CalPolyPomona
College of Engineering

Lab 5 Report

by

Jonathan Huynh #016137719

Adam Godfrey #015981472

Instructor: Dr. Mohamed Aly

Class: ECE 3300L .E02-OU - Verilog Design

July 21st, 2025

Code with Explanation:

BCD Up/Down Counter on 7-Segment Display Top Module Code

```
`timescale 1ns / 1ps
module top_lab5 (
    input wire CLK,
    input wire BTN0,
    input wire BTN1,
    input wire [4:0] SW,
    output wire [6:0] SEG,
    output wire [7:0] AN,
    output wire [4:0] SWLED,
    output wire [7:0] BCDLED
);

wire [31:0] tick_counter;
wire divided_clk;
wire [3:0] unit_bcd, ten_bcd;
wire carry_from_unit;
wire rst_n = BTN0;
wire dir = BTN1;

clock_divider clk_div_inst (
    .sys_clk(CLK),
    .reset_n(rst_n),
    .tick_count(tick_counter)
);

mux32x1 mux_inst (
    .tick_bits(tick_counter),
    .speed_select(SW),
    .clk_out(divided_clk)
);

bcd_up_down_counter unit_counter (
    .clk_pulse(divided_clk),
    .rst_n(rst_n),
    .dir_up(dir),
    .enable_count(1'b1),
    .bcd_value(unit_bcd),
    .carry_pulse(carry_from_unit)
);

bcd_up_down_counter ten_counter (
    .clk_pulse(divided_clk),
    .rst_n(rst_n),
    .dir_up(dir),
    .enable_count(carry_from_unit),
    .bcd_value(ten_bcd),
```

This Verilog module, *top_lab5*, is the top-level design for a two-digit BCD up/down counter system. It takes inputs from a system clock (**CLK**), two push buttons (**BTN0** for *reset* and **BTN1** for *direction*), and a 5-bit switch array (**SW**). The outputs include signals to drive a 7-segment display (**SEG** and **AN**), display switch states (**SWLED**), and show the current BCD values on LEDs (**BCDLED**). Internally, a *clock_divide*r and *mux32x1* module work together to generate a divided clock signal based on the selected speed from the switches.

The system uses two chained *bcd_up_down_counter* modules to represent units and tens digits. The unit counter always counts when the divided clock pulses, while the tens counter only increments or decrements when the unit counter produces a carry. This allows the system to count from 00 to 99 or in reverse, depending on the state of **BTN1**. The *reset* signal (**BTN0**) clears both counters when active. The BCD outputs from both counters are wired to LEDs for debugging and also routed to the 7-segment display driver.

The *seg7_scan* module handles fast multiplexed scanning of the 7-segment display to show the units and tens values. The anode control disables the upper six digits (AN[7:2] = 6'b111111) so only the two least significant digits are used. Additional assignments connect the switch inputs directly to LEDs (**SWLED**) and display the current BCD values on **BCDLED**. This design provides a way to demonstrate BCD counting with adjustable speed and direction.

```
    .carry_pulse()
);

seg7_scan scan_driver (
    .clk_fast(CLK),
    .rst_n(rst_n),
    .units_val(unit_bcd),
    .tens_val(ten_bcd),
    .seg_pins(SEG),
    .dp(),
    .anodes(AN)
);

assign AN[7:2] = 6'b111111;
assign SWLED[4:0] = SW;
assign BCDLED[3:0] = unit_bcd;
assign BCDLED[7:4] = ten_bcd;

endmodule
```

Clock Divider Code

```
`timescale 1ns / 1ps
module clock_divider (
    input wire sys_clk,
    input wire reset_n,
    output reg [31:0] tick_count
);

always @(posedge sys_clk or negedge reset_n) begin
    if (!reset_n)
        tick_count <= 0;
    else
        tick_count <= tick_count + 1;
end

endmodule
```

The *clock_divider* module generates a 32-bit counter (**tick_count**) by incrementing it on every rising edge of the system clock (**sys_clk**). This effectively divides the clock frequency by creating slower pulses at each bit of the counter. The counter resets to zero asynchronously when **reset_n** is low. Each bit in **tick_count** can be used to derive a clock signal at a specific divided frequency, making this module useful for generating slower timing signals from a fast system clock.

