ECE220: Computer Systems and Programming

Fall 2015 - Midterm Exam 1

September 21, 2015

- This is a closed book exam except for 1 sheet of notes
- You may not use a calculator
- Absolutely no interaction between students is allowed
- Manage your time properly to get the most points
- This booklet must be handed in at the end of the exam

Name:_____

NetID:_____

Room:_____

Problem 1 (30 points): Count One

Consider an integer *n* stored in memory location x5000, write a program in LC-3 Assembly to count the number of 1s in its binary representation and store the result back into memory location x5000.

Example input (in R6)	Binary representation (16 bits)	Example output (in memory X5000)
0x0009	0000 0000 0000 1001	2
0x00FF	0000 0000 1111 1111	8
0x000A	0000 0000 0000 1010	2

Implement the **Brian Kernighan's Algorithm** as follows**:**

1 Initialize *count* = 0
2 **If** integer *n* is not zero
(a) Do bitwise & with (*n*-1) and assign the value back to *n*. That is, *n* = *n* & (*n*-1)
(b) Increment count by 1
(c) Go to step 2
3. **Else** return *count*

The code you need to finish is marked by "TODO" in the file **count_bit1.asm**. We have provided you a simple input test file input.asm. Use the following procedures to test your program:

```
~$ lc3as input.asm
~$ lc3as count_bit1.asm
~$ lc3sim
```

In lc3sim console, load the **input1.obj** to initialize the memory first and then load **count_bit1.obj** to run your program:

```
file input.obj
file count_bit1.obj
finish
```

The final result should be stored in memory location x5000.

Problem 2 (40 points): Fibonacci Sequence

For this problem you will write a subroutine in the file **fibonacci.asm** that calculates F_k, the kth Fibonacci number. The subroutine should take input k from R1, and save F_k in R6.

Fibonacci numbers are numbers in the integer sequence 1,1,2,3,5,8,13,… By definition, $F_1 = 1$ and $F_2 = 1$. Each subsequent Fibonacci number is equal to the sum of the two previous numbers.

$$F_i = \begin{cases} 1 & , i = 1 \\ 1 & , i = 2 \\ F_{i-1} + F_{i-2} & , i \geq 3 \end{cases}$$

A table of relevant Fibonacci numbers appears at the end of the problem.

INPUTS

K : A positive integer number stored in **R1**. Your program does not need to handle invalid inputs.

OUTPUTS

$\boldsymbol{F_k}$: The kth Fibonacci number. **The output should be saved in R6**. If F_k is too large to be represented with 16 bit two's complement, the program should output **–1**.

Algorithm

An algorithm to calculate F_k iteratively is shown below in pseudocode. You are **not** required to implement your subroutine this way. Please note that this algorithm is not recursive (like the one discussed in class). It builds the Fibonacci sequence from the lower values up to the higher values.

```
If k<= 2
        F_k = 1
If k => 3
        F_low = 1
        F_high = 1
        For i = 3 to k:
                F_sum = F_low + F_high
                F_low = F_high
                F_high = F_sum
        F_k = F_sum
```

The first 26 Fibonacci numbers:
1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946, 17711, 28657, 46368

Problem 3 (15 points): ASCII Picture Printer

An ASCII picture consists of a number of ASCII characters arranged in a grid, N characters wide by M characters tall (where N an M are integers). N is the number of columns and M the number of rows. Write code to print an ASCII picture, which is stored starting at memory address x5000.

The value at x5000 is the value of N and the number at x5001 is the value of M. Starting at x5002 are ASCII characters. The first N characters are the first row of the image, the next N characters are the second row of the image, and so on.

Your code should print out each character. After printing a row, your code should print a newline. Continue until all M rows have been printed.

Details:

- Write your code in LC-3 Assembly in **ASCIIart.asm**
- You may use any TRAPS you find helpful
- Be sure to print only one newline per row, and no additional spaces
- You may assume that the picture has less than 256 characters. If M or N is less than or equal to 0, print no ASCII characters
- Make sure your code assembles. If your code does not assemble, you may get a 0 for this question

Testing:

Two input files are provided, input1.asm and input2.asm

Assemble your code using the command

~$ lc3as input1.asm
~$ lc3as input2.asm
~$ lc3as *ASCIIart.asm*

You may test the code using the following commands

~$ lc3sim
file input1.obj
file ASCIIart.obj
finish

Example output can be found in input1_out.txt and input2_out.txt

Problem 4 (15 points total): C Concepts

In the file **problem4.c**, record your answer to the following questions.

1. Consider the C variable declarations below

```
char xc[2];
int    x;
long long xl;
```

How many LC-3 16-bit memory locations should be allocated to hold all the variables implied by these statements? Choose one of the answers below. (5 points)

a) 5
b) 6
c) 7
d) 8
Answer: c). * No points for putting down only 7.

