# 第一和二章

讲课方式：讲课会列出知识框图并区分重点部分和非重点部分，由于课程时长有限，主要以重点难点为主，其他知识点会列出，供大家参考，

1. **特别重要知识点，**
2. **重要，考的可能性较大**
3. **比较重要，历年考过**
4. 可以看看，难度不是很大
5. 还行，好，没关系

## 数值和编码

### 1.1数制的相互转换

二进制-->八进制，十六进制

\*数位长度关系：1个八进制、十六进制数位＝3 bit、4 bit

整数部分高位补0，小数部分低位补0

1000101.0111--》（105.34）

例—(13.724)8=(001 011.111 010 100)2=(1011.1110101)2

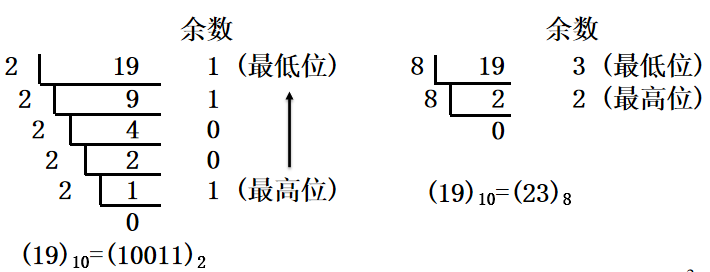
(10011.01)2=(010 011.010)2=(23.2)8

其他进制-->十进制

（11011.1）=>

十进制转-->其他进制

**整数部分采用除基取余法，**

例题：19转换为二进制数及八进制数

**小数部分采用乘基取整法**

**例—将(0.6875)10分别转换成二、八进制数**

**转换成二进制：**

0.6875×2 = 1.375 1 (最高位)

0.375 ×2 = 0.75 0

0.75 ×2 = 1.5 1

0.5 ×2 = 1.0 1 (最低位)

(0.6875)10 = (0.1011)2

**转换成八进制：**

0.6875×8 = 5.5 5 (最高位)

0.5 ×8 = 4.0 4 (最低位)

(0.6875)10 = (0.54)8

### 真值和机器数



## 定点数的表示和运算

### 1.机器数的表示

机器数可分为：

**无符号数**：机器字长的所有二进制位均表示数值

**带符号数**：数值部分和符号均用二进制代码表示

例： 8位机器数为：11011011

若为无符号整数，则 11011011 表示二进制整数。  
 其真值为 11011011＝(219)10

若为带符号整数，则最高位为符号，  
 1 1011011 表示二进制整数 －1011011  
 其真值为 －1011011＝(－91)10

## 原码，补码，反码，移码

### 2.1 原码

**作用**：它是计算机底层最基本的表示方法，有多个01码构成，但是它的缺点就是进行一些算术运算时可能会出错。

**X 0≤X＜2n-1**

**2n-1 - X = 2n-1 +|X| -2n-1＜X≤0**

**[X]原 =**

例1—[+1101]原=01101；[-1101]原=11101

**X 0≤X＜1**

**1-X=1+|X| -1＜X≤0**

**[X]原 =**

①X与[X]原关系—

·[X]原与X表示值的范围相同，

·[+0]原≠[-0]原

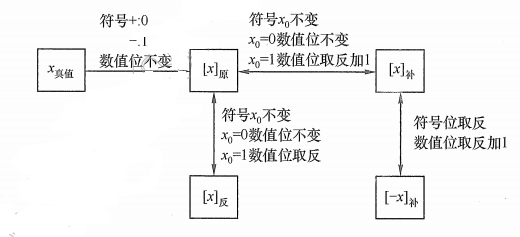
②运算方法—符号与数值分开运算(与手工运算一致)适合于乘除法，加减法较复杂

### 2.2：补码：

由于原码进行加减法比较复杂，甚至还有可能会出错，所以就有了补码

### 2.3：反码：

### 2.4：移码：



### 2.5 定点数的运算

**算术位移**

算术移位的对象是**有符号数**，在移位过程中符号位保持不变

对于正数，由于原码=反码=补码=真值，因此移位后出现的位均以0添之。

对于负数：

①负数的原码数值部分与真值相同，因此在移位时只要使符号位不变，其空位均添0、

②负数的反码各位除符号位外与负数的原码正好相反，因此移位后所添的代码应与原码相

反，即全部添1.