MUX 32x1 Code

```
`timescale 1ns / 1ps
module mux32x1 (
    input wire [31:0] tick_bits,
    input wire [4:0] speed_select,
    output wire clk_out
);
```

The *mux32x1* module selects one of the 32 bits from the **tick_bits** input based on the 5-bit **speed_select** value and outputs it as **clk_out**. Since each bit in **tick_bits** represents a progressively slower clock (from the *clock_divider* module), this acts as a

```
assign clk_out = tick_bits[speed_select];

endmodule
```

32-to-1 multiplexer that allows the user to choose the desired clock frequency by adjusting **speed_select**. This makes it easy to dynamically control the speed of a system, such as a counter, using external inputs like switches.

<u>BCD Up/Down Counter Code</u>

```
module bcd_up_down_counter (
    input wire clk_pulse,
    input wire rst_n,
    input wire dir_up,
    input wire enable_count,
    output reg [3:0] bcd_value,
    output reg carry_pulse
);

always @(posedge clk_pulse or negedge
rst_n) begin
    if (!rst_n) begin
        bcd_value <= 0;
        carry_pulse <= 0;
    end else if (enable_count) begin
        carry_pulse <= 0;

        if (dir_up) begin
            if (bcd_value == 1) begin
                bcd_value <= 0;
                carry_pulse <= 1;
            end else if (bcd_value == 0) begin
                bcd_value <= 9;
            end else begin
                bcd_value <= bcd_value - 1;
            end
        end else begin
            if (bcd_value == 8) begin
                bcd_value <= 9;
                carry_pulse <= 1;
            end else if (bcd_value == 9) begin
                bcd_value <= 0;
            end else begin
                bcd_value <= bcd_value + 1;
            end
        end
    end
end
```

In the *bcd_up_down_counter* module we designed, we created a 4-bit counter that can count either up or down depending on the **dir_up** input. It updates on the rising edge of the clock pulse and can be asynchronously reset with **rst_n**. When counting is enabled, the counter either increments or decrements the BCD value stored in **bcd_value**, and we use **carry_pulse** to signal when the count rolls over like from 9 back to 0 or 0 down to 9.

```
endmodule
```

7 Segment Code

```
`timescale 1ns / 1ps
module seg7_scan (
    input wire clk_fast,
    input wire rst_n,
    input wire [3:0] units_val,
    input wire [3:0] tens_val,
    output reg [6:0] seg_pins,
    output wire dp,
    output reg [1:0] anodes
);

reg [15:0] refresh_timer;
wire current_digit = refresh_timer[13];

assign dp = 1'b1;

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

always @(*) begin
    case (current_digit)
        1'b0: begin
            anodes = 2'b10;
            case (units_val)
                4'd0: seg_pins = 7'b1000000;
                4'd1: seg_pins = 7'b1001111;
                4'd2: seg_pins = 7'b0100100;
                4'd3: seg_pins = 7'b0110000;
                4'd4: seg_pins = 7'b0011001;
                4'd5: seg_pins = 7'b0010010;
                4'd6: seg_pins = 7'b0000010;
                4'd7: seg_pins = 7'b1111000;
                4'd8: seg_pins = 7'b0000000;
                4'd9: seg_pins = 7'b0010000;
                default: seg_pins = 7'b1111111;
            endcase
        end
        1'b1: begin
            anodes = 2'b01;
            case (tens_val)
```

This Verilog module, *seg7_scan*, is responsible for driving a two-digit 7-segment display using time-multiplexing. It receives a high-frequency clock (**clk_fast**) and a reset signal (**rst_n**), along with 4-bit BCD inputs representing the units (**units_val**) and tens (**tens_val**) digits. It outputs 7 segment control lines (**seg_pins**), a decimal point (**dp**, which is disabled), and anode control lines (**anodes**) to select which digit is currently active.

The module uses a 16-bit **refresh_timer** to alternate between displaying the units and tens digits at a rate determined by bit 13 of the timer. This bit acts as a simple frequency divider to toggle the active digit approximately every few milliseconds, creating the illusion that both digits are lit continuously to the human eye. When **current_digit** is 0, the module displays the units digit and activates the corresponding anode (2'b10) when 1, it shows the tens digit and activates its anode (2'b01).

