CS1050 Computer Organization and Digital Design
Lab 09 & 10-Nano processor Design Competition

- Group 45
  - Suwasthiga Nagendramoorthy -220411H
  - Rahavi Srithar - 220614H
  - Parkkavi Sivakaran - 220618A
  - Arthikha Sooriyakumar - 220623J

Contents

        - Schematic diagram
        - Elaborated Diagram
        - Simulation source file
        - Timing diagram
    - Subtractor
        - Design source file
        - Schematic diagram
        - Elaborated Diagram
        - Simulation source file
        - Timing diagram
    - Multiplier
        - Design source file
        - Schematic diagram
        - Elaborated Diagram
        - Simulation source file
        - Timing diagram

- Encoders
    - 4 to 2 Encoder
        - Design source file
        - Schematic diagram
        - Elaborated Diagram
        - Simulation source file
        - Timing diagram

    - 8 to 3 Encoder
        - Design source file
        - Schematic diagram
        - Elaborated Diagram
        - Simulation source file
        - Timing diagram

    - 16 to 4 Encoder
        - Design source file
        - Schematic diagram
        - Elaborated Diagram
        - Simulation source file
        - Timing diagram

- Decoders
    - 3 to 8 Decoder
        - Design source file
        - Schematic diagram
        - Elaborated Diagram

- Simulation source file
- Timing diagram

- Multiplexers
  - 8 to 1 Multiplexers
    - Design source file
    - Schematic diagram
    - Elaborated Diagram
    - Simulation source file
    - Timing diagram

  - 8 way 4 bit Multiplexers
    - Design source file
    - Schematic diagram
    - Elaborated Diagram
    - Simulation source file
    - Timing diagram

  - 2 way 4 bit multiplexers
    - Design source file
    - Schematic diagram
    - Elaborated Diagram
    - Simulation source file
    - Timing diagram

  - 2 way 3 bit multiplexers
    - Design source file
    - Schematic diagram
    - Elaborated Diagram
    - Simulation source file
    - Timing diagram

- Counters
  - Design source file
  - Schematic diagram
  - Elaborated Diagram
  - Simulation source file
  - Timing diagram

- D Flip Flop
  - Design source file

- Schematic diagram
- Elaborated Diagram
- Simulation source file
- Timing diagram

❖ Slow Clock

- Design source file
- Schematic diagram
- Elaborated Diagram
- Simulation source file
- Timing diagram

❖ Conclusion

Introduction

- ❖ In this lab, our main goal was to create a 4-bit Nano Processor that could carry out four different instructions quickly and accurately. We worked together as a team to improve and build upon various essential components we had already made, like a 4-bit Add/Subtract unit, a 3-bit adder, a 3-bit Program Counter (PC), multiplexers, a Register Bank, a Program ROM, an Instruction Decoder, a 7-segment Display, and a slow-clock.
- ❖ We focused on making our design efficient and easy to understand. Instead of using lots of wires, we kept things simple by using 3, 4, and 12-bit buses to connect everything.
- ❖ To make sure our Nano Processor could understand and follow machine language, we turned instructions into binary code and stored them in ROM. We used a slow clock to watch our program execute step by step.
- ❖ As a team, we shared the workload among five members. We kept refining our designs, making changes based on new ideas and needs, until everything fit together seamlessly in the Nano Processor.
- ❖ After all our hard work, we successfully finished and tested our creation thoroughly using simulations and by actually running it on the Basys3 board, proving that it works as intended and showing our expertise in digital design.

Allocated Works For each Members in Group

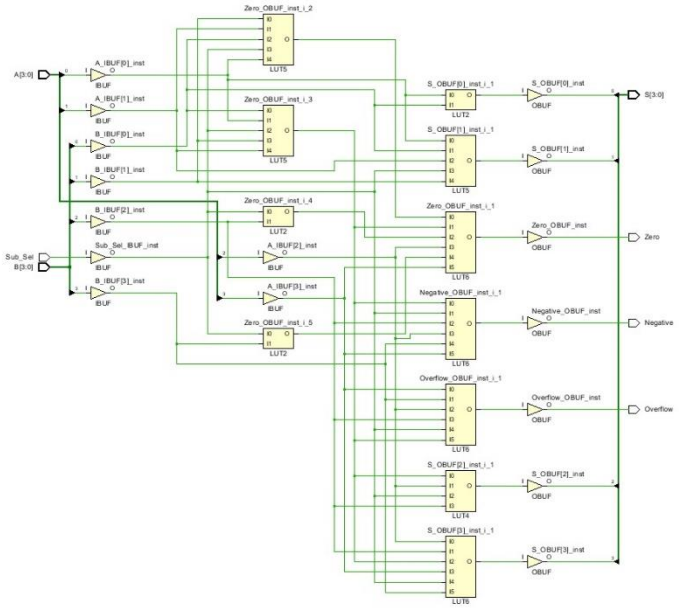
Name	Works	Time Taken
220618A	4-Bit Adder 4-Bit Subtractor 4-Bit Register	40Hours
220623J	Nano Nano with 7seg Reset controller Display 7seg	50Hours
220411H	3-Bit Program Counter Multiplexer 3 to 8 Decoder	40Hours
220614H	3-Bit Adder LUT 7-Segment Display	40Hours

A. 4- bit Adder, Subtractor

a. Design Source File

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

entity Add_Sub_4bit is
    Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
    B : in STD_LOGIC_VECTOR (3 downto 0);
    Sub_Sel : in STD_LOGIC;
    Zero:out std_logic;
    S : out STD_LOGIC_VECTOR (3 downto 0);
    Overflow : out STD_LOGIC;
    Negative:out std_logic);
end Add_Sub_4bit;

architecture Behavioral of Add_Sub_4bit is
component FA port(
    A:in std_logic;
    B:in std_logic;
    C_in:in std_logic;
    S:out std_logic;
    C_out:out std_logic );
end component;

signal FA0_S, FA0_C, FA1_S, FA1_C, FA2_S, FA2_C, FA3_S, FA3_C:std_logic;
signal B0, B1, B2, B3:std_logic;
signal S1 : std_LOGIC_VECTOR (3 downto 0);
signal Overflow0, Zero0:std_logic;

begin
    B0 <= B(0) xor Sub_Sel;
    B1 <= B(1) xor Sub_Sel;
    B2 <= B(2) xor Sub_Sel;
    B3 <= B(3) xor Sub_Sel;



    FA_0 : FA
        port map (
        A => A(0),
        B => B0,
        C_in => Sub_Sel,
        S => S1(0),
        C_Out => FA0_C);
    FA_1 : FA
        port map (
        A => A(1),
        B => B1,
        C_in => FA0_C,
        S => S1(1),
        C_Out => FA1_C);
    FA_2 : FA
        port map (
        A => A(2),
        B => B2,
        C_in => FA1_C,
        S => S1(2),
        C_Out => FA2_C);
    FA_3 : FA
        port map (
        A => A(3),
        B => B3,
        C_in => FA2_C,
        S => S1(3),
        C_Out => FA3_C);

    Overflow0 <= FA2_C xor FA3_C;
    Negative <= Overflow0 xor S1(3);
    Overflow <= Overflow0;
    Zero <= not(S1(0) or S1(1) or S1(2) or S1(3) or Overflow0) ;
    S <= S1;
end Behavioral;
```

b. Elaborated Diagram

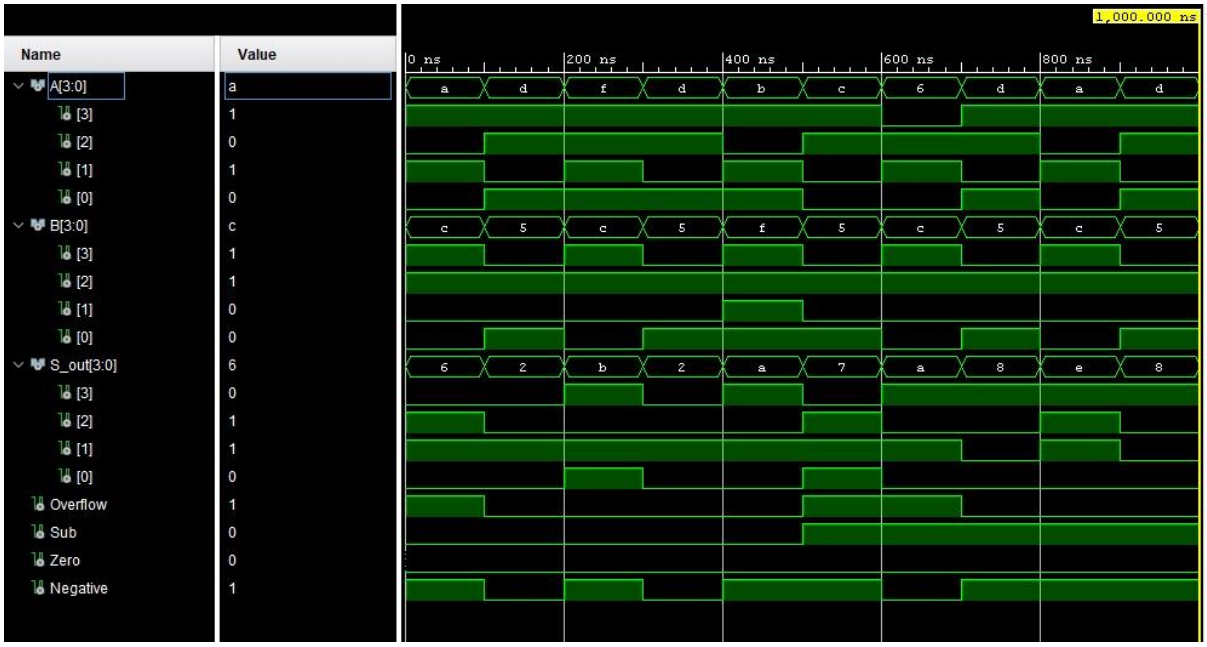


c. Simulation source file

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

entity sim_Add_Sub_4bit is
-- Port ( );
end sim_Add_Sub_4bit;
architecture Behavioral of sim_Add_Sub_4bit is

component Add_Sub_4bit
    Port (  A : in STD_LOGIC_VECTOR (3 downto 0);
            B : in STD_LOGIC_VECTOR (3 downto 0);
            Sub_Sel : in STD_LOGIC;
            Zero:out std_logic;
            S : out STD_LOGIC_VECTOR (3 downto 0);
            Overflow : out STD_LOGIC;
            Negative: out std_logic);
end component;

signal A,B,S_out: std_logic_vector (3 downto 0);
signal Overflow,Sub,Zero,Negative:std_logic;

begin
UUT: Add_Sub_4bit
    PORT MAP(
        A=>A(3 downto 0),
        B=>B(3 downto 0),
        Sub_Sel=>Sub,
        zero=>Zero,
        S=>S_out(3 downto 0),
        Overflow=>Overflow,
        Negative=>Negative
    );

process
begin
--220618A  11 0101 1101 1100 1010
--220623J  11 0101 1101 1100 1111
--220411H  11 0101 1100 1111 1011
--220614H  11 0101 1101 1100 0110
```

```
        Sub<='0';
        A<="1010";
        B<="1100";
        wait for 100ns;
        A<="1101";
        B<="0101";
        wait for 100ns;
        A<="1111";
        B<="1100";
        wait for 100ns;
        A<="1101";
        B<="0101";
        wait for 100ns;
        A<="1011";
        B<="1111";
        wait for 100ns;
        Sub<='1';
        A<="1100";
        B<="0101";
        wait for 100ns;
        A<="0110";
        B<="1100";
        wait for 100ns;
        A<="1101";
        B<="0101";
        wait for 100ns;
        A<="1010";
        B<="1100";
        wait for 100ns;
        A<="1101";
        B<="0101";
        wait for 100ns;
    end process;
    end Behavioral;
```

d. Timing diagram

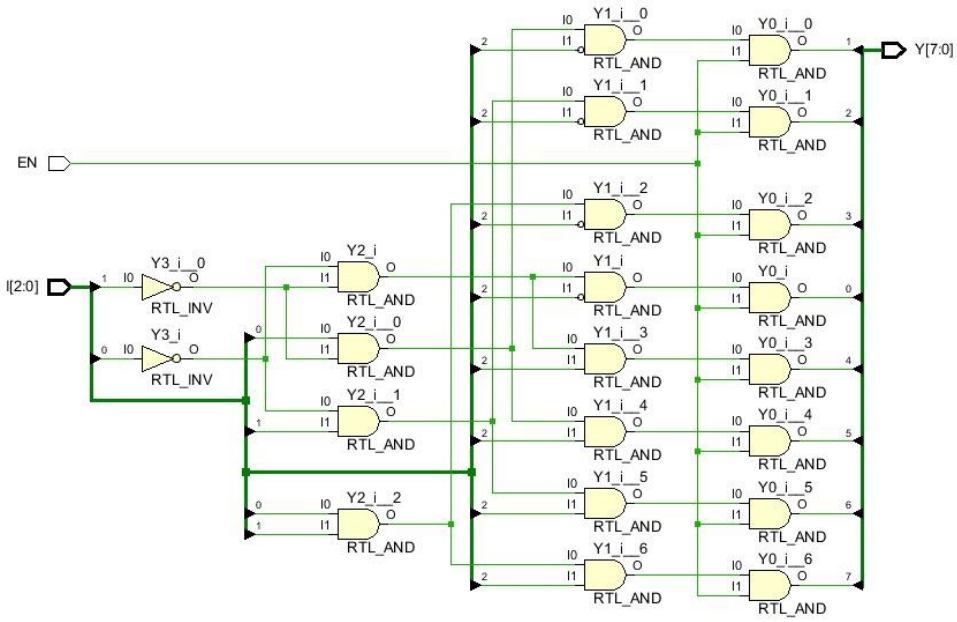


B. Decoder 3 to 8

a. Design Source File

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

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

architecture Behavioral of Decoder_3_to_8 is

begin
    Y(0) <= (NOT I(0)) AND (NOT I(1)) AND (NOT I(2)) AND EN;
    Y(1) <= I(0) AND (NOT I(1)) AND (NOT I(2)) AND EN;
    Y(2) <= (NOT I(0)) AND I(1) AND (NOT I(2)) AND EN;
    Y(3) <= I(0) AND I(1) AND (NOT I(2))  AND EN;
    Y(4) <= (NOT I(0)) AND (NOT I(1)) AND I(2) AND EN;
    Y(5) <= I(0) AND (NOT I(1)) AND I(2)  AND EN;
    Y(6) <= (NOT I(0)) AND I(1) AND I(2) AND EN;
    Y(7) <= I(0) AND I(1) AND I(2)  AND EN;

end Behavioral;
```

b. Elaborated Diagram



c. Simulation source file

```
library IEEE;
use IEEE.STD_LOGIC_1164.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 (2 DOWNTO 0);
       EN : IN STD_LOGIC;
       Y : OUT STD_LOGIC_VECTOR (7 DOWNTO 0));
end component;
SIGNAL I : STD_LOGIC_VECTOR (2 DOWNTO 0);
SIGNAL EN : STD_LOGIC;
SIGNAL Y : STD_LOGIC_VECTOR (7 DOWNTO 0);
begin
UUT : Decoder_3_to_8
PORT MAP(
    I => I,
    EN => EN,
    Y => Y
    );
process
begin
--220618A  11 0101 1101 1100 1010
--220623J  11 0101 1101 1100 1111
--220411H  11 0101 1100 1111 1011
--220614H  11 0101 1101 1100 0110
    EN <= '1';
    I <= "010";
    WAIT FOR 100ns;
    I <= "001";
    WAIT FOR 100ns;
    I <= "111";
    WAIT FOR 100ns;
    I <= "001";
    WAIT FOR 100ns;
    I <= "011";
    WAIT FOR 100ns;
    I <= "110";
    WAIT FOR 100ns;
    I <= "110";
    WAIT FOR 100ns;
    I <= "000";
    WAIT FOR 100ns;
    EN <= '0';
    WAIT FOR 100ns;
    I <= "111";
    WAIT FOR 100ns;
end process;
end behavioral;
```

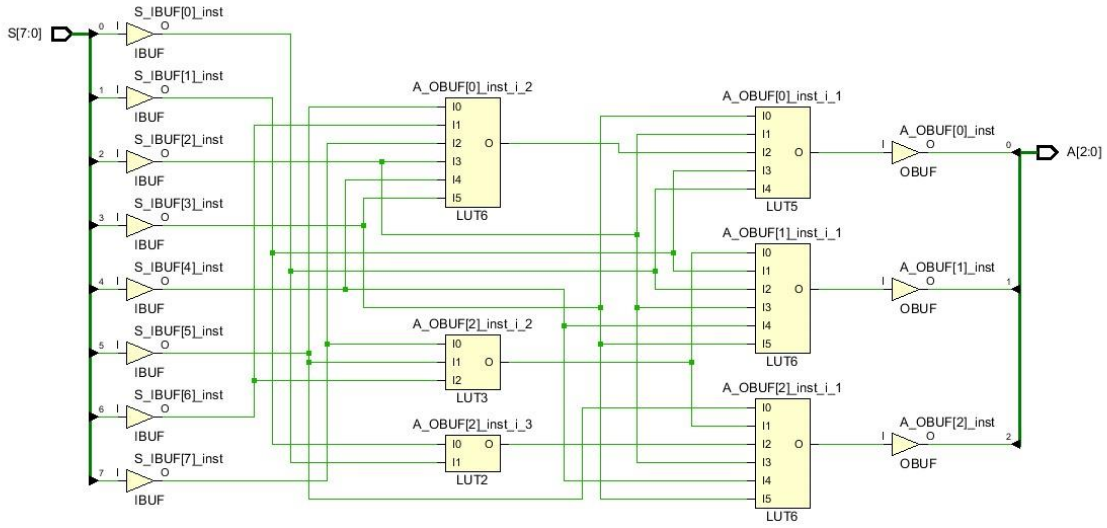
d. Timing diagram

C. Encoder 8 to 3

a. Design Source File

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

entity encoder_8_to_3 is
    Port ( S : in STD_LOGIC_VECTOR (7 downto 0);
           A : out STD_LOGIC_VECTOR(2 downto 0));
end encoder_8_to_3;

architecture Behavioral of encoder_8_to_3 is
    COMPONENT encoder_4_to_2
        Port ( S : in STD_LOGIC_VECTOR (3 downto 0);
               A : out STD_LOGIC_VECTOR(1 downto 0)
              );
    END COMPONENT;

    signal I:std_logic_vector(1 downto 0);

    begin
        A<="000" when S(0)='1' else
        "001" when S(1)='1' else
        "010" when S(2)='1' else
        "011" when S(3)='1' else
        "100" when S(4)='1' else
        "101" when S(5)='1' else
        "110" when S(6)='1' else
        "111" when S(7)='1' else
        "XXX";
  end Behavioral;
```

