Julio Flores (ID# : 016326856)
Victor Perez (ID# : 016196050)

Group I

Session E02

Lab 6

Dual BCD up/down counter, ALU, and Control Display on 7-segment display

Monday

JULY 28, 2025

ECE 3300L

Summer 2025

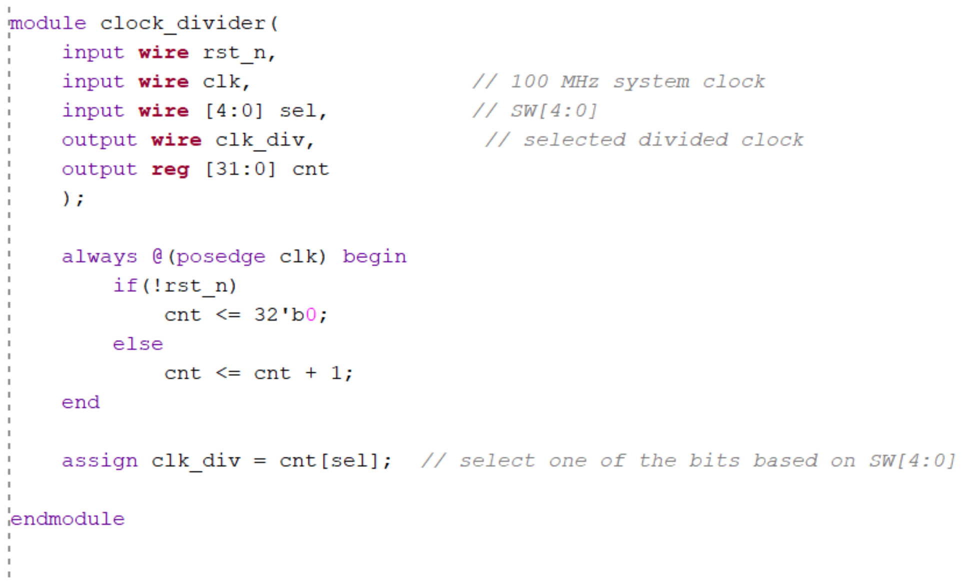
Objective:

The objective of this lab was to design and implement a digital system that combines two independent BCD up/down counters, a simple 4-bit ALU capable of performing addition and subtraction, and a control-display interface using switches and a 7-segment display. The lab aimed to demonstrate key digital logic concepts including clock division, hardware-based counting, and arithmetic operations using combinational logic. By using switches for ALU operation selection and counter direction control, the system provides an interactive way to explore how control logic affects computation and display output. The design also incorporated debugging tools such as LEDs to visualize raw BCD counter values and verify correct system behavior.

Design(Code):

Clock_divider.v

- Clock is a counter used to select a bit based on the switches



```
module clock_divider(
    input wire rst_n,
    input wire clk,                 // 100 MHz system clock
    input wire [4:0] sel,           // SW[4:0]
    output wire clk_div,             // selected divided clock
    output reg [31:0] cnt
    );

    always @(posedge clk) begin
        if(!rst_n)
            cnt <= 32'b0;
        else
            cnt <= cnt + 1;
    end

    assign clk_div = cnt[sel];  // select one of the bits based on SW[4:0]

endmodule
```

Bcd_counter.v

- This counter increases to 9 and resets to 0.

```
module bcd_counter(
    input wire clk,
    input wire rst_n,
    input wire dir_bit,                  // SW7 or SW8
    output reg [3:0] bcd_out       // BCD output (0-9)
);

    always @(posedge clk or negedge rst_n) begin
        if (!rst_n)
            bcd_out <= 4'd0;
        else if (dir_bit)
            bcd_out <= (bcd_out == 9) ? 0 : bcd_out + 1;
        else
            bcd_out <= (bcd_out == 0) ? 9 : bcd_out - 1;
        end

endmodule
```

