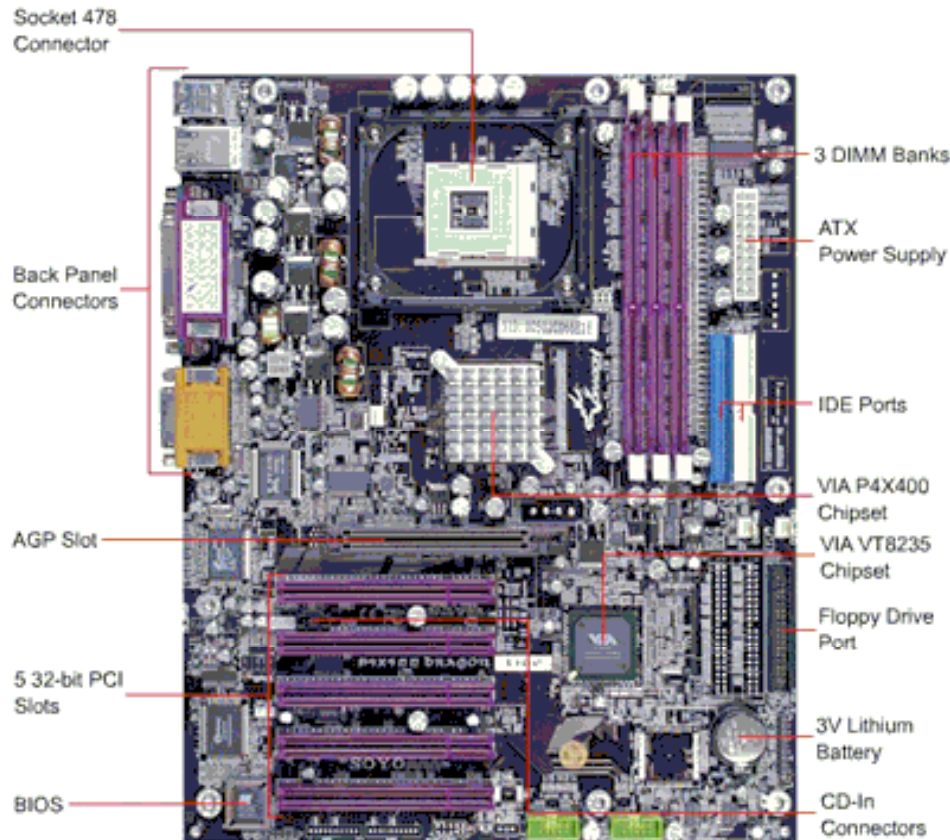


## Bits and Bytes

We want to speak English or some natural language to computers, but there's really only electric currents running inside them.



### *Levels of Abstraction*

Electric currents

0s and 1s– ON current is 1 and an OFF current is 0

Machine Language – Numbers for commands and data

Assembly Language

Some symbols allowed, e.g., Mov X,3456

Still dealing directly with machine: Refer directly to memory cells, registers, devices, etc.

High-Level Language (Java, C++, Python)

Symbolic and logical , with few if any direct references to addresses, registers, or other hardware components

Natural Language, e.g., English

***Bits to bytes, integers and strings***

We have only 0s and 1s, but we want to represent numbers and characters and other data which require numerous 0s and 1s.

A **bit** is a single 0/1

A **byte** is 8 bits, e.g., 01010101

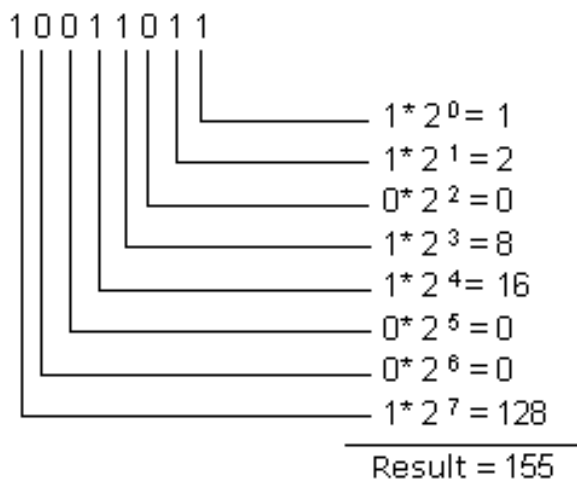
A **word**, which is used to store integers, is often 32 bits (4 bytes)

To represent a positive integer, we must convert binary (the 32 bits) into decimal. We do this by assigning bit  $i$  with the value  $\text{bit} \cdot 2^i$

Another way to put it is that, starting from the right most bit and going left, we assign 1,2,4,8,16,32, etc. to each bit that is 1.

So 1010 is:  $8+0+2+0=10$

The decimal 155 is:



You need to learn how to convert binary–decimal, back and forth, by hand. If you want to check your answers, here's a table:

<http://mistupid.com/computers/binaryconv.htm>

### **Characters**

We say that 8 bits is a byte.

In the past, letters, digits and other characters were represented with one byte (8 bits), with each symbol being mapped to a number between 0–255.

The ASCII table provides the mapping:

<http://www.lookuptables.com/>

So as not to be English–centric, there is now Unicode, which can represent many different languages and requires 2 bytes (16 bits) for representation.

### **Worksheet**

1. Consider the following sixteen bits:

0000 0000 0100 1101

a. What is the value of the number if it represents a positive integer (whole number)?

b. What symbol does it represent? You can use the lookup table above.

2. Encode the following three decimal numbers in binary:

24 1698 43

3. If 16 bits are used to encode positive integers, what is the largest number that can be represented?

4. Why do you think the R, G, and B values of pixels in JES have a limit of 255?

5. What about negative numbers? How do you think these are represented?