Lab 9-10 – Nanoprocessor Design Competition
CS1050 Computer Organization and Digital Design

• Student names and index numbers.

N. Harikishna – 210206B
J. Harismenan - 210207E

• State the assigned lab task in a few sentences

The assigned lab task involves designing and building an Instruction Decoder for a 4-bit nanoprocessor using the instructions provided in Table 1. The decoder should have output pins for each instruction, and the design should be tested using simulation. An Assembly program must also be written to calculate the total of all integers between 1 and 3, with the final answer stored in Register R7. The resulting code should be hard-coded into ROM, and inputs and outputs connected to the circuit for testing on a BASYS 3 board. The team will need to demonstrate the functionality of the nanoprocessor to the instructor and explain each member's contribution.

• Assembly program and its machine code representation

```
"100010001010",  -- MOVI R1,10
"100100000001",  -- MOVI R2,1
"010100000000",  -- NEG R2
"000010100000",  -- ADD R1,R2
"110010000111",  -- JZR R1,7
"110000000011",  -- JZR R0,3
"101110001001",  -- MOVI R7,9
"000100110000"   -- ADD R2,R3
```

• All VHDL codes

Programe 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 Program_counter is
   Port ( Clk_in : in STD_LOGIC;
          Res : in STD_LOGIC;
          D : in STD_LOGIC_VECTOR (2 downto 0);
          Memory : out STD_LOGIC_VECTOR (2 downto 0));
end Program_counter;

architecture Behavioral of Program_counter is
component D_FF
   Port ( D : in STD_LOGIC;
     Res : in STD_LOGIC;
     EN : in STD_LOGIC:='1';
     Clk : in STD_LOGIC;
     Q : out STD_LOGIC);
end component;
begin
D_FF_0: D_FF
   port map(
      D => D(0),
      Res => Res,
      Clk => Clk_in,
      Q => Memory(0)
   );

D_FF_1: D_FF
      port map(
         D => D(1),
         Res => Res,
         Clk => Clk_in,
         Q => Memory(1)
      );

D_FF_2: D_FF
   port map(
      D => D(2),
      Res => Res,
      Clk => Clk_in,
      Q => Memory(2)
   );

end Behavioral;
```

Programe counter_TB

```
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 TB_Program_counter is
--  Port ( );
end TB_Program_counter;

architecture Behavioral of TB_Program_counter is
component Program_counter
   Port ( Clk_in : in STD_LOGIC;
      Res : in STD_LOGIC;
      D : in STD_LOGIC_VECTOR (2 downto 0);
      Memory : out STD_LOGIC_VECTOR (2 downto 0));
end component;
signal Clk_in : std_logic:='0';
signal Res : std_logic;
signal D, Memory : std_logic_vector (2 downto 0);
begin
UUT: Program_counter
   port map(
      Clk_in => Clk_in,
      Res => Res,
      D => D,
      Memory => Memory
   );

process begin
     wait for 50 ns;
     Clk_in <= not (Clk_in);
end process;

process begin
   --Index No:- 210206 => 110 011 010 100 011 110
   --Index No:- 210207 => 110 011 010 100 011 111
   wait for 100 ns;
   Res <= '1';
   D <= "110";

   wait for 100 ns;
   Res <= '0';
   D <= "111";

   wait for 200 ns;
   D <= "011";

   wait for 200 ns;
   Res <= '1';
   D <= "100";

   wait for 200 ns;
   Res <= '0';
   D <= "010";

   wait;
end process;
```

end Behavioral;

Programmable ROM

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.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 P_ROM is
   Port ( Address : in STD_LOGIC_VECTOR (2 downto 0);
          Instruction : out STD_LOGIC_VECTOR (11 downto 0));
end P_ROM;

architecture Behavioral of P_ROM is
   type rom_type is array (0 to 7) of std_logic_vector(11 downto 0);
   signal Prog_Rom: rom_type := (
         "100010001010",  -- MOVI R1,10
         "100100000001",  -- MOVI R2,1
         "010100000000",  -- NEG R2
         "000010100000",  -- ADD R1,R2
         "110010000111",  -- JZR R1,7
         "110000000011",  -- JZR R0,3
         "101110001001",  -- MOVI R7,9
         "000100110000"   -- ADD R2,R3

        -- Counting 1 to f
        --   "100010000001",  -- MOVI R1,1
        --   "001110010000", -- ADD R7,R1
        --   "110000000001", -- JZR R0,1
        --   "001110010000", -- ADD R7,R2  --Below steps are dummy
        --   "001110010000", -- ADD R7,R2
        --   "001110010000", -- ADD R7,R2
        --   "001110010000", -- ADD R7,R2
        --   "110000000111"  -- JZR R0,7

        -- adding 4,3,2,1.....
        --   "100010000110",  -- MOVI R1,5
        --   "100100000001",  -- MOVI R2,1
        --   "010100000000",  -- NEG R2
        --   "000010100000", -- ADD R1,R2
        --   "001110010000", -- ADD R7,R1
        --   "110000000011", -- JZR R0,3   --Below steps are dummy
        --   "001110010000", -- ADD R7,R2
        --   "001110010000" -- ADD R7,R2

        --backward counting
        --"101110001111",  -- MOVI R7,F
        --"100100000001",  -- MOVI R2,1
        --"010100000000",  -- NEG R2
        --"001110100000", -- ADD R7,R2
        --"111110000111", -- JZR R7,7
        --"110000000011", -- JZR R0,3   -- dummy
        --"001110010000", -- ADD R7,R2  -- dummy
        --"111110000111"  -- JZR R7,7
          );

begin
  Instruction <= Prog_Rom(to_integer(unsigned(Address)));
end Behavioral;
```

Programmable ROM _TB

```
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 TB_Program_ROM is
--  Port ( );
end TB_Program_ROM;

architecture Behavioral of TB_Program_ROM is
component P_ROM
    Port ( Address : in STD_LOGIC_VECTOR (2 downto 0);
           Instruction : out STD_LOGIC_VECTOR (11 downto 0));
end component;
signal Address : STD_LOGIC_VECTOR (2 downto 0);
signal Instruction : STD_LOGIC_VECTOR (11 downto 0);
begin
UUT: P_ROM
    Port map ( Address => Address,
               Instruction => Instruction);
process
begin
    --Index No:- 210206 => 110 011 010 100 011 110
    --Index No:- 210207 => 110 011 010 100 011 111
    Address <= "110";
    wait for 200 ns;

    Address <= "111";
    wait for 200 ns;

    Address <= "100";
    wait for 200 ns;

    Address <= "010";
    wait for 200 ns;

    Address <= "011";
    wait for 200 ns;
end process;
end Behavioral;
```

Instruction Decoder

```
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 Ins_Decoder is
   Port ( Ins : in STD_LOGIC_VECTOR (11 downto 0);
     Reg_Jum : in STD_LOGIC_VECTOR (3 downto 0);
     Reg_EN : out STD_LOGIC_VECTOR (2 downto 0);
     Load_Sel : out STD_LOGIC;
     Input_Value : out STD_LOGIC_VECTOR (3 downto 0);
     Reg_Sel_0 : out STD_LOGIC_VECTOR (2 downto 0);
     Reg_Sel_1 : out STD_LOGIC_VECTOR (2 downto 0);
     ADD_Sub_Sel : out STD_LOGIC;
     JMP_Flag : out STD_LOGIC;
     Jmp_Value : out STD_LOGIC_VECTOR (2 downto 0));
end Ins_Decoder;

architecture Behavioral of Ins_Decoder is
component Decoder_2_to_4
   Port ( I : in STD_LOGIC_VECTOR (1 downto 0);
     EN : in STD_LOGIC;
     Y : OUT STD_LOGIC_VECTOR (3 downto 0));
end component;
signal ADD,NEG,MOV,JZR: std_logic;
begin
Decoder_2_4_0: Decoder_2_to_4
   port map(
      I => Ins(11 downto 10),
      EN => '1',
      Y(0) => ADD,
      Y(1) => NEG,
      Y(2) => MOV,
      Y(3) => JZR
   );

Reg_EN <= Ins(9 downto 7);

Reg_Sel_0 <= Ins(9 downto 7);
Reg_Sel_1 <= Ins(6 downto 4);

ADD_Sub_Sel <= NEG;

Load_Sel <= MOV;
Input_Value <= Ins(3 downto 0);

JMP_Flag <= JZR and not(Reg_Jum(3) or Reg_Jum(2) or Reg_Jum(1) or Reg_Jum(0));
Jmp_Value <= Ins(2 downto 0);


end Behavioral;
```

