UNIVERSITY OF
TEXAS
ARLINGTON

CSE 2312: Computer Organization & Assembly Language Programming Summer 2015

Homework #2

Student Name: SOLUTIONS

Student ID: _____

Directions: Answer the questions on the following pages. Show all applicable steps for any problems requiring the use of formulas or calculations. Submit your completed assignment electronically as a single PDF document with this completed coversheet as the first page and your name written at the top of all additional pages. You may also submit the document in person before the deadline, in which case this coversheet must be completed and stapled to your solution pages.

NOTE: Refer to the MIPS Reference Data Card for all questions on this assignment.

1. Consider the following C statement:

```
f = g + (h - 5);
```

a) Assuming *g* and *h* are stored in registers $t0 and $t1, and *f* is to be stored in register $s0, assemble the statement using a minimal number of MIPS instructions.

b) Give the machine representation (binary) of your answer to part a.

2. Assume the following register contents:

```
$t0 = 0XAAAAAAAA, $t1 = 0X12345678
```

a) What is the value of $t2 after the following statements are executed?

```
sll $t2, $t0, 4
or  $t2, $t2, $t1
```

b) What is the value of $t2 after the following statements are executed?

```
sll  $t2, $t0, 4
andi $t2, $t2, -1
```

c) What is the value of $t2 after the following statements are executed?

```
srl  $t2, $t0, 3
andi $t2, $t2, 0xFFEF
```

3. Translate the following MIPS code into C. Assume the variables f, ~~g, h, I, and j~~ are stored in registers $s0, ~~$s1, $s2, $s3, $s4,~~ respectively. Assume the base address of arrays A ~~and B~~ are in registers $s6 ~~and $s7, respectively.~~

```
addi $t0, $s6, 4
add  $t1, $s6, $0
sw   $t1, 0($t0)
lw   $t0, 0($t0)
add  $s0, $t1, $t0
```

4. Consider the following MIPS code:

```
LOOP:   slt  $t2, $0, $t1
        beq  $t2, $0, DONE
        subi $t1, $t1, 1
        addi $s2, $s2, 2
        j    LOOP
DONE:
```

a) Assume that register $t1 is initialized to the value 10 (in base 10), and $s2 is initialized to zero. What is the value of $s2 after executing the code segment above?

b) Convert the MIPS instructions above to C code. Assume that registers ~~$s1~~, $s2, $t1, and $t2 are integers ~~A~~, *B*, *i*, and ~~*temp*~~, respectively.

c) How many MIPS instructions would be executed if the register $t1 was initialized to the value *N*?

typo!

* 5. Translate the following C code to MIPS assembly using a minimal number of instructions. Assume that the values of *a*, *b*, ~~*f*~~, *i*, and j are in registers $s0, $s1, $t0, and $t1, respectively. Also assume that $s2 holds the base address of the array D.

```
for(i = 0; i < a; i++)
     for(j = 0; j < b; j++)
          D[4 * j] = i + j;
```

6. Suppose a specialized version of the MIPS processor, MIPS-A, contains an 8 channel analog-to-digital (a2d) converter for reading analog sensor values, and the "read analog to digital channel" instruction as defined below:

```
ra2dc $rd, channel
```

where `$rd` is the destination register for the sensor reading and `channel` is a constant in the range [0-7] specifying the desired a2d channel to read. Assume that the a2d converter returns unsigned, 16-bit sensor readings (i.e., the largest possible value in the 32-bit `$rd` register after the ra2dc instruction will be 65,535. Also assume that MIPS-A contains only the MIPS core instruction set (left column of the reference data card) and the ra2dc instruction.

a) Write a minimal set of MIPS-A instructions to read the first four a2d channels, compute a single average value, and store the average in register $s0.

① a)

```
addi  $t2, $t1, -5
add   $s0, $t0, $t2
```

b)

I Format :	opcode	rs	rt	immediate		
	8	10	9	-5		
R Format :	opcode	rs	rt	rd	shamt	funct
	0	8	10	16	0	32
	6 bits	5 bits	5 bits	5 bits	5 bits	6 bits
	001000	01010	01001	11111	11111	111011
	000000	01000	01010	10000	00000	100000

```
0010 0001 0100 1001 1111 1111 1111 1011
0000 0001 0000 1010 1000 0000 0010 0000
```

② a) $t2 = $t0 << 4

= 1010 1010 1010 1010 1010 1010 1010 1010 << 4

= 1010 1010 1010 1010 1010 1010 1010 0000

$t2 = $t2 OR $t1

= 1010 1010 1010 1010 1010 1010 1010 0000

OR 0001 0010 0011 0100 0101 0110 0111 1000

= 1011 1010 1011 1110 1111 1110 1111 1000

= 0x BABEFEF8

b) $t2 = 1010 1010 1010 1010 1010 1010 1010 0000

AND 1111 1111 1111 1111 1111 1111 1111 1111

= 1010 1010 1010 1010 1010 1010 1010 0000

= 0x AAAAAAA0

c) $t2 = $t0 >> 3

= 1010 1010 1010 1010 1010 1010 1010 1010

= 0001 0101 0101 0101 0101 0101 0101 0101

$t2 = $t2 AND 0xFFEF

= 0001 0101 0101 0101 0101 0101 0101 0101

AND 0000 0000 0000 0000 1111 1111 1110 1111

= 0000 0000 0000 0000 0101 0101 0100 0101

= 0x5545

③

```
addi $t0, $s6, 4   ⟶  $t0 = &A + 4
add  $t1, $s6, $zero ⟶ $t1 = &A
sw   $t1, 0($t0)   ⟶  A[1] = &A
lw   $t0, 0($t0)   ⟶  $t0 = &A
add  $s0, $t1, $t0 ⟶  s = &A + &A
```

⇒

```
A[1] = &A;
s = 2*&A;
```

④

```
$t2 = 1        $t2 = 1        $t2 = 1          $t2 = 1        $t2 = 1      $t2 = 0
$t1 = 9        $t1 = 8        $t1 = 7    ...   $t1 = 1        $t1 = 0      <DONE>
$s2 = 2        $s2 = 4        $s2 = 6          $s2 = 18       $s2 = 20
<jump>         <jump>         <jump>           <jump>         <jump>
```

a) $s2 = 20

b)

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

c) 5 instructions per loop iteration, N loop iterations ⟶ 5N instructions

(5) $s0 = a, $s1 = b, $t0 = i, $t1 = j

```
        addi  $t0, $zero, 0
        j     $TEST1
LOOP1:  addi  $t1, $zero, 0
        beq   $zero, $zero, TEST2
LOOP2:  add   $t3, $t0, $t1
        sll   $t2, $t1, 4
        add   $t2, $t2, $s2
        sw    $t3, 0($t2)
        addi  $t1, $t1, 1
TEST2:  slt   $t2, $t1, $s1
        bne   $t2, $zero, LOOP2
        addi  $t0, $t0, 1
TEST1:  slt   $t2, $t0, $s0
        bne   $t2, $zero, LOOP1
```

(6)

```
        razdc $t0, 0
        razdc $t1, 1
        add   $t0, $t0, $t1
        razdc $t1, 2
        add   $t0, $t0, $t1
        razdc $t1, 3
        add   $t0, $t0, $t1
        sll   $s0, $t0, 2
```