Lab 9-10 – Nanoprocessor Design Competition

CS1050 – Computer Organization and Digital Design

Department of Computer Science and Engineering

Project By Group 42

Group Members

1. G.D.C.Kaveesha - 220313J
2. W.A.K.Indunil - 220240G
3. J.A.U.C.Jayakody - 220248M
4. M.G.H.A.Chamindi - 220081T

Lab Task

In this lab we were assigned to design a 4-bit processor capable of executing four instructions. To achieve this, we leveraged previously developed components including a 4-bit Add/Subtract unit, 3-bit adder, 3-bit Program Counter (PC), k-way b-bit multiplexers, Register Bank, Program ROM, Instruction Decoder, 7-Segment Display, and a slow clock.

Since this lab was given as a group project the project's workload was distributed among five team members, During the entire process we designed all the components initially and modified them several times to ensure the required functionalities of the processor.

Upon project completion, we successfully engineered a 4-bit Arithmetic Unit capable of performing signed integer addition and subtraction, along with k-way b-bit Multiplexers, Program ROM, and Instruction Decoder. Finally thorough simulation and implementation on the Basys3 board validated component functionality.

Individual contributions of each group member

Member	Contribution	Worked Time
G.D.C.Kaveesha (220313J)	• Nano processor, 8-way 1-bit, 8-way 4-bit, 2-way 3-bit, 2-way 4-bit and 2-way 12-bit Multiplexers, program rom • Test Bench files for each multiplexer, TB_Program_ROM • Multiplier • Lab report	45 Hours
W.A.K.Indunil (220240G)	• Nano processor,3-to-8 Decoder, Registers and Register bank • TB_Registers, TB_Register_Bank, TB_3-to-8 Decoder • Comparator • Lab report	40 Hours
J.A.U.C.Jayakody (220248M)	• Nano processor, Instruction decoder, 3-bit adder, slow clock, constrains files • TB_Instruction_Decoder, TB_slow_clock ,TB_3-bit adder • Complement • Lab report	45 Hours
M.G.H.A.Chamindi (220081T)	• Nano processor, 4-bit Add/Subtract unit, program counter, LUT_16_7_0 • TB_Add_Sub, TB_Program_Counter, TB_LUT_16_7 • JMP • Lab report	40 Hours

Content

1. Introduction

This is a simple microprocessor capable of executing a simple set of instructions. In order to build the microprocessor, we developed and extended the following components.

- 4-bit Add/Subtract unit - This unit have the ability to add and subtracting numbers represented using 2's complement and this was built by modifying 4-bit RCA.
- 3-bit adder – This unit is used to increment the program counter. This was also built by modifying 4-bit RCA.
- 3-bit Program Counter (PC) - Program Counter is used to keep track with the next instruction to be executed. This was built by using three D-flipflops and it can be reset to 0 when required.
- Multiplexers – Here we have used a set of k-way b-bit multiplexers to enable the components in the microprocessor. The types of multiplexers used,

    01. 2-way 3-bit multiplexer

    02. 2-way 4-bit multiplexer

    03. 8-way 4-bit multiplexer

- Register bank – There are eight registers in the register bank and each one can store 4-bit value at a time. 3-to-8 Decoder in the register bank select which register to be enabled using the "register enabled signal".
- Program rom – This is the component which stores the Assembly Program. As the microprocessor only understands machine language, we hard coded the instructions as binary values in the program rom.
- Instructions Decoder - The main function of the instruction decoder is activating necessary components based on the instructions we wish to execute.

The set of instructions

Introduction	Description	Format
MOVI R, d	Move immediate value d to register R, i.e., R← d, R ∈ [0, 7], d ∈ [0, 15]	1 0 R R R 0 0 0 d d d d
ADD Ra, Rb	Add values in registers Ra and Rb and store the result in Ra, i.e., Ra ← Ra + Rb (Ra, Rb ∈ [0, 7])	0 0 Ra Ra Ra Rb Rb Rb 0 0 0 0
NEG R	2's complement of registers R, i.e., R ← – R, R ∈ [0, 7]	0 1 R R R 0 0 0 0 0 0 0
JZR R, d	Jump if value in register R is 0, i.e., If R == 0 PC ← d; Else PC ← PC + 1; (R ∈ [0, 7], d∈ [0, 7])	1 1 R R R 0 0 0 0 d d d

2. Main Module

2.1 Design source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/10/2024 08:45:14 AM
-- Design Name:
-- Module Name: Nano_Processor - 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 Nano_Processor is
    Port ( Clk : in STD_LOGIC; --clock of the basys 3 board
           Reset : in STD_LOGIC; -- reset signal(connect to button of basys 3)
           overflow_LED : out STD_LOGIC; --overflow detector connected to the led of basys3
           zero_LED : out STD_LOGIC; --zero flag detector connected to the led of basys3
           out_LED : out STD_LOGIC_vector(3 downto 0); --we can see the value in register
seven using this
           seven_segment_in : out STD_LOGIC_vector(6 downto 0); --give the value inside reg
seven as an input to sevensegment display
           Anode_Selector : out STD_LOGIC_VECTOR (3 downto 0) --select the seven segment
display
             );
end Nano_Processor;

architecture Behavioral of Nano_Processor is

--component used in nano processor
component Instruction_Decoder is
    Port ( Instruction : in STD_LOGIC_VECTOR (11 downto 0);
           Register_Check_For_Jump : in STD_LOGIC_VECTOR (3 downto 0);
           Register_Enable : out STD_LOGIC_VECTOR (2 downto 0);
           Immediate_Value : out STD_LOGIC_VECTOR (3 downto 0);
           Load_Selector : out STD_LOGIC;
           A_Register_Selector : out STD_LOGIC_VECTOR (2 downto 0);
           B_Register_Selector : out STD_LOGIC_VECTOR (2 downto 0);
           Abb_Sub_Selector : out STD_LOGIC;
           Jump_Flag : out STD_LOGIC;
           Jump_Address : out STD_LOGIC_VECTOR (2 downto 0)
          );
end component;

component twoWay_4bit_Mux is
    Port ( D : in STD_LOGIC_VECTOR (7 downto 0);
           S : in STD_LOGIC;
           Y : out STD_LOGIC_VECTOR (3 downto 0));
```

```
end component;

component Register_bank is
    Port (
    register_enable:in STD_LOGIC_VECTOR (2 downto 0);--to decide which register going to be
enabled
    data:in STD_LOGIC_VECTOR (3 downto 0);
    clk:in STD_LOGIC; --to input slow clock signal
    clr:in STD_LOGIC; --related to reset button
    reg_out:out STD_LOGIC_VECTOR (31 downto 0)
    );
end component;

component four_Bit_Add_Sub_Unit 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);
           overflow  : out STD_LOGIC;
           Zero_flag : out STD_LOGIC;
           C_out : out STD_LOGIC);

end component ;

component Program_Counter is
    Port ( D_in : in STD_LOGIC_VECTOR (0 to 2);
           Push : in STD_LOGIC;
           Clk : in STD_LOGIC;
           D_out : out STD_LOGIC_VECTOR (0 to 2));
end component ;

component  Program_ROM is
    Port ( Memory_select : in STD_LOGIC_VECTOR (2 downto 0);
           Instruction_Bus : out STD_LOGIC_VECTOR (11 downto 0));
end component ;

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

component  twoWay_3bit_Mux is
Port ( D : in STD_LOGIC_VECTOR (5 downto 0);
       S : in STD_LOGIC;
       Y : out STD_LOGIC_VECTOR (2 downto 0));
```

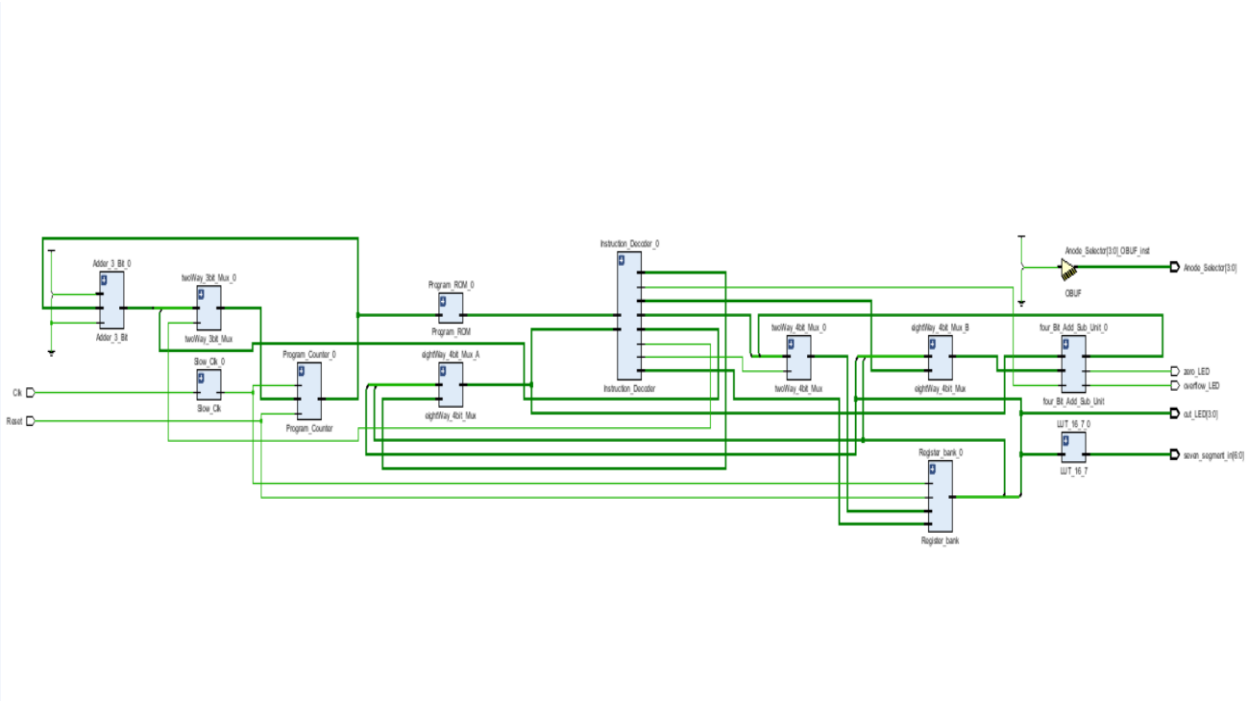
```
end component ;

component eightWay_4bit_Mux is
    Port ( D : in STD_LOGIC_VECTOR (31 downto 0);
           S : in STD_LOGIC_VECTOR (2 downto 0);
           Y : out STD_LOGIC_VECTOR (3 downto 0)
           );
end component;

component Adder_3_Bit is
    Port ( A : in STD_LOGIC_VECTOR (2 downto 0);
           B : in STD_LOGIC_VECTOR (2 downto 0);
           S : out STD_LOGIC_VECTOR (2 downto 0);
           C_in  : in STD_LOGIC ;
           overflow : out STD_LOGIC);
end component;

component LUT_16_7 is
    Port ( address : in STD_LOGIC_VECTOR (3 downto 0);
           data : out STD_LOGIC_VECTOR (6 downto 0));
end component;

--signals
signal register_enable: std_logic_vector(2 downto 0);
signal load_selector: std_logic;
signal immediate_value: std_logic_vector(3 downto 0);
signal A_register_select: std_logic_vector(2 downto 0);
signal B_register_select: std_logic_vector(2 downto 0);
signal add_sub_select: std_logic;
signal jump_flag: std_logic;
signal adderss_to_jump: std_logic_vector(2 downto 0);
signal ins_bus : std_logic_vector(11 downto 0);
signal mem_sel : std_logic_vector(2 downto 0);
signal three_adder_out : std_logic_vector(2 downto 0);
signal three_bit_mux_out : std_logic_vector(2 downto 0);
signal two_way_4_bit_mux_out : std_logic_vector(3 downto 0);
signal A_mux_out : std_logic_vector(3 downto 0);
signal B_mux_out : std_logic_vector(3 downto 0);
signal four_bit_add_sub_out : std_logic_vector(3 downto 0);
signal reg_bank_out : std_logic_vector(31 downto 0);
signal clock_out: std_logic;
signal c_out: std_logic;
signal overflow_3_bit: std_logic;


begin
```

```
-- port mappings
Instruction_Decoder_0 : Instruction_Decoder
port map(
       Instruction => ins_bus,
       Register_Check_For_Jump => A_mux_out,
       Register_Enable  => register_enable ,
       Immediate_Value => immediate_value,
       Load_Selector => load_selector,
       A_Register_Selector  => A_register_select,
       B_Register_Selector   => B_register_select,
       Abb_Sub_Selector  => add_sub_select,
       Jump_Flag  => jump_flag,
       Jump_Address => adderss_to_jump
     );

twoWay_4bit_Mux_0: twoWay_4bit_Mux
Port map ( D(7 downto 4) => immediate_value,
           D(3 downto 0) => four_bit_add_sub_out,
           S => load_selector,
           Y => two_way_4_bit_mux_out);


Register_bank_0 :Register_bank
Port map ( register_enable => register_enable,
           data=> two_way_4_bit_mux_out,
           clk=>clock_out,
           clr=>Reset,
           reg_out(31 downto 0)=>reg_bank_out(31 downto 0)
            );

four_Bit_Add_Sub_Unit_0 :four_Bit_Add_Sub_Unit
Port map ( A=>A_mux_out,
           B=>B_mux_out,
           M=> add_sub_select,
           S=> four_bit_add_sub_out,
           Zero_flag =>Zero_LED,
           overflow=>overflow_LED,
           C_out=>c_out);

Program_Counter_0: Program_Counter
Port map ( D_in => three_bit_mux_out,
           Push => Reset,
           Clk =>clock_out,
           D_out =>  mem_sel);
```

```
Program_ROM_0:Program_ROM
Port map ( Memory_select =>  mem_sel,
           Instruction_Bus=> ins_bus);

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

twoWay_3bit_Mux_0 : twoWay_3bit_Mux
Port map ( D(5 downto 3) => adderss_to_jump,
           D(2 downto 0) => three_adder_out,
           S => jump_flag,
           Y => three_bit_mux_out);

eightWay_4bit_Mux_A : eightWay_4bit_Mux
 Port map(   D =>reg_bank_out ,
          S => A_register_select,
          Y => A_mux_out
          );

eightWay_4bit_Mux_B : eightWay_4bit_Mux
 Port map( D => reg_bank_out,
          S => B_register_select,
          Y => B_mux_out
          );



Adder_3_Bit_0 : Adder_3_Bit
    Port map ( A => "001",
           B => mem_sel,
           S => three_adder_out,
           C_in => '0',
           overflow => overflow_3_bit
           );

LUT_16_7_0 :LUT_16_7
    Port map (
        address => reg_bank_out(31 downto 28),
        data => seven_segment_in
        );

out_LED <= reg_bank_out(31 downto 28);
Anode_Selector <= "1110"; -- select the right most seven segment display
```

```
end Behavioral;
```

2.2 Elaborated design schematic



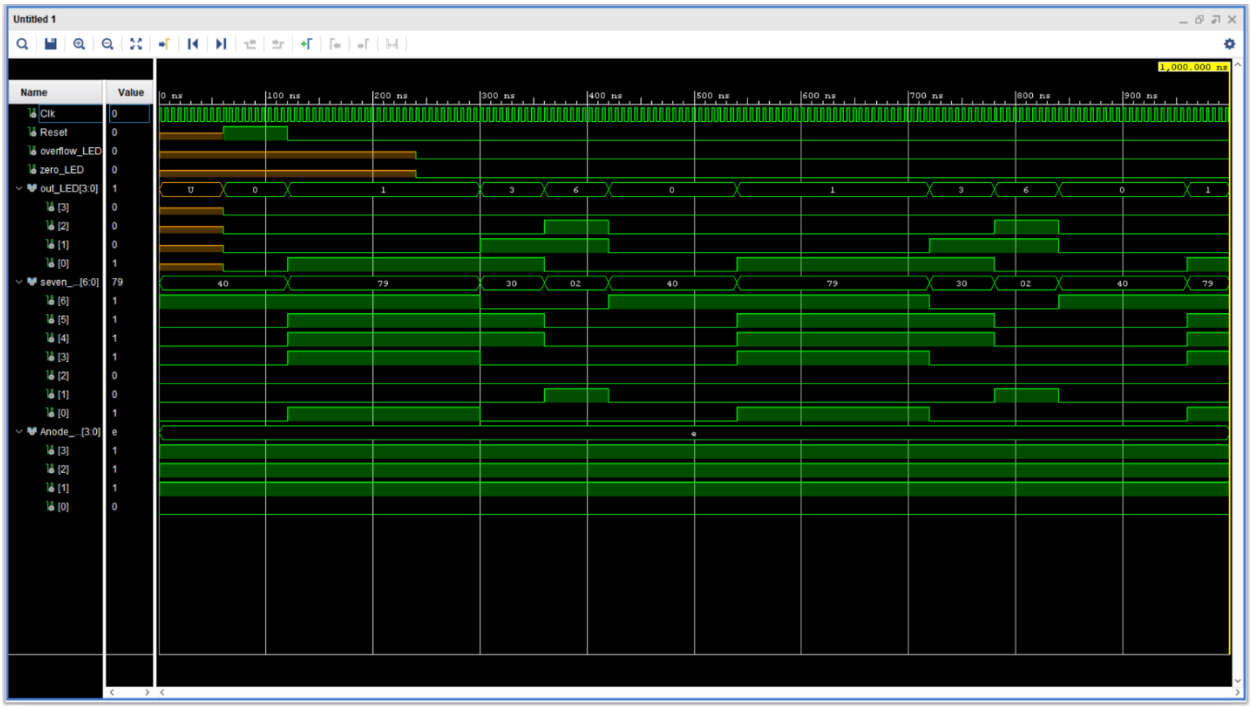
2.3 Simulation source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/10/2024 03:13:54 PM
-- Design Name:
-- Module Name: Nano_Processor_SIM - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
```

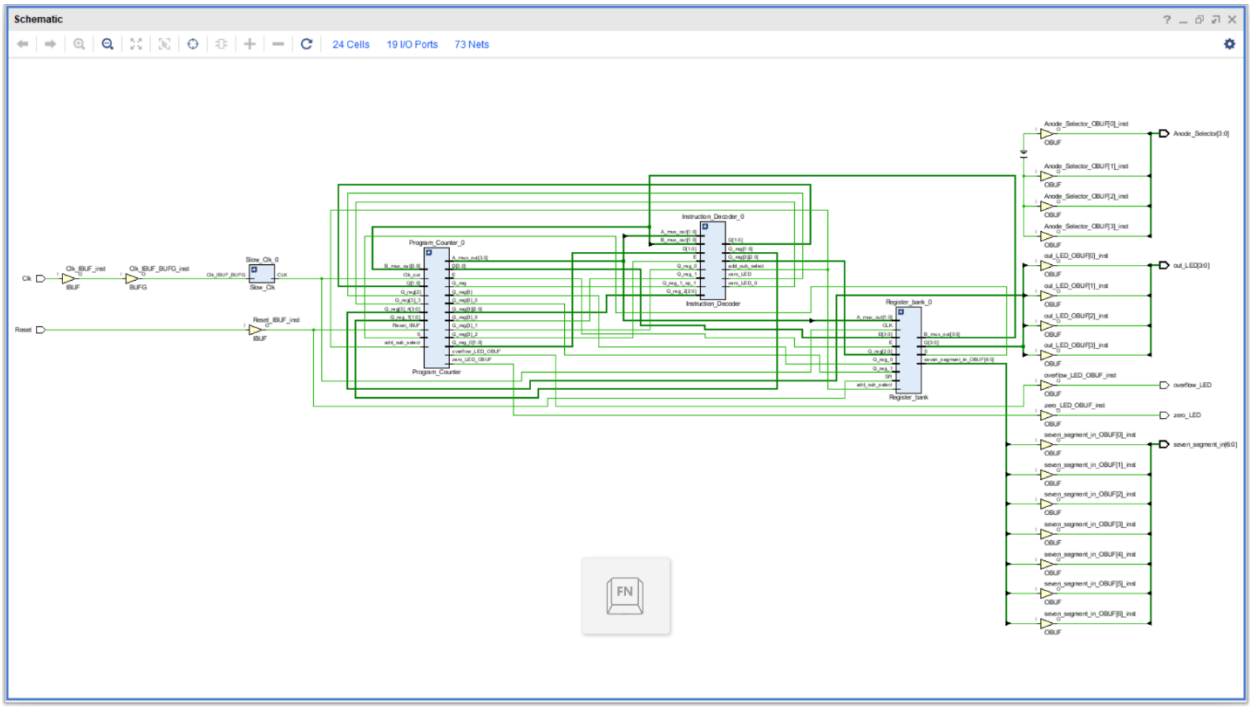
```
-- Additional Comments:
--
----------------------------------------------------------------------------------



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 Nano_Processor_SIM is
--  Port ( );
end Nano_Processor_SIM;

architecture Behavioral of Nano_Processor_SIM is

component Nano_Processor is
    Port ( Clk : in STD_LOGIC;
           Reset : in STD_LOGIC;
           overflow_LED : out STD_LOGIC;
           zero_LED : out STD_LOGIC;
           out_LED : out STD_LOGIC_vector(3 downto 0);
           seven_segment_in : out STD_LOGIC_vector(6 downto 0);
           Anode_Selector : out STD_LOGIC_VECTOR (3 downto 0)
             );
end component;



signal Clk  :  STD_LOGIC;
signal Reset  :  STD_LOGIC;
signal overflow_LED : STD_LOGIC;
signal zero_LED :  STD_LOGIC;
signal out_LED :  STD_LOGIC_vector(3 downto 0);
signal seven_segment_in :  STD_LOGIC_vector(6 downto 0);
signal Anode_Selector :STD_LOGIC_VECTOR (3 downto 0);



begin
```

```
UUT : Nano_Processor
port map(
        Clk =>Clk,
        Reset =>Reset,
        overflow_LED =>overflow_LED,
        zero_LED =>zero_LED ,
        out_LED=>out_LED,
        seven_segment_in =>seven_segment_in,
        Anode_Selector =>Anode_Selector
        );

-- process for the clock
Clock : process begin
 Clk<='1';
 wait for 3 ns;
 Clk<='0';
 wait for 3 ns;
 end process;

process begin
--process of the processor(reset)
 wait for 60 ns;
 Reset <= '1' ;
 wait for 60 ns;
 Reset <= '0';
 wait;
 end process;


end Behavioral;
```

2.4 Timing diagram



2.5 Implemented Design Schematic



3.Component of the Main Module

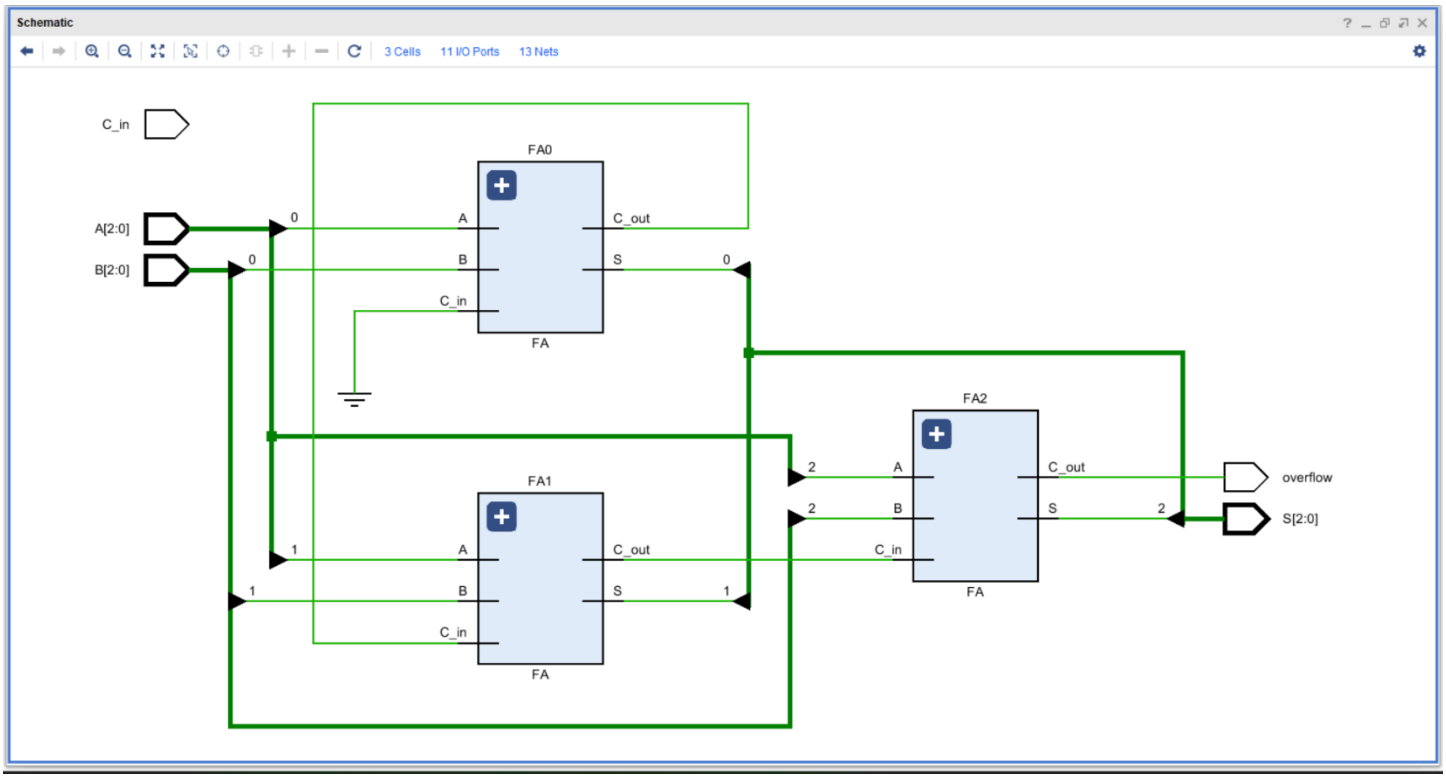
3.1 Three-bit Adder

3.1.1 Design source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/09/2024 02:25:46 PM
-- Design Name:
-- Module Name: 3_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 Adder_3_Bit is
    Port ( A : in STD_LOGIC_VECTOR (2 downto 0);
           B : in STD_LOGIC_VECTOR (2 downto 0);
           S : out STD_LOGIC_VECTOR (2 downto 0);
           C_in  : in STD_LOGIC ;
           overflow : out STD_LOGIC);
end Adder_3_Bit;
```

```
architecture Behavioral of Adder_3_Bit is
component FA
    Port (
        A : in STD_LOGIC;
        B : in STD_LOGIC;
        C_in : in STD_LOGIC;
        S : out STD_LOGIC;
        C_out : out STD_LOGIC);
end component;

signal C0, C1 : STD_LOGIC;

begin

    FA0 : FA
        port map (
            A => A(0),
            B => B(0),
            C_in => '0',
            S => S(0),
            C_out => C0
        );

    FA1 : FA
        port map (
            A => A(1),
            B => B(1),
            C_in => C0,
            S => S(1),
            C_out => C1
        );

    FA2 : FA
        port map (
            A => A(2),
            B => B(2),
            C_in => C1,
            S => S(2),
            C_out => overflow

        );


end Behavioral;
```

3.1.2 Elaborated design schematic



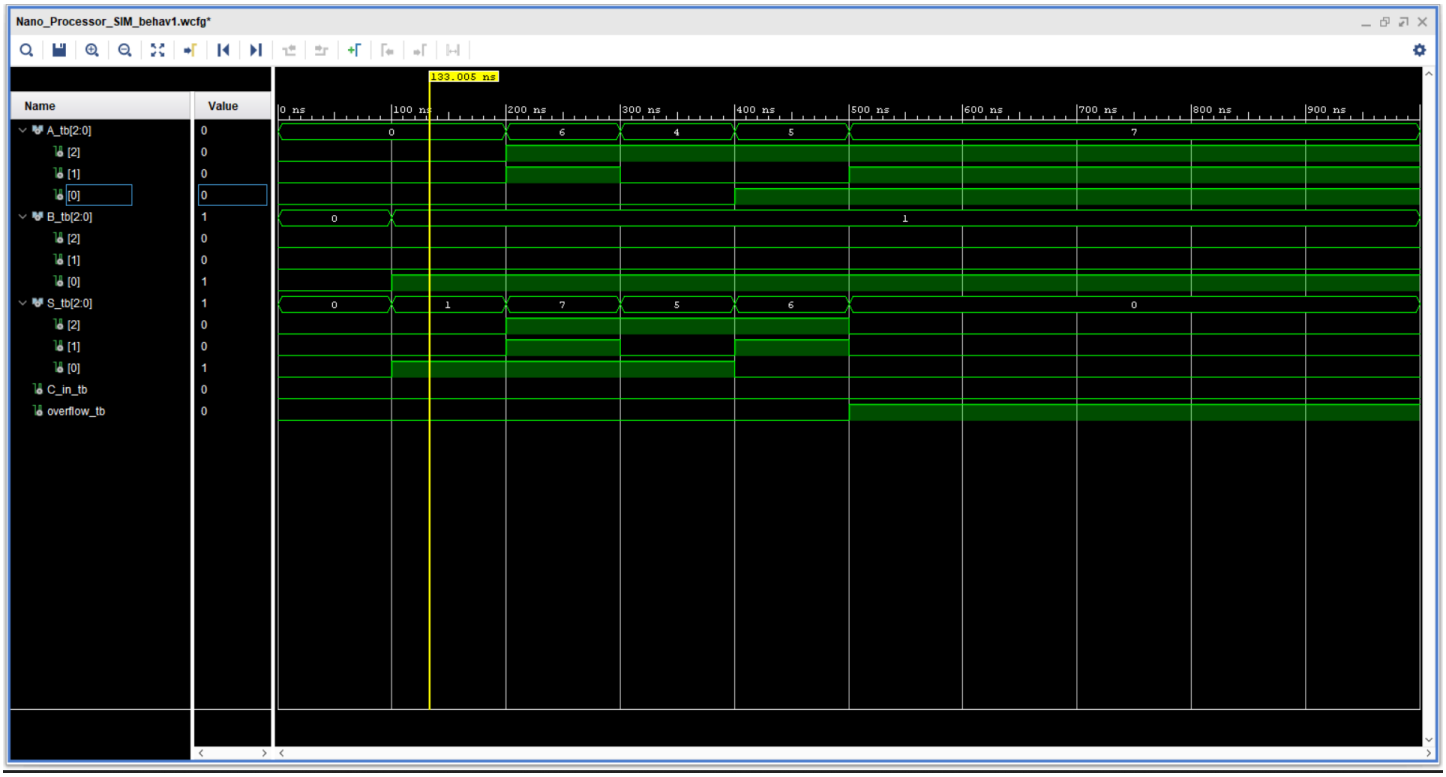
3.1.3 Simulation source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/09/2024 03:12:25 PM
-- Design Name:
-- Module Name: TB_Adder_3_Bit - 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 TB_Adder_3_Bit is
--  Port ( );
end TB_Adder_3_Bit;

architecture Behavioral of TB_Adder_3_Bit is
    component Adder_3_Bit
        PORT (
            A : in STD_LOGIC_VECTOR (2 downto 0);
            B : in STD_LOGIC_VECTOR (2 downto 0);
            S : out STD_LOGIC_VECTOR (2 downto 0);
            C_in : in STD_LOGIC;
            overflow : out STD_LOGIC
        );
    end component;

    SIGNAL A_tb, B_tb, S_tb : STD_LOGIC_VECTOR (2 downto 0);
    SIGNAL C_in_tb, overflow_tb : STD_LOGIC;

begin
    UUT: Adder_3_Bit PORT MAP (
        A => A_tb,
        B => B_tb,
        C_in => C_in_tb,
        S => S_tb,
        overflow => overflow_tb
    );

    process
    begin
        A_tb <= "000";
        B_tb <= "000";
        C_in_tb <= '0';
```

```
        WAIT FOR 100 ns;
        A_tb <= "000";
        B_tb <= "001";
        WAIT FOR 100 ns;
        A_tb <= "110";
        B_tb <= "001";
        WAIT FOR 100 ns;
        A_tb <= "100";
        B_tb <= "001";
        WAIT FOR 100 ns;
        A_tb <= "101";
        B_tb <= "001";
        WAIT FOR 100 ns;
        A_tb <= "111";
        B_tb <= "001";
        WAIT;
    end process;
end Behavioral;
```

