Lab 5 - BCD Arithmetic Logic Unit

Group A - Yuta Akiya, Kyle Le, Megan Luong

Prof. Aly

ECE 4304

Architecture

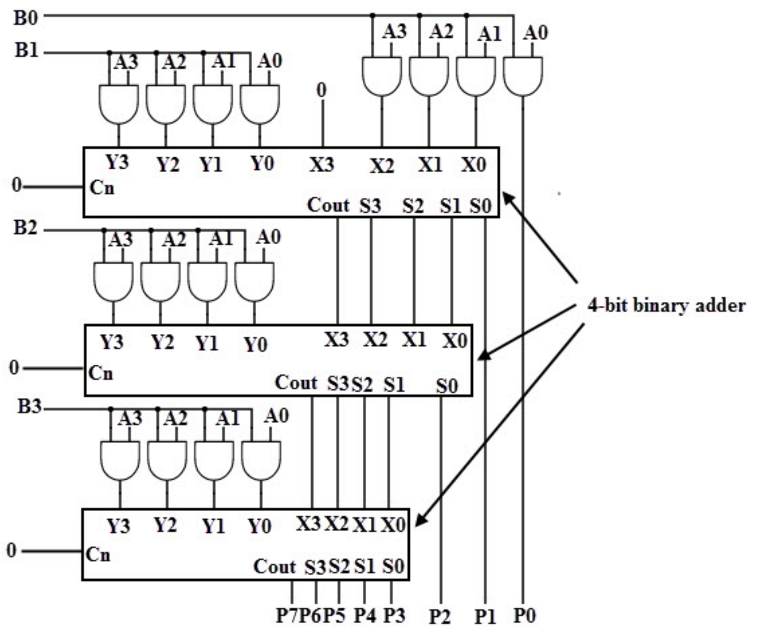
ALU

Addition / Subtraction

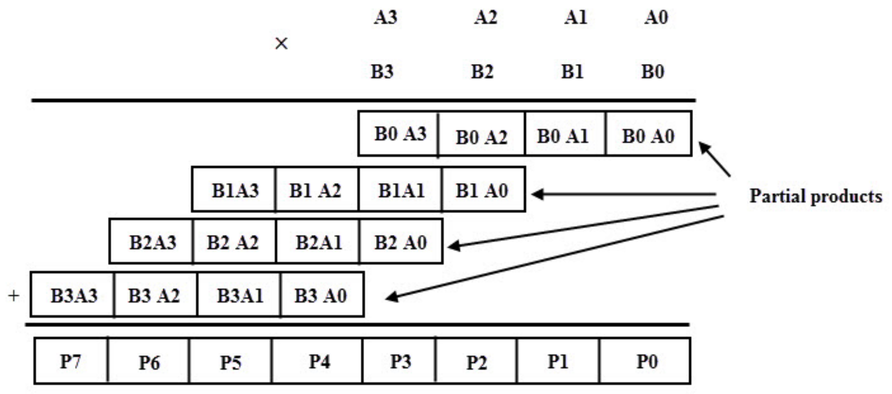
For the addition and subtraction portion of the ALU, no special circuit was designed. Since the ALU involves two 4-bit inputs, built-in synthesized adders were used.

Multiplication

For the multiplication portion of the ALU, it was done using a parallel full adder circuit. By using parallel full adders, the multiplication is able to be done within one 125 MHz clock cycle. The downsides to this architecture is an increase in resources, but an increase in performance.



This design is based on the arithmetic structure of performing multiplication.


