# LAB REPORT

## Lab 9-10 – Nano-Processor Design Competition

CS 2052 Computer Architecture

Dept. of Computer Science and Engineering, University of Moratuwa

- **TEAM MEMBERS**

| NAME | INDEX NUMBER |
|---|---|
| E. A. C. Chandeepa | 200081B |
| M. M. H. A. Premarathna | 200479D |
| W. A. S. P. Wijesuriya | 200731U |
| P. S. N. Pathirathne | 200450G |
| L. T. Darshani | 200100K |

- **LAB TASK**

In this lab, we were assigned to design a simple microprocessor (hence, called a nano processor) capable of executing a simple set of instructions.

Step 01: Identifying the given lab question.

Step 02: Designing the internal structure of the Instruction Decoder and identifying the role of each of the output pins and how to activate them when necessary.

Step 03: Building necessary sub-components and testing each component using simulation.

- 4-bit Add/Subtract Unit
- 3-bit Adder
- 3-bit Program Counter
- 2-way 3-bit Multiplexer
- 2-way 4-bit Multiplexer
- 8-way 4-bit Multiplexer
- Register Bank
- Program ROM

Step 04: Building the top-level design.

Step 05: Writing an Assembly program to calculate the total of all integers between 1 and 3 and store the final answer in Register R7.

Step 06: Converting the Assembly program to machine code and hard code it to ROM.

Step 07: Connecting inputs and outputs.

Step 08: Testing on BASYS3 board and verifying the functionality of the nano processor.

**KEY** (Links to each topic)

## VHDL CODES

### DESIGN SOURCE CODES

- **4-bit Add/Subtract Unit**

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 06/02/2022 10:58:31 AM
-- Design Name:
-- Module Name: RCA_4 - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


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_4 is
   Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
          B : in STD_LOGIC_VECTOR (3 downto 0);
          C_in : in std_logic;
          S : out STD_LOGIC_VECTOR (3 downto 0);
          C_out : out STD_LOGIC;
          Zero : out STD_LOGIC);
end RCA_4;

architecture Behavioral of RCA_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_C, FA1_C, FA2_C: std_logic;
   SIGNAL B_Sig, S_FA : STD_LOGIC_VECTOR (3 downto 0);


begin

  B_Sig(0) <= (B(0) XOR C_in);
  B_Sig(1) <= (B(1) XOR C_in);
  B_Sig(2) <= (B(2) XOR C_in);
  B_Sig(3) <= (B(3) XOR C_in);


  FA_0 : FA
      port map (
      A => A(0),
      B => B_Sig(0),
      C_in => C_in,
      S => S_FA(0),
      C_out => FA0_C);

   FA_1 : FA
      port map (
      A => A(1),
      B => B_Sig(1),
      C_in => FA0_C,
      S => S_FA(1),
      C_out => FA1_C);

   FA_2 : FA
      port map (
      A => A(2),
      B => B_Sig(2),
      C_in => FA1_C,
      S => S_FA(2),
      C_out => FA2_C);

   FA_3 : FA
      port map (
      A => A(3),
      B => B_Sig(3),
      C_in => FA2_C,
      S => S_FA(3),
      C_out => C_out);

S <= S_FA;
Zero <= NOT(S_FA(0)) AND NOT(S_FA(1)) AND NOT(S_FA(2)) AND NOT(S_FA(3)); --AND
NOT(C_out);

end Behavioral
```

- **3-bit Adder**

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 06/02/2022 10:58:31 AM
-- Design Name:
-- Module Name: RCA_4 - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
```

```
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------

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 is
  Port ( A : in STD_LOGIC_VECTOR (2 downto 0);
         C_in : in STD_LOGIC;
         S : out STD_LOGIC_VECTOR (2 downto 0);
         C_out : out STD_LOGIC);
end RCA_3;

architecture Behavioral of RCA_3 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 : std_logic;
```

```
    SIGNAL B : STD_LOGIC_VECTOR (2 downto 0) := "001";

begin

   FA_0 : FA
       port map (
       A => A(0),
       B => B(0),
       C_in => '0',
       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 => C_out);
end Behavioral;
```

- **3-bit Program Counter**

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 06/21/2022 09:05:50 AM
-- Design Name:
-- Module Name: Reg - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


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_3Bit is
    Port ( D : in STD_LOGIC_VECTOR (2 downto 0);
           En : in STD_LOGIC;
           Reset : in STD_LOGIC;
           Clk : in STD_LOGIC;
           Q : out STD_LOGIC_VECTOR (2 downto 0));
```

```
end PC_3Bit;

architecture Behavioral of PC_3Bit is

begin


process (Clk) begin

    if (rising_edge(Clk)) then
      if (Reset = '0') then
        if (NOT(D="000")) then
          Q <= D;
        end if;

      else
        Q <= "000";
      end if;
    end if;

end process;

end Behavioral;
```

- **2-way 3-bit Multiplexer**

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 06/05/2022 08:57:24 PM
-- Design Name:
-- Module Name: Mux_8_to_1 - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
```

```
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


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_to_1_3Bit is
    Port ( S : in STD_LOGIC;
           D0 : in STD_LOGIC_VECTOR (2 downto 0);
           D1 : in STD_LOGIC_VECTOR (2 downto 0);
           Y : out STD_LOGIC_VECTOR (2 downto 0));
end Mux_2_to_1_3Bit;

architecture Behavioral of Mux_2_to_1_3Bit is

begin

    Y(0) <= (NOT(S) AND (D0(0))) OR (S AND (D1(0)));

    Y(1) <= (NOT(S) AND (D0(1))) OR (S AND (D1(1)));

    Y(2) <= (NOT(S) AND (D0(2))) OR (S AND (D1(2)));



end Behavioral;
```

- **2-way 4-bit Multiplexer**

```
----------------------------------------------------------------------------------
-- Company: 
-- Engineer: 
-- 
-- Create Date: 06/05/2022 08:57:24 PM
-- Design Name: 
-- Module Name: Mux_8_to_1 - Behavioral
-- Project Name: 
-- Target Devices: 
-- Tool Versions: 
-- Description: 
-- 
-- Dependencies: 
-- 
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
-- 
----------------------------------------------------------------------------------


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_to_1_4Bit is
    Port ( S : in STD_LOGIC;
           RCA_out : in STD_LOGIC_VECTOR (3 downto 0); --RCA
           Imm_Val : in STD_LOGIC_VECTOR (3 downto 0); --Immediate Value
           Y : out STD_LOGIC_VECTOR (3 downto 0));
end Mux_2_to_1_4Bit;
```

```
architecture Behavioral of Mux_2_to_1_4Bit is

begin


--S = 0 ---> RCA_out
--S = 1 ---> Immediate value


   Y(0) <= (NOT(S) AND (RCA_out(0))) OR (S AND (Imm_Val(0)));

   Y(1) <= (NOT(S) AND (RCA_out(1))) OR (S AND (Imm_Val(1)));

   Y(2) <= (NOT(S) AND (RCA_out(2))) OR (S AND (Imm_Val(2)));

   Y(3) <= (NOT(S) AND (RCA_out(3))) OR (S AND (Imm_Val(3)));

end Behavioral;
```

- **8-way 4-bit Multiplexer**

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 07/10/2022 04:30:20 PM
-- Design Name:
-- Module Name: Mux_8way_4bit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------
```

```
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_8way_4bit is
   Port (
          S : in STD_LOGIC_VECTOR (2 downto 0);
          D0 : in STD_LOGIC_VECTOR (3 downto 0);
          D1 : in STD_LOGIC_VECTOR (3 downto 0);
          D2 : in STD_LOGIC_VECTOR (3 downto 0);
          D3 : in STD_LOGIC_VECTOR (3 downto 0);
          D4 : in STD_LOGIC_VECTOR (3 downto 0);
          D5 : in STD_LOGIC_VECTOR (3 downto 0);
          D6 : in STD_LOGIC_VECTOR (3 downto 0);
          D7 : in STD_LOGIC_VECTOR (3 downto 0);
          Y : out STD_LOGIC_VECTOR (3 downto 0));
end Mux_8way_4bit;

architecture Behavioral of Mux_8way_4bit is
component Mux_8_to_1
   Port ( S : in STD_LOGIC_VECTOR (2 downto 0);
         D : in STD_LOGIC_VECTOR (7 downto 0);
         EN : in STD_LOGIC;
         Y : out STD_LOGIC);
end component;
begin

Mux_8_to_1_0: Mux_8_to_1
PORT MAP(
   S => S,
   D(0)=>D0(0),
   D(1)=>D1(0),
   D(2)=>D2(0),
   D(3)=>D3(0),
   D(4)=>D4(0),
   D(5)=>D5(0),
   D(6)=>D6(0),
   D(7)=>D7(0),
   EN=>'1',
   Y=>Y(0)
);
Mux_8_to_1_1: Mux_8_to_1
PORT MAP(
   S => S,
   D(0)=>D0(1),
   D(1)=>D1(1),
   D(2)=>D2(1),
   D(3)=>D3(1),
   D(4)=>D4(1),
   D(5)=>D5(1),
   D(6)=>D6(1),
   D(7)=>D7(1),
   EN=>'1',
   Y=>Y(1)
);

Mux_8_to_1_2: Mux_8_to_1
PORT MAP(
   S => S,
   D(0)=>D0(2),
   D(1)=>D1(2),
   D(2)=>D2(2),
   D(3)=>D3(2),
   D(4)=>D4(2),
   D(5)=>D5(2),
   D(6)=>D6(2),
   D(7)=>D7(2),
   EN=>'1',
   Y=>Y(2)
);

Mux_8_to_1_3: Mux_8_to_1
```

```
PORT MAP(
   S => S,
   D(0)=>D0(3),
   D(1)=>D1(3),
   D(2)=>D2(3),
   D(3)=>D3(3),
   D(4)=>D4(3),
   D(5)=>D5(3),
   D(6)=>D6(3),
   D(7)=>D7(3),
   EN=>'1',
   Y=>Y(3)
);

end Behavioral;
```

- **Register Bank**

```
----------------------------------------------------------------------------------
-- Company: 
-- Engineer: 
-- 
-- Create Date: 07/10/2022 05:59:28 PM
-- Design Name: 
-- Module Name: Reg_Bank - Behavioral
-- Project Name: 
-- Target Devices: 
-- Tool Versions: 
-- Description: 
-- 
-- Dependencies: 
-- 
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
-- 
----------------------------------------------------------------------------------
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_Bank is
   Port ( Reg_en : in STD_LOGIC_VECTOR (2 downto 0);
          Clk : in STD_LOGIC;
          Reset : in STD_LOGIC;
          Input_D : in STD_LOGIC_VECTOR (3 downto 0);
          Q0 : out STD_LOGIC_VECTOR (3 downto 0);
          Q1 : out STD_LOGIC_VECTOR (3 downto 0);
          Q2 : out STD_LOGIC_VECTOR (3 downto 0);
          Q3 : out STD_LOGIC_VECTOR (3 downto 0);
          Q4 : out STD_LOGIC_VECTOR (3 downto 0);
          Q5 : out STD_LOGIC_VECTOR (3 downto 0);
          Q6 : out STD_LOGIC_VECTOR (3 downto 0);
          Q7 : out STD_LOGIC_VECTOR (3 downto 0));
end Reg_Bank;

architecture Behavioral of Reg_Bank is

component Decoder_3_to_8
   Port ( I : in STD_LOGIC_VECTOR (2 downto 0);
          EN : in STD_LOGIC;
          Y : out STD_LOGIC_VECTOR (7 downto 0));
end component;

component Reg
   Port ( D : in STD_LOGIC_VECTOR (3 downto 0);
          En : in STD_LOGIC;
          Clk : in STD_LOGIC;
          Reset : in STD_LOGIC;
          Q : out STD_LOGIC_VECTOR (3 downto 0));
end component;

