Sergio A. Alvarez
Department of Computer Science
Worcester Polytechnic Institute
Worcester, MA 01609, USA

web: http://www.cs.wpi.edu/~alvarez/
e-mail: alvarez@cs.wpi.edu
phone: (508) 831-5118
fax: (508) 831-5776

Data Representation: Numerical Data

Numbers account for some very common data types, including unsigned integers, signed integers, single-precision floating point numbers, and double-precision floating point numbers. I'll discuss popular binary representations of each of these numerical data types.

Unsigned Integers

Binary positional notation is used to represent unsigned integers in most digital computers. In this notation, a string of bits (binary digits, 1's and/or 0's) represents a uniquely defined integer in the following way. If the bit string contains the n bits $b_{n-1}b_{n-2}\cdots b_1b_0$, the corresponding integer is

$$\sum_{i=0}^{n-1} b_i 2^i$$

The word *positional* refers to the fact that the positions of the bits within the string determine their weights in the above sum. The system being used here is analogous to the ordinary decimal system, where strings like 372 and 25 correspond to specific numbers by way of a sum like that shown above, but involving powers of 10 rather than powers of 2 (that's *decimal positional notation*). For instance, in decimal positional notation, the string 372 represents the unsigned integer $2 * 10^0 + 7 * 10^1 + 3 * 10^2$. Bases other than 10 and 2 may be considered too. In all cases, the weight of a digit is greater the further left it appears in the string.

Example

The bit string 10110 represents the unsigned integer $0 * 2^0 + 1 * 2^1 + 1 * 2^2 + 0 * 2^3 + 1 * 2^4$, which in decimal notation is 22. To make it clear what base is being used in each case, one would usually write $10110b = 22d$. This prevents malicious readers from interpreting the left-hand side as a number greater than ten thousand (hey, suppose you're counting dollars, for example!?!)

Converting decimal to binary

What if you have, say, twenty five dollars? How would this number be represented in binary positional notation?

Repeated guess-and-subtract method. To convert $25d$ to binary, notice that the largest integer power of 2 that is less than or equal to $25d$ is $2^4 = 16d$; write 1 in the bit 4 position of the sought-after binary string. Then repeat the process with the remainder $25d - 16d = 9d$: the largest power of 2 that doesn't overshoot $9d$ is $2^3 = 8d$; write 1 in bit position 3. The remainder is now $9d - 8d = 1$; this is just 2^0, so write 1 in bit position 0. Write 0's in all other bit positions. This yields the bit string 11001.

Repeated division method. Start with $25d$. Divide by 2, getting an integer quotient of $12d$ and a remainder of 1; place this remainder of 0 in bit position 0 of the output string. Divide the quotient $12d$ by 2, getting quotient 6 and remainder 0; place the remainder 0 in bit position 1. Next step yields quotient 3 and remainder 0, so bit position 2 contains 0. Then bit position 3 receives the remainder of 1 from the integer division 3/2, and the quotient of 1 becomes the remainder for the final bit position 4 value of 1. The final bit string is 11001.