ECE 3300L

Lab Report #3

Group E

Paul Kim (ID: 015236949)

Winson Zhu (ID: 016416790)

July 02, 2025

Design:

Implements a 2x1 MUX using AND, OR, NOT gates (gate-level logic)

```
module mux2x1 (
                input a, b,
                input sel,
                output y
               );

wire nsel, a1, b1;

not (nsel, sel);
and (a1, a, nsel);
and (b1, b, sel);
or (y, a1, b1);

endmodule
```

The 16x1 MUX is built by nesting multiple 2x1modules using generate loops

```
module mux16x1 (
               input [15:0] in,
               input [3:0] sel,
               output out
               );
wire [15:0] level1;
wire [7:0] level2;
wire [3:0] level3;
genvar i;
generate
    for (i = 0; i < 8; i = i + 1)
        mux2x1 m1 (.a(in[2*i]), .b(in[2*i+1]), .sel(sel[0]), .y(level1[i]));
    for (i = 0; i < 4; i = i + 1)
        mux2x1 m2 (.a(level1[2*i]), .b(level1[2*i+1]), .sel(sel[1]), .y(level2[i]));
    for (i = 0; i < 2; i = i + 1)
        mux2x1 m3 (.a(level2[2*i]), .b(level2[2*i+1]), .sel(sel[2]), .y(level3[i]));
        mux2x1 m4 (.a(level3[0]), .b(level3[1]), .sel(sel[3]), .y(out));
    endgenerate
endmodule
```

Debounce removes noise and bouncing effects from pushbutton inputs

```
module debounce (
                 input clk,
                 input btn_in,
                 output reg btn_clean
                );

reg [2:0] shift_reg;

always @(posedge clk)
    begin
        shift_reg <= {shift_reg[1:0], btn_in};
        if (shift_reg == 3'b111) btn_clean <= 1;
        else if (shift_reg == 3'b000) btn_clean <= 0;
    end
endmodule
```

Toggle_switch acts as a toggle flip-flop for each select bit, driven by button signals

```
module toggle_switch (
                        input clk,
                        input rst,
                        input btn_raw,
                        output reg state
                      );
wire btn_clean;
reg btn_prev;
debounce db (.clk(clk), .btn_in(btn_raw), .btn_clean(btn_clean));
always @(posedge clk)
    begin
        if (rst)
            begin
                state <= 0;
                btn_prev <= 0;
            end
            else
                begin
                    if (btn_clean && !btn_prev)
                    state <= ~state;
                    btn_prev <= btn_clean;
                end
            end
endmodule
```

Integrates all submodules, connects board inputs (switches/buttons) to outputs (LED)

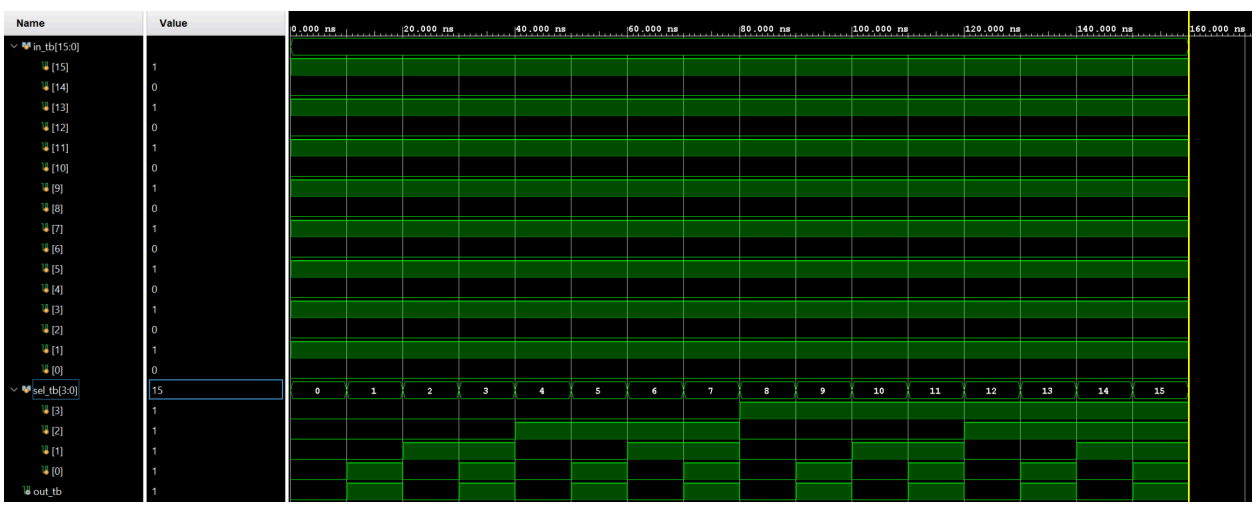
```
module top_mux_lab3 (
                        input clk,
                        input rst,
                        input [15:0] SW,
                        input btnU, btnD, btnL, btnR,
                        output LED0
                    );
wire [3:0] sel;

toggle_switch t0 (.clk(clk), .rst(rst), .btn_raw(btnD), .state(sel[0]));
toggle_switch t1 (.clk(clk), .rst(rst), .btn_raw(btnR), .state(sel[1]));
toggle_switch t2 (.clk(clk), .rst(rst), .btn_raw(btnL), .state(sel[2]));
toggle_switch t3 (.clk(clk), .rst(rst), .btn_raw(btnU), .state(sel[3]));

mux16x1 mux (.in(SW), .sel(sel), .out(LED0));

endmodule
```

