计算机组成原理

第 4 讲

左德承

哈尔滨工业大学计算学部

容错与移动计算研究中心

二、浮点表示

$N = S \times r^j$ 浮点数的一般形式

S 尾数 j 阶码 r 基数（基值）

计算机中 r 取 2、4、8、16 等

当 $r = 2$ $N = 11.0101$

二进制表示

✓ $= 0.110101 \times 2^{\boxed{10}}$ 规格化数

$= 1.10101 \times 2^{1}$

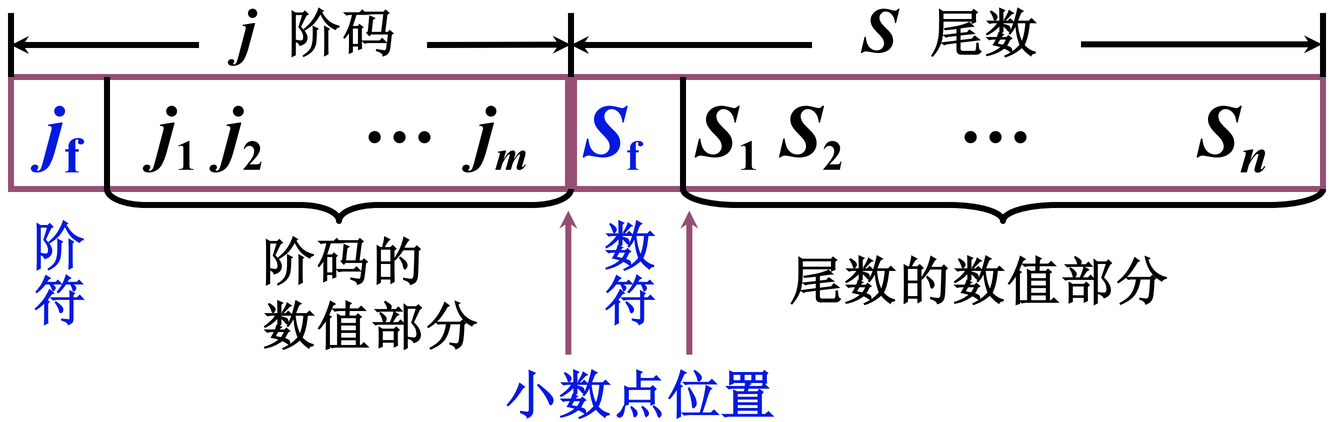
$= 1101.01 \times 2^{-10}$

✓ $= 0.00110101 \times 2^{100}$

计算机中 S 小数、可正可负

j 整数、可正可负

1. 浮点数的表示形式

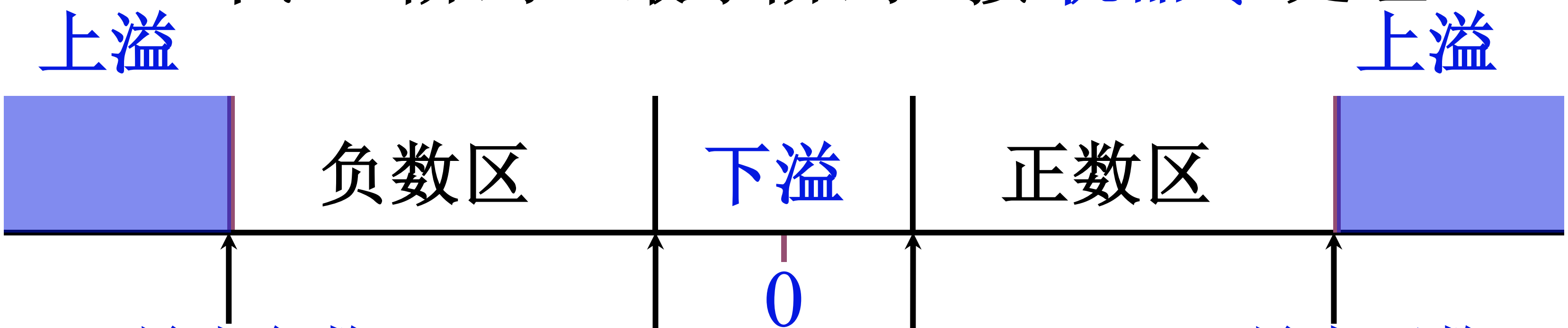


S_f	代表浮点数的符号
n	其位数反映浮点数的精度
m	其位数反映浮点数的表示范围
j_f 和 m	共同表示小数点的实际位置

2. 浮点数的表示范围

上溢　阶码 $>$ 最大阶码

下溢　阶码 $<$ 最小阶码　按 机器零 处理



最小负数

$$-2^{(2^m-1)}\times(1-2^{-n})$$

$$-2^{15}\times(1-2^{-10})$$

最大负数

$$-2^{-(2^m-1)}\times 2^{-n}$$

$$-2^{-15}\times 2^{-10}$$

最小正数

$$2^{-(2^m-1)}\times 2^{-n}$$

$$2^{-15}\times 2^{-10}$$

最大正数

$$2^{(2^m-1)}\times(1-2^{-n})$$

$$2^{15}\times(1-2^{-10})$$

设 $m=4$

$n=10$

练习 2.2

设机器数字长为 24 位，欲表示±3万的十进制数，试问在保证数的最大精度的前提下，除阶符、数符各取1 位外，阶码、尾数各取几位？

解： $\because \quad 2^{14} = 16384 \qquad 2^{15} = 32768$

$\therefore$ 如果是定点数15 位二进制数可反映 ±3 万之间的十进制数

