Binary, Octal, and Hex Integers

Lab 1, due Fri Feb 1 (2400 hrs)[1]

CS 350: Computer Organization & Assembler Language Programming

A. Why?

- Binary integers are one of the basic ways to store information in a modern computer.

B. Outcomes

After this lab, you should be able to

- Represent negative binary integers in sign-magnitude, 1's complement, or 2's complement; to list the pros and cons of each of these three systems; to explain why hardware commonly uses 2's complement to represent negative integers.
- Translate between representations for integers in binary, octal, hex, and decimal (signed and unsigned) and to perform arithmetic using them.
- List the representations of the most positive and negative values in each system; know when overflow occurs and how to recognize it
- Declare, manipulate, and print signed and unsigned binary integers in C of various lengths.

C. Problems [70 points total]

For Problems 1 – 5, answer each question in (a) Sign-magnitude, (b) 1's complement, and (c) 2's complement.

1. [8 = 4*2 points] What does 110111 represent in decimal, and what bitstring is its negative? (I.e., (a) In sign-magnitude, 110111 represents decimal what? What is the in sign-magnitude negative of 110111? (b) …. 1's complement….? (c) …. 2's complement…?)

[1] **What to submit**: Write your solutions to the problems in Part C as comments in your *.c file for Part D. Submit the *.c file (no object file or output file) using Blackboard: Find Lab 1 under Assignments and press the link for uploading your solution. Remember, you get an extension to Monday Feb 4 if you attend lab on Jan 31 or Feb 1.

2. [6 = 3*2 points] What 6-bit string(s) represent zero?
3. [3 = 3*1 points] What is the most negative 6-bit string, and what is its decimal equivalent?
4. [3 points] Does taking the negative of the most negative number cause overflow?
5. [8 = 4*2 points] (a) In unsigned 6 bit arithmetic, what is 111011 + 101110?
   (b) What decimal value is 111011 is the 2's complement representation of?
   (c) What decimal value is 101110 is the 2's complement representation of?
   (d) What decimal value is your bitstring from part (a) the 2's complement representation of? (Hint: Your answers for (b) and (c) should add up to your answer for (d).)

For Problems 6 – 13, rewrite the following additions and subtractions in 6-bit 2's complement. For example, 3 – 5 = –2 is 000011 – 000101 = 000011 + (–000101) = 000011 + 111011 = 111110 = –(000010) = –2

6. [4 points] 12 + 18 = 30
7. [4 points] 13 – 30 = –17
8. [5 points] –25 – 7 = –32
9. [5 points] 24 + 10 = ??? (Be sure to show the decimal result; you should get overflow.)
10. [6 = 3*2 points] Convert 428_{10} to binary, octal, and hexadecimal.
11. [4 points] What is the octal representation of A63CB59_{16} ?
12. [6 = 3*2 points] As an 8-bit sign-magnitude number, FF_{16} represents $–\text{7F}_{16}$ $= –127_{10}$. What does it represent in (a) unsigned binary, (b) 2's complement, and (c) 1's complement?

13. [8 = 4*2 points] (a) What is the 16's complement of $F556_{16}$? Show the results of each of the following steps: (b) Convert $F556_{16}$ to binary; (c) Then take the 2's complement negative; (d) Then convert back to hex..

D. Programming Problem [30 points]

First, compile and run the skeleton file `Lab01_skel.c`.[2] You should get the following output:

```
CS 350 Lab 1 for ***Fill in your name and section ***

type char takes 1 byte(s), has max value of 127 = 0x7f
Adding 1 yields -128 = 0x80

type unsigned char takes 1 byte(s), has max value of 255 = 0xff
Adding 1 yields 0 = 0
```

The program output hopefully is self-explanatory, but the basic idea is that for each of the two types `char` and `unsigned char`, we print out the `sizeof` that type (the number of bytes a value of that type takes), we print out the largest value of that type in decimal and in hex, and we show what happens when you add 1 to that largest value: Since a `char` has 8 bits and uses 1 bit for the sign, the most positive `char` value is $0111\ 1111_2$ = 127 = $7F_{16}$; adding 1 to this yields $1000\ 0000_2$ = −127 = 80_{16}. For an unsigned char, the largest value is $1111\ 1111_2$ = 128 = FF_{16}; adding 1 to this yields $0000\ 0000_2$ = 0 = 0_{16}.

Your Assignment: (1) Replace `***Fill in your name and section ***` with your name and section and (2) Add code so that it works for the types `short int`, `unsigned short int`, `int`, `unsigned int`, `long int`, and `unsigned long int`.

Note: Do research as necessary to find out what the different `printf` formats and modifiers mean, because they won't be all the same. In particular, learn what `%d`, `%u`, and `%x` mean, and learn what the modifiers `#`, `hh`, `h`, and `l` (the letter ell) do.

Grading scheme: 5 points for each of the 6 types, with 1 point each for the correct sizeof the type, the max value in decimal and hex, and the result of adding 1 in decimal and in hex.

[2] In general, skeleton files don't have to compile and run, but this one does.