Mumtahid Akash

Professor Zheng Peng

CSC34300

17/04/2022

Task1:

Introduction:

The lab deals with the implementation and understanding of Instruction decoder and ALU Control unit. These are the sub module or sub functions of Mips 32. Basic task was to understand the concept of decoder and alu control unit. Below is the vhdl code of Instruction decoder and ALU control unit:

VHDL CODE of Decoder:

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

----------------------------------------------------------------------------------

library IEEE;

use IEEE.STD_LOGIC_1164.ALL;



entity DEC is

port(

I_DEC_EN : in STD_LOGIC;

I_DEC_OPCODE: in STD_LOGIC_VECTOR(5 downto 0);

O_DEC_REGDST : out STD_LOGIC;

O_DEC_JUMP : out STD_LOGIC;
```

```
O_DEC_BEQ : out STD_LOGIC;
O_DEC_BNE : out STD_LOGIC;
O_DEC_MEMREAD : out STD_LOGIC;
O_DEC_MEMTOREG : out STD_LOGIC;
O_DEC_ALUOP : out STD_LOGIC_VECTOR(1 downto 0);
O_DEC_MEMWRITE : out STD_LOGIC;
O_DEC_ALUSRC: out STD_LOGIC;
O_DEC_REGWRITE: out STD_LOGIC
);
end DEC;

architecture Behavioral of DEC is
begin

        O_DEC_REGDST <=         '1' when (I_DEC_OPCODE="000000"  and I_DEC_EN='1') else
                                                                '0' when (I_DEC_OPCODE="001000"  and I_DEC_EN='1') else
                                                                '0' when (I_DEC_OPCODE="000100"  and I_DEC_EN='1') else
                                                                '0' when (I_DEC_OPCODE="000101"  and I_DEC_EN='1') else
                                                                '0' when (I_DEC_OPCODE="000010"  and I_DEC_EN='1') else
                                                                '0' when (I_DEC_OPCODE="100011"  and I_DEC_EN='1') else
                                                                '0' when (I_DEC_OPCODE="101011"  and I_DEC_EN='1') else '0';
```

```
O_DEC_JUMP <=                   '0' when (I_DEC_OPCODE="000000"  and I_DEC_EN='1') else
                                        '0' when (I_DEC_OPCODE="001000"  and I_DEC_EN='1') else
                                        '0' when (I_DEC_OPCODE="000100"  and I_DEC_EN='1') else
                                        '0' when (I_DEC_OPCODE="000101"  and I_DEC_EN='1') else
                                        '1' when (I_DEC_OPCODE="000010"  and I_DEC_EN='1') else
                                        '0' when (I_DEC_OPCODE="100011"  and I_DEC_EN='1') else
                                        '0' when (I_DEC_OPCODE="101011"  and I_DEC_EN='1') else '0';

O_DEC_BEQ <=            '0' when (I_DEC_OPCODE="000000"  and I_DEC_EN='1') else
                                        '1' when (I_DEC_OPCODE="001000"  and I_DEC_EN='1') else
                                        '0' when (I_DEC_OPCODE="000100"  and I_DEC_EN='1') else
                                        '1' when (I_DEC_OPCODE="000101"  and I_DEC_EN='1') else
                                        '0' when (I_DEC_OPCODE="000010"  and I_DEC_EN='1') else
                                        '0' when (I_DEC_OPCODE="100011"  and I_DEC_EN='1') else
                                        '0' when (I_DEC_OPCODE="101011"  and I_DEC_EN='1') else '0';

O_DEC_BNE <=            '0' when (I_DEC_OPCODE="000000"  and I_DEC_EN='1') else
                                        '0' when (I_DEC_OPCODE="001000"  and I_DEC_EN='1') else
                                        '1' when (I_DEC_OPCODE="000100"  and I_DEC_EN='1') else
                                        '1' when (I_DEC_OPCODE="000101"  and I_DEC_EN='1') else
                                        '0' when (I_DEC_OPCODE="000010"  and I_DEC_EN='1') else
                                        '0' when (I_DEC_OPCODE="100011"  and I_DEC_EN='1') else
```

```
                                                    '0' when (I_DEC_OPCODE="101011"  and I_DEC_EN='1') else '0';

