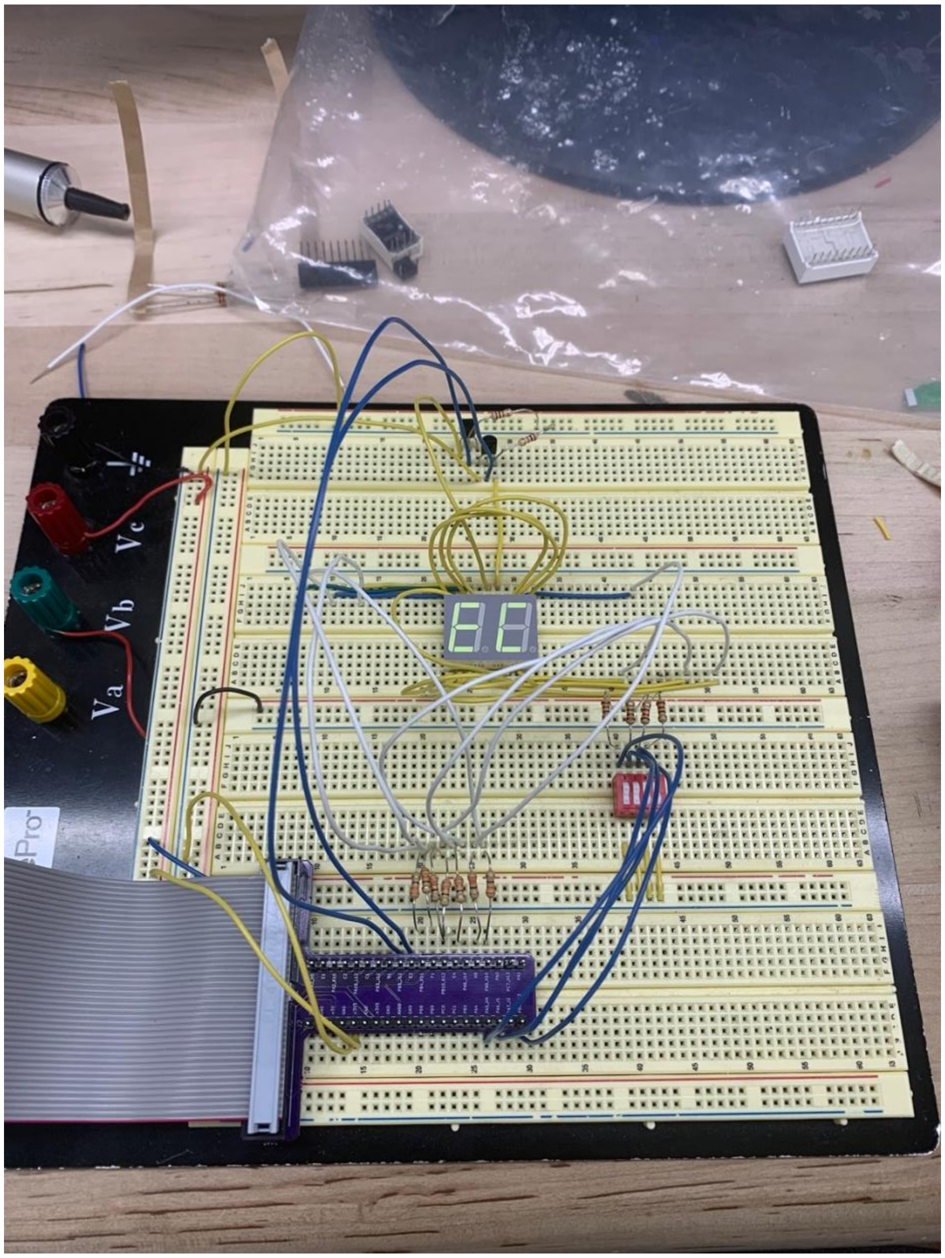
Multiplexed 7-Segment Display

Engineering 155 Lab II Report
September 15th, 2021

John Hearn



Introduction:

The goal of this lab was to implement a time-multiplexing scheme to drive two 7-segment displays with a single set of FPGA I/O pins. We did so by building a simple transistor circuit to amplify the current outputted by the board.

