COMPEX ENGINEERING ACTIVITY

CS_431 DIGITAL SYSTEM DESIGN
LAB SESSION 14

DESIGN PROJECT

SUBMITTED TO: MISS SYEDA RAMISH FATIMA

SUBMITTED BY:

CS_18068 MAHAM ZAKI
CS_18114 AFNAN BAIG
CS_18139 MUHAMMAD ZAIN-UL-ABDIN SIDDIQUI

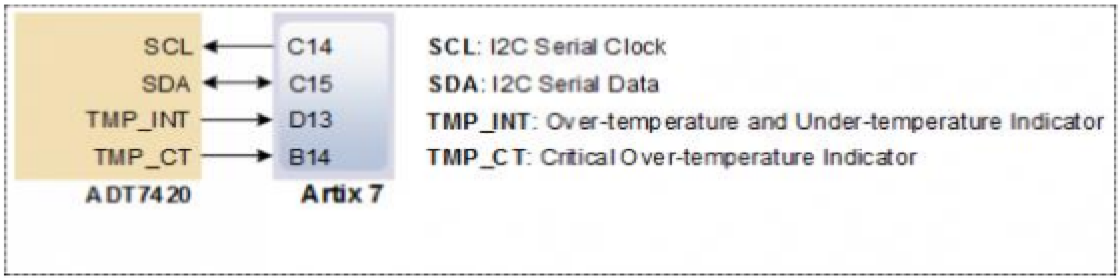
PROBLEM STATEMENT

Design a digital system incorporating FPGA-based realization for a temperature sensor data-based intensity control and color-code using tri-color LED's. Intensity control shall be designed using Pulse Width Modulation (PWM) technique. You are required to take temperature sensor data as an input and incorporate Pulse Width Modulation (PWM) technique to control the intensity level of tri-color LED (along with color coding) for different levels of temperature sensor data. The temperature sensor requires serial communication (I2C protocol) to access its data. The system shall also display the temperature in Celsius or Fahrenheit or both on the time-multiplexed seven segment displays interfaced with FPGA on the FPGA board. Alternatively, you can display the temperature on a monitor interfaced with FPGA through VGA port available on the board. You will be required to exhibit simulation of pulse width modulation as well. Maintain a hierarchical structure of your design with separate modules for different functionalities. Your design should be robust enough to handle any temperature level. The design can be enhanced as per the vision of the design team, however, keeping the basic functionality and requirements in perspective.

METHODOLOGY

The system we design is sensing the temperature and showing the temperature in celsius on 7 segment delays , moreover for different ranges of temperature we did color coding for instance for temperature between 0 to 20 celsius cyan color should be display on RGB LED.Another functionality that has been added to system is intensity control , for an range of temperature for example 40 to 60 the green color rgb should be glow but the intensity of green color light for 40 celsius should be far less than intensity of 60 celsius temperature.

First of all, we uses the temperature sensor “ADT7420” on “Nexys A7” to sense the temperature ,this module onFPGA sense the temperature using I2C protocol.The ADT7420 chip acts as a slave device using the industry standard I2C communication scheme. To communicate with ADT7420 chip, the I2C master must specify a slave address (0x4B) and a flag indicating whether the communication is a read (1) or a write (0). Once specifications are made for communication, a data transfer takes place.The interface between the temperature sensor and FPGA is shown as


SCL
SDA
TMP_INT
TMP_CT
ADT7420
C14
C15
D13
B14
Artix 7
SCL: I2C Serial Clock
SDA: I2C Serial Data
TMP_INT: Over-temperature and Under-temperature Indicator
TMP_CT: Critical Over-temperature Indicator

Secondly we display the temperature on our 7 segment displays, in total we have used 5 7-segment displays. 1st 7-segment is used for displaying the sign of the temperature , next 2 7-segment are used to display the 2 digit temperature , and last 2 7-segment are used to display “ ‘ “ and “C” to indicate degree celsius.Our 7-segment are common anode so to glow an 7-segment we provide an “0” or active low logic.

As we want to continuously display our temperature on 7-segment display which is only possible by toggling this on a very fast speed so that human eye can’t perceive it as toggling rather as continuous display.For this purpose we created an module “counter “ and use this counter bits to continuously glow 7-segments.

Next came the logic of what should be display on 7-segments, so for fourth and fifth LED it is constant which is “ ‘ “ and “C” to indicate degree celsius. The first led is to indicate the sign of temperature and next 2 leds display depend on the temperature that is being sensed.

After this we color coded our temperatures for instance if the temperature is above 80 degree the red color should glow and if temperature is between 40 to 60 degree celsius the green color should glow.this work has been done by providing different color combinations to “R”, “G”,”B” outputs.

Next and the last thing is intensity control which is achieved by Pulse Width Modulation.we use PWM to control intently of our RGB Led for example 40 to 60 the green color rgb should be glow but the intensity of green color light for 40 celsius should be far less then intensity of 60 celsius temperature.Our PWM module based on “duty_cycle” input generate an output signal PWM_out which is only high for given duty_cycle and low other wise.This system is also been implemented to FPGA:

Introduction:

- ***About FPGA?***

  A field-programmable gate array (FPGA) is an integrated circuit (IC) that can be programmed in the field after manufacture. FPGAs are similar in principle to, but hav vastly wider potential application than, programmable read-only memory (PROM) chips. FPGAs are used by engineers in the design of specialized ICs that can later be produced hard-wired in large quantities for distribution to computer manufacturers and end users. Ultimately, FPGAs might allow computer users to tailor microprocessors to meet their own individual needs.

- ***PULSE WIDTH MODULATION (PWM):***

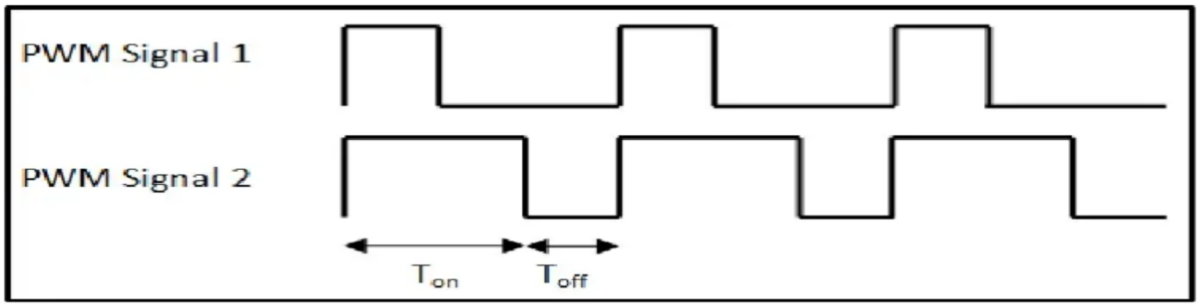
  Pulse-width modulation (PWM), or pulse-duration modulation (PDM), is a modulation technique used to encode a message into a pulsing signal. It is achieved by changing the width of the signal & keeping the time period or frequency same.
  Pulse width modulation (PWM) is a technique for controlling analog circuits with digital outputs. PWM is employed in a wide variety of applications, ranging from measurement and communications to power control and conversion.
  Pulse Width Modulation (PWM) uses digital signals to control power applications, as well as being fairly easy to convert back to analog with a minimum of hardware. Analog systems, such as linear power supplies, tend to generate a lot of heat since they are basically variable resistors carrying a lot of current. Digital systems don't generally generate as much heat. Almost all the heat generated by a switching device is during the transition (which is done quickly), while the device is neither on nor off, but in between. This is because power follows the following formula:

  $$P = E\,I$$

  , or

  $$\text{Watts} = \text{Voltage} \times \text{Current}$$

  One of the parameters of any square wave is duty cycle. Most square waves are 50%, this is the norm when discussing them, but they don't have to be symmetrical.
  The ON time can be varied completely between signal being off to being fully on, 0% to 100%, and all ranges between. Fig is giving a detailed working picture of PWM



- ***APPLICATIONS OF PWM:***

  We can control the brightness of an LED by adjusting the duty cycle.With an RGB (red green blue) LED, you can control how much of each of the three colors you want in the mix of color by dimming them with various amounts

