CS/ECE 252: INTRODUCTION TO COMPUTER ENGINEERING

UNIVERSITY OF WISCONSIN—MADISON

Professor Karthikeyan Sankaralingam
TAs: Suriyha Balaram Sankari, Newsha Ardalani, Rebecca Lam, Kamlesh Prakash, and Yinggang Huang

Examination 4
In Class (50 minutes)
Friday, May 11, 2012
Weight: 17.5%

NO: BOOK(S), NOTE(S), CALCULATORS OF ANY SORT.

This exam has 11 pages, including a blank page at the end. Plan your time carefully, since some problems are longer than others. You must turn in pages 1 to 11.

LAST NAME: _____

FIRST NAME:_____

SECTION: _____

CAMPUS ID# _____

EMAIL ID _____

Question	Maximum Points	Points
1	6	
2	4	
3	5	
4	4	
5	2	
6	5	
7	4	
Total	**30**	

Problem 1 (6 points)
An LC-3 assembly language program is given below.

```
        .ORIG x3000
        LEA    R0, STR1
        LEA    R1, STR2
LOOP    LDR    R2, R0, #0
        BRz    DIFF
        LDR    R3, R1, #0
        BRz    DIFF
        ADD    R0, R0, #1
        ADD    R1, R1, #1
        BRnzp  LOOP
DIFF    LDR    R2, R0, #0
        LDR    R3, R1, #0
        NOT    R3, R3
        ADD    R3, R3, #1
        ADD    R3, R2, R3
        BRp    OUT1
        BRn    OUT2
        BRz    OUT3
OUT1    LEA    R0, STR1
        BRnzp  DONE
OUT2    LEA    R0, STR2
        BRnzp  DONE
OUT3    LEA    R0, STR3
DONE    PUTS
        HALT

STR1    .STRINGZ "I close my eyes only for a moment and the moment's gone"
STR2    .STRINGZ "There's a lady who's sure all that glitters is gold"
STR3    .STRINGZ "Unsealed on a porch a letter sat"
```

Note: there are 57 characters in the first string (including null)

a. Fill in the symbol table created by the first pass of the assembler on the above program for the following labels / symbols (3 points)

Label	**Address**
LOOP	**0x3002**
DIFF	**0x3009**
OUT1	**0x3011**
OUT2	**0x3013**
STR1	**0x3018**
STR2	**0x3050**

b.

c. Assume we don't know anything about the strings stored in STR1, STR2, and STR3. What does the above program do? Give your answer in **one** or **two** sentences. (1 point)

It will compare STR1 and STR2 and output the longer string. Else it will display STR3.

d. What is the output of this program? (2 points)

"I close my eyes only for a moment and the moment's gone"

Problem 2 (4 points) 1 point each

Answer the following short answer questions in **one** or **two** sentences.

a. What are the two methods for accessing I/O devices?

Memory mapped and special I/O instructions

b. Define and contrast callee-save and caller-save.

Callee-save: the called subroutine will save any registers it will use during its execution

Caller-save: the caller of the subroutine will save any registers the subroutine will use before calling it.

The difference is when the registers are saved: before or within the subroutine call.

c. Give one example of a syntax error using the AND **immediate** instruction and explain the error.

```
AND   R0, R0, #50     : Immediate value out of range
AND   R0, R8, #0      : Invalid register
AND   R0, R0, Label   : Invalid input parameter
```

d. Is there a problem with using STR as a label in an LC-3 assembly program? Why or why not?

Yes, STR is an instruction and cannot be used as a label.

