



宿遷學院

计算机组成原理实验报告

系	别:	三系
专	业:	计算机科学与技术
年	级:	2010 级
班	级:	10 计本（1）班
姓	名:	缪广东
学	号:	20100303130
指导老师:		巫宗宾
成	绩:	

目 录

实验一 数值的表示以及运算..... 3

实验二 8 位算术逻辑运算实验..... 16

实验三 带进位控制 8 位算术逻辑运算实验..... 19

实验四 16 位算术逻辑运算实验..... 22

实验五 移位运算器实验..... 25

实验六 存储器实验..... 28

实验七 微控制器实验..... 31

实验一 数值的表示以及运算

1. 真值转换为原码、反码、补码和移码

一、实验目的

掌握真值与原码、反码、补码和移码之间的转换方法。

1. 正数的原码、反码、补码表示都相同。
2. 原码、反码、补码表示的都是有符号数，最高位为符号位，“1”表示负，“0”表示正。
3. 一个定点机器数既可以看作定点小数（小数点位于符号位之后），也可以看作定点整数（小数点位于最低位之后）。
4. 如果真值为负数，反码的数值位等于原码的数值位按位取反，补码的数值位等于原码的数值位按位取反后再加 1，或者反码的数值加 1。
5. 从另一个角度看，如果把机器数看作 8 位无符号整数，真值的数值位为 7 位整数，则补码等于真值+256，移码等于真值+128，反码等于真值+255。
6. $-X$ 的补码等于 X 的补码各位（包括符号位）按位取反，末位加 1。
7. 移码主要用作浮点数的阶码，没有定点小数表示法。一个整数的同字长的移码和补码数值位相同，符号位互反。

二、实验方法与步骤

从界面中可以输入真值（一位符号位，可正可负，7 位数值位，或 0 或 1，每位均有两种状态，故用 8 个双位按钮输入数据）；界面左上角的下拉列表框用于选择将真值转换为何种机器数。当输入真值并选择好目的机器数的类型之后，单击转换按钮，即可显示与真值对应的机器数。

由真值转换为各种机器码的规则如下：

1. 真值转换为原码的规则

- （1）如果真值是正数，原码的最高位为“0”，其余各位与真值的对应位相同；
- （2）如果真值是负数，原码的最高位为“1”，其余各位与真值的对应位相同；
- （3）如果真值是“0”，原码有两种表示法，即 000₁₁₁0 或 100₁₁₁0，分别表示 +0 或 -0；

(4). 因此原码的数值位表示数值的绝对值，而符号位分别用 0 和 1 表示数值的正和负。

(5) 综上所述，如果把原码看成无符号整数，则真值到原码的转换规则是：

当 $x \geq 0$ 时， $[x]_{\text{原}} = x$ ；当 $x \leq 0$ 时， $[x]_{\text{原}} = 2^{(n-1)} - x$ 或 $2^{(n-1)} + |x|$ 。

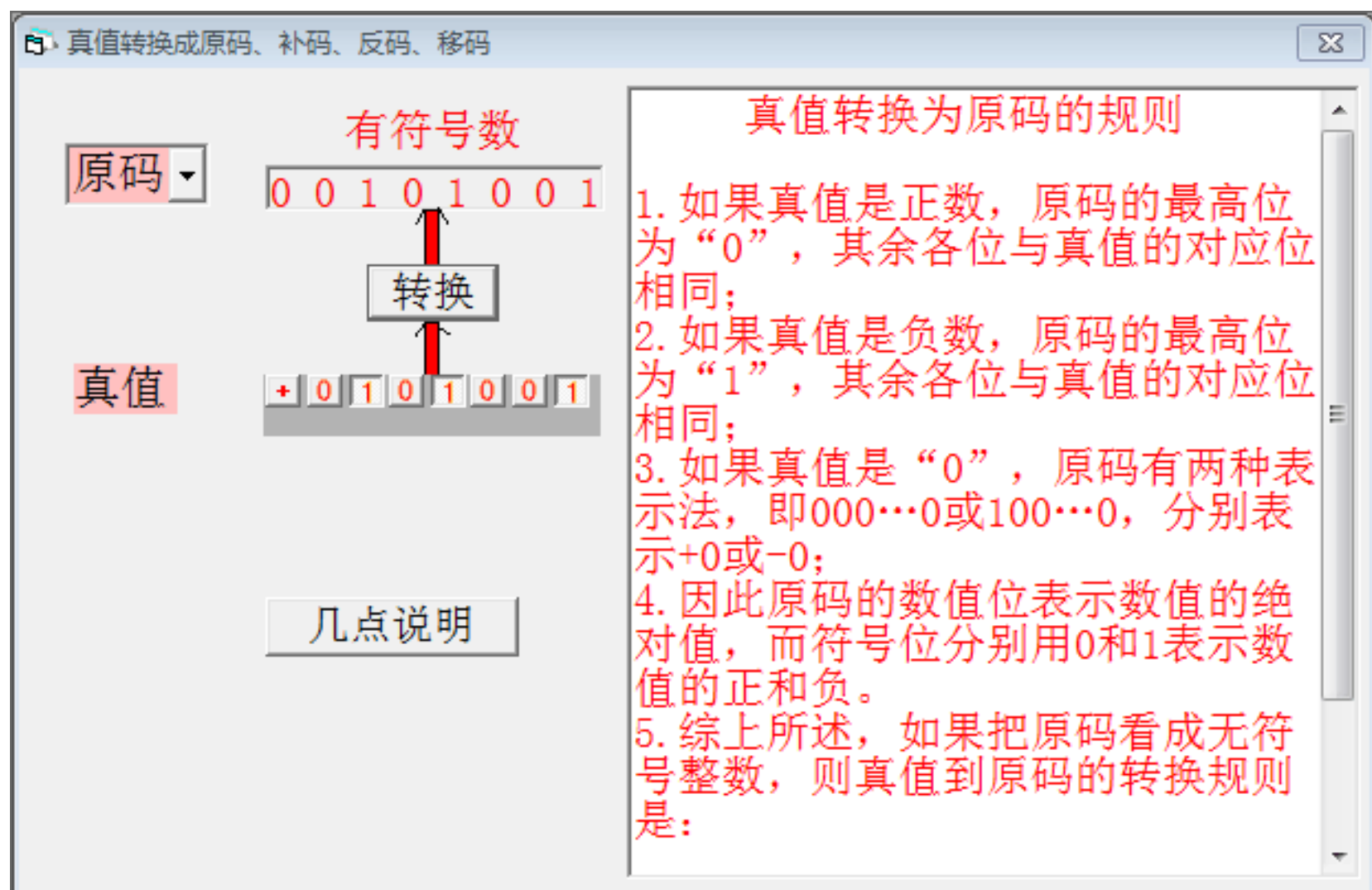


图 1.1 真值转换为原码

1. 真值转换为反码的规则

(1) 如果真值是正数，反码的最高位为“0”，其余各位与真值的对应位相同；

(2) 如果真值是负数，反码的最高位为“1”，其余各位将真值的各位取反；

(3) 如果真值是“0”，反码有两种表示法，即000...0 或 111...1，分别表示+0 或-0；

(4) 综上所述，如果把反码看成无符号整数，则真值到反码的转换规则是：

当 $x \geq 0$ 时， $[x]_{\text{反}} = x$ ；当 $x \leq 0$ 时， $[x]_{\text{反}} = 2^n + x - 1$ (模 2^n)。