Instruction Decoder_TB

```
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 TB_Ins_Decoder is
--  Port ( );
end TB_Ins_Decoder;

architecture Behavioral of TB_Ins_Decoder is
component Ins_Decoder
      Port ( Ins : in STD_LOGIC_VECTOR (11 downto 0);
            Reg_Jum : in STD_LOGIC_VECTOR (3 downto 0);
            Reg_EN : out STD_LOGIC_VECTOR (2 downto 0);
            Load_Sel : out STD_LOGIC;
            Input_Value : out STD_LOGIC_VECTOR (3 downto 0);
            Reg_Sel_0 : out STD_LOGIC_VECTOR (2 downto 0);
            Reg_Sel_1 : out STD_LOGIC_VECTOR (2 downto 0);
            ADD_Sub_Sel : out STD_LOGIC;
            JMP_Flag : out STD_LOGIC;
            Jmp_Value : out STD_LOGIC_VECTOR (2 downto 0));
end component;
signal Instruction : STD_LOGIC_VECTOR (11 downto 0);
signal Reg_to_JUMP, Imm_Value : STD_LOGIC_VECTOR (3 downto 0);
signal Reg_EN, J_V, Reg_M0, Reg_M1 : STD_LOGIC_VECTOR (2 downto 0);
signal Load_Selector, ALU_EN, J_F : STD_LOGIC ;
begin
UUT: Ins_Decoder
      Port map ( Ins => Instruction,
            Reg_Jum => Reg_to_JUMP,
            Reg_EN => Reg_EN,
            Load_Sel => Load_Selector,
            Input_Value => Imm_Value,
            Reg_Sel_0 => Reg_M0,
            Reg_Sel_1 => Reg_M1,
            ADD_Sub_Sel => ALU_EN,
            JMP_Flag => J_F,
            Jmp_Value => J_V);
process
begin
   --Index No:- 210206 => 11 0011 0101 0001 1110
   --Index No:- 210207 => 11 0011 0101 0001 1111
   Instruction <= "100100001110"; -- MOV R2,14
   wait for 100 ns;

   Instruction <= "010100011110"; -- NEG R2
   wait for 125 ns;

   Instruction <= "101100000001"; -- MOV R6,1
   wait for 100 ns;

   Instruction <= "101010001010"; -- MOV R5,10
   wait for 100 ns;

   Instruction <= "001101010000"; -- ADD R6,R5
   wait for 125 ns;
```

```
    Instruction <= "1100100000100"; -- JZR R1,4
    wait for 125 ns;

    Instruction <= "1110100001001"; -- JZR R5,9
    wait for 100 ns;

    Reg_to_JUMP <= "1110";
    wait for 125 ns;

    Reg_to_JUMP <= "0000";
    wait for 100 ns;


end process;


end Behavioral;
```

Register

```
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 Reg is
   Port ( D_in: in STD_LOGIC_VECTOR (3 downto 0);
          Enable : in STD_LOGIC;
          Res : in STD_LOGIC;
          Clk : in STD_LOGIC;
          D_out : out STD_LOGIC_VECTOR (3 downto 0));
end Reg;

architecture Behavioral of Reg is
component D_FF
   Port ( D : in STD_LOGIC;
     Res : in STD_LOGIC;
     EN : in STD_LOGIC;
     Clk : in STD_LOGIC;
     Q : out STD_LOGIC);
end component;
begin
D_FF_0: D_FF
   port map(
      D =>D_in(0),
      Res =>Res,
      EN =>Enable,
      Clk =>Clk,
      Q => D_out(0)
   );

D_FF_1: D_FF
   port map(
      D =>D_in(1),
      Res =>Res,
      EN =>Enable,
      Clk =>Clk,
      Q => D_out(1)
   );

D_FF_2: D_FF
   port map(
      D =>D_in(2),
      Res =>Res,
      EN =>Enable,
      Clk =>Clk,
      Q => D_out(2)
   );

D_FF_3: D_FF
   port map(
      D =>D_in(3),
      Res =>Res,
      EN =>Enable,
      Clk =>Clk,
      Q => D_out(3)
   );
```

end Behavioral;

Register_TB

```
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 TB_Register is
--  Port ( );
end TB_Register;

architecture Behavioral of TB_Register is
component Reg
   Port ( D_in: in STD_LOGIC_VECTOR (3 downto 0);
      Enable : in STD_LOGIC;
      Res : in STD_LOGIC;
      Clk : in STD_LOGIC;
      D_out : out STD_LOGIC_VECTOR (3 downto 0));
end component;
signal Res : STD_LOGIC;
signal D_in, D_out : STD_LOGIC_VECTOR (3 downto 0);
signal Clk, En: std_logic:= '0';
begin
UUT: Reg
   Port map(
        D_in => D_in,
        Enable => En,
        Res => Res,
        Clk => Clk,
        D_out => D_out
   );
process begin
     wait for 50 ns;
     Clk <= not(Clk);
end process;

process
begin
    En <= '1';
    Res <= '1';
    wait for 60 ns;

    --Index No:- 210206 => 11 0011 0101 0001 1110
    --Index No:- 210207 => 11 0011 0101 0001 1111
    Res <= '0';
    D_in <= "1110";
    wait for 100 ns;

    D_in <= "0001";
    wait for 100 ns;

    En <= '0';
    D_in <= "0101";
    wait for 100 ns;

    Res <= '1';
    wait for 100 ns;
```

```
    En <= '1';
    Res <= '0';
    D_in <= "0011";
    wait for 100 ns;

    D_in <= "1111";
    wait for 100 ns;
end process;
end Behavioral;
```

Register Bank

```
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 Register_Bank is
   Port ( Da_In : in STD_LOGIC_VECTOR (3 downto 0);
          Clk : in STD_LOGIC;
          S_Reg : in STD_LOGIC_VECTOR (2 downto 0);
          Res : in STD_LOGIC;
          R0_out : out STD_LOGIC_VECTOR (3 downto 0);
          R1_out : out STD_LOGIC_VECTOR (3 downto 0);
          R2_out : out STD_LOGIC_VECTOR (3 downto 0);
          R3_out : out STD_LOGIC_VECTOR (3 downto 0);
          R4_out : out STD_LOGIC_VECTOR (3 downto 0);
          R5_out : out STD_LOGIC_VECTOR (3 downto 0);
          R6_out : out STD_LOGIC_VECTOR (3 downto 0);
          R7_out : out STD_LOGIC_VECTOR (3 downto 0));
end Register_Bank;

architecture Behavioral of Register_Bank is
component Reg
   Port ( D_in: in STD_LOGIC_VECTOR (3 downto 0);
          Enable : in STD_LOGIC;
          Res : in STD_LOGIC;
          Clk : in STD_LOGIC;
          D_out : out STD_LOGIC_VECTOR (3 downto 0));
end component;
component Decoder_3_to_8
   Port ( I : in STD_LOGIC_VECTOR (2 downto 0);
          EN : in STD_LOGIC:='1';
          Y : out STD_LOGIC_VECTOR (7 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 => S_Reg,
      Y => Reg_En
      );

Reg_0: Reg
    port map(
          D_in => "0000",
          Enable => Reg_En(0),
          Clk => Clk,
          Res => Res,
          D_out => R0_out
          );
Reg_1: Reg
    port map(
          D_in => Da_in,
          Enable => Reg_En(1),
          Clk => Clk,
          Res => Res,
          D_out => R1_out
```

```
        );
Reg_2: Reg
    port map(
        D_in => Da_in,
        Enable => Reg_En(2),
        Clk => Clk,
        Res => Res,
        D_out => R2_out
        );
Reg_3: Reg
    port map(
        D_in => Da_in,
        Enable => Reg_En(3),
        Clk => Clk,
        Res => Res,
        D_out => R3_out
        );
Reg_4: Reg
    port map(
        D_in => Da_in,
        Enable => Reg_En(4),
        Clk => Clk,
        Res => Res,
        D_out => R4_out
        );
Reg_5: Reg
    port map(
        D_in => Da_in,
        Enable => Reg_En(5),
        Clk => Clk,
        Res => Res,
        D_out => R5_out
        );
Reg_6: Reg
    port map(
        D_in => Da_in,
        Enable => Reg_En(6),
        Clk => Clk,
        Res => Res,
        D_out => R6_out
        );
Reg_7: Reg
    port map(
        D_in => Da_in,
        Enable => Reg_En(7),
        Clk => Clk,
        Res => Res,
        D_out => R7_out
        );


end Behavioral;
```

Register Bank_TB

```
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 TB_Register_Bank is
--  Port ( );
end TB_Register_Bank;

architecture Behavioral of TB_Register_Bank is
component Register_Bank
   Port ( Da_In : in STD_LOGIC_VECTOR (3 downto 0);
          Clk : in STD_LOGIC;
          S_Reg : in STD_LOGIC_VECTOR (2 downto 0);
          Res : in STD_LOGIC;
          R0_out : out STD_LOGIC_VECTOR (3 downto 0);
          R1_out : out STD_LOGIC_VECTOR (3 downto 0);
          R2_out : out STD_LOGIC_VECTOR (3 downto 0);
          R3_out : out STD_LOGIC_VECTOR (3 downto 0);
          R4_out : out STD_LOGIC_VECTOR (3 downto 0);
          R5_out : out STD_LOGIC_VECTOR (3 downto 0);
          R6_out : out STD_LOGIC_VECTOR (3 downto 0);
          R7_out : out STD_LOGIC_VECTOR (3 downto 0));
end component;
signal D, R1, R2, R3, R4, R5, R6, R7 : STD_LOGIC_VECTOR (3 downto 0);
signal R0 : STD_LOGIC_VECTOR (3 downto 0) :="0000";
signal Clk : STD_LOGIC:= '0';
signal Res : STD_LOGIC;
signal Sel : STD_LOGIC_VECTOR (2 downto 0);
begin
UUT: Register_Bank
  Port map(
     Da_in => D,
     Clk => Clk,
     S_Reg => Sel,
     Res => Res,
     R0_out => R0,
     R1_out => R1,
     R2_out => R2,
     R3_out => R3,
     R4_out => R4,
     R5_out => R5,
     R6_out => R6,
     R7_out => R7
  );
process begin
       wait for 50 ns;
       Clk <= not(Clk);
end process;

process
begin
      --Index No:- 210206 => 11 0011 0101 0001 1110 (For Data_in)
      --Index No:- 210207 => 110 011 010 100 011 111 (For Register Selection)
      D <= "1110";
      Sel <= "111";
      wait for 100 ns;
```

```
        D <= "0001";
        Sel <= "011";
        wait for 100 ns;

        D <= "0101";
        Sel <= "100";
        wait for 100 ns;

        Res <= '1';
        wait for 100 ns;

        Res <= '0';
        D <= "0011";
        Sel <= "010";
        wait for 100 ns;

        D <= "1100";
        Sel <= "110";
        wait for 100 ns;

    end process;
    end Behavioral;
```

