DD}}{2}$
  2. la WL portata a V_{DD} affinché il transistor possa entrare in conduzione
  3. i due condensatori C_S e C_{BL} sono in parallelo
  4. si ha quindi una **variazione di tensione** ai capi del condensatore C_{BL}. Infatti se C_S sta memorizzando un 1 allora la tensione ai capi del condensatore di *storage* v_{CS} sarà simile a V_{DD} che quindi farà **caricare il condensatore di BL** C_{BL} aumentando la tensione ai suoi capi; viceversa nel caso che il C_S stia memorizzando uno 0. Quindi la variazione ai capi di C_{BL} potrà essere positiva (nel caso si stia memorizzando un 1) o negativa, la variazione di tensione è molto piccola

$$\Delta V_{(1/0)} \simeq \pm \frac{C_S}{C_S + C_{BL}} \left(v_{CS} - \frac{V_{DD}}{2} \right)$$

  5. il cambiamento di tensione lungo la BL viene rilevato dal **sense amplifier** della colonna in cui si trova la cella di memoria in questione, il quale si occuperà di amplificare il segnale rilevato per farlo tornare al valore iniziale così da riportare il condensatore di storage al suo valore iniziale
- **Refresh**: a causa delle correnti di perdita bisogna periodicamente rinfrescare il contenuto di una riga con delle finte letture delle righe, tuttavia è un processo lento

Queste operazioni possono essere svolte sia con DRAM **sincrona** che **asincrona**, tuttavia cambia il funzionamento.

DRAM Asincrona: regolata da segnali di controllo grazie ai quali si possono effettuare le operazioni di lettura e scrittura. I principali sono: **RAS** (row address strobe) il quale serve ad attivare (selezionare) una riga, **CAS** (column address strobe), **WE** (write enable) un comando per attivare il pin cpme ingresso dati e **OE** (output enable) il quale funziona come "complementare" di WE infatti attiva il pin come uscita dati (infatti I/O dei dati è **bidirezionale**).

DRAM Sincrona: chiamata anche **SDRAM**, è una DRAM nella quale le operazioni di lettura/scrittura avvengono sui fronti di salita del segnale di clock. Qualora le operazioni avvengano **sia sul fronte di salita che su quello di discesa** allora di parla di **DDR** (double data rate) SDRAM che sarà, quindi, il doppio più veloce rispetto alle DRAM a single rate.

1.6.3 SRAM

La **Static RAM** è una memoria **volatile** che non ha bisogno del refresh continuo. É composta da **6 transistor**: **due inverter**, ognuno composto da 2 transistor, più **due pass-transistor di selezione** (tramite la WL) che permettono di effettuare lettura e scrittura (tramite le BL, una normale e una complementata).

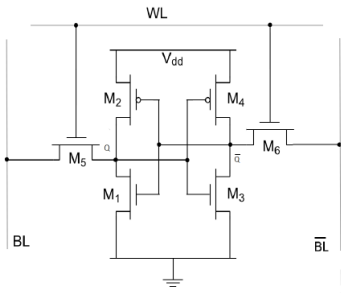


Figure 39: Cella SRAM a 6 transistor

Le operazioni che si possono effettuare sono:

- **Scrittura**: la BL viene forzata al valore desiderato, di conseguenza la $\overline{\text{BL}}$ al valore complementare e successivamente avviene la selezione della riga tramite WL = 1
- **Lettura**:
  1. BL e $\overline{\text{BL}}$ caricate a V_{DD}
  2. WL caricata a V_{DD}
  3. BL e $\overline{\text{BL}}$ variano di potenziale in base al valore memorizzato nella cella
  4. variazione di potenziale amplificata dal *sense amplifier* che viene quindi trasmessa

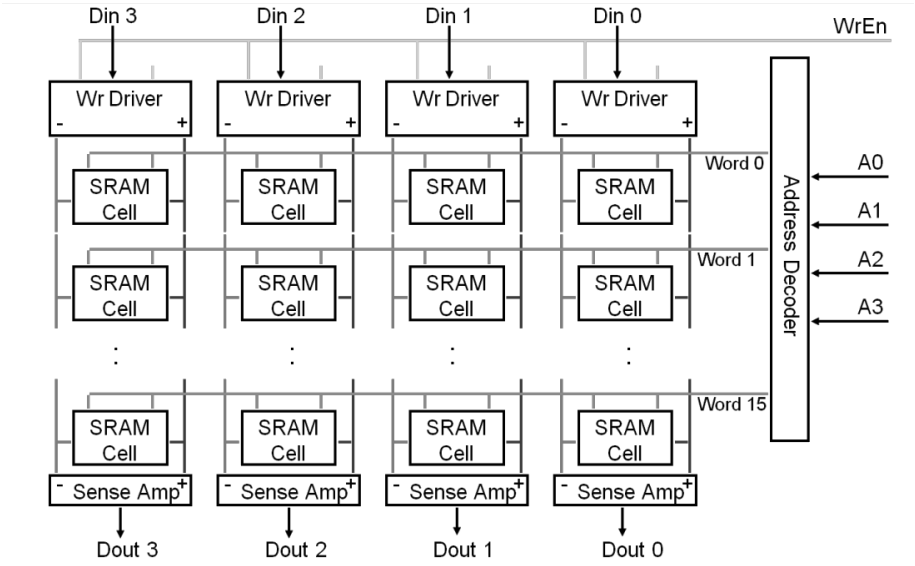


Figure 40: Architettura SRAM 16 Word x 4 bit

1.6.4 CAM

La **Content Addressable Memory** è un speciale tipo di memoria **volatile** utilizzata per ricercare l'indirizzo di un dato (chiamata anche memoria **associativa**) da una tabella di dati memorizzati. La CAM è molto veloce, infatti i suoi applicativi sono le *routing table* o dei particolari tipi di cache.

1.6.5 Mask-Programmed ROM

La **Mask-Programmed Read Only Memory** è una memoria **non volatile** nella quale, tuttavia, si ha solamente l'accesso di lettura. Come indica il nome, la memoria, una volta memorizzati i dati necessari, è realizzata con i connettori direttamente su un livello (maschera) che si applica sul circuito così che non sia modificabile.

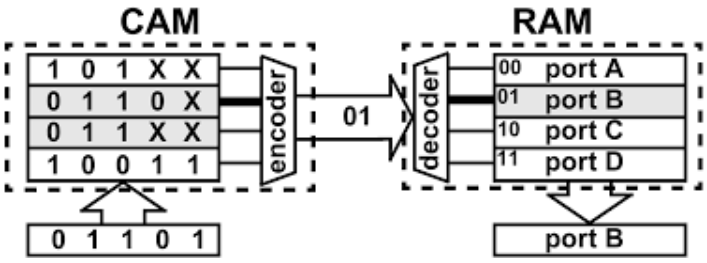


Figure 41: Funzionamento CAM

1.6.6 EPROM

La **Erasable Programmable ROM** è una memoria non **volatile** di sola lettura che, rispetto alla PROM, è modificabile tramite un trattamento a raggi UV, per un numero limitato di volte.

1.6.7 EEPROM

La **Elettrically Erasable Programmable ROM** è una memoria non **volatile** modificabile elettricamente.

1.6.8 FLASH

La memoria **Flash** è un tipo di memoria non volatile di scrittura/lettura. Esso è composto da dei **floating gate MOSFET** che riescono a mantenere al carica elettrica, e quindi anche un bit, per un lungo periodo di tempo. Hanno rivoluzionato le tecnologie inerenti alle memorie in quanto hanno reso possibile il salvataggio o la cancellazione dei dati in un singolo passo grazie a dei software di sistema senza dover utilizzare grandi tensioni o apparecchiature esterne.

Le memorie flash sono di due tipi: flash basate su porte **NOR** e **NAND** che differiscono per architettura e per programmazione.

NOR FLASH: sono delle memorie ad **accesso casuale**, con una **lettura veloce**, ma una **scrittura e cancellazione lenta**.

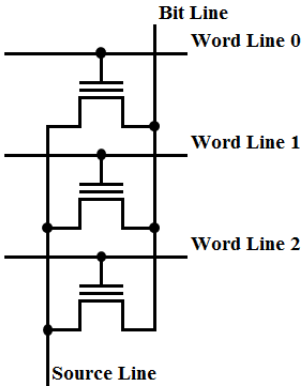


Figure 42: NOR flash

NAND FLASH: sono delle memorie ad **accesso sequenziale**, con un'**alta velocità in scrittura e cancellazione**.

