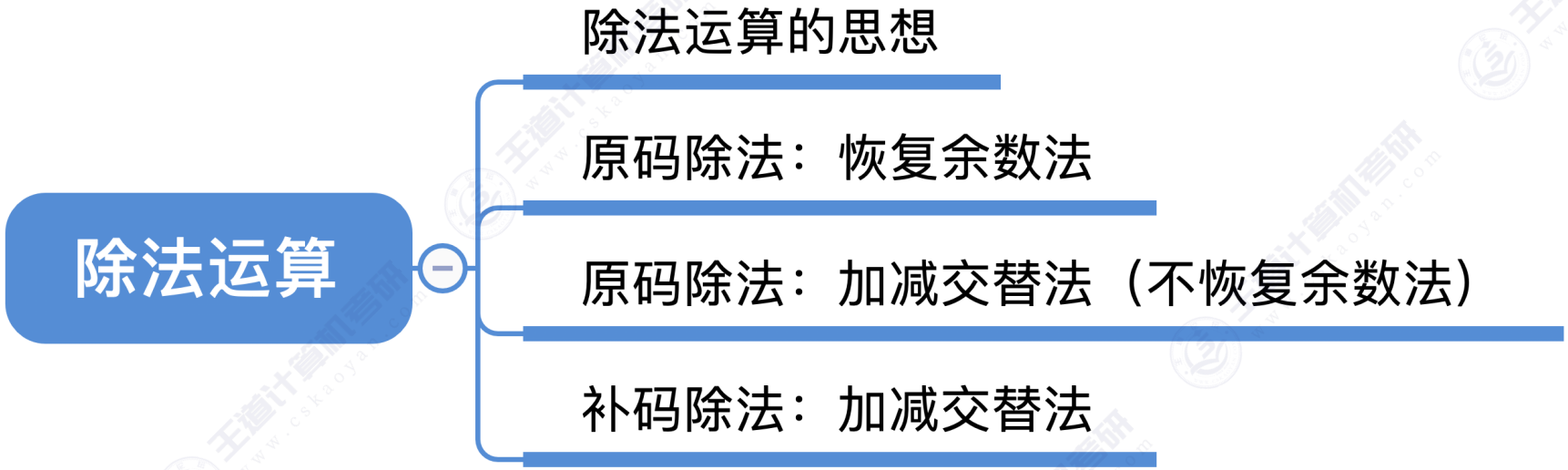
本节内容

定点数

原码除法运算

本节总览



手算除法（十进制）

r 进制： $K_n\ K_{n-1}\ \dots\ K_2\ K_1\ K_0 K_{-1} K_{-2}\ \dots\ K_{-m}$

$$= K_n \times r^n + K_{n-1} \times r^{n-1} + \cdots + K_2 \times r^2 + K_1 \times r^1 + K_0 \times r^0 + K_{-1} \times r^{-1} + K_{-2} \times r^{-2} + \dots + K_{-m} \times r^{-m}$$

0.211÷0.985=?

你怎么这个亚子

```
        0.214
   ------------
985 ) 211
      000
   ------------
      2110
      1970
   ------------
       1400
        985
   ------------
        4150
        3940
   ------------
         210
```

```
          0.214
   ------------
0.985 ) 0.211
        0.000
   ------------
        0.2110
        0.1970
   ------------
        0.01400
        0.00985
   ------------
        0.004150
        0.003940
   ------------
        0.000210
```

0.214 $= 2\times10^{-1} + 1\times10^{-2} + 4\times10^{-3}$

0.985 $= 985 \times 10^{-3}$

$0.985\times0.214 = (985\times2\times10^{-4}) + (985\times1\times10^{-5}) + (985\times4\times10^{-6})$

$=0.1970 + 0.00985 + 0.00394$

x÷y=a (余数b) → x=ay+b

0.211=0.985*0.214 + 0.000210

手算除法（二进制）

符号位	绝对值

两个正数相除

设机器字长为5位（含1位符号位，$n=4$），$x=0.1011$，$y=0.1101$，求x/y

$(0.1011\times2^4)\div(0.1101\times2^4)$

```
          0.1101
01101 ) 01011
        00000
        -----
         10110
         01101
        -------
          10010
          01101
         -------
           01010
           00000
          -------
            10100
            01101
           -------
             0111
```

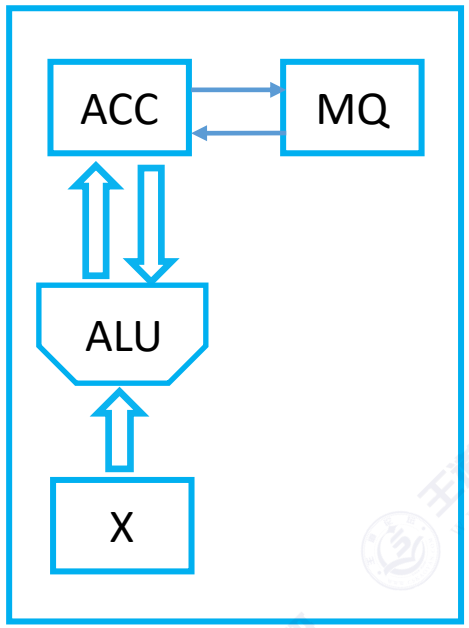
```
          0.1101
0.1101 ) 0.1011
         0.0000
         ------
          0.10110
          0.01101
         --------
           0.010010
           0.001101
          ---------
            0.0001010
            0.0000000
           ----------
             0.00010100
             0.00001101
            -----------
              0.00000111
```

规律：忽略小数点，每确定一位商，进行一次减法，得到4位余数，在余数末尾补0，再确定下一位商。确定5位商即可停止（机器字长为5位）

x/y结果为0.1101，余数为0.00000111

穿越：运算器的基本组成

运算器



运算器：用于实现算术运算（如：加减乘除）、逻辑运算（如：与或非）

ACC：　累加器，用于存放操作数，或运算结果。
MQ：　乘商寄存器，在乘、除运算时，用于存放操作数或运算结果。
X：　通用的操作数寄存器，用于存放操作数
ALU：　算术逻辑单元，通过内部复杂的电路实现算数运算、逻辑运算

Accumulator
Multiple-Quotient Register
Arithmetic and Logic Unit

	加	减	乘	除
ACC	被加数、和	被减数、差	乘积高位	被除数、余数
MQ			乘数、乘积低位	商
X	加数	减数	被乘数	除数

原码除法：恢复余数法

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

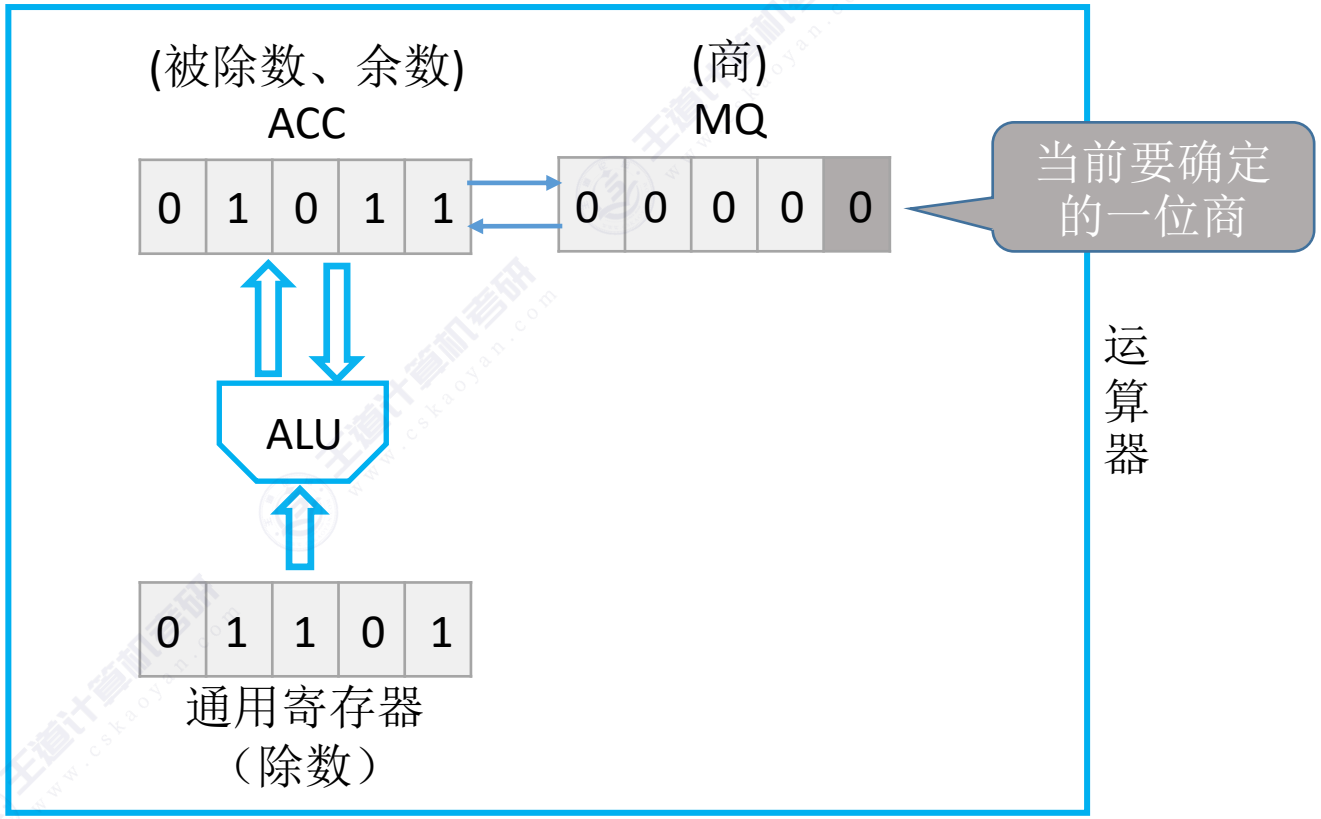
符号单独处理：符号位 = $x_s \oplus y_s$

数值位取绝对值进行除法计算

实现方法：上商0/1，得到余数，余数末尾补0

```
          0.1101
01101 ⌠ 01011
        00000
       ------
        10110
        01101
       ------
        10010
        01101
       ------
        01010
        00000
       ------
        10100
        01101
       ------
         0111
```

手算时，每一位商取0/1 是通过判断当前余数和除数的大小确定的



计算机很傻，会先默认上商1，如果搞错了再改上商0。并“恢复余数”