SIGNAL Y : STD_LOGIC_VECTOR (7 downto 0);
```

```
begin

Decoder_3_to_8_0: Decoder_3_to_8
PORT MAP(
   I=>Reg_en,
   EN =>'1',
   Y=>Y
);

R0 : Reg
PORT MAP(
   D=>"0000",
   En => '0',
   Clk => Clk,
   Reset => Reset,
   Q=> Q0
);

R1 : Reg
PORT MAP(
   D=>Input_D,
   En => Y(1),
   Clk => Clk,
   Reset => Reset,
   Q=> Q1
);

R2 : Reg
PORT MAP(
   D=>Input_D,
   En => Y(2),
   Clk => Clk,
   Reset => Reset,
   Q=> Q2
);

R3 : Reg
PORT MAP(
   D=>Input_D,
   En => Y(3),
   Clk => Clk,
   Reset => Reset,
   Q=> Q3
);

R4 : Reg
PORT MAP(
   D=>Input_D,
   En => Y(4),
   Clk => Clk,
   Reset => Reset,
   Q=> Q4
);

R5 : Reg
PORT MAP(
   D=>Input_D,
   En => Y(5),
   Clk => Clk,
   Reset => Reset,
   Q=> Q5
);

R6 : Reg
PORT MAP(
   D=>Input_D,
   En => Y(6),
   Clk => Clk,
   Reset => Reset,
   Q=> Q6
);

R7 : Reg
PORT MAP(
   D=>Input_D,
   En => Y(7),
   Clk => Clk,
   Reset => Reset,
   Q=> Q7
);
end Behavioral;
```

- **Program ROM**

```
----------------------------------------------------------------------------------
-- Company: 
-- Engineer: 
-- 
-- Create Date: 07/19/2022 11:03:30 AM
-- Design Name: 
-- Module Name: Program_ROM_LUT_8_12 - Behavioral
-- Project Name: 
-- Target Devices: 
-- Tool Versions: 
-- Description: 
-- 
-- Dependencies: 
-- 
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
-- 
----------------------------------------------------------------------------------


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_ROM_LUT_8_12 is
    Port ( address : in STD_LOGIC_VECTOR (2 downto 0);
   data : out STD_LOGIC_VECTOR (11 downto 0));
end Program_ROM_LUT_8_12;
```

```
architecture Behavioral of Program_ROM_LUT_8_12 is

type rom_type is array (0 to 7) of std_logic_vector(11 downto 0);

signal program_ROM : rom_type := (

"100010000011", -- 0
"100100000001", --1
"010100000000", --2
"001110010000", --3
"000010100000", --4
"110010000111", --5
"110000000011", --6
"110000000111" --7
);


begin

data <= program_ROM(to_integer(unsigned(address)));


end Behavioral;
```

| **Line No. of Assembly Code** | **Instructions** | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **I(11)** | **I(10)** | **I (9)** | **I (8)** | **I (7)** | **I (6)** | **I (5)** | **I (4)** | **I (3)** | **I (2)** | **I(1)** | **I(0)** |
| 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 2 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 3 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 6 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 7 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 8 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

- **Instruction Decoder**

```
----------------------------------------------------------------------------------
-- Company: 
-- Engineer: 
-- 
-- Create Date: 07/10/2022 06:49:02 PM
-- Design Name: 
-- Module Name: Ins_Decoder - Behavioral
-- Project Name: 
-- Target Devices: 
-- Tool Versions: 
-- Description: 
-- 
-- Dependencies: 
-- 
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
-- 
----------------------------------------------------------------------------------


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 ( Instruction : in STD_LOGIC_VECTOR (11 downto 0);
         Reg_en : out STD_LOGIC_VECTOR (2 downto 0);
         Load_sel : out STD_LOGIC;
         Imm_Val : out STD_LOGIC_VECTOR (3 downto 0);
         Reg_Sel_A : out STD_LOGIC_VECTOR (2 downto 0);
```

```
           Reg_Sel_B : out STD_LOGIC_VECTOR (2 downto 0);
           Add_Sub_Sel : out STD_LOGIC;
           Jump_add : out STD_LOGIC_VECTOR (2 downto 0);
           Jump_flag : out STD_LOGIC;
           Reg_Chk_Jmp : in STD_LOGIC_VECTOR (3 downto 0));
end Ins_Decoder;

architecture Behavioral of Ins_Decoder is


begin


process(Instruction, Reg_Chk_Jmp)
   begin


   Reg_en <= "000";
   Load_sel <= '0';
   Imm_Val <= "0000";
   Reg_Sel_A <= "000";
   Reg_Sel_B <= "000";
   Add_Sub_Sel <= '0';
   Jump_add <= "000";
   Jump_flag <= '0';




    --==============================================================

   If (Instruction(11)='0' and Instruction(10)='0') then
      --Add instruction

      --Reg En
      Reg_en<= Instruction (9 downto 7);

      --Load select
      Load_sel <= '0';

      --Reg Sel 1
    Reg_Sel_A <= Instruction (9 downto 7);

    --Reg Sel 2
    Reg_Sel_B <= Instruction (6 downto 4);

    --Add/Sub
    Add_Sub_Sel <= '0';

  --=========================================================

  --=========================================================

elsIf (Instruction(11)='0' and Instruction(10)='1') then
      --Neg instruction

      --Reg En
      Reg_en <= Instruction(9 downto 7);

      --Load select
      Load_sel <= '0';

      --Reg Sel 1
      Reg_Sel_A <= "000";

      --Reg Sel 2
      Reg_Sel_B <= Instruction (9 downto 7);

      --Add/Sub
      Add_Sub_Sel <= '1';

  --=========================================================

  --=========================================================

 elsIf (Instruction(11)='1' and Instruction(10)='0') then
        --Move instruction

        --Reg En
        Reg_en <= Instruction(9 downto 7);

        --Load select
```

```
            Load_sel <= '1';

            --Immediate Value
            Imm_Val <= Instruction(3 downto 0);


        --=============================================================

        --=============================================================

           elsIf (Instruction(11)='1' and Instruction(10)='1') then
              --Jump instruction

              --Reg En
              Reg_en <= "000";


              --Reg Sel 1
              Reg_Sel_A <= Instruction (9 downto 7);

             --if Reg_Chk_Jmp = 0 then jump flag = 1
             if (Reg_Chk_Jmp = "0000") then
                Jump_flag <= '1';
                Jump_add <= Instruction(2 downto 0);
             else
                Jump_flag <= '0';
             end if;

    end if;
end process;

end Behavioral;
```

- **Slow Clock**

```
----------------------------------------------------------------------------------
-- Company: 
-- Engineer: 
-- 
-- Create Date: 06/20/2022 07:25:25 PM
-- Design Name: 
-- Module Name: Slow_Clk - Behavioral
-- Project Name: 
-- Target Devices: 
-- Tool Versions: 
-- Description: 
-- 
-- Dependencies: 
-- 
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
-- 
----------------------------------------------------------------------------------


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

architecture Behavioral of Slow_Clk is
```

```
signal count : integer := 1;
signal clk_status : std_logic := '0';

begin

   --1Hz for 100 MHz clock

   process (Clk_in) begin
      if (rising_edge(Clk_in)) then
         count <= count + 1;
         if (count = 5) then
            clk_status <= not clk_status;
            Clk_out <= clk_status;
            count <= 1;
         end if;
     end if;
   end process;

