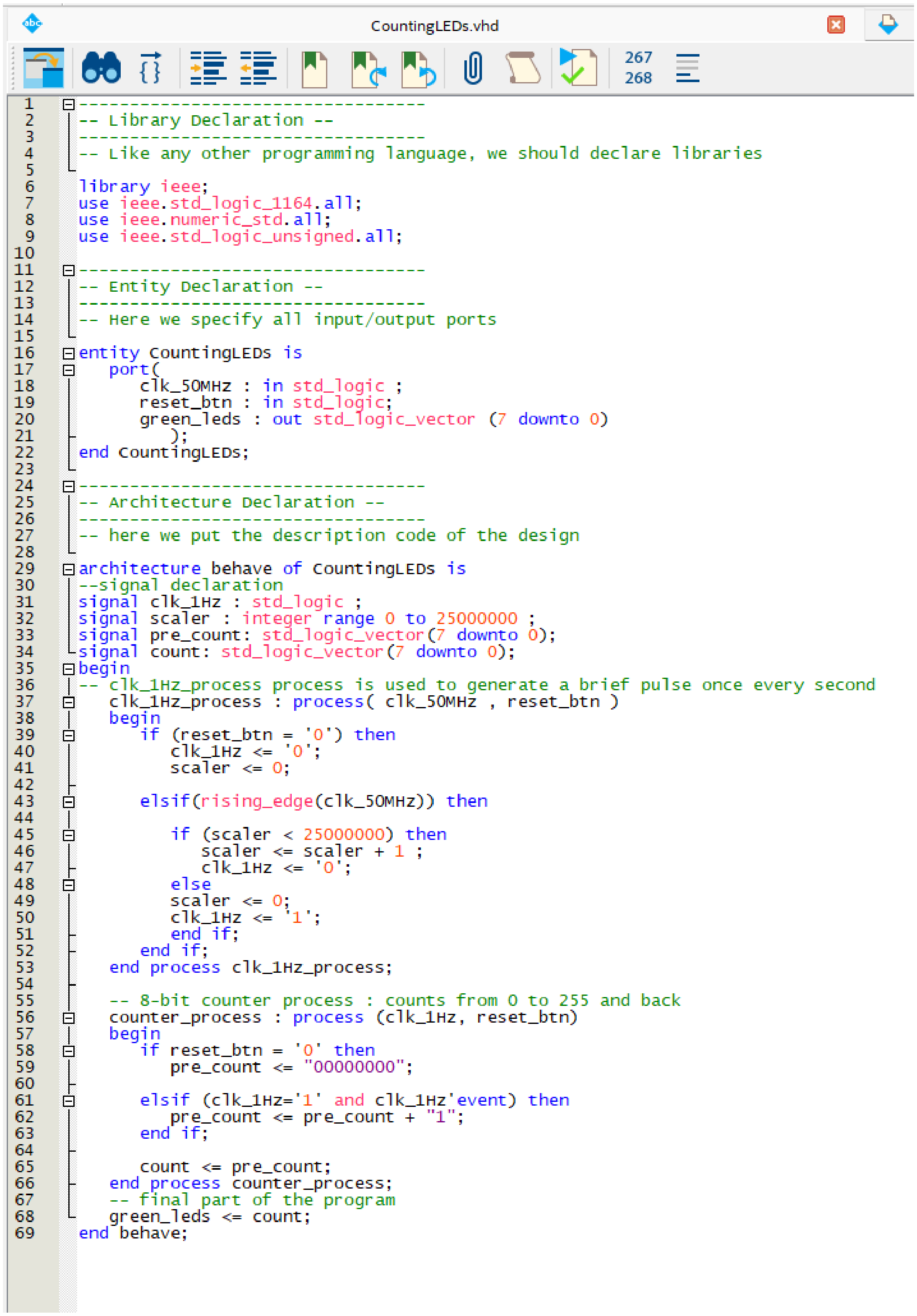
Mark Musil
February 6, 2022

Design of Digital Circuits
Module 5

Ohio University
Lab

Part 1

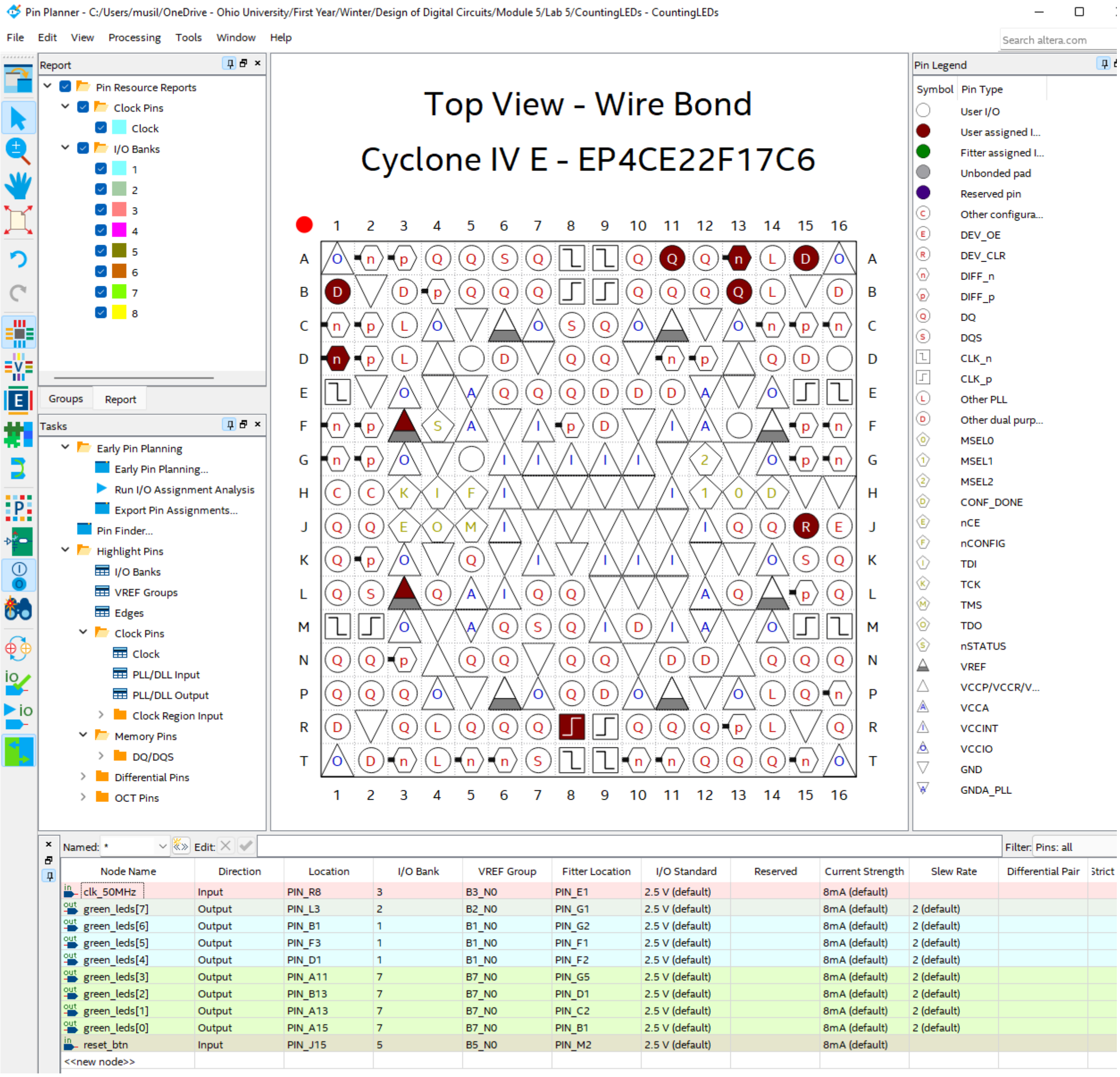
VHDL Code



```vhdl
----------------------------------
-- Library Declaration --
----------------------------------
-- Like any other programming language, we should declare libraries

library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
use ieee.std_logic_unsigned.all;

----------------------------------
-- Entity Declaration --
----------------------------------
-- Here we specify all input/output ports

entity CountingLEDs is
   port(
      clk_50MHz : in std_logic ;
      reset_btn : in std_logic;
      green_leds : out std_logic_vector (7 downto 0)
         );
end CountingLEDs;

----------------------------------
-- Architecture Declaration --
----------------------------------
-- here we put the description code of the design

architecture behave of CountingLEDs is
--signal declaration
signal clk_1Hz : std_logic ;
signal scaler : integer range 0 to 25000000 ;
signal pre_count: std_logic_vector(7 downto 0);
signal count: std_logic_vector(7 downto 0);
begin
-- clk_1Hz_process process is used to generate a brief pulse once every second
   clk_1Hz_process : process( clk_50MHz , reset_btn )
   begin
      if (reset_btn = '0') then
         clk_1Hz <= '0';
         scaler <= 0;

      elsif(rising_edge(clk_50MHz)) then

         if (scaler < 25000000) then
            scaler <= scaler + 1 ;
            clk_1Hz <= '0';
         else
         scaler <= 0;
         clk_1Hz <= '1';
         end if;
      end if;
   end process clk_1Hz_process;

   -- 8-bit counter process : counts from 0 to 255 and back
   counter_process : process (clk_1Hz, reset_btn)
   begin
      if reset_btn = '0' then
         pre_count <= "00000000";

      elsif (clk_1Hz='1' and clk_1Hz'event) then
         pre_count <= pre_count + "1";
      end if;

      count <= pre_count;
   end process counter_process;
   -- final part of the program
   green_leds <= count;
end behave;
```