原码除法：恢复余数法

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

符号单独处理：符号位 = $x_s \oplus y_s$

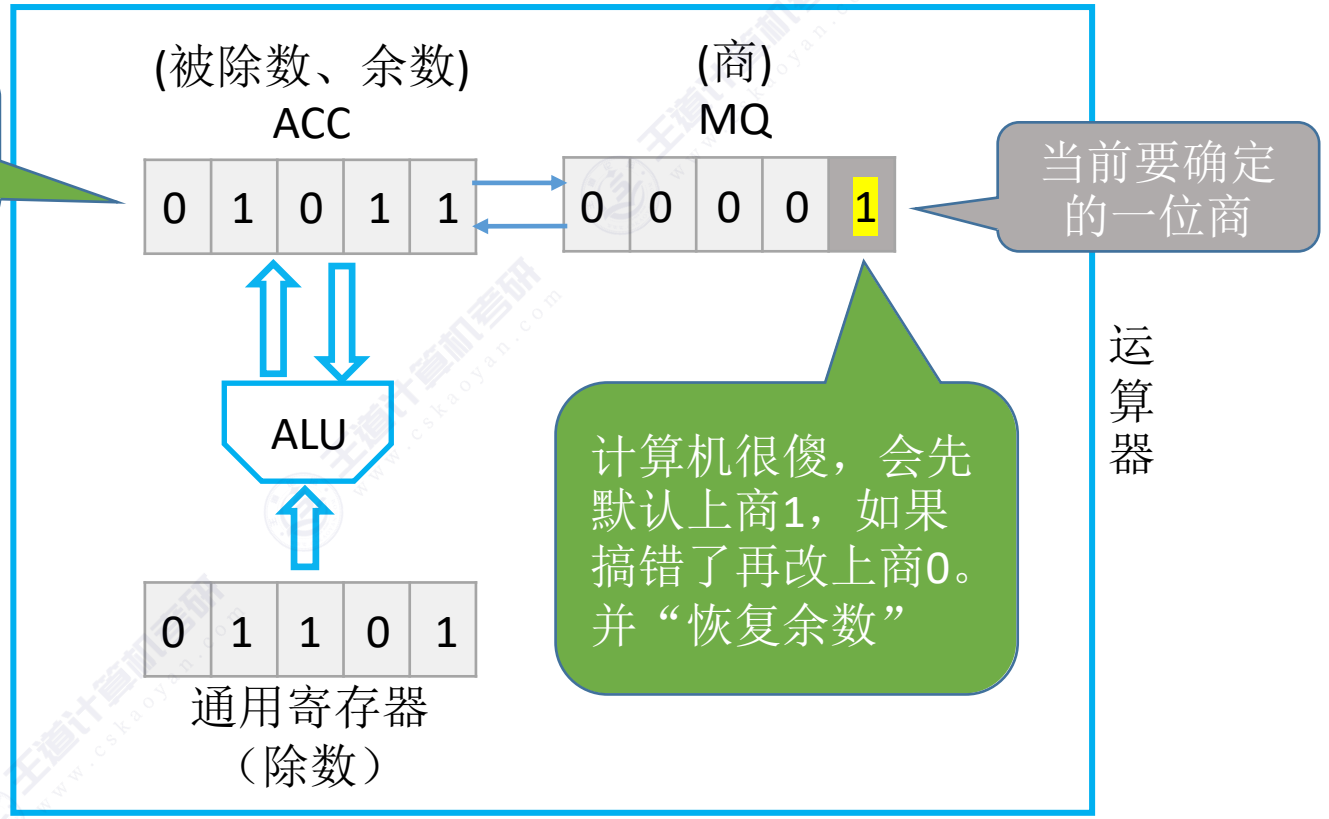
数值位取绝对值进行除法计算

实现方法：上商0/1，得到余数，余数末尾补0

```
          0.1101
01101 ⌠ 01011
        00000
       -------
        10110
        01101
       -------
        10010
        01101
       -------
        01010
        00000
       -------
        10100
        01101
       -------
         0111
```

求余数：
(ACC)-(除数)→ACC

(ACC)+ $[-|y|]_{补}$→ACC
01011+10011 = 11110



计算机很傻，会先默认上商1，如果搞错了再改上商0。并“恢复余数”

原码除法：恢复余数法

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

符号单独处理：符号位 = $x_s \oplus y_s$

数值位取绝对值进行除法计算

实现方法：上商0/1，得到余数，余数末尾补0

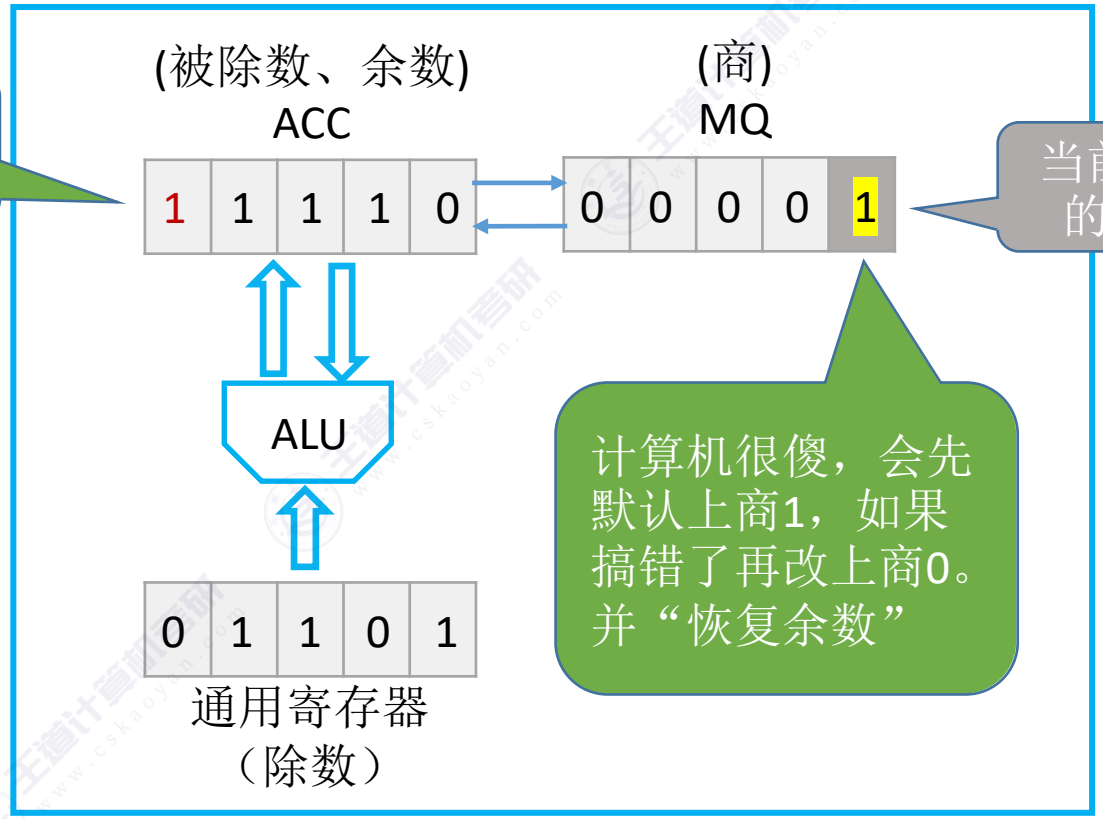
```
          0.1101
01101 ⎞ 01011
        00000
        -----
         10110
         01101
        -----
         10010
         01101
        -----
         01010
         00000
        -----
         10100
         01101
        -----
          0111
```

求余数：
(ACC)-(除数)→ACC

(ACC)+ $[-|y|]_{补}$→ACC
01011+10011 = 11110

相减结果是个负数，说明应该上商0

计算机很傻，会先默认上商1，如果搞错了再改上商0。并“恢复余数”



原码除法：恢复余数法

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

符号单独处理：符号位 = $x_s \oplus y_s$

数值位取绝对值进行除法计算

实现方法：上商0/1，得到余数，余数末尾补0

```
          0.1101
01101 ⎛ 01011
        00000
        -----
        10110
        01101
        -----
        10010
        01101
        -----
        01010
        00000
        -----
        10100
        01101
        -----
        0111
```

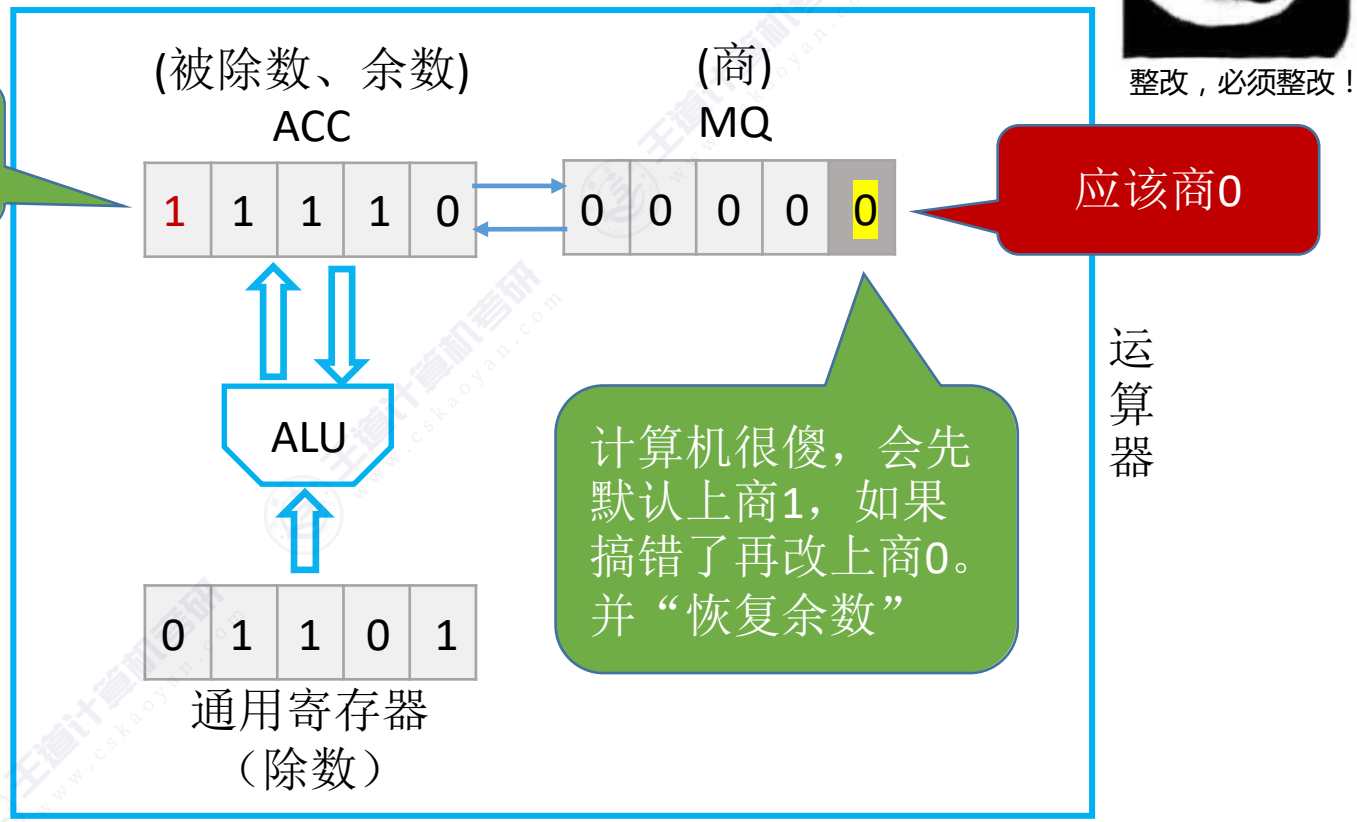
恢复余数：(ACC)+(除数)→ACC

(ACC)+ $[-|y|]_{补}$→ACC

01011+10011 = 11110

(ACC)+ $[|y|]_{补}$→ACC

11110+01101 = 01011



计算机很傻，会先默认上商1，如果搞错了再改上商0。并“恢复余数”

原码除法：恢复余数法

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

符号单独处理：符号位 = $x_s \oplus y_s$

数值位取绝对值进行除法计算

实现方法：上商0/1，得到余数，余数末尾补0

```
          0.1101
01101 ⌠ 01011
        00000
        -----
        10110
        01101
        -----
        10010
        01101
        -----
        01010
        00000
        -----
        10100
        01101
        -----
        0111
```

