# ELC 2137 Lab 6: 7 Segment Display

Forrest Knee

March 9, 2021

## Summary

This lab involved programing a mux, a binary to seven segment display decoder, and a file to merge the mux and decoder. This file programs the basys3 board to convert the switches positions in binary to hexidecmial.

## Q&A

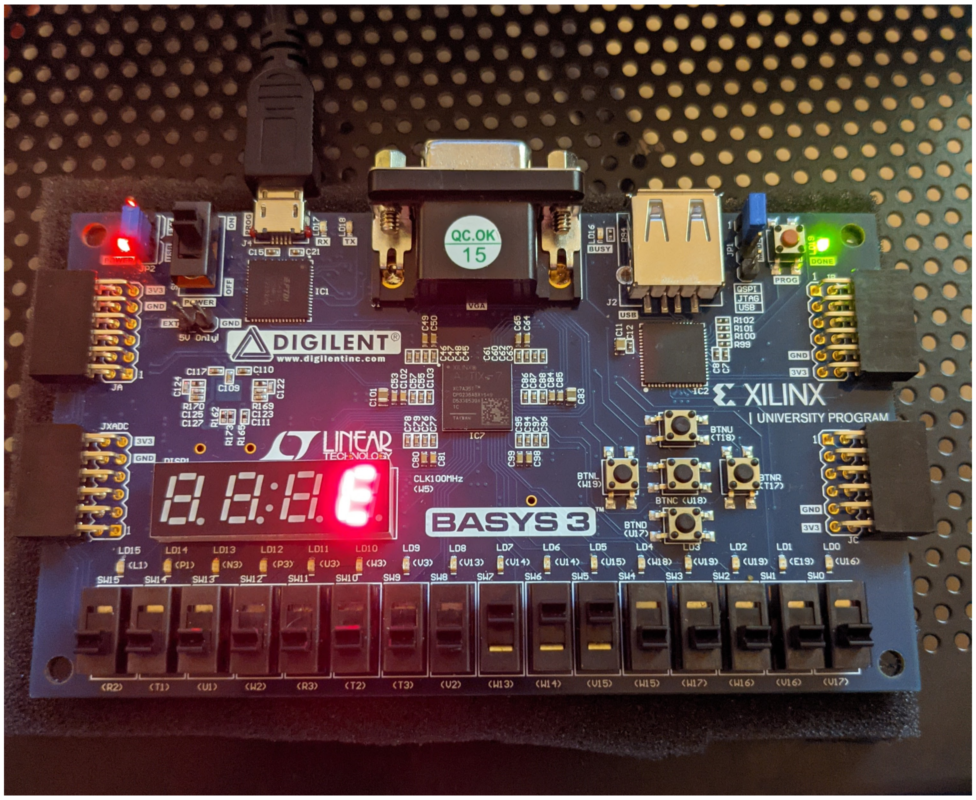
3. List the errors you found during simulation. What does this tell you about why we run simulations?

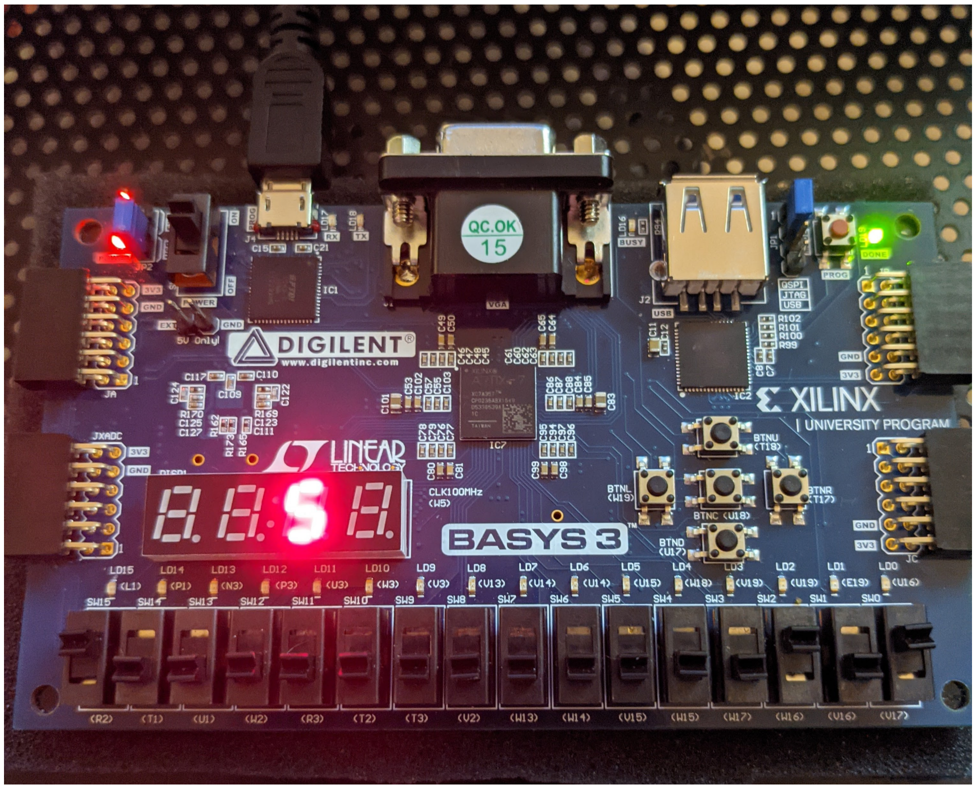
I had made an error in my sseg decoder where my ”F” and ”E” had switched codes which I found through simulation. This shows me how simulations tell of minor errors in code that can be easily identified through testing in an ideal environment.

