ECE 3300L

Lab Report #2

Group A

Edwin Estrada (#015897050)

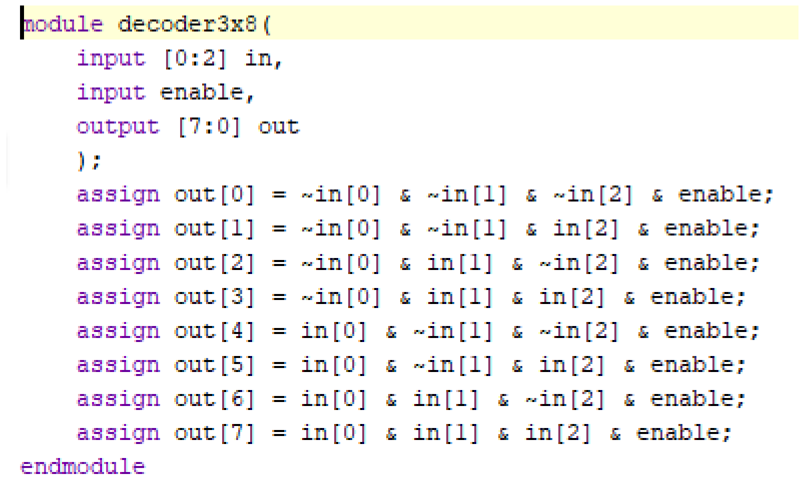
Michelle Lau (#016027427)

June 25, 2025

Design

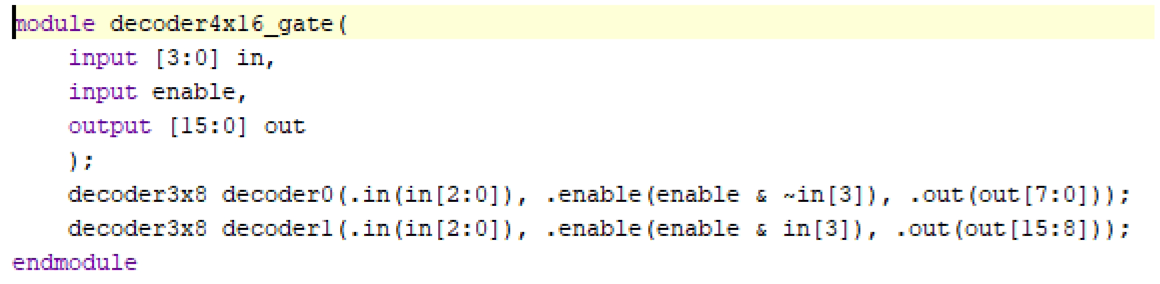
Gate-level design is closely linked to the hardware setup. The physical structure of the circuit has to be known beforehand when implementing a gate-level designed code. To implement the 4x16 decoder on the gate-level, we first created a 3x8 decoder module and then used two 3x8 decoders to make a 4x16 module. We used the MSB of the input as part of the enable for the 3x8 decoders while the rest of the input was used for the 3x8 decoder's input.

Decoder 3x8 Gate-Level Module



```
module decoder3x8(
    input [0:2] in,
    input enable,
    output [7:0] out
    );
    assign out[0] = ~in[0] & ~in[1] & ~in[2] & enable;
    assign out[1] = ~in[0] & ~in[1] & in[2] & enable;
    assign out[2] = ~in[0] & in[1] & ~in[2] & enable;
    assign out[3] = ~in[0] & in[1] & in[2] & enable;
    assign out[4] = in[0] & ~in[1] & ~in[2] & enable;
    assign out[5] = in[0] & ~in[1] & in[2] & enable;
    assign out[6] = in[0] & in[1] & ~in[2] & enable;
    assign out[7] = in[0] & in[1] & in[2] & enable;
endmodule
```