恢复余数：(ACC)+(除数)→ACC

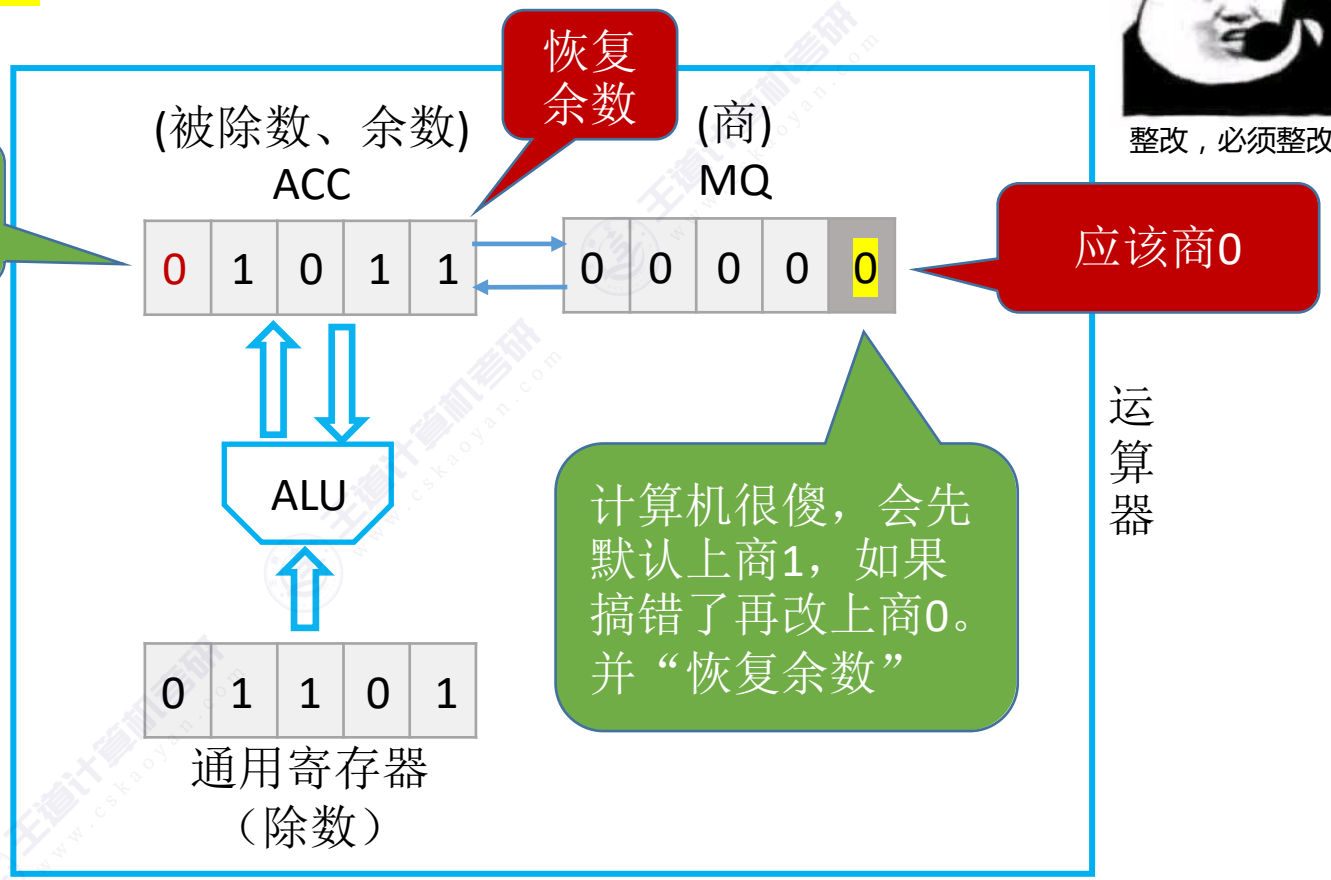
(ACC)+ $[-|y|]_{补}$→ACC

01011+10011 = 11110

(ACC)+ $[|y|]_{补}$→ACC

11110+01101 = 01011

计算机很傻，会先默认上商1，如果搞错了再改上商0。并“恢复余数”



原码除法：恢复余数法

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

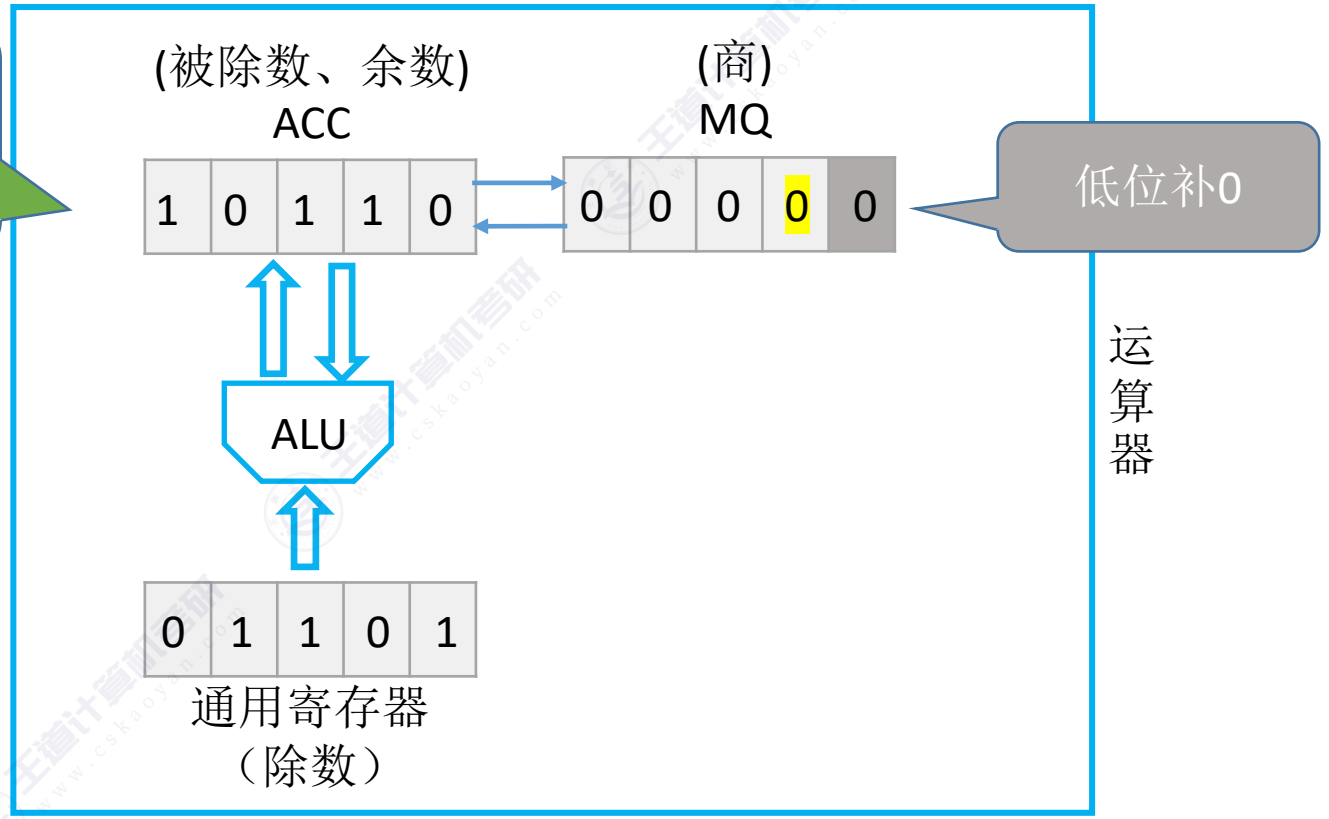
符号单独处理：符号位 = $x_s \oplus y_s$

数值位取绝对值进行除法计算

实现方法：上商0/1，得到余数，余数末尾补0

```
          0.1101
      ┌─────────
01101 ) 01011
        00000
        -----
        10110
        01101
        -----
        10010
        01101
        -----
        01010
        00000
        -----
        10100
        01101
        -----
        0111
```

ACC、MQ整体“逻辑左移”。ACC高位丢弃，MQ低位补0



计算机很傻，会先默认上商1，如果搞错了再改上商0。并“恢复余数”

原码除法：恢复余数法

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

符号单独处理：符号位 = $x_s \oplus y_s$

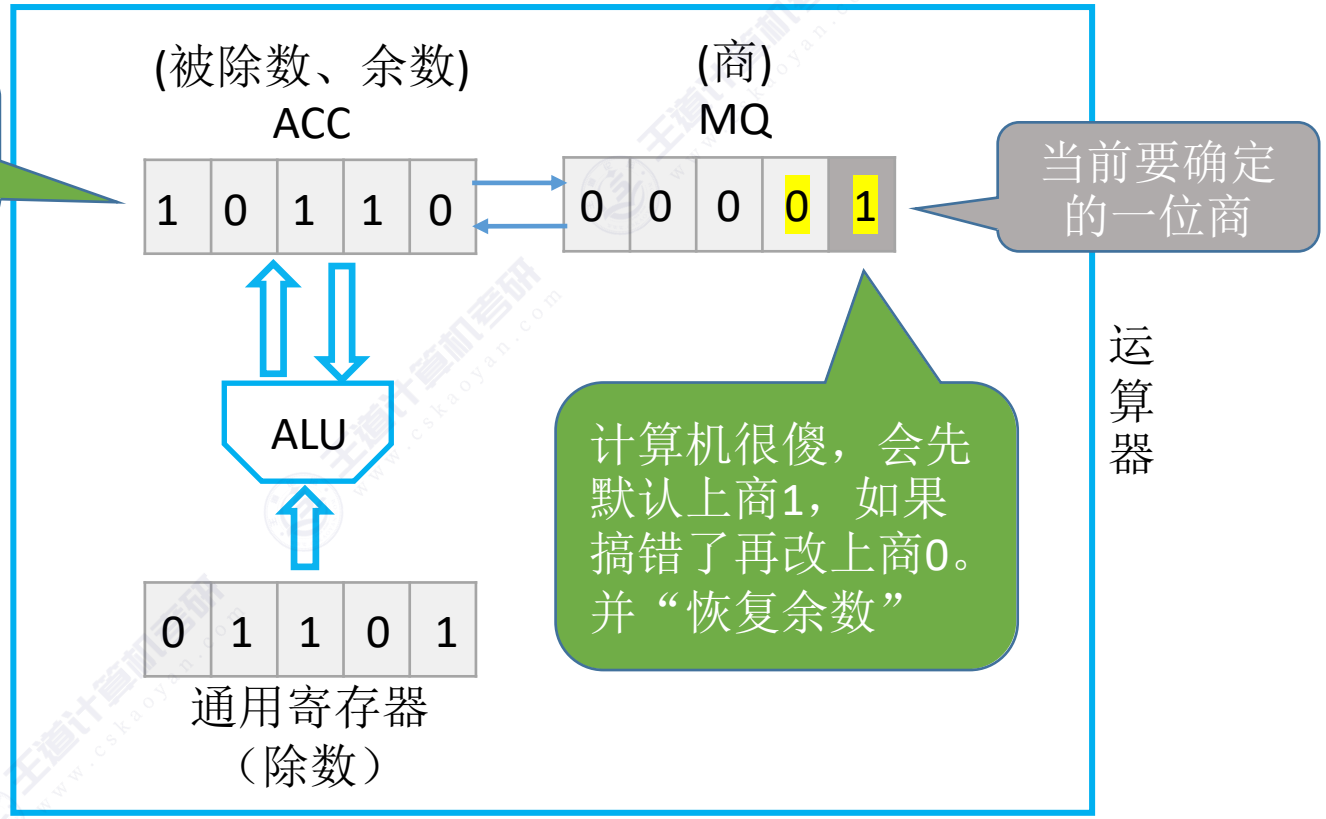
数值位取绝对值进行除法计算

实现方法：上商0/1，得到余数，余数末尾补0

```
          0.1101
01101 ⌠ 01011
        00000
       -------
        10110
        01101
       -------
        10010
        01101
       -------
        01010
        00000
       -------
        10100
        01101
       -------
         0111
```

