北京大学
PEKING UNIVERSITY

智能硬件体系结构

第二讲：电路基础-晶体管与数字电路设计

主讲：陶耀宇、李萌

2024年秋季

注意事项

北京大学 PEKING UNIVERSITY

- 培养学生初步理解**智能时代的硬件芯片的工作原理、设计原理与未来发展方向**

课程名称	智能硬件体系结构 Hardware Architecture for Intelligent computing Systems
课程编号	04632042
学分	2 学分
总学时	34 学时
上课时间	1-16周 每周周三 5-6节
上课地点	一教 202
考核方式	- **作业 8 次：20%，** （2 周 1次，1 次 3-5 题） - **2次编程实验：2*20%，** （传统体系结构、简单 AI 加速器） - **期末考试：30%** - **出勤：10%**
授课教师	陶耀宇，李萌

注意事项

北京大学 PEKING UNIVERSITY

- 课程作业情况

第1次作业题将在**9.20日周五**放在课程网站上

作业提交截止日期是国庆节后的**10月8日**

作业内容将涵盖9.18和9.25号的课程内容

第1次编程作业：**10月8号~11月8号**

下一次课

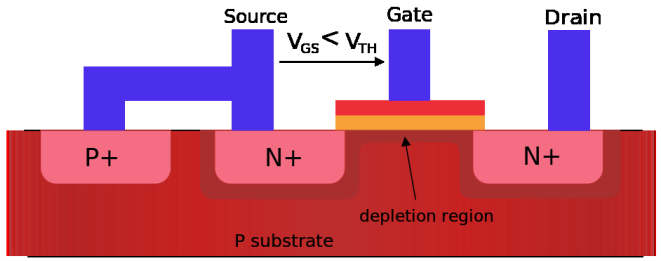
- FSM状态机电路设计
- 存储器电路 – SRAM/DRAM/Flash
- 量化的概念
- Verilog语言基础
- 芯片物理设计基础

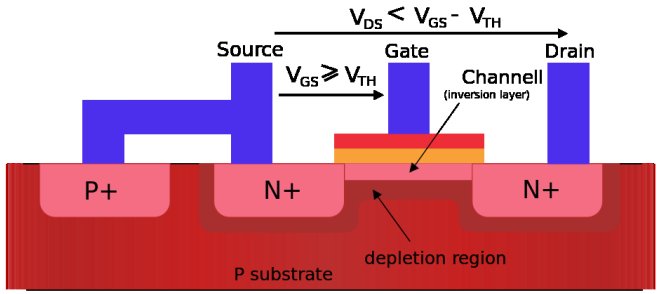
目 录

CONTENTS

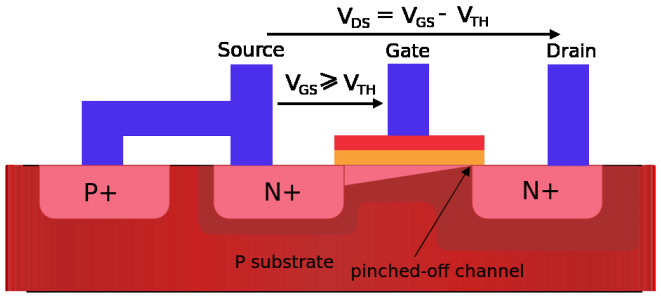
MOSFET晶体 – 现代芯片的基石

- 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）

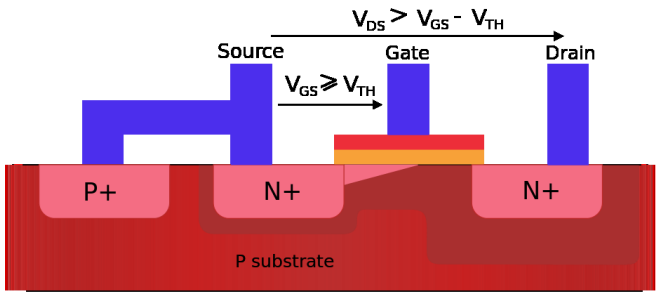




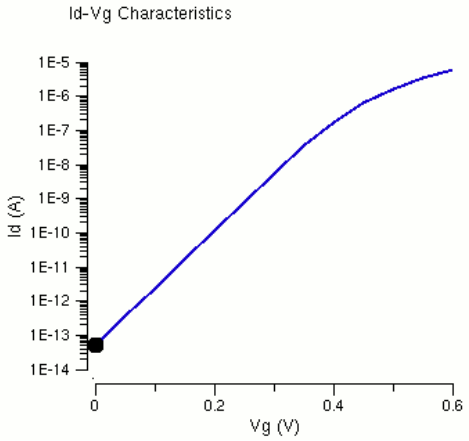
Linear operating region (ohmic mode)

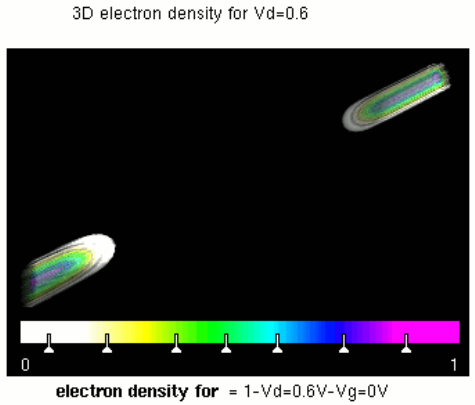


Saturation mode at point of pinch-off



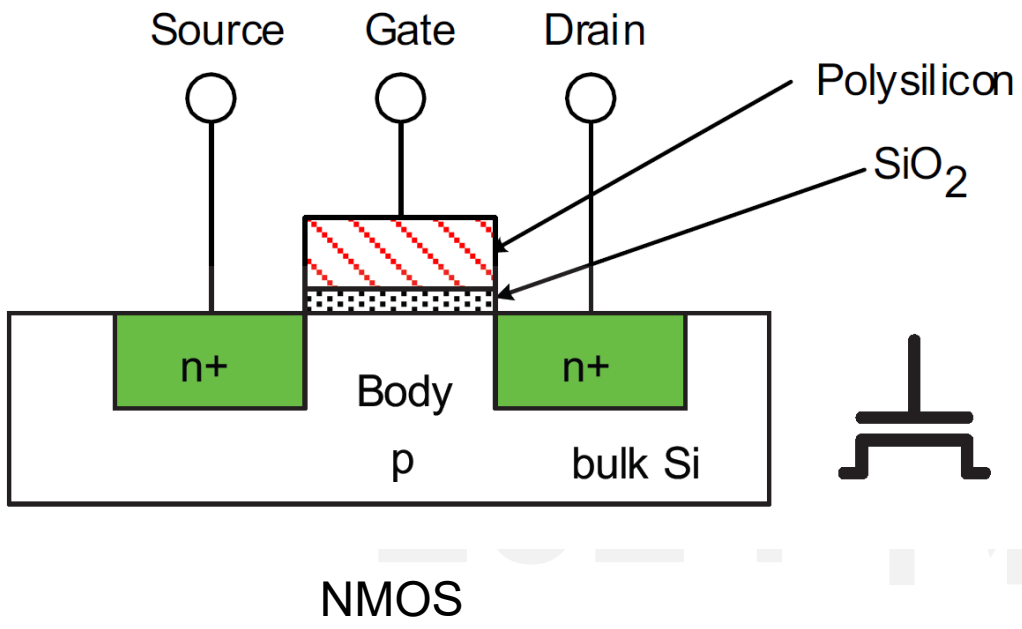
Saturation mode





MOSFET晶体 – 现代芯片的基石

- NMOS与PMOS



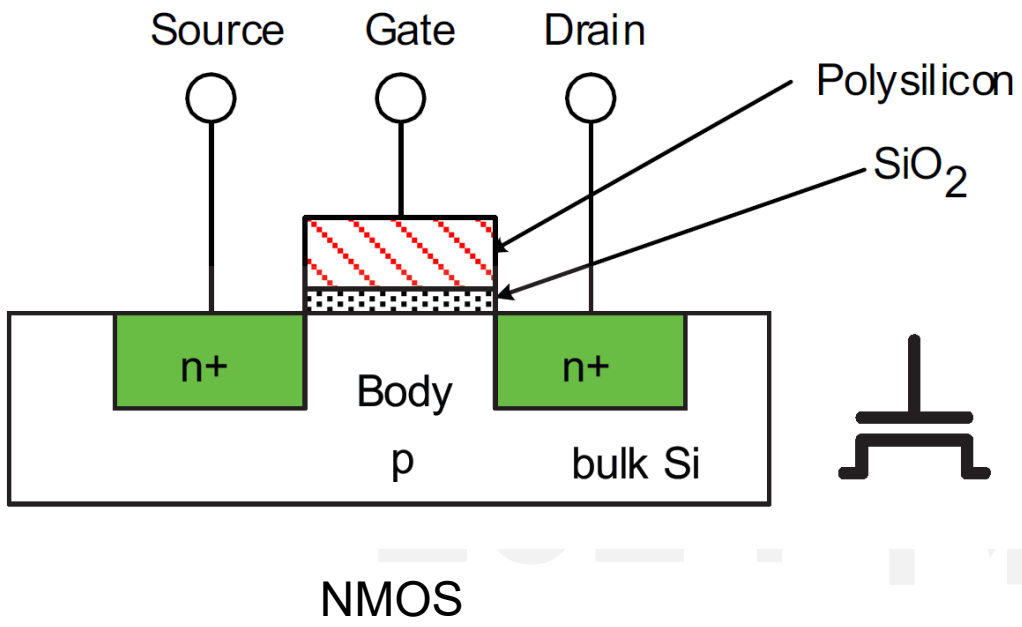
NMOS

When the gate is at a low voltage:

- P-type body is at low voltage
- Source-body and drain-body diodes are OFF
- No current flows, transistor is OFF

MOSFET晶体 – 现代芯片的基石

- NMOS与PMOS



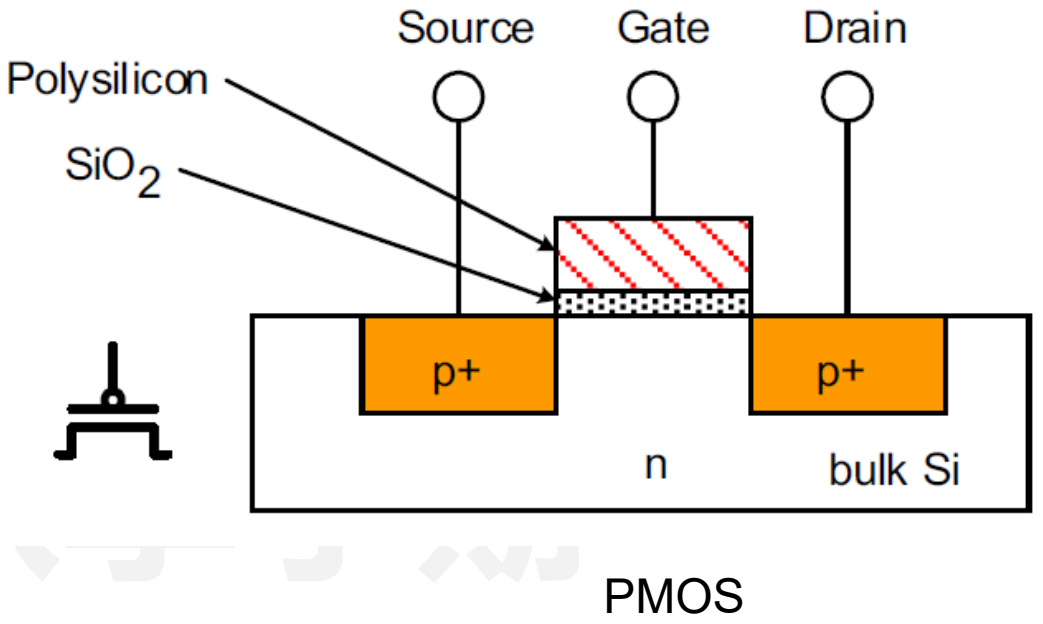
NMOS

When the gate is at a high voltage:

- Positive charge on gate of MOS capacitor; Negative charge attracted to body
- Inverts a channel under gate to n-type
- Now current flow through n-type silicon through channel, transistor ON

MOSFET晶体 – 现代芯片的基石

- NMOS与PMOS



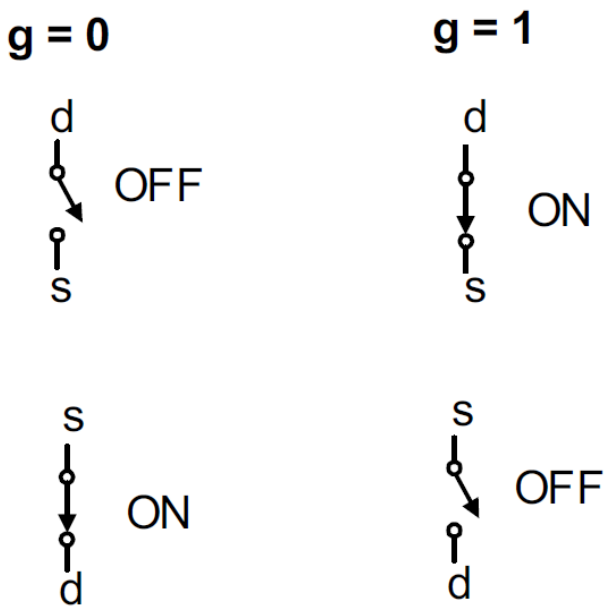
PMOS

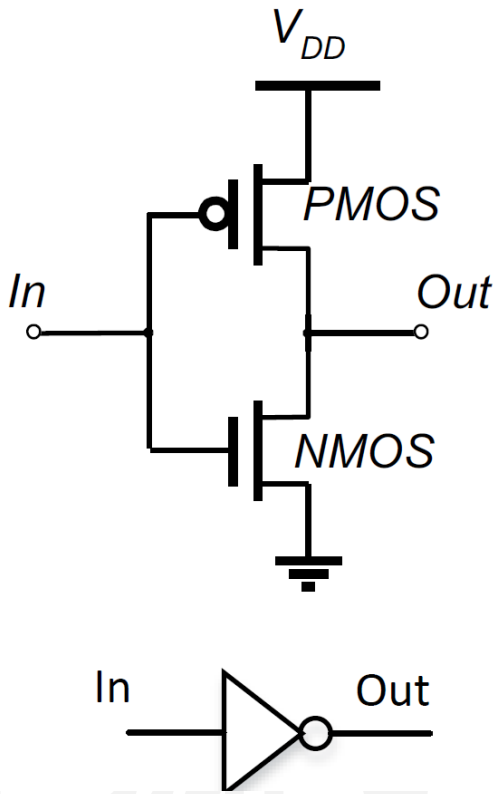
Similar, but doping and voltages reversed

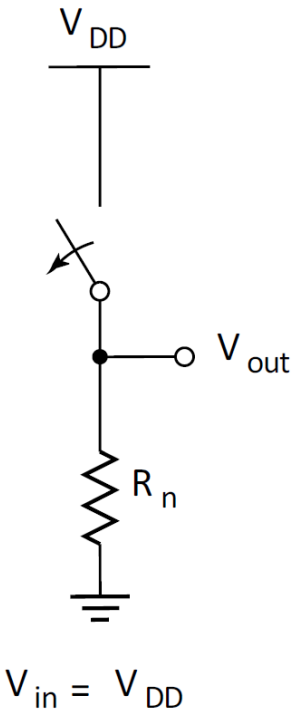
- Body tied to high voltage (VDD)
- Gate low: transistor ON
- Gate high: transistor OFF

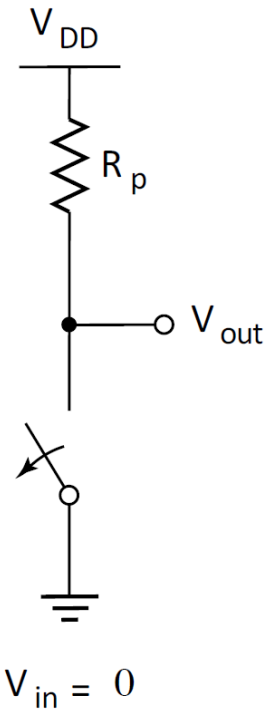
MOSFET作为开关的行为级模型

- NMOS、PMOS和简单反相器









MOSFET组成的静态逻辑电路

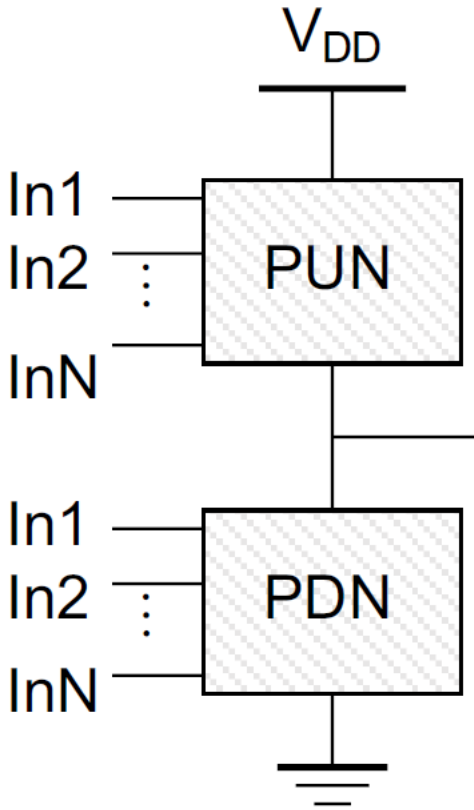
北京大学 PEKING UNIVERSITY

- NMOS、PMOS可以组成PDN和PUN

- 经典静态逻辑电路
- Differential Cascode Voltage Switch (DCVS)
- Pass Gate Logic (传输门逻辑电路)

