

Low Power Design
Project Deliverable

2023-2024

“Low Power Design”

2023-2024

Prof. FRUSTACI

FPGA Analysis

Date	<10/12/2023>
Document	Final Document

Full Name	ID	E-mail Address
Giorgio Ubbriaco	247284	bbrgrg00h11d086x@studeni.ti.unical.it

Contents

I.	TASKS TO BE PERFORMED	5
II.	ABSTRACT	6
1.	INTRODUCTION	7
1.1.	ZEDBOARD™	7
1.2.	FPGA	7
1.3.	VHDL	8
1.4.	VIVADO DESIGN SUITE	8
1.5.	VHDL FOR LOW POWER	9
1.5.1.	<i>Registering Technique</i>	9
1.5.2.	<i>Clock Gating Technique</i>	10
1.5.3.	<i>Hybrid Technique</i>	12
1.6.	RANDOM INPUT GENERATION	12
2.	NON-OPTIMIZED ARCHITECTURE ANALYSIS AND DESIGN	13
2.1.	RTL DESCRIPTION AND SCHEMATIC	14
2.2.	SYNTHESIS	15
2.3.	POWER ANALYSIS	16
2.4.	TIMING ANALYSIS	17
2.5.	MAXIMUM FREQUENCY ANALYSIS	17
2.6.	RESOURCES UTILIZATION ANALYSIS	17
3.	REGISTERING TECHNIQUE ANALYSIS AND DESIGN	18
3.1.	RTL DESCRIPTION AND SCHEMATIC	18
3.2.	SYNTHESIS	19
3.3.	POWER ANALYSIS	20
3.4.	TIMING ANALYSIS	21
3.5.	FREQUENCY ANALYSIS	21
3.6.	RESOURCES UTILIZATION ANALYSIS	21
4.	CLOCK GATING TECHNIQUE ANALYSIS AND DESIGN	22
4.1.	SELECTORS CLOCK GATING	22
4.1.1.	<i>RTL Description and Schematic</i>	22
4.1.2.	<i>Synthesis</i>	23
4.1.3.	<i>Power Analysis</i>	24
4.1.4.	<i>Timing Analysis</i>	25
4.1.5.	<i>Frequency Analysis</i>	25
4.1.6.	<i>Resources Utilization Analysis</i>	25
4.2.	INPUTS CLOCK GATING	26
4.2.1.	<i>RTL Description and Schematic</i>	26
4.2.2.	<i>Synthesis</i>	27
4.2.3.	<i>Power Analysis</i>	28
4.2.4.	<i>Timing Analysis</i>	29
4.2.5.	<i>Frequency Analysis</i>	29
4.2.6.	<i>Resources Utilization Analysis</i>	29
4.3.	ALL CLOCK GATING	30
4.3.1.	<i>RTL Description and Schematic</i>	30
4.3.2.	<i>Synthesis</i>	31
4.3.3.	<i>Power Analysis</i>	32
4.3.4.	<i>Timing Analysis</i>	33
4.3.5.	<i>Frequency Analysis</i>	33

4.3.6. <i>Resources Utilization Analysis</i>	33
5. HYBRID TECHNIQUE ANALYSIS AND DESIGN	34
5.1. RTL DESCRIPTION AND SCHEMATIC.....	34
5.2. SYNTHESIS	35
5.3. POWER ANALYSIS	36
5.4. TIMING ANALYSIS	37
5.5. FREQUENCY ANALYSIS	37
5.6. RESOURCES UTILIZATION ANALYSIS	37
6. CONCLUSIONS	38

I. Tasks to be performed

Prendendo come riferimento il circuito rappresentato nella slide numero 15 del gruppo di slide “CH12_VHDL_for_LowPower”, cercare di ottimizzare la dissipazione di energia del circuito utilizzando una o più tecniche viste a lezione (esempio NON esaustivo: gate level reordering, clock gating, intelligent clock gating, registering, precomputation etc.). Per semplificare il problema, considerare le seguenti cose:

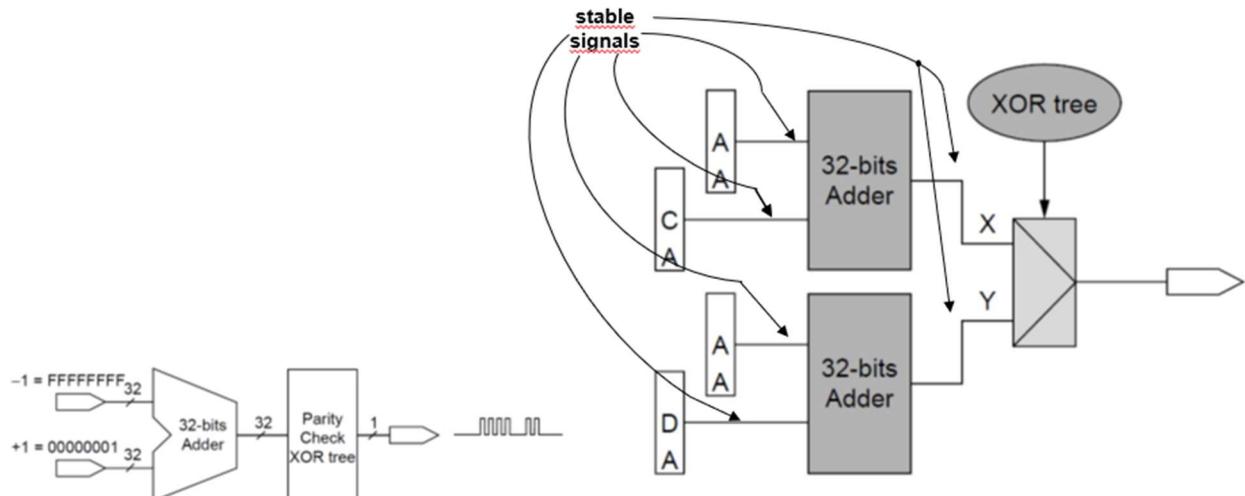
- un registro di uscita a 33 bit
- il sommatore a monte del parity checker sia di 8 bit
- gli ingressi del sommatore a 8 bit a monte del parity checker provengano da due registri a 8 bit

Lo spazio di tutte le possibili combinazioni di tecniche risulta essere molto ampio, per cui non è necessario analizzare TUTTE le possibili combinazioni di tecniche. È sufficiente selezionare alcune tecniche e/o alcune combinazioni di esse che siano ragionevolmente promettenti per la riduzione dell’energia dissipata. Per ognuna delle tecniche/combinazioni di tecniche selezionate, analizzare la dissipazione dinamica di energia (escludere l’energia degli I/O), la frequenza massima di clock e le risorse hardware utilizzate. Commentare i risultati attesi per ogni tecnica e confrontarli con quelli realmente ottenuti. Per analizzare la dissipazione di energia, utilizzare degli opportuni testbench. Nel confrontare la dissipazione di energia delle varie tecniche, confrontare i file SAIF ottenuti e commentare i risultati di conseguenza. Come output principale dell’elaborato, si dovrebbe ottenere una tabella con l’analisi di tutte le tecniche analizzate in cui si evinca la migliore soluzione (e perché).

II. Abstract

Nella parte iniziale del progetto è prevista la descrizione, mediante il linguaggio di descrizione dell'hardware VHDL e il software di progettazione Xilinx Vivado 2018.3, di un circuito digitale non ottimizzato, precedentemente affrontato durante le lezioni, e la sua ottimizzazione mediante tecniche low-powering. Nello specifico, nella parte finale di questo report verranno analizzati i risultati ottenuti al fine di ricercare la migliore soluzione tra quelle proposte. Tra queste strategie, sono state esaminate e testate tecniche quali il registering, il clock gating in svariate configurazioni e un approccio ibrido che combina entrambe le metodologie.

Per quanto riguarda il circuito non ottimizzato, questo presenta 4 input, A, B, C e D, disposti in ingresso a 2 sommatori. Nello specifico, un adder sarà relativo alla somma tra A e C mentre l'altro relativo alla somma tra B e D. Infine, le somme ottenute, verranno disposte in input ad un multiplexer, che secondo un determinato selettore, sceglierà quale dei due risultati ottenuti, mediante sommatori, processare in output. Il selettore viene generato tramite un componente Parity Check al quale viene messo in input il segnale ottenuto dalla somma tra i due selettori previsti in input all'architettura. Tale circuito non ottimizzato verrà affrontato nei dettagli nei successivi capitoli.



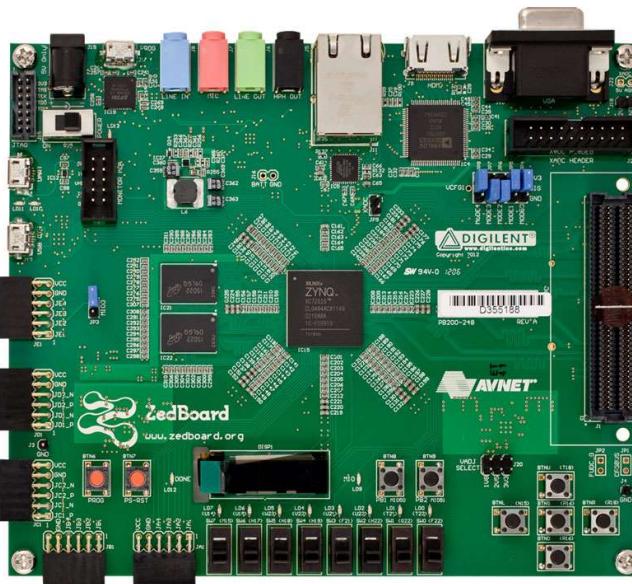
Pertanto, considerando questa progettazione hardware iniziale, verranno implementate, mediante gli strumenti sopra citati, alcune tecniche al fine di ottenere miglioramenti dal punto di vista della potenza e considerando opportuni trade-off.

1. Introduction

In questo capitolo verranno introdotti alcuni concetti teorici utili ad affrontare le tematiche e le analisi svolte all'interno del report del progetto.

1.1. ZedBoard™

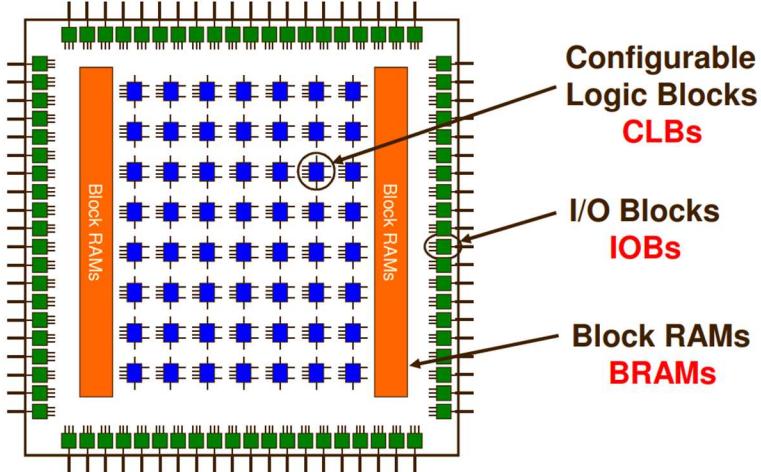
Una piattaforma ideale sia per sviluppatori principianti sia per quelli esperti è la ZedBoard™. Essa è una scheda di sviluppo a basso costo basata sul Xilinx Zynq-7000 All Programmable SoC. È composta dal processore multi-core Cortex-A9 con frequenza di funzionamento massima pari a 667 MHz , una memoria da 512 MB di tipologia DDR3 e 85,000 Series-7 Programmable Logic (PL) tale da essere destinata per un utilizzo in molte applicazioni. Inoltre, prevede uno slot per microSD e diverse periferiche tale da renderla una piattaforma versatile e flessibile in molti usi.



1.2. FPGA

Oggigiorno, l'interesse relativo ai sistemi riprogrammabili tende ad aumentare sempre di più. Tanto è vero che una tra le figure più richieste nel mondo ingegneristico è quella dello sviluppatore competente sia in ambito software che hardware, capace di progettare componenti HW e SW per sistemi elettronici complessi. Nello specifico, sfruttando schede riconfigurabili, è possibile avere flessibilità ed efficienza pur mantenendo una dissipazione di potenza ed un costo moderati. Quindi, si può evincere come la richiesta di chip, che incorporano un microprocessore a cui sono cablate le logiche programmabili, sia sempre più alta.

Uno tra i dispositivi logici programmabili più versatili è l'FPGA. Il Field Programmable Gate Array è un dispositivo logico riconfigurabile composto da un circuito integrato le cui funzionalità logiche sono programmabili tramite linguaggi descrittivi hardware come, ad esempio, VHDL e Verilog.



Al suo interno è presente una matrice di blocchi logici configurabili, denominati CLB (Configurable Logic Blocks), i cui collegamenti fra loro non sono prestabiliti. Ogni blocco presenta una struttura gerarchica. Infatti, è composto da 2 o 4 logic cell ed ognuna di quest'ultime è composta da una o più LUT (Look-up Table). Ogni LUT è composta da una memoria SRAM da 16 bit e da un multiplexer a 4 ingressi tale che ognuna di esse potrà essere progettata affinché riproduca una funzione logica. I collegamenti delle CLB sono organizzati tramite matrici di interruttori programmabili (Programmable Switch Matrix). Inoltre, lungo il perimetro della matrice, sono presenti gli IOB (Input Output Block) che si occupano dell'interfacciamento input-output del circuito con l'esterno.

1.3. VHDL

Uno tra i linguaggi descrittivi dell'hardware più utilizzati è il VHDL (Very High Speed Integrated Circuits Hardware Description Language). Esso, sviluppato dal Dipartimento della Difesa (DoD) statunitense, è uno standard IEEE11 utilizzato per la progettazione di sistemi elettronici digitali. Il VHDL si propone come linguaggio indipendente dall'architettura di implementazione tale da essere usato sia per la sintesi che per la simulazione del circuito progettato. Tanto è vero che, tramite la dichiarazione del modulo che si vuole simulare, è possibile creare un file di testbench che permette di analizzare il comportamento del circuito, cioè l'output generato, a fronte di determinati ingressi imposti al modulo.

Di notevole importanza è la possibilità di una modellazione di tipo gerarchica, cioè descrivere un componente attraverso altri sotto-moduli già progettati. Essi potranno essere connessi tra di loro tramite dei signals. Quest'ultimi appena citati vengono, infatti, utilizzati per modellare l'informazione che transita tra i vari moduli progettati, cioè tra le porte di ognuno e, quindi, assumere i connotati di un'entrata (input) o di un'uscita (output) di un componente logico, oppure semplicemente per realizzare una determinata funzione logica.

