Computer Organization and Architecture

3 Integer, Floating-point and Decimal Representation

Tongwei Ren

Sep. 10, 2018

南京大學
NANJING UNIVERSITY

Review

- A top-level view of computer
- Computer component
  - Memory: cache, memory hierarchy
  - I/O: buffer
  - CPU: interrupt
  - Bus: type

Operation In Program

- Negation: $y = -x$
- Addition: $x = 9 + -8$
- Subtraction: $x = 5 - 3$
- Multiplication: $x = 2 * 5$
- Division: $x = 7 / 9$
- ……

How to represent the numbers?

How to do the operations?

Binary Representation

Binary Representation

- 为了表示出多个数值，必须对多个位进行组合
  - 如果有k位，最多能区分出2^k个不同的值
- 整数类型
  - 无符号整数
  - 有符号整数：原码，反码，补码
    - 原码和反码在进行加法运算时都会造成不必要的硬件需求，于是就出现了补码
    - 二进制补码的运算
    - 二进制-十进制转换

Integer Representation

- Complement representation vs. sign magnitude representation

	complement		sign magnitude	
9	0000 1001		0000 1001	
+ 8	+ 0000 1000		+ 0000 1000	
17	0001 0001	17	0001 0001	17
9	0000 1001		0000 1001	
+ -8	+ 1111 1000		+ 1000 1000	
1	10000 0001	1	1001 0001	-17

Integer Representation

- Value of complement representation

$$[X]_C = X_n X_{n-1} \dots X_2 X_1$$
$$X = -X_n \times 2^{n-1} + \cdots + X_2 \times 2^1 + X_1 \times 2^0$$

−128	64	32	16	8	4	2	1	
1	0	0	0	0	0	1	1	
−128						+2	+1	= −125

- Value range

$$-2^{n-1} \leq X \leq 2^{n-1} - 1$$

Floating-point Representation

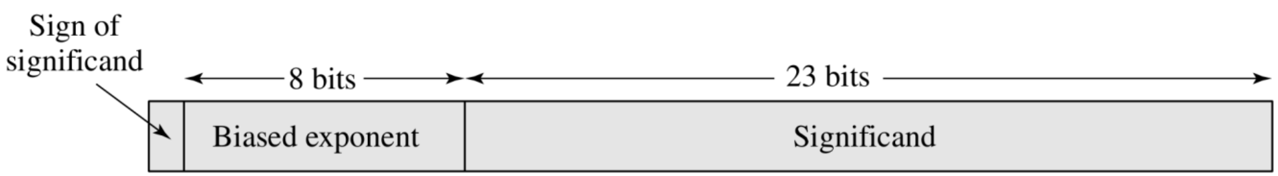
- Real number representation
- The value range of **fixed-point representation** is limited
- Scientific notation

$$\pm S \times B^{\pm E}$$

  - ±(sign): plus or minus
  - S (significand)
  - B (base): implicit and need not be stored because it is the same for all numbers
  - E (exponent)

Floating-point Representation (cont.)

- Real number representation (cont.)



(a) Format

$$
\begin{array}{rcl}
1.1010001 \times 2^{10100} & = 0\ 10010011\ 10100010000000000000000 = & 1.6328125 \times 2^{20} \\
-1.1010001 \times 2^{10100} & = 1\ 10010011\ 10100010000000000000000 = & -1.6328125 \times 2^{20} \\
1.1010001 \times 2^{-10100} & = 0\ 01101011\ 10100010000000000000000 = & 1.6328125 \times 2^{-20} \\
-1.1010001 \times 2^{-10100} & = 1\ 01101011\ 10100010000000000000000 = & -1.6328125 \times 2^{-20}
\end{array}
$$

