University of Essex

CE869 High Level Logic Design Assignment 3

Name: Ye Yihao

Student Number: 1305597

Tutor: Luca Citi

Lecture: High Level Logic Design

Submission Date: 20 / 03 / 2014

WORD COUNT: 2151

Table of contents

1. Project description

The document is relevant to the project which is aim to use the FPGA (Digilent XILINX BASYS2 board) and hardware description language VHDL to implement a digital system with test, debug and verify steps.

To be specific, task for this project is to implement an 8 bit simple CPU. Particular, the program sequencing control flow instruction datapath can be modeled following the Figure 1 left part, while the arithmetic logic instruction datapath can follow a structure like the other one within the same figure. The opcodes for the instructions that the CPU is required to implement are presented in Table 1.

In this project, modular coding style is used for programming which leads to greater exibility, maintainability, modularity, and reusability. At first, the individual component units were implemented in single entity, then program sequencing control flow and arithmetic logic instruction which are the main branches of the datapath were implemented via combining the above entities. The control unit should be programmed after datapath finished and performed well. A microprocessor entity will be created to interface the input and output within the board, this entity should contain both program sequencing control flow and arithmetic logic instruction.

The combinatorial circuits part should use concurrent statements, and sequential circuits part should apply sequential code. What should be attention is the use of non-standard packages and BUFFER ports is forbidden, while the use of INOUT ports is accepted only when strictly necessary.

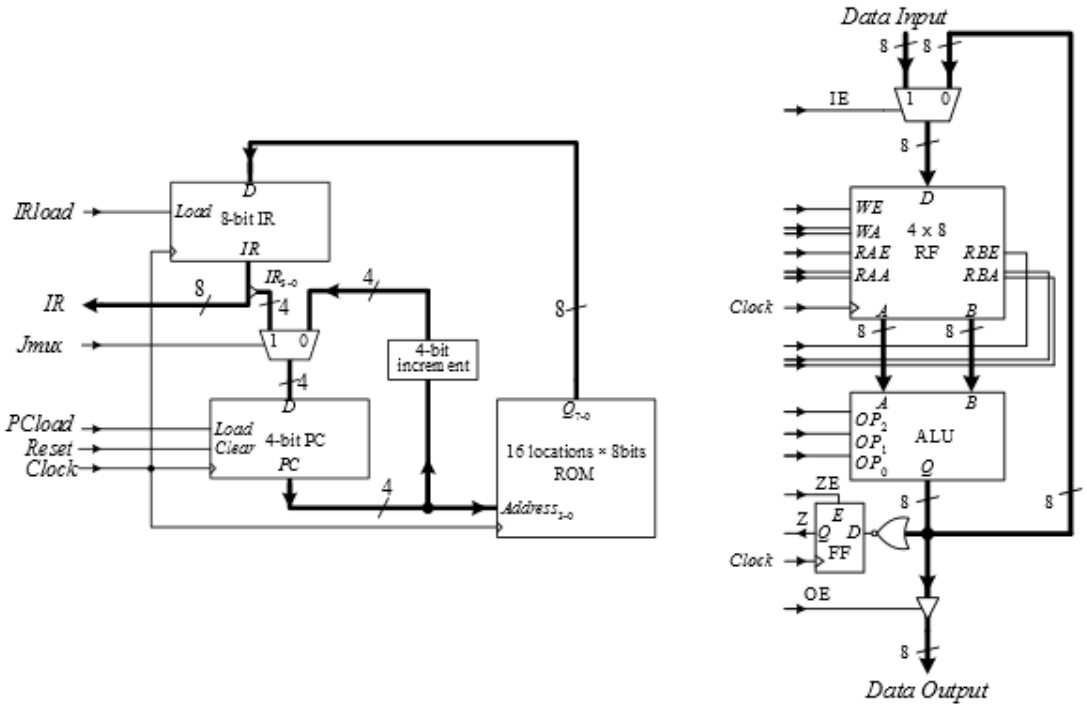


Figure 1. Two parts of datapath

2. Modules construction Process

As in a top-down fashion, the top module should be the main entity which have three input and one output, then is the microprocessor and display (i.e. Seven segments display output), following is the control unit and datapath, inside the datapath, it can also divided into program sequencing control flow instruction datapath and arithmetic logic instruction datapath. Finally, inside these two datapath and control unit, there are the element units e.g. ROM, ALU, RegisterFile etc.

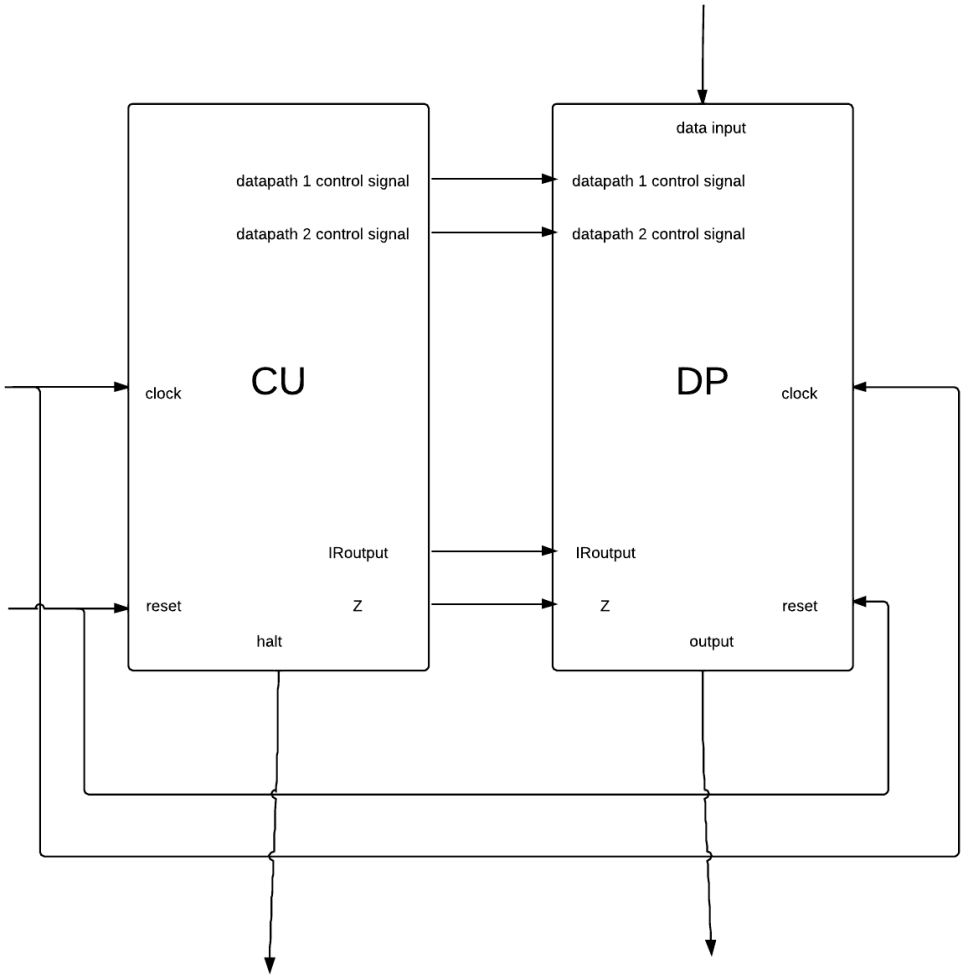


Figure 2. Connection between controlunit and datapath

The whole program should be initially built up depending on the below circuit and logic, the first part should be finished is the elemental units like RF, ALU, IR, in this step, we should configure the port of these units (or block) and define the direction of the ports. While configuring the ports of the units, we also need to build the logic between the ports of a single units, these logic are not directly shown in the figure but in the lecture handout and relevant background knowledge. The logic and condition is really important since it determine the operation of the microprocessor. Moreover, there are something should be noticed during

programming the logic with the units, that is when units work with a clock input, the program statement of it should be sequential otherwise it had better to be concurrent.

After the elemental units finished, we need to combine them via mapping the port according to the Figure (which indicate the relationship of the ports). In this steps, we need to notice the number of bits of ports which sometimes will cause crash, for example, output of instruction register is mapping with the input of the Jmux unit, however, the number of bits of them are different and only four bits of the instruction register output communicate with Jmux.

When the datapath part is nearly finished, we can start the control unit program. This part is more complex than the datapath since the logic of it is not directly shown in circuits or figures. Instead, the logic of control unit is indicated by the instruction table in the assignment specification and the finite state machine model.

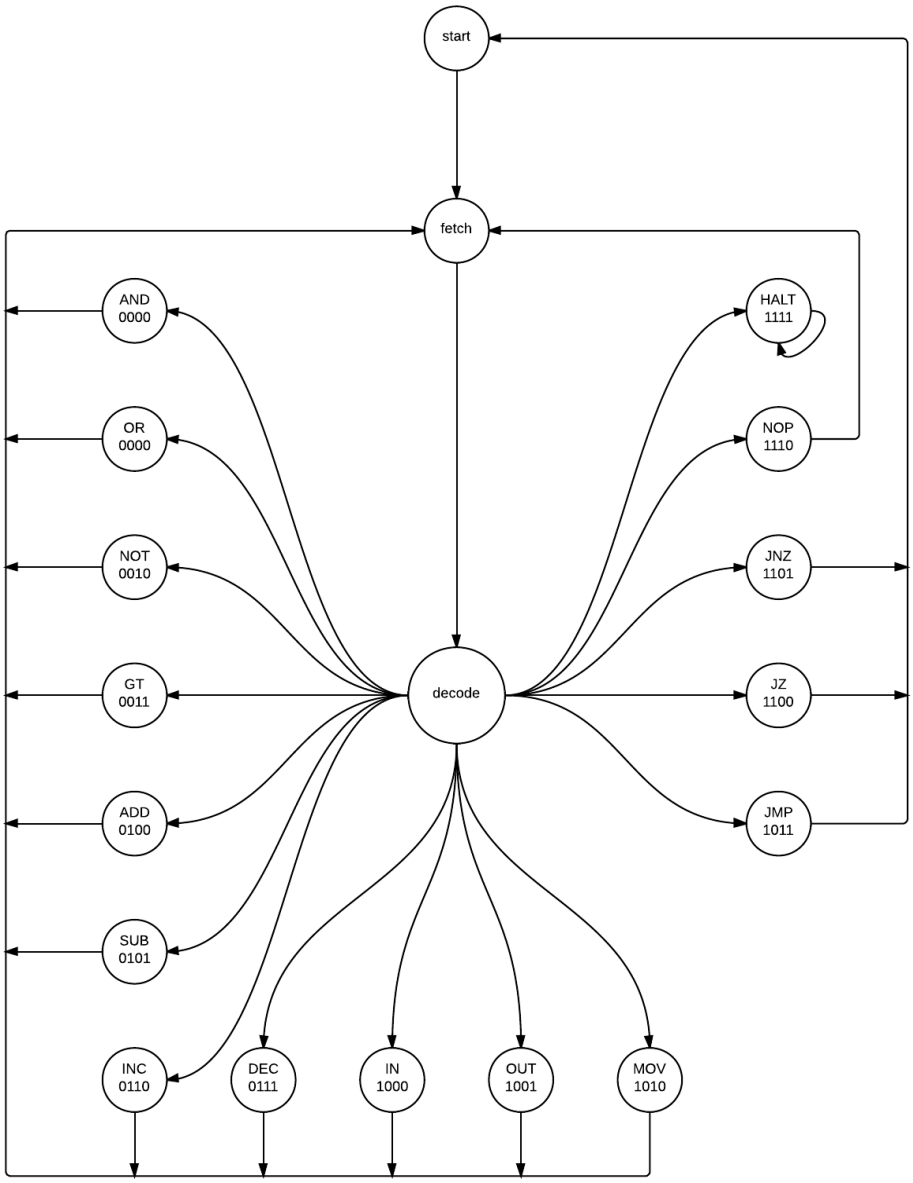


Figure 3. Finite state machine model

Briefly to explain the function and the execution process of the control unit, we should look at the finite state machine model first (see Figure 3.). According to the state machine, the control unit would be placed in start state initially, then it will turns to the next state which is state fetch and similarly goes to the decode state. The function of decode state is to acquire the instruction from the ROM and decide which operation state should be carried out depends on the instruction higher four bits. When the operation state is finished, it will back to the fetch state again and work with the next instruction. Once a circle of finite state machine finished, it usually means one instruction has been executed.

There are 16 different type of instructions can be used in the control unit. As put a set of particular instructions (which means program you design for the processor to execute) into the ROM and then run the microprocessor, the microprocessor will run the program you designed.

