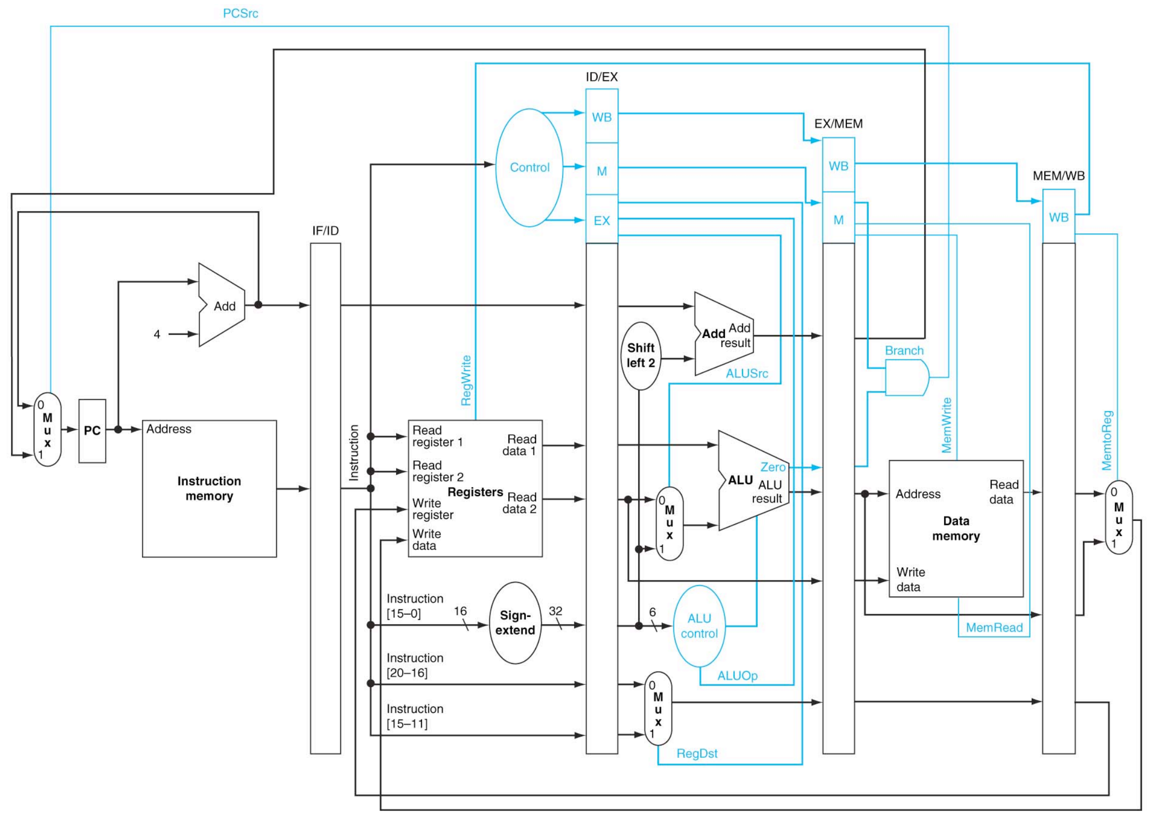
CprE 381 – Computer Organization and Assembly Level Programming

Project C, Version 1.2
First update: 11/12/2013
Last update: 11/27/2013 (regarding project report)

Note: This is a ***three-week project****. You will design and implement three versions of a* ***PipeLined Processor (PLP)****. The project involves substantial design, implementation, integration, and test tasks.* ***Due to the complexity of the assignment, this document is subject to minor changes between now and the due date****.*



Textbook Figure 4.51

Part1: PLPv1, the Simple MIPS Pipeline

You will implement the simple MIPS pipeline similar to Figure 4.51 in the textbook. Start from you single-cycle implementation, insert pipeline registers to make it pipeline. The requirements are as follows:

a. Your top-level CPU composition must be in cpu.vhd, and it must be strongly structural as in Project B.
b. You may model pipeline registers in any style you like. Use reg.vhd as a template.

Demostration Create testing codes that covers all instructions used in your bubble sort code. Demo that your PLPv1 run the codes correctly. During this time, you will describe the various components of your design and how they work together.

