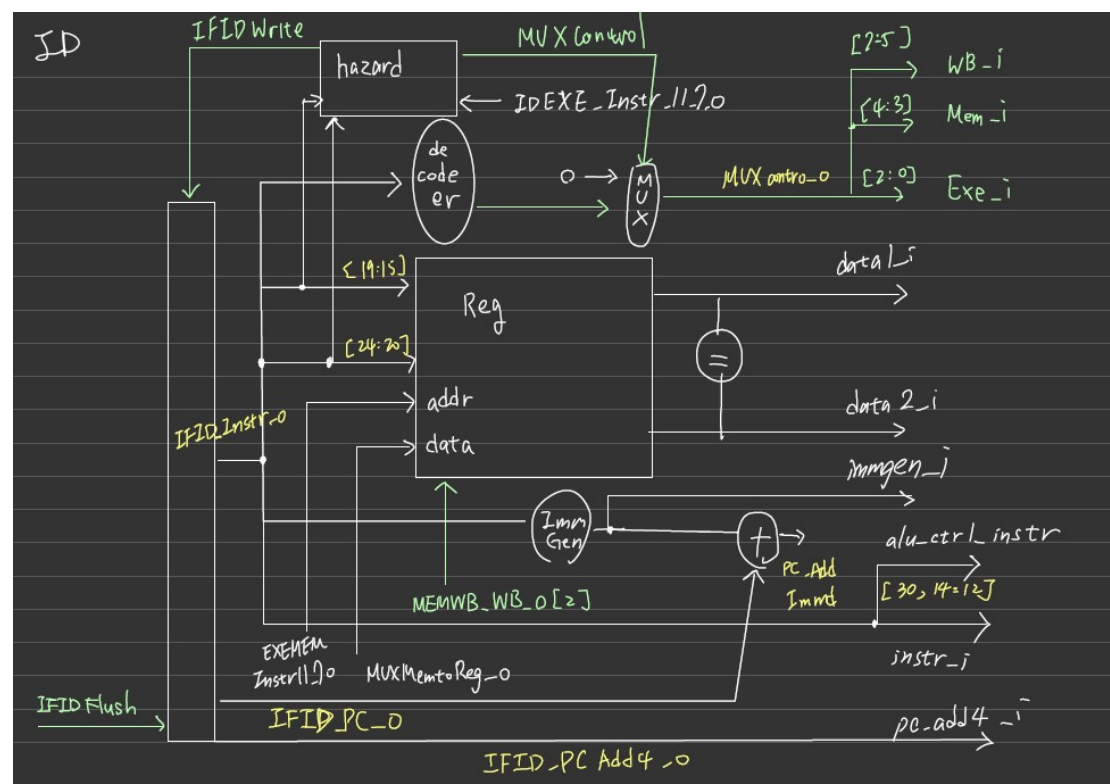
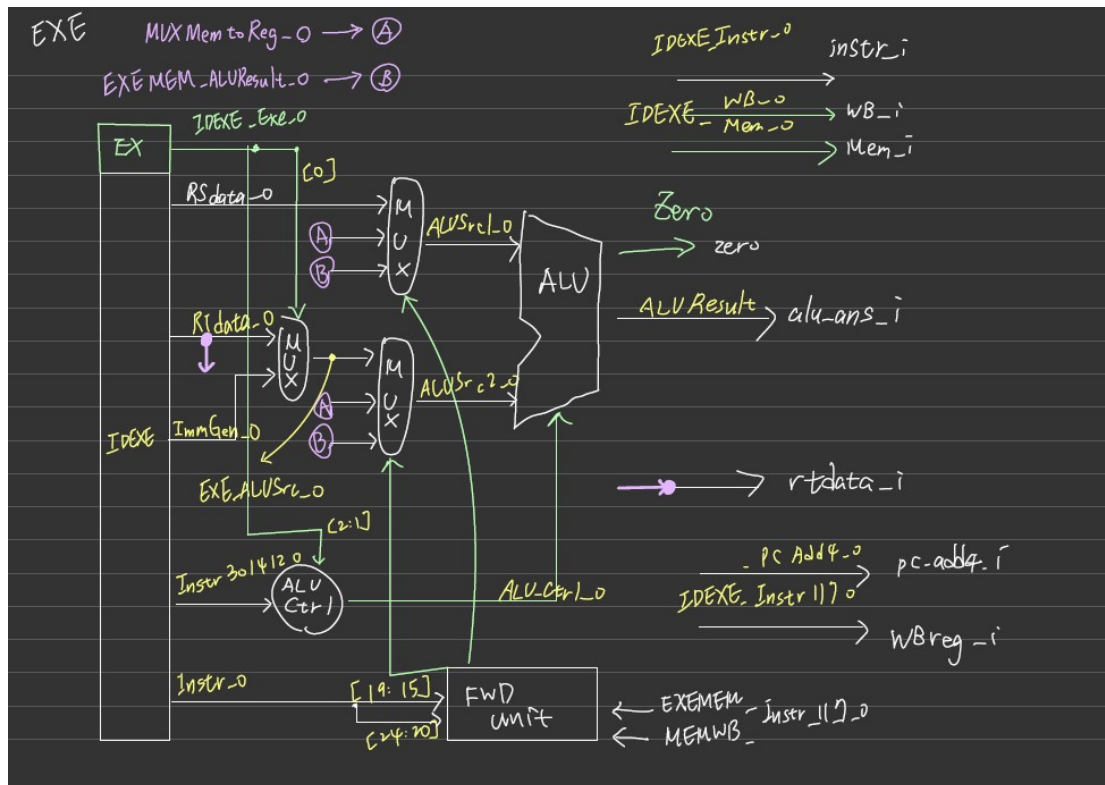