Alu.v

- Sets the the switches as addition or subtraction

```
module alu(
    input [3:0] A,            // units
    input [3:0] B,            // tens
    input [1:0] ctrl,         // SW[6:5]
    output reg [7:0] result
);
    always @(*) begin
        case (ctrl)
            2'b00: result = A + B;
            2'b01: result = A - B;  // underflow will wrap due to unsigned
            default: result = 8'd0;
        endcase
    end
endmodule
```

control_decoder.V

- Assigns inputs and outputs to connect the switches to displays

```
module control_decoder(
    input wire [3:0] switches,      // {SW8, SW7, SW6, SW5}
    output wire [3:0] ctrl_nibble   // for 7-seg display
);

        assign ctrl_nibble = switches;

endmodule
```

Seg7_scan.v

- Case statements initializes the 7-segment displays
- Activates the digits and initializes what number to show.

```verilog
module seg7_scan(
    input wire clk,                         // main system clock
    input wire rst_n,                     // active-low reset
    input wire [3:0] digit0, digit1, digit2 ,            // 1 nibble from switches
    output reg [6:0] SEG,
    output reg [7:0] AN                  // Active-low anode signals
);
    reg [15:0] refresh_counter = 0;
    reg [1:0] sel = 0;
    reg [3:0] current_digit;

    always @(posedge clk or negedge rst_n) begin
        if (!rst_n)
            refresh_counter <= 0;
        else
            refresh_counter <= refresh_counter + 1;
    end

    always @(posedge refresh_counter[15] or negedge rst_n) begin
        if (!rst_n)
            sel <= 0;
        else
            sel <= sel + 1;
    end

    always @(*) begin
        case (sel)
            2'b00: begin
                AN <= 8'b11111110; // Activate digit0
                SEG <= seg_decoder(digit0);
            end
            2'b01: begin
                AN <= 8'b11111101; // Activate digit1
                SEG <= seg_decoder(digit1);
            end
            2'b10: begin
                AN <= 8'b11111011; // Activate digit1
                SEG <= seg_decoder(digit1);
            end
        endcase
    end
```

```verilog
    function [6:0] seg_decoder;
        input [3:0] num;
        case (num)
            4'd0: seg_decoder = 7'b1000000;
            4'd1: seg_decoder = 7'b1111001;
            4'd2: seg_decoder = 7'b0100100;
            4'd3: seg_decoder = 7'b0110000;
            4'd4: seg_decoder = 7'b0011001;
            4'd5: seg_decoder = 7'b0010010;
            4'd6: seg_decoder = 7'b0000010;
            4'd7: seg_decoder = 7'b1111000;
            4'd8: seg_decoder = 7'b0000000;
            4'd9: seg_decoder = 7'b0010000;
            default: seg_decoder = 7'b1111111; // off
        endcase
    endfunction

endmodule
```

Top_lab6.v

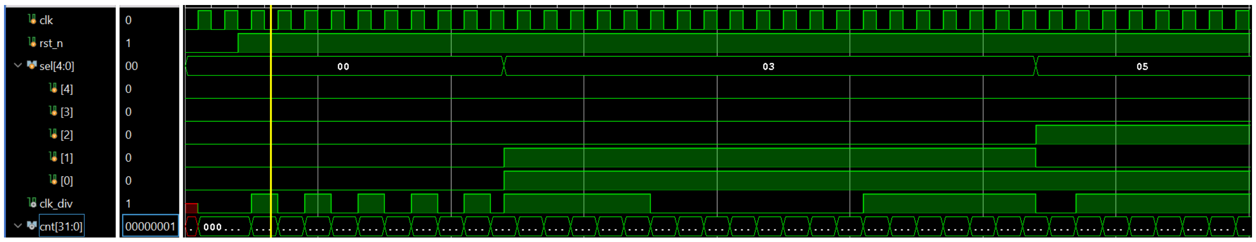
- Assigns every input and output needed for project
- Main module to call all the functions.

