Appunti Elettronica applicata

Brendon Mendicino

November 19, 2022

Contents

1 Circuiti Logici

I circuiti logici si alimentano tramite una tensione continua, i circuiti odierni usano una tensione al di sotto del Volt, il motivo è che la potenza generata dipende principalmente dalla tensione di alimentazione. Una volta i transistor venivano alimentati con circa $5V$.

Gli stati logici di uscita devono prensentare il minor rumore possibile, sono codificati con una tensione alta (vero; 1) o tensione bassa (falso; 0). L'uscita a livello alto di solito si approssima alla tensione di alimentazione V_{dd} e l'uscita bassa al GND. Le tensioni all'interno di un circuito logico al di sopra o al di sotto di una certa soglia V_T vengono assegnate ad un livello basso o ad un livello alto, a differenza si trovano al di sopra o al di sotto di tale soglia. Il motivo di questa scelta è dovuta al rumore, che porterebbe ad un errore nel passaggio delle informazioni.

Le soglie non sono ben definite per tutte le porte logiche, la soluzione che viene adottata all'interno di un circuito integrato è di definire degli intervalli in cui le soglie dei vari transistor si trovano all'interno, questi intevalli compresi nell'intevallo (V_{dd}, GND). Per motivi simili l'uscita di una porta non è detto che sia sempre a V_{dd} o GND (una causa può essre il rumore), si definiscono allora dei valori limite in cui la tensione non poò essere più bassa o più alta di questa soglia.

Per le uscite di sefiniscono V_{OH} e V_{OL}, metre per le entrate si definisce V_{IH} e V_{IL}, garantire queste soglie viene dette **contrratto statico**, in cui il parlatore e l'ascoltore riescono a comunicare. È sempre vero che: $V_{OH} > V_{IH}$, $V_{OL} < V_{IL}$.

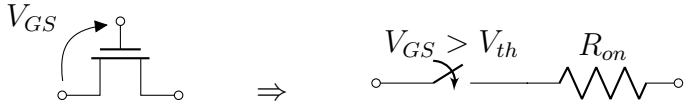
A causa del rumore le uscite possono variare, per questo le soglie di uscita devono avere dei margini di rumore, per evitare che le tensioni vadano a finire nella zona non definita in cui lo stato logico è indefinito.

All'interno di ogni porta logica c'è un comparatore che avrà una soglia (non ben definita), che converte un segnale analogico in un segnale digitale.

L'esmpio di porta logica più facile è l'**inverter**: porta un segnale da basso ad alto o da alto a basso. Andando in laboratorio e facendo delle misurazioni non si potrà mai trovare con precisione la soglia V_T ma è possibile farsi un'idea abbastanza precisa di essa possa trovarsi.

1.1 Transistor MOS

Nei circuiti logici i MOS possono essere modellati come uno switch in serie con una resistenza.



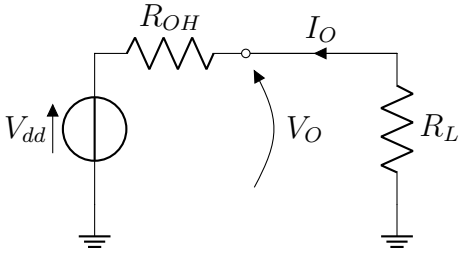
Quando un transistor sarebbe nominalmente spento, ovvero V_{GS} è sotto la soglia V_{th}, una piccola corrente circola comunque (dell'ordine di $10^{-12}A$). Queste correnti

vengono dette **correnti di perdita**, a causa di esse, data dai miliardi di transistor all'interno di un circuito integrato, sorge il problema della potenza comsumata che diventa troppo elevata.
Per creare un invertitore si usano un $pMOS$ ed un $nMOS$. Si è visto empiricamente che è molto efficente (più insensibile al rumore), usare i $pMOS$ per portare l'uscita a livello alto, mentre si usano gli $nMOS$ per portare l'uscita a livello basso.

Un transistor spento è un circuito aperto, un transistor in regione lineare si comporta come un resistore. Analizzare la corrente di perdita si fa solo quando il circuito integrato presenta miliardi di transistor, e la potenza generata diventa rilevante.
La tensione di soglia V_T varia con l'alimentazione, tipicamente se l'alimentazione si alza, la tensione di soglia si alza e viceversa.
Per creare delle funzioni logiche gli inverter non bastano, gli inverter possono, al mossimo diminuire il rumore.
Per realizzare i circiuti logici si uasno i $cMOS$ (complementary MOS). I $cMOS$ si dividono in rete di pull-up (fatta da $pMOS$), ed una rete di pull-down (fatta da $nMOS$). Il motivo per cui questi circuiti vengono detti *complementary* è dovuta al fatto che le funzioni di pull contengono solo un tipo di transistor (per evitare che il pull-up ed il pull-down siano in conduzione contemporaneamente), quindi le due funzioni sono una il complementare dell'altra. Una porta logica realizzata con i transistor MOS è, per sua natura, sempre invertente, questo è dovuto al fatto che la rete di pull-down è realizzata con $nMOS$ e la rete di pull-up è la sua complementare.
Per usare funzioni logiche più complesse si mettono insieme più porte logiche, perché creare una rete combinatoria con molti ingressi è molto costoso e complesso.

1.2 Parametri Statici nei Circuiti a Transistor

Si prenda come esempio l'uscita su un carico resistivo, rappresenta l'uscita su un led o una lampadina. Supponiamo di lavorare a livello logico alto (a tensione di alimentazione).



Ci sarà una tesione sulla resistenza di uscita $V_O = V_{dd} + R_{OH} \cdot I_O$. Per garantire che questa tensione si mantenga al di sopra della V_{OH} dobbiama fare in modo che la corrente in modulo sia minore di un valore massimo. Il parametro in uscita I_{OH} è la massima corrente per cui la soglia del livello alto viene garantita. Ci saranno delle zone di corretto funzionamento, dove a seconda della tensione e la corrente di uscita

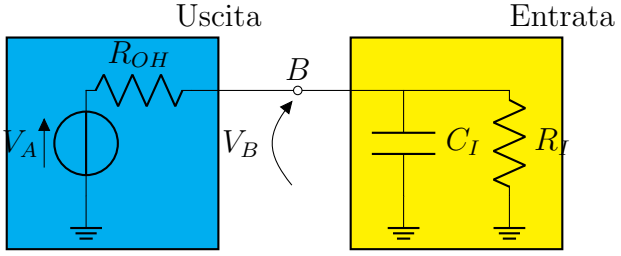
si potrà rientrare o meno dei parametri forniti dal costruttore.
Le porte viste fino ad ora sono dette **totem pole** (se si gurda il cicuito dall'uscita si posso vedere tante faccine, richiamando i totem americani), che sono in maggior parte presente nei circuiti logici, sono modellabile come uno switch tra il livello alto ed il livello basso.
Esistono anche delle porte logiche che oltre ad il livello alto e basso hanno un terzo stato, usate per regolare più parlatori su uno stesso *BUS*. Queste porte vengono dette **buffer tri-state**, non vengono usate nei circuiti integrati.
Esiste anche un altro tipo di porta che contiene solo un $nMOS$ e serve a collegare più disposistivi in parallelo su un BUS seriale, detto **open drain**.

1.3 Parametri Dinamici

Cosa accade quando i segnali variano nel tempo? I segnali reali non hanno pendenza infinita, ci sarà un tempo di commutazione tra gli stati. Sono definiti quattro ritardi fondamentali nei contratti:

- tempo di salita: t_r;
- tempo di discesa: t_f;
- tempo di propagazione;

I **tempi di transizione** sono i tempi di salita e di discesa, si definiscono tra il 10% ed il 90% dell'escursione totale. I **tempi di ritardo** si definiscono tra ritardi di propagazione da ingresso a uscita, questi tempi di propagazione si definiscono da il 50% dell'escursione tra la transizione di entrata ed in uscita.
Da dove arrivano la capacità che vanno caricate? Esistono due tipi capacità: **gate**, **interconnessione** (ignorate in questo corso).
Le capacità del transistor MOS sono poste ad ogni nodo, compreso il nodo *bulk* (di solito non viene mostrato). L'unica capacità utile del MOS è la **capacità di gate**, tra il gate ed il bulk, tutte le altre sono capacità parassite.
Durante l'analisi dei transitori il circuito equivalente può essere considerato come una capacità C_I collegata a massa, si avranno due capacità in parallelo, una del pull-up e una del pull-down.
Il modello di taransizione da basso ad alto tra due porte logiche può essere rappresentato nel modo seguente.



Il ritardo è dovuto alla carica o all scarica di C_I. La tensione V_B è un esponenziale, che effettua una transizione tra V_{OL} e V_{OH}. Quando la tensione supera la tensione di soglia V_T, lo stato commuta. Ci interessano i tempi di ritardo massimo ed tempi di ritardo minino.
Analogamente per la transizione da alto a basso, la resistenza di uscita R_{OL} saraà pilota a massa e la V_B deve sempre essere attraversata per commutare di stato. Un osservazione interessante è il fatto il tempo di discesa non dipende dalle tensioni gioco, ma solo dalla capacità C_I e della reisistenza R_{OL}. Infatti il tempo di propagazine da stato alto a stato basso è:

Definition 1.1 – Tempo di Discesa

$$t_{pHL} = f(R_{OL} \cdot C_I) = 0.69 R_{OL} C_I$$

Nel caso di una struttura totem pole, la R_{OH} e la R_{OL} hanno dei volori molto simili se pur diversi, dunque i tempi di salita e di sicesa sono molto simili.
In una struttura open drain, i tempi di salita sono molto maggiori dei tempi di discesa.

Cosa accade quando si pilotano più porte logiche da una sola uscita? Le capacità sono in parallelo e vengo sommate, dunque con n porte logiche in parallelo, si avrà un ritardo n-volte maggiore. Nella realtà non si pilotano più di quattro porte contemporaneamente. Se la transizione è molto lenta, si rimana molto tempo nella zona di incertezza, questo porta a dei problemi, perché alcune porte potrebbero leggere 0 o 1. Quando il segnale è nella zona di incertezza non si ha margine di rumore, dunque le transizioni devono essere il più veloci possibili.
Quando si hanno delle porte pass-gate, tra due porte logiche, i tempi di ritardo vengono sensibilmente allungati. Queste porte portano ad avere una resistsenza in cascata con la resistenza in ingresso con una capacità parassita in parallelo.

Per ridurre i ritardi sarebbe necessario ridurre la reisitenza di uscita e la capacità di carico, ridurle entrambe non è possibile, il motivo è che ridurre la capacità di carico aumenta la resistenza dello stadio successivo. Velocizzare i circuiti consiste nel bilanciare questi due valori.

Il massimo numero di ingressi che posso essere collegati ad un'uscita è detta *FANOUT*.
Per collegare due sisitemi che si trovano a lunga distanza si usa una codifica differenziale, in modo da ridurre l'effetto dei disturbi. Nel caso di comunicazioni a distanze elevate si preferisce un passaggio di informazione seriale piuttosto che in parallelo, questo perché i segnali seriali viaggiano a magiori velocictaà (alti Hz) di quelli in parallelo.

Quando si hanno una porta pilotante e una o più porte pilotate, la tensione viene decisa dal pilotante e la corrente viene decisa dal pilotato (deve essere la più piccola possibile). Le correnti tra dei *cMOS* la corrente entra in gioco solo in codizione dinamiche, non statiche. Per analizzare le compatibilità tra due porte bisogna:

1. Verificare la compatibilità a livello di tensione: $V_{OL} < V_{IL}, \quad V_{OH} > V_{IH}$;

2. Calcora la correnti richieste per lo stato alto e basso:
   - trascurabili per i MOS
   - non-trascurabili per BJT
3. Verificare che le correnti in modulo dei ricevitori sia minore di quella che il parlatore può erogare

Example 1.1

Schema:

- vecchie porte logiche (assorbono corrente in condizioni statiche)
- $R1 = 10k\Omega$ collegato al GND
- $R2 = 1k\Omega$ collegato a V_{dd}
- quatro porte collegate all'uscita
- $V_{OL} = 0.4V, \quad V_{OH} = 3V, \quad I_{OL} = 4mA, \quad I_{OH} = -1mA$
- $V_{IL} = 0.8V, \quad V_{IH} = 2V, \quad I_{IL} = -0.8mA, \quad I_{IH} = 0.2mA$
- $V_{dd} = 5V$

Il motivo per cui si hanno due resistenze come carico, è dovuto dal fatto che se pur aumentando il consumo di energia si può diminuire il tempo di ritardo.
Tensioni:

- Stato L: $V_{OL} < V_{IL}; 0.4V < 0.8V$
- Stato H: $V_{OH} > V_{IH}; 3V > 2V$

Correnti:

correnti	ALTO	BASSO	R1 ALTO	R2 BASSO
solo porte	$4 \cdot 0.2mA$	$4 \cdot 0.8mA$	$0.8mA$	$3.2mA$
resistenze	0	0	$\frac{V_{dd}}{10k\Omega} = 0.5mA$	$\frac{V_{dd}}{1k\Omega} = 5mA$
totale	$0.8mA$	$3.2mA$	$1.3mA$	$8.2mA$
verifica	$1mA$	$4mA$	$1mA$	$4mA$

Per le risistenze viene considerato il caso peggiore possibile, ovvero la tensione di alimentazione $5V$. Come è possibile vedere le correnti assorbite in presenza di $R1$ ed $R2$ è maggiore in modulo della correte massima erogabile dall'uscita. Due possibilli soluzione potrebbero essere: aumentare le resistenze; combiare le porte logiche con tollerazne più alte.

Example 1.2

Calcolare t_{DLH} tra due porte con:

$$V_{OL} = 3V; V_{OH} = 3V; V_T = 2V; V_{dd} = 4V$$

$$C = 50pF; R_0 = 200\Omega; R_I = 1M\Omega$$

Con $V(0) = 0V; \quad V(\infty) = 3V$:

$$\begin{aligned} V(t) = V_T &= (V(\infty) - V(0))(1 - e^{-\frac{t}{RC}}) + V(0) \\ 2 &= 3 \cdot (1 - e^{-\frac{t}{RC}}) \\ -\frac{t}{RC} &= -\log(3) \\ t &= RC\log(3) = \boxed{11ns} \end{aligned}$$

1.4 Consumo di Potenza

Nella storia sono due i periodi in cui il consumo di potenza fu un problema. Agli albori i circuiti bipolari che consumavano corrente anche in condizioni statiche; all'inzio degli anni 2000, dove le correnti di perdita diventorono troppo elevate (milioni di transistor).

