**UNCA CSCI 255**
**Exam 2 Fall 2014**
9 November, 2014

This is a closed book and closed notes exam. It is to be turned in by 3:00 PM.

Communication with anyone other than the instructor is not allowed during the exam. Furthermore, calculators, cell phones, and any other electronic or communication devices may not be used during this exam. Anyone needing a break during exams must leave their exam with the instructor. Cell phones or computers may not be used during breaks.

*If you want partial credit for imperfect answers, explain the reason for your answer!*

Name:________________________________

**Problem 1 (8 points) Decimal to two's complement conversion**
Convert the following four signed decimal numbers into eight-bit *two's complement* representation. Some of these numbers may be outside the range of representation for eight-bit two's complement numbers. Write "out-of-range" for those cases.

| | |
|---|---|
| 256 | -256 |
| -100 | 100 |

**Problem 2 (8 points) Two's complement to decimal conversion**
Convert the following four eight-bit *two's complement* numbers into signed decimal representation.

| | |
|---|---|
| 10011001 | 10000011 |
| 00011000 | 11111111 |

**Problem 3 (8 points) Adding signed numbers**

Add the following pairs of eight-bit *two's complement* numbers **and indicate which additions result in an overflow by writing one of "overflow" or "no overflow" in each box**. You must write either "overflow" or "no overflow" in each box in addition to the result of the addition.

| | |
|---|---|
| $\begin{array}{r} 01010011 \\ +\ \underline{11010011} \end{array}$ | $\begin{array}{r} 01101101 \\ +\ \underline{01101101} \end{array}$ |
| $\begin{array}{r} 10011011 \\ +\ \underline{10101011} \end{array}$ | $\begin{array}{r} 11010111 \\ +\ \underline{01101001} \end{array}$ |

**Problem 4 (6 points) Memories**

Consider a memory with 4 G words where each word is 8 bits.

How many bits are contained in this memory?

How many bits are required to address the words of this memory?

Remember to express your answers in the CSCI 255 way.

**Problem 5 (10 points) Truth table to Boolean expression**
Write a boolean expression that will implement the following truth table, where A, B, and C are inputs and X is the single output.

| A | B | C | X |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

**Problem 6 (10 points) Truth table *or* Boolean expression to circuit**
Draw a digital logic circuit that will implement the truth table in the preceding problem. You may also implement the circuit that implements your Boolean expression of that problem. Either method should give the same result.

**Problem 7 (15 points)**

Below is a section of six MIPS32 instructions written in MIP32 assembly language.

```
            slt     $t5,$t6,$zero
            beq     $t5,$zero,SkipIf
            nop
            sw      $zero,0($t7)
            addi    $t0,$t0,1
SkipIf:     addi    $t1,$t1,1
```

In the table below these six MIPS32 instructions are repeated. Write the 32 bits needed to encode each of the non-nop instructions in the MIPS32 instruction set.

```
            slt     $t5,$t6,$zero

_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
```

```
            beq     $t5,$zero,SkipIf

_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
```

```
            nop
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
            sw      $zero,0($t7)

_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
```

```
            addi    $t0,$t0,1

_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
```

```
SkipIf:     addi    $t1,$t1,-1

_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
```

**Problem 8 (15 points)**
If the style of the sixth homework, rewrite the following section of C code into C code that only uses two control structures:

```
goto label ;
if (expression) goto label ;
```

Also, do not use the `?:` operator of C and Java to simulate an `if`-`then`-`else`.

This specifically means that you can't use the `for`, `while`, `switch`, or even the statement block delimiters `{` and `}`. You can use the `if`, but if must be followed by a `goto` statement.

Here is the C program:

```
zeroCount = 0 ;
for (i=0; i<size; ++i) {
  if (V[i] == 0) {
    zeroCount = zeroCount + 1;
  }
}
```

**Problem 9 (20 points)**
Translate the following C subroutine into MIPS32 assembly langauge.

```
int zeroes(int V[0], int size) {
  int zeroCount = 0 ;
  for (i=0; i<size; ++i) {
    if (V[i] == 0) {
      zeroCount = zeroCount + 1;
    }
  }
  return zeroCount ;
}
```

***Here you go.*** By the way, be sure to return 0 when size is 0 or less.

```
              .globl  zeroes
zeroes:
```

```
              .end    zeroes
              .size   zeroes, .-zeroes
```