3.1.4 Timing diagram



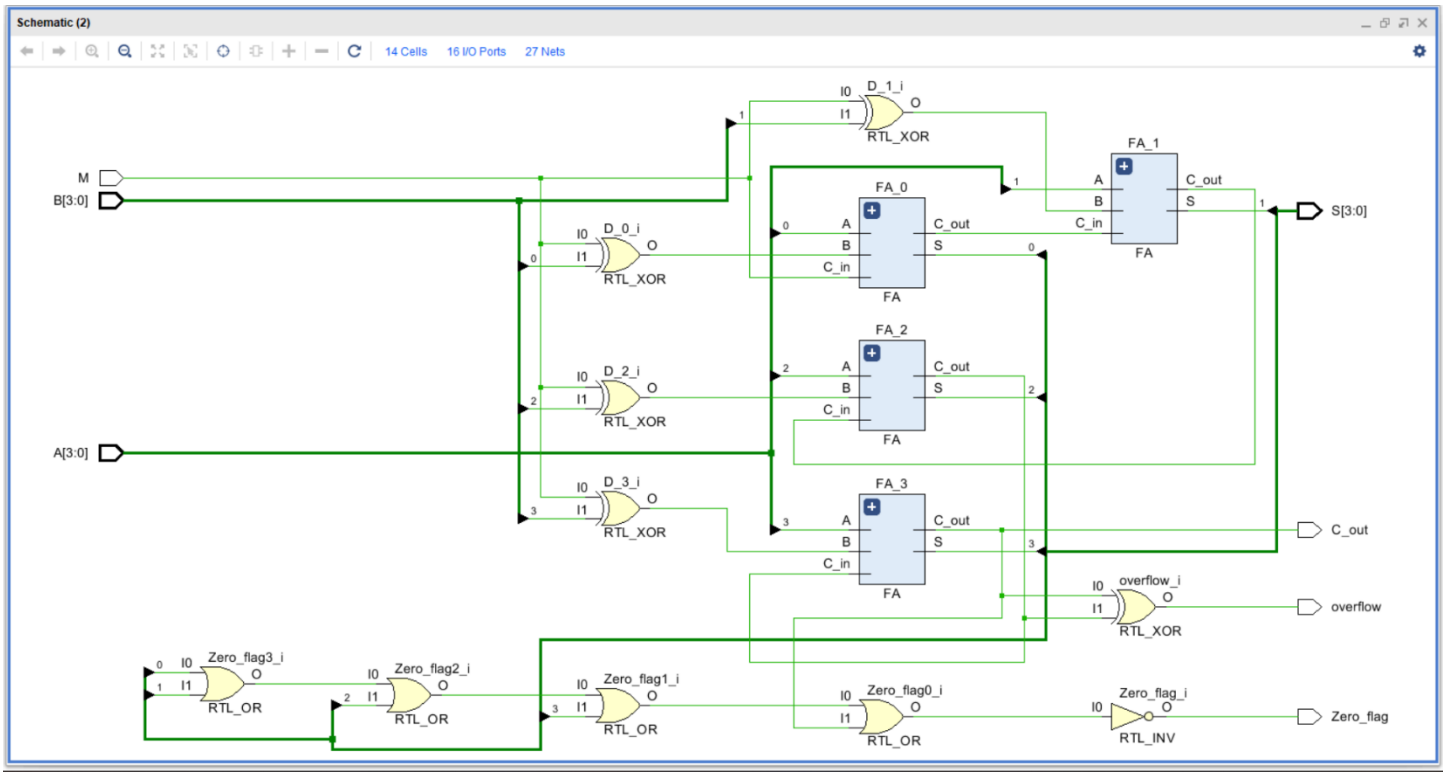
3.2 Four-bit Addition Subtraction Unit

3.2.1 Design source file

```
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 four_Bit_Add_Sub_Unit 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);
           overflow  : out STD_LOGIC;
           Zero_flag : out STD_LOGIC;
           C_out : out STD_LOGIC);

end four_Bit_Add_Sub_Unit;

architecture Behavioral of four_Bit_Add_Sub_Unit is
component FA
PORT(
    A : in STD_LOGIC;
    B : in STD_LOGIC;
    C_in : in STD_LOGIC;
    S : out STD_LOGIC;
    C_out : out STD_LOGIC);
end component;

SIGNAL  FA0_C:STD_LOGIC;
SIGNAL  FA1_C:STD_LOGIC;
SIGNAL  FA2_C:STD_LOGIC;
SIGNAL  C_final:STD_LOGIC;
SIGNAL D,S_out:STD_LOGIC_VECTOR(3 downto 0);
begin

FA_0 : FA
PORT MAP (
```

```
    A=>A(0),
    B => D(0),
    C_in => M,
    S => S_out(0),
    C_out=> FA0_C
    );
FA_1 : FA
PORT MAP (
    A=>A(1),
    B => D(1),
    C_in => FA0_C,
    S => S_out(1),
    C_out=> FA1_C
    );
FA_2 : FA
PORT MAP (
    A=>A(2),
    B => D(2),
    C_in => FA1_C,
    S => S_out(2),
    C_out=> FA2_C
    );
FA_3 : FA
PORT MAP (
    A=>A(3),
    B => D(3),
    C_in => FA2_C,
    S => S_out(3),
    C_out=> C_final
    );
C_out<= C_final;
S<=S_out;
D(0)<= M XOR B(0);
D(1)<= M XOR B(1);
D(2)<= M XOR B(2);
D(3)<= M XOR B(3);

Zero_flag<=NOT(S_out(0)OR S_out(1)OR S_out(2)OR S_out(3)OR C_final);
overflow<= C_final XOR FA2_C;

end Behavioral;
```

3.2.2 Elaborated design schematic



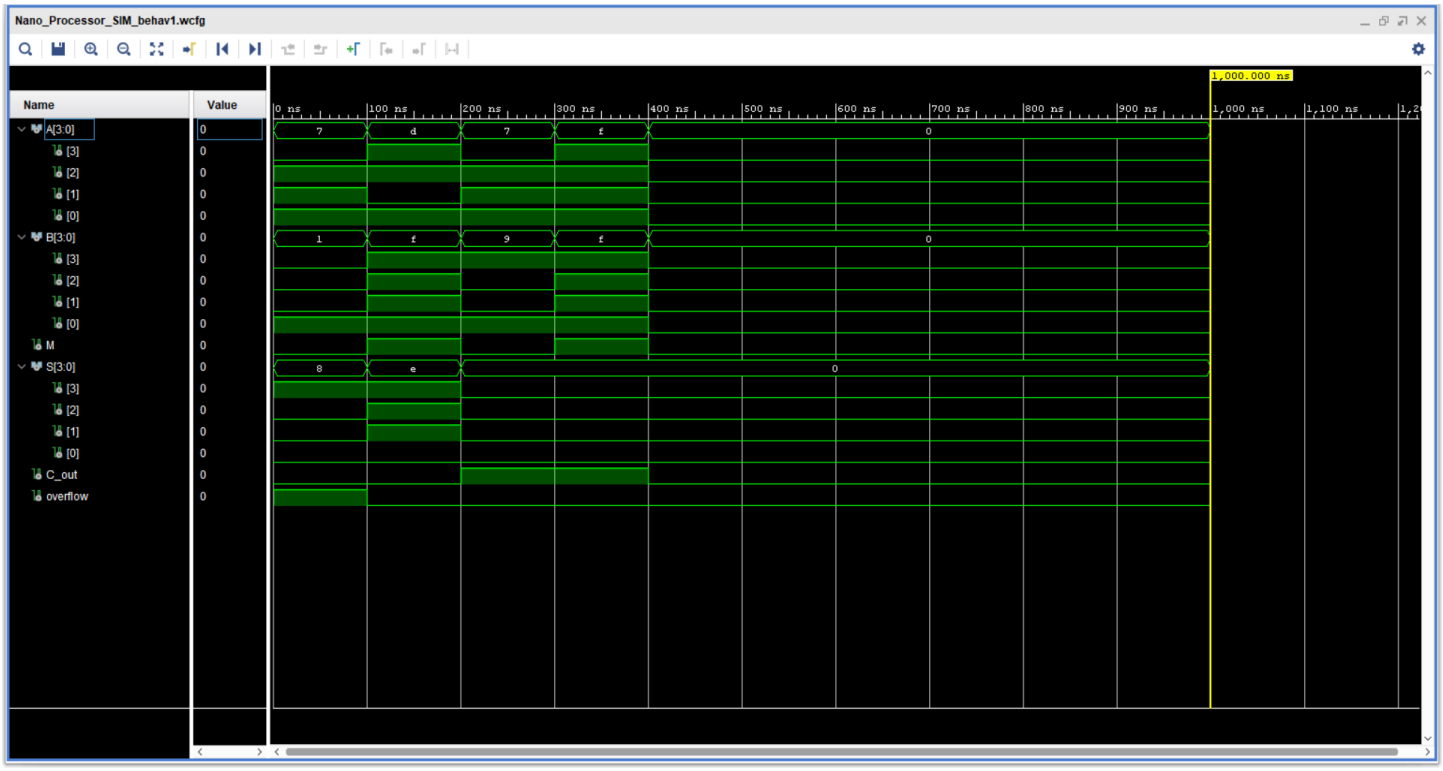
3.2.3 Simulation source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/11/2024 07:14:07 PM
-- Design Name:
-- Module Name: TB_four_Bit_Add_sub_Unit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------
```

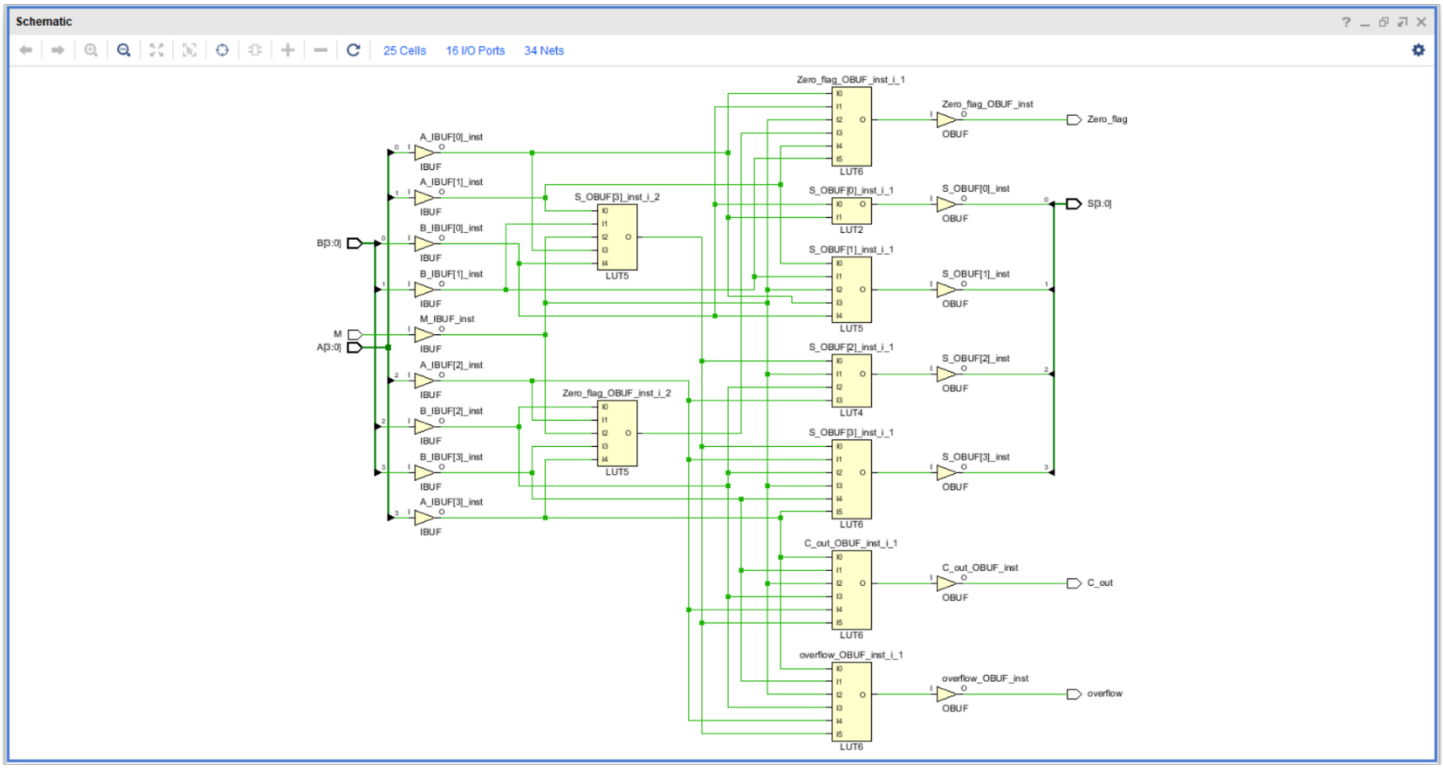
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- 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 TB_four_Bit_Add_sub_Unit is
--  Port ( );
end TB_four_Bit_Add_sub_Unit;

architecture Behavioral of TB_four_Bit_Add_sub_Unit is
component four_Bit_Add_sub_Unit
    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);
           C_Out : out STD_LOGIC;
           overflow : out STD_LOGIC);
end component;

signal A, B, S : STD_LOGIC_VECTOR (3 downto 0);
signal M, C_out,overflow: STD_LOGIC;

begin
UUT: four_Bit_Add_sub_Unit
 PORT MAP(
        A => A,
        B => B,
        M => M,
        S => S,
        C_out => c_out,
        overflow=>overflow
    );

    process
    begin

    M <= '0';
```

```
    A <= "0111";
    B <= "0001";
    wait for 100ns;

    M <= '1';
    A <= "1101";
    B <= "1111";
    wait for 100ns;

    M <= '0';
    A <= "0111";
    B <= "1001";
    wait for 100ns;

     M <= '1';
     A <= "1111";
     B <= "1111";
     wait for 100ns;


     M <= '0';
     A <= "0000";
     B <= "0000";
     wait;

end process;

end Behavioral;
```

3.2.4 Timing diagram



3.2.5 Implemented Design Schematic



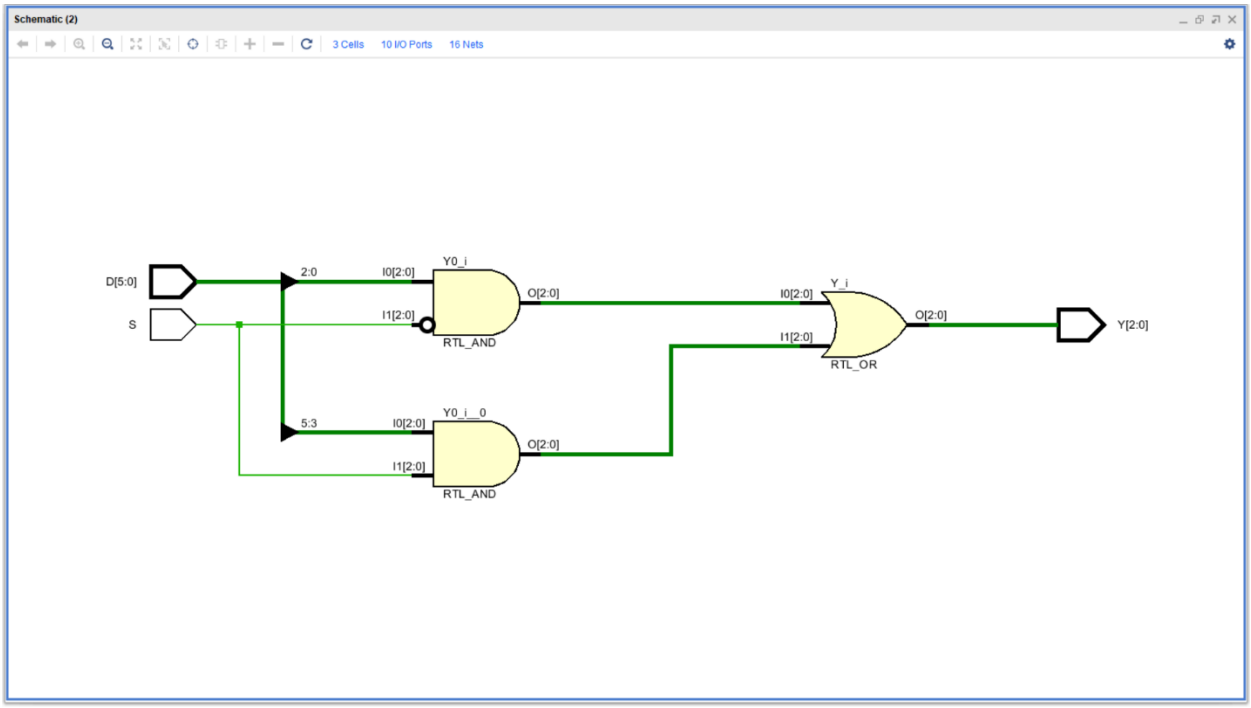
3.3 Two-way three-bit Multiplexer

3.3.1 Design source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/08/2024 10:58:10 PM
-- Design Name:
-- Module Name: twoWay_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 twoWay_3bit_Mux is
Port ( D : in STD_LOGIC_VECTOR (5 downto 0);-- input
       S : in STD_LOGIC;-- switch
       Y : out STD_LOGIC_VECTOR (2 downto 0));--output
end twoWay_3bit_Mux;
architecture Behavioral of twoWay_3bit_Mux is
signal S_vec : STD_LOGIC_VECTOR(2 downto 0);--to create a vector

begin

S_vec <= S&S&S;--create a vector using std logic (s)
```

```
Y <= (D(2 downto 0) AND (NOT S_vec)) OR (D(5 downto 3) AND S_vec);-- logic for the
multiplexer

end Behavioral;
```

3.3.2 Elaborated design schematic



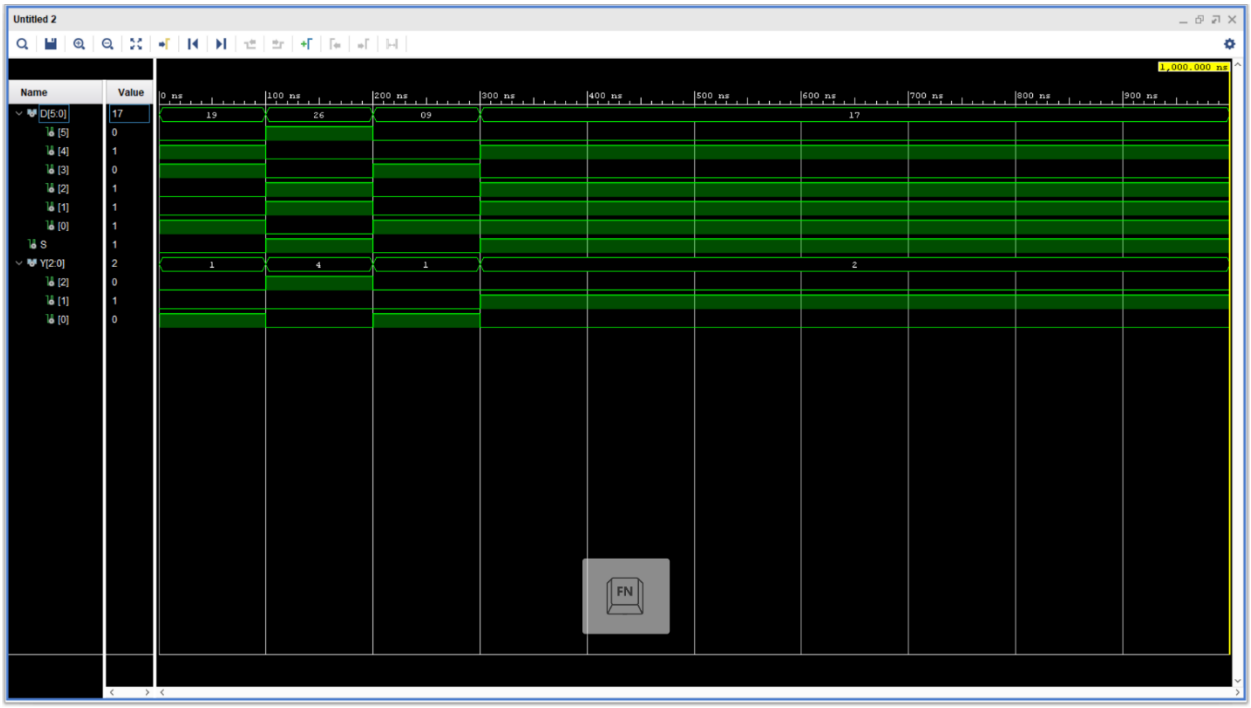
3.3.3 Simulation source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/08/2024 11:00:36 PM
-- Design Name:
-- Module Name: twoWay_3bit_Mux_SIM - 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 twoWay_3bit_Mux_SIM is
--  Port ( );
end twoWay_3bit_Mux_SIM;

architecture Behavioral of twoWay_3bit_Mux_SIM is
component twoWay_3bit_Mux is
Port ( D : in STD_LOGIC_VECTOR (5 downto 0);
       S : in STD_LOGIC;
       Y : out STD_LOGIC_VECTOR (2 downto 0));
end component;

signal D : STD_LOGIC_VECTOR (5 downto 0);
signal S : STD_LOGIC;
signal Y : STD_LOGIC_VECTOR (2 downto 0);

begin
UUT: twoWay_3bit_Mux PORT MAP(
    D => D,
    S => S,
    Y => Y
);

process begin

    D <= "011001";--selector
    s <= '0';--switch
    wait for 100ns;
    D <= "100110";
    s <= '1';
    wait for 100ns;
```

```
    D <= "001001";
    s <= '0';
    wait for 100ns;
    D <= "010111";
    s <= '1';
    wait;

end process;
```

3.3.4 Timing diagram



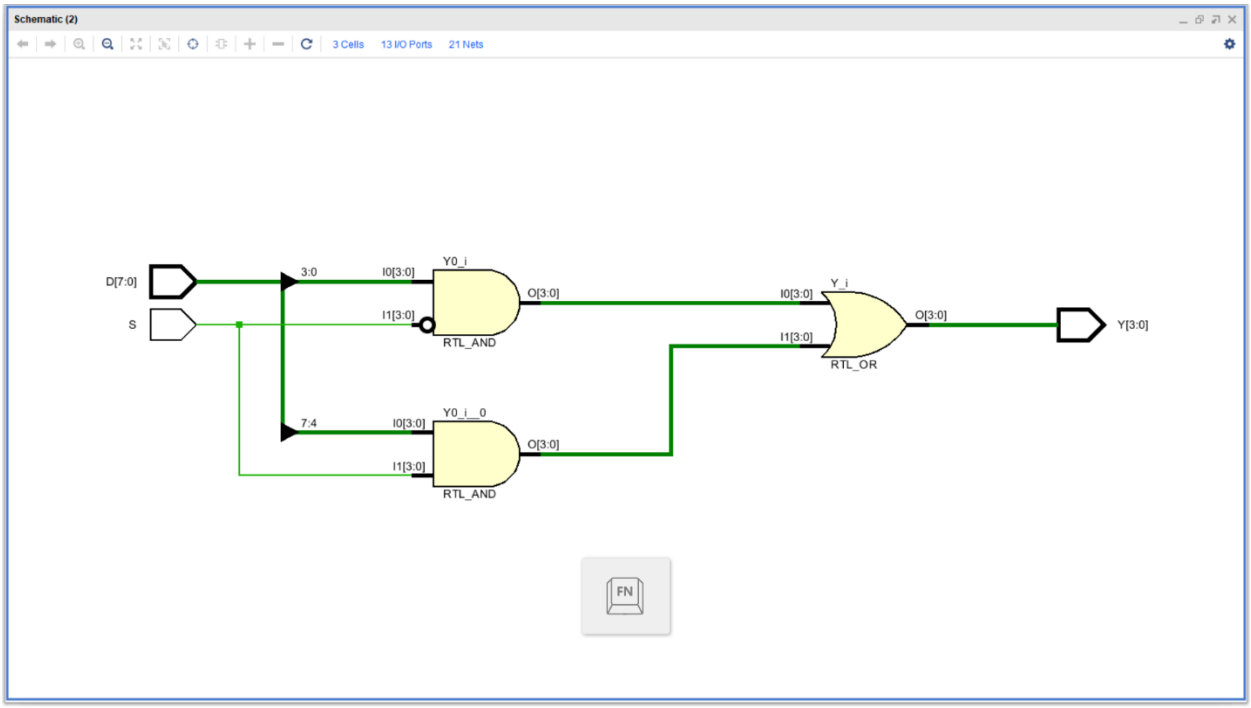
3.4 Two way four-bit Multiplexer

3.4.1 Design source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/08/2024 07:45:11 AM
-- Design Name:
-- Module Name: 2-way 4-bit 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 twoWay_4bit_Mux is
    Port ( D : in STD_LOGIC_VECTOR (7 downto 0);--input for the mux
           S : in STD_LOGIC;--swich for the mux
           Y : out STD_LOGIC_VECTOR (3 downto 0));--output of the mux
end twoWay_4bit_Mux;

architecture Behavioral of twoWay_4bit_Mux is

signal S_vec : STD_LOGIC_VECTOR(3 downto 0);-- use to create a vector

begin

S_vec <= S & S & S & S;-- create a vector using std logic(s)
Y <= (D(3 downto 0) AND (NOT S_vec)) OR (D(7 downto 4) AND S_vec) ;--logic for the mux
end Behavioral;
```

3.4.2 Elaborated design schematic



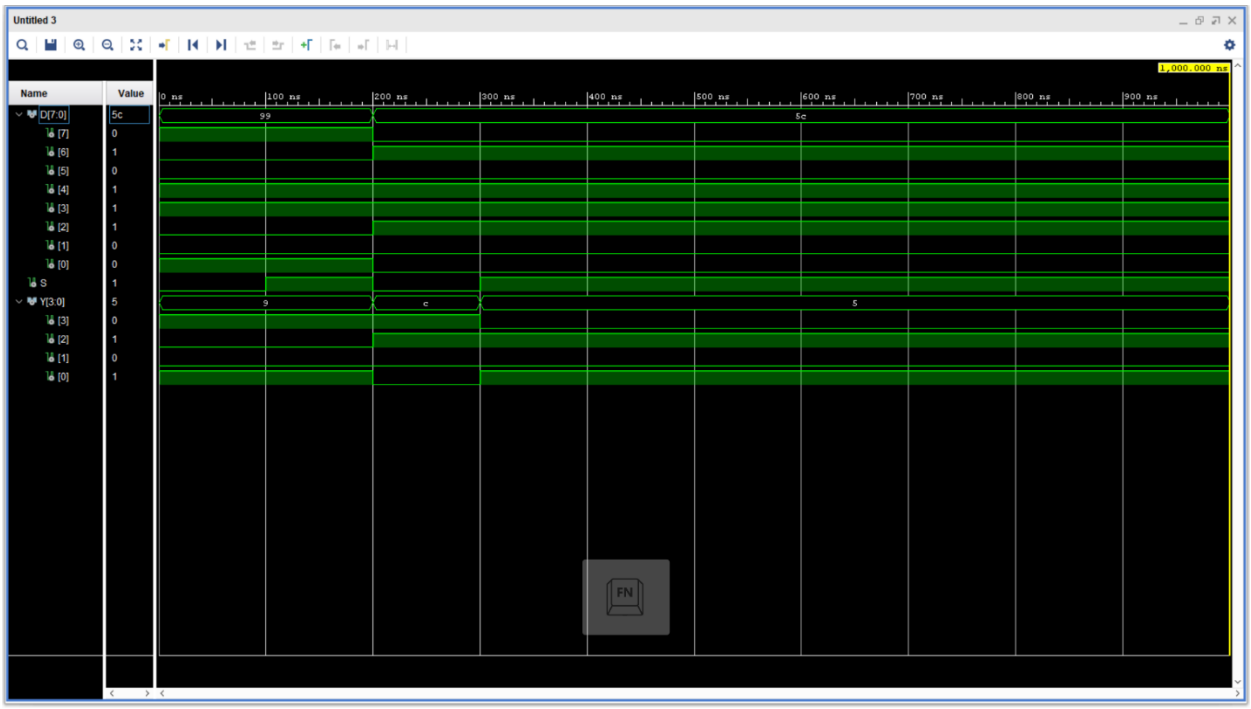
3.4.3 Simulation source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/08/2024 08:01:26 AM
-- Design Name:
-- Module Name: twoWay_4bit_Mux_SIM - 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 twoWay_4bit_Mux_SIM is
--  Port ( );
end twoWay_4bit_Mux_SIM;

architecture Behavioral of twoWay_4bit_Mux_SIM is
component twoWay_4bit_Mux is
    Port ( D : in STD_LOGIC_VECTOR (7 downto 0);
           S : in STD_LOGIC;
           Y : out STD_LOGIC_VECTOR (3 downto 0));
end component;

signal D : STD_LOGIC_VECTOR (7 downto 0);
signal S : STD_LOGIC;
signal Y : STD_LOGIC_VECTOR (3 downto 0);

begin
UUT: twoWay_4bit_Mux PORT MAP(
    D => D,
    S => S,
    Y => Y
);

process begin

    D <= "10011001";--input
    s <= '0';--selector
    wait for 100ns;
    D <= "10011001";
    s <= '1';
    wait for 100ns;
    D <= "01011100";
    s <= '0';
    wait for 100ns;
    D <= "01011100";
```

```
    s <= '1';
    wait;

end process;

end Behavioral;
```

3.4.4 Timing diagram



3.5 Eight-way four-bit Multiplexer

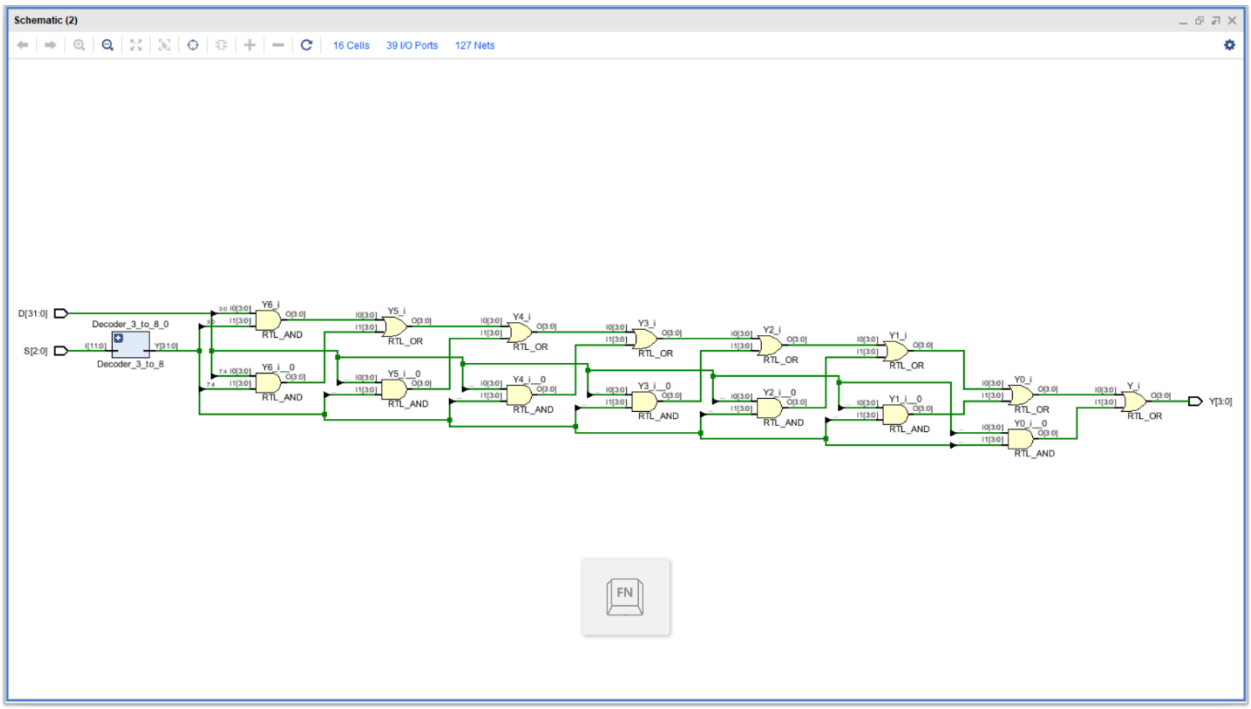
3.5.1 Design source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/08/2024 05:53:03 PM
-- Design Name:
-- Module Name: eightWay_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 eightWay_4bit_Mux is
    Port ( D : in STD_LOGIC_VECTOR (31 downto 0);--input
    -- I used a 32 bit bus for the simplicity
           S : in STD_LOGIC_VECTOR (2 downto 0);--swich
           Y : out STD_LOGIC_VECTOR (3 downto 0)--output
           );
end eightWay_4bit_Mux;

architecture Behavioral of eightWay_4bit_Mux is

-- used a decoder to make the multiplexer
component Decoder_3_to_8
    port(
    I: in STD_LOGIC_VECTOR;
    Y: out STD_LOGIC_VECTOR );
    end component;

signal Yd : STD_LOGIC_VECTOR (31 downto 0);--to get the output of the decoder
signal S_vec : STD_LOGIC_VECTOR (11 downto 0);-- to make a vector

begin

Decoder_3_to_8_0 : Decoder_3_to_8
     port map(
     I =>S_vec ,
     Y => Yd );

