Politecnico di Torino
III Facoltà di Ingegneria

Sistemi elettronici a basso consumo

Relazioni di laboratorio

Laurea Magistrale in Ingegneria Elettronica
Orientamento: Sistemi Elettronici

Gruppo n. 9

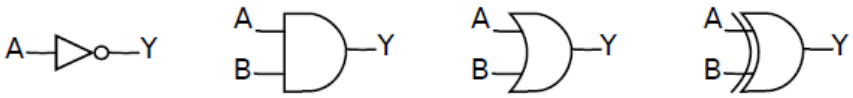
Autori:
Favero Simone, Micelli Federico, Spanna Francesca

Contents

Laboratorio 1 - Power Estimation: probabilistic techniques

1.1 Calcolo di probabilità e attività: porte logiche elementari

Il primo esercizio consiste nel valutare le probabilità e attività dell'uscita di quattro porte logiche elementari: NOT, AND, OR e XOR.



Mentre la probabilità di uscita del gate è definita dalla funzione logica stessa, la switching activity è valutata allo stesso modo per tutti i casi, mediante la seguente formula:

$$A = 2 \cdot P1 \cdot (1 - P1)$$

dove $P1$ indica la probabilità che l'uscita assuma valore logico alto.

Di seguito è riportata l'analisi delle porte logiche richieste, considerando ingressi equiprobabili e scorrelati.

- **NOT**
  $P(Y = 1) = 1 - P(A = 1) = 0.5$
  $A(Y) = 0.5$
- **AND**
  $P(Y = 1) = P(A = 1) \cdot P(B = 1) = 0.25$
  $A(Y) = 0.375$
- **OR**
  $P(Y = 1) = 1 - ((1 - P(A = 1)) \cdot (1 - P(B = 1))) = 0.75$
  $A(Y) = 0.375$
- **XOR**
  $P(Y = 1) = P(A = 1) \cdot (1 - P(B = 1)) + P(B = 1) \cdot (1 - P(A = 1)) = 0.5$
  $A(Y) = 0.5$

Simulando il test bench fornito tramite ModelSim, è possibile ottenere un file riportante il numero di commutazioni di ogni segnale del circuito durante il tempo di simulazione.
Il testbench fornito sfrutta un generatore di numeri casuali per generare gli ingressi delle porte, rendendo questi ultimi equiprobabili e statisticamente indipendenti.
Sono riportati i seguenti valori:

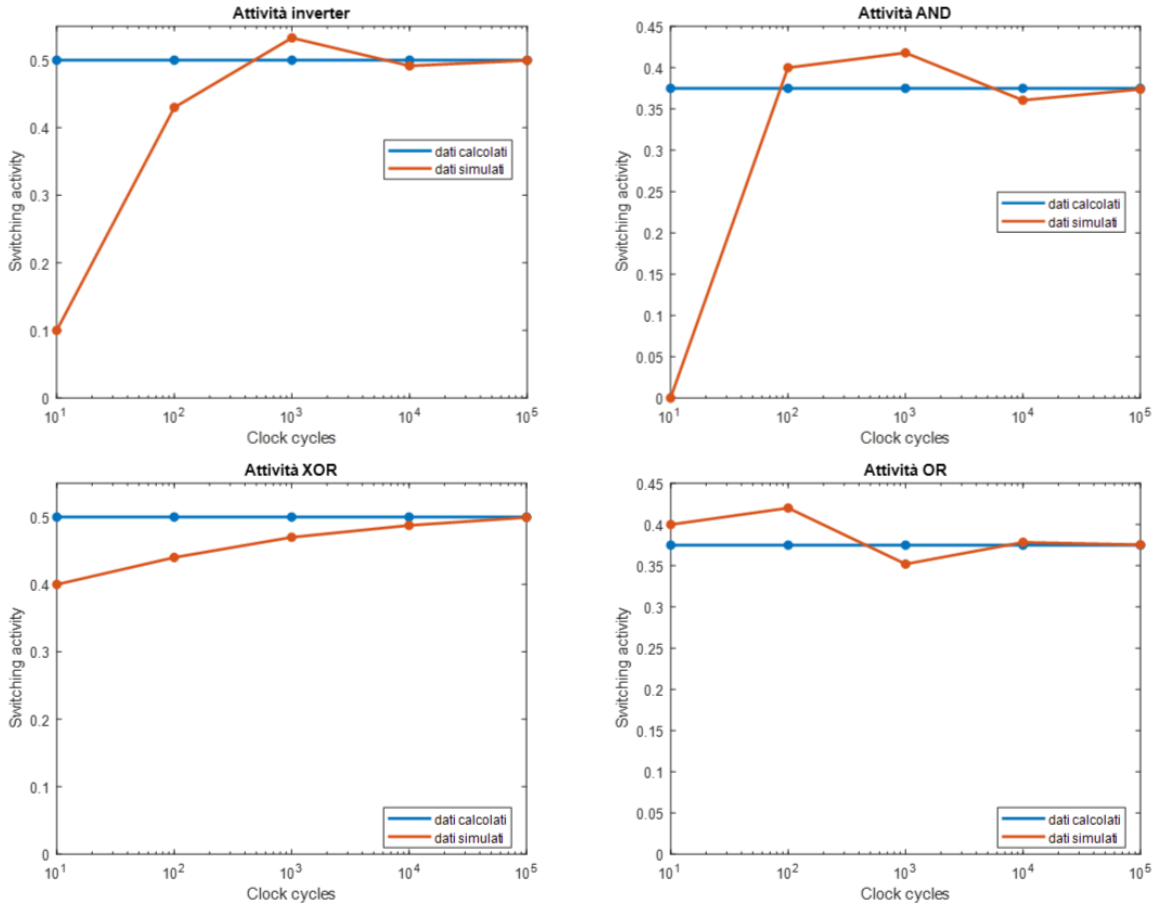
Tc(CK)	Tc(INV)	Tc(AND)	Tc(OR)	Tc(XOR)
20	1	0	4	4
200	43	40	42	44
2000	533	418	352	470
20000	4916	3606	3784	4876
200000	49967	37834	37541	49939

È possibile stimare la switching activity dividendo il numero di commutazioni di un nodo per il numero di colpi di clock della relativa simulazione.
Dal momento che il parametro Tc si riferisce al numero totale di commutazioni, il numero di cicli di clock è ottenuto dividendo per due il parametro Tc(CK).

I risultati dei calcoli sono riportati nella seguente tabella.

Tc(CK)	Esw(INV)	Esw(AND)	Esw(OR)	Esw(XOR)
20	0.1	0	0.4	0.4
200	0.43	0.40	0.42	0.44
2000	0.533	0.418	0.352	0.470
20000	0.4916	0.3606	0.3784	0.4876
200000	0.4997	0.3783	0.3754	0.4994

Per garantire una migliore visualizzazione dei dati ottenuti al variare del tempo di simulazione, sono stati realizzati i seguenti grafici.



Si osserva che, all'aumentare del tempo di simulazione, la stima dell'attività risulta sempre più accurata. In particolare, nel caso analizzato, si osserva che per un numero di cicli di clock superiore a 10000, i dati simulati sono confrontabili con quelli teorici.

1.2 Calcolo di probabilità e attività: half adder e full adder

Dalle tavole di verità di Half Adder e Full Adder si ottengono le seguenti funzioni:

- **Half adder**

S = A XOR B
Cout = A AND B

- **Full adder**

S = A XOR B XOR Cin
Cout = A AND B AND Cin

Partendo dalle funzioni logiche che descrivono le uscite è stato possibile ricavare le probabilità associate alle uscite e le relative attività.

- **Half adder**

$$P(S=1) = P(A=1)\cdot((1-P(B=1)) + P(B=1)\cdot(1-P(A=1))$$

$$P(Cout=1) = P(A=1)\cdot P(B=1)$$

$$A(S) = 2\cdot P(S=1)\cdot(1-P(S=1))$$

$$A(Cout) = 2\cdot P(Cout=1)\cdot(1-P(Cout=1))$$

- **Full adder**

$$\begin{aligned} P(S=1) = {} & P(A=1)\cdot(1-P(B=1))\cdot(1-P(Cin=1)) + \\ & + P(B=1)\cdot(1-P(A=1))\cdot(1-P(Cin=1)) + \\ & + P(Cin=1)\cdot(1-P(A=1))\cdot(1-P(B=1)) + \\ & + P(A=1)\cdot P(B=1)\cdot P(Cin=1) \end{aligned}$$

$$\begin{aligned} P(Cout=1) = {} & P(A=1)\cdot P(B=1)\cdot(1-P(Cin=1)) + \\ & + P(Cin=1)\cdot P(A=1)\cdot(1-P(B=1)) + \\ & + P(Cin=1)\cdot P(B=1)\cdot(1-P(A=1)) + \\ & + P(A=1)\cdot P(B=1)\cdot P(Cin=1) \end{aligned}$$

$$A(S) = 2\cdot P(S=1)\cdot(1-P(S=1))$$

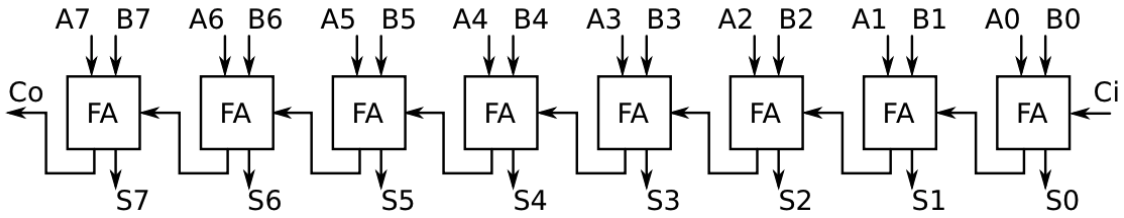
$$A(Cout) = 2\cdot P(Cout=1)\cdot(1-P(Cout=1))$$

I risultati ottenuti sono riportati nella seguente tabella.

P(A) = P(B) = 0.5

	P(S=1)	A(S)	P(Cout=1)	A(Cout)
HA	0.5	0.5	0.25	0.375
FA	0.5	0.5	0.5	0.5

Sfruttando i risultati ottenuti per il singolo full adder, è stato possibile ottenere le probabilità e le attività di un ripple carry adder avente parallelismo pari a 8 bit.



Per tale analisi sono stati considerati equiprobabili gli ingressi A e B mentre il carry in ingresso al primo full adder è stato fissato a 0.
I risultati ottenuti sono riportati nella seguente tabella.

P(A) = P(B) = 0.5

	S7	S6	S5	S4	S3	S2	S1	S0	Cout
P(Y=1)	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.4980
A(Y)	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5

Variando le probabilità associate agli ingressi, in particolare considerando P(A=1) = 0.4 e P(B=1) = 0.6, si ottengono i seguenti risultati.

P(A) = 0.4 e P(B) = 0.6

	S7	S6	S5	S4	S3	S2	S1	S0	Cout
P(Y=1)	0.52	0.5104	0.5054	0.5028	0.5015	0.5008	0.5004	0.5002	0.4973
A(Y)	0.4992	0.4998	0.4999	0.5	0.5	0.5	0.5	0.5	0.5

È possibile notare come la probabilità delle varie uscite sia funzione delle probabilità degli ingressi. In particolare, la probabilità delle uscite relative alle somme aumenta. Si nota che ai bit relativi agli stadi iniziali sono associati valori che non si discostano molto dal caso di ingressi equiprobabili mentre i bit più significativi subiscono una variazione più marcata.

Al fine di avere un riscontro sui dati stimati, sono state effettuate due simulazioni tramite ModelSim: in un primo caso è stato considerato un ritardo limitato alle uscite relative alle

somme; successivamente è stato introdotto un ulteriore ritardo anche sul carry in uscita di ogni full adder. Come descritto precedentemente, il file fornito da Modelsim riporta il numero di commutazioni dei segnali, di conseguenza l'attività è stata calcolata in modo analogo all'esercizio precedente.

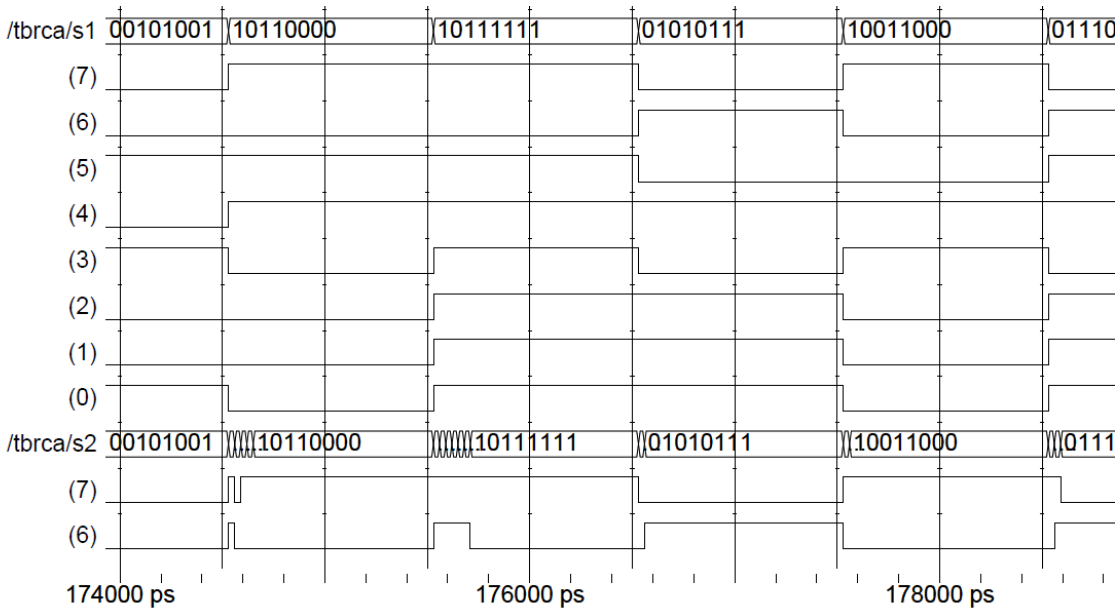
	A(S7)	A(S6)	A(S5)	A(S4)	A(S3)	A(S2)	A(S1)	A(S0)	A(Cout)
Sim. 1	0.42	0.51	0.525	0.425	0.49	0.505	0.51	0.46	0.615
Sim. 2	1.25	1.21	1.065	0.995	1.05	1.005	0.89	0.46	0.615

Si nota come la prima simulazione abbia fornito dati comparabili con quelli precedentemente stimati. Nella seconda simulazione, invece, la presenza di un ritardo sul carry di uscita di ogni full adder ha come conseguenza un generale aumento dell'attività delle uscite di somma. Tale ritardo causa infatti glitch, responsabili dell'aumento della E_{sw}.