Instruction	Encoding	Affects	Operation
AND Raa, Rbb	0000aabb	Z	Raa ← Raa AND Rbb
OR Raa, Rbb	0001aabb	Z	Raa ← Raa OR Rbb
NOT Raa	0010aaxx	Z	Raa ← NOT Raa
GT Raa, Rbb	0011aabb	Z	Set Z to 0 if Raa > Rbb,
ADD Raa, Rbb	0100aabb	Z	Raa ← Raa + Rbb
SUB Raa, Rbb	0101aabb	Z	Raa ← Raa - Rbb
INC Raa	0110aaxx	Z	Raa ← Raa + 1
DEC Raa	0111aaxx	Z	Raa ← Raa - 1
IN Raa	1000aaxx		Raa ← Input
OUT Raa	1001aaxx		Output ← Raa
MOV Raa, Rbb	1010aabb		Raa ← Rbb
JMP aaaa	1011aaaa		Jump to aaaa
JZ aaaa	1100aaaa		Jump to aaaa if Z
JNZ aaaa	1101aaaa		Jump to aaaa if Z
NOP	1110xxxx		Do nothing
HALT	1111xxxx		Halt

Table 1. 16 instruction applied in the system

As the control unit is in one of the states, it will output the corresponding control signal (e.g. IE, WE, RAE, RAA, OP, ZE etc.) to datapath. In this way, control unit control the data flow of the datapath so that it implement the instruction.

Control Word	Encoding	IRload	Jmux	Pcload	IE	WE	RAE	RBE	ZE	OE
0	Start	0	0	0	0	0	0	0	0	0
1	Fetch	1	0	1	0	0	0	0	0	0
2	Decode	0	0	0	0	0	0	0	0	0
3	AND	0	0	0	0	1	1	1	1	0
4	OR	0	0	0	0	1	1	1	1	0
5	NOT	0	0	0	0	1	1	0	1	0
6	GT	0	0	0	0	0	1	1	1	0
7	ADD	0	0	0	0	1	1	1	1	0
8	SUB	0	0	0	0	1	1	1	1	0
9	INC	0	0	0	0	1	1	0	1	0
10	DEC	0	0	0	0	1	1	0	1	0
11	IN	0	0	0	1	1	0	0	0	0
12	OUT	0	0	0	0	0	1	0	0	1
13	MOV	0	0	0	0	1	0	1	0	0
14	JMP	1	1	1	0	0	0	0	0	0
15	JZ	1/0	1/0	1/0	0	0	0	0	0	0
16	JNZ	1/0	1/0	1/0	0	0	0	0	0	0
17	NOP	0	0	0	0	0	0	0	0	0
18	HALT	0	0	0	0	0	0	0	0	0

Table 2. Corresponding control signal with states

Then combine control unit and datapath into the entity named microprocessor and map the ports between them.

3. Test, debug and simulation

As the process of build up the microprocessor was completed, it should come to the test part before designing and running the assembly program.

To test the microprocessor, live debugging and testbenches is two different method we can choose.

First, use a debug entity to test the arithmetic/logic datapath respectively. The switches and buttons on the Diligent board were employed to manually simulate the control signals from the Program sequencing/control flow instruction datapath and the seven segment display was used to show the result of each operation. Additionally, designer should notice that as the arithmetic/logic datapath worked well at once and then there was no need to change it any more.

Second, to simulate the control unit and the microprocessor, just use the testbanches. Being different from the arithmetic/logic datapath, the control unit does not work right away and simulations were required to take many times to identify different problems. Part of Issues related with the assertion of the signals during execution of the state machine were identified

and fixed during using the testbench for the control unit, after that we can create the a testbench for the microprocessor, and the whole microprocessor including control unit and datapath was simulated.

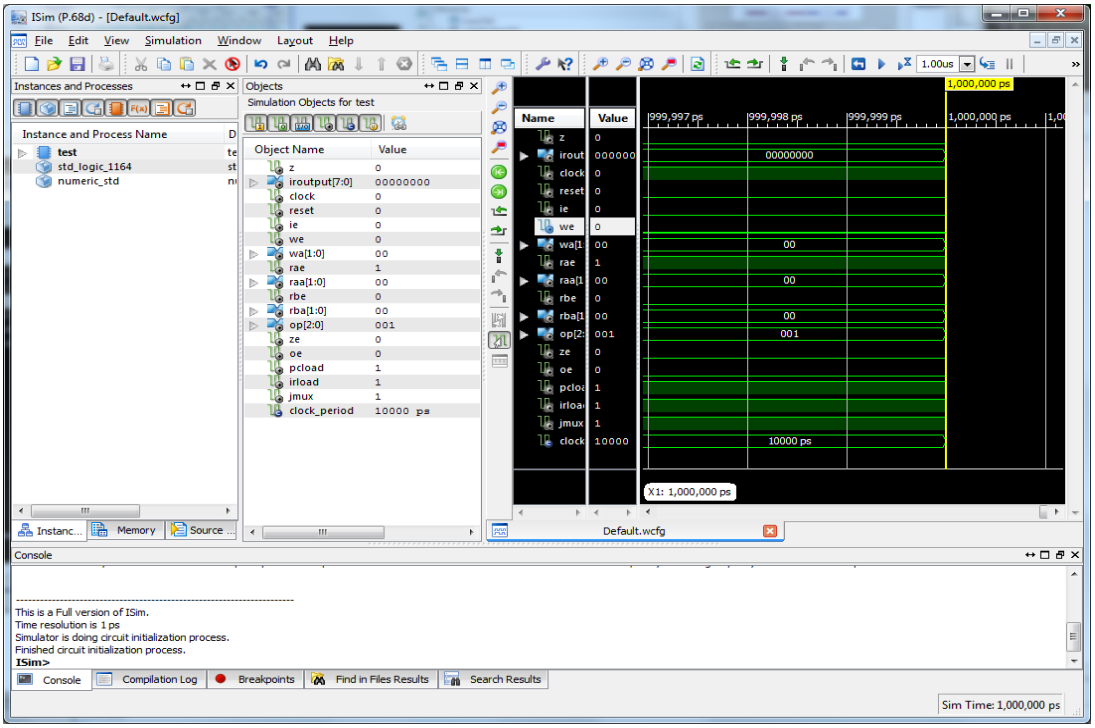


Figure 4. Simulation process

4. Revise process

The problem with mapping the ports cost some time to solve, in this project, the error cases include defined the same signal more than once, this is because as entities need communication between their port and the name of the port is different, it is easy to happen redefine and miss of define. To decrease the error, it is better to choose a common name for all the connected ports within a same circuit branch.

The problem with the type of the signal, like STD_LOGIC_VECTOR can not carry the plus and minus operation directly, instead, it should be transformed to the unsigned type to carry the operation first and then transformed to the original type again. A better way to improve it is use unsigned type for the majority of the signal at the beginning of the program. When the program go ahead, it become more and more difficult to change the type back.

The problem with jump did not work once, the problem then was found out to be related to the signal used in the PC unit, in the previous version, a counter signal was defined to carry the operation of PC unit and then transfer the value to the PCoutput, as PCoutput directly carry the operation of PC unit instead of the counter, the problem with jump solved.

The final problem is when it is in halt state, seven segments do not display the output of ALU while just display zero.

5. Discussion of design alternatives

The design alternatives in this project include add the alternative mode with button control clock input which can help to debug the program in a slow and controllable speed, in this mode, another useful alternative is LEDs are linking to the instruction register output, in this case, designer can detect which of the current state is for the system. It will be more helpful combining the use of former two functions. And the normal mode is just meet the requirements of the assignment, mode can be changed between them by a small Adjustments of code before turn it on.

6. Conclusions

Briefly, the designed digital system meets the specifications of the assignment. Modular architecture really contributes in implementing the main part of the project.

The three tasks have been well implemented which indicate the structure, code and the design do not have much problem.

The purpose of the project is to make students to have a deep understand with the system architecture and the use of hardware description language VHDL through implementing a digital system with it as well as test, debug and simulate the system. In this example, the purpose of the project seems to be achieved. However although the designed microprocessor is simple, the program of VHDL is still complicate. A simulation is following the implementation of the program to ensure the microprocessor work precisely. The simulation also help the designer to grasp the time of how the control interact with datapath and the port setting as normal sequence diagram do. It is required to simulating the microprocessor and going through the simulation, instructions should be carried in order to make system work precisely.

Finally, tasks were well implemented and I think the project is really helpful for students to grasp the structure and principle of microprocessor as well as have a deeper understand with VHDL language and initially apply testbenches during the VHDL simulation.

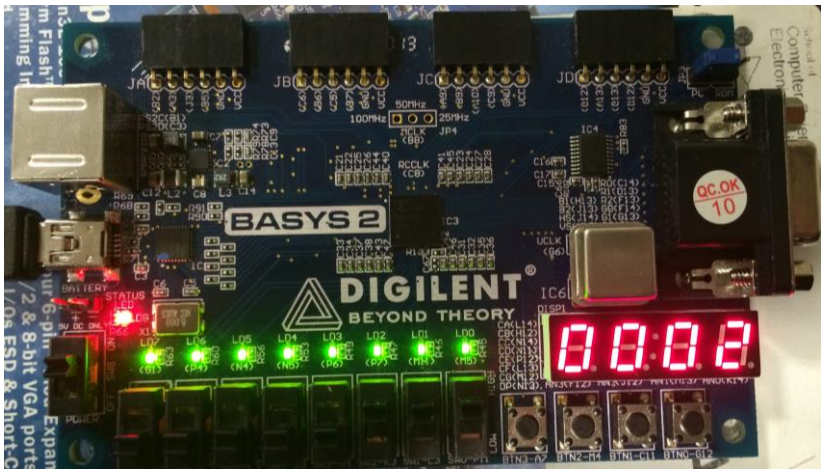
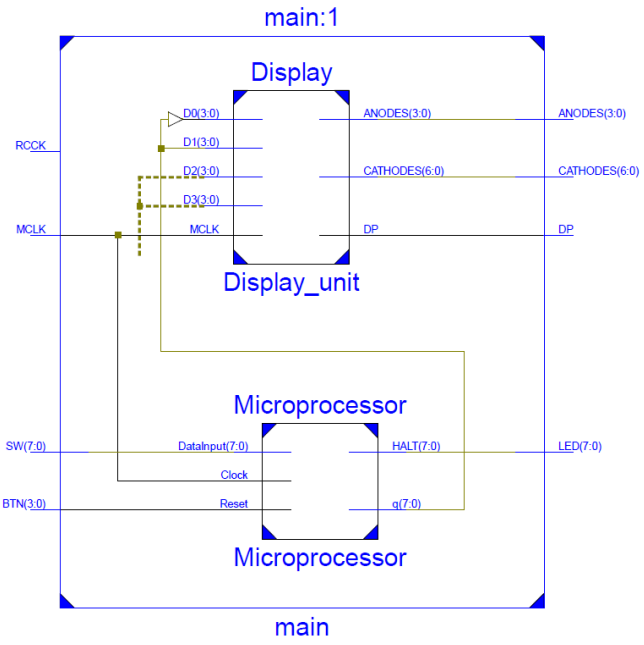
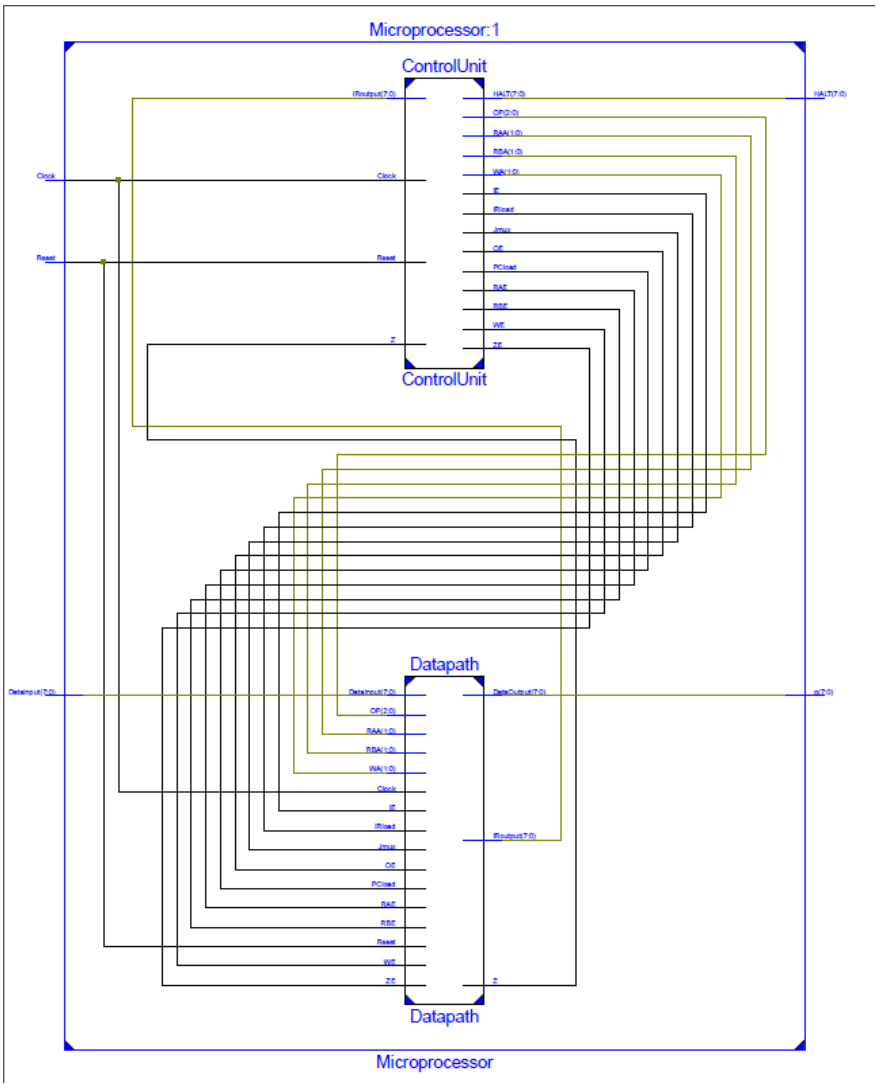