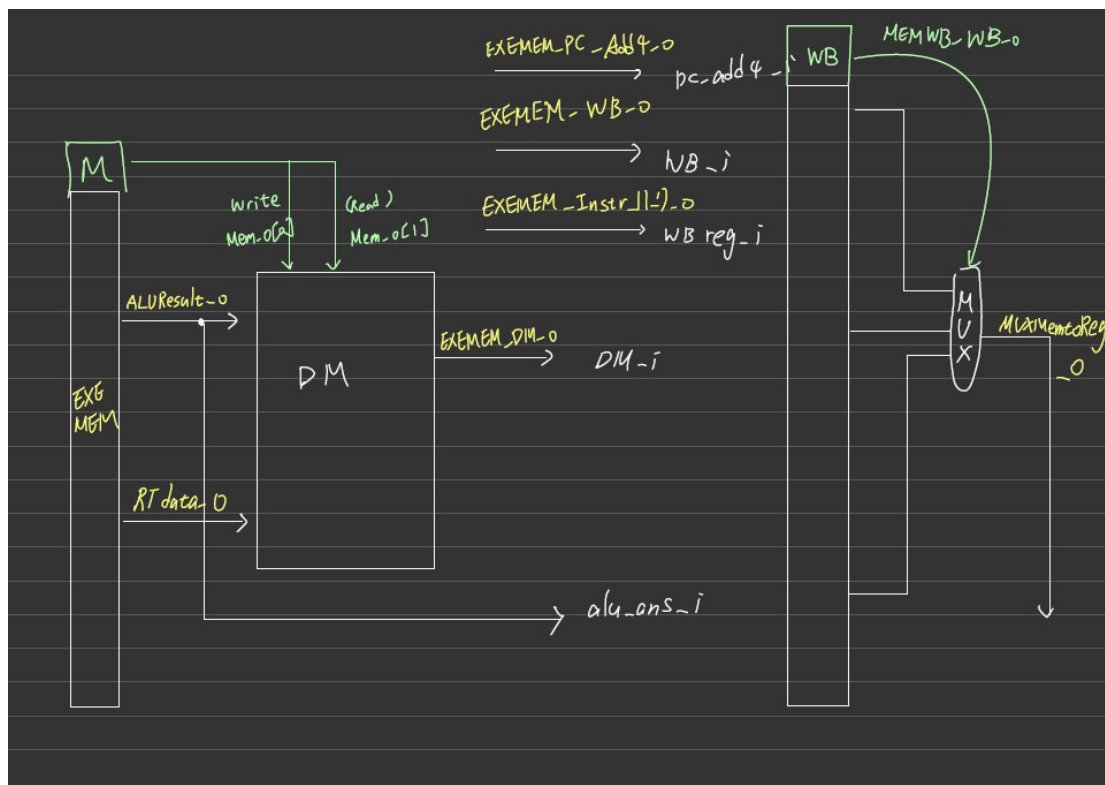