经典静态逻辑电路

- NMOS、PMOS可以组成PDN和PUN



PMOS only

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

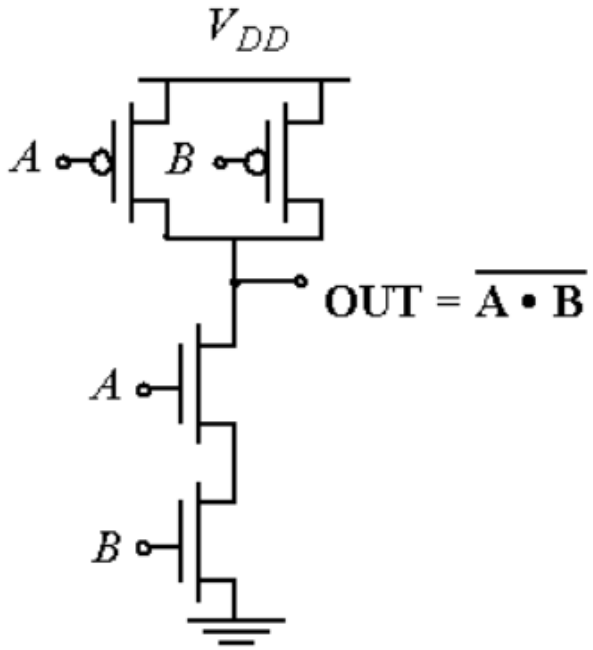
PUN and PDN are *dual* networks

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



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

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

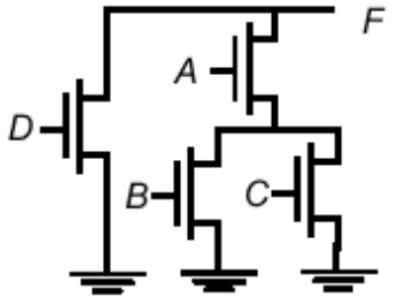
So OUT = Complement of PDN function

Also OUT = PUN function with each input inverted

经典静态逻辑电路

- 由PDN和PUN组成的复杂静态逻辑电路

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



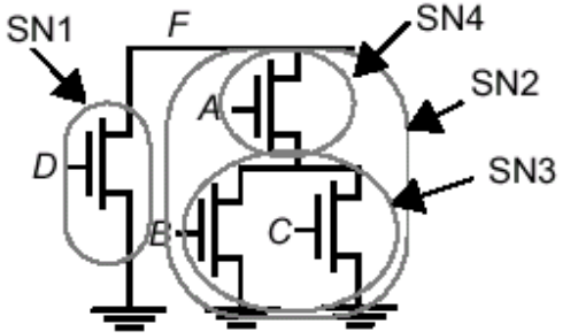
Create PDN directly, ignoring inversion

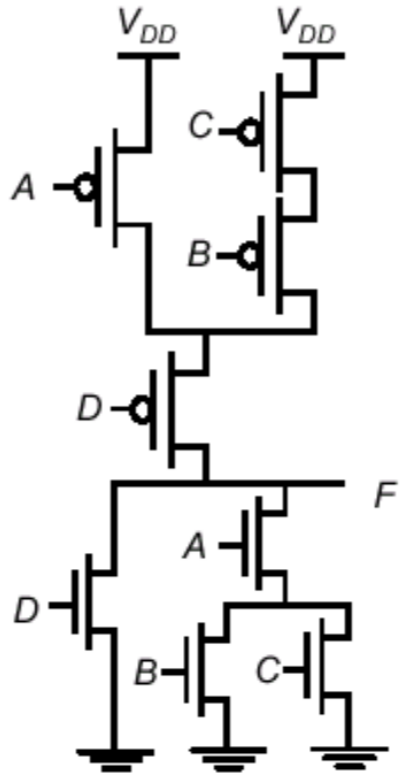
Ex: D + X means that D should be in parallel with X

Identify subnets of PDN

Inside SN3, B and C are parallel, so in PUN, they will be series

Continue in bottom up fashion

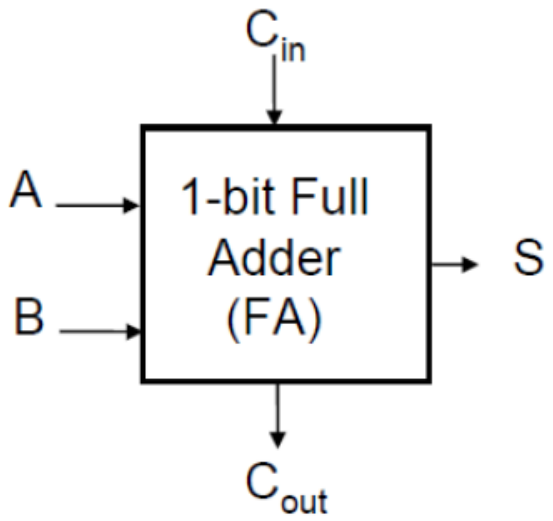




作业题

- 简单1bit加法器电路

- A[n-1:0] + B[n-1:0] = S[n-1:0]

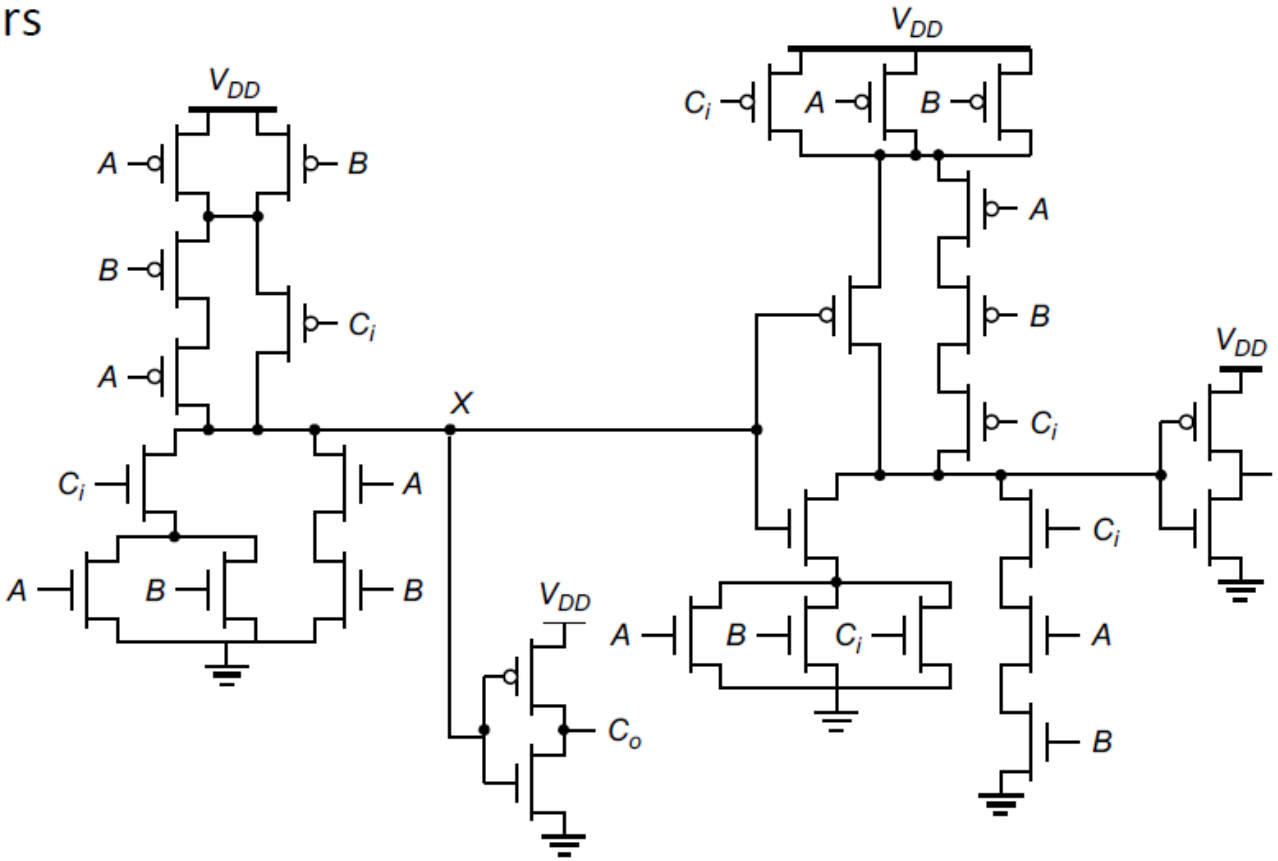


A	B	C_{in}	C_{out}	S	carry status
0	0	0	0	0	kill
0	0	1	0	1	kill
0	1	0	0	1	propagate
0	1	1	1	0	propagate
1	0	0	0	1	propagate
1	0	1	1	0	propagate
1	1	0	1	0	generate
1	1	1	1	1	generate

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

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



MOSFET组成的静态逻辑电路

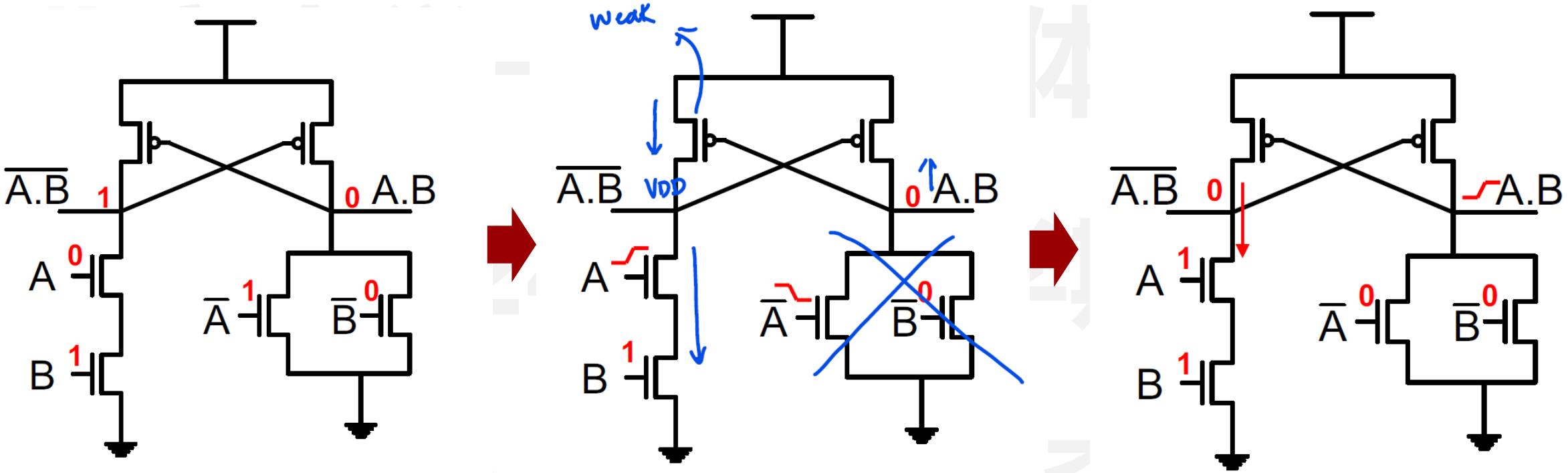
北京大学 PEKING UNIVERSITY

- NMOS、PMOS可以组成PDN和PUN

- 经典静态逻辑电路
- Differential Cascode Voltage Switch (DCVS)
- Pass Gate Logic (传输门逻辑电路)

DCVS静态逻辑电路

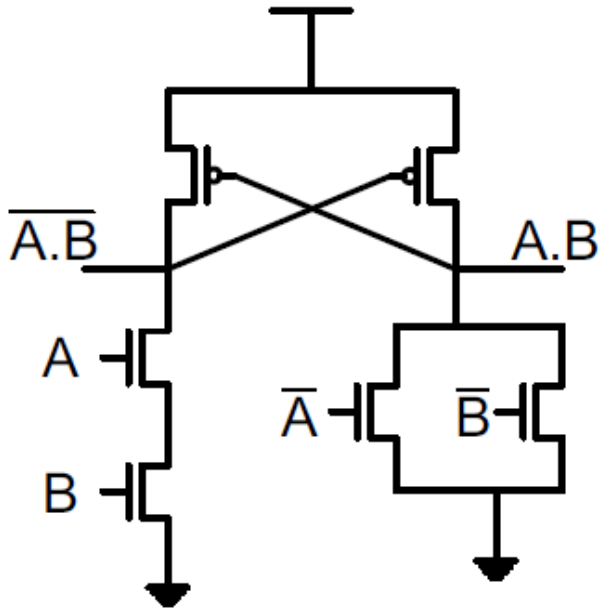
- 简单的AND/NAND电路



作业题

DCVS静态逻辑电路

- 简单的AND/NAND电路



- Advantages:
  - No PMOS duality
    - Lower input cap.
    - Use only NMOS
  - Faster than CMOS
  - Can evaluate complex logic trees in 1 stage

- Disadvantages:
  - Need complementary inputs (dual rail)
  - Cross-bar current
    - Sensitive to input timing
  - Sizing of PMOS is hard
    - Too large → PDN does not switch the output
    - Too small → Slow rise time

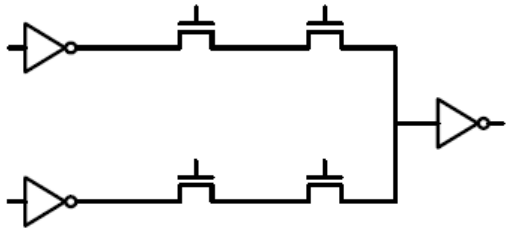
MOSFET组成的静态逻辑电路

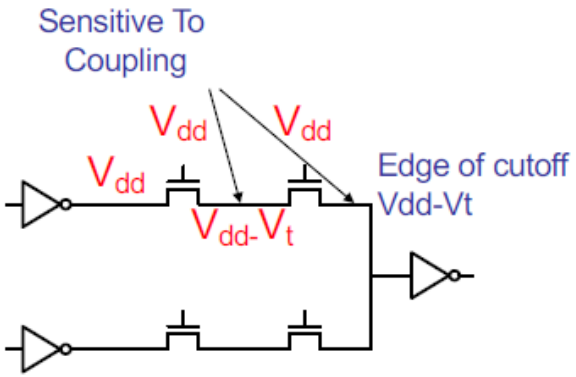
- NMOS、PMOS可以组成PDN和PUN

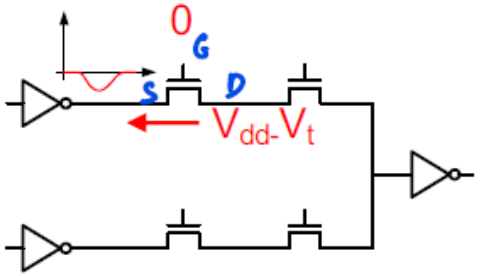
- 经典静态逻辑电路
- Differential Cascode Voltage Switch (DCVS)
- Pass Gate Logic (传输门逻辑电路)

传输门逻辑电路

- 传输门型电路 – Pass Transmission Logic







- Advantages:
  - Speed: 20-50% faster (less capacitance)
    - In some circuits, such as XOR/MUX
  - Lower dynamic power

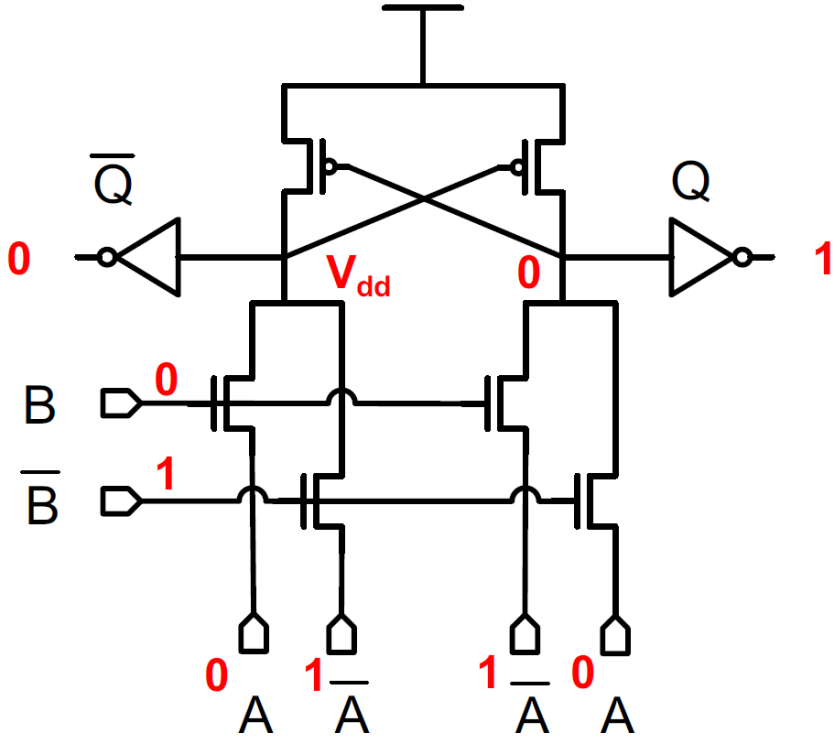
- Disadvantages:
  - Very sensitive to noise: 2 types
    - Vt drop → must level restore someplace
    - Sensitive to undershot voltage noise on inputs
      - Buffers at inputs suppresses this effect
  - Body effect → limit # of FETs in series