A3 A2 A1 A0
×
B3 B2 B1 B0
B0 A3 | B0 A2 | B0 A1 | B0 A0
B1A3 | B1 A2 | B1A1 | B1 A0
B2A3 | B2 A2 | B2A1 | B2 A0
+ B3A3 | B3 A2 | B3A1 | B3 A0
P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0
Partial products

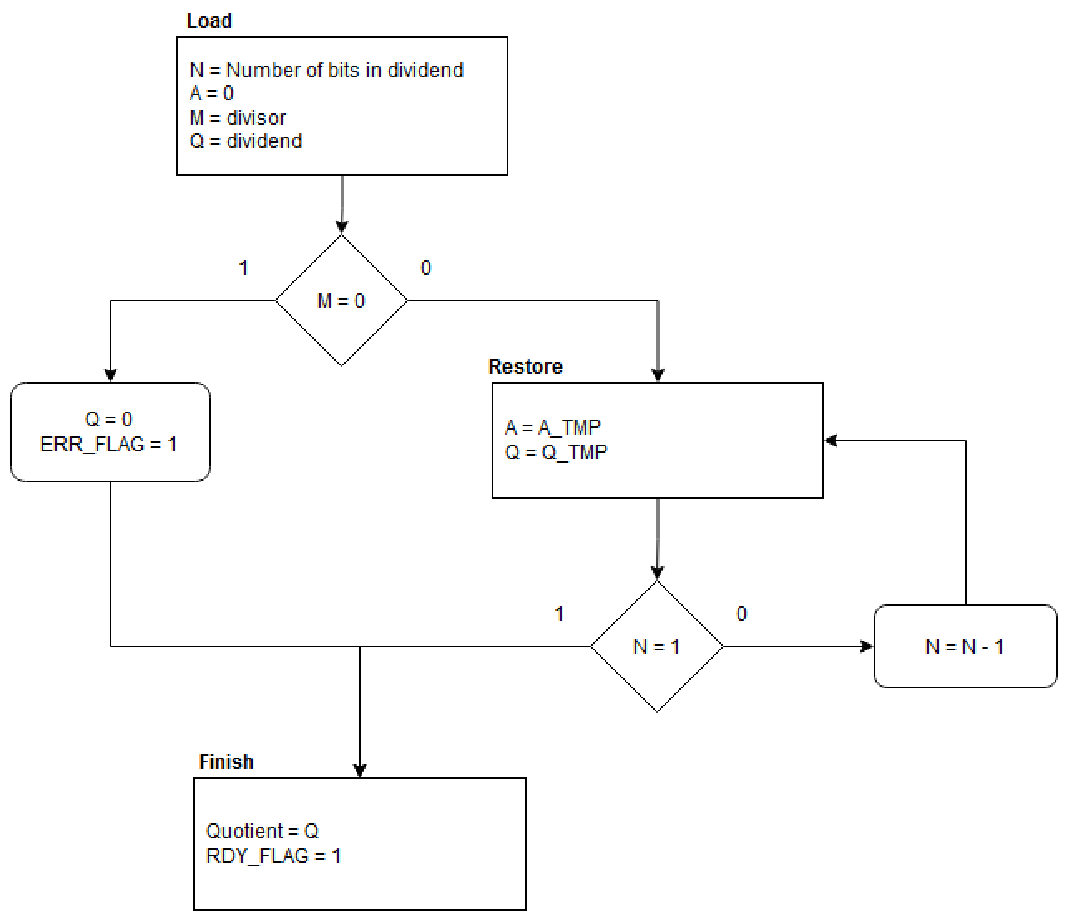
Images from: https://www.electronicshub.org/binary-multiplication/

