Lab 1: Computation & Data Representations

Due Date: Thursday 2/2/2017 11:59PM

This lab covers the first two lectures. Feel free to refer to my lecture notes (posted on the course page) and the recommended textbook (Patt & Patel) Chapters 1 and 2. Be sure to go to your labs on Friday to get assistance from the TAs. When you encounter a number written x_y you should read it as "x in base y." For example, 2_{10} means "2 base 10" (decimal 2) and 10_2 means "10 base 2" (binary 10, same thing as decimal 2). Numbers without a subscript are assumed to be in decimal.

If you're unsure about a result, be sure to explain your thinking. In general, you should show your work. A well-reasoned but wrong answer (for example with a minor error) will receive more points than a wrong answer with no context at all.

Problems (*100 points total*)

1. For the following, use no more than 3 sentences in your answer.

   a) Succinctly explain Turing's thesis.

   Any computation that can be performed can be performed by a Turing Machine. Put another way, any computable function can be computed by a TM.

   b) What are the components of a Turing machine?

   Read/Write head, Control, Tape. You might also split the control into two parts, the current state *and the* state transition table.

2. Explain some advantages digital computers have over analog computers.

   Digital circuits are simpler. There is less room for error (digital signals are less prone to noise*, that is small variations in a signal won't matter, because it's either* on *or* off*. With digital numbers, its easier to add precision (ability to represent more numbers). We just use another bit! Not so easy with an analog computer (e.g. you might have to increase the radius of a gear). With the digital case, the extra bits are already there.*

3. How would one write 9_{10} in *unary* (base one)?

   111111111_1

4. Write out the following numbers in terms of powers of 2. Omit the powers of 2 that are not present. For example, 43_{10} would be written as :

$$(1 \times 2^5) + (1 \times 2^3) + (1 \times 2^1) + (1 \times 2^0)$$

   a) 32

   (1×2^5)

   b) 15

   $(1 \times 2^3) + (1 \times 2^2) + (1 \times 2^1) + (1 \times 2^0)$

   c) 65

   $(1 \times 2^6) + (1 \times 2^0)$

   d) 10

   $(1 \times 2^3) + (1 \times 2^1)$

5. Convert 11101_2 to decimal. Assume it is unsigned.

   29_{10}

6. Solve for x in

$$2^x = 00100000_2$$

   5

7. Convert 01011001_2 to decimal. Assume unsigned again.

   89_{10}

8. a) Given a 4-bit string of binary digits, how many unique binary strings are there?

   $2^4 = 16$

   b) What's the largest *unsigned* integer we can represent with this many bits?

   $2^4 - 1 = 16 - 1 = 15$

   c) How many unique binary strings for an n-bit string?

   2^n

   d) And the largest unsigned integer with n bits?

   $2^n - 1$

9. Add 20 and 35 *in binary* (give the answer in binary). Assume 8-bit unsigned integers. Show your work.

   00110111_2

10. Subtract 20 from 87 *in binary*. Again, 8-bit unsigned.

    $01000011_2 = 67_{10}$

11. Take the *one's complement* of 01110101_2. Give the result in binary and in decimal.

    $10001010_2 = -117$

12. a) We have a 7-bit signed, *two's complement* integer. What are the maximum (most positive) and minimum (most negative) values we can represent with it?

    Max: $+63$*; Min:* -64

    b) For an n-bit signed, 2's complement int, what are the most positive and negative numbers we can represent?

    Max: $2^{n-1} - 1$*; Min:* -2^{n-1}

13. Fill in the rows in the following table. The first row is an example for your reference. Put an X when the number cannot be represented. Assume 6 bits.

Number	**Signed Magnitude**	**One's Comp.**	**Two's Comp**	**Unsigned**
-10	101010	110101	110110	X
-6	100110	111001	111010	*X*
7	000111	000111	000111	000111
-1	100001	111110	111111	*X*
10	001010	001010	001010	001010
-32	*X*	*X*	100000	*X*
31	011111	011111	011111	011111

14. Subtract 10_{10} from 7_{10} $(7 - 10)$ *in binary* using 6-bit, signed, 2's complement integers. **Hint**: $x - y = x + (-y)$.

    This is a case where, rather than subtracting, you should use the convenience of 2's complement to add the negative. Instead of doing $000111_2 - 001010_2$ *we can take the 2's complement negative of* 10 *and add it:*

    $$000111_2 + 110110_2 = 111101_2 = -3_{10}$$

Hand-in Instructions

Make sure to put your name on your submission. Submissions without names will be given zero points!

Physical : If you're submitting a written copy, hand it to one of the TAs or to the instructor. You can also leave it in the instructor's mailbox in the CS department office, but make sure to get it time stamped when you do (see the "Submitting Work" section of the syllabus).

Digital : If you would like to submit an electronic copy, note that I will only accept PDF files (no Word docs please). Again, see the "Submitting Work" section of the syllabus. Please do not take a poorly lit picture of your assignment. Your grade will suffer commensurately with our inability to read your work. Once you have a PDF, you should submit it on `fourier`. You should name your file `yourid-lab1.pdf` where `yourid` is the thing in front of the @hawk.iit.edu in your e-mail address.

You can first get your PDF (for example, for me it might be called `kh123-lab1.pdf`) onto `fourier` like so:

```
[me@mylocalmachine]$ scp kh123-lab1.pdf kh123@fourier.cs.iit.edu:
```

Then you can login to `fourier` via ssh and submit it:

```
[kh123@fourier]$ cp kh123-lab1.pdf /home/khale/HANDIN/lab1
```