Score: _____ Name: _____

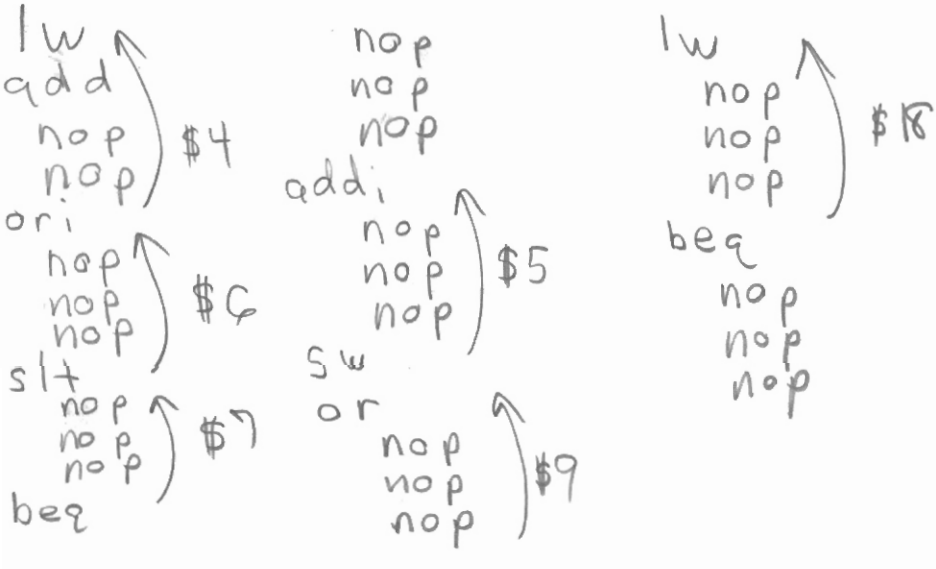
ECE 3055 Quiz 4 Wednesday, February 16

The program below is executed on the 5 stage pipelined MIPS described in chapter 6. Answer the following questions about this program.

```
loop_top:   lw      $4,100($0)
            add     $5,$5,$7
            ori     $6,$4,4
            slt     $7,$6,$3
            beq     $7,$0,foobar
            addi    $5,$5,6
            sw      $7,200($5)
foobar:     or      $9,$5,$9
            lw      $18,100($9)
            beq     $18,$0,loop_top
```

Assume the control unit **does not have** any hazard detection, forwarding, a new branch compare circuit, or automatic branch flushing, and that the register file will not write and then read a new register value in one clock cycle. Rewrite the code sequence by adding the minimum number of NOP instructions (do not reorder or change instructions) to eliminate all potential data and branch hazards. Assume other non-NOP instructions follow the last branch in the original code sequence above.

Total number of NOPs required 23



Assume the control unit is improved by adding the hazard and forwarding unit as outlined in the text and the current lab assignment, adding a branch compare unit to the decode stage, and the register file writes then reads a new value in a single clock cycle. Determine the number of clock cycles required to complete the first loop execution (i.e. executes code in loop and branches back to top of loop and is just ready to fetch sw again) of the original code sequence. Assume the inner branch is not taken.

Assume the processor starts this program initially at power up. if there were no hazards or branch

flushing, the original program would require 10 clock cycles for execution. (***do not include the time to initially fill the pipeline at power up***).

But this program will need to stall and/or flush the pipeline an additional 2 clock cycles so

a total of 12 **clock cycles** is required for execution (***do not include the time to initially fill the pipeline***).

This program achieves a CPI (clocks per instruction) of 1.2 .(*do not include time to fill the pipeline here*)