(b) Examples

Normalized Number

- Any floating-point number can be expressed in many ways

$$0.110 \times 2^5, 110 \times 2^2, 0.0110 \times 2^6$$

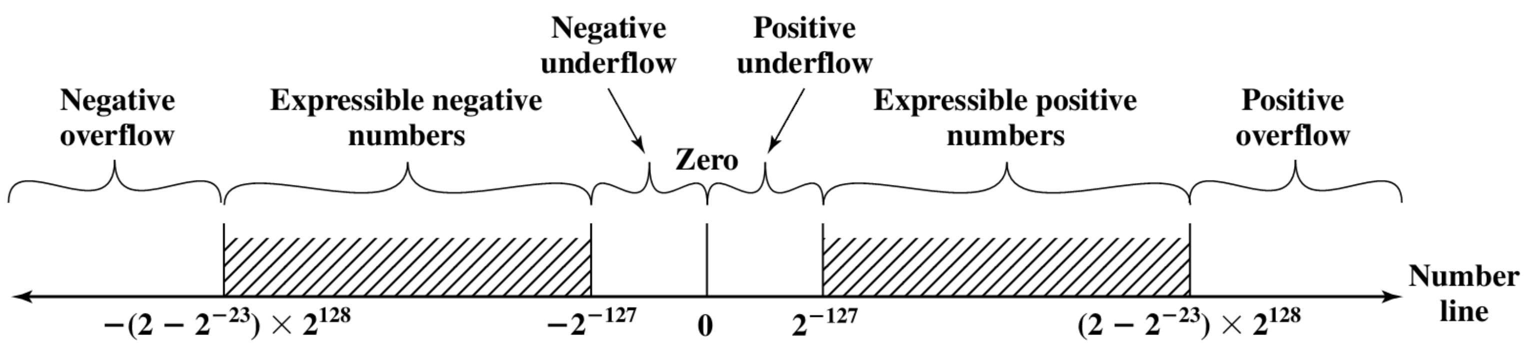
- Normalized representation

$$\pm 1.bbb \ldots b \times 2^{\pm E}$$

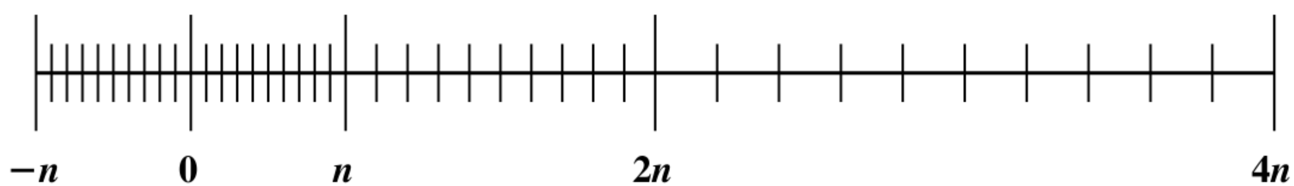
  - The sign is stored in the first bit of the word
  - The first bit of the true significand is always 1 and need not be stored in the significand field
  - The value 127 is added to the true exponent to be stored in the exponent field
  - The base is 2

Normalized Number (cont.)

- Value range
  - Negative numbers between $-(2-2^{-23}) \times 2^{128}$ and -2^{-127}
  - Positive numbers between 2^{-127} and $(2-2^{-23}) \times 2^{128}$



(b) Floating-point numbers



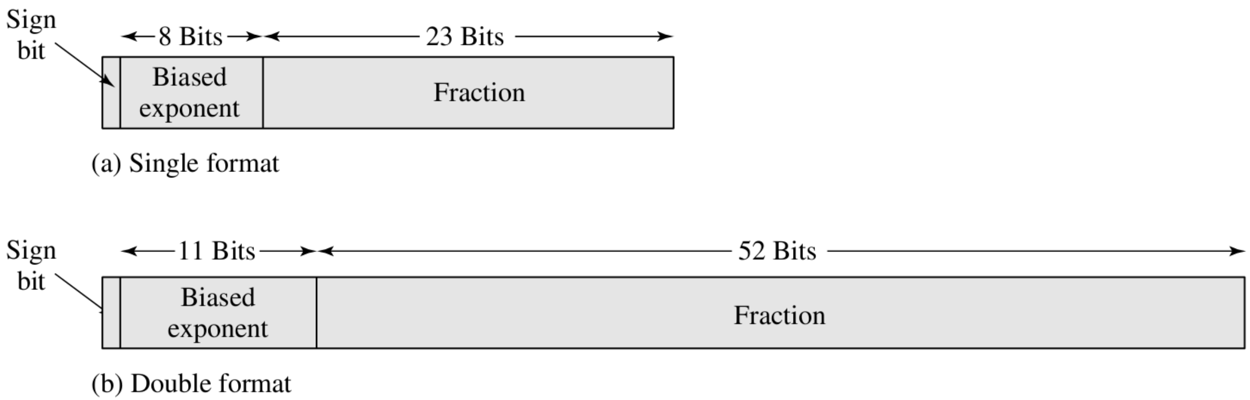
[袁睿, 131250088]

Normalized Number (cont.)

