CS1050 - Computer Organization and Digital Design

Lab 9 & 10 Nano Processor Design Competition

Group 24

- MADAELA M.P.G.R.V.M. 230381X
- KESHARA K.P.W.D. 230334H
- SENANAYAKE H.P.V.R. 230595G
- SOMARATHNA M.D.A.M. 230621K

Lab Task - Introduction

The objective of this lab was to design a 4-bit nanoprocessor capable of executing basic instructions using components developed in previous labs. Our team designed and implemented two versions of the processor:

1. Minimal Design – Implements the core required instructions: ADD, SUB, MOVI, and JZR.

2. Extended Design – Adds support for AND, OR, MUL, and a 4-bit comparator, with improvements such as tri-state buffer-based multiplexers and enhanced ALU functionality.

Each version was built with separate modules in VHDL, tested via simulations, and demonstrated on the BASYS 3 board.

Table of Contents

Design with Minimum Qualifications

Features:

- 4-bit signed computation using 2's complement
- Program Counter reset via button
- LEDs display Register R7 output
- 7-Segment Display shows Register R7 value
- Flag LEDs: Sign, Carry, Overflow, Zero

Instruction Set:

- MOVI R, d → Move immediate value
- ADD Ra, Rb → Ra ← Ra + Rb
- NEG R → 2's complement
- JZR R, d → Jump if R == 0

Process => 3 + 2 + 1 = 6

Assembly Code:

```
MOVI R1,4          R1 <- 4
MOVI R2,1          R2 <- 1
NEG  R2            R2 <- -1
ADD  R1,R2         R1 <- R2 + R1
JZR  R1,7
ADD  R7,R1         R7 <- R7 + R1
JZR  R0,3
JZR  R0,7
```

Machine Code:

```
100010000100
100100000001
010100000000
000010100010
110010000111
001110010010
110000000011
110000000111
```

Design Components and Implementations:

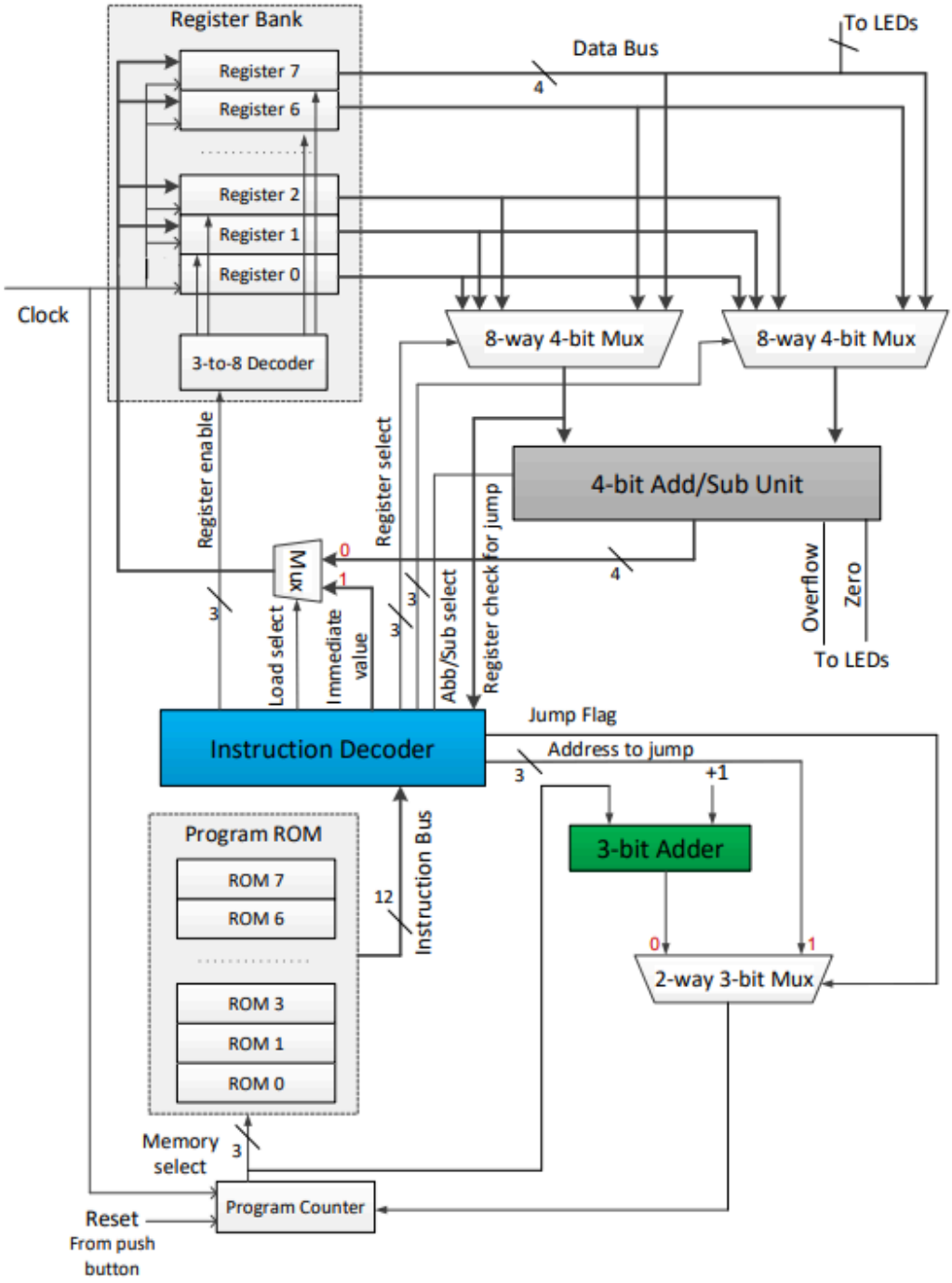
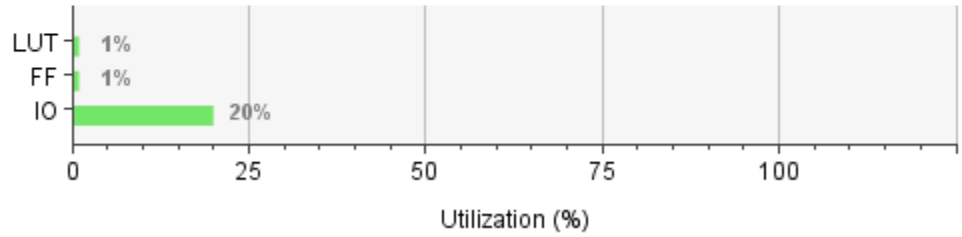


Figure 1 – High-level diagram of the nanoprocessor.

Resource Utilization

Resource	Utilization	Available	Utilization %
LUT	37	20800	0.18
FF	49	41600	0.12
IO	21	106	19.81



1. Nanoprocessor

- VHDL Code:

```
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  -- Uncomment the following library declaration if using
26  -- arithmetic functions with Signed or Unsigned values
27  --use IEEE.NUMERIC_STD.ALL;
28
29  -- Uncomment the following library declaration if instantiating
30  -- any Xilinx leaf cells in this code.
31  --library UNISIM;
32  --use UNISIM.VComponents.all;
33
34  entity Nano_Processor is
35      Port ( Clk : in STD_LOGIC;
36             Reset : in STD_LOGIC;
37             Flags : out STD_LOGIC_VECTOR( 3 downto 0);
38             Dis_LED : out STD_LOGIC_VECTOR (3 downto 0);
39             Dis_7Seg : out STD_LOGIC_VECTOR (6 downto 0);
40             AnodeSelector : out STD_LOGIC_VECTOR (3 downto 0)
41             );
42  end Nano_Processor;
43
44  architecture Behavioral of Nano_Processor is
45
46  component Register_Bank
47      Port (Reg_EN : in STD_LOGIC_VECTOR (2 downto 0);
48            Clk : in STD_LOGIC;
49            MUX_Out : in STD_LOGIC_VECTOR (3 downto 0);
50            Reset : in STD_LOGIC;
51            R0_Out : out STD_LOGIC_VECTOR (3 downto 0);
52            R1_Out : out STD_LOGIC_VECTOR (3 downto 0);
53            R2_Out : out STD_LOGIC_VECTOR (3 downto 0);
54            R3_Out : out STD_LOGIC_VECTOR (3 downto 0);
55            R4_Out : out STD_LOGIC_VECTOR (3 downto 0);
56            R5_Out : out STD_LOGIC_VECTOR (3 downto 0);
57            R6_Out : out STD_LOGIC_VECTOR (3 downto 0);
58            R7_Out : out STD_LOGIC_VECTOR (3 downto 0)
59            );
60  end component;
```

```
61
62  component ALU
63      port (
64            A : in std_logic_vector (3 downto 0);
65            B : in std_logic_vector (3 downto 0);
66            Flag_EN : in std_logic;
67            Selector : in std_logic_vector (2 downto 0);
68            Y : out std_logic_vector (3 downto 0);
69            Flag_Reg : out STD_LOGIC_VECTOR (3 downto 0));
70  end component;
71
72  component Adder_3
73      Port ( A : in STD_LOGIC_VECTOR (2 downto 0);
74             S : out STD_LOGIC_VECTOR (2 downto 0);
75             C_out : out STD_LOGIC);
76  end component;
77
78  component Program_Counter
79      Port ( Clk : in STD_LOGIC;
80             Reset : in STD_LOGIC;
81             Data_Bus: in STD_LOGIC_VECTOR (2 downto 0);
82             Mem_Selector : out STD_LOGIC_VECTOR (2 downto 0));
83  end component;
84
85  component Instruction_Decoder
86      Port ( Instruction_Bus : in STD_LOGIC_VECTOR (11 downto 0);
87             Reg_Check_Jump : in STD_LOGIC_VECTOR (3 downto 0);
88             Reg_Sele1 : out STD_LOGIC_VECTOR (2 downto 0);
89             Reg_Sele2 : out STD_LOGIC_VECTOR (2 downto 0);
90             Immediate_Value : out STD_LOGIC_VECTOR (3 downto 0);
91             Func : out STD_LOGIC_VECTOR (2 downto 0);
92             Load_Sele : out STD_LOGIC;
93             Reg_EN : out STD_LOGIC_VECTOR (2 downto 0);
94             Flag_EN : out STD_LOGIC;
95             Jump_Flag : out STD_LOGIC;
96             Address_to_Jump : out STD_LOGIC_VECTOR (2 downto 0));
97  end component;
98
99  component ROM
100     Port ( address : in STD_LOGIC_VECTOR (2 downto 0);
101            data : out STD_LOGIC_VECTOR (11 downto 0));
102 end component;
103
104 component Mux_2_3
105     Port ( A1 : in STD_LOGIC_VECTOR (2 downto 0);
106            A2 : in STD_LOGIC_VECTOR (2 downto 0);
107            Selector : in STD_LOGIC;
108            Output : out STD_LOGIC_VECTOR (2 downto 0));
109 end component;
110
111 component Mux_2_4
112     Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
113            B : in STD_LOGIC_VECTOR (3 downto 0);
114            Selector : in STD_LOGIC;
115            Mux_out : out STD_LOGIC_VECTOR (3 downto 0));
116 end component;
117
118 component Mux_8_4
119     Port ( A1 : in STD_LOGIC_VECTOR (3 downto 0);
120            A2 : in STD_LOGIC_VECTOR (3 downto 0);
121            A3 : in STD_LOGIC_VECTOR (3 downto 0);
122            A4 : in STD_LOGIC_VECTOR (3 downto 0);
123            A5 : in STD_LOGIC_VECTOR (3 downto 0);
124            A6 : in STD_LOGIC_VECTOR (3 downto 0);
125            A7 : in STD_LOGIC_VECTOR (3 downto 0);
126            A8 : in STD_LOGIC_VECTOR (3 downto 0);
127            Selector : in STD_LOGIC_VECTOR (2 downto 0);
128            Output : out STD_LOGIC_VECTOR (3 downto 0));
129 end component;
```

```
130
131 component LUT is
132     Port ( address : in STD_LOGIC_VECTOR (3 downto 0);
133            data : out STD_LOGIC_VECTOR (6 downto 0));
134 end component;
135
136 component Slow_Clk
137     Port ( Clk_in : in STD_LOGIC;
138            Clk_out : out STD_LOGIC);
139 end component;
140
141 signal Display_out : STD_LOGIC_VECTOR (3 downto 0);
142 signal SlowClk_out : std_logic;
143
144 signal Load_sele, Jump_Flag, Flag_EN_ALU : std_logic;
145
146 signal Mem_Selector, Output_2_3bit_MUX ,Address_to_Jump, Output_3bit_Adder : std_logic_vector(2 downto 0);
147 signal Selector_8_4bit_MUX_1, Selector_8_4bit_MUX_2, Reg_EN: std_logic_vector(2 downto 0);
148
149 signal Instruction_Bus: std_logic_vector(11 downto 0);
150
151 signal Output_8_4bit_MUX_1,Output_8_4bit_MUX_2, Immediate_Value ,Output_2_4bit_MUX : std_logic_vector(3 downto 0);
152 signal R0_Out,R1_Out,R2_Out,R3_Out,R4_Out,R5_Out,R6_Out,R7_Out : std_logic_vector (3 downto 0);
153
154 signal ALU_Output, Flag_Reg  : std_logic_vector (3 downto 0);
155 signal Func  : std_logic_vector (2 downto 0);
156
157
158 begin
159
160     Slow_Clock : Slow_Clk
161         Port map(
162             Clk_in => Clk,
163             Clk_out => SlowClk_out
164          );
165
166     Program_Counter_Unit : Program_Counter
167         Port map (
168             Clk => SlowClk_out,
169             Reset => Reset,
170             Data_Bus => Output_2_3bit_MUX,
171             Mem_Selector => Mem_Selector);
172
173      Program_ROM : ROM
174          Port map (
175             address => Mem_Selector,
176             data => Instruction_Bus);
177
178      Mux_2_3_Unit : Mux_2_3
179          Port map (
180             A1 => Output_3bit_Adder,
181             A2 => Address_to_Jump,
182             Selector => Jump_Flag,
183             Output => Output_2_3bit_MUX);
184
185     Inst_Decoder : Instruction_Decoder
186         Port map (
187             Instruction_Bus  => Instruction_Bus,
188             Reg_Check_Jump => Output_8_4bit_MUX_1,
189             Func => Func,
190             Reg_Sele1 => Selector_8_4bit_MUX_1,
191             Reg_Sele2 => Selector_8_4bit_MUX_2,
192             Immediate_Value => Immediate_Value,
193             Load_Sele => Load_sele,
194             Reg_EN => Reg_EN,
195             Flag_EN => Flag_EN_ALU,
196             Jump_Flag => Jump_Flag,
197             Address_to_Jump => Address_to_Jump);
```

```
198
199     Reg_Bank : Register_Bank
200         Port map (
201             Reg_EN => Reg_EN,
202             Clk => SlowClk_out,
203             MUX_Out => Output_2_4bit_MUX,
204             Reset => Reset,
205             R0_Out => R0_Out,
206             R1_Out => R1_Out,
207             R2_Out => R2_Out,
208             R3_Out => R3_Out,
209             R4_Out => R4_Out,
210             R5_Out => R5_Out,
211             R6_Out => R6_Out,
212             R7_Out => R7_Out
213             );
214
215     Mux_2_4_Unit : Mux_2_4
216         Port map (
217             A => ALU_Output,
218             B => Immediate_Value,
219             Selector => Load_sele,
220             Mux_out => Output_2_4bit_MUX );
221
222      Mux_8_4_1 : Mux_8_4
223         Port map (
224             A1=>R0_Out,
225             A2=>R1_Out,
226             A3=>R2_Out,
227             A4=>R3_Out,
228             A5=>R4_Out,
229             A6=>R5_Out,
230             A7=>R6_Out,
231             A8=>R7_Out,
232             Selector => Selector_8_4bit_MUX_1,
233             Output => Output_8_4bit_MUX_1 );
234
235     Mux_8_4_2 : Mux_8_4
236        Port map (
237            A1=>R0_Out,
238            A2=>R1_Out,
239            A3=>R2_Out,
240            A4=>R3_Out,
241            A5=>R4_Out,
242            A6=>R5_Out,
243            A7=>R6_Out,
244            A8=>R7_Out,
245            Selector => Selector_8_4bit_MUX_2,
246            Output => Output_8_4bit_MUX_2);
247
248     Arithmetic_Logic_Unit : ALU
249        port map(
250            A => Output_8_4bit_MUX_1,
251            B => Output_8_4bit_MUX_2,
252            Selector => Func,
253            Flag_EN => Flag_EN_ALU,
254            Y => ALU_Output,
255            Flag_Reg => Flag_Reg
256            );
257
258
259     Adder_3_bit : Adder_3
260        Port map (
261            A => Mem_Selector,
262            S => Output_3bit_Adder );
```