La potenza viene usata in parte per il funzionamento interno del modulo, in parte viene buttata via. La potenza viene fornita dalla tensione di alimentazione, mentre la variazione della potenza è detta dalla variazione di corrente assorbita. Le forti correnti richiedono conduttori grandi (altrimenti si brucerebbero). Inoltre vengono generati dei disturbi elettromagnetici che potrebber causare il malfunzionamento dei dispositivi vicini. Il calore generato deve essere dissipato.

Ci sono due tipi di potenze consumate:

Statica corrente e tensioni statiche: *potenza inutile* che non si può ridurre a 0 (in un circuito deriva da un transistor in interdizione). Si deve cercare di ridurre questa componente il più possibile. Se un transistor consuma meno potenza statica allora va anche più lento e vicerversa. La potenza statica dipende dalla corrente sotto soglia I_{off} e dall'alimentazione.

$$P_S = I_{off} \cdot V_{AL} \tag{1}$$

Dinamica *potenza utile*, dipende dalla tecnologia, dall'alimentazione e soprattutto dal carico capacitivo. Quando carichiamo e scarichiamo il condenstore F-volte al secondo si ottiene al formula della potenza dissipata.

$$P_D = F \cdot C \cdot V^2 \tag{2}$$

Per una data tecnologia il prodotto $P_D \cdot T_P$ (dove T_P è il ritardo), è costante.

Example 1.3

Calcolare il consumo di potenza *statica* e *dinamica* di un circuito $cMOS$ con:

- Alimentazione $2V$
- In media 50‘000‘000 transistor spenti
- 100‘000‘000 ingressi con $C = 1fF$

$\boxed{C1}$ Frequenza di commutazione media $F_m = 200MHz$ e $I_{off} = 1nA$ per dispositivo:

$$P_S = 1nA \cdot 2V \cdot (5 \cdot 10^7) = 0.1W$$

$$P_D = 1fF \cdot 200MHz \cdot (2V)^2 \cdot 10^8 = 80W$$

$\boxed{C2}$ Frequenza di commutazione media $F_m = 2GHz$ e $I_{off} = 100nA$ per dispositivo:

$$P_S = 100nA \cdot 2V \cdot (5 \cdot 10^7) = 10W$$

$$P_D = 1fF \cdot 2GHz \cdot (2V)^2 \cdot 10^8 = 800W$$

1.5 Domande di Fine Capitolo

1. *Perché i sistemi elettronici sono sempre più ”digitali“?*
   Perché presentano un intolleranza ai rumori maggiore, possibilità di ricostruire l’informazione.

2. *Elencare i parametri statici di porte logiche TP, 3S.*
   TP: resistenze equivalenti R_{OH} e R_{OL}; tensioni minime di soglia V_{OL} e V_{IL}, tensioni massime di soglia V_{OH} e V_{IH}; corrente erogata I_O nei bipolari; correnti di perdita.
   3S: metre è abilito ha le stesse caratteristiche del TP, mentre non è abilitato ha: terzo stato a tensione Z; corrente di perdita I_{OZ};

3. *Quali sono le condizioni di compatibilità tra porte logiche?*
   $V_{OL} < V_{IL}$; $V_{OH} > V_{IH}$; stato alto $|I_O| < |I_{OH}|$; stato basso $|I_O| < |I_{OL}|$;

4. *Che cosa è il margine di rumore?*
   È un intervallo tra (V_{IH}, V_{OH}) o (V_{OL}, V_{IL}), garantisce che il rumure non faccia andare l’uscita all’interno della zone non definita.

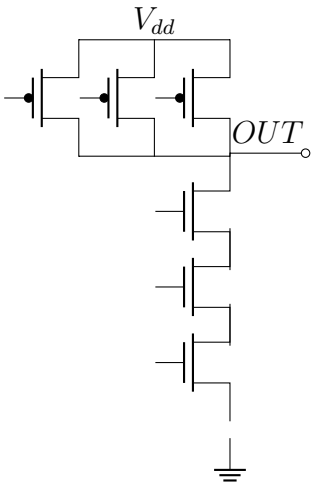
5. *Come si verifica il corretto interfacciamento in un circuito logico?*
   Analizzando la corrente erogata e controllando che la tensione risultante non vada al di fuori delle soglie.

6. *Elencare i parametri dinamici di porte logiche TP, 3S.*
   Tempi di ritardo di propagazione e tempi di transizione, entrambi da basso ad alto e da alto a basso.

7. *Cosa è e da cosa dipende il Fan Out?*
   Il FANOUT è il massimo numero di porte collegabili ad un'uscita, dipende dalle capacità di entrata delle porte.

8. *Quali sono i vantaggi dei segnali differenziali?*
   Hanno una maggiore resistenza al rumore.

9. *Quanti MOS occorrono per un NAND a 3 ingressi CMOS?*
   Occorono 6 MOS.



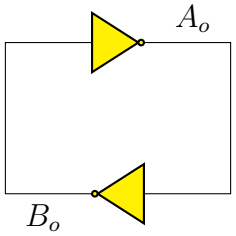
10. *Da cosa dipende la potenza dissipata in un circuito logico?*
    La potenza statica dipende dalla corrente di uscita (compresa della corrente di perdita). La potenza dinamica dipende dalla tecnologia e dalla capacità in entrata.

11. *Su quali elementi si può intervenire per ridurre il consumo dinamico di un circuito logico?*
    Si può intervenire sulla frequenza e sulla tensione di alimentazione.

12. *Quanto varia la potenza dinamica dissipata in una logica CMOS dimezzando la tensione di alimentazione? Che cosa si "perde" facendolo?*
    La potenza dinamcia diminuisce di quattro volte ma il ritardo aumento di quattro volte. Il motivo è che il prodotto tra potenza e ritardo è costante per una data tecnologia.

2 Circuiti Bistabili

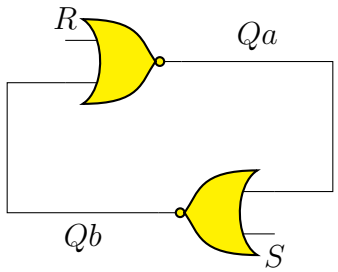
Un circuito bistabile è un circuito con almeno due stati stabili, grazie ad essi si possono realizzare dei circuiti per immagazzinare dei dati, questi avranno le loro caratteristiche statiche e dinamcihe.
Per immagazzinare un informazione è necessario realizzare un circuito bistabile. Ad esempio due inverter collagati ad anello in cui i due stati stabili sono: $S0$ ($A_o = H, B_o = L$); $S1$ ($A_o = L, B_o = H$).



È possibile dimostrare che in tutti i sistemi bistabili esiste un terzo punto in cui il circuito è stabile, questo stato viene mantenuto per un tempo casuale, questo punto si trova vicino alla soglia di commutazione. Il problema di questa condizione meta-stabile è che non è noto il tempo nella quale rimarrà in questa condizione, è molto probile che ci rimanga per poco tempo ma, c'è una probabilità che rimanga in quello stato per un tempo molto lungo. Il compito di un progettista è anche quello di non far andare un circuito sequenziale nella zone di meta-stabilità.

Un modo molto semplice di portare il dispositivo in uno stato desiderato è quello di aggiungere delle entrate. Un esempio sono i **flip-flop set reset** (FF-SR) creato con delle porte NOR, in cui l'entrata di reset forza l'uscita a L e l'entrata forza l'uscita ad H. Un FF-SR ricorda il passato: l'entrata foraza l;uscita ad un valore stabilito per un intervallo di tempo indefinito. I possibili stati sono:

- $S = 1, R = 0 \Rightarrow Qa = 1, Qb = 0$;
- $S = 0, R = 1 \Rightarrow Qa = 0, Qb = 1$;
- $S = 0, R = 0 \Rightarrow$ stato precedente;
- $S = 1, R = 1 \Rightarrow$ le due condizioni non sono complementari, *codizione proibita*, può portare a condizioni di meta-stabilità;



È possibile realizzare un flip-flop anche con delle porte NAND, in cui le entrate sono invertite (lo stato di 11 non è più proibito, lo è 00).

Quando il dispositivo viene acceso l'uscita non è nota a priori, potrà anche essere in codizione meta-stabile. Nel momento in cui porta si porta RESET a 1 (con porte di tipo NAND), Q diventerà 1 e se SET sarà 1, Q diventerà 0. L'impulso di SET o RESET deve essere abbastanza largo, il che garantsice che il circuito passi ad uno stato stabile.

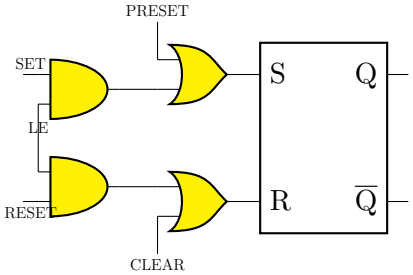
Quale può essere un'utilizzo?
In una tastiera: quando si preme un pulsante le vibrazione provocate dalle discesa potrebbero far cambiare l'uscita moltissime volte, grazie a questo dispositivo si scartano i segnali forniti da rimbalzi successivi, infatti i SET successivi non avranno alcun effeto sull'uscita del FF, si crea un dispositivo anti-rimbalzo.

Esistono due tipi di circuiti sequenziali, ed in particolare di tipi di FF:

- I circuiti asincroni: possono cambiare l'uscita in qualunque momento.
- I circuiti sincroni: sono pilotati da un ingresso di clock (CLK), che non pilota l'uscita ma fornisce un metodo di sincronizzazione mentre, le altre entrate (che portano i dati) pilotano le uscite. Usando l'ingresso di clock, si possono studiare i comportamenti tramite le macchine a stato finito o l'algebra booleana. Quindi le funzioni vengono scomposte in *temporizzazioni* e *valori*. Nei circuiti sincroni la terminologia è molto stadardizzata.

Nei circuiti sincroni la terminologia è molto stadardizzata, mentre nel è lo stesso caso per quelli asincroni.

Si possono allora creare dei FF in cui i segnali vengono salvati solo mentre il segnale di clock (o Latch Enable) è attivo.



Definition 2.1 – FF Master-Slave

Un FF di tipo Master-Slave è fatto dalla cascata di due FF latch con abilitazione complementare, ovvero con CK e $\overline{CK}$.

- $CK = 0$: abilita il prima latch e blocca il secondo → Master trasparente, Slave in memoria;
- $CK = 1$: abilita il secondo lathc e blocca il primo → Master in memoria, Slave trasparente;

Dunque il dato viene memorizzato solo dalla trasizione del CK da 0 a 1, quindi sul fronte di salita. Se neghiamo il CK l'informazione verrà copita sul fronte di siscesa

Anche i FF-MS hanno un valore di PRESET e CLEAR che saranno collegati ai due latch.

Example 2.1 – Latch D

Un Latch-D è un flip-flop con un singlo ingresso D, di tipo FF-MS. D viene mandato nel SET, mentre $\overline{D}$ veine mandato nel RESET. Il Latch Enable (LE) permette di settare l'uscista al valore di D durante la sua fase di trasparenza. Il Latch-D possiede anche dei segnali di PRESET e CLEAR.

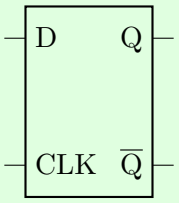


Figure 1: Schema di un Latch-D

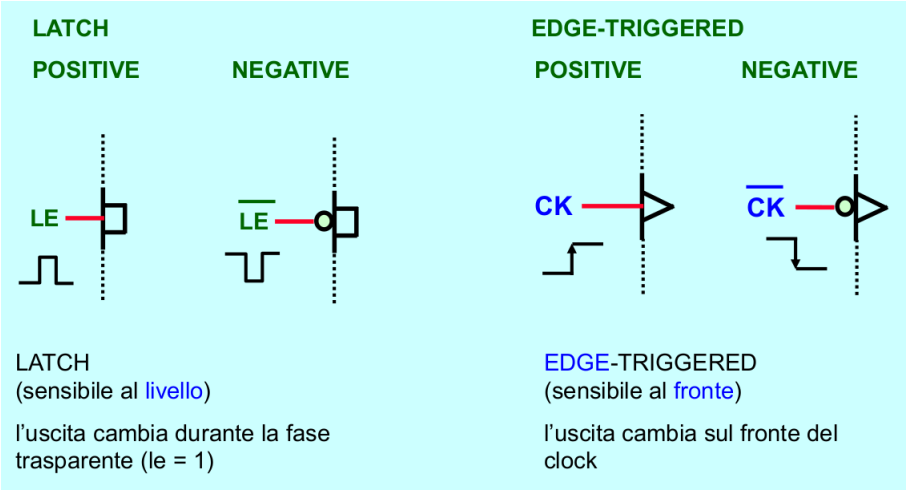
Per tutti i circuiti sequenziali, vista la non conoscenza dello stato iniziale, vengono usati dei segnali di PRESET e di CLEAR. La motivazione per usare questi segnali è che a ridosso del latch ci sono delgi altri latch ed altri circuiti combinatori, dunque l'interdizione dovrebbe partire dall'inizio del circuito, facendo un grande sforzo. Si usa CLEAR per impostare lo stato dei FF a 0 (per tutti i circuiti). Una volta settate le uscite a 0 si possono mandare le prime istruzioni. Lo stato di PRESET viene utilizzato anche quando il LE non è a 1. I Latch-D sono molto difficili da controllare, dunque sono raramente utilizzati.

Si deve garantire una finistra temporale in cui il dato deve essere stabile durante la transizione del clock, perché un FF-MS commuta la sua uscita solo ful fronte di salita del clock.. Esistono anche dei FFMS che lavorano su due fronti di salita ma, non vengono usati all'interno di circuiti integrati, vengono utilizzati nella memorie in cui la frequenza di clock diventa limitante per il tipo di circuito, vengono detti FF-D

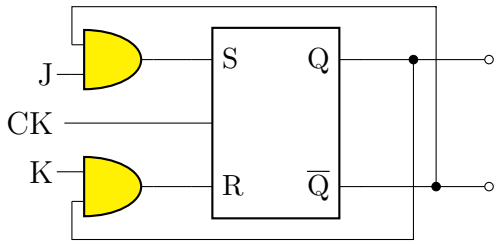
Dual Edge (FF-DDE).

