E3: 231 Digital Systems Design with FPGAs

Jangam Abhinav

SR : 24925

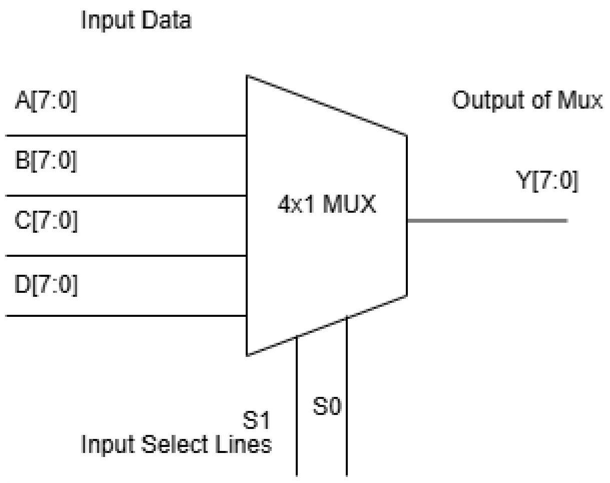
Assignment 1

1. **Write Verilog code to implement an 8-bit, 4-to-1 multiplexer. Do the functional and timing simulation. Target Device is Xilinx Artix-7 XC7A35T- ICPG236C (Family Artix-7, Part XC7A35T, Package CPG236, Speed Grade -1) Submit the source Verilog codes, RTL blocks, Resource utilization, Timing report, and The timing (functional and post-route) simulation waveforms.**

Introduction

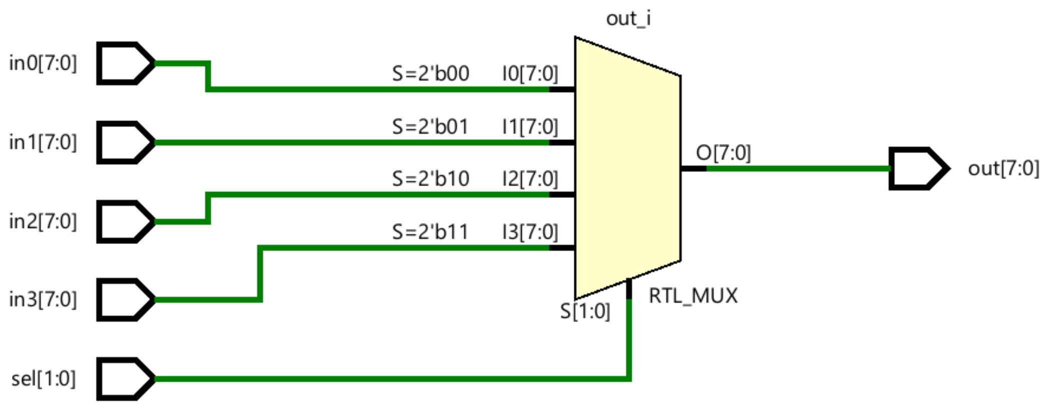
In this task, an **8-bit, 4-to-1 multiplexer (MUX)** is designed and implemented using Verilog. The multiplexer selects one of the four 8-bit inputs based on the 2-bit select lines. The logic is implemented using a case statement, ensuring correct mapping of inputs to outputs. The design is simulated to verify its correctness through functional and timing analysis. The target device for FPGA implementation is the **Xilinx Artix-7 XC7A35T**. Here we are going to design and implement a 8 bit, 4x1 MUX. So we are feeding an input of 8-bit length having Little Endian format. The logic for the MUX is written in Verilog using the 'case' statements and assigning the input according to Select lines accordingly. Here there are 2 select lines as it is a 4x1 mux.

Architecture Design



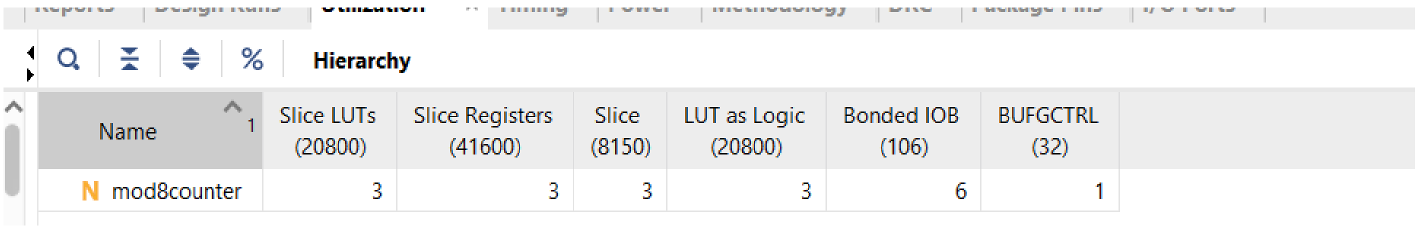
S1	**S0**	**Output(Y)[7:0]**
0	**0**	**A[7:0]**
0	**1**	**B[7:0]**
1	**0**	**C[7:0]**
1	**1**	**D[7:0]**

RTL Block



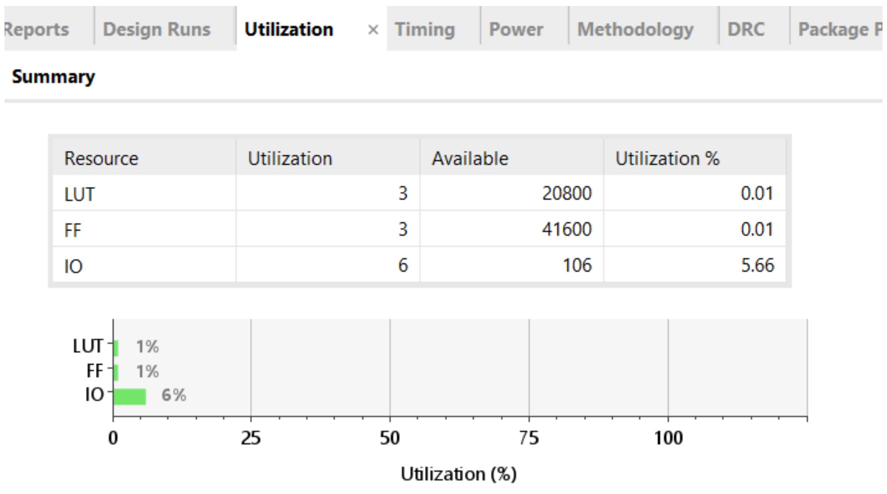
Resource Utilization

Site Type	Utilization	Available	Utilized %
Slice	7	8150	0.08
Slice LUTs	8	20800	0.04
IO Ports	42	106	39.62



Hierarchy

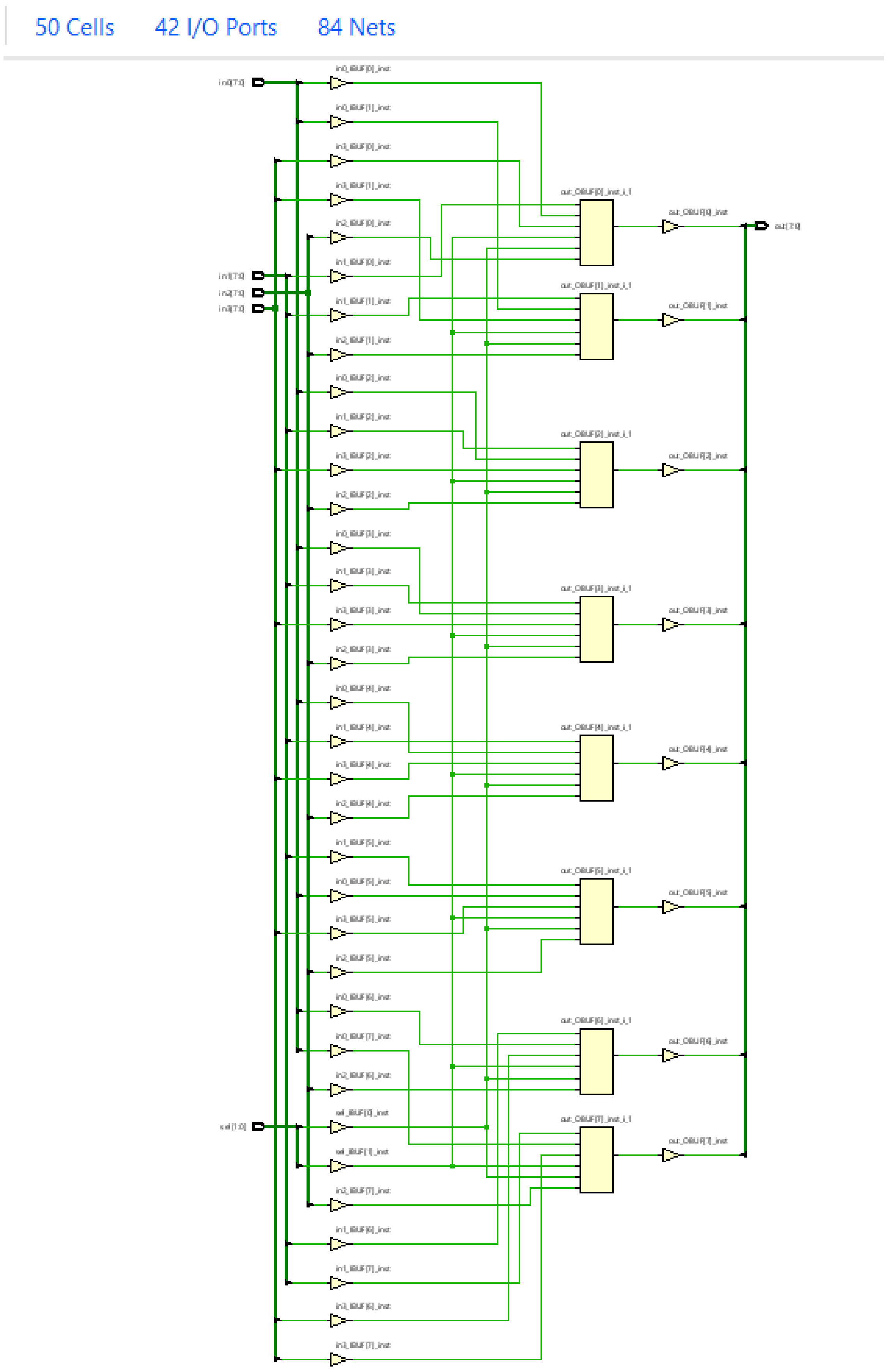
Name	Slice LUTs (20800)	Slice Registers (41600)	Slice (8150)	LUT as Logic (20800)	Bonded IOB (106)	BUFGCTRL (32)
mod8counter	3	3	3	3	6	1



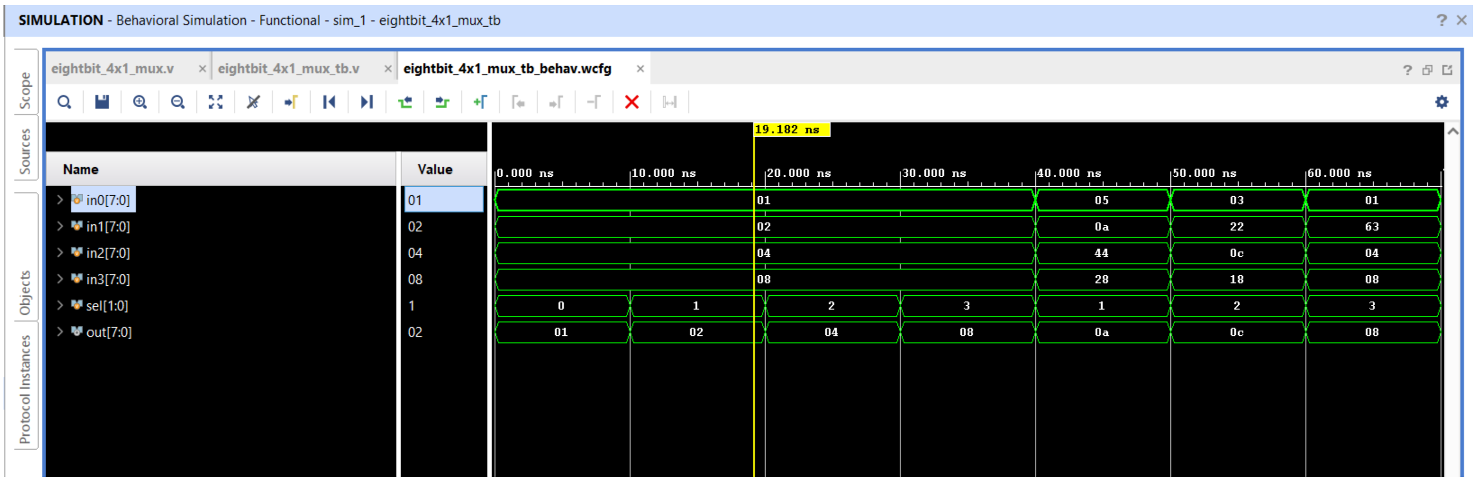
Summary

Resource	Utilization	Available	Utilization %
LUT	3	20800	0.01
FF	3	41600	0.01
IO	6	106	5.66

Schematic after Implementation

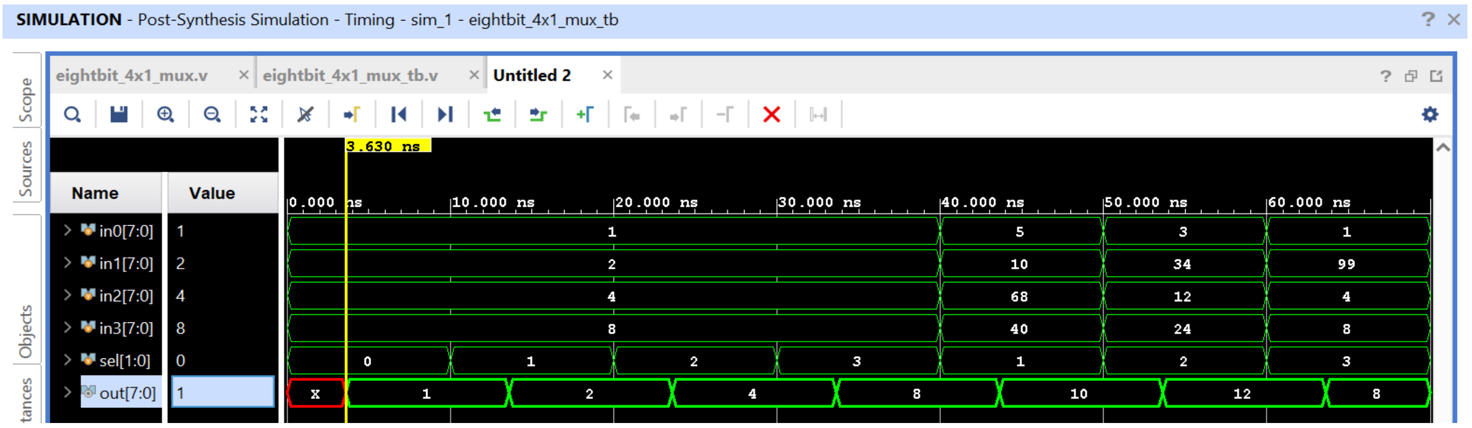


Timing Diagram (Behavioural)



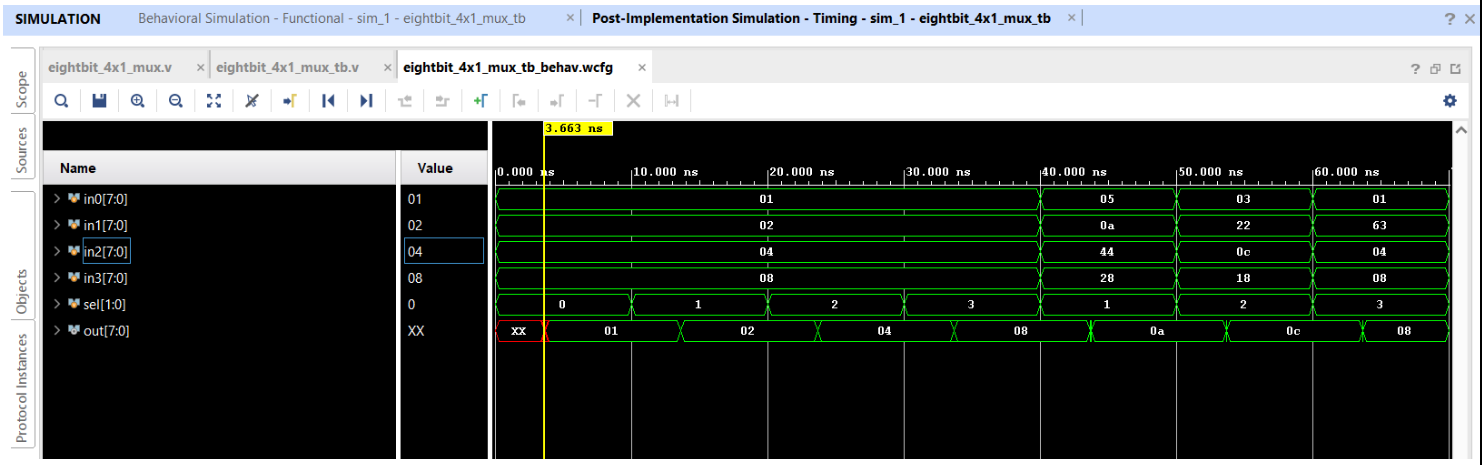
SIMULATION - Behavioral Simulation - Functional - sim_1 - eightbit_4x1_mux_tb
eightbit_4x1_mux.v
eightbit_4x1_mux_tb.v
eightbit_4x1_mux_tb_behav.wcfg
Scope
Sources
Objects
Protocol Instances
19.182 ns
Name
Value
0.000 ns
10.000 ns
20.000 ns
30.000 ns
40.000 ns
50.000 ns
60.000 ns
in0[7:0] 01 | 01 | 05 | 03 | 01
in1[7:0] 02 | 02 | 0a | 22 | 63
in2[7:0] 04 | 04 | 44 | 0c | 04
in3[7:0] 08 | 08 | 28 | 18 | 08
sel[1:0] 1 | 0 | 1 | 2 | 3 | 1 | 2 | 3
out[7:0] 02 | 01 | 02 | 04 | 08 | 0a | 0c | 08

