北京大学
PEKING UNIVERSITY

人工智能的硬件基石

从物理器件到计算架构

第二讲：新兴技术简介、半导体晶体管基本原理

主讲：陶耀宇

2026年春季

注意事项

北京大学 PEKING UNIVERSITY

- 课程作业情况

- **作业将在下周一，3月底-4月中旬、4月中旬-5月初、5月中旬-6月初**

  2-3周完成时间

- **第1次lab时间：4月10日-5月10日**
- **第2次lab时间：5月10日-6月15日**

注意事项

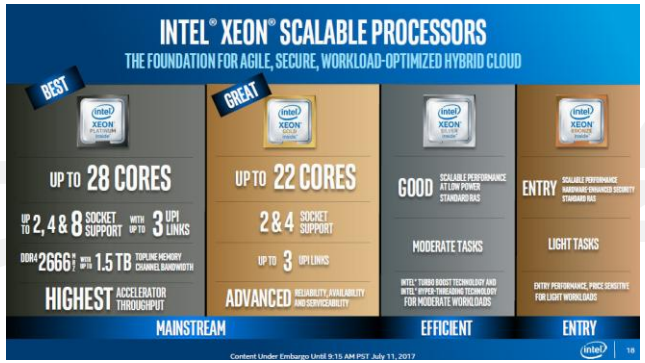
北京大学 PEKING UNIVERSITY

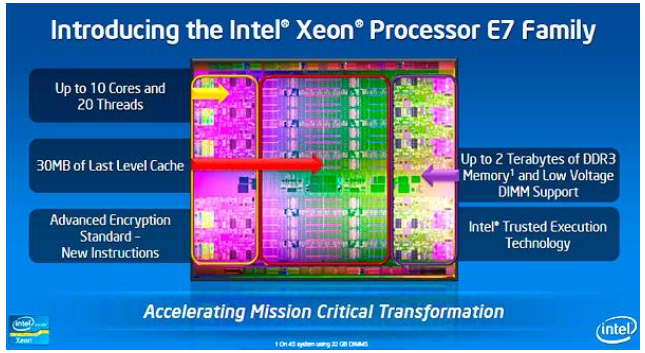
- **课程作业情况**

- CLAB平台：请各位选课同学在第2-3周，**使用学号登陆CLab平台**，并同意用户协议以激活账号
- 激活账号是后续lab能够进行的必要条件，请务必完成！
- Clab网址：clab.pku.edu.cn
- **Clab问题请联系助教李中源、罗子翔同学**

中国的“卡脖子”领域之计算架构：高性能处理器芯片

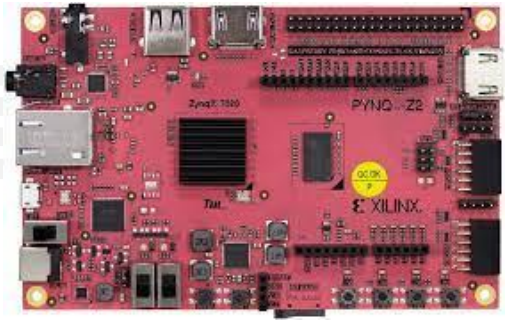
- 我国在高性能计算芯片CPU、GPU、FPGA的指令集与架构设计领域目前落后较多





高性能CPU遭美国出口管制禁运

高性能GPU遭美国出口管制禁运



高性能可编程逻辑FPGA与美国主流厂商Altera、Xilinx差距明显

国产龙芯3C5000目前已可商用，但性能与至强系列仍有显著差异

国产GPU尚处于初级阶段

国产GPU包括摩尔线程、壁仞科技、燧原科技、天数智芯、景嘉微等，与英伟达差距很大

国产FPGA包括紫光同创、安路科技、复旦微等，在并行规模、功能灵活性上急需进步

中国的“卡脖子”领域之芯片设计软件：EDA产业

- 我国在高性能的电路辅助设计与仿真工业软件方面目前与发达国家差距明显

现状

2019年

EDA三巨头Cadence、Synopsys、Mentor等已经对华为断供EDA工具，不再出售新license

2022年

2022年8月美国将停止对中国出口GAA相关EDA软件，GAA主要用于3nm及以下晶体管

国产EDA软件目前门类已补齐，但制程支持、设计仿真性能、与晶圆厂对接等多方面仍处于落后状态

目 录

CONTENTS

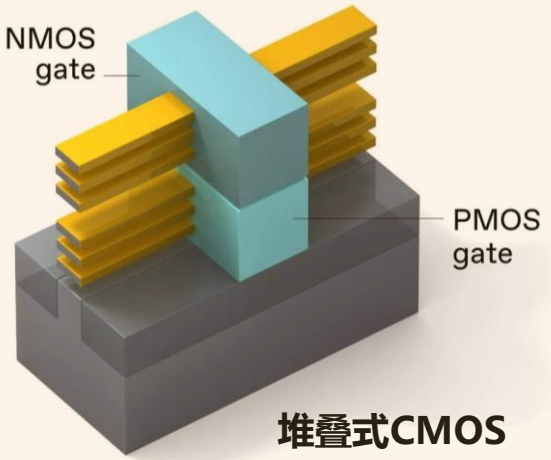
融合新器件、新架构、新计算是后摩尔时代体系结构的发展趋势

北京大学 PEKING UNIVERSITY

- 融合新器件、新架构、新计算是突破后摩尔时代大算力、高能效瓶颈的重大关键技术领域

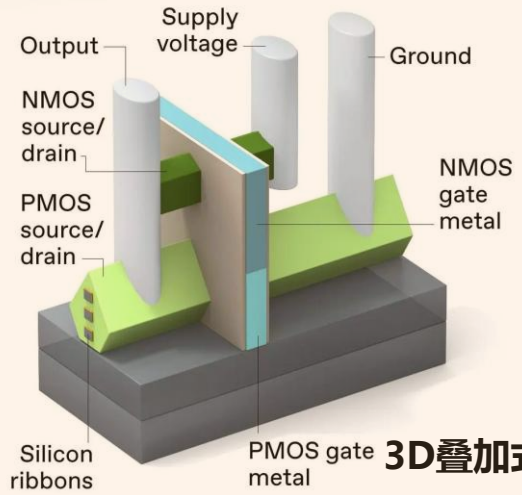
代表性智能芯片新兴技术– 新器件：高密度的逻辑器件

- 未来三维堆叠式晶体管与碳管器件

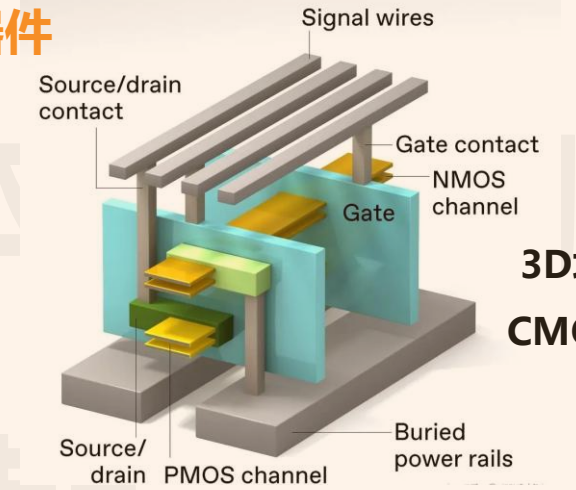


堆叠式CMOS

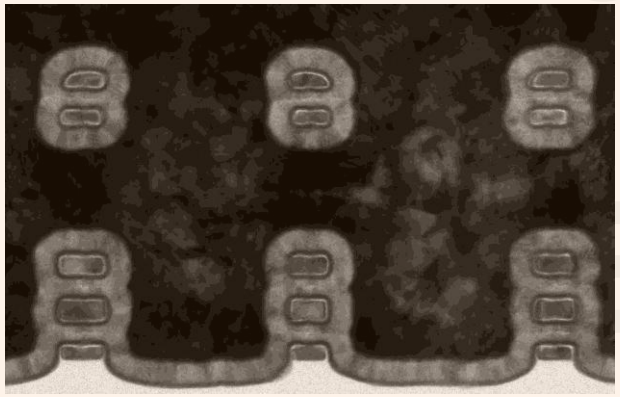
新兴硅基器件

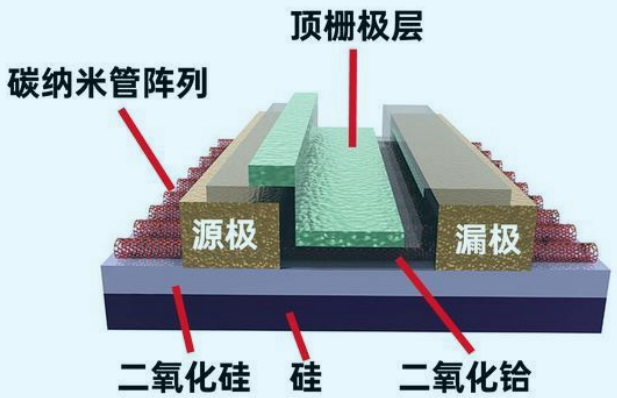


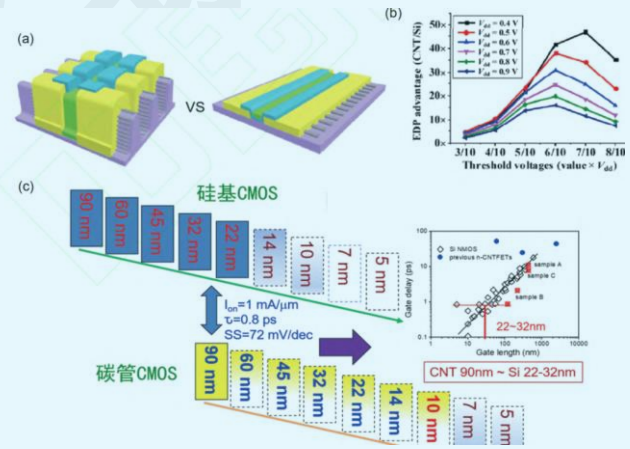
3D叠加式非门



3D堆叠的CMOS连接



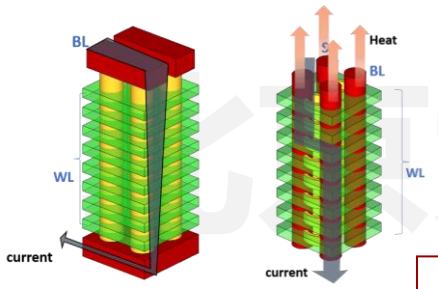




新兴碳基器件

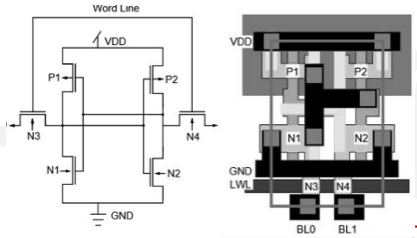
代表性智能芯片新兴技术– 新器件：存储-计算融合器件

- 未来存储器介质材料的创新



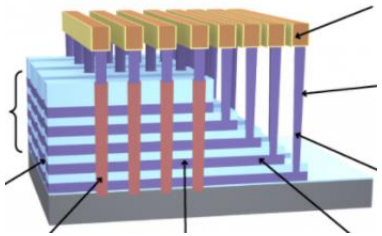
DRAM

优点：工艺成熟、密度高
缺点：速度低、刷新、只近存
非易失性：否
适合场景：冯氏架构过渡



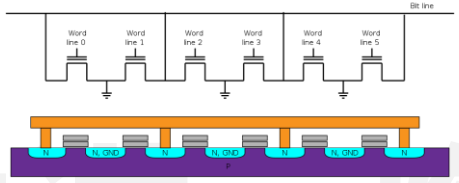
SRAM

优点：工艺成熟、IP化应用
缺点：能效低、密度低
非易失性：否
适合场景：端侧、边缘中小算力



SSD/Nand Flash

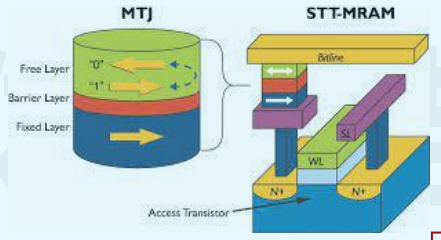
优点：工艺成熟、容量大、成本低
缺点：速度低、只能近存
非易失性：是
适合场景：云端大容量



Nor Flash

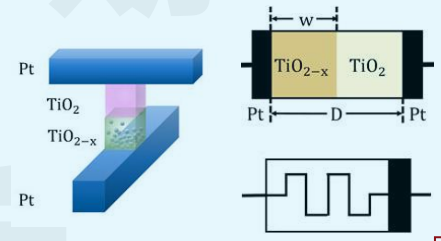
优点：工艺成熟、密度高、成本低
缺点：对PVT变化敏感、能效低
非易失性：是
适合场景：端侧、边缘低成本

新兴器件



磁器件 (MRAM)

优点：能效、速度、密度高
缺点：与CMOS大规模集成难
非易失性：是
适合场景：端侧、边缘中小算力

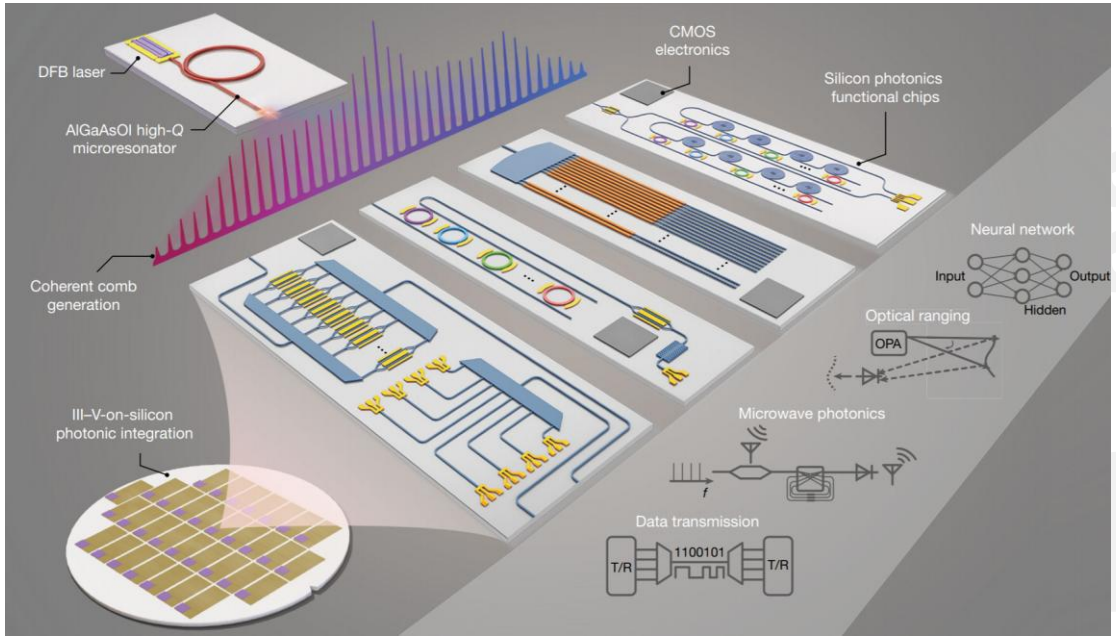


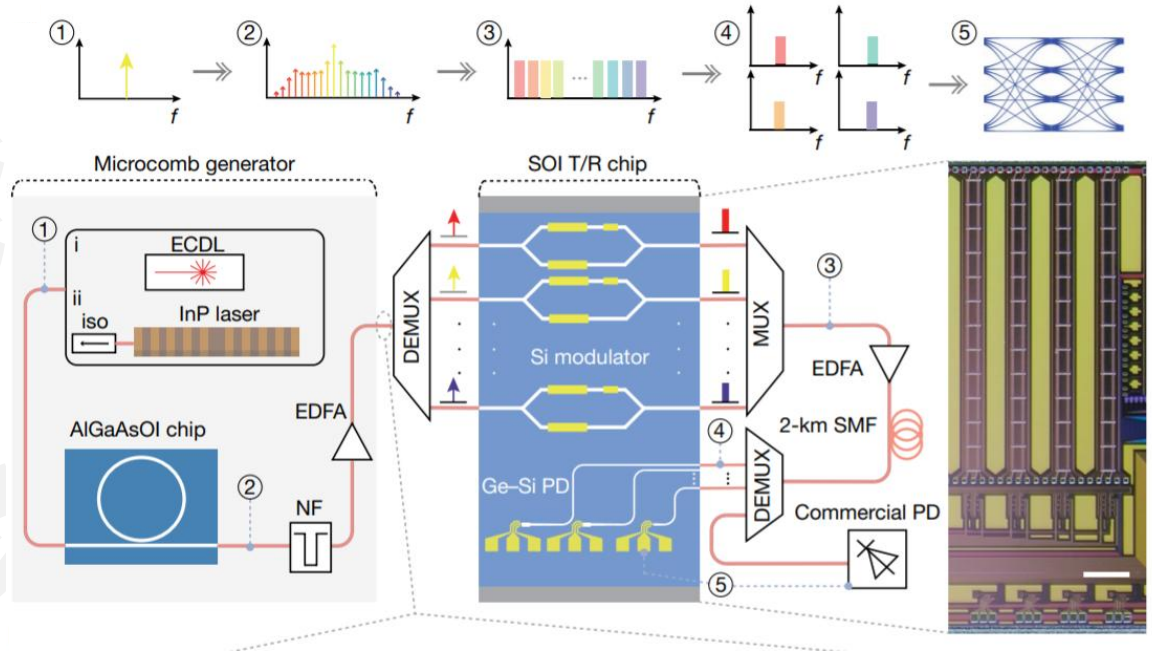
忆阻器 (RRAM/PCM)

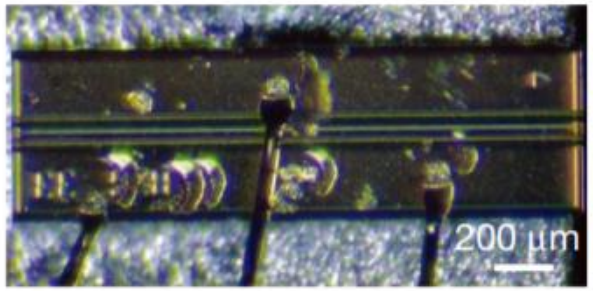
优点：算力、能效、密度高
缺点：工艺爬坡成熟中
非易失性：是
适合场景：云边端大算力

