Section
Chapter 2.1 Machine-level representation of numbers.

一、定点数的表示
※ 以下内容主为中国特色，即使不读也不会有任何损失.
请注重 Aside 及英语描述部分.

1、原码：符号化的数值. (记作 $[x]_原$ 或 $[x]_o$)
•规则：前面加符号位，正数：0
负数：1.

	真值	原码 $[x]_o$	Tru Val	$[x]_o$
正数	$x=+0.x_1\cdots x_n$	$0.x_1\cdots x_n$	$x=+x_1x_2\cdots x_n$	$0x_1\cdots x_n$
负数	$x=-0.x_1\cdots x_n$	$1.x_1\cdots x_n$	$x=-x_1\cdots x_n$	$1x_1\cdots x_n$

形式化地：

$$[x]_o=\begin{cases}x & 0\le x<1\\ 1+|x| & -1<x\le 0\end{cases}$$

定点小数

$$[x]_o=\begin{cases}x & 0\le x<2^n\\ 2^n+|x| & -2^n<x\le 0\end{cases}$$

定点整数.

•不足：2个"0"表示.
要为"+","-"分别设计电路.
•好处：对称区间

2、反码 (1's complement).
•规则：当真值为正数时，与原码相同 (记作 $[x]_反$ 或 $[x]_n$)
负 完全相反

	真值	$[x]_n$	真值	$[x]_n$
正数	$x=+0.x_1\cdots x_n$	$0.x_1\cdots x_n$	$x=+x_1\cdots x_n$	$0.x_1\cdots x_n$
负数	$x=-0.x_1\cdots x_n$	$1.\overline{x_1}\cdots\overline{x_n}$	$x=-x_1\cdots x_n$	$1.\overline{x_1}\cdots\overline{x_n}$

形式化地：

$$\text{decimal } [x]_n=\begin{cases}x & 0\le x<1\\ \underline{(2-2^{-n})+x} & -1<x\le 0\end{cases}$$

发生借位.

$$\text{integer } [x]_n=\begin{cases}x & 0\le x<2^n\\ (2^{n+1}-1)+x & -2^n<x<0\end{cases}$$

• 缺点：存在 $+0$ 与 -0 两种表示法.

加减时应循环进位 (如右)

例 $+0.101$ $\quad -1$

$\downarrow \qquad\qquad \downarrow\ 1.001$

$0.101 \qquad 0.110$

$$\begin{array}{r} 0.101 \\ +\ 0.110 \\ \hline 0.011 \\ +1 \\ \hline 0.100 \end{array}$$

• 特点：对称区间.

3、称(2的)补码：真值$\underline{模}$ 最高位进位的权值. 记作 $[x]_c$

(1) 形式化

$$[x]_c = \begin{cases} x & 0 \le x < 1 \\ 2-x & -1 \le x \le 0 \end{cases} \pmod 2$$

$$[x]_c = \begin{cases} x & 0 \le x < 2^n \\ 2^{n+1}+x & -2^n \le x \le 0 \end{cases} \pmod{2^{n+1}}.$$

遵循了一个循环群，因而较为美丽. 可见数学之美在生活中尤为重要.

(2) 求算方法

1° 定义(略). 同上面

2° 如果是正数，不用动；

如果是负数，值要取反，末位加 1.

"证明"? $\cancel{-x+x \equiv 0 \pmod{2^{n+1}}}$ ~~有而，取~~

$\cancel{-x+x}$

$\cancel{-x+x=0}$

$\cancel{-x+x=2^{n+1} \Rightarrow -x=2^{n+1}+x}$

注意到 $\sim x + x = \underbrace{11\cdots1}_{\text{全为1的二进制串}} \xrightarrow[\text{条件}]{\text{补码}} -1$

即 $\sim x + 1 = -x$.

(3) (4的) 补码：采用2个二进制位作为符号 $\begin{cases} 00 & \text{正数} \\ 11 & \text{负数} \end{cases}$.

此时模 2^{n+2}.

此为便于检测溢出.
- 01 → 正溢出 positive ovfl
- 10 → 负溢出 negative ovfl.

4、移码. 仅可用于定点整数之表示. 平行移动至

$[x]_{\text{move}}$ 移 ($[X]_m$) 记作

• 规则：$[x]_m = x + \underbrace{bias}_{\text{常用 } 2^n}$.

$[-2^n, 2^n-1] \mapsto [0, 2^{n+1}-1]$.

• 缺点：运算较复杂.

• 优点：可直接比大小.

