Final Project EE 5193 (16-bit RISC MIPS Processor)

Problem Statement: Create a 16-bit RISC MIPS Processor. The project will be able to properly execute 10 instructions from memory and store the values in memory, then display them on the seven-segment display on the Nexys board. Below are the layout and instructions given.

You will need four steps to execute an instruction

Fetch (instruction at memory location PC)

Decode

Execute

Update PC

Instructions to be demonstrated: LW 5 201 1001_0101_1100_1001 LW 6 202 1001_0110_1100_1010 ADD 7 5 6 0000_0111_0101_0110

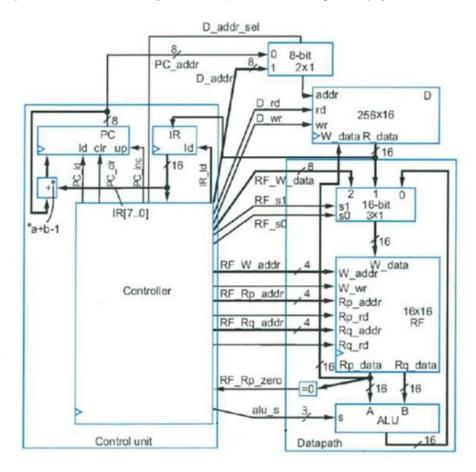
SW 7 203 1010_0111_1100_1011 LI 8 250 1000_1000_1111_1010 SUB 4 8 5 0001_0100_1000_0101

SW 4 204 1010_0100_1100_1100 SRA 3 7 0110_0010_0111_0000 XOR 2 3 4 0100 0010 0011 0100

SW 2 205 1010_0010_1100_1101

Store suitable initial values at memory locations 201 and 202

Output: Show the values in memory locations 203, 204 and 205 in 7-segment display once all instructions are executed.



1. Arithmetic (Two's Complement) ALU operation (2)

Operation	Opcode	Destination	Source	Target	Description
ADD	0000	Rd	Rs	Rt	ADD: Rd = Rs + Rt
					Operands A and B stored in register locations Rs and Rt are added and written to the destination register specified by Rd.
SUB	0001	Rd	Rs	Rt	SUB: Rd = Rs - Rt Operand B (Rt) is subtracted from Operand A (Rs) and written to Rd.

2. Logical ALU operation (6)

Operation	Opcode	Destination	Source	Target	Description
AND 0	0011	Rd	Rs ·	Rt	AND: Rd = Rs & Rt
					Operand A (Rs) is bitwise anded with
					Operand B (Rt) and written into Rd.
OR 0	0011	Rd	Rs	Rt	OR: Rd = Rs Rt
					Operand A (Rs) is bitwise ored with
					Operand B (Rt) and written into Rd.
XOR C	0100	Rd	Rs	Rt	XOR: Rd = Rs ^ Rt
					Operand A (Rs) is bitwise Xored with
					Operand B (Rt) and written into Rd.
NOT	0101	Rd	Rs		NOT: Rd = ~Rs
					Operand A (Rs) is bitwise inverted and
					written into Rd.
SLA 0	0110	Rd	Rs		SLA: Rd = Rs << 1
					Operand A (Rs) is arithmetically shifted
					to the left by one bit and written into
					Rd.
SRA	0111	Rd	Rs		SRA: Rd = Rs >> 1
					Operand A (Rs) is arithmetically shifted
					to the right by one bit and written into
					Rd. The MSB (sign bit) will be
					preserved for this operation.

3. Memory operations (3)

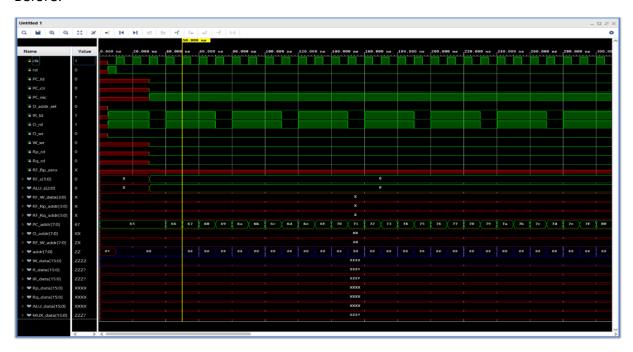
Operation	Opcode	Destination	Source	Target	Description
LÍ	1000	Rd	Imm		LI: Rd = 8-bit Sign extended Immediate The 8-bit immediate in the Instruction word is sign-extended to 16-bits and written into the register specified by Rd.
LW	1001	Rd	Dir		LW: Rd = Mem[Dir] The memory word specified by the address Dir is loaded into register Rd.
SW	1010	Rt	Dir		SW: Mem[Dir] = Rt The data in register Rt is stored into the memory location Dir.

Approach: My approach involved using Assignment 3 (ASM+D) to properly divide up the sections of the project into modules and have them all initialized on a single top module that would just merge all the different pieces.

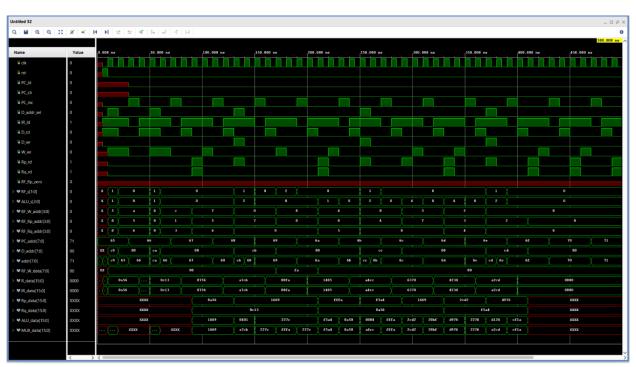
Problems: I encountered several problems during the simulation step. I had disconnected wires (showing Z as an output), failed signals (showing x as an output), and failing modules. I corrected the modules by making sure that all the proper connections were correct in the top module. That helped as well with the high impedance outputs. I also went ahead and checked

the reg signals in each individual module to make sure that they had the right size. For failing signals, I made sure that I was using non-blocking signals through most of the modules to make sure that the signals were being transmitted at the right time. I was able to properly get all the information transmitted throughout the different steps of the instruction cycle. Below are the before and after of the simulation waves.