S_vec <= S(2)&S(2)&S(2)&S(2)&S(1)&S(1)&S(1)&S(1)&S(0)&S(0)&S(0)&S(0);-- create a vector from
std logic(s)
```

```
    --logic for the multiplexer
    Y <=            ( (D(3 downto 0) AND Yd(3 downto 0))OR
                    (D(7 downto 4) AND Yd(7 downto 4))OR
                    (D(11 downto 8) AND Yd(11 downto 8))OR
                    (D(15 downto 12) AND Yd(15 downto 12))OR
                    (D(19 downto 16) AND Yd(19 downto 16))OR
                    (D(23 downto 20) AND Yd(23 downto 20))OR
                    (D(27 downto 24) AND Yd(27 downto 24))OR
                    (D(31 downto 28) AND Yd(31 downto 28)));

end Behavioral;
```

3.5.2 Elaborated design schematic



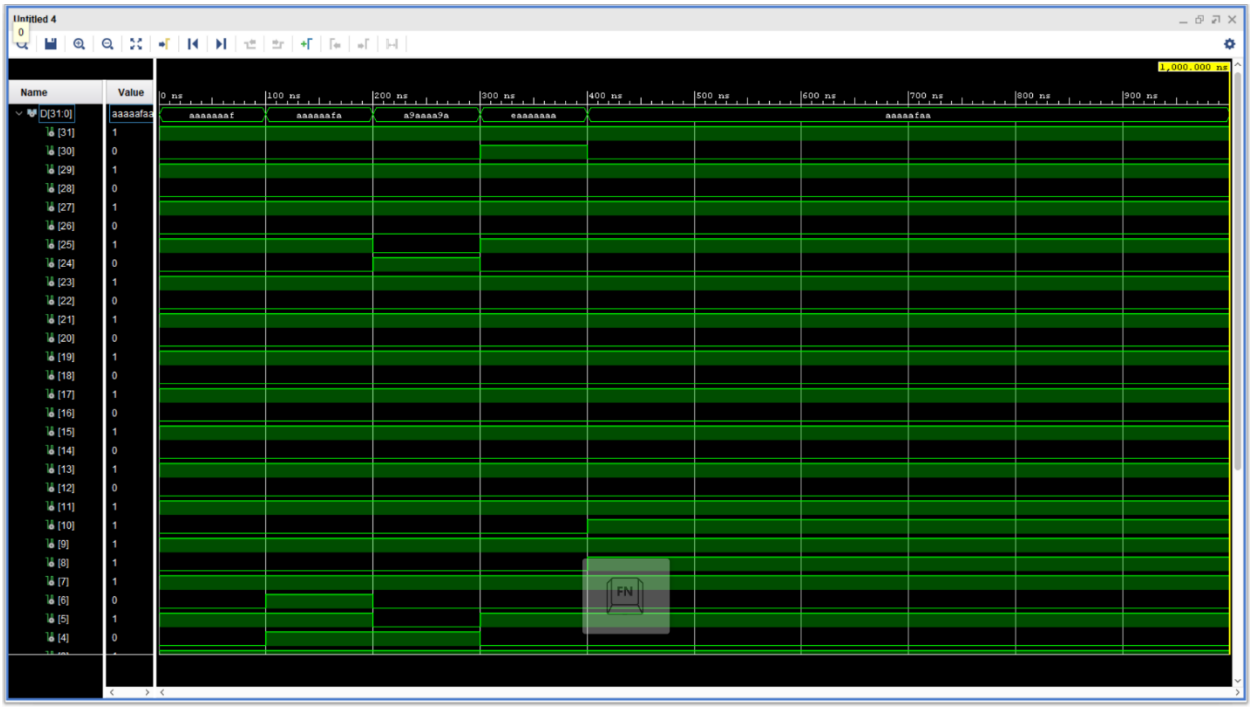
3.5.3 Simulation source file

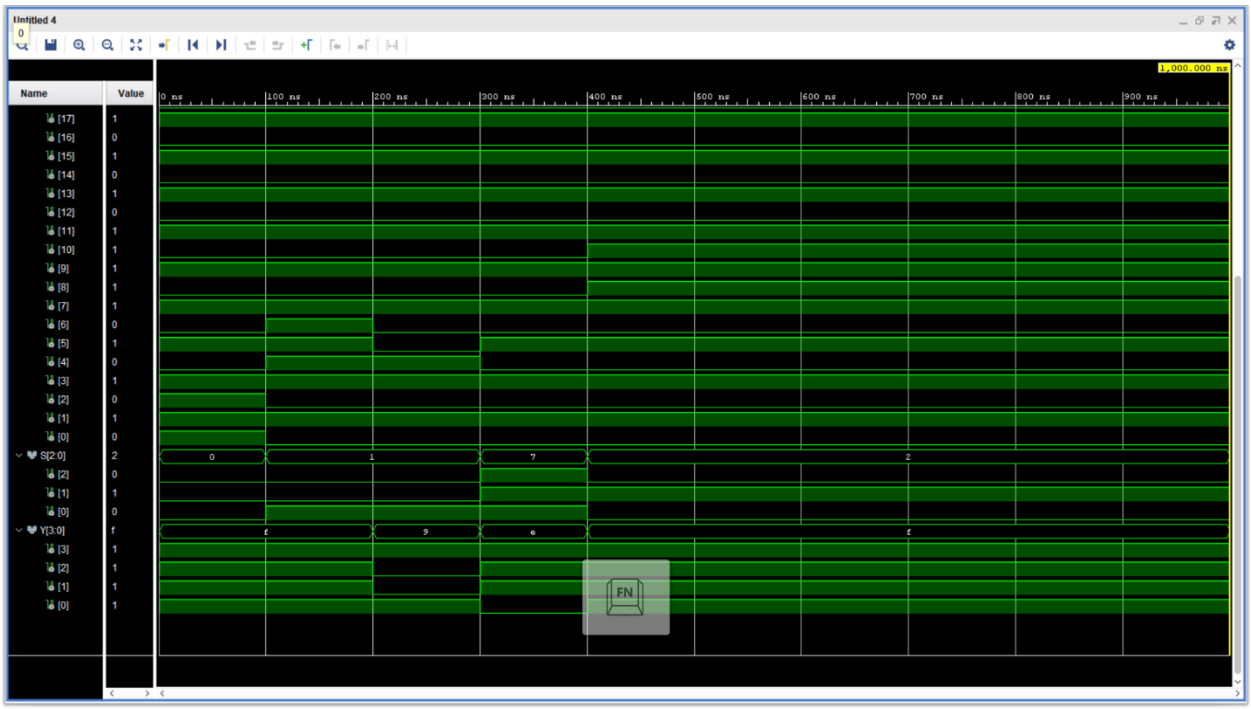
```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/08/2024 06:06:08 PM
-- Design Name:
-- Module Name: eightWay_4bit_Mux_SIM - Behavioral
```

```
-- 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 eightWay_4bit_Mux_SIM is
--  Port ( );
end eightWay_4bit_Mux_SIM;

architecture Behavioral of eightWay_4bit_Mux_SIM is
component eightWay_4bit_Mux is
    Port ( D : in STD_LOGIC_VECTOR (31 downto 0);
           S : in STD_LOGIC_VECTOR (2 downto 0);
           Y : out STD_LOGIC_VECTOR (3 downto 0));
end component;
signal D : STD_LOGIC_VECTOR (31 downto 0);
signal S : STD_LOGIC_VECTOR (2 downto 0);
signal Y : STD_LOGIC_VECTOR (3 downto 0);

begin
UUT : eightWay_4bit_Mux port map(
    D => D,
    S => S,
    Y => Y
);
process begin
    -- input
    D <= "10101010101010101010101010101111";
    -- selector
```

```
    S <= "000";
    wait for 100 ns;
    --input
    D <= "10101010101010101010101011111010";
    --selector
    S <= "001";
    wait for 100 ns;
    --input
    D <= "10101001101010101010101010011010";
    --selector
    S <= "001";
    wait for 100 ns;
    --input
    D <= "11101010101010101010101010101010";
    --selector
    S <= "111";
    wait for 100 ns;
    --input
    D <= "10101010101010101010111110101010";
    --selector
    S <= "010";
    wait;
end process;
end Behavioral;
```

3.5.4 Timing diagram





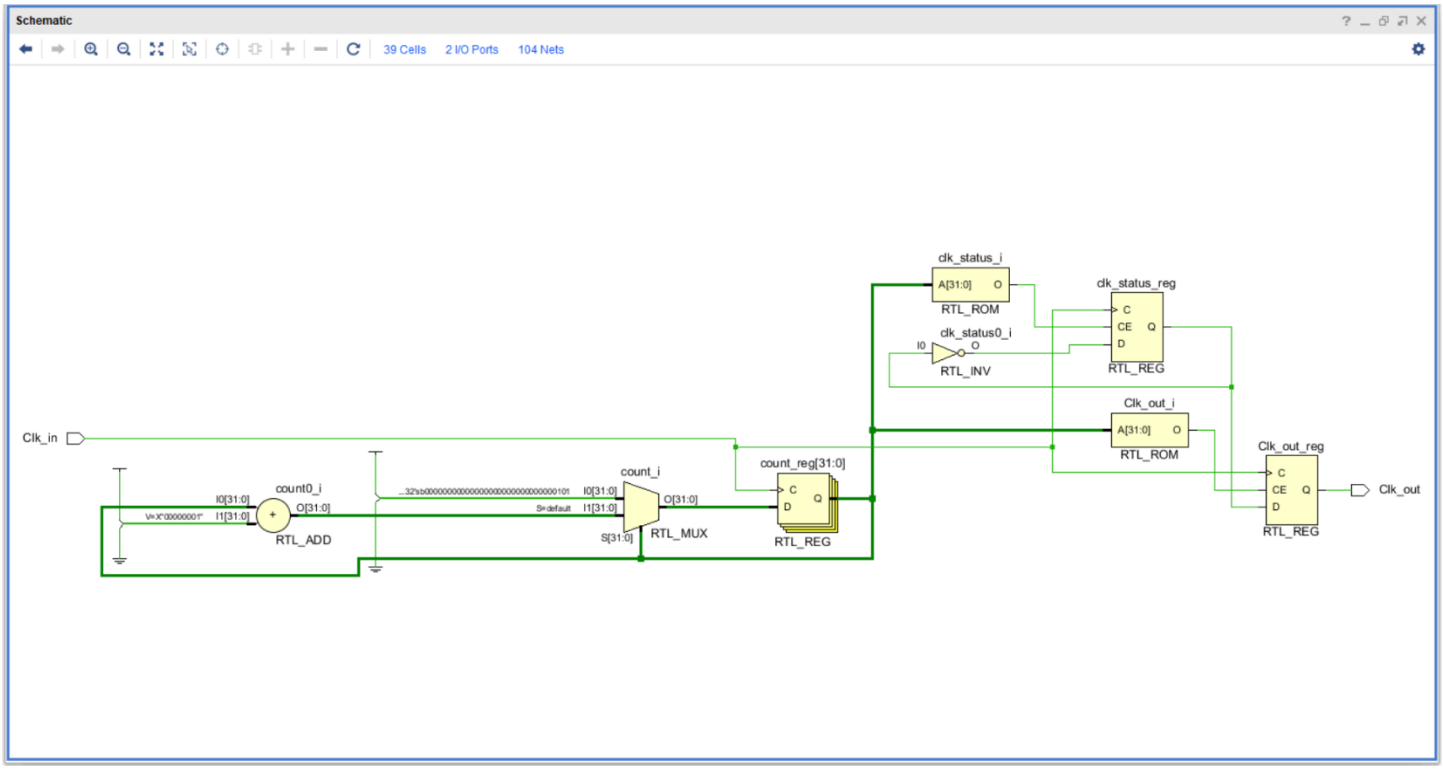
3.6 Slow clock

3.6.1 Design source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 03/05/2024 03:39:23 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

    process (Clk_in) begin
        if (rising_edge(Clk_in)) then
            count <= count + 1;
            if (count =100000000) then
                clk_status <= not clk_status;
                Clk_out <= clk_status;
                count <= 1;
            end if;
        end if;
    end process;

end Behavioral;
```

3.6.2 Elaborated design schematic

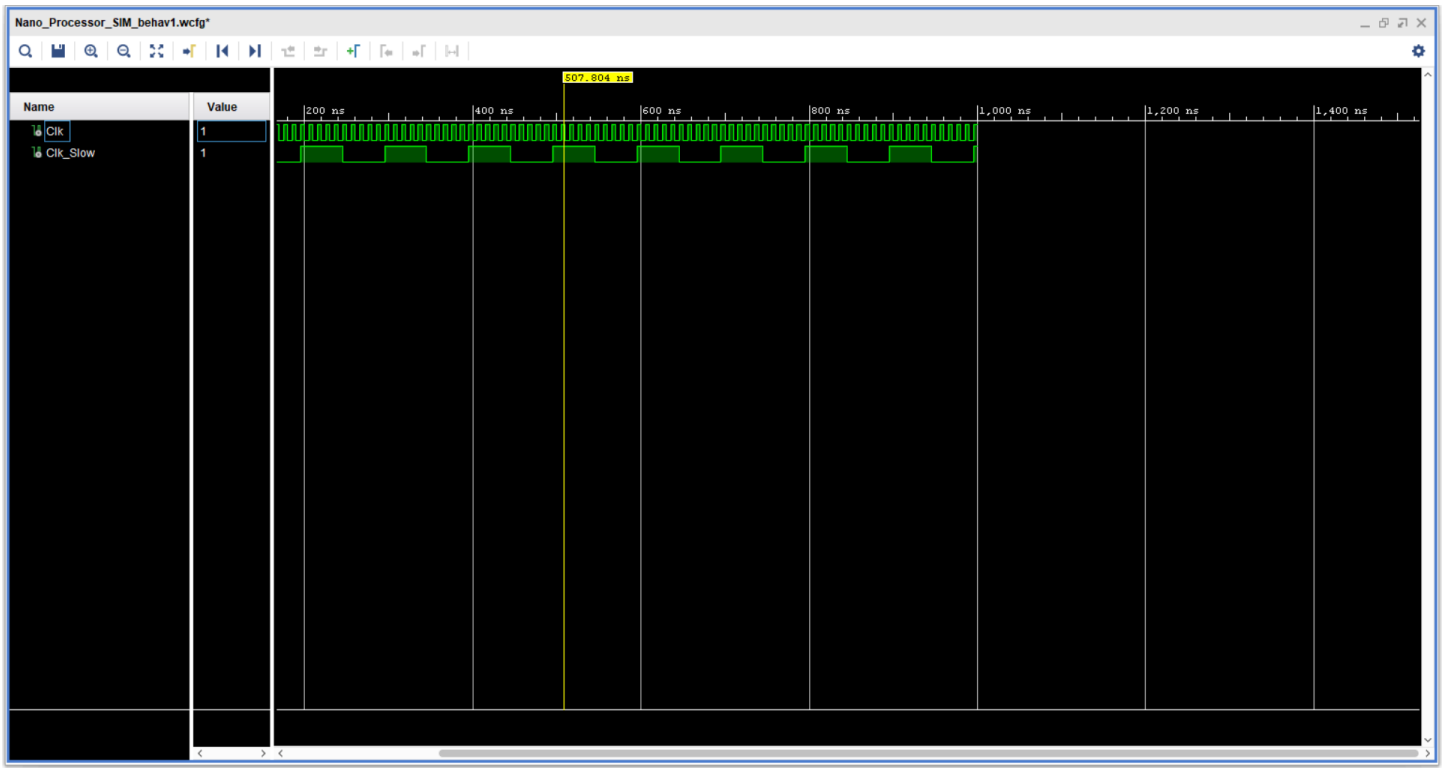


3.6.3 Simulation source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/11/2024 02:27:58 AM
-- Design Name:
-- Module Name: TB_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 TB_Slow_Clk is
--  Port ( );
end TB_Slow_Clk;

architecture Behavioral of TB_Slow_Clk is
    component Slow_Clk
    port( Clk_in : in STD_LOGIC;
          Clk_out : out STD_LOGIC);
    end component;
    signal Clk: STD_LOGIC :='0';
    signal Clk_Slow: STD_LOGIC;



    begin

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

    process begin

    wait for 5ns;
    Clk <= NOT Clk;

    end process;


end Behavioral;
```

3.6.4 Timing diagram



3.7 Program counter

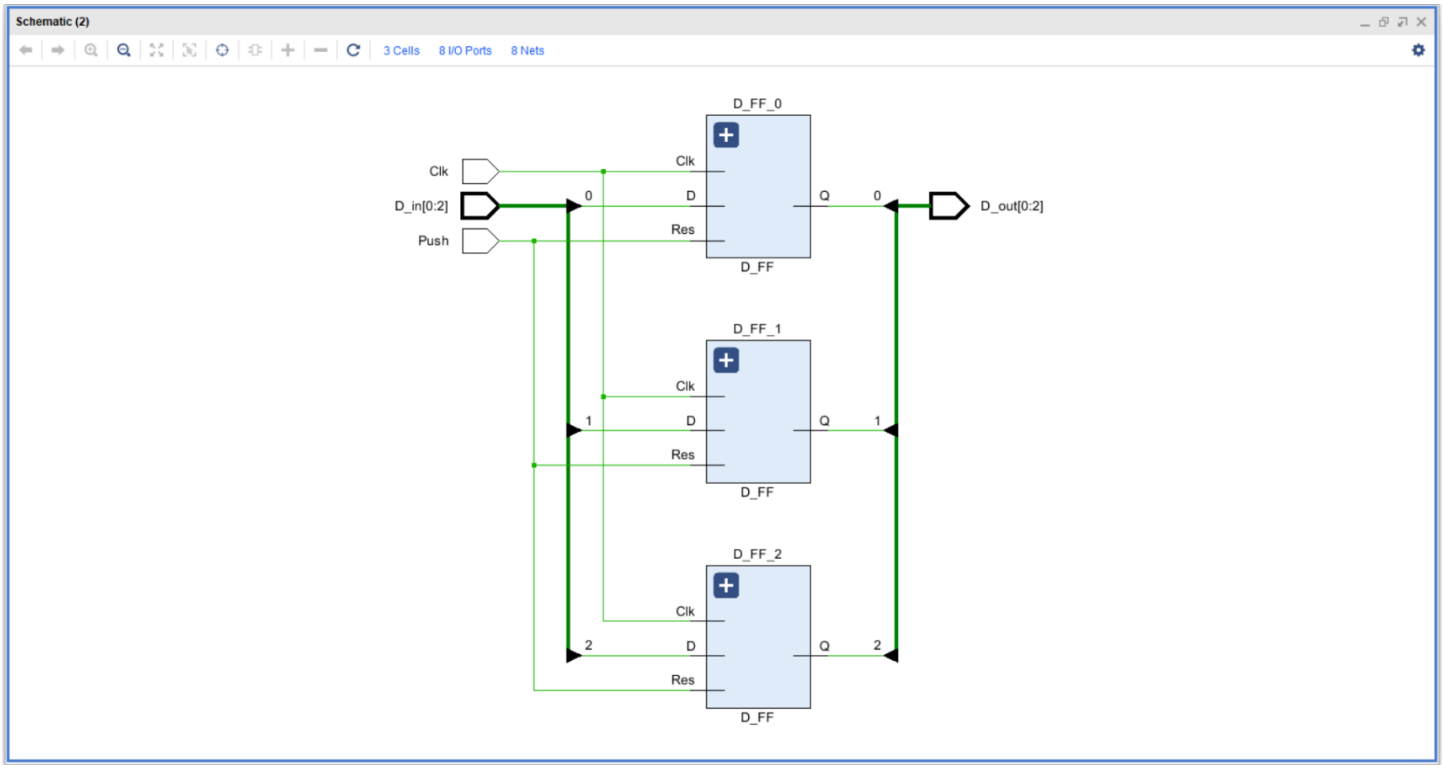
3.7.1 Design source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/08/2024 08:02:45 PM
-- Design Name:
-- Module Name: Program_Counter - 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_Counter is
    Port ( D_in : in STD_LOGIC_VECTOR (0 to 2);
           Push : in STD_LOGIC; -- reset button
           Clk : in STD_LOGIC;
           D_out : out STD_LOGIC_VECTOR (0 to 2));
end Program_Counter;

architecture Behavioral of Program_Counter is
component D_FF
Port ( D : in STD_LOGIC;
       Res : in STD_LOGIC;
       Clk : in STD_LOGIC;
       Q : out STD_LOGIC;
       Qbar : out STD_LOGIC);
end component;
 begin
D_FF_0 : D_FF
PORT MAP(
  D=>D_in(0),
  Res=>Push,
  Clk=>Clk,
  Q=>D_out(0)
);
D_FF_1 : D_FF
PORT MAP(
  D=>D_in(1),
  Res=>Push,
  Clk=>Clk,
  Q=>D_out(1)
);
D_FF_2 : D_FF
PORT MAP(
  D=>D_in(2),
  Res=>Push,
```

```
  Clk=>Clk,
  Q=>D_out(2)
);
end Behavioral;
```

3.7.2 Elaborated design schematic



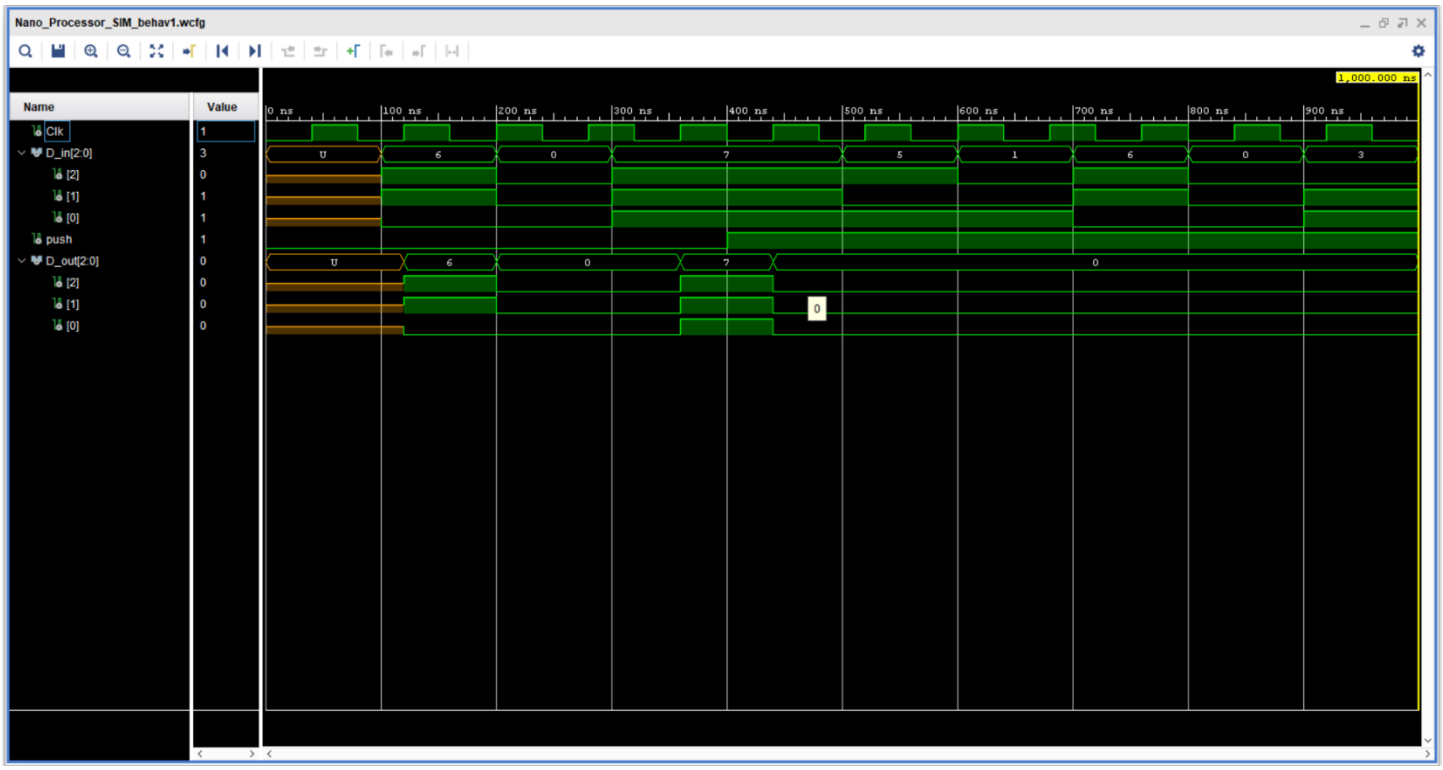
3.7.3 Simulation source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/21/2024 09:59:07 AM
-- Design Name:
-- Module Name: TB_Program_Counter - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
```

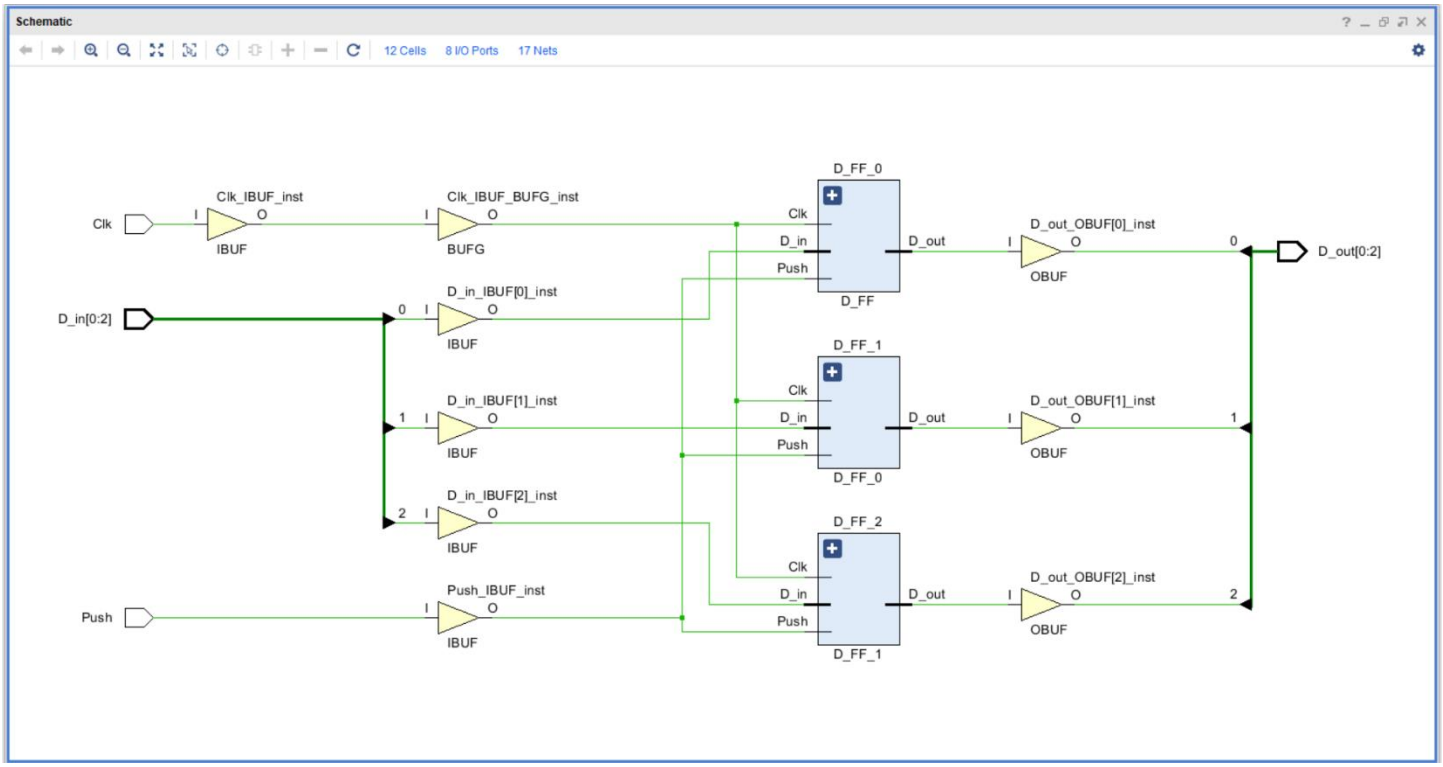
```
-- Additional Comments:
--
----------------------------------------------------------------------------------



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 TB_Program_Counter is
--  Port ( );
end TB_Program_Counter;

architecture Behavioral of TB_Program_Counter is
component Program_Counter
    Port ( D_in : in STD_LOGIC_VECTOR (0 to 2);
       Push : in STD_LOGIC; -- reset button
       Clk : in STD_LOGIC;
       D_out : out STD_LOGIC_VECTOR (0 to 2));
end component;
signal Clk:std_logic:='0';
signal D_in : STD_LOGIC_VECTOR (2 downto 0);
signal push: STD_LOGIC;
signal D_out : STD_LOGIC_VECTOR (2 downto 0);

begin
UUT:Program_Counter
 port map(
 D_in=>D_in,
 Push=>push,
 Clk=>Clk,
 D_out=>D_out);

 Process

 begin
  wait for 40ns;
```

```
  Clk<=not(Clk);
 end process;

 process
 begin
    push<='0';
    wait for 100ns;
    D_in<="110";
    wait for 100ns;
    D_in<="000";
    wait for 100ns;
    D_in<="111";
    wait for 100ns;
    push<='1';
    wait for 100ns;
    D_in<="101";
    wait for 100ns;
    D_in<="001";
    wait for 100ns;
    D_in<="110";
    wait for 100ns;
    D_in<="000";
    wait for 100ns;
    D_in<="011";
 wait;

end process;


end Behavioral;
```

3.7.4 Timing diagram



3.7.5 Implemented Design Schematic



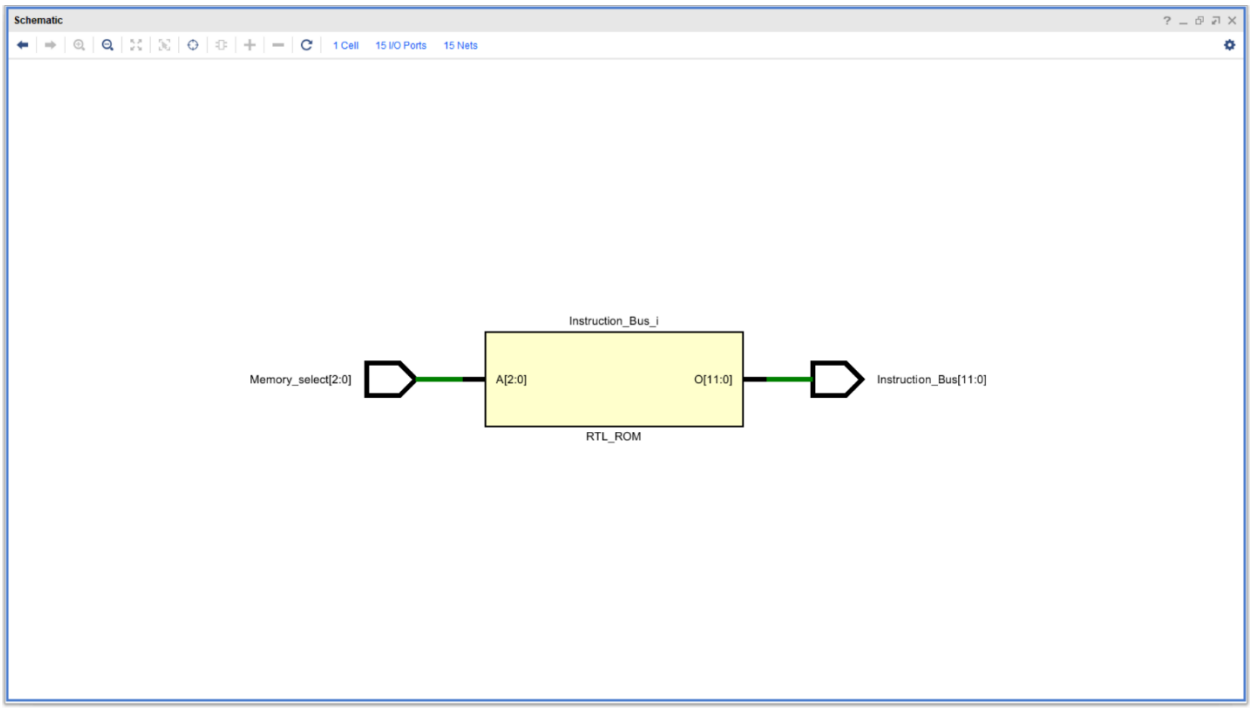
3.8 Program ROM

3.8.1 Design source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/07/2024 07:39:27 PM
-- Design Name:
-- Module Name: Program 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;
use ieee.numeric_std.all;

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

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

entity Program_ROM is
    Port ( Memory_select : in STD_LOGIC_VECTOR (2 downto 0);
           Instruction_Bus : out STD_LOGIC_VECTOR (11 downto 0));
end Program_ROM;

architecture Behavioral of Program_ROM is

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