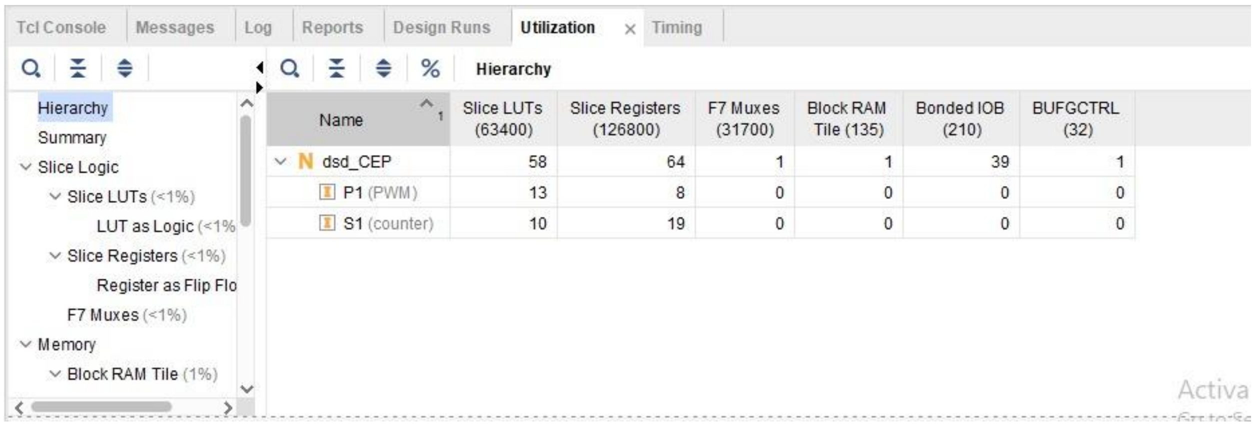
.
Modules Used::

There are a total of three modules used in this project. Which are divided as:

- ***Top module "DSD CEP":***
  This is the main module carrying all the functionalities and all other modules are instantiating in it.
- ***"PWM" module:***
  Tis module is used to generate pulse width modulation to control the intensity of RGB LED.
- ***Counter module:***
  Counter module is used to control the toggling of a 7 segment display such that it display continuous output on 7-segments.

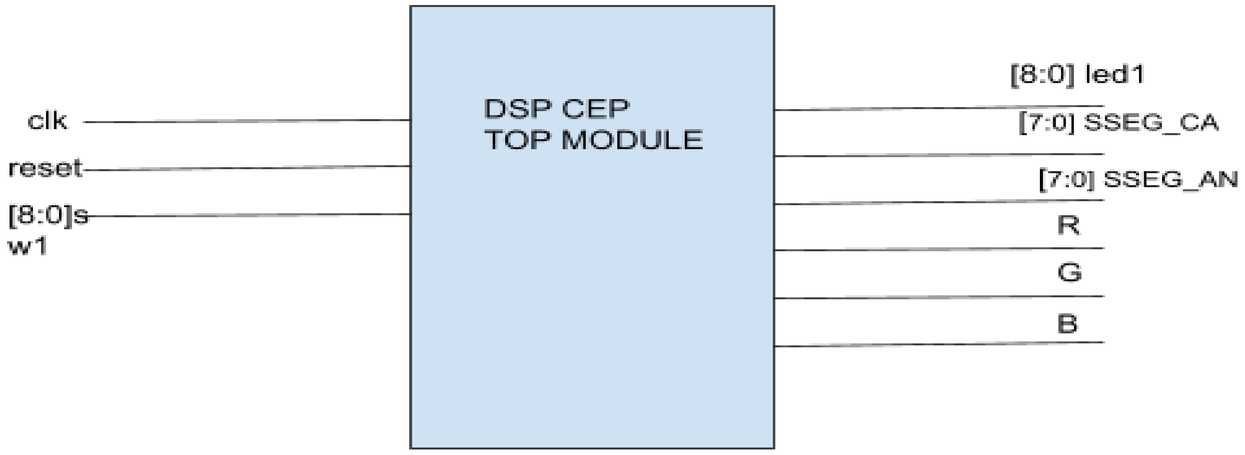
RESOURCE UTILIZATION:

Resource utilization is the count of hardware components are used for the system. The hardware may be consist of flip flops, lookup tables, muxes,IOB etc.for example in our project we use 58LUT out of 63400 which is 0.09% LUT usage. the complete resource utilization chart for our system is:



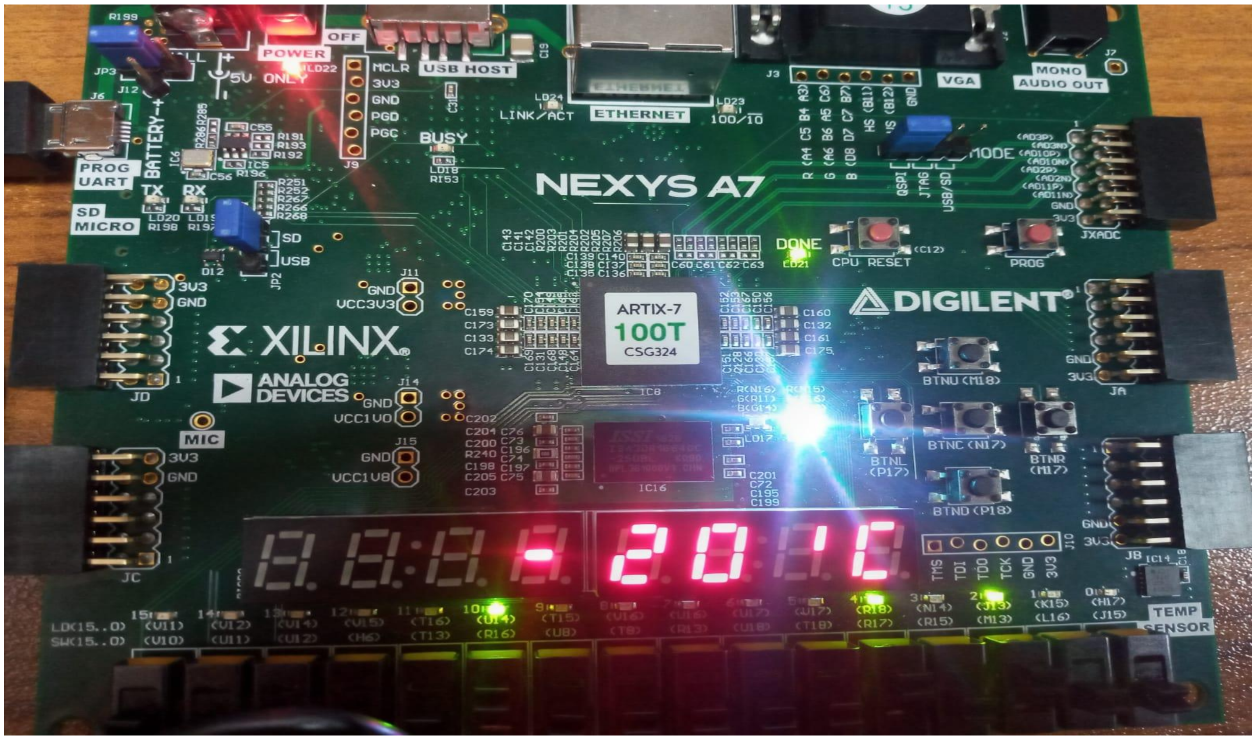
Name	Slice LUTs (63400)	Slice Registers (126800)	F7 Muxes (31700)	Block RAM Tile (135)	Bonded IOB (210)	BUFGCTRL (32)
dsd_CEP	58	64	1	1	39	1
P1 (PWM)	13	8	0	0	0	0
S1 (counter)	10	19	0	0	0	0

MODULES BLOCK DIAGRAM:

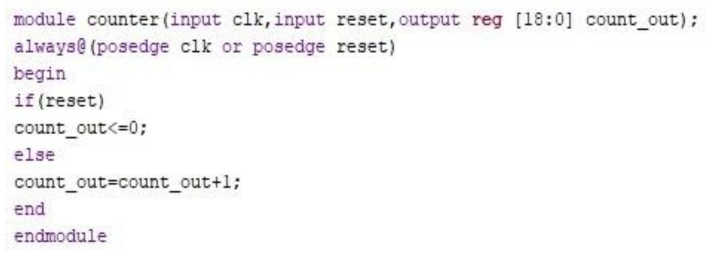


OPTIMIZATION:

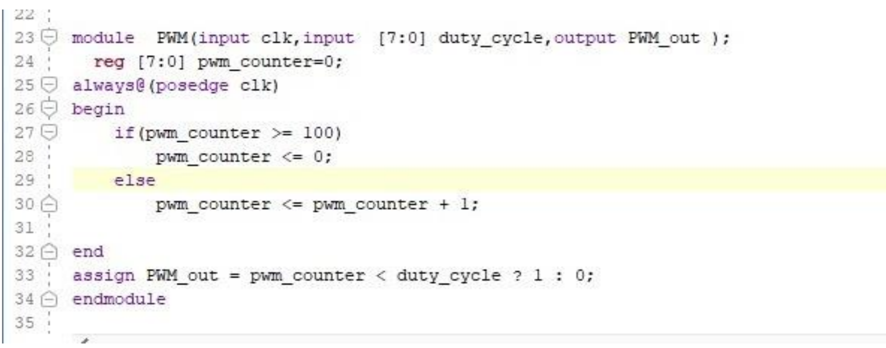
The optimization in our code is that it can handle a range of temperatures which are negative .Also we display our temperatures in proper celsius format for better readability and understanding.