Mux_2_Way_4_Bit

```
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_2_Way_4_Bit is
    Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
           B : in STD_LOGIC_VECTOR (3 downto 0);
           JF : in STD_LOGIC;
           Q : out STD_LOGIC_VECTOR (3 downto 0));
end Mux_2_Way_4_Bit;

architecture Behavioral of Mux_2_Way_4_Bit is
component Mux_2_to_1
  Port ( A : in STD_LOGIC;
         B : in STD_LOGIC;
         Sel : in STD_LOGIC;
         Y : out STD_LOGIC);
end component;
begin
Mux_0: Mux_2_to_1
    Port map(
         A => A(0),
         B => B(0),
         Sel => JF,
         Y => Q(0));

Mux_1: Mux_2_to_1
    Port map(
         A => A(1),
         B => B(1),
         Sel => JF,
         Y => Q(1));

Mux_2: Mux_2_to_1
    Port map(
         A => A(2),
         B => B(2),
         Sel => JF,
         Y => Q(2));

Mux_3: Mux_2_to_1
    Port map(
         A => A(3),
         B => B(3),
         Sel => JF,
         Y => Q(3));

end Behavioral;
```

Mux_2_Way_4_Bit_TB

```
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 TB_Mux_2_4 is
--  Port ( );
end TB_Mux_2_4;

architecture Behavioral of TB_Mux_2_4 is
component Mux_2_Way_4_Bit
   Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
      B : in STD_LOGIC_VECTOR (3 downto 0);
      JF : in STD_LOGIC;
      Q : out STD_LOGIC_VECTOR (3 downto 0));
end component;
signal A,B,Q : STD_LOGIC_VECTOR (3 downto 0);
signal Jump : STD_LOGIC;
begin
UUT: Mux_2_Way_4_Bit
   port map (
      A => A,
      B => B,
      JF => Jump,
      Q => Q );
process
begin
    --Index No:- 210206 => 11 0011 0101 0001 1110
    --Index No:- 210207 => 11 0011 0101 0001 1111
    A <= "1110";
    B <= "1111";
    Jump <= '1';

    wait for 250 ns;
    A <= "0001";
    B <= "0101";
    Jump <= '0';

    wait for 250 ns;
    A <= "0011";
    B <= "1110";
    Jump <= '1';

    wait for 250 ns;
    A <= "0101";
    B <= "0001";
    Jump <= '0';

    wait;
end process;
end Behavioral;
```

Mux_2_Way_3_Bit

```
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_2_Way_3_Bit is
    Port ( A : in STD_LOGIC_VECTOR (2 downto 0);
           B : in STD_LOGIC_VECTOR (2 downto 0);
           JF : in STD_LOGIC;
           Q : out STD_LOGIC_VECTOR (2 downto 0));
end Mux_2_Way_3_Bit;

architecture Behavioral of Mux_2_Way_3_Bit is
component Mux_2_to_1
  Port ( A : in STD_LOGIC;
         B : in STD_LOGIC;
         Sel : in STD_LOGIC;
         Y : out STD_LOGIC);
end component;
begin
Mux_0: Mux_2_to_1
    Port map(
         A => A(0),
         B => B(0),
         Sel => JF,
         Y => Q(0));

Mux_1: Mux_2_to_1
    Port map(
         A => A(1),
         B => B(1),
         Sel => JF,
         Y => Q(1));

Mux_2: Mux_2_to_1
    Port map(
         A => A(2),
         B => B(2),
         Sel => JF,
         Y => Q(2));


end Behavioral;
```

Mux_2_Way_3_Bit_TB

```
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 TB_Mux_2_3 is
--  Port ( );
end TB_Mux_2_3;

architecture Behavioral of TB_Mux_2_3 is
component Mux_2_Way_3_Bit
   Port ( A : in STD_LOGIC_VECTOR (2 downto 0);
      B : in STD_LOGIC_VECTOR (2 downto 0);
      JF : in STD_LOGIC;
      Q : out STD_LOGIC_VECTOR (2 downto 0));
end component;
signal A,B,Q : STD_LOGIC_VECTOR (2 downto 0);
signal Jump : STD_LOGIC;
begin
UUT: Mux_2_Way_3_Bit
   port map (
      A => A,
      B => B,
      JF => Jump,
      Q => Q );
process
begin
   --Index No:- 210206 => 110 011 010 100 011 110
   --Index No:- 210207 => 110 011 010 100 011 111
   A <= "110";
   B <= "111";
   Jump <= '1';

   wait for 250 ns;
   A <= "011";
   B <= "100";
   Jump <= '0';

   wait for 250 ns;
   A <= "010";
   B <= "011";
   Jump <= '1';

   wait for 250 ns;
   A <= "100";
   B <= "110";
   Jump <= '0';

   wait;

end process;
end Behavioral;
```

Mux_8_Way_4_Bit

```
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 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);
          Sel_R : in STD_LOGIC_VECTOR (2 downto 0);
          Q : out STD_LOGIC_VECTOR (3 downto 0));
end Mux_8_Way_4_Bit;

architecture Behavioral of Mux_8_Way_4_Bit is
component Mux_8_to_1
     Port ( D : in STD_LOGIC_VECTOR (7 downto 0);
            S : in STD_LOGIC_VECTOR (2 downto 0);
            EN : in STD_LOGIC:='1';
            Y : out STD_LOGIC);
end component;
begin
Mux_8_1_0 : Mux_8_to_1
    port map(
      D(0) => R0(0), D(1) => R1(0), D(2) => R2(0), D(3) => R3(0),
      D(4) => R4(0), D(5) => R5(0), D(6) => R6(0), D(7) => R7(0),
      S => Sel_R,
      Y => Q(0)
      );

Mux_8_1_1 : Mux_8_to_1
    port map(
      D(0) => R0(1), D(1) => R1(1), D(2) => R2(1), D(3) => R3(1),
      D(4) => R4(1), D(5) => R5(1), D(6) => R6(1), D(7) => R7(1),
      S => Sel_R,
      Y => Q(1)
       );


Mux_8_1_2 : Mux_8_to_1
    port map(
      D(0) => R0(2), D(1) => R1(2), D(2) => R2(2), D(3) => R3(2),
      D(4) => R4(2), D(5) => R5(2), D(6) => R6(2), D(7) => R7(2),
      S => Sel_R,
      Y => Q(2)
      );

Mux_8_1_3 : Mux_8_to_1
    port map(
      D(0) => R0(3), D(1) => R1(3), D(2) => R2(3), D(3) => R3(3),
      D(4) => R4(3), D(5) => R5(3), D(6) => R6(3), D(7) => R7(3),
      S => Sel_R,
```

```
    Y => Q(3)
    );
end Behavioral;
```

Mux_8_Way_4_Bit_TB

```
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 TB_Mux_8_4 is
--  Port ( );
end TB_Mux_8_4;

architecture Behavioral of TB_Mux_8_4 is
component Mux_8_Way_4_Bit
      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);
           Sel_R : in STD_LOGIC_VECTOR (2 downto 0);
           Q : out STD_LOGIC_VECTOR (3 downto 0));
end component;
signal R0,R1,R2,R3,R4,R5,R6,R7,Q : STD_LOGIC_VECTOR (3 downto 0);
signal  Selector : STD_LOGIC_VECTOR (2 downto 0);
begin
UUT: Mux_8_Way_4_Bit
  Port map(
     R0 => R0,
     R1 => R1,
     R2 => R2,
     R3 => R3,
     R4 => R4,
     R5 => R5,
     R6 => R6,
     R7 => R7,
     Sel_R => Selector,
     Q => Q
     );
process
begin
    --Index No:- 210206 => 11 0011 0101 0001 1110
    --Index No:- 210207 => 1100 1101 0100 0111 11
    R0 <= "0001";
    R1 <= "1110";
    R2 <= "0111";
    R3 <= "0101";
    R4 <= "0011";
    R5 <= "1100";
    R6 <= "1101";
    R7 <= "0100";

    --Index No:- 210206 => 110 011 010 100 011 110
    --Index No:- 210207 => 110 011 010 100 011 111
    Selector <= "110";
    wait for 200 ns;
```

```
    Selector <= "011";
    wait for 200 ns;

    Selector <= "010";
    wait for 200 ns;

    Selector <= "100";
    wait for 200 ns;

    Selector <= "111";
    wait for 200 ns;

end process;
end Behavioral;
```

4 Bit Adder/Subtracter

```
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_4 is
    Port ( A : in STD_LOGIC_VECTOR (3 DOWNTO 0);
           B : in STD_LOGIC_VECTOR (3 DOWNTO 0);
           Ctrl: in STD_LOGIC;
           S : out STD_LOGIC_VECTOR (3 DOWNTO 0);
           C_out : out STD_LOGIC;
           Zero : out STD_LOGIC);
end Add_Sub_4;

architecture Behavioral of Add_Sub_4 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,B_0,B_1,B_2,B_3: std_logic;
SIGNAL S_In : STD_LOGIC_VECTOR (3 DOWNTO 0);
begin
B_0 <= B(0) XOR Ctrl;
B_1 <= B(1) XOR Ctrl;
B_2 <= B(2) XOR Ctrl;
B_3 <= B(3) XOR Ctrl;

 FA_0 : FA
  port map (
   A => A(0),
   B => B_0,
   C_in => Ctrl,
   S => S_In(0),
   C_Out => FA0_C);

 FA_1 : FA
  port map (
   A => A(1),
   B => B_1,
   C_in => FA0_C,
   S => S_In(1),
   C_Out => FA1_C);

 FA_2 : FA
  port map (
   A => A(2),
   B => B_2,
   C_in => FA1_C,
   S => S_In(2),
   C_Out => FA2_C);
```

```
 FA_3 : FA
  port map (
   A => A(3),
   B => B_3,
   C_in => FA2_C,
   S => S_In(3),
   C_Out => C_out);
S <= S_In;
Zero <= NOT (S_In(0) OR S_In(1) OR S_In(2) OR S_In(3));
end Behavioral;
```

