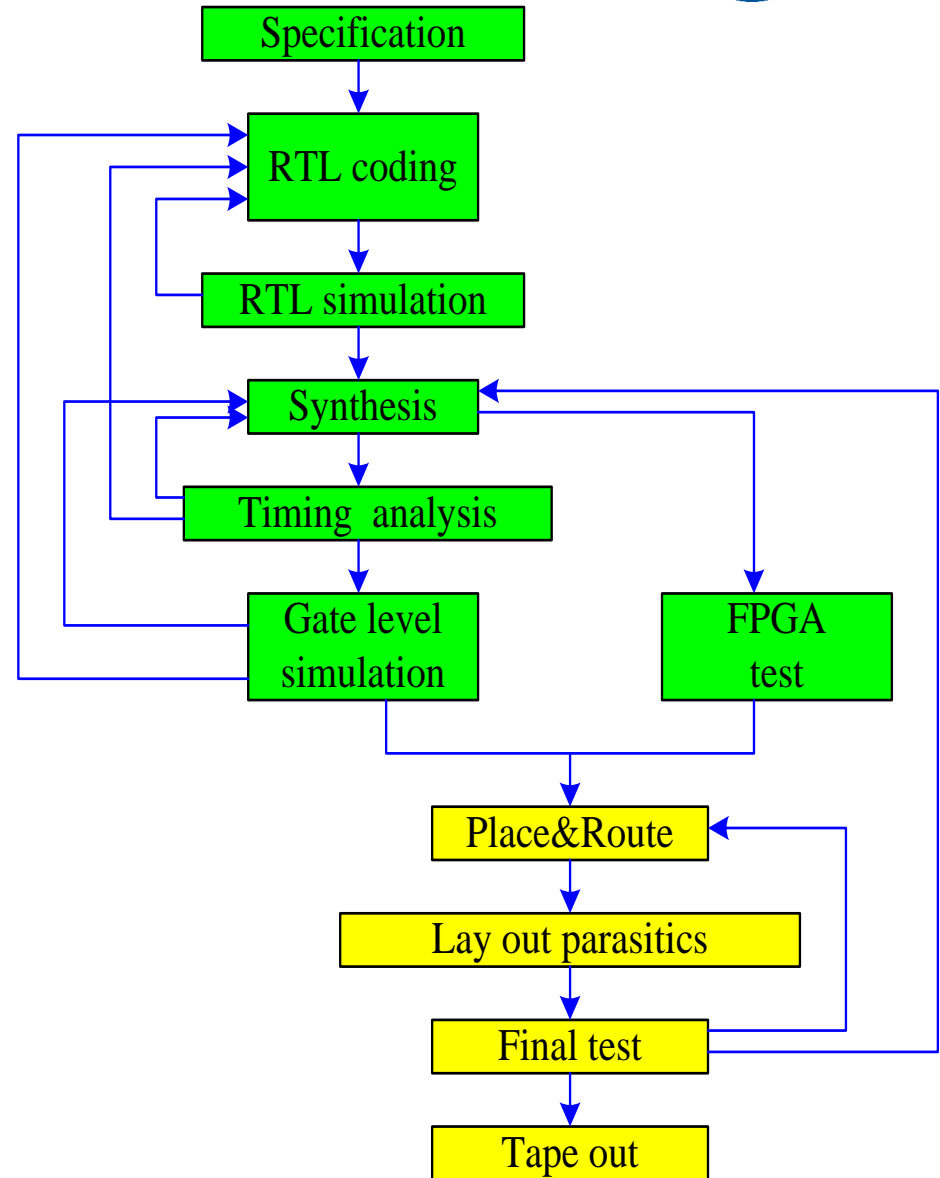


Typical IC Design Flow



- ✓ 组合逻辑设计
 - ✓ 时序逻辑设计
 - ✓ 硬件描述语言
 - ✓ 数字模块
- 最简单也最实用！



第二章 组合逻辑设计



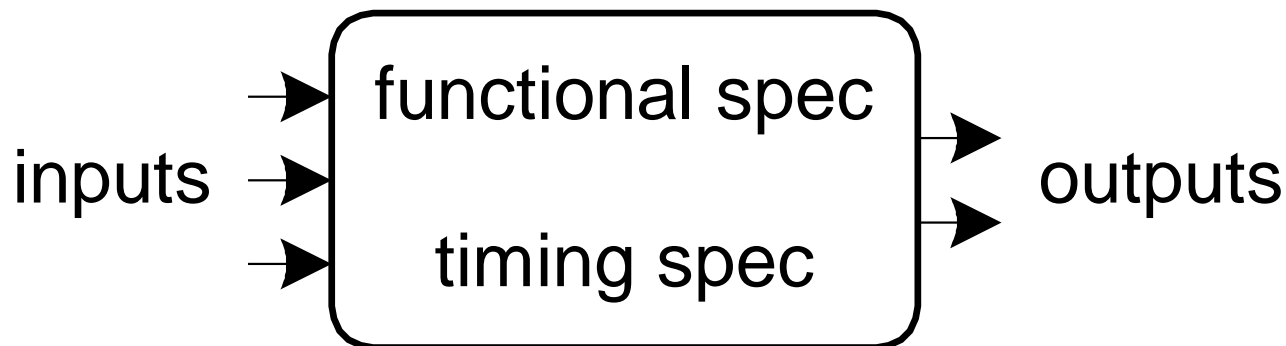
- 介绍
- 布尔表达式
- 布尔代数
- 从逻辑到门
- 多级组合逻辑
- X和Z
- 卡诺图
- 组合逻辑模块
- 时序

Application Software	>"hello world!"
Operating Systems	
Architecture	
Micro-architecture	
Logic	
Digital Circuits	
Analog Circuits	
Devices	
Physics	

数字电路（数字逻辑电路）



- 一个**逻辑**电路的组成
- 输入
- 输出
- 功能规范
- **时序规范** (! ! !)

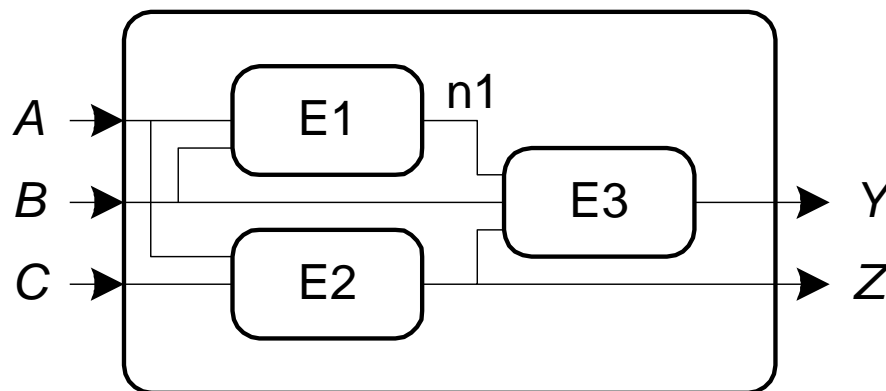


数字电路



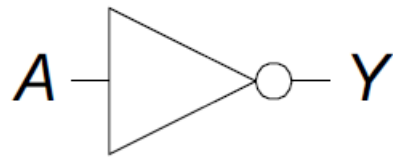
- 结点
 - Inputs: A, B, C
 - Outputs: Y, Z
 - Internal: $n1$
- 电路元件
 - $E1, E2, E3$
 - Each a circuit

搭积木



- 逻辑门对二进制变量进行逻辑运算，是最简单的数字电路
 - NOT, AND, OR, NAND, NOR, etc.
- 单输入
 - NOT gate, 缓冲器 buffer
- 双输入
 - AND, OR, XOR, NAND, NOR, XNOR
- 多输入