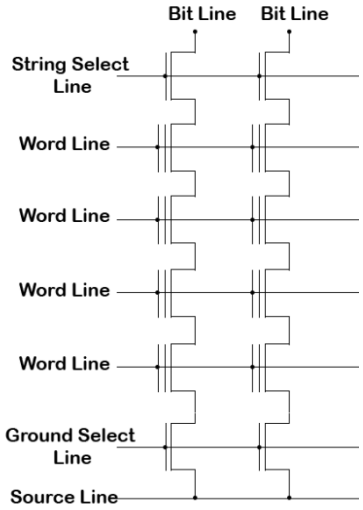


Figure 43: NAND flash

2 Bus e Interconnessioni

Un'interconnessione ideale è vista come un singolo nodo *equipotenziale*, quindi una connessione nella quale l'uscita (**driver**) e l'ingresso (**receiver**) sono equivalenti per tensioni e correnti (tensione/corrente di input ≡ tensione/corrente di output). Tuttavia, nel mondo reale il conduttore della linea di interconnessione non è equipotenziale, né il segnale d'uscita non è un'onda quadra perfetta, in quanto ogni segnale è affetto da **rumore e ritardi**.

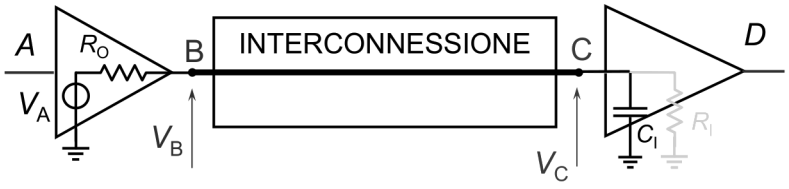


Figure 44: Interfaccia di interconnessione

2.1 Linea di Trasmissione

É un modello di trasmissione più accurato dei modelli RLC ed è necessario per collegamenti lunghi e segnali con transizioni veloci (ripidità dei fronti di clock, non la frequenza).

2.1.1 Parametri elettrici e fisici

I principali parametri **elettrici** per descrivere una linea di trasmissione sono:

1. $\mathbf{Z_\infty}$ impedenza caratteristica
2. $\mathbf{L}$ lunghezza
3. $\mathbf{v_P}$ velocità di propagazione
4. $\mathbf{t_P}$ tempo di propagazione

mentre quelli **fisici** sono:

1. $\mathbf{L_U}$ induttanza unitaria
2. $\mathbf{C_U}$ capacità unitaria

Le relazioni che legano parametri elettrici e fisici sono le seguenti

$$\begin{cases} Z_\infty = \sqrt{\frac{L_U}{C_U}} \\ v_P = \frac{1}{\sqrt{L_U \cdot C_U}} \\ t_P = \frac{L}{v_P} \end{cases} \tag{5}$$

2.1.2 Riflessioni

Il segnale arriva in forma digitale, quindi si forma un gradino di tensione, da 0 V a V_A (tensione del driver), la tensione che esce dal driver è

$$V_B(0) = \frac{Z_\infty}{R_O + Z_\infty} V_A \tag{6}$$

dove R_O è la resistenza del driver. Il segnale si sposta quindi tra i due punti senza subire distorsioni in $t = t_P$.

Una volta giunto alla terminazione (lato remoto), dove è presente una resistenza di terminazione R_T se la resistenza è uguale all'impedenza allora la terminazione assorbe tutta l'energia, altrimenti si genera un'onda **riflessa** che si muove verso il driver.

Introduciamo quindi il **coefficiente di riflessione** Γ_T

$$\Gamma_T = \frac{R_T - Z_\infty}{R_T + Z_\infty} \tag{7}$$

Parliamo di linea **chiusa** su $R_T = Z_\infty$ se $\Gamma_T = 0$ e in questo caso tutta l'energia incidente viene dissipata dall'energia **non** generando alcuna onda riflessa. Invece la linea è **aperta** se $R_T \to \infty$, quindi $\Gamma = 1$ e si genera un'**onda riflessa** di ampiezza pari a quella incidente con la tensione totale alla terminazione raddoppiata. Infine, nel caso in cui la linea sia in **corto** quindi $R_T = 0$ e $\Gamma_T = -1$ la tensione totale alla terminazione è nulla con una tensione dell'onda riflessa uguale e opposta a quella incidente.

Di seguito il circuito completo di una linea di trasmissione dove $V_B(0) = \frac{Z_\infty}{R_O + Z_\infty} V_A = V_1$, $V_2 = \Gamma_T V_1$ e $V_C = V_1 + V_2$.

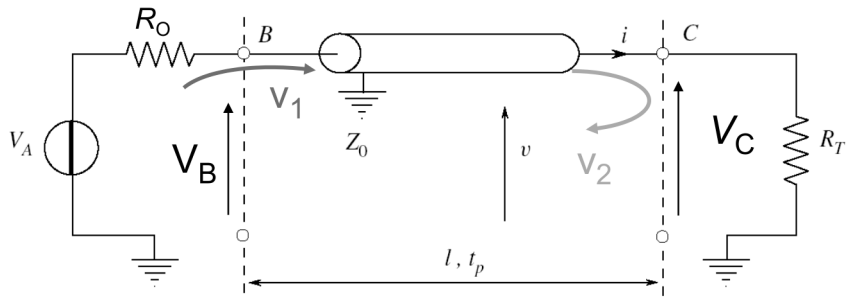


Figure 45: Circuito equivalente

V_1 prende il nome di **onda incidente** e V_2 prende il nome di **onda riflessa** che si propaga verso il driver.

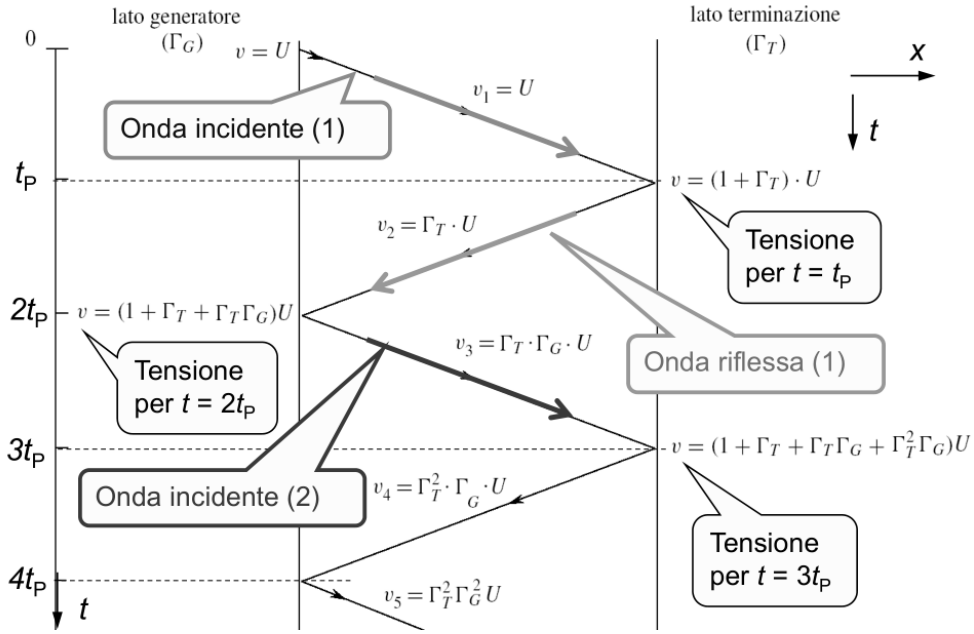


Figure 46: Rappresentazione temporale di onda incidente e riflessa

2.1.3 Parametri temporali

- $\mathbf{t_{TX}}$ tempo di trasmissione: ritardo con cui viene rilevata la variazione di stato logico e in generale $t_{TX} \neq t_P$ in quanto esse sono riconosciute **solo quando viene attraversata il valore di soglia** V_{TH} il che non è garantito dalla propagazione da un estremo all'altro
- $\mathbf{t_K}$ tempo di **skew**: Δt_{TK} ovvero $t_{TXmax} - t_{TXmin}$.

2.1.4 IWS e RWS

La transizione impressa dal driver prende il nome di **primo gradino** che determina un'onda incidente che si propaga lungo la linea. L'ampiezza del gradino dipende da R_O e da Z_∞, in base al loro rapporto si hanno comportamenti diversi:

1. $R_O < Z_\infty$: **primo gradino ampio** il che rende possibile attraversare la soglia con il primo gradino (quindi *basso tempo di trasmissione*); avviene la commutazione sull'onda **incidente** (**I**ncident **W**ave **S**witching).

2. $R_O = Z_\infty$: **driver adattato**, il primo gradino è ari alla metà della tensione a regime e la commutazione avviene sull'onda **riflessa** (**R**eflected **W**ave **S**witching). $\Gamma_D = 0$ quindi non ci sarà nessuna riflessione lato driver.

3. $R_O > Z_\infty$: il primo gradino sarà più basso e non riuscirà ad attraversare la soglia del ricevitore. Le soglie saranno solamente attraversate dopo **riflessioni multiple** (*alto tempo di trasmissione*).