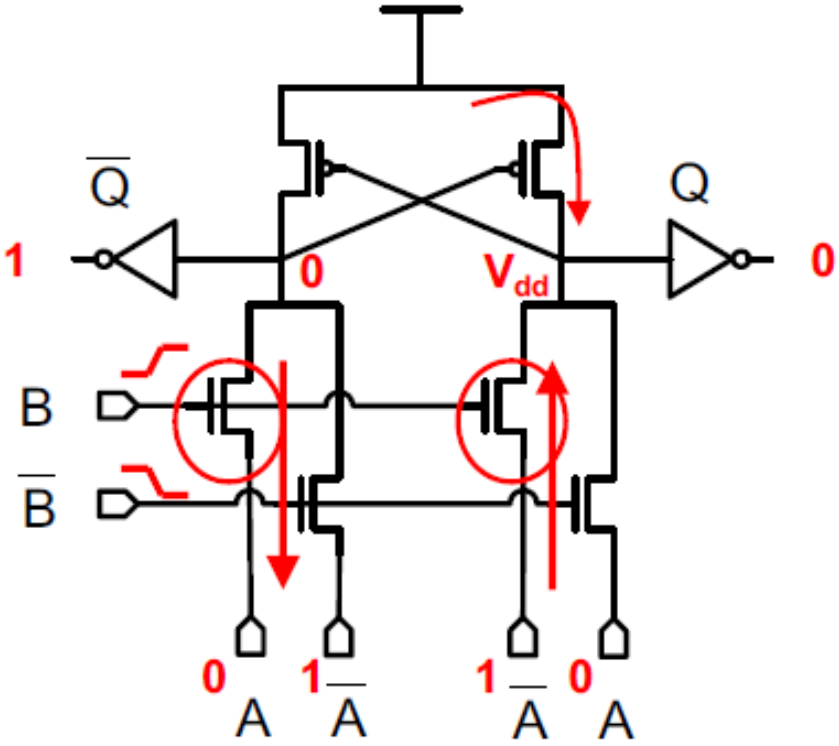
传输门逻辑电路

- 传输门型电路 – Pass Transmission Logic

A	B	Q
1	1	1
0	1	0
1	0	0
0	0	1

xnor





Dual rail pass gate logic with differential cascode voltage switch logic

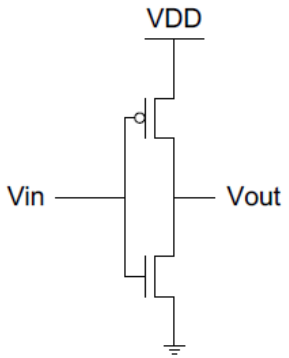
- Combination of DCVS and PTL, Very fast
- Strong PMOS, slow fall; Weak PMOS, slow rise

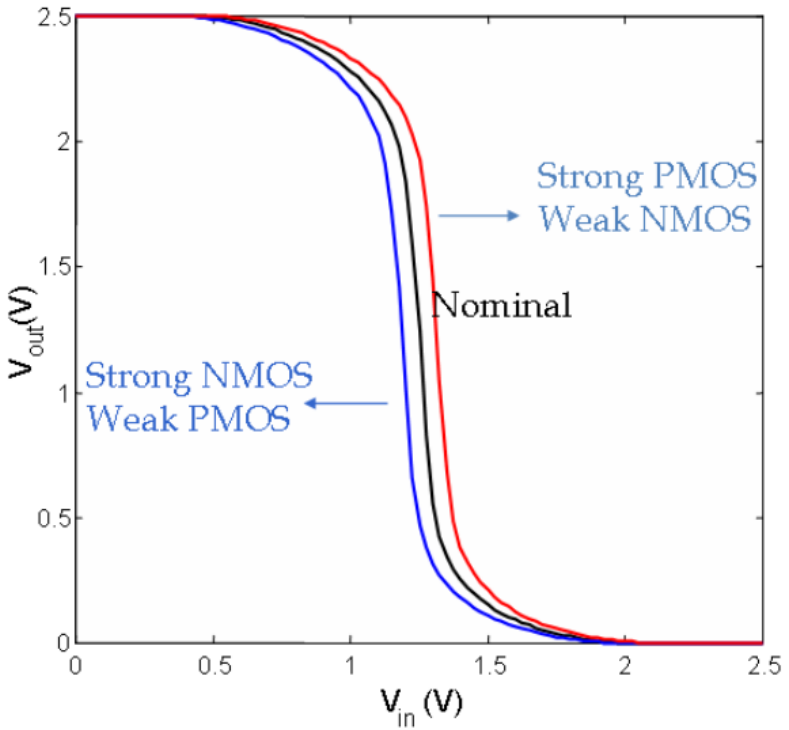
目 录

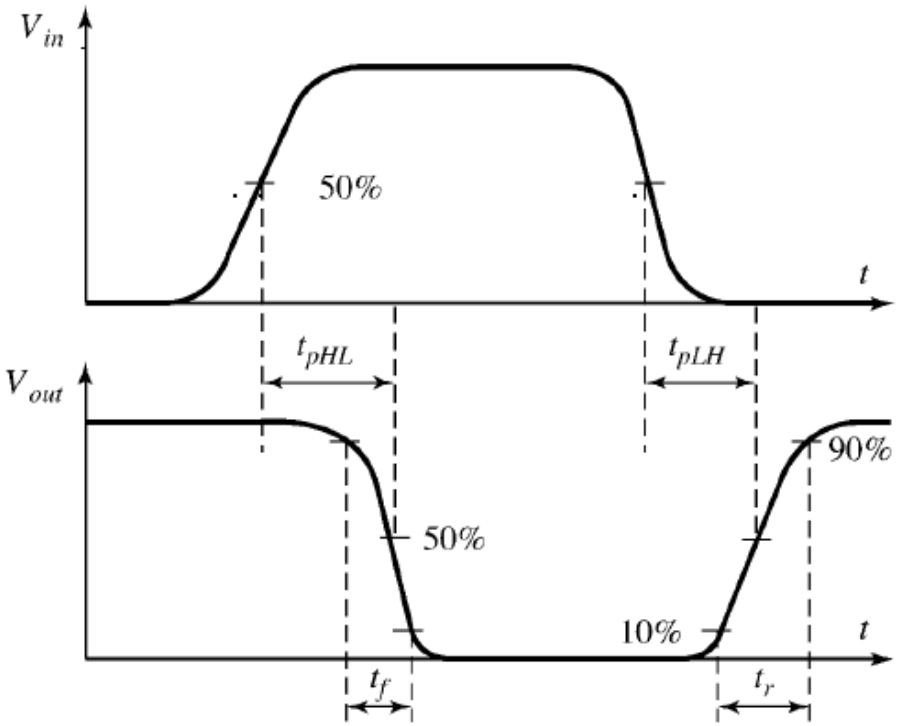
CONTENTS

什么是电路的延迟

- 以Inverter反相器为例



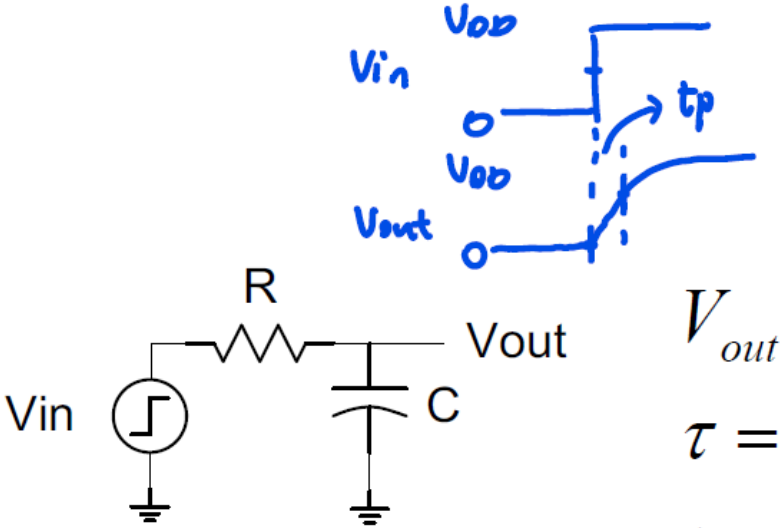




什么是电路的延迟

- 一阶RC延迟分析

A First-Order RC Network



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

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

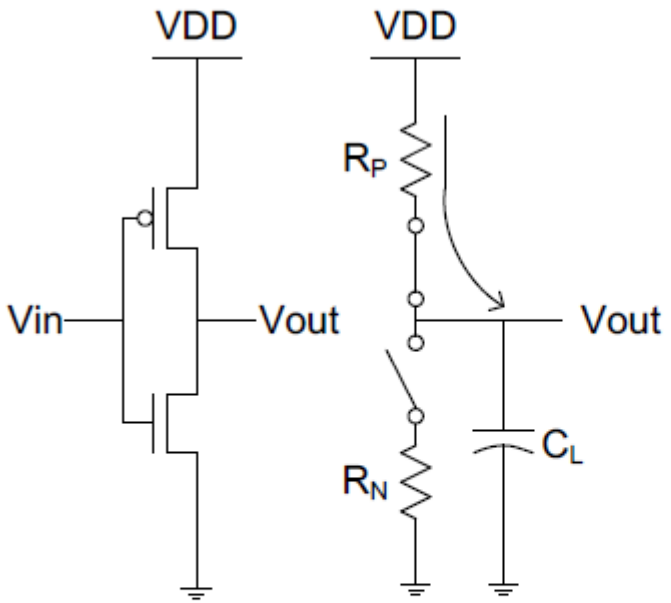
$$-t/\tau = -\ln 2$$

$$\tau = R \times C$$

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

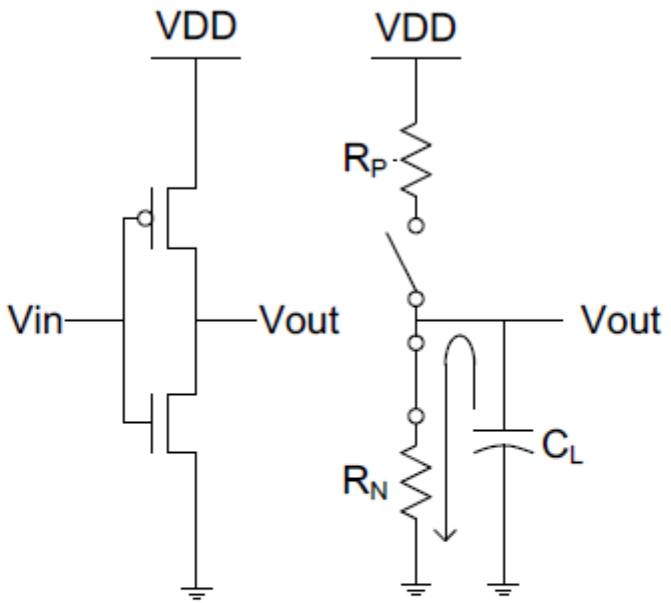
反相器延迟

- 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

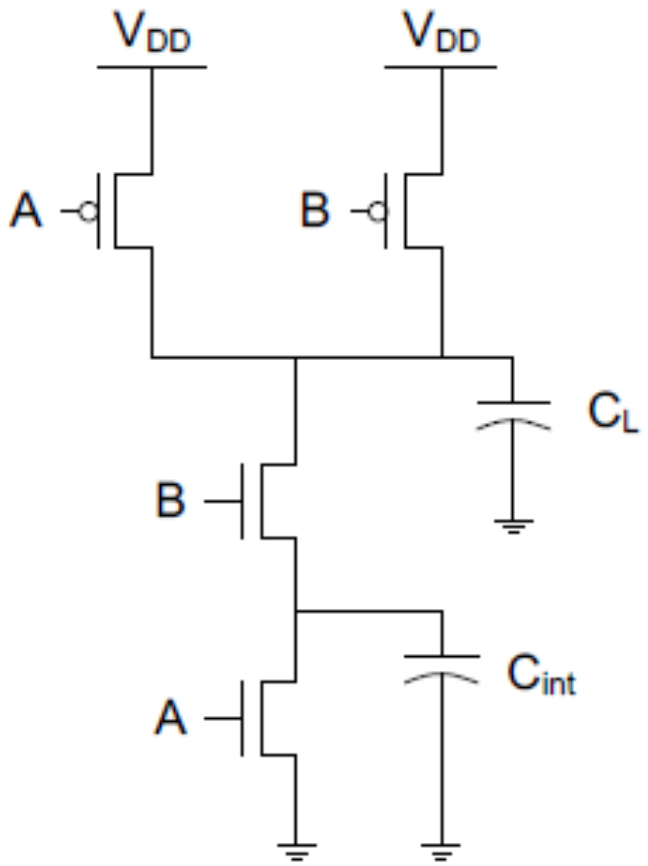
- 利用一阶RC延迟分析方法

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

- Keep capacitances small – lower C
  - Compact layout, good placement (short wires & no diffusion routing)
- Increase transistor sizes – lower R
  - Watch out for self-loading! – parasitic C increases!
- Increase VDD
  - Not usually possible due to reliability and power penalties

输入Pattern对延迟的影响

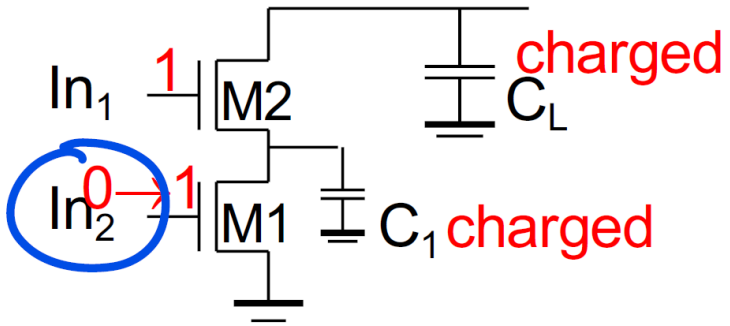
- 利用一阶RC延迟分析方法



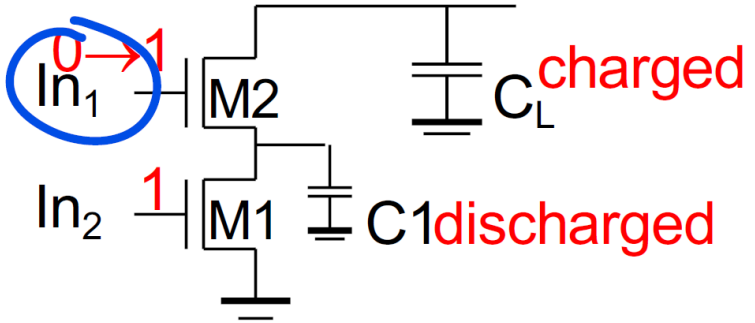
- Delay is dependent on the pattern of inputs
- *Ignore C_{int} for the moment!*
- Low to high transition
  - both inputs go low
    - delay is 0.69 $R_p/2$ C_L
  - one input goes low
    - delay is 0.69 R_p C_L
- High to low transition
  - both inputs go high
    - delay is 0.69 $2R_n$ C_L

利用Transistor Ordering提升逻辑速度

- 复杂的Transistor Ordering需要仿真工具支持



delay determined by time to discharge C_L, C_1

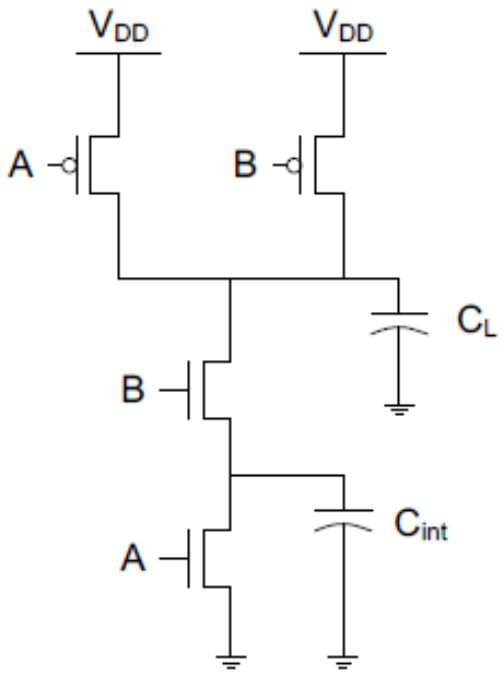


delay determined by time to discharge C_L

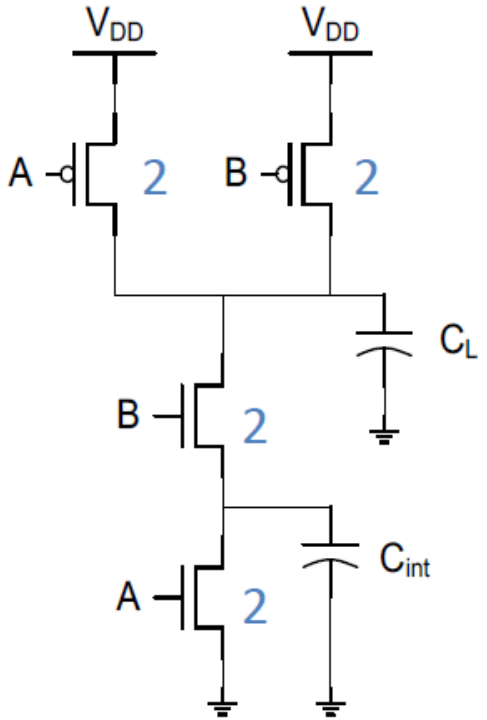
作业题

逻辑电路的Transistor Sizing

- 目标是将PDN和PUN的延迟进行匹配

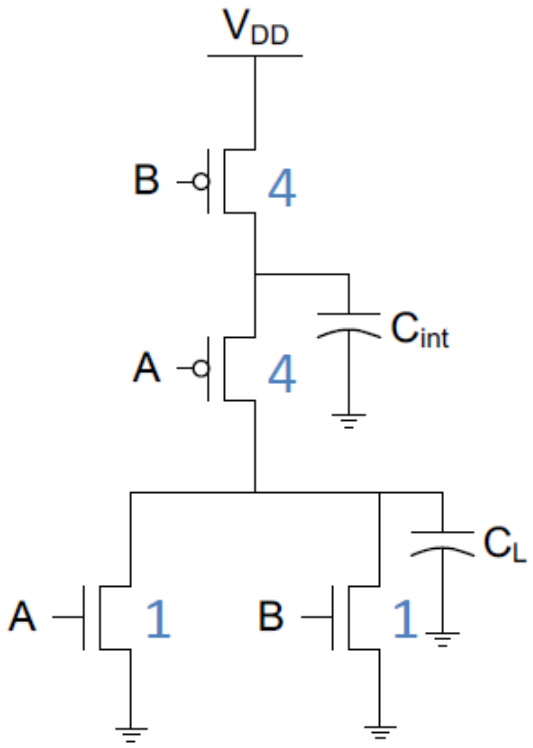


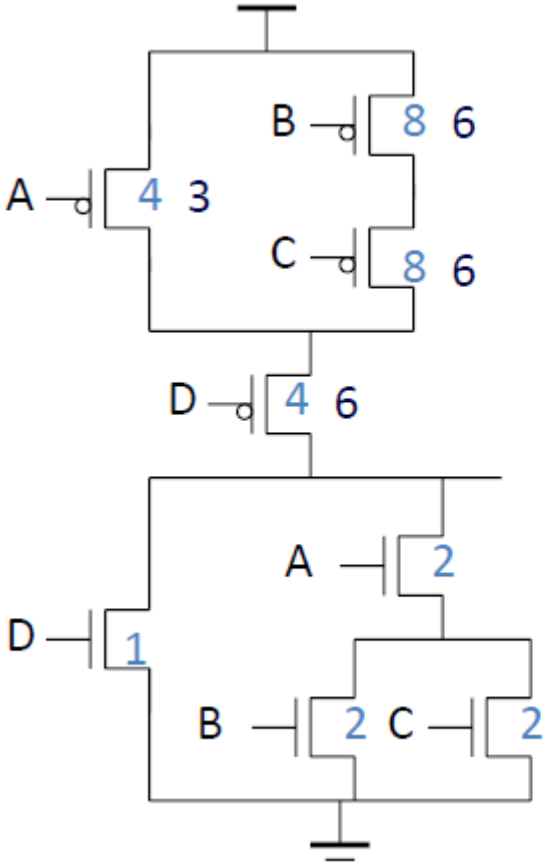
- Delay is dependent on the pattern of inputs
- *Ignore C_{int} for the moment!*
- Low to high transition
  - both inputs go low
    - delay is 0.69 $R_p/2$ C_L
  - one input goes low
    - delay is 0.69 R_p C_L
- High to low transition
  - both inputs go high
    - delay is 0.69 $2R_n$ C_L



