Computer Organization

1. The input fields of each pipeline register:

IF/ID:
Adder: PC+4.
Instr: The instruction from instruction memory.

ID/EX:
Adder1: PC+4.
ReadData1: The data of rs.
ReadData2: The data of rt.
SignExtend: The output of sign extension.
Rt: The index of rt.
Rd: The index of rd.
RegWrite: The control signal of whether to write register.
MemtoReg: The control signal of choosing the source of write back data to be from the result of ALU or from memory.
Branch: The control signal of whether the instruction is branch or not.
MemRead: The control signal of whether can or cannot read memory.
MemWrite: The control signal of whether can or cannot write memory.
ALUSrc: The control signal of choosing the second input of ALU to be rt data or the sign extension.
ALUOp: The ALU operation code.
RegDst: The control signal of choosing the write back destination to be rt or rd.

EX/MEM:
Adder2: The branch destination.
ALUresult: The ouput of ALU.
ReadData2: The data from rt.
WriteReg: The reference of write back register.
RegWrite: The control signal of whether can or cannot write register.
Branch: The control signal of whether the instruction is branch or not.
Zero: Whether the output of ALU is zero or not.
MemtoReg: The control signal of choosing the source of write back data to be from the result of ALU or from memory.
MemRead: The control signal of whether can or cannot read memory.
MemWrite: The control signal of whether can or cannot write memory.

MEM/WB:
ReadData: The data from the memory.
ALUresult: The output of ALU.
WriteReg: The reference of write back register.
RegWrite: The control signal of whether can or cannot write register.
Branch: The control signal of whether the instruction is branch or not.
MemtoReg: The control signal of choosing the source of write back data to be from the result of ALU or from memory.

2. Compared with lab4, the extra modules:
IF_ID.v: The registers of IF/ID.
ID_EX.v: The registers of ID/EX.
EX_MEM.v: The registers of EX/MEM.
MEM_WB.v: The registers of MEM/WB.

3. Explain your control signals in sixth cycle (both test patterns CO_P5_test_data1 and CO_P5_test_data2 are needed):

The control signals in sixth cycle are expressed as the tables below. The row means the control signal and the column means the instruction in the stage.

The symbols in the blocks:
1: The signal is one.
0: The signal is zero.
x: The control signal doesn't exist in this stage.

CO_P5_test_data1:

Stages & instr / Control Signals	ID or	EX and	MEM addi	WB addi
RegWrite	1	1	1	1
ALUOp	010	010	x	x
ALUSrc	0	0	x	x
RegDst	1	1	x	x
Branch	0	0	0	x
MemWrite	0	0	0	x
MemRead	0	0	0	x
MemtoReg	1	1	1	1

CO_P5_test_data2:

Stages & instr / Control Signals	ID sw	EX addi	MEM addi	WB addi
RegWrite	0	1	1	1
ALUOp	000	011	x	x
ALUSrc	1	1	x	x
RegDst	0	0	x	x
Branch	0	0	0	x
MemWrite	1	0	0	x
MemRead	0	0	0	x
MemtoReg	0	1	1	1

4. Problems you met and solutions:

Pipeline cpu is more complex than single cycle cpu. Needs to systematically declare the wires and the registers to avoid confusion. And for debug, display all the pipeline registers to view and fix the problem more efficiently.

5. Summary:

在本次的作業中，實作一個 pipeline cpu 讓我對其結構更加熟悉，順便提升我撰寫 verilog 的能力。建構這樣的系統以及 debug 雖然花了我不少時間，但學到的東西頗為充實。