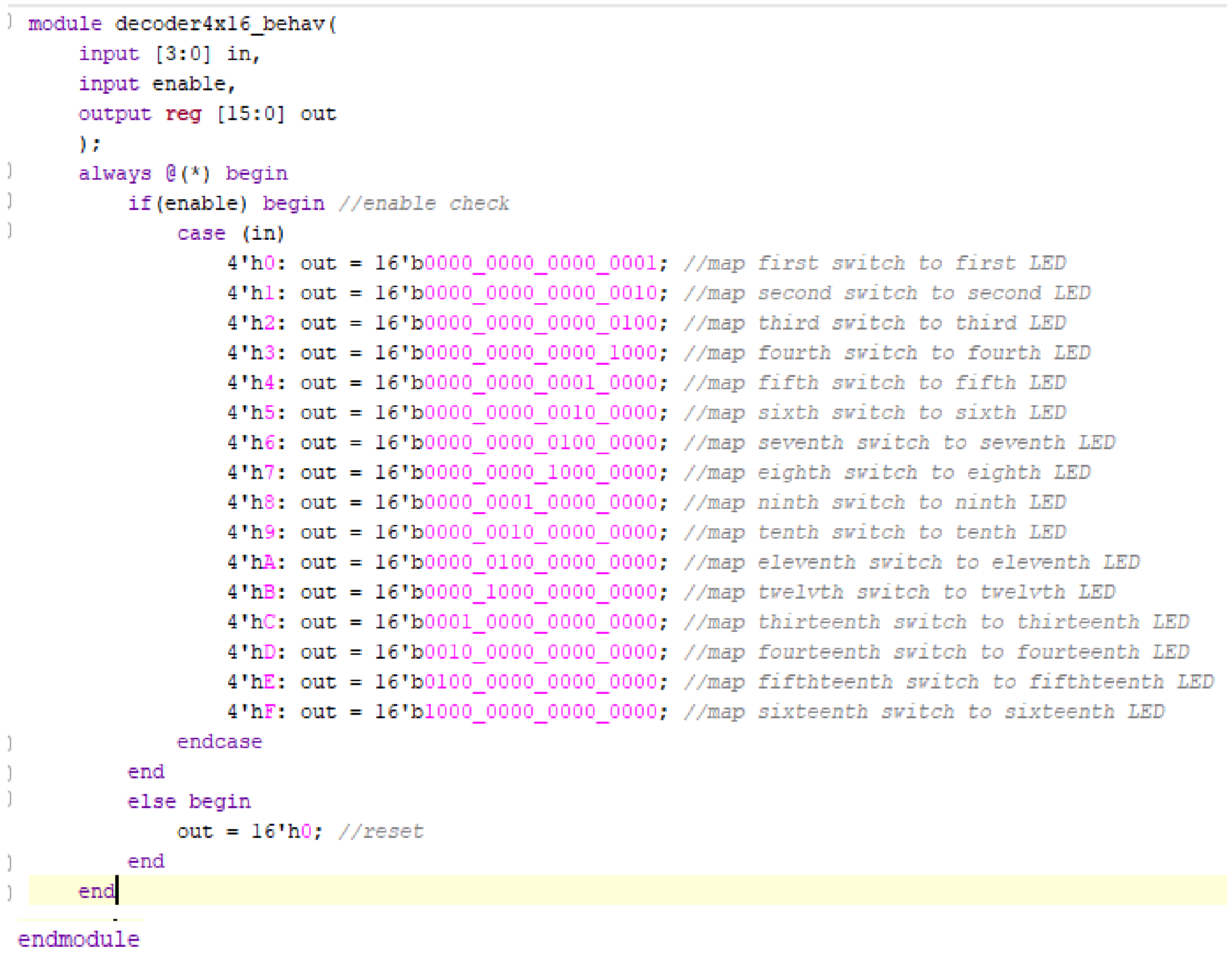
Decoder 4x16 Gate-Level Module



```
module decoder4x16_gate(
    input [3:0] in,
    input enable,
    output [15:0] out
    );
    decoder3x8 decoder0(.in(in[2:0]), .enable(enable & ~in[3]), .out(out[7:0]));
    decoder3x8 decoder1(.in(in[2:0]), .enable(enable & in[3]), .out(out[15:8]));
endmodule
```

On the other hand, behavioral design is able to describe the circuit's overall behavior in order to achieve the same results. The physical structure of the circuit is not needed and can be understood logically. Using behavioral modelling to create this version of the 4x16 decoder involved using an always block, then using a conditional block for the enable input and a case block for the select line inputs.