O_DEC_MEMREAD <=            '0' when (I_DEC_OPCODE="000000"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="001000"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="000100"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="000101"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="000010"  and I_DEC_EN='1') else
                                                    '1' when (I_DEC_OPCODE="100011"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="101011"  and I_DEC_EN='1') else '0';

O_DEC_MEMTOREG <= '0' when (I_DEC_OPCODE="000000"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="001000"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="000100"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="000101"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="000010"  and I_DEC_EN='1') else
                                                    '1' when (I_DEC_OPCODE="100011"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="101011"  and I_DEC_EN='1') else '0';

O_DEC_MEMWRITE <=  '0' when (I_DEC_OPCODE="000000"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="001000"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="000100"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="000101"  and I_DEC_EN='1') else
```

```
                                                    '0' when (I_DEC_OPCODE="000010"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="100011"  and I_DEC_EN='1') else
                                                    '1' when (I_DEC_OPCODE="101011"  and I_DEC_EN='1') else '0';

O_DEC_ALUSRC <=                 '0' when (I_DEC_OPCODE="000000"  and I_DEC_EN='1') else
                                                    '1' when (I_DEC_OPCODE="001000"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="000100"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="000101"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="000010"  and I_DEC_EN='1') else
                                                    '1' when (I_DEC_OPCODE="100011"  and I_DEC_EN='1') else
                                                    '1' when (I_DEC_OPCODE="101011"  and I_DEC_EN='1') else '0';

O_DEC_REGWRITE <=    '1' when (I_DEC_OPCODE="000000"  and I_DEC_EN='1') else
                                                    '1' when (I_DEC_OPCODE="001000"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="000100"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="000101"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="000010"  and I_DEC_EN='1') else
                                                    '1' when (I_DEC_OPCODE="100011"  and I_DEC_EN='1') else
                                                    '0' when (I_DEC_OPCODE="101011"  and I_DEC_EN='1') else '0';

O_DEC_ALUOP <=           "10" when (I_DEC_OPCODE="000000"  and I_DEC_EN='1')else
                                                    "00" when (I_DEC_OPCODE="001000"  and I_DEC_EN='1')else
```

```
                                        "01" when (I_DEC_OPCODE="000100"  and I_DEC_EN='1')else
                                        "11" when (I_DEC_OPCODE="000101"  and I_DEC_EN='1')else
                                        "00" when (I_DEC_OPCODE="000010"  and I_DEC_EN='1')else
                                        "00" when (I_DEC_OPCODE="100011"  and I_DEC_EN='1') else
                                        "00" when (I_DEC_OPCODE="101011"  and I_DEC_EN='1') else "00";

end Behavioral;
```

VHDL CODE of ALU CONTROL UNIT (ACU):

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

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



entity ACU is
 Port ( I_ACU_ALUOp : in  STD_LOGIC_VECTOR (1 downto 0);
        I_ACU_Funct : in  STD_LOGIC_VECTOR (5 downto 0);
          O_ACUCTL : out  STD_LOGIC_VECTOR (3 downto 0));
```

```
end ACU;

architecture Behavioral of ACU is

begin

        process(I_ACU_ALUOp,I_ACU_Funct)
        begin
                case I_ACU_ALUOp is
                                when "00" => O_ACUCTL <="0010";
                                when "01" => O_ACUCTL          <="0110";
                                when others =>
                                case I_ACU_Funct is
                                                                when "100000" => O_ACUCTL <=
                                                                        "0010";
                                                                when "100010" => O_ACUCTL<=
                                                                        "0110";
                                                                when "100100" => O_ACUCTL <=
                                                                        "0000";
                                                                when "100101" => O_ACUCTL <=
                                                                        "0001";
```

```
                                        when "101010" => O_ACUCTL<=
                                                "0111";
                                        when others => O_ACUCTL <=
                                                "----";
                                end case;
                        end case;
                end process;
end Behavioral;
```