Problem 3 (5 points)
The following LC-3 subroutine implements the PUTS service routine. This means it will output the string stored at the address stored in R0 and then return to normal execution.

```
        .ORIG x0466
        ST     R1, R1_TMP
        ST     R2, R2_TMP
L1      LDI    R2, DSR
        BRzp   L1
        LDR    R1, R0, #0
        BRz    DONE
        STI    R1, DDR
        ADD    R0, R0, #1
        BRnzp  L1
DONE    LD     R2, R2_TMP
        LD     R1, R1_TMP
        RET

DSR     .FILL     xFE04    ; Address of DSR
DDR     .FILL     xFE06    ; Address of DDR
R1_TMP     .FILL 0
R2_TMP     .FILL 0
```

a. Fill in the blanks. **There should be one instruction per line and no NOPs.** (3 points)

b. Assume the above assembly code is a service routine that is a part of the LC-3 OS. Suppose we can call it using TRAP x56. Where is the address of the trap vector, and what are its contents? (2 points)

Give all answers in hex

Address of TRAP vector	**Contents at this memory location**
0x56	**0x0466**

Problem 4 (4 points)

The following program reads two inputs from memory locations A and B and compares their contents. If M[A] < M[B], the program terminates with the value 1 in memory location COMPARE_RESULT, otherwise it stores zero in COMPARE_RESULT and terminates. What is the problem with the following code? How can you resolve it? Answer in **one** or **two** sentences in the box provided.

```
                .ORIG       x3000
                LD          R0, A
                LD          R1, B
                JSR         COMPARE
                HALT

COMPARE         ST          R7, R7_TMP
                JSR         NEG          ; compute -R0
                ADD         R0, R0, R1   ; compute R1-R0
                BRP         STORE_ONE
STORE_ZERO      LD          R0, ZERO
                ST          R0, COMPARE_RESULT
                BR          END_COMP
STORE_ONE       LD          R0, ONE
                ST          R0, COMPARE_RESULT
END_COMP        LD          R7, R7_TMP
                RET

NEG             ST          R7, R7_TMP
                NOT         R0, R0
                ADD         R0, R0, #1
                LD          R7, R7_TMP
                RET

A               .BLKW       1
B               .BLKW       1
R7_TMP          .BLKW       1
COMPARE_RESULT  .BLKW       1

ZERO            .FILL       0
ONE             .FILL       1
                .END
```

ANSWER:

Problem: R7_TMP is used for storing the return address of two subroutines that one of them is called inside the other. This causes the return address for COMPARE subroutine that is stored in R7_TMP to be overwritten when the NEG subroutine is being called.

Solution: Store R7 for COMPARE(NEG) subroutine in R7_TMP but for the NEG(COMPARE) subroutine in a memory location other than R7_TMP.

Problem 5 (2 points)

A programmer decides to design a variant of the LC-3 that does not need a keyboard status register. Instead, he creates a readable/writable keyboard data and status register (KBDSR). With the KBDSR, a program requiring keyboard input waits until a nonzero value appears in the KBDSR and reads this value into R0. The nonzero value is the ASCII value of the last key pressed. Then the program writes a zero into the KBDSR indicating that it has read the data. **Fill in the following blanks to implement this new scheme. (1 instruction per line)**

```
ZERO        LDI     R0, C
            BRz     ZERO

NONZERO     AND     Ri, Ri, #0
            STI     Ri, C

            BRnzp   NEXT_TASK

C           .FILL   xFE00     ; Address of KBDSR
```

Ri could be any register other than R0

Problem 6 (5 points)

Select **only one answer** choice for the following questions about LC-3:

1. Suppose `JSRR R5` is stored at memory location x400F and R5 contains x3009. The execution of JSRR instruction will cause:
    a. R7 to be loaded with x300A and the PC to be loaded with x400F
    b. R7 to be loaded with x4010 and the PC to be loaded with x300A
    c. ***R7 to be loaded with x4010 and the PC to be loaded with x3009***
    d. R7 to be loaded with x3010 and the PC to be loaded with x400F
2. Which of the following is **not true** about interrupt driven I/O?
    a. ***The processor must routinely check the status register for the device until new data arrives or the device is ready***
    b. The device controls the interaction by sending a special signal to the processor when it is ready
    c. It has built in priority levels for different device requests
    d. It is more efficient than polling
4. At the end of which phase of instruction cycle does an interrupt-driven I/O processor test for an interrupt signal?
    a. ***Store Result***
    b. Execute
    c. Decode
    d. Fetch
5. In LC-3, in what range of addresses are the trap vectors stored?
    a. xFDFF - xFFFF
    b. xFE00- xFFFF
    c. xFF00 - xFFFF
    d. ***x0000 - x00FF***
5. Which one is the highest priority level?
    a. ***PL7***
    b. PL0
    c. PL1
    d. They are all equal in priority

