ECE 3300L

Lab Report #4

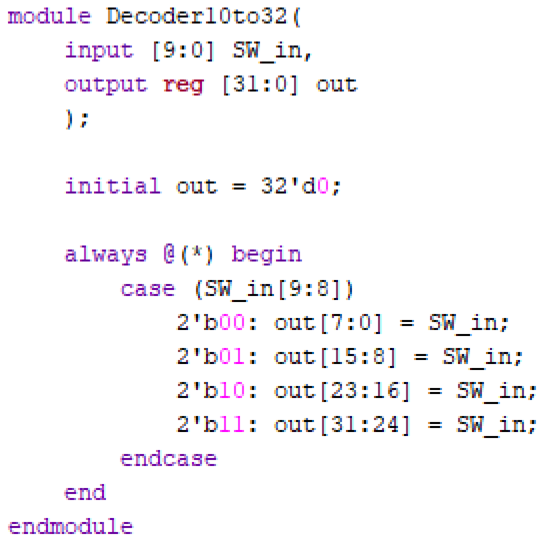
Group A

Edwin Estrada (#015897050)

Michelle Lau (#016027427)

July 11, 2025

Design
10 to 32 Decoder

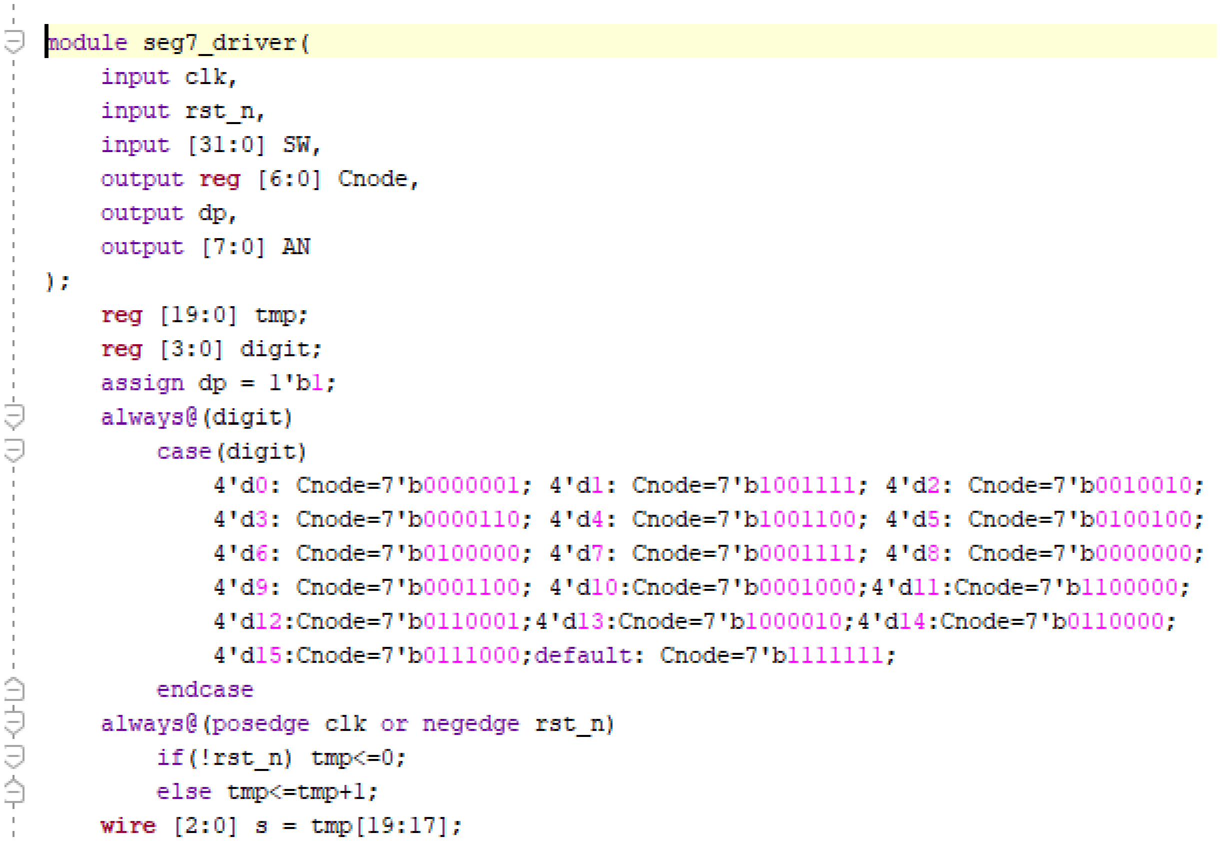


```
module Decoder10to32(
    input [9:0] SW_in,
    output reg [31:0] out
    );

    initial out = 32'd0;

    always @(*) begin
        case (SW_in[9:8])
            2'b00: out[7:0] = SW_in;
            2'b01: out[15:8] = SW_in;
            2'b10: out[23:16] = SW_in;
            2'b11: out[31:24] = SW_in;
        endcase
    end
endmodule
```