4. How many wires are connected to the 7-segment display? If the segments were not all connected together, how many wires would there have to be? Why do we prefer the current method vs. separating all of the segments?

There are 12 wires connected to the seven segment display. If the segments were not all connected together, there would be 72 wires connecting each positive and negative end of each display. The current method of using a common anode and multiplexer simplifies the construction of the display without sacraficing functionality.

## Results





## Code

Listing 1: Mux Source Code

```
‘timescale 1ns / 1ps

//Forrest Knee


module mux2_4b(
    input [3:0] in0,
    input [3:0] in1,
    input [1:0] sel,
    output [3:0] out
    );

    assign out = sel ? in0 : in1;

endmodule
```

Listing 2: Seven Segment Decoder

```
‘timescale 1ns / 1ps

//Forrest Knee Lab 6

module sseg_decoder(
    input [3:0] num,
    output  reg[6:0] sseg
    );

    always @*
        case(num)
            4'h0: sseg =7'b1000000;
            4'h1: sseg =7'b1111001;
            4'h2: sseg =7'b0100100;
            4'h3: sseg =7'b0110000;
            4'h4: sseg =7'b0011001;
            4'h5: sseg =7'b0010010;
            4'h6: sseg =7'b0000010;
            4'h7: sseg =7'b1111000;
            4'h8: sseg =7'b0000000;
            4'h9: sseg =7'b0010000;
            4'hA: sseg= 7'b0001000;
            4'hB: sseg= 7'b0000011;
            4'hC: sseg= 7'b1000110;
            4'hD: sseg= 7'b0100001;
            4'hE: sseg= 7'b0000110;
            4'hF: sseg= 7'b0001110;
        endcase


endmodule
```

Listing 3: Seven Segment and Mux Merger

```
‘timescale 1ns / 1ps



module sseg1(
    input [15:0] sw, //switches
    output [3:0] an, //7-seg digits
    output [6:0] seg, //7-seg segments
    output dp // decimal point
    );

    wire [3:0] muxout;

    mux2_4b mux2(

    .in0(sw[3:0]), .in1(sw[7:4]), .sel(sw[15]), .out(muxout)
    );

    sseg_decoder decode(
    .num(muxout), .sseg(seg)
    );

    assign an[1] = ~sw[15];
    assign an[0] = sw[15];
    assign an[3]=1;
    assign an[2]=1;
    assign dp=1;

endmodule
```

Listing 4: Mux Simulation

```
‘timescale 1ns / 1ps

//Forrest Knee Lab 6

module mux_SIM();

    reg [3:0] tin0;
    reg [3:0] tin1;
    reg [1:0] tsel;
    wire [3:0] tout;

    mux2_4b mux1(

    .in0(tin0), .in1(tin1), .sel(tsel), .out(tout));

    initial begin

    //Case 1

    tin0[0] = 0;
    tin0[1] = 0;
    tin0[2] = 0;
    tin0[3] = 0;
    tin1[0] = 1;
    tin1[1] = 0;
    tin1[2] = 0;
    tin1[3] = 0;
    tsel = 0;
    #10

    //Case 2

    tin0[0] = 0;
    tin0[1] = 1;
    tin0[2] = 0;
    tin0[3] = 0;
    tin1[0] = 1;
    tin1[1] = 1;
    tin1[2] = 0;
    tin1[3] = 0;
    tsel = 1;
    #10

    //Case 3

    tin0[0] = 0;
    tin0[1] = 0;
    tin0[2] = 1;
    tin0[3] = 0;
    tin1[0] = 1;
    tin1[1] = 0;
    tin1[2] = 1;
    tin1[3] = 0;
    tsel = 0;
    #10

    //Case 4

    tin0[0] = 0;
    tin0[1] = 1;
    tin0[2] = 1;
    tin0[3] = 0;
    tin1[0] = 1;
    tin1[1] = 1;
    tin1[2] = 1;
    tin1[3] = 0;
    tsel = 1;
    #10
    $finish;
    end

endmodule
```

Listing 5: Seven Segment Decoder Simulation

```
‘timescale 1ns / 1ps

module sseg_SIM();

reg [3:0] tnum;
wire [6:0] tsseg;

sseg_decoder s_g(

    .num(tnum), .sseg(tsseg));

  integer i;

initial begin

    for(i=0; i<=8’hf; i=i+1) begin
    tnum = i;
    #10;
    end
    $finish;
    end


endmodule
```

Listing 6: Merge Code Simulation

```
‘timescale 1ns / 1ps

//Forrest Knee lab 6

module sseg1_SIM();

    reg [15:0] sw;
    wire [3:0] an;
    wire [6:0] seg;
    wire [1:0] dp;

    sseg1 name1(
    .sw(sw), .an(an), .seg(seg), .dp(dp));

    initial begin

    sw= 16’h0000;
    sw[14:8]= 0;
    // case 1

     sw[7:0] = 8’hA;
     sw[15]= 1’b0;
     #10
     sw[15] = 1’b1;
     #10

    // case 2
```

```
     sw[7:0] = 8'hB;
     sw[15]= 1'b0;
     #10
     sw[15] = 1'b1;
     #10


     // case 3

     sw[7:0] = 8'hC;
     sw[15]= 1'b0;
     #10
     sw[15] = 1'b1;
     #10

     // case 4

     sw[7:0] = 8'hD;
     sw[15]= 1'b0;
     #10
     sw[15] = 1'b1;
     #10
    $finish;
    end


endmodule
```