Lab 9 – 10

Building up the Nano-Processor

- 210302P – Kulasekara K.M.S.N
- 210306G – Kulathunga K.A.J.T

Lab Task :-

- Implement 4-bit Add-Subtract unit , 3-bit Adder , 3-bit Program counter (PC) , Multiplexers (2-way 3-bit , 2-way 4-bit , 8-way 4-bit) , Register Bank , Program ROM , Instruction Decoder in VHDL.
- Design a simple nano-processor by combining the above components.
- Simulate the components separately to ensure that we can get the expected results.
- Simulate the nano-processor and guarantee the accuracy of its outputs.
- Test on BASYS 3 board and verify the functionality of the nano-processor.

Assembly Program :-

Assembly Programme	Machine code Representation
MOVI R1, 3	100010000011
MOVI R2, 1	100100000001
NEG R2	010100000000
ADD R7, R1	001110010000
ADD R1, R2	000010100000
JZR R1, 7	110010000111
JZR R0, 3	110000000011
JZR R0, 7	110000000111

VHDL codes

Add – Sub Unit

RCA_4.vhd

C:/Users/DOWNLOAD/Lab9_10/Lab9_10.srcs/sources_1/imports/new/RCA_4.vhd

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity Add_Sub_Unit is
    Port ( A : in STD_LOGIC_VECTOR(3 downto 0);
           B : in STD_LOGIC_VECTOR(3 downto 0);
           S : out STD_LOGIC_VECTOR(3 downto 0);
           M : in STD_LOGIC;     -- (M=0) for addition and (M=1) for subtraction
           Overflow : out STD_LOGIC;
           Zero : out STD_LOGIC);
end Add_Sub_Unit;

architecture Behavioral of Add_Sub_Unit is

component FA
    port (
    A: in std_logic;
    B: in std_logic;
    C_in: in std_logic;
    S: out std_logic;
    C_out: out std_logic);

end component;

 SIGNAL FA0_S, FA0_C, FA1_S, FA1_C, FA2_S, FA2_C, FA3_S, FA3_C : std_logic;
 SIGNAL S_Copy : STD_LOGIC_VECTOR (3 downto 0);
 SIGNAL B0_xor_M,B1_xor_M,B2_xor_M,B3_xor_M : std_logic;

begin

FA_0 : FA
    port map (
```

Type here to search 88°F ENG 9:18 PM 6/10/2023

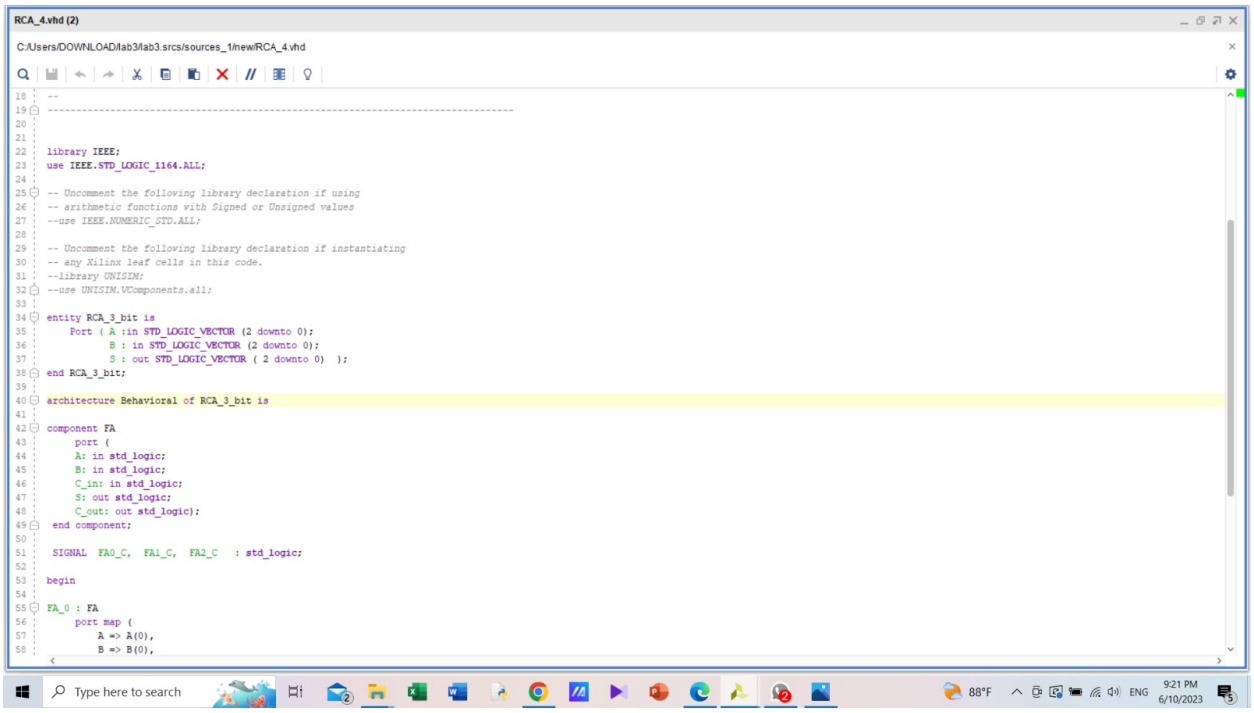
RCA_4.vhd

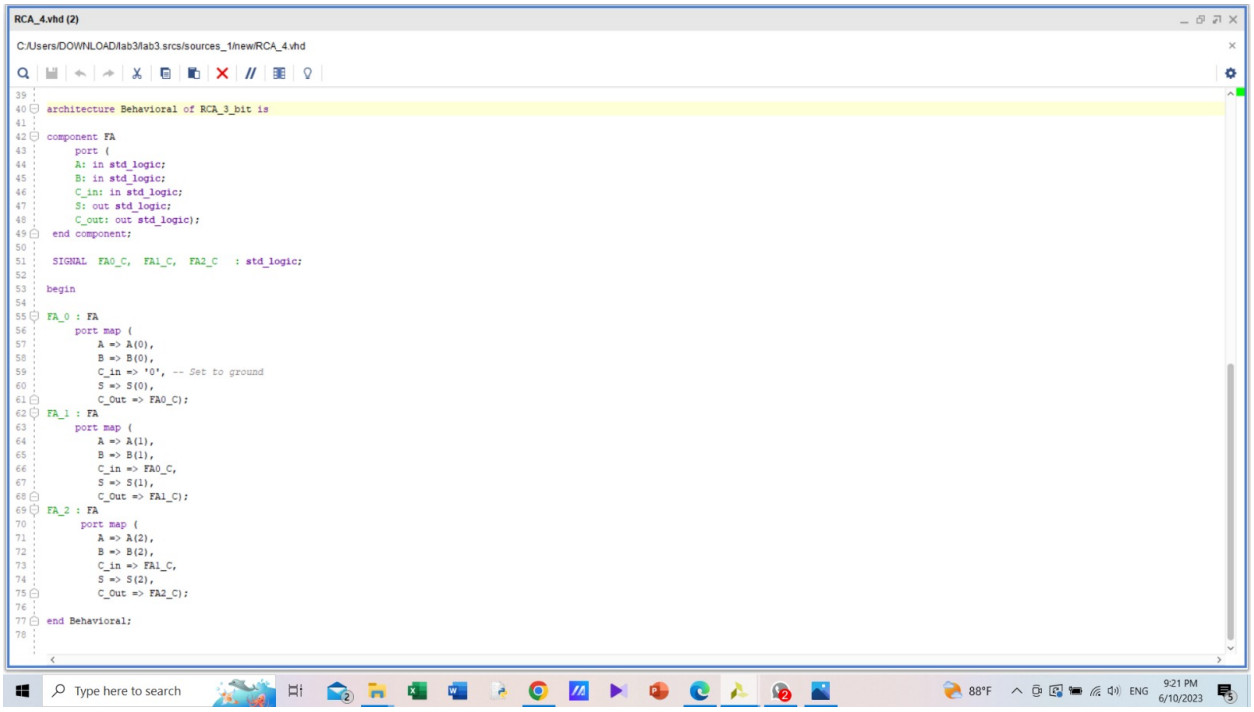
C:/Users/DOWNLOAD/Lab9_10/Lab9_10.srcs/sources_1/imports/new/RCA_4.vhd

```
FA_0 : FA
    port map (
        A => A(0),
        B => B0_xor_M ,
        C_in => M,
        S => S_Copy(0),
        C_Out => FA0_C);
FA_1 : FA
    port map (
        A => A(1),
        B => B1_xor_M,
        C_in => FA0_C,
        S => S_Copy(1),
        C_Out => FA1_C);
FA_2 : FA
     port map (
        A => A(2),
        B => B2_xor_M,
        C_in => FA1_C,
        S => S_Copy(2),
        C_Out => FA2_C);
FA_3 : FA
     port map (
        A => A(3),
        B => B3_xor_M,
        C_in => FA2_C,
        S => S_Copy(3),
        C_Out => FA3_C);

B0_xor_M <= B(0) xor M;
B1_xor_M <= B(1) xor M;
B2_xor_M <= B(2) xor M;
B3_xor_M <= B(3) xor M;

overflow <= FA3_C xor FA2_C;
zero <= not(S_Copy(0) or S_Copy(1) or S_Copy(2) or S_Copy(3));
s<=S_Copy;
end Behavioral;
```

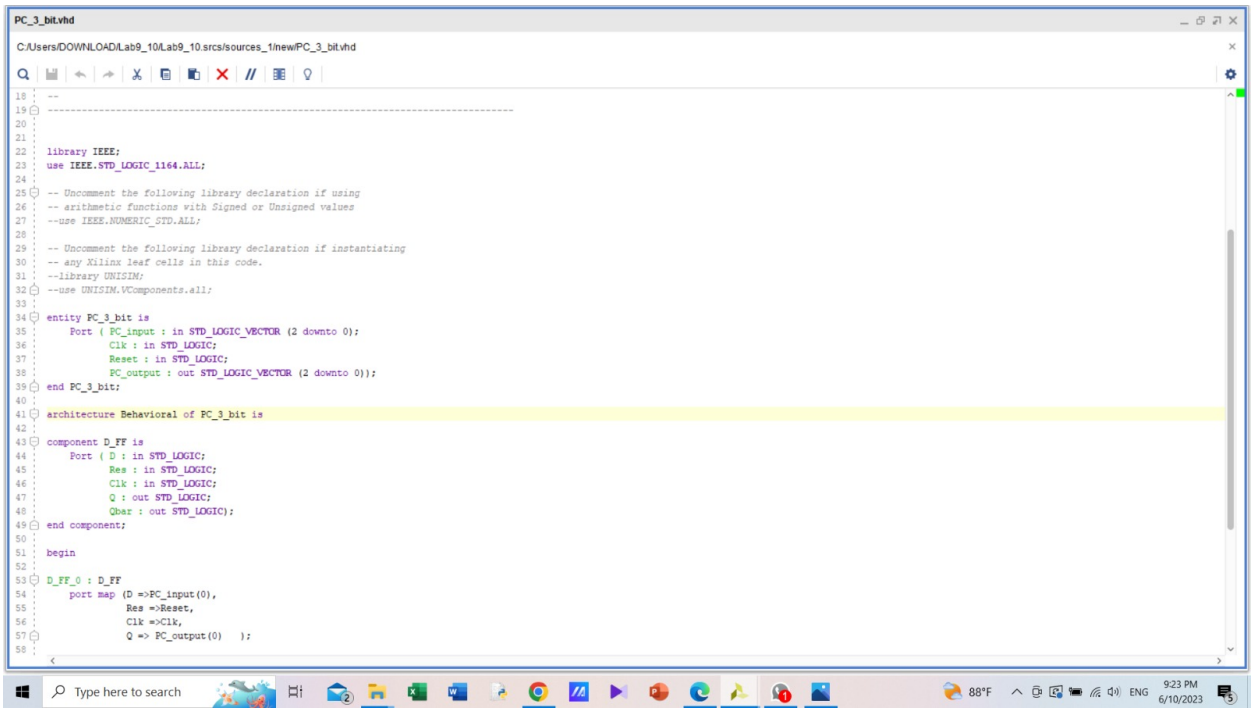
Type here to search 88°F ENG 9:18 PM 6/10/2023

3-bit adder

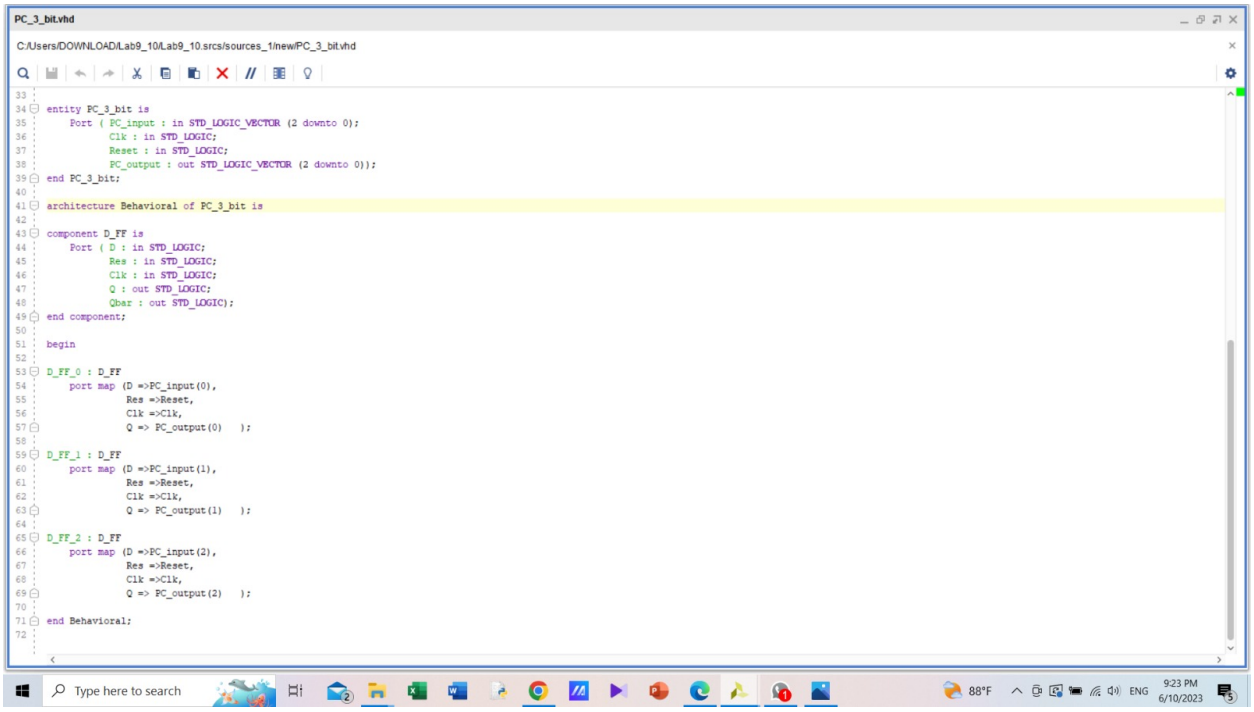

RCA_4.vhd (2)
C:/Users/DOWNLOAD/lab3/lab3.srcs/sources_1/new/RCA_4.vhd
--
----
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;
-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;
entity RCA_3_bit is
Port (A :in STD_LOGIC_VECTOR (2 downto 0);
B : in STD_LOGIC_VECTOR (2 downto 0);
S : out STD_LOGIC_VECTOR (2 downto 0));
end RCA_3_bit;
architecture Behavioral of RCA_3_bit is
component FA
port (
A: in std_logic;
B: in std_logic;
C_in: in std_logic;
S: out std_logic;
C_out: out std_logic);
end component;
SIGNAL FA0_C, FA1_C, FA2_C : std_logic;
begin
FA_0 : FA
port map (
A => A(0),
B => B(0),
Type here to search
88°F
ENG
9:21 PM
6/10/2023


RCA_4.vhd (2)
C:/Users/DOWNLOAD/lab3/lab3.srcs/sources_1/new/RCA_4.vhd
architecture Behavioral of RCA_3_bit is
component FA
port (
A: in std_logic;
B: in std_logic;
C_in: in std_logic;
S: out std_logic;
C_out: out std_logic);
end component;
SIGNAL FA0_C, FA1_C, FA2_C : std_logic;
begin
FA_0 : FA
port map (
A => A(0),
B => B(0),
C_in => '0', -- Set to ground
S => S(0),
C_Out => FA0_C);
FA_1 : FA
port map (
A => A(1),
B => B(1),
C_in => FA0_C,
S => S(1),
C_Out => FA1_C);
FA_2 : FA
port map (
A => A(2),
B => B(2),
C_in => FA1_C,
S => S(2),
C_Out => FA2_C);
end Behavioral;
Type here to search
88°F
ENG
9:21 PM
6/10/2023

3-bit Program Counter

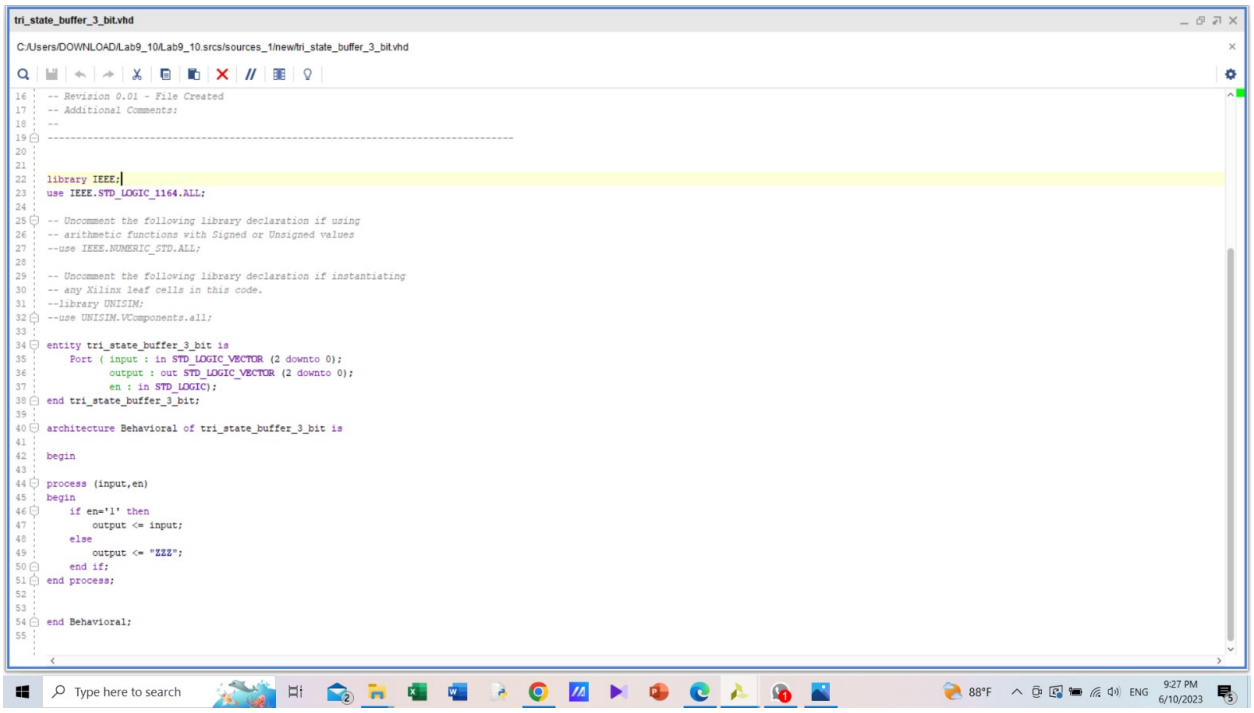



