Corso di Laurea Triennale in
Ingegneria Industriale

Prova Finale

Titolo

Relatore:
Dalla Betta Gian-Franco

Co-relatore:
Parmesan Luca

Studente:
Dalle Vedove Matteo, 203063

Anno Accademico 2020-2021

Ringraziamenti

Spazio dei ringraziamenti

Preambolo

Spesso nell'utilizzo comune di dispositivi digitali si dà per scontato il funzionamento intrinseco degli stessi, pensando che il computer ragiona solo con valori binari *0 ed 1*, *tensione alta e tensione bassa*. Spesso si dimentica però che, come ogni oggetto, anche i componenti che costituiscono i nostri device tecnologici sono, in origine, dei componenti analogici.

Lo scopo di questo documento è dunque quello di studiare come dei dispositivi analogici, in particolari i *transistor MOS*, possono essere utilizzati in ambito digitale, evidenziandone dunque i limiti fisici e dinamici legati alla loro implementazione. L'obiettivo sarà dunque quello di descrivere i principali circuiti che sono posti alla base di ogni calcolatore digitale, come le porte logiche, fino ad arrivare all'analisi di circuiti come il sommatore e moltiplicatore binario in tecnologia *c-MOS*.

L'approccio utilizzato per analizzare il problema sarà più di tipo simulativo: gli schematici sono realizzati tramite il software open source `XSchem` [4], mentre le simulazioni vengono effettuate mediante l'utilizzo del simulatore `ngspice` [8]. Per rendere tutto l'approccio il più reale ed applicativo possibile verranno utilizzati i modelli spice rilasciati pubblicamente mediante il progetto open source `google-skywater PDK` [2].

Indice

Capitolo 1

Introduzione all'approccio SPICE

A livello accademico è stato descritto il principio di funzionamento dei *mosfet*, ossia dei transistor che utilizzano l'effetto di campo che si instaura tra uno substrato semiconduttivo e un metallo ossidato per movimentare delle cariche elettriche. In base al drogaggio dei terminali di *source* e *drain*, complementare a quello di *bulk*, è possibile suddividere i transistori in due famiglie: gli *n-mos* (drogaggio di tipo n) e i *p-mos* (drogaggio di tipo p). In particolare la relazione statica che lega la corrente che scorre tra i terminali di drain e source è funzione della differenza di tensione V_{gs} tra *gate* e source, ma anche della differenza di tensione V_{ds} tra drain e source secondo l'equazione:

$$I = K_n \frac{W}{L} \left[(V_{gs} - V_{tn}) V_{ds} - \frac{V_{ds}^2}{2} \right] \tag{1.1}$$

In questa relazione è possibile osservare la presenza di 3 parametri fondamentali a determinare il comportamento del transistor: la *conducibilità intrinseca* K_n, proprietà caratteristica del processo tecnologico, e le dimensioni caratteristiche W (larghezza) e L (lunghezza) del canale conduttivo. Nella caratteristica statica fondamentale è anche la *tensione di soglia* V_{tn} che è funzione sia dalla tecnologia utilizzata per realizzare i transistor, ma anche dalla differenza di tensione V_{bs} tra bulk e source.

Il modello presentato in equazione 1.1 è in realtà una versione approssimata della caratteristica di trasferimento reale di un transistor MOS e trascura molti fenomeni elettromagnetici che nella realtà dovrebbero essere considerati, soprattutto per dimensioni dei canali conduttivi che al giorno d'oggi raggiungono dei valori estremamente piccoli (ordine dei nanometri); questo modello può essere utile a livello didattico per concepire il funzionamento di alcuni circuiti semplici, tuttavia per problemi più complessi un approccio analitico approssimato può portare a risultati fuorvianti rispetto ad un comportamento reale.
Un approccio simulativo è infatti più indicato per poter analizzare le prestazioni di circuiti più complessi in quanto a prova di errori (una volta che ci si è assicurati di aver implementato correttamente gli schematici) e permette di stimare effetti elettro-magnetici che analiticamente sarebbero impossibili da considerare.

In ambito elettronico per effettuare delle simulazioni numeriche di circuiti si utilizzano i software cosiddetti *SPICE* (acronimo di *Simulation Program with Integrated Circuit Emphasis*); in particolare tra le numerose soluzioni disponibili sul mercato nel proseguimento del seguente testo verrà utilizzato il software gratuito `XScheme` [4] per la realizzazione degli schematici che verranno successivamente simulati tramite l'applicativo `ngspice` [8].

1.1 Parametri di simulazione

Per poter effettuare delle simulazioni è necessario fornire al software una raccolta con le informazioni da utilizzare per modellare il transistor, ossia è necessario specificare tutti i parametri che possono essere sia geometrici, ma anche legati alle proprietà dei materiali.

Facendo diretto riferimento ai parametri presenti nell'equazione 1.1 per un transistor è necessario in primo luogo indicare la conducibilità intrinseca `Kp` $[A/V^2]$, la lunghezza `L` $[m]$ e la larghezza `W` $[m]$ del canale conduttivo. Altri parametri geometrici che possono essere utilizzati per migliorare l'analisi sono perimetro e area per il terminale di drain (parametri `PD` $[m]$ e `AD` $[m^2]$) e il terminale source (parametri `PS` e `AS`).

Come parametri funzionali per il calcolo della caratteristica statica dei MOSFET si menziona la tensione di soglia, modellata tramite il parametro `Vto` $[V]$. L'effetto body, dovuto alla differenza di tensione tra bulk e source, richiede invece di specificare il relativo coefficiente `Gamma` $\left[V^{0.5}\right]$ e il potenziale superficiale `Phi` $[V]$. Come ultimo parametro di un transistor si menziona il coefficiente di modulazione di lunghezza di canale `Lambda` $\left[V^{-1}\right]$.

		famiglia di transistor	
parametro	unità	n-MOS	p-MOS
`K`	$[\mu A/V^2]$	$\backsim 50$	$\backsim 20$
`W`	$[\mu m]$	$0.42 - 7$	$0.42 - 7$
`L`	$[\mu m]$	$0.15 - 8$	$0.15 - 8$

Tabella 1.1: *valori di riferimento per i principali parametri di simulazione per transistor.*

Un componente reale in condizioni statiche presenta delle perdite di corrente sia tra drain e source, sia tra gate e source, nel cosiddetto fenomeno della *current leakage* (perdita di corrente) associato alle correnti parassite. Analizzando invece il comportamento dinamico del circuito è possibile osservare che i MOSFET presentano un'inerzia alla trasmissione di carica (rispetto ad ogni coppia di terminali): tali effetti di *capacità parassite* possono essere modellate tramite l'inserimento nello schematico di capacità equivalenti.
Nella pratica le relazioni che determinano correnti e capacità parassite sono complesse (in quanto espressioni fortemente non lineari) e dipendenti da molti parametri dei transistor stessi: non esiste dunque un modello univoco che può essere utilizzato per la simulazione dei circuiti (ad un livello di complessità simil-realistico), ma in generale ogni produttore mette a disposizione dei progettisti i propri modelli spice che possono dunque essere inclusi negli schematici per effettuare delle simulazioni dalla valenza più reale e applicativa.

1.2 *Process Design Kit*: skywater

Il *Process Design Kit*, spesso abbreviato dall'acronimo *PDK*, è una suite di librerie e applicativi che permettono una progettazione corretta di un circuito integrato. In questi kit sono contenuti infatti tutti i modelli spice (sia dal modello lineare più semplice, sia a modelli del 4° ordine più complessi) che possono essere utilizzati per le simulazioni, oltre che a una serie di informazioni che vincolano la progettazione per permettere di ottenere un prodotto che sia effettivamente producibile e utilizzabile nel mondo reale. Per esempio, oltre a tutte le informazioni riguardanti ingombri fisici, i PDK contengono le proprietà per simulare con maggior precisione le correnti e capacità parassite che si generano nel prodotto finito in funzione della disposizione su chip dei transistor.

`skywater PDK` [2], come dice il nome stesso, è dunque un PDK rilasciato pubblicamente frutto della collaborazione di Google con la fondazione Skywater; questo progetto, per come riportato dal team di sviluppo del PDK stesso, è ancora in fase sperimentale e dunque può non essere perfettamente accurato, tuttavia si osserva che lo stesso progetto deriva direttamente da PDK utilizzati da anni a livello professionale.
L'idea alla base di questo progetto open source è quella di permettere a tutte le persone di progettare e prototipare circuiti integrati, permettendo la realizzazione pratica sfruttando il processo produttivo a $130nm$ fornito da `SkyWater Technology foundry` [7].

Sfruttando la suite di software composta da `XSchem`, `ngspice` e `skywater PDK` è possibile realizzare degli schematici e dei circuiti che si avvicinano il più possibile a dei circuiti reali per poter effettuare delle simulazioni.

Contenuti del PDK

La libreria skywater mette a disposizione sostanzialmente 2 categorie di modelli spice per la simulazione:

- le *primitive cells*, abbreviate PR, ossia i modelli associati ai mosfet (sia a 4 pin, sia a 3 pin con bulk collegato a massa), ma anche per altri componenti passivi che possono essere integrati su chip quali svariati tipi di resistenze (in funzione della potenza dissipabile), di capacità MIM (*metal-insulator-metal*), diodi e diodi varicap;

- la *digital standard cells*, abbreviate SC, sono invece già dei circuiti combinatori che sfruttano l'interconnessione delle celle primitive per realizzare porte logiche (and, or, not...) e altri circuiti combinatori (latch, multiplexer...).

Nella libreria sono presenti diverse varianti di transistori caratterizzati principalmente dalle differenze di tensioni ammissibili tra le coppie di terminali dei componenti (si trovano componenti che funzionano per tensioni V_{gs}, V_{ds} di valori $1.8V$, $3.3V$, $5.0V$, $20V$). Leggendo la documentazione [5] rilasciata dagli sviluppatori è inoltre possibile individuare i valori di lunghezza L e larghezza W ammissibili per la produzione di ogni tipo di mosfet.

La libreria delle standard cells derivanti dalle celle primitive sono divise in diverse famiglie caratterizzate dagli appellativi:

- *high density* (HD) e *high density low leakage* (HDLL), ossia porte logiche la cui caratteristica è di avere ingombri su chip più bassi (pari a $0.46 \times 2.72\mu m^2$) in modo da aumentare la densità di integrazione; la seconda tipologia, come si evince dal nome, è caratterizzata inoltra da una bassa dispersione di corrente elettrica. La tensione di alimentazione è posta a $1.8V$;

- le celle a bassa tensione (alimentazione $< 2.0V$) sono classificate in base alla velocità di commutazione dei gate secondo gli appellativi *low speed* (LS), *medium speed* (MS) e *high speed* (HS); in questa categoria è possibile rilevare anche le celle a basso consumo di potenza (categoria *low power* LP). L'ingombro su scheda di queste celle elementari è pari a $0.48 \times 3.33\mu m^2$;

- *high voltage* (HVL) sono invece delle celle con tensione di alimentazione pari a $5.0V$ con ingombro su scheda di $0.48 \times 4.07\mu m^2$.

Corner spice models La descrizione tramite un modello matematico astratto per predire il comportamento empirico di un componente analogico non sempre risulta essere accurato

per via dei parametri di influenza esterni che il calcolatore non può considerare (come correnti e capacità parassite che si instaurano inevitabilmente tra i componenti).

Per poter effettuare delle valutazioni più pratiche dei circuiti progettati, il pdk fornisce al progettista i cosiddetti modelli spice *corner* che sono tarati su particolari casi di funzionamento. In particolare, per convenzione è possibile individuare il comportamento dinamico tipico (*typical* T), veloce (*fast* F) e lento (*slow* S).
Importando, per esempio, in una simulazione il modello corner spice FS, il simulatore considererà come comportamento di funzionamento quello veloce per gli n-mos (rispetto ai quali i transitori risulteranno essere più veloci), mentre per i p-mos considererà un comportamento lento (i transitori risulteranno avere costanti di tempo più elevate).

Componenti utilizzati

Effettuata questa premessa sui contenuti del pdk skywater, nel proseguimento del seguente documento per la progettazione e simulazione dei circuiti si utilizzeranno modelli di transistor n-mos e p-mos con tensione nominale a 1.8V, in linea con le tensioni di alimentazione utilizzate nei microprocessori nei primi anni 2000, periodo a cui è possibile far risalire il processo produttivo di skywater.

Facendo riferimento al processore `Intel Pentium III Tualatin` [3] rilasciato sul mercato nel 2001, in quanto prodotto con un processo a $130nm$, è possibile stimare l'ingombro medio di un transistor ad un valore di circa $1.82\mu m^2$: tramite una ricerca per approssimazioni successive si determinano le dimensioni caratteristiche dei mosfet in modo da ricavare una caratteristica statica bilanciata ottenendo $W/L = 1/0.45\mu m/\mu m$ per gli n-mos e $W/L = 4.8/0.45\mu m/\mu m$ per i p-mos. Con tali dimensioni dei mosfet, l'ingombro medio di una coppia di transistor è pari a $2.61\mu m^2$.

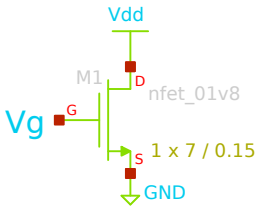

Figura 1.1: *schematico di un transistor n-mos con bulk a massa, drain posto alla tensione di alimentazione V_{dd}, gate posto alla tensione V_g e source a massa.*

Simulando il circuito in figura 1.1 è possibile ottenere la caratteristica statica (figura 1.2) di trasferimento che determina la corrente I_n che fluisce dal drain verso il source in funzione delle tensioni differenziali V_{ds} e V_{gs}. Tramite questa si può anche determinare la tensione di soglia V_{tn} del transistor che è pari a circa $0.6V$.

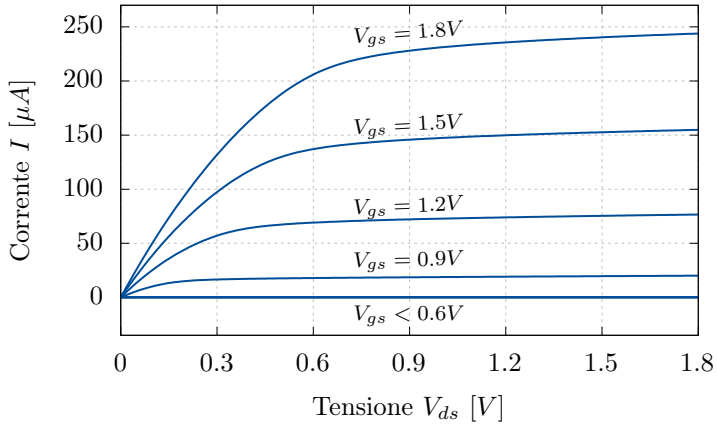

Figura 1.2: *caratteristica statica di uscita ottenuta mediante simulazione* `dc sweep` *del circuito in figura 1.1.*

Caratteristiche dinamiche Individuate le principali caratteristiche statiche, è possibile osservare dei comportamenti dinamici del circuito legati in particolare agli andamenti dei transitori.