求余数：
(ACC)-(除数)→ACC

(ACC)+ $[-|y|]_{补}$→ACC
10110+10011 = 01001



计算机很傻，会先默认上商1，如果搞错了再改上商0。并“恢复余数”

原码除法：恢复余数法

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

符号单独处理：符号位 = $x_s \oplus y_s$

数值位取绝对值进行除法计算

实现方法：上商0/1，得到余数，余数末尾补0

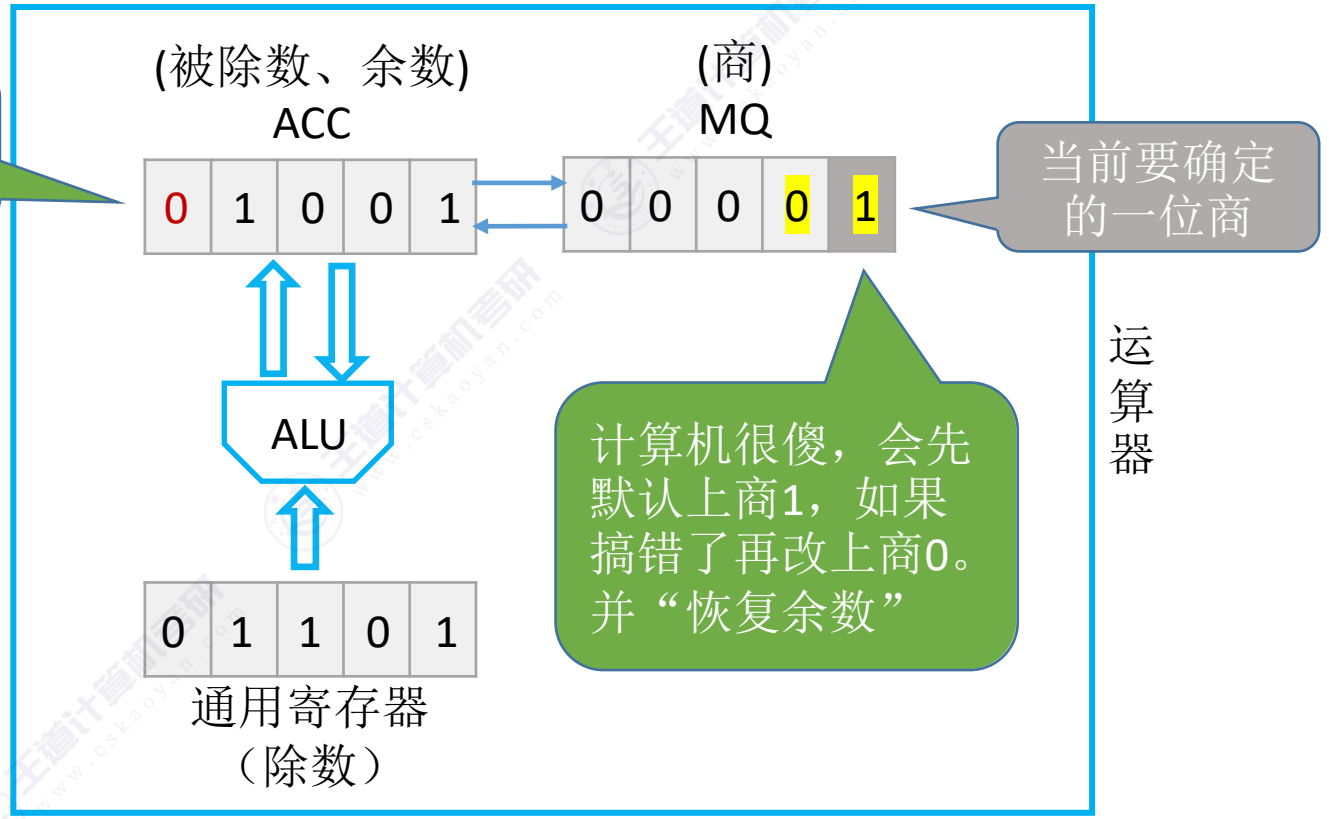
```
          0.1101
01101 ⌠ 01011
        00000
       ------
        10110
        01101
       ------
        10010
        01101
       ------
        01010
        00000
       ------
        10100
        01101
       ------
         0111
```

求余数：
(ACC)-(除数)→ACC

(ACC)+ $[-|y|]_{补}$→ACC
10110+10011 = 01001

相减结果是个正数，上商 1 是没错滴~

计算机很傻，会先默认上商1，如果搞错了再改上商0。并“恢复余数”



原码除法：恢复余数法

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

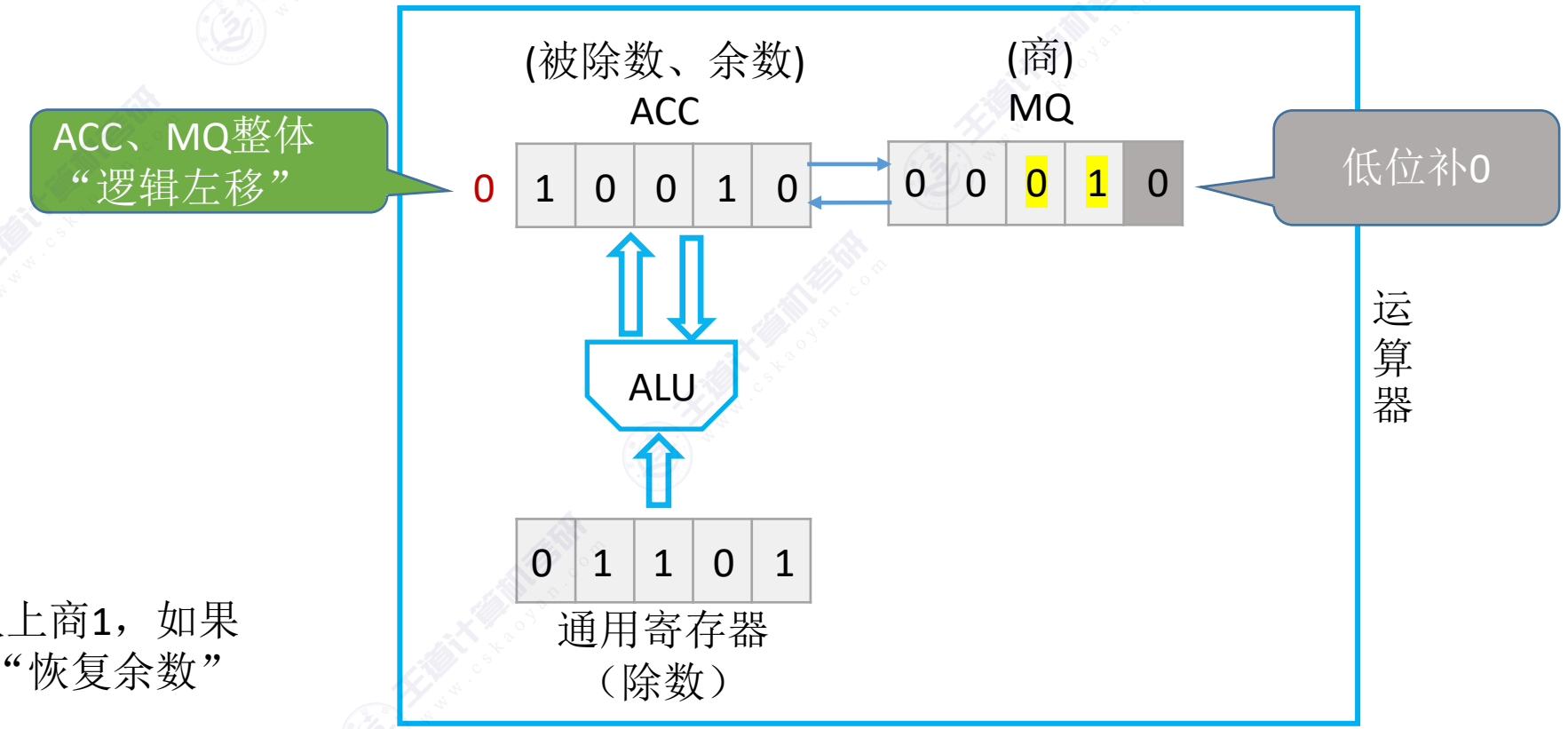
$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

符号单独处理：符号位 = $x_s \oplus y_s$

数值位取绝对值进行除法计算

实现方法：上商0/1，得到余数，余数末尾补0

```
          0.1101
01101 ⌠ 01011
        00000
       ------
        10110
        01101
       ------
        10010
        01101
       ------
        01010
        00000
       ------
        10100
        01101
       ------
         0111
```



计算机很傻，会先默认上商1，如果搞错了再改上商0。并“恢复余数”

原码除法：恢复余数法

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

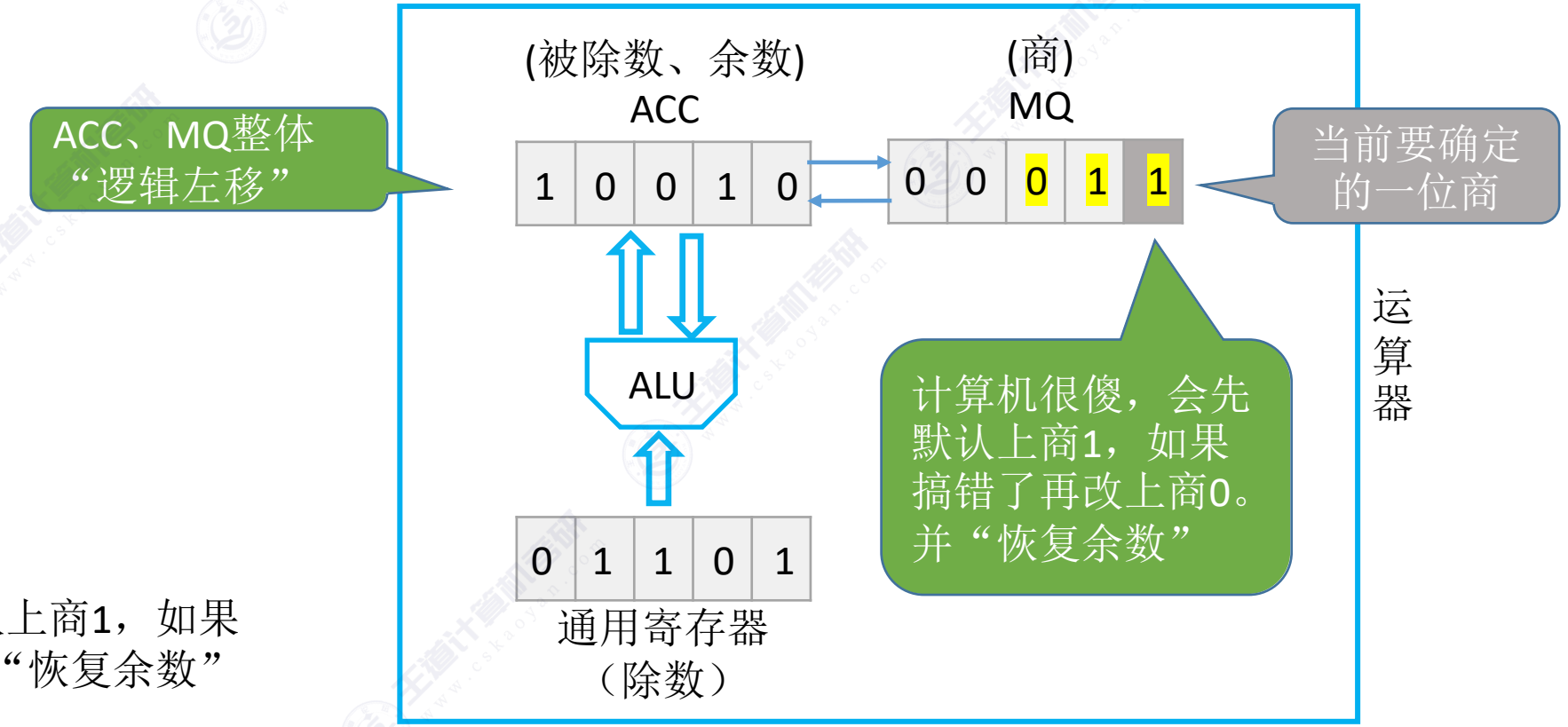
符号单独处理：符号位 = $x_s \oplus y_s$

数值位取绝对值进行除法计算

实现方法：上商0/1，得到余数，余数末尾补0

```
          0.1101
      ┌─────────
01101 ) 01011
        00000
        ------
        10110
        01101
        ------
        10010
        01101
        ------
        01010
        00000
        ------
        10100
        01101
        ------
        0111
```

