

## 本节主题

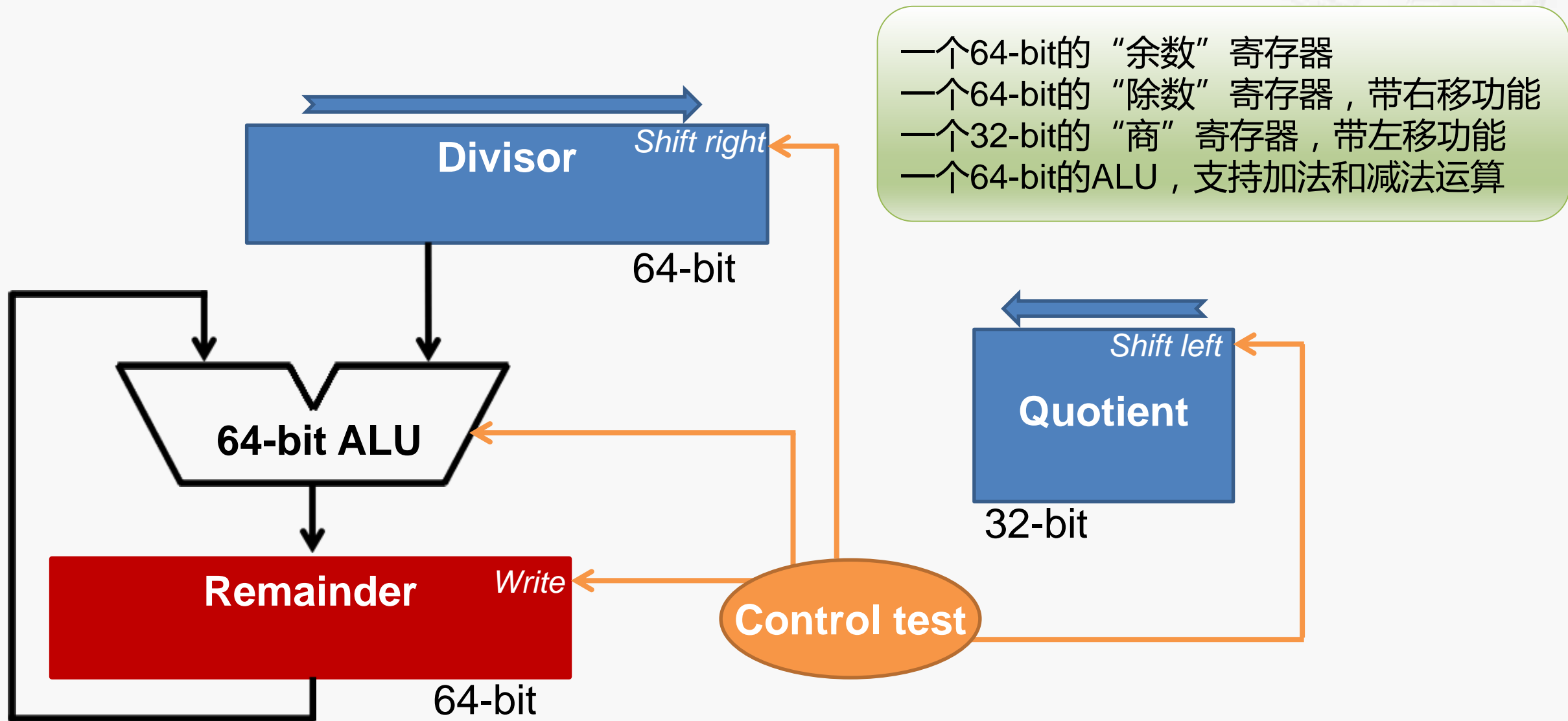


# 除法器的优化

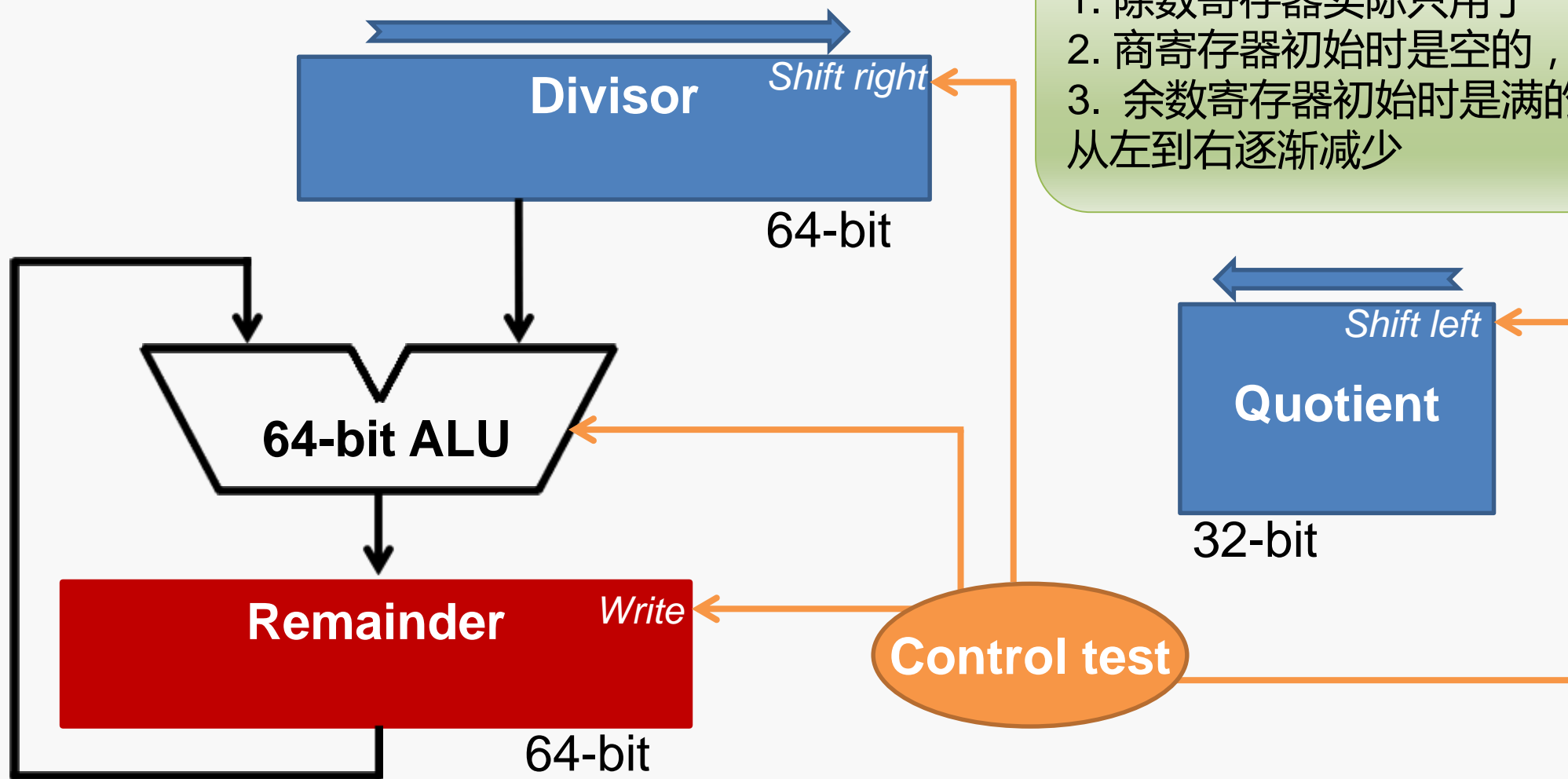
北京大学·慕课  
计算机组成  
制作人：陆俊林