Before:



After:



Implementation, on the other hand, was not able to work properly. I was able to get past the synthesis step after a small correction but was not able to properly implement it. I got the error below:

```
> • [Place 30-58] IO placement is infeasible. Number of unplaced IO Ports (156) is greater
     than number of available sites (60).
     The following are banks with available pins:
     IO Group: 0 with: SioStd: LVCMOS18 VCCO = 1.8 Termination: 0 TermDir: BiDi Rangeld:
     Drv: 12 has only 60 sites available on device, but needs 156 sites.
     Term: ALU data[0]
     Term: ALU_data[1]
     Term: ALU_data[2]
     Term: ALU_data[3]
     Term: ALU_data[4]
     Term: ALU_data[5]
     Term: ALU_data[6]
     Term: ALU_data[7]
     Term: ALU data[8]
     Term: ALU_data[9]
     Term: ALU_data[10]
     Term: ALU_data[11]
     Term: ALU_data[12]
     Term: ALU_data[13]
     Term: ALU_data[14]
     Term: ALU_data[15]
     Term: IR_data[0]
     Term: IR_data[1]
     Term: IR_data[2]
     Term: IR_data[3]
     Term: IR_data[4]
     Term: IR_data[5]
     Term: IR_data[6]
     Term: IR_data[7]
     Term: IR_data[8]
     Term: IR_data[9]
     Term: IR_data[10]
     Term: IR_data[11]
     Term: IR_data[12]
     Term: IR_data[13]
     Term: IR_data[14]
     Term: IR_data[15]
     Term: MUX_data[0]
     Term: MUX_data[1]
     Term: MUX_data[2]
     Term: MUX data[3]
     Term: MUX_data[4]
     Term: MUX_data[5]
     Term: MUX_data[6]
     Term: MUX_data[7]
     Term: MUX_data[8]
     Term: MUX_data[9]
     Term: MUX_data[10]
     Term: MUX_data[11]
     Term: MUX_data[12]
     Term: MUX_data[13]
     Term: MUX_data[14]
     Term: MUX_data[15]
```

I believe the error is not resolvable in the time left to submit the project. Due to this, I will be submitting the assignment as is. If there was more time to go ahead and work on this project, then I would redesign the project to make sure that it could properly work in the Nexys board.

Top Module:

```
RISC MIPS processor.v
                                                                                    _ D 7 X
C:/Users/oscao/Final_Project/Final_Project.srcs/sources_1/new/RISC_MIPS_processor.v
                                                                                         ×
Q 🕍 ♠ → ¾ 🗎 🗈 🗙 // 🎟 🏗 ♀
                                                                                         ø
 1 'timescale 1ns / 1ps
3 1 // Company: EE 5193 FPGA & HDL
   // Engineer: Oscar Oscrnic
 5
    // Create Date: 08/08/2022 07:44:18 PM
    // Design Name: Top Module
    // Module Name: RISC_MIPS_processor
    // Project Name: Final Project
    // Target Devices:
    // Tool Versions:
12
  // Description:
13
  // Dependencies:
14
15 : //
16 : // Revision:
17 : // Revision 0.01 - File Created
18 : // Additional Comments:
19 1
21
22
23 module RISC_MIPS_processor(
24
       input clk i, rst,
25
       inout PC_ld, PC_clr, PC_inc, D_addr_sel, IR_ld, D_rd, D_wr, W_wr, Rp_rd, Rq_rd, RF_Rp_zero,
26
       inout [1:0] RF s,
27
       inout [2:0] ALU_s,
28
       inout [3:0] RF_W_addr, RF_Rp_addr, RF_Rq_addr,
29
       inout [7:0] PC_addr, D_addr, addr, RF_W_data,
       inout [15:0] R data, IR data, Rp data, Rq data, ALU data, MUX data,
30
31
       output [7:0] disp_seg_o, disp_an_o
32 !
       );
33
34 🖯
       controller UUT1(clk_i, rst, RF_Rp_zero, IR_data, PC_ld, PC_clr, PC_inc, D_addr_sel,
                     IR ld, D rd, D wr, W wr, Rp rd, Rq rd, D addr, RF W data, RF s,
36 🖨
                     RF W addr, RF Rp addr, RF Rq addr, ALU s);
37 ⊖
       regfile UUT2 (W wr, Rp rd, Rq rd, RF W addr, RF Rp addr, RF Rq addr, MUX data,
38 ⊖
                   RF_Rp_zero, Rp_data, Rq_data);
39 1
       memory_unit UUT3(D_rd, D_wr, addr, Rp_data, R_data);
      ALU UUT4 (ALU_s, Rp_data, Rq_data, ALU_data);
40
41
       PC_unit UUT5(PC_ld, PC_clr, PC_inc, IR_data, PC_addr);
       IR_unit UUT6(IR_ld, R_data, IR_data);
42
43
       memory_mux UUT7(D_addr_sel, PC_addr, D_addr, addr);
        data mux UUT8 (RF s, RF W data, R data, ALU data, MUX data);
44
45
      swnumber UUT9(D_addr, Rp_data, disp_seg_o, disp_an_o);
46
47 @ endmodule
     <
```