逻辑电路的Transistor Sizing

- 目标是将PDN和PUN的延迟进行匹配





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

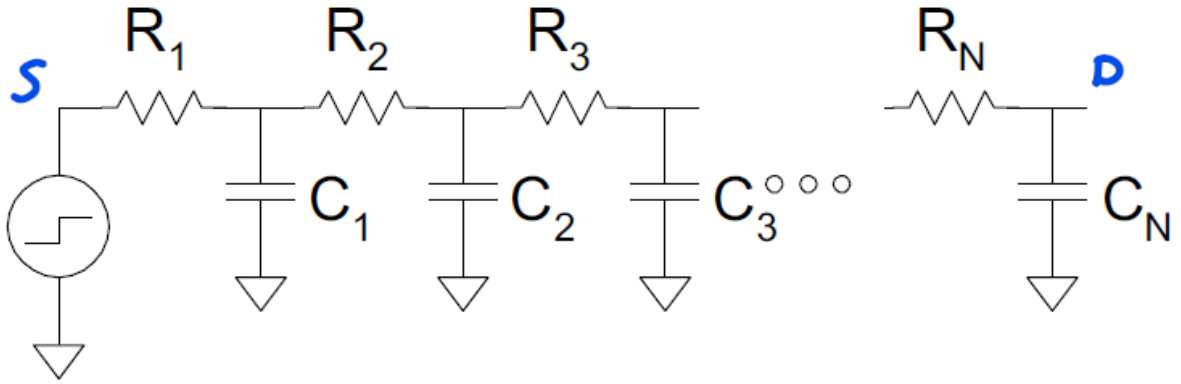
逻辑电路的Elmore Delay模型

- 拓展多级的RC模型

  - ON transistors look like resistors
  - Circuit network modeled as *RC ladder*
  - Elmore delay of RC ladder
  - Apply to complex gates (i.e., stacks), also interconnect (later)

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

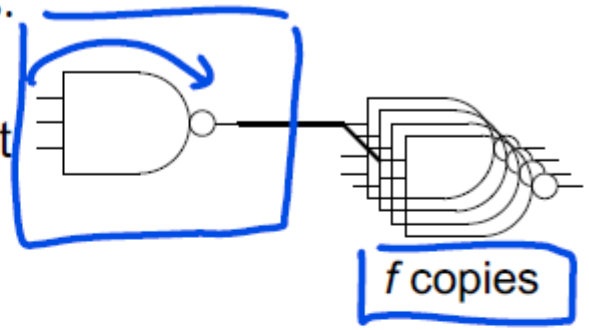
$$= R_1 C_1 + (R_1 + R_2) C_2 + \ldots + (R_1 + R_2 + \ldots + R_N) C_N$$

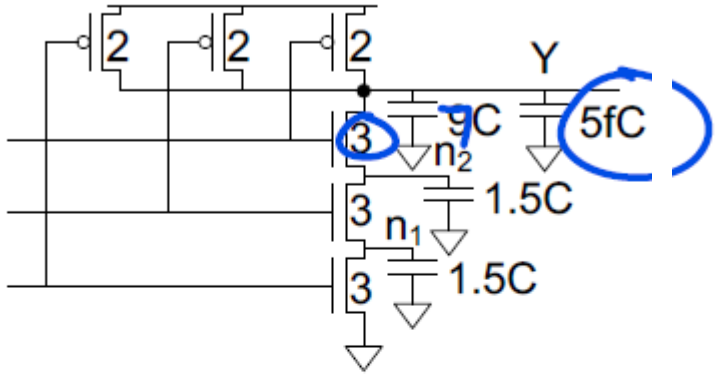


逻辑电路的Elmore Delay模型

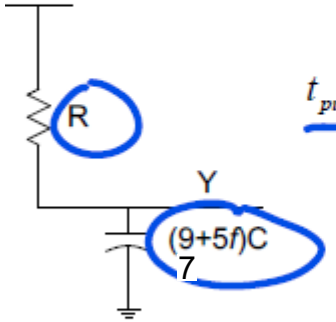
- 拓展多级的RC模型 – 3-input NAND gates

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

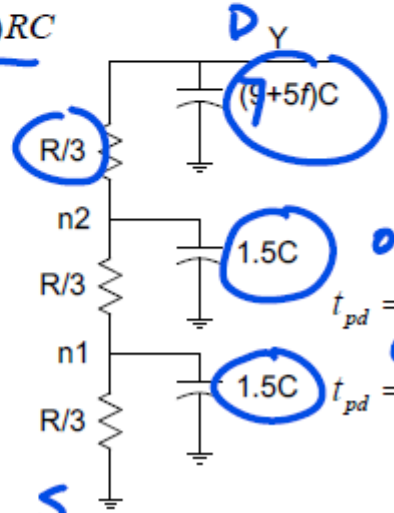




Worst-case pull up



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



Worst-case pull down

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

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

作业题

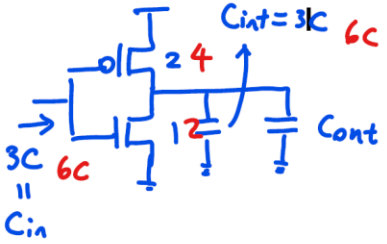
引入逻辑功效

- **最基本的问题：如何对一串逻辑门电路进行Transistor Sizing以最小化延时?**

逻辑门电路的延迟: d = f + p

- *f*: *effort delay* = *gh* (a.k.a. stage effort)
  - Again has two components
    - *g*: *logical effort*
      - Measures relative ability of gate to deliver current
      - *g* o 1 for inverter
    - *h*: *electrical effort* = C_{out} / C_{in}
      - Ratio of output to input capacitance
      - Sometimes called fanout
- *p*: parasitic delay
  - Represents delay of gate driving no load
  - Set by internal parasitic capacitance

d = t_p/τ t ~ 9ps in typical 130nm process



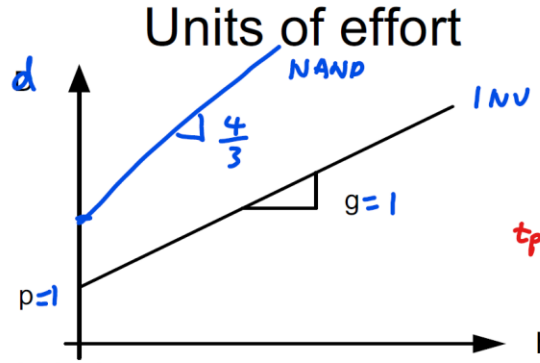
$t_p = 0.69\,R(3C + C_{int})$

$= 0.69(3RC) + 0.69\,R\,C_{int}$

$d = 1 + \frac{C_{int}}{3C}$

$= \frac{C_{int}}{C_{in}} + 1$

$= gh + p \quad (p = 1)$



- Reference is the inverter:
  $g_{inv} = 1$
  $p_{inv} = 1$
- g is a function of the complexity of a gate, not its size
- p is a function of the technology and gate type
- τ ~ 9ps in typical 130nm process

$d = f + p$

$= gh + p$

$= h + p$

$t_p = 0.69 \cdot \frac{R}{2}(6C + C_{int})$

$= 0.69(3RC) + 0.69\frac{R}{2}C_{int}$

$d = 1 + \frac{C_{int}}{6C}$

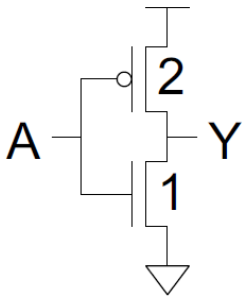
$= \frac{C_{int}}{C_{in}} + 1$

$= gh + p \quad (p = 1)$

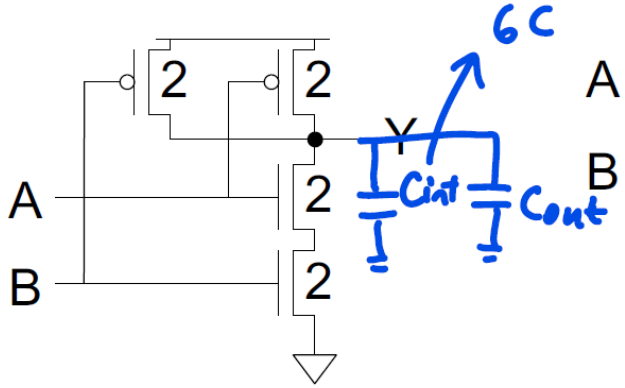
不同逻辑门电路的Logical Effort (g)

- **最基本的问题：如何对一串逻辑门电路进行Transistor Sizing以最小化延时?**

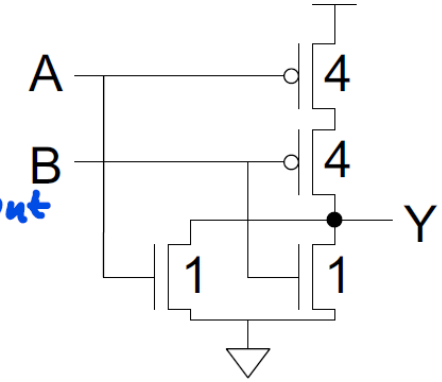
  - DEF: *Logical effort is the ratio of the input capacitance of a gate to the input capacitance of an inverter delivering the same output current (i.e, effort delay under same loading)*.
  - Measure from delay vs. fanout plots
  - Or estimate by counting transistor widths



$C_{in} = 3$
$g = 3/3$



$C_{in} = 4C$
$g = 4/3$



$C_{in} = 5$
$g = 5/3$

不同逻辑门电路的Logical Effort (g)

- 最基本的问题：如何对一串逻辑门电路进行Transistor Sizing以最小化延时?

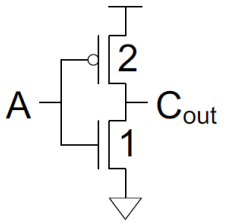
Logical effort (g)

	Number of inputs					
Gate type	**1**	**2**	**3**	**4**	**5**	**n**
Inverter	1					
NAND		4/3	5/3	6/3	7/3	(n + 2)/3
NOR		5/3	7/3	9/3	11/3	(2n + 1)/3
Multiplexer		2	2	2	2	2

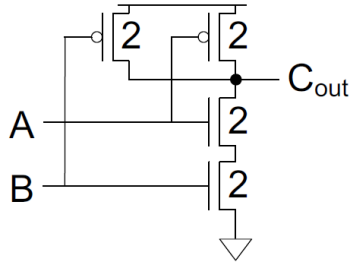
不同逻辑门电路的寄生延迟 (p)

- **最基本的问题：如何对一串逻辑门电路进行Transistor Sizing以最小化延时?**

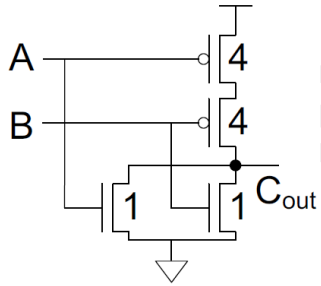
- Measure from delay vs. fanout plots
- Or estimate by counting self loading on output node for sizing with equal output current to inverter
- For simplification, we ignore internal node capacitance



C_{out} = 3
p = 3/3



C_{out} = 6
p = 6/3 = 2

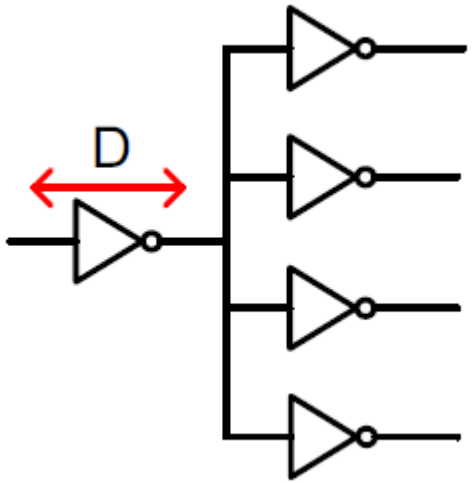


C_{out} = 6
p = 6/3 = 2

Gate type	Parasitic Delay
Inverter	1
n-input NAND	n
n-input NOR	n
n-way multiplexer	2n

利用逻辑功效计算电路延迟

- 单级 - 反相器FO4经典例子与复杂逻辑例子



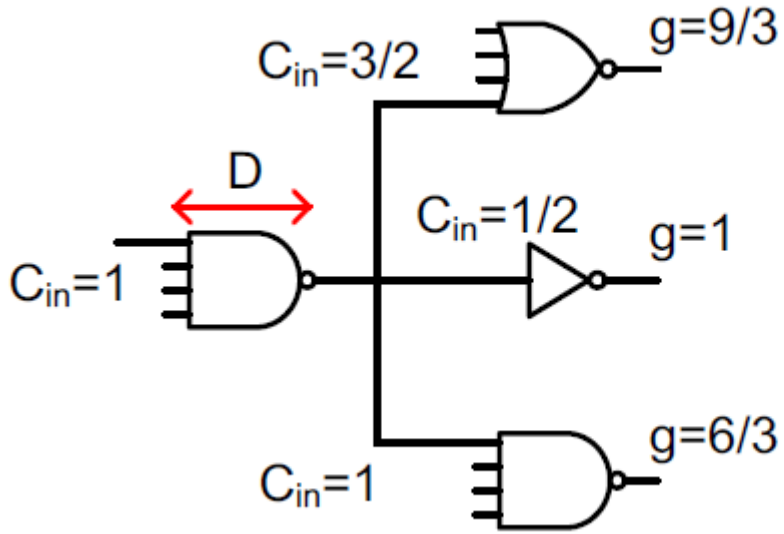
$$C_{out} = 4C_{in}$$

$$h = 4$$

$$g = 1$$

$$p = 1$$

$$D = 4 + 1 = 5\tau$$



$$h = 3$$

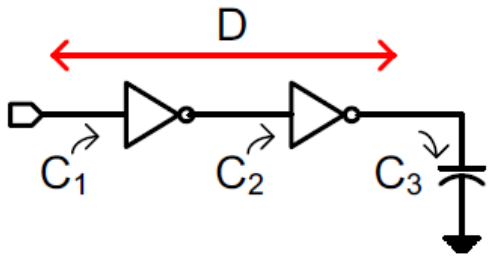
$$g = \frac{6}{3}$$

$$p = 4 \times 1$$

$$D = 10\tau$$

利用逻辑功效计算电路延迟

- 多级电路延迟计算



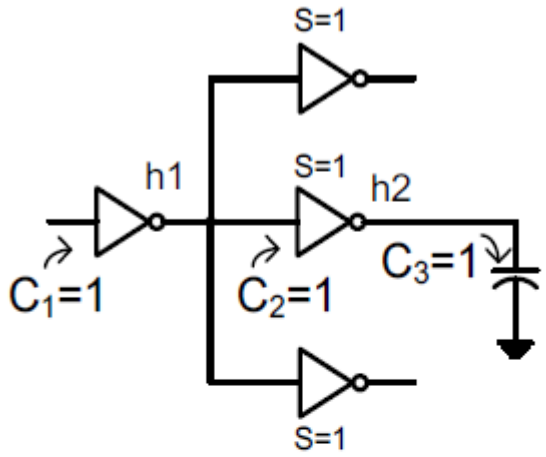
Define path effort H as:

$$H = \frac{C_3}{C_1}$$

$$h_1 = \frac{C_2}{C_1}$$

$$h_2 = \frac{C_3}{C_2}$$

$$\Rightarrow \quad H = h_1 h_2$$



Without branching: $h_1 h_2 = H$

With branching:

$$h_2 = \frac{C_3}{C_2}$$

$$h_1 = \frac{3C_2}{C_1}$$

$$h_1 h_2 = 3H$$

Need to account for branching!