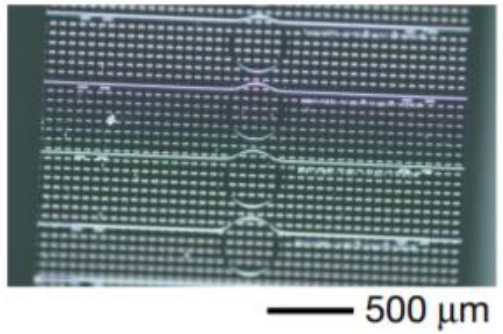
代表性新兴技术 – 新器件：光器件与片上光互连技术

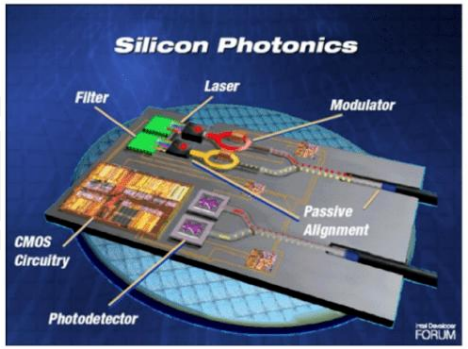
- 片上集成光电子通信系统有望突破信号传递延时的瓶颈，打破金属互连的物理上限

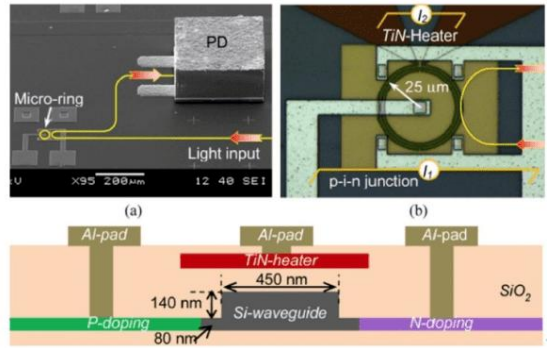






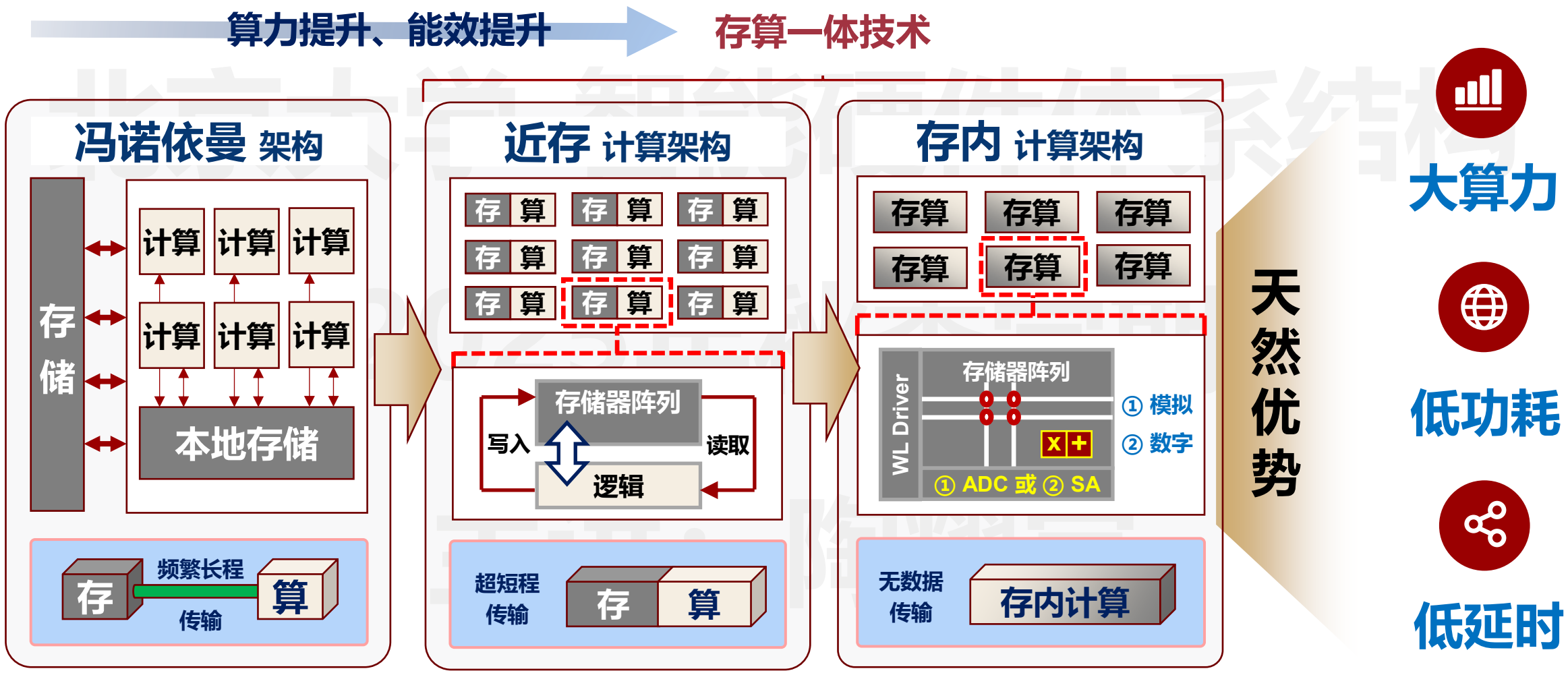






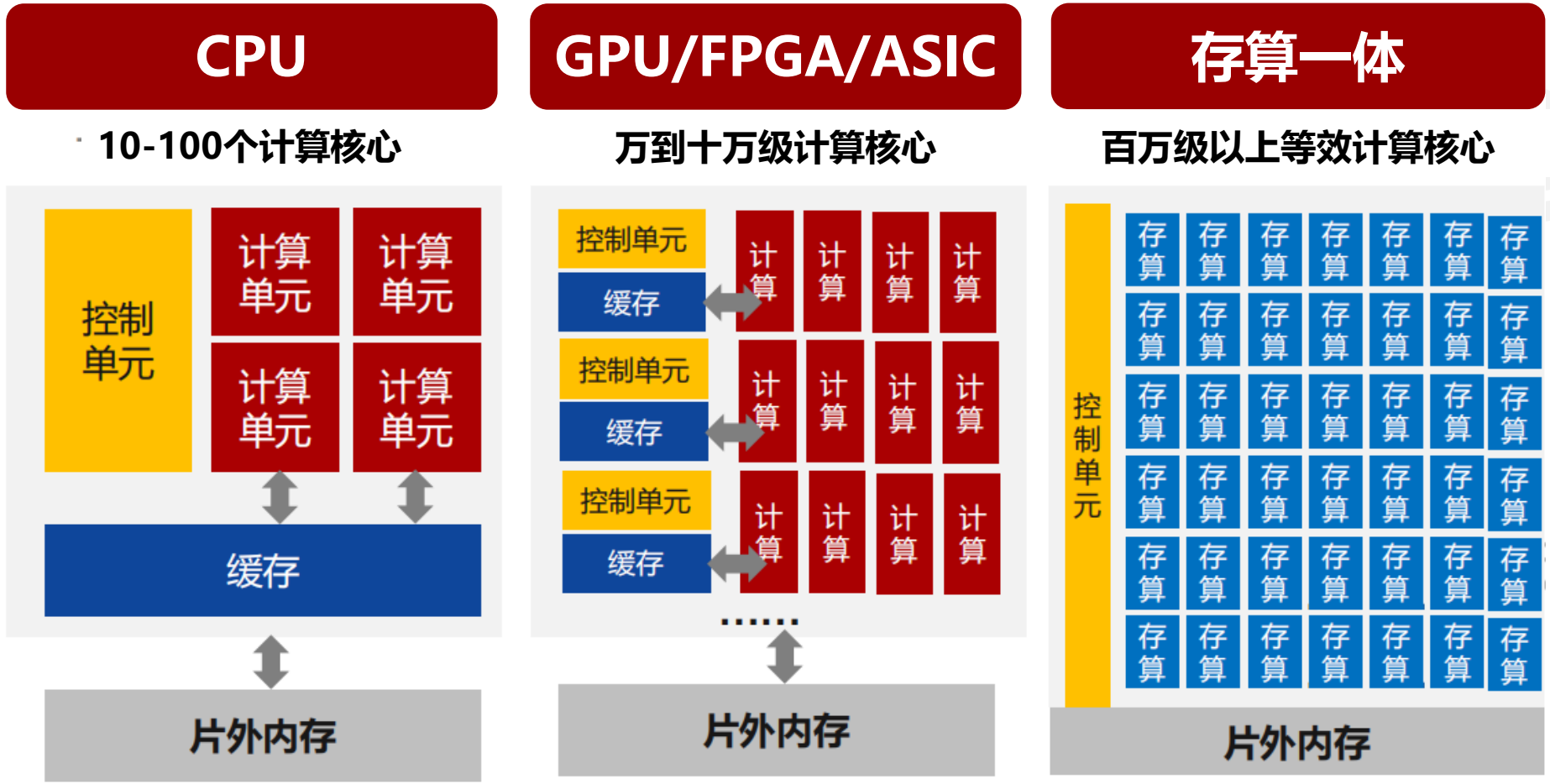
代表性智能芯片新兴技术 – 新架构：存算一体

- 存算一体技术成为后摩尔时代打破算力瓶颈的重要路径



存算一体成为打破AI大模型推理算力极具潜力的技术路径

- 存算一体提供比GPU等冯氏芯片高多个数量级的并发度，有效支撑AI大模型推理

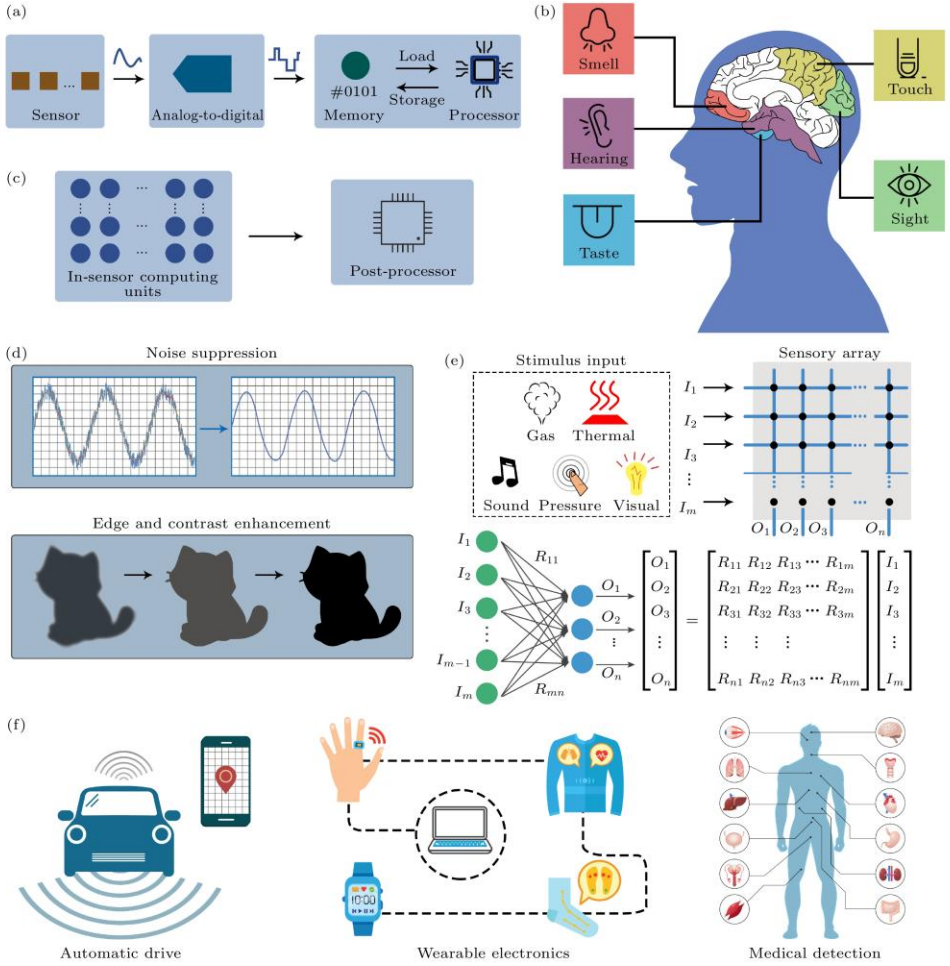


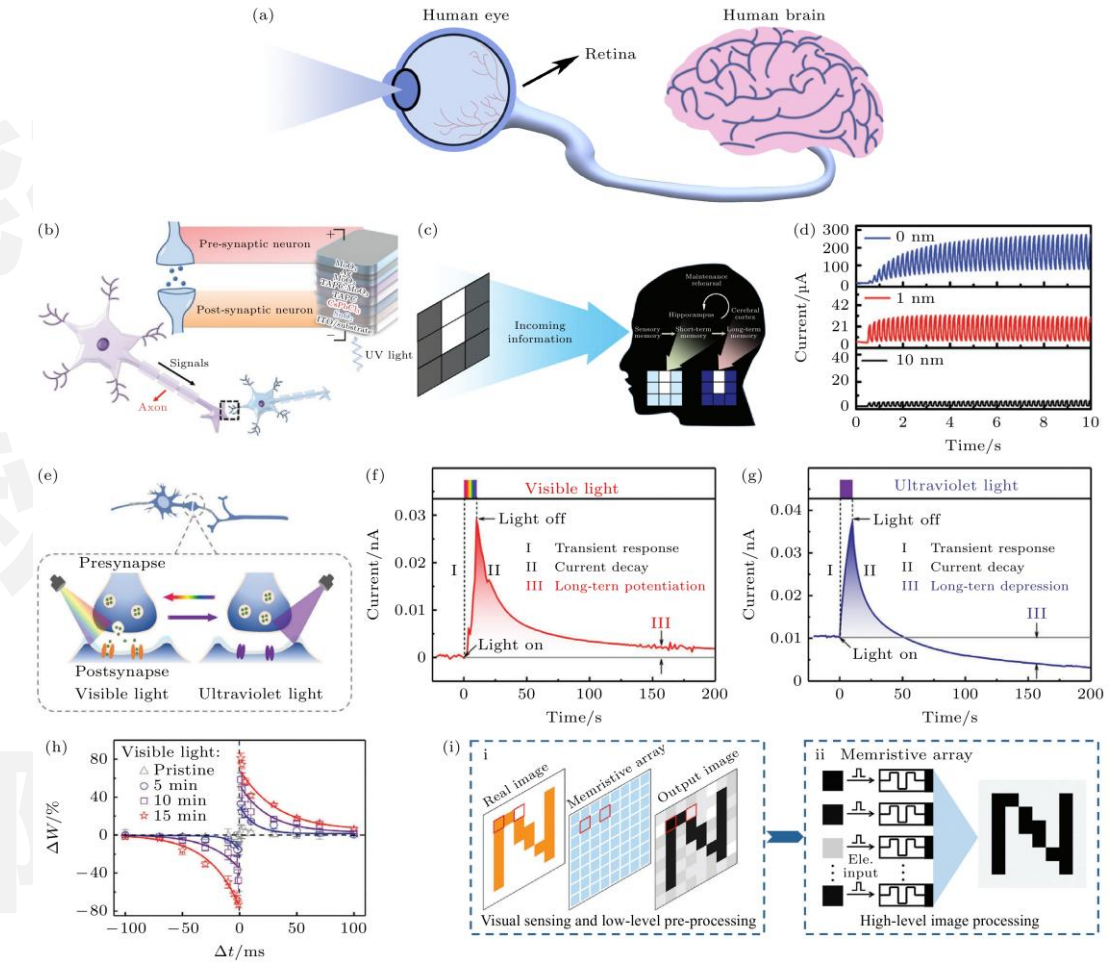
存算一体如何对AI大模型进行有效支持?

现有AI大模型推理基本上基于GPU/FPGA/ASIC等冯氏芯片

代表性智能芯片新兴技术 – 新架构：感存算一体

- 将传感、计算、存储融为一体，大幅降低系统功耗和计算延时，应用前景广阔



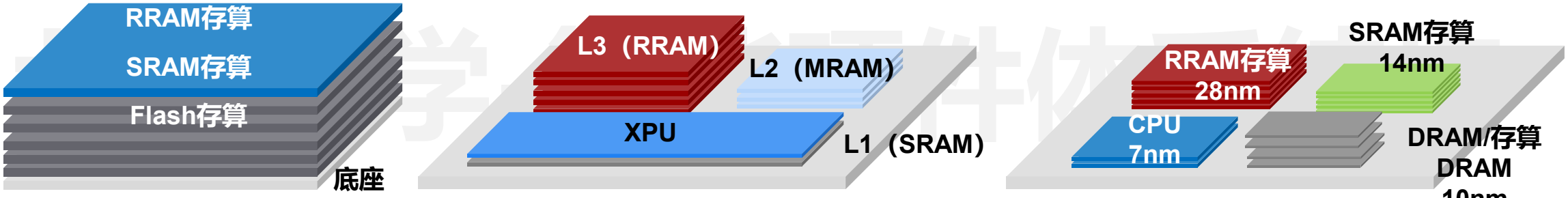


视觉感存算一体芯片与硬件系统

代表性智能芯片新兴技术 – 新架构：三维异质集成

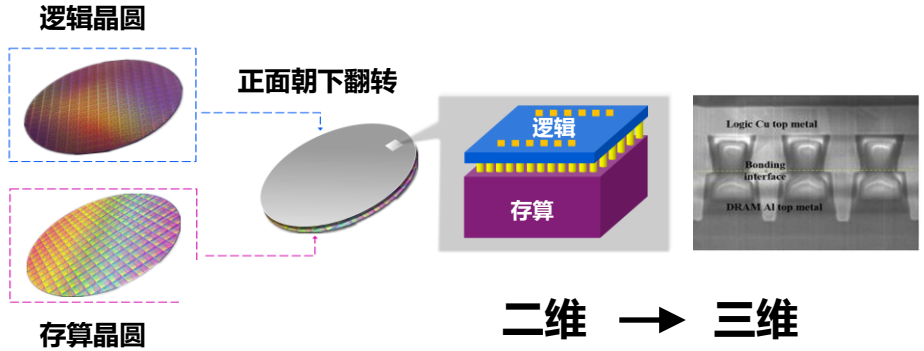
- 协同先进封装技术，实现多种芯片方案相结合

先进三维集成芯片示例图



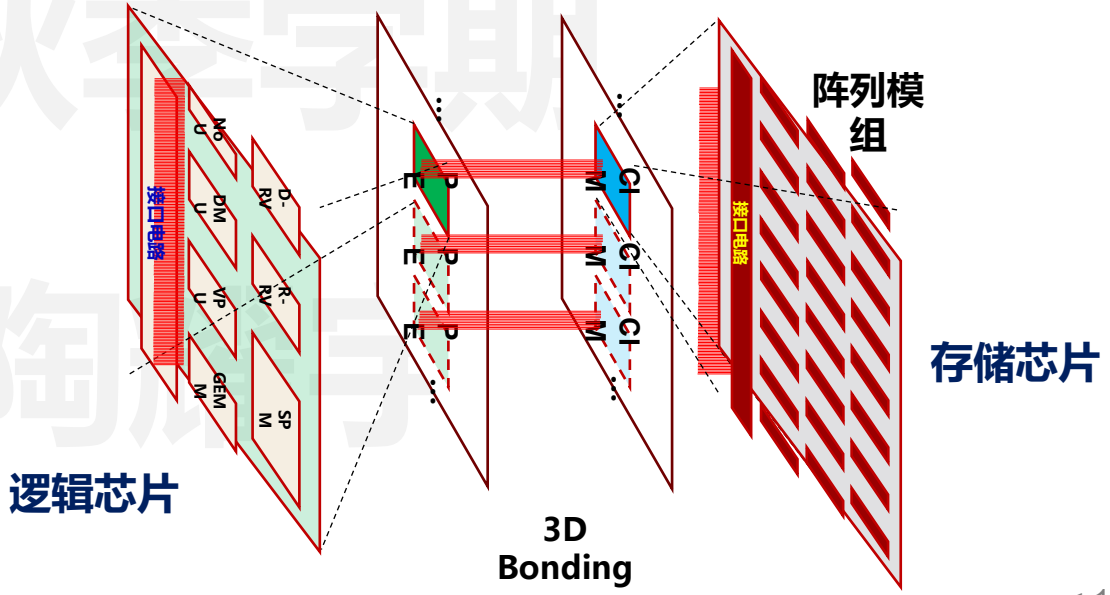
三维集成　　多级存储器堆叠SoC　　异构小芯粒封装

混合键合异质三维集成



二维 → 三维

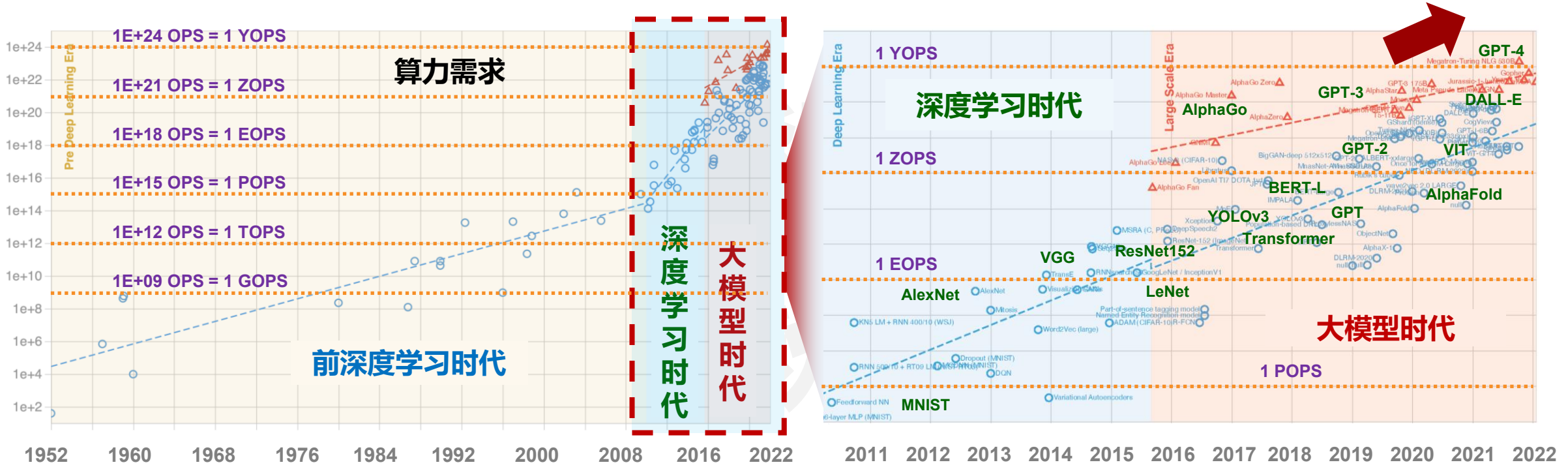
解决线路拥塞、突破面积约束、兼容不同制程、发挥各自优势



代表性智能芯片新兴技术 – 新计算：AI大模型

北京大学 PEKING UNIVERSITY

- 以AI大模型为代表的新一代人工智能系统对高性能AI芯片提出了新的要求



历史时期	算力需求	翻倍间隔
前深度学习时代 1952 – 2010	30 KOPS – 200 TOPS	21.3月
深度学习时代 2010 – 2022	700 TOPS – 2 EOPS	5.7月
大模型时代 2016 – 2022	1 ZOPS – 1 YOPS	9.9月

代表性AI大模型	参数量	算力需求
GPT-4	~1.5万亿个	~2.7 YOPS
GPT-3	~1746亿个	~314 ZOPS
GPT-3 Small	~1.25亿个	~224 EOPS

芯片性能成为支撑智能系统从量变产生质变的基石

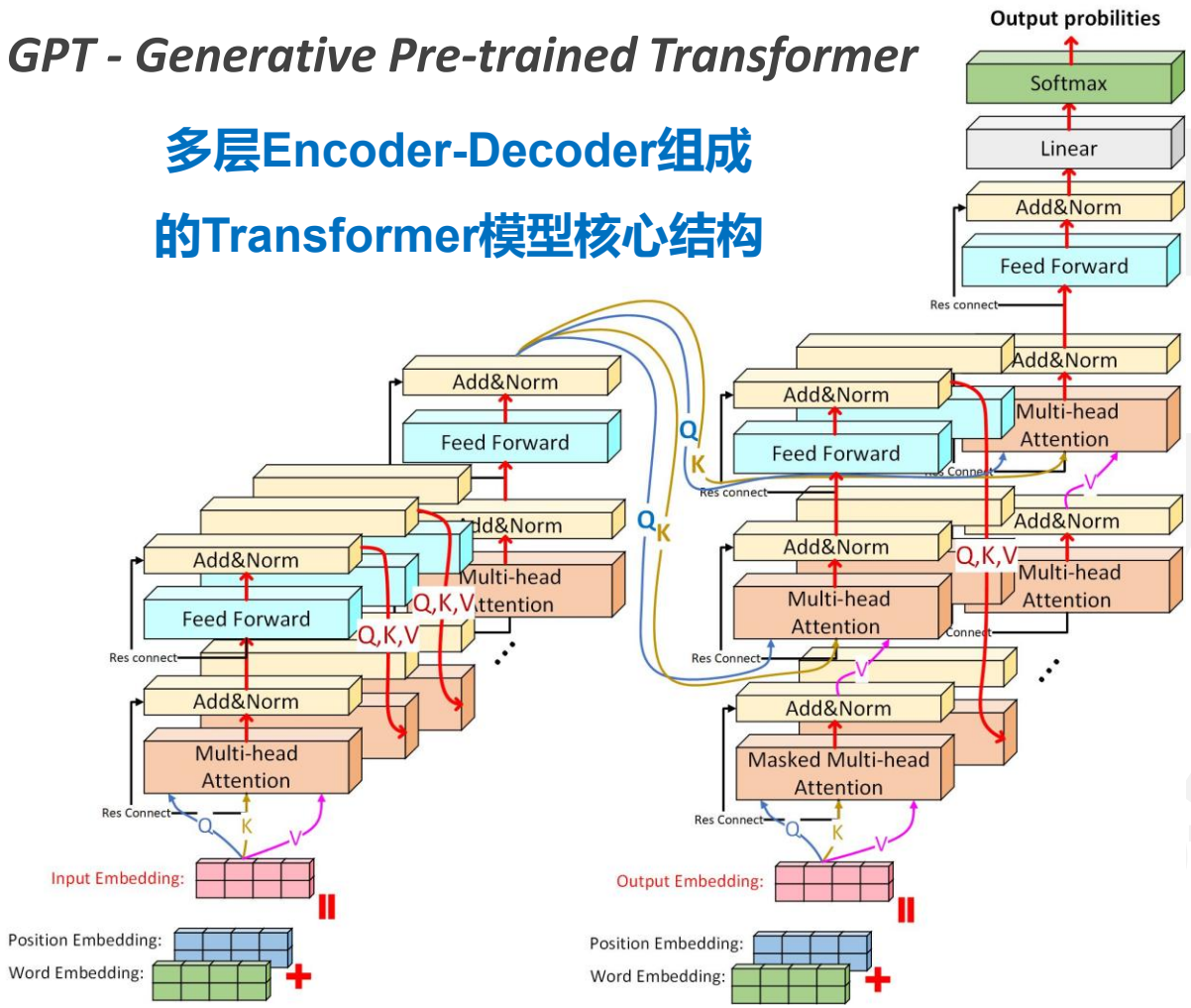
当前AI大模型以Transformer为基干网络 (以GPT为例)

- Decoder-Encoder层数、Token数量、掩码Mask尺寸、特征矩阵尺寸急剧增大

GPT - Generative Pre-trained Transformer

多层Encoder-Decoder组成

的Transformer模型核心结构



输入输出层

Input / Output Embedding

Positional Encoding

自注意力层

Multi-Head Attention

Masked Multi-Head Attention

归一化&前馈层

Add & Norm

Feed Forward

概率输出层

Linear Norm

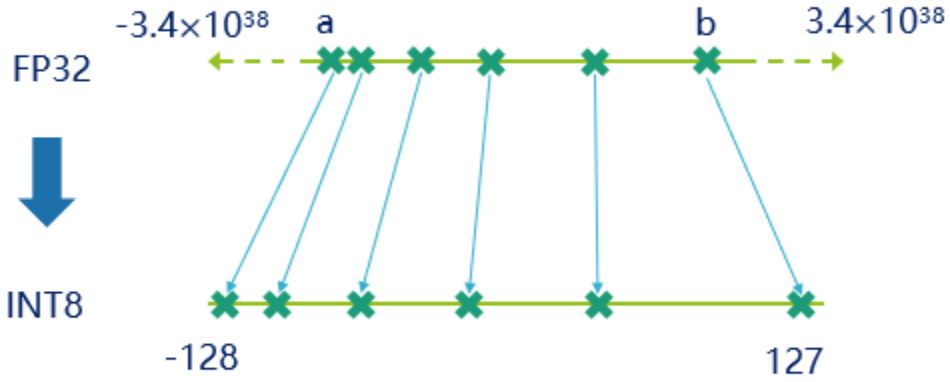
Softmax

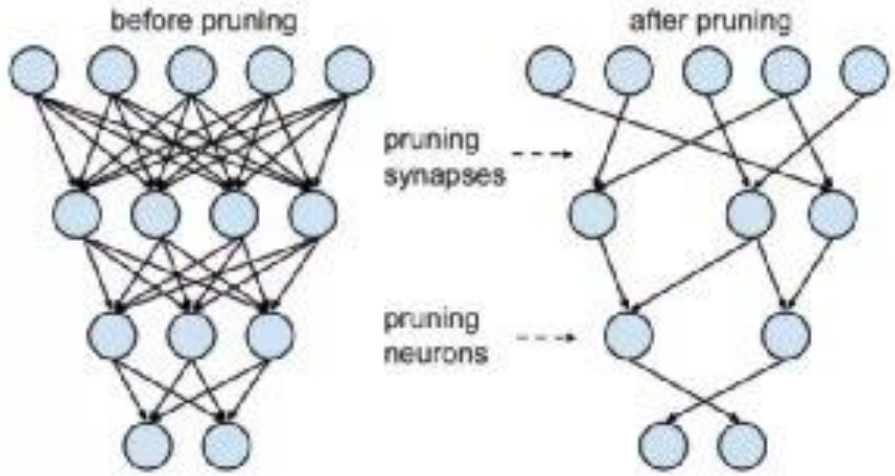
微软/OpenAI提出了**LongNet**，将Transformer的

Token数提高到了10亿级别，并持续提升

软硬件协同设计

- 面对复杂应用，单纯的硬件设计已经不足以支持性能需求，需要软硬件协同设计



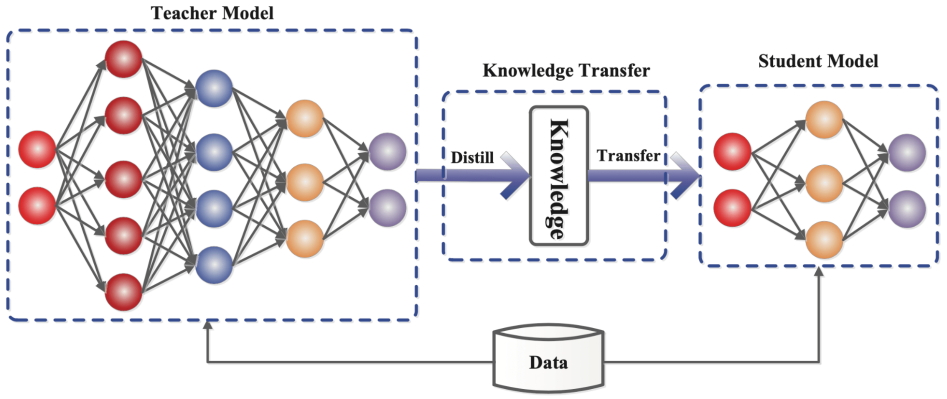


模型量化：将高精度的权重量化为低精度的权重，以一定的精度损失为代价换取更小的存储和计算开销

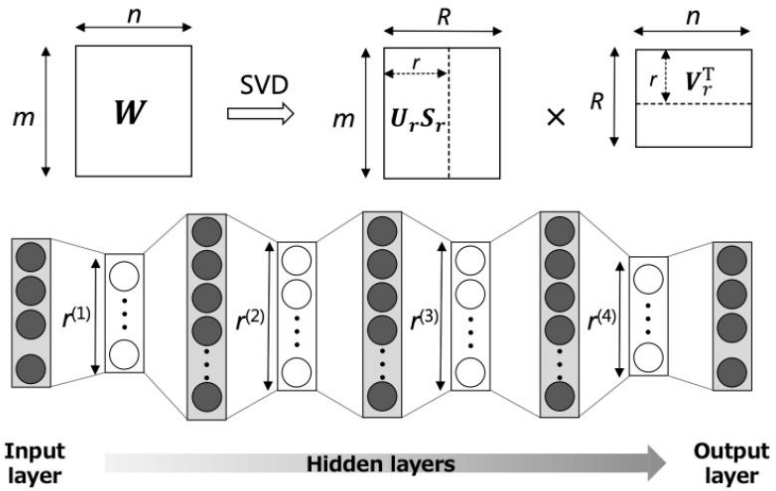
模型剪枝：将神经网络中重要性较小的神经元和权重删除，减少计算量，加速神经网络推理

软硬件协同设计

- 面对复杂应用，单纯的硬件设计已经不足以支持性能需求，需要软硬件协同设计



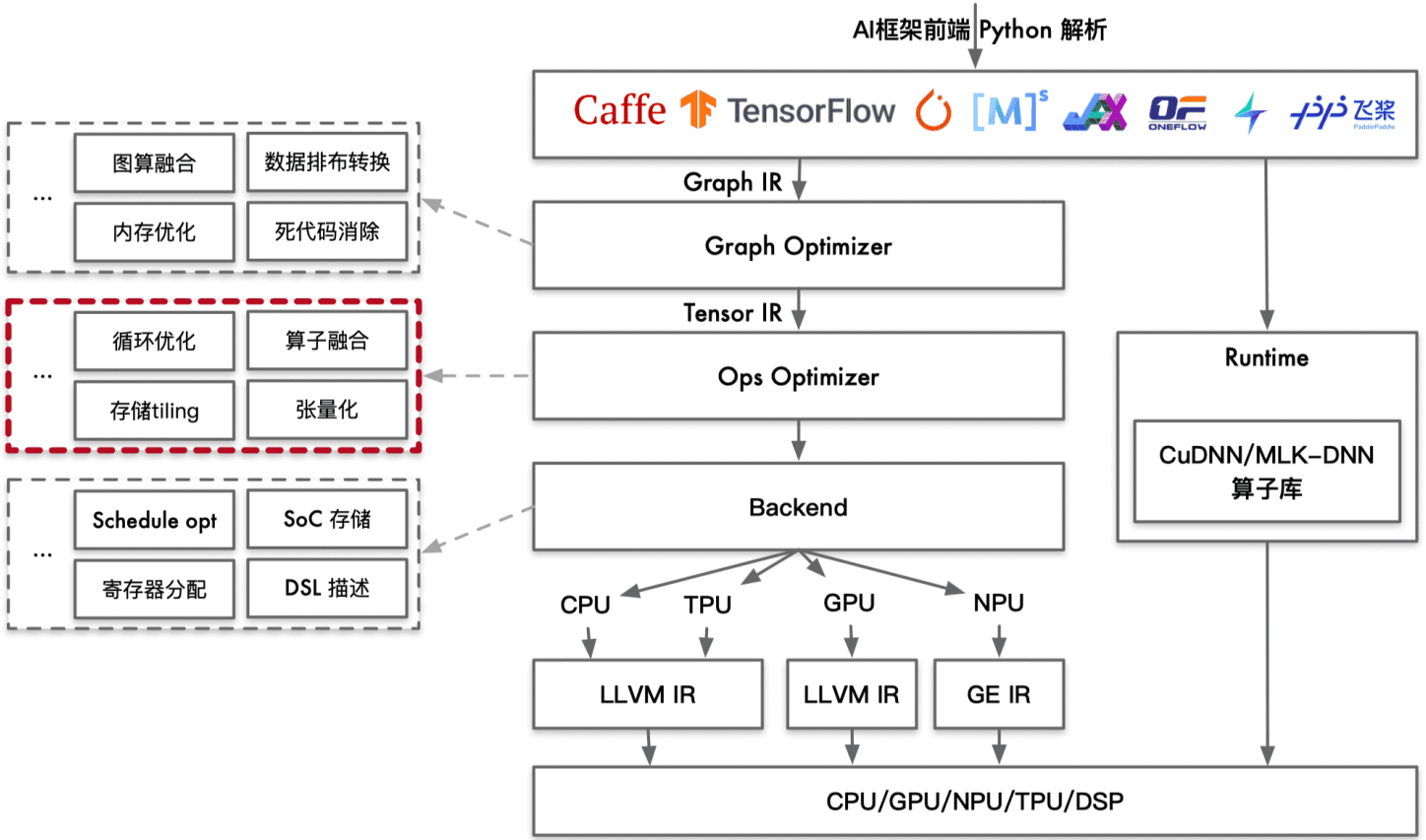
知识蒸馏：将规模较大的模型作为teacher model训练一个较小的student model，在尽可能保证性能的情况下减小模型规模