Figure 5. Task 2 implementation

The diagram of the Microprocessor and datapath is shown below:





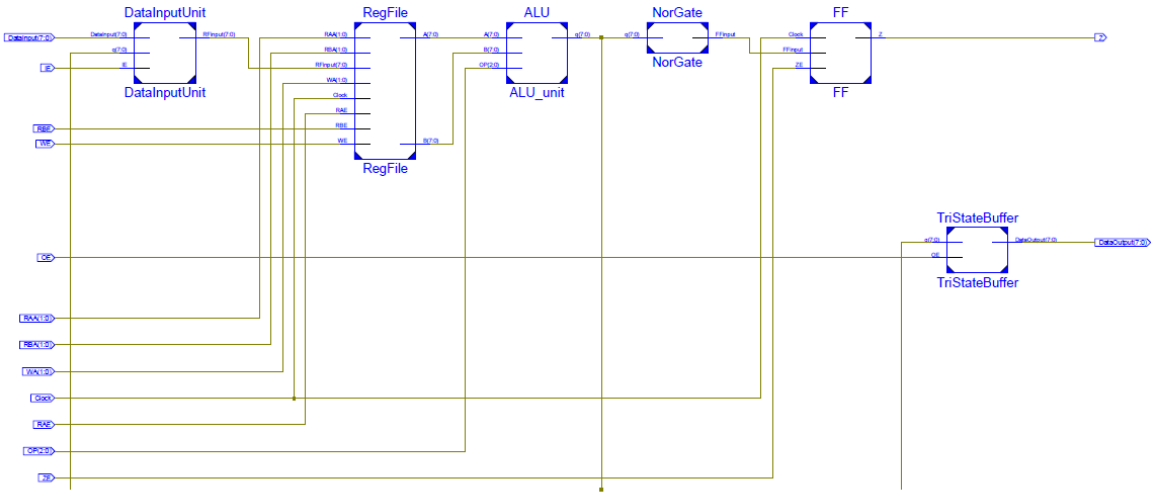


Figure 6. Arithmetic logic instruction datapath

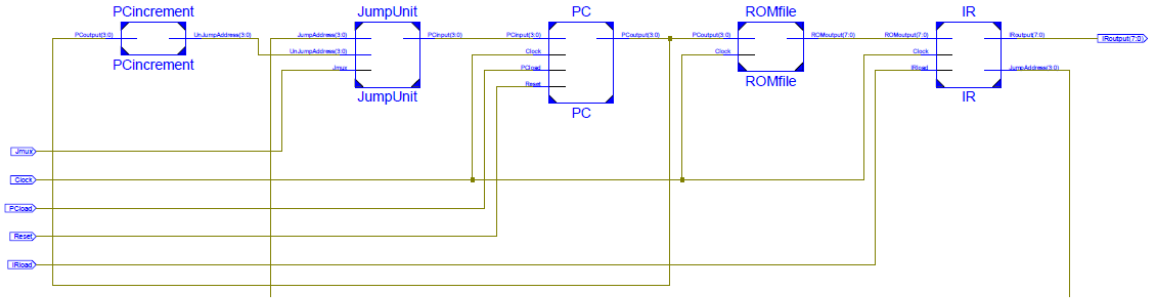


Figure 7. Program sequencing control flow instruction datapath

References

[1] Citi, Luca, High Level Logic Design Lecture, Topic: “General Purpose Microprocessor Design,” 5A.303, School of Computer Science and Electronic Engineering, University of Essex, Colchester, Essex, Mar. 3, 2013.

[2] Hwang, Enoch O., “Microprocessor Design Principles and Practices With VHDL,” Brooks / Cole, 2004.

[3] Citi, Luca, High Level Logic Design course forum,“testing the arithmetic/logic instruction datapath,” https://moodle.essex.ac.uk/mod/forum/discuss.php?d=81649, [Mar, 11, 2014].

Appendix:

Main:

```
----------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    12:17:43 02/20/2014
-- Design Name:
-- Module Name:    main - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity main is
   Port ( SW       : in  UNSIGNED (7 downto 0);--switch
    RCCK     : in  STD_LOGIC;
    MCLK     : in  STD_LOGIC;--MCLK clock
    BTN      : in  STD_LOGIC_VECTOR (3 downto 0);--four buttons ?????
   DP       : out STD_LOGIC;
    LED      : out UNSIGNED (7 downto 0);--eight bits LED
   CATHODES : out STD_LOGIC_VECTOR (6 downto 0);--signal control seven
segments
   ANODES   : out STD_LOGIC_VECTOR (3 downto 0));--signal control four
seven segments display

   attribute LOC : string;
   attribute CLOCK_DEDICATED_ROUTE : boolean;
    attribute LOC of SW   : signal is "N3 E2 F3 G3 B4 K3 L3 P11";
    attribute LOC of RCCK : signal is "C8";
    attribute LOC of MCLK : signal is "B8";
    attribute LOC of BTN : signal is "A7 M4 C11 G12";
    attribute CLOCK_DEDICATED_ROUTE of BTN : signal is FALSE;
   attribute LOC of DP  : signal is "N13";
   attribute LOC of CATHODES : signal is "M12 L13 P12 N11 N14 H12 L14";
```

```
    attribute LOC of LED : signal is "G1 P4 N4 N5 P6 P7 M11 M5";
   attribute LOC of ANODES   : signal is "K14 M13 J12 F12";

end main;

architecture Behavioral of main is
    signal D0, D1, D2, D3 : UNSIGNED (3 downto 0); --signal control all the
seven segments display
    signal q : STD_LOGIC_VECTOR (7 downto 0);   --result of the operation
logic with signal A and B

    signal DataInput: STD_LOGIC_VECTOR(7 downto 0);
    signal IE: STD_LOGIC;
    signal WE: STD_LOGIC;                           -- write enable
    signal WA: STD_LOGIC_VECTOR(1 downto 0);    -- write address
    signal RAE: STD_LOGIC;                          -- read port A enable
    signal RAA: STD_LOGIC_VECTOR(1 downto 0);-- read port A address
    signal RBE: STD_LOGIC;                          -- read port B enable
    signal RBA: STD_LOGIC_VECTOR(1 downto 0);-- read port B address
    signal OP: STD_LOGIC_VECTOR (2 downto 0);
    signal ZE: STD_LOGIC;
    signal Z: STD_LOGIC;
    signal OE: STD_LOGIC;
    --DataOutput: OUT STD_LOGIC_VECTOR(7 downto 0);

    signal PCload: STD_LOGIC;
    signal Reset: STD_LOGIC;
    signal IRload: STD_LOGIC;
    signal IRoutput: STD_LOGIC_VECTOR(7 downto 0);
    signal Jmux: STD_LOGIC;

    signal Clock: STD_LOGIC;

begin

    DataInput <= STD_LOGIC_VECTOR(SW);  --put the input of switchs to
DataInput
    Reset <= BTN(3);--use left most button as reset input
    Clock <= MCLK;--use MCLK as the clock input


    ControlUnit : entity ControlUnit (Behavioral)
            port map (IE => IE, WE => WE, WA => WA, RAE => RAE, RAA => RAA,
RBE => RBE,
                      RBA => RBA, OP => OP, ZE => ZE, Z => Z, OE => OE,
PCload => PCload,
                      IRload => IRload, IRoutput => IRoutput, Jmux =>
Jmux,
                      Clock => Clock, Reset => Reset);
    --ControlUnit entity, which send control signal to datapath


    Datapath : entity Datapath (Behavioral)
            port map (IE => IE, WE => WE, WA => WA, RAE => RAE, RAA => RAA,
RBE => RBE,
                      RBA => RBA, OP => OP, ZE => ZE, Z => Z, OE => OE,
PCload => PCload,
                      IRload => IRload, IRoutput => IRoutput, Jmux =>
Jmux,
                      Clock => Clock, Reset => Reset, DataInput =>
DataInput, DataOutput => q);
```

```
    --Datapath entity, which implement the function of datapath


    Display_unit : entity Display (Behavioral)
            port map (CATHODES => CATHODES, ANODES => ANODES, DP => DP,
MCLK => MCLK,
                        D0 => D0, D1 => D1, D2 => D2, D3 => D3);
    --Display entity, which display the output of the program


    D1 <= UNSIGNED(q (7 downto 4)); --second right seven segments display
show the higher four bits of q
    D0 <= UNSIGNED(q (3 downto 0)); --most right seven segments display
show the lower four bits of q
    LED <= UNSIGNED(IRoutput);  --LED indicate the value of IRoutput

end Behavioral;
```

ALU:

```
----------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    12:33:37 02/20/2014
-- Design Name:
-- Module Name:    ALU - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity ALU is
        Port (
```

```
                OP : in  STD_LOGIC_VECTOR (2 downto 0);
                A : in  STD_LOGIC_VECTOR (7 downto 0);--input A
                B : in  STD_LOGIC_VECTOR (7 downto 0);--input B
                q : out  STD_LOGIC_VECTOR (7 downto 0));--result of the
operation logic with signal A and B

end ALU;

architecture Behavioral of ALU is

begin

    PROCESS
    BEGIN

    IF OP = "000" THEN q <= A and B;
    ELSIF OP = "001" THEN q <= A or B;
    ELSIF OP = "010" THEN q <= not A;
    ELSIF OP = "011" and A > B THEN q <= "11111111";
    ELSIF OP = "100" THEN q <= STD_LOGIC_VECTOR(UNSIGNED(A) + UNSIGNED(B));
    ELSIF OP = "101" THEN q <= STD_LOGIC_VECTOR(UNSIGNED(A) - UNSIGNED(B));
    ELSIF OP = "110" THEN q <= STD_LOGIC_VECTOR(UNSIGNED(A) + 1);
    ELSIF OP = "111" THEN q <= STD_LOGIC_VECTOR(UNSIGNED(A) - 1);
    END IF;

    END PROCESS;
    --process which implement all the logic operation for signal A and B


end Behavioral;
```

