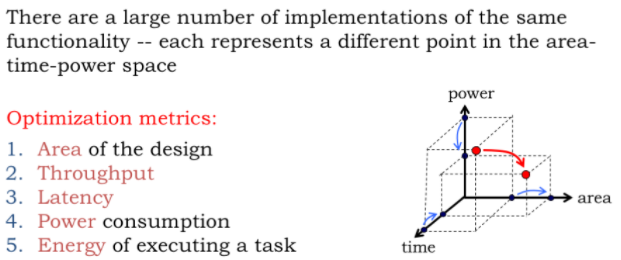
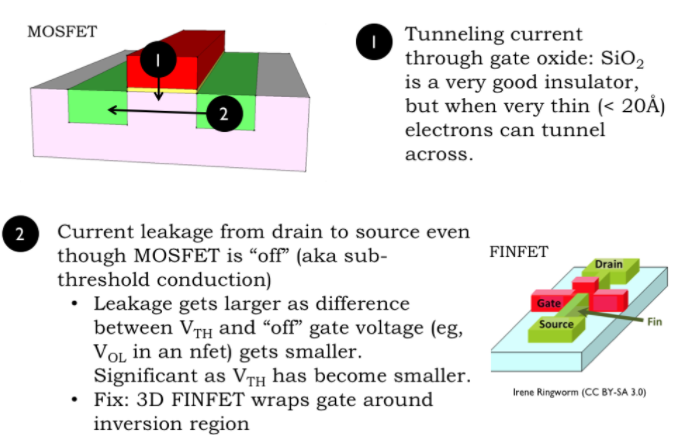
# 课件

## 优化你的设计



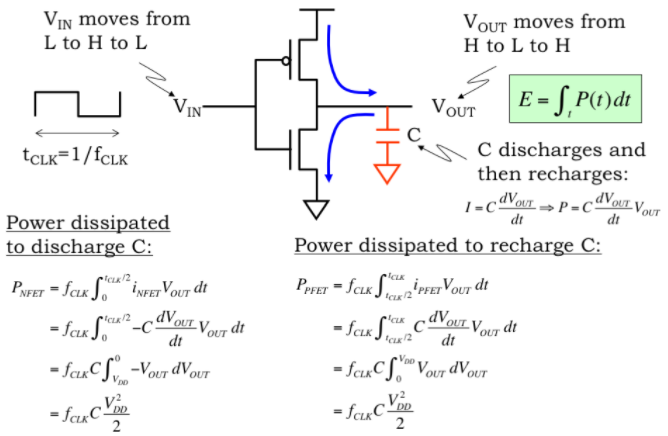
同一功能的不同实现，在空间、时间、能量维度，各有一个值，根据系统目的，选择合适方案

