ELC 2137 Lab 08: 4-digit Display

Yiting Wang

October 22, 2020

Summary

We have created a 2-digit, 7-segment display and a BCD converter. In this lab, we expand this to a 4-digit display and add the ability to switch between hexadecimal and decimal (BCD) output. After completing this lab, we should be able to:
Use parameters to create flexible, reusable modules.
Import modules, modify modules, and use them to design a modular system.

Q&A

There is no question in the lab 08 assignment.

Results

Firgure 1 is the simulation waveform and ERT of the mux2.

Time (ns):	0	10	20	30
in1	0001	0001	0011	0011
in0	0000	0000	0101	0101
sel	0	1	0	1
out	0000	0001	0101	0011

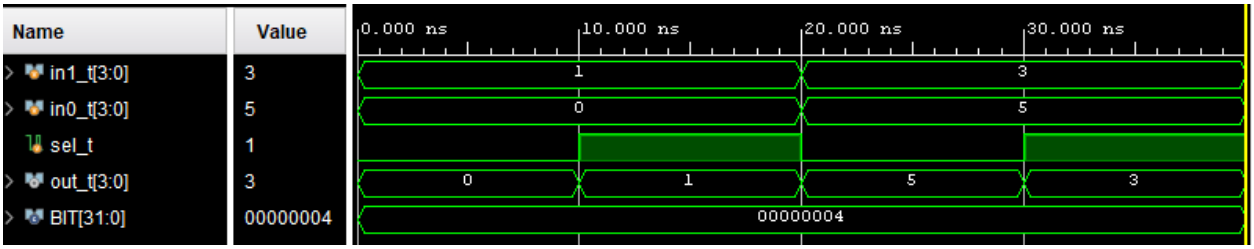

Figure 1: the simulation waveform and ERT of the mux2

Firgure 2 is the simulation waveform and ERT of the mux4.

Time (ns):	0	10	20	30
in3	0001	0001	0001	0001
in2	0000	0000	0000	0000
in1	0011	0011	0011	0011
in0	0101	0101	0101	0101
sel	00	01	10	11
out	0101	0011	0000	0001

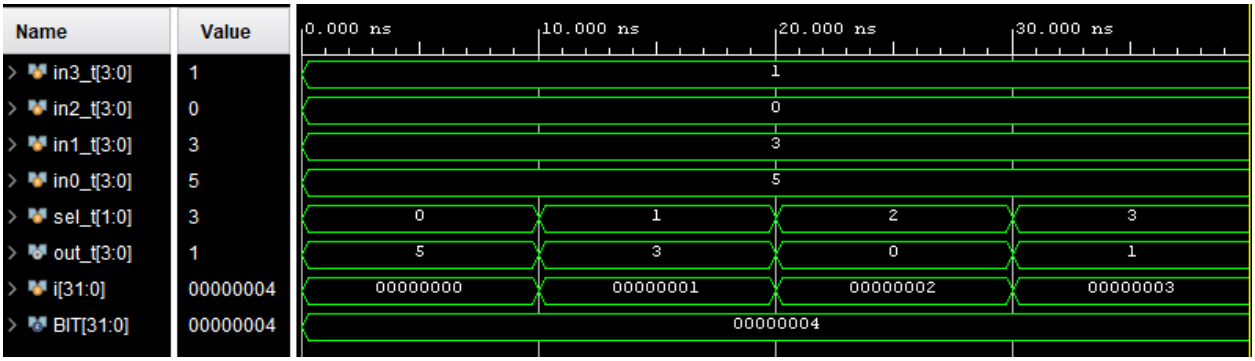

Figure 2: the simulation waveform and ERT of the mux4

Firgure 3 is the simulation waveform and ERT of the anode decoder.