Design Methodology:

In designing the hardware for the system, the main point of concern was using the transistors to power both displays and making sure that the resistance in the collector was high enough.

$$\frac{3.3V}{2k\Omega} = 1.65mA$$

which is in the allowable current range for both transistors and DIP switch!

Fig. 1: Resistor calculation

In designing the software, three SystemVerilog modules were created in Quartus. One main module where the calculations for LEDs and which display to power are done, one module to calculate the timing for which display to power (to facilitate the time-multiplexing), and one module to assign the segments high and low.

The system took in three inputs:

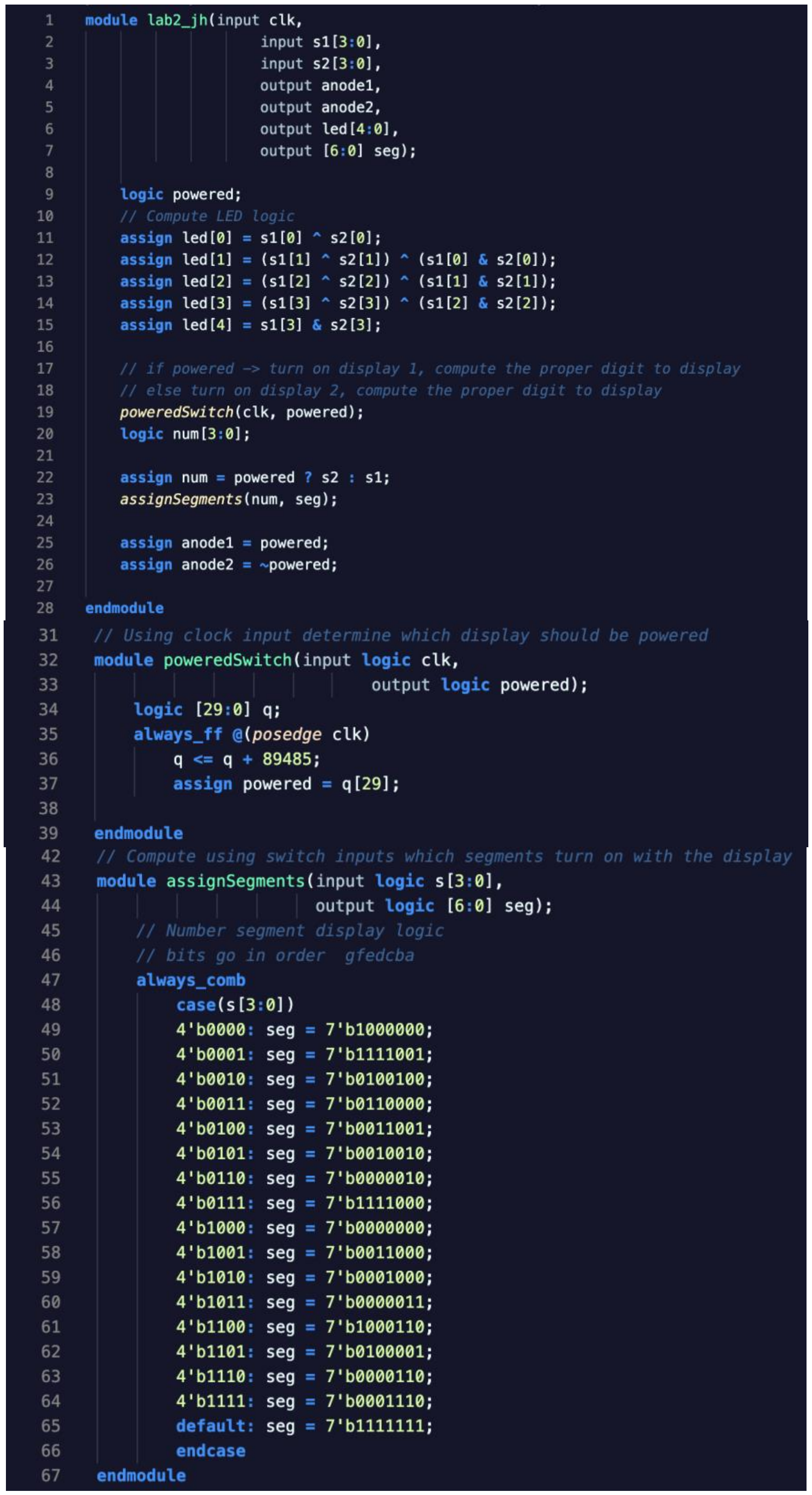
- **clk** (board's internal 12MHz clock),
- **s1** (board-mounted DIP switches),
- **s2** (breadboard-mounted DIP switches)

