

ECE3300L

Summer 2025 Semester

Lab 4

Kevin Yu

Noah Bocanegra

7/11/25

For the fourth lab, we are tasked with verifying a binary to hexadecimal to decimal converter. SW[0:31] will be set to 8 different numbers, where SW[0:3] will represent one number, SW[4:7] will represent another number and so on. The module will also have an internal counter, where each positive edge will increment the counter, and a low reset signal will set it to 0. By taking the counter's 3 most significant digits, the module will determine which SW number will be chosen to be output by the Cnode wires to be displayed on the 7 segment display.

We were given a lot of code to use for our seven segment display, but we had to modify it to fit our parameters. We only have 16 switches, so the 32 switch input was not going to work. We also wanted to use all 16 switches which meant that our reset had to instead be mapped out to a button. We could've used a debounce module once again, but we found it was much easier to just change the active high to active low by modifying the rst_n line and the negedge to a posedge. Below is the driver module for the seven segment display:

```
`timescale 1ns / 1ps
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 posedge 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

```

Constraint:

```

set_property -dict { PACKAGE_PIN E3  IOSTANDARD LVCMOS33 } [get_ports { clk }];
#IO_L12P_T1_MRCC_35 Sch=clk100mhz
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports {clk}];

```

##Switches

```

set_property -dict { PACKAGE_PIN J15  IOSTANDARD LVCMOS33 } [get_ports { SW[0] }];
#IO_L24N_T3_RS0_15 Sch=sw[0]
set_property -dict { PACKAGE_PIN L16  IOSTANDARD LVCMOS33 } [get_ports { SW[1] }];
#IO_L3N_T0_DQS_EMCCLK_14 Sch=sw[1]
set_property -dict { PACKAGE_PIN M13  IOSTANDARD LVCMOS33 } [get_ports { SW[2] }];
#IO_L6N_T0_D08_VREF_14 Sch=sw[2]
set_property -dict { PACKAGE_PIN R15  IOSTANDARD LVCMOS33 } [get_ports { SW[3] }];
#IO_L13N_T2_MRCC_14 Sch=sw[3]
set_property -dict { PACKAGE_PIN R17  IOSTANDARD LVCMOS33 } [get_ports { SW[4] }];
#IO_L12N_T1_MRCC_14 Sch=sw[4]
set_property -dict { PACKAGE_PIN T18  IOSTANDARD LVCMOS33 } [get_ports { SW[5] }];
#IO_L7N_T1_D10_14 Sch=sw[5]
set_property -dict { PACKAGE_PIN U18  IOSTANDARD LVCMOS33 } [get_ports { SW[6] }];
#IO_L17N_T2_A13_D29_14 Sch=sw[6]
set_property -dict { PACKAGE_PIN R13  IOSTANDARD LVCMOS33 } [get_ports { SW[7] }];
#IO_L5N_T0_D07_14 Sch=sw[7]
set_property -dict { PACKAGE_PIN T8   IOSTANDARD LVCMOS18 } [get_ports { SW[8] }];
#IO_L24N_T3_34 Sch=sw[8]
set_property -dict { PACKAGE_PIN U8   IOSTANDARD LVCMOS18 } [get_ports { SW[9] }];
#IO_25_34 Sch=sw[9]

```

```

set_property -dict { PACKAGE_PIN R16  IOSTANDARD LVCMOS33 } [get_ports { SW[10] }];
#IO_L15P_T2_DQS_RDWR_B_14 Sch=sw[10]
set_property -dict { PACKAGE_PIN T13  IOSTANDARD LVCMOS33 } [get_ports { SW[11] }];
#IO_L23P_T3_A03_D19_14 Sch=sw[11]
set_property -dict { PACKAGE_PIN H6   IOSTANDARD LVCMOS33 } [get_ports { SW[12] }];
#IO_L24P_T3_35 Sch=sw[12]
set_property -dict { PACKAGE_PIN U12  IOSTANDARD LVCMOS33 } [get_ports { SW[13] }];
#IO_L20P_T3_A08_D24_14 Sch=sw[13]
set_property -dict { PACKAGE_PIN U11  IOSTANDARD LVCMOS33 } [get_ports { SW[14] }];
#IO_L19N_T3_A09_D25_VREF_14 Sch=sw[14]
set_property -dict { PACKAGE_PIN V10  IOSTANDARD LVCMOS33 } [get_ports { SW[15] }];
#IO_L21P_T3_DQS_14 Sch=sw[15]

