Calvin Truong
ECE 3300L-03
Dr. Nazari
9/24/18

Pre-Lab 4

1)
clk_gen.v

```
1   `timescale 1ns / 1ps
2   //////////////////////////////////////////////////////////////////////////////////
3   // Module:clk_gen
4   // Engineer: Calvin Truong
5   //
6
7   module clk_gen(Clk100MHz, reset_n, Clk1KHzEn);
8       input Clk100MHz, reset_n;
9       output Clk1KHzEn;
10
11      localparam MAX_COUNT = 9;
12      reg Clk1KHzEn = 0;
13      reg [16:0] cnt = 0;
14
15      always @(posedge Clk100MHz)
16          begin
17              if (!reset_n)
18                  cnt <= 0;
19              else if (cnt == MAX_COUNT)
20                  cnt <= 0;
21              else
22                  cnt <= cnt + 1;
23          end
24
25      always @(posedge Clk100MHz)
26          begin
27              if(!reset_n)
28                  Clk1KHzEn <= 0;
29              else if (cnt == MAX_COUNT)
30                  Clk1KHzEn <= 1;
31              else
32                  Clk1KHzEn <= 0;
33          end
34  endmodule
```

2)
data_gen.v

```
1  `timescale 1ns / 1ps
2  //////////////////////////////////////////////////////////////////////////////////
3  // Module:data_gen
4  // Engineer: Calvin Truong
5  //
6
7  module data_gen(X, Y, Data, XDP);
8      input [3:0] X;
9      input [1:0] Y;
10     output reg [31:0] Data;
11     output reg [7:0] XDP;
12
13     always @ (X or Y)
14         begin
15             Data = {28'hECE3300,X};
16             XDP = {6'b010101, Y};
17         end
18 endmodule
```

3)

counter.v

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Module:counter
// Engineer: Calvin Truong
//

module counter(Clk100MHz, Clk1KHzEn, reset_n, NumDigits, cnt);
    input Clk100MHz, Clk1KHzEn, reset_n;
    input [2:0] NumDigits;
    output reg [2:0] cnt = 0;

    always @ (posedge Clk1KHzEn)
        begin
            if(!reset_n)
                cnt <= 0;
            else if(NumDigits == 0)
                cnt <= 3'b000;
            else if(cnt == NumDigits)
                cnt <= 3'b000;
            else if (NumDigits > 1)
                cnt <= cnt + 1;
        end
endmodule
```

seven_seg_display.v

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Module: seven_seg_display
// Engineer: Calvin Truong
//

module seven_seg_display(Clk100MHz, Clk1KHzEn, NumDigits, Data, XDP, reset_n, CA, CB, CC, CD, CE, CF, CG, DP, AN);
    input Clk100MHz, Clk1KHzEn, reset_n;
    input [2:0] NumDigits;
    input [31:0] Data;
    input [7:0] XDP;
    output CA, CB, CC, CD, CE, CF, CG;
    output reg DP;
    output [7:0] AN;

    wire [2:0] DigitSelect;
    wire[6:0] wireDec;
    reg [3:0] D;

    //module counter(clk100MHz, clk1KHzEn, reset_n, NumDigits, count);
    counter u_counter(.Clk100MHz(Clk100MHz),
                      .Clk1KHzEn(Clk1KHzEn),
                      .reset_n(reset_n),
```

```
                        .NumDigits(NumDigits),
                        .cnt(DigitSelect)
                        );

    //module decoder_3_8(X, En, Yout);
    decoder_3_8 u_decoder_3_8(.X(DigitSelect),
                              .En(1'b1),
                              .Yout(AN)
                              );

    //multiplexer with the block
    always @ (posedge Clk100MHz)
        begin
            case (DigitSelect)
                3'b000: {DP,D} <= {XDP[0], Data[3:0]};
                3'b001: {DP,D} <= {XDP[1], Data[7:4]};
                3'b010: {DP,D} <= {XDP[2], Data[11:8]};
                3'b011: {DP,D} <= {XDP[3], Data[15:12]};
                3'b100: {DP,D} <= {XDP[4], Data[19:16]};
                3'b101: {DP,D} <= {XDP[5], Data[23:20]};
                3'b110: {DP,D} <= {XDP[6], Data[27:24]};
                3'b111: {DP,D} <= {XDP[7], Data[31:28]};
            endcase
        end

    seven_segment_decoder u_seven_segment_decoder(.X(D),
                                                  .Dec(wireDec)
                                                  );
    assign CA = wireDec[6];
    assign CB = wireDec[5];
    assign CC = wireDec[4];
    assign CD = wireDec[3];
    assign CE = wireDec[2];
    assign CF = wireDec[1];
    assign CG = wireDec[0];
endmodule
```