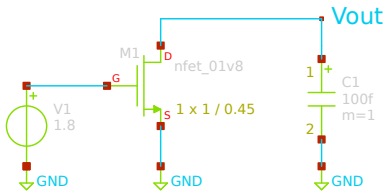

Figura 1.3: *circuito per simulare la scarica di un condensatore C_1 di capacità $0.75pF$, posto inizialmente ad una tensione $V_{out} = 1.8V$, mediante l'utilizzo di transistor n-mos.*

Il circuito più semplice da considerare a tale fine è quello riportato in figura 1.3 che si basa sullo scaricare una capacità di $0.75pF$ mediante l'utilizzo di transistori n-mos. In questo caso si pone al massimo la tensione $V_{gs} = 1.8V$ e si effettua una simulazione sul transitorio. Posto che al tempo iniziale la tensione in uscita V_{out} fosse pari allo stato alto $1.8V$, utilizzando i diversi modelli corner spice forniti è possibile effettuare dei diagrammi di rappresentazione dei transitori.

Facendo riferimenti ai risultati in figura 1.4, è possibile osservare che modelli corner diversi, rispetto al nome loro assegnato, producono dei risultati distinti. Ipotizzando di concludere il transitorio al 90% di escursione del segnale, ossia quando la tensione in uscita raggiunge il valore $V_{out} = 0.18V$, si ottengono i tempi per i 3 modelli pari a:

$$t_{slow} \approx 6.88ns \qquad t_{typical} \approx 6.03ns \qquad t_{fast} \approx 5.36ns$$

Ove non diversamente specificato nella prosecuzione del documento tutti comportamenti transitori verranno valutati rispetto ad un comportamento tipico dei transistor.

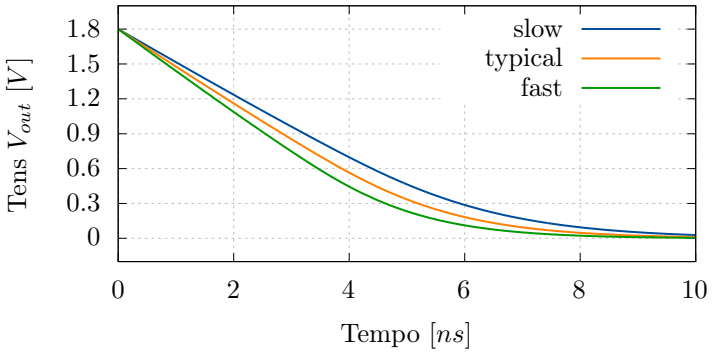

Figura 1.4: *evoluzione della tensione in uscita V_{out} dovuta alla scarica della capacità da $0.75pF$ mediante un mosfet (circuito in figura 1.3).*

Capitolo 2

Porte logiche in tecnologia c-MOS

I principali componenti attuali utilizzati nei dispositivi digitali sono realizzati mediante l'implementazione su chip di transistor opportunamente connessi. Come visto i mosfet sono degli oggetti che sono intrinsecamente analogici, tuttavia il loro principio di funzionamento li rende altamente adatti a realizzare funzioni digitali.

A livello digitale infatti i transistor possono essere considerati come degli interruttori che permettono o negano il passaggio di corrente tra i propri terminali. Considerando infatti la caratteristica statica dell'n-mos (figura 1.2, pagina 5) è possibile osservare che se si pone una tensione di gate V_g nulla (più in generale inferiore della tensione di soglia V_{tn}) il dispositivo non permette il passaggio di corrente ai suoi capi (indipendentemente dalla tensione differenziale V_{ds} applicata); usciti da questa fascia di interdizione è possibile osservare invece che, in funzione della tensione V_{gs}, è possibile avere un passaggio di corrente attraverso i terminali del mosfet.
Dualmente si dimostra che se la tensione V_g applicata al gate di un transitor p-mos è elevata (tale per cui la differenza $|V_{gs}|$ sia minore della tensione di soglia $|V_{tp}|$) allora il componente risulta interdetto e non permette il passaggio di corrente.

Questa peculiarità nel funzionamento duale dei componenti è particolarmente utile nelle implementazioni digitali dove in generale si considerano i segnali di tensione (intrinsecamente analogici) come dei segnali binari di valore basso (0) associato alla tensione di massa e valore alto (1) associato alla tensione di alimentazione V_{dd}.

Lo scopo di questo capitolo è dunque quello di osservare e rappresentare le porte logiche che compongono ogni circuito combinatorio di un dispositivo tecnologico.

2.1 Not gate

La porta logica più semplice da realizzare, composta da solamente due transistori, è il *gate not*, ossia l'invertitore logico che realizza la seguente tabella di verità:

input	output
0	1
1	0

L'implementazione circuitale di questa porta è mostrata in figura 2.1 ed è realizzata ponendo in serie un p-mos con un n-mos: l'ingresso V_{in} del segnale digitale viene applicato ad entrambi i gate dei transistor, mentre il segnale in uscita V_{out} viene rilevato nel collegamento tra i due drain dei mosfet.

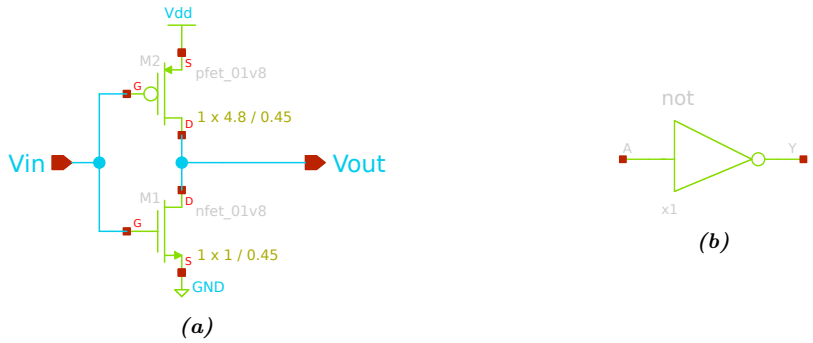

Figura 2.1: *implementazione di un invertitore logico in tecnologia c-mos (a) e relativa rappresentazione simbolica per circuiti logici (b).*

Per comprendere il funzionamento del sistema è sufficiente considerare la prima legge di Kirchhoff bilanciando la corrente al nodo rispetto alla quale si rileva il segnale in uscita V_{out} che si traduce nell'eguagliare le correnti generate dai due transistor:

$$I_n = I_p$$

A questo punto è possibile procedere con l'analisi del circuito per casi:

- nel caso in cui la tensione in ingresso sia bassa ($V_{in} = 0$) allora l'n-mos risulterebbe essere interdetto ($I_n = 0$), mentre il p-mos è posto in regime di saturazione. L'unica condizione che permette al p-mos di non far scorrere corrente attraverso i suoi terminali è quella per cui la tensione differenziale V_{ds} è nulla: questo porta ad affermare che

$$V_d = V_s \qquad \Rightarrow \qquad V_{out} = V_{dd}$$

- analogamente nel caso in cui l'ingresso si trovi ad una tensione in ingresso alta ($V_{in} = V_{dd}$), il p-mos risulterà interdetto, non permettendo il passaggio di alcuna corrente. Condizione necessaria affinché anche l'n-mos annulli la corrente attraverso i suoi terminali è che la tensione differenziale V_{ds} sia nulla, e dunque

$$V_{out} = 0$$

In figura 2.2 è possibile invece osservare la caratteristica statica del dispositivo ottenuta mediante una simulazione.

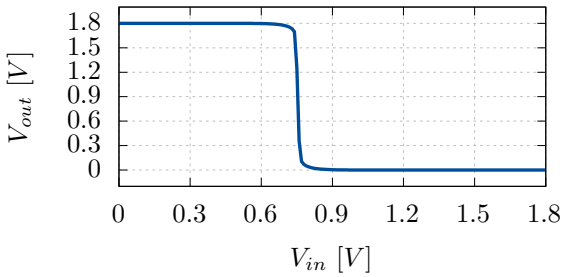

Figura 2.2: *funzione di trasferimento statica dell'invertitore logico di figura 2.1.*

L'implementazione delle porte logiche in tecnologia c-mos è caratterizzata da una forte immunità al rumore. Definendo infatti con V_{ilmax} la massima tensione per l'ingresso a cui è associata una corrispondenza con il segnale basso (ossia ogni tensione $V < V_{ilmax}$ è considerato un 0 binario) e con V_{ihmin} la minima tensione in ingresso riconosciuta come segnale alto, dall'analisi della caratteristica di trasferimento (figura 2.2) si osserva che i valori ad esso associati sono circa $V_{ilmax} = 0.6V$ e $V_{ihmin} = 0.9V$ rispetto ad una tensione di alimentazione $V_{dd} = 1.8V$. Questo significa che il segnale in uscita da una porta logica a valle può acquisire fino a $0.6V$ di rumore senza inficiare sul corretto funzionamento del circuito in quanto il segnale verrebbe ricevuto e considerato correttamente.

Caratteristiche dinamiche

Nota la caratteristica statica del circuito logico, di rilevante interesse pratico è l'analisi dinamica dello stesso, in quanto permetterà di stabilire a regime quale sarà la massima frequenza di commutazione della porta logica.

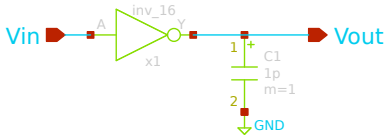

Figura 2.3: *schema circuitale di riferimento per l'analisi della risposta dinamica di un invertitore logico che deve pilotare un circuito a valle modellato da una capacità di* $0.75pF$.

A tale fine è necessario considerare un'invertitore, come in figura 2.3, che pilota un circuito a valle che può essere modellato, come indicato nella documentazione di skywater [6], mediante una capacità di carico C_{load} collegata all'uscita di valore $0.75pF$.

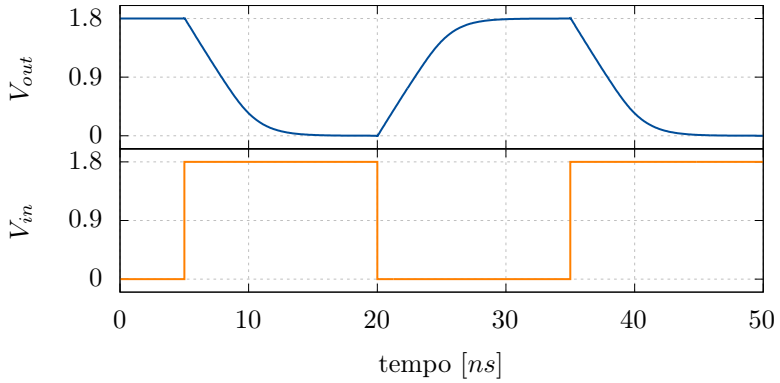

Figura 2.4: *risposta dell'invertitore logico (in figura 2.3) ad un'onda quadra in ingresso di periodo* $30ns$ *e duty cycle del* 50%. *Le tensioni sono espresse in volt.*

In figura 2.4 è possibile leggere la risposta dell'invertitore ad un'onda quadra in ingresso. Data la simmetricità di comportamento scelta grazie ai rapporti W/L diversi dei transistor p-mos ed n-mos è possibile osservare che i transistori di salita e discesa sono molto simili tra loro.

Rispetto al valore della capacità scelto di $0.75pF$ (considerato rappresentativo di un generico circuito a valle) è possibile valutare i ritardi di propagazione dei segnali, ossia i

tempi che intercorrono tra le commutazioni dell'ingresso e la rispettiva variazione d'uscita. Tali parametri possono essere valutati singolarmente sia per una variazione dell'ingresso da alto a basso τ_{hl}, ma anche per lo stesso che passa da basso a alto τ_{lh} (in questo caso i tempi sono calcolati al 95% dell'escursione di tensione):

$$\tau_{lh} = 7.21ns \qquad \tau_{hl} = 7.14ns$$

2.2 Nor gate

La porta logica *nor*, coincidente con la negazione del gate *or*, è un gate che, come il *nand*, costituisce un *operatore universale*, ossia in grado di realizzare tramite delle opportune interconnessioni tutte le funzioni logiche digitali. Tale porta a due (o più ingressi) rispetta la seguente tabella di verità:

$V_{in,1}$	$V_{in,2}$	V_{out}
0	0	1
0	1	0
1	0	0
1	1	0

Si osserva dunque che tale porta logica determina un'uscita alta solamente se tutti i suoi ingressi sono bassi, mentre in tutti gli altri casi l'uscita è bassa.

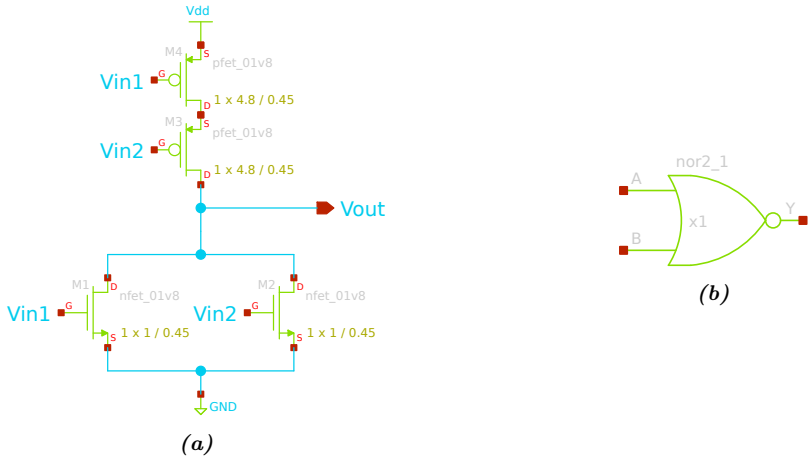

Figura 2.5: *implementazione della porta logica nor in tecnologia c-mos (a) e la relativa rappresentazione simbolica (b).*

In figura 2.5 è dunque possibile osservare un'implementazione della porta logica nor in tecnologia c-mos.
Tale circuito può essere descritto analizzando le possibili combinazioni di ingresso presenti nella tabella di verità:

- nel caso in cui entrambi gli ingressi si trovino ad un valore logico basso (prima riga della tabella di verità), allora risulterebbero interdetti i transistor a substrato n, mentre la rete di pull-up composta dai due p-mos in serie risulterebbe essere attiva. L'unico modo per garantire corrente nulla in uscita dal circuito è quello di avere tensione differenziale V_{ds} dei p-mos nulla, ossia nel caso in cui $V_{out} = V_{dd}$, verificando la tabella di verità;

- in tutti gli altri casi in cui almeno un segnale si trova in uno stato di tensione alto si osserva che la rete di pull-up sarà sicuramente interdetta (il p-mos associato all'ingresso alto non permetterebbe infatti passaggio di corrente) e dunque la tensione in uscita sarà determinata dalla rete di pull-down degli n-mos che risulteranno attivi. Sempre per imposizione della condizione di corrente nulla al nodo d'uscita si ottiene che la tensione differenziale V_{ds} degli n-mos deve essere nulla e dunque $V_{out} = 0$.

Tale comportamento può essere verificato mediante una simulazione del transitorio (il cui risultato è mostrato in figura 2.6), imponendo come ingressi delle onde quadre di periodo multiplo per analizzare le risposte del circuito alle varie combinazioni di ingressi e determinare così le principali caratteristiche dinamiche.