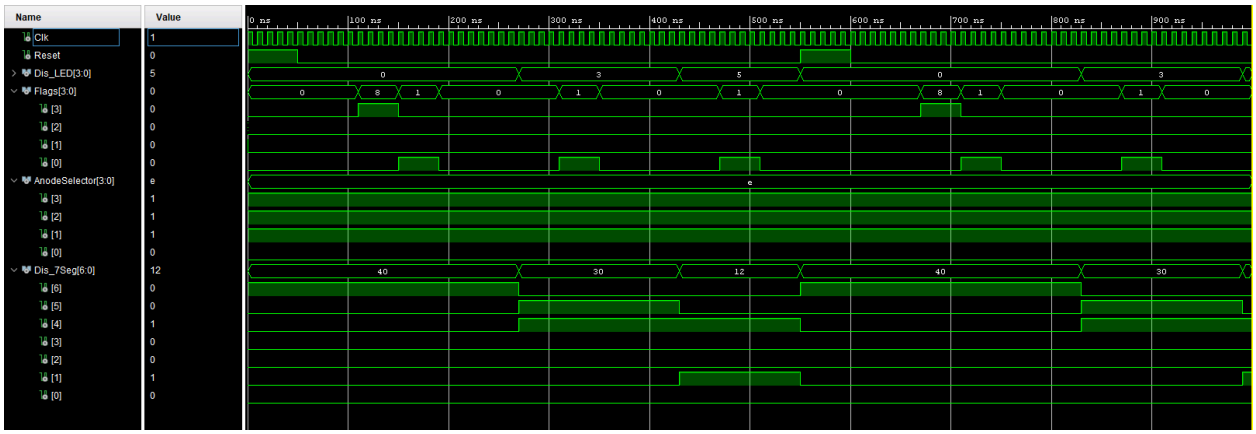
```
263
264         LUT_Unit : LUT
265            port map(
266                address => Display_out,
267                data => Dis_7Seg
268             );
269
270    Display_out <= R7_Out;
271    Flags <= Flag_Reg;
272    Dis_LED <= Display_out;
273    AnodeSelector <= "1110";
274
275    end Behavioral;
276
```

- Testbench Code:

```
22   library IEEE;
23   use IEEE.STD_LOGIC_1164.ALL;
24
25   -- Uncomment the following library declaration if using
26   -- arithmetic functions with Signed or Unsigned values
27   --use IEEE.NUMERIC_STD.ALL;
28
29   -- Uncomment the following library declaration if instantiating
30   -- any Xilinx leaf cells in this code.
31   --library UNISIM;
32   --use UNISIM.VComponents.all;
33
34   entity TB_Nano_Processor is
35   --  Port ( );
36   end TB_Nano_Processor;
37
38   architecture Behavioral of TB_Nano_Processor is
39
40   component Nano_Processor
41           Port (  Clk : in STD_LOGIC;
42                   Reset : in STD_LOGIC;
43                   Flags : out STD_LOGIC_VECTOR( 3 downto 0);
44                   Dis_LED : out STD_LOGIC_VECTOR (3 downto 0);
45                   Dis_7Seg : out STD_LOGIC_VECTOR (6 downto 0);
46                   AnodeSelector : out STD_LOGIC_VECTOR (3 downto 0)
47                 );
48   end component;
49
50   signal Clk : std_logic := '0';
51   signal Reset : std_logic;
52   signal Dis_LED, Flags, AnodeSelector : std_logic_vector (3 downto 0);
53   signal Dis_7Seg : std_logic_vector (6 downto 0);
54
55   begin
56       UUT : Nano_Processor
57           port map (
58               Clk => Clk,
59               Reset => Reset,
60               Dis_LED => Dis_LED,
61               Dis_7Seg => Dis_7Seg,
62               Flags => Flags,
63               AnodeSelector => AnodeSelector
64            );
65
```

```
65
66  process begin
67      Clk <= not Clk;
68      wait for 5 ns;
69  end process;
70
71  process begin
72
73      Reset <= '1';
74      wait for 50 ns;
75
76      Reset <= '0';
77      wait for 500 ns;
78
79      Reset <= '1';
80      wait for 50 ns;
81
82      Reset <= '0';
83      wait;
84  end process;
85
86  end Behavioral;
87
```

- Timing Diagram:



2. Program Counter

- VHDL Code:

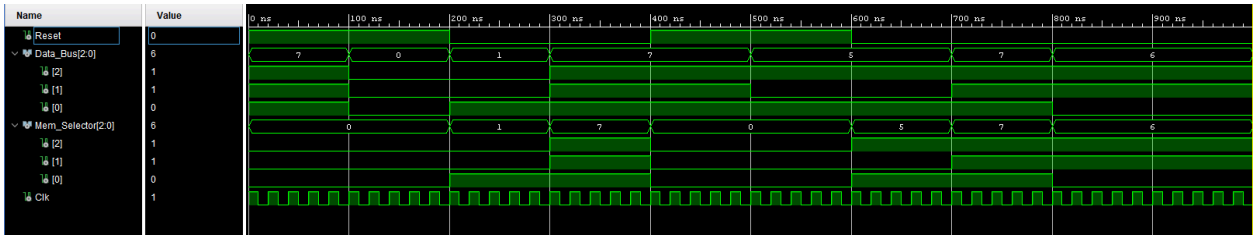
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- 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 ( Clk : in STD_LOGIC;
           Reset : in STD_LOGIC;
           Data_Bus: in STD_LOGIC_VECTOR (2 downto 0);
           Mem_Selector : out STD_LOGIC_VECTOR (2 downto 0));
end Program_Counter;

architecture Behavioral of Program_Counter is

component D_FF
    Port ( D : in STD_LOGIC;
           Res : in STD_LOGIC;
           Clk : in STD_LOGIC;
           Q : out STD_LOGIC;
           Qbar : out STD_LOGIC);
end component;

begin

    D_FF_0 : D_FF
        port map(
            D => Data_Bus(0),
            Res => Reset,
            Clk => Clk,
            Q => Mem_Selector(0)
        );

    D_FF_1 : D_FF
        port map(
            D => Data_Bus(1),
            Res => Reset,
            Clk => Clk,
            Q => Mem_Selector(1)
        );

    D_FF_2 : D_FF
           port map(
           D => Data_Bus(2),
           Res => Reset,
           Clk => Clk,
           Q => Mem_Selector(2)
        );

end Behavioral;
```

- Testbench Code:

```
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  -- Uncomment the following library declaration if using
26  -- arithmetic functions with Signed or Unsigned values
27  --use IEEE.NUMERIC_STD.ALL;
28
29  -- Uncomment the following library declaration if instantiating
30  -- any Xilinx leaf cells in this code.
31  --library UNISIM;
32  --use UNISIM.VComponents.all;
33
34  entity TB_Program_Counter is
35  --  Port ( );
36  end TB_Program_Counter;
37
38  architecture Behavioral of TB_Program_Counter is
39
40  component Program_Counter
41      Port ( Clk : in STD_LOGIC;
42             Reset : in STD_LOGIC;
43             Data_Bus: in STD_LOGIC_VECTOR (2 downto 0);
44             Mem_Selector : out STD_LOGIC_VECTOR (2 downto 0));
45  end component;
46
47  signal  Reset : std_logic;
48  signal Data_Bus, Mem_Selector : std_logic_vector (2 downto 0);
49  signal Clk : std_logic := '0';
50  begin
51  UUT : Program_Counter port map (
52          Clk => Clk,
53          Reset => Reset,
54          Data_Bus => Data_Bus,
55          Mem_Selector => Mem_Selector
56        );
57
58  process begin
59      Clk <= not Clk;
60      wait for 10 ns;
61  end process;
62
63  process begin
64
65  --Index Numbers
66      --230381    111 000 001 111 111 101
67      --230334    111 000 001 101 111 110
68      --230595    111 000 010 111 000 011
69      --230621    111 000 011 000 110 101
70
71      Reset <= '1';
72
73      --230381    111 000 001 111 111 101  using this index for the following test cases
74      Data_Bus <= "111";
75      wait for 100 ns;
76
77      Data_Bus <= "000";
78      wait for 100 ns;
79  -------------------------------------------------------
80      Reset <= '0';
81
82      Data_Bus <= "001";
83      wait for 100 ns;
84
85      Data_Bus <= "111";
86      wait for 100 ns;
87  -------------------------------------------------------
88      Reset <= '1';
89
90      Data_Bus <= "111";
91      wait for 100 ns;
92
93      Data_Bus <= "101";
94      wait for 100 ns;
95  -------------------------------------------------------
```

```
96      Reset <= '0';
97
98      --230334    111 000 001 101 111  using this index for the following test cases
99      Data_Bus <= "101";
100     wait for 100 ns;
101
102     Data_Bus <= "111";
103     wait for 100 ns;
104
105     Data_Bus <= "110";
106     wait ;
107 end process;
108 end Behavioral;
```

- Timing Diagram:



3. Instruction Decoder

- VHDL Code

```
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  -- Uncomment the following library declaration if using
26  -- arithmetic functions with Signed or Unsigned values
27  --use IEEE.NUMERIC_STD.ALL;
28
29  -- Uncomment the following library declaration if instantiating
30  -- any Xilinx leaf cells in this code.
31  --library UNISIM;
32  --use UNISIM.VComponents.all;
33
34  entity Instruction_Decoder is
35      Port ( Instruction_Bus : in STD_LOGIC_VECTOR (11 downto 0);
36             Reg_Check_Jump : in STD_LOGIC_VECTOR (3 downto 0);
37             Reg_Sele1 : out STD_LOGIC_VECTOR (2 downto 0);
38             Reg_Sele2 : out STD_LOGIC_VECTOR (2 downto 0);
39             Immediate_Value : out STD_LOGIC_VECTOR (3 downto 0);
40             Func : out STD_LOGIC_VECTOR (2 downto 0);
41             Load_Sele : out STD_LOGIC;
42             Reg_EN : out STD_LOGIC_VECTOR (2 downto 0);
43             Flag_EN : out STD_LOGIC; -- Enable the Flags in ALU
44             Comp_EN : out STD_LOGIC; -- Enable the Comparator in ALU
45             Jump_Flag : out STD_LOGIC;
46             Address_to_Jump : out STD_LOGIC_VECTOR (2 downto 0));
47  end Instruction_Decoder;
48
49  architecture Behavioral of Instruction_Decoder is
50
51  signal Operator : std_logic_vector (1 downto 0);
52  signal FuncValue : std_logic_vector (2 downto 0);
53
```

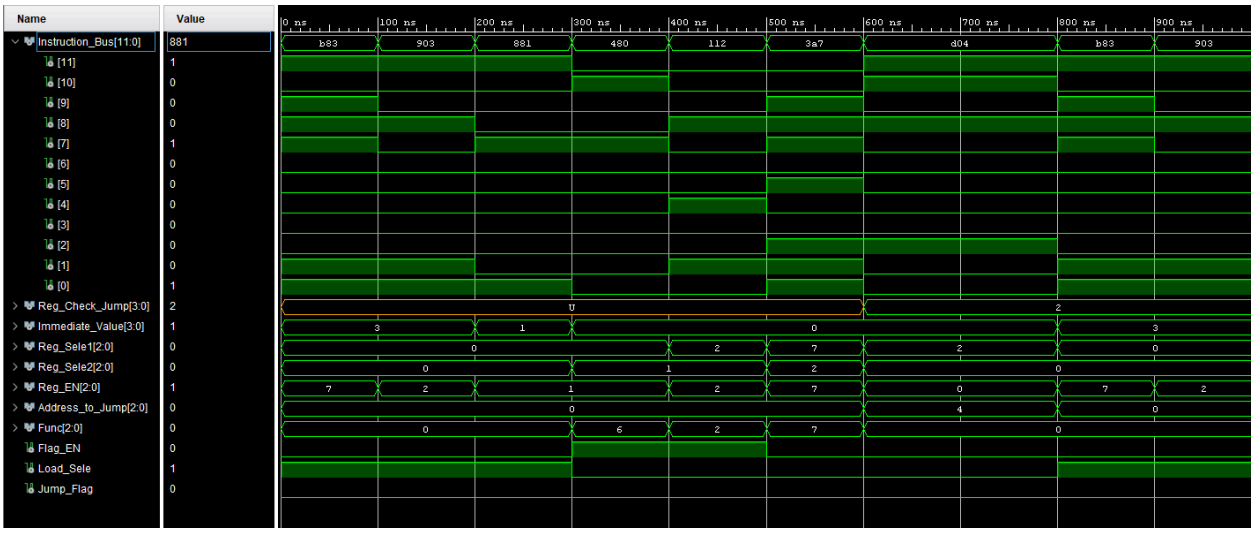
```
54  begin
55      Operator <= Instruction_Bus(11 downto 10);
56
57      process (Operator, Instruction_Bus, Reg_Check_Jump, FuncValue) begin
58          Jump_Flag <= '0';
59          Flag_EN <= '0';
60          Comp_EN <= '0';
61          Load_Sele <= '0';
62          Reg_Sele1 <= "000";
63          Reg_Sele2 <= "000";
64          Immediate_Value <= "0000";
65          Reg_EN <= "000";
66          Address_to_Jump <= "000";
67          Func <= "000";
68
69          FuncValue <= Instruction_Bus(2 downto 0);
70
71          if Operator = "00" then --add
72              Comp_EN <= '1';
73              Reg_Sele1 <= Instruction_Bus(9 downto 7);
74              Reg_Sele2 <= Instruction_Bus(6 downto 4);
75              Reg_EN <= Instruction_Bus(9 downto 7);
76              Func <= FuncValue;
77              if ( FuncValue = "010" or FuncValue = "110" ) then
78                  Flag_EN <= '1';
79              end if;
80
81          elsif Operator = "01" then --sub
82              Comp_EN <= '1';
83              Reg_EN <= Instruction_Bus(9 downto 7);
84              Flag_EN <= '1';
85              Reg_Sele2 <= Instruction_Bus(9 downto 7);
86              Func <= "110";
87
88          elsif Operator = "10" then --move
89
90              Immediate_Value <= Instruction_Bus(3 downto 0);
91              Reg_EN <= Instruction_Bus(9 downto 7);
92              Load_Sele <= '1';
93
94          elsif Operator = "11" then --jump
95              Address_to_Jump <= Instruction_Bus(2 downto 0);
96              Reg_Sele1 <= Instruction_Bus(9 downto 7);
97              if Reg_Check_Jump = "0000" then
98                  Jump_Flag <= '1';
99              end if;
100
101         end if;
102     end process;
103 end Behavioral;
104
```

- Testbench Code