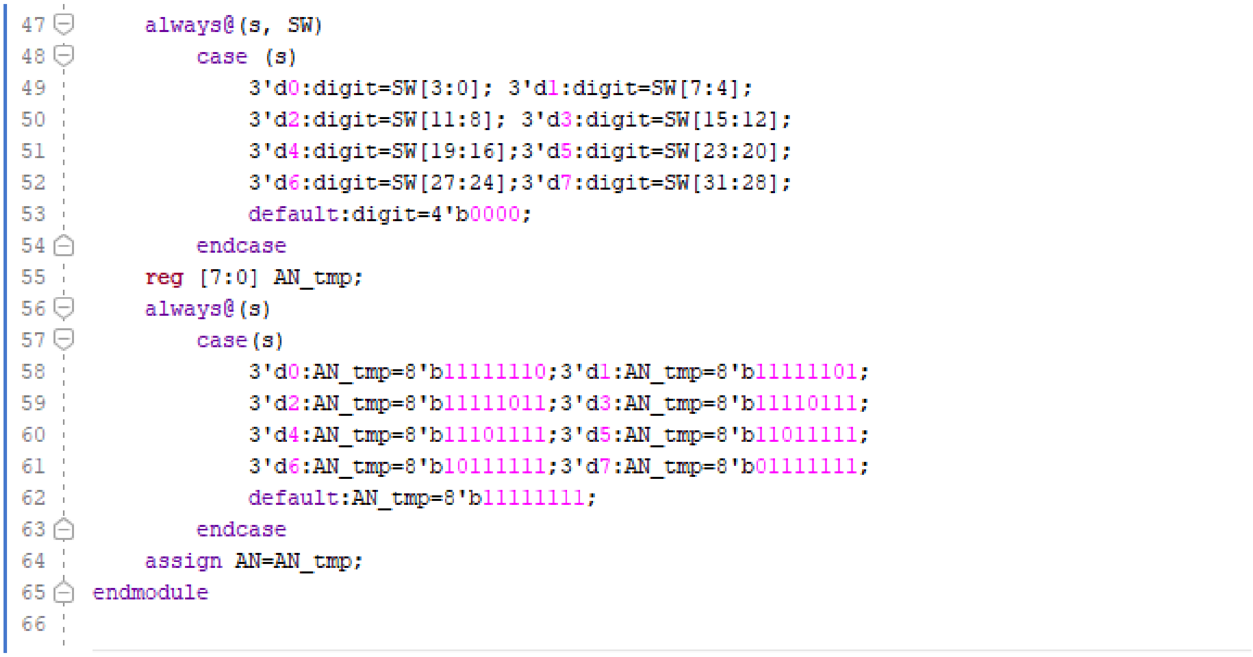
Controlling the 8-digit 7-seg display involves using a 32-bit input bus. To control the 32-input bus, we use 10 of the 16 switches provided.

The MSB being the reset.
The next 2 of them are used to determine which 8-bit part of the 32-bit bus (in other words, which set of two digits on the display) to select.
The other 8 switches set the value of that 8-bit part of the 32-bit bus.

7-Segment Display Driver



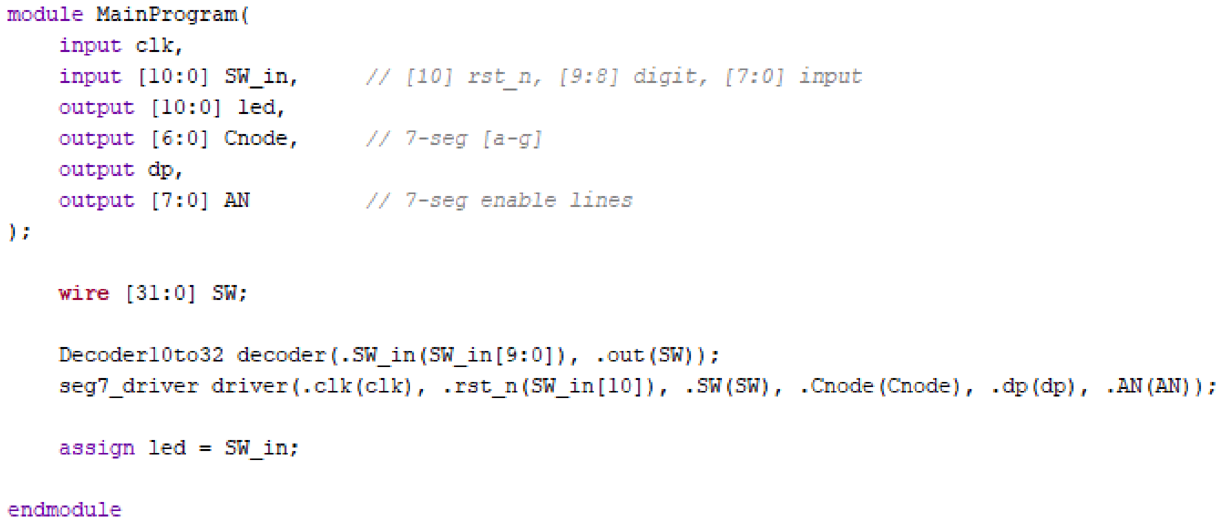
```
module seg7_driver(
    input clk,
    input rst_n,
    input [31:0] SW,
    output reg [6:0] Cnode,
    output dp,
    output [7:0] AN
);
    reg [19:0] tmp;
    reg [3:0] digit;
    assign dp = 1'b1;
    always@(digit)
        case(digit)
            4'd0: Cnode=7'b0000001; 4'd1: Cnode=7'b1001111; 4'd2: Cnode=7'b0010010;
            4'd3: Cnode=7'b0000110; 4'd4: Cnode=7'b1001100; 4'd5: Cnode=7'b0100100;
            4'd6: Cnode=7'b0100000; 4'd7: Cnode=7'b0001111; 4'd8: Cnode=7'b0000000;
            4'd9: Cnode=7'b0001100; 4'd10:Cnode=7'b0001000;4'd11:Cnode=7'b1100000;
            4'd12:Cnode=7'b0110001;4'd13:Cnode=7'b1000010;4'd14:Cnode=7'b0110000;
            4'd15:Cnode=7'b0111000;default: Cnode=7'b1111111;
        endcase
    always@(posedge clk or negedge rst_n)
        if(!rst_n) tmp<=0;
        else tmp<=tmp+1;
    wire [2:0] s = tmp[19:17];
```



```
    always@(s, SW)
        case (s)
            3'd0:digit=SW[3:0]; 3'd1:digit=SW[7:4];
            3'd2:digit=SW[11:8]; 3'd3:digit=SW[15:12];
            3'd4:digit=SW[19:16];3'd5:digit=SW[23:20];
            3'd6:digit=SW[27:24];3'd7:digit=SW[31:28];
            default:digit=4'b0000;
        endcase
    reg [7:0] AN_tmp;
    always@(s)
        case(s)
            3'd0:AN_tmp=8'b11111110;3'd1:AN_tmp=8'b11111101;
            3'd2:AN_tmp=8'b11111011;3'd3:AN_tmp=8'b11110111;
            3'd4:AN_tmp=8'b11101111;3'd5:AN_tmp=8'b11011111;
            3'd6:AN_tmp=8'b10111111;3'd7:AN_tmp=8'b01111111;
            default:AN_tmp=8'b11111111;
        endcase
    assign AN=AN_tmp;
endmodule
```