Division

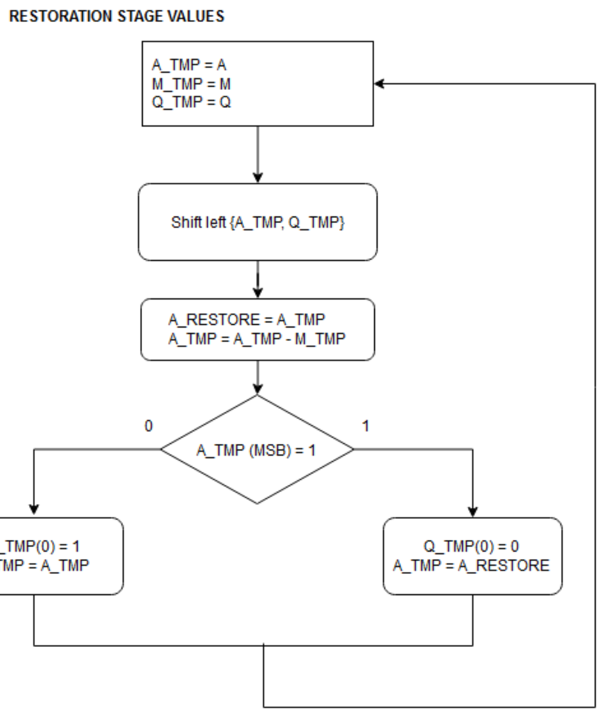
For the division portion of the ALU, it was done using the restoring division method. This provided a quick and efficient division method that did not take as many clock cycles as compared to purely arithmetic division.

Since this design is multi-cycled, a finite-state machine was implemented to deal with the logic.

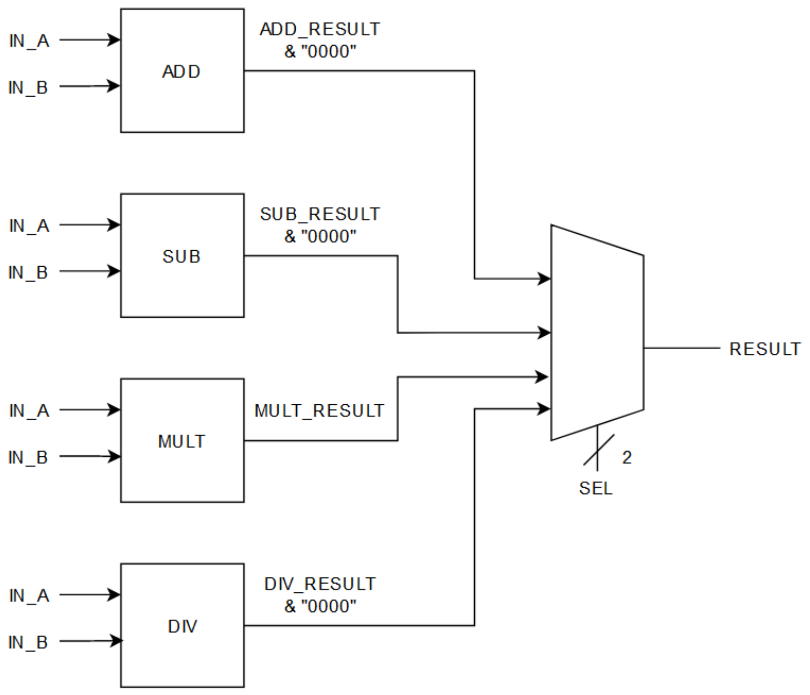
Division FSM



A separate image was made to specify how the values in the Restore stage are calculated.