```

LEDs

```

set_property -dict { PACKAGE_PIN H17  IOSTANDARD LVCMOS33 } [get_ports { LED[0] }];
#IO_L18P_T2_A24_15 Sch=led[0]
set_property -dict { PACKAGE_PIN K15  IOSTANDARD LVCMOS33 } [get_ports { LED[1] }];
#IO_L24P_T3_RS1_15 Sch=led[1]
set_property -dict { PACKAGE_PIN J13  IOSTANDARD LVCMOS33 } [get_ports { LED[2] }];
#IO_L17N_T2_A25_15 Sch=led[2]
set_property -dict { PACKAGE_PIN N14  IOSTANDARD LVCMOS33 } [get_ports { LED[3] }];
#IO_L8P_T1_D11_14 Sch=led[3]
set_property -dict { PACKAGE_PIN R18  IOSTANDARD LVCMOS33 } [get_ports { LED[4] }];
#IO_L7P_T1_D09_14 Sch=led[4]
set_property -dict { PACKAGE_PIN V17  IOSTANDARD LVCMOS33 } [get_ports { LED[5] }];
#IO_L18N_T2_A11_D27_14 Sch=led[5]
set_property -dict { PACKAGE_PIN U17  IOSTANDARD LVCMOS33 } [get_ports { LED[6] }];
#IO_L17P_T2_A14_D30_14 Sch=led[6]
set_property -dict { PACKAGE_PIN U16  IOSTANDARD LVCMOS33 } [get_ports { LED[7] }];
#IO_L18P_T2_A12_D28_14 Sch=led[7]
set_property -dict { PACKAGE_PIN V16  IOSTANDARD LVCMOS33 } [get_ports { LED[8] }];
#IO_L16N_T2_A15_D31_14 Sch=led[8]
set_property -dict { PACKAGE_PIN T15  IOSTANDARD LVCMOS33 } [get_ports { LED[9] }];
#IO_L14N_T2_SRCC_14 Sch=led[9]
set_property -dict { PACKAGE_PIN U14  IOSTANDARD LVCMOS33 } [get_ports { LED[10] }];
#IO_L22P_T3_A05_D21_14 Sch=led[10]
set_property -dict { PACKAGE_PIN T16  IOSTANDARD LVCMOS33 } [get_ports { LED[11] }];
#IO_L15N_T2_DQS_DOUT_CSO_B_14 Sch=led[11]
set_property -dict { PACKAGE_PIN V15  IOSTANDARD LVCMOS33 } [get_ports { LED[12] }];
#IO_L16P_T2_CSI_B_14 Sch=led[12]
set_property -dict { PACKAGE_PIN V14  IOSTANDARD LVCMOS33 } [get_ports { LED[13] }];
#IO_L22N_T3_A04_D20_14 Sch=led[13]
set_property -dict { PACKAGE_PIN V12  IOSTANDARD LVCMOS33 } [get_ports { LED[14] }];
#IO_L20N_T3_A07_D23_14 Sch=led[14]

```

```
set_property -dict { PACKAGE_PIN V11  IOSTANDARD LVCMOS33 } [get_ports { LED[15] }];  
#IO_L21N_T3_DQS_A06_D22_14 Sch=led[15]
```