Given a 100Mhz base clock on the board, the board counts to 1 million and resets the counter, changing each anode of the 7-seg display to 0 one at a time throughout the time the board is counting to 1 million. With the 100Mhz clock, that means the 7-seg display is blinking 100 times per second (100hz), giving the illusion to the user that the seven seg display is on. Then, what 4-bit part of the 32-bit input to select depends on the current count of the 1 million, depending on the 3 highest bits of the counter. This makes sure each digit on the 7-seg display represents a unique value. Additionally, anytime digit is changed from the counter reaching certain values, a digit on the 7-seg display will display its respective value given by the 4-bit part of the 32-bit input.

High Level Implementation



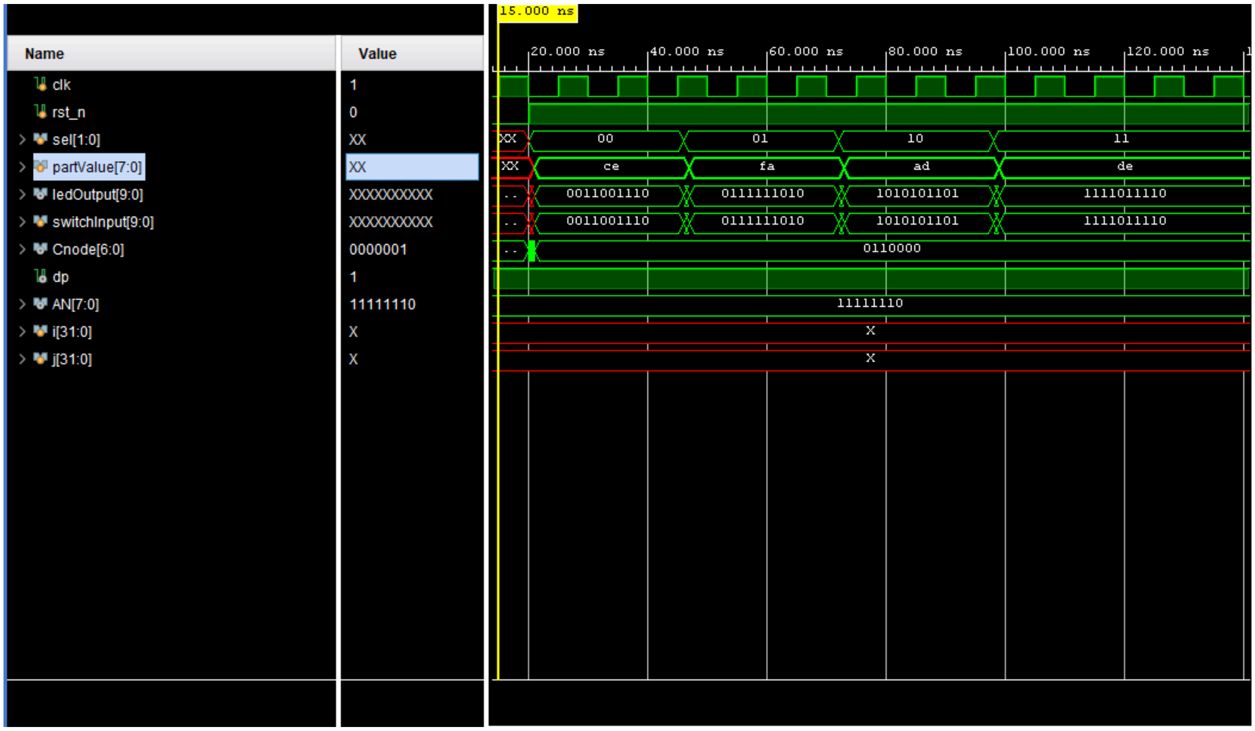
```
module MainProgram(
    input clk,
    input [10:0] SW_in,      // [10] rst_n, [9:8] digit, [7:0] input
    output [10:0] led,
    output [6:0] Cnode,      // 7-seg [a-g]
    output dp,
    output [7:0] AN          // 7-seg enable lines
);

    wire [31:0] SW;

    Decoder10to32 decoder(.SW_in(SW_in[9:0]), .out(SW));
    seg7_driver driver(.clk(clk), .rst_n(SW_in[10]), .SW(SW), .Cnode(Cnode), .dp(dp), .AN(AN));

    assign led = SW_in;

endmodule
```

Connecting everything together, the switches go into the decoder, which is output into a declared wire. The output from the wire goes into the driver, with Cnode, dp, and AN going into the 7-seg display. We also map the switches to their respective LED to show on the board that a switch is on.