NOT

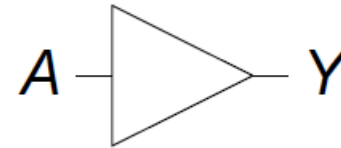


$$Y = \overline{A}$$

A	Y
0	1
1	0

真值表

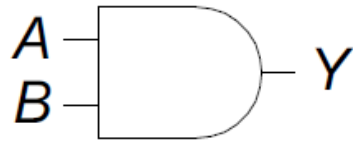
BUF



$$Y = A$$

A	Y
0	0
1	1

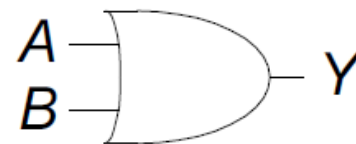
AND



$$Y = AB$$

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

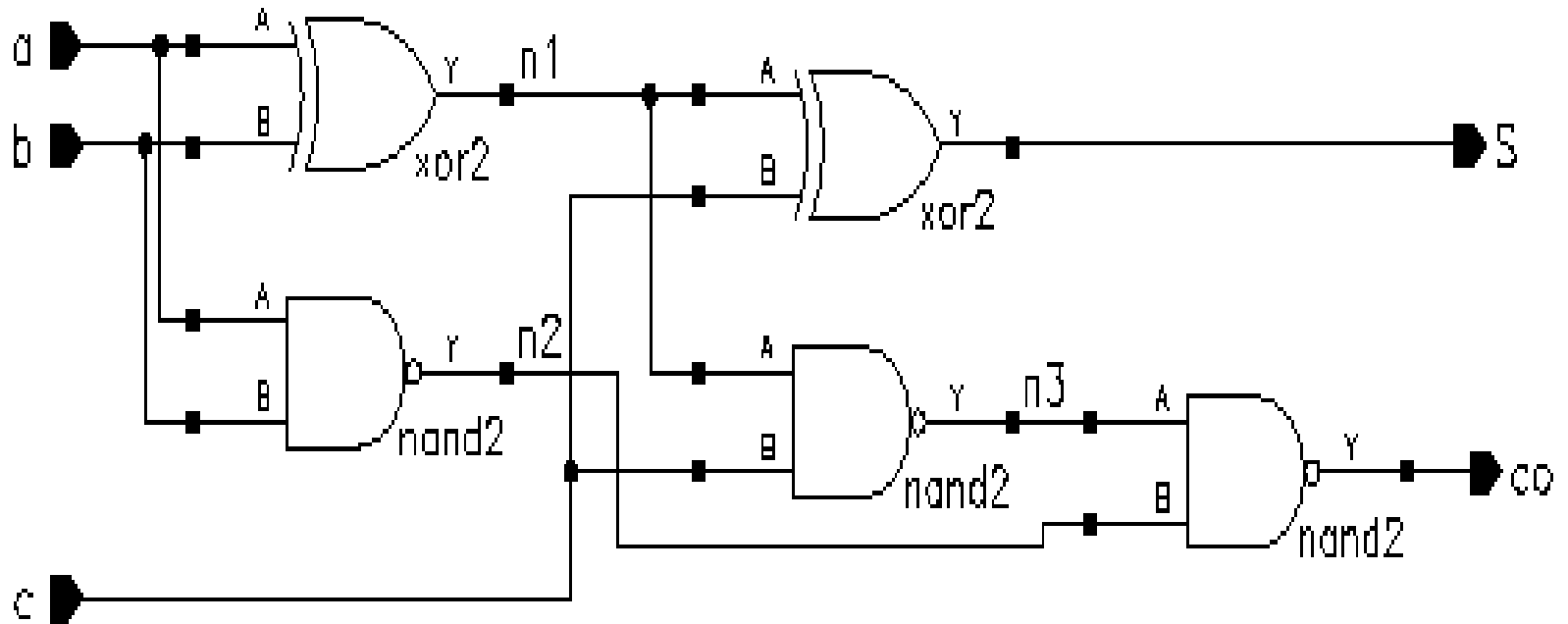
OR



$$Y = A + B$$

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

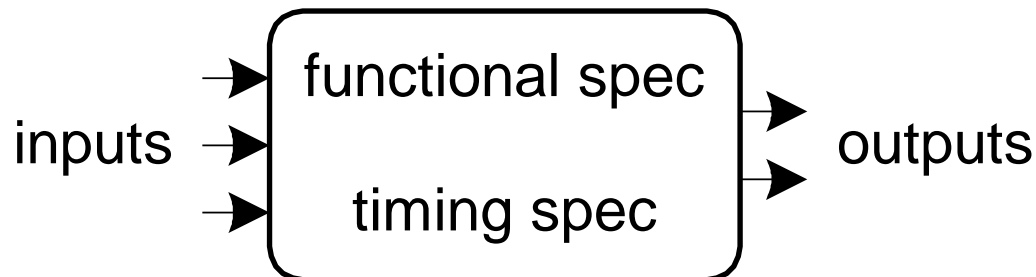
用门单元实现以下电路(写出TestBench)



逻辑电路类型



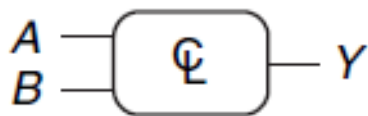
- 组合逻辑电路
 - 无记忆
 - 电路输出仅仅取决于输入值
- 时序逻辑电路
 - 有记忆功能
 - 输出取决于当前与之前输入值



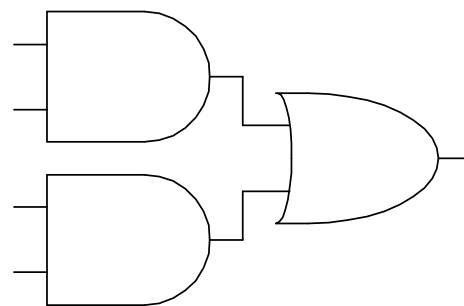
组合逻辑电路构成

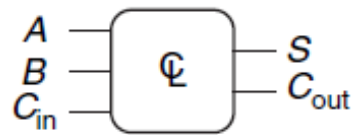
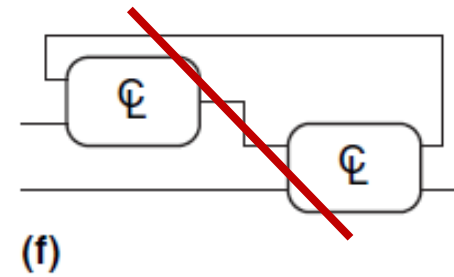
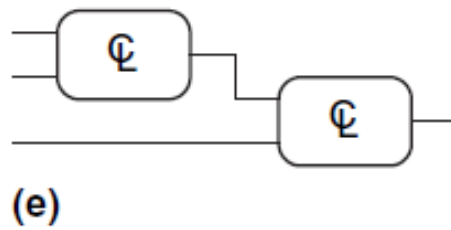
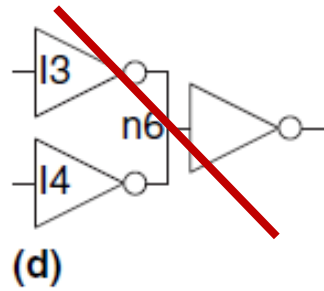
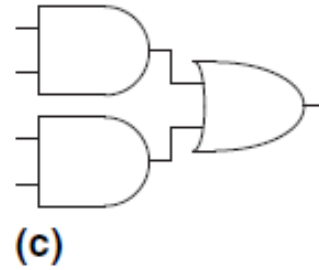
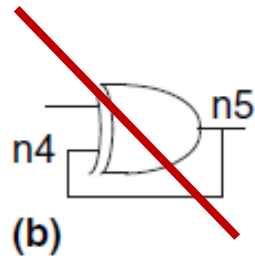
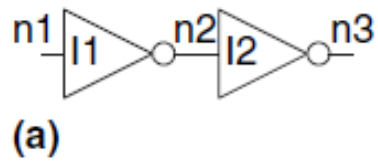