and had four outputs:

- **anode1** (logic to power display 1),
- **anode2** (logic to power display 2),
- **led** (LED outputs),
- **seg** (indicates which segments to power)

It was decided that the displays should run at 1000Hz and using a divide-by-2^N counter this feature was made possible. Using the equation $f_{out} = f_{clk} * (p / 2^N)$, it was found that the combination of p = 89485, N = 30 gives an accurate approximation of 1000Hz.

Technical Documentation:



```
module lab2_jh(input clk,
               input s1[3:0],
               input s2[3:0],
               output anode1,
               output anode2,
               output led[4:0],
               output [6:0] seg);

    logic powered;
    // Compute LED logic
    assign led[0] = s1[0] ^ s2[0];
    assign led[1] = (s1[1] ^ s2[1]) ^ (s1[0] & s2[0]);
    assign led[2] = (s1[2] ^ s2[2]) ^ (s1[1] & s2[1]);
    assign led[3] = (s1[3] ^ s2[3]) ^ (s1[2] & s2[2]);
    assign led[4] = s1[3] & s2[3];

    // if powered -> turn on display 1, compute the proper digit to display
    // else turn on display 2, compute the proper digit to display
    poweredSwitch(clk, powered);
    logic num[3:0];

    assign num = powered ? s2 : s1;
    assignSegments(num, seg);

    assign anode1 = powered;
    assign anode2 = ~powered;

endmodule
// Using clock input determine which display should be powered
module poweredSwitch(input logic clk,
                     output logic powered);
    logic [29:0] q;
    always_ff @(posedge clk)
        q <= q + 89485;
        assign powered = q[29];

endmodule
// Compute using switch inputs which segments turn on with the display
module assignSegments(input logic s[3:0],
                      output logic [6:0] seg);
    // Number segment display logic
    // bits go in order  gfedcba
    always_comb
        case(s[3:0])
        4'b0000: seg = 7'b1000000;
        4'b0001: seg = 7'b1111001;
        4'b0010: seg = 7'b0100100;
        4'b0011: seg = 7'b0110000;
        4'b0100: seg = 7'b0011001;
        4'b0101: seg = 7'b0010010;
        4'b0110: seg = 7'b0000010;
        4'b0111: seg = 7'b1111000;
        4'b1000: seg = 7'b0000000;
        4'b1001: seg = 7'b0011000;
        4'b1010: seg = 7'b0001000;
        4'b1011: seg = 7'b0000011;
        4'b1100: seg = 7'b1000110;
        4'b1101: seg = 7'b0100001;
        4'b1110: seg = 7'b0000110;
        4'b1111: seg = 7'b0001110;
        default: seg = 7'b1111111;
        endcase
endmodule
```

