LAB REPORT

Lab 9-10 – Nano processor Design Competition

- **TEAM MEMBERS**

NAME	INDEX NUMBER
Nayanathara P.M.C.	210417X
Pathirana D.P.	210449V

- **LAB TASK**

In this lab, we were assigned to design a very 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
    - Instruction Decoder
  - 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 input, output pins, testing on BASYS3 board and verifying the functionality of the nano processor.

- **ASSEMBLY PROGRAM**

1. MOVI R1, 1
2. MOVI R2, 2
3. MOVI R3, 3
4. MOVI R7, 0
5. ADD R7, R1
6. ADD R7,R2
7. ADD R7,R3
8. JZR R0,7

- **MACHINE CODE REPRESENTATION**

1. 100010000001 --MOVI R1,1
2. 100100000010 --MOVI R2,2
3. 100110000011 --MOVI R3,3
4. 101110000000 --MOVI R7,0
5. 001110010000 --ADD R7,R1
6. 001110100000 --ADD R7,R2
7. 001110110000 --ADD R7,R3
8. 110000000111 --JZR R0,7

- **VHDL CODES**

DESIGN SOURCES

- 4-bit Add/Subtract unit

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 08.06.2023 11:57:12
-- Design Name:
-- Module Name: Add_Sub_Unit - 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 Add_Sub is
    Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
           B : in STD_LOGIC_VECTOR (3 downto 0);
           M : in STD_LOGIC;
           S : out STD_LOGIC_VECTOR (3 downto 0);
           carry : out STD_LOGIC;
           Zero : out STD_LOGIC);
end Add_Sub;

architecture Behavioral of Add_Sub is
```

```
COMPONENT RCA4
Port ( A0 : in STD_LOGIC;
      A1 : in STD_LOGIC;
      A2 : in STD_LOGIC;
      A3 : in STD_LOGIC;
      B0 : in STD_LOGIC;
      B1 : in STD_LOGIC;
      B2 : in STD_LOGIC;
      B3 : in STD_LOGIC;
      C_in : in STD_LOGIC;
      S0 : out STD_LOGIC;
      S1 : out STD_LOGIC;
      S2 : out STD_LOGIC;
      S3 : out STD_LOGIC;
      c_out : out STD_LOGIC);
END COMPONENT;
signal b0,b1,b2,b3,zero0,s0,s1,s2,s3 : std_logic;
begin
RCA0 : RCA4
PORT MAP(
   A0 => A(0),
   A1 => A(1),
   A2 => A(2),
   A3 => A(3),
   B0 => b0,
   B1 => b1,
   B2 => b2,
   B3 => b3,
   C_in => M,
   S0 => s0,
   S1 => s1,
   S2 => s2,
   S3 => s3,
   c_out => carry
);

b0 <= B(0) XOR M;
b1 <= B(1) XOR M;
b2 <= B(2) XOR M;
b3 <= B(3) XOR M;
Zero <= NOT(s0 OR s1 OR s2 OR s3);
S(0) <= s0;
S(1) <= s1;
S(2) <= s2;
S(3) <= s3;


end Behavioral;
```

- 3-bit Adder

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 31.05.2023 15:05:41
-- Design Name:
-- Module Name: three_bit_adder - 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 three_bit_adder is
   Port ( A : in STD_LOGIC_VECTOR (2 downto 0);
          B : in STD_LOGIC_VECTOR (2 downto 0);
          Cin : in STD_LOGIC;
          Sum : out STD_LOGIC_VECTOR (2 downto 0);
          Cout : out STD_LOGIC);
end three_bit_adder;

architecture Behavioral of three_bit_adder 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 : std_logic;

begin
  FA0: FA
  port map(
  A => A(0),
  B => '1',
  C_in => Cin,
  S => Sum(0),
  C_out => FA0_C);

  FA1: FA
  port map(
  A => A(1),
  B => '0',
  C_in => FA0_C,
  S => Sum(1),
  C_out => FA1_C);

  FA2: FA
  port map(
  A => A(2),
  B => '0',
  C_in => FA1_C,
  S => Sum(2),
  C_out => Cout);

end architecture Behavioral;
```

- 3-bit Program Counter

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 01.06.2023 08:04:16
-- Design Name:
-- Module Name: PC - 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 is
   Port ( Clk : in STD_LOGIC;
          Reset : in STD_LOGIC;
          Count_in : in STD_LOGIC_VECTOR (2 downto 0);
          Count_out : out STD_LOGIC_VECTOR (2 downto 0));
end PC;

architecture Behavioral of PC is
   component D_FF is
      port(
          D : in STD_LOGIC;
          Res : in STD_LOGIC;
          Clk : in STD_LOGIC;
          Q : out STD_LOGIC;
          Qbar : out STD_LOGIC);
   end component;

--signal D : std_logic_vector (2 downto 0);

begin

D_FF0 : D_FF
   port map(
      D => Count_in(0),
      Clk => Clk,
      Res => Reset,
      Q => Count_out(0));
```

```
D_FF1 : D_FF
   port map(
      D => Count_in(1),
      Clk => Clk,
      Res => Reset,
      Q => Count_out(1));

D_FF2 : D_FF
   port map(
       D => Count_in(2),
       Clk => Clk,
       Res => Reset,
       Q => Count_out(2));
end Behavioral;
```

- 2-way 3-bit Multiplexer

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 31.05.2023 04:58:17
-- Design Name:
-- Module Name: Mux_2way_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 Mux_2way_3bit is
    Port ( A : in STD_LOGIC_VECTOR(2 downto 0);
           B : in STD_LOGIC_VECTOR(2 downto 0);
           S : in STD_LOGIC;
           Q : out STD_LOGIC_VECTOR(2 downto 0));
end Mux_2way_3bit;

architecture Behavioral of Mux_2way_3bit is

begin
    process(A,B)
    begin
        if S = '0' then
            Q <= A;
        else
            Q <= B;
        end if;
   end process;

end Behavioral;
```

- 2-way 4-bit Multiplexer

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 31.05.2023 14:45:41
-- Design Name:
-- Module Name: Mux_2way_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_2way_4bit is
   Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
          B : in STD_LOGIC_VECTOR (3 downto 0);
          S : in STD_LOGIC;
          Q : out STD_LOGIC_VECTOR (3 downto 0));
end Mux_2way_4bit;

architecture Behavioral of Mux_2way_4bit is

begin
   process(A,B)
   begin
      if S = '1' then
         Q <= A;
      else
         Q <= B;
      end if;
  end process;

end Behavioral;
```

- 8-way 4-bit Multiplexer

```
----------------------------------------------------------------------------------
-- Company: 
-- Engineer: 
-- 
-- Create Date: 31.05.2023 15:35:09
-- 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 ( D0,D1,D2,D3,D4,D5,D6,D7 : in STD_LOGIC_VECTOR (3 downto 0);
           S : in STD_LOGIC_VECTOR (2 downto 0);
           Q : out STD_LOGIC_VECTOR (3 downto 0));
end Mux_8way_4bit;

architecture Behavioral of Mux_8way_4bit is
begin
    process(S,D0,D1,D2,D3,D4,D5,D6,D7)
    begin
        case S is
            when "000" =>
              Q <= D0;
            when "001" =>
              Q <= D1;
            when "010" =>
              Q <= D2;
            when "011" =>
              Q <= D3;
            when "100" =>
              Q <= D4;
            when "101" =>
              Q <= D5;
            when "110" =>
              Q <= D6;
            when "111" =>
              Q <= D7;
            when others =>
              Q <= (others => '0');  --Default case
        end case;
    end process;
end Behavioral;
```

- Register Bank

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 01.06.2023 16:44:00
-- 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_Enable : in STD_LOGIC_VECTOR(2 downto 0);
        Reset : in STD_LOGIC;
        D : in STD_LOGIC_VECTOR (3 downto 0);
        Clk : in STD_LOGIC;
        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 Slow_Clk
   port( Clk_in : in STD_LOGIC;
       Clk_out : out STD_LOGIC);
END COMPONENT;

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);
--signal slow_clock : STD_LOGIC;

begin
Decoder_3_to_8_0 : Decoder_3_to_8
    PORT MAP(
        I => Reg_Enable,
        EN => '1',
        Y => Y);
R0 : Reg
    PORT MAP(
        D => "0000", --Hard code R0 to 0
        EN => Y(0),
        Clk => Clk,
        Reset =>Reset,
        Q => Q0);

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

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

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

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

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

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

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

- Program ROM

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 02.06.2023 01:04:32
-- Design Name:
-- Module Name: Prog_ROM - 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 Prog_ROM is
   Port ( Memo_select : in STD_LOGIC_VECTOR (2 downto 0);
          Ins_bus : out STD_LOGIC_VECTOR (11 downto 0));
end Prog_ROM;

architecture Behavioral of Prog_ROM is
type rom_type is array (0 to 7) of std_logic_vector(11 downto 0);
      signal program_ROM : rom_type := (
        "100010000001",  --MOVI R1,1
         "100100000010", --MOVI R2,2
         "100110000011", --MOVI R3,3
         "101110000000", --MOVI R7,0
         "001110010000", --ADD R7,R1
         "001110100000", --ADD R7,R2
         "001110110000", --ADD R7,R3
         "110000000111"  --JZR R0,7


      );
begin
   Ins_bus <= program_ROM(to_integer(unsigned(Memo_select)));

end Behavioral;
```

- Instruction Decoder

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 02.06.2023 18:28:52
-- 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 ( Data : in STD_LOGIC_VECTOR (11 downto 0);
           Reg_check : in STD_LOGIC_VECTOR (3 downto 0);
           Reg_enable : out STD_LOGIC_VECTOR (2 downto 0);
           Immediate_value : out STD_LOGIC_VECTOR (3 downto 0);
           Load_select : out STD_LOGIC;
           Reg_select0 : out STD_LOGIC_VECTOR (2 downto 0);
           Reg_select1 : out STD_LOGIC_VECTOR (2 downto 0);
           Add_sub_select : out STD_LOGIC;
           Jump_flag : out STD_LOGIC;
           Jump_address : out STD_LOGIC_VECTOR (2 downto 0));
end Ins_decoder;
```

```
architecture Behavioral of Ins_decoder is

signal OPCODE : STD_LOGIC_VECTOR(1 downto 0);


begin
process(Data, Reg_check)
 begin
 Reg_enable <= "000";
 Load_select <= '0';
 Immediate_value <= "0000";
 Reg_select0 <= "000";
 Reg_select1 <= "000";
 Add_sub_select <= '0';
 Jump_address <= "000";
 Jump_flag <= '0';

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

 If (Data(11)='0' and Data(10)='0') then
 --Add instruction
 --Reg En
 Reg_enable<= Data (9 downto 7);

 --Load select
 Load_select <= '0';

 --Reg Sel 1
 Reg_select0 <= Data (9 downto 7);

 --Reg Sel 2
 Reg_select1 <= Data (6 downto 4);

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

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

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

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

 --Reg En
 Reg_enable <= Data(9 downto 7);
 --Load select
 Load_select <= '0';
 --Reg Sel 1
 Reg_select0 <= "000";

 --Reg Sel 2
 Reg_select1 <= Data (9 downto 7);

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

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

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

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

 --Reg En
 Reg_enable <= Data(9 downto 7);

 --Load select
 Load_select <= '1';

 --Immediate Value
 Immediate_value <= Data(3 downto 0);

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

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

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

 --Reg En
 Reg_enable <= "000";

 --Reg Sel 1
 Reg_select0 <= Data (9 downto 7);

 --if Reg_Chk_Jmp = 0 then jump flag = 1
 if (Reg_check = "0000") then
 Jump_flag <= '1';
 Jump_address <= Data(2 downto 0);
 else
 Jump_flag <= '0';
 end if;
 end if;
end process;
end Behavioral;
```

- Slow Clock

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 08.04.2023 08:53:10
-- 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
    --For 100MHz input clock this generates 1Hz clock
    process (Clk_in) begin
```

```
         if (rising_edge(Clk_in)) then
            count <= count+1;           --increment counter
           --if (count=50000000) then
            if (count=4) then    --count 50M pulses (1/2 of period)
               clk_status <= not clk_status;  --invert clock status
               Clk_out <= clk_status;
               count <=1;              --reset counter
            end if;
         end if;
      end process;

   end Behavioral;
```