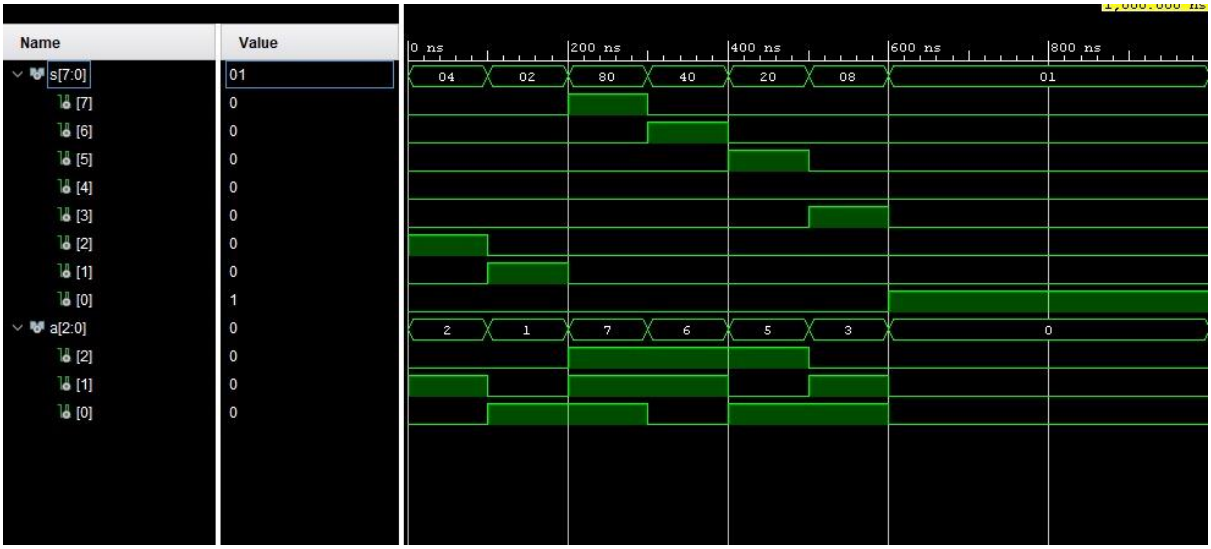
b. Elaborated Diagram



c. Simulation source file

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

entity encoder_8_to_3_Sim is

end encoder_8_to_3_Sim;

architecture Behavioral of encoder_8_to_3_Sim is
    COMPONENT encoder_8_to_3
        port(S:in std_logic_vector(7 downto 0);
             A:out std_logic_vector(2 downto 0));
     END COMPONENT;
     SIGNAL s:std_logic_vector(7 downto 0);
     signal a:std_logic_vector(2 downto 0);
    --220618A  110 101 110 111 001 010
    --220623J  110 101 110 111 001 111
    --220411H  110 101 110 011 111 011
    --220614H  110 101 110 111 000 110
    begin
    UUT :encoder_8_to_3 PORT MAP(
        S=>s,
        A=>a);

     process
     begin
         s<="00000100";
         wait for 100 ns;
         s<="00000010";
         wait for 100ns;
         s<="10000000";
         wait for 100 ns;
         s<="01000000";
         wait for 100ns;
         s<="00100000";
          wait for 100 ns;
          s<="00001000";
          wait for 100ns;
          s<="00000001";
          wait for 100 ns;
         wait;
     end process;

end Behavioral;
```

d. Timing diagram.

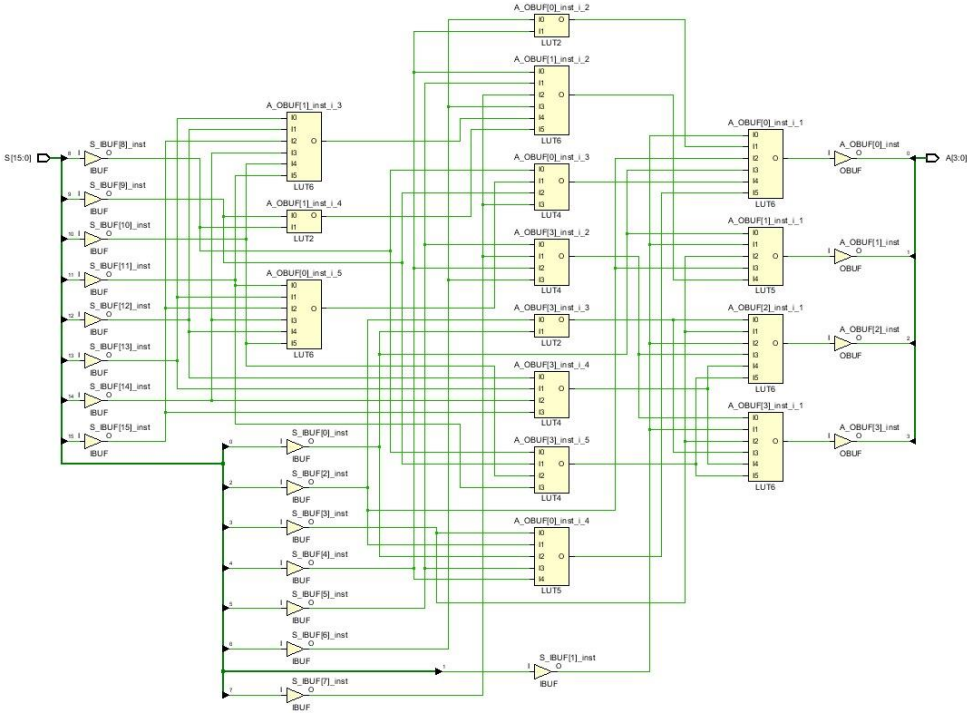


D. Encoder 16 to 4

a. Design Source File.

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

entity encoder_16_to_4 is
    Port ( S : in STD_LOGIC_VECTOR (15 downto 0);
           A : out STD_LOGIC_VECTOR(3 downto 0));
end encoder_16_to_4;

architecture Behavioral of encoder_16_to_4 is

begin
    A<="0000" when S(0)='1' else
    "0001" when S(1)='1' else
    "0010" when S(2)='1' else
    "0011" when S(3)='1' else
    "0100" when S(4)='1' else
    "0101" when S(5)='1' else
    "0110" when S(6)='1' else
    "0111" when S(7)='1' else
    "1000" when S(8)='1' else
    "1001" when S(9)='1' else
    "1010" when S(10)='1' else
    "1011" when S(11)='1' else
    "1100" when S(12)='1' else
    "1101" when S(13)='1' else
    "1110" when S(14)='1' else
    "1111" when S(15)='1' else
    "XXXX";

end Behavioral;
```

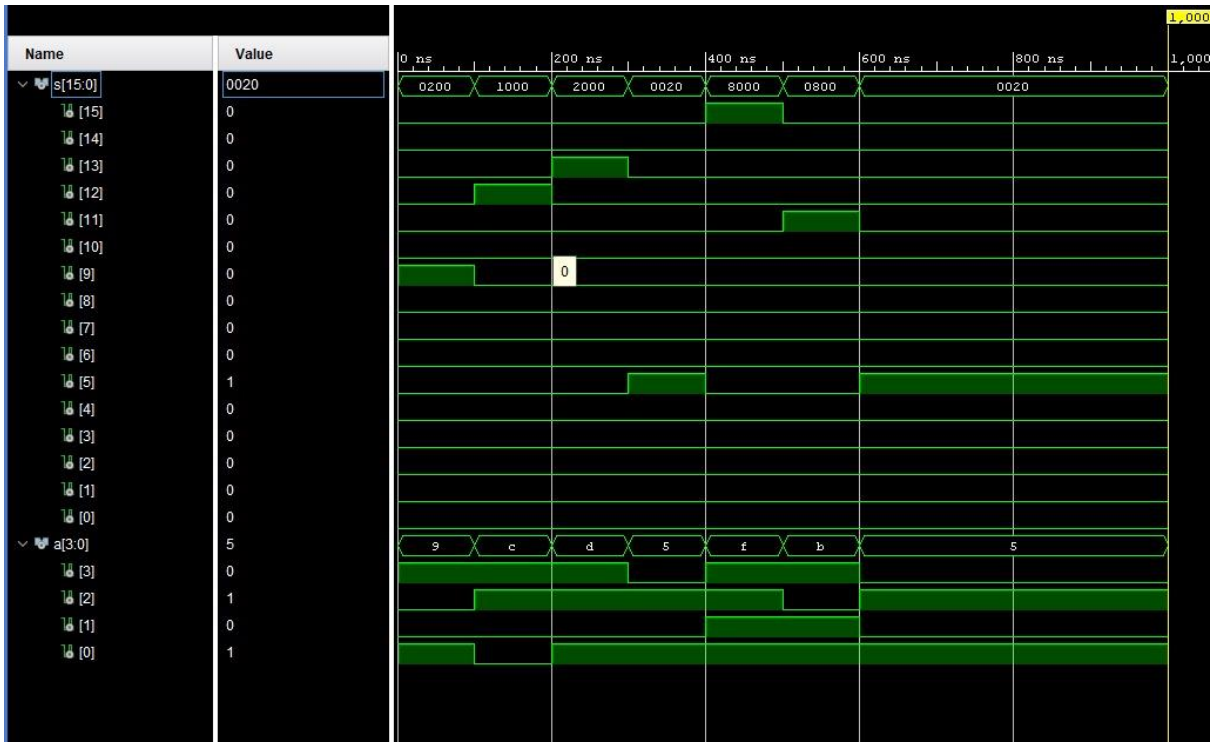
b. Elaborated Diagram



c. Simulation source file

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

entity encoder_16_to_4_sim is

end encoder_16_to_4_sim;

architecture Behavioral of encoder_16_to_4_sim is
    COMPONENT encoder_16_to_4
        port(S:in std_logic_vector(15 downto 0);
            A:out std_logic_vector(3 downto 0));
     END COMPONENT;
     SIGNAL s:std_logic_vector(15 downto 0);
     signal a:std_logic_vector(3 downto 0);
     --220618A  11 0101 1101 1100 1010
     --220623J  11 0101 1101 1100 1111
     --220411H  11 0101 1100 1111 1011
     --220614H  11 0101 1101 1100 0110

    begin
        UUT :encoder_16_to_4 PORT MAP(
            S=>s,
            A=>a);
         process
         begin
         s<="0000001000000000";
         wait for 100 ns;
         s<="0001000000000000";
         wait for 100ns;
         s<="0010000000000000";
         wait for 100 ns;
         s<="0000000000100000";
         wait for 100ns;
         s<="1000000000000000";
         wait for 100 ns;
         s<="0000100000000000";
         wait for 100ns;
         s<="0000000000100000";
         wait;
     end process;

end Behavioral;
```

d. Timing diagram

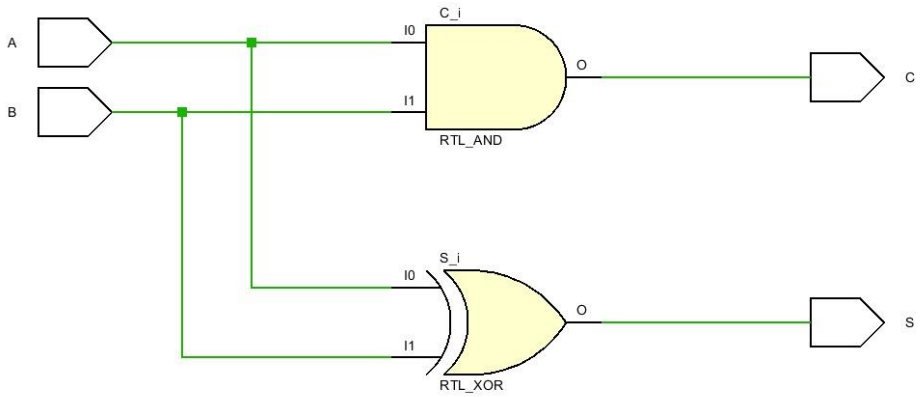


E. HA

a. Design Source File

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

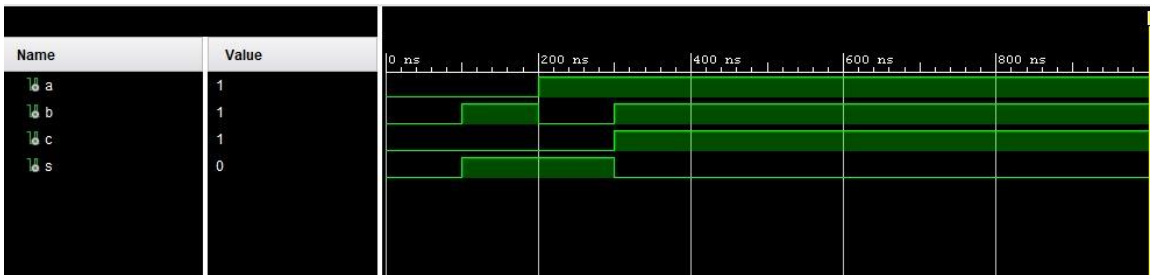
b. Elaborated Diagram



c. Simulation source file

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

entity HA_Sim is
--  Port ( );
end HA_Sim;

architecture Behavioral of HA_Sim is
COMPONENT HA
    port(A,B:in std_logic;
        S,C:out std_logic);
END COMPONENT;
signal a,b,c,s :std_logic;
begin
UUT: HA PORT MAP(
    A=>a,
    B=>b,
    C=>c,
    S=>s);
 process
 begin
    a<='0';
    b<='0';
    wait for 100ns;
    b<='1';
    wait for 100ns;
    a<='1';
    b<='0';
    wait for 100ns;
    b<='1';
    wait;
 end process;

end Behavioral;
```

d. Timing diagram.

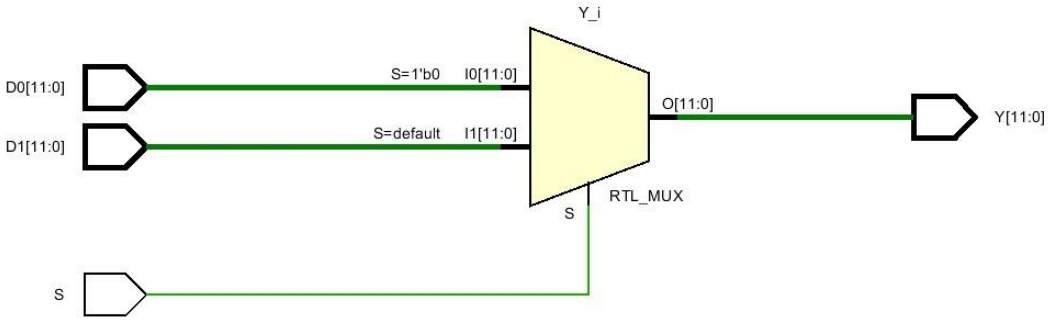


F. 12bit Multiplexer 2 to 1

a. 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 MUX_2_to_1_12 is
    Port (  D0 : in STD_LOGIC_VECTOR(11 downto 0);
            D1 : in STD_LOGIC_VECTOR(11 downto 0);
            S : in STD_LOGIC;
            Y : out STD_LOGIC_VECTOR(11 downto 0));
end MUX_2_to_1_12;

architecture Behavioral of MUX_2_to_1_12 is

begin
    Y <= D0 when (S = '0') else D1;
end Behavioral;
```

b. Elaborated Diagram

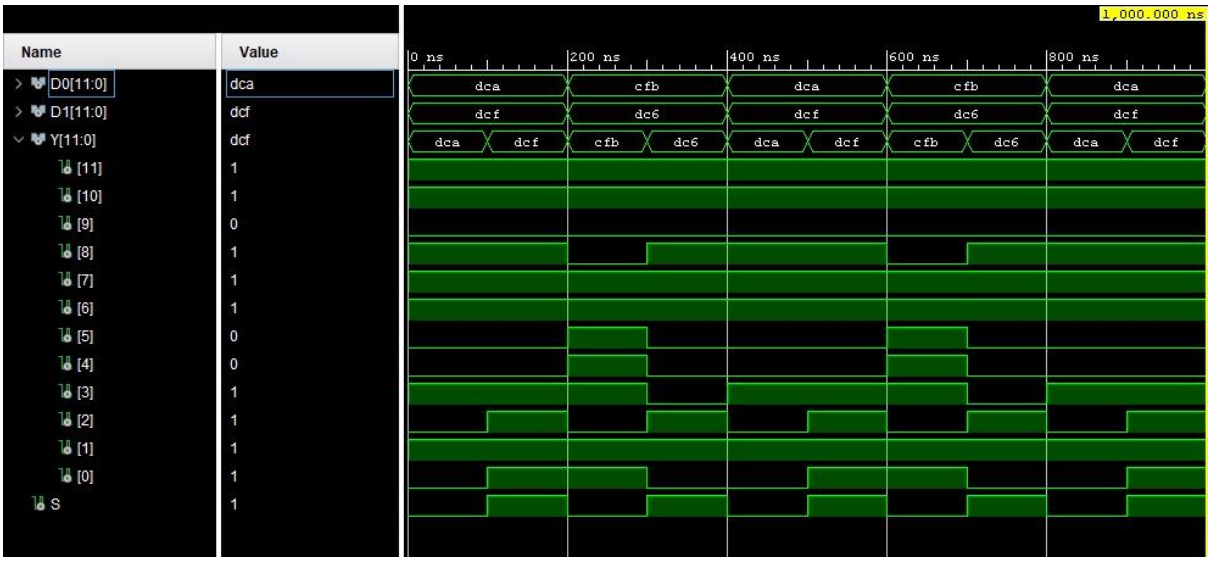


c. Simulation source file

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

entity TB_MUX_2_to_1_12 is
-- Port ( );
end TB_MUX_2_to_1_12;
architecture Behavioral of TB_MUX_2_to_1_12 is

component MUX_2_to_1_12
    Port (  D0 : in STD_LOGIC_VECTOR(11 downto 0);
            D1 : in STD_LOGIC_VECTOR(11 downto 0);
            S : in STD_LOGIC;
            Y : out STD_LOGIC_VECTOR(11 downto 0));
end component;

signal D0,D1,Y : std_logic_vector(11 downto 0);
signal S : std_logic;

begin
UUT: MUX_2_to_1_12
    port map(
        D0 => D0,
        D1 => D1,
        S => S,
        Y => Y
    );
```

```
process
begin
--220618A  11 0101 1101 1100 1010
--220623J  11 0101 1101 1100 1111
--220411H  11 0101 1100 1111 1011
--220614H  11 0101 1101 1100 0110
    D0 <= "110111001010";
    D1 <= "110111001111";
    S <= '0';
    wait for 100 ns;
    S <= '1';
    wait for 100 ns;
    D0 <= "110011111011";
    D1 <= "110111000110";
    S <= '0';
    wait for 100 ns;
    S <= '1';
    wait for 100 ns;
    D0 <= "110111001010";
    D1 <= "110111001111";
    S <= '0';
    wait for 100 ns;
    S <= '1';
    wait for 100 ns;
    D0 <= "110011111011";
    D1 <= "110111000110";
    S <= '0';
    wait for 100 ns;
    S <= '1';
    wait for 100 ns;
    D0 <= "110111001010";
    D1 <= "110111001111";
    S <= '0';
    wait for 100 ns;
    S <= '1';
    wait;
end process;
end Behavioral;
```

d. Timing diagram

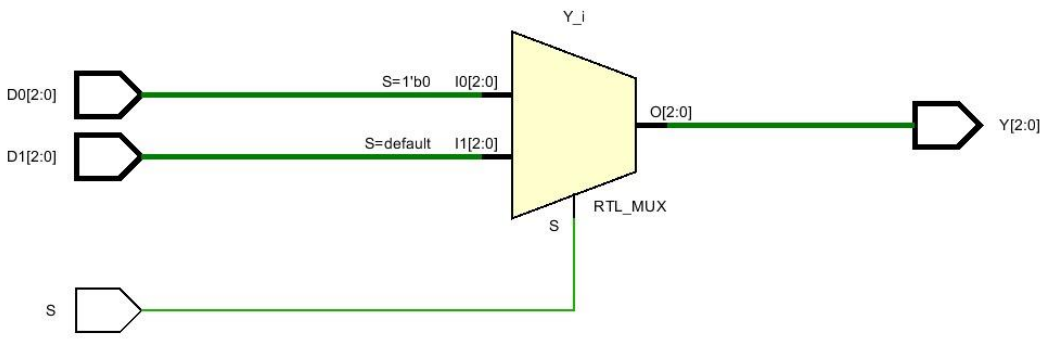


G. 3bit Multiplexer 2 to 1

a. Design Source File

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

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity MUX_2_to_1_3 is
    Port ( D0 : in STD_LOGIC_VECTOR (2 downto 0);
    D1 : in STD_LOGIC_VECTOR (2 downto 0);
    S : in STD_LOGIC;
    Y : out STD_LOGIC_VECTOR (2 downto 0));
end MUX_2_to_1_3;

architecture Behavioral of MUX_2_to_1_3 is
begin
    Y <= D0 when (S = '0') else D1;
end Behavioral;
```

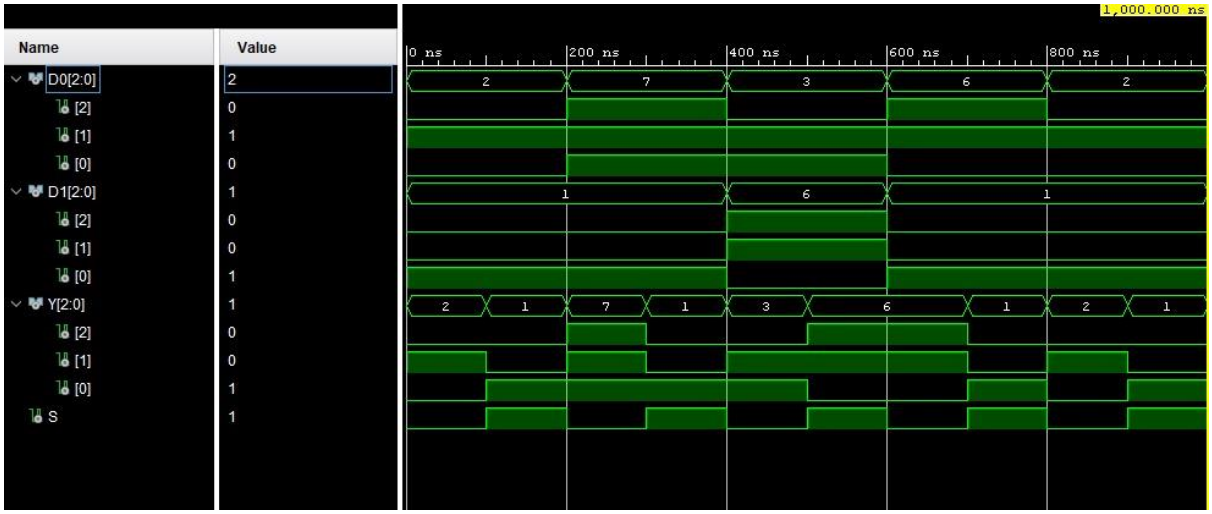
b. Elaborated Diagram



c. Simulation source file

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

entity TB_MUX_2_to_1_3 is
-- Port ( );
end TB_MUX_2_to_1_3;
architecture Behavioral of TB_MUX_2_to_1_3 is

component MUX_2_to_1_3
    Port (  D0 : in STD_LOGIC_VECTOR(2 downto 0);
            D1 : in STD_LOGIC_VECTOR(2 downto 0);
            S : in STD_LOGIC;
            Y : out STD_LOGIC_VECTOR(2 downto 0));
end component;

signal D0,D1,Y : std_logic_vector(2 downto 0);
signal S : std_logic;

begin
UUT: MUX_2_to_1_3
    port map(
        D0 => D0,
        D1 => D1,
        S => S,
        Y => Y
    );

process
begin
--220618A  11 0101 1101 1100 1010
--220623J  11 0101 1101 1100 1111
--220411H  11 0101 1100 1111 1011
--220614H  11 0101 1101 1100 0110
    D0 <= "010";
    D1 <= "001";
    S <= '0';
    wait for 100 ns;
    S <= '1';
    wait for 100 ns;
    D0 <= "111";
    D1 <= "001";
    S <= '0';
    wait for 100 ns;
    S <= '1';
    wait for 100 ns;
    D0 <= "011";
    D1 <= "110";
    S <= '0';
    wait for 100 ns;
    S <= '1';
    wait for 100 ns;
    D0 <= "110";
    D1 <= "001";
    S <= '0';
    wait for 100 ns;
    S <= '1';
    wait for 100 ns;
    D0 <= "010";
    D1 <= "001";
    S <= '0';
    wait for 100 ns;
    S <= '1';
    wait;
end process;
end Behavioral;
```