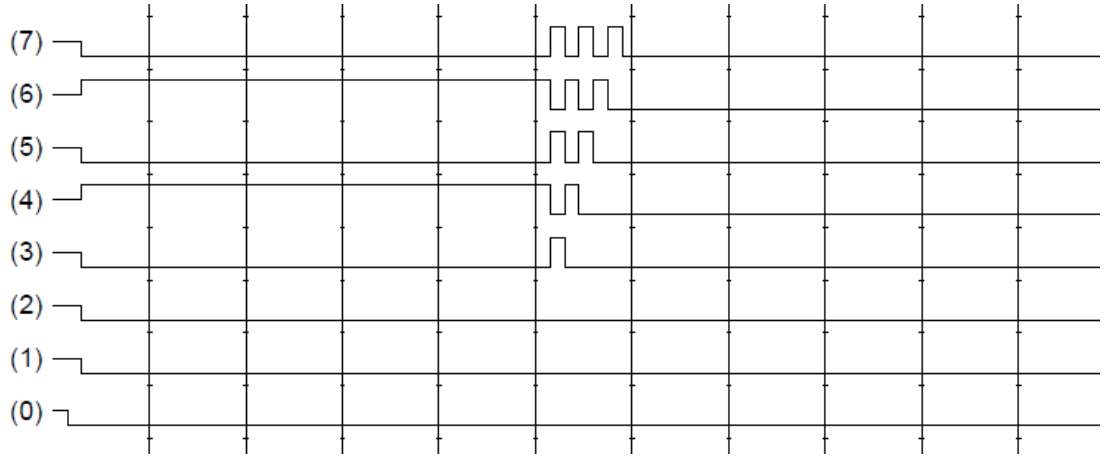
Tale risultato è particolarmente evidente considerando la E_{sw} totale dei due casi: si nota che la presenza del ritardo sui carry out raddoppia tale valore.

$$A(S) = \sum_{i=0}^{N-1} A(S_i)$$

	Sim. 1	Sim. 2
A(S)	3.845	7.925



È possibile notare, attraverso le waveforms, la presenza di commutazioni indesiderate nel secondo caso, rappresentato dal vettore di segnali s2.



In particolare, nell'immagine sopra riportata è possibile osservare come i bit relativi all'uscita di somma, a partire dal quarto, presentino una continua commutazione fino al raggiungimento del valore logico corretto. La causa di tale comportamento è da ricercare nelle transizioni a cui gli ingressi sono stati sottoposti nella simulazione. Le due somme, calcolate una di seguito all'altra, sono le seguenti:

	Somma 1	Somma 2
Addendo 1	10101000	00000100
Addendo 2	10101000	11111100
Risultato	01010000	00000000
Carry in	01010000	

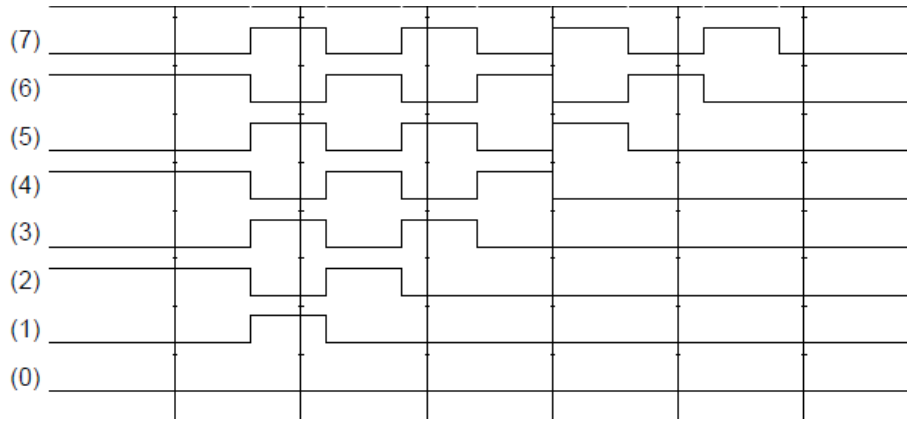
In particolare è possibile osservare che la somma 2 è soggetta ad una propagazione di carry dal terzo bit in poi, inoltre è importante tenere in considerazione i carry in ingresso agli stadi lasciati dal risultato della somma 1, in quanto questi saranno una potenziale fonte di glitch propagandosi anch'essi lungo gli stadi.

In questo caso, la combinazione di carry lasciati dal calcolo precedente e la nuova operazione richiesta (somma 2) fanno si che il sommatore risulti in una condizione per cui, ad ogni colpo di clock, le uscite degli stadi successivi al secondo continueranno a commutare in modo alternato fino a che il carry introdotto dalla seconda operazione non raggiungerà il suddetto stadio, portando così il bit di somma ad un risultato finale.

Si può notare che i primi 3 bit della somma non sono affetti da tale fenomeno, questi infatti non sono soggetti alla propagazione di carry derivanti dalle due somme. È possibile ricostruire una situazione simile per questi primi bit, ponendo in ingresso un susseguirsi di due somme leggermente differenti dalle precedenti:

	Somma 1	Somma 2
Addendo 1	10101010	00000001
Addendo 2	10101010	11111111
Risultato	01010100	00000000
Carry in	01010100	

In tal modo il fenomeno di propagazione dei carry e delle commutazioni delle uscite di somma è esteso anche per i bit meno significativi. Di seguito è riportato un risultato della simulazione ottenuto ponendo tali valori in ingresso al sommatore.



1.3 Sintesi e analisi di potenza di un RCA

È possibile effettuare una stima di potenza avanzata attraverso la piattaforma Synopsys. Partendo infatti dalla descrizione VHDL del RCA fornito, il software è in grado di compiere una sintesi ottimizzata e valutare parametri del circuito basandosi su una libreria di celle caratterizzate dal punto di vista di area, consumi e ritardi.

Per poter stimare la potenza dinamica è necessario conoscere la frequenza di lavoro del circuito, ottenuta valutando il percorso combinatorio con ritardo maggiore: è possibile valutare tale parametro del circuito attraverso un timing report. Il ritardo massimo riportato è pari a 0.78ns, di conseguenza è stato scelto un periodo di clock pari a 1ns.
Successivamente è possibile procedere con l'analisi relativa alla potenza. Il comando power report consente di ottenere una generica stima della potenza dissipata dal circuito, distinguendo tre contributi:

- **leakage**: consumo statico dovuto alle correnti di leakage dei dispositivi utilizzati
- **internal**: contributo alla potenza dinamica totale dovuto alla corrente di corto circuito
- **switching**: contributo alla potenza dinamica dovuto alle commutazioni dei nodi e alle relative cariche/scariche delle capacità

Nella seguente tabella è riportata la suddivisione dei tre contributi.

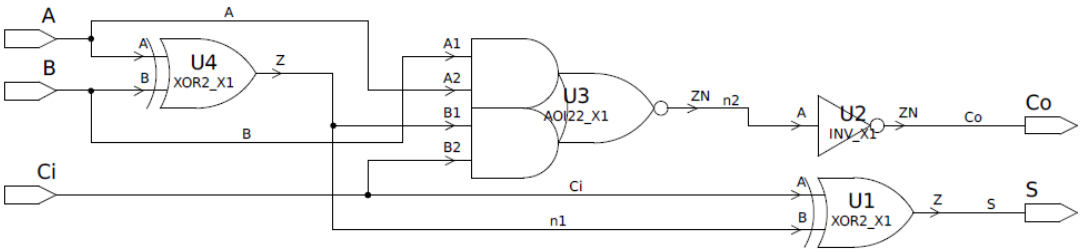
Internal Power	Switching Power	Leakage Power	Total Power
16.7429 uW	9.7406 uW	953.4833 nW	27.4370 uW

L'apporto maggiore al consumo totale è dato dalla potenza dinamica, in particolare dal consumo della potenza interna.
È interessante notare la suddivisione del consumo in potenza per i vari moduli interni (FA) dell'RCA. È possibile ottenere ciò tramite un power report di tipo gerarchico. La distribuzione dei consumi è riportata nella seguente tabella.

Blocco	Switching Power	Internal Power	Leakage Power	Total Power
FA(8)	0.848 uW	2.154 uW	119.291 nW	3.121 uW
FA(7)	1.293 uW	2.150 uW	118.962 nW	3.562 uW
FA(6)	1.332 uW	2.191 uW	119.340 nW	3.642 uW
FA(5)	1.328 uW	2.171 uW	119.110 nW	3.618 uW
FA(4)	1.278 uW	2.101 uW	119.294 nW	3.498 uW
FA(3)	1.263 uW	2.090 uW	118.879 nW	3.471 uW
FA(2)	1.229 uW	2.002 uW	119.508 nW	3.351 uW
FA(1)	1.171 uW	1.884 uW	119.999 nW	3.175 uW
RCA	9.741 uW	16.743 uW	953.483 nW	27.437 uW

Si nota come la suddivisione risulti simile per tutti i FA fatta eccezione per FA(8). In particolare si differenzia dagli altri per il consumo relativo alla potenza di switch. Si ipotizza che quest'ultima risulti inferiore rispetto agli altri full adder a causa del minor carico di uscita.

È inoltre possibile approfondire l'analisi di potenza considerando i contributi delle celle interne ai singoli moduli. In particolare, vengono messi in luce i contributi delle celle che compongono i singoli FA, sempre suddivisi nei tre contributi precedentemente descritti. Per comprendere l'origine del consumo inferiore del FA8, quest'ultimo è stato analizzato come cella ed è stato confrontato con FA1. Lo schema del circuito interno e i risultati ottenuti sono riportati di seguito.



FA(1)

Cella	Switching Power	Internal Power	Leakage Power	Total Dynamic Power
U4	0.5488 uW	0.5899 uW	36.1637 nW	1.138 uW
U3	0.1749 uW	0.3374 uW	32.5747 nW	0.512 uW
U2	0.3964 uW	0.1377 uW	14.2499 nW	0.534 uW
U1	0.0512 uW	0.8205 uW	36.0111 nW	0.872 uW

FA(8)

Cella	Switching Power	Internal Power	Leakage Power	Total Dynamic Power
U4	0.5365 uW	0.5757 uW	36.1637 nW	1.112 uW
U3	0.2155 uW	0.4278 uW	32.7466 nW	0.643 uW
U2	0.0332 uW	0.1774 uW	14.2174 nW	0.211 uW
U1	0.0624 uW	0.9733 uW	36.1631 nW	1.036 uW

La differenza nel consumo di potenza di switching è individuata nella cella U2, legata al carry in uscita dal singolo FA.
È possibile ottenere informazioni aggiuntive sul calcolo della switching power relativa ai nodi interni ad un modulo con un'analisi di tipo net verbose. In tale analisi sono riportati i contributi capacitivi necessari al calcolo di tale potenza, espressi nodo per nodo.

FA(1)

Nodo	Total Net Load	Static Probability	Toggle Rate	Switching Power
n1	4.694 fF	0.493	0.1932	0.5488 uW
n2	2.010 fF	0.488	0.1439	0.1749 uW
S	0.310 fF	0.507	0.2735	0.0512 uW
Cout	4.554 fF	0.512	0.1439	0.3964 uW

FA(8)

Nodo	Total Net Load	Static Probability	Toggle Rate	Switching Power
n1	4.694 fF	0.499	0.1889	0.5365 uW
n2	2.010 fF	0.484	0.1772	0.2155 uW
S	0.310 fF	0.497	0.3328	0.0624 uW
Cout	0.310 fF	0.516	0.1772	0.0332 uW

Si nota come il nodo legato al carry in uscita presenti una capacità inferiore per FA8. Ciò conferma l'ipotesi legata al carico inferiore.
Analizzando i vari FA con analisi di tipo net verbose si nota inoltre che i contributi di capacità relativi ai nodi della somma (S) siano trascurabili rispetto agli altri contributi. Di conseguenza si evince che il contributo di potenza relativo a tale nodo è di un ordine di grandezza inferiore rispetto agli altri.

Da un report di tipo net verbose è inoltre possibile confrontare i valori di static probability e toggle rate dei nodi di uscita di un FA derivanti dalla simulazione con le probabilità in uscita e le attività stimate precedentemente in modo teorico.
Si nota come i valori di probabilità dei nodi di uscita, S e Cout, siano distribuiti attorno al valore 0.5, dunque confrontabili con il caso precedentemente considerato di ingressi equiprobabili.
Il toggle rate rappresenta la frequenza media di commutazione associata al singolo nodo, che identifica, con il suo reciproco, il periodo medio con cui tale nodo presenta una commutazione: attraverso tale valore è possibile stimare la switching activity dei nodi di somma e dei carry in uscita. In tal caso, poichè il periodo di clock è pari a 1ns, il valore di toggle rate riportato coincide con la switching activity del segnale. E' possibile notare come i valori simulati si distanzino da quelli teorici, in quanto all'interno del metodo probabilistico non viene tenuta in considerazione la correlazione spaziale e temporale tra diversi segnali. Tale fatto mette in luce, di conseguenza, le limitazioni che l'utilizzo di un modello probabilistico comporta.