VERILOG SOURCE CODE:

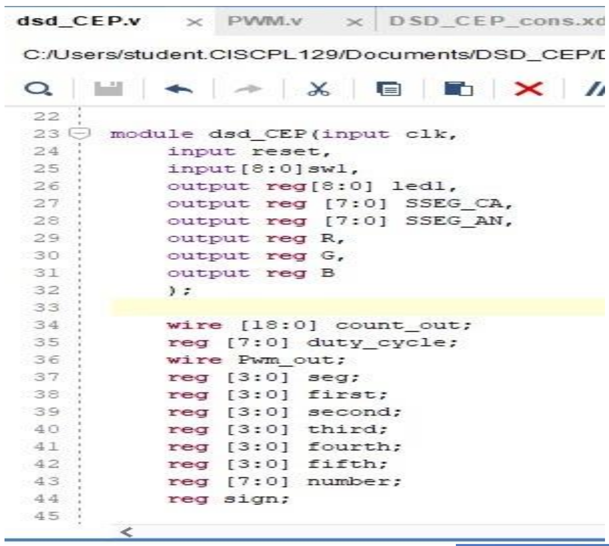


```verilog
module counter(input clk,input reset,output reg [18:0] count_out);
always@(posedge clk or posedge reset)
begin
if(reset)
count_out<=0;
else
count_out=count_out+1;
end
endmodule
```



```verilog

module  PWM(input clk,input  [7:0] duty_cycle,output PWM_out );
  reg [7:0] pwm_counter=0;
always@(posedge clk)
begin
    if(pwm_counter >= 100)
        pwm_counter <= 0;
    else
        pwm_counter <= pwm_counter + 1;

end
assign PWM_out = pwm_counter < duty_cycle ? 1 : 0;
endmodule

```

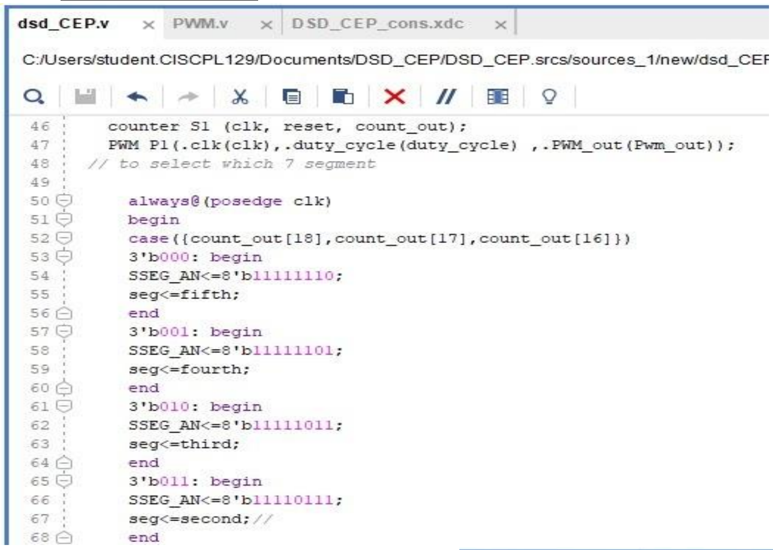
Top module is :



dsd_CEP.v   PWM.v   DSD_CEP_cons.xd

C:/Users/student.CISCPL129/Documents/DSD_CEP/D

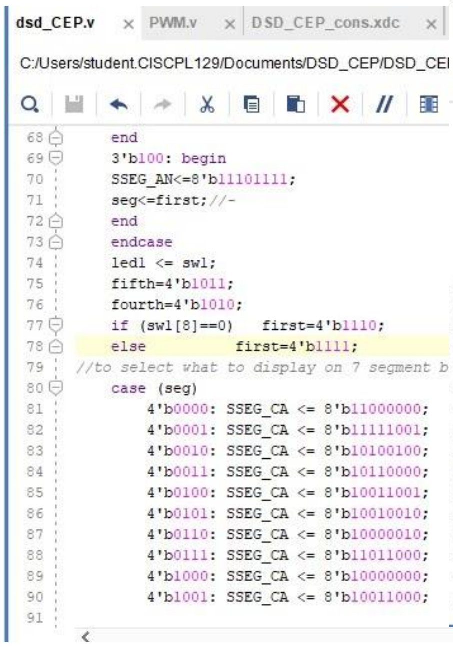
```verilog

module dsd_CEP(input clk,
        input reset,
        input[8:0]swl,
        output reg[8:0] led1,
        output reg [7:0] SSEG_CA,
        output reg [7:0] SSEG_AN,
        output reg R,
        output reg G,
        output reg B
        );

        wire [18:0] count_out;
        reg [7:0] duty_cycle;
        wire Pwm_out;
        reg [3:0] seg;
        reg [3:0] first;
        reg [3:0] second;
        reg [3:0] third;
        reg [3:0] fourth;
        reg [3:0] fifth;
        reg [7:0] number;
        reg sign;

```



dsd_CEP.v   PWM.v   DSD_CEP_cons.xdc

C:/Users/student.CISCPL129/Documents/DSD_CEP/DSD_CEP.srcs/sources_1/new/dsd_CEF

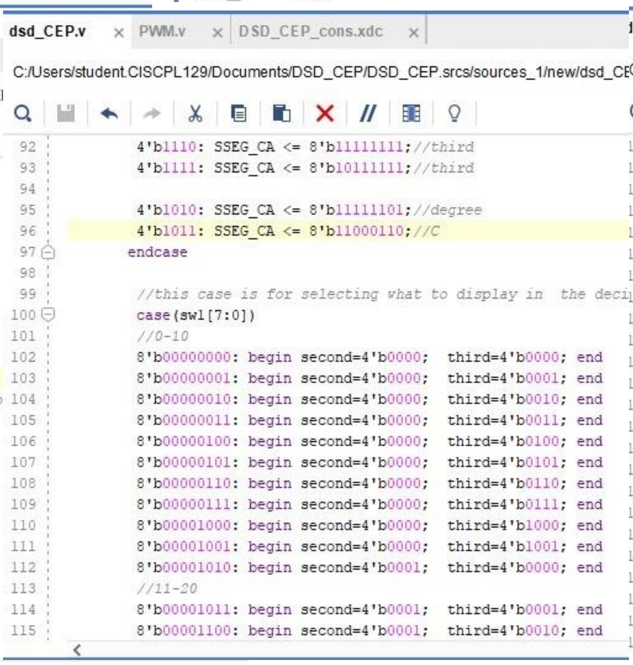
```verilog
     counter S1 (clk, reset, count_out);
     PWM P1(.clk(clk),.duty_cycle(duty_cycle) ,.PWM_out(Pwm_out));
  // to select which 7 segment

      always@(posedge clk)
      begin
      case({count_out[18],count_out[17],count_out[16]})
      3'b000: begin
      SSEG_AN<=8'b11111110;
      seg<=fifth;
      end
      3'b001: begin
      SSEG_AN<=8'b11111101;
      seg<=fourth;
      end
      3'b010: begin
      SSEG_AN<=8'b11111011;
      seg<=third;
      end
      3'b011: begin
      SSEG_AN<=8'b11110111;
      seg<=second;//
      end
```