```
    signal Program_ROM : rom_type := (

    -- operations to add 1 to 3
           "101110000001", -- 0 move value 1 to seventh register
           "100010000010", -- 1 move value 2 to 1 st register
           "100100000011", -- 2 move value 3 to 2 nd register
           "001110010000", -- 3 add reg 1 and reg 7 and store value to reg 7
           "001110100000", -- 4 add reg 2 and reg 7 and store value to reg 7
           "110000000000", -- 5 jump 0th line of the program rom if 0 th register value = 0000
           "000000000000", -- 6
           "000000000000"  -- 7

    -- operations to check NEG,JNZ,MOVI commands

--         "101110000010", -- 0  move value 2 to seventh register
--         "011110000000", -- 1  get the negative of the value in seventh register
--         "110000000100", -- 2  jump 4th line of the program rom if 0 th register value = 0000
--         "101110001111", -- 3  move value 15 to seventh register
--         "101110000011", -- 4  move value 3 to seventh register
--         "110000000000", -- 5  jump 0th line of the program rom if 0 th register value = 0000
--         "110000000000", -- 6  jump 0th line of the program rom if 0 th register value = 0000
--           "110000000000"  -- 7  jump 0th line of the program rom if 0 th register value = 0000

                            );
Begin

Instruction_Bus <= Program_ROM(to_integer(unsigned(Memory_select)));


end Behavioral;
```

3.8.2 Elaborated design schematic



3.9 Register bank

3.9.1 Design source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/10/2024 08:56:05 AM
-- Design Name:
-- Module Name: Register_bank - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------



library IEEE;
```

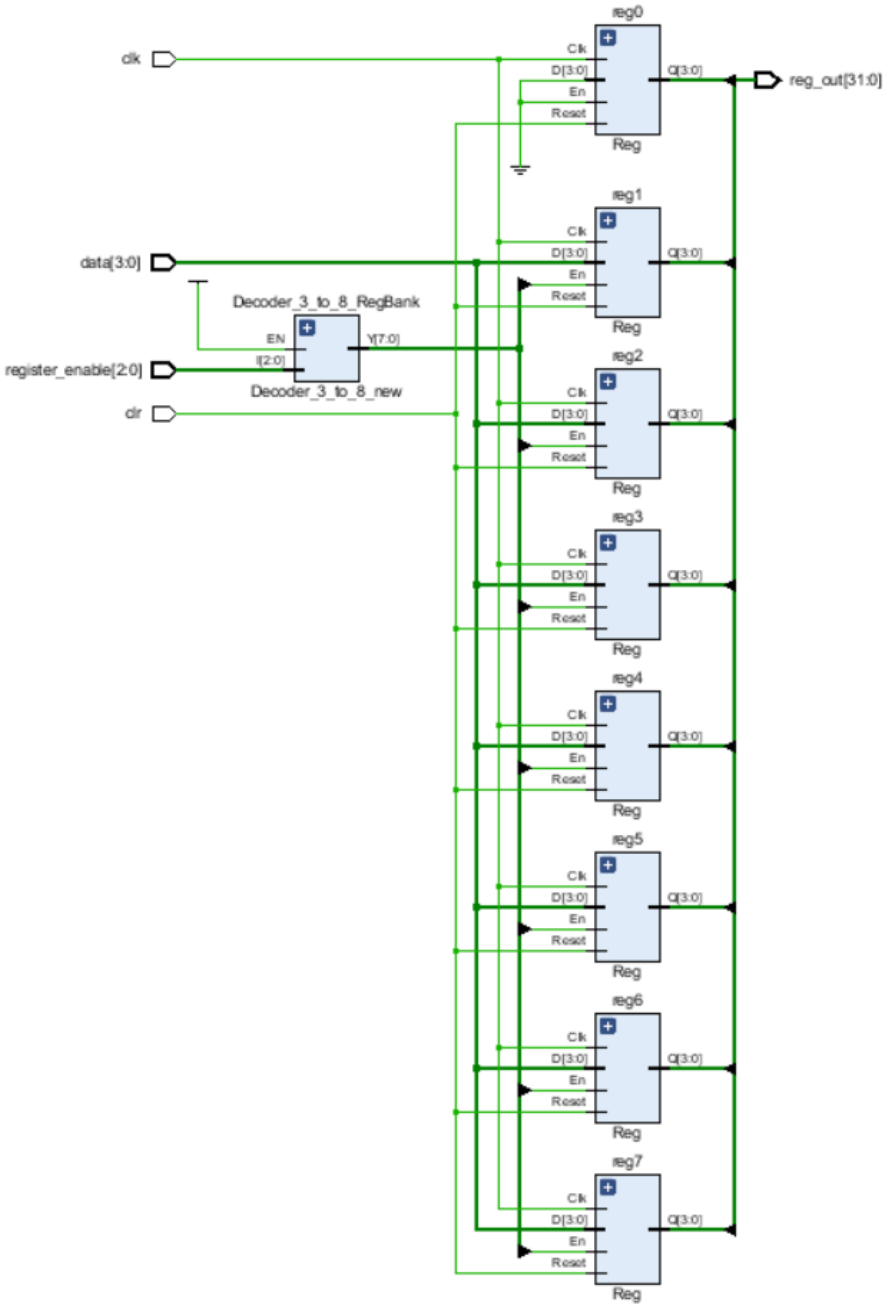
```
use IEEE.STD_LOGIC_1164.ALL;

-- 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 Register_bank is
    Port (
    register_enable:in STD_LOGIC_VECTOR (2 downto 0);--to decide which register going to be
enabled
    data:in STD_LOGIC_VECTOR (3 downto 0);
    clk:in STD_LOGIC; --to input slow clock signal
    clr:in STD_LOGIC; --related to reset button
    reg_out:out STD_LOGIC_VECTOR (31 downto 0)
    );
end Register_bank;

architecture Behavioral of Register_bank is

component Decoder_3_to_8_new is
 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);
           Q : out STD_LOGIC_VECTOR (3 downto 0);
           en : in STD_LOGIC;
           clk : in STD_LOGIC;
           reset : in STD_LOGIC
           );
end component;

signal en_r0,en_r1,en_r2,en_r3,en_r4,en_r5,en_r6,en_r7: STD_LOGIC; --to enable registers
separately

begin

Decoder_3_to_8_RegBank:Decoder_3_to_8_new
```

```
    port map( I=>register_enable,
              EN=>'1',
              Y(0)=>en_r0,
              Y(1)=>en_r1,
              Y(2)=>en_r2,
              Y(3)=>en_r3,
              Y(4)=>en_r4,
              Y(5)=>en_r5,
              Y(6)=>en_r6,
              Y(7)=>en_r7);
reg0:Reg
    port map(
    D=>"0000", --we need this value to implement NEG instruction
    Q=>reg_out(3 downto 0),
    en=>'0',
    clk=>clk,
    reset=>clr
    );
 reg1:Reg
        port map(
        D=>data,
        Q=>reg_out(7 downto 4),
        en=>en_r1,
        clk=>clk,
        reset=>clr
        );



reg2:Reg
        port map(
        D=>data,
        Q=>reg_out(11 downto 8),
        en=>en_r2,
        clk=>clk,
        reset=>clr
        );

reg3:Reg
        port map(
        D=>data,
        Q=>reg_out(15 downto 12),
        en=>en_r3,
        clk=>clk,
        reset=>clr
        );
```

```
reg4:Reg
        port map(
        D=>data,
        Q=>reg_out(19 downto 16),
        en=>en_r4,
        clk=>clk,
        reset=>clr
        );
reg5:Reg
        port map(
        D=>data,
        Q=>reg_out(23 downto 20),
        en=>en_r5,
        clk=>clk,
        reset=>clr
        );
reg6:Reg
        port map(
        D=>data,
        Q=>reg_out(27 downto 24),
        en=>en_r6,
        clk=>clk,
        reset=>clr
        );
reg7:Reg
        port map(
        D=>data,
        Q=>reg_out(31 downto 28),
        en=>en_r7,
        clk=>clk,
        reset=>clr
        );
end Behavioral;
```

3.9.2 Elaborated design schematic

9 Cells 41 I/O Ports 50 Nets



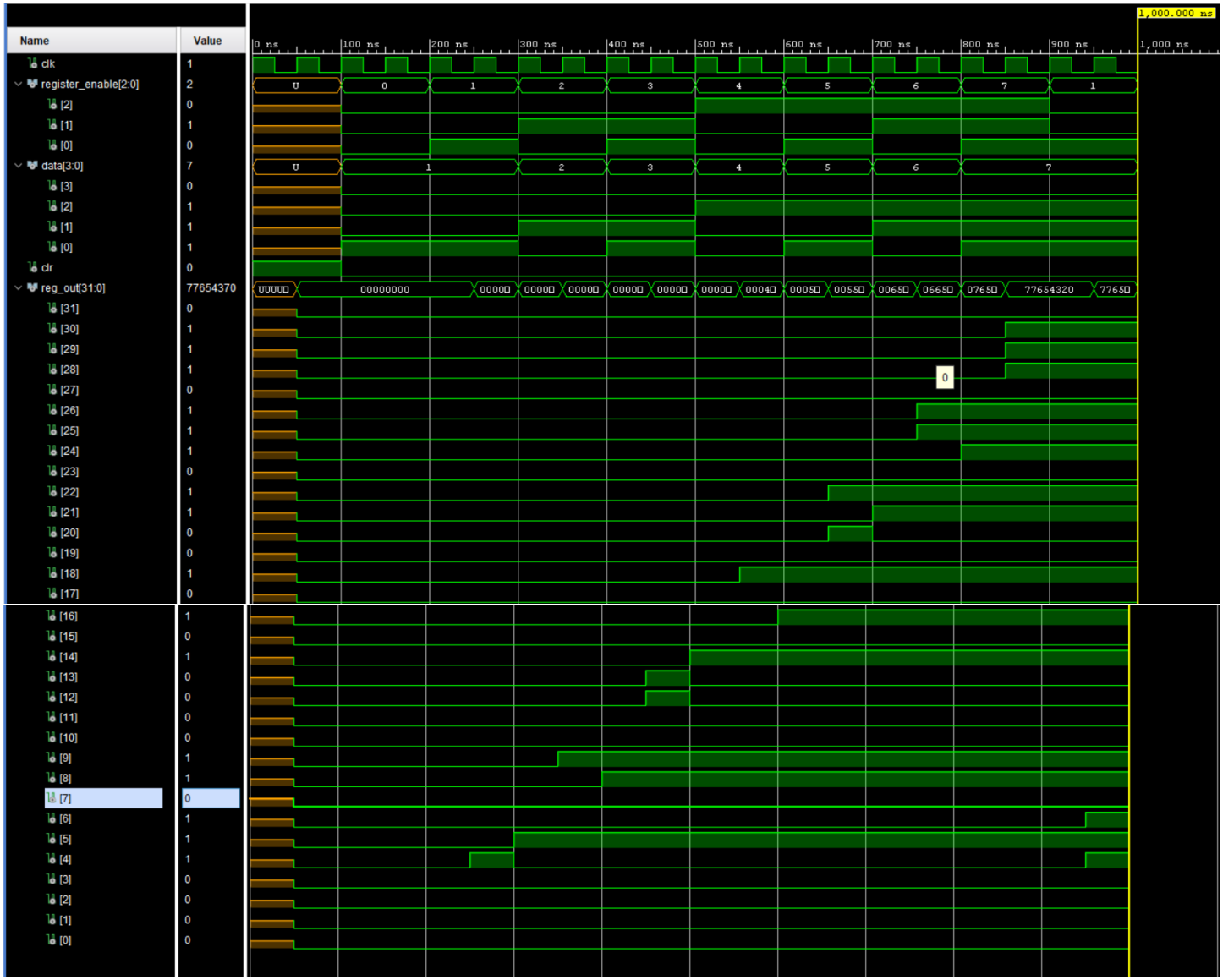
3.9.3 Simulation source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/11/2024 07:15:13 PM
-- Design Name:
-- Module Name: TB_Register_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 TB_Register_Bank is
--  Port ( );
end TB_Register_Bank;
architecture Behavioral of TB_Register_Bank is
component Register_bank is
    Port (
    register_enable:in STD_LOGIC_VECTOR (2 downto 0);--to decide which register going to be
enabled
    data:inout STD_LOGIC_VECTOR (3 downto 0);
    clk:in STD_LOGIC; --to input slow clock signal
    clr:in STD_LOGIC; --related to reset button
    reg_out:inout STD_LOGIC_VECTOR (31 downto 0)
    );
end component;
signal register_enable: STD_LOGIC_VECTOR (2 downto 0);
```

```
signal data:STD_LOGIC_VECTOR (3 downto 0);
signal clk: STD_LOGIC;
signal clr: STD_LOGIC;
signal reg_out: STD_LOGIC_VECTOR (31 downto 0);
begin
UUT : Register_Bank
    Port map (
    register_enable => register_enable,
    data => data,
    clk => clk,
    clr => clr,
    reg_out => reg_out
    );
process begin
    clk <= '1';
    wait for 25 ns;
    clk <= '0';
    wait for 25 ns;
end process;
process begin
    clr <= '1';
    wait for 100 ns;
    clr <= '0';
    register_enable <= "000";
    data <= "0001";
    wait for 100 ns;
    register_enable <= "001";
    data <= "0001";
    wait for 100 ns;
    register_enable <= "010";
    data <= "0010";
    wait for 100 ns;
    register_enable <= "011";
    data <= "0011";
    wait for 100 ns;
    register_enable <= "100";
    data <= "0100";
    wait for 100 ns;
    register_enable <= "101";
    data <= "0101";
    wait for 100 ns;
    register_enable <= "110";
    data <= "0110";
    wait for 100 ns;
    register_enable <= "111";
    data <= "0111";
```

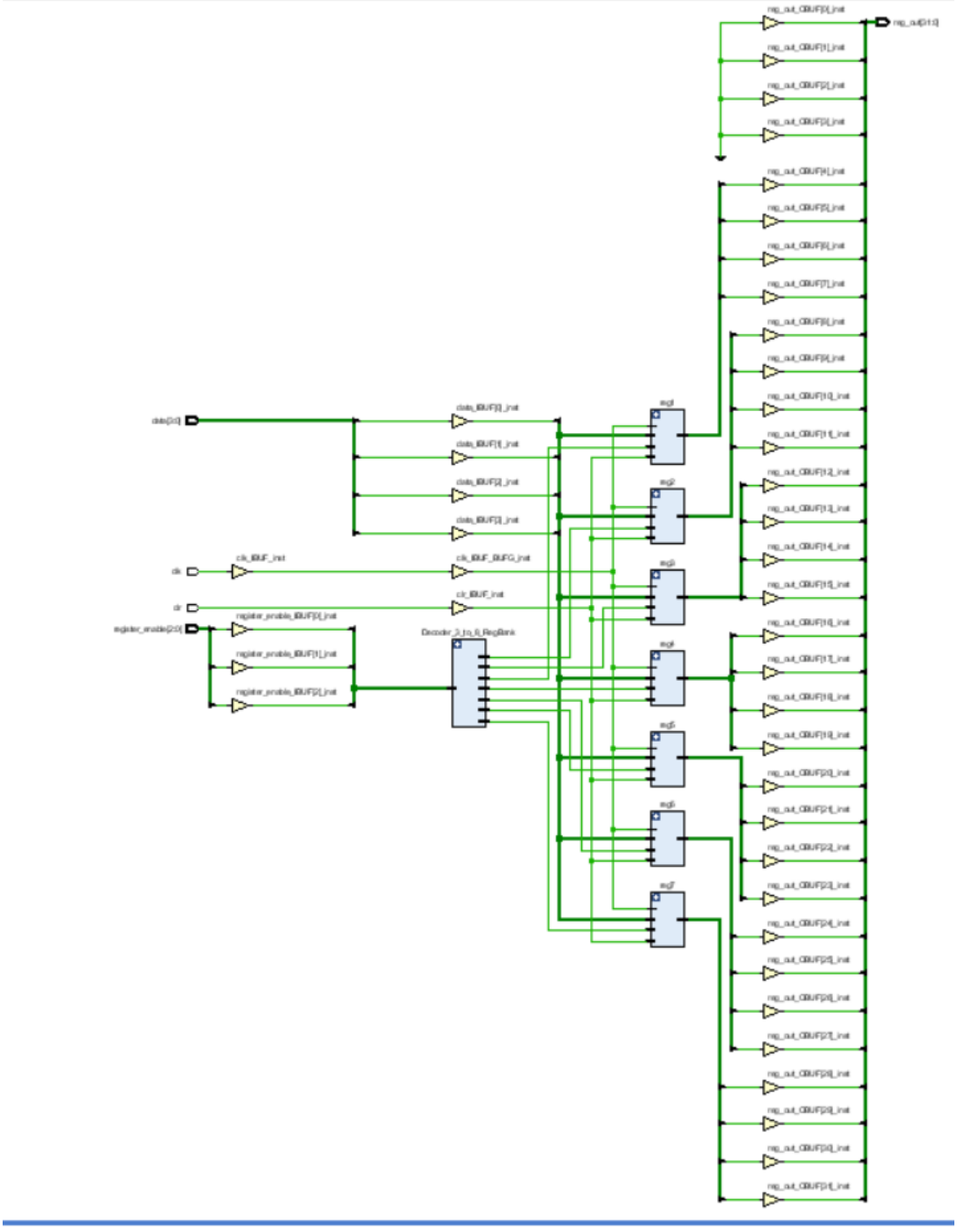
```
    wait for 100 ns;
    register_enable <= "001";
    wait for 100 ns;
    register_enable <= "010";
    wait;

end process;
end Behavioral;
```

3.9.4 Timing diagram



3.9.5 Implemented Design Schematic

50 Cells 41 I/O Ports 87 Nets



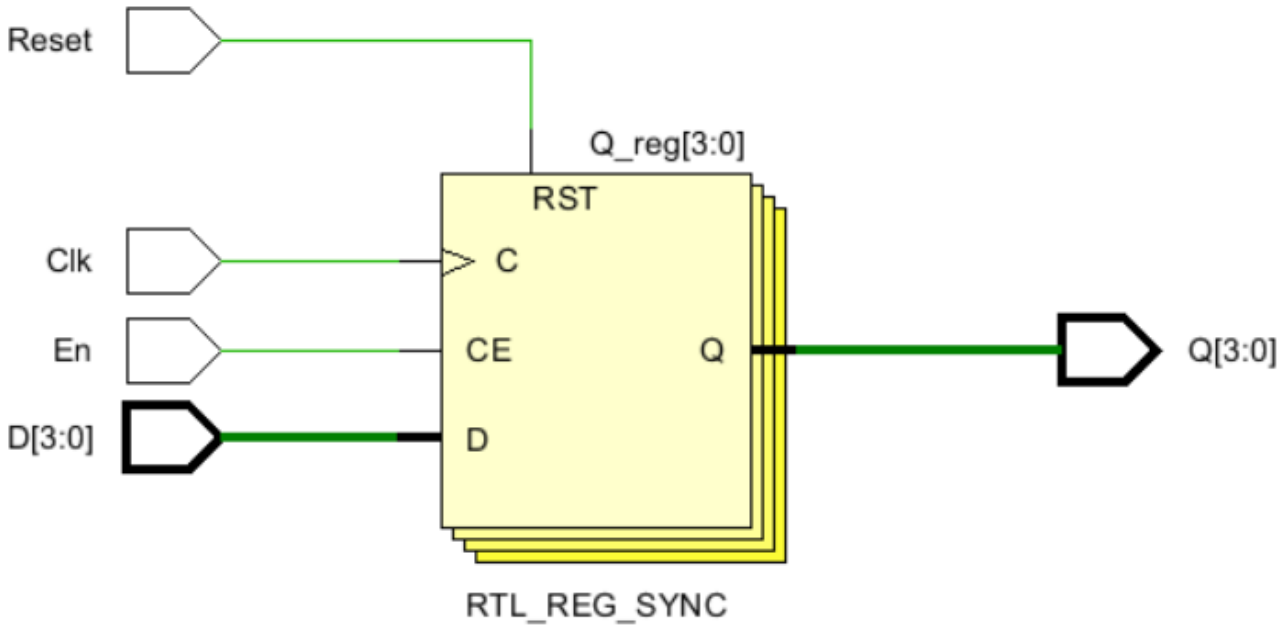
3.10 Register

3.10.1 Design source file

```
-- Company:
-- Engineer:
--
-- Create Date: 03/14/2024 02:38:30 PM
-- 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;
--…


entity Reg is
    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 Reg;

architecture Behavioral of Reg is
```

```
component D_F_F
Port ( D : in STD_LOGIC;
       Res : in STD_LOGIC;
       Clk : in STD_LOGIC;
       En : in STD_LOGIC;
       Q : out STD_LOGIC;
       Qbar : out STD_LOGIC);
end component;


begin

    process (Clk, Reset)
    begin

        if (rising_edge(Clk)) then
             if (Reset = '1') then
                   Q <= "0000";  -- Reset all bits to 0
             elsif En = '1' then
                Q <= D;
            end if;
        end if;
    end process;


end Behavioral;
```

3.10.2 Elaborated design schematic



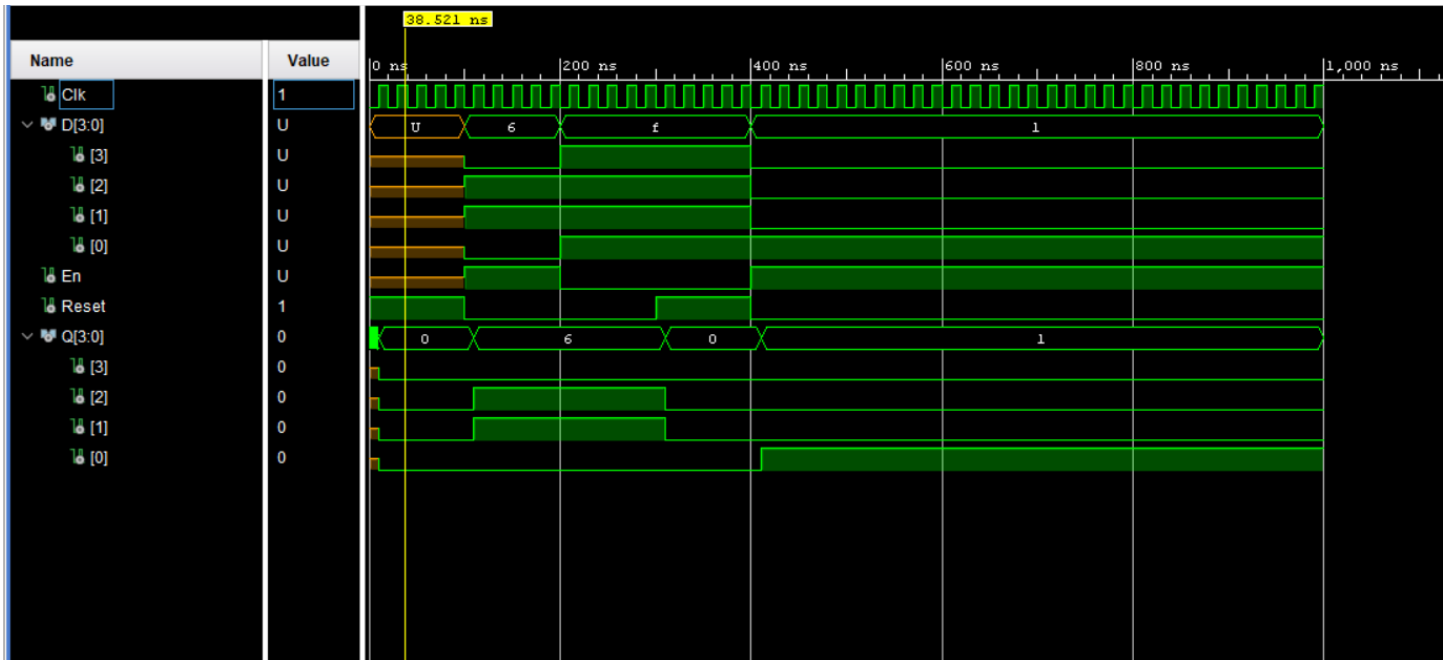
3.10.3 Simulation source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/11/2024 06:21:29 PM
-- Design Name:
-- Module Name: TB_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 TB_Reg is
--  Port ( );
end TB_Reg;

architecture Behavioral of TB_Reg is

component Reg is
    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 D : STD_LOGIC_VECTOR (3 downto 0);
signal En : STD_LOGIC;
signal Clk : STD_LOGIC;
signal Reset: STD_LOGIC;
signal Q : STD_LOGIC_VECTOR (3 downto 0);

begin

UUT: Reg
    Port map( D => D,
           En => En,
           Clk => Clk,
           Reset => Reset,
           Q => Q);

process begin
    clk <= '0';
    wait for 10 ns;
```

```
    clk <= '1';
    wait for 10 ns;
end process;

process begin
    Reset <= '1';
    wait for 100 ns;
    Reset <= '0';
    En <= '1';
    D <= "0110";
    wait for 100 ns;
    En <= '0';
    D <= "1111";
    wait for 100 ns;
    Reset <= '1';
    wait for 100 ns;
    Reset <= '0';
    En <= '1';
    D <= "0001";
    wait;

end process;




end Behavioral;
```

3.10.4 Timing diagram



3.11 Three to eight Decoder

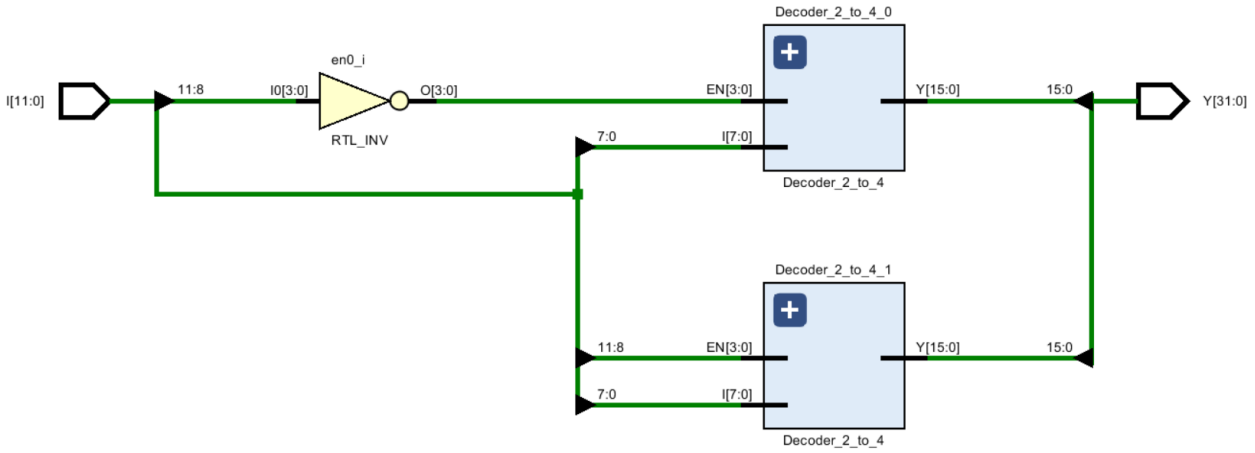
3.11.1 Design source file

```
4   ----------------------------------------------------------------------------------
5   -- Company:
6   -- Engineer:
7   --
8   -- Create Date: 02/20/2024 03:34:00 PM
9   -- Design Name:
10  -- Module Name: Decoder_3_to_8 - Behavioral
11  -- Project Name:
12  -- Target Devices:
13  -- Tool Versions:
14  -- Description:
15  --
16  -- Dependencies:
17  --
18  -- Revision:
19  -- Revision 0.01 - File Created
20  -- Additional Comments:
21  --
22  ----------------------------------------------------------------------------------
23
24  library IEEE;
25  use IEEE.STD_LOGIC_1164.ALL;
26
```

```
-- 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 Decoder_3_to_8 is
    Port ( I : in STD_LOGIC_VECTOR (11 downto 0);--input for the decoder
           Y : out STD_LOGIC_VECTOR (31 downto 0));--output
end Decoder_3_to_8;

architecture Behavioral of Decoder_3_to_8 is

--use 2 to 4 decoders to build a 3 to 8 decoder
component Decoder_2_to_4
 port(
 I: in STD_LOGIC_VECTOR;--input
 EN: in STD_LOGIC_VECTOR;--enable
 Y: out STD_LOGIC_VECTOR--output
 );
 end component;

 -- signals to make decoder
 signal I0,I1 : STD_LOGIC_VECTOR (7 downto 0);
 signal Y0,Y1 : STD_LOGIC_VECTOR (15 downto 0);
 signal en0,en1 :  STD_LOGIC_VECTOR (3 downto 0);
 signal  I2 :  STD_LOGIC_VECTOR (3 downto 0);

begin

 Decoder_2_to_4_0 : Decoder_2_to_4
 port map(
 I => I0,
 EN => en0,
 Y => Y0 );

 Decoder_2_to_4_1 : Decoder_2_to_4
 port map(
 I => I1,
 EN => en1,
 Y => Y1 );

-- logic for the decoder
 en0 <= NOT(I(11 downto 8));
```

```
74   en1 <= I(11 downto 8);
75   I0 <= I(7 downto 0);
76   I1 <= I(7 downto 0);
77   I2 <= I(11 downto 8);
78   Y(15 downto 0) <= Y0;
79   Y(31 downto 16) <= Y1;
80  end Behavioral;
81
```

