LAB#10

Hardware Development

**Introduction:**

In this lab we have learnt to make a counter. We have used interfaced hardware to implement our program on it.

In the hardware board, we have used 7-segment display and three input switches initially to control the output of the counter.

Input 1: Reset

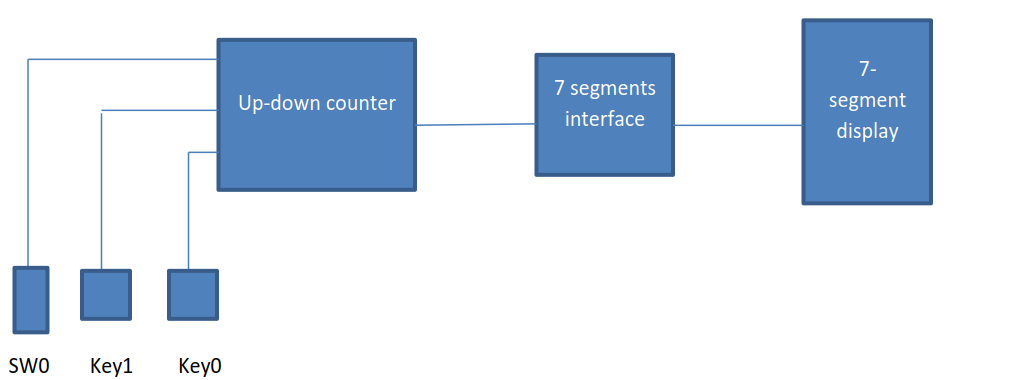
Input 2: to increment the value of the counter.

Input 3: to set the counter run upside or downside

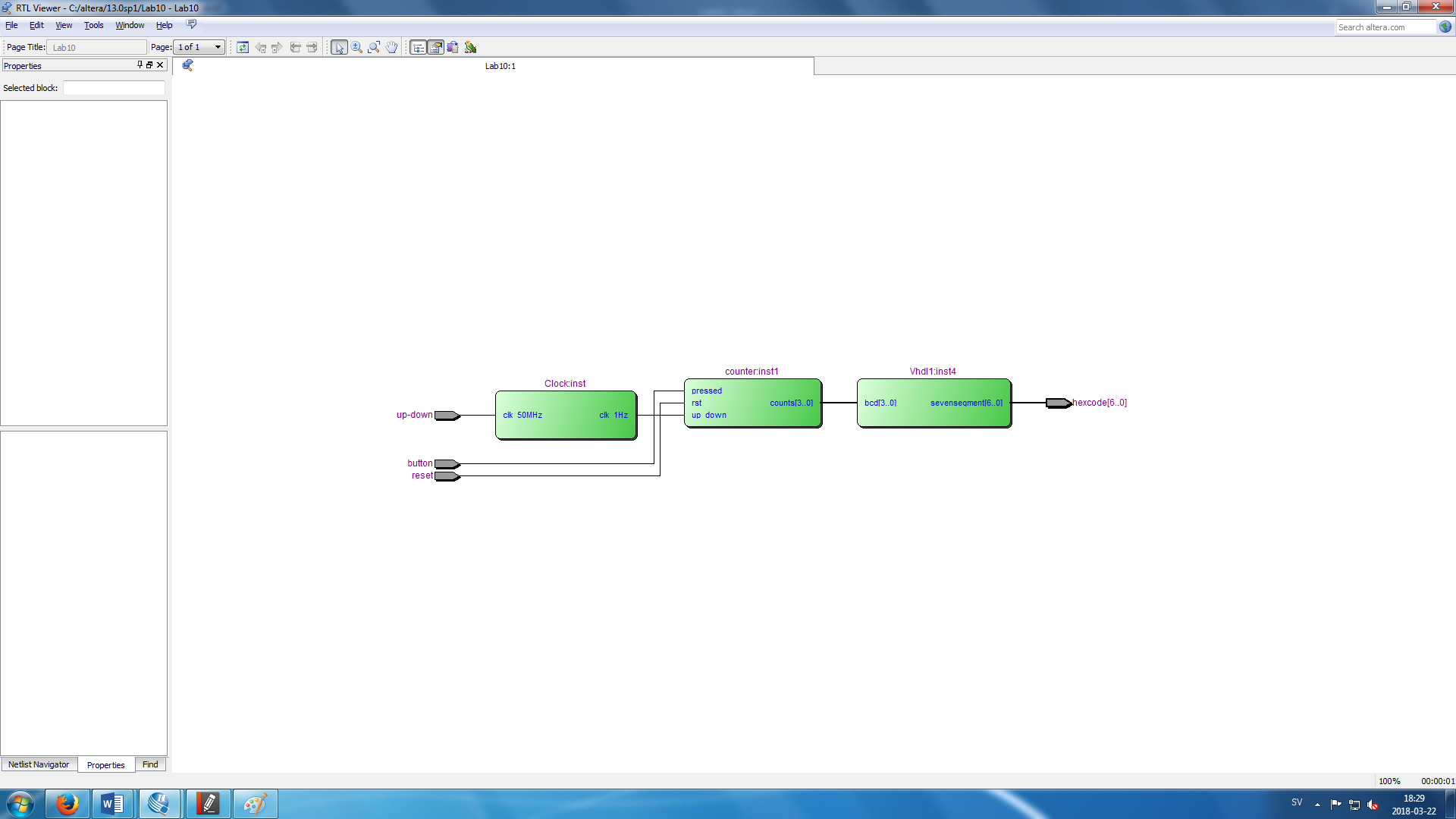
First, we made a counter through a VHDL code and then we created a code which will take input from a counter and display it accordingly on seven segment. Then, we updated a block diagram file so that we can see both the created blocks and then we assigned them the needed inputs and outputs. Then we assigned the input pins by using Pin Planner. We check the working of the counter by resetting it or making it run upside or downside.