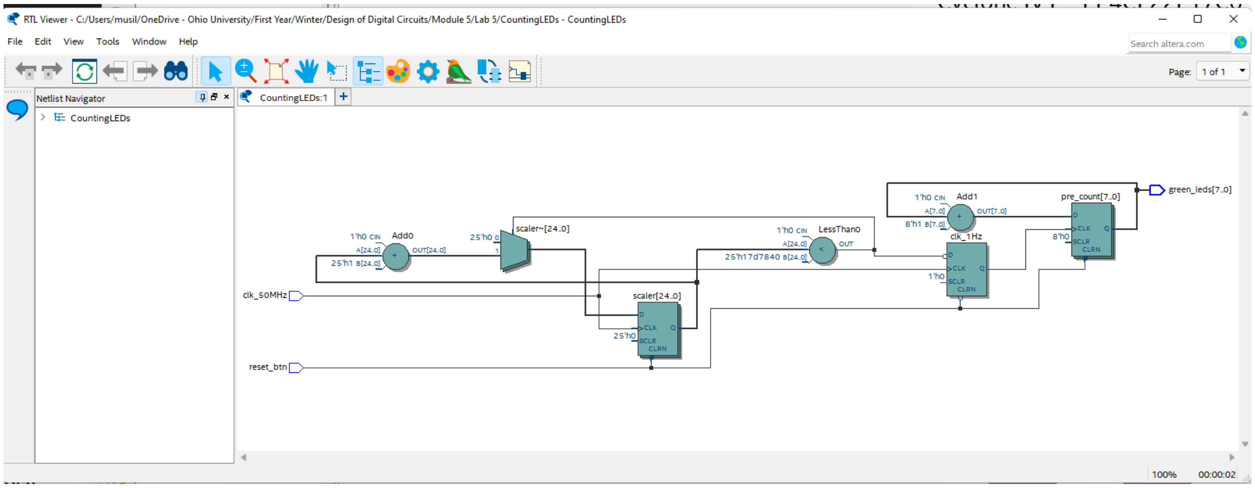
Mark Musil Design of Digital Circuits Ohio University
February 6, 2022 Module 5 Lab