d. Timing diagram

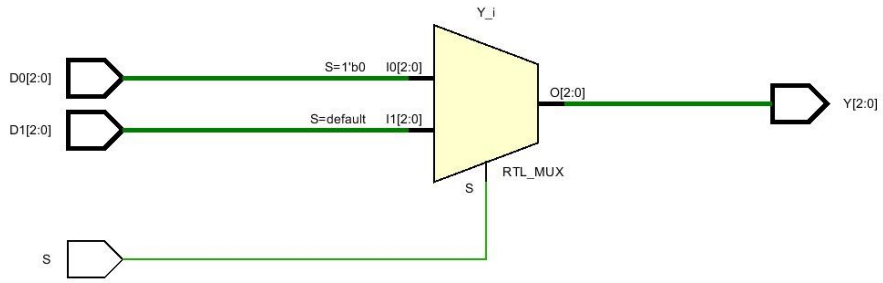


H. 4bit Multiplexer 2 to 1

a. Design Source File

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

entity MUX_2_to_1_4 is
    Port ( D0 : in STD_LOGIC_VECTOR(3 downto 0);
    D1 : in STD_LOGIC_VECTOR(3 downto 0);
    S : in STD_LOGIC;
    Y : out STD_LOGIC_VECTOR(3 downto 0));
end MUX_2_to_1_4;

architecture Behavioral of MUX_2_to_1_4 is
begin
    Y <= D0 when (S = '0') else D1;
end Behavioral;
```

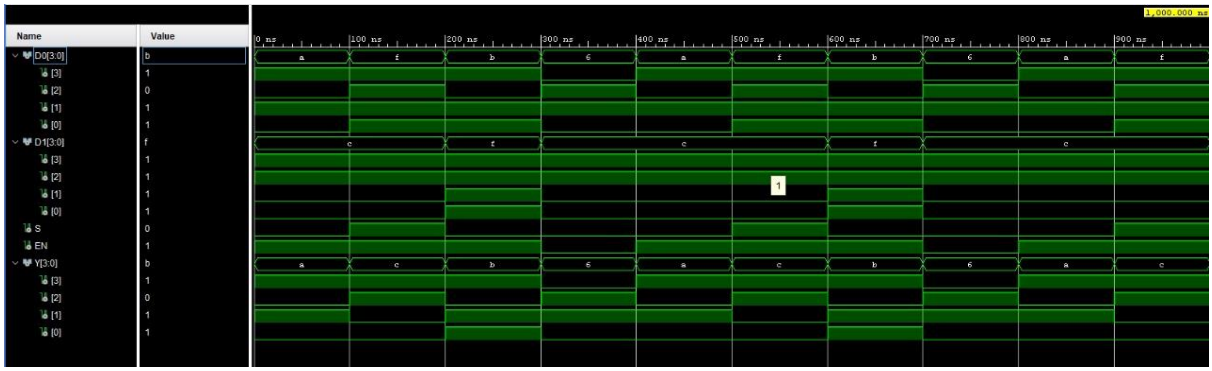
b. Elaborated Diagram



c. Simulation source file

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

entity Mux_2_to_1_4bit_Sim is
--  Port ( );
end Mux_2_to_1_4bit_Sim;

architecture Behavioral of Mux_2_to_1_4bit_Sim is

component MUX_2_TO_1_4bit
    Port (
                S : in STD_LOGIC;
                Y : out STD_LOGIC_VECTOR (3 downto 0);
                D0 : in STD_LOGIC_VECTOR (3 downto 0);
                D1 : in STD_LOGIC_VECTOR (3 downto 0));
end component;
SIGNAL D0,D1 : STD_LOGIC_VECTOR (3 downto 0);
SIGNAL S : STD_LOGIC;
SIGNAL EN : STD_LOGIC;
SIGNAL Y : STD_LOGIC_VECTOR (3 downto 0);
begin
UUT : MUX_2_TO_1_4bit
port map (
    D0 => D0,
    D1 => D1,
    S => S,
    Y => Y
    );
process
begin
--220618A  11 0101 1101 1100 1010
--220623J  11 0101 1101 1100 1111
--220411H  11 0101 1100 1111 1011
--220614H  11 0101 1101 1100 0110
EN <= '1';
D0 <= "1010";
D1 <= "1100";
S <= '0';
WAIT FOR 100ns;
D0 <= "1111";
D1 <= "1100";
S <= '1';
WAIT FOR 100ns;
D0 <= "1011";
D1 <= "1111";
S <= '0';
WAIT FOR 100ns;
EN <= '0';
D0 <= "0110";
D1 <= "1100";
WAIT FOR 100ns;
end process;
end Behavioral;
```

d. Timing diagram



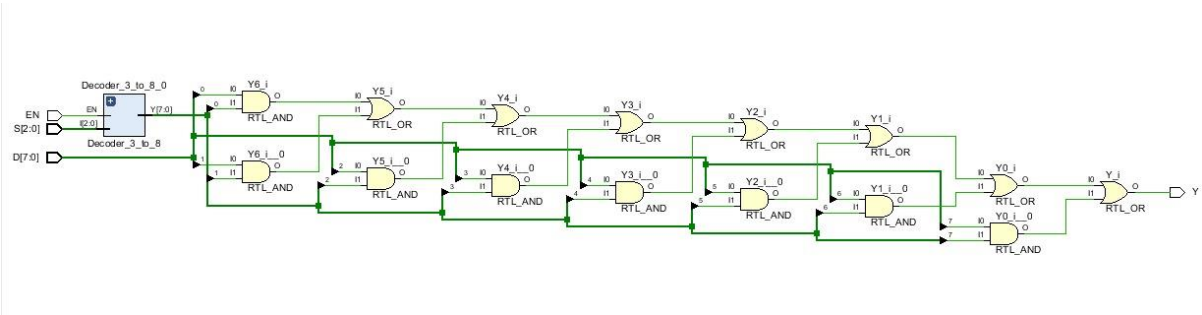
I. Multiplexer 8 to 1

a. Design Source File

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

entity MUX_8_to_1_4 is
    Port ( R0 : in STD_LOGIC_VECTOR (3 downto 0);
    R1 : in STD_LOGIC_VECTOR (3 downto 0);
    R2 : in STD_LOGIC_VECTOR (3 downto 0);
    R3 : in STD_LOGIC_VECTOR (3 downto 0);
    R4 : in STD_LOGIC_VECTOR (3 downto 0);
    R5 : in STD_LOGIC_VECTOR (3 downto 0);
    R6 : in STD_LOGIC_VECTOR (3 downto 0);
    R7 : in STD_LOGIC_VECTOR (3 downto 0);
    S : in STD_LOGIC_VECTOR (2 downto 0);
    --EN : in STD_LOGIC;
    Y : out STD_LOGIC_VECTOR (3 downto 0));
end MUX_8_to_1_4;

architecture Behavioral of MUX_8_to_1_4 is
component MUX_8_to_1
    Port ( D : in STD_LOGIC_VECTOR (7 downto 0);
    S : in STD_LOGIC_VECTOR (2 downto 0);
    EN : in STD_LOGIC;
    Y : out STD_LOGIC);
end component;
signal Enable : STD_LOGIC := '1';

begin
    MUX_0 : MUX_8_to_1
        Port map(
            D(0) => R0(0),
            D(1) => R1(0),
            D(2) => R2(0),
            D(3) => R3(0),
            D(4) => R4(0),
            D(5) => R5(0),
            D(6) => R6(0),
            D(7) => R7(0),
            EN => Enable,
            S => S,
            Y => Y(0)
            );
```

```
            );
    MUX_1 : MUX_8_to_1
        Port map(
            D(0) => R0(1),
            D(1) => R1(1),
            D(2) => R2(1),
            D(3) => R3(1),
            D(4) => R4(1),
            D(5) => R5(1),
            D(6) => R6(1),
            D(7) => R7(1),
            EN => Enable,
            S => S,
            Y => Y(1)
            );
    MUX_2 : MUX_8_to_1
        Port map(
            D(0) => R0(2),
            D(1) => R1(2),
            D(2) => R2(2),
            D(3) => R3(2),
            D(4) => R4(2),
            D(5) => R5(2),
            D(6) => R6(2),
            D(7) => R7(2),
            EN => Enable,
            S => S,
            Y => Y(2)
            );
    MUX_3 : MUX_8_to_1
        Port map(
            D(0) => R0(3),
            D(1) => R1(3),
            D(2) => R2(3),
            D(3) => R3(3),
            D(4) => R4(3),
            D(5) => R5(3),
            D(6) => R6(3),
            D(7) => R7(3),
            EN => Enable,
            S => S,
            Y => Y(3)
            );
end Behavioral;
```

b. Elaborated Diagram

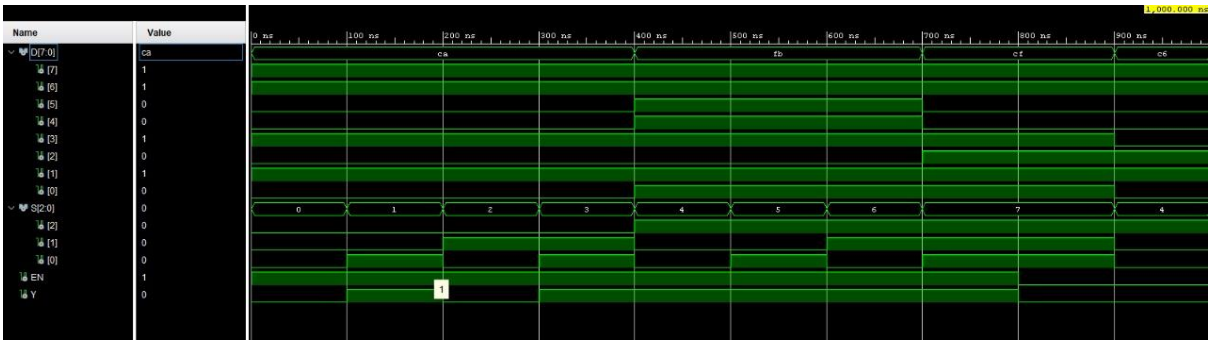


c. Simulation source file

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

entity Mux_8_to_1_Sim is
-- Port ( );
end Mux_8_to_1_Sim;

architecture Behavioral of Mux_8_to_1_Sim is

COMPONENT MUX_8_to_1
    PORT( D : in STD_LOGIC_VECTOR (7 downto 0);
          S : in STD_LOGIC_VECTOR (2 downto 0);
          EN : in STD_LOGIC;
          Y : out STD_LOGIC     );
end component;
SIGNAL D : STD_LOGIC_VECTOR (7 downto 0);
SIGNAL S : STD_LOGIC_VECTOR (2 downto 0);
SIGNAL EN : STD_LOGIC;
SIGNAL Y : STD_LOGIC;
begin
UUT : MUX_8_to_1
port map (
    D => D,
    S => S,
    EN => EN,
    Y => Y
    );
```

```
process
begin
--220618A  11 0101 1101 1100 1010
--220623J  11 0101 1101 1100 1111
--220411H  11 0101 1100 1111 1011
--220614H  11 0101 1101 1100 0110

EN <= '1';
D  <= "11001010" ;
S <= "000";
WAIT FOR 100ns;
S <= "001";
WAIT FOR 100ns;
S <= "010";
WAIT FOR 100ns;
S <= "011";
WAIT FOR 100ns;
D  <= "11111011" ;
S <= "100";
WAIT FOR 100ns;
S <= "101";
WAIT FOR 100ns;
S <= "110";
WAIT FOR 100ns;
S <= "111";
D  <= "11001111" ;
WAIT FOR 100ns;
EN <= '0';
WAIT FOR 100ns;
D  <= "11000110" ;
S <= "100";
WAIT FOR 100ns;
end process;
end Behavioral;
```

d. Timing diagram



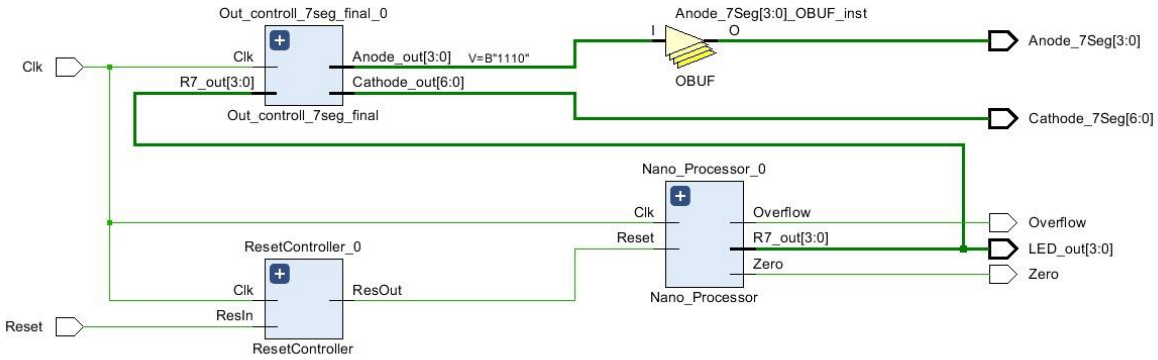
J. Nano 7seg final

a. Design Source File

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

entity Nano_Processor_with_7SegOut_final is
    Port ( Clk : in STD_LOGIC;
    Reset : in STD_LOGIC;
    --ROMSel: in STD_LOGIC;
    Overflow : out STD_LOGIC;
    Zero : out STD_LOGIC;
    LED_out : out STD_LOGIC_VECTOR (3 downto 0);
    Anode_7Seg : out STD_LOGIC_VECTOR (3 downto 0);
    Cathode_7Seg : out STD_LOGIC_VECTOR (6 downto 0));
end Nano_Processor_with_7SegOut_final;

architecture Behavioral of Nano_Processor_with_7SegOut_final is
component Nano_Processor
    Port ( Clk : in STD_LOGIC;
    Reset : in STD_LOGIC;
    --ROMSel: in STD_LOGIC;
    Overflow : out STD_LOGIC;
    Zero : out STD_LOGIC;
    R7_out : out STD_LOGIC_VECTOR (3 downto 0));
end component;

component Out_controll_7seg_final
    Port ( Clk : in STD_LOGIC;
    R7_out : in STD_LOGIC_VECTOR (3 downto 0);
    Anode_out : out STD_LOGIC_VECTOR (3 downto 0);
    Cathode_out : out STD_LOGIC_VECTOR (6 downto 0));
end component;

component ResetController
    Port ( ResIn : in STD_LOGIC;
    Clk : in STD_Logic;
    ResOut : out STD_LOGIC);
end component;

signal R7_output:std_logic_vector(3 downto 0);
signal ResINPUT:std_logic;
signal Anode_7_out : std_logic_vector(3 downto 0) :="1110";
```

```
begin

    ResetController_0:ResetController
        port map(
        ResIn=>Reset,
        Clk=>Clk,
        ResOut=>ResINPUT );

    Nano_Processor_0:Nano_processor
        port map(
        Clk=>Clk,
        Reset=>ResINPUT,
        --ROMSel=>ROMSel,
        Overflow=>Overflow,
        Zero=>Zero,
        R7_out=>R7_output );

LED_out<=R7_output;

    Out_controll_7seg_final_0 :Out_controll_7seg_final
        Port map (
        Clk =>Clk,
        R7_out=>R7_output,
        Anode_out=>Anode_7_out,
        Cathode_out=>Cathode_7Seg);

Anode_7Seg <= Anode_7_out;
end Behavioral;
```

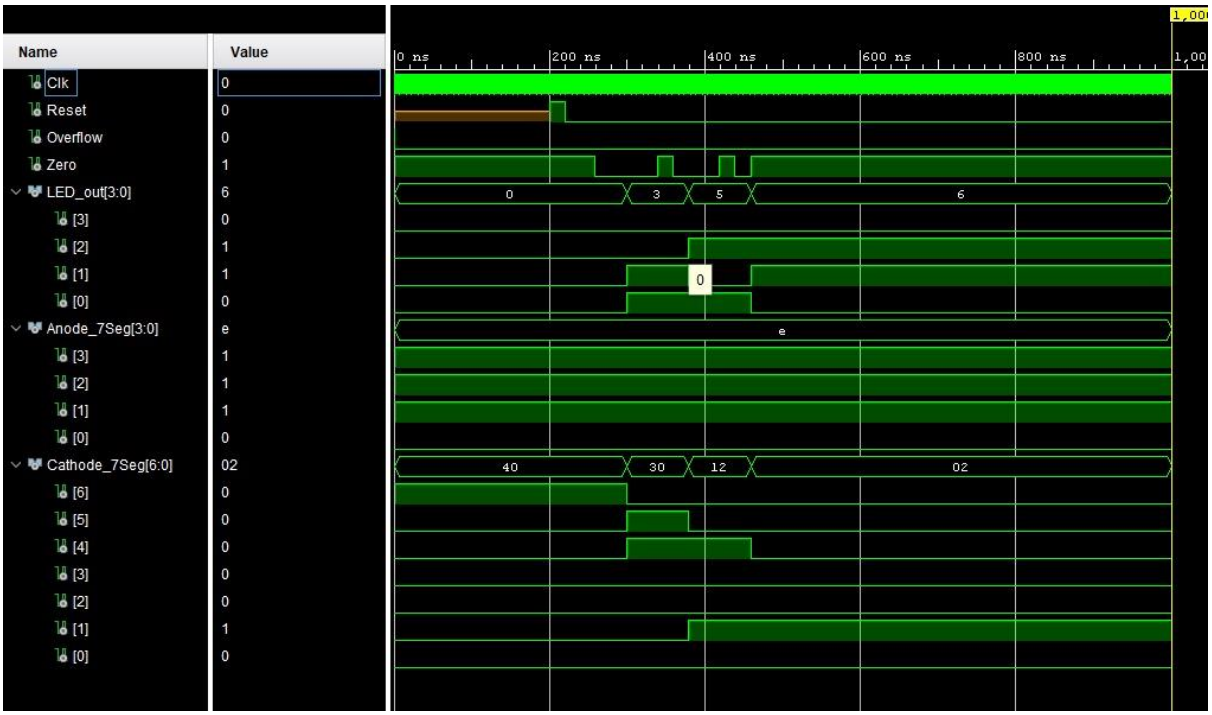
b. Elaborated Diagram



c. Simulation source file

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

entity TB_Nano_Processor_with_7SegOut_final is
--  Port ( );
end TB_Nano_Processor_with_7SegOut_final;

architecture Behavioral of TB_Nano_Processor_with_7SegOut_final is
component Nano_Processor_with_7SegOut_final
    Port ( Clk : in STD_LOGIC;
    Reset : in STD_LOGIC;
    --ROMSel: in STD_LOGIC;
    Overflow : out STD_LOGIC;
    Zero : out STD_LOGIC;
    LED_out : out STD_LOGIC_VECTOR (3 downto 0);
    Anode_7Seg : out STD_LOGIC_VECTOR (3 downto 0);
    Cathode_7Seg : out STD_LOGIC_VECTOR (6 downto 0));
end component;

signal Clk : std_logic := '0';
signal Reset : std_logic;
--signal ROMSel : std_logic:='0';
signal Overflow,Zero: std_logic;
signal LED_out,Anode_7Seg : STD_LOGIC_VECTOR (3 downto 0);
signal Cathode_7Seg :STD_LOGIC_VECTOR (6 downto 0);

begin
    UUT:Nano_Processor_with_7SegOut_final
    port map(
    Clk=>Clk,
    Reset=>Reset,
    --ROMSel=>ROMSel,
    OverFlow=>Overflow,
    Zero=>Zero,
    LED_out => LED_out,
    Anode_7Seg => Anode_7Seg,
    Cathode_7Seg => Cathode_7Seg);

process
begin
    wait for 1ns;
    Clk<=not(Clk);
    end process;
    process
    begin
    wait for 200ns;
    Reset<='1';
    wait for 20ns;
    Reset<='0';
    wait for 1500ns;
    --ROMSel<='1';
    wait for 200ns;
    Reset<='1';
    wait for 10ns;
    Reset<='0';
    wait;
end process;
end Behavioral;
```

d. Timing diagram



K. Nano Processor

a. Design Source File

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

entity Nano_Processor is
    Port ( Clk : in STD_LOGIC;
    Reset : in STD_LOGIC;
    --ROMSel: in STD_LOGIC;
    Overflow : out STD_LOGIC;
    Zero : out STD_LOGIC;
    R7_out : out STD_LOGIC_VECTOR (3 downto 0));
end Nano_Processor;