end Behavioral;
```

- **Nano Processor**

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 07/11/2022 12:13:32 AM
-- Design Name:
-- Module Name: Nanoprocessor - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------

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 Nanoprocessor is
   Port ( Clk : in STD_LOGIC;
          Reset : in STD_LOGIC;
          Overflow : out STD_LOGIC;
          Zero : out STD_LOGIC;
          R7 : out STD_LOGIC_VECTOR (3 downto 0);
          Seven_seg_out : out STD_LOGIC_VECTOR (6 downto 0);
```

```
        an : out STD_LOGIC_VECTOR (3 downto 0)
        );
end Nanoprocessor;

architecture Behavioral of Nanoprocessor is

--Importing components================================

component SevenSeg_LUT_16_7
   Port ( address : in STD_LOGIC_VECTOR (3 downto 0);
      data : out STD_LOGIC_VECTOR (6 downto 0));
end component;

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

component Reg_Bank
   Port ( Reg_en : in STD_LOGIC_VECTOR (2 downto 0);
      Clk : in STD_LOGIC;
      Reset : in STD_LOGIC;
      Input_D : in STD_LOGIC_VECTOR (3 downto 0);
      Q0 : out STD_LOGIC_VECTOR (3 downto 0);
      Q1 : out STD_LOGIC_VECTOR (3 downto 0);
      Q2 : out STD_LOGIC_VECTOR (3 downto 0);
      Q3 : out STD_LOGIC_VECTOR (3 downto 0);
      Q4 : out STD_LOGIC_VECTOR (3 downto 0);
      Q5 : out STD_LOGIC_VECTOR (3 downto 0);
      Q6 : out STD_LOGIC_VECTOR (3 downto 0);
      Q7 : out STD_LOGIC_VECTOR (3 downto 0));
end component;


component AU
   Port ( D0 : in STD_LOGIC_VECTOR (3 downto 0);
      D1 : in STD_LOGIC_VECTOR (3 downto 0);
      D2 : in STD_LOGIC_VECTOR (3 downto 0);
      D3 : in STD_LOGIC_VECTOR (3 downto 0);
      D4 : in STD_LOGIC_VECTOR (3 downto 0);
      D5 : in STD_LOGIC_VECTOR (3 downto 0);
```

```
        D6 : in STD_LOGIC_VECTOR (3 downto 0);
        D7 : in STD_LOGIC_VECTOR (3 downto 0);
        Reg_SelA : in STD_LOGIC_VECTOR (2 downto 0);
        Reg_SelB : in STD_LOGIC_VECTOR (2 downto 0);
        Add_Sub_Sel : in STD_LOGIC;
        Reg_Chk_Jmp : out STD_LOGIC_VECTOR (3 downto 0);
        RCA_out : out STD_LOGIC_VECTOR (3 downto 0);
        Overflow : out STD_LOGIC;
        Zero : out STD_LOGIC);
end component;


component Mux_2_to_1_4Bit
    Port ( S : in STD_LOGIC;
        RCA_out : in STD_LOGIC_VECTOR (3 downto 0); --RCA
        Imm_Val : in STD_LOGIC_VECTOR (3 downto 0); --Immediate Value
        Y : out STD_LOGIC_VECTOR (3 downto 0));
end component;


component Ins_Decoder
    Port ( Instruction : in STD_LOGIC_VECTOR (11 downto 0);
        Reg_en : out STD_LOGIC_VECTOR (2 downto 0);
        Load_sel : out STD_LOGIC;
        Imm_Val : out STD_LOGIC_VECTOR (3 downto 0);
        Reg_Sel_A : out STD_LOGIC_VECTOR (2 downto 0);
        Reg_Sel_B : out STD_LOGIC_VECTOR (2 downto 0);
        Add_Sub_Sel : out STD_LOGIC;
        Jump_add : out STD_LOGIC_VECTOR (2 downto 0);
        Jump_flag : out STD_LOGIC;
        Reg_Chk_Jmp : in STD_LOGIC_VECTOR (3 downto 0));
end component;


component Program_ROM_LUT_8_12
    Port ( address : in STD_LOGIC_VECTOR (2 downto 0);
        data : out STD_LOGIC_VECTOR (11 downto 0));
end component;


component PC_3Bit
```

```
  Port ( D : in STD_LOGIC_VECTOR (2 downto 0);
    En : in STD_LOGIC;
    Reset : in STD_LOGIC;
    Clk : in STD_LOGIC;
    Q : out STD_LOGIC_VECTOR (2 downto 0));
end component;


component RCA_3
  Port ( A : in STD_LOGIC_VECTOR (2 downto 0);
    C_in : in STD_LOGIC;
    S : out STD_LOGIC_VECTOR (2 downto 0);
    C_out : out STD_LOGIC);
end component;


component Mux_2_to_1_3Bit
  Port ( S : in STD_LOGIC;
    D0 : in STD_LOGIC_VECTOR (2 downto 0);
    D1 : in STD_LOGIC_VECTOR (2 downto 0);
    Y : out STD_LOGIC_VECTOR (2 downto 0));
end component;



--Declaring signals========================================

signal Imm_Val_Ins_to_Mux, RCA_out, Reg_Chk_Jmp, Data_Mux_to_Reg_Bank,
Data_bus0, Data_bus1, Data_bus2, Data_bus3, Data_bus4, Data_bus5, Data_bus6,
Data_bus7: STD_LOGIC_VECTOR(3 downto 0);
signal Ins_Bus : STD_LOGIC_VECTOR(11 downto 0);
signal Reg_bus, Mem_Bus, Reg_SelA , Reg_SelB, Count_Mux_to_PC,
Mem_RCA3_to_Mux, Jmp_Add: STD_LOGIC_VECTOR (2 downto 0);
signal Add_Sub_Sel, Load_Select_Ins_to_Mux, C_out_RCA_3, Jmp_Flag: STD_LOGIC;
signal Slow_Clk_Signal: STD_LOGIC;



begin
```

```
--Port
mapping=============================================================

SevenSeg_LUT_16_7_0 : SevenSeg_LUT_16_7
port map(
address => Data_bus7,
data => Seven_seg_out);

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


Reg_Bank_0 : Reg_Bank
port map (
     Reg_en => Reg_bus,
     Clk => Slow_Clk_Signal,
     Reset => Reset,
     Input_D => Data_Mux_to_Reg_Bank,
     Q0 => Data_bus0,
     Q1 => Data_bus1,
     Q2 => Data_bus2,
     Q3 => Data_bus3,
     Q4 => Data_bus4,
     Q5 => Data_bus5,
     Q6 => Data_bus6,
     Q7 => Data_bus7 );

AU_0 : AU
port map (
   D0 => Data_bus0,
   D1 => Data_bus1,
   D2 => Data_bus2,
   D3 => Data_bus3,
   D4 => Data_bus4,
   D5 => Data_bus5,
   D6 => Data_bus6,
   D7 => Data_bus7,
   Reg_SelA => Reg_SelA,
   Reg_SelB => Reg_SelB,
        Add_Sub_Sel => Add_Sub_Sel,
        Reg_Chk_Jmp => Reg_Chk_Jmp,
        RCA_out => RCA_out,
        Overflow => Overflow,
        Zero => Zero
    );

    Mux_2_to_1_4Bit_0 : Mux_2_to_1_4Bit
    port map (
        S => Load_Select_Ins_to_Mux,
        RCA_out => RCA_out,
        Imm_Val => Imm_Val_Ins_to_Mux,
        Y => Data_Mux_to_Reg_Bank);

    Program_ROM_LUT_8_12_0 : Program_ROM_LUT_8_12
    port map (
        address => Mem_Bus,
        data => Ins_Bus);

    PC_3Bit_0 : PC_3Bit
    port map (
        D => Count_Mux_to_PC,
        EN => '1',
        Reset => Reset,
        Clk => Slow_Clk_Signal,
        Q => Mem_Bus);

    RCA_3_0 : RCA_3
    port map (
        A => Mem_Bus,
        C_in => '0',
        S => Mem_RCA3_to_Mux,
        C_out => C_out_RCA_3);

    Mux_2_to_1_3Bit_0 : Mux_2_to_1_3Bit
    port map (
        S => Jmp_Flag,
        D0 => Mem_RCA3_to_Mux,
        D1 => Jmp_Add,
        Y => Count_Mux_to_PC);
```

```
Ins_Decoder_0 : Ins_Decoder
port map (
   Instruction => Ins_Bus,
   Reg_en => Reg_bus,
   Load_sel => Load_Select_Ins_to_Mux,
   Imm_Val => Imm_Val_Ins_to_Mux,
   Reg_Sel_A => Reg_SelA,
   Reg_Sel_B => Reg_SelB,
   Add_Sub_Sel => Add_Sub_Sel,
   Jump_add => Jmp_Add,
   Jump_flag => Jmp_Flag,
   Reg_Chk_Jmp => Reg_Chk_Jmp);


R7 <= Data_bus7;

an <= "1110";

end Behavioral;
```

**SIMULATION SOURCE CODES**

- **4-bit Add/Subtract Unit**

```
----------------------------------------------------------------------------------
-- Company: 
-- Engineer: 
-- 
-- Create Date: 07/20/2022 12:24:52 AM
-- Design Name: 
-- Module Name: TB_RCA - Behavioral
-- Project Name: 
-- Target Devices: 
-- Tool Versions: 
-- Description: 
-- 
-- Dependencies: 
-- 
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
-- 
----------------------------------------------------------------------------------


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_4_TB is
--  Port ( );
end RCA_4_TB;
```

```
architecture Behavioral of RCA_4_TB is

   COMPONENT RCA_4
   Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
          B : in STD_LOGIC_VECTOR (3 downto 0);
          C_in : in std_logic;
          S : out STD_LOGIC_VECTOR (3 downto 0);
          C_out : out STD_LOGIC;
          Zero : out STD_LOGIC);
   END COMPONENT;

   signal A :std_logic_vector(3 downto 0);
   signal B :  STD_LOGIC_VECTOR (3 downto 0);
   signal S :  STD_LOGIC_VECTOR (3 downto 0);
   signal c_in,c_out,zero :std_logic;
begin

UUT: RCA_4 PORT MAP(

   A=>A,
   B=>B,
   C_in => c_in,
   S=>S,
   C_out => c_out,
   Zero => zero
);

   process
   begin

       C_in<='1';

      --1110 - 0100 ANSWER=1010
      A<="1110";
      B<="0100";

      WAIT FOR 100 ns; -- after 100 ns change inputs

      --0000 - 0111
      --answer = 1001
      A<="0000";
```

```
        B<="0111";

        WAIT FOR 100 ns; -- after 100 ns change inputs
        --0110 + 0110
        --answer = 1100
        C_in<='0';
        A<="0110";
        B<="0110";

        WAIT FOR 100 ns; -- after 100 ns change inputs
        --0011 + 1011 ANSWER=1100
        A<="0011";
        B<="1011";

        WAIT FOR 100 ns; -- after 100 ns change inputs
        --0011 + 0001 ANSWE=0100
        A<="0011";
        B<="0001";

        WAIT FOR 100 ns; -- after 100 ns change inputs
        --0011 + 0001 ANSWE=0100
        A<="0000";
        B<="0000";

        WAIT FOR 100 ns; -- after 100 ns change inputs
        --0000 + 1011 ANSWER=1011
        A<="0000";
        B<="1011";
        WAIT; -- will wait forever

    end process;

end Behavioral;
```

- **3-bit Adder**

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 06/02/2022 11:07:00 AM
-- Design Name:
-- Module Name: TB_4_RCA - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


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_TB is
--  Port ( );
end RCA_3_TB;

architecture Behavioral of RCA_3_TB is
```

```
   COMPONENT RCA_3
      Port ( A : in STD_LOGIC_VECTOR (2 downto 0);
           C_in : in STD_LOGIC;
           S : out STD_LOGIC_VECTOR (2 downto 0);
           C_out : out STD_LOGIC);
   END COMPONENT;

   signal A,S : STD_LOGIC_VECTOR (2 downto 0);
   SIGNAL c_in,c_out:std_logic;

begin

UUT: RCA_3 PORT MAP(
   A=>A,
   C_in => c_in,
   C_out => c_out,
   S=>S);

   process
   begin

   --A = 000
   --answer = 001
      C_in<='0';
      A<="000";

      WAIT FOR 100 ns; -- after 100 ns change inputs

      --A = 010
      --answer = 011

     A<="001";

      WAIT FOR 100 ns; -- after 100 ns change inputs

      --A = 110
      --answer = 111

      A<="010";
      WAIT FOR 100NS;
      A<="011";
```

```
        WAIT FOR 100NS;
        A<="100";
        WAIT FOR 100NS;
        A<="101";
        WAIT FOR 100NS;
        A<="110";
        WAIT FOR 100NS;
        A<="111";




        WAIT; -- will wait forever

    end process;


end Behavioral;
```

- **3-bit Program Counter**

```
----------------------------------------------------------------------------------
-- Company: 
-- Engineer: 
-- 
-- Create Date: 07/19/2022 03:17:53 PM
-- Design Name: 
-- Module Name: TB_PC_3Bit - Behavioral
-- Project Name: 
-- Target Devices: 
-- Tool Versions: 
-- Description: 
-- 
-- Dependencies: 
-- 
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
-- 
----------------------------------------------------------------------------------
```

```
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_3Bit_TB is
--  Port ( );
end PC_3Bit_TB;

architecture Behavioral of PC_3Bit_TB is
component PC_3Bit

Port ( D : in STD_LOGIC_VECTOR (2 downto 0);
        En : in STD_LOGIC;
        Reset : in STD_LOGIC;
        Clk : in STD_LOGIC;
        Q : out STD_LOGIC_VECTOR (2 downto 0));
end component;
signal D,Q:std_logic_vector(2 downto 0):="000";
signal En,Reset,Clk:std_logic:='0';

begin
UUT:PC_3Bit
port map(
D=>D,
En=>En,
Reset=>Reset,
Clk=>Clk,
Q=>Q
);
process
begin
      Clk <=  NOT(Clk);
```

```
    wait for 20ns;
end process;
process
BEGIN
  Reset<='1';
  wait for 40ns;
  Reset<='0';
  WAIT FOR 40NS;
  D<="000";
  WAIT FOR 40NS;

