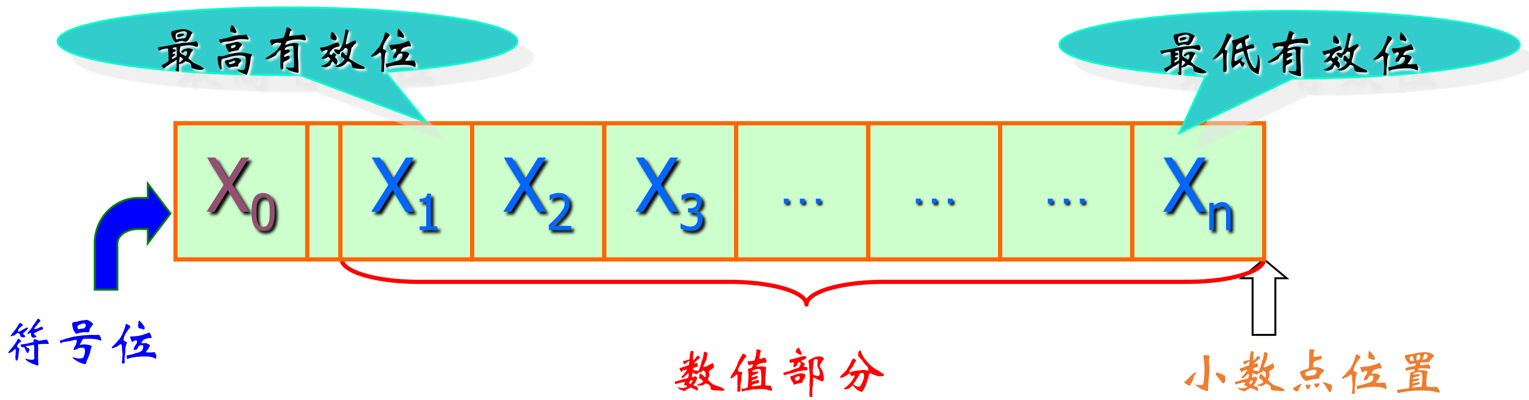
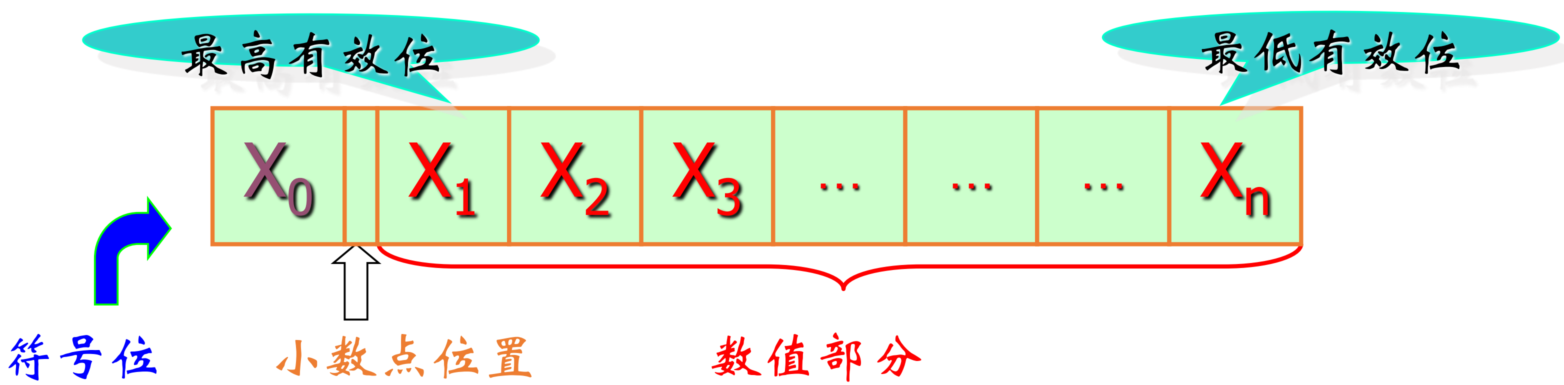
浮点数的编码原理

- 定点整数



最高有效位
最低有效位
X_0 X_1 X_2 X_3 … … … X_n
符号位
数值部分
小数点位置

- 定点小数



最高有效位
最低有效位
X_0 X_1 X_2 X_3 … … … X_n
符号位
小数点位置
数值部分

浮点数

- 十进制：　　　123.45
- 科学计数法：　1.2345×10^2

其中，　1.2345 为尾数，10 为基数，2 为指数

计算机中如何表示?