Simulation Waveform

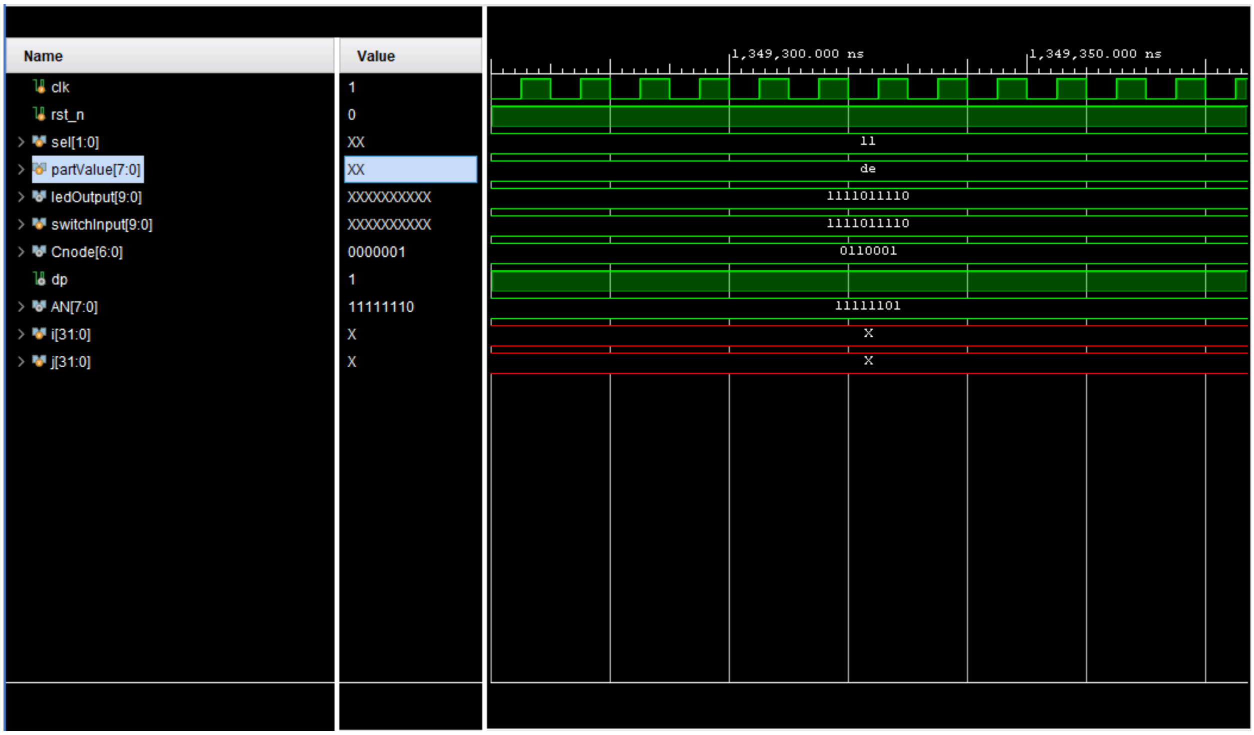
This simulation displays “DEADFACE” on the 8-digit 7-seg display.

We begin by toggling reset. We then use the 10 switches and the 10 to 32 decoder to give SW a value of 0xDEADFACE. AN selects the rightmost digit to turn on, with Cnode having a value of “0110000” (the letter E).



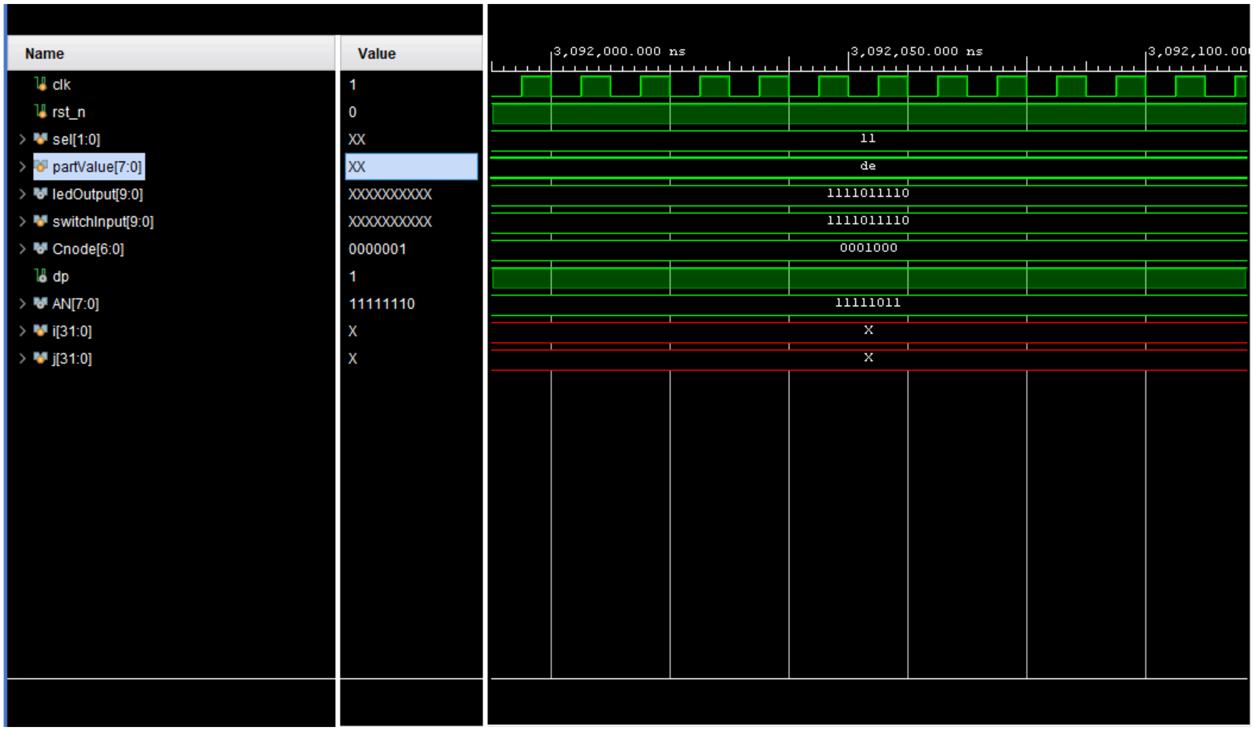
Name	Value
clk	1
rst_n	0
sel[1:0]	XX
partValue[7:0]	XX
ledOutput[9:0]	XXXXXXXXXX
switchInput[9:0]	XXXXXXXXXX
Cnode[6:0]	0000001
dp	1
AN[7:0]	11111110
i[31:0]	X
j[31:0]	X

AN now selects the second rightmost bit, with Cnode having a value of “0110001” (the second rightmost digit on the display shows the letter C).