RGB LEDs

```
#set_property -dict { PACKAGE_PIN R12  IOSTANDARD LVCMOS33 } [get_ports { LED16_B  
}]; #IO_L5P_T0_D06_14 Sch=led16_b  
#set_property -dict { PACKAGE_PIN M16  IOSTANDARD LVCMOS33 } [get_ports { LED16_G  
}]; #IO_L10P_T1_D14_14 Sch=led16_g  
#set_property -dict { PACKAGE_PIN N15  IOSTANDARD LVCMOS33 } [get_ports { LED16_R  
}]; #IO_L11P_T1_SRCC_14 Sch=led16_r  
#set_property -dict { PACKAGE_PIN G14  IOSTANDARD LVCMOS33 } [get_ports { LED17_B  
}]; #IO_L15N_T2_DQS_ADV_B_15 Sch=led17_b  
#set_property -dict { PACKAGE_PIN R11  IOSTANDARD LVCMOS33 } [get_ports { LED17_G  
}]; #IO_0_14 Sch=led17_g  
#set_property -dict { PACKAGE_PIN N16  IOSTANDARD LVCMOS33 } [get_ports { LED17_R  
}]; #IO_L11N_T1_SRCC_14 Sch=led17_r
```

##7 segment display

```
set_property -dict { PACKAGE_PIN T10  IOSTANDARD LVCMOS33 } [get_ports { Cnode[6] }];  
#IO_L24N_T3_A00_D16_14 Sch=ca  
set_property -dict { PACKAGE_PIN R10  IOSTANDARD LVCMOS33 } [get_ports { Cnode[5] }];  
#IO_25_14 Sch=cb  
set_property -dict { PACKAGE_PIN K16  IOSTANDARD LVCMOS33 } [get_ports { Cnode[4] }];  
#IO_25_15 Sch=cc  
set_property -dict { PACKAGE_PIN K13  IOSTANDARD LVCMOS33 } [get_ports { Cnode[3] }];  
#IO_L17P_T2_A26_15 Sch=cd  
set_property -dict { PACKAGE_PIN P15  IOSTANDARD LVCMOS33 } [get_ports { Cnode[2] }];  
#IO_L13P_T2_MRCC_14 Sch=ce  
set_property -dict { PACKAGE_PIN T11  IOSTANDARD LVCMOS33 } [get_ports { Cnode[1] }];  
#IO_L19P_T3_A10_D26_14 Sch=cf  
set_property -dict { PACKAGE_PIN L18  IOSTANDARD LVCMOS33 } [get_ports { Cnode[0] }];  
#IO_L4P_T0_D04_14 Sch=cg  
set_property -dict { PACKAGE_PIN H15  IOSTANDARD LVCMOS33 } [get_ports { dp }];  
#IO_L19N_T3_A21_VREF_15 Sch=dp  
set_property -dict { PACKAGE_PIN J17  IOSTANDARD LVCMOS33 } [get_ports { AN[0] }];  
#IO_L23P_T3_FOE_B_15 Sch=an[0]  
set_property -dict { PACKAGE_PIN J18  IOSTANDARD LVCMOS33 } [get_ports { AN[1] }];  
#IO_L23N_T3_FWE_B_15 Sch=an[1]  
set_property -dict { PACKAGE_PIN T9   IOSTANDARD LVCMOS33 } [get_ports { AN[2] }];  
#IO_L24P_T3_A01_D17_14 Sch=an[2]  
set_property -dict { PACKAGE_PIN J14  IOSTANDARD LVCMOS33 } [get_ports { AN[3] }];  
#IO_L19P_T3_A22_15 Sch=an[3]  
set_property -dict { PACKAGE_PIN P14  IOSTANDARD LVCMOS33 } [get_ports { AN[4] }];  
#IO_L8N_T1_D12_14 Sch=an[4]
```

```

set_property -dict { PACKAGE_PIN T14  IOSTANDARD LVCMOS33 } [get_ports { AN[5] }];
#IO_L14P_T2_SRCC_14 Sch=an[5]
set_property -dict { PACKAGE_PIN K2   IOSTANDARD LVCMOS33 } [get_ports { AN[6] }];
#IO_L23P_T3_35 Sch=an[6]
set_property -dict { PACKAGE_PIN U13   IOSTANDARD LVCMOS33 } [get_ports { AN[7] }];
#IO_L23N_T3_A02_D18_14 Sch=an[7]

