CDA 4205 Computer Architecture

Kenneth Neads U3036-9779

2/3/2019

Assignment 2: MIPS Instructions and Assembly Language

1. (4 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 …

1. A 2's complement signed integer?
   1. -1,391,460,350
2. A MIPS instruction?
   1. Sw 101011
   2. 01000 = 8 = $t
   3. 10000 = 16 = $s
   4. 0000 0000 0000 0010 = 2
   5. Sw $t, 2($s)
3. (4 pts) Determine the absolute value of a signed integer. Show the implementation of the following pseudo-instruction using three real instructions:

**abs $t1, $t2**

addu $t1, $t2, $zero

bgez $t1, next

subw $t1, $zero, $t2

next:

1. (16 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.
2. **move $t1, $t2**
   1. **addu $t1, $t2, $zero**
3. **clear $t5**
   1. **addu $t5, $zero, $zero**
4. **li $t5, imm32**
   1. **lui $t5, top**
   2. **ori $t5, $t5, bot**
5. **addi $t5, $t3, imm32**
   1. **lui $t6, top**
   2. **ori $t6, $t6, bot**
   3. **add $t5, $t3, $t6**
6. **beq $t5, imm32, Label**
   1. **lui $t6, top**
   2. **ori, $t6, $t6, bot**
   3. **beq $t5, $t6, Label**
7. **ble $t5, $t3, Label**
   1. **slt $t6, $t3, $t5**
8. **bgt $t5, $t3, Label**
   1. **slt $t6, $t3, $t5**
   2. **bne $t6, $zero, Label**
9. **bge $t5, $t3, Label**
   1. **slt $t6, $t5, $t3**
   2. **beq $t6, $zero, Label**
10. (8 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.
11. **if ((a > b) || (b > c)) {d = 1;}**
    1. **bgt $s0, $s1, loop**
    2. **ble $s1, $s2, next**
    3. **Loop: ori $s3, $zero, 1**
    4. **next**
12. **if ((a <= b) && (b > c)) {d = 1;}**
    1. **bgt $s0, $s1, next**
    2. **ble $s1, $s2, n ext**
    3. **ori $s3, $zero, 1**
    4. **next:**
13. (8 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.

1. (8 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 // $t0 = 0**

**addi $t1, $zero, 1 // $t1 = 1**

**loop: slt $t2, $a0, $t1 // if (number < i)**

**bne $t2, $zero, finish // jump to finish if (I > number)**

**add $t0, $t0, $t1 // $t0 = $t0 + $t1**

**addi $t1, $t1, 2 // $t1 = $t1 + 2**

**j loop // jump to loop in the beginning**

**finish: add $v0, $t0, $zero // output is total of $t0**

**outputs all the odd positive integers added together that are less than or equal to a number**

1. (12 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 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 //main counter if equal**

**skip: addi $t1, $t1, 4**

**bne $t1, $a1, inner**

**slt $t2, $t5, $v0**

**bne $t2, $zero, next**

**add $v0, $t5, $zero //changes biggest freq**

**add $v1, $t4, $zero // changes value for big**

**next: addi $t0, $t0, 4**

**bne $t0, $a1, outer**

**the code returns the most frequently use value as well as how many times it is used**

**value : $v1**

**frequency $v0**

* **Submission Requirements**
* Your solutions must be in a single file with a file name yourname-hw1.
* If scanned from hand-written copies, then the writing must be legible, or loss of credits may occur.
* Only submissions via the link on Canvas where this description is downloaded are graded. Submissions to any other locations on Canvas will be ignored.
* Late submissions are accepted for a maximum of 3 late days with 20% assignment credit off as late penalization. Assignments submitted after 3 late days will not be accepted.