```
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity PC_3_bit is
    Port ( PC_input : in STD_LOGIC_VECTOR (2 downto 0);
           Clk : in STD_LOGIC;
           Reset : in STD_LOGIC;
           PC_output : out STD_LOGIC_VECTOR (2 downto 0));
end PC_3_bit;

architecture Behavioral of PC_3_bit is

component D_FF is
    Port ( D : in STD_LOGIC;
           Res : in STD_LOGIC;
           Clk : in STD_LOGIC;
           Q : out STD_LOGIC;
           Qbar : out STD_LOGIC);
end component;

begin

D_FF_0 : D_FF
    port map (D =>PC_input(0),
              Res =>Reset,
              Clk =>Clk,
              Q => PC_output(0)   );
```

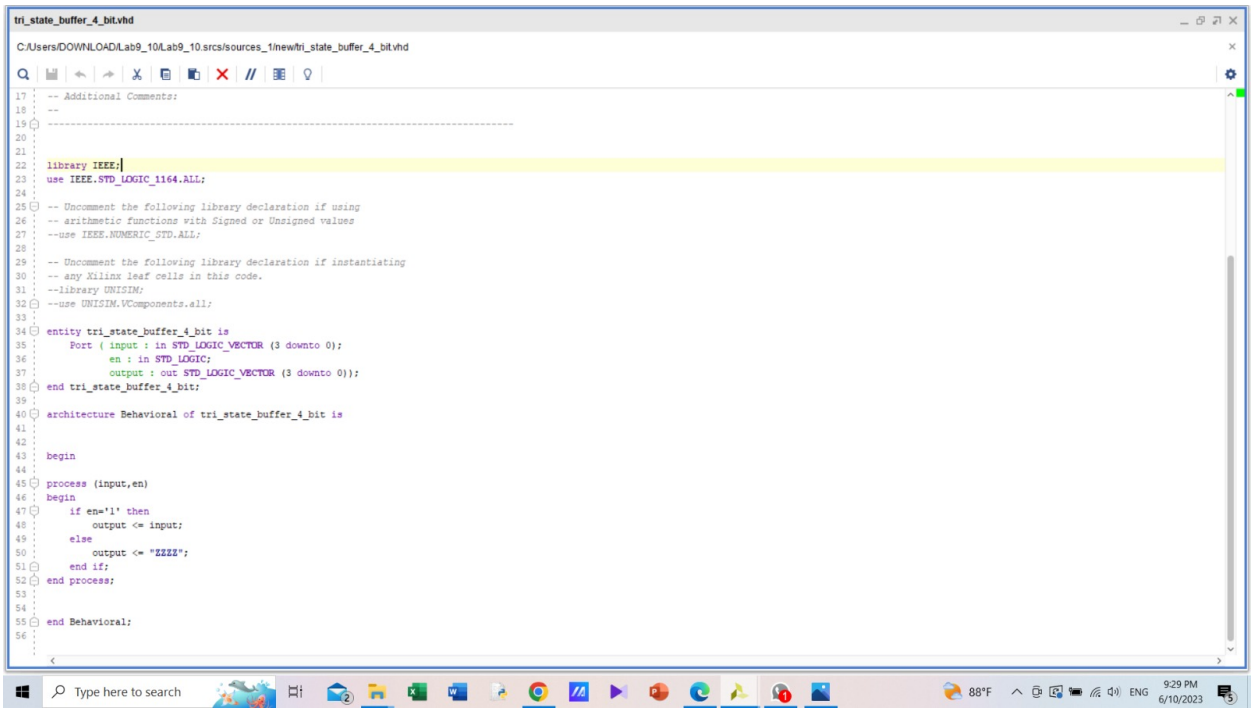



```
entity PC_3_bit is
    Port ( PC_input : in STD_LOGIC_VECTOR (2 downto 0);
           Clk : in STD_LOGIC;
           Reset : in STD_LOGIC;
           PC_output : out STD_LOGIC_VECTOR (2 downto 0));
end PC_3_bit;

architecture Behavioral of PC_3_bit is

component D_FF is
    Port ( D : in STD_LOGIC;
           Res : in STD_LOGIC;
           Clk : in STD_LOGIC;
           Q : out STD_LOGIC;
           Qbar : out STD_LOGIC);
end component;

begin

D_FF_0 : D_FF
    port map (D =>PC_input(0),
              Res =>Reset,
              Clk =>Clk,
              Q => PC_output(0)   );

D_FF_1 : D_FF
    port map (D =>PC_input(1),
              Res =>Reset,
              Clk =>Clk,
              Q => PC_output(1)   );

D_FF_2 : D_FF
    port map (D =>PC_input(2),
              Res =>Reset,
              Clk =>Clk,
              Q => PC_output(2)   );

end Behavioral;
```

Tri state 3-bit buffer

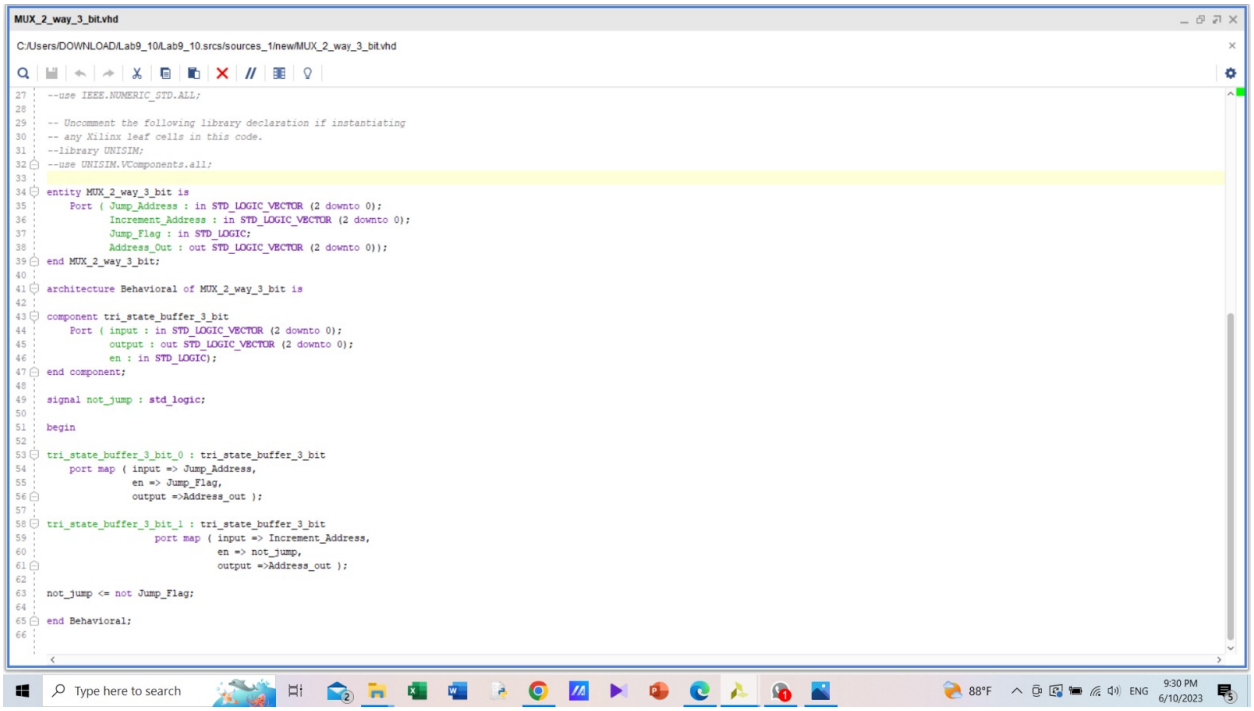


Tri state 4-bit buffer

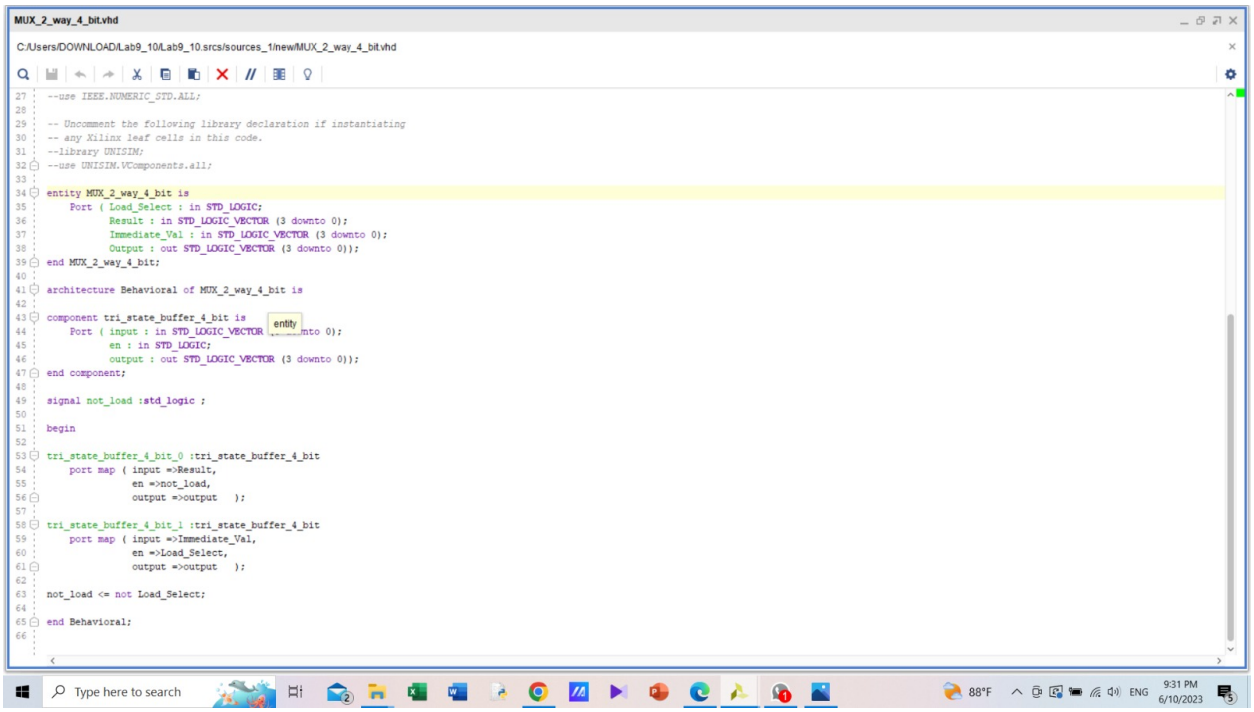


Multiplexers

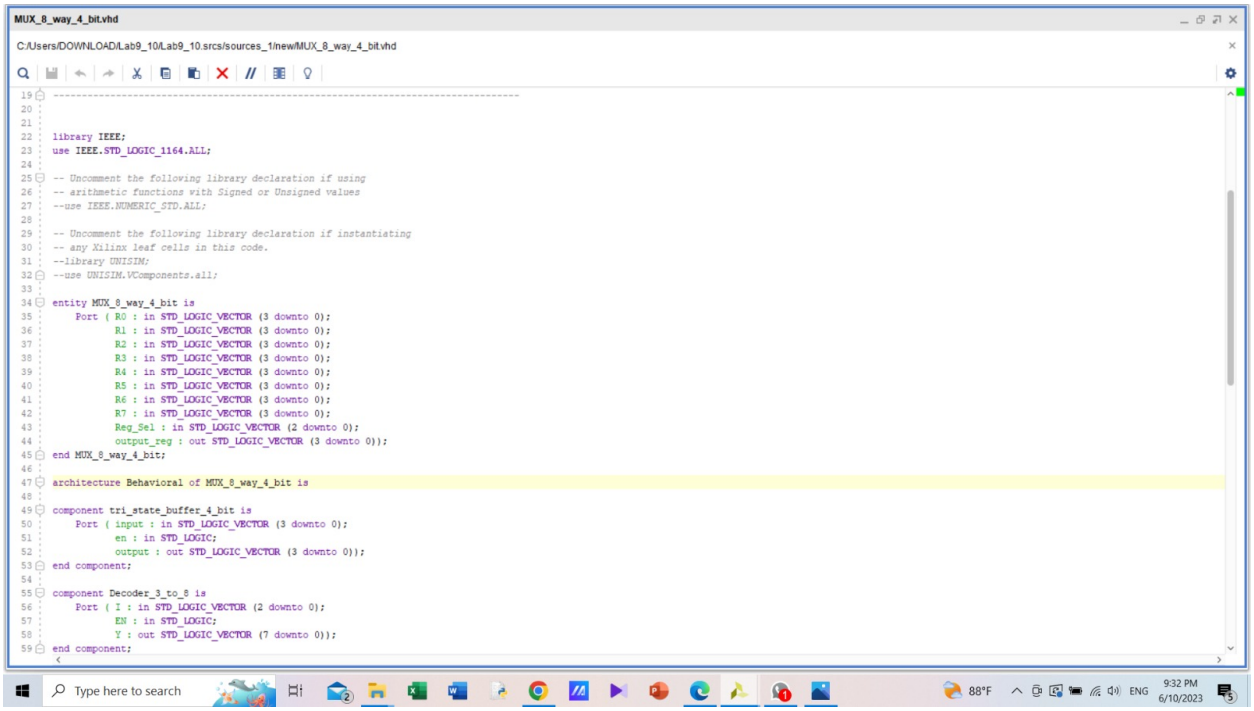
2 way 3-bit

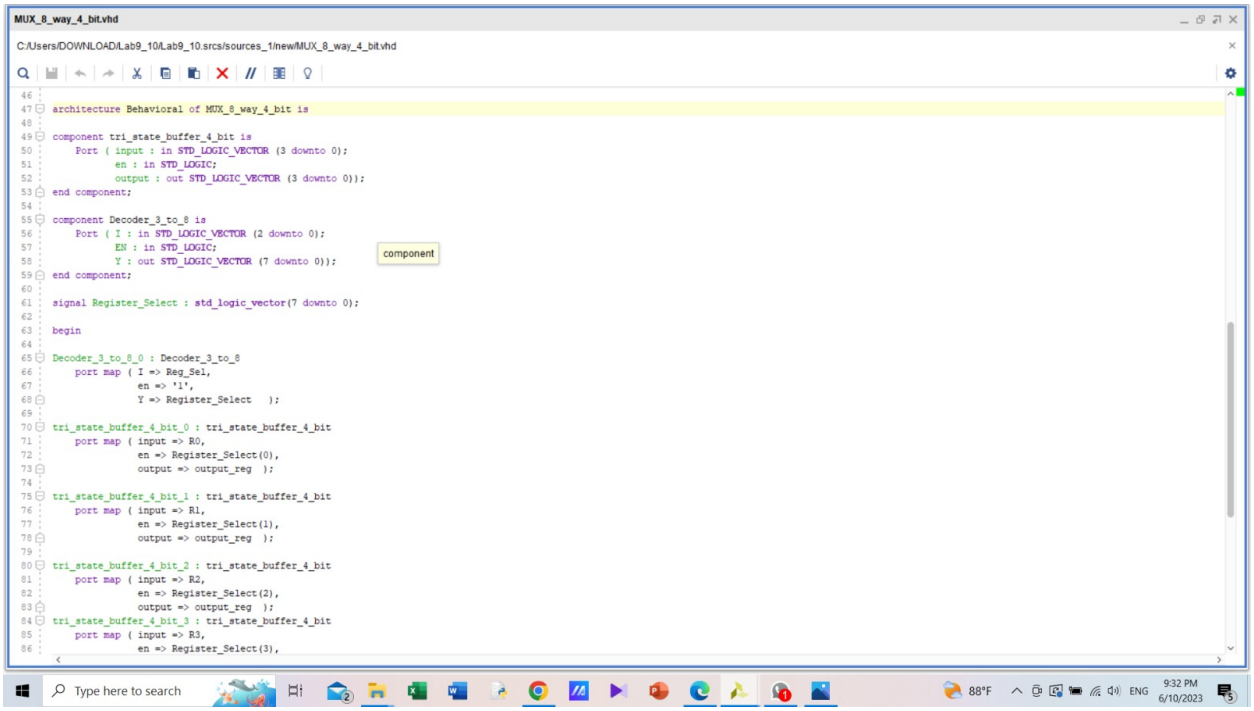


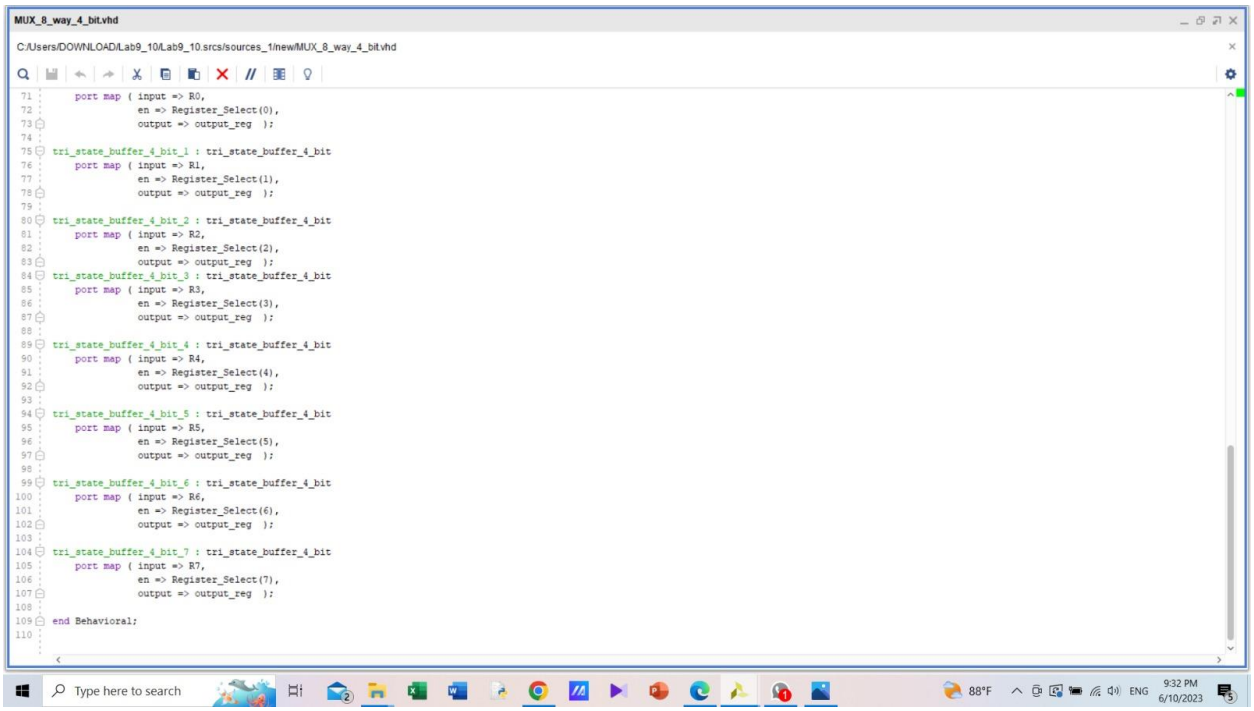
2 way 4-bit



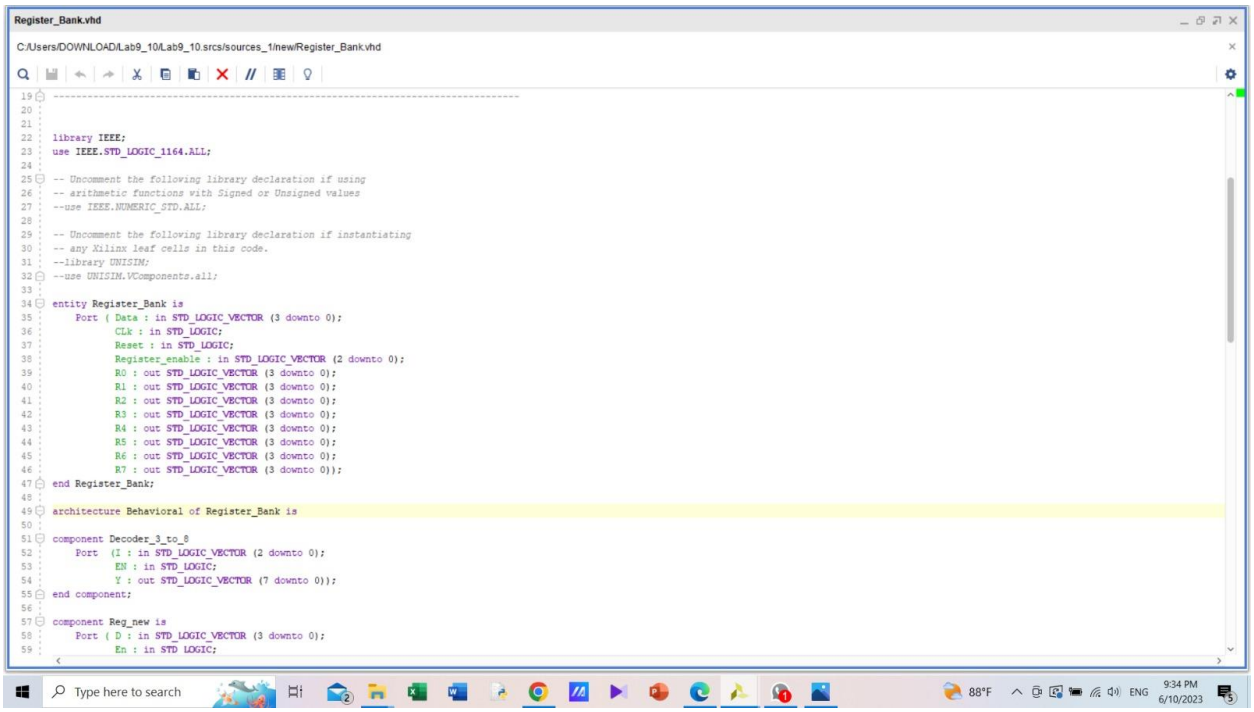
8 way 4-bit


MUX_8_way_4_bit.vhd
C:/Users/DOWNLOAD/Lab9_10/Lab9_10.srcs/sources_1/new/MUX_8_way_4_bit.vhd
Type here to search
88°F
ENG
9:32 PM
6/10/2023


MUX_8_way_4_bit.vhd
C:/Users/DOWNLOAD/Lab9_10/Lab9_10.srcs/sources_1/new/MUX_8_way_4_bit.vhd
component
Type here to search
88°F
ENG
9:32 PM
6/10/2023



Register bank



Register_Bank.vhd

C:/Users/DOWNLOAD/Lab9_10/Lab9_10.srcs/sources_1/new/Register_Bank.vhd

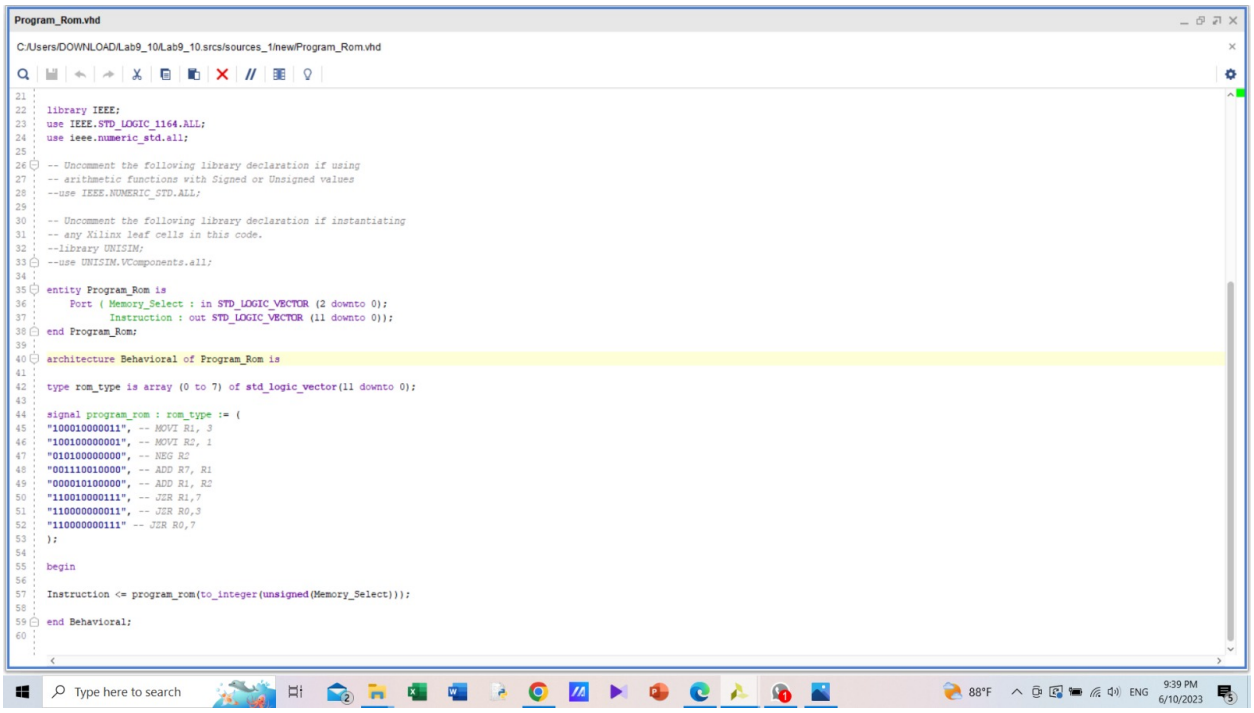
```
end component;