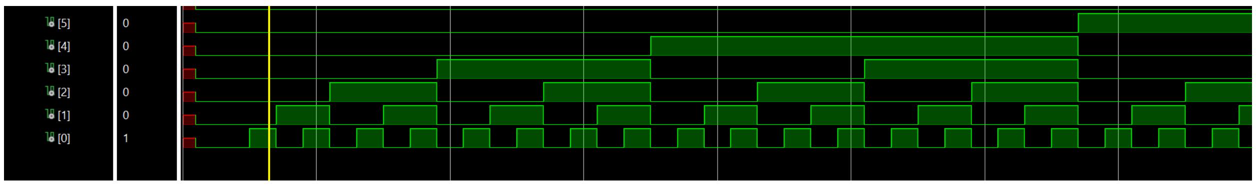
```
module top_lab6(
    input wire clk,
    input wire BTN0,
    input wire [8:0] SW,
    output wire [6:0] SEG,
    output wire [7:0] AN,
    output wire [7:0] LED
);

       wire rst_n = ~BTN0; // Active-low reset, BTN0 pressed = reset active
    wire clk_div;
    wire [31:0] count;
    wire [3:0] units_BCD, tens_BCD;
    wire [7:0] result;
    wire [3:0] ctrl_nibble;

    // Clock Divider
    clock_divider clkdiv_inst (
        .clk(clk),
        .sel(SW[4:0]),
        .rst_n(rst_n),
        .clk_div(clk_div),
        .cnt(count)
    );

    // BCD Counters
    bcd_counter units (
        .clk(clk_div),
        .rst_n(rst_n),
        .dir_bit(SW[7]),
        .bcd_out(units_BCD)
    );

    bcd_counter tens (
        .clk(clk_div),
        .rst_n(rst_n),
        .dir_bit(SW[8]),
        .bcd_out(tens_BCD)
    );

    // ALU
    alu alu_inst (
        .A(units_BCD),
        .B(tens_BCD),
        .ctrl(SW[6:5]),
        .result(result)
    );
    // Control Decoder
    control_decoder decoder (
        .switches(SW[8:5]),
        .ctrl_nibble(ctrl_nibble)
    );

    // 7-Segment Scanner
    seg7_scan display (
        .clk(clk),
        .rst_n(rst_n),
        .digit0(result[3:0]),
        .digit1(result[7:4]),
        .digit2(ctrl_nibble),
        .AN(AN),
        .SEG(SEG)
    );

    // LED debug output
    assign LED = result;

endmodule
```

Testbench and Waveform:

Clock_divider_tb.v

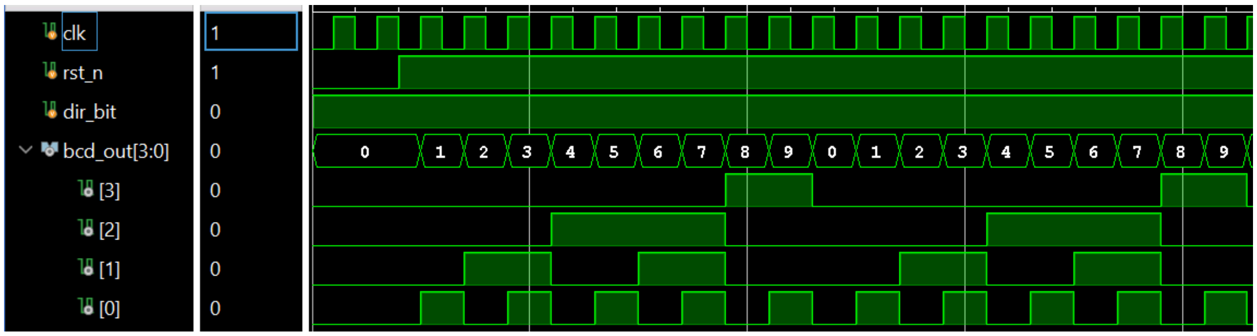
- This test bench shows the functionality of the clock as an increment of bits





