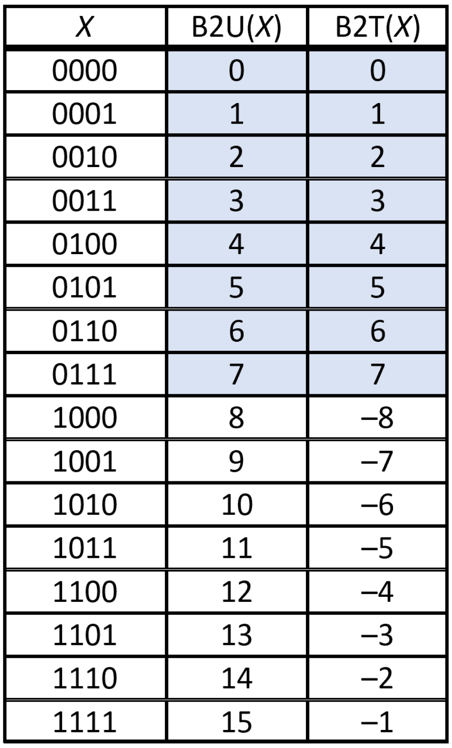
Encoding Integers

Signed & Unsigned

B2U (Unsigned)

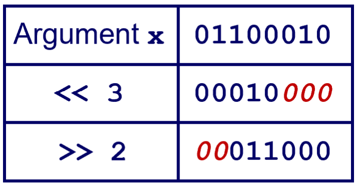
B2T (Signed)



X	B2U(*X*)	B2T(*X*)
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	–8
1001	9	–7
1010	10	–6
1011	11	–5
1100	12	–4
1101	13	–3
1110	14	–2
1111	15	–1

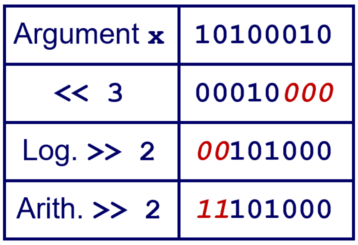
Shift Operations

Left Shift: x << y



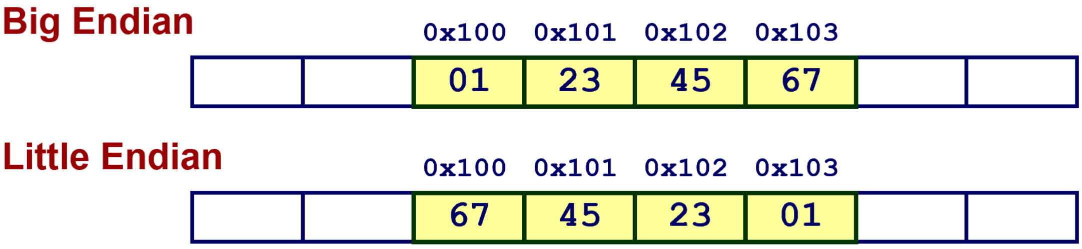
Argument **x**	01100010
<< 3	00010*000*
>> 2	*00*011000

Right Shift: x >> y



Argument x	10100010
<< 3	00010*000*
Log. >> 2	*00*101000
Arith. >> 2	*11*101000

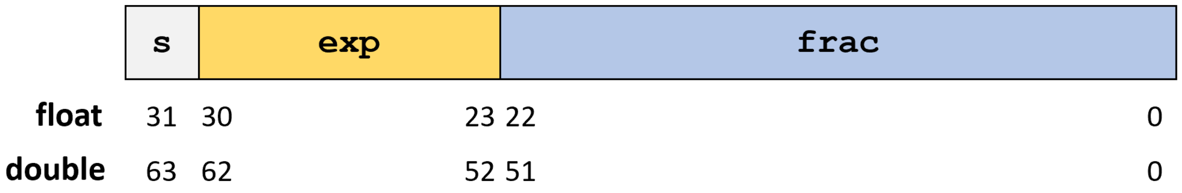
Bytes Ordering



IEEE Floating Point

Floating Point Form

Floating Point Representation



s | exp | frac

float 31 30 23 22 0

double 63 62 52 51 0

- Numerical form

$$v = (-1)^s\, M\ 2^E$$

  - **Sign bit s** determines whether number is negative or positive
  - **Significand M** is normally a fractional value in range [1.0,2.0).
  - **Exponent E** weights value by power of two