# 除法器的实现（第一版）



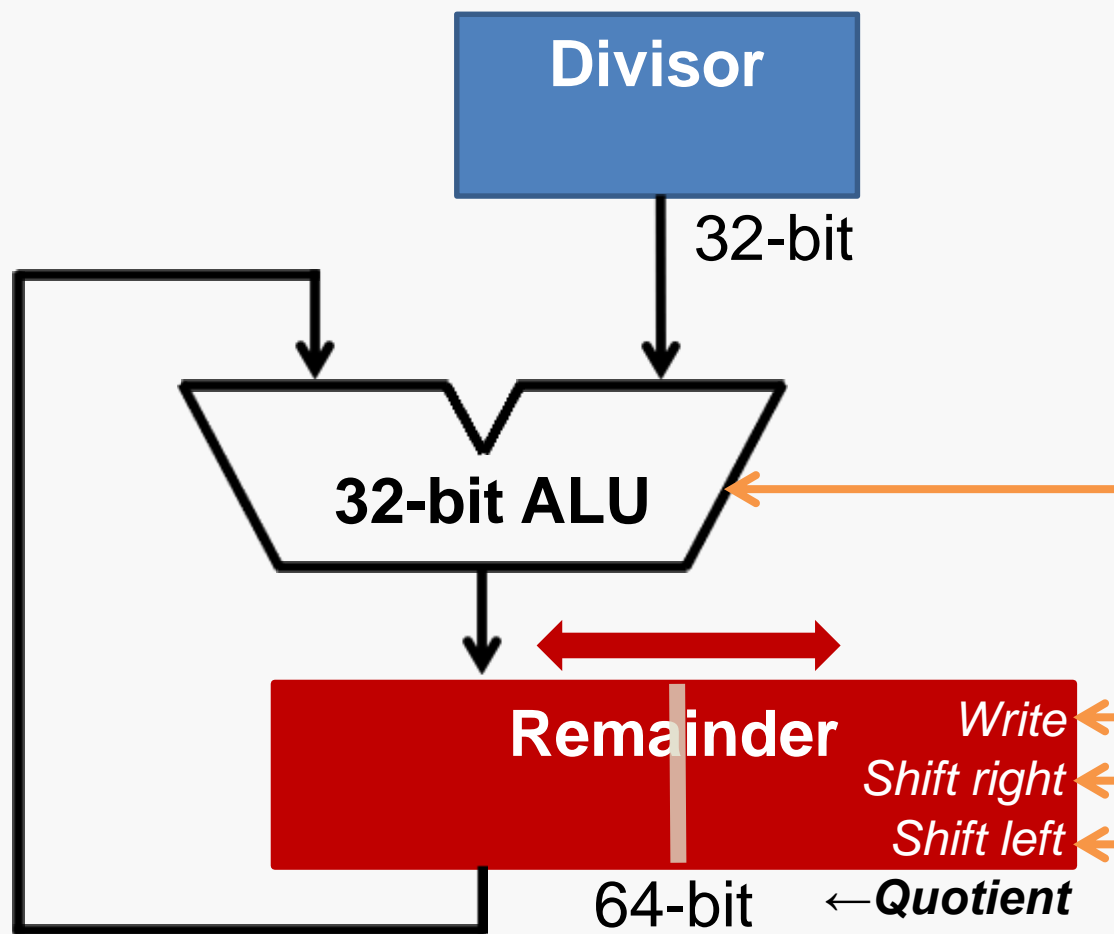
# 除法器的面积优化



从减小面积的角度进行分析：

1. 除数寄存器实际只用了一半
2. 商寄存器初始时是空的，从右到左逐位填满
3. 余数寄存器初始时是满的，有实际意义的位从左到右逐渐减少

# 除法器的实现（第二版）



## 优化方案

1. 除数寄存器缩小为32-bit，无需支持移位
2. 取消商寄存器
3. 64-bit ALU缩小为32-bit
4. 余数寄存器只有高32-bit参与加减法运算
5. 余数寄存器需支持左移和右移
6. 商从右端逐位移入余数寄存器
7. 运算结束时，商占据余数寄存器的低32-bit