Controller:

```
controller.v
                                                                                  _ D 7 X
C:/Users/oscao/Final_Project/Final_Project.srcs/sources_1/new/controller.v
Q 🕍 🖴 🔏 📵 🗈 🗙 // 🎟 🏗 Q
                                                                                        o
  1 'timescale 1ns / 1ps
 3 1 // Company: EE 5193 FPGA & HDL
    // Engineer: Oscar Oscrnic
 5 : //
 6 | // Create Date: 08/06/2022 08:13:35 PM
 7 // Design Name: Control Unit
 8 | // Module Name: controller
 9 // Project Name: Final Project
 10
    // Target Devices:
    // Tool Versions:
 11 :
 12
     // Description:
 13
    // Dependencies:
 14
 15
 16 // Revision:
17 : // Revision 0.01 - File Created
 18 : // Additional Comments:
 19 1/
 21
22
23 module controller(
        input clk_i, rst, RF_Rp_zero,
24 :
 25
        input [15:0] IR_data,
       output PC_ld, PC_clr, PC_inc, D_addr_sel, IR_ld, D_rd, D_wr, W_wr, Rp_rd, Rq_rd, output [7:0] D_addr, RF_W_data, output [1:0] RF_s,
 26
 27
 28
       output [3:0] RF_W_addr, RF_Rp_addr, RF_Rq_addr,
 29
       output [2:0] alu_s
 30
 31
        );
 32
 33
       reg PC ld, PC clr, PC inc, D addr sel, IR ld, D rd, D wr, W wr, Rp rd, Rq rd;
 34
       reg [7:0] D_addr, RF_W_data;
 35
       reg [1:0] RF s;
 36
        reg [3:0] RF W addr, RF Rp addr, RF Rq addr;
 37
        reg [2:0] alu_s;
 38
 39
        reg [1:0] state, nstate;
 40
 41 ⊖
        always @(posedge rst or posedge clk_i) begin
 42 🖯
           if (rst) begin
               state <= 2'b00;
 43
 44
               nstate <= 2'b00;
45
               end
46
            else
47 ⊝
                state <= nstate;
48 🖨
49
50 ⊖
        always @(state) begin
51 🖯
           case (state)
 52
                2'b00: nstate <= 2'b01;
 53
               2'b01: nstate <= 2'b11;
            2'b11: nstate <= 2'b10:
 54
```

```
controller.v
                                                                                             _ D 7 X
C:/Users/oscao/Final_Project/Final_Project.srcs/sources_1/new/controller.v
Q 🕍 🖴 λ 📵 🛍 🗶 // 🖩 🖼 Ω
                                                                                                   o
 50 €
         always @(state) begin
 51 🖯
             case (state)
 52
                 2'b00: nstate <= 2'b01;
 53
                 2'b01: nstate <= 2'b11;
 54
                 2'b11: nstate <= 2'b10;
                 2'b10: nstate <= 2'b00;
 55
 56 €
              endcase
 57 🖨
         end
 58
 59 ⊖
         always @(state) begin
 60 ⊖
             case (state)
 61 🖯
                 2'b00: begin //Fetch
                     D_addr_sel = 0; IR_ld = 1; D_rd = 1; D_wr = 0; PC_inc = 0; end
 62 A
 63 🖯
                 2'b01: begin //Decode
 64 🖯
                     case (IR data[15:12])
 65 □
                         4'b00000: begin //ADD
                             D_addr_sel = 0; D_rd = 0; D_wr = 0; RF_s = 2'b00; W_wr = 0;
 66
                             Rp_rd = 1; Rq_rd = 1; alu_s = IR_data[14:12];
 67
                             RF_W_addr = IR_data[11:8]; RF_Rp_addr = IR_data[7:4];
 68
                             RF_Rq_addr = IR_data[3:0]; D_addr = 8'h00; RF_W_data = 8'h00;
 69
 70 0
                             end
 71 🖯
                         4'b00001: begin //SUB
 72 :
                             D addr sel = 0; D rd = 0; D wr = 0; RF s = 2'b00; W wr = 0;
                             Rp rd = 1; Rq rd = 1; alu s = IR data[14:12];
 73
 74
                             RF W addr = IR data[11:8]; RF Rp addr = IR data[7:4];
 75
                             RF Rq addr = IR data[3:0]; D addr = 8'h00; RF W data = 8'h00;
76 0
77 🖯
                         4'b0010: begin //AND
 78
                             D_addr_sel = 0; D_rd = 0; D_wr = 0; RF_s = 2'b00; W_wr = 0;
 79
                             Rp_rd = 1; Rq_rd = 1; alu_s = IR_data[14:12];
 RO
                             RF_W_addr = IR_data[11:8]; RF_Rp_addr = IR_data[7:4];
 81
                             RF_Rq_addr = IR_data[3:0]; D_addr = 8'h00; RF_W_data = 8'h00;
 82 A
                             end
 83 ⊖
                         4'b0011: begin //OR
 84
                             D addr sel = 0; D rd = 0; D wr = 0; RF s = 2'b00; W wr = 0;
 85
                             Rp rd = 1; Rq rd = 1; alu s = IR data[14:12];
 86
                             RF W addr = IR data[11:8]; RF Rp addr = IR data[7:4];
 87
                             RF_Rq_addr = IR_data[3:0]; D_addr = 8'h00; RF_W_data = 8'h00;
 88 🖨
 89 □
                         4'b0100: begin //XOR
                             D addr sel = 0; D rd = 0; D wr = 0; RF s = 2'b00; W wr = 0;
 90
 91
                             Rp_rd = 1; Rq_rd = 1; alu_s = IR_data[14:12];
                             RF_W_addr = IR_data[11:8]; RF_Rp_addr = IR_data[7:4];
 92
 93
                             RF_Rq_addr = IR_data[3:0]; D_addr = 8'h00; RF_W_data = 8'h00;
 94 A
                             end
 95 ₪
                         4'b0101: begin //NOT
 96
                             D addr sel = 0; D rd = 0; D wr = 0; RF s = 2'b00; W wr = 0;
 97
                             Rp rd = 1; Rq rd = 0; alu s = IR data[14:12];
                             RF_W_addr = IR_data[11:8]; RF_Rp_addr = IR_data[7:4];
 98
 99 白
                             RF Rq addr = 4'h00; D addr = 8'h00; RF W data = 8'h00; end
                          4'b0110: begin //SLA
101
                             D_addr_sel = 0; D_rd = 0; D_wr = 0; RF_s = 2'b00; W_wr = 0;
102
                             Ro rd = 1: Ro rd = 0: alu s = IR data[14:12]:
```

```
controller.v
                                                                                         _ D 7 X
C:/Users/oscao/Final_Project/Final_Project.srcs/sources_1/new/controller.v
Q 🕍 ← → % 📵 🛍 🗙 // 🎟 🏗 Q
                                                                                               o
                         4'b0110: begin //SLA
101
                            D addr sel = 0; D rd = 0; D wr = 0; RF s = 2'b00; W wr = 0;
102
                            Rp_rd = 1; Rq_rd = 0; alu_s = IR_data[14:12];
103
                            RF_W_addr = IR_data[11:8]; RF_Rp_addr = IR_data[7:4];
104 @
                            RF_Rq_addr = 4'h00; D_addr = 8'h00; RF_W_data = 8'h00; end
105 🖨
                         4'b0111: begin //SRA
106
                            D_addr_sel = 0; D_rd = 0; D_wr = 0; RF_s = 2'b00; W_wr = 0;
107
                            Rp_rd = 1; Rq_rd = 0; alu_s = IR_data[14:12];
                            RF W_addr = IR_data[11:8]; RF_Rp_addr = IR_data[7:4];
108
109 🖨
                            RF Rg addr = 4'h00; D addr = 8'h00; RF W data = 8'h00; end
                        4'b1000: begin //LI
                            D addr sel = 0; D rd = 0; D wr = 0; RF s = 2'b10; W wr = 1;
111
112
                            Rp rd = 0; Rq rd = 0; alu s = IR data[14:12];
113
                            RF_W_addr = IR_data[11:8]; RF_W_data = IR_data[7:0];
114 白
                            RF_Rp_addr = 4'h00; RF_Rq_addr = 4'h00; D_addr = 8'h00; end
115 🖨
                         4'b1001: begin //LW
116
                            D addr sel = 1; D rd = 1; D wr = 0; RF s = 2'b01; W wr = 1;
117
                            Rp rd = 0; Rq rd = 0; alu s = IR data[14:12];
118
                            RF_W_addr = IR_data[11:8]; D_addr = IR_data[7:0];
119 A
                            RF_Rp_addr = 4'h00; RF_Rq_addr = 4'h00; RF_W_data = 8'h00; end
120 🖨
                         4'b1010: begin //SW
121
                            D_addr_sel = 1; D_rd = 0; D_wr = 1; RF_s = 2'b01; W_wr = 0;
122
                            Rp_rd = 1; Rq_rd = 0; alu_s = IR_data[14:12];
123
                            RF Rp addr = IR data[11:8]; D addr = IR data[7:0];
                            RF W data = 8'h00; RF Rq addr = 4'h00; RF W addr = 4'h00; end
124 A
125 ⊝
                     endcase
126 🖨
                 end
127 ⊖
                 2'b11: begin //Execute
128 ⊕
                     case (IR_data[15:12])
129 🖨
                         4'b00000: begin //ADD
130
                            D_addr_sel = 0; D_rd = 0; D_wr = 0; RF_s = 2'b00; W_wr = 1;
131
                            Rp_rd = 0; Rq_rd = 0; alu_s = IR_data[14:12];
132
                            RF_W_addr = IR_data[11:8]; RF_Rp_addr = IR_data[7:4];
                            133
134 🖯
                            end
135 ⊖
                         4'b0001: begin //SUB
136
                            D addr sel = 0; D rd = 0; D wr = 0; RF s = 2'b00; W wr = 1;
137
                            Rp_rd = 0; Rq_rd = 0; alu_s = IR_data[14:12];
138
                            RF_W_addr = IR_data[11:8]; RF_Rp_addr = IR_data[7:4];
139
                            RF_Rq_addr = IR_data[3:0]; D_addr = 8'h00; RF_W_data = 8'h00;
140 A
                            end
141 0
                         4'b0010: begin //AND
142
                            D_addr_sel = 0; D_rd = 0; D_wr = 0; RF_s = 2'b00; W_wr = 1;
143
                            Rp_rd = 0; Rq_rd = 0; alu_s = IR_data[14:12];
                            RF_W_addr = IR_data[11:8]; RF_Rp_addr = IR_data[7:4];
144
                            145
146 ⊕
                            end
147 ⊖
                         4'b0011: begin //OR
148
                            D addr sel = 0; D rd = 0; D wr = 0; RF s = 2'b00; W wr = 1;
                            Rp rd = 0; Rq rd = 0; alu s = IR data[14:12];
149
150
                            RF W addr = IR data[11:8]; RF Rp addr = IR data[7:4];
151
                            RF Rq addr = IR data[3:0]; D addr = 8'h00; RF W data = 8'h00;
152 ℮
153 ⊝
                         4'b0100: begin //XOR
```

```
controller.v
                                                                                             _ D 7 X
C:/Users/oscao/Final_Project/Final_Project.srcs/sources_1/new/controller.v
Q 📓 🖴 🥕 🐰 📵 🛍 🗶 // 🎟 🏗 Q
                                                                                                   Ò
                              RF Rq addr = IR data[3:0]; D addr = 8'h00; RF W data = 8'h00;
146 (
                              end
147 ⊖
                          4'b0011: begin //OR
148
                             D_addr_sel = 0; D_rd = 0; D_wr = 0; RF_s = 2'b00; W_wr = 1;
149
                              Rp_rd = 0; Rq_rd = 0; alu_s = IR_data[14:12];
150
                             RF_W_addr = IR_data[11:8]; RF_Rp_addr = IR_data[7:4];
151
                             RF Rq addr = IR data[3:0]; D addr = 8'h00; RF W data = 8'h00;
152 A
                             end
153 G
                         4'b0100: begin //XOR
154
                             D_addr_sel = 0; D_rd = 0; D_wr = 0; RF_s = 2'b00; W_wr = 1;
155
                             Rp_rd = 0; Rq_rd = 0; alu_s = IR_data[14:12];
                             RF_W_addr = IR_data[11:8]; RF_Rp_addr = IR_data[7:4];
156
157
                             RF_Rq_addr = IR_data[3:0]; D_addr = 8'h00; RF_W_data = 8'h00;
158 🖯
                             end
159 日
                         4'b0101: begin //NOT
160
                             D_addr_sel = 0; D_rd = 0; D_wr = 0; RF_s = 2'b00; W_wr = 1;
161
                             Rp rd = 0; Rq rd = 0; alu s = IR data[14:12];
162
                             RF W addr = IR data[11:8]; RF Rp addr = IR data[7:4];
163 🖨
                             RF_Rq_addr = 4'h0; D_addr = 8'h00; RF_W_data = 8'h00; end
164 ⊖
                         4'b0110: begin //SLA
165
                             D_addr_sel = 0; D_rd = 0; D_wr = 0; RF_s = 2'b00; W_wr = 1;
166
                              Rp_rd = 0; Rq_rd = 0; alu_s = IR_data[14:12];
167
                              RF_W_addr = IR_data[11:8]; RF_Rp_addr = IR_data[7:4];
                             RF_Rq_addr = 4'h0; D_addr = 8'h00; RF_W_data = 8'h00; end
168 A
169 0
                         4'b0111: begin //SRA
170
                             D_addr_sel = 0; D_rd = 0; D_wr = 0; RF_s = 2'b00; W_wr = 1;
171
                             Rp_rd = 0; Rq_rd = 0; alu_s = IR_data[14:12];
172
                             RF_W_addr = IR_data[11:8]; RF_Rp_addr = IR_data[7:4];
173 ⊕
                             RF_Rq_addr = 4'h0; D_addr = 8'h00; RF_W_data = 8'h00; end
174 🖨
                         4'b1000: begin //LI
175
                             D addr sel = 0; D rd = 0; D wr = 0; RF s = 2'b10; W wr = 0;
176
                              Rp rd = 0; Rq rd = 0; alu s = IR data[14:12];
177
                              RF_W_addr = IR_data[11:8]; RF_W_data = IR_data[7:0];
178
                             RF_Rp_addr = 4'h0; RF_Rq_addr = 4'h0; D_addr = 8'h00; end
179 ⊖
                          4'b1001: begin //LW
180
                             D addr sel = 0; D rd = 0; D wr = 0; RF s = 2'b01; W wr = 0;
181
                             Rp rd = 0; Rq rd = 0; alu s = IR data[14:12];
182
                             RF W addr = IR data[11:8]; D addr = IR data[7:0];
                             RF_Rp_addr = 4'h0; RF_Rq_addr = 4'h0; RF_W_data = 8'h00; end
183 A
184 ⊖
                          4'b1010: begin //SW
185
                             D_addr_sel = 0; D_rd = 0; D_wr = 0; RF_s = 2'b01; W_wr = 0;
186
                             Rp_rd = 0; Rq_rd = 0; alu_s = IR_data[14:12];
187
                             RF_Rp_addr = IR_data[11:8]; D_addr = IR_data[7:0];
188 🖯
                             RF W data = 8'h00; RF Rq addr = 4'h0; RF W addr = 4'h0; end
189白
                     endcase
190 €
                 end
191 ⊖
                 2'b10: begin //Update PC
192
                      PC_ld = 0; PC_clr = 0; PC_inc = 1; D_addr_sel = 0; IR_ld = 0;
193
                     D_rd = 0; D_wr = 0; RF_s = 0; W_wr = 0; Rp_rd = 0; Rq_rd = 0;
194 🖯
                     alu s = 0; end
195 白
              endcase
196 🖨
         end
197 @ endmodule
198 :
```