Problem 7 (4 points)

Suppose the LC-3 is modified such that the TRAP vector table exists in the range of addresses from 0x0000 to 0x0007. The following tables show the TRAP vector table and the relevant system service

routines.

TRAP vector table:

Address	Memory Content
0x0000	0x000A
0x0001	0x000F
0x0002	0x000E
0x0003	0x000A
0x0004	0x000B
0x0005	0x000D
0x0006	0x000F
0x0007	0x000C

Service routines:

Address	Label	Instruction
0x000A		ADD R0, R0, #10
0x000B		ST R0, L2
0x000C		RET
0x000D	L1	LD R0, L3
0x000E		BRn L1
0x000F		RET
0x0010	L2	.FILL x100A
0x0011	L3	.FILL x100B

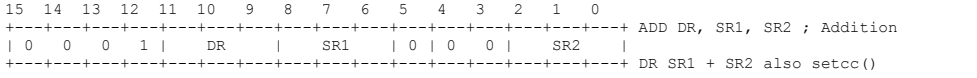
Suppose we are running a program and we reach to TRAP x0 instruction at memory location xABCD. Show the MAR trace for a TRAP x0 call. **Give all the answers in HEX.**
You may not need to use all the blank boxes.

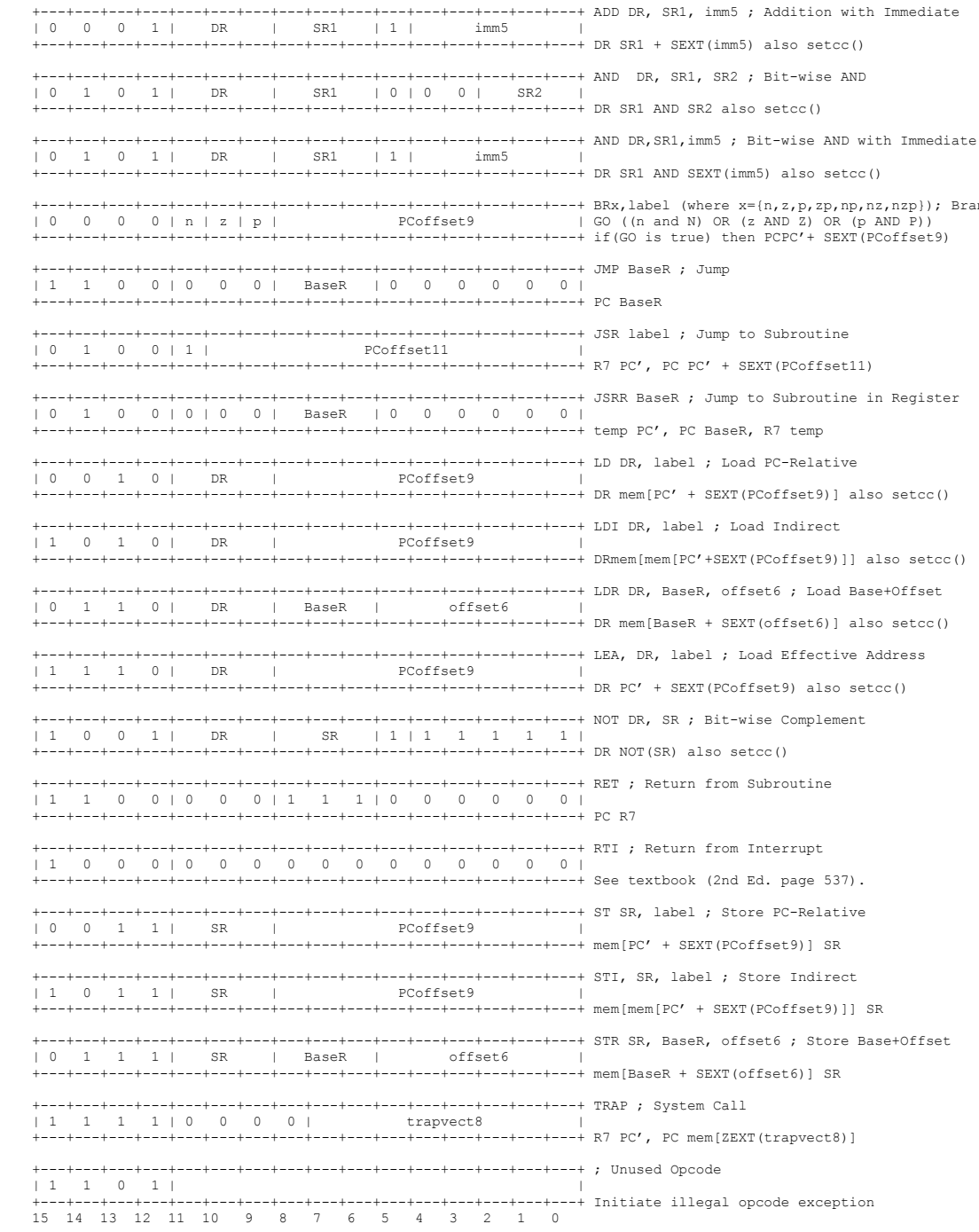
MAR Trace
xABCD
x0000
x000A
x000B
x0010
x000C