## CMOS静态能源消耗



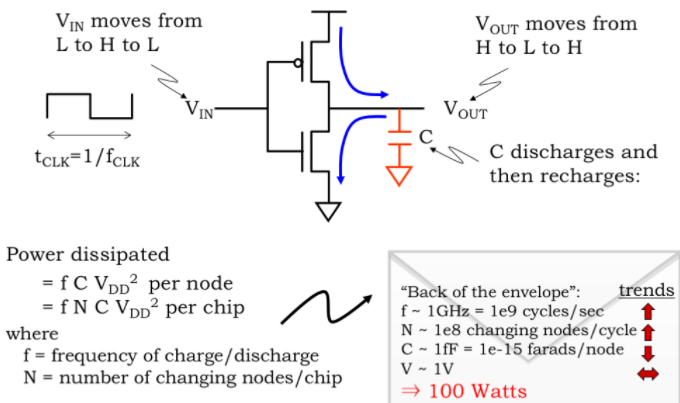
门极与基质之间、源极与漏极之间在off时，也会有微弱电流，造成能源消耗

## CMOS动态能源消耗



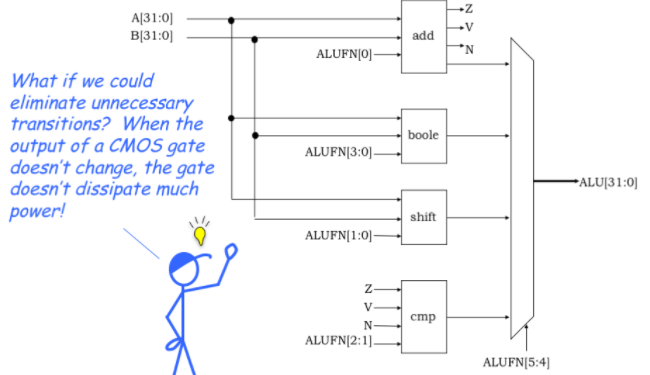
以反相器为例，输出电压0-1、1-0，电容充放电，都会消耗能量

## CMOS动态能源消耗2



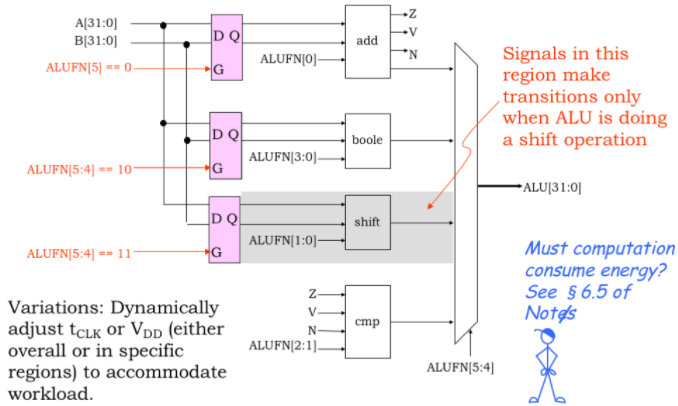
大概估算现代CPU功率

## 我们如何节省能源



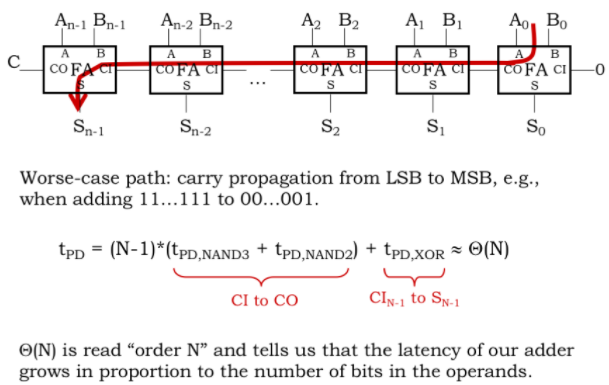
ALU运算单元，会执行add、boolean、shift、cmp运算，但只会需要一个结果，另外3个白算了，这是个可以节省能源的点。

## 更少转变->更少能量



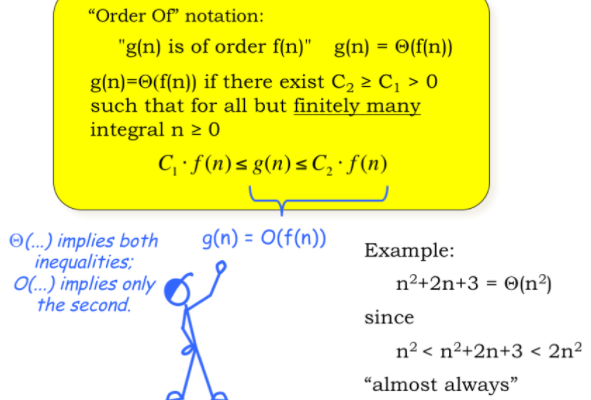
每个module前放锁存器，仅在需要该模块参与计算时，才打开，从而在不需要该模块计算时，让其电压不发生转变，减少能源消耗