Fig. 2: SystemVerilog code

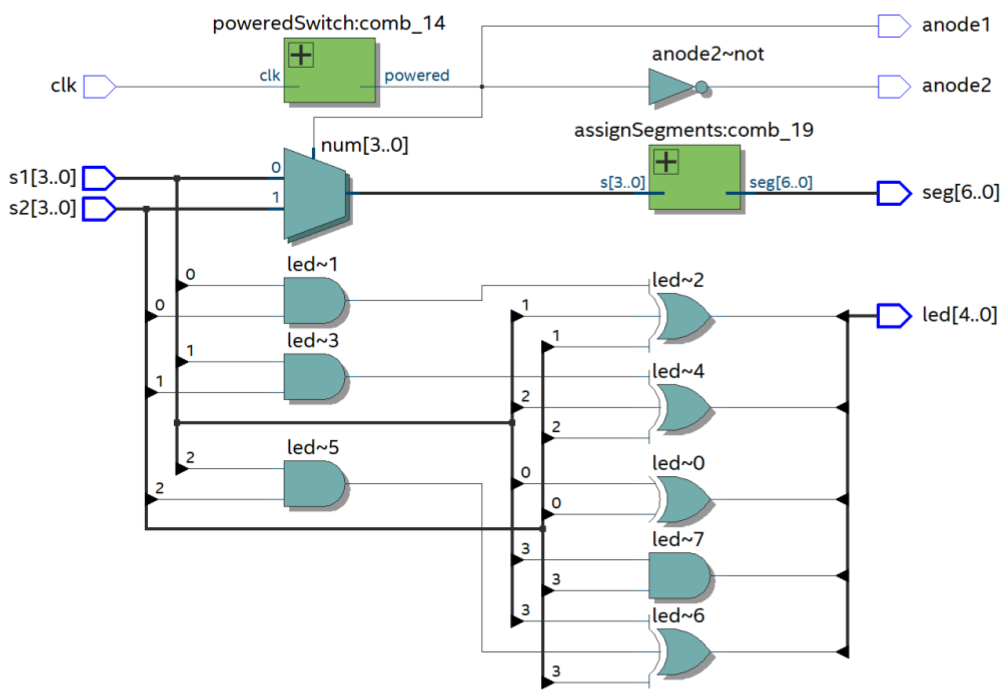
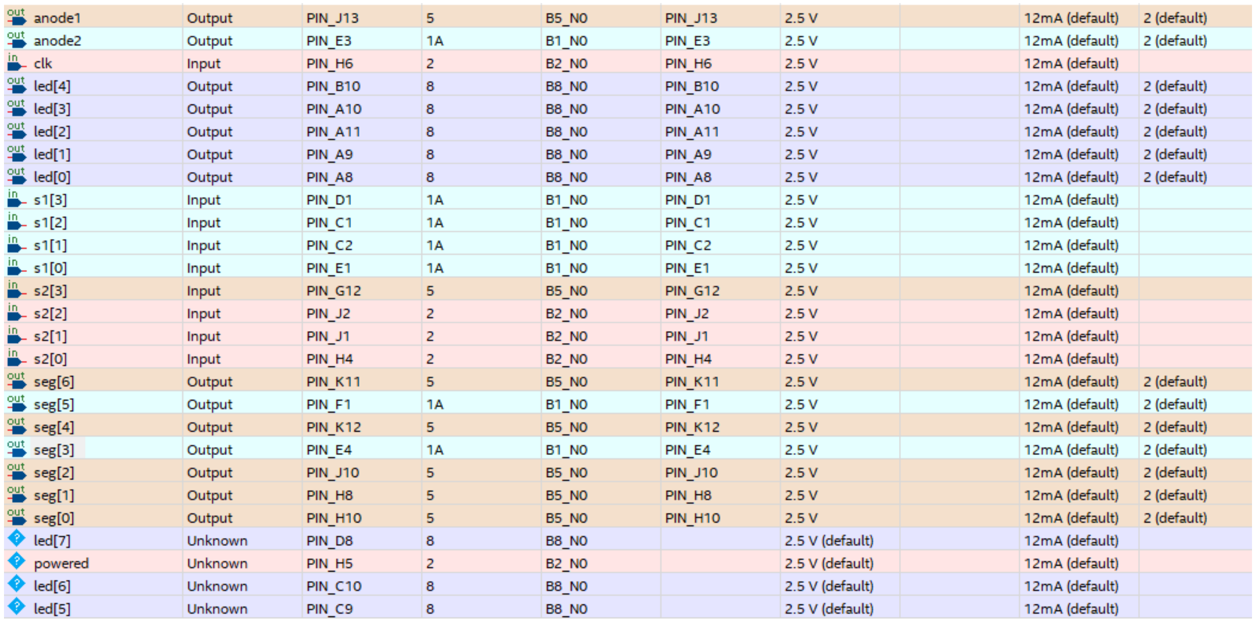