- 每一个电路元件本身是组合电路
- 每一个电路结点或者是一个电路的输入、或者是连接外部电路的一个输出端
- 电路不包含回路
- **Example:**



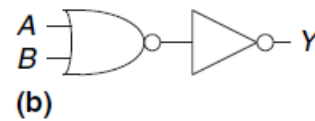
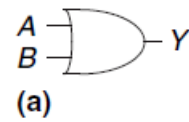
$$Y = F(A, B) = A + B$$





$$S = A \oplus B \oplus C_{in}$$

$$C_{out} = AB + AC_{in} + BC_{in}$$



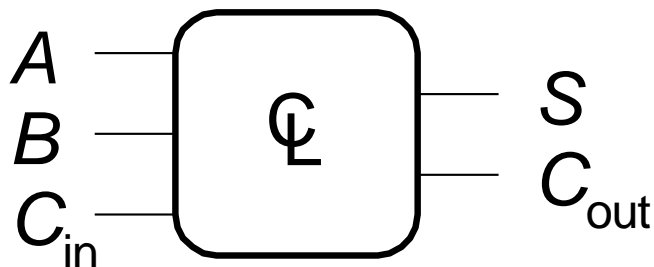
布尔表达式



- 处理的变量是TRUE和FALSE（类似电路）

- 例子: $S = F(A, B, C_{in})$

$$C_{out} = F(A, B, C_{in})$$



$$S = A \oplus B \oplus C_{in}$$

$$C_{out} = AB + AC_{in} + BC_{in}$$

一些定义



- 取反、非:

$\bar{A}, \bar{B}, \bar{C}$

- 项: 变量与它的非

$A, \bar{A}, B, \bar{B}, C, \bar{C}$

- 乘积项 (蕴涵项) : 多个项AND

$ABC, \bar{A}\bar{C}, BC$

- 最小项: 包含全部输入变量的乘积项

$ABC, \bar{A}\bar{B}\bar{C}, ABC$

- 最大项: 包含了全部输入的和

$(A+\bar{B}+C), (\bar{A}+B+\bar{C}), (\bar{A}+\bar{B}+C)$

优先级



- 非优先级最高 (NOT)
- 接下来是与 (AND)
- 最后是或 (OR)

与或式



- 所有的等式都可以用与或式表示
- 每一行给出了最小项

A	B	Y	minterm	minterm name
0	0	0	$\overline{A} \overline{B}$	m_0
0	1	1	$\overline{A} B$	m_1
1	0	0	$A \overline{B}$	m_2
1	1	1	$A B$	m_3

与或式



- 所有的等式都可以用与或式表示
- 每一行给出了最小项
- 可用输出为TRUE的所有最小项之和的形式**写出任意一个真值表的布尔表达式**

A	B	Y	minterm	minterm name
0	0	0	$\bar{A} \bar{B}$	m_0
0	1	1	$\bar{A} B$	m_1
1	0	0	$A \bar{B}$	m_2
1	1	1	$A B$	m_3

与或式



- 所有的等式都可以用与或式表示
- 每一行给出了最小项（**输出Y为TRUE**）
- 可用输出为TRUE的所有最小项之和的形式写出任意一个真值表的布尔表达式
- **由多个积（AND构成的最小项）的和（OR）构成了函数的与或式范式。**
- **同一个真值表总能写出唯一的布尔代数**

A	B	Y	minterm	minterm name
0	0	0	$\bar{A} \bar{B}$	m_0
0	1	1	$\bar{A} B$	m_1
1	0	0	$A \bar{B}$	m_2
1	1	1	$A B$	m_3

$$Y = F(A, B) = AB + \bar{A}B = \Sigma(1, 3)$$

真值表的与或式布尔表达式



<i>A</i>	<i>B</i>	<i>Y</i>
0	0	1
0	1	0
1	0	1
1	1	1

$$Y = \bar{A}\bar{B} + A\bar{B} + AB$$

<i>A</i>	<i>B</i>	<i>C</i>	<i>Y</i>
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

$$Y = \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + \bar{A}BC + A\bar{B}\bar{C} + ABC$$

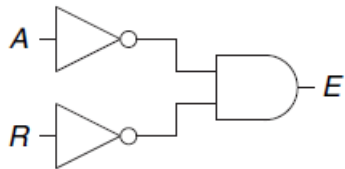
例子：设计电路



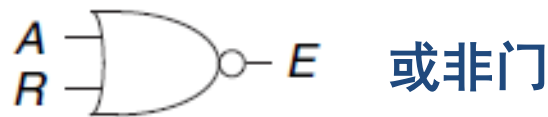
Ben正在野炊，如果下雨或者那儿有蚂蚁，Ben将不能享受野炊。
设计一个电路，只有当Ben野炊时，它输出为TRUE

A	R	E
0	0	1
0	1	0
1	0	0
1	1	0

$$E = \overline{A} \overline{R} \text{ or } E = \Sigma(0)$$



$$R = \overline{A + R}$$



或与式



- 所有布尔表达式也可以写成或与式 (POS)
- 真值表的每一行对应FALSE的一个最大项 (输出Y为FALSE)
- 再将最大式求和, (A=0, B=0时, Y为FALSE)
- 先或后与

A	B	Y	maxterm	maxterm name
0	0	0	$A + B$	M_0
0	1	1	$A + \overline{B}$	M_1
1	0	0	$\overline{A} + B$	M_2
1	1	1	$\overline{A} + \overline{B}$	M_3

$$Y = F(A, B) = (A + B)(\overline{A} + B) = \Pi(0, 2)$$

真值表的**或**与式布尔表达式



A	B	Y
0	0	1
0	1	0
1	0	1
1	1	1

与或式 $Y = \bar{A}\bar{B} + A\bar{B} + AB$

或与式 $Y = A + \bar{B}$

A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

与或式

$$Y = \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + \bar{A}B\bar{C} + \bar{A}BC + A\bar{B}\bar{C} + A\bar{B}C + AB\bar{C} + ABC$$

$$Y = \bar{A}\bar{C} + \bar{A}\bar{B} + AC$$

或与式

$$Y = (A+B+\bar{C})(A+\bar{B}+\bar{C})(\bar{A}+\bar{B}+C)$$

例子：设计电路

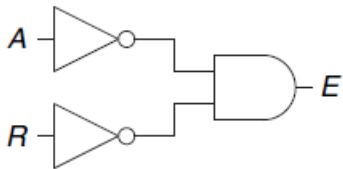


Ben正在野炊，如果下雨或者那儿有蚂蚁，Ben将不能享受野炊。
设计一个电路，只有当Ben野炊时，它输出为TRUE（或与式）

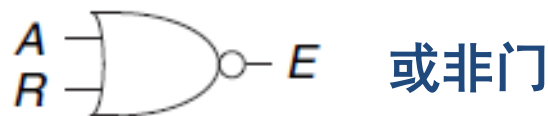
A	R	E
0	0	1
0	1	0
1	0	0
1	1	0

与或式 $E = \bar{A} \bar{R}$ or $E = \Sigma(0)$

或与式 $E = (A + \bar{R})(\bar{A} + R)(\bar{A} + \bar{R})$



$$R = \overline{A + R}$$





- 1) 同一真值表的与或式 与 或与式 的表达是等价的。
- 2) 当真值表只有少数行输出为TRUE是，与或式产生较短等式。否则或与式产生较短等式

例子



- 你去餐厅吃饭但不想吃肉夹馍(**E**)
 - 餐厅开门 (**O**)
 - 餐厅只有肉夹馍 (**C**)
- 写一个真值表决定你是否吃午饭 (**E**).