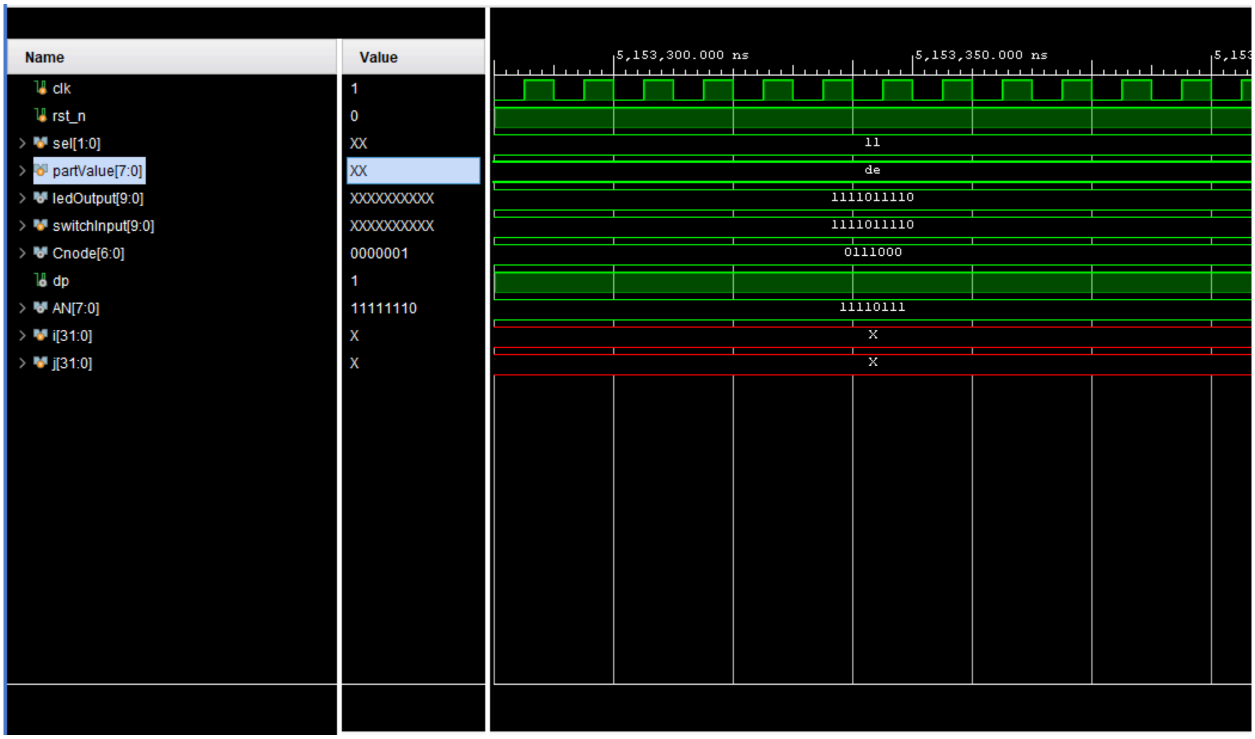
Name	Value
clk	1
rst_n	0
sel[1:0]	XX
partValue[7:0]	XX
ledOutput[9:0]	XXXXXXXXXX
switchInput[9:0]	XXXXXXXXXX
Cnode[6:0]	0000001
dp	1
AN[7:0]	11111110
i[31:0]	X
j[31:0]	X

AN now selects the third rightmost bit, with Cnode having a value of “0001000” (the third rightmost digit on the display shows the letter A).