- There is a trade-off between range and precision
  - Increase number of exponent bits: expand the range of expressible numbers, but reduce number precision
  - Increase number of significand bits: increase number precision, but reduce the range of expressible numbers
- Using a larger base
  - Achieve a greater range for the same number of exponent bits, but less precision

IEEE Standard 754

- Define both a 32-bit single and a 64-bit double format



- Define two extended formats
  - Include additional bits in the exponent (extended range) and in the significand (extended precision)
  - Lessen the chance of excessive round off error and intermediate overflow

IEEE Standard 754 (cont.)

- Format parameters

Parameter	Format			
	Single	Single Extended	Double	Double Extended
Word width (bits)	32	≥ 43	64	≥ 79
Exponent width (bits)	8	≥ 11	11	≥ 15
Exponent bias	127	unspecified	1023	unspecified
Maximum exponent	127	≥ 1023	1023	≥ 16383
Minimum exponent	-126	≤ -1022	-1022	≤ -16382
Number range (base 10)	$10^{-38}, 10^{+38}$	unspecified	$10^{-308}, 10^{+308}$	unspecified
Significand width (bits)*	23	≥ 31	52	≥ 63
Number of exponents	254	unspecified	2046	unspecified
Number of fractions	2^{23}	unspecified	2^{52}	unspecified
Number of values	1.98×2^{31}	unspecified	1.99×2^{63}	unspecified

IEEE Standard 754 (cont.)

- Interpretation

	Single Precision (32 bits)				Double Precision (64 bits)			
	Sign	**Biased exponent**	**Fraction**	**Value**	**Sign**	**Biased exponent**	**Fraction**	**Value**
positive zero	0	0	0	0	0	0	0	0
negative zero	1	0	0	-0	1	0	0	-0
plus infinity	0	255 (all 1s)	0	∞	0	2047 (all 1s)	0	∞
minus infinity	1	255 (all 1s)	0	$-\infty$	1	2047 (all 1s)	0	$-\infty$
quiet NaN	0 or 1	255 (all 1s)	$\neq 0$	NaN	0 or 1	2047 (all 1s)	$\neq 0$	NaN
signaling NaN	0 or 1	255 (all 1s)	$\neq 0$	NaN	0 or 1	2047 (all 1s)	$\neq 0$	NaN
positive normalized nonzero	0	$0 < e < 255$	f	$2^{e-127}(1.f)$	0	$0 < e < 2047$	f	$2^{e-1023}(1.f)$
negative normalized nonzero	1	$0 < e < 255$	f	$-2^{e-127}(1.f)$	1	$0 < e < 2047$	f	$-2^{e-1023}(1.f)$
positive denormalized	0	0	$f \neq 0$	$2^{e-126}(0.f)$	0	0	$f \neq 0$	$2^{e-1022}(0.f)$
negative denormalized	1	0	$f \neq 0$	$-2^{e-126}(0.f)$	1	0	$f \neq 0$	$-2^{e-1022}(0.f)$

IEEE Standard 754 (cont.)

- Example

```
0.5 = 0.100…0B = (1.00..0)2×2^(-1)

0 01111110 000…00 (23)


-0.4375 = -0.01110…0B = - (1.110..0)2×2^(-2)

1 01111101 110…00 (21)
```

Decimal Representation

- Problem of floating-point arithmetic
  - Limitation in precision
  - High cost in conversion
- Application requirement
  - Calculation of long numerical string: accountancy, …
- Solution
  - Represent 0, 1, …, 9 with four-bits **Binary-Coded Decimal** (BCD) , and calculate directly

Decimal Representation (cont.)

- Natural Binary Coded Decimal (NBCD, 8421 code)
  - 0 ~ 9: 0000 ~ 1001
  - Sign: Use four most significant bits
    - Positive: 1100 / 0
    - Negative: 1101 / 1
  - Examples
    - +2039: **1100** 0010 0000 0011 1001 / **0** 0010 0000 0011 1001
    - -1265: **1101** 0001 0010 0110 0101 / **1** 0001 0010 0110 0101
- Other Binary Coded Decimal
  - 2421, 5211, 4311, …

Summary

- Integer representation
- Floating-point Representation
- Decimal representation

Thank You

rentw@nju.edu.en

南京大學

NANJING UNIVERSITY