Control Unit:

```
----------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    12:05:25 03/13/2014
-- Design Name:
-- Module Name:    ControlUnit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
```

```
-- arithmetic functions with Signed or Unsigned values
use IEEE.NUMERIC_STD.ALL;

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

entity ControlUnit is
    port(IE: OUT STD_LOGIC;
          WE: OUT STD_LOGIC;                              -- write enable
          WA: OUT STD_LOGIC_VECTOR(1 downto 0); -- write address
          RAE: OUT STD_LOGIC;                             -- read port A
enable
          RAA: OUT STD_LOGIC_VECTOR(1 downto 0);-- read port A address
          RBE: OUT STD_LOGIC;                             -- read port B
enable
          RBA: OUT STD_LOGIC_VECTOR(1 downto 0);-- read port B address
          OP: OUT  STD_LOGIC_VECTOR (2 downto 0);   --operation code
          ZE: OUT STD_LOGIC;
          Z: IN STD_LOGIC;
          OE: OUT STD_LOGIC;

          PCload: OUT STD_LOGIC;    --PC enable
          IRload: OUT STD_LOGIC;        --IR enable
          IRoutput: IN STD_LOGIC_VECTOR(7 downto 0);
          Jmux: OUT STD_LOGIC;

          Clock: IN STD_LOGIC;  --clock signal
          Reset: IN STD_LOGIC); --reset signal

end ControlUnit;

architecture Behavioral of ControlUnit is
    TYPE STATE_TYPE is (STATE_START, STATE_FETCH,
                            STATE_DECODE, STATE_AND, STATE_OR, STATE_NOT,
                            STATE_GT, STATE_ADD, STATE_SUB, STATE_INC,
                            STATE_DEC, STATE_IN, STATE_OUT, STATE_MOV,
                            STATE_JMP, STATE_JZ, STATE_JNZ, STATE_NOP,
STATE_HALT);
                --state type set
    SIGNAL CUR_STATE, NX_STATE: STATE_TYPE; --state declare

    SIGNAL aa,bb: STD_LOGIC_VECTOR(1 downto 0);

begin

    aa <= IRoutput(3 downto 2);
    bb <= IRoutput(1 downto 0);



    state_registers : process (RESET, Clock)
    begin
        if RESET = '1' then
            CUR_STATE <= STATE_START;

        elsif rising_edge(Clock) then
            CUR_STATE <= NX_STATE;

        end if;
```

```
    end process;
    --when reset detect, set the state to initial state start



    state_transition : process (IRoutput, CUR_STATE)
    begin
        case CUR_STATE is

            when STATE_START =>

                Jmux <= '1';
                IE <= '0'; WE <= '0'; WA <= "00"; RAE <= '0'; RAA <= "00";
RBE <= '0'; RBA <= "00"; OP <= "000"; ZE <= '0';

                IRload <= '0'; PCload <= '0';

                OE <= '0';

                NX_STATE <= STATE_FETCH;
                --state start operation


            when STATE_FETCH =>

                IE <= '0'; WE <= '0'; WA <= "00"; RAE <= '1'; RAA <= aa;
RBE <= '0'; RBA <= "00"; OP <= "001"; ZE <= '0'; --OE <= '0';

                IRload <= '1'; PCload <= '1'; Jmux <= '1';

                NX_STATE <= STATE_DECODE;
                --state FETCH operation

            when STATE_DECODE =>

                Jmux <= '1';
                WA <= "00"; RAA <= "00"; RBA <= "00"; OP <= "000";

                IRload <= '0'; PCload <= '0';
                IE <= '0'; WE <= '0'; RAE <= '0'; RBE <= '0'; ZE <= '0'; OE
<= '0';

                if (IRoutput(7 downto 4) = "0000") then NX_STATE <=
STATE_AND;
                elsif (IRoutput(7 downto 4) = "0001") then NX_STATE <=
STATE_OR;
                elsif (IRoutput(7 downto 4) = "0010") then NX_STATE <=
STATE_NOT;
                elsif (IRoutput(7 downto 4) = "0011") then NX_STATE <=
STATE_GT;
                elsif (IRoutput(7 downto 4) = "0100") then NX_STATE <=
STATE_ADD;
                elsif (IRoutput(7 downto 4) = "0101") then NX_STATE <=
STATE_SUB;
                elsif (IRoutput(7 downto 4) = "0110") then NX_STATE <=
STATE_INC;
                elsif (IRoutput(7 downto 4) = "0111") then NX_STATE <=
STATE_DEC;
                elsif (IRoutput(7 downto 4) = "1000") then NX_STATE <=
STATE_IN;
```

```
                elsif (IRoutput(7 downto 4) = "1001") then NX_STATE <=
STATE_OUT;
                elsif (IRoutput(7 downto 4) = "1010") then NX_STATE <=
STATE_MOV;
                elsif (IRoutput(7 downto 4) = "1011") then NX_STATE <=
STATE_JMP;
                elsif (IRoutput(7 downto 4) = "1100") then NX_STATE <=
STATE_JZ;
                elsif (IRoutput(7 downto 4) = "1101") then NX_STATE <=
STATE_JNZ;
                elsif (IRoutput(7 downto 4) = "1110") then NX_STATE <=
STATE_NOP;
                elsif (IRoutput(7 downto 4) = "1111") then NX_STATE <=
STATE_HALT;
                end if;
                --state decode operation, execute the instruction of ROM by
process the corresponding state


            when STATE_AND => -- STATE_AND

                IRload <= '0'; Jmux <= '1'; PCload <= '0';
                IE <= '0'; ZE <= '0'; OE <= '0';

                WE <= '1'; WA <= aa; RAE <= '1'; RAA <= aa; RBE <= '1'; RBA
<= bb; OP <= "000";

                NX_STATE <= STATE_FETCH;--

            when STATE_OR =>    --state or

                IRload <= '0'; Jmux <= '1'; PCload <= '0';
                IE <= '0'; ZE <= '0'; OE <= '0';

                WE <= '1'; WA <= aa; RAE <= '1'; RAA <= aa; RBE <= '1'; RBA
<= bb; OP <= "001";

                NX_STATE <= STATE_FETCH;--

            when STATE_NOT =>   --state not

                IRload <= '0'; Jmux <= '1'; PCload <= '0';
                IE <= '0'; RBE <= '0'; RBA <= "00"; ZE <= '0'; OE <= '0';

                WE <= '1'; WA <= aa; RAE <= '1'; RAA <= aa; OP <= "010";

                NX_STATE <= STATE_FETCH;--

            when STATE_GT => --state GT

                IRload <= '0'; Jmux <= '1'; PCload <= '0';
                IE <= '0'; WE <= '0'; WA <= "00"; OE <= '0';

                RAE <= '1'; RAA <= aa; RBE <= '1'; RBA <= bb; OP <= "011";
ZE <= '1';

                NX_STATE <= STATE_FETCH;--

            when STATE_ADD => --state ADD

                IRload <= '0'; Jmux <= '1'; PCload <= '0';
```

```
                IE <= '0'; ZE <= '0'; OE <= '0';

                WE <= '1'; WA <= aa; RAE <= '1'; RAA <= aa; RBE <= '1'; RBA
<= bb; OP <= "100";

                NX_STATE <= STATE_FETCH;--

            when STATE_SUB => --state SUB

                IRload <= '0'; Jmux <= '1'; PCload <= '0';
                IE <= '0'; OE <= '0';

                WE <= '1'; WA <= aa; RAE <= '1'; RAA <= aa; RBE <= '1'; RBA
<= bb; OP <= "101";     ZE <= '1';--??

                NX_STATE <= STATE_FETCH;--

            when STATE_INC => --state INC

                IRload <= '0'; Jmux <= '1'; PCload <= '0';
                IE <= '0'; RBE <= '0'; RBA <= "00"; ZE <= '0'; OE <= '0';

                WE <= '1'; WA <= aa; RAE <= '1'; RAA <= aa; OP <= "110";

                NX_STATE <= STATE_FETCH;--

            when STATE_DEC => --state DEC

                IRload <= '0'; Jmux <= '1'; PCload <= '0';
                IE <= '0'; RBE <= '0'; RBA <= "00"; OE <= '0';

                WE <= '1'; WA <= aa; RAE <= '1'; RAA <= aa; OP <= "111";
ZE <= '1'; --??

                NX_STATE <= STATE_FETCH;--
            ------------------------------------------------------------------
----------------------------------------------
            when STATE_IN => --state IN, input data

                IRload <= '0'; Jmux <= '1'; PCload <= '0';
                RAE <= '0'; RAA <= "00"; RBE <= '0'; RBA <= "00"; OP <=
"000"; ZE <= '0'; OE <= '0';

                IE <= '1'; WE <= '1'; WA <= aa;

                NX_STATE <= STATE_FETCH;

            when STATE_OUT => --state OUT, output data

                IRload <= '0'; Jmux <= '1'; PCload <= '0';
                IE <= '0'; WE <= '0'; WA <= "00"; ZE <= '0';

                RAE <= '1'; RAA <= aa; RBE <= '0'; RBA <= "00"; OP <=
"001"; OE <= '1';

                NX_STATE <= STATE_FETCH;

            when STATE_MOV => --state MOV

                IRload <= '0'; Jmux <= '1'; PCload <= '0';
                IE <= '0'; RAA <= "00"; ZE <= '0'; OE <= '0';
```

```
                WE <= '1'; WA <= aa; RAE <= '0'; RBE <= '1'; RBA <= bb; OP
<= "001";

                NX_STATE <= STATE_FETCH;

            when STATE_JMP => --state JMP, jump

                IE <= '0'; WE <= '0'; WA <= "00"; RAE <= '0'; RAA <= "00";
RBE <= '0'; RBA <= "00"; OP <= "000"; ZE <= '0'; OE <= '0';

                IRload <= '1';
                Jmux <= '0'; PCload <= '1';

                NX_STATE <= STATE_START;

            when STATE_JZ => --state JZ, jump if Z = 1

                IRload <= '0';
                IE <= '0'; WE <= '0'; WA <= "00"; RAE <= '0'; RAA <= "00";
RBE <= '0'; RBA <= "00"; OP <= "000"; OE <= '0';

                    ZE <= '1';
                IF Z = '0' THEN
                    Jmux <= '1';
                ELSE
                    Jmux <= '0';
                    PCload <= '1';
                END IF;

                NX_STATE <= STATE_START;

            when STATE_JNZ => --state JNZ, jump if Z = 0

                IRload <= '0';
                IE <= '0'; WE <= '0'; WA <= "00"; RAE <= '0'; RAA <= "00";
RBE <= '0'; RBA <= "00"; OP <= "000"; OE <= '0';

                    ZE <= '1';
                IF Z = '1' THEN
                    Jmux <= '1';
                ELSE
                    Jmux <= '0';
                    PCload <= '1';
                END IF;

                NX_STATE <= STATE_START;

            when STATE_NOP => --state NOP, do not thing

                --IRload <= '0'; Jmux <= '0'; PCload <= '0';
                --IE <= '0'; WE <= '0'; WA <= "00"; RAE <= '0'; RAA <=
"00"; RBE <= '0'; RBA <= "00"; OP <= "000"; ZE <= '0'; OE <= '0';

                NX_STATE <= STATE_FETCH;

            when STATE_HALT => --state HALT, stop

                --IRload <= '0'; Jmux <= '0'; PCload <= '0';
                --IE <= '0'; WE <= '0'; WA <= "00"; RAE <= '0'; RAA <=
"00"; RBE <= '0'; RBA <= "00"; OP <= "000"; ZE <= '0'; OE <= '0';
```

```
                NX_STATE <= STATE_HALT;

        end case;

    end process;


end Behavioral;
```

Datainput:

```
----------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    18:53:19 03/12/2014
-- Design Name:
-- Module Name:    DataInputUnit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity DataInputUnit is
    port(DataInput: IN STD_LOGIC_VECTOR(7 downto 0);
          q: IN STD_LOGIC_VECTOR(7 downto 0);--data ouput
          IE: IN STD_LOGIC; --datainput enable
          RFinput: OUT STD_LOGIC_VECTOR(7 downto 0));-- transfer data to
register file
end DataInputUnit;
```

```
architecture Behavioral of DataInputUnit is

begin

PROCESS
BEGIN
    IF IE = '1' THEN
        RFinput <= DataInput;
    ELSE
        RFinput <= q;
    END IF;
    --when IE equals 1, receive data from outside, or receive feedback data
END PROCESS;

end Behavioral;
```

Datapath:

```
----------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    04:43:36 03/17/2014
-- Design Name:
-- Module Name:    Datapath - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity Datapath is
    port(DataInput: in STD_LOGIC_VECTOR(7 downto 0);
          IE: IN STD_LOGIC;
          WE: IN STD_LOGIC;                          -- write enable
          WA: IN STD_LOGIC_VECTOR(1 downto 0);  -- write address
```

```
           RAE: IN STD_LOGIC;                              -- read port A
enable
           RAA: IN STD_LOGIC_VECTOR(1 downto 0);-- read port A address
           RBE: IN STD_LOGIC;                              -- read port B
enable
           RBA: IN STD_LOGIC_VECTOR(1 downto 0);-- read port B address
           OP: in  STD_LOGIC_VECTOR (2 downto 0);-- operation code
           ZE: IN STD_LOGIC;      --FF enable
           Clock: in  STD_LOGIC; --clock signal
           Z: OUT STD_LOGIC;
           OE: IN STD_LOGIC;      --OE signal
           DataOutput: OUT STD_LOGIC_VECTOR(7 downto 0);


           PCload: IN STD_LOGIC; --PC enable
           Reset: IN STD_LOGIC;  --reset signal
           IRload: IN STD_LOGIC; --IR unit enable
           IRoutput: OUT STD_LOGIC_VECTOR(7 downto 0);   --instruction send
out
           Jmux: IN STD_LOGIC); --PC address jump or not

end Datapath;

architecture Behavioral of Datapath is

begin

    Datapath1 : entity Datapath1 (Behavioral)
            port map (DataInput => DataInput, IE => IE, WE => WE, WA => WA,
RAE => RAE,
                          RAA => RAA, RBE => RBE, RBA => RBA, OP => OP, ZE
=> ZE, Clock => Clock, Z => Z, OE => OE, DataOutput => DataOutput);
    --Datapath1 entity, which implement the arithmetic instruction

    Datapath2 : entity Datapath2 (Behavioral)
            port map (PCload => PCload, Reset => Reset, Clock => Clock,
IRload => IRload, Jmux => Jmux, IRoutput => IRoutput);
    --Datapath2 entity, which implement the program sequencing flow
instruction


end Behavioral;
```

Datapath1:

```
----------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    16:13:19 03/12/2014
-- Design Name:
-- Module Name:    Datapath1 - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
```

```
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity Datapath1 is
    port(DataInput: in STD_LOGIC_VECTOR(7 downto 0);
          IE: IN STD_LOGIC;

          WE: IN STD_LOGIC;                          -- write enable
          WA: IN STD_LOGIC_VECTOR(1 downto 0);  -- write address
          RAE: IN STD_LOGIC;                              -- read port A
enable
          RAA: IN STD_LOGIC_VECTOR(1 downto 0);-- read port A address
          RBE: IN STD_LOGIC;                              -- read port B
enable
          RBA: IN STD_LOGIC_VECTOR(1 downto 0);-- read port B address

          OP: in  STD_LOGIC_VECTOR (2 downto 0);    --operation code

          ZE: IN STD_LOGIC;
          Clock: in  STD_LOGIC; --clock signal
          Z: OUT STD_LOGIC;

          OE: IN STD_LOGIC;
          DataOutput: OUT STD_LOGIC_VECTOR(7 downto 0));

end Datapath1;


architecture Behavioral of Datapath1 is
    signal RFinput: STD_LOGIC_VECTOR(7 downto 0);
    signal A, B : STD_LOGIC_VECTOR(7 downto 0); --input A and B, controlled
by switch
    signal FFinput: STD_LOGIC;
    signal q : STD_LOGIC_VECTOR(7 downto 0);    --result of the operation
logic with signal A and B

    signal tep:STD_LOGIC_VECTOR(7 downto 0);

begin

    DataInputUnit : entity DataInputUnit (Behavioral)
            port map (DataInput => DataInput, IE => IE, q => q, RFinput =>
RFinput);
    --DataInputUnit entity, which implement the input of data
```

```
    RegFile : entity RegFile (Behavioral)
            port map (RFinput => RFinput, A => A, B => B,
                        Clock => Clock, WE => WE, WA => WA,
                        RAE => RAE, RAA => RAA, RBE => RBE, RBA => RBA);
    --RF entity, which store the operated value

    ALU_unit : entity ALU (Behavioral)
            port map (A => A, B => B, OP => OP, q => q);
    --ALU entity, which implement the logic operation with signal A and B

    NorGate : entity NorGate (Behavioral)
            port map (q => q, FFinput => FFinput);
    --NorGate entity, which implement the norgate function

    FF : entity FF (Behavioral)
            port map (Clock => Clock, FFinput => FFinput, ZE => ZE, Z =>
Z);
    --FF entity, which implement the Z out put

    TriStateBuffer : entity TriStateBuffer (Behavioral)
            port map (OE => OE, q => q, DataOutput => tep);
    --TriStateBuffer entity, which implement the output enable

DataOutput<=tep;

end Behavioral;
```

