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

Student Names and Index Numbers:

- Sajeenthiran P. : 210553J
- G.D.C. Samaranayake :210557B

Lab Task

Design a simple nano processor capable of executing the following assembly instructions.

1. ADD(Addition) register B to register A
2. MOV(Move) value d to register A
3. NEG(Negation) 2's complement of register R
4. JZR(Jump) Jump to register A if the value is 0 in register B

Build the nano processor using the following components.

- Register Bank
- ROM
- 3-bit Program Counter (PC)
- Instruction Decoder
- 3 bit Adder
- 4-bit Add/Subtract unit
- 2-way 3-bit multiplexer
- 2-way 4-bit multiplexer
- 8-way 4-bit multiplexer
- Slow clock
- Lookup Table

• Write an assembly program to calculate the total between 1 to 3.

• Simulate addition of 1,2 and 3 and load it to Reg7

• Test on BASYS 3 and verify the functionality of the nano processor.

• Demonstrate the circuit to the instructor

Assembly program and Machine code representation

MOVI R7, 3 -: 101110000011
MOVI R1, 3 -: 100010000011
MOVI R2, 1 -: 100100000001
NEG R2 -: 010100000000
ADD R1, R2 -: 000010100000
ADD R7, R1 -: 001110010000
JZR R1, 6 -: 110010000110
JZR R0, 4 -: 110000000100

Optimized assembly program

MOVI R7, 3 :- 101110000011
MOVI R1, 1 :- 100010000001
MOVI R2, 2 :- 100100000010
ADD R7, R2 :- 001110100000
ADD R7, R1 :- 001110010000

NanoProcessor

Design sources

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/23/2023 02:32:24 PM
-- Design Name:
-- Module Name: Microprocessor - 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 Microprocessor is
    Port ( Reset : in STD_LOGIC;
           Clk : in STD_LOGIC;
           Led_7_Seg : out STD_LOGIC_VECTOR (6 downto 0);
           AN : out STD_LOGIC_VECTOR (3 downto 0);
           Overflow : out STD_LOGIC;
```

```
        Zero : out STD_LOGIC;
        Reg_7_output : out STD_LOGIC_VECTOR(3 downto 0));

end Microprocessor;

architecture Behavioral of Microprocessor is

component Reg_Bank
    Port ( I : in STD_LOGIC_VECTOR (3 downto 0);
        RegEn : in STD_LOGIC_VECTOR (2 downto 0);
        Clk : in STD_LOGIC;
        Reset : in STD_LOGIC;
        Reg0 : out STD_LOGIC_VECTOR (3 downto 0);
        Reg1 : out STD_LOGIC_VECTOR (3 downto 0);
        Reg2 : out STD_LOGIC_VECTOR (3 downto 0);
        Reg3 : out STD_LOGIC_VECTOR (3 downto 0);
        Reg4 : out STD_LOGIC_VECTOR (3 downto 0);
        Reg5 : out STD_LOGIC_VECTOR (3 downto 0);
        Reg6 : out STD_LOGIC_VECTOR (3 downto 0);
        Reg7 : out STD_LOGIC_VECTOR (3 downto 0));
end component;

component ROM
    Port ( address : in STD_LOGIC_VECTOR (2 downto 0);
        instruction : out STD_LOGIC_VECTOR (11 downto 0));
end component;

component Instruction_Decoder
    Port ( instruction : in STD_LOGIC_VECTOR (11 downto 0);
        register_check_for_jump : in STD_LOGIC_VECTOR (3 downto 0);
        load_select : out STD_LOGIC;
        immediate_value : out STD_LOGIC_VECTOR (3 downto 0);
        register_enable : out STD_LOGIC_VECTOR (2 downto 0);
        register_select_mux_1 : out STD_LOGIC_VECTOR (2 downto 0);
        register_select_mux_2 : out STD_LOGIC_VECTOR (2 downto 0);
        add_or_subtract : out STD_LOGIC;
        jump_flag : out STD_LOGIC;
        address_to_jump : out STD_LOGIC_VECTOR (2 downto 0));
end component;

component Program_Counter
    Port ( Res : in STD_LOGIC;
        Clk : in STD_LOGIC;
        D_IN : in STD_LOGIC_VECTOR (2 downto 0);
```

```
        Q : out STD_LOGIC_VECTOR (2 downto 0));
end component;



component Add_Sub_4bit
    Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
           B : in STD_LOGIC_VECTOR (3 downto 0);
           Cin : in STD_LOGIC;
           M : in STD_LOGIC;
           S : out STD_LOGIC_VECTOR (3 downto 0);
           Cout : out STD_LOGIC;
           Zero : out STD_LOGIC);
end component;

component Add_3bit
    Port ( A : in STD_LOGIC_VECTOR (2 downto 0);
           B : in STD_LOGIC_VECTOR (2 downto 0);
           Cin : in STD_LOGIC;
           S : out STD_LOGIC_VECTOR (2 downto 0);
           Cout : out STD_LOGIC);
end component;

component MUX_8_to_1_4bit
    Port ( D0 : in STD_LOGIC_VECTOR (3 downto 0);
           D1 : in STD_LOGIC_VECTOR (3 downto 0);
           D2 : in STD_LOGIC_VECTOR (3 downto 0);
           D3 : in STD_LOGIC_VECTOR (3 downto 0);
           D4 : in STD_LOGIC_VECTOR (3 downto 0);
           D5 : in STD_LOGIC_VECTOR (3 downto 0);
           D6 : in STD_LOGIC_VECTOR (3 downto 0);
           D7 : in STD_LOGIC_VECTOR (3 downto 0);
           S : in STD_LOGIC_VECTOR (2 downto 0);
           Y : out STD_LOGIC_VECTOR (3 downto 0));
end component;

component MUX_2_to_1_4bit
    Port ( load_select : in STD_LOGIC;
           D0 : in STD_LOGIC_VECTOR (3 downto 0);
           D1 : in STD_LOGIC_VECTOR (3 downto 0);
           Y : out STD_LOGIC_VECTOR (3 downto 0));

end component;
```

```
component MUX_2_to_1_3bit
    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 Slow_Clk
        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;

signal data: STD_LOGIC_VECTOR (6 downto 0);
signal
Reg_In,Reg0,Reg1,Reg2,Reg3,Reg4,Reg5,Reg6,Reg7,immediate_value,add_sub_output,Y0,Y
1,address: STD_LOGIC_VECTOR (3 downto 0);
signal RegEn,memory_select,Program_Counter_In,address_to_jump,Add_3bit_output,A,B :
STD_LOGIC_VECTOR (2 downto 0);
signal instruction : STD_LOGIC_VECTOR (11 downto 0);
signal load_select,add_or_subtract,jump_flag,Cout,slow_clock : STD_LOGIC;


begin

Reg_Bank_0 : Reg_Bank
port map(
I => Reg_In,
RegEn => RegEn,
Clk => slow_clock,
Reset => Reset,
Reg0 => Reg0,
Reg1 => Reg1,
Reg2 => Reg2,
Reg3 => Reg3,
Reg4 => Reg4,
Reg5 => Reg5,
Reg6 => Reg6,
Reg7 => Reg7
);
```

```
ROM_0 : ROM
port map(
address => memory_select,
instruction => instruction
);

Program_Counter_0 : Program_Counter
port map(
Res => Reset,
Clk => slow_clock,
D_IN => Program_Counter_In,
Q => memory_select
);

Instruction_Decoder_0 : Instruction_Decoder
port map(
instruction => instruction,
register_check_for_jump => Y0,
load_select => load_select,
immediate_value => immediate_value,
register_enable => RegEn,
register_select_mux_1 => A,
register_select_mux_2 => B,
add_or_subtract => add_or_subtract,
jump_flag => jump_flag,
address_to_jump => address_to_jump

);

Add_3bit_0 : Add_3bit
port map(
A => memory_select,
B => "001",
Cin => '0',
S => Add_3bit_output,
Cout => Cout
);

MUX_2_to_1_3bit_0 : MUX_2_to_1_3bit
port map(
D0 => Add_3bit_output,
D1 => address_to_jump,
S => jump_flag,
```

```
Y => Program_Counter_In
);

MUX_2_TO_1_4bit_0: MUX_2_TO_1_4bit
port map(
load_select => load_select,
D0 => immediate_value,
D1 => add_sub_output,
Y => Reg_In
);

Add_Sub_4bit_0: Add_Sub_4bit
port map(
A => Y0,
B => Y1,
Cin => add_or_subtract,
M => add_or_subtract,
S => add_sub_output,
Cout => Overflow,
Zero => Zero

);

MUX_8_to_1_4bit_0 : MUX_8_to_1_4bit
port map(
D0 => Reg0,
D1 => Reg1,
D2 => Reg2,
D3 => Reg3,
D4 => Reg4,
D5 => Reg5,
D6 => Reg6,
D7 => Reg7,
S => A,
Y => Y0

);

MUX_8_to_1_4bit_1 : MUX_8_to_1_4bit
port map(
D0 => Reg0,
D1 => Reg1,
D2 => Reg2,
D3 => Reg3,
```

```
D4 => Reg4,
D5 => Reg5,
D6 => Reg6,
D7 => Reg7,
S => B,
Y => Y1
);

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

LUT_16_7_0 : LUT_16_7
            Port map(
               address => Reg7,
               data => Led_7_Seg
            );