4 Bit Adder/Subtracter_TB

```
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 TB_Add_Sub_4 is
--  Port ( );
end TB_Add_Sub_4;

architecture Behavioral of TB_Add_Sub_4 is
component Add_Sub_4
   Port ( A : in STD_LOGIC_VECTOR (3 DOWNTO 0);
      B : in STD_LOGIC_VECTOR (3 DOWNTO 0);
      Ctrl: in STD_LOGIC;
      S : out STD_LOGIC_VECTOR (3 DOWNTO 0);
      C_out : out STD_LOGIC;
      Zero : out STD_LOGIC);
end component;
SIGNAL A, B, S  : std_logic_vector (3 downto 0);
SIGNAL Ctrl,C_out: std_logic;
begin
UUT: Add_Sub_4 PORT MAP(
 A => A,
 B => B,
 Ctrl => Ctrl,
 S => S,
 C_out => C_out
 );

process
begin
--Index No:- 210206 => 11 0011 0101 0001 1110
--Index No:- 210207 => 11 0011 0101 0001 1111
  A <= "1110";
  B <= "1111";
  Ctrl <= '0';

 WAIT FOR 100 ns;
  B <= "0001";

 WAIT FOR 100 ns;
  Ctrl <= '1';
  B <= "0101";


 WAIT FOR 100 ns;
 A <= "0101";
 B <= "0001";

 WAIT FOR 100 ns;
 Ctrl <= '1';
 A <= "0011";
 B <= "0101";


 WAIT FOR 100 ns;
```

```
 Ctrl <= '0';
 A <= "0011";
 B <= "0101";


end process;
end Behavioral;
```

3 Bit Adder

```
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_3 is
Port ( I_In : in STD_LOGIC_VECTOR (2 DOWNTO 0);
       I_Out : out STD_LOGIC_VECTOR (2 DOWNTO 0));
end Add_3;

architecture Behavioral of Add_3 is
  component FA
   port (
    A: in std_logic;
    B: in std_logic:='0';
    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  : std_logic;
begin
 FA_0 : FA
 port map (
  A => I_In(0),
  C_in => '1', -- Set to ground
  S => I_Out(0),
  C_Out => FA0_C);

FA_1 : FA
 port map (
  A => I_In(1),
  C_in => FA0_C,
  S => I_Out(1),
  C_Out => FA1_C);

FA_2 : FA
 port map (
  A => I_In(2),
  C_in => FA1_C,
  S => I_Out(2));

end Behavioral;
```

3 Bit Adder_TB

```
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 TB_Add_3 is
--  Port ( );
end TB_Add_3;

architecture Behavioral of TB_Add_3 is
component Add_3
        Port ( I_In : in STD_LOGIC_VECTOR (2 DOWNTO 0);
               I_Out : out STD_LOGIC_VECTOR (2 DOWNTO 0));
end component;
SIGNAL  I_In, I_Out : std_logic_vector (2 downto 0);
begin
UUT: Add_3
  port map(
           I_In  => I_In,
           I_Out => I_Out);

process
begin
   --Index No:- 210206 => 110 011 010 100 011 110
   --Index No:- 210207 => 110 011 010 100 011 111
   I_In <= "110";

   wait for 200ns;
   I_In <= "111";

   wait for 200ns;
   I_In <= "100";

   wait for 200ns;
   I_In <= "010";

   wait for 200ns;
   I_In <= "011";

   wait;

end process;
end Behavioral;
```

7 Segment Display

```
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 LUT_16_7 is
   Port ( Address : in STD_LOGIC_VECTOR (3 downto 0);
          Data : out STD_LOGIC_VECTOR (6 downto 0);
          Anode : out STD_LOGIC_VECTOR (3 downto 0));
end LUT_16_7;

architecture Behavioral of LUT_16_7 is

begin
process (Address) begin
   case Address is
                          --abcdefg
          when "0000" => Data <= "1000000"; -- "0"
          when "0001" => Data <= "1111001"; -- "1"
          when "0010" => Data <= "0100100"; -- "2"
          when "0011" => Data <= "0110000"; -- "3"
          when "0100" => Data <= "0011001"; -- "4"
          when "0101" => Data <= "0010010"; -- "5"
          when "0110" => Data <= "0000010"; -- "6"
          when "0111" => Data <= "1111000"; -- "7"
          when "1000" => Data <= "0000000"; -- "8"
          when "1001" => Data <= "0010000"; -- "9"
          when "1010" => Data <= "0001000"; -- a
          when "1011" => Data <= "0000011"; -- b
          when "1100" => Data <= "1000110"; -- C
          when "1101" => Data <= "0100001"; -- d
          when "1110" => Data <= "0000110"; -- E
          when "1111" => Data <= "0001110"; -- F
          when others => Data <= "0000000";
   end case;

end process;
   Anode <= "1110";
end Behavioral;
```

7 Segment Display_TB

```
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 TB_7_Seg is
--  Port ( );
end TB_7_Seg;

architecture Behavioral of TB_7_Seg is
component LUT_16_7
      Port ( Address : in STD_LOGIC_VECTOR (3 downto 0);
           Data : out STD_LOGIC_VECTOR (6 downto 0);
           Anode : out STD_LOGIC_VECTOR (3 downto 0));
end component;
signal Data_in, Anode : STD_LOGIC_VECTOR (3 downto 0);
signal seg7_out : STD_LOGIC_VECTOR (6 downto 0);
begin
UUT: LUT_16_7
   port map(
      Address => Data_in,
      Data => seg7_out,
      Anode => Anode
   );

process
begin
    --Index No:- 210206 => 11 0011 0101 0001 1110
    --Index No:- 210207 => 11 0011 0101 0001 1111
    Data_in <= "1110";
    wait for 200 ns;

    Data_in <= "1111";
    wait for 200 ns;

    Data_in <= "0101";
    wait for 200 ns;

    Data_in <= "0011";
    wait for 200 ns;

    Data_in <= "0001";
    wait for 200 ns;

end process;
end Behavioral;
```

Nano Processor

```
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 Nano_Processor is
   Port ( Clk : in STD_LOGIC;
          Reset : in STD_LOGIC;
          Zero : out STD_LOGIC;
          Overflow : out STD_LOGIC;
          LED : out STD_LOGIC_VECTOR (3 downto 0);
          Seg_7 : out STD_LOGIC_VECTOR (6 downto 0);
          Anode : out STD_LOGIC_VECTOR (3 downto 0));
end Nano_Processor;

architecture Behavioral of Nano_Processor is
component Slow_Clk
 Port ( Clk_in : in STD_LOGIC;
       Clk_out : out STD_LOGIC);
end component;

component P_ROM
   Port ( Address : in STD_LOGIC_VECTOR (2 downto 0);
      Instruction : out STD_LOGIC_VECTOR (11 downto 0));
end component;

component Ins_Decoder
   Port ( Ins : in STD_LOGIC_VECTOR (11 downto 0);
          Reg_Jum : in STD_LOGIC_VECTOR (3 downto 0);
          Reg_EN : out STD_LOGIC_VECTOR (2 downto 0);
          Load_Sel : out STD_LOGIC;
          Input_Value : out STD_LOGIC_VECTOR (3 downto 0);
          Reg_Sel_0 : out STD_LOGIC_VECTOR (2 downto 0);
          Reg_Sel_1 : out STD_LOGIC_VECTOR (2 downto 0);
          ADD_Sub_Sel : out STD_LOGIC;
          JMP_Flag : out STD_LOGIC;
          Jmp_Value : out STD_LOGIC_VECTOR (2 downto 0));
end component;

component Program_counter
    Port ( Clk_in : in STD_LOGIC;
      Res : in STD_LOGIC;
      D : in STD_LOGIC_VECTOR (2 downto 0);
      Memory : out STD_LOGIC_VECTOR (2 downto 0));
end component;

component Mux_2_Way_3_Bit
   Port ( A : in STD_LOGIC_VECTOR (2 downto 0);
      B : in STD_LOGIC_VECTOR (2 downto 0);
      JF : in STD_LOGIC;
      Q : out STD_LOGIC_VECTOR (2 downto 0));
end component;

component Add_3
   Port ( I_In : in STD_LOGIC_VECTOR (2 DOWNTO 0);
          I_Out : out STD_LOGIC_VECTOR (2 DOWNTO 0));
```

```
end component;

component Mux_8_Way_4_Bit
   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);
      Sel_R : in STD_LOGIC_VECTOR (2 downto 0);
      Q : out STD_LOGIC_VECTOR (3 downto 0));
end component;

component Mux_2_Way_4_Bit
   Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
      B : in STD_LOGIC_VECTOR (3 downto 0);
      JF : in STD_LOGIC;
      Q : out STD_LOGIC_VECTOR (3 downto 0));
end component;

component Add_Sub_4
   Port ( A : in STD_LOGIC_VECTOR (3 DOWNTO 0);
      B : in STD_LOGIC_VECTOR (3 DOWNTO 0);
      Ctrl: in STD_LOGIC;
      S : out STD_LOGIC_VECTOR (3 DOWNTO 0);
      C_out : out STD_LOGIC;
      Zero : out STD_LOGIC);
end component;

component LUT_16_7
       Port ( Address : in STD_LOGIC_VECTOR (3 downto 0);
            Data : out STD_LOGIC_VECTOR (6 downto 0);
            Anode : out STD_LOGIC_VECTOR (3 downto 0));
end component;

component Register_Bank
   Port ( Da_In : in STD_LOGIC_VECTOR (3 downto 0);
      Clk : in STD_LOGIC;
      S_Reg : in STD_LOGIC_VECTOR (2 downto 0);
      Res : in STD_LOGIC;
      R0_out : out STD_LOGIC_VECTOR (3 downto 0);
      R1_out : out STD_LOGIC_VECTOR (3 downto 0);
      R2_out : out STD_LOGIC_VECTOR (3 downto 0);
      R3_out : out STD_LOGIC_VECTOR (3 downto 0);
      R4_out : out STD_LOGIC_VECTOR (3 downto 0);
      R5_out : out STD_LOGIC_VECTOR (3 downto 0);
      R6_out : out STD_LOGIC_VECTOR (3 downto 0);
      R7_out : out STD_LOGIC_VECTOR (3 downto 0));
end component;

signal S_Clk, Load_Selector, Add_sub_EN, JUMP_flag : STD_LOGIC;
signal Mem_Sel, Register_En, R_Sel_M1, R_Sel_M0, Add_3_out, Address_to_JUMP, Address : STD_LOGIC_VECTOR (2 DOWNTO 0);
signal Instruction_Bus : STD_LOGIC_VECTOR (11 DOWNTO 0);
signal Data_Reg, Immediate_Value, Add_sub_out, Reg0, Reg1, Reg2, Reg3, Reg4, Reg5, Reg6, Reg7 : STD_LOGIC_VECTOR (3 DOWNTO 0);
signal Mux_8_4_1_Out, Mux_8_4_0_Out : STD_LOGIC_VECTOR (3 DOWNTO 0);
begin
Slow_Clock : Slow_Clk
 Port map ( Clk_in => Clk,
         Clk_out => S_Clk);

ProgramCounter : Program_counter
    Port map ( Clk_in => S_Clk,
            Res => Reset,
            D => Address,
```

```
            Memory => Mem_Sel );

Programe_ROM :  P_ROM
   Port map ( Address => Mem_Sel,
              Instruction => Instruction_Bus);

Instruction_Decoder : Ins_Decoder
   Port map ( Ins => Instruction_Bus,
         Reg_Jum => Mux_8_4_0_Out,
         Reg_EN => Register_En,
         Load_Sel => Load_Selector,
         Input_Value => Immediate_Value,
         Reg_Sel_0 => R_Sel_M0,
         Reg_Sel_1 => R_Sel_M1,
         ADD_Sub_Sel => Add_sub_EN,
         JMP_Flag => JUMP_flag,
         Jmp_Value => Address_to_JUMP );

RegisterBank :  Register_Bank
        Port map ( Da_In => Data_Reg,
              Clk => S_Clk,
              S_Reg => Register_En,
              Res => Reset,
              R0_out => Reg0,
              R1_out => Reg1,
              R2_out => Reg2,
              R3_out => Reg3,
              R4_out => Reg4,
              R5_out => Reg5,
              R6_out => Reg6,
              R7_out => Reg7);

Mux_2_4 :  Mux_2_Way_4_Bit
         Port map ( A => Add_sub_out,
                    B => Immediate_Value,
                    JF => Load_Selector,
                    Q => Data_Reg );

Add_Sub_4_Unit :  Add_Sub_4
         Port map ( A => Mux_8_4_1_Out,
               B => Mux_8_4_0_Out,
               Ctrl => Add_sub_EN,
               S => Add_sub_out,
               C_out => Overflow,
               Zero => Zero);

Mux_8_4_0 : Mux_8_Way_4_Bit
        Port map(
            R0 => Reg0,
            R1 => Reg1,
            R2 => Reg2,
            R3 => Reg3,
            R4 => Reg4,
            R5 => Reg5,
            R6 => Reg6,
            R7 => Reg7,
            Sel_R => R_Sel_M0,
            Q => Mux_8_4_0_Out);

Mux_8_4_1 : Mux_8_Way_4_Bit
                  Port map (
                      R0 => Reg0,
                      R1 => Reg1,
                      R2 => Reg2,
                      R3 => Reg3,
                      R4 => Reg4,
```

