**Base conversions:**

Example: 0.375 with base 10 to base 2 conversion is 0.011

0.375 \* 2 = 0.75 0

0.75 \* 2 = 1.5 1

0.5 \* 2 = 1 1

In reverse: 0.(2^-1)(2^-2)(2^-3)....

Coefficients are either 1 or 0 for base 2

Example: 5 in base 10 to base 2 conversion is 101

Example: 6 in base 10 to base 2 conversion is 110

Example: 0.5 in base 10 to base 2 is 0.1 because it’s ½

Or by this method: 0.5 \* 2 = 1 —> 1

**Decimal to Octal conversion:**

175 in base 10 to octal

175/8 = 21 with remainder 7

21/8 = 2 with remainder 5

2/8 = 0 with remainder 2

**So** 175 base 10 is 2578

Remainders go in reverse order

**Binary to Octal conversion:** take groups of 3 binary digits (range from 0 to 7)

8 = 2^3

Each group of 3 bits is an octal digit

| octal | binary |
| --- | --- |
| 0 | 000 |
| 1 | 001 |
| 2 | 010 |
| 3 | 011 |
| 4 | 100 |
| 5 | 101 |
| 6 | 110 |
| 7 | 111 |

Example 1 011 101.1101 to base 8 each group of 3 binary digits corresponds to an octal base number

Is 135.648: Last digit is 4 because we need to extend the zeroes

1011101.1101 to Hexadecimal: extend zeros to 4 digits **(Hexadecimal is 0 to 9 and A to F for 10-15)**

0101 1101 . 1101

= 5D.D

**Binary Arithmetic**

* In decimal, we carry when the sum is 10 or more
* In binary we carry when the sum is 2 or more

**Binary Multiplication is the same as decimal multiplication, we only operate with 1s and 0s**

122\*17 in hexadecimal

How many bits are needed to represent a range of sounds with frequencies from 1 to 1,000,000?

Answer: at least 20 bits because at the very minimum 2^20 is 1048576

**Signed Binary Numbers**

**What is 2s complement used for?:** 2's complement is used for representing signed numbers and performing arithmetic operations such as subtraction, addition, etc. The positive number is simply represented as a magnitude form.

10110

01001 is 1s complement

01010 is 2s complement

2s complement is just 1s complement with 1 added to the last bit

Ex. 101100

010011 is 1s complement

010100 is 2s complement

1001110

0110001 is 1s complement

0110010 is 2s complement

2s complement of 010 0000 0000 —> 1111 1111 + 0000 0001 —> 0000 0000 overflow is ignored

**3 Types of signed representations**

* Sign and magnitude
* 1s complement: invert all bits of original value
* 2s complement: add 1 to the rightmost bit of 1s comp.

In signed magnitude, if leading bit is 1, number is negative, and if it’s 0 then number is positive

1101 is -5 signed, maximum is -7 with 1111: now we can represent -7 to 7; same range as unsigned but magnitude range is halved: this is because the unsigned 4 bit range is from 0 to 15

Two’s complement of -5 in unsigned 4 bit is -3

**Examples:** 4 bit unsigned is 0 to 15, 4 bit signed is -7 to 7, 8 bit signed is -127 to 127

Using 2s complement to subtract

7-3

0111 - 0011

Which is 7 + 2s comp of 3

0111

+1101

—------

0100 which corresponds to 4 so the math checks out