

CSARCH Lecture Series: Binary Floating-Point format for Double Precision

Sensei RL Uy
College of Computer Studies
De La Salle University
Manila, Philippines



Copyright Notice

This lecture contains copyrighted materials and is use solely for instructional purposes only, and not for redistribution.

Do not edit, alter, transform, republish or distribute the contents without obtaining express written permission from the author.

Overview

Reflect on the following questions:

- How is double-precision floating-point data stored in the memory?
- Given the code below, how are double-float data stored in the memory?

```
int main()
{
    double var, var1;
    var = 2.5;
    var1 = -1.28e2;
}
```

Overview

- How is double-precision floating-point data stored in the memory?
- This sub-module introduces the IEEE-754 double-precision floating-point format
- The objective is as follows:
 - ✓ Describe the process of representing double-precision floating-point data using IEEE-754 standard

IEEE-754 double-precision floating-point format

Sign	Exponent representation	Fraction part of significand
1	11	52
s	e'	f

normalized to this format
before representation:

$$1.f \times 2^e$$

- IEEE-754 single-precision floating-point format is 64-bit in width
- The 64-bit is partitioned as 1 bit for sign bit; 11 bits for exponent representation and 52 bits for the fractional part of the significand
 - Significand in binary
 - Base 2
 - sign bit: 0 → positive; 1 → negative
 - $e' = e + 1023$
 - significand normalized to 1.f (implied 1)

IEEE-754 double-precision floating-point format

Example: $+1.00111_2 \times 2^5$

normalized format: (same) $+1.00111_2 \times 2^5$

Significand in binary?	Yes
Base-2?	Yes
Normalized?	Yes
Sign bit	0
$e' = e+1023$	$5+1023=1028$ [100 0000 0100]

for brevity, can be written as 0011 10...0

Answer:

Sign	Exponent representation	Fraction part of significand
0	100 0000 0100	0011 1000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

Hex: 0x4043800000000000

IEEE-754 double-precision floating-point format

Example: $-100.111_2 \times 2^{-7}$

normalized format: $-1.00111_2 \times 2^{-5}$

Significand in binary?	Yes
Base-2?	Yes
Normalized?	No
Sign bit	1
$e' = e+127$	$-5+1023=1018$ [011 1111 1010]

Answer:

Sign	Exponent representation	Fraction part of significand
1	011 1111 1010	0011 10...0

Hex: 0xBFA3800000000000

IEEE-754 double-precision floating-point format

Example: $-0.000100111_2 \times 2^{15}$

normalized format: $-1.00111_2 \times 2^{11}$

Significand in binary?	Yes
Base-2?	Yes
Normalized?	No
Sign bit	1
$e' = e+127$	$11+1023=1034$ [100 0000 1010]

Answer:

Sign	Exponent representation	Fraction part of significand
1	100 0000 1010	001110...0

Hex: 0xC0A3800000000000

IEEE-754 double-precision floating-point format

Example: +4.0 +100.0₂ × 2⁰

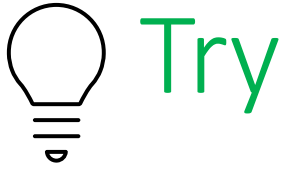
normalized format: +1.000₂ × 2²

Significand in binary?	No
Base-2?	No
Normalized?	No
Sign bit	0
e' = e+127	2+1023=1025 [1000 000 0001]

Answer:

Sign	Exponent representation	Fraction part of significand
0	1000 0001	0...0

Hex: 0x4010000000000000



label	Address (hex)	Memory data (hex)
var1	0010	
var	0000	

```
int main()
{
    double var, var1;
    var = 2.5;
    var1 = -1.28e2;
}
```



label	Address (hex)	Memory data (hex)
var1	0010	C060 0000 0000 0000
var	0000	4004 0000 0000 0000

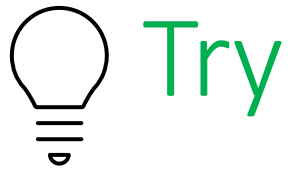
```
int main()
{
    double var, var1;
    var = 2.5;
    var1 = -1.28e2;
}
```

$$+2.5 = 10.1_2 \times 2^0 = 1.01_2 \times 2^1$$

Sign	Exponent representation	Fraction part of significand
0	100 0000 0000	010...0

$$-1.28e2 = -128.0 = 100000000.0_2 \times 2^0 = 1.0_2 \times 2^7$$

Sign	Exponent representation	Fraction part of significand
1	100 0000 0110	0...0



IEEE-754 double-precision Floating-point	Decimal equivalent
0xC00C000000000000	
0x400E000000000000	



IEEE-754 double-precision Floating-point	Decimal equivalent
0xC00C000000000000	-3.5
0x400E000000000000	+3.75

Sign	Exponent representation	Fraction part of significand
1	10000000000	110...0

$$e = e' - 127$$

$$= 1024 - 1023$$

$$= 1$$

significand: 1.110

implied 1

Answer:

$$-1.75 \times 2^1$$

$$= -3.5$$

Sign	Exponent representation	Fraction part of significand
0	10000000000	1110...0

$$e = e' - 1023$$

$$= 1024 - 1023$$

$$= 1$$

significand: 1.111

Answer:

$$+1.875 \times 2^1$$

$$= +3.75$$

To recall ...

- What have we learned:
 - ✓ Describe the process of representing double-precision floating-point data using IEEE-754 standard