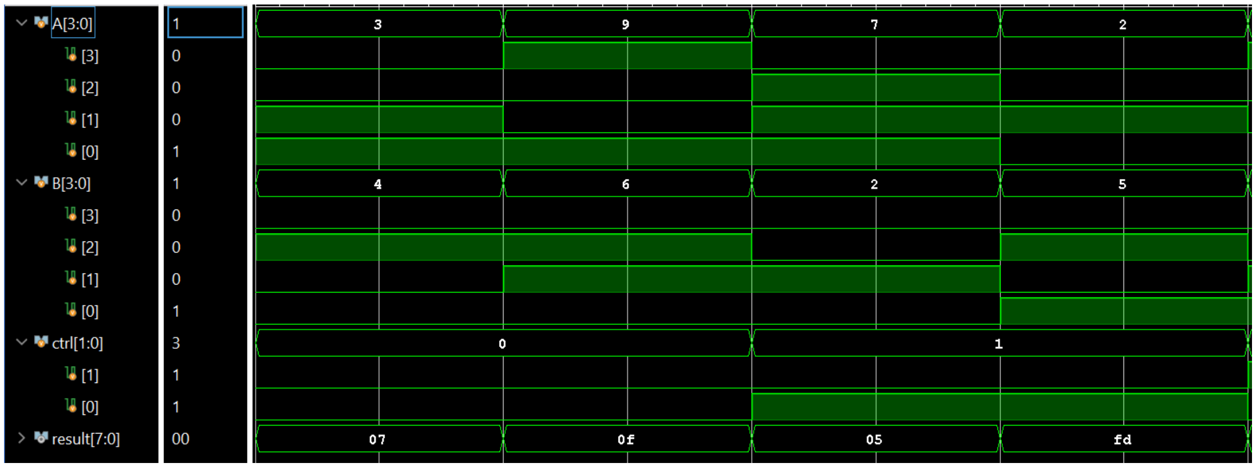
Bcd_counter_tb.v

- Loops the counter from 0-9



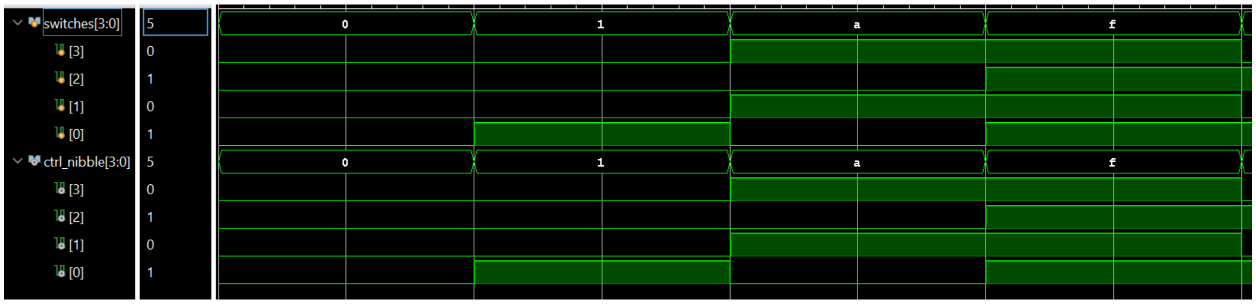
Alu_tb.v

- Bit 0 of the first digit and Bit 0 of the second digit are both 1 so the control is shown to be 2. The addition of both bits.