计算机很傻，会先默认上商1，如果搞错了再改上商0。并“恢复余数”



原码除法：恢复余数法

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

符号单独处理：符号位 = $x_s \oplus y_s$

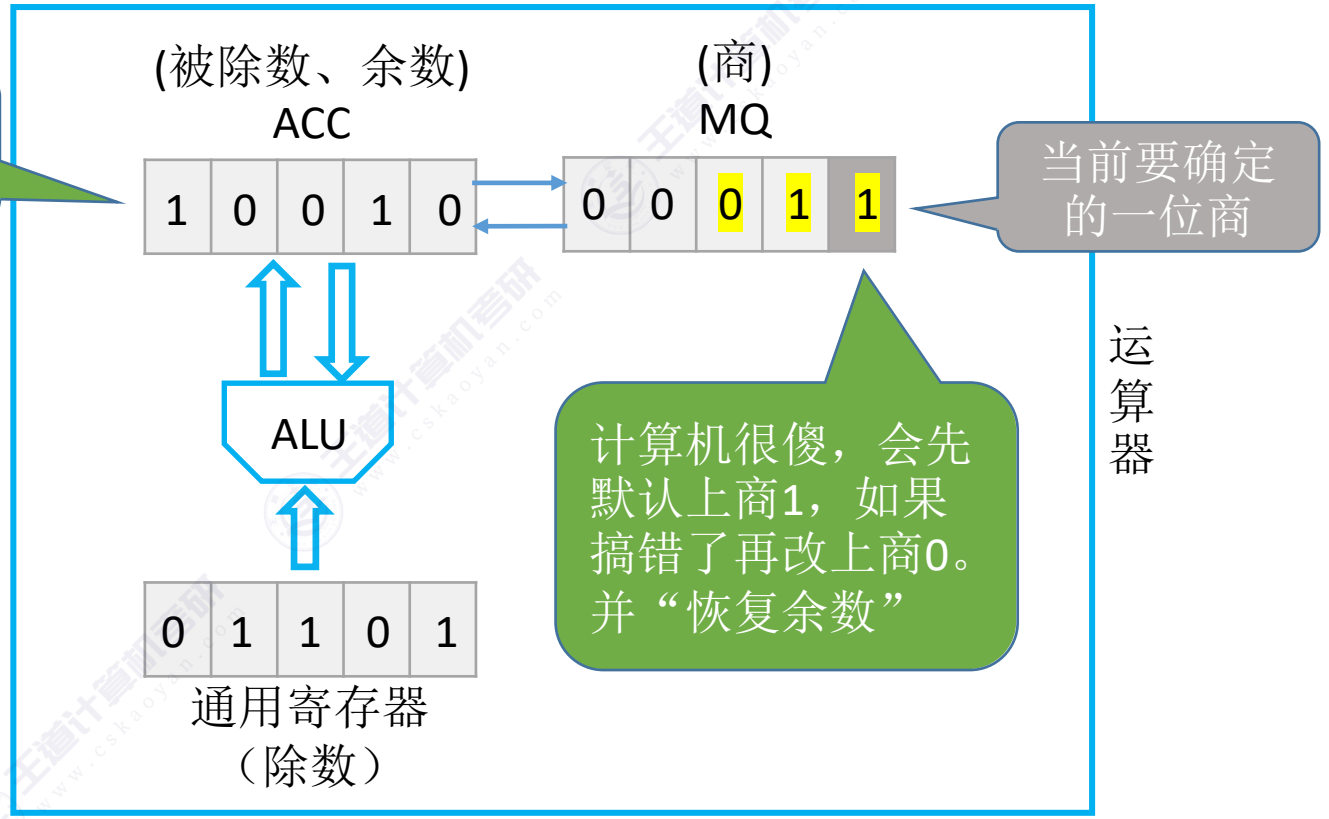
数值位取绝对值进行除法计算

实现方法：上商0/1，得到余数，余数末尾补0

```
          0.1101
01101 ⎞ 01011
        00000
        -----
        10110
        01101
        -----
        10010
        01101
        -----
        01010
        00000
        -----
        10100
        01101
        -----
        0111
```

求余数：
(ACC)-(除数)→ACC

(ACC)+ $[-|y|]_{补}$→ACC
10010+10011 = 00101



计算机很傻，会先默认上商1，如果搞错了再改上商0。并“恢复余数”

原码除法：恢复余数法

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

符号单独处理：符号位 = $x_s \oplus y_s$

数值位取绝对值进行除法计算

实现方法：上商0/1，得到余数，余数末尾补0

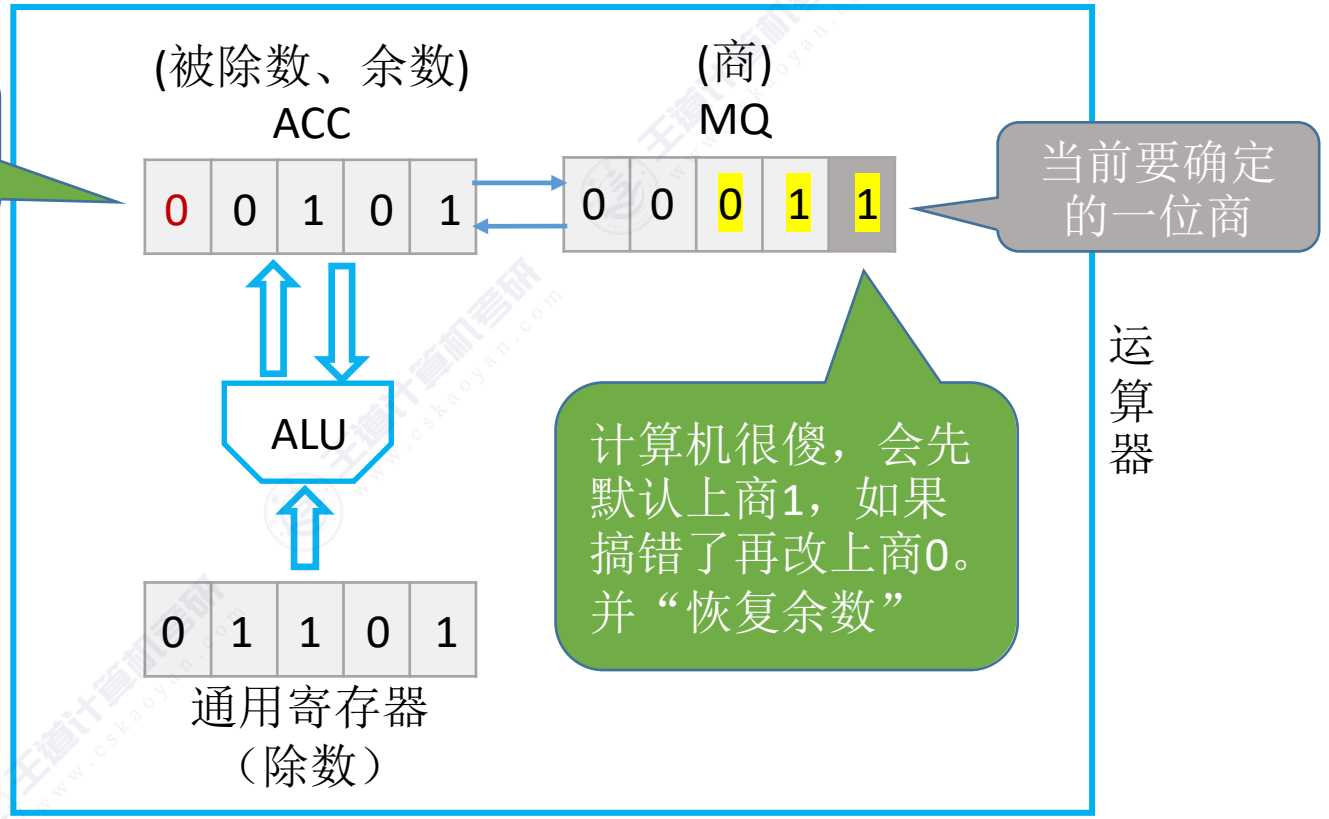
```
          0.1101
01101 ⌠ 01011
        00000
       ------
        10110
        01101
       ------
        10010
        01101
       ------
        01010
        00000
       ------
        10100
        01101
       ------
         0111
```

求余数：
(ACC)-(除数)→ACC

(ACC)+ $[-|y|]_{补}$→ACC
10010+10011 = 00101

相减结果是个正数，上商 1 是没错滴~

计算机很傻，会先默认上商1，如果搞错了再改上商0。并“恢复余数”



原码除法：恢复余数法

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

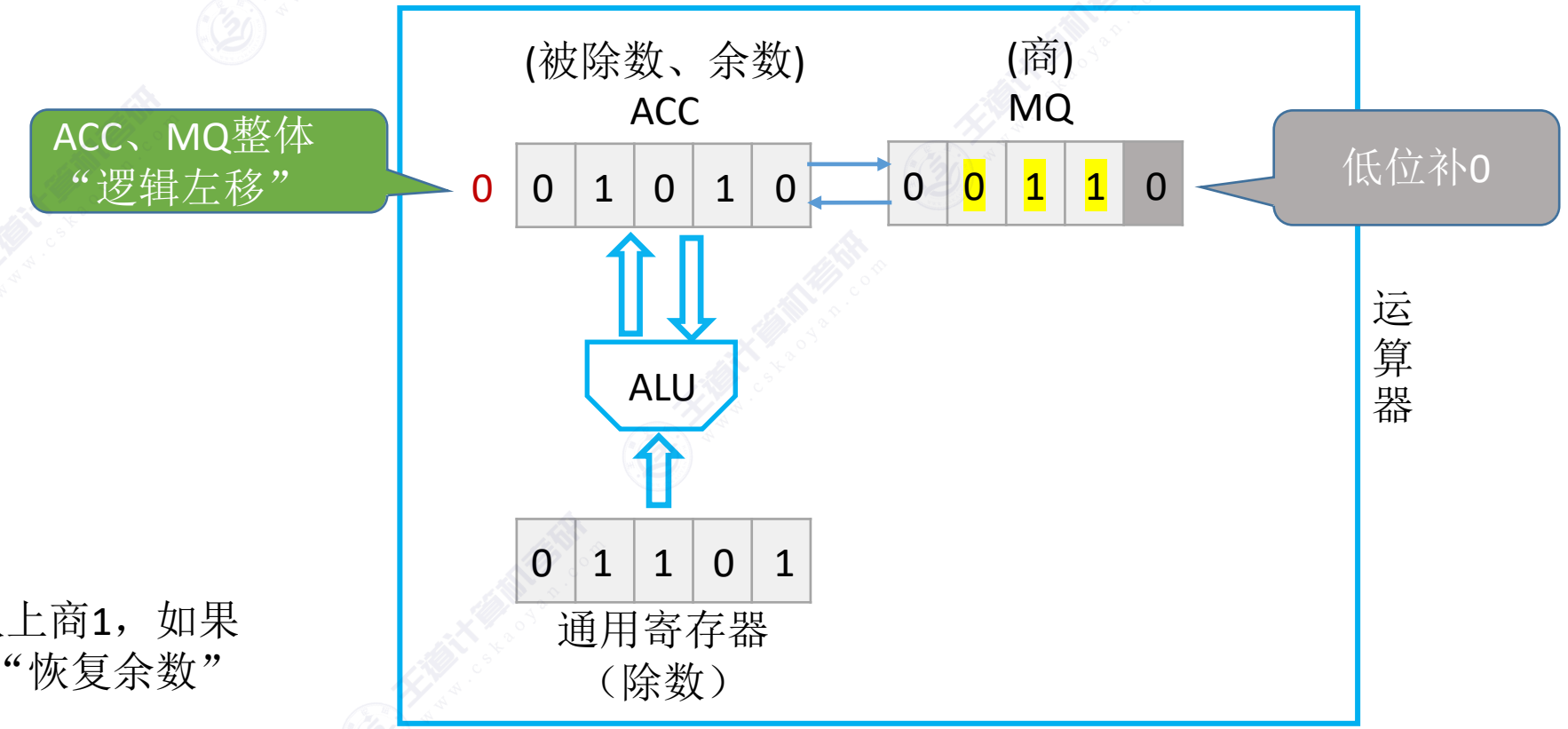
$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