Pin Planner



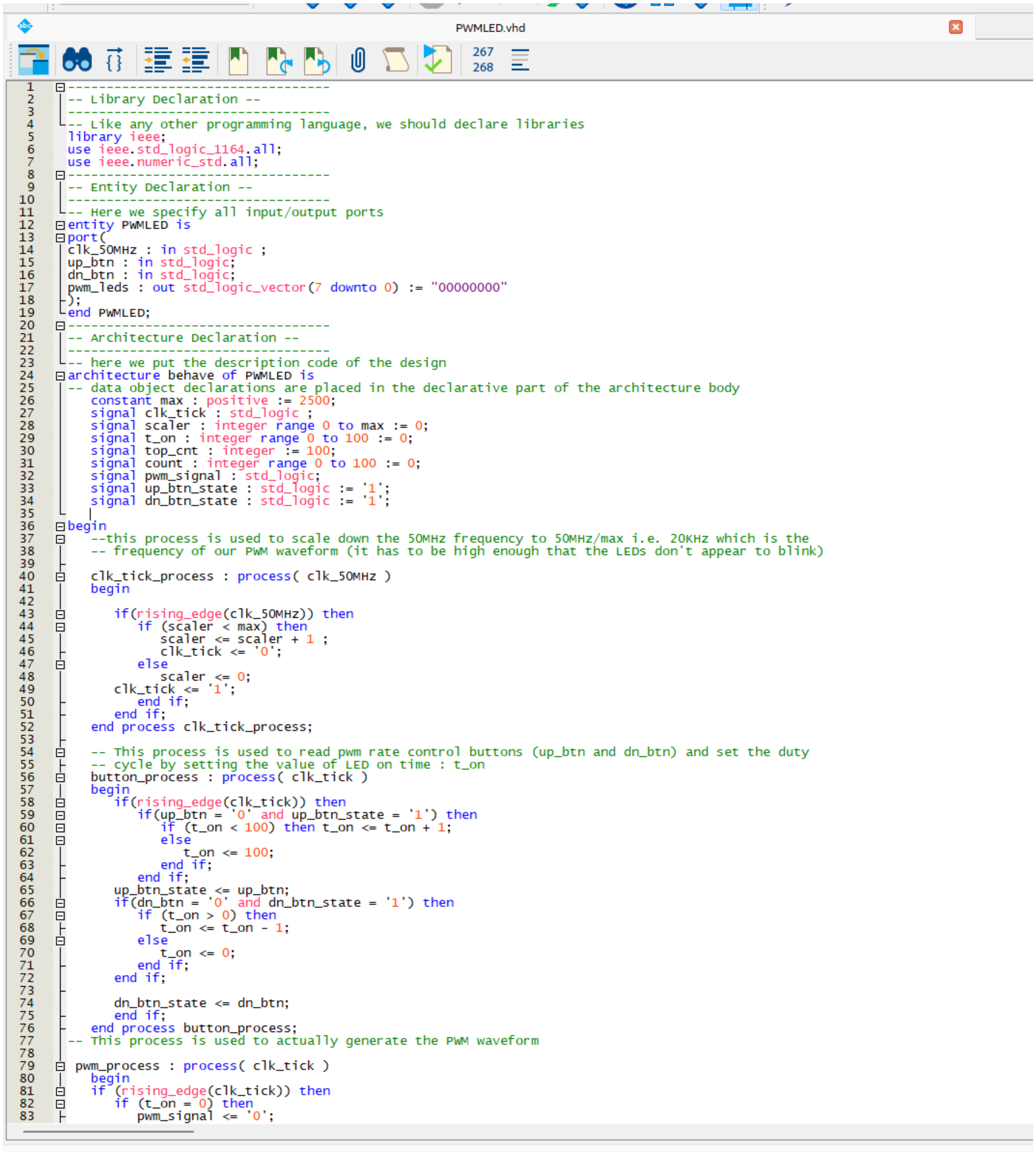
Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location	I/O Standard	Reserved	Current Strength	Slew Rate	Differential Pair	Strict
clk_50MHz	Input	PIN_R8	3	B3_N0	PIN_E1	2.5 V (default)		8mA (default)			
green_leds[7]	Output	PIN_L3	2	B2_N0	PIN_G1	2.5 V (default)		8mA (default)	2 (default)		
green_leds[6]	Output	PIN_B1	1	B1_N0	PIN_G2	2.5 V (default)		8mA (default)	2 (default)		
green_leds[5]	Output	PIN_F3	1	B1_N0	PIN_F1	2.5 V (default)		8mA (default)	2 (default)		
green_leds[4]	Output	PIN_D1	1	B1_N0	PIN_F2	2.5 V (default)		8mA (default)	2 (default)		
green_leds[3]	Output	PIN_A11	7	B7_N0	PIN_G5	2.5 V (default)		8mA (default)	2 (default)		
green_leds[2]	Output	PIN_B13	7	B7_N0	PIN_D1	2.5 V (default)		8mA (default)	2 (default)		
green_leds[1]	Output	PIN_A13	7	B7_N0	PIN_C2	2.5 V (default)		8mA (default)	2 (default)		
green_leds[0]	Output	PIN_A15	7	B7_N0	PIN_B1	2.5 V (default)		8mA (default)	2 (default)		
reset_btn	Input	PIN_J15	5	B5_N0	PIN_M2	2.5 V (default)		8mA (default)			
<<new node>>											

