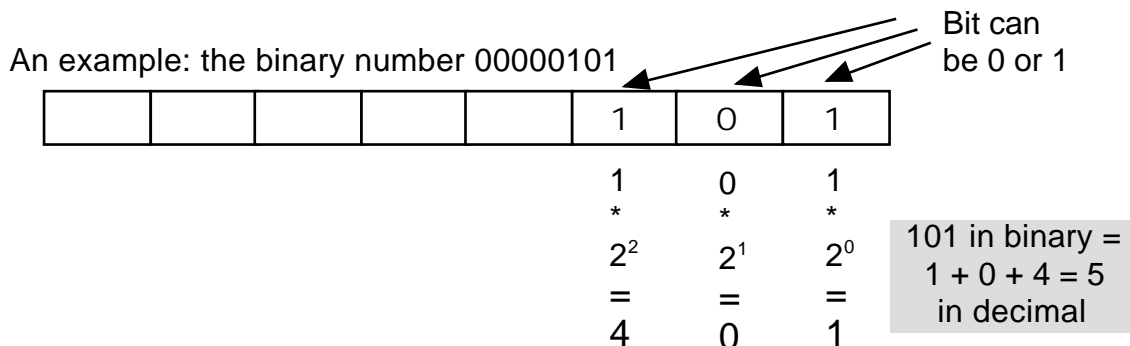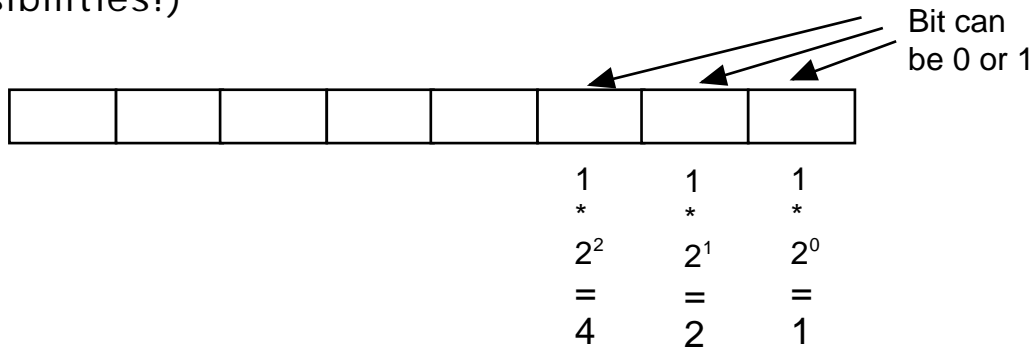# Bits and Bytes

In a <u>BIT</u> (<u>b</u>inary dig<u>it</u>) we can store only: *zero* (0) or *one* (1)
We call this digital or binary data.

In a <u>BYTE</u> (8 Bits) we can store also only binary data (0 or 1). By defining a CODE we can represent all possible characters. In one BYTE

we can store up to $2^8 = 256$ different characters.

(A little Math; the base of the system is 2 digits, you have 8 positions, so you have 2 to the power of 8 = 256 possibilities!)

Bit can be 0 or 1

$$1 * 2^2 = 4 \qquad 1 * 2^1 = 2 \qquad 1 * 2^0 = 1$$

Bit can be 0 or 1

An example: the binary number 00000101

					1	**0**	**1**

$$1 * 2^2 = 4 \qquad 0 * 2^1 = 0 \qquad 1 * 2^0 = 1$$

101 in binary = 1 + 0 + 4 = 5 in decimal

CODED DATA

In America they defined an international code, which could be used all over the world. It's called the ASCII-table. (American Standard Code for Information Interchange).

It works simular to this:

BYTE		Character
0100	0001	A
0100	0002	B
0100	0003	C
0010 0001		!

All numbers and special characters (like : > + etc.) have their own ASCII-code.
In this way, the computer can work with characters and numbers.

Bits and Bytes

BIT
contains 0 or 1

BYTE
can store:
- a character
- a number (real or integer)
- a computer instruction
- a memory address
- a (bit of) sound / video

The computer (in fact the CPU) works with the bytes. Of course it needs to be told if the zero's and one's are representing an ASCII-character, a number, an instruction or an address (or sound, video . . .)

0011 0000 could be:
- the character: 0
- the number: 48
- the instruction: add 2 numbers
- the address: sector 2, track 6