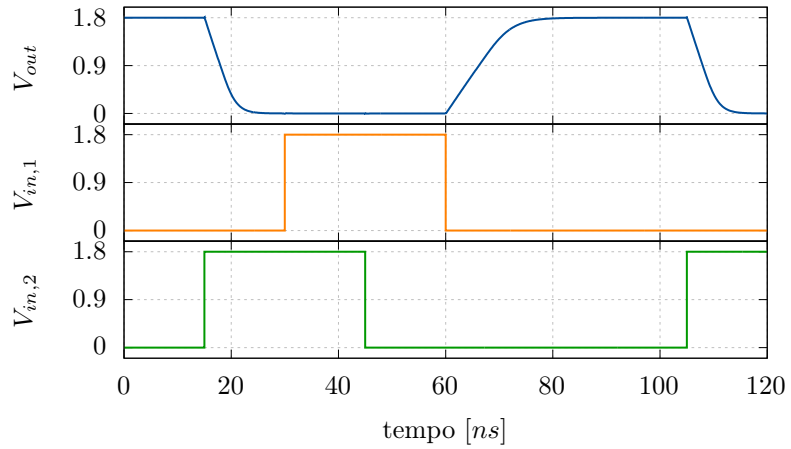

Figura 2.6: *risposta del gate nor a due ingressi di onde quadre di periodi multipli. Come nel caso dell'invertitore (fig. 2.3) il circuito a valle della porta logica è modellato da una capacità di carico di* $0.75pF$*. Le tensioni sono espresse in volt.*

Calcolando i tempi di propagazione dei segnali per un'escursione del 95% della tensione di alimentazione, è possibile osservare che il tempo di transizione del segnale dal valore logico alto a quello basso è minore rispetto alla transizione inversa ($7.22ns$ vs $14.73ns$): tale effetto è giustificabile considerando il fatto che la rete di pulldown composta dal parallelo di n-mos permette di generare una quantità di corrente maggiore rispetto alla rete di pull-up determinata dalla serie di p-mos. Un modo dunque per correggere questo sbilanciamento sarebbe di rideterminare dei nuovi parametri W/L per i transistor in modo da bilanciare il comportamento.

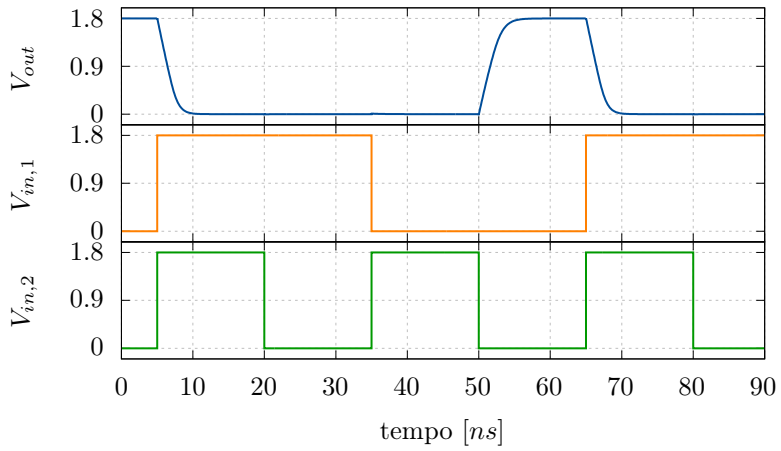

Figura 2.7: *risposta del circuito nor ottenuta aumentando il rapporto W/L dei p-mos al valore $4.8/0.15\mu m/\mu m$; i mosfet a substrato n rimangono invarianti con un rapporto $1/0.45\mu m/\mu m$. Le tensioni sono espresse in volt.*

Diminuendo la larghezza W dei transistor p-mos al valore $0.15\mu m$ il comportamento del circuito tende a bilanciarsi (come si può osservare in figura 2.7), con tempi di transizione da uscita alta a bassa di $3.61ns$ e da uscita bassa ad alta di $4.31ns$.

2.3 Nand gate

La porta logica *nand*, coincidente con la negazione del gate *and*, è il gate che, per costruzione, risulta duale alla porta nor appena studiata ed è caratterizzata dalla tabella di verità

$V_{in,1}$	$V_{in,2}$	V_{out}
0	0	1
0	1	1
1	0	1
1	1	0

La realizzazione circuitale della porta logica, come in figura 2.8, è molto simile al gate nor, dove tuttavia la rete di pull-up è realizzata da un parallelo (e non da una serie) di p-mos, mentre la rete di pull-down è realizzata da una serie (e non da un parallelo) di transistor n-mos. Avendo osservato l'asimmetria di comportamento dovuta al collegamento in serie/parallelo dei mosfet vista nel gate nor, si procede sin da principio a diminuire la larghezza W degli n-mos al valore di $0.15\mu m$.

Il funzionamento del circuito può essere analizzato studiando le possibili combinazioni dei segnali in ingresso:

- nel caso in cui entrambi i segnali si trovino ad un valore logico alto (quarta riga della tabella di verità) la rete di pull-up risulterebbe interdetta (entrambi i p-mos sono spenti per via della tensione V_{gs} nulla), mentre la rete di pull-down a transistor accesi permette un passaggio di corrente. Dovendo la stessa essere nulla per il bilancio della corrente al nodo d'uscita, si osserva necessariamente che $V_{out} = 0$;

- in tutti gli altri casi in cui almeno un ingresso sia allo stato logico basso, allora sicuramente uno degli n-mos sarebbe interdetto (e dunque di conseguenza lo è anche la

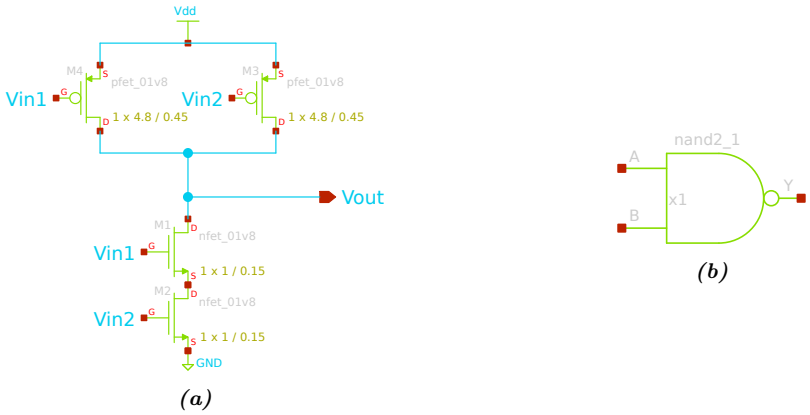

Figura 2.8: *implementazione della porta logica nand in tecnologia c-mos (a) e la relativa rappresentazione schematica (b).*

rete di pull-down), mentre almeno uno dei p-mos rimarrebbe accesso, permettendo il passaggio di corrente nella rete di pull-up. Il bilancio della corrente al nodo d'uscita permette infine di stabilire che il relativo stato logico è alto ($V_{out} = V_{dd}$).

Mediante una simulazione del transitorio nel tempo (con capacità di carico di $0.75pF$) è dunque possibile verificare il comportamento logico del circuito, arrivando ai risultati mostrati in figura 2.9.

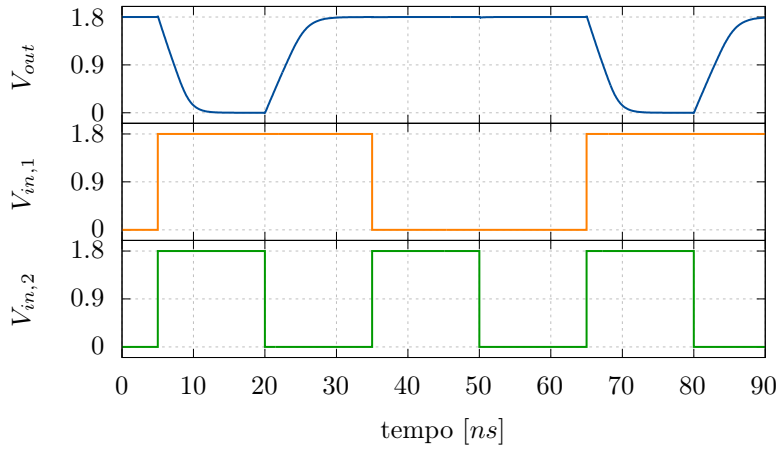

Figura 2.9: *risposta del gate nand a due ingressi di onde quadre di periodi multipli. Come nel caso dell'invertitore (fig. 2.3) il circuito a valle della porta logica è modellato da una capacità di carico di* $0.75pF$*. Le tensioni sono espresse in volt.*

A questo punto è possibile misurare il tempo di commutazione dell'uscita nel passaggio da tensione alta a bassa (misurata al 95% dell'escursione del segnale), che si attesta al valore di $5.51ns$, mentre il passaggio del segnale da basso ad alto dura $7.14ns$.

2.4 And & or gate

Avendo studiato singolarmente i gate nand, nor e not, la costruzione dei gate logici *and* e *or* deriva direttamente tramite la connessione in sequenza della rispettiva porta negata con l'invertitore logico. In particolare le tabelle di verità che i due componenti devono realizzare sono:

$V_{in,1}$	$V_{in,2}$	$V_{out,and}$	$V_{out,or}$
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

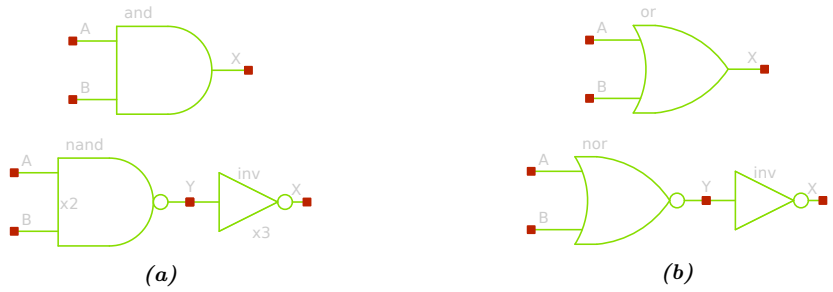

Figura 2.10: *rappresentazione simbolica (sopra) e blocchi elementari che costituiscono (sotto) i gate logici and (a) e or (b).*

In figura 2.10 si osserva la rappresentazione simbolica delle porte logiche che espandendone le implementazioni in tecnologia c-mos determinano gli schematici riportati in figura 2.11.

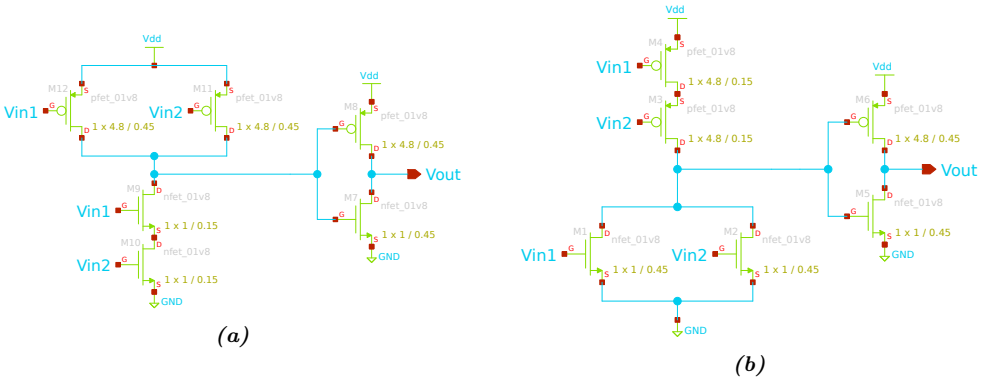

Figura 2.11: *implementazione in tecnlogia c-mos della porta logica and (a) e or (b).*

E' inoltre possibile procedere con le simulazioni dei transitori (con una capacità di carico di $0.75pF$) che determinano gli andamenti temporali mostrati in figura 2.12. Analizzando i tempi di propagazione dei segnali all'interno dei circuiti si arriva a dei tempi massimi per i due circuiti che si attestano ai valori

$$\tau_{max,and} \approx 7.31ns \qquad \tau_{max,or} \approx 7.21ns$$

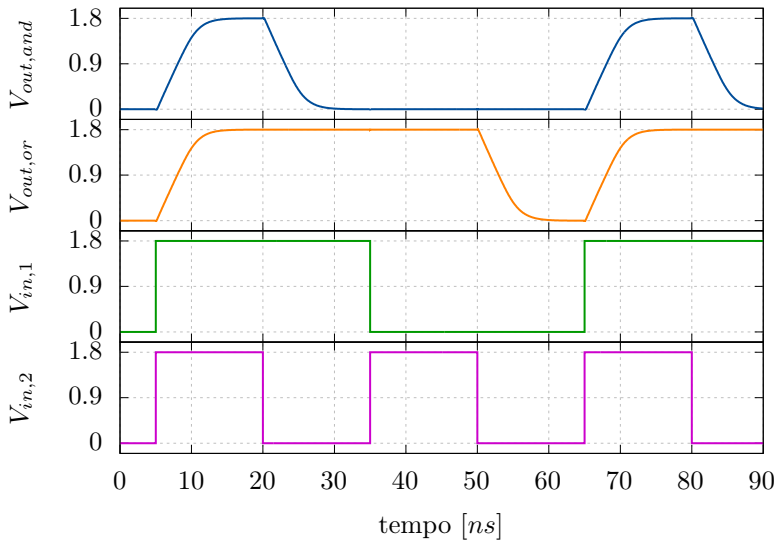

Figura 2.12: *risposta dei gate and e or a ingressi di onde quadre a periodi multipli. Come nel caso dell'invertitore (fig. 2.3) il circuito a valle della porta logica è modellato da una capacità di carico di* $0.75pF$*. Le tensioni sono espresse in volt.*

Xor gate

Un'altra porta logica che verrà utilizzato nello sviluppo di alcuni circuiti combinatori è il gate logico *xor*, ossia l'or esclusivo, per il quale l'uscita è alta solamente se uno degli ingressi è alto, la cui tabella di verità è

$V_{in,1}$	$V_{in,2}$	V_{out}
0	0	0
0	1	1
1	0	1
1	1	0

In figura 2.13 è mostrata la rappresentazione schematica del gate logico xor mediante l'interconnessione di porte logiche precedentemente studiate.

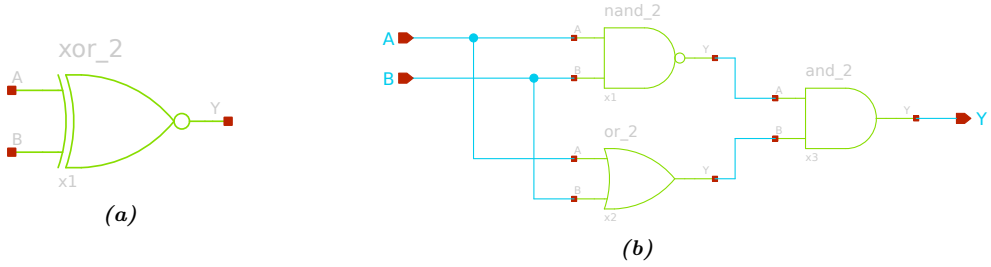

Figura 2.13: *schema rappresentativo del gate xor (a) e relativa costruzione mediante l'utilizzo di altre porte logiche (b).*

2.5 Prestazione dei gate logici

Avendo implementato in tecnologia c-mos i vari gate logici e avendone valutato le prestazioni, è possibile iniziare a teorizzare le velocità, espressa come frequenza di propagazione dei segnali in ingresso, dei circuiti logici che essi potranno costituire.

Considerando la capacità di carico di $0.75pF$ per tutti i circuiti utilizzati, si osserva che il transitorio di carica/scarica maggiore è associato alla porta logica and con un periodo τ pari a $7.31ns$: questo permette di stabilire la massima frequenza f cui possono operare i circuiti senza perdere la corretta funzionalità del circuito che vale

$$f = \frac{1}{\tau} \approx 136MHz$$

Questo valore di riferimento tuttavia non può essere identificativo della velocità cui può funzionare un circuito logico più complesso, in quanto l'interconnessione di più porte logiche può aumentare la capacità equivalente di carico e dunque il tempo di propagazione del segnale all'interno del circuito.