CONFRONTO VERBOSE RCA

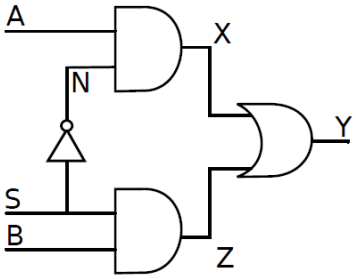
E' stato richiesto inoltre un power report di tipo net verbose relativo all'intero RCA. Si può osservare come i contributi capacitivi relativi ai nodi delle uscite di somma siano significativamente inferiori rispetto ai contributi dei nodi relativi ai carry tra uno stadio e quello successivo. Anche in questo caso la motivazione è dovuta al carico capacitivo inferiore delle uscite di somma, che non risultano collegate ad ulteriori blocchi. Tuttavia, osservando i contributi totali riportati dall'analisi di tipo verbose, è possibile osservare una differenza tra questi e quelli presenti nel power report precedentemente richiesto.

	Internal Power	Switching Power	Leakage Power	Total Power
Power report	16.7429 uW	9.7406 uW	953.4833 nW	27.4370 uW
Net Verbose	8.848 uW	3.766 uW	403.098 nW	12.614 uW

I contributi riportati dall'analisi di tipo net verbose risultano proporzionalmente inferiori. La possibile causa di ciò è da attribuirsi al tipo di analisi effettuata da Synopsys: in particolare la keyword -net utilizzata ha richiesto uno studio della potenza relativo ai nodi interni dell'architettura dell'RCA, presenti tra i vari FA connessi. In tale analisi i contributi dei nodi interni ai FA stessi sono stati tralasciati. Al contrario un'analisi di tipo power report fornisce un risultato completo comprendendo all'interno l'analisi di tutti i nodi dei blocchi che compongono l'architettura.

1.4 MUX: generazione e propagazione di glitch

L'obiettivo di questo esercizio è lo studio delle conseguenze introdotte dai ritardi delle porte, in particolare all'interno del multiplexer rappresentato nella seguente figura.

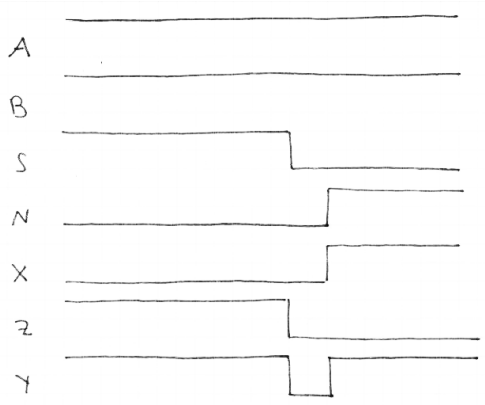


In questo caso particolare tutte le porte sono esenti da ritardi fatta eccezione per l'inverter, caratterizzato da un ritardo di propagazione pari a 0.1 ns.

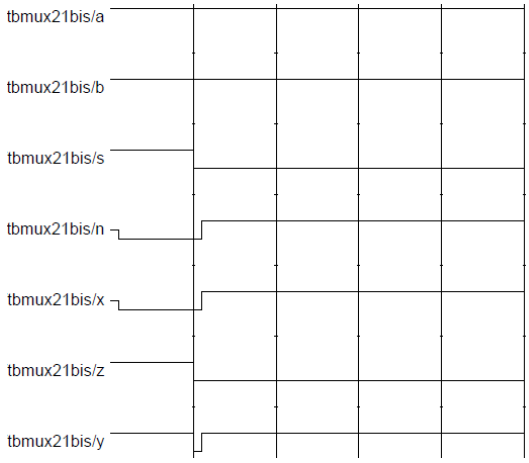
All'interno del file *tb_mux21_glitch.vhd* è stato possibile identificare la combinazione dei segnali di ingresso con i quali il multiplexer è stato testato.

```
A <= '1';
B <= '1';
S <= '1', '0' after 1 ns;
```

In particolare, è possibile notare dal codice VHDL sopra riportato che inizialmente gli ingressi assumono tutti un valore logico alto. Dopo 1ns il segnale S commuta. L'uscita, in un caso ideale, non dovrebbe presentare commutazioni.
Tuttavia, si ipotizza che, a causa del ritardo di propagazione introdotto dall' inverter, vi sia un intervallo temporale in cui i nodi interni X e Z assumono entrambi un valore logico basso, portando quindi l'uscita Y a 0. Tale comportamento atteso è mostrato nella seguente figura.



Per mezzo di una simulazione ModelSim è stato possibile osservare attraverso le waveforms il comportamento reale dei segnali. Il risultato di tale simulazione è riportato nella seguente immagine.



È possibile visualizzare direttamente sulla waveform dell'uscita Y il glitch causato dall'inverter. Tale comportamento è in linea con quanto atteso.

Al fine di analizzare altre possibili combinazioni degli ingressi che possano generare un glitch in uscita è stata realizzata una mappa di Karnaugh rappresentante la funzione logica del multiplexer.

S \ A B	00	01	11	10
0	0	0	1	1
1	0	1	1	0

In particolare, è possibile notare che il circuito precedentemente riportato sia l'implementazione della funzione logica coperta da due implicanti, rappresentata nella figura precedente.
Il glitch analizzato è dovuto ad una transizione degli ingressi che consegue in un passaggio tra un implicante e l'altro. Tale problema potrebbe essere risolto inserendo, in modo ridondante, un terzo implicante per evitare la transizione precedentemente trattata.

S \ A B	00	01	11	10
0	0	0	1	1
1	0	1	1	0

Tale ragionamento porta alla conclusione che l'unica combinazione in grado di causare un glitch in uscita sia quella presa in analisi.

Essendo i glitch associati a commutazioni di nodi, questi portano un contributo aggiuntivo al consumo totale di potenza dinamica. In particolare l'energia sprecata durante queste commutazioni spurie è la somma dei consumi durante le due transizioni del segnale. Ciascuna delle due contribuisce all'energia secondo la seguente equazione:

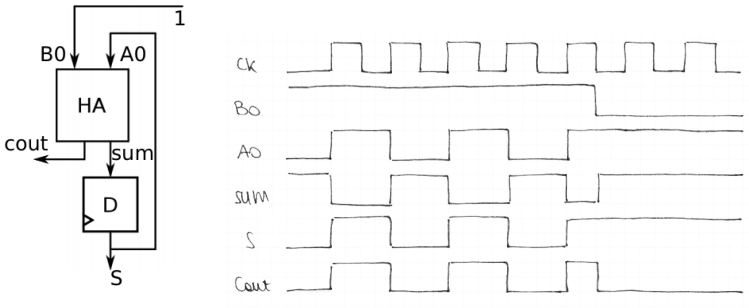
$$E = C \cdot V^2$$

dove C rappresenta la capacità di carico e V la tensione a cui viene caricata tale capacità. Metà di tale contributo di energia è usata per caricare o scaricare la capacità di carico associata all'uscita, mentre la restante parte viene dissipata. Di conseguenza il consumo di energia totale associato ad un glitch è dato da:

$$E = 2 \cdot C \cdot V^2$$

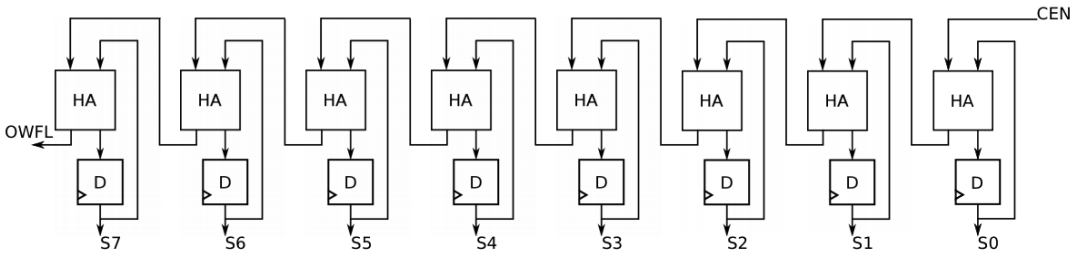
1.5 Calcolo di probabilità e attività: contatore sincrono

Nella prima parte di questo esercizio è stato analizzato il timing del blocco rappresentato nella figura seguente.

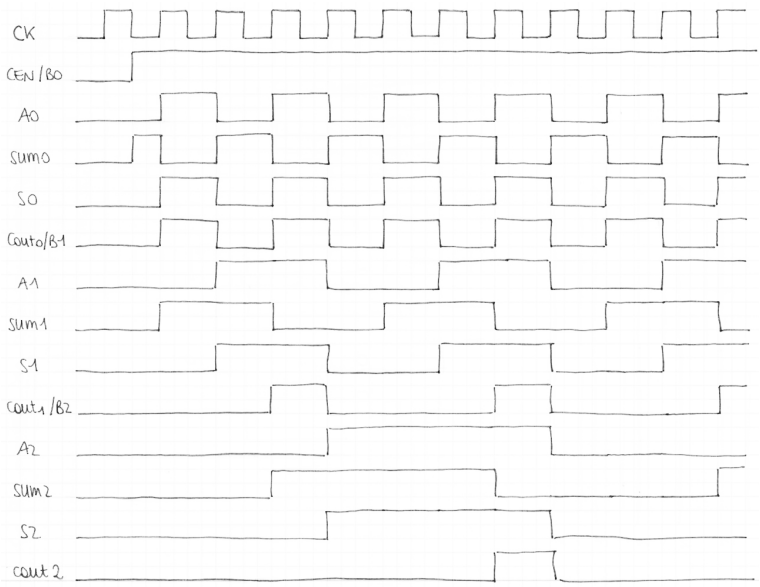


In particolare è stato possibile stabilire il suo comportamento atteso. Quando il segnale B0 è asserito, l'uscita S presenta una commutazione per ogni fronte sensibile del clock. Al contrario, quando B0 non è attivo, l'uscita non presenta alcuna commutazione.

È possibile utilizzare tali strutture per realizzare un contatore sincrono, come mostrato in figura. In particolare, i segnali di CEN e OVFL rappresentano rispettivamente l'enable del contatore e il terminal count di questo.



Considerando infatti il comportamento delle uscite S0, S1 e S2 quando il segnale di enable è attivo si ottiene il seguente timing.



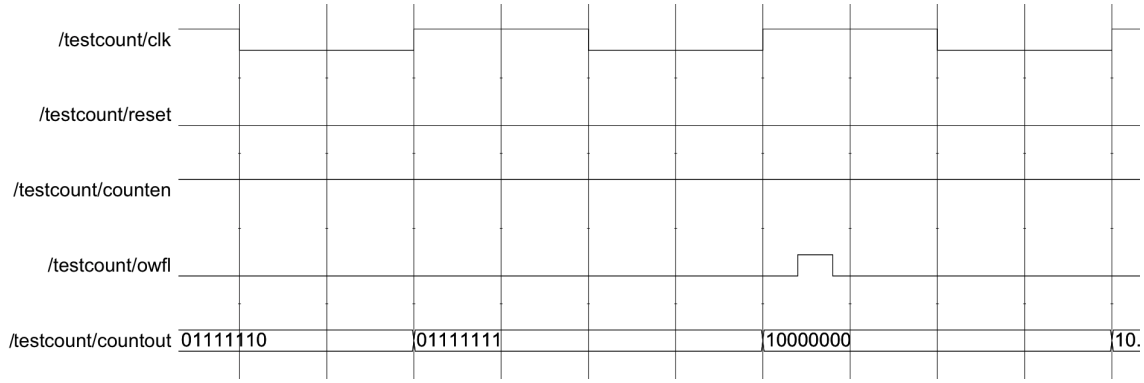
Dalle considerazioni fatte in precedenza è possibile stimare il numero di commutazioni delle 8 uscite e del terminal count durante un intero ciclo di conta. Inoltre, nella tabella seguente sono stati aggiunti i dati ottenuti tramite power report, ottenuto attraverso una simulazione ModelSim. Il clock utilizzato per la simulazione ha un periodo di 2ns.

Segnale	Transizioni stimate	Transizioni simulate
S0	255	257
S1	127	128
S2	63	64
S3	31	32
S4	15	16
S5	7	8
S6	3	4
S7	1	2
OVFL	1	16

Si può notare che, senza considerare il segnale OVFL, i risultati ottenuti in entrambi i casi siano confrontabili, tuttavia è presente una leggera differenza, pari a 1 o 2 colpi di clock, dovuta al fatto che è stato scelto un tempo di simulazione leggermente superiore al tempo di conta totale. Riguardo al segnale di OVFL, il risultato simulato è distante da quello atteso, in quanto il segnale presenta un numero di commutazioni decisamente maggiore. Tale segnale, al contrario di quelli di uscita della somma (rappresentanti il conteggio), non presenta un elemento sequenziale che lo interfacci con l'esterno. Per tale motivo sono ben visibili tutte le commutazioni spurie del segnale stesso, dovute ai ritardi di propagazione definiti all'interno dei vari stati del contatore.

Dalla simulazione è stato inoltre possibile osservare l'andamento delle waveforms delle uscite,

riportato nella figura sottostante.