81.1.1 Elaborated design schematic



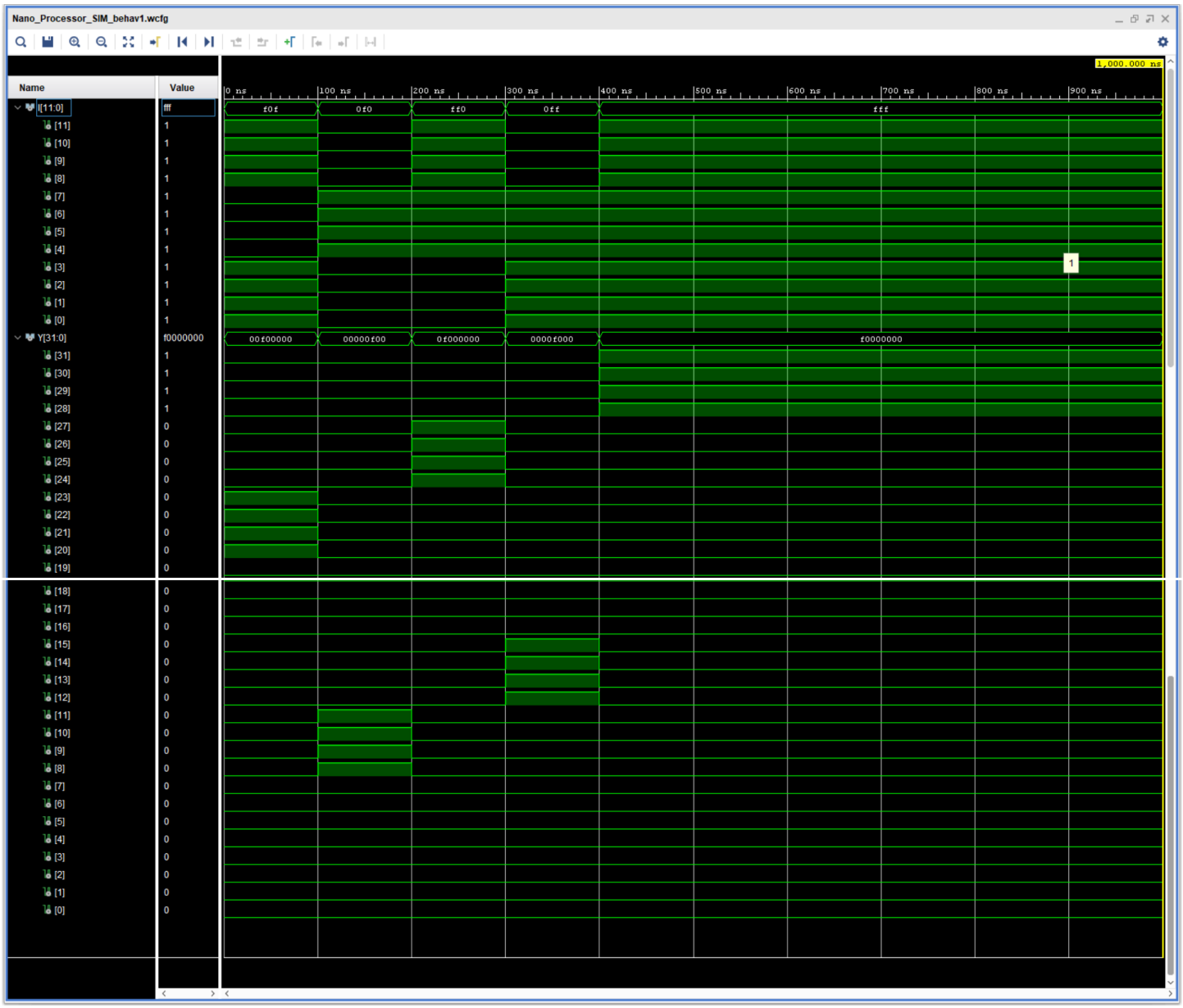
81.1.2 Simulation source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/23/2024 08:46:46 PM
-- Design Name:
-- Module Name: Decoder_3_to_8_sim - 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 Decoder_3_to_8_sim is
--  Port ( );
end Decoder_3_to_8_sim;

architecture Behavioral of Decoder_3_to_8_sim is

component Decoder_3_to_8 is
    Port ( I : in STD_LOGIC_VECTOR (11 downto 0);--input for the decoder
           Y : out STD_LOGIC_VECTOR (31 downto 0));--output
end component;

signal I : STD_LOGIC_VECTOR (11 downto 0);
signal Y : STD_LOGIC_VECTOR (31 downto 0);

begin

UUT :Decoder_3_to_8 port map
    ( I => I,
      Y => Y);

process begin

    -- input
    I <= "111100001111";
    wait for 100 ns;
```

```
    --input
    I <= "000011110000";

    wait for 100 ns;
    --input
    I <= "111111110000";

    wait for 100 ns;
    --input
    I <= "000011111111";

    wait for 100 ns;
    --input
    I <= "111111111111";

    wait;

end process;
end Behavioral;
```

81.1.3 Timing diagram



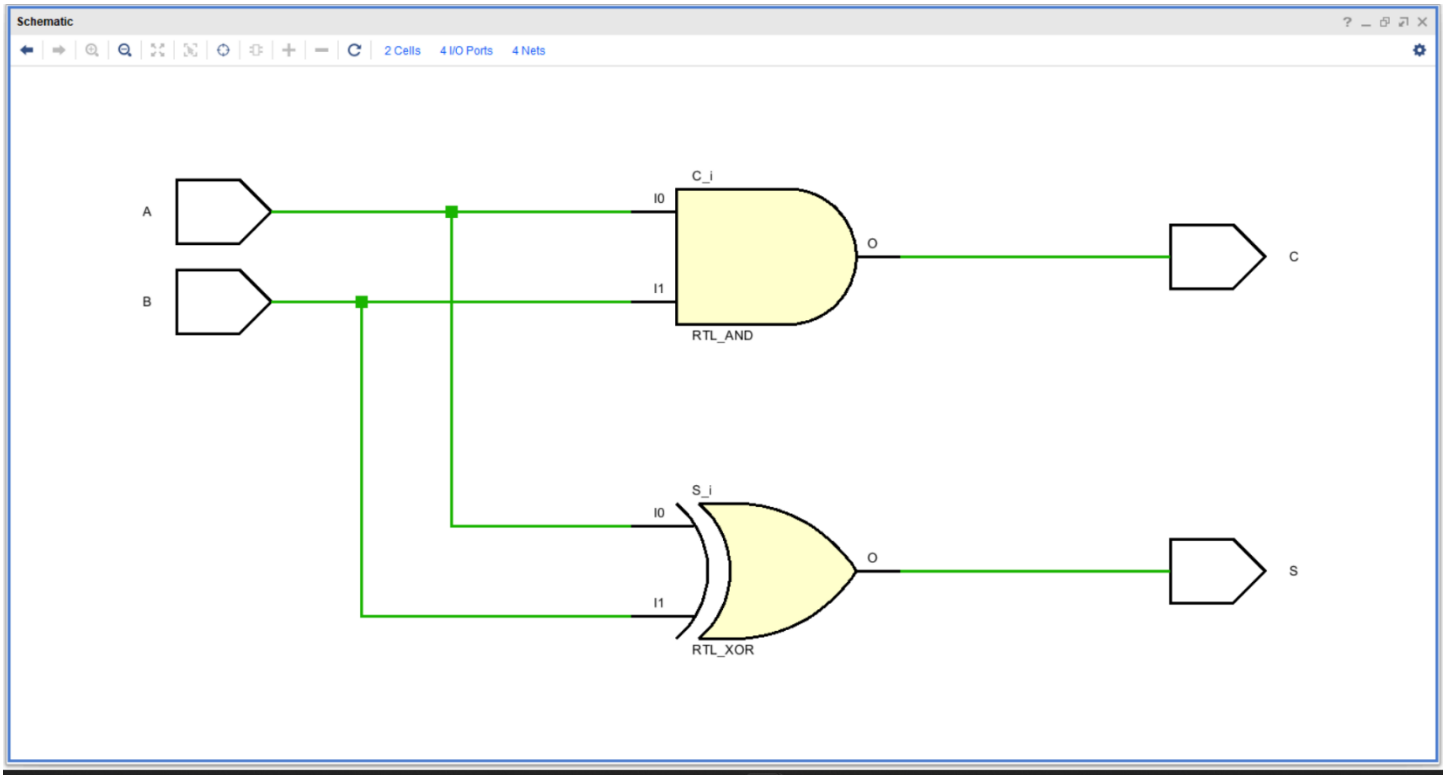
81.2 Half adder

81.2.1 Design source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 02/14/2024 12:36:10 AM
-- Design Name:
-- Module Name: HA - 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 HA is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           S : out STD_LOGIC;
           C : out STD_LOGIC);
end HA;

architecture Behavioral of HA is

begin
    S <= A XOR B;
    C <= A AND B;
end Behavioral;
```

81.2.2 Elaborated design schematic



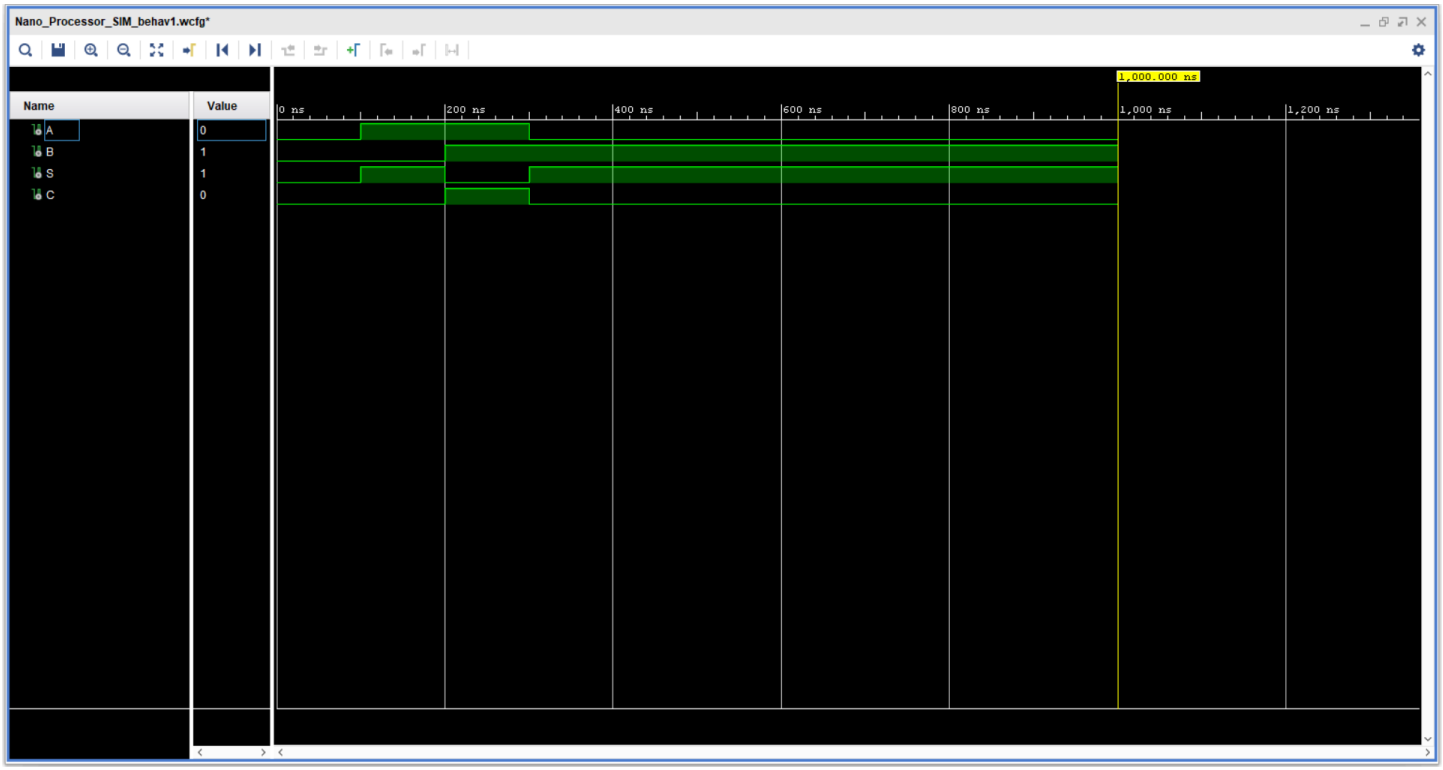
81.2.3 Simulation source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 02/14/2024 12:48:03 AM
-- Design Name:
-- Module Name: TB_HA - 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 TB_HA is
--  Port ( );
end TB_HA;

architecture Behavioral of TB_HA is
     COMPONENT HA
        PORT( A,B : IN STD_LOGIC;
              S,C : OUT STD_LOGIC);
     END COMPONENT;
     SIGNAL A,B : std_logic;
     SIGNAL S,C: std_logic;

begin
     UUT: HA PORT MAP(
            A => A,
            B => B,
            S => S,
            C => C
);
    process
 begin
    A <= '0';
    B <= '0';
    WAIT FOR 100 ns;
    A <= '1';
    WAIT FOR 100 ns;
    B <= '1';
    WAIT FOR 100 ns;
    A <= '0';
    WAIT;
 end process;
```

```
end Behavioral;
```

81.2.4 Timing diagram



81.3 Full adder

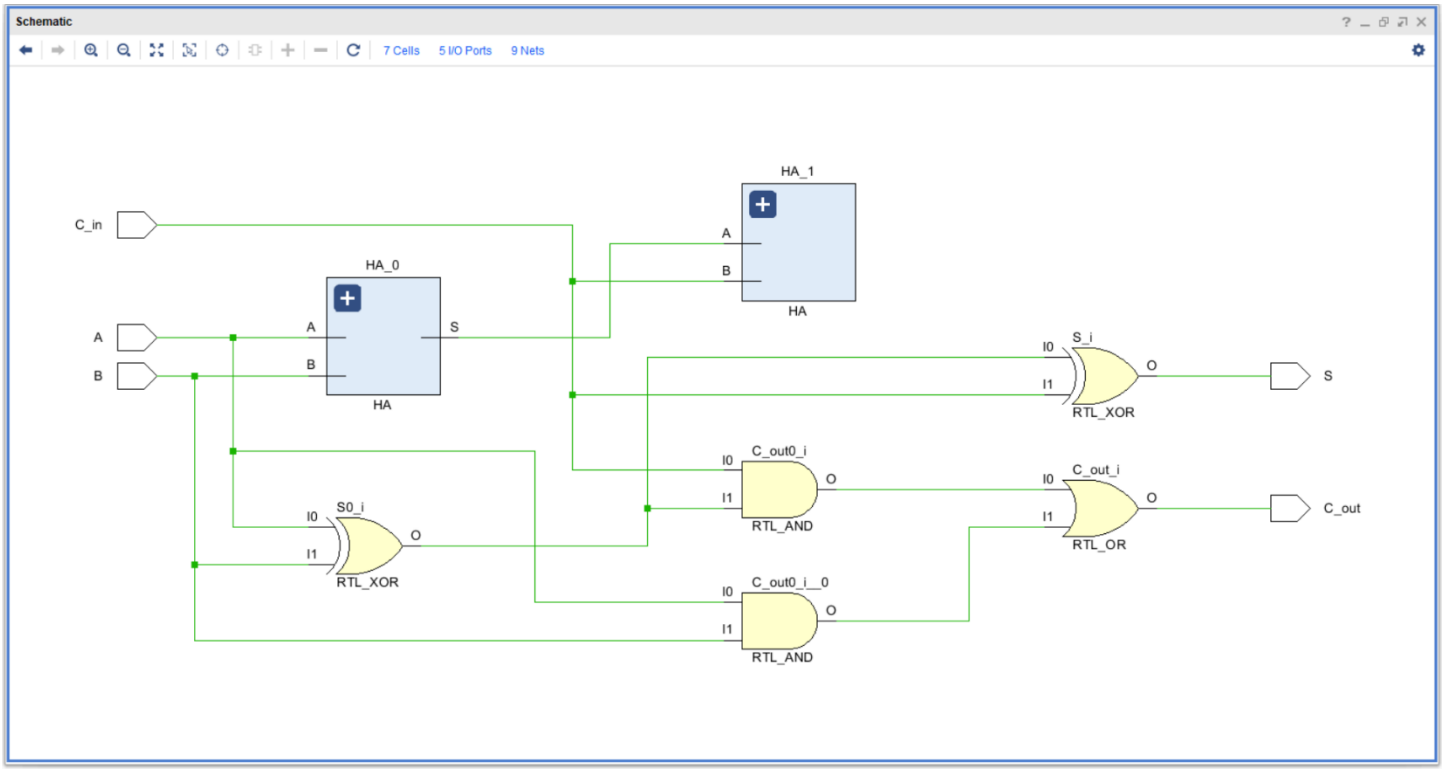
81.3.1 Design source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 02/14/2024 01:07:45 AM
-- Design Name:
-- Module Name: FA - 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 FA is
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C_in : in STD_LOGIC;
           S : out STD_LOGIC;
           C_out : out STD_LOGIC);
end FA;

architecture Behavioral of FA is
component HA
     port (
     A: in std_logic;
     B: in std_logic;
     S: out std_logic;
     C: out std_logic);
     end component;
     SIGNAL HA0_S, HA0_C, HA1_S, HA1_C : std_logic;
begin
HA_0 : HA
 port map (
     A => A,
     B => B,
     S => HA0_S,
     C => HA0_C
     );
 HA_1 : HA
 port map (
     A => HA0_S,
     B => C_in,
     S => HA1_S,
     C => HA1_C
```

```
    );
 C_out <= (C_in AND ( A XOR B))OR(A AND B);
 S<= (A XOR B) XOR C_in;

end Behavioral;
```

81.3.2 Elaborated design schematic



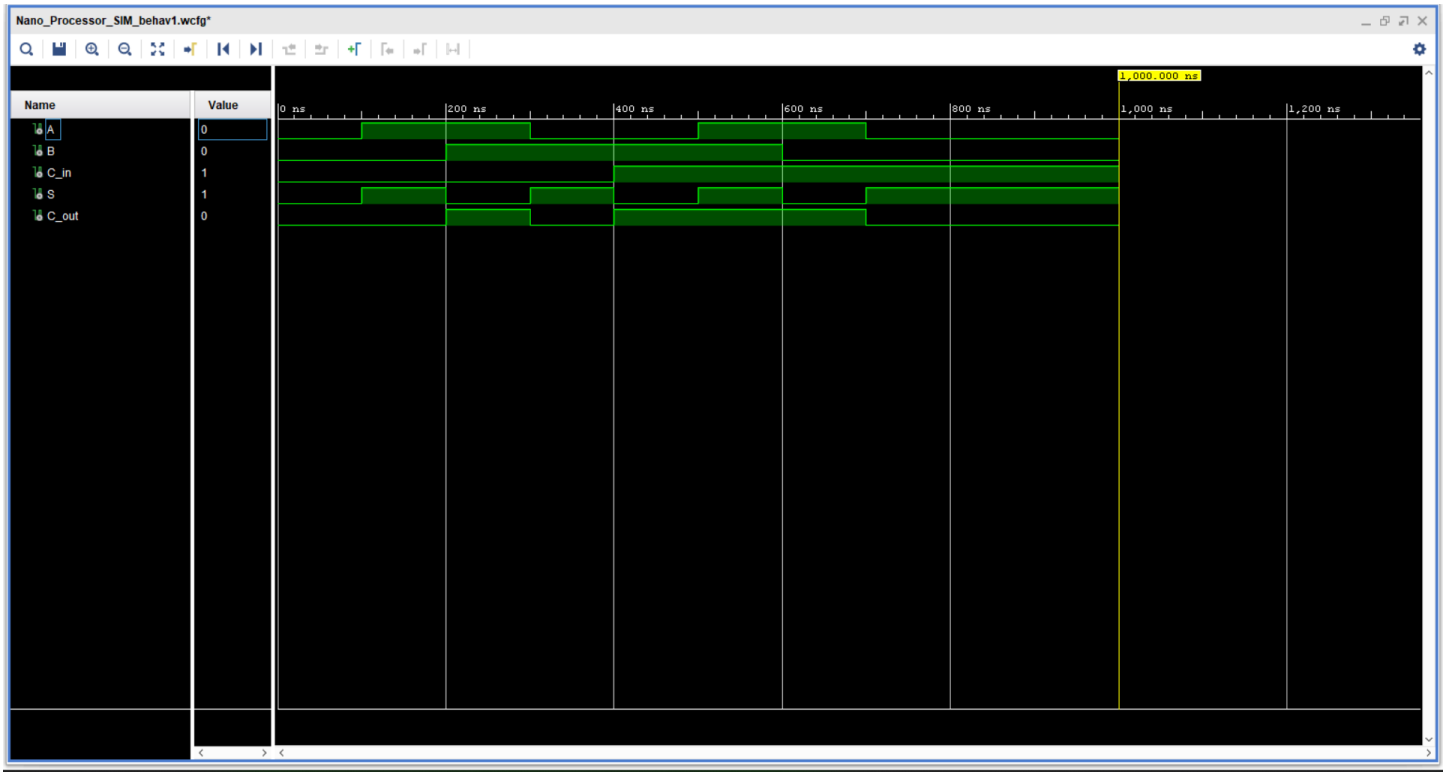
81.3.3 Simulation source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 02/14/2024 01:20:48 AM
-- Design Name:
-- Module Name: TB_FA - 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 TB_FA is
--  Port ( );
end TB_FA;

architecture Behavioral of TB_FA is
        COMPONENT FA
        PORT( A,B,C_in : IN STD_LOGIC;
              S,C_out : OUT STD_LOGIC);
        END COMPONENT;
        SIGNAL  A,B,C_in: std_logic;
        SIGNAL S,C_out : std_logic;
begin
    UUT: FA PORT MAP(
            A=> A,
            B=> B,
            C_in => C_in,
            S => S,
            C_out =>C_out
            );
     process
      begin
      A <= '0';
      B <= '0';
      C_in <= '0';
      WAIT FOR 100 ns;
      A <= '1';
      WAIT FOR 100 ns;
      B <= '1';
```

```
      WAIT FOR 100 ns;
      A <= '0';
      WAIT FOR 100 ns;
      C_in <= '1';
      WAIT FOR 100 ns;
      A <= '1';
      WAIT FOR 100 ns;
      B <= '0';
      WAIT FOR 100 ns;
      A <= '0';
      WAIT;
      end process;
end Behavioral;
```

81.3.4 Timing diagram



81.4 Seven segment ROM

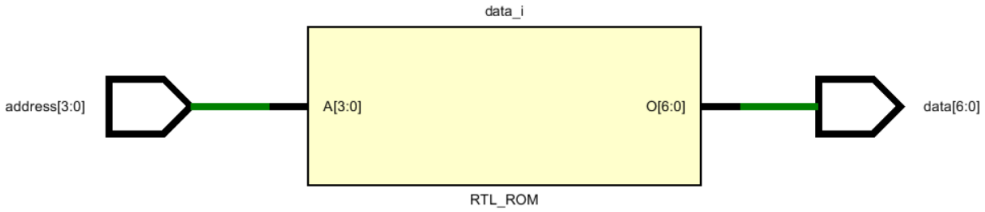
81.4.1 Design source file

```
----------------------------------------------------------------------------------
-- Company:
```

```
-- Engineer:
--
-- Create Date: 05/02/2023 03:15:50 PM
-- Design Name:
-- Module Name: LUT_16_7 - 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;
use ieee.numeric_std.all;

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

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

entity LUT_16_7 is
    Port ( address : in STD_LOGIC_VECTOR (3 downto 0);
           data : out STD_LOGIC_VECTOR (6 downto 0));
end LUT_16_7;

architecture Behavioral of LUT_16_7 is

type rom_type is array (0 to 15) of std_logic_vector(6 downto 0);

 signal sevenSegment_ROM : rom_type := (
 "1000000", -- 0
 "1111001", -- 1
 "0100100", -- 2
```

```
 "0110000", -- 3
 "0011001", -- 4
 "0010010", -- 5
 "0000010", -- 6
 "1111000", -- 7
 "0000000", -- 8
 "0010000", -- 9
 "0001000", -- a
 "0000011", -- b
 "1000110", -- c
 "0100001", -- d
 "0000110", -- e
 "0001110" -- f
);

begin

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

end Behavioral;
```

81.4.2 Elaborated design schematic



81.5 Instruction decoder

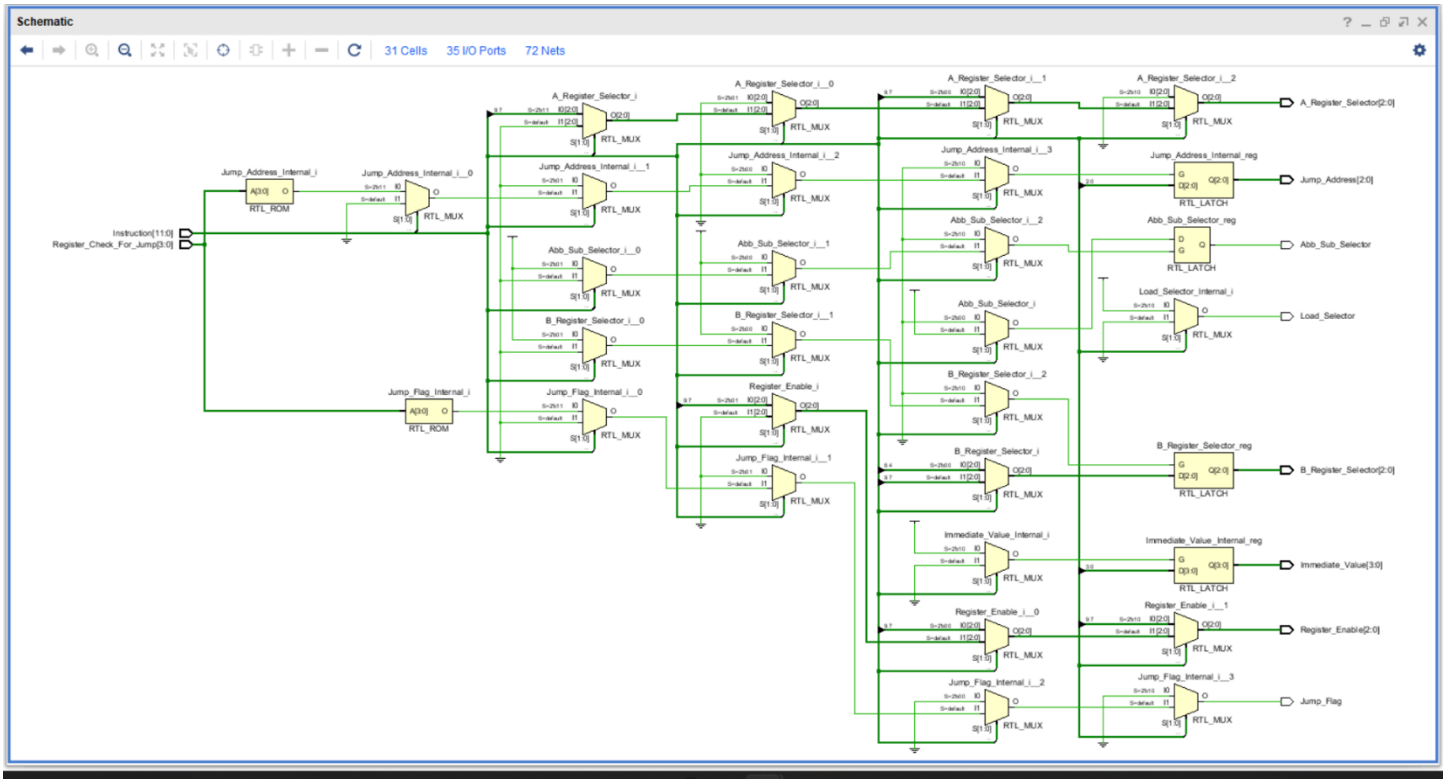
81.5.1 Design source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/08/2024 07:41:28 PM
-- Design Name:
-- Module Name: Instruction_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 Instruction_Decoder is
    Port ( Instruction : in STD_LOGIC_VECTOR (11 downto 0);
           Register_Check_For_Jump : in STD_LOGIC_VECTOR (3 downto 0);
           Register_Enable : out STD_LOGIC_VECTOR (2 downto 0);
           Immediate_Value : out STD_LOGIC_VECTOR (3 downto 0);
           Load_Selector : out STD_LOGIC;
           A_Register_Selector : out STD_LOGIC_VECTOR (2 downto 0);
           B_Register_Selector : out STD_LOGIC_VECTOR (2 downto 0);
           Abb_Sub_Selector : out STD_LOGIC;
           Jump_Flag : out STD_LOGIC;
           Jump_Address : out STD_LOGIC_VECTOR (2 downto 0)
           );

end Instruction_Decoder;

architecture Behavioral of Instruction_Decoder is
    signal Load_Selector_Internal : STD_LOGIC;
    signal Immediate_Value_Internal : STD_LOGIC_VECTOR (3 downto 0);
    signal Jump_Flag_Internal : STD_LOGIC;
    signal Jump_Address_Internal : STD_LOGIC_VECTOR (2 downto 0);
begin
    process (Instruction, Register_Check_For_Jump) is
    begin

    -- inicialize values of the outputs
      Jump_Flag_Internal <= '0';-- Initialize Jump_Flag_Internal to 0
      Register_Enable <= "000"; -- Initialize Register_Enable to 000
      A_Register_Selector <= "000";-- Initialize A_Register_Selector to 000
      Load_Selector_Internal <= '0';-- Initialize Load_Selector_Internal to 0

            -- Move Instruction
            if (Instruction(11 downto 10) ="10") then
                Immediate_Value_Internal <= Instruction(3 downto 0);
```

```
            Load_Selector_Internal <= '1';
            Register_Enable <= Instruction(9 downto 7);
            Jump_Flag_Internal <= '0';


        -- Add values Instruction
        elsif (Instruction(11 downto 10)="00") then
            A_Register_Selector <= Instruction(9 downto 7);
            B_Register_Selector <= Instruction(6 downto 4);
            Abb_Sub_Selector <= '0';
            Load_Selector_Internal <= '0';
            Jump_Flag_Internal <= '0';
            Register_Enable <= Instruction(9 downto 7);



        -- 2's complement Instruction
        elsif (Instruction(11 downto 10) ="01") then
            B_Register_Selector <= Instruction(9 downto 7);
            A_Register_Selector <= "000";
            Abb_Sub_Selector <= '1';
            Load_Selector_Internal <= '0';
            Jump_Flag_Internal <= '0';
            Register_Enable <= Instruction(9 downto 7);


        -- Jump Instruction
        elsif (Instruction(11 downto 10)="11") then
            A_Register_Selector <= Instruction(9 downto 7);
            if(Register_Check_For_Jump = "0000") then
                Jump_Flag_Internal <= '1';
                Jump_Address_Internal <= Instruction(2 downto 0);
            else
                Jump_Flag_Internal <= '0';
            end if;

        end if;
    end process;

    -- Assign internal signals to output signals
    Immediate_Value <= Immediate_Value_Internal;
    Load_Selector <= Load_Selector_Internal;
    Jump_Flag <= Jump_Flag_Internal;
    Jump_Address <= Jump_Address_Internal;
end Behavioral;
```