1.4. Vivado Design Suite

La Vivado Design Suite è una suite di strumenti progettata da Xilinx per aumentare la produttività complessiva della progettazione, dell'integrazione e dell'implementazione di sistemi che utilizzano dispositivi UltraScale™, 7 series e Versal®, MPSOC Zynq® UltraScale+™ e SoC Zynq®-7000. Essa è utilizzata per la sintesi e l'analisi di progetti scritti in un linguaggio descrittivo dell'hardware di tipologia HDL. Vivado risulta essere un'evoluzione rispetto a Xilinx ISE poiché introduce funzionalità per lo sviluppo di SoC12 e per la sintesi ad alto livello. Inoltre, la suite permette di analizzare, verificare e modificare il progetto ad ogni fase del processo di progettazione.

1.5. VHDL for Low Power

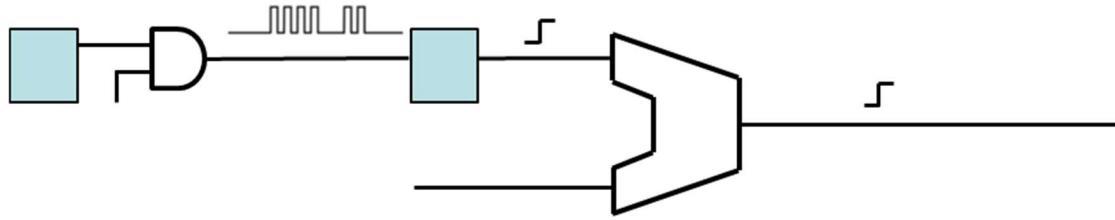
Il linguaggio di descrizione dell'hardware VHDL, oltre a permettere una corretta interpretazione da parte di alcuni software riguardo l'architettura hardware che si vuole progettare, consente di implementare attraverso metodi impliciti ed esplicativi tecniche che possono effettivamente migliorare alcuni parametri di progettazione come, ad esempio, la misura della potenza. Nello specifico, è possibile esprimere attraverso alcuni costrutti software alcune metodologie low powering come, ad esempio, il Clock Gating Esplicito. Inoltre, attraverso l'omissione di altri costrutti è possibile implementare invece il Clock Gating Implicito. Oppure ancora, attraverso, l'introduzione di altri componenti hardware è possibile attuare ulteriori strategie come, ad esempio, il Registering. In aggiunta, attraverso la re-organizzazione dello schematic, è possibile mettere in atto tecniche low powering più complesse che potrebbero migliorare le prestazioni del circuito stesso come, ad esempio, la tecnica conosciuta con l'appellativo di Gate Level Reordering. Bisogna notare che le tecniche a disposizione per effettuare miglioramenti ad alcuni aspetti cruciali, del circuito che viene considerato, sono molteplici e, soprattutto, la loro applicazione in alcuni casi risulta essere differente rispetto ad altri. Inoltre, è necessario specificare che l'applicazione di alcune di queste tecniche è più semplice rispetto ad altre ed in alcuni casi risulta essere più efficiente per alcuni punti di vista mentre in altre situazioni è fondamentale applicare strategie più complesse dal punto di vista della progettazione.

In questo progetto sono state utilizzate, come precedentemente specificato, tre tecniche: la strategia di registering, quella del clock gating e, infine, un ibrido tra le due appena citate. Nello specifico, tali tecniche, dal punto di vista teorico, verranno illustrate nei successivi paragrafi.

In generale, l'analisi della potenza prevede considerazioni a livello statico e a livello dinamico. Nello specifico, le tecniche di riduzione della potenza statica includono variazioni di V_{TH} , V_{DD} e ulteriori strategie. Pertanto, applicare una di queste tecniche vuol dire modificare il processo tecnologico relativo al MOS o dell'architettura globale del sistema. Quindi, dal punto di vista hardware, nel caso di utilizzo di FPGA, tali tecniche non sono realmente utilizzabili a livello RTL. Viceversa, l'analisi e l'attuazione di tecniche rivolte alla diminuzione della potenza dinamica risulta essere più adeguata alle soluzioni hardware in questione. Nello specifico, analizzando l'attività di switching dell'architettura progettata e, pertanto, considerando il parametro caratteristico associato, cioè l'activity factor α , è possibile modellare soluzioni low powering associate all'applicazioni considerata. Infatti, nel seguente report, per ogni strategia utilizzata, verrà considerato un α , relativo ad un determinato segnale, per spiegare a livello pratico come questo parametro possa influenzare la potenza dinamica associata al circuito. Inoltre, per effettuare una stima accurata della potenza, precedentemente citata, verrà considerata la generazione del file .saif mediante l'utilizzo di stimoli digitali in input all'architettura in questione. Nello specifico, la simulazione del circuito, in condizioni random (mediante generazione di input random tramite script python), permette di analizzare l'attività di switching di ogni singolo nodo e ottenere una stima della potenza totale del circuito. Ovviamente, più si considerano stimoli random e più la stima del report di potenza sarà accurata.

1.5.1. Registering Technique

La tecnica del registering consiste nell'utilizzo di registri per limitare l'attività di switching dovuta ai glitch che si propagano lungo le interconnessioni. Infatti, tale fenomeno non si ferma localmente ma viene propagato, tramite il segnale in cui sono presenti, attraverso i moduli. Nello specifico, i glitch sono molto importanti dal punto di vista della potenza perché essi comportano continui switching e, pertanto, facendo peggiorare il parametro di activity factor e il valore di potenza dinamica associata. Questo vuol dire che, maggiore saranno i glitch maggiore sarà l'attività di switching e la potenza dinamica corrispondente.



Pertanto, l'introduzione di registri all'interno dell'architettura hardware permetterebbe di ridurre l'attività dei glitch e il corrispondente switching. Ovviamente, questo vuol dire considerando un maggiore numero di risorse, una maggiore latenza e una differente distribuzione del clock all'interno del layout.

Nello specifico, avere una propagazione dei glitch vuol dire avere un valore di transition count TC elevato, cioè sta a significare che l'attività di switching è elevata. Una conferma di ciò può essere analizzata nel dettaglio facendo riferimento rispettivamente al file .saif dell'architettura non ottimizzata e di quella relativa all'applicazione del registering. Prendendo, ad esempio, un segnale comune ad entrambe le architetture si può notare come tale parametro venga ridotto utilizzando la tecnica low powering del registering:

$(D\backslash[0\backslash] (T0\ 802384) (T1\ 395517) (TX\ 2099) (TZ\ 0) (TB\ 0) (TC\ 426))$

$(D\backslash[0\backslash] (T0\ 598787) (T1\ 599738) (TX\ 1475) (TZ\ 0) (TB\ 0) (TC\ 99))$

La prima riga è relativa al file .saif dell'architettura non ottimizzata mentre la seconda è relativa alla soluzione hardware mediante registering. Si può notare come il valore di transition count da 426 viene ridotto al valore di 99. Pertanto, considerando tale approccio sull'intera architettura, ci si aspetta una riduzione della potenza dinamica dal momento che l'attività di switching viene ridotta tramite tale tecnica.

1.5.2. Clock Gating Technique

La tecnica del Clock Gating fa parte della tipologia di strategie implementabili, attraverso esplicitazione e non, di costrutti tramite linguaggio VHDL. Nello specifico, considerando il segnale di clock enable, è possibile specificare se aggiornare il dato di un determinato registro/flipflop e, pertanto, effettuare miglioramenti dal punto dell'attività di switching. Questo vuol dire che, se non c'è necessità di aggiornare un dato in un determinato istante, allora si può limitare tale attività di switching utilizzando il segnale di clock enable, cioè se $CE=1$ allora il dato viene aggiornato altrimenti no. Questa tecnica è possibile implementarla sia esplicitamente sia implicitamente. Nel primo caso, tale metodo consiste nel verificare il valore logico del pin di clock enable esplicitando proprio il suo uso tramite una direttiva if nel codice VHDL.

```

...
process(clk)
begin
    if rising_edge(clk) then
        if CE = '1' then
            Q <= D;
        end if;
    end if;
end process;
...

```

Per quanto riguarda, invece, la tecnica di clock gating隐式 (implicita) prevede di non esplicitare una determinata condizione. Un esempio potrebbe essere quello di non esplicitare cosa accade ad un determinato segnale nel momento in cui il reset assume il livello logico alto:

```

...
process(clk)
begin
    if rising_edge(clk) then
        if RST = '1' then
            reg1 <= '0';
        else
            reg1 <= a;
            reg2 <= b;
        end if;
    end if;
end process;
...

```

Pertanto, non specificare cosa accade al segnale reg2 nel momento in cui il segnale RST='1' vuol dire applicare la tecnica di clock gating sul segnale reg2, cioè, nel caso in cui tale condizione sia soddisfatta, tale segnale non viene aggiornato.

Inoltre, il clock gating implicito è possibile esprimere tramite un multiplexer. Questo vuol dire che, nel momento in cui bisogna esprimere le condizioni per cui i segnali in ingresso possono essere processati in uscita secondo un certo selettore, all'interno della definizione di tale modulo non viene esplicitato direttamente un determinato caso. Pertanto, non specificare un determinato caso vuol dire effettuare il clock gating secondo una certa logica. Questo vuol dire che il clock enable CE sarà abilitato secondo una certa funzione logica data dalla definizione di tale multiplexer:

```

...
process(clk)
begin
    if rising_edge(clk) then
        case SEL is
            when "00" => MUX_OUT <= A;
            when "01" => MUX_OUT <= B;
            when "10" => MUX_OUT <= C;
            when others => null;
        end case;
    end if;
end process;
...

```

Ad esempio, in questo caso non viene esplicitato cosa accade nel caso in cui il selettore risulta assumere il valore logico "11". Pertanto, analizzando i bit che può assumere il segnale SEL è possibile notare come il segnale di clock enable CE è possibile ricavarlo tramite l'analisi di una tabella di verità. In questo caso, la LUT che dovrà calcolare il valore logico di CE dovrà prevedere il comportamento della funzione logica NAND. Infatti, si avrà che $CE = \overline{sel1} \cdot \overline{sel2}$ tale che per le leggi di De Morgan è possibile esplicitare secondo la seguente funzione logica $CE = \overline{sel1} + \overline{sel2}$.

Bisogna specificare che, in tale progetto il clock gating implicito utilizzato è il primo descritto, cioè prevede la non specificazione esplicita della condizione in cui il RST risulta avere il valore logico basso. Inoltre, bisogna precisare che tale tecnica è stata applicata per i selettori del circuito sopra citato, per gli ingressi A, B, C e D, e, infine, sia per i selettori sia per gli ingressi appena citati.

1.5.3. Hybrid Technique

La tecnica ibrida prevede la sperimentazione dell'utilizzo di più tecniche low power combinata per ottenere un possibile risultato migliore rispetto al considerare ogni strategia singolarmente. In questo caso, il sistema ibrido prevede l'utilizzo del registering combinato alla tecnica del clock gating implicito.

1.6. Random Input Generation

Affinché il report di potenza che verrà generato dal tool sia affidabile, la simulazione delle architetture deve essere effettuata considerando input random. Pertanto, si considera la seguente funzione Python, presente nel file collocato al path “./python/RandomInputsGenerator/TopArchitectureRandInGenerator.py”:

```
...
def TopArchitectureRandInGenerator():
    # Set the number of lines you want in the file
    num_lines = 100
    file_path = os.path.abspath('..../vivado/registering/sim/')
    filename = 'TopArchitectureRandomInputsGenerator.txt'
    top_architecture_random_inputs_file_path = os.path.join(file_path, filename)

    # Generating and writing random values to the file
    with open(top_architecture_random_inputs_file_path, 'w') as file:
        for _ in range(num_lines):
            a_bin = generate_binary_32bit()
            b_bin = generate_binary_32bit()
            c_bin = generate_binary_32bit()
            d_bin = generate_binary_32bit()
            sel1_bin = generate_binary_8bit()
            sel2_bin = generate_binary_8bit()
            write_to_file(
                file=file,
                a_bin=a_bin,
                b_bin=b_bin,
                c_bin=c_bin,
                d_bin=d_bin,
                sel1_bin=sel1_bin,
                sel2_bin=sel2_bin
            )
    ...

```

Nello specifico, gli input generati verranno salvati all'interno di un file .txt nel seguente modo:

```
aTB <= "01001011000010011011010100001101";
bTB <= "11111110001011101001001100111111";
cTB <= "00110110110011011111010101011111";
dTB <= "00110111010000100011101011011101";
sel1TB <= "01011011";
sel2TB <= "11001100";
wait for T_CLK;
```

In particolare, saranno generati input random nel modo sopra allegato per un numero di volt pari a *num_lines*.

