

### Lecture 3 normal case.

$$\begin{array}{r} 1 \ 1 \ 0 \ 1 \\ 0 \ 1 \ 0 \ 1 \\ \hline 1 \ 0 \ 0 \ 1 \ 0 \end{array} \quad \begin{array}{l} -3 \\ 1 \\ 2 \end{array}$$

Complementary overflow:

$$\begin{array}{r} 1 \ 1 \ 0 \ 1 \\ 1 \ 0 \ 1 \ 0 \\ \hline \end{array} \quad \begin{array}{l} -3 \\ -6 \end{array}$$

$$\begin{array}{r} \text{X} \ 0 \ 1 \ 1 \ 1 \\ \hline \end{array} \quad \begin{array}{l} 7 \end{array}$$

negative overflow

$$\begin{array}{r} 0 \ 1 \ 1 \ 1 \\ 0 \ 1 \ 0 \ 1 \\ \hline \end{array} \quad \begin{array}{l} 7 \\ 3 \end{array}$$

positive overflow

$$\begin{array}{r} 1 \ 1 \ 0 \ 0 \\ \hline \end{array} \quad \begin{array}{l} -4 \end{array}$$

Wraps Around

① if  $\text{sum} \geq 2^{w-1}$   $\rightarrow$  Become negative 2) At most once

② if  $\text{sum} < -2^{w-1}$   $\rightarrow$  Becomes positive 3) At most once

Standard Multi Function: ignore high order  $w$  bits

Implements:  $\text{UMult}_w(u, v) = u \cdot v \bmod 2^w$ .

$$5 \times 5 = 25 \quad \boxed{0001 \mid 1001} = 9 = 25 \bmod 16$$

But when in the signed number  $9 \rightarrow -7$

$$\begin{array}{c} \text{Complementary} \rightarrow \begin{array}{r} 1 \ 1 \ 0 \ 1 \quad \text{signed} \quad -3 \\ 1 \ 1 \ 1 \ 0 \quad \text{unsigned} \quad 14 \end{array} \Rightarrow \begin{array}{c} \text{while} \\ \boxed{\phantom{000} \mid 0 \ 1 \ 1 \ 0} \\ \text{lower bits} \end{array} \quad \begin{array}{l} 13 \times 14 = 182 \\ -3 + (-2) = -5 \end{array}$$

while we can have  $x = \sum_{i=0}^{w-1} x_i 2^i$

$$\text{and when } x \ll 2 \Rightarrow x' = \sum_{i=0}^{w-1} x_i 2^{i+2} = 4x$$

in the reverse,  $u \gg k \Leftrightarrow \lfloor u / 2^k \rfloor$  read it to 0.

for example:  $0110 \ 6 \Rightarrow \gg 1 \ 0011 \ 3 \Rightarrow 0001 \ 1 \leftarrow \text{unsigned}$

But when using the complementary number if the number is negative.

$$\text{eg: } 1010 \ -6 \Rightarrow \gg 1 \ 1101 \ -3 \Rightarrow \gg 1 \ 1110 \ -2$$

Copy the sign value to the empty space. the device need 30 T.

$X \rightarrow -X$  all the bits are flipped, and add one in the tail

after this, all the bits are reversed and add one to it. we will get  $X$  again.

When using variable in the for-loop, you should guarantee it will not cause memory error.

the max memory address is  $2^4$  bits.

$$\text{and } 2^{10} \approx 10^3 \Rightarrow 2^{40} \approx 10^{12} \quad 2^7 \approx 128 \Rightarrow 2^{47} \approx 1.28 \times 10^{14}$$

Addr=0	Addr=4	Addr=8	Addr=12	...	32 bit
addr=0	addr=8	...			64 bit

## Byte Ordering

Conventions:

1) Big Endian: Sun, PPC Mac. Least significant byte has highest address.

2) Little Endian: x86, ARM, zos, and windows.