component Reg_new is
    Port ( D : in STD_LOGIC_VECTOR (3 downto 0);
           En : in STD_LOGIC;
           Clk : in STD_LOGIC;
           Res : in STD_LOGIC;
           Q : out STD_LOGIC_VECTOR (3 downto 0));
end component;

signal Reg_en :std_logic_vector(7 downto 0);
begin

Decoder_3_to_8_0 : Decoder_3_to_8
    port map ( I => Register_enable,
               EN =>'1',
               Y => Reg_en  );

Reg_0 : Reg_new
    port map ( D => Data,
               En =>Reg_en(0),
               Clk => Clk,
               Res => Reset,
               Q => R0   );

Reg_1 : Reg_new
    port map ( D => Data,
               En =>Reg_en(1),
               Clk => Clk,
               Res => Reset,
               Q => R1   );

Reg_2 : Reg_new
    port map ( D => Data,
               En =>Reg_en(2),
               Clk => Clk,
               Res => Reset,
               Q => R2   );

Reg_3 : Reg_new
    port map ( D => Data,
```

Register_Bank.vhd

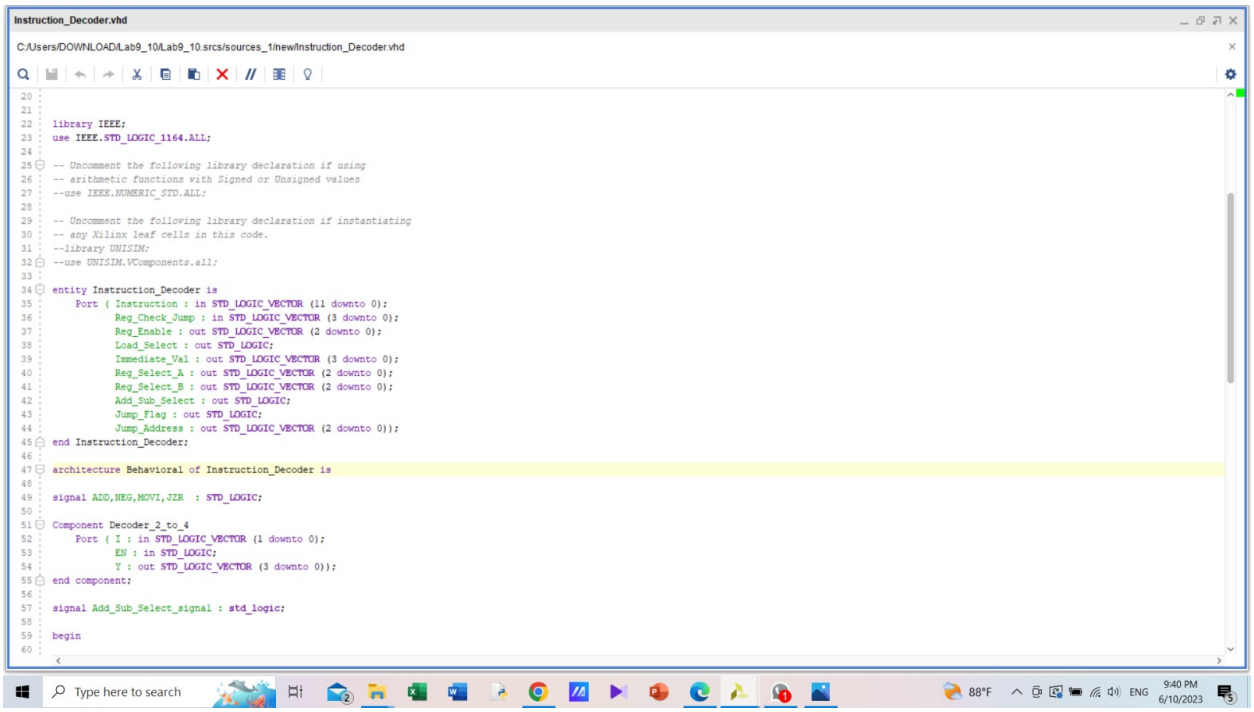
C:/Users/DOWNLOAD/Lab9_10/Lab9_10.srcs/sources_1/new/Register_Bank.vhd

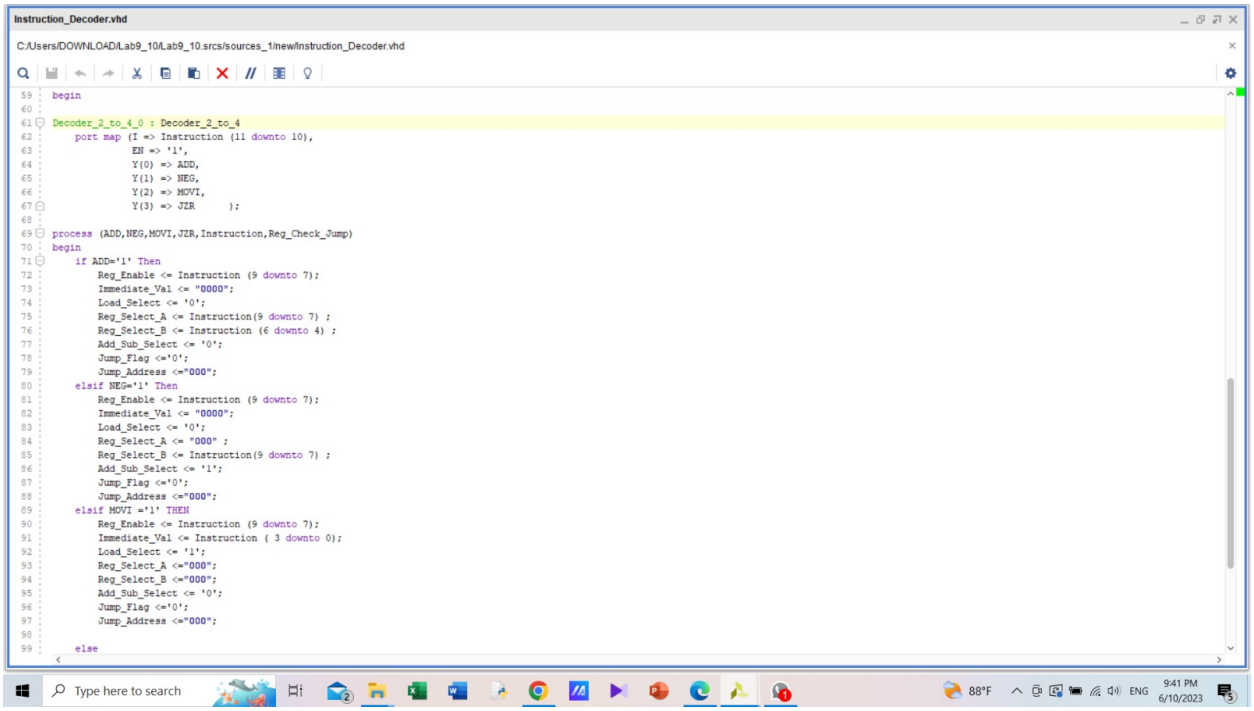
```
               Res => Reset,
               Q => R2   );

Reg_3 : Reg_new
    port map ( D => Data,
               En =>Reg_en(3),
               Clk => Clk,
               Res => Reset,
               Q => R3   );

Reg_4 : Reg_new
    port map ( D => Data,
               En =>Reg_en(4),
               Clk => Clk,
               Res => Reset,
               Q => R4   );

Reg_5 : Reg_new
    port map ( D => Data,
               En =>Reg_en(5),
               Clk => Clk,
               Res => Reset,
               Q => R5   );

Reg_6 : Reg_new
    port map ( D => Data,
               En =>Reg_en(6),
               Clk => Clk,
               Res => Reset,
               Q => R6   );

Reg_7 : Reg_new
    port map ( D => Data,
               En =>Reg_en(7),
               Clk => Clk,
               Res => Reset,
               Q => R7   );

end Behavioral;
```

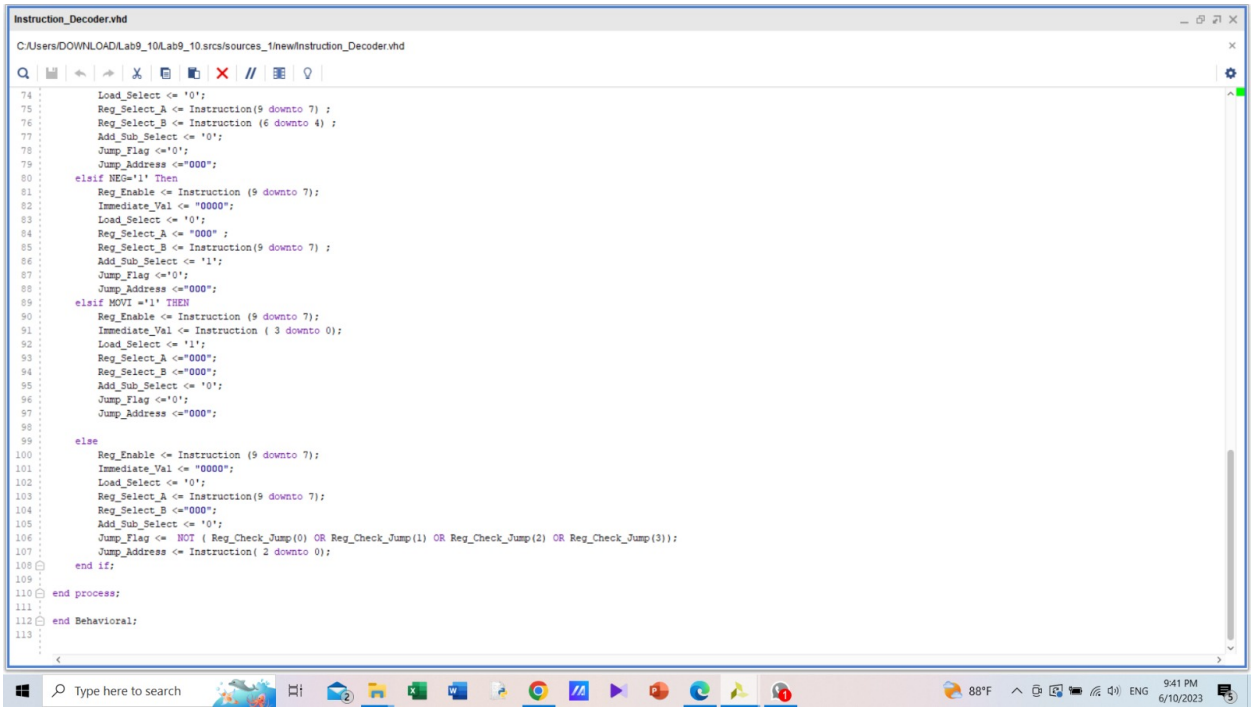
Program ROM



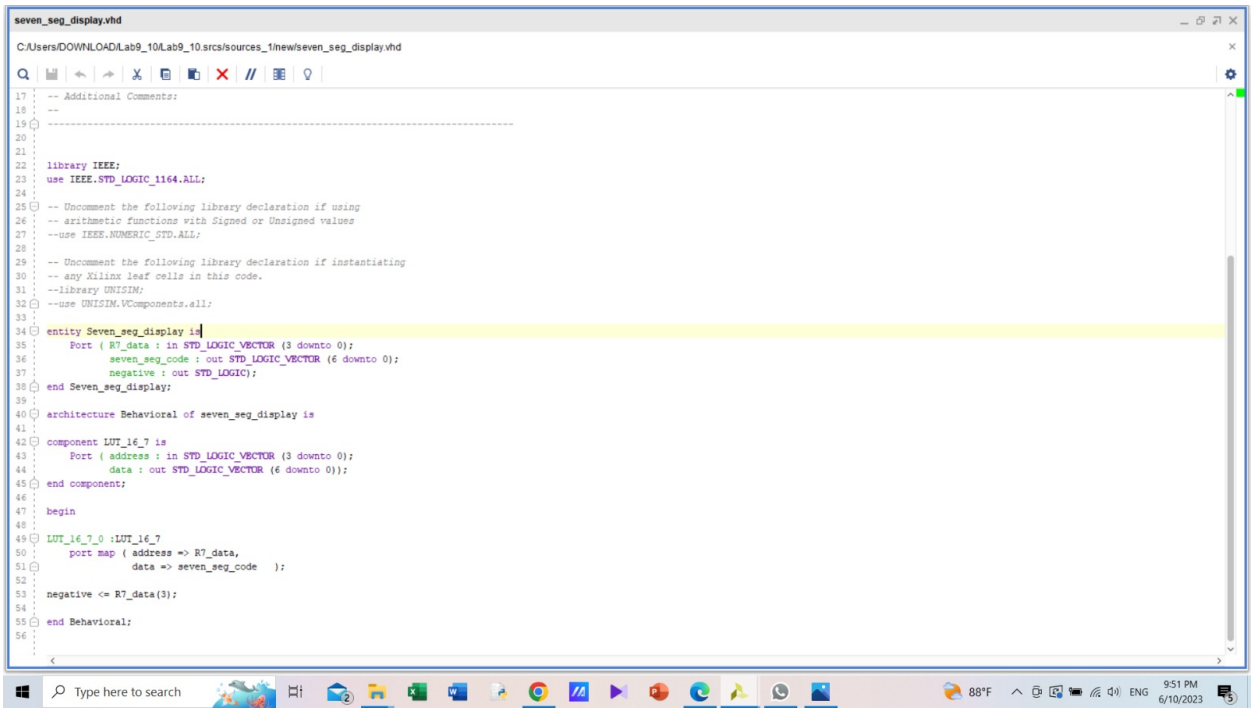
Instruction decoder



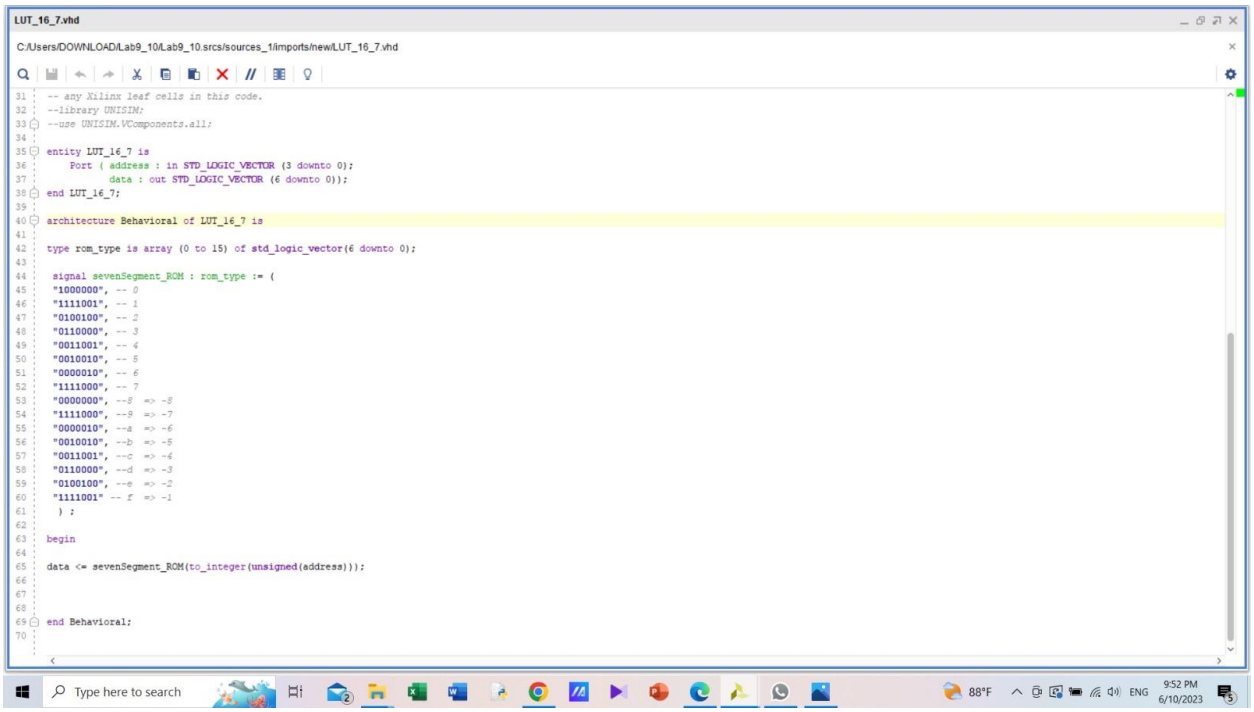

Instruction_Decoder.vhd
C:/Users/DOWNLOAD/Lab9_10/Lab9_10.srcs/sources_1/new/Instruction_Decoder.vhd
begin
Decoder_2_to_4_0 : Decoder_2_to_4
port map (I => Instruction (11 downto 10),
EN => '1',
Y(0) => ADD,
Y(1) => NEG,
Y(2) => MOVI,
Y(3) => JZR);
process (ADD,NEG,MOVI,JZR,Instruction,Reg_Check_Jump)
begin
if ADD='1' Then
Reg_Enable <= Instruction (9 downto 7);
Immediate_Val <= "0000";
Load_Select <= '0';
Reg_Select_A <= Instruction(9 downto 7) ;
Reg_Select_B <= Instruction (6 downto 4) ;
Add_Sub_Select <= '0';
Jump_Flag <='0';
Jump_Address <="000";
elsif NEG='1' Then
Reg_Enable <= Instruction (9 downto 7);
Immediate_Val <= "0000";
Load_Select <= '0';
Reg_Select_A <= "000" ;
Reg_Select_B <= Instruction(9 downto 7) ;
Add_Sub_Select <= '1';
Jump_Flag <='0';
Jump_Address <="000";
elsif MOVI ='1' THEN
Reg_Enable <= Instruction (9 downto 7);
Immediate_Val <= Instruction (3 downto 0);
Load_Select <= '1';
Reg_Select_A <="000";
Reg_Select_B <="000";
Add_Sub_Select <= '0';
Jump_Flag <='0';
Jump_Address <="000";
else
Type here to search
88°F
ENG
9:41 PM
6/10/2023


Instruction_Decoder.vhd
C:/Users/DOWNLOAD/Lab9_10/Lab9_10.srcs/sources_1/new/Instruction_Decoder.vhd
Load_Select <= '0';
Reg_Select_A <= Instruction(9 downto 7) ;
Reg_Select_B <= Instruction (6 downto 4) ;
Add_Sub_Select <= '0';
Jump_Flag <='0';
Jump_Address <="000";
elsif NEG='1' Then
Reg_Enable <= Instruction (9 downto 7);
Immediate_Val <= "0000";
Load_Select <= '0';
Reg_Select_A <= "000" ;
Reg_Select_B <= Instruction(9 downto 7) ;
Add_Sub_Select <= '1';
Jump_Flag <='0';
Jump_Address <="000";
elsif MOVI ='1' THEN
Reg_Enable <= Instruction (9 downto 7);
Immediate_Val <= Instruction (3 downto 0);
Load_Select <= '1';
Reg_Select_A <="000";
Reg_Select_B <="000";
Add_Sub_Select <= '0';
Jump_Flag <='0';
Jump_Address <="000";
else
Reg_Enable <= Instruction (9 downto 7);
Immediate_Val <= "0000";
Load_Select <= '0';
Reg_Select_A <= Instruction(9 downto 7);
Reg_Select_B <="000";
Add_Sub_Select <= '0';
Jump_Flag <= NOT (Reg_Check_Jump(0) OR Reg_Check_Jump(1) OR Reg_Check_Jump(2) OR Reg_Check_Jump(3));
Jump_Address <= Instruction(2 downto 0);
end if;
end process;
end Behavioral;
Type here to search
88°F
ENG
9:41 PM
6/10/2023

Seven segment display

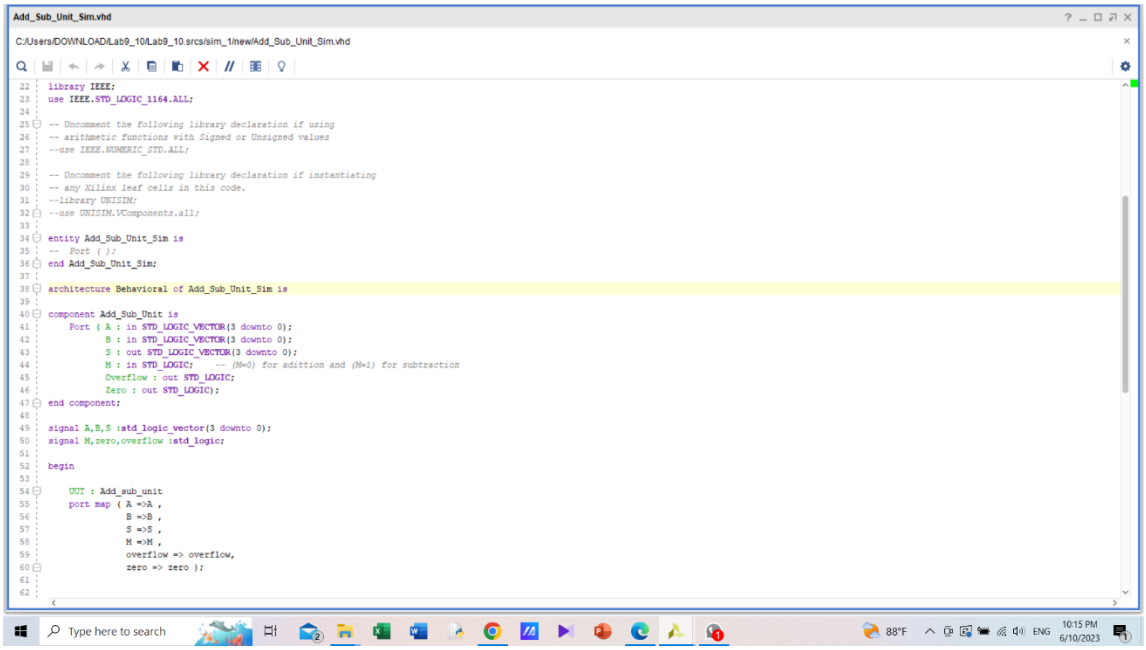


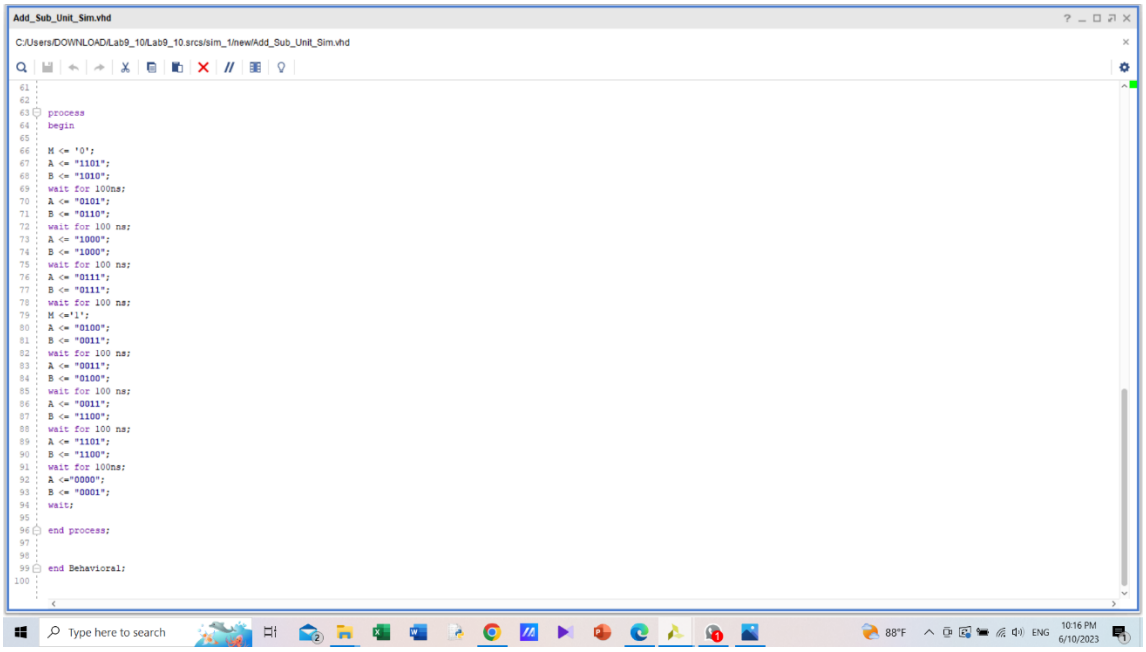
Lut_16_7



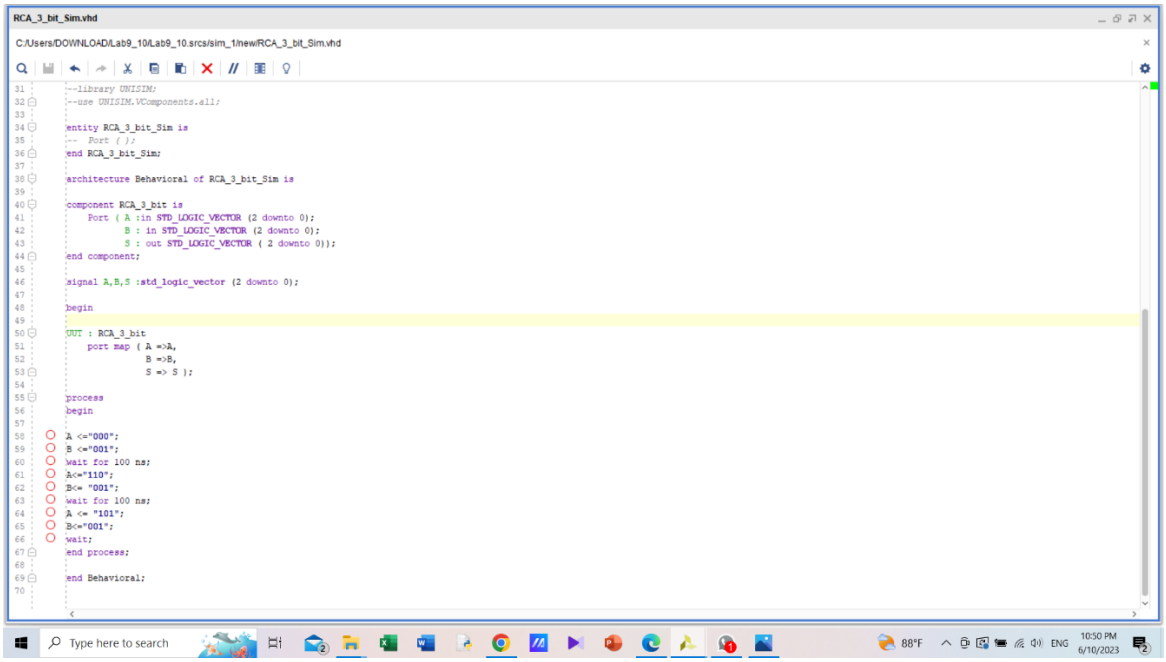
Test Bench Files

Add/Sub Unit

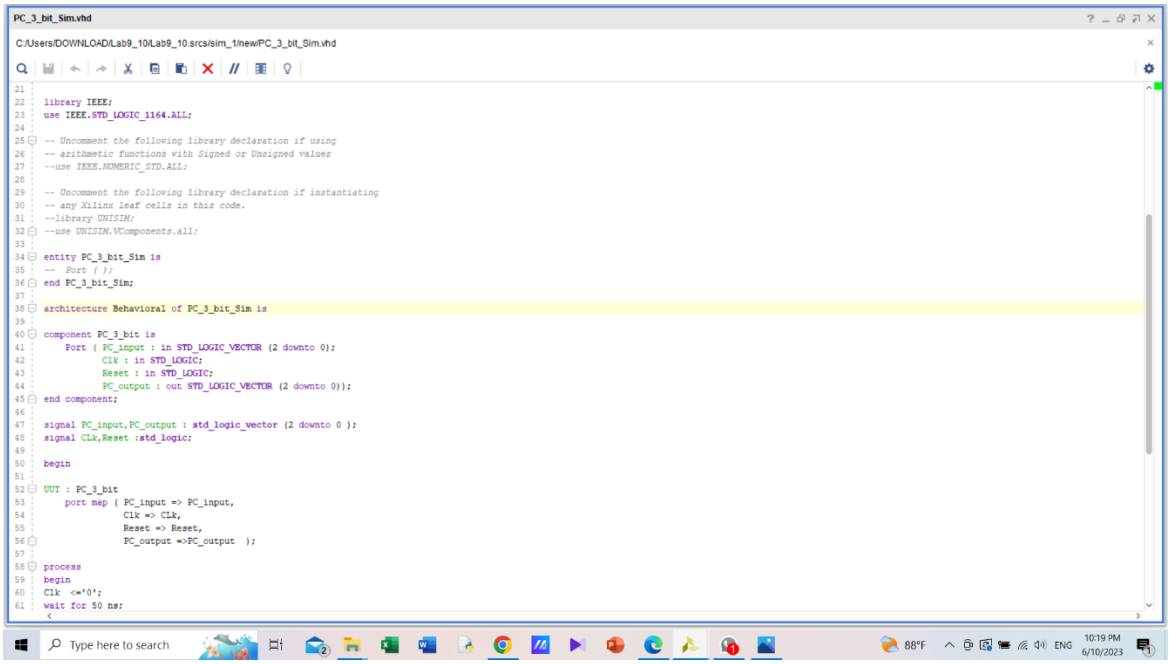


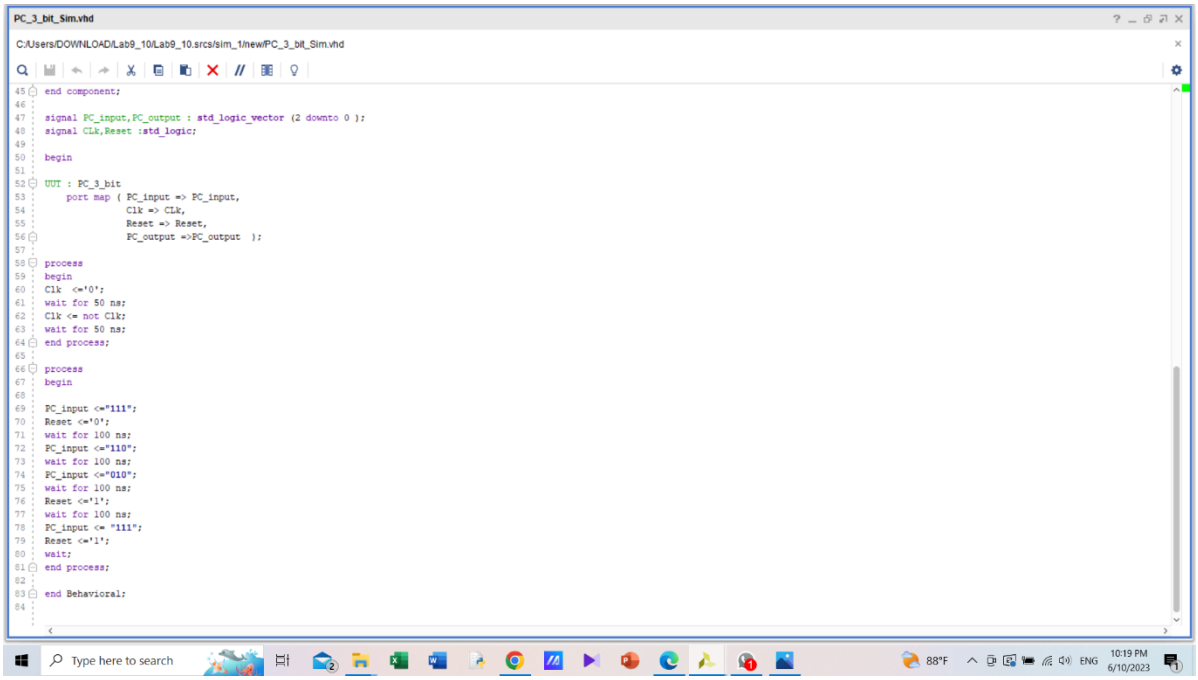


3-bit Adder

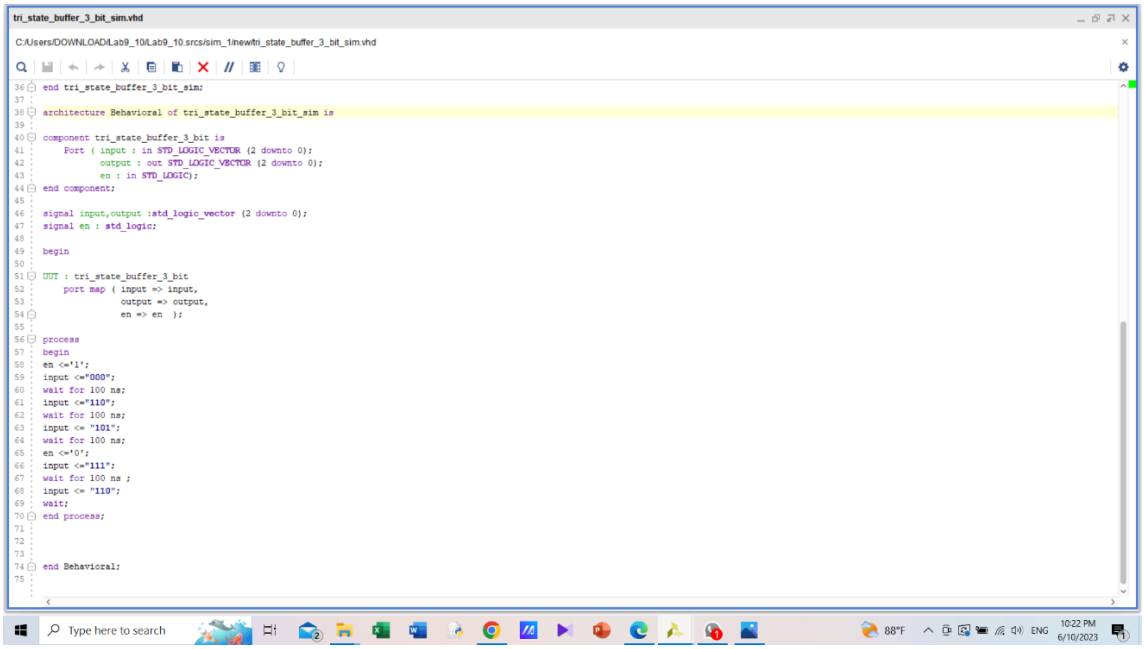


3-bit Program Counter

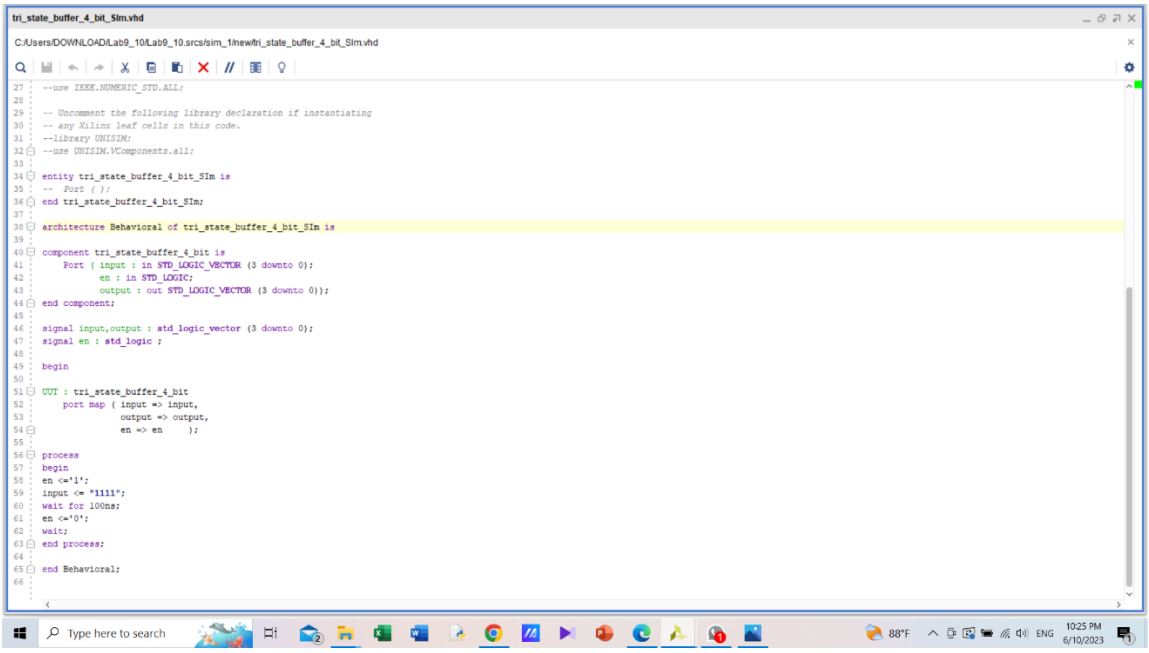