符号单独处理：符号位 = $x_s \oplus y_s$

数值位取绝对值进行除法计算

实现方法：上商0/1，得到余数，余数末尾补0

```
          0.1101
01101 ⌠ 01011
        00000
       ------
        10110
        01101
       ------
        10010
        01101
       ------
        01010
        00000
       ------
        10100
        01101
       ------
         0111
```



计算机很傻，会先默认上商1，如果搞错了再改上商0。并“恢复余数”

原码除法：恢复余数法

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

符号单独处理：符号位 = $x_s \oplus y_s$

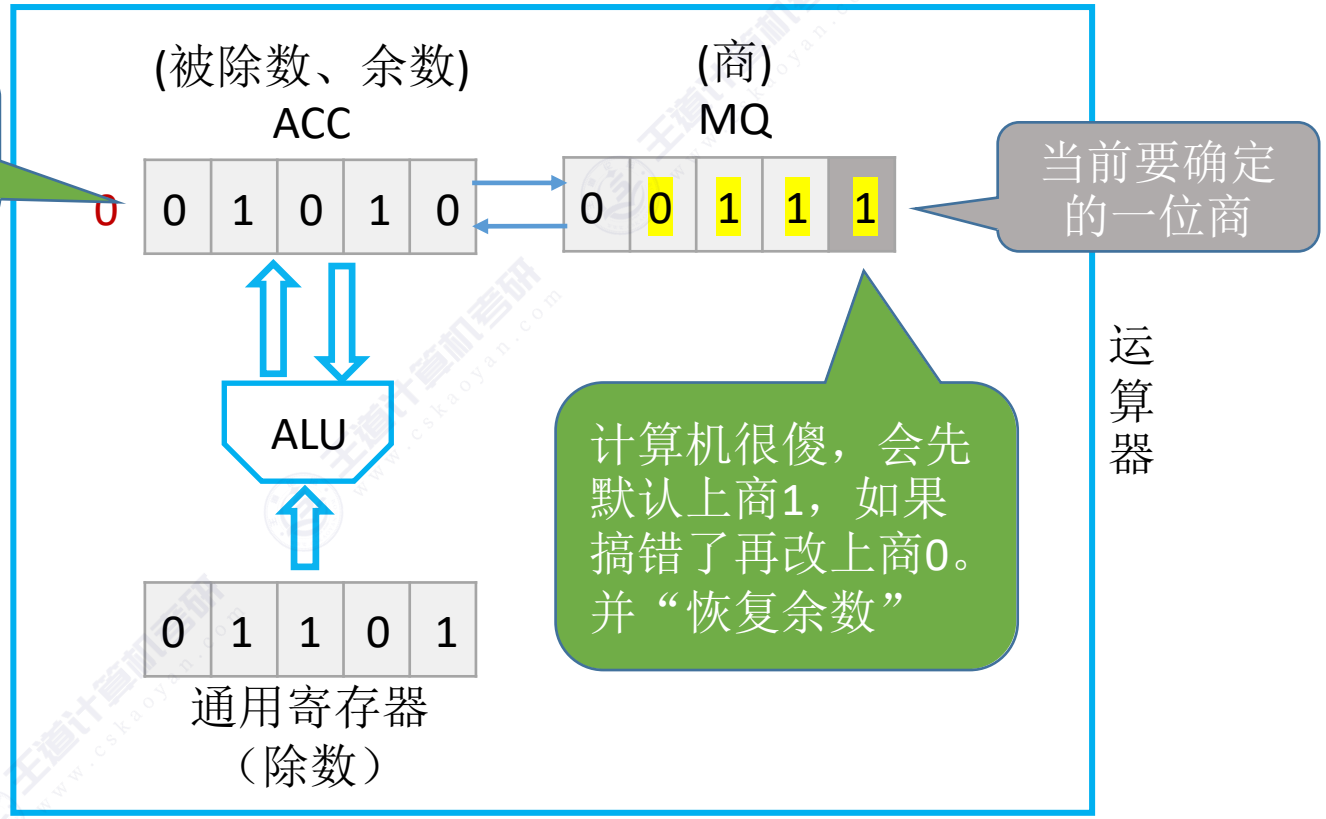
数值位取绝对值进行除法计算

实现方法：上商0/1，得到余数，余数末尾补0

```
          0.1101
01101 ⌠ 01011
        00000
       -------
        10110
        01101
       -------
        10010
        01101
       -------
        01010
        00000
       -------
        10100
        01101
       -------
         0111
```

求余数：
(ACC)-(除数)→ACC

(ACC)+ $[-|y|]_{补}$→ACC
01010+10011 = 11101



计算机很傻，会先默认上商1，如果搞错了再改上商0。并“恢复余数”

原码除法：恢复余数法

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

符号单独处理：符号位 = $x_s \oplus y_s$

数值位取绝对值进行除法计算

实现方法：上商0/1，得到余数，余数末尾补0

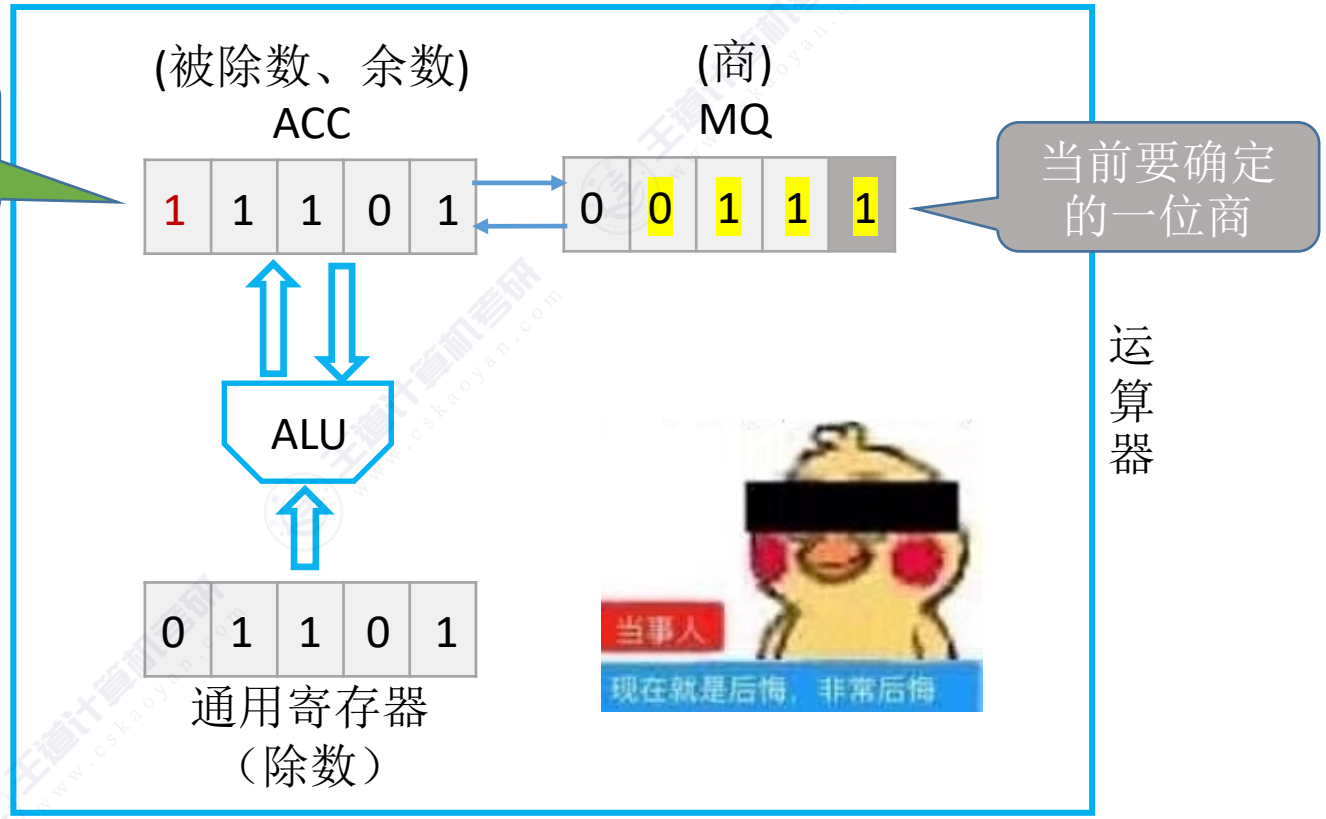
```
          0.1101
01101 ⌠ 01011
        00000
        -----
        10110
        01101
        -----
        10010
        01101
        -----
        01010
        00000
        -----
        10100
        01101
        -----
         0111
```

求余数：
(ACC)-(除数)→ACC

(ACC)+ $[-|y|]_{补}$→ACC
01010+10011 = 11101

相减结果是个负数，不该上商 1

计算机很傻，会先默认上商1，如果搞错了再改上商0。并“恢复余数”



原码除法：恢复余数法

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

符号单独处理：符号位 = $x_s \oplus y_s$

数值位取绝对值进行除法计算

实现方法：上商0/1，得到余数，余数末尾补0

```
          0.1101
01101 ⌠ 01011
        00000
        ------
        10110
        01101
        ------
        10010
        01101
        ------
        01010
        00000
        ------
        10100
        01101
        ------
        0111
```

