Computer Organization

1. The input fields of each pipeline register:

IF/ID

reg11 - input: PC_add1 (32b), output: PC_add_temp0 (32b)

reg12 - input: instr0 (32b), output: instr (32b)

ID/EX

reg21 - input: PC_add_temp0 (32b), output: PC_add_temp1 (32b)

reg22 - input: ReadData1_temp (32b), output: ReadData1 (32b)

reg23 - input: ReadData2_temp (32b), output: ReadData2 (32b)

reg24 - input: signextend_temp (32b), output: signextend (32b)

reg25 - input: instr[20:11] (10b), output: instr_3 (10b)

let Sexe = all the signal for EXE = {RegDst (2b), ALUOP (3b), ALUSrc (1b)}

let Smem = all the signal for MEM = {Branch (1b), MemWrite (1b), MemRead (1b)}

let Swb = all the signal for EXE = {MemtoReg (2b), RegWrite (1b)}

regEX - input: Sexe_temp, output: Sexe

regMEM1 - input: Smem_temp0, output: Smem_temp1

regWB1 - input: Swb_temp0, output: Swm_temp1

EX/MEM

reg31 - input: PC_add2_temp (32b), output: PC_add2 (32b)

reg32 - input: zero_temp (1b), output: zero (1b)

reg33 - input: ALUResult_temp (32b), output: ALUResult (32b)

reg34 - input: ReadData2 (32b), output: ReadData2_temp2 (32b)

reg35 - input: WriteReg_addr (5b), output: WriteReg_addr_temp0 (5b)

regMEM2 - input: Smem_temp1, output: Smem

regWB2 - input: Swb_temp1, output: Swm_temp2

MEM/WB

reg41 - input: DM_ReadData_temp (32b), output: DM_ReadData (32b)

reg42 - input: ALUResult (32b), output: ALUResult_temp2 (32b)

reg43 - input: WriteReg_addr_temp0 (5b), output: WriteReg_addr_temp1 (5b)

regWB3 - input: Swb_temp2, output: Swm

2. Compared with lab4, the extra modules:

Pipline_Reg (in Pipeline_reg.v) :

when reset is 0, assign 0 to output,

else when posedge of clk, assign input to output

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

WB : MemtoReg & RegWrite
MEM : Branch & MemWrite & MEMRead
EX : ALUSrc & RegDst & ALUOP
the control signal of a cycle is the signal for an instruction under the stage.

Picture:

CO_P5_test_data1	CO_P5_test_data2
RegDst : 01 ALUSrc : 0 ALUOP : 010 Branch : 0 MemWrite : 0 MemRead : 0 MemtoReg : 00 RegWrite : 1	RegDst : 00 ALUSrc : 1 ALUOP : 011 Branch : 0 MemWrite : 0 MemRead : 0 MemtoReg : 00 RegWrite : 1

4. Problems you met and solutions:

I put the data in MemtoReg in a wrong way and got x for almost all the register. I traced back the process by printing out the signals when posedge and finally found out that what is should do is just reverse the connection of the MUX.

5. Summary:

In this lab, I modified the single cycle processor designed in Lab4 to a pipelined processor by inserting pipeline registers and connecting them to the original design.