2. Non-Optimized Architecture Analysis and Design

L'architettura non ottimizzata, precedentemente descritta, è possibile consultarla nel file " \progetti\progetto-3\vivado\not_optimized\src\TopArchitecture.vhd". Qui di seguito viene riportato un estratto di codice relativo alla descrizione tramite linguaggio di descrizione dell'hardware VHDL dell'architettura corrispondente.

```
...
-- Register Inputs definitions
-- Inputs
RegA : Reg generic map(nBitsInputs) port map(a, clk, rst, aReg);
RegB : Reg generic map(nBitsInputs) port map(b, clk, rst, bReg);
RegC : Reg generic map(nBitsInputs) port map(c, clk, rst, cReg);
RegD : Reg generic map(nBitsInputs) port map(d, clk, rst, dReg);
-- Sels
RegSel1 : Reg generic map(nBitsSels) port map(sel1, clk, rst, sel1Reg);
RegSel2 : Reg generic map(nBitsSels) port map(sel2, clk, rst, sel2Reg);

-- RCAunsigned Sels definition
RCAunsignedSel1Sel2 : RCAunsigned generic map(nBitsSels) port map(sel1Reg, sel2Reg, sel12RCA);

-- ParityCheck definition
ParityCheckSel12 : ParityCheck generic map(nBitsSels+1) port map(sel12RCA, selRCAInputs);

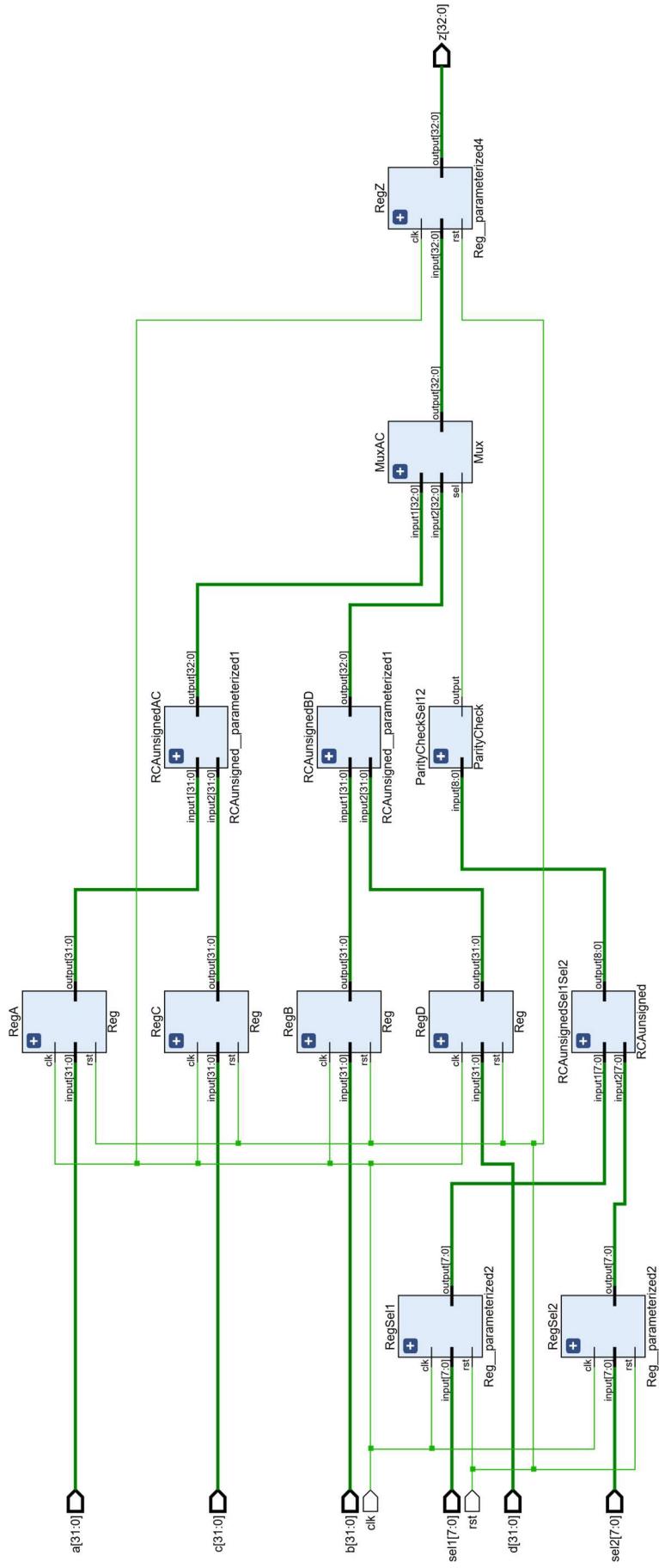
-- AC and BD RCAunsigned definitions
RCAunsignedAC : RCAunsigned generic map(nBitsInputs) port map(aReg, cReg, acRCA);
RCAunsignedBD : RCAunsigned generic map(nBitsInputs) port map(bReg, dReg, bdRCA);

-- Mux definition
MuxAC : Mux generic map(nBitsInputs+1) port map(acRCA, bdRCA, selRCAInputs, acbdMux);

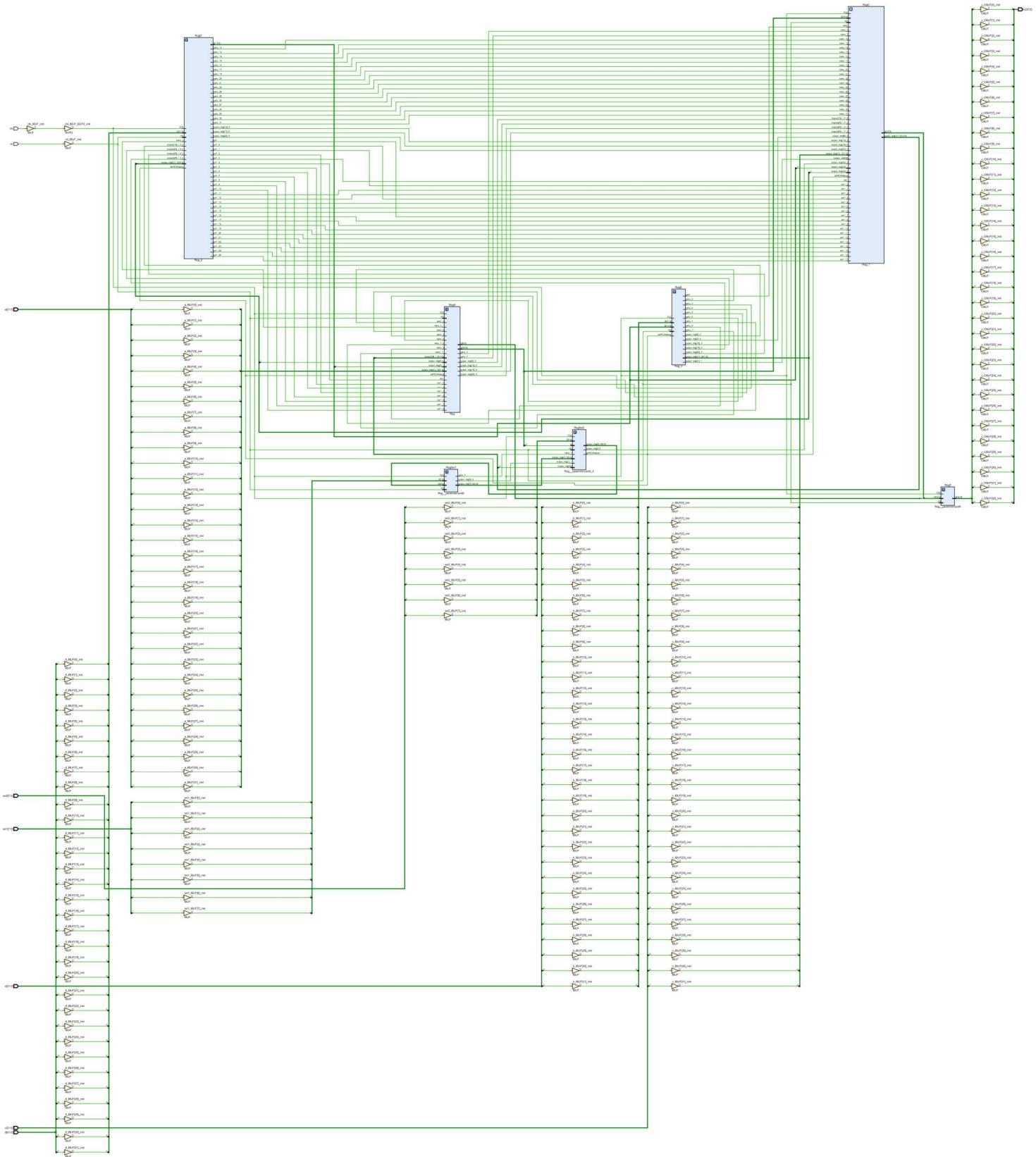
-- Register Output definition
RegZ : Reg generic map(nBitsInputs+1) port map(acbdMux, clk, rst, zReg);

-- Signal result allocation
z <= zReg;
...
```

2.1. RTL Description and Schematic

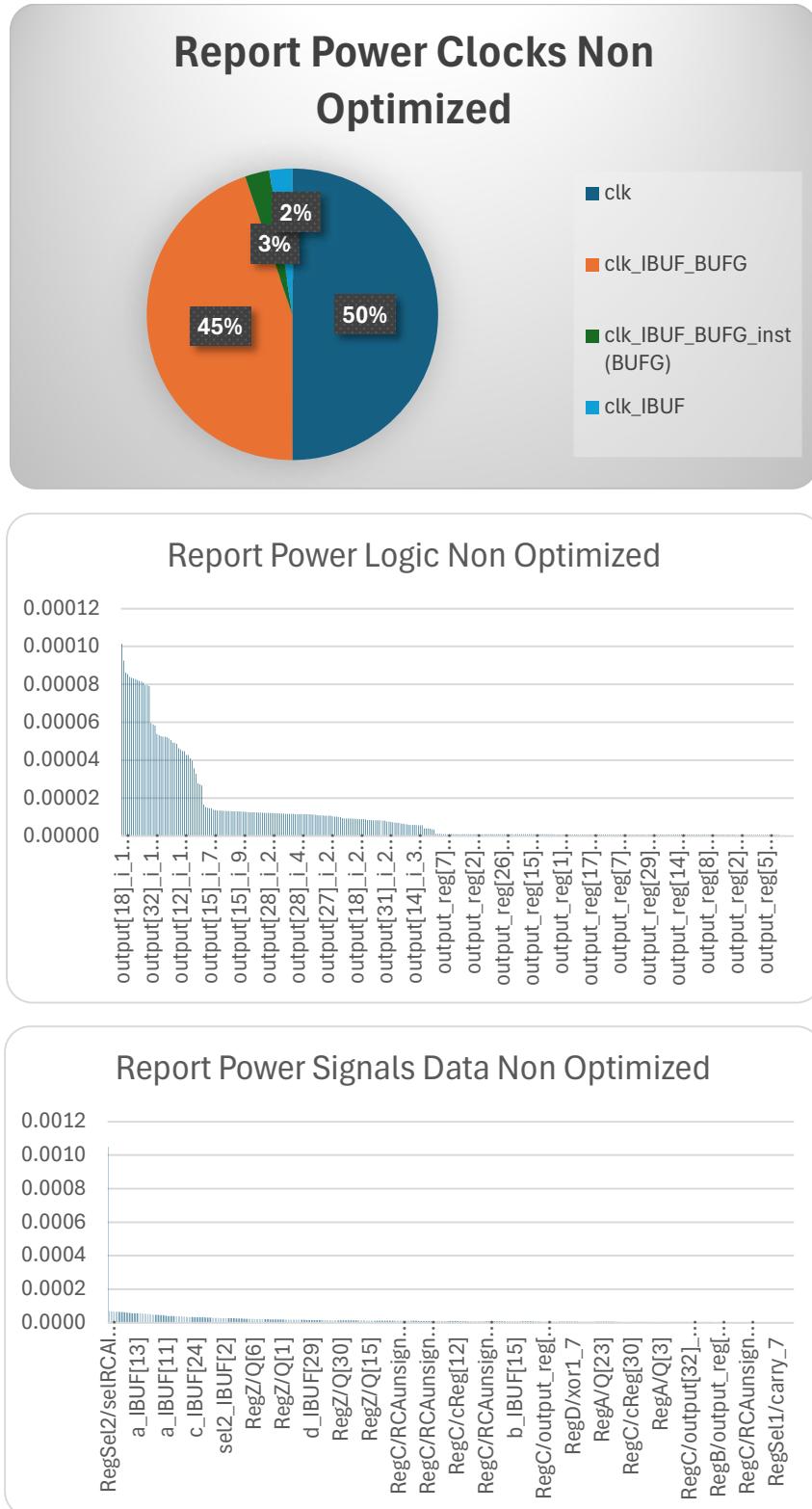


2.2. Synthesis



2.3. Power Analysis

Dopo aver effettuato l'implementazione e aver aggiunto al progetto il file .saif, generato in seguito alla simulazione Post-Implementation Timing Simulation dove sono stati considerati input random, vengono qui di seguito allegati i plot relativi ai report di potenza corrispondenti rispettivamente ai segnali di clock relativi ai FF, ai segnali di logica e ai segnali di dati relativi alle interconnessioni.



2.4. Timing Analysis

Qui di seguito viene riportata la tabella rappresentante la timing analysis relativa all'architettura non ottimizzata.

Worst Negative Slack WNS [ns]	Worst Hold Slack WHS [ns]
1.009	0.147

Nello specifico, con il termine slack ci si riferisce al margine temporale disponibile per completare un path critico all'interno del circuito. Esso indica la differenza tra il tempo disponibile per completare il percorso e il tempo effettivamente richiesto per farlo. Uno slack positivo indica che il percorso è temporizzato correttamente, mentre uno slack negativo indica che il percorso non rispetta i requisiti di timing.

Il WNS, cioè il Worst Negative Slack, indica la peggiore differenza temporale tra il tempo disponibile e il tempo necessario per completare un path critico all'interno del circuito. In particolare, indica il path con il margine di slack più basso, cioè il percorso che è più critico dal punto di vista temporale. Può essere positivo se il path viene completato prima del tempo previsto oppure negativo se il percorso richiede più tempo di quello disponibile.

Per quanto riguarda il margine di hold, esso indica una misura del margine temporale disponibile per mantenere stabile il dato su un percorso di temporizzazione. Nello specifico, esso indica la differenza tra il tempo minimo di hold (il tempo minimo che un dato deve essere stabile prima del fronte di clock successivo) e il tempo effettivo di hold ottenuto dal percorso. Un hold slack positivo indica che il dato è stabile per un periodo di tempo maggiore del minimo richiesto, mentre un hold slack negativo indica che il dato non è stabile per il tempo minimo richiesto.

Il WHS, cioè il Worst Hold Slack, indica la peggiore differenza temporale tra il tempo minimo di hold e il tempo effettivo di hold ottenuto per tutti i percorsi di temporizzazione all'interno del circuito. Pertanto, WHS indica il percorso con il margine di hold più basso, cioè il percorso che è più critico dal punto di vista del mantenimento del dato stabile tra due clock consecutivi. Come il WNS, il WHS può essere positivo se il percorso rispetta il tempo minimo di hold, oppure negativo se il percorso non soddisfa il requisito di hold.