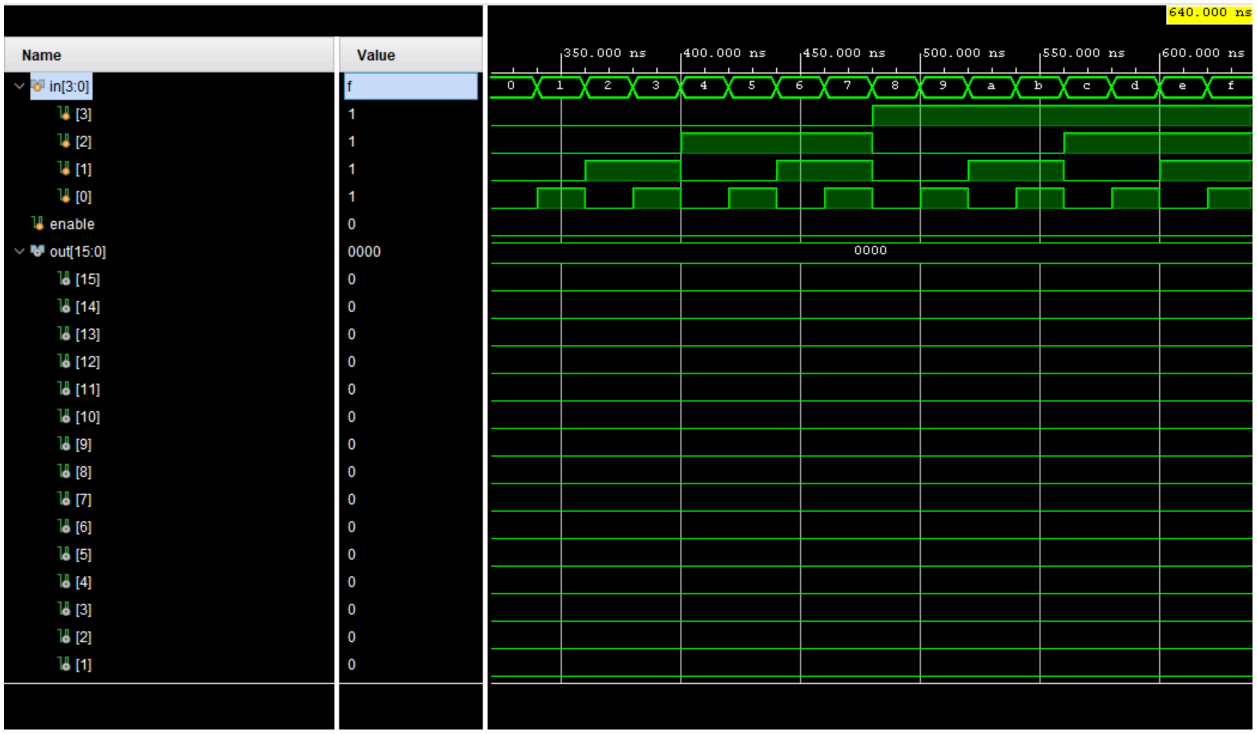
Decoder 4x16 Behavioral Modelling Module