In generale valgono le seguenti relazioni:

Riflessione	$\mathbf{t_{TXmax}}$	$\mathbf{t_K}$
Incident	t_P	t_P
Reflected	$2t_P$	$2t_P$
Multiple	$\sim N \cdot 2t_P$	-

Table 6: Relazione riflessione-temporizzazione

2.2 Cicli di trasferimento

L'obiettivo di un ciclo di trasferimento è quello di garantire un corretto trasferimento di informazione nonostante le variazioni di temporizzazioni causate dal tempo di *skew* (disallineamento temporale). Le principali operazioni di trasferimento di informazioni sono la *scrittura* (attivato dalla sorgente) e la *lettura* (richiesto dalla lettura). Per garantire la correttezza di queste due operazioni è importante rispettare *tempo di skew, hold e setup.*

Ci sono due tecniche base per effettuare dei cicli di trasferimento: **temporizzazione fissa o adattiva**. La prima prevede un protocollo **sincrono** che opera sul **worst case scenario**, mentre la seconda prevede un protocollo **asincrono** con dei segnali di *acknowledge* a regolare il trasferimento.

2.2.1 Ciclo sincrono

In questo protocollo si effettuano le operazioni con dei **ritardi fissi** che possano sempre garantire la ricezione corretta dei dati.

Ciclo Scrittura Sincrono La sorgente (**master**) deve **conoscere le temporizzazioni** della destinazione per effettuare un trasferimento dati corretto che avviene nel seguente modo:

1. la sorgente invia l'informazione

2. aspetta un tempo pari al *tempo di setup più un tempo di skew* per garantire la corretta ricezione al FF

3. invia il segnale di **STB** (strobe) che funge da clock per il FF

4. aspetta un tempo pari al *tempo di hold più un tempo di skew* per garantire il tempo di hold del FF

5. rimuove segnale di informazione e STB

Di seguito il tempo totale di un ciclo di scrittura sincrono

$$t_{WR} = t_{SU} + t_H + 2t_K \tag{8}$$

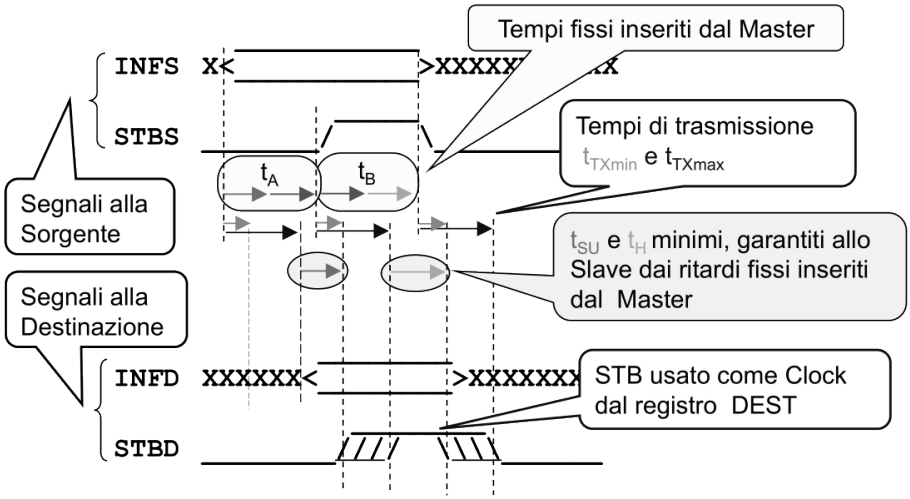


Figure 47: Andamento temporale ciclo di scrittura sincrono

Ciclo Lettura Sincrono Nel ciclo di lettura il master è la **destinazione** (FF) che aggiunge i ritardi necessari per la richiesta allo slave delle informazioni e avviene nel seguente modo:

1. la destinazione invia un segnale di richiesta alla sorgente
2. dopo un tempo di trasmissione è necessario un tempo di **accesso** (ad esempio alla memoria) t_A per reperire il dato
3. la sorgente invia l'informazione che arriva a destinazione con un ritardo pari al tempo di trasmissione
4. la destinazione una volta ricevuto il dato aspetta un tempo di setup richiesto dal FF
5. il FF memorizza l'informazione al fronte di salita
6. la destinazione attende un tempo di hold e rimuove il segnale di richiesta
7. dopo un tempo di trasmissione la sorgente rimuove il segnale di informazione
8. dopo un tempo di trasmissione la rimozione del segnale di informazione giunge alla destinazione

Di seguito il tempo totale di un ciclo di lettura sincrono

$$t_{RD} = t_A + t_{SU} + t_H + 4t_{TXmax} \tag{9}$$

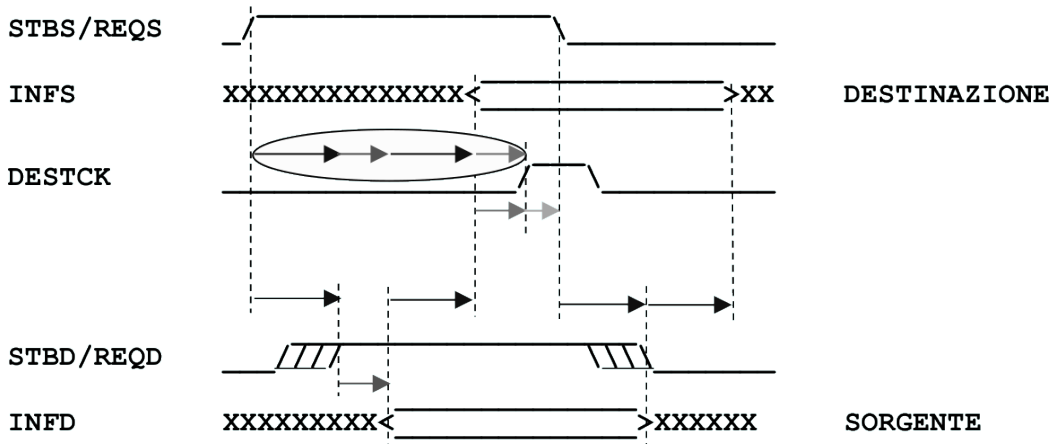


Figure 48: Andamento temporale ciclo di lettura sincrono

2.2.2 Ciclo asincrono

Un ciclo asincrono è caratterizzato dalla presenza di operazioni con **conferma** ovvero segnali di **ACK**, che devono essere regolati da entrambi i moduli (trasmissione e ricezione).

Ciclo Scrittura Asincrono Le operazioni avvengono in una sequenza basata sull'interazione tra sorgente e destinazione effettuando i seguenti passaggi:

1. la sorgente invia l'informazione
2. attende un tempo di *skew* per garantire che il comando di *strobe* (ACK) arrivi alla destinazione sempre dopo il segnale di informazione
3. invia il comando di strobe alla destinazione
4. trascorso un tempo di trasmissione, la logica di controllo attende il tempo di *setup* richiesto dai FF
5. invia il segnale di clock al FF che memorizza l'informazione
6. invia un ACK alla sorgente per informarla che può rimuovere i segnali di informazione e *strobe*
7. trascorso un tempo di trasmissione la sorgente rimuove segnali di informazione e *strobe*
8. trascorso un tempo di trasmissione la logica rimuove il segnale di ACK
9. trascorso un tempo di trasmissione il segnale di ACK giunge disattivato alla sorgente

$$t_{WR} = t_K + t_{SU} + t_H + 4t_{TXmax} \tag{10}$$

É da notare che t_{SU} e t_H sono **garantiti dallo slave** ovvero la destinazione.

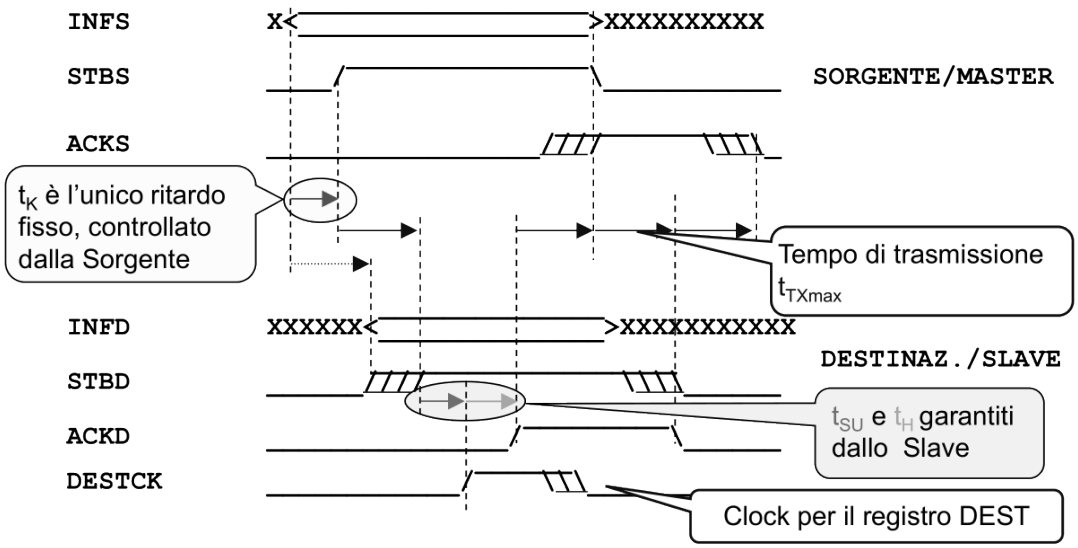


Figure 49: Andamento temporale ciclo di scrittura asincrona

Ciclo Lettura Asincrono Nel ciclo di lettura asincrono si aggiunge un segnale di ACK che la destinazione deve inviare alla sorgente per comunicare che il segnale di informazione è pronto nel seguente modo:

1. la destinazione invia un segnale di richiesta alla sorgente
2. dopo un tempo di trasmissione è necessario un *tempo di accesso* prima che l'informazione richiesta sia pronta
3. la sorgente invia l'informazione
4. dopo un tempo di *skew* la sorgente invia il segnale di ACK alla destinazione

5. dopo un tempo di trasmissione il FF memorizza l'informazione al fronte di salita del segnale di *ACK*
6. dopo un tempo pari ai *tempi di setup e di hold* richiesti dal FF, la destinazione rimuove il segnale di richiesta
7. dopo un tempo di trasmissione la sorgente rimuove i segnali di informazione e di *ACK*
8. dopo un tempo di trasmissione la rimozione dei segnali di informazione e di *ACK* giunge alla destinazione

$$t_{RD} = t_A + t_K + t_{SU} + t_H + 4t_{TXmax} \tag{11}$$

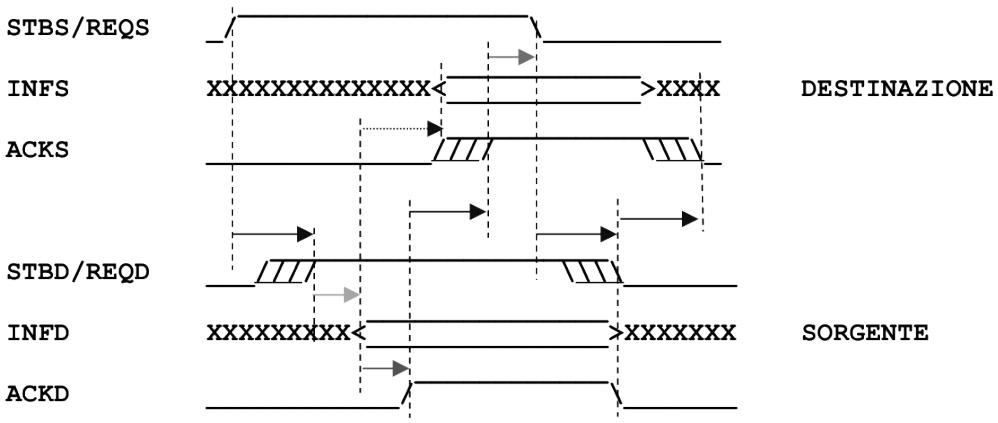


Figure 50: Andamento temporale ciclo di lettura asincrona

2.2.3 Prestazioni di un bus

Le prestazioni di un bus si valutano in base alla **quantità di informazione trasferita nell'unità di tempo**. Essa prende il nome di **throughput** ed è pari a

$$T = W \cdot S$$

dove W è la larghezza del bus, mentre S è la velocità del bus indicata come $\frac{\text{cicli}}{\text{secondo}}$. Inoltre se un ciclo ha durata t_C allora

$$S = \frac{1}{t_C}$$

2.2.4 Miglioramento delle prestazioni

Uno dei migliori protocolli da poter utilizzare per migliorare le prestazioni è il **Source Synchronous** nel quale i segnali controllo sono gestiti da entrambe le unità, sorgente e destinazione così da poter avviare un nuovo trasferimento prima che sia concluso quello precedente (*pipeline*).

Infine, a livello di ciclo, il più efficiente è il **DDR** (Double Data Rate) nel quale i segnali di comando (*STB/ACK*) vengono commutati su entrambe le transizioni così da avere un minor consumo e una maggiore velocità.

3 Sistemi di Acquisizione Dati

I sistemi di acquisizione dati sono dei dispositivi elettronici in grado di rilevare e memorizzare grandezze **analogiche/digitali** per poterle elaborare da un microprocessore. In natura non esistono dei segnali digitali, che tuttavia, sono più facili da analizzare, elaborare e trasmettere. Per questo sono state introdotte tecniche di **conversione** da analogico a digitale (e viceversa).

3.1 Sistemi di Conversione A/D/A

Per passare da un segnale analogico ad uno digitale si effettuano due passaggi: **campionamento e quantizzazione**. Il primo si riferisce ad una discretizzazione in **tempo**, mentre il secondo in **ampiezza**. Uno degli aspetti da non sottovalutare, tuttavia, è la perdita di informazione nella conversione da analogico a digitale.

3.1.1 Campionamento e Aliasing

Il campionamento consiste nella moltiplicazione del segnale analogico per un treno di impulsi.

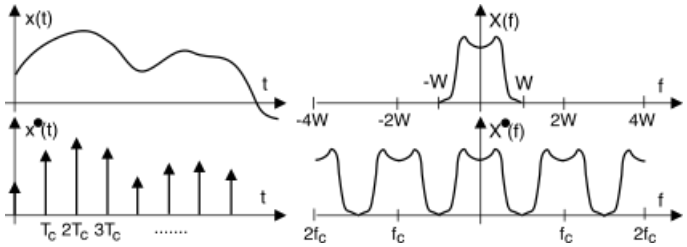


Figure 51: A sinistra un campionamento nel dominio del tempo, a destre nel dominio della frequenza

Come possiamo notare, nello spettro dell'analisi in frequenza del segnale campionato, la **banda fondamentale** si ripete periodicamente per multipli di una frequenza speciale chiamata **cadenza di campionamento**. Per ricostruire il segnale campionato serve un **filtro passa basso** che elimina le bande secondarie. Tuttavia, per far sì che la ricostruzione del segnale avvenga senza perdita di informazioni è necessario che la banda principale e quelle secondarie **non si sovrappongano**. Questo effetto prende il nome di **aliasing** e può essere evitato se si può applicare il teorema di Nyquist (campionamento) il quale afferma che se

$$F_S > 2F_M \tag{12}$$

dove F_S è la frequenza di campionamento e F_M è la frequenza della fondamentale, allora il segnale può essere sempre ricostruito. Tuttavia, una filtro reale non ha attenuazione infinita, sarà sempre presente un **rumore di aliasing** residuo.

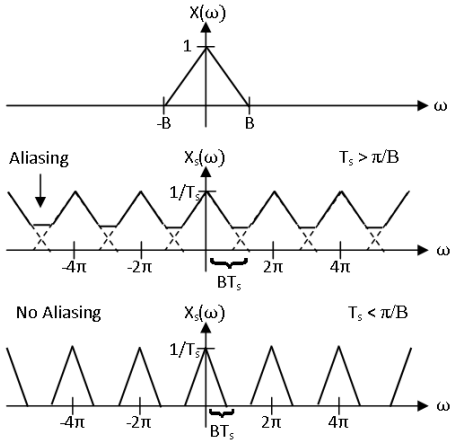


Figure 52: Effetto dell'aliasing

La conversione avviene tramite il modulo chiamato **Sample-Hold**, il cui obiettivo è quello di campionare il segnale in ingresso e mantenerlo stabile durante tutta la conversione fino al campione successivo.

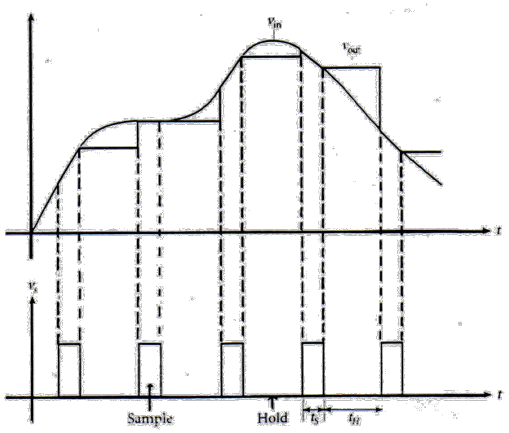


Figure 53: Modulo S/H

Il mantenimento modifica lo spettro del segnale quindi in fase di ricostruzione del segnale originario bisogna tenere in considerazione anche questa modifica che introduce una distorsione, attenuata da una *correzione spettrale*.

3.1.2 Quantizzazione e relativo errore

Un segnale analogico assume infiniti valori in un campo di *tensioni* (*input range* S). Invece un segnale digitale ha una **risoluzione finita**, infatti con N bit si possono esprimere 2^N valori diversi.