Since this Lab is quite complicated, I separated the block diagram of each parts.





MEM and WB



Implementation results:

```
kkmelon@DESKTOP-TEEM7HG: ~/ComputerOrganization/Lab05_Pipeline_CPU
Pipeline_CPU.v:276: warning: Port 5 (Mem_i) of EXEMEM_register expects 3 bits, got 2.
Pipeline_CPU.v:276:      : Padding 1 high bits of the port.
Pipeline_CPU.v:276: warning: Port 13 (Mem_o) of EXEMEM_register expects 3 bits, got 2.
Pipeline_CPU.v:276:      : Padding 1 high bits of the port.

***** CASE 1 *****
Testcase 1 pass
***** CASE 2 *****
Testcase 2 pass
***** CASE 3 *****
Testcase 3 pass
***** CASE 4 *****
Testcase 4 pass
***** CASE 5 *****
Testcase 5 pass
***** CASE 6 *****
Testcase 6 pass
***** CASE 7 *****
Testcase 7 pass
***** CASE 8 *****
Testcase 8 pass
***** CASE 9 *****
Testcase 9 pass
***** CASE 10 *****
Testcase 10 pass
***** CASE 11 *****
Testcase 11 pass
***** CASE 12 *****
Testcase 12 pass
***** CASE 13 *****
Testcase 13 pass

Basic Score:30
Medium Score:40
Advanced Score:30
Total Score:100
kkmelon@DESKTOP-TEEM7HG:~/ComputerOrganization/Lab05_Pipeline_CPU$ _
```

Problems encountered:

1. Register output should be reset. We didn't set the value to zero during initial reset, so our PC value is wrong initially.

```
Lab5/Lab5Code/EXEMEM_register.v
@@ -21,14 +21,25 @@ module EXEMEM_register (
21      21      output reg [31:0] pc_add4_o
22      22      );
23      23      /* Write your code HERE */
24      - always @(posedge clk_i) begin
25      -     instr_o <= instr_i;
26      -     WB_o <= WB_i;
27      -     Mem_o <= Mem_i;
28      -     zero_o <= zero_i;
29      -     alu_ans_o <= alu_ans_i;
30      -     rtdata_o <= rtdata_i;
31      -     WBreg_o <= WBreg_i;
32      -     pc_add4_o <= pc_add4_i;
33
34      + always @(posedge clk_i or negedge rst_i) begin
35      +     if(!rst_i) begin
36      +         instr_o <= 0;
37      +         WB_o <= 0;
38      +         Mem_o <= 0;
39      +         zero_o <= 0;
40      +         alu_ans_o <= 0;
41      +         rtdata_o <= 0;
42      +         WBreg_o <= 0;
43      +         pc_add4_o <= 0;
44      +     end else begin
45      +         instr_o <= instr_i;
46      +         WB_o <= WB_i;
47      +         Mem_o <= Mem_i;
48      +         zero_o <= zero_i;
49      +         alu_ans_o <= alu_ans_i;
50      +         rtdata_o <= rtdata_i;
51      +         WBreg_o <= WBreg_i;
52      +         pc_add4_o <= pc_add4_i;
53      +     end
54
55      end
56
57      endmodule
```

2. I didn't put branch signal in the MUXPCSrc, so the PC value was wrong

Showing 4 changed files with 3,157 additions and 1,171 deletions.

```
Lab5/Lab5Code/Pipeline_CPU.v

@@ -97,7 +97,7 @@ Adder PC_plus_4_Adder(
    97    97    .src2_i(Imm_4),
    98    98    .sum_o(PC_Add4)
    99    99    );
+ 100      - assign MUXPCSrc = (Rdata_o == RTdata_o) ? 1'b1 : 1'b0; // beq
+ 100      + assign MUXPCSrc = (Branch && Rdata_o == RTdata_o) ? 1'b1 : 1'b0; // beq
    101    101    MUX_2to1 MUX_PCSrc(
    102    102    .data0_i(PC_Add4),
    103    103    .data1_i(PC_Add_Immediate),
```