Timing Diagram(Post-Synthesis)



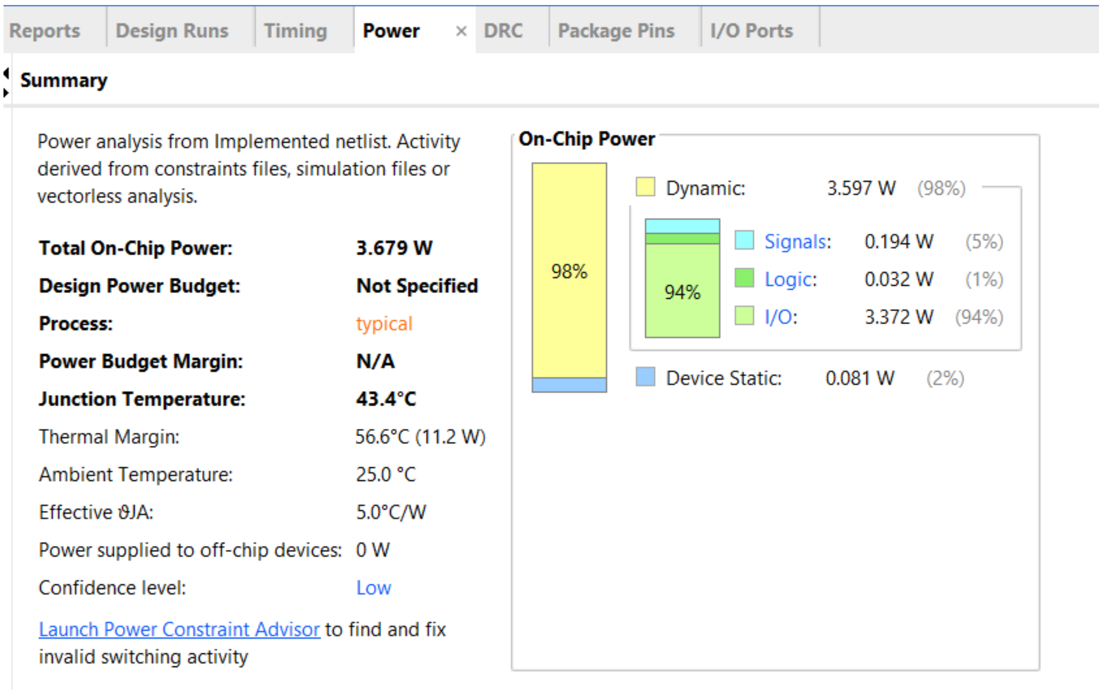
SIMULATION - Post-Synthesis Simulation - Timing - sim_1 - eightbit_4x1_mux_tb
eightbit_4x1_mux.v
eightbit_4x1_mux_tb.v
Untitled 2
Scope
Sources
Objects
tances
3.630 ns
Name
Value
0.000 ns
10.000 ns
20.000 ns
30.000 ns
40.000 ns
50.000 ns
60.000 ns
in0[7:0] 1 | 1 | 5 | 3 | 1
in1[7:0] 2 | 2 | 10 | 34 | 99
in2[7:0] 4 | 4 | 68 | 12 | 4
in3[7:0] 8 | 8 | 40 | 24 | 8
sel[1:0] 0 | 0 | 1 | 2 | 3 | 1 | 2 | 3
out[7:0] 1 | X | 1 | 2 | 4 | 8 | 10 | 12 | 8

Timing Diagram (Post Implementation)



SIMULATION
Behavioral Simulation - Functional - sim_1 - eightbit_4x1_mux_tb
Post-Implementation Simulation - Timing - sim_1 - eightbit_4x1_mux_tb
eightbit_4x1_mux.v
eightbit_4x1_mux_tb.v
eightbit_4x1_mux_tb_behav.wcfg
Scope
Sources
Objects
Protocol Instances
3.663 ns
Name
Value
0.000 ns
10.000 ns
20.000 ns
30.000 ns
40.000 ns
50.000 ns
60.000 ns
in0[7:0] 01 | 01 | 05 | 03 | 01
in1[7:0] 02 | 02 | 0a | 22 | 63
in2[7:0] 04 | 04 | 44 | 0c | 04
in3[7:0] 08 | 08 | 28 | 18 | 08
sel[1:0] 0 | 0 | 1 | 2 | 3 | 1 | 2 | 3
out[7:0] XX | XX | 01 | 02 | 04 | 08 | 0a | 0c | 08

Power Utilization



Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

Total On-Chip Power:	**3.679 W**
Design Power Budget:	**Not Specified**
Process:	typical
Power Budget Margin:	**N/A**
Junction Temperature:	**43.4°C**
Thermal Margin:	56.6°C (11.2 W)
Ambient Temperature:	25.0 °C
Effective ϑJA:	5.0°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Launch Power Constraint Advisor to find and fix invalid switching activity

On-Chip Power

🟨	Dynamic:	3.597 W	(98%)
🟦	Signals:	0.194 W	(5%)
🟩	Logic:	0.032 W	(1%)
🟩	I/O:	3.372 W	(94%)
🟦	Device Static:	0.081 W	(2%)

Conclusion

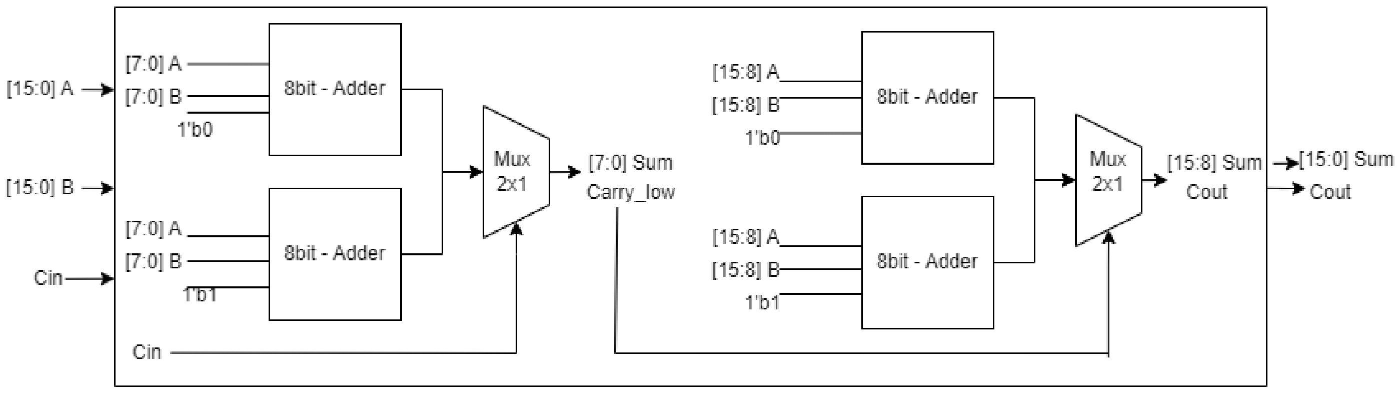
The functional simulation confirmed the correct operation of the 4-to-1 multiplexer. From the post-implementation timing report, a delay of **3.663 ns** was observed, which is attributed to routing and interconnect delays in the FPGA. The post-synthesis waveform also aligns with the expected behaviour. The power consumption was measured to be **3.7W**, demonstrating efficient resource utilization.

2.Write Verilog model with continuous assignment for a 16-bit carry select adder. Do the functional and timing simulation. Target Device is Xilinx Artix-7 XC7A35T- ICPG236C (Family Artix-7, Part XC7A35T, Package CPG236, Speed Grade -1).

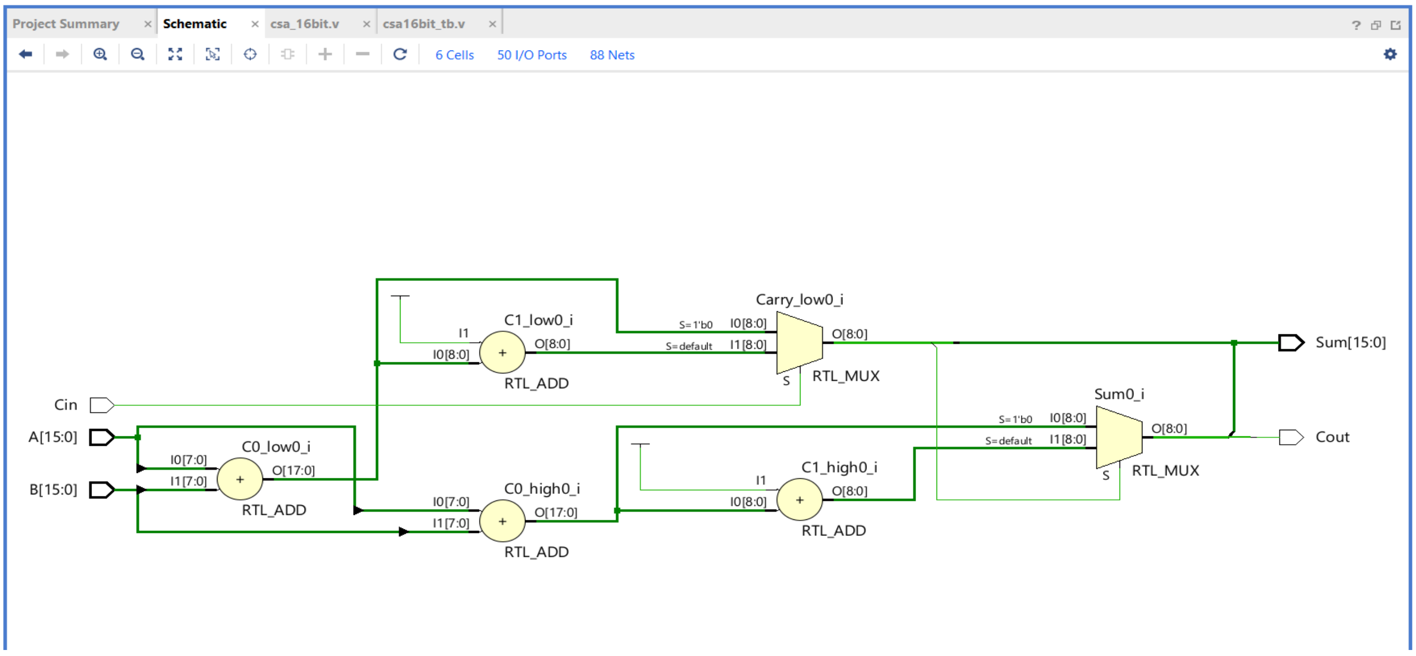
Introduction

A 16-bit carry select adder is implemented using two 8-bit adders. The carry output from the lower 8-bit adder propagates into the higher 8-bit adder. This structure is chosen to optimize addition performance compared to a traditional ripple-carry adder. The design is verified through functional and timing simulation using continuous assignment in Verilog.