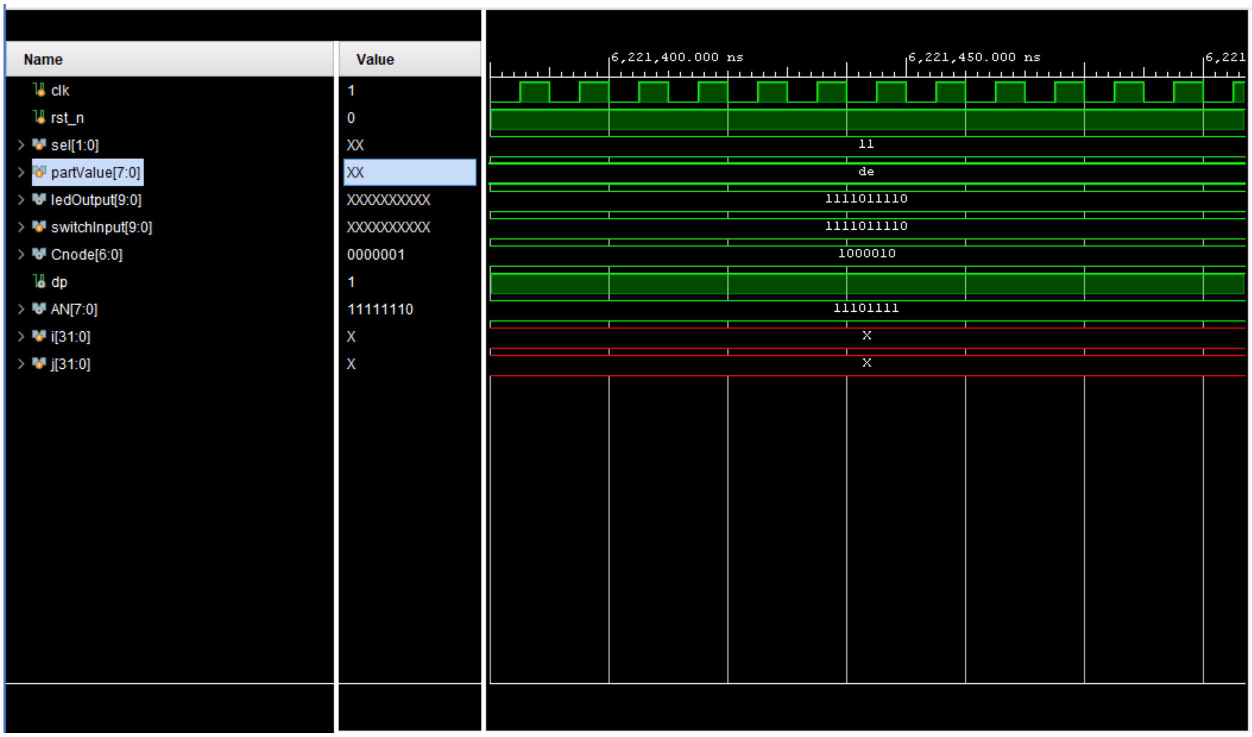
Name	Value
clk	1
rst_n	0
sel[1:0]	XX
partValue[7:0]	XX
ledOutput[9:0]	XXXXXXXXXX
switchInput[9:0]	XXXXXXXXXX
Cnode[6:0]	0000001
dp	1
AN[7:0]	11111110
i[31:0]	X
j[31:0]	X

AN now selects the fourth rightmost bit, with Cnode having a value of “0111000” (the fourth rightmost digit on the display shows the letter F).



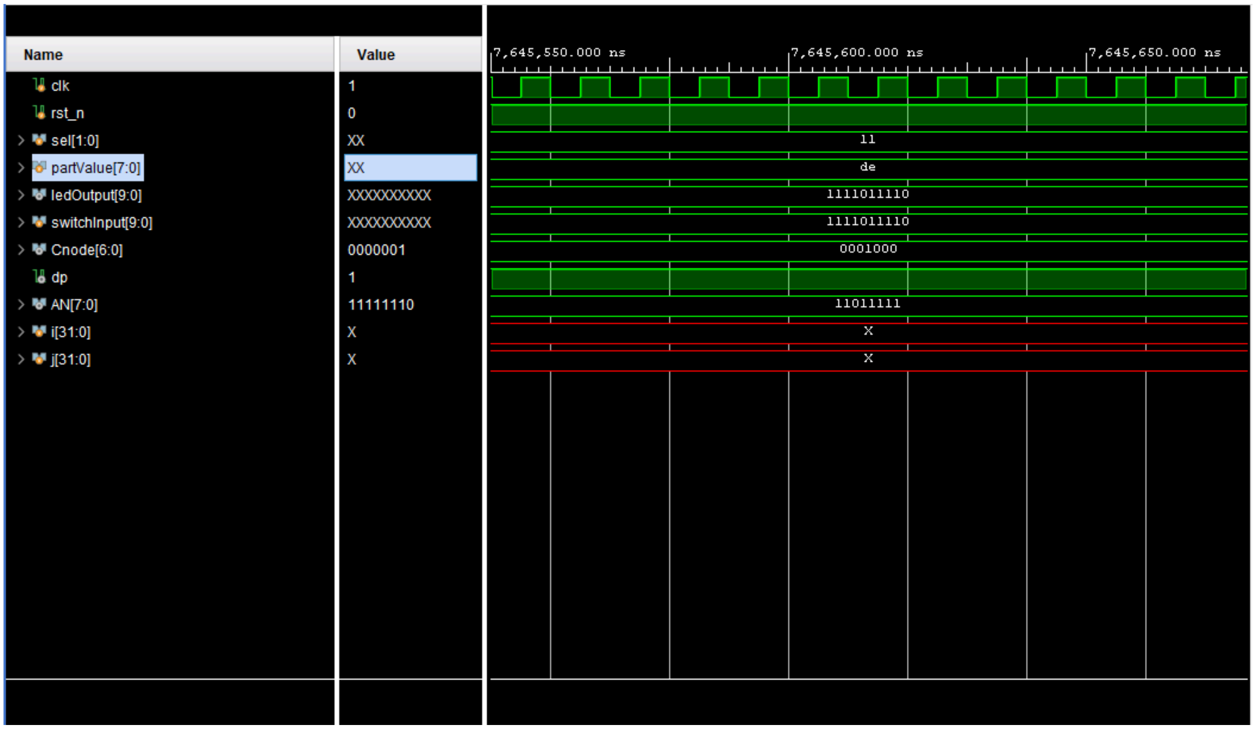
Name	Value
clk	1
rst_n	0
sel[1:0]	XX
partValue[7:0]	XX
ledOutput[9:0]	XXXXXXXXXX
switchInput[9:0]	XXXXXXXXXX
Cnode[6:0]	0000001
dp	1
AN[7:0]	11111110
i[31:0]	X
j[31:0]	X

AN now selects the fourth leftmost bit, with Cnode having a value of “1000010” (the fourth leftmost digit on the display shows the letter D).