- Nano Processor

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 03.06.2023 09:51:58
-- 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;
        zero : out STD_LOGIC;
        Carry : out STD_LOGIC;
        R7 : out STD_LOGIC_VECTOR(3 downto 0);
        Display_7_Seg : out STD_LOGIC_VECTOR(6 downto 0);
        Anode_activate : out STD_LOGIC_VECTOR (3 downto 0));
end NanoProcessor;

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

component PC
  Port(
     Clk : in STD_LOGIC;
     Reset : in STD_LOGIC;
     Count_in : in STD_LOGIC_VECTOR (2 downto 0);
     Count_out : out STD_LOGIC_VECTOR (2 downto 0));
end component;

component Prog_ROM
  Port ( Memo_select : in STD_LOGIC_VECTOR (2 downto 0);
        Ins_bus : out STD_LOGIC_VECTOR (11 downto 0));
end component;

component  Ins_decoder
  Port ( Data : in STD_LOGIC_VECTOR (11 downto 0);
        Reg_check : in STD_LOGIC_VECTOR (3 downto 0);
        Reg_enable : out STD_LOGIC_VECTOR (2 downto 0);
        Immediate_value : out STD_LOGIC_VECTOR (3 downto 0);
        Load_select : out STD_LOGIC;
        Reg_select0 : out STD_LOGIC_VECTOR (2 downto 0);
        Reg_select1 : out STD_LOGIC_VECTOR (2 downto 0);
        Add_sub_select : out STD_LOGIC;
        Jump_flag : out STD_LOGIC;
        Jump_address : out STD_LOGIC_VECTOR (2 downto 0));
end component;

component three_bit_adder
  Port ( A : in STD_LOGIC_VECTOR (2 downto 0);
        B : in STD_LOGIC_VECTOR (2 downto 0);
        Cin : in STD_LOGIC;
        Sum : out STD_LOGIC_VECTOR (2 downto 0);
        Cout : out STD_LOGIC);
end component;
```

```
component Mux_2way_3bit
   Port ( A : in STD_LOGIC_VECTOR(2 downto 0);
          B : in STD_LOGIC_VECTOR(2 downto 0);
          S : in STD_LOGIC;
          Q : out STD_LOGIC_VECTOR(2 downto 0));
end component;

component Mux_2way_4bit
   Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
          B : in STD_LOGIC_VECTOR (3 downto 0);
          S : in STD_LOGIC;
          Q : out STD_LOGIC_VECTOR (3 downto 0));
end component;

component Add_Sub
   Port ( A : in STD_LOGIC_VECTOR(3 downto 0);
          B : in STD_LOGIC_VECTOR(3 downto 0);
          M  : in STD_LOGIC;
          S : out STD_LOGIC_VECTOR(3 downto 0);
          carry : out STD_LOGIC;
          zero : out STD_LOGIC);
end component;

component Reg_Bank
   Port ( Reg_Enable : in STD_LOGIC_VECTOR(2 downto 0);
          Reset : in STD_LOGIC;
          D : in STD_LOGIC_VECTOR (3 downto 0);
          Clk : in STD_LOGIC;
          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 LUT_16_7
   port(address : in STD_LOGIC_VECTOR (3 downto 0);
        data : out STD_LOGIC_VECTOR (6 downto 0));
end component;

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

```
end component;

signal signalIn, signalOut, reg_enable, jadd, sum, muxIn0, muxIn1 : STD_LOGIC_VECTOR(2
downto 0);
signal instruction : STD_LOGIC_VECTOR(11 downto 0);
signal reg_check, value, muxIn2, reg_value, q0, q1, q2, q3, q4, q5, q6, q7, out0, out1 :
STD_LOGIC_VECTOR(3 downto 0);
signal loadsel, sel, jflag, Carry0, zero0, clk_out, clock : STD_LOGIC;
signal DisplaySeg : STD_LOGIC_VECTOR(6 downto 0);

begin
slow_clock0 : Slow_Clk
   port map(
      Clk_in => Clk,
      Clk_out => clk_out);

Program_Counter : PC
   port map(
      Clk => clk_out,
      Reset => Reset,
      Count_in => signalIn,
      Count_out => signalOut);

Prog_ROM0 : Prog_ROM
   port map(
      Memo_select => signalOut,
      Ins_bus => instruction);

Instruction_Decoder : Ins_decoder
   port map(
      Data => instruction,
      Reg_check => out0,
      Reg_enable => reg_enable,
      Immediate_value => value,
      Load_select => loadsel,
      Reg_select0 => muxIn0,
      Reg_select1 => muxIn1,
      Add_sub_select => sel,
      Jump_flag => jflag,
      Jump_address => jadd);

Mux_2way_4bit0 : Mux_2way_4bit
   port map(
      A => value,
      B => muxIn2,
      S => loadsel,
      Q => reg_value);
```

```
Reg_Bank0 : Reg_Bank
  port map(
    Reg_Enable => reg_enable,
    Reset => Reset,
    D => reg_value,
    Clk => clk_out,
    Q0 => q0,
    Q1 => q1,
    Q2 => q2,
    Q3 => q3,
    Q4 => q4,
    Q5 => q5,
    Q6 => q6,
    Q7 => q7);

Mux_8way_4bit0 : Mux_8way_4bit
  port map(
    D0 => q0,
    D1 => q1,
    D2 => q2,
    D3 => q3,
    D4 => q4,
    D5 => q5,
    D6 => q6,
    D7 => q7,
    S => muxIn0,
    Q => out0);

Mux_8way_4bit1 : Mux_8way_4bit
  port map(
    D0 => q0,
    D1 => q1,
    D2 => q2,
    D3 => q3,
    D4 => q4,
    D5 => q5,
    D6 => q6,
    D7 => q7,
    S => muxIn1,
    Q => out1);

Add_Sub0 : Add_Sub
  port map(
    A(0) => out0(0),
    A(1) => out0(1),
    A(2) => out0(2),
    A(3) => out0(3),
    B(0) => out1(0),
      B(1) => out1(1),
      B(2) => out1(2),
      B(3) => out1(3),
      M => sel,
      S(0) => muxIn2(0),
      S(1) => muxIn2(1),
      S(2) => muxIn2(2),
      S(3) => muxIn2(3),
      zero => zero0,
      carry => Carry0);

three_bit_adder0 : three_bit_adder
   port map(
      A => signalOut,
      B => "001",
      Cin => '0',
      Sum => sum);

Mux_2way_3bit0 : Mux_2way_3bit
   port map(
      A => sum,
      B => jadd,
      S => jflag,
      Q => signalIn);

LUT_16_70 : LUT_16_7
   port map(
      address => q7,
      data => DisplaySeg);


R7 <= q7;
zero <= zero0;
Carry <= Carry0;
Display_7_Seg <= DisplaySeg;
Anode_activate <= "1110";

end Behavioral;
```

SIMULATION SOURCE

- 4-bit Add/Subtract Unit

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 08.06.2023 12:39:27
-- Design Name:
-- Module Name: Sim_Add_Sub_Unit - 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 Sim_Add_Sub is
--  Port ( );
end Sim_Add_Sub;

architecture Behavioral of Sim_Add_Sub is
 COMPONENT Add_Sub
  Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
           B : in STD_LOGIC_VECTOR (3 downto 0);
           M : in STD_LOGIC;
           S : out STD_LOGIC_VECTOR (3 downto 0);
           carry : out STD_LOGIC;
```

```
        Zero : out STD_LOGIC);
END COMPONENT;


   signal A_tb, B_tb : STD_LOGIC_VECTOR(3 downto 0);
   signal M_tb : STD_LOGIC;
   signal S_tb : STD_LOGIC_VECTOR(3 downto 0);
   signal C_out_tb, Zero_tb : STD_LOGIC;

begin

   uut: Add_Sub
      Port Map (
         A => A_tb,
         B => B_tb,
         M => M_tb,
         S => S_tb,
         carry => C_out_tb,
         Zero => Zero_tb
      );

   stimulus: process
   begin
      A_tb <= "0000";
      B_tb <= "0000";
      M_tb <= '0';
      wait for 10 ns;

      A_tb <= "0011";
      B_tb <= "0011";
      M_tb <= '0';
      wait for 10 ns;

      A_tb <= "0000";
      B_tb <= "0111";
      M_tb <= '1';
      wait for 10 ns;



      wait;
   end process;

end Behavioral;
```

- 3- bit Adder

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 31.05.2023 14:59:44
-- Design Name:
-- Module Name: Sim_three_bit_adder - 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 Sim_three_bit_adder is
--  Port ( );
end Sim_three_bit_adder;

architecture Behavioral of Sim_three_bit_adder is
 COMPONENT three_bit_adder
     PORT(Cin: IN STD_LOGIC;
         A,B : in STD_LOGIC_VECTOR (2 downto 0);
         Cout : OUT STD_LOGIC;
         Sum : out STD_LOGIC_VECTOR (2 downto 0));
 END COMPONENT;

Signal Cin,Cout : STD_LOGIC;
Signal A,Sum : STD_LOGIC_VECTOR(2 downto 0);
Signal B: STD_LOGIC_VECTOR(2 downto 0) := "001";

begin
UUT: three_bit_adder PORT MAP(
        A => A,
        B => B,
        Cin => Cin,
        Sum => Sum,
        Cout => Cout
);
PROCESS
       BEGIN

           A(0) <= '0';
           A(1) <= '0';
           A(2) <= '0';

           Cin <= '0';  --- for the adder
           WAIT FOR 100 ns;

           A(0) <= '1';
           A(1) <= '0';
           A(2) <= '1';


           WAIT FOR 100 ns;

           A(0) <= '1';
           A(1) <= '1';
           A(2) <= '1';


           WAIT;
    END PROCESS;

end Behavioral;
```

- 3-bit Program Counter

```
----------------------------------------------------------------------------------
-- Company: 
-- Engineer: 
-- 
-- Create Date: 06.06.2023 09:29:33
-- Design Name: 
-- Module Name: Sim_PC - 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 Sim_PC is
--  Port ( );
end Sim_PC;

architecture Behavioral of Sim_PC is
   component PC is
      Port(
          Clk : in STD_LOGIC;
          Reset : in STD_LOGIC;
          Count_in : in STD_LOGIC_VECTOR(2 downto 0);
          Count_out : out STD_LOGIC_VECTOR(2 downto 0)
      );
   end component;
```

```
    signal Clk : STD_LOGIC := '0';
    signal Reset : STD_LOGIC := '0';
    signal Count_in : STD_LOGIC_VECTOR(2 downto 0) := (others => '0');
    signal Count_out : STD_LOGIC_VECTOR(2 downto 0);

    constant CLK_PERIOD : time := 10 ns;

begin
    UUT: PC
    port map (
        Clk => Clk,
        Reset => Reset,
        Count_in => Count_in,
        Count_out => Count_out
    );

    Clk_process: process
    begin
        while now < 1000 ns loop  -- Run the simulation for 1000 ns
            Clk <= '0';
            wait for CLK_PERIOD / 2;
            Clk <= '1';
            wait for CLK_PERIOD / 2;
        end loop;
        wait;
    end process;

    Stimulus_process: process
    begin
        Reset <= '1';  -- Assert reset
        wait for 20 ns;
        Reset <= '0';  -- Deassert reset

        Count_in <= "000";
        wait for CLK_PERIOD;
        Count_in <= "001";
        wait for CLK_PERIOD;
        Count_in <= "010";
        wait for CLK_PERIOD;
        Count_in <= "011";
        wait for CLK_PERIOD;
        Reset <= '1';
        Count_in <= "100";
        wait for CLK_PERIOD;
        Count_in <= "101";
        wait for CLK_PERIOD;
        Count_in <= "110";
```

```
      wait for CLK_PERIOD;
      Count_in <= "111";
      wait for CLK_PERIOD;

      wait;
   end process;

end Behavioral;
```