Datapath2:

```
----------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    16:13:40 03/12/2014
-- Design Name:
-- Module Name:    Datapath2 - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity Datapath2 is
    port(PCload: IN STD_LOGIC; --PC enable
          Reset: IN STD_LOGIC;  --reset signal
          Clock: IN STD_LOGIC;  --clock signal

          IRload: IN STD_LOGIC; --IR enable
          IRoutput: OUT STD_LOGIC_VECTOR(7 downto 0);

          Jmux: IN STD_LOGIC);

end Datapath2;

architecture Behavioral of Datapath2 is
    SIGNAL PCoutput: STD_LOGIC_VECTOR(3 downto 0);
    SIGNAL PCinput: STD_LOGIC_VECTOR(3 downto 0);
    --SIGNAL JumpUnitOutput: STD_LOGIC_VECTOR(3 downto 0);

    SIGNAL UnJumpAddress: STD_LOGIC_VECTOR(3 downto 0);
    SIGNAL JumpAddress: STD_LOGIC_VECTOR(3 downto 0);

    SIGNAL ROMoutput: STD_LOGIC_VECTOR(7 downto 0);

begin

    PC : entity PC (Behavioral)
            port map (PCload => PCload, Reset => Reset, Clock => Clock,
                        PCoutput => PCoutput, PCinput => PCinput);
    --PC entity, which implement the PC address send out

    PCincrement : entity PCincrement (Behavioral)
            port map (PCoutput => PCoutput, UnJumpAddress =>
UnJumpAddress);
    --PCincrement entity, which implement the increament of previous PC
address

    ROMfile : entity ROMfile (Behavioral)
            port map (Clock => Clock, PCoutput => PCoutput, ROMoutput =>
ROMoutput);
    --ROMfile entity, which store the instructions

    IR : entity IR (Behavioral)
            port map (Clock => Clock, IRload => IRload, ROMoutput =>
ROMoutput,
                        IRoutput => IRoutput, JumpAddress => JumpAddress);
    --IR entity, which implement the next instruction send out

    JumpUnit : entity JumpUnit (Behavioral)
            port map (JumpAddress => JumpAddress, UnJumpAddress =>
UnJumpAddress,
                        Jmux => Jmux, PCinput => PCinput);
    --JumpUnit entity, which implement the choose of jump address or point
to next address



end Behavioral;
```

Display:

```
----------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    13:53:08 02/27/2014
-- Design Name:
-- Module Name:    Display - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity Display is
        Port ( CATHODES : out STD_LOGIC_VECTOR (6 downto 0);--signal
control seven segments
               ANODES : out STD_LOGIC_VECTOR (3 downto 0);--signal control
four seven segments display
               DP : out STD_LOGIC;--display the decimal point
               D0, D1, D2, D3 : IN UNSIGNED (3 downto 0);--signal control
all the seven segments display
               --CK : in STD_LOGIC;
               MCLK : in  STD_LOGIC);--MCLK clock
end Display;

architecture Behavioral of Display is
    SIGNAL disp_ck : STD_LOGIC;
    SIGNAL fr_cnt : UNSIGNED (10 DOWNTO 0);
begin
    four_digits_unit : ENTITY four_digits(Behavioral)
        PORT MAP (d3 => d3, d2 => d2, d1 => d1, d0 => d0, DP => DP,
        ck => disp_ck, cathodes => cathodes, anodes => anodes);
    --four digits display used for display the output of the program

    fr_1kHz : PROCESS
```

```
    BEGIN
    WAIT UNTIL RISING_EDGE(mclk);
    disp_ck <= fr_cnt(fr_cnt'HIGH);
    fr_cnt <= fr_cnt + 1;
    END PROCESS;
    --divide the mclk clock, which used for scan and update the four digits
display

end Behavioral;
```

FF:

```
----------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    13:42:55 03/06/2014
-- Design Name:
-- Module Name:    FF - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity FF is
    port(Clock: IN STD_LOGIC;--clock signal
          ZE: IN STD_LOGIC;--ze signal, which enable the FF unit
          FFinput: IN STD_LOGIC;--norgate output
          Z: OUT STD_LOGIC);--Z
end FF;

architecture Behavioral of FF is

begin
    PROCESS
    BEGIN
        WAIT UNTIL RISING_EDGE(Clock);
```

```
        IF ZE = '1' THEN
            Z <= FFinput;
        END IF;
        --when ZE equals 1, Z receive the data from norgate
    END PROCESS;

end Behavioral;
```

Four digits unit:

```
----------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    15:03:39 01/28/2014
-- Design Name:
-- Module Name:    four_digits_unit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity four_digits is
    Port (D0, D1, D2, D3 : in  UNSIGNED (3 downto 0);--signal control all
the seven segments display
           CK : in  STD_LOGIC;--divided MCLK clock
           DP : out  STD_LOGIC;--decimal point
           CATHODES : out STD_LOGIC_VECTOR (6 downto 0);--signal control
seven segments
           ANODES   : out STD_LOGIC_VECTOR (3 downto 0));--signal control
four seven segments display
end four_digits;

architecture Behavioral of four_digits is
    signal DIGIT : UNSIGNED (3 downto 0);--digit signal which present the
output value in binary type
```

```
    signal cnt : UNSIGNED (1 downto 0);-- cnt which indicate the clock
runing

begin

    one_digit_unit : entity one_digit(Behavioral)
        port map (DIGIT => DIGIT,
            CATHODES => CATHODES);
-- one_digit_unit calling.

    PROCESS
    BEGIN

        WAIT UNTIL RISING_EDGE(CK);
        cnt <= cnt + 1;

    END PROCESS;
-- cnt which indicate the clock runing

    PROCESS
    BEGIN
        CASE cnt IS
        WHEN "00" =>
            DIGIT <= D0; ANODES <= "1110"; DP <= '1';
        WHEN "01" =>
            DIGIT <= D1; ANODES <= "1101"; DP <= '1';
        WHEN "10" =>
            DIGIT <= D2; ANODES <= "1011"; DP <= '1';
        WHEN OTHERS =>
            DIGIT <= D3; ANODES <= "0111"; DP <= '1';
        END CASE;

    END PROCESS;
-- Display different seven segments display when cnt change

end Behavioral;
```

IR (instruction register):

```
----------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    13:41:53 03/06/2014
-- Design Name:
-- Module Name:    IR - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
```

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

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

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

entity IR is
    port(Clock: IN STD_LOGIC; --clock signal
         IRload: IN STD_LOGIC; --IR unit enable signal
          ROMoutput: IN STD_LOGIC_VECTOR(7 downto 0); -- next instruction
          JumpAddress: OUT STD_LOGIC_VECTOR(3 downto 0); --PC increament
address, not jump address
          IRoutput: OUT STD_LOGIC_VECTOR(7 downto 0)); --next instruction
send to control unit
end IR;

architecture Behavioral of IR is

begin
    PROCESS
    BEGIN
        WAIT UNTIL RISING_EDGE(Clock); -- detect a clock
        IF IRload = '1' THEN
            JumpAddress <= ROMoutput(3 downto 0);
            IRoutput <= ROMoutput;
            --when IR enabled, send out next instruction and send the jump
address to PC
        END IF;
    END PROCESS;

end Behavioral;
```

Jumpunit:

```
----------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    18:44:02 03/12/2014
-- Design Name:
-- Module Name:    JumpUnit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
```

```
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity JumpUnit is
    port(JumpAddress: IN STD_LOGIC_VECTOR(3 downto 0);
          UnJumpAddress: IN STD_LOGIC_VECTOR(3 downto 0);
          Jmux: IN STD_LOGIC;
          PCinput: OUT STD_LOGIC_VECTOR(3 downto 0));
end JumpUnit;

architecture Behavioral of JumpUnit is

begin


    WITH Jmux SELECT
        PCinput <= JumpAddress WHEN '0',
                   UnJumpAddress WHEN OTHERS;
    --when jmux signal equals 0, PC jump with jump address, otherwise
increament itself

end Behavioral;
```

Norgate:

```
----------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    19:19:18 03/12/2014
-- Design Name:
-- Module Name:    NorGate - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
```

```
-- Additional Comments:
--
----------------------------------------------------------------------------
-------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity NorGate is
    port(q: IN STD_LOGIC_VECTOR(7 downto 0);--data output
          FFinput: OUT STD_LOGIC);--norgate output

end NorGate;

architecture Behavioral of NorGate is

begin
    PROCESS
    BEGIN
        IF q = "00000000" THEN
            FFinput <= '1';
        ELSE
            FFinput <= '0';
        END IF;
        --when q equals to 0, FF send out 1,or send out 0
    END PROCESS;

end Behavioral;
```

One digit:

```
----------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    13:28:38 01/16/2014
-- Design Name:
-- Module Name:    one_digit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
```

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

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

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

entity one_digit is
    Port ( DIGIT : in  UNSIGNED (3 downto 0);--digit signal which present
the output value in binary type
           CATHODES : out  STD_LOGIC_VECTOR (6 downto 0));--signal control
seven segments
end one_digit;

architecture Behavioral of one_digit is

begin
    WITH DIGIT SELECT
        CATHODES <= "1000000" WHEN "0000",--seven segments display shows 0
                    "1111001" WHEN "0001",--seven segments display
shows 1
                    "0100100" WHEN "0010",--seven segments display
shows 2
                    "0110000" WHEN "0011",--seven segments display
shows 3
                    "0011001" WHEN "0100",--seven segments display
shows 4
                    "0010010" WHEN "0101",--seven segments display
shows 5
                    "0000010" WHEN "0110",--seven segments display
shows 6
                    "1111000" WHEN "0111",--seven segments display
shows 7
                    "0000000" WHEN "1000",--seven segments display
shows 8
                    "0010000" WHEN "1001",--seven segments display
shows 9
                    "0001000" WHEN "1010",--seven segments display
shows A
                    "0000000" WHEN "1011",--seven segments display
shows B
                    "1000110" WHEN "1100",--seven segments display
shows C
                    "1000000" WHEN "1101",--seven segments display
shows D
                    "0000110" WHEN "1110",--seven segments display
shows E
                    "0001110" WHEN OTHERS;--seven segments display
shows F

end Behavioral;
```

OP:

```
----------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    13:38:53 02/20/2014
-- Design Name:
-- Module Name:    OP - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity OP is
    Port ( BTN : in  STD_LOGIC_VECTOR (3 downto 0);
           A : in  UNSIGNED (7 downto 0);
              B : in  UNSIGNED (7 downto 0);
              --LED : out  UNSIGNED (7 downto 0);
              q : out UNSIGNED (7 downto 0));
end OP;

architecture Behavioral of OP is

begin


    PROCESS
    BEGIN
--  IF BTN = "0000" THEN LED <= A and B;
--  ELSIF BTN = "0001" THEN LED <= A or B;
--  ELSIF BTN = "0010" THEN LED <= not A;
--  ELSIF BTN = "0011" and A > B THEN LED <= "11111111";
--  ELSIF BTN = "0100" THEN LED <= A + B;
--  ELSIF BTN = "0101" THEN LED <= A - B;
--  ELSIF BTN = "0110" THEN LED <= A - 1;
--  ELSIF BTN = "0111" THEN LED <= A + 1;
--  END IF;
```

```
    IF BTN = "0000" THEN q <= A and B;
    ELSIF BTN = "0001" THEN q <= A or B;
    ELSIF BTN = "0010" THEN q <= not A;
    ELSIF BTN = "0011" and A > B THEN q <= "11111111";
    ELSIF BTN = "0100" THEN q <= A + B;
    ELSIF BTN = "0101" THEN q <= A - B;
    ELSIF BTN = "0110" THEN q <= A + 1;
    ELSIF BTN = "0111" THEN q <= A - 1;
    END IF;

    END PROCESS;


end Behavioral;
```

PC:

```
----------------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    13:42:09 03/06/2014
-- Design Name:
-- Module Name:    PC - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------
-------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity PC is
    port(PCload: IN STD_LOGIC; --enable the PC to get address from input
          Reset: IN STD_LOGIC;  --reset signal
          Clock: IN STD_LOGIC;  --clock signal
          PCoutput: OUT STD_LOGIC_VECTOR(3 downto 0); --next instruction
address
          PCinput: IN STD_LOGIC_VECTOR(3 downto 0)); --PC input
end PC;
```

```
architecture Behavioral of PC is
--  SIGNAL counter: STD_LOGIC_VECTOR(3 downto 0);
begin
    PROCESS
    BEGIN
        WAIT UNTIL RISING_EDGE(Clock); --detect a clock

        IF Reset = '1' THEN
            PCoutput <= "0000"; -- when reset detected, set the PC address
to 0
        ELSIF PCload = '1' THEN
            PCoutput <= PCinput; --when no reset and PC enabled, load the
value of PC input
        END IF;


    END PROCESS;

end Behavioral;
```