恢复余数：(ACC)+(除数)→ACC

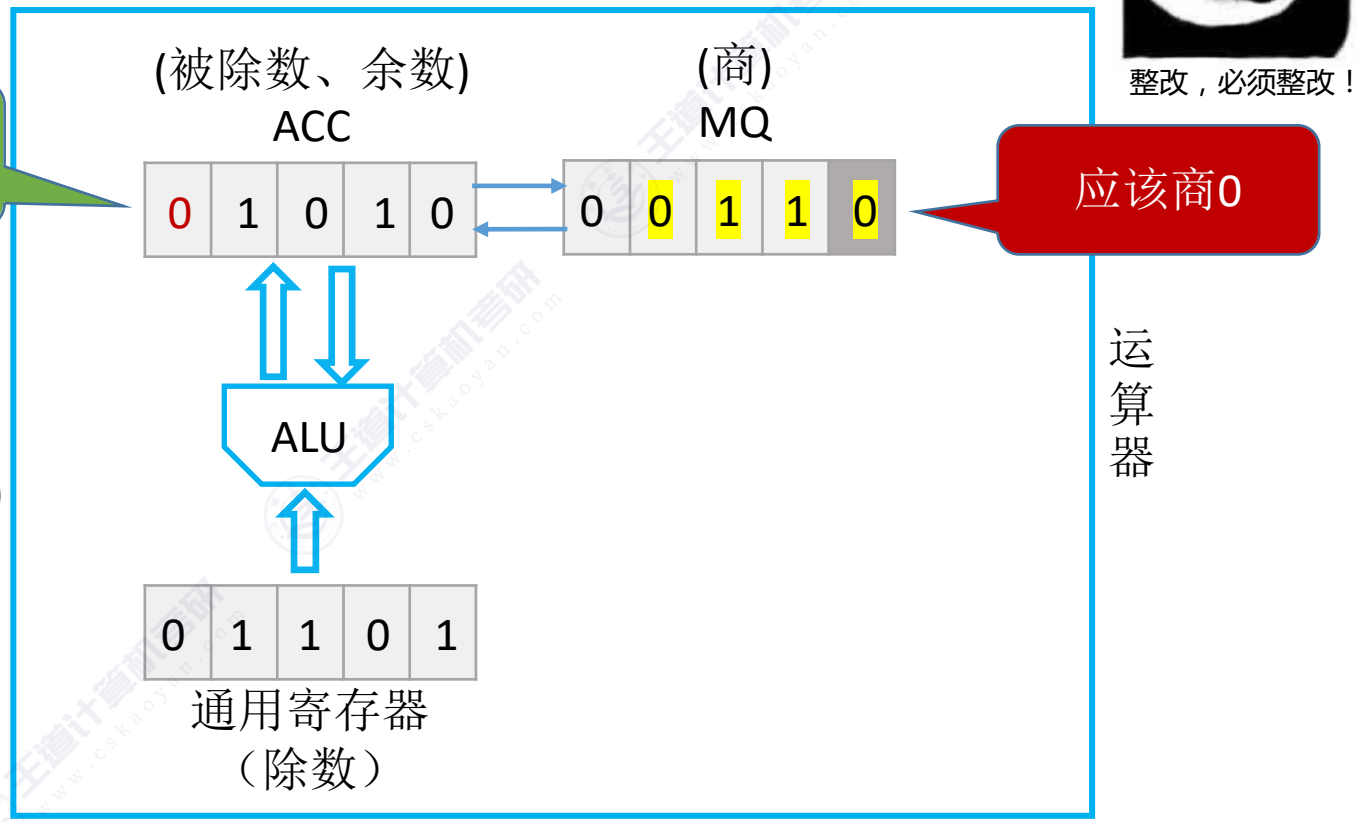
(ACC)+ $[-|y|]_{补}$→ACC

01010+10011 = 11101

(ACC)+ $[|y|]_{补}$→ACC

11101+01101 = 01010

整改，必须整改！



计算机很傻，会先默认上商1，如果搞错了再改上商0。并“恢复余数”

原码除法：恢复余数法

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

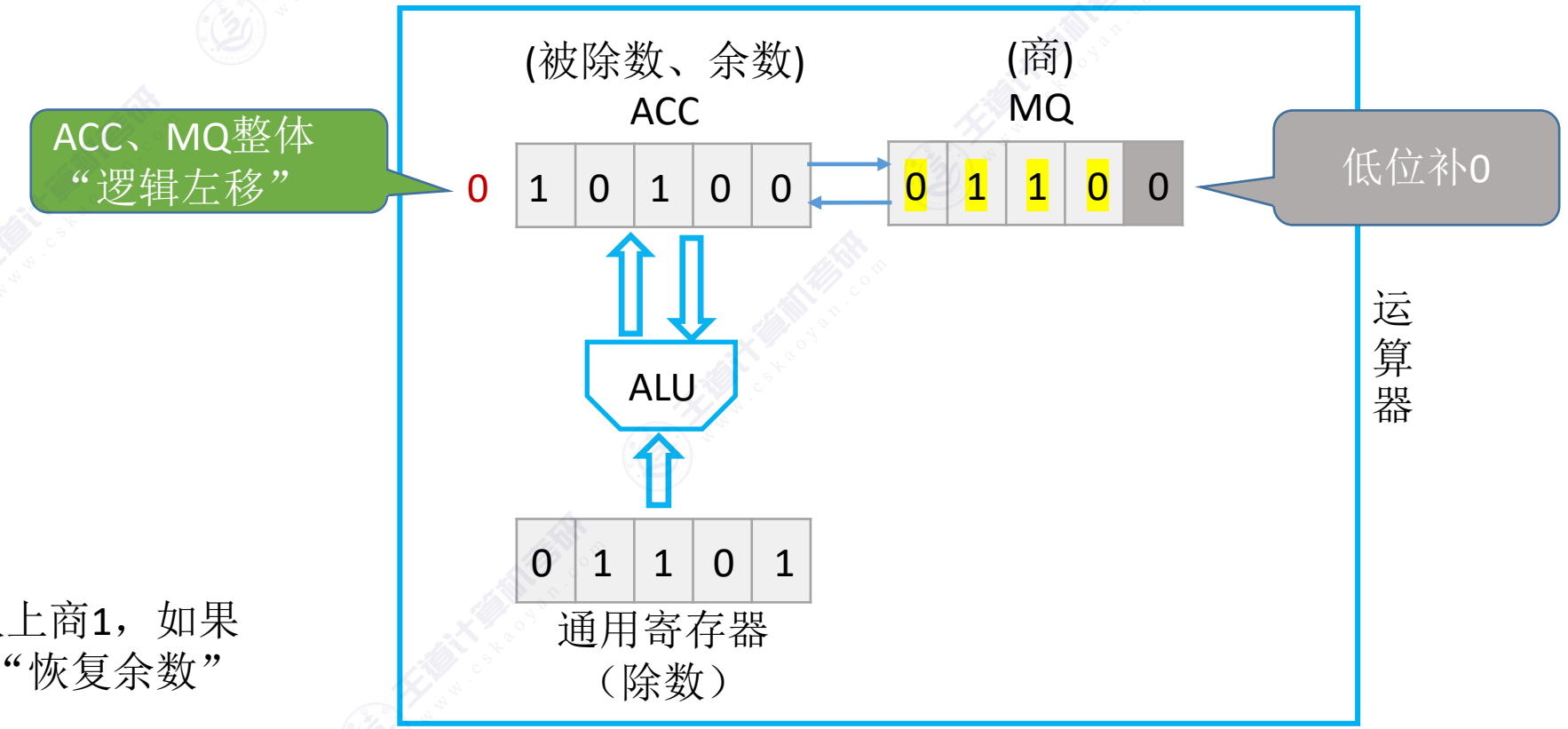
$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

符号单独处理：符号位 = $x_s \oplus y_s$

数值位取绝对值进行除法计算

实现方法：上商0/1，得到余数，余数末尾补0

```
          0.1101
01101 ⌠ 01011
        00000
       ------
        10110
        01101
       ------
        10010
        01101
       ------
        01010
        00000
       ------
        10100
        01101
       ------
         0111
```



计算机很傻，会先默认上商1，如果搞错了再改上商0。并“恢复余数”

原码除法：恢复余数法

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

符号单独处理：符号位 = $x_s \oplus y_s$

数值位取绝对值进行除法计算

实现方法：上商0/1，得到余数，余数末尾补0

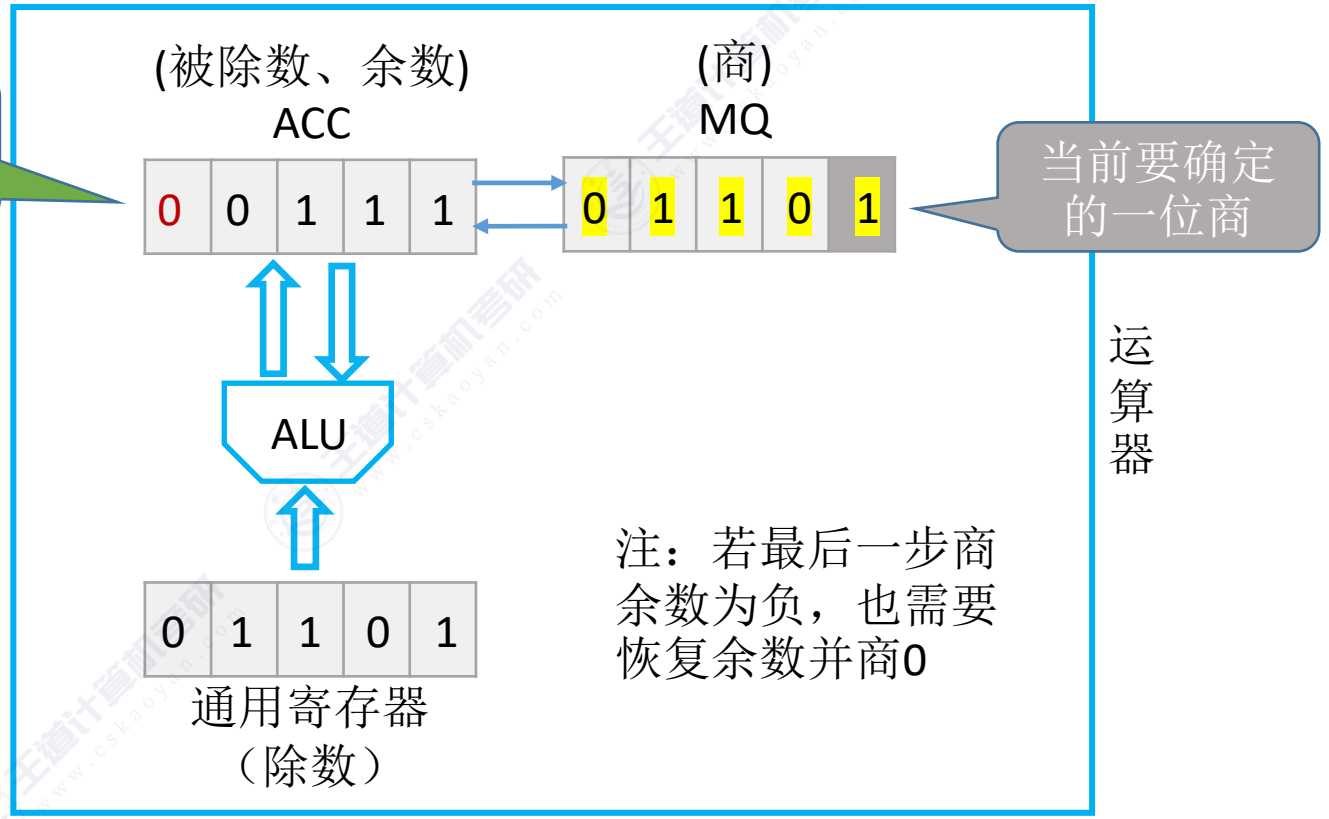
```
          0.1101
01101 ⌠ 01011
        00000
       ------
        10110
        01101
       ------
        10010
        01101
       ------
        01010
        00000
       ------
        10100
        01101
       ------
         0111
```

求余数：
(ACC)-(除数)→ACC

(ACC)+ $[-|y|]_{补}$→ACC
10010+10011 = 00111

相减结果是个正数，应上商 1

计算机很傻，会先默认上商1，如果搞错了再改上商0。并“恢复余数”