$$b_i = \frac{C_{on-path} + C_{off-path}}{C_{on-path}}$$

$$\Rightarrow$$

$$C_{on-path,2} = C_2$$

$$C_{off-path,2} = 2C_2$$

$$b_1 = \frac{3C_2}{C_2} = 3$$

利用逻辑功效计算电路延迟

- 多级电路延迟计算

$$H = \frac{C_{out}}{C_{in}}$$

$$B = \prod b_i$$

$$BH = \prod h_i$$

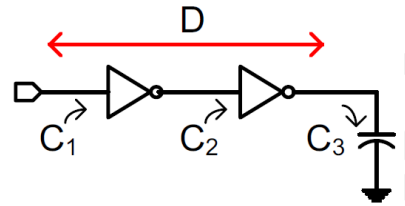
$$G = \prod g_i$$

$$P = \sum p_i$$

Path Effort

$$F = GBH = \prod g_i h_i$$

Total delay $D = g_1 h_1 + p_1 + g_2 h_2 + p_2$



Substitute H $D = g_1 h_1 + p_1 + g_2 \frac{H}{h_1} + p_2$

$$\min\{D\} \Rightarrow \frac{\partial D}{\partial h_1} = g_1 - g_2 \frac{H}{h_1^2} = 0$$ Minimize the delay

$$g_1 - g_2 \frac{h_2}{h_1} = 0$$ Solve for the minimum delay

$$g_1 h_1 = g_2 h_2$$

$$f_1 = f_2$$

Delay is optimal when effort delays are equal

利用逻辑功效计算电路延迟

- 多级电路延迟计算例子

Logical effort (g)

	Number of inputs					
Gate type	**1**	**2**	**3**	**4**	**5**	**n**
Inverter	1					
NAND		4/3	5/3	6/3	7/3	(n + 2)/3
NOR		5/3	7/3	9/3	11/3	(2n + 1)/3
Multiplexer		2	2	2	2	2

Minimum Delay

$$F = GBH = \prod g_i h_i$$

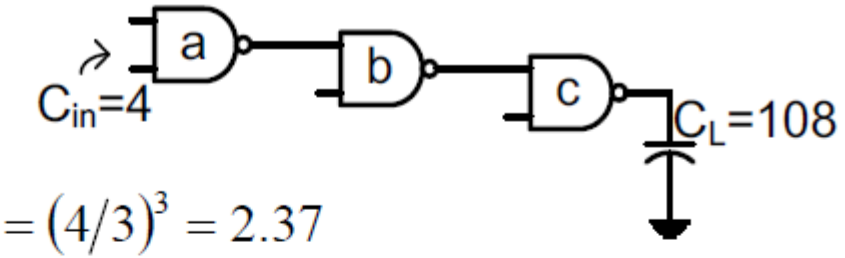
Stage effort of stage i: f_i $\quad g_i h_i = f_i$

Optimal stage effort is: $\quad \hat{f} = f_i$

For N stages: $\quad F = \hat{f}^N \Rightarrow \boxed{\hat{f} = F^{1/N}}$

Minimum Delay: $\quad \hat{D} = \sum_i g_i h_i + p_i$

$$\hat{D} = N\hat{f} + \sum_i p_i = \boxed{N\hat{f} + P}$$



$$G = g_a g_b g_c = (4/3)^3 = 2.37$$

$$B = 1$$

$$H = C_L / C_{in} = 27$$

$$F = GBH = 2.37 \times 27 = 63.99$$

$$\hat{f} = F^{1/3} \approx 4$$

$$\hat{D} = 3(4) + 3(2p_{inv}); \; p_{inv} = 1\tau$$

$$\hat{D} = 12 + 6 \times 1 = 18\tau$$

利用逻辑功效计算电路延迟

- 多级电路延迟计算例子

Logical effort (g)

	Number of inputs					
Gate type	**1**	**2**	**3**	**4**	**5**	**n**
Inverter	1					
NAND		4/3	5/3	6/3	7/3	(n + 2)/3
NOR		5/3	7/3	9/3	11/3	(2n + 1)/3
Multiplexer		2	2	2	2	2

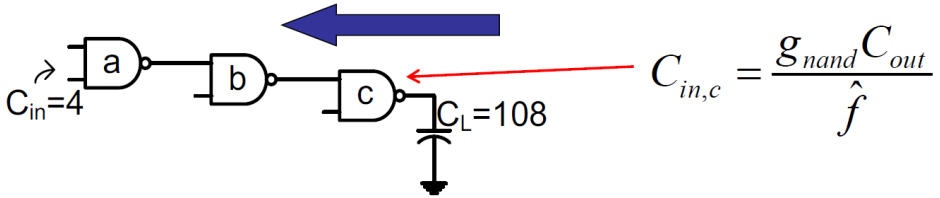
Compute: $\hat{f} = g \cdot h$

$h_i = \hat{f}/g_i = C_{out}/C_{in}$

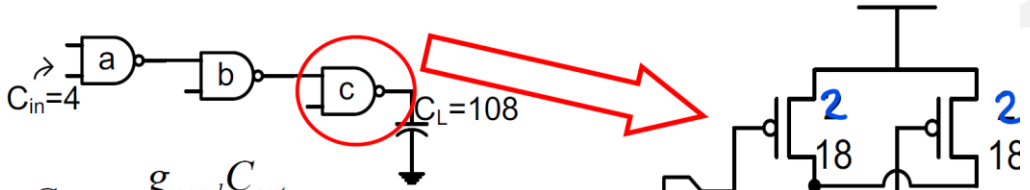
$$C_{in} = \frac{g_i C_{out}}{\hat{f}}$$

$C_{out} = \frac{\hat{f} C_{in}}{g_i}$

Work backwards to size each gate



$$C_{in,c} = \frac{g_{nand} C_{out}}{\hat{f}}$$



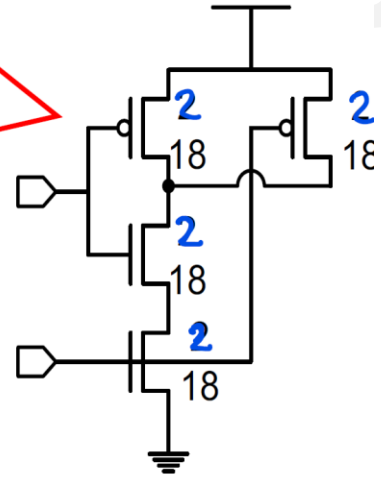
$$C_{in,c} = \frac{g_{nand} C_{out}}{\hat{f}}$$

$$C_{in,c} = \frac{4/3 \times 108}{4} = 36$$

$$C_{in,b} = \frac{4/3 \times 36}{4} = 12$$

$$C_{in,a} = \frac{4/3 \times 12}{4} = 4$$

Check work by verifying input cap spec is met!



$$C_{in} = \frac{2}{4} \times 36$$

$$C_{in} = 18$$

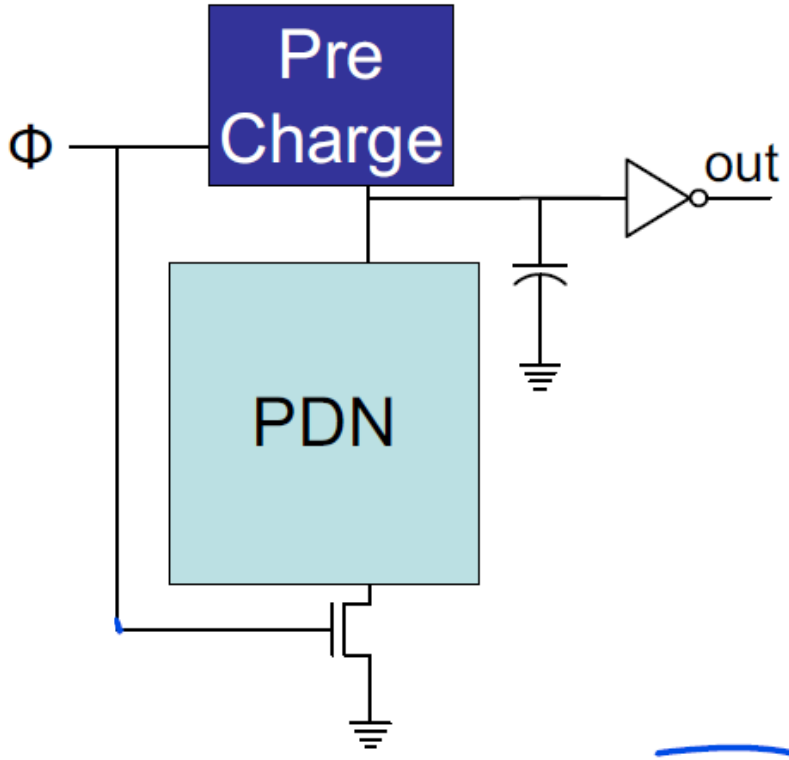
作业题

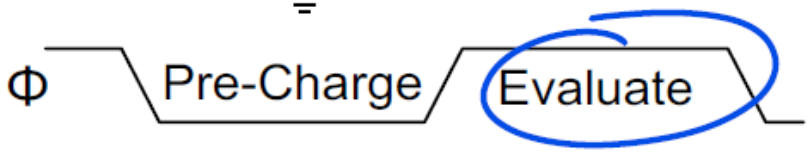
目 录

CONTENTS

基本的Domino逻辑门电路

- 利用时钟将逻辑计算分为预充电和计算两步



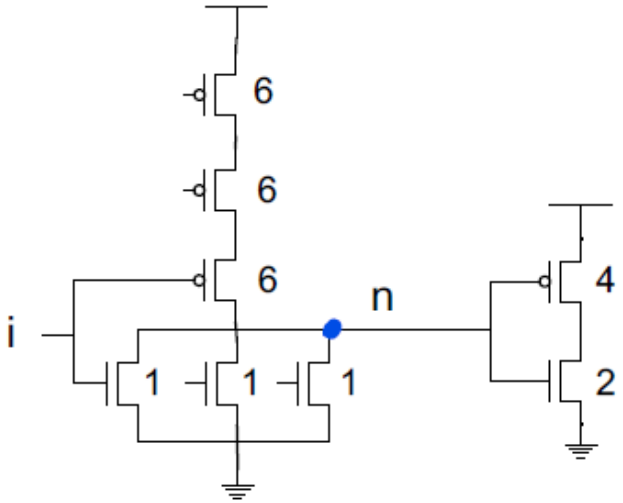


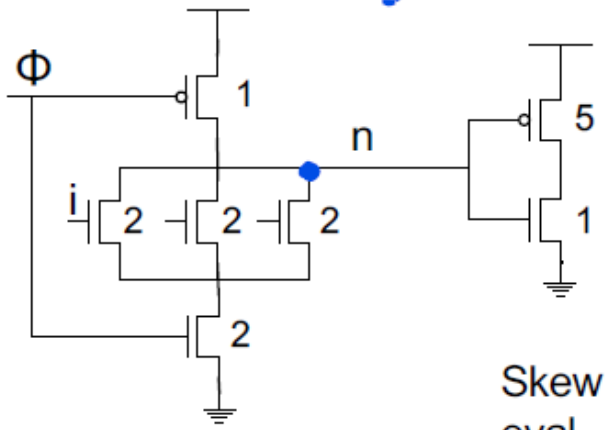
- Divide the clock in 2 phases:
  - Precharge
    - Output Low
    - Dyn. cap precharge
    - PDN Off
  - Evaluate
    - Conditional discharge
    - Input must be stable and monotonic L→H

Domino逻辑门电路的好处

- 对于复杂的电路逻辑，能够大幅降低Logical Effort从而降低电路延迟

3-input OR gate





Skew for eval

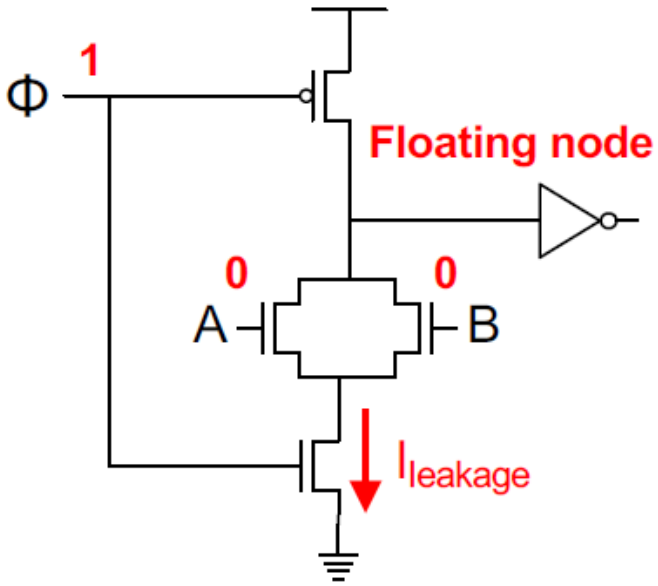
- Advantages:
  - Faster than CMOS
  - Input capacitance is lower
  - Early switch point
  - Inverter P/N > 2 (only rising delay important)

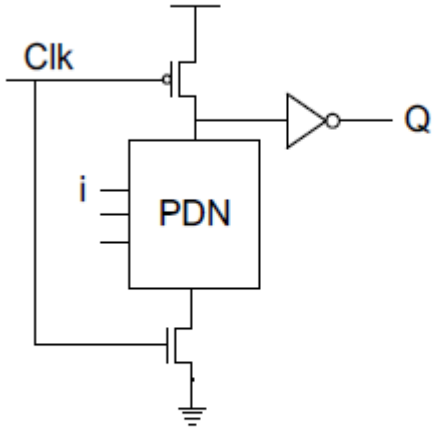
Static:
$C_i = 7$
$C_n = 9$

Dynamic:
$C_i = 2$
$C_n = 7$

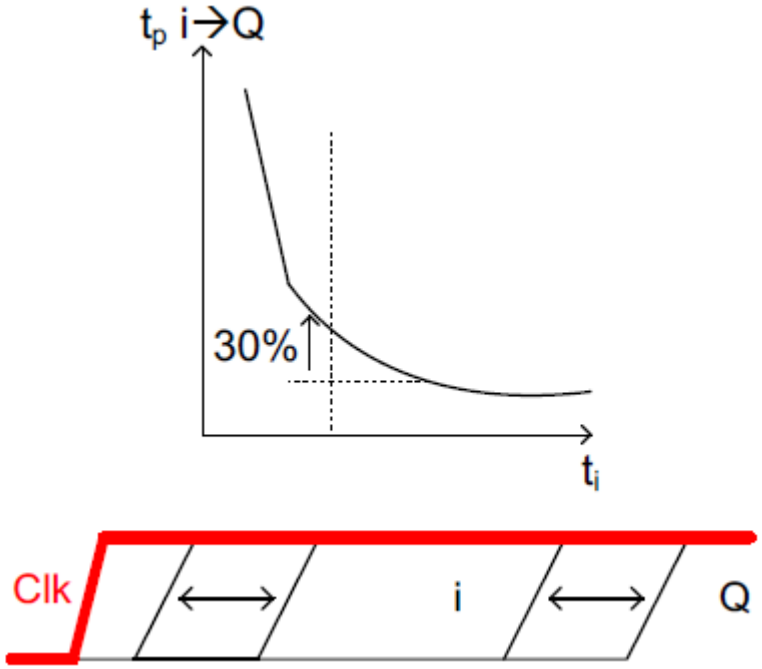
Domino逻辑门电路的问题

- 漏电流问题（Leakage）和输入保持时间问题





First gate in domino chain ↑ delay



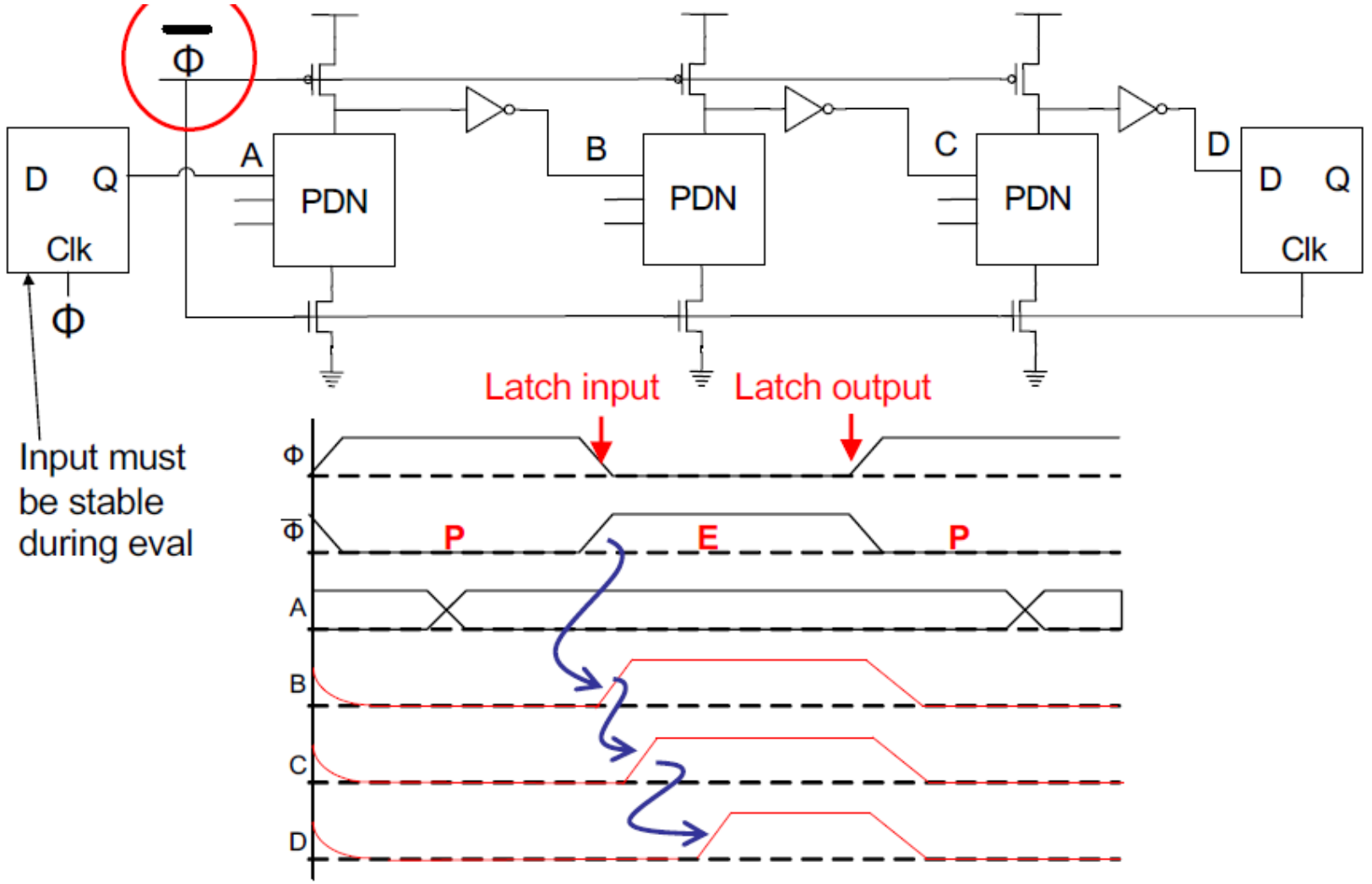
Dynamic node is floating during evaluation