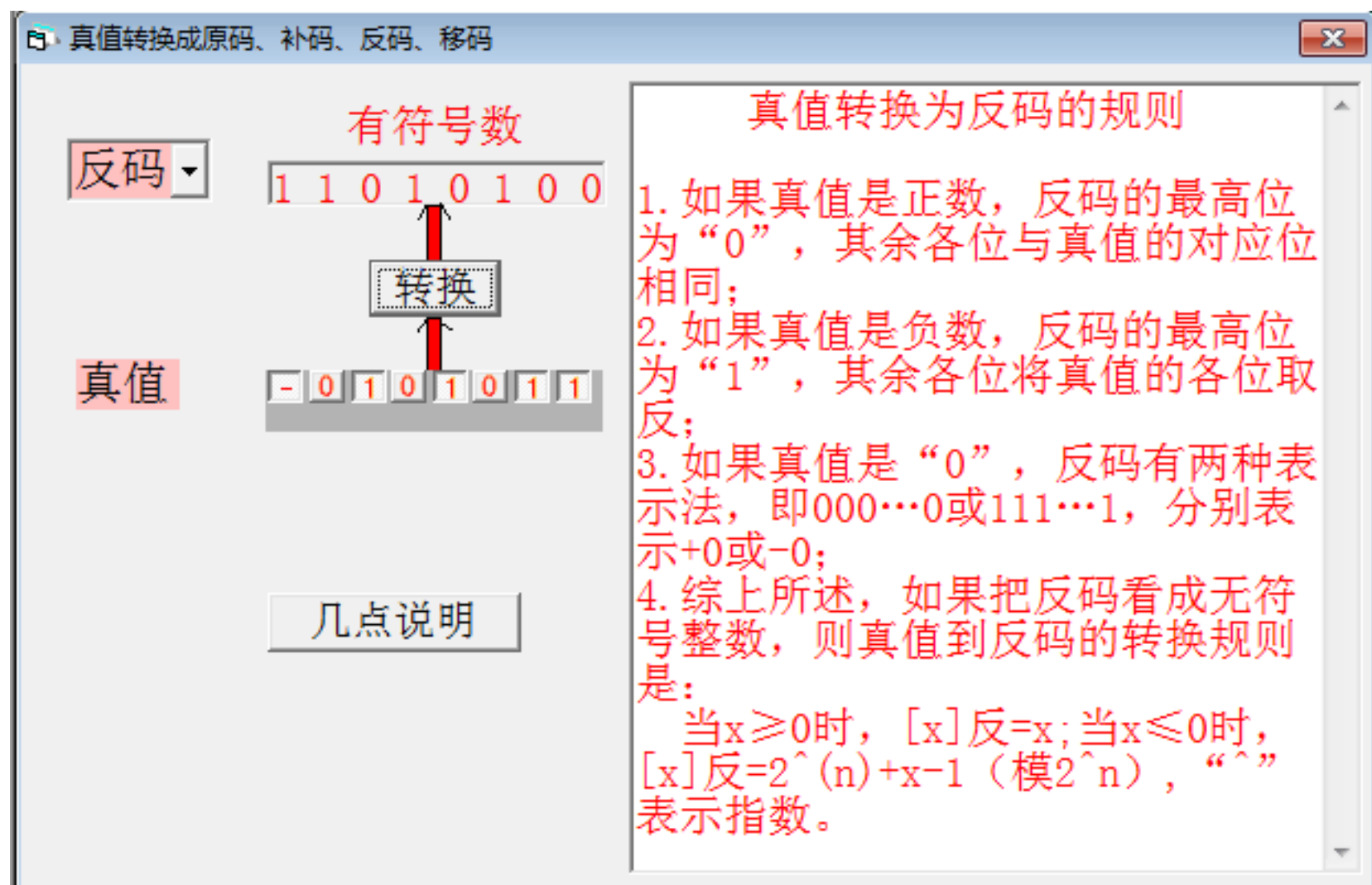


图 1.2 真值转换为反码

1. 真值转换为补码的规则

(1) 如果真值是正数，补码的最高位为“0”，其余各位与真值的对应位相同；

(2) 如果真值是负数，补码的最高位为“1”，其余各位将真值的各位取反，末位加 1；

(3) 如果真值是“0”，补码表示法 is 唯一的，即 000...0，在补码中 100...0 表示-1（如果看作定点小数）或 2^{n-1} （如果看作定点整数），最高位既表示符号（负数），也表示数值 1（或 2^{n-1} ），不能表示-0；

(4) 负数补码的另一种求法是，最高位（符号位）为“1”，从真值的低位向高位数第一个 1 以及其右边的各位保持不变，而其左边的各位按位取反，即可得到补码的数值位。

(5) 综上所述，如果把补码看成无符号整数，则真值到补码的转换规则是：

$$[x]_{\text{补}} = 2^n + x \quad (\text{模 } 2^n)。$$

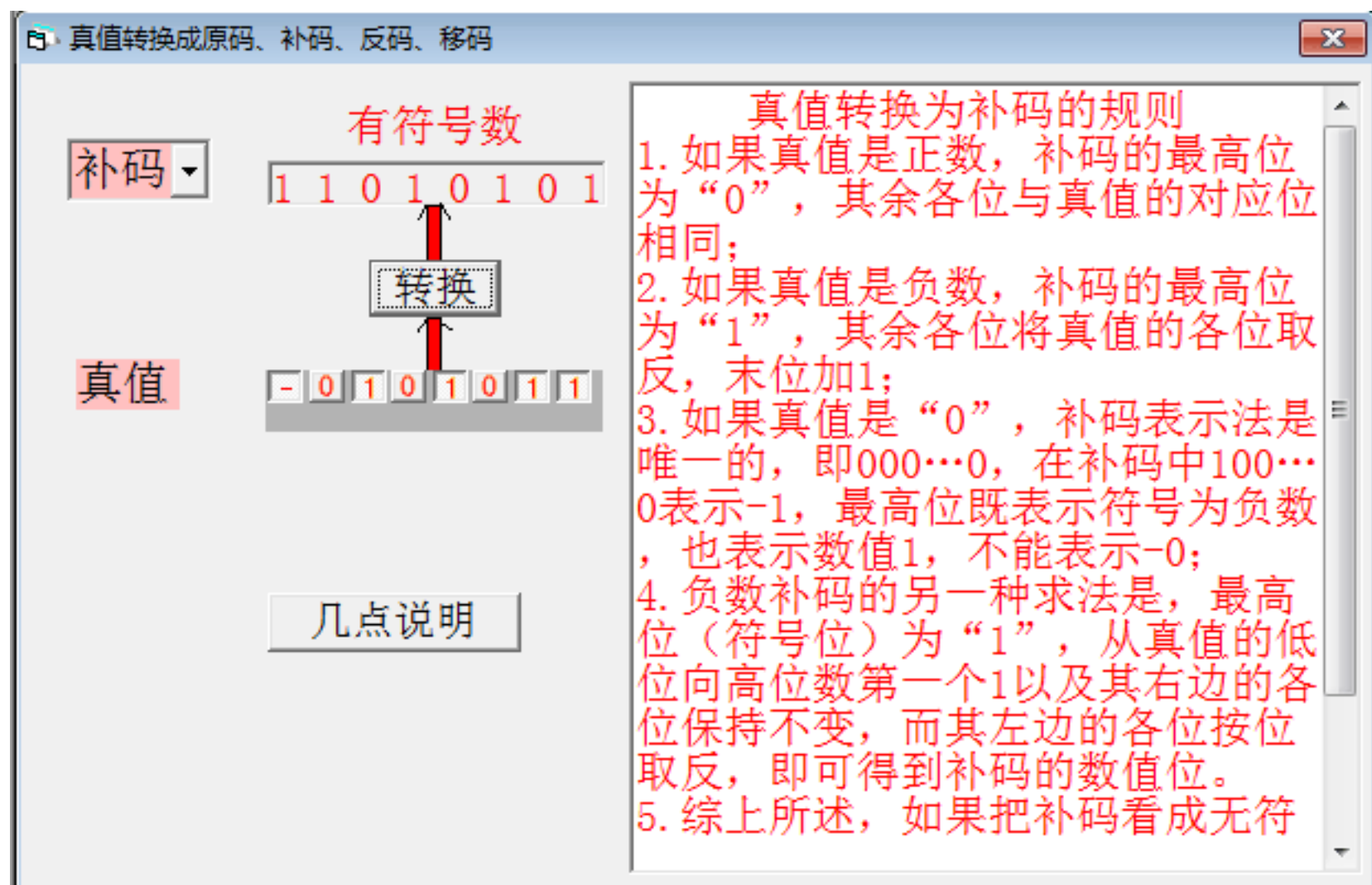


图 1.3 真值转换为补码

2. 真值转换为移码（定点整数）的规则

- (1) 如果真值是正数，移码的最高位为“1”，其余各位与真值的对应位相同；
- (2) 如果真值是负数，移码的最高位为“0”，其余各位将真值的各位取反，末位加 1；
- (3) 如果真值是“0”，移码表示法是唯一的，即 100...0；
- (4) 不难看出，移码与补码相比，数值位完全相同，而符号位相反。两者的转换非常容易。
- (5) 移码只用来表示浮点数中的阶码，不存在定点小数格式。
- (6) 综上所述，如果把移码看成无符号整数，则真值到移码的转换规则是：