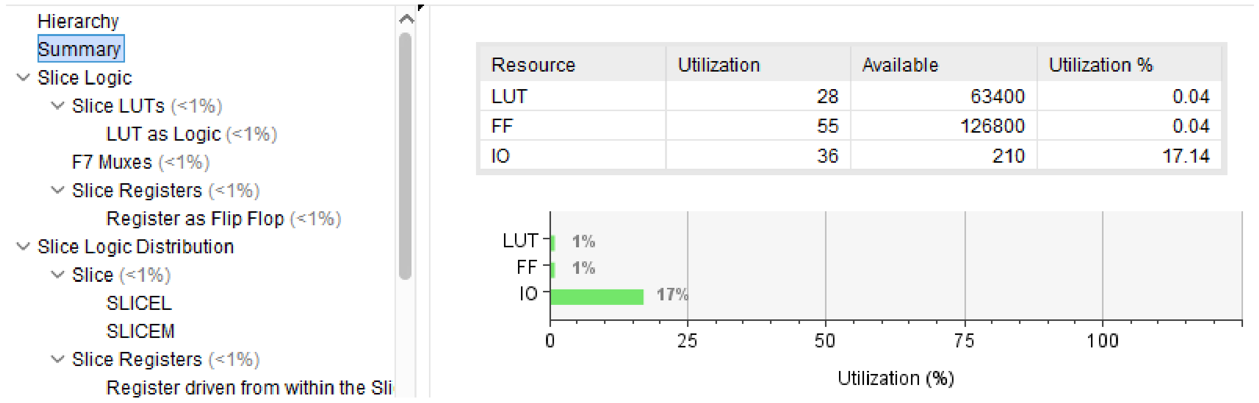
Each digit is converted from its 4-bit BCD value to a corresponding 7-segment encoding using a case statement. If the value is outside 0–9, the display shows blank (7'b1111111). This design allows efficient scanning of two 7-segment digits using minimal pins and ensures flicker-free operation using the fast input clock and timer-based digit switching.

```
            4'd0: seg_pins = 7'b1000000;
            4'd1: seg_pins = 7'b1001111;
            4'd2: seg_pins = 7'b0100100;
            4'd3: seg_pins = 7'b0110000;
            4'd4: seg_pins = 7'b0011001;
            4'd5: seg_pins = 7'b0010010;
            4'd6: seg_pins = 7'b0000010;
            4'd7: seg_pins = 7'b1111000;
            4'd8: seg_pins = 7'b0000000;
            4'd9: seg_pins = 7'b0010000;
            default: seg_pins = 7'b1111111;
         endcase
      end
   endcase
end

endmodule
```

Screenshot Proofs:

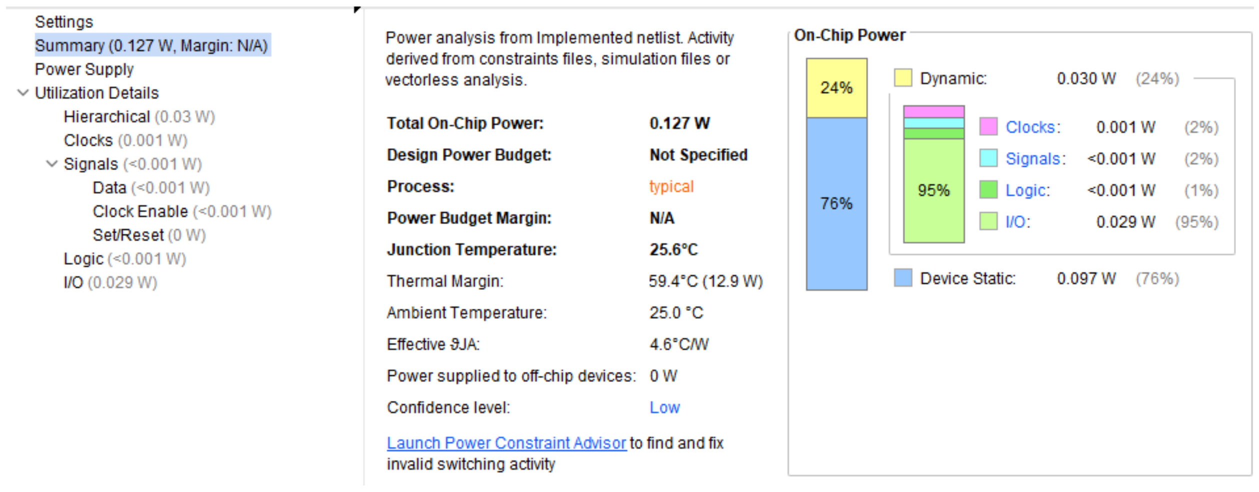
Resource Utilization Table:



Hierarchy
Summary
Slice Logic
Slice LUTs (<1%)
LUT as Logic (<1%)
F7 Muxes (<1%)
Slice Registers (<1%)
Register as Flip Flop (<1%)
Slice Logic Distribution
Slice (<1%)
SLICEL
SLICEM
Slice Registers (<1%)
Register driven from within the Sli

Resource	Utilization	Available	Utilization %
LUT	28	63400	0.04
FF	55	126800	0.04
IO	36	210	17.14

Power Utilization:



Settings
Summary (0.127 W, Margin: N/A)
Power Supply
Utilization Details
Hierarchical (0.03 W)
Clocks (0.001 W)
Signals (<0.001 W)
Data (<0.001 W)
Clock Enable (<0.001 W)
Set/Reset (0 W)
Logic (<0.001 W)
I/O (0.029 W)

Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

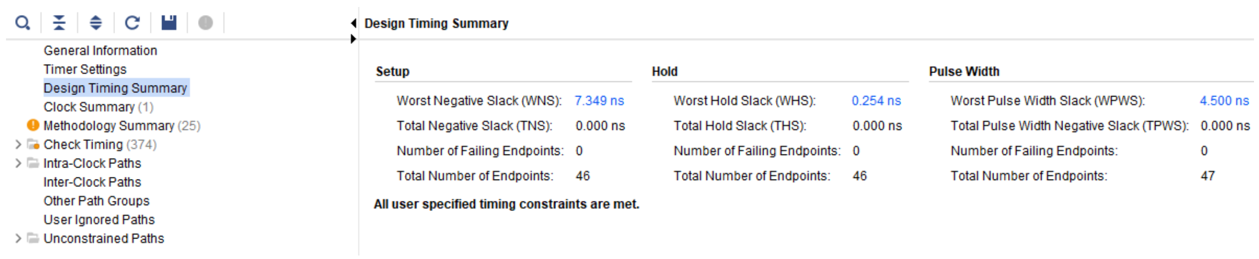
Total On-Chip Power:	**0.127 W**
Design Power Budget:	**Not Specified**
Process:	typical
Power Budget Margin:	**N/A**
Junction Temperature:	**25.6°C**
Thermal Margin:	59.4°C (12.9 W)
Ambient Temperature:	25.0 °C
Effective ϑJA:	4.6°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Launch Power Constraint Advisor to find and fix invalid switching activity

On-Chip Power

Dynamic:		0.030 W	(24%)
	Clocks:	0.001 W	(2%)
	Signals:	<0.001 W	(2%)
	Logic:	<0.001 W	(1%)
	I/O:	0.029 W	(95%)
Device Static:		0.097 W	(76%)

Timing Summary:



General Information
Timer Settings
Design Timing Summary
Clock Summary (1)
Methodology Summary (25)
Check Timing (374)
Intra-Clock Paths
Inter-Clock Paths
Other Path Groups
User Ignored Paths
Unconstrained Paths