<i>O</i>	<i>C</i>	<i>E</i>
0	0	0
0	1	0
1	0	1
1	1	0

与或式 & 或与式



O	C	E	minterm
0	0	0	$\overline{O} \overline{C}$
0	1	0	$\overline{O} C$
1	0	1	$O \overline{C}$
1	1	0	$O C$

$$\begin{aligned} E &= O\overline{C} \\ &= \Sigma(2) \end{aligned}$$

O	C	E	maxterm
0	0	0	$O + C$
0	1	0	$O + \overline{C}$
1	0	1	$\overline{O} + C$
1	1	0	$\overline{O} + \overline{C}$

$$\begin{aligned} E &= (O + C)(O + \overline{C})(\overline{O} + \overline{C}) \\ &= \Pi(0, 1, 3) \end{aligned}$$

与或式 & 或与式 课堂练习



58 第2章

A	B	Y	A	B	C	Y	A	B	C	Y	A	B	C	D	Y	A	B	C	D	Y
0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	1
0	1	0	0	0	1	0	0	0	1	0	0	0	0	1	1	0	0	0	1	0
1	0	1	0	1	0	0	0	1	0	1	0	0	1	0	1	0	0	1	0	0
1	1	1	0	1	1	0	0	1	1	0	0	1	1	1	1	0	0	1	1	1
			1	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0
			1	0	1	0	1	0	1	0	1	0	0	1	0	1	0	1	1	1
			1	1	0	0	1	1	0	0	1	1	0	0	0	1	1	0	1	1
			1	1	1	1	1	1	1	1	0	1	1	0	0	1	1	0	1	0
											1	0	0	0	1	1	0	0	0	0
											1	0	0	1	0	1	0	0	1	1
											1	0	1	0	1	0	1	0	1	1
											1	0	1	1	0	1	1	1	0	0
											1	1	0	0	0	1	1	0	0	1
											1	1	0	1	0	1	1	0	1	0
											1	1	1	0	1	1	1	0	1	0
											1	1	1	1	0	1	1	1	0	0
															1	1	1	1	1	1

图 2-80 习题 2.1 和习题 2.3 的真值表

布尔代数



- 简化布尔表达式的公理和定理
- 与普通代数相似，更简单，只有0和1
- 对偶性： 如果符合0和1互换，操作符ANDs 和 ORs互换后任然成立，用(')表示对偶式

布尔代数的公理 (Axiom)



Number	Axiom	Name
A1	$B = 0 \text{ if } B \neq 1$	二进制字段
A2	$\bar{0} = 1$	NOT
A3	$0 \bullet 0 = 0$	AND/OR
A4	$1 \bullet 1 = 1$	AND/OR
A5	$0 \bullet 1 = 1 \bullet 0 = 0$	AND/OR

布尔代数的公理 (Axiom)



Number	Axiom	Name
A1	$B = 0 \text{ if } B \neq 1$	二进制字段
A2	$\bar{0} = 1$	NOT
A3	$0 \bullet 0 = 0$	AND/OR
A4	$1 \bullet 1 = 1$	AND/OR
A5	$0 \bullet 1 = 1 \bullet 0 = 0$	AND/OR

对偶替换 (Dual: Replace) : \bullet with $+$
 0 with 1

布尔代数的公理 (Axiom)



Number	Axiom	Dual	Name
A1	$B = 0 \text{ if } B \neq 1$	$B = 1 \text{ if } B \neq 0$	二进制字段
A2	$\bar{0} = 1$	$\bar{1} = 0$	NOT
A3	$0 \bullet 0 = 0$	$1 + 1 = 1$	AND/OR
A4	$1 \bullet 1 = 1$	$0 + 0 = 0$	AND/OR
A5	$0 \bullet 1 = 1 \bullet 0 = 0$	$1 + 0 = 0 + 1 = 1$	AND/OR

Dual: Replace:

- with +
- 0 with 1

单变量定理



Number	Theorem	Name
T1	$B \bullet 1 = B$	同一性定理
T2	$B \bullet 0 = 0$	零元定理
T3	$B \bullet B = B$	重叠定理
T4	$\overline{\overline{B}} = B$	回旋定理
T5	$B \bullet \overline{B} = 0$	互补定理

单变量定理



Number	Theorem	Name
T1	$B \bullet 1 = B$	同一性定理
T2	$B \bullet 0 = 0$	零元定理
T3	$B \bullet B = B$	重叠定理
T4	$\overline{\overline{B}} = B$	回旋定理
T5	$B \bullet \overline{B} = 0$	互补定理

Dual: Replace: • with +
 0 with 1

单变量定理



Number	Theorem	Dual	Name
T1	$B \bullet 1 = B$	$B + 0 = B$	同一性定理
T2	$B \bullet 0 = 0$	$B + 1 = 1$	零元定理
T3	$B \bullet B = B$	$B + B = B$	重叠定理
T4	$\overline{\overline{B}} = B$		回旋定理
T5	$B \bullet \overline{B} = 0$	$B + \overline{B} = 1$	互补定理

Dual: Replace: • with +
 0 with 1

T1: 同一性定理



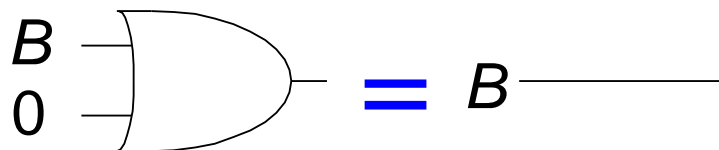
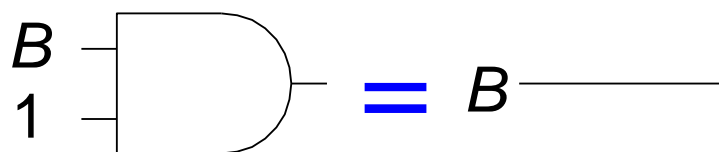
- $\mathbf{B} \cdot \mathbf{1} = \mathbf{B}$

- $\mathbf{B} + \mathbf{0} = \mathbf{B}$

T1: 同一性定理



- $B \cdot 1 = B$
- $B + 0 = B$



一般来说，逻辑门需花费成本、功耗和延迟，用导线来代替门电路更有利

T2: 零元定理



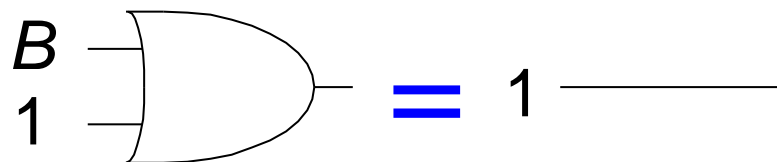
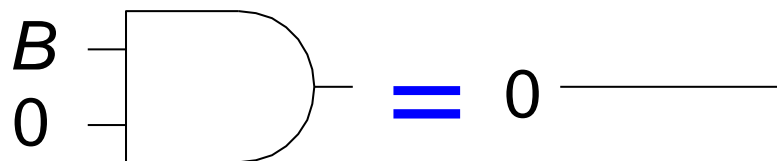
- $\mathbf{B} \cdot \mathbf{0} = \mathbf{0}$

- $\mathbf{B} + \mathbf{1} = \mathbf{1}$

T2: 零元定理



- $B \cdot 0 = 0$
- $B + 1 = 1$



一般来说，逻辑门需花费成本、功耗和延迟，用导线来代替门电路更有利

T3: 重叠定理

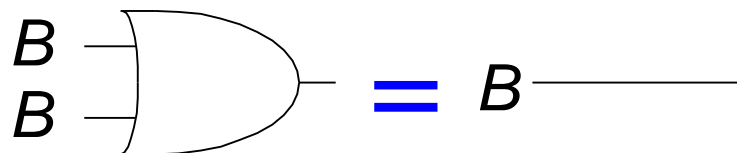
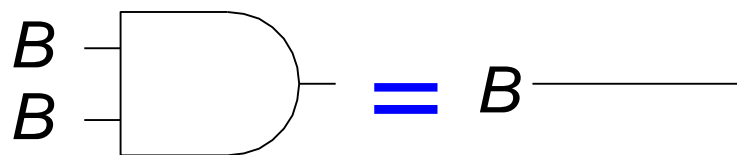


- $\mathbf{B} \cdot \mathbf{B} = \mathbf{B}$
- $\mathbf{B} + \mathbf{B} = \mathbf{B}$