AN <= "1110";
Reg_7_output <= Reg7;

end Behavioral;
```

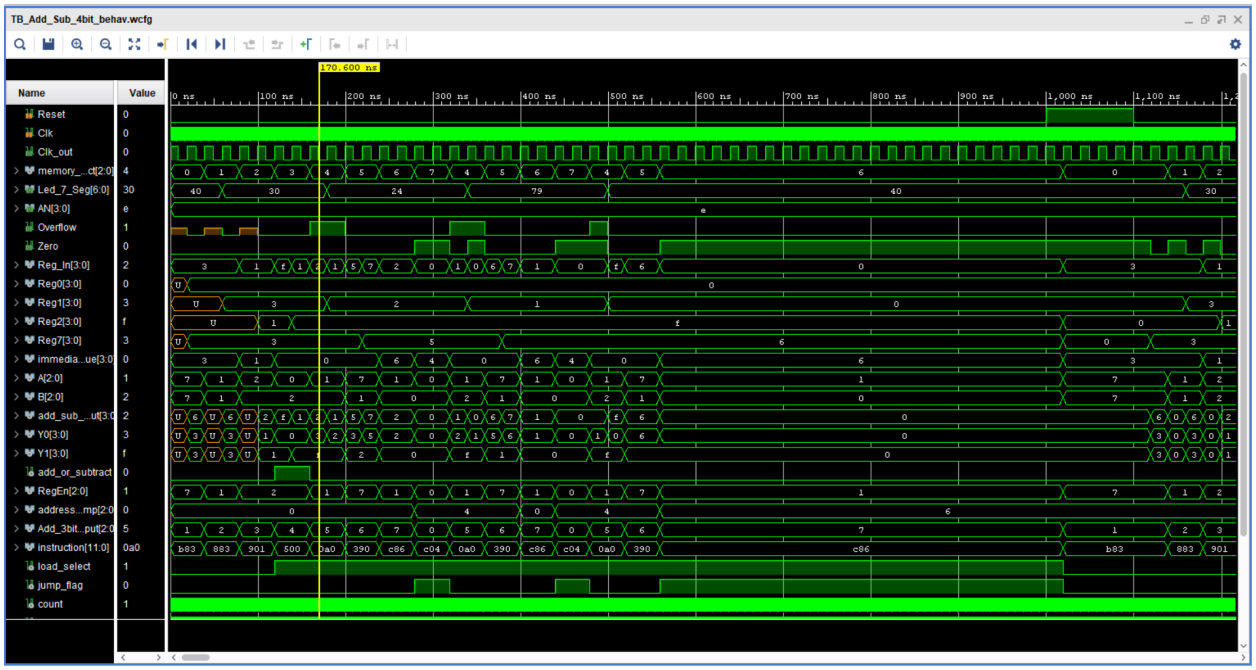
Test Bench

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/23/2023 04:07:42 PM
-- Design Name:
-- Module Name: TB_Microprocessor - 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_Microprocessor is
--  Port ( );
end TB_Microprocessor;

architecture Behavioral of TB_Microprocessor is

component Microprocessor
    Port ( Reset : in STD_LOGIC;
           Clk : in STD_LOGIC;
           Led_7_Seg : out STD_LOGIC_VECTOR (6 downto 0);
           AN : out STD_LOGIC_VECTOR (3 downto 0);
           Overflow : out STD_LOGIC;
           Zero : out STD_LOGIC);
end component;

signal Reset,Clk,Overflow,Zero : STD_LOGIC;
signal AN : STD_LOGIC_VECTOR (3 downto 0);
signal Led_7_Seg : STD_LOGIC_VECTOR (6 downto 0);

begin
UUT : Microprocessor
port map(
Reset => Reset,
Clk => Clk,
Led_7_Seg => Led_7_Seg,
AN => AN,
Overflow => Overflow,
Zero => Zero
```

```
);
Clock : process
   begin
      Clk <='0';
         wait for 1ns;
         Clk <='1';
         wait for 1ns;

         Clk <='1';
end process;

process
   begin
      Reset<='0';
      wait for 1000ns;
      Reset<='1';
      wait for 100ns;
      Reset<='0';
      wait;

end process;
end Behavioral;
```

Timing Diagram



Design Sources

Register Bank

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/18/2023 11:16:11 PM
-- Design Name:
-- Module Name: Reg_Bank - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity Reg_Bank is
    Port ( I : in STD_LOGIC_VECTOR (3 downto 0);
           RegEn : in STD_LOGIC_VECTOR (2 downto 0);
           Clk : in STD_LOGIC;
           Reset : in STD_LOGIC;
           Reg0 : out STD_LOGIC_VECTOR (3 downto 0);
           Reg1 : out STD_LOGIC_VECTOR (3 downto 0);
```

```
        Reg2 : out STD_LOGIC_VECTOR (3 downto 0);
        Reg3 : out STD_LOGIC_VECTOR (3 downto 0);
        Reg4 : out STD_LOGIC_VECTOR (3 downto 0);
        Reg5 : out STD_LOGIC_VECTOR (3 downto 0);
        Reg6 : out STD_LOGIC_VECTOR (3 downto 0);
        Reg7 : out STD_LOGIC_VECTOR (3 downto 0));
end Reg_Bank;

architecture Behavioral of Reg_Bank is

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

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

signal RegENBus : STD_LOGIC_VECTOR (7 downto 0);

begin

Decoder_3_8_0 : Decoder_3_to_8
 port map(
 I => RegEn,
 EN => '1',
 Y => RegENBus
 );

 Reg_0 : Reg
 port map(
 D => "0000",
 EN => '1',
 Res => Reset,
 Clk => Clk,
 Q => Reg0
 );
```

```
Reg_1 : Reg
 port map(
 D => I,
 EN => RegENBus(1),
 Res => Reset,
 Clk => Clk,
 Q => Reg1
 );

 Reg_2 : Reg
  port map(
  D => I,
  EN => RegENBus(2),
  Res => Reset,
  Clk => Clk,
  Q => Reg2
  );

  Reg_3 : Reg
   port map(
   D => I,
   EN => RegENBus(3),
   Res => Reset,
   Clk => Clk,
   Q => Reg3
   );

   Reg_4 : Reg
    port map(
    D => I,
    EN => RegENBus(4),
    Res => Reset,
    Clk => Clk,
    Q => Reg4
    );

    Reg_5 : Reg
     port map(
     D => I,
     EN => RegENBus(5),
     Res => Reset,
     Clk => Clk,
     Q => Reg5
     );
```

```
      Reg_6 : Reg
       port map(
       D => I,
       EN => RegENBus(6),
       Res => Reset,
       Clk => Clk,
       Q => Reg6
       );

       Reg_7 : Reg
        port map(
        D => I,
        EN => RegENBus(7),
        Res => Reset,
        Clk => Clk,
        Q => Reg7
        );



end Behavioral;
```

Program ROM

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/17/2023 02:54:57 PM
-- Design Name:
-- Module Name: 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 ROM is
    Port ( address : in STD_LOGIC_VECTOR (2 downto 0);
           instruction : out STD_LOGIC_VECTOR (11 downto 0));
end ROM;

architecture Behavioral of ROM is

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

--signal ROM : rom_type := (

--"100010001010", --MOVI R1, 10
--"100100000001", --MOVI R2, 1
--"010100000000", --NEG R2
--"000010100000", --ADD R1, R2
--"110010000111", --JZR R1, 7
--"110000000011"  --JZR R0, 3
--);

signal ROM : rom_type := (
"101110000011", --MOVI R7, 3    -0
"100010000011", --MOVI R1, 3    -1
"100100000001", --MOVI R2, 1    -2
"010100000000", --NEG R2        -3
"000010100000", --ADD R1, R2    -4
"001110010000", --ADD R7, R1    -5
```

```
"110010000110", --JZR R1, 6     -6
"110000000100" --JZR R0, 4      -7
);

--optimzed assembly program
--signal ROM : rom_type := (
--"101110000011", --MOVI R7, 3    -0
--"100010000001", --MOVI R1, 1    -1
--"100100000010", --MOVI R2, 2    -2

--"001110100000", --ADD R7, R2    -4
--"001110010000", --ADD R7, R1    -4
--"000000000000",
--"000000000000",
--"000000000000"

--);

begin
instruction <= ROM(to_integer(unsigned(address)));


end Behavioral;
```

3-bit Program Counter (PC)

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/23/2023 01:53:48 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 ( Res : in STD_LOGIC;
           Clk : in STD_LOGIC;
           D_IN : in STD_LOGIC_VECTOR (2 downto 0);
           Q : out STD_LOGIC_VECTOR (2 downto 0):= "000");
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;




    SIGNAL Q0, Q1, Q2 : STD_LOGIC;
    SIGNAL D0, D1, D2 : STD_LOGIC;
    SIGNAL Clk_slow : STD_LOGIC;  -- INTERNAL CLOCK
    SIGNAL D  : STD_LOGIC_VECTOR (2 downto 0) := "000";
    SIGNAL count : integer :=0;

begin
```

```
--begin
process (Clk) begin
   if(rising_edge(Clk)) then

   if(Res='1')
      then
      Q<="000";
      count<=0;

      else
      if (D_IN="000") then

       else
       if(count=1)then
          count<=0;
          Q<=D_IN;

       else
          count<=1;
       end if;

      end if;
      end if;

   end if ;
end process;

end Behavioral;
```

Instruction Decoder

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/23/2023 01:32:28 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);
           load_select : out STD_LOGIC;
           immediate_value : out STD_LOGIC_VECTOR (3 downto 0);
           register_enable : out STD_LOGIC_VECTOR (2 downto 0);
           register_select_mux_1 : out STD_LOGIC_VECTOR (2 downto 0);
           register_select_mux_2 : out STD_LOGIC_VECTOR (2 downto 0);
           add_or_subtract : out STD_LOGIC;
           jump_flag : out STD_LOGIC;
           address_to_jump : out STD_LOGIC_VECTOR (2 downto 0));
end Instruction_Decoder;

architecture Behavioral of Instruction_Decoder is

begin

process (instruction,register_check_for_jump) is
begin
  if (instruction(11 downto 10 )="10") then
    register_enable<=instruction(9 downto 7 );
    immediate_value<=instruction (3 downto 0);
    load_select<='0'; -- we define this , when load_slect =0 chose intermidiate value ;
    register_select_MUX_1<=instruction(9 downto 7 );--default , no need for this case ,ignore it;
```

```
  register_select_mux_2<=instruction(9 downto 7 );--default , no need for this case ,ignore it;
  add_or_subtract<='0'; --default , no need for this case ,ignore it;
  jump_flag<='0';

elsif ((instruction(11 downto 10 )="00")) then -- add and store
  register_enable<=instruction(9 downto 7 );
  immediate_value<=instruction (3 downto 0);
  load_select<='1'; -- we define this , when load_slect =0 chose intermidiate value ;
  register_select_MUX_1<=instruction(9 downto 7 );--default , no need for this case ,ignore it;
  register_select_mux_2<=instruction(6 downto 4 );--default , no need for this case ,ignore it;
  add_or_subtract<='0'; --select add operation
  jump_flag<='0';

elsif ((instruction(11 downto 10 )="01")) then -- complement case
  register_enable<=instruction(9 downto 7 );-- fetch register address;
  immediate_value<=instruction (3 downto 0);--default , no need for this case ,ignore it;
  load_select<='1'; --default , no need for this case ,ignore it;
  register_select_MUX_1<= "000";--default , no need for this case ,ignore it;
  register_select_mux_2<= instruction( 9 downto 7);--default , no need for this case ,ignore it;
  add_or_subtract<='1'; --default , no need for this case ,ignore it;
  jump_flag<='0';



 else --"11" -- jump case

  register_enable<=instruction(9 downto 7 );

  immediate_value<=instruction (3 downto 0);
  load_select<='1'; --default , no need for this case ,ignore it;
  register_select_MUX_1<=instruction(9 downto 7 );--default , no need for this case ,ignore it;
  register_select_mux_2<="000";--default , no need for this case ,ignore it;
  add_or_subtract<='0'; --default , no need for this case ,ignore it;
  if ( register_check_for_jump ="0000") then
     address_to_jump<=instruction(2 downto 0 );
     jump_flag<='1';
   else
     jump_flag<='0';
     address_to_jump<="000"; --default , no need for this case ,ignore it;

    end if ;
```

```
  end if;

end process;



end Behavioral;
```

