Hexadecimal Numbers

Hexadecimal numbers, or hex numbers, are used when we want to work at a higher level with binary numbers. They are often used in programming as a short hand way to reference bits.

For instance:

Instead of referring to a number in binary as 10010110_2, you could use hex to refer to it as 96_{16}.

If you want to specify the base of a number in a print statement, you can do that in the Arduino programming language.

```
Serial.println(sonarDistance, HEX);
```

(the other ones are, BIN, OCT, DEC)

Let's count!

Decimal	Hex	Binary	Octal
0	0	0000	0
1	1	0001	1
2	2	0010	2
3	3	0011	3
4	4	0100	4
5	5	0101	5
6	6	0110	6
7	7	0111	7
8	8	1000	10
9	9	1001	11
10	A	1010	12
11	B	1011	13
12	C	1100	14
13	D	1101	15
14	E	1110	16
15	F	1111	17

To specify a hex number in the C language you precede the number with a `0x`, e.g. `0x96, or 0xE1`
the hex “letters” get treated just as if it were a number.

What about conversion? It’s the same as any number system. The digit on the far right has a weight of 16^0, the next 16^1, 16^2, etc. just like our binary number digits were weighted 2^0, 2^1, 2^2, etc.

A fun thing to do is the convert your age to hex.
$17_{10} = 11_{16}$
You get a lot younger. (My age is 3B – I guess it’s only good for certain ages, like 40 which becomes 28 – ah, those were the days!)

Usually we want to convert from hex to binary in our heads to make it correspond to bits in a register on a device we’re programming. If you do this regularly, the hex letters become memorized.

I included octal in the table. Octal is basically obsolete, but has been used in the distant past on certain computers like the one I worked with at UCSD, the Terak, to learn assembly language (it also had an 8” floppy drive). Octal works the same way, the digit weightings are 8^0, 8^1, 8^2, etc. Octal is a way to increase you age, 73_8, oh my, but hey, how about 21_8, maybe some value there.