## 提升速度：加法器例子

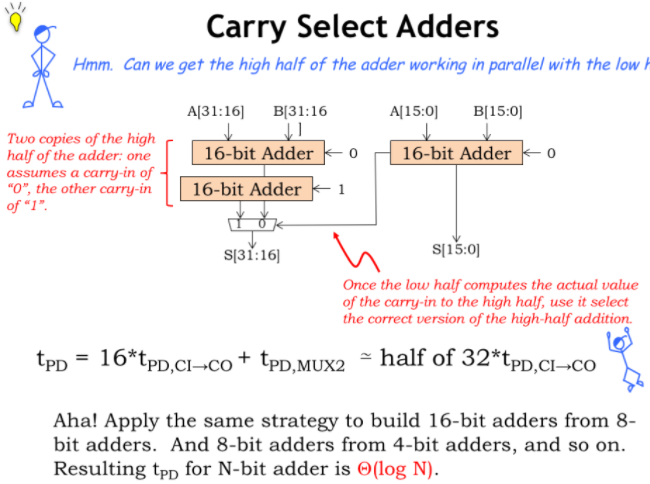


当N足够大时，位数提升2倍，延时提升2倍

## 性能/消耗分析

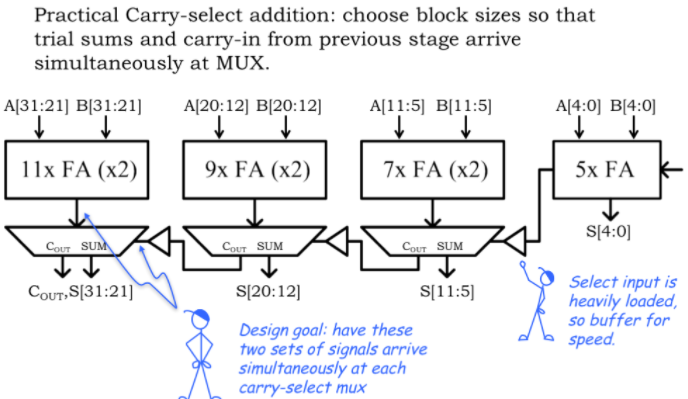


## 进位选择加法器



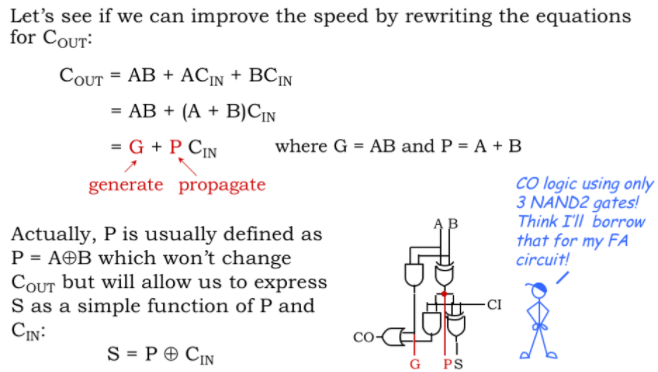
32位，需要5次选择器

## 32位仅为选择加法器

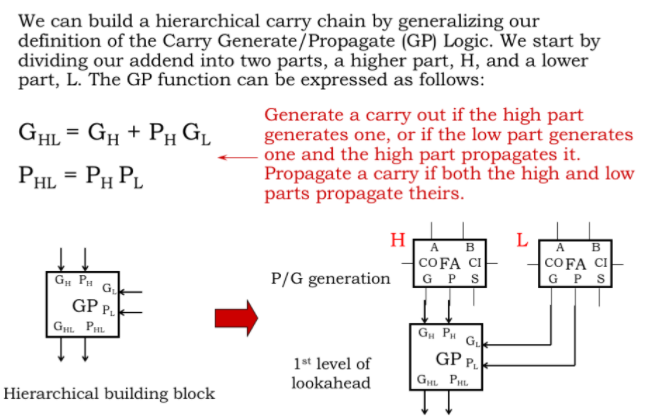


计算11位花的时间=计算9位花的时间+加法器时间，加法器为计算2位所花时间，故计算11位所花时间+计算2位所花时间，大约为计算13位所花时间，32/13约为2.5倍