Per migliorare la stabilità nel funzionamento è possibile supporre che all'interno di un ciclo di clock il tempo legato al transitorio sia il 10% del periodo complessivo: questo dunque riduce la frequenza di operatività del circuito al valore

$$f \approx 13.6MHz$$

Questo valore di sicurezza permette così di stabilire una frequenza di operatività del circuito che assicuri una corretta trasmissione dei dati all'interno delle varie sezioni circuitali che compongono un microprocessore.

Capitolo 3

Circuiti logici asincroni e reti combinatorie

Avendo studiato le porte logiche in tecnologia c-mos è possibile iniziare a progettare i primi circuiti logici con funzione di memoria che compongono ogni dispositivo di elaborazione digitale, quali i vari tipi di *latch* che permettono di comporre delle particolari strutture di registri quali il contatore binario, ma anche dei circuiti a valenza logica per operazioni binarie, quali il circuito sommatore e moltiplicatore.

3.1 SR latch

Il *latch set-reset*, abbreviato generalmente come *latch SR*, è un circuito logico asincrono dalla bassa complessità che permette di memorizzare delle informazioni binarie. Tale circuito, come mostrato in figura 3.1, è realizzato mediante un'opportuna retroazione di porte logiche nor e la funzione realizzata dal circuito viene descritta dal nome stesso del dispositivo.

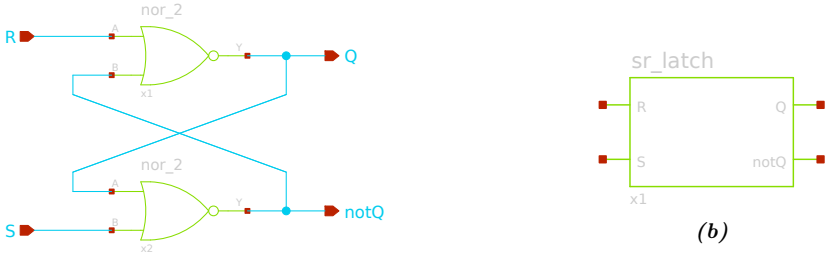

Figura 3.1: *implementazione circuitale di un latch SR (a) e relativa rappresentazione schematica semplificata (b).*

Costruendo la tabella 3.1 di verità di questo tipo di latch è possibile descrivere l'uscita, generalmente indicata con Q, del dispositivo: nel caso venga imposto un segnale alto al terminale di S allora anche l'uscita viene impostata (funzione di *set*) alla tensione di alimentazione, mentre se viene imposto un segnale alto al terminale R l'uscita viene ripristinata (funzione di *reset*) alla tensione di massa.
Nel caso in cui nessuno dei due terminali d'ingresso si trovi allo stato alto, ossia non sia

specificata nessuna tra le azioni di set o reset, allora la retroazione garantisce un'uscita che rimane invariata allo stato precedente del sistema. Condizione di ingresso invalida per il sistema è invece quella per cui entrambi gli ingressi S ed R si trovano ad uno stato logico alto: in questo caso sia l'uscita Q che il suo negato $\overline{Q}$ risultano poste allo stato logico basso (non verificando la disuguaglianza $Q \neq \overline{Q}$). Partendo dal presupposto che tale condizione a livello pratico non dovrebbe mai avvenire, in quanto non ha senso impostare e ripristinare lo stesso segnale contemporaneamente, il circuito ritorna ad uno stato funzionale "stabile" e di corretto funzionamento non appena uno dei segnali viene riportato ad un valore logico basso.

S	R	Q	$\overline{Q}$	
0	0	$\backsim$	$\backsim$	
0	1	0	1	
1	0	1	0	
1	1	0	0	stato invalido

Tabella 3.1: *tabella di verità del latch SR. Con $\backsim$ si indica che lo stato del segnale rimane invariato rispetto al valore che aveva precedentemente acquisito.*

Il principio di funzionamento del latch SR è dovuto alla natura della porta logica nor che lo compone, e si basa sul fatto che l'uscita del circuito è alta solamente se entrambi gli ingressi si trovano ad uno stato basso. Il collegamento in retroazione che determina l'ingresso di un gate con l'uscita dell'altro permette di espletare la funzione per cui solamente una delle due uscite può essere alta contemporaneamente nel caso in cui gli ingressi siano ad uno stato logico basso. Forzando invece un ingresso (per esempio S) allo stato logico alto, allora si impone che l'uscita del gate associato (in questo caso $\overline{Q}$) venga posta a 0: il gate complementare (associato all'ingresso R) osserverà così entrambi gli ingressi bassi ponendo l'uscita Q allo stato logico alto.
A questo punto anche se si rimuovesse l'ingresso S alto, lo stato del circuito rimarrebbe invariato in quanto il gate associato a tale terminale risulterebbe attivato dalla retroazione dovuta al segnale Q che è alto. Si può analizzare la commutazione del circuito nel caso di segnale di reset alto in modo speculare.

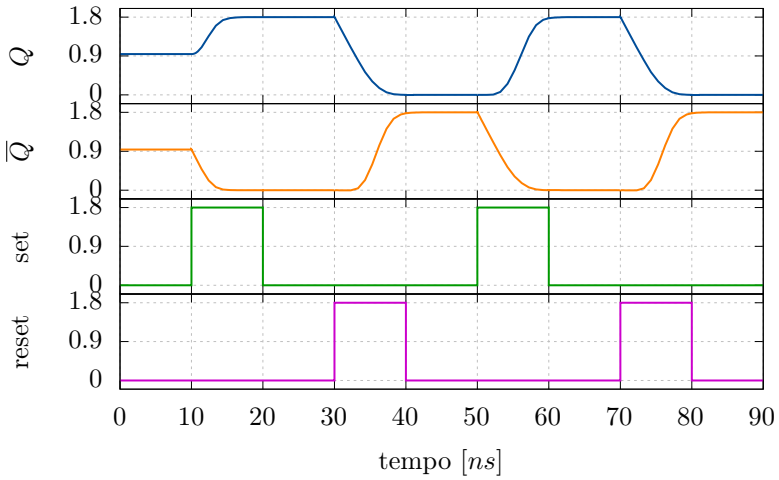

Figura 3.2: *rappresentazione dell'andamento del tempo del segnale d'uscita Q e del suo negato $\overline{Q}$ in funzione degli impulsi di set e di reset. Le uscite sono state collegate a massa con una capacità di carico di $0.75pF$ rappresentativa del circuito a valle del latch SR.*

In figura 3.2 è rappresentato l'andamento nel tempo delle uscite del circuito in funzione di impulsi in ingresso ai segnali di set e reset. Si può osservare che fino al primo impulso le uscite non sono ben definite, e questo è dovuto alla non inizializzazione del circuito. Nel caso ideale ottenuto mediante simulazione le uscite si trovano ad una tensione $V_{dd}/2$, tuttavia nel caso reale un'uscita prevarrebbe sull'altra per stato logico per via delle imperfezioni di realizzazione dei transistor (per esempio i rapporti W/L o la conducibilità intrinseca potrebbero differire leggermente tra i transistor che compongono i gate, portando ad uno sbilanciamento del circuito), determinando un'uscita iniziale generalmente ignota.

Va inoltre osservato che il segnale alto in ingresso deve rimanere in tale stato per un tempo necessario a garantire la rispettiva commutazione dell'uscita, pena la non corretta funzionalità del circuito stesso. Considerando la capacità di carico di $0.75pF$ il valore dell'impulso alto deve essere al minimo pari a $t_{min} = 7.28ns$.

SR latch con segnale di *enable*

Spesso i circuiti digitali sono predisposti con dei cosiddetti segnali di *enable*, ossia di abilitazione degli ingressi. Questo significa che è possibile specificare al dispositivo quando lo stesso può leggere i segnali in ingresso oppure ignorarli.

Sfruttando la convenzione per la quale il segnale di abilitazione è attivo quando il rispettivo stato logico è alto, è possibile creare un latch SR con funzione di enable anteponendo agli ingressi S, R dei gate nor delle porte nand, come in figura 3.3.

In questo caso gli ingressi S, R possono essere processati dal latch solamente se il segnale di enable è alto; nel caso in cui il segnale di abilitazione sia basso, allora gli ingressi che giungerebbero al latch sarebbero entrambi 0, con conseguente invarianza dell'uscita.

In figura 3.4 è possibile osservare come cambia la risposta di questa variante del latch SR, mantenendo inalterati i segnali di set e reset in ingresso.

Si osserva così che solo i primi impulsi di set e reset determinano una variazione dell'uscita del latch, in quanto sono gli unici segnali all'interno della fascia temporale del segnale di enable allo stato alto. Al tempo $t = 65ns$ il segnale di abilitazione scende al valore logico

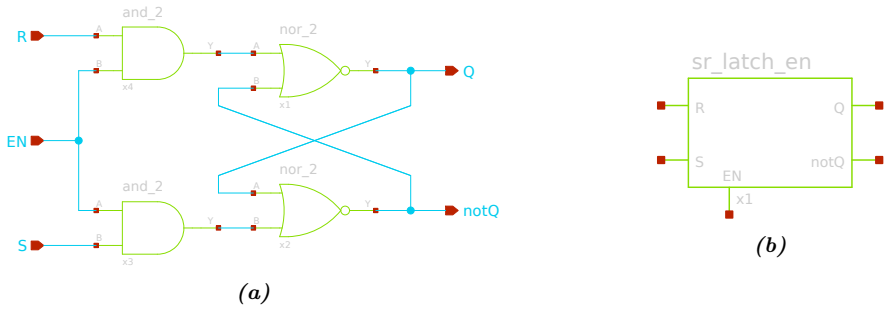

Figura 3.3: *implementazione circuitale di un latch SR con ingresso di enable (a) e relativa rappresentazione schematica semplificata (b).*

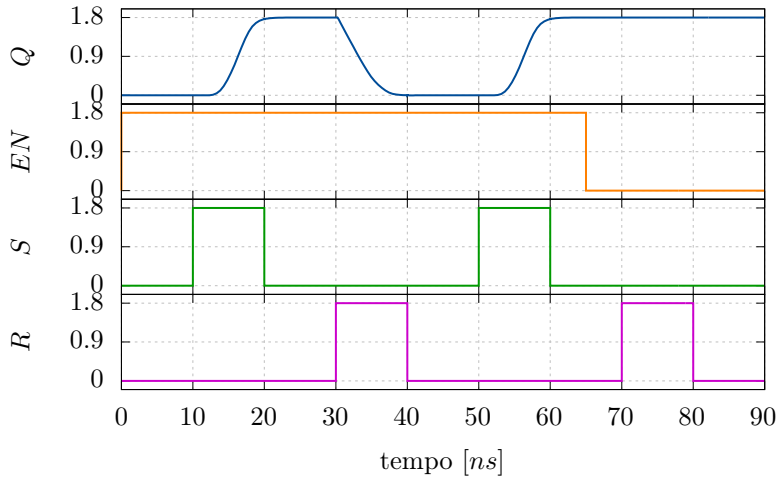

Figura 3.4: *rappresentazione dell'andamento del tempo del segnale d'uscita Q in funzione degli impulsi di set e reset, considerando inoltre il segnale di abilitazione EN. Le uscite del latch state collegate a massa con una capacità di carico di $0.75V$ rappresentativa del circuito a valle. Tutti i valori lungo l'asse delle ordinate sono da ritenersi in volt.*

basso, e dunque ignora il successivo segnale di reset (che pertanto non influenza l'uscita del latch).

SR flip-flop

Una caratteristica fondamentale che caratterizza tutti i circuiti digitali per migliorarne l'efficienza e la stabilità è il sincronismo: in ogni circuito integrato infatti è possibile trovare un generatore di sincronismo che invia (ad una frequenza prestabilita) i segnali di *clock*. Questi elementi, realizzati solitamente tramite circuiti astabili, generano di fatto delle onde quadre di periodo costante e pre-determinato (individuato dalla risonanza di elementi ceramici o dalle costanti di tempo di filtri RC) del quale è anche possibile stabilire il duty cycle, ossia la percentuale di tempo del segnale alto rispetto al periodo complessivo.

La transizione del segnale da basso ad alto (o viceversa) determina il sincronismo tra tutti i gli elementi collegati al circuito stesso, determinando con cadenza regolare quando procedere con le operazioni successive.

Analizzando il latch SR con enable descritto in precedenza, l'elemento di sincronismo può essere introdotto proprio nel segnale di abilitazione del circuito digitale stesso. Un problema che si osserva è che tuttavia questo elemento di memoria, allo stato attuale, commuta l'uscita in funzione degli impulsi di set-reset che avvengono fintanto che il segnale di clock risulta essere alto.

Si introduce dunque la necessità di individuare dei circuiti *edge detector* che permettono di rilevare le transizioni del segnale di clock e alle quali si associano degli impulsi (come delle onde quadre) con tempo in stato logico alto di durata molto bassa. Sfruttando questi circuiti è possibile garantire la lettura agli ingressi S ed R del latch solamente nell'intervallo di tempo in cui si ha la commutazione del segnale. Il circuito edge detector deve inoltre garantire un tempo di segnale alto in uscita che sia sufficiente per gli ingressi a pilotare il circuito a valle.

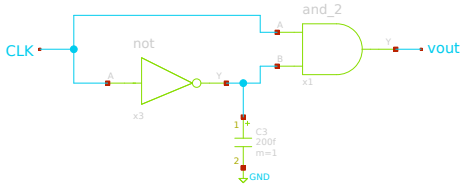

Figura 3.5: *implementazione circuitale di un edge detector che rileva la commutazione dell'ingresso da basso a alto.*

Un circuito che si può realizzare semplicemente mediante l'interconnessione di gate logici è l'edge detector mostrato in figura 3.5, composto da un invertitore logico e da una porta and.
Nonostante ci si aspetta che l'uscita di tale circuito sia sempre bassa (in quanto il prodotto logico tra un segnale e il suo negato è sempre basso), la presenza delle capacità nei gate logici (e in questo caso è aggiunta anche una capacità supplementare di $1.5pF$) determinano un ritardo nella commutazione dell'invertitore. Considerando il fronte di salita del segnale di clock, gli ingressi al gate and saranno entrambi alti in quanto l'invertitore necessiterà di una determinata quantità di tempo per invertire la tensione alla sua uscita, dovendo anche scaricare la relativa capacità.

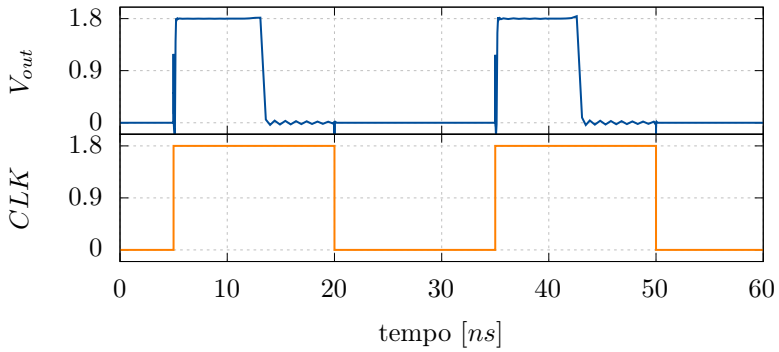

Figura 3.6: *risposta del circuito di edge detection ad un segnale di clock di periodo* $30ns$ *e duty cycle del* 50%.