$$[x]_{\text{移}} = 2^{(n-1)} + x \quad (\text{模 } 2^n)。$$

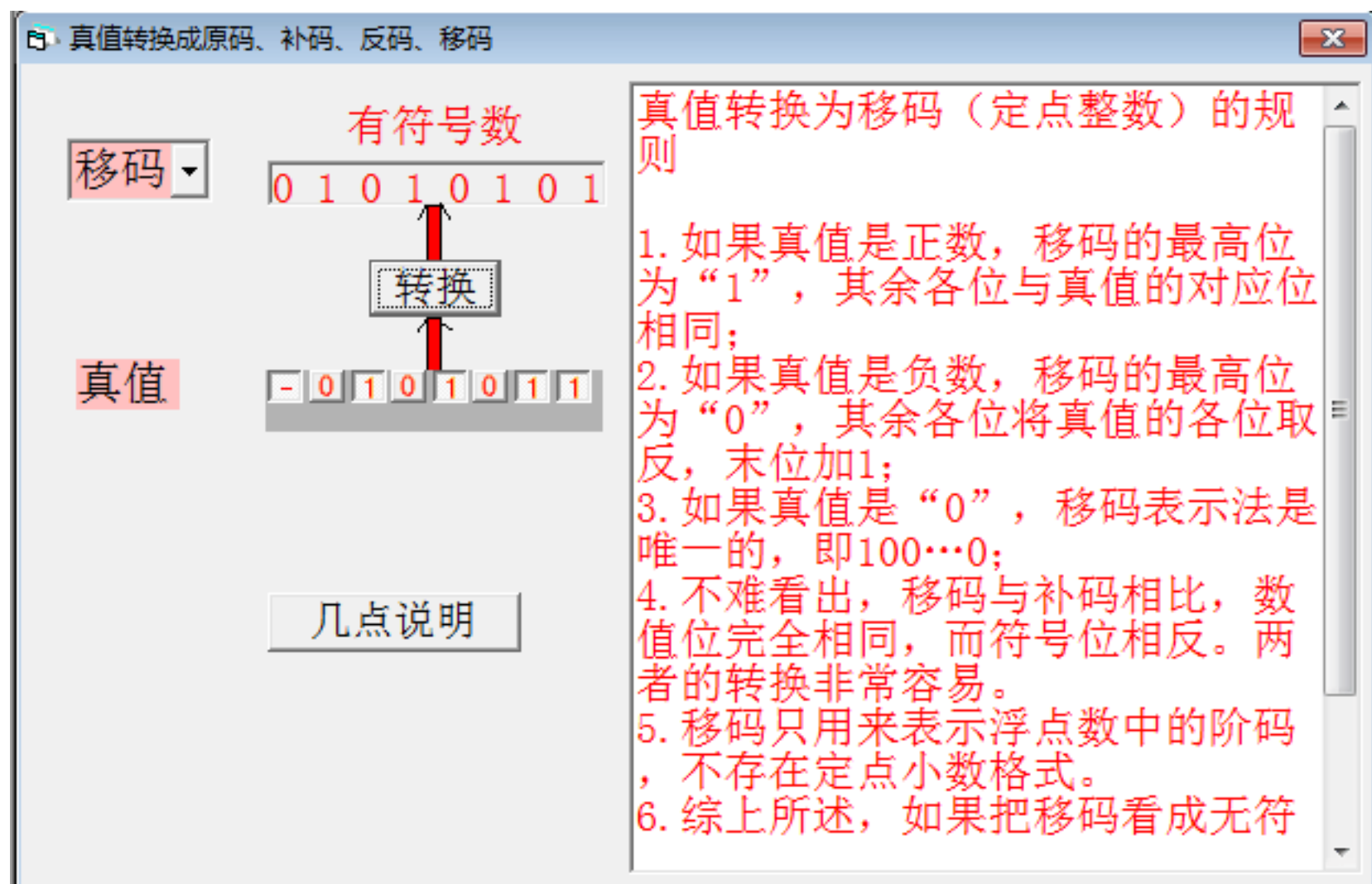


图 1.4 真值转换为移码

2. 浮点数表示

一、实验目的

掌握 IEEE-754 浮点数的表示及它与十进制之间的相互转换方法。

二、实验原理

1. 实数的 IEEE 754 表示形式

一个实数 V 在 IEEE 754 标准中可以用 $V = (-1)^s \times M \times 2^E$ 的形式表示，说明如下：

(1) 符号 s (sign) 决定实数是正数 ($s=0$) 还是负数 ($s=1$)，对数值 0 的符号位特殊处理。

(2) 有效数字 M (significand) 是二进制小数， M 的取值范围在 $1 \leq M < 2$ 或 $0 \leq M < 1$ 。

(3) 指数 E (exponent) 是 2 的幂，它的作用是对浮点数加权。

2. 浮点格式

浮点格式是一种数据结构，它规定了构成浮点数的各个字段，这些字段的布局，及其算术解释。IEEE 754 浮点数的数据位被划分为 3 个字段，对以上

参数值进行编码：

(1) 一个单独的符号位 s 直接编码符号 s 。

(2) k 位的偏置指数 e ($e = e_{k-1} \dots e_1 e_0$) 编码指数 E ，移码表示。

(3) n 位的小数 f (fraction) ($f = f_{n-1} \dots f_1 f_0$) 编码有效数字 M ，原码表示。

示。

3. 浮点数的分类

根据阶码 E 的值，被编码的浮点数可分成三种类型。

(1) 规格化数

当有效数字 M 在范围 $1 \leq M < 2$ 中且阶码 E 的位模式 $e_{k-1} \dots e_1 e_0$ 既不全是 0 也不全是 1 时，浮点格式所表示的数都属于规格化数。这种情况中小数 f ($0 \leq f < 1$) 的二进制表示为 $0.f_{n-1} \dots f_1 f_0$ 。有效数字 $M = 1 + f$ ，即 $M = 1.f_{n-1} \dots f_1 f_0$ (其中小数点左侧的数值位称为前导有效位)。我们总是能调整指数 E ，使得有效数字 M 在范围 $1 \leq M < 2$ 中，这样有效数字的前导有效位总是 1，因此该位不需显示表示出来，只需通过指数隐式给出。

需要特别指出的是，指数 e 要加上一个偏置值 $Bias$ ，转换成无符号的阶码 E ，也就是说指数 e 要以移码的形式在存放计算机中。且 E 、 e 和 $Bias$ 三者的对应关系为 $E = e + Bias$ ，其中 $Bias = 2^{k-1} - 1$ 。

(2) 非规格化数

当阶码 E 的位模式 $e_{k-1} \dots e_1 e_0$ 全为零 (即 $e = 0$) 时，浮点格式所表示的数是非规格化数。这种情况下， $e = 1 - Bias$ ，有效数字 $M = f = 0.f_{n-1} \dots f_1 f_0$ ，有效数字的前导有效位为 0。

非规格化数的引入有两个目的。其一是它提供了一种表示数值 0 的方法，其二是它可用来表示那些非常接近于 0.0 的数。

(3) 特殊数

当阶码 E 的位模式 $e_{k-1} \dots e_1 e_0$ 全为 1 时，小数 f 的位模式 $f_{n-1} \dots f_1 f_0$ 全为 0 (即 $f = 0$) 时，该浮点格式所表示的值表示无穷， $s = 0$ 时是 $+\infty$ ， $s = 1$ 时是 $-\infty$ 。

当阶码 E 的位模式 $e_{k-1} \dots e_1 e_0$ 全为 1 时，小数 f 的位模式 $f_{n-1} \dots f_1 f_0$ 不为 0 ($f_{n-1} \dots f_1 f_0$ 至少有一个非零即 $f \neq 0$) 时，该浮点格式所表示的值被称为 NaN (Not a Number)。比如当计算 $\infty - \infty$ 时用作返回值，或者用于表示未初始化的数据。

3. IEEE 754 浮点存储格式

IEEE 754 标准准确地定义了单精度和双精度浮点格式，并为这两种基本格式的分别定义了扩展格式。

(1) 单精度格式

IEEE 单精度浮点格式共 32 位，包含三个构成字段：23 位小数 M，8 位阶码 E，1 位符号 s。将这些字段连续存放在一个 32 位字里，并对其进行编码。

其中第 31 位包含符号 s，23:30 位包含 8 位阶码 E；0:22 位包含 23 位的小数 M；

(2) 双精度格式

IEEE 双精度浮点格式共 64 位，占 2 个连续 32 位字，包含三个构成字段：52 位的小数 M，11 位的阶码 E，1 位的符号位 s。将这 2 个连续的 32 位字整体作为一个 64 位的字，进行重新编号。

三、实验步骤

IEEE-754标准规定的32位浮点数格式

符号位 S (1位) | 阶码 E (移码) (8位) | 尾数 M (补码或原码) (23位)

0 10000010 01101100000000000000000

求浮点数的十进制值 | 十进制数的浮点数表示

原码格式变换过程

第一步：求指数e 指数e=阶码-127=130-127=(3)10
 第二步 求尾数 包含隐藏位1的尾数 1.M=1.011011
 第三步 求真值 $x=(-1)^s \times 1.M \times 2^e = + (1.011011) \times 2^3 = +1011.011 = +11.375$

关于 IEEE-754 标准的 32 位浮点数 (短实数) 格式的说明

符号位 S 表示数符 (用 0 和 1 分别表示正数和负数)；假设一个数的尾数 M，基数为 2，指数为 e，则阶码 $E=e+127$ ，尾数 M 可以用原码或补码表示。但 IEEE-754 标准规定，尾数的绝对值必须 ≥ 0.5 ，对原码表示法，意味着，尾数的最高位为 1，且不占尾数的有效位，即实际的尾数为 24 位，为 1.M。故对原码而言，实际的数值为：
 $x=(-1)^S \times (1.M) \times 2^{E-127}$