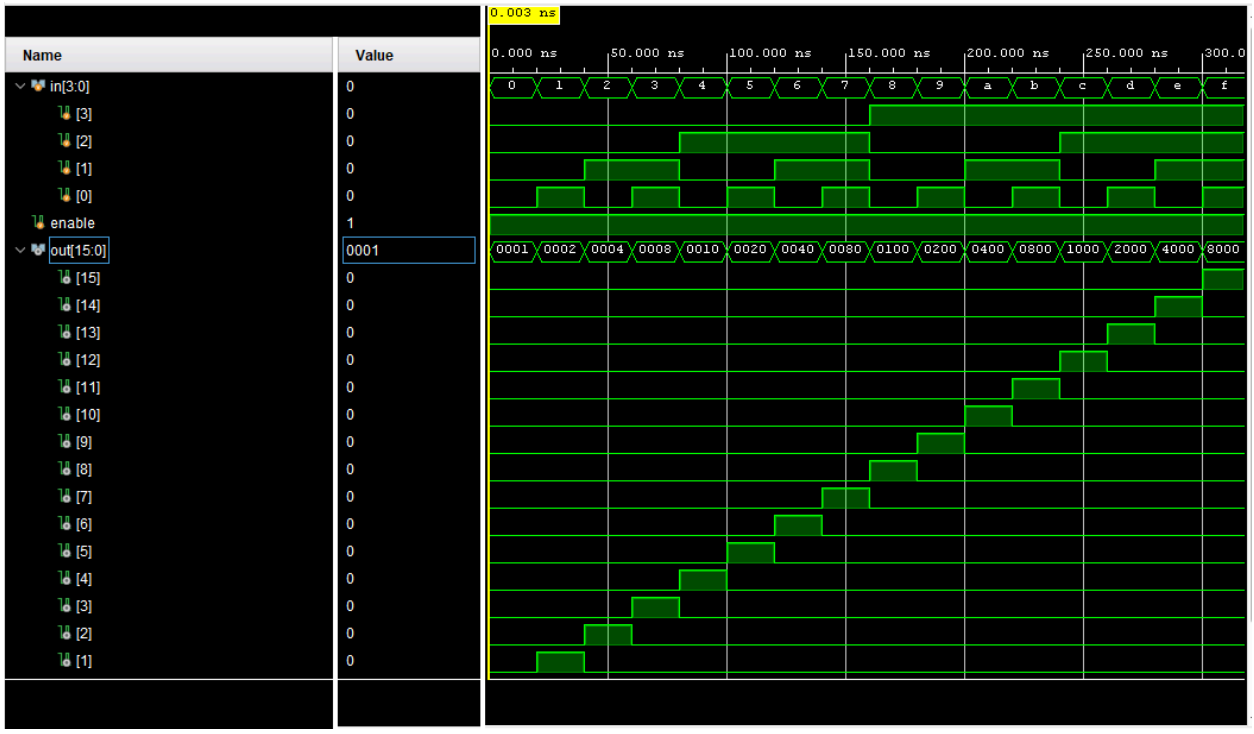
```verilog
module decoder4x16_behav(
    input [3:0] in,
    input enable,
    output reg [15:0] out
    );
    always @(*) begin
        if(enable) begin //enable check
            case (in)
                4'h0: out = 16'b0000_0000_0000_0001; //map first switch to first LED
                4'h1: out = 16'b0000_0000_0000_0010; //map second switch to second LED
                4'h2: out = 16'b0000_0000_0000_0100; //map third switch to third LED
                4'h3: out = 16'b0000_0000_0000_1000; //map fourth switch to fourth LED
                4'h4: out = 16'b0000_0000_0001_0000; //map fifth switch to fifth LED
                4'h5: out = 16'b0000_0000_0010_0000; //map sixth switch to sixth LED
                4'h6: out = 16'b0000_0000_0100_0000; //map seventh switch to seventh LED
                4'h7: out = 16'b0000_0000_1000_0000; //map eighth switch to eighth LED
                4'h8: out = 16'b0000_0001_0000_0000; //map ninth switch to ninth LED
                4'h9: out = 16'b0000_0010_0000_0000; //map tenth switch to tenth LED
                4'hA: out = 16'b0000_0100_0000_0000; //map eleventh switch to eleventh LED
                4'hB: out = 16'b0000_1000_0000_0000; //map twelvth switch to twelvth LED
                4'hC: out = 16'b0001_0000_0000_0000; //map thirteenth switch to thirteenth LED
                4'hD: out = 16'b0010_0000_0000_0000; //map fourteenth switch to fourteenth LED
                4'hE: out = 16'b0100_0000_0000_0000; //map fifthteenth switch to fifthteenth LED
                4'hF: out = 16'b1000_0000_0000_0000; //map sixteenth switch to sixteenth LED
            endcase
        end
        else begin
            out = 16'h0; //reset
        end
    end
endmodule
```