  D<="001";
  wait for 40ns;
  D<="010";
  wait for 40ns;
  D<="011";
  wait for 40ns;
  D<="100";
  wait for 40ns;
  D<="101";
  wait for 40ns;
  D<="110";
  wait for 40ns;
  D<="111";
  wait ;


end process;


end Behavioral;
```

- **2-way 3-bit Multiplexer**

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 07/20/2022 02:54:29 AM
-- Design Name:
-- Module Name: Mux_2_to_1_4Bit_TB - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


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_to_1_3Bit_TB is
--  Port ( );
end Mux_2_to_1_3Bit_TB;

architecture Behavioral of Mux_2_to_1_3Bit_TB is
```

```
component Mux_2_to_1_3Bit
   Port ( S : in STD_LOGIC;
      D0 : in STD_LOGIC_VECTOR (2 downto 0); --RCA
      D1 : in STD_LOGIC_VECTOR (2 downto 0); --Immediate Value
      Y : out STD_LOGIC_VECTOR (2 downto 0));
end component;

signal d0, d1, y : STD_LOGIC_VECTOR (2 downto 0);
signal s : STD_LOGIC;

begin

UUT: Mux_2_to_1_3Bit
port map (
S => s,
D0 => d0,
D1 => d1,
Y => y
);


process
begin

--d0 = 1, d1 = 2
d0 <= "001";
d1 <= "010";

--change s
s <= '0';
wait for 100ns;

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

--d0 = 6, d1 = 7
d0 <= "110";
d1 <= "111";

--change s
s <= '0';
```

```
wait for 100ns;

s <= '1';
wait;

end process;

end Behavioral;
```

- **2-way 4-bit Multiplexer**

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 07/20/2022 02:54:29 AM
-- Design Name:
-- Module Name: Mux_2_to_1_4Bit_TB - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


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_to_1_4Bit_TB is
--  Port ( );
end Mux_2_to_1_4Bit_TB;

architecture Behavioral of Mux_2_to_1_4Bit_TB is

component Mux_2_to_1_4Bit
   Port ( S : in STD_LOGIC;
      RCA_out : in STD_LOGIC_VECTOR (3 downto 0); --RCA
      Imm_Val : in STD_LOGIC_VECTOR (3 downto 0); --Immediate Value
      Y : out STD_LOGIC_VECTOR (3 downto 0));
end component;

signal rca_out, imm_val, y : STD_LOGIC_VECTOR (3 downto 0);
signal s : STD_LOGIC;

begin

UUT: Mux_2_to_1_4Bit
port map (
S => s,
RCA_out => rca_out,
Imm_Val => imm_val,
Y => y
);


process
begin

--rca_out = 1, imm_val = 2
rca_out <= "0001";
imm_val <= "0010";

--change s
s <= '0';
wait for 100ns;
```

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

--rca_out = 10(a), imm_val = 11(b)
rca_out <= "1010";
imm_val <= "1011";

--change s
s <= '0';
wait for 100ns;

s <= '1';
wait;

end process;

end Behavioral;
```

- **8-way 4-bit Multiplexer**

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 07/20/2022 02:31:50 AM
-- Design Name:
-- Module Name: Mux_8way_4bit_TB - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------
```

```
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_8way_4bit_TB is
--  Port ( );
end Mux_8way_4bit_TB;

architecture Behavioral of Mux_8way_4bit_TB is

component Mux_8way_4bit
   Port (
      S : in STD_LOGIC_VECTOR (2 downto 0);
      D0 : in STD_LOGIC_VECTOR (3 downto 0);
      D1 : in STD_LOGIC_VECTOR (3 downto 0);
      D2 : in STD_LOGIC_VECTOR (3 downto 0);
      D3 : in STD_LOGIC_VECTOR (3 downto 0);
      D4 : in STD_LOGIC_VECTOR (3 downto 0);
      D5 : in STD_LOGIC_VECTOR (3 downto 0);
      D6 : in STD_LOGIC_VECTOR (3 downto 0);
      D7 : in STD_LOGIC_VECTOR (3 downto 0);
      Y : out STD_LOGIC_VECTOR (3 downto 0));
end component;

signal d0, d1, d2, d3, d4, d5, d6, d7, y : STD_LOGIC_VECTOR (3 downto 0);
signal s : STD_LOGIC_VECTOR (2 downto 0);

begin

UUT: Mux_8way_4bit
port map(
S => s,
D0 => d0,
D1 => d1,
D2 => d2,
D3 => d3,
D4 => d4,
D5 => d5,
D6 => d6,
D7 => d7,
Y => y
);

process
   begin

   --Here let's set D0 - D7 to the corresponding decimal value * 2
   -- Example : D0 -> 0, D1 -> 2, D2 -> 4...
   d0 <= "0000";
   d1 <= "0010";
   d2 <= "0100";
   d3 <= "0110";
   d4 <= "1000";
   d5 <= "1010";
   d6 <= "1100";
   d7 <= "1110";

   --now let's change S and check the output
   s <= "000";
   wait for 100 ns;

   s <= "001";
   wait for 100 ns;

   s <= "010";
   wait for 100 ns;

   s <= "011";
   wait for 100 ns;

   s <= "100";
   wait for 100 ns;
```

```
  s <= "101";
  wait for 100 ns;

  s <= "110";
  wait for 100 ns;

  s <= "111";
  wait;

end process;

end Behavioral;
```

- **Register Bank**

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 07/21/2022 08:06:12 AM
-- Design Name:
-- Module Name: TB_Reg_Bank - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


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_Bank_TB is
--  Port ( );
end Reg_Bank_TB;

architecture Behavioral of Reg_Bank_TB is
component Reg_Bank
   Port ( Reg_en : in STD_LOGIC_VECTOR (2 downto 0);
          Clk : in STD_LOGIC;
          Reset : in STD_LOGIC;
          Input_D : in STD_LOGIC_VECTOR (3 downto 0);
          Q0 : out STD_LOGIC_VECTOR (3 downto 0);
          Q1 : out STD_LOGIC_VECTOR (3 downto 0);
          Q2 : out STD_LOGIC_VECTOR (3 downto 0);
          Q3 : out STD_LOGIC_VECTOR (3 downto 0);
          Q4 : out STD_LOGIC_VECTOR (3 downto 0);
          Q5 : out STD_LOGIC_VECTOR (3 downto 0);
          Q6 : out STD_LOGIC_VECTOR (3 downto 0);
          Q7 : out STD_LOGIC_VECTOR (3 downto 0));
end component;

signal Reg_en : STD_LOGIC_VECTOR (2 downto 0);
signal Clk,Reset : STD_LOGIC := '0';
signal Input_D ,Q0,Q1,Q2,Q3,Q4,Q5,Q6,Q7 : STD_LOGIC_VECTOR (3 downto 0);

begin

UUT : Reg_Bank
   PORT MAP(
      Reg_en => Reg_en,
      Clk=> Clk,
      Reset=> Reset,
      Input_D=> Input_D,
      Q0=>Q0,
      Q1=>Q1,
      Q2=>Q2,
      Q3=>Q3,
      Q4=>Q4,
      Q5=>Q5,
      Q6=>Q6,
      Q7=>Q7 );


      process
         begin
         wait for 10ns;
         Clk <= NOT(Clk);
      end process;

      process
         begin

         Reset <= '1';
         wait for 50ns;

         Reset<='0';
         Reg_en<="001";
         Input_D<="0001";
         wait for 105ns;

         Reg_en<="010";
         Input_D<="1001";
         wait for 102ns;

         Reset <= '1';
         wait for 50ns;

         Reset<='0';
         Reg_en<="111";
         Input_D<="0001";
         wait for 105ns;

         Reg_en<="001";
         Input_D<="1001";
         wait for 102ns;
```

```
        Reg_en<="101";
        Input_D<="1101";
        wait for 103ns;

        Reg_en<="110";
        Input_D<="0011";

        wait;


    end process;

end Behavioral;
```

- **Program ROM**

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 07/19/2022 11:03:30 AM
-- Design Name:
-- Module Name: Program_ROM_LUT_8_12 - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


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_ROM_LUT_8_12 is
   Port ( address : in STD_LOGIC_VECTOR (2 downto 0);
  data : out STD_LOGIC_VECTOR (11 downto 0));
end Program_ROM_LUT_8_12;




architecture Behavioral of Program_ROM_LUT_8_12 is

type rom_type is array (0 to 7) of std_logic_vector(11 downto 0);

signal program_ROM : rom_type := (

"100010000011", -- 0
"100100000001", --1
"010100000000", --2
"001110010000", --3
"000010100000", --4
"110010000111", --5
"110000000011", --6
"110000000111" --7
);

begin

data <= program_ROM(to_integer(unsigned(address)));

end Behavioral;
```

- **Instruction Decoder**

```
----------------------------------------------------------------------------------
-- Company: 
-- Engineer: 
-- 
-- Create Date: 07/10/2022 10:41:37 PM
-- Design Name: 
-- Module Name: Ins_Dec_TB - Behavioral
-- Project Name: 
-- Target Devices: 
-- Tool Versions: 
-- Description: 
-- 
-- Dependencies: 
-- 
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
-- 
----------------------------------------------------------------------------------


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_Dec_TB is
--  Port ( );
end Ins_Dec_TB;

architecture Behavioral of Ins_Dec_TB is
```

```
    COMPONENT Ins_Decoder
    Port ( Instruction : in STD_LOGIC_VECTOR (11 downto 0);
          Reg_Chk_Jmp : in STD_LOGIC_VECTOR (3 downto 0);
          Reg_en : out STD_LOGIC_VECTOR (2 downto 0);
          Load_sel : out STD_LOGIC;
          Imm_Val : out STD_LOGIC_VECTOR (3 downto 0);
          Reg_Sel_A : out STD_LOGIC_VECTOR (2 downto 0);
          Reg_Sel_B : out STD_LOGIC_VECTOR (2 downto 0);
          Add_Sub_Sel : out STD_LOGIC;
          Jump_add : out STD_LOGIC_VECTOR (2 downto 0);
          Jump_flag : out STD_LOGIC);
    END COMPONENT;


signal Instruction : STD_LOGIC_VECTOR (11 downto 0);
signal Reg_en : STD_LOGIC_VECTOR (2 downto 0);
signal Load_sel : STD_LOGIC;
signal Imm_Val : STD_LOGIC_VECTOR (3 downto 0);
signal Reg_Sel_A : STD_LOGIC_VECTOR (2 downto 0);
signal Reg_Sel_B : STD_LOGIC_VECTOR (2 downto 0);
signal Add_Sub_Sel : STD_LOGIC;
signal Jump_add : STD_LOGIC_VECTOR (2 downto 0);
signal Jump_flag : STD_LOGIC;
signal Reg_Chk_Jmp : STD_LOGIC_VECTOR (3 downto 0);



begin

UUT: Ins_Decoder PORT MAP(
    Instruction => Instruction,
    Reg_Chk_Jmp => Reg_Chk_Jmp,
    Reg_en => Reg_en,
    Load_sel => Load_sel ,
    Imm_Val => Imm_Val ,
    Reg_Sel_A => Reg_Sel_A ,
    Reg_Sel_B => Reg_Sel_B ,
    Add_Sub_Sel => Add_Sub_Sel ,
    Jump_add => Jump_add ,
    Jump_flag => Jump_flag );
```