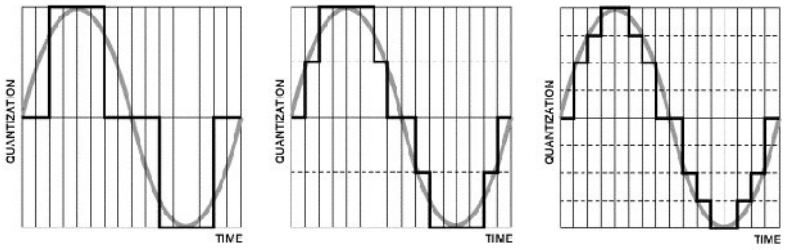


Figure 54: Diversi tipi di quantizzazione

La quantizzazione, tuttavia, introduce un **errore di quantizzazione**, indicato con ϵ_q ed è la differenza fra il segnale analogico reale e il segnale digitale quantizzato dello stesso.

$$|\epsilon_q| \leq \frac{S}{2^{N+1}} \tag{13}$$

Con una quantizzazione *uniforme* (gli intervalli analogici sono tutti uguali) si può anche affermare che

$$|\epsilon_q| \leq \frac{1}{2}\text{LSB}$$

dove LSB è l'ampiezza di ciascun intervallo ($A_d = \frac{S}{2^N} = \text{LSB}$). La qualità del segnale quantizzato è rappresentato dal rapporto tra il segnale e il rumore $\mathbf{SNR}_q$ (*signal-noise ratio*) ed espresso come il rapporto tra la potenza di segnale σ_A^2 e la potenza del rumore di quantizzazione σ_{eq}^2

$$SNR_q = \frac{\sigma_A^2}{\sigma_{eq}^2} \tag{14}$$

dove la potenza del rumore di quantizzazione si può esprimere come

$$\sigma_{eq}^2 = \frac{A_d^2}{12} = \frac{S^2}{12 \cdot 2^{2N}}$$

Per i segnali che arrivano dal *fondo scala* (ovvero l'ampiezza del segnale è uguale al fondo scala) valgono le seguenti relazioni

- segnale **sinusoidale**
$$SNR_q = (6N + 1.76)\text{dB}$$
- segnale **triangolare**
$$SNR_q = 6N\text{dB}$$
- **voce** (distribuzione di potenza gaussiana)
$$SNR_q = (6N - 4.77)\text{dB}$$

La **precisione effettiva**, tuttavia, non dipende esclusivamente dal rumore di quantizzazione, ma dalla totalità degli errori dei moduli presenti (*S/H, quantizzazione,aliasing,...)* e si rappresenta come il rapporto tra il **segnale** e il **rumore totale** e prende il nome di SNR_tot. Da esso deriva anche il concetto di **ENOB** (effective number of bits) che rappresenta il numero di bit effettivamente significativi per il convertitore, e si calcola sostituendo SNR_tot a SNR_q nella formula sinusoidale trovando

$$N = ENOB = \frac{SNR_{tot} - 1.76}{6} \tag{15}$$

3.2 Convertitori D/A

Un convertitore D/A è un componente elettronico in grado di convertire i segnali da **digitale ad analogico**.

3.2.1 Errori

Un **DAC** (digital analogic converter) ha una caratteristica ideale è **lineare** ed è soggetto a due tipologie di errori: **statici** o **dinamici**.

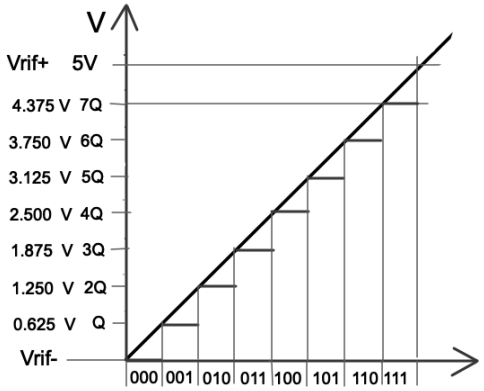


Figure 55: Caratteristica ideale di un convertitore D/A lineare

Errori Statici: riguardano il comportamento a regime, ovvero con segnale ad ingresso costante. Essi sono:

- **Errori di non linearità**: riguardano il confronto tra la caratteristica reale e la retta approssimante. In particolare:

1. l'errore di non-linearità **integrale** ϵ_{nli} è il massimo scostamento *totale* (una fascia d'ampiezza) tra la caratteristica reale e la retta approssimante
2. l'errore di non-linearità **differenziale** ϵ_{nid}, preso un LSB, è lo scostamento *locale* della caratteristica reale dalla retta approssimante

$$\epsilon_{nid} = A_D - A'_D$$

3. se la non-linearità differenziale è maggiore di 1 LSB allora si h errore di **non monotonicità**

- **Errori di linearità**: riguardano il confronto tra la retta approssimante la curva reale e la retta ideale, ma possono essere compensati tramite una taratura del circuito. In particolare:

1. l'**errore di offset** ϵ_0 rappresenta il valore dell'**intercetta** nell'asse delle y (sarà una tensione chiamata V_{off}
2. l'**errore di guadagno** ϵ_g è pari alla **differenza di pendenza** tra la retta approssimante e la retta ideale

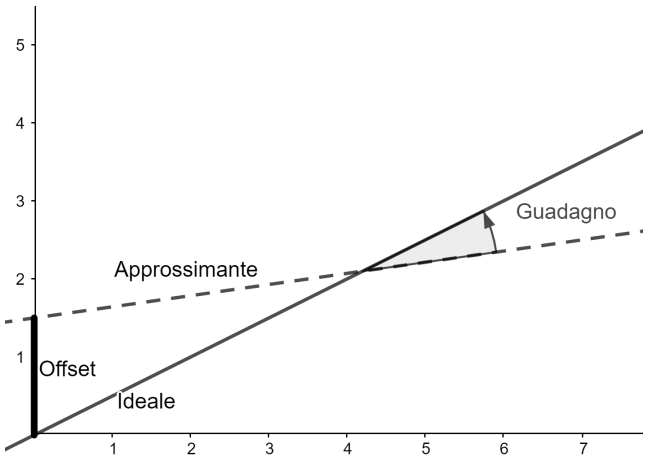


Figure 56: Esempi di errori di linearità

Errori Dinamici: riguardano il comportamento a transitorio, ovvero con un segnale ad ingresso variabile. Essi sono:

- **tempo di assetto** che rappresenta il lasso di tempo nel quale l'uscita del convertitore DAC impiega per **portarsi al nuovo valore**. Il transitorio si considera esaurito quando l'uscita inizia ad oscillare intorno al nuovo valore restando entro la fascia $\pm$ 1 LSB legata all'errore di quantizzazione intrinseco
- **glitch** rappresenta una **variazione sostanziale** dell'uscita in un breve lasso di tempo (dovuto alle differenze nei ritardi commutazione)

3.2.2 Circuiti di conversione D/A

L'uscita analogica A è ottenuta dalla **somma delle grandezze elementari** in cui la grandezza elettrica di riferimento è stata scomposta

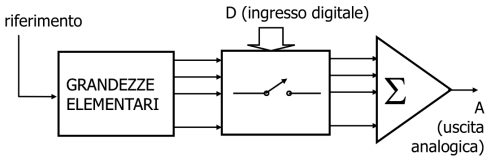


Figure 57: Schema a blocchi di un DAC

Le tecniche di **conversione** di base sono:

- conversione a **grandezze uniformi**: è basata sulla somma di variabili di peso uguale (ovvero 1) inserite in quantità corrispondente al valore dell'ingresso digitale

$$\text{output} = D \cdot LSB$$

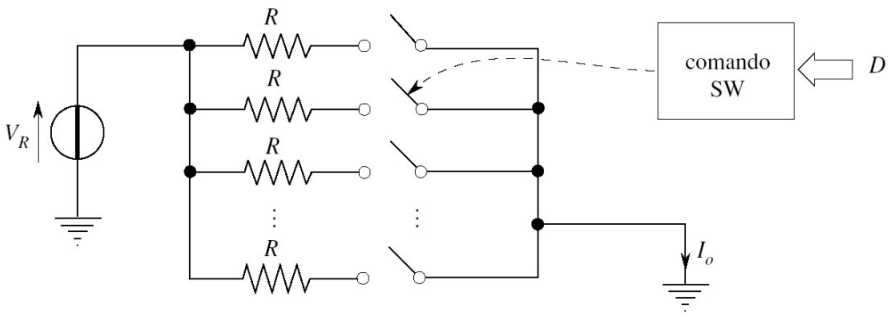


Figure 58: Circuito convertitore a grandezze uniformi

Il circuito è composto da **resistenze tutte uguali** in parallelo il cui ramo può essere aperto o meno in base al valore D. L'**uscita** è una **corrente** ottenuta come somma di correnti uguali.

- conversione a **grandezze pesate**: è basata sulla somma di variabili di peso corrispondente alle potenze di 2

$$\text{output} = 2^i \cdot D_i$$

Il circuito è simile a quello sopra descritto, con la differenza che le resistenze **sono pesate** e hanno la funzione di emulare le potenze di 2 (ad esempio $9 = 2^3 + 2^0$, per indicare il 9 saranno selezionate la resistenza in posizione 0 e quella in posizione 3). Questa configurazione, tuttavia, presenta il difetto di avere una **forte dinamica** dei valori delle resistenze ($0 \longrightarrow 2^{N-1}$).