2.5. Maximum Frequency Analysis

L'analisi della frequenza massima è relativa alla possibile frequenza che il circuito può raggiungere facendo alcune considerazioni riguardo parametri relativi all'architettura non ottimizzata.

$$t_{\text{setup}} = 1.009 \text{ ns}$$

$$t_{\text{max}} = T_{\text{clock}} - t_{\text{setup}} = (10 - 1.009)\text{ns} = 8.991 \text{ ns}$$

$$f_{\text{max}} = \frac{1}{t_{\text{max}}} = \frac{1}{8.991 \text{ ns}} = \frac{1}{8.991 \times 10^{-9} \text{ s}} = 0.111222333 \times 10^9 \text{ Hz} = 0.111222333 \text{ GHz}$$

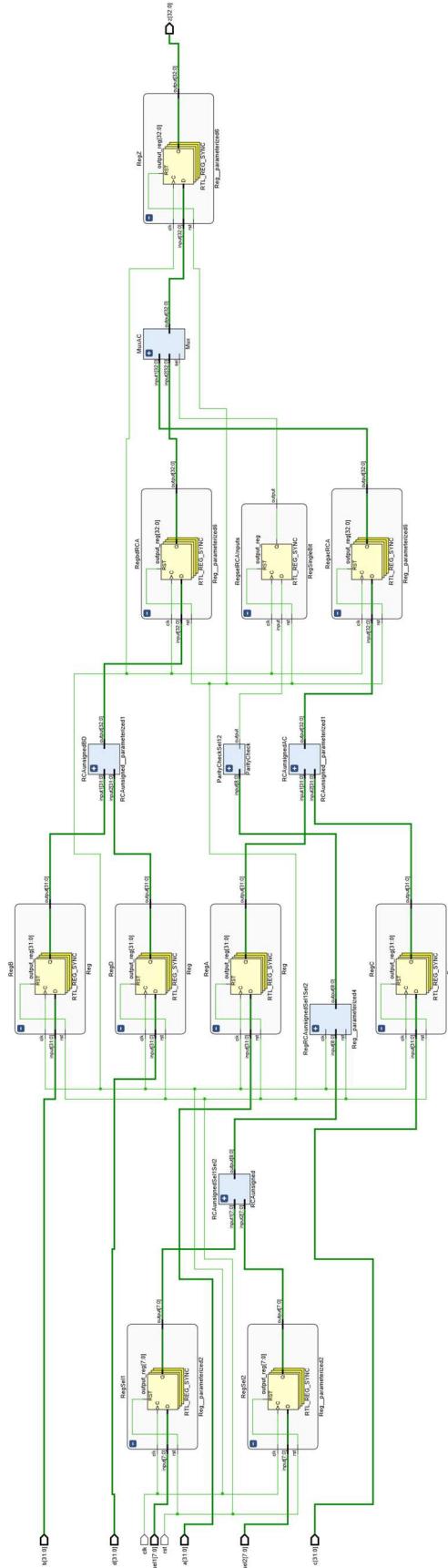
2.6. Resources Utilization Analysis

Qui di seguito viene riportata la tabella rappresentante l'utilizzazione delle risorse relativa all'architettura non ottimizzata.

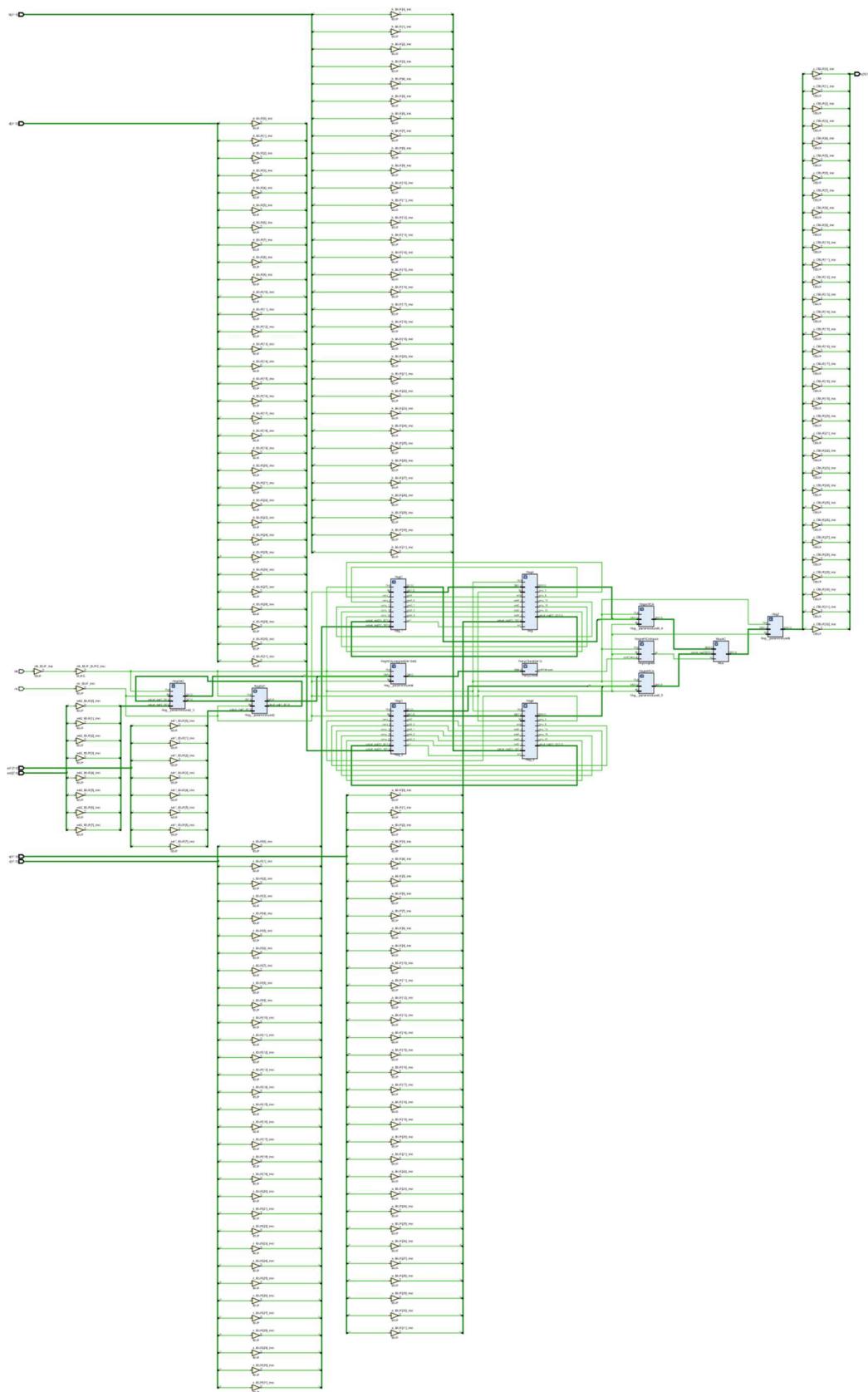
Resource	Utilization	Available	Utilization %
LUT	132	53200	0.25
FF	177	106400	0.17
IO	179	200	89.50
BUFG	1	32	3.13

3. Registering Technique Analysis and Design

3.1. RTL Description and Schematic

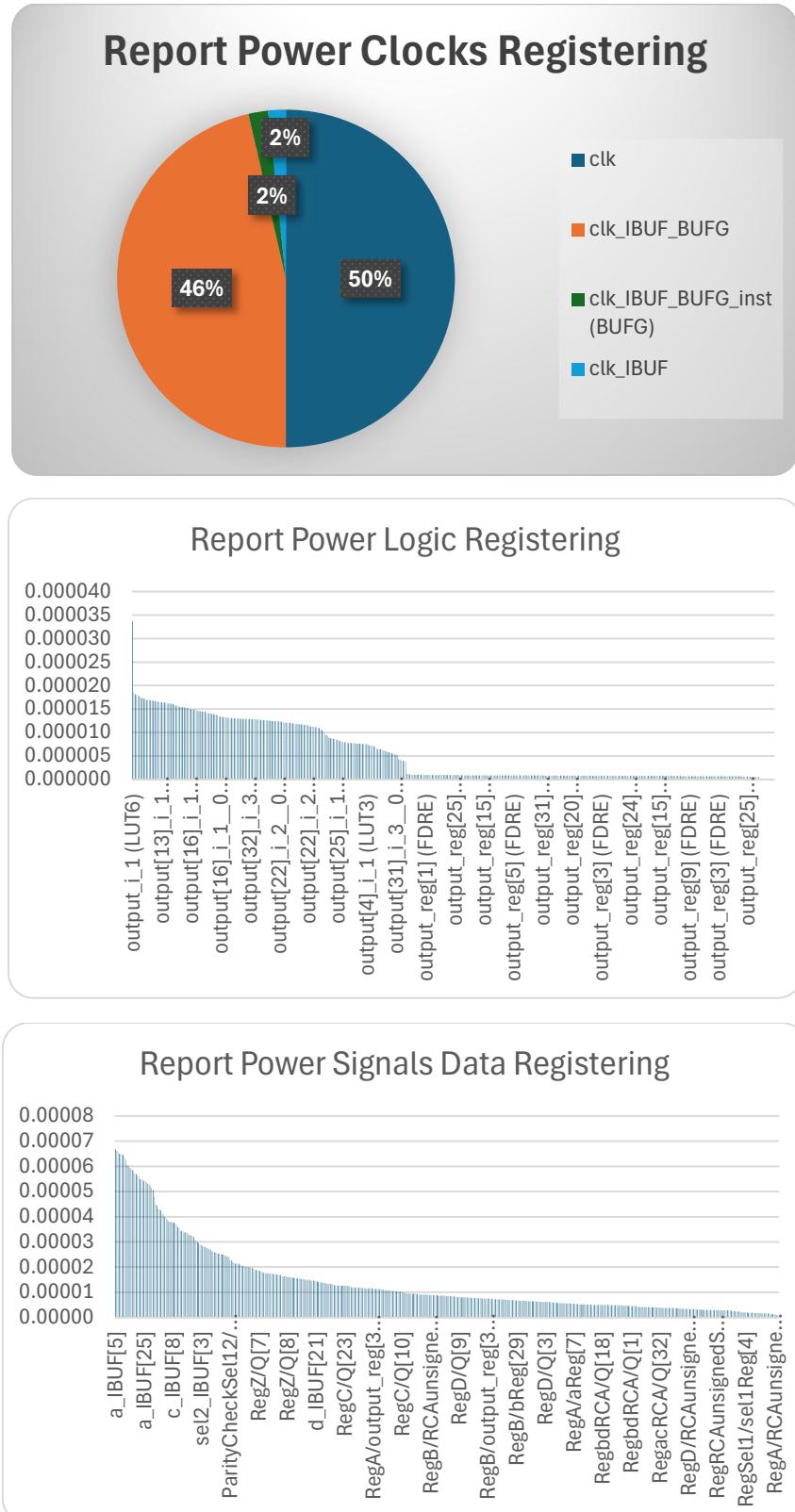


3.2. Synthesis



3.3. Power Analysis

Dopo aver effettuato l'implementazione e aver aggiunto al progetto il file .saif, generato in seguito alla simulazione Post-Implementation Timing Simulation dove sono stati considerati input random, vengono qui di seguito allegati i plot relativi ai report di potenza corrispondenti rispettivamente ai segnali di clock relativi ai FF, ai segnali di logica e ai segnali di dati relativi alle interconnessioni.



3.4. Timing Analysis

Qui di seguito viene riportata la tabella rappresentante la timing analysis relativa all'architettura ottimizzata con tecnica di registering.

Worst Negative Slack WNS [ns]	Worst Hold Slack WHS [ns]
1.052	0.12

3.5. Frequency Analysis

L'analisi della frequenza massima è relativa alla possibile frequenza che il circuito può raggiungere facendo alcune considerazioni riguardo parametri relativi all'architettura ottimizzata con tecnica di registering.

$$t_{setup} = 1.052 \text{ ns}$$

$$t_{max} = T_{clock} - t_{setup} = (10 - 1.052)\text{ns} = 8.948 \text{ ns}$$

$$f_{max} = \frac{1}{t_{max}} = \frac{1}{8.948 \text{ ns}} = \frac{1}{8.948 \times 10^{-9} \text{ s}} = 0.111756817 \times 10^9 \text{ Hz} = 0.111756817 \text{ GHz}$$

3.6. Resources Utilization Analysis

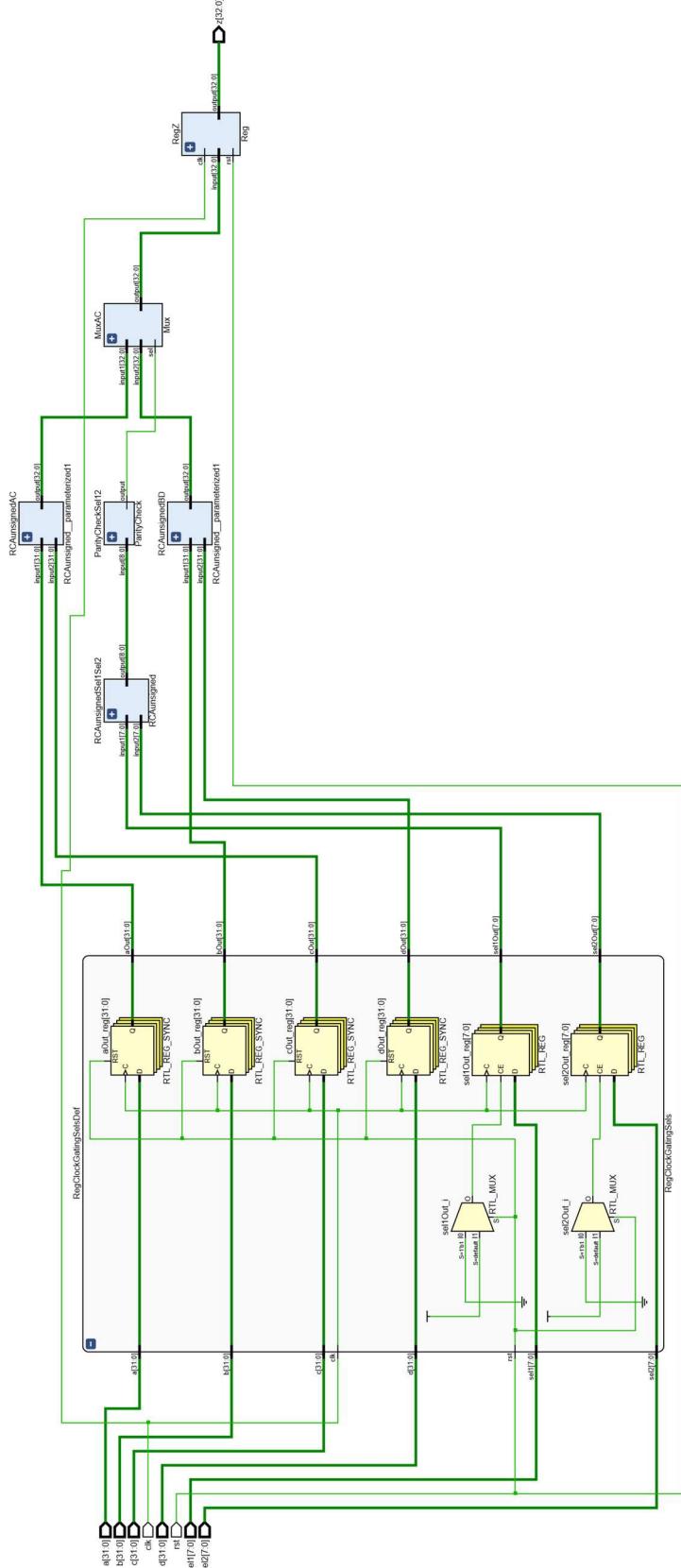
Qui di seguito viene riportata la tabella rappresentante l'utilizzazione delle risorse relativa all'architettura ottimizzata tramite tecnica di registering.