Simulation:

We created a functional testbench (tb_mux16x1.v) module that has a set input pattern 16'b1010_1010_1010_1010. The code cycled select inputs from sel = 0 to 15 and verified that output (out_tb) always matched the correct in_tb[sel].

```
module tb_mux16x1;
    // Testbench signals
    reg [15:0] in_tb;
    reg [3:0] sel_tb;
    wire out_tb;
    // Instantiate
    mux16x1 DUT (
        .in(in_tb),
        .sel(sel_tb),
        .out(out_tb)
    );
    integer i;  //  Declare loop variable
    initial begin
        // Initialize inputs
        in_tb = 16'b1010_1010_1010_1010;  // Example input pattern
        sel_tb = 4'b0000;
        // Test all select values from 0 to 15
        for (i = 0; i < 16; i = i + 1) begin
            sel_tb = i;
            #10;  // Wait 10ns
            if (out_tb !== in_tb[i])
                $fatal(1, "FAIL: sel=%0d | Expected=%b | Got=%b", i, in_tb[i], out_tb);
            else
                $display("PASS: sel=%0d | out=%b", i, out_tb);
        end
        $display(" All tests completed successfully.");
        $finish;
    end
endmodule
```

Waveform Screenshot:



Console Output Screenshot:

```
Tcl Console   Messages   Log
# }
# }
# run 1000ns
PASS: sel=0 | out=0
PASS: sel=1 | out=1
PASS: sel=2 | out=0
PASS: sel=3 | out=1
PASS: sel=4 | out=0
PASS: sel=5 | out=1
PASS: sel=6 | out=0
PASS: sel=7 | out=1
PASS: sel=8 | out=0
PASS: sel=9 | out=1
PASS: sel=10 | out=0
PASS: sel=11 | out=1
PASS: sel=12 | out=0
PASS: sel=13 | out=1
PASS: sel=14 | out=0
PASS: sel=15 | out=1
 All tests completed successfully.
$finish called at time : 160 ns : File "C:/Users/sbpau/Downloads/ECE3300L Lab3 GroupE/ECE3300L Lab3 GroupE.srcs/sim 1/new/tb mux16x1.v" Line 55
```

Implementation:

Site Type	Used	Fixed	Prohibited	Available	Util%
Slice LUTs*	12	0	0	63400	0.02
LUT as Logic	12	0	0	63400	0.02
LUT as Memory	0	0	0	19000	0.00
Slice Registers	24	0	0	126800	0.02
Register as Flip Flop	24	0	0	126800	0.02
Register as Latch	0	0	0	126800	0.00
F7 Muxes	2	0	0	31700	<0.01
F8 Muxes	1	0	0	15850	<0.01

Total On-Chip Power (W)	0.100
Design Power Budget (W)	Unspecified*
Power Budget Margin (W)	NA
Dynamic (W)	0.003
Device Static (W)	0.097
Effective TJA (C/W)	4.6
Max Ambient (C)	84.5
Junction Temperature (C)	25.5
Confidence Level	Low
Setting File	---
Simulation Activity File	---
Design Nets Matched	NA

Contributions:

Paul Kim - Testbench, Simulation - 50% contribution

Winson Zhu - Implementation, Hardware Demo - 50% contribution