3 bit Adder

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/19/2023 03:58:12 PM
-- Design Name:
-- Module Name: Add_3bit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity Add_3bit is
    Port ( A : in STD_LOGIC_VECTOR (2 downto 0);
           B : in STD_LOGIC_VECTOR (2 downto 0);
           Cin : in STD_LOGIC;
           S : out STD_LOGIC_VECTOR (2 downto 0);
           Cout : out STD_LOGIC);
end Add_3bit;

architecture Behavioral of Add_3bit 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 => B(0),
 C_in => '0', -- Set to ground
 S => S(0),
 C_out => FA0_C);

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

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


end Behavioral;
```

4-bit Add/Subtract unit

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/19/2023 01:57:17 PM
-- Design Name:
-- Module Name: Add_Sub_4bit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity Add_Sub_4bit is
    Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
           B : in STD_LOGIC_VECTOR (3 downto 0);
           Cin : in STD_LOGIC;
           M : in STD_LOGIC;
           S : out STD_LOGIC_VECTOR (3 downto 0);
           Cout : out STD_LOGIC;
           Zero : 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 BFAin,SUM : STD_LOGIC_VECTOR (3 downto 0);

begin
BFAin(0) <= B(0) XOR M;
BFAin(1) <= B(1) XOR M;
BFAin(2) <= B(2) XOR M;
BFAin(3) <= B(3) XOR M;


FA_0 : FA
 port map (
 A => A(0),
 B => BFAin(0),
 C_in => M,
 S => SUM(0),
 C_Out => FA0_C);
```

```
 FA_1 : FA
 port map (
 A => A(1),
 B => BFAin(1),
 C_in => FA0_C,
 S => SUM(1),
 C_Out => FA1_C);

 FA_2 : FA
 port map (
 A => A(2),
 B => BFAin(2),
 C_in => FA1_C,
 S => SUM(2),
 C_Out => FA2_C);

 FA_3 : FA
 port map (
 A => A(3),
 B => BFAin(3),
 C_in => FA2_C,
 S => SUM(3),
 C_Out => Cout);

 S <= SUM;
 zero<='1' when (SUM ="0000") else '0';

end Behavioral;
```

2-way 3-bit multiplexer

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/19/2023 03:58:12 PM
-- Design Name:
-- Module Name: Add_3bit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
```

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


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity Add_3bit is
    Port ( A : in STD_LOGIC_VECTOR (2 downto 0);
           B : in STD_LOGIC_VECTOR (2 downto 0);
           Cin : in STD_LOGIC;
           S : out STD_LOGIC_VECTOR (2 downto 0);
           Cout : out STD_LOGIC);
end Add_3bit;

architecture Behavioral of Add_3bit 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 => B(0),
 C_in => '0', -- Set to ground
 S => S(0),
 C_out => FA0_C);

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

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


end Behavioral;
```

2-way 4-bit multiplexer

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/23/2023 02:10:51 PM
-- Design Name:
-- Module Name: MUX_2_to_1_4bit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
```

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


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity MUX_2_to_1_4bit is
    Port ( load_select : in STD_LOGIC;
           D0 : in STD_LOGIC_VECTOR (3 downto 0);
           D1 : in STD_LOGIC_VECTOR (3 downto 0);
           Y : out STD_LOGIC_VECTOR (3 downto 0));
end MUX_2_to_1_4bit;

architecture Behavioral of MUX_2_to_1_4bit is

begin

process (D0,D1,load_select) is
begin
  if (load_select='0') then
      Y <= D0;

  else
      Y <= D1;
  end if;

end process;


end Behavioral;
```

8-way 4-bit multiplexer

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/23/2023 02:03:15 PM
-- Design Name:
-- Module Name: MUX_8_to_1_4bit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity MUX_8_to_1_4bit is
    Port ( D0 : in STD_LOGIC_VECTOR (3 downto 0);
           D1 : in STD_LOGIC_VECTOR (3 downto 0);
           D2 : in STD_LOGIC_VECTOR (3 downto 0);
           D3 : in STD_LOGIC_VECTOR (3 downto 0);
           D4 : in STD_LOGIC_VECTOR (3 downto 0);
```

```
        D5 : in STD_LOGIC_VECTOR (3 downto 0);
        D6 : in STD_LOGIC_VECTOR (3 downto 0);
        D7 : in STD_LOGIC_VECTOR (3 downto 0);
        S : in STD_LOGIC_VECTOR (2 downto 0);
        Y : out STD_LOGIC_VECTOR (3 downto 0));
end MUX_8_to_1_4bit;

architecture Behavioral of MUX_8_to_1_4bit is

begin

--with S select
--    Y <= D0 when "000",
--         D1 when  "001",
--         D2 when "010",
--         D3 when "011",
--         D4 when "100",
--         D5 when "101",
--         D6 when "110",
--         D7 when "111",
--         "000"  when others;

process (D0,D1,D2,D3,D4,D5,D6,D7,S) is
begin
  if (S="000") then
      Y <= D0;
       elsif (S="001") then
      Y <= D1;
       elsif (S="010") then
      Y <= D2;
       elsif (S="011") then
      Y <= D3;
       elsif (S="100") then
      Y <= D4;
       elsif (S="101") then
      Y <= D5;
       elsif (S="110") then
      Y <= D6;
       elsif (S="111") then
      Y <= D7;


  else
      Y <= D0;
```

```
  end if;

end process;



end Behavioral;
```

Slow clock

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/08/2023 02:38:07 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 := '1' );
end Slow_Clk;

architecture Behavioral of Slow_Clk is

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

begin
 --For 100 MHz input clock this generates 1 Hz clock
 process(Clk_in)
  begin
   if (rising_edge(Clk_in)) then
      count <=count +1;

    if (count = 10000000)then
         --if (count = 5)then

         clk_status <= not clk_status;
         Clk_out <=clk_status;
         count <=1;
       end if ;

  end if ;

  end process;

end Behavioral;
```

Lookup Table 7 Segment

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/04/2023 08:35:01 AM
```

```
-- 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;
```

TEST_BENCH_CODE

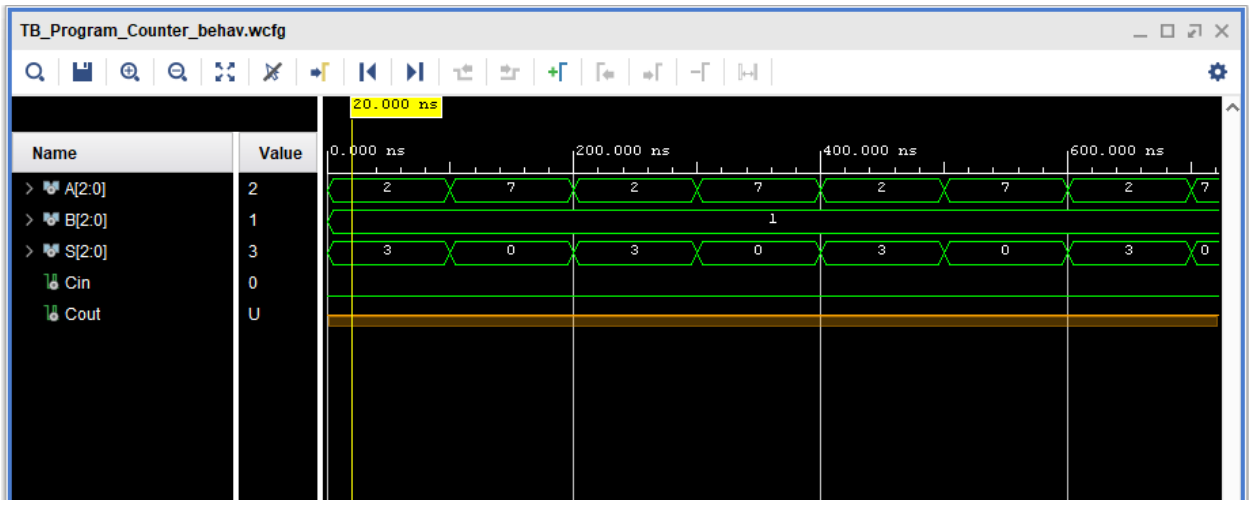
TB_3_bit_adder

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/19/2023 04:03:48 PM
-- Design Name:
-- Module Name: TB_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 TB_3_bit_adder is
--  Port ( );
end TB_3_bit_adder;

architecture Behavioral of TB_3_bit_adder is

component Adder_03_bit
    Port ( A : in STD_LOGIC_VECTOR (2 downto 0);

           Cin : in STD_LOGIC;
           S : out STD_LOGIC_VECTOR (2 downto 0);
           Cout : out STD_LOGIC);
end component;

signal A,B,S : STD_LOGIC_VECTOR (2 downto 0);
signal Cin,Cout : STD_LOGIC;

begin
uut: Adder_03_bit
port map(
A => A,

Cin => Cin,
S => S,
Cout => Cout
);
```

```
process
begin
cin<='0';
B <= "001";
A <= "010";
wait for 100ns;

A <= "111";
wait for 100ns;

end process;


end Behavioral;
```