Resource	Utilization	Available	Utilization %
LUT	132	53200	0.25
FF	253	106400	0.24
IO	179	200	89.50
BUFG	1	32	3.13

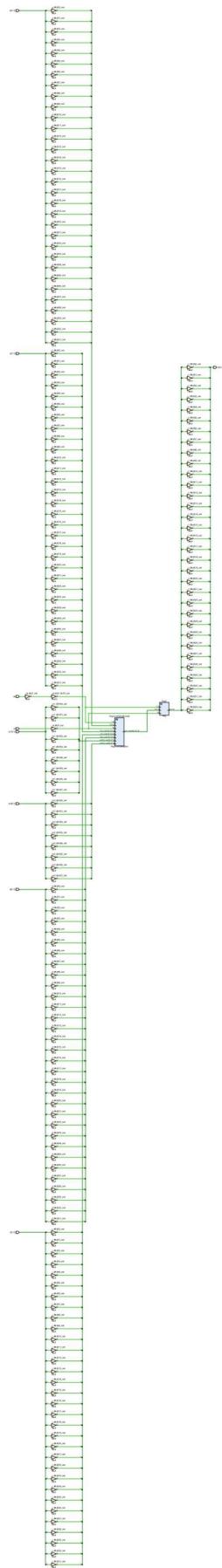
4. Clock Gating Technique Analysis and Design

4.1. Selectors Clock Gating

4.1.1. RTL Description and Schematic

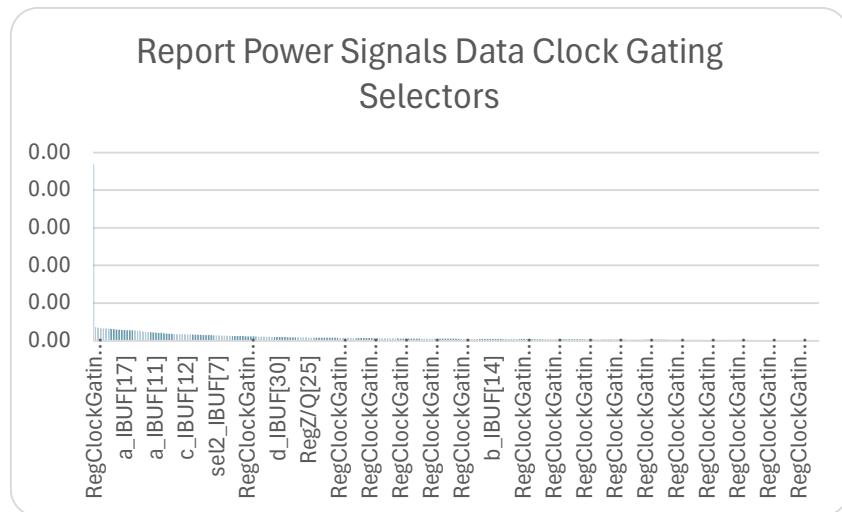
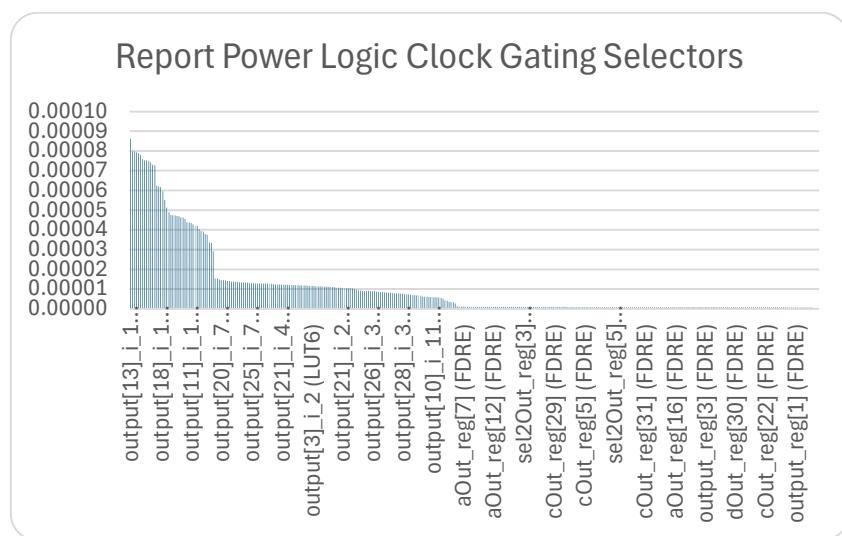
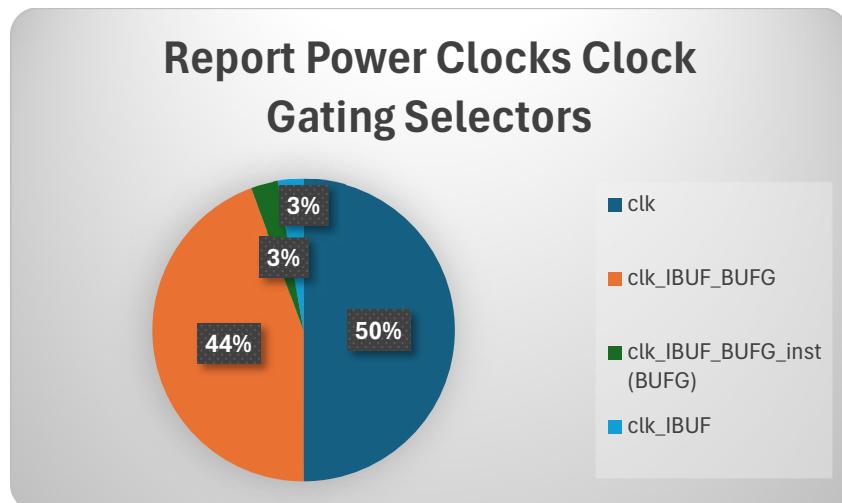


4.1.2. Synthesis



4.1.3. Power Analysis

Dopo aver effettuato l'implementazione e aver aggiunto al progetto il file .saif, generato in seguito alla simulazione Post-Implementation Timing Simulation dove sono stati considerati input random, vengono qui di seguito allegati i plot relativi ai report di potenza corrispondenti rispettivamente ai segnali di clock relativi ai FF, ai segnali di logica e ai segnali di dati relativi alle interconnessioni.



4.1.4. Timing Analysis

Qui di seguito viene riportata la tabella rappresentante la timing analysis relativa all'architettura ottimizzata con tecnica di clock gating applicata ai selettori.

Worst Negative Slack WNS [ns]	Worst Hold Slack WHS [ns]
1.383	0.224

4.1.5. Frequency Analysis

L'analisi della frequenza massima è relativa alla possibile frequenza che il circuito può raggiungere facendo alcune considerazioni riguardo parametri relativi all'architettura ottimizzata con tecnica di clock gating applicata ai selettori.

$$t_{setup} = 1.383 \text{ ns}$$

$$t_{max} = T_{clock} - t_{setup} = (10 - 1.383) \text{ ns} = 8.617 \text{ ns}$$

$$f_{max} = \frac{1}{t_{max}} = \frac{1}{8.617 \text{ ns}} = \frac{1}{8.617 \times 10^{-9} \text{ s}} = 0.116049669 \times 10^9 \text{ Hz} = 0.116049669 \text{ GHz}$$

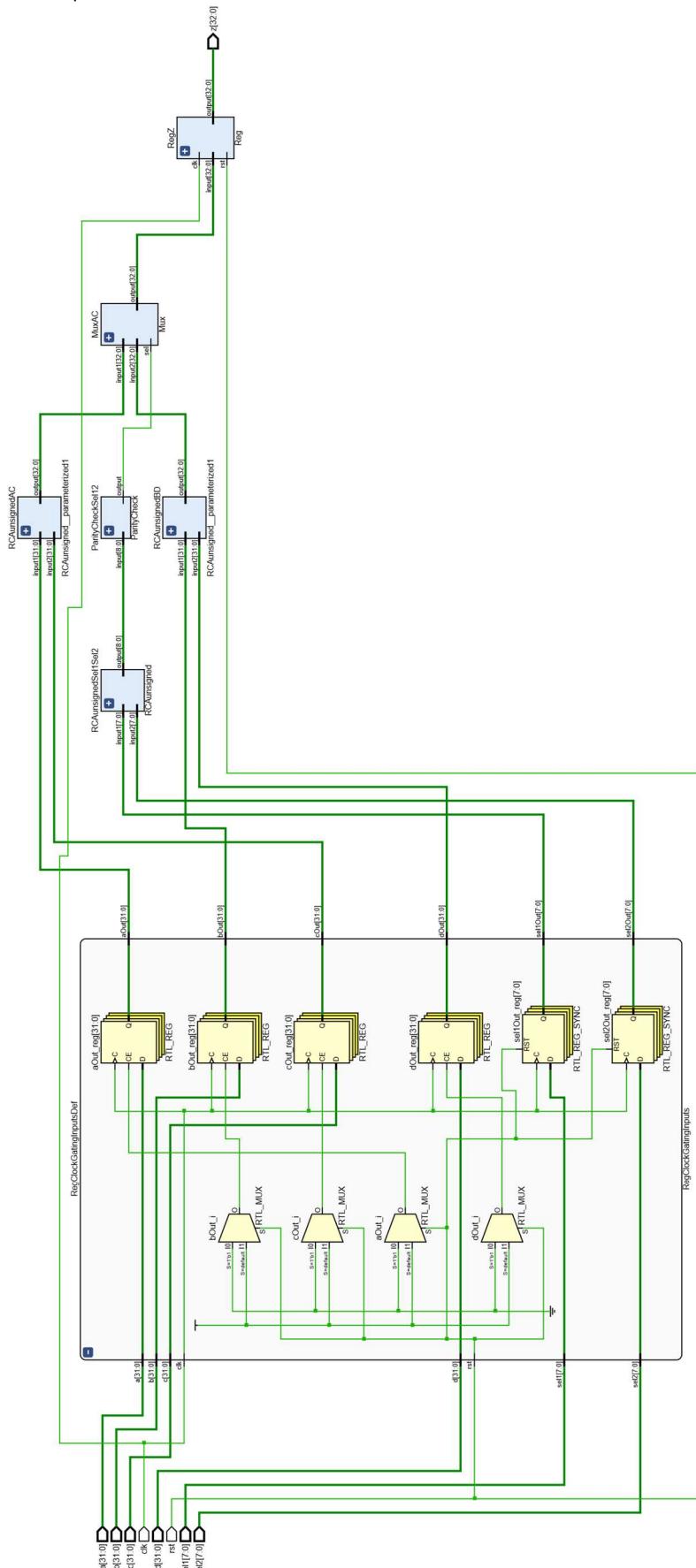
4.1.6. Resources Utilization Analysis

Qui di seguito viene riportata la tabella rappresentante l'utilizzazione delle risorse relativa all'architettura ottimizzata tramite tecnica di clock gating applicata ai selettori.

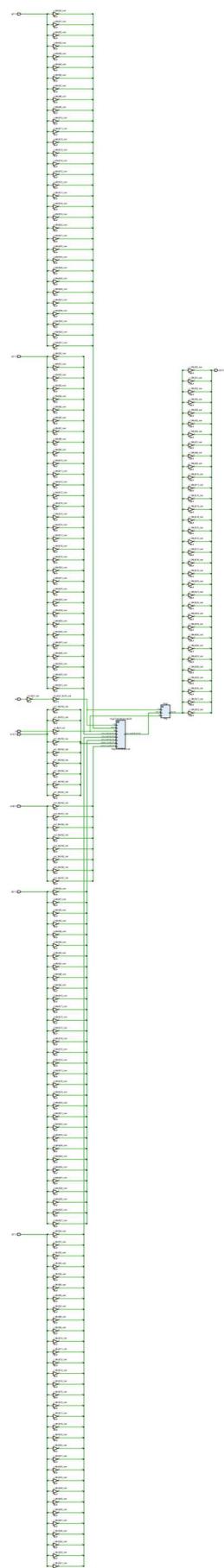
Resource	Utilization	Available	Utilization %
LUT	127	53200	0.24
FF	177	106400	0.17
IO	179	200	89.50
BUFG	1	32	3.13