T3: 重叠定理



- $B \cdot B = B$
- $B + B = B$



一般来说，逻辑门需花费成本、功耗和延迟，用导线来代替门电路更有利

T4: 回旋定理

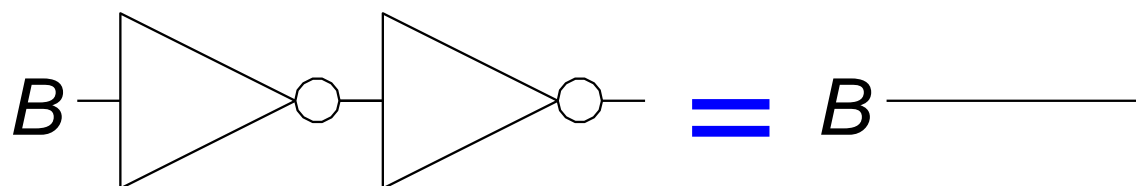


- $\overline{\overline{\mathbf{B}}} = \mathbf{B}$

T4: 回旋定理



- $\overline{\overline{B}} = B$



一般来说，逻辑门需花费成本、功耗和延迟，用导线来代替门电路更有利

T5: 互补定理

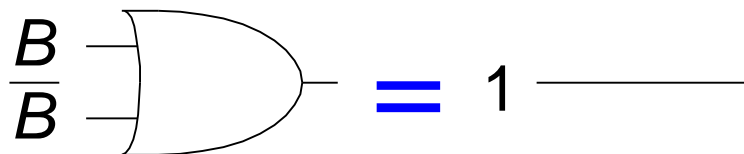
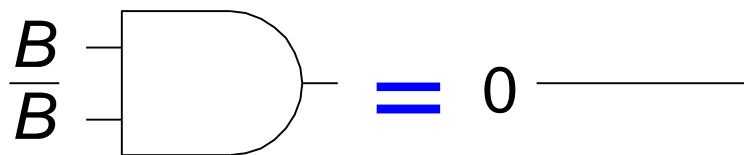


- $B \cdot \bar{B} = 0$
- $B + \bar{B} = 1$

T5: 互补定理



- $B \cdot \bar{B} = 0$
- $B + \bar{B} = 1$



一般来说，逻辑门需花费成本、功耗和延迟，用导线来代替门电路更有利

基本定理



Number	Theorem	Dual	Name
T1	$B \bullet 1 = B$	$B + 0 = B$	同一性定理
T2	$B \bullet 0 = 0$	$B + 1 = 1$	零元定理
T3	$B \bullet B = B$	$B + B = B$	重叠定理
T4	$\overline{\overline{B}} = B$		回旋定理
T5	$B \bullet \overline{B} = 0$	$B + \overline{B} = 1$	互补定理

Dual: Replace:

- with +
- 0 with 1

其他定理



Number	Theorem	Name
T6	$B \bullet C = C \bullet B$	交换律
T7	$(B \bullet C) \bullet D = B \bullet (C \bullet D)$	结合律
T8	$B \bullet (C + D) = (B \bullet C) + (B \bullet D)$	分配律
T9	$B \bullet (B + C) = B$	吸收律
T10	$(B \bullet C) + (B \bullet \bar{C}) = B$	合并律
T11	$(B \bullet C) + (\bar{B} \bullet D) + (C \bullet D) = (B \bullet C) + (\bar{B} \bullet D)$	一致律

其他定理



Number	Theorem	Name
T6	$B \bullet C = C \bullet B$	交换律
T7	$(B \bullet C) \bullet D = B \bullet (C \bullet D)$	结合律
T8	$B \bullet (C + D) = (B \bullet C) + (B \bullet D)$	分配律
T9	$B \bullet (B + C) = B$	吸收律
T10	$(B \bullet C) + (B \bullet \bar{C}) = B$	合并律
T11	$(B \bullet C) + (\bar{B} \bullet D) + (C \bullet D) = (B \bullet C) + (\bar{B} \bullet D)$	一致律

怎样证明这些定理是正确的？

怎样证明



- **Method 1: 归纳法**
- **Method 2: 使用其他的定理或公理简化方程**
 - 使等式的一边看起来像另一边

归纳法



- 也叫穷举法
- 检查每一种输入情况
- 如果两个表达式对每个可能的输入组合产生相同的值，则表达式是相等的

$$T 9: \quad B \cdot (B+C) = B$$

$$T 10: \quad (B \cdot C) + (B \cdot \bar{C}) = B$$

归纳法



Number	Theorem	Name
T6	$B \cdot C = C \cdot B$	交换律

<i>B</i>	<i>C</i>	<i>BC</i>	<i>CB</i>	
0	0	0	0	
0	1	0	0	
1	0	0	0	
1	1	1	1	2^n

T7:



Number	Theorem	Name
T7	$(B \bullet C) \bullet D = B \bullet (C \bullet D)$	结合律

T8:



Number	Theorem	Name
T8	$B \cdot (C + D) = (B \cdot C) + (B \cdot D)$	分配律

T9: 吸收律



Number	Theorem	Name
T9	$B \bullet (B+C) = B$	吸收律

T9: 吸收律



Number	Theorem	Name
T9	$B \cdot (B+C) = B$	吸收律

Prove true by:

- Method 1: 归纳法
- Method 2: 其他定理及公理

T9: 吸收律



Number	Theorem	Name
T9	$B \cdot (B+C) = B$	吸收律

	B	C	$(B+C)$	$B(B+C)$
Method 1:	0			
Perfect Induction	0			
	0			
	1			
	1			
	0			
	1			
	1			

T9: 吸收律



Number	Theorem	Name
T9	$B \cdot (B+C) = B$	吸收律

B	C	$(B+C)$	$B(B+C)$
0	0	0	0
0	1	1	0
1	0	1	1
1	1	1	1

T9: 吸收律



Number	Theorem	Name
T9	$B \bullet (B+C) = B$	吸收律

Method 2: Prove true using other axioms and theorems.

$$\begin{aligned} B \bullet (B+C) &= B \bullet B + B \bullet C \\ &= B + B \bullet C \\ &= B \bullet (1 + C) \\ &= B \bullet (1) \\ &= B \end{aligned}$$

T8: 分配律
T3: 同一性
T8: 分配律
T2: 零元
T1: 同一性

T10: 吸收律



Number	Theorem	Name
T10	$(B \bullet C) + (B \bullet \overline{C}) = B$	吸收律

Prove true using other axioms and theorems:

$$\begin{aligned} B \bullet C + B \bullet \overline{C} &= B \bullet (C + \overline{C}) && \text{T8: 分配律} \\ &= B \bullet (1) && \text{T5': 互补} \\ &= B && \text{T1: 同一} \end{aligned}$$

对称



#	Theorem	Dual	Name
T6	$B \bullet C = C \bullet B$	$B + C = C + B$	交换律
T7	$(B \bullet C) \bullet D = B \bullet (C \bullet D)$	$(B + C) + D = B + (C + D)$	结合律
T8	$B \bullet (C + D) = (B \bullet C) + (B \bullet D)$	$B + (C \bullet D) = (B + C) (B + D)$	合并律
T9	$B \bullet (B + C) = B$	$B + (B \bullet C) = B$	吸收律
T10	$(B \bullet C) + (B \bullet \bar{C}) = B$	$(B + C) \bullet (B + \bar{C}) = B$	合并律
T11	$(B \bullet C) + (\bar{B} \bullet D) + (C \bullet D) = (B \bullet C) + (\bar{B} \bullet D)$	$(B + C) \bullet (\bar{B} + D) \bullet (C + D) = (B + C) \bullet (\bar{B} + D)$	一致律

Dual: Replace:

- with +
- 0 with 1

布尔定理



#	Theorem	Dual	Name
T6	$B \bullet C = C \bullet B$	$B + C = C + B$	交换律
T7	$(B \bullet C) \bullet D = B \bullet (C \bullet D)$	$(B + C) + D = B + (C + D)$	结合律
T8	$B \bullet (C + D) = (B \bullet C) + (B \bullet D)$	$B + (C \bullet D) = (B + C) (B + D)$	合并律
T9	$B \bullet (B + C) = B$	$B + (B \bullet C) = B$	吸收律
T10	$(B \bullet C) + (B \bullet \bar{C}) = B$	$(B + C) \bullet (B + \bar{C}) = B$	合并律
T11	$(B \bullet C) + (B \bullet D) + (C \bullet D) = (B \bullet C) + (B \bullet D)$	$(B + C) \bullet (B + D) \bullet (C + D) = (B + C) \bullet (B + D)$	一致律

Warning: T8' differs from traditional algebra:

OR (+) distributes over AND (•)

对偶式子中OR 分配高于 AND, 相比T8 不同之处

布尔定理



#	Theorem	Dual	Name
T6	$B \bullet C = C \bullet B$	$B + C = C + B$	交换律
T7	$(B \bullet C) \bullet D = B \bullet (C \bullet D)$	$(B + C) + D = B + (C + D)$	结合律
T8	$B \bullet (C + D) = (B \bullet C) + (B \bullet D)$	$B + (C \bullet D) = (B + C) (B + D)$	合并律
T9	$B \bullet (B + C) = B$	$B + (B \bullet C) = B$	吸收律
T10	$(B \bullet C) + (B \bullet \overline{C}) = B$	$(B + C) \bullet (B + \overline{C}) = B$	合并律
T11	$(B \bullet C) + (B \bullet D) + (C \bullet D) = (B \bullet C) + (B \bullet D)$	$(B + C) \bullet (B + D) \bullet (C + D) = (B + C) \bullet (B + D)$	一致律

Axioms and theorems are useful for *simplifying* equations.

等式简化



将一个方程化简为隐式的个数最少，其中
每个隐式的文字量最少

简化方法



- 分配律 (T8, T8')

$$B(C+D) = BC + BD$$

$$B + CD = (B + C)(B + D)$$

- 吸收律 (T9')

$$A + AP = A$$

- 合并律 (T10)

$$PA + \overline{PA} = P$$

- Expansion

$$P = PA + \overline{PA}$$

$$A = A + \overline{AP}$$

- Duplication

$$A = A + A$$

- “Simplification” theorem

$$PA + \overline{A} = P + \overline{A}$$

简化方法 证明



“Simplification” theorem

$$PA + \bar{A} = P + \bar{A}$$

Method 1: $PA + \bar{A} = PA + (\bar{A} + \bar{A}P)$
 $= PA + P\bar{A} + \bar{A}$
 $= P(A + \bar{A}) + \bar{A}$
 $= P(1) + \bar{A}$
 $= P + \bar{A}$

T9' 吸收律

T6 交换律

T8 分配律

T5' 互补定理

T1 同一性定理

简化方法 证明



“Simplification” theorem

$$PA + \bar{A} = P + \bar{A}$$

Method 2: $PA + \bar{A} = (\bar{A} + A) (\bar{A} + P)$
 $= 1(\bar{A} + P)$
 $= \bar{A} + P$

T8' 分配律

T5' 补偿定理

T1 同一性定理

T11: 一致律



Number	Theorem	Name
T11	$(B \bullet C) + (\bar{B} \bullet D) + (C \bullet D) = (B \bullet C) + (\bar{B} \bullet D)$	一致律

Prove using other theorems and axioms:

T11: 一致律



Number	Theorem	Name
T11	$(B \cdot C) + (\bar{B} \cdot D) + (C \cdot D) = (B \cdot C) + (\bar{B} \cdot D)$	一致律

Prove using other theorems and axioms:

$$B \cdot C + \bar{B} \cdot D + C \cdot D$$

$$= BC + \bar{B}D + (CDB + C\bar{D}\bar{B})$$

T10: 合并律

$$= BC + \bar{B}D + BCD + \bar{B}CD$$

T6: 交换律

$$= BC + BCD + \bar{B}D + \bar{B}CD$$

T6: 交换律

$$= (BC + BCD) + (\bar{B}D + \bar{B}CD)$$

T7: 结合律

$$= BC + \bar{B}D$$

T9': 吸收律

简化方法回顾



- Distributivity (T8, T8')

$$B(C+D) = BC + BD$$

$$B + CD = (B + C)(B + D)$$

- Covering (T9')

$$A + AP = A$$

- Combining (T10)

$$\overline{PA} + PA = P$$

- Expansion

$$P = \overline{PA} + PA$$

$$A = A + AP$$

- Duplication

$$A = A + A$$

- “Simplification” theorem

$$\overline{PA} + A = P + A$$

$$PA + \overline{A} = P + \overline{A}$$

简化例子



Example 1:

$$Y = AB + A\bar{B}$$

简化例子



Example 1:

$$Y = AB + A\bar{B}$$

$$Y = A$$

T10: Combining 合并律

or

$$= A(B + \bar{B})$$

$$= A(1)$$

$$= A$$

T8: Distributivity 分配律

T5': Complements 互补定理

T1: Identity 同一性定理

简化例子



Example 2:

$$Y = A(AB + ABC)$$

简化例子



Example 2:

$$Y = A(AB + ABC)$$

$$= A(AB(1 + C))$$

$$= A(AB(1))$$

$$= A(AB)$$

$$= (AA)B$$

$$= AB$$

T8: Distributivity 分配律

T2': Null Element 零元定理

T1: Identity 同一性定理

T7: Associativity 结合律

T3: Idempotency 重叠定理

简化例子



Example 3:

$$Y = A'BC + A'$$

$$\text{Recall: } A' = \bar{A}$$

简化例子



Example 3:

$$Y = A'BC + A'$$

$$= A'$$

or

$$= A'(BC + 1)$$

$$= A'(1)$$

$$= A'$$

$$\text{Recall: } A' = \bar{A}$$

T9' Covering吸收律

T8: Distributivity分配律

T2': Null Element零元定理

T1: Identity同一性定理

基本定理



Number	Theorem	Dual	Name
T1	$B \bullet 1 = B$	$B + 0 = B$	同一性定理
T2	$B \bullet 0 = 0$	$B + 1 = 1$	零元定理
T3	$B \bullet B = B$	$B + B = B$	重叠定理
T4	$\overline{\overline{B}} = B$		回旋定理
T5	$B \bullet \overline{B} = 0$	$B + \overline{B} = 1$	互补定理

Dual: Replace:

- with +
- 0 with 1

布尔定理



#	Theorem	Dual	Name
T6	$B \bullet C = C \bullet B$	$B + C = C + B$	交换律
T7	$(B \bullet C) \bullet D = B \bullet (C \bullet D)$	$(B + C) + D = B + (C + D)$	结合律
T8	$B \bullet (C + D) = (B \bullet C) + (B \bullet D)$	$B + (C \bullet D) = (B + C) (B + D)$	合并律
T9	$B \bullet (B + C) = B$	$B + (B \bullet C) = B$	吸收律
T10	$(B \bullet C) + (B \bullet \overline{C}) = B$	$(B + C) \bullet (B + \overline{C}) = B$	合并律
T11	$(B \bullet C) + (B \bullet D) + (C \bullet D) = (B \bullet C) + (B \bullet D)$	$(B + C) \bullet (B + D) \bullet (C + D) = (B + C) \bullet (B + D)$	一致律

Axioms and theorems are useful for *simplifying* equations.