```
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  -- Uncomment the following library declaration if using
26  -- arithmetic functions with Signed or Unsigned values
27  --use IEEE.NUMERIC_STD.ALL;
28
29  -- Uncomment the following library declaration if instantiating
30  -- any Xilinx leaf cells in this code.
31  --library UNISIM;
32  --use UNISIM.VComponents.all;
33
34  entity TB_Instruction_Decoder is
35  --  Port ( );
36  end TB_Instruction_Decoder;
37
38  architecture Behavioral of TB_Instruction_Decoder is
39
40  component Instruction_Decoder
41      Port ( Instruction_Bus : in STD_LOGIC_VECTOR (11 downto 0);
42             Reg_Check_Jump : in STD_LOGIC_VECTOR (3 downto 0);
43             Reg_Sele1 : out STD_LOGIC_VECTOR (2 downto 0);
44             Reg_Sele2 : out STD_LOGIC_VECTOR (2 downto 0);
45             Immediate_Value : out STD_LOGIC_VECTOR (3 downto 0);
46             Func : out STD_LOGIC_VECTOR (2 downto 0);
47             Load_Sele : out STD_LOGIC;
48             Reg_EN : out STD_LOGIC_VECTOR (2 downto 0);
49             Flag_EN : out STD_LOGIC; -- Enable the Flags in ALU
50             Comp_EN : out STD_LOGIC; -- Enable the Comparator in ALU
51             Jump_Flag : out STD_LOGIC;
52             Address_to_Jump : out STD_LOGIC_VECTOR (2 downto 0));
53  end component;
54
55  signal Instruction_Bus : STD_LOGIC_VECTOR (11 downto 0);
56  signal Reg_Check_Jump, Immediate_Value : STD_LOGIC_VECTOR (3 downto 0);
57  signal Reg_Sele1, Reg_Sele2, Reg_EN, Address_to_Jump, Func : STD_LOGIC_VECTOR (2 downto 0);
58  signal Flag_EN, Load_Sele, Jump_Flag : STD_LOGIC;
59
60  begin
61  UUT : Instruction_Decoder port map(
62         Instruction_Bus => Instruction_Bus,
63         Reg_Check_Jump => Reg_Check_Jump,
64         Flag_EN => Flag_EN,
65         Func => Func,
66         Reg_Sele1 => Reg_Sele1,
67         Reg_Sele2 => Reg_Sele2,
68         Immediate_Value => Immediate_Value,
69         Load_Sele => Load_Sele,
70         Reg_EN => Reg_EN,
71         Jump_Flag => Jump_Flag,
72         Address_to_Jump => Address_to_Jump
73      );
74
75
76  stimulus_process: process
77           begin
78
79           Instruction_Bus <= "101110000011";
80           wait for 100 ns;
81
82           Instruction_Bus <= "100100000011";
83           wait for 100 ns;
84
85           Instruction_Bus <= "100010000001";
86           wait for 100 ns;
87
88           Instruction_Bus <= "010010000000";
89           wait for 100 ns;
90
```

```
91          Instruction_Bus <= "000100010010";
92          wait for 100 ns;
93
94          Instruction_Bus <= "001110100111";
95          wait for 100 ns;
96
97          Instruction_Bus <= "110100000100";
98          Reg_Check_Jump <= "0010";
99          wait for 100 ns;
100
101         Reg_Check_Jump <= "0010";
102         wait for 100 ns;
103
104         end process stimulus_process;
105
106         end Behavioral;
107
```

- Timing Diagram



4. ALU

- VHDL Code

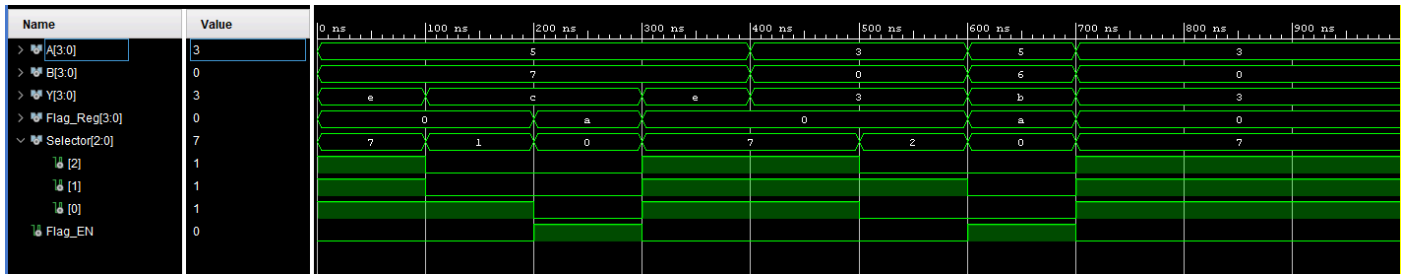
```
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  -- Uncomment the following library declaration if using
26  -- arithmetic functions with Signed or Unsigned values
27  --use IEEE.NUMERIC_STD.ALL;
28
29  -- Uncomment the following library declaration if instantiating
30  -- any Xilinx leaf cells in this code.
31  --library UNISIM;
32  --use UNISIM.VComponents.all;
33
34  entity ALU is
35      port (
36            A : in std_logic_vector (3 downto 0);
37            B : in std_logic_vector (3 downto 0);
38            Flag_EN : in std_logic;
39            Selector : in std_logic_vector (2 downto 0);
40            Y : out std_logic_vector (3 downto 0);
41            Flag_Reg : out STD_LOGIC_VECTOR (3 downto 0)
42            );
43  end ALU;
44
45  architecture Behavioral of ALU is
46
47
48  component Add_Sub_4
49      Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
50             B : in STD_LOGIC_VECTOR (3 downto 0);
51             M : in STD_LOGIC;
52             S : out STD_LOGIC_VECTOR (3 downto 0);
53             Flag_Reg : out STD_LOGIC_VECTOR (3 downto 0));
54  end component;
55
56  signal A_plus_B : std_logic_vector (3 downto 0);
57  signal Flags : std_logic_vector (3 downto 0);
58
59  begin
60
61      ADD_SUB_UNIT : Add_Sub_4
62          port map(
63              A => A,
64              B => B,
65              M => Selector(2),
66              S => A_plus_B,
67              Flag_Reg => Flags
68          );
69
70
71
72  Flag_Reg(0) <= Flags(0) AND Flag_EN;
73  Flag_Reg(1) <= Flags(1) AND Flag_EN;
74  Flag_Reg(2) <= Flags(2) AND Flag_EN;
75  Flag_Reg(3) <= Flags(3) AND Flag_EN;
76  Y <= A_plus_B;
77
78  end Behavioral;
```

- Testbench Code

```
22      library IEEE;
23      use IEEE.STD_LOGIC_1164.ALL;
24
25      -- Uncomment the following library declaration if using
26      -- arithmetic functions with Signed or Unsigned values
27      --use IEEE.NUMERIC_STD.ALL;
28
29      -- Uncomment the following library declaration if instantiating
30      -- any Xilinx leaf cells in this code.
31      --library UNISIM;
32      --use UNISIM.VComponents.all;
33
34      entity TB_ALU is
35      --  Port ( );
36      end TB_ALU;
37
38      architecture Behavioral of TB_ALU is
39
40      component ALU
41          port (
42                  A : in std_logic_vector (3 downto 0);
43                  B : in std_logic_vector (3 downto 0);
44                  Flag_EN : in std_logic;
45                  Selector : in std_logic_vector (2 downto 0);
46                  Y : out std_logic_vector (3 downto 0);
47                  Flag_Reg : out STD_LOGIC_VECTOR (3 downto 0)
48                );
49      end component;
50
51      signal A, B, Y, Flag_Reg : STD_LOGIC_VECTOR (3 downto 0);
52      signal Selector : STD_LOGIC_VECTOR (2 downto 0);
53      signal Flag_EN : STD_LOGIC;
54
55      begin
56          UUT : ALU
57              port map(
58                 A => A,
59                 B => B,
60                 Flag_EN => Flag_EN,
61                 Selector => Selector,
62                 Y => Y,
63                 Flag_Reg => Flag_Reg
64               );
65
66      process begin
67
68      --Index Numbers
69          --230381    111 000 001 111 111 101
70          --230334    111 000 001 101 111 110
71          --230595    111 000 010 111 000 011
72          --230621    111 000 011 000 110 101
73
74      --230381    111 000 001 111 111 101   using this for the following test cases
75          A <= "0101";
76          B <= "0111";
77          Flag_EN <= '0';
78
79          Selector <= "111";
80          wait for 100 ns;
81
82          Selector <= "001";
83          wait for 100 ns;
84
85          Flag_EN <= '1';
86          Selector <= "000";
87          wait for 100 ns;
88
89          Flag_EN <= '0';
90          Selector <= "111";
91          wait for 100 ns;
92
```

```
93      --230595    111 000 010 111 000 011   using this for the following test cases
94          A <= "0011";
95          B <= "0000";
96          Selector <= "111";
97          wait for 100 ns;
98
99          Selector <= "010";
100         wait for 100 ns;
101
102     --230621    111 000 011 000 110 101   using this for the following test cases
103         A <= "0101";
104         B <= "0110";
105         Flag_EN <= '1';
106         Selector <= "000";
107         wait for 100 ns;
108
109         A <= "0011";
110         B <= "0000";
111         Flag_EN <= '0';
112         Selector <= "111";
113         wait;
114
115     end process;
116
117     end Behavioral;
```

- Timing Diagram



5. 4-Bit Register

- VHDL Code