Timing Diagram



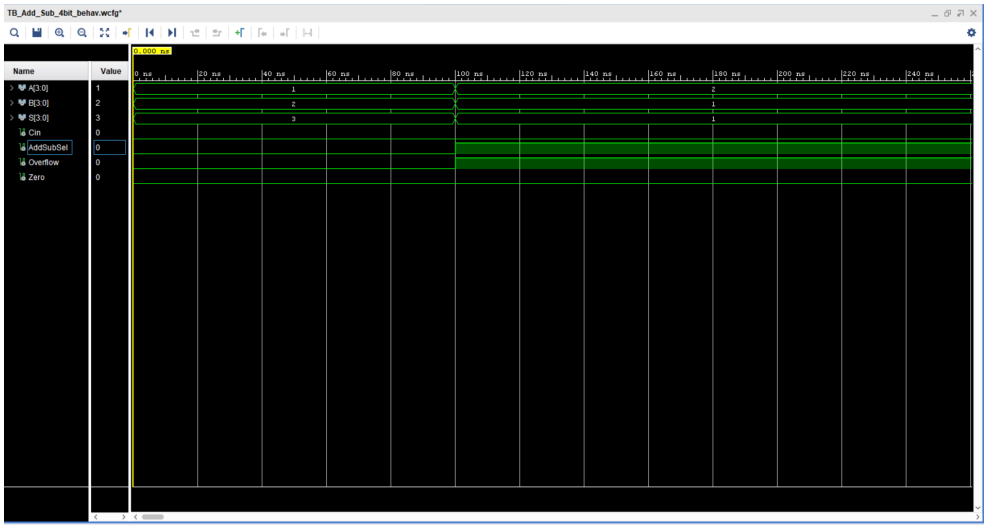
TB_Add_Sub_4bit

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/19/2023 02:29:00 PM
-- Design Name:
-- Module Name: TB_Add_Sub_4bit - Behavioral
```

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


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity TB_Add_Sub_4bit is
--  Port ( );
end TB_Add_Sub_4bit;

architecture Behavioral of TB_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);
               Cin : in STD_LOGIC;
               M : in STD_LOGIC;
               S : out STD_LOGIC_VECTOR (3 downto 0);
               Cout : out STD_LOGIC;
               Zero : out STD_LOGIC);

    end component;

signal A,B,S : STD_LOGIC_VECTOR (3 downto 0);
signal Cin,AddSubSel,Overflow,Zero : STD_LOGIC;
```

```
begin

UUT: Add_Sub_4bit
port map(
    A => A,
    B => B,
    Cin => Cin,
    M => AddSubSel,
    S => S,
    Cout => Overflow,
    Zero => Zero
    );

process
begin
Cin <= '0';
A <= "0001";
B <= "0010";
AddSubSel <= '0';

wait for 100ns;
A <= "0010";
B <= "0001";
AddSubSel <= '1';

wait ;

end process;


end Behavioral;
```

Timing Diagram



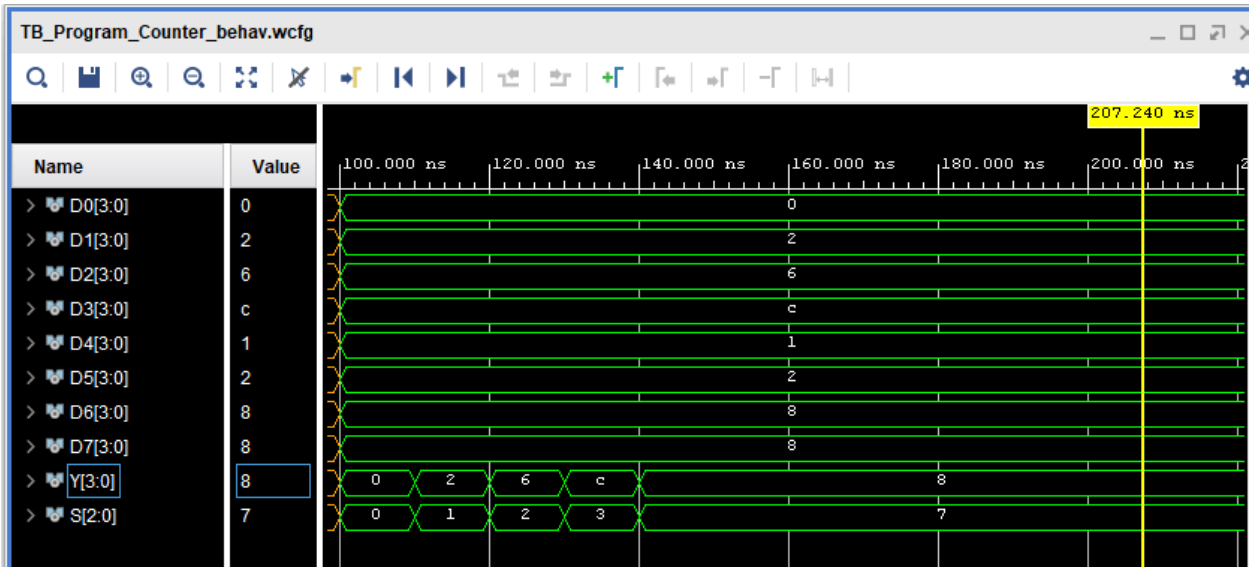
TB_MUX_8_to_1_4bit.vhd

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/23/2023 02:06:59 PM
-- Design Name:
-- Module Name: TB_MUX_8_to_1_4bit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------
```

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

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

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

entity TB_MUX_8_to_1_4bit is
--  Port ( );
end TB_MUX_8_to_1_4bit;

architecture Behavioral of TB_MUX_8_to_1_4bit is

component MUX_8_to_1_4_bit
Port ( D0 : in STD_LOGIC_VECTOR (3 downto 0);
       D1 : in STD_LOGIC_VECTOR (3 downto 0);
       D2 : in STD_LOGIC_VECTOR (3 downto 0);
       D3 : in STD_LOGIC_VECTOR (3 downto 0);
       D4 : in STD_LOGIC_VECTOR (3 downto 0);
       D5 : in STD_LOGIC_VECTOR (3 downto 0);
       D6 : in STD_LOGIC_VECTOR (3 downto 0);
       D7 : in STD_LOGIC_VECTOR (3 downto 0);
       Y : out STD_LOGIC_VECTOR (3 downto 0);
       S : in STD_LOGIC_VECTOR ( 2 downto 0));
end component;
signal D0,D1,D2,D3,D4,D5,D6,D7 : std_logic_vector ( 3 downto 0);
signal Y : std_logic_vector (3 downto 0);
signal S : std_logic_vector (2 downto 0);
begin
UUT : MUX_8_to_1_4_bit port map(
   D0=>D0,
   D1=>D1,
   D2=>D2,
   D3=>D3,
   D4=>D4,
   D5=>D5,
   D6=>D6,
   D7=>D7,
   Y=>Y,
```

```
    S=>S);

    process
    begin
    wait for 100 ns;
    D0<="0000";
    D1<="0010";
    D2<="0110";
    D3<="1100";
    D4<="0001";
    D5<="0010";
    D6<="1000";
    D7<="1000";

    S<="000";
    wait for 10ns;
     S<="001";
    wait for 10ns;
    S<="010";
    wait for 10ns;
     S<="011";
    wait for 10ns;
     S<="111";
    wait for 10ns;

    end process;


end Behavioral;
```

Timing Diagram



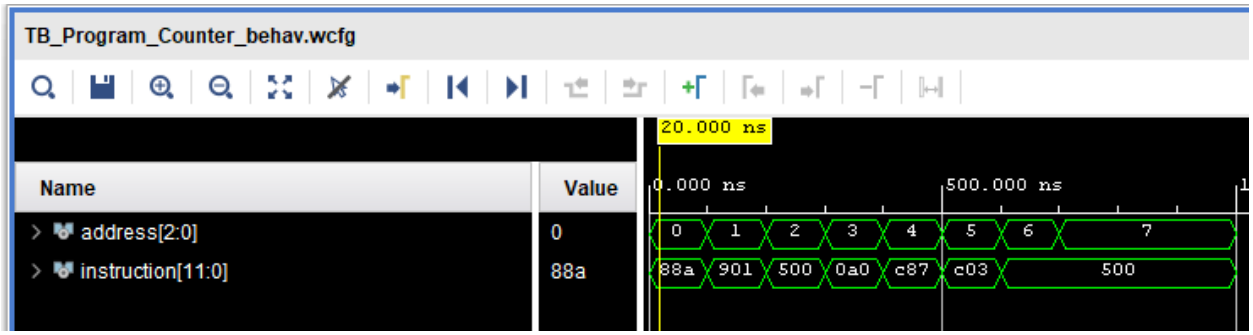
TB_ROM.vhd

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/17/2023 09:39:13 PM
-- Design Name:
-- Module Name: TB_ROM - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


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

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

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

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

architecture Behavioral of TB_ROM is
component ROM
    Port ( address : in STD_LOGIC_VECTOR (2 downto 0);
           instruction : out STD_LOGIC_VECTOR (11 downto 0));
end component;

signal address : STD_LOGIC_VECTOR (2 downto 0);
signal instruction : STD_LOGIC_VECTOR (11 downto 0);

begin
uut : ROM port map ( address => address,
                     instruction => instruction);

process
begin
address <= "000";

WAIT for 100 ns;
address <= "001";

WAIT for 100 ns;
address <= "010";

WAIT for 100 ns;
address <= "011";

WAIT for 100 ns;
address <= "100";

WAIT for 100 ns;
```

```
address <= "101";

WAIT for 100 ns;
address <= "110";

WAIT for 100 ns;
address <= "111";