Time (ns):	0	10	20	30
in	00	01	10	11
out	1110	1101	1011	0111

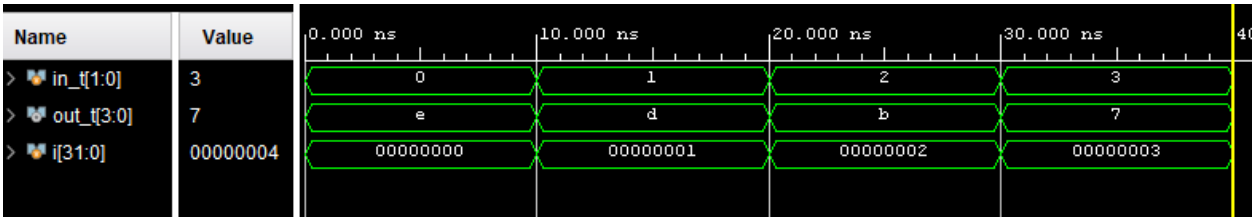

Figure 3: the simulation waveform and ERT of the anode decoder

This is the picture of a copy of the relevant TCL Console output for Top-Level Simulation.

```
# run 1000ns
    +--------- Digit 3 (? = incorrect segment values)
    |+-------- Decimal point
    ||+------- Digit 2 (? = incorrect segment values)
    |||+------ Decimal point
    ||||+----- Digit 1 (? = incorrect segment values)
    |||||+---- Decimal point
    ||||||+--- Digit 0 (? = incorrect segment values)
    |||||||+-- Decimal point
    ||||||||
-->       0
-->       C
-->       2
-->     1
-->   0
-->   1
-->     4
-->       0
--> 0
--> 1
--> 0
--> -
-->   4
-->     8
-->       C
$finish called at time : 150 ns : File "H:/DL2137_Yiting/Lab08/basys3.sv" Line 98
```

Code

File Inclusion

Listing 1: mux2 Verilog code

```
`timescale 1ns / 1ps
//
    //////////////////////////////////////////////////////////////////////////////////

// Company: ELC 2137
// Engineer: Yiting Wang
// Create Date: 10/15/2020
//
    //////////////////////////////////////////////////////////////////////////////////



module mux2
    #(parameter BIT=4)
    (
```

```
    input [BIT-1:0] in1,
    input [BIT-1:0] in0,
    input sel,
    output [BIT-1:0] out
    );

    assign out = sel ? in1 : in0;

endmodule
```

File Inclusion

Listing 2: mux2 Test Benches Verilog code

```
`timescale 1ns / 1ps
//
   //////////////////////////////////////////////////////////////////////////////////

// Company: ELC 2137
// Engineer: Yiting Wang
// Create Date: 10/15/2020
//
   //////////////////////////////////////////////////////////////////////////////////



module mux2_test();
localparam BIT=4;
    reg [BIT-1:0] in1_t, in0_t;
    reg sel_t;
    wire [BIT-1:0] out_t;

    mux2 dut(
    .in1(in1_t),
    .in0(in0_t),
    .sel(sel_t),
    .out(out_t)
    );

    initial begin
        in1_t = 16'h1111; in0_t = 16'h0000; sel_t = 0; #10;
        sel_t = 1; #10;
        in1_t = 16'h1013; in0_t = 16'h0105; sel_t = 0; #10;
        sel_t = 1; #10;
        $finish;
    end

endmodule
```

File Inclusion

Listing 3: mux4 Verilog code

```
`timescale 1ns / 1ps
```

```
//
   //////////////////////////////////////////////////////////////////////////////////

// Company: ELC 2137
// Engineer: Yiting Wang
// Create Date: 10/15/2020
//
   //////////////////////////////////////////////////////////////////////////////////



module mux4
    #(parameter BIT=4)
    (
    input [BIT-1:0] in3, in2, in1, in0,
    input [1:0] sel,
    output reg [BIT-1:0] out
    );

    always @*
        case(sel)
            2'b11: out = in3;
            2'b10: out = in2;
            2'b01: out = in1;
            default: out = in0;
        endcase

endmodule
```

Listing 4: mux4 Test Benches Verilog code

File Inclusion

```
`timescale 1ns / 1ps
//
   //////////////////////////////////////////////////////////////////////////////////

// Company: ELC 2137
// Engineer: Yiting Wang
// Create Date: 10/15/2020
//
   //////////////////////////////////////////////////////////////////////////////////



module mux4_test();
    localparam BIT=4;
    reg [BIT-1:0] in3_t, in2_t, in1_t, in0_t;
    reg [1:0] sel_t;
    wire [BIT-1:0] out_t;

    mux4 dut(
    .in3(in3_t), .in2(in2_t), .in1(in1_t), .in0(in0_t),
    .sel(sel_t),
```

```
    .out(out_t)
    );

    integer i;

    initial begin
        in3_t = 16'h1111; in2_t = 16'h0000; in1_t = 16'h1013; in0_t = 16'
           h0105;
        for (i=0; i<=8'h3; i=i+1) begin
            sel_t = i;
            #10;
        end
        $finish;
    end

endmodule
```