- Encoding
  - Most Significant Bit s is sign bit **s**
  - exp field encodes **E** (but is not equal to E)
  - frac field encodes **M** (but is not equal to M)

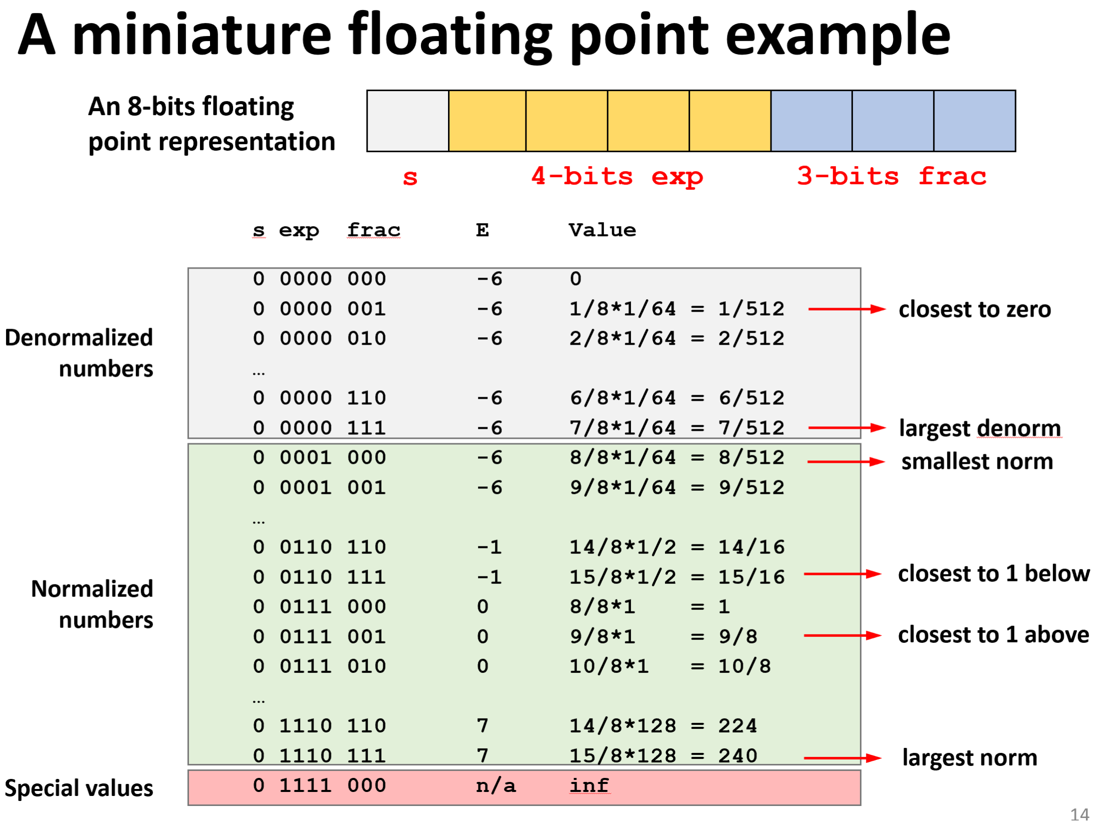
Bias = 2 ^ [k-1]-1 (k is exp bits)

exp = E + Bias ==> to base (2)

Normalized & Denormalized Values

NORMALIZED	DENORMALIZED
exp != 000…0 / 111…1	exp = 000…0
E = exp - Bias	E = 1 - Bias
M = 1. xxx…x	M = 0.xxx…x

Example:



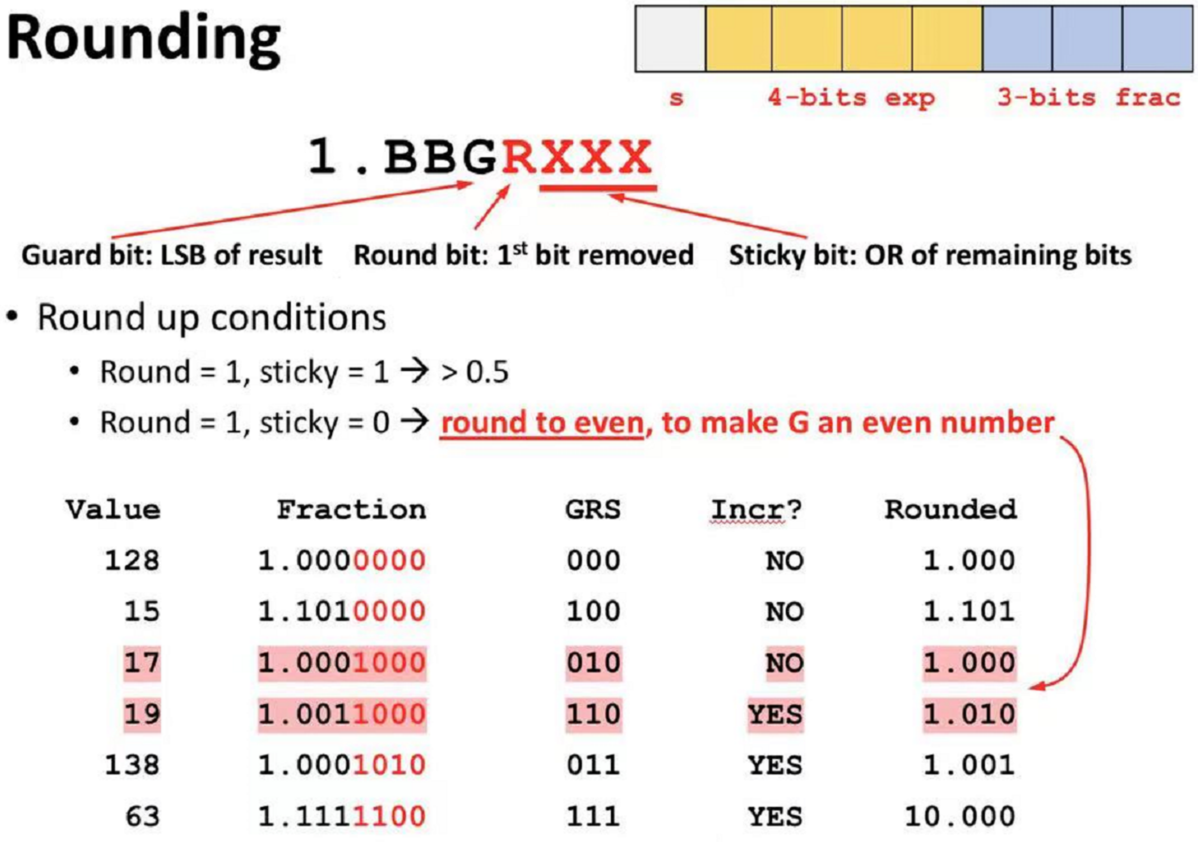
A miniature floating point example

An 8-bits floating point representation

s 4-bits exp 3-bits frac

	s exp frac	E	Value	
Denormalized numbers	0 0000 000	-6	0	
	0 0000 001	-6	1/8*1/64 = 1/512	closest to zero
	0 0000 010	-6	2/8*1/64 = 2/512	
	…			
	0 0000 110	-6	6/8*1/64 = 6/512	
	0 0000 111	-6	7/8*1/64 = 7/512	largest denorm
Normalized numbers	0 0001 000	-6	8/8*1/64 = 8/512	smallest norm
	0 0001 001	-6	9/8*1/64 = 9/512	
	…			
	0 0110 110	-1	14/8*1/2 = 14/16	
	0 0110 111	-1	15/8*1/2 = 15/16	closest to 1 below
	0 0111 000	0	8/8*1 = 1	
	0 0111 001	0	9/8*1 = 9/8	closest to 1 above
	0 0111 010	0	10/8*1 = 10/8	
	…			
	0 1110 110	7	14/8*128 = 224	
	0 1110 111	7	15/8*128 = 240	largest norm
Special values	0 1111 000	n/a	inf	

Rounding



Rounding

s 4-bits exp 3-bits frac

1.BBGRXXX

Guard bit: LSB of result Round bit: 1st bit removed Sticky bit: OR of remaining bits

- Round up conditions
  - Round = 1, sticky = 1 → > 0.5
  - Round = 1, sticky = 0 → round to even, to make G an even number

Value	Fraction	GRS	Incr?	Rounded
128	1.0000000	000	NO	1.000
15	1.1010000	100	NO	1.101
17	1.0001000	010	NO	1.000
19	1.0011000	110	YES	1.010
138	1.0001010	011	YES	1.001
63	1.1111100	111	YES	10.000