Regfile:

```
regfile.v
                                                                            ? _ D 7 X
C:/Users/oscao/Final_Project/Final_Project.srcs/sources_1/new/regfile.v
Q 🕍 ← 🥕 % 📵 🗈 🗙 // 🎟 🏗 ♀
                                                                                    ٠
 1 'timescale 1ns / 1ps
3 // Company: EE 5193 FPGA & HDL
  // Engineer: Oscar Oscrnic
5 : //
6 1 // Create Date: 08/07/2022 03:03:52 PM
7
   // Design Name: Register File Module
8
  // Module Name: regfile
   // Project Name: Final Project
   // Target Devices:
11
    // Tool Versions:
    // Description:
12
13
  // Dependencies:
14
15
16 // Revision:
17 : // Revision 0.01 - File Created
18 : // Additional Comments:
19 1/
21
22
23 module regfile(
24
       input W_wr, Rp_rd, Rq_rd,
25
       input [3:0] W_addr, Rp_addr, Rq_addr,
      input [15:0] mux,
26
27
       output RF_Rp_zero,
28
       output [15:0] Rp_data, Rq_data
29
       );
30
31
      reg RF_Rp_zero;
32
       reg [15:0] RF [15:0];
33
       reg [15:0] Rp data, Rq data;
34
35 ♥
      always @(Rp_rd or Rp_addr) begin
36 ⊜
         if (Rp_rd)
37 ℮
             Rp_data <= RF[ Rp_addr ];
38 ⊖
       end
39 ;
40 🖨
       always @(Rq_rd or Rq_addr) begin
41 🕀
         if (Rq_rd)
42 🖨
             Rq_data <= RF[ Rq_addr ];
43 🖨
       end
44 1
45 🖨
       always @(W_wr or W_addr or mux) begin
46 🖨
         if (W_wr)
47 ⊖
              RF[ W addr ] <= mux;
48 🖨
49
50 ₽
       always @(Rp_data) begin
51 🖯
          if (Rp_data == 0)
52 🖨
              RF_Rp_zero <= 1;
53 ⊖
       end
54 A endmodule
```