dsd_CEP.v   PWM.v   DSD_CEP_cons.xdc

C:/Users/student.CISCPL129/Documents/DSD_CEP/DSD_CEI

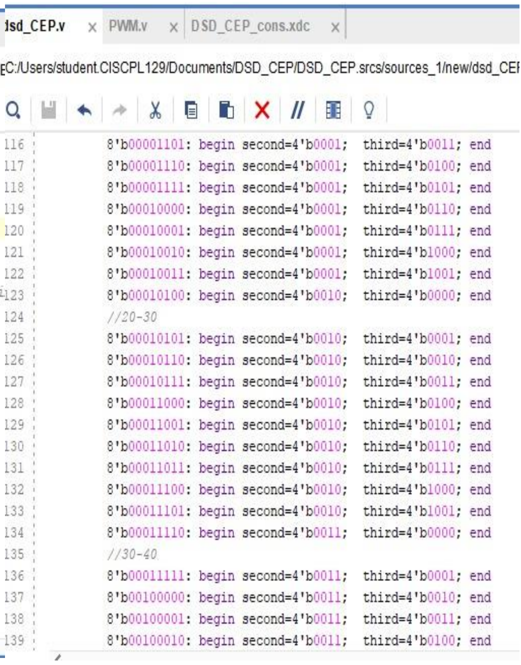
```verilog
      end
      3'b100: begin
      SSEG_AN<=8'b11101111;
      seg<=first;//-
      end
      endcase
      led1 <= swl;
      fifth=4'b1011;
      fourth=4'b1010;
      if (swl[8]==0)   first=4'b1110;
      else             first=4'b1111;
//to select what to display on 7 segment b
      case (seg)
          4'b0000: SSEG_CA <= 8'b11000000;
          4'b0001: SSEG_CA <= 8'b11111001;
          4'b0010: SSEG_CA <= 8'b10100100;
          4'b0011: SSEG_CA <= 8'b10110000;
          4'b0100: SSEG_CA <= 8'b10011001;
          4'b0101: SSEG_CA <= 8'b10010010;
          4'b0110: SSEG_CA <= 8'b10000010;
          4'b0111: SSEG_CA <= 8'b11011000;
          4'b1000: SSEG_CA <= 8'b10000000;
          4'b1001: SSEG_CA <= 8'b10011000;

```



dsd_CEP.v   PWM.v   DSD_CEP_cons.xdc

C:/Users/student.CISCPL129/Documents/DSD_CEP/DSD_CEP.srcs/sources_1/new/dsd_CE

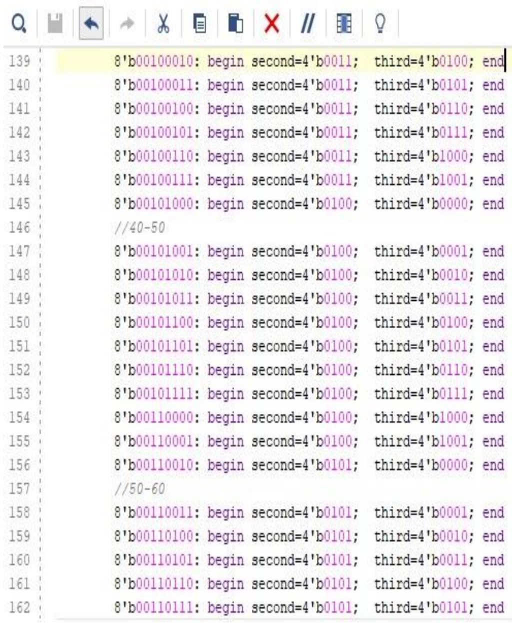
```verilog
          4'b1110: SSEG_CA <= 8'b11111111;//third
          4'b1111: SSEG_CA <= 8'b10111111;//third

          4'b1010: SSEG_CA <= 8'b11111101;//degree
          4'b1011: SSEG_CA <= 8'b11000110;//C
         endcase

          //this case is for selecting what to display in  the deci
          case(swl[7:0])
          //0-10
          8'b00000000: begin second=4'b0000;  third=4'b0000; end
          8'b00000001: begin second=4'b0000;  third=4'b0001; end
          8'b00000010: begin second=4'b0000;  third=4'b0010; end
          8'b00000011: begin second=4'b0000;  third=4'b0011; end
          8'b00000100: begin second=4'b0000;  third=4'b0100; end
          8'b00000101: begin second=4'b0000;  third=4'b0101; end
          8'b00000110: begin second=4'b0000;  third=4'b0110; end
          8'b00000111: begin second=4'b0000;  third=4'b0111; end
          8'b00001000: begin second=4'b0000;  third=4'b1000; end
          8'b00001001: begin second=4'b0000;  third=4'b1001; end
          8'b00001010: begin second=4'b0001;  third=4'b0000; end
          //11-20
          8'b00001011: begin second=4'b0001;  third=4'b0001; end
          8'b00001100: begin second=4'b0001;  third=4'b0010; end
```



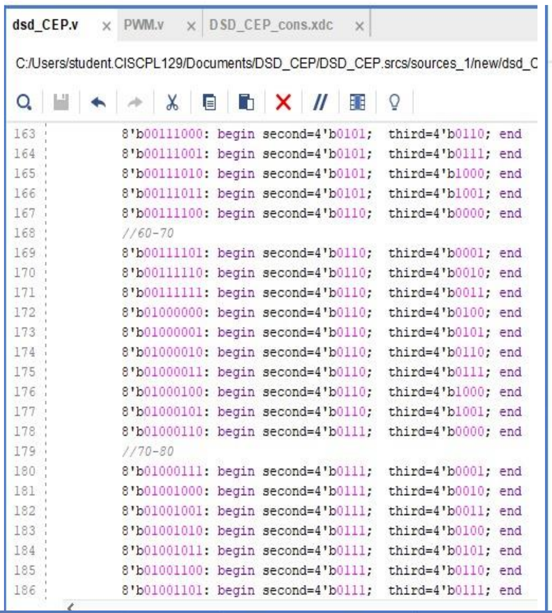
dsd_CEP.v   PWM.v   DSD_CEP_cons.xdc

C:/Users/student.CISCPL129/Documents/DSD_CEP/DSD_CEP.srcs/sources_1/new/dsd_CEF

```verilog
          8'b00001101: begin second=4'b0001;  third=4'b0011; end
          8'b00001110: begin second=4'b0001;  third=4'b0100; end
          8'b00001111: begin second=4'b0001;  third=4'b0101; end
          8'b00010000: begin second=4'b0001;  third=4'b0110; end
          8'b00010001: begin second=4'b0001;  third=4'b0111; end
          8'b00010010: begin second=4'b0001;  third=4'b1000; end
          8'b00010011: begin second=4'b0001;  third=4'b1001; end
          8'b00010100: begin second=4'b0010;  third=4'b0000; end
          //20-30
          8'b00010101: begin second=4'b0010;  third=4'b0001; end
          8'b00010110: begin second=4'b0010;  third=4'b0010; end
          8'b00010111: begin second=4'b0010;  third=4'b0011; end
          8'b00011000: begin second=4'b0010;  third=4'b0100; end
          8'b00011001: begin second=4'b0010;  third=4'b0101; end
          8'b00011010: begin second=4'b0010;  third=4'b0110; end
          8'b00011011: begin second=4'b0010;  third=4'b0111; end
          8'b00011100: begin second=4'b0010;  third=4'b1000; end
          8'b00011101: begin second=4'b0010;  third=4'b1001; end
          8'b00011110: begin second=4'b0011;  third=4'b0000; end
          //30-40
          8'b00011111: begin second=4'b0011;  third=4'b0001; end
          8'b00100000: begin second=4'b0011;  third=4'b0010; end
          8'b00100001: begin second=4'b0011;  third=4'b0011; end
          8'b00100010: begin second=4'b0011;  third=4'b0100; end
```

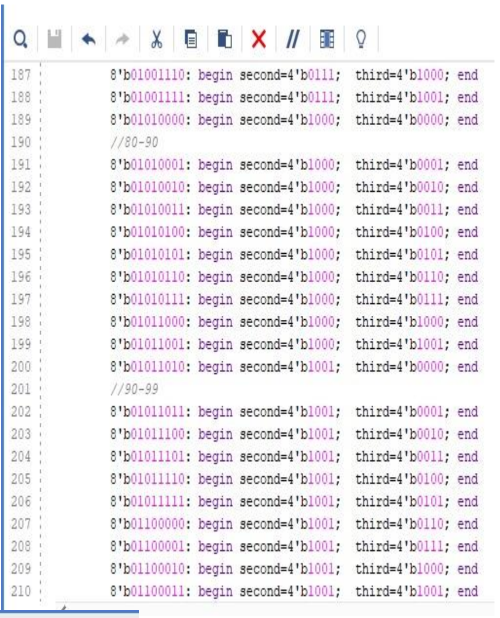


```verilog
        8'b00100010: begin second=4'b0011;  third=4'b0100; end
        8'b00100011: begin second=4'b0011;  third=4'b0101; end
        8'b00100100: begin second=4'b0011;  third=4'b0110; end
        8'b00100101: begin second=4'b0011;  third=4'b0111; end
        8'b00100110: begin second=4'b0011;  third=4'b1000; end
        8'b00100111: begin second=4'b0011;  third=4'b1001; end
        8'b00101000: begin second=4'b0100;  third=4'b0000; end
        //40-50
        8'b00101001: begin second=4'b0100;  third=4'b0001; end
        8'b00101010: begin second=4'b0100;  third=4'b0010; end
        8'b00101011: begin second=4'b0100;  third=4'b0011; end
        8'b00101100: begin second=4'b0100;  third=4'b0100; end
        8'b00101101: begin second=4'b0100;  third=4'b0101; end
        8'b00101110: begin second=4'b0100;  third=4'b0110; end
        8'b00101111: begin second=4'b0100;  third=4'b0111; end
        8'b00110000: begin second=4'b0100;  third=4'b1000; end
        8'b00110001: begin second=4'b0100;  third=4'b1001; end
        8'b00110010: begin second=4'b0101;  third=4'b0000; end
        //50-60
        8'b00110011: begin second=4'b0101;  third=4'b0001; end
        8'b00110100: begin second=4'b0101;  third=4'b0010; end
        8'b00110101: begin second=4'b0101;  third=4'b0011; end
        8'b00110110: begin second=4'b0101;  third=4'b0100; end
        8'b00110111: begin second=4'b0101;  third=4'b0101; end
```

