**CDA3100 Midterm Exam, Spring 2011**

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. What is the binary representation for 105010 in 16 bits?

**(a) 0000 0100 0001 1010**

(b) 0000 1000 0001 0000

(c) 0000 1000 0000 0110

(d) None of the above.

The number 1050 can be expressed as:

1024 + 16 + 8 + 2

So, the answer is: 100 0001 1010 (a)

1. What is the two’s complement binary representation for -3910 in 8 bits?

(a) 1110 0111

**(b) 1101 1001**

(c) 1110 1001

(d) None of the above.

39 in binary is 00100111. After 2’s complement, -39 = 11011001 (b)

1. What is -31.12510 represented as a single precision floating point number?

**(a) 1 10000011 111 1001 0000 0000 0000 0000**

(b) 1 10000100 111 0001 0000 0000 0000 0000

(c) 0 10000100 111 0001 0000 0000 0000 0000

(d) None of the above.

 Sign bit: 1

 31.12510 = 11111.0012 = 1.1111001 x 24

 Exponent: 127+4 = 10000011.

Mantissa: 1111001

32 bit float: 1 10000011 111 1001 0000 0000 0000 0000 (a)

1. Suppose $t0 is storing 30, $t1 is storing 20. After the following instructions, what will be the value in $t2?

sub $t2, $t0, $t1

srl $t2, $t2, 2

 ori $t2, $t2, 10

(a) 8

**(b)10**

(c)18

(d) None of the above.

$t2=30-20 = 10.

After srl, t2=10/4 = 2.

In binary, 2 = 0010, 10 = 1010. 0010 | 1010 = 1010. So, t2=10 (b)

1. 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 value in $t0?

 addi $s0, $s0, 32

 lw $t0, 4($s0)

 andi $t0, $t0, 1

(a) 0

(b) 8

(c) 9

**(d) None of the above.**

After the addi, s0 points to the location of 8.

Lw loads 9 into t0. 9 in binary is 1001.

0001 & 1001 = 0001.

After andi, t0 should be 1.

1. Suppose $t0 and $t1 are holding 4 and 5, respectively. After the following instructions, what will be in $t1?

slt $t0, $t0, $t1

beq $t0, $0, p6L1

addi $t0, $t1, 1

 p6L1: addi $t0, $t0, 1

(a) 2

**(b) 7**

(c) 1

(d) None of the above.

 After slt, t0=1 (t0<t1). So, the branch condition is false. Both add instructions are executed. At the end of this code snippet, t0=5+1+1=7 (b)

1. 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 value in $t0?

ori $t0, $0, 0

p7L0: lw $t1, 0($s0)

add $t0, $t0, $t1

andi $t1, $t0, 16

bne $t1, $0, p7L1

 addi $s0, $s0, 4

 j p7L0

 p7L1:

**(a) 21**

(b) 15

(c) 45

(d) None of the above.

$t0 holds the sum of all the values encountered up to the current iteration.

The result of the andi will be non-zero for all values of t0>=16. The first time this is true is when t0=1+2+3+4+5+6=21. The program exits the loop at this point. (a)

1. Suppose function p8f1 is called. Which of the following statements is true?

p8f1: li $a0, 1

 jal p8f2

 add $v0, $v0, $a0

 jr $ra

p8f2: add $v0, $a0, $a0

 jr $ra

**(a) Function p8f1 can never return to the caller.**

(b) Function p8f1 returns to the caller. When it returns, $v0 is 2.

(c) Function p8f1 returns to the caller. When it returns, $v0 is 3.

(d) None of the above.

Function p8f1 does not store $ra anywhere. It is overwritten when function p8f2 is called. So, p8f1 cannot return to its caller. (a)

1. Suppose the following code is executed. Which of the following statements is true?

li.s $f0, -10.0

abs.s $f1, $f0

 add.s $f2, $f0, $f1

 mfc1 $t0, $f2

 beq $t0, $0, p9L1

(a) $f2 will store 0.

(b) The program will jump to p9L1 after the beq instruction.

**(c) Both of the above.**

(d) None of the above.

 f0 = -10.0, f1=10.0. So, f2=0. Value in f2 is stored in t0. So, t0=0. The program will jump to p9L1. (c )

1. Suppose a MIPS interrupt handler modifies $at but nothing else. Consider two code segments.

Segment 1:

add $t0, $t1, $t2

ori $t3, $t1, 100

Segment 2:

bgt $t0, $t1, p10L1

sub $t0, $t1, $t2

 Which of the following statements is true?

**(a) The interrupt handler will not cause problem if evoked during the execution of segment 1.**

(b) The interrupt handler will not cause problem if evoked during the execution of segment 2.

(c) Both of the above.

(d) None of the above.

The $at register is used to store intermediate values temporarily. It is used in a lot of pseudo instructions including the bgt. Bgt maps to

slt $at,$rt,$rs;

bne $at,$zero,Label

So, if an interrupt occurs during segment 2, the interrupt handler will modify $at and may cause a problem. So, (b) is false.

Segment 1 does not use $at. So, it is not affected by the interrupt handler. (a) is true.

**Part II. Short answer questions**

1. (**10** points) Please read the following code and write down the content in array A after funct returns.

 .data

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

 .text

 .globl main

main:

 la $a0, A

 li $a1, 9

 jal funct

 li $v0, 10 # exit

 syscall

funct:

 sll $t0, $a1, 2

 add $t0, $t0, $a0

funct\_L0:

 lw $t8, 0($a0)

 lw $t9, 0($t0)

 sw $t9, 0($a0)

 sw $t8, 0($t0)

 addi $a0, $a0, 4

 addi $t0, $t0, -4

 bgt $t0, $a0, funct\_L0

 jr $ra

9,8,7,6,5,4,3,2,1,0

This program reverses the array A, switching values at the ends and moving inward. When the 2 pointers (a0 and t0) cross each other, the function returns.

1. (**10** points) Write a **complete** function called “F1” that takes in $a0 the starting address of an array and adds every third of the elements in the array until the sum is a multiple of 8. It returns the sum in $v0. **It is guaranteed to you that at some point the sum will be a multiple of 8.**

For example, if the array is 1,2,3,4,5,6,11,8,9, it should add 1 with 4, then with 11, at which point it should return 16.

Please note that

* You do not have to save $a0 in your function.
* It is not recommended to use division to determine whether a number is a multiple of 8.
* **If more than 7 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 # intitialize $v0 to 0. $v0 holds the sum

Loop: lw $t1, 0($a0) # load the number from the array

 add $v0, $v0, $t1 # add current number to sum

 andi $t2, $v0,0x07 # extract the last 3 bits

 addi $a0,$a0,12 # move 3 array elements to the next

 # number required

bne $t2, $0, Loop # if sum was a multiple of 8, the last 3

# bits will be 0. If not, go back to Loop

# and add another number.

jr $ra # return

Any multiple of 8 will have 0’s in the last 3 bits.

8: 1000

16: 10000

24: 11000

32: 100000

40: 101000

48: 110000 …

We extract the last 3 bits by performing an And operation with 00000111 (7)

Then, if the last 3 bits were 0, the result of the And will be 0, and the number was a multiple of 8.