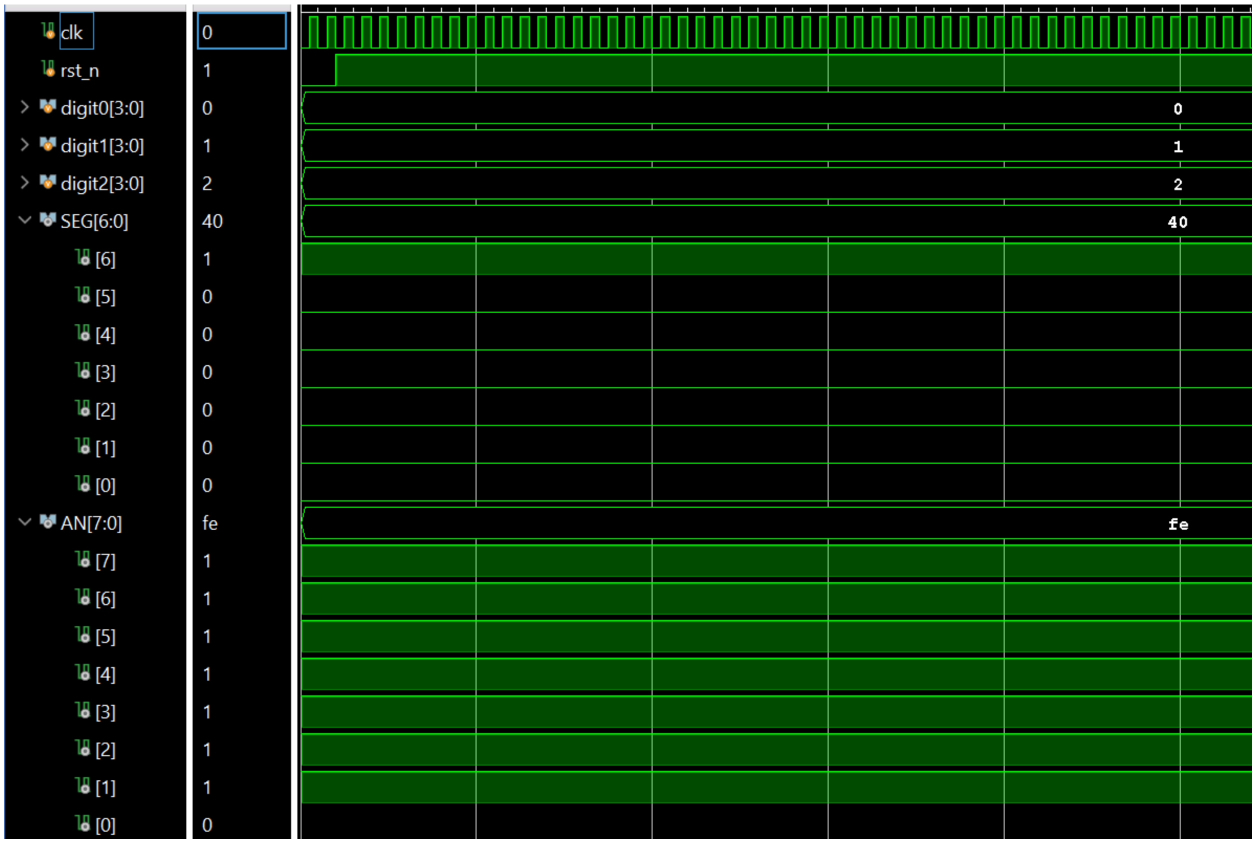
Control_decoder_tb.v

- Showing that the input and output equivalent



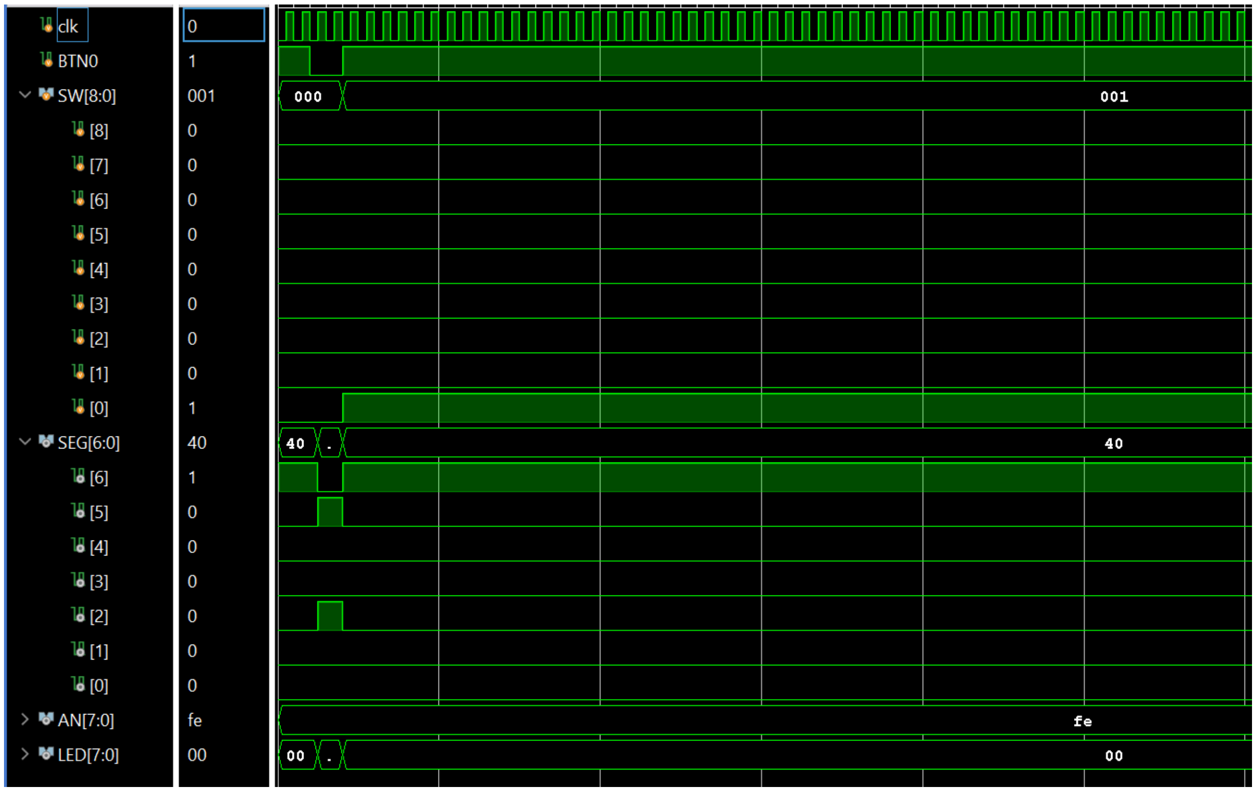
Seg7_scan_tb

- AN connects the 7 seg displays
- First 7-seg has everything turned on showing except for bit 6 showing number 8
- Second 7-seg has everything turned off except bit 0 showing number 0



Top_lab6_tb.v

- Initializes and calls all functions



Implementation:

```
+-------------------------+------+-------+------------+-----------+-------+
|        Site Type        | Used | Fixed | Prohibited | Available | Util% |
+-------------------------+------+-------+------------+-----------+-------+
| Slice LUTs*             |   35 |     0 |          0 |     63400 |  0.06 |
|   LUT as Logic          |   35 |     0 |          0 |     63400 |  0.06 |
|   LUT as Memory         |    0 |     0 |          0 |     19000 |  0.00 |
| Slice Registers         |   68 |     0 |          0 |    126800 |  0.05 |
|   Register as Flip Flop |   58 |     0 |          0 |    126800 |  0.05 |
|   Register as Latch     |   10 |     0 |          0 |    126800 | <0.01 |
| F7 Muxes                |    4 |     0 |          0 |     31700 |  0.01 |
| F8 Muxes                |    0 |     0 |          0 |     15850 |  0.00 |
+-------------------------+------+-------+------------+-----------+-------+
```

Contributions:

Julio Flores: 50% - Verilog Code (seg7_scan, top_lab5, control decoder, and testbench for these modules) and report

Victor Perez : 50% - Verilog Code (clk divider, and bcd counter modules, Alu, and testbench for these modules) and report

Reflection:

In this lab we designed and implemented the integrated BCD up/down counters, 4-bit ALU, and control-display interface in addition to having individual components like clock dividers, counters, and arithmetic logic units working together. Furthermore, debugging tools like LEDs for visualizing raw BCD values was used for verifying the system's behavior and troubleshooting. This lab solidified the understanding of combinational logic for arithmetic operations and the practical application of sequential logic in counting.