低秩分解：将大规模权重分解为两个小规模的权重矩阵相乘（SVD），减小矩阵向量乘的计算量

软硬件协同设计

- 编译层面优化



在程序编译过程中

对算子、存储tiling

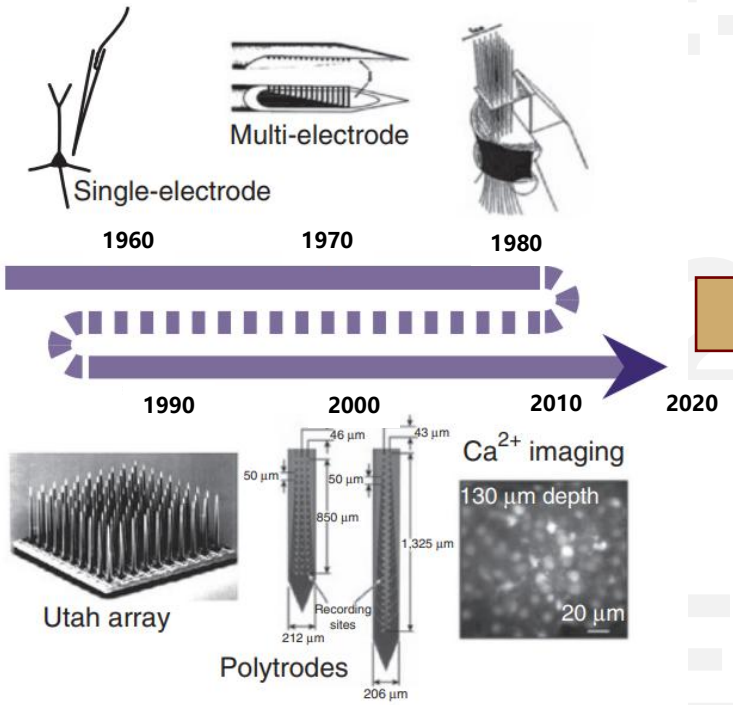
和寄存器分配等等

方面进行优化

代表性新兴技术 – 新计算：脑机接口芯片与系统

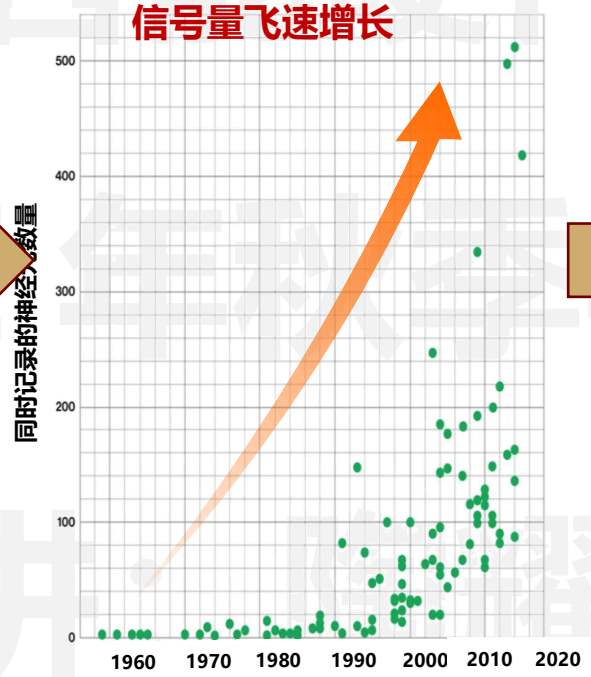
- 为脑机接口服务的芯片与系统将在未来数十年成为人类发展的方向之一

脑机数据采集工具的发展



Stevenson, I., et al. Nat Neuroscience

脑机数据量的“摩尔定律”

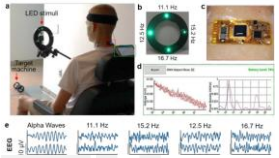


Stevenson, I., et al. Nat Neuroscience

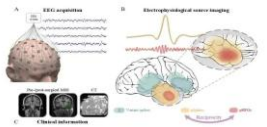
脑机打字

脑控无人机

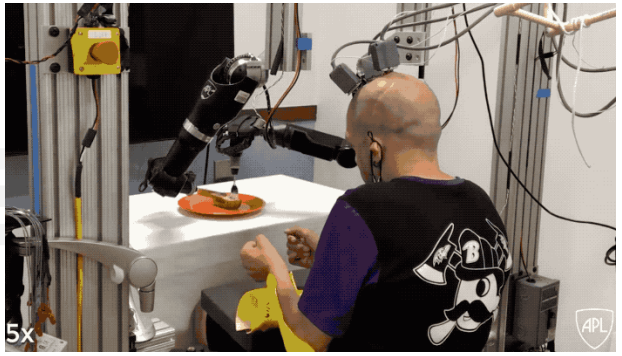
脑控座椅



脑机癫痫监测



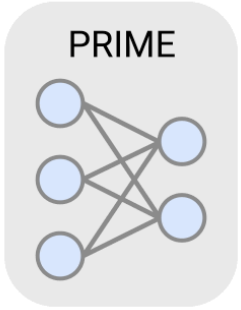
脑控机械臂



代表性新兴技术 – 新方法：AI设计AI芯片

- 设计AI芯片架构 -> 利用AI设计AI芯片架构

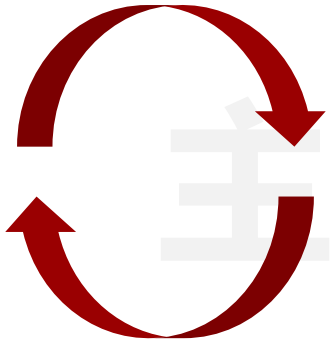
参数化硬件单元库　　针对某类任务的最优芯片设计



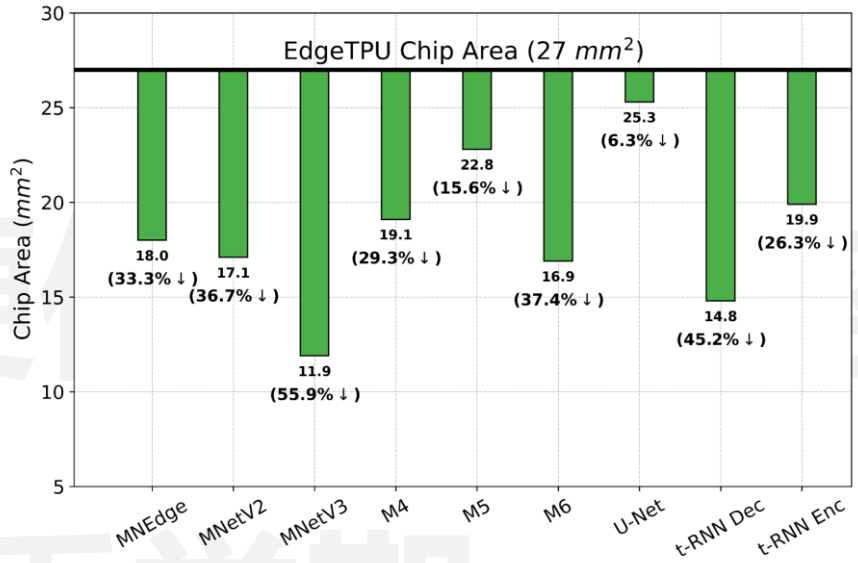
Optimizer

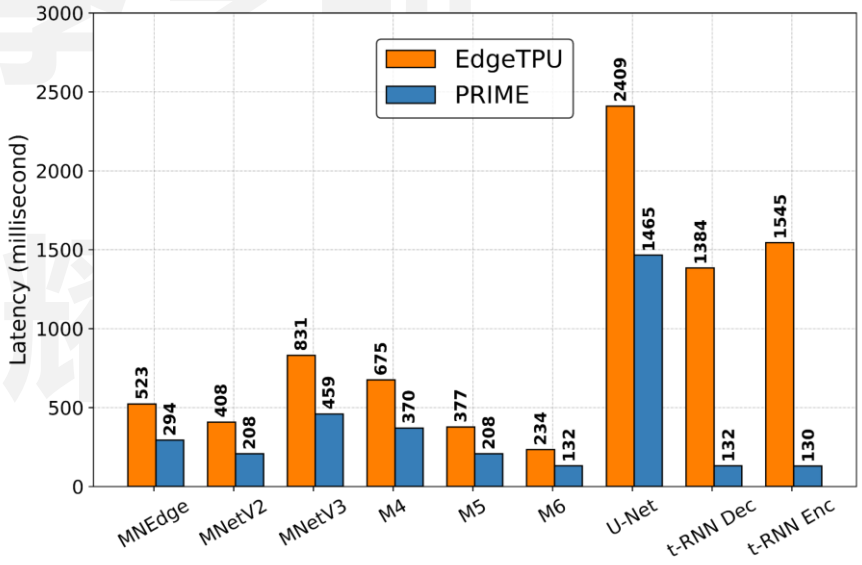
RL、大模型等方式

设计AI芯片的AI模型



AI芯片

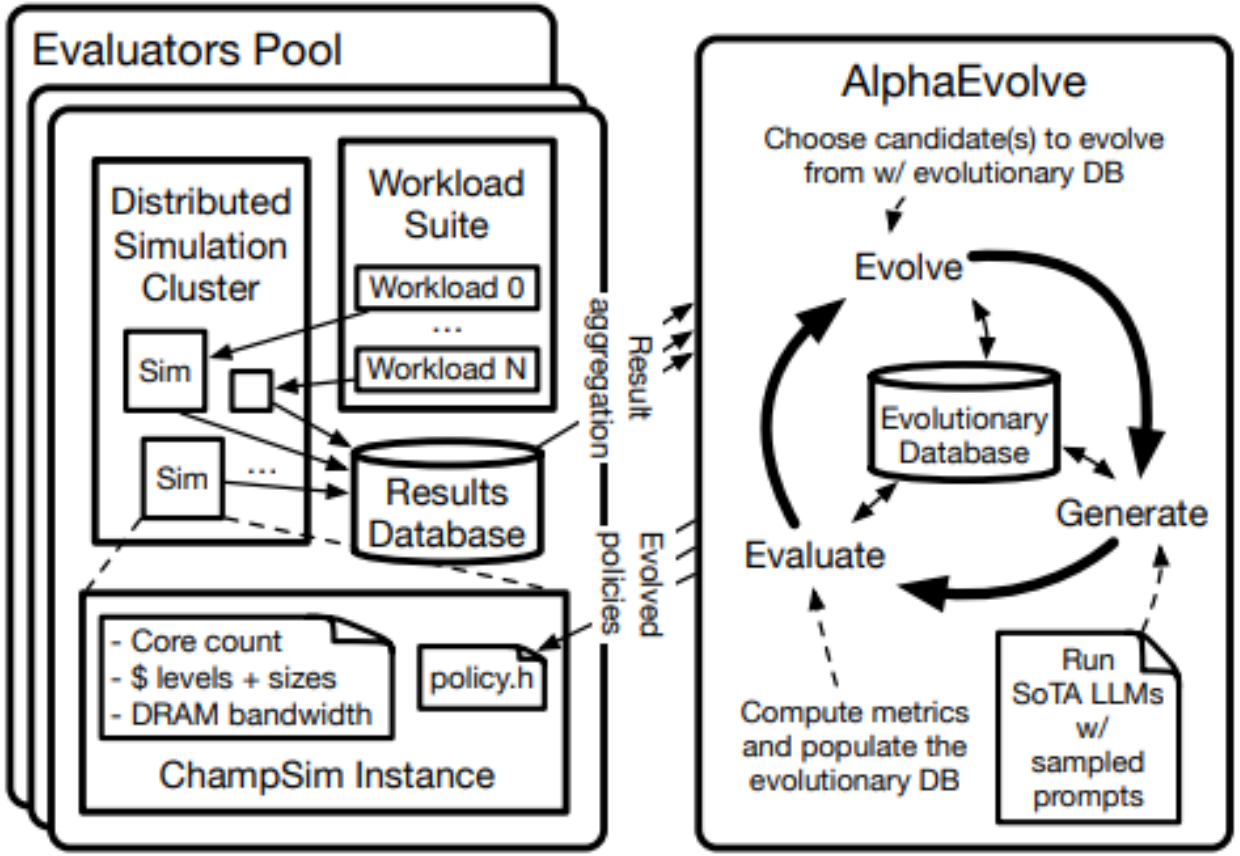




代表性新兴技术 – 新方法：AI设计AI芯片

- 设计AI芯片架构 -> 利用AI设计AI芯片架构

ArchAgent 2026: Agentic AI-driven Computer Architecture



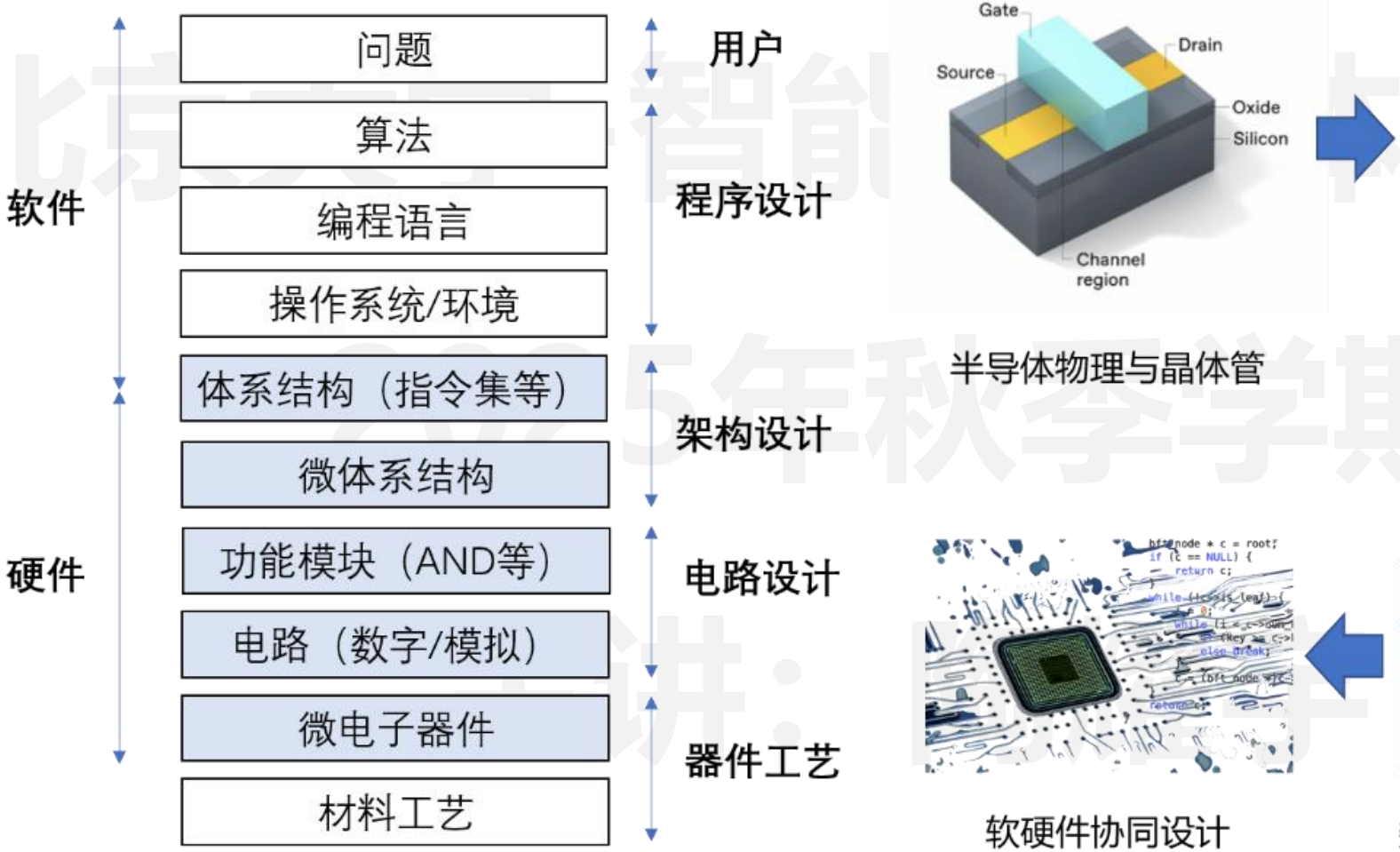
- ArchAgent（基于AI的芯片架构设计系统）的高层系统架构图
- 在此示例中，新颖的**缓存替换策略候选方案**由 AlphaEvolve 在 ChampSim（一款流行的基于 trace 的微架构模拟器）中自动设计与实现
- ChampSim 会被编译并运行指定的工作负载集（例如 SPEC），以根据目标指标（例如 IPC）来评估新策略。这一过程将迭代进行，**ArchAgent 会不断提出并评估该策略中的新逻辑/机制**

目 录

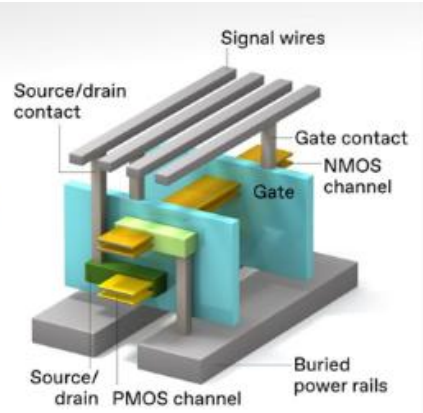
CONTENTS

MOSFET晶体 – 现代芯片的基石

- 本课程从MOSFET开始，从基础微电子器件出发

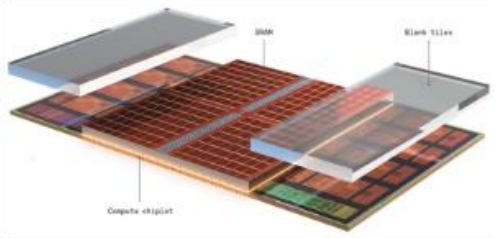


半导体物理与晶体管



数字逻辑与复杂电路

软硬件协同设计

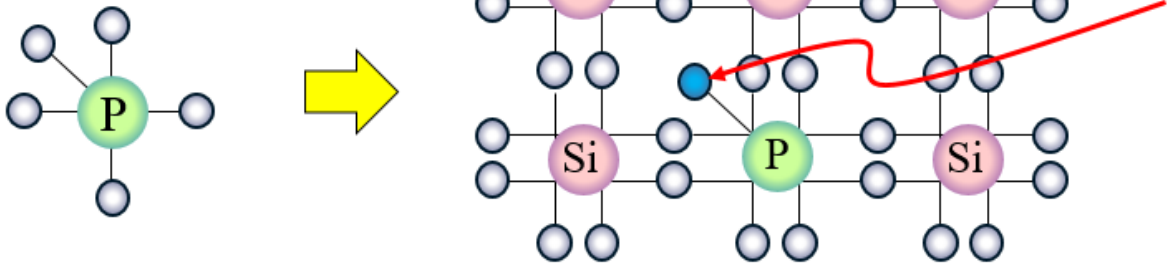


数据流、指令与芯片架构

MOSFET晶体 – 现代芯片的基石

- N型与P型半导体的概念

n型半导体



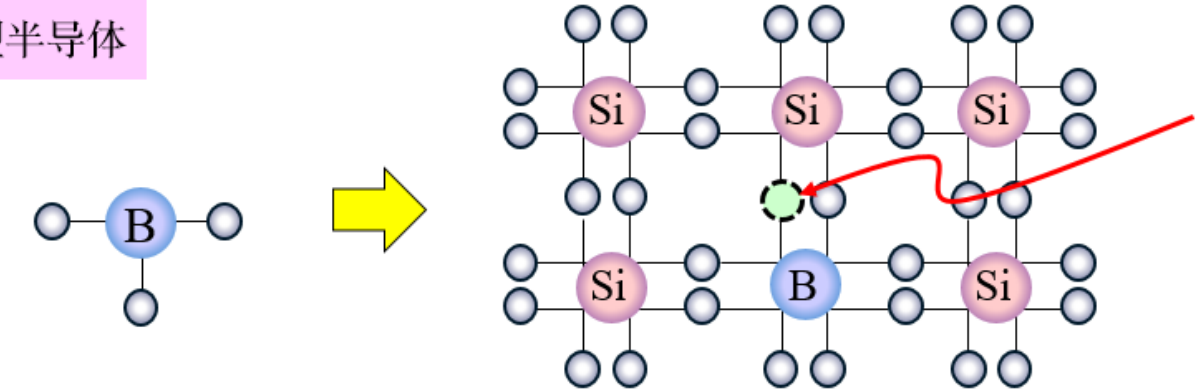
电子处于“多余”状态。

半导体内富含自由电子。

外加电压后电子就会被吸向+极

半导体变为导电的状态

p型半导体

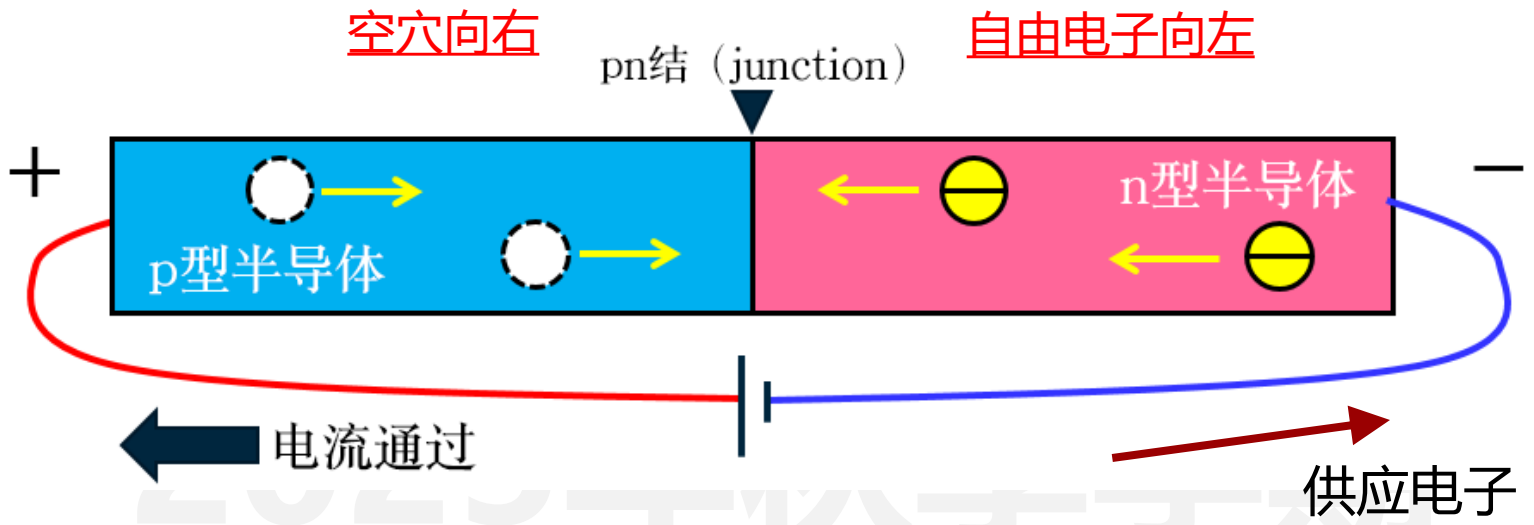


没有电子的“空位”状态

这种空位叫空穴，也就是说空穴中无实体，也叫虚拟粒子

MOSFET晶体 – 现代芯片的基石

- PN结的概念 – 导通状态



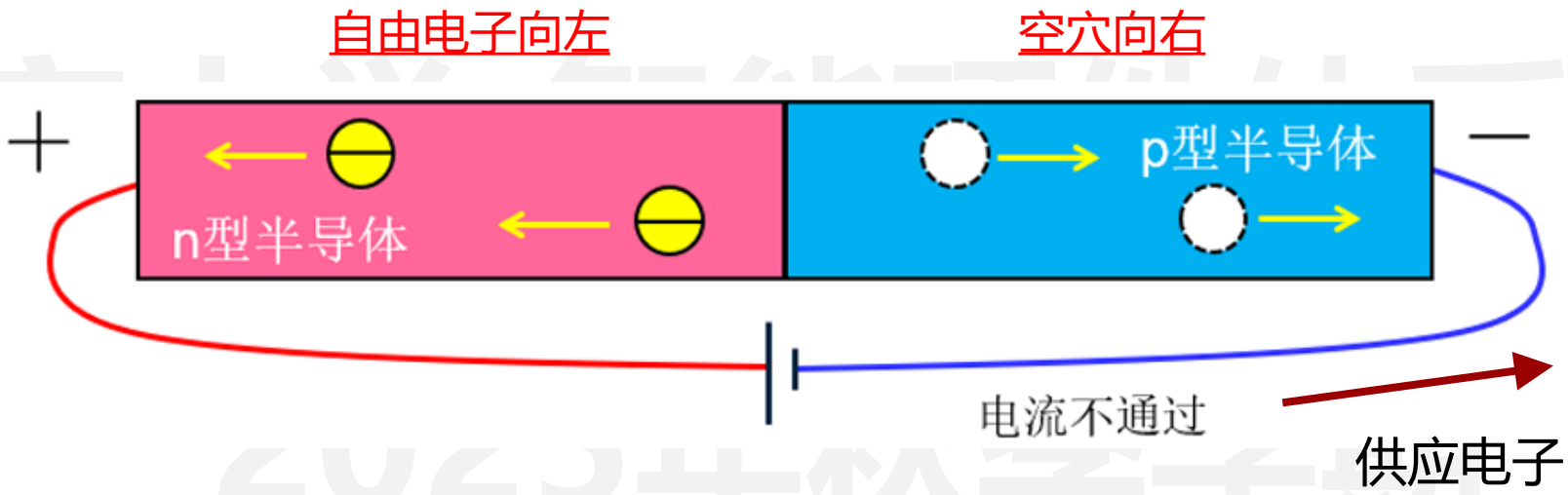
- 对PN结外加电压使P为正极，空穴和电子都向结面移动

  当空穴与电子在结面（Junction）相遇时，电子飞入空穴，两者抵消
- **相应的新电子从电源补充流入n层**，同时电子从p层流出而产生新的空穴，如此反复，电流不断通过