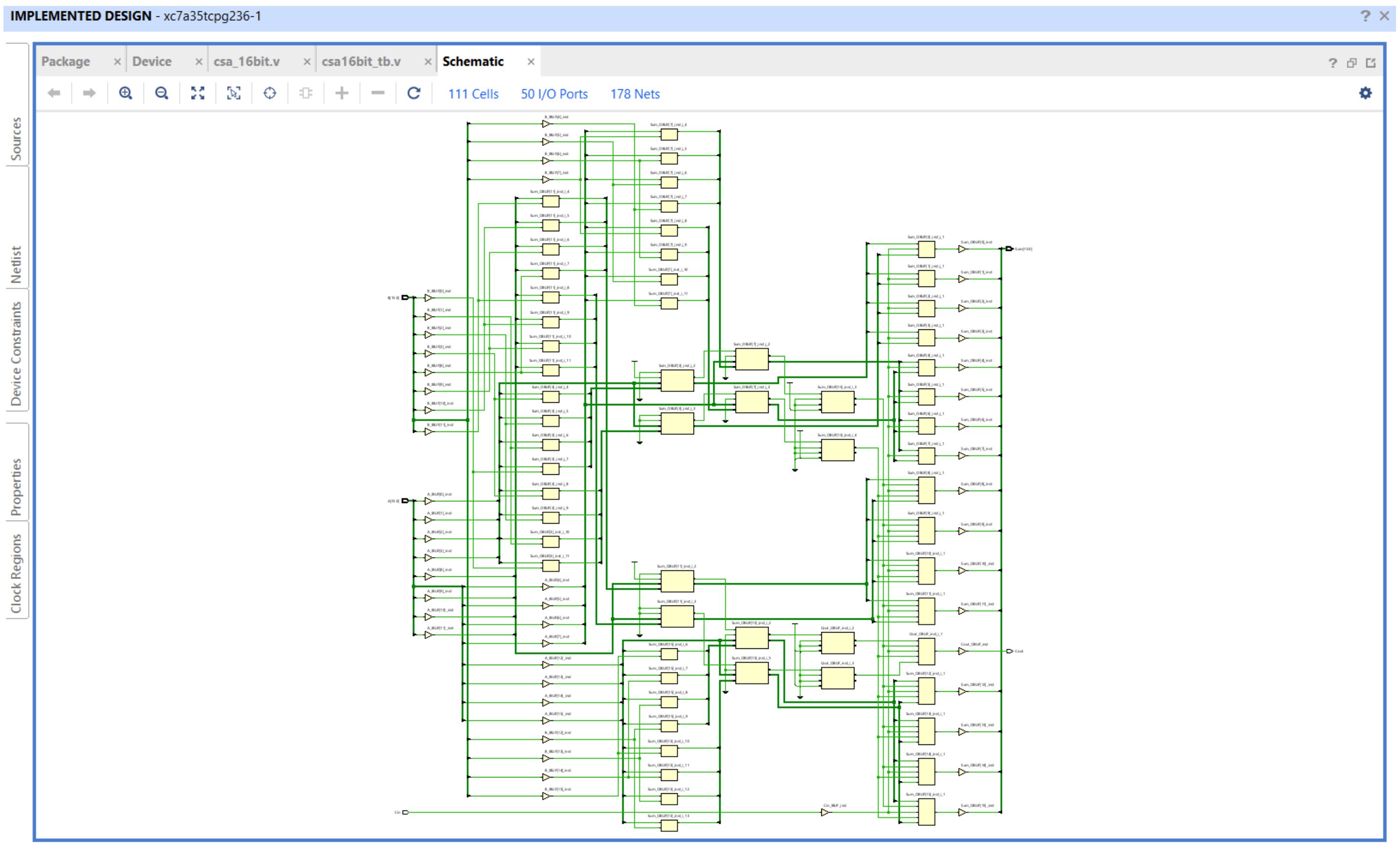
Architectural Design



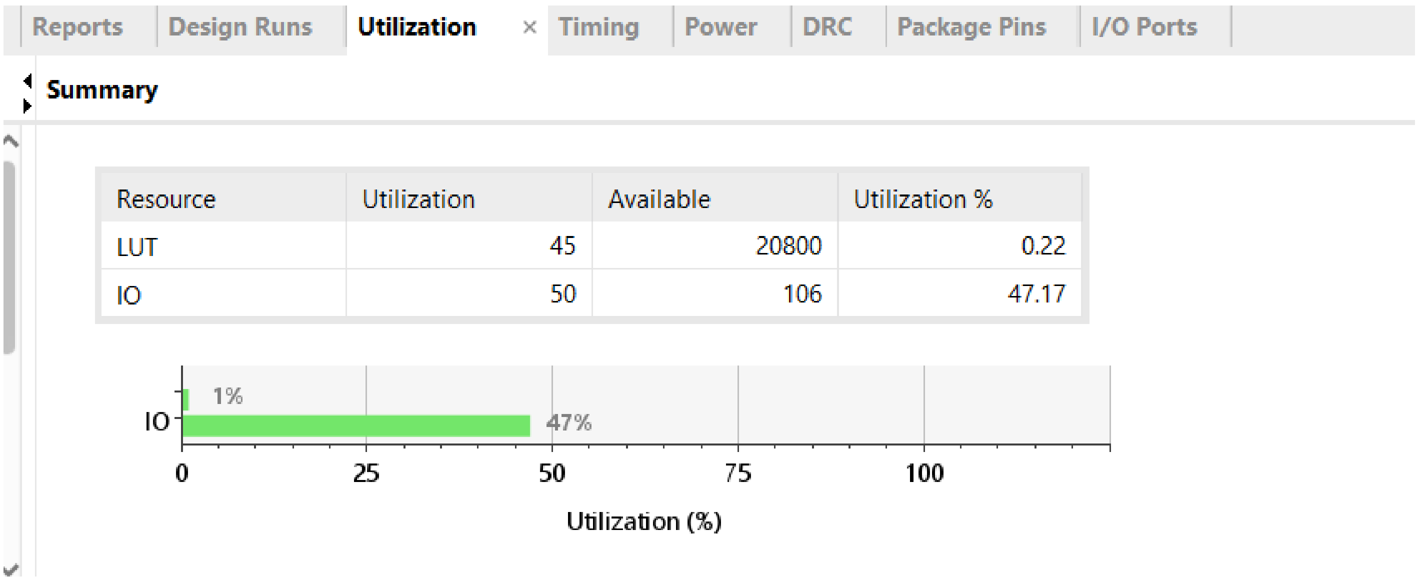
RTL Blocks



Schematic after implementation



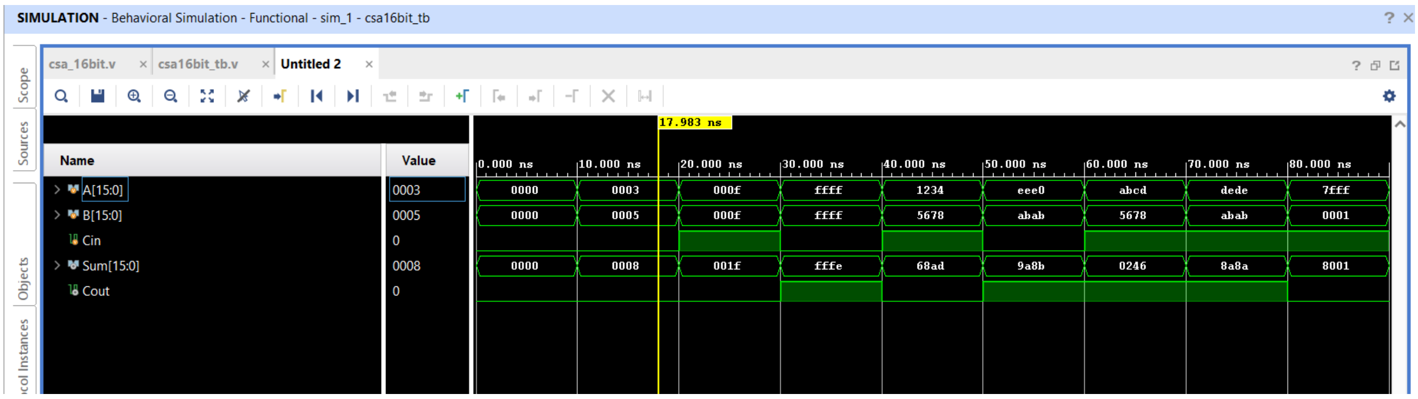
Resource Utilization



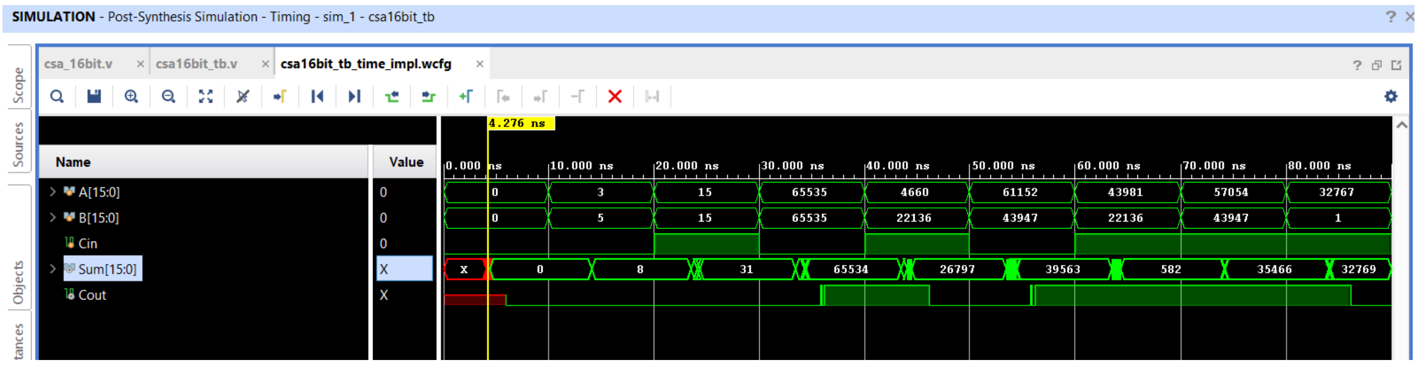
Resource	Utilization	Available	Utilization %
LUT	45	20800	0.22
IO	50	106	47.17

Site Type	Utilization	Available	Utilized %
Slice	16	8150	0.20
Slice LUTs	45	20800	0.22
IO Ports	50	106	47.17

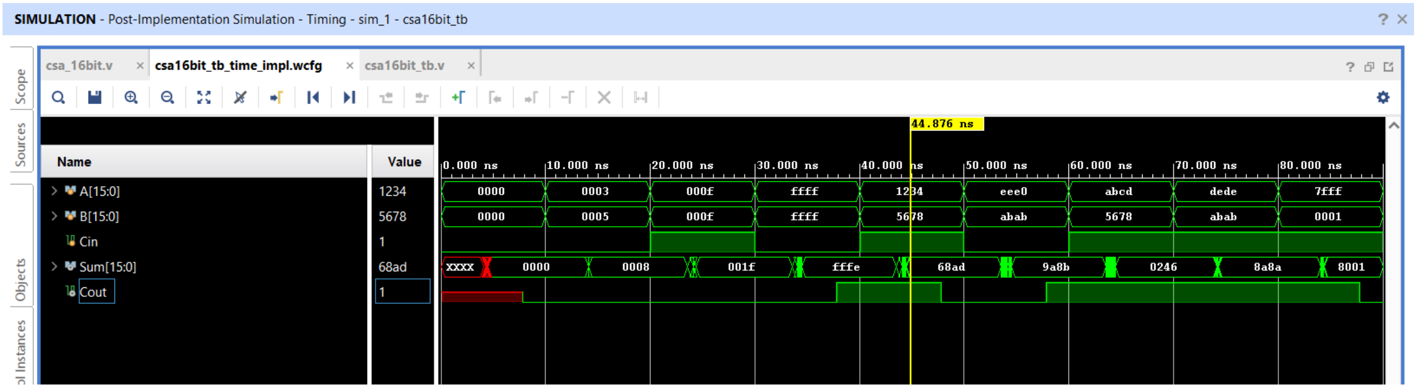
Timing Diagram(Behavioural)


SIMULATION - Behavioral Simulation - Functional - sim_1 - csa16bit_tb
csa_16bit.v
csa16bit_tb.v
Untitled 2
17.983 ns
Name
Value
A[15:0] 0003
B[15:0] 0005
Cin 0
Sum[15:0] 0008
Cout 0
0.000 ns 10.000 ns 20.000 ns 30.000 ns 40.000 ns 50.000 ns 60.000 ns 70.000 ns 80.000 ns
0000 0003 000f ffff 1234 eee0 abcd dede 7fff
0000 0005 000f ffff 5678 abab 5678 abab 0001
0000 0008 001f fffe 68ad 9a8b 0246 8a8a 8001

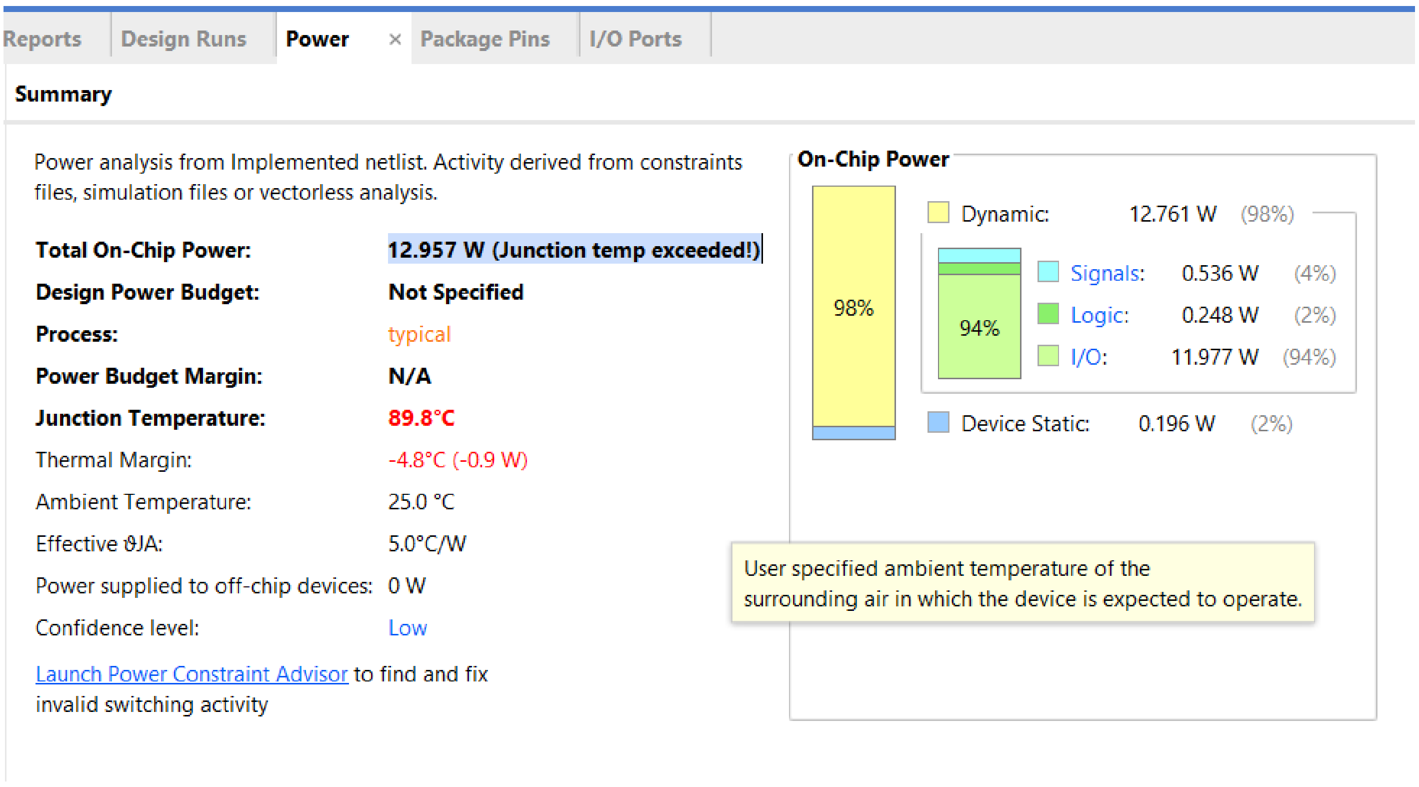
Timing Diagram(Post-Synthesis)


SIMULATION - Post-Synthesis Simulation - Timing - sim_1 - csa16bit_tb
csa_16bit.v
csa16bit_tb.v
csa16bit_tb_time_impl.wcfg
4.276 ns
Name
Value
A[15:0] 0
B[15:0] 0
Cin 0
Sum[15:0] X
Cout X
0.000 ns 10.000 ns 20.000 ns 30.000 ns 40.000 ns 50.000 ns 60.000 ns 70.000 ns 80.000 ns
0 3 15 65535 4660 61152 43981 57054 32767
0 5 15 65535 22136 43947 22136 43947 1
X 0 8 31 65534 26797 39563 582 35466 32769

Timing Diagram(Post-Implementation)


SIMULATION - Post-Implementation Simulation - Timing - sim_1 - csa16bit_tb
csa_16bit.v
csa16bit_tb_time_impl.wcfg
csa16bit_tb.v
44.876 ns
Name
Value
A[15:0] 1234
B[15:0] 5678
Cin 1
Sum[15:0] 68ad
Cout 1
0.000 ns 10.000 ns 20.000 ns 30.000 ns 40.000 ns 50.000 ns 60.000 ns 70.000 ns 80.000 ns
0000 0003 000f ffff 1234 eee0 abcd dede 7fff
0000 0005 000f ffff 5678 abab 5678 abab 0001
XXXX 0000 0008 001f fffe 68ad 9a8b 0246 8a8a 8001

Power Consumption(Post-Implementation)



Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

Total On-Chip Power:	**12.957 W (Junction temp exceeded!)**
Design Power Budget:	**Not Specified**
Process:	typical
Power Budget Margin:	**N/A**
Junction Temperature:	**89.8°C**
Thermal Margin:	-4.8°C (-0.9 W)
Ambient Temperature:	25.0 °C
Effective ϑJA:	5.0°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Launch Power Constraint Advisor to find and fix invalid switching activity

On-Chip Power

🟨	Dynamic:	12.761 W	(98%)
🟦	Signals:	0.536 W	(4%)
🟩	Logic:	0.248 W	(2%)
🟩	I/O:	11.977 W	(94%)
🟦	Device Static:	0.196 W	(2%)

User specified ambient temperature of the surrounding air in which the device is expected to operate.

Conclusion

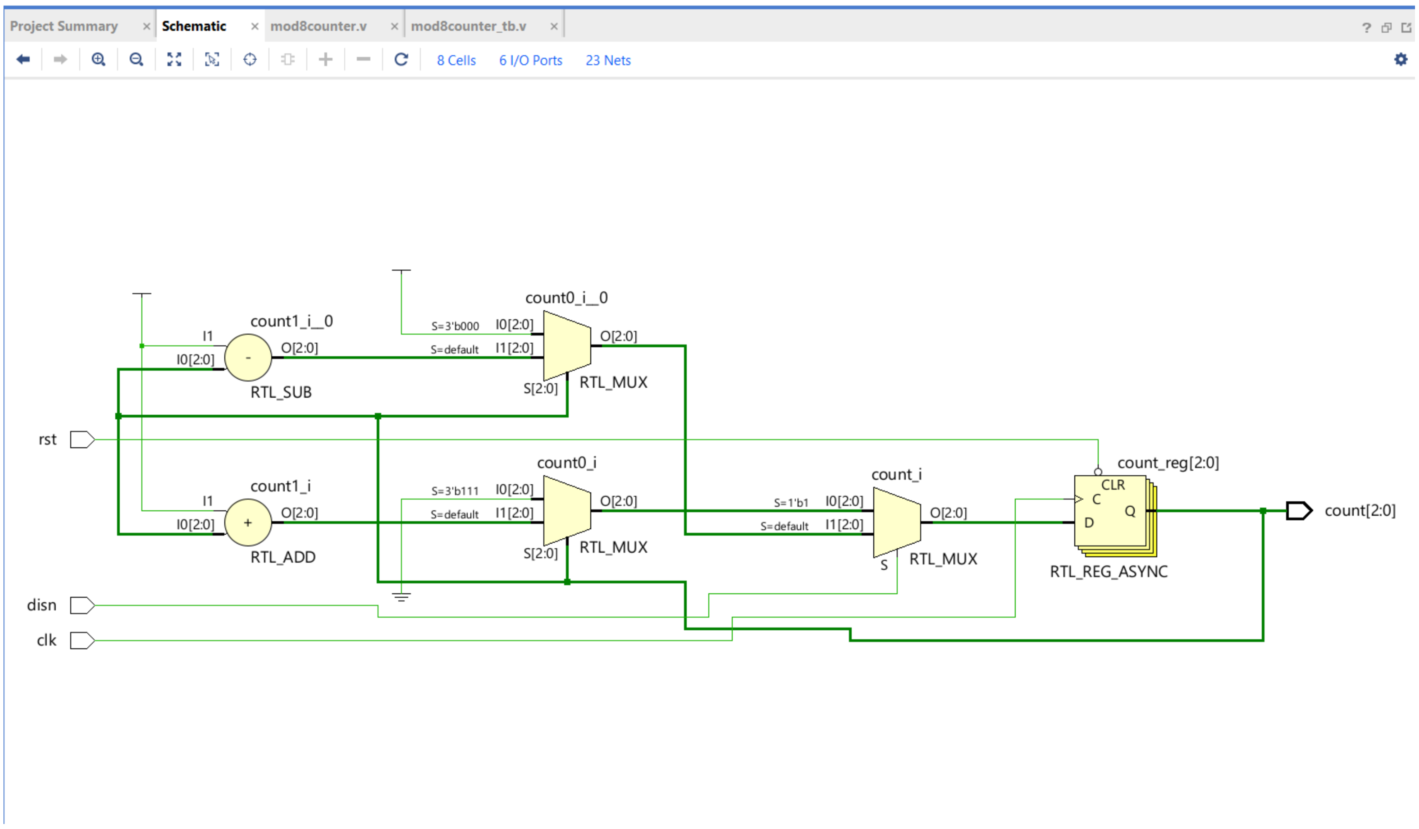
The simulation results validated the expected functionality of the carry-select adder. Timing analysis revealed **positive slack**, meaning all timing constraints were met. However, post-implementation analysis showed a **delay due to interconnects and LUT propagation**. The observed power consumption was **13W**, which is higher than the previous design but acceptable given the complexity of the operation.

3.A] Implementing a simple Mod-8 Up-Down counter with Synchronous Reset .