```
process
begin

   Reg_Chk_Jmp <= "0000";
   Instruction <= "100110000110";   --MOVI 3,6
   WAIT FOR 100 ns; -- after 100 ns change inputs

   Instruction <= "101000001001";   --MOVI 4,9
   WAIT FOR 100 ns; -- after 100 ns change inputs

  -- Reg_Chk_Jmp <= "0000";
   Instruction <= "000111000000";   --ADD 3,4
   WAIT FOR 100 ns; -- after 100 ns change inputs

   Instruction <= "011000000000";   --NEG 4
   WAIT FOR 100 ns; -- after 100 ns change inputs

   Instruction <= "111110000001";   --JZR 7,1
   Reg_Chk_Jmp <= "0000";
   WAIT FOR 100 ns; -- after 100 ns change inputs

   Instruction <= "110110000010";   --JZR 3,2
   Reg_Chk_Jmp <= "0001";
   WAIT ; -- after 100 ns change inputs



end process;

end Behavioral;
```

- **Slow Clock**

```
----------------------------------------------------------------------------------
-- Company: 
-- Engineer: 
-- 
-- Create Date: 07/13/2022 09:36:49 PM
-- Design Name: 
-- Module Name: Slow_Clk_TB - Behavioral
-- Project Name: 
-- Target Devices: 
-- Tool Versions: 
-- Description: 
-- 
-- Dependencies: 
-- 
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
-- 
----------------------------------------------------------------------------------


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 Slow_Clk_TB is
--  Port ( );
end Slow_Clk_TB;

architecture Behavioral of Slow_Clk_TB is
```

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

signal Clk, Clk_out : STD_LOGIC := '0';


begin

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

process
   begin
   wait for 10ns;
   Clk <= NOT(Clk);

end process;

end Behavioral;
```

- **Nano Processor**

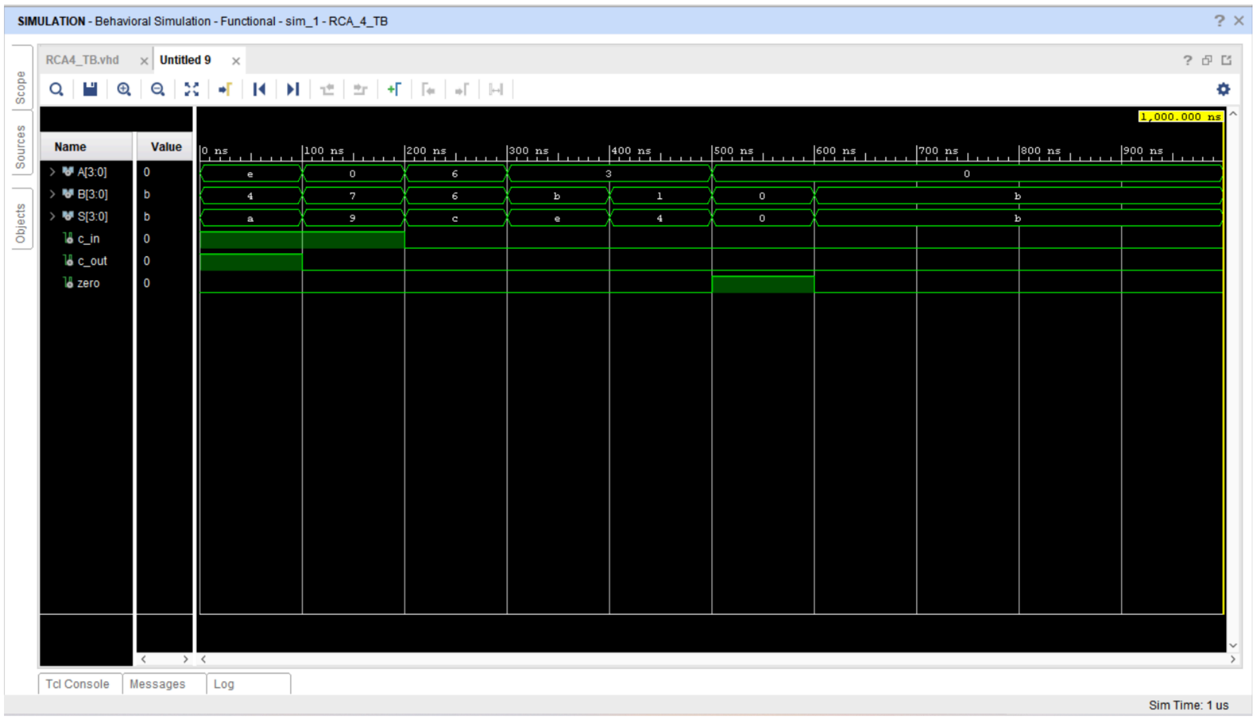
```
----------------------------------------------------------------------------------
-- Company: 
-- Engineer: 
-- 
-- Create Date: 07/11/2022 10:31:30 PM
-- Design Name: 
-- Module Name: Nanoprocessor_TB - Behavioral
-- Project Name: 
-- Target Devices: 
-- Tool Versions: 
-- Description: 
-- 
-- Dependencies: 
-- 
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
-- 
----------------------------------------------------------------------------------


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 Nanoprocessor_TB is
--  Port ( );
end Nanoprocessor_TB;

architecture Behavioral of Nanoprocessor_TB is
```

```
component Nanoprocessor
   Port ( Clk : in STD_LOGIC;
      Reset : in STD_LOGIC;
      Overflow : out STD_LOGIC;
      Zero : out STD_LOGIC;
      R7 : out STD_LOGIC_VECTOR (3 downto 0));
end component;

signal Reset, Overflow, Zero : STD_LOGIC;
signal Clk : STD_LOGIC := '0';
signal R7 : STD_LOGIC_VECTOR (3 downto 0);

begin

UUT: Nanoprocessor PORT MAP(
   Clk => Clk,
   Reset => Reset,
   Overflow => Overflow,
   Zero => Zero,
   R7 => R7
)

process
   begin
   wait for 1ns;
   Clk <= NOT(Clk);
end process;

process
begin

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

Reset <= '0';
wait;

end process;
```