MOSFET晶体 – 现代芯片的基石

- PN结的概念 – 断开状态

形成depletion region（耗散层）

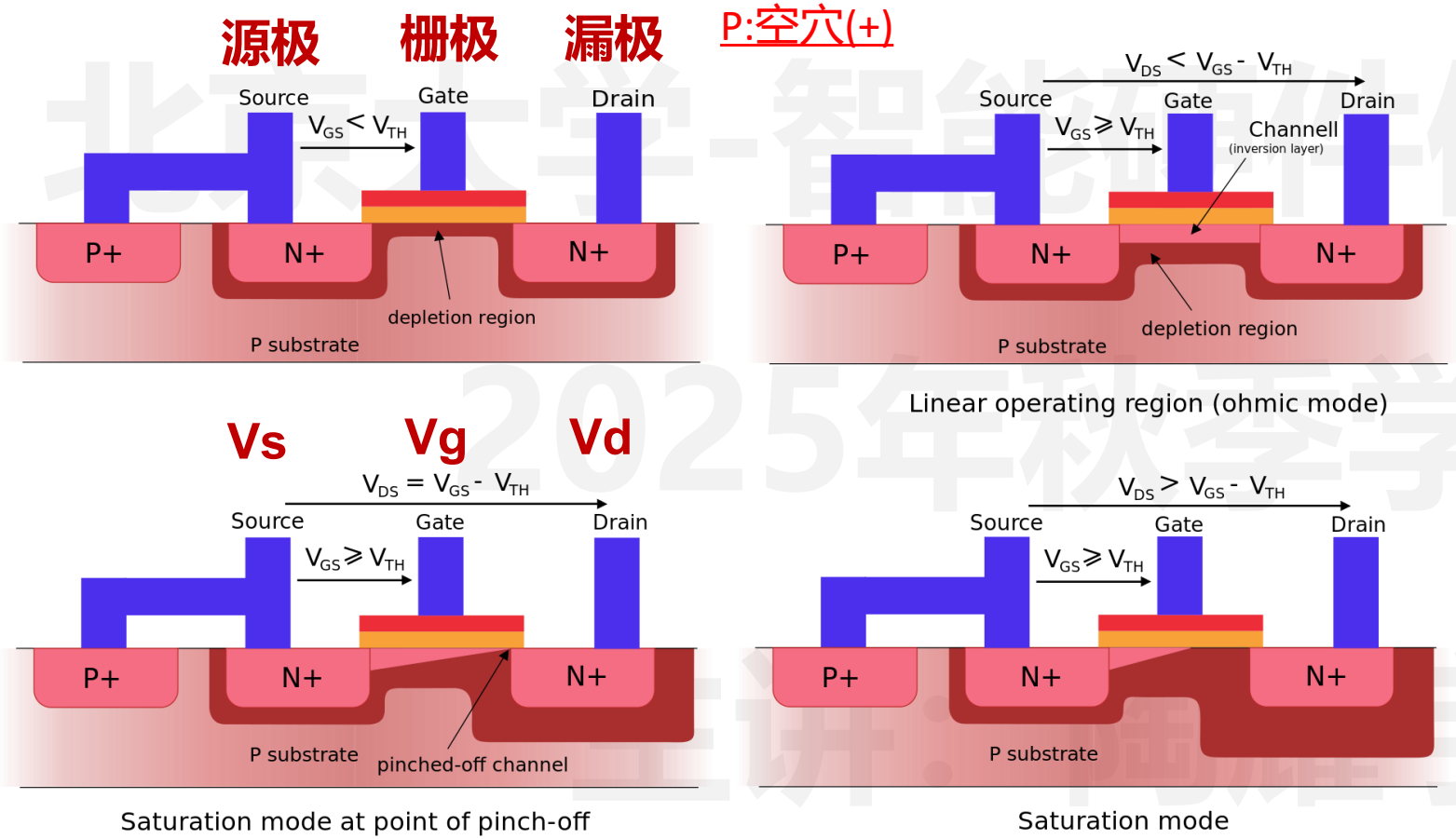


- 对PN结外加电压使N为正极。
- **空穴和电子向相互远离的方向移动**，因此不会在结面相遇，电流无法通过
- **在结面附近会形成既无空穴又无电子存在的区域，叫做耗尽层**，它会产生耐压。
- 从上述可知pn结**具有整流作用**

MOSFET晶体 – 现代芯片的基石

- MOSFET结的概念 – 栅极(gate)、漏极(drain)和源极(source)

N: 自由电子(-)
P:空穴(+)



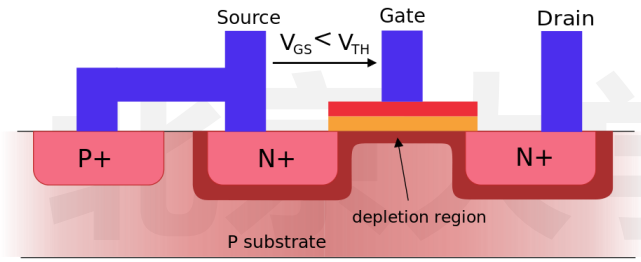
- 当栅极-源极间的电压Vgs<Vth时，N接了电源正极无论漏极-源极间的电压Vds为多少，**因为PN结的单向导通性**，不会有电流从漏极流向源极

- 当Vgs超过阈值电压Vth后，**会形成一个横跨二氧化硅层的电场**，在这种电场的作用下，**SiO2层和P区交界处附近的电子会被吸引至SiO2侧**，在SiO2侧形成一个局部电子浓度相对较高的带状区域(沿着SiO2表面)，即**N型导电沟道**(depeletion)

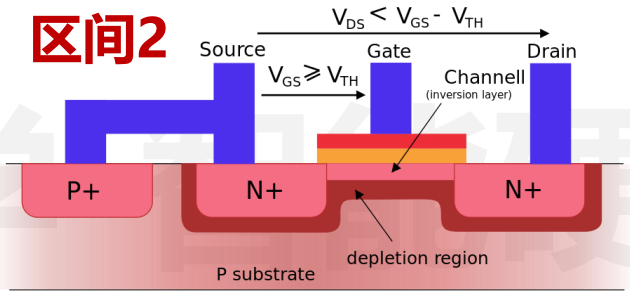
- 从N半导体电子被“吸引”向哪一侧理解上述工作区间

MOSFET晶体 – 现代芯片的基石

- MOSFET有三个工作区间：断开、线性（欧姆区间）、饱和（电压不随电流线性增加）

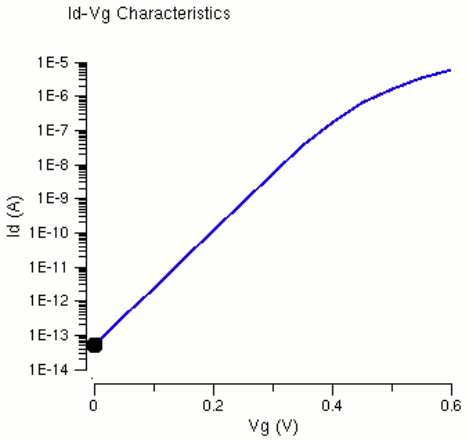


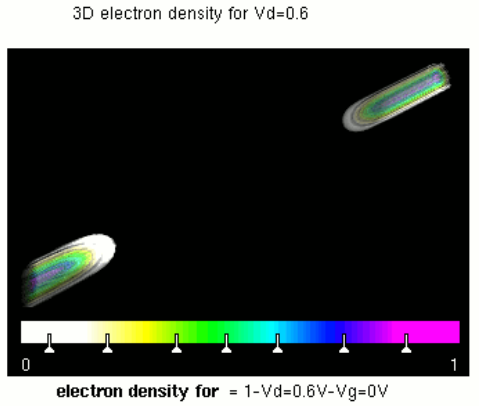
区间1　核心：Vg不足

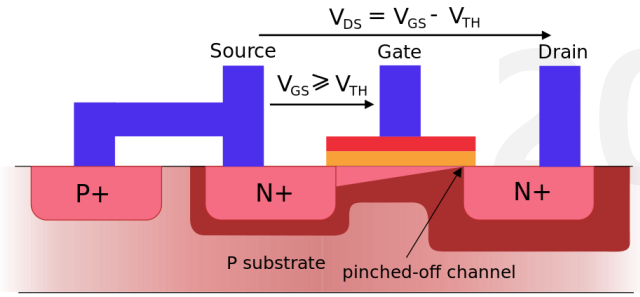


核心：Vd不足

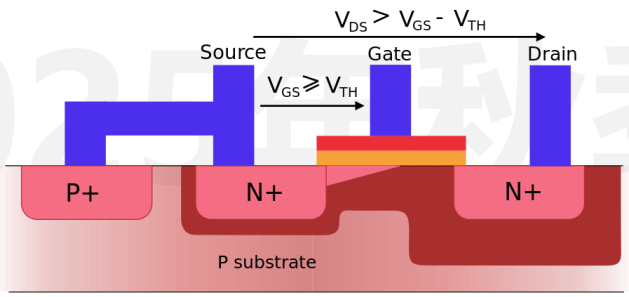
Linear operating region (ohmic mode)







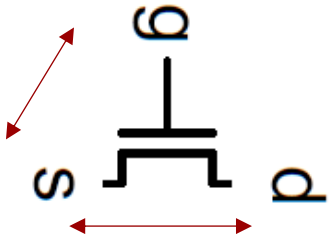
Saturation mode at point of pinch-off



Saturation mode

区间3　核心：Vd过大

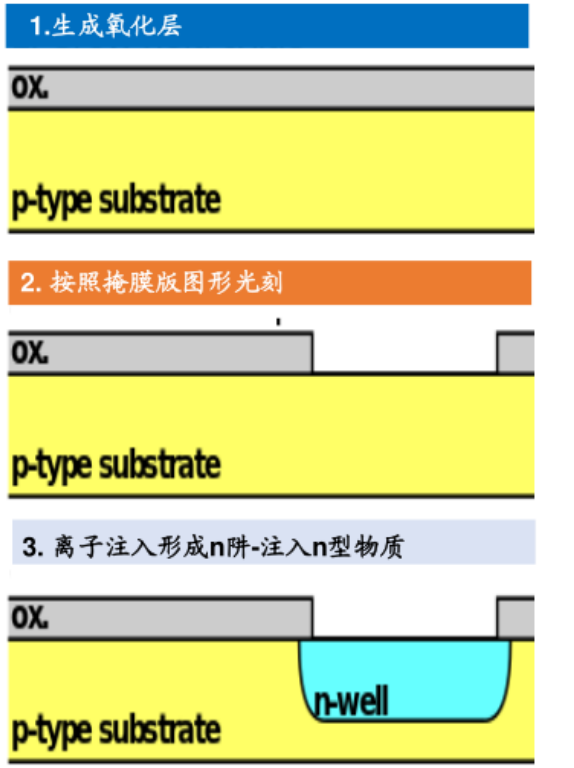
Vth：MOSFET的阈值电压
与工艺相关



1、断开区间：Vgs < Vth　2、线性区间：Vds < Vgs - Vth　3、饱和区间：Vds > Vgs - Vth

MOSFET晶体 – 现代芯片的基石

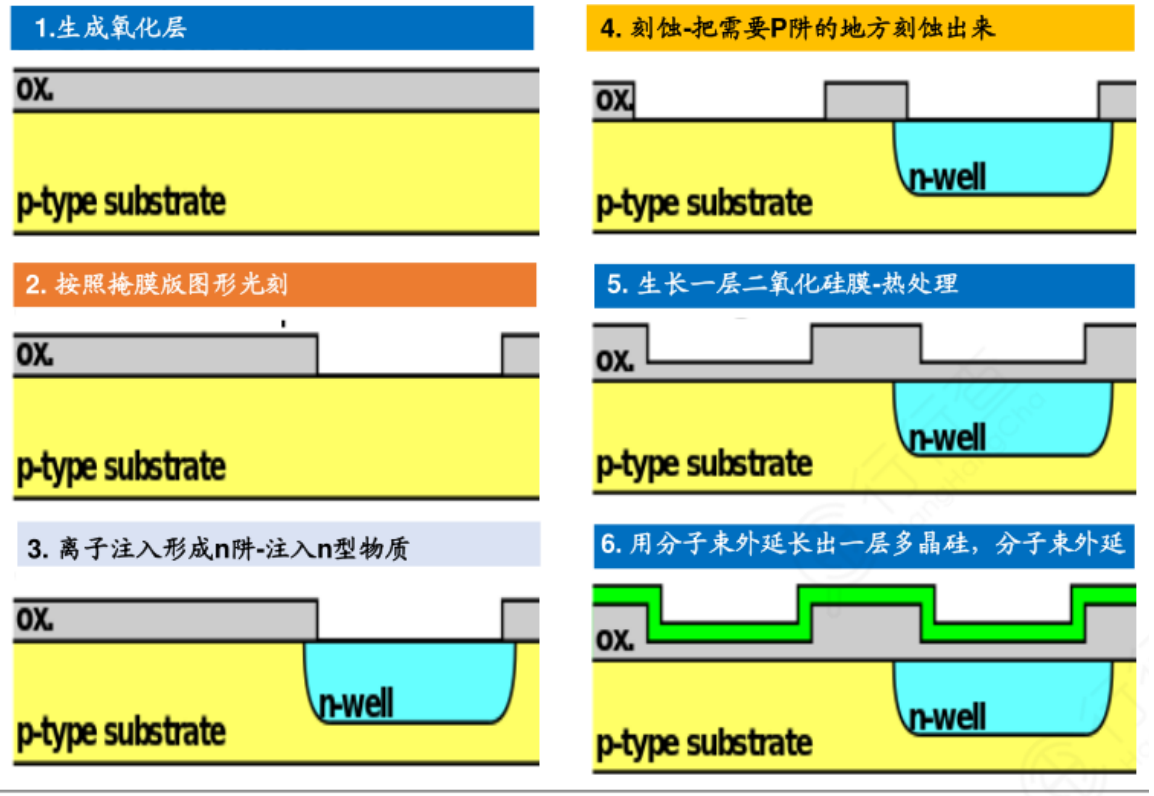
- MOSFET的制造过程总结



1、制造N型半导体区域

MOSFET晶体 – 现代芯片的基石

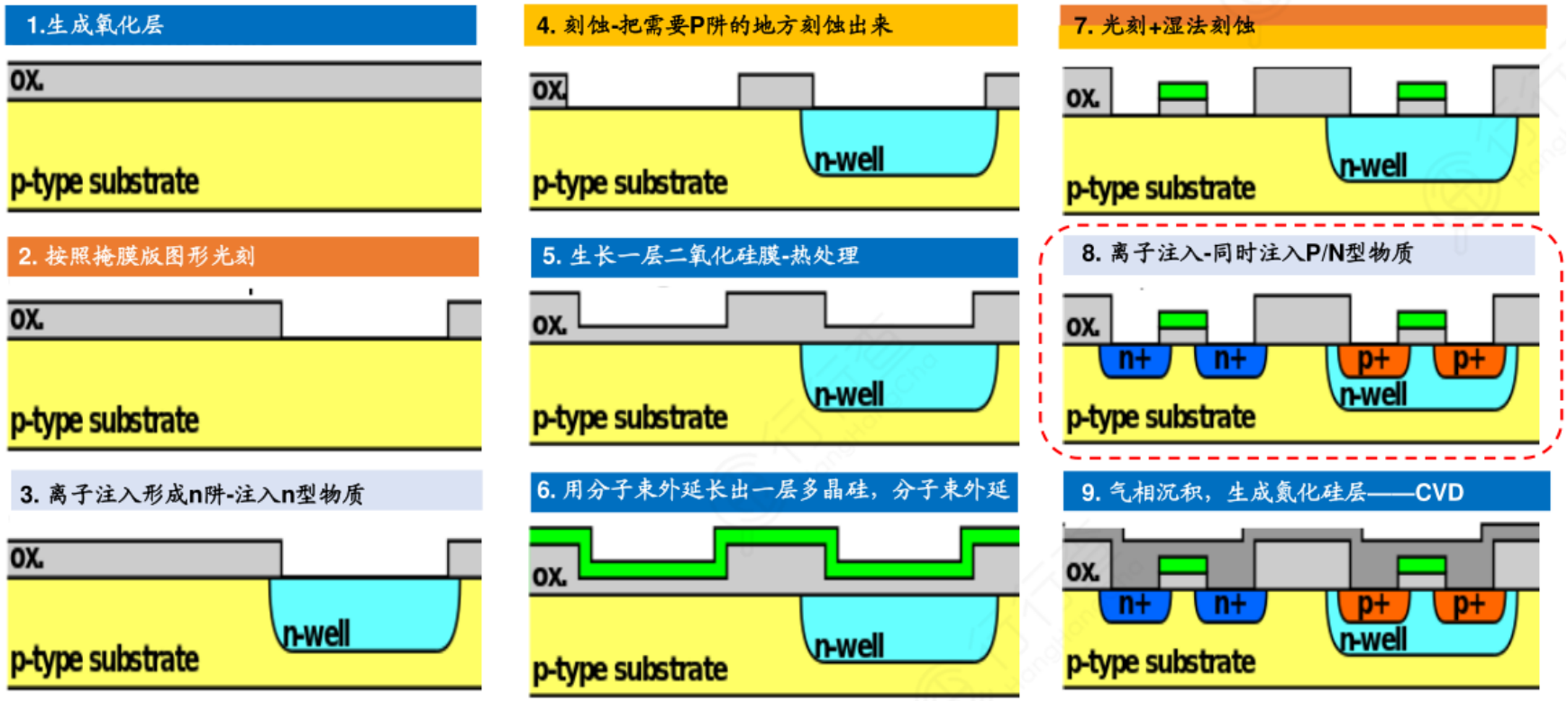
- MOSFET的制造过程总结



1、制造N型半导体区域　　2、形成晶体管基本结构

MOSFET晶体 – 现代芯片的基石

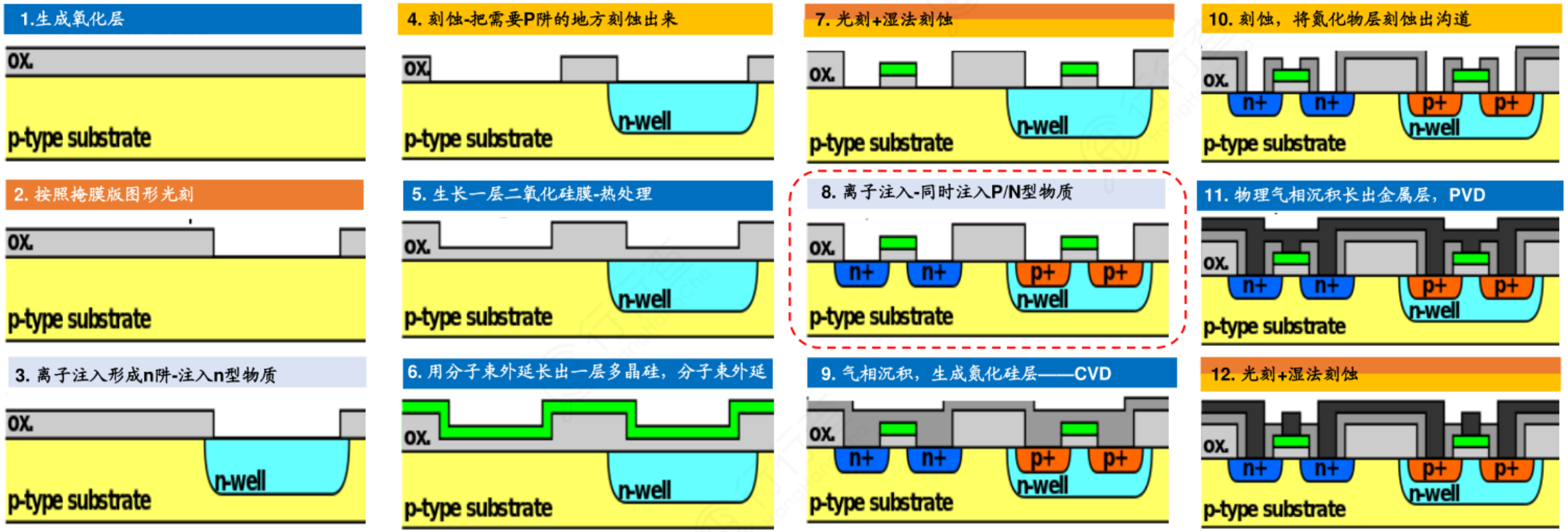
- MOSFET的制造过程总结



1、制造N型半导体区域　　2、形成晶体管基本结构　　3、形成晶体管完整结构

MOSFET晶体 – 现代芯片的基石

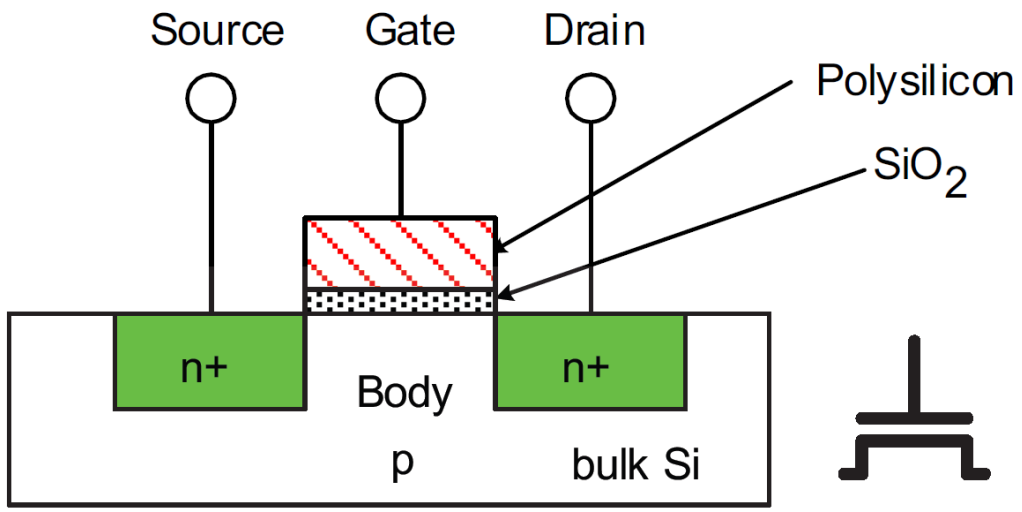
- MOSFET的制造过程总结



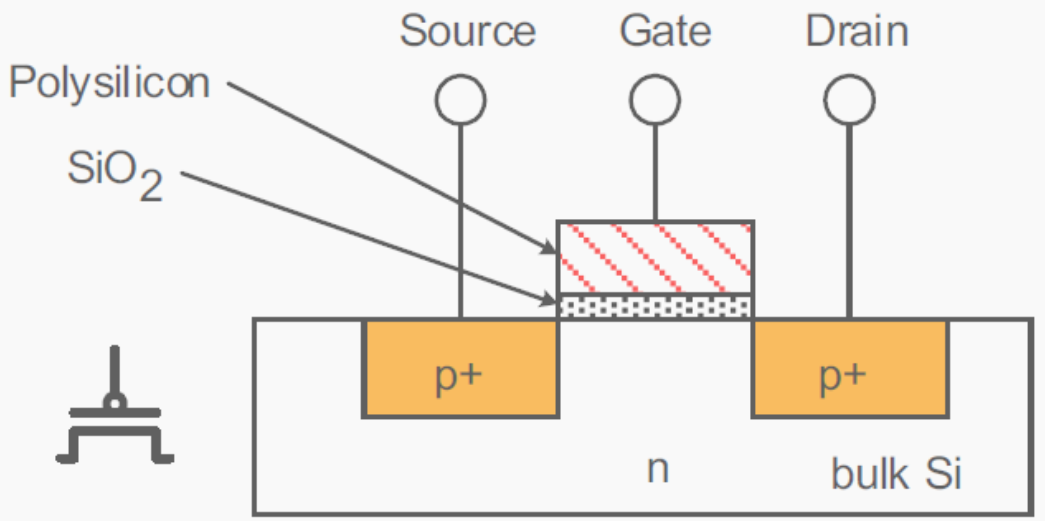
1、制造N型半导体区域　　2、形成晶体管基本结构　　3、形成晶体管完整结构　　4、电极生长与连线

