Department of Electrical and Computer Engineering
The University of Texas at Austin

EE 306, Fall, 2006
Yale Patt, Instructor
TAs: Aseem Bathla, Cameron Davison, Lisa de la Fuente, Phillip Duran, Jose Joao, Jasveen Kaur, Rustam Miftakhutdinov, Veynu Narasiman, Nady Obeid, Poorna Samanta.
Exam 2, November 15, 2006

Name: Solution

Problem 1 (15 points): 15

Problem 2 (10 points): 10

Problem 3 (15 points): 15

Problem 4 (20 points): 20

Problem 5 (20 points): 20

Problem 6 (20 points): 20

Total (100 points): 100

Note: Please be sure that your answers to all questions (and all supporting work that is required) are contained in the space provided.

Note: Please be sure your name is written legibly on each sheet of the exam.

I will not cheat on this exam.

Solution

Signature

GOOD LUCK!

Name: Solution

Problem 1 (15 points)

Part a (5 points): An Assembly Language program contains the instruction `ADD R1, R2, #25`. What will the Assembler produce as a result?

Answer:
An error message saying that #25 cannot be represented in 5 bits.

Part b (5 points): An LC-3 machine language program contains the instruction:

0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

The PC contains the address of this instruction. What will the LC-3 computer do? Be careful. This question will require a little thought. The answer "The computer will execute the above instruction" will earn zero points.

Answer:
This instruction branches unconditionally to itself so the computer will be stuck in an infinite loop executing this instruction.

Part c (5 points): The HC-1 (Humongous Computer) has N different opcodes, 100 registers, and supports **operate** instructions just like the LC-3. It also allows immediate operands in the "style" of the LC-3, expressing values between -512 and +511. What is the maximum value for N if an instruction is 4 bytes long? Please show your work.

(DR) (SR1) (imm)
32 - 7 - 7 - 1 - 10 = 7 (1 = steering bit)

$2^7 = 128$

Answer: 128

Name: Solution

Problem 2 (10 points)

The following program is assembled and loaded into the LC3 simulator. Before execution, a breakpoint is set at the TRAP x25 instruction. The **run** button is pressed. What is the value of R1 when the breakpoint is reached?

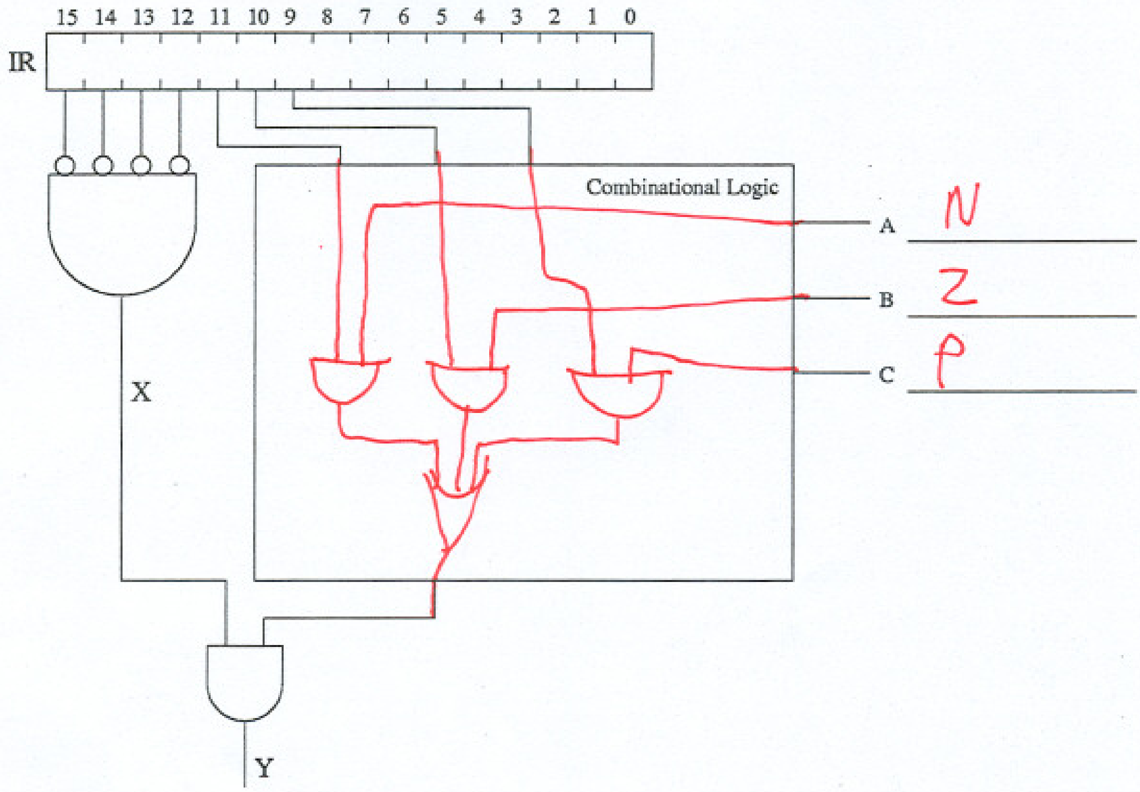
```
        .ORIG   x3000
        LD      R0, A
        AND     R1, R1, #0
        STR     R1, R0, #3
        ADD     R1, R1, #5   ← turns into NOP
        TRAP    x25
A       .FILL   x3000
        .END
```