- 2-way 3-bit Multiplexer

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 31.05.2023 14:10:09
-- Design Name:
-- Module Name: Sim_2way_3bit_MUX - 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 Sim_2way_3bit_MUX is
--  Port ( );
end Sim_2way_3bit_MUX;
```

```
architecture Behavioral of Sim_2way_3bit_MUX is
COMPONENT Mux_2way_3bit
   PORT(
      A : in STD_LOGIC_VECTOR(2 downto 0);
      B : in STD_LOGIC_VECTOR(2 downto 0);
      S : in STD_LOGIC;
      Q : out STD_LOGIC_VECTOR(2 downto 0));
END COMPONENT;

SIGNAL A : std_logic_vector (2 downto 0);
SIGNAL B : std_logic_vector (2 downto 0);
SIGNAL S : std_logic;
SIGNAL Q : std_logic_vector (2 downto 0);

begin
UUT: Mux_2way_3bit PORT MAP(
   A=>A,
   B=>B,
   S=>S,
   Q=>Q);

 process
 begin
   S<='1';
   A<="000";
   B<="001";

   WAIT FOR 100ns;
   S<='0';
   A<="001";
   B<="010";

   WAIT FOR 100ns;
   S<='1';
   A<="010";
   B<="011";

   WAIT FOR 100ns;
   S<='0';
   A<="011";
   B<="100";

   WAIT FOR 100ns;
   S<='1';
   A<="100";
   B<="101";

   WAIT FOR 100ns;
```

```
        S<='0';
        A<="101";
        B<="110";

        WAIT FOR 100ns;
        S<='1';
        A<="110";
        B<="111";
        WAIT;
      end process;

      end Behavioral;
```

- 2-way 4-bit Multiplexer

```
      ----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 31.05.2023 14:53:15
-- Design Name:
-- Module Name: Sim_2way_4bit_MUX - 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 Sim_2way_4bit_MUX is
--  Port ( );
end Sim_2way_4bit_MUX;

architecture Behavioral of Sim_2way_4bit_MUX is
COMPONENT Mux_2way_4bit
   PORT(
      A : in STD_LOGIC_VECTOR (3 downto 0);
      B : in STD_LOGIC_VECTOR (3 downto 0);
      S : in STD_LOGIC;
      Q : out STD_LOGIC_VECTOR (3 downto 0));
END COMPONENT;

SIGNAL A : STD_LOGIC_VECTOR(3 downto 0);
SIGNAL B : STD_LOGIC_VECTOR(3 downto 0);
SIGNAL S : STD_LOGIC;
SIGNAL Q : STD_LOGIC_VECTOR(3 downto 0);

begin
UUT : Mux_2way_4bit PORT MAP(
   A=>A,
   B=>B,
   S=>S,
   Q=>Q);

process
begin
   S<='0';
   A<="0000";
   B<="0101";

   WAIT FOR 100ns;
   S<='0';
   A<="0110";
   B<="0111";

   WAIT FOR 100ns;
   S<='0';
   A<="1110";
   B<="0100";

   WAIT FOR 100ns;
   S<='0';
   A<="0111";
   B<="0000";

   WAIT;

 end process;

end Behavioral;
```

- 8-way 4-bit Multiplexer

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 31.05.2023 14:53:15
-- Design Name:
-- Module Name: Sim_2way_4bit_MUX - 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 Sim_2way_4bit_MUX is
--  Port ( );
end Sim_2way_4bit_MUX;

architecture Behavioral of Sim_2way_4bit_MUX is
COMPONENT Mux_2way_4bit
   PORT(
      A : in STD_LOGIC_VECTOR (3 downto 0);
      B : in STD_LOGIC_VECTOR (3 downto 0);
      S : in STD_LOGIC;
      Q : out STD_LOGIC_VECTOR (3 downto 0));
END COMPONENT;
```

```
SIGNAL A : STD_LOGIC_VECTOR(3 downto 0);
SIGNAL B : STD_LOGIC_VECTOR(3 downto 0);
SIGNAL S : STD_LOGIC;
SIGNAL Q : STD_LOGIC_VECTOR(3 downto 0);

begin
UUT : Mux_2way_4bit PORT MAP(
   A=>A,
   B=>B,
   S=>S,
   Q=>Q);

process
begin
   S<='0';
   A<="0000";
   B<="0101";

   WAIT FOR 100ns;
   S<='0';
   A<="0110";
   B<="0111";

   WAIT FOR 100ns;
   S<='0';
   A<="1110";
   B<="0100";

   WAIT FOR 100ns;
   S<='0';
   A<="0111";
   B<="0000";

   WAIT;

 end process;

end Behavioral;
```

- Register Bank

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 01.06.2023 17:39:25
-- Design Name:
-- Module Name: Sim_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 Sim_Reg_Bank is
--  Port ( );
end Sim_Reg_Bank;

architecture Behavioral of Sim_Reg_Bank is

COMPONENT Reg_Bank
   port(
         Reg_Enable : in STD_LOGIC_VECTOR(2 downto 0);
         Reset : in STD_LOGIC;
         D : in STD_LOGIC_VECTOR (3 downto 0);
         Clk : in STD_LOGIC;
         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_Enable : STD_LOGIC_VECTOR(2 downto 0);
signal D,Q0,Q1,Q2,Q3,Q4,Q5,Q6,Q7 : STD_LOGIC_VECTOR(3 downto 0);
signal Reset,Clk : STD_LOGIC;

begin
UUT : Reg_Bank
   PORT MAP(
      Reg_Enable => Reg_Enable,
      Reset => Reset,
      D =>D,
      Clk => Clk,
      Q0 => Q0,
      Q1 => Q1,
      Q2 => Q2,
      Q3 => Q3,
      Q4 => Q4,
      Q5 => Q5,
      Q6 => Q6,
      Q7 => Q7);
 Clk_process: process
                begin
                   Clk <= '0';
                   wait for 5 ns;
                   Clk <= '1';
                   wait for 5 ns;
                end process;

       process
       begin

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

       Reset<='0';
       Reg_Enable<="001";
       D<="0001";
       wait for 105ns;
```

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

        Reset<='0';
        Reg_Enable<="111";
        D<="0001";
        wait for 105ns;

        Reg_Enable<="001";
        D<="1001";
        wait for 102ns;

         Reg_Enable<="010";
               D<="1001";
               wait for 102ns;
        Reg_Enable<="101";
        D<="1101";
        wait for 103ns;

         Reg_Enable<="110";
         D<="0011";


     WAIT;
end process;

end Behavioral;
```

- Program ROM

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 06.06.2023 09:46:39
-- Design Name:
-- Module Name: Sim_Prog_Rom - 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 Sim_Prog_Rom is
--  Port ( );
end Sim_Prog_Rom;

architecture Behavioral of Sim_Prog_Rom is
   component Prog_ROM is
      Port (
         Memo_select : in STD_LOGIC_VECTOR(2 downto 0);
         Ins_bus : out STD_LOGIC_VECTOR(11 downto 0)
      );
   end component;

   signal Memo_select : STD_LOGIC_VECTOR(2 downto 0) := (others => '0');
   signal Ins_bus : STD_LOGIC_VECTOR(11 downto 0);

begin
   UUT: Prog_ROM
   port map (
      Memo_select => Memo_select,
      Ins_bus => Ins_bus
   );
   Stimulus_process: process
      begin
         -- Example test case: Read instructions from different memory locations
         Memo_select <= "000";
         wait for 10 ns;
         assert Ins_bus = "100010000011" report "Incorrect instruction" severity error;

         Memo_select <= "001";
         wait for 10 ns;
         assert Ins_bus = "100100000001" report "Incorrect instruction" severity error;
```

```
            Memo_select <= "010";
            wait for 10 ns;
            assert Ins_bus = "010100000000" report "Incorrect instruction" severity error;

            Memo_select <= "011";
            wait for 10 ns;
            assert Ins_bus = "001110010000" report "Incorrect instruction" severity error;

            Memo_select <= "100";
            wait for 10 ns;
            assert Ins_bus = "000010100000" report "Incorrect instruction" severity error;

            Memo_select <= "101";
            wait for 10 ns;
            assert Ins_bus = "110010000111" report "Incorrect instruction" severity error;

            Memo_select <= "110";
            wait for 10 ns;
            assert Ins_bus = "110000000011" report "Incorrect instruction" severity error;

            Memo_select <= "111";
            wait for 10 ns;
            assert Ins_bus = "101010101010" report "Incorrect instruction" severity error;

            wait;
        end process;

end Behavioral;
```

- Instruction Decoder

```
----------------------------------------------------------------------------------
-- Company: 
-- Engineer: 
-- 
-- Create Date: 03.06.2023 08:24:28
-- Design Name: 
-- Module Name: Sim_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 Sim_Ins_decoder is
--  Port ( );
end Sim_Ins_decoder;

architecture Behavioral of Sim_Ins_decoder is
   COMPONENT Ins_decoder
      PORT(
         Data : in STD_LOGIC_VECTOR (11 downto 0);
         Reg_check : in STD_LOGIC_VECTOR (3 downto 0);
         Reg_enable : out STD_LOGIC_VECTOR (2 downto 0);
         Immediate_value : out STD_LOGIC_VECTOR (3 downto 0);
         Load_select : out STD_LOGIC;
         Reg_select0 : out STD_LOGIC_VECTOR (2 downto 0);
         Reg_select1 : out STD_LOGIC_VECTOR (2 downto 0);
         Add_sub_select : out STD_LOGIC;
         Jump_flag : out STD_LOGIC;
         Jump_address : out STD_LOGIC_VECTOR (2 downto 0));
   END COMPONENT;
signal Data : STD_LOGIC_VECTOR (11 downto 0);
signal Reg_check,Immediate_value : STD_LOGIC_VECTOR (3 downto 0);
signal Reg_enable,Reg_select0,Reg_select1,Jump_address :  STD_LOGIC_VECTOR (2 downto
0);
signal Load_select,Add_sub_select,Jump_flag : STD_LOGIC;
begin

   UUT: Ins_decoder
      port map (
          Data => Data,
          Reg_check => Reg_check,
          Reg_enable => Reg_enable,
          Immediate_value => Immediate_value,
          Load_select => Load_select,
          Reg_select0 => Reg_select0,
            Reg_select1 => Reg_select1,
            Add_sub_select => Add_sub_select,
            Jump_flag => Jump_flag,
            Jump_address => Jump_address
          );

    process
    begin
      Data <= "100010000011"; --MOVI R1,3
      Reg_check <= "0011";
      WAIT FOR 100ns;

      Data <= "100100000001"; --MOVI R2,1
      Reg_check <= "0001";
      WAIT FOR 100ns;

      Data <= "010100000000"; --NEG R2
      Reg_check <= "1011";
      WAIT FOR 100ns;

      Data <= "001110010000"; --ADD R7,R1
      Reg_check <= "0111";
      WAIT FOR 100ns;

      Data <= "000010100000"; --ADD R1,R2
      Reg_check <= "1011";
      WAIT FOR 100ns;

      Data <= "110010000111"; --JZR R1,7
      Reg_check <= "0000";
      WAIT;
    end process;

end Behavioral;
```

- Slow Clock

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/31/2023 07:39:29 PM
-- Design Name:
-- Module Name: Slow_Clk_TB - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
```