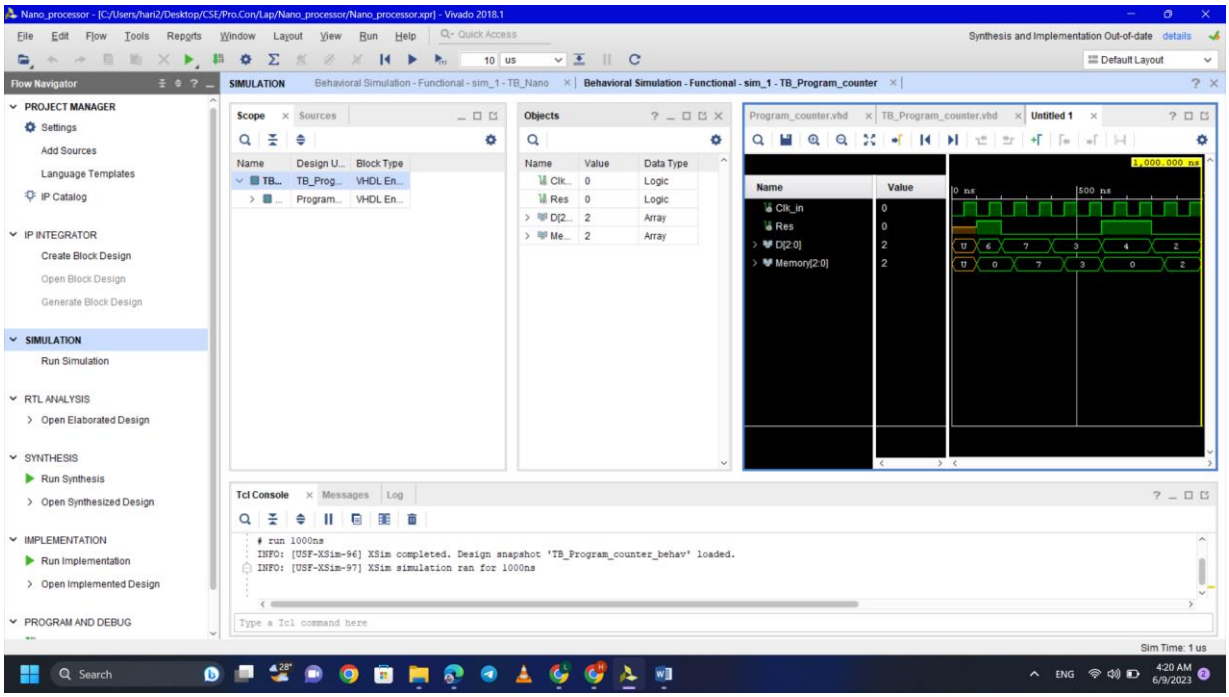
```
                    R5 => Reg5,
                    R6 => Reg6,
                    R7 => Reg7,
                    Sel_R => R_Sel_M1,
                    Q => Mux_8_4_1_Out );
Adder_3_Bit :  Add_3
            Port map ( I_In => Mem_Sel,
                  I_Out => Add_3_out);

Mux_2_3 : Mux_2_Way_3_Bit
      Port map( A => Add_3_out,
           B => Address_to_JUMP,
           JF => JUMP_flag,
           Q => Address);

Seg_7_dis : LUT_16_7
      Port map ( Address => Reg7,
            Data => Seg_7,
            Anode => Anode );

LED <= Reg7;
end Behavioral;
```

Nano Processor_TB

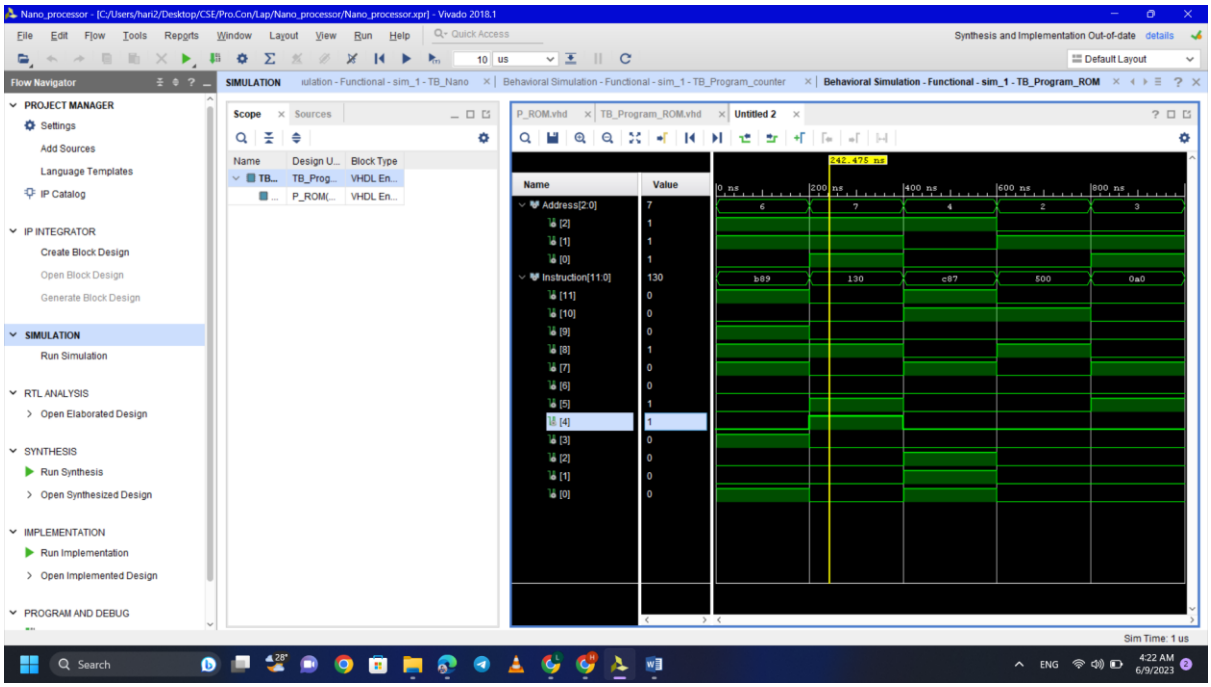
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- 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 TB_Nano is
--  Port ( );
end TB_Nano;

architecture Behavioral of TB_Nano is
component Nano_Processor
   Port ( Clk : in STD_LOGIC;
      Reset : in STD_LOGIC;
      Zero : out STD_LOGIC;
      Overflow : out STD_LOGIC;
      LED : out STD_LOGIC_VECTOR (3 downto 0);
      Seg_7 : out STD_LOGIC_VECTOR (6 downto 0);
      Anode : out STD_LOGIC_VECTOR (3 downto 0));
end component;
signal Clk : STD_LOGIC:= '0';
signal Reset, Zero, Overflow : STD_LOGIC;
signal Led_out, Anode : STD_LOGIC_VECTOR (3 downto 0);
signal To_7_seg : STD_LOGIC_VECTOR (6 downto 0);