WAIT;
end process;
end Behavioral;
```

Timing Diagram



TB_Reg_Bank.vhd

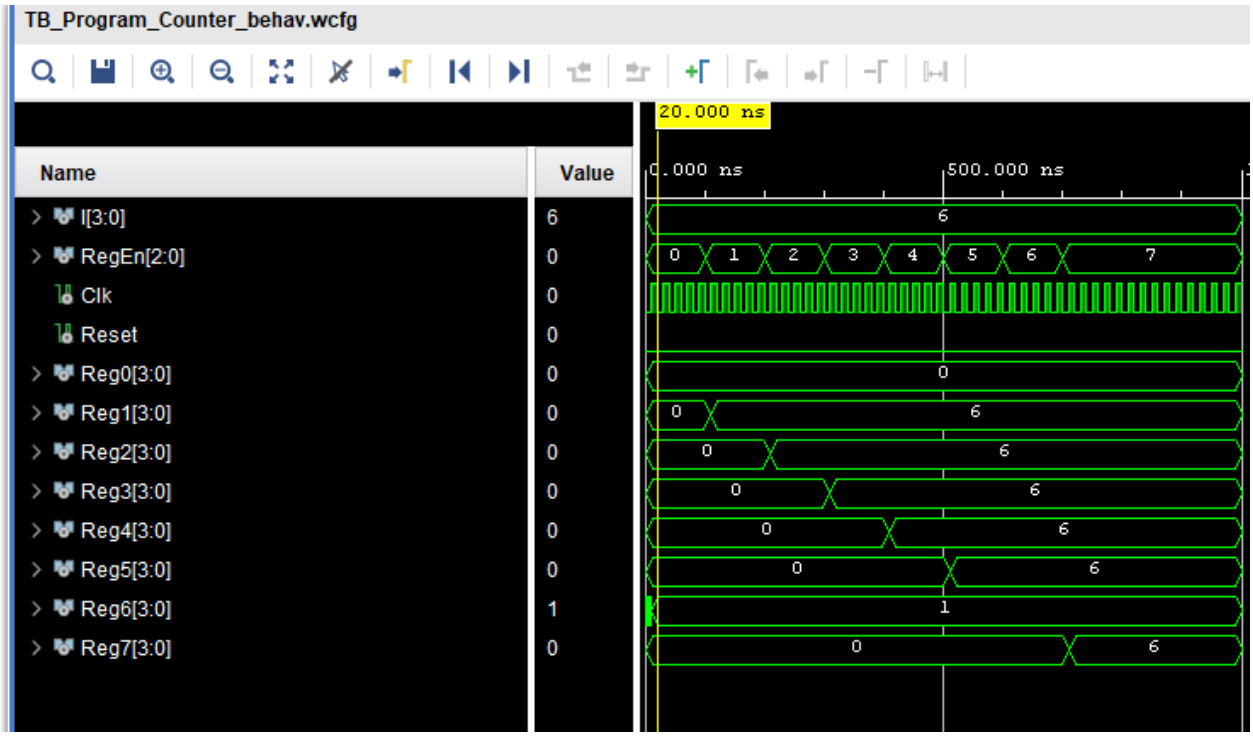
```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/19/2023 09:35:46 AM
-- Design Name:
-- Module Name: TB_Reg_Bank - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
```

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


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

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

architecture Behavioral of TB_Reg_Bank is
    component Reg_Bank
        Port ( I : in STD_LOGIC_VECTOR (3 downto 0);
               RegEn : in STD_LOGIC_VECTOR (2 downto 0);
               Clk : in STD_LOGIC;
               Reset : in STD_LOGIC;
               Reg0 : out STD_LOGIC_VECTOR (3 downto 0);
               Reg1 : out STD_LOGIC_VECTOR (3 downto 0);
               Reg2 : out STD_LOGIC_VECTOR (3 downto 0);
               Reg3 : out STD_LOGIC_VECTOR (3 downto 0);
               Reg4 : out STD_LOGIC_VECTOR (3 downto 0);
               Reg5 : out STD_LOGIC_VECTOR (3 downto 0);
               Reg6 : out STD_LOGIC_VECTOR (3 downto 0);
               Reg7 : out STD_LOGIC_VECTOR (3 downto 0));
    end component;

signal I : STD_LOGIC_VECTOR (3 downto 0);
signal RegEn : STD_LOGIC_VECTOR (2 downto 0);
signal Clk : STD_LOGIC;
signal Reset : STD_LOGIC := '0';
signal Reg0,Reg1,Reg2,Reg3,Reg4,Reg5,Reg6,Reg7 : STD_LOGIC_VECTOR (3 downto 0);

begin
```

```
UUT: Reg_Bank
port map(
 I => I,
 RegEn => RegEn,
 Clk => Clk,
 Reset => Reset,
 Reg0 => Reg0,
 Reg1 => Reg1,
 Reg2 => Reg2,
 Reg3 => Reg3,
 Reg4 => Reg4,
 Reg5 => Reg5,
 Reg6 => Reg6,
 Reg7 => Reg7
);


Clock : process
    begin
        clk<='0';
wait for 10ns;
 clk<='1';
wait for 10ns;
end process;

process
begin
   I <= "0110";
   RegEn <= "000";
   wait for 100ns;

   I <= "0110";
   RegEn <= "001";
   wait for 100ns;

   I <= "0110";
   RegEn <= "010";
   wait for 100ns;

   I <= "0110";
   RegEn <= "011";
   wait for 100ns;
```

```
  I <= "0110";
  RegEn <= "100";
  wait for 100ns;

  I <= "0110";
  RegEn <= "101";
  wait for 100ns;

  I <= "0110";
  RegEn <= "110";
  wait for 100ns;

   I <= "0110";
   RegEn <= "111";
   wait;



end process;

end Behavioral;
```

Timing Diagram

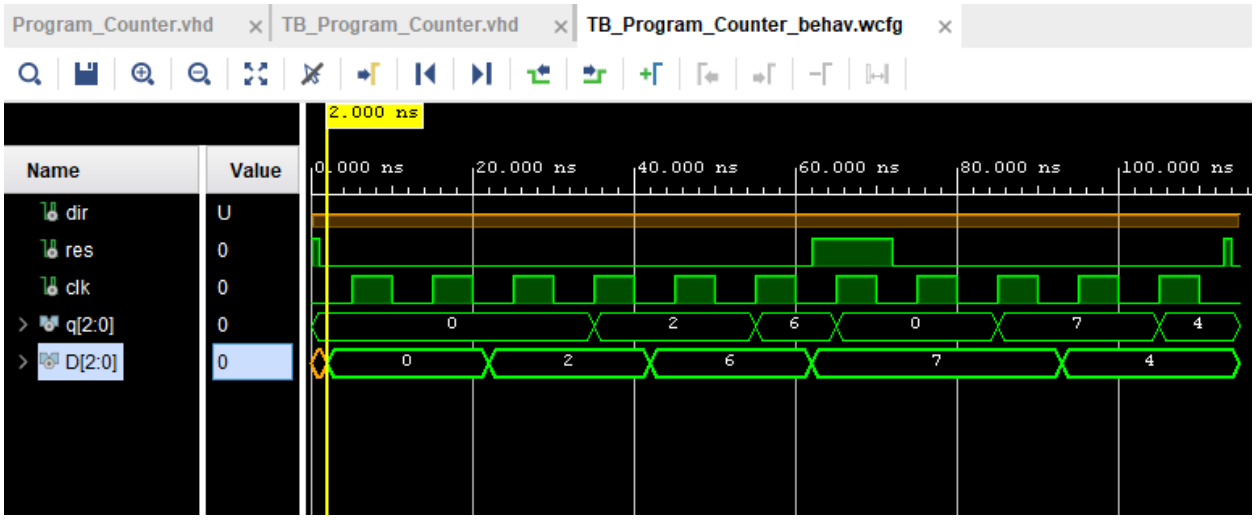


TB_Program_Counter.vhd

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/17/2023 02:18:46 PM
-- 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(2 downto 0);
           Res : in STD_LOGIC;
           Clk : in STD_LOGIC;
```

```
        Q : out STD_LOGIC_VECTOR (2 downto 0));
END COMPONENT;

SIGNAL dir, res : STD_LOGIC;
SIGNAL clk : STD_LOGIC := '0';
SIGNAL q,D : STD_LOGIC_VECTOR (2 DOWNTO 0);

begin
  UUT: Program_Counter PORT MAP(
  D_IN=>D,
       Res => res,
       Clk => clk,
       Q => q
    );
    process
    begin
       WAIT FOR 5ns;
       clk <= NOT(clk);
    end process;

     process
    begin

    res<='1';
    wait for 1ns;
    res<='0';
    wait for 1ns;