RTL Window


RTL Viewer - C:/Users/musil/OneDrive - Ohio University/First Year/Winter/Design of Digital Circuits/Module 5/Lab 5/CountingLEDs - CountingLEDs
File Edit View Tools Window Help
Search altera.com
Page: 1 of 1
Netlist Navigator
CountingLEDs
CountingLEDs:1
1'h0 CIN
Add0
A[24..0]
25'h1 B[24..0]
OUT[24..0]
25'h0 0
1
scaler~[24..0]
clk_50MHz
reset_btn
scaler[24..0]
D
CLK
Q
25'h0
SCLR
CLRN
1'h0 CIN
LessThan0
A[24..0]
25'h17d7840 B[24..0]
OUT
clk_1Hz
1'h0
1'h0 CIN
Add1
A[7..0]
8'h1 B[7..0]
OUT[7..0]
pre_count[7..0]
8'h0
green_leds[7..0]
100%
00:00:02

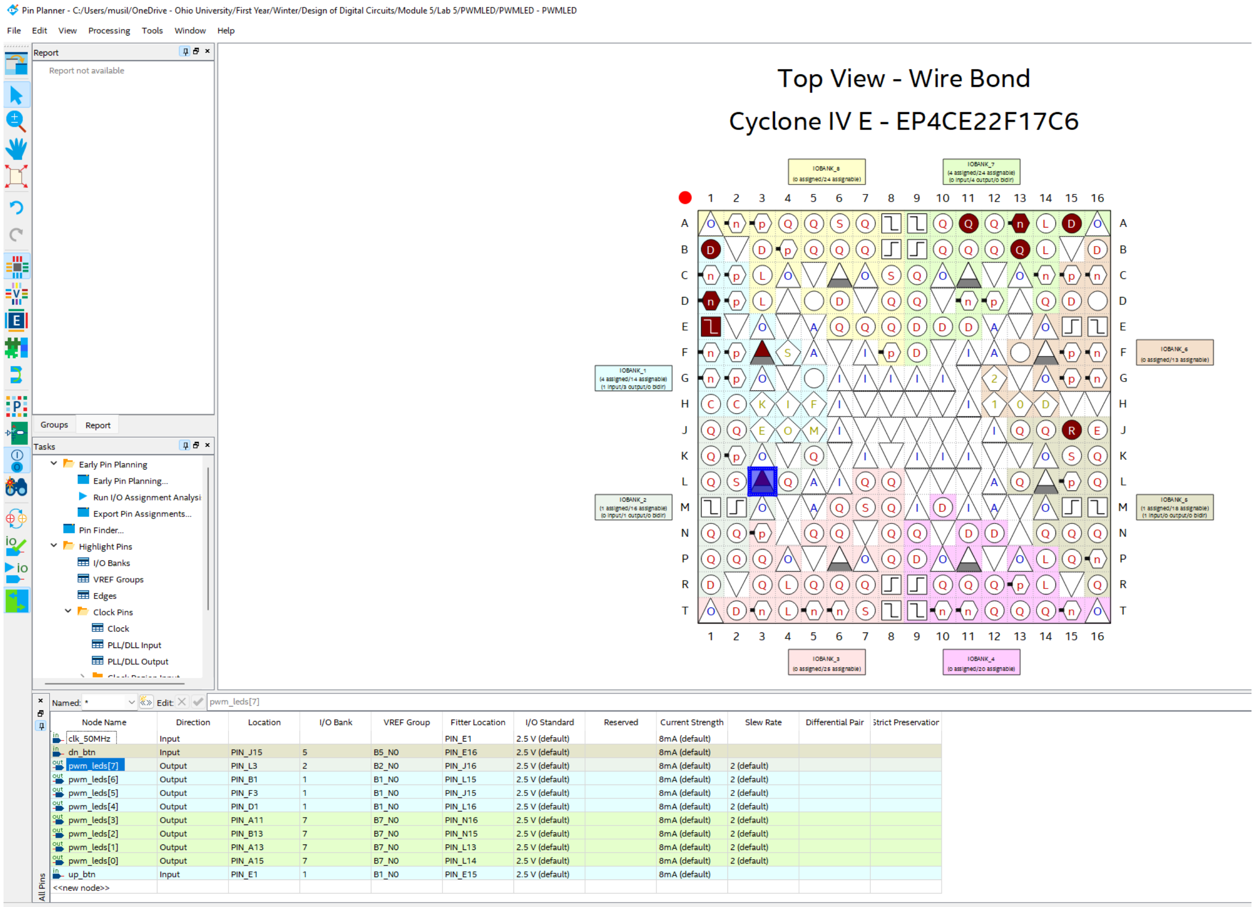
Part 2

VHDL Code



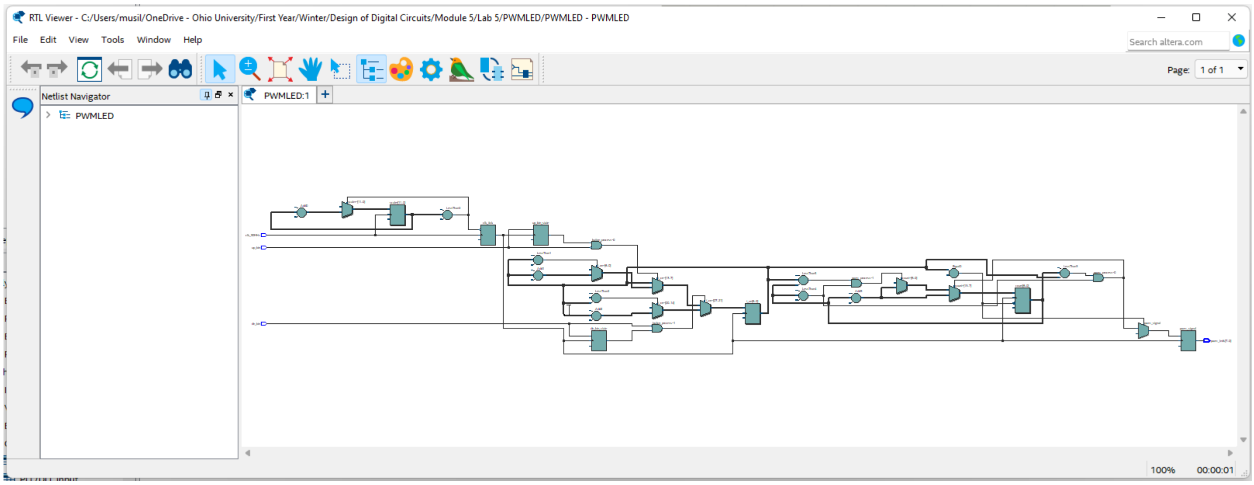
```vhdl
----------------------------------
-- Library Declaration --
----------------------------------
-- Like any other programming language, we should declare libraries
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
----------------------------------
-- Entity Declaration --
----------------------------------
-- Here we specify all input/output ports
entity PWMLED is
port(
clk_50MHz : in std_logic ;
up_btn : in std_logic;
dn_btn : in std_logic;
pwm_leds : out std_logic_vector(7 downto 0) := "00000000"
);
end PWMLED;
----------------------------------
-- Architecture Declaration --
----------------------------------
-- here we put the description code of the design
architecture behave of PWMLED is
-- data object declarations are placed in the declarative part of the architecture body
   constant max : positive := 2500;
   signal clk_tick : std_logic ;
   signal scaler : integer range 0 to max := 0;
   signal t_on : integer range 0 to 100 := 0;
