Problem Set # 5
Due: 03 December, 2003 in class

Instructions:
The homework you turn in should be your own work. You may discuss problems with your colleagues. But though you may share ideas and possible approaches to solving problems, you may not share actual solutions.

Do not forget to write your TA's name and your discussion session time on the homework you turn in.

Please staple your homework.

Note: Numbers in parentheses are problem numbers from text.

1. (7.6)

Our assembler has crashed and we need your help! Create a symbol table and assemble the instructions at labels D, E, and F for the program below. You may assume another module deposits a positive value into A before this module executes.

```
.ORIG   x3000
AND    R0, R0, #0
D      LD    R1, A
        AND    R2, R1, #1
        BRp    B
E      ADD   R1, R1, #-1
B      ADD   R0, R0, R1
        ADD   R1, R1, #-2
F      BRp    B
        ST    R0, C
        TRAP   x25
A      .BLKW  x1
C      .BLKW  x1
.END
```

In no more than 15 words, what does the above program do?
2. (7.10)
The following program has an error in it. Identify the error and explain how to fix it.

```
ADD R3, R3, #30
ST R3, A
HALT
A .FILL #0
```

Will this error be detected when this code is assembled or when this code is run on the LC-3?

3. (7.24)
We want the following program to shift R3 to the left by four bits, but it has an error in it. Identify the error and explain how to fix it.

```
.ORIG x3000
AND R2, R2, #0
AND R2, R2, #4
LOOP BRz DONE
ADD R2, R2, #-1
ADD R3, R3, R3
BR LOOP
DONE HALT
.END
```

4. (8.12)
Adam H. decided to design a variant of the LC-3 that did not need a keyboard status register. Instead, he created a readable/writable keyboard data and status register (KBDSR), which contains the same data as the KBDR. With the KBDSR, a program requiring keyboard input would wait until a nonzero value appeared in the KBDSR. The nonzero value would be ASCII value of the last key press. Then the program would write a zero into the KBDSR indicating that it had read the key press. Modify the basic input service of section 8.2.2 to implements Adam’s scheme.

5. (9.2)
a. How many trap service routines can be implemented in the LC-3? Why?
b. Why must a RET instruction be used to return from a TRAP routine? Why won’t a BR (Unconditional Branch) instruction work instead
c. How many accesses to memory are made during the processing of a TRAP instruction? Assume the TRAP is already in the IR.

6. (9.10)
The starting address of the trap routine is stored at the address specified in the TRAP instruction. Why isn’t the first instruction of the trap routine stored at that address instead? Assume each trap service routine requires at most 16 instructions. Modify the semantics of the LC-3 TRAP instruction so that the trap vector provides the starting address of the service routine.

7. (9.14)
The following service routine stops the computer by clearing the RUN latch, bit [15] of the machine control register. The latch is cleared by the instruction in line 14, and the computer stops. What purpose is served by the instructions on lines 19 through 1C?

```
01 .ORIG xFD70 ;Where this routine resides
02 ST R7, SaveR7
03 ST R1, SaveR1 ;R1: a temp for MC register
04 ST R0, SaveR0 ;R0: used as working space
05
06 ; print message that machine is halting
07
08 LD R0, NewLn
09 TRAP x21
0A LEA R0, Message
0B TRAP x22
0C LD R0, NewLn
0D TRAP x21
0E ;
0F ; clear bit 15 at xFFFE to stop the machine
10 ;
11 LDI R1, MCR ; Load MC register into R1
12 LD R0, MASK ; R0 = x7FFF
13 AND R0, R1, R0 ; Mask to clear the top bit
14 STI R0, MCR ; Store R0 into MC register
15 ;
16 ; return from HALT routine.
17 ; (how can this routine return if the machine is halted above?)
```
18 ;
19     LD    R1, SaveR1 ; Restore registers
1A     LD    R0, SaveR0
1B     LD    R7, SaveR7
1C     RET
1D ;
1E ; Some constants
1F ;
20 NewLn   .FILL  x000A
21 SaveR0  .BLKW  1
22 SaveR1  .BLKW  1
23 SaveR7  .BLKW  1
24 Message .STRINGZ "Halting the machine."
25 MCR     .FILL  xFFFE ; Address of MCR
26 MASK    .FILL  x7FFF ; Mask to clear the top bit
27 .END

8.
Can a subroutine call another subroutine? If so, is there anything special the calling subroutine must do? Explain your answer.