## 想要：更快地进位逻辑



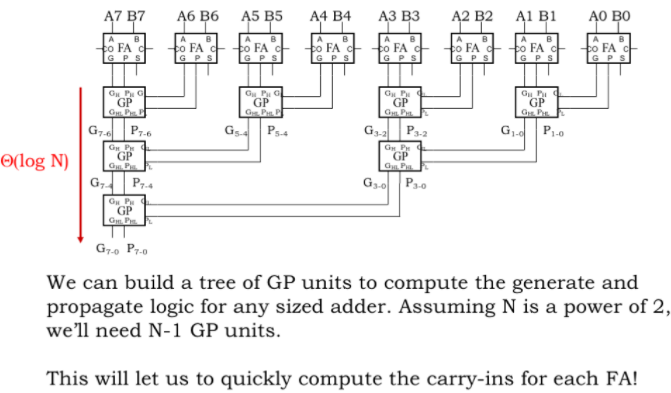
## 看进位的加法器（carry look-ahead）



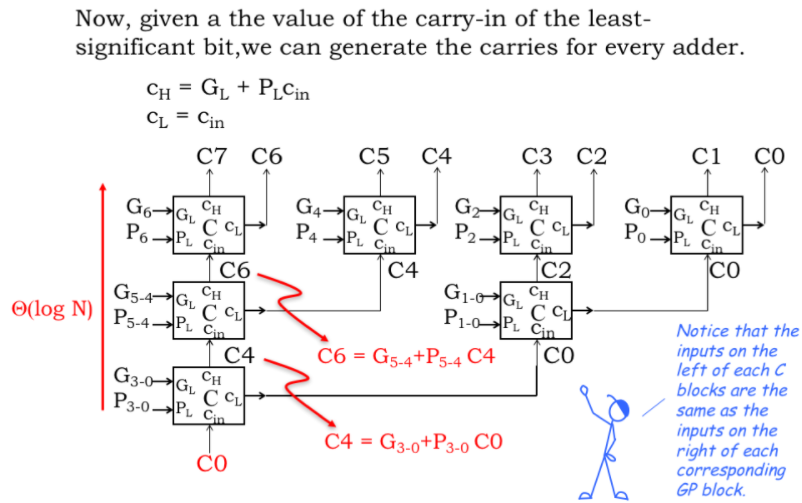
G：1表明发生进位

P：1表明当前一个模块有进位时会发生进位，0表示无论前一个模块是否进位，都不会产生进位

