Binaries Conversion

Purpose:   
1. knowing to convert Base N is a need for programming  
2. To be able to convert Floating point numbers back and forth  
3. Bit Manipulation in any application (logic)  
4. Optimization  
5. Debugging

* Important conversion to know for Programming:
  + HEX (Base 16) Base: 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F, A=10, B=11, C=12, D=13, E=14, F=15
  + OCT (Base 8) = Base: 0,1,2,3,4,5,6,7
  + DEC (Base 10) = Base: 0,1,2,3,4,5,6,7,8,9
  + BIN (Base 2) = Base: 0,1

Convert DEC to ANY Base N

Convert Base 10 to Base 2 by Division Example:  
2 | 20 | (Setting Up)  
 10 | 0 (20 / 2 = 10 remainder 0)  
 5 | 0 (10 / 2 = 5 remainder 0)  
 2 | 1 (5 / 2 = 2 remainder 1)  
 1 | 0 (2 / 2 = 1 remainder 0)  
    1 (1 < 2, stop)   
BIN = 10100 (Concat Remainders from bottom up)

Convert Base 10 to Base 3 by Division Example:  
3 | 20 | (Setting Up)  
        6 | 2 (20 / 3 = 6 remainder 2)  
        2 | 0 (6 / 3 = 2 remainder 0)  
 2 (2 < 3, stop)  
Base 3 = 202 (Concat Remainders from bottom up)

Convert Base 10 to OCT by Division Example:  
8 | 20 | (Setting Up)  
        2 | 4 (20 / 8 = 2 remainder 4)  
        2 (2 < 8, stop)  
OCT = 024 (Concat Remainders from bottom up)  
NOTE: THE 0 in front represent the number as an OCT base 8

Convert Base 10 to HEX by Division Example:  
16 | 20 | (Setting Up)  
           1 | 4 (20 / 16 = 1 remainder 4)  
           1 (1 < 16, stop)  
HEX = 0x14 (Concat Remainders from bottom up)  
NOTE: the 0x in front represent the number as a HEX base 16

Convert DEC to ANY Base N

Convert Base 2 to DEC by Division Example: Given BIN = 10100  
Starting from LEAST significant bit:  
1 0 1 0 0  
1 \* 2^4  + 0 \* 2^3  + 1 \* 2^2 +  0 \* 2^1 + 0 \* 2^0  
= 16 + 0 + 4 + 0 + 0 = 20

| **Binary (base 2)** | **number \* baseN ^ bit position** | **Result (sum all together)** |
| --- | --- | --- |
| 0 | 0 \* 2^0 | 0 |
| 0 | 0 \* 2^1 | 0 |
| 1 | 1 \* 2^2 | 4 |
| 0 | 0 \* 2^3 | 0 |
| 1 | 1 \* 2^4 | 16 |

DEC = 16 + 0 + 4 + 0 + 0 = 20

Convert Base 3 to DEC by Division Example:

Given Base 3 = 202  
2 \* 3^2 + 0 \* 3^1 + 2 \* 3^0 = 18 + 0 + 2 = 20

| **Binary (base 3)** | **number \* baseN ^ bit position** | **Result (sum all together)** |
| --- | --- | --- |
| 2 | 2 \* 3^0 | 2 |
| 0 | 0 \* 3^1 | 0 |
| 2 | 2 \* 3^2 | 18 |

DEC = 18 + 0 + 2 = 20

Convert OCT to DEC by Division Example:

Given OCT = 024 = 2 x 8^1 + 4 x 8^0

| **Binary (base 8)** | **number \* baseN ^ bit position** | **Result (sum all together)** |
| --- | --- | --- |
| 4 | 4 \* 8^0 | 4 |
| 2 | 2 \* 8^1 | 16 |

DEC = 16 + 4 = 20

Convert HEX to DEC by Division Example:

Given HEX = 0x14

| **Binary (base 16)** | **number \* baseN ^ bit position** | **Result (sum all together)** |
| --- | --- | --- |
| 4 | 4 \* 16^0 | 4 |
| 1 | 1 \* 16^1 | 16 |

DEC = 16 + 4 = 20

Note In C/C++:  
DEC is represent : 1234567890 (normal)  
OCT is represent : 012345670 (with a zero at the front)  
HEX is represent : 0x12345678, 0x9ABCDEF1 (with a”0x”)

## Convert Base N to ANY DEC

Convert Base 2 to DEC by Division Example:

Given BIN = 10100

Starting from LEAST significant bit:

(Most) (Least)

1 0 1 0 0

1 x 2^4 + 0 x 2^3 + 1 x 2^2 + 0 x 2^1 + 0 x 2^0

16 + 0 + 4 + 0 + 0

0 - 0 x 2^0 = 0 x 1 = 0

0 - 0 x 2^1 = 0 x 2 = 0

1 - 1 x 2^2 = 1 x 4 = 4

0 - 0 x 2^3 = 1 x 8 = 0

1 - 1 x 2^4 = 1 x 16 = 16

DEC = 16 + 0 + 4 + 0 + 0 = 20

Convert Base 3 to DEC by Division Example:

Given Base 3 = 202

(Most) (Least)

2 0 2

2 x 3^2 + 0 x 3^1 + 2 x 3^0

18 + 0 + 2

DEC = 18 + 0 + 2 = 20

Convert OCT to DEC by Division Example:

Given OCT = 024

(Most) (Least)

2 4

2 x 8^1 + 4 x 8^0

16 + 4

DEC = 16 + 4 = 20

Convert HEX to DEC by Division Example:

Given HEX = 0x14

(Most) (Least)

1 4

1 x 16^1 + 4 x 16^0

16 + 4

DEC = 16 + 4 = 20

## 

## Pros and Tips

Convert HEX to BIN:

Given HEX = 0x1AE4F872

Each character represents 4 bit, so it should be straight forward

1 A E 4 F 8 7 2

0001 1010 1110 0100 1111 1000 0111 0010

4 bit partition: 0001\_1010\_1110\_0100\_1111\_1000\_0111\_0010

In Memory: 00011010111001001111100001110010

Convert HEX to OCT:

3 bit partition: 00\_011\_010\_111\_001\_001\_111\_100\_001\_110\_010

0 3 2 7 1 1 7 4 1 6 2

00 011 010 111 001 001 111 100 001 110 010

OCT = 03271174162

Convert OCT to BIN:

Given HEX = 01257346

Each character represents 3 bit, so it should be straight forward

1 2 5 7 3 4 6

001 010 101 111 011 100 110

3 bit partition: 001\_010\_101\_111\_011\_100\_110

In Memory: 001010101111011100110

Convert OCT to HEX:

4 bit partition: 0\_0101\_0101\_1110\_1110\_0110

0 5 5 E E 6

0 0101 0101 1110 1110 0110

HEX: 0x055EE6