原码除法：恢复余数法

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

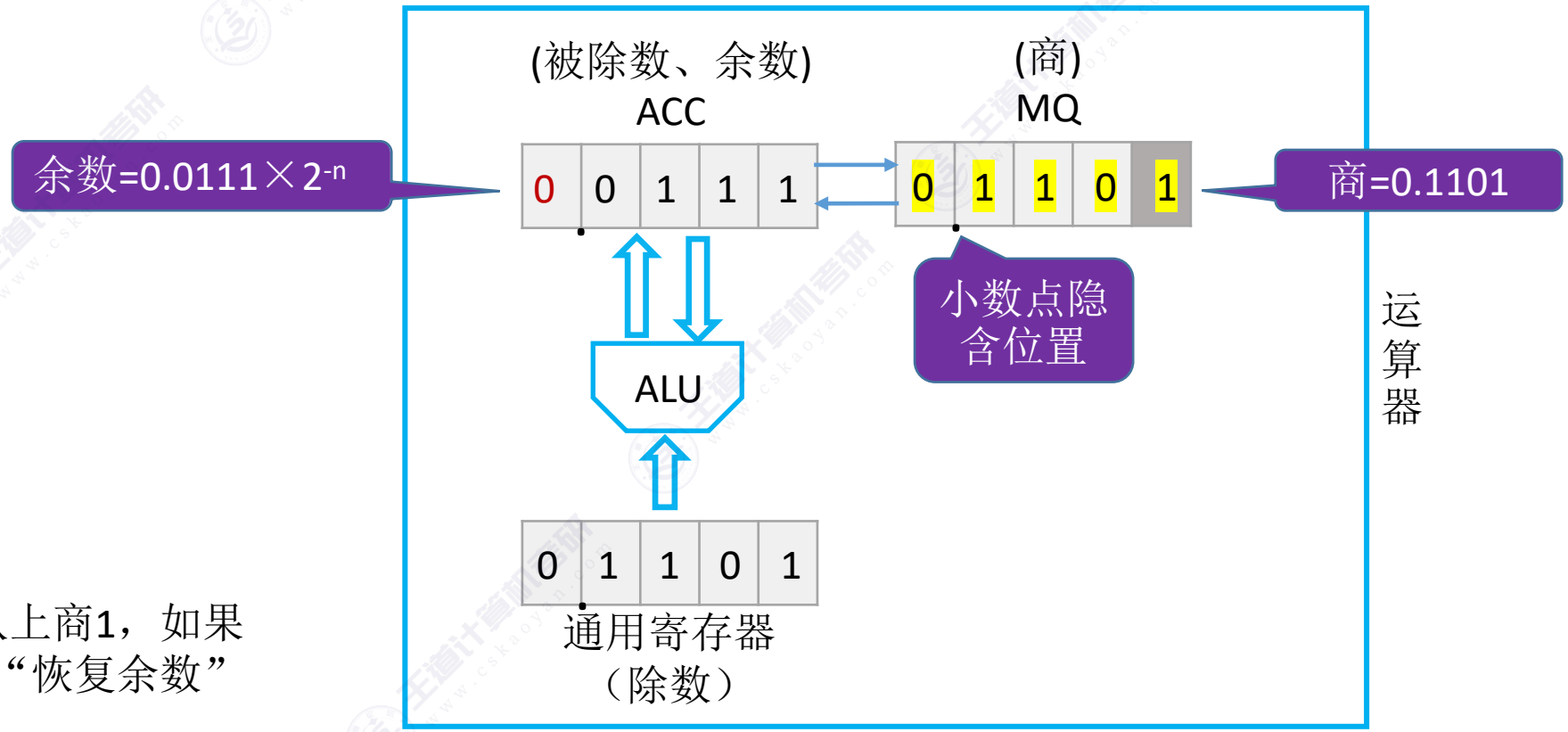
$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

符号单独处理：符号位 = $x_s \oplus y_s$

数值位取绝对值进行除法计算

实现方法：上商0/1，得到余数，余数末尾补0

```
          0.1101
01101 ⎞ 01011
        00000
       ------
        10110
        01101
       ------
        10010
        01101
       ------
        01010
        00000
       ------
        10100
        01101
       ------
         0111
```



计算机很傻，会先默认上商1，如果搞错了再改上商0。并“恢复余数”

原码除法：恢复余数法（手算）

符号位	绝对值

符号位与数值位分开处理

设机器字长为5位（含1位符号位，$n=4$），$x=0.1011$，$y=0.1101$，采用原码恢复余数法求x/y

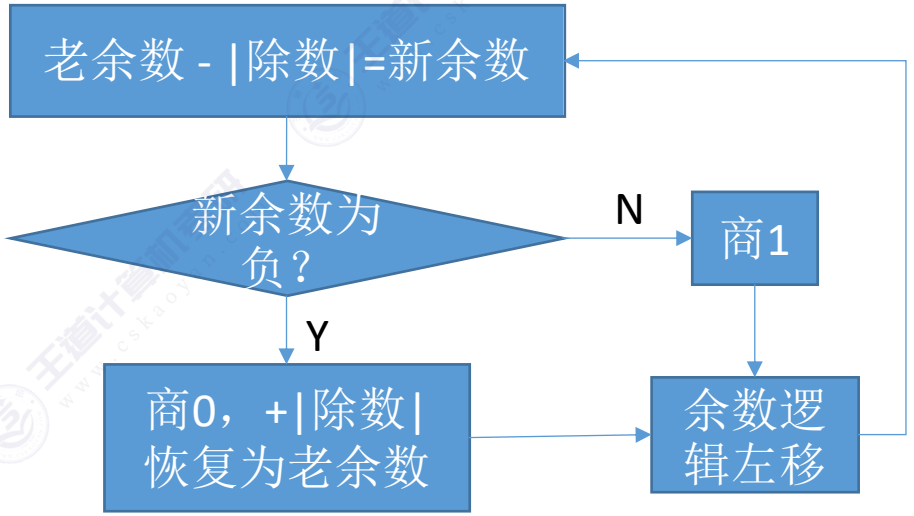
$|x|=0.1011$，$|y|=0.1101$，$[|y|]_{补}=0.1101$，$[-|y|]_{补}=1.0011$

被除数/余数		商
	0.1011	
$+[-\|y\|]_{补}$	1.0011	
	1.1110	0
$+[\|y\|]_{补}$	0.1101	
	0.1011	
左移	1.0110	
$+[-\|y\|]_{补}$	1.0011	
	0.1001	01
左移	1.0010	
$+[-\|y\|]_{补}$	1.0011	
	0.0101	011
…		…

余数为负，就要商0，并恢复余数

逻辑左移

余数为正，就要商1，不用恢复余数



左移n次，上商n+1次

最后一次上商余数不左移

原码除法：恢复余数法（手算）

能否不恢复余数？

符号位	绝对值

符号位与数值位分开处理

设机器字长为5位（含1位符号位，n=4），x=0.1011，y=0.1101，采用原码恢复余数法求x/y

$|x|$=0.1011，$|y|$=0.1101，$[|y|]_{补}$=0.1101，$[-|y|]_{补}$=1.0011

	被除数/余数		商
	0.1011		
$+[-\|y\|]_{补}$	1.0011		
	1.1110	a	0
$+[\|y\|]_{补}$	0.1101	b	
	0.1011	a+b	
左移	1.0110	(a+b)×2 = 2a + 2b	
$+[-\|y\|]_{补}$	1.0011	(a+b)×2 - b = 2a + 2b – b = 2a + b	
	0.1001		01
左移	1.0010		
$+[-\|y\|]_{补}$	1.0011		
	0.0101		011
	…		…

余数a为负

若余数为负，则可直接商0，并让余数左移1位再加上|除数|

原码除法：加减交替法

又名：不恢复余数法

符号位与数值位分开处理

设机器字长为5位（含1位符号位，$n=4$），$x=0.1011$，$y=0.1101$，采用原码加减交替除法求x/y

$|x|=0.1011$，$|y|=0.1101$，$[|y|]_{补}=0.1101$，$[-|y|]_{补}=1.0011$

$Q_s=x_s\oplus y_s=0\oplus 0=0$

得$x/y=+0.1101$

余0.0111×2^{-4}

被除数/余数		商	ACC	MQ
	0.1011		01011	00000
$+[-\|y\|]_{补}$	1.0011			
	1.1110	0	11110	00000
左移	1.1100		11100	00000
$+[\|y\|]_{补}$	0.1101			
	0.1001	01	01001	00001
左移	1.0010		10010	00010
$+[-\|y\|]_{补}$	1.0011			
	0.0101	011	00101	00011
左移	0.1010		01010	00110
$+[-\|y\|]_{补}$	1.0011			
	1.1101	0110	11101	00110
左移	1.1010		11010	01100
$+[\|y\|]_{补}$	0.1101			
	0.0111	01101	00111	01101

若余数为负，则可直接商0，让余数左移1位再加上|除数|，得到下一个新余数

若余数为正，则商1，让余数左移1位再减去|除数|，得到下一个新余数

若余数为负，需商0，并$+[|y|]_{补}$得到正确余数

注：余数的正负性与商相同

恢复余数法：当余数为负时商0，并+|除数|，再左移，再-|除数|

加减交替法：当余数为负时商0，并左移，再+|除数|

原码除法：加减交替法

又名：不恢复余数法

符号位与数值位分开处理

设机器字长为5位（含1位符号位，$n=4$），$x=0.1011$，$y=0.1101$，采用原码加减交替除法求x/y

$|x|=0.1011$，$|y|=0.1101$，$[|y|]_{补}=0.1101$，$[-|y|]_{补}=1.0011$

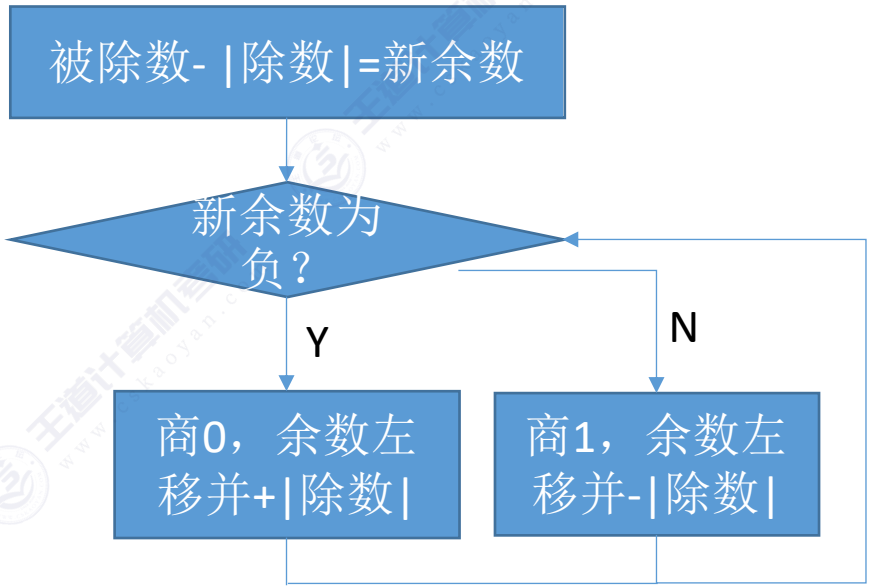
$Q_s=x_s\oplus y_s=0\oplus 0=0$

得$x/y=+0.1101$

余0.0111×2^{-4}

被除数/余数		商
	0.1011	
$+[-\|y\|]_{补}$	1.0011	
	1.1110	0
左移	1.1100	
$+[\|y\|]_{补}$	0.1101	
	0.1001	01
左移	1.0010	
$+[-\|y\|]_{补}$	1.0011	
	0.0101	011
左移	0.1010	
$+[-\|y\|]_{补}$	1.0011	
	1.1101	0110
左移	1.1010	
$+[\|y\|]_{补}$	0.1101	
	0.0111	01101

若余数为负，需商0，并$+[|y|]_{补}$得到正确余数



加/减n+1次，每次加减确定一位商；

左移n次（最后一次加减完不移位）

最终可能还要再多一次加