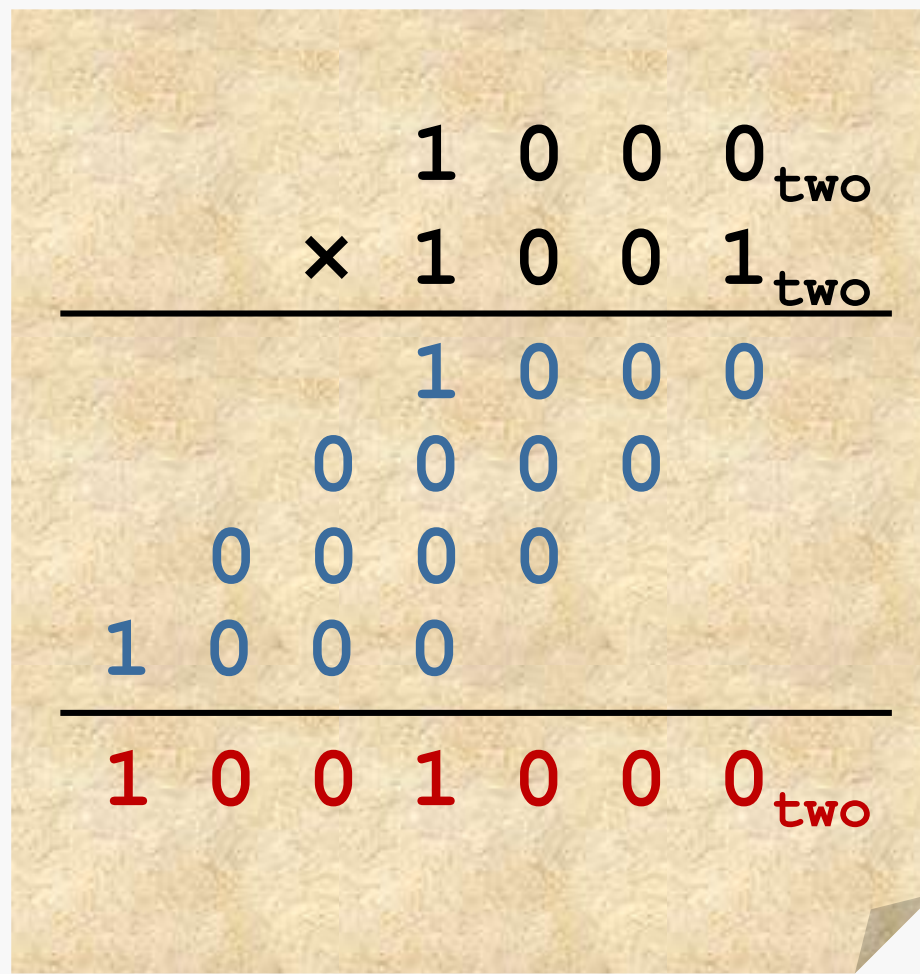
## 原先的结构：

- 一个64-bit的“余数”寄存器
- 一个64-bit的“除数”寄存器，带右移功能
- 一个32-bit的“商”寄存器，带左移功能
- 一个64-bit的ALU，支持加法和减法运算