Si una terminologia:

- Latch:
  - memorizza D (LE $1 \rightarrow 0$);
  - stato di trasparenza per $LE = 1$;
  - stato di memoria per $LE = 0$;
- FF D (MS):
  - memorizza D (LE $0 \rightarrow 1$);
  - L'uscita commuta solo sulle transizioni del clock;



Il FF-JK ci permette di avere delle funzionalità leggermente più complesse: con le due entrate a 0 il valore di Q viene mantenuto lo stato precedente, con J a 1 Q va a 1, con K a 1 Q va a 0, con J e K entrambi a 1 l'uscita assume $Q*$ (lo stato al clock predente ma negato).



2.1 Ritardi nelle Porte Loiche Sequenziali

FF-SR Le porte NOR introducono dei ritardi, per far si che i cambiamenti abbiano effetto, i comandi di SET e RESET devono avere una durata minima (di solito il doppio della durata massima del tempo di propagazione nelle porte). Nello porte si ha un ritardo di propagaizone da cui il valore di Q deve arrivare a $Q*$ e viceversa. Se l'intervallo è troppo lento il valore di Q rimane invariato, mentre nel caso peggiore rimane in posizione meta-stabile, che va evitato.

FF-D Ci sono tre vincoli principali:

- Tempi di S e R interni appropriati (come sopra);
- Margine di tempo in cui D deve rimanere stabile nell'intorno del fronte di salta del clock, deve rispettare i tempi di set-up t_{su} e di hold t_h, se l'entrata non è stabile, nel caso peggiore l'uscita oscilla fino al prossimo colpo di clock.
- Durata minima del clock, non deve essere più veloce del tempo di commutazione;

Il ritardo dato dalla della meta-stabilità è detto **tempo di risoluzione**, tra tutti i ritardi non è conosciuto a priori, infatti è preferibile evitarlo.

Cosa accade con più FF collegati tra di loro (di solito non accade)?
Si deve rispettare il tempo set-up, il tempo della transizione allo stato alto del segnale, al tempo di hold piu il tempo di discesa.

2.2 Domande di Fine Capitolo

1. *Un latch nella condizione di trasparenza:*
   Mantiene lo stato precedente, anche se l'ingresso varia.
2. *Quante porte NAND a 2 ingressi servono per realizzare un D-FF (master-slave)?*
   6: 4 per i FF, 2 per gli inverter.
3. *Il D-FF Master-Slave ha una condizione di trasparenza?* Si, quando il clock è a 1, il master si travo in trasparenza, mentre quando il clock è a 1 lo slave è in trasparenza.
4. *In un D-latch se D=0, EN=0, PRESET=1, CLEAR=0, Q = ?* $Q = 1$
5. *Tracciare Q per un D-FF con D collegato a Q* (per 4 colpi di CK) se Q inizialmente vale 0.*

CK=0: D = 1; Q = 0;

CK=1: D = 0; Q = 1;

CK=2: D = 1; Q = 0;

CK=3: D = 0; Q = 1;

CK=4: D = 1; Q = 0;

6. *Che cosa sono i tempi di setup e di hold?*
   Il tempo di setup è l'intervallo di tempo mimino prima del fronte di salita del clock in cui il segnale in ingresso deve rimanere stabile, mentre il tempo di hold è l'intervallo minimo dopo il fronte di salita in cui l'entrata deve rimanere stabile.

7. *Descrivere il comportamento di un FF in condizione metastabile.*
   Un FF in condizione metastabile ha delle oscillazioni non definite fino a quando un nuovo colpo di clock ristabilizza il sistema, se non si ha un sistema come un MS allora il FF può rimanre vicino a tale condizione per un tempo indefinito.

3 Circuiti Sequenziali

Si è visto che è possibile rappresetare le informazioni in modo parallelo o seriale. Si possono usare i FF per rappresentare i dati su entrbe le codifiche o per passaere da una all'altra.

A parità di frequenza di clock:

- Connessione Seriale: piu lenta, piu economica, minor consumo, piu usata su distanze lunghe (SATA, Ethernet)
- Connessione Parallela: piu veloce, consumo maggiore di potenza, piu usate su distanze corte (all'interno dei circuiti integrati)

Nel momento in cui i FF sono colleagati in serie i dati vengono trasportati in modo sequenziale. Il segnale di ingresso viene ritardato di n-volte tante quante sono i FF. Si puo usare per fare dei filtri digitali (visto a teoria dei segnali). Se si prendono le uscite per ogni FF se possono convertire le informazioni seriali in parallele. È possibile usare una variante del circuito per realizzare la trasformazione opposta. Nei CI sono implemente delle funzioni che permettono di far uscire l'informazione in modo seriale,

3.1 Tipoligie di Registri

- **PIPO** (Parallel Input Parallel Output):
  Possiede N bit, un clock o un latch, comandi vari;
- **SISO** (Serial Input Serial Output):
  Una cascata di FF-D con clock e RESET in comune. Viene anche detto reistro a scorrimento (Shift-Register);
- **SIPO** (Serial Input Parallel Output):
  Come un SISO ma può convertier un insieme di dati seriali in parallelo;
- **PISO** (Parallel Input Serial Output):
  Permette di carciare in parallelo i singoli FF;

Si può creare un circuito che permette di usare le combinzioni dei tipi conessione sia in input che in output.

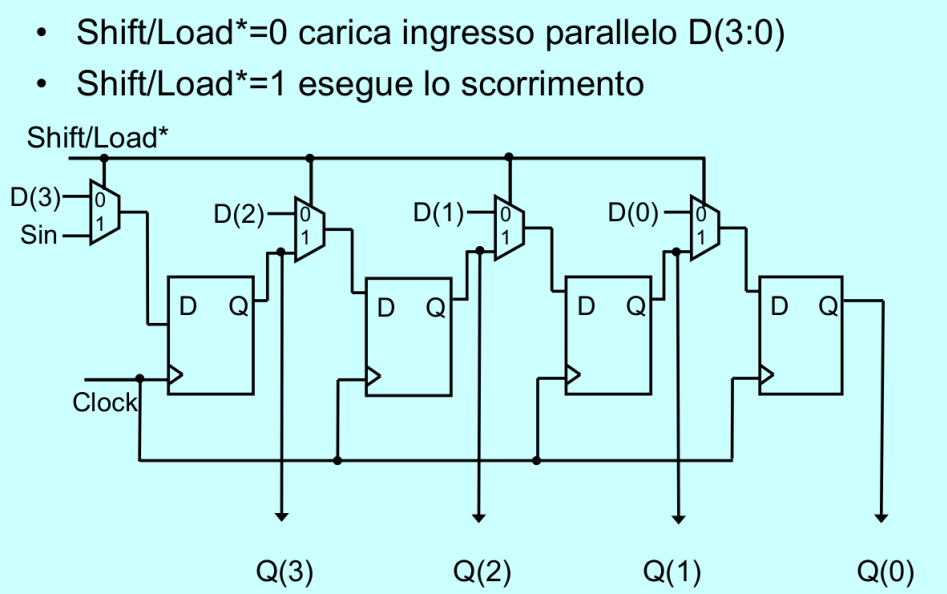


Figure 2: Shift-Register Completo

Example 3.1 – Contatore

Il contatore è un contatore in modulo N. Se, per esempio, si vogliono indirizzare le istruzioni, N sarà pari al log_2(numero di indirizzi). Il contatore più semplice da realizzare è in modulo 4, si mette l'uscita di un FF all'entrata del clock del prossimo FF, entrambi i FF hanno l'uscita $Q*$ collegata alla propria D. Ad ogni colpo di colck $Q1$ e $Q2$ rappresentano il numero del contatore. Il motivo per cui questo è un contatore asincorono è dato dal fatto che l'entrata CLK del secondo FF non è pilota dal clock della CPU ma dall'uscita del primo FF, dunque il ritardo cresce in modo scalare.

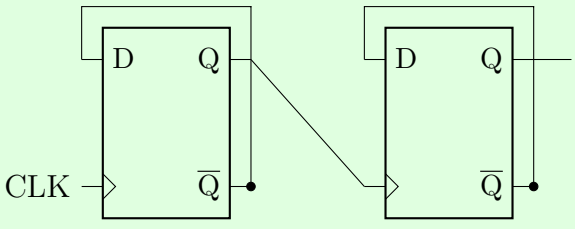


Figure 3: Contatore in *mod* 4

Si possono usare i FF-JK con gli ingressi collegati ad 1 (ad ogni colpo di clock la stato commuta) e CLK negato, si crea così un contatore che conta oppure no a differenze del segnale in ingresso (contatore con abilitazione).

Si può realizzare un contatore sicrono usando i FF-JK in combinzione con della logica combinatoria. Le porte AND vanno a verificare quando tutti gli stati precendi sono ad 1, dunque il prossimo stadio commuta e quelli precedenti vanno a 0.

Per realizzare un contatore non in modulo 2^N ma, usando un numero qualsiasi si usano le entrate si RESET dei FF. Viene utilizzata una logica combinatoria collegata a tutti i RESET, quando si raggiunge il numero prestabilito tutti i FF vangono porati a 0 e si riparte a contare.

3.2 Macchina a Stati Finiti

Per progettare funzionalità più complicate diventa molto difficile usare solo porte logiche, è per questo motivo che viene utilizzato il modello delle **macchine a stati finiti** (FSM). Le FSM sono fatte da stati iniziali e da stati, ogni stato può effettuare una transizione ad un altro se sono rispettate delle condizioni. Le uscite verso l'esterno possono essere associate a degli stati o a delle transizioni. Per avere una FSM determinitstica è necessario che le condizioni sulle transizioni siano mutuamente esclusive. Le FSM vengono rappresente come dei grafi in cui ogni vertice rappresenta uno stato ed ogni arco rappresenta la condizione di transizione.

In generale c'è un interconnessione di porte logiche, che calcolano lo stato futuro a partire dallo stato presente (i valori dei FF) e delle entrate, collegate poi ad un rete combinatoria che produce le uscite. Se ci fosse una connessione tra entrate ed uscita la FSM viene detta **macchina di Mealy**, altrimenti viene detta **macchina di Moore**.

Si passa da un modello a grafo ad implementazione con porte logiche codificando ogni stato con un simbolo unico, usando ad esempio una codifica binaria.

3.3 Analisi dei Ritardi

Nell'intorno del fronte di salita $(t_r - t_{su}, t_r + t_h)$, si è visto come il segnale in ingresso necessita di rimanere stabile. Nel momento in cui viene introdotta una logica si può analizzare questo ritardo. Ci interessano due ritardi: il minimo ed il massimo, cammino più corto tra l'uscita di un FF e l'entrata del successivo per tutti i FF del circuito.

Il massimo è correlato alla soddisfazione tempi di setup: ogni volta che c'è una trasizione sul clock le uscite dei FF cambieranno, ci sarà un ritardo da quando si vede un cambiamento nell'uscita (t), ci sarà un ritardo causato delle porte logiche (t_{LCMAX}). Il periodo di clock non può essere minore delle somma di questi ritardi.

$$T_{CLK} > t + t_{LCMAX} + t_{su}$$

Per soddifare il tempo massimo è necessario modificare il periodo di clock (molto facile).

Il minimo è correlato al tempo di hold (non si deve arrivare al FF successivo troppo velocenmente): le porte logiche hanno bisogno di un tempo minimo per cui ricevano l'uscita delle porte loghiche (t_{LCMIN}).

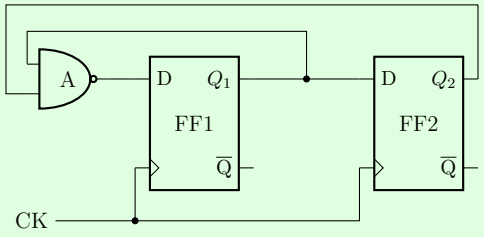
$$t_h > t + t_{LCMIN}$$

In generale esistono più metodi di realizzare la stessa fuznione booleana con dei circuiti combinatori e dei FF. La differenza tra le diverse implementazioni si basa sui ritardi massimi e l'area di silicio occupata, una bouana oppressimazione dell'area occupata è data dalla somma dei transistor presenti nel circuitito.

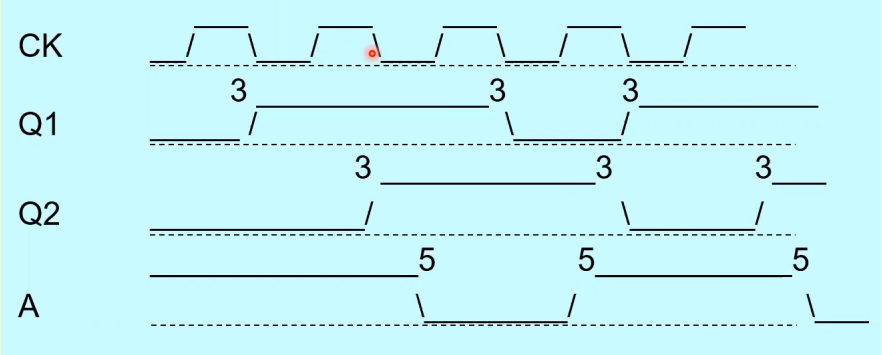
Example 3.2

Analisi di un circuito sequenziale. Tracciare il la sequenza di segnali, con stato iniziale $Q_i = 0$ e calcolare la F_{max} del clock, considerando i dati:

$$T_{NAND} = 2ns; \quad T_{CKQ} = 3ns; \quad T_{SU} = 1ns; \quad T_H = 1ns;$$



Segnali con i rispettivi ritaridi in ns.