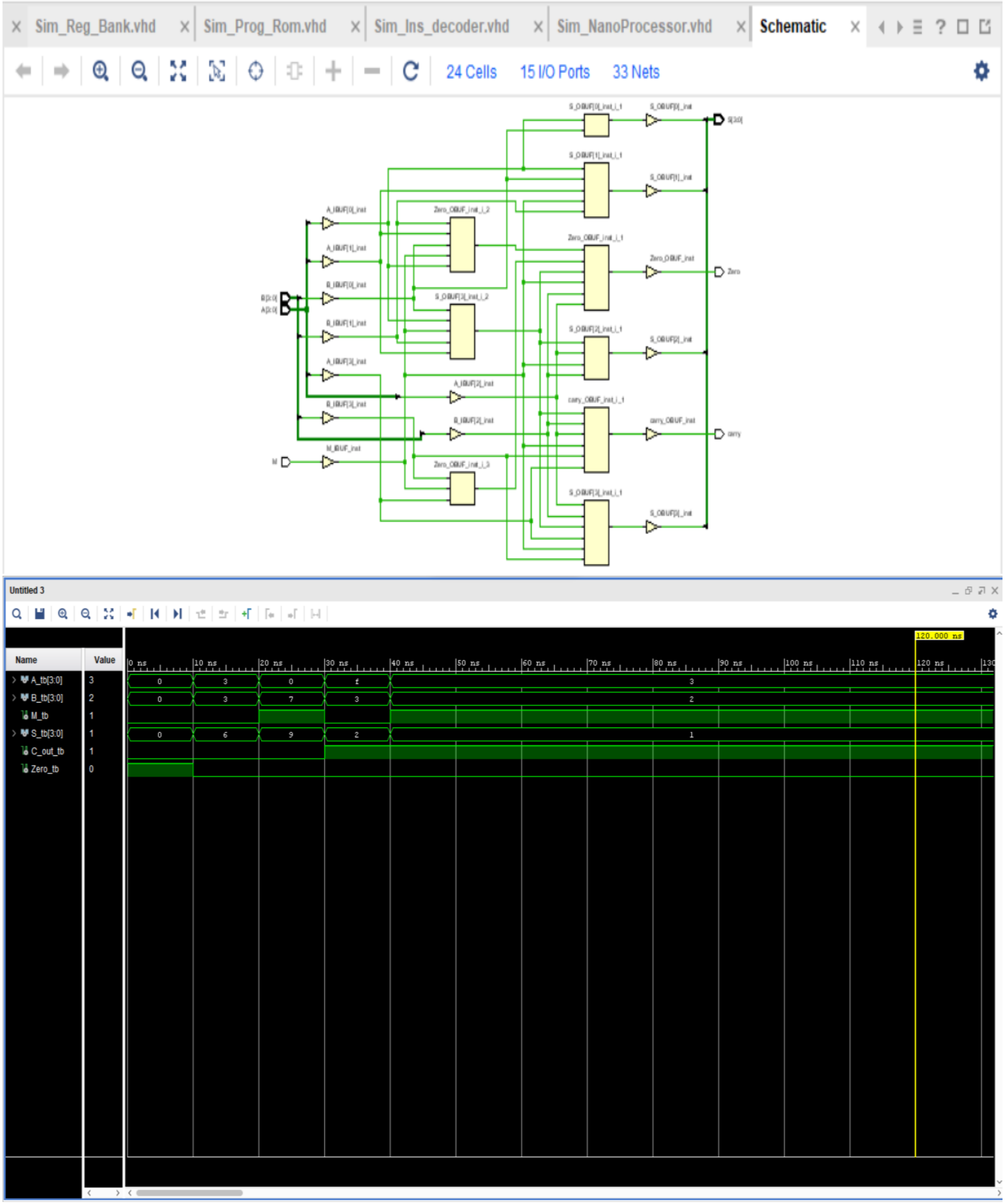
```
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------

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: 06.06.2023 16:09:47
-- Design Name: 
-- Module Name: Sim_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 Sim_NanoProcessor is
--  Port ( );
end Sim_NanoProcessor;

architecture Behavioral of Sim_NanoProcessor is
  component NanoProcessor is
    Port ( Clk : in STD_LOGIC;
           Reset : in STD_LOGIC;
           zero : out STD_LOGIC;
           Carry : out STD_LOGIC;
           R7 : out STD_LOGIC_VECTOR(3 downto 0);
           Display_7_Seg : out STD_LOGIC_VECTOR(6 downto 0);
           Anode_activate : out STD_LOGIC_VECTOR (3 downto 0));
  end component;

  signal Clk, Reset, zero, Carry : STD_LOGIC;
  signal R7 : STD_LOGIC_VECTOR(3 downto 0);
  signal Display_7_Seg : STD_LOGIC_VECTOR(6 downto 0);
  signal Anode_activate : STD_LOGIC_VECTOR(3 downto 0);

begin
  -- Instantiate the NanoProcessor
  uut : NanoProcessor
    port map (
      Clk => Clk,
      Reset => Reset,
      zero => zero,
      Carry => Carry,
      R7 => R7,
      Display_7_Seg => Display_7_Seg,
      Anode_activate => Anode_activate);

   -- Clock process
  Clk_process: process
  begin
    Clk <= '0';
    wait for 5 ns;
    Clk <= '1';
    wait for 5 ns;
  end process;


 stimulus_process: process
  begin
    Reset <= '1';
    wait for 100 ns;
    Reset <= '0';


    -- Add additional stimulus or checks if needed

    -- Wait for simulation to finish
    wait;
  end process stimulus_process;
end Behavioral;
```

