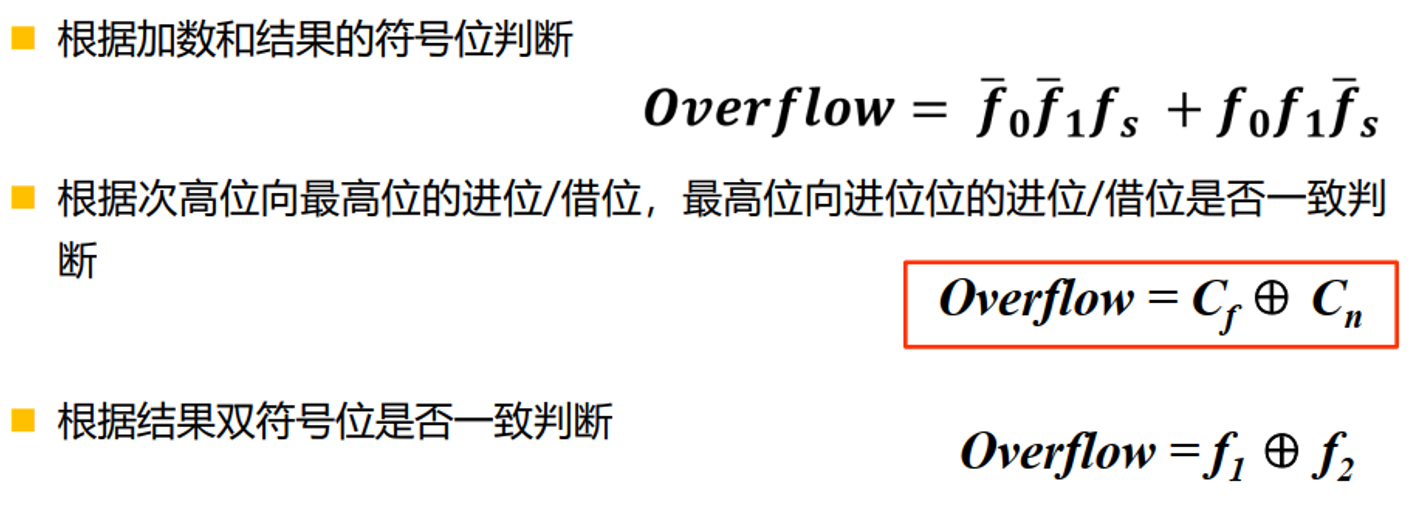
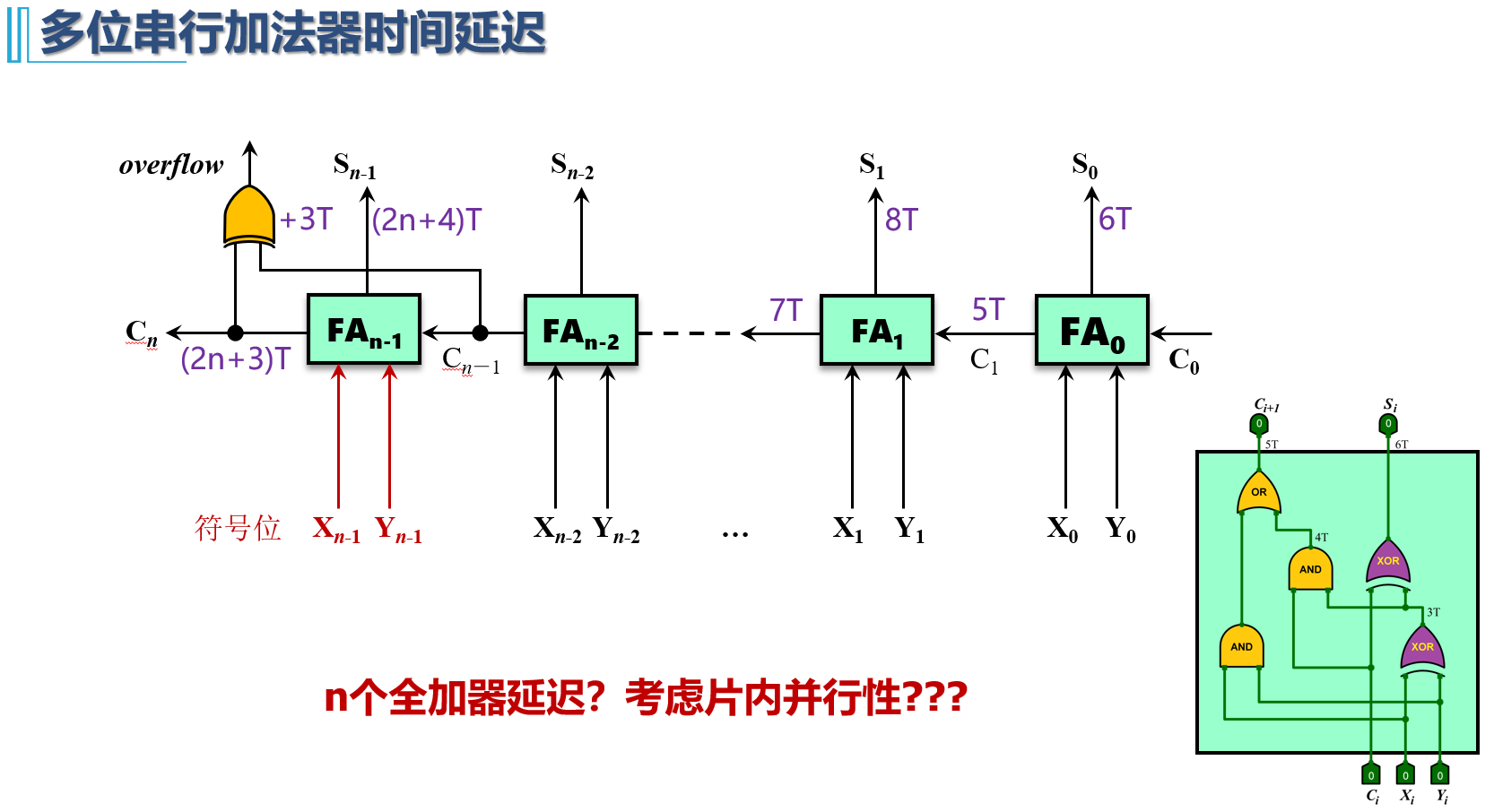
1. 回顾：[-Y]的补码

