# Homework Assignment #2 – MIPS Instructions

CDA 3100, Computer Organization I

⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯

**Problem 1 (10 points)** After the following instructions, what will be the value in $t0?

addi $t0, $0, 10 0 + 10 = 10 🡪 t0

ori $t0, $t0, 3 1010 OR 0011 = 1011 = 11 🡪 t0

$t0 = 11

**Problem 2 (10 points)** Assume the current values in $t0 and $t1 are 0 and 1, respectively. What will be the value in $t0 after the following instructions?

xori $t0, $t1, 7 0001 OR 0111 = 0110 = 6 🡪 t0

sll $t0, $t0, 2 6 \* 2^2 = 24 🡪 t0

t0 = 24

**Problem 3 (10 points)** Suppose a word array is 0,1,2,3,4,5,6,7,8,9 and its starting address is in $t0. After the following instructions, what will be the values in this array?

lw $t1, 8($t0) t1 = 2

sll $t1, $t1, 2 2 \* 2^2 = 8 🡪 t1

add $t0, $t0, $t1 0 + 8 = 8 🡪 t0

sw $t1, 8($t0) Starting at 2 in the array, we increment to 4, and replace it with t1, which is 8.

t1 = 8

t0 = 8

Answer: 0123856789

**Problem 4 (10 points)** Suppose $t0 and $t1 are holding 0 and 1, respectively. When the program enters this segment, right before the program executes the instruction at p4L2, what will be the value in $t0?

beq $t0, $t1, p4L1 t0 DNE t1; no branch.

p4L0:add $t0, $t0, $t1 0 + 1 = 1 🡪 t0

p4L1:bne $t0, $t1, p4L0 t0 equals t1; no branch.

p4L2:

t0 = 1

**Problem 5 (30 points)** The following problems deal with translating from C to MIPS. Assume that the variables f,g,h are assigned to registers $s0, $s1, $s2, respectively. Assume that the base address of the arrays A and B are registers $s6 and $s7, respectively.

(a) f=g+h+B[4];

(b) f=g-A[B[4]];

For the C statements above, what is the corresponding MIPS assembly code?

1. f=g+h+B[4];

g = s1, h = s2, B[4] = 16(s7)

lw $t0, 16($s7) #Stores B[4]

add $t1, $s1, $s2 #Stores g+h in t1

add $s0, $t0, $t1 #g+h+B[4]

1. f=g-A[B[4]];

lw $t0, 16($s7) #Stores B[4]

sll $t0, $t0, 2 #Creates the offset

lw $t1, $t0($s6) #Uses the t0 offset to fetch A[B[4]]

sub $s0, $s1, $t1 #g – A[B[4]]

**Problem 6 (30 points)** Write MIPS assembly code that implements the following: if $t0 is not equal to $t1 and if $t0 a multiple of 4, multiply $t0 by 3; otherwise, do nothing.

bne $t0, $t1, L1 #IF t0 != t1, then move to L1. This will be true.

j Exit; #Exits this segment of code if the first condition is untrue

L1:

andi $t2, $t0, 3 #t2 will equal 0 if it is a multiple of 4

beq $t2, $0, L2 #Checks if it is a multiple of 4

j Exit;

L2: sll $t0, $t0, 2 #Multiplies t0 by 4

sub $t0, $t0, $t0 #Subtracts t0 from t0 to achieve t0 \* 3

Exit: