LAB4 - 2 digit counter using 2 seven-seg-displays

Group A :

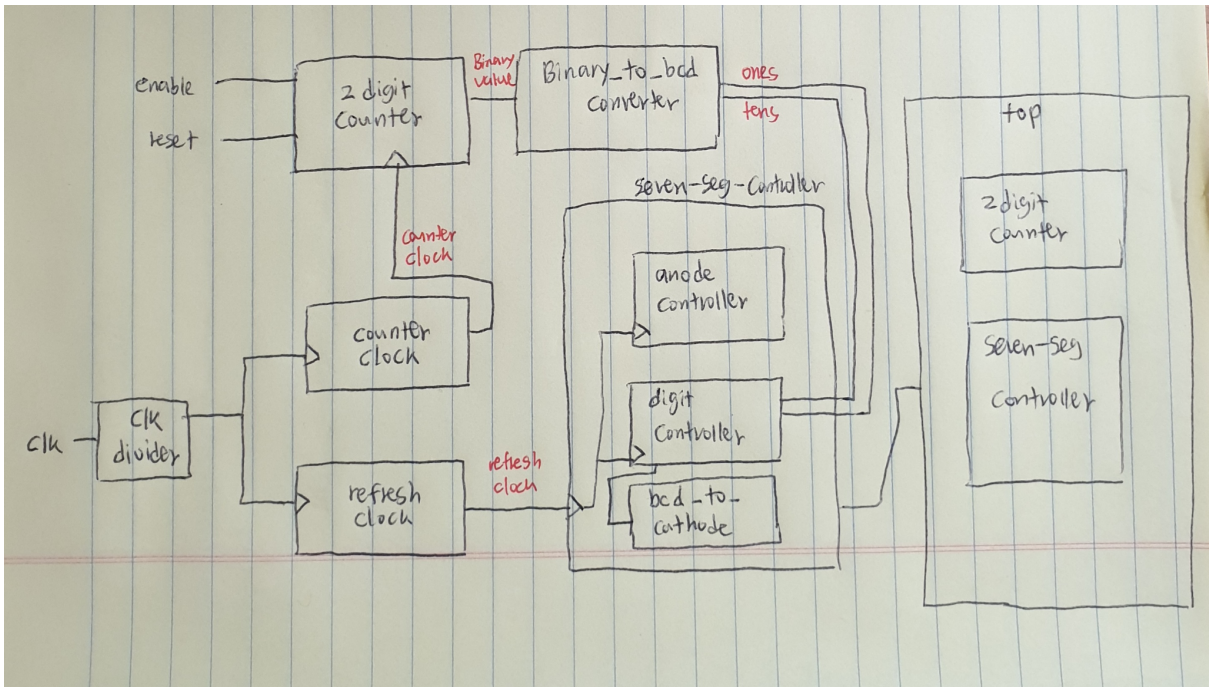
- **Jihun Choi(Bronco ID#: 011841206)**
- **Jacky Li(Bronco ID#:014181245)**

Prof.Aly

ECE3300L.02 - Summer 2022

This lab4 is about designing a 2 digit counter (0~99) using 2 seven-seg-displays by instantiation using vivado software and test the results with FPGA board.

reference: https://www.youtube.com/watch?v=s4lPOQ1VAkU&t=362s



1) we create clock_divider module to divide clock for seven segment display

(input clk, output divided_clk)

2) we create refreshcounter module to have separate counter to be used in anode controller and digit controller

(input refresh_clock, output refreshcounter)

3) we create anode_controller module to control which 7 sevenment to be turned on by using refreshcounter

(input refreshcounter, output [7:0] AN since there are 8 seven seg available, but we are only using 2)

4) we create digit_controller module to control which locations of digit(one's or ten's) to show by using refrescounter

(input refreshcounter, [3:0] ones, [3:0] tens, output [3:0] digit)

5) we create binary_to_bcd_converter module to convert binary number to bcd format using idea of shift

(input [7:0] number, output [3:0] ones, output [3:0] tens)

refrence:

https://www.google.com/search?q=binary+to+bcd+shift+add+3&rlz=1C1CHZN_koKR958KR958&oq=&aqs=chrome.3.69i59i450l8.43413477j0j15&sourceid=chrome&ie=UTF-8#kpvalbx=_MATAYp-fNKDVkPIPiLybOA16

6) we create bcd_to_cathode module to connect bcd number (0~9) to corresponding cathode for seven segment display

(input [3:0] digit, output reg [6:0] cathode)

7) we create two_digit_counter module to write function of the main idea (reset and enable to pause)

(input clk, enable, reset , output [7:0] counter)

8) we create seven_seg_controller to connect refresh counter module to anode_controller and digit_controller.

