UNCA CSCI 255
Final Exam Fall 2012
11 December, 2012

This is a closed book and closed notes exam. It is to be turned in by 10:30 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 (6 points) Decimal to two's complement conversion
Convert the following six signed decimal numbers into seven-bit two's complement representation. Some of these numbers may be outside the range of representation for seven-bit two's complement numbers. Write "out-of-range" for those cases.

-63	**63**
-1	**1**
-40	**40**

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

1100011	**0011100**
1010101	**0000000**

Problem 3 (6 points) Adding
Add the following pairs of seven-bit two's complement numbers **and indicate which additions result in an overflow by writing either "overflow" or "no overflow" in each box**.

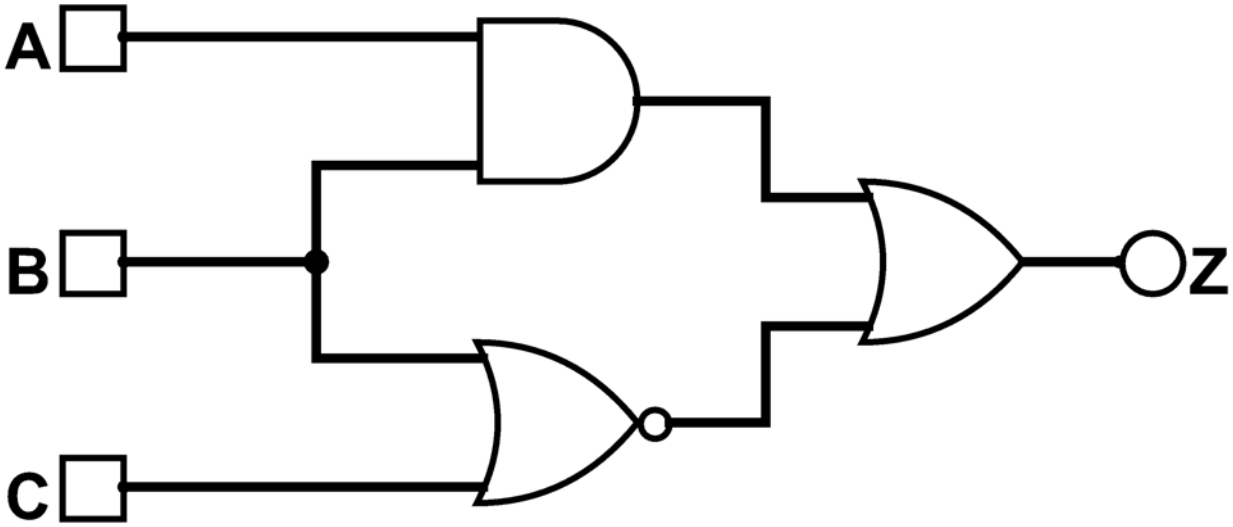
0001001 + 0000101	0110000 + 0010000
0110001 + 1100010	0011100 + 1100011
1001001 + 1000101	1110000 + 1010000

Problem 4 (4 points) Memories
A 16 MB memory has a 16-bit word size. How many words are contained in this memory?

A memory has 2 G words. Each word contains 16 bits. How many bytes are contained in this memory?

How many bits are required to address the 2 G words of this memory?

Problem 5 (8 points) Digital logic to truth table
A gate-level circuit is shown below with three inputs on the left and a single output on the right. Complete the truth table so that it corresponds to this digital logic circuit.



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 (2 points) Digital logic to Boolean expression
Write a Boolean expression that corresponds to the logic circuit shown in Problem 5. You can build on your Problem 5 answer if that seems appropriate.

Problem 7 (8 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	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Problem 8 (2 points) Truth table to Boolean expression
Write a Boolean expression that corresponds to the truth table shown in Problem 7. You can build on your Problem 7 answer if that seems appropriate.

Problem 9 (5 points) Boolean expression to truth table

Complete the truth table below so that it corresponds to the following Boolean equation

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

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

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

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

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

Problem 10 (5 points) Boolean expression to digital logic

On the remainder of this page, draw a logic circuit at the gate level that will implement the Boolean equation given in Problem 9. You can build on your Problem 9 answer if that seems appropriate.

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 11 (8 points) Instruction memory modes
In the tables below the values of several registers and memory locations are given.

Registers	
W0	0x00B6
W1	0x0004
W2	0x0802
W3	0x0806

Memory	
0x0800	0x00AA
0x0802	0x0007
0x0804	0x2012
0x0806	0x0800

For the following eight PIC24 instructions, list **all** registers or memory locations that are modified by the instruction along with the value, in either decimal or hexadecimal, stored into the *modified* register or location. Assume that each instruction is executed using the values shown above. That is, the instruction executions are *not* sequential.

`MOV  #17,W3`
`MOV  #17,[W3]`
`MOV  #17,[--W3]`
`INC  W0,W2`
`ADD  W0,#0x3,W2`
`ADD  W0,W3,W2`
`AND  W0,#0x3,W2`
`MOV  [W2+W1],W2`

Problem 12 (8 points) Machine coding
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,W6,W10	_ _
ADD W2,#6,W10	_ _
ADD #0x55,W3	_ _
ADD SR	_ _
MOV SR,WREG	_ _
MOV WREG,SR	_ _
MOV W10,SR	_ _
MOV #10,W10	_ _

Problem 13 (2 points) PWM
Consider the following sentence taken from the last CSCI 255 lab.

> The brightness of the LED depends on the duty cycle, not the pulse width.

What do the terms “duty cycle” and “pulse width” mean in this sentence? If you are wordy, you can write on the back; but a illustration might be the best answer.

In problems 14 to 16, spread over the next three pages, you are going to write short sections of PIC24 assembly code corresponding to C statements. In writing this code, assume that `i` and `j` are *signed* 16-bit integers, declared in C as follows:

```
int   i ;
int   j ;
```

In PIC24 assembly code, they could be allocated using the `.space` directive:

```
i:   .space  2
j:   .space  2
```

For 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,i        ;; no literal with move to SFR
add   i,W0        ;; no add of SFR to working register
add   i,j         ;; no add of two SFR's
```

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

Problem 14 (6 points) Arithmetic

```
while (i < 2012) {
    i = 3*i + 1 ;
}
```

Problem 15 (6 points) Tricky conditions

```
// This code works the same in both C and Java
if (i < j && ++i != 0) {
    --j ;
}
```

Problem 16 (6 points) Loopy code

```
// This code works the same in both C and Java
for (j=0; i<10; ++j) {
    i = i+j ;
}
```

Problem 17 (6 points) Blinky code
In a few labs this semester, you encountered the two following C `#define`'s.

```
#define REDLEDINPUTPIN        _TRISB12
#define REDLEDOUTPUTVALUE     _LATB12
```

and used a C routine called **`__delay_ms`** which can be called to pause the processor.

Using these two `#define`'s and the `__delay_ms` routine, complete the following `main` routine so that the red LED connected to pin B12 flashes on and off 10 times a second.

```
int main(void) {
```

Problem 18 (8 points) Homework 10 again
Use the following C declaration in this question.

```
typedef struct {
  int lo ;
  int hi ;
} range ;
```

Now write sections of C code to perform the following tasks:

- Declare the variable `pairRec` to be of type `range`.

- Declare the variable `pairPtr` to be a pointer to a variable of type `range`.

- Initialize `pairRec`.

- Point `pairPtr` to `pairRec`.

- Set the `lo` field of `pairRec` to 17.

- Write a function called `zeroRec` that is passed a pointer to a variable of type `range`. The function should set both fields of the referenced `range` type to 0.