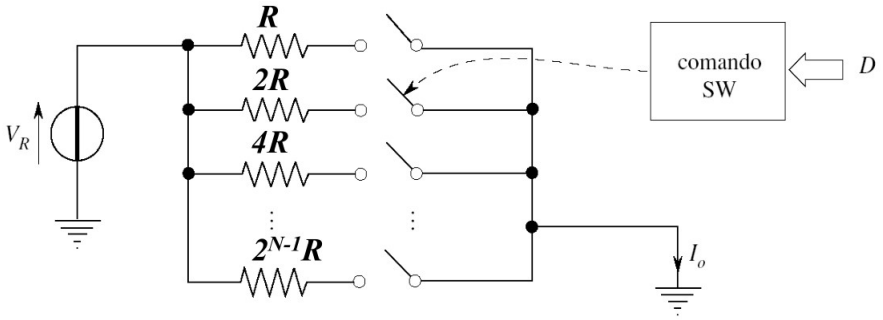


Figure 59: Circuito convertitore a grandezze pesate

Il problema della forte dinamica può essere ovviato grazie ad una configurazione particolare chiamata **rete a scala**. Di seguito una rappresentazione.

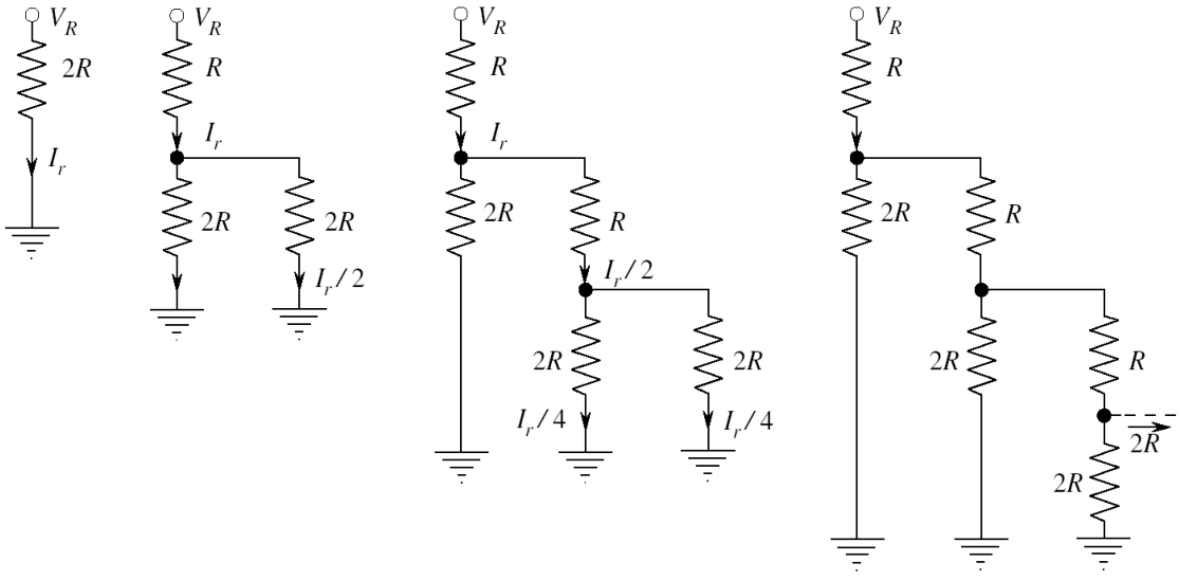


Figure 60: Genesi della rete a scala

La creazione di questa rete a scala è molto facile, basta **dividere in due** la corrente che circola nel ramo più a destra. Questa configurazione, oltre a **diminuire la dinamica**, infatti usa **solamente resistenze di valore R e 2R**, essa può essere espansa **senza vincoli** sui valori delle resistenze ed infine con la conversione Norton/Thevenin si possono ricavare **uscite in tensione**.

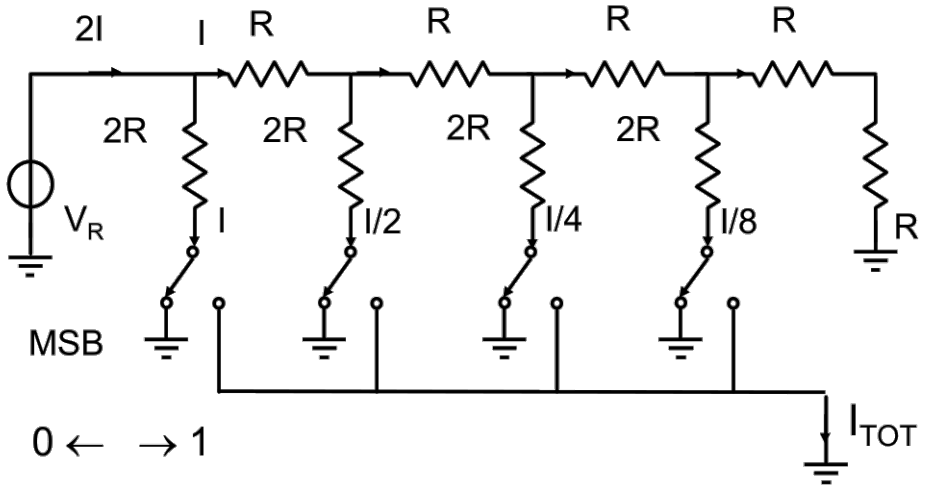


Figure 61: Un altro modo di rappresentare la rete a scala

La *corrente totale* avrà una forma del tipo $I_{TOT} = I + \frac{I}{4} + \frac{I}{8}$ che assume la rappresentazione binaria **1011**. Inoltre, le grandezze da sommare possono essere delle **cariche elettriche**, quindi utilizzare dei **condensatori**) al posto delle resistenze.

Il parametro più **critico** che può portare a degli errori non indifferenti (causa un errore di **non-linearità**) è la **precisione** dei rami **MSB** in quanto, **maggiore** è il peso di un ramo, più grande è l'errore in uscita (*quindi diversi rami pesati determinano errori differenti in uscita*).

3.3 Sistemi di Conversione A/D

Un **ADC** (Analog to Digital Converter) è un circuito elettrico in grado di convertire un segnale **continuo** come una tensione, in un segnale **discreto**. Esistono diversi tipi di convertitori classificabili in base a **velocità** (legato al tempo di conversione T_C in quanto rappresenta il *numero di conversioni al secondo*) e **complessità** (legata al numero di **comparatori** utilizzati). I due parametri sono **inversamente proporzionali**, quando aumenta la complessità diminuisce il ritardo e viceversa.

3.3.1 ADC Flash Parallelo

É uno dei convertitori più **veloci in assoluto** e in relazione al segnale analogico in ingresso produce un'uscita lineare che poi sarà convertita in valori binari da un **encoder**. É formato da 2^{N-1} comparatori che in uscita producono uno 0 oppure un 1 in base alla comparazione con il valore di **soglia**.

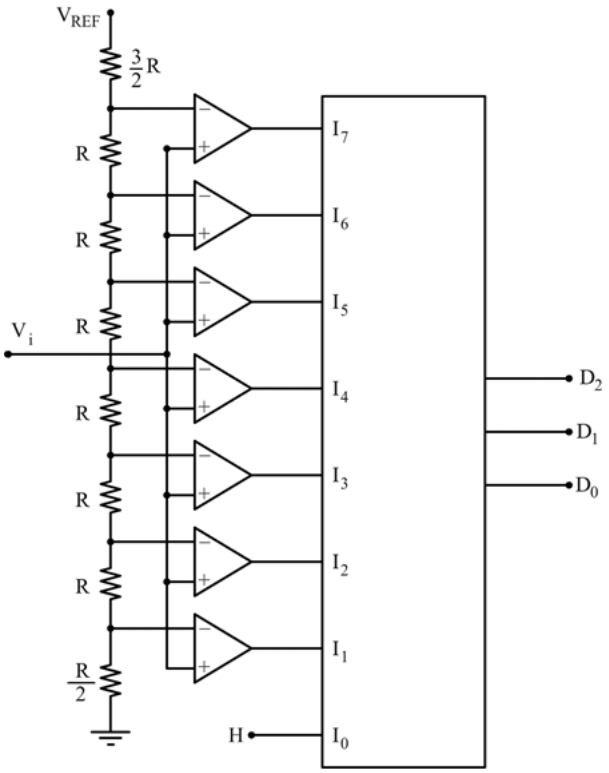


Figure 62: Circuito ADC Flash

Opera in **parallelo**, tutti i comparatori decidono contemporaneamente il valore in uscita comparato alla soglia, infatti occorre **solamente un ciclo di confronto** per convertire N bit.

3.3.2 Convertitore ad Inseguimento

É un circuito che converte l'uscita D con un DAC in **reazione** così da poter confrontare il segnale approssimato A' con il segnale in ingresso A. Ad ogni colpo di clock il contatore viene incrementato in base al rapporto A/A'. In particolare:

- se $A' < A$ il contatore viene incrementato di 1 LSB
- se $A' > A$ il contatore viene decrementato di un LSB

Si procede in questo modo finché il segnale di reazione A' non raggiunge la migliore approssimazione di A.