4.2. Inputs Clock Gating

4.2.1. RTL Description and Schematic

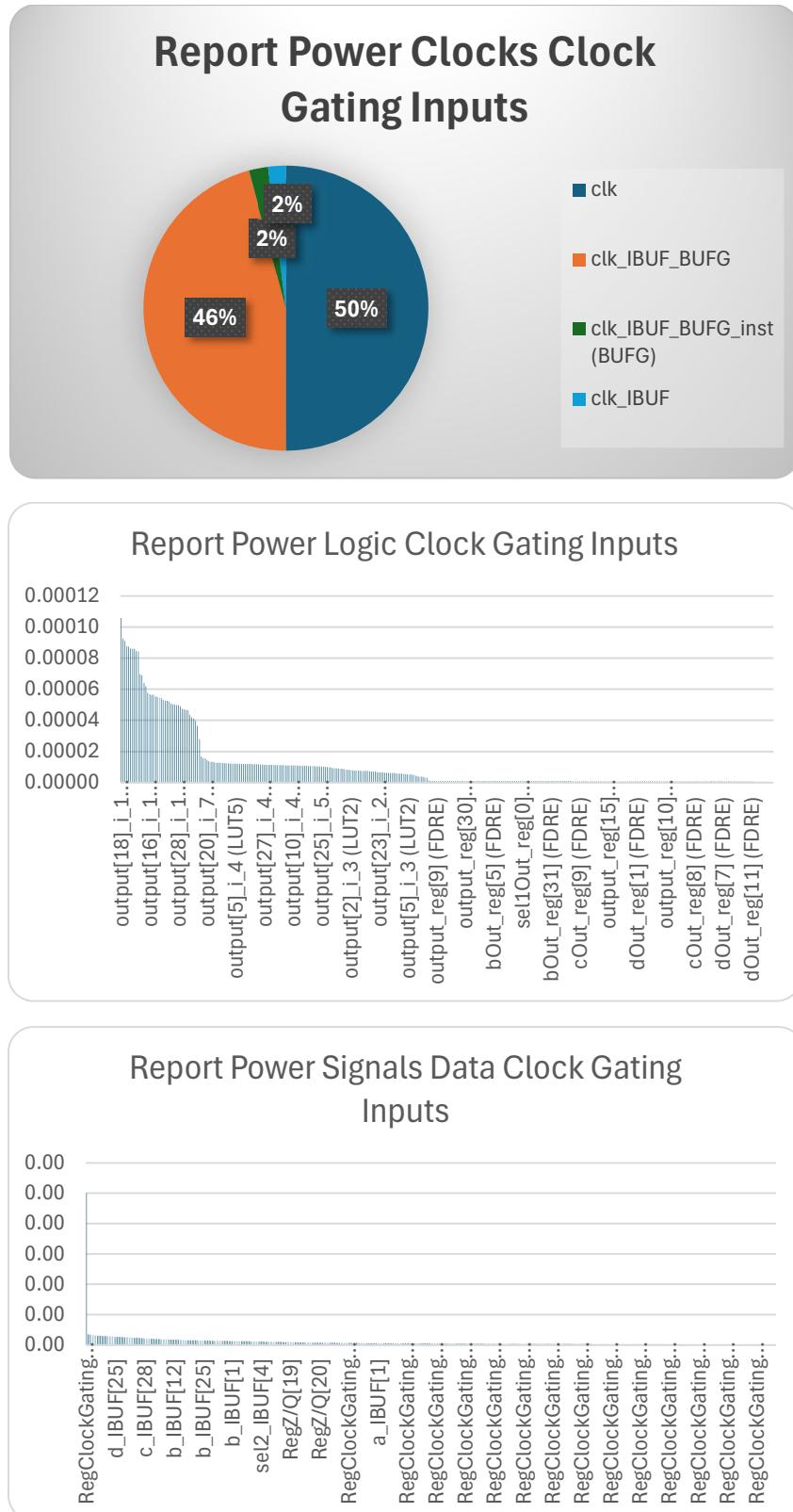


4.2.2. Synthesis



4.2.3. Power Analysis

Dopo aver effettuato l'implementazione e aver aggiunto al progetto il file .saif, generato in seguito alla simulazione Post-Implementation Timing Simulation dove sono stati considerati input random, vengono qui di seguito allegati i plot relativi ai report di potenza corrispondenti rispettivamente ai segnali di clock relativi ai FF, ai segnali di logica e ai segnali di dati relativi alle interconnessioni.



4.2.4. Timing Analysis

Qui di seguito viene riportata la tabella rappresentante la timing analysis relativa all'architettura ottimizzata con tecnica di clock gating applicata agli input A, B, C e D.

Worst Negative Slack WNS [ns]	Worst Hold Slack WHS [ns]
1.499	0.163

4.2.5. Frequency Analysis

L'analisi della frequenza massima è relativa alla possibile frequenza che il circuito può raggiungere facendo alcune considerazioni riguardo parametri relativi all'architettura ottimizzata con tecnica di clock gating applicata agli input A, B, C e D.

$$t_{setup} = 1.499 \text{ ns}$$

$$t_{max} = T_{clock} - t_{setup} = (10 - 1.499) \text{ ns} = 8.501 \text{ ns}$$

$$f_{max} = \frac{1}{t_{max}} = \frac{1}{8.501 \text{ ns}} = \frac{1}{8.501 \times 10^{-9} \text{ s}} = 0.117633219 \times 10^9 \text{ Hz} = 0.117633219 \text{ GHz}$$

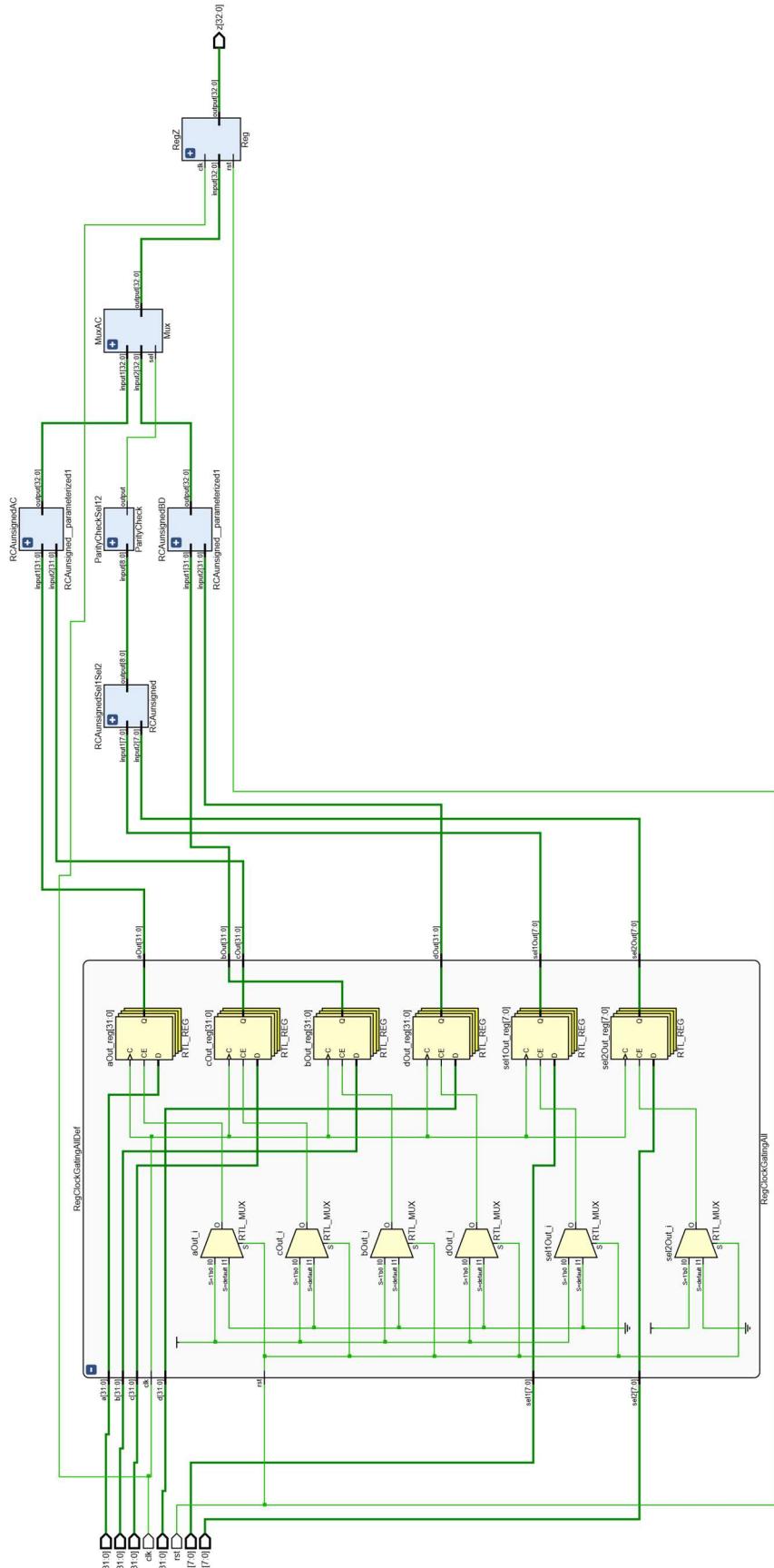
4.2.6. Resources Utilization Analysis

Qui di seguito viene riportata la tabella rappresentante l'utilizzazione delle risorse relativa all'architettura ottimizzata tramite tecnica di clock gating applicata agli input A, B, C e D.

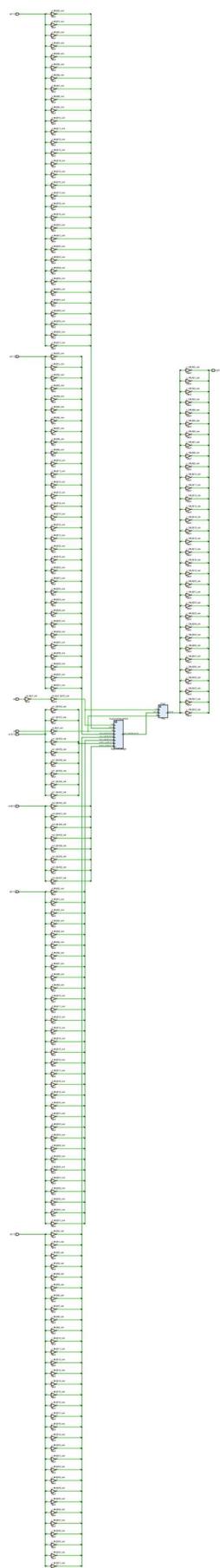
Resource	Utilization	Available	Utilization %
LUT	125	53200	0.23
FF	177	106400	0.17
IO	179	200	89.50
BUFG	1	32	3.13

4.3. All Clock Gating

4.3.1. RTL Description and Schematic

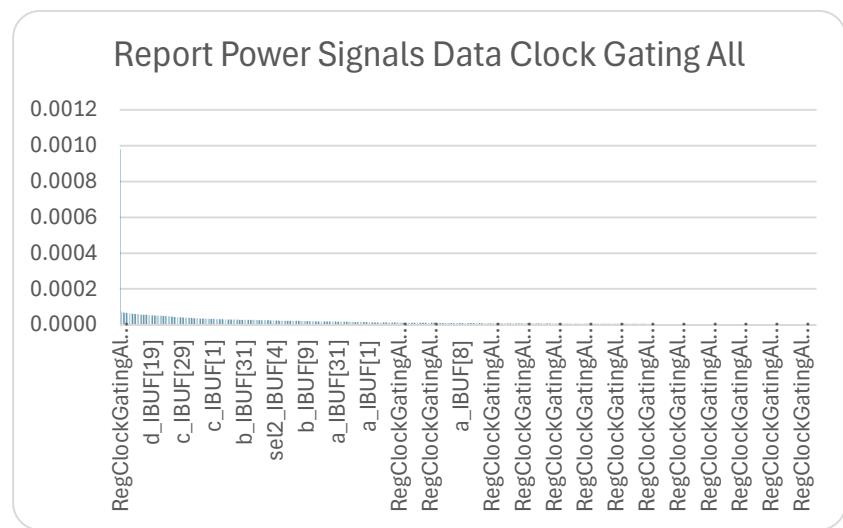
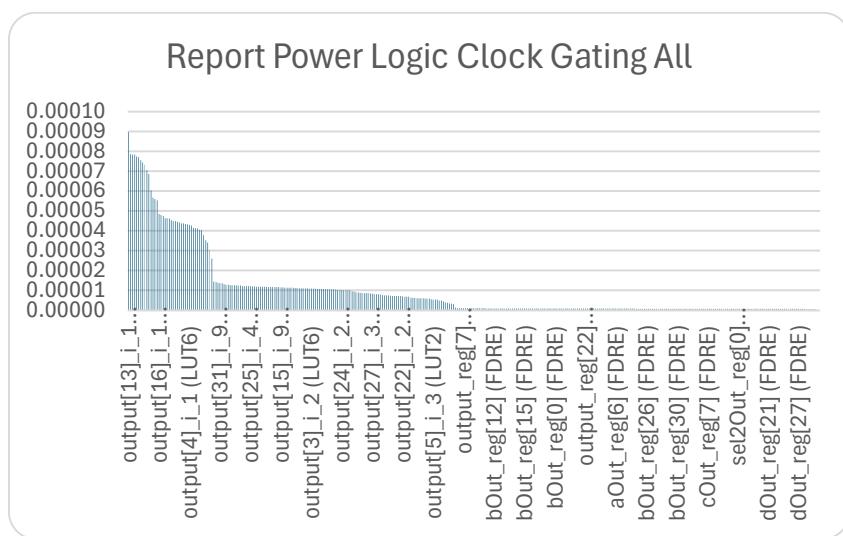
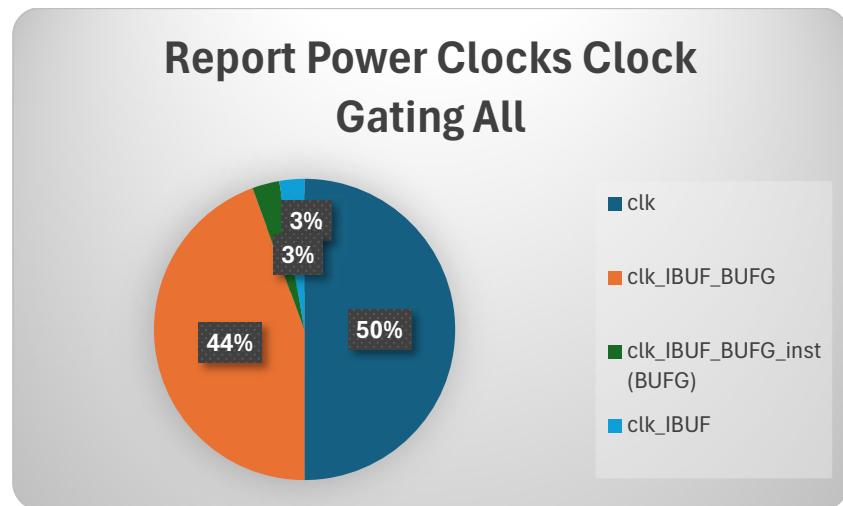


4.3.2. Synthesis



4.3.3. Power Analysis

Dopo aver effettuato l'implementazione e aver aggiunto al progetto il file .saif, generato in seguito alla simulazione Post-Implementation Timing Simulation dove sono stati considerati input random, vengono qui di seguito allegati i plot relativi ai report di potenza corrispondenti rispettivamente ai segnali di clock relativi ai FF, ai segnali di logica e ai segnali di dati relativi alle interconnessioni.



4.3.4. Timing Analysis

Qui di seguito viene riportata la tabella rappresentante la timing analysis relativa all'architettura ottimizzata con tecnica di clock gating applicata sia ai selettori sia agli input A, B, C e D.