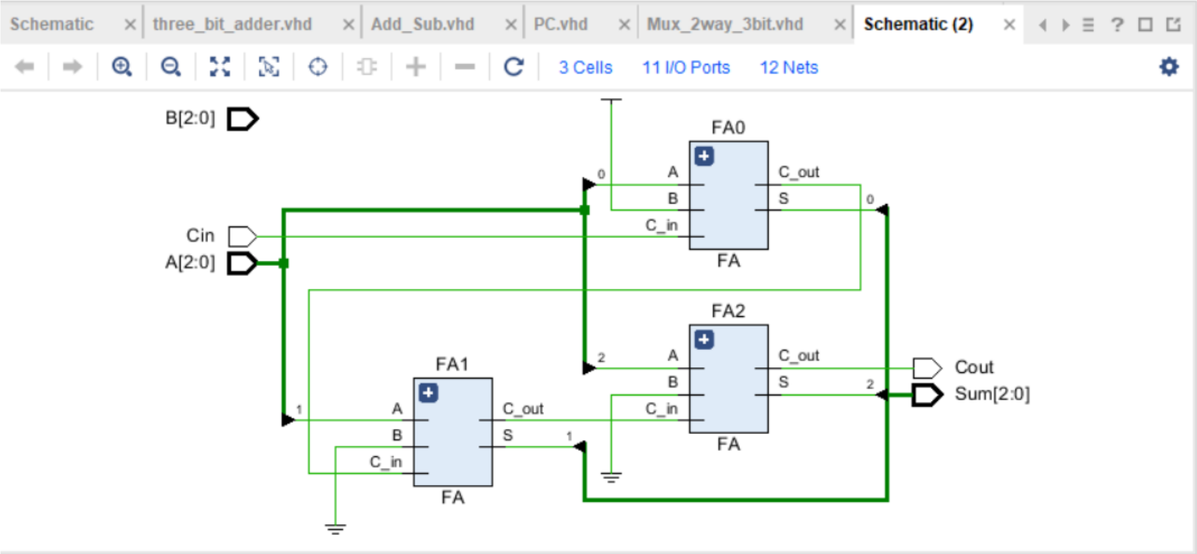
<u>**TIME DIAGRAMS**</u>

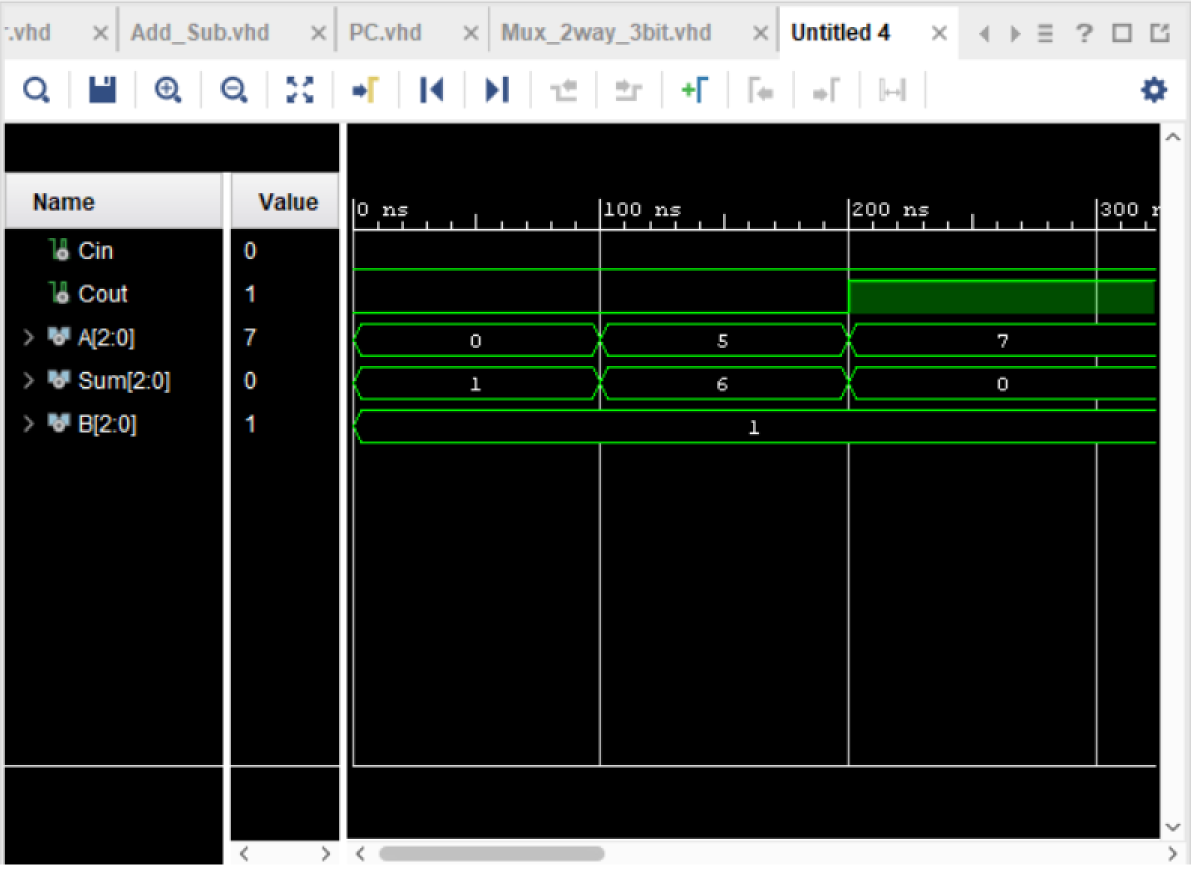
- **4-bit Add/Subtract Unit**



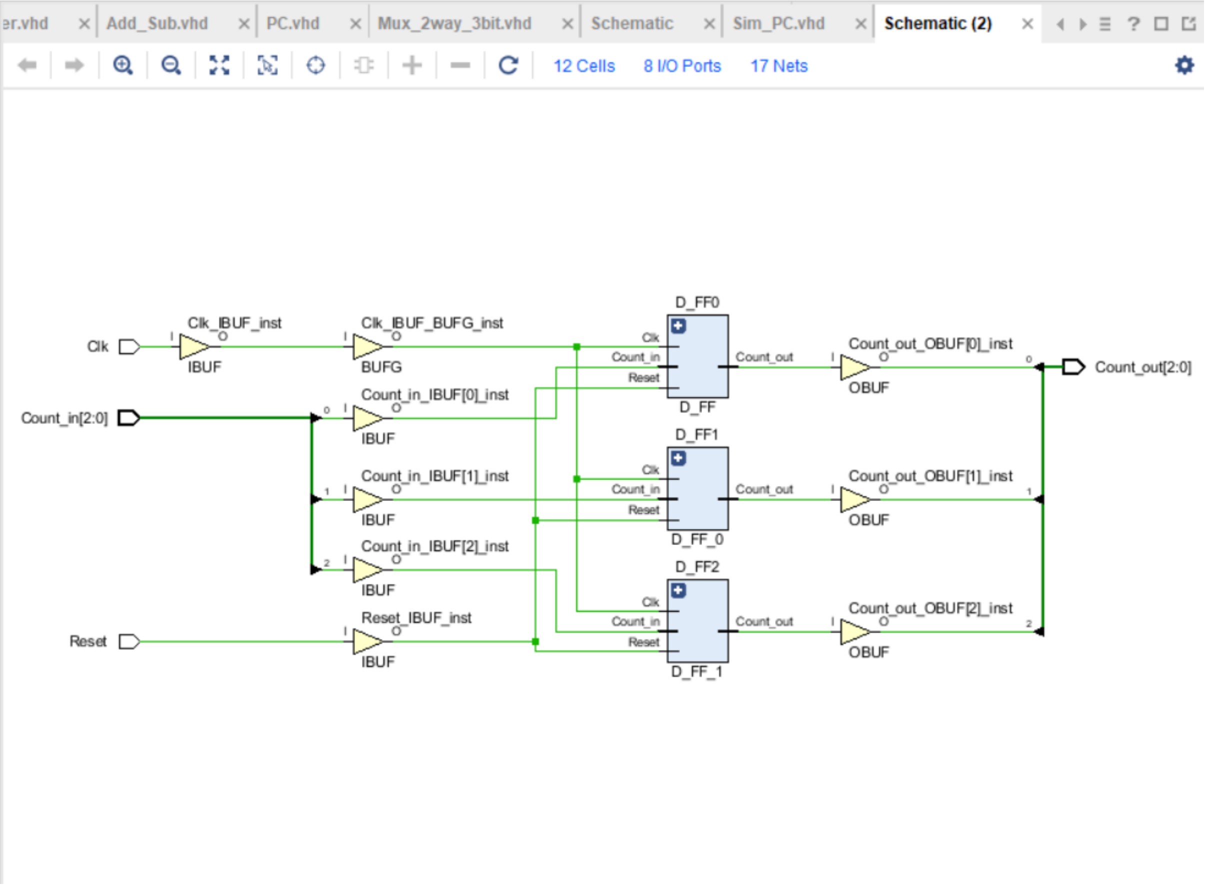
Name	Value
A_tb[3:0]	3
B_tb[3:0]	2
M_tb	1
S_tb[3:0]	1
C_out_tb	1
Zero_tb	0

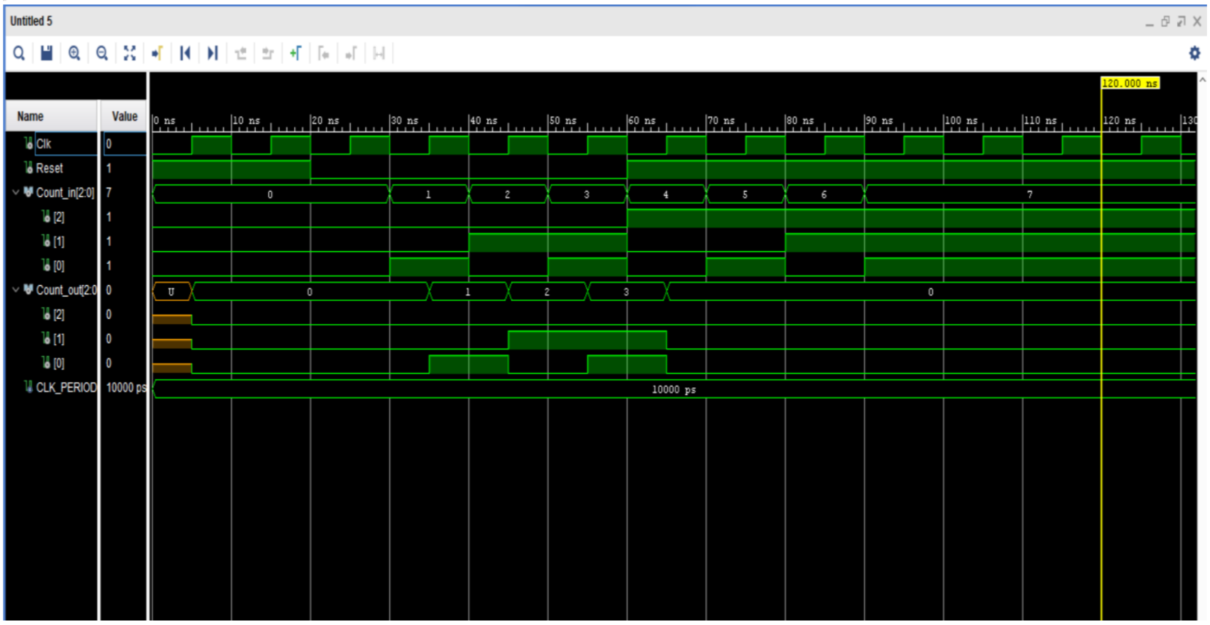
- **3-bit Adder**


Schematic
three_bit_adder.vhd
Add_Sub.vhd
PC.vhd
Mux_2way_3bit.vhd
Schematic (2)
3 Cells
11 I/O Ports
12 Nets
B[2:0]
Cin
A[2:0]
FA0
FA1
FA2
A
B
C_in
C_out
S
FA
Cout
Sum[2:0]

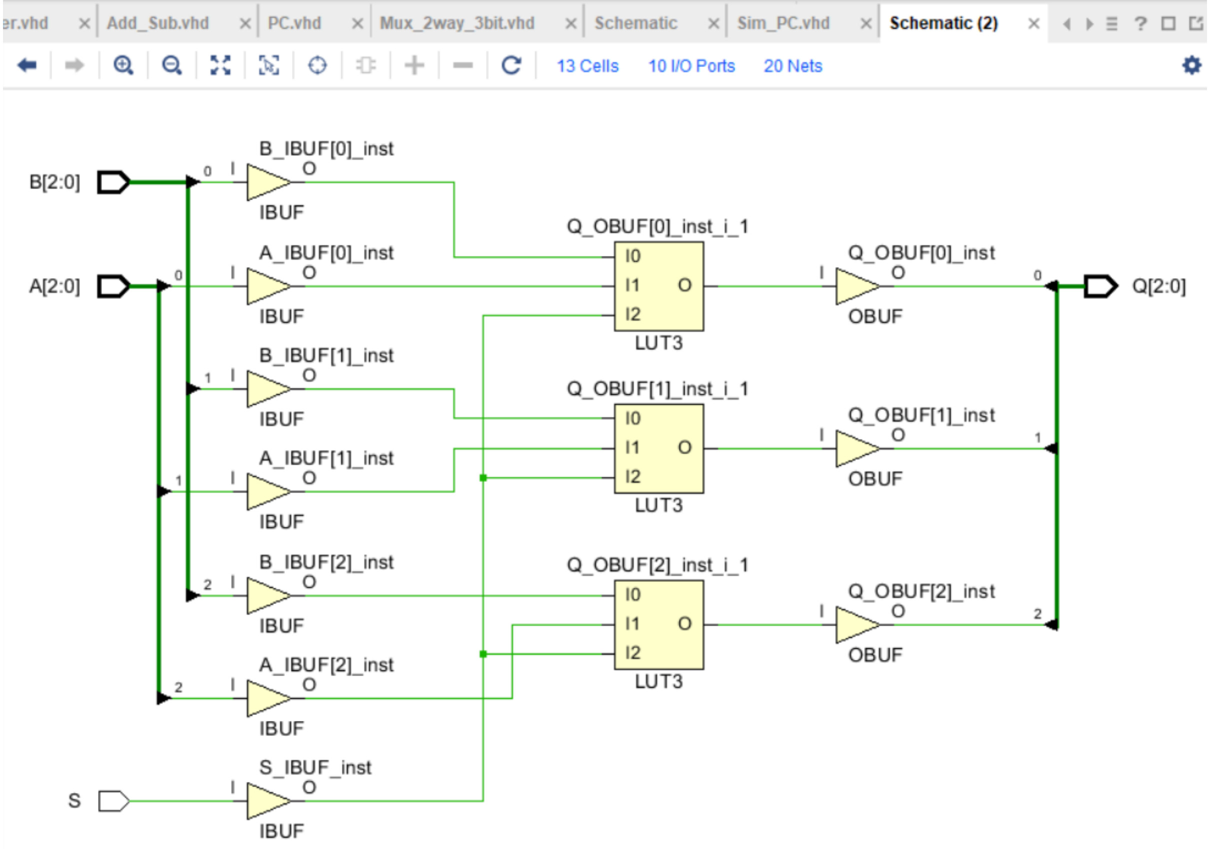

.vhd
Add_Sub.vhd
PC.vhd
Mux_2way_3bit.vhd
Untitled 4
Name
Value
Cin 0
Cout 1
A[2:0] 7
Sum[2:0] 0
B[2:0] 1
0 ns
100 ns
200 ns
300 ns
0
5
7
1
6
0
1

