Data Storage Concepts (in progress)

1 Byte = 8 bits

Number systems

We use the base 10, or decimal, number system.

- 10 is called the base.
- We have 10 digits, that is, 0 through 9.
- The largest digit is 9 (one less than the base).

Let's generalize:

In all number systems, the following is true:

1. The number of digits is equal to the base.
2. The largest digit is one less than the base.
3. The base, in its own number systems looks like this: 10.

Example.

In Base 10:

$25744_{(10)} = (2 \times 10^4) + (5 \times 10^3) + (7 \times 10^2) + (4 \times 10^1) + (4 \times 10^0)$

In Base 2:

$11101_{(2)} = (1 \times 2^4) + (1 \times 2^3) + (1 \times 2^2) + (0 \times 2^1) + (1 \times 2^0) = 29_{(10)}$

This is the decimal equivalent of the binary number.

[Exercise: How to convert from decimal to binary?]

base 10, base 2, base 8, base 16

Exercise:

Perform the indicated operations:

1. $124_{(10)}$ → _______________ $_{(16)}$
2. $124_{(10)}$ → _______________ $_{(8)}$
3. $124_{(10)}$ → _______________ $_{(2)}$
4. $A_{(16)} + 5_{(16)}$ → _______________ $_{(16)}$
5. $11001_{(2)}$ → _______________ $_{(10)}$

Convert each of the following binary values to octal (base 8):

6. 0110 0111 ans. ________________________
7. 1101 1110 ans. ________________________
8. 0011 1100 ans. ________________________
9. 1011 1101 0111 1010 ans. ________________________
10. 0000 1111 0100 1000 ans. ________________________

Convert each of the following binary values to hexadecimal (base 16):

11. 0110 0111 ans. ________________________
12. 1101 1110 ans. ________________________
13. 0011 1100 ans. ________________________
14. 1011 1101 0111 1010 ans. ________________________
15. 0000 1111 0100 1000 ans. ________________________

Storing numbers as binary codes

What size number can we store in 1 byte? (if only positive integers are needed)

	___	___	___	___	___	___	___	___	≤ 255 $_{(10)}$
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	
=	128	64	32	16	8	4	2	1	

Two's Complement Code

If we wish to allow for negative numbers, we use "two's complement," e.g.:

	0	0	0	0	0	0	0	0	= 0 $_{(10)}$
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	
=	-128	64	32	16	8	4	2	1	

	1	0	0	0	0	0	1	1	= -125 $_{(10)}$
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	
=	-128	64	32	16	8	4	2	1	

etc.
An integer stored in "two's complement" can go from -128 to +127.

Floating point values: To store, e.g., the number 123.45, first convert to binary:
$123_{(10)}$ = $01111011_{(2)}$ and $.45_{(10)}$ = $01110011_{(2)}$ (same as 115/255)
So, $123.45_{(10)}$ = $1111011.01110011_{(2)}$
The decimal point is then "floated" so all bits are on the right: .111101101110011 x 2^7 (the exponent is 7 because the point was floated 7 bits to the left)
The mantissa and exponent are stored separately. Generally, the mantissa .11110110111011 is stored in 23 bits; the exponent 7 is stored in 8 bits; leaving 1 bit for the sign of the number.