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

signal SlowClk:std_logic;
signal EN : STD_LOGIC:='1';

component MUX_8_to_1_4
    Port ( R0 : in STD_LOGIC_VECTOR (3 downto 0);
    R1 : in STD_LOGIC_VECTOR (3 downto 0);
    R2 : in STD_LOGIC_VECTOR (3 downto 0);
    R3 : in STD_LOGIC_VECTOR (3 downto 0);
    R4 : in STD_LOGIC_VECTOR (3 downto 0);
    R5 : in STD_LOGIC_VECTOR (3 downto 0);
    R6 : in STD_LOGIC_VECTOR (3 downto 0);
    R7 : in STD_LOGIC_VECTOR (3 downto 0);
    S : in STD_LOGIC_VECTOR (2 downto 0);
    EN : in STD_LOGIC;
    Y : out STD_LOGIC_VECTOR (3 downto 0));
end component;

component RegisterBank
    Port ( Clk : in STD_LOGIC;
    Reset : in STD_LOGIC;
    D : in STD_LOGIC_VECTOR (3 downto 0);
    R0 : out STD_LOGIC_VECTOR (3 downto 0);
    R1 : out STD_LOGIC_VECTOR (3 downto 0);
    R2 : out STD_LOGIC_VECTOR (3 downto 0);
    R3 : out STD_LOGIC_VECTOR (3 downto 0);
    R4 : out STD_LOGIC_VECTOR (3 downto 0);
    R5 : out STD_LOGIC_VECTOR (3 downto 0);
    R6 : out STD_LOGIC_VECTOR (3 downto 0);
    R7 : out STD_LOGIC_VECTOR (3 downto 0);
    I : in STD_LOGIC_VECTOR (2 downto 0));
end component;
```

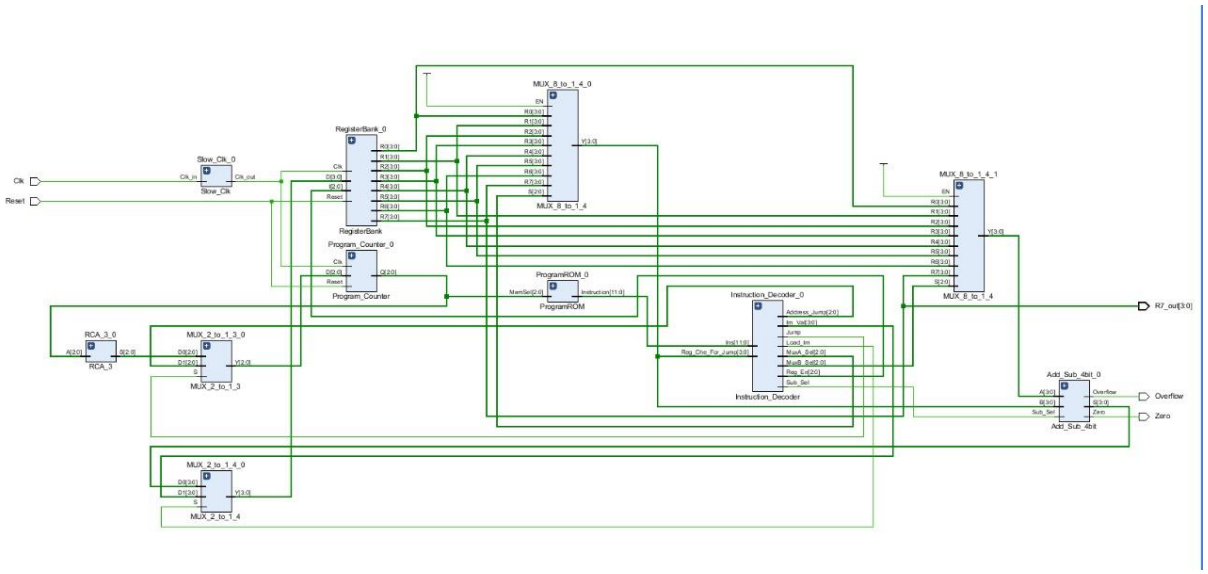
```
component Program_Counter
    Port ( Clk : in STD_LOGIC;
    Reset : in STD_LOGIC;
    D : in STD_LOGIC_VECTOR (2 downto 0);
    Q : out STD_LOGIC_VECTOR (2 downto 0));
end component;

signal Ins0:std_logic_vector(11 downto 0);
signal PC_out,J_A,A_3,M0_S,M1_S,R_E:std_logic_vector(2 downto 0);
signal I_V,M3,M1,M0,A_4:std_logic_vector(3 downto 0);
signal R0,R1,R2,R3,R4,R5,R6,R7:std_logic_vector(3 downto 0);
signal J,L_S,Sub:std_logic;
signal M2:std_logic_vector(2 downto 0);

begin
    Slow_Clk_0:Slow_Clk
        port map(
        Clk_in=>Clk,
        Clk_out=>SlowClk);
    Program_Counter_0:Program_Counter
        port map(
        Clk=>SlowClk,
        Reset=>Reset,
        D=>M2,
        Q=>PC_out);
    ProgramROM_0:ProgramROM
        port map(
        MemSel=>PC_out,
        Instruction=>Ins0);
    MUX_2_to_1_4_0:MUX_2_to_1_4
        port map(
        D0=>A_4,
        D1=>I_V,
        S=>L_S,
        Y=>M3);
    MUX_2_to_1_3_0:MUX_2_to_1_3
        port map(
        D0=>A_3,
        D1=>J_A,
        S=>J,
        Y=>M2);
    MUX_8_to_1_4_0:MUX_8_to_1_4
        port map(
        R0=>R0,
        R1=>R1,
        R2=>R2,
        R3=>R3,
        R4=>R4,
        R5=>R5,
        R6=>R6,
        R7=>R7,
```

```
component Add_Sub_4bit
    Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
    B : in STD_LOGIC_VECTOR (3 downto 0);
    S : out STD_LOGIC_VECTOR (3 downto 0);
    Overflow : out STD_LOGIC;
    Sub_Sel : in STD_LOGIC;
    Zero : out STD_LOGIC
);
end component;

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

component Instruction_Decoder
    Port (
    Ins : in STD_LOGIC_VECTOR (11 downto 0);
    Reg_Che_For_Jump : in STD_LOGIC_VECTOR (3 downto 0);
    Reg_En : out STD_LOGIC_VECTOR(2 downto 0);
    Load_Im : out STD_LOGIC;
    Im_Val : out STD_LOGIC_VECTOR(3 downto 0);
    MuxA_Sel : out STD_LOGIC_VECTOR (2 downto 0);
    MuxB_Sel : out STD_LOGIC_VECTOR (2 downto 0);
    Sub_Sel : out STD_LOGIC;
    Jump : out STD_LOGIC;
    Address_Jump : out STD_LOGIC_VECTOR (2 downto 0));
end component;

component MUX_2_to_1_3
    Port ( D0 : in STD_LOGIC_VECTOR (2 downto 0);
    D1 : in STD_LOGIC_VECTOR (2 downto 0);
    S : in STD_LOGIC;
    Y : out STD_LOGIC_VECTOR (2 downto 0));
end component;

component MUX_2_to_1_4
    Port ( D0 : in STD_LOGIC_VECTOR(3 downto 0);
    D1 : in STD_LOGIC_VECTOR(3 downto 0);
    S : in STD_LOGIC;
    Y : out STD_LOGIC_VECTOR(3 downto 0));
end component;

component ProgramROM
    Port ( MemSel : in STD_LOGIC_VECTOR (2 downto 0);
    Instruction : out STD_LOGIC_VECTOR (11 downto 0));
end component;
```

component

```
            Y=>M1,
            EN => EN,
            S=>M1_S);
    RegisterBank_0:RegisterBank
            port map(
            Clk=>SlowClk,
            Reset=>Reset,
            D=>M3,
            R0=>R0,
            R1=>R1,
            R2=>R2,
            R3=>R3,
            R4=>R4,
            R5=>R5,
            R6=>R6,
            R7=>R7,
            I=>R_E);
    Add_Sub_4bit_0:Add_Sub_4bit
            port map(
            A=>M1,
            B=>M0,
            Sub_Sel=>Sub,
            S=>A_4,
            Overflow=>Overflow,
            Zero=>Zero
            );
    RCA_3_0:RCA_3
            port map(
            A=>PC_out,
            S=>A_3);
    Instruction_Decoder_0:Instruction_Decoder
            Port map (
            Ins =>Ins0,
            Reg_Che_For_Jump =>M0,
            Reg_En =>R_E,
            Load_Im =>L_S,
            Im_Val =>I_V,
            MuxA_Sel =>M0_S,
            MuxB_Sel =>M1_S,
            Sub_Sel =>Sub,
            Jump =>J,
            Address_Jump =>J_A);
R7_out<=R7;
end Behavioral;
```

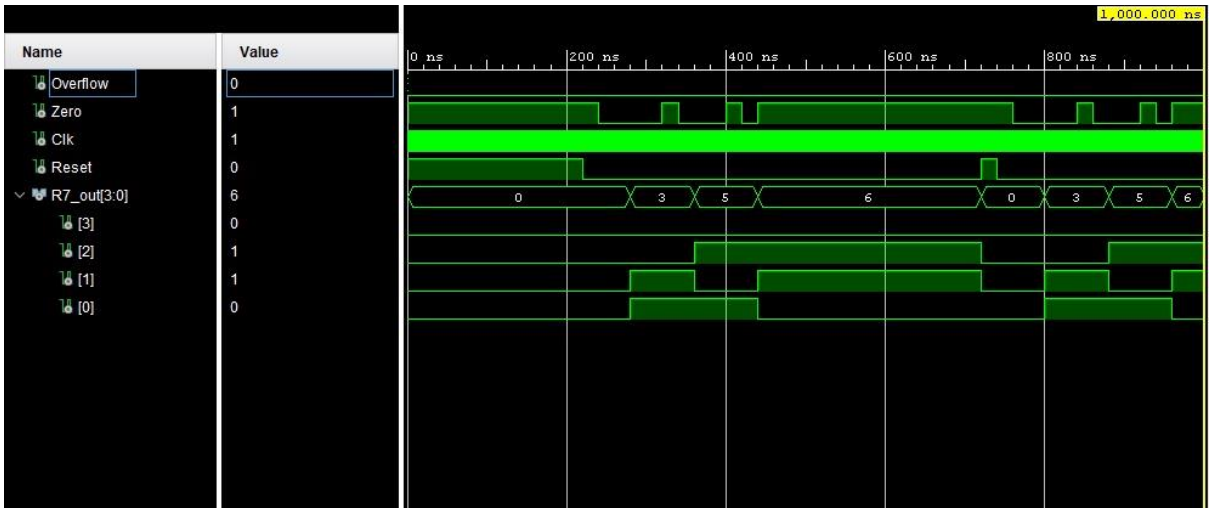
b. Elaborated Diagram



c. Simulation source file

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

entity sim_Nano_Processor is
-- Port ( );
end sim_Nano_Processor;
architecture Behavioral of sim_Nano_Processor is

component Nano_Processor
    Port (  Clk : in STD_LOGIC;
            Reset : in STD_LOGIC;
            Overflow : out STD_LOGIC;
            Zero : out STD_LOGIC;
            R7_out : out STD_LOGIC_VECTOR (3 downto 0));
end component;

signal Overflow,Zero:std_logic;
signal Clk,Reset:std_logic:='1';
signal R7_out :STD_LOGIC_VECTOR (3 downto 0);

begin
UUT:Nano_processor
    port map(
        Clk=>Clk,
        Reset=>Reset,
        OverFlow=>Overflow,
        Zero=>Zero,
        R7_out=>R7_out
    );

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

process
begin
    wait for 200ns;
    Reset<='1';
    wait for 20ns;
    Reset<='0';
    wait for 500ns;
    Reset<='1';
    wait for 20ns;
    Reset<='0';
    wait for 1500ns;
    Reset<='1';
    wait for 20ns;
    Reset<='0';
    wait;
end process;
end Behavioral;
```

d. Timing diagram

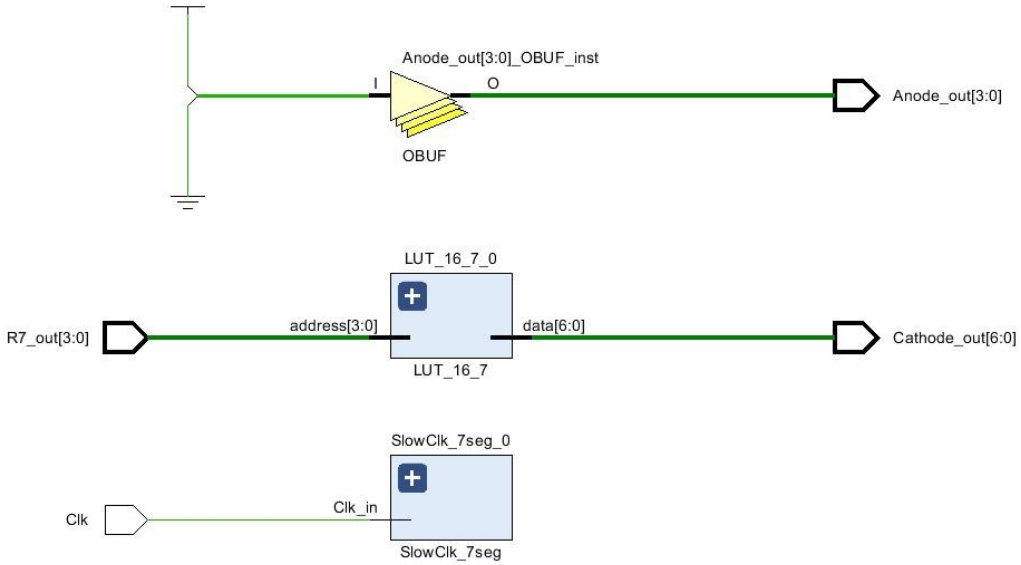


L. Out Control 7 seg

a. Design Source File

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

entity Out_controll_7seg_final is
    Port ( Clk : in STD_LOGIC;
    R7_out : in STD_LOGIC_VECTOR (3 downto 0);
    Anode_out : out STD_LOGIC_VECTOR (3 downto 0);
    Cathode_out : out STD_LOGIC_VECTOR (6 downto 0));
end Out_controll_7seg_final;

architecture Behavioral of Out_controll_7seg_final is

component SlowClk_7seg
    Generic (count_max:integer:= 15000000); --15 million
    Port ( Clk_in : in STD_LOGIC;
        Clk_out : out STD_LOGIC);
end component;

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

b. Elaborated Diagram



c. Simulation source file

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

entity Sim_Out_Control_7Seg_final is
--  Port ( );
end Sim_Out_Control_7Seg_final;

architecture Behavioral of Sim_Out_Control_7Seg_final is
component Out_controll_7seg_final
    Port (  Clk : in STD_LOGIC;
            R7_out : in STD_LOGIC_VECTOR (3 downto 0);
            Anode_out : out STD_LOGIC_VECTOR (3 downto 0);
            Cathode_out : out STD_LOGIC_VECTOR (6 downto 0));
end component ;
signal Clk: std_logic := '1';
signal R7_out,Anode_out :STD_LOGIC_VECTOR (3 downto 0);
signal Cathode_out :STD_LOGIC_VECTOR (6 downto 0);

begin
UUT:Out_controll_7seg_final
    port map(
        Clk=>Clk,
        R7_out=>R7_out,
        Anode_out => Anode_out,
        Cathode_out=>Cathode_out);

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

process
begin
--220618A  11 0101 1101 1100 1010
--220623J  11 0101 1101 1100 1111
--220411H  11 0101 1100 1111 1011
--220614H  11 0101 1101 1100 0110
    wait for 100ns;
    R7_out <= "1010";
    wait for 100ns;
    R7_out <= "1100";
    wait for 100ns;
    R7_out <= "1111";
    wait for 100ns;
    R7_out <= "1100";
    wait for 100ns;
    R7_out <= "1011";
    wait for 100ns;
    R7_out <= "0110";
    wait;
end process;
end Behavioral;
```

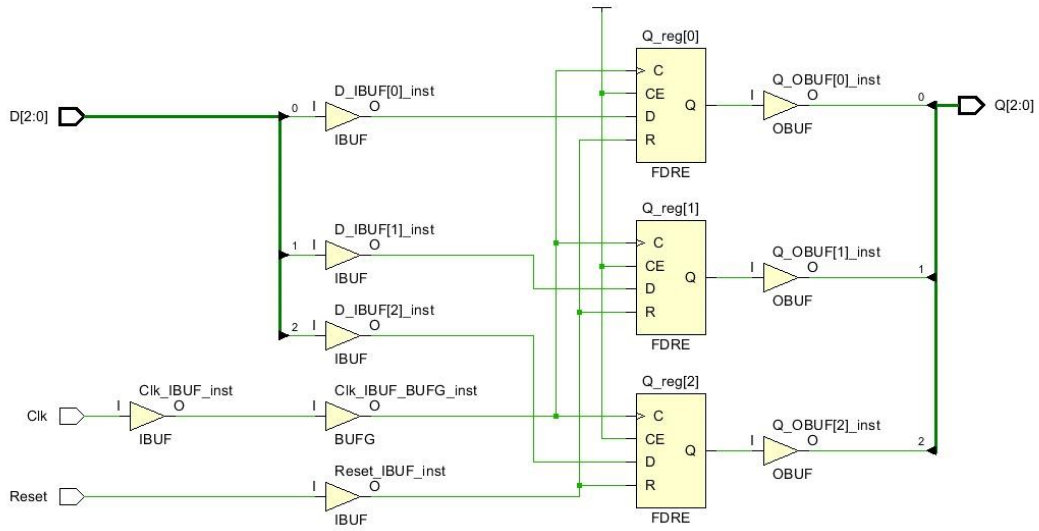
d. Timing diagram

M. Program Counter

a. Design Source File

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

entity Program_Counter is
    Port ( Clk : in STD_LOGIC;
    Reset : in STD_LOGIC;
    D : in STD_LOGIC_VECTOR (2 downto 0);
    Q : out STD_LOGIC_VECTOR (2 downto 0));
end Program_Counter;

architecture Behavioral of Program_Counter is

begin
    process (Clk)
    begin
        if (rising_edge(Clk)) then
            if Reset = '1' then
                Q <= "000";
            else
                Q <= D;
                --Qbar <= not D;
            end if;
        end if;
    end process;
end Behavioral;
```

b. Elaborated Diagram



c. Simulation source file

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


entity TB_Program_Counter is
-- Port ( );
end TB_Program_Counter;
architecture Behavioral of TB_Program_Counter is

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

signal Q,D :STD_LOGIC_VECTOR (2 downto 0);
signal Res,Clk:std_logic:='0';

begin
UUT:Program_Counter
    port map(
        Clk=>Clk,
        Reset=>Res,
        D=>D,
        Q=>Q);
```

```
process
begin
--220618A  11 0101 1101 1100 1010
--220623J  11 0101 1101 1100 1111
--220411H  11 0101 1100 1111 1011
--220614H  11 0101 1101 1100 0110
    wait for 40ns;
    Clk<=not(Clk);
end process;

process
begin
    Res<='1';
    wait for 50ns;
    D<="010";
    wait for 50ns;
    D<="001";
    wait for 50ns;
    D<="111";
    wait for 50ns;
    Res<='0';
    wait for 100ns;
    D<="001";
    wait for 100ns;
    D<="011";
    wait for 100ns;
    D<="111";
    wait for 100ns;
    D<="110";
    wait for 100ns;
    D<="001";
    wait;
end process;
end Behavioral;
```

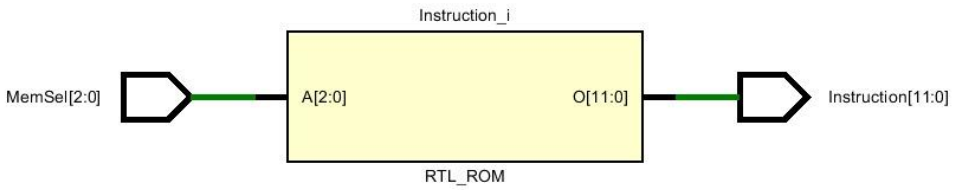
d. Timing diagram

N. Program Rom

a. Design Source File

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

entity ProgramROM is
    Port ( MemSel : in STD_LOGIC_VECTOR (2 downto 0);
    Instruction : out STD_LOGIC_VECTOR (11 downto 0));
end ProgramROM;

architecture Behavioral of ProgramROM is

    type rom_type is array (0 to 7) of std_logic_vector(11 downto 0);
        signal ProgramROM_0 : rom_type :=(
        "100010000011", -- MOVI R1,3 --line->0
        "100110000001", -- MOVI R3,1
        "010110000000", -- NEG R3
        "001110010000", -- ADD R7,R1 --line->3
        "000010110000", -- ADD R1,R3
        "110010000111", -- JZR R1,7
        "110000000011", -- JZR R0,3
        "110000000111" -- JZR R0,7 --line->7
    );
begin
    Instruction <= ProgramROM_0(to_integer(unsigned(MemSel)));
end Behavioral;
```