- 尾数（Mantissa）
- 基数（Base）
- 指数（Exponent）
- 一个表示正负的符号

浮点数的编码

构成：阶码 E，尾数 M，符号位 S，基数 R

S	E	M

- S为符号位，表示该浮点数的正负
- M称为尾数，是一个纯小数
- E是阶数，称为浮点数的指数，是一个整数，
- R为基(比例因子），计算机采用二进制表示浮点数： R=2

所表示数值： $N = (-1)^{S} \times M \times R^{E}$

如何得到浮点数的编码?

- 数据通常既包括整数也包括小数部分。
- 例如： 13.125（十进制）

1. 确定数据的符号位，正数为0，负数为1
2. 转化为二进制 1101.001 （二进制）
3. 将数据按照一定比例因子缩小成定点小数来表示

   例如： $\mathbf{1101.001_2 = 0.1101001} \times 2^{100}$ （比例因子为2）

4. 确定阶码和尾数

$0.1101001 \times 2^{100} = 0.01101001 \times 2^{101}$ 计算机中采用哪一种?

浮点数的规格化（normalization）

- 尾数最高有效位为1的数 称为规格化数
- 例： $13.125 = 0.1101001 \times 2^{100}$
- 为了数据表示的唯一性
- 为了在尾数中表示最多的有效数据位
- 两种规格化数
  - 0.1XXXXX
  - 1.XXXXX （IEEE 754 采用）

浮点数的范围与精度

构成：阶码 E，尾数 M，符号位 S，基数 R

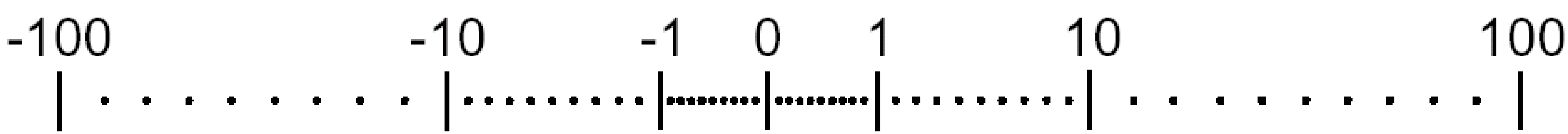
S	E	M

- 阶码位数越多，数据表示的范围就越大
- 尾数位数越多，数据表示的精度越高
- 机器字长一定时，阶码越长，表示范围越大，精度越低；反之亦然

浮点数的范围与精度

- 阶码、尾数的长度均确定的情况下，浮点数可以表示的数值是确定的、个数是有限的

绝对值越大，浮点数分布越稀疏



举例

- 例1： 8位定点小数可表示的范围
  - 0.0000001 --- 0.1111111
  - 1/128 --- 127/128 （准确表达127个有理数）
- 例2： 设阶码3位（原码）， 尾数4位（原码）
  - 可表示正数：$2^{-11}*0.0001$ --- $2^{11}*0.1111$
  - 0.0000001 --- 111.1
- 计算机中可表达的浮点数的个数和范围也是有限的

小结

- 浮点数的表达方式

  阶码E，尾数M，符号位S，基数R

  $N = (-1)^S \times M \times R^E$

- 浮点数的规格化

  数据表示的唯一性、在尾数中表示最多的有效数据位

- 浮点数的表达范围与精度

  机器字长一定时，阶码越长，表示范围越大，精度越低