CDA3100 Midterm Exam, Summer 2013

Name : _____

Instructions:

1. *This is a close-book and close-notes exam.*
2. *You have **75** minutes to answer the questions.*
3. *Please write down your name on the top of this page first before you start answering any question. Answer all questions directly on the exam papers. If you need additional sheets, please let me know.*
4. *Partial credit is possible for an answer. However, please try to be concise and make your exam as neat as possible.*

Part I. Multiple choice questions.

Please select one answer for each question by circling the index of the answer. **8** points for each question. You may also write down a short sentence in the provided space to explain your choice. If your choice is wrong but your explanation is partially correct, partial credit will be given.

1. Given this 8-bit unsigned binary number, 1100 1001, what is the decimal representation?

   (a) 201_{10}
   (b) 55_{10}
   (c) -55_{10}
   (d) None of the above.

   128+64+8+1, a

2. What is the two's complement binary representation for -71_{10} in 8 bits?

   (a) 1011 1110
   (b) 1011 1001
   (c) 0100 0111
   (d) None of the above.

   b. 71-> 64+4+2+1->01000111->10111000->10111001

3. What is -25.875_{10} represented as a single precision floating point number?

   (a) 0 10000011 100 1101 0000 0000 0000 0000
   (b) 0 10000011 110 0111 1000 0000 0000 0000
   (c) 1 10000011 100 1111 0000 0000 0000 0000
   (d) None of the above.

   25.875->11001.111->1.1001111*2^4

   1 10000011 100 1111 0000 0000 0000 0000

   c

4. Suppose `$t0` is storing 10, `$t1` is storing 25. After the following instructions, what will be the value in `$t2`?

```
sub $t2, $t1, $t0
andi $t2, $t0, 3
xor $t2, $t2, $t0
```

   (a) 6
   (b) 9
   (c) 10
   (d) None of the above.

10&3 = 2

2^10 = 8

d

5. Suppose word array `A` stores `0,1,2,3,4,5,6,7,8,9`, in this order. Assume the starting address of `A` is in `$s0`. After the following instructions, what will be the values this array?

```
addi $t0, $s0, 20
lw $t1, 0($t0)
sw $t1, -8($t0)
```

(a) `0,1,2,5,4,5,6,7,8,9`
(b) `0,1,4,3,4,5,6,7,8,9`
(c) `0,1,2,3,4,5,4,7,8,9`
(d) None of the above.

a

6. Suppose `$t0` and `$t1` are holding `10` and `5`, respectively. After the following instructions, what will be in `$t0`?

```
      slt $t0, $t1, $t0
      bne $t0, $0, L1
      sll $t0, $t1, 2
      j L1
L1:   addi $t0, $t0, 1
```

(a) `1`
(b) `2`
(c) `21`
(d) None of the above.

b

7. Suppose word array `A` stores `0,1,2,3,4,5,6,7,8,9,` in this order. Assume the starting address of `A` is currently in `$s0`. After the following instructions, what will be the value in `$t1`?

```
      ori $t0, $0, 18
L0:   lw $t1, 0($s0)
```

```
        sll $t1, $t1, 2
        blt $t0, $t1, L1
        addi $s0, $s0, 8
        j L0
L1:
```

(a) 2
(b) 3
(c) 6
(d) None of the above.

d. should be 24

8. After the following instructions are executed (stop execution when program gets to the ellipsis), what will be the value in `$v0`?

```
        li $a0, 2
        jal fun
        add $v0, $v0, $a0
        jal fun
        add $v0, $v0, $a0
        jal fun
        add $v0, $v0, $a0
        jal fun
        add $v0, $v0, $a0
        addi $v0, $a0, 5

        …

fun:    addi $v0, $a0, 3
        addi $ra, $ra, 12
        jr $ra
```

(a) 7
(b) 10
(c) 19
(d) None of the above.

a

9. Which of the following statements about MIPS interrupt is true?

(a) An interrupt handler has to save `$ra`.
(b) An interrupt can be triggered only when executing certain instructions.
(c) An interrupt handler can save registers to the stack.
(d) None of the above.

d

10. Assuming fun1 is called from your main function, what happens during the execution of fun1?

```
      fun1:
            addi $sp, $sp, -4
            sw $ra, 0($sp)
            jal fun2
            sw $ra, 0($sp)
            addi $sp, $sp, 4
            jr $ra
      fun2:
            addi $sp, $sp, 4
            lw $ra, 0($sp)
      jal fun3
      lw $ra, 0($sp)
      addi $sp, $sp, -4
      jr $ra
fun3:
      jr $ra
```

(a) All functions execute to completion (reach a jr $ra instruction)
(b) The stack is properly maintained during execution
(c) An infinite loop is formed and the main function cannot be returned to
(d) None of the above.

c or d, typo so either can be correct as main cannot be returned to, but it's not an infinite loop. It loads an invalid value from the stack. This can be done in this specific instance, but it will load a non-address and thus can't jump to it leading to a crash.

Part II. Short answer questions

1. (**10** points) Please read the following code and write down the content in register $v1 after the program finishes execution.

```
      .data
A:    .word 0, 1, 2, 3, 4, 5, 6, 7, 8, 9


      .text
      .globl main
main:
      la $s0, A
      li $s1, 9
      li $s2, -1
L1:   addi $s2, $s2, 1
```

```
        bge $s2, $s1, exit
        sll $t0, $s2, 2
        add $t0, $t0, $s0
        lw $a0, 0($t0)
        jal funct
        blt $v0, $0, L1
exit:
        move $v1, $v0
        li $v0, 10
        syscall


funct:
        li $t0, 0
        li $v0, -1
L2:     addi $t0, $t0, 1
        bge $t0, $a0, exit2
        div $t1, $a0, $t0
        mul $t1, $t1, $t1
        bne $t1, $a0, L2
        move $v0, $a0
exit2:
        jr $ra
```


2. (**10** points) Write a **complete** function called "`F1`" that takes in `$a0` a number and returns in `$v0` a sum of the weights of the bits**.** Each bit will be weighted at index+1, e.g. index 2 will have a weight of 3 and index 0 will have a weight of 1.

   For example, if `$a0` is `57`, it should return `16` because `57` in binary is "0000 0000 0000 0000 0000 0000 0011 1001" going from bit 0 to bit 31, the weights will be 1, 4, 5, 6. Then 1+4+5+6 = 16.

Please note that

- You do not have to save `$a0` or any other registers in your function.
- **If more than 10 instructions are used for this function, 2 points will be deducted for each additional instruction until no more points are left**. Keep in mind that some pseudo instructions may result in more than one real instruction and the number of real instructions will be counted.

```
F1:
      li $v0, 0
      li $t0, 1
      li $t1, 1
L1:   and $t2, $a0, $t0
      beq $t2, $0, L2
      add $v0, $v0, $t1
L2:   sll $t0, $t0, 1
      addi $t1, $t1, 1
      bne $t0, $0, L1
Exit:
      jr $ra
```