Page No.:
Date:
YOUVA

* Signed and Unsigned Numbers:

An n bit signed binary number consists of two parts:

(1) Denoting the sign of the number and
(2) Denoting the magnitude of the number.

The MSB is always a sign bit.

0, 1
↓
denotes '+', '-'

smallest eight bit number 0000 0000
largest eight bit number 1111 1111
Range: 0 to 255

Binary Number

ve ; The 1s complement: +ve are same but we need to define the negative number in system

Unsigned number 0 0 1 1 0 1 1 0
1s complement 1 1 0 0 1 0 0 1

All -ve no.s have the Binary MSB = 1

Complement of Number = -ve number convert as +ve

-ve Binary Numbers - The 2s complement: (For -ve Binary no.)

do not put 1 in MSB of a Binary number to make it negative.

We must take the 2s complement of the number. Taking the 2s complement of the number will cause the MSB to become 1.

Unsigned Number 0 0 1 1 0 1 1 0
1s complement 1 1 0 0 1 0 0 1
2s complement + 1
1 1 0 0 1 0 1 0

IF we are using signed Binary numbers and MSB is already logic 1, it means the value is the 2s complement of the number.

Representation of signed numbers Using 2s complement:

+ $(12)_{10}$ in 2s complement form.

Binary number		1	1	0	0
1s complement		0	0	1	1
2s complement				+	1
		0	1	0	0
With sign bit	0	0	1	0	0
	+				

- $(-10)_{10}$ in 2s complement form.

Binary Number		1	0	1	0
1s complement		0	1	0	1
2s complement					+ 1
		1	0	0	0
with sign bit	1	1	0	0	0
	-				

* Addition- subtraction of Signed Numbers Using 2s complement Addition:

Ex: Add $(27)_{10}$ and $(-11)_{10}$ using complementory representation for the -ve value.

$(27)_{10} = (011011)_2$
$(11)_{10} = (001011)_2$

Get the 2s complement of $(001011)_2$ = ~~001011~~

Is com. 110100
+ 1
2s comple. 110101

~~Is 110100~~
~~for 2s +~~ ~~1~~
110101

$(27)_{10}$ 011011
2s com. + 110101
1 010000

"If 1 carry occur in solution ignore it from the result to obtain correct solution."

Note: Here, carry 1 is ignored and solution is $(010000)_2$

Ex: Subtroct $(25)_{10}$ From $(42)_{10}$

$(25)_{10} = (011001)_2$
$(42)_{10} = (101010)_2$

42
-25
17

Get the 2s complement of $(011001)_2$ = ~~011001~~

Is comp. 100110
+ 1
2s comp. 100111

~~Is 100110~~
~~for 2s +~~ ~~1~~
~~100111~~

101010
- 011001
10001

$(42)_{10}$ 101010
$+(-25)_{10}$ + 100111
1 010001

result is: $(010001)_2$

* Binary Coding :-

↳ Numerical data (0,1..9) is not the only form of data which is handled by the computer. Alphanumeric data (A, B, C .. Z) and some special characters such as =, −, +, * - also required.

* BCD →

↳ Binary coded Decimal is a method of using Binary digits to represent the decimal digits 0-9.

↳ A decimal digit is represented by Four Binary Digits.

↳ The BCD coding is the binary equivalent of the decimal digit.

$(5319)_{10} = (?)_{BCD}$

5 3 19 = 0101 0011 0001 1001

Devim

* ASCII - 8 bit "code"

" American Standard Code for Information Interchange"

ASCII-7 This code was originally designed by as a seven bit code. ASCII - 2^7 - 128 characters codes.

Later on IBM developed a new version of ASCII called ASCII-8, which made use of all eight bits providading 2^8 - 256 symbols.

↳ ASCII coding contains upper case, Lower case, 0-9.

Ex: 01110111 01101111

w o r d s

119 111 114 100 115

0111 0111 01101111 01110010

w 119 o 111 r 114

01100110 01110011

d 100 s 115

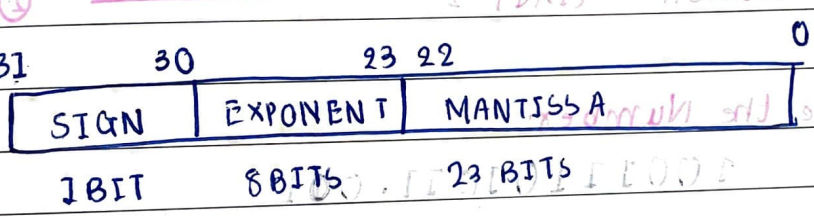
* Floating Point representation:
  - (a) Mantissa
  - (b) Base
  - (c) Exponent

	Number	Base	Exponent
4×10^6	04	10	06
1160×2^8	1160 110	2	08
1537.379	1537379	10	-3

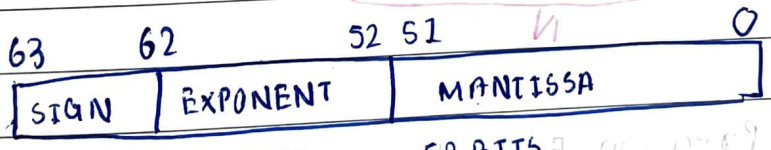
Example:

IEEE 754 Floating point number representation.

(a) Single Precision Format



(b) Double Precision Format



* Represent $(1259.125)_{10}$ in single and double precision format

4 ① Convert Decimal to Binary.

$(1259)_{10} = (10011101011)_2$

$(0.125)_{10}$ =

$0.125 \times 2 = 0.250$	0	
$0.250 \times 2 = 0.500$	0	
$0.50 \times 2 = 1.000$	1	$= (001)_2$

$1259.125 = (10011101011.001)_2$

② Normalize the Number

Single Precision: $(1.N)\ 2^{E-127}$ ——— ①

Double Precision: $(1.N)\ 2^{E-1023}$ ——— ②

Normalize the Number:

10011101011.001

$\therefore 1.00111010111001 \times 2^{10}$

N

③ Single Precision Format

$(1.N)\ 2^{E-127}$ 128 Bit

Normalized Number: $1.00111010111001 \times 2^{10}$ compare with eq² ①

$E - 127 = 10$

$E = 137$

Convert in Binary Numbers...

$(137)_{10} = (1000\ 1001)_2$

31 S	30 E	23 22 MANTISSA 0
0	10001001	00111010110011..00...
1 Bit	8 Bits	23 Bits

Number is +ve so put 0, otherwise 1

Exponent in Binary

Fraction Part

④ Double Precision Format

$$(1.N)_2^{E-1023}$$

Normalized the Number : $1.00111010110011 \times 2^{10}$

Compare with eq^n ②.

$E - 1023 = 10$

$E = 1033$

Convert in Binary No.

$(1033)_2 = (10000001001)_2$

S	E	M
0	10000001001	0.00111010110011..00..0
1 Bit	11 Bits	52 Bits