Fig. 3: RTL schematic showing logic elements



anode1	Output	PIN_J13	5	B5_N0	PIN_J13	2.5 V		12mA (default)	2 (default)
anode2	Output	PIN_E3	1A	B1_N0	PIN_E3	2.5 V		12mA (default)	2 (default)
clk	Input	PIN_H6	2	B2_N0	PIN_H6	2.5 V		12mA (default)	
led[4]	Output	PIN_B10	8	B8_N0	PIN_B10	2.5 V		12mA (default)	2 (default)
led[3]	Output	PIN_A10	8	B8_N0	PIN_A10	2.5 V		12mA (default)	2 (default)
led[2]	Output	PIN_A11	8	B8_N0	PIN_A11	2.5 V		12mA (default)	2 (default)
led[1]	Output	PIN_A9	8	B8_N0	PIN_A9	2.5 V		12mA (default)	2 (default)
led[0]	Output	PIN_A8	8	B8_N0	PIN_A8	2.5 V		12mA (default)	2 (default)
s1[3]	Input	PIN_D1	1A	B1_N0	PIN_D1	2.5 V		12mA (default)	
s1[2]	Input	PIN_C1	1A	B1_N0	PIN_C1	2.5 V		12mA (default)	
s1[1]	Input	PIN_C2	1A	B1_N0	PIN_C2	2.5 V		12mA (default)	
s1[0]	Input	PIN_E1	1A	B1_N0	PIN_E1	2.5 V		12mA (default)	
s2[3]	Input	PIN_G12	5	B5_N0	PIN_G12	2.5 V		12mA (default)	
s2[2]	Input	PIN_J2	2	B2_N0	PIN_J2	2.5 V		12mA (default)	
s2[1]	Input	PIN_J1	2	B2_N0	PIN_J1	2.5 V		12mA (default)	
s2[0]	Input	PIN_H4	2	B2_N0	PIN_H4	2.5 V		12mA (default)	
seg[6]	Output	PIN_K11	5	B5_N0	PIN_K11	2.5 V		12mA (default)	2 (default)
seg[5]	Output	PIN_F1	1A	B1_N0	PIN_F1	2.5 V		12mA (default)	2 (default)
seg[4]	Output	PIN_K12	5	B5_N0	PIN_K12	2.5 V		12mA (default)	2 (default)
seg[3]	Output	PIN_E4	1A	B1_N0	PIN_E4	2.5 V		12mA (default)	2 (default)
seg[2]	Output	PIN_J10	5	B5_N0	PIN_J10	2.5 V		12mA (default)	2 (default)
seg[1]	Output	PIN_H8	5	B5_N0	PIN_H8	2.5 V		12mA (default)	2 (default)
seg[0]	Output	PIN_H10	5	B5_N0	PIN_H10	2.5 V		12mA (default)	2 (default)
led[7]	Unknown	PIN_D8	8	B8_N0		2.5 V (default)		12mA (default)	
powered	Unknown	PIN_H5	2	B2_N0		2.5 V (default)		12mA (default)	
led[6]	Unknown	PIN_C10	8	B8_N0		2.5 V (default)		12mA (default)	
led[5]	Unknown	PIN_C9	8	B8_N0		2.5 V (default)		12mA (default)	

