Department of Electrical and Computer Engineering

The University of Texas at Austin

EE 306, Fall 2023
Problem Set 1
Due: **September 6**, before class
Yale N. Patt, Instructor
TAs: Chester Cai, Kayvan Mansoorshahi, Sophia Jiang, Jaeyoung Park, Varun Arumugam, Anna Guo, Nadia Houston, Asher Nederveld, Edgar Turcotte

Note: Please complete and **print** the student information sheet, and turn it on September 6th (the same day the problem set is due). Your submission should be **a single page** and include a recognizable recent photo of yourself.

Instructions: You are encouraged to work on the problem set in groups and turn in one problem set for the entire group. **The problem sets are to be submitted on Gradescope**. Only one student should submit the problem set on behalf of the group, but everyone should create a gradescope account and be tagged on the homework. If you have questions about this, please contact a TA.

1. (Adapted from problem 1.5 in the textbook)
   Say we had a "black box," which takes two numbers as input and outputs their sum. See Figure 1.10a in the Textbook. Say we had another box capable of multiplying two numbers together. See figure 1.10b. We can connect these boxes together to calculate p * (m + n). See Figure 1.10c. Assume we have an unlimited number of these boxes. Show how to connect them together to calculate:
   a. ax+b
   b. The average of the four input numbers w, x, y, and z
   c. $a^2 + 2ab + b^2$ (can you do it with one add box and one multiply box?)
   d. a^6 (can you do it using only 3 multiply boxes?)

   Hint: You may use constant numbers as inputs. A single value may be used multiple times as an input

2. (Adapted from 1.14)
   Suppose we wish to put a set of names in alphabetical order. We call the act of doing so sorting. One algorithm that can accomplish that is called the *bubble sort*. We could program our bubble sort algorithm in C, and compile the C program to execute on an x86 ISA. The x86 ISA can be implemented with an Intel Core microarchitecture. Let us call the sequence "Bubble Sort, C program, x86 ISA, Core microarchitecture" one transformation process.
   Assume we have available four sorting algorithms and can program in C, C++, Pascal, Fortran, and COBOL. We have available compilers that can translate

from each of these to either x86 or SPARC, and we have available three different microarchitectures for x86 and three different microarchitectures for SPARC.

   a. How many transformation processes are possible?
   b. Write three examples of transformation processes.
   c. How many transformation processes are possible if instead of three different microarchitectures for x86 and three different microarchitectures for SPARC, there were three for x86 and five for SPARC.

3. (2.3)
   a. Assume that there are about 400 students in your class. If every student is to be assigned a unique bit pattern, what is the minimum number of bits required to do this?
   b. How many more students can be admitted to the class without requiring additional bits for each student's unique bit pattern?

4. (Adapted from 2.13)
   Without changing their values, convert the following 2's complement binary numbers into 8-bit 2's complement numbers.
   a. 010110
   b. 1101
   c. 1111111000
   d. 01

5. (Adapted from 2.17)
   Compute the following. Assume each operand is a 2's complement binary number.
   a. 01 + 1011
   b. 11 + 01010101
   c. 0101 + 110
   d. 10101 + 01010

   Represent the answer with the same number of bits as the longer operand.

6. In class we learned that a bit is a number with one of two possible values, 0 or 1. Similarly, a *trit* is a number with one of three possible values: 0, 1, or 2. Given n trits, how many unique items can we represent? Now consider an arbitrary number system where each digit can be a number with m possible values. Given n of those digits, how many unique items can we represent?

7. For the following 8-bit binary numbers, write down the decimal number they represent as unsigned, signed magnitude, 1’s complement, and 2’s complement representations, respectively.
   a. 01010101
   b. 10000000

c. 11111111

~~8. Express the value 0.3 in the 32-bit floating point format that we discussed in class. Feel free to only show fraction bits [22:15], rather than all the fraction bits, [22:0]. Notation: The symbol [22:15] signifies all 8 bits from bit 22 to bit 15.~~

~~9. Convert the following floating point representations to its decimal equivalent~~
   ~~a. 1 10000010 10101001100000000000000~~
   ~~b. 1 11111111 00000000000000000000000~~
   ~~c. 0 00000000 00000001000000000000000~~

10. (Adapted from 2.50)
    Perform the following logical operations. Express your answers in hexadecimal notation.

    a. xABCD OR x9876
    b. x1234 XOR x1234
    c. xFEED AND (NOT(xBEEF))

11. (2.54)

    Fill in the truth table for the equations given. The first line is done as an example.

Q1 = NOT (NOT(X) OR (X AND Y AND Z))
Q2 = NOT ((Y OR Z) AND (X AND Y AND Z))

X	Y	Z	Q1	Q2
0	0	0	0	1