Si nota che, in un preciso intervallo temporale, il segnale OVFL assume un valore logico alto durante il conteggio, comportamento inatteso da parte di un contatore. Tale glitch avviene in seguito alla commutazione del bit più significativo (S7); infatti, per un certo intervallo di tempo, in ingresso all'half adder del blocco 7 sono presenti due ingressi con valore logico alto: uno dovuto al carry in uscita dal blocco precedente che, a causa dei ritardi, non ha ancora commutato il suo valore, e l'altro relativo all'uscita attuale S7.

Successivamente, sempre attraverso un power report, è stato possibile analizzare l'attività dei segnali interni al contatore. I risultati sono riportati nella seguenti tabelle.

Segnale	Commutazioni	Attività
SUM(0)	258	0.992
SUM(1)	385	1.377
SUM(2)	320	1.231
SUM(3)	224	0.862
SUM(4)	144	0.554
SUM(5)	88	0.338
SUM(6)	52	0.200
SUM(7)	30	0.115

Segnale	Commutazioni	Attività
Cout(0)	257	0.988
Cout(1)	256	0.985
Cout(2)	192	0.738
Cout(3)	128	0.492
Cout(4)	80	0.308
Cout(5)	48	0.185
Cout(6)	28	0.108
Cout(7)	16	0.062

È possibile notare come l'attività delle somme in uscita dagli half adder sia differente rispetto alle uscite sincrone dei flip-flop. Infatti, a causa dei ritardi di propagazione dei carry in uscita dai singoli stadi, si generano una serie di glitch, che, propagandosi lungo la serie di half adder, causano commutazioni multiple dei nodi SUM.

Similmente all'analisi effettuata per il blocco 7 è possibile ipotizzare comportamenti simili legati al carry in uscita anche per gli altri blocchi. Tale vettore di carry presenta infatti un'attività complessiva maggiore rispetto a quella stimata nel caso senza ritardi.

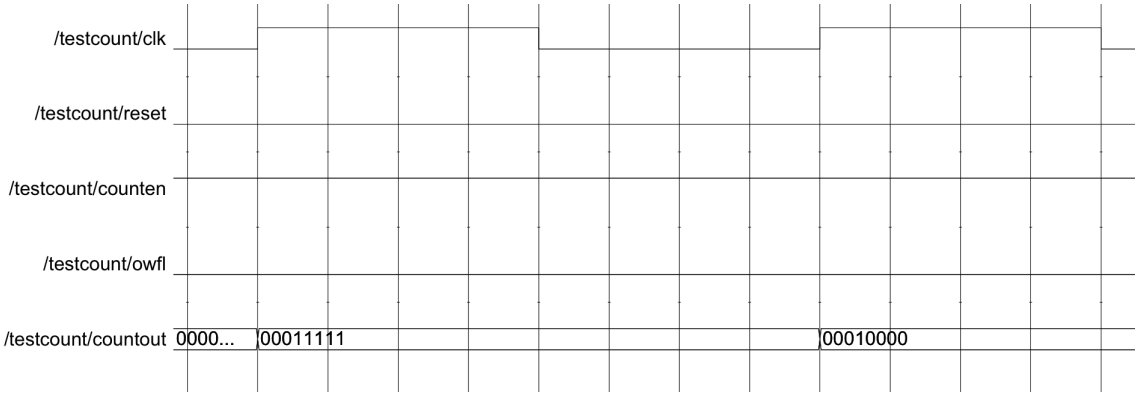
Nella simulazione precedentemente effettuata i ritardi sono stati modellizzati in modo tale da

presentare un valore significativamente inferiore rispetto al periodo di clock.

$$Periodo = 2ns \qquad Ritardo\_somma = 0.2ns \qquad Ritardo\_carry = 0.2ns$$

Si nota che in questo caso il periodo di clock è sufficientemente elevato per garantire un corretto funzionamento del contatore: la propagazione del carry out può avvenire correttamente in tutti gli stadi.

Riducendo il periodo di clock a 0.8ns è possibile osservare un comportamento inatteso, rappresentato in figura, da parte del contatore.



L'immagine mette in luce una transizione da un conteggio all'altro:

$$00011111 \rightarrow 00010000$$

Tale errore è dovuto alla mancata completa propagazione del carry in tutti gli stadi, infatti il clock campiona le uscite dei flip-flops prima che tutti gli stadi abbiano raggiunto un valore stabile.

La presenza di glitch all'interno del contatore in analisi comporta un incremento dei consumi rispetto al caso ideale. Una possibile soluzione, che potrebbe essere adottata per garantire un numero inferiore di commutazioni spurie modificando il VHDL, è legata all'aumento del ritardo coinvolto nel calcolo della somma per i singoli HA. Se tale tempo fosse infatti maggiore o uguale del massimo tempo di propagazione del carry attraverso tutti gli stadi, le uscite SUM non presenterebbero più un elevato numero di glitch.

FORSE CI STAREBBE VERIFICARE CON UNA SIMULAZIONE E NEL CASO METTERE I RISULTATI

COSE DA FARE:
- Capire il toggle rate nel verbose (ESW)IPOTESI FATTE
- Capire l'incongruenza del verbose RCA FATTO
- Spiegare il worst case del sommatore FATTO
- Fare mappa di Karnaugh FATTO
- Plot segnali MUX (fatto da noi) FATTO
- Plot dei segnali del blocchetto contatore e dei primi bit del contatore (fatto da noi) FATTO

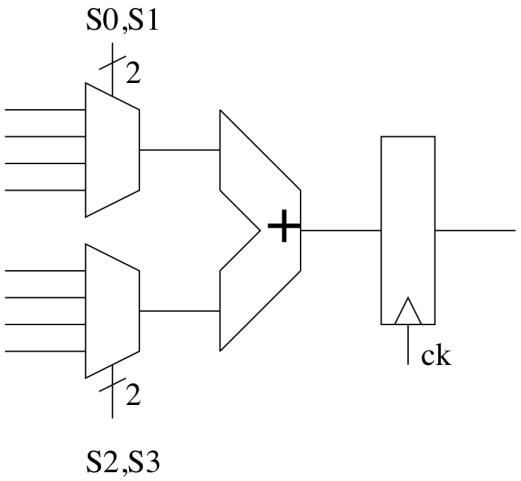
- Completare le tabelle FATTO
- Revisione grafica FATTO

Laboratorio 2 - FSM Assignment and VHDL Synthesis

2.1 FSM State Assignment

2.1.1 Progetto

L'obiettivo del progetto è la definizione di una macchina a stati al fine di realizzare un algoritmo di somma tra 6 input, utilizzando il datapath in figura seguente e scegliendo l'ordine con cui collegare i diversi ingressi ai multiplexer.



In particolare, al fine di minimizzare i consumi della macchina a stati, è necessario ridurre al minimo il numero di commutazioni di:

- variabili di stato
- segnali di controllo

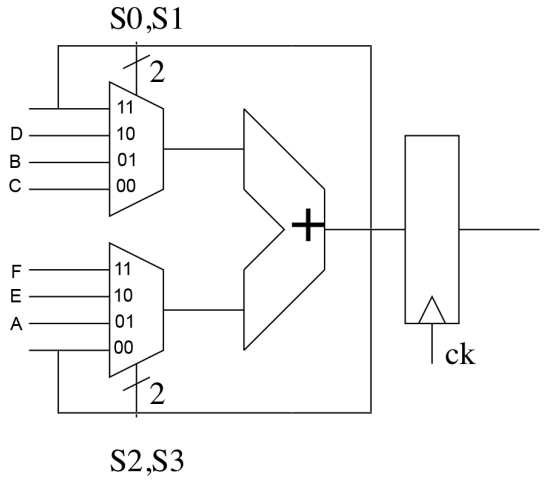
Essendo i segnali di controllo coincidenti con i selettori dei multiplexer, è stata innanzitutto stabilita una sequenza per la selezione dei diversi input tale per cui il numero di commutazioni durante l'esecuzione dell'intero algoritmo di somma fosse minimo.

Operazione	S3	S2	S1	S0
A+B	0	1	0	1
S+C	0	0	0	0
S+D	0	0	1	0
E+S	1	0	1	1
F+S	1	1	1	1

È possibile notare come sia necessario riportare il valore della somma, denominato S, in ingresso ad entrambi i multiplexer.
Nella transizione da uno stato all'altro il numero di commutazioni coincide con la variazione degli addendi; ne consegue che il massimo numero di commutazioni durante una transizione risulta pari 2.
Considerando un ciclo di esecuzione dell'algoritmo, è possibile stimare il numero totale di commutazioni sui segnali di selezione e la loro switching activity.

	S3	S2	S1	S0
Commutazioni in un ciclo	2	2	2	2
Attività	0.4	0.4	0.4	0.4

Il numero di commutazioni totale risulta quindi pari a 8.
È stata scelta dunque la seguente configurazione degli ingressi



Il passo successivo è la scelta della codifica dei singoli stati. In tale contesto è importante tenere in considerazione che una generica FSM presenta una rete per la generazione degli output e una rete per il calcolo dello stato futuro.

Una codifica ottimale punta alla minimizzazione delle commutazioni in entrambe le reti. Per tale scopo si è deciso di far coincidere le due reti, in modo che la logica utilizzata per generare le

uscite possa essere sfruttata anche per la definizione dello stato futuro. In particolare, la codifica dello stato futuro si ottiene a partire dalle uscite di controllo dello stato attuale.

Operazione	Codifica dello stato	S3	S2	S1	S0
A+B	1 1 1	0	1	0	1
S+C	1 0 1	0	0	0	0
S+D	0 0 0	0	0	1	0
E+S	0 1 0	1	0	1	1
F+S	0 1 1	1	1	1	1

È possibile notare come tutte le transizioni presentino una distanza di Hamming pari a 1, fatta eccezione della transizione da S+C a S+D, nella quale tale distanza è pari a 2.
In tale contesto, la sola logica combinatoria necessaria è relativa alla valutazione delle uscite a partire dalla codifica dello stato attuale.

Si è notato che tali funzioni logiche presentano un'implementazione semplice con un numero limitato di porte. Di seguito sono riportate le mappe di Karnaugh per le funzioni logiche sopra descritte.

S3

C2,C1 / C0	00	01	11	10
0	0	1	-	-
1	-	1	0	0

S2

C2,C1 / C0	00	01	11	10
0	0	0	-	-
1	-	1	1	0

S1

C2,C1 / C0	00	01	11	10
0	1	1	-	-
1	-	1	1	0

S0

C2,C1 / C0	00	01	11	10
0	0	1	-	-
1	-	1	1	0

È possibile notare che le funzioni logiche presentano implicanti in comune, semplificando maggiormente la logica necessaria.

2.1.2 Descrizione VHDL e simulazione Modelsim

La macchina a stati precedentemente definita è stata descritta attraverso il VHDL, specificando in particolare la codifica degli stati progettata.

DOBBIAMO METTERE IL VHDL? fare eventuale appendice

Al fine della simulazione, datapath e FSM sono stati inclusi in un'unica entity per ottenere il Processing Element complessivo. È stato dunque realizzato un testbench per verificare il corretto susseguirsi degli stati e le relative transizioni dei segnali. In particolare, è stata effettuata una simulazione di 2000 colpi di clock, che ha fornito i seguenti risultati attraverso un power report.

Segnale	Tc	Activity
Clock	4000	2
S3	799	0.3995
S2	802	0.4010
S1	799	0.3995
S0	802	0.4010

Le attività ottenute sono comparabili con quelle stimate in fase di progetto.

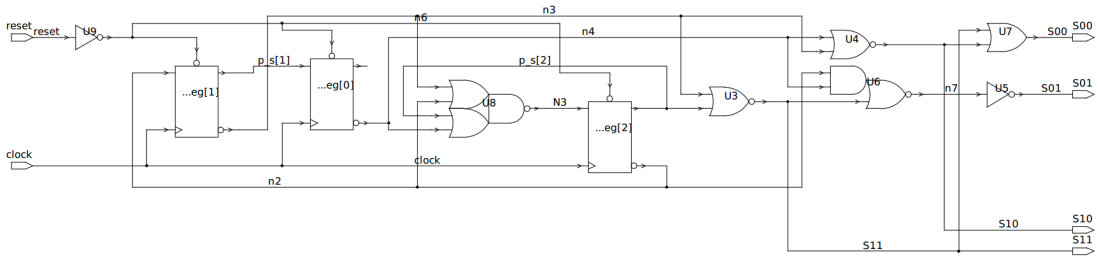
2.2 Sintesi del VHDL

In tale sezione è stata effettuata la sintesi della top level entity che comprende sia FSM sia datapath.
Inizialamente i file sono stati analizzati nel corretto ordine ed è stata elaborata la struttura del Processing Element.
Successivamente è stato generato un clock di periodo pari a 10 ns.

create_clock -name "CLK" -period 10 CLK

È stata verificata la presenza del clock generato attraverso il comando **report_clock**.
Successivamente è stata effettuata la sintesi attraverso la direttiva **compile -exact map**.

In seguito è stato possibile visualizzare, in modo gerarchico, la mappatura del circuito con la libreria scelta. Nell'immagine seguente, in particolare, è riportato lo schematico della macchina a stati.



È possibile notare, come atteso, che il numero di porte logiche utilizzato risulta limitato.
Sono stati effettuati una serie di report inizialmente sulla top level entity per poi analizzare in modo più approfondito la macchina a stati.
In primo luogo è stata ottenuta una stima dell'area delle celle utilizzate nella sintesi, attraverso il comando **report_area**.

Total cell area = 295.53 um2

Successivamente, al fine di valutare lo slack, sono stati analizzati i 10 percorsi combinatori con ritardi maggiori.

report_timing -nworst 10

Per semplicità sono riportati i dati relativi al critical path.