Fig. 4: Pin placements

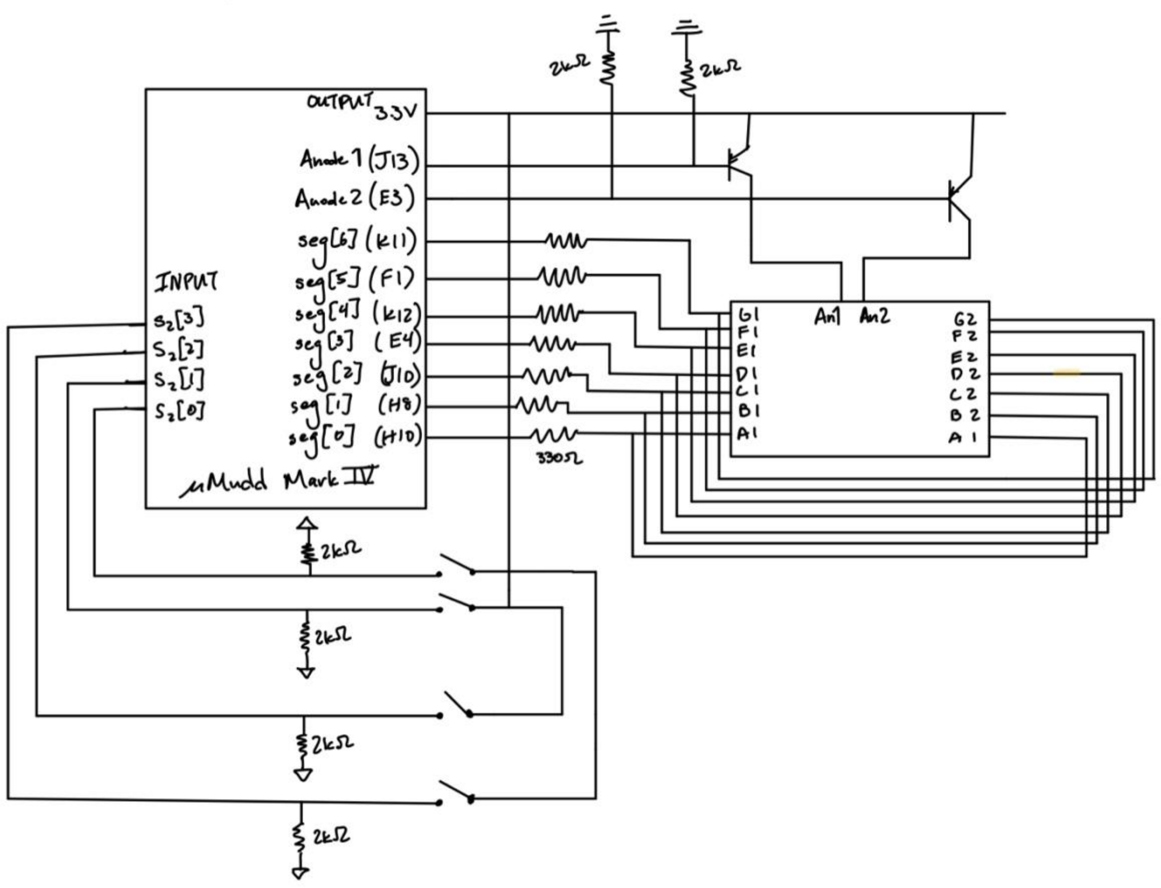


Fig. 5: Circuit schematic

Results and Discussion:

I was successful in building a time-multiplexed 7-segment display. However, at the time of writing I am running into issues with certain segments not lighting up properly in certain cases. All segments will light up with certain cases. Otherwise, the LEDs are adding up properly and the display is seamless.

If I were to redo this lab I would first refresh myself on the function of transistors and draw a circuit schematic before anything else.

Conclusion:

I feel that I was successful in completing all tasks and may have been limited by the board I was testing with. The system works as intended except for a few cases on the display, which I'm unsure is a problem with my wiring or with internal issues with the board.

This lab took roughly 15 hours to complete. A little more instruction would have been nice, but I'm sure that the point of this lab was to get us to start thinking independently more.