图 1.5 浮点数转换为十进制数

第一步 求指数 e

第二步 求尾数 包含隐藏位 1 的尾数 1.M

第三步 求真值 $x=(-1)_s \times 1.M \times 2^e$

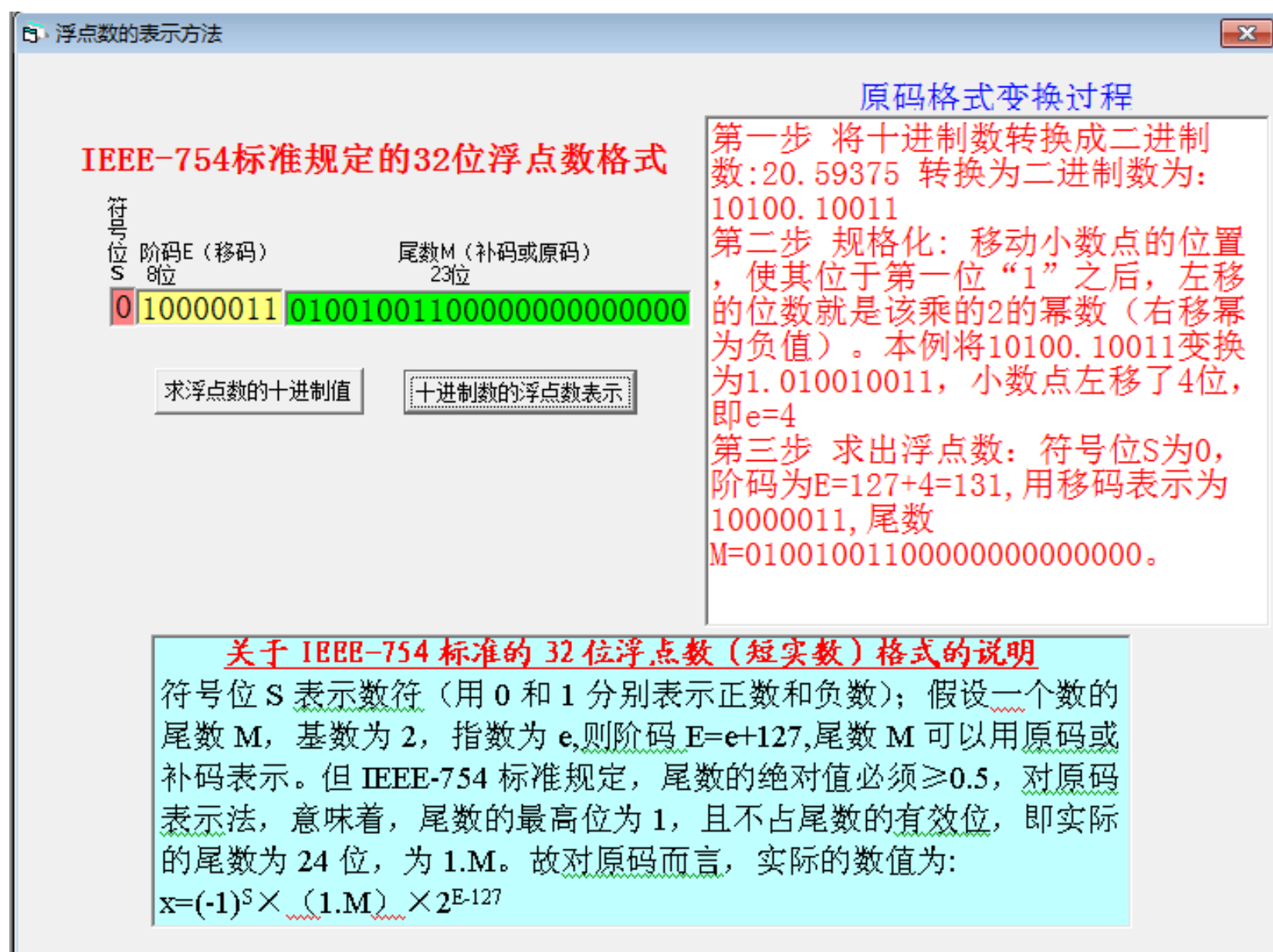


图 1.6 十进制数转换为浮点数

第一步 将十进制数转换成二进制数：例如，20.59375 转换为二进制数为：10100.10011

第二步 规格化：移动小数点的位置，使其位于第一位“1”之后，左移的位数就是该乘的 2 的幂数（右移幂为负值）。将 10100.10011 变换为 1.010010011，小数点左移了 4 位，即 $e=4$

第三步 求出浮点数：符号位 S 为 0，阶码为 $E=127+4=131$ ，用移码表示为 10000011，尾数 $M=010010011000000000000000$ 。

3. 补码加减法运算

一、实验目的

掌握补码加减法的运算规则及溢出判别

二、实验原理

1. 补码的加法运算规则是：

- (1) 有符号的两个数采用补码形式。
- (2) 进行加法运算可以把符号位和数值位一起进行运算（若符号位有进位，则丢掉）。
- (3) 结果为两数之和的补码形式。

2. 补码的减法（ $[X-Y]_{\text{补}} = [X]_{\text{补}} + [-Y]_{\text{补}}$ ）运算规则是：

(1) 有符号的两个数采用补码形式。

(2) 对减数进行求负操作 $[-Y]_{\text{补}}$ 。一般称已知 $[Y]_{\text{补}}$ 求得 $[-Y]_{\text{补}}$ 的过程叫变补或求负。已知 $[+Y]_{\text{补}}$ 求 $[-Y]_{\text{补}}$ 的规则是全部位（含符号位）按位取反后再加1。求 $[X]_{\text{补}} + [-Y]_{\text{补}}$ 的和。

(3) 结果为两数之差的补码形式。

3. 溢出判别

(1) 溢出

如果两个操作数没有超出补码的表示范围，而运算结果超出补码的表示范围，称为“溢出”，结果大于补码的表示范围的上限，称为“上溢”；结果小于补码的表示范围的下限，称为“下溢”。

(2) 判别

通常有两种判断溢出的方法，一是双符号位法，即两个操作数以及结果均用两个符号位（变形补码），如果结果的两个符号位一致，表示没有溢出，如果不一致，表示溢出。最高的一位始终表示运算结果的正确的符号：0 为正（或上溢），1 为负（或下溢）。

第二种方法是采用单符号位运算，但根据符号位和最高有效位产生的进位是否相同而判定是否溢出：如果两进位位相同，无溢出；不同则表明溢出。当溢出时，符号位进位为 0，表示上溢；为 1 表示下溢。