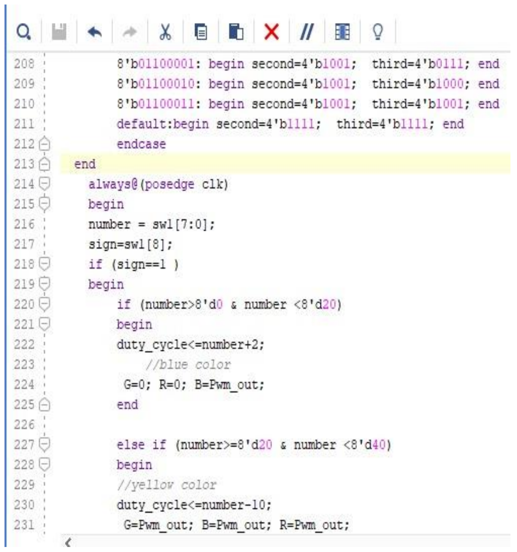


C:/Users/student.CISCPL129/Documents/DSD_CEP/DSD_CEP.srcs/sources_1/new/dsd_C

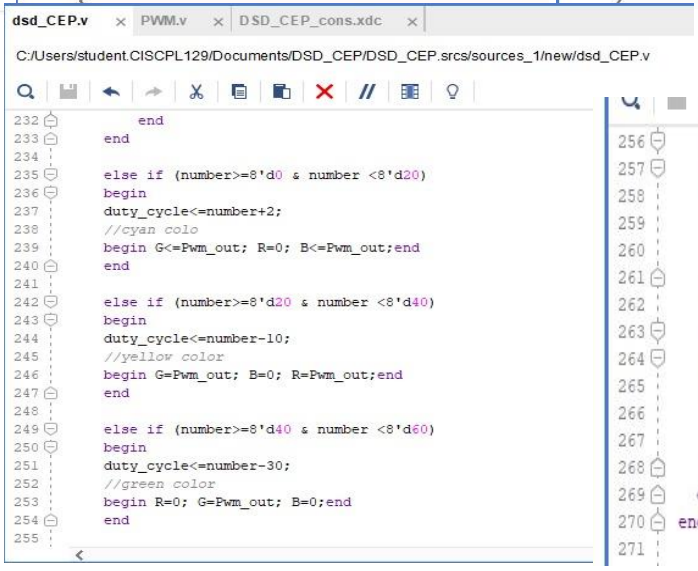
```verilog
        8'b00111000: begin second=4'b0101;  third=4'b0110; end
        8'b00111001: begin second=4'b0101;  third=4'b0111; end
        8'b00111010: begin second=4'b0101;  third=4'b1000; end
        8'b00111011: begin second=4'b0101;  third=4'b1001; end
        8'b00111100: begin second=4'b0110;  third=4'b0000; end
        //60-70
        8'b00111101: begin second=4'b0110;  third=4'b0001; end
        8'b00111110: begin second=4'b0110;  third=4'b0010; end
        8'b00111111: begin second=4'b0110;  third=4'b0011; end
        8'b01000000: begin second=4'b0110;  third=4'b0100; end
        8'b01000001: begin second=4'b0110;  third=4'b0101; end
        8'b01000010: begin second=4'b0110;  third=4'b0110; end
        8'b01000011: begin second=4'b0110;  third=4'b0111; end
        8'b01000100: begin second=4'b0110;  third=4'b1000; end
        8'b01000101: begin second=4'b0110;  third=4'b1001; end
        8'b01000110: begin second=4'b0111;  third=4'b0000; end
        //70-80
        8'b01000111: begin second=4'b0111;  third=4'b0001; end
        8'b01001000: begin second=4'b0111;  third=4'b0010; end
        8'b01001001: begin second=4'b0111;  third=4'b0011; end
        8'b01001010: begin second=4'b0111;  third=4'b0100; end
        8'b01001011: begin second=4'b0111;  third=4'b0101; end
        8'b01001100: begin second=4'b0111;  third=4'b0110; end
        8'b01001101: begin second=4'b0111;  third=4'b0111; end
```



```verilog
        8'b01001110: begin second=4'b0111;  third=4'b1000; end
        8'b01001111: begin second=4'b0111;  third=4'b1001; end
        8'b01010000: begin second=4'b1000;  third=4'b0000; end
        //80-90
        8'b01010001: begin second=4'b1000;  third=4'b0001; end
        8'b01010010: begin second=4'b1000;  third=4'b0010; end
        8'b01010011: begin second=4'b1000;  third=4'b0011; end
        8'b01010100: begin second=4'b1000;  third=4'b0100; end
        8'b01010101: begin second=4'b1000;  third=4'b0101; end
        8'b01010110: begin second=4'b1000;  third=4'b0110; end
        8'b01010111: begin second=4'b1000;  third=4'b0111; end
        8'b01011000: begin second=4'b1000;  third=4'b1000; end
        8'b01011001: begin second=4'b1000;  third=4'b1001; end
        8'b01011010: begin second=4'b1001;  third=4'b0000; end
        //90-99
        8'b01011011: begin second=4'b1001;  third=4'b0001; end
        8'b01011100: begin second=4'b1001;  third=4'b0010; end
        8'b01011101: begin second=4'b1001;  third=4'b0011; end
        8'b01011110: begin second=4'b1001;  third=4'b0100; end
        8'b01011111: begin second=4'b1001;  third=4'b0101; end
        8'b01100000: begin second=4'b1001;  third=4'b0110; end
        8'b01100001: begin second=4'b1001;  third=4'b0111; end
        8'b01100010: begin second=4'b1001;  third=4'b1000; end
        8'b01100011: begin second=4'b1001;  third=4'b1001; end
```

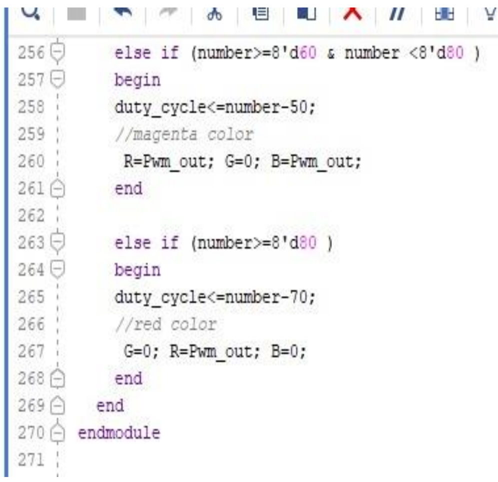


```verilog
        8'b01100001: begin second=4'b1001;  third=4'b0111; end
        8'b01100010: begin second=4'b1001;  third=4'b1000; end
        8'b01100011: begin second=4'b1001;  third=4'b1001; end
        default:begin second=4'b1111;  third=4'b1111; end
        endcase
  end
    always@(posedge clk)
    begin
    number = swl[7:0];
    sign=swl[8];
    if (sign==1 )
    begin
        if (number>8'd0 & number <8'd20)
        begin
        duty_cycle<=number+2;
            //blue color
         G=0; R=0; B=Pwm_out;
        end

        else if (number>=8'd20 & number <8'd40)
        begin
        //yellow color
        duty_cycle<=number-10;
         G=Pwm_out; B=Pwm_out; R=Pwm_out;
```