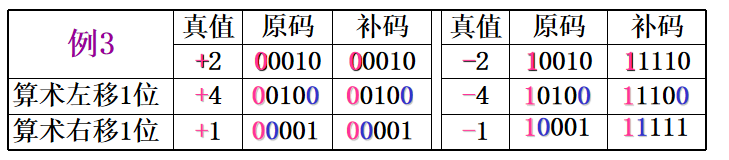
③分析面质够得到补码的过程发现，当对其由低位向高位找到第一个“1”时，在此p

左边的各位均与对应的反码相同，而在此“1”右边的各位（包括此“1”在内）均与对

应的原码相同。因此负数的补码左移时，因空位出现在低位，则添补的代码与原码相同，

即添0：右移时因空位出现在高位，则添补的代码应与反码相同，即添1.

对于原码，左移一位若不产生溢出，相当于乘以2（与水进制的左移一位相当于乘以10类似)，右移一位，若不考虑因移出而舍去的末位尾数，相当于除以2.



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 码制 | 真值 | 算术移位运算规则 | | |
| 符号位 | 算术左移空位添补 | 算术右移空位添补 |
| 原码 | 正/负数 | 保持  不变 | 补0 (数值低位) | 补0      (数值高位) |
| 补码 | 正/负数 | 补0 (数值低位) | 补符号位 (数值高位) |

|  |  |  |  |
| --- | --- | --- | --- |
| **例4** | **原/补/反码** | **原码** | **补码** |
| **00100001(+33)** | **10100001(-33)** | **11011111(-33)** |
| **①左移1位** |  |  |  |
| **再左移1位** |  |  |  |
| **②右移1位** |  |  |  |
| **再右移4位** |  |  |  |
| **再右移1位** |  |  |  |

**逻辑位移：**

操作数类型：无符号机器数

运算规则：机器数整体移位，移出的数丢弃，出现的空位补0即可

**循环位移（非重点）**

### 2.6定点数的加减法

**原码的加减法规则：**

加法规则：先判符号位，若相同，则绝对值相加结果符号位不变：若不同，则做减法，绝

对值大的数减去绝对值小的数，结果符号位与绝对值大的数相同。

减法规则：两个原码表示的数相减，首先将减数符号取反，然后将被减数与符号取反后的减

数按原码加法进行运算。

利用原码进行两数相加运算时，首先要判别两数符号，若同号则做加法，若异号则做减法。

在利用原码进行两数相减运算时，不仅要判别两数符号，使得同号相减，异号相加；还要判别两数绝对值的大小，用绝对值大的数减去绝对值小的数，取绝对值大的数符号为结果的符号。

可见原码表示不便于实现加减运算。

**补码的加减法规则：**

1)参与运算的两个操作数均用补码表示。

2）按二进制运算短则运算，逢二进一。

3）符号位与数值位按同样规则一起参与运算，符号位运算产生的进位要丢掉，结果的符号

位由运算键出。

4)补码加减运算依据下面的公式进行。当参加运算的数是定点小数时，模M=2；当参加运

算的数是定点整数时，模M=2"+1.

[A+]=[A]++[B](mod M)

[A-B]=[A]\*+[-B](mod M)

也就是说，若做加法，则两数的补码直接相加：若做减法，则将被减数与减数的机器负数

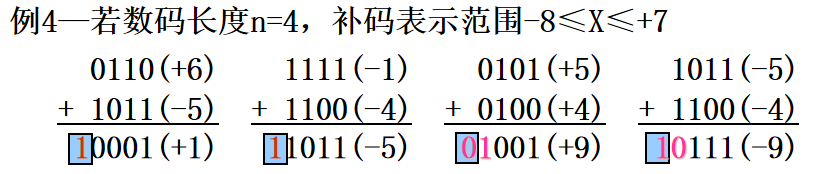
相加。

例2—设A=-01010，B=+01101，用补码求[A+B]补、[A-B]补

例3—设A=+6/16，B=+9/16，用补码求[A+B]补、[A-B]补

### 2.7溢出判断

溢出条件：同号数相加、结果与加数符号不同时溢出!

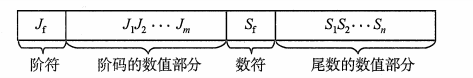


2.8定点数的乘除法

## 浮点数的表示和运算

### 4.1浮点数的表示

表示：N=rEM



### 4.2规格化浮点数

### 4.3 IEEE标准

### 4.4 浮点数的加减运算

## 算术逻辑单元ALU