三、实验步骤

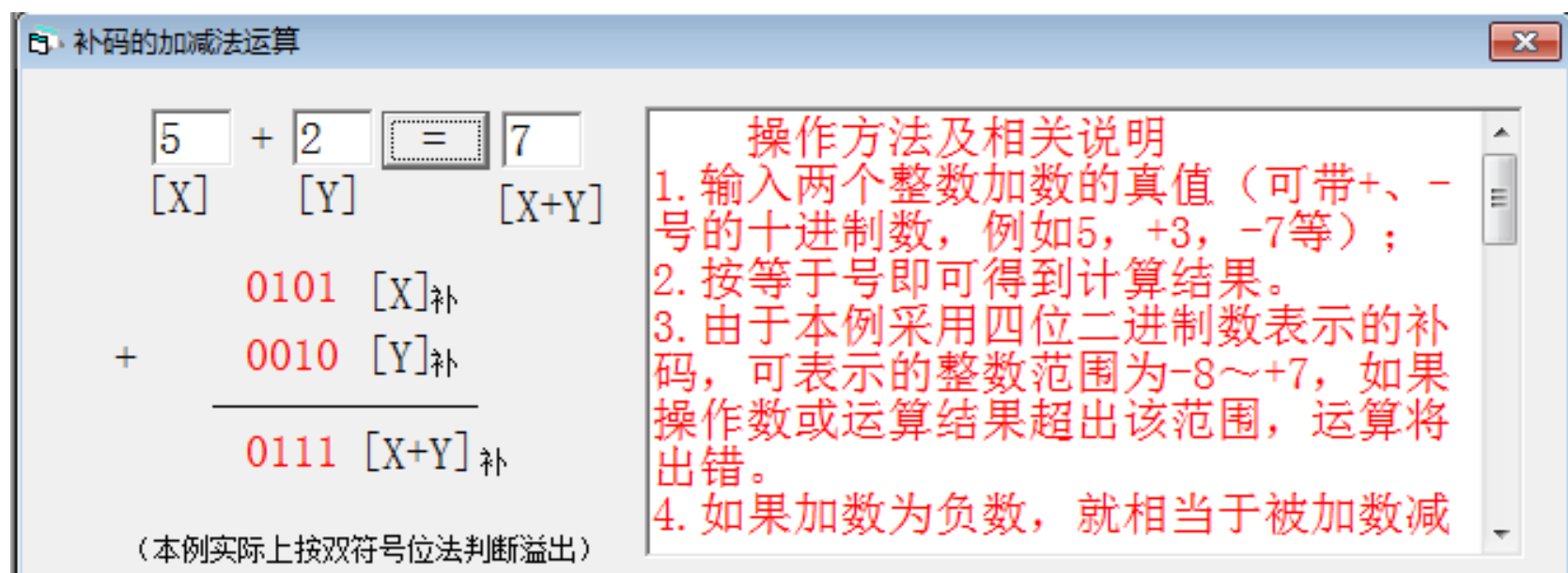


图 1.7 补码的加法运算

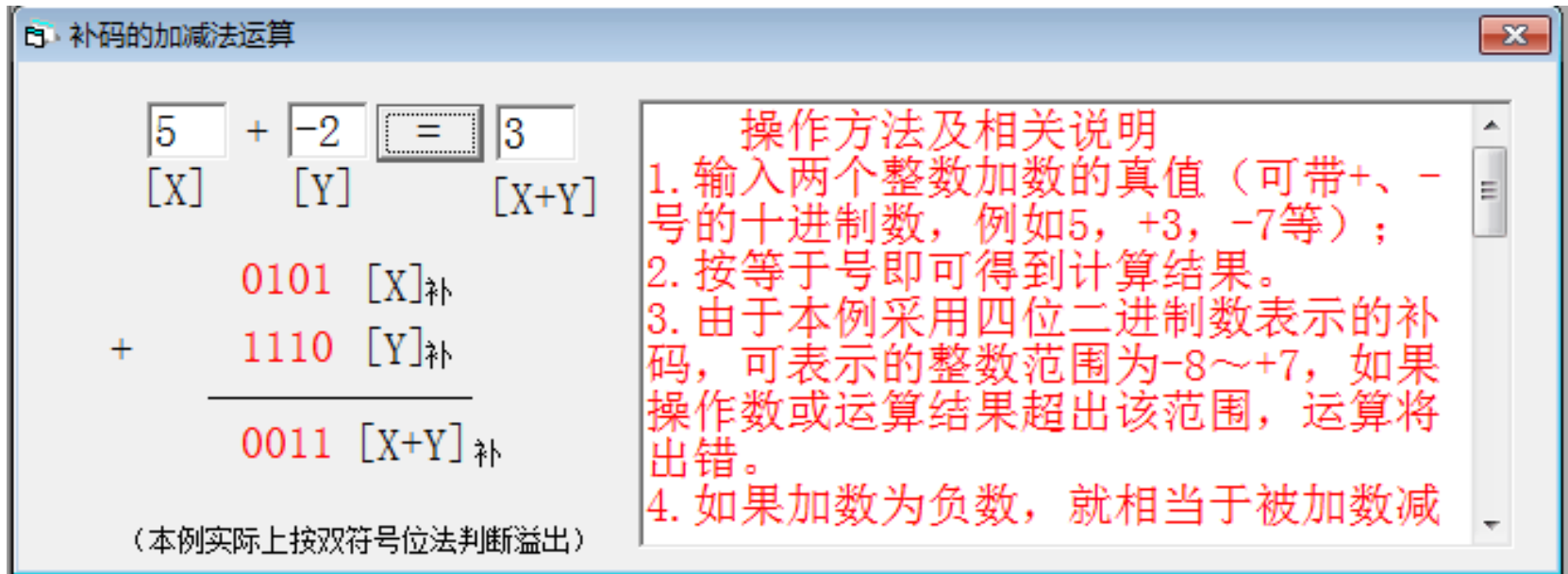


图 1.8 补码的减法运算

3. 原码、补码的乘除法

一 实验目的

- 1. 了解原码、补码一位乘的原理，掌握原码一位乘的算法
- 2. 了解加减交替原、补码一位除的原理，掌握加减交替原码一位除的算法。

二、实验方法及步骤

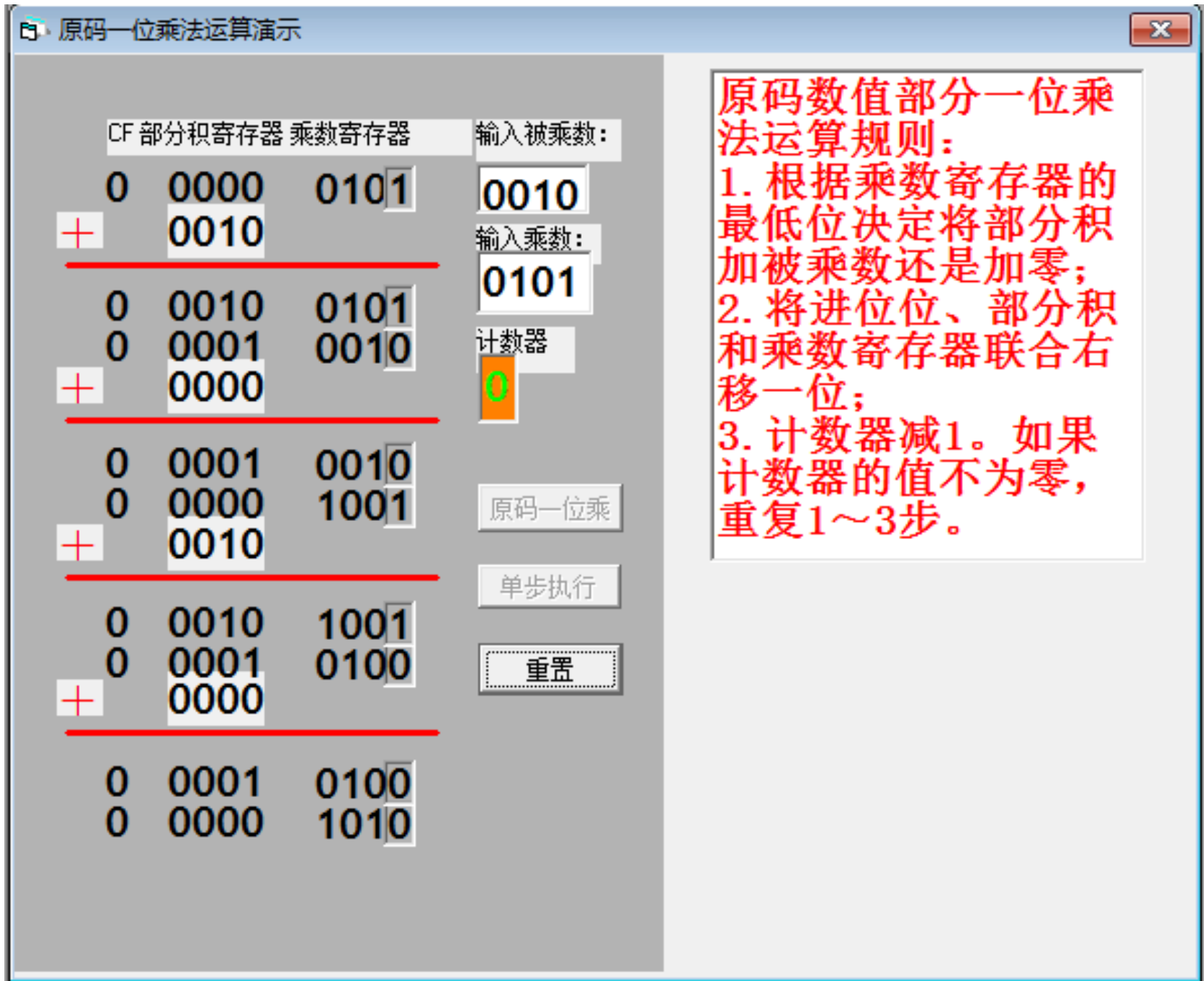


图 1.9 原码一位乘法运算



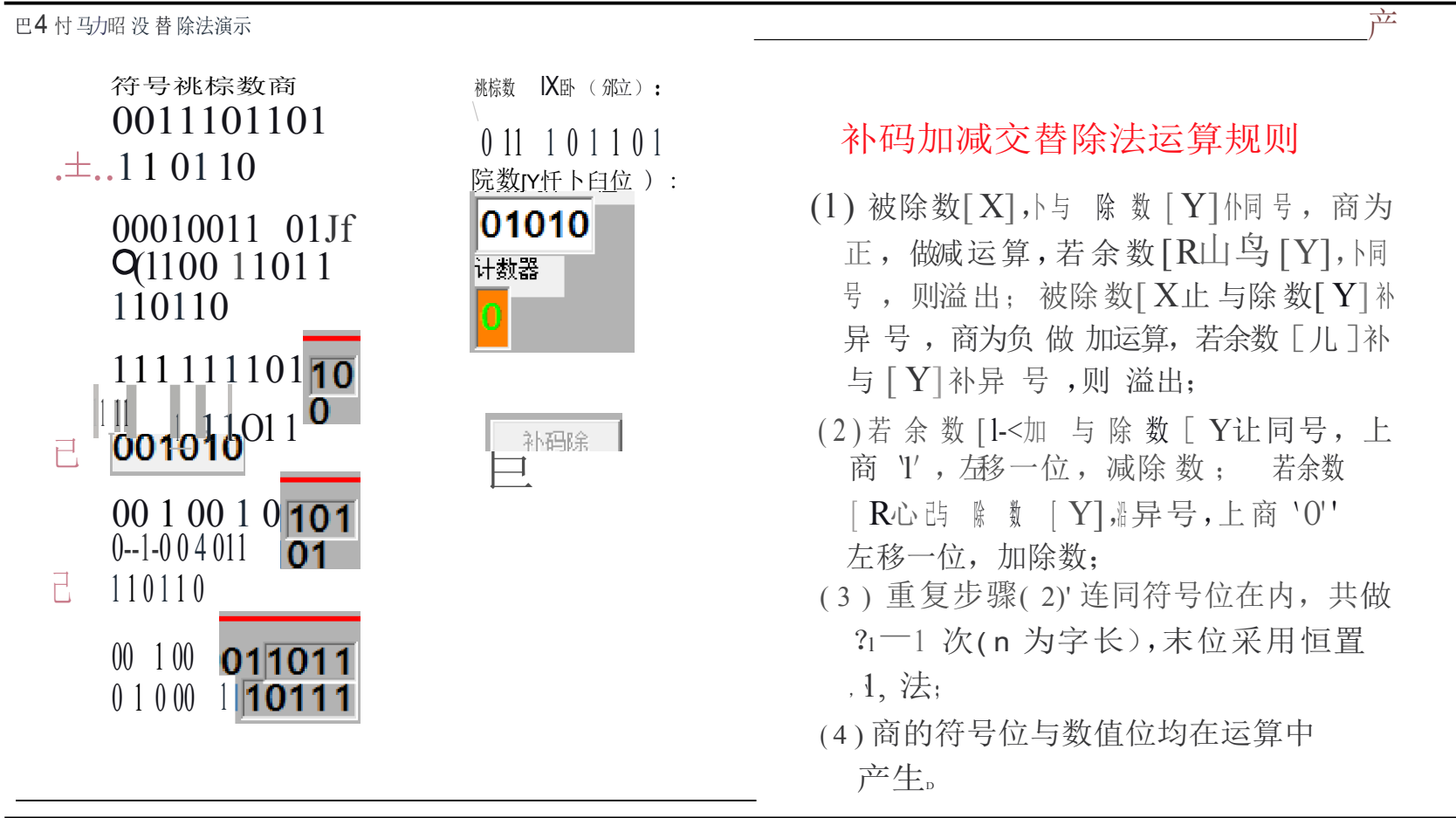


图 1.12 补码加减交替除法

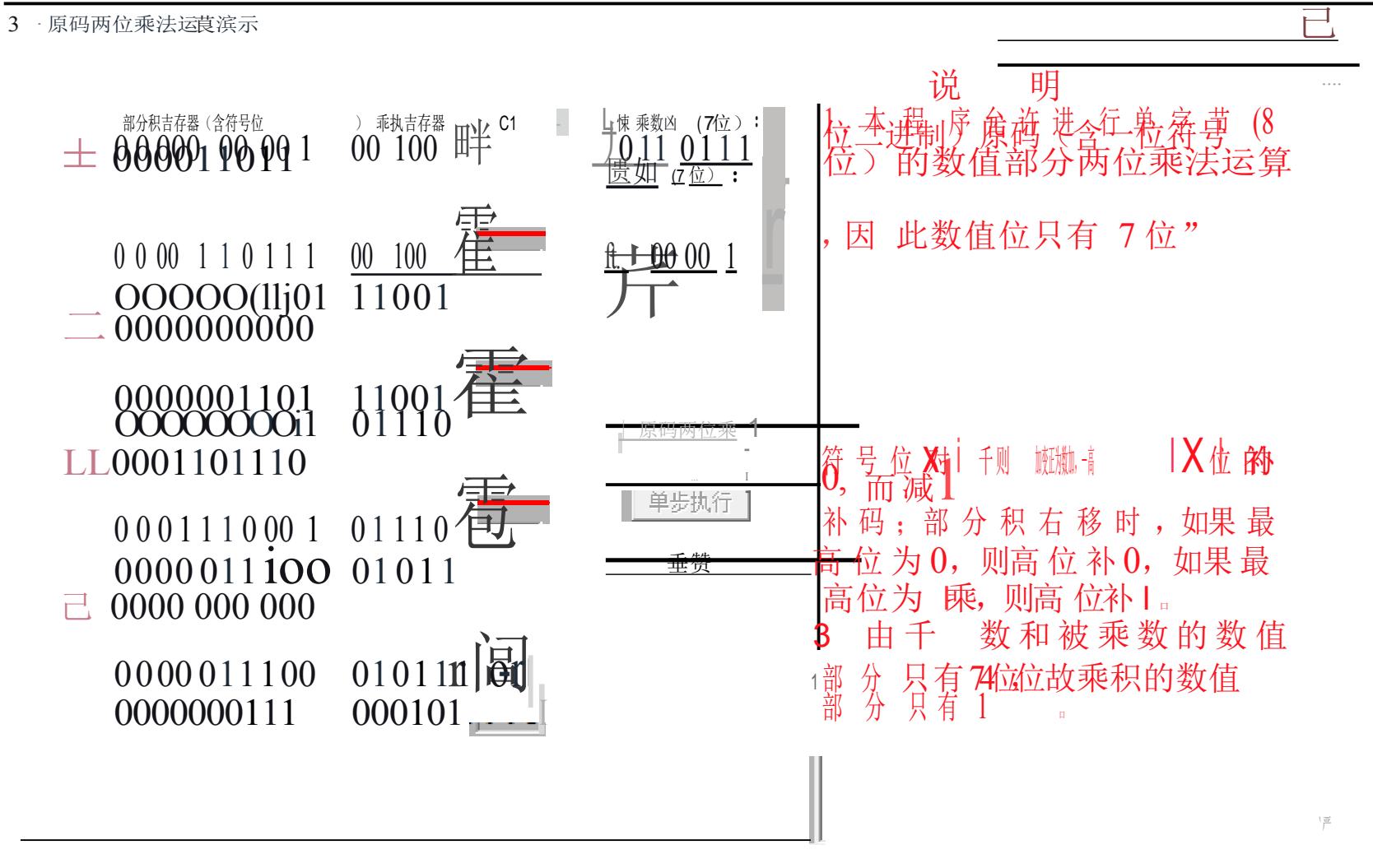


图 1.13 原码两位乘法

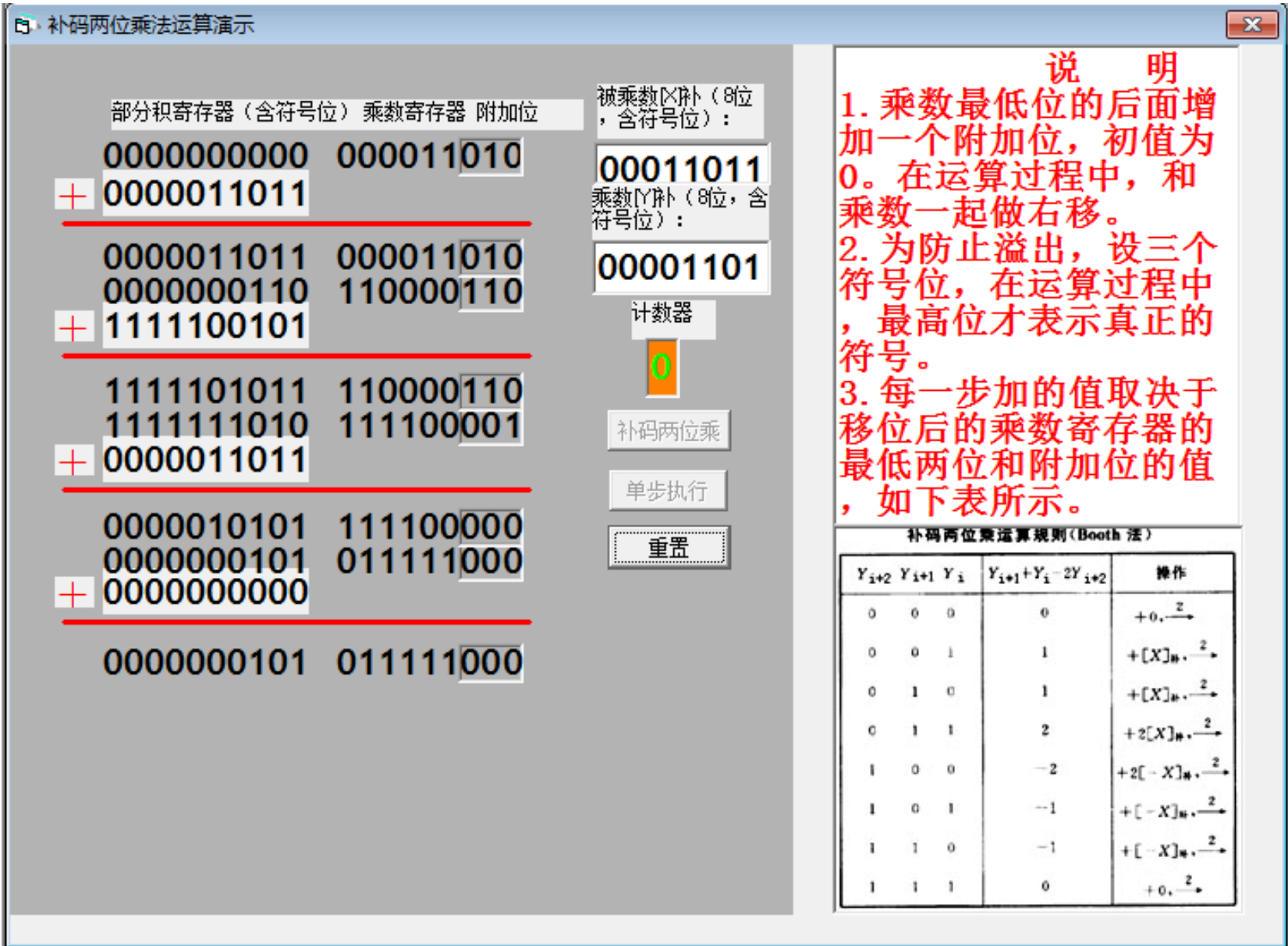


图 1.14 补码两位乘法

三、实验总结

1. 学会了真值与原码、反码、补码和移码之间的转换方法。
2. 学会了 IEEE-754 浮点数的表示及它与十进制之间的相互转换方法。