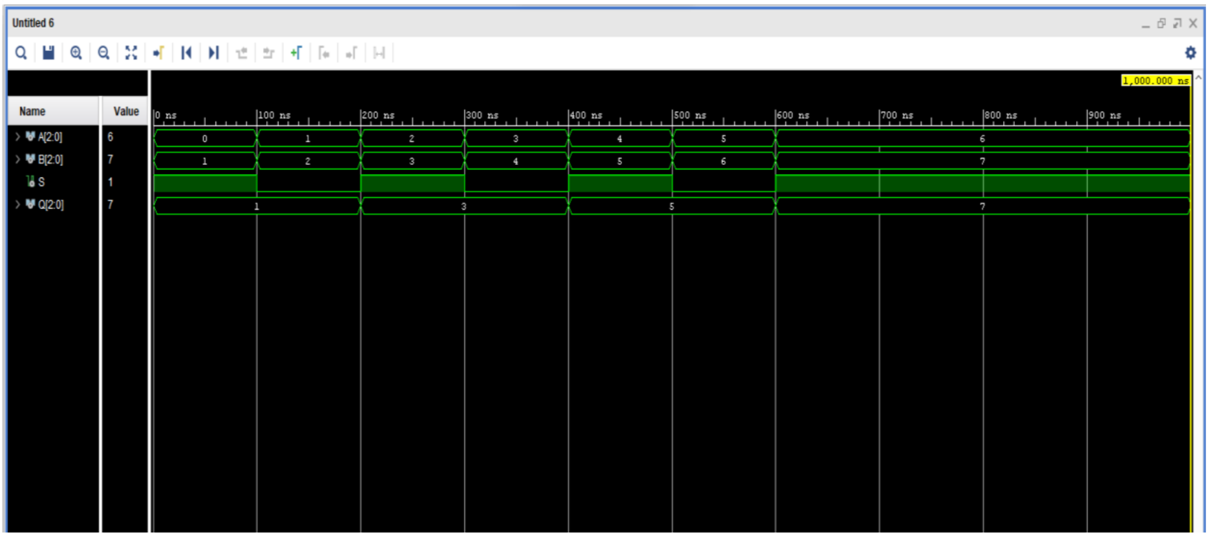
- **3-bit Program Counter**



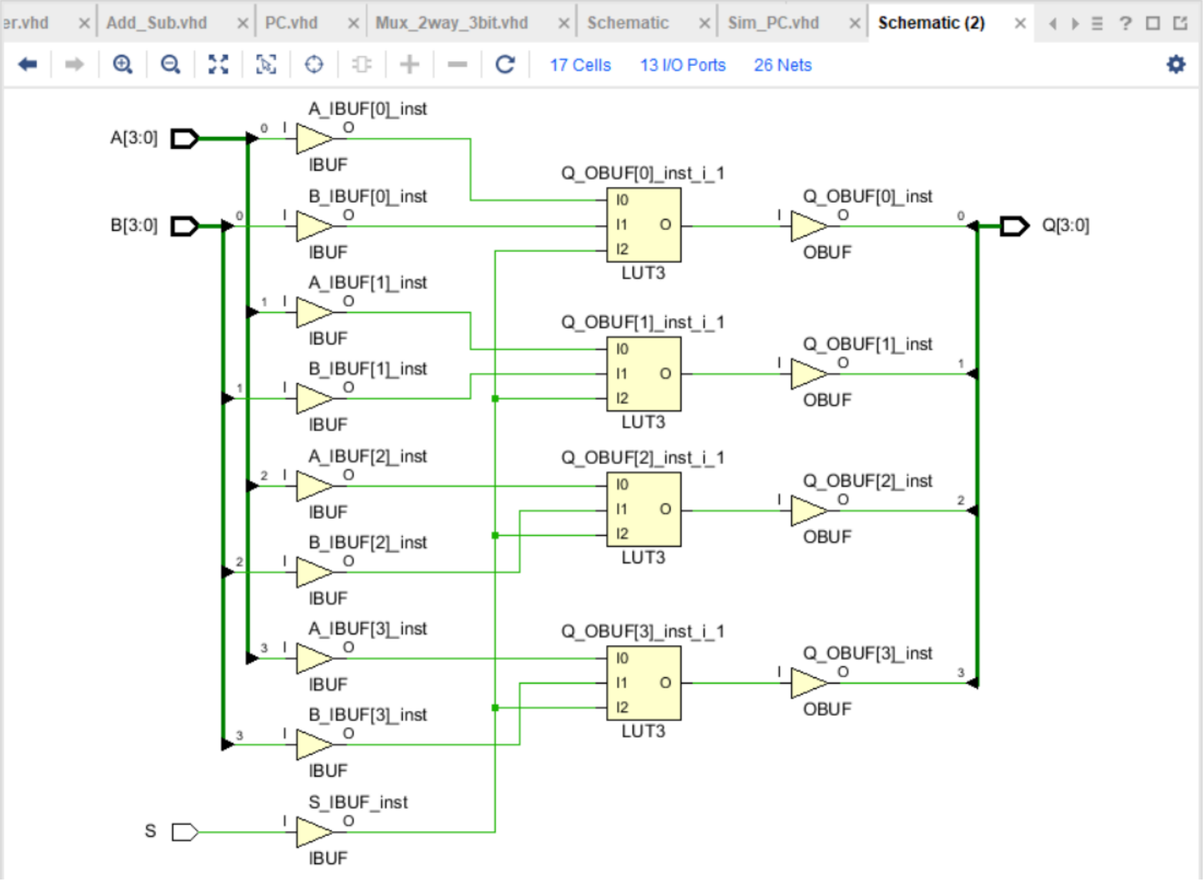


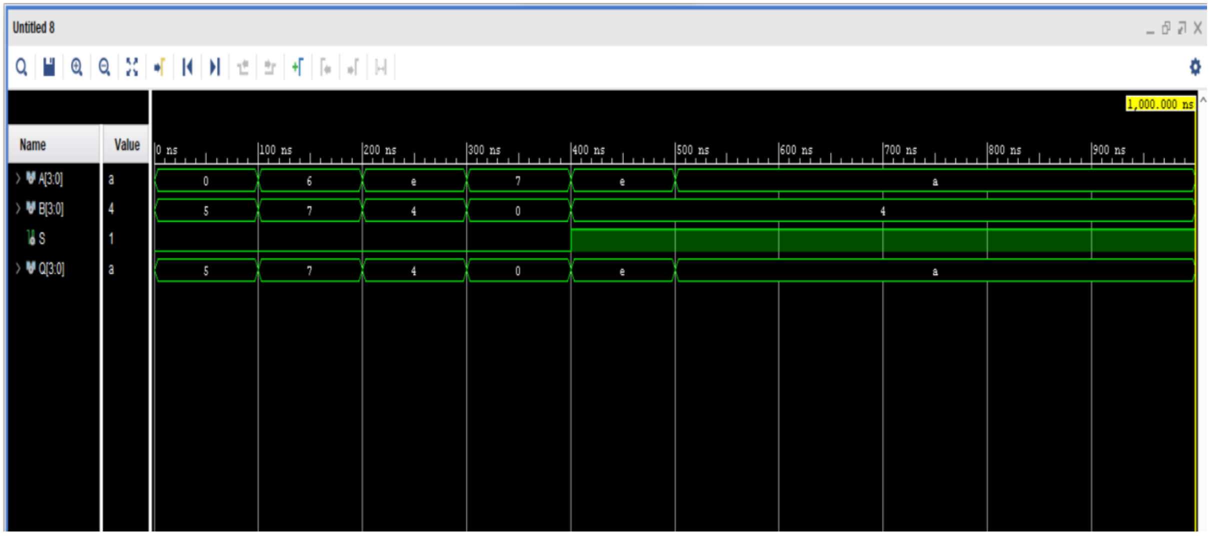
- **2-way 3-bit Multiplexer**



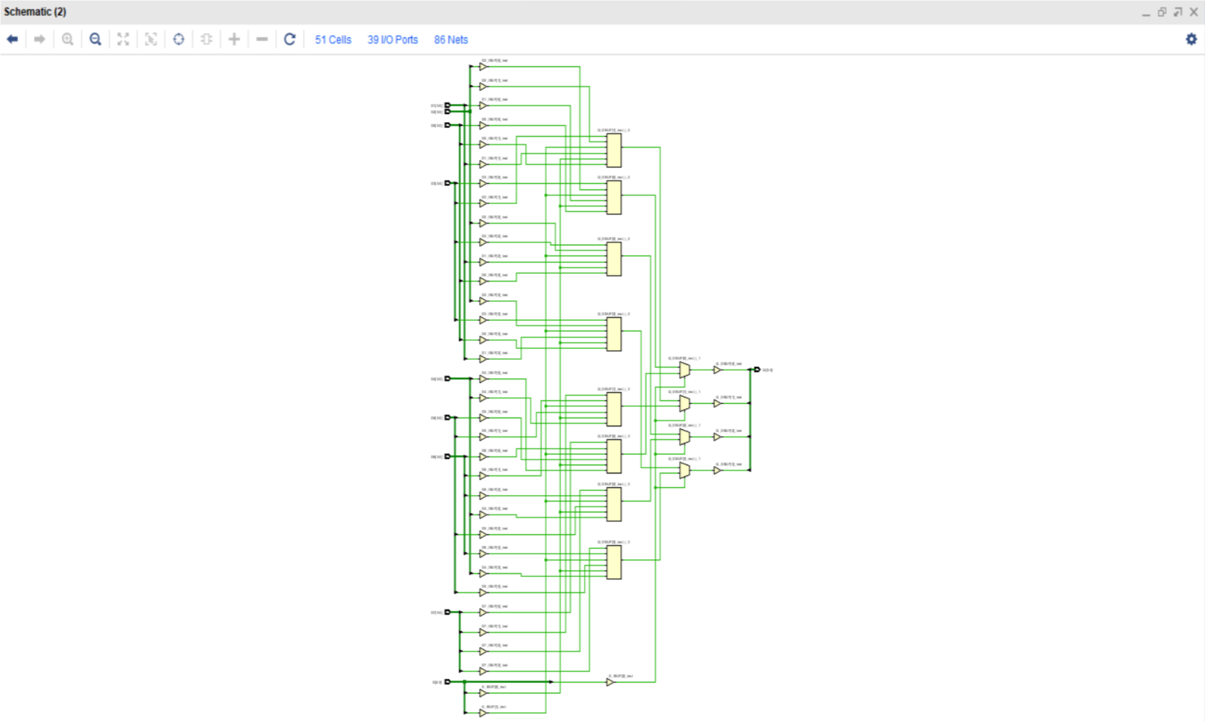


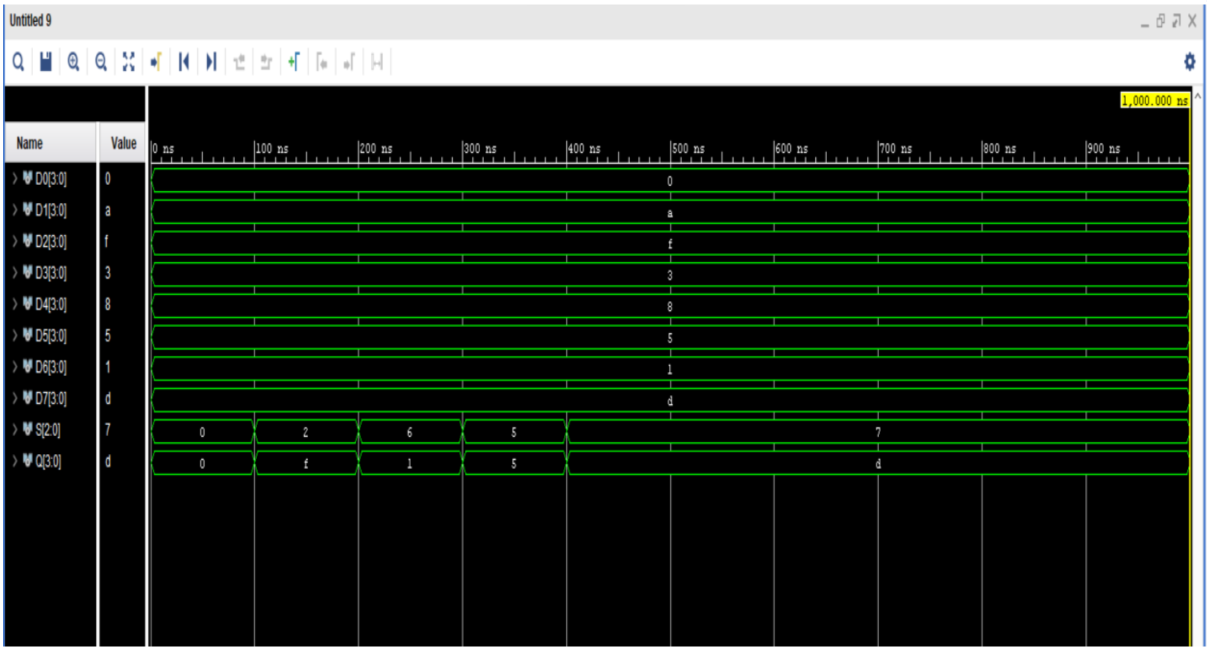
- **2-way 4-bit Multiplexer**





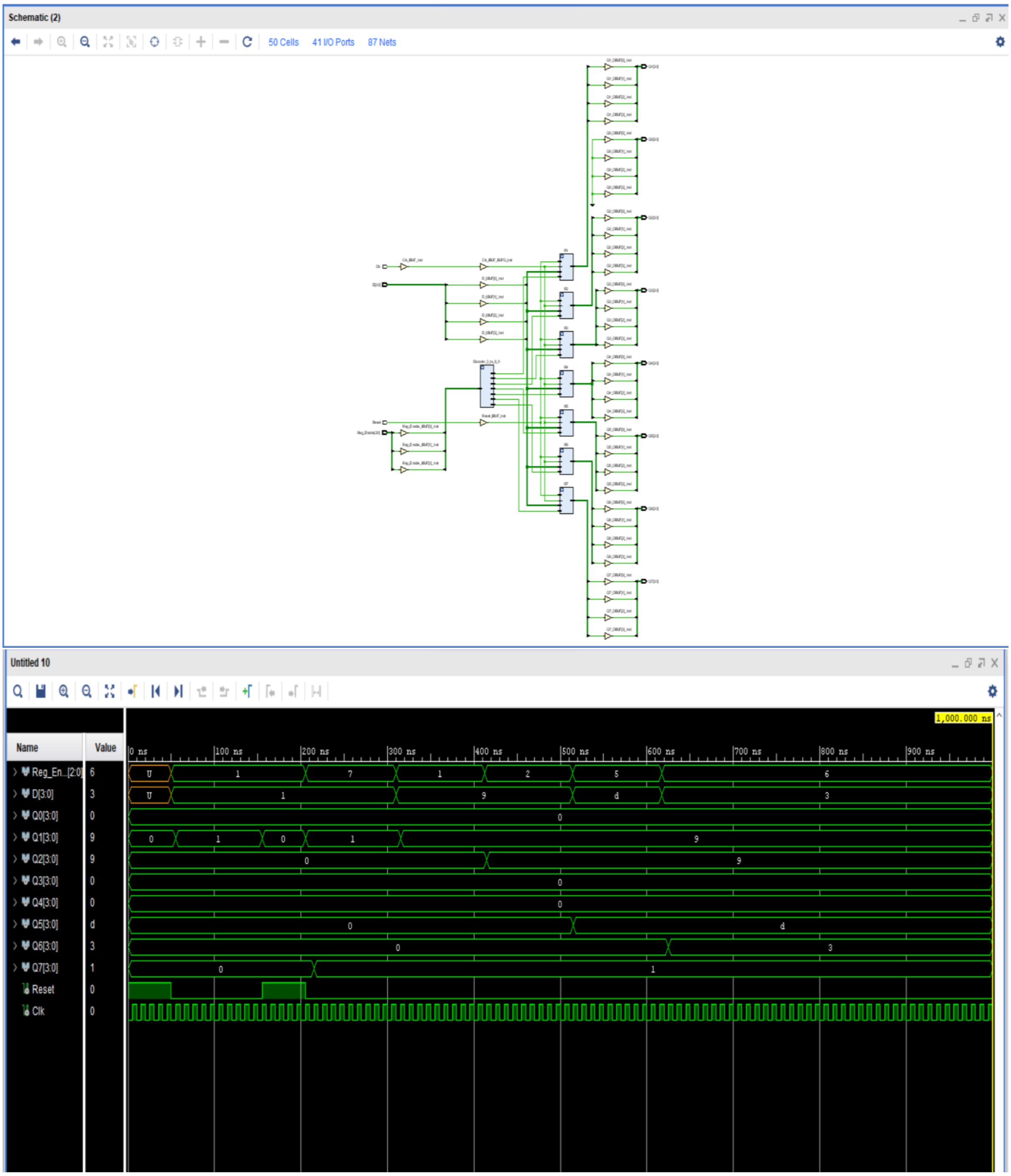
- **8-way 4-bit Multiplexer**