Il periodo minimo che il clock deve rispettare è: $T_{min} > T_{NAND} + T_{CKQ} + T_{SU} = 6ns$

$$\rightarrow F_{max} = \frac{1}{2ns + 3ns + 1ns} = 166MHz$$

Il cammino più breve da un FF ad un altro va da Q_2 a D_2, infatti il ritardo minimo della logica combinatoria è zero in questo caso. Il vincolo sul tempo di hold è rispettato.

$$T_H < T_{CKQ} = 2ns$$

3.4 Domande di Fine Capitolo

1. *Tracciare lo schema di un divisore asincrono modulo 32 realizzato con D-FF.*

2. *Quanti FF occorrono per realizzare un contatore modulo 17?*
   Servon 5 FF.

3. *Tracciare lo schema logico (porte e FF) di un convertitore da flusso di bit seriale a rappresentazione parallela.*

4. *Una FSM ha 9 stati. Quanti FF occorrono per realizzarla?*
   Servono 4 FF.

5. *Si deve realizzare un contatore modulo 4 (FSM controllo della lavatrice). Quale è la massima frequenza operativa se: i FF hanno ritardo di 8 ns; le porte NAND, NOR, NOT e EXOR hanno ritardo di 11 ns; il tempo di setup dei FF è di 9 ns ?*
   Per realizzare una AND serve una porta NAND ed una NOT (per il contatore serve una porta XOR ed una AND in serie con una XOR). Il tempo max dello lagica sarà $T_{CL} = 11ns \cdot 3 = 33ns$, il tempo minino di clock sarà: $T_{CK} = 33ns + 8ns + 9ns = 50ns \quad \rightarrow 20MHz$.

6. *Realizzare un SIPO a 8 bit utilizzando due registri SIPO da 4 bit.*
   Si collega il Serial Out del primo al Serial In del secondo.

7. *Quali parametri determinano la massima frequenza di clock per una FSM?*
   Tempo di ritardo dei FF; massimo tempo di ritardo della logica combinatoria; tempo di setup;

4 Comparatori

Un comparatore ha il seguente funzionamento: quando l'ingresso è sopra una certa soglia manda l'uscita in H, quando l'ingresso è sotto la soglia manda l'uscita in L. Un comparatore viene costrutito usando gli amplificatori operazionali (OP). Per realizzare un comparatore invertente basta invertiere i morsetti. Quando al segnale di entrata viene sommato del rumore si possono creare delle trasiszioni molto veloci, date dal fatto che il segnale oscilla motlte volte attorno alla soglia, il numero alto di transizioni nell'intorno della soglia causano un maggior spreco di potenza. Per risolver questo problema si usano due soglie diverse, una per lo stato da L a H ed una per lo stato da H a L, in questo modo si eliminano le transizioni causate dal rumore, a patto che la soglia del rumore non sia oltrepassata. Un comparatore con queste soglie viene detto **comparatore con isteresi**, in cui l'intervallo tra le due soglie fa da margine per il rumore.

Il comparatore con isteresi ha due caratteristiche: una quando passa da H a L ed una quando passa da L a H. I comparatori senza isteresi vengono realizzati con OP ad anello aperto, mentre i comparatori con isteresi vengono creati con OP ad anello chiuso con retroazione postiva.

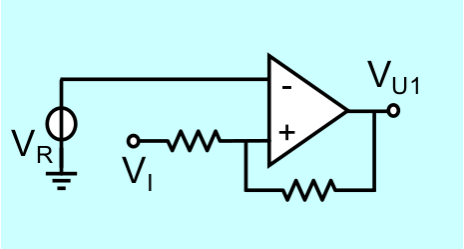


Figure 4: Comparatore con Isteresi

Nella pratica si compra direttamente il CI con il comparatore e le resistenze che servono a noi. Un comparatore con isteresi è anche detto **Trigger di Schmitt**. Il valori dei threshold dipendono dall'alimentazione e da altri fattori, come tolleranze di fabbricaione.

Utitilizzando i comparatori con isteresi si possono realizzare delle forme d'onda quadra, ciò è possibile avendo un circuito RC pilotato ad onde quadre, dove la tensione in uscita al condesatore sarà una serie di onde, date dalla carica e dalla scarica. Il condensatore messo in retroazione con un comparatore di Shmitt da proprio questo effetto, infatti quando viene oltrepassata la soglia del comparatore il segnale in retroazione sarà invertito e dunque porterà il condensatore a scaricarsi provocando un'altra transizione della soglia.

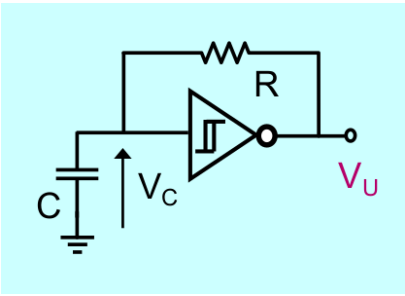


Figure 5: Generatore ad Onda Quadra

4.1 Limiti Operativi

Resistenza di reazione:

- Max: in R deve circolare una corrente maggiore della corrente di ingresso del comparatore;

- Min: R è vista come carico all'uscita, dunque valori troppo bassi limitano la dinamica di uscita;

Il **condensatore** non ha limiti su quanto può essere grande, mentre il valore più piccolo è dato dalle capcità parassita di ingresso.

La **frequenza** minima è determinata dalla capacità massima e dalla resistenza massima (C_{max}, R_{max}), mentre la frequenza massima è determinata dallo slew rate del comparatore.

4.2 Oscillatori ad Anello

Un circuito che contiene un numero dispari di inverter diventa un oscillatore, il circuito diventa instabile, dove il periodo dell'oscillazione è dato dalla somma delle propagazioni in salita ed in discesa degli inverter, tante volte quante sono gli inverter. Si usano gli inverter per creare degli oscillatori perchè le resistenza e le capacità fisse dei circuiti con i comparatori sono più costose e più imprecise.

Per riuscire a migliore la precisione delgi inverter bisongna inserire nella rete di reazione un oggetto che migliora la precisione.

Si utilizza il quarzo (materiale piezoelettrico), che risponde ad una tensione con delle oscillazioni molto precise, ha come modello elttrico una combinazione accordata di resistenze e capacità. L'utilizzo di cristalli al quarzo è il metodo più economico per realizzare degli oscillatori. La resistenza di reazione serve per permettere al quarzo di operare in condizioni favorevoli.

Per controllare la frequenza di clock bisongna essere precisi ma flessibili: nel caso in cui il carico computazionale varii nel tempo (eg sistema di un celluare, dove il carico viene ragolato a differenza delle operazioni che devono eseguire). In questo caso si usanao gli **anelli ad aggancio di fase** (PLL: Phase-Locked Loop). Usando le reazione per controllare la frequnza, un oscillatore controllato in tensione può anche non essere preciso, questa imprecisione verrà attenuata proprio dalla rezione.

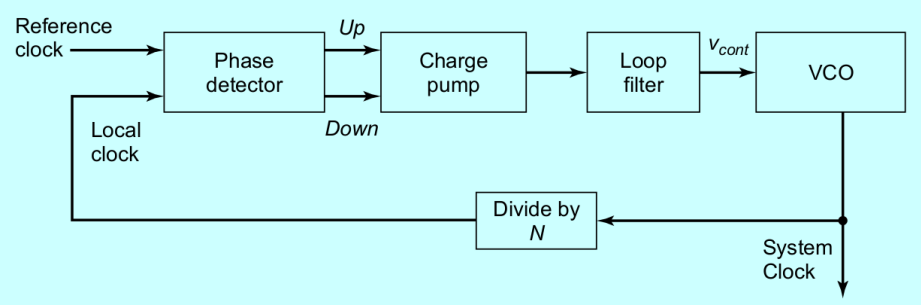


Figure 6: Phase-Locked Loop

- **Rivelatore di fase:** indica se la frequnza deve salire o scendere. Cicruito logico che verifica quale dei due segnali arriva prima, aumenta o diminuisce in reazione la frequenza dell'oscillatore, mantenendo allineate le due fasi.

- **Charge Pump:** transforma il comando "vai su" o "vai giu" del rilavatore di fase in tesioni, regolondo le tesione di alimentazione della catena degli inverter in modo continuo.
- **VCO (Voltage-Controlled Oscillator:** Il charge pump da piccoli impulsi in modo da aumentare o diminuire la tensione di alimentazione, infatti un inverter ha una frequenza minore al diminuire della tensione di alimentazine e vicerversa. L'anello di reazione aggancia il VCO all'oscillatore di riferimento, con alta precisione e con una frequenza minore. La tesione di uscita andrà poi andra ai FF del circuito.

Per ridurre la potenza si puo ridurre la tensione di alimentazione del circuito o si può abbassare la frequenza.

4.3 Domande di Fine Capitolo

1. *Quali parametri descrivono un comparatore di soglia?*
   Se ha isterisi oppure no, valori della tensione di uscita e valori delle tensioni di soglia

2. *Per quale motivo i comparatori hanno isteresi ?*
   Per ridurre l'influenza del rumore. Causa molte commutazioni in uscita con maggiore spreco di potenza.

3. *Tracciare lo schema di un generatore di onda quadra, e indicare cosa determina le frequenze massima e minima ottenibili*
   Schema in (Figure 5). La frequenza minima è determinata dai valori massimi di resistenza e capacita, mentre quella massima da quanta corrente si è in grado di alimentare la capacità e da quanto grande 'e la capacità (slew rate).

4. *In un generatore di onda quadra con integratore e trigger, come cambia la frequenza raddoppiando la capacità dell'integratore?*
   La frequenza raddoopia.

5. *Perché non si usano gli oscillatori ad anello per generare un clock stabile?*
   Perche i parametri di questi oscillatori cambiano con la temperatura, è difficile travare errori bassi nelle capacità e nelle resistenza. Non si puo vendere un componente con un incertezza dell 1% sulla frequenza.

6. *Quali sono i vantaggi degli oscillatori a quarzo?*
   Gli oscillatori al quarzo sono molto precisi, possono oscillare a frequenze molto alte, sono molto precisi, sono molto economici da realizzare.

7. *Perché si usa un PLL per generare frequenze di clock dell'ordine dei GHz?*
   Usando i PLL si puo ridurre la frequenza quando non è necessario, creare un

oscillatore al quarzo che gira sui GHz è molto costoso, inoltre la precisione deve essre molto alta.

5 Logiche Programmabili

I primi CI sono usciti negli anni '60, creati interamente sul silicio con BJT. Nel '71 si passa ad usare i MOS, con un consumo di postenza molto minore $\implies$ aumento del numero di transistor sul CI. La **Legge di Moore** è una legge economica che stabilisce una relazione econuumica tra i produttori dei CI ed i consumatori, la legge di Moore dice che: il numero di trasistor raddoppia sulla superficie del silicio ogni 2 anni. Questo è possibile perchè la dimensione dei transistor si riduce, ma intorno al 2010, arrivati intorno ai $30nm$ di lunghezza di un singolo transistor, la velocità è dimuita. Per drogare il canale vengono usati pochi atomi, addirittura 2-3, dunque le differernze tra transistor possono addirittura variare del 50% o 30%. Per questo motivo si è aumento il numero di core, con un conseguente aumento della quantita di silicio, con un aumento dei prezzi. Il motivo per cui il costo dei trasistor cresce è dato al fatto che: le maschere per creare il CI diventano sempre più costose, per generare, i CI servono sempre più maschere attroverso l'uso dell'interferenza dei raggi.

Il costo di un CI è dato da: il costo delle maschere ed il costo di progettazione che è un costo *una tantum* (NRE cost: Non-Recurring Engineer cost), che diviso per il numero di CI si somma al costo unitario di produzione.

Ci sono essenzialmente 4 tipi di CI:

- Comodity (Commercial Off The Shlef): processori, memorie, (usati in ampiamente, con costi solo per il silicio);
- Custom: circuiti specializzati (come circuititi di potenza, analogici, ...);
- Semicustom: ad esempio la Apple che vende il suo prodotto con il suo HW, ha bisogno di un gruppo che progetta un codice che giri su l'HW ed un gruppo che progetti l'HW, si hanno elevati costi di progettazione, ma il prodotto è molto performante;
- Circuiti Logici Programmabili: ci sta un codice che pilota i transistor, potendo cambiare il funzionamento delle porte logiche a piacimento.

Quando un'azienda non può permettersi di sviluppppare internamente un prodotto compra un'azienda che lo puo costruire, o ne compra i diritti di utilizzo. Comprare componenti o un'azienda intera è necessario all'interno dell'industria elettronica, dato degli enormi costi dello sviluppo, ciò permette di creare sistemi molto complessi. Il problema risiede nel mettere insieme tutto ciò, per capirne le specifice e determinare come tutti questi componenti vanno ad integrarsi tra di loro.

La segmentazione dell'industria è la seguente:

- **Progetto di sistemi:** (Marelli) miglia di aziende fanno progetti di sistemi, grandi o piccoli. A volte può includere la produzione;
- **Progetto di circuititi integrati:** (Apple, ARM) aziende specializzate per gli alti costi degli NRE e delle maschere;

- **Fabbricazione di circuiti integrati:** (Intel, Samsung) molto specializzata per gli altissimi costi degli impianti di produzione.

Nel caso dei circuti full custom si lavora a tutti i passi del progetto manualmente. Si ha la totale flessibilità ma i costi sono altissimi, si puo fare solo per circuititi con circa 200 transistor, le memorie sono dei circuititi full custum;

Circuiti Semicustum: sono progettati a partire da porte logiche, chi progetta questi circuti ha a disposizione una libreria di porte che si possono utilizzare nel circuito, non si è necessario calcolare le posizioni all'interno del silicio.

Circuiti Programmabili: si decide tramite software come le porte si dovranno comportare. I CI Programmabili sono a metà tra spese e tempi di produzione.

5.1 Componenti Programmabili

L'asppetto fondamentalo è l'elevato numero di clienti per tali circuiti, dunque il prezzo di progetto viene spalmato sulle grandi quantità vendute. Il CI deve essere molto flessibile per includere una grande quantità di casi. I CI possono essere programmati permanentemente o riprogrammabili dinamicamente (tipicamente si usa in fase di prototipazione). Un esempio di CI riprogrammabile on the fly, sono le base station delle reti cellulari che hanno visto upgrade di protocolli e di getsione dei dati.