简化例子



Example 4:

$$Y = AB'C + ABC + A'BC$$

简化例子



Example 4:

$$Y = AB'C + ABC + A'BC$$

$$\begin{aligned} &= AB'C + \mathbf{ABC} + \mathbf{ABC} + A'BC && \text{T3': Idempotency 重叠定理} \\ &= (AB'C+ABC) + (ABC+A'BC) && \text{T7': Associativity 结合律} \\ &= AC + BC && \text{T10: Combining 合并律} \end{aligned}$$

德.摩根定理



Number	Theorem	Name
T12	$\overline{B_0 \bullet B_1 \bullet B_2 \dots} = \overline{B_0} + \overline{B_1} + \overline{B_2} \dots$	德.摩根定理

非常有力工具：所有项乘积的补等于每个项各自取补后相加。同样，所有项相加的补等于每个项各自取补后相乘

德.摩根定理: 对称



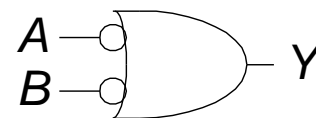
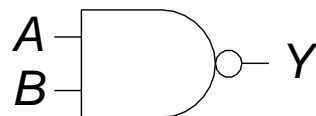
#	Theorem	Dual	Name
T12	$\overline{B_0 \bullet B_1 \bullet B_2 \dots} = \overline{B_0 + B_1 + B_2 \dots}$	$\overline{B_0 + B_1 + B_2 \dots} = \overline{B_0 \bullet B_1 \bullet B_2 \dots}$	德.摩根定理

非常有力工具：所有项乘积的补等于每个项各自取补后相加。同样，所有项相加的补等于每个项各自取补后相乘

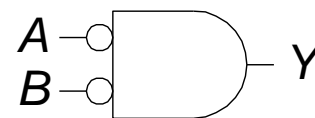
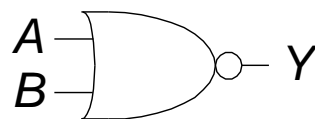
德.摩根定理



- $Y = \overline{AB} = \overline{A} + \overline{B}$



- $Y = \overline{A + B} = \overline{A} \cdot \overline{B}$



德.摩根定理 例子1



$$Y = \overline{\overline{(A+BD)}C}$$

德.摩根定理 例子1



$$\begin{aligned} Y &= \overline{\overline{(A+BD)}C} \\ &= \overline{\overline{(A+BD)} + \overline{C}} \\ &= \overline{\overline{A} \bullet \overline{\overline{BD}}} + C \\ &= \overline{\overline{A} \bullet (BD)} + C \\ &= \overline{A}BD + C \end{aligned}$$

德.摩根定理 例子 2



$$Y = \overline{\overline{ACE+D}} + B$$

德.摩根定理 例子 2



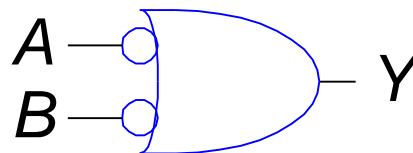
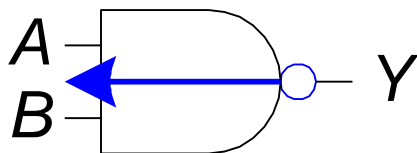
$$\begin{aligned} Y &= \overline{\overline{ACE+D}} + B \\ &= \overline{\overline{ACE+D}} \cdot \overline{B} \\ &= \overline{\overline{ACE} \cdot \overline{D}} \cdot \overline{B} \\ &= \overline{(\overline{AC+E}) \cdot D} \cdot \overline{B} \\ &= \overline{(AC+\overline{E}) \cdot D} \cdot \overline{B} \\ &= \overline{ACD + D\overline{E}} \cdot \overline{B} \\ &= \overline{ABCD} + \overline{BDE} \end{aligned}$$

“推泡泡”



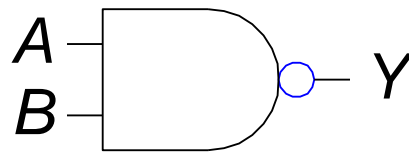
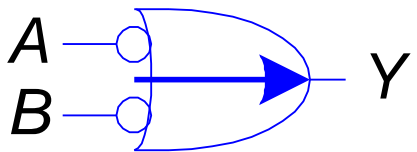
• 反向:

- 主体改变 (与门 \leftrightarrow 或门)
- 泡泡推到输入



• 前向:

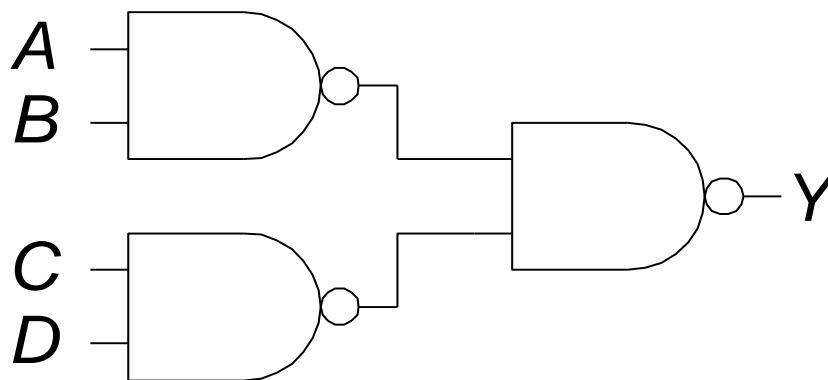
- 主体改变 (与门 \leftrightarrow 或门)
- 泡泡推到输出



推泡泡



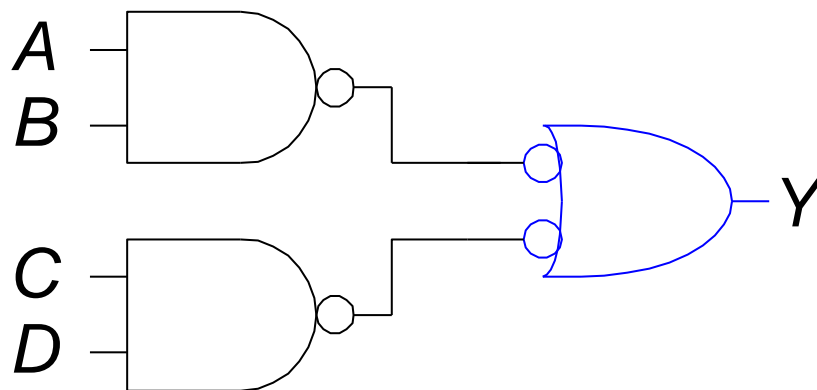
- 快速写出此电路表达式?



推泡泡



- 电路表达式?