– Leakage current of NMOS can discharge it

输入保持时间需要够长

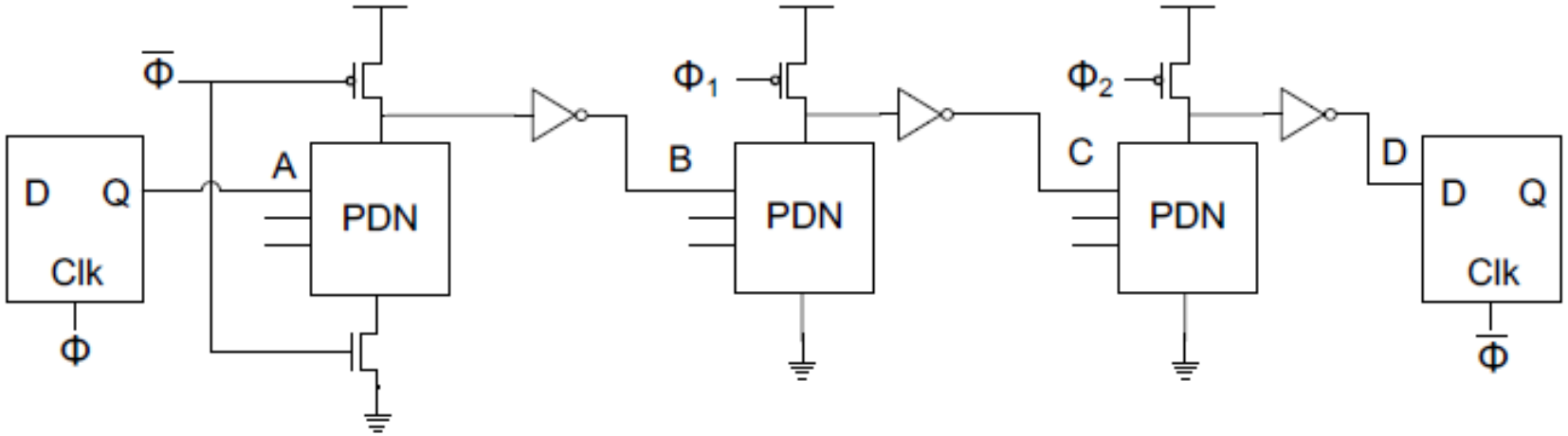
Domino Cascading

- Domino逻辑门的多级互连



Footless Domino

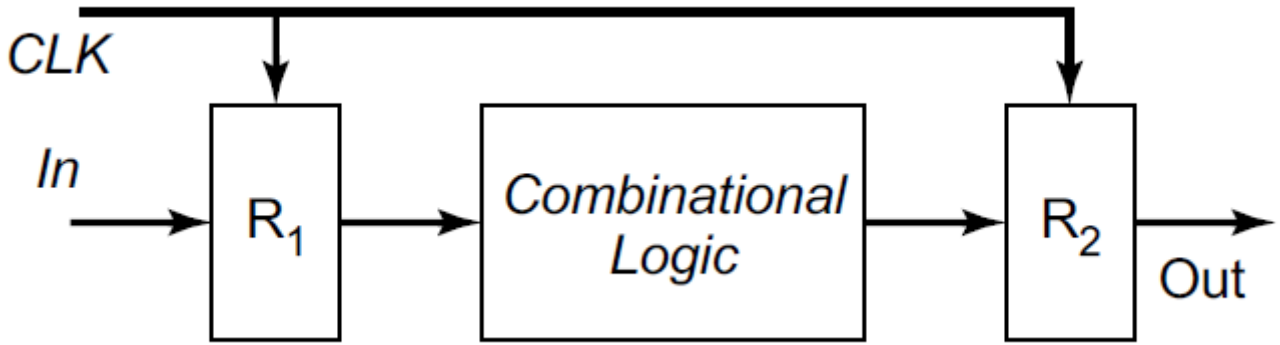
- Domino逻辑门的多级互连



- Advantages:
  - Faster than classic domino → one less NMOS
    - Why is footer necessary in the first stage?
    - Input not guaranteed during precharge

电路时序的基本概念

- 同步时序（Synchronous Timing）

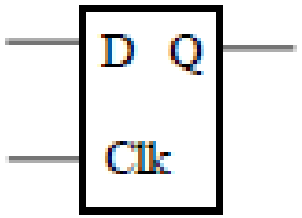


触发器
（Register）

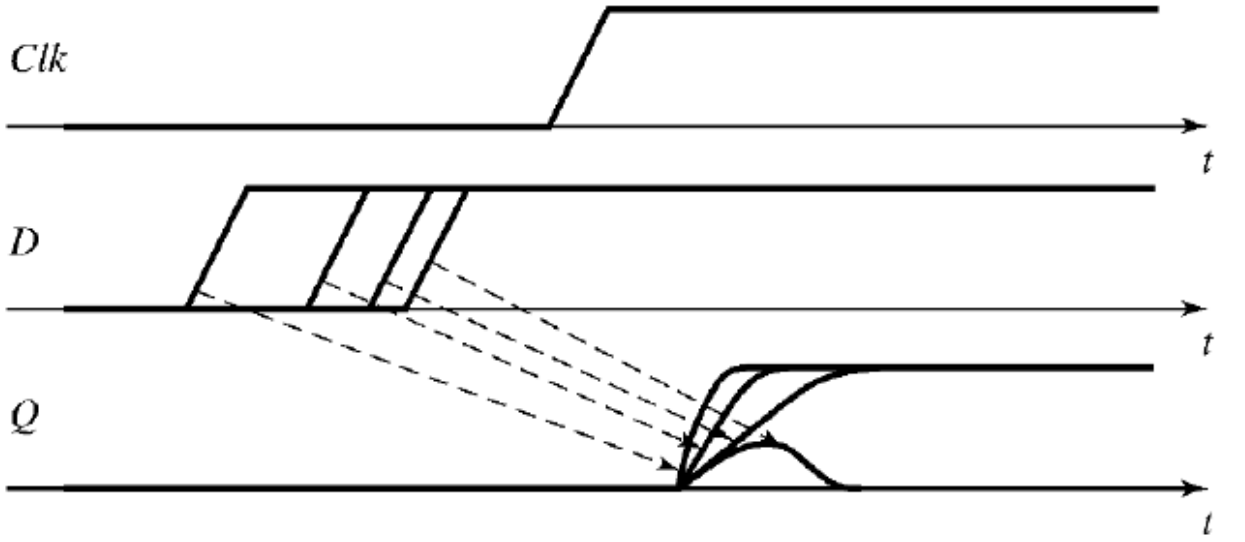
组合逻辑（各种逻辑门电路）

电路时序的基本概念

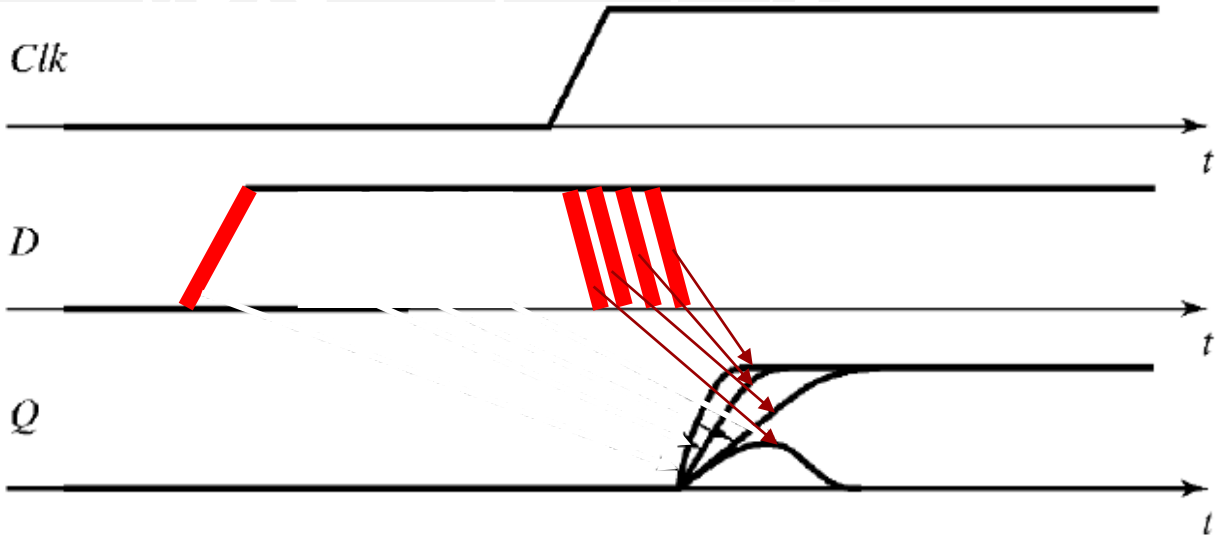
- 触发器的Setup Time和Hold Time



Setup Time

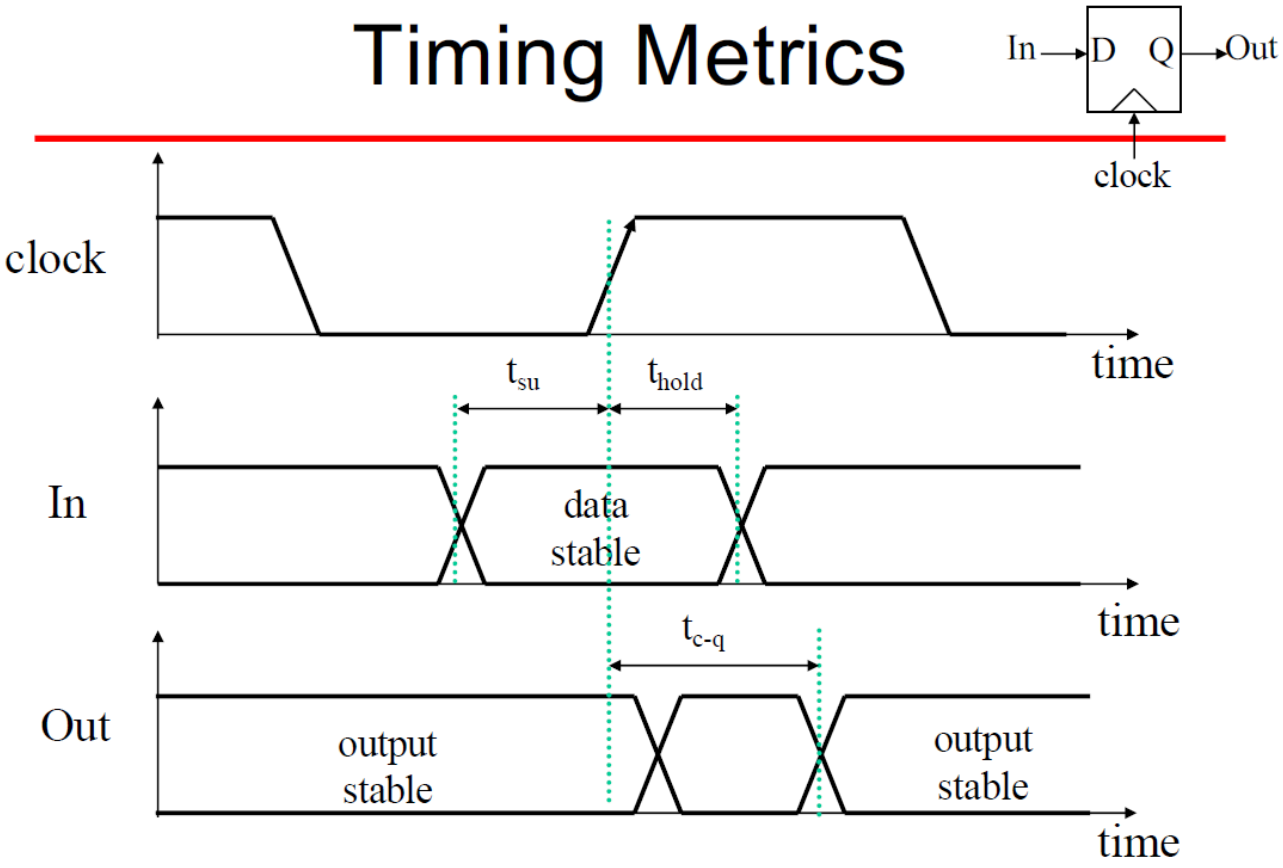


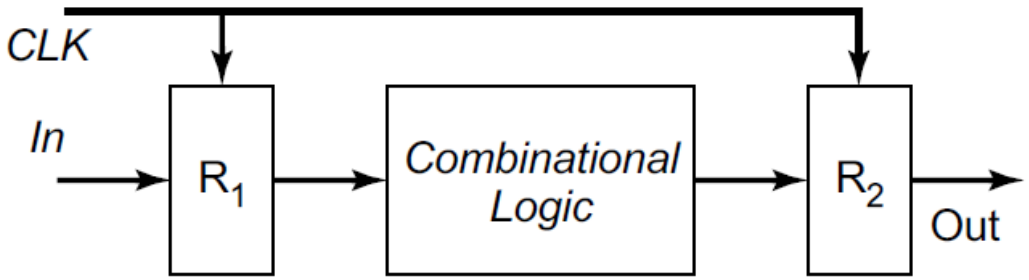
Hold Time



电路时序的基本概念

- 同步时序（Synchronous Timing）





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

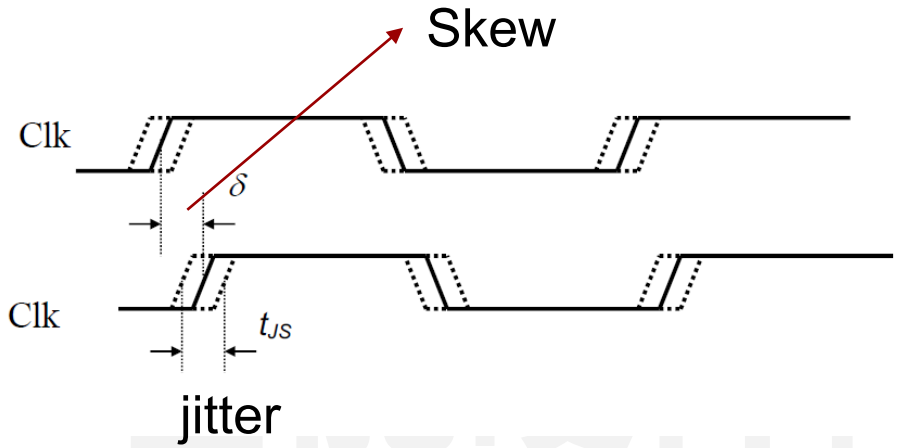
min

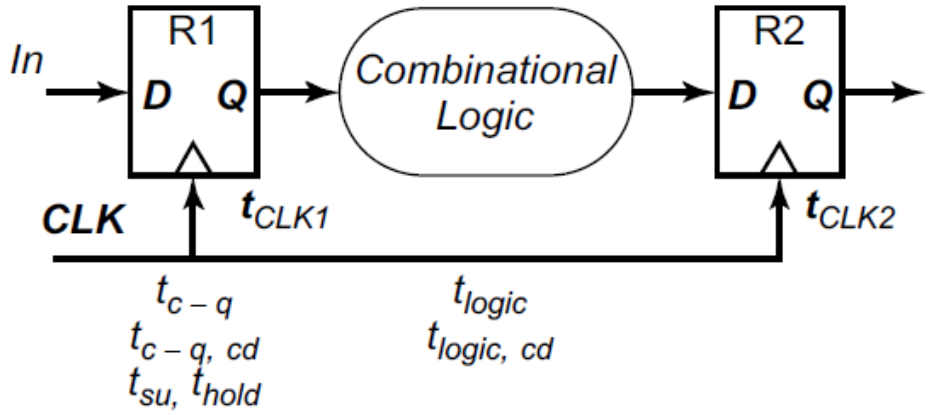
$$T \geq t_{c\text{-}q} + t_{plogic,max} + t_{su}$$

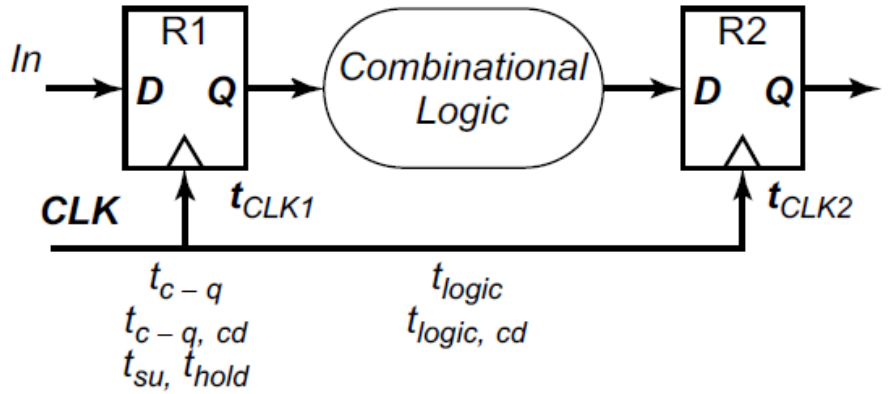
max

电路时序的基本概念

- 时钟的不稳定性







Minimum cycle time:

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

作业题

Hold time constraint:

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

Worst case is when receiving edge arrives early (negative δ)

