x

# ECE 3055a Laboratory Assignment 3
## Due Date: Thursday, March 2

**(Part I - 70%)** Once the MIPS is pipelined as in Lab 2, data hazards can occur between the five instructions present in the pipeline. As an example consider the following program:

```
Sub    $2,$1,$3
Add    $4,$2,$5
```

The subtract instruction stores a result in register 2 and the following add instruction uses register 2 as a source operand. The new value of register 2 is written into the register file by SUB $2,$1,$3 in the write-back stage after the old value of register 2 was read out by ADD $4,$2,$5 in the decode stage. This problem is fixed by adding two forwarding muxes to each ALU input in the execute stage. In addition to the existing values feeding in the two ALU inputs, the forwarding multiplexers can also select the last ALU result or the last value in the data memory stage. These muxes are controlled by comparing the rd, rt, and rs register address fields of instructions in the decode, execute, or data memory stages. Instruction rd fields will need to be added to the pipelines in the execute, data memory, and write-back stages for the forwarding compare operations. Since register 0 is always zero, do not forward register 0 values.

Add forwarding control to the pipelined model developed in Lab 2. Test your VHDL model by running a simulation of the example program shown in Figures 6.41-42 of *Computer Organization and Design The Hardware/Software Interface* by Patterson and Hennessy.

Two forwarding multiplexers must also be added to the Idecode module so that a register file write and read to the same register work correctly in one clock cycle. If the register file write address equals one of the two read addresses, the register file write data value should be forwarded out the appropriate read data port instead of the normal register file data value.

Use the following test program that has been modified for eight registers:

```
Sub    $2,$1,$3
And    $4,$2,$5
Or     $3,$2,$6
Add    $5,$4,$2
Slt    $1,$6,$7
```

Sections 6.4 and 6.5 of *Computer Organization and Design The Hardware/Software Interface* contain additional background information on forwarding.

**(Part II - 20%)** When a branch is taken, several of the instructions that follow a branch have already been loaded into the pipeline. A process called flushing is used to prevent the execution of these instructions. Several of the pipeline registers are cleared so that these instructions do not store any values to registers or memory or cause a forwarding operation. Add branch flushing to the pipelined MIPS VHDL model as shown in Figures 6.51 and 6.52 of the *Computer Organization and Design The Hardware/Software Interface* by Patterson and Hennessy.

Use the following test program:

```
        Beq $0,$1,label1
        Add $1,$1,$2
        Add $2,$0,$1
label1: Beq $1,$2,label2
        Add $1,$1,$1
        Add $2,$0,$0
        Nop
label2: Beq $2,$1,label2
```

Section 6.6 of *Computer Organization and Design The Hardware/Software Interface* contains additional background information on branch hazards.

**(Part III - 10%)** After completing Part I, Add LW/SW forwarding to the pipelined model. This will allow an LW to be followed by an SW that uses the same register. One solution is suggested in *Computer Organization and Design The Hardware/Software Interface* by Patterson and Hennessy on page 488. Write a test program and verify correct operation in a simulation.