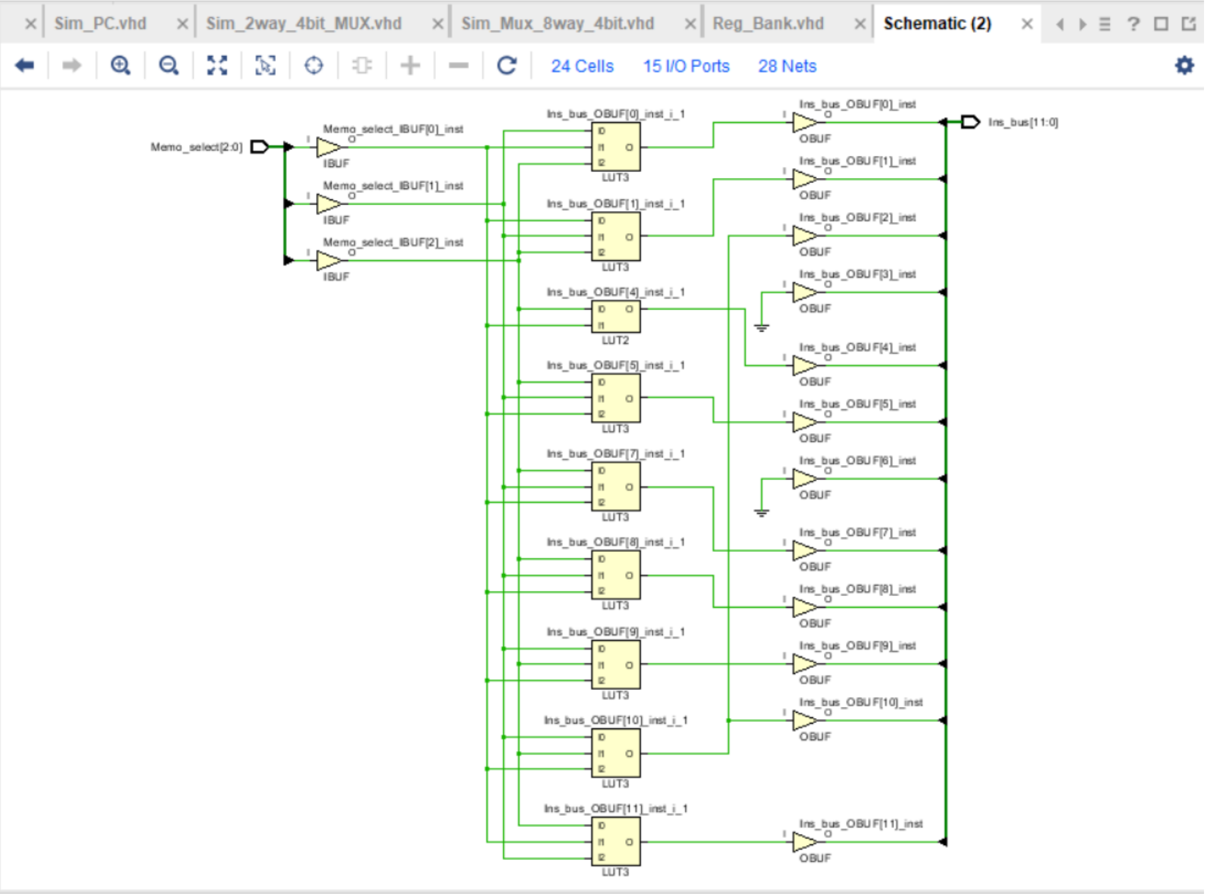
Name	Value
D0[3:0]	0
D1[3:0]	a
D2[3:0]	f
D3[3:0]	3
D4[3:0]	8
D5[3:0]	5
D6[3:0]	1
D7[3:0]	d
S[2:0]	7
Q[3:0]	d

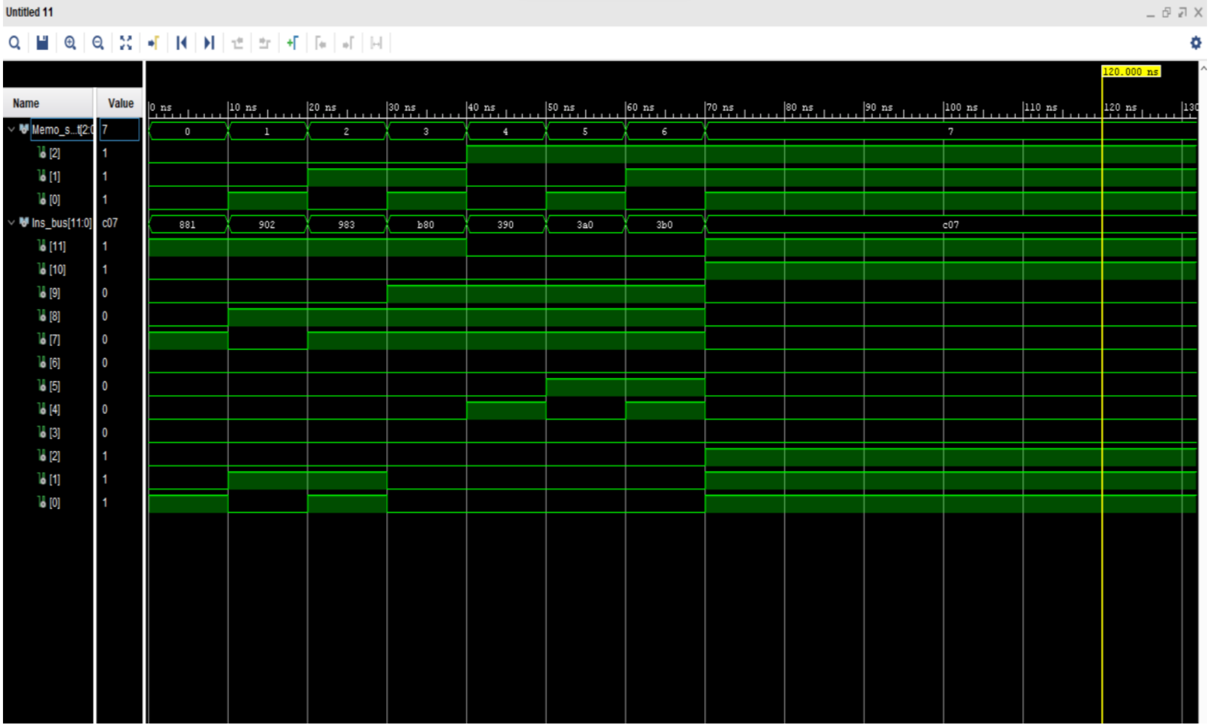
- **Register Bank**



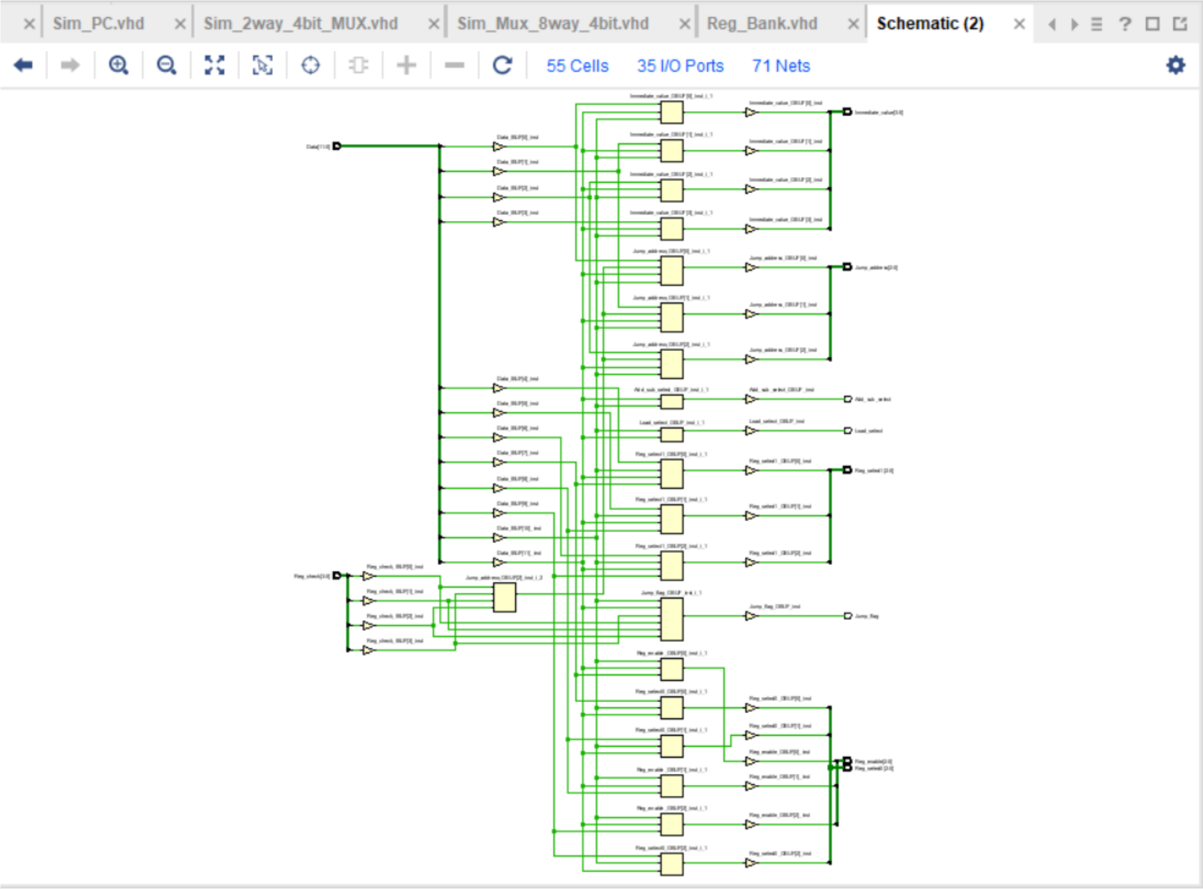
Name	Value
Reg_En_[2:0]	6
D[3:0]	3
Q0[3:0]	0
Q1[3:0]	9
Q2[3:0]	9
Q3[3:0]	0
Q4[3:0]	0
Q5[3:0]	d
Q6[3:0]	3
Q7[3:0]	1
Reset	0
Clk	0