La configurazione è memorizzata in memoria. Un CI programmabile è fatto da:

- Memoria
- Fuznione logiche relizzate con tabelle di lookup o

Programmabel Logic Array (PLA): fatte da matrici di invertitori porte AND e OR. I fili vengono bruciati per creare i ponti di collegamento che andranno a creare le funzioni logiche.

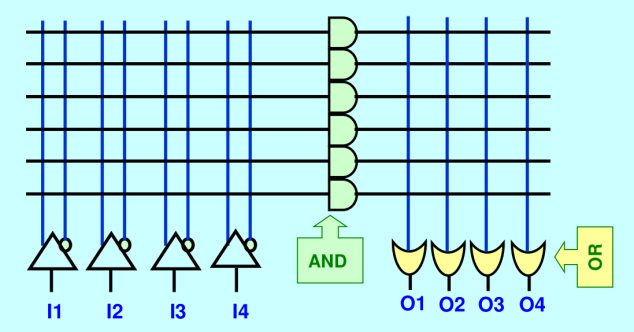


Figure 7: Logiaca PLA

Field Programmabel Gate Array (FPGA): fatta da celle logiche che realizzano qualsiasi funzione logica fino ad un certo numero di ingressi in cui sono presenti

anche dei registri. Le connessioni sono anche programmabili e le celle di ingresso ed uscita consentono di collegare il circuito con un circuito stampato. Ogni cella ha una memoria in cui si mette le fuzione logica, si possono realizzare quasi tutte le combinazioni di fuznioni con un N ingressi. La memoria può esser caricata su una RAM, dei registri (volatile), o memorie non volatili in cui si usano fusibili, queste ultime si usa perchè sono immuni al fenomeno di distruzione causto dai raggi cosmici.

5.2 Domande di Fine Capitolo

1. *Descrivere da cosa dipende il costo di un circuito integrato.*
   Dal costo di produzione e dal costo di progettazione.

2. *Perché il progetto deve essere rapido?*
   Perchè se il progetto viene distribuito troppo tardi la finestra di mercato ha un tempo finito, dunque il coomponente fatturerà fintanto che la finestra rimane aperta. Con un distribuizione ritardata il prezzo dei costi di produzione non si riesce a colmare.

3. *Cosa significa "COTS"?*
   Vuol dire Commercial Off-The-Shelf. Sono dei componenti dedicati al general purpose.

4. *Spiegare le differenze tra full custom, semicustom, logiche programmabili.*
   Le full custom sono circuiti progettati al livello delle porte logiche, hanno alti costi progettuali. Le semicustom usano moduli già utilizzati ed assemblati instieme, con costi di produzione medi. Le logiche programmabili

5. *Quali sono i vantaggi delle logiche programmabili?*
   Quando si hanno pruduzioni relativamente piccole ma non si vuole un CI che consumi troppo, sono intermedi sia in termini di effecenza che in termini di programmazione.

6. *Per quali applicazioni un componente logico programmabile è più indicato di un microprocessore?*
   Applicazione ad elevate prestazione e con un elevato numero di componenti venduti.

7. *Elencare le tecnologie di memoria usabili nei circuiti logici programmabili.*
   RAM; Registri; PROM; EPROM; EEPROM; FLASH.

8. *Quando è conveniente realizzare sistemi elettronici su componenti programmabili?*
   –

6 Verilog

Il Verilog è un linguaggio di programmazione con la concorrenza come aspetto principale, descrivendo il comportamento dei vari moduli. Esistono anche altri linguaggi come il VHDL e SystemC. L'idea per la nascita di verilog è simulare il comportamento di un circuito logico.

La sintassi del Varilog è molto simile al C. Un modulo è un componente elettronico che può essere usato con degli ingressi e delle uscite:

- Definizione del modulo: le uscite vanno prima degli ingressi;
- Lista di porte;
- Lista delle dichiarazioni delle porte;
- Dichiarazione dei parametri;
- Istruzioni del preprocessore;
- Descrizione del comportamento tra un modulo;

Il modello di programmazione base nel varilog sono le porte logiche primitive. Ogni segale interno al modulo va definito come `wire`, le porte sono definite implicitamente come `wire`.

Per descrivere i ritardi si una il `#`. Un ritardo ha la forma:

```
#(Tp);  #(Tmin:Typ:Tmax);  #(Tplh, Tphl);
```

L'unità di misure è definita da una direttiva:

```
1   `timescale <time_unit>/<time_precision>
```

Example 6.1 – Half Adder Modulo

```
1 module hadd(Sum, Carry, X, Y);
2 input X, Y;
3 output Sum, Carry;
4
5 xor #(2:3:4, 5) (Sum, X, Y);
6 and #(3.567) (Carry, X, Y);
7 endmodule
```

Si possono creare più istanza di un modulo anche dando un nome.

```
1   ...
2   hadd M1 (...)
3   hadd M2 (...)
```

Le assegnazioni continue sono uno dei modi di assegnazione dai registri. È un modo ad alto livello di assegnazione della logica combinatoria, dove il tempo è scandito dal segnale di clock e le assegnazioni avvengono quando un operando cambia valore. Le espressioni possono contenere espressioni booleane aritmetiche. La destinazione deve essere `wire`:

```
assign out = a & b | c;    // espressione booleana
assign eq = (a + b == c); // soommatore e comparatore
wire #10 inv = ~in;        // inverter con ritardo 10 time unit
wire [7:0] c = a + b;      // somma a 8 bit
```

Example 6.2 – Porta myXOR

```
module my_xor(C,A,B)
output C;
intput A, B;

assign #2 C = (A ^ B);
endmodule
```

2 timeunit dopo che A o B sono cambiati di valore il risulatato di A xor B viene assegnato a C.
IMPORTANTE: I ritardi servono solo per simulazione, non hanno significato per la sintesi.

Example 6.3 – Sommatore a 4 bit

```
module adder4 (Sum, Cout, A, B, Cin);
input [3:0] A, B;
input Cin;
output [3:0] Sum;
output Cout;
assign {Cout, Sum} = A + B + Cin;
endmodule
```

{Cout, S} è un concatenazione di bit.

Si possono usare variabili definite con la keyword `parameter`, che sono ridefinibili durante l'instanziazione del modulo.

Quando si usa un modulo si possono usare le nominazioni dei parametri e delgi argomenti, questo è utile quando si hanno centinaia di argomenti e parametri.

```
adder adder_8 #(8) (co, s, a, b, ci);
adder adder_8 #(width=8) (.a(a), .b(b),
    .cin(ci), .cout(co), .sum(su));
```

Si possono lasciare dell uscite non connesse non nominando gli argomenti durante l'instaziazione.

Gli **always block** sono processi concorrenti, rappresentati come blocchi di codice procedurale che contengono istruzioni sequenziali in cui l'ordine di moduli ed instaziazione non conta. L'always posside una **sensitivity list** che definisce esattamente quando un always block viene eseguito una volta dalla prima istruzione all'ultime. Un'always può essere:

- attivo sul livello (latch o logica combinatoria):

```
always @(a or b) || always @(*)
                                \|
    tutti i segnali che compaino sulle espressioni
```

- attivo sul fronte (FF sincrono):

```
always @(posedge Clock or Reset)
```

In verilog esistono due tipi di dato, quelli senza memoria `wire` e quelli con memoria `reg`. I `wire` sono le usciute di tutti gli `assignment`, i `reg` sono le uscite di tutti gli always block.

Esistono anche delle espressioni condizionali con cui si possono creare ad esempio le istruzioni di una ALU:

```
if( alu_func == 2'b00)
    aluout = a + b;
else if( alu_func == 2'b01)
    aluout = a - b;
else if( alu_func == 2'b10)
    aluout = a & b;
else
    aluout = a | b;
end
```

Usando gli switch-case si possono usare dei don't care per effettuare delle minimizzazioni.

Per modellare un latch si usa un modo canonico che tutti i sintetizzatori possono capire.

```
module latch (CLK, Reset, D, Q);
input CLK, Reset;
input D;
output reg Q;

always @(D or Reset or CLK) begin
    if(Reset)
        Q = 0;
    else if(CLK)
        Q = D;
end;
endmodule
```

0ppure un FF con reset sincrono:

```
module FF (CLK, Reset, D, Q);
input CLK, Reset;
output D;
output reg Q;

always @(posedge CLK) begin
    if(Reset)
        Q = 0;
    else if(CLK)
        Q = D;
end;
endmodule
```

6.1 Flusso di Progetto

Passa attraverso dei passi standard.

1. L'ingresso è il modello RTL della funzionalità da relizzare;
2. Verifica delle specifiche, con la simulazione a livello funzionale, all'ingresso della simulazione si possono introdurre delle test bench scritte in verilog;
3. Si effettua la sintesi logica, ad esempio: periodo di clock che si vuole utilizzare, ecc... ottenendo un circuito logico che corrisponde alla funzionalità di quello in entrata;
4. Il circuito creato dalla sintesi logica verrà piazzato sulla FPGA, decidnendo lo posizione fisica dei blocchi e le modalità di collegamento;
5. Viene genereato un bit-stream: volori che verrano caricati dentro l'FPGA per configuralrla.

Sintesi Logica:
Fa un lavoro di compilazione, crearo attraverso la semntica del linguaggio, un circuito corrispondente non ottimizzato. Si usano in una fase successiva, i teoremi dell'algebra booleana per minimizzare il numero di porte logiche, minimizzando costi e ritardi. L'uscita della sintesi logica sono un insieme di porte collegate tra di loro, scritte in un liguaggio per descrivere i circuti. Si ottiene anche un'approssimazione dei tempi di ritardo (si sapranno solo quando i circuiti saranno piazzati sul silicio). Usata l'algebra booleana si traducono gli insiemi di porte in look up table (LUT). Una volta ottenute le LUT si possono andare a piazzare sul silicio, cercando di minimizzare la lunghezza delle interconnessioni.

–

Per progrettare gli ASIC, si spende un tempo motlo maggiore per verifcare che nel circuito non siano presenti dei bug. Le verifiche costano molto di più.

Flusso per ASIC

1. Verifica del codice verilog;
   –
2. Test Insertion: si inserice della logica per collaudare il circuito;
3. Analisi dei ritardi;
4. Fase dei piazzamento delle interconnessioni;
5. Tracciamento dell'albero del clock (molto difficile);
6. Fase di estrazione delle maschere fisiche, si effettua la verifica che la logica della maschere sia effettivamente uguale alla logica RTL.

In FPGA le porte sono già piazzate sul silicio, in cui le LUT vanno piazzate, mentre negli ASIC l'organizzazione è piazzata su delle righe (simil pali della luce), un cattivo piazzamento può causare delle lunghe interconnessione che cuasa delle congestioni (frequenza minore).

Dopo la fase di piazzamento vengono effettuate le interconnessioni. Si usano algoritmi come quello di Dijkstra, tenendo conto dei ritradi e delle congestioni.

7 Memorie a Semiconduttore

Classificaionzi:

- Possono essere sia lette che scritte;
- Possono essere

Tutte le memorie usano dei protocolli di accesso, che vengono attuati attraverso dei segnali di controllo (molto simili al clock), i dati possono essere letti o scritti dopo un tempo di stabilizzazione. In questo caso possiamo avere due tipi di segnali: READ e WRITE. Il motivo per cui i segnali sono separate è perché i dati letti e scritti viaggiano sugli stessi circuiti. Ci sono dei ritrdi minimi e massini tra le letture e le scritture. Nei FF questi ritardi sono unici, nelle memorie i tempi minimi sono molto differenti.

Da un punto di vista logica le memoria sono organizzate in gruppi di parole in cui ogni parola ha un'indirizzo, ogni lettura trascferisce una quantità di dati in blocchi di parole. Internamente l'architettura è organizzata in una matrice approssivativamente quadra, minimizzando le lunghezze, di fatto l'indirizzo a N bit viene suddiviso in due parti, una che seleziona la riga ed una che seleziona la colonna.

Le righe attivano la **wordline**.

7.1 DRAM

Tra le memorie la DRAM ha la cella di memoria più piccola, formata da:

- Un filo di bitline;
- Un filo di wordline;
- Un transistorn permette di caricare la capacità C_S, questa ccapacità perde continuamente elettroni duvuto alla corrente di perdita del trasistor, se il potenziale è basso si legge L se il potenziale è alto si legge H, questa capcità è molto piccola (perchè si vuole usare la minor area), è più piccola delle capacità parassite del filo di BL. In scrittura si attiva la wordline e si porta la capacità a potenziale alto o basso. In lettura BL viene caricata $V_{dd}/2$, è più facile caricare o scaricare di poco la line, il valore reale sarà eleborato successivamente con dei circuiti differenziali.

Fatta una lettura il dato viene distrutto, quindi dopo ogni lettura una riga deve essere riscritta, in questo modo viene anche ripristinato dalle correnti di perdita. Queste operazioni di riscrittura devo essere fatte periodicamente, questa operazione viene nominata **rinfresco**, in memorie questa procedura viene implementata automaticamente a livello di HW.

Per la lettura viene usata una cella di memoria caricata a $V_{dd}/2$, il seganle letto e il segnale della cella dummy viene mandato in un amplificatore differenziale. Questo viene fatto per minimizzare il disturbo dal rumore, essendo le due wordline in parallelo hanno gli stessi disturbi (eliminati dall'amplificatore differenziale).

...

L'operazione di pilotaggio dei Sense Amplifier sono molto più lente delle logica combinatoria.

...

Avere tante sotto RAM piccoline migliora la velocità, diminuendo le capacità parassita del BL.

...

Uno dei primi metodi di lettura à la **fast page page mode** (o **burst**), in cui si leggono le colonnes successive alla parola letta, molto veloce.

Oggi le RAM sono tutte sincrone (hanno un clock che temporizza), inoltre

...

Si possono usare le pipeline con una **burst lenght**, mentre si legge una riga si preparano le parole successive. I burst possono essere usati sia in lettura che in scrittura.

7.2 SRAM

Si possono usare gli inverter per realizzare. Ci servono due BL complementate, il motivo è che non risuciremmo a invertire lo stato del circuito bistabile. La velocità è dell'ordine della logica combinatoria. Ci sono dei trasistor di precarica che portano ...