PC increment:

```
----------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    19:05:41 03/12/2014
-- Design Name:
-- Module Name:    PCincrement - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity PCincrement is
    port(PCoutput: IN STD_LOGIC_VECTOR(3 downto 0); --PC address
```

```
          UnJumpAddress: OUT STD_LOGIC_VECTOR(3 downto 0)); --Output of
increment units
end PCincrement;

architecture Behavioral of PCincrement is

begin
    UnJumpAddress <= (STD_LOGIC_VECTOR(UNSIGNED(PCoutput) + 1));
    --when no jump is required, increase PC address and extract the next
instruction

end Behavioral;
```

RF:

```
----------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    13:42:37 03/06/2014
-- Design Name:
-- Module Name:    RF - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity RegFile is
    port(Clock: IN STD_LOGIC;
          WE: IN STD_LOGIC;                          -- write enable
          WA: IN STD_LOGIC_VECTOR(1 downto 0);  -- write address
          RAE: IN STD_LOGIC;                              -- read port A
enable
          RAA: IN STD_LOGIC_VECTOR(1 downto 0);-- read port A address
```

```
          RBE: IN STD_LOGIC;                              -- read port B
enable
          RBA: IN STD_LOGIC_VECTOR(1 downto 0);-- read port B address
          RFinput: IN STD_LOGIC_VECTOR(7 downto 0);
          A, B: OUT STD_LOGIC_VECTOR(7 downto 0));
end RegFile;

architecture Behavioral of RegFile is
    SUBTYPE reg IS STD_LOGIC_VECTOR(7 downto 0);
    TYPE regArray IS ARRAY(0 to 3) OF reg;
    SIGNAL RF: regArray;
begin

    WritePort: PROCESS
    BEGIN
        WAIT UNTIL RISING_EDGE(Clock);
        IF WE = '1' THEN
            RF(TO_INTEGER(UNSIGNED(WA))) <= RFinput;
        END IF;
        --when write enable, write input data to register file location
    END PROCESS;

    --Read Port A
    A <= RF(TO_INTEGER(UNSIGNED(RAA))) WHEN RAE = '1'
        ELSE (OTHERS => '0');

    --Read Port B
    B <= RF(TO_INTEGER(UNSIGNED(RBA))) WHEN RBE = '1'
        ELSE (OTHERS => '0');

end Behavioral;
```

ROM:

```
----------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    13:42:24 03/06/2014
-- Design Name:
-- Module Name:    ROM - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
```

```
-- 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 primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity ROMfile is
    port(Clock: IN STD_LOGIC;   --clock signal
          PCoutput: IN STD_LOGIC_VECTOR(3 downto 0); --PC address
          ROMoutput: OUT STD_LOGIC_VECTOR(7 downto 0)); --exstracted
instruction
end ROMfile;

architecture Behavioral of ROMfile is
    SUBTYPE unitIR IS STD_LOGIC_VECTOR(7 downto 0); --8 bits for each
location of ROM
    TYPE IRarray IS ARRAY(0 to 15) OF unitIR; --16 locations/address for
one ROM

    --TASK1
--  CONSTANT ROM : IRarray := (
--        "10000000", --            IN B
--        "01000100", -- loop:  ADD A B
--        "01110000", --            DEC B
--        "11010001", --            JNZ loop
--        "10010100", --            OUT A
--        "11111111", --            halt
--        "11111111", --            halt
--        "11111111", --            halt
--        "11111111", --            halt
--        "11111111", --            halt
--        "11111111", --            halt
--        "11111111", --            halt
--        "11111111", --            halt
--        "11111111", --            halt
--        "11111111"  --            halt
--  );

    --TASK2
--  CONSTANT ROM : IRarray := (
--          "10000000", --          IN A
--          "11101111", --          NOP
--          "01100100", --          INC B
--          "01100100", --          INC B
--          "01100100", --          INC B
--          "01100100", --          INC B
--          "01100100", --          INC B
--          "00110100", -- loop:    GT B,A
--          "11011110", --          JNZ end2
--          "01010001", --          SUB A,B
--          "11001100", --          JZ  end1
--          "10110111", --          JMP loop
--          "10011000", -- end1:    OUT C
--          "11111111", --          HALT
--          "10010000", -- end2:    OUT A
--          "11111111"  --          HALT
--  );
```

```
    --TASK3 2.0
    CONSTANT ROM : IRarray := (
            "10001100", --          IN B3 (N)
            "00001111", --          AND B3 B3
            "11001011", --          JZ end
            "01111100", --          DEC B3
            "11001011", --          JZ end
            "01101000",  --             INC B2
            "01001000", -- loop:        ADD B2 B0
            "10100001", --          MOV B0 B1
            "10100110", --          MOV B1 B2
            "01111100", --          DEC B3
            "11010110", --         JNZ loop
            "10011000", -- end:     OUT B2
            "11111111", --         HALT
            "11111111", --          HALT
            "11111111", --          HALT
            "11111111"  --          HALT
    );


--      --TASK3 1.0
--  CONSTANT ROM : IRarray := (
--          "10000000", --          IN A0 (N)
--          "01101000", --          INC A2
--          "01110000", --          DEC A0
--          "01110000", --          DEC A0
--          "11001110", --          JZ end2
--          "01000110", -- loop:        ADD A1 A2
--          "01110000", --          DEC A0
--          "11001100", --          JZ end1
--          "01001001", --          ADD A2 A1
--          "01110000", --         DEC A0
--          "11001110", --          JZ end2
--          "10110101", --         JMP loop
--          "10010100", -- end1:        OUT A1
--          "11111111",  --             HALT
--          "10011000", -- end2:        OUT A2
--          "11111111"  --          HALT
--  );


begin
    PROCESS
    BEGIN
        WAIT UNTIL RISING_EDGE(Clock);
            ROMoutput <= ROM(TO_INTEGER(UNSIGNED(PCoutput)));
            --as clock detect, extract the instruction with the PC address

    END PROCESS;


end Behavioral;
```

TristateBuffer:

```
----------------------------------------------------------------------------
-------
-- Company:
-- Engineer:
--
-- Create Date:    19:23:16 03/12/2014
-- Design Name:
-- Module Name:    TriStateBuffer - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------
-------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

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

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

entity TriStateBuffer is
    port(OE: IN STD_LOGIC;--oe signal
          q: IN STD_LOGIC_VECTOR(7 downto 0);--data output
          DataOutput: OUT STD_LOGIC_VECTOR(7 downto 0));--data send out

end TriStateBuffer;

architecture Behavioral of TriStateBuffer is

begin
    PROCESS
    BEGIN
        IF OE = '1' THEN
            DataOutput <= q;
        END IF;
        --when oe = 1, send out data
    END PROCESS;

end Behavioral;
```

Repository:

Revision: 19
Author: yyea
Date: 01:45:23, 20 March 2014
Message:
totally complete the whole things.

Modified : /Assignment2/ALU/Assignment1.gise
Modified : /Assignment2/ALU/ControlUnit.vhd
Modified : /Assignment2/ALU/PC.vhd
Modified : /Assignment2/ALU/ROM.vhd
Modified : /Assignment2/ALU/TriStateBuffer.vhd
Modified : /Assignment2/ALU/_ngo/netlist.lst
Modified : /Assignment2/ALU/_xmsgs/bitgen.xmsgs
Modified : /Assignment2/ALU/_xmsgs/map.xmsgs
Modified : /Assignment2/ALU/_xmsgs/par.xmsgs
Modified : /Assignment2/ALU/_xmsgs/xst.xmsgs
Modified : /Assignment2/ALU/main.bgn
Modified : /Assignment2/ALU/main.bit
Modified : /Assignment2/ALU/main.bld
Modified : /Assignment2/ALU/main.cmd_log
Modified : /Assignment2/ALU/main.drc
Modified : /Assignment2/ALU/main.ncd
Modified : /Assignment2/ALU/main.ngc
Modified : /Assignment2/ALU/main.ngd
Modified : /Assignment2/ALU/main.ngr
Modified : /Assignment2/ALU/main.pad
Modified : /Assignment2/ALU/main.par
Modified : /Assignment2/ALU/main.pcf
Modified : /Assignment2/ALU/main.ptwx
Modified : /Assignment2/ALU/main.syr
Modified : /Assignment2/ALU/main.twr
Modified : /Assignment2/ALU/main.twx
Modified : /Assignment2/ALU/main.unroutes
Modified : /Assignment2/ALU/main.vhd
Modified : /Assignment2/ALU/main_guide.ncd
Modified : /Assignment2/ALU/main_map.map
Modified : /Assignment2/ALU/main_map.mrp
Modified : /Assignment2/ALU/main_map.ncd
Modified : /Assignment2/ALU/main_map.ngm
Modified : /Assignment2/ALU/main_map.xrpt
Modified : /Assignment2/ALU/main_ngdbuild.xrpt
Modified : /Assignment2/ALU/main_pad.csv
Modified : /Assignment2/ALU/main_pad.txt
Modified : /Assignment2/ALU/main_par.xrpt
Modified : /Assignment2/ALU/main_summary.html
Modified : /Assignment2/ALU/main_summary.xml
Modified : /Assignment2/ALU/main_usage.xml

Modified : /Assignment2/ALU/main_xst.xrpt
Modified : /Assignment2/ALU/usage_statistics_webtalk.html
Modified : /Assignment2/ALU/webtalk.log
Modified : /Assignment2/ALU/webtalk_pn.xml
Modified : /Assignment2/ALU/xlnx_auto_0_xdb/cst.xbcd
Modified : /Assignment2/ALU/xst/work/hdllib.ref
Modified : /Assignment2/ALU/xst/work/hdpdeps.ref
Modified : /Assignment2/ALU/xst/work/sub00/vhpl08.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl09.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl12.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl13.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl16.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl17.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl30.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl31.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl36.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl37.vho

Revision: 18
Author: yyea
Date: 16:59:41, 19 March 2014
Message:
solve the jump bug and program for the tasks.

Modified : /Assignment2/ALU/Assignment1.gise
Modified : /Assignment2/ALU/PC.vhd
Modified : /Assignment2/ALU/ROM.vhd
Modified : /Assignment2/ALU/_ngo/netlist.lst
Modified : /Assignment2/ALU/_xmsgs/par.xmsgs
Modified : /Assignment2/ALU/_xmsgs/pn_parser.xmsgs
Modified : /Assignment2/ALU/_xmsgs/xst.xmsgs
Modified : /Assignment2/ALU/iseconfig/main.xreport
Modified : /Assignment2/ALU/main.bgn
Modified : /Assignment2/ALU/main.bit
Modified : /Assignment2/ALU/main.bld
Modified : /Assignment2/ALU/main.cmd_log
Modified : /Assignment2/ALU/main.drc
Modified : /Assignment2/ALU/main.ncd
Modified : /Assignment2/ALU/main.ngc
Modified : /Assignment2/ALU/main.ngd
Modified : /Assignment2/ALU/main.ngr
Modified : /Assignment2/ALU/main.pad
Modified : /Assignment2/ALU/main.par
Modified : /Assignment2/ALU/main.pcf
Modified : /Assignment2/ALU/main.ptwx
Modified : /Assignment2/ALU/main.syr
Modified : /Assignment2/ALU/main.twr
Modified : /Assignment2/ALU/main.twx
Modified : /Assignment2/ALU/main.unroutes

Modified : /Assignment2/ALU/main_envsettings.html
Modified : /Assignment2/ALU/main_guide.ncd
Modified : /Assignment2/ALU/main_map.map
Modified : /Assignment2/ALU/main_map.mrp
Modified : /Assignment2/ALU/main_map.ncd
Modified : /Assignment2/ALU/main_map.ngm
Modified : /Assignment2/ALU/main_map.xrpt
Modified : /Assignment2/ALU/main_ngdbuild.xrpt
Modified : /Assignment2/ALU/main_pad.csv
Modified : /Assignment2/ALU/main_pad.txt
Modified : /Assignment2/ALU/main_par.xrpt
Modified : /Assignment2/ALU/main_summary.html
Modified : /Assignment2/ALU/main_summary.xml
Modified : /Assignment2/ALU/main_usage.xml
Modified : /Assignment2/ALU/main_xst.xrpt
Modified : /Assignment2/ALU/usage_statistics_webtalk.html
Modified : /Assignment2/ALU/webtalk.log
Modified : /Assignment2/ALU/webtalk_pn.xml
Modified : /Assignment2/ALU/xlnx_auto_0_xdb/cst.xbcd
Modified : /Assignment2/ALU/xst/work/hdllib.ref
Modified : /Assignment2/ALU/xst/work/hdpdeps.ref
Modified : /Assignment2/ALU/xst/work/sub00/vhpl12.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl13.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl16.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl17.vho

Revision: 17
Author: yyea
Date: 15:01:07, 18 March 2014
Message:
still debug for jump state