Memory:

```
memory_unit.v
                                                                                   _ D # X
C:/Users/oscao/Final_Project/Final_Project.srcs/sources_1/new/memory_unit.v
Q III ← → X III III F Q
                                                                                        ø
3 1 // Company: EE 5193 FPGA & HDL
4
   // Engineer: Osoar Osornio
   // Create Date: 08/07/2022 12:53:30 AM
7
    // Design Name: 256 bit Memory
8
    // Module Name: memory_unit
 9
    // Project Name: Final Project
    // Target Devices:
11
    // Tool Versions:
12
   // Description:
13 : //
14 | // Revision:
15 // Revision 0.01 - File Created
17
18 - module memory_unit(
19
       input D rd, D wr,
20 🖯
       input [7:0] addr,
       input [15:0] W_data,
21
22
       output [15:0] R_data
23
       );
24
25
      reg [15:0] D [255:0];
26 ;
      reg [15:0] R data;
27
       integer i;
28
29 □
       initial begin
       for (i = 0; i < 256; i = i + 1)

D[i] <= 16'h0000;
30 ⊖
31 🖨
32
33 1
          D[201] <= 16'h0A56; //decimal = 2646
          D[202] <= 16'h0Cl3; //decimal = 3091
34
35 !
          D[101] <= 16'h95C9; //LW 5 201
36
          D[102] <= 16'h96CA; //LW 6 202
37
          D[103] <= 16'h0756; //ADD 7 5 6
          D[104] <= 16'hA7CB; //SW 7 203
38
          D[105] <= 16'h88FA; //LI 8 250
D[106] <= 16'h1485; //SUB 4 8 5
39
40
          D[107] <= 16'hA4cc; //SW 4 204
41
          D[108] <= 16'h6370; //SRA 3 7
42
          D[109] <= 16'h4234; //XOR 2 3 4
43
          D[110] <= 16'hA2CD; //SW 2 205
44
45 🖨
46
47 ₽
       always @(D_rd or D_wr or addr or W_data) begin
48 🖨
        if (D_wr)
49 0
              D[ addr ] <= W data;
50 ♥
          if (D_rd)
51 🖨
              R_data <= D[ addr ];
52 🖨
        end
53 !
54 A endmodule
```