```
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  -- Uncomment the following library declaration if using
26  -- arithmetic functions with Signed or Unsigned values
27  --use IEEE.NUMERIC_STD.ALL;
28
29  -- Uncomment the following library declaration if instantiating
30  -- any Xilinx leaf cells in this code.
31  --library UNISIM;
32  --use UNISIM.VComponents.all;
33
34  entity Register_4 is
35      Port ( D : in STD_LOGIC_VECTOR (3 downto 0);
36             EN : in STD_LOGIC;
37             Clk : in STD_LOGIC;
38             Reset : in STD_LOGIC;
39             Q : out STD_LOGIC_VECTOR (3 downto 0));
40  end Register_4;
41
42  architecture Behavioral of Register_4 is
43
44  begin
45
46      process (Clk,EN,Reset) begin
47          if Reset = '0' then
48              if (rising_edge(Clk)) then
49                  if EN = '1' then
50                      Q <= D;
51                  end if;
52              end if;
53          else
54              Q <= "0000";
55          end if;
56      end process;
57
58  end Behavioral;
59
```

5. Register Bank

- VHDL Code

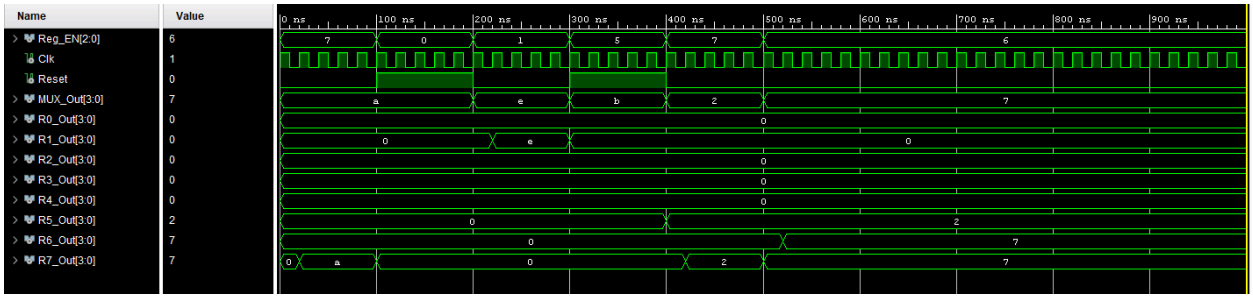
```
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  -- Uncomment the following library declaration if using
26  -- arithmetic functions with Signed or Unsigned values
27  --use IEEE.NUMERIC_STD.ALL;
28
29  -- Uncomment the following library declaration if instantiating
30  -- any Xilinx leaf cells in this code.
31  --library UNISIM;
32  --use UNISIM.VComponents.all;
33
34  entity Register_Bank is
35      Port (Reg_EN : in STD_LOGIC_VECTOR (2 downto 0);
36            Clk : in STD_LOGIC;
37            MUX_Out : in STD_LOGIC_VECTOR (3 downto 0);
38            Reset : in STD_LOGIC;
39            R0_Out : out STD_LOGIC_VECTOR (3 downto 0);
40            R1_Out : out STD_LOGIC_VECTOR (3 downto 0);
41            R2_Out : out STD_LOGIC_VECTOR (3 downto 0);
42            R3_Out : out STD_LOGIC_VECTOR (3 downto 0);
43            R4_Out : out STD_LOGIC_VECTOR (3 downto 0);
44            R5_Out : out STD_LOGIC_VECTOR (3 downto 0);
45            R6_Out : out STD_LOGIC_VECTOR (3 downto 0);
46            R7_Out : out STD_LOGIC_VECTOR (3 downto 0)
47            );
48  end Register_Bank;
49
50  architecture Behavioral of Register_Bank is
51
52  component Decoder_3_8
53      Port ( I : in STD_LOGIC_VECTOR (2 downto 0);
54             EN : in STD_LOGIC;
55             Y : out STD_LOGIC_VECTOR (7 downto 0));
56  end component;
57
58  component Register_4 is
59      Port ( D : in STD_LOGIC_VECTOR (3 downto 0);
60             EN : in STD_LOGIC;
61             Clk : in STD_LOGIC;
62             Reset : in STD_LOGIC;
63             Q : out STD_LOGIC_VECTOR (3 downto 0));
64  end component;
65
66  SIGNAL Select_Reg : STD_LOGIC_VECTOR (7 downto 0);
67
68  begin
69
70      Decoder_3_8_0 : Decoder_3_8
71            Port map( I=>Reg_EN,
72                      EN=>'1',
73                      Y=>Select_Reg );
74
75      R0 : Register_4
76            Port map ( D=>"0000",
77                  EN=>Select_Reg(0),
78                  Clk=>Clk,
79                  Reset=>Reset,
80                  Q=> R0_Out );
81
82      R1 : Register_4
83           Port map ( D=>MUX_Out,
84                  EN=>Select_Reg(1),
85                  Clk=>Clk,
86                  Reset=>Reset,
87                  Q=> R1_Out );
88
```

```
89      R2 : Register_4
90           Port map ( D=>MUX_Out,
91                  EN=>Select_Reg(2),
92                  Clk=>Clk,
93                  Reset=>Reset,
94                  Q=> R2_Out );
95
96      R3 : Register_4
97           Port map ( D=>MUX_Out,
98                  EN=>Select_Reg(3),
99                  Clk=>Clk,
100                 Reset=>Reset,
101                 Q=> R3_Out );
102
103     R4 : Register_4
104          Port map ( D=>MUX_Out,
105                 EN=>Select_Reg(4),
106                 Clk=>Clk,
107                 Reset=>Reset,
108                 Q=> R4_Out );
109
110     R5 : Register_4
111          Port map ( D=>MUX_Out,
112                 EN=>Select_Reg(5),
113                 Clk=>Clk,
114                 Reset=>Reset,
115                 Q=> R5_Out );
116
117     R6 : Register_4
118          Port map ( D=>MUX_Out,
119                 EN=>Select_Reg(6),
120                 Clk=>Clk,
121                 Reset=>Reset,
122                 Q=> R6_Out );
123
124     R7 : Register_4
125          Port map ( D=>MUX_Out,
126                 EN=>Select_Reg(7),
127                 Clk=>Clk,
128                 Reset=>Reset,
129                 Q=> R7_Out );
130
131 end Behavioral;
132
```

- Testbench Code

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

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

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

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

architecture Behavioral of TB_Register_Bank is

component Register_Bank is
    Port (Reg_EN : in STD_LOGIC_VECTOR (2 downto 0);
          Clk : in STD_LOGIC;
          MUX_Out : in STD_LOGIC_VECTOR (3 downto 0);
          Reset : in STD_LOGIC;
          R0_Out : out STD_LOGIC_VECTOR (3 downto 0);
          R1_Out : out STD_LOGIC_VECTOR (3 downto 0);
          R2_Out : out STD_LOGIC_VECTOR (3 downto 0);
          R3_Out : out STD_LOGIC_VECTOR (3 downto 0);
          R4_Out : out STD_LOGIC_VECTOR (3 downto 0);
          R5_Out : out STD_LOGIC_VECTOR (3 downto 0);
          R6_Out : out STD_LOGIC_VECTOR (3 downto 0);
          R7_Out : out STD_LOGIC_VECTOR (3 downto 0));
end component;

SIGNAL Reg_EN : STD_LOGIC_VECTOR (2 downto 0);
SIGNAL Clk,Reset :  STD_LOGIC;
SIGNAL MUX_Out,R0_Out,R1_Out,R2_Out,R3_Out,R4_Out,R5_Out,R6_Out,R7_Out :  STD_LOGIC_VECTOR (3 downto 0);

begin
UUT : Register_Bank
    Port map(Reg_EN => Reg_EN,
          Clk => Clk,
          MUX_Out => MUX_Out,
          Reset => Reset,
          R0_Out => R0_Out,
          R1_Out => R1_Out,
          R2_Out => R2_Out,
          R3_Out => R3_Out,
          R4_Out => R4_Out,
          R5_Out => R5_Out,
          R6_Out => R6_Out,
          R7_Out => R7_Out);

process begin
    Clk <= '1';
    wait for 10 ns;
    Clk <='0';
    wait for 10 ns;
end process;

process begin

--Index Numbers
    --230381    111 000 001 111 111 101
    --230334    111 000 001 101 111 110
    --230595    111 000 010 111 000 011
    --230621    111 000 011 000 110 101

--230334    111 000 001 101 111 110  using this for the following test cases
    Reg_EN <= "111";
    Reset <= '0';
    MUX_Out <= "1010";
    wait for 100ns;

    Reg_EN <= "000";
    Reset <= '1';
    MUX_Out <= "1010";
    wait for 100ns;

```

- Timing Diagram



5. 3-Bit Adder

- VHDL Code

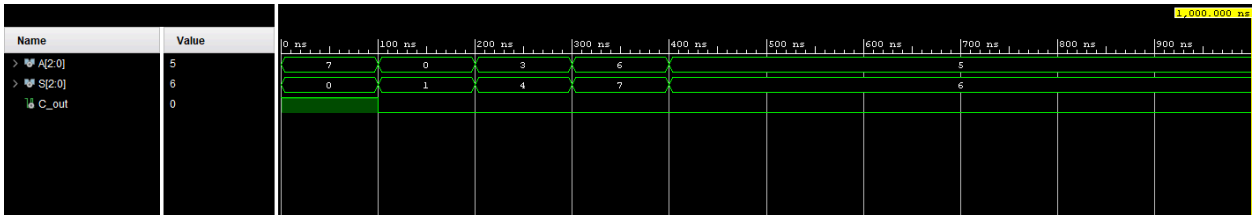
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity Adder_3 is
    Port ( A : in STD_LOGIC_VECTOR (2 downto 0);
           S : out STD_LOGIC_VECTOR (2 downto 0);
           C_out : out STD_LOGIC);
end Adder_3;

architecture Behavioral of Adder_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 C1,C2,C3 : std_logic;
signal OutSum : std_logic_vector (2 downto 0);
signal B : std_logic := '1';

begin
    FA_0 : FA
        port map(
            A => A(0),
            B => B,
            C_in => '0',
            S => OutSum(0),
            C_out => C1
        );

    FA_1 : FA
        port map(
            A => A(1),
            B => '0',
            C_in => C1,
            S => OutSum(1),
            C_out => C2
        );

```

```
    FA_2 : FA
        port map(
            A => A(2),
            B => '0',
            C_in => C2,
            S => OutSum(2),
            C_out => C3
        );

S <= OutSum;
C_out <= C3;


end Behavioral;
```

- Testbench Code

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

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

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

entity TB_Adder_3 is
--  Port ( );
end TB_Adder_3;

architecture Behavioral of TB_Adder_3 is

component Adder_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;

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

begin
UUT : Adder_3 port map(
        A => A,
        S => S,
        C_out => C_out
    );

process begin

    --230621    111 000 011 000 110 101 using this for the following test cases
   A <= "111";
   wait for 100 ns;

   A <= "000";
   wait for 100 ns;

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

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

   A <= "101";
   wait;

end process;

end Behavioral;
```

- Timing Diagram



.6. Add/Sub Unit

- VHDL Code

```
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_4 is
    Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
           B : in STD_LOGIC_VECTOR (3 downto 0);
           M : in STD_LOGIC;
           S : out STD_LOGIC_VECTOR (3 downto 0);
           Flag_Reg : out STD_LOGIC_VECTOR (3 downto 0));
end Add_Sub_4;

architecture Behavioral of Add_Sub_4 is

component FA
    Port ( A : in STD_LOGIC;
           B : in STD_LOGIC;
           C_in : in STD_LOGIC;
           S : out STD_LOGIC;
           C_out : out STD_LOGIC);
end component;
signal BM, OutSum : std_logic_vector(3 downto 0);
signal C1,C2,C3,C4, OverFlow : std_logic;


begin

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

    FA_0 : FA
        port map(
            A => A(0),
            B => BM(0),
            C_in => M,
            S => OutSum(0),
            C_out => C1
        );

```

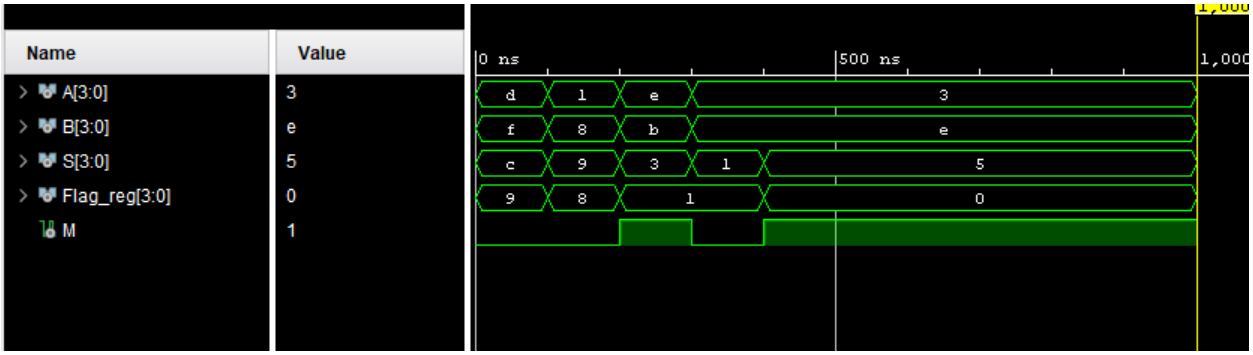
```
    FA_1 : FA
        port map(
            A => A(1),
            B => BM(1),
            C_in => C1,
            S => OutSum(1),
            C_out => C2
        );

    FA_2 : FA
        port map(
            A => A(2),
            B => BM(2),
            C_in => C2,
            S => OutSum(2),
            C_out => C3
        );

    FA_3 : FA
        port map(
            A => A(3),
            B => BM(3),
            C_in => C3,
            S => OutSum(3),
            C_out => C4
        );

        S <= OutSum ;
        OverFlow <= C4 XOR C3;

        Flag_Reg(0) <= C4;                                                          --C
        Flag_Reg(1) <= OverFlow;                                                    --O
        Flag_Reg(2) <= NOT (OutSum(0) OR OutSum(1) OR OutSum(2) OR OutSum(3));      --Z
        Flag_Reg(3) <= OutSum(3);                                                   --S

end Behavioral;
```

- Testbench Code

```
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_4 is
--  Port ( );
end TB_Add_Sub_4;

architecture Behavioral of TB_Add_Sub_4 is

component Add_Sub_4
    Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
           B : in STD_LOGIC_VECTOR (3 downto 0);
           M : in STD_LOGIC;
           S : out STD_LOGIC_VECTOR (3 downto 0);
--           Overflow_Flag : out STD_LOGIC;
--           C_out : out STD_LOGIC;
--           Zero_Flag : out STD_LOGIC;
--           Sign_Flag : out STD_LOGIC
           Flag_Reg : out STD_LOGIC_VECTOR (3 downto 0));
end component;

--signal A,B,S : std_logic_vector (3 downto 0);
--signal M, Overflow_Flag, C_out, Zero_Flag, Sign_Flag: std_logic;
signal A,B,S, Flag_reg : std_logic_vector (3 downto 0);
signal M: std_logic;

begin
    UUT: Add_Sub_4 port map(
        A => A,
        B => B,
        M => M,
        S => S,
        Flag_Reg => Flag_Reg
        );
```

```
66      process begin
67
68      --   Index Numbers
69      --    230381    11 1000 0001 1111 1101
70      --    230334    11 1000 0001 1011 1110
71      --    230595    11 1000 0010 1110 0011
72      --    230621    11 1000 0011 0001 1011
73
74          A <= "1101";
75          B <= "1111";
76          M <= '0';
77          wait for 100 ns;
78
79          A <= "0001";
80          B <= "1000";
81          M <= '0';
82          wait for 100 ns;
83
84          A <= "1110";
85          B <= "1011";
86          M <= '1';
87          wait for 100 ns;
88
89          A <= "0011";
90          B <= "1110";
91          M <= '0';
92          wait for 100 ns;
93
94          A <= "0011";
95          B <= "1110";
96          M <= '1';
97          wait ;
98
99      end process;
100     end Behavioral;
```