After all this, we replaced the button with a clock to increment or decrement the counter automatically with the clock. Initially, we provided 50MHz which was too high to see the result through human eye so we implemented a frequency divider algorithm which divides the frequency 50 times.

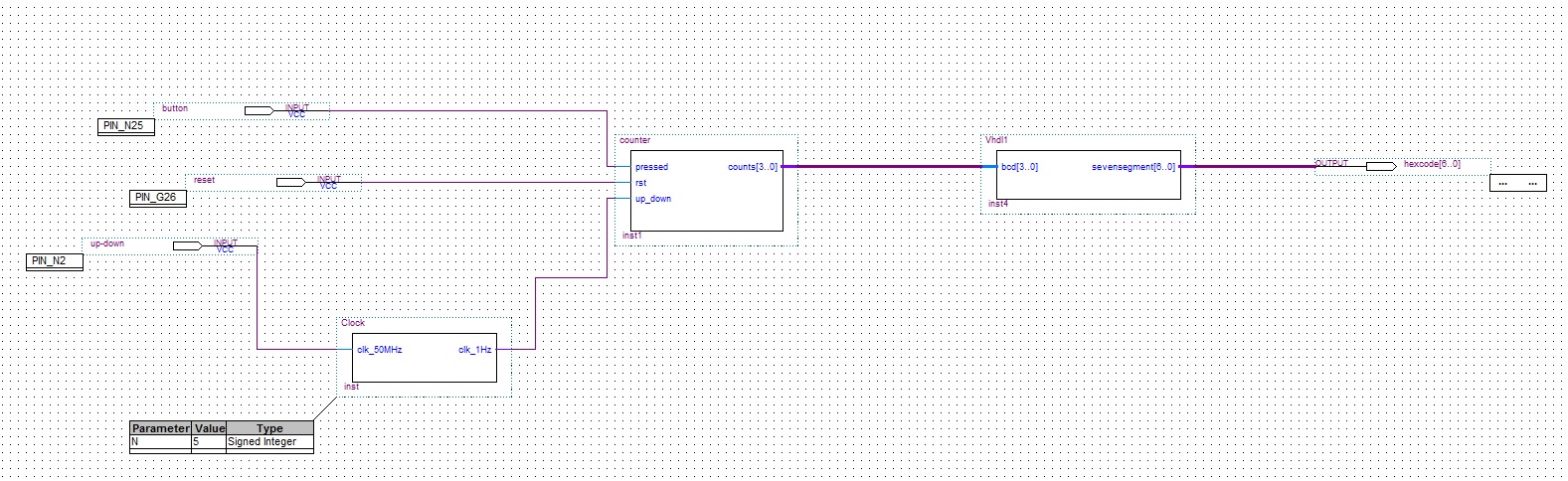
Lastly, we have implemented a Signal Tap Analyzer to analyze logic with respect to time.



