CS1356 Introduction to Information Engineering

Midterm, 2010/10/18 (Four pages, 15 questions, 110 points)

1. Convert the following base ten representations to equivalent 12-bit two's complement representations. (4 pt)
   a. 1568 b. −1569

2. Convert the following bit patterns to equivalent base ten representations, and express the bit pattern in hexadecimal notation. (6 pt)
   a. 010010011011 b. 010111011101

3. Performing the following additions. Assume the bit patterns represent values in two's complement notation. Identify the cases that will cause overflow. (9 pt)

   a.
   ```
     1010111
   + 0101001
   ```
   b.
   ```
     1101101
   + 1100011
   ```
   c.
   ```
     1001101
   + 1100001
   ```

4. Write the answer to each of the following logic problems. (6 pt)

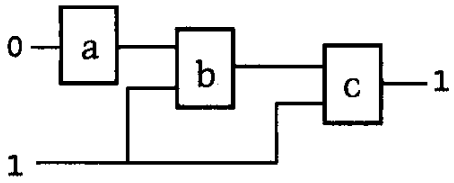
   a.
   ```
     10101010
   AND 11110000
   ```
   b.
   ```
     10101010
   OR 11110000
   ```
   c.
   ```
     10101010
   XOR 11110000
   ```

5. Decode the following bit patterns to the **base ten** representations using the floating point format described as follows. (4 pt)
   (a) The most significant bit is for sign: 0 for positive numbers and 1 for negative numbers.
   (b) The next three bits are for exponent. The exponent uses **excess notation**, whose range is from -4 to 3.
   (c) The last four bits are for mantissa, whose value is in the **range [1, 2)**.
   a. 01101111 b. 10111011

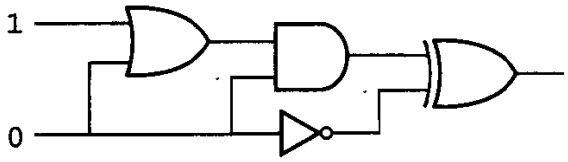
6. Encode the following numbers to the 8-bit floating point format described in problem 5, and indicate which one has a truncation error. (6 pt)
   a. -8.25 b. 0.1

7. What is the largest number the format described in problem 5 can represent? And what is the smallest positive number? Express them in base ten notations. (5 pt)

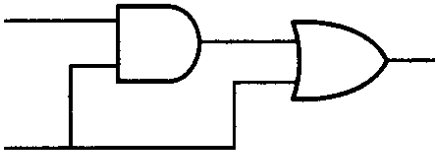
8. Answer the following questions about logic gates.
   a. In the circuit below, the rectangles represent the **different type** of gates. Based on the given input and output information, write out a possible group of a , b, and c. The gate involved in each rectangle is an AND, OR, NOT, or XOR. (6 pt)



   b. What is the output of the circuit below? (2 pt)



   c. For the following circuit, identify the input combinations that produce an output of 1. (Please write down all the possible answers.)(2 pt)



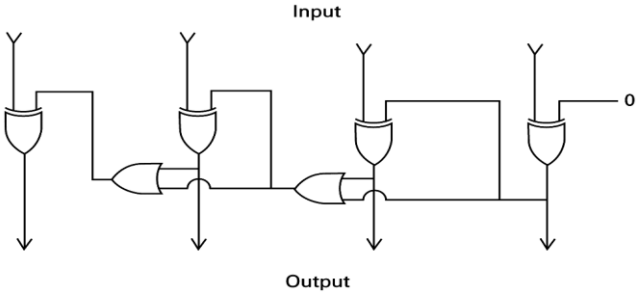
9. Right table shows a portion of a machine's memory containing a program written in the language described in the attached table. Assuming that the machine is started with program counter = 00. Describe which registers and memory cells are modified and what their values are when the machine halts? (10 pt)

Address	Contents
00	15
01	08
02	20
03	3F
04	53
05	05
06	33
07	00
08	C0
09	00

10. Write a short program (use at most 5 instructions) in the machine language described in the attached table to compute the **absolute value** of an 8-bit floating point, whose format is as described in question 5, stored at memory cell 30_H and place the result to memory cell 31_H. Assume your program is placed in memory starting at address 00_H. (*Hint: Put a 0 in the most significant bit without disturbing the other bits*.) (10 pt)

11. Write a short program in the machine language described in the attached table to compute the **absolute value** of an 8-bit integer (in 2's complement representation) stored at memory cell 30_H and place the result to memory cell 31_H. Assume your program is placed in memory starting at address 00_H. (10 pt)

12. In the textbook, the "*copying and complement*" procedure is used to convert a binary number to its 2's complement representation. It works as follows. First, it copies the bit pattern from right to left, until a 1 has been copied and then complements each remaining bit. (10 pt)
    a. Prove the algorithm can produce the correct 2's complement for any finite length bit pattern.
    b. Prove the following circuit correctly implements the algorithm for 4-bit binary numbers.



13. We said that all the functions in digital computers can be implemented by using gates, and showed how an adder and a memory unit (flip-flop) are made in class. Here we want to implement the *decoder* inside a CPU (used to decode op-code). A 2-input, 4-output decoder works as follows. When the input is 00, the first output is 1 and the rest are 0; when the input is 01, the second output is 1 and the rest are 0; when the input is 10, the third output is 1 and the rest are 0; and when the input is 11, the fourth output is 1 and the rest are 0. (10 pt)
    a. Show the truth table of this 2-input, 4-output decoder.
    b. Draw the gate circuit that implement this decoder using AND and NOT gates.

14. A multifunction electronic rice cooker has a microprocessor to setup time and the desired temperature, and can be used to cook many different dishes. Do you think this rice cooker is "*programmable*"? why or why not? (5 pt)

15. A song is converted to a digital music, which uses sampling rate 40,000Hz and uses 65,536 different values to store each sample. If a song is 5 minutes long, how large is the digitalized music data? Express your answer in mega-byte and round it to an integer. (5 pt)

Opcode	Operand	Description
1	RXY	**LOAD** the register R with the bit pattern found in the memory cell whose address is XY.
2	RXY	**LOAD** the register R with the bit pattern XY.
3	RXY	**STORE** the bit pattern found in register R in the memory cell whose address is XY.
4	0RS	**MOVE** the bit pattern found in register R to register S.
5	RST	**ADD** the bit patterns in registers S and T as though they were two's complement representations and leave the result in register R.
6	RST	**ADD** the bit patterns in registers S and T as though they represented values in floating point notation and leave the floating-point result in register R.
7	RST	**OR** the bit patterns in registers S and T and place the result in register R.
8	RST	**AND** the bit patterns in registers S and T and place the result in register R.
9	RST	**EXCLUSIVE OR** the bit patterns in registers S and T and place the result in register R.
A	R0X	**ROTATE** the bit pattern in register R one bit to the right X times. Each time place the bit that started at the low-order end at the high-order end.
B	RXY	**JUMP** to the instruction located in the memory cell at address XY if the bit pattern in register R is equal to the bit pattern in register number 0. Otherwise, continue with the normal sequence of execution. (The jump is implemented by copying XY into the PC during the execute phase.)
C	000	**HALT** execution.