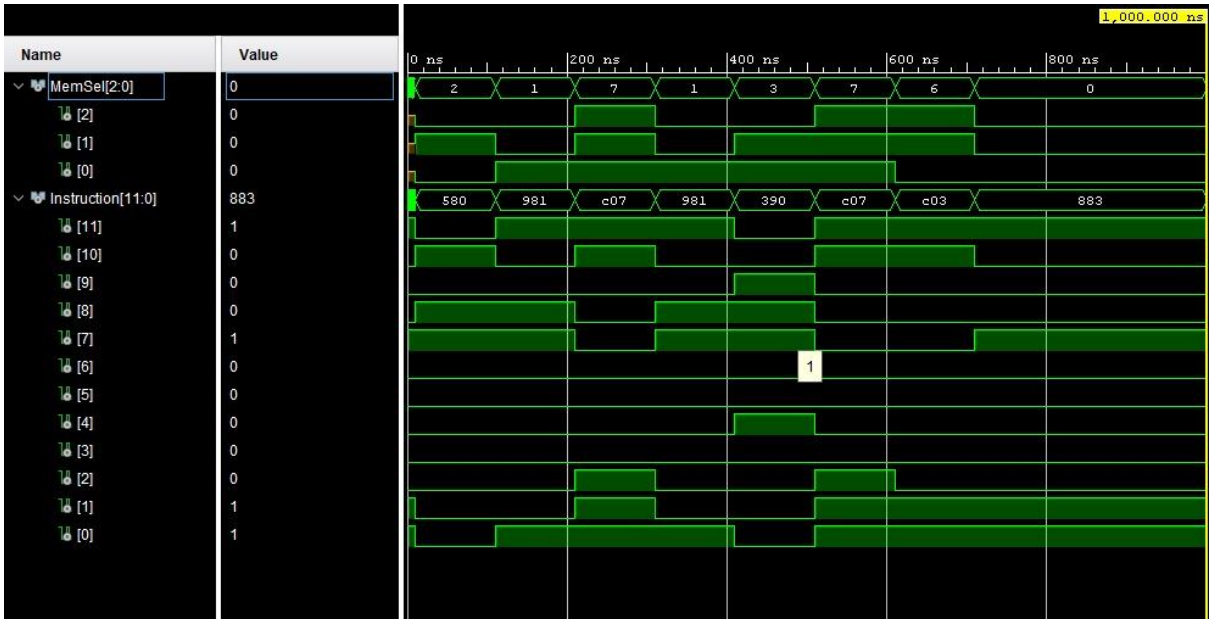
b. Elaborated Diagram



c. Simulation source file

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

entity RomSim is
-- Port ( );
end RomSim;
architecture Behavioral of RomSim is
component ProgramROM
    port(
        MemSel : in STD_LOGIC_VECTOR (2 downto 0);
        Instruction : out STD_LOGIC_VECTOR (11 downto 0)
    );
end component;
Signal MemSel : STD_LOGIC_VECTOR(2 downto 0);
Signal Instruction : STD_LOGIC_VECTOR(11 downto 0);

begin
UUT : ProgramROM
    port map(
        MemSel => MemSel,
        Instruction => Instruction
    );
process
--220618A  11 0101 1101 1100 1010
--220623J  11 0101 1101 1100 1111
--220411H  11 0101 1100 1111 1011
--220614H  11 0101 1101 1100 0110
begin
    wait for 10ns;
    MemSel <= "010";
    wait for 100ns;
    MemSel <= "001";
    wait for 100ns;
    MemSel <= "111";
    wait for 100ns;
    MemSel <= "001";
    wait for 100ns;
    MemSel <= "011";
    wait for 100ns;
    MemSel<="111";
    Wait for 100ns;
    MemSel<="110";
    Wait for 100ns;
    MemSel<="000";
    wait;
end process;
end Behavioral;
```

d. Timing diagram

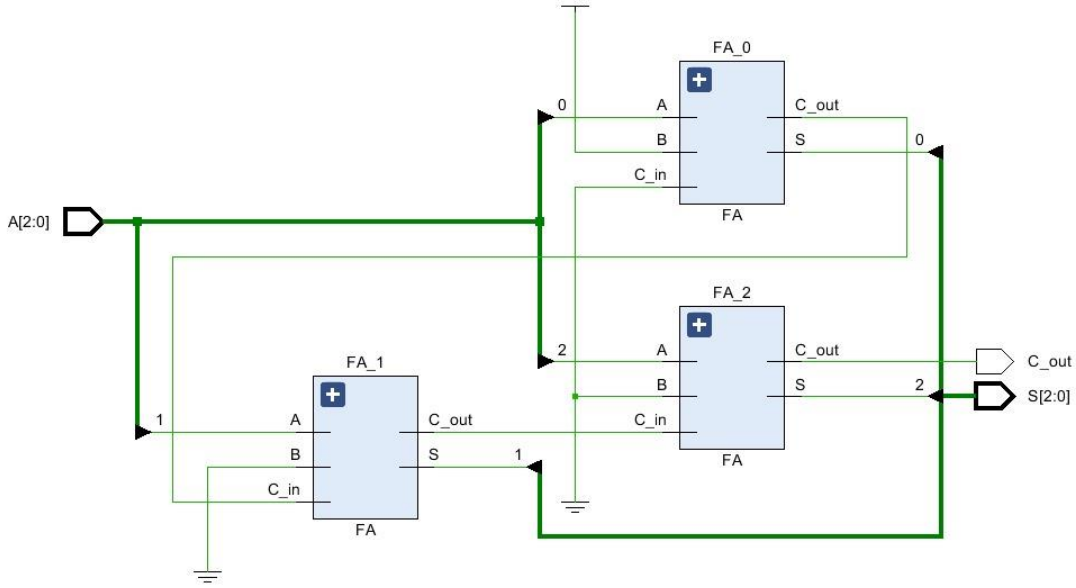


O. RCA

a. Design Source File

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

entity RCA_3 is
    Port ( A : in STD_LOGIC_VECTOR (2 downto 0);
    S: out STD_LOGIC_VECTOR (2 downto 0);
    C_out : out STD_LOGIC);
    end RCA_3;
architecture Behavioral of RCA_3 is
    component FA
    port (
    A: in std_logic;
    B: in std_logic;
    C_in: in std_logic;
    S: out std_logic;
    C_out: out std_logic);
    end component;
    signal FA0_S, FA0_C, FA1_S, FA1_C, FA2_S, FA2_C: std_logic;

    begin
    FA_0 : FA
        port map (
            A => A(0),
            B => '1',
            C_in => '0', -- Set to ground
            S => S(0),
            C_out => FA0_C);
    FA_1 : FA
        port map (
            A => A(1),
            B => '0',
            C_in => FA0_C,
            S => S(1),
            C_out => FA1_C);
    FA_2 : FA
        port map (
            A => A(2),
            B => '0',
            C_in => FA1_C,
            S => S(2),
            C_out => FA2_C);
            C_out <= FA2_C;
end Behavioral;
```

b. Elaborated Diagram

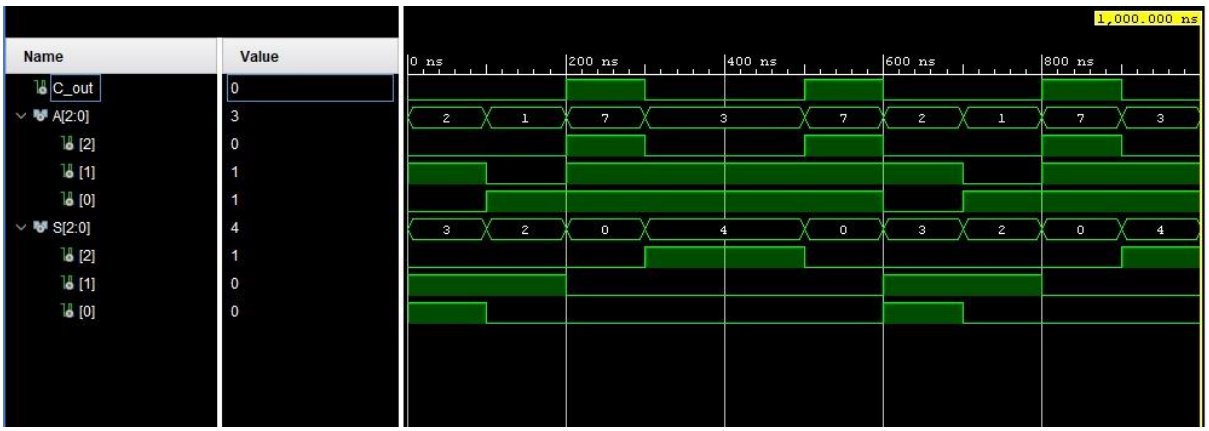


c. Simulation source file

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

entity TB_RCA_3 is
-- Port ( );
end TB_RCA_3;
architecture Behavioral of TB_RCA_3 is

component RCA_3
    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;
            C_out : out STD_LOGIC);
end component;
```

```
signal C_out:std_logic;
signal A,S :STD_LOGIC_VECTOR (2 downto 0);

begin
UUT:RCA_3
    port map(
        A=>A,
        S=>S,
        C_out=>C_out
    );
--220618A  11 0101 1101 1100 1010
--220623J  11 0101 1101 1100 1111
--220411H  11 0101 1100 1111 1011
--220614H  11 0101 1101 1100 0110
process
begin
    A<="010";
    wait for 100ns;
    A<="001";
    wait for 100ns;
    A<="111";
    wait for 100ns;
    A<="011";
    wait for 100ns;
    A<="011";
    wait for 100ns;
    A<="111";
    wait for 100ns;
end process;
end Behavioral;
```

d. Timing diagram

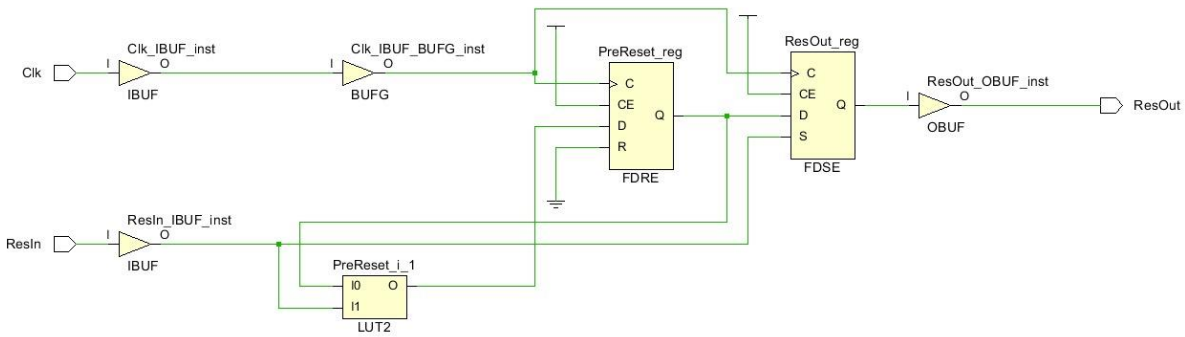


P. Reset Controller

a. Design Source File

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

entity ResetController is
    Port ( ResIn : in STD_LOGIC;
    Clk : in STD_LOGIC;
    ResOut : out STD_LOGIC);
end ResetController;

architecture Behavioral of ResetController is

signal PreReset:std_logic:='1';

begin
    process (Clk)
    begin
        if (rising_edge(Clk)) then
            if ResIn='1' then
                ResOut<='1';
                PreReset<='0';
            elsif PreReset='1' then
                ResOut<='1';
            else
                ResOut<='0';
            end if;
        end if;
    end process;
end Behavioral;
```

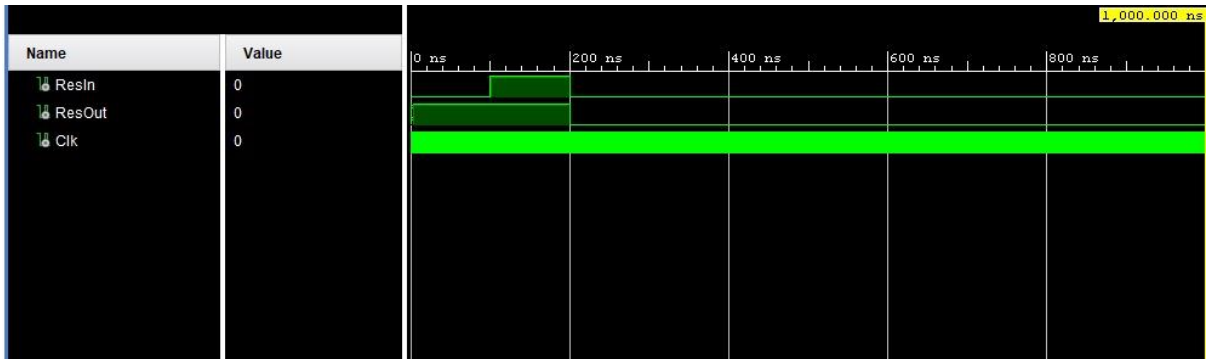
b. Elaborated Diagram



c. Simulation source file

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


entity Sim_Reset_Controller is
--  Port ( );
end Sim_Reset_Controller;

architecture Behavioral of Sim_Reset_Controller is
component ResetController
    Port (  ResIn : in STD_LOGIC;
            Clk : in STD_LOGIC;
            ResOut : out STD_LOGIC);
end component;
signal ResIn,ResOut:std_logic;
signal Clk:std_logic := '0';

begin
UUT:ResetController
    port map(
        Clk=>Clk,
        ResIn=>ResIn,
        ResOut => ResOut);

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

process
begin
    ResIn <= '0';
    wait for 100ns;
    ResIn <= '1';
    wait for 100ns;
    ResIn <= '0';
    wait;
end process;
end Behavioral;
```

d. Timing diagram

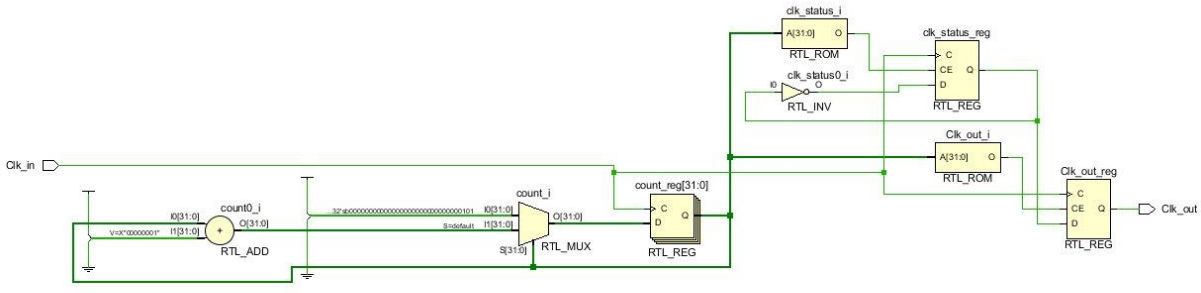


Q. Slow Clk

a. Design Source File

```
library IEEE;
use IEEE.STD_LOGIC_1164.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=5) then --count 15M pulses
            clk_status<=NOT (clk_status);
            Clk_out<=clk_status;
            count<=1; --Reset counter
        end if;
    end if;
end process;
end Behavioral;
```

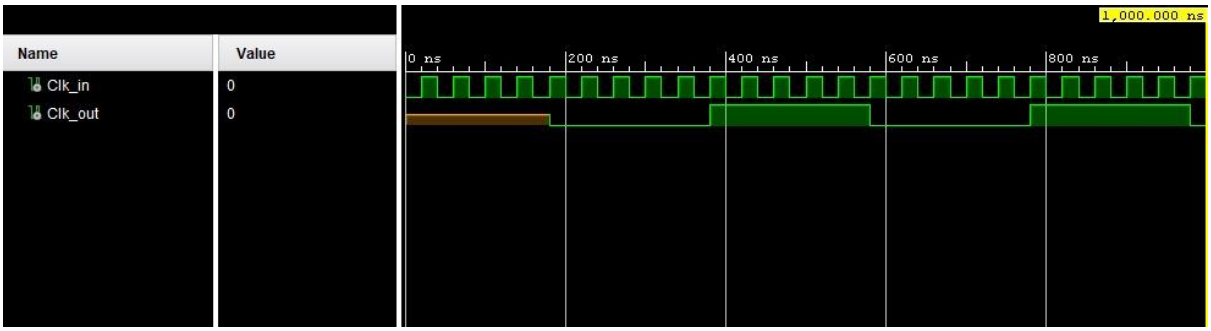
b. Elaborated Diagram



c. Simulation source file

```
library IEEE;
use IEEE.STD_LOGIC_1164.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_in : STD_LOGIC:='0';
signal Clk_out : STD_LOGIC;

begin
UUT: Slow_Clk
    port map(
        Clk_in=>Clk_in,
        Clk_out=>Clk_out );
process
begin
    wait for 20ns;
    Clk_in <= NOT(Clk_in);
end process;
end Behavioral;
```

d. Timing diagram

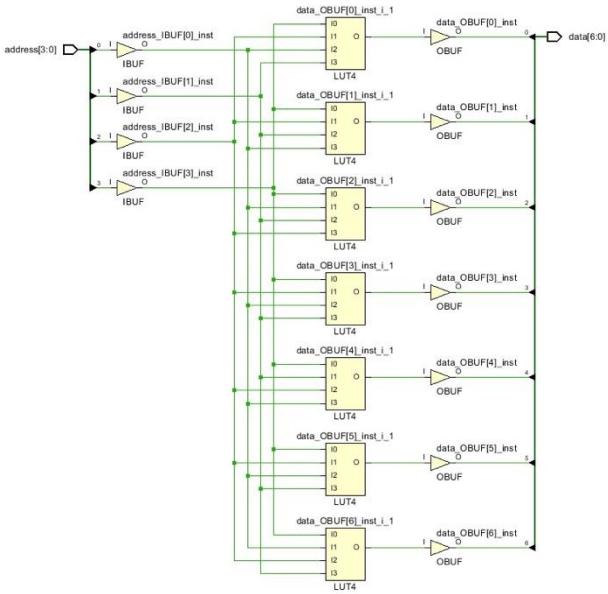


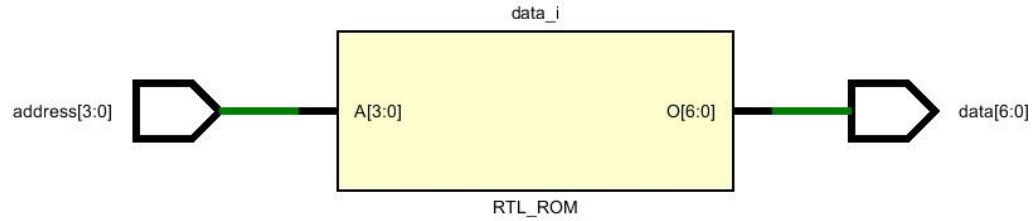
R. LUT

a. Design Source File

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.numeric_std.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   -line 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;
```

b. Elaborated Diagram



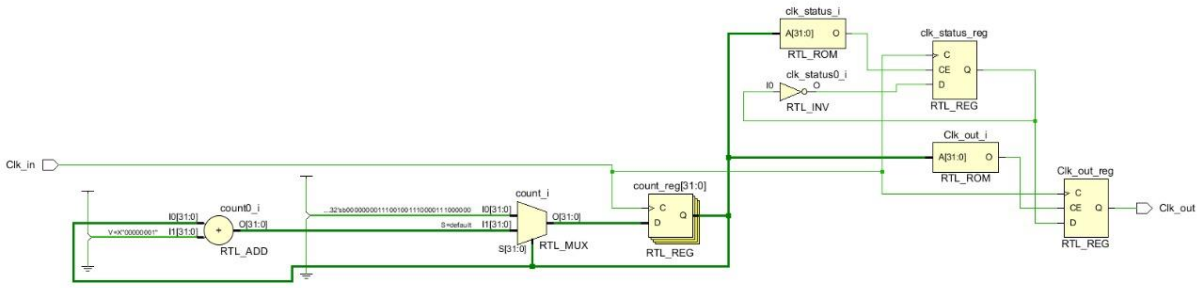


S. Slow Clk 7seg

a. Design Source File

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

entity SlowClk_7seg is
    Generic (count_max:integer:= 15000000); --100000000
    Port ( Clk_in : in STD_LOGIC;
    Clk_out : out STD_LOGIC);
end SlowClk_7seg;