ALU:

```
ALU.v
                                                                             _ D 7 X
                                                                                  ×
C:/Users/oscao/Final_Project/Final_Project.srcs/sources_1/new/ALU.v
Q III ← → X III III 7 Q
                                                                                  ٥
 1 'timescale 1ns / 1ps
3 : // Company: EE 5193 FPGA & HDL
4
   // Engineer: Oscar Oscrnio
5
    // Create Date: 08/07/2022 03:28:25 PM
    // Design Name: ALU Module
   // Module Name: ALU
9
   // Project Name: Final Project
  // Target Devices:
10
   // Tool Versions:
11
  // Description:
12
13
  // Dependencies:
14
15 : //
16 : // Revision:
17 | // Revision 0.01 - File Created
18 : // Additional Comments:
19
21
22
23 - module ALU(
     input [2:0] s,
24
25
       input [15:0] Rp_data, Rq_data,
26
       output [15:0] ALU data
27
28
29
       reg [15:0] ALU_data;
30
31 🖯
       always @(*) begin
32 ⊖
         case (s)
33 1
             3'b000: ALU_data = Rp_data + Rq_data;
34
            3'b001: ALU_data = Rp_data - Rq_data;
35 !
            3'b010: ALU data = Rp data & Rq data;
36
            3'b011: ALU data = Rp data | Rq data;
37 1
            3'b100: ALU_data = Rp_data ^ Rq_data;
38 ;
            3'b101: ALU_data = ~Rp_data;
39 ;
            3'b110: ALU_data = Rp_data << 1;
40
             3'b111: ALU_data = Rp_data >> 1;
41 0
          endcase
42 🖨
       end
43
44 @ endmodule
45
```

Program Counter:

```
PC unit.v
                                                                           _ D 7 X
C:/Users/oscao/Final_Project/Final_Project.srcs/sources_1/new/PC_unit.v
Q 📓 ← 🥕 % 🖥 🛍 🗙 // 🖩 🙃 Ω
                                                                                ٠
1 'timescale 1ns / 1ps
3 1 // Company: EE 5193 FPGA & HDL
   // Engineer: Oscar Oscrnic
5
   // Create Date: 08/07/2022 03:50:29 PM
   // Design Name: Program Counter Module
   // Module Name: PC unit
  // Project Name: Final Project
  // Target Devices:
10
   // Tool Versions:
11
  // Description:
12
13
14 // Dependencies:
15 1/
16 : // Revision:
17 | // Revision 0.01 - File Created
18 : // Additional Comments:
19 //
21
22
23 module PC_unit(
24 input PC_ld, PC_clr, PC_inc,
      input [15:0] IR,
25
26
      output [7:0] PC_addr
27 1
      );
28
29 ;
      reg [7:0] PC_addr, offset;
30
      initial begin
31 🖯
         PC_addr = 101;
32 !
33 🖨
34
35 ₪
      always @ (PC_ld or PC_clr or PC_inc or IR) begin
36 0 //
       if (PC clr)
37 : //
              PC addr <= 0;
38 //
          if (PC_ld) begin
39 : //
              offset <= IR[7:0];
40 0 //
              PC = PC + offset - 1; end
41 ⊖
        if (PC_inc)
42 🖨
            PC_addr = PC_addr + 1;
43 🖨
      end
44 !
45 🖨 endmodule
```

Instruction Register Unit:

```
IR unit.v
                                                                         _ D 7 X
C:/Users/oscao/Final_Project/Final_Project.srcs/sources_1/new/IR_unit.v
Q 📓 ← → 🐰 🖺 🛍 🗙 // 🎟 🙃 Q
                                                                              ۰
   'timescale 1ns / 1ps
3 1 // Company: EE 5193 FPGA & HDL
   // Engineer: Oscar Oscrnic
5
  // Create Date: 08/07/2022 12:41:29 AM
   // Design Name: Instruction Register
8
  // Module Name: IR_unit
  // Project Name: Final Project
10 : // Target Devices:
11
  : // Tool Versions:
12 : // Description:
13 : //
14 // Dependencies:
15 : //
16 // Revision:
17 | // Revision 0.01 - File Created
18 // Additional Comments:
19 : //
21
22
23 module IR_unit(
24
      input IR_ld,
25
      input [15:0] R_data,
      output [15:0] IR_data
26
27
28
29
      reg [15:0] IR_data;
30 :
31 🖯
      always @(IR_ld or R_data) begin
32 ♥
         if (IR ld)
33 🖨
            IR_data <= R_data;
34 🖨
35 !
36 🖨 endmodule
37
```

Memory MUX:

```
memory_mux.v
                                                                          _ D 7 X
C:/Users/oscao/Final_Project/Final_Project.srcs/sources_1/new/memory_mux.v
Q 📓 🦘 🔏 🖺 🛍 🗶 // 🎟 🏗 🔉
                                                                               ٠
 1 'timescale 1ns / 1ps
// Company: EE 5193 FPGA & HDL
   // Engineer: Oscar Oscrnic
 5
   // Create Date: 08/07/2022 12:31:09 AM
   // Design Name: 8-bit address MUX
   // Module Name: memory mux
   // Project Name: Final Project
  // Target Devices:
10
   // Tool Versions:
11
  // Description:
12
13
14 : // Dependencies:
15 1/
16 : // Revision:
17 // Revision 0.01 - File Created
18 : // Additional Comments:
19 1/
21
22
23 module memory_mux(
24
     input D_addr_sel,
      input [7:0] PC_addr, D_addr,
25
26
      output [7:0] addr
27
      );
28
29 !
      reg [7:0] addr;
30
31 🖨
      always @(D addr sel or PC addr or D addr) begin
32
33 ⊖
         case (D_addr_sel)
34
             1'b0: addr <= PC_addr;
35
             1'bl: addr <= D_addr;
             default: addr <= 8'hFF;
36
37 🖨
         endcase
38 ⊖
       end
39 :
40 endmodule
```