4)

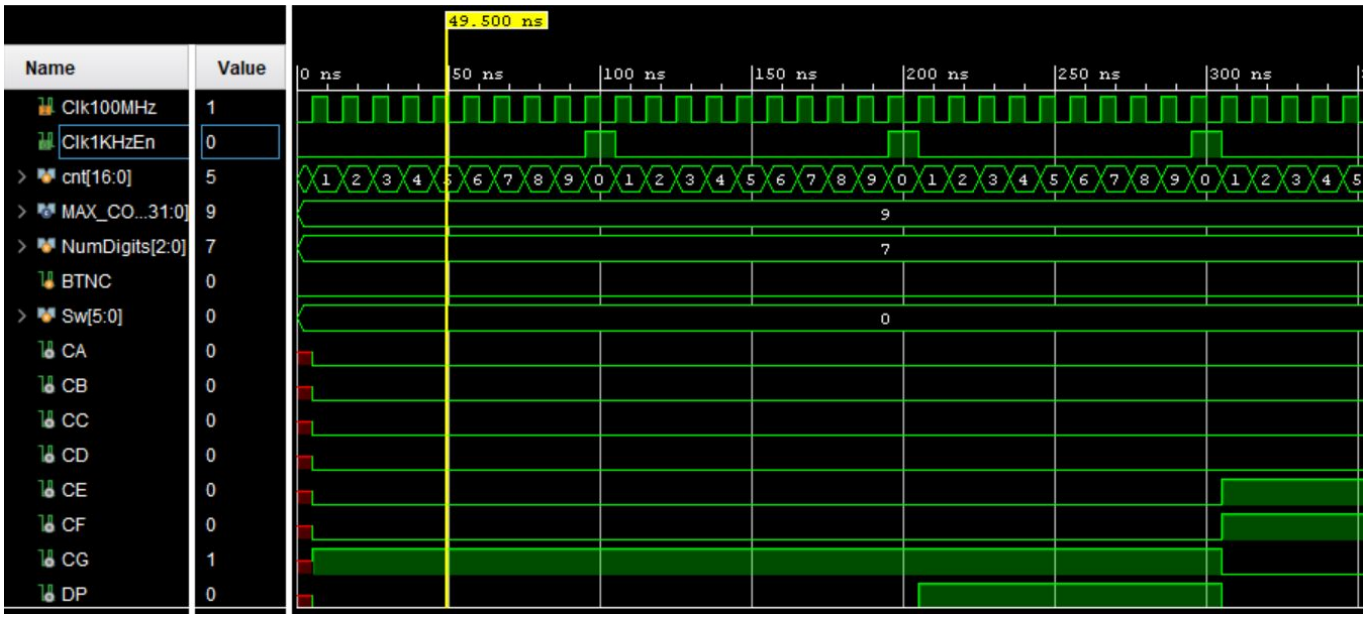
Lab4_top.v

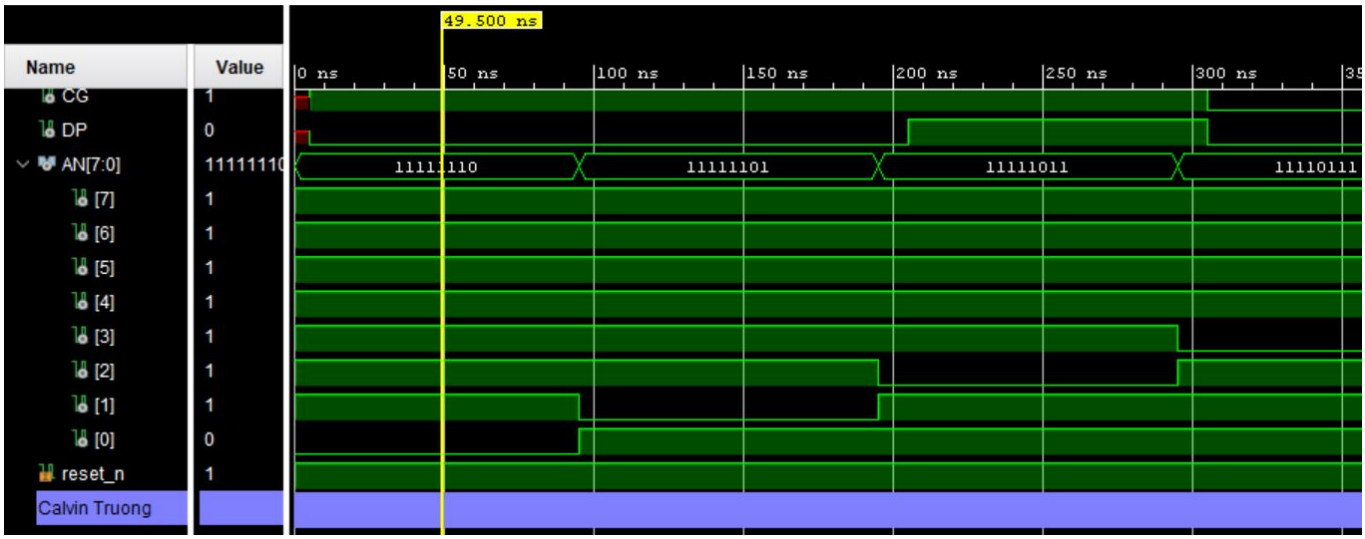
```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Module:lab4_top
// Engineer: Calvin Truong
//

module lab4_top(Clk100MHz, NumDigits, Sw, BTNC, CA, CB, CC, CD, CE, CF, CG, DP, AN);
    input Clk100MHz, BTNC;
    input [2:0] NumDigits;
    input [5:0] Sw;
    output CA,CB,CC,CD,CE,CF,CG,DP;
    output [7:0] AN;
    wire reset_n;
    wire Clk1KHzEn;
    wire [7:0] XDP;
    wire [31:0] Data;

    not (reset_n,BTNC);

    clk_gen u_clk_gen(.Clk100MHz(Clk100MHz),
                      .reset_n(reset_n),
                      .Clk1KHzEn(Clk1KHzEn)
                     );

    data_gen u_data_gen(.X(Sw[3:0]),
                        .Y(Sw[5:4]),
                        .Data(Data),
                        .XDP(XDP)
                        );

    seven_seg_display u_seven_seg_display(.Clk100MHz(Clk100MHz),
                                          .Clk1KHzEn(Clk1KHzEn),
                                          .NumDigits(NumDigits),
                                          .Data(Data),
                                          .XDP(XDP),
                                          .reset_n(reset_n),
                                          .CA(CA),
                                          .CB(CB),
                                          .CC(CC),
                                          .CD(CD),
                                          .CE(CE),
                                          .CF(CF),
                                          .CG(CG),
                                          .DP(DP),
                                          .AN(AN)
                                          );
    assign reset_n = ~BTNC;
endmodule
```