architecture Behavioral of SlowClk_7seg 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=count_max) then
            clk_status<= not(clk_status);
            clk_out<= clk_status;
            count<=1;
        end if;
    end if;
end process;
end Behavioral;
```

b. Elaborated Diagram

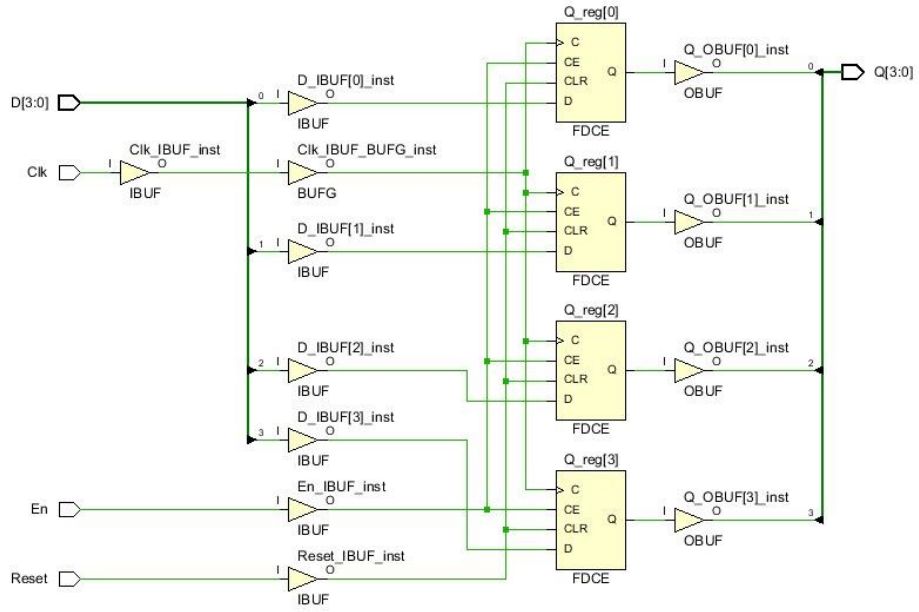


T. Register

a. Design Source File

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

entity Register_4bit is
    Port ( D : in STD_LOGIC_VECTOR (3 downto 0);
           Reset : in STD_LOGIC;
           Clk : in STD_LOGIC;
           Q : out STD_LOGIC_VECTOR (3 downto 0);
           En : in STD_LOGIC);
end Register_4bit;

architecture Behavioral of Register_4bit is

begin
process (Clk) begin
    if Reset = '0' then
        if (rising_edge(Clk)) then
            if En = '1' then
                Q <= D;
            end if;

        end if;
    else
         Q <= (others => '0');
    end if;
end process;

end Behavioral;
```

b. Schematic diagram

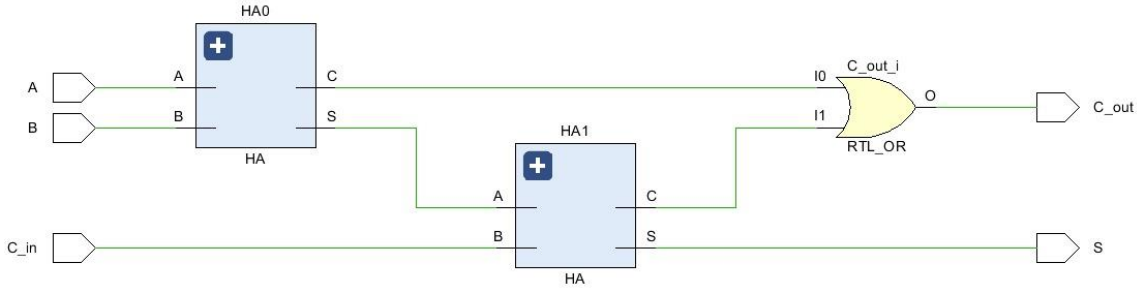


U. FA

a. Design Source File

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

library IEEE;
use IEEE.STD_LOGIC_1164.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,B:in std_logic;
        C,S:out std_logic);
    END COMPONENT;
    SIGNAL HA0_C,HA0_S,HA1_C,HA1_S:std_logic;
begin
    HA0:HA
        PORT MAP(
                A=>A,
                B=>B,
                S=>HA0_S,
                C=>HA0_C);
     HA1:HA
        PORT MAP(
                A=>HA0_S,
                B=>C_in,
                S=>HA1_S,
                C=>HA1_C);
    S<=HA1_S;
    C_out<=HA0_C OR HA1_C;


end Behavioral;
```

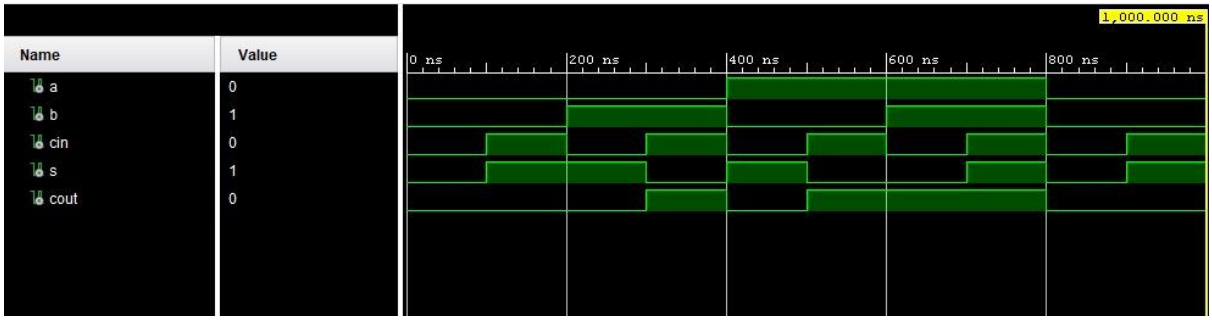
b. Elaborated Diagram



c. Simulation source file

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

entity FA_sim is
--  Port ( );
end FA_sim;

architecture Behavioral of FA_sim is
    COMPONENT FA
        port(A,B,C_in:in std_logic;
             S,C_out:out std_logic);
    END COMPONENT;
    SIGNAL a,b,cin,s,cout:std_logic;

begin
UUT:FA PORT MAP(
    A=>a,
    B=>b,
    C_in=>cin,
    S=>s,
    C_out=>cout);
 process
 begin
    a<='0';
    b<='0';
    cin<='0';
    wait for 100ns;
    cin<='1';
    wait for 100ns;
    b<='1';
    cin<='0';
    wait for 100ns;
    cin<='1';
    wait for 100ns;
    a<='1';
    b<='0';
    cin<='0';
    wait for 100ns;
    cin<='1';
    wait for 100ns;
    b<='1';
    cin<='0';
    wait for 100ns;
    cin<='1';
    wait for 100ns;
 end process;

end Behavioral;
```

d. Timing diagram

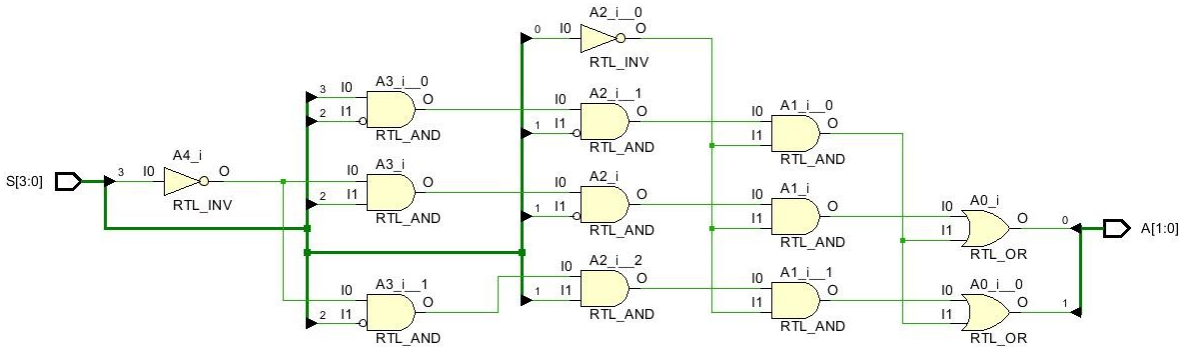


V. Encoder 4 to 2

a. Design Source File

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

entity encoder_4_to_2 is
    Port ( S : in STD_LOGIC_VECTOR (3 downto 0);
           A : out STD_LOGIC_VECTOR(1 downto 0));
end encoder_4_to_2;

architecture Behavioral of encoder_4_to_2 is

begin
    A(0)<=(NOT(S(3)) AND S(2) AND NOT(S(1)) AND NOT(S(0))) OR (S(3) AND NOT(S(2)) AND NOT(S(1)) AND NOT(S(0)));
    A(1)<=(NOT(S(3)) AND NOT(S(2)) AND S(1) AND NOT(S(0))) OR (S(3) AND NOT(S(2)) AND NOT(S(1)) AND NOT(S(0)));

end Behavioral;
```

b. Elaborated Diagram

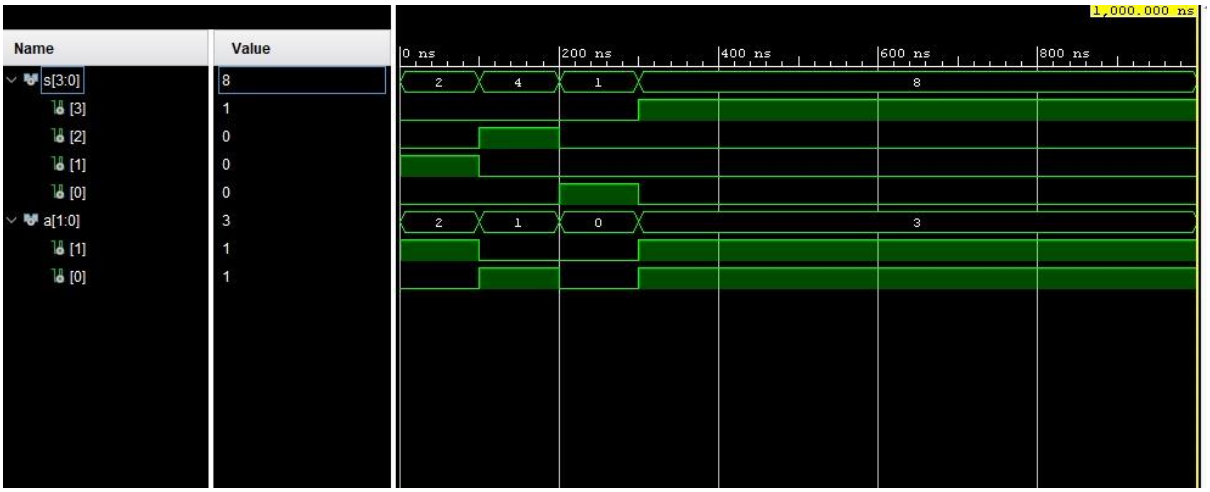


c. Simulation source file

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

entity encoder_4_to_2_sim is
end encoder_4_to_2_sim;

architecture Behavioral of encoder_4_to_2_sim is
    COMPONENT encoder_4_to_2
        port(S:in std_logic_vector(3 downto 0);
            A:out std_logic_vector(1 downto 0));
     END COMPONENT;
     SIGNAL s:std_logic_vector(3 downto 0);
     signal a:std_logic_vector(1 downto 0);
          --220618A  11 01 01 11 01 11 00 10 10
          --220623J  11 01 01 11 01 11 00 11 11
          --220411H  11 01 01 11 00 11 11 10 11
          --220614H  11 01 01 11 01 11 00 01 10

    begin
        UUT :encoder_4_to_2 PORT MAP(
            S=>s,
            A=>a);
      process
             begin
             s<="0010";
             wait for 100 ns;
             s<="0100";
             wait for 100ns;
             s<="0001";
             wait for 100 ns;
             s<="1000";
             wait;
         end process;

end Behavioral;
```

ole | Messages | Log | Reports

d. Timing diagram



W.Register Bank

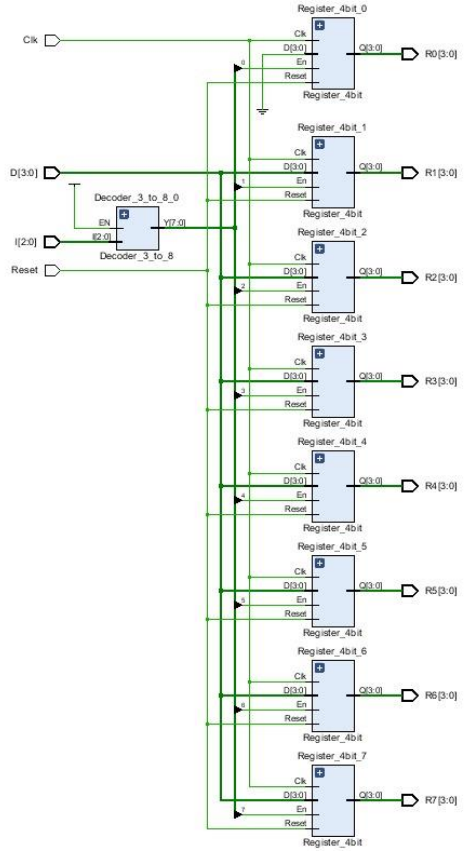
a. Design Source File

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

entity RegisterBank is
    Port ( Clk : in STD_LOGIC;
    Reset : in STD_LOGIC;
    D : in STD_LOGIC_VECTOR (3 downto 0);
    R0 : out STD_LOGIC_VECTOR (3 downto 0);
    R1 : out STD_LOGIC_VECTOR (3 downto 0);
    R2 : out STD_LOGIC_VECTOR (3 downto 0);
    R3 : out STD_LOGIC_VECTOR (3 downto 0);
    R4 : out STD_LOGIC_VECTOR (3 downto 0);
    R5 : out STD_LOGIC_VECTOR (3 downto 0);
    R6 : out STD_LOGIC_VECTOR (3 downto 0);
    R7 : out STD_LOGIC_VECTOR (3 downto 0);
    I : in STD_LOGIC_VECTOR (2 downto 0));
end RegisterBank;

architecture Behavioral of RegisterBank is

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

component Register_4bit
    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 EN : STD_LOGIC:='1';
signal Y : STD_LOGIC_VECTOR (7 downto 0);
```

```
begin
    Decoder_3_to_8_0:Decoder_3_to_8
        port map(
            I => I,
            EN => EN,
            Y => Y
        );
    Register_4bit_0:Register_4bit
        port map(
            D => "0000",
            EN => Y(0),
            Clk => Clk,
            Reset => Reset,
            Q => R0
        );
    Register_4bit_1:Register_4bit
        port map(
            D => D,
            EN => Y(1),
            Clk => Clk,
            Reset => Reset,
            Q => R1
        );
    Register_4bit_2:Register_4bit
        port map(
            D => D,
            EN => Y(2),
            Clk => Clk,
            Reset => Reset,
            Q => R2
        );
    Register_4bit_3:Register_4bit
        port map(
            D => D,
            EN => Y(3),
            Clk => Clk,
            Reset => Reset,
            Q => R3
        );
```

```
    );
Register_4bit_4:Register_4bit
    port map(
        D => D,
        EN => Y(4),
        Clk => Clk,
        Reset => Reset,
        Q => R4
    );
Register_4bit_5:Register_4bit
    port map(
        D => D,
        EN => Y(5),
        Clk => Clk,
        Reset => Reset,
        Q => R5
    );
Register_4bit_6:Register_4bit
    port map(
        D => D,
        EN => Y(6),
        Clk => Clk,
        Reset => Reset,
        Q => R6
    );
Register_4bit_7:Register_4bit
    port map(
        D => D,
        EN => Y(7),
        Clk => Clk,
        Reset => Reset,
        Q => R7
    );
end Behavioral;
```

b. Elaborated Diagram

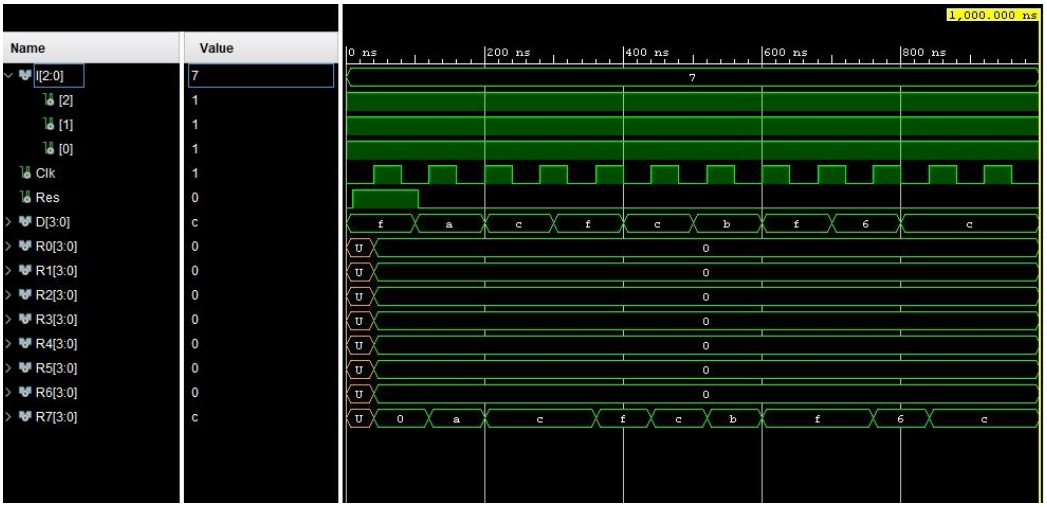


c. Simulation source file

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

entity TB_RegisterBank is
-- Port ( );
end TB_RegisterBank;
architecture Behavioral of TB_RegisterBank is

component RegisterBank
    Port (  Clk : in STD_LOGIC;
            Reset : in STD_LOGIC;
            D : in STD_LOGIC_VECTOR (3 downto 0);
            R0 : out STD_LOGIC_VECTOR (3 downto 0);
            R1 : out STD_LOGIC_VECTOR (3 downto 0);
            R2 : out STD_LOGIC_VECTOR (3 downto 0);
            R3 : out STD_LOGIC_VECTOR (3 downto 0);
            R4 : out STD_LOGIC_VECTOR (3 downto 0);
            R5 : out STD_LOGIC_VECTOR (3 downto 0);
            R6 : out STD_LOGIC_VECTOR (3 downto 0);
            R7 : out STD_LOGIC_VECTOR (3 downto 0);
            I : in STD_LOGIC_VECTOR (2 downto 0));
end component;

Signal I:STD_LOGIC_VECTOR (2 downto 0):="111";
Signal Clk, Res: std_logic:='0';
Signal D: STD_LOGIC_VECTOR (3 downto 0):="1111";
Signal R0, R1, R2, R3, R4, R5, R6, R7 : STD_LOGIC_VECTOR (3 downto 0);
```

```
begin
UUT: RegisterBank
    port map(
        Clk => Clk,
        Reset => Res,
        D => D,
        R0 => R0,
        R1 => R1,
        R2 => R2,
        R3 => R3,
        R4 => R4,
        R5 => R5,
        R6 => R6,
        R7 => R7,
        I => I
    );
process
begin
    wait for 40ns;
    Clk <= Not(Clk);
    end process;
    process begin
    wait for 10ns;
    Res <= Not(Res);
    wait for 95ns;
    Res <= Not(Res);
    wait for 5ns;
    I <= "111";
    wait;
end process;
--220618A  11 0101 1101 1100 1010
--220623J  11 0101 1101 1100 1111
--220411H  11 0101 1100 1111 1011
--220614H  11 0101 1101 1100 0110
process
begin
    wait for 100ns;
    D <= "1010";
    wait for 100ns;
    D <= "1100";
    wait for 100ns;
    D <= "1111";
    wait for 100ns;
    D <= "1100";
    wait for 100ns;
    D <= "1011";
    wait for 100ns;
    D <= "1111";
    wait for 100ns;
    D <= "0110";
    wait for 100ns;
    D <= "1100";
    wait;
```