2. True or False (2.5 points each)

Ascribe T (True) or F (False) to both statements below

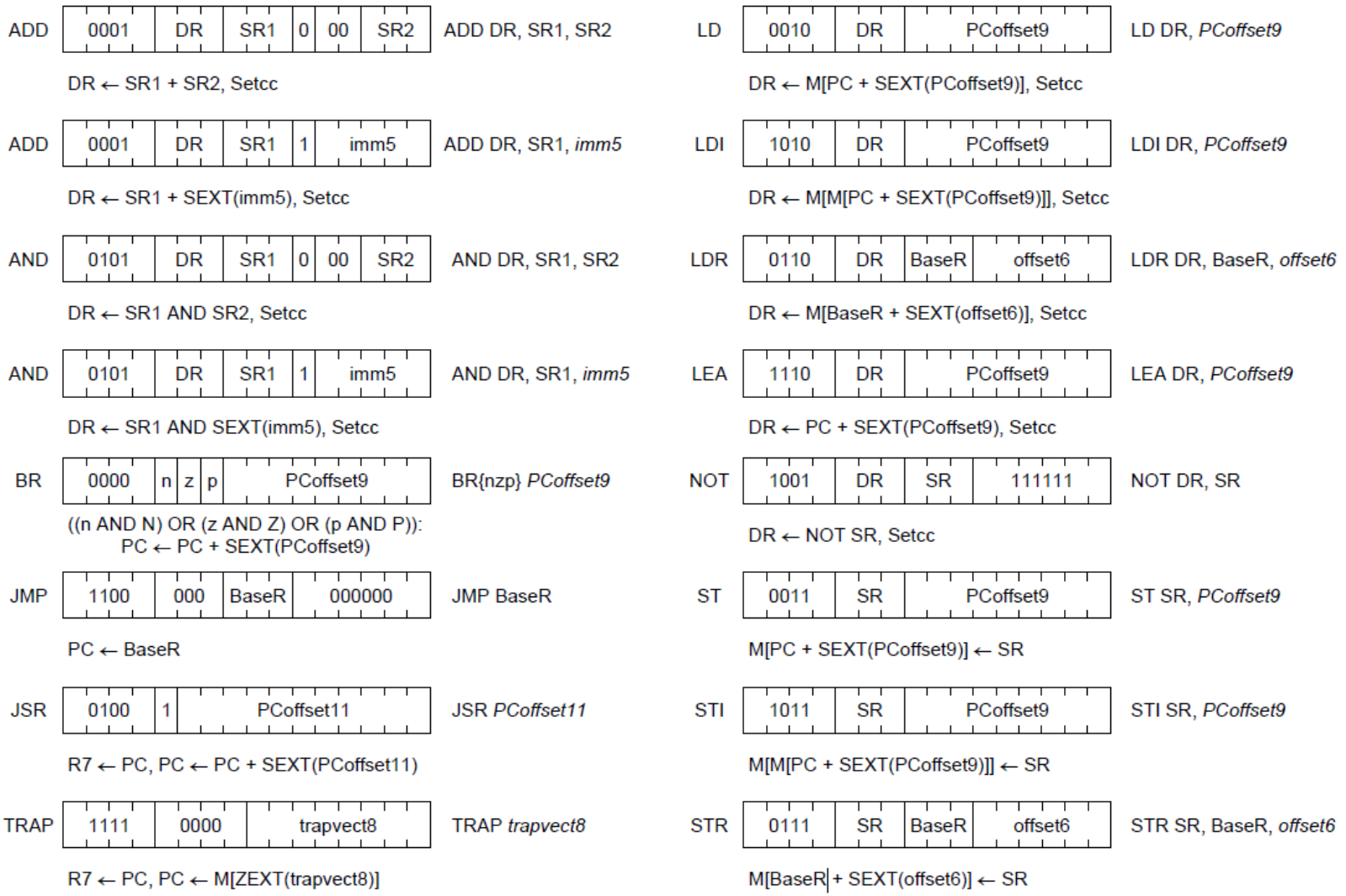
a) A significant difference between C and LC-3 assembly language is that it is trivial to transforms LC-3 assembly language to LC-3 machine language, whereas some work is needed to transform C to LC-3 machine language.
b) A significant difference between C and LC-3 assembly language is that C has no notion of 'memory address', while the LC-3 assembly language does.
Answer: a) True; b) False.

3. True or False (5 points)

A powerful difference between C and LC-3 assembly language is that the C preprocessing step can symbolically transform textual symbols in the C program file into native C statements, whereas the syntax of LC-3 assembly language makes any such initial preprocessing step impossible.

Answer: False.

NOTES: RTL corresponds to execution (after fetch!); JSRR not shown



End of ECE 220 Midterm Exam 1