Come si può osservare dalla figura 3.6 questo circuito così realizzato permette di rilevare i fronti di salita del segnale di clock generano un relativo segnale alto per una durata di circa $8.9ns$, ossia il tempo generalmente necessario ai gate logici per effettuare delle eventuali

commutazioni di segnali. Variando la capacità del circuito è possibile aumentare o diminuire il periodo di pulsazione dovuta al fronte di salita.

Avendo dunque determinato un circuito di edge detection è possibile implementare il *flip-flop SR* (figura 3.7), ossia un latch SR che sfrutta il circuito appena realizzato per commutare i suoi ingressi solamente in corrispondenza dei fronti di salita del segnale di clock.

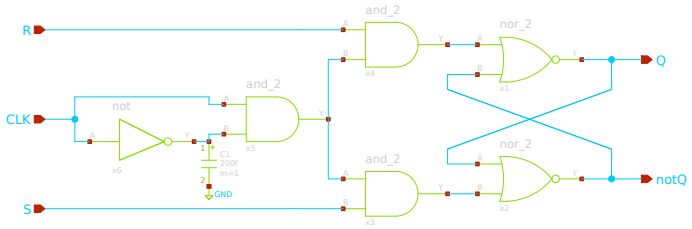

Figura 3.7: *implementazione circuitale di un flip-flop SR.*

Effettuando delle simulazioni sui transitori (figura 3.8) è possibile infatti verificare il corretto funzionamento, come previsto, del circuito: il flip-flop commuta opportunamente la sua uscita in funzione dei segnali di set/reset solamente in presenza di un fronte positivo del segnale di clock.

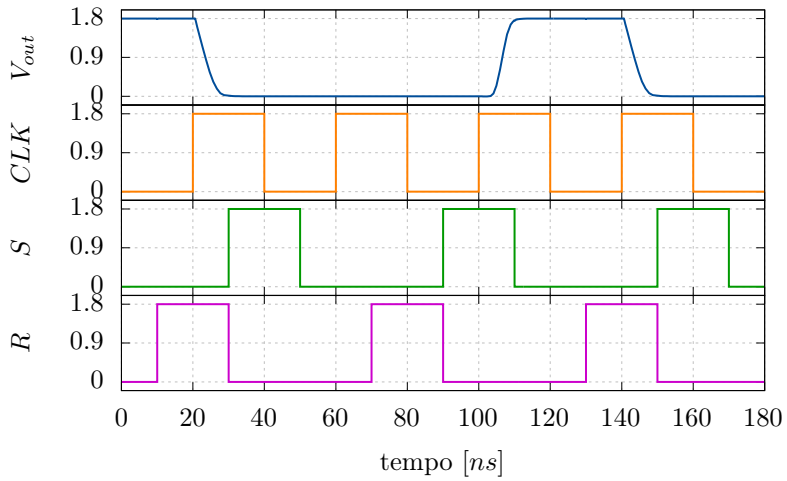

Figura 3.8: *risposta dell'uscita di un flip-flop SR. Il circuito a valle è modellato da una capacità di* $0.75pF$*. Le tensioni sono espresse in volt.*

3.2 JK flip-flop

Un problema del latch SR e dei circuiti da esso derivati è l'invalidità dello stato con entrambi gli ingressi alti (come si osserva nell'ultima riga della tabella di verità 3.1 a pagina 18). Questo inconveniente rende il circuito instabile rispetto a ingressi contemporaneamente alti (in quanto non è possibile stabilire in maniera deterministica il funzionamento del circuito stesso nel momento in cui verranno tolti gli ingressi) e limita l'applicabilità di questo sistema di memoria.

Per risolvere questo problema è stato inventato il *flip-flop JK* i cui ingressi sono comunemente indicati dalle lettere J (associato alla funzione set) e K (associato alla funzione di reset). Come indicato nella tabella di verità 3.2, il comportamento generale del circuito rimane invariato se non nel caso di presenza di due ingressi alti contemporaneamente, momento nel quale le uscite Q e $\overline{Q}$ commutano rispettivamente i loro stati da alto a basso e viceversa.

S	R	Q	$\overline{Q}$
0	0	$\backsim$	$\backsim$
0	1	0	1
1	0	1	0
1	1	commutazione	

Tabella 3.2: *tabella di verità dei latch e flip-flop JK. Con $\backsim$ si indica che lo stato del segnale rimane invariato rispetto al valore che aveva precedentemente acquisito.*

Tale effetto viene ottenuto retroazionando l'uscita dei gate nor con un terzo ingresso agli and di abilitazione del segnale (oltre che alla retroazione al gate nor stesso), come si può osservare in figura 3.9

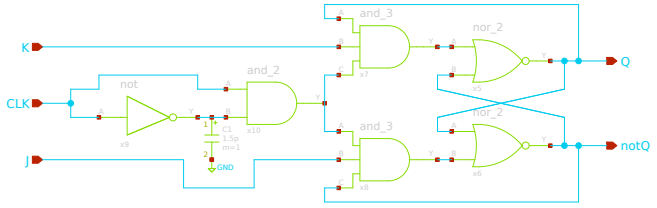

Figura 3.9: *implementazione circuitale di un flip-flop JK. Essendo il circuito un flip-flop, il segnale di clock viene opportunamente filtrato dall'edge detector.*

Tale realizzazione permette infatti di eliminare la possibilità di tensione alta agli ingressi di set e reset del latch SR che lo compone, ma solamente l'azione complementare all'uscita può essere realizzata. Per esempio se l'uscita si trovasse allo stato logico alto, solamente l'ingresso di reset potrà essere potrà essere attivato per commutare l'uscita.

Simulando il circuito nel dominio del tempo è possibile verificare il funzionamento del flip-flop, osservando che l'implementazione (per come attualmente realizzata) presenta un problema critico nella commutazione.

Rispetto ai risultati mostrati in figura 3.10 per alcune commutazioni del segnale in uscita il transitorio non segue una funzione monotona, ma tende a oscillare per stabilizzarsi ad uno stato logico alto o basso (non necessariamente corretto rispetto all'ingresso del circuito) solamente quando il circuito di edge detection termina il suo segnale di abilitazione alto.

Questo fenomeno prende il nome di *gate racing* ed dovuto al fatto che i segnali in uscita Q e $\overline{Q}$ sono retroazionati alla porta and di abilitazione dei segnali in ingresso: questo porta ad un rincorrersi delle uscite retroazionate che determinano l'instabilità critica del circuito per come così realizzato.

JK flip-flop master-slave

Per eliminare il problema di funzionamento critico del gate racing del flip-flop JK è possibile costruire una sua variante che prende il nome di *master-slave* che è sostanzialmente composta

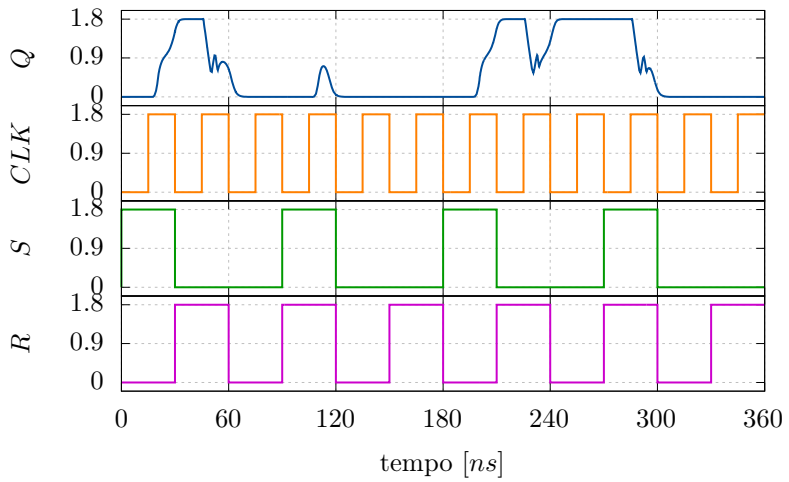

Figura 3.10: *risposta di un flip-flop JK; il circuito a valle è modellato da una capacità di* $0.75pF$*. Le tensioni sono espresse in volt.*

da due flip-flop in sequenza. Come risulta evidente dell'implementazione circuitale (figura 3.11) il primo latch è detto master per via della sua funzione di interfacciamento con gli ingressi esterni, mentre il secondo (detto slave) è pilotato direttamente dal master.

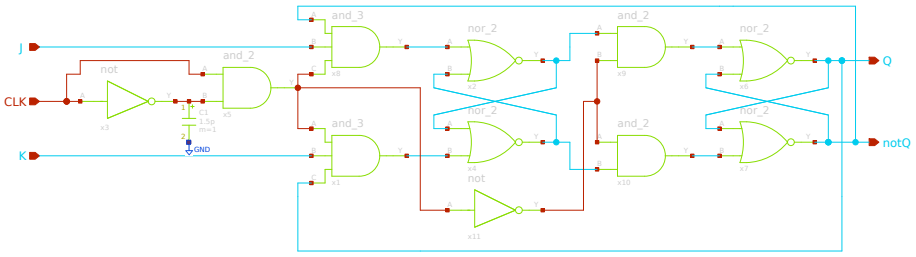

Figura 3.11: *implementazione circuitale di un flip-flop JK master-slave. Le connessioni in rosso evidenziano il circuito di clock.*

Il fenomeno del gate racing viene eliminato mediante l'implementazione di un circuito di abilitazione particolare per i due latch che compongono il circuito; infatti il segnale di clock (filtrato dal circuito di edge detection) viene utilizzato per abilitare il master del flip-flop, mentre per pilotare lo slave è necessario considerare il segnale negato. Questo significa che, allo stesso tempo, solo uno dei due latch può commutare (in quanto se un latch presenta uno stato di abilitazione alto, l'altro necessariamente lo osserverà basso).

Questo introduce necessariamente un ritardo di commutazione dell'uscita, dipendente principalmente dal fronte alto generato dal circuito di edge detection in quanto tale impulso viene utilizzato per pilotare la prima fase del flip-flop; successivamente viene disabilitato il master e abilitato lo slave che quindi commuta l'uscita Q e $\overline{Q}$ in funzione degli ingressi di set J e reset K applicati al circuito.

Mediante questa implementazione circuitale, il cui risultato della simulazione è mostrato in figura 3.12, è possibile osservare come il fenomeno del gate racing è sparito e si verifica la perfetta funzionalità del circuito, a discapito di un piccolo ritardo di propagazione del segnale in ingresso.

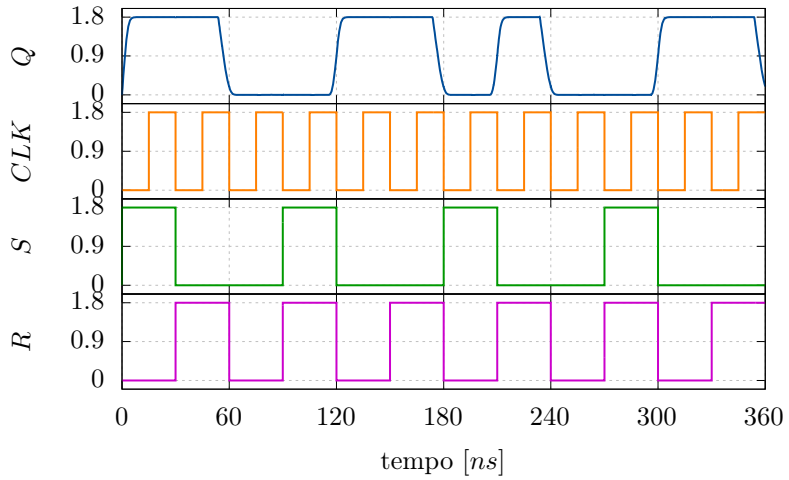

Figura 3.12: *risposta di un flip-flop JK in configurazione master-slave; il circuito a valle è modellato da una capacità di* $0.75pF$*. Le tensioni sono espresse in volt.*

3.3 Contatore binario

Dal punto di vista applicativo di fondamentale importanza è costruire un dispositivo in grado di contare il numero di impulsi trascorsi in un determinato intervallo di tempo. Questo tipo di componente viene infatti spesso utilizzato in convertitori di segnale da analogico a digitale ad elevata risoluzione in bit, ma anche nel registro della memoria cache denominato *program counter*, il cui ruolo è quello di individuare le istruzioni da far eseguire al calcolatore.

Tale circuito può essere realizzato mediante un opportuno collegamento di flip-flop JK master-slave, sfruttando la sua funzionalità di *divisore di clock binario*. Collegando infatti entrambi i segnali di ingresso J e K al valore logico alto, allora si forza il circuito a lavorare in un continuo stato di commutazione dell'uscita ad ogni fronte di salita del segnale di clock. Ad ogni periodo di clock corrisponde dunque un solo stato alto o basso dell'uscita del flip-flop: considerando invece un insieme di due periodi di clock tale effetto determina un'onda quadra di periodo doppio rispetto ai fronti di salita in ingresso, come è possibile osservare in figura 3.13.

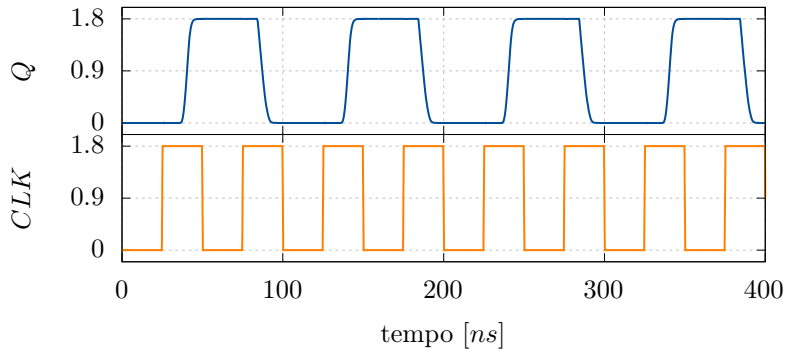

Figura 3.13: *risposta dell'uscita* Q *di un flip-flop JK master-slave considerando entrambi gli ingressi* J, K *allo stato logico alto in funzione del segnale di clock. I terminali in uscita sono stati collegati a massa mediante una capacità di carico di* $0.75pF$*. Le tensioni sono espresse in volt.*

In riferimento a questa simulazione si osserva che il segnale in ingresso, di periodo $50ns$, presenta una frequenza doppia rispetto all'uscita del flip-flop JK master-slave nei quali il periodo è infatti di $100ns$.

Collegando in serie più flip-flop connettendo l'uscita Q del dispositivo con l'ingresso di clock del dispositivo immediatamente successivo, è possibile creare dunque un contatore binario che sfrutta per l'appunto la funzionalità di divisione binaria di clock del latch master-slave.

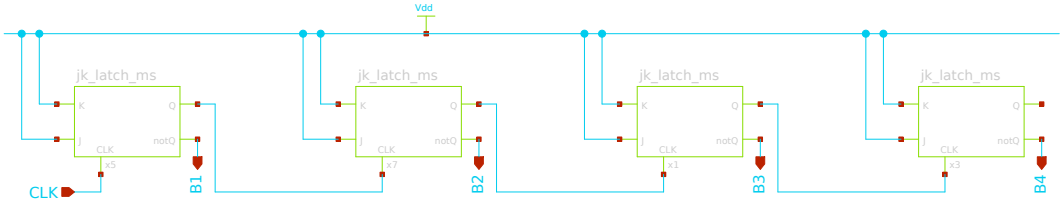

Figura 3.14: *schema a blocchi di un contatore binario con risoluzione a 4 bit.*

Facendo riferimento al contatore binario in figura 3.14, si osserva che B_1 è il bit meno significativo, mentre B_4 è il bit più significativo.

