

ICS 233 – Computer Architecture & Assembly Language

Assignment 2: MIPS Instructions and Assembly Language

1. (2 pts) Bits have no inherent meaning. Given the 32-bit pattern:

1010 1101 0001 0000 0000 0000 0000 0010

What does it represent, assuming it is ...

- A 2's complement signed integer?
 - A MIPS instruction?
2. (2 pts) Find the shortest sequence of MIPS instructions to:
- Determine if there is a carry out from the addition of two registers **\$t3** and **\$t4**. Place the carry out (**0** or **1**) in register **\$t2**. It can be done in two instructions.
 - Determine the absolute value of a signed integer. Show the implementation of the following pseudo-instruction using three real instructions:
abs \$t1, \$t2
3. (4 pts) For each pseudo-instruction in the following table, produce a minimal sequence of actual MIPS instructions to accomplish the same thing. You may use the **\$at** for some of the sequences. In the following table, **imm32** refers to a 32-bit constant.

Pseudo-instruction
move \$t1, \$t2
clear \$t5
li \$t5, imm32
addi \$t5, \$t3, imm32
beq \$t5, imm32, Label
ble \$t5, \$t3, Label
bgt \$t5, \$t3, Label
bge \$t5, \$t3, Label

4. (2 pts) Translate the following statements into MIPS assembly language. Assume that *a*, *b*, *c*, and *d* are allocated in \$s0, \$s1, \$s2, and \$s3. All values are signed 32-bit integers.
- if ((a > b) || (b > c)) {d = 1;}**
 - if ((a <= b) && (b > c)) {d = 1;}**

5. (3 pts) Consider the following fragment of C code:

```
for (i=0; i<=100; i=i+1) { a[i] = b[i] + c; }
```

Assume that a and b are arrays of words and the base address of a is in \$a0 and the base address of b is in \$a1. Register \$t0 is associated with variable i and register \$s0 with c. Write the code in MIPS.

6. (3 pts) Add comments to the following MIPS code and describe in one sentence what it computes. Assume that \$a0 is used for the input and initially contains n, a positive integer. Assume that \$v0 is used for the output.

```
begin:    addi $t0, $zero, 0
          addi $t1, $zero, 1
loop:    slt  $t2, $a0, $t1
          bne $t2, $zero, finish
          add $t0, $t0, $t1
          addi $t1, $t1, 2
          j   loop
finish:  add  $v0, $t0, $zero
```

7. (4 pts) The following code fragment processes an array and produces two important values in registers \$v0 and \$v1. Assume that the array consists of 5000 words indexed 0 through 4999, and its base address is stored in \$a0 and its size (5000) in \$a1. Describe in one sentence what this code does. Specifically, what will be returned in \$v0 and \$v1?

```
          add  $a1, $a1, $a1
          add  $a1, $a1, $a1
          add  $v0, $zero, $zero
          add  $t0, $zero, $zero
outer:    add  $t4, $a0, $t0
          lw   $t4, 0($t4)
          add  $t5, $zero, $zero
          add  $t1, $zero, $zero
inner:    add  $t3, $a0, $t1
          lw   $t3, 0($t3)
          bne $t3, $t4, skip
          addi $t5, $t5, 1
skip:    addi $t1, $t1, 4
          bne $t1, $a1, inner
          slt  $t2, $t5, $v0
          bne $t2, $zero, next
          add  $v0, $t5, $zero
          add  $v1, $t4, $zero
next:    addi $t0, $t0, 4
          bne $t0, $a1, outer
```