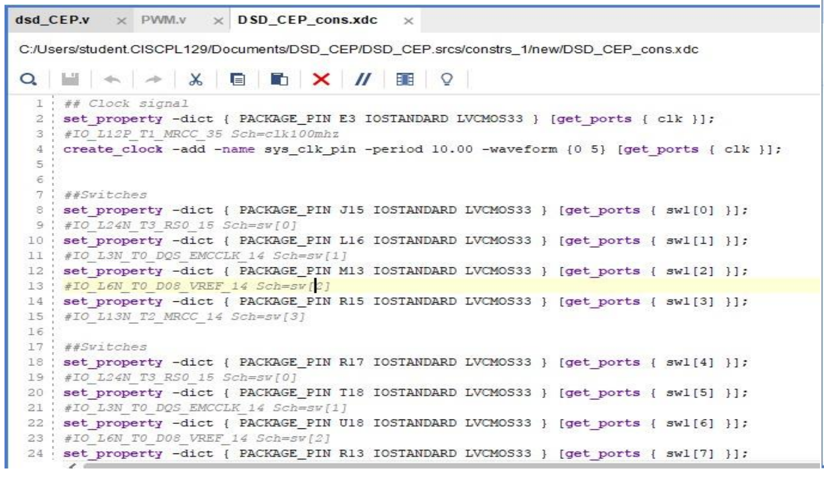
C:/Users/student.CISCPL129/Documents/DSD_CEP/DSD_CEP.srcs/sources_1/new/dsd_CEP.v

```verilog
            end
        end

        else if (number>=8'd0 & number <8'd20)
        begin
        duty_cycle<=number+2;
        //cyan colo
        begin G<=Pwm_out; R=0; B<=Pwm_out;end
        end

        else if (number>=8'd20 & number <8'd40)
        begin
        duty_cycle<=number-10;
        //yellow color
        begin G=Pwm_out; B=0; R=Pwm_out;end
        end

        else if (number>=8'd40 & number <8'd60)
        begin
        duty_cycle<=number-30;
        //green color
        begin R=0; G=Pwm_out; B=0;end
        end

```



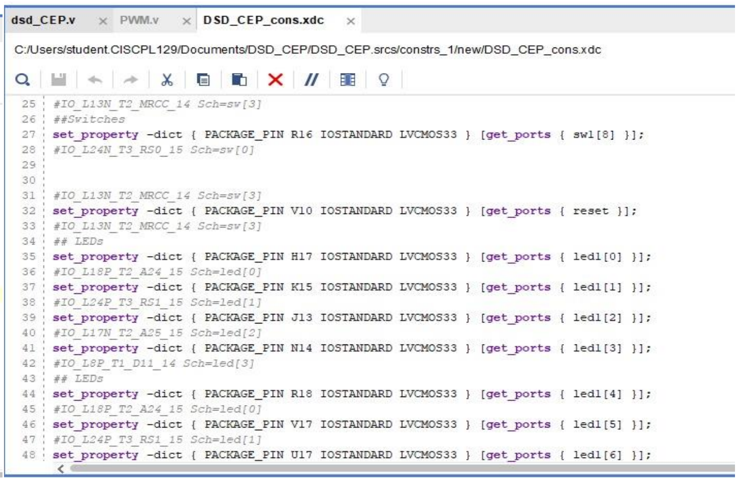
```verilog
    else if (number>=8'd60 & number <8'd80 )
    begin
    duty_cycle<=number-50;
    //magenta color
     R=Pwm_out; G=0; B=Pwm_out;
    end

    else if (number>=8'd80 )
    begin
    duty_cycle<=number-70;
    //red color
     G=0; R=Pwm_out; B=0;
    end
  end
endmodule

```

CONSTRAINT FILE:



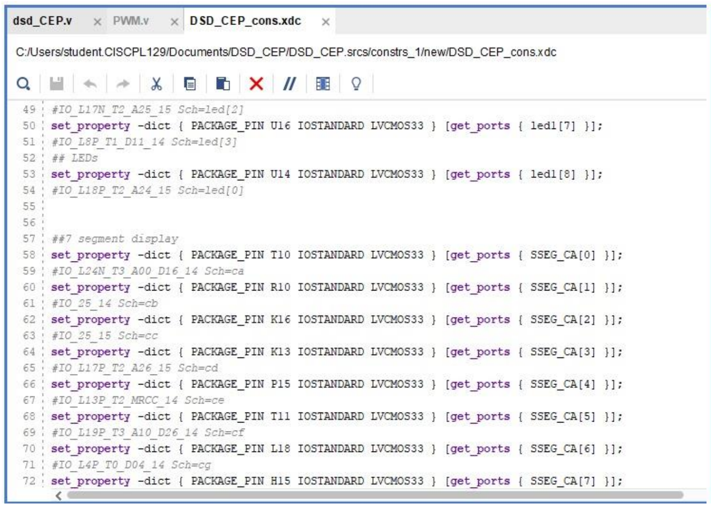
C:/Users/student.CISCPL129/Documents/DSD_CEP/DSD_CEP.srcs/constrs_1/new/DSD_CEP_cons.xdc

```tcl
## Clock signal
set_property -dict { PACKAGE_PIN E3 IOSTANDARD LVCMOS33 } [get_ports { clk }];
#IO_L12P_T1_MRCC_35 Sch=clk100mhz
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports { clk }];


##Switches
set_property -dict { PACKAGE_PIN J15 IOSTANDARD LVCMOS33 } [get_ports { swl[0] }];
#IO_L24N_T3_RS0_15 Sch=sw[0]
set_property -dict { PACKAGE_PIN L16 IOSTANDARD LVCMOS33 } [get_ports { swl[1] }];
#IO_L3N_T0_DQS_EMCCLK_14 Sch=sw[1]
set_property -dict { PACKAGE_PIN M13 IOSTANDARD LVCMOS33 } [get_ports { swl[2] }];
#IO_L6N_T0_D08_VREF_14 Sch=sw[2]
set_property -dict { PACKAGE_PIN R15 IOSTANDARD LVCMOS33 } [get_ports { swl[3] }];
#IO_L13N_T2_MRCC_14 Sch=sw[3]

##Switches
set_property -dict { PACKAGE_PIN R17 IOSTANDARD LVCMOS33 } [get_ports { swl[4] }];
#IO_L24N_T3_RS0_15 Sch=sw[0]
set_property -dict { PACKAGE_PIN T18 IOSTANDARD LVCMOS33 } [get_ports { swl[5] }];
#IO_L3N_T0_DQS_EMCCLK_14 Sch=sw[1]
set_property -dict { PACKAGE_PIN U18 IOSTANDARD LVCMOS33 } [get_ports { swl[6] }];
#IO_L6N_T0_D08_VREF_14 Sch=sw[2]
set_property -dict { PACKAGE_PIN R13 IOSTANDARD LVCMOS33 } [get_ports { swl[7] }];
```



C:/Users/student.CISCPL129/Documents/DSD_CEP/DSD_CEP.srcs/constrs_1/new/DSD_CEP_cons.xdc

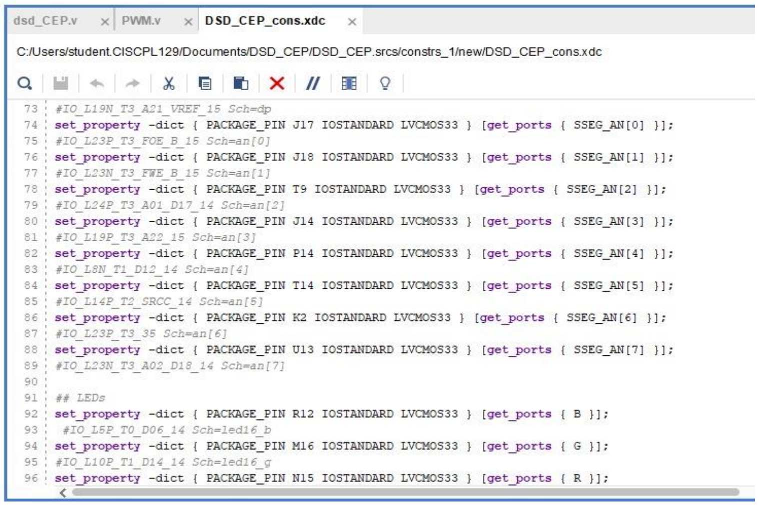
```tcl
#IO_L13N_T2_MRCC_14 Sch=sw[3]
##Switches
set_property -dict { PACKAGE_PIN R16 IOSTANDARD LVCMOS33 } [get_ports { swl[8] }];
#IO_L24N_T3_RS0_15 Sch=sw[0]


#IO_L13N_T2_MRCC_14 Sch=sw[3]
set_property -dict { PACKAGE_PIN V10 IOSTANDARD LVCMOS33 } [get_ports { reset }];
#IO_L13N_T2_MRCC_14 Sch=sw[3]
## LEDs
set_property -dict { PACKAGE_PIN H17 IOSTANDARD LVCMOS33 } [get_ports { ledl[0] }];
#IO_L18P_T2_A24_15 Sch=led[0]
set_property -dict { PACKAGE_PIN K15 IOSTANDARD LVCMOS33 } [get_ports { ledl[1] }];
#IO_L24P_T3_RS1_15 Sch=led[1]
set_property -dict { PACKAGE_PIN J13 IOSTANDARD LVCMOS33 } [get_ports { ledl[2] }];
#IO_L17N_T2_A25_15 Sch=led[2]
set_property -dict { PACKAGE_PIN N14 IOSTANDARD LVCMOS33 } [get_ports { ledl[3] }];
#IO_L8P_T1_D11_14 Sch=led[3]
## LEDs
set_property -dict { PACKAGE_PIN R18 IOSTANDARD LVCMOS33 } [get_ports { ledl[4] }];
#IO_L18P_T2_A24_15 Sch=led[0]
set_property -dict { PACKAGE_PIN V17 IOSTANDARD LVCMOS33 } [get_ports { ledl[5] }];
#IO_L24P_T3_RS1_15 Sch=led[1]
set_property -dict { PACKAGE_PIN U17 IOSTANDARD LVCMOS33 } [get_ports { ledl[6] }];
```



