UNCA CSCI 255
Exam 2 Fall 2009 Solution
20 November, 2009

Problem 1 (12 points) Adding
Add the following pairs of six-bit two's complement numbers **and indicate which additions result in an overflow**.

010011 + 000111 **011010** ***no overflow***	101100 + 101100 **011000** ***overflow***
110000 + 110000 **100000** ***no overflow***	001110 + 011000 **100110** ***overflow***

Problem 2 (13 points) Memory
A computer memory has 32-bit words stored in 8 M locations. What is the size of this memory in bits?

32 × 8 M or 256 M

How many address bits are needed to address the 8 M words of this memory?

23, as $2^{23} = 2^3 \times 2^{20} = 8M$

A computer memory has 32 M **bytes** stored in 16-bit words. How many words are stored in this memory?

32 M × 8 / 16 or 16 M

Problem 3 (12 points) Decoding an LC-3 program
In the table below is an LC-3 binary program that is loaded into six memory locations starting at x4000. Complete the rightmost column by decoding the LC-3 binary instructions into LC-3 assembly instructions.

	0100000000000000	.ORIG x4000
x4000	1110001000000101	**LEA R1,DatA**
x4001	0010010000000100	**LD R2,DatA**
x4002	0001001001000010	**LabB ADD R1,R1,R2**
x4003	0000100111111110	**BRn LabB**
x4004	0011001000000001	**ST R1,DatA**
x4005	1111000000100101	HALT ;TRAP x25
x4006	0100000000000100	**LabB .FILL x4004**

Problem 4 (13 points) Assignments Revisited

Write an LC-3 *subroutine* in assembler language that receives an argument in register `R1` and returns a value in register `R0`. If register `R1` contains a value greater than 255, a 1 should be returned. Otherwise, a 0 should be returned.

```
        .ORIG   x3000
        AND     R0,R0,#0
        ADD     R1,R1,#0    ;; As pointed out in one solution
        BRnz    NoAdd       ;; this is needed for x8000-x80ff
        LD      R2,M255
        ADD     R2,R2,R1
        BRnz    NoAdd
        ADD     R0,R0,#1
NoAdd   RET
M255    .FILL   #-255
        .END
```

Problem 5 (25 points) Hand assembled

Use the symbol table shown below in this question.

BUNCOMBE	x3315
HAYWOOD	x3375
MACON	x33D5
MADISON	x3435

Write the appropriate 16-bit LC-3 machine language word, in binary or hex, for each assembly language statement shown in the left column of the table below. Assume that the instruction is located at address x3311 in all cases. If the assembly language statement is illegal, state the reason why.

ADD	R0,R2,#16	**Illegal, 16 is too big for immediate value**
ADD	R7,R7,MACON	**Illegal, third operation can't be an address**
AND	R2,R3,#-3	**0101010011111101**
BR	BUNCOMBE	**0000111000000011**
BRz	HAYWOOD	**0000010001100011**
JSR	MADISON	**0100100100100011**
LDI	R7,MADISON	**Illegal, MADISON is out of range**
LDR	R2,R3,#20	**0110010011010100**
LEA	R2,R3,x20	**Illegal, too many operations for LEA**
ST	R3,R5	**Illegal, second operation can't be a register**
STI	R3,MACON	**1011011011000011**
STR	R3,R4,R5	**Illegal, third operation can't be a register**

Problem 6 (18 points)
Assume that the eight LC/3 registers have the values shown on the left below and that the eight words of memory starting at memory location x3220 have the values shown on the right.

Register	*Value*
R0	x0000
R1	x0000
R2	x2222
R3	x3333
R4	x0000
R5	x0000
R6	x6666
R7	x0000

Address	*Value*
x3220	x0000
x3221	x1111
x3222	x2121
x3223	x0000
x3224	x0000
x3225	x5151
x3226	x0000
x3227	x0000

For the nine addresses shown below, write a single LC/3 instruction to load the value **stored in** the specified memory location into register 1. (For example, when x3222 is specified, x2121 should be stored in R4.) Assume that each instruction is located at memory address x3200.

If this location cannot be loaded in one instruction, state why this is not possible.

x0000	**LDR R1,R0,#0**
x1111	**LDI R1,x20**
x1112	***out of range***
x2233	**LDR R1,R2,x11**
x3101	**LD R1,#-256**
x3200	**LD R1,#-1**
x3201	**LD R1,#0**
x3301	***out of range***
x3344	**LDR R1,R3,x11**
x6646	**LDR R1,R6,#-32**
x6666	**LDR R1,R6,#0**
x6686	***out of range***