```

Testbench:

Our testbench tests all the possible outputs of the 7 segment display by iterating through numbers 0-8 for the first numbers set on SW[0:31], then it switches the SW[0:31] numbers to 9-15. However, during testing, we were unable to get AN to output anything other than ZZ, which is something we don't particularly understand why, as all the other outputs of the module are in line with our expectations. In simulation, we also modified the module code in order to reduce the amount of time needed to wait for the digit to switch numbers to be output by Cnode.

```

module seg7_driver_tb(

);
reg clk_tb, rst_n_tb;
reg [31:0] SW_tb;
wire [6:0] Cnode_tb;
wire dp_tb;
wire [7:0] An_tb;

integer i;

seg7_driver A(
    .clk(clk_tb),
    .rst_n(rst_n_tb),
    .SW(SW_tb),
    .Cnode(Cnode_tb),
    .dp(dp_tb),
    .AN(AN_tb)
);

always
begin
    #5 clk_tb = ~clk_tb;
end
initial

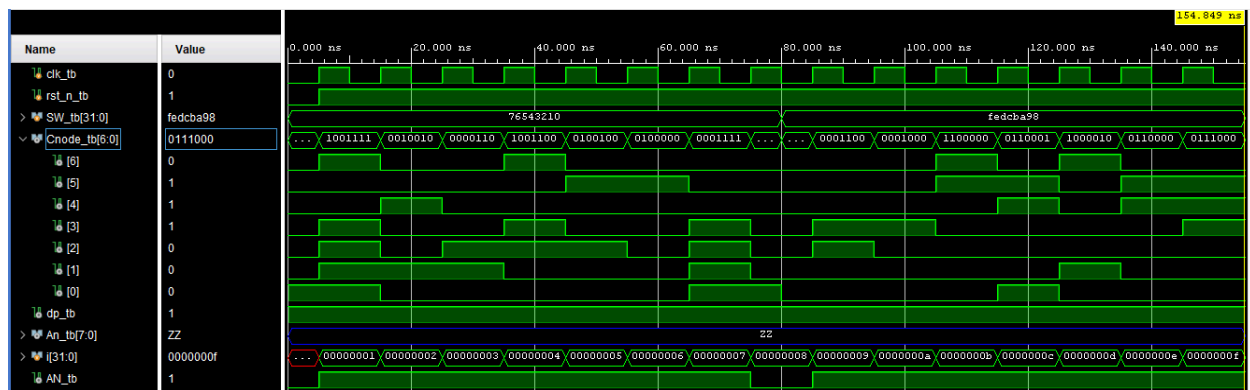
```

```

begin
    SW_tb = 32'b0111_0110_0101_0100_0011_0010_0001_0000;
    #80;
    SW_tb = 32'b1111_1110_1101_1100_1011_1010_1001_1000;
end

initial
begin
    clk_tb = 0;
    rst_n_tb = 0;
    #5 rst_n_tb = 1;
    for(i = 1; i < 16; i = i+1)
    begin
        $display("Current i = ", i);
        #10;
    end
    $finish;
end
end

```



In conclusion, we were able to use the seven segment displays for the first time, and apply some of our previous lab lessons, such as understanding the way the mux in the seven segment display code, and use these lessons to further our understanding of the A7 board and vivado programming.

Work Division:

Kevin Yu: Lab report + behavioral simulation

Noah Bocanegra: Lab report + XDC file + board demo