$$2^{\boxed{15}} \times 0.\underbrace{\times\times\times \cdots \times\times\times}_{n\text{位}}$$

$$m = 4, \ 5, \ 6, \cdots$$

满足 最大精度 可取 $m = 4, \quad n = 18$

3. 浮点数的规格化形式

$r = 2$　尾数最高位为 1

$r = 4$　尾数最高 2 位不全为 0

$r = 8$　尾数最高 3 位不全为 0

基数不同，浮点数的规格化形式不同

4. 浮点数的规格化

$r = 2$	左规	尾数左移 1 位，阶码减 1
	右规	尾数右移 1 位，阶码加 1
$r = 4$	左规	尾数左移 2 位，阶码减 1
	右规	尾数右移 2 位，阶码加 1
$r = 8$	左规	尾数左移 3 位，阶码减 1
	右规	尾数右移 3 位，阶码加 1

基数 r 越大，可表示的浮点数的范围越大

基数 r 越大，浮点数的精度降低

例如：设 $m=4$，$n=10$，$r=2$

2.2

尾数规格化后的浮点数表示范围

最大正数 $2^{+1111}\times 0.\underbrace{1111111111}_{10\text{个}1} = 2^{15}\times(1-2^{-10})$

最小正数 $2^{-1111}\times 0.1\underbrace{000000000}_{9\text{个}0} = 2^{-15}\times 2^{-1} = 2^{-16}$

最大负数 $2^{-1111}\times(-0.1\underbrace{000000000}_{9\text{个}0}) = -2^{-15}\times 2^{-1} = -2^{-16}$

最小负数 $2^{+1111}\times(-0.\underbrace{1111111111}_{10\text{个}1}) = -2^{15}\times(1-2^{-10})$

三、举例

例 6.13　将 $+\frac{19}{128}$ 写成二进制定点数、浮点数及在定点机和浮点机中的机器数形式。其中数值部分均取 10 位，数符取 1 位，浮点数阶码取 5 位（含1位阶符）。