MOSFET晶体 – 现代芯片的基石

- NMOS与PMOS



NMOS



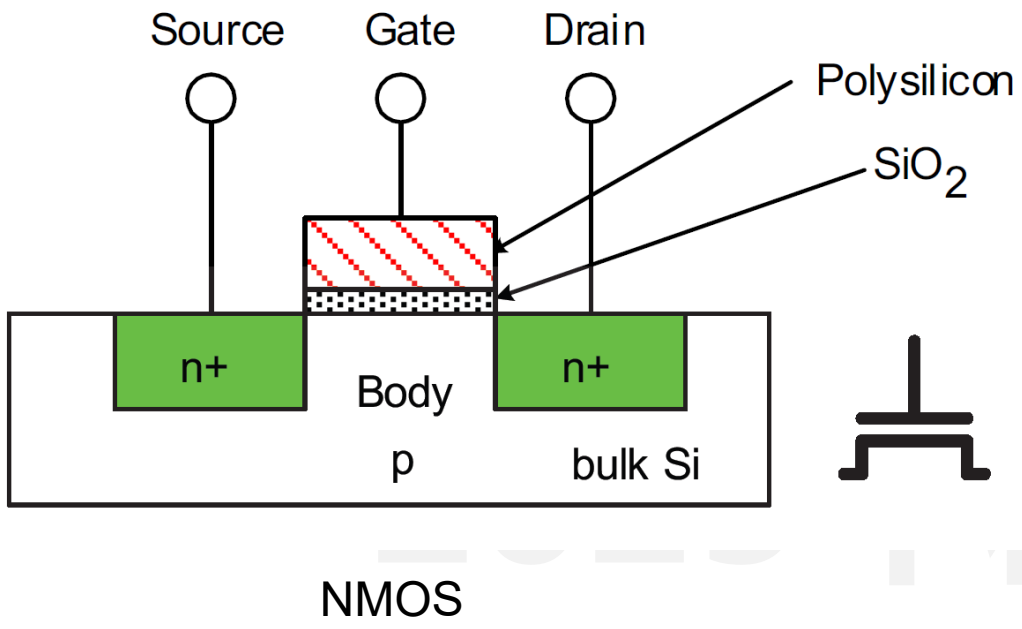
PMOS

当栅极处于低电压：

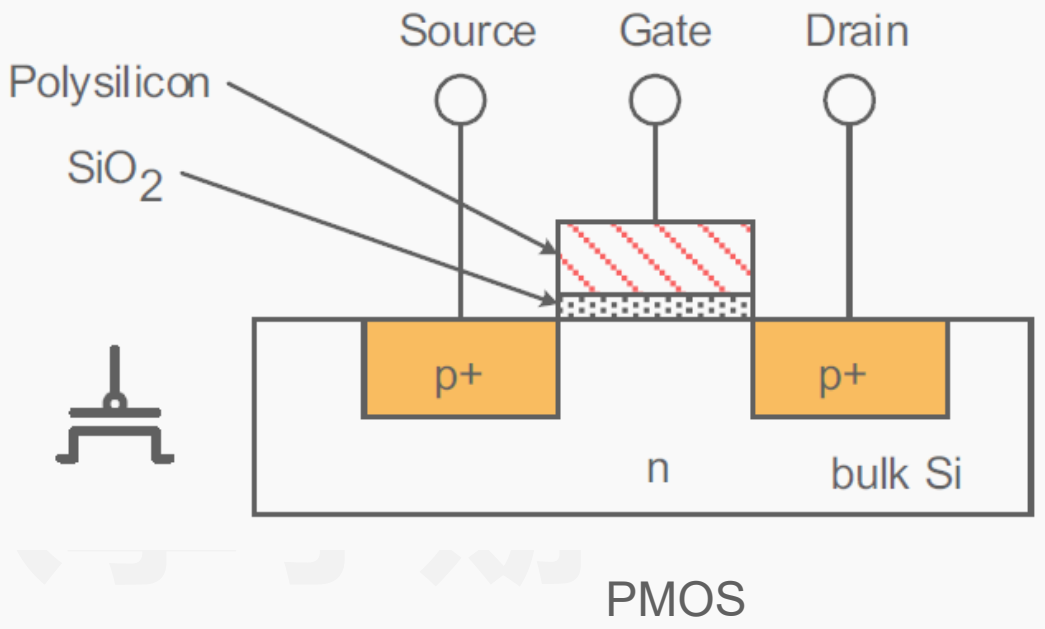
§ 体端（body）接地，即低电压
§ 栅极（gate）下通道中不存在导电通路
§ 无电流流动，晶体管关闭

MOSFET晶体 – 现代芯片的基石

- NMOS与PMOS



NMOS



PMOS

当栅极处于高电压：

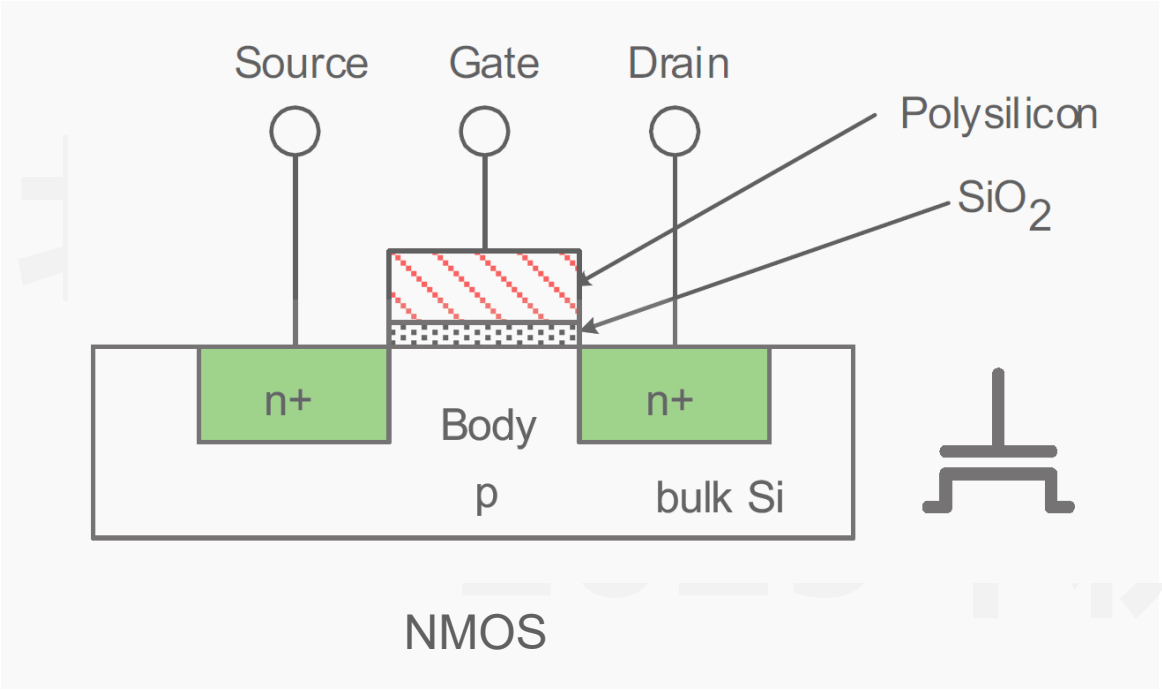
§MOS 电容栅极上有正电荷，体端则有负电荷

§将栅极下的通道反转为 n 型

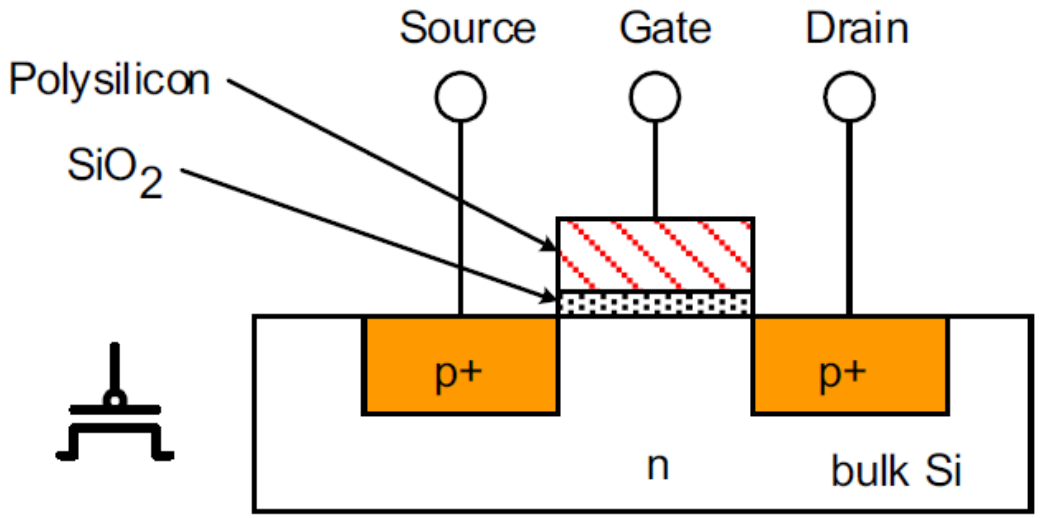
§现在电流通过通道流过n型硅，晶体管导通

MOSFET晶体 – 现代芯片的基石

- NMOS与PMOS



NMOS



PMOS

类似NMOS，但掺杂和电压相反

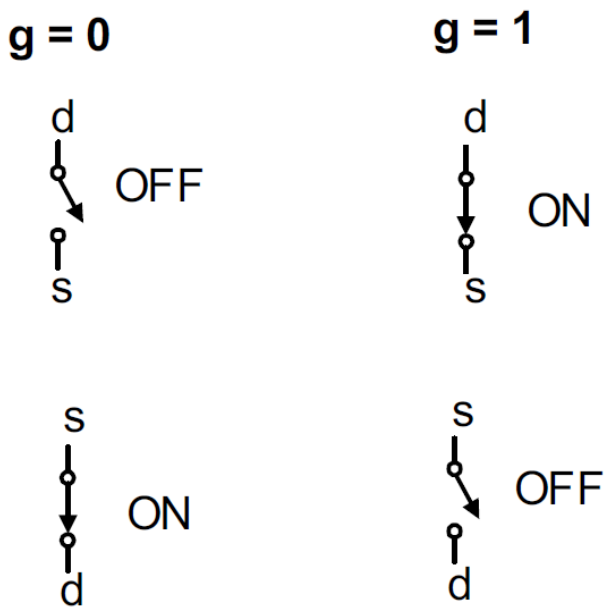
§体端与高电平相连（VDD）

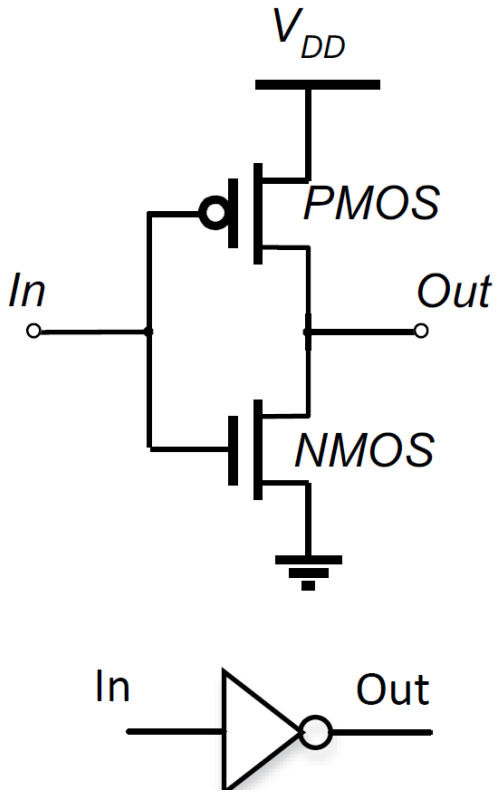
§**栅极低电平**：晶体管导通

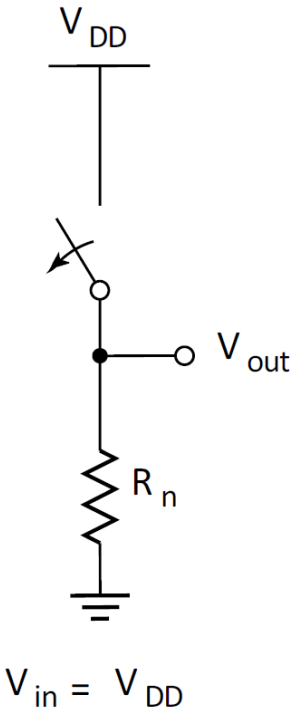
§**栅极高电平**：晶体管关闭

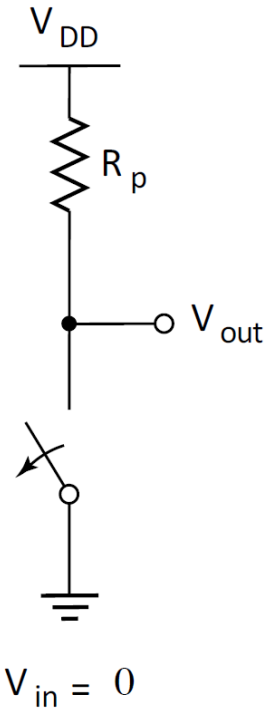
MOSFET作为开关的行为级模型

- NMOS、PMOS和简单反相器



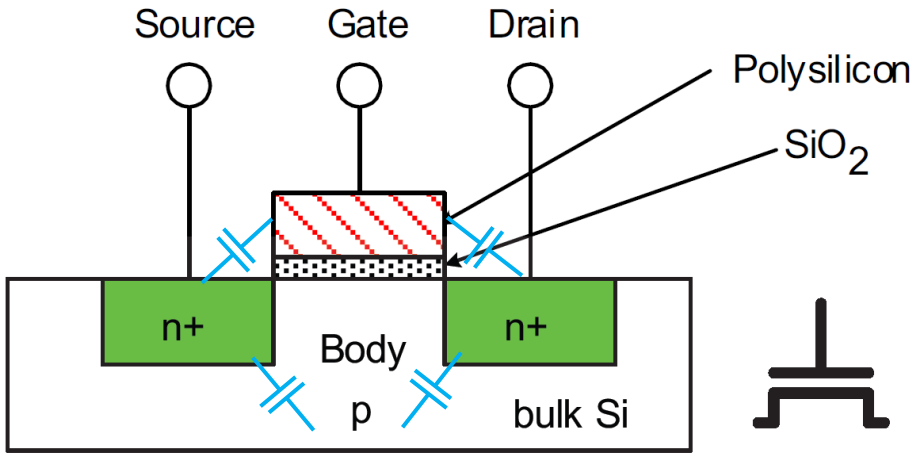




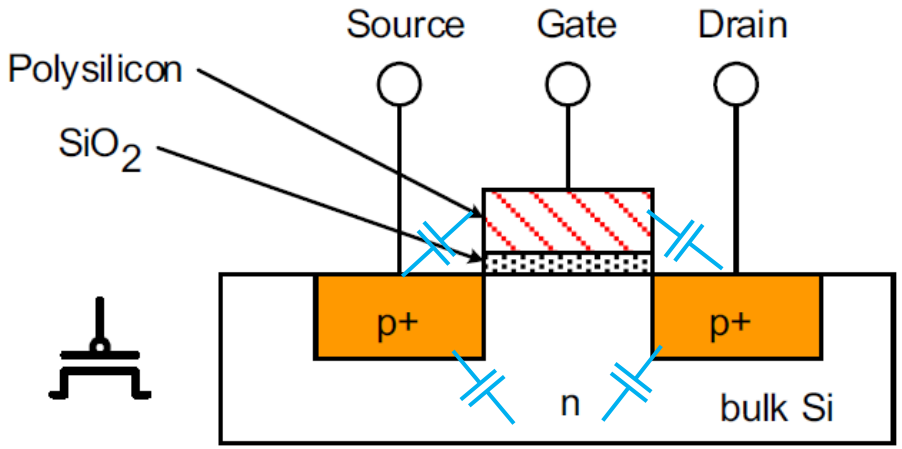


MOSFET晶体 – 现代芯片的基石

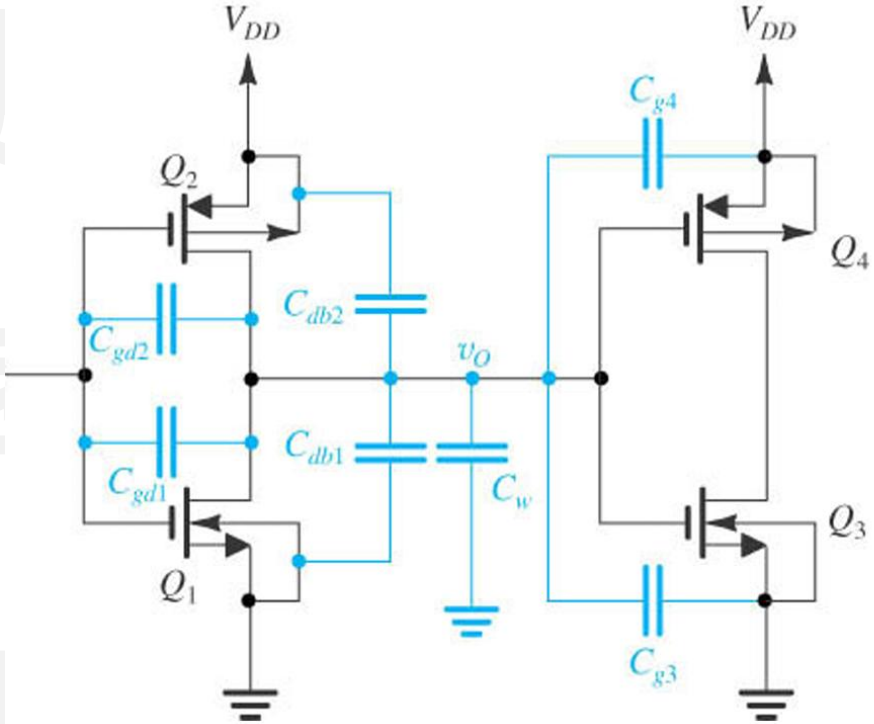
- NMOS与PMOS的寄生电容



NMOS

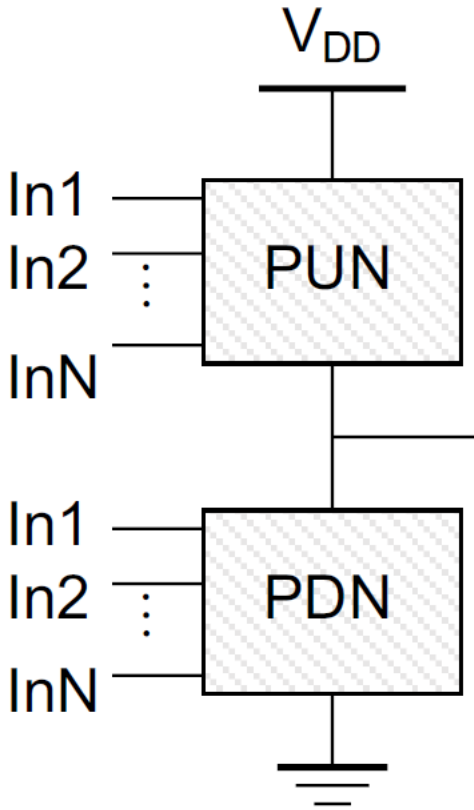


PMOS



经典静态逻辑电路

- NMOS、PMOS可以组成PDN和PUN



PMOS only

上拉逻辑设计：

Pull-up network: make a connection from V_{dd} to F when F(In1,In2…)=1

F(In1,In2,…InN)

NMOS only

Pull-down network: make a connection from Ground to F when F(In1,In2…)=0

下拉逻辑设计：

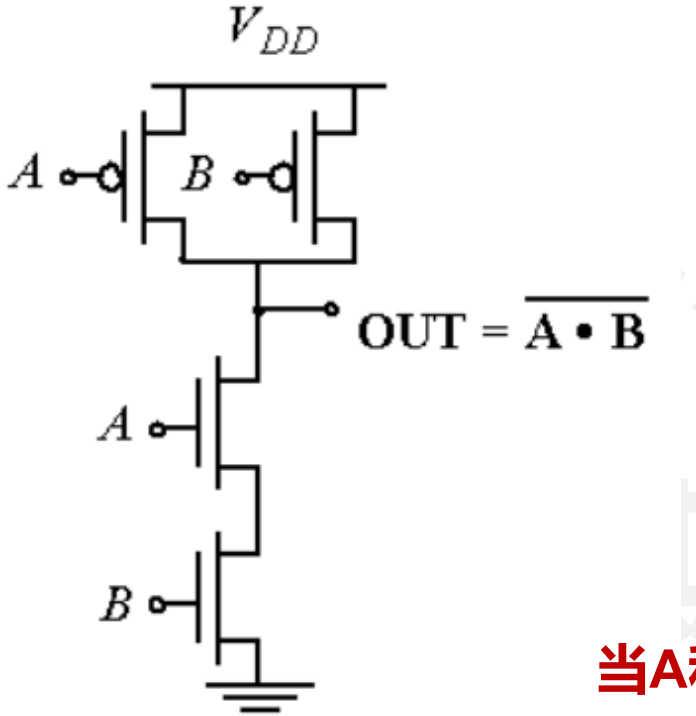
PUN and PDN are *dual* networks

经典静态逻辑电路

- 由PDN和PUN组成的NAND门电路

A	B	Out
0	0	1
0	1	1
1	0	1
1	1	0

Truth Table of a 2 input NAND gate



$OUT = \overline{A \bullet B}$ NAND门

PDN: Connects OUT to ground when A●B=1

PUN: Connects OUT to V_{dd} when $\overline{A} + \overline{B} = 1$

So OUT = Complement of PDN function

Also OUT = PUN function with each input inverted

当A和B同为1时，输出为0

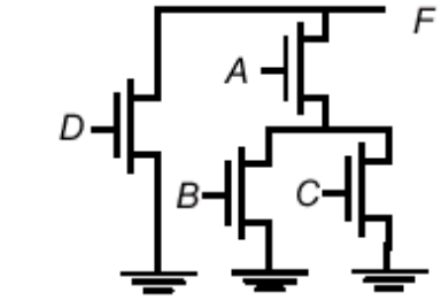
当A和B有0时，输出为1

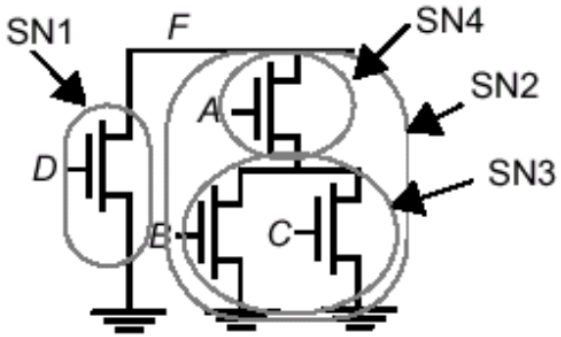
经典静态逻辑电路

- 由PDN和PUN组成的复杂静态逻辑电路

$$F = \overline{D + (A\,(B + C))}$$

什么情况下F为0 -> D+A(B+C) = 1





忽略取非操作，直接构建PDN

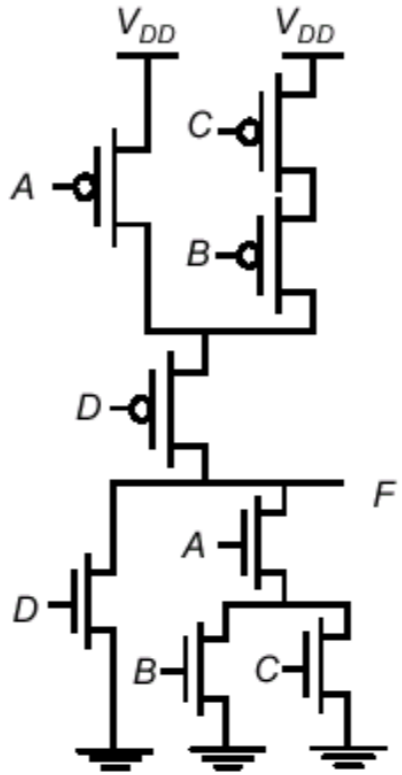
Ex：D+X意味着D与X并联

构建PDN的亚网络

在SN3内，B和C是并联的，

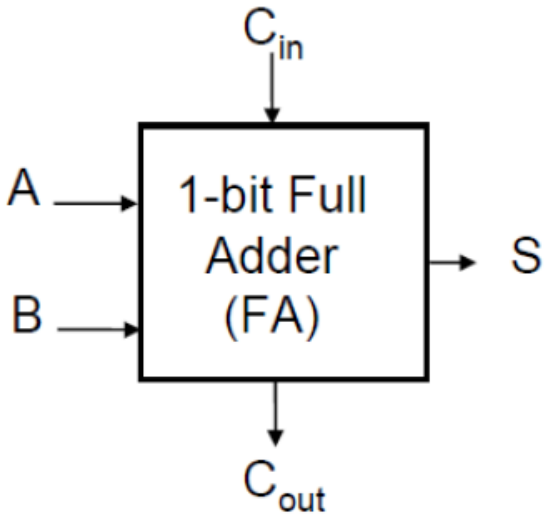
则在PUN中，他们串联

自下向上构建



经典静态逻辑电路

- 简单1bit加法器电路
  - A[n-1:0] + B[n-1:0] = S[n-1:0]



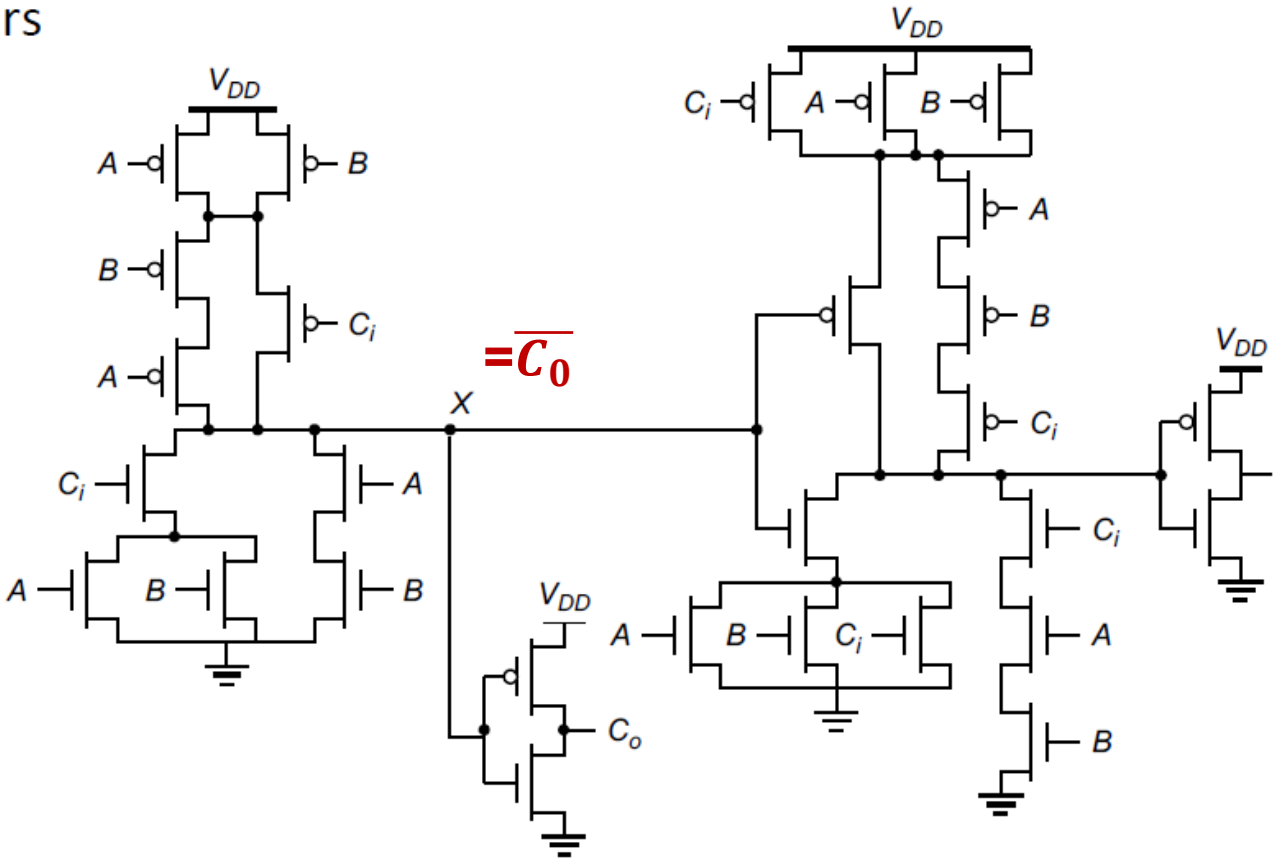
A	B	C_{in}	C_{out}	S	carry status
0	0	0	0	0	kill
0	0	1	0	1	kill
0	1	0	0	1	propagate
0	1	1	1	0	propagate
1	0	0	0	1	propagate
1	0	1	1	0	propagate
1	1	0	1	0	generate
1	1	1	1	1	generate