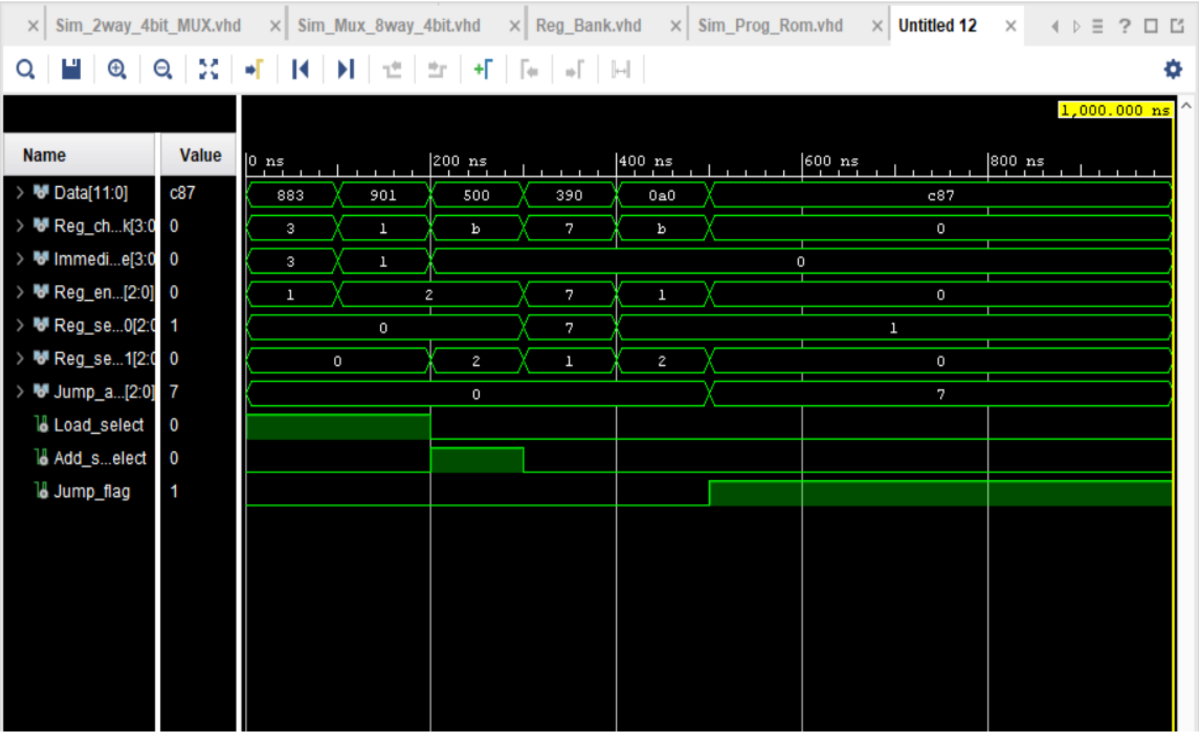
- **Program ROM**


Sim_PC.vhd
Sim_2way_4bit_MUX.vhd
Sim_Mux_8way_4bit.vhd
Reg_Bank.vhd
Schematic (2)
24 Cells
15 I/O Ports
28 Nets
Memo_select[2:0]
Memo_select_IBUF[0]_inst
Memo_select_IBUF[1]_inst
Memo_select_IBUF[2]_inst
IBUF
Ins_bus_OBUF[0]_inst_i_1
Ins_bus_OBUF[1]_inst_i_1
Ins_bus_OBUF[4]_inst_i_1
Ins_bus_OBUF[5]_inst_i_1
Ins_bus_OBUF[7]_inst_i_1
Ins_bus_OBUF[8]_inst_i_1
Ins_bus_OBUF[9]_inst_i_1
Ins_bus_OBUF[10]_inst_i_1
Ins_bus_OBUF[11]_inst_i_1
LUT3
LUT2
Ins_bus_OBUF[0]_inst
Ins_bus_OBUF[1]_inst
Ins_bus_OBUF[2]_inst
Ins_bus_OBUF[3]_inst
Ins_bus_OBUF[4]_inst
Ins_bus_OBUF[5]_inst
Ins_bus_OBUF[6]_inst
Ins_bus_OBUF[7]_inst
Ins_bus_OBUF[8]_inst
Ins_bus_OBUF[9]_inst
Ins_bus_OBUF[10]_inst
Ins_bus_OBUF[11]_inst
OBUF
Ins_bus[11:0]

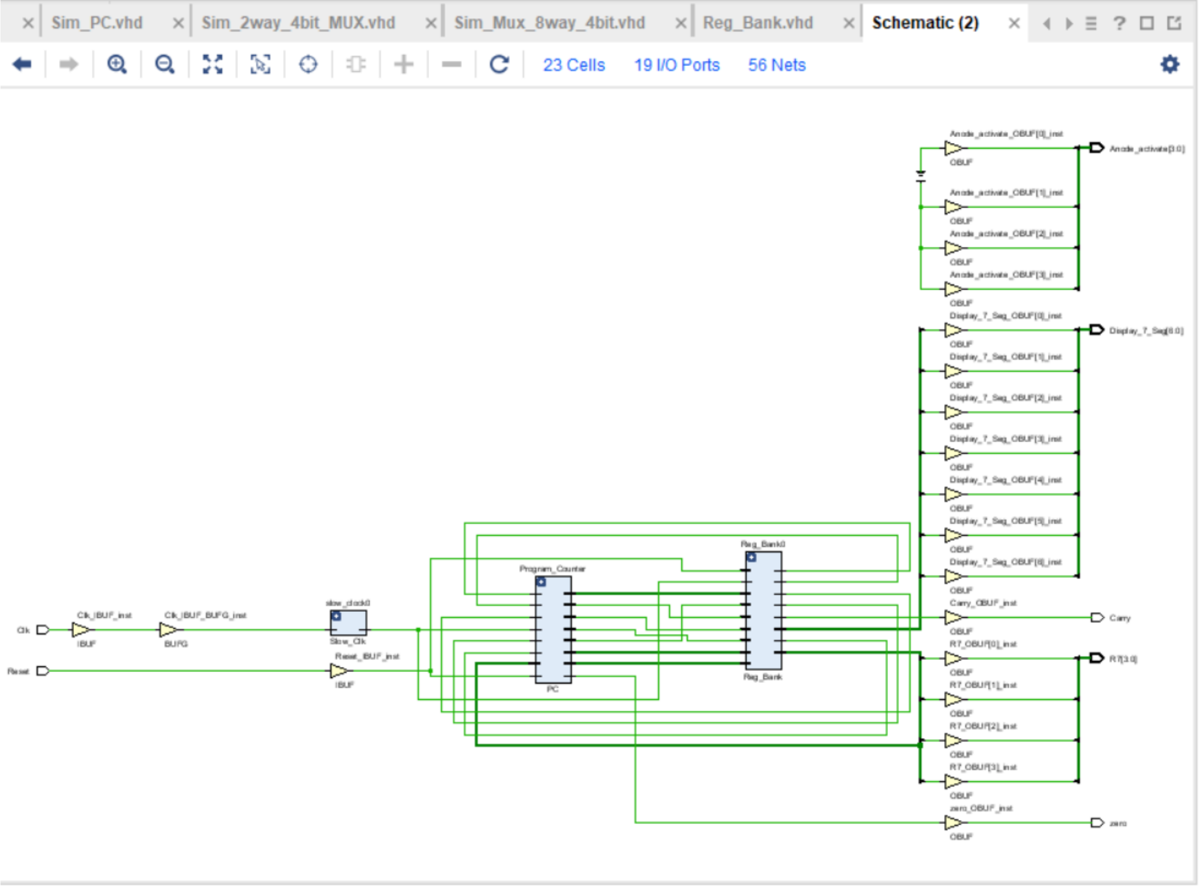

Untitled 11
Name
Value
Memo_s...t[2:0]
7
[2] 1
[1] 1
[0] 1
Ins_bus[11:0]
c07
[11] 1
[10] 1
[9] 0
[8] 0
[7] 0
[6] 0
[5] 0
[4] 0
[3] 0
[2] 1
[1] 1
[0] 1
120.000 ns
0 ns
10 ns
20 ns
30 ns
40 ns
50 ns
60 ns
70 ns
80 ns
90 ns
100 ns
110 ns
120 ns
0
1
2
3
4
5
6
7
881
902
983
b80
390
3a0
3b0
c07

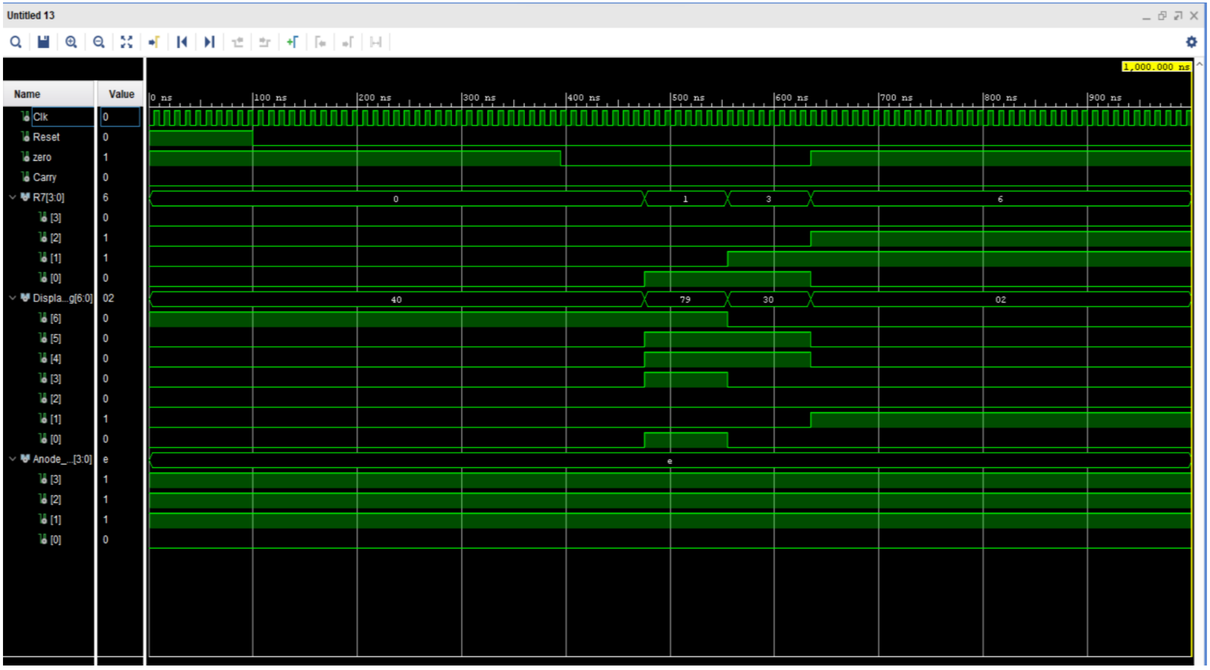
- **Instruction Decoder**





- **Nano Processor**




Untitled 13

Name	Value
Clk	0
Reset	0
zero	1
Carry	0
R7[3:0]	6
[3]	0
[2]	1
[1]	1
[0]	0
Displa...g[6:0]	02
[6]	0
[5]	0
[4]	0
[3]	0
[2]	0
[1]	1
[0]	0
Anode_...[3:0]	e
[3]	1
[2]	1
[1]	1
[0]	0

CONCLUSIONS

➢ This project gave us a 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 were recall 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 subcomponents 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.

CONTRIBUTION OF EACH TEAM MEMBER

NAME	Designed parts of the Nano Processor	No. of hours spend
Nayanathara P.M.C	We created each component of the Nano processor through a discussion method. • 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	16
Pathirana D. P		16