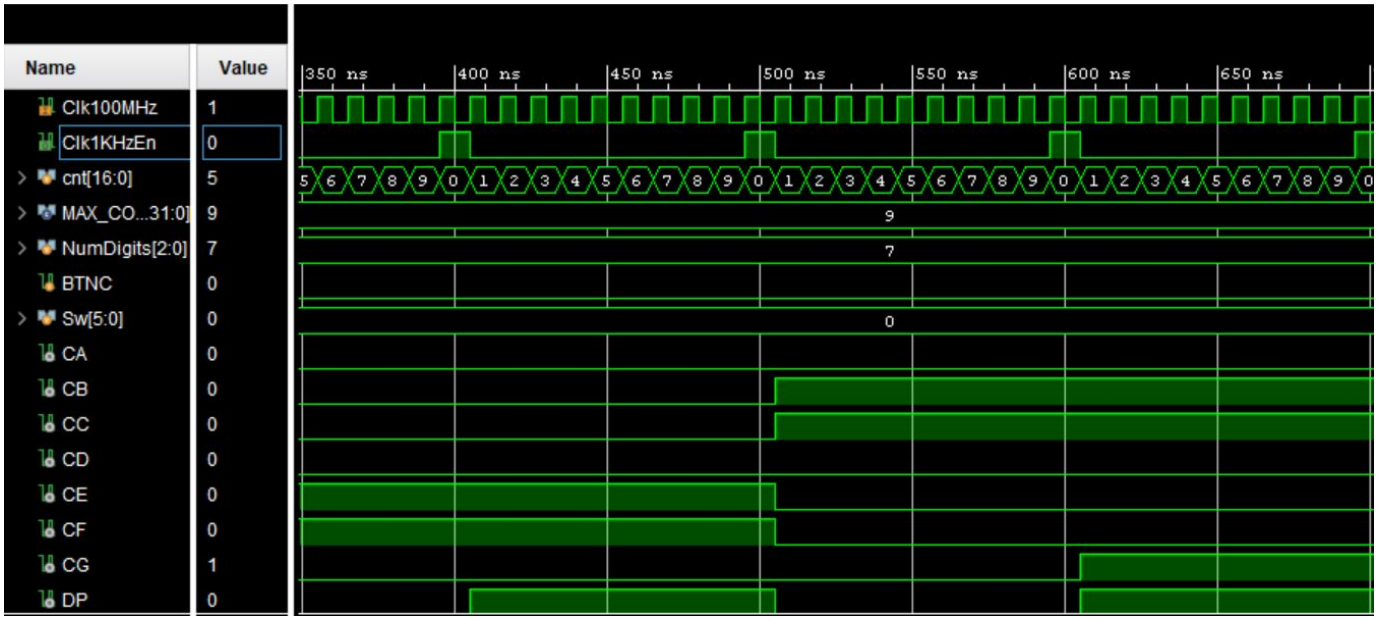
5)
Lab4_tb.v

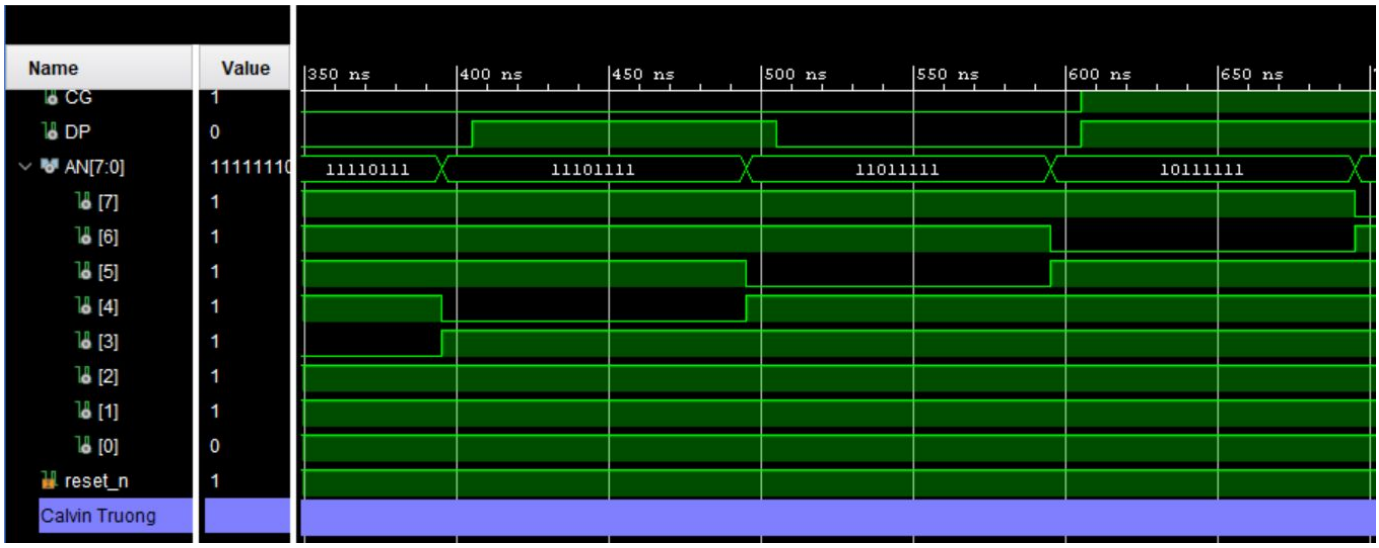
```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Module: lab4_tb
// Engineer: Calvin Truong
//

module lab4_tb();
    reg Clk100MHz, BTNC;
    reg [2:0] NumDigits;
    reg [5:0] Sw;

    wire CA,CB,CC,CD,CE,CF,CG,DP;
    wire [7:0] AN;

    lab4_top u_lab4_top (.Clk100MHz(Clk100MHz),
                         .NumDigits(NumDigits),
                         .Sw(Sw),
                         .BTNC(BTNC),
                         .CA(CA),
                         .CB(CB),
                         .CC(CC),
                         .CD(CD),
                        .CE(CE),
                        .CF(CF),
                        .CG(CG),
                        .DP(DP),
                        .AN(AN)
                        );

    initial
        begin
            Clk100MHz = 0;
            BTNC = 0;
            NumDigits = 7;
            Sw = 0;
        end
    always #5 Clk100MHz = ~Clk100MHz;
endmodule
```