Test Bench:

```
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY TEST_MODULE IS
PORT (

        O_REGDST : OUT  std_logic;
        O_JUMP : OUT  std_logic;
        O_BEQ : OUT  std_logic;
        O_BNE : OUT  std_logic;
        O_MEMREAD : OUT  std_logic;
        O_MEMTOREG : OUT  std_logic;
        O_MEMWRITE : OUT  std_logic;
        O_ALUSRC : OUT  std_logic;
        O_REGWRITE : OUT  std_logic;
```

```
      O_ALUCTL : OUT  std_logic_vector(3 downto 0)

);
END TEST_MODULE;

ARCHITECTURE behavior OF TEST_MODULE IS

   -- Component Declaration for the Unit Under Test (UUT)

   COMPONENT DEC
   PORT(
      I_DEC_EN : IN  std_logic;
      I_DEC_OPCODE : IN  std_logic_vector(5 downto 0);
      O_DEC_REGDST : OUT  std_logic;
      O_DEC_JUMP : OUT  std_logic;
      O_DEC_BEQ : OUT  std_logic;
      O_DEC_BNE : OUT  std_logic;
      O_DEC_MEMREAD : OUT  std_logic;
      O_DEC_MEMTOREG : OUT  std_logic;
      O_DEC_ALUOP : OUT  std_logic_vector(1 downto 0);
      O_DEC_MEMWRITE : OUT  std_logic;
      O_DEC_ALUSRC : OUT  std_logic;
      O_DEC_REGWRITE : OUT  std_logic
     );
   END COMPONENT;


   COMPONENT ACU
 PORT(
                              I_ACU_ALUOp : in  STD_LOGIC_VECTOR (1 downto 0);
      I_ACU_Funct : in  STD_LOGIC_VECTOR (5 downto 0);
      O_ACUCTL : out  STD_LOGIC_VECTOR (3 downto 0)
                              );
   END COMPONENT;
```

```
  --Inputs
  signal I_DEC_EN : std_logic := '0';
          signal    I_EN :   std_logic:= '0';
          signal   I_INSTR :   std_logic_vector(31 downto 0):=(others=>'0');

          --Outputs

  signal O_DEC_ALUOP : std_logic_vector(1 downto 0);

  -- No clocks detected in port list. Replace <clock> below with
  -- appropriate port name


BEGIN

          -- Instantiate the Unit Under Test (UUT)
  uut_DEC: DEC PORT MAP (
        I_DEC_EN => I_EN,
        I_DEC_OPCODE => I_INSTR(31 downto 26),
        O_DEC_REGDST => O_REGDST,
        O_DEC_JUMP => O_JUMP,
        O_DEC_BEQ => O_BEQ,
        O_DEC_BNE => O_BNE,
        O_DEC_MEMREAD => O_MEMREAD,
        O_DEC_MEMTOREG => O_MEMTOREG,
        O_DEC_ALUOP => O_DEC_ALUOP,
        O_DEC_MEMWRITE => O_MEMWRITE,
        O_DEC_ALUSRC => O_ALUSRC,
        O_DEC_REGWRITE => O_REGWRITE
       );


  uut_ACU: ACU PORT MAP (
```

```
      I_ACU_ALUOp => O_DEC_ALUOP,
      I_ACU_Funct => I_INSTR(5 downto 0),
      O_ACUCTL => O_ALUCTL
    );



-- Stimulus process
stim_proc: process
begin
  -- hold reset state for 100 ns.
                    I_INSTR<=X"00123A00";
  wait for 100 ns;
                    I_EN<='1';

  wait for 100 ns;
                    I_INSTR<=X"00123A00";
  wait for 100 ns;
                    I_INSTR<=X"20123A00";
  wait for 100 ns;
                    I_INSTR<=X"10123A00";
  wait for 100 ns;
                    I_INSTR<=X"14123A00";
  wait for 100 ns;
                    I_INSTR<=X"08123A00";
  wait for 100 ns;
                    I_INSTR<=X"8C123A00";
  wait for 100 ns;
                    I_INSTR<=X"AC123A00";
  wait for 100 ns;
                    I_INSTR<=X"20123A00";

  wait for 100 ns;
                    I_INSTR<=X"00123A20";
```