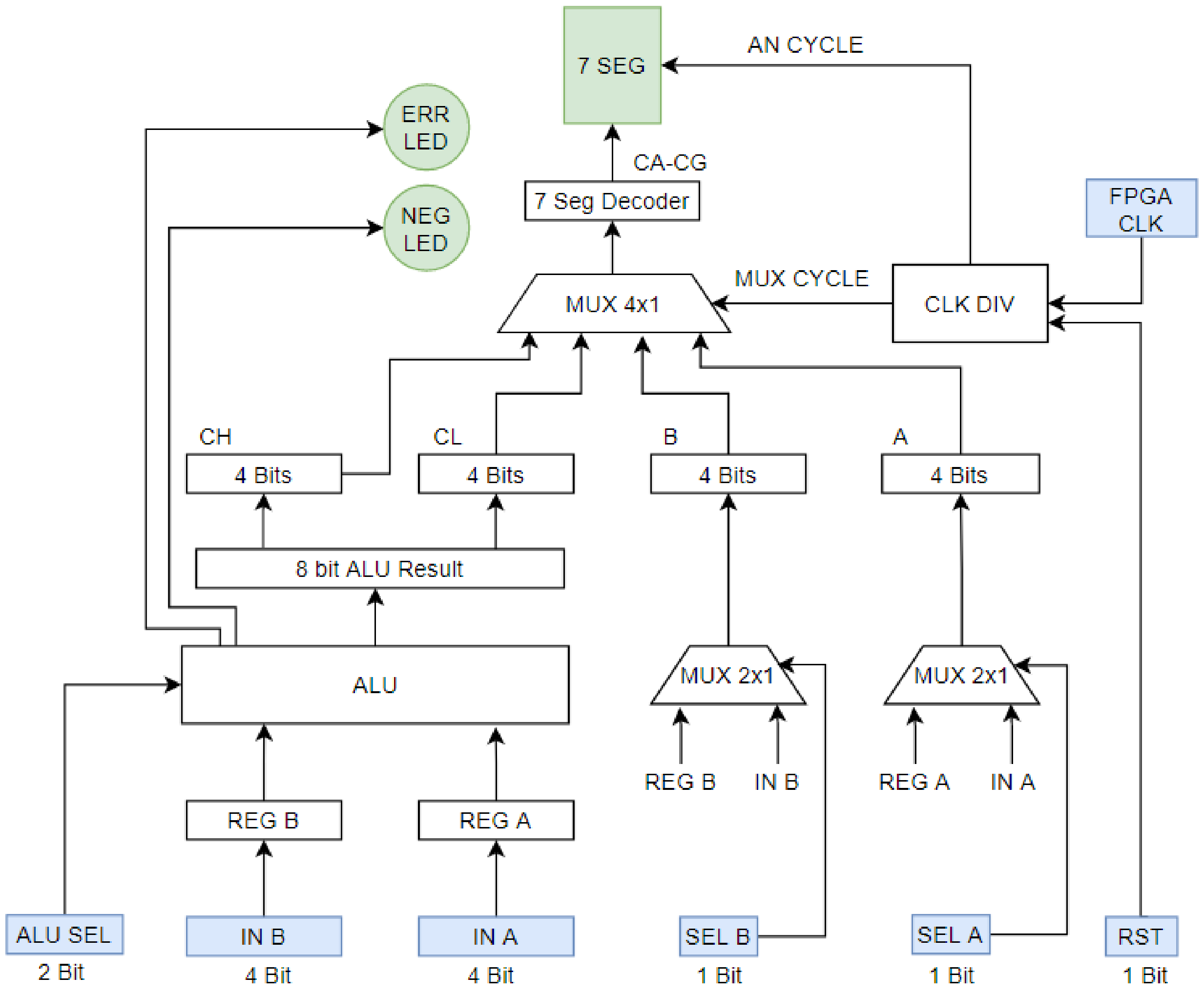


Full ALU circuit



Full Project Architecture

The highest level of the project architecture implements the FPGA clock, switches, and LEDs to visualize and create an interface for the ALU module which was detailed in the previous sections. Seven segment displays are used to display both inputs A and B (in hex/BCD) and the output C in two digit BCD. A clock divider is used to refresh the 4 seven segment displays being used on the board refreshes every 1.31 ms similar to lab 3. The ALU select bits specify the desired operation on the ALU, where “00” is addition, “01” is subtraction, “10” is multiplication, and “11” is division. The ERR LED is turned on when there is a division by 0 error and the NEG LED is turned on when a subtraction operation results in a negative number.



Code Design/Tricks

ALU

Addition / Subtraction

Since the inputs involve a low amount of bits, the addition and subtraction portions for the ALU can be programmed using behavioral structure.