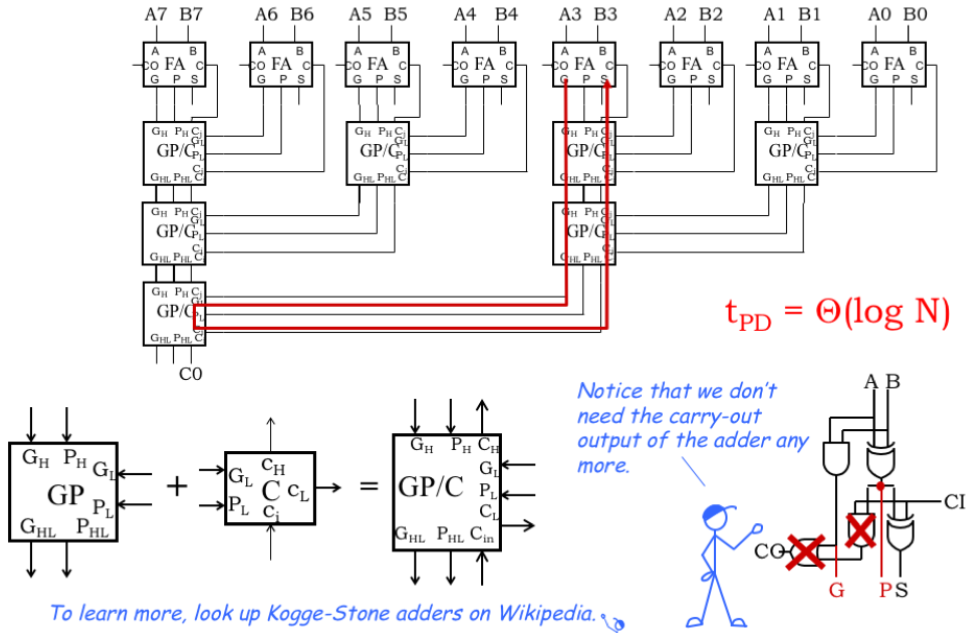
## 8位CLA（生成G&P）



## 8位CLA（生成carry-in）

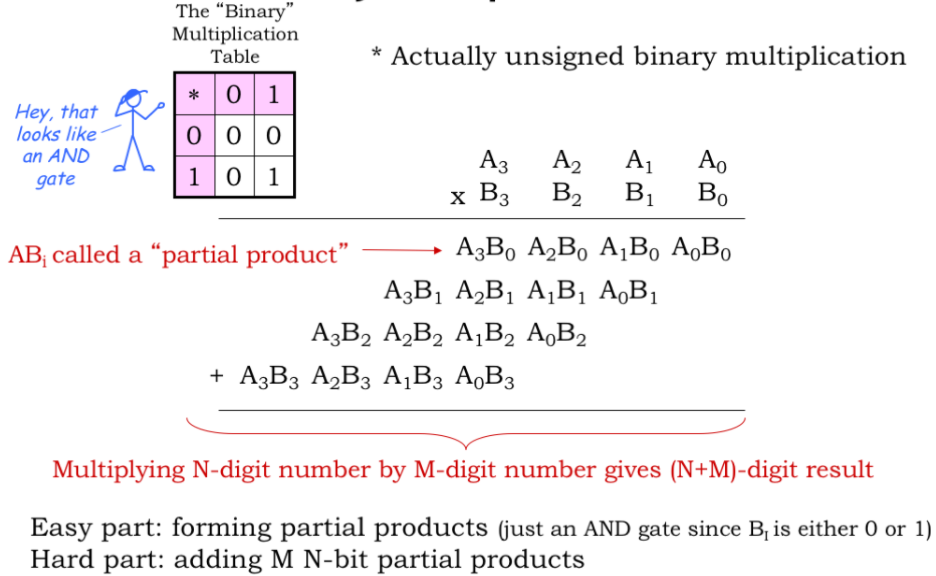


## 8位CLA

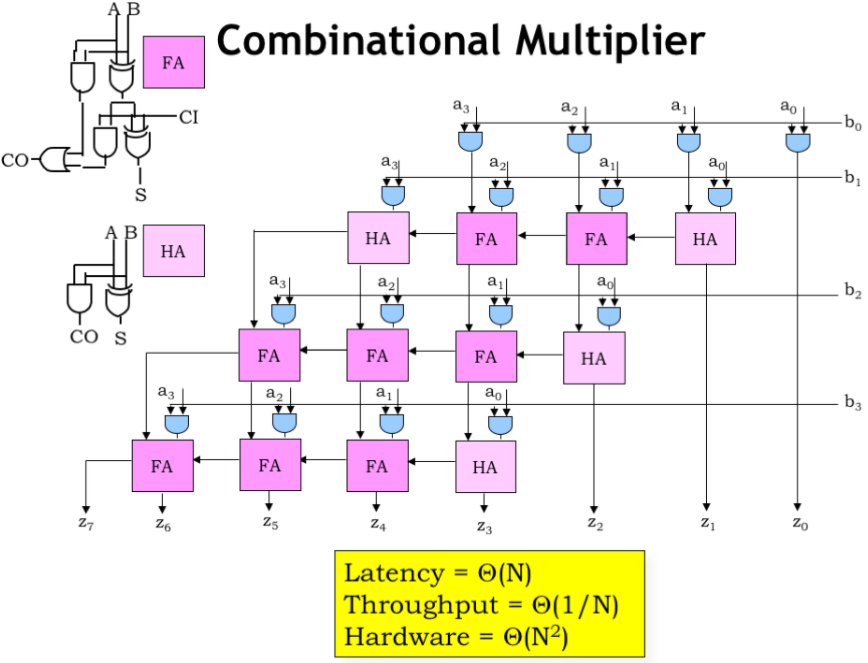