单比特加法器逻辑设计

- $C_o = AB + BC_i + AC_i = AB + (A + B)C_i$
- $S = A \oplus B \oplus C_i = ABC_i + \overline{C_o}(A + B + C_i)$
- $G = AB,\ K = \overline{A}\,\overline{B},\ P = A \oplus B$

经典静态逻辑电路

- 简单1bit加法器电路
  - $C_o = AB + BC_i + AC_i = AB + (A+B)C_i$
  - $S = A \oplus B \oplus C_i = ABC_i + \overline{C_o}(A+B+C_i)$
  - 28 transistors

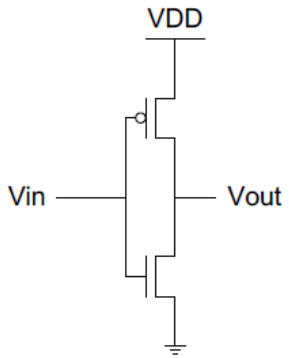


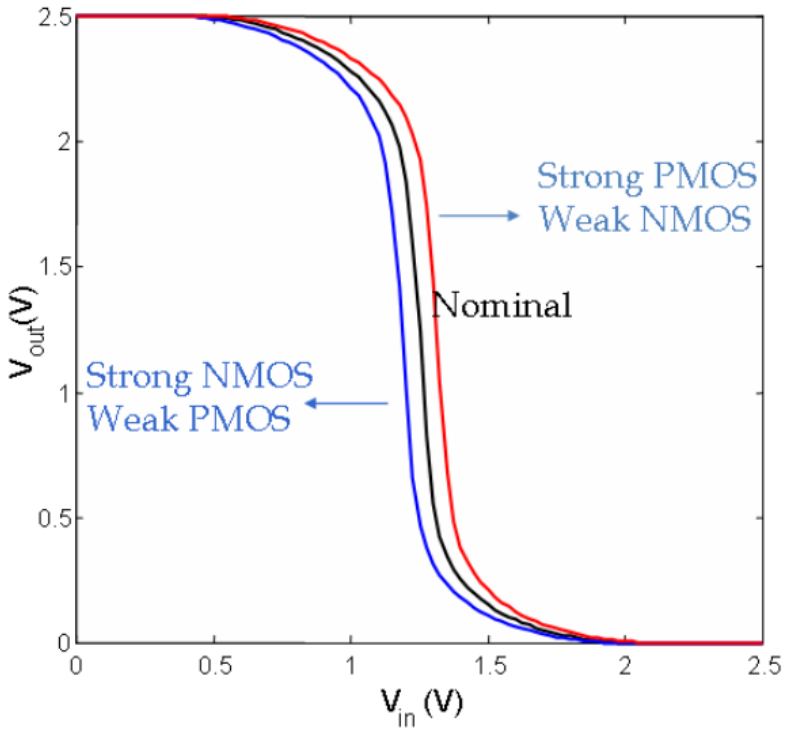
目 录

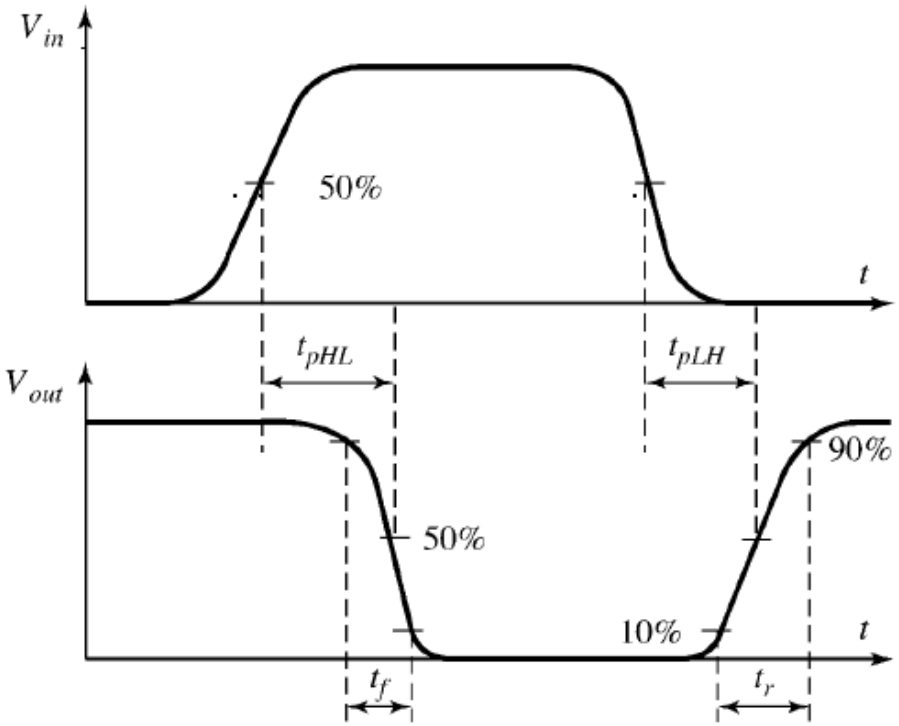
CONTENTS

什么是电路的延迟

- 以Inverter反相器为例



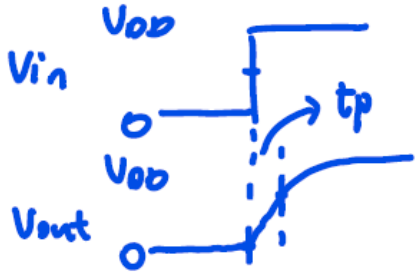


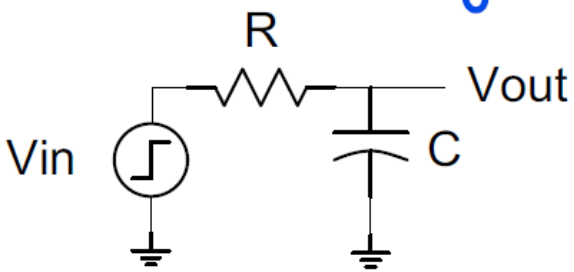


什么是电路的延迟

- 一阶RC延迟分析

A First-Order RC Network





$$V_{out}(t) = (1 - e^{-t/\tau})V_{DD} = \frac{1}{2}V_{DD}$$

$$e^{-t/\tau} = \frac{1}{2}$$

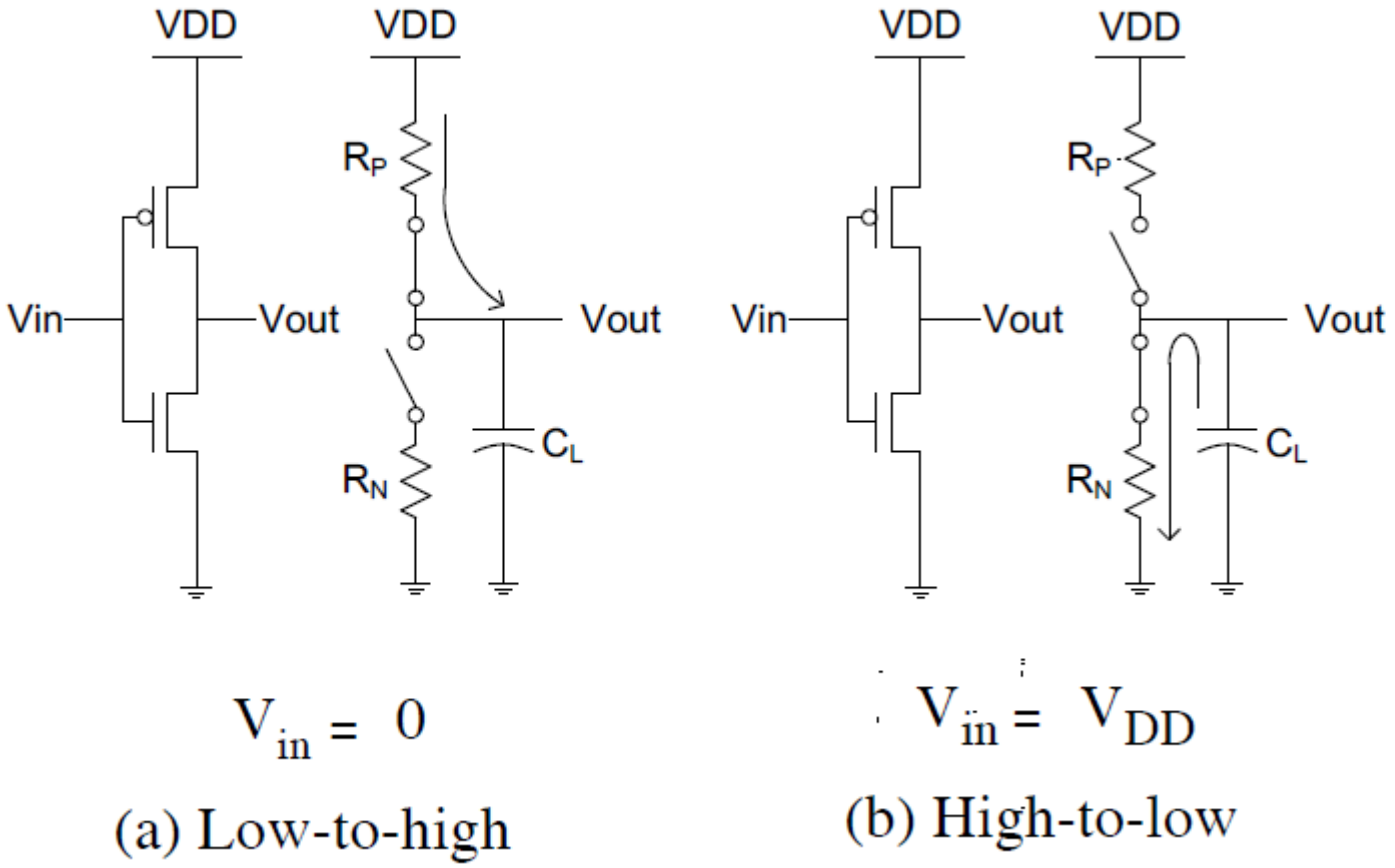
$$-t/\tau = -\ln 2$$

$$\tau = R \times C$$

$$t_p = \ln(2)\tau = 0.69R \times C \qquad t = \ln(2)\tau$$

反相器延迟

- 利用一阶RC延迟分析方法



$V_{in} = 0$

(a) Low-to-high

$V_{in} = V_{DD}$

(b) High-to-low

$$t_{pHL} = f(R_N \times C_L)$$

$$t_{pHL} = 0.69 R_N \times C_L$$

$$t_{pLH} = 0.69 R_P \times C_L$$

降低延迟的设计方法

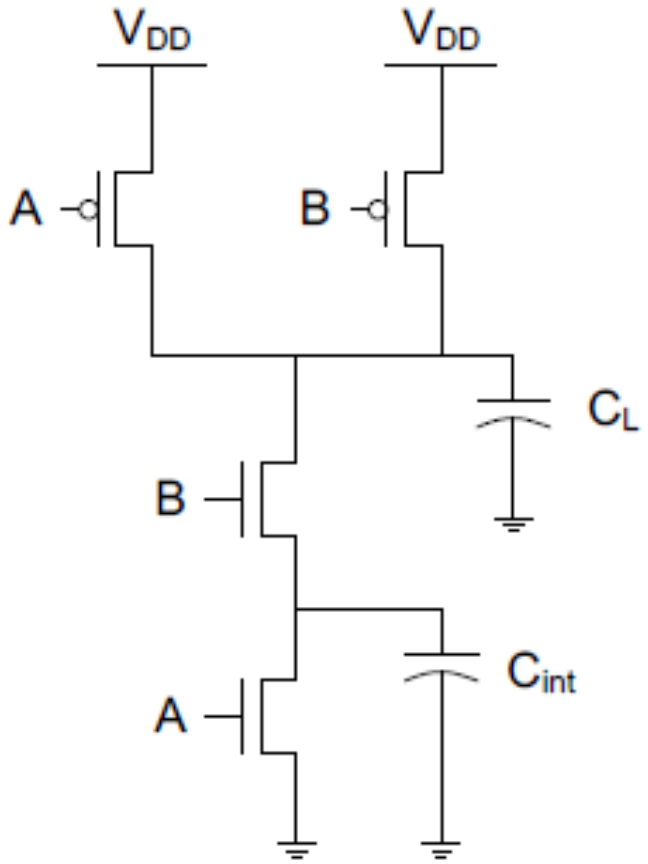
- 利用一阶RC延迟分析方法

$$t_{pHL} = f(R_N \times C_L)$$
$$t_{pHL} = 0.69 R_N \times C_L$$
$$t_{pLH} = 0.69 R_P \times C_L$$

- **利用较小的电容 – 降低C**
  - 版图紧凑, 布局合理
  - 保持较短走线&减少diffusion routing
- **增加晶体管尺寸 – 降低 R**
  - 避免self-loading出现，否则会导致寄生电容增大
- **增加电源电压**
  - 同时会影响可靠性与功耗，因而一般不采用

输入Pattern对延迟的影响

- 利用一阶RC延迟分析方法

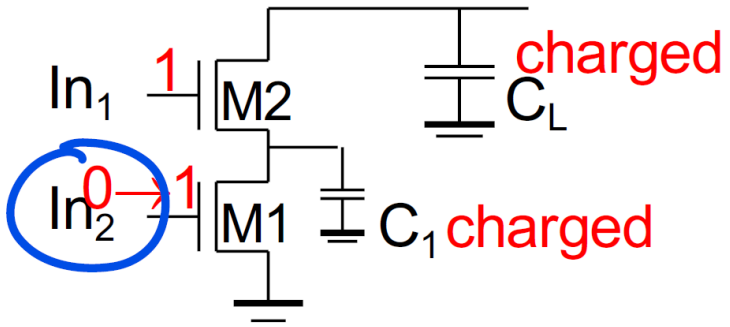


电路延迟与输入的顺序有关！

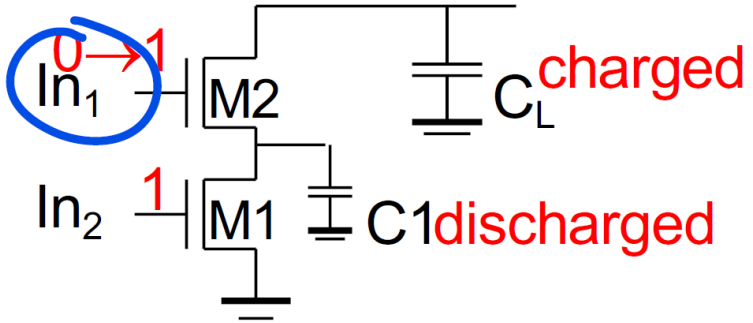
- *Ignore C_{int} for the moment!*
- Low to high transition
  - both inputs go low
    - delay is 0.69 $R_p/2$ C_L
  - one input goes low
    - delay is 0.69 R_p C_L
- High to low transition
  - both inputs go high
    - delay is 0.69 $2R_n$ C_L

利用Transistor Ordering提升逻辑速度

- 复杂的Transistor Ordering需要仿真工具支持



延迟由C_L与C_1放电时间决定



延迟由C_L放电时间决定

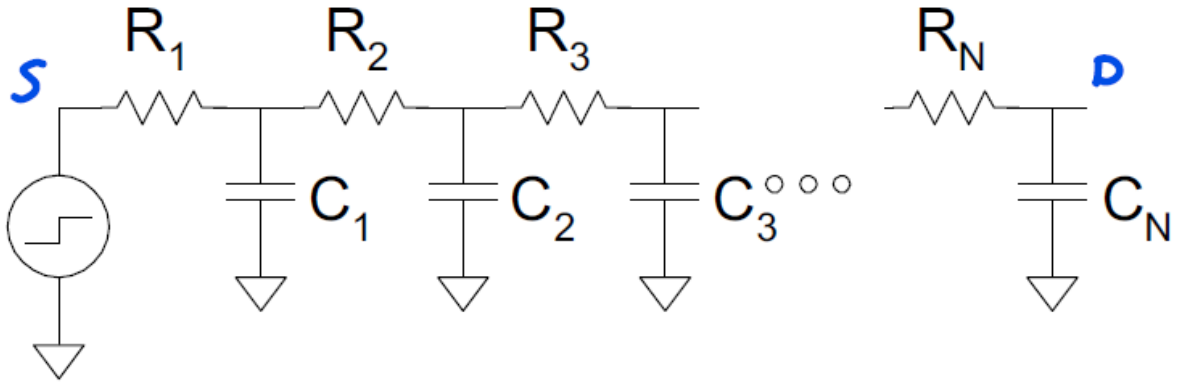
逻辑电路的Elmore Delay模型

- 拓展多级的RC模型

  - 导通晶体管看作电阻
  - 电路网络建模为RC阶梯
  - RC阶梯的Elmore延迟
  - Apply to complex gates（i.e.,stacks),also interconnect（later）

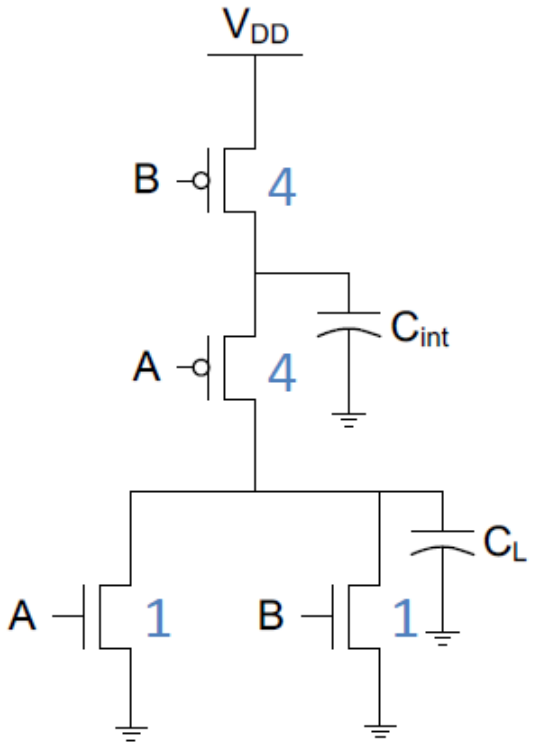
$$t_{pd} \approx \sum_{\text{nodes } i} R_{i-to-source} C_i$$

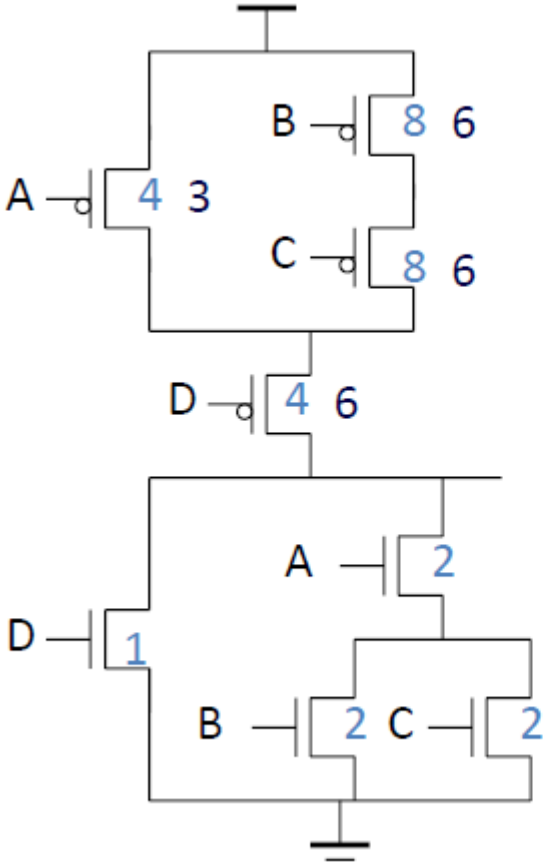
$$= R_1 C_1 + (R_1 + R_2) C_2 + \ldots + (R_1 + R_2 + \ldots + R_N) C_N$$



逻辑电路的Transistor Sizing

- 目标是将PDN和PUN的延迟进行匹配



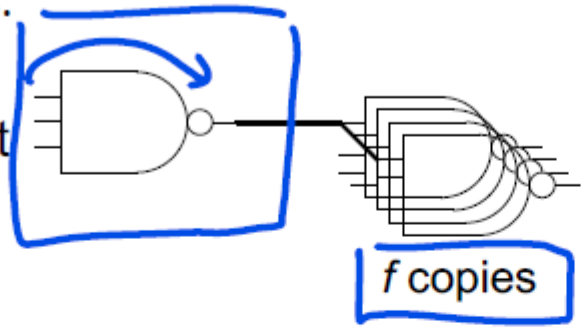


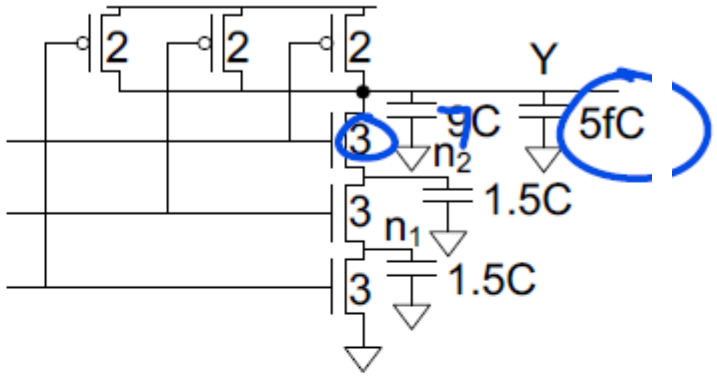
$$\overline{OUT = D + A \bullet (B + C)}$$

逻辑电路的Elmore Delay模型

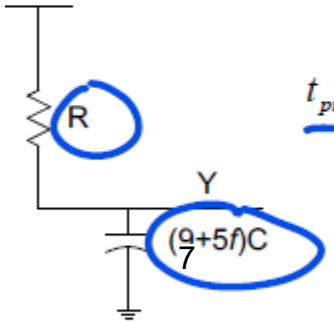
- 拓展多级的RC模型 – 3-input NAND gates

  - Estimate worst-case rising and falling delay of 3-input NAND driving *f* identical gates.
  - R=13KΩ/unit
  - C=0.21fF/unit

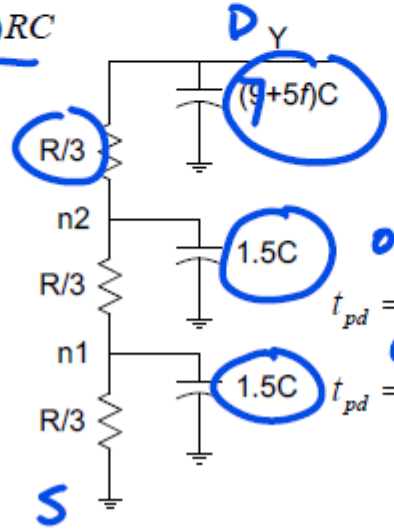




Worst-case pull up



$t_{pu} = (9+5f)RC$



Worst-case pull down

$t_{pd} = 1.5C\frac{R}{3} + 1.5C\left(\frac{R}{3}+\frac{R}{3}\right) + [(9+5f)C]\left(\frac{R}{3}+\frac{R}{3}+\frac{R}{3}\right)$