For subtraction, it was important to detect if the result was a negative number. This allows the output to be displayed properly as a BCD onto the 7-segment displays. To do this, the registers used for subtraction had their size increased by 1 and converted to signed. If the result was negative, the result was 2’s complement and the negative flag was set high. If the result was positive, no changes were made and the negative flag was set low. To keep the output 4-bits, the value taken was without the signed bit.

Multiplication

For multiplication, the design was cascaded and looped, similar to the generic decoder. According to the design, an input array was made for the Y inputs for the full adders. Then, a carry array for handling the values from one full adder to another was made. The final output is taken from certain stages of the carry array and the final value.

Division

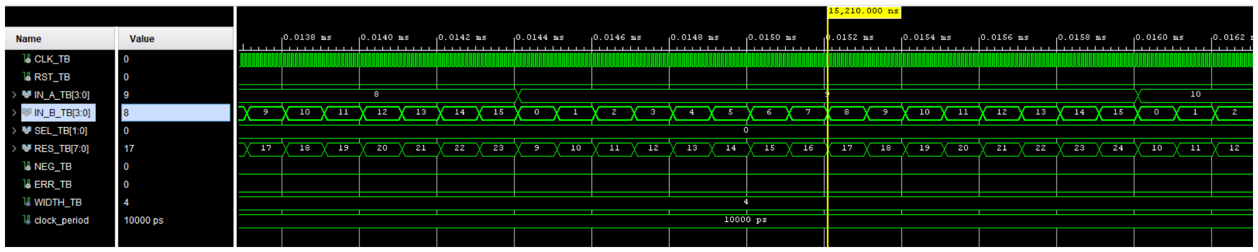
For division, a finite state machine was created to implement the design in the architecture. A state was made for the load, restore, and finish, with a separate process for handling the restorative values. Since the clock cycles required to update the 7-segment displays is exponentially larger than the clock cycles required for division, there is enough time for the values to update before the display updates.

Full ALU Circuit

All the components for the ALU functions were combined together into the ALU entity. Depending on the input switches provided by the user, the appropriate output is muxed out of the ALU component. Since the addition, subtraction, and division components will only produce a 4-bit output with 4-bit inputs, their values will have to be padded with leading 0’s. This is to keep the size in-line with the 8-bit output of the multiplication component.