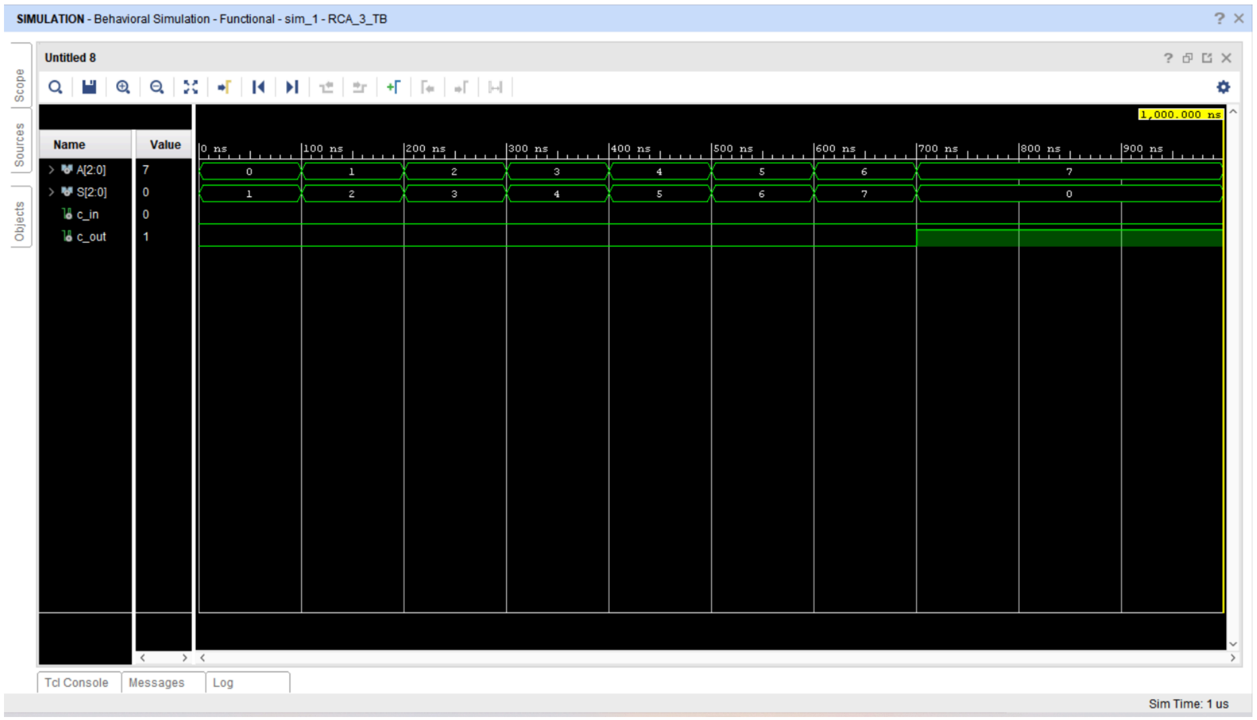
```
end Behavioral;
```

## TIMING DIAGRAMS

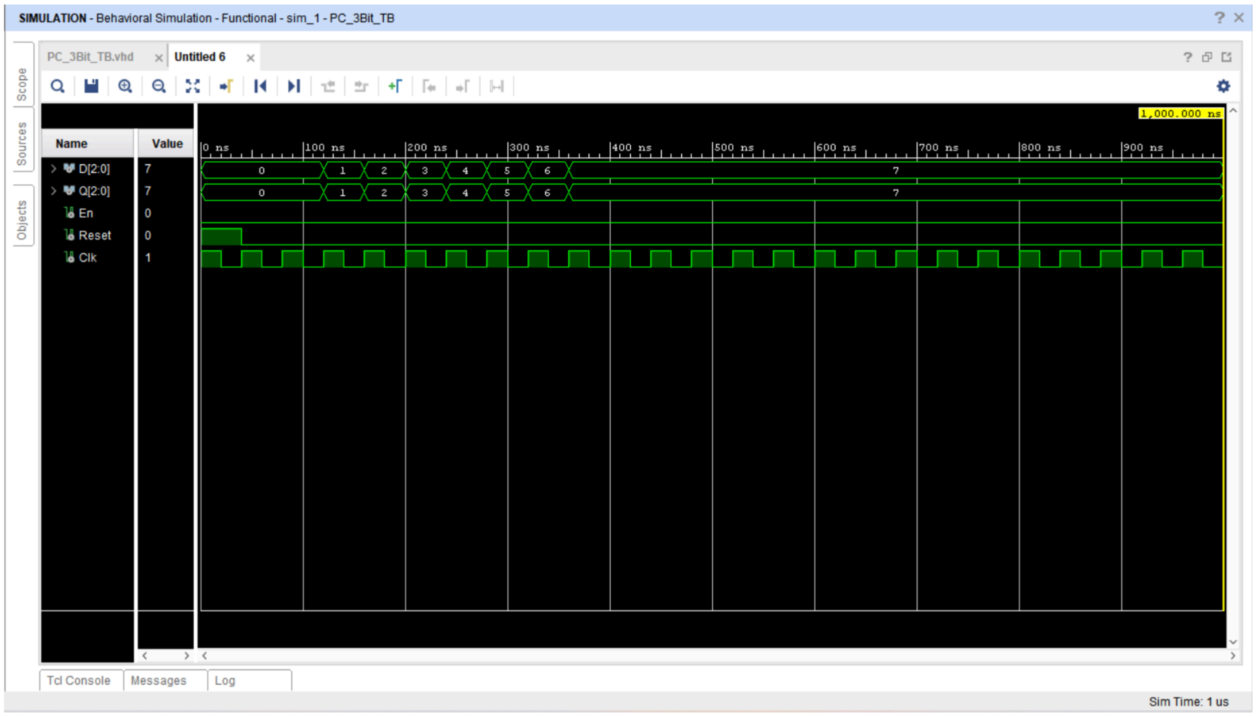
- **4-bit Add/Subtract Unit**



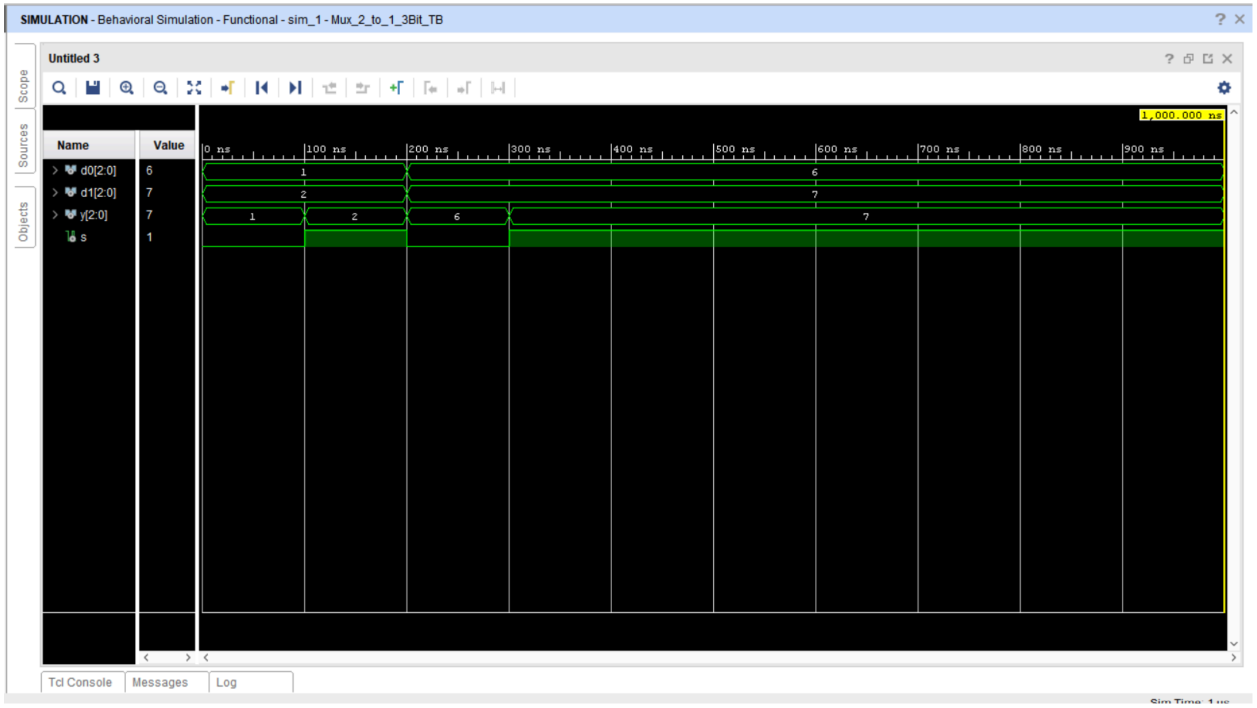
- **3-bit Adder**



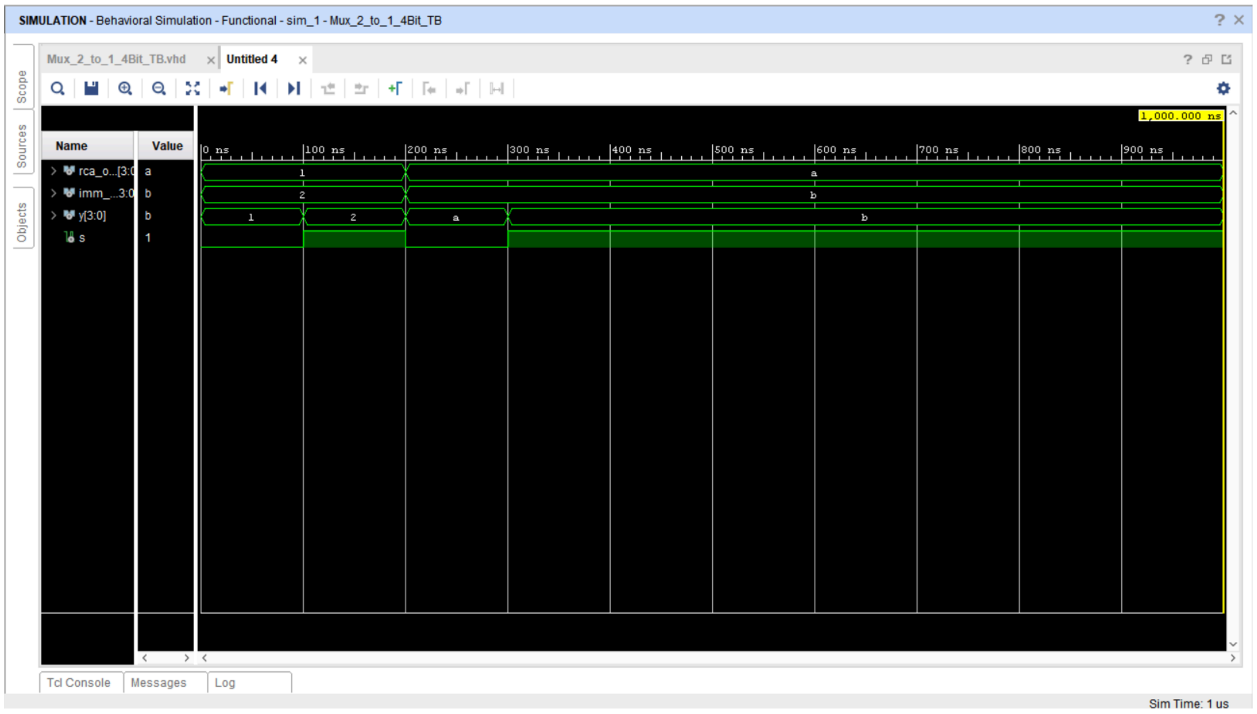
- **3-bit Program Counter**



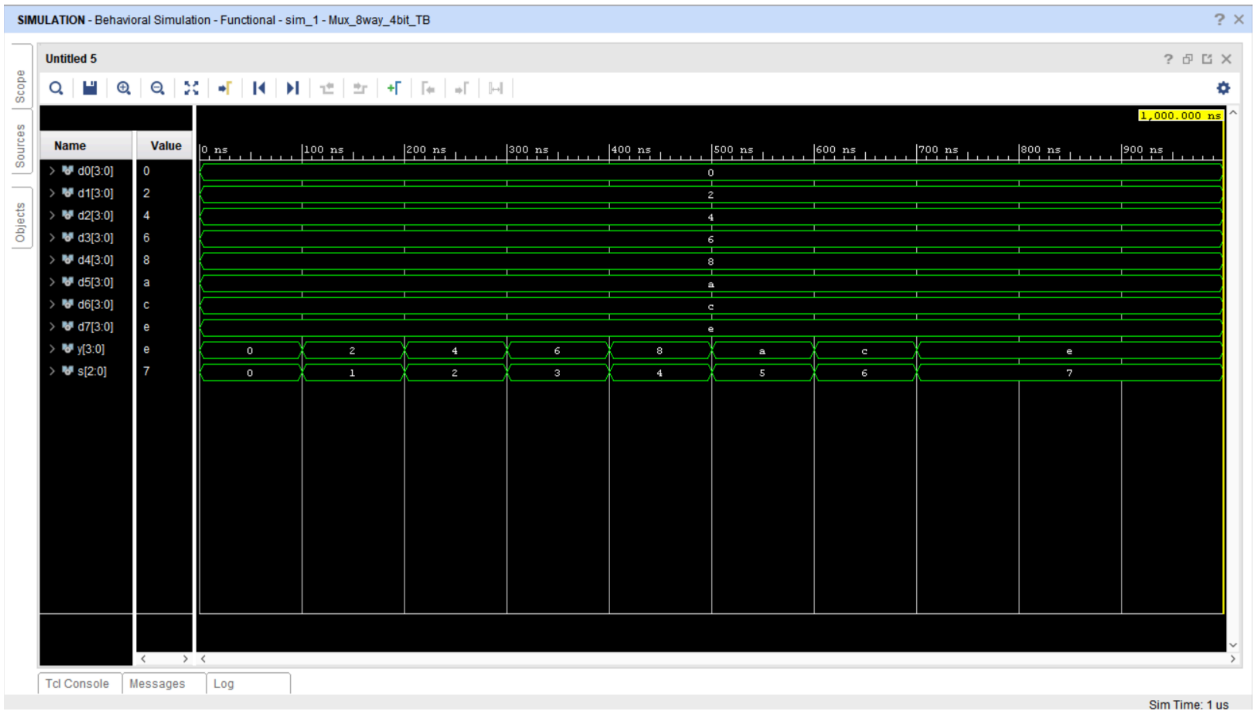
- **2-way 3-bit Multiplexer**



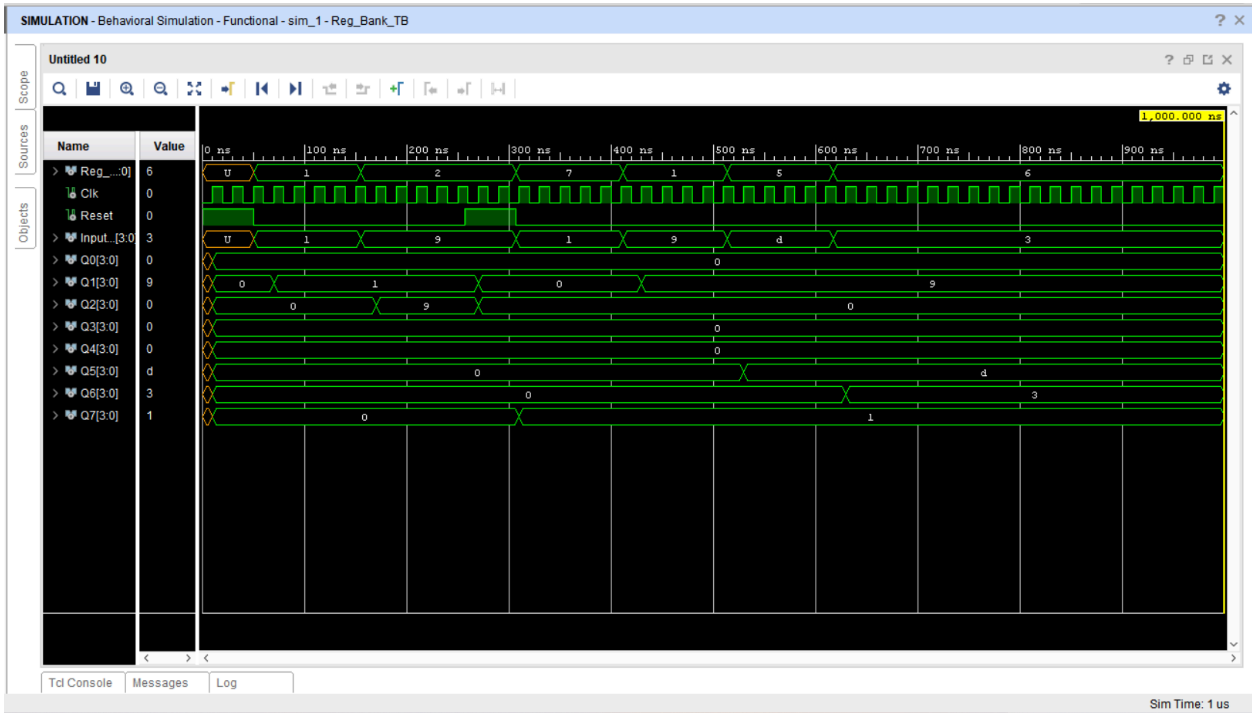
- **2-way 4-bit Multiplexer**



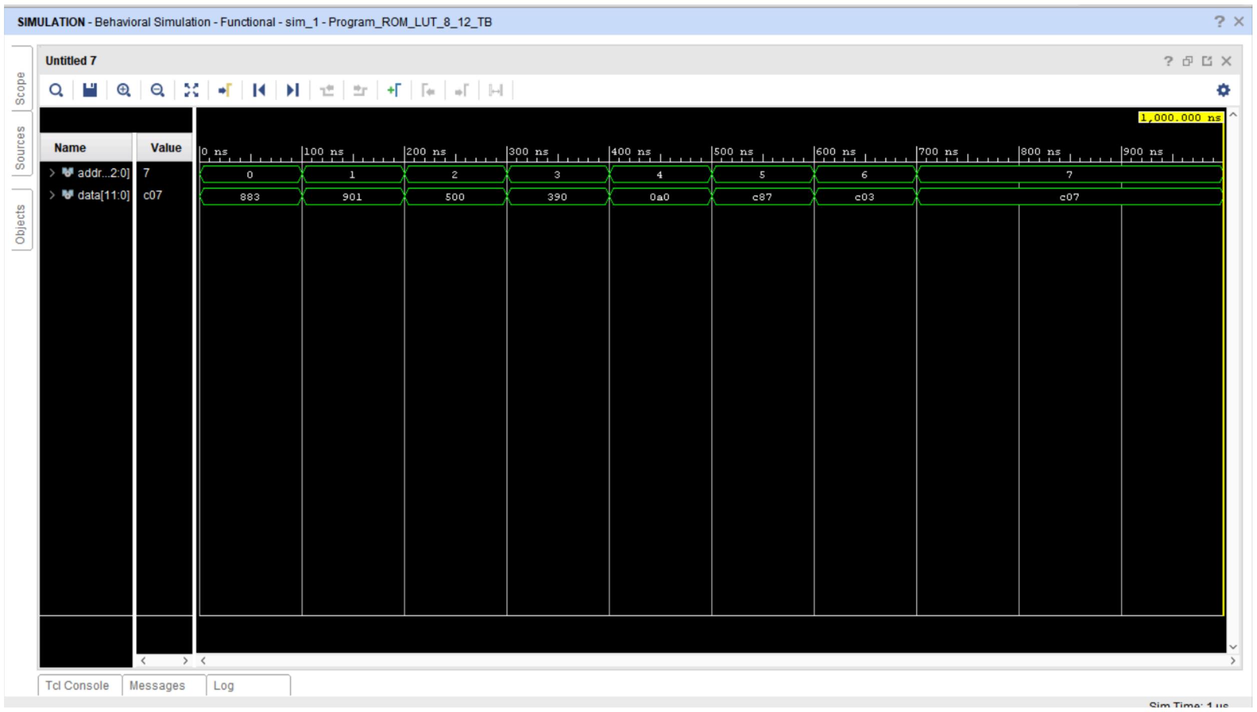
- **8-way 4-bit Multiplexer**



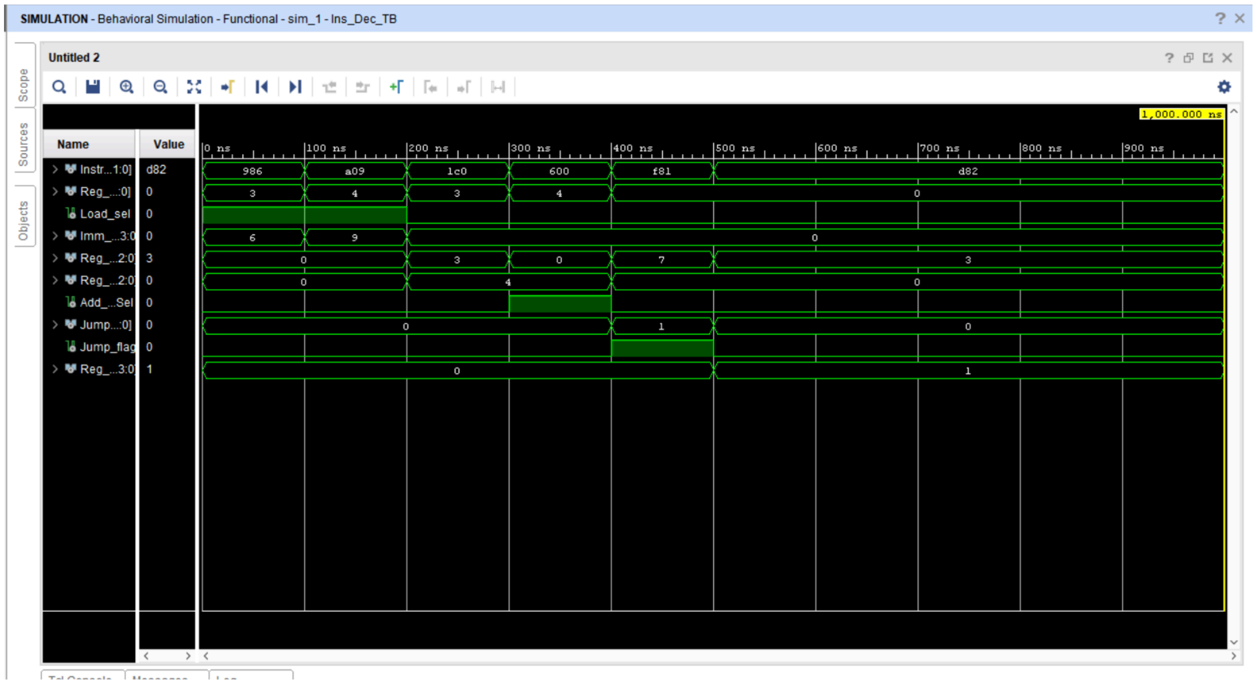
- **Register Bank**



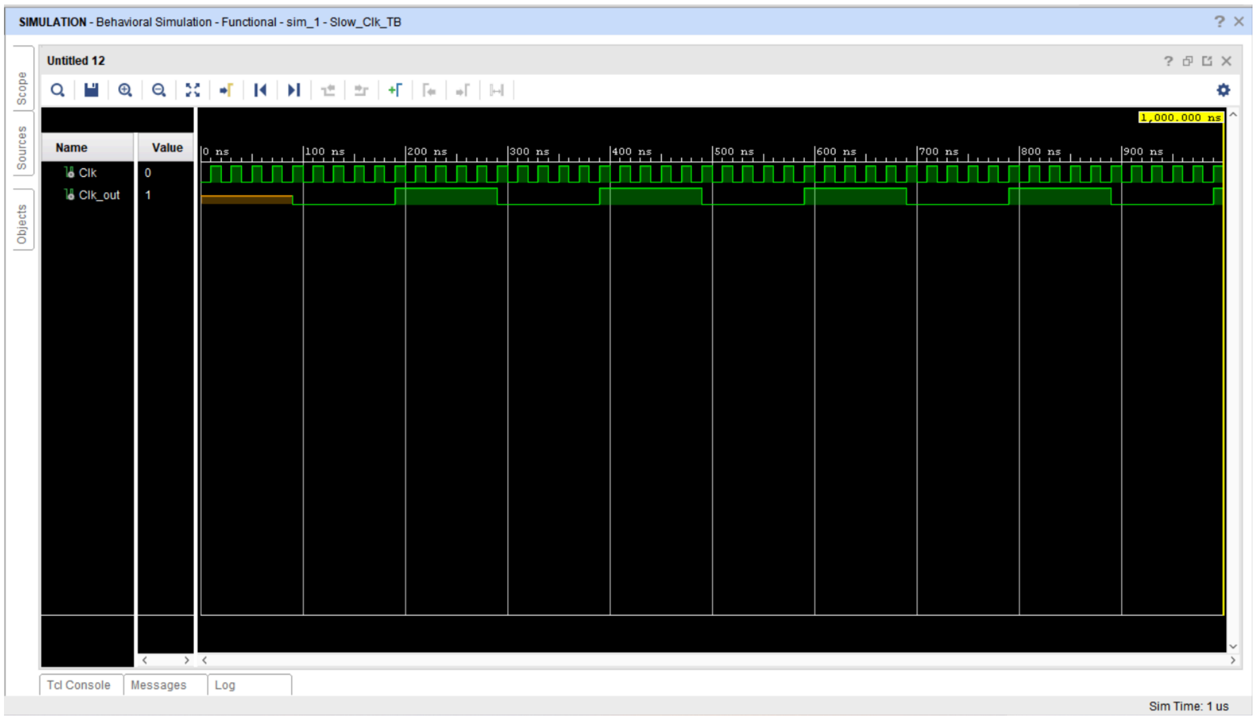
- **Program ROM**



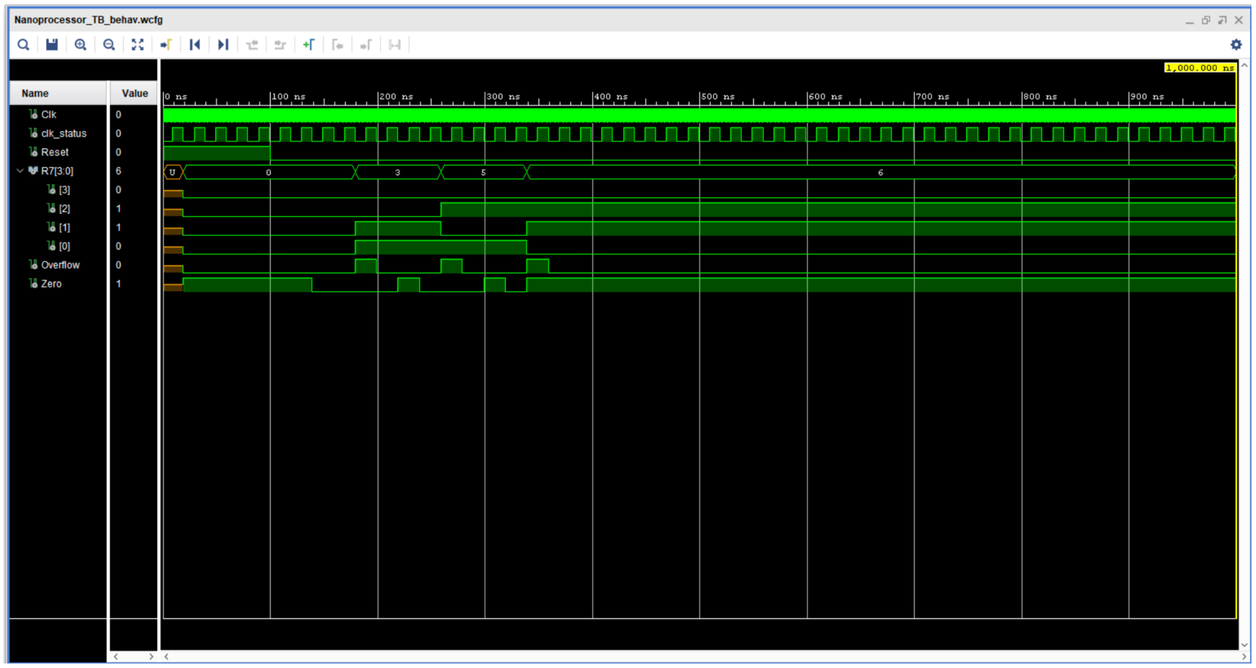
- **Instruction Decoder**



- **Slow Clock**



- **Nano Processor**



## ASSEMBLY PROGRAM

```
1) MOVI R1,3      ;R1 <= 3
2) MOVI R2,1      ;R2 <= 1
3) NEG R2         ;R2 <= -R2
4) ADD R7,R1      ;R7 <= R7 + R1
5) ADD R1,R2      ;R1 <= R1 + R2
6) JZR R1,7       ;If (R1==0), Jump to line 7
7) JZR R0,3       ;If (R0==0), Jump to line 3
8) JZR R0,7       ;If (R==0), Jump to line 7
```

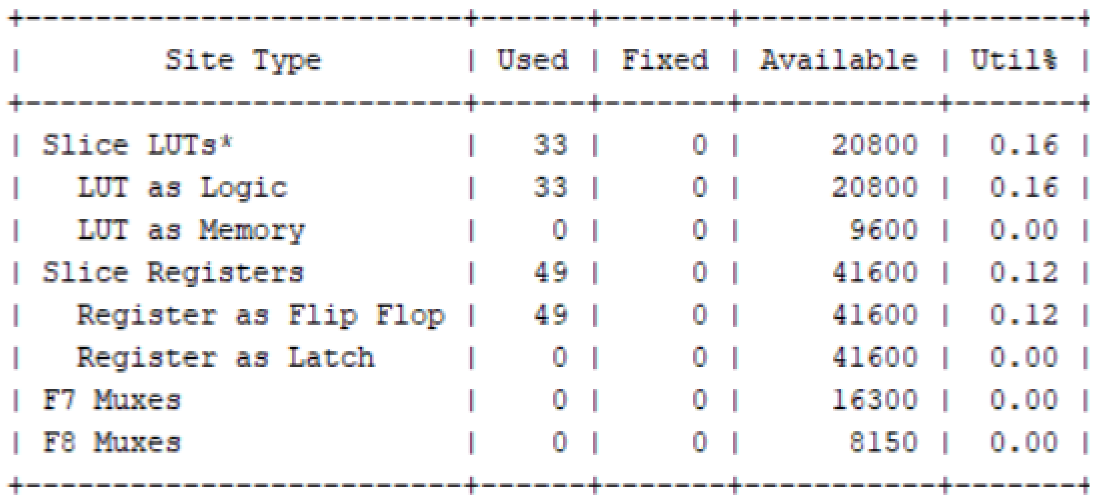
## MACHINE CODE REPRESENTAION

```
1) 1 0 0 0 1 0 0 0 0 0 1 1
2) 1 0 0 1 0 0 0 0 0 0 0 1
3) 0 1 0 1 0 0 0 0 0 0 0 0
4) 0 0 1 1 1 0 0 1 0 0 0 0
5) 0 0 0 0 1 0 1 0 0 0 0 0
6) 1 1 0 0 1 0 0 0 0 1 1 1
7) 1 1 0 0 0 0 0 0 0 0 1 1
8) 1 1 0 0 0 0 0 0 0 1 1 1
```

## CONCLUSIONS

- This project gave us vast knowledge about how a microprocessor is internally structured, how multiple components work together inside a processor, how instructions are decoded and how they are stored.
- While building the Nano Processor, we recalled some previous lab activities. It made our work easier rather than building them again.
- As our microprocessor is only 4-bit wide, we cannot work on a larger problem. So, we can calculate only the total of all integers between 1 and 3.
- As the microprocessor only understands machine language, we need to hardcode assembly instructions as binary values.
- The use of buses simplifies the design rather than running so many wires around.
- Results are generated by giving clock pulses.
- By simulating each component separately, we confirmed the proper work of the sub-components before building the microprocessor.
- This project also enhanced our team working skills. We discussed the ideas of all team members on designing each component of the Nano Processor. Then we ended up creating the nano processor through a discussion method.

## FINAL LUT & FF COUNT



```
+-------------------------+------+-------+-----------+-------+
|        Site Type        | Used | Fixed | Available | Util% |
+-------------------------+------+-------+-----------+-------+
| Slice LUTs*             |   33 |     0 |     20800 |  0.16 |