**RTL View:**



**Block Diagram:**



**CODE:**

**Counter:**

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_signed.ALL;

ENTITY counter IS

PORT (pressed: IN STD\_LOGIC;

rst: IN STD\_LOGIC;

up\_down: IN STD\_LOGIC;

counts: OUT STD\_LOGIC\_VECTOR(3 downto 0)

);

END ENTITY;

ARCHITECTURE updown\_counter OF counter IS

BEGIN

process(up\_down, rst)

variable COUNT: STD\_LOGIC\_VECTOR(3 downto 0);

BEGIN

if (rst = '0') then

COUNT:= (others=>'0');

elsif (up\_down'event and up\_down = '1') then

case pressed IS

WHEN '1' => COUNT:= COUNT+1;

WHEN others => COUNT:= COUNT-1;

END case;

counts <= COUNT;

END IF;

END PROCESS;

END ARCHITECTURE;

**Display:**

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.All;

ENTITY Vhdl1 IS

PORT (bcd: IN STD\_LOGIC\_VECTOR(3 downto 0);

sevensegment: OUT STD\_LOGIC\_VECTOR(6 downto 0));

END ENTITY;

ARCHITECTURE RTL OF Vhdl1 IS

--0 1 2 3 4 5 6 7 8 9 A B

--C D E F

--Figure 1: Hexadecimal digits

BEGIN

process (bcd)

BEGIN

case bcd is

when "0000" => sevensegment<="1000000"; --0

when "0001" => sevensegment<="1111001"; --1

when "0010" => sevensegment<="0100100"; --2

when "0011" => sevensegment<="0110000"; --3

when "0100" => sevensegment<="0011001"; --4

when "0101" => sevensegment<="0010010"; --5

when "0110" => sevensegment<="0000010"; --6

when "0111" => sevensegment<="1111000"; --7

when "1000" => sevensegment<="0000000"; --8

when "1001" => sevensegment<="0010000"; --9

when "1010" => sevensegment<="0001000"; --10

when "1011" => sevensegment<="0000011"; --11

when "1100" => sevensegment<="1000110"; --12

when "1101" => sevensegment<="0100001"; --13

when "1110" => sevensegment<="0000110"; --14

when "1111" => sevensegment<="0001110"; --15

when others => sevensegment<="XXXXXXX";

END case;

END process;

END RTL;

**Clock:**

LIBRARY ieee;

USE ieee.std\_logic\_1164.ALL;

USE ieee.std\_logic\_unsigned.ALL;

ENTITY Clock IS

GENERIC(N: integer:=5);

PORT (

clk\_50MHz: IN std\_logic;

clk\_1Hz: OUT std\_logic

);

END Entity;

ARCHITECTURE clock\_div of Clock IS

BEGIN

process(clk\_50MHz)

VARIABLE count50: integer range 0 to 50000000:=0;

BEGIN

IF(Rising\_edge(clk\_50MHz)) THEN

count50:=count50+1;

IF (count50=1 and count50<=N) THEN

clk\_1Hz<='0';

END if;

IF(count50>N and count50<2\*N) THEN

clk\_1Hz<='1';

END if;

IF (count50=2\*N) THEN

count50:=0;

END if;

END if;

END process;

END ARCHITECTURE;

**Conclusion:**

We have learnt to work on FPGA systems through VHDL programs in this lab. Now, we can make our hardware implementation simpler by using block diagram function.