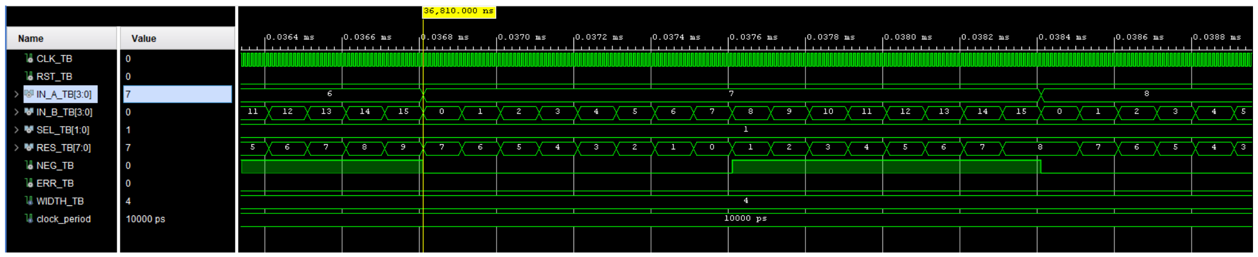
Corner Cases

Addition Testbench



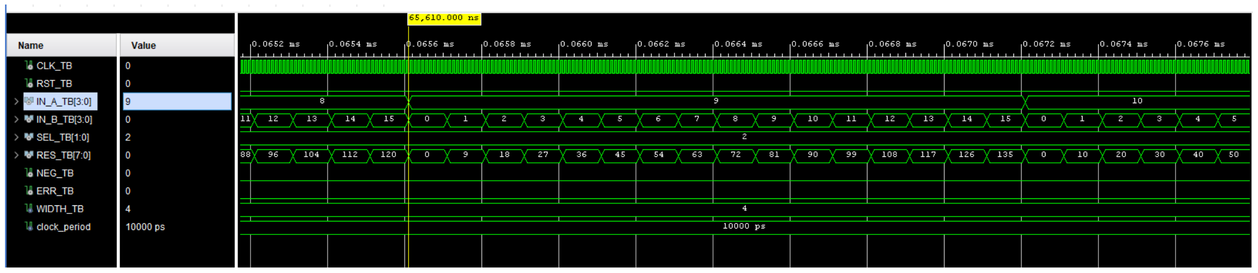
Addition is valid for all values (0-15) + (0-15), covering all possible cases for 4-bit addition.

Subtraction Testbench



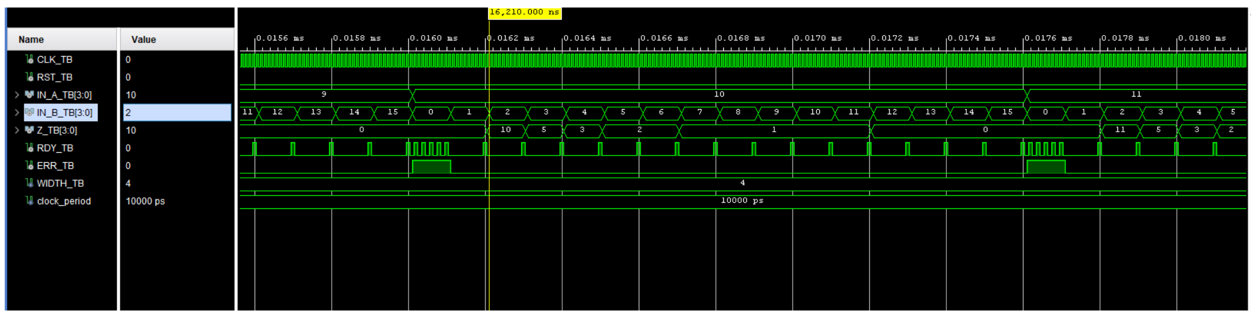
For subtraction, an edge case occurs when the result is a negative number. The testbench shows that the design deals with it promptly, returning the absolute value with a negative flag.

Multiplication Testbench



For multiplication, an edge case occurs when the result is larger than 4-bits. The design needs to ensure that the output will fit into 8-bits. The testbench shows that values are valid from 9 * (0-15), which produces results greater than 4-bits.

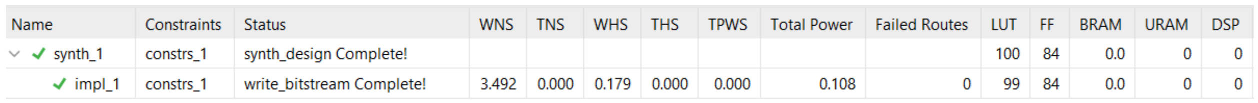
Division Testbench



When divide by 0 occurs, the ERR flag goes high to show an error. Else, the calculations take the rounded-down quotient of A / B.