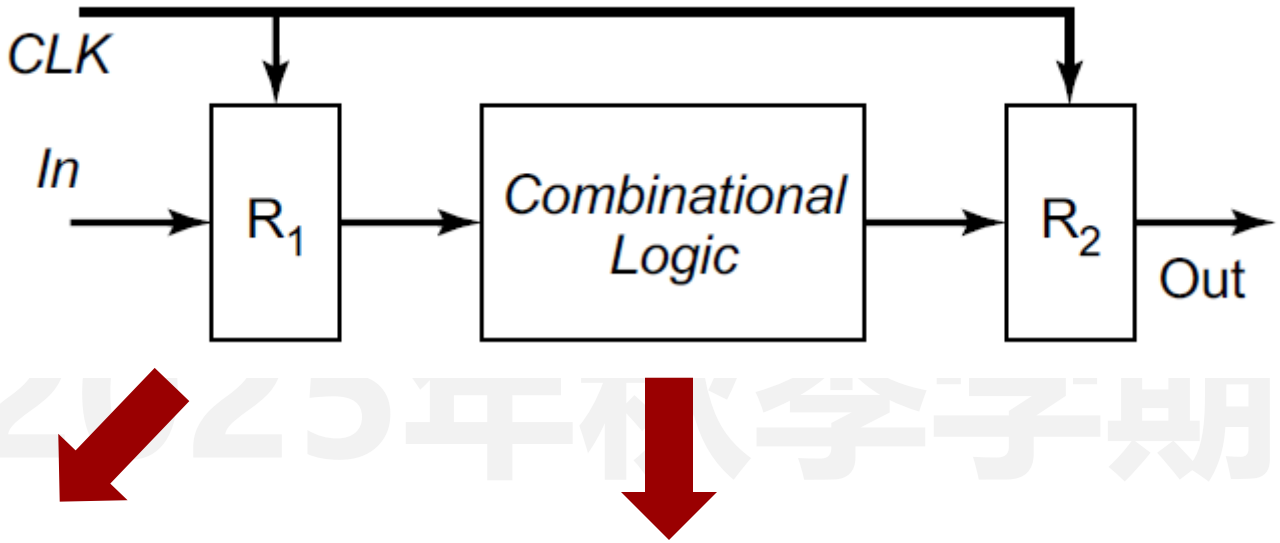
$t_{pd} = (10.5+5f)RC$

目 录

CONTENTS

电路时序的基本概念

- 同步时序（Synchronous Timing）

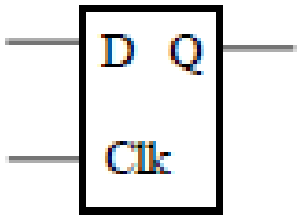


触发器
（Register）

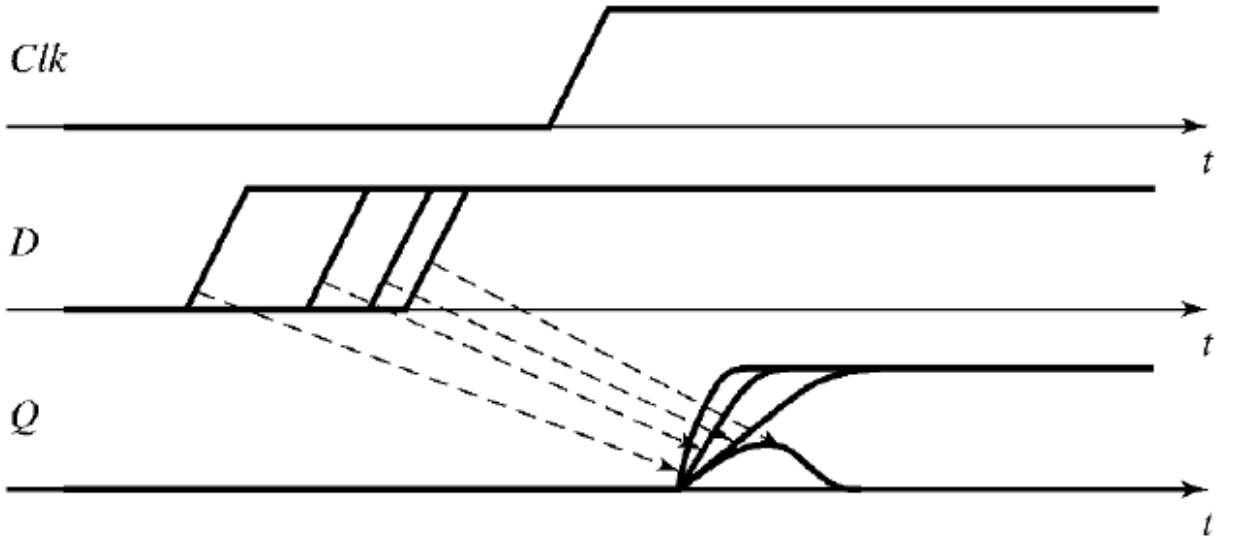
组合逻辑（各种逻辑门电路）

电路时序的基本概念

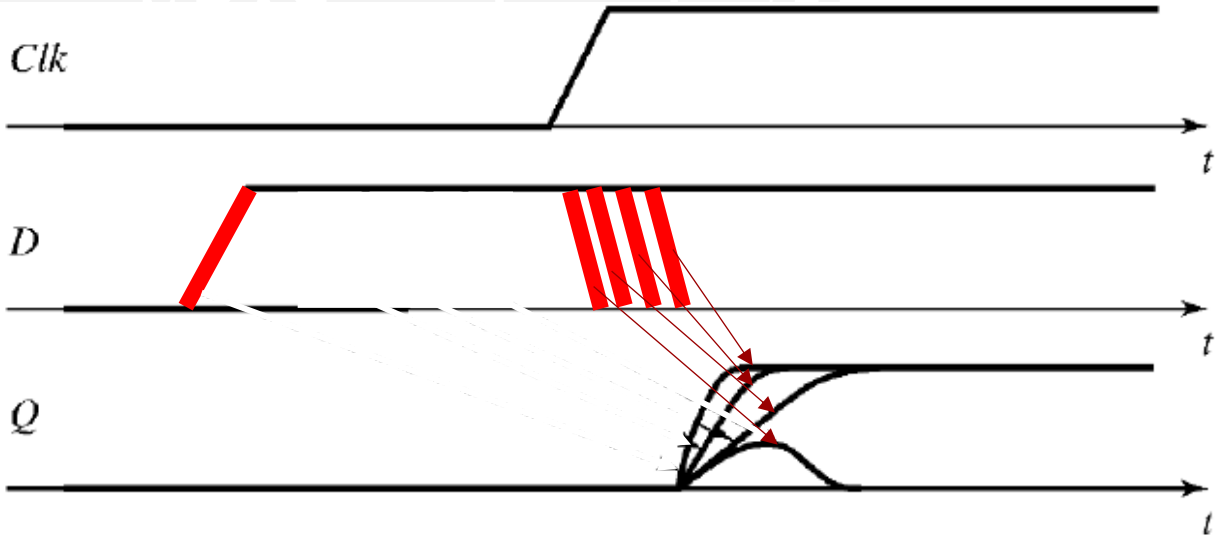
- 触发器的Setup Time和Hold Time



Setup Time

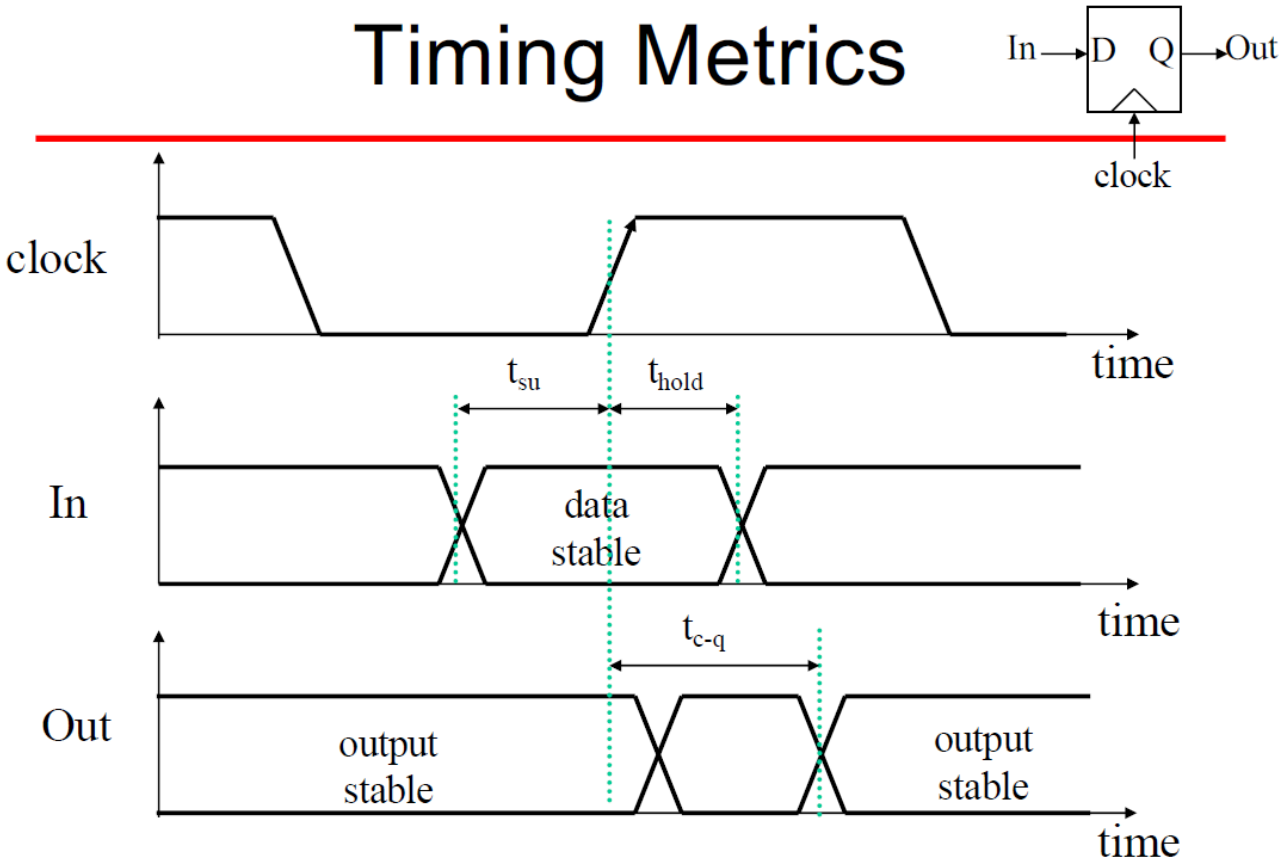


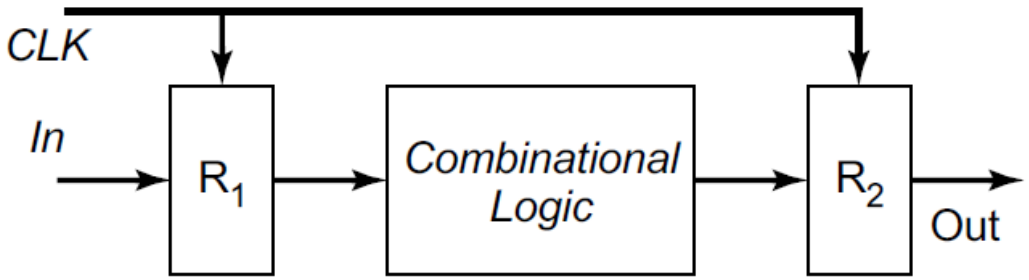
Hold Time



电路时序的基本概念

- 同步时序（Synchronous Timing）





$$T_{c\text{-}q} + t_{plogic,min} \geq t_{hold}$$

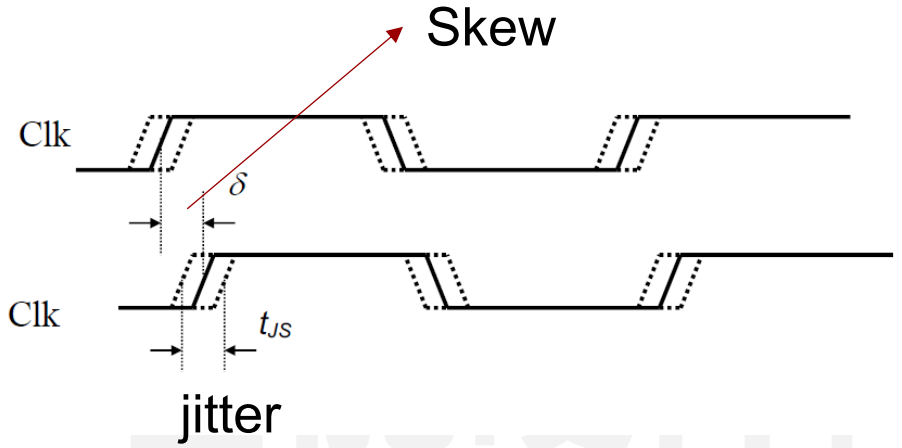
min

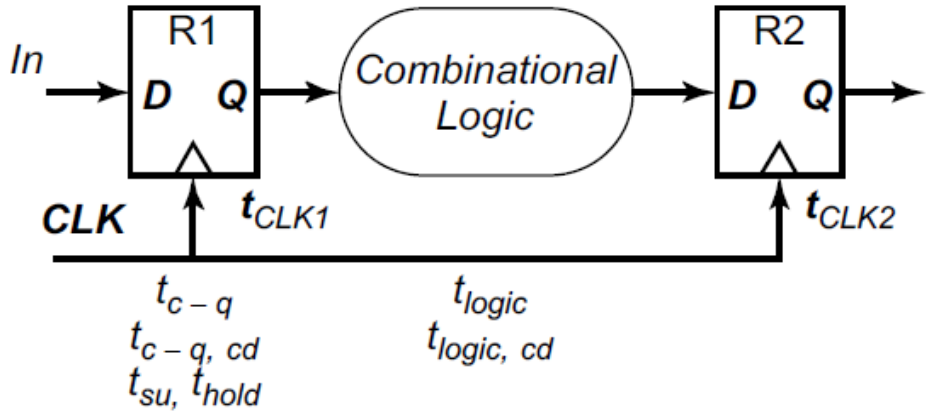
$$T \geq t_{c\text{-}q} + t_{plogic,max} + t_{su}$$

max

电路时序的基本概念

- 时钟的不稳定性

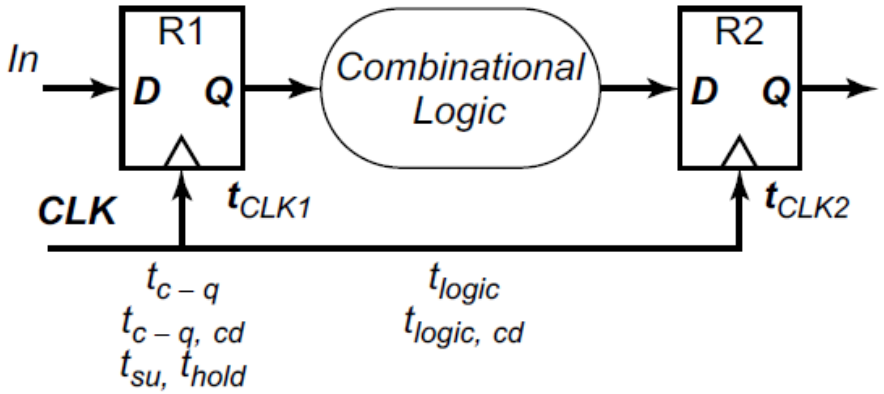




Minimum cycle time:

$T \geq t_{c\text{-}q} + t_{su} + t_{logic} - \delta$

最坏情况为接收边沿过早到达（negative δ）



Hold time constraint:

$t_{(c\text{-}q,\ cd)} + t_{(logic,\ cd)} > t_{hold} + \delta$

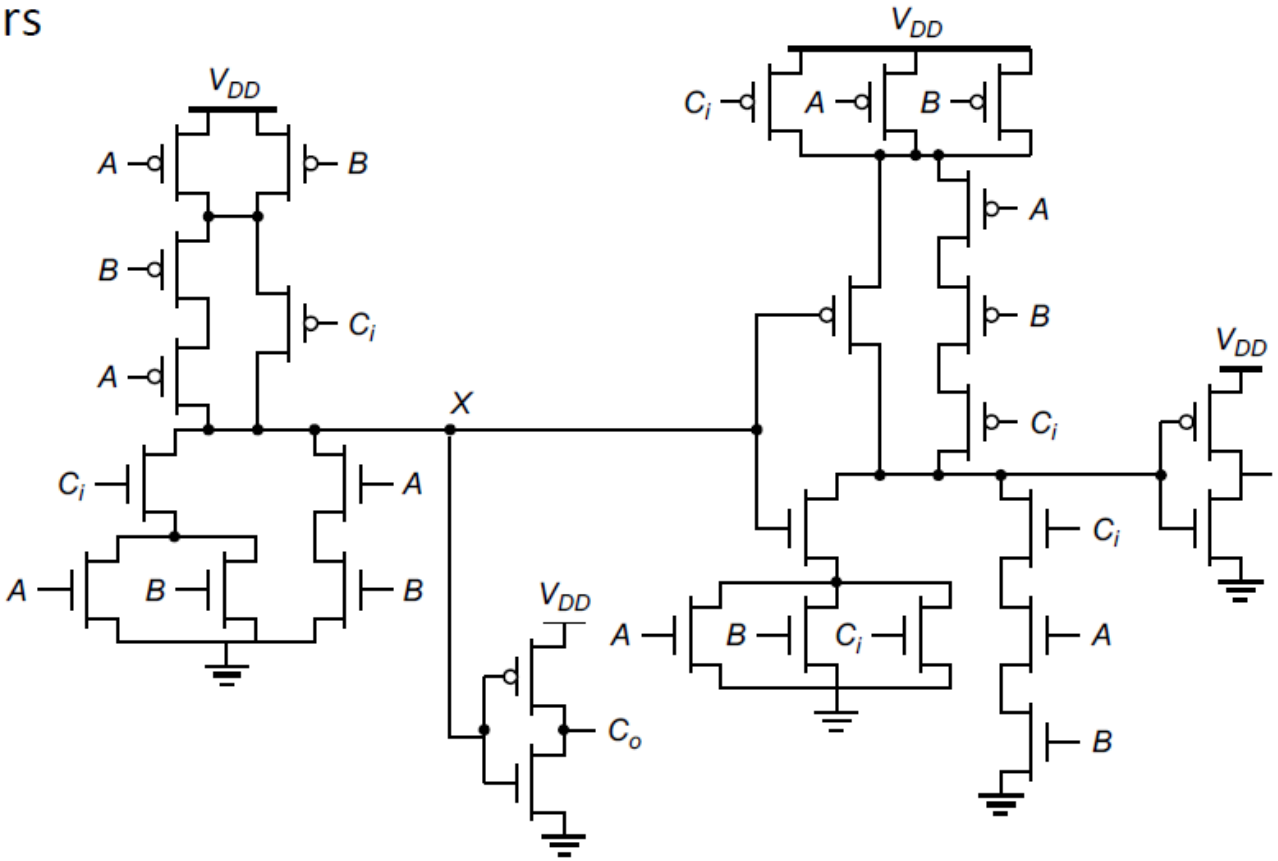
最坏情况为接收边沿过晚到达（正偏差）
数据和时钟之间的竞争
Cd：contamination delay（最快可能延迟）

目 录

CONTENTS

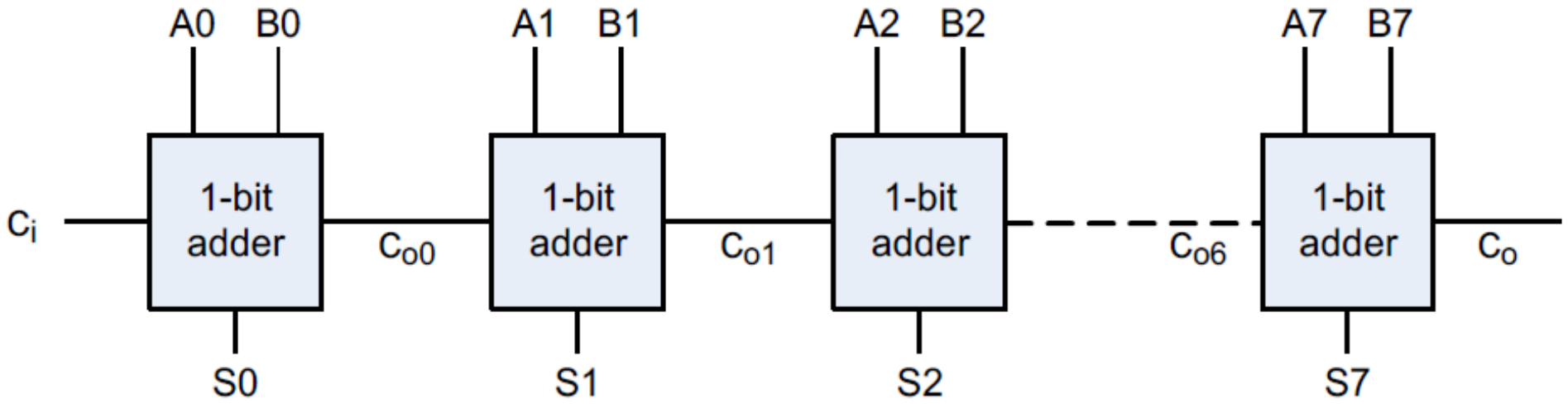
加法器设计

- 简单1bit加法器电路
  - $C_o = AB + BC_i + AC_i = AB + (A + B)C_i$
  - $S = A \oplus B \oplus C_i = ABC_i + \overline{C_o}(A + B + C_i)$
  - 28 transistors



加法器设计

- Ripple Carry加法器电路



最差延迟与比特数呈线性关系

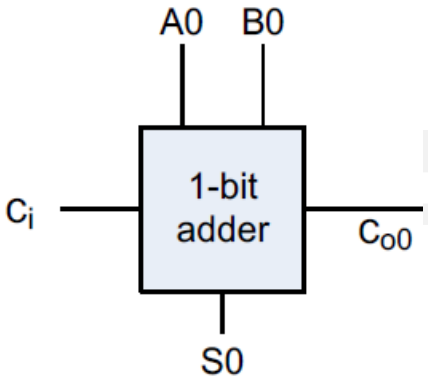
$$t_d = O(N)$$

$$t_{adder} = (N\text{-}1)t_{carry} + t_{sum}$$

Goal: Make the fastest possible carry path circuit

加法器设计

- 基于PGK的加法器设计方法



Generate (G) = AB

Propagate (P) = A ⊕ B

- Generate: C_{out} = 1 independent of C_{in}
  - G = A • B
- Propagate: C_{out} = C_{in}
  - P = A ⊕ B
- Kill: C_{out} = 0 independent of C_{in}
  - K = ~A • ~B

$$C_o(G,P) = G + PC_i$$

$$S(G,P) = P \oplus C_i$$

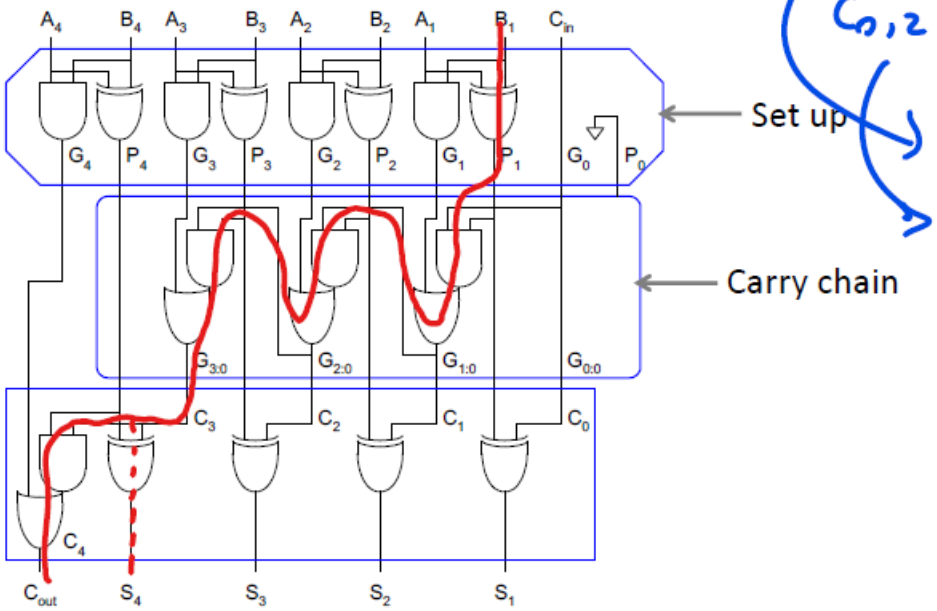
$P = A \oplus B$

$P = A + B$

- 基于PGK的加法器设计方法

Carry-Ripple using P and G

$$C_{i:0} = G_i + P_i \cdot C_{i-1:0}$$
$$G_{0:0} = C_{in}$$
$$P_{0:0} = 0$$
$$C_{out,i} = G_{i:0}$$



$C_{0,1} = G_1 + P_1 C_{in}$

$C_{0,2} = G_2 + P_2 C_{0,1}$

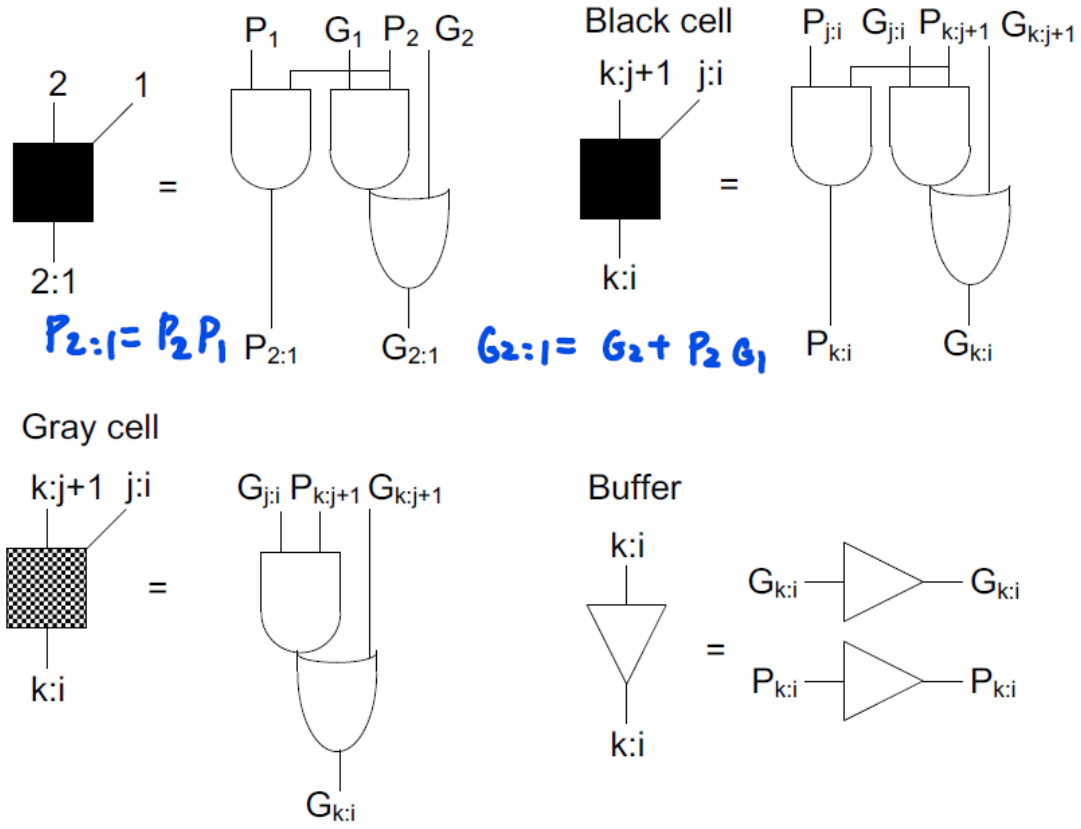
$G_{1:0} = G_1 + P_1 G_0$

$G_{2:0} = G_2 + P_2 G_{1:0}$

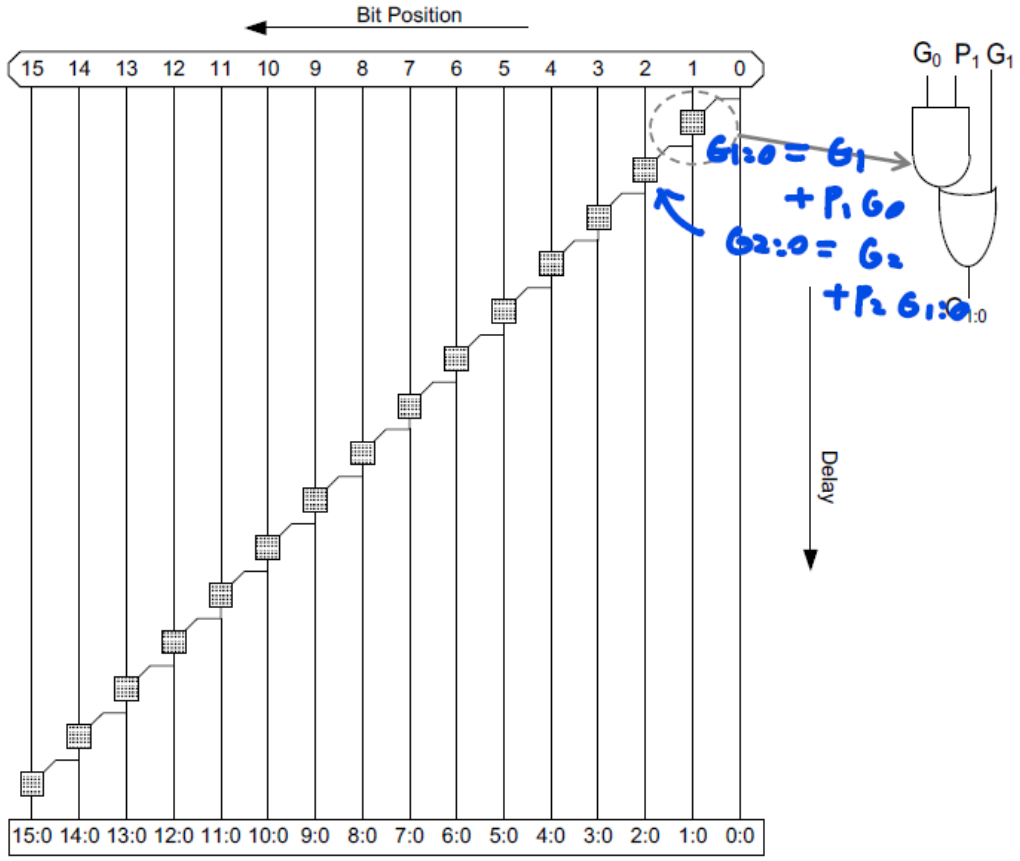
$t_{adder} = t_{setup} + (N-1)\, t_{carry} + \max(t_{carry}, t_{sum})$