Il sense amplifier è fatto da un

Come la DRAM abbiamo il WE ed il OE (Write Enable e Output Enable). Oggi si usano le SRAM sincrone

All'interno della RAM sono presenti delle parti di memoria ridontdai per correggere gli errori. Per momorizzare i bit si usano dei codi di autocorrezione, oppure segnalare un errore. Al posto di usare memoria sul silicio ridondante si possono usare questi codici di autocorrezione.

7.3 Memorie Indirizzabili a Contenuto

(CAM) (Content Addressable Memory) Al contrario delle RAM diciamo il contenuto che ci interessa e risponde con l'indirizzo o gli indirizzi a cui i volori corrispondono (potreebbe anche non esistere).

Il ciclo di scrittura funziona alla stesso modo delle RAM.

Per leggere l'indirizzo si utilizza un modo molto simile all'indirizzamento dei pacchetti IP. La CAM viene usata ad esmepio nelle cache.

7.4 Memorie Non-Volatili

Queste memorie possono essere di tue tipi differenti:

- Read-Only: la memoria può essere scritta solo in fase di programmazione
- Riprogrammabili: possono essere riscrivibili, questo però prende più tempo

Nelle RO si usano i transistor per portare a livillo basso o alto il segnale, mettre si usa un circutito vuoto per il complementare.

Si usano i transistor come pull-up perché sarebbe più complicata fare una resistenza su un transistor.

Anche se oggi non sono più usate i due tipi di NAND e NOR vengono ancora usato nelle memorie riprogrammabili.

7.5 FLASH

Le momerie flash (vengono dette cosi perchè quando viene effettuata la scrittura è fatta a blocchi di parole). Per creare i bit si iniettano delle carica nei gate flottanti. (NOR) Per cancellare la carica iniettiamo una carica negativa sul gate flottante, che andraà a schermare il pontenziale positivo della WL evitando che si formi il canale sulla BL.

(NAND) Si iniettano cariche positive sul gate flottante molto. SSL e GSL selezionano alcuni blocchi all'interno della Flash.

NAND Flash: si fa tunneling per iniettare le cariche nel canale, per mandare via le cariche si usa usa un potenziale alto sul canale. La struttura predominante è quella orizzontale. Per sfruttare più spazio su usa la terza dimensione.

Le flash sfruttano a pieno la terza dimensione.

Le scritture singole sono possibili solo sulle NOR, mentre la riscrittura a blocchi è possibile su entrambe. La lettura a accesso casule della NAND è 200 volte più lenta della NOR, per il primo accesso, mentre la lettura delle parole adiacenti è molto veloce. Le operazioni di scrittura sono più veloci nella NAND, la banda di scrittura è molto maggiore nelle NAND. Eliminare un blocco nella NOR è molto più lento che nelle NAND perchè si deve cancellare la carica dei singoli transistor.

Le operazione sulla FLASH: La scrittura individuale si può fare solo settndo a 0. Dunque si deve prima mettere tutto il blocco ad 1 e poi andare a mettere gli zeri ad ogni cella.

L'organizzazione è suddiva in: blocchi, pagine, byte. (La cancellazione avviene suli blocchi).

La vita di una memoria riprogrammabile è finita, gli elettroni passano attraverso ed ad ogni riprogrammazione il materiale si rovina, dunque dopo un numero di cicli di scrittura i gate non più a mantenere le cariche. Allora hanno bisgno di un controllore delle memoria intelligente che distribuisce le momoria uniformemente nei vari blocchi, in modo da evitare la rottura.

8 Interconnessioni

I problemi dei sistemi di interconnessione sono divisi per: potenza (trasporto di energia), segnali (trasporto dell'informazione). La priorità nelle linee di interconnessione è l'**integrità del segnale**: mantenere l'informazione consistente. Le linee di interconnessione per trasferire dei dati usano un livello basso ed uno alto, interpretati come zeri e uni che sono valori interpretati da tensioni e correnti.

8.1 Rumore Temporale

Supponiamo di avere due segnali che hanno delle transizioni, questi due segnali rappresentano i dati da trasferire e devono essere sincroni, può accadere che le transizioni non siano sincrone tra di loro, la differenza temporale tra le transizioni dei segnali è detta **skew** t_K. Supponiamo invece di avere un segnale che possiede una temporizzazione (ovvere deve avere una transizione in un dato momento), ma questa avviene in anticipo o in ritardo, il tempo di ritardo è detto **jitter** t_J. Jitter e Skew portano a problemi di temporizzazione dei segnali.

Un modello ideale di interconnessione è un filo equipotenziale senza rumori tra il driver ed il receiver. In generale per analizzare il passaggio delle informazioni si usa un modello ISO/OSI, quindi vengono usate delle interfaccie che comunicano con gli stessi livelli. Si studiano i livelli più bassi: livello fisico e data link.

8.2 Livello Elettrico

Le linee di interconnessione che si trovano a livello fisico (correnti e tensioni), collegano dei circuiti digitali da dei driver e da dei reciver che convertono il segnale da digitale ad analogico e viceversa. Gli obbiettivi sono: la massima velocità ed assenza di errori.

Un driver viene modellato come un generatore di tensione in serie con un resistenza, mentre un reciver è modellato come con resistenza a massa (di solito infinita) ed una capacità a massa, che comporta un ritardo nelle transizioni.

Figure 8: Modello Driver Reciver

Per modellare un linea di interconnessione si possono usare diversi approcci. Un modello ideale è una linea equipotenziale. Un modello più relistico è un modello RC, in cui i tempi di ritardo dipendo dalla costante di tempo $\tau = RC$ e dalla tensione di soglia, il ritardo massimo dipende da V_{IH} mentre il tempo minimo dipende da V_{IL}.

Il ritardo con cui viene rilevata una rilevata una variazione di stato logico è detto ritardo di trasmissione (anche detto ritrado di propagazione nel modello simplificato) t_{TX}. Quando si ha una variazione della costante di tempo si hanno due tempi di ritardo, uno minimo ed uno massimo: per definizione il **tempo di skew** è proprio la differenza tra questi due, infatti ogni segnale può avere un tempo di trasmissione compreso tra i due.

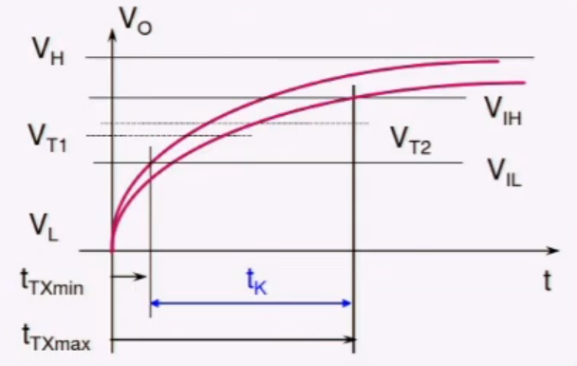


Figure 9: Variazione Di Skew

9 Modelli a Linea

Si studia la propagazione di segnali all'interno di un filo, dove gli obbiettivi sono:

- massima velocità;
- assenza di errori;

Modello RC: il filo presenta un resistenza in serie ed una capcità a massa che rappresenta le capcità parassite.

Si può considerare un filo composto da piccoli pezzettini, dove le resistenze in serie e parallelo possono essere trascurate, portando questi pezzettini a lunghezza infinitesima si ottiene la **linea di trasmissione**. Se le resistenze sono trascurabili la linea è detta **senza perdite** (induttanze e capicità non dissipano potenza).

9.1 Paramentri di una Linea di Trasmissione

I parametri sono:

- Z_∞: impedenza caratteristica;
- lunghezza;
- velocità di propagazione;

- tempo di propagazione;

Se la linea è senza perdita il segnale in entrata passa in modo intatto all'altra parte della linea.

Trasmissione senza perdite:

- R in serie nulle: buon conduttore;
- G parallelo nullo: buon isolante;

Con perdite:

- all'interno di circuiti integrati;

Si può definire la capacità e l'induttanza per metro di linea:

- L_u: induttanza unitaria;
- C_u: capcità unitaria;

Il **parametri elettrici**:

- impedenza caratteristica:
$$Z_\infty = \sqrt{\frac{L_u}{C_u}}$$
- velocità di propagazione:
$$P = \frac{1}{\sqrt{L_u C_u}}$$
- tempo di propagazione (tempo che impiega il segnale per spostarsi lungo la linea):
$$t_P = \frac{L}{P}$$

Sono costretto ad usare un modello a linea quando i tempi di transizione (t_r, t_f) sono piccoli rispetto ai tempi di propazione (t_P).

9.2 Riflessioni

Supponiamo di avere un driver collegato al nodo B, che entra in una linea di trasmissione con: Z_∞ e t_P. All'ingresso avrò un segnale attenuato, dato dal partitore tra R_O e Z_∞.

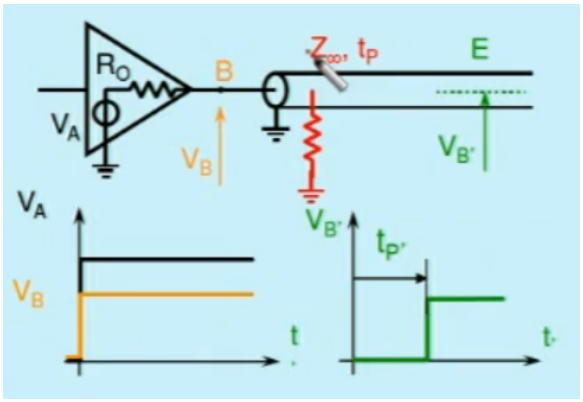


Figure 10: Liea Pilotata Con Gradino

Il gradino che si sposta progressivamente dal trasmettitore a ricevitore è detta **onda progressiva**, dove in ogni punto della linea vale $Z_\infty = \frac{V(t)}{I(t)}$, che rimane costante. Se si ha una variazione di impedenza in un punto della linea (passaggio da una $Z_{\infty 1}$ a $Z_{\infty 2}$), il circuito non è dissipativo, dunque il rapporto $\frac{V}{I}$ deve variare, questo porta la nuova Z_∞ a non assorbe tutta l'energia dell'onda progressiva, quindi l'unica soluzione è che la linea genera un'onda che ritorna indietro, l'onda che torna indietro è detta **onda riflessa**. Il segnale riflesso viene riflesso a sua volta quando torna indietro e così via, creando una serie di rimbalzi infiniti.

L'onda riflessa ha ampiezza:

$$\boxed{V_r = \Gamma_T V_p}$$

Con V_r: tensione riflessa; V_p: tensione propagata.

Definition 9.1 – Coefficente di Riflessione

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

In generale l'onda riflessa ha un'ampiezza, dopo una discontinuità da Z_∞ a Z_1 di:

$$V_r = \Gamma_1 V_p \implies \Gamma_1 = (Z_1 - Z_\infty)/(Z_1 + Z_\infty)$$

Nel caso dinamico si aggiunge una **terminazione** che ha impedenza infinita pi˘ vicino possible alla Z_∞ della linea, collegata al terminale del receiver.

Per rappresentare le trasmissioni si usa (la più comune) un diagramma a traliccio.

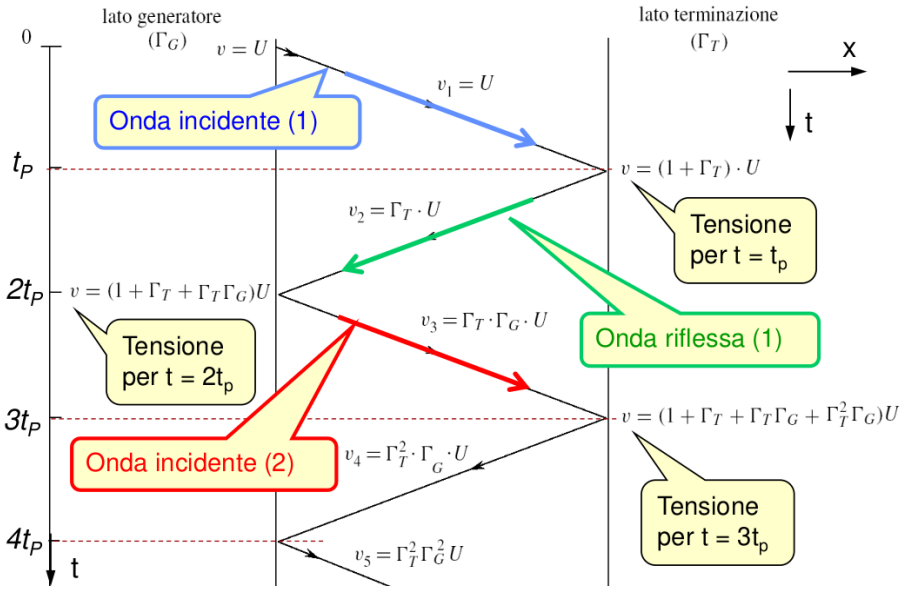


Figure 11: Diagramma A Traliccio Della Riflessione

Ogni coefficente è minore di 1 (la successione converge a 0), questo porta le riflessioni a convergere verso un valore.

10 Connessione con Linee

It tempo di skew è dato dal tempo di trasmissione massimo meno il minimo.
Parametri del trasmettitore e ricevitore: $V_{OH|OL|IH|IL}$. I parametri che influenzano t_{TX} e t_K:

- parametri di TX e RX (V_H, V_L e V_T);
- propagazione: riflessioni, terminazioni (impedenze all'inizio ed alla fine), discontinuità, diafonia;
- carico (in particolare cariche capacitivi);
- rumore di massa;

10.1 Tipologie di Connessione

Collegamenti punto-punto: un unico driver e receiver, dove le condizini di propagazinoe sono ben definite.

Collegamenti a BUS: si possono collegare più periferiche diverse tra di loro, c'è uno o più driver, anche nel mezzo della linea, che comunicano con più receriver diversi tra di loro.