- Timing Diagram



7. Multiplexers

2-way 3-bit MUX

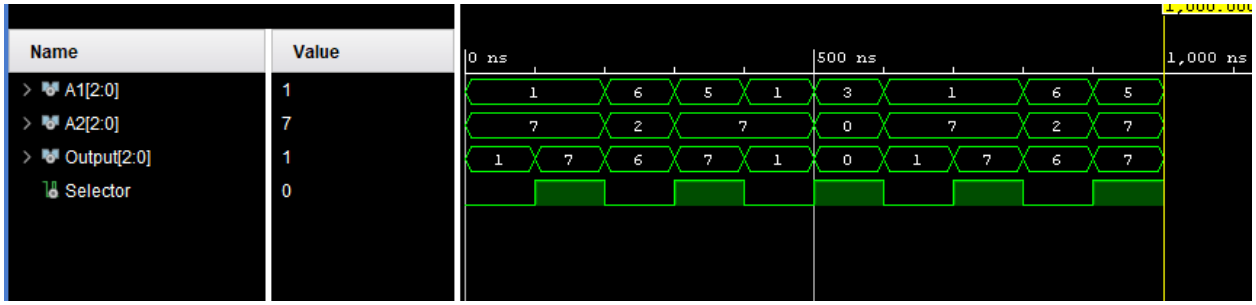
- VHDL code

```
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_3 is
    Port ( A1 : in STD_LOGIC_VECTOR (2 downto 0);
           A2 : in STD_LOGIC_VECTOR (2 downto 0);
           Selector : in STD_LOGIC;
           Output : out STD_LOGIC_VECTOR (2 downto 0));
end Mux_2_3;

architecture Behavioral of Mux_2_3 is

begin

    Output <= A1 when Selector = '0' else A2;

end Behavioral;

```

- Testbench code

```
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_2_3 is
--  Port ( );
end TB_Mux_2_3;

architecture Behavioral of TB_Mux_2_3 is

component Mux_2_3
    Port ( A1 : in STD_LOGIC_VECTOR (2 downto 0);
           A2 : in STD_LOGIC_VECTOR (2 downto 0);
           Selector : in STD_LOGIC;
           Output : out STD_LOGIC_VECTOR (2 downto 0));
end component;

signal A1, A2, Output : std_logic_vector (2 downto 0);
signal Selector : std_logic;

begin
UUT : Mux_2_3 port map(
        A1 => A1,
        A2 => A2,
        Selector => Selector,
        Output => Output
    );

```

- Timing diagram



2-way 4-bit MUX

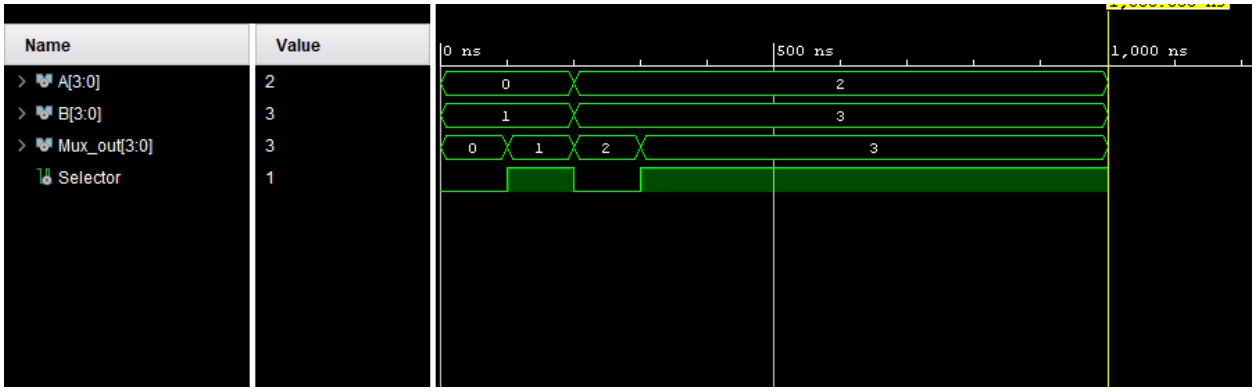
- VHDL code

```
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  -- Uncomment the following library declaration if using
26  -- arithmetic functions with Signed or Unsigned values
27  --use IEEE.NUMERIC_STD.ALL;
28
29  -- Uncomment the following library declaration if instantiating
30  -- any Xilinx leaf cells in this code.
31  --library UNISIM;
32  --use UNISIM.VComponents.all;
33
34  entity Mux_2_4 is
35      Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
36             B : in STD_LOGIC_VECTOR (3 downto 0);
37             Selector : in STD_LOGIC;
38             Mux_out : out STD_LOGIC_VECTOR (3 downto 0));
39  end Mux_2_4;
40
41  architecture Behavioral of Mux_2_4 is
42
43  begin
44
45      Mux_out <= A when Selector = '0' else B;
46
47  end Behavioral;
48
```

- Testbench code

```
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_2_4 is
--  Port ( );
end TB_Mux_2_4;

architecture Behavioral of TB_Mux_2_4 is

component Mux_2_4
    Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
           B : in STD_LOGIC_VECTOR (3 downto 0);
           Selector : in STD_LOGIC;
           Mux_out : out STD_LOGIC_VECTOR (3 downto 0));
end component;

signal A,B,Mux_out:std_logic_vector(3 downto 0);
signal Selector: std_logic;

begin
    UUT:Mux_2_4
        PORT MAP(
        A=>A,
        B=>B,
        Mux_out=>Mux_out,
        Selector=>Selector
    );

process begin

    A<="0000";
    B<="0001";
    Selector<='0';
    wait for 100ns;

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

    A<="0010";
    B<="0011";
    Selector<='0';
    wait for 100ns;

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

    wait;
end process;

end Behavioral;
```

- Timing diagram



8-way 4-bit MUX

- VHDL code

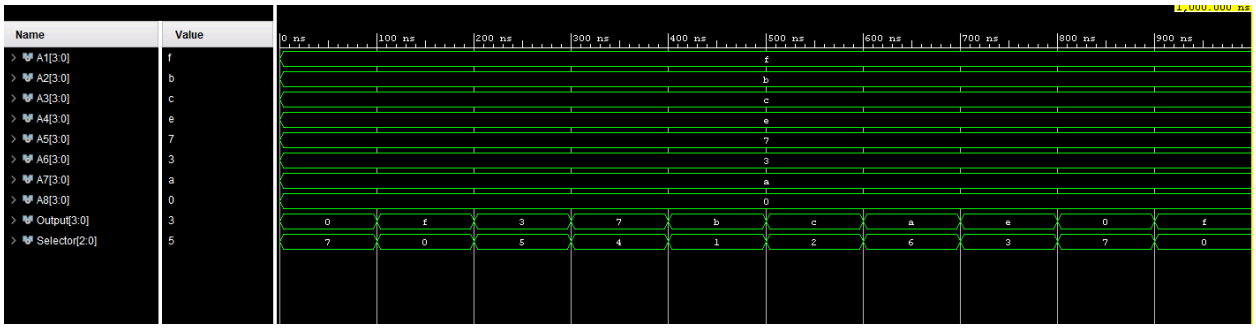
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- 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_4 is
    Port (
        A1 : in STD_LOGIC_VECTOR (3 downto 0);
        A2 : in STD_LOGIC_VECTOR (3 downto 0);
        A3 : in STD_LOGIC_VECTOR (3 downto 0);
        A4 : in STD_LOGIC_VECTOR (3 downto 0);
        A5 : in STD_LOGIC_VECTOR (3 downto 0);
        A6 : in STD_LOGIC_VECTOR (3 downto 0);
        A7 : in STD_LOGIC_VECTOR (3 downto 0);
        A8 : in STD_LOGIC_VECTOR (3 downto 0);
        Selector : in STD_LOGIC_VECTOR (2 downto 0);
        Output : out STD_LOGIC_VECTOR (3 downto 0)
        );
end Mux_8_4;

architecture Behavioral of Mux_8_4 is


begin

  process(Selector, A1, A2, A3, A4, A5, A6, A7, A8)
   begin
       case Selector is
           when "000" => Output <= A1;
           when "001" => Output <= A2;
           when "010" => Output <= A3;
           when "011" => Output <= A4;
           when "100" => Output <= A5;
           when "101" => Output <= A6;
           when "110" => Output <= A7;
           when others => Output <= A8; -- covers "111"
       end case;
   end process;
end Behavioral;
```

- Testbench code

```
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_4 is
--  Port ( );
end TB_Mux_8_4;

architecture Behavioral of TB_Mux_8_4 is

component Mux_8_4
    Port (
        A1 : in STD_LOGIC_VECTOR (3 downto 0);
        A2 : in STD_LOGIC_VECTOR (3 downto 0);
        A3 : in STD_LOGIC_VECTOR (3 downto 0);
        A4 : in STD_LOGIC_VECTOR (3 downto 0);
        A5 : in STD_LOGIC_VECTOR (3 downto 0);
        A6 : in STD_LOGIC_VECTOR (3 downto 0);
        A7 : in STD_LOGIC_VECTOR (3 downto 0);
        A8 : in STD_LOGIC_VECTOR (3 downto 0);
        Selector : in STD_LOGIC_VECTOR (2 downto 0);
        Output : out STD_LOGIC_VECTOR (3 downto 0)
        );
end component;

signal A1,A2,A3,A4,A5,A6,A7,A8,Output : std_logic_vector (3 downto 0);
signal Selector : std_logic_vector (2 downto 0);

begin
UUT : Mux_8_4 port map(
         A1 => A1,
         A2 => A2,
         A3 => A3,
         A4 => A4,
         A5 => A5,
         A6 => A6,
         A7 => A7,
         A8 => A8,
         Selector => Selector,
         Output => Output
         );

process begin
    A1 <= "1111";
    A2 <= "1011";
    A3 <= "1100";
    A4 <= "1110";
    A5 <= "0111";
    A6 <= "0011";
    A7 <= "1010";
    A8 <= "0000";

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

    Selector <= "000";
    wait for 100 ns;

    Selector <= "101";
    wait for 100 ns;

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

```
93
94          Selector <= "001";
95          wait for 100 ns;
96
97          Selector <= "010";
98          wait for 100 ns;
99
100         Selector <= "110";
101         wait for 100 ns;
102
103         Selector <= "011";
104         wait for 100 ns;
105
106     end process;
107     end Behavioral;
108
```

- Timing diagram



8. Program ROM

- VHDL code

```
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  use ieee.numeric_std.all;
26
27  -- Uncomment the following library declaration if using
28  -- arithmetic functions with Signed or Unsigned values
29  --use IEEE.NUMERIC_STD.ALL;
30
31  -- Uncomment the following library declaration if instantiating
32  -- any Xilinx leaf cells in this code.
33  --library UNISIM;
34  --use UNISIM.VComponents.all;
35
36  entity ROM is
37      Port ( address : in STD_LOGIC_VECTOR (2 downto 0);
38             data : out STD_LOGIC_VECTOR (11 downto 0));
39  end ROM;
40
41  architecture Behavioral of ROM is
42
43  type rom_type is array (0 to 7) of std_logic_vector(11 downto 0);
44  signal prosseser_ROM : rom_type := (
45
```

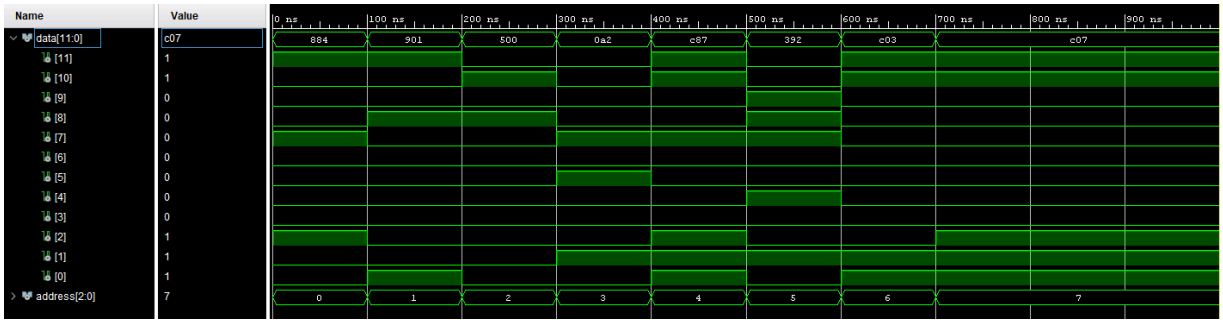
```
--        process => 3 + 2 + 1 = 6
          "100010000100", -- MOVI R1,4        R1 <- 4            0
          "100100000001", -- MOVI R2,1        R2 <- 1            1
          "010100000000", -- NEG  R2          R2 <- -1           2
          "000010100010", -- ADD  R1,R2       R1 <- R2 + R1      3
          "110010000111", -- JZR  R1,7                           4
          "001110010010", -- ADD  R7,R1       R7 <- R7 + R1      5
          "110000000011", -- JZR  R0,3                           6
          "110000000111" -- JZR  R0,7                            7

    );

begin

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

end Behavioral;

```

- Testbench code

```
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);
           data : out STD_LOGIC_VECTOR (11 downto 0));
end component;

signal data:std_logic_vector(11 downto 0);
signal address:std_logic_vector(2 downto 0);

begin
    UUT: ROM
        PORT MAP(
        address=>address,
        data=>data
    );

process begin

    address<="000";
    wait for 100ns;

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

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

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

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

```

```
71
72      address<="101";
73      wait for 100ns;
74
75      address<="110";
76      wait for 100ns;
77
78      address<="111";
79      wait ;
80
81  end process;
82  end Behavioral;
83
```

- Timing diagram



9. 3 to 8 Decoder

- VHDL code

```
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  -- Uncomment the following library declaration if using
26  -- arithmetic functions with Signed or Unsigned values
27  --use IEEE.NUMERIC_STD.ALL;
28
29  -- Uncomment the following library declaration if instantiating
30  -- any Xilinx leaf cells in this code.
31  --library UNISIM;
32  --use UNISIM.VComponents.all;
33
34  entity Decoder_3_8 is
35      Port ( I : in STD_LOGIC_VECTOR (2 downto 0);
36             EN : in STD_LOGIC;
37             Y : out STD_LOGIC_VECTOR (7 downto 0));
38  end Decoder_3_8;
39
40  architecture Behavioral of Decoder_3_8 is
41  component Decoder_2_4
42     Port ( I : in STD_LOGIC_VECTOR (1 downto 0);
43            EN : in STD_LOGIC;
44            Y : out STD_LOGIC_VECTOR (3 downto 0));
45  end component;
46
47  SIGNAL input_vec : std_logic_vector (1 downto 0);
48  SIGNAL out_vec_1,out_vec_2 : std_logic_vector (3 downto 0);
49  SIGNAL EN_dec1, EN_dec2 : std_logic;
50
51  begin
52      EN_dec1 <= EN and not(I(2));
53      EN_dec2 <= EN and I(2);
54      input_vec<= I(1 downto 0);
55
```

```
    Decoder_2_4_0: Decoder_2_4
        port map(I=>input_vec,
                EN=>EN_dec1,
                Y => out_vec_1);
    Decoder_4_1: Decoder_2_4
        port map(I=>input_vec,
                EN=>EN_dec2,
                Y => out_vec_2);

    Y(3 downto 0)<= out_vec_1;
    Y(7 downto 4)<= out_vec_2;

end Behavioral;
```