Simulation

Simulation Waveform

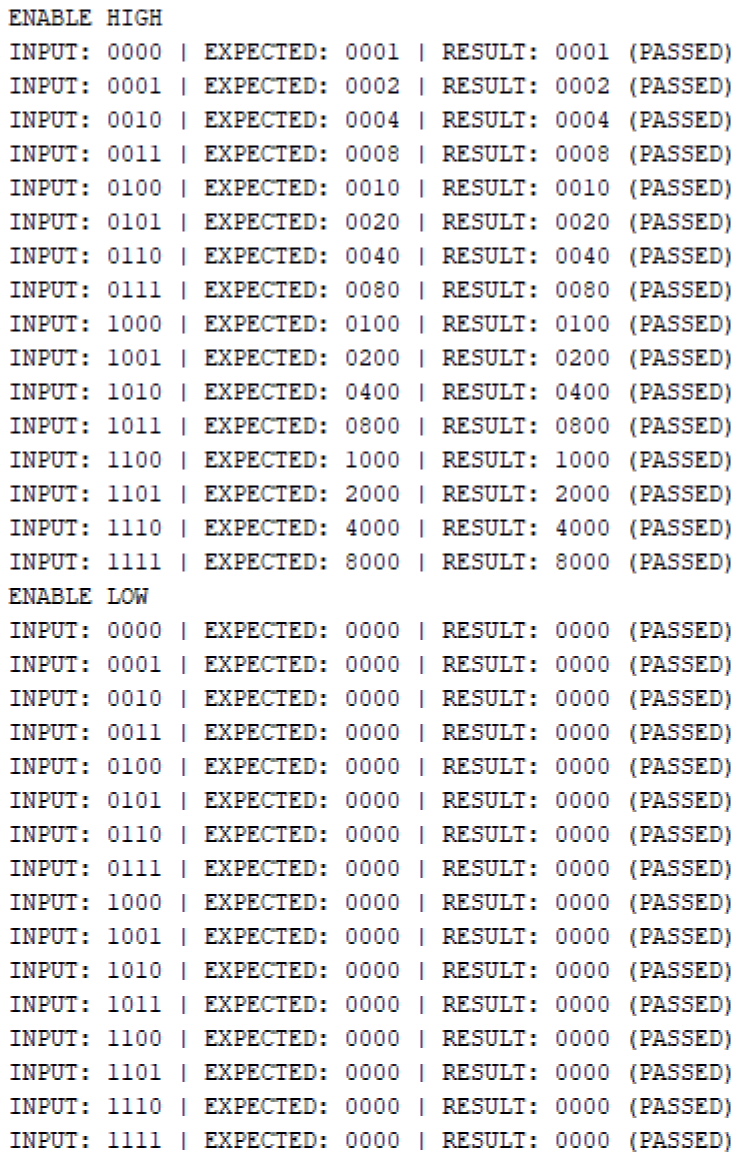


Name	Value
in[3:0]	f
[3]	1
[2]	1
[1]	1
[0]	1
enable	0
out[15:0]	0000
[15]	0
[14]	0
[13]	0
[12]	0
[11]	0
[10]	0
[9]	0
[8]	0
[7]	0
[6]	0
[5]	0
[4]	0
[3]	0
[2]	0
[1]	0



Name	Value
in[3:0]	0
[3]	0
[2]	0
[1]	0
[0]	0
enable	1
out[15:0]	0001
[15]	0
[14]	0
[13]	0
[12]	0
[11]	0
[10]	0
[9]	0
[8]	0
[7]	0
[6]	0
[5]	0
[4]	0
[3]	0
[2]	0
[1]	0