Va inoltre rilevato che l'uscita del segnale Q viene immessa come segnale di clock nel flip-flop JK seguente, mentre l'uscita che misura il valore in uscita è rilevata dall'uscita negata $\overline{Q}$: questo è dovuto al fatto che l'interconnessione così implementata genera dei fronti di salita complementari al valore effettivo. Supponendo infatti che all'istante iniziale tutte le uscite Q dei flip-flop siano poste allo stato logico basso, al primo ingresso di clock tutte commuterebbero, per via di una reazione a cascata, allo stato logico alto. L'effetto che si osserva dal circuito è dunque quello di contatore inverso (che dal valore massimo scorre al valore minimo), tuttavia leggendo le uscite negate $\overline{Q}$ si torna a leggere il conteggio in modo corretto.

Tramite una simulazione nel dominio del tempo, il cui risultato è raffigurato in figura 3.15, è possibile osservare il corretto funzionamento del circuito. Considerando per esempio di voler calcolare il numero di fronti di salita del segnale di clock al tempo $t = 1.5\mu s$ è sufficiente analizzare i singoli bit: $B_1 = 0$, $B_2 = 1$, $B_3 = 1$ e $B_4 = 1$. Componendo il numero binario è possibile anche effettuare la conversione in decimale:

$$1110_2 = 14_{10}$$

Contando il numero di clock effettivamente trascorsi, si osserva che essi sono 15, tuttavia il primo di questi impulsi (per via delle condizioni iniziali del circuito) serve a impostare a 0 tutti bit del circuito, e dunque non viene conteggiato dal contatore binario.

Da questo circuito è possibile ricavare delle soluzioni più complesse che permettano di azzerare il conteggio oppure di abilitare (o meno) la lettura del segnale di clock, in quanto tali funzionalità (allo stato di implementazione attuale) non possono essere effettuate.

Contatore con segnale di reset e di enable

Sfruttando il circuito appena descritto per effettuare il conteggio binario mediante la rilevazione di impulsi di clock, è possibile aggiungere delle funzionalità allo stesso mediante l'aggiunta di altri segnali e porte logiche.

In figura 3.16 è mostrata un'implementazione circuitale di un contatore binario al quale sono state aggiunte le funzionalità di reset (segnale RST) e di abilitazione al conteggio

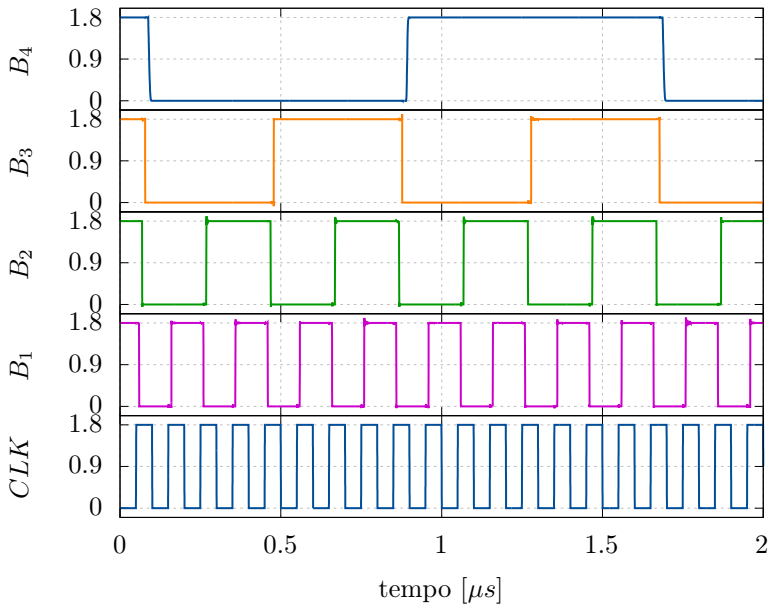

Figura 3.15: *andamento in funzione del tempo del valore dei bit rilevati dal contatore binario a 4 bit (figura 3.14) in funzione del segnale di clock CLK. Le uscite del circuito sono poste a massa tramite una capacità di carico di $0.75pF$. Le tensioni sono espresse in volt.*

(segnale EN). Per realizzare questo tipo di circuito è stato necessario separare il circuito di edge detection del flip-flop JK master-slave dal latch JK vero e proprio.

Funzione di abilitazione La funzione di abilitazione del conteggio è semplice da realizzare, e si basa sul filtrare il segnale di clock in ingresso mediante una porta logica and il cui secondo input è il segnale di abilitazione. Quando EN si trova allo stato alto allora gli impulsi di clock potranno essere correttamente inviati al circuito di edge detection, mentre se EN fosse basso nessun impulso verrebbe mandato come ingresso al contatore.

Funzione di reset La funzione di reset è più complessa ed è costituita dalle connessioni in blu in figura 3.16. Tale comportamento desiderato è ottenuto mediante due passaggi: quan-

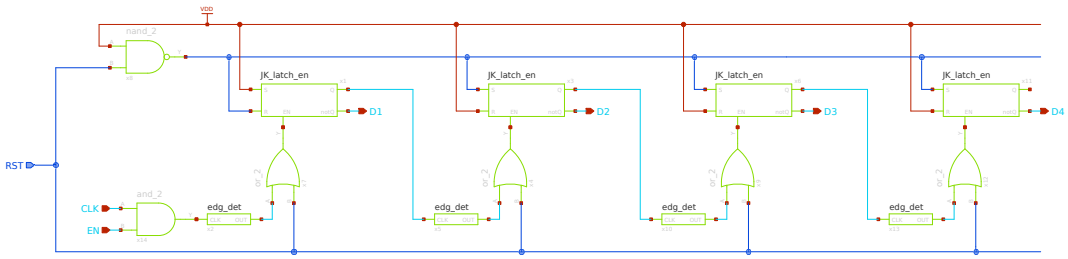

Figura 3.16: *schematico di un contatore binario con risoluzione a 4 bit con segnale di reset e di abilitazione al conteggio.*

do il segnale di reset assume uno stato logico alto tutti i latch vengono resettati (tranne il primo latch che per riuscire a rilevare correttamente il primo impulso deve essere impostato per avere un'uscita alta) mediante il gate nand (sfruttandone la sua tabella di verità) i cui ingressi sono il segnale RST e la tensione di alimentazione.
Il secondo step è quello di abilitare gli ingressi di tutti i latch JK: in questo modo ogni primo stadio dei latch del contatore verrà inizializzato al valore iniziale prestabilito. Nell'immediato l'uscita di reset non risulta evidente, tuttavia appena il segnale RST torna allo stato logico basso il latch master invierà il segnale allo slave effettuando di fatto il ripristino ad uno stato noto del circuito.

Simulazione Nella pagina seguente, in figura 3.17, è possibile osservare una simulazione nel dominio del tempo del circuito, osservando come le uscite rispettino le condizione di reset imposte dal relativo segnale e l'abilitazione permetta (o meno) il conteggio degli impulsi di clock.

3.4 Sommatore binario

Half adder

Un circuito che permette di effettuare la somma binaria di due numeri è composto da una serie di moduli elementari che permettono di effettuare la somma bit-a-bit. Per questo è importante studiare le varianti di questa sezioni circuitali per comprendere il funzionamento del circuito che dovranno comporre.

Il primo elemento che si può studiare è l'*half adder* (mezzo sommatore), un elemento che viene utilizzato per determinare la somma di due bit. Tale circuito deve dunque prevedere necessariamente due ingressi A e B, associati ai valori logici da sommare, mentre le uscite sono 2: il risultato R della somma binaria e l'eventuale riporto C (dall'inglese *carry*).

La somma di due bit in un sistema binario può essere velocemente analizzata tramite la realizzazione della tabella di verità 3.3, dove la terza colonna R_{10} rappresenta il risultato in cifra decimale.

A	B	R_{10}	C	R
0	0	0	0	0
0	1	1	0	1
1	0	1	0	1
1	1	2	1	0

Tabella 3.3: *tabella di verità associata alla somma di due bit A e B. La colonna R_{10} rappresenta il risultato della somma binaria espressa come cifra decimale. Il risultato della somma può essere letto digitalmente concatenando il carry C con il risultato R.*

Dalla tabella di verità che deve essere verificata dal circuito del mezzo sommatore è evidente osservare come il carry C sia determinato dal prodotto logico degli ingressi binari, mentre R sia ottenuto come or esclusivo degli stessi. Questo permette di implementare circuitalmente le porte logiche come in figura 3.18.

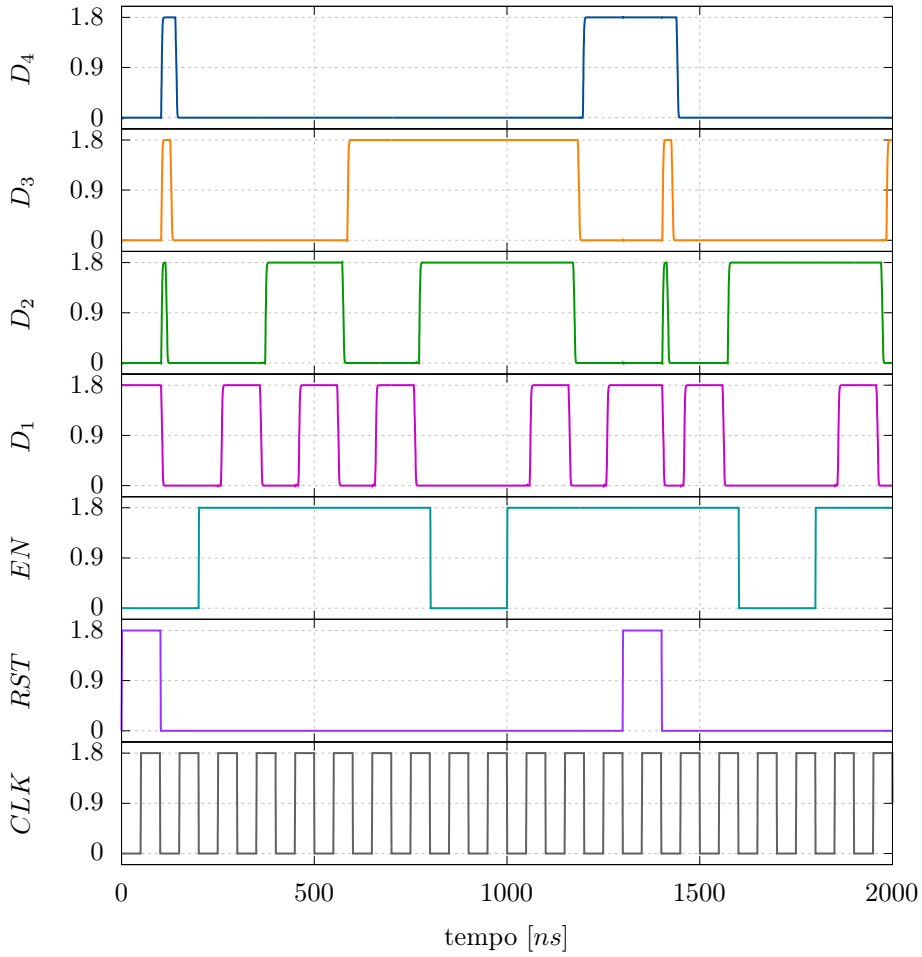

Figura 3.17: *andamento in funzione del tempo dei bit di un contatore binario a 4 bit (figura 3.16) in funzione del segnale di clock CLK, di abilitazione EN e reset RST. Ogni uscita del circuito è collegata a massa mediante una capacità di carico di $0.75pF$. Le tensioni sono espresse in volt.*

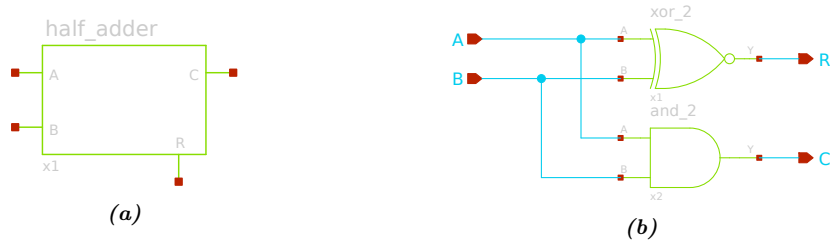

Figura 3.18: *rappresentazione simbolica del blocco half adder (a) e relativa implementazione mediante gate logici (b).*

Il corretto funzionamento del circuito mezzo sommatore può essere verificato anche mediante una simulazione nel dominio del tempo i cui risultati sono mostrati in figura 3.19.

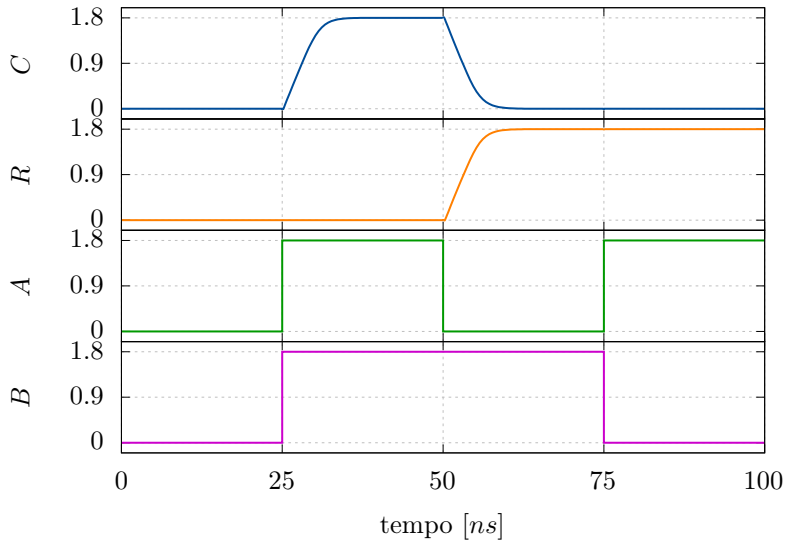

Figura 3.19: *risposta nel dominio del tempo delle uscite dell'half adder in funzione delle possibili combinazioni d'ingresso. Le uscite del circuito sono collegate a massa mediante una capacità di carico di* $0.75pf$*. Le tensioni sono misurate in volt.*

Full adder

Il mezzo sommatore appena descritto è funzionante rispetto alla somma binaria, tuttavia presenta delle limitazioni che ne impediscono l'utilizzo pratico. Infatti questo circuito non permette di considerare un eventuale riporto dovuto ad un sommatore posto a monte.

Per ovviare a questo problema è possibile costruire il *full adder*, ossia un circuito che è in grado di effettuare la somma di due bit A e B considerando anche l'eventuale riporto in ingresso C_{in} da un sommatore a monte. Le uscite, in analogia al mezzo sommatore, sono il risultato R dell'operazione di somma bit-a-bit e il riporto in uscita C_{out}.

Anche in questo caso è possibile realizzare il circuito del full adder mediante previa costruzione della tabella di verità 3.4.