Worst Negative Slack WNS [ns]	Worst Hold Slack WHS [ns]
1.107	0.202

4.3.5. Frequency Analysis

L'analisi della frequenza massima è relativa alla possibile frequenza che il circuito può raggiungere facendo alcune considerazioni riguardo parametri relativi all'architettura ottimizzata con tecnica diclock gating applicata sia ai selettori sia agli input A, B, C e D.

$$t_{setup} = 1.107 \text{ ns}$$

$$t_{max} = T_{clock} - t_{setup} = (10 - 1.107) \text{ ns} = 8.893 \text{ ns}$$

$$f_{max} = \frac{1}{t_{max}} = \frac{1}{8.893 \text{ ns}} = \frac{1}{8.893 \times 10^{-9} \text{ s}} = 0.112447992 \times 10^9 \text{ Hz} = 0.112447992 \text{ GHz}$$

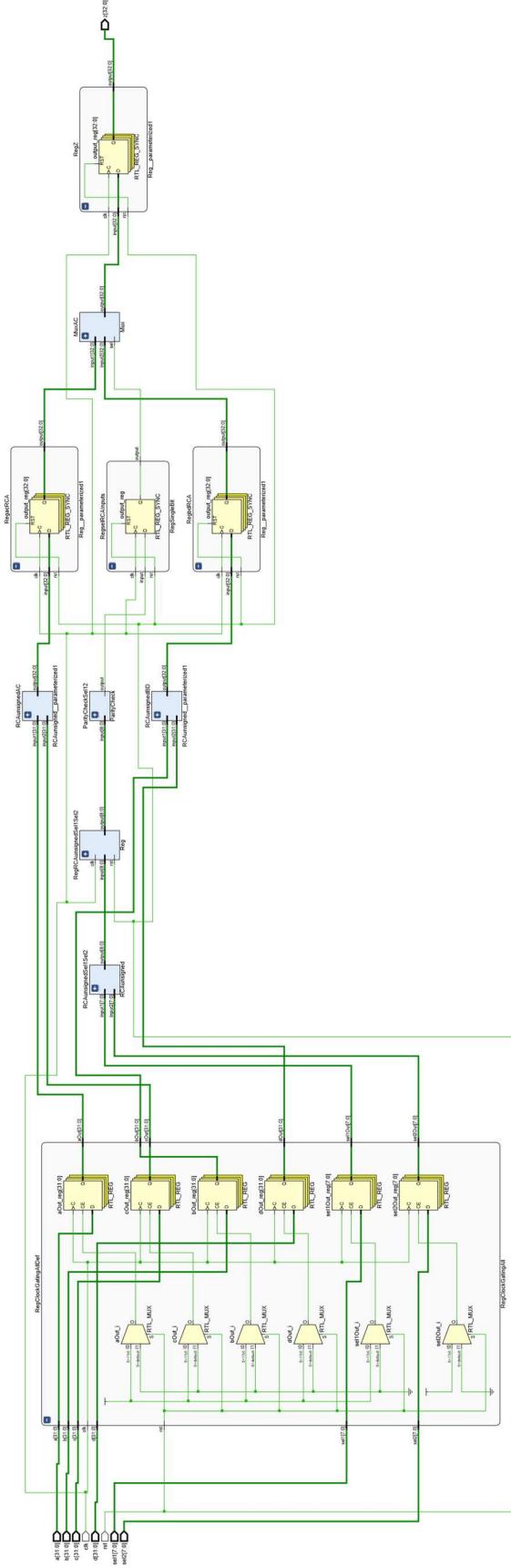
4.3.6. Resources Utilization Analysis

Qui di seguito viene riportata la tabella rappresentante l'utilizzazione delle risorse relativa all'architettura ottimizzata tramite tecnica di clock gating applicata sia ai selettori sia agli input A, B, C e D.

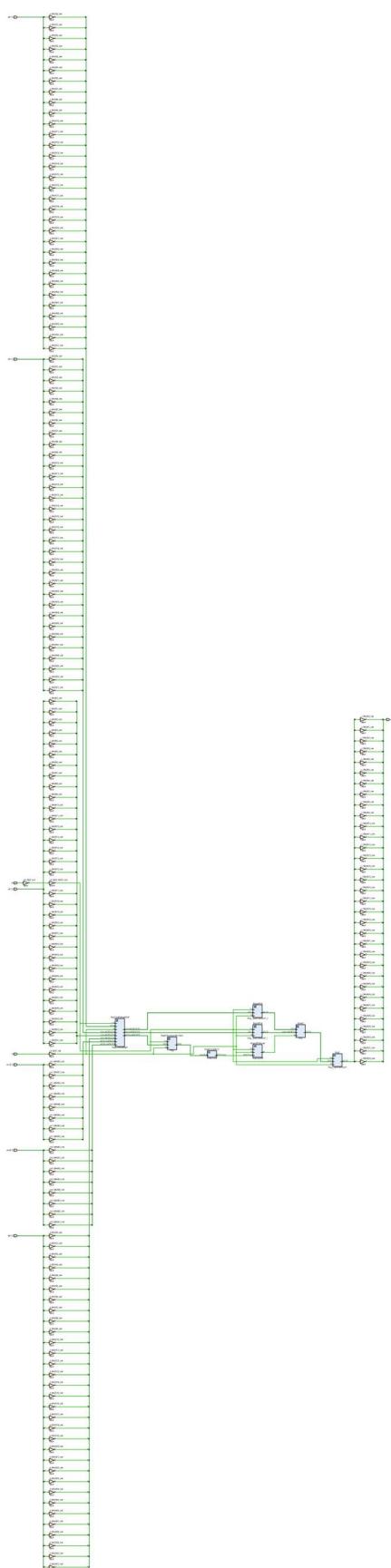
Resource	Utilization	Available	Utilization %
LUT	125	53200	0.23
FF	177	106400	0.17
IO	179	200	89.50
BUFG	1	32	3.13

5. Hybrid Technique Analysis and Design

5.1. RTL Description and Schematic

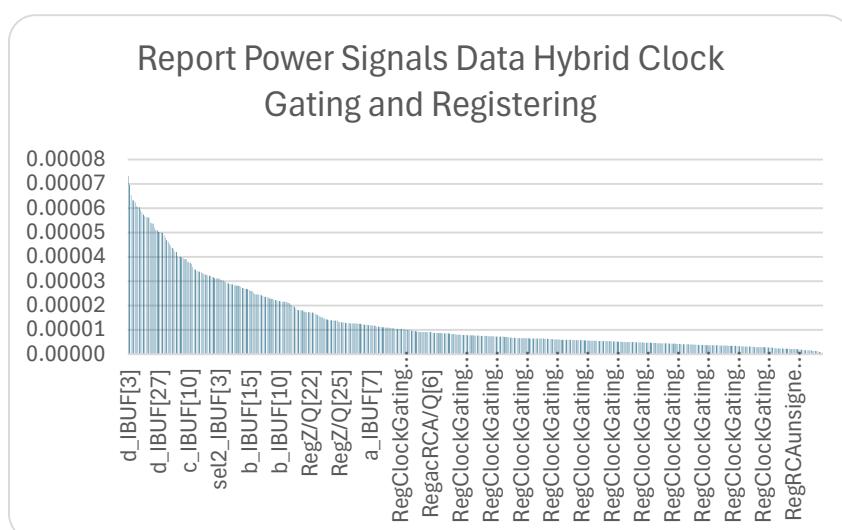
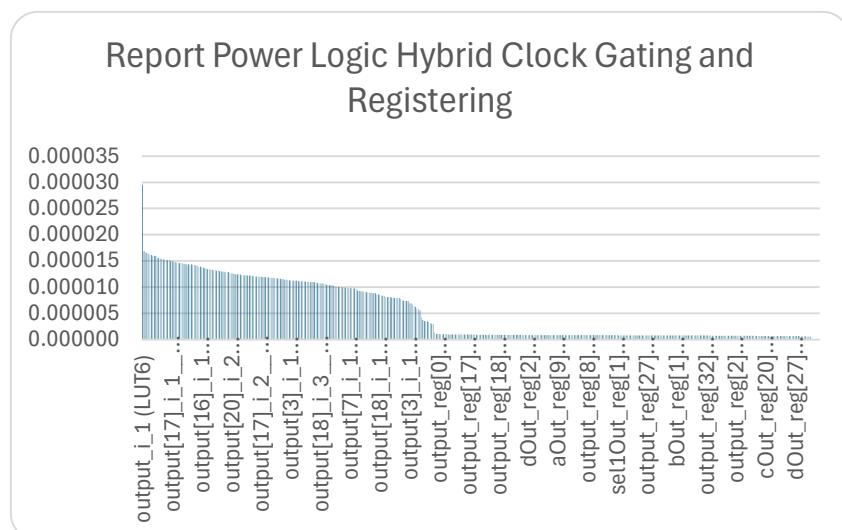
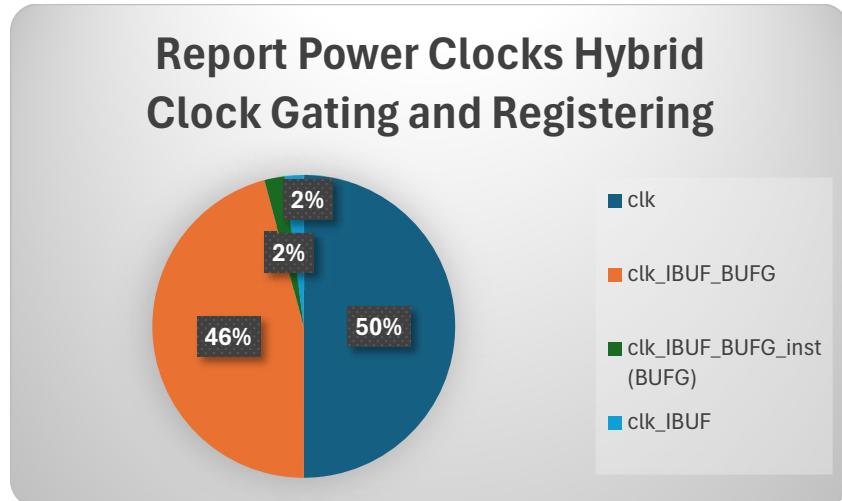


5.2. Synthesis



5.3. Power Analysis

Dopo aver effettuato l'implementazione e aver aggiunto al progetto il file .saif, generato in seguito alla simulazione Post-Implementation Timing Simulation dove sono stati considerati input random, vengono qui di seguito allegati i plot relativi ai report di potenza corrispondenti rispettivamente ai segnali di clock relativi ai FF, ai segnali di logica e ai segnali di dati relativi alle interconnessioni.



5.4. Timing Analysis

Qui di seguito viene riportata la tabella rappresentante la timing analysis relativa all'architettura ottimizzata con tecnica di registering e di clock gating applicata sia ai selettori sia agli input A, B, C e D.

Worst Negative Slack WNS [ns]	Worst Hold Slack WHS [ns]
1.107	0.117

5.5. Frequency Analysis

L'analisi della frequenza massima è relativa alla possibile frequenza che il circuito può raggiungere facendo alcune considerazioni riguardo parametri relativi all'architettura ottimizzata con tecnica di registering e di clock gating applicata sia ai selettori sia agli input A, B, C e D.

$$t_{setup} = 1.107 \text{ ns}$$

$$t_{max} = T_{clock} - t_{setup} = (10 - 1.107) \text{ ns} = 8.893 \text{ ns}$$

$$f_{max} = \frac{1}{t_{max}} = \frac{1}{8.893 \text{ ns}} = \frac{1}{8.893 \times 10^{-9} \text{ s}} = 0.112447992 \times 10^9 \text{ Hz} = 0.112447992 \text{ GHz}$$

5.6. Resources Utilization Analysis

Qui di seguito viene riportata la tabella rappresentante l'utilizzazione delle risorse relativa all'architettura ottimizzata tramite tecnica di registering e di clock gating applicata sia ai selettori sia agli input A, B, C e D.

Resource	Utilization	Available	Utilization %
LUT	132	53200	0.25
FF	253	106400	0.24
IO	179	200	89.50
BUFG	1	32	3.13

6. Conclusions

In questo capitolo conclusivo verranno riportati tutti i risultati ottenuti per ogni soluzione architetturale presentata nei capitoli precedenti. Nello specifico, per ognuna verrà mostrata la potenza totale ed ognuna delle sue componenti precedentemente citate, il margine di slack peggiore, la massima frequenza raggiungibile dal circuito e l'utilizzazione delle risorse in termini di LUT e FF usate.

	Non Optimized Design	Registering Design	Clock Gating Selectors Design	Clock Gating Inputs Design	Clock Gating All Design	Hybrid Clock Gating All and Registering Design
Power Analysis						
Clocks Power [mW]	1.710899174	2.528798999	1.588799176	2.039098181	1.488800277	2.010098426
Signals Data Power [mW]	8.201719262	7.142795715	8.104214445	8.084338158	8.016914129	7.060251199
Logic Power [mW]	3.869700013	2.553441096	3.688085126	3.809792688	3.470152849	2.412233967
Total Power [mW]	13.78231845	12.22503581	13.38109875	13.93322903	12.97586726	11.48258359
Timing Analysis						
Clock Constraint [ns]	10	10	10	10	10	10
WNS [ns]	1.009	1.052	1.383	1.499	1.107	1.107
Frequency Analysis						
Maximum Clock Frequency [MHz]	111.2223334	111.7568172	116.0496693	117.6332196	112.4479928	112.4479928
Resources Utilization Analysis						
LUT [#]	132	132	122	125	125	132
FF [#]	177	253	177	177	177	253

Si può notare come la minore potenza totale si ha in corrispondenza della soluzione ibrida dove è stata utilizzata sia la tecnica del registering sia quella del clock gating sui segnali A, B, C, D e selettori. In particolare, considerando l'architettura non ottimizzata iniziale e la soluzione ibrida, la variazione percentuale, associata alla potenza totale, sarà la seguente:

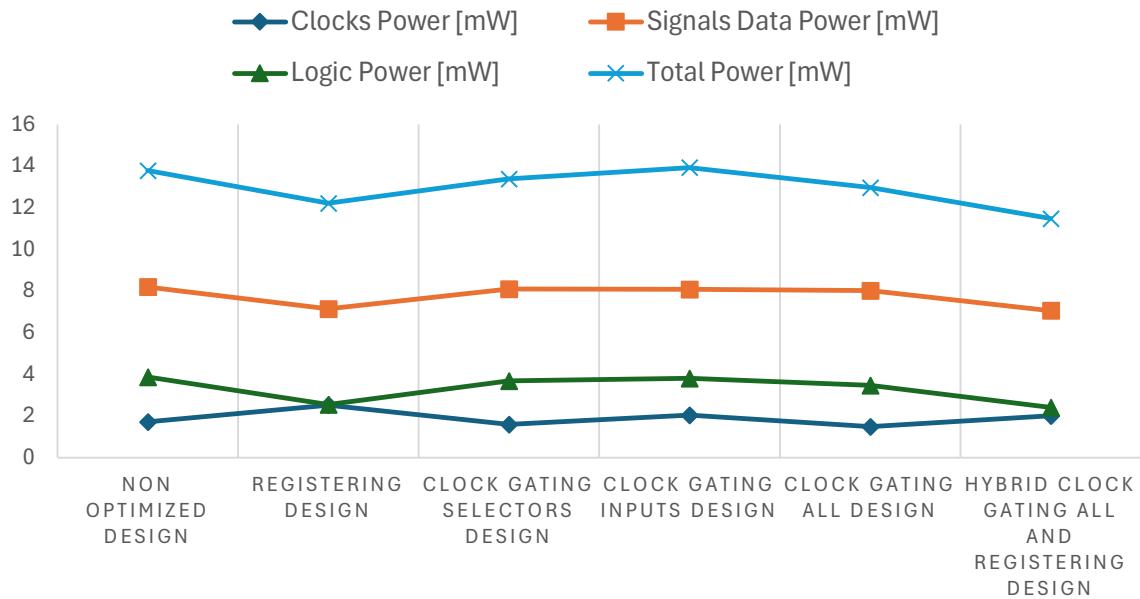
$$\Delta\% = \left(\frac{11.48258359 - 13.78231845}{13.78231845} \right) \cdot 100 \approx -16.686\%$$

Pertanto, adottando la strategia appena citata, si è riusciti a diminuire la potenza totale del 16.686%. Nello specifico, una variazione importante si è avuta in corrispondenza della potenza associata alla logica. Infatti, questa risulta essere diminuita di circa il 37.66%.