# 除法的性能优化分析

## 回顾：乘法的特点

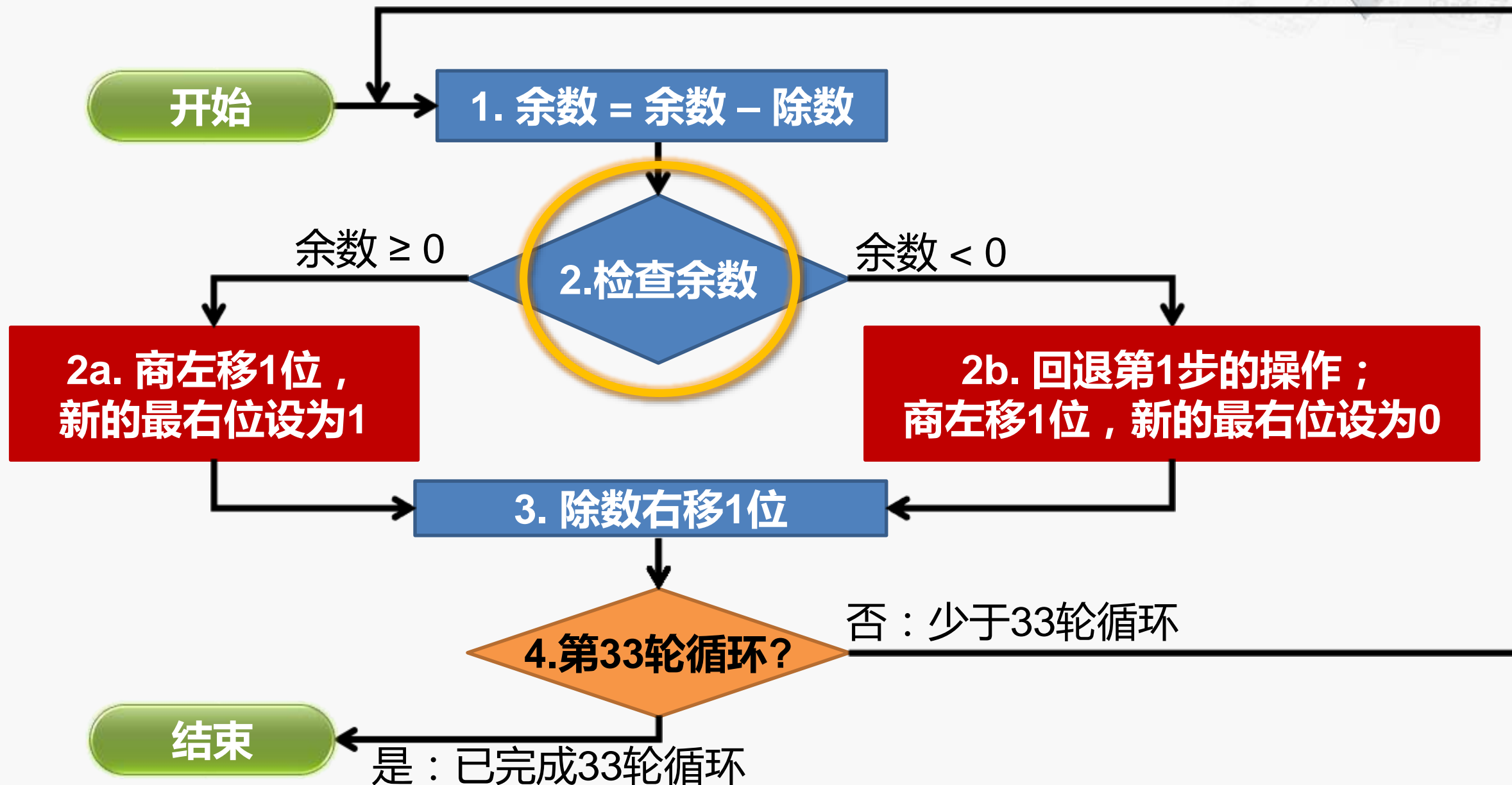
- 每个部分积都是独立的
- 可以并行计算各个部分积



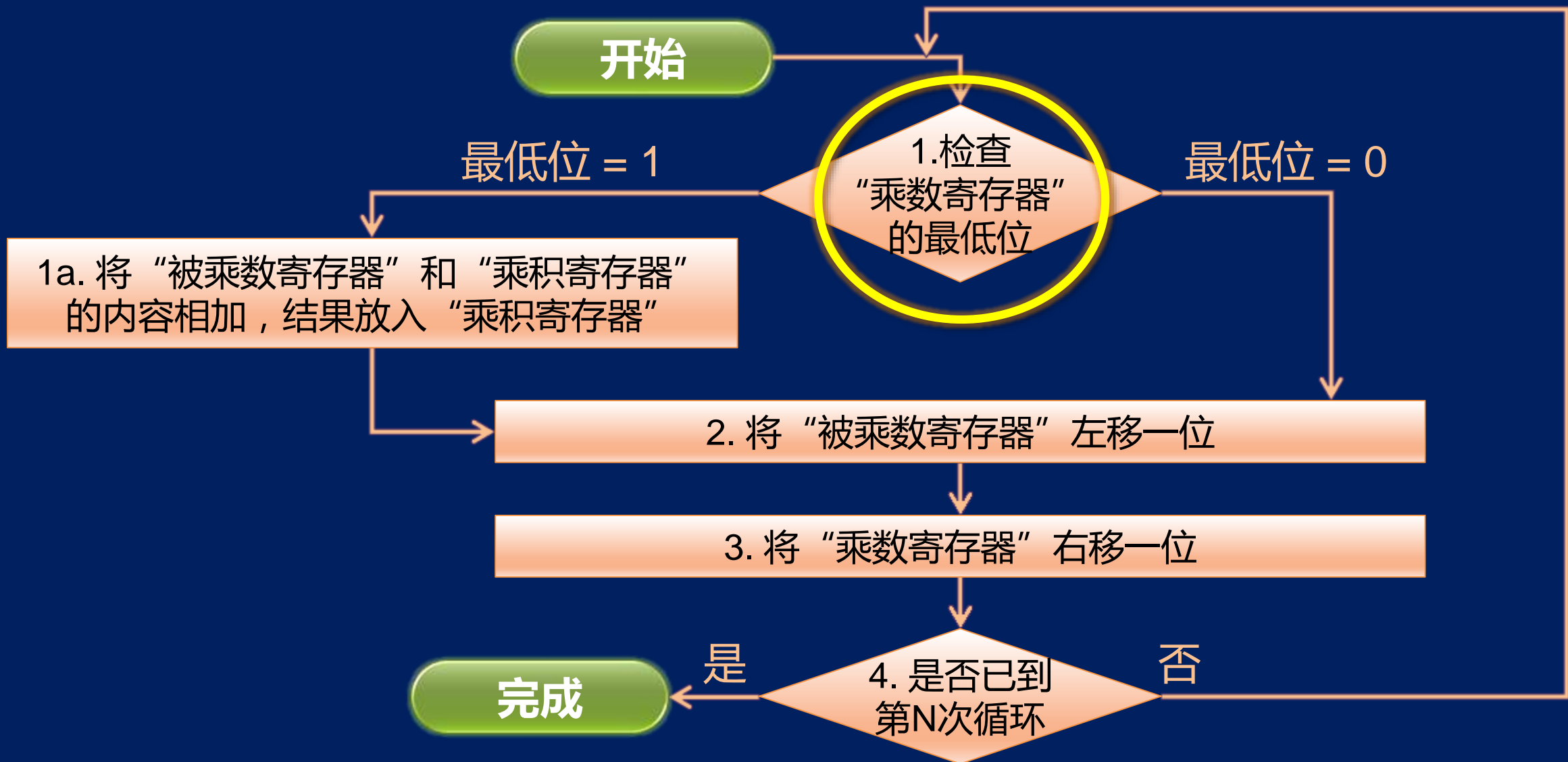
A hand-drawn illustration of a multiplication problem on a piece of paper. The numbers are written in a grid-like format. The first number is 1001000<sub>two</sub> and the second is 1001<sub>two</sub>. The partial products are shown in blue, and the final result is shown in red.

$$\begin{array}{r} \phantom{100}1001000_{\text{two}} \\ \times \phantom{100}1001_{\text{two}} \\ \hline \phantom{100}1001000 \\ \phantom{100}0000000 \\ \phantom{100}0000000 \\ \phantom{100}0000000 \\ 1001000 \\ \hline 10011001000_{\text{two}} \end{array}$$