Data MUX:

```
data mux.v
                                                                            _ D 7 X
C:/Users/oscao/Final_Project/Final_Project.srcs/sources_1/new/data_mux.v
Q 🕍 🖴 🧦 🖺 🛍 🗶 // 🎟 🎏 Q
                                                                                  ۰
 // Company: EE 5193 FPGA & HDL
   // Engineer: Oscar Osornio
5
   // Create Date: 08/07/2022 01:31:16 AM
   // Design Name: 16 bit datapath MUX
   // Module Name: data mux
-8
  // Project Name: Final Project
9
10 : // Target Devices:
11 // Tool Versions:
12 | // Description:
13 : //
14 : // Dependencies:
15 : //
16 1 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 : //
21
22
23 module data_mux(
24
      input [1:0] RF_s,
      input [7:0] RF_W_data,
25
      input [15:0] R_data, ALU_data,
26
27
      output [15:0] mux_data
28
29
30 ;
      reg [15:0] mux_data;
31
32 🖨
      always @(RF_s or RF_W_data or R_data or ALU_data) begin
33 ♥
         case (RF s)
34 :
            2'b00: mux_data <= ALU_data;
35
            2'b01: mux_data <= R_data;
36 ;
            2'b10: mux_data <= {{8{ RF_W_data[7] }}, RF_W_data[7:0]};
37 :
             default: mux_data <= ALU_data;
38 ♠
          endcase
39 ⊖
      end
40 :
41 endmodule
42 ;
```

Seven Segment Display:

```
swnumber.v
                                                                                 _ D # X
C:/Users/oscao/Final_Project/Final_Project.srcs/sources_1/new/swnumber.v
Q 🕍 4 / X 🗐 🛍 🗙 // 🎟 🏗 🔉
                                                                                      o
 1 'timescale 1ns / 1ps
 3 // Company: EE 5193 FPGA & HDL
 4 // Engineer: Oscar Oscrnic
 5 !
    // Create Date: 08/10/2022 12:51:08 PM
     // Design Name: Seven Segment Display
    // Module Name: swnumber
   : // Project Name: Final Project
 10 // Target Devices:
 11 ! // Tool Versions:
 12 : // Description:
 13 : //
 14
    // Dependencies:
 15
    // Revision:
 16
    // Revision 0.01 - File Created
 17
 18 / // Additional Comments:
 19
 21
 22
 23 module swnumber (
24
       input [7:0] D addr,
       input [15:0] Rp data,
 26 ;
       output [7:0] disp seg o,
 27
       output [7:0] disp_an_o
 28
       );
 29
 30
       reg [7:0] seg;
       reg [7:0] seg0, seg1, seg2, seg3;
 31
 32
       reg [7:0] loc;
 33
 34 🖯
        always @(Rp_data[3:0]) begin
 35 ⊖
          case (Rp_data[3:0])
 36
               4'b0000: seg0 = 8'b1100_0000;
 37
               4'b0001: seg0 = 8'b1111_1001;
 38
               4'b0010: seg0 = 8'b1010_0100;
               4'b0011: seg0 = 8'b1011_0000;
 39
 40
               4'b0100: seg0 = 8'b1001_1001;
 41
 42
              4'b0101: seg0 = 8'b1001_0010;
               4'b0110: seg0 = 8'b1000_0010;
 43
               4'b0111: seg0 = 8'b1111 1000;
 44
 45
 46
              4'b1000: seg0 = 8'b1000_0000;
               4'b1001: seg0 = 8'b1001_0000;
 47
 48
               4'b1010: seg0 = 8'b1111 0111;
 49
               4'b1011: seg0 = 8'b1111_1100;
 50
 51
               4'b1100: seg0 = 8'b1011_1001;
               4'b1101: seg0 = 8'b1101_1110;
 52
 53
               4'b1110: seg0 = 8'b1111_1001;
        4'b1111: sea0 = 8'b1111 0001;
 54
```

```
swnumber.v
                                                                                            _ D 7 X
C:/Users/oscao/Final_Project/Final_Project.srcs/sources_1/new/swnumber.v
                                                                                                  ×
Q 📓 ← → X 📵 🗈 🗙 // 🎟 🖼 Q
                                                                                                  o
                 4'b1100: seg0 = 8'b1011 1001;
 51
                 4'b1101: seg0 = 8'b1101 1110;
 52
 53
                 4'b1110: seg0 = 8'b1111 1001;
54
                 4'b1111: seg0 = 8'b1111 0001;
55 ⊖
             endcase
56 🖨
57
58 ⊖
        always @(Rp_data[7:4]) begin
59 D
            case (Rp_data[7:4])
 60
                 4'b0000: seg1 = 8'b1100_0000;
                 4'b0001: seg1 = 8'b1111_1001;
 61
 62
                 4'b0010: seg1 = 8'b1010 0100;
 63
                 4'b0011: seg1 = 8'b1011 0000;
 64
                 4'b0100: seg1 = 8'b1001 1001;
 65
 66
                 4'b0101: seg1 = 8'b1001 0010;
 67
                 4'b0110: seg1 = 8'b1000 0010;
                 4'b0111: seg1 = 8'b1111_1000;
 68
 69
 70
                 4'b1000: seg1 = 8'b1000 0000;
 71
                 4'b1001: seg1 = 8'b1001_0000;
 72
                 4'b1010: seg1 = 8'b1111_0111;
 73
                 4'b1011: seg1 = 8'b1111_1100;
 74
 75
                 4'b1100: seg1 = 8'b1011 1001;
76
                 4'b1101: seg1 = 8'b1101 1110;
77
                 4'b1110: seg1 = 8'b1111 1001;
78
                 4'b1111: seg1 = 8'b1111_0001;
79 ⊖
             endcase
80 白
        end
81
82 E
        always @(Rp_data[11:8]) begin
83 🖨
           case (Rp_data[11:8])
                 4'b0000: seg2 = 8'b1100 0000;
 84
 85
                 4'b0001: seg2 = 8'b1111 1001;
 86
                 4'b0010: seg2 = 8'b1010_0100;
 87
                 4'b0011: seg2 = 8'b1011_0000;
 88
                4'b0100: seg2 = 8'b1001_1001;
 89
                 4'b0101: seg2 = 8'b1001_0010;
 90
                 4'b0110: seg2 = 8'b1000 0010;
 91
 92
                 4'b0111: seg2 = 8'b1111 1000;
 93
                4'b1000: seg2 = 8'b1000 0000;
 94
                 4'b1001: seg2 = 8'b1001 0000;
 95
 96
                 4'b1010: seg2 = 8'b1111_0111;
 97
                 4'b1011: seg2 = 8'b1111_1100;
 98
                 4'b1100: seg2 = 8'b1011_1001;
 99
                 4'b1101: seg2 = 8'b1101_1110;
                 4'b1110: seg2 = 8'b1111_1001;
101
102
                 4'b1111: seg2 = 8'b1111_0001;
103 白
             endcase
104 A end
```

```
swnumber.v
                                                                                        _ D 7 X
C:/Users/oscao/Final_Project/Final_Project.srcs/sources_1/new/swnumber.v
Q 🕍 ← → 🐰 🗎 🗈 🗙 // 🎟 🖼 ♀
                                                                                               o
103 🖨
           endcase
104 🖨
        end
105
106 🖨
        always @(Rp_data[15:12]) begin
107 ⊖
            case (Rp_data[15:12])
108
                4'b0000: seg3 = 8'b1100_0000;
109
               4'b0001: seg3 = 8'b1111_1001;
110 :
               4'b0010: seg3 = 8'b1010_0100;
111
                4'b0011: seg3 = 8'b1011 0000;
112
113 ;
               4'b0100: seg3 = 8'b1001_1001;
114
                4'b0101: seg3 = 8'b1001_0010;
                4'b0110: seg3 = 8'b1000_0010;
115
116
                4'b0111: seg3 = 8'b1111_1000;
117
118
                4'b1000: seg3 = 8'b1000 0000;
119
                4'b1001: seg3 = 8'b1001 0000;
                4'b1010: seg3 = 8'b1111_0111;
121
                4'b1011: seg3 = 8'b1111_1100;
122
123
               4'b1100: seg3 = 8'b1011_1001;
124
               4'b1101: seg3 = 8'b1101_1110;
125
                4'b1110: seg3 = 8'b1111_1001;
126
                4'b1111: seg3 = 8'b1111 0001;
127 🖨
            endcase
128 🖯
       end
129
130 🖨
        always @(D_addr) begin
131 🖯
          if (D_addr == 8'd203 || D_addr == 8'd204 || D_addr == 8'd205) begin
132
               loc = 8'b1111_0000;
133
               seg[0] = seg0;
134 ;
               seg[1] = seg1;
135
               seg[2] = seg2;
136
                seg[3] = seg3;
137 ⊜
            end
138 🖨
139
140
        assign disp_seg_o = seg;
141
        assign disp_an_o = loc;
142
143 @ endmodule
```

