Homework#1 for ECE 152
Introduction (Chapter 1)

<u>Hardcopy</u> is due in class on Monday, January 23

All homework must be done in a group of 2 students. Each group should turn in one hardcopy in class. If your handwriting is unreadable by typical humans, please type your homework.

1) [10 points] Sometimes software optimization can dramatically improve the performance of a computer system. (You probably remember this idea from CS 100, right? Now we're going to look at the same idea at a lower level.) Assume that a CPU can perform an exponentiation (X^Y) in 20ns, a multiplication in 10ns, an addition or subtraction in 1ns, and a left shift (by any amount) in 5ns. How long will it take for the CPU (without any software optimization) to naively calculate $c = 2^a 2^b$? How would you optimize the equation so that it will take less time? (Hint: there are two optimizations that will help.)

2) [10 points] A high level program can be translated by a compiler (and assembled) into any number of different, but functionally equivalent, machine language programs. (A simplistic and not particularly insightful example of this is that we can take the high level code C=A+B and represent it with either add C, A, B or add C, B, A.) When you compile a program, you can tell the compiler how much effort it should put into trying to create code that will run faster. If you type `g++ -O0 -o myProgramUnopt myProgram.C`, you'll get unoptimized code. If you type `g++ -O3 -o myProgramOpt myProgram.C`, you'll get highly optimized code. Please perform this experiment on the program myProgram.C located at http://www.ee.duke.edu/~sorin/ece152/homework/myProgram.C. Compile it both with and without optimizations. *Compare the runtimes of each and write what you observe.* (To time a program on a Unix machine, type "`time myProgram`", and then look at the number before the "u", as in the "0.40u" below. This number represents the time spent executing **<u>u</u>**ser code.)

```
sorin@carbon.ee.duke.edu [29] time myProgramUnopt
Finished
0.40u 0.02s 0:00.43 97.6%
```

3) [5 points] Represent the integer $+123_{10}$ in 10-bit 2's complement.

4) [5 points] Represent the integer -123_{10} in 10-bit 2's complement.

5) [5 points] Represent "G. Hill #33" in ASCII. Write your result in base 10. Pay careful attention to capital vs. lower-case letters as well as spaces.

6) [5 points] Consider the C-like code in Figure 1. What values do the last three lines print out? If it's printing an address, tell me what variable's address it is printing. Assume that the "print" statement "does what you'd expect" - it prints out the value of the address or variable after it.

```
int* p[20];
int q[20];
int i;
for (i=0; i<20; i++){
  q[i] = 3*i;
  p[i] = &q[19-i];
}
print q[3];
print p[11];
print *p[7];
```

FIGURE 1. Code for Question 6

7) [5 points] You have your choice between two desktop computers. One has an Intel Bloomfield (core i7) processor with 4 cores. The other laptop has an AMD Interlagos processor with 4 cores. Both systems are otherwise identical (same caches, memory, disk, etc.) and are the same price. Which desktop do you want? You must explain your answer to get credit.