Scratch Page

LC-3 Instruction Set (Entered by Mark D. Hill on 03/14/2007; last update 03/15/2007)

PC': incremented PC. setcc(): set condition codes N, Z, and P. mem[A]:memory contents at address A.
SEXT(immediate): sign-extend immediate to 16 bits. ZEXT(immediate): zero-extend immediate to 16 bits.




ADD DR, SR1, imm5 ; Addition with Immediate
DR SR1 + SEXT(imm5) also setcc()
AND DR, SR1, SR2 ; Bit-wise AND
DR SR1 AND SR2 also setcc()
AND DR,SR1,imm5 ; Bit-wise AND with Immediate
DR SR1 AND SEXT(imm5) also setcc()
BRx,label (where x={n,z,p,zp,np,nz,nzp}); Branch
GO ((n and N) OR (z AND Z) OR (p AND P))
if(GO is true) then PCPC'+ SEXT(PCoffset9)
JMP BaseR ; Jump
PC BaseR
JSR label ; Jump to Subroutine
R7 PC', PC PC' + SEXT(PCoffset11)
JSRR BaseR ; Jump to Subroutine in Register
temp PC', PC BaseR, R7 temp
LD DR, label ; Load PC-Relative
DR mem[PC' + SEXT(PCoffset9)] also setcc()
LDI DR, label ; Load Indirect
DRmem[mem[PC'+SEXT(PCoffset9)]] also setcc()
LDR DR, BaseR, offset6 ; Load Base+Offset
DR mem[BaseR + SEXT(offset6)] also setcc()
LEA, DR, label ; Load Effective Address
DR PC' + SEXT(PCoffset9) also setcc()
NOT DR, SR ; Bit-wise Complement
DR NOT(SR) also setcc()
RET ; Return from Subroutine
PC R7
RTI ; Return from Interrupt
See textbook (2nd Ed. page 537).
ST SR, label ; Store PC-Relative
mem[PC' + SEXT(PCoffset9)] SR
STI, SR, label ; Store Indirect
mem[mem[PC' + SEXT(PCoffset9)]] SR
STR SR, BaseR, offset6 ; Store Base+Offset
mem[BaseR + SEXT(offset6)] SR
TRAP ; System Call
R7 PC', PC mem[ZEXT(trapvect8)]
; Unused Opcode
Initiate illegal opcode exception
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0