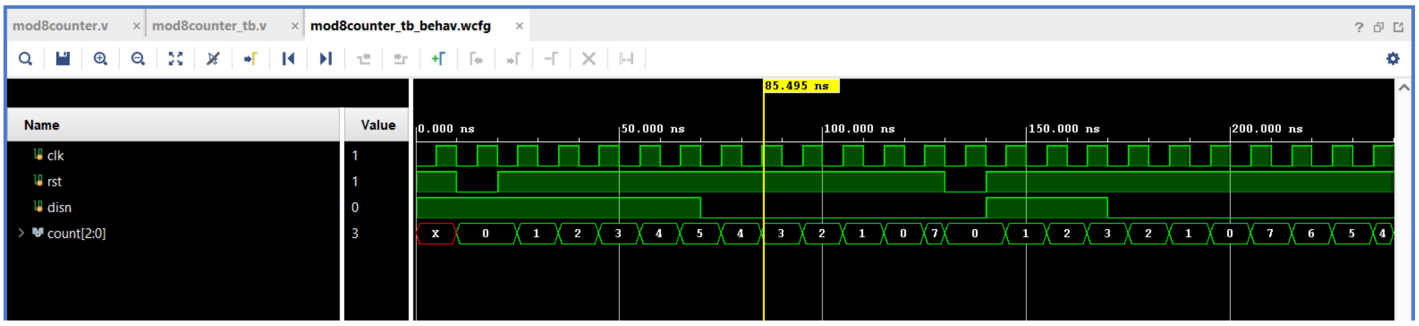
Introduction

A **Mod-8 counter** is implemented using Verilog with a synchronous active-low reset. The counter increments from 0 to 7 and then resets or decrements back down, depending on the mode. The implementation uses an if-else structure to control the count direction. Functional and timing simulations are performed to ensure correct operation.Here we have implemented a Mod-8 counter with synchronous reset . It has been implemented using a simple if else statement until it reaches 7 or 0 (since it is Mod-8) , it is assigned to down count or up count accordingly . Since we need to implement a Synchronous active low Reset , the count will be initialised to 0 once the reset is made 0. The exact function has been implemented in Verilog using the ‘if-else’ statements.

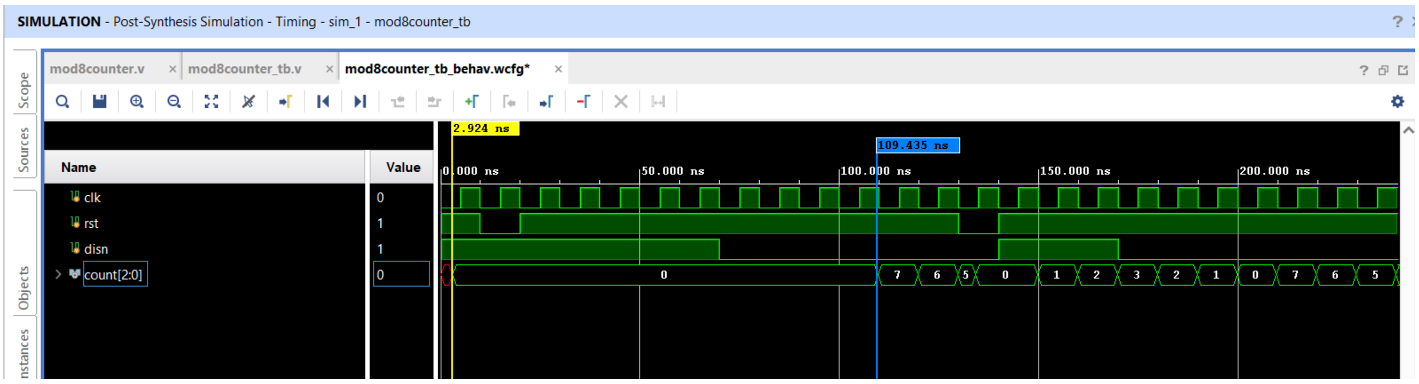
RTL Block



Timing(Behavioural Simulation)



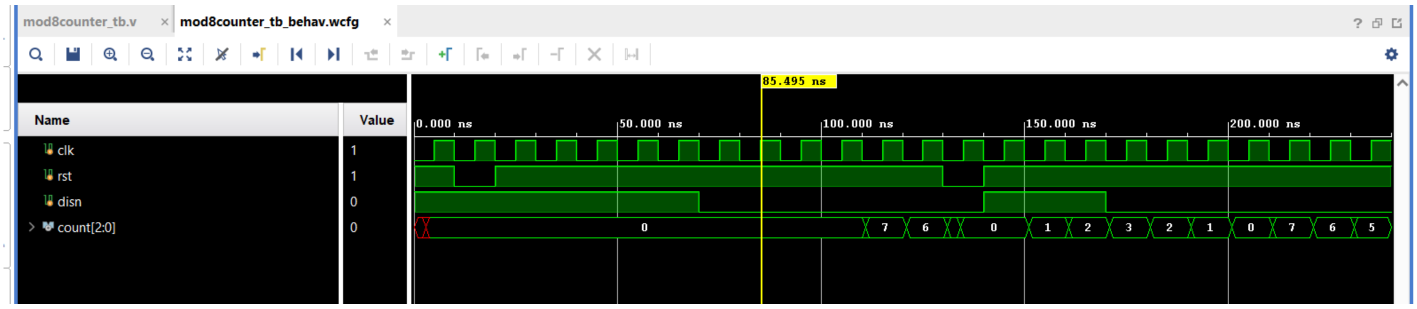
Timing (Post-Synthesis)


SIMULATION - Post-Synthesis Simulation - Timing - sim_1 - mod8counter_tb

mod8counter.v | mod8counter_tb.v | **mod8counter_tb_behav.wcfg***

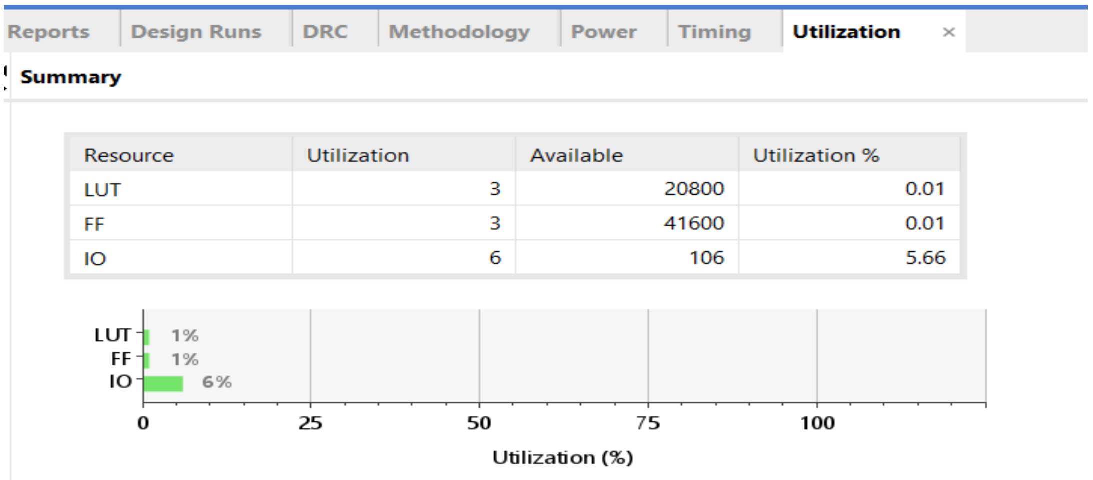
Name	Value
clk	0
rst	1
disn	1
count[2:0]	0

Timing (Post-Implementation)


mod8counter_tb.v | **mod8counter_tb_behav.wcfg**

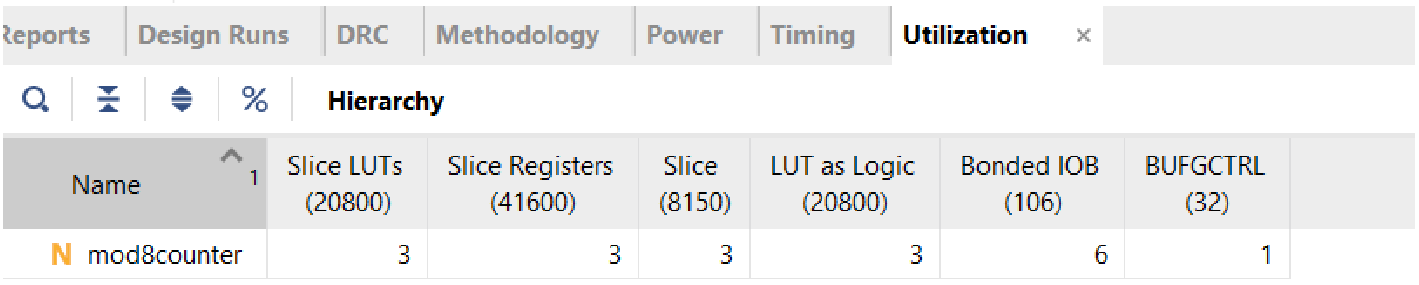
Name	Value
clk	1
rst	1
disn	0
count[2:0]	0

Utilization report


Reports | Design Runs | DRC | Methodology | Power | Timing | **Utilization**

Summary

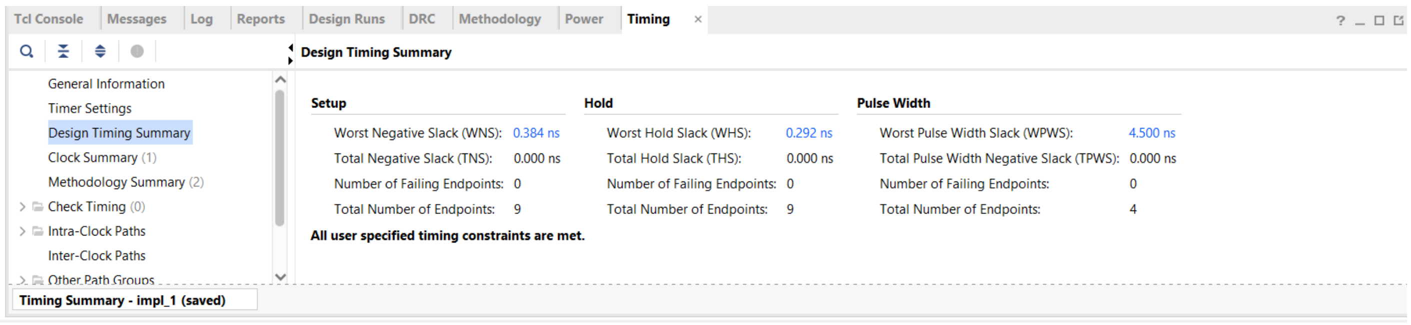
Resource	Utilization	Available	Utilization %
LUT	3	20800	0.01
FF	3	41600	0.01
IO	6	106	5.66


Reports | Design Runs | DRC | Methodology | Power | Timing | **Utilization**

Hierarchy

Name 1	Slice LUTs (20800)	Slice Registers (41600)	Slice (8150)	LUT as Logic (20800)	Bonded IOB (106)	BUFGCTRL (32)
N mod8counter	3	3	3	3	6	1

Timing Report



General Information
Timer Settings
Design Timing Summary
Clock Summary (1)
Methodology Summary (2)
Check Timing (0)
Intra-Clock Paths
Inter-Clock Paths
Other Path Groups

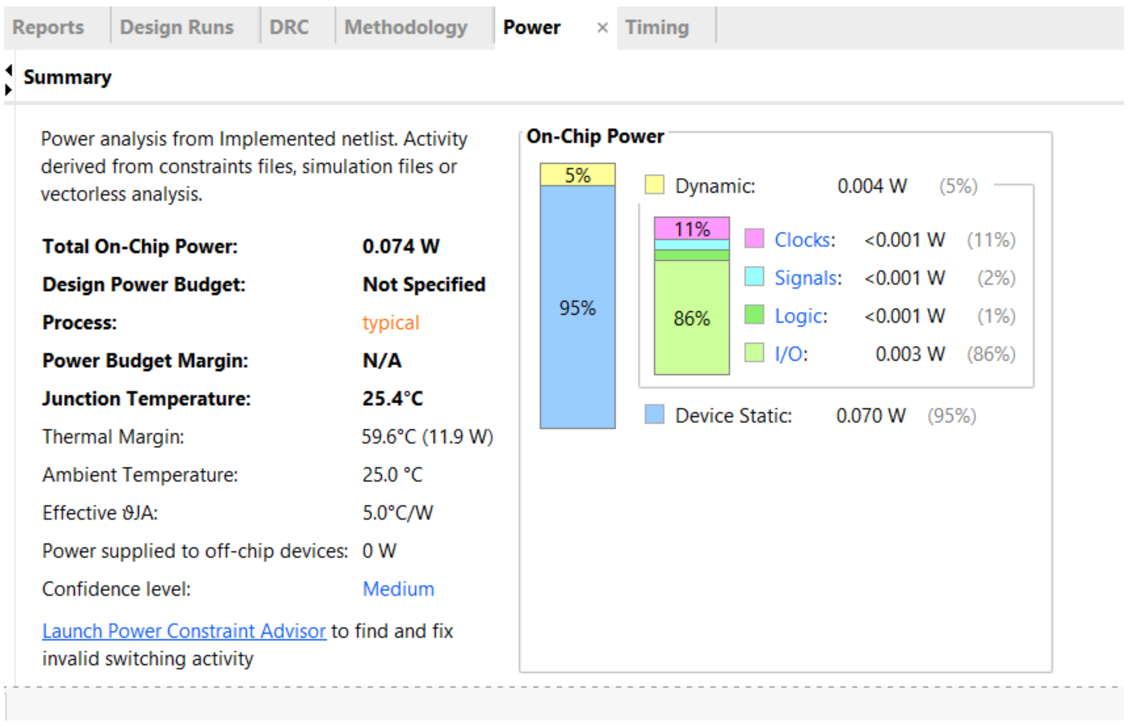
Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	0.384 ns	Worst Hold Slack (WHS):	0.292 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	9	Total Number of Endpoints:	9	Total Number of Endpoints:	4

All user specified timing constraints are met.

Timing Summary - impl_1 (saved)

Power Summary



Summary

Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

Total On-Chip Power:	**0.074 W**
Design Power Budget:	**Not Specified**
Process:	typical
Power Budget Margin:	**N/A**
Junction Temperature:	**25.4°C**
Thermal Margin:	59.6°C (11.9 W)
Ambient Temperature:	25.0 °C
Effective ϑJA:	5.0°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Medium

Launch Power Constraint Advisor to find and fix invalid switching activity

On-Chip Power

Dynamic:	0.004 W	(5%)
Clocks:	<0.001 W	(11%)
Signals:	<0.001 W	(2%)
Logic:	<0.001 W	(1%)
I/O:	0.003 W	(86%)
Device Static:	0.070 W	(95%)

Conclusion

The simulation waveforms confirmed that the counter functioned correctly. The timing report indicated **positive slack**, meaning no violations in timing constraints. The power consumption of the implementation was measured at **0.074W**, demonstrating a low-power design. The functional simulation waveform obtained is as expected. From the timing report, the observed slack is positive and all specified timing constraints are met. A power consumption of 0.074 W is observed.

3B a) Design a synchronous mod-8 up-down counter (with synchronous active low reset) using Verilog. Simulate and test with a testbench. b) Add the following functionalities as discussed in the class: a. Hold previous value b. Count up c. Count down d. Count up-down (first up, then down) e. Load an input value f. Local synchronous reset

Now implementing these extra functions in the counter.

a. Hold previous value

b. Count up

c. Count down

d. Count up-down (first up, then down)

e. Load an input value

f. Local synchronous reset

Introduction

A **procedural block with case statements** was used to implement these features, ensuring efficient mode selection. Here in order to implement the additional functionalities in the MOD8 counter designed above, a procedural block was used along with case statements to choose the modes . The modes were defined accordingly

Functionality	***Mode value assigned[2:0]***
Hold Previous Value	000
Count up	001
Count down	010
Count up-down(first up ,then down)	011
Load an input value	100
Local Synchronous reset	101

The functionality was verified using the timing waveform. There was a definite delay observed in the post-synthesis and even more in the subsequent post-implementation simulation timing waveform generated. In the implementation of the above functionalities using the 'case' statement for various modes, the 'default value' was put as count which in turn represents that if any other undefined modes (say 111) were selected it would still remember the previous count and proceed to wait until any other valid mode is selected and proceeds accordingly. Also in the case of 'Load an Input value" (100) mode, the load is typically given as inputs through switches in FPGA implementation but we have given a default of 4 bits for assigning the load value there is a good chance that the assigned load value is very much greater than so we have assigned