begin
UUT: Nano_Processor
   port map(
        Clk => Clk,
        Reset => Reset,
        Zero => Zero,
        Overflow => Overflow,
        LED => Led_out,
        Seg_7 => To_7_seg,
        Anode => Anode
   );
process
begin
    wait for 5 ns;
    Clk <= NOT(Clk);
end process;
process
begin
    wait for 95 ns;
    Reset <= '1';

    wait for 5 ns;
    Reset <= '0';
    wait;
end process;
end Behavioral;
```

• All timing diagrams

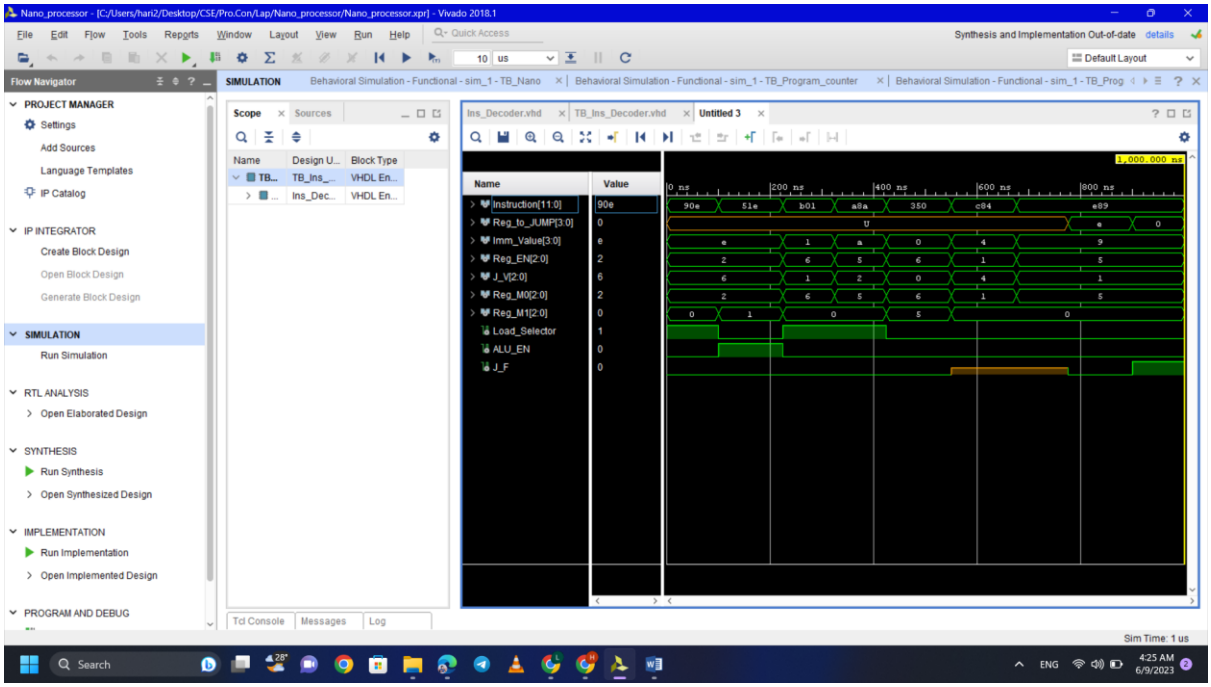
Program Counter



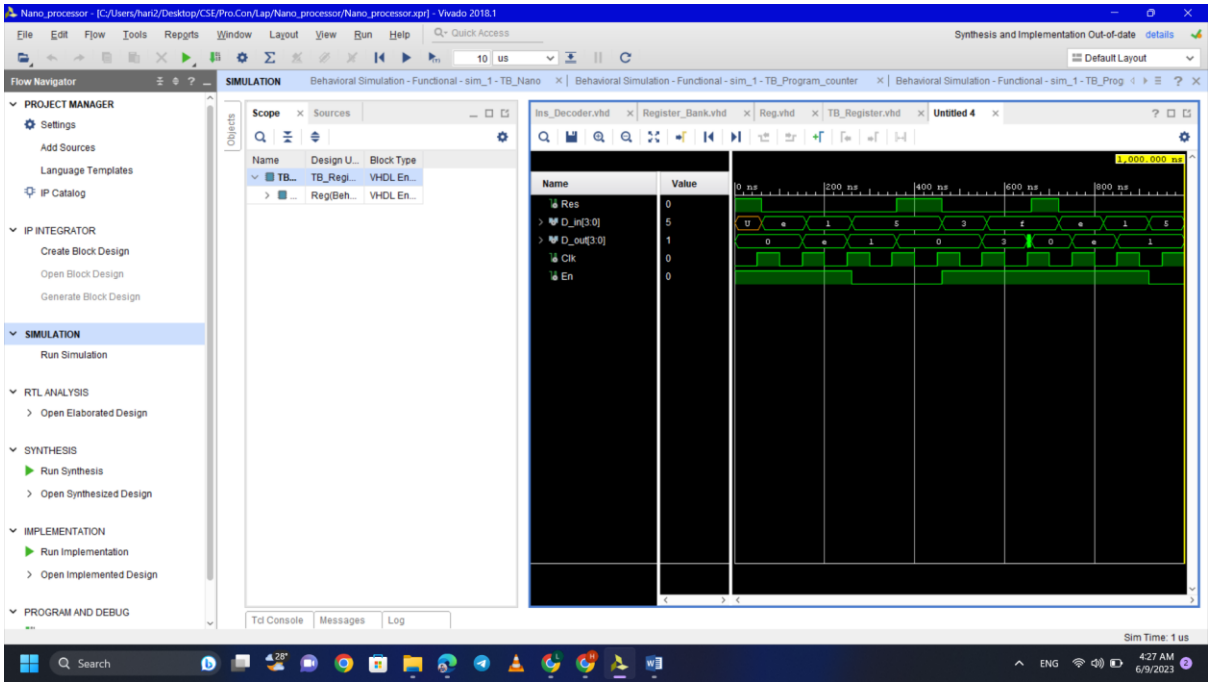
Programmable ROM



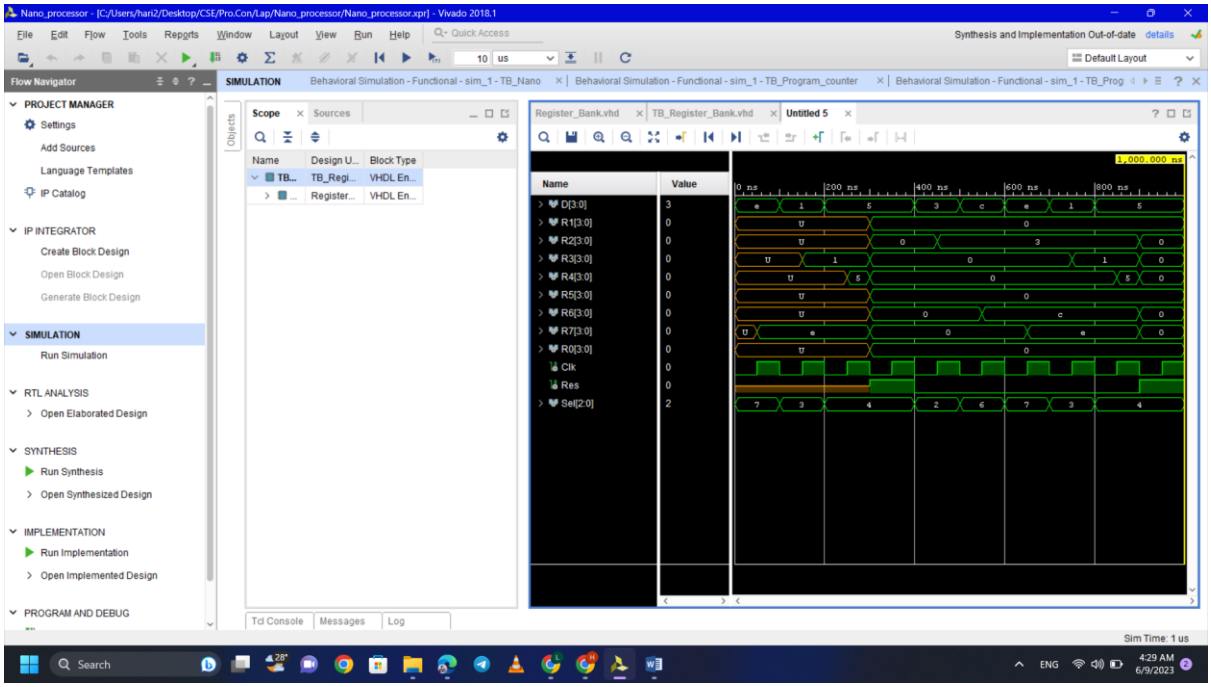
Instruction Decoder



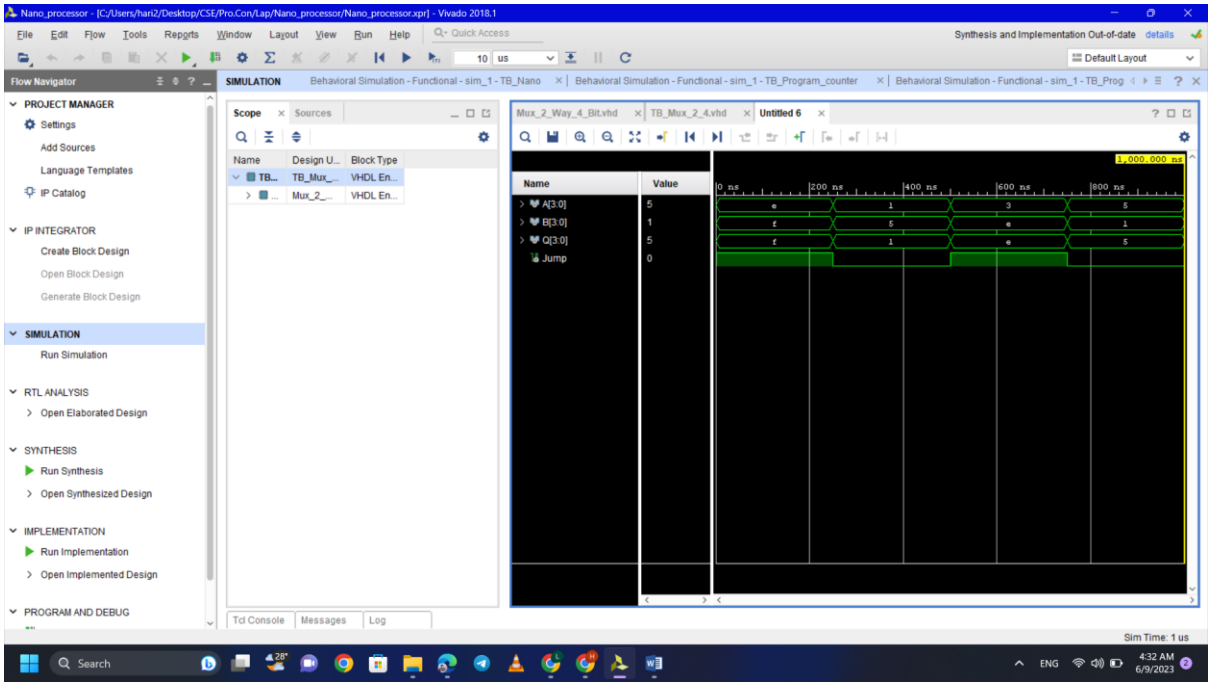
Register



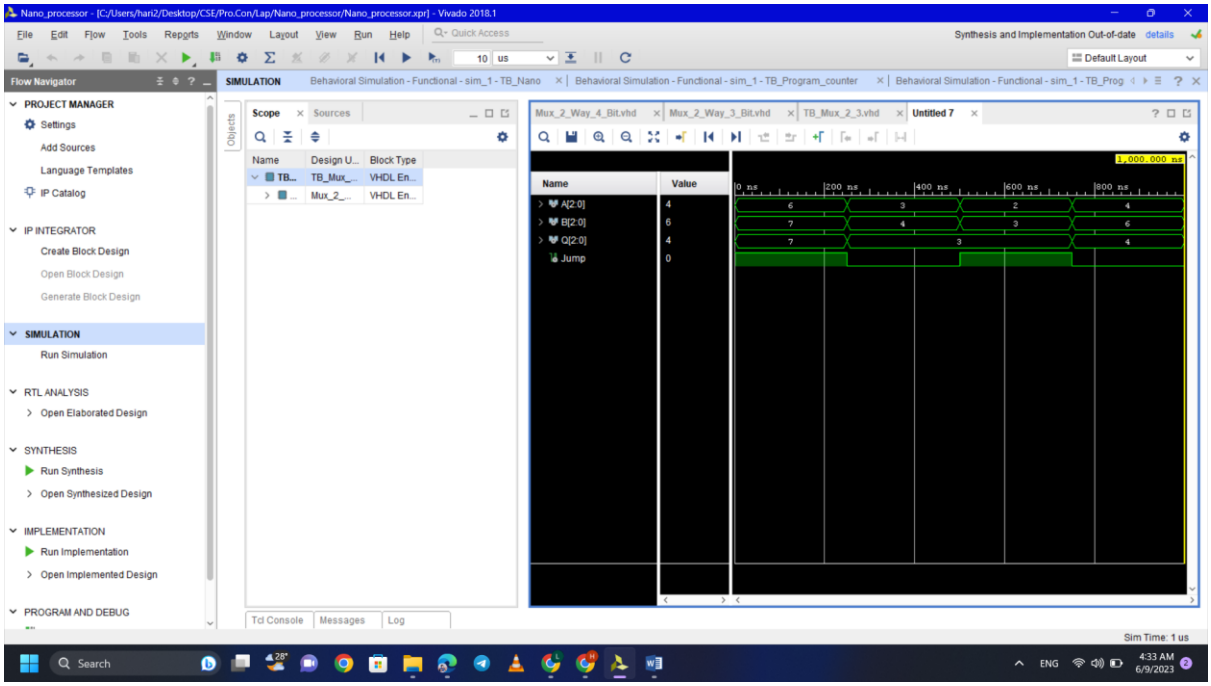
Register Bank