Clock period	10 ns
Data required time	9.97 ns
Data arrival time	1.94 ns
Slack	8.02 ns

Come è possibile notare, lo slack risulta particolarmente elevato se confrontato con il periodo di clock. Nel report sono inoltre presenti i ritardi relativi agli elementi compresi tra lo start-point e l'end-point del percorso critico.

È possibile ottenere informazioni riguardo alla potenza dissipata attraverso il comando **report_power**. In particolare, si evidenziano i tre contributi relativi alla top level entity.

Internal Power	29.4711 uW
Switching Power	13.2924 uW
Leakage Power	5.4747 uW
Total Power	48.2382 uW

Sono stati ottenuti dati aggiuntivi relativi al consumo di potenza dei singoli moduli dell'architettura, tramite l'aggiunta della direttiva **-hier**.

FSM	4.504 uW	9.3%
Datapath	43.734 uW	90.7%

Il consumo della macchina a stati rappresenta una percentuale ridotta rispetto al valore totale. Al fine di approfondire lo studio relativo ai nodi interni del circuito, è stata realizzata un'analisi di tipo **-net**. In particolare sono riportati i risultati relativi ai nodi dei segnali di selezione dei multiplexer.

Segnale	Net Load [fF]	Static Probability	Toggle Rate	Switching Power [uW]
sel11_wire	13.792	0.120	0.0246	0.2055
sel01_wire	10.222	0.253	0.0272	0.1682
sel10_wire	8.489	0.720	0.0301	0.1547
sel00_wire	7.058	0.720	0.0301	0.1286

Il periodo medio di variazione dei selettori, ottenibile dal toggle rate, è confrontabile con le variazioni medie che avvengono su tali segnali in un ciclo di esecuzione. È importante tenere in considerazione la possibile presenza di commutazioni non previste a causa dell'attivazione del segnale di reset durante la simulazione, il quale è caratterizzato da una probabilità non nulla di commutazione.

È possibile imporre dei vincoli sui vari parametri coinvolti nel processo di sintesi. In particolare, è stato generato un segnale di clock alla massima frequenza consentita dal circuito. La stima di tale frequenza è stata effettuata a partire dall'analisi del critical path precedentemente riportato.

Clock period	1.94 ns
Data required time	1.91 ns
Data arrival time	1.91 ns
Slack	0 ns

Come atteso, lo slack relativo al percorso critico risulta essere nullo, di conseguenza il circuito opera alla massima frequenza supportata. Dal momento che la potenza dissipata è funzione della frequenza di lavoro, è stato riscontrato un aumento dei consumi.

Internal Power	151.3501 uW
Switching Power	69.4127 uW
Leakage Power	5.5735 uW
Total Power	226.3363 uW

A seguito di un incremento della frequenza di un fattore di circa 5, la potenza è aumentata proporzionalmente.

È inoltre possibile imporre un vincolo sul contributo totale di potenza dinamica, attraverso la direttiva

set_max_dynamic_power

Tale valore è stato fissato a 40 uW, leggermente inferiore rispetto alla potenza dinamica ottenuta nelle prime simulazioni.
In particolare, è stato considerato il clock di partenza di periodo pari a 10 ns. Di seguito sono riportati i risultati ottenuti.

	No constraints	Power constraints
Internal Power	29.4711 uW	33.8116 uW
Switching Power	13.2924 uW	13.3092 uW
Total Dynamic Power	42.76 uW	47.12 uW

I risultati ottenuti dalla sintesi non risultano in linea con quanto atteso, infatti è registrato un aumento del consumo di potenza dinamica.
Si ipotizza che il consumo di potenza dinamica ottenuto nella prima sintesi fosse già prossimo al minimo globale.

Successivamente è stata effettuata un'analisi più dettagliata della FSM accedendovi tramite la direttiva **current_instance FSM**.
Attraverso il comando **report_fsm** è stato possibile verificare la corretta implementazione della codifica progettata.

È possibile risalire alla suddivisione del consumo della potenza dinamica all'interno delle celle che costituiscono la FSM attraverso un'analisi di potenza di tipo **-cell**. In particolare, nella seguente tabella vengono riportati i tre contributi maggiori relativi al consumo di potenza dinamica.

ps_reg[1]	1.143 uW
ps_reg[2]	0.979 uW
ps_reg[0]	0.924 uW

Si nota che tali contributi sono relativi alle variabili di stato, che presentano numerose commutazioni durante l'esecuzione dell'algoritmo.
Uno studio più approfondito sulla switching power relativa ai nodi interni è ottenuta tramite un'analisi di tipo **-net**. In particolare, nella seguente tabella, sono riportati i dati relativi ai segnali di uscita della FSM, ovvero i segnali di selezione dei multiplexer.

Segnale	Net Load [fF]	Static Probability	Toggle Rate	Switching Power [uW]
S11	13.792	0.120	0.0246	0.2055
S01	10.222	0.253	0.0272	0.1682
S10	8.489	0.720	0.0301	0.1547
S00	7.058	0.720	0.0301	0.1286

È possibile notare che tali valori coincidono con quelli ottenuti precedentemente durante l'analisi -net della top level entity. Il contributo maggiore del consumo è dovuto alle capacità di carico dei nodi in analisi, che assumono valori significativi rispetto agli altri nodi. Questo è il principale motivo per cui tali segnali contribuiscono maggiormente, data la parità di toggle rate.

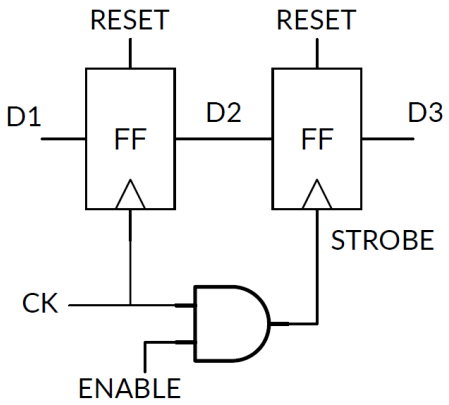
Per ragioni di praticità, tali sintesi sono state effettuate raggruppando i comandi e le direttive in uno script in modo darendere il processo automatico. Di seguito è riportato uno degli script utilizzati.

```
gui_start
analyze -library WORK -format vhdl
{/home/lp20.9/Desktop/low_power_2020/Lab_2/codifica_3bit_bis/syn/dpadder.vhd
/home/lp20.9/Desktop/low_power_2020/Lab_2/codifica_3bit_bis/syn/fsm1.vhd
/home/lp20.9/Desktop/low_power_2020/Lab_2/codifica_3bit_bis/syn/FSM_Adder.vhd}
elaborate FSM_ADDER -architecture BEHAVIOR -library WORK
change_selection -name __slctBus18 -type {cell design} _sel426
create_clock -name "CLK" -period 10 {CLK}
compile -exact_map
report_timing -nworst 10 > report/simple_clk_10/report_clock.txt
report_area > report/simple_clk_10/report_area.txt
report_power > report/simple_clk_10/report_power.txt
report_power -hier > report/simple_clk_10/report_power_hier.txt
report_power -net > report/simple_clk_10/report_power_net.txt
current_instance FSM
report_power -hier > report/simple_clk_10/report_power_hier_FSM.txt
report_power -net > report/simple_clk_10/report_power_net_FSM.txt
report_fsm > report/simple_clk_10/report_fsm_encoding.txt
```

Laboratorio 3 - Clock gating e soluzioni architetturali

3.1 Un primo approccio al clock gating

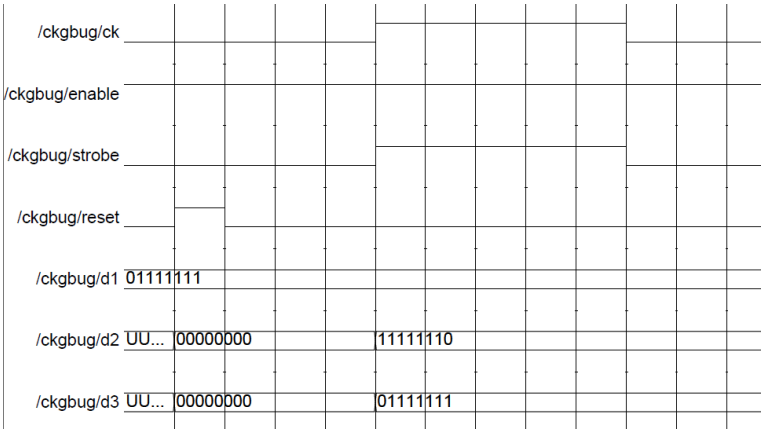
Il clock gating è una tecnica di riduzione dei consumi a livello architetturale, il cui scopo è impedire la commutazione del segnale di clock in determinate parti del circuito non utilizzate.
È fornita una descrizione VHDL del seguente circuito.



Dal file VHDL è possibile notare il valore assegnato ai segnali D1, D2 e D3:

```
signal D1:      std_logic_vector(byte-1 downto 0) := ('0', others => '1'); -- 01111111
signal D2:      std_logic_vector(0 to byte-1) := (others => '0'); -- 00000000
signal D3:      std_logic_vector(byte-1 downto 0) := (others => '1'); -- 1111111
```

Essendo il segnale di clock enable sempre attivo, è atteso che l'uscita D3 assuma un valore pari a D1 dopo due cicli di clock. Tale comportamento non è verificato da una prima simulazione ModelSim effettuata.
Come possibile notare dalla seguente figura, al primo fronte attivo di clock successivo al reset, il valore di D1 è già riportato in uscita al flip-flop controllato dal segnale di strobe.

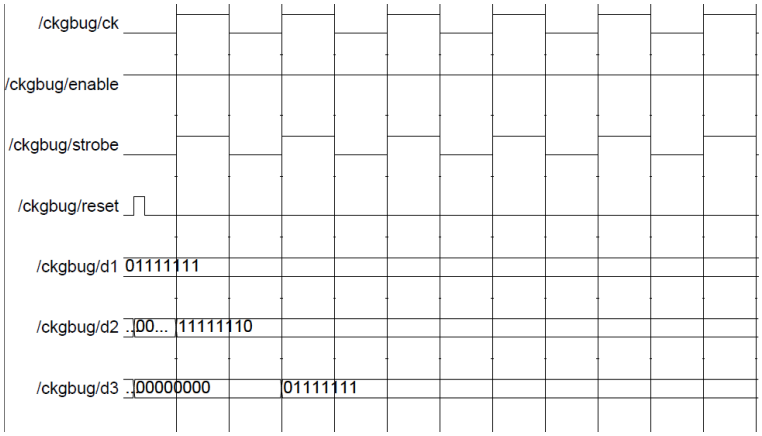


Ciò accade poiché il simulatore, in assenza di informazioni sui ritardi, deve comunque simulare il circuito mantenendo una causalità nella propagazione dei segnali all'interno degli elementi descritti.
Come conseguenza, l'informazione relativa al clock, a causa della presenza di una porta AND, è riportata all'ingresso del secondo flip-flop uno step simuazione dopo rispetto al primo. Tale comportamento è deleterio in quanto altera il timing previsto del circuito.

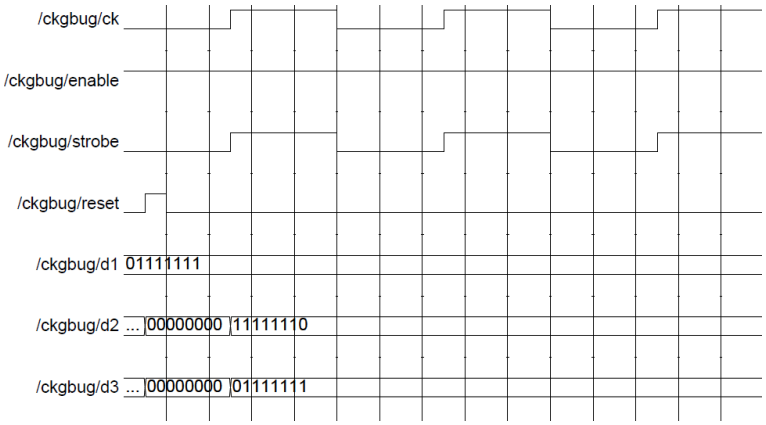
Per assicurare il corretto funzionamento del circuito, è sufficiente indicare la presenza di un ritardo CK $\rightarrow$ Q per entrambi i flip-flop. A tale scopo è stato modificato il file **fd.vhd**.

```
architecture PLUTO of FD is -- flip flop D con reset ASINCRONO

begin

        PASYNCH: process(CK,RESET)
        begin
          if RESET='1' then
            Q <= '0';
          elsif CK'event and CK='1' then -- positive edge triggered:
            Q <= D after 0.1 ps; -- copia l'ingresso sull'uscita
          end if;
        end process;

end PLUTO;
```

Il ritardo aggiunto sulla propagazione dell'uscita permette di raggiungere il risultato atteso dalla simulazione.



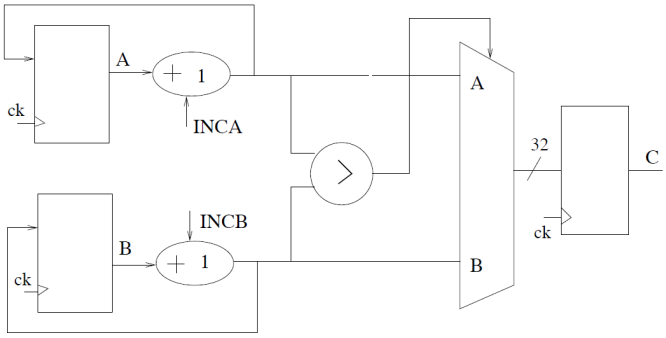
Se la porta AND è caratterizzata da un ritardo maggiore rispetto al ritardo CK $\rightarrow$ Q precedentemente inserito, il valore campionato in uscita non è quello corretto. In particolare, l'errore riscontrato è analogo a prima, come mostrato nella seguente figura.