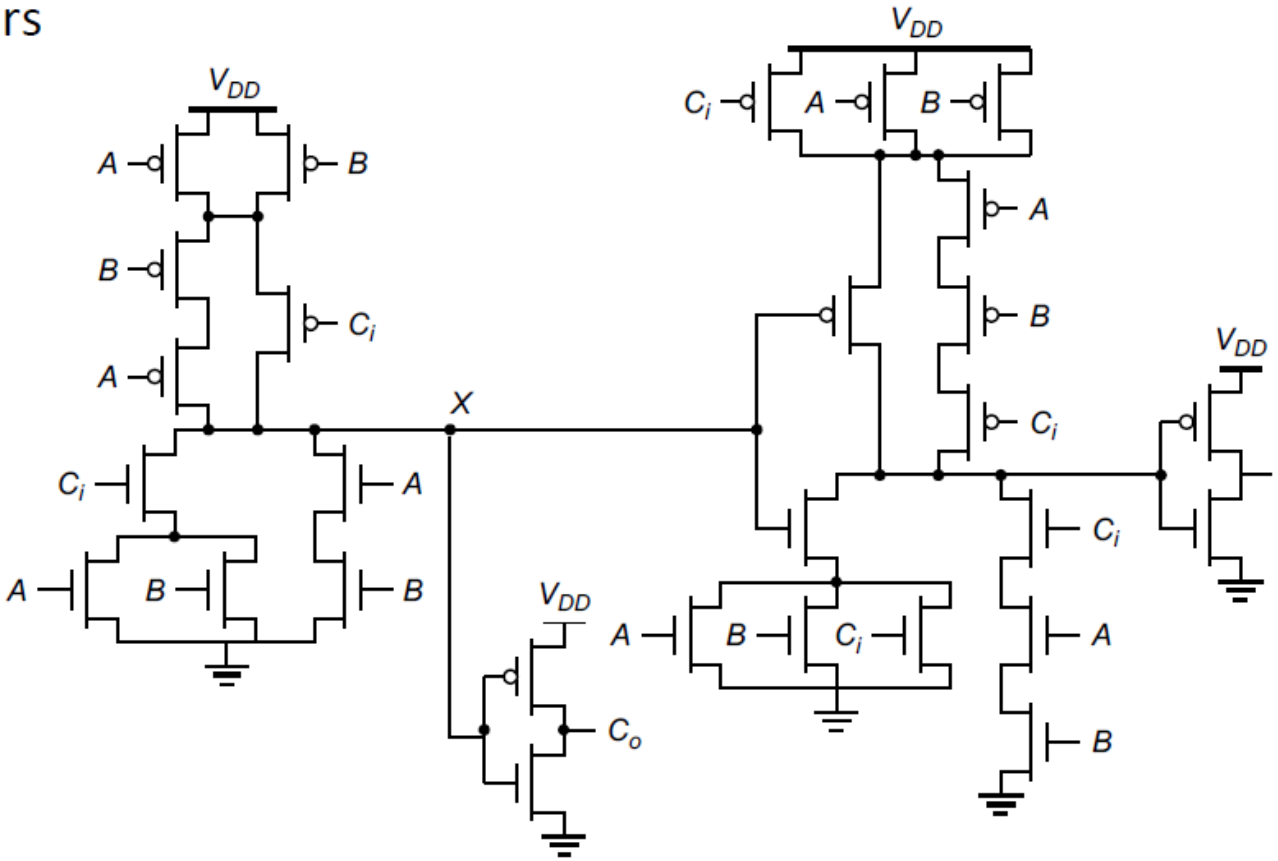
Worst case is when receiving edge arrives late (positive skew)
Race between data and clock
cd: contamination delay (fastest possible delay)

目 录

CONTENTS

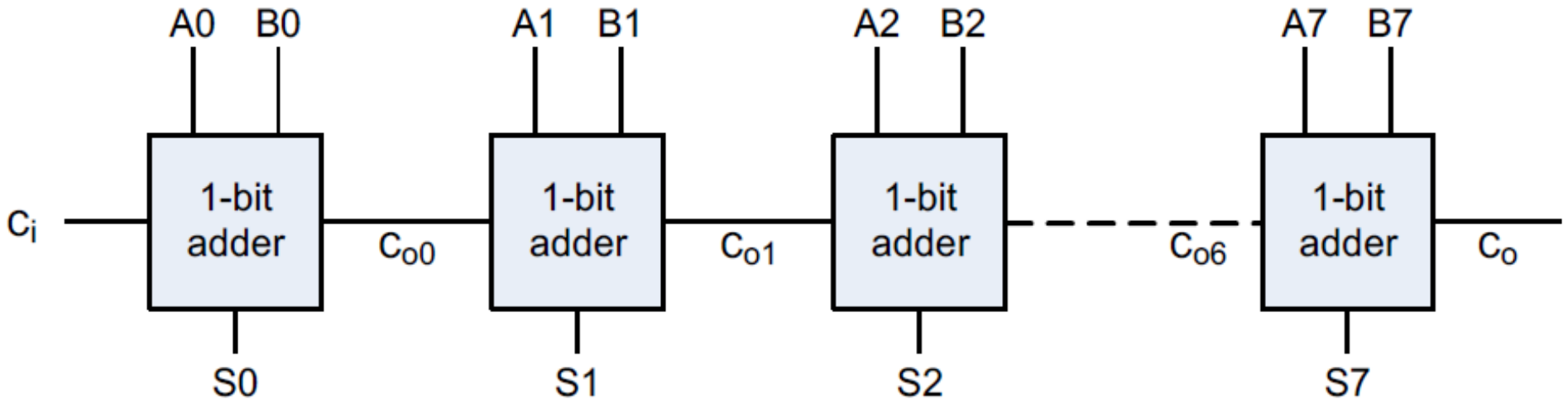
加法器设计

- 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



Worst case delay linear with the number of bits

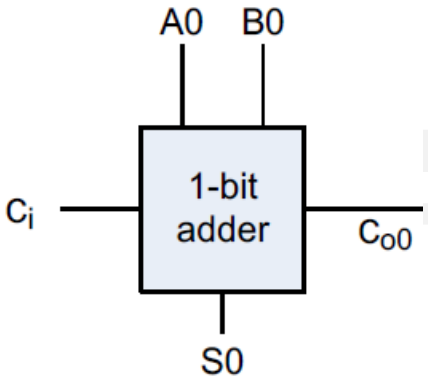
$$t_d = O(N)$$

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

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

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

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

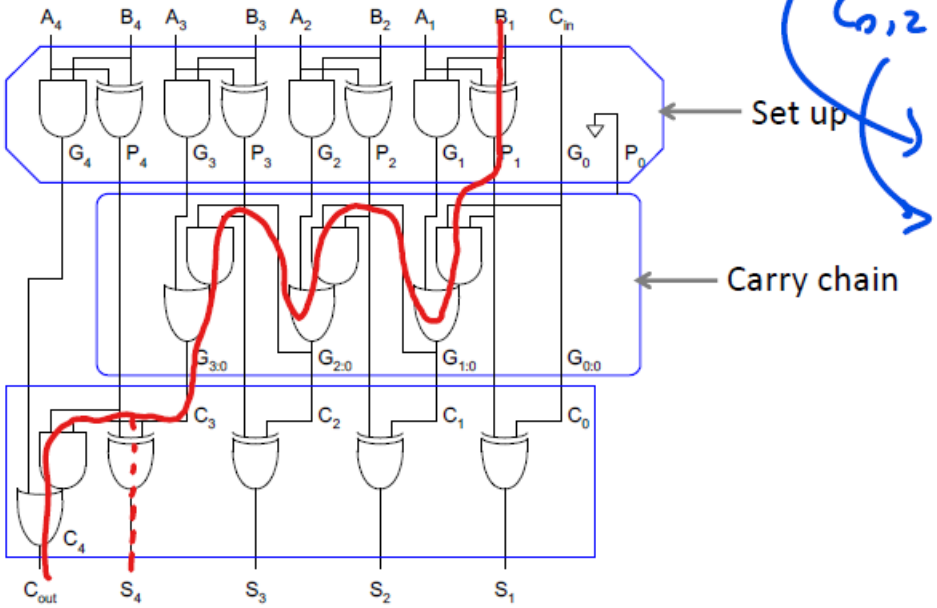
$P = A \oplus B$

$P = A + B$

- 基于PGK的加法器设计方法

Carry-Ripple using P and G

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



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

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

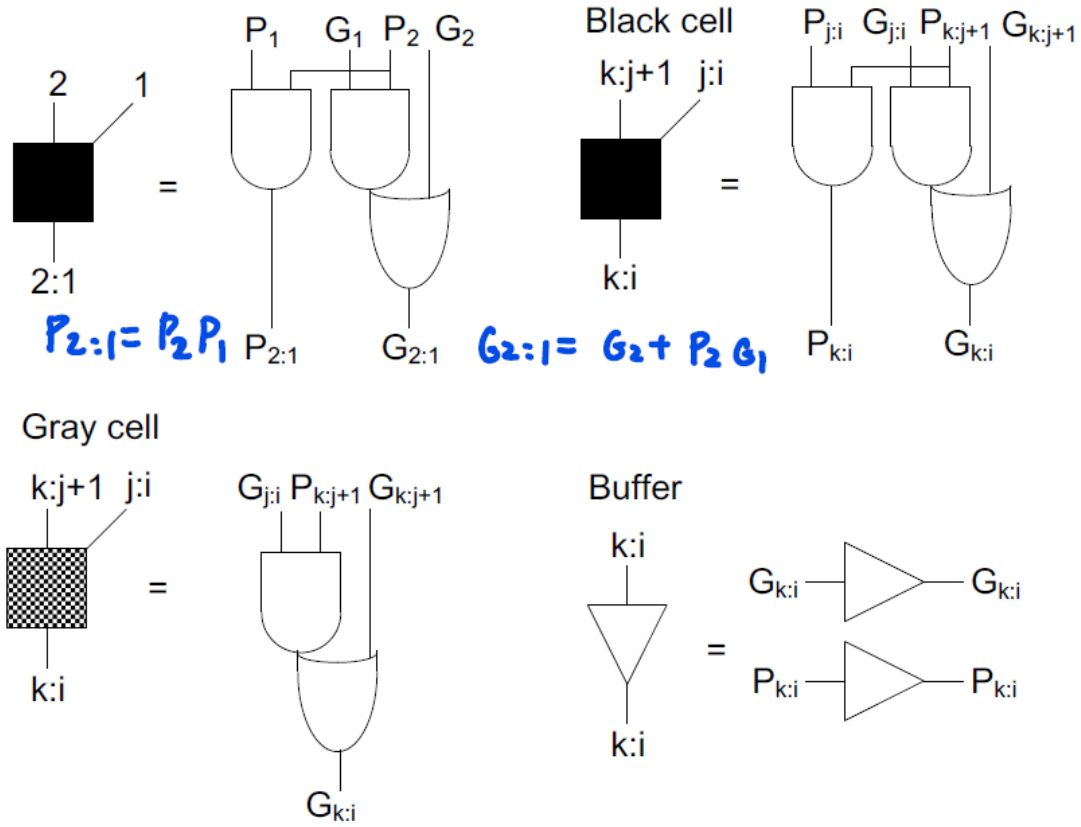
$G_{1:0} = G_1 + P_1 G_0$

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

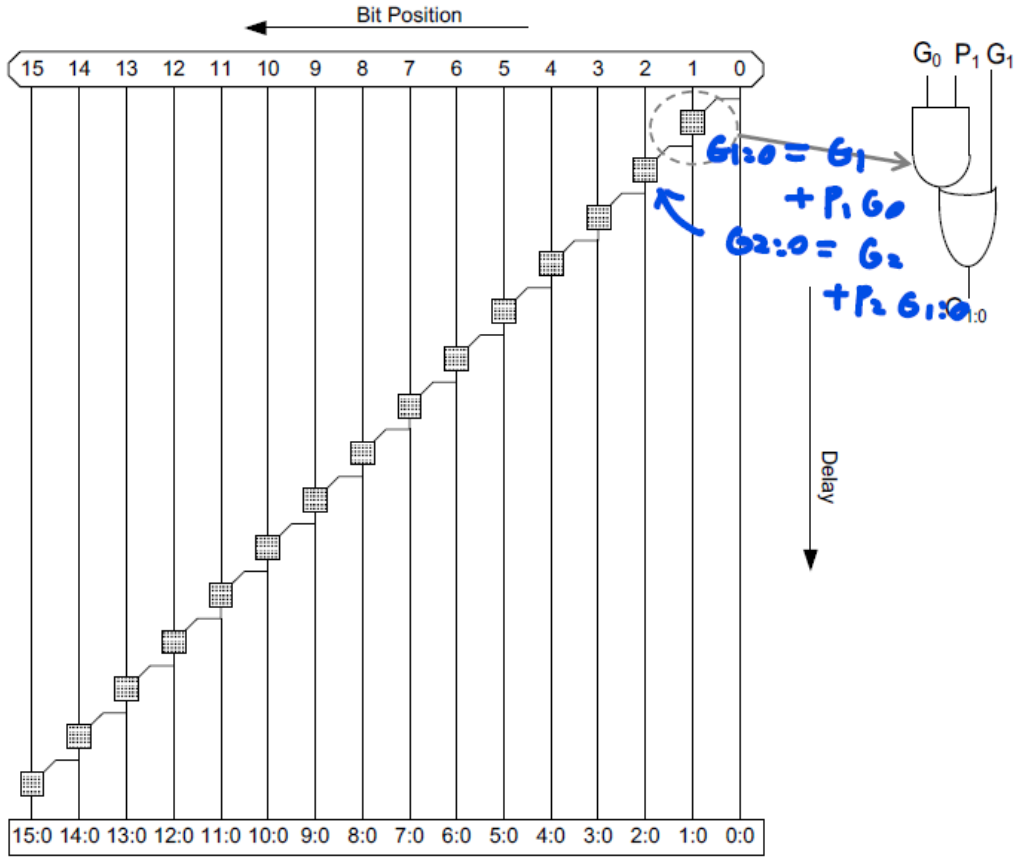
$t_{adder} = t_{setup} + (N-1)\, t_{carry} + \max(t_{carry}, t_{sum})$