   signal top_cnt : integer := 100;
   signal count : integer range 0 to 100 := 0;
   signal pwm_signal : std_logic;
   signal up_btn_state : std_logic := '1';
   signal dn_btn_state : std_logic := '1';

begin
   --this process is used to scale down the 50MHz frequency to 50MHz/max i.e. 20KHz which is the
   -- frequency of our PWM waveform (it has to be high enough that the LEDs don't appear to blink)

   clk_tick_process : process( clk_50MHz )
   begin

      if(rising_edge(clk_50MHz)) then
         if (scaler < max) then
            scaler <= scaler + 1 ;
            clk_tick <= '0';
         else
            scaler <= 0;
      clk_tick <= '1';
         end if;
      end if;
   end process clk_tick_process;

   -- This process is used to read pwm rate control buttons (up_btn and dn_btn) and set the duty
   -- cycle by setting the value of LED on time : t_on
   button_process : process( clk_tick )
   begin
      if(rising_edge(clk_tick)) then
         if(up_btn = '0' and up_btn_state = '1') then
            if (t_on < 100) then t_on <= t_on + 1;
            else
               t_on <= 100;
            end if;
         end if;
      up_btn_state <= up_btn;
      if(dn_btn = '0' and dn_btn_state = '1') then
         if (t_on > 0) then
            t_on <= t_on - 1;
         else
            t_on <= 0;
         end if;
      end if;

      dn_btn_state <= dn_btn;
      end if;
   end process button_process;
-- This process is used to actually generate the PWM waveform

 pwm_process : process( clk_tick )
   begin
   if (rising_edge(clk_tick)) then
      if (t_on = 0) then
         pwm_signal <= '0';
```