Then, connect digit_controller with bcd_to_cathode

(input refresh_clock, input [3:0] ones, tens, output [7:0] AN, output [6:0] cathode)

9) we create top module to connect two_digit_counter to be connected with other modules

10) we create a top.xdc file and connect the correct ports.

11) we performed the simulation,

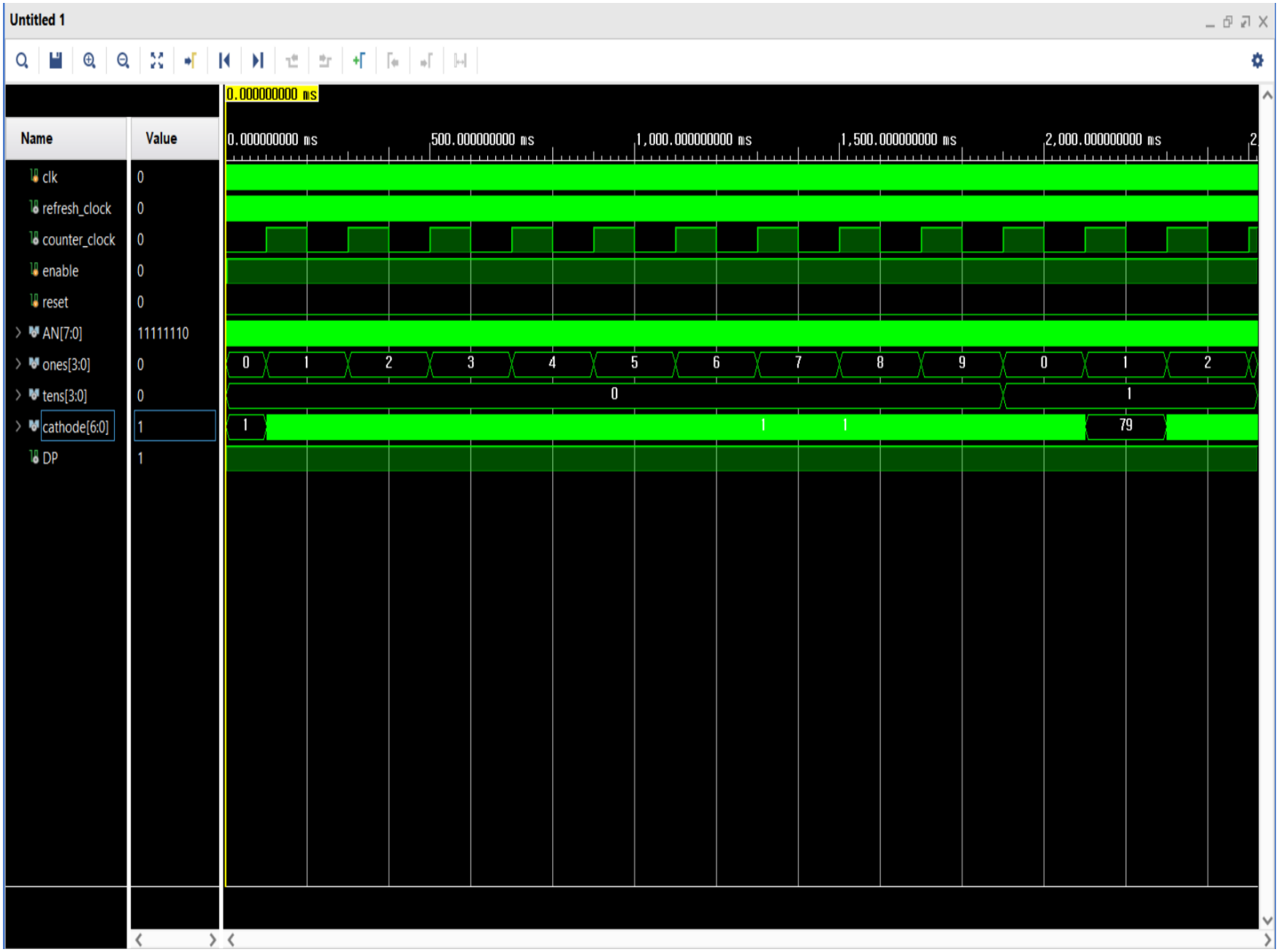
```
module top_tb();

                reg clk = 0;
                reg enable =0;
                reg reset = 0;
                wire [7:0] AN;
                wire [6:0] cathode;
                wire DP=1;


                top top1(
                                .clk(clk),
                                .enable(enable),
                                .reset(reset),
                                .AN(AN),
                                .cathode(cathode),
                                .DP(DP)
                        );

                always #5 clk = ~clk;
                initial
                   begin
                        #500
                             enable =1;
                   end


endmodule
```


