Quiz 2 **Solution** CSCI 255 Spring 2001

9 April, 2001

Problem 1 (64 points): 3:15-3:47

In this problem you are asked to write **eight** *independent* sections of LC-2 assembly code to set registers `R0` or `R1` or LC-2 memory locations based on constants, the present values of R3 and R4, or LC-2 memory locations. You may use registers `R6` or `R7` as "scratch" registers but should not modify any other registers. You must assume that your code will be located somewhere between memory locations `x3000` and `x30FF`. You may use `.fill`'s when needed to initial memory locations. You should assume that these `.fill`'s would also be stored in memory locations `x3000` to `x30FF`.

In these subproblems, the code to implement is given in the psuedo-C notation used in class lectures. `R`*n* will be used as a reference to LC-2 register *n*. `M[`*n*`]` will be used as a reference to LC-2 memory location *n*.

There are many possible right answers. These are probably the shortest.

`R0 ← 5 * R3 ;`		**ADD**	**R0,R3,R3**
		ADD	**R0,R0,R0**
		ADD	**R0,R0,R3**
`R0 ← R3 – R4 ;`		**NOT**	**R0,R4**
		ADD	**R0,R0,#1**
		ADD	**R0,R0,R3**
`R0 ← R3 & R4 ;`		**AND**	**R0,R3,R4**

`if (R3 == 15)`
`  R0 ← R4 ;`
`else`
`  R0 ← R4 + 1 ;`
		ADD	**R0,R4,#0**

		ADD	**R6,R3,#-15**
		BRz	**DONE**
		ADD	**R0,R0,#1**
	DONE	**…**	

`R0 ← R4 ;`
`while (R0 < 107)`
`  R0 ← R0 + R0 ;`
		LD	**6,M107**

		ADD	**R0,R4,#0**
		BR	**MDLOOP**
	BGLOOP	**ADD**	**R0,R0,R0**
	MDLOOP	**ADD**	**R7,R6,R0**
		BRn	**BGLOOP**
		…	
	M107	**.FILL**	**#-107**

M[x3100] ← M[x3100] + 5 ;			
M[x4100] ← M[x4100] + 5 ;		LDI	R6,PTR
		ADD	R6,R6,#5
		STI	R6,PTR
		…	
	PTR	.FILL	x3010
R0 ← R3 + 1 ; R1 ← R4 + '1' ;		ADD	R0,R3,#1
		LD	R6,ASC1
		ADD	R1,R4,R6
		…	
	ASC1	.FILL	X31

Problem 2 (16 points): 3:47-3:55

Translate into LC-2 machine language (binary) program the LC-2 assembly language program shown below:

A fairly common problem was starting the first instruction (LD) at x3001 rather than x3000.

	.ORIG	x3000	
	LD	R1,MX	
	LDI	R2,MX	
	LEA	R3,MX	
	LDR	R4,R1,#1	0110100001000001
	HALT		1111000000100101
MX	.FILL	0x3006	0011000000000110
MY	.FILL	0x3007	0011000000000111
MZ	.FILL	0x3008	0011000000001000
	.END		

Problem 3 (12 points): 3:55-4:01

What are the values of registers R1 to R4 after the LC-2 assembly language program in Problem 2 is executed?

R1 = **x3006**
R2 = **x3007**
R3 = **x3005**
R4 = **x3008**

If you (incorrectly) assumed that MX was located at x3006, then R1, R2, and R3 would be set to x3006 and R4 would be set to x3007.

Quiz 5 CSCI 255 Spring 2002 **Solution**

1 April, 2002

Problem 1 (72 points)

Write four separate LC/2 assembly programs to compute the following four C statements.

```
R4 = 4*R3 - 16 ;

            ADD         R4,R3,R3
            ADD         R4,R4,R4
            ADD         R4,R4,#-16
```

Grading

+17	Correct but more than 3 memory accesses
+16	Correct but more than 5 memory accesses
+15	Correct but more than 7 memory accesses
-5	Modifies register other than R3
-5	Simple logic error (like doubling too few or too many times)

```
R2 = R6 - R5 - 1 ;

            NOT         R2,R5       ; Remember ~R5 is -R5-1
            ADD         R2,R6,R2
```

Grading

+17	Correct but more than 2 memory accesses
+16	Correct but more than 4 memory accesses
+15	Correct but more than 6 memory accesses
-5	Modifies register other than R2
-5	Simple logic error (not adding in 1 for negation, adding –1 then inverting)

```
if (R2 < 0)
     R3 = R7 + 150 ;
else
     R4 = R7 + 150 ;

            ST          R0,SAVER0
            LD          R0,C150
            ADD         R2,R2,#0
            BRn         STIN4
            ADD         R3,R0,R7
            BRnzp       RST0
STIN4       ADD         R4,R0,R7
RST0        LD          R0,SAVER0
            .....
C150        .FILL       #150
SAVER0      .BLKW       0
```

Grading

+17	Correct but more than 11 memory accesses
+16	Correct but more than 14 memory accesses
+15	Correct but more than 17 memory accesses
+17	A lot of duplicate code for R7 + 150
-5	Modifies register other than R3 (if R2<0) or R4 (if R2≥0)
-6	One control structure problem (no BR to common end point, misdirected branch)
-9	Multiple control structure problems
-5	Simple logic error (adding in 150 without first loading)