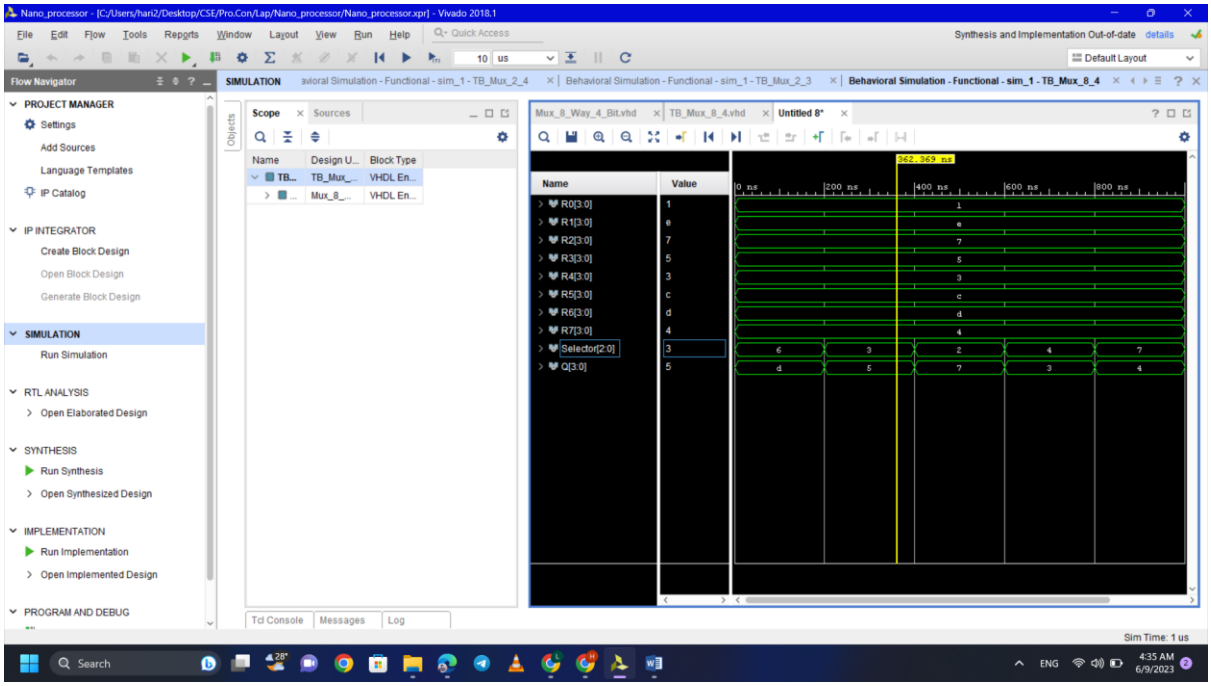
Mux_2_Way_4_Bit



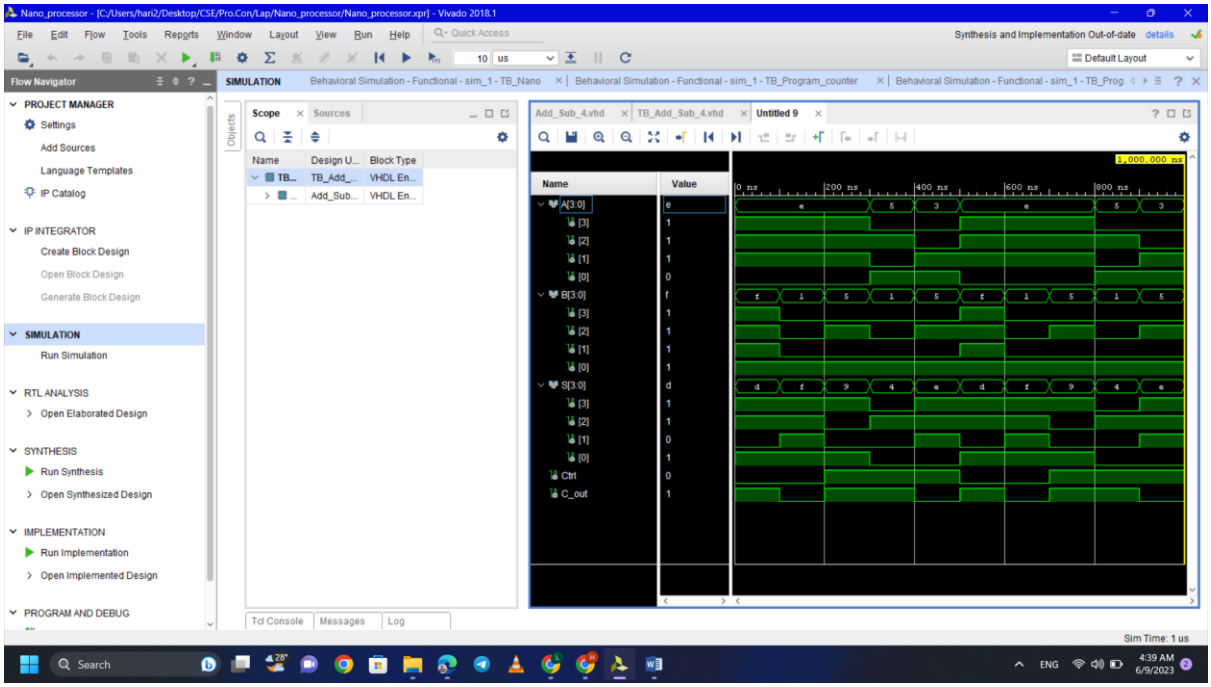
Mux_2_Way_3_Bit



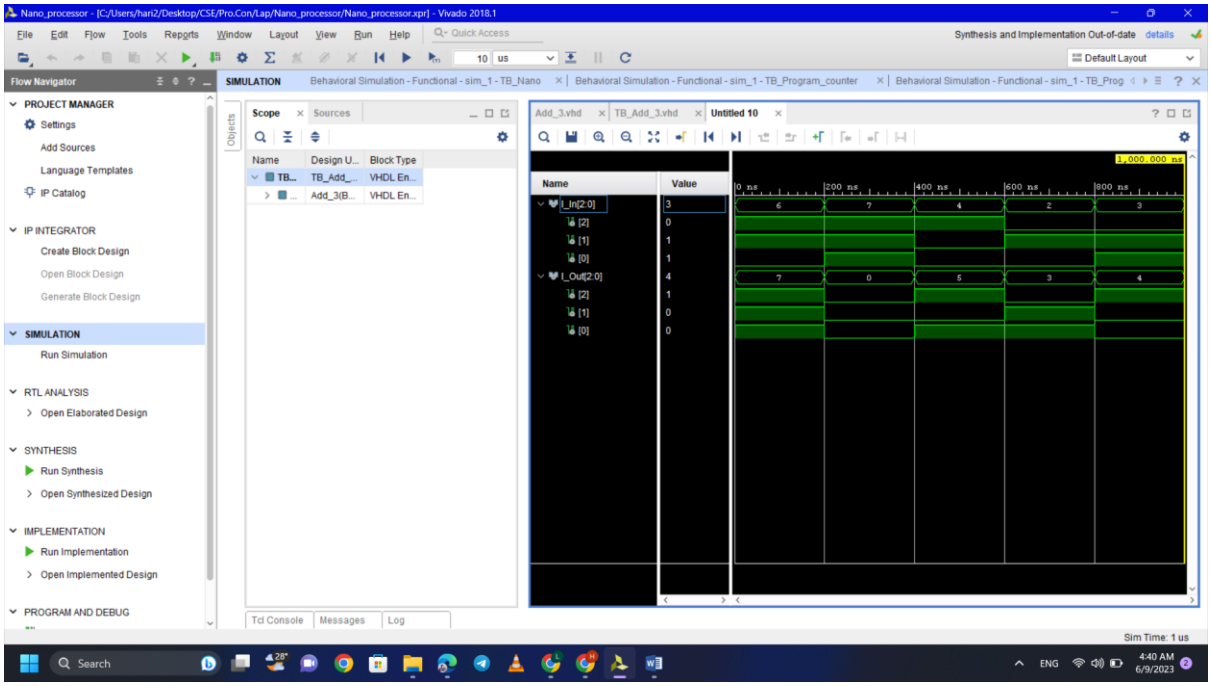
Mux_8_Way_4_Bit



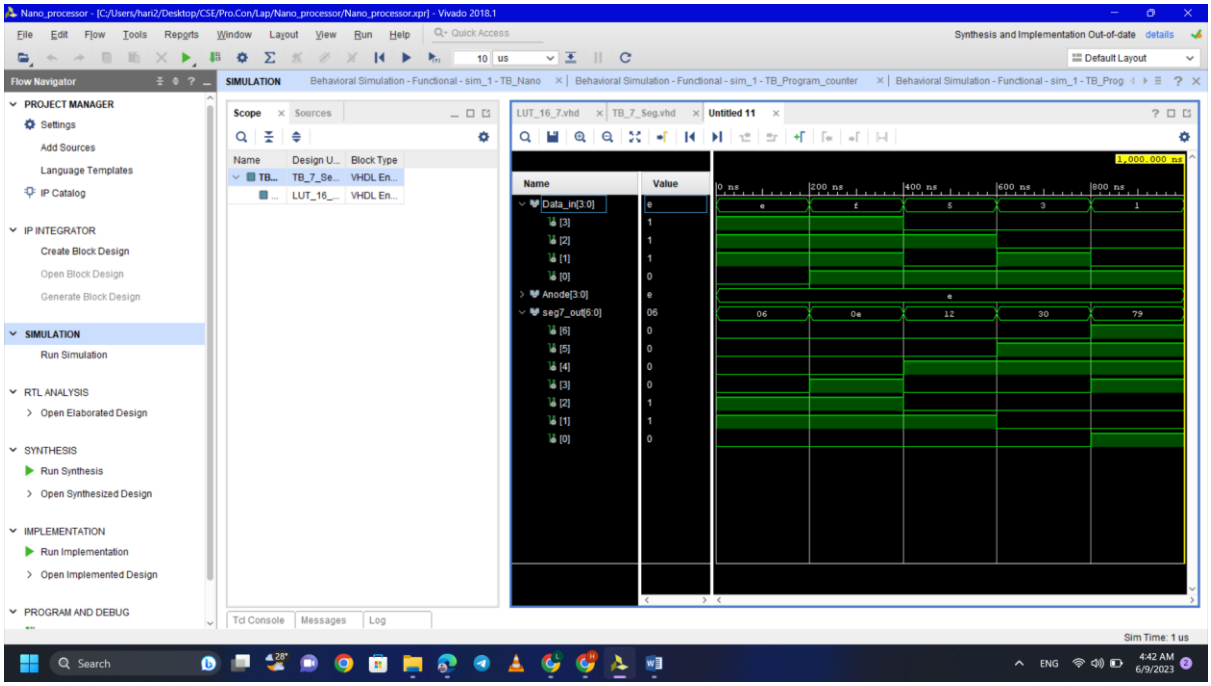
4 Bit Adder/Subtracter



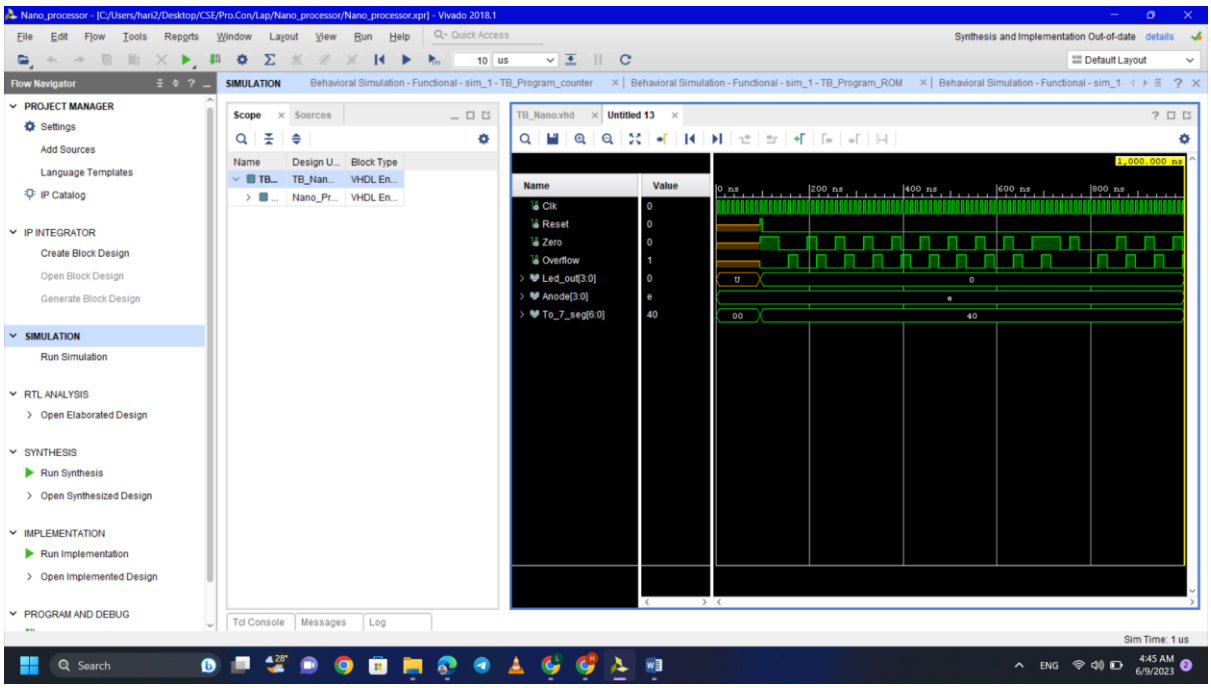
3 Bit Adder



7 Segment Display



Nano Processor



• Conclusions from the lab

In this lab, we designed and developed a 4-bit processor capable of executing 4 instructions. We gained knowledge on designing and developing the arithmetic unit, instruction decoder, and multiplexers/tri-state buses. Through simulation and testing on the BASYS 3 development board, we were able to verify the functionality of our nanoprocessor. Overall, this lab provided valuable experience in processor design and teamwork skills.

• Clearly describe the contribution of each team member to project and number of hours spent

N. Harikishna – 210206B (23 Hours)

- Program counter
- Instruction Decoder
- Register
- Register Bank
- Multiplexer 8 Way 4 Bit
- Nano processor
- Adder/ Subractor 4 Bit_TB
- Adder 3 Bit_TB
- 7 Segment Display_TB
- Programmable ROM_TB
- Multiplexer 2 Way 4 Bit_TB
- Multiplexer 2 Way 3 Bit_TB

J. Harismenan - 210207E (21 Hours)

- Adder/Subractor 4 Bit
- Adder 3 Bit
- 7 Segment Display
- Programmable ROM
- Multiplexer 2 Way 4 Bit
- Multiplexer 2 Way 3 Bit
- Program counter _TB
- Instruction Decoder _TB
- Register _TB
- Register Bank _TB
- Multiplexer 8 Way 4 Bit _TB
- Nano processor _TB