|   LUT as Logic          |   33 |     0 |     20800 |  0.16 |
|   LUT as Memory         |    0 |     0 |      9600 |  0.00 |
| Slice Registers         |   49 |     0 |     41600 |  0.12 |
|   Register as Flip Flop |   49 |     0 |     41600 |  0.12 |
|   Register as Latch     |    0 |     0 |     41600 |  0.00 |
| F7 Muxes                |    0 |     0 |     16300 |  0.00 |
| F8 Muxes                |    0 |     0 |      8150 |  0.00 |
+-------------------------+------+-------+-----------+-------+
```

## HOW WE OPTIMIZED

- We used "if" statements (which are already optimized) in the instruction decoder so that the LUT count, and the FF count get lower.
- We first tried using two half adders for implementing the full adder. But after several experiments, we discovered that developing the full adder from the scratch with the help of logic gates, is more optimized, which made a lesser utilization of resources.
- We tried using more abstraction levels and divided the code into more components. (Ex: Making an AU as an abstracted level of RCA's and multiplexers.)

# <u>ENHANCED NANO-PROCESSOR</u>

After developing and optimizing the nano-processor according to the given lab task we further improved our nano-processor design as follows.

**<u>Improvement 1: Improving the capacities of registers, instructions, and other components</u>**

We improved the capacities of the components in the nano-processor as follows.

| Improvement | Old value | New Value |
|---|---|---|
| **Number of registers in the registry bank** | 8 | 16 |
| **Size of each register** | 4 bits | 8 bits |
| **Number of instructions that can be stored** | 8 | 16 |
| **Size of each instruction** | 12 bits | 16 bits |
| **Types of instructions we can use** | 4 | 16 |

* According to these requirements, the sizes of the buses and other relevant components also have increased.

Ex : Data Bus – from 4-bit to 8-bit

Address Bus – from 3-bit to 4-bit

RCA – from 4-bit to 8-bit

**<u>Improvement 2 : Adding an instruction to the required position</u>**

- ✓ The most significant improvement in our Enhanced Nano-Processor is giving the chance for the user to **upload** his/her **own set of instructions** into the processor **without plugging the board in to a computer**.
- ✓ For this we have replaced the Read-only memory (which was used in the basic nano-processor) with a rewritable memory (RAM).
- ✓ Also, for a better user experience, we have used the inbuilt Seven Segment display of BASYS 3 board as an output device.

Given below are the steps of how the user can input an instruction to the RAM.

- Step 1:

  Press the Right button .

  *Now the display is changed to "Adr", as the user should next input the address of the instruction to be entered.*