3. Our opcode is not assigned to instr_i[6:0], so the decoder output was always xxx;

Showing 5 changed files with 3,380 additions and 3,210 deletions.

```
Lab5/Lab5Code/Decoder.v

@@ -15,14 +15,14 @@ module Decoder(
    15    15    );
    16    16
    17    17    //Internal Signals
+ 18      - wire    [7:1:0]    opcode;
+ 18      + wire    [7:1:0]    opcode = instr_i[6:0];
    19    19    wire    [3:1:0]    funct3;
    20    20    wire    [3:1:0]    Instr_field;
    21    21    wire    [9:0]    Ctrl_o;
    22    22
    23    23    /* Write your code HERE */
    24    24    always @(*) begin
+ 25      - casez(instr_i)
+ 25      + casez(opcode)
    26    26          7'b0110011: begin //R-type
    27    27              RegWrite = 1;
    28    28              Branch = 0;
```

4. I fed the wrong value to EXEMEM_register initially, it should be rt_data

```
Lab5/Lab5Code/Pipeline_CPU.v

@@ -281,7 +281,7 @@ EXEMEM_register EXEMEM(
281 281     .Mem_i(IDEXE_Mem_o),
282 282     .zero_i(Zero),
283 283     .alu_ans_i(ALUResult),
284 -     .rtdata_i(ALUSrc2_o),
+     .rtdata_i(IDEXE_RTdata_o),
285 285     .WBreg_i(IDEXE_Instr_11_7_o),
286 286     .pc_add4_i(IDEXE_PC_add4_o),
287 287
```

5. In some testcases, there are slti and slli instructions, but we didn't cover them in Lab4, so the decoder value was wrong.

```
krz-max committed 4 hours ago

Showing 5 changed files with 1,414 additions and 1,342 deletions.

Lab5/Lab5Code/Decoder.v

@@ -45,7 +45,7 @@ always @(*) begin
45 45     MemWrite = 0;
46 46     // ALUSrcA = 1'bx;
47 47     ALUSrc = 1;
48 -     ALUOp = (funct3 == 3'b010) ? 2'b10 : 2'b00; // 3'b010 is for slti,
+     ALUOp = (funct3 == 3'b010 || funct3 == 3'b001) ? 2'b10 : 2'b00; //
49 49     end
50 50     7'b0000011: begin //Load
51 51     RegWrite = 1;
```

6. Because in Lab4, the writeback MUX was divided into two 2to1 MUX, and we modified using a 3to1 MUX, so the control bit should not be don't care, or the output of the MUX will be xxx.

Showing 6 changed files with 1,437 additions and 1,568 deletions.

Lab5/Lab5Code/Decoder.v

```
@@ -88,7 +88,7 @@ always @(*) begin
88 88 Branch = 0;
89 89 Jump = 1;
90 90 WriteBack1 = 1;
91 - WriteBack0 = 1'bx;
91 + WriteBack0 = 1'b0;
92 92 MemRead = 0;
93 93 MemWrite = 0;
94 94 // ALUSrcA = 0;

@@ -100,7 +100,7 @@ always @(*) begin
100 100 Branch = 0;
101 101 Jump = 1;
102 102 WriteBack1 = 1;
103 - WriteBack0 = 1'bx;
103 + WriteBack0 = 1'b0;
104 104 MemRead = 0;
105 105 MemWrite = 0;
106 106 // ALUSrcA = 1;
```

7. I am not sure why in there wasn't a shiftleft1 in Lab4, so I tried removing the unit in Lab5, after that, we passed all the testcases.

Showing 4 changed files with 1,470 additions and 1,347 deletions.

Lab5/Lab5Code/Pipeline_CPU.v

```
@@ -191,7 +191,7 @@ Shift_Left_1 SL1(
191 191 .data_o(SL1_o)
192 192 );
193 193 + Adder Branch_Adder(
194 - .src1_i(SL1_o),
194 + .src1_i(Imm_Gen_o),
195 195 .src2_i(IFID_PC_o),
196 196 .sum_o(PC_Add_Immediate)
197 197 );
```