81.5.2 Elaborated design schematic



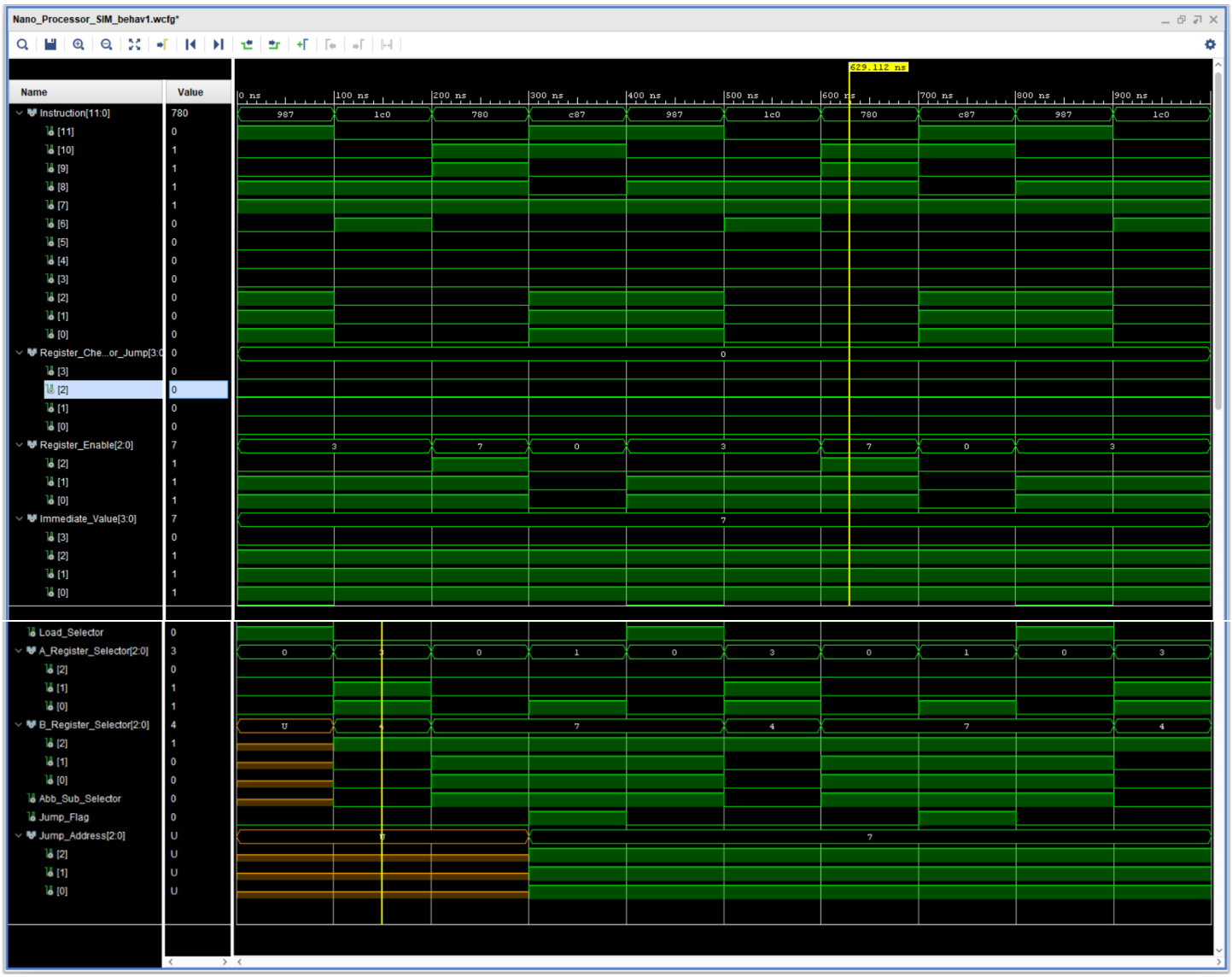
81.5.3 Simulation source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/10/2024 10:08:11 PM
-- Design Name:
-- Module Name: TB_Instruction_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 TB_Instruction_Decoder is
--  Port ( );
end TB_Instruction_Decoder;

architecture Behavioral of TB_Instruction_Decoder is
component Instruction_Decoder is
    Port ( Instruction : in STD_LOGIC_VECTOR (11 downto 0);
           Register_Check_For_Jump : in STD_LOGIC_VECTOR (3 downto 0);
           Register_Enable : out STD_LOGIC_VECTOR (2 downto 0);
           Immediate_Value : out STD_LOGIC_VECTOR (3 downto 0);
           Load_Selector : out STD_LOGIC;
           A_Register_Selector : out STD_LOGIC_VECTOR (2 downto 0);
           B_Register_Selector : out STD_LOGIC_VECTOR (2 downto 0);
           Abb_Sub_Selector : out STD_LOGIC;
           Jump_Flag : out STD_LOGIC;
           Jump_Address : out STD_LOGIC_VECTOR (2 downto 0));
end component;

signal Instruction :  STD_LOGIC_VECTOR (11 downto 0);
signal Register_Check_For_Jump : STD_LOGIC_VECTOR (3 downto 0);
signal Register_Enable : STD_LOGIC_VECTOR (2 downto 0);
signal Immediate_Value :  STD_LOGIC_VECTOR (3 downto 0);
signal Load_Selector :  STD_LOGIC;
signal A_Register_Selector :  STD_LOGIC_VECTOR (2 downto 0);
signal B_Register_Selector :  STD_LOGIC_VECTOR (2 downto 0);
signal Abb_Sub_Selector :  STD_LOGIC;
signal Jump_Flag :  STD_LOGIC;
signal Jump_Address :  STD_LOGIC_VECTOR (2 downto 0);

begin
UUT : Instruction_Decoder
```

```
port map(
        Instruction =>Instruction,
        Register_Check_For_Jump =>Register_Check_For_Jump ,
        Register_Enable=>Register_Enable ,
        Immediate_Value=>Immediate_Value ,
        Load_Selector => Load_Selector,
        A_Register_Selector=>A_Register_Selector ,
        B_Register_Selector=>B_Register_Selector ,
        Abb_Sub_Selector=>Abb_Sub_Selector ,
        Jump_Flag=>Jump_Flag ,
        Jump_Address=> Jump_Address);
process begin
Instruction <= "100110000111" ;--move value 7 to 3 rd register
 Register_Check_For_Jump <="0000";
wait for 100ns;
Instruction <= "000111000000" ;--add value in reg 3 and 4
 Register_Check_For_Jump <="0000";
wait for 100ns;
Instruction <= "011110000000" ;--two's complement of register  7
 Register_Check_For_Jump <="0000";
wait for 100ns;
Instruction <= "110010000111" ;--jump to 7 th instruction if reg 1 is 0000
 Register_Check_For_Jump <="0000";
wait for 100ns;


end process;


end Behavioral;
```

81.5.4 Timing diagram



81.6 D flipflop

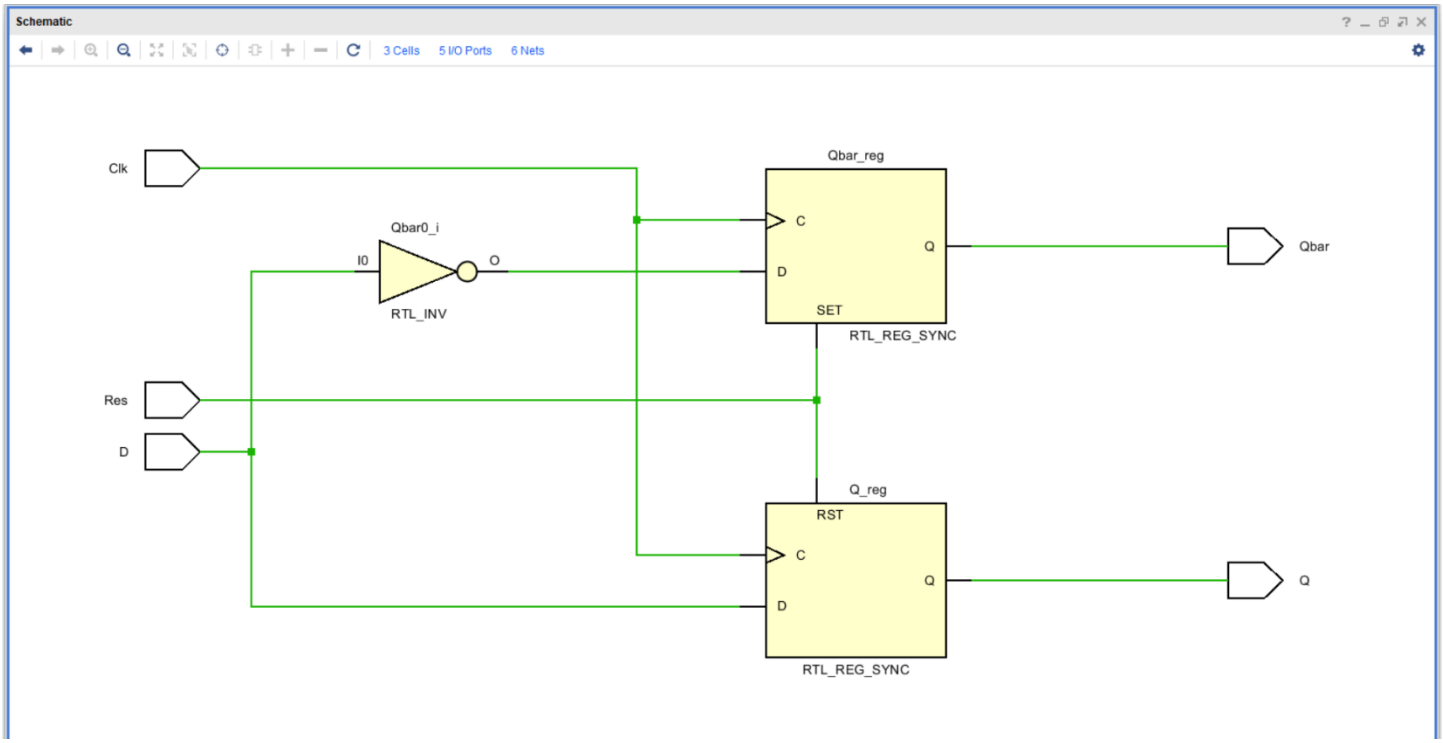
81.6.1 Design source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 03/05/2024 02:30:31 PM
-- Design Name:
-- Module Name: D_FF - 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 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 D_FF;

architecture Behavioral of D_FF is

begin
    process (Clk) begin

        if (rising_edge(Clk)) then
            if Res = '1' then
                Q <= '0';
                Qbar <= '1';
            else
                Q <= D;
                Qbar <= not D;
            end if;
        end if;
    end process;
```

```
end Behavioral;
```

81.6.2 Elaborated design schematic



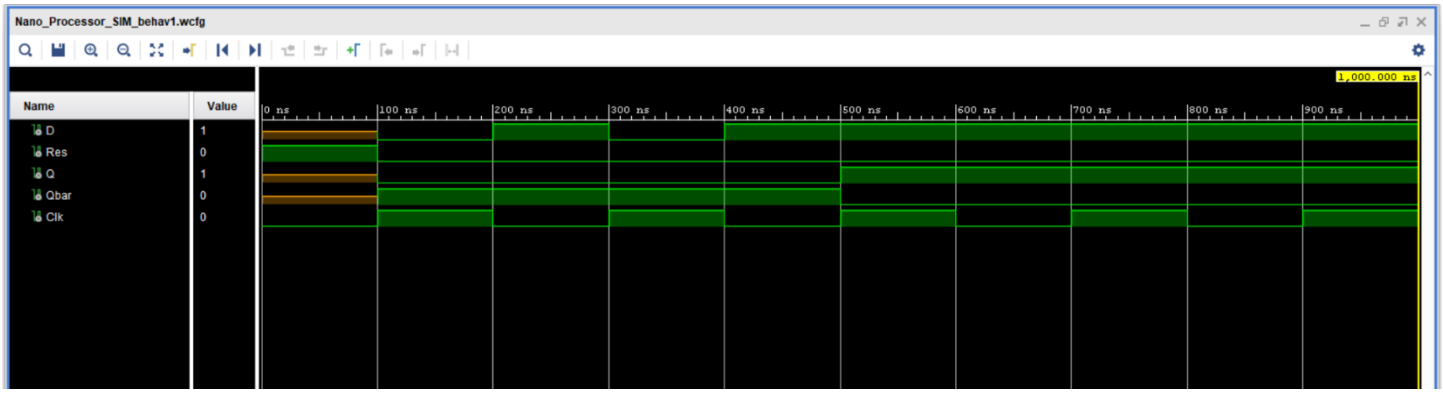
81.6.3 Simulation source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 02/27/2024 12:08:03 PM
-- Design Name:
-- Module Name: D_FF_Sim - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------
```

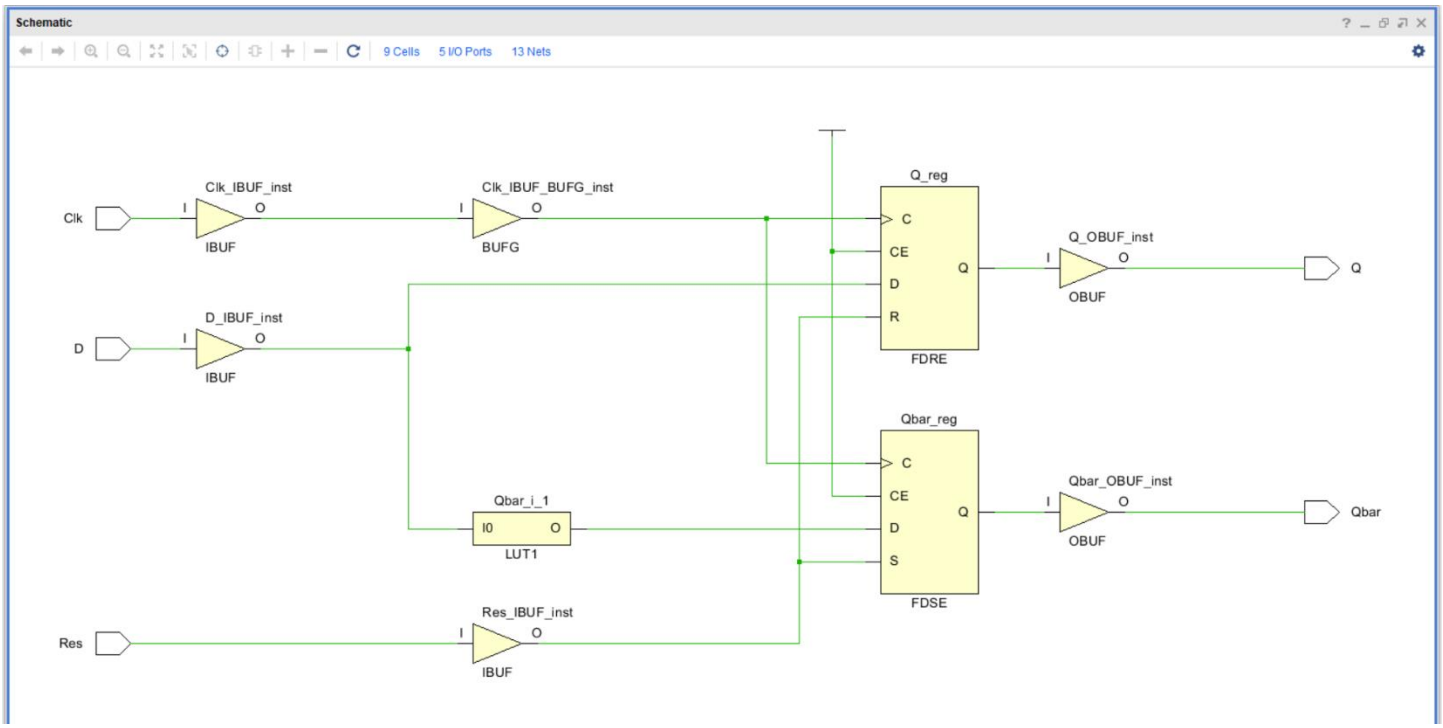
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- 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 D_FF_Sim is
--  Port ( );
end D_FF_Sim;

architecture Behavioral of D_FF_Sim is
component D_FF
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,Res,Q,Qbar : STD_LOGIC;
signal Clk : STD_LOGIC:= '0';


begin
UUT : D_FF
PORT MAP(
D=>D,
Res => Res,
Clk=>Clk,
Q =>Q,
Qbar=>Qbar
);

process
begin

wait for 100ns;
Clk <= NOT(Clk);
```

```
end process;

process
begin

Res <= '1';
wait for 100ns;
Res <= '0';

D <= '0';
wait for 100ns;
D <= '1';
wait for 100ns;
D <= '0';
wait for 100ns;
D <= '1';
wait;
end process;

end Behavioral;
```

81.6.4 Timing diagram



81.6.5 Implemented Design Schematic



4. Constrain file of main module

```
## Clock signal
set_property PACKAGE_PIN W5 [get_ports Clk]
    set_property IOSTANDARD LVCMOS33 [get_ports Clk]
    create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports Clk]

## LEDs
set_property PACKAGE_PIN U16 [get_ports {out_LED[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {out_LED[0]}]
set_property PACKAGE_PIN E19 [get_ports {out_LED[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {out_LED[1]}]
set_property PACKAGE_PIN U19 [get_ports {out_LED[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {out_LED[2]}]
set_property PACKAGE_PIN V19 [get_ports {out_LED[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {out_LED[3]}]
set_property PACKAGE_PIN P1 [get_ports {zero_LED}]
    set_property IOSTANDARD LVCMOS33 [get_ports {zero_LED}]
set_property PACKAGE_PIN L1 [get_ports {overflow_LED}]
    set_property IOSTANDARD LVCMOS33 [get_ports {overflow_LED}]


##7 segment display
```

```
set_property PACKAGE_PIN W7 [get_ports {seven_segment_in[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seven_segment_in[0]}]
set_property PACKAGE_PIN W6 [get_ports {seven_segment_in[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seven_segment_in[1]}]
set_property PACKAGE_PIN U8 [get_ports {seven_segment_in[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seven_segment_in[2]}]
set_property PACKAGE_PIN V8 [get_ports {seven_segment_in[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seven_segment_in[3]}]
set_property PACKAGE_PIN U5 [get_ports {seven_segment_in[4]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seven_segment_in[4]}]
set_property PACKAGE_PIN V5 [get_ports {seven_segment_in[5]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seven_segment_in[5]}]
set_property PACKAGE_PIN U7 [get_ports {seven_segment_in[6]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seven_segment_in[6]}]

set_property PACKAGE_PIN U2 [get_ports {Anode_Selector[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Anode_Selector[0]}]
set_property PACKAGE_PIN U4 [get_ports {Anode_Selector[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Anode_Selector[1]}]
set_property PACKAGE_PIN V4 [get_ports {Anode_Selector[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Anode_Selector[2]}]
set_property PACKAGE_PIN W4 [get_ports {Anode_Selector[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Anode_Selector[3]}]


##Buttons
set_property PACKAGE_PIN U18 [get_ports Reset]
    set_property IOSTANDARD LVCMOS33 [get_ports Reset]
```

5. Allocations of inputs and outputs on the BASYS3 Board

- Center Button --> for Reset operation
  - We reduced the clock speed of the internal clock of board from 100MHz to 1Hz. Therefore, hold Reset for at least 2 to 3 seconds to reset the registers and program counter.
- LED0-LED3 --> Display register 7 value
- Seven segment Display 1 --> Display register 7 value

- LED15 --> Overflow Flag
- LED14 --> Zero Flag
- LED07 -->Equal_Comparater Flag (Additional Features added Design)

6. Design Primitives and Slice Logic

6.1 Slice Logic

1. Slice Logic

Site Type	Used	Fixed	Available	Util%
Slice LUTs*	75	0	20800	0.36
LUT as Logic	75	0	20800	0.36
LUT as Memory	0	0	9600	0.00
Slice Registers	50	0	41600	0.12
Register as Flip Flop	45	0	41600	0.11
Register as Latch	5	0	41600	0.01
F7 Muxes	0	0	16300	0.00
F8 Muxes	0	0	8150	0.00

* Warning! The Final LUT count, after physical optimizations and full implementation, is typically lower. Run opt_design after synthesis, if not already completed, for a more realistic count.

6.2 Primitives

7. Primitives

Ref Name	Used	Functional Category
FDRE	45	Flop & Latch
LUT6	28	LUT
LUT4	24	LUT
LUT2	21	LUT
OBUF	18	IO
LUT5	16	LUT
LUT3	12	LUT
CARRY4	8	CarryLogic
LDCE	5	Flop & Latch
IBUF	2	IO
LUT1	1	LUT
BUFG	1	Clock

7. Optimizations

- We used "S_vec <= S & S & S & S;" in our multiplexers to generate a vector from a bit. So, we can get inputs as single bit or a small vector. Therefore, we reduced the wires from instruction decoder to multiplexer.
- We used direct logic for register instead of using so many D-flip-flops. Therefore, we improved the readability of the design.

8. Additional features

We extended our nano processor by adding JMP, COMPARATOR, MULTIPLIER and COMPLEMENT operations.

- JMP - Jump to a specific line in program rom. (without considering register value)
- Comparator – Compare two 4-bit binary values and gives "1" as the output if they are equal.
- Multiplier – get two 4-bit binary values as inputs and gives their multiplication as the output.
- Complement – output the one's complement of a 4-bit binary value.

The set of instructions

Introduction	Description	Format
MOVI R, d	Move immediate value d to register R, i.e., R ← d, R ∈ [0, 7], d ∈ [0, 15]	0 1 0 R R R 0 0 0 d d d d
ADD Ra, Rb	Add values in registers Ra and Rb and store the result in Ra, i.e., Ra ← Ra + Rb (Ra, Rb ∈ [0, 7])	0 0 0 Ra Ra Ra Rb Rb Rb 0 0 0 0
NEG R	2's complement of registers R, i.e., R ← – R, R ∈ [0, 7]	0 0 1 R R R 0 0 0 0 0 0 0
JZR R, d	Jump if value in register R is 0, i.e., If R == 0 PC ← d; Else PC ← PC + 1; (R ∈ [0, 7], d∈ [0, 7])	0 1 1 R R R 0 0 0 0 d d d
MUL Ra, Rb	Multiply values in registers Ra and Rb and store the result in Ra, i.e., Ra ← Ra x Rb (Ra, Rb ∈ [0, 7])	1 1 0 Ra Ra Ra Rb Rb Rb 0 0 0 0

JMP R, d	Jump to d th instruction, PC ← d; d∈ [0, 7])	1 0 0 0 0 0 0 0 0 0 d d d
NOT R	Complement of registers R, i.e., R ← complement of (R), R ∈ [0, 7]	1 1 1 R R R 0 0 0 0 0 0 0 0
COM Ra, Rb	Compare values in registers Ra and Rb give output as a LED output . (Ra, Rb ∈ [0, 7])	1 0 1 Ra Ra Ra Rb Rb Rb 0 0 0 0

9 Additional features components

9.1 Multiplier

9.1.1 Design source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 02/27/2024 04:24:05 PM
-- Design Name:
-- Module Name: Multiplier_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 Multiplier_4 is
Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
```

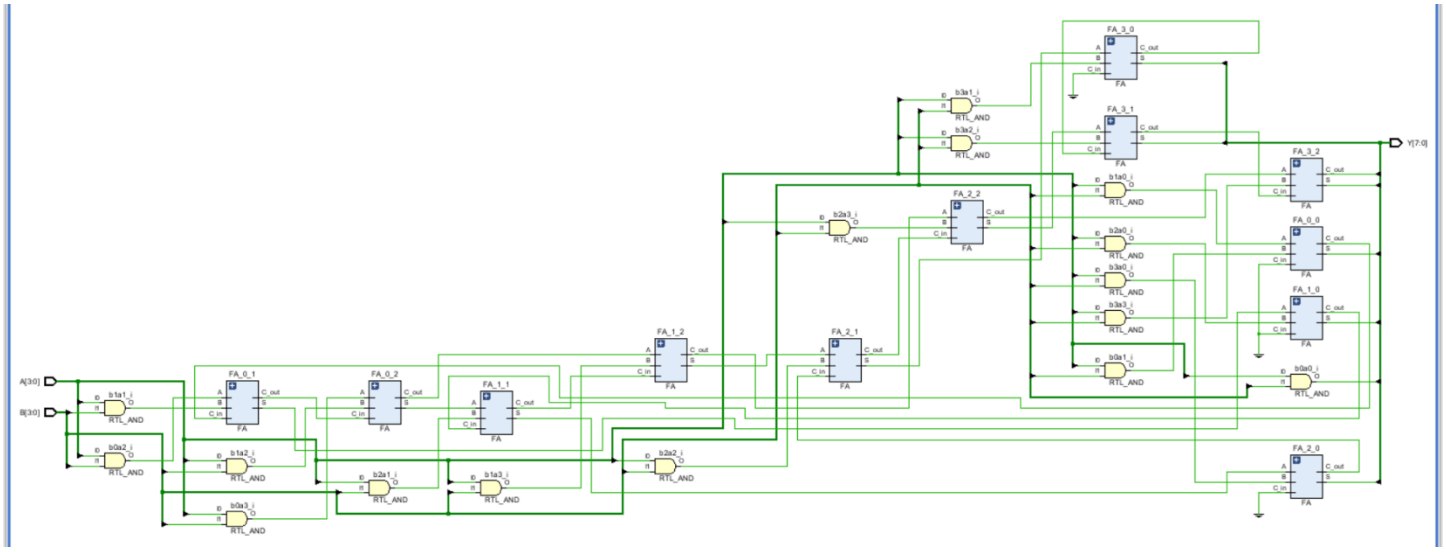
```
B : in STD_LOGIC_VECTOR (3 downto 0);
Y : out STD_LOGIC_VECTOR (7 downto 0));
end Multiplier_4;
architecture Behavioral of Multiplier_4 is
component FA is
Port (
A : in std_logic;
B : in std_logic;
C_in : in std_logic;
S : out std_logic;
C_out : out std_logic
);
end component;
signal b0a0, b0a1, b0a2, b0a3, b1a0, b1a1, b1a2, b1a3, b2a0, b2a1, b2a2, b2a3,
b3a0, b3a1, b3a2, b3a3 : std_logic;
signal s_0_0 , s_0_1 , s_1_0 , s_0_2 , s_1_2 , s_1_1 , s_2_0 , s_2_1 , s_2_2 ,
s_3_0 , s_3_1 ,s_3_2 : std_logic;
signal c_0_0 , c_0_1 , c_1_0 , c_0_2 , c_1_2 , c_1_1 , c_2_0 , c_2_1 , c_2_2 ,
c_3_0 , c_3_1 ,c_3_2 : std_logic;
begin
FA_0_0 : FA port map
(
A => b1a0,
B => b0a1,
C_in => '0'
,
S => s_0_0,
C_out => c_0_0
);
FA_0_1 : FA port map
(
A => b0a2,
B => b1a1,
C_in => c_0_0,
S => s_0_1,
C_out => c_0_1
);
FA_1_0 : FA port map
(
A => s_0_1,
B => b2a0,
C_in => '0'
,
S => s_1_0,
C_out => c_1_0
);
```

```
FA_0_2 : FA port map
(
A => b0a3,
B => b1a2,
C_in => c_0_1,
S => s_0_2,
C_out => c_0_2
);
FA_1_1 : FA port map
(
A => s_0_2,
B => b2a1,
C_in => c_1_0,
S => s_1_1,
C_out => c_1_1
);
FA_2_0 : FA port map
(
A => s_1_1,
B => b3a0,
C_in => '0'
,
S => s_2_0,
C_out => c_2_0
);
FA_1_2 : FA port map
(
A => c_0_2,
B => b1a3 ,
C_in => c_1_1,
S => s_1_2,
C_out => c_1_2
);
FA_2_1 : FA port map
(
A => s_1_2,
B => b2a2,
C_in => c_2_0,
S => s_2_1,
C_out => c_2_1
);
FA_3_0 : FA port map
(
A => s_2_1,
B => b3a1,
C_in => '0'
```

```
,
S => s_3_0,
C_out => c_3_0
);
FA_2_2 : FA port map
(
A => c_1_2,
B => b2a3,
C_in => c_2_1,
S => s_2_2,
C_out => c_2_2
);
FA_3_1 : FA port map
(
A => s_2_2,
B => b3a2,
C_in => c_3_0,
S => s_3_1,
C_out => c_3_1
);
FA_3_2 : FA port map
(
A => c_2_2,
B => b3a3,
C_in => c_3_1,
S => s_3_2,
C_out => c_3_2
);
b0a0 <= a(0) AND b(0);
b1a0 <= a(0) AND b(1);
b2a0 <= a(0) AND b(2);
b3a0 <= a(0) AND b(3);
b0a1 <= a(1) AND b(0);
b1a1 <= a(1) AND b(1);
b2a1 <= a(1) AND b(2);
b3a1 <= a(1) AND b(3);
b0a2 <= a(2) AND b(0);
b1a2 <= a(2) AND b(1);
b2a2 <= a(2) AND b(2);
b3a2 <= a(2) AND b(3);
b0a3 <= a(3) AND b(0);
b1a3 <= a(3) AND b(1);
b2a3 <= a(3) AND b(2);
b3a3 <= a(3) AND b(3);
Y(0) <= b0a0;
Y(1) <= s_0_0;
```

```
Y(2) <= s_1_0;
Y(3) <= s_2_0;
Y(4) <= s_3_0;
Y(5) <= s_3_1;
Y(6) <= s_3_2;
Y(7) <= c_3_2;
end Behavioral;
```

9.1.2 Elaborated design schematic



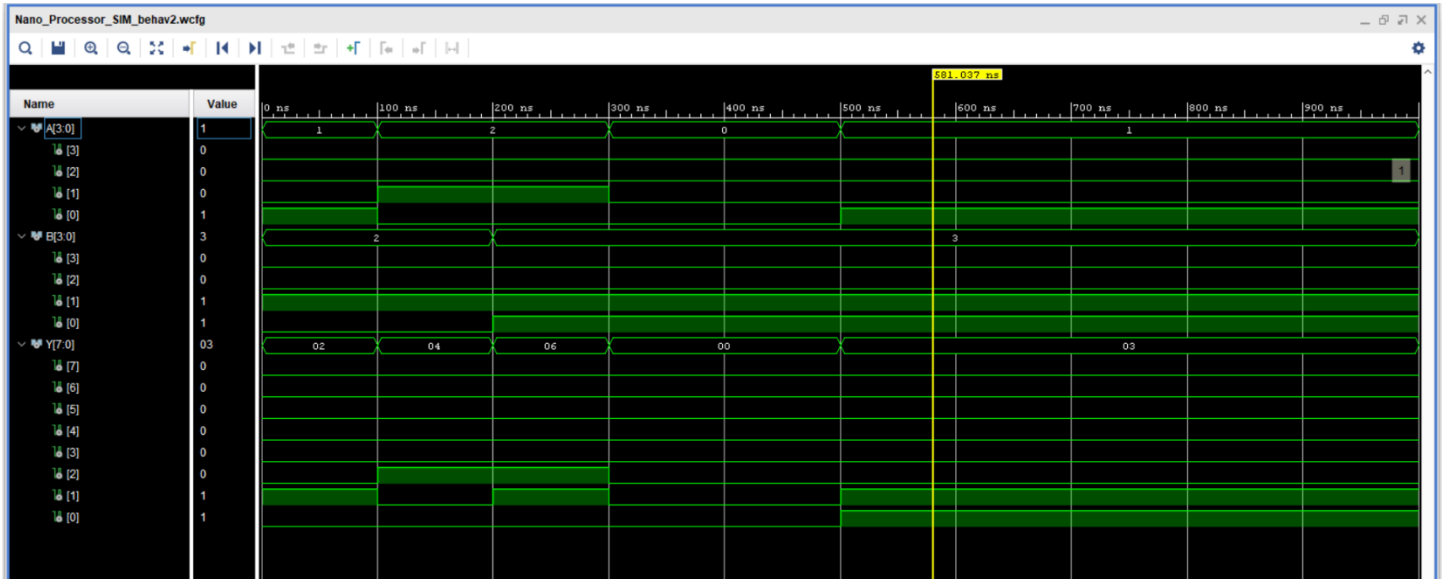
9.1.3 Simulation source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 02/27/2024 07:36:40 PM
-- Design Name:
-- Module Name: TB_Multiplier_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 TB_Multiplier_4 is
--  Port ( );
end TB_Multiplier_4;

architecture Behavioral of TB_Multiplier_4 is

component Multiplier_4 is
  Port ( A : in STD_LOGIC_VECTOR (0 to 3);
           B : in STD_LOGIC_VECTOR (0 to 3);
           Y : out STD_LOGIC_VECTOR (0 to 7));
end component;
signal A,B : STD_LOGIC_VECTOR(3 downto 0);
signal Y : STD_LOGIC_VECTOR(7 downto 0);

begin
UUT : Multiplier_4 PORT MAP (
A=>A,
B=>B,
Y=>Y
);
process begin
A<="0001";
B<="0010";
wait for 100 ns;
A<="0010";
wait for 100 ns;
B<="0011";
wait for 100 ns;
A<="0000";
```

```
wait for 100 ns;
B<="0011";
wait for 100 ns;
A<="0001";
wait ;

end process;

end Behavioral;
```