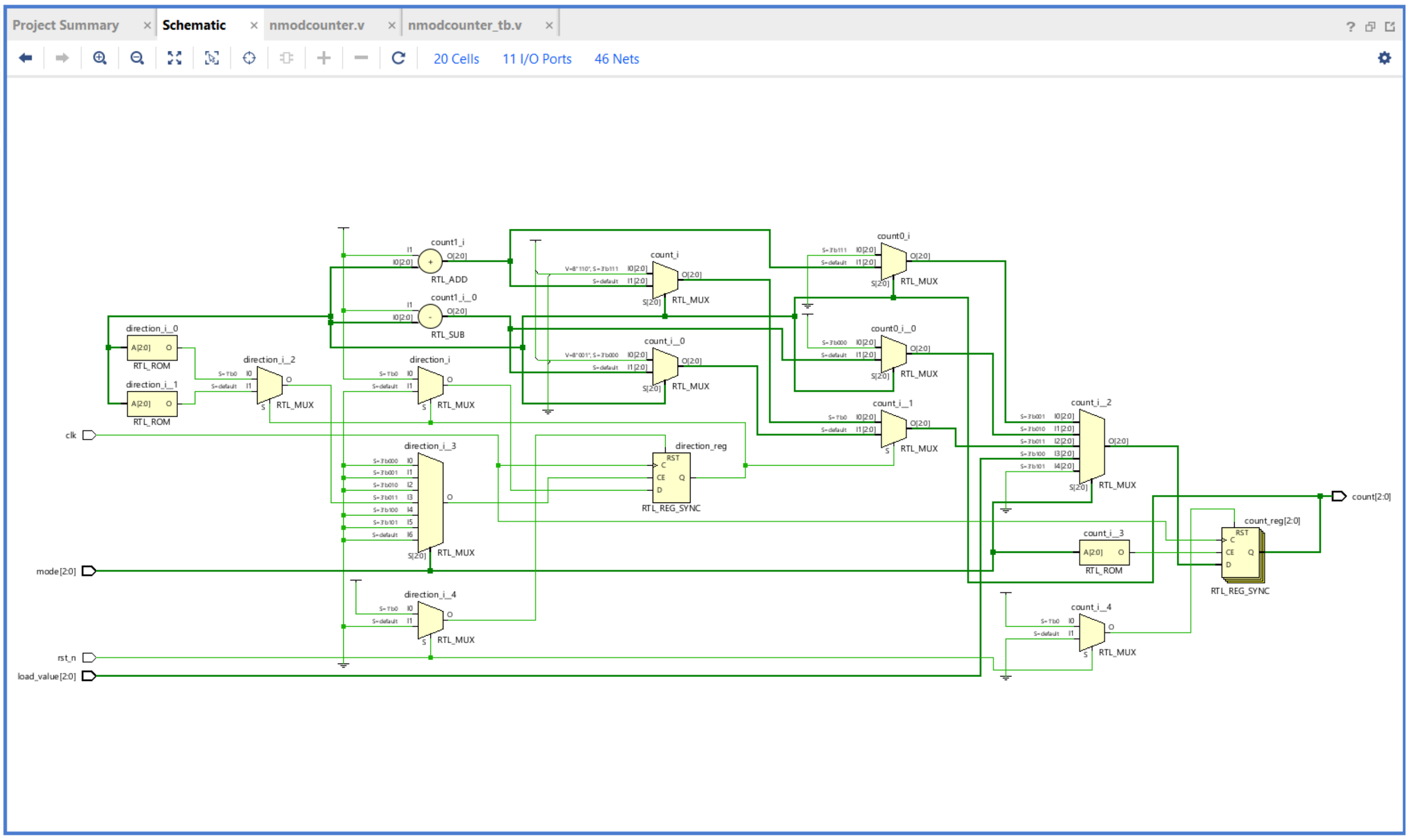
" count = Load value % 8 ",

so that if any higher value than 8 (since here it is mod 8 counter) here it would take the Mod of that value and proceed accordingly. So that we can prevent the excessive load values say(1111,1100 etc) . The similar ' case ' statements were used to execute the functionality of Up counter, this is done by using an iternary operator such that if

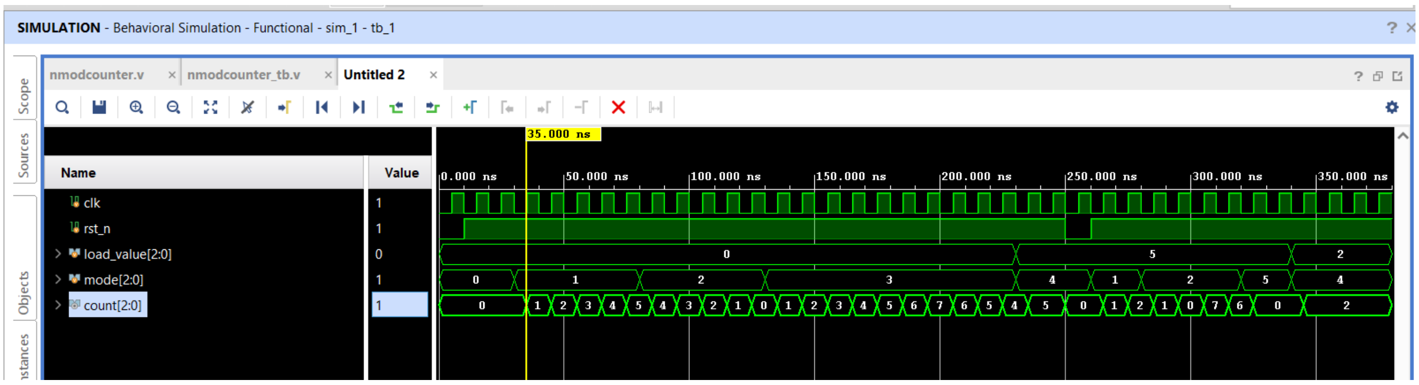
'count = 3'b111',

then it will initialize to 0 , else it will just add 1 to the count i.e it would count up. Similar for count down and , the same logic is also given to Up-down counter.

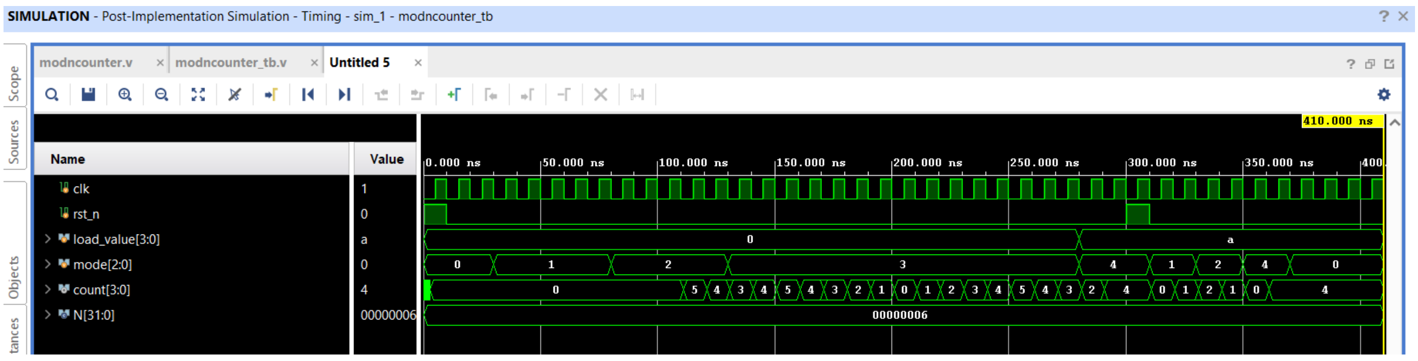
RTL Block



Timing Report(Behavioural)



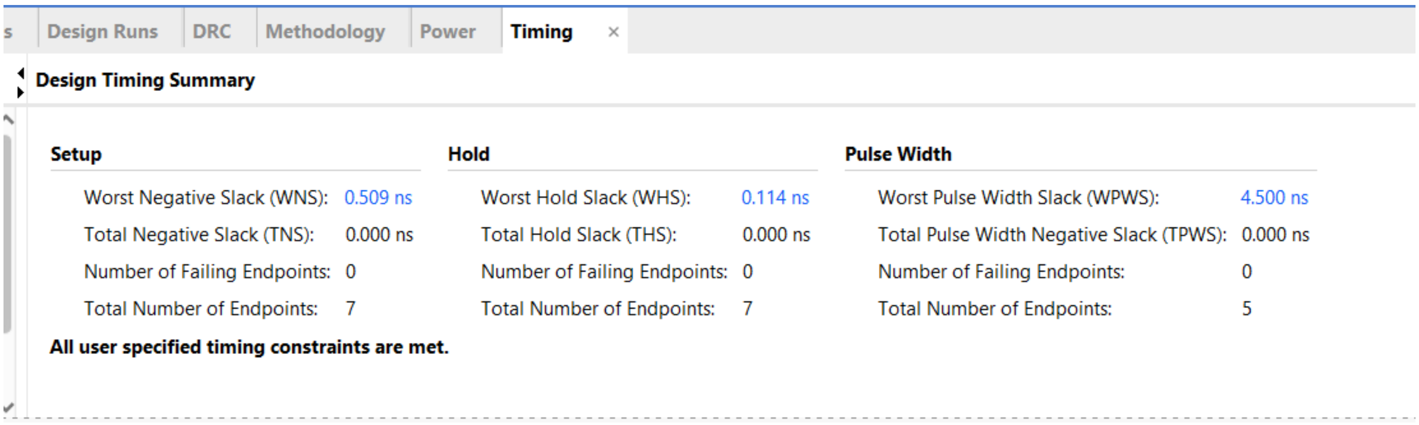
Timing (Post Implementation)



SIMULATION - Post-Implementation Simulation - Timing - sim_1 - modncounter_tb

Name	Value
clk	1
rst_n	0
load_value[3:0]	a
mode[2:0]	0
count[3:0]	4
N[31:0]	00000006

Timing Report

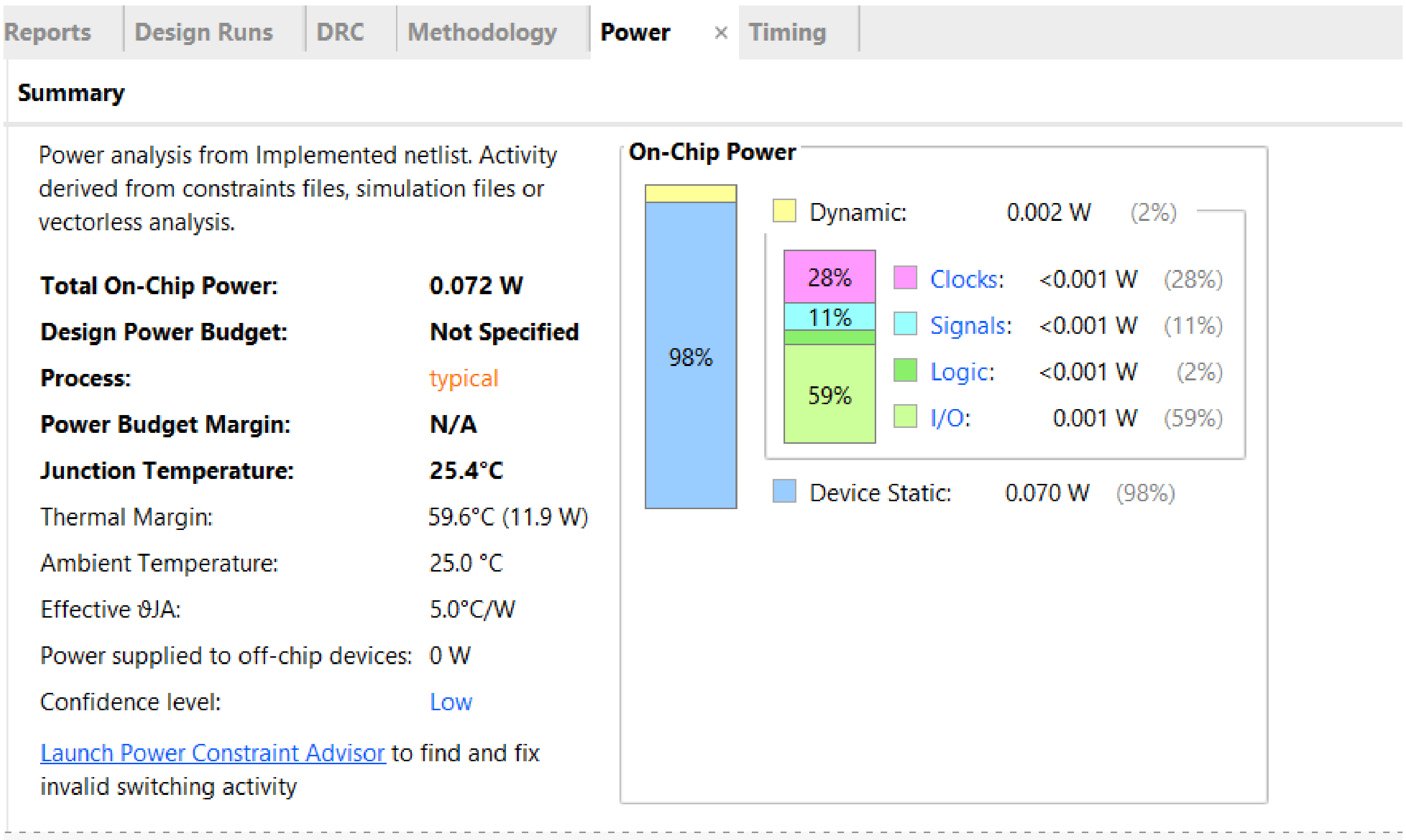


Design Timing Summary

Setup		**Hold**		**Pulse Width**	
Worst Negative Slack (WNS):	0.509 ns	Worst Hold Slack (WHS):	0.114 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	7	Total Number of Endpoints:	7	Total Number of Endpoints:	5

All user specified timing constraints are met.

Power Report



Summary

Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

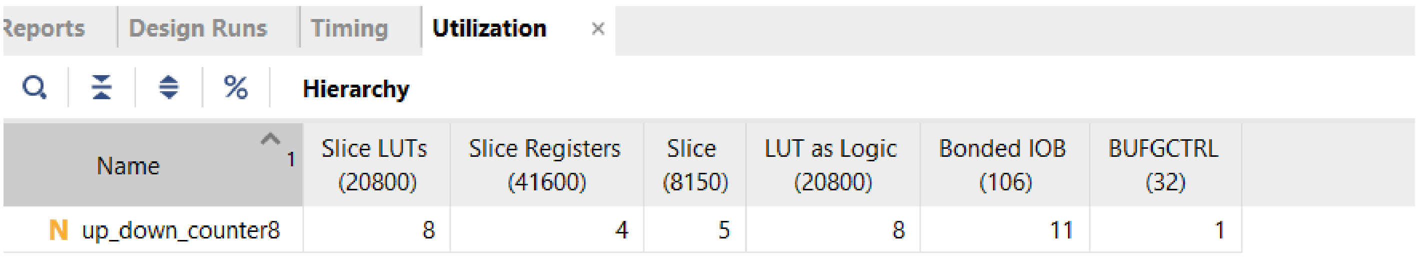
Total On-Chip Power:	**0.072 W**
Design Power Budget:	**Not Specified**
Process:	typical
Power Budget Margin:	**N/A**
Junction Temperature:	**25.4°C**
Thermal Margin:	59.6°C (11.9 W)
Ambient Temperature:	25.0 °C
Effective ϑJA:	5.0°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Launch Power Constraint Advisor to find and fix invalid switching activity

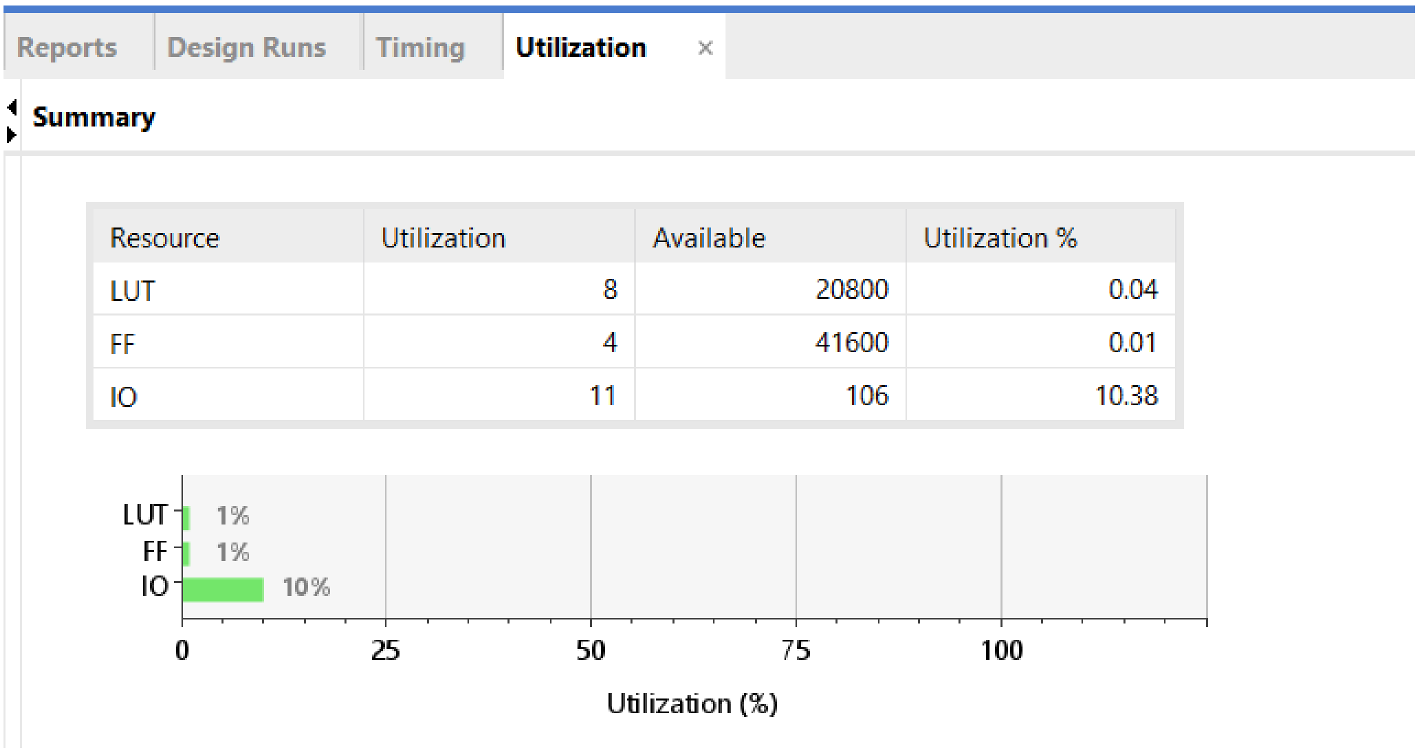
On-Chip Power

Dynamic:	0.002 W	(2%)
Clocks:	<0.001 W	(28%)
Signals:	<0.001 W	(11%)
Logic:	<0.001 W	(2%)
I/O:	0.001 W	(59%)
Device Static:	0.070 W	(98%)

Resource Utilization



Name	Slice LUTs (20800)	Slice Registers (41600)	Slice (8150)	LUT as Logic (20800)	Bonded IOB (106)	BUFGCTRL (32)
N up_down_counter8	8	4	5	8	11	1



Resource	Utilization	Available	Utilization %
LUT	8	20800	0.04
FF	4	41600	0.01
IO	11	106	10.38

Conclusion

The implemented functionalities were successfully verified through **functional and post-implementation simulations**. Timing analysis showed **increased delay** due to additional logic, but the design remained within acceptable limits. The **default case** ensured that the counter held its last state if an undefined mode was selected. The **load function** was modified to prevent loading values exceeding the Mod-8 range.