PC_3_bit_Sim.vhd
C:/Users/DOWNLOAD/Lab9_10/Lab9_10.srcs/sim_1/new/PC_3_bit_Sim.vhd
end component;
signal PC_input,PC_output : std_logic_vector (2 downto 0);
signal Clk,Reset :std_logic;
begin
UUT : PC_3_bit
port map (PC_input => PC_input,
Clk => Clk,
Reset => Reset,
PC_output =>PC_output);
process
begin
Clk <='0';
wait for 50 ns;
Clk <= not Clk;
wait for 50 ns;
end process;
process
begin
PC_input <="111";
Reset <='0';
wait for 100 ns;
PC_input <="110";
wait for 100 ns;
PC_input <="010";
wait for 100 ns;
Reset <='1';
wait for 100 ns;
PC_input <= "111";
Reset <='1';
wait;
end process;
end Behavioral;
Type here to search
88°F
ENG
10:19 PM
6/10/2023

Tri state 3-bit buffer

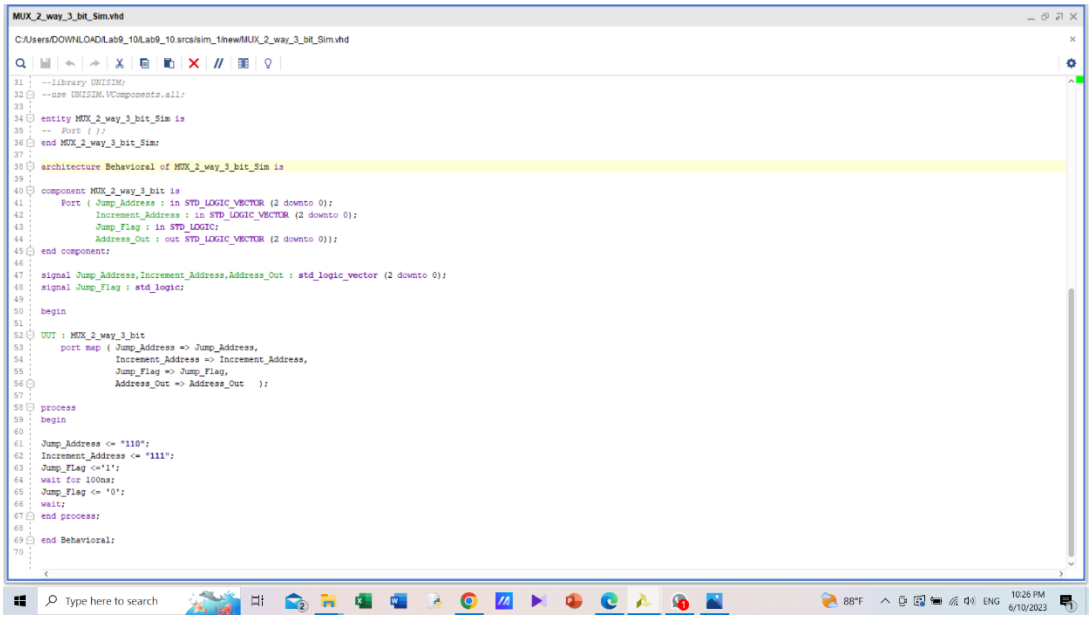


Tri state 4-bit buffer

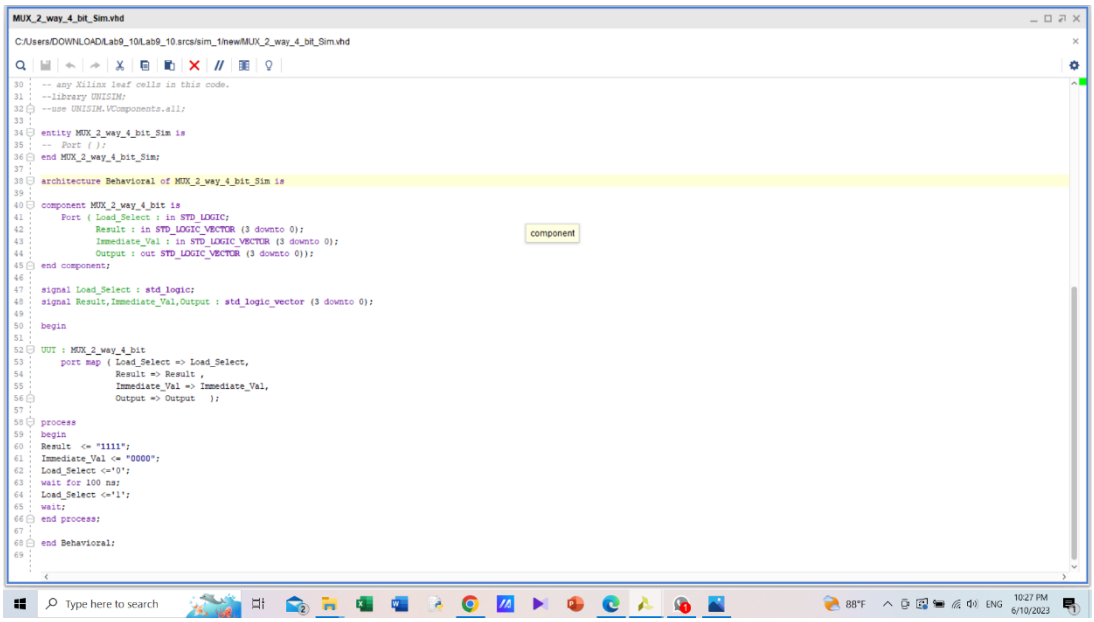


Multiplexers

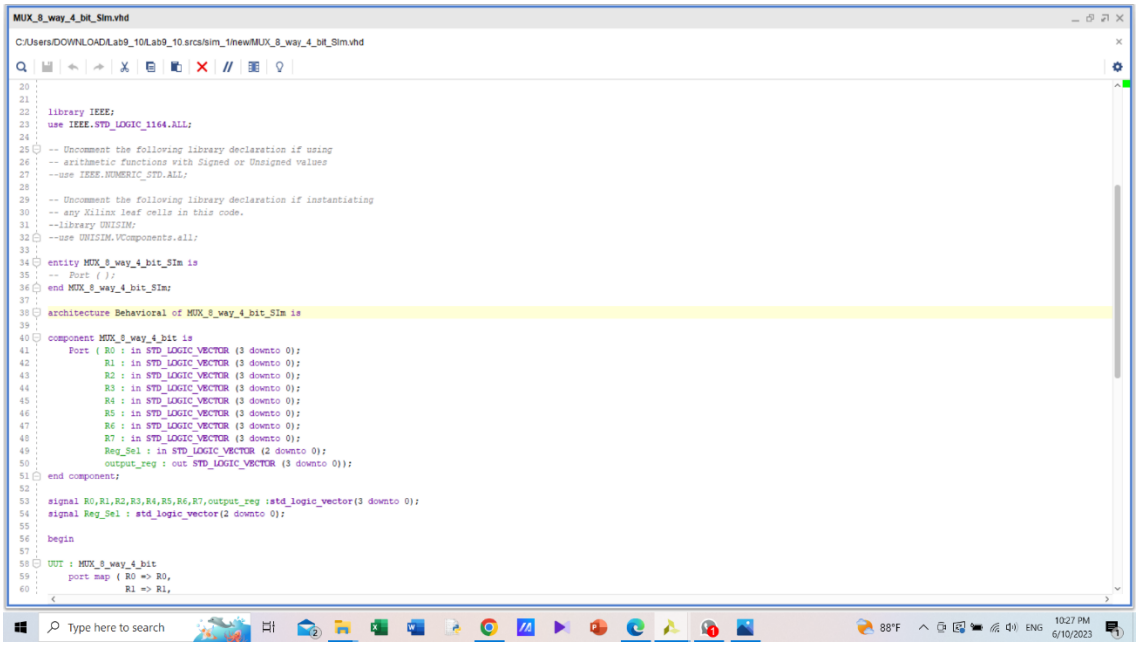
2 way 3-bit

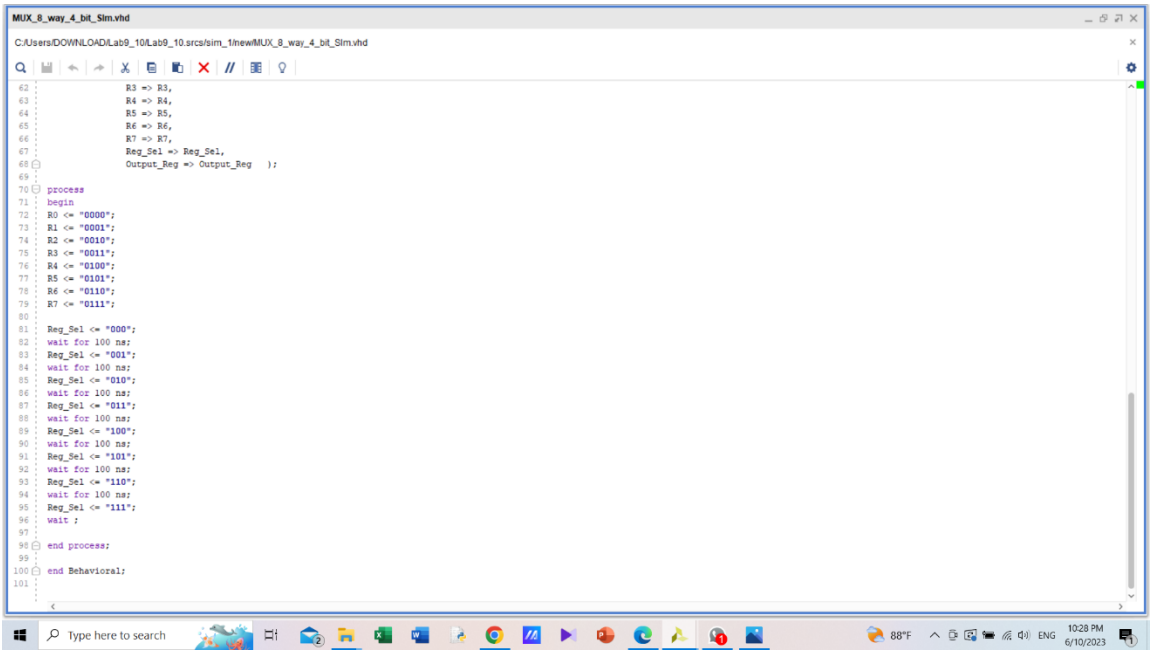


2 way 4-bit

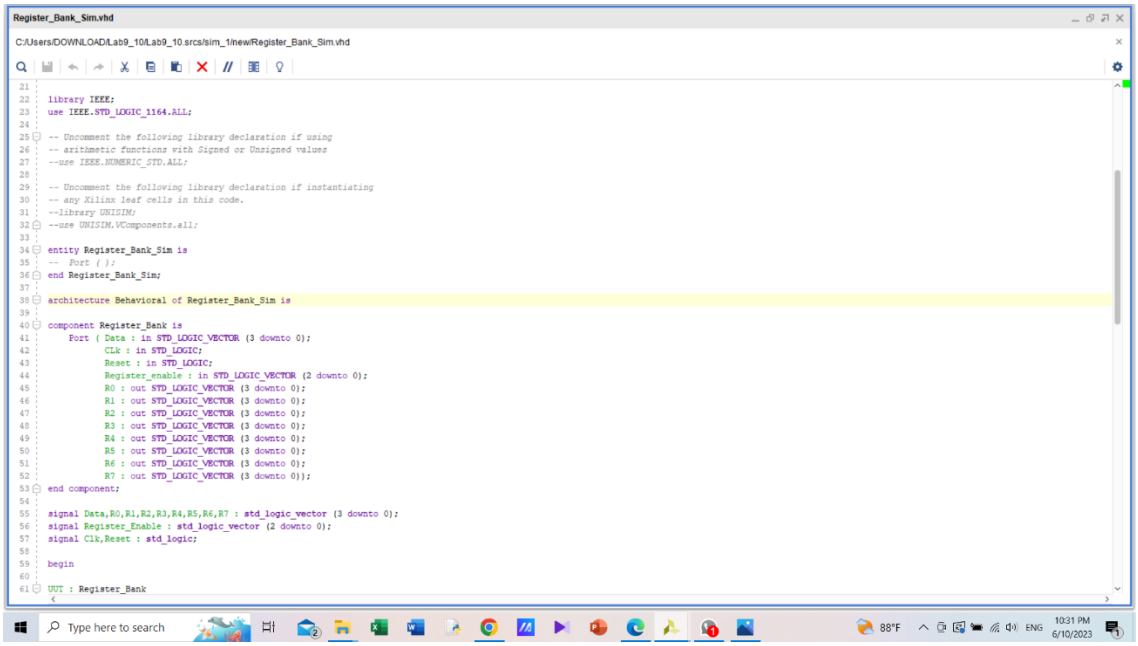


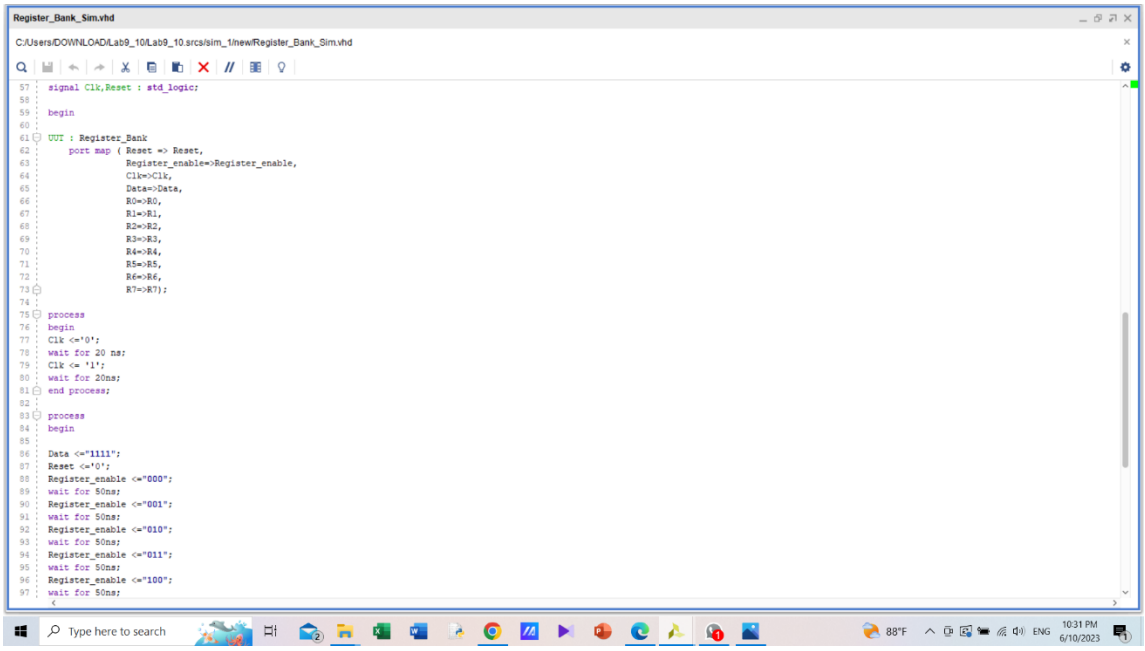
8 way 4-bit

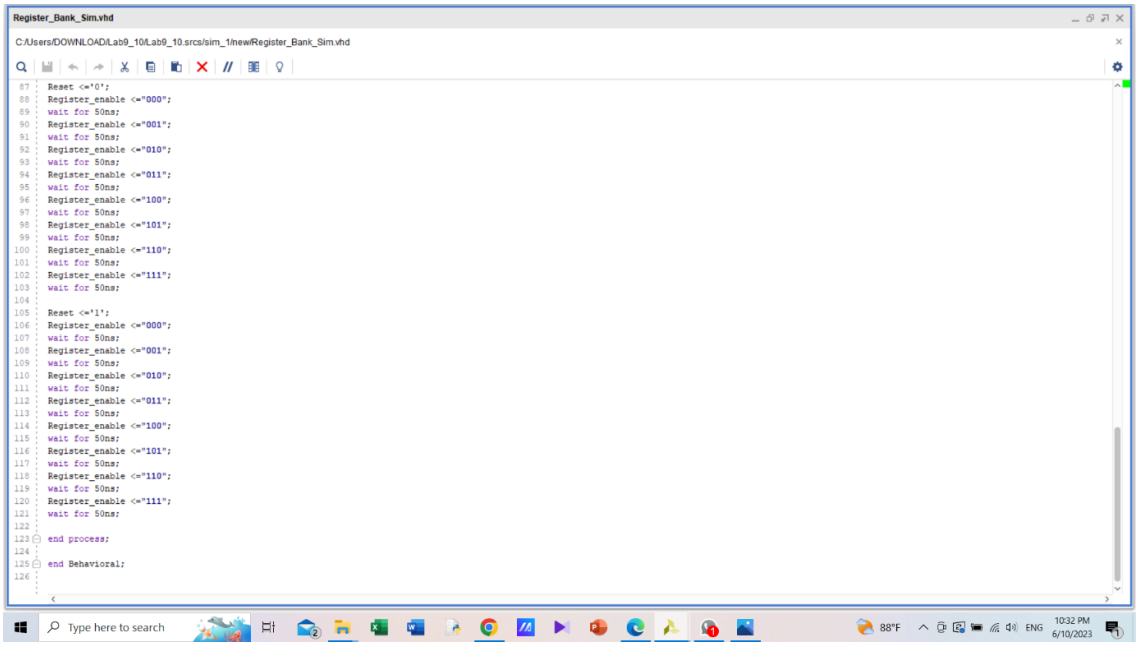

MUX_8_way_4_bit_Slm.vhd
C:/Users/DOWNLOAD/Lab9_10/Lab9_10.srcs/sim_1/new/MUX_8_way_4_bit_Slm.vhd
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;
-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;
entity MUX_8_way_4_bit_Slm is
-- Port ( );
end MUX_8_way_4_bit_Slm;
architecture Behavioral of MUX_8_way_4_bit_Slm is
component MUX_8_way_4_bit is
Port (R0 : in STD_LOGIC_VECTOR (3 downto 0);
R1 : in STD_LOGIC_VECTOR (3 downto 0);
R2 : in STD_LOGIC_VECTOR (3 downto 0);
R3 : in STD_LOGIC_VECTOR (3 downto 0);
R4 : in STD_LOGIC_VECTOR (3 downto 0);
R5 : in STD_LOGIC_VECTOR (3 downto 0);
R6 : in STD_LOGIC_VECTOR (3 downto 0);
R7 : in STD_LOGIC_VECTOR (3 downto 0);
Reg_Sel : in STD_LOGIC_VECTOR (2 downto 0);
output_reg : out STD_LOGIC_VECTOR (3 downto 0));
end component;
signal R0,R1,R2,R3,R4,R5,R6,R7,output_reg :std_logic_vector(3 downto 0);
signal Reg_Sel : std_logic_vector(2 downto 0);
begin
UUT : MUX_8_way_4_bit
port map (R0 => R0,
R1 => R1,
Type here to search
88°F
ENG
10:27 PM
6/10/2023


MUX_8_way_4_bit_Slm.vhd
C:/Users/DOWNLOAD/Lab9_10/Lab9_10.srcs/sim_1/new/MUX_8_way_4_bit_Slm.vhd
R3 => R3,
R4 => R4,
R5 => R5,
R6 => R6,
R7 => R7,
Reg_Sel => Reg_Sel,
Output_Reg => Output_Reg);
process
begin
R0 <= "0000";
R1 <= "0001";
R2 <= "0010";
R3 <= "0011";
R4 <= "0100";
R5 <= "0101";
R6 <= "0110";
R7 <= "0111";
Reg_Sel <= "000";
wait for 100 ns;
Reg_Sel <= "001";
wait for 100 ns;
Reg_Sel <= "010";
wait for 100 ns;
Reg_Sel <= "011";
wait for 100 ns;
Reg_Sel <= "100";
wait for 100 ns;
Reg_Sel <= "101";
wait for 100 ns;
Reg_Sel <= "110";
wait for 100 ns;
Reg_Sel <= "111";
wait ;
end process;
end Behavioral;
Type here to search
88°F
ENG
10:28 PM
6/10/2023

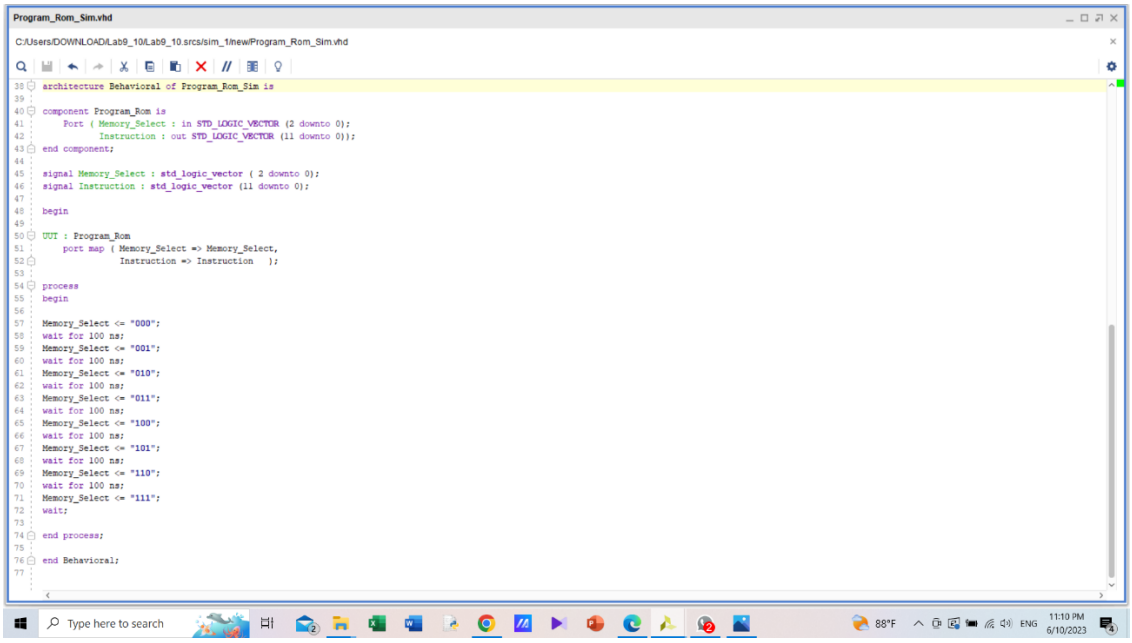