加法器设计

- 基于PGK的加法器设计方法



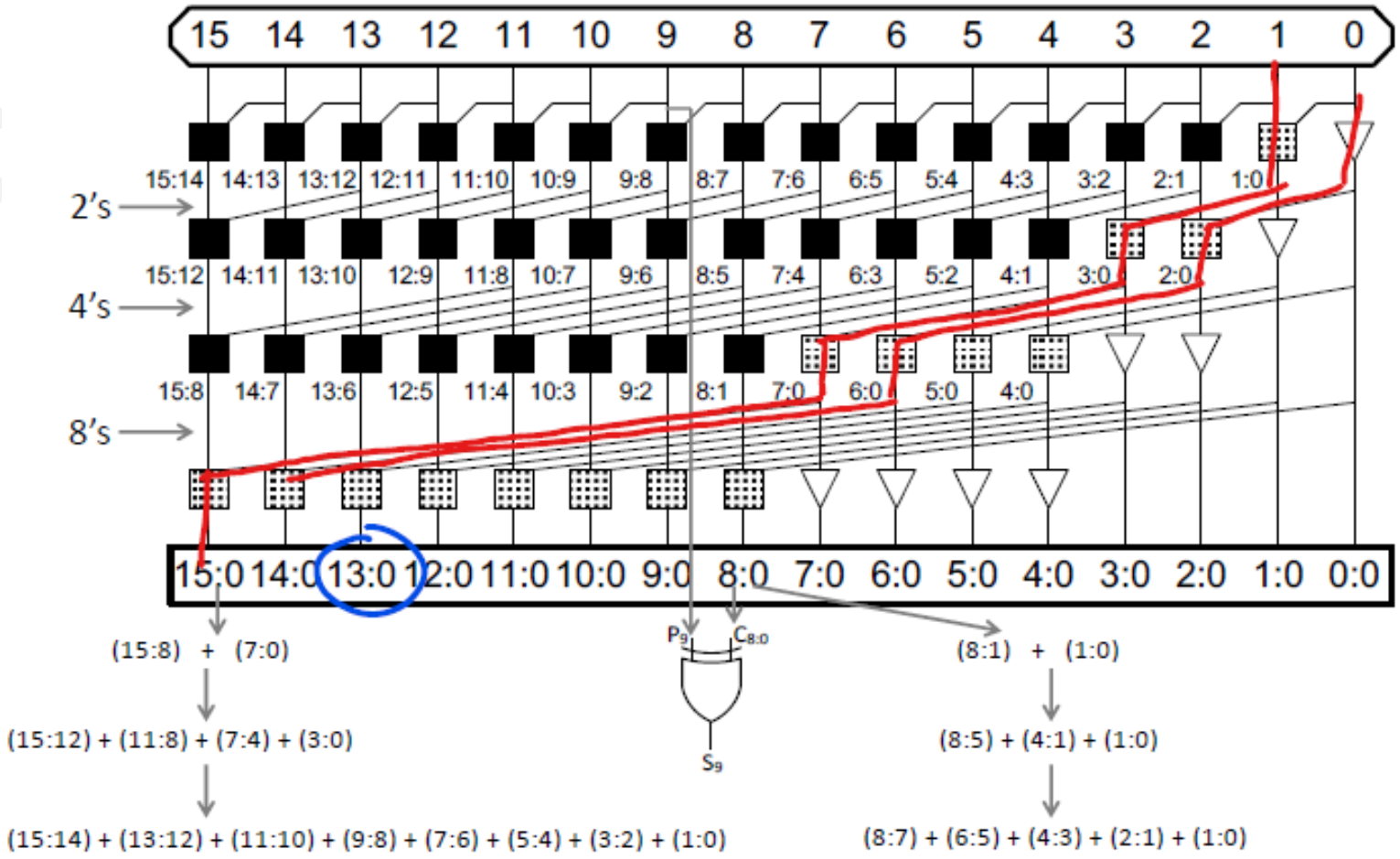
PG生成逻辑



Carry Ripple的PG图

下节课

- 基于PGK的加法器设计方法 – 复杂PG树加法器



$\log_2(N)$

$G_{15:0} = G_{13:6} + P_{13:6}\, G_{5:0}$

$G_{13:6} = G_{13:10} + P_{13:10}\, G_{9:6}$

$P_{13:6} = P_{13:10}\, P_{9:6}$

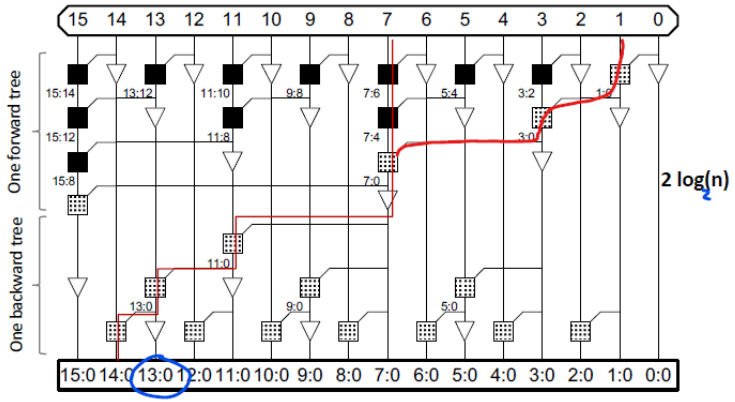
$G_{5:0} = G_{5:2} + P_{5:2}\, G_{1:0}$

作业题

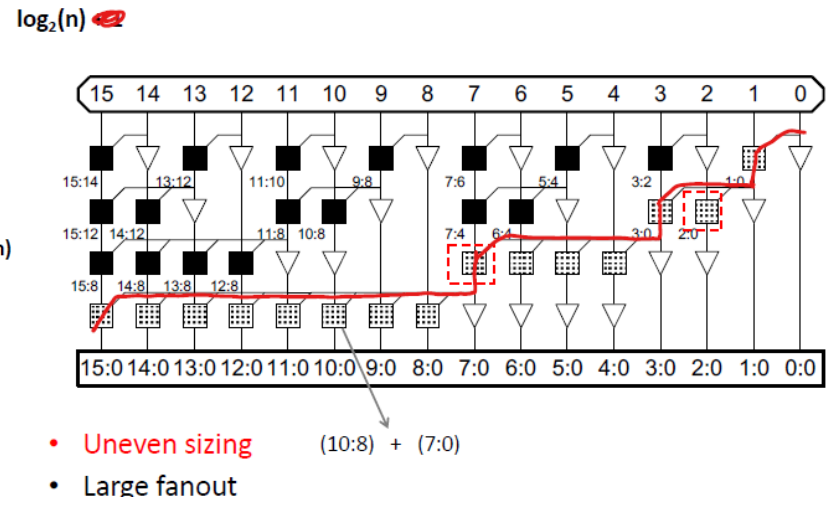
加法器设计

- 基于PGK的加法器设计方法 – 复杂PG树加法器

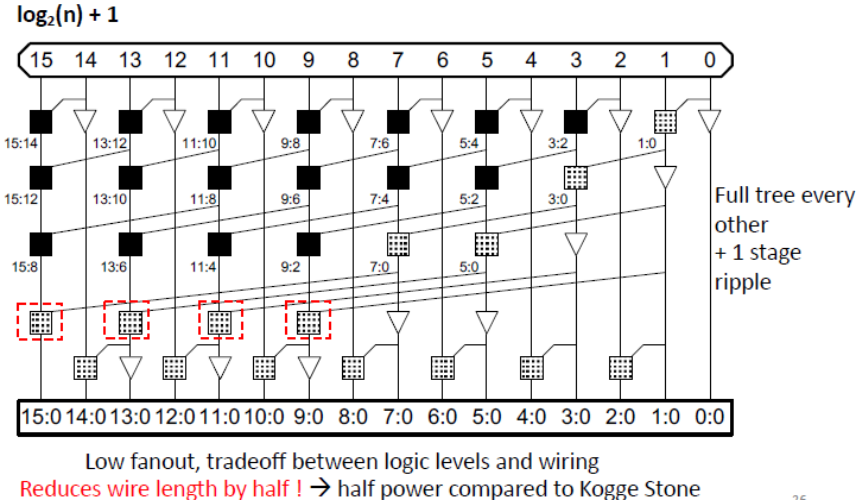
Brent-Kung



Sklansky



Han-Carlson



- Kogge-Stone: low logic levels, low fanout, high wiring
- Brent-Kung: low fanout, low wiring, high logic levels
- Sklansky: low logic levels, low wiring, high fanout

乘法器设计

- 乘法器设计的核心是部分和累加

Example:

1100	:	12_{10}	multiplicand
0101	:	5_{10}	multiplier
1100			
0000			partial products
1100			
0000			
00111100	:	60_{10}	product

M x N比特乘法

- 产生N个M比特部分乘积
- 求和得到M+N比特的结果

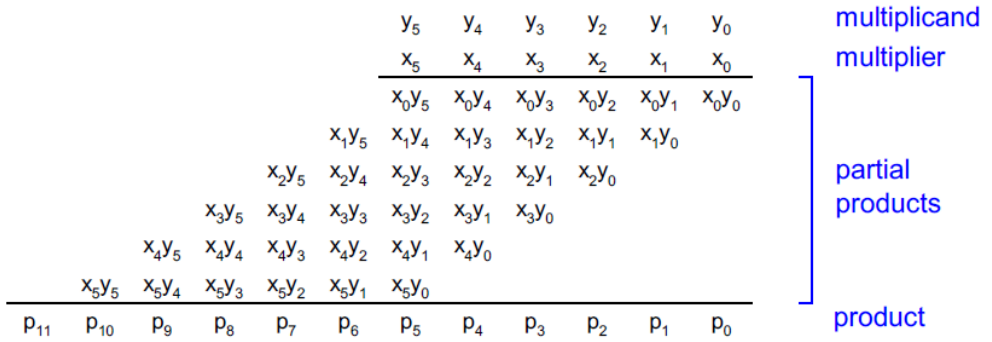
乘法器设计

- 乘法器设计的核心是部分和累加

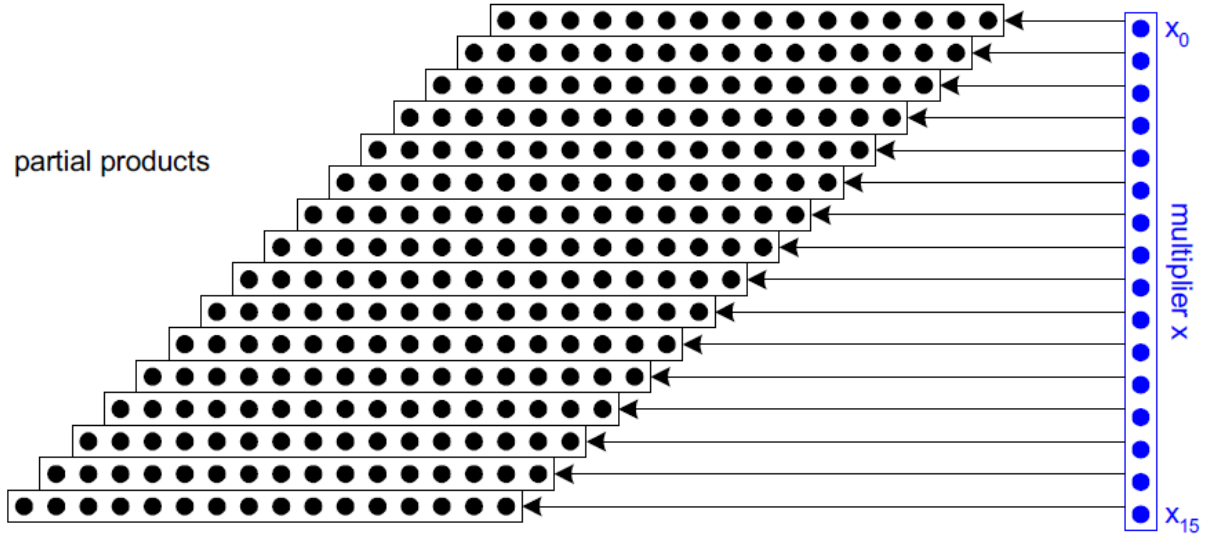
Multiplicand: $Y = (y_{M-1}, y_{M-2}, \ldots, y_1, y_0)$

Multiplier: $X = (x_{N-1}, x_{N-2}, \ldots, x_1, x_0)$

Product:
$$P = \left(\sum_{j=0}^{M-1} y_j 2^j\right)\left(\sum_{i=0}^{N-1} x_i 2^i\right) = \sum_{i=0}^{N-1}\sum_{j=0}^{M-1} x_i y_j 2^{i+j}$$

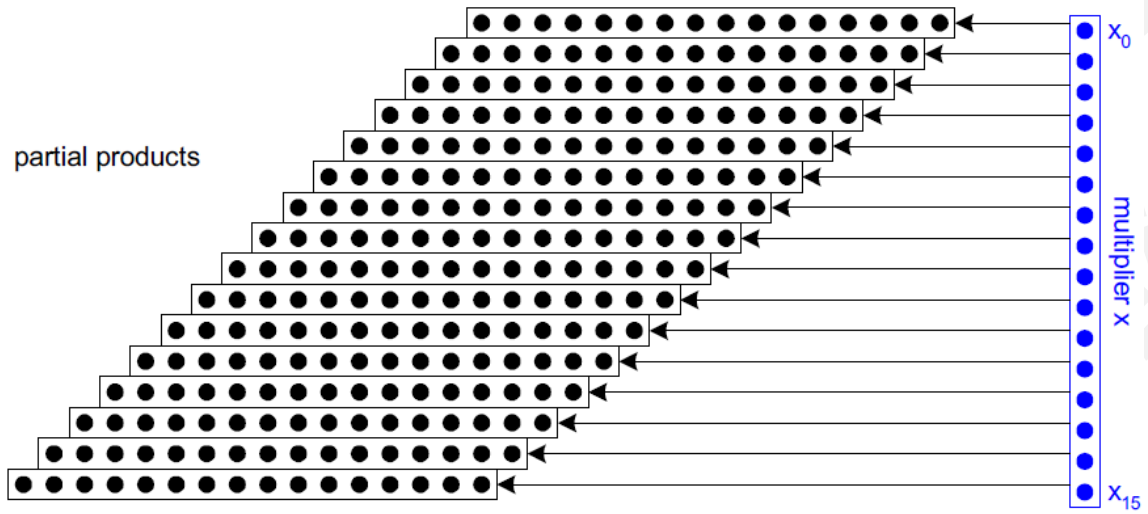


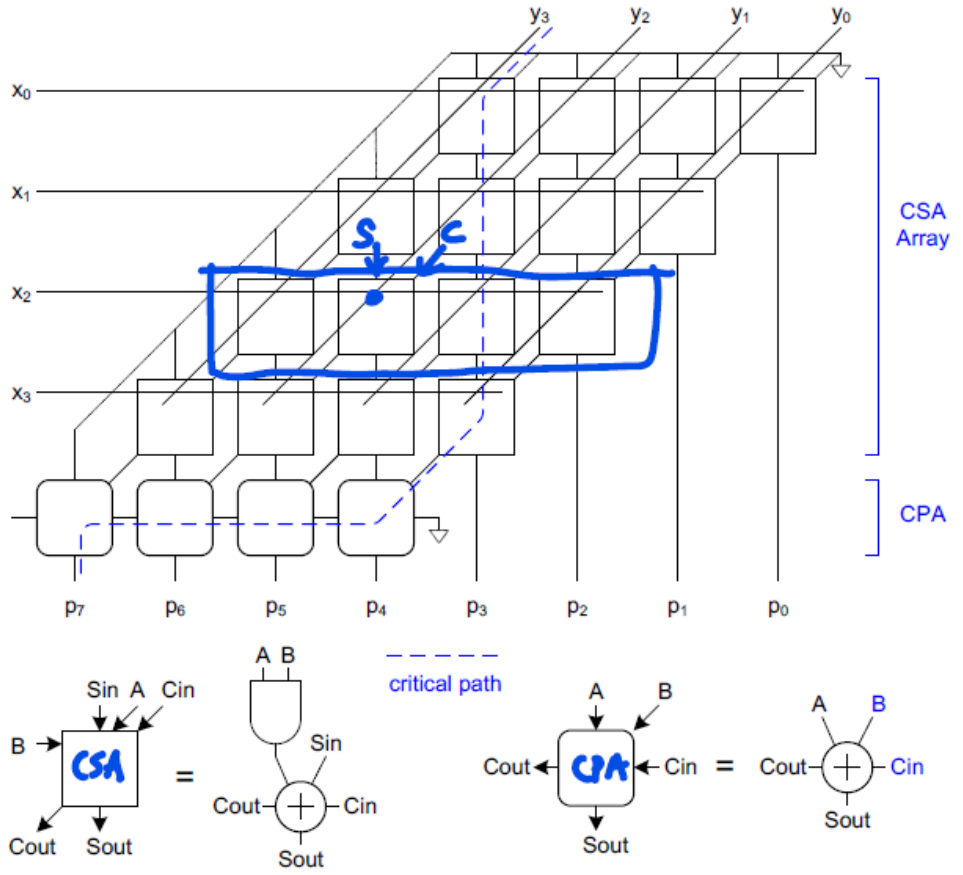
Each dot represents a bit



乘法器设计

- 乘法器设计的核心是部分和累加





乘法器设计

- 如何减少部分和累加的次数?

- Array multiplier requires N partial products

- If we looked at groups of r bits, we could form N/r partial products.
  - Faster and smaller?
  - Called radix-2^r encoding

- Ex: r = 2: look at pairs of bits
  - Form partial products of 0, Y, 2Y, 3Y
  - First three are easy, but 3Y requires adder ☹

```
                1   1   0   0
              (0   1) (0   1
              ---------------
                a   a   a   a
            b   b   b   b
          -----------------
```

X PP
(0 0) 0
(0 1) Y
(1 0) 2Y (4Y-2Y)
(1 1) 3Y = (4Y-Y)

乘法器设计

- 如何减少部分和累加的次数 – 布斯编码（Radix-2^r)

  - Instead of 3Y, try –Y, then increment next partial product to add 4Y
  - Similarly, for 2Y, try –2Y + 4Y in next partial product

Inputs			Partial Product	Booth Selects		
x_{2i+1}	x_{2i}	x_{2i-1}	PP_i	$SINGLE_i$	$DOUBLE_i$	NEG_i
0	0	0	0	0	0	0
0	0	1	Y	1	0	0
0	1	0	Y	1	0	0
0	1	1	$2Y$	0	1	0
1	0	0	$-2Y$	0	1	1
1	0	1	$-Y$	1	0	1
1	1	0	$-Y$	1	0	1
1	1	1	$-0\ (=0)$	0	0	1

4y-2y 1 0

4y-y 1 1

Y -Y Y

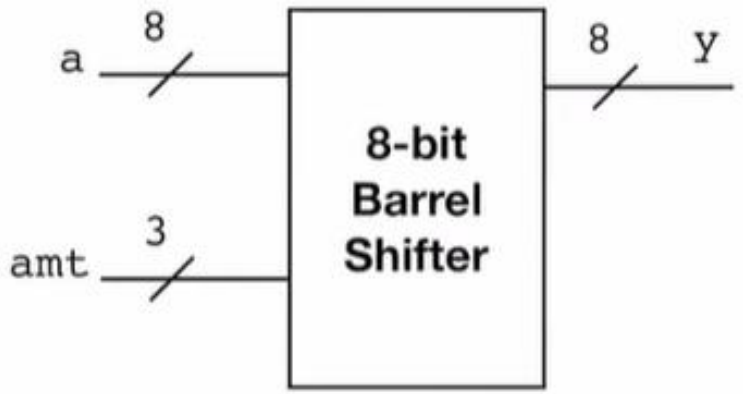
作业题

位移器设计

- Shifter也是重要的数字电路模块之一

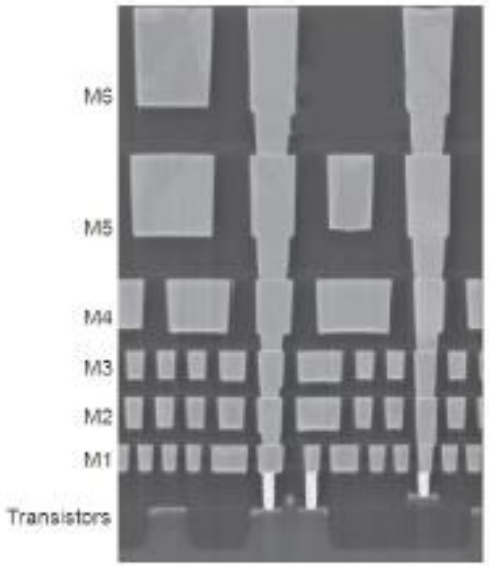
```
module barrel_shifter
    (
        input logic [7:0] a,
        input logic [2:0] amt,
        output logic [7:0] y
    );

    always_comb
        case(amt)
            3'b000: y = a;
            3'b001: y = {a[0], a[7:1]};
            3'b010: y = {a[1:0], a[7:2]};
            3'b011: y = {a[2:0], a[7:3]};
            3'b100: y = {a[3:0], a[7:4]};
            3'b101: y = {a[4:0], a[7:5]};
            3'b110: y = {a[5:0], a[7:6]};
            3'b111: y = {a[6:0], a[7]};
            default: y = a;
        endcase
endmodule
```



线路分析

- Wire Geometry



Intel 90 nm Stack

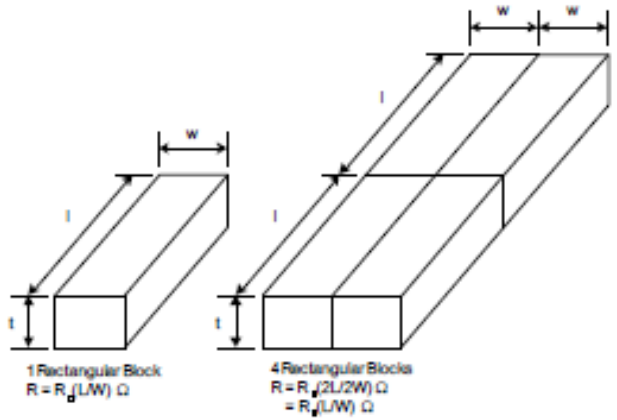
Intel 45 nm Stack

线电阻的计算方式

- ρ = *resistivity* (Ω*m)

$$R = \frac{\rho}{t}\frac{l}{w} = R_{\square}\frac{l}{w}$$

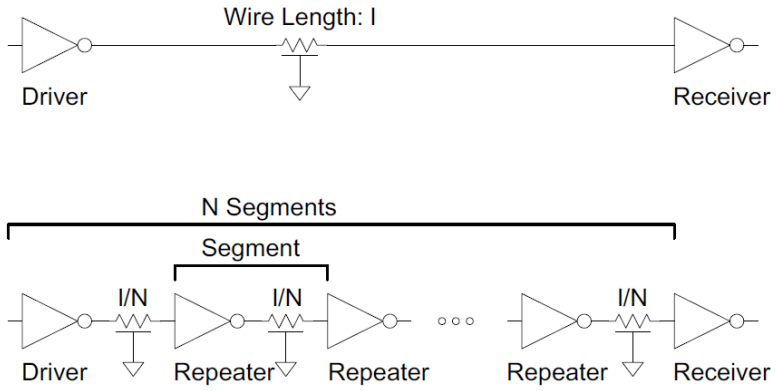
- $R_{\square}$ = *sheet resistance* (Ω/□)
  - □ is a dimensionless unit(!)
- Count number of squares
  - $R = R_{\square} * (\#\ of\ squares)$



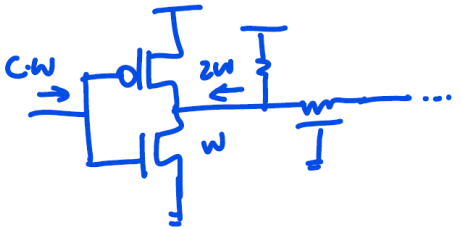
线路分析

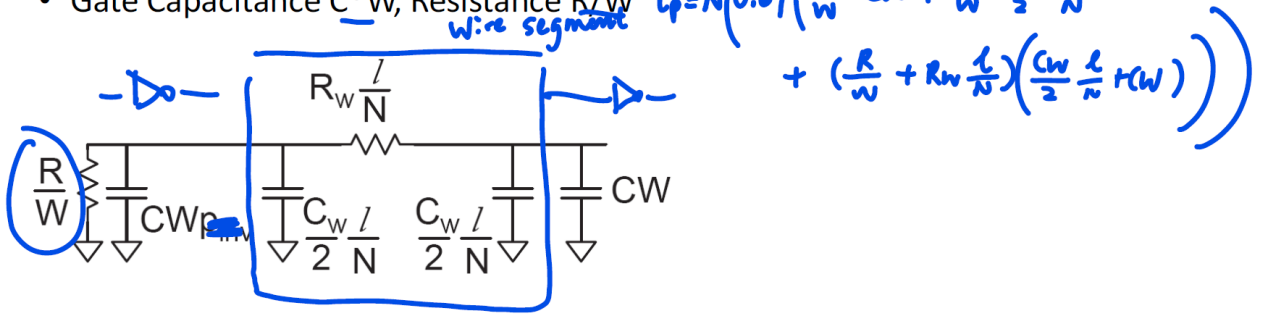
- **Wire Repeaters**

- R和C正比与l
- RC延迟正比于l平方
  --对于长走线是完全不可接受的
- 将长走线切分为N段更短的走线
  --通过反相器或者缓冲器来驱动每一段走线



- How many repeaters should we use?
- How large should each one be?
- Equivalent Circuit
  - Wire length l/N
    - Wire Capacitance $C_w * l/N$, Resistance $R_w * l/N$
  - Inverter width W (nMOS = W, pMOS = 2W)
    - Gate Capacitance C*W, Resistance R/W





- Wire Repeaters

- Write equation for Elmore Delay
  - Differentiate with respect to W and N
  - Set equal to 0, solve

unit wire segment

unit inv resistance

unit inv cap

unit wire cap

unit wire res

$$\frac{l}{N} = \sqrt{\frac{2RC'}{R_w C_w}}$$

$$C' = C(1 + p_{\text{inv}})$$

$$W = \sqrt{\frac{RC_w}{R_w C'}}$$

作业题