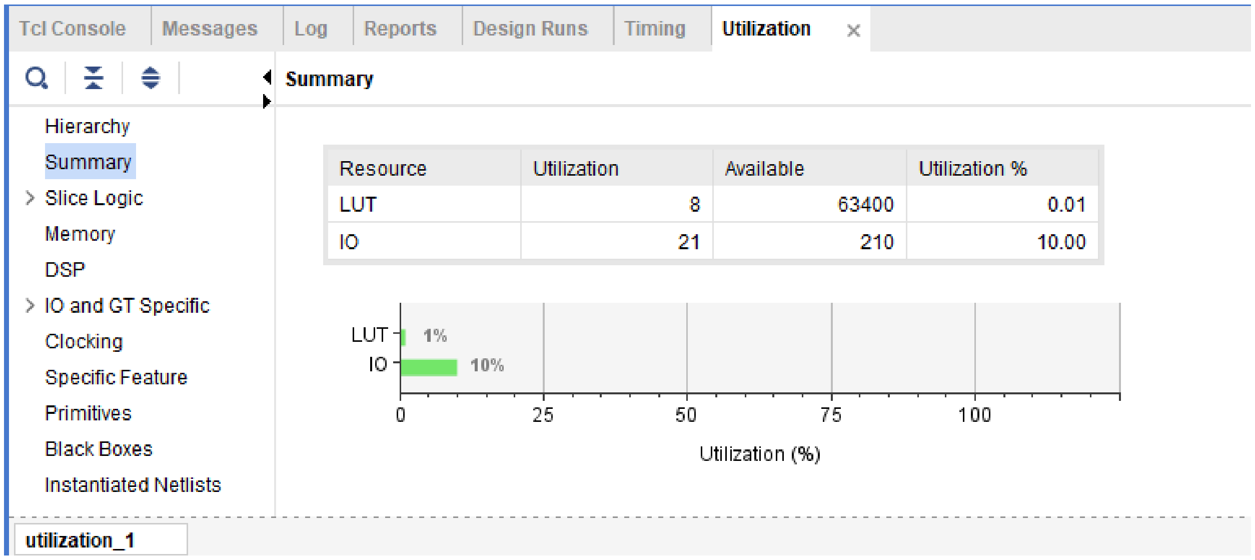
Simulation Log



```
ENABLE HIGH
INPUT: 0000 | EXPECTED: 0001 | RESULT: 0001 (PASSED)
INPUT: 0001 | EXPECTED: 0002 | RESULT: 0002 (PASSED)
INPUT: 0010 | EXPECTED: 0004 | RESULT: 0004 (PASSED)
INPUT: 0011 | EXPECTED: 0008 | RESULT: 0008 (PASSED)
INPUT: 0100 | EXPECTED: 0010 | RESULT: 0010 (PASSED)
INPUT: 0101 | EXPECTED: 0020 | RESULT: 0020 (PASSED)
INPUT: 0110 | EXPECTED: 0040 | RESULT: 0040 (PASSED)
INPUT: 0111 | EXPECTED: 0080 | RESULT: 0080 (PASSED)
INPUT: 1000 | EXPECTED: 0100 | RESULT: 0100 (PASSED)
INPUT: 1001 | EXPECTED: 0200 | RESULT: 0200 (PASSED)
INPUT: 1010 | EXPECTED: 0400 | RESULT: 0400 (PASSED)
INPUT: 1011 | EXPECTED: 0800 | RESULT: 0800 (PASSED)
INPUT: 1100 | EXPECTED: 1000 | RESULT: 1000 (PASSED)
INPUT: 1101 | EXPECTED: 2000 | RESULT: 2000 (PASSED)
INPUT: 1110 | EXPECTED: 4000 | RESULT: 4000 (PASSED)
INPUT: 1111 | EXPECTED: 8000 | RESULT: 8000 (PASSED)
ENABLE LOW
INPUT: 0000 | EXPECTED: 0000 | RESULT: 0000 (PASSED)
INPUT: 0001 | EXPECTED: 0000 | RESULT: 0000 (PASSED)
INPUT: 0010 | EXPECTED: 0000 | RESULT: 0000 (PASSED)
INPUT: 0011 | EXPECTED: 0000 | RESULT: 0000 (PASSED)
INPUT: 0100 | EXPECTED: 0000 | RESULT: 0000 (PASSED)
INPUT: 0101 | EXPECTED: 0000 | RESULT: 0000 (PASSED)
INPUT: 0110 | EXPECTED: 0000 | RESULT: 0000 (PASSED)
INPUT: 0111 | EXPECTED: 0000 | RESULT: 0000 (PASSED)
INPUT: 1000 | EXPECTED: 0000 | RESULT: 0000 (PASSED)
INPUT: 1001 | EXPECTED: 0000 | RESULT: 0000 (PASSED)
INPUT: 1010 | EXPECTED: 0000 | RESULT: 0000 (PASSED)
INPUT: 1011 | EXPECTED: 0000 | RESULT: 0000 (PASSED)
INPUT: 1100 | EXPECTED: 0000 | RESULT: 0000 (PASSED)
INPUT: 1101 | EXPECTED: 0000 | RESULT: 0000 (PASSED)
INPUT: 1110 | EXPECTED: 0000 | RESULT: 0000 (PASSED)
INPUT: 1111 | EXPECTED: 0000 | RESULT: 0000 (PASSED)
```