Register bank



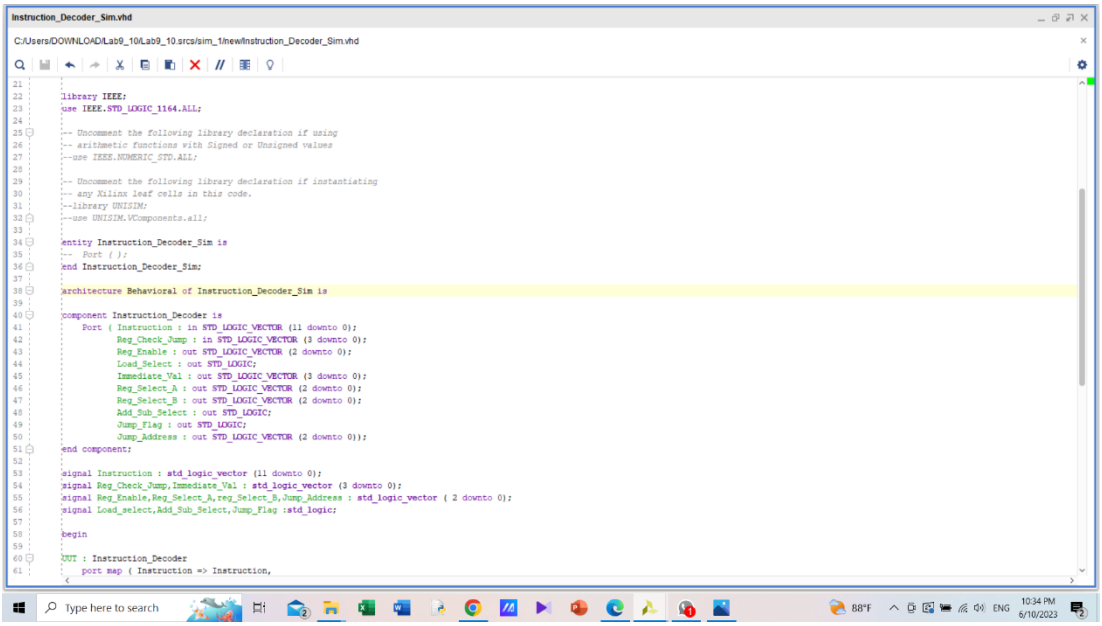


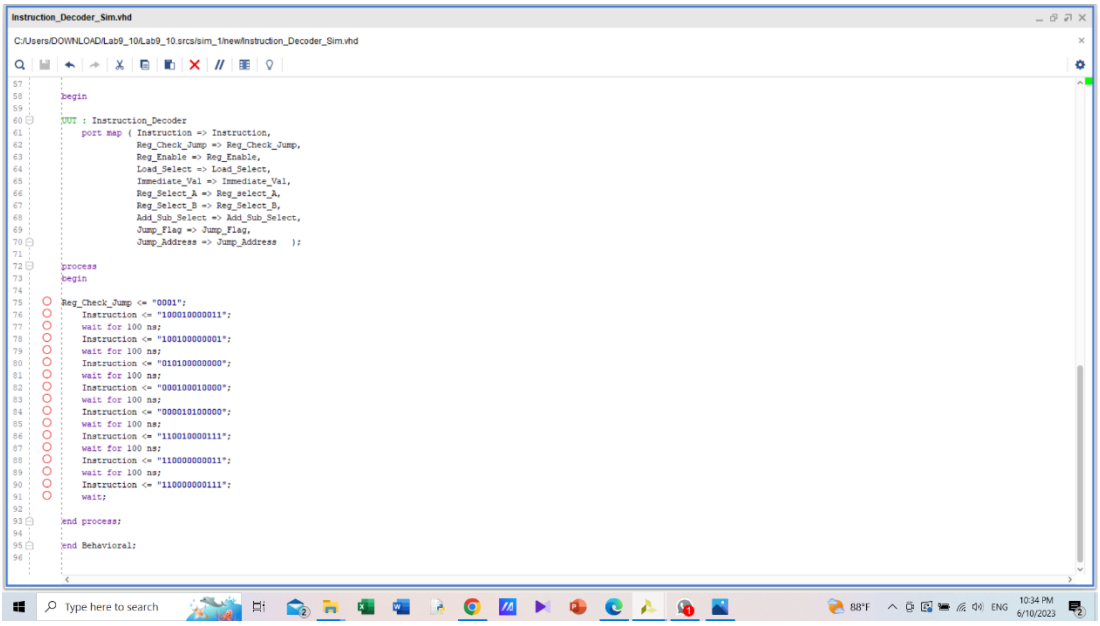

Register_Bank_Sim.vhd
C:/Users/DOWNLOAD/Lab9_10/Lab9_10.srcs/sim_1/new/Register_Bank_Sim.vhd
Reset <='0';
Register_enable <="000";
wait for 50ns;
Register_enable <="001";
wait for 50ns;
Register_enable <="010";
wait for 50ns;
Register_enable <="011";
wait for 50ns;
Register_enable <="100";
wait for 50ns;
Register_enable <="101";
wait for 50ns;
Register_enable <="110";
wait for 50ns;
Register_enable <="111";
wait for 50ns;
Reset <='1';
Register_enable <="000";
wait for 50ns;
Register_enable <="001";
wait for 50ns;
Register_enable <="010";
wait for 50ns;
Register_enable <="011";
wait for 50ns;
Register_enable <="100";
wait for 50ns;
Register_enable <="101";
wait for 50ns;
Register_enable <="110";
wait for 50ns;
Register_enable <="111";
wait for 50ns;
end process;
end Behavioral;
Type here to search
88°F
ENG
10:32 PM
6/10/2023

Program ROM


Program_Rom_Sim.vhd
C:/Users/DOWNLOAD/Lab9_10/Lab9_10.srcs/sim_1/new/Program_Rom_Sim.vhd
architecture Behavioral of Program_Rom_Sim is
component Program_Rom is
Port (Memory_Select : in STD_LOGIC_VECTOR (2 downto 0);
Instruction : out STD_LOGIC_VECTOR (11 downto 0));
end component;
signal Memory_Select : std_logic_vector (2 downto 0);
signal Instruction : std_logic_vector (11 downto 0);
begin
UUT : Program_Rom
port map (Memory_Select => Memory_Select,
Instruction => Instruction);
process
begin
Memory_Select <= "000";
wait for 100 ns;
Memory_Select <= "001";
wait for 100 ns;
Memory_Select <= "010";
wait for 100 ns;
Memory_Select <= "011";
wait for 100 ns;
Memory_Select <= "100";
wait for 100 ns;
Memory_Select <= "101";
wait for 100 ns;
Memory_Select <= "110";
wait for 100 ns;
Memory_Select <= "111";
wait;
end process;
end Behavioral;
Type here to search
88°F
ENG
11:10 PM
6/10/2023

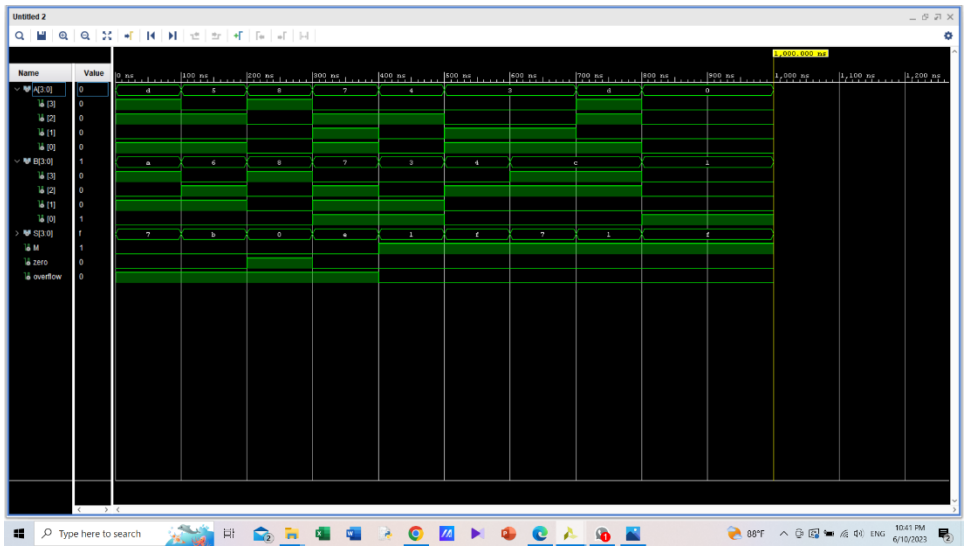
Instruction decoder



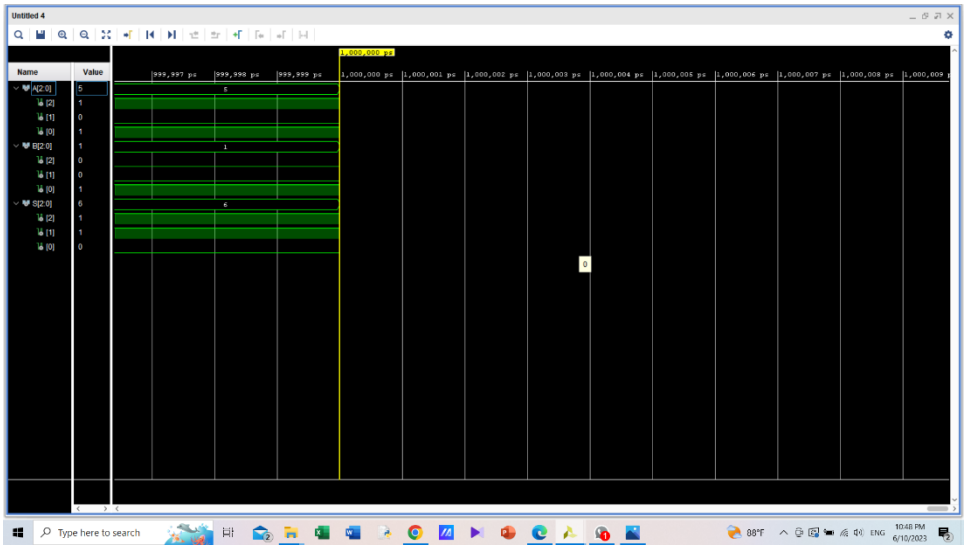


Timing Diagrams

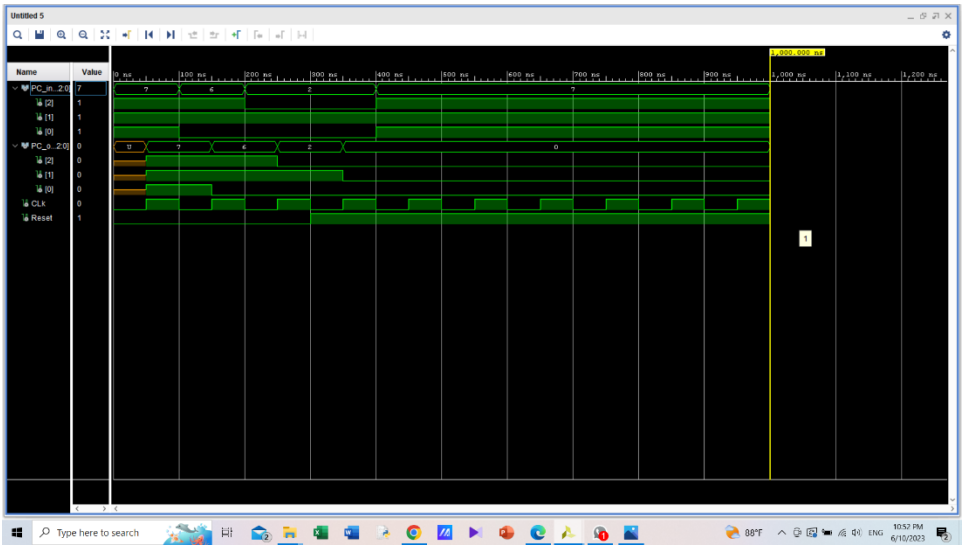
Add/sub unit



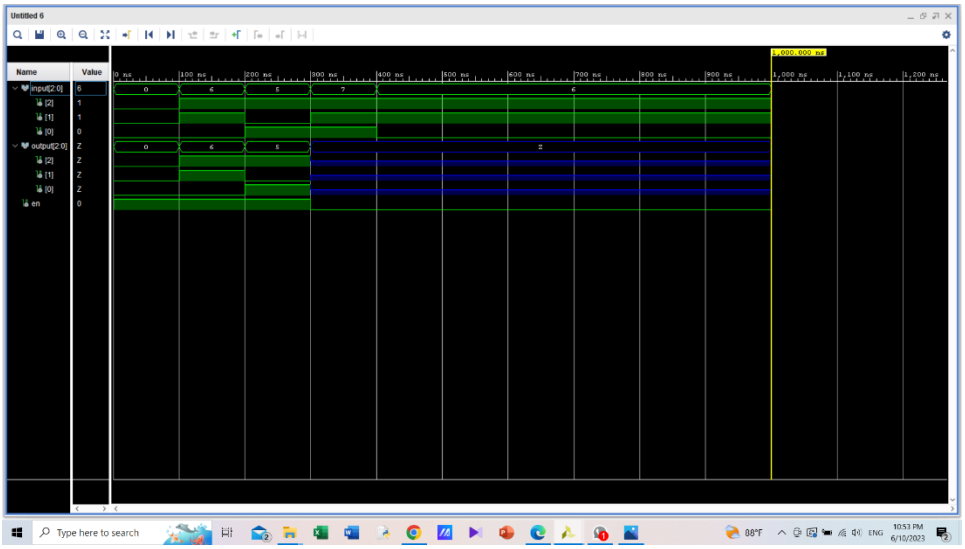
3-bit Adder



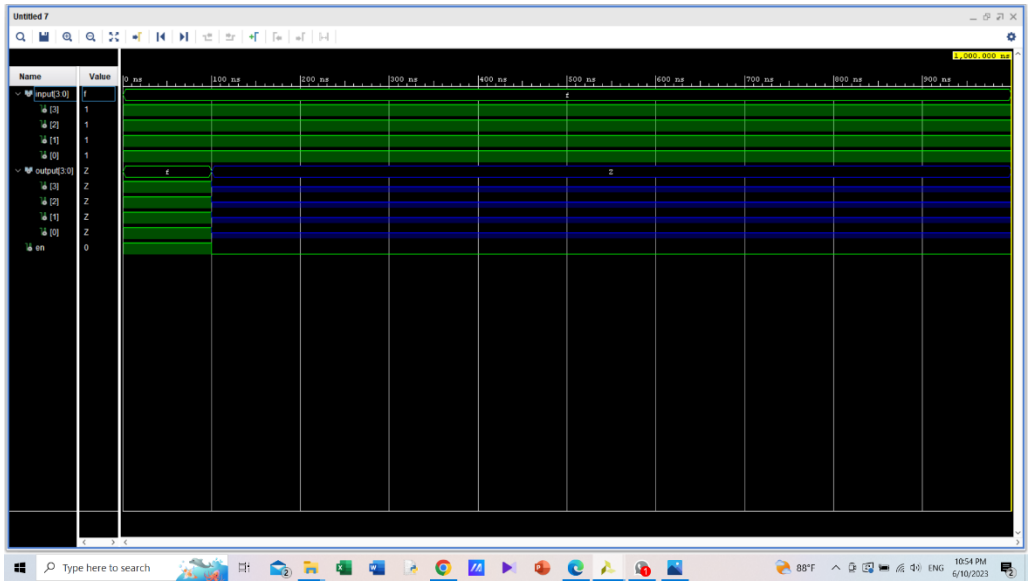
3-bit Program Counter



Tri state 3-bit buffer

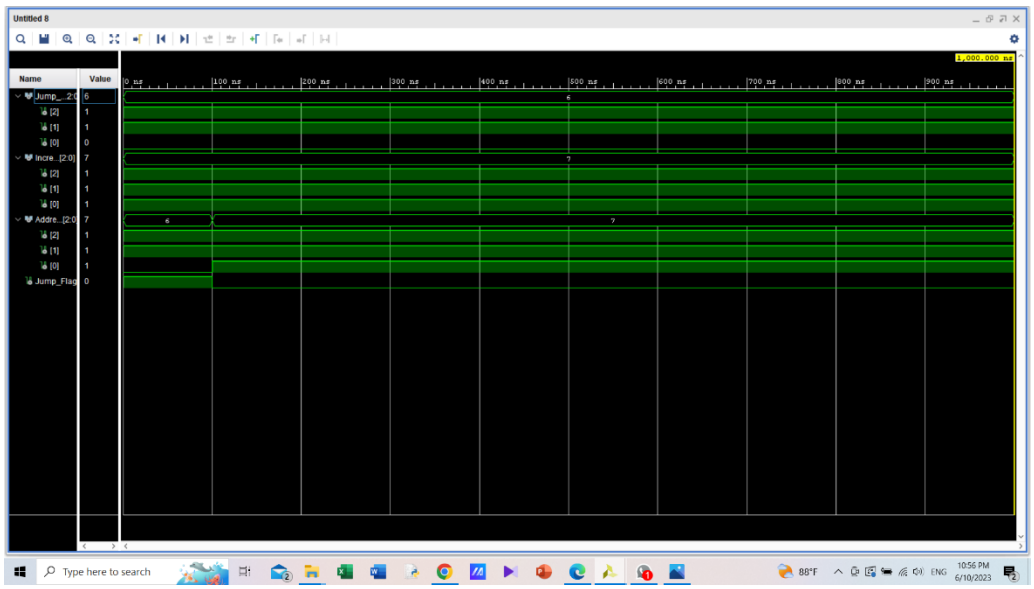


Tri state 4-bit buffer

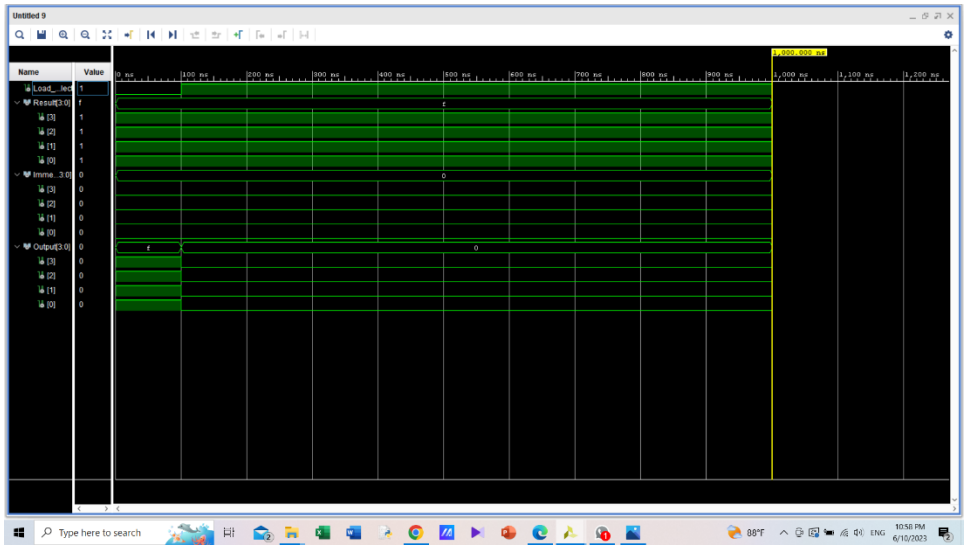


Multiplexers

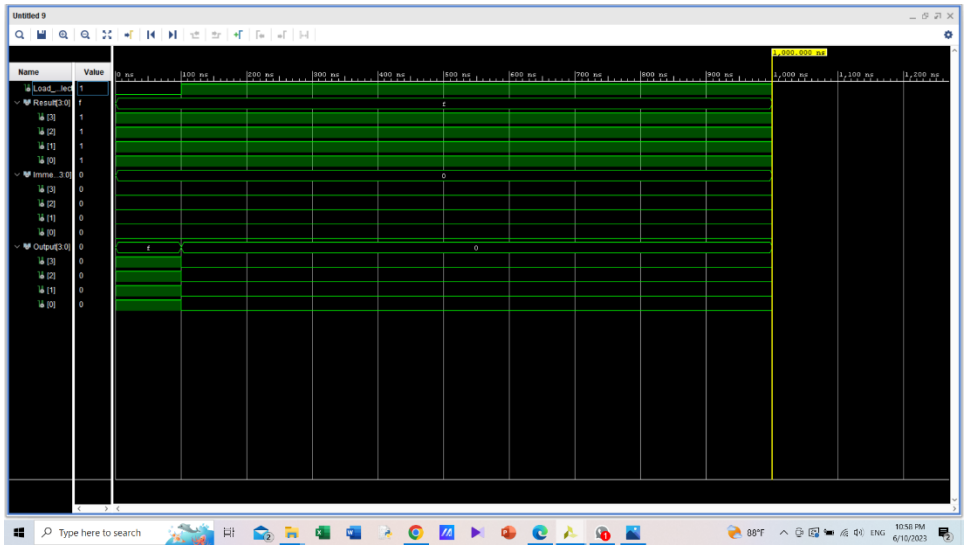
2 way 3-bit



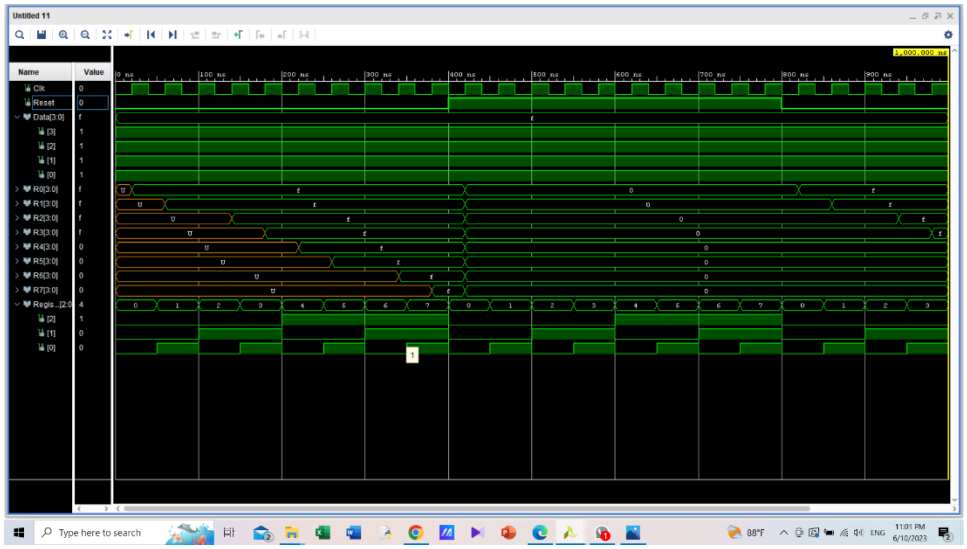
2 way 4-bit



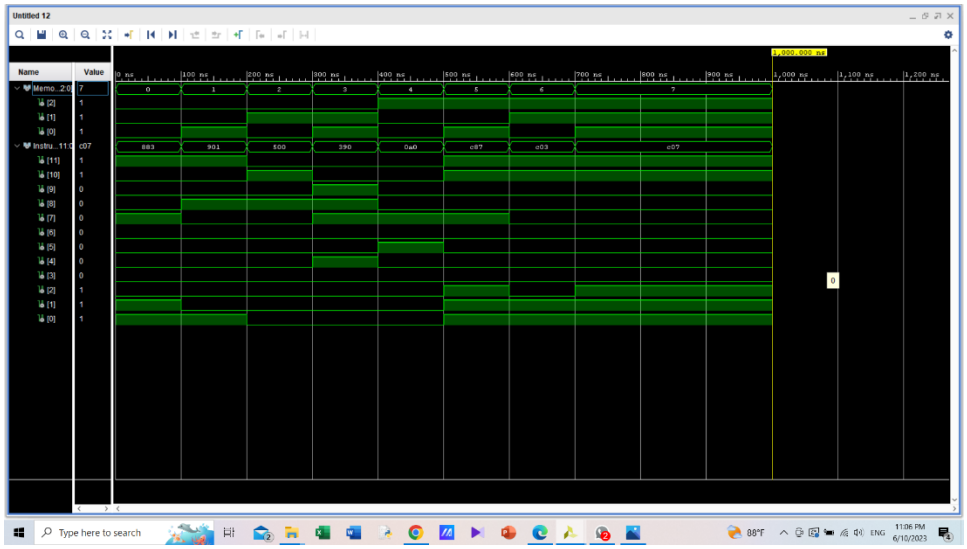
8 way 4-bit



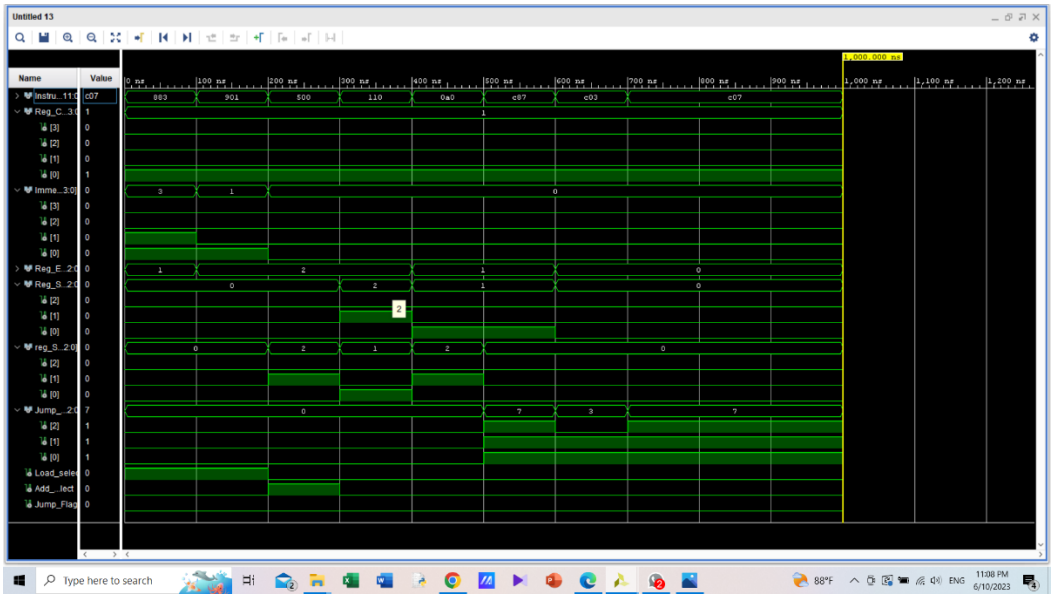
Register bank



Program ROM

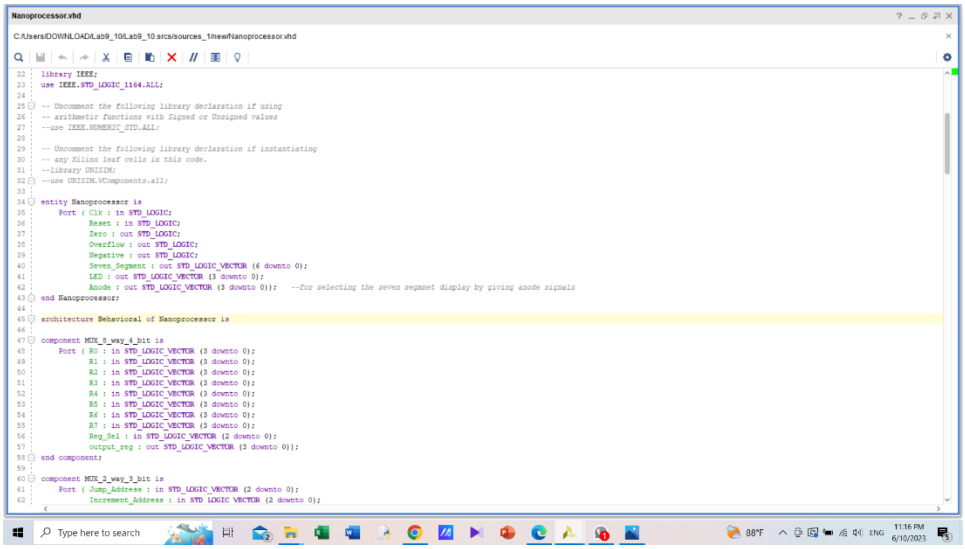


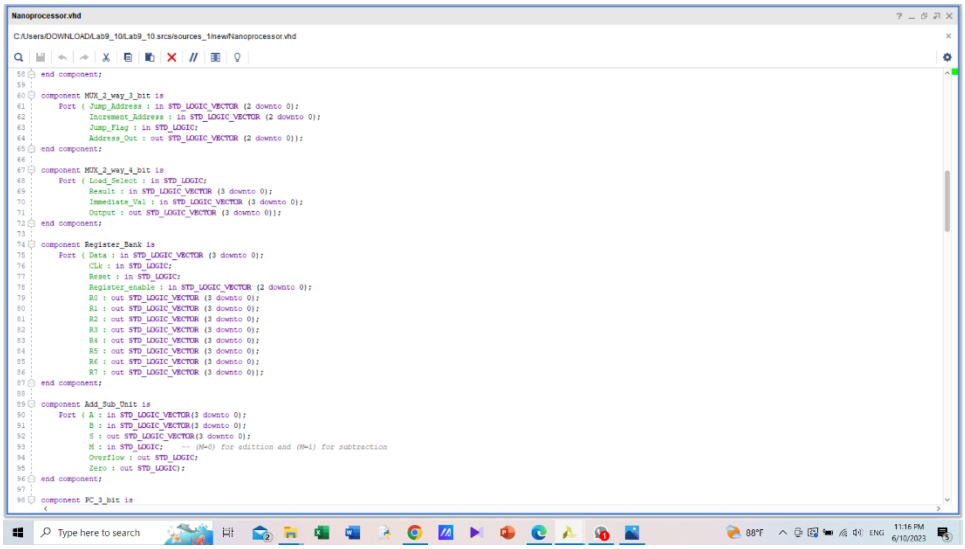
Instruction decoder



Nano – Processor

VHDL





Nanoprocessor.vhd

C:/Users/DOWNLOAD/Lab9_10/Lab9_10.srcs/sources_1/new/Nanoprocessor.vhd