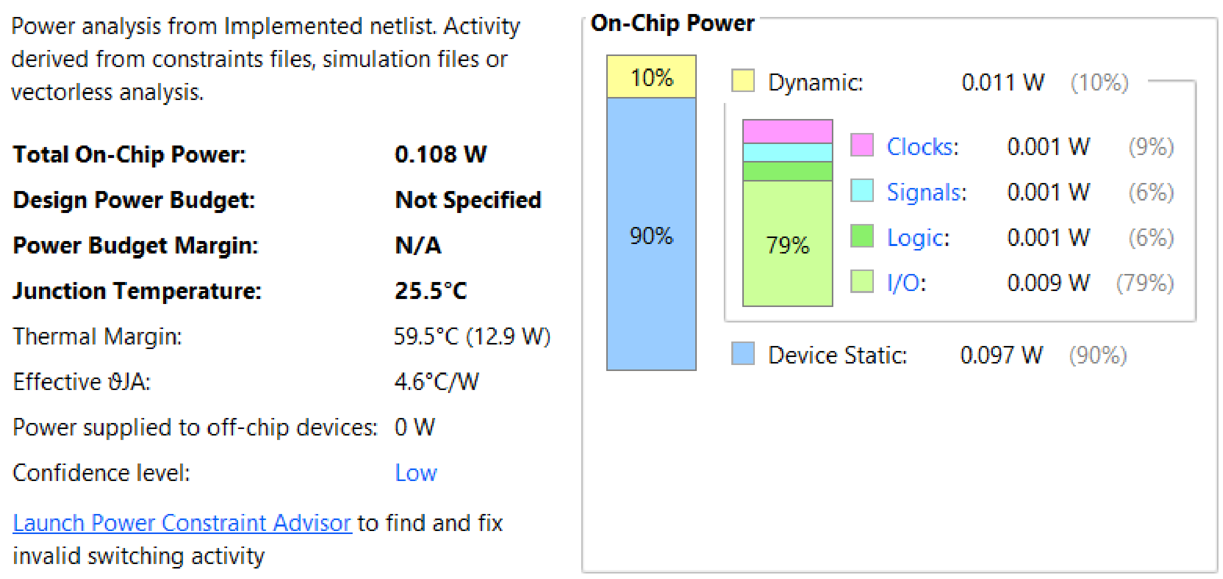
Area/Resource Information

Resource Usage of Entire System



Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAM	URAM	DSP
✓ synth_1	constrs_1	synth_design Complete!								100	84	0.0	0	0
✓ impl_1	constrs_1	write_bitstream Complete!	3.492	0.000	0.179	0.000	0.000	0.108	0	99	84	0.0	0	0

Power Usage Details



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

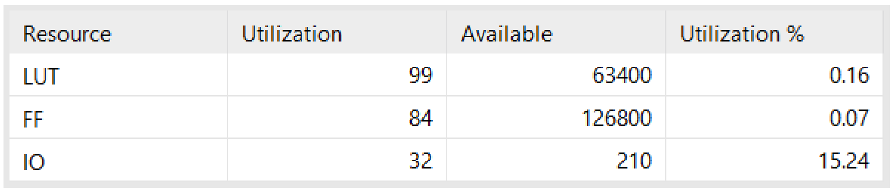
Total On-Chip Power:	**0.108 W**
Design Power Budget:	**Not Specified**
Power Budget Margin:	**N/A**
Junction Temperature:	**25.5°C**
Thermal Margin:	59.5°C (12.9 W)
Effective ϑJA:	4.6°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Launch Power Constraint Advisor to find and fix invalid switching activity

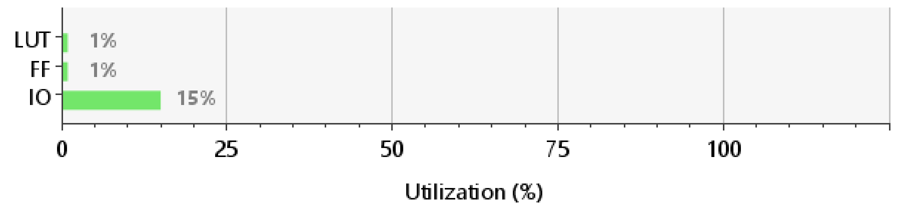
On-Chip Power

🟨	Dynamic:	0.011 W	(10%)
🟪	Clocks:	0.001 W	(9%)
🟦	Signals:	0.001 W	(6%)
🟩	Logic:	0.001 W	(6%)
🟩	I/O:	0.009 W	(79%)
🟦	Device Static:	0.097 W	(90%)

Utilization



Resource	Utilization	Available	Utilization %
LUT	99	63400	0.16
FF	84	126800	0.07
IO	32	210	15.24


LUT
FF
IO
Utilization (%)