UNCA CSCI 255
Exam 2 Fall 2013
6 November, 2013

This is a closed book and closed notes exam. It is to be turned in by 11:15 AM. Calculators, PDA's, cell phones, and any other electronic or communication devices may not be used during this exam.
If you want partial credit for imperfect answers, explain the reason for your answer!

Name:_____

Problem 1 (4 points) Decimal to two's complement conversion
Convert the following two signed decimal numbers into six-bit two's complement representation. Some of these numbers may be outside the range of representation for six-bit two's complement numbers. Write "out-of-range" for those cases.

-28	**40**

Problem 2 (4 points) Two's complement to decimal conversion
Convert the following two five-bit two's complement numbers into signed decimal representation.

01010	**10101**

Problem 3 (6 points) Memories
A 1 GB memory has a 32-bit word size. How many words are contained in this memory?

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

Problem 4 (6 points) Adding
Add the following three pairs of six-bit two's complement numbers **and indicate which additions result in an overflow by writing either "overflow" or "no overflow" in each box**.

001100 + 011111	001111 + 011111
001110 + 110010	

Problem 5 (10 points) Boolean expression to truth table
Complete the truth table below so that it corresponds to the following Boolean equation

$Z = B + \overline{A}\, B\, \overline{C}$

If you prefer that your inversions be primes, you can think of the equation as

$Z = B + A'\, B\, C'$

Or, if you really like Java and C conditional expressions, you can go with

```
Z = B || !A && B && !C
```

A	B	C	Z
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Problem 6 (15 points) Truth table to digital logic
Draw a logic circuit at the gate level that will implement the following truth table, where A, B, and C are inputs and Z is the single output.

A	B	C	Z
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

Problem 7 (5 points) Truth table to Boolean expression
Write a Boolean expression that corresponds to the truth table shown in Problem 6. You can build on your Problem 6 answer if that seems appropriate.

Problem 8 (12 points)
In the tables below the values of several registers and memory locations are given.

Registers	
W0	0x00CA
W1	0x0A04
W2	0x0828
W3	0xBEEF
W4	0x0200

Memory	
0x0A00	0x00FF
0x0A02	0x0000
0x0A04	0x0200
0x0A06	0x0800
0x0A08	0x0A00

For the following X PIC instructions, list any registers or memory locations that are modified by the instruction along with the value stored into the register or location. Assume that each instruction is executed using the values shown above. That is, the instruction executions are *not* sequential. You can give the new value in either decimal or hexadecimal.

```
MOV   #0xFADE,W2
```

```
AND   #0x777,W3
```

```
SUB   W0,W0,W2
```

```
MOV   W3,[--W1]
```

```
MOV   [W1++],W2
```

```
MOV   [W1+#4],W0
```

Problem 9 (12 points)
In the left column there are six PIC24 instructions. In the right column write the 24 bits needed to encode each instruction in the PIC24 instruction set.

add W2,W3,W5	**_____**
add #0x23,W7	**_____**
add LATB,WREG	**_____**
add W2,#3,W5	**_____**
mov [W1],W3	**_____**
mov #-3,W3	**_____**

Setup for the remaining problems
In the remaining problem, similar to what you saw in lab, you are going to write short sections of PIC24 assembly code corresponding to C statements. In writing these code, assume that `x` and `z` are 16-bit *signed* integers, which could be declared in C as follows:

```
int   x ;
int   z ;
int   V[100] ;
```

In PIC24 assembly code, they would be declared using the `.space` directive:

```
x:   .space  2
z:   .space  2
V:   .space  200
```

In each of the programs a short section of C code will be given. You are to translate that code into a PIC24 code. You are free to use `WREG` and any of the sixteen PIC24 working registers in your answers. In some instances you will also need to use and define labels.

Just to make sure you don't make any beginner errors in the excitement of working these problems, keep in mind that the following are not legal PIC24 instructions and should **not** be used in your solutions.

```
mov   #7,x        ;; no literal with move to SFR
add   x,W0        ;; no add of SFR to working register
add   x,z         ;; no add of two SFR's
```

Often you will need to load `x` or `z` into a register before using them in a calculation.

Problem 10 (4 points)

```
x = V[z] ;
```

Problem 11 (4 points)

```
z = (x & 0xF) << 8 ;
```

Problem 12 (9 points)

```
if (x < z) {
    x = x + z ;
} else {
    z = x ;
}
```

Problem 13 (9 points)

```
x = 1 ;
while (x <= z) {
    x = 2*x ;
}
```