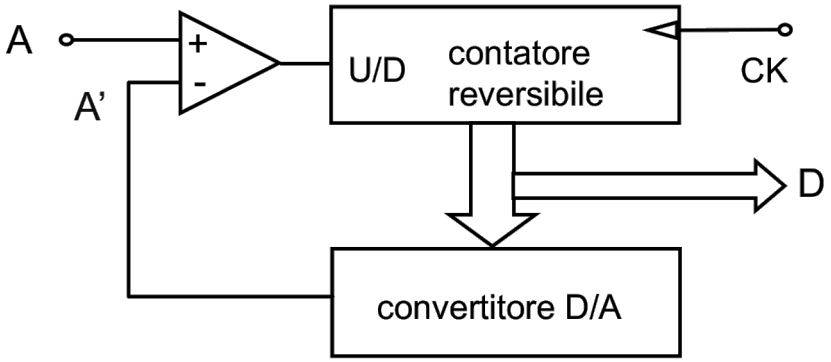


Figure 63: Schema di un ADC ad inseguimento

É **molto lento** in quanto nel caso peggiore occorrono 2^N cicli di confronto per convertire N bit, ma è **semplice** in quanto occorre solamente un comparatore.

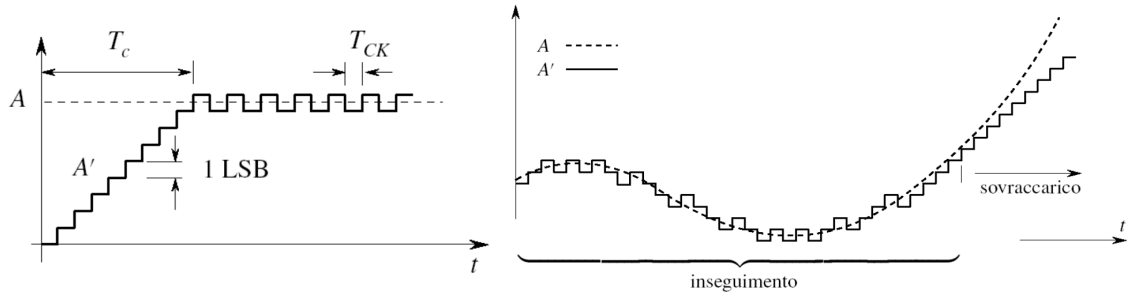


Figure 64: L'inseguimento può essere *costante* o *variabile*

3.3.3 Convertitore ad Approssimazioni Successive

É un circuito con reazione che a ogni colpo di clock determina il valore di un bit consecutivamente a partire dal MSB fino al LSB: l'ingresso A viene ogni volta confrontato con **la metà del campo rimasto** (inizialmente il campo è pari al fondo scala **S**). In particolare:

- se A si trova nella metà inferiore del campo rimasto il bit è 0 e il nuovo campo corrente è la metà inferiore
- se A si trova nella metà superiore del campo rimasto il bit è 1 e il nuovo campo corrente è la metà superiore

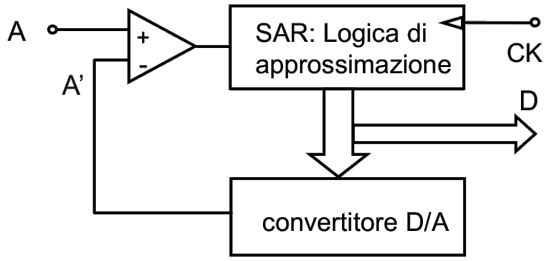


Figure 65: Schema di un ADC ad approssimazioni successive

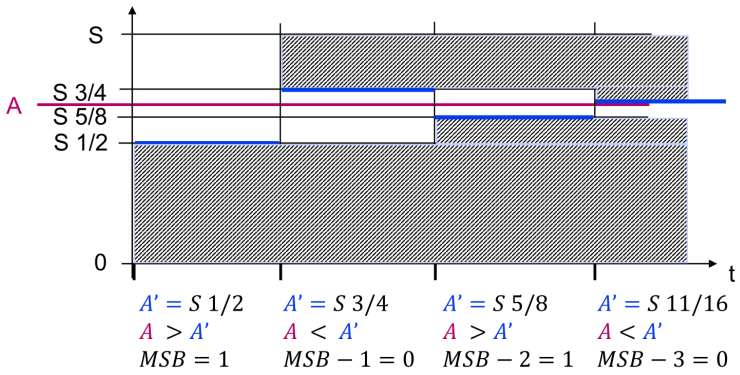


Figure 66: Andamento del campo corrente

É **lento** in quanto nel caso peggiore occorrono N cicli di confronto per convertire N bit, ma è **semplice** in quanto occorre solamente un comparatore.

3.4 Multiplexer e Sample/Hold

3.4.1 Multiplexer

Il **multiplexer** è un **banco di interruttori** realizzato con transistori MOS e tramite un comando di selezione viene chiuso solamente un interruttore per selezionare il canale di ingresso. Gli elementi che producono effetti di **non-idealità** sono:

- la **resistenza equivalente** R_{ON} per l'interruttore chiuso
- la **corrente di perdita** I_{OFF} per gli interruttori aperti
- la **capacità parassita** C_P costituita dalle capacità parassite del multiplexer e del carico

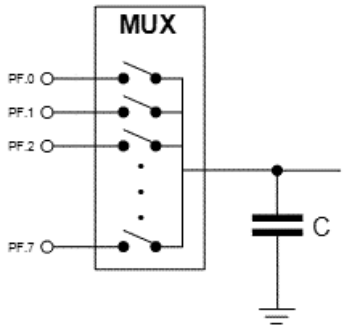


Figure 67: Schema di un multiplexer

3.4.2 Sample/Hold

Il circuito **Sample and Hold** (abbreviato **S&H**) è un campionatore utilizzato come interfaccia tra un **segnale analogico** che varia velocemente nel tempo e un dispositivo successivo, spesso un convertitore analogico-digitale. L'effetto di questo circuito è di **mantenere il valore analogico costante** per il tempo necessario al convertitore. Le fasi operative di questo modulo sono le seguenti:

- acquisizione e inseguimento, **track**. L'uscita è uguale all'ingresso, quindi il S&H diventa un **amplificatore con guadagno unitario**. In questa fase abbiamo degli errori **statici** (*guadagno, offset, non-linearità* e **dinamici** (*tempo di settling, limiti di banda*).

- campionamento, **sample**. Viene **campionato il valore dell'ingresso** da mantenere costante (anche se non avviene in maniera istantanea). In questa fase i due parametri da tenere in considerazione sono il **tempo di apertura** (tempo che trascorre dal comando di hold alla completa apertura dell'interruttore) e il **tempo di assetto** (tempo in cui l'uscita scende, con una variazione di ampiezza chiamata *piedistallo*, a un valore più basso di quello campionato).

  Il tempo di assetto è affetto da **rumore** dovuto dal **Jitter di campionamento** che produce un errore in ampiezza del segnale. Possiamo, quindi, valutare il **SNRj** (signal to noise ratio) di jitter calcolando il rapporto tra segnale e rumore dovuto.

- mantenimento, **hold**. Il valore campionato viene **mantenuto costante** durante la conversione. In questa fase abbiamo degli errori di **decadimento** (il valore dell'uscita diminuisce nel tempo) e di **feedthrough** (una minima parte delle variazioni dell'ingresso viene riportata sull'uscita a causa dell'imperfetto isolamento).

- **nuova acquisizione**. L'uscita passa dal valore mantenuto al valore **corrente dell'ingresso**.

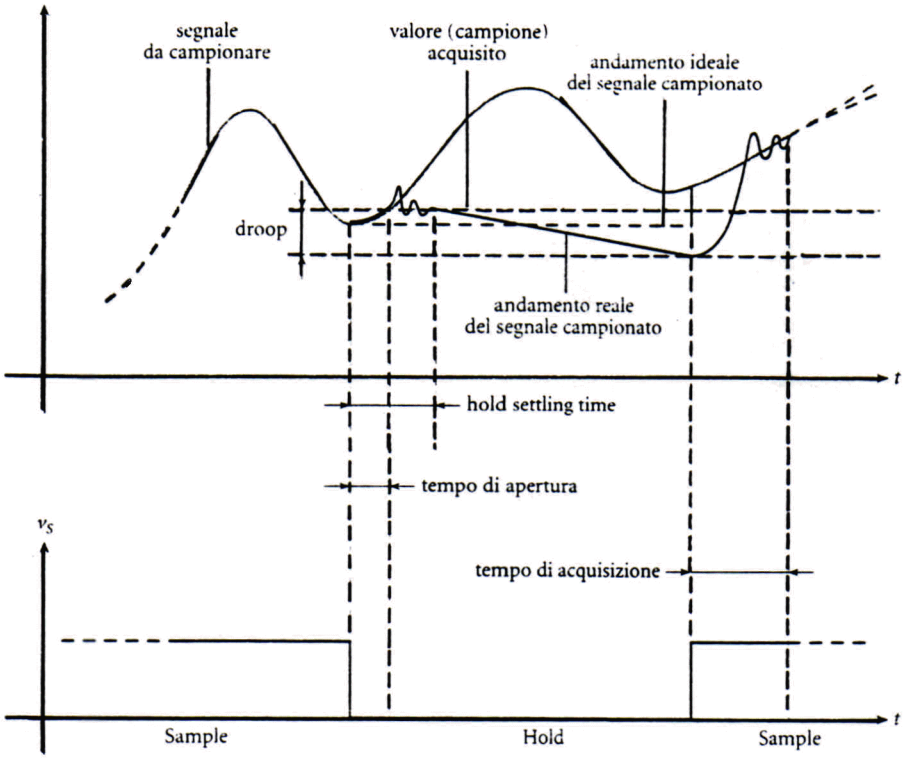


Figure 68: Tempistiche riguardanti il S&H

3.5 Condizionamento del segnale

Il convertitore A/D può avere diversi tipi di ingresso

- tensione o corrente
- riferito a massa o differenziale

Per adattare il segnale di ingresso all'A/D, come dinamica e come tipo di segnale, si inserisce un amplificatore di condizionamento (di diversi tipi).