Modified : /Assignment2/ALU/Assignment1.gise
Modified : /Assignment2/ALU/ControlUnit.vhd
Modified : /Assignment2/ALU/Datapath1.vhd
Modified : /Assignment2/ALU/JumpUnit.vhd
Modified : /Assignment2/ALU/PC.vhd
Modified : /Assignment2/ALU/ROM.vhd
Modified : /Assignment2/ALU/_ngo/netlist.lst
Modified : /Assignment2/ALU/_xmsgs/bitgen.xmsgs
Modified : /Assignment2/ALU/_xmsgs/map.xmsgs
Modified : /Assignment2/ALU/_xmsgs/par.xmsgs
Modified : /Assignment2/ALU/_xmsgs/xst.xmsgs
Modified : /Assignment2/ALU/iseconfig/Assignment1.projectmgr
Modified : /Assignment2/ALU/iseconfig/main.xreport
Modified : /Assignment2/ALU/main.bgn
Modified : /Assignment2/ALU/main.bit
Modified : /Assignment2/ALU/main.bld
Modified : /Assignment2/ALU/main.cmd_log

Modified : /Assignment2/ALU/main.drc
Modified : /Assignment2/ALU/main.ncd
Modified : /Assignment2/ALU/main.ngc
Modified : /Assignment2/ALU/main.ngd
Modified : /Assignment2/ALU/main.ngr
Modified : /Assignment2/ALU/main.pad
Modified : /Assignment2/ALU/main.par
Modified : /Assignment2/ALU/main.pcf
Modified : /Assignment2/ALU/main.ptwx
Modified : /Assignment2/ALU/main.syr
Modified : /Assignment2/ALU/main.twr
Modified : /Assignment2/ALU/main.twx
Modified : /Assignment2/ALU/main.unroutes
Modified : /Assignment2/ALU/main_envsettings.html
Modified : /Assignment2/ALU/main_guide.ncd
Modified : /Assignment2/ALU/main_map.map
Modified : /Assignment2/ALU/main_map.mrp
Modified : /Assignment2/ALU/main_map.ncd
Modified : /Assignment2/ALU/main_map.ngm
Modified : /Assignment2/ALU/main_map.xrpt
Modified : /Assignment2/ALU/main_ngdbuild.xrpt
Modified : /Assignment2/ALU/main_pad.csv
Modified : /Assignment2/ALU/main_pad.txt
Modified : /Assignment2/ALU/main_par.xrpt
Modified : /Assignment2/ALU/main_summary.html
Modified : /Assignment2/ALU/main_summary.xml
Modified : /Assignment2/ALU/main_usage.xml
Modified : /Assignment2/ALU/main_xst.xrpt
Modified : /Assignment2/ALU/usage_statistics_webtalk.html
Modified : /Assignment2/ALU/webtalk.log
Modified : /Assignment2/ALU/webtalk_pn.xml
Modified : /Assignment2/ALU/xlnx_auto_0_xdb/cst.xbcd
Modified : /Assignment2/ALU/xst/work/hdllib.ref
Modified : /Assignment2/ALU/xst/work/hdpdeps.ref
Modified : /Assignment2/ALU/xst/work/sub00/vhpl12.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl13.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl16.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl17.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl20.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl21.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl32.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl33.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl36.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl37.vho

Revision: 16
Author: yyea
Date: 07:27:40, 18 March 2014
Message:

debug the microprocessor

Modified : /Assignment2/ALU/ALU.vhd
Modified : /Assignment2/ALU/Assignment1.gise
Modified : /Assignment2/ALU/ControlUnit.vhd
Modified : /Assignment2/ALU/DataInputUnit.vhd
Modified : /Assignment2/ALU/Datapath.vhd
Modified : /Assignment2/ALU/Datapath1.vhd
Modified : /Assignment2/ALU/Datapath2.vhd
Modified : /Assignment2/ALU/FF.vhd
Modified : /Assignment2/ALU/IR.vhd
Modified : /Assignment2/ALU/JumpUnit.vhd
Modified : /Assignment2/ALU/NorGate.vhd
Modified : /Assignment2/ALU/PC.vhd
Modified : /Assignment2/ALU/PCincrement.vhd
Modified : /Assignment2/ALU/RF.vhd
Modified : /Assignment2/ALU/ROM.vhd
Modified : /Assignment2/ALU/TriStateBuffer.vhd
Modified : /Assignment2/ALU/_ngo/netlist.lst
Modified : /Assignment2/ALU/_xmsgs/bitgen.xmsgs
Modified : /Assignment2/ALU/_xmsgs/map.xmsgs
Modified : /Assignment2/ALU/_xmsgs/par.xmsgs
Modified : /Assignment2/ALU/_xmsgs/pn_parser.xmsgs
Modified : /Assignment2/ALU/_xmsgs/xst.xmsgs
Modified : /Assignment2/ALU/iseconfig/Assignment1.projectmgr
Modified : /Assignment2/ALU/main.bgn
Modified : /Assignment2/ALU/main.bit
Modified : /Assignment2/ALU/main.bld
Modified : /Assignment2/ALU/main.cmd_log
Modified : /Assignment2/ALU/main.drc
Modified : /Assignment2/ALU/main.ncd
Modified : /Assignment2/ALU/main.ngc
Modified : /Assignment2/ALU/main.ngd
Modified : /Assignment2/ALU/main.ngr
Modified : /Assignment2/ALU/main.pad
Modified : /Assignment2/ALU/main.par
Modified : /Assignment2/ALU/main.pcf
Modified : /Assignment2/ALU/main.ptwx
Added : /Assignment2/ALU/main.stx
Modified : /Assignment2/ALU/main.syr
Modified : /Assignment2/ALU/main.twr
Modified : /Assignment2/ALU/main.twx
Modified : /Assignment2/ALU/main.unroutes
Modified : /Assignment2/ALU/main.vhd
Modified : /Assignment2/ALU/main_bitgen.xwbt
Modified : /Assignment2/ALU/main_envsettings.html
Modified : /Assignment2/ALU/main_guide.ncd
Modified : /Assignment2/ALU/main_map.map
Modified : /Assignment2/ALU/main_map.mrp

Modified : /Assignment2/ALU/main_map.ncd
Modified : /Assignment2/ALU/main_map.ngm
Modified : /Assignment2/ALU/main_map.xrpt
Modified : /Assignment2/ALU/main_ngdbuild.xrpt
Modified : /Assignment2/ALU/main_pad.csv
Modified : /Assignment2/ALU/main_pad.txt
Modified : /Assignment2/ALU/main_par.xrpt
Modified : /Assignment2/ALU/main_summary.html
Modified : /Assignment2/ALU/main_summary.xml
Modified : /Assignment2/ALU/main_usage.xml
Modified : /Assignment2/ALU/main_xst.xrpt
Modified : /Assignment2/ALU/usage_statistics_webtalk.html
Modified : /Assignment2/ALU/webtalk.log
Modified : /Assignment2/ALU/webtalk_pn.xml
Modified : /Assignment2/ALU/xlnx_auto_0_xdb/cst.xbcd
Modified : /Assignment2/ALU/xst/work/hdllib.ref
Modified : /Assignment2/ALU/xst/work/hdpdeps.ref
Modified : /Assignment2/ALU/xst/work/sub00/vhpl06.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl07.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl08.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl09.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl12.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl13.vho
Modified : /Assignment2/ALU/xst/work/sub00/vhpl14.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl15.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl16.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl17.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl18.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl19.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl20.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl21.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl22.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl23.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl24.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl25.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl26.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl27.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl28.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl29.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl30.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl31.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl32.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl33.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl34.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl35.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl36.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl37.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl38.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl39.vho

Revision: 15
Author: yyea
Date: 23:09:58, 17 March 2014
Message:
debug the whole program

Modified : /Assignment2/ALU/Assignment1.gise
Modified : /Assignment2/ALU/Assignment1.xise
Modified : /Assignment2/ALU/ControlUnit.vhd
Modified : /Assignment2/ALU/FF.vhd
Modified : /Assignment2/ALU/NorGate.vhd
Modified : /Assignment2/ALU/PC.vhd
Modified : /Assignment2/ALU/TriStateBuffer.vhd
Modified : /Assignment2/ALU/_xmsgs/pn_parser.xmsgs
Modified : /Assignment2/ALU/_xmsgs/xst.xmsgs
Modified : /Assignment2/ALU/iseconfig/Assignment1.projectmgr
Modified : /Assignment2/ALU/iseconfig/main.xreport
Modified : /Assignment2/ALU/main.cmd_log
Modified : /Assignment2/ALU/main.prj
Deleted : /Assignment2/ALU/main.stx
Modified : /Assignment2/ALU/main.syr
Modified : /Assignment2/ALU/main.vhd
Modified : /Assignment2/ALU/main_envsettings.html
Modified : /Assignment2/ALU/main_summary.html
Modified : /Assignment2/ALU/main_vhdl.prj
Modified : /Assignment2/ALU/main_xst.xrpt
Modified : /Assignment2/ALU/webtalk_pn.xml
Modified : /Assignment2/ALU/xst/work/hdllib.ref
Modified : /Assignment2/ALU/xst/work/hdpdeps.ref
Added : /Assignment2/ALU/xst/work/sub00/vhpl12.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl13.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl14.vho

Revision: 14
Author: yyea
Date: 05:31:07, 17 March 2014
Message:
combine controllunit and datapath in one main entity

Modified : /Assignment2/ALU/ControlUnit.vhd
Added : /Assignment2/ALU/Datapath.vhd
Modified : /Assignment2/ALU/Datapath2.vhd
Modified : /Assignment2/ALU/_xmsgs/pn_parser.xmsgs
Modified : /Assignment2/ALU/main.vhd

Revision: 13
Author: yyea
Date: 04:43:10, 17 March 2014

Message:
finish the controll unit

Modified : /Assignment2/ALU/ControlUnit.vhd
Modified : /Assignment2/ALU/_xmsgs/pn_parser.xmsgs
Modified : /Assignment2/ALU/iseconfig/Assignment1.projectmgr
Modified : /Assignment2/ALU/iseconfig/main.xreport
Modified : /Assignment2/ALU/main_summary.html

Revision: 12
Author: yyea
Date: 14:05:12, 13 March 2014
Message:
More or less finish combine the datapath 1,2 and start the control unit
---
Modified : /Assignment2/ALU/Assignment1.xise
Added : /Assignment2/ALU/ControlUnit.vhd
Modified : /Assignment2/ALU/DataInputUnit.vhd
Modified : /Assignment2/ALU/Datapath1.vhd
Modified : /Assignment2/ALU/Datapath2.vhd
Modified : /Assignment2/ALU/JumpUnit.vhd
Modified : /Assignment2/ALU/ROM.vhd
Modified : /Assignment2/ALU/TriStateBuffer.vhd
Modified : /Assignment2/ALU/_xmsgs/pn_parser.xmsgs
Modified : /Assignment2/ALU/iseconfig/Assignment1.projectmgr
Modified : /Assignment2/ALU/iseconfig/main.xreport
Modified : /Assignment2/ALU/main_summary.html

Revision: 11
Author: yyea
Date: 20:29:09, 12 March 2014
Message:
Combine the entities of datapath 1

Modified : /Assignment2/ALU/Datapath1.vhd
Modified : /Assignment2/ALU/iseconfig/Assignment1.projectmgr
Modified : /Assignment2/ALU/iseconfig/main.xreport
Modified : /Assignment2/ALU/main_summary.html

Revision: 10
Author: yyea
Date: 20:24:35, 12 March 2014
Message:

Modified : /Assignment2/ALU/ALU.vhd
Modified : /Assignment2/ALU/Assignment1.xise
Added : /Assignment2/ALU/DataInputUnit.vhd
Modified : /Assignment2/ALU/Datapath1.vhd

Modified : /Assignment2/ALU/FF.vhd
Added : /Assignment2/ALU/JumpUnit.vhd
Added : /Assignment2/ALU/NorGate.vhd
Modified : /Assignment2/ALU/PC.vhd
Added : /Assignment2/ALU/PCincrement.vhd
Modified : /Assignment2/ALU/RF.vhd
Added : /Assignment2/ALU/TriStateBuffer.vhd
Modified : /Assignment2/ALU/_xmsgs/pn_parser.xmsgs
Modified : /Assignment2/ALU/iseconfig/Assignment1.projectmgr

Revision: 9
Author: yyea
Date: 18:37:23, 12 March 2014
Message:
Fill the ROM, RF, FF, IR etc entities

Modified : /Assignment2/ALU/FF.vhd
Modified : /Assignment2/ALU/IR.vhd
Modified : /Assignment2/ALU/PC.vhd
Modified : /Assignment2/ALU/RF.vhd
Modified : /Assignment2/ALU/ROM.vhd
Modified : /Assignment2/ALU/_xmsgs/pn_parser.xmsgs

Revision: 8
Author: yyea
Date: 16:51:34, 12 March 2014
Message:
Work with initial datapath