       D<="000";
       WAIT FOR 20ns;
       D<="010";
       WAIT FOR 20ns;
       D<="110";
       WAIT FOR 20ns;
       D<="111";
         res<='1';
    wait for 10ns;
    res<='0';
    wait for 1ns;
       WAIT FOR 20ns;
         D<="100";
       WAIT FOR 20ns;
    end process;
end Behavioral;
```

Timing Diagram



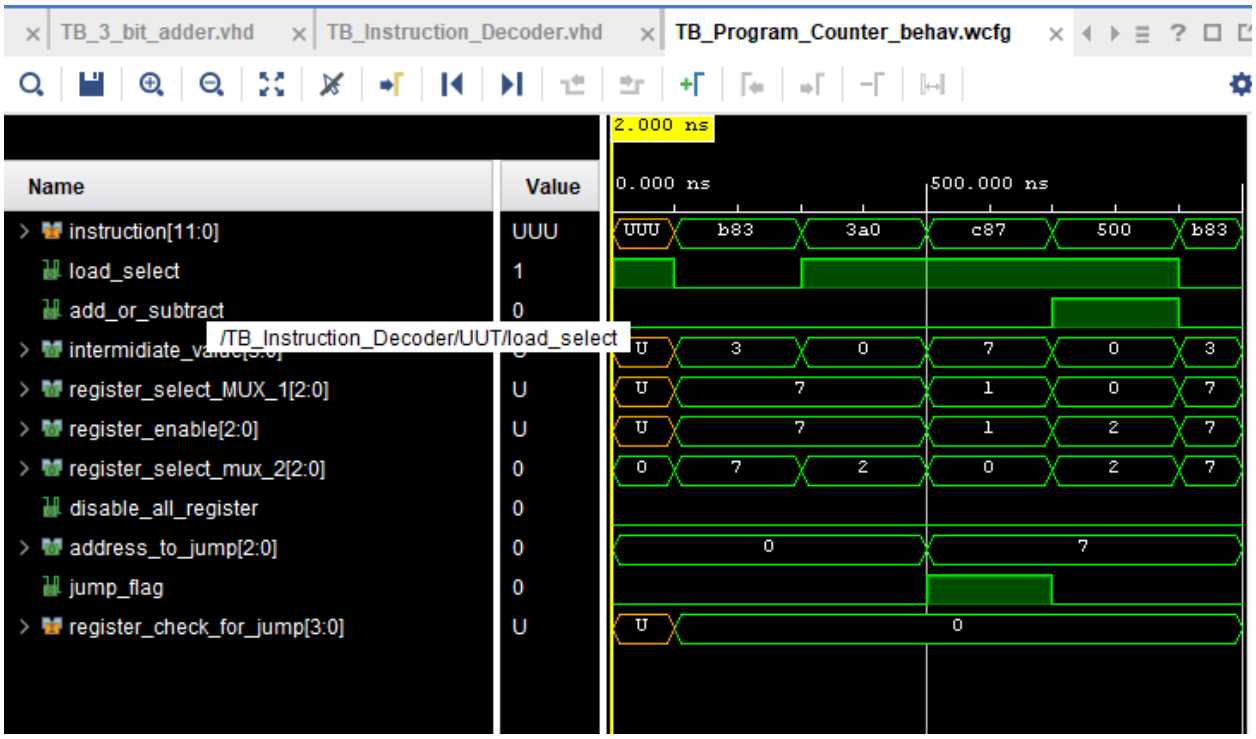
TB_Instruction_Decoder.vhd

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05/18/2023 04:14:51 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  port
         ( instruction : in STD_LOGIC_VECTOR (11 downto 0);
        load_select : out STD_LOGIC;
        add_or_subtract : out STD_LOGIC ;
        intermidiate_value : out STD_LOGIC_VECTOR(3 downto 0);
        register_select_MUX_1 : out STD_LOGIC_VECTOR (2 downto 0);
        register_enable : out STD_LOGIC_VECTOR (2 downto 0);
        register_select_mux_2 : out STD_LOGIC_VECTOR (2 downto 0);
        disable_all_register : out STD_LOGIC ;
        address_to_jump : out STD_LOGIC_VECTOR (2 downto 0)  ;
        jump_flag : out STD_LOGIC;
        register_check_for_jump : in STD_LOGIC_VECTOR (3 downto 0));

end component;
signal instruction : STD_LOGIC_VECTOR (11 downto 0);
signal load_select,add_or_subtract,disable_all_register ,jump_flag: STD_LOGIC;
signal register_enable,register_select_mux_2,register_select_MUX_1,address_to_jump
:STD_LOGIC_VECTOR (2 downto 0);
signal intermidiate_value ,register_check_for_jump :STD_LOGIC_VECTOR(3 downto 0);

begin
UUT : Instruction_Decoder  port map
( instruction=>instruction ,
        load_select=>load_select ,
        add_or_subtract=>add_or_subtract,
```

```
        intermidiate_value=>intermidiate_value,
        register_select_MUX_1=>register_select_MUX_1 ,
        register_enable=>register_enable ,
        register_select_mux_2=>register_select_mux_2 ,
        disable_all_register=>disable_all_register ,
        address_to_jump=>address_to_jump ,
        jump_flag=>jump_flag ,
        register_check_for_jump=>register_check_for_jump );
 process
  begin
  wait for 100ns;
  instruction <="101110000011";
  register_check_for_jump<="0000";
  wait for 100ns;
  wait for 100ns;
  instruction <="001110100000";
  register_check_for_jump<="0000";
  wait for 100ns;
  wait for 100ns;
  instruction <="110010000111";
  register_check_for_jump<="0000";
  wait for 100ns;
  wait for 100ns;
  instruction <="010100000000";
  register_check_for_jump<="0000";
  wait for 100ns;
  end process;

end Behavioral;
```

Timing Diagram



Constraints

This file is a general .xdc for the Basys3 rev B board
To use it in a project:
- uncomment the lines corresponding to used pins
- rename the used ports (in each line, after get_ports) according to the top level signal names in the project

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]

```
## Switches
#set_property PACKAGE_PIN V17 [get_ports {sw[0]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {sw[0]}]
#set_property PACKAGE_PIN V16 [get_ports {sw[1]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {sw[1]}]
#set_property PACKAGE_PIN W16 [get_ports {sw[2]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {sw[2]}]
#set_property PACKAGE_PIN W17 [get_ports {sw[3]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {sw[3]}]
#set_property PACKAGE_PIN W15 [get_ports {sw[4]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {sw[4]}]
#set_property PACKAGE_PIN V15 [get_ports {sw[5]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {sw[5]}]
#set_property PACKAGE_PIN W14 [get_ports {sw[6]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {sw[6]}]
#set_property PACKAGE_PIN W13 [get_ports {sw[7]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {sw[7]}]
#set_property PACKAGE_PIN V2 [get_ports {sw[8]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {sw[8]}]
#set_property PACKAGE_PIN T3 [get_ports {sw[9]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {sw[9]}]
#set_property PACKAGE_PIN T2 [get_ports {sw[10]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {sw[10]}]
#set_property PACKAGE_PIN R3 [get_ports {sw[11]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {sw[11]}]
#set_property PACKAGE_PIN W2 [get_ports {sw[12]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {sw[12]}]
#set_property PACKAGE_PIN U1 [get_ports {sw[13]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {sw[13]}]
#set_property PACKAGE_PIN T1 [get_ports {sw[14]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {sw[14]}]
#set_property PACKAGE_PIN R2 [get_ports {sw[15]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {sw[15]}]


## LEDs
set_property PACKAGE_PIN U16 [get_ports {Reg_7_output[0]}]
	set_property IOSTANDARD LVCMOS33 [get_ports {Reg_7_output[0]}]
set_property PACKAGE_PIN E19 [get_ports {Reg_7_output[1]}]
	set_property IOSTANDARD LVCMOS33 [get_ports {Reg_7_output[1]}]
```

```
set_property PACKAGE_PIN U19 [get_ports {Reg_7_output[2]}]
        set_property IOSTANDARD LVCMOS33 [get_ports {Reg_7_output[2]}]
set_property PACKAGE_PIN V19 [get_ports {Reg_7_output[3]}]
        set_property IOSTANDARD LVCMOS33 [get_ports {Reg_7_output[3]}]
#set_property PACKAGE_PIN W18 [get_ports {led[4]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {led[4]}]
#set_property PACKAGE_PIN U15 [get_ports {led[5]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {led[5]}]
#set_property PACKAGE_PIN U14 [get_ports {led[6]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {led[6]}]
#set_property PACKAGE_PIN V14 [get_ports {led[7]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {led[7]}]
#set_property PACKAGE_PIN V13 [get_ports {led[8]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {led[8]}]
#set_property PACKAGE_PIN V3 [get_ports {led[9]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {led[9]}]
#set_property PACKAGE_PIN W3 [get_ports {led[10]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {led[10]}]
#set_property PACKAGE_PIN U3 [get_ports {led[11]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {led[11]}]
#set_property PACKAGE_PIN P3 [get_ports {led[12]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {led[12]}]
#set_property PACKAGE_PIN N3 [get_ports {led[13]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {led[13]}]
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 {seg[0]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {seg[0]}]
#set_property PACKAGE_PIN W6 [get_ports {seg[1]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {seg[1]}]
#set_property PACKAGE_PIN U8 [get_ports {seg[2]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {seg[2]}]
#set_property PACKAGE_PIN V8 [get_ports {seg[3]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {seg[3]}]
#set_property PACKAGE_PIN U5 [get_ports {seg[4]}]
```

```
        #set_property IOSTANDARD LVCMOS33 [get_ports {seg[4]}]
#set_property PACKAGE_PIN V5 [get_ports {seg[5]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {seg[5]}]
#set_property PACKAGE_PIN U7 [get_ports {seg[6]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {seg[6]}]

#set_property PACKAGE_PIN V7 [get_ports dp]
        #set_property IOSTANDARD LVCMOS33 [get_ports dp]

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

set_property PACKAGE_PIN W7 [get_ports {Led_7_Seg[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Led_7_Seg[0]}]
set_property PACKAGE_PIN W6 [get_ports {Led_7_Seg[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Led_7_Seg[1]}]
set_property PACKAGE_PIN U8 [get_ports {Led_7_Seg[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Led_7_Seg[2]}]
set_property PACKAGE_PIN V8 [get_ports {Led_7_Seg[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Led_7_Seg[3]}]
set_property PACKAGE_PIN U5 [get_ports {Led_7_Seg[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Led_7_Seg[4]}]
set_property PACKAGE_PIN V5 [get_ports {Led_7_Seg[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Led_7_Seg[5]}]
set_property PACKAGE_PIN U7 [get_ports {Led_7_Seg[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Led_7_Seg[6]}]

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

```
set_property IOSTANDARD LVCMOS33 [get_ports {AN[3]}]
##Buttons
set_property PACKAGE_PIN U18 [get_ports Reset]
	set_property IOSTANDARD LVCMOS33 [get_ports Reset]
#set_property PACKAGE_PIN T18 [get_ports btnU]
	#set_property IOSTANDARD LVCMOS33 [get_ports btnU]
#set_property PACKAGE_PIN W19 [get_ports btnL]
	#set_property IOSTANDARD LVCMOS33 [get_ports btnL]
#set_property PACKAGE_PIN T17 [get_ports btnR]
	#set_property IOSTANDARD LVCMOS33 [get_ports btnR]
#set_property PACKAGE_PIN U17 [get_ports btnD]
	#set_property IOSTANDARD LVCMOS33 [get_ports btnD]