```
component PC_3_bit is
    Port ( PC_input : in STD_LOGIC_VECTOR (2 downto 0);
           Clk : in STD_LOGIC;
           Reset : in STD_LOGIC;
           PC_output : out STD_LOGIC_VECTOR (2 downto 0));
end component;

component RCA_3_bit is
    Port ( A :in STD_LOGIC_VECTOR (2 downto 0);
           B : in STD_LOGIC_VECTOR (2 downto 0);
           S : out STD_LOGIC_VECTOR ( 2 downto 0));
end component;

component Instruction_Decoder is
    Port ( Instruction : in STD_LOGIC_VECTOR (11 downto 0);
           Reg_Check_Jump : in STD_LOGIC_VECTOR (3 downto 0);
           Reg_Enable : out STD_LOGIC_VECTOR (2 downto 0);
           Load_Select : out STD_LOGIC;
           Immediate_Val : out STD_LOGIC_VECTOR (3 downto 0);
           Reg_Select_A : out STD_LOGIC_VECTOR (2 downto 0);
           Reg_Select_B : out STD_LOGIC_VECTOR (2 downto 0);
           Add_Sub_Select : out STD_LOGIC;
           Jump_Flag : out STD_LOGIC;
           Jump_Address : out STD_LOGIC_VECTOR (2 downto 0));
end component;

component Seven_seg_display is
    Port ( R7_data : in STD_LOGIC_VECTOR (3 downto 0);
           seven_seg_code : out STD_LOGIC_VECTOR (6 downto 0);
           negative : out STD_LOGIC);
end component;

component Program_Rom is
    Port ( Memory_Select : in STD_LOGIC_VECTOR (2 downto 0);
           Instruction : out STD_LOGIC_VECTOR (11 downto 0));
end component;

component Slow_Clk is
    Port ( Clk_in : in STD_LOGIC;
           Clk_out : out STD_LOGIC);
```


Nanoprocessor.vhd

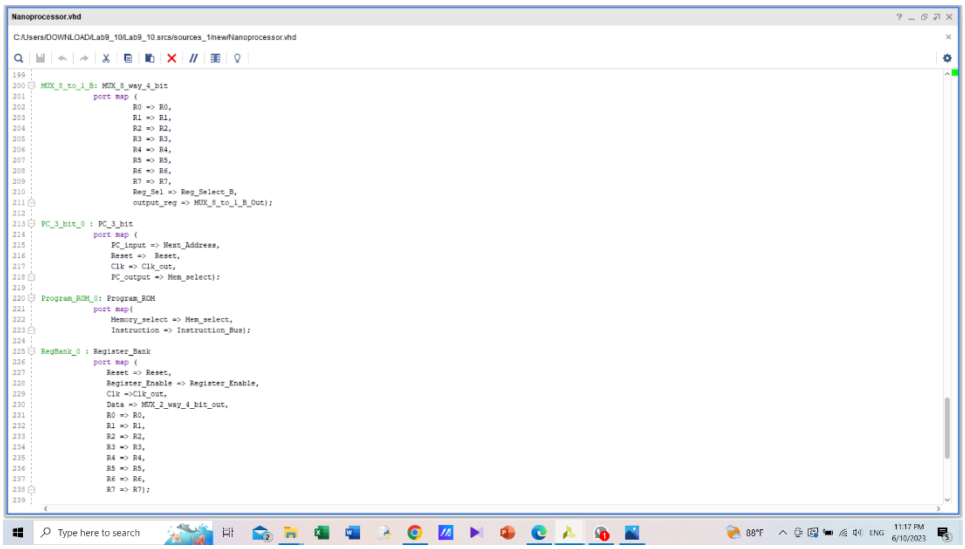
C:/Users/DOWNLOAD/Lab9_10/Lab9_10.srcs/sources_1/new/Nanoprocessor.vhd

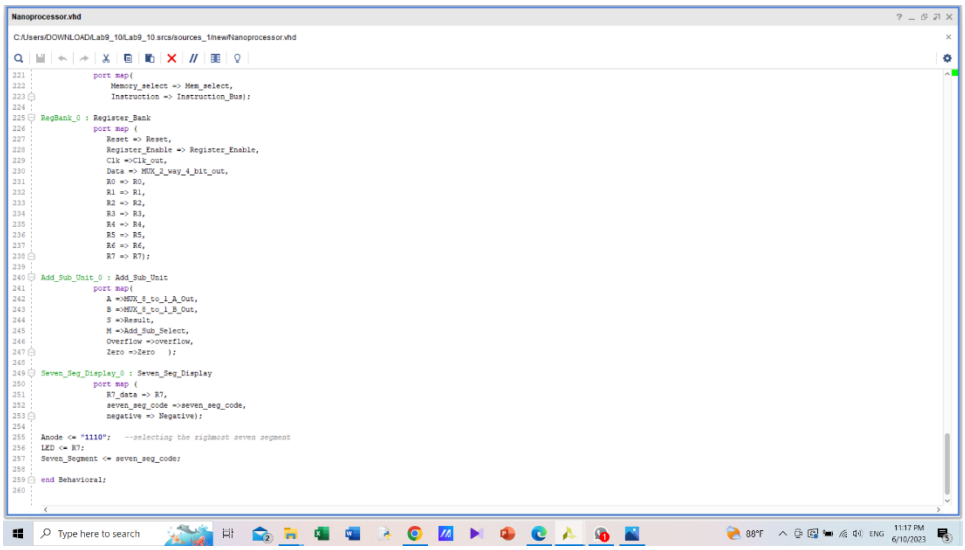
```
end component;

component Slow_Clk is
    Port ( Clk_in : in STD_LOGIC;
           Clk_out : out STD_LOGIC);
end component;

signal Instruction_Bus : std_logic_vector ( 11 downto 0);
signal Clk_Out, Jump_Flag, Load_Select, Add_Sub_Select: std_logic;
signal  Next_Address,Mem_select,Jump_Address,Increment_address,Register_Enable,Reg_Select_A,Reg_Select_B :std_logic_vector(2 downto 0);
signal MUX_8_to_1_A_Out,MUX_8_to_1_B_Out,Immediate_val,Result,MUX_2_way_4_bit_out:std_logic_vector(3 downto 0);
signal R0,R1,R2,R3,R4,R5,R6,R7 :std_logic_vector ( 3 downto 0);
signal seven_seg_code : std_logic_vector(6 downto 0);

begin

instruction_decoder_0 : instruction_decoder
    port map (
              Instruction => Instruction_Bus,
              Reg_Check_Jump =>MUX_8_to_1_A_Out,
              Reg_Enable => Register_Enable,
              Load_Select => Load_Select,
              Immediate_val => Immediate_val,
              Reg_Select_A => Reg_Select_A,
              Reg_Select_B => Reg_Select_B,
              Add_Sub_Select => Add_Sub_Select,
              Jump_Flag => Jump_Flag,
              Jump_Address => Jump_Address   );