3C. a) Design a synchronous mod-8 up-down counter (with synchronous active low reset) using Verilog. Simulate and test with a testbench.

b) Add the following functionalities as discussed in the class:

a. Hold previous value

b. Count up

c. Count down

d. Count up-down (first up, then down)

e. Load an input value

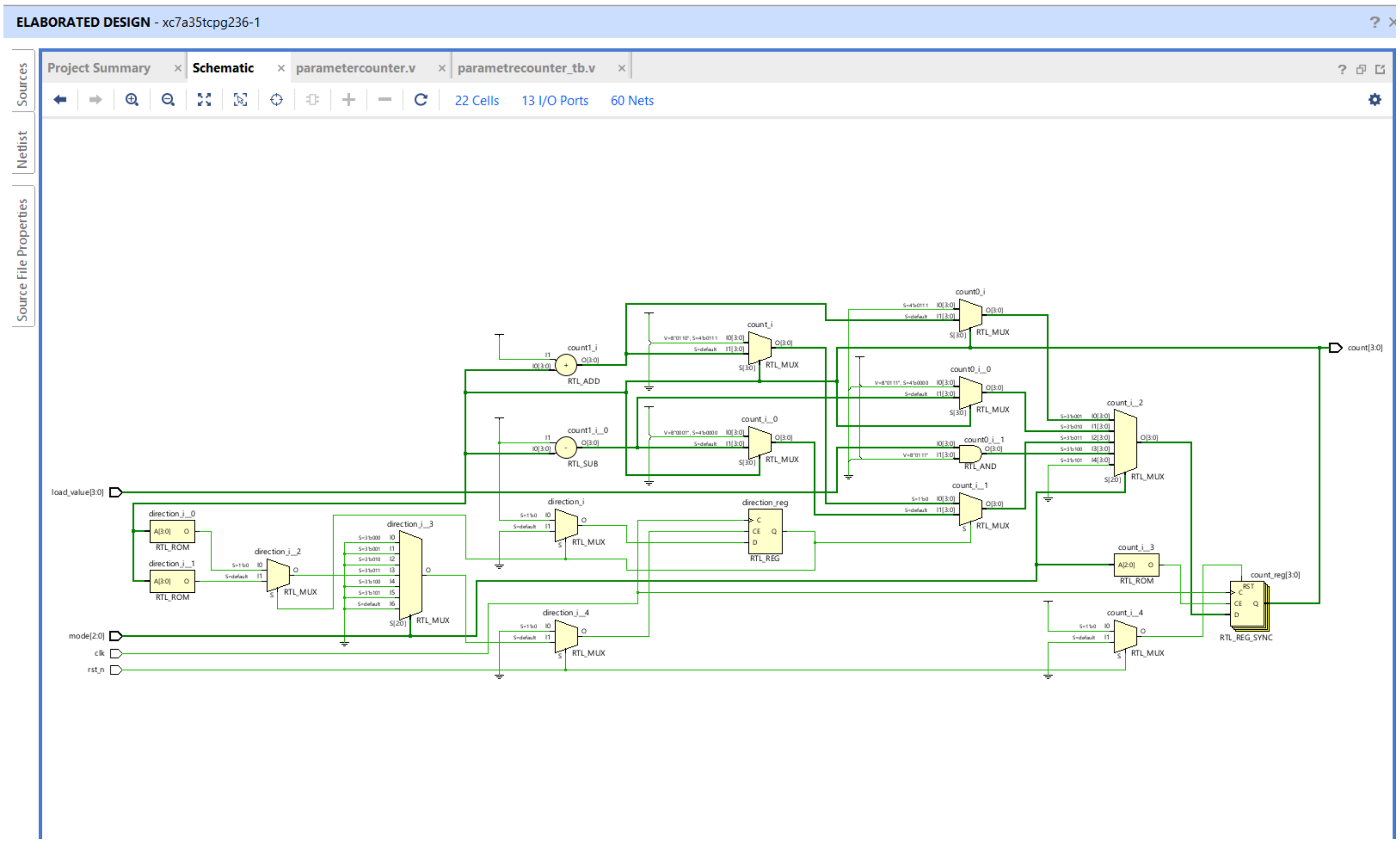
f. Local synchronous reset

c)Parameterize and make the counter behave as a mod-N up-down counter (N ≤ 16).

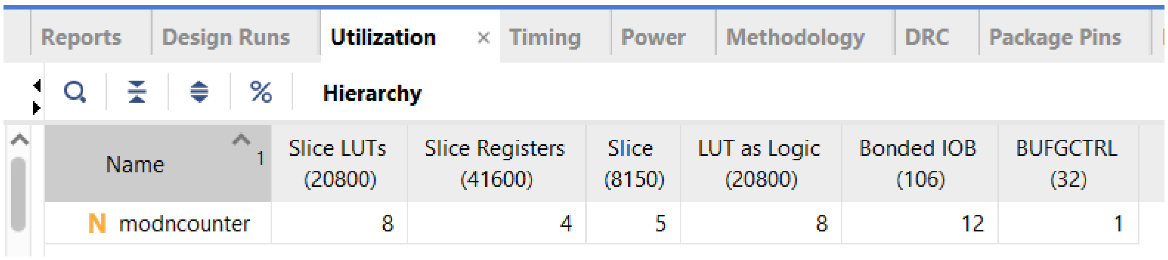
Introduction

Now we will have to parameterize the code so that it can count up to the N value provided N is less than or equal to 16. So, it must be Mod 16 counter or less. Now by modifying the highest bit i.e. (111) in the previous code to (N-1). And implementing the (Load% N) , in assigning the load statement, we have configured the counter to work as a Mod N counter, wherever the value has to be given accordingly. So the module has now been assigned to work for Mod N counter according to the value of N assigned.

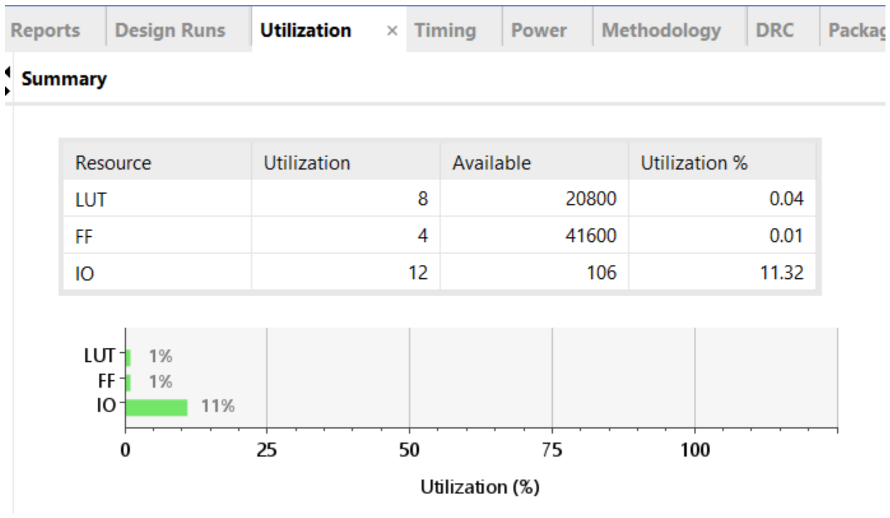
RTL Schematic



Resource Utilization

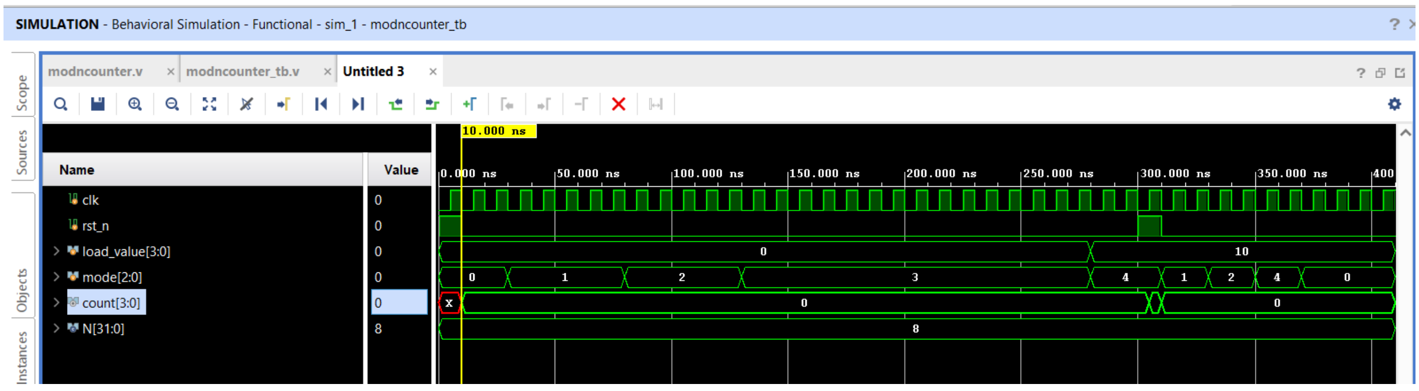


Name	Slice LUTs (20800)	Slice Registers (41600)	Slice (8150)	LUT as Logic (20800)	Bonded IOB (106)	BUFGCTRL (32)
N modncounter	8	4	5	8	12	1



Resource	Utilization	Available	Utilization %
LUT	8	20800	0.04
FF	4	41600	0.01
IO	12	106	11.32

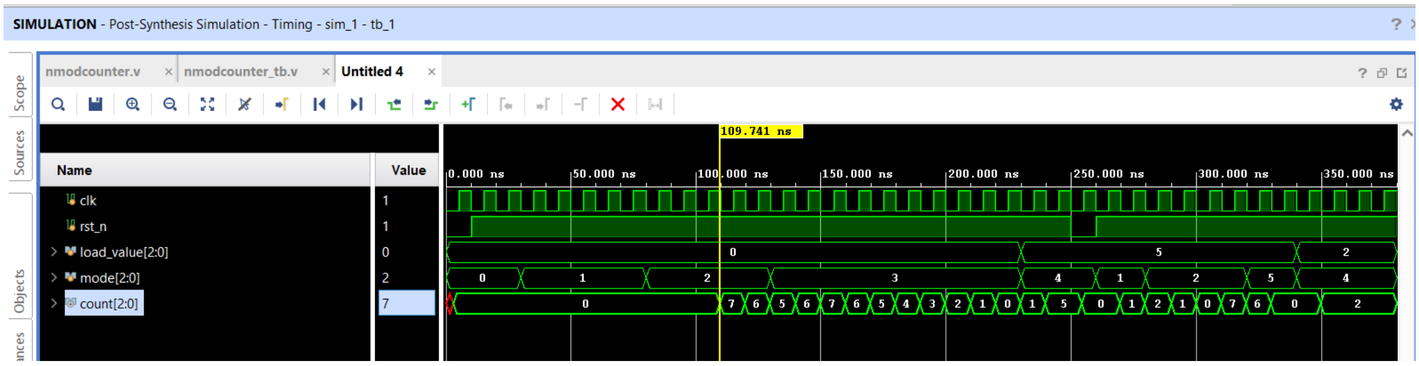
Timing Diagram(Behavioural)



SIMULATION - Behavioral Simulation - Functional - sim_1 - modncounter_tb

Name	Value
clk	0
rst_n	0
load_value[3:0]	0
mode[2:0]	0
count[3:0]	0
N[31:0]	8

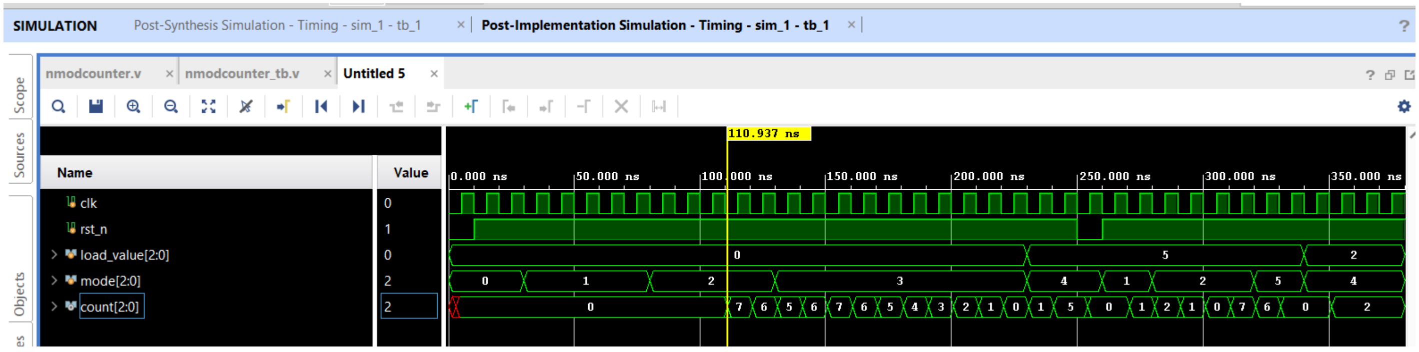
Timing Diagram(Post-Synthesis)



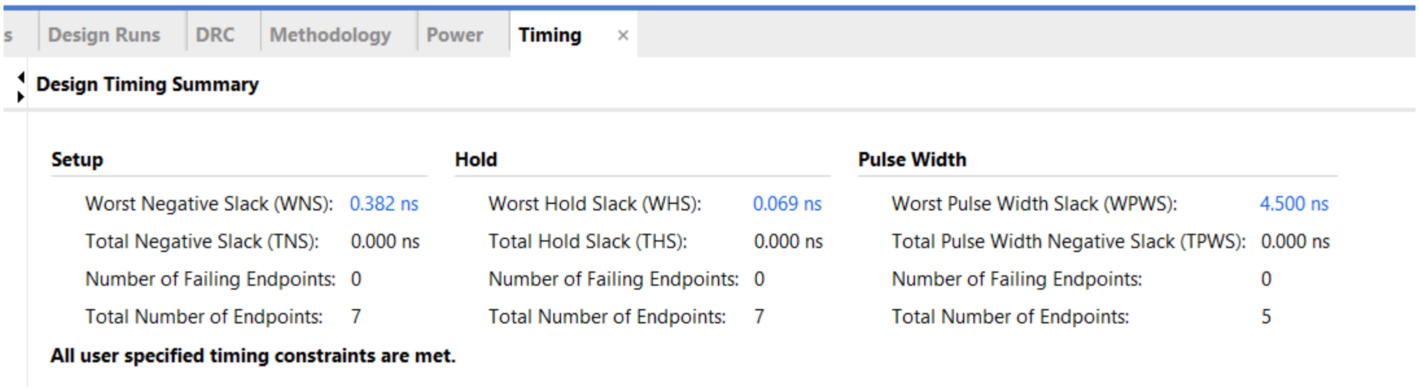
SIMULATION - Post-Synthesis Simulation - Timing - sim_1 - tb_1

Name	Value
clk	1
rst_n	1
load_value[2:0]	0
mode[2:0]	2
count[2:0]	7

Timing Diagram(Post Implementation)



Timing Report



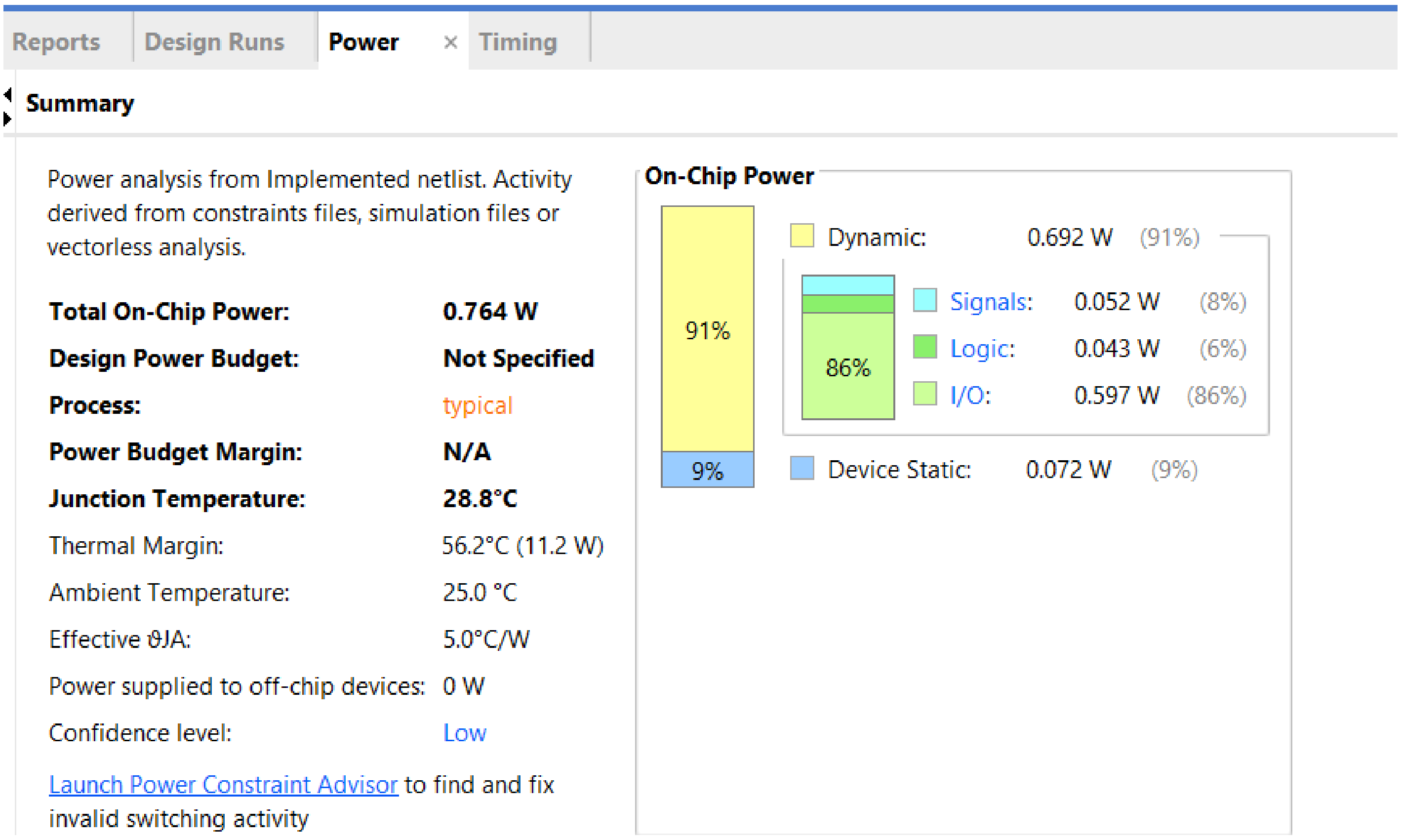
s | Design Runs | DRC | Methodology | Power | Timing

Design Timing Summary

Setup		**Hold**		**Pulse Width**	
Worst Negative Slack (WNS):	0.382 ns	Worst Hold Slack (WHS):	0.069 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	7	Total Number of Endpoints:	7	Total Number of Endpoints:	5

All user specified timing constraints are met.