A	B	C_{in}	R_{10}	C_{out}	R
0	0	0	0	0	0
0	1	0	1	0	1
1	0	0	1	0	1
1	1	0	2	1	0
0	0	1	1	0	1
0	1	1	2	1	0
1	0	1	2	1	0
1	1	1	3	1	1

Tabella 3.4: *tabella di verità di un full adder associata alla somma di due bit A e B con riporto in ingresso C_{in}. La colonna R_{10} rappresenta il risultato della somma binaria espressa come cifra decimale. Il risultato della somma può essere letto digitalmente concatenando il riporto in uscita C_{out} con il risultato R.*

A questo punto è possibile costruire un circuito, mediante l'utilizzo di porte logiche, in grado di realizzare tale tabella di verità la cui implementazione logica è mostrata in figura 3.20 e rispetto ad essa è possibile trarre delle conclusioni sul funzionamento del circuito.

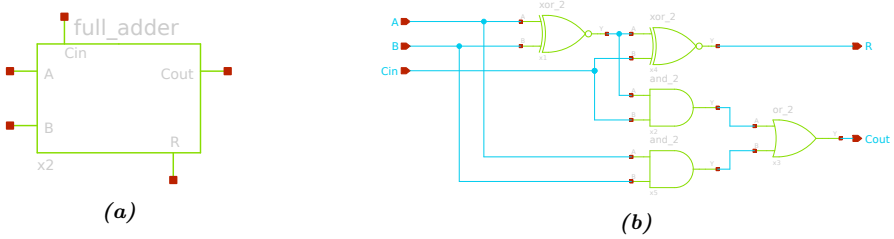

Figura 3.20: *rappresentazione simbolica del blocco half adder (a) e relativa implementazione mediante gate logici (b).*

La cascata di circuiti xor (componenti `x1` e `x2`) permette di determinare univocamente il risultato R del bit corrente, in quanto determinerà un valore logico alto in uscita solamente se un numero dispari di ingressi fossero anch'essi posti ad alta tensione.

La definizione del riporto in uscita C_{out} prevede invece la considerazione di due casi: se gli ingressi A, B fossero stati logici alti, allora automaticamente il riporto in uscita dovrà essere posto ad alta tensione, e tale funzione è realizzata mediante il gate and `x5`. Una seconda casistica da considerare è quella complementare per cui la somma tra A e B non generasse riporto, ma potrebbe generarla la somma insieme al riporto in ingresso. Questo caso è analizzato dal gate and `x2` per cui gli ingressi sono il carry-in e il risultato della somma bit-a-bit degli ingressi (ottenuta dal gate xor `x1`). Il carry-out è dunque alto se almeno una delle due situazioni descritte è verificata.

Sommatore binario a 4 bit

Avendo appena descritto l'elemento circuitale alla base della somma di due bit distinti (con eventuale riporto in ingresso), è possibile realizzare uno schematico che permette la somma di due dati in forma binaria tramite la concatenazione dei full adder, come si può osservare in figura 3.21.

In questo caso i segnali di input sono i numeri binari A e B, oltre ad un eventuale riporto in ingresso C_{in} (che può essere utilizzato per fare la somma di numeri a più bit rispetto

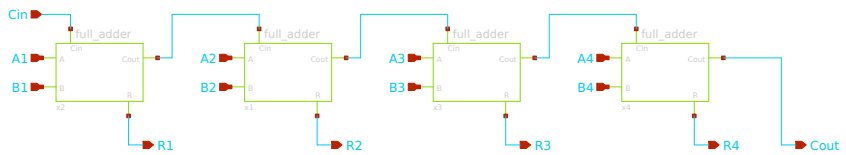

Figura 3.21: *implementazione di un sommatore binario a 4 bit.*

all'architettura del sistema o può essere utilizzato rispetto alla codifica *1 complementare* dei numeri binari) mentre l'uscita è determinata dai bit del risultato R e dal riporto in uscita C_{out}. Come nel caso del contatore, il suffisso numerico 1 indica il bit meno significativo, mentre 4 il bit più significativo.

Effettuare una simulazione completa di tutte le possibili combinazioni di ingresso (che sono in totale $16^2 = 256$) diventa praticamente difficile da effettuare ma soprattutto da visualizzare, per questo all'interno del documento viene mostrata una casistica rappresentativa ottenuta fissando un addendo e facendo variare l'altro.

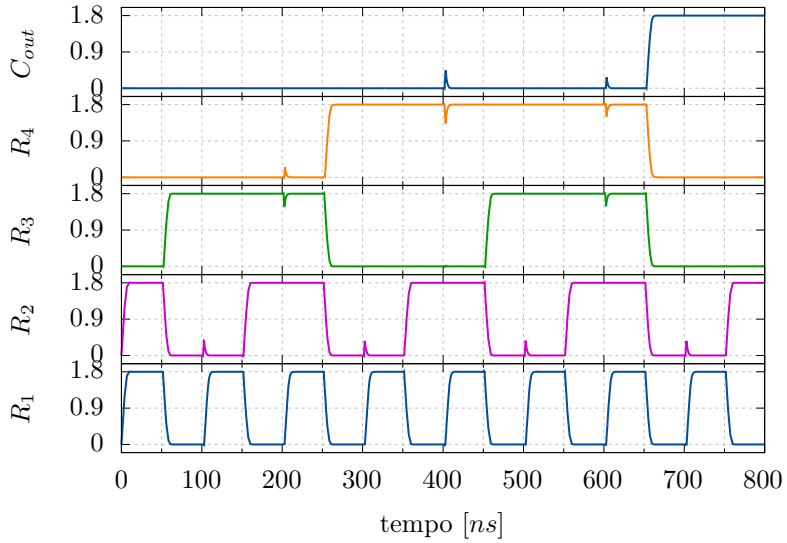

Figura 3.22: *simulazione sul funzionamento del circuito di somma binaria. I grafici evidenziano l'andamento temporale dell'uscita dovuto alla somma di un ingresso costante $B = 3_{10} = 0011_2$ e un ingresso A variabile da $0_{10} = 0000_2$ a $15_{10} = 1111_2$ che viene incrementato ogni $50ns$. Ogni uscita è collegata a massa mediante una capacità di carico di $0.75pF$. Le tensioni sono misurate in volt.*

Considerando la simulazione in figura 3.22 dove si suppone un ingresso costante (in questo caso $B = 3_{10} = 0011_2$) è possibile valutare la risposta del circuito per le 16 combinazioni in ingresso del segnale A considerando che lo stesso aumenta di un'unità ogni $50ns$. Per esempio considerando l'intervallo di tempo di $300 - 350ns$ associato ad un ingresso $A = 6_{10} = 0110_2$, si osserva che il risultato della somma è dato dai bit

$$R = R_4R_3R_2R_1 = 1001_2 = 9_{10} \qquad , \qquad C_{out} = 0$$

Si verifica dunque il funzionamento del circuito. Si osserva inoltre che per i tempi $t > 650ns$ ($A \geq 13_{10}$) il riporto in uscita C_{out} commuta allo stato logico alto: questo comportamento risulta essere corretto in quanto numeri a 4 bit possono rappresentare solamente 16 numeri

decimali (da 0 a 15). Le uscite per cui $A \geq 13_{10}$ (essendo $B = 3_{10}$) determinano una somma che è fuori dal range di rappresentazione a 4 bit, per questo viene posto ad alto il riporto in uscita (che si può considerare come il quinto bit più significativo del risultato) e i bit di risultato R_i lavorano come di consueto. Utilizzando C_{out} come quinto bit significativo è possibile verificare anche le somme per $A \geq 13_{10}$: considerando come esempio l'ultimo ingresso $A = 15_{10} = 1111_2$ (per $t \in [750, 800]ns$) l'uscita del circuito è data dall'espressione

$$R = C_{out}R_4R_3R_2R_1 = 10010_2 = 18_{10}$$

3.5 Moltiplicatore binario

L'operazione di moltiplicazione binaria è una delle funzioni più complesse da realizzare a livello hardware, in quanto richiede la presenza di un numero elevato di componenti o svariati cicli di clock per svolgere l'operazione matematica. La moltiplicazione può essere effettuata secondo diverse metodologie che possono essere sia di tipo algoritmico (sfruttando una procedura che utilizza circuito sommatore e registri a scorrimento) oppure implementate a livello hardware.

In questo caso si farà riferimento a quest'ultima metodologia, in quanto più immediata da analizzare. Per capire come effettuare l'operazione è necessario schematizzare il funzionamento richiesto effettuando manualmente un'operazione di moltiplicazione binaria, dove in questo caso si utilizza un moltiplicando a a 4 bit e un moltiplicatore b di 3 bit.

		a_4	a_3	a_2	a_1	$\times$
			b_3	b_2	b_1	$=$
		a_4b_1	a_3b_1	a_2b_1	a_1b_1	$+$
	a_4b_2	a_3b_2	a_2b_2	a_1b_2		$+$
a_4b_3	a_3b_3	a_2b_3	a_1b_3			$=$
a_4b_3	$a_4b_2 + a_3b_3$	$a_4b_1 + a_3b_2a_2b_3$	$a_3b_1 + a_2b_2a_1b_3$	$a_2b_1 + a_1b_2$	a_1b_1	$=$

I vari termini ottenuti nelle somme parziali sono determinati direttamente dal prodotto logico di due bit, e dunque possono essere determinati (a livello circuitale) velocemente mediante gate and. A questo punto è invece necessario effettuare la somma binaria delle varie somme parziali, considerando che le stesse devono essere *spostate a sinistra* di un'unità ad ogni iterazione (come viene effettuato normalmente nel prodotto decimale ed è rappresentato nella tabella riassuntiva).

In figura 3.23 è mostrato il circuito combinatorio che realizza l'operazione di moltiplicazione binaria mediante l'utilizzo di gate and oltre che al circuito half adder (HA) e full adder (FA); le interconnessioni in blu rappresentano il percorso degli ingressi del moltiplicando nel circuito, mentre in rosso sono evidenziati i bit del moltiplicatore.

I problemi legati ad un'implementazione in tecnologia c-mos, come così mostrata, è che richiede un'elevata quantità di risorse hardware intese come numero di transistor e dunque spazio su chip. Considerando infatti un moltiplicando di n bit e un moltiplicatore di m bit, allora il numero di linee di circuiti di addizione è pari a $m - 1$ (con n full/half adder l'una) e l'uscita può richiedere l'utilizzo fino a $n + m$ bit. Lo schematico risulta essere "compatto" per valori di m, n sufficientemente piccoli (in questo caso $m = 4, n = 3$), tuttavia in sistemi reali dove può essere necessario effettuare operazioni tra dati a 16 bit o più, l'impatto risulta essere importante.

Simulazione Come nel caso del circuito sommatore, testare esaustivamente tutte le possibili combinazioni d'ingresso risulta complesso, tuttavia è possibile mostrare la correttezza

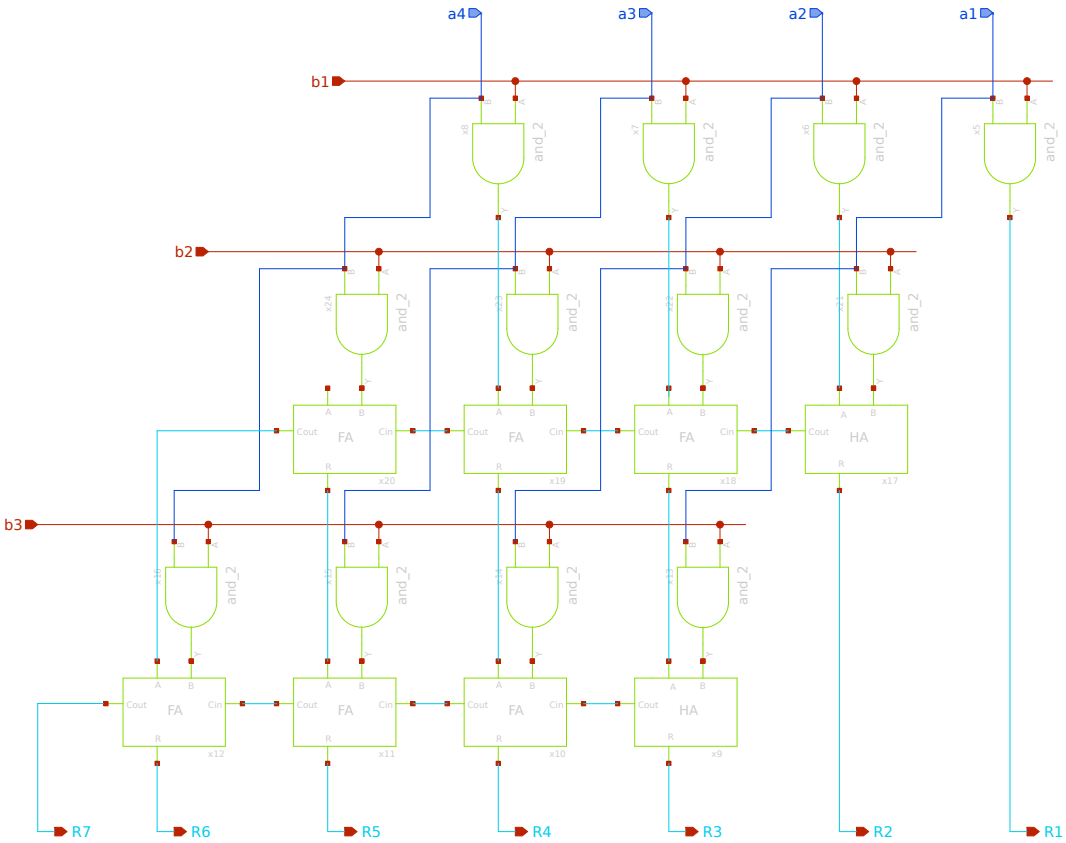

Figura 3.23: *implementazione circuitale di un moltiplicatore binario con moltiplicando a 4 bit e moltiplicatore a 3 bit.*

di funzionamento del circuito fissando un parametro e facendo variare l'altro termine. In figura 3.24 è mostrata una simulazione nel dominio del tempo fissando a 5_{10} il moltiplicando e facendo variare da 0 a 7 il moltiplicatore.