d. Timing diagram



X. Constraints file

```
## 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 {LED_out[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {LED_out[0]}]
set_property PACKAGE_PIN E19 [get_ports {LED_out[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {LED_out[1]}]
set_property PACKAGE_PIN U19 [get_ports {LED_out[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {LED_out[2]}]
set_property PACKAGE_PIN V19 [get_ports {LED_out[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {LED_out[3]}]
set_property PACKAGE_PIN P1 [get_ports {Zero}]
set_property IOSTANDARD LVCMOS33 [get_ports {Zero}]
set_property PACKAGE_PIN L1 [get_ports {Overflow}]
set_property IOSTANDARD LVCMOS33 [get_ports {Overflow}]

##7 segment display
set_property PACKAGE_PIN W7 [get_ports {Cathode_7Seg[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Cathode_7Seg[0]}]
set_property PACKAGE_PIN W6 [get_ports {Cathode_7Seg[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Cathode_7Seg[1]}]
set_property PACKAGE_PIN U8 [get_ports {Cathode_7Seg[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Cathode_7Seg[2]}]
set_property PACKAGE_PIN V8 [get_ports {Cathode_7Seg[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Cathode_7Seg[3]}]
set_property PACKAGE_PIN U5 [get_ports {Cathode_7Seg[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Cathode_7Seg[4]}]
set_property PACKAGE_PIN V5 [get_ports {Cathode_7Seg[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Cathode_7Seg[5]}]
set_property PACKAGE_PIN U7 [get_ports {Cathode_7Seg[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Cathode_7Seg[6]}]

set_property PACKAGE_PIN U2 [get_ports {Anode_7Seg[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Anode_7Seg[0]}]
set_property PACKAGE_PIN U4 [get_ports {Anode_7Seg[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Anode_7Seg[1]}]
set_property PACKAGE_PIN V4 [get_ports {Anode_7Seg[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Anode_7Seg[2]}]
set_property PACKAGE_PIN W4 [get_ports {Anode_7Seg[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Anode_7Seg[3]}]

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

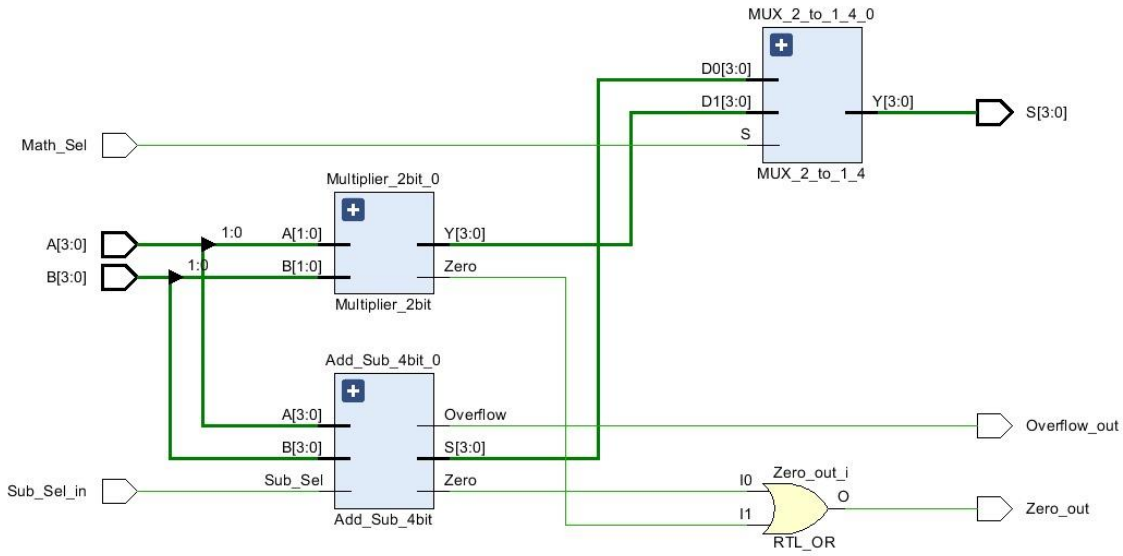
Y. Add-Sub-Mul unit

a. Design Source File

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

entity Add_Sub_Mul_Unit is
    Port (   A : in STD_LOGIC_VECTOR (3 downto 0);
             B : in STD_LOGIC_VECTOR (3 downto 0);
             Math_Sel : in STD_LOGIC;          --select Add_Sub_4bit and Multiplier_2bit
             Sub_Sel_in : in STD_LOGIC;          --only for Add_Sub_4bit
             S : out STD_LOGIC_VECTOR (3 downto 0);
             Zero_out : out std_logic;
             Overflow_out :out STD_LOGIC);
end Add_Sub_Mul_Unit;

architecture Behavioral of Add_Sub_Mul_Unit is

component Add_Sub_4bit
    port( A : in STD_LOGIC_VECTOR (3 downto 0);
        B : in STD_LOGIC_VECTOR (3 downto 0);
        Sub_Sel : in STD_LOGIC;
        S : out STD_LOGIC_VECTOR (3 downto 0);
        Zero : out std_logic;
        Overflow :out STD_LOGIC);
end component;

--Since this is a 4 bit processor it can only give 4 bit output
--Therefore 2x2 multiplier is used
component Multiplier_2bit
    port( A : in STD_LOGIC_VECTOR (1 downto 0);
          B : in STD_LOGIC_VECTOR (1 downto 0);
          Y : out STD_LOGIC_VECTOR (3 downto 0);
          Zero : out STD_LOGIC);
end component;

component MUX_2_to_1_4
        port (
            D0, D1 : in STD_LOGIC_VECTOR (3 downto 0);
            S : in STD_LOGIC;
            Y : out STD_LOGIC_VECTOR (3 downto 0)
        );
end component;

signal Out_Add_Sub : std_logic_vector(3 downto 0);
signal Out_Multiplier : std_logic_vector(3 downto 0);
signal Mux_Out : std_logic_vector(3 downto 0);
signal Overflow_Add, Zero_Add, Zero_Mul : std_logic;
```

```
begin

  Add_Sub_4bit_0: Add_Sub_4bit
      port map(
        A => A,
        B => B,
        Sub_Sel => Sub_Sel_in,
        S => Out_Add_Sub,
        Zero => Zero_Add,
        Overflow => Overflow_Add
      );

    Multiplier_2bit_0: Multiplier_2bit
        port map(
        A => A(1 downto 0),
        B => B(1 downto 0),
        Y => Out_Multiplier,
        Zero => Zero_Mul
    );

    MUX_2_to_1_4_0: MUX_2_to_1_4
          port map (
              D0 => Out_Add_Sub,
              D1 => Out_Multiplier,
              S => Math_Sel,      -- Select Add/Sub or Multiply
              Y => Mux_Out
          );


      S <= Mux_Out;
      Zero_out <= Zero_Add OR Zero_Mul;
      Overflow_out <= Overflow_Add;

  end Behavioral;
```

b. Elaborated Diagram

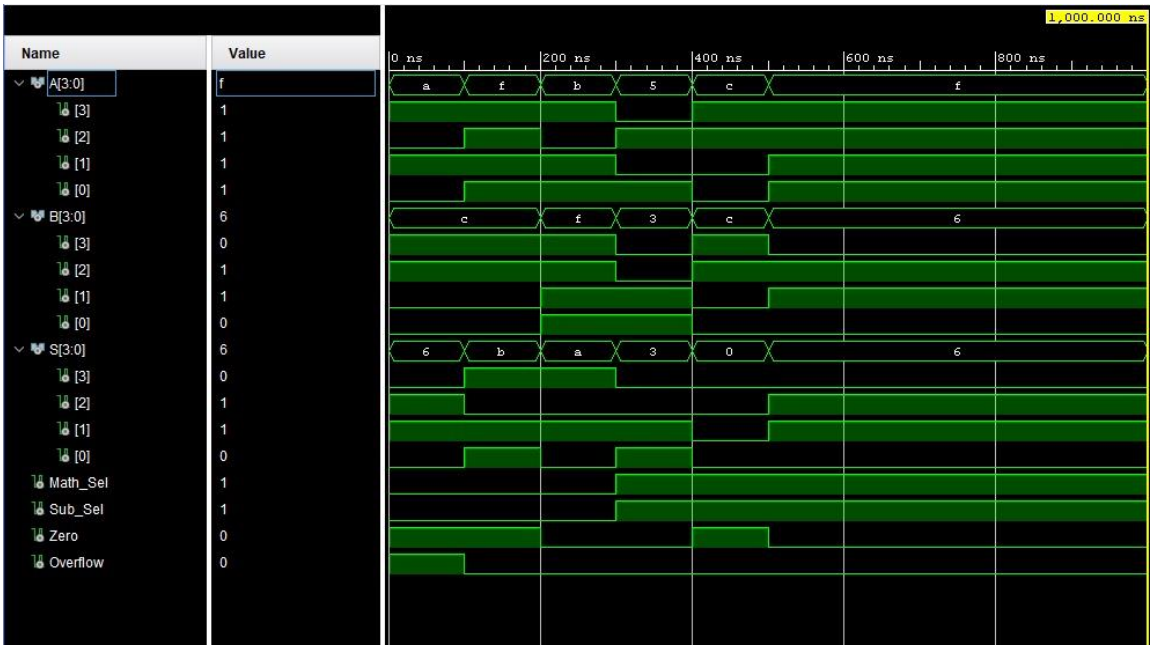


c. Simulation source file

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

entity Add_Sub_Mul_Sim is
--  Port ( );
end Add_Sub_Mul_Sim;

architecture Behavioral of Add_Sub_Mul_Sim is

component Add_Sub_Mul_Unit
    Port (   A : in STD_LOGIC_VECTOR (3 downto 0);
             B : in STD_LOGIC_VECTOR (3 downto 0);
             Math_Sel : in STD_LOGIC;
             Sub_Sel_in : in STD_LOGIC;
             S : out STD_LOGIC_VECTOR (3 downto 0);
             Zero_out : out std_logic;
             Overflow_out :out STD_LOGIC);
end component;

signal A, B, S : std_logic_vector (3 downto 0);
signal Math_Sel, Sub_Sel, Zero, Overflow : std_logic;

begin
    UUT: Add_Sub_Mul_Unit
        Port map( A => A(3 downto 0),
                  B => B(3 downto 0),
                  S => S(3 downto 0),
                  Math_Sel => Math_Sel,
                  Sub_Sel_in => Sub_sel,
```

```
    process
    begin
         --220618A  0011 0101 1101 1100 1010
         --220623J  0011 0101 1101 1100 1111
         --220411H  0011 0101 1100 1111 1011
         --220614H  0011 0101 1101 1100 0110

          Math_Sel <= '0'; -- Select addition/subtraction initially
          Sub_Sel <= '0';  -- Set for addition
          A <= "1010";
          B <= "1100";
          wait for 100ns;
          A <= "1111";
          B <= "1100";
          wait for 100ns;
          A <= "1011";
          B <= "1111";
          wait for 100ns;

          Math_Sel <= '1'; -- Select multiplication
          Sub_Sel <= '1';  -- Set for subtraction
          A <= "0101";
          B <= "0011";
          wait for 100ns;
          A <= "1100";
          B <= "1100";
          wait for 100ns;
          A <= "1111";
          B <= "0110";
          wait;
     end process;
end behavioral;
```

d. Timing diagram

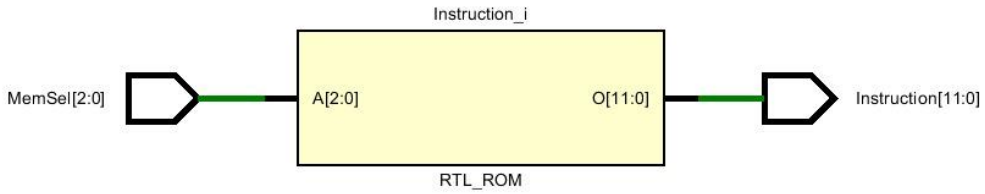


Z. ROM_2

a. Design Source File

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

entity ROM_2 is
    Port ( MemSel : in STD_LOGIC_VECTOR (2 downto 0);
           Instruction : out STD_LOGIC_VECTOR (11 downto 0));
end ROM_2;

architecture Behavioral of ROM_2 is

type rom_type is array (0 to 3) of std_logic_vector(11 downto 0);

signal ROM_2 : rom_type :=(
         --Using new instruction MUL Ra, Rb  =  Ra <- Ra*Rb
         -- Formatting instruction as 00RaRaRaRbRbRb0001
         --Since this is a 4 bit processor it can only give 4 bit output
         --Therefore 2x2 multiplier is used

         --Another program to multiply 2 and 3
            "101110000011", -- MOVI R7,3 --line->0
            "100100000010", -- MOVI R2,2
            "001110100001", -- MUL R7,R2
            "110000000011" -- JZR R0,3  --line->3  --End instructions
           );

begin
    Instruction <= ROM_2(to_integer(unsigned(MemSel)));

end Behavioral;
```

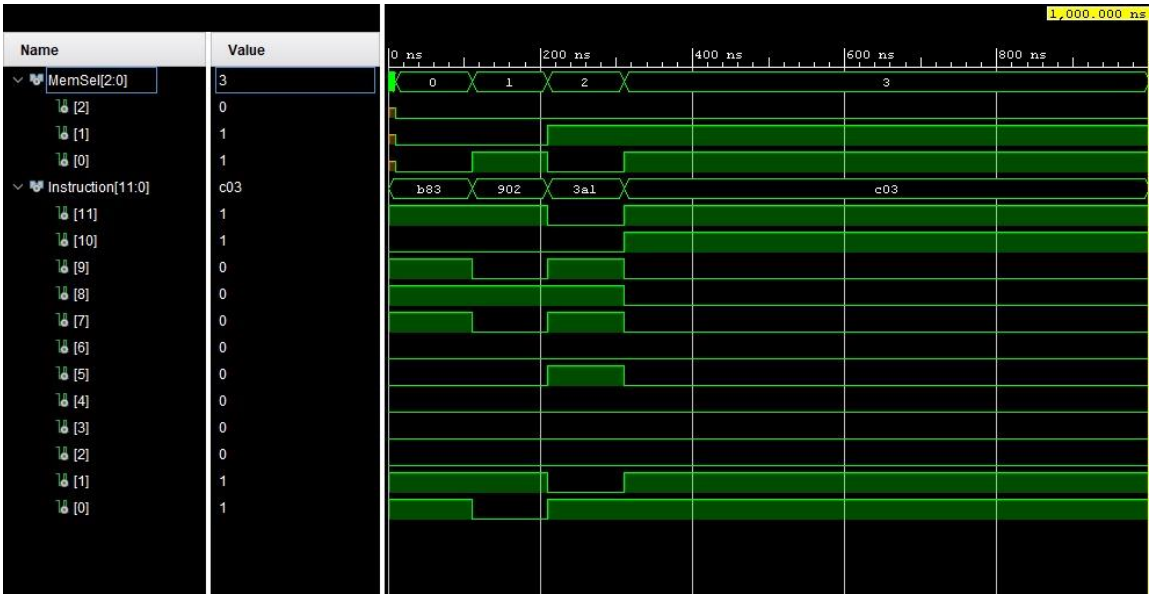
b. Elaborated Diagram



c. Simulation source file

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

entity ROM_2_Sim is
--  Port ( );
end ROM_2_Sim;

architecture Behavioral of ROM_2_Sim is

component ROM_2
    port(
       MemSel : in STD_LOGIC_VECTOR (2 downto 0);
       Instruction : out STD_LOGIC_VECTOR (11 downto 0)
       );
   end component;

Signal MemSel : STD_LOGIC_VECTOR(2 downto 0);
Signal Instruction : STD_LOGIC_VECTOR(11 downto 0);

begin
    UUT : ROM_2 port map(
    MemSel => MemSel,
    Instruction => Instruction
    );

    process
    begin
        wait for 10ns;
        MemSel <= "000";
        wait for 100ns;
        MemSel <= "001";
        wait for 100ns;
        MemSel <= "010";
        wait for 100ns;
        MemSel <= "011";

        wait;
    end process;

end Behavioral;
```

d. Timing diagram



AA. Featured nano

a. Design Source File

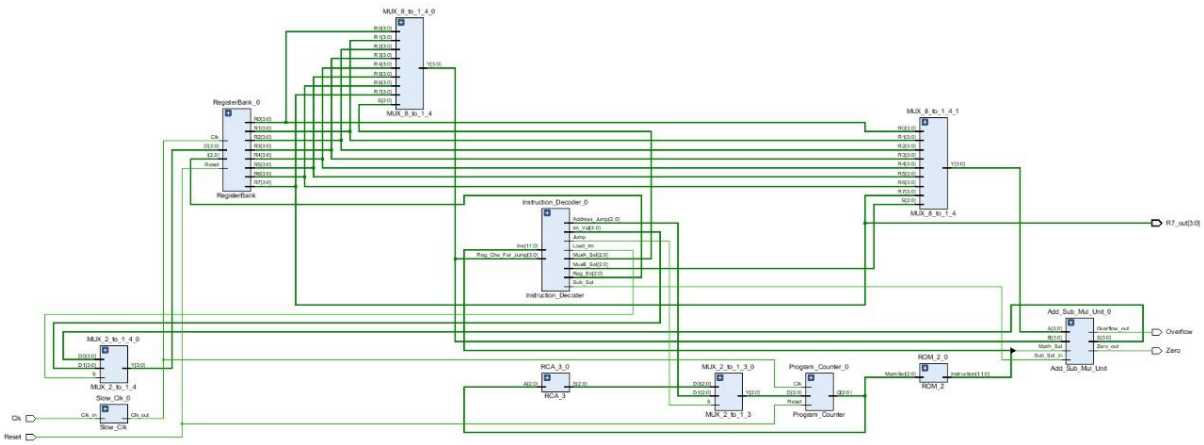
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Featured_Nano_processor is
 Port (  Clk : in STD_LOGIC;
         Reset : in STD_LOGIC;
         Overflow : out STD_LOGIC;
         Zero : out STD_LOGIC;
         R7_out : out STD_LOGIC_VECTOR (3 downto 0));
end Featured_Nano_processor;

architecture Behavioral of Featured_Nano_processor is

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

signal SlowClk:std_logic;

component MUX_8_to_1_4
         Port ( R0 : in STD_LOGIC_VECTOR (3 downto 0);
         R1 : in STD_LOGIC_VECTOR (3 downto 0);
         R2 : in STD_LOGIC_VECTOR (3 downto 0);
         R3 : in STD_LOGIC_VECTOR (3 downto 0);
         R4 : in STD_LOGIC_VECTOR (3 downto 0);
         R5 : in STD_LOGIC_VECTOR (3 downto 0);
         R6 : in STD_LOGIC_VECTOR (3 downto 0);
         R7 : in STD_LOGIC_VECTOR (3 downto 0);
         S : in STD_LOGIC_VECTOR (2 downto 0);
         Y : out STD_LOGIC_VECTOR (3 downto 0));
end component;

component RegisterBank
         Port ( Clk : in STD_LOGIC;
         Reset : in STD_LOGIC;
         D : in STD_LOGIC_VECTOR (3 downto 0);
         R0 : out STD_LOGIC_VECTOR (3 downto 0);
         R1 : out STD_LOGIC_VECTOR (3 downto 0);
         R2 : out STD_LOGIC_VECTOR (3 downto 0);
         R3 : out STD_LOGIC_VECTOR (3 downto 0);
         R4 : out STD_LOGIC_VECTOR (3 downto 0);
         R5 : out STD_LOGIC_VECTOR (3 downto 0);
         R6 : out STD_LOGIC_VECTOR (3 downto 0);
         R7 : out STD_LOGIC_VECTOR (3 downto 0);
         I : in STD_LOGIC_VECTOR (2 downto 0));
end component;
```

```
component Add_Sub_Mul_Unit
        Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
               B : in STD_LOGIC_VECTOR (3 downto 0);
               Math_Sel : in STD_LOGIC;
               Sub_Sel_in : in STD_LOGIC;          --only for Add_Sub_4bit
               S : out STD_LOGIC_VECTOR (3 downto 0);
               Zero_out : out std_logic;
               Overflow_out :out STD_LOGIC);
   end component;

 component RCA_3
    Port ( A : in STD_LOGIC_VECTOR (2 downto 0);
    S: out STD_LOGIC_VECTOR (2 downto 0);
    C_out : out STD_LOGIC);
   end component;
component Instruction_Decoder
       Port (
       Ins : in STD_LOGIC_VECTOR (11 downto 0);
       Reg_Che_For_Jump : in STD_LOGIC_VECTOR (3 downto 0);
       Reg_En : out STD_LOGIC_VECTOR(2 downto 0);
       Load_Im : out STD_LOGIC;
       Im_Val : out STD_LOGIC_VECTOR(3 downto 0);
       MuxA_Sel : out STD_LOGIC_VECTOR (2 downto 0);
       MuxB_Sel : out STD_LOGIC_VECTOR (2 downto 0);
       Sub_Sel : out STD_LOGIC;
       Jump : out STD_LOGIC;
       Address_Jump : out STD_LOGIC_VECTOR (2 downto 0));
end component;
   component MUX_2_to_1_3
    Port ( D0 : in STD_LOGIC_VECTOR (2 downto 0);
    D1 : in STD_LOGIC_VECTOR (2 downto 0);
    S : in STD_LOGIC;
    Y : out STD_LOGIC_VECTOR (2 downto 0));
   end component;
   component MUX_2_to_1_4
    Port ( D0 : in STD_LOGIC_VECTOR(3 downto 0);
    D1 : in STD_LOGIC_VECTOR(3 downto 0);
    S : in STD_LOGIC;
    Y : out STD_LOGIC_VECTOR(3 downto 0));
   end component;

   component ROM_2
    Port ( MemSel : in STD_LOGIC_VECTOR (2 downto 0);
    Instruction : out STD_LOGIC_VECTOR (11 downto 0));
   end component;

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

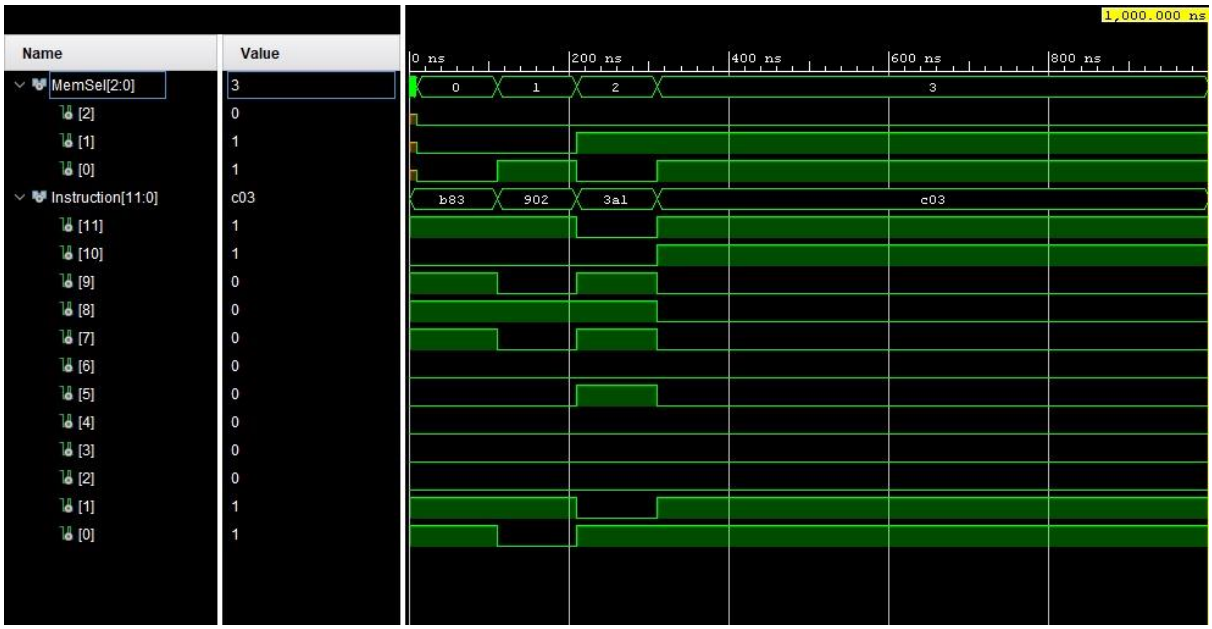
```
signal Ins0:std_logic_vector(11 downto 0);
signal PC_out,J_A,A_3,M0_S,M1_S,R_E:std_logic_vector(2 downto 0);
signal I_V,M3,M1,M0,A_4:std_logic_vector(3 downto 0);
signal R0,R1,R2,R3,R4,R5,R6,R7:std_logic_vector(3 downto 0);
signal J, L_S, Sub :std_logic;
signal M2:std_logic_vector(2 downto 0);