dsd_CEP.v | PWM.v | DSD_CEP_cons.xdc

C:/Users/student.CISCPL129/Documents/DSD_CEP/DSD_CEP.srcs/constrs_1/new/DSD_CEP_cons.xdc

```
#IO_L17N_T2_A25_15 Sch=led[2]
set_property -dict { PACKAGE_PIN U16 IOSTANDARD LVCMOS33 } [get_ports { led1[7] }];
#IO_L8P_T1_D11_14 Sch=led[3]
## LEDs
set_property -dict { PACKAGE_PIN U14 IOSTANDARD LVCMOS33 } [get_ports { led1[8] }];
#IO_L18P_T2_A24_15 Sch=led[0]


##7 segment display
set_property -dict { PACKAGE_PIN T10 IOSTANDARD LVCMOS33 } [get_ports { SSEG_CA[0] }];
#IO_L24N_T3_A00_D16_14 Sch=ca
set_property -dict { PACKAGE_PIN R10 IOSTANDARD LVCMOS33 } [get_ports { SSEG_CA[1] }];
#IO_25_14 Sch=cb
set_property -dict { PACKAGE_PIN K16 IOSTANDARD LVCMOS33 } [get_ports { SSEG_CA[2] }];
#IO_25_15 Sch=cc
set_property -dict { PACKAGE_PIN K13 IOSTANDARD LVCMOS33 } [get_ports { SSEG_CA[3] }];
#IO_L17P_T2_A26_15 Sch=cd
set_property -dict { PACKAGE_PIN P15 IOSTANDARD LVCMOS33 } [get_ports { SSEG_CA[4] }];
#IO_L13P_T2_MRCC_14 Sch=ce
set_property -dict { PACKAGE_PIN T11 IOSTANDARD LVCMOS33 } [get_ports { SSEG_CA[5] }];
#IO_L19P_T3_A10_D26_14 Sch=cf
set_property -dict { PACKAGE_PIN L18 IOSTANDARD LVCMOS33 } [get_ports { SSEG_CA[6] }];
#IO_L4P_T0_D04_14 Sch=cg
set_property -dict { PACKAGE_PIN H15 IOSTANDARD LVCMOS33 } [get_ports { SSEG_CA[7] }];
```

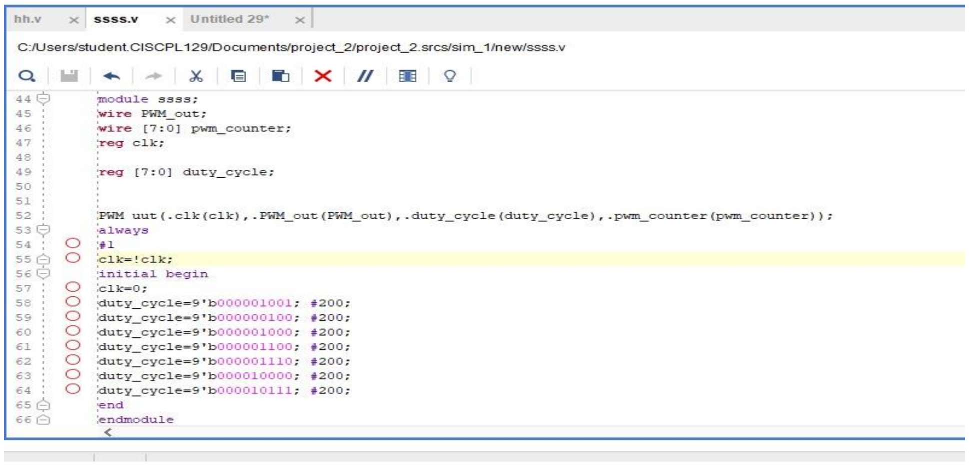


dsd_CEP.v | PWM.v | DSD_CEP_cons.xdc

C:/Users/student.CISCPL129/Documents/DSD_CEP/DSD_CEP.srcs/constrs_1/new/DSD_CEP_cons.xdc

```
#IO_L19N_T3_A21_VREF_15 Sch=dp
set_property -dict { PACKAGE_PIN J17 IOSTANDARD LVCMOS33 } [get_ports { SSEG_AN[0] }];
#IO_L23P_T3_FOE_B_15 Sch=an[0]
set_property -dict { PACKAGE_PIN J18 IOSTANDARD LVCMOS33 } [get_ports { SSEG_AN[1] }];
#IO_L23N_T3_FWE_B_15 Sch=an[1]
set_property -dict { PACKAGE_PIN T9 IOSTANDARD LVCMOS33 } [get_ports { SSEG_AN[2] }];
#IO_L24P_T3_A01_D17_14 Sch=an[2]
set_property -dict { PACKAGE_PIN J14 IOSTANDARD LVCMOS33 } [get_ports { SSEG_AN[3] }];
#IO_L19P_T3_A22_15 Sch=an[3]
set_property -dict { PACKAGE_PIN P14 IOSTANDARD LVCMOS33 } [get_ports { SSEG_AN[4] }];
#IO_L8N_T1_D12_14 Sch=an[4]
set_property -dict { PACKAGE_PIN T14 IOSTANDARD LVCMOS33 } [get_ports { SSEG_AN[5] }];
#IO_L14P_T2_SRCC_14 Sch=an[5]
set_property -dict { PACKAGE_PIN K2 IOSTANDARD LVCMOS33 } [get_ports { SSEG_AN[6] }];
#IO_L23P_T3_35 Sch=an[6]
set_property -dict { PACKAGE_PIN U13 IOSTANDARD LVCMOS33 } [get_ports { SSEG_AN[7] }];
#IO_L23N_T3_A02_D18_14 Sch=an[7]

## LEDs
set_property -dict { PACKAGE_PIN R12 IOSTANDARD LVCMOS33 } [get_ports { B }];
 #IO_L5P_T0_D06_14 Sch=led16_b
set_property -dict { PACKAGE_PIN M16 IOSTANDARD LVCMOS33 } [get_ports { G }];
#IO_L10P_T1_D14_14 Sch=led16_g
set_property -dict { PACKAGE_PIN N15 IOSTANDARD LVCMOS33 } [get_ports { R }];
```

TEST BENCH CODE FOR PWM:

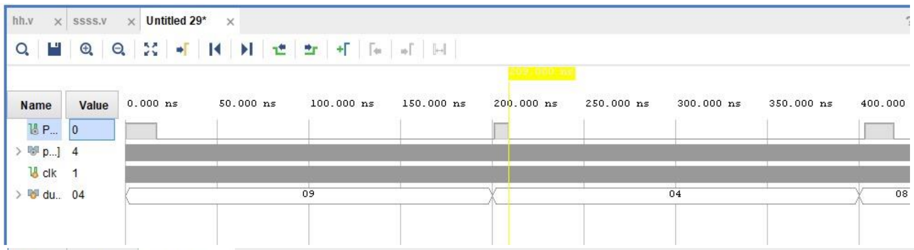


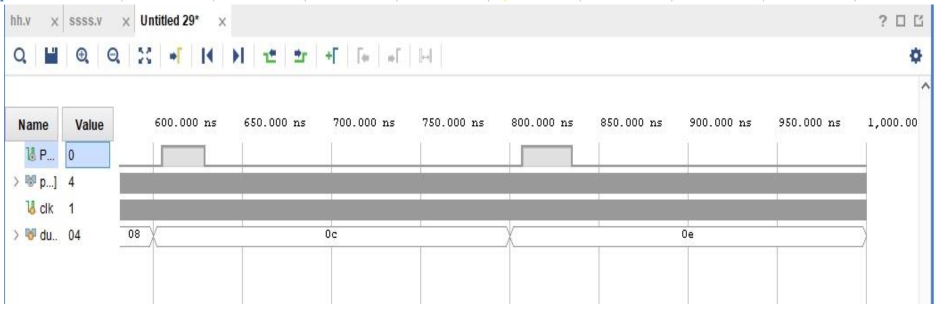
hh.v | ssss.v | Untitled 29*

C:/Users/student.CISCPL129/Documents/project_2/project_2.srcs/sim_1/new/ssss.v

```
module ssss;
wire PWM_out;
wire [7:0] pwm_counter;
reg clk;

reg [7:0] duty_cycle;


PWM uut(.clk(clk),.PWM_out(PWM_out),.duty_cycle(duty_cycle),.pwm_counter(pwm_counter));
always
#1
clk=!clk;
initial begin
clk=0;
duty_cycle=9'b000001001; #200;
duty_cycle=9'b000000100; #200;
duty_cycle=9'b000001000; #200;
duty_cycle=9'b000001100; #200;
duty_cycle=9'b000001110; #200;
duty_cycle=9'b000010000; #200;
duty_cycle=9'b000010111; #200;
end
endmodule
```