9.1.4 Timing diagram



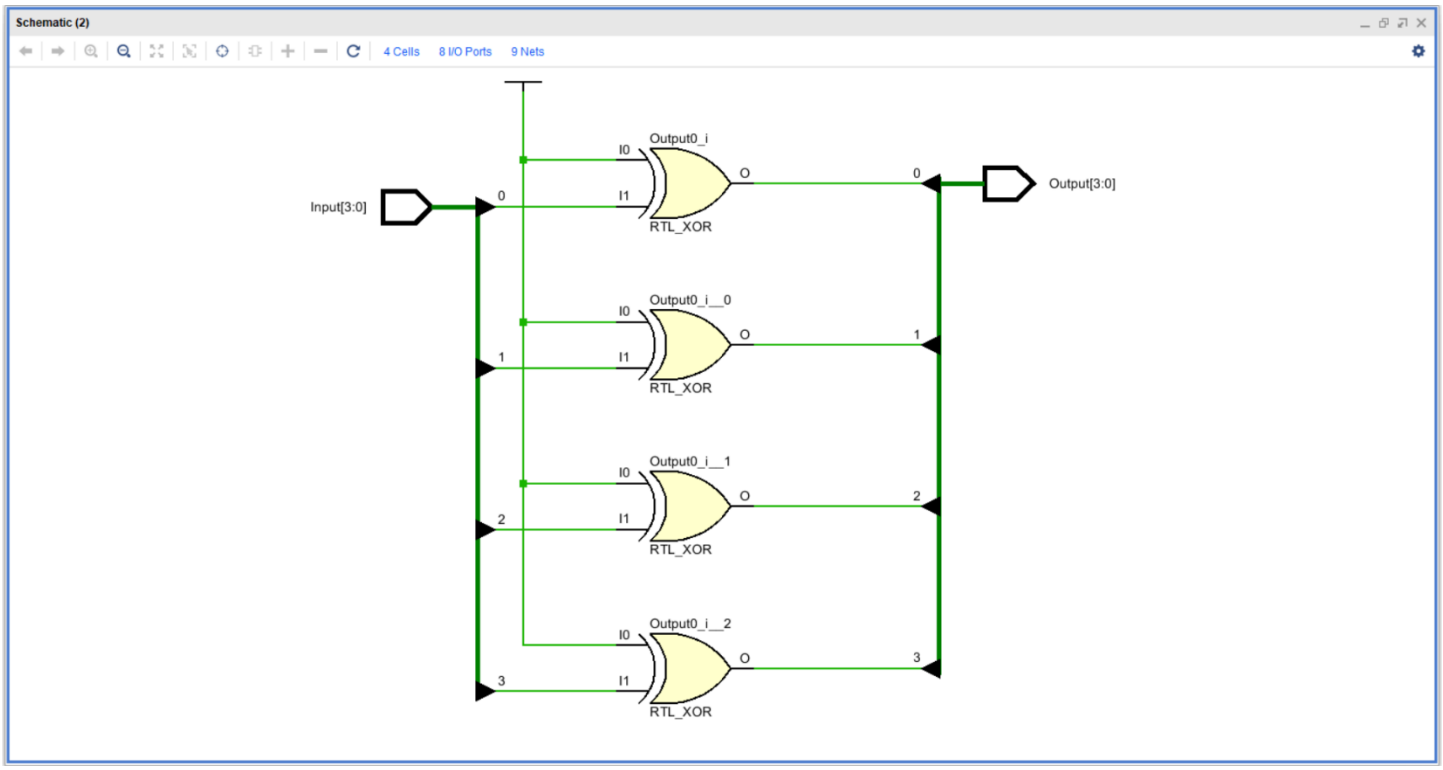
9.2 Complement

9.2.1 Design source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/21/2024 12:37:06 PM
-- Design Name:
-- Module Name: Complement - 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 Complement is
   Port ( Input : in STD_LOGIC_VECTOR (3 downto 0);
          Output : out STD_LOGIC_VECTOR (3 downto 0));
end Complement;

architecture Behavioral of Complement is

begin

Output(0)<= '1' XOR Input(0);
Output(1)<= '1' XOR Input(1);
Output(2)<= '1' XOR Input(2);
Output(3)<= '1' XOR Input(3);

end Behavioral;
```

9.2.2 Elaborated design schematic

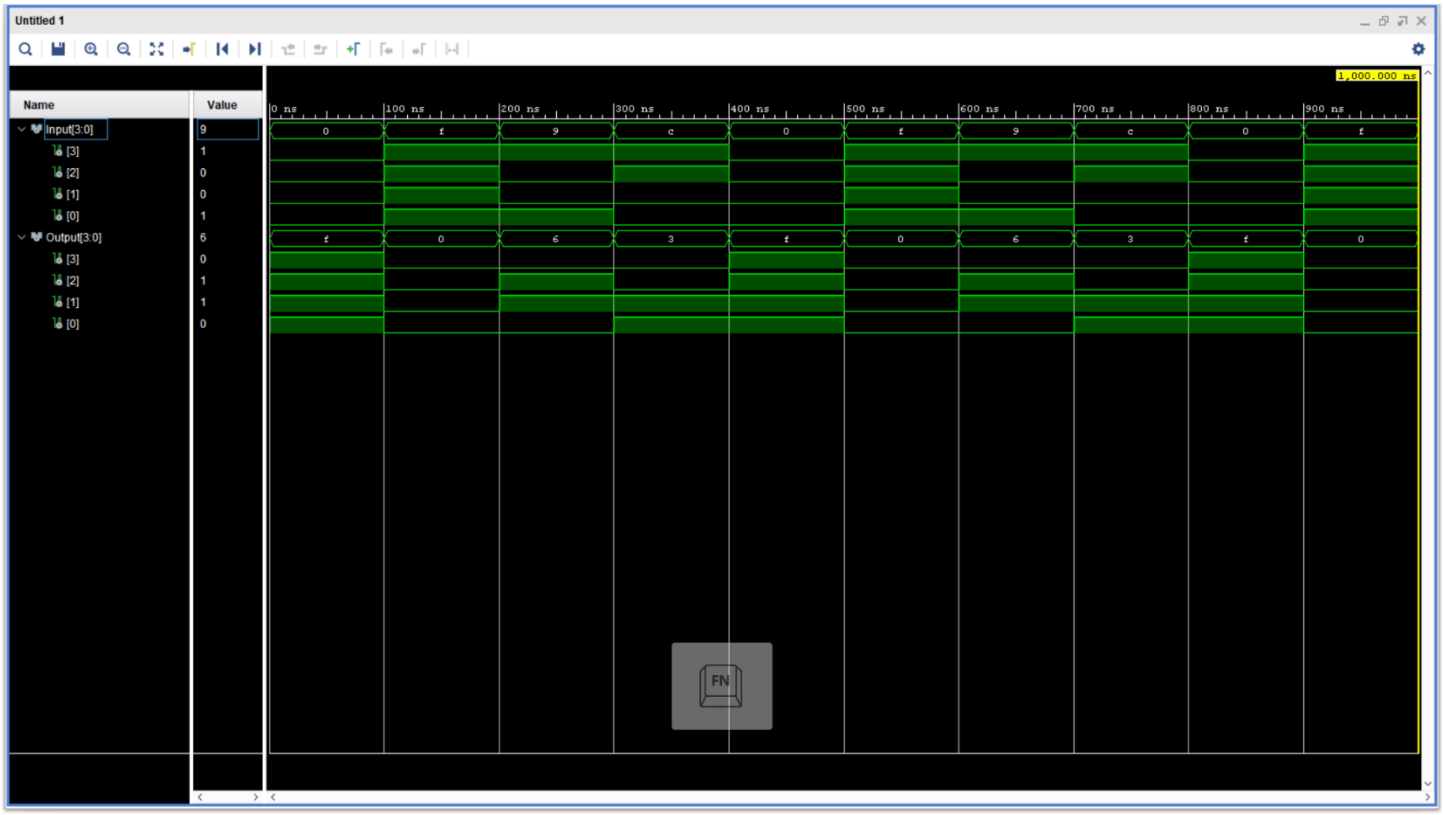


9.2.3 Simulation source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/26/2024 08:20:23 AM
-- Design Name:
-- Module Name: TB_compliment - 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 TB_compliment is
--  Port ( );
end TB_compliment;

architecture Behavioral of TB_compliment is

component Complement is
   Port ( Input : in STD_LOGIC_VECTOR (3 downto 0);
          Output : out STD_LOGIC_VECTOR (3 downto 0));
end component;

signal Input :  STD_LOGIC_VECTOR (3 downto 0);
signal Output :  STD_LOGIC_VECTOR (3 downto 0);

begin

UUT : Complement
port map(
    Input => Input,
    Output => Output
);

process begin

        input <= "0000";
        wait for 100 ns;
        input <= "1111";
        wait for 100 ns;
        input <= "1001";
        wait for 100 ns;
        input <= "1100";
        wait for 100 ns;
end process;
end Behavioral;
```

9.2.4 Timing diagram



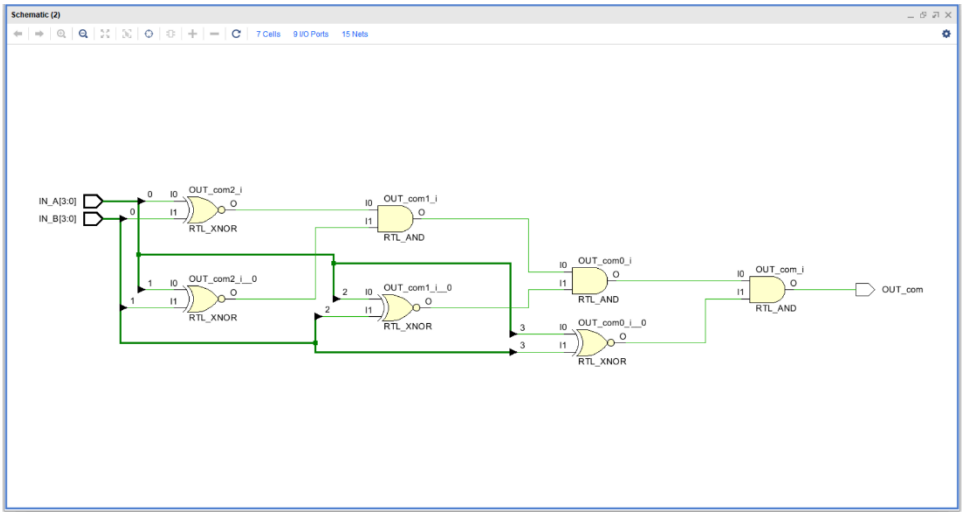
9.3 Comparator

9.3.1 Design source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/24/2024 02:34:37 PM
-- Design Name:
-- Module Name: Equal_comparator - 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 Equal_comparator is
    Port ( IN_A : in STD_LOGIC_VECTOR (3 downto 0);
           IN_B : in STD_LOGIC_VECTOR (3 downto 0);
           OUT_com : out STD_LOGIC);
end Equal_comparator;

architecture Behavioral of Equal_comparator is

begin
OUT_com <= (IN_A(0) XNOR IN_B(0)) AND
            (IN_A(1) XNOR IN_B(1))AND
            (IN_A(2) XNOR IN_B(2))AND
            (IN_A(3) XNOR IN_B(3));

end Behavioral;
```

9.3.2 Elaborated design schematic



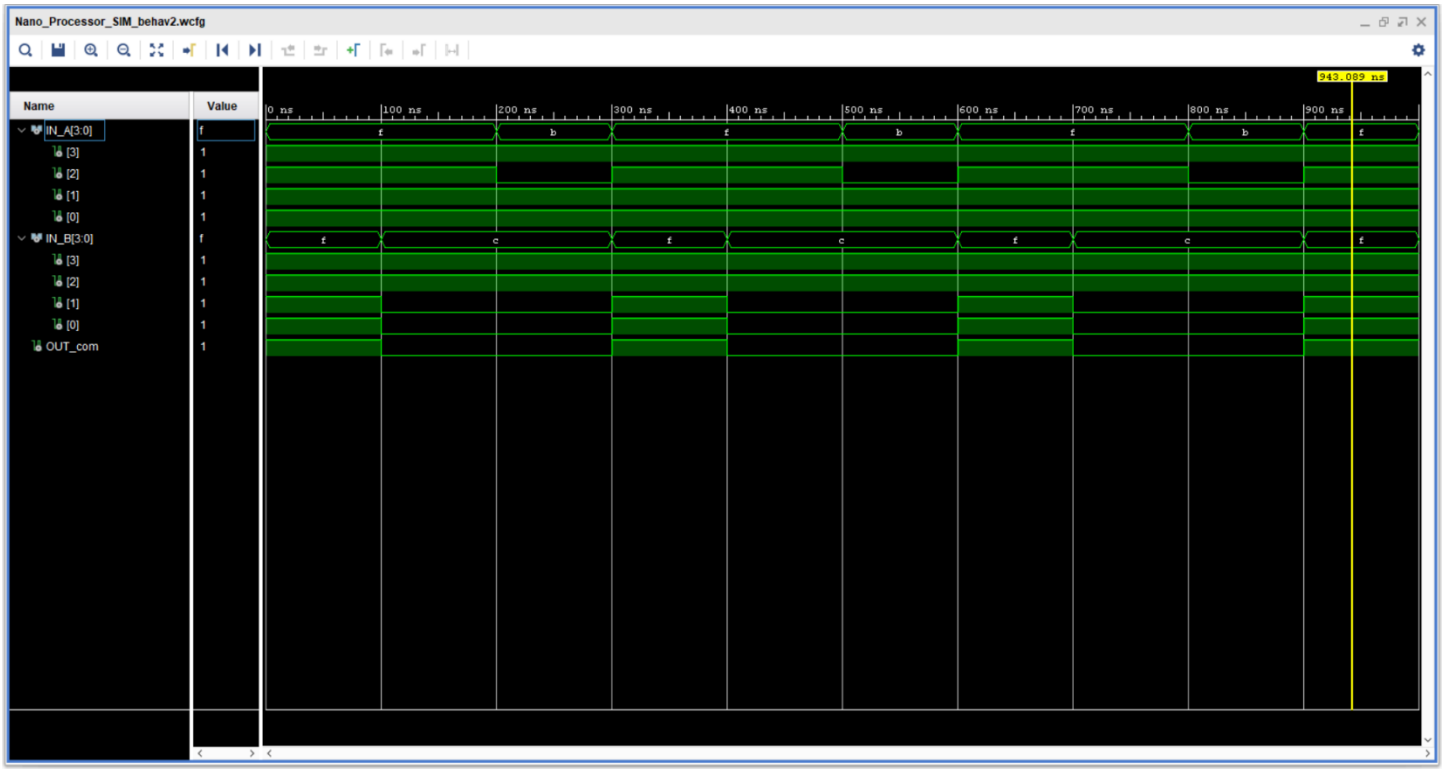
9.3.3 Simulation source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/24/2024 02:43:49 PM
-- Design Name:
-- Module Name: TB_Equal_comparator - 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 TB_Equal_comparator is
--  Port ( );
end TB_Equal_comparator;

architecture Behavioral of TB_Equal_comparator is

component Equal_comparator is
    Port ( IN_A : in STD_LOGIC_VECTOR (3 downto 0);
           IN_B : in STD_LOGIC_VECTOR (3 downto 0);
           OUT_com : out STD_LOGIC);
end component;

signal IN_A : STD_LOGIC_VECTOR (3 downto 0);
signal IN_B : STD_LOGIC_VECTOR (3 downto 0);
signal OUT_com : STD_LOGIC;

begin

UUT : Equal_comparator
port map (
    IN_A => IN_A,
    IN_B => IN_B,
    OUT_com => OUT_com
);

process begin

        IN_A <= "1111";
        IN_B <= "1111";
        wait for 100 ns;

        IN_A <= "1111";
        IN_B <= "1100";
        wait for 100 ns;

        IN_A <= "1011";
        IN_B <= "1100";
        wait for 100 ns;
```

```
end process;
end Behavioral;
```

9.3.4 Timing diagram



10 Updated Module

10.1 Updated main module

10.1.1 Design source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/10/2024 08:45:14 AM
-- Design Name:
-- Module Name: Nano_Processor - 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 Nano_Processor is
    Port ( Clk : in STD_LOGIC; --clock of the basys 3 board
           Reset : in STD_LOGIC; -- reset signal(connect to button of basys 3)
           overflow_LED : out STD_LOGIC; --overflow detector connected to the led of basys3
           zero_LED : out STD_LOGIC:='0'; --zero flag detector connected to the led of basys3
           out_LED : out STD_LOGIC_vector(3 downto 0); --we can see the value in register
seven using this
           seven_segment_in : out STD_LOGIC_vector(6 downto 0); --give the value inside reg
seven as an input to sevensegment display
           Anode_Selector : out STD_LOGIC_VECTOR (3 downto 0) ;--select the seven segment
display
           comparator_out : out STD_LOGIC
             );
end Nano_Processor;

architecture Behavioral of Nano_Processor is

--component used in nano processor
component Instruction_Decoder is
    Port ( Instruction : in STD_LOGIC_VECTOR (12 downto 0);
           Register_Check_For_Jump : in STD_LOGIC_VECTOR (3 downto 0);
```

```
           Register_Enable : out STD_LOGIC_VECTOR (2 downto 0);
           Immediate_Value : out STD_LOGIC_VECTOR (3 downto 0);
           Load_Selector : out STD_LOGIC;
           A_Register_Selector : out STD_LOGIC_VECTOR (2 downto 0);
           B_Register_Selector : out STD_LOGIC_VECTOR (2 downto 0);
           Abb_Sub_Selector : out STD_LOGIC;
           Jump_Flag : out STD_LOGIC;
           Jump_Address : out STD_LOGIC_VECTOR (2 downto 0);
           component_selector: out STD_LOGIC_VECTOR (1 downto 0)

          );
end component;

component Complement is
   Port ( Input : in STD_LOGIC_VECTOR (3 downto 0);
          Output : out STD_LOGIC_VECTOR (3 downto 0));
end component;

component Equal_comparator is
    Port ( IN_A : in STD_LOGIC_VECTOR (3 downto 0);
           IN_B : in STD_LOGIC_VECTOR (3 downto 0);
           OUT_com : out STD_LOGIC);
end component;



component tri_state_buffer is
    Port (
        enable : in  STD_LOGIC;
        input_data : in  STD_LOGIC_VECTOR (3 downto 0);
        output_data : out  STD_LOGIC_VECTOR (3 downto 0)
    );
end component;

component four_way_4_bit_mux is
    Port ( I_4 : in STD_LOGIC_VECTOR (15 downto 0);
           S_4 : in STD_LOGIC_VECTOR (1 downto 0);
           Y_4 : out STD_LOGIC_VECTOR (3 downto 0));
end component;

component Multiplier_4 is
Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
B : in STD_LOGIC_VECTOR (3 downto 0);
Y : out STD_LOGIC_VECTOR (7 downto 0));
end component;
```

```
component twoWay_4bit_Mux is
    Port ( D : in STD_LOGIC_VECTOR (7 downto 0);
           S : in STD_LOGIC;
           Y : out STD_LOGIC_VECTOR (3 downto 0));
end component;

component Register_bank is
    Port (
    register_enable:in STD_LOGIC_VECTOR (2 downto 0);--to decide which register going to be
enabled
    data:in STD_LOGIC_VECTOR (3 downto 0);
    clk:in STD_LOGIC; --to input slow clock signal
    clr:in STD_LOGIC; --related to reset button
    reg_out:out STD_LOGIC_VECTOR (31 downto 0)
    );
end component;

component four_Bit_Add_Sub_Unit 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);
           Zero_flag : out STD_LOGIC;
           overflow  : out STD_LOGIC;
           C_out : out STD_LOGIC);

end component ;

component Program_Counter is
    Port ( D_in : in STD_LOGIC_VECTOR (0 to 2);
           Push : in STD_LOGIC;
           Clk : in STD_LOGIC;
           D_out : out STD_LOGIC_VECTOR (0 to 2));
end component ;

component  Program_ROM is
    Port ( Memory_select : in STD_LOGIC_VECTOR (2 downto 0);
           Instruction_Bus : out STD_LOGIC_VECTOR (12 downto 0));
end component ;

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

```
component  twoWay_3bit_Mux is
Port ( D : in STD_LOGIC_VECTOR (5 downto 0);
       S : in STD_LOGIC;
       Y : out STD_LOGIC_VECTOR (2 downto 0));
end component ;

component eightWay_4bit_Mux is
    Port ( D : in STD_LOGIC_VECTOR (31 downto 0);
           S : in STD_LOGIC_VECTOR (2 downto 0);
           Y : out STD_LOGIC_VECTOR (3 downto 0)
           );
end component;

component Adder_3_Bit is
    Port ( A : in STD_LOGIC_VECTOR (2 downto 0);
           B : in STD_LOGIC_VECTOR (2 downto 0);
           S : out STD_LOGIC_VECTOR (2 downto 0);
           C_in  : in STD_LOGIC ;
           overflow : out STD_LOGIC);
end component;

component LUT_16_7 is
    Port ( address : in STD_LOGIC_VECTOR (3 downto 0);
           data : out STD_LOGIC_VECTOR (6 downto 0));
end component;

--signals
signal register_enable: std_logic_vector(2 downto 0);
signal load_selector: std_logic;
signal immediate_value: std_logic_vector(3 downto 0);
signal A_register_select: std_logic_vector(2 downto 0);
signal B_register_select: std_logic_vector(2 downto 0);
signal add_sub_select: std_logic;
signal jump_flag: std_logic;
signal adderss_to_jump: std_logic_vector(2 downto 0);
signal ins_bus : std_logic_vector(12 downto 0);
signal mem_sel : std_logic_vector(2 downto 0);
signal three_adder_out : std_logic_vector(2 downto 0);
signal three_bit_mux_out : std_logic_vector(2 downto 0);
signal two_way_4_bit_mux_out : std_logic_vector(3 downto 0);
signal A_mux_out : std_logic_vector(3 downto 0);
signal B_mux_out : std_logic_vector(3 downto 0);
signal add_sub_out : std_logic_vector(3 downto 0);
signal multiplier_out : std_logic_vector(3 downto 0);
signal component_out : std_logic_vector(3 downto 0);
signal reg_bank_out : std_logic_vector(31 downto 0);
```

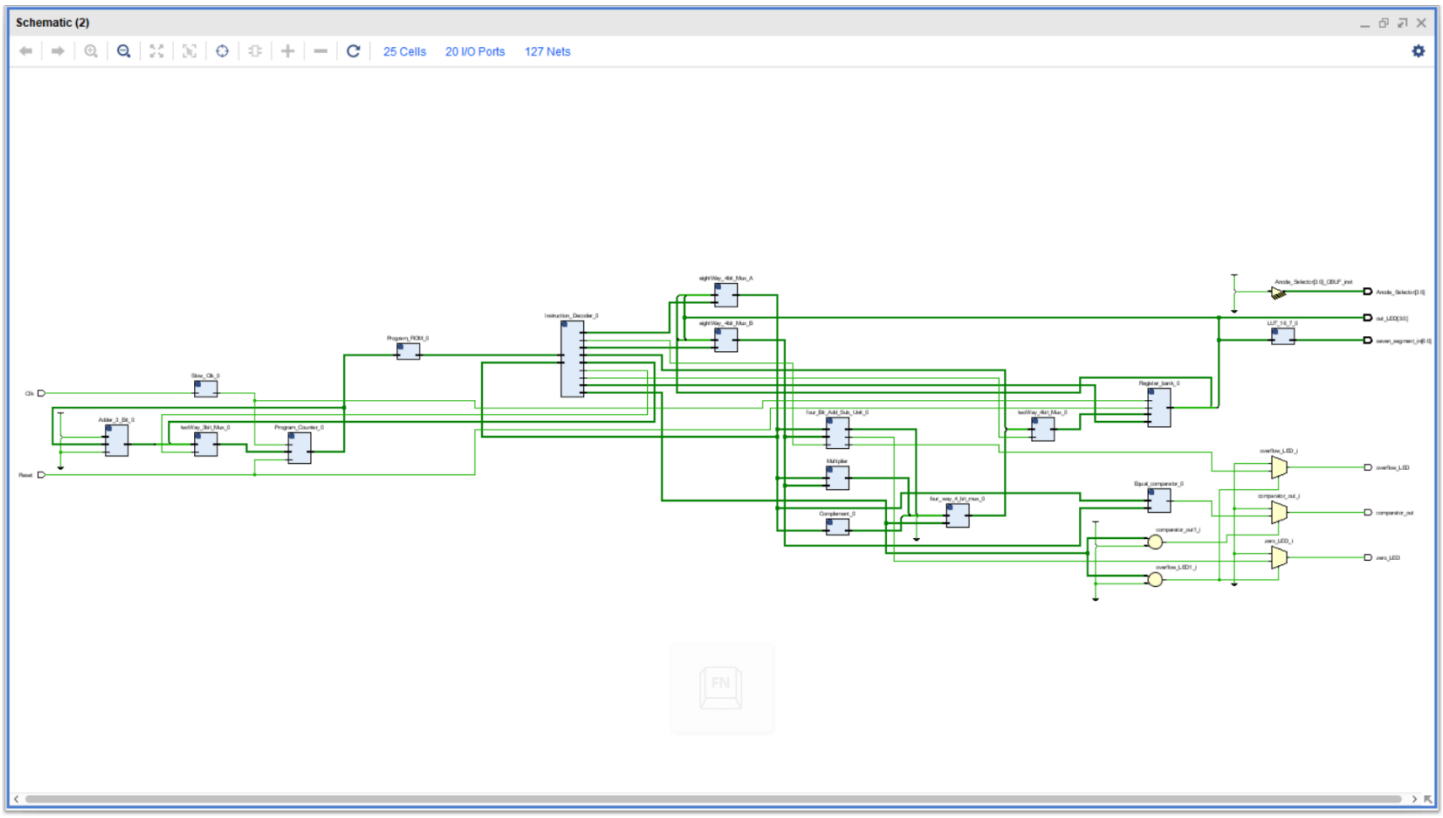
```
signal clock_out: std_logic;
signal c_out: std_logic;
signal overflow_3_bit: std_logic;
signal Y :  STD_LOGIC_VECTOR (7 downto 0);
signal component_selector:STD_LOGIC_VECTOR (1 downto 0);
signal complement_out : std_logic_vector(3 downto 0);
signal  comparator_out_LED :  STD_LOGIC;
signal overflow_LED0 :  STD_LOGIC; --overflow detector connected to the led of basys3
signal zeroflag_LED0 :  STD_LOGIC; --zero flag detector connected to the led of basys3
begin

-- port mappings
Instruction_Decoder_0 : Instruction_Decoder
        port map(
              Instruction => ins_bus,
              Register_Check_For_Jump => A_mux_out,
              Register_Enable  => register_enable ,
              Immediate_Value => immediate_value,
              Load_Selector => load_selector,
              A_Register_Selector  => A_register_select,
              B_Register_Selector   => B_register_select,
              Abb_Sub_Selector  => add_sub_select,
              Jump_Flag  => jump_flag,
              Jump_Address => adderss_to_jump,
              component_selector =>component_selector
            );

twoWay_4bit_Mux_0: twoWay_4bit_Mux
Port map ( D(7 downto 4) => immediate_value,
           D(3 downto 0) => component_out,
           S => load_selector,
           Y => two_way_4_bit_mux_out);

four_way_4_bit_mux_0: four_way_4_bit_mux
Port map (
           I_4(15 downto 12) =>complement_out,
           I_4(11 downto 8) =>"0000",
           I_4(7 downto 4) => multiplier_out,
           I_4(3 downto 0) => add_sub_out,
           S_4 => component_selector,
           Y_4 => component_out);

Equal_comparator_0 : Equal_comparator
 Port map(
           IN_A => A_mux_out,
           IN_B => B_mux_out,
```

```
           OUT_com =>comparator_out_LED);


Complement_0:Complement
   Port map ( Input =>A_mux_out,
          Output =>complement_out);


Multiplier : Multiplier_4
Port map(   A => A_mux_out,
            B => B_mux_out,
            Y =>Y);


Register_bank_0 :Register_bank
Port map ( register_enable => register_enable,
           data=> two_way_4_bit_mux_out,
           clk=>clock_out,
           clr=>Reset,
           reg_out(31 downto 0)=>reg_bank_out(31 downto 0)
            );

four_Bit_Add_Sub_Unit_0 :four_Bit_Add_Sub_Unit
Port map ( A=>A_mux_out ,
           B=>B_mux_out,
           M=> add_sub_select,
           S=> add_sub_out,
           Zero_flag =>Zeroflag_LED0,
           overflow=>overflow_LED0,
           C_out=>c_out);

Program_Counter_0: Program_Counter
Port map ( D_in => three_bit_mux_out,
           Push => Reset,
           Clk =>clock_out,
           D_out =>  mem_sel);

Program_ROM_0:Program_ROM
Port map ( Memory_select =>  mem_sel,
           Instruction_Bus=> ins_bus);

Slow_Clk_0: Slow_Clk
Port map ( Clk_in=>Clk,
           Clk_out=>clock_out);
```

```
twoWay_3bit_Mux_0 : twoWay_3bit_Mux
Port map ( D(5 downto 3) => adderss_to_jump,
           D(2 downto 0) => three_adder_out,
           S => jump_flag,
           Y => three_bit_mux_out);

eightWay_4bit_Mux_A : eightWay_4bit_Mux
 Port map(   D =>reg_bank_out ,
          S => A_register_select,
          Y => A_mux_out
          );

eightWay_4bit_Mux_B : eightWay_4bit_Mux
 Port map( D => reg_bank_out,
          S => B_register_select,
          Y => B_mux_out
          );



Adder_3_Bit_0 : Adder_3_Bit
    Port map ( A => "001",
           B => mem_sel,
           S => three_adder_out,
           C_in => '0',
           overflow => overflow_3_bit
           );

LUT_16_7_0 :LUT_16_7
    Port map (
        address => reg_bank_out(31 downto 28),
        data => seven_segment_in
        );

out_LED <= reg_bank_out(31 downto 28);

multiplier_out <= Y(3 downto 0);

Anode_Selector <= "1110";
process(component_selector ,comparator_out_LED)
begin
if(component_selector/="10") then
comparator_out<='0';
else
comparator_out<= comparator_out_LED;
```

```
end if;
end process;
process(component_selector ,overflow_LED0,zeroflag_LED0)
begin
if(component_selector/="00") then
overflow_LED <='0';
zero_LED <='0';
else
overflow_LED <=overflow_LED0;
zero_LED <=zeroflag_LED0;
end if;
end process;


end Behavioral;
```

10.1.1 Elaborated design schematic



10.1.2 Elaborated design schematic

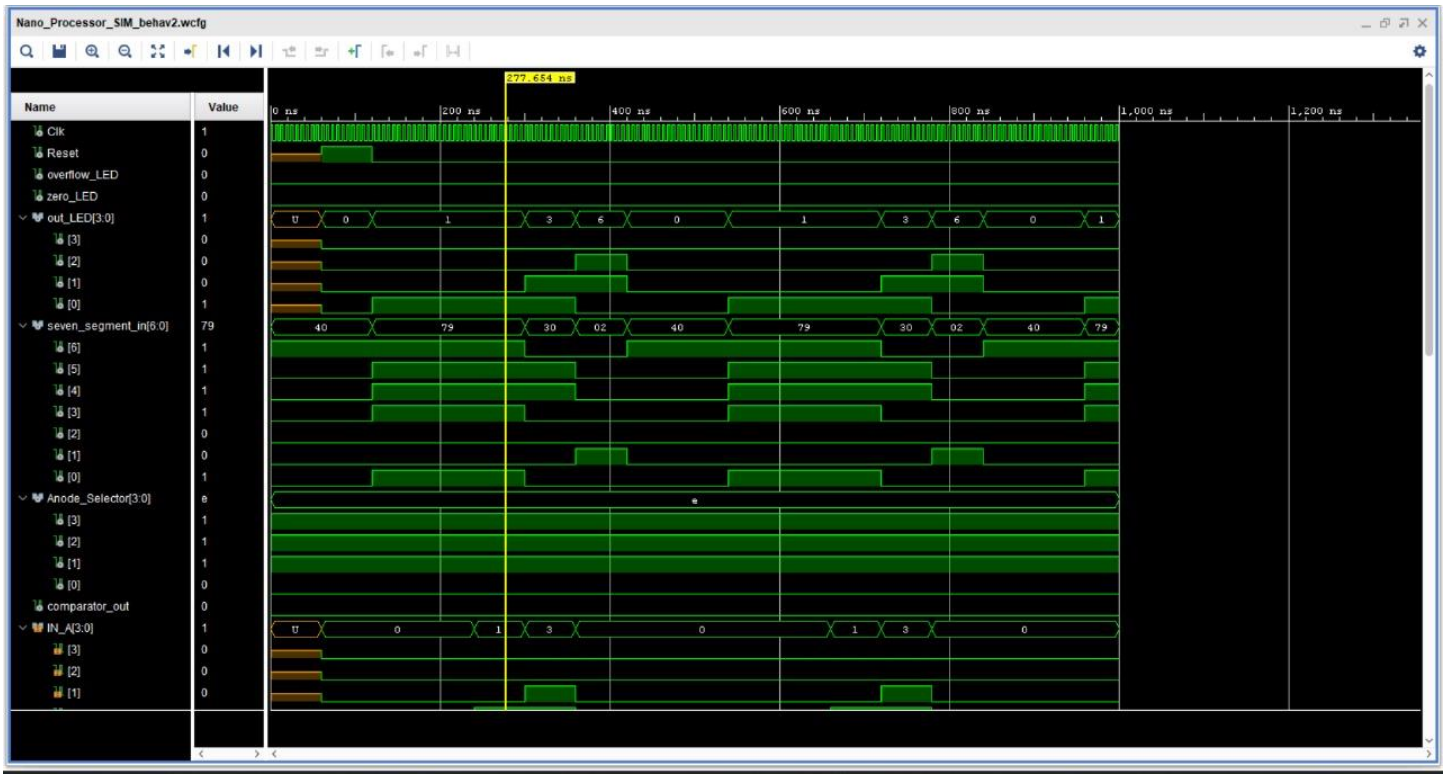
```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/10/2024 03:13:54 PM
-- Design Name:
-- Module Name: Nano_Processor_SIM - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- 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 Nano_Processor_SIM is
--  Port ( );
end Nano_Processor_SIM;

architecture Behavioral of Nano_Processor_SIM is
```

```
component Nano_Processor is
    Port ( Clk : in STD_LOGIC;
           Reset : in STD_LOGIC;
           overflow_LED : out STD_LOGIC;
           zero_LED : out STD_LOGIC;
           out_LED : out STD_LOGIC_vector(3 downto 0);
           seven_segment_in : out STD_LOGIC_vector(6 downto 0);
           Anode_Selector : out STD_LOGIC_VECTOR (3 downto 0);
           comparator_out : out STD_LOGIC
             );
end component;


signal Clk  :  STD_LOGIC;
signal Reset  :  STD_LOGIC;
signal overflow_LED : STD_LOGIC;
signal zero_LED :  STD_LOGIC;
signal out_LED :  STD_LOGIC_vector(3 downto 0);
signal seven_segment_in :  STD_LOGIC_vector(6 downto 0);
signal Anode_Selector :STD_LOGIC_VECTOR (3 downto 0);
signal comparator_out : STD_LOGIC;

begin

UUT : Nano_Processor
port map(
        Clk =>Clk,
        Reset =>Reset,
        overflow_LED =>overflow_LED,
        zero_LED =>zero_LED ,
        out_LED=>out_LED,
        seven_segment_in =>seven_segment_in,
        Anode_Selector =>Anode_Selector,
        comparator_out =>comparator_out
        );

-- process for the clock
Clock : process begin
 Clk<='1';
 wait for 3 ns;
 Clk<='0';
 wait for 3 ns;
 end process;

process begin
--process of the processor(reset)
```

