Positional Number Systems

Humans are used to base 10

They are based on powers of 10.

eg,

5	4	3	2	1	0
3	2	5	8	6	7

position no. = power (p) 10^p

ie, $3 \times 10^5 + 2 \times 10^4 + 5 \times 10^3$

Indicating the numbers

Subscripts are used to determine decimals and binary numbers. ten, 10, d are used for decimals two, 2, b are used for binary.

eg 3_{ten} 11_{two}

3_{10} 11_{2}

3_{d} 11_{b}

$$\boxed{\text{digit} - d \times \text{base}^{i} \quad i - \text{position}}$$

i will start at the right & beginning with 0

Binary Numbers

- Base 2 is used in computers.
- Digits go from 0 - 1.
- Binary digits are known as bits.
- two bit

1st pos
00 - 0 pos
01
10
11

2 - Base - how to work out

3 2 1 0
1101 $2^0 = 1$ $2^1 = 2$ $2^2 = 4$ $2^3 = 8$

Bit - Most basic unit of information in a digital comp.

Byte - 8 bits

Word - 2 or more bytes

Representing positive integers

Least significant bit - Utmost right 2^0

Most significant bit - utmost left 2^{31}

Endianness

Enianness defines the order in which bytes are arranged into words.

Endian	First byte	last byte	
Big	Most significant	least	

Small	least significant	most	

Signed Magnitude representation

+ and − signs may be assigned. But these lead to less bits for actual numbers

(−2) 1 0000010

(+2) 0 0000010

- This can lead to programming headache as both +0 and −0 are valid.

Arithmetics

Carry ons.

```
          1   1 1 — carry ons
  signs  [0]   1 0 0 1 1 0 0
         [0] + 0 0 1 1 1 1 1
         -----------------------
         [0]   1 1 0 1 0 1 1
```

If carry ons exceed the bit count, then you will get an error. (Overflow)

2's complement

For positive integers, you leave it as is.

For negative integers, you flip the binary numbers eg 110 → 001, then add 1.

$\boxed{\begin{array}{l}\text{radix complement} \\ r^d - N\end{array}}$

overflow may occur in 2's complement
but does not result in error

3 $27 \div 16$

33_8 1 11

1 0011110 1101

$1 + 4 + 8 + 32 + 64 + 128 + 256 + 2048$

2541

158 r 13

hexa

9ED 317 r 5

3775 5

~~83~~

$5 + 448 +$

$7 + 56 +$

$\cancel{512 \times} 64 \times 3$

$5 + 56 + (64 \times 7) + 3 \times 512$

40

0×1

$4 \times 8 = 32$

400

0×1

4×64

256

$480 = 320$

$470 = 312$

4755

$5 + 40 + (7 \times 64) + 4 \times 512$

Octal (4755)

~~10~~

1 1 1   1

0 0 0 1 0 1 0 1

1 1 1 1 0 0 0 ~~0~~

0 0 0 0 0 1 1 0

1 1 1 1

0 0 0 0 1 1 1 1

1 1 1 0 1 0 1 1

1 1 1 1 1 0 1 0

1 1 1 0 1 0 1 ~~0~~

0 1 1 1 1 1 1 ~~0~~

64 32 16 8 4 2 1

1 1 1 1 0 1 0 0

128 64 32 16 4 2