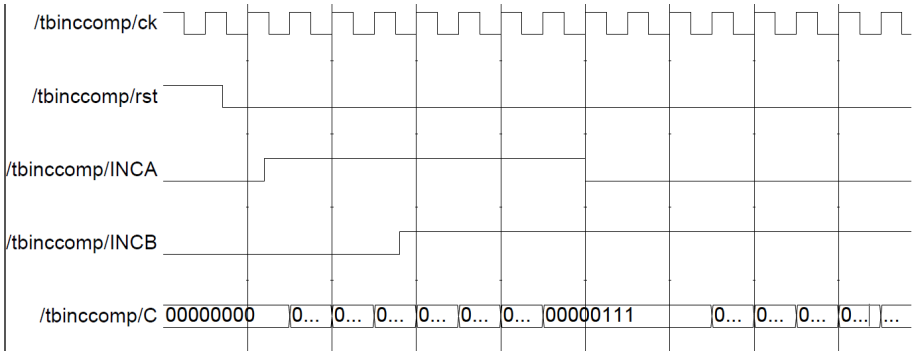
3.2 Clock gating per un circuito complesso

3.2.1 Simulazione del circuito

All'interno del file **inccomp.vhd** è presente una descrizione del circuito mostrato nella figura seguente.

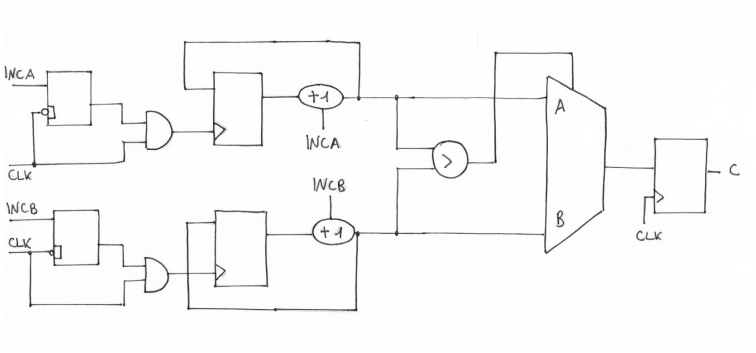


Dal codice si evince che il circuito è in grado di incrementare e confrontare due valori, rappresentati su 8 bit, contenuti nei registri A e B, e portare il maggiore di essi in uscita. In particolare, l'operazione di incremento è gestita attraverso INCA e INCB, due segnali attivi alti. È stato realizzato un testbench con lo scopo di simulare e verificare il comportamento del circuito.



Come possibile notare dalle forme d'onda sopra riportate, il comportamento è in linea con quanto atteso.
Infatti, il segnale INCA viene campionato per tre colpi di clock prima che INCB risulti attivo, ciò porta il valore contenuto in REG_A uguale a 3. In seguito, asserendo INCB, anche il valore contenuto in REG_B inizia ad aumentare. Nei cicli di clock in cui entrambi INCA e INCB sono asseriti, la differenza tra i due dati rimane costante. Nel momento in cui INCA viene negato, il valore in uscita dal circuito rimane costante fino a che il dato contenuto in REG_B risulti maggiore: questo accade esattamente dopo tre cicli di clock.

Nel caso in cui i segnali di incremento non siano asseriti, non è necessario far commutare il clock all'interno dei registri, in quanto nessun valore deve essere aggiornato; a tal proposito, è possibile sfruttare la tecnica del clock gating. Una possibile implementazione è mostrata nella seguente figura.



È possibile notare la presenza di due latch atti a filtrare eventuali glitch presenti sugli ingressi INCA e INCB.

3.2.2 Sintesi del circuito

Inizialmente è stata effettuata una sintesi in assenza di direttive per l'implementazione del clock gating. Successivamente alla generazione di un clock di periodo pari a 5 ns, è stato effettuato un power report includendo anche il contributo dei nodi relativi agli ingressi attraverso la direttiva **-include_input_net**.

Power report - no clock gating, statistica di default

Internal power [uW]	Net switching power [uW]	Leakage power [uW]	Total power [uW]
38.7514	13.6944	3.7969	56.2428

Un'analisi di potenza di tipo **-net** permette di ottenere informazioni più dettagliate sulla statistica dei segnali e sulle capacità coinvolte.
In particolare, si nota che le capacità relative ai nodi di ingresso assumono un valore significativo, di conseguenza è importante tenere in considerazione tali nodi in una stima di potenza.
Sono riportate le informazioni statistiche relative ai segnali di clock, reset, INCA e INCB.

Segnale	Static probability	Toggle rate
clock	0.5	0.4
reset	0.5	0.02
INCA	0.5	0.02
INCB	0.5	0.02

Tali valori non sono realistici rispetto ai segnali presi in considerazione.
È possibile effettuare stime più accurate comunicando al sintetizzatore i dati statistici relativi ai segnali. In particolare, è stata inizialmente modificata la statistica del segnale di reset, impostando la static probability pari a 0.

Power report - no clock gating, reset modificato

Internal power [uW]	Net switching power [uW]	Leakage power [uW]	Total power [uW]
27.6360	14.8140	4.18	46.6302

In seguito, impostando le statistiche relative ai segnali INCA e INCB con static probability pari a 0.12 e toggle rate pari a 0.025, come specificato nella consegna, si ottengono i seguenti risultati.

Power report - no clock gating, reset e ingressi modificati

Internal power [uW]	Net switching power [uW]	Leakage power [uW]	Total power [uW]
21.5273	10.8718	4.0921	36.4911

Si nota una sensibile riduzione dei consumi dovuti al minor numero di commutazioni medie dei segnali in ingresso.
Per ottenere il numero totale di celle utilizzato dal sintetizzatore, è stata realizzata un'analisi di tipo **-cell**.

Numero celle	Area [um^2]
74	229.29

Attraverso una particolare descrizione VHDL, è possibile notificare al sintetizzatore la possibilità di implementare la tecnica del clock gating, che può essere richiesta nella fase di sintesi attraverso la direttiva **-gate_clock**. Utilizzando sempre un clock di periodo 5 ns, sono stati ottenuti i seguenti risultati, includendo sempre i contributi dovuti ai nodi di ingresso.

Power report - clock gating, statistiche di default

Internal power [uW]	Net switching power [uW]	Leakage power [uW]	Total power [uW]
32.9555	10.8986	3.9349	47.7882

Come atteso, il clock gating ha portato ad una riduzione della potenza generale rispetto al caso in cui tale tecnica non è stata implementata. In particolare, è possibile notare una riduzione della potenza dinamica e un leggero aumento della potenza di leakage: si ipotizza che tale incremento sia dovuto ad un aumento del numero di celle.
Successivamente, come effettuato in precedenza, sono state modificate le statistiche relative ai segnali di reset, INCA e INCB.

Power report - clock gating, reset e ingressi modificati

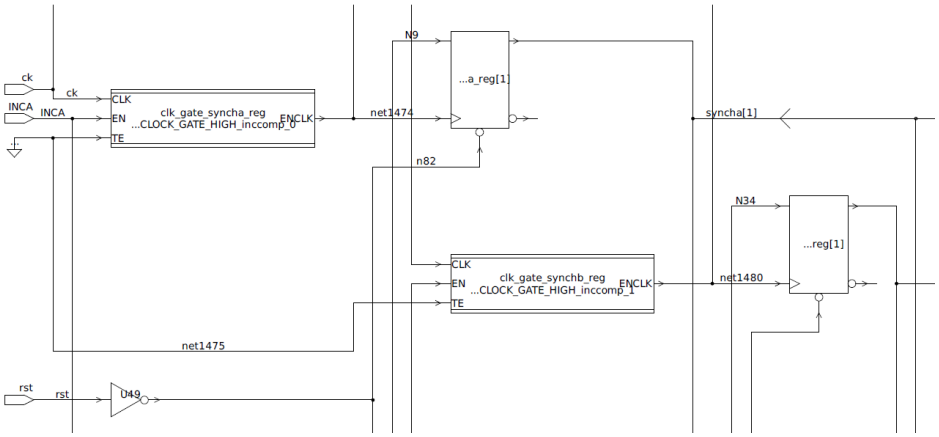
Internal power [uW]	Net switching power [uW]	Leakage power [uW]	Total power [uW]
13.3592	5.8499	4.2570	23.4661

L'effetto del clock gating risulta più marcato adottando una statistica più realistica dei segnali presi in considerazione.

Tuttavia, tale implementazione richiede l'impiego di un numero maggiore di celle atte all'inserimento della logica necessaria alla gestione del clock gating.

Numero celle	Area [um^2]
78	238.07

È possibile navigare all'interno dello schematico al fine di individuare le celle relative all'implementazione di tale tecnica.



Si nota la presenza di due strutture aggiuntive che controllano il clock in ingresso ai due registri REG_A e REG_B. In particolare, è possibile visualizzare l'interno di tali strutture, costituite da un latch attivo basso seguito da una porta AND.

Si nota che non è presente nessuna struttura adibita al clock gating sul registro in uscita. Infatti, dalla descrizione VHDL, il sintetizzatore non è in grado di riconoscere la possibilità di applicare il clock gating. A tal proposito, il codice è stato modificato al fine di includere questa opzione.

```
architecture behavioral of inccomp is

signal syncha, synchb : std_logic_vector(7 downto 0);

begin

        p1: process(ck,rst)
        variable tmpa, tmpb : std_logic_vector(7 downto 0);
        begin
          if rst='1' then
                syncha <= (others => '0');
                synchb <= (others => '0');
                C <= (others => '0');
          elsif ck'event and ck='1' then
                tmpa:= syncha;
                tmpb:= synchb;
                if INCA='1' then
                        syncha <= syncha+1;
                        tmpa:= tmpa+1;
                end if;
                if INCB='1' then
                        synchb <= synchb+1;
                        tmpb:= tmpb+1;
                end if;
                if (INCA = '1' OR INCB = '1') then
                        if ((tmpa)>(tmpb)) then
                                C <= tmpa;
                        else
                                C <= tmpb;
                        end if;
                end if;
          end if;
        end process;
end behavioral;
```

In particolare, è stato notato che all'interno del process per la descrizione dell'architettura, l'aggiornamento dei registri REG_A e REG_B avviene successivamente a due condizioni di if relative al valore di INCA e INCB. Si ipotizza che tramite tale descrizione il sintetizzatore sia in grado di applicare il clock gating. Di conseguenza, tale sintassi è stata adottata anche per descrivere l'aggiornamento del registro in uscita, il quale deve essere aggiornato solo quando almeno un segnale di incremento viene asserito.

Nelle seguenti tabelle sono riportati i risultati relativi ai consumi.

Power report - statistiche di default

	No clock gating	Clock gating
Internal power [uW]	39.1897	33.1194
Net switching power [uW]	14.8742	10.8040
Leakage power [uW]	4.6358	4.0346
Total power [uW]	58.6997	47.9580

Power report - reset modificato

	No clock gating	Clock gating
Internal power [uW]	28.3881	27.8883
Net switching power [uW]	16.4611	11.9228
Leakage power [uW]	4.6611	4.4199
Total power [uW]	49.5103	44.2311

Power report - reset e ingressi modificati

	No clock gating	Clock gating
Internal power [uW]	21.3760	10.0773
Net switching power [uW]	11.4117	4.1839
Leakage power [uW]	4.5424	4.3468
Total power [uW]	37.3300	18.6080

Si nota che il clock gating porta ad una riduzione generale della potenza, come già verificato in precedenza. Tuttavia, applicando il clock gating, la potenza di leakage presenta una riduzione: si ipotizza che tale comportamento sia correlato ad una riduzione delle celle utilizzate.
Per verificare ciò è stato effettuato un report di tipo **-cell**.

Cell report - reset e ingressi modificati

	No clock gating	Clock gating
Numero celle	96	81
Area [um^2]	244.72	243.66

L'ipotesi di riduzione del numero di celle è confermata, tuttavia tale risultato è in controtendenza con quanto atteso successivamente all'implementazione di clock gating: ci si aspetta un incremento dell'area totale dovuto all'inserimento della logica atta all'implementazione di tale tecnica.

Si ipotizza che, a causa della modifica apportata al file VHDL, il sintetizzatore, nel caso in cui non sia specificato di applicare il clock gating, interpreti la descrizione inserendo della logica aggiuntiva per verificare la condizione di if. Al contrario, applicando la direttiva **-gate_clock**, la descrizione viene interpretata correttamente dal sintetizzatore.
Il numero di celle risulta comunque maggiore rispetto al caso in cui il clock gating viene applicato solamente ai registri REG_A e REG_B.

3.2.3 Simulazione di consumi tramite back-annotation

Al fine di ottenere una stima più accurata del consumo di potenza del circuito preso in considerazione, è necessario avere valutazione più realistica dell'attività dei singoli nodi del circuito

sintetizzato. A tale scopo è possibile effettuare una simulazione back-annotated, generando post-sintesi un file contenente la descrizione della netlist del circuito.
L'ambiente di simulazione ModelSim è in grado, partendo dal file di cui sopra, di svolgere diverse simulazioni atte a caratterizzare l'attività dei nodi in questione. Tali risultati vengono riportati in un apposito file che, successivamente ad una conversione, può essere letto dal CAD di sintesi. Attraverso le informazioni ottenute, si è in grado di simulare più realisticamente il comportamento del circuito in termini di potenza.