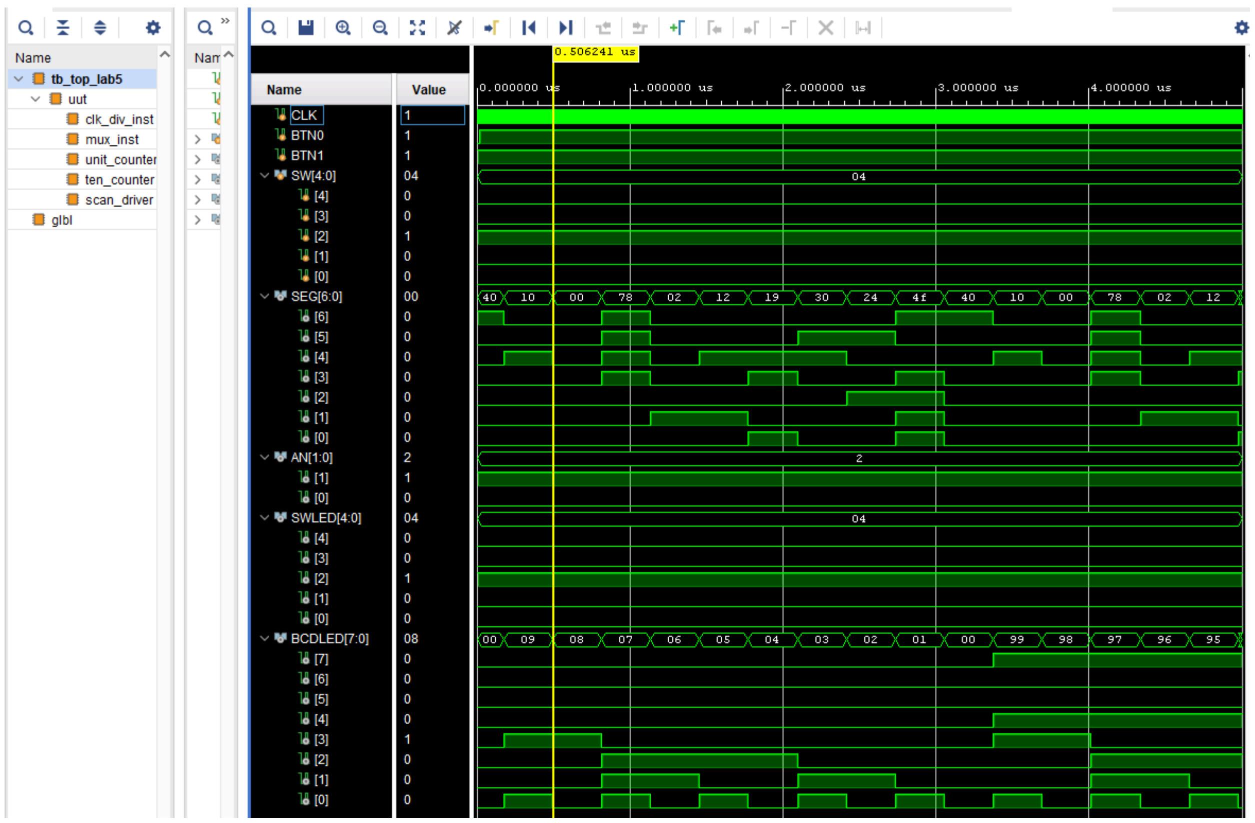
Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	7.349 ns	Worst Hold Slack (WHS):	0.254 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	46	Total Number of Endpoints:	46	Total Number of Endpoints:	47

All user specified timing constraints are met.

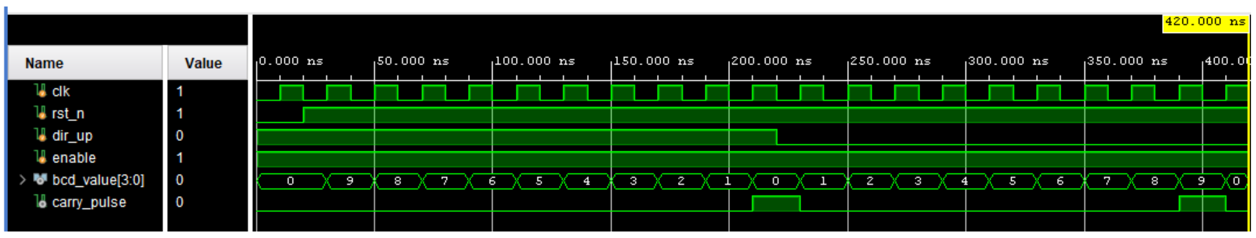
Simulation Waveform:

BCD Up/Down Counter on 7-Segment Display Full Waveform:



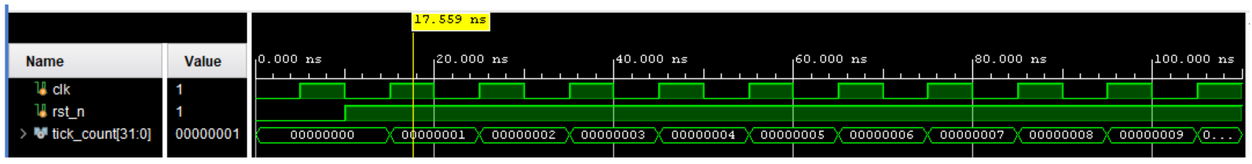
Name	Value
CLK	1
BTN0	1
BTN1	1
SW[4:0]	04
[4]	0
[3]	0
[2]	1
[1]	0
[0]	0
SEG[6:0]	00
[6]	0
[5]	0
[4]	0
[3]	0
[2]	0
[1]	0
[0]	0
AN[1:0]	2
[1]	1
[0]	0
SWLED[4:0]	04
[4]	0
[3]	0
[2]	1
[1]	0
[0]	0
BCDLED[7:0]	08
[7]	0
[6]	0
[5]	0
[4]	0
[3]	1
[2]	0
[1]	0
[0]	0