Pin Planner



Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location	I/O Standard	Reserved	Current Strength	Slew Rate	Differential Pair	Strict Preservation
clk_50MHz	Input				PIN_E1	2.5 V (default)		8mA (default)			
dn_btn	Input	PIN_J15	5	B5_N0	PIN_E16	2.5 V (default)		8mA (default)			
pwm_leds[7]	Output	PIN_L3	2	B2_N0	PIN_J16	2.5 V (default)		8mA (default)	2 (default)		
pwm_leds[6]	Output	PIN_B1	1	B1_N0	PIN_L15	2.5 V (default)		8mA (default)	2 (default)		
pwm_leds[5]	Output	PIN_F3	1	B1_N0	PIN_J15	2.5 V (default)		8mA (default)	2 (default)		
pwm_leds[4]	Output	PIN_D1	1	B1_N0	PIN_L16	2.5 V (default)		8mA (default)	2 (default)		
pwm_leds[3]	Output	PIN_A11	7	B7_N0	PIN_N16	2.5 V (default)		8mA (default)	2 (default)		
pwm_leds[2]	Output	PIN_B13	7	B7_N0	PIN_N15	2.5 V (default)		8mA (default)	2 (default)		
pwm_leds[1]	Output	PIN_A13	7	B7_N0	PIN_L13	2.5 V (default)		8mA (default)	2 (default)		
pwm_leds[0]	Output	PIN_A15	7	B7_N0	PIN_L14	2.5 V (default)		8mA (default)	2 (default)		
up_btn	Input	PIN_E1	1	B1_N0	PIN_E15	2.5 V (default)		8mA (default)			
<<new node>>											

RTL Window


RTL Viewer - C:/Users/musil/OneDrive - Ohio University/First Year/Winter/Design of Digital Circuits/Module 5/Lab 5/PWMLED/PWMLED - PWMLED
File Edit View Tools Window Help
Search altera.com
Page: 1 of 1
Netlist Navigator
PWMLED
PWMLED:1
100%
00:00:01