Computer Organization

1. The input fields of each pipeline register:

```
// IF/ID
Pipeline_Reg #(.size(64)) IF_ID(
                .clk_i(clk_i),
                .rst_n(rst_n),
                .data_i({PC_add1, instr}),
                .data_o({PC_add1_ID, instr_ID})
                );
```

```
// ID/EX
Pipeline_Reg #(.size(161)) ID_EX(
            .clk_i(clk_i),
            .rst_n(rst_n),
            .data_i({ReadData1, ReadData2, RegWrite, ALUOP, ALUSrc, RegDst, Branch, MemWrite, MemRead, MemtoReg, signextend, instr_ID[20:0], PC_add1_ID}),
            .data_o({ReadData1_EX, ReadData2_EX, RegWrite_EX, ALUOP_EX, ALUSrc_EX, RegDst_EX, Branch_EX, MemWrite_EX, MemRead_EX, MemtoReg_EX, signextend_EX, instr_EX, PC_add1_EX})
            );
```

```
// EX/MEM
Pipeline_Reg #(.size(108)) EX_MEM(
            .clk_i(clk_i),
            .rst_n(rst_n),
            .data_i({PC_add2, ALUResult, zero, WriteReg_addr, ReadData2_EX, RegWrite_EX, Branch_EX, MemRead_EX, MemWrite_EX, MemtoReg_EX}),
            .data_o({PC_add2_MEM, ALUResult_MEM, zero_MEM, WriteReg_addr_MEM, ReadData2_MEM, RegWrite_MEM, Branch_MEM, MemRead_MEM, MemWrite_MEM, MemtoReg_MEM})
            );
```

```
// MEM/WB
Pipeline_Reg #(.size(72)) MEM_WB(
                .clk_i(clk_i),
                .rst_n(rst_n),
                .data_i({DM_ReadData, ALUResult_MEM, WriteReg_addr_MEM, RegWrite_MEM, MemtoReg_MEM}),
                .data_o({DM_ReadData_WB, ALUResult_WB, WriteReg_addr_WB, RegWrite_WB, MemtoReg_WB})
                );
```

2. Compared with lab4, the extra modules:

Pipeline_reg.v

Pipeline_CPU.v

3. Explain your control signals in sixth cycle (both test patterns C0_P5_test_data1 and C0_P5_test_data2 are needed):

Picture:

C0_P5_test_data1	C0_P5_test_data2
ALUOp0 = 0 ALUOp1 = 1 ALUSrc = 0 Branch = 0 Regwrite = 1 RegDst = 1 MemtoReg = 0	ALUOp0 = 0 ALUOp1 = 1 ALUSrc = 1 Branch = 0 Regwrite = 1 RegDst = 0 MemtoReg = 0

MemWrite = 0 MemRead = 0	MemWrite = 0 MemRead = 0

4. Problems you met and solutions:

1.MemtoReg 控制的 MUX 圖畫反

一開始怎麼跑都是錯的，幸好有看討論區有人提及就解決了

2.size 沒算好

因為我嫌一個一個帶太麻煩，所以一次就是帶整個階段要帶的 signal

不過也造成了 size 大小很容易算錯，或許寫一起不是一個最好的選擇。

5. Summary:

這次是本學期最後一次作業，而且也是把從 HW2 開始寫的 verilog 一步步強化、擴建，最終組合成了這個有基本功能的 pipeline CPU，做完的瞬間非常有成就感，同時也對 CPU 的運作相比過去相比了解了非常多，很感謝這學期教授與助教們的細心指導。