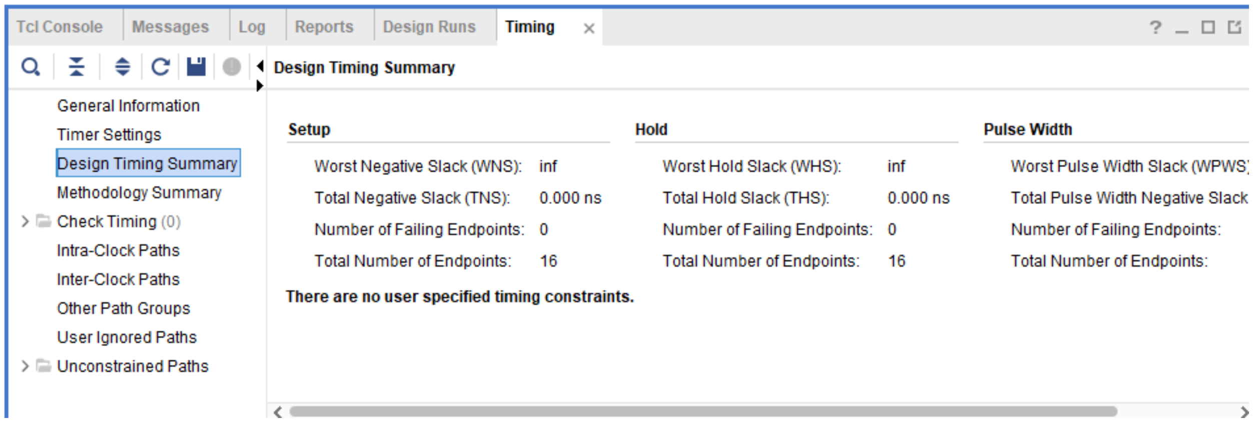
Implementation

Utilization Table:



Resource	Utilization	Available	Utilization %
LUT	8	63400	0.01
IO	21	210	10.00

Timing Summary:



Setup

Worst Negative Slack (WNS): inf
Total Negative Slack (TNS): 0.000 ns
Number of Failing Endpoints: 0
Total Number of Endpoints: 16

Hold

Worst Hold Slack (WHS): inf
Total Hold Slack (THS): 0.000 ns
Number of Failing Endpoints: 0
Total Number of Endpoints: 16

Pulse Width

Worst Pulse Width Slack (WPWS)
Total Pulse Width Negative Slack
Number of Failing Endpoints:
Total Number of Endpoints:

There are no user specified timing constraints.

Contribution

Michelle Lau (50%) - Implementation and board demo
Edwin Estrada (50%) - Verilog programming and test benching

Board Demo

https://youtu.be/3gXKFNydQbw