Power Utilization(post Implementation)



Reports | Design Runs | Power | Timing

Summary

Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

Total On-Chip Power:	**0.764 W**
Design Power Budget:	**Not Specified**
Process:	typical
Power Budget Margin:	**N/A**
Junction Temperature:	**28.8°C**
Thermal Margin:	56.2°C (11.2 W)
Ambient Temperature:	25.0 °C
Effective ϑJA:	5.0°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Launch Power Constraint Advisor to find and fix invalid switching activity

On-Chip Power

Dynamic:	0.692 W	(91%)	
Signals:	0.052 W	(8%)	
Logic:	0.043 W	(6%)	
I/O:	0.597 W	(86%)	
Device Static:	0.072 W	(9%)	

Conclusion

The parameterized counter was successfully verified through functional simulations. The dynamic Mod-N implementation enabled **scalability**, making the design more flexible. Power and timing analysis showed that increasing **N** slightly increased the delay but remained within acceptable FPGA constraints.

3D. a) Design a synchronous mod-8 up-down counter (with synchronous active low reset) using Verilog. Simulate and test with a testbench.

b) Add the following functionalities as discussed in the class:

a. Hold previous value

b. Count up

c. Count down

d. Count up-down (first up, then down)

e. Load an input value

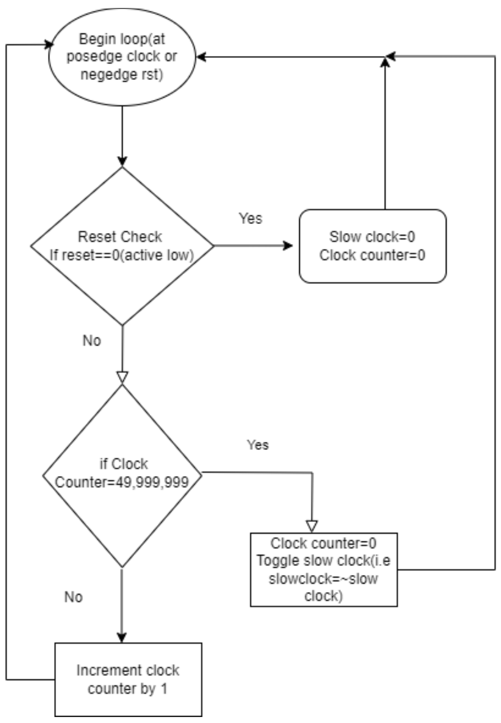
f. Local synchronous reset

c)Parameterize and make the counter behave as a mod-N up-down counter (N ≤ 16).

d) Now, let us implement the design on the FPGA. To visualize the counter you designed using the LEDs on the Basys3 board, the clock frequency must be reduced from 100MHz (default Basys3 board frequency) to ~1 Hz. Design a clock frequency divider block (basic implementation) to achieve this. Map N (as an input) to the switches available on the board. Also, convert the output to thermometric code for better visualization with the LEDs.

Introduction

The counter was **implemented on the Basys3 FPGA board**, and its output was visualized using the **on-board LEDs**. Since the FPGA clock operates at **100 MHz**, a **clock divider** was designed to reduce the frequency to **1 Hz** for human-visible counting. Additionally, the counter output was converted into **thermometric code** for better LED visualization.. In this problem we were forced to design a clock of 1Hz , since then counter can actually count for every 1 second. We have used the internal clock of the FPGA as a source which is of 100MHz. We have now counted 1000000000. Since these many oscillations occur in one second, which is literally the meaning of Frequency(no of oscillations per second, hence by counting upto (MAX_Frqu_count/2), in order to generate 0.5sec ON time and 0.5 sec OFF time there by generating a 1sec(1Hz) clock. The steps are explained in detail here.


Begin loop(at posedge clock or negedge rst)
Reset Check If reset==0(active low)
Yes
Slow clock=0 Clock counter=0
No
if Clock Counter=49,999,999
Yes
Clock counter=0 Toggle slow clock(i.e slowclock=~slow clock)
No
Increment clock counter by 1

Here we will implement a Thermometric LED visualization of the counter on the FPGA. The count from the previous answers is taken and another new module for thermometric code conversion is created. This is done by simply Left shifting 1 according to the value of count , then subtracting 1 to get thermometric code. The steps to get the above said procedure is explained below

Initial Values : **count=5**

So , we left shift 1 by 5.

Step 1: Left Shift

Initial value of 1: 0000000000000001 (binary)

Shift left by 5:   0000000000100000 (binary)

Step 2: Subtract 1

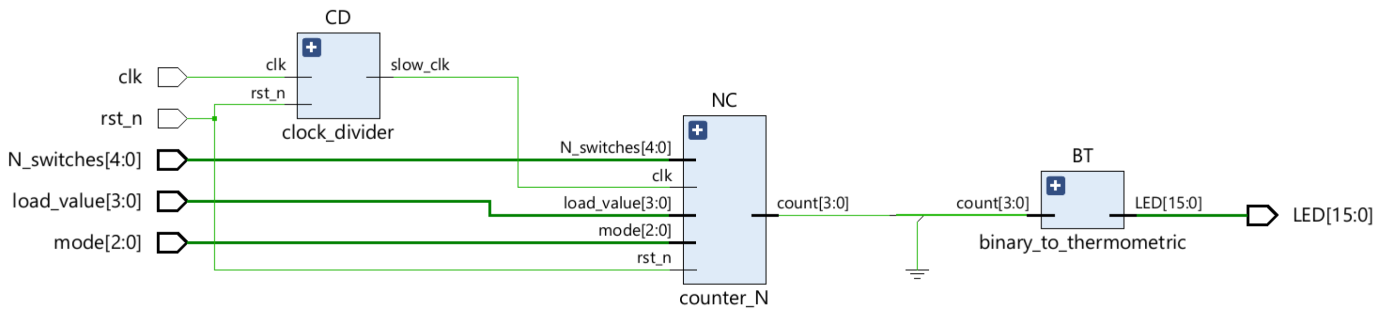
Shifted value: 0000000000100000 (binary)

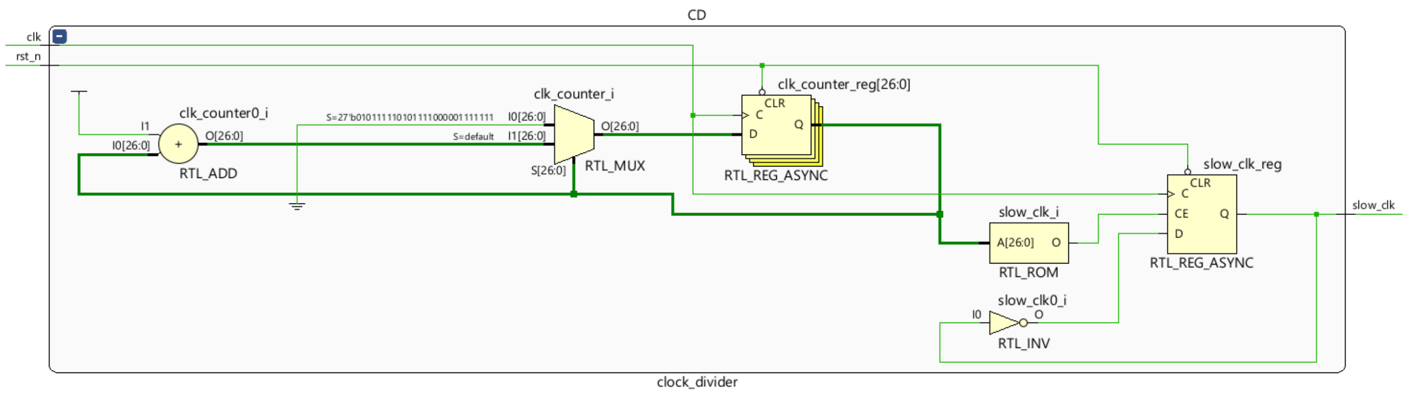
Subtract 1:   0000000000011111 (binary)

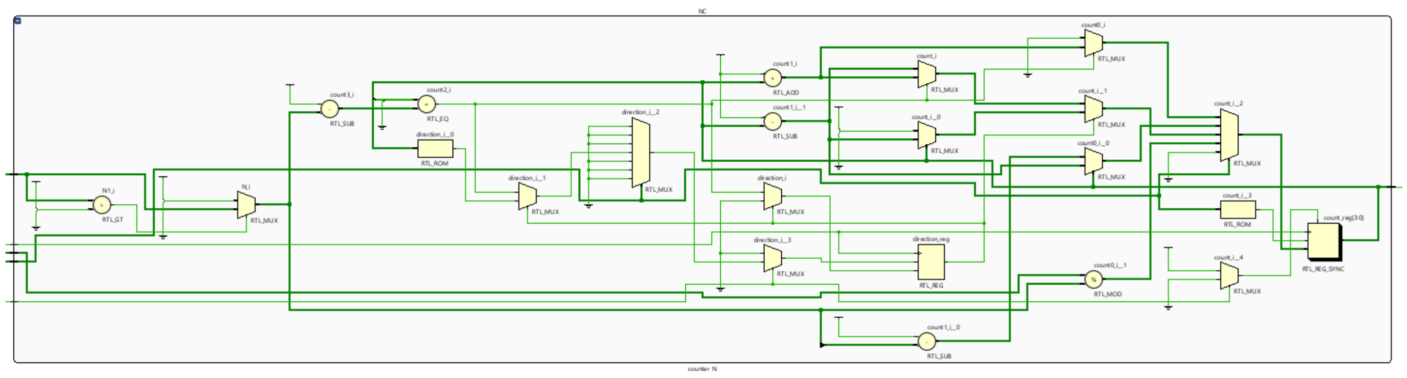
Final Thermometric Output:

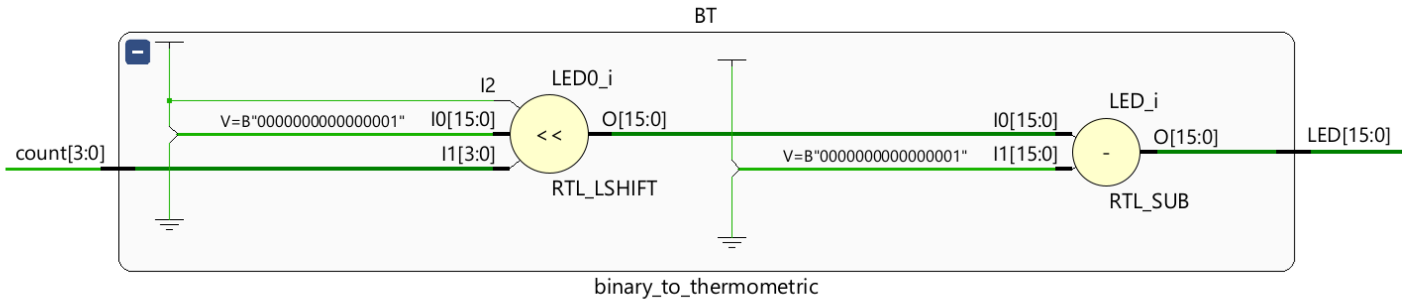
LED = 0000000000011111 (binary)

RTL Block

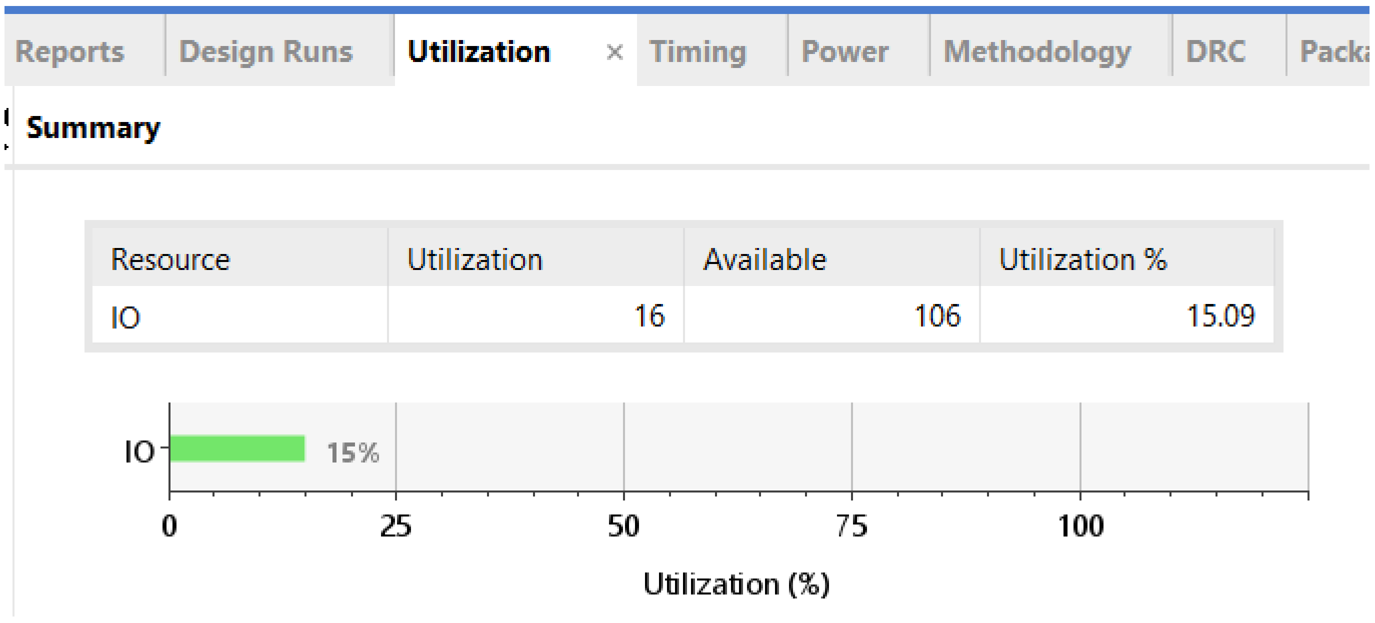








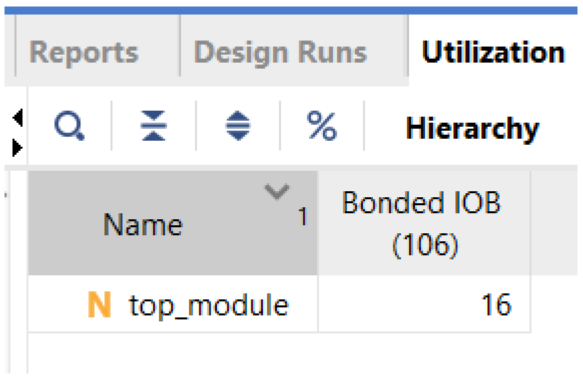
Resource Utilization



Reports | Design Runs | **Utilization** × | Timing | Power | Methodology | DRC | Packa