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

Program_Counter_0:Program_Counter
     port map(
     Clk=>SlowClk,
     Reset=>Reset,
     D=>M2,
     Q=>PC_out);

ROM_2_0 : ROM_2
     port map(
     MemSel=>PC_out,
     Instruction=>Ins0);

MUX_2_to_1_4_0:MUX_2_to_1_4
 port map(
 D0=>A_4,
 D1=>I_V,
 S=>L_S,
 Y=>M3);

MUX_2_to_1_3_0:MUX_2_to_1_3
 port map(
 D0=>A_3,
 D1=>J_A,
 S=>J,
 Y=>M2);

MUX_8_to_1_4_0:MUX_8_to_1_4
 port map(
 R0=>R0,
 R1=>R1,
 R2=>R2,
 R3=>R3,
 R4=>R4,
 R5=>R5,
 R6=>R6,
 R7=>R7,
 Y=>M0,
 S=>M0_S);
```

```
    RegisterBank_0:RegisterBank
     port map(
     Clk=>SlowClk,
     Reset=>Reset,
     D=>M3,
     R0=>R0,
     R1=>R1,
     R2=>R2,
     R3=>R3,
     R4=>R4,
     R5=>R5,
     R6=>R6,
     R7=>R7,
     I=>R_E);

    Add_Sub_Mul_Unit_0 : Add_Sub_Mul_Unit
        port map(
         A => M1,
         B => M0,
         Math_Sel => Ins0(0),
         Sub_Sel_in => Sub ,
         S => A_4,
         Zero_out => Zero,
         Overflow_out => Overflow );

    RCA_3_0:RCA_3
     port map(
     A=>PC_out,
     S=>A_3);
    Instruction_Decoder_0:Instruction_Decoder
    Port map (
     Ins =>Ins0,
     Reg_Che_For_Jump =>M0,
     Reg_En =>R_E,
     Load_Im =>L_S,
     Im_Val =>I_V,
     MuxA_Sel =>M0_S,
     MuxB_Sel =>M1_S,
     Sub_Sel =>Sub,
     Jump =>J,
     Address_Jump =>J_A);

  R7_out<=R7;
end behavioral;
```

b. Elaborated Diagram



c. Timing diagram

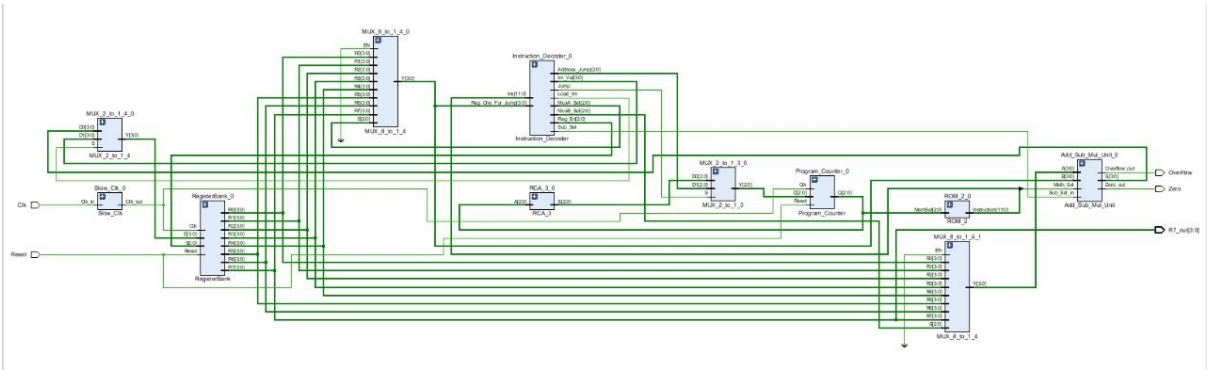


BB. Featured_nano with 7seg

a. Design Source File

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

entity Featured_Nano_Processor_with_7Seg is
 Port ( Clk : in STD_LOGIC;
 Reset : in STD_LOGIC;
 Overflow : out STD_LOGIC;
 Zero : out STD_LOGIC;
 LED_out : out STD_LOGIC_VECTOR (3 downto 0);
 Anode_7Seg : out STD_LOGIC_VECTOR (3 downto 0);
 Cathode_7Seg : out STD_LOGIC_VECTOR (6 downto 0));
end Featured_Nano_Processor_with_7Seg;

architecture Behavioral of Featured_Nano_Processor_with_7Seg is

 component Featured_Nano_processor
 Port ( Clk : in STD_LOGIC;
 Reset : in STD_LOGIC;
 Overflow : out STD_LOGIC;
 Zero : out STD_LOGIC;
 R7_out : out STD_LOGIC_VECTOR (3 downto 0));
 end component;

 component Out_controll_7seg_final
 Port ( Clk : in STD_LOGIC;
 R7_out : in STD_LOGIC_VECTOR (3 downto 0);
 Anode_out : out STD_LOGIC_VECTOR (3 downto 0);
 Cathode_out : out STD_LOGIC_VECTOR (6 downto 0));
 end component;

 component ResetController
 Port ( ResIn : in STD_LOGIC;
 Clk : in STD_Logic;
 ResOut : out STD_LOGIC);
 end component;

 signal R7_output:std_logic_vector(3 downto 0);
 signal ResINPUT:std_logic;
begin
ResetController_0:ResetController
 port map(
 ResIn=>Reset,
 Clk=>Clk,
  ResOut=>ResINPUT );

 Featured_Nano_processor_0 : Featured_Nano_processor
  port map(
  Clk=>Clk,
  Reset=>ResINPUT,
  Overflow=>Overflow,
  Zero=>Zero,
  R7_out=>R7_output );


 Out_controll_7seg_final_0:Out_controll_7seg_final
  Port map (
  Clk =>Clk,
  R7_out=>R7_output,
  Anode_out=>Anode_7Seg,
  Cathode_out=>Cathode_7Seg);

 end Behavioral;
```

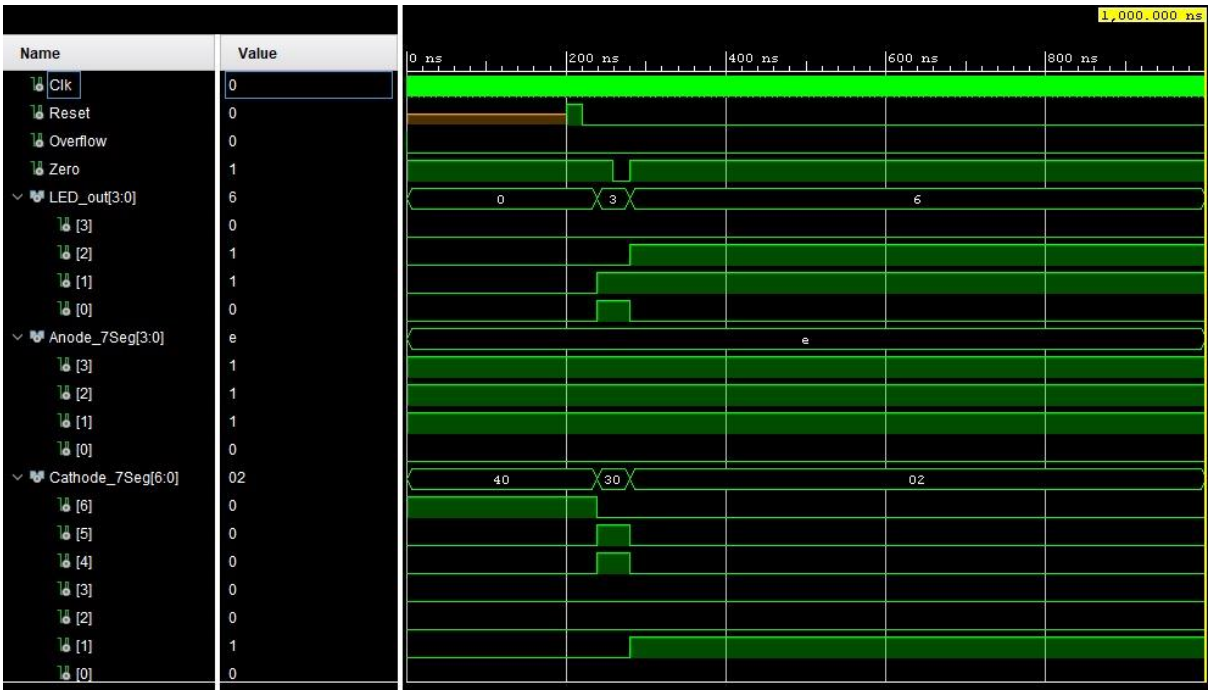
b. Elaborated Diagram



c. Simulation source file

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

entity TB_Featured_Nano_Processor_with_7Seg is
--  Port ( );
end TB_Featured_Nano_Processor_with_7Seg;

architecture Behavioral of TB_Featured_Nano_Processor_with_7Seg is
component Featured_Nano_Processor_with_7Seg
    Port ( Clk : in STD_LOGIC;
    Reset : in STD_LOGIC;
    Overflow : out STD_LOGIC;
    Zero : out STD_LOGIC;
    LED_out : out STD_LOGIC_VECTOR (3 downto 0);
    Anode_7Seg : out STD_LOGIC_VECTOR (3 downto 0);
    Cathode_7Seg : out STD_LOGIC_VECTOR (6 downto 0));
end component;

signal Clk : std_logic := '0';
signal Reset : std_logic;
signal Overflow,Zero: std_logic;
signal LED_out,Anode_7Seg : STD_LOGIC_VECTOR (3 downto 0);
signal Cathode_7Seg :STD_LOGIC_VECTOR (6 downto 0);

begin
    UUT: Featured_Nano_Processor_with_7Seg
    port map(
    Clk=>Clk,
    Reset=>Reset,
    OverFlow=>Overflow,
    Zero=>Zero,
    LED_out => LED_out,
    Anode_7Seg => Anode_7Seg,
    Cathode_7Seg => Cathode_7Seg);

process
begin
    wait for 1ns;
    Clk<=not(Clk);
    end process;
    process
    begin
    wait for 200ns;
    Reset<='1';
    wait for 20ns;
    Reset<='0';
    wait for 1500ns;
    wait for 200ns;
    Reset<='1';
    wait for 10ns;
    Reset<='0';
    wait;
end process;
```

d. Timing diagram

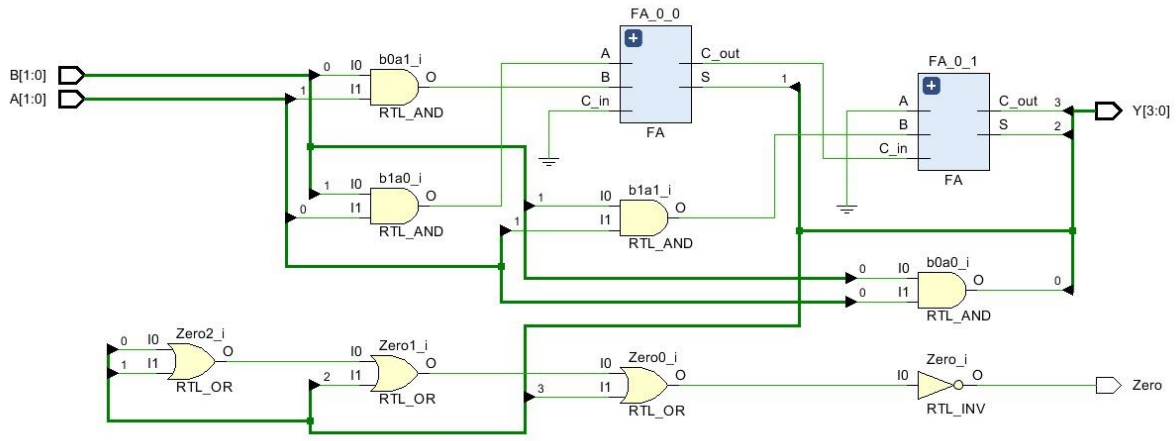


CC. Multiplier - 2bit

a. Design Source File

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity Multiplier_2bit is
     Port ( A : in STD_LOGIC_VECTOR (1 downto 0);
            B : in STD_LOGIC_VECTOR (1 downto 0);
            Y : out STD_LOGIC_VECTOR (3 downto 0);
            Zero : out STD_LOGIC);
end Multiplier_2bit;

architecture Behavioral of Multiplier_2bit is

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

SIGNAL b0a0, b0a1, b1a0, b1a1 :std_logic;
SIGNAL s_0_0, s_0_1, c_0_0, c_0_1 :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 => '0',
         B => b1a1,
         C_in => c_0_0,
         S => s_0_1,
         C_out => c_0_1 );

b0a0 <= B(0) and A(0);
b0a1 <= B(0) and A(1);
b1a0 <= B(1) and A(0);
b1a1 <= B(1) and A(1);

Y(0) <= b0a0 ;
Y(1) <= s_0_0 ;
Y(2) <= s_0_1 ;
Y(3) <= c_0_1 ;

Zero <= not(b0a0 or s_0_0 or s_0_1 or c_0_1);
end Behavioral;
```

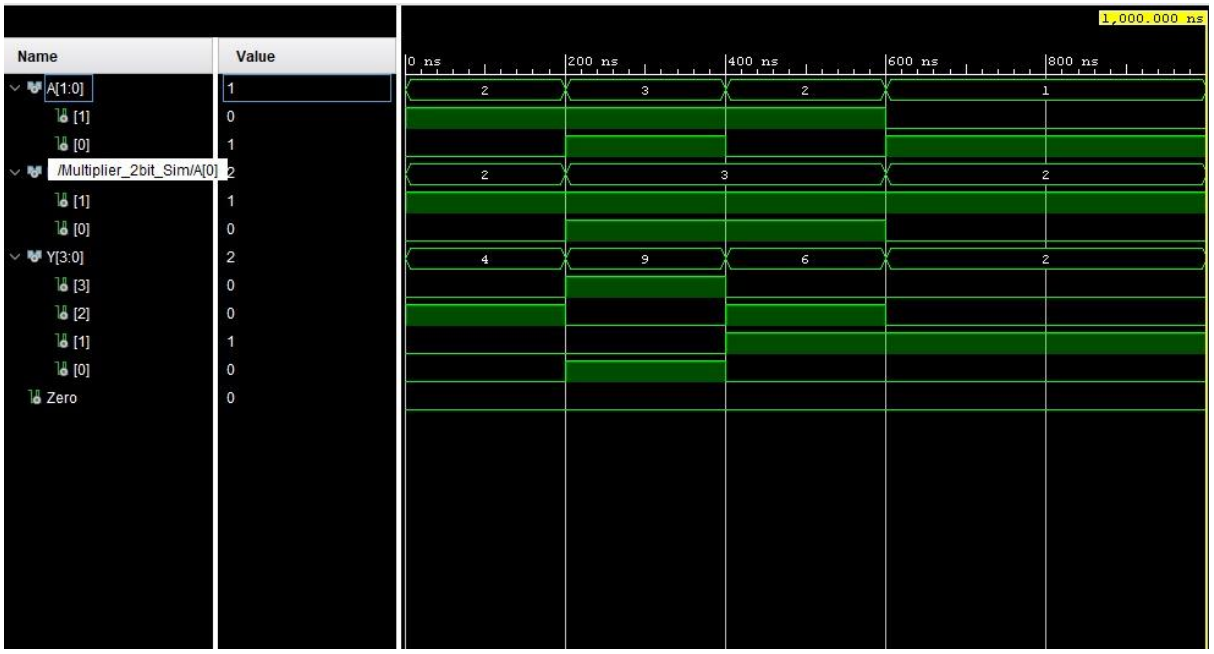
b. Elaborated Diagram



c. Simulation source file

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

entity Multiplier_2bit_Sim is
--  Port ( );
end Multiplier_2bit_Sim;

architecture Behavioral of Multiplier_2bit_Sim is

component Multiplier_2bit
    Port( A : in std_logic_vector(1 downto 0);
          B : in std_logic_vector(1 downto 0);
          Y : out std_logic_vector(3 downto 0);
          Zero : out STD_LOGIC);
end component;

signal A : std_logic_vector(1 downto 0);
signal B : std_logic_vector(1 downto 0);
signal Y : std_logic_vector(3 downto 0);
signal Zero : STD_LOGIC;

begin
    UUT: Multiplier_2bit
        Port map(
            A => A,
            B => B,
            Y => Y,
            Zero => Zero);

    process
    begin
         --220618A  11 0101 1101 1100 1010
         --220623J  11 0101 1101 1100 1111
         --220411H  11 0101 1100 1111 1011
         --220614H  11 0101 1101 1100 0110

        A <= "10";
        B <= "10";
        Wait for 200ns;
        A <= "11";
        B <= "11";
        Wait for 200ns;
        A <= "10";
        B <= "11";
        Wait for 200ns;
        A <= "01";
        B <= "10";
        Wait;
    end process;
end Behavioral;
```

d. Timing diagram



Problems we Faced and Solutions:

- **Problems:**
  - Negative flags can introduce confusion and make code harder to understand, leading to potential errors in program logic. Selecting the appropriate ROM can be challenging, especially considering factors like compatibility, performance, and size constraints.
  - Poorly chosen variable names can make code difficult to maintain and comprehend, reducing overall readability and increasing the likelihood of bugs.
  - The project encountered a bottleneck due to an excessive number of hard drives, impeding the implementation process.
  - The project encountered instances of "undefined" values within simulation files.We are posing challenges to the verification and testing processes.

- **Solutions:**
  - We instead don't use negative flags, and we opt for descriptive variable names that clearly convey our purpose as a solution.
  - We prioritize thorough research and testing when selecting ROMs. Consider factors such as compatibility with hardware and the specific requirements of our application. Document our decision-making process to facilitate future updates or modifications.
  - We invest time in selecting meaningful and descriptive variable names that accurately reflect our purpose and usage within the codebase. We follow established naming conventions to promote consistency and clarity across the project.
  - We adopted a approach aimed at identifying and resolving the root causes of the undefined values in the simulation files. By enforcing robust coding practices and diligently addressing any discrepancies discovered during testing, We successfully eliminated the presence of undefined values in the simulation files.

Conclusion

From this lab, we managed to create a 4-bit arithmetic unit that can handle adding and subtracting signed numbers. We faced some challenges when putting together the different parts, mainly because everyone had their own ideas about how things should work. But after talking it out a lot as a group, we got on the same page.

Getting all the components to work together in the Nano Processor was tough at first. We had issues like weird numbers showing up on the board, but we fixed that by adding something called a Reset controller. It stops the board from showing weird numbers until everything's ready to go. We also had trouble showing negative numbers on the display, but we figured it out after a bunch of group talks.

Using buses helped a lot with dealing with all the different connections. It made things easier to manage in the software we were using. And we tried to keep things simple in our designs, using basic parts whenever we could to make more complicated stuff.

Working together on this project really helped us get better at communicating and sharing the workload. We learned a lot about teamwork and making things work together smoothly.