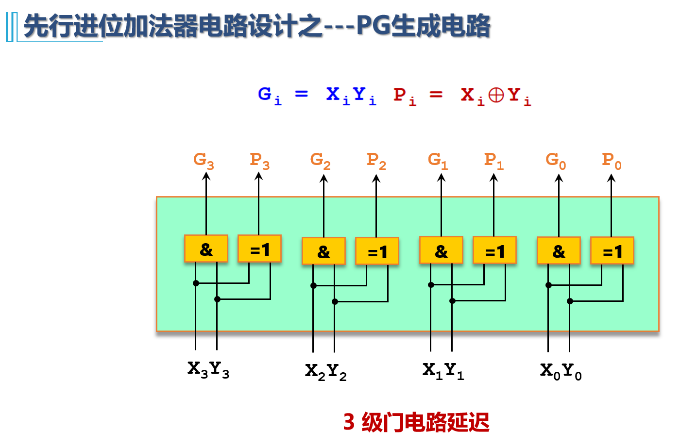
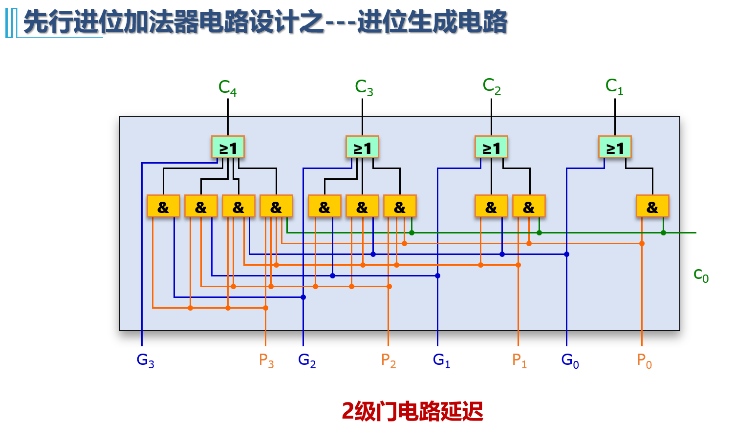
[-Y]的补码等于[Y]的补码连同符号位，逐位取反，末位加一。