##Pmod Header JA
##Sch name = JA1
#set_property PACKAGE_PIN J1 [get_ports {JA[0]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {JA[0]}]
##Sch name = JA2
#set_property PACKAGE_PIN L2 [get_ports {JA[1]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {JA[1]}]
##Sch name = JA3
#set_property PACKAGE_PIN J2 [get_ports {JA[2]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {JA[2]}]
##Sch name = JA4
#set_property PACKAGE_PIN G2 [get_ports {JA[3]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {JA[3]}]
##Sch name = JA7
#set_property PACKAGE_PIN H1 [get_ports {JA[4]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {JA[4]}]
##Sch name = JA8
#set_property PACKAGE_PIN K2 [get_ports {JA[5]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {JA[5]}]
##Sch name = JA9
#set_property PACKAGE_PIN H2 [get_ports {JA[6]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {JA[6]}]
##Sch name = JA10
#set_property PACKAGE_PIN G3 [get_ports {JA[7]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {JA[7]}]
```

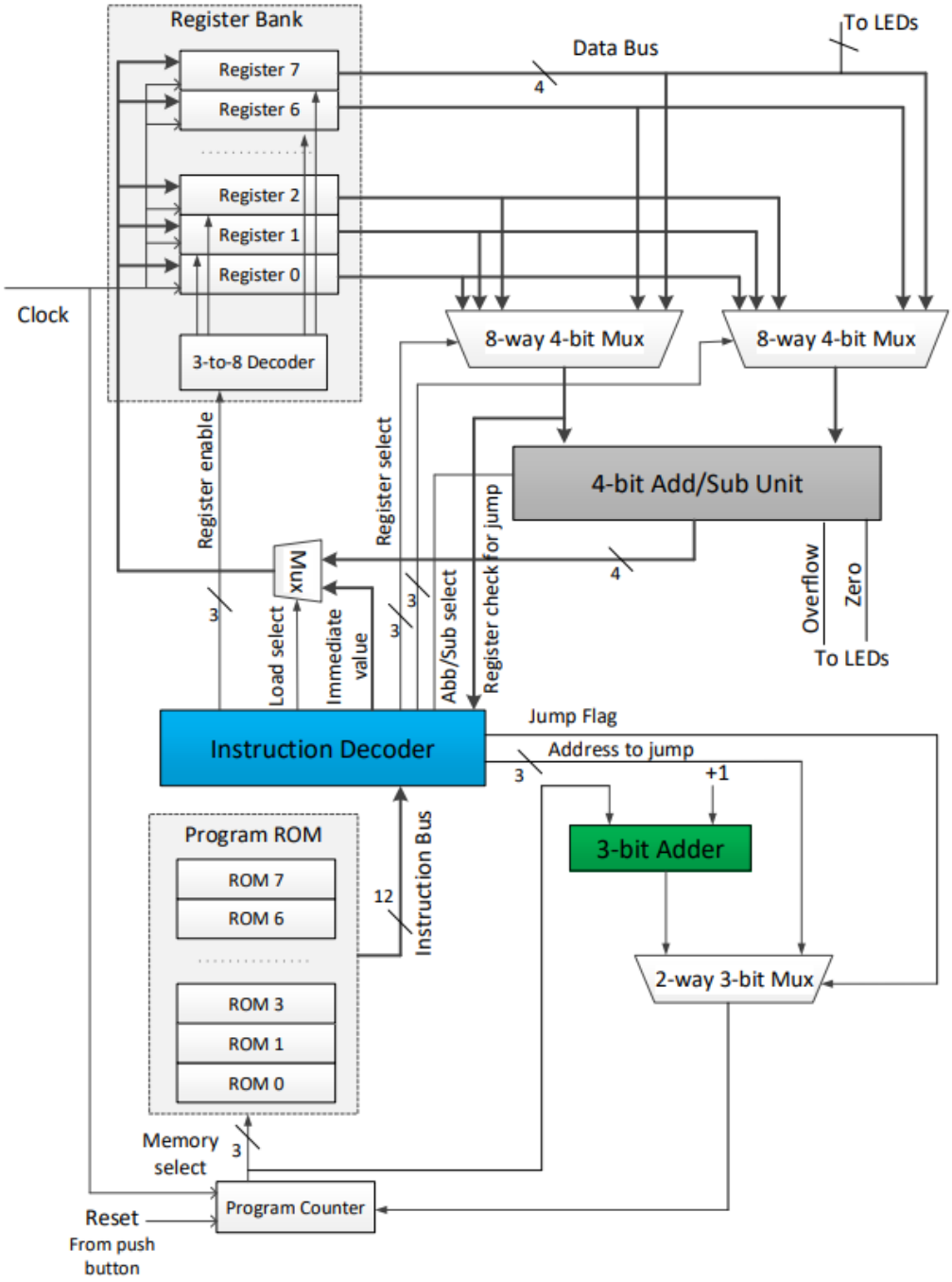
```
##Pmod Header JB
##Sch name = JB1
#set_property PACKAGE_PIN A14 [get_ports {JB[0]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {JB[0]}]
##Sch name = JB2
#set_property PACKAGE_PIN A16 [get_ports {JB[1]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {JB[1]}]
##Sch name = JB3
#set_property PACKAGE_PIN B15 [get_ports {JB[2]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {JB[2]}]
##Sch name = JB4
#set_property PACKAGE_PIN B16 [get_ports {JB[3]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {JB[3]}]
##Sch name = JB7
#set_property PACKAGE_PIN A15 [get_ports {JB[4]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {JB[4]}]
##Sch name = JB8
#set_property PACKAGE_PIN A17 [get_ports {JB[5]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {JB[5]}]
##Sch name = JB9
#set_property PACKAGE_PIN C15 [get_ports {JB[6]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {JB[6]}]
##Sch name = JB10
#set_property PACKAGE_PIN C16 [get_ports {JB[7]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {JB[7]}]



##Pmod Header JC
##Sch name = JC1
#set_property PACKAGE_PIN K17 [get_ports {JC[0]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {JC[0]}]
##Sch name = JC2
#set_property PACKAGE_PIN M18 [get_ports {JC[1]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {JC[1]}]
##Sch name = JC3
#set_property PACKAGE_PIN N17 [get_ports {JC[2]}]
```

```
        #set_property IOSTANDARD LVCMOS33 [get_ports {JC[2]}]
##Sch name = JC4
#set_property PACKAGE_PIN P18 [get_ports {JC[3]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {JC[3]}]
##Sch name = JC7
#set_property PACKAGE_PIN L17 [get_ports {JC[4]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {JC[4]}]
##Sch name = JC8
#set_property PACKAGE_PIN M19 [get_ports {JC[5]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {JC[5]}]
##Sch name = JC9
#set_property PACKAGE_PIN P17 [get_ports {JC[6]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {JC[6]}]
##Sch name = JC10
#set_property PACKAGE_PIN R18 [get_ports {JC[7]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {JC[7]}]


##Pmod Header JXADC
##Sch name = XA1_P
#set_property PACKAGE_PIN J3 [get_ports {JXADC[0]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {JXADC[0]}]
##Sch name = XA2_P
#set_property PACKAGE_PIN L3 [get_ports {JXADC[1]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {JXADC[1]}]
##Sch name = XA3_P
#set_property PACKAGE_PIN M2 [get_ports {JXADC[2]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {JXADC[2]}]
##Sch name = XA4_P
#set_property PACKAGE_PIN N2 [get_ports {JXADC[3]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {JXADC[3]}]
##Sch name = XA1_N
#set_property PACKAGE_PIN K3 [get_ports {JXADC[4]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {JXADC[4]}]
##Sch name = XA2_N
#set_property PACKAGE_PIN M3 [get_ports {JXADC[5]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {JXADC[5]}]
##Sch name = XA3_N
#set_property PACKAGE_PIN M1 [get_ports {JXADC[6]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {JXADC[6]}]
```

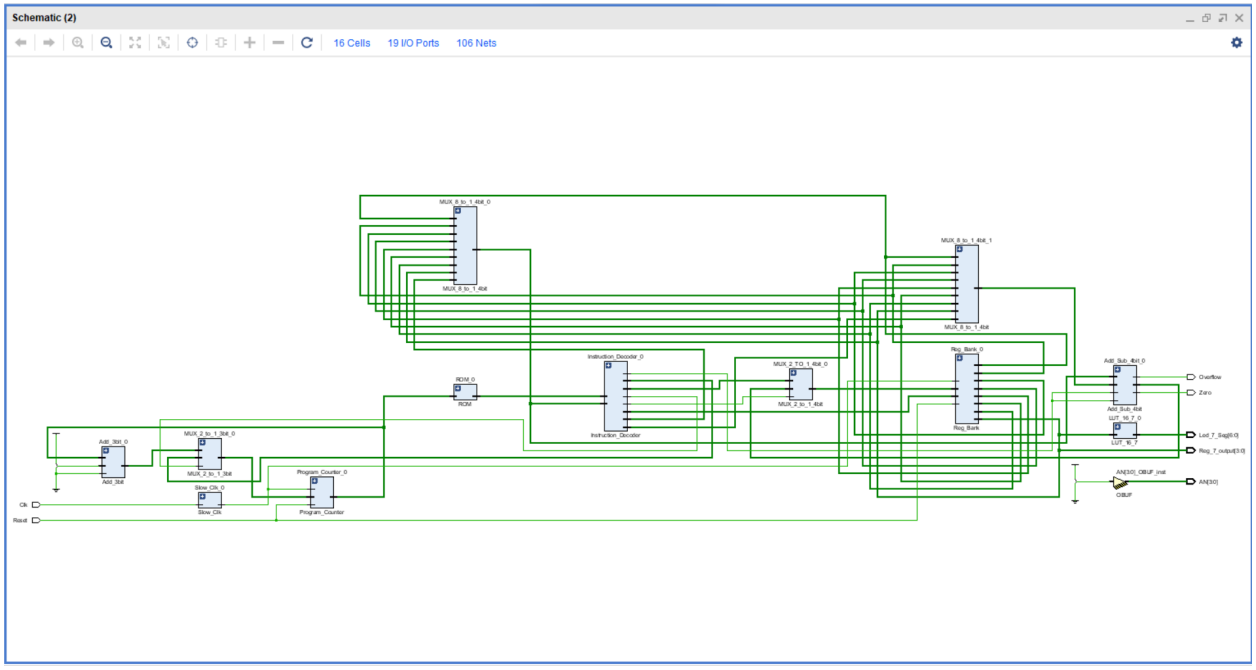
```
##Sch name = XA4_N
#set_property PACKAGE_PIN N1 [get_ports {JXADC[7]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {JXADC[7]}]