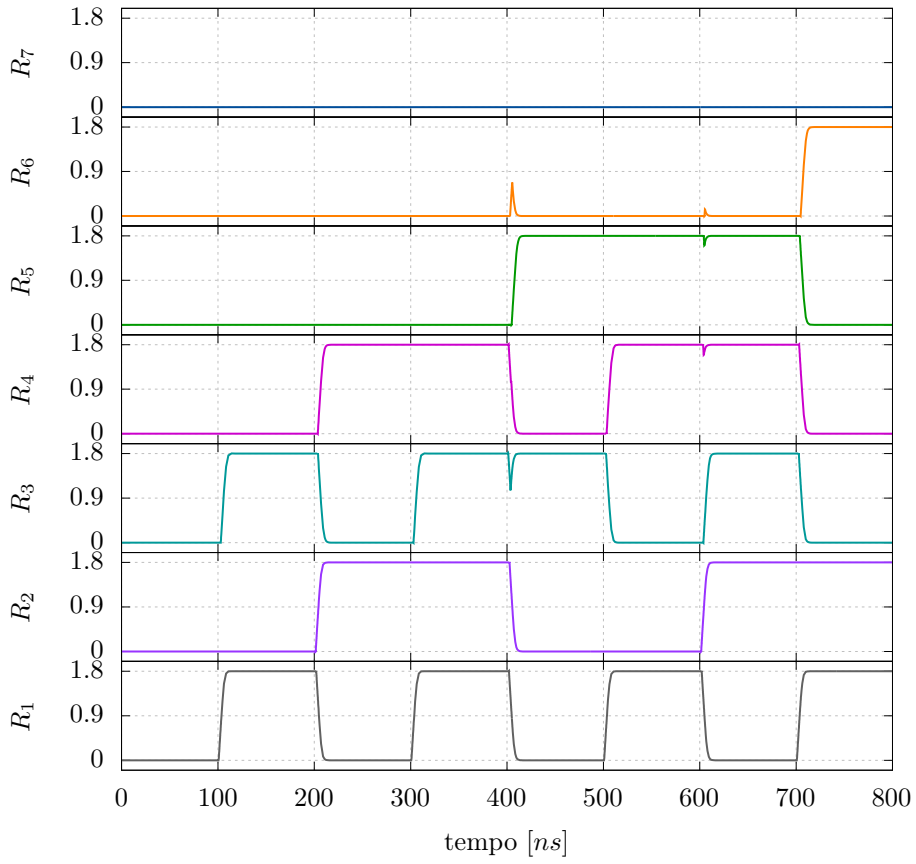

Figura 3.24: *bit in uscita ottenuti mediante la moltiplicazione di un moltiplicando costante pari a* $0101_2 = 5_{10}$ *e un moltiplicatore che incrementa ogni* $100ns$ *di una unità dal valore* $0000_2 = 0_{10}$ *al valore* $1111_2 = 15_{10}$*. Tutte le uscite sono collegate a massa mediante una capacità di carico di* $0.75pF$*. Le tensioni sono espresse in volt.*

Capitolo 4

Elementi di memoria

Di fondamentale importanza per l'utilizzo pratico è realizzare dei circuiti con funzionalità di memoria che permettano di immagazzinare dati che sono accessibili al calcolatore per l'elaborazione. Nel capitolo precedente sono state mostrati dei tipi di circuiti in grado di memorizzare dati binari (come i latch e i flip-flop), tuttavia essi hanno utilità pratica nell'implementazione di circuiti logici con funzioni specifiche, ma non sono adatti a realizzare circuiti di memoria in quanto richiedono un numero elevato di transistor e dunque di ingombro su chip (rispetto alle soluzioni che verranno qua esposte), limitando la quantità di memoria che è possibile predisporre.

In primo luogo verranno descritti i principali componenti per la gestione dei conflitti di tensione nell'interconnessione di diversi componenti digitali, ossia i *tri-state buffer*. Verranno poi analizzate due principali celle di memoria *RAM* (*Random Access Memory*): la versione statica, più performante, e la versione dinamica, soluzione dall'ingombro su chip ridotto.

4.1 Tri-state buffer

I sistemi digitali sono realizzati mediante un'interconnessione di diverse sezioni circuitali (come i vari sommatori e moltiplicatori binari, ma anche registri) alle quali deve essere permesso di scambiare i dati elaborate in maniera efficiente. Tutti questi elementi sfruttano il cosiddetto *bus dati* per condividere le informazioni, tuttavia è necessario implementare un sistema che sia in grado di stabilire quale circuito sia in grado di scrivere e/o leggere le informazioni sul bus dati. Questa operazione è necessaria per evitare il problema del *bus contention* dovuta ad una scrittura simultanea di circuiti. Se infatti due sezioni circuitali provassero a imporre una tensione logica diversa, allora sul bus dovrebbe scorrere un'elevata quantità di corrente in quanto la resistenza del circuito è pressoché nulla rispetto alla differenza di tensione cui il bus verrebbe sottoposto.

Tale funzione viene realizzata tramite i cosiddetti *tri-state buffer* (e *inverter*), ossia buffer a tre possibili stati che, tramite un apposito segnale di abilitazione in ingresso, permettono (o meno) il passaggio del segnale in ingresso al bus dati. La tabella di verità realizzata può essere schematizzata come

In	*En*	*Out* buffer	*Out* inverter
0	1	0	1
1	1	1	0
0	0	Hi-Z	Hi-Z
1	0	Hi-Z	Hi-Z

Come si osserva nel caso di segnale di abilitazione alto, il buffer permette di avere un'uscita che corrisponde al valore logico in ingresso (l'invertitore avrà invece l'uscita negata). La peculiarità interessante del circuito è invece è quanto il segnale di abilitazione è basso: in questo caso l'uscita (rispetto all'ingresso) si pone in una condizione di alta impedenza, "scollegando" l'ingresso dall'uscita e rendendo le due non influenzabili.

La tabella di verità mostra un componente tri-state che permette la comunicazione se il segnale di abilitazione è alto, tuttavia è anche possibile realizzare dei circuiti che richiedono un segnale di abilitazione basso per il trasferimento dell'ingresso.

L'implementazione circuitale di un invertitore tri-state è piuttosto semplice ed è rappresentata in figura 4.1 ed è composto sostanzialmente da una serie di 2 p-mos e 2 n-mos opportunamente pilotati.

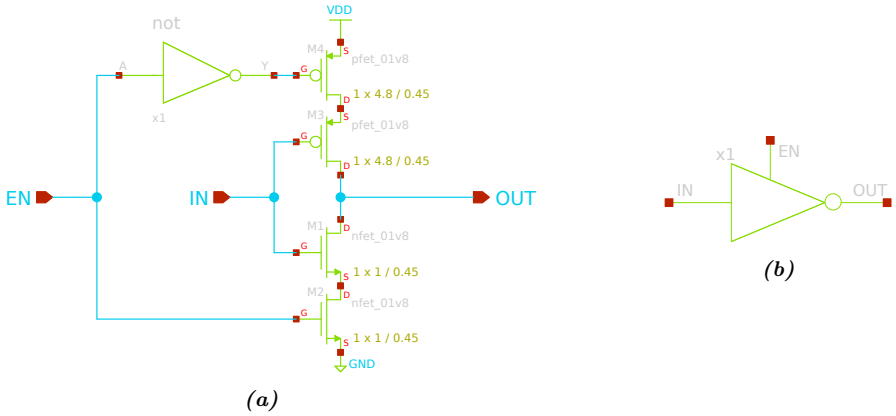

Figura 4.1: *schematico (a) e relativa rappresentazione (b) di un invertitore tri-state.*

In particolare si osserva che i due transistor più interni, collegati all'ingresso dei dati, determinano di fatto un invertitore c-mos. L'aspetto che caratterizza il circuito invece è dato dai transitor più esterni collegati all'ingresso di abilitazione EN (segnale che per il p-mos viene negato). Il funzionamento del circuito può dunque essere analizzato per casi:

- nel caso in cui l'ingresso di abilitazione si trovi allo stato logico alto, allora i transistori ad essi associati risultano entrambi contemporaneamente "accessi", permettendo lo scorrimento di corrente. L'uscita è dunque determinata dall'invertitore associato all'input;
- nel caso in cui l'ingresso di abilitazione si trovi allo stato logico basso invece i transistor associati risultano interdetti: questo di fatto pone in interdizione il circuito dell'invertitore che indipendentemente dall'ingresso non può permettere uno scorrimento di corrente e dunque l'uscita è posta in alta impedenza rispetto all'ingresso.

Un tri-state buffer viene realizzato a partire dal tri-state inverter ponendo in serie all'uscita un invertitore logico.

4.2 RAM statica

Le memorie *RAM*, *Random Access Memory*, mantengono il loro nome per motivi storici e sono caratterizzate dall'avere tempo di accesso ai dati costante indipendentemente dalla

posizione degli stessi nello slot di memoria (al contrario di altri tipi di memorie). Questo tipo di memorie sono definite volatili in quanto non adatte a conservare dati, i quali verranno persi non appena verrà tolta l'alimentazione al circuito.

Allo stato attuale le memorie ram possono essere catalogate come *dinamiche*, ossia che richiedono un continuo aggiornamento dei dati immagazzinati per mantenere attive le informazioni salvate, e che costituiscono i moduli di memoria ram commerciali per i computer attuali. Esiste anche la memoria di tipo *statico* che al contrario del caso precedente non richiede un aggiornamento del modulo per mantenere le informazioni ed è dunque più veloce (al costo di un numero maggiore di transistor e maggior ingombro su chip) e per questo è utilizzata all'interno dei microprocessori per realizzare le memorie cache di livello 1, 2 e 3, le cui dimensioni in memoria sono generalmente ridotte ma è necessario che le stesse siano prestanti.

Una cella da 1 bit di memoria statica, abbreviata generalmente come *s-ram*, è costituita da un anello di invertitori logici (che permettono di immagazzinare le informazioni digitali) e da due n-mos in grado di controllare quando la cella è in grado di leggere/scrivere le informazioni sul bus dati e la realizzazione schematica è riportata in figura 4.2.

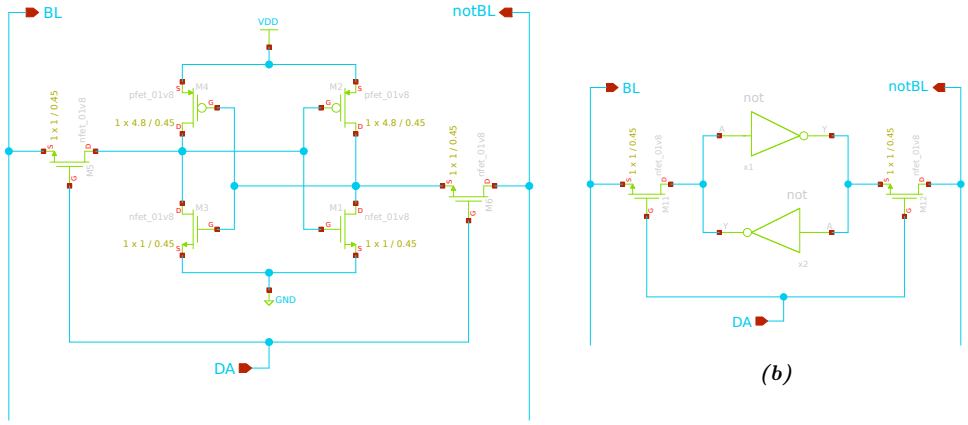

Figura 4.2: *implementazione circuitale di una cella ad un bit di una ram statica (a). Nella figura (b) è evidenziato l'anello di invertitori logici.*

Rispetto allo schematico è possibile osservare come la cella di memoria sia collegata a due cosidette *bit lines* BL il cui ruolo logico si dimostra essere complementare. Nel momento in cui il segnale di *word line* WL viene posto al valore logico alto allora i due n-mos che determinano la connessione dell'anello di invertitori con il bus dati permettono di eguagliare le tensioni delle due parti.

La funzione di memoria viene realizzata dall'anello di invertitori logici: la loro concatenazione in anello chiuso rafforza la differenza di stato logico dei due componenti. Considerando infatti che la parte a sinistra del circuito si trovi ad uno stato logico alto (rispetto alla figura 4.2.b), percorrendo l'anello in senso orario si osserva che l'invertitore impone un segnale logico basso alla parte a destra, che a sua volta rafforzerà il segnale logico alto della parte a sinistra.

Si spiega così la dualità delle due bit lines: l'anello di invertitori determina espressamente due uscite che sono necessariamente allo stato logico complementare.

Pilotando le bit lines è possibile, abilitando la comunicazione della cella dati, scrivere le informazioni dal bus dati nella memoria statica; al contrario se la linea dati non è pilotata sarà la cella a pilotare le bit lines.

Le memorie attuali sono ottenute collegando alle stesse bit lines un'elevata quantità di celle elementari il cui accesso (che può essere effettuato solo per una cella ad ogni istante) può essere determinata da una logica di decodifica come un multiplexer. Una struttura così implementata permette di immagazzinare solamente dati ad 1 bit, tuttavia è sufficiente porre in parallelo un numero congruo di queste strutture per realizzare sistemi di memoria per un numero maggiore di bit.

4.3 RAM dinamica

La memoria ram dinamica (abbreviata d-ram), come già affermato, presenta un tempo di accesso ai dati maggiore rispetto alla controparte statica in quanto è necessario aggiornare ad una determinata frequenza tutte le celle di memoria.

La cella elementare della ram dinamica è caratterizzata dalla presenza di un solo transistor e di una capacità che funge di fatto come dispositivo di memorizzazione delle informazioni.

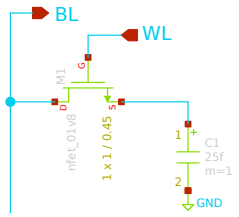

Figura 4.3: *implementazione circuitale di una cella di memoria d-ram ad un bit.*

Il funzionamento del circuito può essere analizzato in riferimento allo schema di una cella di memoria dinamica mostrato in figura 4.3. Come nel caso della controparte statica, le operazioni di lettura/scrittura dati avvengono su una bit line BL e l'abilitazione alla comunicazione (mediante il segnale word line WL) avviene tramite un n-mos. Il bit di informazione è associato, come detto, alla carica di un condensatore (che rispetto alle celle di memoria degli anni Novanta il valore nominale è di circa $25 - 30fF$ [1]). La bit line può infatti effettuare un'operazione di scrittura sul transistor imponendo la tensione differenziale ai suoi capi che potrà poi essere successivamente letta dalla linea stessa (se essa non viene pilotata da altri circuiti).

Il problema di questo tipo di circuito è che la capacità non è un dispositivo ideale e dunque nel tempo è destinata a scaricarsi per via delle correnti parassite. La memoria è detta dinamica per la necessità di eseguire delle operazioni di *refresh*, ossia di ri-aggiornamento dei dati di tutte le celle per evitare di perdere informazioni. Ogni scheda di memoria d-ram è infatti sottoposta a cicli periodici di lettura di tutte le celle con conseguente riscrittura dello stato logico connesso. Questo introduce necessariamente delle latenze nella comunicazione e trasmissione dei dati.

La valutazione delle prestazioni delle memorie ram era inizialmente calcolata in base al tempo di accesso ai dati (in funzione dell'operazione che si vuole eseguire) con valori che potevano variare dai $20-30ns$ a $80-90ns$ [1]. Le memorie dinamiche attualmente utilizzate

sui pc, appartenenti allo standard DDR *Double Data Rate*, sono invece caratterizzate principalmente dalla frequenza di aggiornamento (*refresh rate*) che si attesta tra i $1\,600mHz$ e i $4\,000mHz$, e dalla latenza di clock, ossia il numero di cicli della memoria ram necessari a comunicare i dati (compresi tra 13 e 20).

Bibliografia e sitografia

[1] Changhyun Kim (Samsung Electronics). *Basic DRAM Operations slides*. A cura di Berkeley University of California. 1995. URL: https://people.eecs.berkeley.edu/~pattrsn/294/LEC9/lec.html.

[2] SkyWater Technology Foundry Google. *Google Skywater PDK*. URL: https://github.com/google/skywater-pdk.

[3] Hardware Museum. *CPU History Tour (1999-2001)*. URL: http://hw-museum.cz/article/6/cpu-history-tour--1999---2001-/3.

[4] Stefan Frederik Schippers. *XSchem*. URL: https://github.com/StefanSchippers/xschem.

[5] *SkyWater Device Details Documentation*. URL: https://skywater-pdk.readthedocs.io/en/latest/rules/device-details.html.

[6] *SkyWater Foundry Provided Standard Cell Libraries Documentation*. URL: https://skywater-pdk.readthedocs.io/en/latest/contents/libraries/foundry-provided.html.

[7] *SkyWater Technology foundry*. URL: https://www.skywatertechnology.com/.

[8] ngspice team U.C. Berkley CAD Group. *ngspice*. URL: http://ngspice.sourceforge.net/.

Per la stesura del seguente documento sono stati consultati:

- *CMOS Memory*, Giuseppe Ianaccone, corso di Electronic System (anno 2016), Università di Pisa, http://www.iannaccone.org/ES2016/