La struttura di riferimento sarà:

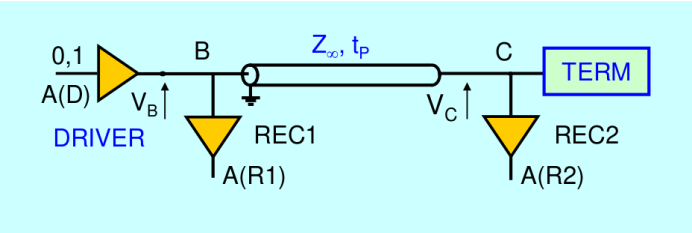


Figure 12: Struttura Di Riferimento Punto Punto

Il primo gradino viene impresso dal driver, che viene caricato da Z_∞, la terminazione determina una riflessione. Suppuniamo:

- basso $R_O/Z_\infty \rightarrow$ bus veloci: l'onda incidente è molto ampia e non sono necessiare riflessioni, ma il coefficente sul driver è minore di 1, dunque potrebbero crearsi delle oscillazioni e dunque provocare delle commutazioni multiple nel receiver. Ci vuole un driver ad alta corrente, la terminazione dissipa energia.

- alto $R_O/Z_\infty \rightarrow$ bus lenti (possono volerci più riflessioni per superare la tensione di soglia): il primo gradino è basso, dunque si hanno bisogno di più riflessioni per raggiunger la soglia, ma avendo R_O alto il coefficente al driver è positivo dunque non si creano oscillazioni. I due coefficenti di riflessione sono positivi, il motivo per cui anche il Γ_R è positivo, implica che la resistenza al receiver è maggiore di quella del driver, se non fosse così la resistenza sarebbe vicino a massa, e le riflessioni tenderebbero asintoticamente a $0V$. Con driver a bassa corrente la tensione di soglia viene attraversata dopo diversi tempi di propagazione. Se l'andamento è molto lento il tempo di skew diventa molto alto (la differenza tra i tempi trasmissione massimo e minimo sono molto alti).

- driver adattato $R_O = Z_\infty$: adattamento lato driver, il primo gradino è pari alla metà della tensione a vuoto, si deve aspettare almeno una riflessione per superare la tensione di soglia. L'onda riflessa non si riflette a sua volta, il motivo è che il coefficente al driver è nullo ($R_O = Z_\infty$). Ha anche un consumo minore perchè non c'è bisogno di adattamento al receiver (*far end*).

10.2 Tipi di Terminazione

Terminaione in Parallelo: terminazione tra il filo della linea e massa (o una tensione fissa, come la tensione di alimentazione). Se metto una $R_T = Z_\infty$ non ho riflessione. Ha basso ritardo ed alti consimi ma può essere pilotato in un punto intermedio.

Terminaione in Serie: adatto il driver con una R_S tale che: $R_S + R_O = Z_\infty$. Ho anche una $\Gamma_T = 1$. Ha basso velocità e bassi consumi ma non può essere pilotato in punto intermedio.

Gli effetti del carico capacitivo diminuiscono l'impedenza caratteristice e la velocità di propagazione. Questo rendi il sitema più lento e rende il driver non più adattato, oltre ad aumentare la potenza consumata. Bisogna limitare gli effetti delle cariche capacitive, ad esempio un clock che deve far commutare 100 driver, si usa un buffer che carica i drivver al posto della linea diretta.

10.3 Collegamenti Multi-Punto

Nei BUS si devono cercare di limitare le derivazioni ed i carichi capacitivi (se colleghiamo il BUS direttamente la capacità totale vista all'inizio della linea sarà molto ampia, dunque i tempi saranno lunghissimi, ad avare tutte i carichi capacitivi dati dai driver e dai receiver presenti sulla linea), per evitre questo non si collega mai un linea in modo diretto alla piastra ma si usano, ad esempio, dei **transceiver** o dei buffer.

Vediamo come l'effetto dei tensioni viene passata al livello superiore nella gerarchia ISO/OSI. Per ottenere velocità elevate occorre minimizzare lo skew e garantire delle specifiche temporali, creando così dei protocolli a livello di ciclo, che devono garantire l'integrità dei dati trasmessi.

10.4 Problemi

Problem 10.1

Un'interconnessione punto-punto ha:

- $Z_\infty = 70\Omega; t_p = 10ns;$
- Receiver: $V_{AL} = 5V; V_{IH} = 3V; V_{IL} = 0.8V;$

Calcolare i coefficenti di riflessione al driver ed al receiver, il tempo di trasmissione e skew per le coppie di resitenze:

$$(R_O, R_T) \implies (100, \infty); (100, 200); (50, 200); (50, 100); (10, 100); (10, 10);$$

- $(100, \infty)$:

$$\Gamma_D = 0.18;\ \Gamma_R = 1$$

$$t_{TX} =;\ t_K =$$

- $(100, 200)$:

$$\Gamma_D = 0.18;\ \Gamma_R = 0.48$$

$$t_{TX} =;\ t_K =$$

- $(50, 200)$:

$$\Gamma_D = -0.16;\ \Gamma_R = 0.48$$

$$t_{TX} =;\ t_K =$$

- $(50, 100)$:

$$\Gamma_D = -0.16;\ \Gamma_R = 0.18$$

$$t_{TX} =;\ t_K =$$

- $(10, 100)$:

$$\Gamma_D = -0.75;\ \Gamma_R = 0.18$$

$$t_{TX} =;\ t_K =$$

- $(10, 10)$:

$$\Gamma_D = -0.75;\ \Gamma_R = -0.75$$

$$t_{TX} =;\ t_K =$$

11 Cicli Base di Trasferimento

Esistono due metodi per creare una sincronizzazione tra la linee di trasmissione: sincrona (scandita da un clock), asincrona. Si passa dal livello **fisico** al livello **cilco**. Nel livello di ciclo si trasferiscono singoli stati logici: interpretati a partire da delle tenioni. Nel livello di ciclo si trasferiscono gruppi di bit: i vincoli importanti da rispettare sono i tempi di set-up e di hold, visto che si opera con dei FF.

Per trasmettere un dato si necessitano di almeno due segnali (uno porta il dato, uno porta il clock ad esempio). Al driver ed al receiver si devono ripettare i tempi di setup e di hold. Date due linee di trasmissione esistono due tempi t_1 e t_2 (t_0 = tempo iniziale):

$$t_1 = t_0 + t_{TXmedio} + t_K$$

$$t_2 = t_0 + t_{su(D)} + t_{TXmedio}$$

La differenza tra questi due tempi è proprio il tempo di setup del receiver:

$$t_{su(R)} = t_2 - t_1 = t_{su(D)} - t_K$$

Lo skew riduce i margini di temporizzazione.

11.1 Sorgente-Destinazione

Per modellare questi ritardi si una modello **sorgente-distinazione**, la destinazione è un FF che fa una fotografia in un istante in cui si è sicuri che tutti i bit abbiano un giusto valore. Il clock comune a tutti i FF non viene collegato direttamente ma viene usato un buffer, che ha il vantaggio di avere riflessioni molto più lente che non con un collegamento diretto al clock.

Esistono due tipi di trasmissione di dati:

- Una sorgente vuole **srivere** dei dati su un destinatario: la sorgente manda i dati ed segnale di controllo verso una destinazione;
- Una destinazione vuole **leggere** dei dati da una sorgente: la destinazione richiede dei dati, il flusso dei dati e dei segnali di controllo viaggiano in verso opposto;

Per evitare la metastabilità i tempi di setup e hold devono essere rispettati, due tecniche base sono: protocollo sincrono (a temporizzazione fissa, dove vengono dati dei riferimenti temporali fissi, dove vengono cosiderati sempre i casi peggiori, solitamnte scanditi da un clock); temporizzazione asincrono (temporizzazione adattiva, dove i tempi sono variabili e le terminazioni delle operazoini sono dettate da dei segnali di ACK).

11.2 Scrittura

Si hanno cicli di scrittura con dei ritardi fissi (t_A, t_B), dove c'è un intervallo temporale di scambio di informazioni, al termine di questo inervallo i dati non cambiano più. Essendoci dei ritardi prefissati noti a priori è possibile rispettari i vincoli sul ricevitore di setup e di hold, perchè tutti i parametri sono controllati dalla sorgente.

Al tempo t_0 i dati sono stabili, al ricevitore i dati sono incerti prima dell'arrivo dei dati dal ricevitore, dunque il tempo dopo la quale i dati al ricevitore saranno stabili sarà il tempo di ritardo massimo, perchè come al solito si devono considerare i casi peggiori, ma anche il datsarà anche stabile per meno tempo, perchè il tempo minino dopo che i dati non vengo più trasferiti dal trasmettitore è il tempo minimo di trasmissione. Il tempo di setup è il tempo in cui l'input deve essere grantito stabile prima del colpo di clock al FF, questo tempo corrisponde alla differenza tra l'ultima transizione del dato alla prima transizione del clock: $t_{su} = t_A + t_{TXmin} - t_{TXmax} = t_A - t_K > t_{su(FF)}$.

Analogamnte per il tempo di hold, si calcola: $t_h = t_B + t_{TXmin} - t_{TXmaz} = t_B - t_K > t_{h(FF)}$, dove la scelta dei tempi dipende solo dallo sckew e dei tempi di setup. Anche qui lo sckew diminuisce i margini di temporizzazione. Con questo metodo si possono calcolare i valori di t_A e t_B, in modo da rispettare i vincoli sul destinatario.

Nei casi di scritture **broadcast** (scritture sui più destinatari contemporaneamente), si devono rispettari i tempi del destinatario più lento.

11.3 Ciclo Asincrono

Di base si ha sempre una sergente che trasmette dei dati ad una destinazione, come prima si ha una transizione per strobe. Lo strobe arriva ad una macchina a stati che determina quando mandare il tempo di clock al FF ed inoltre manda indietro al trasmettitore un segnale di ACK. L'interlacciamnento STB/ACK è detta **handshake**.

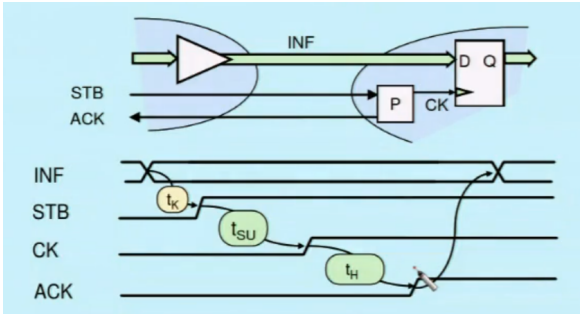


Figure 13: Ciclo Asincrono

I ritardi (di setup, di hold e di skew) generati dalla macchina a stati, sono creati da un ulteriore clock ad alta frequenza che attraverso un logica sincrona che conta il ritardo, è molto conveniente usare un clock a frequenza più alta perchè la logica

ascincrona è molto lenta duqnue, è molto facile generare un clock con dei margini elevati.

Quando l'ACK al trasmettitore deve essere a 0 prima di far ripartire un ciclo sono detti **cicli chiusi**, altrimenti **cicli aperti** in cui l'analisi è più complicata ma sono più efficenti.

12 Protocolli di BUS

Le tecniche di indirizzamento sono gestite da un singolo master, i sistemi con più master I servizi forniti dal livello di ciclo forniscono dei dati fornendo delle specifiche temporali. Il livello di ciclo non si interessa del tipo di dato trasmesso. I protocolli estesi permettono il trasferimento di dati in madalità 1-a-N e N-a-M (N master e M master). Fornire questi servizi comporta un consumo di energia e di tempo, determinari dal livello di ciclo.

Vengono definiti degli indirizzi per identificare i diversi dati, uno o più dati posso essere passati da un sistema ad un altro, questo viene detto **transazione**, dove si passa un unico pezzo di informazione, il master incomincia la transazione.

Il trasferimento punto-punto comporta la conoscenza completa del trasmettitore e del ricevitori.

I sistemi multi-punto comprendono più master e più slave. Più unita usano lo stesso BUS fisico, una volta identificati i participanti alla transazione si iniziono i cicli di lettura o scrittura.

Nei sistemi a BUS è possible rimuovere o aggiungere dei moduli (**sistema modulare aperto**), avento tempi di sckew, setup e hold variabili (non sono uguali per moduli diversi). Non è possibile mettere un numero indefinito di moduli perchè comporterebbe ad un ritardo massimo indeifinto, per questo motivo si definiscono delle regole ben procise a cui tutti i moduli devono obbedire (protocolli), altrimenti si rompe tutto.

Se devo collegare più sistemi in modalità punto-punto le specifiche sono ben definite, questo però richiede di fare delle procedure di routing, mentre nel mutlipunto ho un unica BUS dove l'unica azione che deve essere performata è l'arbitraggio. Nel punto-punto si ottengono velocità maggiori che nul multi-punto, nul multi-punto si deve tenere conto del sistema più lento, oltre a mantenere un'informazione che rappresenta il destinatario.

Nei sistemi a BUS si devono considerare dei cicli specifici per selezionare le unità partecipanti, in particolare decidere il master e lo slave. Genericamente c'è prima una fase di allocazione(l'unico master che può utilizzare il canale), l'indirizzamente (la decisione dello slave), il trasferimento dei dati tra master e slave.

Sistemi a singolo master: nello slave c'è un qualcosa che decodifica l'indirizzo comunicato dallo slave, che permette di decidere quale slave il master inizia a comunicare, per fare ciò si usano dei comparatori, decodificatori e logica. Ci sono delle temporizzazzioni che dopo il riconoscimento della connessione col master partono i controlli per lo strobe, e gli altri segnali. Questi dati si trovano all'interno dello slave allocati in una parte specifica del circuito, che verranno gestiti attraverso degli indirizzi, per distinguerli tra di loro. Per velocizzare la trasmissione si utilizzano anche negli slave dei buffer verso il BUS.

12.1 Tecniche di Indirizzamento

Per indirizzare gli slave esistono due tipi di selezione:

- codificata: ho N bit che selezionano 1 elemento tra 2^N elementi, ogni elemento ha un indirizzo assegnato a lui codificato;
- decodificata: si quando si hanno pochi slave collegati al BUS, la selezione avviene su M bit tra M diversi slave;

La selezione decodificata a differenza di quella codificata è più veloce perchè non va effettuata la decodifica.

L'indirizzamento può essere geografico (dove la selezione dipende dalla posizione rispetto agli slot collegati al BUS), o logico (dove la selezione dello slave dipende dal suo nome).

12.2 Protocolli Multi-Punto

Occorre identificare la coppia master-slave, che può essere decisa da un'unità centrale o dalla decisione comune. Quando due master richiedono contemporaneamente l'utilizzo del BUS viene generata una **collisione**, che assolutamnte evitata.