##VGA Connector
#set_property PACKAGE_PIN G19 [get_ports {vgaRed[0]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {vgaRed[0]}]
#set_property PACKAGE_PIN H19 [get_ports {vgaRed[1]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {vgaRed[1]}]
#set_property PACKAGE_PIN J19 [get_ports {vgaRed[2]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {vgaRed[2]}]
#set_property PACKAGE_PIN N19 [get_ports {vgaRed[3]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {vgaRed[3]}]
#set_property PACKAGE_PIN N18 [get_ports {vgaBlue[0]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {vgaBlue[0]}]
#set_property PACKAGE_PIN L18 [get_ports {vgaBlue[1]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {vgaBlue[1]}]
#set_property PACKAGE_PIN K18 [get_ports {vgaBlue[2]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {vgaBlue[2]}]
#set_property PACKAGE_PIN J18 [get_ports {vgaBlue[3]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {vgaBlue[3]}]
#set_property PACKAGE_PIN J17 [get_ports {vgaGreen[0]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {vgaGreen[0]}]
#set_property PACKAGE_PIN H17 [get_ports {vgaGreen[1]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {vgaGreen[1]}]
#set_property PACKAGE_PIN G17 [get_ports {vgaGreen[2]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {vgaGreen[2]}]
#set_property PACKAGE_PIN D17 [get_ports {vgaGreen[3]}]
	#set_property IOSTANDARD LVCMOS33 [get_ports {vgaGreen[3]}]
#set_property PACKAGE_PIN P19 [get_ports Hsync]
	#set_property IOSTANDARD LVCMOS33 [get_ports Hsync]
#set_property PACKAGE_PIN R19 [get_ports Vsync]
	#set_property IOSTANDARD LVCMOS33 [get_ports Vsync]



##USB-RS232 Interface
#set_property PACKAGE_PIN B18 [get_ports RsRx]
	#set_property IOSTANDARD LVCMOS33 [get_ports RsRx]
```

```
#set_property PACKAGE_PIN A18 [get_ports RsTx]
        #set_property IOSTANDARD LVCMOS33 [get_ports RsTx]


##USB HID (PS/2)
#set_property PACKAGE_PIN C17 [get_ports PS2Clk]
        #set_property IOSTANDARD LVCMOS33 [get_ports PS2Clk]
        #set_property PULLUP true [get_ports PS2Clk]
#set_property PACKAGE_PIN B17 [get_ports PS2Data]
        #set_property IOSTANDARD LVCMOS33 [get_ports PS2Data]
        #set_property PULLUP true [get_ports PS2Data]


##Quad SPI Flash
##Note that CCLK_0 cannot be placed in 7 series devices. You can access it using the
##STARTUPE2 primitive.
#set_property PACKAGE_PIN D18 [get_ports {QspiDB[0]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {QspiDB[0]}]
#set_property PACKAGE_PIN D19 [get_ports {QspiDB[1]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {QspiDB[1]}]
#set_property PACKAGE_PIN G18 [get_ports {QspiDB[2]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {QspiDB[2]}]
#set_property PACKAGE_PIN F18 [get_ports {QspiDB[3]}]
        #set_property IOSTANDARD LVCMOS33 [get_ports {QspiDB[3]}]
#set_property PACKAGE_PIN K19 [get_ports QspiCSn]
        #set_property IOSTANDARD LVCMOS33 [get_ports QspiCSn]
```

High-level diagram of the nanoprocessor



Elaborated Design



utilization report

| Tool Version : Vivado v.2018.2 (win64) Build 2258646 Thu Jun 14 20:03:12 MDT 2018
| Date : Sun Jun 11 11:47:43 2023
| Host : LAPTOP-KKLN5P31 running 64-bit major release (build 9200)
| Command : report_utilization -file {D:/2nd sem/Computer Organizations and Digital Design/Lab/Lab_9/MicroProcessor/MicroProcessor.srcs/sources_1/new/UtilizationReport} -name utilization_2
| Design : Microprocessor
| Device : 7a35tcpg236-1
| Design State : Synthesized

Utilization Design Information

Table of Contents
---
1. Slice Logic
1.1 Summary of Registers by Type
2. Memory
3. DSP
4. IO and GT Specific
5. Clocking
6. Specific Feature
7. Primitives
8. Black Boxes
9. Instantiated Netlists

1. Slice Logic

Site Type	Used	Fixed	Available	Util%
Slice LUTs*	50	0	20800	0.24
LUT as Logic	50	0	20800	0.24
LUT as Memory	0	0	9600	0.00
Slice Registers	70	0	41600	0.17
Register as Flip Flop	67	0	41600	0.16
Register as Latch	3	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.

1.1 Summary of Registers by Type

Total	Clock Enable	Synchronous	Asynchronous
0	_	-	-
0	_	-	Set
0	_	-	Reset
0	_	Set	-
0	_	Reset	-
0	Yes	-	-

0	Yes	-	Set
3	Yes	-	Reset
9	Yes	Set	-
58	Yes	Reset	-

2. Memory

Site Type	Used	Fixed	Available	Util%
Block RAM Tile	0	0	50	0.00
RAMB36/FIFO*	0	0	50	0.00
RAMB18	0	0	100	0.00

* Note: Each Block RAM Tile only has one FIFO logic available and therefore can accommodate only one FIFO36E1 or one FIFO18E1. However, if a FIFO18E1 occupies a Block RAM Tile, that tile can still accommodate a RAMB18E1

3. DSP

Site Type	Used	Fixed	Available	Util%
DSPs	0	0	90	0.00

4. IO and GT Specific

Site Type	Used	Fixed	Available	Util%
Bonded IOB	19	19	106	17.92
IOB Master Pads	7			
IOB Slave Pads	12			
Bonded IPADs	0	0	10	0.00
Bonded OPADs	0	0	4	0.00
PHY_CONTROL	0	0	5	0.00

PHASER_REF	0	0	5	0.00
OUT_FIFO	0	0	20	0.00
IN_FIFO	0	0	20	0.00
IDELAYCTRL	0	0	5	0.00
IBUFDS	0	0	104	0.00
GTPE2_CHANNEL	0	0	2	0.00
PHASER_OUT/PHASER_OUT_PHY	0	0	20	0.00
PHASER_IN/PHASER_IN_PHY	0	0	20	0.00
IDELAYE2/IDELAYE2_FINEDELAY	0	0	250	0.00
IBUFDS_GTE2	0	0	2	0.00
ILOGIC	0	0	106	0.00
OLOGIC	0	0	106	0.00

5. Clocking

Site Type	Used	Fixed	Available	Util%
BUFGCTRL	2	0	32	6.25
BUFIO	0	0	20	0.00
MMCME2_ADV	0	0	5	0.00
PLLE2_ADV	0	0	5	0.00
BUFMRCE	0	0	10	0.00
BUFHCE	0	0	72	0.00
BUFR	0	0	20	0.00

6. Specific Feature

Site Type	Used	Fixed	Available	Util%
BSCANE2	0	0	4	0.00
CAPTUREE2	0	0	1	0.00
DNA_PORT	0	0	1	0.00
EFUSE_USR	0	0	1	0.00
FRAME_ECCE2	0	0	1	0.00
ICAPE2	0	0	2	0.00
PCIE_2_1	0	0	1	0.00

| STARTUPE2 | 0 | 0 | 1 | 0.00 |
| XADC | 0 | 0 | 1 | 0.00 |
+-------------+------+-------+-----------+-------+

7. Primitives

+----------+------+---------------------+
| Ref Name | Used | Functional Category |
+----------+------+---------------------+
FDRE	58	Flop & Latch
LUT4	22	LUT
OBUF	17	IO
LUT5	17	LUT
LUT6	13	LUT
FDSE	9	Flop & Latch
LUT3	8	LUT
CARRY4	8	CarryLogic
LUT2	5	LUT
LDCE	3	Flop & Latch
IBUF	2	IO
BUFG	2	Clock
LUT1	1	LUT
+----------+------+---------------------+

8. Black Boxes

+----------+------+
| Ref Name | Used |
+----------+------+

9. Instantiated Netlists

+----------+------+
| Ref Name | Used |
+----------+------+

Conclusion

- In this Lab , a 4 bit nanoprocessor was designed . The design was tested using simulator and demonstrated using the Basys 3 FPGA board . The nanoprocessor can execute instructions such as ADD , MOV , JZR and NEG .
- To reduce the usage of LUT and FlipFlop we minimized the complexity of the nanoprocessor.
- An Instruction is executed in 2 cycles to increase the reliability of the processor . for example , in executing the ADD , R1 ,R2 , we add the values in R1 and R2 in one cycle and store the results in the next cycle .
- The above is implemented using a common count variable in program counter and register bank.
- Another optimized assembly program to add the numbers from 1 to 3 was designed which reduced the usage of LUTs. But the program doesn't demonstrate the use of the JZR and NEG assembly instruction as instructed in the lab sheet.
- Without using 4 DFFs to create a 4-bit register, a single component using behavioral approach was used.
- Muxes were implemented without using the decoders (The behavioral approach was used.)
- We can optimize the design by hardcoding the zero rather than storing it in the R0 register but it was instructed to store the zero vale in the register R0.
- We can optimize the design by using a behavioral approach when incrementing the program counter but it was instructed to use a 3 bit adder.
- In conclusion, the project successfully designed a simple nanoprocessor capable of executing a set of instructions(calculation of total between the numbers 1 and 3). Through the development and extension of various components, the project demonstrated the construction of essential modules for microprocessor design. The project offered hands-on experience and knowledge in digital logic design, VHDL programming, and microprocessor architecture.

Contribution from each member

Building individual components

Team Member	Contribution	Hours spent
Sajeenthiran P. -210553J	Instruction decoder MUX 8 to 1 - 4 bit MUX 2 to 1 -4 -bit MUX 2 to 1 -3 -bit Program counter	10hrs
G.D.C. Samaranayake -:210557B	ROM Register bank 4 bit -Adder/ Substractor 3 bit adder subtractor Slow clock	13 hrs

Assembling and debugging Time

Team Member	Hours spent
Sajeenthiran P. -210553J	13hrs
G.D.C. Samaranayake -:210557B	16hrs