BCD Up/Down Counter Module Waveform:



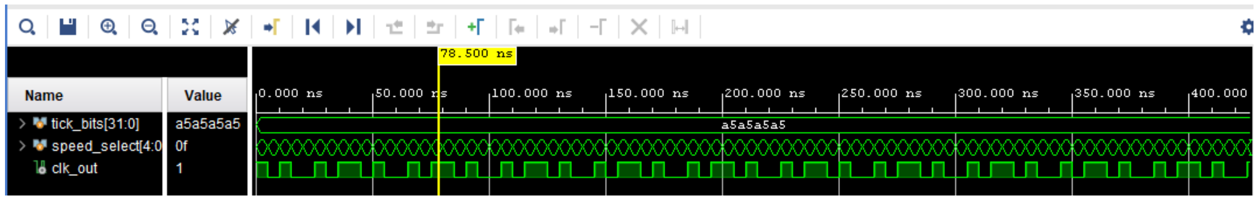
Name	Value
clk	1
rst_n	1
dir_up	0
enable	1
bcd_value[3:0]	0
carry_pulse	0

Clock Divider Module Waveform:

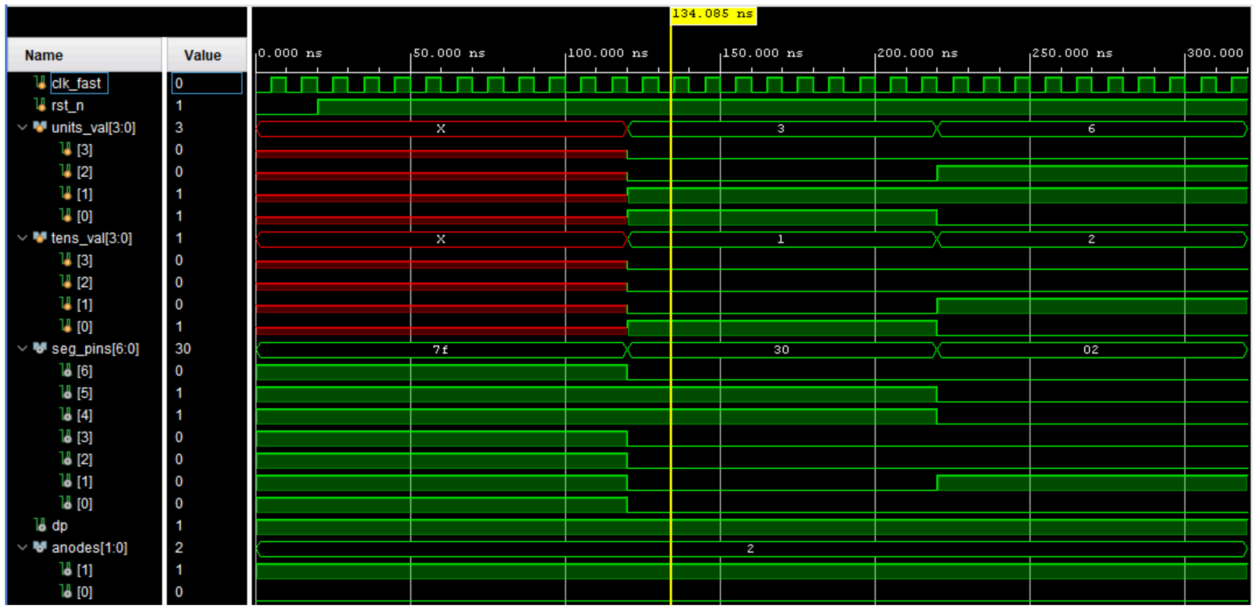


Name	Value
clk	1
rst_n	1
tick_count[31:0]	00000001

MUX 32x1 Module Waveform:



7 Segment Module Waveform:



Partner Contributions:

Team Member	**Contribution**	**% Effort**
Jonathan Huynh	**Code:** [Clock Divide, BCD Up/Down Counter, 32x1 MUX] **Testbench:** [Top Module and 32x1 MUX] **Additional:** Synthesis/Implementation and Demo	50%
Adam Godfrey	**Code:** [7 Segment and Top Module] **Testbench:** [7 Segment, BCD Up/Down, Clock Divider] **Additional:** Simulation and Lab Report	50%