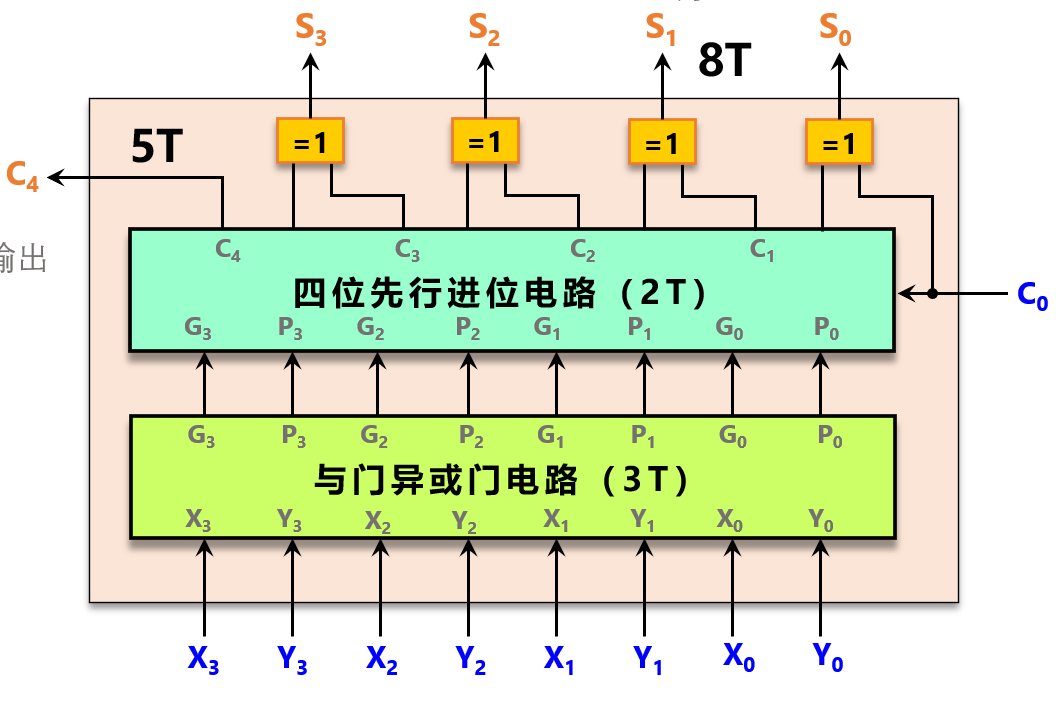
1. **[X−Y]补 = [X]补＋ [−Y]补 = [X]补−[Y]补**
2. 有符号数溢出判断方法：