3.5.1 Circuiti di protezione

Il segnale proviene dall'esterno è soggetto a cariche statiche, disturbi elettromagnetici, rumore, contatti accidentali e così via. Per non danneggiare i componenti del circuito è necessario, quindi, **limitare la tensione di ingresso**. I circuiti utilizzati per proteggere il circuito sono dei **clamp a diodi** verso massa/alimentazione, **diodi Zener** e altri componenti speciali. Tutti questi circuiti **limitano** la tensione di uscita tra valori specifici.

3.5.2 Filtri anti-aliasing

Il filtro anti-aliasing è un filtro analogico utilizzato prima del campionamento di un segnale, al fine di restringere la banda del segnale stesso per soddisfare approssimativamente il teorema del campionamento di Nyquist. Sapendo che f_B è la **banda del filtro di ricostruzione** e f_S è la **cadenza di campionamento**, il filtro:

- non deve attenuare fino a f_B
- attenuare di SNR_A a $f_S - f_B$
- da f_B a $f_S - f_B$ la dinamica è $\frac{f_S - f_B}{f_B}$. In questo range, inoltre, ogni polo attenua di

$$A_P = 20\log_{10}\frac{f_S - f_B}{f_B}\text{dB}$$

Inoltre il numero di poli necessari per progettare un filtro si calcola

$$P = \frac{SNR_A}{A_P}$$

3.5.3 Errore totale e SNR

Il rapporto segnale-rumore totale SNR_{TOT} di un sistema di conversione A/D dipende da vari errori, tra cui:

- **errore di quantizzazione**: legato al numero di bit del convertitore A/D
- **rumore di aliasing**: dovuto alle sovrapposizioni degli spettri del segnale campionato
- **errore di jitter**: introdotto dal modulo di sample e hold in fase di campionamento
- altri errori della catena di **condizionamento** (ad es. errore di guadagno degli amplificatori).

$$SNR_{TOT} = 10\log_{10}\frac{1}{\sum A_i} \tag{16}$$

4 Circuiti di Potenza

4.1 Raddrizzatore

Il **raddrizzatore** è un dispositivo che viene utilizzato per trasformare la corrente elettrica **da alternata in continua**.

4.1.1 Raddrizzatore a singola semionda

Il segnale d'ingresso, sinusoidale, viene applicato a un **diodo** in serie alla resistenza di carico. Se il catodo è rivolto verso il carico, il diodo consente il passaggio delle sole semionde positive, lasciando a zero il valore della tensione in corrispondenza delle semionde negative.

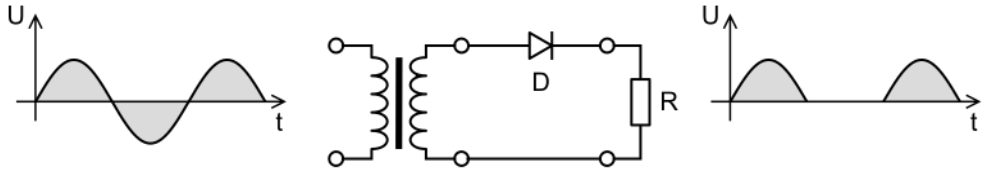


Figure 69: Raddrizzatore a singola semionda

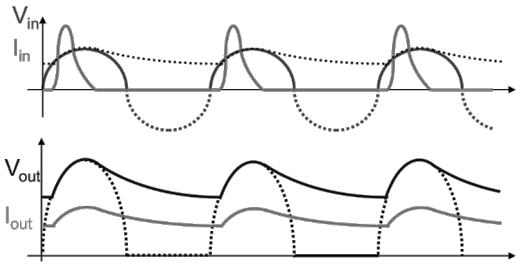


Figure 70: Raddrizzatore a singola semionda

I parametri fondamentali di questa tipologia di raddrizzatore sono:

- **tensione di ripple**
$$V_{ri} = \frac{I_O}{fC}$$
- **tensione di picco**
$$V_{pi} = V_{in} - V_d$$
dove V_d è uguale alla caduta di tensione singola su un diodo.
- **valore medio**
$$V_m = \frac{V_{pi}}{\pi}$$
- **valore efficace**
$$V_{eff} = \frac{V_{pi}}{2}$$

4.1.2 Raddrizzatore a semionda intera

É un circuito formato da **due diodi** che trasforma un'onda alternata in un'onda sempre positiva, "capovolgendo" la semionda negativa. I parametri fondamentali di questa tipologia di raddrizzatore sono:

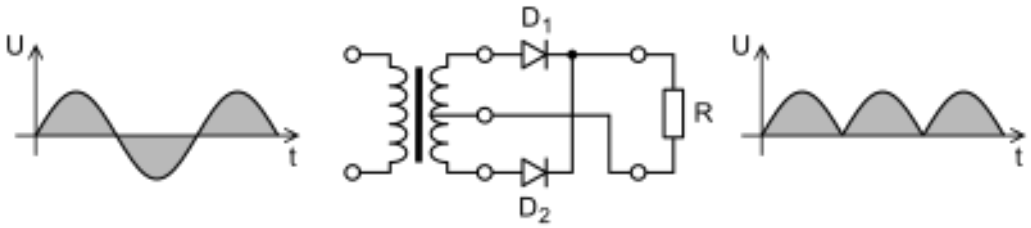


Figure 71: Raddrizzatore a doppia semionda

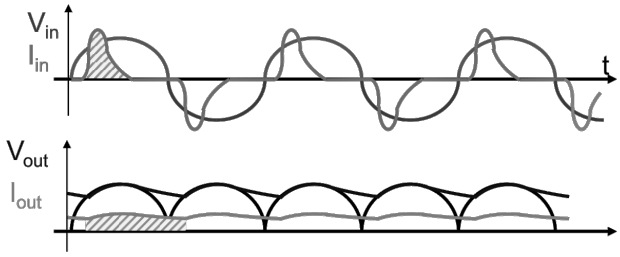


Figure 72: Raddrizzatore a doppia semionda

- **tensione di ripple**
$$V_{ri} = \frac{I_O}{2fC}$$
- **tensione di picco**
$$V_{pi} = V_{in} - 2V_d$$
dove V_d è uguale alla caduta di tensione doppia sui diodi.
- **valore medio**
$$V_m = \frac{2V_{pi}}{\pi}$$
- **valore efficace**
$$V_{eff} = \frac{V_{pi}}{\sqrt{2}}$$

4.2 Regolatori

Un regolatore è un dispositivo elettronico che **converte** una sorgente di corrente continua **da una tensione ad un'altra**, mantenendo la tensione di uscita **costante** indipendentemente dal valore della tensione in ingresso.

4.2.1 Regolatori serie

I regolatori in serie pongono in serie al carico un componente capace di offrire una resistenza variabile (generalmente, un transistor), il cui valore viene costantemente regolato da una **retroazione negativa**. Le correnti e soprattutto le tensioni di perdita nel regolatore serie causano una perdita di potenza da cui possiamo affermare che l'efficienza vale:
$$\eta = \frac{V_{out}}{V_{in}}$$

4.2.2 Regolatori a commutazione

Il sistema di alimentazione a commutazione è simile a quello lineare, ma con alcune differenze:

- il regolatore è a **commutazione**, ovvero l'elemento base è l'interruttore
- il trasformatore è posto dopo il regolatore per poter lavorare a **frequenze più alte**
- nella parte di controllo in retroazione, il segnale a onda quadra che commuta l'interruttore è fornito da un oscillatore digitale

I regolatori a commutazione hanno un rendimento η pari a **0.8÷0.9 V** e più alto dei regolatori lineari, grazie alle basse perdite dell'interruttore e della cella LC.

Si hanno diverse tipologie di regolatori a commutazione e definito D il **duty cycle** avremo:

- **buck** $\eta = \mathrm{D}$
- **boost** $\eta = \frac{1}{1-\mathrm{D}}$
- **buck-boost** $\eta = -\frac{\mathrm{D}}{1-\mathrm{D}}$