解：　设 $x = +\frac{19}{128}$

二进制形式　$x = 0.0010011$

定点表示　$x = 0.0010011\,000$

浮点规格化形式　$x = 0.1001100000 \times 2^{-10}$

定点机中　$[x]_{原} = [x]_{补} = [x]_{反} = 0.0010011000$

浮点机中　$[x]_{原} = 1, 0010;\ 0.1001100000$

$[x]_{补} = 1, 1110;\ 0.1001100000$

$[x]_{反} = 1, 1101;\ 0.1001100000$

例 6.14 将 –58 表示成二进制定点数和浮点数，并写出它在定点机和浮点机中的三种机器数及阶码为移码、尾数为补码的形式（其他要求同上例）。

解： 设 $x = -58$

二进制形式 $x = -111010$

定点表示 $x = -0000\,111010$

浮点规格化形式 $x = -(0.1110100000) \times 2^{110}$

定点机中

$[x]_{原} = 1, 0000111010$

$[x]_{补} = 1, 1111000110$

$[x]_{反} = 1, 1111000101$

浮点机中

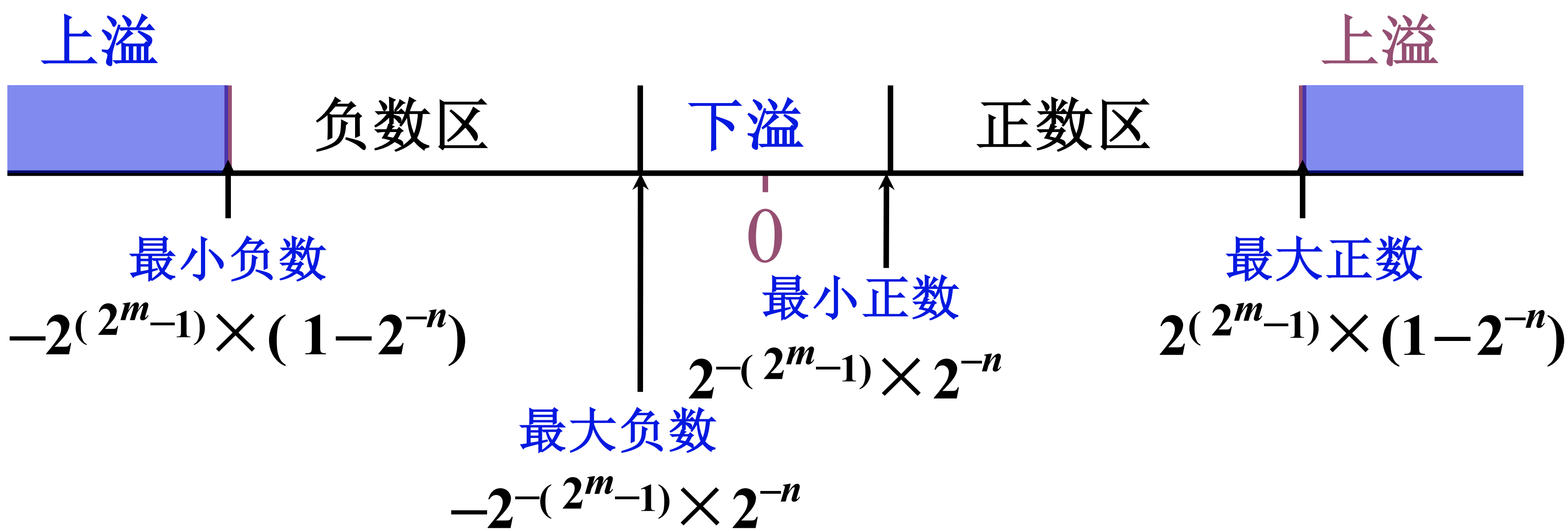
$[x]_{原} = 0, 0110; 1. 1110100000$

$[x]_{补} = 0, 0110; 1. 0001100000$

$[x]_{反} = 0, 0110; 1. 0001011111$

$[x]_{阶移、尾补} = 1, 0110; 1. 0001100000$

例6.15　写出对应下图所示的浮点数的补码形式。设 $n = 10$，$m = 4$，阶符、数符各取 1位。



解：

	真值	补码
最大正数	$2^{15} \times (1-2^{-10})$	0,1111; 0.1111111111
最小正数	$2^{-15} \times 2^{-10}$	1,0001; 0.0000000001
最大负数	$-2^{-15} \times 2^{-10}$	1,0001; 1.1111111111
最小负数	$-2^{15} \times (1-2^{-10})$	0,1111; 1.0000000001