Note that you may avoid pipeline hazards by inserting nop instructions. For example, for the following code:

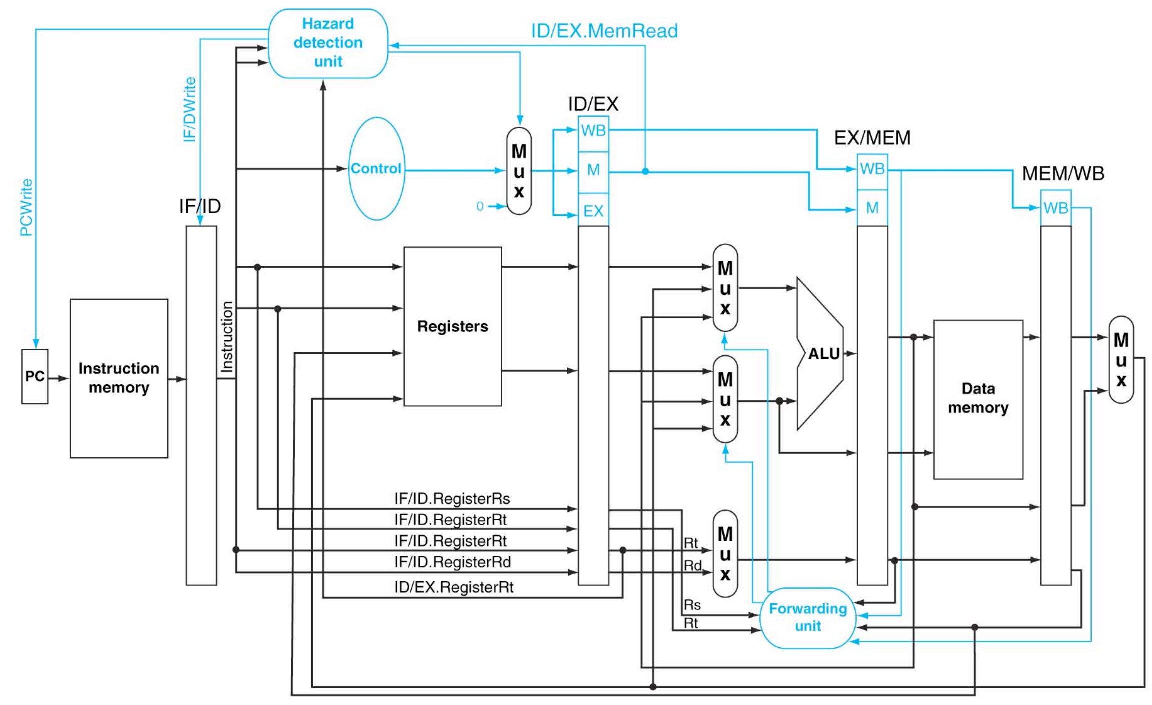
```
sll   $t1, $t2, 2
add   $t1, $a0, $t1
lw    $t0, 0($t1)
lw    $t2, 4($t1)
sw    $t2, 0($t1)
sw    $t0, 4($t1)
```

You may avoid data hazards by inserting the following nop instructions.

```
sll   $t1, $t2, 2
nop                   # insert 2 nops to avoid data hazard
nop
add   $t1, $a0, $t1
nop                   # insert 2 nops to avoid data hazard
nop
lw    $t0, 0($t1)
lw    $t2, 4($t1)
nop                   # insert 2 nops to avoid data hazard
nop
sw    $t2, 0($t1)
sw    $t0, 4($t1)
```

Part 2: PLPv2, pipelined processor with data forwarding and hazard detection.

Add a data forwarding unit and a data hazard detection unit to your PLPv1 to form PLPv2, in a way similar to what Figure 4.60 shows.