Un meccanismo per prevenire la collisione è un meccanismo a **token**: il gettone viene passato da master a master, nel momento in cui uno di loro deve usare il BUS ed ha il gettone in mano lo blocca per se ed inizia ad utilizzarlo, se non ha nessuna richiesta da fare lo passa al prossima master, per relizzare questo si devono definire dei meccanismi per passare il token.

Un altro meccanimso è quello del **collision detection**: se un master si accorge che il BUS è gia in uso, stoppa la sua connessione ed aspetta un tempo casuale prima di riconnettersi, altrimeti lo utilizza.

Un altro meccanimso è l'**arbitraggio**: un'unità centrale gestisce le rechieste e valuta a chi dare l'autorizzazione per il BUS, per gestire l'arbitraggio esistono due politoche:

- **temporale**: si usa politica first come first served;
- **logica**: si definisce una priorità diversa per le differenti periferiche, dove alcune hanno il diritto di precedenza sulla altre, utilizzando questa politica si deve evitare la **starvation**;

12.3 Valutazione di Prestazioni

Le prestazioni si misurano attraverso dei parametri, il più importante è il **throughput**, che è uguale al numero di bit trasferiti nell'unità di tempo:

$$T = \text{larghezza del bus} \times \text{velocità del bus}$$

La velocità del BUS si mirura in cilci al secondo: $= \frac{1}{t_c}$. La durata del ciclo t_x dipende da: parametri elettrici, parametri del modulo e dai parametri del protocollo.

Anche qui si hanno dei protocolli per il trasferimento dei dati, che possono essere sincroni ed asincroni. Il protocollo migliore è il **source synchronous** che permette ad entrambe le untità che stanno comunicando di gestire i segnali di controllo, in questo modo è necessario conoscere solo i ritardi.

I **BUS multiplati** sono le strutture base per BUS paralleli: infatti sui BUS multiplati gli indirizzi, i dati e i segnali di controllo viaggiano sulgi stessi conduttori, il motivo è che è molto oneroso in termini di spazion e di costo aver molti conduttori, pin, driver e receiverdiversi per ogni tipo diverso di dato. C'è bisogno di scandire dei cicli in cui ogni viaggiano solo dei tipi di dato nello stesso istante.

Un alro modo di migliorare le prestazione è nel ciclo standar di lettura e scrittura ci sono due cicli. Il consumo è dato dal numero di transazioni che vengono effettuate, durante una transazione si hanno due transizioni (una verso l'alto ed una verso il basso), viene sfruttato allora un cilce di **Double Data Rate** (**DDR** o **Dual-edge**), che sfrutta entrambe le transizioni dei segnali, compresi lo strobe e l'ACK, migliorando in questo modo i consumi e la velocità.

Source cynchronous

La parte di trasferimento del dato, di solito, è solo una piccola frazione del tempo di ciclo, nel trasferimento a blocco si fa un solo arbitraggio, mendando un unico indirizzo si iniziano a mandare quelli successivi, fino a N blocchi. Questo viene fatto perchè è molto probabile che i dati necessari siano vicini all'indirizzo che si è richiesto. Nelle memorie a indirizzamento a blocchi c'è bisongo di una logica di controllo prima della memoria che ricevuto un indirizzo genera quelli successivi che andranno a prendere i dati in memoria.

12.4 Domande

Quando e perché deve essere presente una operazione di indirizzamento? In quali sistemi è necessaria una operazione di arbitrazione? Descrivere il funzionamento di un arbitro di bus. Cosa si intende per "prestazioni" di un bus? Indicare i parametri che determinano le prestazioni di un bus. Da cosa dipende il tempo di ciclo minimo? Confrontare i tempi minimi di ciclo per protocolli sincroni e asincroni. Descrivere i vantaggi dei protocolli DDR. Quali vantaggi si ottengono con i bus multiplati? Descrivere i vantaggi di transazioni con trasferimento a blocchi.

— Se ci sono più slave, o se ho una memoria con molti dati;

— Quando si hanno più master;

— Decide chi è il master attraverso dei meccanismi di: token passing, collision detection, priority queue o coda LIFO;

— Dipende dal throughput (bit o byte o parole al secondo);

— Dimensione e tempo di ciclo (tempo di trasferimento, sckew, tipo di protocollo, se sincrono o asincrono, dual data rate, suorce synchronous);

— Che vengono sfruttate entrambe le transizioni del clock (velocità doppia e consumi ridotti);

— Meno connessioni, meno trasizioni, meno consumo, meno materiale, più logica per gestire i tipi diversi di dati.

— Meno tempo di indirizzamento, trasmettere più dati con un solo indirizzamento;

13 Collegamenti Seriali

Nella connessione seriale si ha un unico filo che trasmetti i dati in tempi successivi. Nel trasferimento vengono usati degli shift register per convertire i dati da parallelo a seriale e viceversa. Il trasferimento di N dati comporta N cicli. I pochi segnali riducono le interferenza elettromagntiche e il disallineamento temporale, causato dallo skew.

Esisteno dei collegamenti autosincronizzanti dove il clock viaggia sullo stesso filo dei dati (detto **clock-data embedding**). Il vantaggio dei collegamenti seriali è il ridotto numero di conduttori, alte velocità su lunghe distanze. I problemi del trasferimento serialo sono il trasferimento di N simboli diversi, oltre ad avere dei proble di sincronizzazione con il sistema ricevente.

Si parla di bit-rate per rappresentare il numero di bit al scecondo, ed di buad-rate, ovvero di numero di simboli al secondo. Esistono simboli binari, come l'NRZ: Non Return to Zero, e l'RZ: Return to Zero.

L'interferenza intersimbolica è l'interfernza causata da fili che si trovano vicini alla linea di trasmissione, che compartano un aumento del tempo di ciclo, il motivo è che porta e del rumore. Nel collegamento in parallelo viene detto **diafonia**, mentre nei seriali è detto **Inter Symbolic Interference**, causato dalle riflessioni sul filo, anche sul bit successivo.

I collegamenti sono fatti da passaggi da un registro PISO ad uno SIPO, i dati vengono trasferiti su una linea di trasferimento.

14 Cicli di Trasferimento

Come abbiamo già visto lo scopo delle linee di collegamento è qeullo di trasferire i dati tra più sistemi. Si deve passare necessariamente dal livello **fisico**, il trasferimento dei singoli bit, che introduce del rumore. Per trasferire dei dati è necessario definire dei protocolli per interpretare i dati.

A livello di **ciclo**, ovvero al livello di trasferimento di gruppi di bit (e.g. byte), il protocolle deve assicurarsi che all'arrivo del gruppo di bit, essi siano corretti ed abbiano un ritardo ben definito. Per descrivere un sistema i bit che viaggiano in parallelo hanno dei vincoli diversi, dati dai diversi parametri che ogni linea possiede. Appena si ha la certezza che i dati sono arrivati con certezza di procede a passare i dati al sistema.

Questi sistemi fanno affidamento su un livello fisico che viene gestito da un driver che passa i dati attraverso un collegamento che arrivano ad un reciver, che decofica i dati che gli arrivano e poi converti i segnali in dati per il sistema a cui è collegato.

15 Integrità di Segnale

La diafonia è un aspetto di interazione tra più linee, in cui il segnale su ogni linea interferisce direttametne con i segnali scambiati sulle altre linee.

Il **Crosstalk** o la **diafonia** è il passaggio di seganle. Gli accoppiamenti che si vengono a creare sono di tipo: induttivo e capacitivo tra conduttori diversi o di accoppiamento per maglie comuni per lo stesso conduttore.

15.1 Accoppiamento tra Conduttori

Le linee vicene tra di loro, possono avere degli accoppiamenti induttivi o capacitivi, causate dai disturbi elettromagnetici.

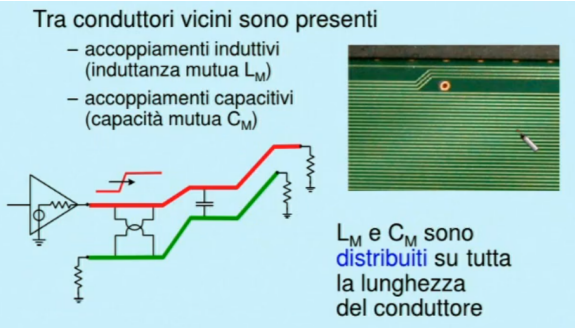


Figure 14: Accoppiamnto Tra Conduttori

Qunando si ha un linea pilotata, essa crea un disturbo, detta **linea disturbante**. Qunado il driver invia un segnale ci sono delle riflessioni, supponendo che la **linea disturbata** abbia una terminazione adattate, sulla linea si vede un linea incidente ed un onda riflessa.

I parametri di accoppiamento sono, e vengono causati dalla uniche transizioni dei segnali, i disturbi sono legati alla pendenza dei fronti.

Per ridurre il crosstalk si possono rallentare i fronti di salita (comporando un rallentamento dell'intero sistema) oppure usando dei segnali diffenziali (ognuno compensa gli errori dell'altro). Oppure si possono ridurre gli effeti del crosstalk mettendo: un filtro nel receiver o usando della tecniche di ritrasmissione dei dati.

15.2 Riduzione la Pendenza dei Fronti

Per ridurre la pendenza (derivata, $\frac{dv}{dt} = \frac{\Delta V}{\Delta t}$). Si può ridurre ΔV, con un consumo minore di potenza) o aumentare Δt con un uso di logica più lenta.

15.3 Accoppiamenti Induttivi

...

I vantaggi dei segnali differenziali, hanno immunità al segnale di **modo comune**, di può ridurre l'escursione senza senza l'aumento della sensibilità al rumore, inoltre la corrente totale è costante.

Gli svantaggi sono che si devono usare due piste e 2 piedini per ciascun segnale, inoltre richiede delle tecniche analogiche per ricostruire il segnale.

La distribuzione del clock comporta dei disturbi. O

...

16 Sistemi di Conversione A/D/A

I segnali analogici (continui nel tempo e nell'ampiezza), sono campionati in segnali digitali. Si hanno due discretizzazioni

17 Esercizi

17.1 Porte Logiche CMOS

Problem 17.1

1. Completare lo schema logico in figura;
2. Calcorare la t_{rise} al nodo E;
3. Calcolare la t_p della rete di pull-up e dell'inverter sapendo che all'uscita sono collegati 10 inverter;

I MOS hanno le seguenti caratteristiche:

$$R_{on}10k\Omega; \quad C_{gate} = 5fF; \quad V_{dd} = 1.5V.$$

Solution:

1. Sium
2. Fare traise
3. $\rightarrow$

Problem 17.2

1. Implementare una una porta cMOS con funzione: $U = (A \cdot B + C \cdot D)*$
2. Determinare t_p min e max sapendo che: $R_{on} = 20k\Omega$ e l'uscita U è collegata ad un carico di $C = 50fF$.

Solution:

1. ok
2. $\implies$

$$\begin{cases} t_{pHL,min} = 0.69 \cdot 50pF \cdot 20k\Omega = 0.69ns \\ t_{pHL,max} = 0.69 \cdot 50pF \cdot 40k\Omega = 1.4ns \end{cases}.$$

$$\begin{cases} t_{pLH,min} = 0.69 \cdot 50pF \cdot 20k\Omega = 0.69ns \\ t_{pLH,max} = 0.69 \cdot 50pF \cdot 40k\Omega = 1.4ns \end{cases}.$$

Problem 17.3

17.2 B2

Problem 17.4

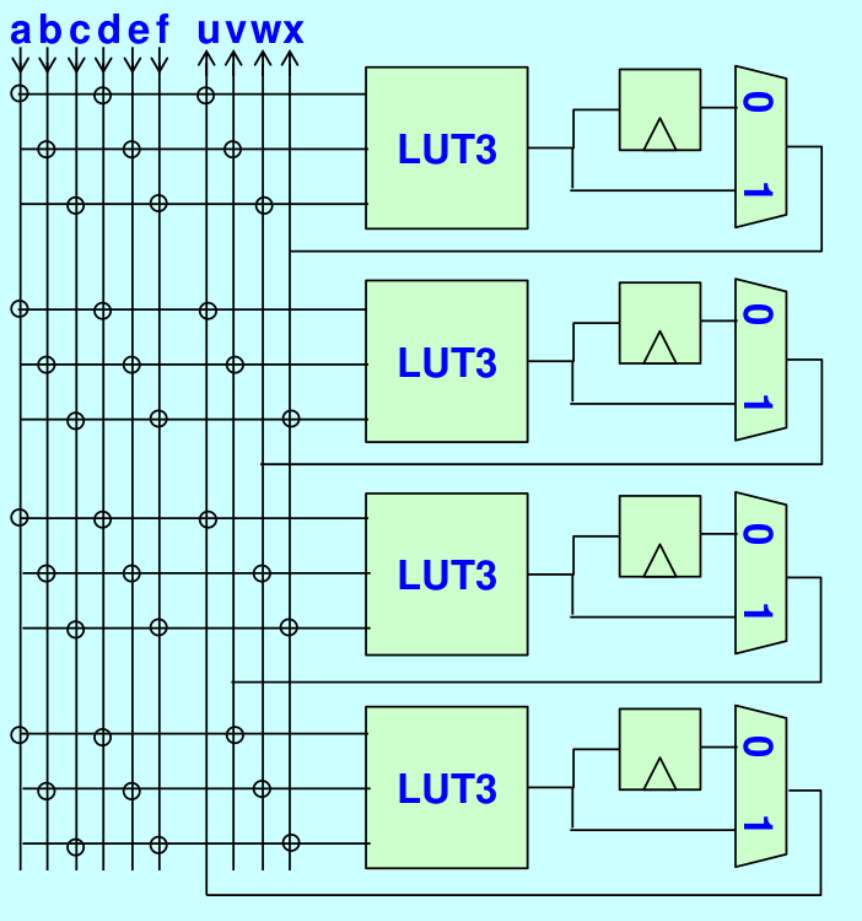


Figure 15: logica-programmabile-1.png

18 Esami

Il tempo di propagazione L-H di una porta CMOS NAND a 2 ingressi con una capacita di carico $C_G = 0.1fF$ collegata all'uscita $t_P = 200ps$.