Problem Set 5

Justin Ely

605.204.81.FA15 Computer Organization

06 October, 2015

2.20)

2.25.1)

The J-type instruction would be the most appropriate format. OR I-TYPE? Kinda like a branch?

2.25.2)

```
    sgt, $t3, $t2, $zero
    bnq, $t3, $zero, sub
sub:
    addi, $t3, $t3, -1
    j LOOP
```

2.26.1)

This MIPS loop subtracts 1 from $t1 and adds 2 to $s2 every iteration that $t1 is not equal to 0. With $t1 initially set to 10, the loop will go through 10 iterations before finishing. This means $s2 will have a value of 20 after the last iteration.

2.26.2)

A literal translation line-by-line looks like the following for each iteration:

```
if (0 < i) {
    TEMP = 1;
}
else {
    TEMP = 0'
}
```

```
if (TEMP == 0){
    break;
}

i--;
B = B+2;
```

While a more concise version of the whole loop could be done with a while loop:

```
while (i > 0){
    i--;
    B = B+2;
}
```

2.26.3)

Each loop executers 5 instructions, except for the last which only executes 2 (passing by the add, subtract, and jump). If $t1 is initialized to N, then the total number of instructions executed is $5 * (N - 1) + 2$.

Pseudocode: abs $s4, $s1)

The absolute value pseudocode takes 3 MIPS instructions to complete: SLT to check if less than 0, BEQ skip over the SUB step if already positive, and a SUB step to correct the negative.

```
     slt, $t1, $s1, $0
     beq, $t1, $zero, skip
     sub, $s4, $0, $s1
skip:
```

Pseudocode: rol $t7, $t3, 8)

The roll left pseudo code takes 3 MIPS instructions: a SLL to rotate the non-discarded bits to their final place, a SRL to move the bits discarded by the previous shift to their final location, and an OR to combine the two temporary outputs into the final register.

```
sll $t1, $t3, 8
srl $t2, $t3, 24
or $t7, $t2, $t1
```

Pseuodocode: ld $t5, 0($t8))