For eg:  $X(\text{variable}) = 0x01234567$  &  $X = 0x100$

Big

	0x100	0x101	0x102	0x103		
	01	23	45	67		

Little

	0x100	0x101	0x102	0x103		
	67	45	23	01		

## Puzzle

$$(X) \quad X < 0 \quad \nRightarrow \quad (X * 2) < 0$$

$$(X) \quad X \geq 0 \quad \Rightarrow \quad X \text{ is a number } \geq 0$$

$$(X) \quad X \& 7 == 7 \quad \Rightarrow \quad (X \ll 3) < 0$$

$$(X) \quad X > -1 \quad (\text{is never true})$$

$$(X) \quad X > Y \quad \stackrel{X}{\Rightarrow} \quad -X < -Y$$

$$(X) \quad X * X \geq 0$$

$$(X) \quad X > 0 \ \& \ y > 0 \Rightarrow X + y > 0$$

$$\checkmark \quad X \geq 0 \quad \Rightarrow \quad -X \leq 0$$

$$(X) \quad X \leq 0 \quad \Rightarrow \quad -X \geq 0 \quad (\text{if the width is 8 and it will represent } -128 \sim 127 \text{ and } -128 \text{ is the specific one})$$

$$\checkmark \quad (X) \quad -X \gg 7 == -1$$

# Lecture 4 Floating point

## Fractional Binary Number

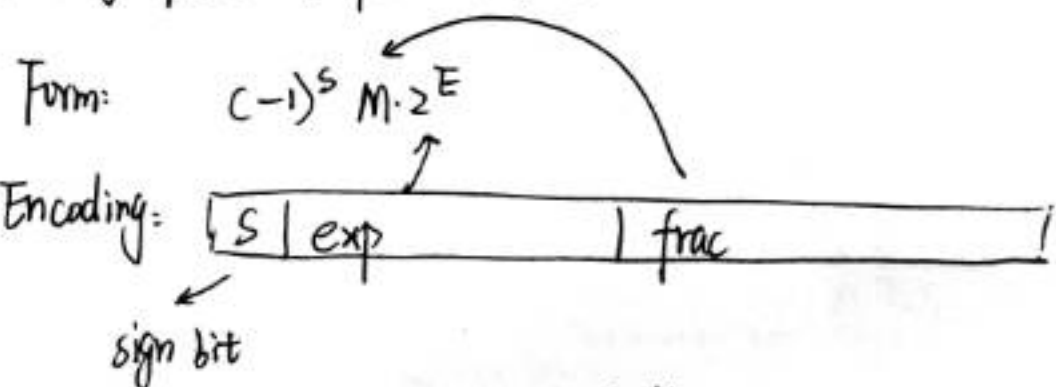
Value	Representations
5 3/4	$101.11_2$
2 7/8	$10.111_2$
1 7/8	$1.0111_2$

1) Divide by 2 by shifting right (unsigned)

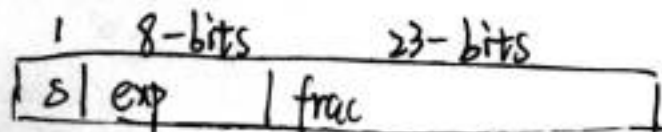
2) Numbers of form  $0.1111..._2$  are just below 1.0

We may use notation  $1.0 - \epsilon$

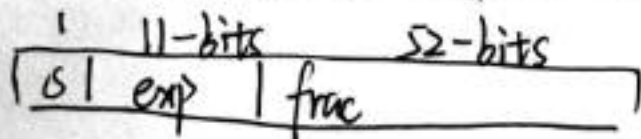
## Floating point Representation



1) Single precision: 32 bits



2) Double precision: 64 bits



"Normalized" value

$\text{exp} \neq 000...000$  and  $\text{exp} \neq 111...111$

$E = \text{Exp} - \text{Bias}$ . while Exp: unsigned value of exp field

$\text{Bias} = 2^{k-1} - 1$  where  $k$  is number of exponent bits.