Modified : /Assignment2/ALU/Assignment1.gise
Modified : /Assignment2/ALU/Assignment1.xise
Added : /Assignment2/ALU/Datapath1.vhd
Added : /Assignment2/ALU/Datapath2.vhd
Added : /Assignment2/ALU/FF.vhd
Added : /Assignment2/ALU/IR.vhd
Added : /Assignment2/ALU/PC.vhd
Added : /Assignment2/ALU/RF.vhd
Added : /Assignment2/ALU/ROM.vhd
Modified : /Assignment2/ALU/_ngo/netlist.lst
Modified : /Assignment2/ALU/_xmsgs/par.xmsgs
Modified : /Assignment2/ALU/_xmsgs/pn_parser.xmsgs
Modified : /Assignment2/ALU/_xmsgs/xst.xmsgs
Modified : /Assignment2/ALU/iseconfig/Assignment1.projectmgr
Modified : /Assignment2/ALU/iseconfig/main.xreport
Modified : /Assignment2/ALU/main.bgn
Modified : /Assignment2/ALU/main.bit
Modified : /Assignment2/ALU/main.bld
Modified : /Assignment2/ALU/main.cmd_log
Modified : /Assignment2/ALU/main.drc

Modified : /Assignment2/ALU/main.ncd
Modified : /Assignment2/ALU/main.ngc
Modified : /Assignment2/ALU/main.ngd
Modified : /Assignment2/ALU/main.ngr
Modified : /Assignment2/ALU/main.pad
Modified : /Assignment2/ALU/main.par
Modified : /Assignment2/ALU/main.pcf
Modified : /Assignment2/ALU/main.syr
Modified : /Assignment2/ALU/main.twr
Modified : /Assignment2/ALU/main.twx
Modified : /Assignment2/ALU/main.unroutes
Modified : /Assignment2/ALU/main_guide.ncd
Modified : /Assignment2/ALU/main_map.map
Modified : /Assignment2/ALU/main_map.mrp
Modified : /Assignment2/ALU/main_map.ncd
Modified : /Assignment2/ALU/main_map.ngm
Modified : /Assignment2/ALU/main_map.xrpt
Modified : /Assignment2/ALU/main_ngdbuild.xrpt
Modified : /Assignment2/ALU/main_pad.csv
Modified : /Assignment2/ALU/main_pad.txt
Modified : /Assignment2/ALU/main_par.xrpt
Modified : /Assignment2/ALU/main_summary.html
Modified : /Assignment2/ALU/main_summary.xml
Modified : /Assignment2/ALU/main_usage.xml
Modified : /Assignment2/ALU/main_xst.xrpt
Modified : /Assignment2/ALU/usage_statistics_webtalk.html
Modified : /Assignment2/ALU/webtalk.log
Modified : /Assignment2/ALU/webtalk_pn.xml
Modified : /Assignment2/ALU/xst/work/hdllib.ref
Modified : /Assignment2/ALU/xst/work/hdpdeps.ref

Revision: 7
Author: yyea
Date: 02:43:42, 28 February 2014
Message:
final comment and modify.

Modified : /Assignment2/ALU/ALU.vhd
Modified : /Assignment2/ALU/Assignment1.xise
Modified : /Assignment2/ALU/OP.vhd
Modified : /Assignment2/ALU/four_digits_unit.vhd
Modified : /Assignment2/ALU/main.vhd
Modified : /Assignment2/ALU/one_digit.vhd
Modified : /Assignment2/VGA/VGA.xise
Modified : /Assignment2/VGA/disc.vhd
Modified : /Assignment2/VGA/main.vhd

Revision: 6
Author: yyea

Date: 02:41:50, 28 February 2014
Message:
totally finish the functions.

Added : /Assignment2/ALU
Added : /Assignment2/ALU/ALU.vhd
Added : /Assignment2/ALU/Assignment1.gise
Added : /Assignment2/ALU/Assignment1.xise
Added : /Assignment2/ALU/Display.vhd
Added : /Assignment2/ALU/OP.vhd
Added : /Assignment2/ALU/_ngo
Added : /Assignment2/ALU/_ngo/netlist.lst
Added : /Assignment2/ALU/_xmsgs
Added : /Assignment2/ALU/_xmsgs/bitgen.xmsgs
Added : /Assignment2/ALU/_xmsgs/map.xmsgs
Added : /Assignment2/ALU/_xmsgs/ngdbuild.xmsgs
Added : /Assignment2/ALU/_xmsgs/par.xmsgs
Added : /Assignment2/ALU/_xmsgs/pn_parser.xmsgs
Added : /Assignment2/ALU/_xmsgs/trce.xmsgs
Added : /Assignment2/ALU/_xmsgs/xst.xmsgs
Added : /Assignment2/ALU/four_digits_summary.html
Added : /Assignment2/ALU/four_digits_unit.vhd
Added : /Assignment2/ALU/iseconfig
Added : /Assignment2/ALU/iseconfig/Assignment1.projectmgr
Added : /Assignment2/ALU/iseconfig/four_digits.xreport
Added : /Assignment2/ALU/iseconfig/main.xreport
Added : /Assignment2/ALU/main.bgn
Added : /Assignment2/ALU/main.bit
Added : /Assignment2/ALU/main.bld
Added : /Assignment2/ALU/main.cmd_log
Added : /Assignment2/ALU/main.drc
Added : /Assignment2/ALU/main.lso
Added : /Assignment2/ALU/main.ncd
Added : /Assignment2/ALU/main.ngc
Added : /Assignment2/ALU/main.ngd
Added : /Assignment2/ALU/main.ngr
Added : /Assignment2/ALU/main.pad
Added : /Assignment2/ALU/main.par
Added : /Assignment2/ALU/main.pcf
Added : /Assignment2/ALU/main.prj
Added : /Assignment2/ALU/main.ptwx
Added : /Assignment2/ALU/main.stx
Added : /Assignment2/ALU/main.syr
Added : /Assignment2/ALU/main.twr
Added : /Assignment2/ALU/main.twx
Added : /Assignment2/ALU/main.unroutes
Added : /Assignment2/ALU/main.ut
Added : /Assignment2/ALU/main.vhd
Added : /Assignment2/ALU/main.xpi

Added : /Assignment2/ALU/main.xst
Added : /Assignment2/ALU/main_bitgen.xwbt
Added : /Assignment2/ALU/main_envsettings.html
Added : /Assignment2/ALU/main_guide.ncd
Added : /Assignment2/ALU/main_map.map
Added : /Assignment2/ALU/main_map.mrp
Added : /Assignment2/ALU/main_map.ncd
Added : /Assignment2/ALU/main_map.ngm
Added : /Assignment2/ALU/main_map.xrpt
Added : /Assignment2/ALU/main_ngdbuild.xrpt
Added : /Assignment2/ALU/main_pad.csv
Added : /Assignment2/ALU/main_pad.txt
Added : /Assignment2/ALU/main_par.xrpt
Added : /Assignment2/ALU/main_summary.html
Added : /Assignment2/ALU/main_summary.xml
Added : /Assignment2/ALU/main_usage.xml
Added : /Assignment2/ALU/main_vhdl.prj
Added : /Assignment2/ALU/main_xst.xrpt
Added : /Assignment2/ALU/one_digit.vhd
Added : /Assignment2/ALU/usage_statistics_webtalk.html
Added : /Assignment2/ALU/webtalk.log
Added : /Assignment2/ALU/webtalk_pn.xml
Added : /Assignment2/ALU/xlnx_auto_0_xdb
Added : /Assignment2/ALU/xlnx_auto_0_xdb/cst.xbcd
Added : /Assignment2/ALU/xst
Added : /Assignment2/ALU/xst/dump.xst
Added : /Assignment2/ALU/xst/dump.xst/main.prj
Added : /Assignment2/ALU/xst/dump.xst/main.prj/ngx
Added : /Assignment2/ALU/xst/dump.xst/main.prj/ngx/notopt
Added : /Assignment2/ALU/xst/dump.xst/main.prj/ngx/opt
Added : /Assignment2/ALU/xst/file graph
Added : /Assignment2/ALU/xst/projnav.tmp
Added : /Assignment2/ALU/xst/work
Added : /Assignment2/ALU/xst/work/hdllib.ref
Added : /Assignment2/ALU/xst/work/hdpdeps.ref
Added : /Assignment2/ALU/xst/work/sub00
Added : /Assignment2/ALU/xst/work/sub00/vhpl00.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl01.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl02.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl03.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl04.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl05.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl06.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl07.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl08.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl09.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl10.vho
Added : /Assignment2/ALU/xst/work/sub00/vhpl11.vho
Added : /Assignment2/VGA/Triangle.vhd

Added : /Assignment2/VGA/VGA.gise
Added : /Assignment2/VGA/_ngo
Added : /Assignment2/VGA/_ngo/netlist.lst
Added : /Assignment2/VGA/_xmsgs
Added : /Assignment2/VGA/_xmsgs/bitgen.xmsgs
Added : /Assignment2/VGA/_xmsgs/map.xmsgs
Added : /Assignment2/VGA/_xmsgs/ngdbuild.xmsgs
Added : /Assignment2/VGA/_xmsgs/par.xmsgs
Added : /Assignment2/VGA/_xmsgs/pn_parser.xmsgs
Added : /Assignment2/VGA/_xmsgs/trce.xmsgs
Added : /Assignment2/VGA/_xmsgs/xst.xmsgs
Added : /Assignment2/VGA/iseconfig
Added : /Assignment2/VGA/iseconfig/VGA.projectmgr
Added : /Assignment2/VGA/iseconfig/main.xreport
Added : /Assignment2/VGA/main.bgn
Added : /Assignment2/VGA/main.bit
Added : /Assignment2/VGA/main.bld
Added : /Assignment2/VGA/main.cmd_log
Added : /Assignment2/VGA/main.drc
Added : /Assignment2/VGA/main.lso
Added : /Assignment2/VGA/main.ncd
Added : /Assignment2/VGA/main.ngc
Added : /Assignment2/VGA/main.ngd
Added : /Assignment2/VGA/main.ngr
Added : /Assignment2/VGA/main.pad
Added : /Assignment2/VGA/main.par
Added : /Assignment2/VGA/main.pcf
Added : /Assignment2/VGA/main.prj
Added : /Assignment2/VGA/main.ptwx
Added : /Assignment2/VGA/main.stx
Added : /Assignment2/VGA/main.syr
Added : /Assignment2/VGA/main.twr
Added : /Assignment2/VGA/main.twx
Added : /Assignment2/VGA/main.unroutes
Added : /Assignment2/VGA/main.ut
Added : /Assignment2/VGA/main.xpi
Added : /Assignment2/VGA/main.xst
Added : /Assignment2/VGA/main_bitgen.xwbt
Added : /Assignment2/VGA/main_envsettings.html
Added : /Assignment2/VGA/main_guide.ncd
Added : /Assignment2/VGA/main_map.map
Added : /Assignment2/VGA/main_map.mrp
Added : /Assignment2/VGA/main_map.ncd
Added : /Assignment2/VGA/main_map.ngm
Added : /Assignment2/VGA/main_map.xrpt
Added : /Assignment2/VGA/main_ngdbuild.xrpt
Added : /Assignment2/VGA/main_pad.csv
Added : /Assignment2/VGA/main_pad.txt
Added : /Assignment2/VGA/main_par.xrpt

Added : /Assignment2/VGA/main_summary.html
Added : /Assignment2/VGA/main_summary.xml
Added : /Assignment2/VGA/main_usage.xml
Added : /Assignment2/VGA/main_vhdl.prj
Added : /Assignment2/VGA/main_xst.xrpt
Added : /Assignment2/VGA/rectangle.vhd
Added : /Assignment2/VGA/usage_statistics_webtalk.html
Added : /Assignment2/VGA/webtalk.log
Added : /Assignment2/VGA/webtalk_pn.xml
Added : /Assignment2/VGA/xlnx_auto_0_xdb
Added : /Assignment2/VGA/xlnx_auto_0_xdb/cst.xbcd
Added : /Assignment2/VGA/xst
Added : /Assignment2/VGA/xst/dump.xst
Added : /Assignment2/VGA/xst/dump.xst/main.prj
Added : /Assignment2/VGA/xst/dump.xst/main.prj/ngx
Added : /Assignment2/VGA/xst/dump.xst/main.prj/ngx/notopt
Added : /Assignment2/VGA/xst/dump.xst/main.prj/ngx/opt
Added : /Assignment2/VGA/xst/file graph
Added : /Assignment2/VGA/xst/projnav.tmp
Added : /Assignment2/VGA/xst/work
Added : /Assignment2/VGA/xst/work/hdllib.ref
Added : /Assignment2/VGA/xst/work/hdpdeps.ref
Added : /Assignment2/VGA/xst/work/sub00
Added : /Assignment2/VGA/xst/work/sub00/vhpl00.vho
Added : /Assignment2/VGA/xst/work/sub00/vhpl01.vho
Added : /Assignment2/VGA/xst/work/sub00/vhpl02.vho
Added : /Assignment2/VGA/xst/work/sub00/vhpl03.vho
Added : /Assignment2/VGA/xst/work/sub00/vhpl04.vho
Added : /Assignment2/VGA/xst/work/sub00/vhpl05.vho
Added : /Assignment2/VGA/xst/work/sub00/vhpl06.vho
Added : /Assignment2/VGA/xst/work/sub00/vhpl07.vho
Added : /Assignment2/VGA/xst/work/sub00/vhpl08.vho
Added : /Assignment2/VGA/xst/work/sub00/vhpl09.vho
Added : /Assignment2/VGA/xst/work/sub00/vhpl10.vho