Answer: 0

Name: Solution

Problem 3 (15 points)

The figure below shows some of the digital logic associated with an implementation of the LC-3 computer.



Part a (3 points): What information is provided by the signal labeled X in the diagram (in ten words or fewer, please)?

Answer:
Determines if the instruction is a branch.

Part b (3 points): What information is provided by the signal labeled Y in the diagram (in ten words or fewer, please)? Note that this signal depends on both the output of the AND gate and the output of the combinational logic block.

Answer:
Determines if the branch should be taken.

Part c (3 points): Identify the three inputs labeled A, B, and C in the space provided.

Part d (6 points): Design the combinational logic (inside the block shown) that is necessary for the signal Y to act as required by the LC-3.

Name: Solution

Problem 4 (20 points)

Part a (5 points): Generate the symbol table for the program below. You may not need all of the spaces provided.

```
        .ORIG  x3000
        JSR    A
        BR     END
A       LD     R0, LF
        TRAP   x21
        LEA    R0, EE306
        TRAP   x22
        LEA    R0, AWE
        TRAP   x22
        ADD    R7, R7, #-1
        RET
END     TRAP   x25
EE306   .STRINGZ "EE306 IS"
AWE     .STRINGZ " AWESOME!"
BUFFER  .BLKW 3
LF      .FILL  x0A
        .END
```

Symbol	Address
A	x3002
END	x300A
EE306	x300B
AWE	x3014
BUFFER	x301E
LF	x3021

Part b (15 points): What does this program output? Be specific. Also, be careful.

Answer:

EE306 IS AWESOME! AWESOME! AWESOME! . . .

(infinitely outputs AWESOME!)

Name: Solution

Problem 5 (20 points)

In Problem set 5, you were asked to write the subroutine BIN_GET which inputs a binary number entered from the keyboard and stores the zero extended value into R0. The program below calls the subroutine MOD_BIN_GET which in addition to storing the zero extended value in R0, also stores the total number of bits entered into R1. For example, if the user types 10010, R0 would contain 0000000000010010, and R1 would contain the value 5. Assume the user types from 1 to 16 binary digits.

After calling the MOD_BIN_GET subroutine, the program below sign extends the value that the user entered from the keyboard and stores the result in R0. However, a few instructions have been left out. Your job: complete the program. Note: Each box corresponds to 1 missing instruction.

```
        .ORIG  x3000
        JSR    MOD_BIN_GET
        AND    R2, R2, #0
        ADD    R2, R2, #1   ; R2 = 0000000000000001
        NOT    R3, R2       ; R3 = 1111111111111110
LOOP    ADD    R1, R1, #-1
        BRz    DONE
        ADD    R2, R2, R2
        ADD    R3, R3, R3
        BRnzp  LOOP
DONE    AND    R2, R0, R2
        BRz    SKIP
        ADD    R0, R0, R3
SKIP    HALT
        .END
```

Name: Solution

Problem 6 (20 points)

A program executing on the LC-3 computer encounters a breakpoint set at address x3500, and halts. The computer operator does not change the state of the computer in any way, but immediately presses the **run** button to resume execution.

The table below shows the contents of MAR and MDR for the first six memory accesses that the LC-3 performs after resuming execution.

Your job: Fill in the missing entries.

	MAR	MDR
1st:	x3500	xF017
2nd:	x0017	x0F00
3rd:	x0F00	xE209
4th:	x0F01	xB20F
5th:	x0F11	x4000
6th:	x4000	x0F0A

LEA R1, #9 (R1 = x0F0A

STI R1, #15