加法器设计

- 基于PGK的加法器设计方法



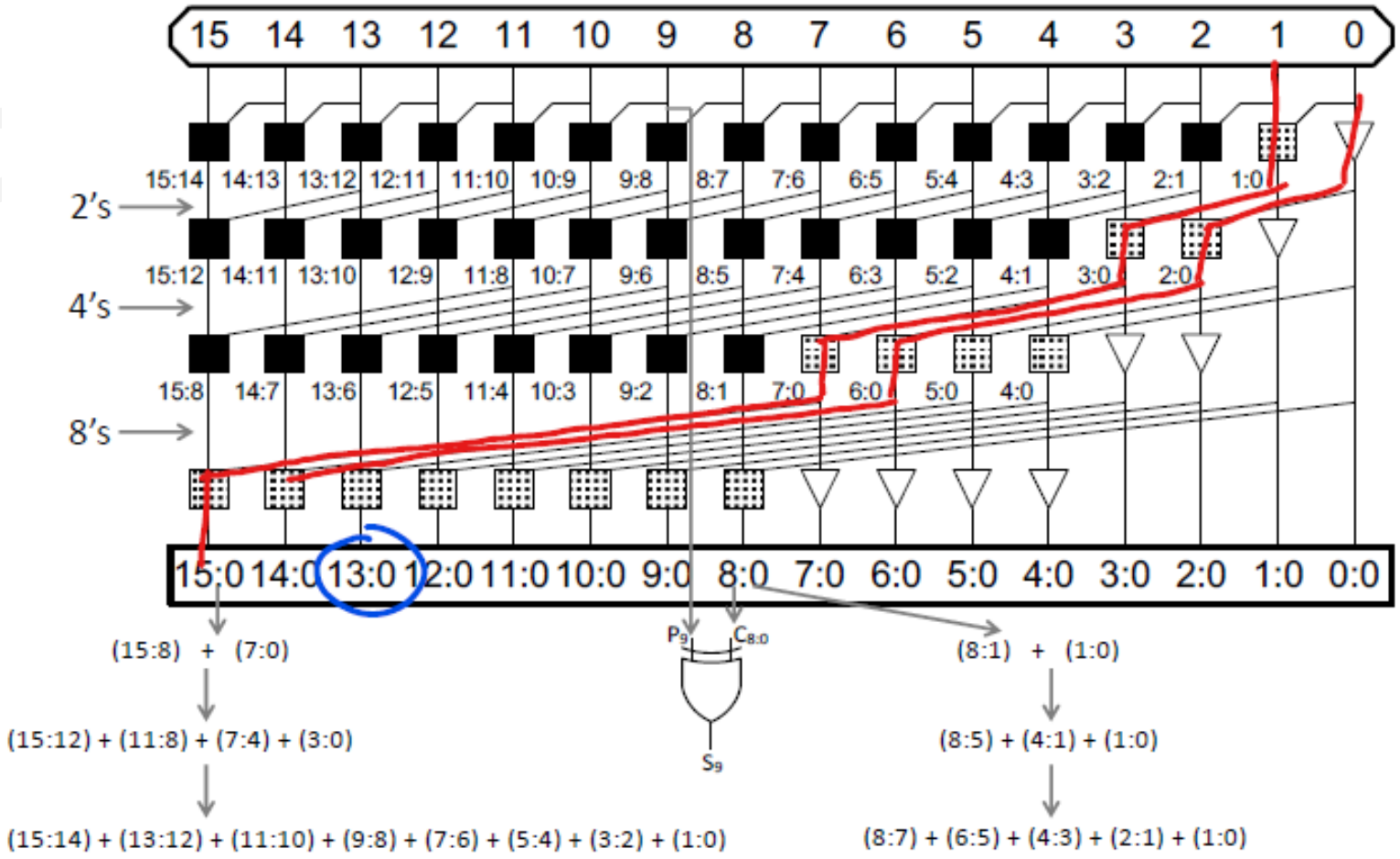
PG生成逻辑



Carry Ripple的PG图

加法器设计

- 基于PGK的加法器设计方法 – 复杂PG树加法器



$\log_2(N)$

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

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

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

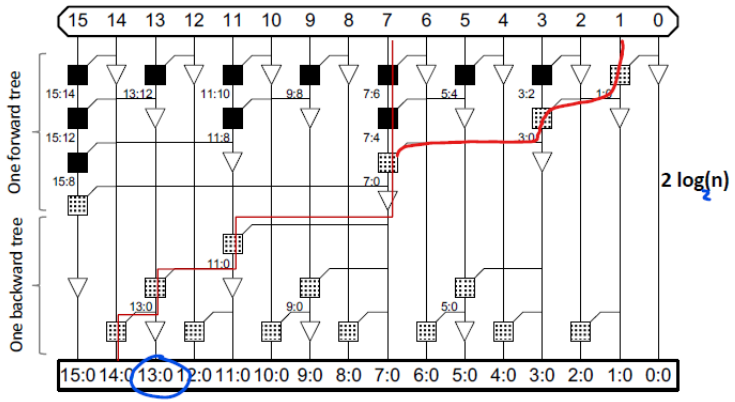
$G_{5:0} = G_{5:2} + P_{5:2}\, G_{1:0}$

作业题

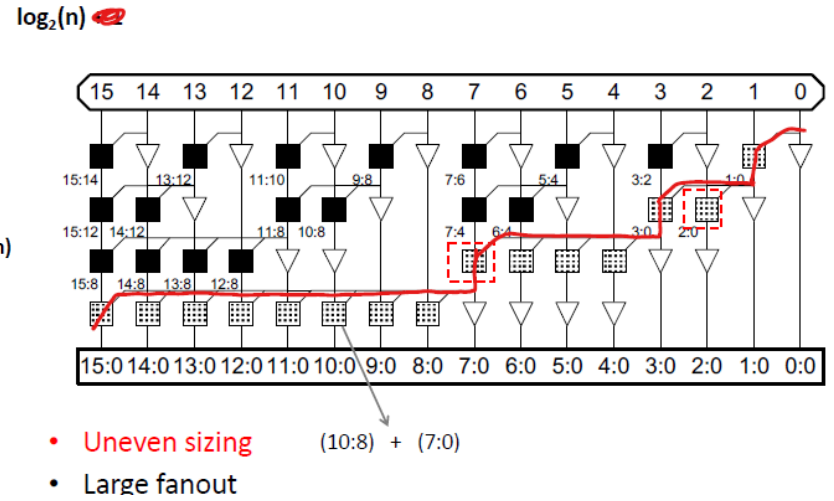
加法器设计

- 基于PGK的加法器设计方法 – 复杂PG树加法器

Brent-Kung

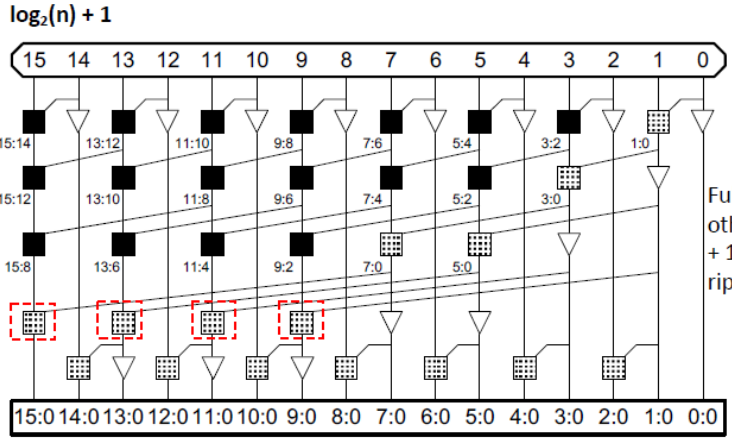


Sklansky



- Uneven sizing (10:8) + (7:0)
- Large fanout

Han-Carlson



Low fanout, tradeoff between logic levels and wiring
Reduces wire length by half ! → half power compared to Kogge Stone

- 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
  1100                       products
 0000
--------
00111100  :  60(10)          product
```

M x N-bit multiplication

- Produce N M-bit partial products
- Sum these to produce M+N-bit product

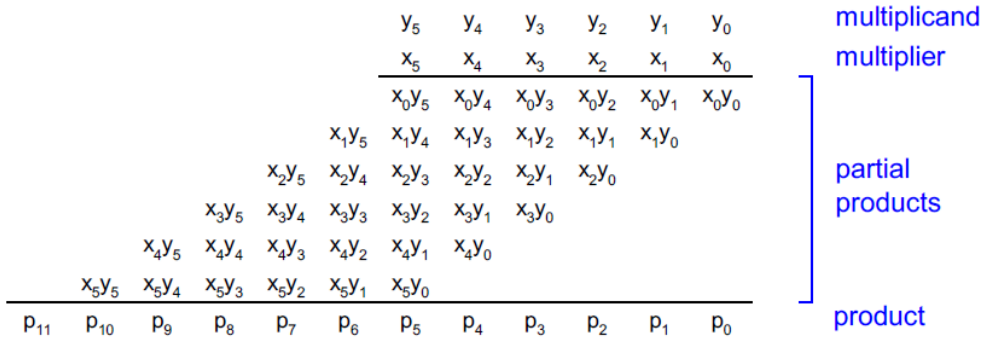
乘法器设计

- 乘法器设计的核心是部分和累加

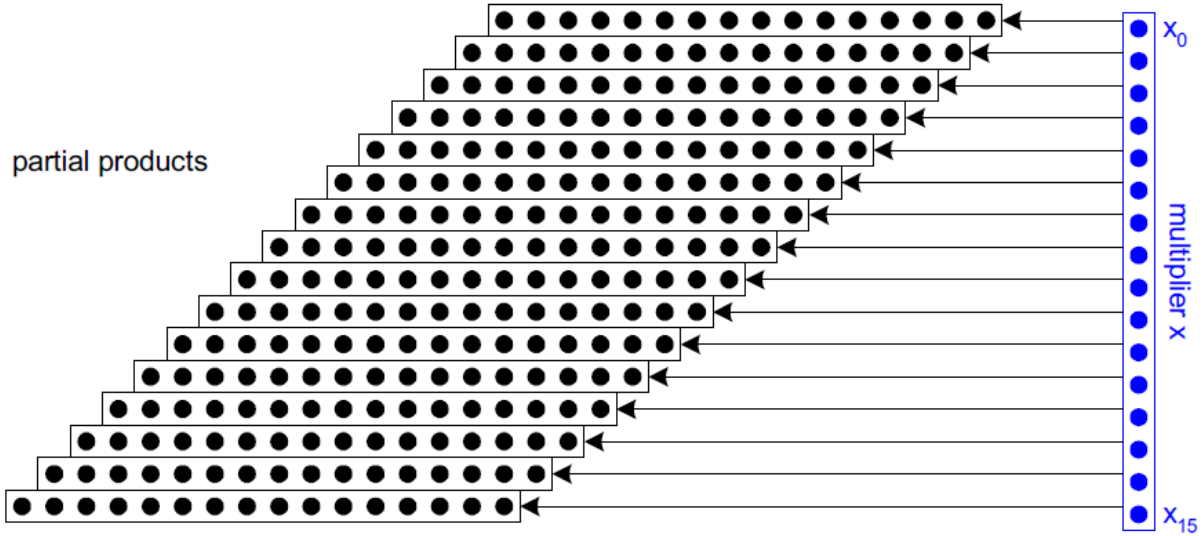
Multiplicand: $Y = (y_{M-1}, y_{M-2}, \ldots, y_1, y_0)$

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

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

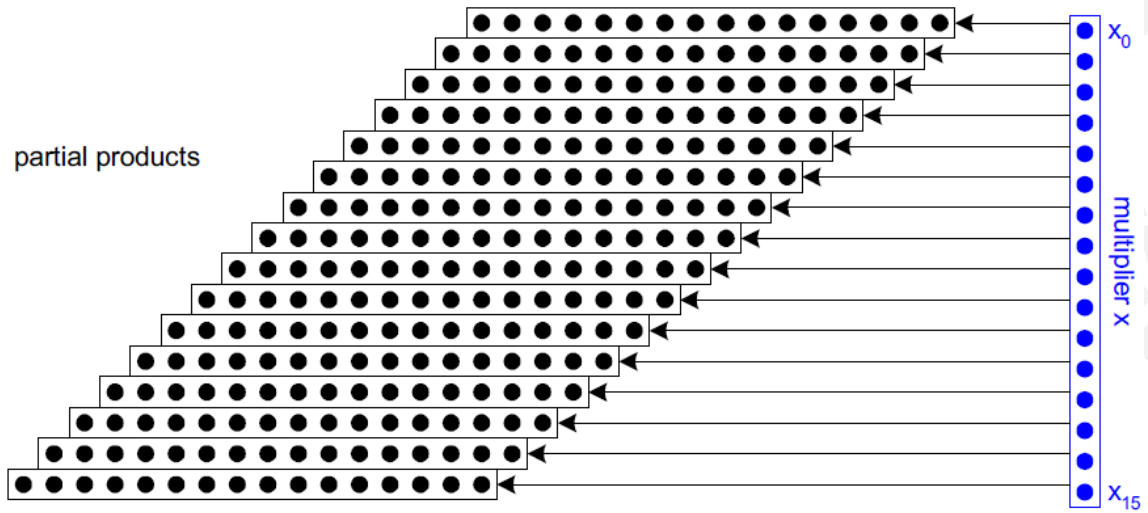


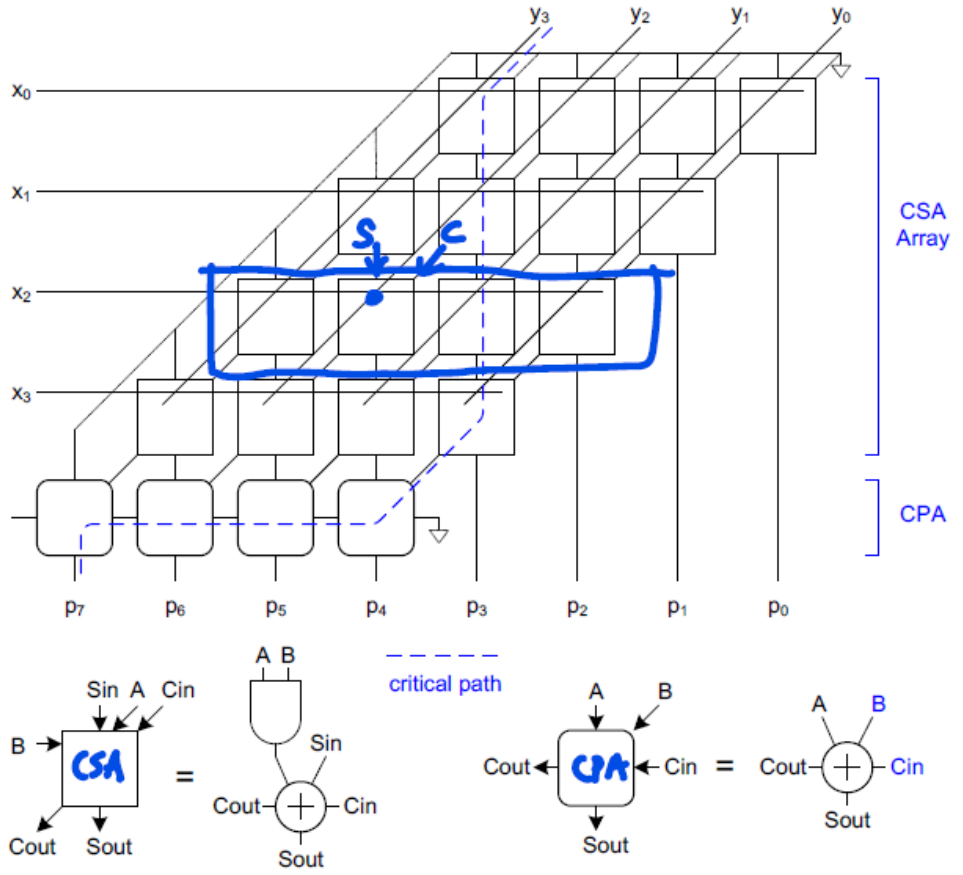
Each dot represents a bit



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 (0 0) (0 1) (1 0) (1 1)

PP 0 Y 2Y (4Y-2Y) 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

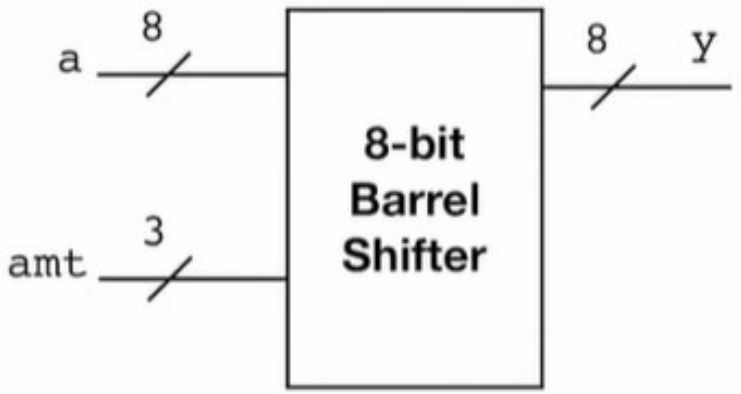
作业题

位移器设计

- Shifter也是重要的数字电路模块之一

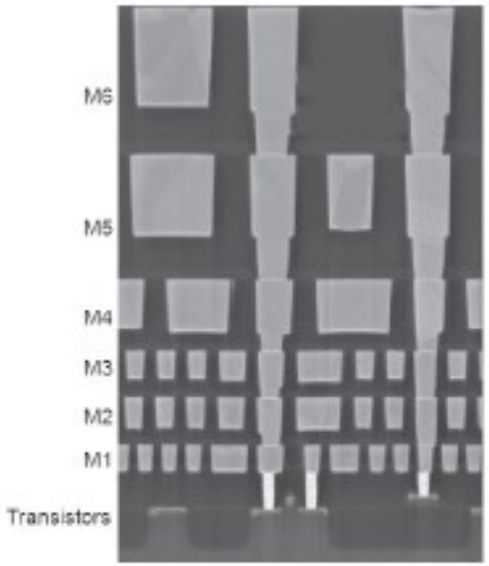
```
module barrel_shifter
    (
        input logic [7:0] a,
        input logic [2:0] amt,
        output logic [7:0] y
    );

    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

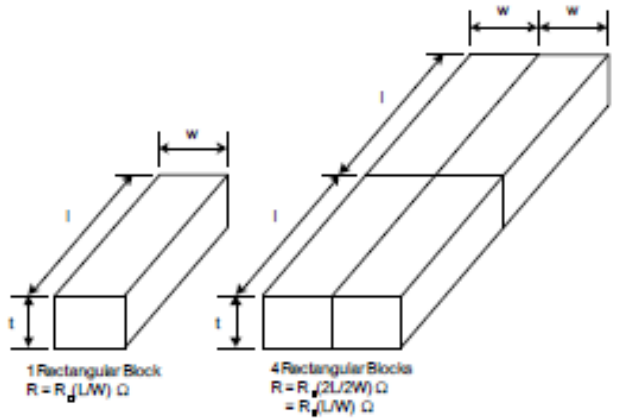
Intel 45 nm Stack

线电阻的计算方式

- ρ = *resistivity* (Ω*m)

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

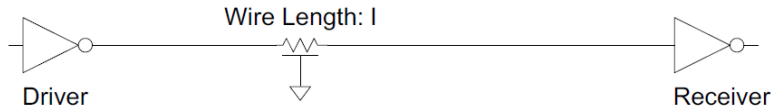
- $R_{\square}$ = *sheet resistance* (Ω/□)
  - □ is a dimensionless unit(!)
- Count number of squares
  - $R = R_{\square} * (\#\ of\ squares)$

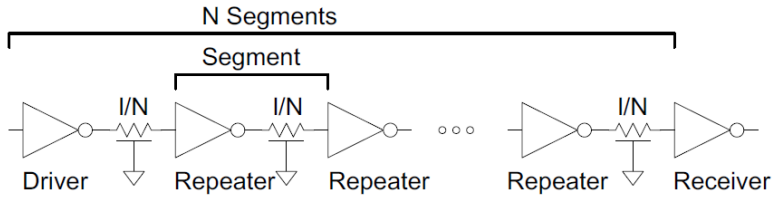


北京大学 PEKING UNIVERSITY

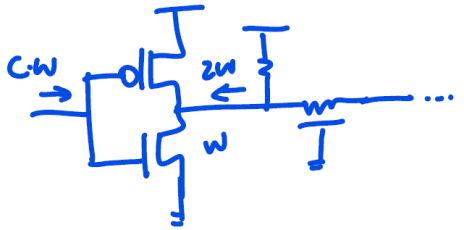
• Wire Repeaters

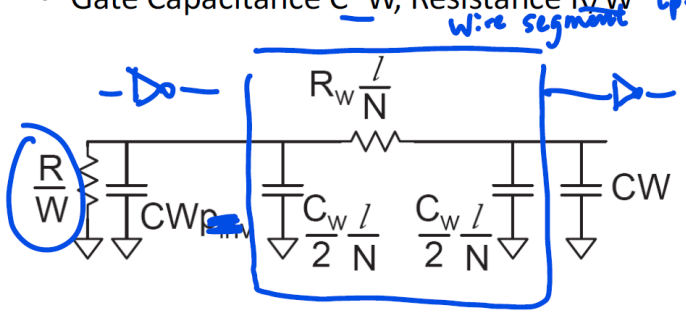
- R and C are proportional to l
- RC delay is proportional to l^2
  - Unacceptably great for long wires
- Break long wires into N shorter segments
  - Drive each one with an inverter or buffer





- 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





$$t_p = N\left(0.69\left(\frac{R}{W} CW + \frac{R}{W}\frac{CW}{2}\frac{1}{N} + \left(\frac{R}{W} + R_w \frac{1}{N}\right)\left(\frac{CW}{2}\frac{1}{N} + CW\right)\right)\right)$$

- 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