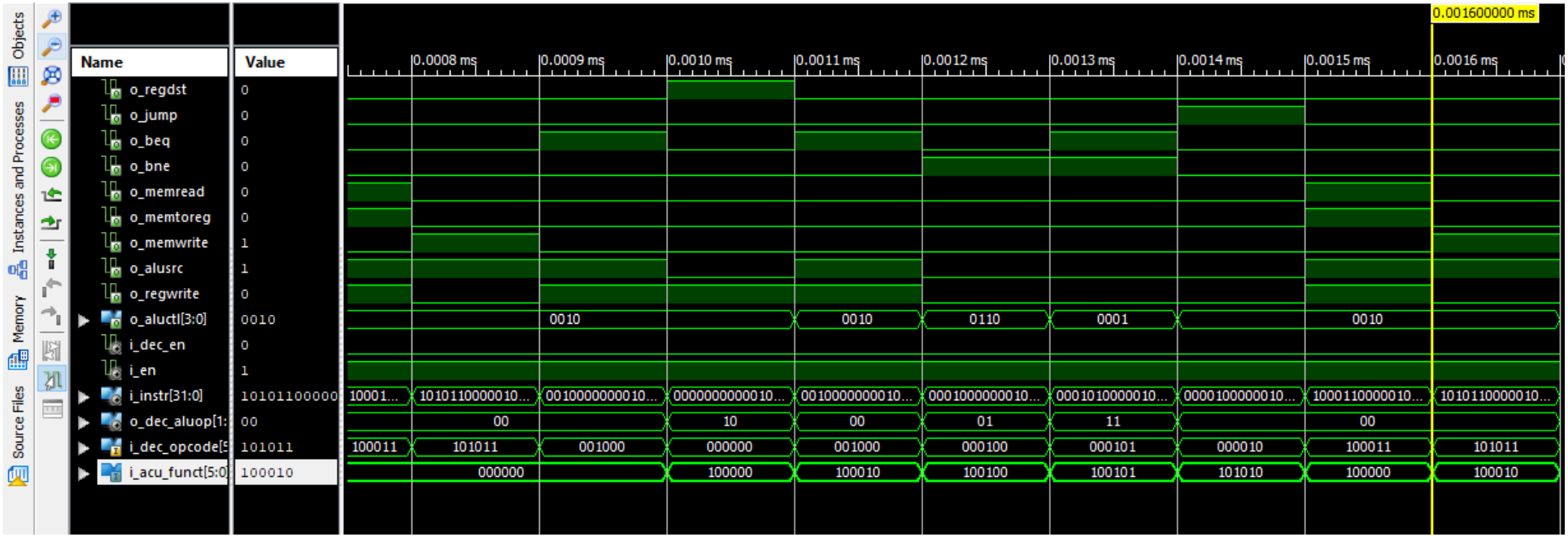
```
    wait for 100 ns;
                I_INSTR<=X"20123A22";
    wait for 100 ns;
                I_INSTR<=X"10123A24";
    wait for 100 ns;
                I_INSTR<=X"14123A25";
    wait for 100 ns;
                I_INSTR<=X"08123A2A";
    wait for 100 ns;
                I_INSTR<=X"8C123A20";
    wait for 100 ns;
                I_INSTR<=X"AC123A22";
    wait for 100 ns;
                I_INSTR<=X"20123A2A";

    -- insert stimulus here

    wait;
  end process;


END;
```

Simulation Waveform:


Objects
Instances and Processes
Memory
Source Files
Name
Value
o_regdst
o_jump
o_beq
o_bne
o_memread
o_memtoreg
o_memwrite
o_alusrc
o_regwrite
o_aluctl[3:0]
i_dec_en
i_en
i_instr[31:0]
o_dec_aluop[1:
i_dec_opcode[5
i_acu_funct[5:0]
0
0
0
0
0
0
1
1
0
0010
0
1
10101100000
00
101011
100010
0.0008 ms
0.0009 ms
0.0010 ms
0.0011 ms
0.0012 ms
0.0013 ms
0.0014 ms
0.0015 ms
0.0016 ms
0.001600000 ms
0010
0010
0110
0001
0010
00
10
00
01
11
00
100011
101011
001000
000000
001000
000100
000101
000010
100011
101011
000000
100000
100010
100100
100101
101010
100000
100010