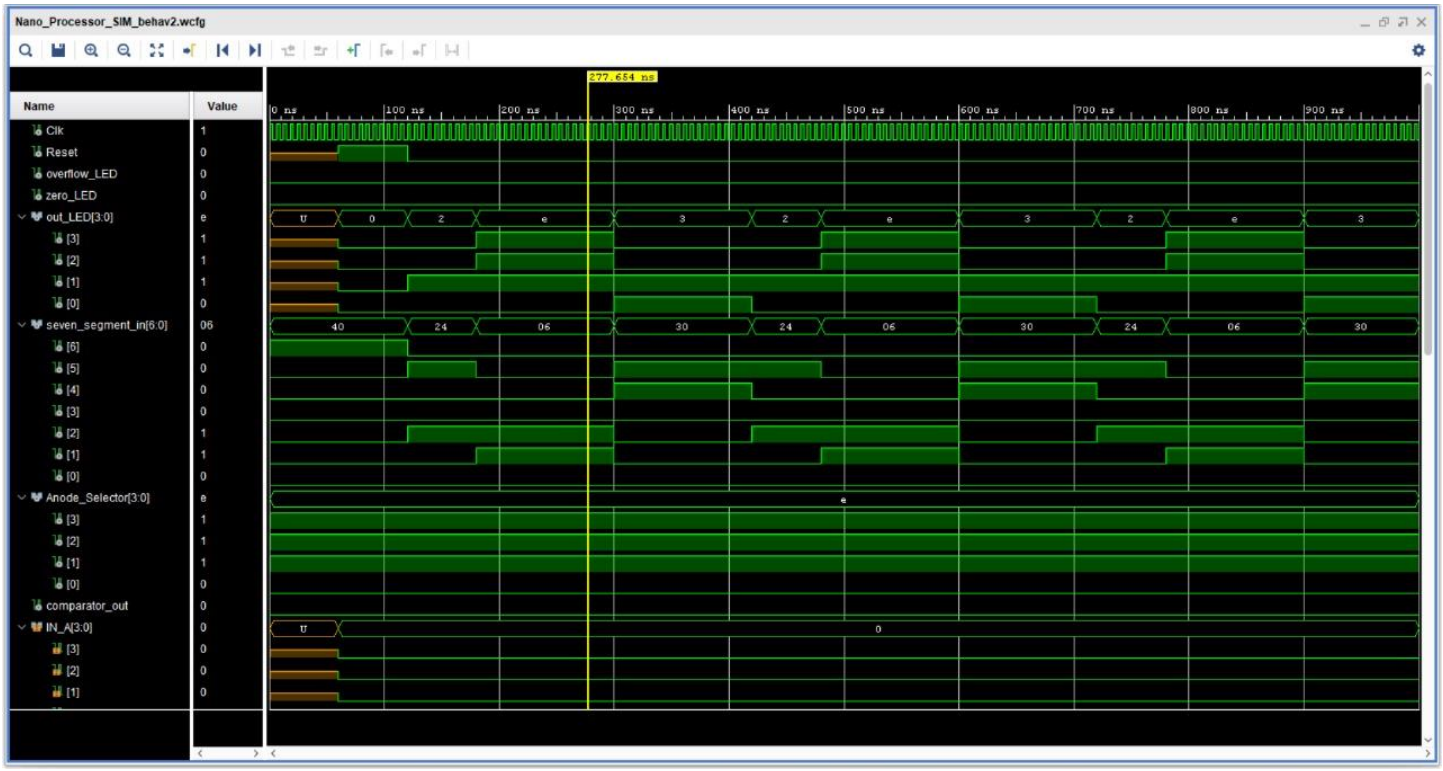
```
 wait for 60 ns;
 Reset <= '1' ;
 wait for 60 ns;
 Reset <= '0';
 wait;
 end process;
end Behavioral;
```

10.1.3 Timing diagram

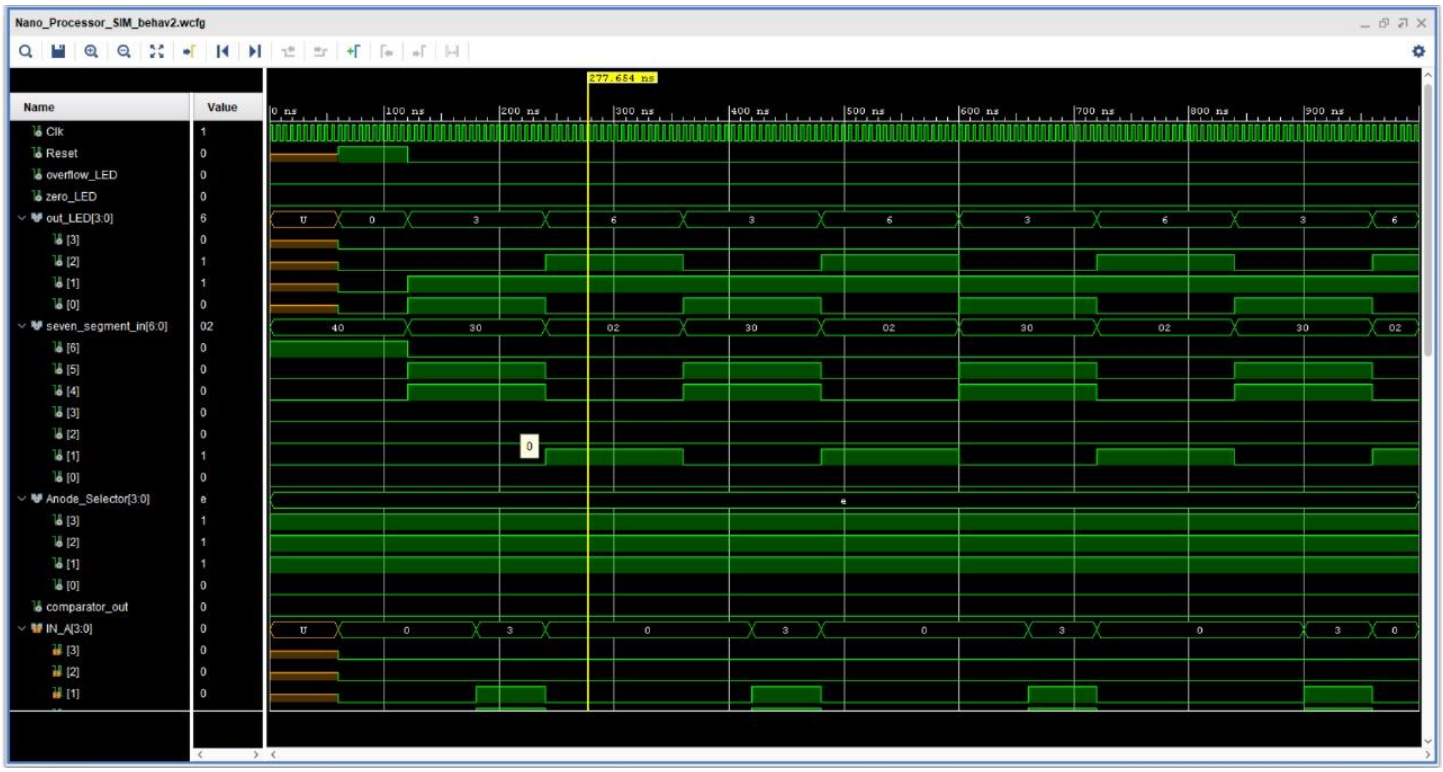
ROM-1



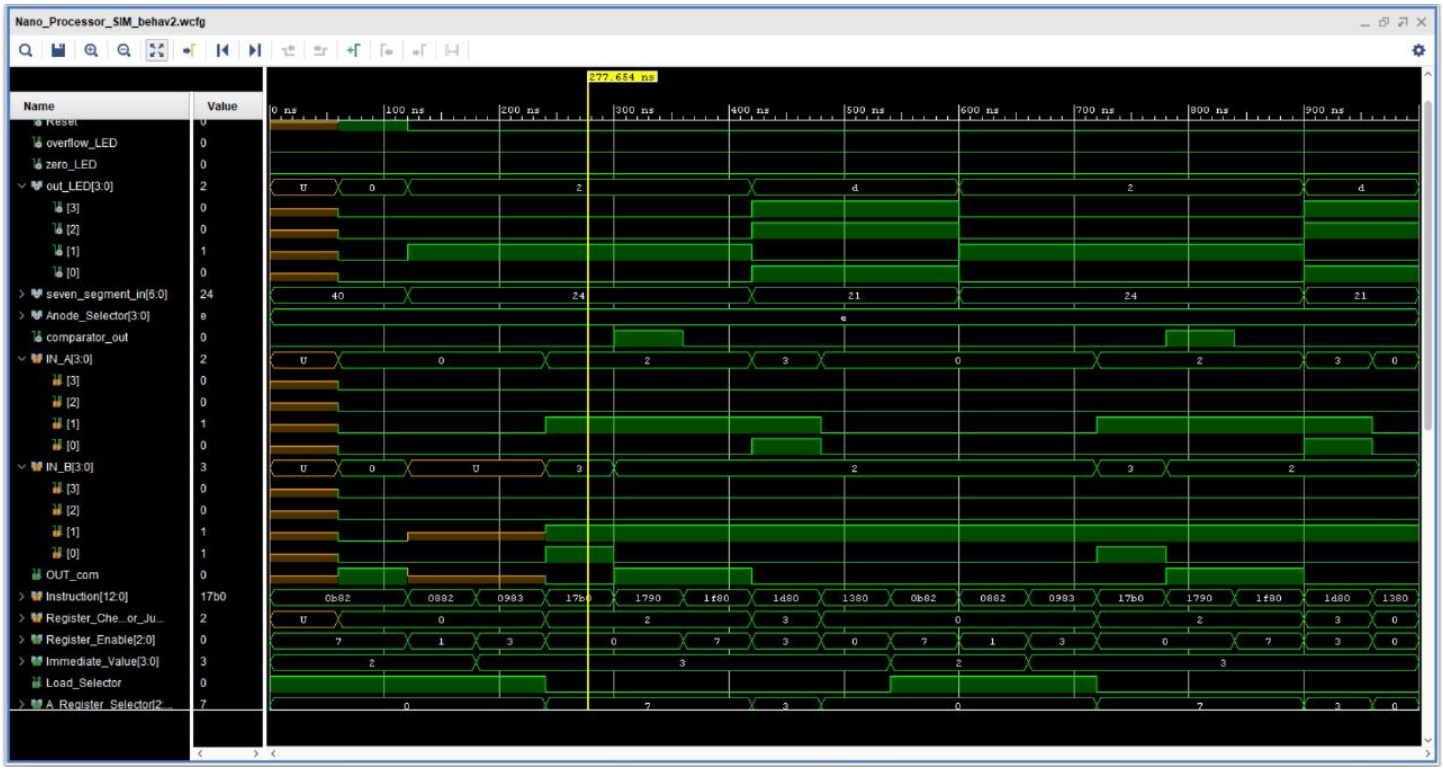
ROM-2


Nano_Processor_SIM_behav2.wcfg
277.654 ns
Name
Value
Clk
Reset
overflow_LED
zero_LED
out_LED[3:0]
seven_segment_in[6:0]
Anode_Selector[3:0]
comparator_out
IN_A[3:0]

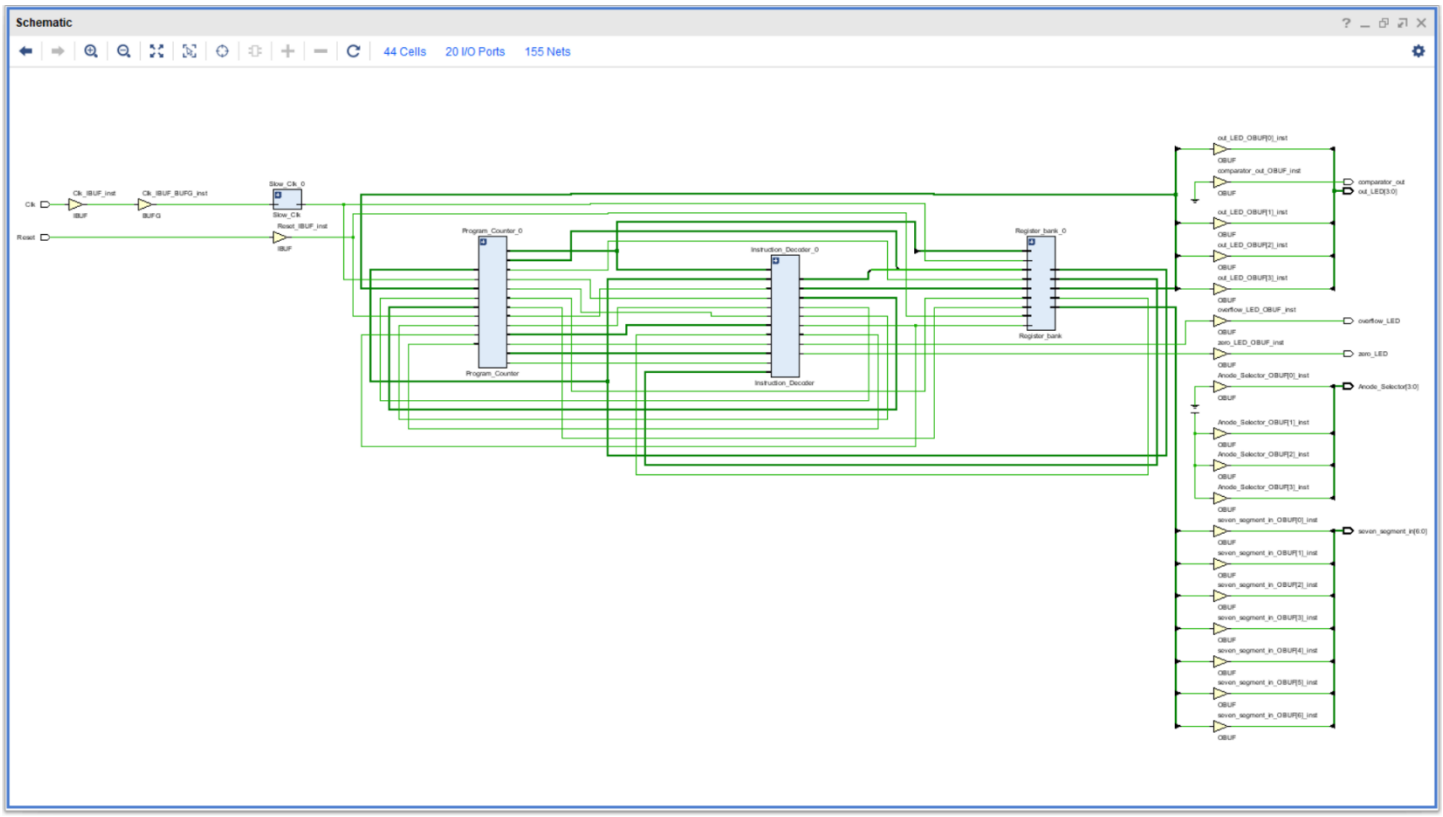
ROM-3



ROM-4



10.1.4 Implemented Design Schematic



10.2 Updated main module components

10.2.1 Instruction decoder

10.2.1.1 Design source file

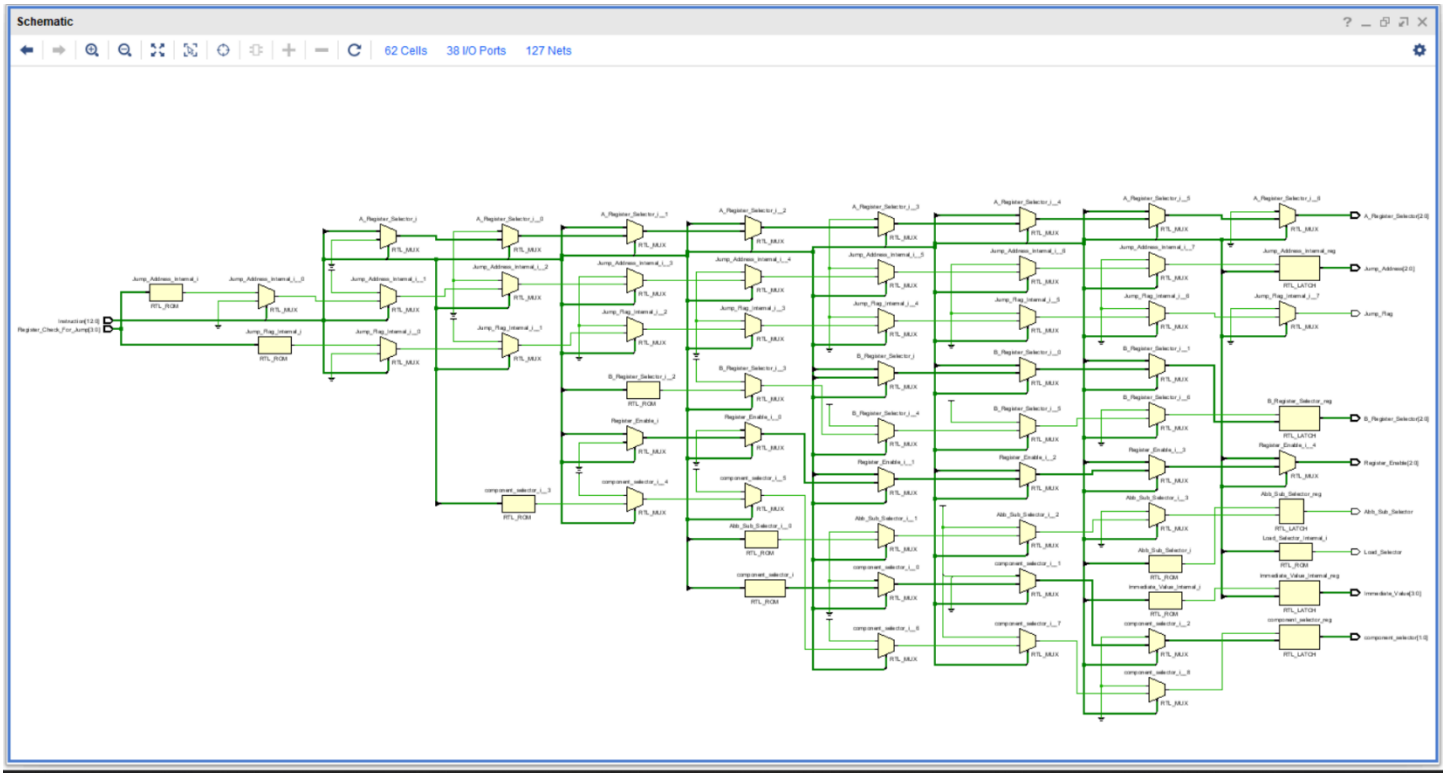
```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/08/2024 07:41:28 PM
-- Design Name:
-- Module Name: Instruction_Decoder - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- 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 Instruction_Decoder is
    Port ( Instruction : in STD_LOGIC_VECTOR (12 downto 0);
           Register_Check_For_Jump : in STD_LOGIC_VECTOR (3 downto 0);
           Register_Enable : out STD_LOGIC_VECTOR (2 downto 0);
           Immediate_Value : out STD_LOGIC_VECTOR (3 downto 0);
           Load_Selector : out STD_LOGIC;
           A_Register_Selector : out STD_LOGIC_VECTOR (2 downto 0);
           B_Register_Selector : out STD_LOGIC_VECTOR (2 downto 0);
           Abb_Sub_Selector : out STD_LOGIC;
           Jump_Flag : out STD_LOGIC;
           Jump_Address : out STD_LOGIC_VECTOR (2 downto 0);
           component_selector: out STD_LOGIC_VECTOR (1 downto 0)
           );

end Instruction_Decoder;

architecture Behavioral of Instruction_Decoder is
    signal Load_Selector_Internal : STD_LOGIC;
    signal Immediate_Value_Internal : STD_LOGIC_VECTOR (3 downto 0);
    signal Jump_Flag_Internal : STD_LOGIC;
    signal Jump_Address_Internal : STD_LOGIC_VECTOR (2 downto 0);
begin
    process (Instruction, Register_Check_For_Jump) is
    begin
    -- inicialize values of the outputs
      Jump_Flag_Internal <= '0';
      Register_Enable <= "000";
      A_Register_Selector <= "000";
      Load_Selector_Internal <= '0';
```

```
        -- Move
            if (Instruction(12 downto 10) ="010") then
                Immediate_Value_Internal <= Instruction(3 downto 0);
                Load_Selector_Internal <= '1';
                Register_Enable <= Instruction(9 downto 7);
                Jump_Flag_Internal <= '0';


            -- Add values
            elsif (Instruction(12 downto 10)="000") then
                component_selector<="00";
                A_Register_Selector <= Instruction(9 downto 7);
                B_Register_Selector <= Instruction(6 downto 4);
                Abb_Sub_Selector <= '0';
                Load_Selector_Internal <= '0';
                Jump_Flag_Internal <= '0';
                Register_Enable <= Instruction(9 downto 7);

             --multiply values
             elsif (Instruction(12 downto 10)="110") then
                 component_selector<="01";
                 A_Register_Selector <= Instruction(9 downto 7);
                 B_Register_Selector <= Instruction(6 downto 4);
                 Load_Selector_Internal <= '0';
                 Jump_Flag_Internal <= '0';
                 Register_Enable <= Instruction(9 downto 7);



            -- 2's complement
            elsif (Instruction(12 downto 10) ="001") then
                component_selector<="00";
                B_Register_Selector <= Instruction(9 downto 7);
                A_Register_Selector <= "000";
                Abb_Sub_Selector <= '1';
                Load_Selector_Internal <= '0';
                Jump_Flag_Internal <= '0';
                Register_Enable <= Instruction(9 downto 7);


           -- comparator
            elsif (Instruction(12 downto 10)="101") then
                component_selector<="10";
                B_Register_Selector <= Instruction(6 downto 4);
```

```
            A_Register_Selector <= Instruction(9 downto 7);


        --complement
        elsif (Instruction(12 downto 10)="111") then
            component_selector<="11";
            A_Register_Selector <= Instruction(9 downto 7);
            Load_Selector_Internal <= '0';
            Jump_Flag_Internal <= '0';
            Register_Enable <= Instruction(9 downto 7);


        -- JMP
        elsif (Instruction(12 downto 10)="100") then
                Jump_Flag_Internal <= '1';
                Jump_Address_Internal <= Instruction(2 downto 0);


        -- JNZ
        elsif (Instruction(12 downto 10)="011") then
            A_Register_Selector <= Instruction(9 downto 7);
            if(Register_Check_For_Jump = "0000") then
                Jump_Flag_Internal <= '1';
                Jump_Address_Internal <= Instruction(2 downto 0);
            else
                Jump_Flag_Internal <= '0';
            end if;

        end if;
    end process;

    -- Assign internal signals to output signals
    Immediate_Value <= Immediate_Value_Internal;
    Load_Selector <= Load_Selector_Internal;
    Jump_Flag <= Jump_Flag_Internal;
    Jump_Address <= Jump_Address_Internal;
end Behavioral;
```

10.2.1.2 Elaborated design schematic



10.2.1.3 Simulation source file

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/10/2024 10:08:11 PM
-- Design Name:
-- Module Name: TB_Instruction_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 TB_Instruction_Decoder is
--  Port ( );
end TB_Instruction_Decoder;

architecture Behavioral of TB_Instruction_Decoder is
component Instruction_Decoder is
    Port ( Instruction : in STD_LOGIC_VECTOR (12 downto 0);
           Register_Check_For_Jump : in STD_LOGIC_VECTOR (3 downto 0);
           Register_Enable : out STD_LOGIC_VECTOR (2 downto 0);
           Immediate_Value : out STD_LOGIC_VECTOR (3 downto 0);
           Load_Selector : out STD_LOGIC;
           A_Register_Selector : out STD_LOGIC_VECTOR (2 downto 0);
           B_Register_Selector : out STD_LOGIC_VECTOR (2 downto 0);
           Abb_Sub_Selector : out STD_LOGIC;
           Jump_Flag : out STD_LOGIC;
           Jump_Address : out STD_LOGIC_VECTOR (2 downto 0);
           component_selector: out STD_LOGIC_VECTOR (1 downto 0));
end component;

signal Instruction :  STD_LOGIC_VECTOR (12 downto 0);
signal Register_Check_For_Jump : STD_LOGIC_VECTOR (3 downto 0);
signal Register_Enable : STD_LOGIC_VECTOR (2 downto 0);
signal Immediate_Value :  STD_LOGIC_VECTOR (3 downto 0);
signal Load_Selector :  STD_LOGIC;
signal A_Register_Selector :  STD_LOGIC_VECTOR (2 downto 0);
signal B_Register_Selector :  STD_LOGIC_VECTOR (2 downto 0);
signal Abb_Sub_Selector :  STD_LOGIC;
signal Jump_Flag :  STD_LOGIC;
signal Jump_Address :  STD_LOGIC_VECTOR (2 downto 0);
signal component_selector:  STD_LOGIC_VECTOR (1 downto 0);
begin
UUT : Instruction_Decoder
port map(
```

```
        Instruction =>Instruction,
        Register_Check_For_Jump =>Register_Check_For_Jump ,
        Register_Enable=>Register_Enable ,
        Immediate_Value=>Immediate_Value ,
        Load_Selector => Load_Selector,
        A_Register_Selector=>A_Register_Selector ,
        B_Register_Selector=>B_Register_Selector ,
        Abb_Sub_Selector=>Abb_Sub_Selector ,
        Jump_Flag=>Jump_Flag ,
        Jump_Address=> Jump_Address,
        component_selector=>component_selector);
process begin
Instruction <= "0100110000111" ;--move value 7 to 3 rd register
Register_Check_For_Jump <="0000";
wait for 100ns;
Instruction <= "0000111000000" ;--add value in reg 3 and 4
Register_Check_For_Jump <="0000";
wait for 100ns;
Instruction <= "0011110000000" ;--two's complement of register  7
Register_Check_For_Jump <="0000";
wait for 100ns;
Instruction <= "0110010000111" ;--jump to 7 th instruction if reg 1 is 0000
Register_Check_For_Jump <="0000";
wait for 100ns;
Instruction <= "1101110010000" ;-- multiply 7 th register and  1 st register
 Register_Check_For_Jump <="0000";
wait for 100ns;
Instruction <= "1011110110000" ; --  compare reg 7 and reg 3
 Register_Check_For_Jump <="0000";
wait for 100ns;
Instruction <=  "1111110000000" ;-- complement of reg 7
 Register_Check_For_Jump <="0000";
wait for 100ns;
Instruction <= "1001110000000";-- jump to 000 instruction
 Register_Check_For_Jump <="0000";
wait for 100ns;

end process;


end Behavioral;
```

10.2.1.4Timing diagram

10.2.2 Program Rom

10.2.2.1 Design source file

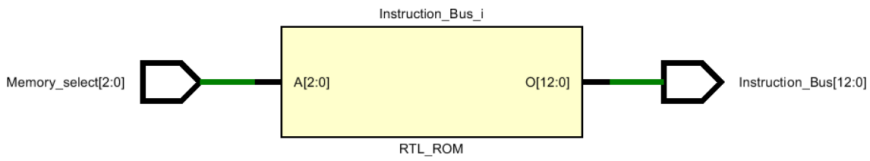
```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/07/2024 07:39:27 PM
-- Design Name:
-- Module Name: Program ROM - Behavioral
-- Project Name:
```

```
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.numeric_std.all;

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

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

entity Program_ROM is
    Port ( Memory_select : in STD_LOGIC_VECTOR (2 downto 0);
           Instruction_Bus : out STD_LOGIC_VECTOR (12 downto 0));
end Program_ROM;

architecture Behavioral of Program_ROM is

type rom_type is array (0 to 7) of std_logic_vector(12 downto 0);

    signal Program_ROM : rom_type := (

     --operations to add 1 to 3
                                "0101110000001", -- 0 move value 1 to seventh register
                                "0100010000010", -- 1 move value 2 to 1 st register
                                "0100100000011", -- 2 move value 3 to 2 nd register
                                "0001110010000", -- 3 add reg 1 and reg 7 and store value to
reg 7
                                "0001110100000", -- 4 add reg 2 and reg 7 and store value to
reg 7
                                "0101110000000", -- 5 jump 0th line of the program rom if 0
th register value = 0000
```

```
                                "0110000000000", -- 6
                                "0000000000000"  -- 7

--    -- operations to check NEG,JNZ,MOVI commands

--                                "0101110000010", -- 0  move value 2 to seventh register
--                                "0011110000000", -- 1  get the negative of the value in
seventh register
--                                "0110000000100", -- 2  jump 4th line of the program rom if
0 th register value = 0000
--                                "0101110001111", -- 3  move value 15 to seventh register
--                                "0101110000011", -- 4  move value 3 to seventh register
--                                "0110000000000", -- 5  jump 0th line of the program rom if
0 th register value = 0000
--                                "0110000000000", -- 6  jump 0th line of the program rom if
0 th register value = 0000
--                                "0110000000000"  -- 7  jump 0th line of the program rom if
0 th register value = 0000

--    -- operations for multiply
--                                   "0101110000011", -- 0 move value 1 to seventh register
--                                   "0100010000010", -- 1 move value 2 to 1 st register
--                                   "1101110010000", -- 2 multiply 7 th register and  1 st
register
--                                   "0110000000000", -- 3 jump 0th line of the program rom
if 0 th register value = 0000
--                                   "0001110100000", -- 4
--                                   "0110000000000", -- 5
--                                   "1100000000000", -- 6
--                                   "1100000000000"  -- 7

--    -- operations for complement and compare
--                                  "0101110000010", -- 0 move value 2 to 7 th register
--                                  "0100010000010", -- 1 move value 2 to 1 st register
--                                  "0100110000011", -- 2 move value 3 to 3 rd register
--                                  "1011110110000", -- 3 compare reg 7 and reg 3
--                                  "1011110010000", -- 4 compare reg 7 and reg 1
--                                  "1111110000000", -- 5 complement of reg 7
--                                  "1110110000000", -- 6 complement of reg 7
--                                  "1001110000000"  -- 7 jump to 000 instruction
);
begin



Instruction_Bus <= Program_ROM(to_integer(unsigned(Memory_select)));
```

```
end Behavioral;
```

10.2.2.1 Elaborated design schematic



11 Errors and Error handling

- The timing diagram of the final circuit was correct, but when we generated the bit stream and programmed the basys3 board we got an error due to an error in the instruction decoder.
  - ✓ We initialized all values of the outputs of the instruction decoder.

12 Conclusion

From this lab, we were able to design and develop a 4-bit arithmetic unit that can add and subtract signed integers. Each of us worked on implementing different components and combine them to design microprocessor but fitting them all together was tough.

After completing the basic structure of the processor, some additional features were added to optimize the functionalities. We also simulated each component individually before adding it to our processor. Furthermore, hierarchical design techniques and simple components were used wherever possible to design complex components. Using buses for handling inputs and outputs allowed us to greatly simplify our design instead of running so many wires around.

Beyond technical skills, this group project enriched our collaborative abilities, nurturing communication, coordination, and shared responsibility among team members. Each challenge surmounted, and lesson learned, contributed to our collective growth and solidified the importance of teamwork in tackling intricate engineering tasks.