机器零

- 当浮点数 尾数为 0 时，不论其阶码为何值按机器零处理
- 当浮点数 阶码等于或小于它所表示的最小数 时，不论尾数为何值，按机器零处理

如 $m = 4$　　$n = 10$

当阶码和尾数都用补码表示时，机器零为

×, × × × ×；　0. 0 0 … 0

（阶码 = −16） 1, 0 0 0 0 ；　×.× × … ×

当阶码用移码，尾数用补码表示时，机器零为

0, 0 0 0 0;　0. 0 0 … 0

有利于机器中 “ 判 0 ” 电路的实现

四、IEEE 754 标准

S	阶码（含阶符）	尾　　数

数符　　　　小数点位置

尾数为规格化表示

非"0"的有效位最高位为"1"（隐含）

	符号位 *S*	阶码	尾数	总位数
短实数	1	8	23	32
长实数	1	11	52	64
临时实数	1	15	64	80

“Father” of the IEEE 754 standard

- **直到80年代初，各个机器内部的浮点数表示格式还没有统一，因而相互不兼容，机器之间传送数据时，带来麻烦**
- **1970年代后期， IEEE成立委员会着手制定浮点数标准**
- **1985年完成浮点数标准IEEE 754的制定**
- 现在所有计算机都采用IEEE 754来表示浮点数

This standard was primarily the work of one person, UC Berkeley math professor William Kahan.

www.cs.berkeley.edu/~wkahan/ieee754status/754story.html

Prof. William Kahan

摘自“深入理解计算机系统”

IEEE 754 Floating Point Standard

规格化数：+/-1.xxxxxxxxxx$_{two}$ x 2^{Exponent}

Single Precision： (Double Precision is similar)

S	Exponent	Significand
1 bit	8 bits	23 bits

- Sign bit: 1 表示negative ; 0表示 positive
- Exponent（阶码 / 指数）：
  - •SP规格化数阶码范围为0000 0001 (-126) ~ 1111 1110 (127)
  - •bias为127 (single), 1023 (double)

全0和全1用来表示特殊值

为什么用127？若用128, 则阶码范围为多少？

- Significand（尾数）：
  - • 规格化尾数最高位总是1，所以隐含表示，省1位
  - • 1 + 23 bits （ single），1 + 52 bits （double）

0000 0001 (-127) ～ 1111 1110 (126)

SP: $(-1)^S$ x (1 + Significand) x $2^{(Exponent-127)}$

DP: $(-1)^S$ x (1 + Significand) x $2^{(Exponent-1023)}$

摘自“深入理解计算机系统”

Ex: Converting Binary FP to Decimal

BEE00000H is the hex. Rep. Of an IEEE 754 SP FP number

1	0111 1101	110 0000 0000 0000 0000 0000

$$(-1)^S \times (1 + \text{Significand}) \times 2^{(\text{Exponent}-127)}$$

- **Sign: 1 => negative**
- **Exponent:**
  - $0111\ 1101_{two} = 125_{ten}$
  - **Bias adjustment: 125 - 127 = -2**
- **Significand:**

  $1 + 1\text{x}2^{-1} + 1\text{x}2^{-2} + 0\text{x}2^{-3} + 0\text{x}2^{-4} + 0\text{x}2^{-5} + \ldots$

  $=1+2^{-1} +2^{-2} = 1+0.5 +0.25 = 1.75$
- **Represents:** $-1.75_{ten}\text{x}2^{-2} = -0.4375$

摘自“深入理解计算机系统”

Ex: Converting Decimal to FP

-12.75

1. **Denormalize: -12.75**
2. **Convert integer part:**

   $12 = 8 + 4 = 1100_2$
3. **Convert fractional part:**

   $.75 = .5 + .25 = .11_2$
4. **Put parts together and normalize:**

   $1100.11 = 1.10011 \times 2^3$
5. **Convert exponent:** $127 + 3 = 128 + 2 = 1000\ 0010_2$

1	1000 0010	100 1100 0000 0000 0000 0000

The Hex rep. is C14C0000H

摘自“深入理解计算机系统”