Untitled 1
0.000000000 ms
Name
Value
clk
0
refresh_clock
0
counter_clock
0
enable
0
reset
0
AN[7:0]
11111110
ones[3:0]
0
tens[3:0]
0
cathode[6:0]
1
DP
1
0.000000000 ms
500.000000000 ms
1,000.000000000 ms
1,500.000000000 ms
2,000.000000000 ms
0 1 2 3 4 5 6 7 8 9 0 1 2
0 1
1 79

Simulation video was not included in the demo video, since the laptop that was testing the simulation was not able to handle for running it for 1 seconds. So, only a screen shot was provided.

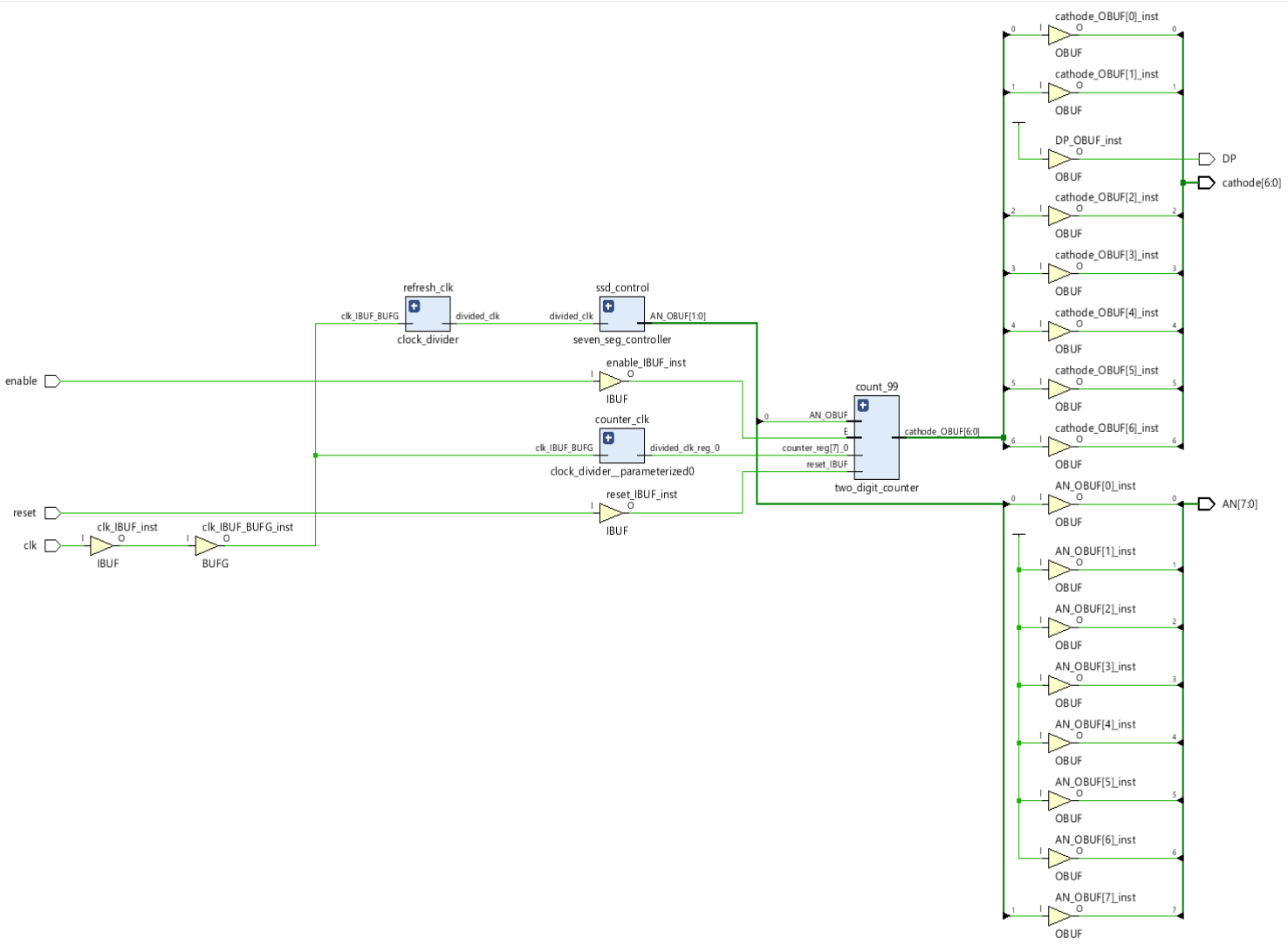
Video link:

https://youtu.be/2OLKznhubEw

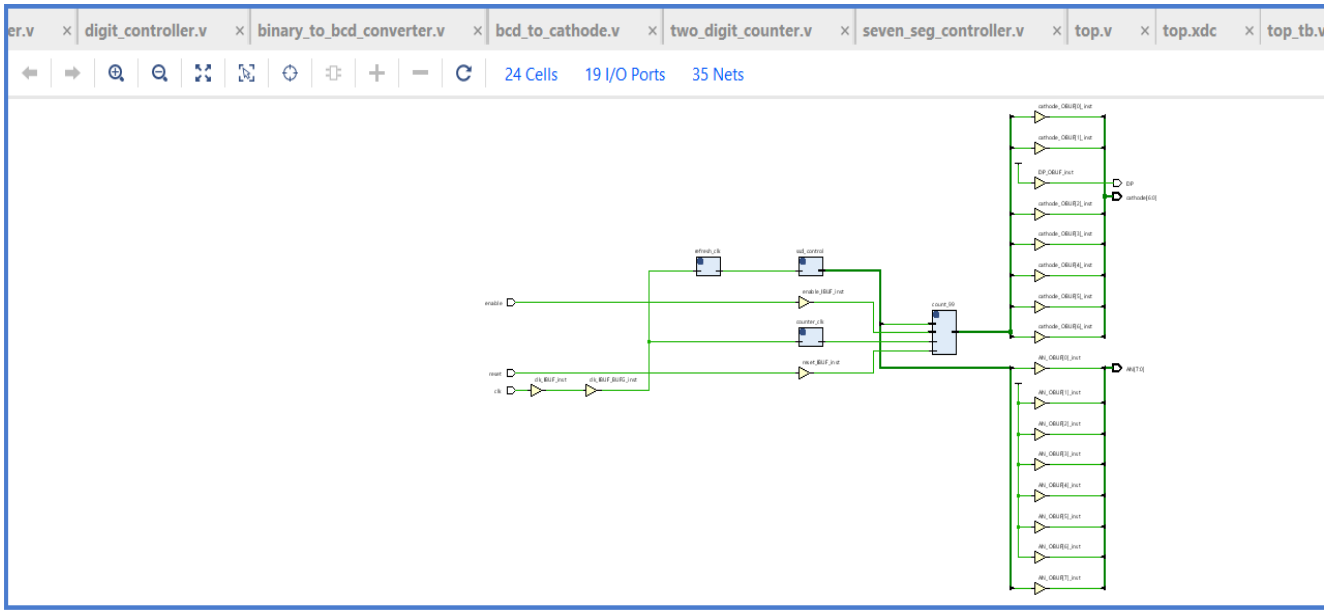
or

https://drive.google.com/file/d/1CL9M76JCyxCSSrKhnL69roycP3X1VjHo/view?usp=sharing

Elaborated Design:



Utilization:



Tcl Console | Messages | Log | Reports | Design Runs | Power | DRC | Methodology | Timing | **Utilization**

Hierarchy

- Hierarchy
- Summary
- Slice Logic
  - Slice LUTs (<1%)
    - LUT as Logic (<1%)
  - Slice Registers (<1%)
    - Register as Flip Flop (<
- Slice Logic Distribution

Name	Slice LUTs (63400)	Slice Registers (126800)	Slice (15850)	LUT as Logic (63400)	Bonded IOB (210)	BUFGCTRL (32)
top	43	75	38	43	19	1
count_99 (two_digit_counter)	22	8	8	22	0	0
counter_clk (clock_divider__parameterized0)	10	33	16	10	0	0
refresh_clk (clock_divider)	10	33	16	10	0	0
ssd_control (seven_seg_controller)	1	1	2	1	0	0

Power report:

Tcl Console | Messages | Log | Reports | Design Runs | **Power** | DRC | Methodology | Timing | Utilization

Summary

- Settings
- Summary (0.124 W, Margin: N/A)
- Power Supply
- Utilization Details
  - Hierarchical (0.027 W)
  - Clocks (0.001 W)
  - Signals (0.001 W)
    - Data (0.001 W)
    - Clock Enable (<0.001 W)
    - Set/Reset (<0.001 W)
  - Logic (<0.001 W)
  - I/O (0.025 W)

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

Total On-Chip Power:	**0.124 W**
Design Power Budget:	**Not Specified**
Power Budget Margin:	**N/A**
Junction Temperature:	**25.6°C**
Thermal Margin:	59.4°C (12.9 W)
Effective ϑJA:	4.6°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Medium

Launch Power Constraint Advisor to find and fix invalid switching activity