10. LUT (7 Segment Display)

- VHDL code

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

architecture Behavioral of LUT 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;
```

11. Clock Generator (Slow Clock)

- VHDL code

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

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

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

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

architecture Behavioral of Slow_Clk is

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

begin

    process (Clk_in) begin
        if (rising_edge(Clk_in)) then
            count <= count + 1;
            --if (count = 50000000) then
            if (count = 2) then
                clk_status <= not clk_status;
                Clk_out <= clk_status;
                count <= 1;
            end if;
        end if;
    end process;


end Behavioral;
```

12. Constraint 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}]

## Switches
set_property PACKAGE_PIN V17 [get_ports {Reset}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Reset}]


## LEDs
set_property PACKAGE_PIN U16 [get_ports {Dis_LED[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_LED[0]}]
set_property PACKAGE_PIN E19 [get_ports {Dis_LED[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_LED[1]}]
set_property PACKAGE_PIN U19 [get_ports {Dis_LED[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_LED[2]}]
set_property PACKAGE_PIN V19 [get_ports {Dis_LED[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_LED[3]}]

set_property PACKAGE_PIN P3 [get_ports {Flags[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Flags[3]}]
set_property PACKAGE_PIN N3 [get_ports {Flags[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Flags[1]}]
set_property PACKAGE_PIN P1 [get_ports {Flags[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Flags[2]}]
set_property PACKAGE_PIN L1 [get_ports {Flags[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Flags[0]}]

##7 segment display
set_property PACKAGE_PIN W7 [get_ports {Dis_7Seg[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_7Seg[0]}]
set_property PACKAGE_PIN W6 [get_ports {Dis_7Seg[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_7Seg[1]}]
set_property PACKAGE_PIN U8 [get_ports {Dis_7Seg[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_7Seg[2]}]
set_property PACKAGE_PIN V8 [get_ports {Dis_7Seg[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_7Seg[3]}]
set_property PACKAGE_PIN U5 [get_ports {Dis_7Seg[4]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_7Seg[4]}]
set_property PACKAGE_PIN V5 [get_ports {Dis_7Seg[5]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_7Seg[5]}]
set_property PACKAGE_PIN U7 [get_ports {Dis_7Seg[6]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_7Seg[6]}]

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


set_property CFGBVS VCCO [current_design]
set_property CONFIG_VOLTAGE 3.3 [current_design]

```

Design with Extended Features

Added Features:

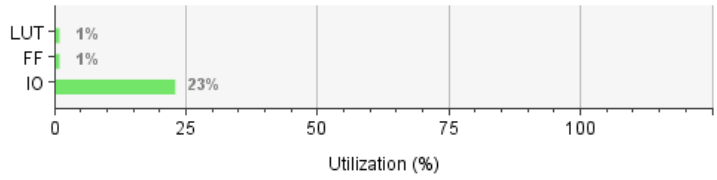
- AND, OR, MUL support
- Comparator with output flags (Equal, Greater Than, Less Than)
- Tri-State Buffer-based MUXes
- Optimized ALU for multiple operations
- Extra LEDs for Comparator Flags

Additional Instructions:

- AND Ra, Rb
- OR Ra, Rb
- MUL Ra, Rb

Resource Utilization

Resource	Utilization	Available	Utilization %
LUT	89	20800	0.43
FF	45	41600	0.11
IO	24	106	22.64



Process => 3 MUL 2 = 6

0101110000011 -- MOVI R7,3 R7 <- 3

0100010000010 -- MOVI R1,2 R1 <- 2

1101110010000 -- MUL R7,R1 R7 <- R7 MUL R1

0110000000011 -- JZR R0,3

Design Components and Implementations

1. Extended Nanoprocessor

- VHDL Code

```
21
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  -- Uncomment the following library declaration if using
26  -- arithmetic functions with Signed or Unsigned values
27  --use IEEE.NUMERIC_STD.ALL;
28
29  -- Uncomment the following library declaration if instantiating
30  -- any Xilinx leaf cells in this code.
31  --library UNISIM;
32  --use UNISIM.VComponents.all;
33
34  entity Nano_Processor is
35      Port ( Clk : in STD_LOGIC;
36             Reset : in STD_LOGIC;
37             Flags : out STD_LOGIC_VECTOR( 3 downto 0);
38             Dis_LED : out STD_LOGIC_VECTOR (3 downto 0);
39             Dis_7Seg : out STD_LOGIC_VECTOR (6 downto 0);
40             Comparator_out : out STD_LOGIC_VECTOR (2 downto 0);
41             AnodeSelector : out STD_LOGIC_VECTOR (3 downto 0)
42             );
43  end Nano_Processor;
```

```
45  architecture Behavioral of Nano_Processor is
46
47  component Register_Bank
48      Port (Reg_EN : in STD_LOGIC_VECTOR (2 downto 0);
49            Clk : in STD_LOGIC;
50            MUX_Out : in STD_LOGIC_VECTOR (3 downto 0);
51            Reset : in STD_LOGIC;
52            R0_Out : out STD_LOGIC_VECTOR (3 downto 0);
53            R1_Out : out STD_LOGIC_VECTOR (3 downto 0);
54            R2_Out : out STD_LOGIC_VECTOR (3 downto 0);
55            R3_Out : out STD_LOGIC_VECTOR (3 downto 0);
56            R4_Out : out STD_LOGIC_VECTOR (3 downto 0);
57            R5_Out : out STD_LOGIC_VECTOR (3 downto 0);
58            R6_Out : out STD_LOGIC_VECTOR (3 downto 0);
59            R7_Out : out STD_LOGIC_VECTOR (3 downto 0)
60            );
61  end component;
62
63  component ALU
64      port (
65            A : in std_logic_vector (3 downto 0);
66            B : in std_logic_vector (3 downto 0);
67            Flag_EN : in std_logic;
68            Comp_EN : in std_logic;
69            Selector : in std_logic_vector (2 downto 0);
70            Y : out std_logic_vector (3 downto 0);
71            Flag_Reg : out STD_LOGIC_VECTOR (3 downto 0);
72            equal : out STD_LOGIC;
73            greater : out STD_LOGIC;
74            lesser : out STD_LOGIC
75            );
76  end component;
77
78  component Adder_3
79      Port ( A : in STD_LOGIC_VECTOR (2 downto 0);
80             S : out STD_LOGIC_VECTOR (2 downto 0);
81             C_out : out STD_LOGIC);
82  end component;
```

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

component Instruction_Decoder
    Port ( Instruction_Bus : in STD_LOGIC_VECTOR (12 downto 0);
           Reg_Check_Jump : in STD_LOGIC_VECTOR (3 downto 0);
           Reg_Sele1 : out STD_LOGIC_VECTOR (2 downto 0);
           Reg_Sele2 : out STD_LOGIC_VECTOR (2 downto 0);
           Immediate_Value : out STD_LOGIC_VECTOR (3 downto 0);
           Func : out STD_LOGIC_VECTOR (2 downto 0);
           Load_Sele : out STD_LOGIC;
           Reg_EN : out STD_LOGIC_VECTOR (2 downto 0);
           Flag_EN : out STD_LOGIC;
           Comp_EN : out STD_LOGIC;
           Jump_Flag : out STD_LOGIC;
           Address_to_Jump : out STD_LOGIC_VECTOR (2 downto 0));
end component;

component ROM
    Port ( address : in STD_LOGIC_VECTOR (2 downto 0);
           data : out STD_LOGIC_VECTOR (12 downto 0));
end component;

component Mux_2_3
    Port ( A1 : in STD_LOGIC_VECTOR (2 downto 0);
           A2 : in STD_LOGIC_VECTOR (2 downto 0);
           Selector : in STD_LOGIC;
           Output : out STD_LOGIC_VECTOR (2 downto 0));
end component;

component Mux_2_4
    Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
           B : in STD_LOGIC_VECTOR (3 downto 0);
           Selector : in STD_LOGIC;
           Mux_out : out STD_LOGIC_VECTOR (3 downto 0));
end component;
```

```
component Mux_8_4
    Port ( A1 : in STD_LOGIC_VECTOR (3 downto 0);
           A2 : in STD_LOGIC_VECTOR (3 downto 0);
           A3 : in STD_LOGIC_VECTOR (3 downto 0);
           A4 : in STD_LOGIC_VECTOR (3 downto 0);
           A5 : in STD_LOGIC_VECTOR (3 downto 0);
           A6 : in STD_LOGIC_VECTOR (3 downto 0);
           A7 : in STD_LOGIC_VECTOR (3 downto 0);
           A8 : in STD_LOGIC_VECTOR (3 downto 0);
           Selector : in STD_LOGIC_VECTOR (2 downto 0);
           Output : out STD_LOGIC_VECTOR (3 downto 0));
end component;

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

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

signal Display_out : STD_LOGIC_VECTOR (3 downto 0);
signal SlowClk_out : std_logic;

signal Load_sele, Jump_Flag, Flag_EN_ALU, Comp_EN_ALU : std_logic;

signal Mem_Selector, Output_2_3bit_MUX ,Address_to_Jump, Output_3bit_Adder : std_logic_vector(2 downto 0);
signal Selector_8_4bit_MUX_1, Selector_8_4bit_MUX_2, Reg_EN: std_logic_vector(2 downto 0);

signal Instruction_Bus: std_logic_vector(12 downto 0);

signal Output_8_4bit_MUX_1,Output_8_4bit_MUX_2, Immediate_Value ,Output_2_4bit_MUX : std_logic_vector(3 downto 0);
signal R0_Out,R1_Out,R2_Out,R3_Out,R4_Out,R5_Out,R6_Out,R7_Out : std_logic_vector (3 downto 0);

signal ALU_Output, Flag_Reg  : std_logic_vector (3 downto 0);
signal Func  : std_logic_vector (2 downto 0);
```

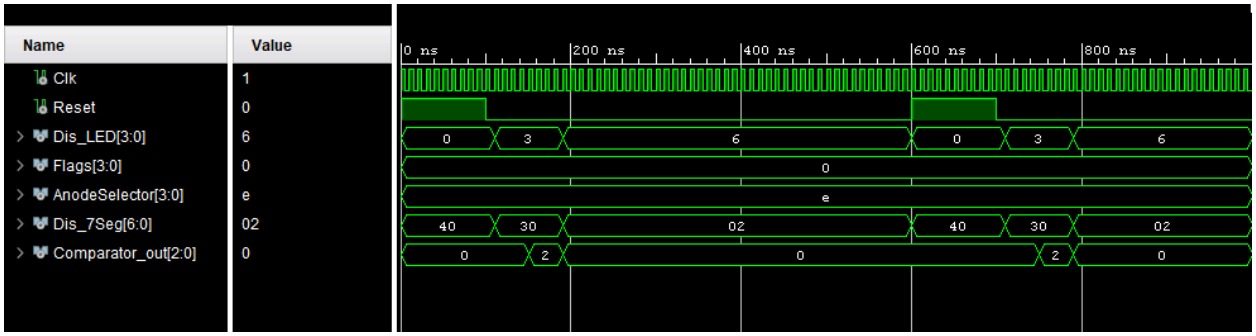
```
164  begin
165
166      Slow_Clock : Slow_Clk
167          Port map(
168              Clk_in => Clk,
169              Clk_out => SlowClk_out
170           );
171
172      Program_Counter_Unit : Program_Counter
173          Port map (
174              Clk => SlowClk_out,
175              Reset => Reset,
176              Data_Bus => Output_2_3bit_MUX,
177              Mem_Selector => Mem_Selector);
178
179       Program_ROM : ROM
180           Port map (
181              address => Mem_Selector,
182              data => Instruction_Bus);
183
184       Mux_2_3_Unit : Mux_2_3
185           Port map (
186              A1 => Output_3bit_Adder,
187              A2 => Address_to_Jump,
188              Selector => Jump_Flag,
189              Output => Output_2_3bit_MUX);
191      Inst_Decoder : Instruction_Decoder
192          Port map (
193              Instruction_Bus  => Instruction_Bus,
194              Reg_Check_Jump => Output_8_4bit_MUX_1,
195              Func => Func,
196              Reg_Sele1 => Selector_8_4bit_MUX_1,
197              Reg_Sele2 => Selector_8_4bit_MUX_2,
198              Immediate_Value => Immediate_Value,
199              Load_Sele => Load_sele,
200              Reg_EN => Reg_EN,
201              Flag_EN => Flag_EN_ALU,
202              Comp_EN => Comp_EN_ALU,
203              Jump_Flag => Jump_Flag,
204              Address_to_Jump => Address_to_Jump);
205
206      Reg_Bank : Register_Bank
207          Port map (
208              Reg_EN => Reg_EN,
209              Clk => SlowClk_out,
210              MUX_Out => Output_2_4bit_MUX,
211              Reset => Reset,
212              R0_Out => R0_Out,
213              R1_Out => R1_Out,
214              R2_Out => R2_Out,
215              R3_Out => R3_Out,
216              R4_Out => R4_Out,
217              R5_Out => R5_Out,
218              R6_Out => R6_Out,
219              R7_Out => R7_Out
220              );
221
222      Mux_2_4_Unit : Mux_2_4
223          Port map (
224              A => ALU_Output,
225              B => Immediate_Value,
226              Selector => Load_sele,
227              Mux_out => Output_2_4bit_MUX );
```

```
229     Mux_8_4_1 : Mux_8_4
230        Port map (
231            A1=>R0_Out,
232            A2=>R1_Out,
233            A3=>R2_Out,
234            A4=>R3_Out,
235            A5=>R4_Out,
236            A6=>R5_Out,
237            A7=>R6_Out,
238            A8=>R7_Out,
239            Selector => Selector_8_4bit_MUX_1,
240            Output => Output_8_4bit_MUX_1 );
241
242     Mux_8_4_2 : Mux_8_4
243        Port map (
244            A1=>R0_Out,
245            A2=>R1_Out,
246            A3=>R2_Out,
247            A4=>R3_Out,
248            A5=>R4_Out,
249            A6=>R5_Out,
250            A7=>R6_Out,
251            A8=>R7_Out,
252            Selector => Selector_8_4bit_MUX_2,
253            Output => Output_8_4bit_MUX_2);
254
255     Arithmetic_Logic_Unit : ALU
256        port map(
257            A => Output_8_4bit_MUX_1,
258            B => Output_8_4bit_MUX_2,
259            Selector => Func,
260            Flag_EN => Flag_EN_ALU,
261            Comp_EN => Comp_EN_ALU,
262            Y => ALU_Output,
263            Flag_Reg => Flag_Reg,
264            equal => Comparator_out(0),
265            greater => Comparator_out(1),
266            lesser => Comparator_out(2));
267
268
269     Adder_3_bit : Adder_3
270        Port map (
271            A => Mem_Selector,
272            S => Output_3bit_Adder );
273
274     LUT_Unit : LUT
275        port map(
276            address => Display_out,
277            data => Dis_7Seg
278         );
279
280 Display_out <= R7_Out;
281 Flags <= Flag_Reg;
282 Dis_LED <= Display_out;
283 AnodeSelector <= "1110";
284
285 end Behavioral;
286
```