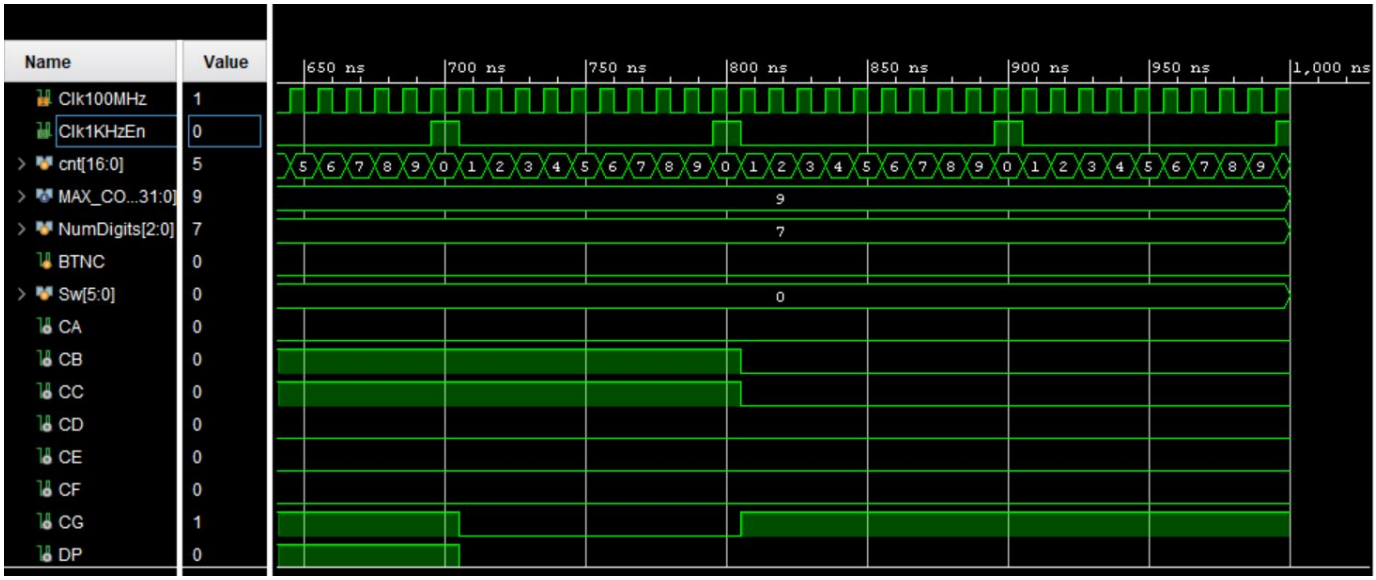
6) Simulation Result

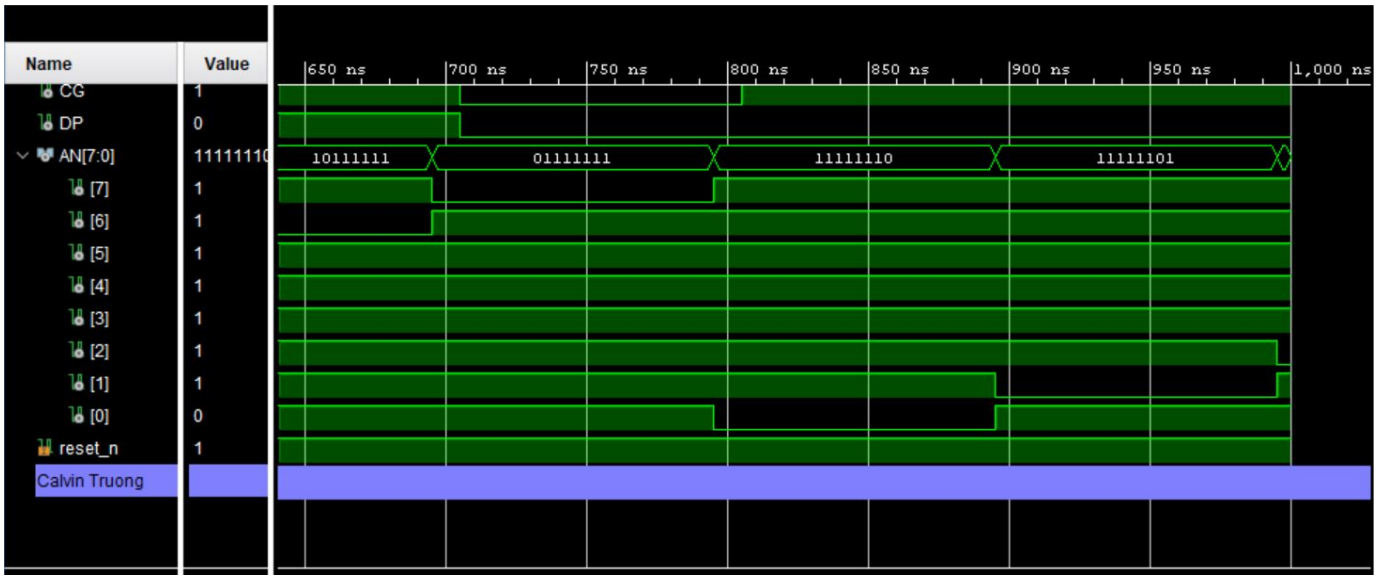





Name
Value
Clk100MHz
Clk1KHzEn
cnt[16:0]
MAX_CO...31:0]
NumDigits[2:0]
BTNC
Sw[5:0]
CA
CB
CC
CD
CE
CF
CG
DP
1
0
5
9
7
0
0
0
0
0
0
0
1
0
350 ns
400 ns
450 ns
500 ns
550 ns
600 ns
650 ns


Name
Value
CG
DP
AN[7:0]
[7]
[6]
[5]
[4]
[3]
[2]
[1]
[0]
reset_n
Calvin Truong
1
0
11111111
1
1
1
1
1
1
1
0
1
11110111
11101111
11011111
10111111
350 ns
400 ns
450 ns
500 ns
550 ns
600 ns
650 ns


Name
Value
650 ns
700 ns
750 ns
800 ns
850 ns
900 ns
950 ns
1,000 ns
Clk100MHz
1
Clk1KHzEn
0
cnt[16:0]
5
MAX_CO...31:0]
9
NumDigits[2:0]
7
BTNC
0
Sw[5:0]
0
CA
0
CB
0
CC
0
CD
0
CE
0
CF
0
CG
1
DP
0


Name
Value
650 ns
700 ns
750 ns
800 ns
850 ns
900 ns
950 ns
1,000 ns
CG
1
DP
0
AN[7:0]
11111110
10111111
01111111
11111110
11111101
[7]
1
[6]
1
[5]
1
[4]
1
[3]
1
[2]
1
[1]
1
[0]
0
reset_n
1
Calvin Truong