1) single precision: 127 (Exp: 1...254, E: -126...127)

2) Double precision: 1023 (Exp: 1...2046, E: -1022...1023)

$M = 1.xxx...x_2$  while  $xxx...x$ : bits of frac field

Minimum when  $\text{frac} = 000...0$  ( $M = 1.0$ ) Max when  $\text{frac} = 111...1$  ( $M = 2.0 - \epsilon$ )

eg.  $F = 15213_{10} \Rightarrow 15213_{10} = 1.1101101101101_2 \times 2^{13}$

$M = 1.1101101101101$   $E = 13$   $\text{Bias} = 127$   $\text{Exp} = 140 = 10001100$

Result:  $\frac{0}{s} \quad \frac{10001100}{\text{exp}} \quad \frac{11011011011010000000}{\text{frac}}$

Summary:

$0 \leq \text{Exp} \leq 255$ ,  $-127 \leq E \leq 128$

Special Values

Condition:  $\text{exp} = 111...1$

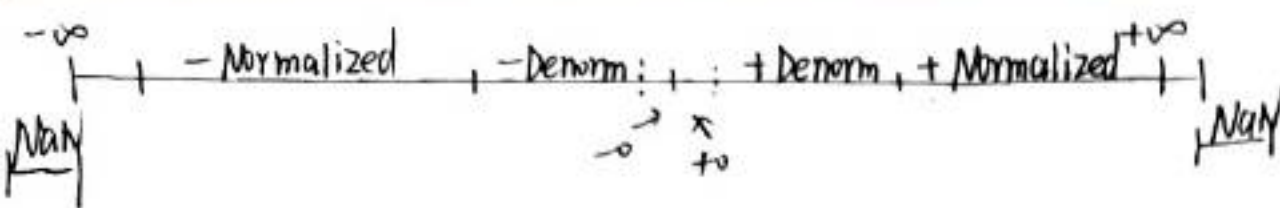
1)  $\text{frac} = 000...0 \rightarrow$  represent infinity: Both positive and negative

2)  $\text{frac} \neq 000...0 \rightarrow$  Not-a-Number (NaN)

Condition:  $\text{exp} = 000...0 \rightarrow$  Denormalized value

1)  $\text{frac} = 000...0 \rightarrow$  represent zero. 2)  $\text{frac} \neq 000...0$  Numbers chosen to 0.0





## Fp Multiplication

$$(-1)^{s_1} M_1 2^{E_1} \times (-1)^{s_2} M_2 2^{E_2}$$

Exact result:  $(-1)^s M 2^E$

①  $s = s_1 \wedge s_2$     ②  $M = M_1 \times M_2$     ③  $E = E_1 + E_2$

if  $M \geq 2$  shift  $M$  right, increment  $E$

Puzzles.

①  $x == (\text{int})(\text{float}) x$  (X)

②  $x == (\text{int})(\text{double}) x$  (V)

③  $f == (\text{float})(\text{double}) f$  (V)

④  $d == (\text{double})(\text{float}) d$  (X)

⑤  $f == -(f)$  (V)

⑥  $2/3 == 2/3.0$  (X)

⑦  $d < 0.0 \Rightarrow ((d * 2) < 0.0)$  (V)

⑧  $d > f \Rightarrow -f > -d$  (V)

⑨  $d * d \geq 0.0$  (X)

⑩  $(d + f) - d == f$  (X)

## Rounding

Nearest even (default)

$1.40 \rightarrow 1$      $1.60 \rightarrow 2$      $1.50 \rightarrow 2$      $2.50 \rightarrow 2$

$-1.50 \rightarrow -2$

when exactly halfway between two possible values

Round so that least significant digit is even.

Rounding Binary Numbers

odd is 1, even is 0.

eg. value	Binary	Rounded	(bits right of binary point) Action	Rounded value
$2 \frac{7}{32}$	10.00011	10.00	$< \frac{1}{2}$ - down	2
$2 \frac{3}{16}$	10.00110	10.01	$> \frac{1}{2}$ - up	$2 \frac{1}{4}$
$2 \frac{7}{8}$	10.11100	11.00	$\frac{1}{2}$ - up	3
$2 \frac{5}{8}$	10.10100	10.10	$\frac{1}{2}$ - down	$2 \frac{1}{2}$