- Testbench Code

```
21
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  -- Uncomment the following library declaration if using
26  -- arithmetic functions with Signed or Unsigned values
27  --use IEEE.NUMERIC_STD.ALL;
28
29  -- Uncomment the following library declaration if instantiating
30  -- any Xilinx leaf cells in this code.
31  --library UNISIM;
32  --use UNISIM.VComponents.all;
33
34  entity TB_Nano_Processor is
35  --  Port ( );
36  end TB_Nano_Processor;
37
38  architecture Behavioral of TB_Nano_Processor is

40  component Nano_Processor
41          Port (  Clk : in STD_LOGIC;
42                  Reset : in STD_LOGIC;
43                  Flags : out STD_LOGIC_VECTOR( 3 downto 0);
44                  Dis_LED : out STD_LOGIC_VECTOR (3 downto 0);
45                  Dis_7Seg : out STD_LOGIC_VECTOR (6 downto 0);
46                  Comparator_out : out STD_LOGIC_VECTOR (2 downto 0);
47                  AnodeSelector : out STD_LOGIC_VECTOR (3 downto 0)
48                 );
49  end component;
50
51  signal Clk : std_logic := '0';
52  signal Reset : std_logic;
53  signal Dis_LED, Flags, AnodeSelector : std_logic_vector (3 downto 0);
54  signal Dis_7Seg : std_logic_vector (6 downto 0);
55  signal Comparator_out : std_logic_vector (2 downto 0);
56
57  begin
58      UUT : Nano_Processor
59          port map (
60              Clk => Clk,
61              Reset => Reset,
62              Dis_LED => Dis_LED,
63              Dis_7Seg => Dis_7Seg,
64              Comparator_out => Comparator_out,
65              Flags => Flags,
66              AnodeSelector => AnodeSelector
67           );
68
69  process begin
70      Clk <= not Clk;
71      wait for 5 ns;
72  end process;
73
74  process begin
75      Clk <= not Clk;
76      wait for 5 ns;
77  end process;

79  process begin
80      Reset <= '1';
81      wait for 100 ns;
82
83      Reset <= '0';
84      wait for 500 ns;
85
86      Reset <= '1';
87      wait for 100 ns;
88
89      Reset <= '0';
90      wait;
91  end process;
92
93  end Behavioral;
```

- Timing Diagram



2. ALU with Arithmetic & Logic

- ADD, SUB, AND, OR, MUL operations

```
21
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  -- Uncomment the following library declaration if using
26  -- arithmetic functions with Signed or Unsigned values
27  --use IEEE.NUMERIC_STD.ALL;
28
29  -- Uncomment the following library declaration if instantiating
30  -- any Xilinx leaf cells in this code.
31  --library UNISIM;
32  --use UNISIM.VComponents.all;
33
34  entity ALU is
35      port (
36            A : in std_logic_vector (3 downto 0);
37            B : in std_logic_vector (3 downto 0);
38            Flag_EN : in std_logic;
39            Comp_EN : in std_logic;
40            Selector : in std_logic_vector (2 downto 0);
41            Y : out std_logic_vector (3 downto 0);
42            Flag_Reg : out STD_LOGIC_VECTOR (3 downto 0);
43            equal : out STD_LOGIC;
44            greater : out STD_LOGIC;
45            lesser : out STD_LOGIC
46            );
47  end ALU;
48
49  architecture Behavioral of ALU is
50
51  component Comp_4 is
52      Port ( num1 : in STD_LOGIC_VECTOR(3 downto 0);
53             num2 : in STD_LOGIC_VECTOR(3 downto 0);
54             EN : in STD_LOGIC;
55             equal : out STD_LOGIC;
56             greater : out STD_LOGIC;
57             lesser : out STD_LOGIC );
58  end component;
59
60  component Mux_2_4
61      Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
62             B : in STD_LOGIC_VECTOR (3 downto 0);
63             Selector : in STD_LOGIC;
64             Mux_out : out STD_LOGIC_VECTOR (3 downto 0));
65  end component;
66
67  component Mux_4_4
68      Port (
69         A1 : in STD_LOGIC_VECTOR (3 downto 0);
70         A2 : in STD_LOGIC_VECTOR (3 downto 0);
71         A3 : in STD_LOGIC_VECTOR (3 downto 0);
72         A4 : in STD_LOGIC_VECTOR (3 downto 0);
73         Selector : in STD_LOGIC_VECTOR (1 downto 0);
74         Output : out STD_LOGIC_VECTOR (3 downto 0));
75  end component;
```

```
76
77  component Add_Sub_4
78      Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
79             B : in STD_LOGIC_VECTOR (3 downto 0);
80             M : in STD_LOGIC;
81             S : out STD_LOGIC_VECTOR (3 downto 0);
82             Flag_Reg : out STD_LOGIC_VECTOR (3 downto 0));
83  end component;
84
85  component Multiplier_4
86      Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
87             B : in STD_LOGIC_VECTOR (3 downto 0);
88             Y : out STD_LOGIC_VECTOR (3 downto 0));
89  end component;

91  signal not_B : std_logic_vector (3 downto 0);
92  signal Mux_1_out, Mux_2_out : std_logic_vector (3 downto 0);
93  signal A_AND_B : std_logic_vector (3 downto 0);
94  signal A_OR_B : std_logic_vector (3 downto 0);
95  signal A_plus_B : std_logic_vector (3 downto 0);
96  signal A_into_B : std_logic_vector (3 downto 0);
97  signal Flags : std_logic_vector (3 downto 0);
98
99  begin
100     not_B <= NOT B;
101
102     MUX_1 : Mux_2_4
103         port map(
104             A => B,
105             B => not_B,
106             Selector => Selector(2),
107             Mux_out => Mux_1_out
108         );
109
110     A_AND_B <= Mux_1_out AND A;
111     A_OR_B <= Mux_1_out OR A;
112
113     ADD_SUB_UNIT : Add_Sub_4
114         port map(
115             A => A,
116             B => B,
117             M => Selector(2),
118             S => A_plus_B,
119             Flag_Reg => Flags
120         );
121
122
123      Multiplier_Unit : Multiplier_4
124         port map (
125             A => A(3 downto 0),
126             B => B(3 downto 0),
127             Y => A_into_B (3 downto 0)
128          );
129
130      MUX_2 : Mux_4_4
131         port map(
132             A1 => A_AND_B,
133             A2 => A_OR_B,
134             A3 => A_plus_B,
135             A4 => A_into_B,
136             Selector => Selector(1 downto 0),
137             Output => Mux_2_out
138          );
139
140      Comparator : Comp_4
141         port map(
142             num1 => A,
143             num2=>B,
144             EN => Comp_EN,
145             equal => equal,
146             greater => greater,
147             lesser => lesser);
148
149
150 Flag_Reg(0) <= Flags(0) AND Flag_EN;
151 Flag_Reg(1) <= Flags(1) AND Flag_EN;
152 Flag_Reg(2) <= Flags(2) AND Flag_EN;
153 Flag_Reg(3) <= Flags(3) AND Flag_EN;
154 Y <= Mux_2_out;
155
156 end Behavioral;
157
```

3. Comparator Unit

- Flags: Equal, Greater than, Less than

```
21
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  -- Uncomment the following library declaration if using
26  -- arithmetic functions with Signed or Unsigned values
27  --use IEEE.NUMERIC_STD.ALL;
28
29  -- Uncomment the following library declaration if instantiating
30  -- any Xilinx leaf cells in this code.
31  --library UNISIM;
32  --use UNISIM.VComponents.all;
33
34  entity Comp_4 is
35      Port ( num1 : in STD_LOGIC_VECTOR(3 downto 0);
36             num2 : in STD_LOGIC_VECTOR(3 downto 0);
37             EN : in STD_LOGIC;
38             equal : out STD_LOGIC;       --num1 = num2
39             greater : out STD_LOGIC;     --num1 > num2
40             lesser : out STD_LOGIC );    --num1 < num2
41  end Comp_4;
42
43  architecture Behavioral of Comp_4 is
44
45  component Comp_1 is
46         Port ( num1 : in STD_LOGIC;
47                num2 : in STD_LOGIC;
48                EN : in STD_LOGIC;
49                equal : out STD_LOGIC;
50                greater : out STD_LOGIC;
51                lesser : out STD_LOGIC );
52  end component;
53
54  SIGNAL E_3,G_3,L_3,E_2,G_2,L_2,E_1,G_1,L_1,E_0,G_0,L_0,e,g,l : STD_LOGIC;
55
56  begin
57
58      Comp_1_MSB : Comp_1
59          Port map( num1 => num1(3),
60                 num2 => num2(3),
61                 EN => EN,
62                 equal => E_3,
63                 greater => G_3,
64                 lesser => L_3 );
65
66      Comp_1_2 : Comp_1
67         Port map( num1 => num1(2),
68                num2 => num2(2),
69                EN => E_3,
70                equal => E_2,
71                greater => G_2,
72                lesser => L_2 );
73
74      Comp_1_1 : Comp_1
75        Port map( num1 => num1(1),
76               num2 => num2(1),
77               EN => E_2,
78               equal => E_1,
79               greater => G_1,
80               lesser => L_1 );
```

```
    Comp_1_LSB : Comp_1
     Port map( num1 => num1(0),
            num2 => num2(0),
            EN => E_1,
            equal => E_0,
            greater => G_0,
            lesser => L_0 );

    e <= E_0;
    g <= G_3 OR G_2 OR G_1 OR G_0;
    l <= L_3 OR L_2 OR L_1 OR L_0;

    process (num1,num2,EN,e,g,l) begin
        if (num1(3)='1' AND num2(3)='0') then
            equal <= '0';
            greater <= '0';
            lesser <= EN;
        elsif (num1(3)='0' AND num2(3)='1') then
            equal <= '0';
            greater <= EN;
            lesser <= '0';
        else
            equal <= e;
            greater <= g;
            lesser <= l;
        end if;
    end process;

end Behavioral;
```

4. Multiplier

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

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

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

entity Multiplier 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));
end Multiplier;

architecture Behavioral of Multiplier 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 b0a0, b0a1, b1a0, b1a1 : std_logic;
signal s_0_0, s_0_1, c_0_0, c_0_1 : std_logic;

begin

```

```
56
57  FA_0_0: FA port map(
58       A => b0a1,
59       B => b1a0,
60       C_in => '0',
61       S => s_0_0,
62       C_out => c_0_0
63  );
64
65  FA_0_1: FA port map(
66       A => '0',
67       B => b1a1,
68       C_in => c_0_0,
69       S => s_0_1,
70       C_out => c_0_1
71  );
72
73    b0a0 <= A(0) and B(0);
74    b1a0 <= A(0) and B(1);
75    b0a1 <= A(1) and B(0);
76    b1a1 <= A(1) and B(1);
77
78    Y(0) <= b0a0;
79    Y(1) <= s_0_0;
80    Y(2) <= s_0_1;
81    Y(3) <= c_0_1;
82
83  end Behavioral;
```

5. Program Rom

```
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  use ieee.numeric_std.all;
26
27  -- Uncomment the following library declaration if using
28  -- arithmetic functions with Signed or Unsigned values
29  --use IEEE.NUMERIC_STD.ALL;
30
31  -- Uncomment the following library declaration if instantiating
32  -- any Xilinx leaf cells in this code.
33  --library UNISIM;
34  --use UNISIM.VComponents.all;
35
36  entity ROM is
37      Port ( address : in STD_LOGIC_VECTOR (2 downto 0);
38             data : out STD_LOGIC_VECTOR (12 downto 0));
39  end ROM;
40
41  architecture Behavioral of ROM is
42
43  type rom_type is array (0 to 7) of std_logic_vector(12 downto 0);
44  signal prosseser_ROM : rom_type := (
45
46  --        process => 3 MUL 2 = 6
47          "0101110000011", -- MOVI R7,3     R7 <- 3
48          "0100010000010", -- MOVI R1,2     R1 <- 2
49          "1101110010000", -- MUL  R7,R1    R7 <- R7 MUL R1
50          "0110000000011", -- JZR  R0,3
51          "0000000000000",
52          "0000000000000",
53          "0000000000000",
54          "0000000000000"
55
```

```
--        process => 3 + 2 + 1 = 6
--         "0100010000100", -- MOVI R1,4        R1 <- 4            0
--         "0100100000001", -- MOVI R2,1        R2 <- 1            1
--         "0010100000000", -- NEG  R2          R2 <- -1           2
--        "0000010100010", -- ADD  R1,R2       R1 <- R2 + R1      3
--         "0110010000111", -- JZR  R1,7                           4
--         "0001110010010", -- ADD  R7,R1       R7 <- R7 + R1      5
--         "0110000000011", -- JZR  R0,3                           6
--         "0110000000111" -- JZR  R0,7                            7

--         process => (-1) AND 5 = 5
--         "0101110001111", -- MOVI R7,-1     R7 <- -1
--         "0100010000101", -- MOVI R1,5      R1 <- 5
--         "0001110010000", -- AND  R7,R1     R7 <- R7 AND R1
--         "0110000000011", -- JZR  R0,3
--         "0000000000000",
--         "0000000000000",
--         "0000000000000",
--         "0000000000000"