将G&P和C相结合，可以得到GP/C，使得全加器延迟由N变为logN

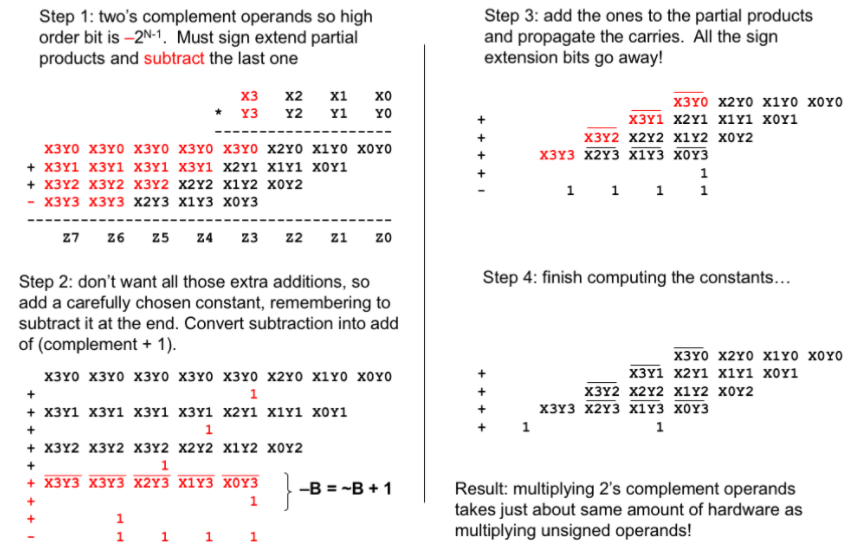
## 二进制乘法



## 组合乘法器

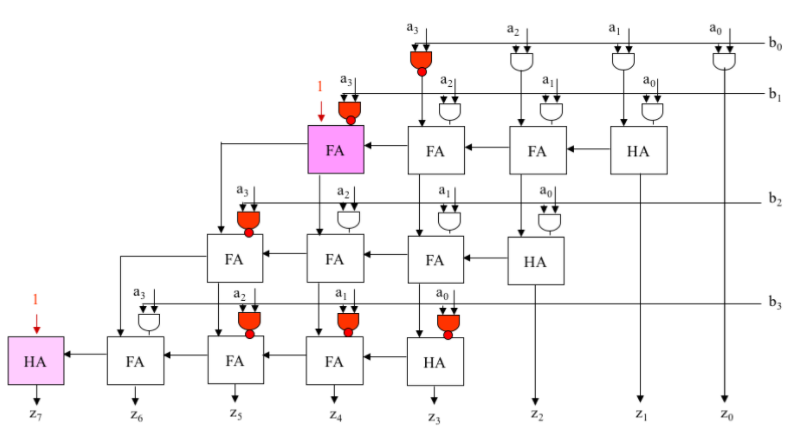


## 二进制补码乘法

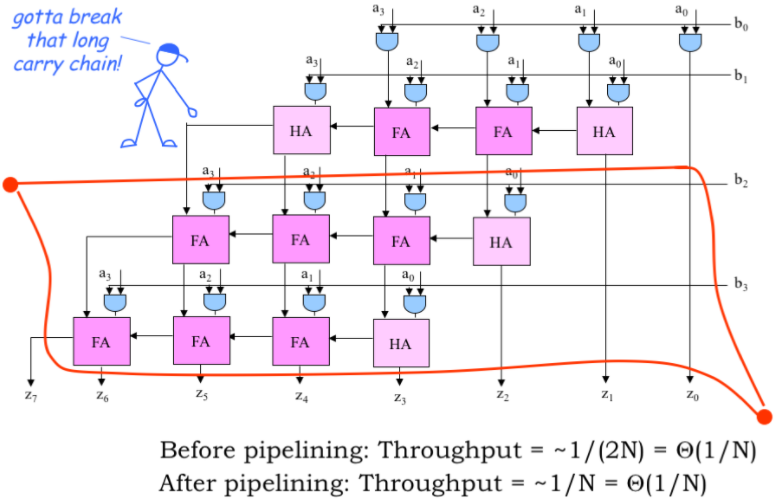