Slow_Clk_0 : Slow_Clk
            port map(
                  Clk_in => Clk,
                  Clk_out => Clk_out);

RCA3Bit_0 : RCA_3_bit
            port map(
                  A => Mem_select,
                  B => "001",
                  S => Increment_Address );

MUX_2_way_3bit_0 : MUX_2_way_3_bit
```


Nanoprocessor.vhd

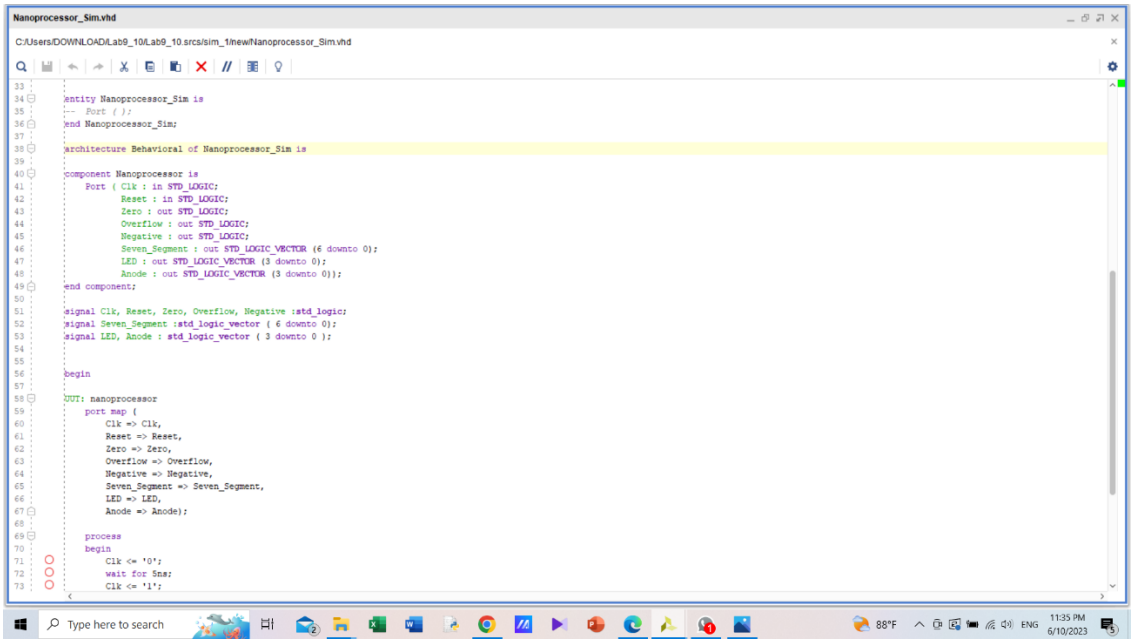
C:/Users/DOWNLOAD/Lab9_10/Lab9_10.srcs/sources_1/new/Nanoprocessor.vhd

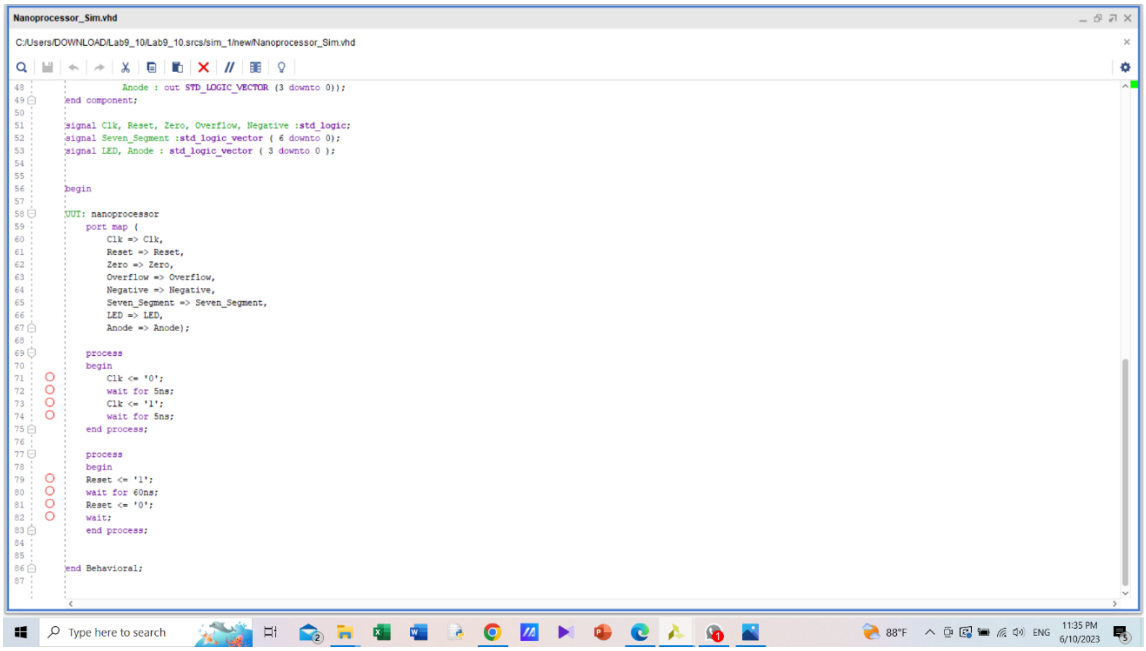
```
                  A => Mem_select,
                  B => "001",
                  S => Increment_Address );

MUX_2_way_3bit_0 : MUX_2_way_3_bit
            port map(
                  jump_Address => jump_Address,
                  Increment_Address => Increment_address ,
                  Jump_flag => jump_flag,
                  Address_Out => Next_Address   );

MUX_2_way_4bit_0 : MUX_2_way_4_bit
            port map (
                  Load_Select => Load_Select,
                  Result => Result,
                  Immediate_Val =>Immediate_val,
                  Output => MUX_2_way_4_bit_out   );

MUX_8_to_1_A: MUX_8_way_4_bit
            port map (
                  R0 => R0,
                  R1 => R1,
                  R2 => R2,
                  R3 => R3,
                  R4 => R4,
                  R5 => R5,
                  R6 => R6,
                  R7 => R7,
                  Reg_Sel => Reg_Select_A,
                  output_reg => MUX_8_to_1_A_Out);

MUX_8_to_1_B: MUX_8_way_4_bit
            port map (
                  R0 => R0,
                  R1 => R1,
                  R2 => R2,
                  R3 => R3,
                  R4 => R4,
                  R5 => R5,
                  R6 => R6,
                  R7 => R7,
```


Nanoprocessor.vhd
C:/Users/DOWNLOAD/Lab9_10/Lab9_10.srcs/sources_1/new/Nanoprocessor.vhd
MUX_8_to_1_B: MUX_8_way_4_bit
port map (
R0 => R0,
R1 => R1,
R2 => R2,
R3 => R3,
R4 => R4,
R5 => R5,
R6 => R6,
R7 => R7,
Reg_Sel => Reg_Select_B,
output_reg => MUX_8_to_1_B_Out);
PC_3_bit_0 : PC_3_bit
port map (
PC_input => Next_Address,
Reset => Reset,
Clk => Clk_out,
PC_output => Mem_select);
Program_ROM_0: Program_ROM
port map(
Memory_select => Mem_select,
Instruction => Instruction_Bus);
RegBank_0 : Register_Bank
port map (
Reset => Reset,
Register_Enable => Register_Enable,
Clk =>Clk_out,
Data => MUX_2_way_4_bit_out,
R0 => R0,
R1 => R1,
R2 => R2,
R3 => R3,
R4 => R4,
R5 => R5,
R6 => R6,
R7 => R7);
Type here to search
88°F
ENG
11:17 PM
6/10/2023

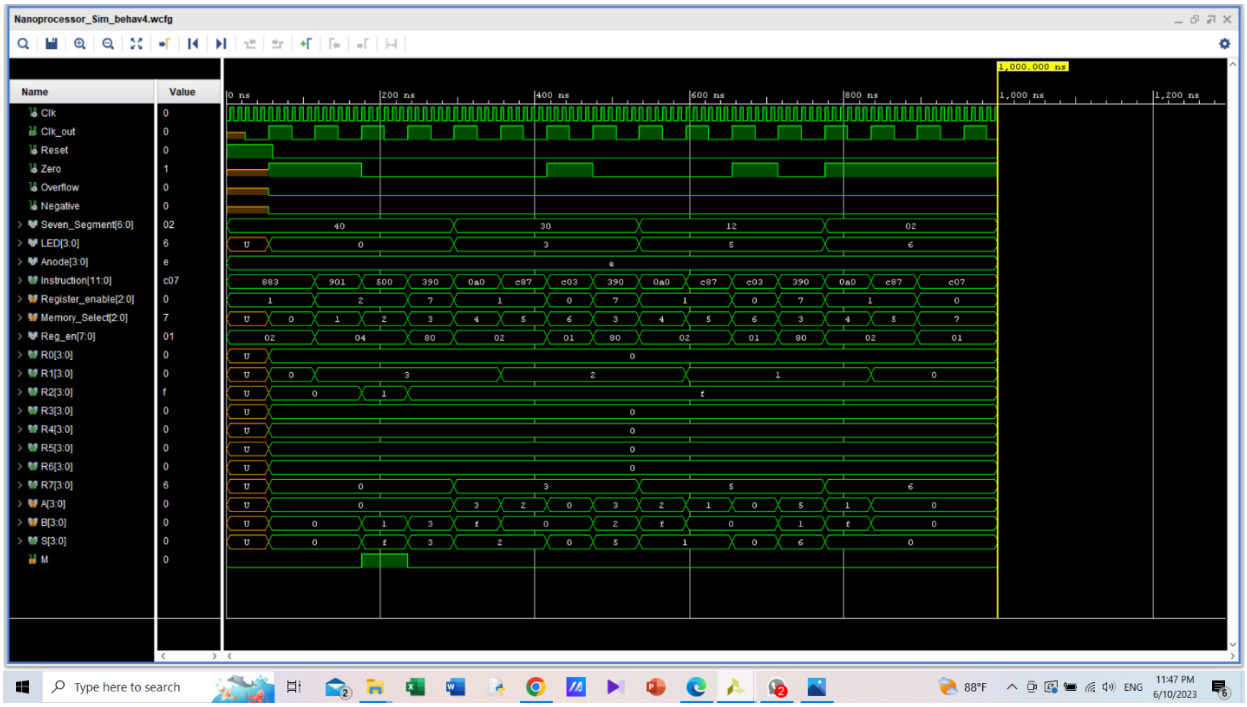

Nanoprocessor.vhd
C:/Users/DOWNLOAD/Lab9_10/Lab9_10.srcs/sources_1/new/Nanoprocessor.vhd
port map(
Memory_select => Mem_select,
Instruction => Instruction_Bus);
RegBank_0 : Register_Bank
port map (
Reset => Reset,
Register_Enable => Register_Enable,
Clk =>Clk_out,
Data => MUX_2_way_4_bit_out,
R0 => R0,
R1 => R1,
R2 => R2,
R3 => R3,
R4 => R4,
R5 => R5,
R6 => R6,
R7 => R7);
Add_Sub_Unit_0 : Add_Sub_Unit
port map(
A =>MUX_8_to_1_A_Out,
B =>MUX_8_to_1_B_Out,
S =>Result,
M =>Add_Sub_Select,
Overflow =>overflow,
Zero =>Zero);
Seven_Seg_Display_0 : Seven_Seg_Display
port map (
R7_data => R7,
seven_seg_code =>seven_seg_code,
negative => Negative);
Anode <= "1110"; --selecting the rightmost seven segment
LED <= R7;
Seven_Segment <= seven_seg_code;
end Behavioral;
Type here to search
88°F
ENG
11:17 PM
6/10/2023

Nano-processor test bench

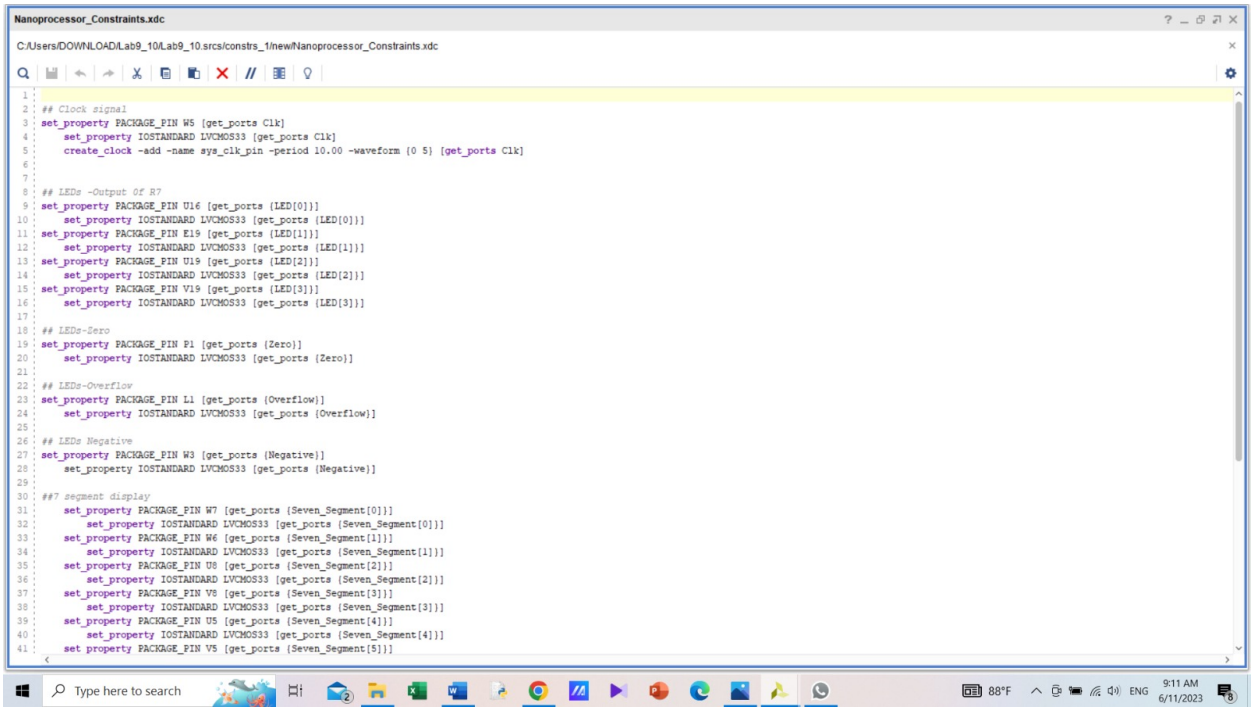


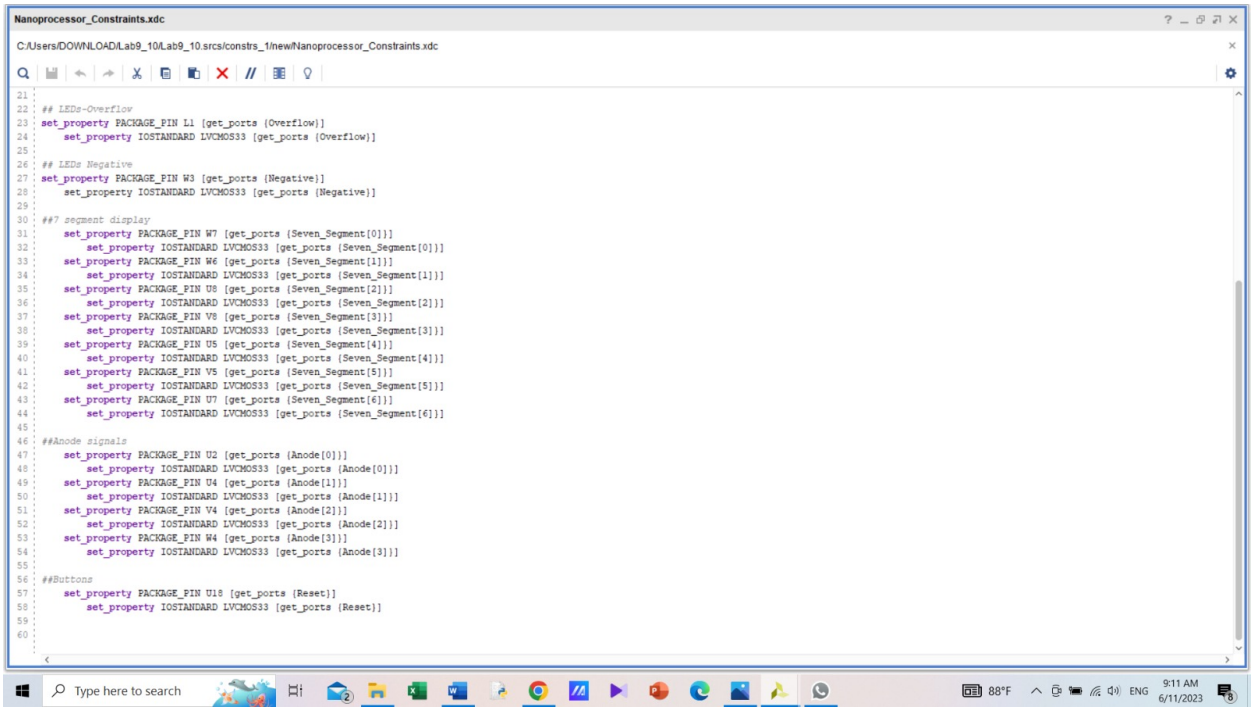


Timing Diagram



Constraints :-





Conclusions :-

- We can simulate the nano-processor before implementing it on the hardware to check for logical errors in our codes. So, we don't have to use hardware components for testing purposes.
- We can approach a bigger problem by dividing it into smaller sub problems.
- There can be differences between our simulation and how the actual circuit behaves in hardware.
- The slow clock must be adjusted to a suitable value for visualizing the output in a realistic way.
- This simulation process allows for optimization and improvement of the design, leading to a more efficient and reliable nano-processor.
- Working as a team will not only allow for the sharing of responsibilities and workload but also foster effective communication and coordination among team members.
- Each team member can bring their unique skills and perspectives to the project, leading to a more comprehensive and well-rounded nano-processor design.

Contributions :-

Kulasekara K M S N

Designed :-

- Instruction decoder.
- Program ROM
- 7 segment display
- 3-bit adder
- Program counter 20 hours

Kulathunga K A J T

Designed :-

- Combination of the components.
- Tri state buffers and Multiplexers.
- Add-Sub Unit
- Register bank 18 hours