--         process => 1 OR 7 = 7
--         "0101110000001", -- MOVI R7,1     R7 <- 1
--         "0100010000111", -- MOVI R1,7     R1 <- 7
--         "0001110010001", -- OR  R7,R1     R7 <- R7 OR R1
--         "0110000000011", -- JZR  R0,3
--         "0000000000000",
--         "0000000000000",
--         "0000000000000",
--         "0000000000000"


--         process => 1 SUB 7 = (-6)
--         "0101110000001", -- MOVI R7,1     R7 <- 1
--         "0100010000111", -- MOVI R1,7     R1 <- 7
--         "0001110010110", -- SUB  R7,R1    R7 <- R7 SUB R1
--         "0110000000011", -- JZR  R0,3
--         "0000000000000",
--         "0000000000000",
--         "0000000000000",
--         "0000000000000"


    );

begin

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

end Behavioral;

```

6. MUX & Tri-State Buffers

- Tri-State Buffer

```
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  -- Uncomment the following library declaration if using
26  -- arithmetic functions with Signed or Unsigned values
27  --use IEEE.NUMERIC_STD.ALL;
28
29  -- Uncomment the following library declaration if instantiating
30  -- any Xilinx leaf cells in this code.
31  --library UNISIM;
32  --use UNISIM.VComponents.all;
33
34  entity Tri_State_Buffer_4 is
35      Port ( I : in STD_LOGIC_VECTOR (3 downto 0);
36             EN : in STD_LOGIC;
37             Q : out STD_LOGIC_VECTOR (3 downto 0));
38  end Tri_State_Buffer_4;
39
40  architecture Behavioral of Tri_State_Buffer_4 is
41
42  begin
43    process (I,EN) begin
44        if EN = '1' then
45            Q <= I;
46        else
47            Q <= "ZZZZ";
48        end if;
49      end process;
50
51  end Behavioral;
52
```

- 2-way 3-bit Mux

```
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  -- Uncomment the following library declaration if using
26  -- arithmetic functions with Signed or Unsigned values
27  --use IEEE.NUMERIC_STD.ALL;
28
29  -- Uncomment the following library declaration if instantiating
30  -- any Xilinx leaf cells in this code.
31  --library UNISIM;
32  --use UNISIM.VComponents.all;
33
34  entity Mux_2_3 is
35      Port ( A1 : in STD_LOGIC_VECTOR (2 downto 0);
36             A2 : in STD_LOGIC_VECTOR (2 downto 0);
37             Selector : in STD_LOGIC;
38             Output : out STD_LOGIC_VECTOR (2 downto 0));
39  end Mux_2_3;
40
41  architecture Behavioral of Mux_2_3 is
42
43  component Tri_State_Buffer is
44      Port ( I : in STD_LOGIC_VECTOR (2 downto 0);
45             EN : in STD_LOGIC;
46             Q : out STD_LOGIC_VECTOR (2 downto 0));
47  end component;
48
49  signal enable_signals : STD_LOGIC_VECTOR(1 downto 0) ;
50
```

```
begin

     enable_signals(0) <= NOT Selector;
     enable_signals(1) <= Selector;


     buffer_0: Tri_State_Buffer
          port map (
              I => A1,
              EN => enable_signals(0),
              Q => Output
          );

     buffer_1: Tri_State_Buffer
          port map (
              I => A2,
              EN => enable_signals(1),
              Q => Output
          );

end Behavioral;
```

- 2-way 4-bit Mux

```
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_4 is
    Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
           B : in STD_LOGIC_VECTOR (3 downto 0);
           Selector : in STD_LOGIC;
           Mux_out : out STD_LOGIC_VECTOR (3 downto 0));
end Mux_2_4;

architecture Behavioral of Mux_2_4 is

component Tri_State_Buffer_4 is
    Port ( I : in STD_LOGIC_VECTOR (3 downto 0);
       EN : in STD_LOGIC;
       Q : out STD_LOGIC_VECTOR (3 downto 0));
end component;

signal enable_signals : STD_LOGIC_VECTOR(1 downto 0) := "01" ;
begin

     enable_signals(0) <= NOT Selector;
     enable_signals(1) <= Selector;

     buffer_0: Tri_State_Buffer_4
          port map (
              I => A,
              EN => enable_signals(0),
              Q => Mux_out
          );

```

```
    buffer_1: Tri_State_Buffer_4
          port map (
              I => B,
              EN => enable_signals(1),
              Q => Mux_out
          );

end Behavioral;
```

- 4-way 4-bit Mux

```
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_4_4 is
    Port (
       A1 : in STD_LOGIC_VECTOR (3 downto 0);
       A2 : in STD_LOGIC_VECTOR (3 downto 0);
       A3 : in STD_LOGIC_VECTOR (3 downto 0);
       A4 : in STD_LOGIC_VECTOR (3 downto 0);
       Selector : in STD_LOGIC_VECTOR (1 downto 0);
       Output : out STD_LOGIC_VECTOR (3 downto 0));
end Mux_4_4;

architecture Behavioral of Mux_4_4 is

component Tri_State_Buffer_4
    Port ( I : in STD_LOGIC_VECTOR (3 downto 0);
           EN : in STD_LOGIC;
           Q : out STD_LOGIC_VECTOR (3 downto 0));
end component;

component Decoder_2_4
    Port ( I : in STD_LOGIC_VECTOR (1 downto 0);
           EN : in STD_LOGIC;
           Y : out STD_LOGIC_VECTOR (3 downto 0));
end component;

signal enable_signals : std_logic_vector (3 downto 0);
begin
    Decoder : Decoder_2_4
        port map(
            I => Selector,
            EN => '1',
            Y => enable_signals
        );

    buffer_0: Tri_State_Buffer_4
        port map (
            I => A1,
            EN => enable_signals(0),
            Q => Output
        );
```

```
    buffer_1: Tri_State_Buffer_4
        port map (
            I => A2,
            EN => enable_signals(1),
            Q => Output
        );

    buffer_2: Tri_State_Buffer_4
        port map (
            I => A3,
            EN => enable_signals(2),
            Q => Output
        );

    buffer_3: Tri_State_Buffer_4
        port map (
            I => A4,
            EN => enable_signals(3),
            Q => Output
        );

end Behavioral;
```

- 8-way 4-bit Mux

```
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_4 is
    Port (
        A1 : in STD_LOGIC_VECTOR (3 downto 0);
        A2 : in STD_LOGIC_VECTOR (3 downto 0);
        A3 : in STD_LOGIC_VECTOR (3 downto 0);
        A4 : in STD_LOGIC_VECTOR (3 downto 0);
        A5 : in STD_LOGIC_VECTOR (3 downto 0);
        A6 : in STD_LOGIC_VECTOR (3 downto 0);
        A7 : in STD_LOGIC_VECTOR (3 downto 0);
        A8 : in STD_LOGIC_VECTOR (3 downto 0);
        Selector : in STD_LOGIC_VECTOR (2 downto 0);
        Output : out STD_LOGIC_VECTOR (3 downto 0)
        );
end Mux_8_4;

architecture Behavioral of Mux_8_4 is

component Tri_State_Buffer_4 is
    Port ( I : in STD_LOGIC_VECTOR (3 downto 0);
       EN : in STD_LOGIC;
       Q : out STD_LOGIC_VECTOR (3 downto 0));
end component;

component Decoder_3_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 enable_signal : STD_LOGIC_VECTOR (7 downto 0);
```

```
65  begin
66
67      Decoder_3_8_0 : Decoder_3_8
68          port map(
69              I => Selector,
70              EN => '1',
71              Y => enable_signal
72      );
73
74      buffer_1 : Tri_State_Buffer_4
75          port map(
76              I => A1,
77              EN => enable_signal(0),
78              Q  => Output
79      );
80
81      buffer_2 : Tri_State_Buffer_4
82          port map(
83              I => A2,
84              EN => enable_signal(1),
85              Q  => Output
86      );
87
88      buffer_3 : Tri_State_Buffer_4
89          port map(
90              I => A3,
91              EN => enable_signal(2),
92              Q  => Output
93      );
94
95      buffer_4 : Tri_State_Buffer_4
96          port map(
97              I => A4,
98              EN => enable_signal(3),
99              Q  => Output
100     );
101
102     buffer_5 : Tri_State_Buffer_4
103         port map(
104             I => A5,
105             EN => enable_signal(4),
106             Q  => Output
107     );
108
109     buffer_6 : Tri_State_Buffer_4
110         port map(
111             I => A6,
112             EN => enable_signal(5),
113             Q  => Output
114     );
115
116     buffer_7 : Tri_State_Buffer_4
117         port map(
118             I => A7,
119             EN => enable_signal(6),
120             Q  => Output
121     );
122
123     buffer_8 : Tri_State_Buffer_4
124         port map(
125             I => A8,
126             EN => enable_signal(7),
127             Q  => Output
128     );
129
130 end Behavioral;
```

7. Constraint 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 {Dis_LED[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_LED[0]}]
set_property PACKAGE_PIN E19 [get_ports {Dis_LED[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_LED[1]}]
set_property PACKAGE_PIN U19 [get_ports {Dis_LED[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_LED[2]}]
set_property PACKAGE_PIN V19 [get_ports {Dis_LED[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_LED[3]}]

set_property PACKAGE_PIN U14 [get_ports {Comparator_out[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Comparator_out[0]}]
set_property PACKAGE_PIN V14 [get_ports {Comparator_out[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Comparator_out[1]}]
set_property PACKAGE_PIN V13 [get_ports {Comparator_out[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Comparator_out[2]}]

set_property PACKAGE_PIN P3 [get_ports {Flags[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Flags[3]}]
set_property PACKAGE_PIN N3 [get_ports {Flags[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Flags[1]}]
set_property PACKAGE_PIN P1 [get_ports {Flags[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Flags[2]}]
set_property PACKAGE_PIN L1 [get_ports {Flags[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Flags[0]}]


##7 segment display
set_property PACKAGE_PIN W7 [get_ports {Dis_7Seg[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_7Seg[0]}]
set_property PACKAGE_PIN W6 [get_ports {Dis_7Seg[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_7Seg[1]}]
set_property PACKAGE_PIN U8 [get_ports {Dis_7Seg[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_7Seg[2]}]
set_property PACKAGE_PIN V8 [get_ports {Dis_7Seg[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_7Seg[3]}]
set_property PACKAGE_PIN U5 [get_ports {Dis_7Seg[4]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_7Seg[4]}]

set_property PACKAGE_PIN V5 [get_ports {Dis_7Seg[5]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_7Seg[5]}]
set_property PACKAGE_PIN U7 [get_ports {Dis_7Seg[6]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Dis_7Seg[6]}]

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

set_property PACKAGE_PIN V17 [get_ports {Reset}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Reset}]

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


set_property CFGBVS VCCO [current_design]
set_property CONFIG_VOLTAGE 3.3 [current_design]
```

Problems Faced

1. **Load Function via Switches**: We attempted to implement a load functionality that would allow users to input data via the board's switches into a register instead of using hardcoded values from the ROM. This feature failed due to a timing issue between input capture and the program counter incrementing to the next instruction. We propose resolving this in future work so that dynamic inputs from the switches can be supported for real-time operations.

2. **Instruction Width Limitation**: In the extended design, the increased number of supported operations (arithmetic and logical) required more bits to encode the functions. The minimal design used only 2 bits for instruction types, which was sufficient for 4 functions. For the extended version, we expanded this to 3 bits to accommodate additional instructions. As a result, our instruction width increased from 12 bits to 13 bits, requiring changes across the entire datapath including the ROM, instruction decoder, and bus widths.

3. **Reset Mechanism Using Pushbutton**: Initially, we used a pushbutton from the BASYS 3 board as the reset signal. While pressing the button set the reset signal to 1 (correctly resetting the Program Counter and registers), releasing the button immediately reset it to 0, causing inconsistent behavior. Since a proper reset requires a transition from 1 to 0 to restart execution, a pushbutton was not reliable. We resolved this by using a switch instead. Turning the switch ON sets reset to 1 (performs reset), and turning it OFF sets reset to 0 (allows the program to run again), ensuring consistent and expected reset behavior.

Conclusion

We successfully designed, implemented, and tested both Minimal and Extended versions of a 4-bit NanoProcessor using VHDL on the BASYS 3 FPGA. The Minimal Design supported core instructions like MOVI, ADD, NEG, and JZR, achieving the essential functionality of a simple processor.

The Extended Design introduced additional instructions (AND, OR, MUL), a comparator unit with EQ, GT, LT flags, and more advanced components like tri-state MUXes and an optimized ALU. These enhancements required expanding the instruction width and modifying the datapath accordingly.

Each module was thoroughly tested using simulations and timing diagrams to ensure correct operation. This project strengthened our understanding of processor architecture, VHDL-based hardware design, and system integration on FPGAs, while preparing us for more advanced digital design challenges.

Contributions

Member	Contributions	Worked Time
MADAELA M.P.G.R.V.M. 230381X	● Designed and implemented Minimal Nanoprocessor and Extended Nanoprocessor top modules. ● Port mapping of Minimal and Extended processor components. ● Developed testbenches and timing diagrams for both processor versions. ● Contributed to VHDL code for integration and functional validation. ● Coordinated group tasks and contributed to final report compilation.	**40 hours**

KESHARA K.P.W.D. 230334H	• Designed and implemented the Instruction Decoder, Program Counter, and Program ROM for both processor versions. • Created corresponding testbenches and timing diagrams. • Assisted in developing machine code and instruction encoding for both versions. • Worked on constraint files (Basys3.xdc) and ensured board compatibility.	**40 hours**
SENANAYAKE H.P.V.R. 230595G	• Developed and tested the ALU, including additional operations (AND, OR, MUL) in the extended version. • Implemented and validated the Comparator Unit (EQ, GT, LT flags). • Handled add/sub unit, 4-bit adder, and negation logic. • Worked on extended ALU testbenches and diagrams.	**40 hours**
SOMARATHNA M.D.A.M. 230621K	• Designed and implemented MUXes (2-way/4-way/8-way) and Tri-State Buffers. • Built and tested Register Bank, 4-bit Registers, and related components. • Developed 3 to 8 Decoder, LUT for 7-Segment Display, and Slow Clock Generator. • Responsible for final report formatting and proofreading.	**40 hours**