3. 学会了补码加减法的运算规则及溢出判别
4. 学会了原码、补码一位乘的原理，掌握原码一位乘的算法
5. 学会了加减交替原码、补码一位除的原理，掌握加减交替原码一位除的算法。

实验二 8 位算术逻辑运算实验

一、实验目的

- 1、掌握简单运算器的数据传送通路组成原理。
- 2、验证算术逻辑运算功能发生器 74LS181 的组合功能。

二、实验内容

1、实验原理

实验中所用的运算器数据通路如图所示。其中运算器由两片 74LS181 以并 / 串形成 8 位字长的 ALU 构成。运算器的输出经过一个三态门 74LS245 (U33) 到 ALU01 插座，实验时用 8 芯排线和内部数据总线 BUSD0~D7 插座 BUS1~6 中的任一个相连，内部数据总线通过 LZD0~LZD7 显示灯显示；运算器的两个数据输入端分别由二个锁存器 74LS273 (U29、U30) 锁存，两个锁存器的输入并联后连至插座 ALUBUS，实验时通过 8 芯排线连至外部数据总线 EXD0~D7 插座 EXJ1~EXJ3 中的任一个；参与运算的数据来自于 8 位数据开并 KD0~KD7，并经过一三态门 74LS245 (U51) 直接连至外部数据总线 EXD0~EXD7，通过数据开关输入的数据由 LD0~LD7 显示。

图中算术逻辑运算功能发生器 74LS181 (U31、U32) 的功能控制信号 S3、S2、S1、S0、CN、M 并行相连后连至 SJ2 插座，实验时通过 6 芯排线连至 6 位功能开关插座 UJ2，以手动方式用二进制开关 S3、S2、S1、S0、CN、M 来模拟 74LS181 (U31、U32) 的功能控制信号 S3、S2、S1、S0、CN、M；其它电平控制信号 LDDR1、LDDR2、ALUB'、SWB'、以手动方式用二进制开关 LDDR1、LDDR2、ALUB、SWB 来模拟，这几个信号有自动和手动两种方式产生，通过跳线器切换，其中 ALUB'、SWB'、为低电平有效，LDDR1、LDDR2 为高电平有效。

另有信号 T4 为脉冲信号，在手动方式进行实验时，只需将跳线器 J23 上 T4 与手动脉冲发生开关的输出端 SD 相连，按动手动脉冲开关，即可获得实验所需的单脉冲。

以上内容仅为本文档的试下载部分，为可阅读页数的一半内容。如要下载或阅读全文，请访问：
<https://d.book118.com/368052017014006100>