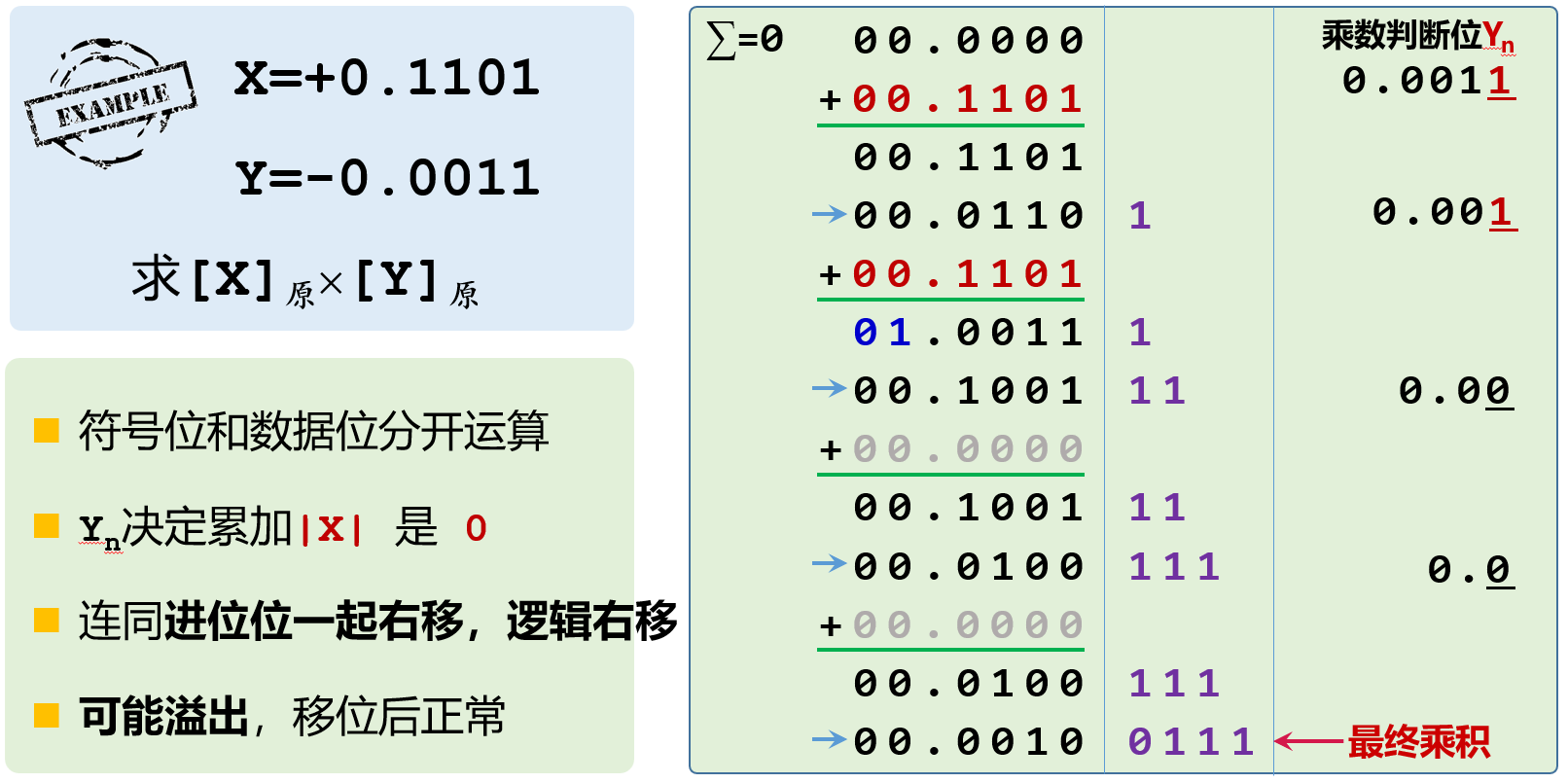
1. 异或门的延迟时间3T
2. 多位串行加法器时间延迟，一位全加器产生进位Ci：5T，产生结果Si：6T，串行横着加2T，竖着加3T