File Inclusion

Listing 5: anode decoder Verilog code

```
`timescale 1ns / 1ps
//
   //////////////////////////////////////////////////////////////////////////////////

// Company: ELC 2137
// Engineer: Yiting Wang
// Create Date: 10/15/2020
//
   //////////////////////////////////////////////////////////////////////////////////



module anode_decoder(
    input [1:0] in,
    output reg [3:0] out
    );

    always @*
        case(in)
            2'b11: out = 4'b0111;
            2'b10: out = 4'b1011;
            2'b01: out = 4'b1101;
            default: out = 4'b1110;
        endcase

endmodule
```

File Inclusion

Listing 6: anode decoder Test Benches Verilog code

```
`timescale 1ns / 1ps
```

```
//
    //////////////////////////////////////////////////////////////////////////////////

// Company: ELC 2137
// Engineer: Yiting Wang
// Create Date: 10/15/2020
//
    //////////////////////////////////////////////////////////////////////////////////



module anode_decoder_test();

    reg [1:0] in_t;
    wire [3:0] out_t;

    anode_decoder dut(
    .in(in_t),
    .out(out_t)
    );

    integer i;

    initial begin
        for (i=0; i<=8'h3; i=i+1) begin
            in_t = i;
            #10;
        end
        $finish;
    end

endmodule
```

File Inclusion

Listing 7: sseg4 Verilog code

```
`timescale 1ns / 1ps
//
    //////////////////////////////////////////////////////////////////////////////////

// Company: ELC 2137
// Engineer: Yiting Wang
// Create Date: 10/15/2020
//
    //////////////////////////////////////////////////////////////////////////////////



module sseg4(
    input [15:0] data,
    input hex_dec,
    input sign,
    input [1:0] digit_sel,
```

```
    output reg [6:0] seg,
    output dp,
    output reg [3:0] an
    );

    wire [15:0] out_bcd11, out_mux2;
    wire [3:0] out_mux4;
    wire [6:0] out_sseg_decoder;

    bcd11 dut0(
        .in(data[10:0]),
        .ones(out_bcd11[3:0]), .tens(out_bcd11[7:4]), .hund(out_bcd11
           [11:8]), .thou(out_bcd11[15:12])
    );

    mux2 #( .BIT(16) ) dut1(
        .in1(data), .in0(out_bcd11), .sel(hex_dec),
        .out(out_mux2)
    );

    mux4  #( .BIT(4) ) dut2(
        .in3(out_mux2[15:12]), .in2(out_mux2[11:8]), .in1(out_mux2[7:4]),
           .in0(out_mux2[3:0]), .sel(digit_sel),
        .out(out_mux4)
    );

    sseg_decoder dut3(
        .num(out_mux4),
        .sseg(out_sseg_decoder)
    );



    anode_decoder dut4(
        .in(digit_sel),
        .out(an)
    );

    wire nan;
    assign nan = ~an[3];

    wire sel_mux2;
    assign sel_mux2 = sign & nan;

    mux2  #( .BIT(7) ) dut5(
        .in1(7'b0111111), .in0(out_sseg_decoder), .sel(sel_mux2),
        .out(seg)
    );

    assign dp = 1;

endmodule
```

File Inclusion

Listing 8: sseg4 manual Verilog code

```
`timescale 1ns / 1ps
//
   //////////////////////////////////////////////////////////////////////////////////

// Company: ELC 2137
// Engineer: Yiting Wang
// Create Date: 10/15/2020
//
   //////////////////////////////////////////////////////////////////////////////////



module sseg4_manual(
    input [15:0] sw,
    input clk,

    output [6:0] seg,
    output dp,
    output [3:0] an
    );

    sseg4 my_sseg(
        .data({ 4'b0000, sw[11:0] }), .hex_dec(sw[15]), .sign(sw[14]), .
           digit_sel(sw[13:12]),
        .seg(seg), .dp(dp), .an(an)
    );

    assign clk = 1;

endmodule
```