- Step 2:

  Set the address as a binary number using the last 4 of 16 inbuilt switches and press the Right Button.

  *Now the display is changed to "Ins", as the user should next input the instruction as a 16-bit value.*

- Step 3:

  Set the instruction as a binary number using the 16 inbuilt switches and press the Right Button.

  *Now the display is changed to "Done" indicating the instruction was successfully stored in the memory.*

- Step 4:

  Press the Right Button again to exit the “Edit Mode”.

  *Now the display is changed to “CSE” indicating that the processor is now in the “Performance Mode”.*

- Step 5:

  Now press the Up Button to run the program you just entered !!!

**Note :**

- There is a “**RAM-RESET**” button in the nano-processor, which is connected to the **Center Button** of the BASYS 3 Board.
  When the RAM-RESET button is pressed, the instruction memory is loaded with the **default set of instructions**, which can perform the lab task given in Lab 9.

## ❖ UPGRADED INSTRUCTION SET

When we upgraded the size of the instructions, we came up with a new instruction format.

| Instruction | Description | Format (16-bit instruction) |
|---|---|---|
| Mov R, D | Move the value D into register R | 0010 RRRR DDDDDDDD |
| Add R1, R2 | Add value in R2 to the value in R1 | 0000 $R_1R_1R_1R_1$ 0000 $R_2R_2R_2R_2$ |
| Neg R | Negate the value in R | 0001 RRRR 0000 0000 |
| JZR R, D | If value in R = 0, then PC = D | 0011 RRRR 0000 DDDD |

Because we have now increased the number of types of instructions that can be used from 4 to 16, the processor can further be improved to support instructions like Multiplication, Division, Modulus, Increment, Decrement, Copy, Wait, etc.

## CONTRIBUTION OF EACH TEAM MEMBER

| Name | Designed parts of the Nano Processor | No. of hours spent |
|---|---|---|
| E. A. C. Chandeepa | We created each component of the Nano processor through a discussion.<br>• 4-bit Add/Subtract Unit<br>• 3-bit Adder<br>• 3-bit Program Counter<br>• 2-way 3-bit Multiplexer<br>• 2-way 4-bit Multiplexer<br>• 8-way 4-bit Multiplexer<br>• Register Bank<br>• Program ROM<br>• Instruction Decoder | 7 Hours<br>Extra 15 hours for the improvements (For increasing the size of improved components and rewritable instructions memory) |
| M. M. H. A. Premarathna | | 7 Hours<br>Extra 8 hours for the discussion on improvements |
| W. A. S. P. Wijesuriya | | 7 Hours<br>Extra 10 hours for the improvements<br>(For creating the seven-segment display handler and the enclosure nano processor) |
| P. S. N. Pathirathne | | 7 Hours<br>Extra 8 hours for the discussion on improvements |
| L. T. Darshani | | 7 Hours<br>Extra 8 hours for the discussion on improvements |