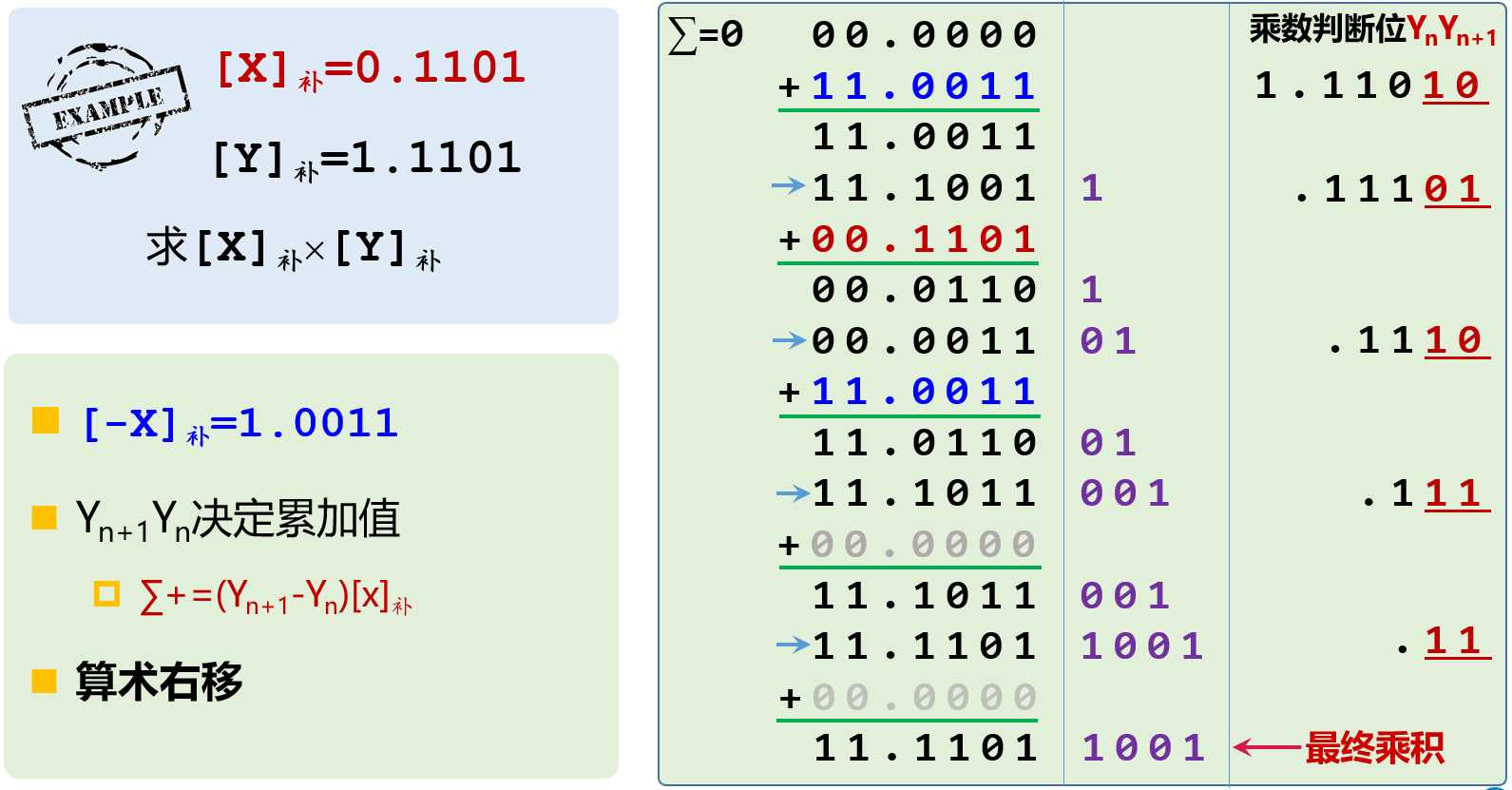
1. 先行进位加法器的延时



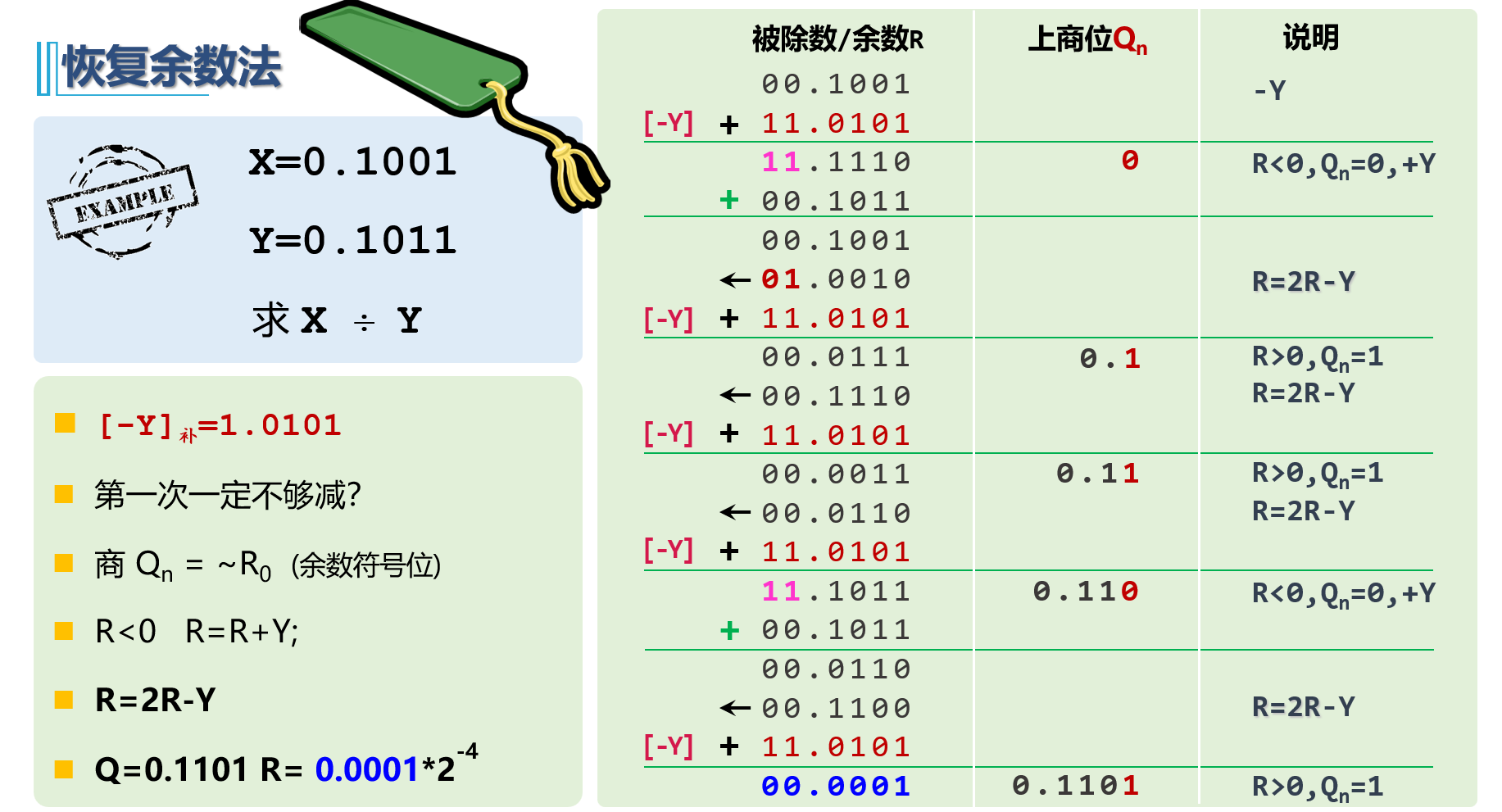
1. 原码一位乘法：绝对值相乘，符号位异或，逻辑右移，补0，n次加法，n次移位



1. 补码一位乘法：用两位符号位，先写出[-X]补，y最后添一位，用Yn+1-Yn，符号位参与运算，每次移位用算术右移，补符号位，最后一次加法不移位，故n次移位，n+1次加法。