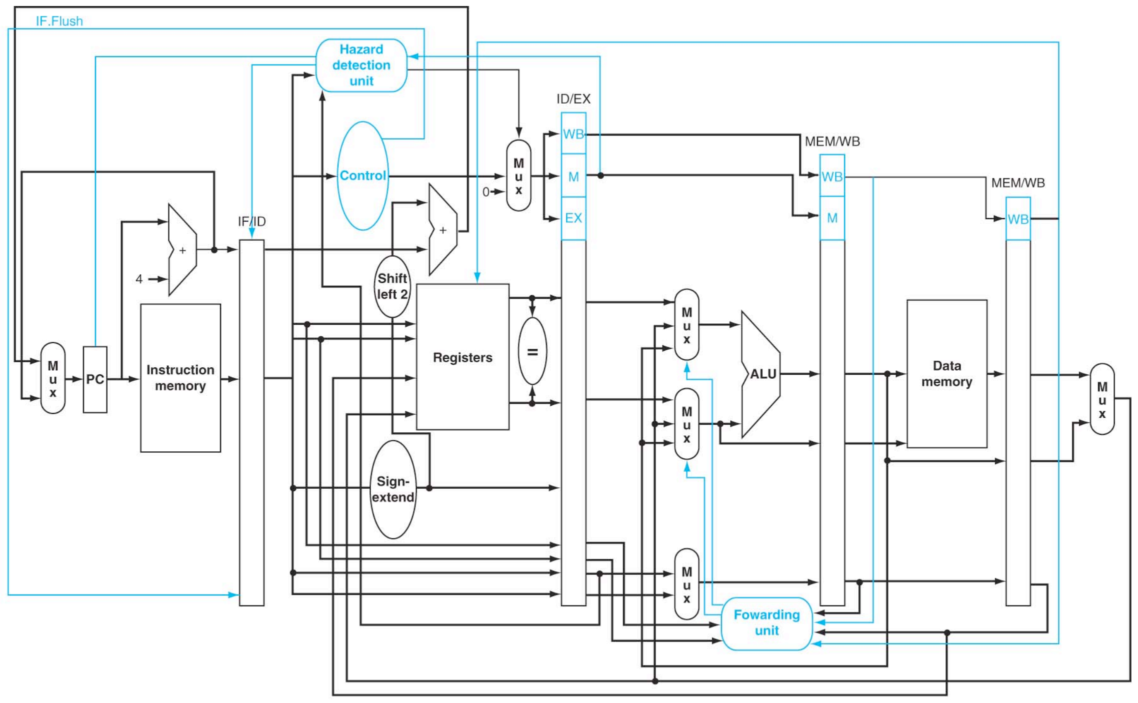
Textbook Figure 4.60

(a) Implement the forwarding unit as a VHDL module using your preferred VHDL style.
(b) Implement the hazard detection unit as a VHDL module using your preferred VHDL style.
(c) Integrate your forwarding unit and hazard detection unit into PLPv1 to form your PLPv2. You may need to revise the PC register and the IF/ID and ID/EX pipeline registers.

Demonstration Create testing codes that covers all instructions used in the bubble sort code. Demo that your PLPv2 runs the codes correctly. During this time, you will describe howyou're your forwarding and hazard detection units work.

3) PLPv3: Pipelined processor with data forwarding, data hazard detection, and control hazard detection.

The following diagram shows the MIPS pipeline with branch comparison at the ID stage. *You are NOT required to implement that pipeline, but you may borrow the ideas and the implementation techniques from the design.*



P&H Figure 4.65.

Instead of resolving branches at the ID stage, you may either keep it at the MEM stage or move it to the EX stage. The pipeline bubbles will be 3 cycles and 2 cycles, respectively, for each taken branch. If a branch is resolved at the EX stage, instructions at the IF and ID stages will be flushed. If a branch is resolved at the MEM stage, instructions at the IF, ID and EX stages will be flushed.

Revise your PLPv3 implementation to support the branch instructions.

Demonstration Demo that your PLPv3 can run the bubble sort code correctly.

PROJECT REPORT:

- Write a short report (a couple of ages) in free format to describe your implementation.
    - Part 1: Describe the fields in each pipeline register.
    - Part 2: Describe your VHDL programming for data forwarding and pipeline stalling.
    - Part 3: State at what stage the branch comparison is done, and then describe your VHDL programming for pipeline flushing.

DEMO AND SUBMISSION:

- Print out a hard copy of the evaluation from (Project-C-*Eval.pdf* on the class web site).
- Complete the demos to your TA and ask your TA to sign on the evaluation form.

- Create a zip file *Project-C-submit.zip*, including the completed code and document from the four lab parts and the lab report. Put code for parts 1, 2, 3 under directories /P1, /P2, and /P3, respectively.
- The file names in your zip file should be self-explained.
- Submit the zip file on BlackBoard Learn under “Project C” assignment.