Test Bench (Without Seven Segment Display):

```
processor_tb.v
                                                                                   ? _ D 7 X
C:/Users/oscao/Final_Project/Final_Project.srcs/sim_1/new/processor_tb.v
Q 🕍 ← → 🐰 🖹 🛍 🗙 // 🎟 🎏 Q
                                                                                           o
   'timescale 1ns / 1ps
// Company: EE 5193 FPGA & HDL
   // Engineer: Oscar Oscrnic
5: 11
6 1 // Create Date: 08/08/2022 08:38:27 PM
7
   // Design Name: RISC MIPS Processor Test Bench
   // Module Name: processor tb
9
    // Project Name: Final Project
10 // Target Devices:
11 | // Tool Versions:
12 // Description:
13 1/
14
   // Dependencies:
15
16 // Revision:
17 : // Revision 0.01 - File Created
18 : // Additional Comments:
19 1/
21
22
23 module processor_tb();
24
25
       reg clk, rst;
26
       wire PC_ld, PC_clr, PC_inc, D_addr_sel, IR_ld, D_rd, D_wr, W_wr, Rp_rd, Rq_rd, RF_Rp_zero;
       wire [1:0] RF_s;
27
28
      wire [2:0] ALU s;
29
      wire [3:0] RF_W_addr, RF_Rp_addr, RF_Rq_addr;
30
       wire [7:0] PC_addr, D_addr, addr, RF_W_data;
31
       wire [15:0] R_data, IR_data, Rp_data, Rq_data, ALU_data, MUX_data;
32
33 ⊖
      RISC MIPS processor UUT(clk, rst, PC_ld, PC_clr, PC_inc, D_addr_sel, IR_ld, D_rd, D_wr,
34
                           W_wr, Rp_rd, Rq_rd, RF_Rp_zero, RF_s, ALU_s, RF_W_addr, RF_Rp_addr,
35
                           RF_Rq_addr, PC_addr, D_addr, addr, RF_W_data, R_data, IR_data,
36 白
                           Rp_data, Rq_data, ALU_data, MUX_data);
37
38 ⊖
       initial begin
       #5 rst = 1;
39 ;
          #5 rst = 0;
40
41 🖯
      end
42
43 🖯
       initial begin
44
         #5 clk = 0;
45 🖨
          forever begin
46
             #5 clk = ~clk;
47 ⊖
48 🖨
       end
49
50 ⊖
      initial
         #500 $finish;
51 🖨
52
53
54 A endmodule
```

Test Bench (Memory Module):

```
memory_tb.v
                                                                   ? _ D 7 X
C:/Users/oscao/Final_Project/Final_Project.srcs/sim_1/new/memory_tb.v
Q 🕍 🐟 🔏 🖺 🛍 🗶 // 🎟 🏗 Q
                                                                          o
1 'timescale 1ns / 1ps
3 1 // Company: EE 5193 FPGA & HDL
  // Engineer: Oscar Oscrnic
5: //
6 1 // Create Date: 08/09/2022 07:24:05 PM
  // Design Name: Memory Unit Test Bench
8 | // Module Name: memory tb
  // Project Name: Final Project
10 // Target Devices:
   // Tool Versions:
11
   // Description:
12
13
   // Dependencies:
14 !
15
16
   // Revision:
17
   // Revision 0.01 - File Created
   // Additional Comments:
22
23 module memory_tb();
24
25 reg clk, rst;
26 | reg D_rd, D_wr;
  reg [7:0] addr;
27
28 wire [15:0] Rp_data, R_data;
29
30 | memory_unit UUT(D_rd, D_wr, addr, Rp_data, R_data);
31
32 🖯 initial begin
33 :
   #5 rst = 1;
       #5 rst = 0;
34
35 ⊕ end
36
37 🖨 initial begin
    #5 clk = 0;
39 ;
     D_rd = 0;
40
     addr = 8'h44;
41 ⊕ forever begin
42 ;
         #5 clk = ~clk;
43
         D rd = ~D rd;
44 🖯
         if (D_rd)
45 🖨
             addr = addr + 1;
46 0
      end
47 @ end
48
49 endmodule
50 ;
```