Internal power [uW]	Net switching power [uW]	Leakage power [uW]	Total power [uW]
39.6187	18.3103	4.6799	62.6089

Segnale	Static probability	Toggle rate
clock	0.5	0.4
reset	0.004	0.0005
INCA	0.019	0.001
INCB	0.986	0.0005

È possibile notare che i valori di static probability e toggle rate sono differenti rispetto a quanto impostato precedentemente su Synopsys. In particolare, l'ingresso INCB ha una probabilità prossima a 1 e una switching acitivity molto bassa.
Si ipotizza che i consumi elevati siano dovuti al quasi continuo incremento del valore contenuto nel registro REG_B. Tale fatto mette in luce l'importanza di un testbench in grado di rappresentare correttamente la statistica degli input del circuito.

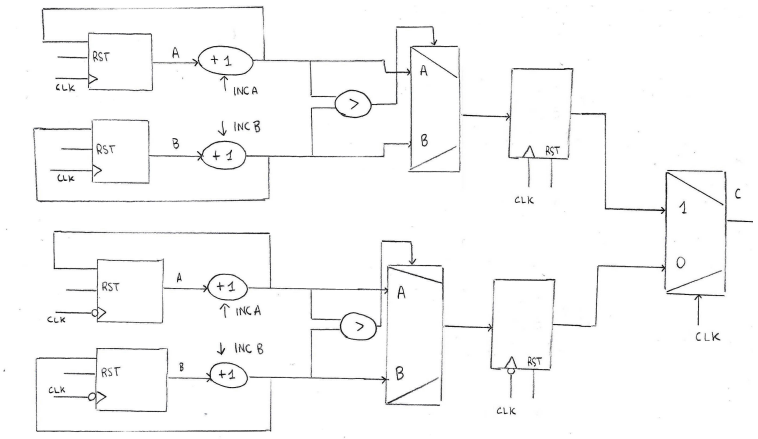
3.3 Pipelining e parallelizzazione

Al fine di ridurre il consumo di potenza dinamica di un generico circuito che presenta un datapath delimitato da barriere di registri, è possibile applicare una serie di soluzioni architetturali.

In particolare, è preso in considerazione il medesimo circuito usato nei punti precedenti, il quale è stato caratterizzato dal punto di vista di ritardo, potenza e area.

- percorso critico = reg + inc + comp + mux = 140 ns
- frequenza massima = 7.14MHz
- potenza @ 5MHz = 3 reg + 2 inc + comp + mux = 10.73 uW
- area = 3 reg + 2 inc + comp + mux = 1747 um^2

È possibile apportare dei miglioramenti in termini di consumo di potenza adottando svariate tecniche a livello architetturale: una di queste consiste nella parallelizzazione del datapath come mostrato nella seguente figura.



In particolare, duplicando il datapath e inserendo un opportuno multiplexer, è possibile far lavorare i componenti a metà della frequenza originale, mantenendo comunque lo stesso throughput in uscita del circuito originale. Data l'espressione per valutare il consumo di potenza dinamica:

$$P = C_{eff}\ V_{dd}^2\ f_{clk}$$

è possibile notare come, in questo caso, il dimezzamento della frequenza di funzionamento sia compensato dal raddoppio della capacità totale.
Tuttavia, si nota che i componenti hanno il doppio del tempo a disposizione per completare le loro operazioni, pertanto è possibile ridurre la tensione di alimentazione al fine di aumentare il ritardo di propagazione in questi ultimi, rispettando comunque il limite imposto dal percorso critico.

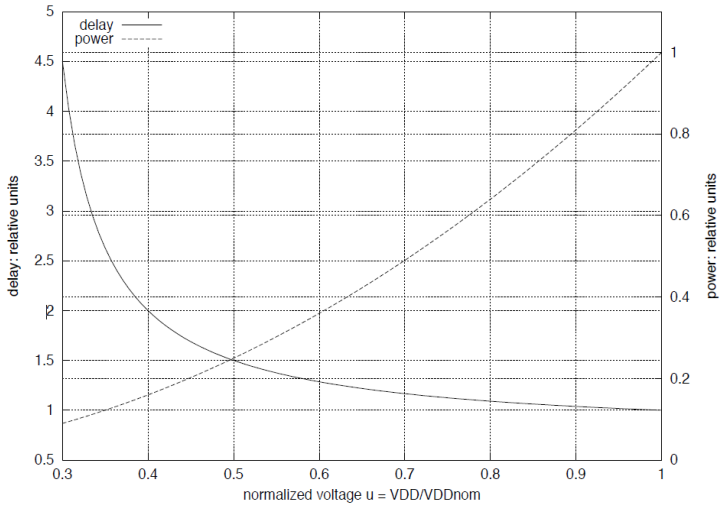
Dal momento che la tensione di alimentazione contribuisce in modo quadratico al consumo di potenza, quest'ultima risulterà ridotta di un fattore u^2, dove u rappresenta il fattore di riduzione di V_{dd}.

$$P(u) = P_{nom}\ u^2$$

I ritardi di propagazione non si riducono linearmente con la tensione di alimentazione ma seguono un andamento che può essere descritto dalla seguente formula:

$$T(u) = T_{nom}\frac{0.75u}{u-0.25}$$

Il seguente grafico mostra, in unità relative, l'andamento di potenza e ritardo in funzione del fattore di scalamento.



La modifica apportata all'architettura consente, come mostrato in precedenza, di lavorare ad una frequenza pari alla metà della frequenza nominale, di conseguenza si stima che i ritardi non possano essere incrementati al di sopra di un fattore 2.
Considerando ciò e osservando il grafico è stato scelto u = 0.5, corrispondente ad un buon trade-off tra riduzione dei consumi e aumento dei ritardi. Successivamente è stata caratterizzata nuovamente la tabella che descrive i parametri dei componenti.

Cella	Ritardo [ns]	Potenza @2.5MHz [uW]	Area [um^2]
registro	3	0.075	319
incrementatore	60	0.319	256
comparatore	126	0.27	161
mux	21	0.209	117

Come atteso, si nota che un aumento dei ritardi dei singoli componenti corrisponde ad una riduzione dei loro consumi.

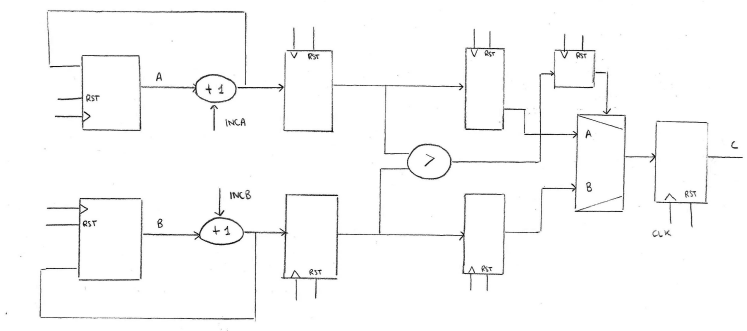
- percorso critico = reg + inc + comp + mux = 210 ns
- frequenza massima = 4.76 MHz

- potenza @ 2.5 MHz = 2·(3 reg + 2 inc + comp + mux) + mux(overhead) = 2.89 uW
- area = 2·(3 reg + 2 inc + comp + mux) + mux = 3611 um^2

Si può notare che la potenza è stata ridotta di un fattore 3.71 a fronte di un raddoppio dell'area. In particolare, è possbile osservare l'effetto dell'overhead del mux in uscita nella valutazione del percorso critico, che risulta maggiore.

Un'altra tecnica a livello architetturale applicabile in questo contesto è la pipeline. L'inserimento di registri all'interno del percorso combinatorio permette di ridurre il percorso critico, mantenendo comunque un throughput elevato.
Riducendo il percorso critico si potrebbe lavorare ad una frequenza maggiore rispetto alla frequenza nominale, tuttavia si decide di mantenere la stessa frequenza del circuito originale, avendo quindi la possibilità di incrementare i ritardi attraverso una riduzione di V_{dd}.

In questo caso si è deciso di isolare, attraverso due barriere di registri, il comparatore, essendo il componente che presenta il maggior ritardo di propagazione. Di conseguenza è stato ottenuto il seguente datapath.



Si fa presente che per garantire la coerenza è necessario inserire elementi di memoria in modo da assicurare il corretto timing di dati e controlli. È dunque possibile stimare il nuovo percorso critico del circuito, considerando la tensione di alimentazione nominale:

$$\text{percorso critico: 1 reg + 1 comp} = 86 \text{ ns}$$

tale valore consente di lavorare ad una massima frequenza pari a 11.63 MHz. Date le considerazioni precedenti, nel contesto attuale è sufficiente lavorare ad una frequenza di 5 MHz, pari a quella nominale.
Il massimo fattore di scalamento dei ritardi di propagazione è pari a:

$$\frac{11.63}{5} = 2.33$$

A tale fattore di scalamento corrisponde una $u_{max} = 0.37$, ricavata invertendo la formula precedentemente riportata. In questo caso, è stata ricercata la massima riduzione possibile della tensione di alimentazione.

Cella	Ritardo [ns]	Potenza @5MHz [uW]	Area [um^2]
registro	4.6	0.082	319
incrementatore	92.5	0.349	256
comparatore	194.25	0.296	161
mux	32.4	0.228	117

- percorso critico = reg + comp = 198.85 ns
- frequenza massima = 5.03 MHz
- potenza @ 5 MHz = 3 reg + 2 inc + comp + mux + 5 reg (pipe) = 1.86 uW
- area = 3 reg + 2 inc + comp + mux + 5 reg (pipe) = 3342 um^2

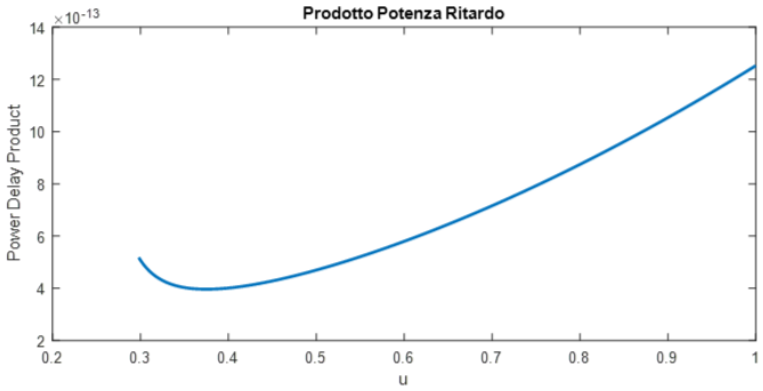
Come atteso, si nota una significativa riduzione del consumo di potenza, pari ad un fattore 5.77. Inoltre, il circuito risulta più compatto rispetto al caso della parallelizzazione.

In generale, la tecnica della pipeline permette di raggiungere risultati migliori in termini di trade-off tra potenza e area. Tuttavia tale soluzione modifica radicalmente il timing del sistema, il quale rimane invariato nel caso di utilizzo della tecnica di parallelizzazione.

Di conseguenza, la tecnica migliore dipende fortemente dal contesto di progetto: nel caso di un sistema da progettare si privilegia la pipeline; al contrario, nel caso sia necessaria un'ottimizzazione di un sistema già progettato è prediletta la tecnica della parallelizzazione in quanto meno invasiva.

Per confronto si è cercato il minimo consumo di potenza applicando solamente la tecnica della parallelizzazione, con parallelismo 2. In seguito alla riduzione di V_{dd}, il minimo consumo di potenza dinamica ottenibile è pari a 1.85 uW, scegliendo u = 0.4, consumo comparabile a quello ottenuto con la tecnica di pipeline.
Per raggiungere un'ulteriore riduzione dei consumi, è necessario introdurre un terzo livello di parallelismo: in questo caso è possibile selezionare u = 0.37, ottenendo quindi un consumo di potenza pari a 1.58 uW.

Partendo dalla soluzione parallelizzata, sono stati ricercati possibili miglioramenti unendo le due tecniche. Il grafico seguente mostra il prodotto potenza ritardo in funzione del fattore u, considerando la presenza di pipeline e di un parallelismo pari a 2.

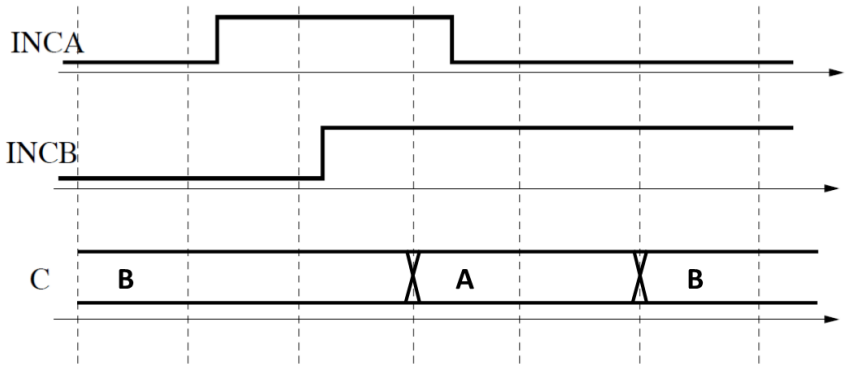


Si evince che sarebbe possibile utilizzare un fattore di scalamento della tensione di alimentazione inferiore a 0.37, tuttavia riducendolo vi è un'inversione di tendenza della curva mostrata. Ciò significa che per valori di u al di sotto del punto di minimo, il guadagno in termini di potenza risparmiata è poco significativo rispetto ai ritardi introdotti.