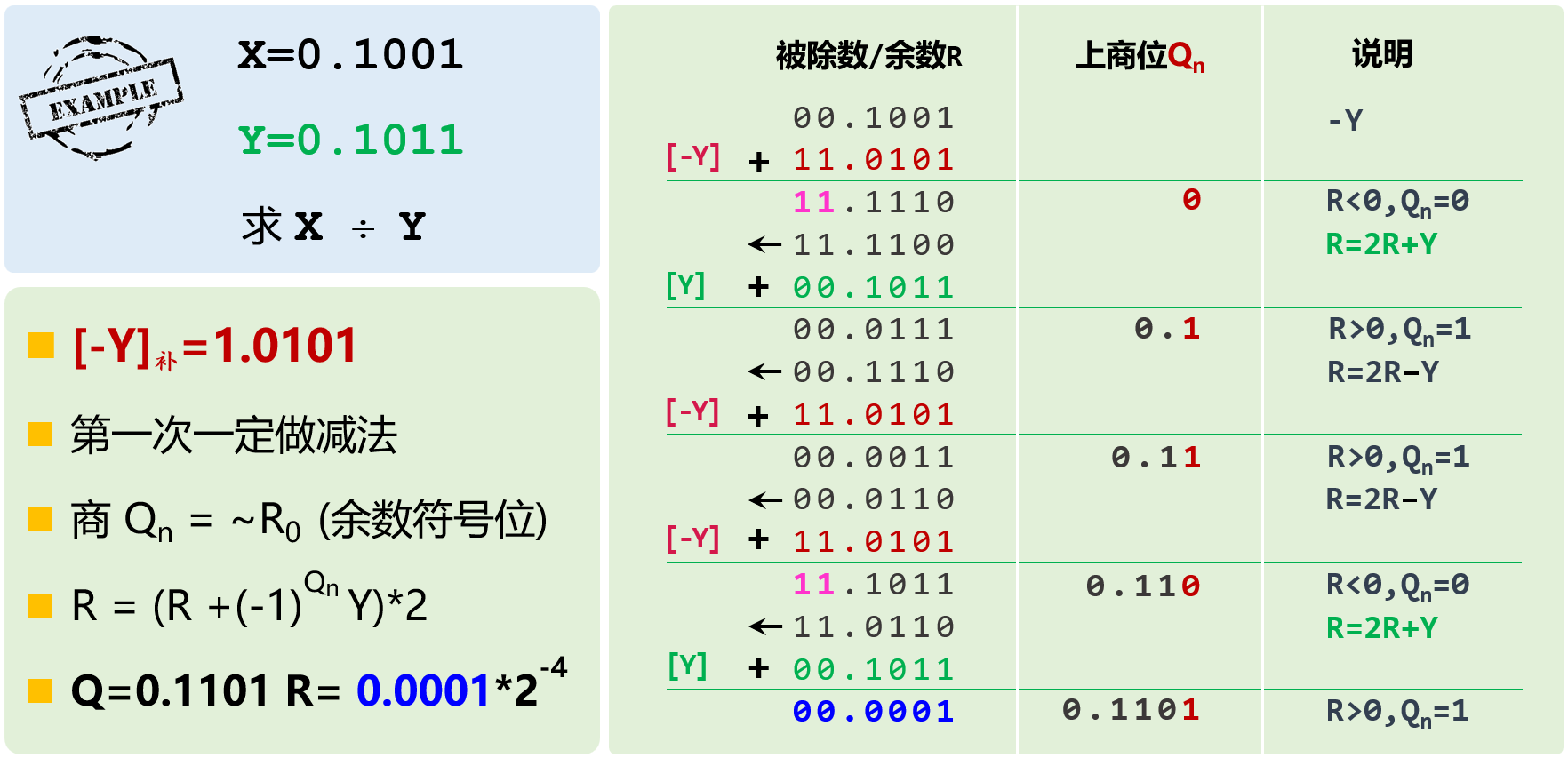


1. 除法都是绝对值除法，商的符号位由被除数和除数的符号相异或，余数跟被除数符号一样
2. 恢复余数法：真正的余数要乘以2的（-n）次方，要算-y的补码



1. 不恢复余数法（加减交替法）要算-y的补码

若最后余数小于0，则要加上y回到正数，n+1次加法，n次移位



1. 浮点数的加法，对阶的时候，小阶对大阶（小指数往大指数靠拢）。规格化的时候，目标形式：00.1XX，11.0XX
2. IEEE754下，规格化数为1.XXX，尾数的计算为原码的加减法
3. 对于浮点数，阶码溢出才是真的溢出，（即符号位01和10）

溢出的判断：

Ⅰ. 对阶操作不会引起阶码上溢或下溢

Ⅱ. 右规和尾数舍入都可能引起阶码上溢

Ⅲ. 左规时可能引起阶码下溢

Ⅳ. 尾数溢出时结果不一定溢出

1. 规格化数：尾数表示的真值的绝对值>=0.5，<1。补码表示的-0.5不是规格化
2. 规格化数： 原码：0.1XXX，1.1XXX，

补码：0.1XXX，1.0XXX