二、浮点数表示

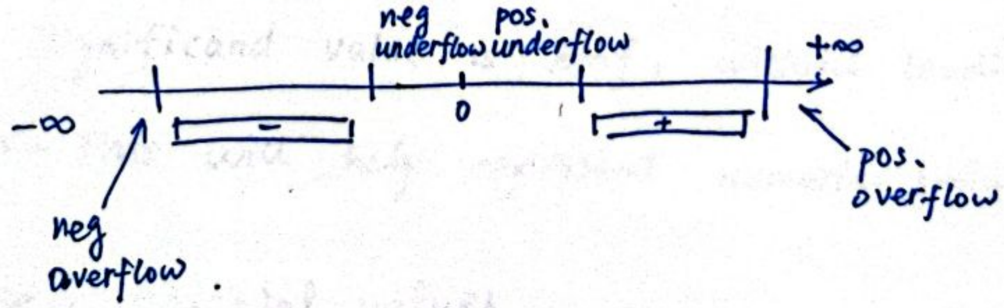
1、初步想法：表达为

$N = 2^{E} \times M$

E → 阶码 定点整数

M → 尾数 定点小数.

e_0	$e_1 \cdots e_k$	m_0	$m_1 \cdots m_n$
阶符	阶码 exponential	数符	尾数 mantissa



问题：有不同表示形式. 如 $2^1 \times 0.1$, $2^2 \times 0.005$ 等.

↳ 规格化 (normalize).

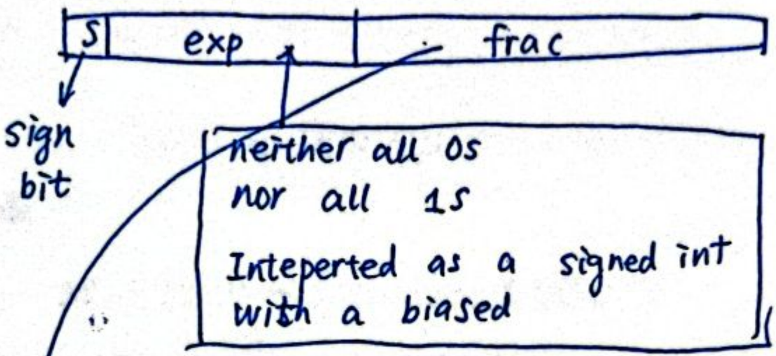
尾数的绝对值(真值)应大于 $(0.1)_2$

2、IEEE 754 标准 $V=(-1)^S \times M \times 2^E$

(1) ~~32位单精度浮点数~~

有3类数据
- Normalized Values
- Denormalized Values
- Special Values : $+\infty$, $-\infty$, NaN.

Case 1、Normalized values



$$E = e - \text{Bias}$$
$$= e - (2^{k-1} - 1).$$

Make transformation smother

fractional value f: $0 \le f < 1$,
having binary repr.
$$0.f_{n-1} \cdots f_0.$$

$M = 1 + f$. (Implied leading 1).

Case 2、Denormalized

$E = 1 - \text{Bias}$

Significand value is $M = f$, without leading 1 repr.

↳ This will help represent numeric value 0.

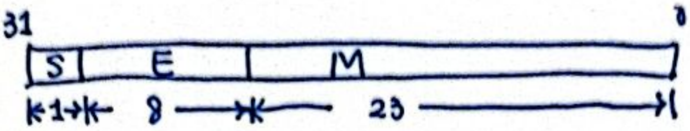
Case 3. Special values

| S | 111…1 | 000 | Infinity

| S | 111…1 | ≠0 | Not a number: $\sqrt{-1}$, $\infty - \infty$.

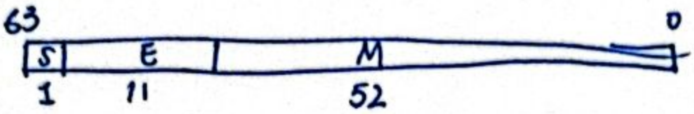
具体地

float:



偏移量：$2^{8-1}-1=127$.

double



在本课程中，~~仅讨论~~ 习题中仅有 float 类型的.

Eg1、将 20.59375 转换为 IEEE 754 的 float 十六进制机器码.

首先：$20.59375 = (10100.10011)$.

移动小数点，变为 1.M 的形式：

$$1.010010011 \times 2^4.$$

由此，$S=0$，$E=e+127=131=(10000011)_2$.

$$M=010010011$$

$\Rightarrow$ 0 1000 0011 01001001100 ···· 0.

Eg2、求 IEEE 754 float 值 $(C1360000)_{16}$ 对应的十进制值.

首先将十六进制数展开为二进制数

$$1\boxed{100\ 0001\ 0}011\ 0110\ 0000\ 0000\ 0000\ 0000$$

知 $S=1$

$$E=e-127=128+2-127=3.$$

$$f=\frac{1}{4}+\frac{1}{8}+\frac{1}{32}+\frac{1}{64}=\frac{16+8+2+1}{64}=\frac{27}{64}.$$

因此为

$$-2^3\times\left(1+\frac{27}{64}\right)=\cancel{8\times}-\frac{27}{8}=-11.375.$$