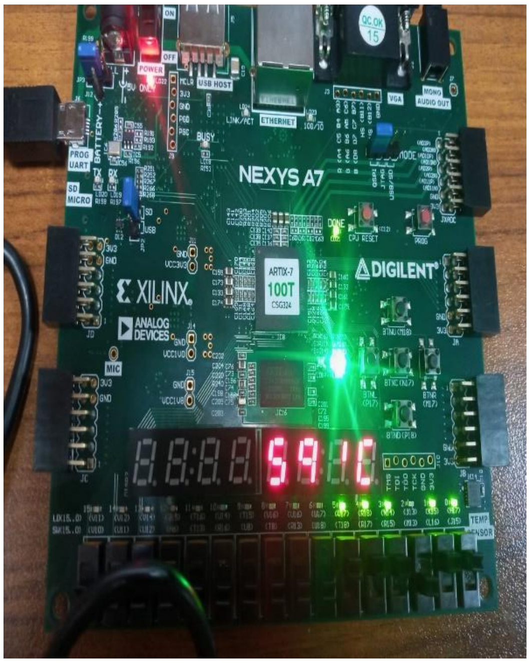
SIMULATION RESULTS:(here clock has perfect pulses up and down ,in picture it may seen as constant 1 but it is not, it seems like this due to long picture width)



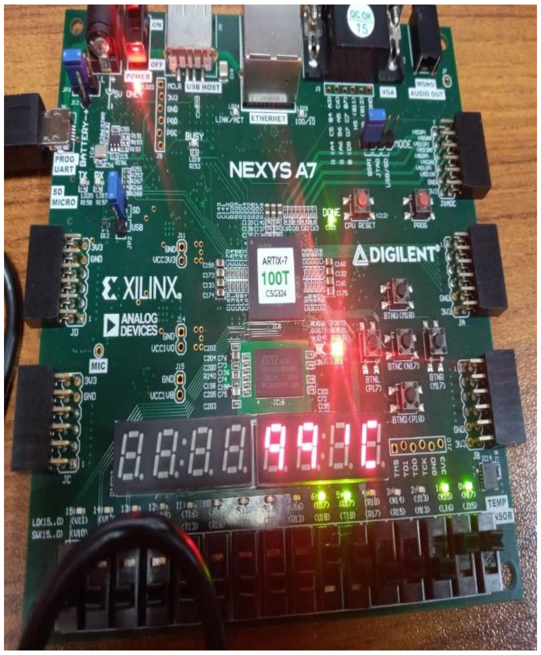


RESULT ANALYSIS:CODE OUTPUTS:

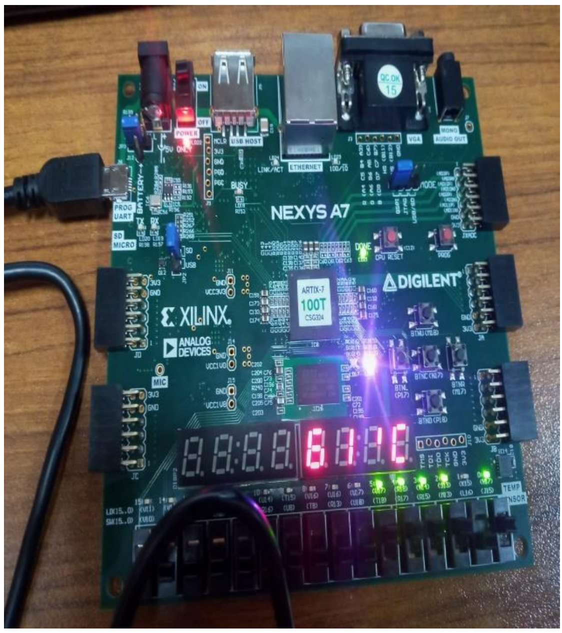
At temperature =59 (green color)



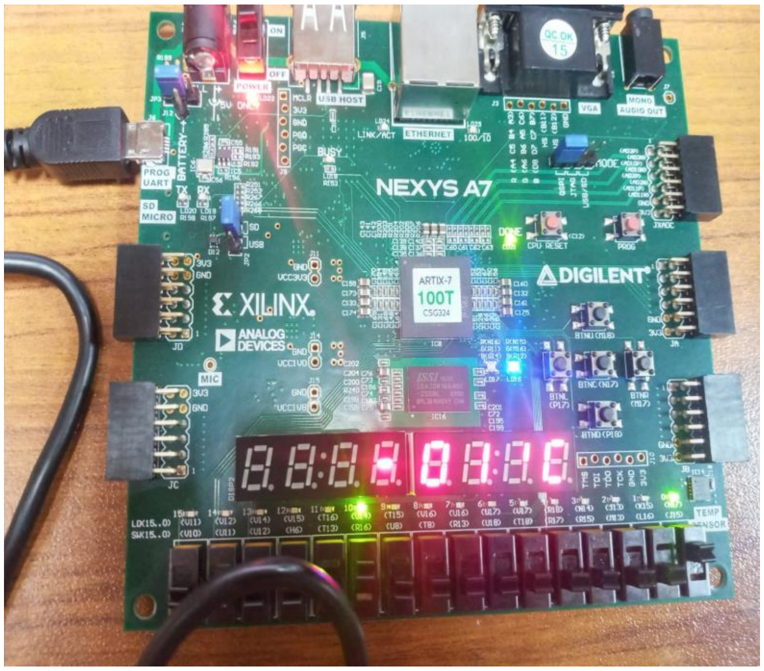
At temperature =99 (red color)



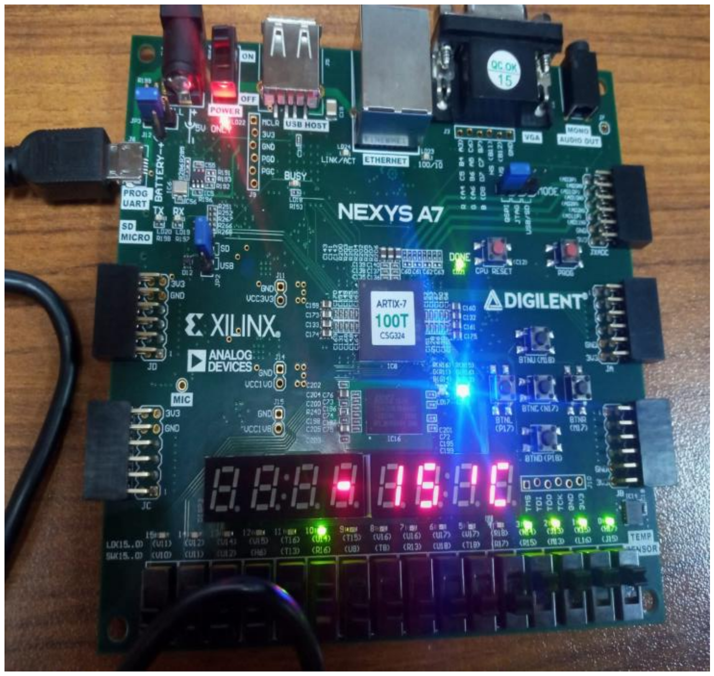
At temperature =61 (magenta color)



At temperature = -1 (blue color with less intensity)



At temperature = -15 (blue color with greater intensity)



The result of the system has been taken at different test values, and their correct color coding and intensity level has been observed . it performs exactly as it is been expected which has been shown in above diagrams at different input values.

<u>***RTL SCHEMATICS:***</u>