Summary

Resource	Utilization	Available	Utilization %
IO	16	106	15.09



Reports | Design Runs | **Utilization**

Hierarchy

Name	Bonded IOB (106)
top_module	16

Pin Distribution on FPGA

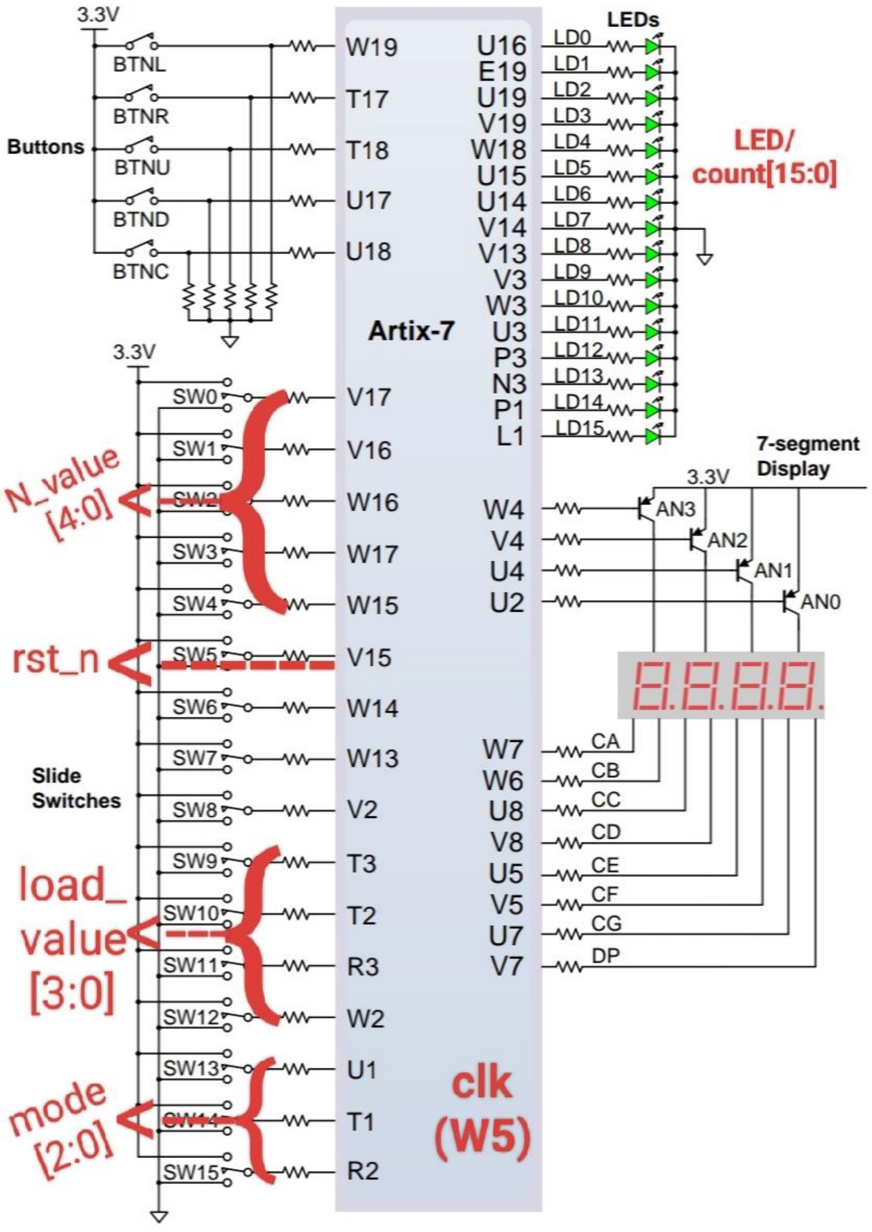
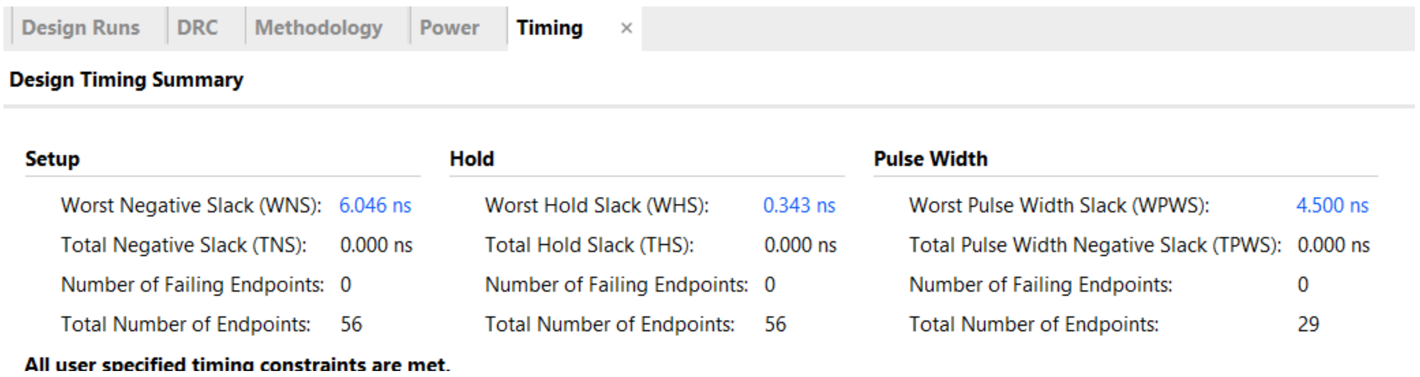


Figure 16. General purpose I/O devices on the Basys 3.

Timing Report

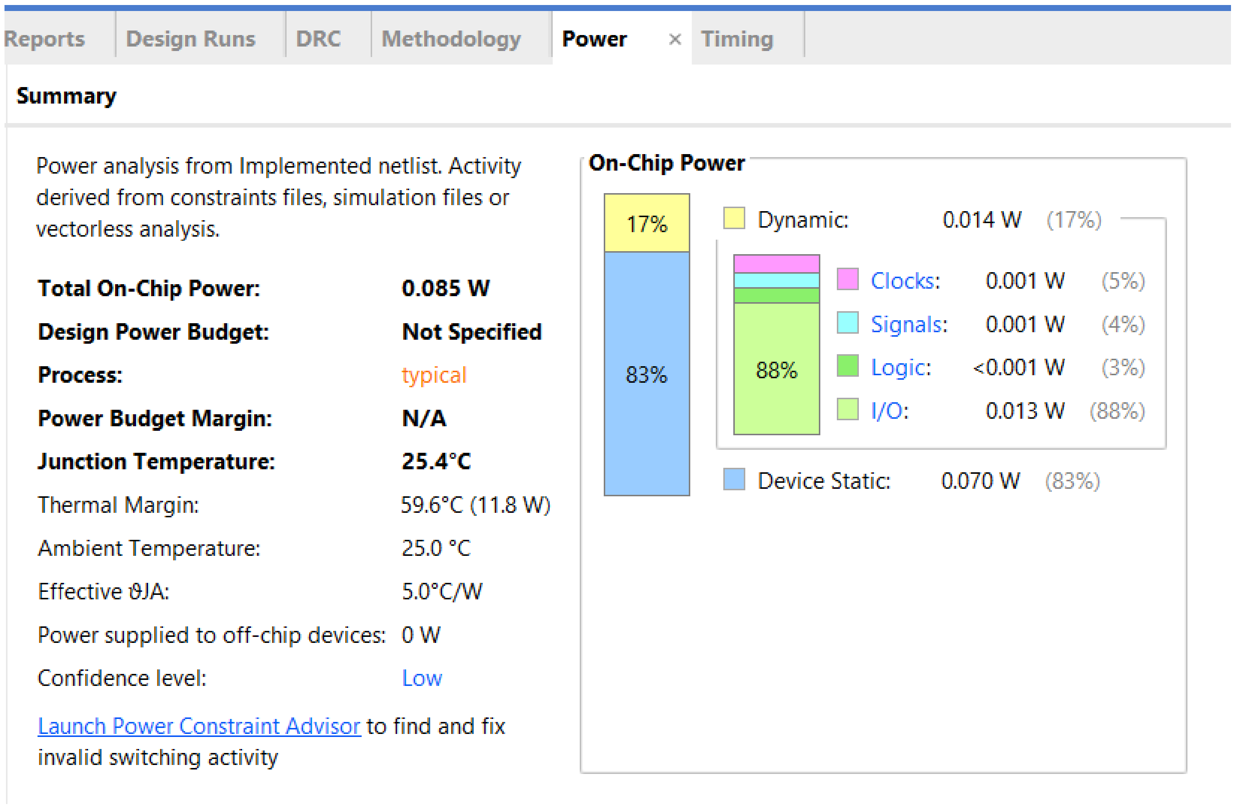

Design Runs | DRC | Methodology | Power | **Timing** ×

Design Timing Summary

Setup		**Hold**		**Pulse Width**	
Worst Negative Slack (WNS):	6.046 ns	Worst Hold Slack (WHS):	0.343 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	56	Total Number of Endpoints:	56	Total Number of Endpoints:	29

All user specified timing constraints are met.

Power Utilization



Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

Total On-Chip Power:	**0.085 W**
Design Power Budget:	**Not Specified**
Process:	typical
Power Budget Margin:	**N/A**
Junction Temperature:	**25.4°C**
Thermal Margin:	59.6°C (11.8 W)
Ambient Temperature:	25.0 °C
Effective ϑJA:	5.0°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Launch Power Constraint Advisor to find and fix invalid switching activity

On-Chip Power

Dynamic:	0.014 W	(17%)	
Clocks:	0.001 W	(5%)	
Signals:	0.001 W	(4%)	
Logic:	<0.001 W	(3%)	
I/O:	0.013 W	(88%)	
Device Static:	0.070 W	(83%)	

Conclusion

The **1 Hz clock divider** was successfully implemented, and the counter operated at a visible rate. The **thermometric code conversion** allowed a better LED display by progressively lighting up LEDs according to the counter value. Timing analysis showed that the **clock divider introduced some delay**, but the FPGA was able to handle the design efficiently.

3E. a) Design a synchronous mod-8 up-down counter (with synchronous active low reset) using Verilog. Simulate and test with a testbench.

b) Add the following functionalities as discussed in the class:

a. Hold previous value

b. Count up

c. Count down

d. Count up-down (first up, then down)

e. Load an input value

f. Local synchronous reset

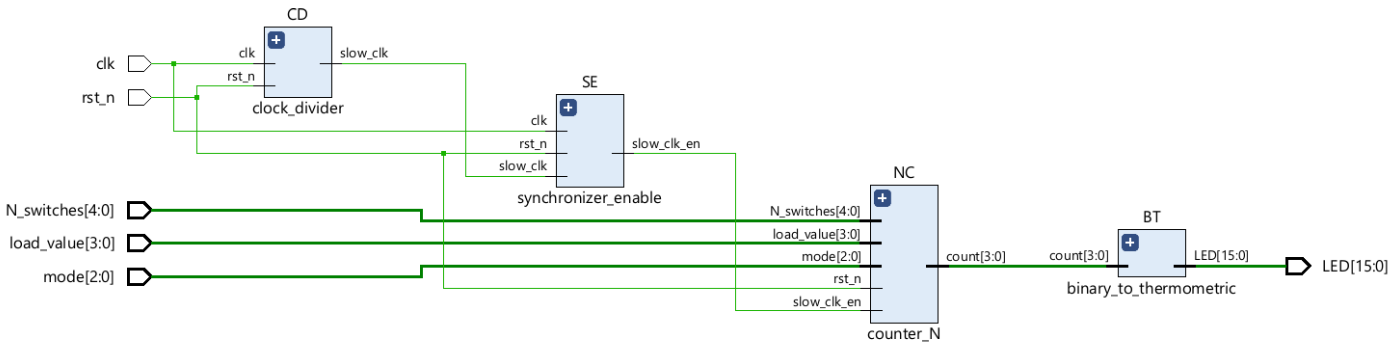
c)Parameterize and make the counter behave as a mod-N up-down counter (N ≤ 16).

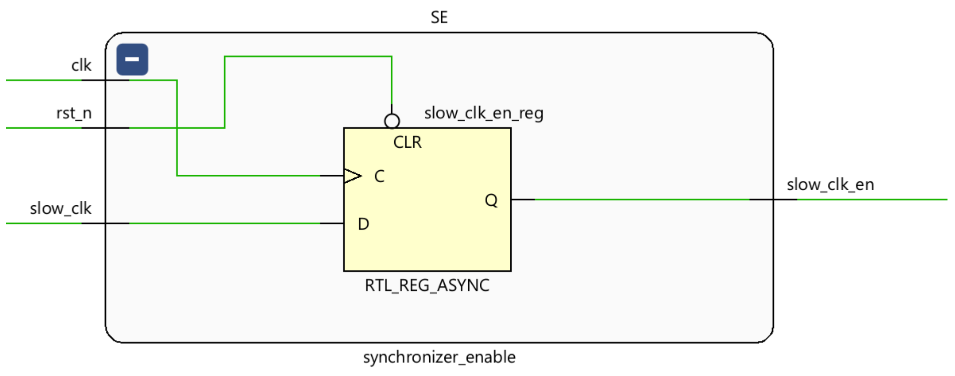
d) Now, let us implement the design on the FPGA. To visualize the counter you designed using the LEDs on the Basys3 board, the clock frequency must be reduced from 100MHz (default Basys3 board frequency) to ~1 Hz. Design a clock frequency divider block (basic implementation) to achieve this. Map N (as an input) to the switches available on the board. Also, convert the output to thermometric code for better visualization with the LEDs.

e) This internally generated clock is not part of the clock tree synthesis and might cause timing violations as dedicated FPGA clock generators do not generate it. Hence, a slower clock enable signal is recommended, which allows all logic in the design to be driven by the same clock. Design this circuit and demonstrate the results on the FPGA.

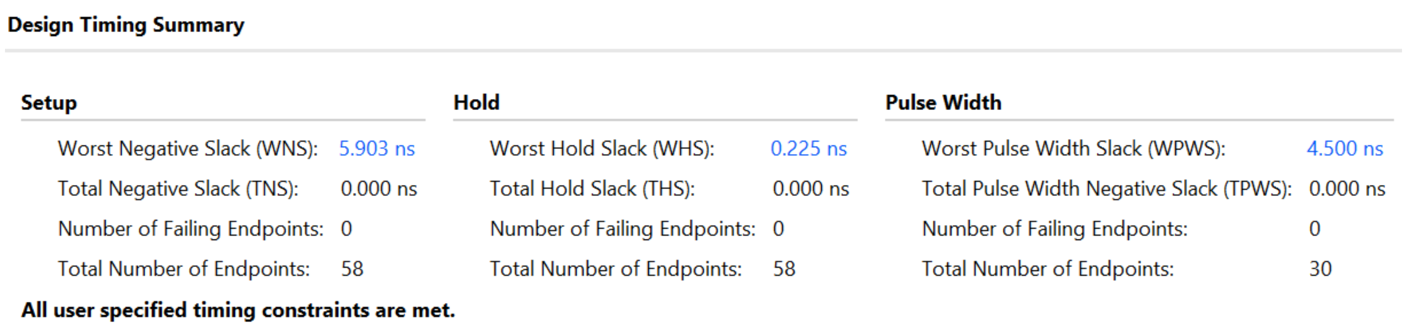
Introduction

In FPGA design, using an internally generated divided clock (slow_clk) can lead to **timing violations and clock tree synthesis issues**. Instead of generating a new slow clock, a **clock enable signal (slow_clk_en)** was implemented. This method ensures that the counter **updates at 1 Hz while all logic remains driven by the original 100 MHz clock**.





Timing Report

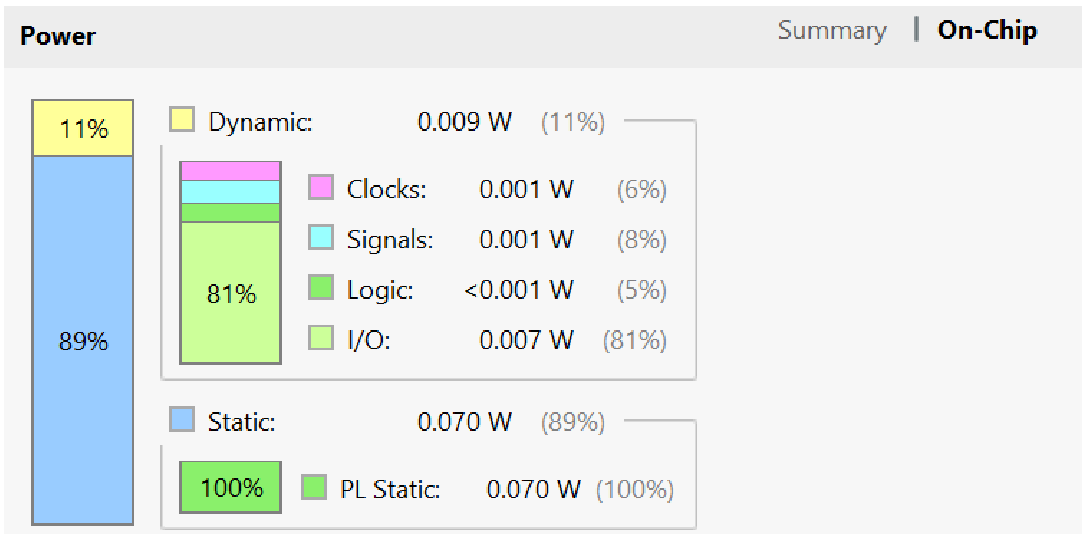


Design Timing Summary

Setup		**Hold**		**Pulse Width**	
Worst Negative Slack (WNS):	5.903 ns	Worst Hold Slack (WHS):	0.225 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	58	Total Number of Endpoints:	58	Total Number of Endpoints:	30

All user specified timing constraints are met.

Power Report



Power Summary | **On-Chip**

Dynamic:	0.009 W	(11%)	
	Clocks:	0.001 W	(6%)
	Signals:	0.001 W	(8%)
	Logic:	<0.001 W	(5%)
	I/O:	0.007 W	(81%)
Static:	0.070 W	(89%)	
	PL Static:	0.070 W	(100%)

Conclusion

The **clock enable approach** resolved potential **clock domain crossing issues**, ensuring **smoother FPGA timing closure**. The counter **remained synchronous to the 100 MHz clock**, eliminating problems caused by internal clock division. This implementation improved **timing accuracy, reduced clock skew**, and ensured FPGA-friendly synthesis. Each module was tested for **functional correctness, timing performance, and power utilization**. The final FPGA implementations demonstrated proper design principles, showcasing **real-world digital system design challenges and solutions**.