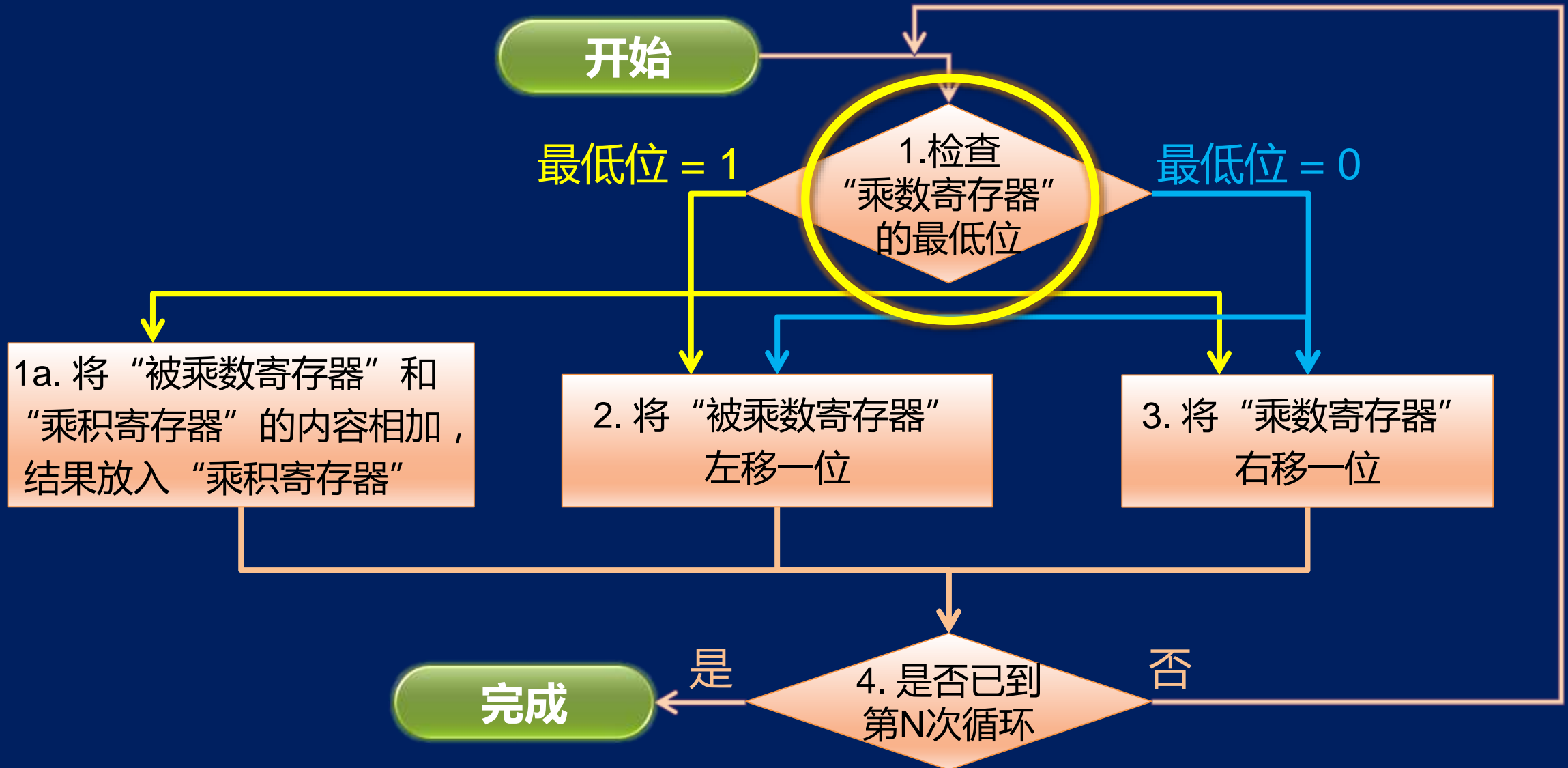
# 除法的性能优化分析



# 对比：N位乘法器的工作流程图

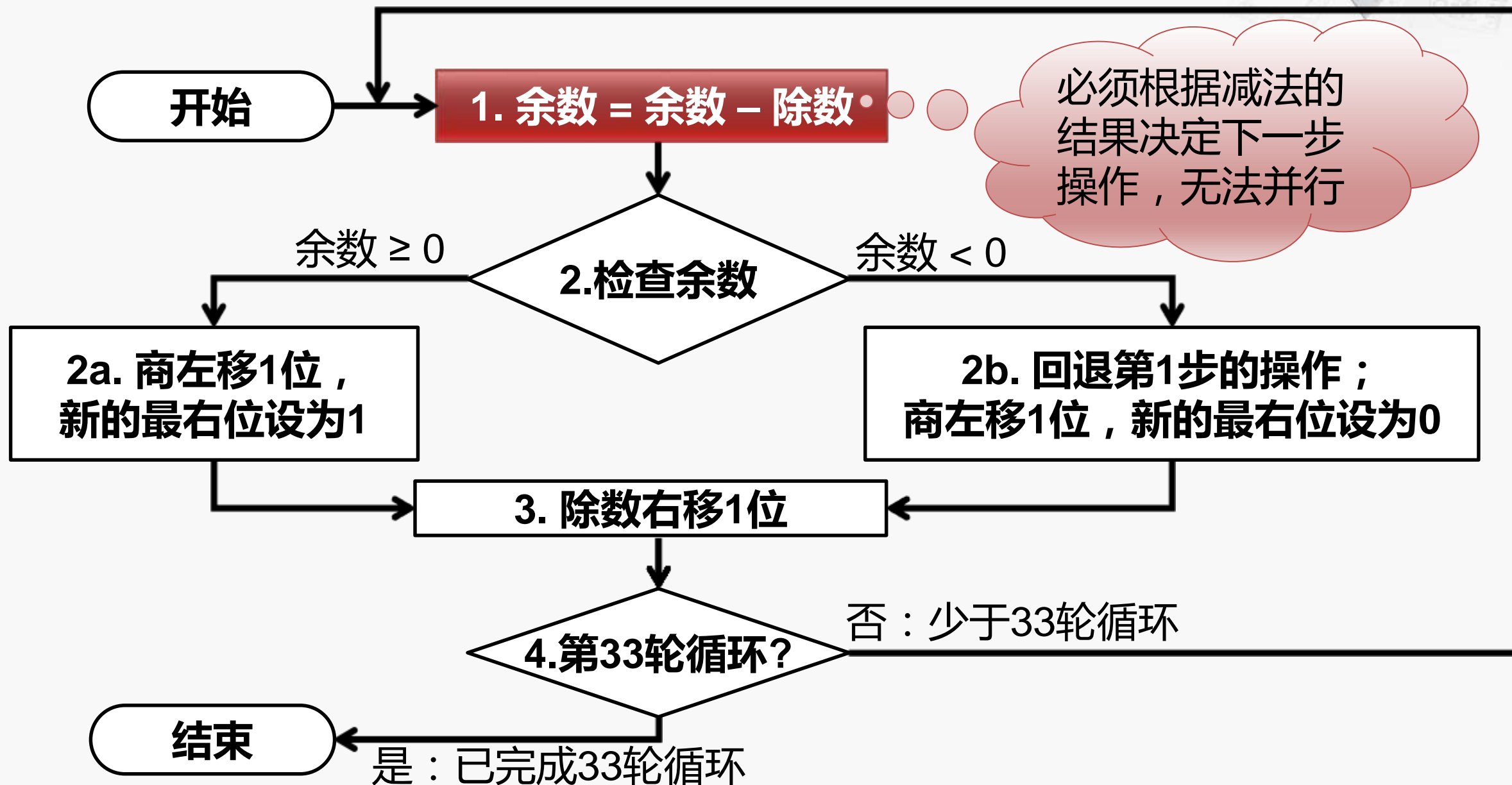


# 对比：N位乘法器的工作流程（优化后）





# 除法的性能优化分析



## 本节小结



# 除法器的优化

北京大学·慕课  
计算机组成  
制作人：陆俊林

