

# Binary and Hex

**How to count like a computer**

# Why Binary?

- The basic unit of storage in a computer is the *bit* (binary digit), which can have one of just two values: 0 or 1.
- This is easier to implement in hardware than a unit that can take on 10 different values.
  - For instance, it can be represented by a transistor being off (0) or on (1).
  - Alternatively, it can be a magnetic stripe that is magnetized with North in one direction (0) or the opposite (1).
- Binary also has a convenient and natural association with logical values of False (0) and True (1).

# Building on Binary

- Binary bits are grouped together to allow them to represent more information:
  - A *nybble* is a group of 4 bits, e.g. 1011
  - A *byte* is a group of 8 bits, e.g. 11010010
  - A *word* is a larger grouping: usually 32 bits. A *halfword* is half as many bits as a word, thus usually 16 bits. A *doubleword* is twice as many bits, usually 64 bits. However, computers have been designed with various word sizes, such as 36, 48, or 60 bits.
- Clearly, the number of possible combinations of a group of N bits is  $2^N = 2 \times 2 \times 2 \dots \times 2$  (N 2s). Thus:
  - A nybble can form  $2^4=16$  combinations
  - A byte can form  $2^8=256$  combinations
  - A 32-bit word can form  $2^{32}=4,294,967,296$  combinations

# Building on Binary

- Each combination of a group of bits can be assigned a symbolic meaning. For instance, bytes can be used to represent text by associating each byte value with a character. For example, the ISO-8859-1 character encoding (an extension of the older ASCII code) assigns the value 00101110 to mean a period ('.'), 01000001 to mean capital 'A', and so on.
- It is also possible to represent numbers using binary. The binary numbering system is like the decimal system, except that the only two digits used are 0 and 1, and digits are multiplied by powers of 2 instead of 10.

# Decimal and Binary Numbers

In the decimal system, each digit has a weight that is 10 times the weight of its neighbor to the right. Thus:

$$1234 = (((1 \times 10 + 2) \times 10 + 3) \times 10) + 4$$

Similarly, in a binary number, each digit has a weight that is 2 times the weight of its neighbor to the right. Thus:

$$1011 = (((1 \times 2 + 0) \times 2 + 1) \times 2 + 1$$

Expressing this in decimal it is  $((2 \times 2 + 1) \times 2 + 1 = 5 \times 2 + 1 = 11$ .

# Binary Numbers

Here are the first 16 binary numbers (using 4 bits), and their decimal equivalents.

$$0000 = 0$$

$$0001 = 1$$

$$0010 = 2$$

$$0011 = 3$$

$$0100 = 4$$

$$0101 = 5$$

$$0110 = 6$$

$$0111 = 7$$

$$1000 = 8$$

$$1001 = 9$$

$$1010 = 10$$

$$1011 = 11$$

$$1100 = 12$$

$$1101 = 13$$

$$1110 = 14$$

$$1111 = 15$$

# Binary and Hexadecimal

- Because binary numbers are rather unwieldy, programmers prefer to use a more compact way to represent them. Historically, octal (base 8) and hexadecimal (base 16, or hex) have been used.
  - Octal has the advantage that it uses only familiar digits, but it groups digits by threes, which is inconvenient for word sizes that are multiples of 4. So it is seldom used nowadays.
  - Hexadecimal works nicely for bytes and for word sizes that are multiples of 4, but requires the introduction of 6 new digits.
- Conversion between binary and decimal is slow and is preferably avoided if there is no need. Octal and hex allow immediate conversion to and from binary.

# Binary and Hexadecimal

Hexadecimal works by grouping binary bits into groups of 4 (starting from the right). Each group (a nybble) is assigned a hex digit value. The digits are the same as for decimal up to 9, and then letters A through F are used for 10 through 15.

0000 = 0

1000 = 8

0001 = 1

1001 = 9

0010 = 2

1010 = A

0011 = 3

1011 = B

0100 = 4

1100 = C

0101 = 5

1101 = D

0110 = 6

1110 = E

0111 = 7

1111 = F

Thus the 16-bit binary number

1011 0010 1010 1001

converted to hex is

B2A9

# The RGB Color System

- HTML styles define only 16 colors that are guaranteed to be recognized by name. Most browsers recognize many other color names, but there is no guarantee. So many web developers use “RGB” values to define colors.
- RGB values are based on the Red, Green, Blue primary color system. Just about every color that the human eye can perceive can be represented by a combination of specific amounts of each primary color.
- To define colors numerically, each of the three primary colors is assigned a numerical strength. The three strengths of Red, Green, and Blue define the color.

# Expressing Colors in Hex

- It is convenient to define the strengths of the primary colors using 8-bit numbers, or bytes. An 8-bit number can be represented using two hex digits. The value of the hex number ranges from 00 to FF, corresponding to a range from 0 to 255 decimal.
- A strength of 0 means the color is completely absent. A strength of FF or 255 means the color is at its maximum.
- An RGB color is thus represented by three 8-bit numbers, or six hex digits. In HTML, a value is signified as hexadecimal by prefixing it with a '#' sign. For example, "#E703A0" would be such a value.

# Expressing Colors in Hex

- Note that 40 is  $\frac{1}{4}$  of full strength, 80 is  $\frac{1}{2}$ , C0 is  $\frac{3}{4}$ .
- Here are some examples of colors expressed in hex:
  - "#000000" Black (all colors off)
  - "#FFFFFF" White (all colors max)
  - "#FF0000" Red (red max, green & blue off)
  - "#00FF00" Green
  - "#0000FF" Blue
  - "#FFFF00" Yellow (equal amounts of red & green)
  - "#AEEEEE" Turquoise
  - "#808080" Gray (all colors half max)