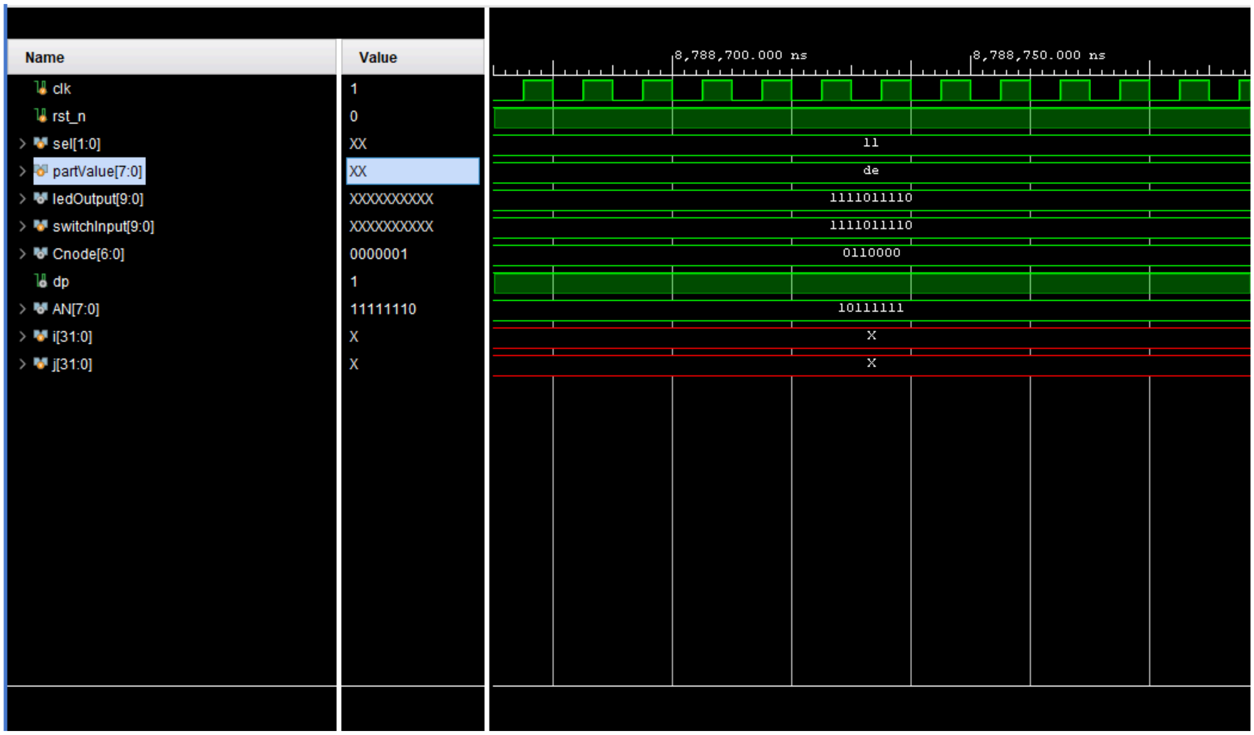
Name	Value
clk	1
rst_n	0
sel[1:0]	XX
partValue[7:0]	XX
ledOutput[9:0]	XXXXXXXXXX
switchInput[9:0]	XXXXXXXXXX
Cnode[6:0]	0000001
dp	1
AN[7:0]	11111110
i[31:0]	X
j[31:0]	X

AN now selects the third leftmost bit, with Cnode having a value of “0001000” (the third leftmost digit on the display shows the letter A).



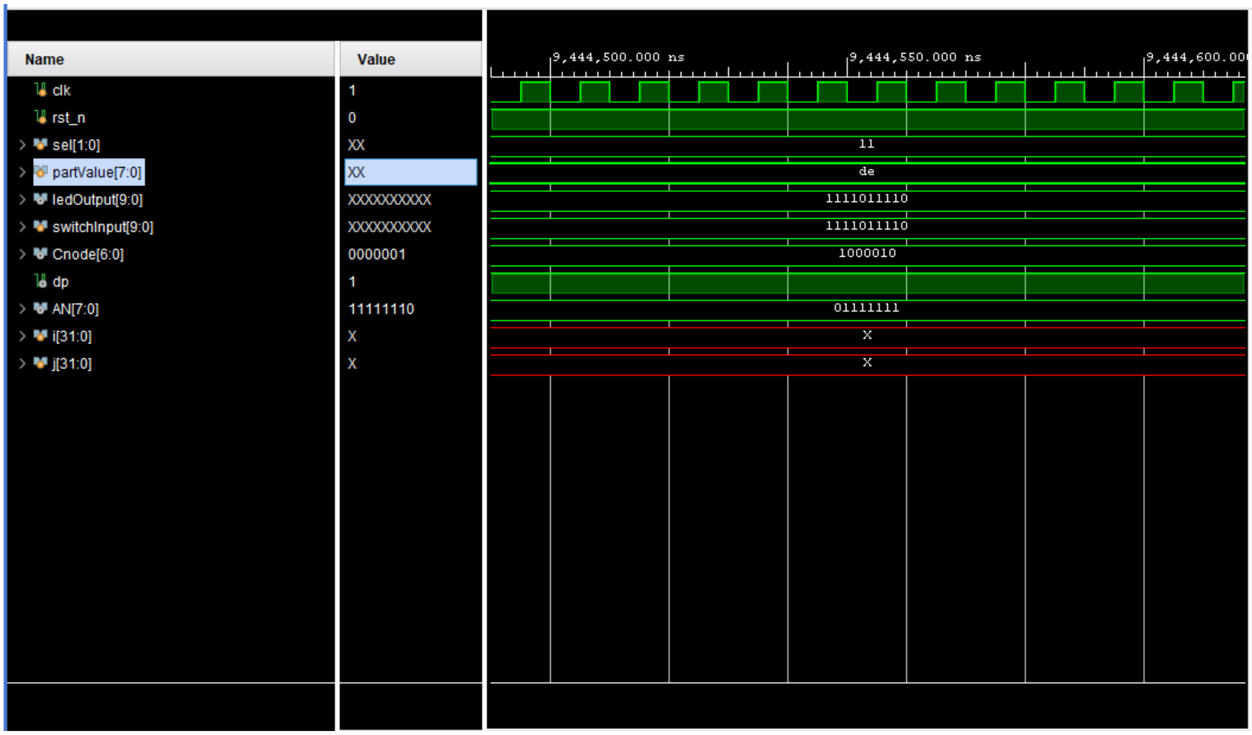
Name	Value	
clk	1	
rst_n	0	
sel[1:0]	XX	11
partValue[7:0]	XX	de
ledOutput[9:0]	XXXXXXXXXX	1111011110
switchInput[9:0]	XXXXXXXXXX	1111011110
Cnode[6:0]	0000001	0001000
dp	1	
AN[7:0]	11111110	11011111
i[31:0]	X	X
j[31:0]	X	X