Bisogna fare riferimento, ovviamente, anche all'utilizzazione delle risorse in corrispondenza di ogni soluzione implementata. In particolare, la soluzione ibrida presenta un aumento di circa il 42.93% dei FF utilizzati rispetto alla soluzione non ottimizzata iniziale. Questo aumento è dovuto in seguito alla strategia del registering adottata che presenta un utilizzo maggiore dei FF rispetto alle soluzioni che non lo prevedono. Tanto è vero che, in corrispondenza delle soluzioni proposte dove è previsto l'utilizzo della tecnica del registering, l'aumento dei FF impiegati è pressoché il medesimo. Si può notare, inoltre, che la tecnica del clock gating ha permesso una leggera diminuzione, in corrispondenza di alcune soluzioni proposte, del numero delle LUT impiegate.

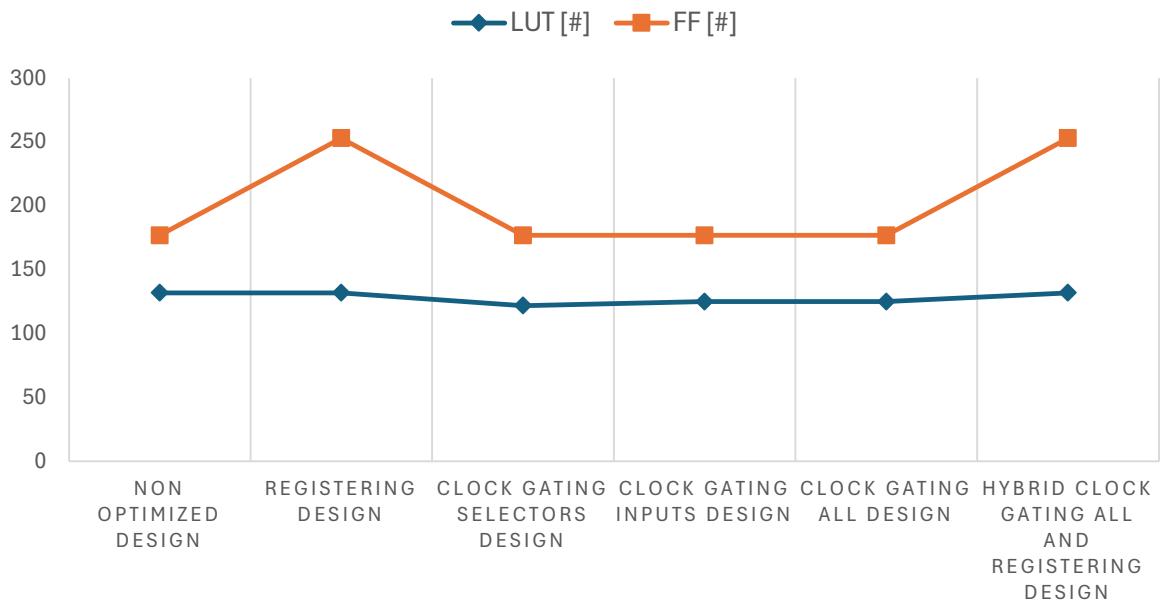
Si può notare, inoltre, che le soluzioni proposte, in particolare la tecnica ibrida e il clock gating sui segnali A, B, C, D e selettori, ha permesso, a parità di periodo di clock considerato, un aumento della frequenza massima di funzionamento.

POWER ANALYSIS



Si può notare come il trend della Signals Data Power risulta essere pressoché lo stesso per le varie soluzioni mentre, come già sopra citato, la Logic Power e la Clocks Power risultano essere ridotte per alcune soluzioni.

RESOURCES UTILIZATION ANALYSIS



Pertanto, dopo aver analizzato nel dettaglio la potenza e l'utilizzazione delle risorse associate alle singole soluzioni proposte, si potrebbe definire una soluzione "migliore" sia dal punto di vista low power sia dal punto di vista dell'utilizzazione delle risorse. In particolare, la strategia ibrida implementata risulta essere la migliore dal punto di vista della potenza associata. Infatti, ad essa corrisponde il valore di potenza minimo registrato tra tutte le soluzioni proposte. Bisogna, comunque, ricordare che tale risultato è stato ottenuto considerando circa il 42.93% in più di risorse. Pertanto, facendo un'analisi puramente low powering, questa soluzione risulta essere la migliore scelta per ottimizzare l'architettura iniziale. Prendendo in considerazione anche l'utilizzazione delle risorse, la strategia del clock gating sui segnali A, B, C, D e selettori, pur presentando un

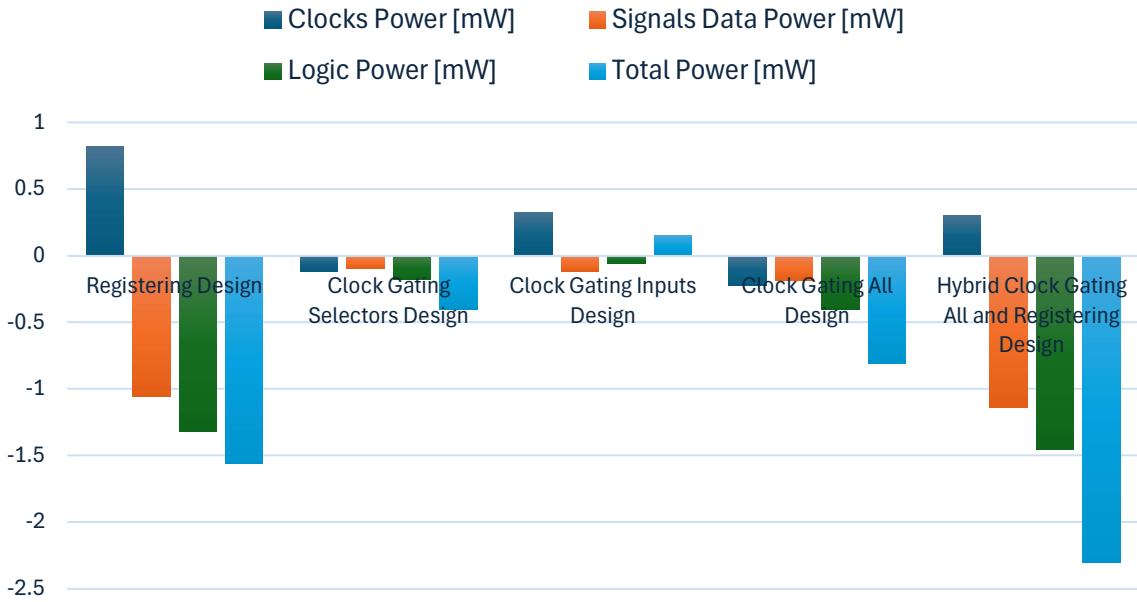
piccolo aumento di dissipazione di potenza rispetto alla soluzione ibrida, permette di avere sia una diminuzione della potenza sia delle risorse rispetto alla soluzione non ottimizzata. Questo aspetto lo si può notare più nel dettaglio nella seguente tabella:

ABSOLUTE ERROR	Non Optimized Design	Registering Design	Clock Gating Selectors Design	Clock Gating Inputs Design	Clock Gating All Design	Hybrid Clock Gating All and Registering Design
Power Analysis						
Clocks Power [mW]	1.710899174	0.817899825	-0.122099998	0.328199007	-0.222098897	0.299199251
Signals Data Power [mW]	8.201719262	-1.058923546	-0.097504817	-0.117381103	-0.184805132	-1.141468063
Logic Power [mW]	3.869700013	-1.316258917	-0.181614887	-0.059907325	-0.399547163	-1.457466045
Total Power [mW]	13.78231845	-1.557282638	-0.401219702	0.150910579	-0.806451193	-2.299734857
Timing Analysis						
Clock Constraint [ns]	10	0	0	0	0	0
WNS [ns]	1.009	0.043	0.374	0.49	0.098	0.098
Frequency Analysis						
Maximum Clock Frequency [MHz]	111.2223334	0.534483721	4.827335814	6.410886177	1.225659359	1.225659359
Resources Utilization Analysis						
LUT [#]	132	0	-10	-7	-7	0
FF [#]	177	76	0	0	0	76

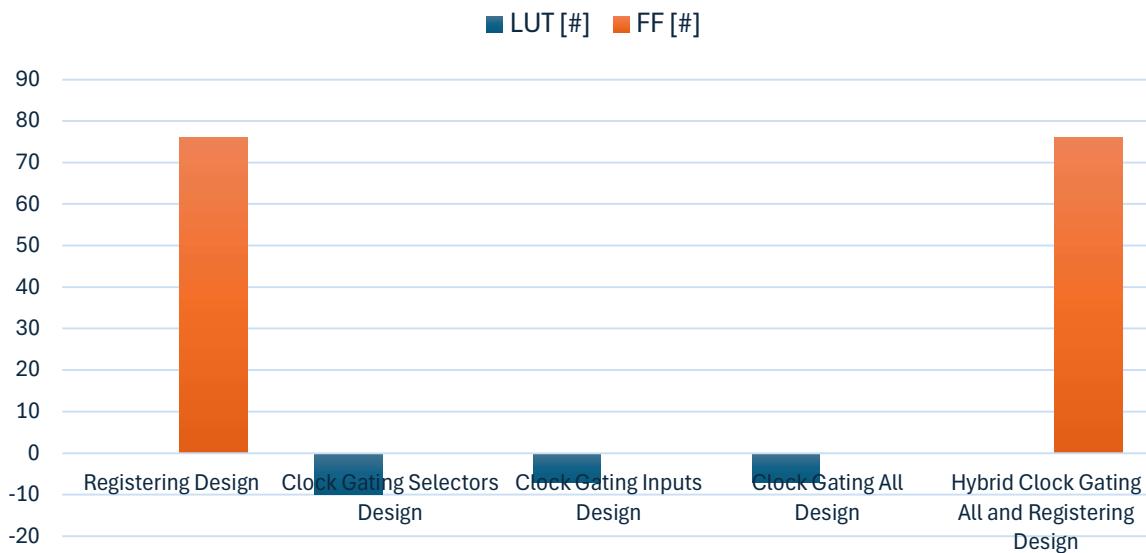
In particolare, la tabella sopra allegata mostra l'errore assoluto rispetto alla prima colonna, rappresentante il design non ottimizzato, che è stata lasciata per semplicità. Nello specifico, i valori negativi relativi alla potenza e all'utilizzazione delle risorse indicano una diminuzione del parametro corrispondente rispetto all'architettura non ottimizzata, cioè significando un miglioramento, mentre i valori positivi indicano un incremento del parametro corrispondente rispetto al design iniziale, cioè denotando un peggioramento. Viceversa, un valore positivo in corrispondenza del parametro della frequenza massima di funzionamento indica un miglioramento rispetto al design iniziale non ottimizzato.

Si può notare come la tecnica del registering permette di ottenere una diminuzione di circa 1.55 mW in corrispondenza della potenza totale pur presentando un incremento della Clocks Power rispetto al design iniziale. Ricordando che il registering fa uso di un numero maggiore di FF rispetto all'architettura iniziale, questo giustifica questo aumento di potenza precedentemente citato. Inoltre, rispetto ad altre soluzioni proposte, la tecnica del registering presenta un aumento della frequenza massima di funzionamento leggermente inferiore. Tanto è vero che le soluzioni più complesse come quella ibrida e quella del Clock Gating All presentano un aumento di circa 1.22 MHz mentre le strategie di Clock Gating Selectors e di Clock Gating Inputs presentano rispettivamente un aumento di circa ben 4.82 MHz e 6.41 MHz .

POWER ABSOLUTE ERROR ANALYSIS



RESOURCES UTILIZATION ABSOLUTE ERROR ANALYSIS



Inoltre, analizzando anche graficamente gli errori assoluti ottenuti, si può notare come il numero di FF associati alla soluzione del Clock Gating All risulta essere praticamente la medesima di quella non ottimizzata mentre il numero di LUT associato risulta essere, addirittura, inferiore rispetto all'architettura iniziale. Tali ottimizzazioni, inoltre, sono state ottenute in corrispondenza di una diminuzione della potenza totale pari circa al 5.8%.

Pertanto, si può affermare che la soluzione migliore dal punto di vista del low powering risulta essere la strategia ibrida poiché presenta la minore potenza totale. Considerando, invece, un trade-off tra la potenza totale e l'utilizzazione delle risorse, a parità di massima frequenza di funzionamento, la migliore strategia che può essere adottata in questo contesto è la soluzione Clock Gating All, cioè quella che prevede il clock gating implicito sui segnali A, B, C, D e selettori.