补码乘法，经过种种化简后，与无符号乘法所需硬件几乎一样

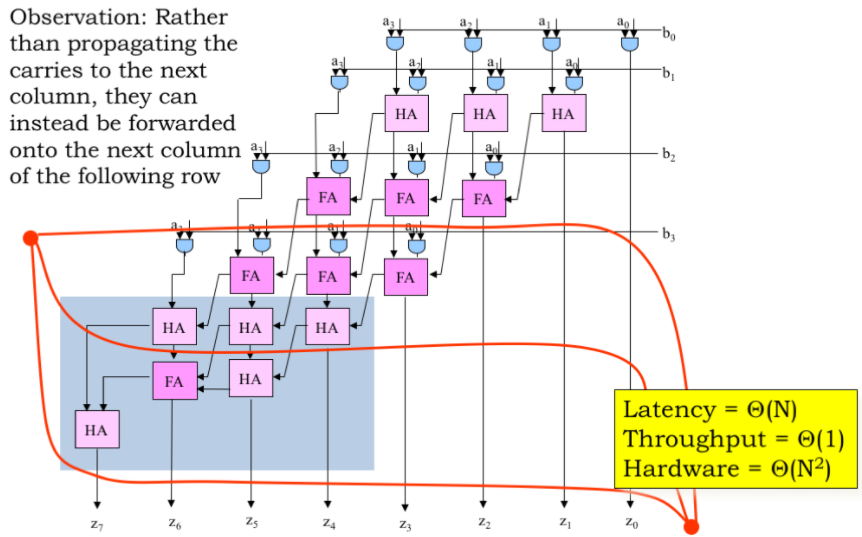
## 二进制补码乘法器



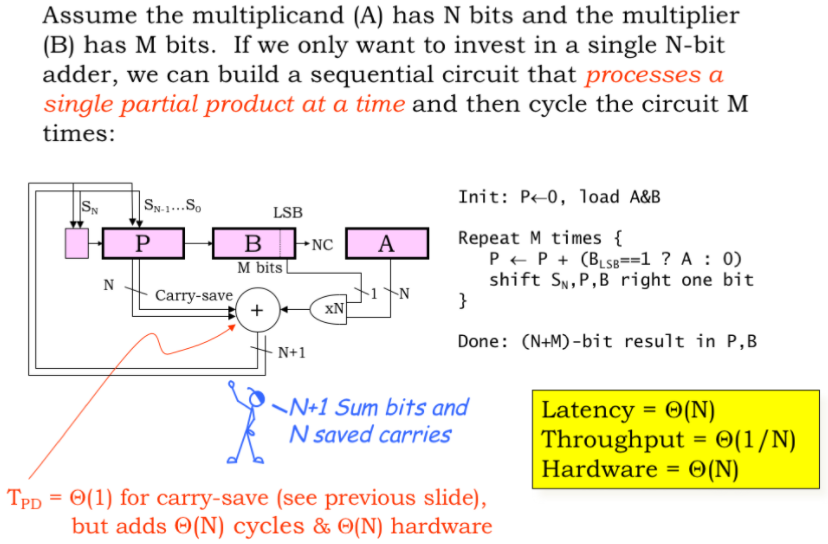
## 通过流水线提升吞吐



## 无进位流水线乘法器



## 用串行逻辑节约空间



## 总结