AN now selects the second leftmost bit, with Cnode having a value of “0110000” (the second leftmost digit on the display shows the letter E).



Name	Value	
clk	1	
rst_n	0	
sel[1:0]	XX	11
partValue[7:0]	XX	de
ledOutput[9:0]	XXXXXXXXXX	1111011110
switchInput[9:0]	XXXXXXXXXX	1111011110
Cnode[6:0]	0000001	0110000
dp	1	
AN[7:0]	11111110	10111111
i[31:0]	X	X
j[31:0]	X	X

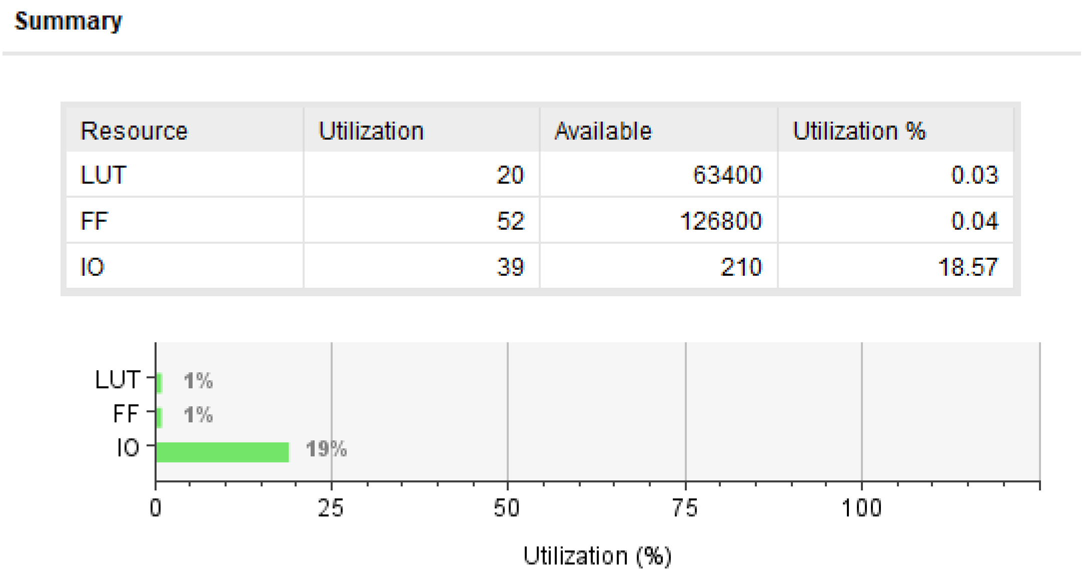
AN now selects the leftmost bit, with Cnode having a value of “1000010” (the leftmost digit on the display shows the letter D).



Name	Value
clk	1
rst_n	0
sel[1:0]	XX
partValue[7:0]	XX
ledOutput[9:0]	XXXXXXXXXX
switchInput[9:0]	XXXXXXXXXX
Cnode[6:0]	0000001
dp	1
AN[7:0]	11111110
i[31:0]	X
j[31:0]	X

Implementation

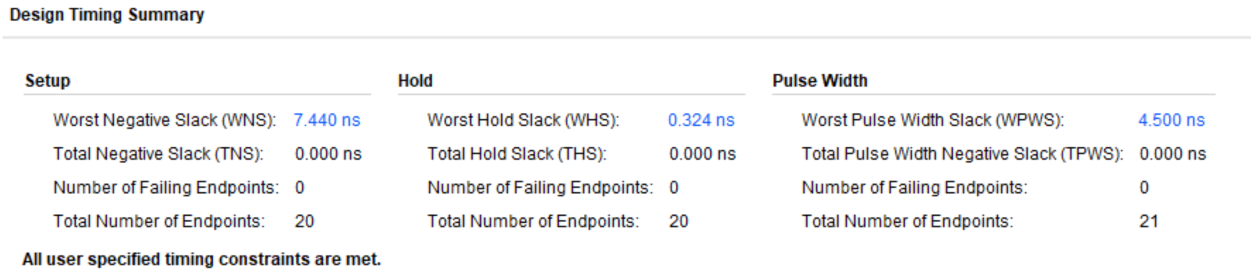
Utilization Table:



Summary

Resource	Utilization	Available	Utilization %
LUT	20	63400	0.03
FF	52	126800	0.04
IO	39	210	18.57

Timing Summary:



Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	7.440 ns	Worst Hold Slack (WHS):	0.324 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:	20	Total Number of Endpoints:	20	Total Number of Endpoints:	21

All user specified timing constraints are met.

Contribution

Michelle Lau (50%) - Debugging, implementation and board demo
Edwin Estrada (50%) - Programming, debugging, and test benching

Board Demo

< Demo.MOV >