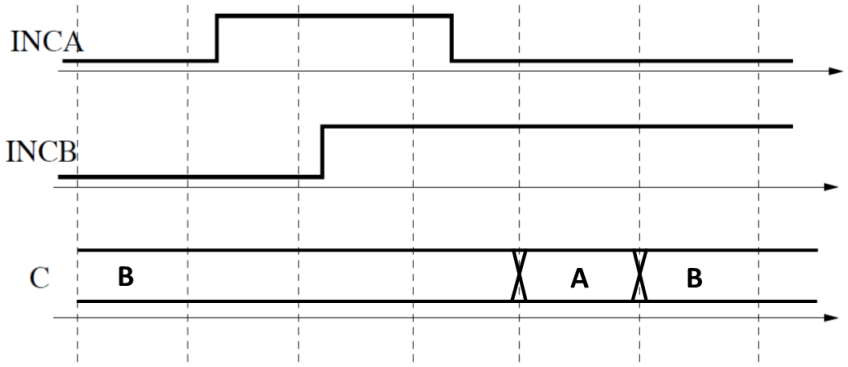
Si fa presente che l'analisi effettuata è relativa alla sola potenza dinamica, infatti una riduzione significativa di V_{dd} comporta l'utilizzo di transistor caratterizzati da una tensione di soglia minore, aumentando il contributo di corrente di leakage.

3.3.1 Il problema della coerenza

Applicando la tecnica della parallelizzazione all struttura di base, è possibile notare che la coerenza dei dati in uscita non è assicurata.
Considerando le forme d'onda in ingresso come mostrato nella figura seguente, è possibile stimare l'andamento atteso dell'uscita C nel caso del circuito originale.



Al contrario nel caso di implementazione con 2 livelli di parallelizzazione si ha il comportamento seguente.



Si nota che il comportamento è inatteso in quanto non vi è coerenza tra i registri delle due sottostrutture. Per assicurare un corretto funzionamento del circuito, è necessario che i dati con cui lavorano le due entità siano condivisi, in modo da garantire la coerenza.

Una possibile soluzione potrebbe essere l'impiego di due soli registri REG_A e REG_B condivisi da entrambe le strutture in parallelo e scritti ad una frequenza doppia.

- schema clock gating FATTO
- datapath parallelizzazione FATTO
- datapath pipeline FATTO
- forme d'onda FATTO

Bus encoding

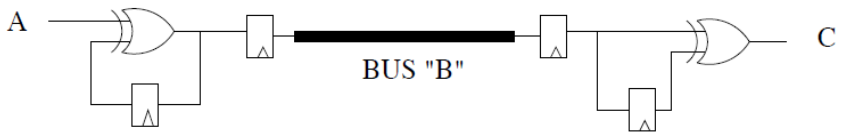
4.1 Introduzione alle tecniche utilizzate

Lo scopo di questa esercitazione è testare diverse codifiche utilizzate per ridurre le commutazioni sui bus. È fondamentale, in un contesto low power, prestare attenzione ai consumi relativi alle interconnessioni, in quanto ad esse è associata una capacità significativa.
In particolare, sono state testate le seguenti quattro tecniche:

- **Bus normale**
  Interconnessione priva di qualunque tipo di codifica e, per tale motivo non ridondante.

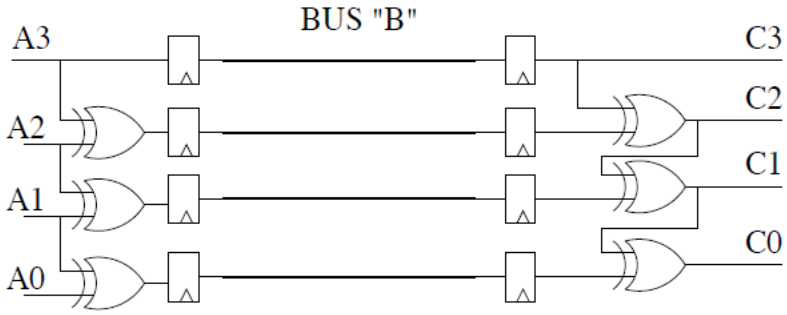
- Bus invert
  Il valore logico sull'interconnesione è invertito se la distanza di Hamming rispetto al valore precedente è maggiore di $\frac{N}{2}$, dove N è il parallelismo dell'informazione. Si tratta di una codifica ridondante, infatti è necessario notificare al ricevitore l'eventuale inversione dell'informazione inviata. Tale codifica è idonea alla trasmissione dei dati, in quanto la statistica di questi è casuale rispetto a quella di indirizzi e controlli.

- **Transition based**
  L'informazione è codificata non in un livello logico ma in una transizione della linea, che avviene quando il dato assume un valore pari a 1. Si nota che tale codifica non è ridondante dal momento che non sono necessari ulteriori segnali.
  I circuiti di codifica e decodifica fanno uso di porte XOR e flip flop per memorizzare il valore precedente, che viene comparato con quello attuale. Si nota che, in trasmissione, un valore di A pari a 1 porta alla commutazione del bus rispetto al valore assunto precedentemente. In ricezione, invece, quando è identificata una transizione sul bus, confrontando il valore attuale con quello precedente, l'uscita C assume un valore logico alto.



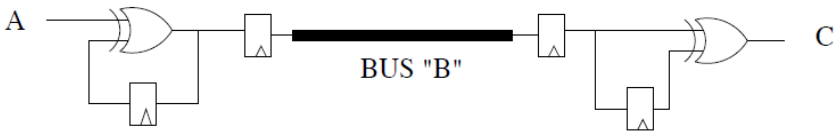
- **Gray code**
  La codifica ha come principale scopo il mantenimento della distanza di Hamming pari a 1 tra due informazioni consecutive. Tale codifica risulta appropriata per la trasmissione

degli indirizzi. Si nota inoltre che questa codifica non prevede l'invio di ulteriori segnali di controllo, è pertanto non ridondante.
In particolare, il seguente schema mostra i circuiti di codifica e decodifica.



È possibile notare come un ingresso sia codificato correttamente solo andando in sequenza.

- **T0 encoding**
  Tale codifica ha il principale scopo di eliminare le commutazioni sul bus, nel caso in cui l'informazione sia sequenziale: per tale motivo è particolarmente adatta a indirizzi. Quando è identificata una sequenza dal trasmettitore, quest'ultimo mantiene l'indirizzo di partenza fisso sul bus e abilita un apposito segnale aggiuntivo atto a segnalare al ricevitore la sequenzialità dell'informazione.
  Per tale motivo è necessario implementare un incrementatore al ricevitore per generare l'indirizzo corretto.
  In caso di salto, quest'ultimo viene segnalato cambiando l'indirizzo inviato sul bus e negando il segnale seq. Si nota quindi che tale tecnica è ridondante.
  block scheme



Basandosi sul principio di funzionamento sopra descritto, è stata realizzata la seguente descrizione VHDL.

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;
use work.all;

entity T0beh is
port (  ck : in std_logic;
        rst : in std_logic;
        A : in std_logic_vector(7 downto 0);
        B : out std_logic_vector(8 downto 0);
        C : out std_logic_vector(7 downto 0));
end T0beh;

architecture behavioral of T0beh is

signal AOLD, C_tmp, buss : std_logic_vector(7 downto 0);
signal seq : std_logic;
begin

        penc: process(ck,rst)
        begin
          if rst='1' then
            AOLD <= (others => '0');
            seq <= '0';
                buss <= (others => '0');
          elsif ck'event and ck='1' then
            if (A = AOLD + 1) then
              AOLD <= A;
                   seq <= '1';
            else
                   AOLD <= A;
              buss <= A;
                   seq <= '0';
            end if;
          end if;
        end process;

        B(8) <= seq;
        B(7 DOWNTO 0) <= buss;


        pdec: process(ck,rst)
        begin
          if rst='1' then
            C_tmp <= (others => '0');
          elsif ck'event and ck='1' then
            if seq='1' then
              C_tmp <= C_tmp + 1;
            else
              C_tmp <= buss;
            end if;
          end if;
        end process;

        C <= C_tmp;

end behavioral;
```

4.2 Simulazione

Attraverso la piattaforma ModelSim è possibile, attraverso un testbench, ottenere una stima dell'attività relativa alle diverse tecniche di codifica implementate.
Dal momento che alcune codifiche sono più appropriate al trattamento di dati o indirizzi, vi è la possibilità di generare sequenze di informazioni con opportune distribuzioni statistiche.

All'interno del file tb_encdec.v è possibile, attraverso due direttive, scegliere la statistica dei segnali da simulare opportunamente generata tramite due process. In particolare, si nota come il process relativo ai dati prelevi questi ultimi dal file rdin.txt dove è presente una sequenza pseudo-random di valori rappresentati su 8 bit.

Al contrario, il process relativo agli indirizzi, procede in sequenza fino al raggiungimento dell'indirizzo 63, successivamente al quale simula un salto prelevando l'indirizzo successivo dal file di cui sopra.
Tutte le tecniche di codifica sono state simulate sia con dati che con indirizzi, riportando le attività dell'ingresso A e delle linee di bus B. Come primo risultato, è interessante notare la diversa attività delle due statistiche dell'informazione in ingresso.

	Address		Data	
Nodo	Tc	E_{sw}	Tc	E_{sw}
A(7)	97		4974	
A(6)	118		5022	
A(5)	312		4987	
A(4)	625		4972	
A(3)	1250		4960	
A(2)	2500		4965	
A(1)	5000		5024	
A(0)	10000		5060	
Totale				

Come atteso, la statistica relativa ai dati presenta attività maggiori e confrontabili tra loro. Per quanto riguarda gli indirizzi, invece, si nota un'attività che mostra la sequenzialità.

4.2.1 Bus normal

	Address		Data	
Nodo	Tc	E_{sw}	Tc	E_{sw}
BUSNORM(7)	97		4974	
BUSNORM(6)	118		5021	
BUSNORM(5)	312		4987	
BUSNORM(4)	624		4972	
BUSNORM(3)	1249		4959	
BUSNORM(2)	2499		4965	
BUSNORM(1)	4999		5023	
BUSNORM(0)	9999		5060	
Totale	19897		39961	

Come atteso, in assenza di codifica, l'attività del bus rispecchia quella dell'ingresso.

4.2.2 Bus invert

	Address		Data	
Nodo	Tc	E_{sw}	Tc	E_{sw}
inv	624		3650	
BUSINV(7)	527		3576	
BUSINV(6)	506		3611	
BUSINV(5)	312		3617	
BUSINV(4)	0		3640	
BUSINV(3)	625		3613	
BUSINV(2)	1875		3601	
BUSINV(1)	4375		3639	
BUSINV(0)	9375		3722	
Totale	18219		32669	

Come atteso, osservando i valori totali, è possibile notare come sia presente una significativa riduzione delle commutazioni per quanto riguarda i dati rispetto al bus normal. Al contrario, le attività degli indirizzi risultano confrontabili.
La riduzione delle commutazioni è tanto più marcata quante più transizioni con una distanza di Hamming maggiore di 4 sono presenti nei dati; attraverso uno script Matlab è stata stimata la percentuale di tali transizioni sul totale, che risulta pari a circa il 36 perc.
se avanza tempo fare simulazione con perc = 100

4.2.3 Transition based

	Address		Data	
Nodo	Tc	E_{sw}	Tc	E_{sw}
TRAN(7)	4816		5003	
TRAN(6)	3776		4946	
TRAN(5)	4992		4938	
TRAN(4)	4992		4997	
TRAN(3)	5000		5125	
TRAN(2)	5000		4989	
TRAN(1)	5000		5000	
TRAN(0)	5000		4994	
Totale	38576		39992	

Si nota come tale codifica non sia indicata per gli indirizzi in quanto la sequenza stessa ha come prerogativa il mantenimento di valori logici alti sulle linee di bus.
Per quanto riguarda i dati, è atteso un miglioramento nel caso in cui la probabilità di avere valori logici alti sia sufficientemente ridotta. Tuttavia, analizzando il file rndin.txt è possibile osservare la presenza di numerosi 1 all'interno dei dati.
se avanza tempo fare simulazione file pochi uni

4.2.4 Gray coding

	Address		Data	
Nodo	Tc	E_{sw}	Tc	E_{sw}
GRAY(7)	97		4974	
GRAY(6)	55		5087	
GRAY(5)	194		4952	
GRAY(4)	312		4981	
GRAY(3)	625		4937	
GRAY(2)	1250		5056	
GRAY(1)	2500		4962	
GRAY(0)	5000		5075	
Totale	10033		40024	

Come atteso, tale codifica non ha benefici sulla codifica dei dati ma presenta notevoli miglioramenti per quanto riguarda gli indirizzi. L'attività totale degli indirizzi risulta infatti dimezzata rispetto al caso bus normal.

4.2.5 T0

	Address		Data	
Nodo	Tc	E_{sw}	Tc	E_{sw}
seq	119		72	
T0(7)	29		4974	
T0(6)	24		5021	
T0(5)	0		4987	
T0(4)	0		4972	
T0(3)	0		4957	
T0(2)	0		4963	
T0(1)	0		5001	
T0(0)	0		5028	
Totale	172		39975	

Si nota un netto miglioramento per la trasmissione degli indirizzi, in particolare l'attività molto basse mette in luce che gli indirizzi generati mostrano un'elevata sequenzialità.
In linea con i risultati attesi, i dati presentano un'elevata attività.

4.3 Sintesi

Lo scopo di questa sezione è l'analisi del consumo di potenza relativo alle varie tecniche implementate. Tale stima è stata effettuata attraverso il procedimento di back-annotation descritto nel laboratorio precedente, in modo da ottenere stime verosimili sia per dati che per indirizzi.

L'intero processo è basato su un insieme di script forniti che descrivono la sequenza delle operazioni da svolgere che coinvolgono i tool ModelSim e Synopsys. L'analisi è inoltre effettuata considerando diversi carichi capacitivi per le linee di interconnessione.

Power	Address				Data			
[uW]	1fF	10fF	50fF	100fF	1fF	10fF	50fF	100fF
BUSNORM	29		4974					
BUSINV	24		5021					
TRAN-BASED	0		4987					
GRAY	0		4972					
T0	0		4957					

ciao%.