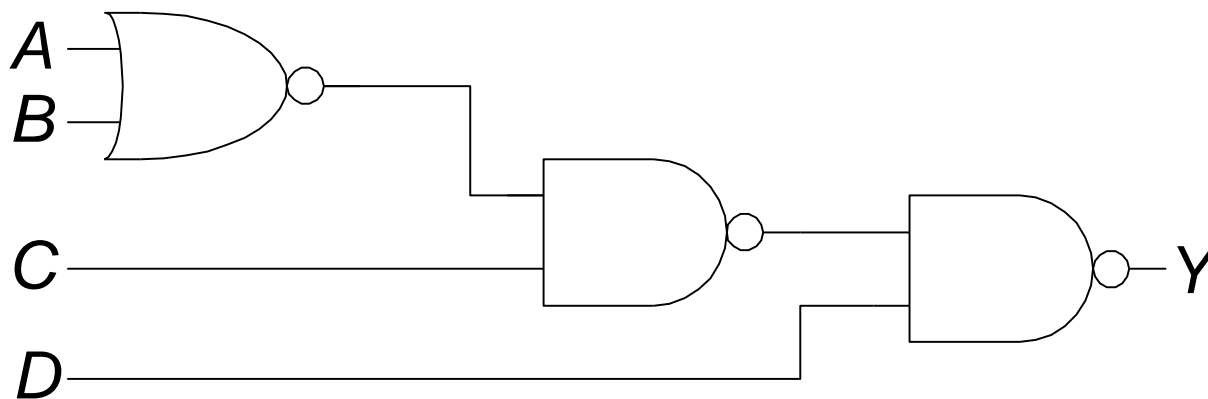


$$Y = AB + CD$$

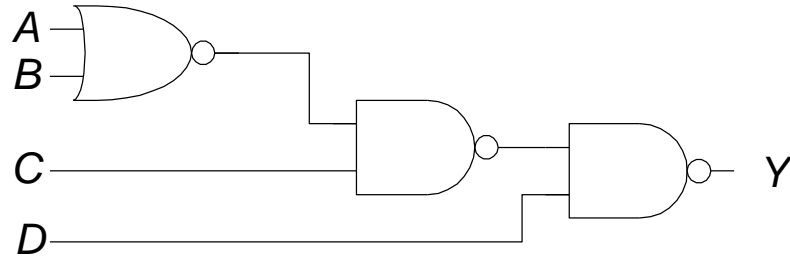
推泡泡规则



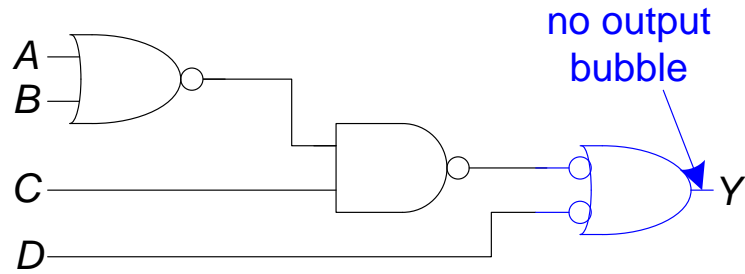
- 从输出开始, 一直到输入
- 改变主体



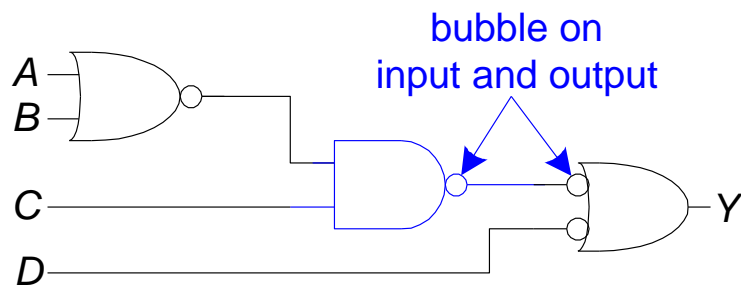
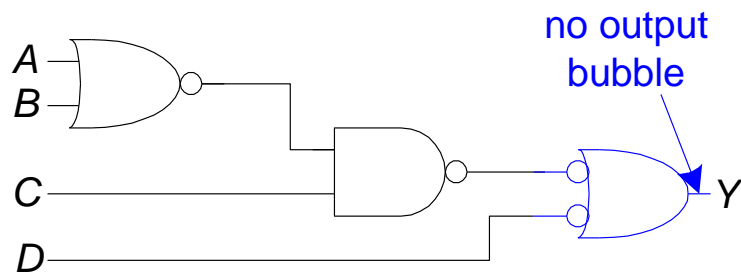
推泡泡例子



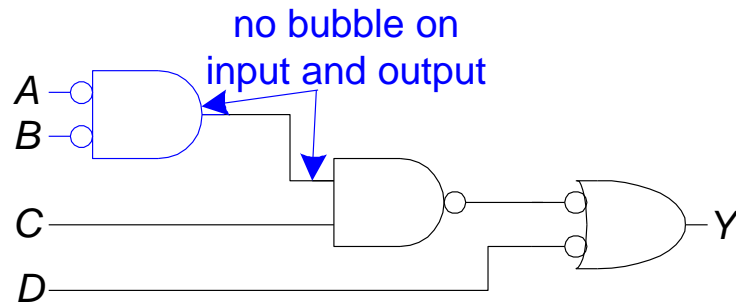
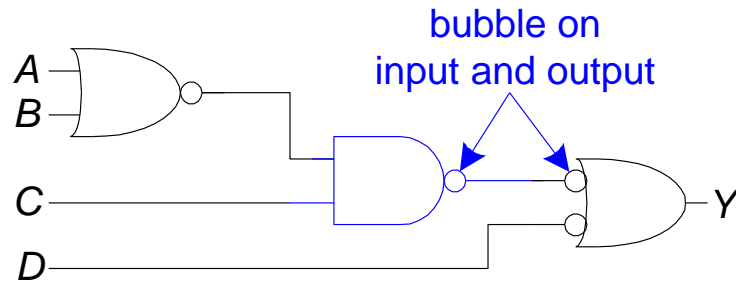
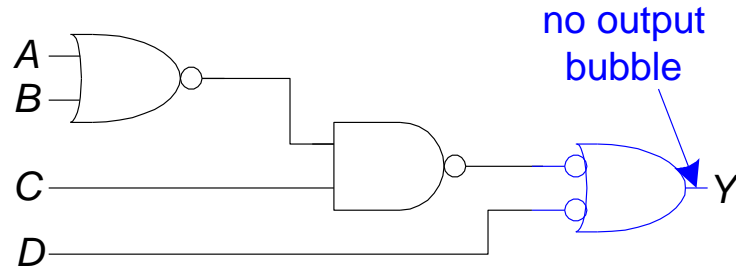
推泡泡例子



推泡泡例子



推泡泡例子





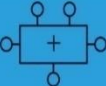






$$Y = \overline{A} \overline{B} C + \overline{D}$$

第二章 组合逻辑设计



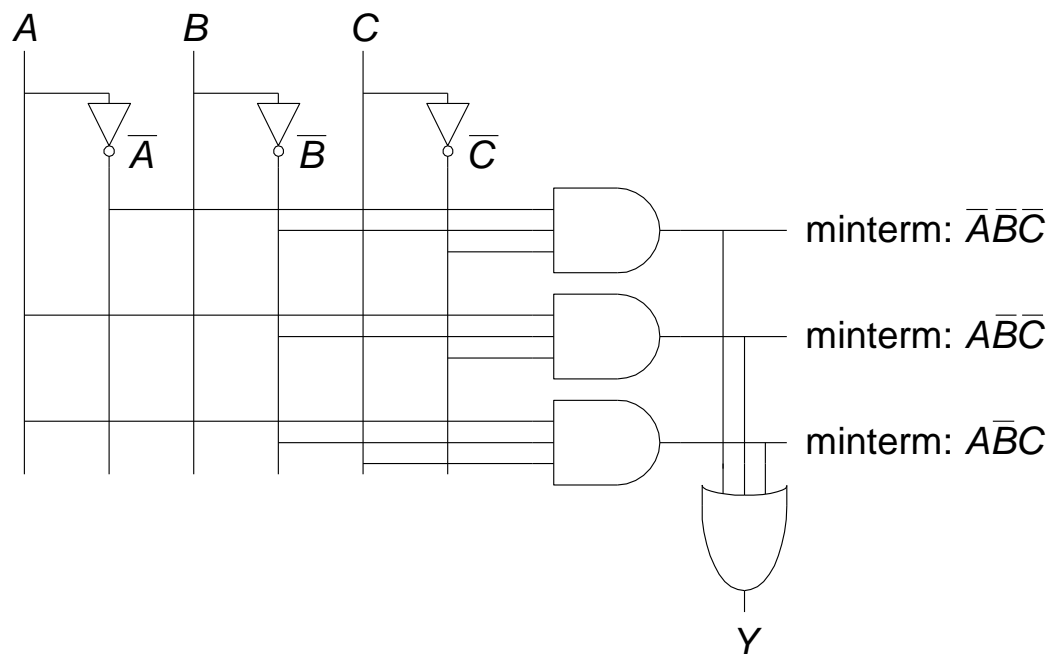
- 介绍（组合逻辑电路，）
- 布尔表达式（最小项、最大项、与或 或与）
- 布尔代数（真值表、布尔表达式、电路、化简）
- 从逻辑到门
- 多级组合逻辑
- X和Z
- 卡诺图
- 组合逻辑模块
- 时序

Application Software	 >"hello world!"
Operating Systems	
Architecture	
Micro-architecture	
Logic	
Digital Circuits	
Analog Circuits	
Devices	
Physics	

从逻辑到门



- 两级逻辑: 与门之后接或门
- 例子: $Y = \overline{A}B\overline{C} + A\overline{B}\overline{C} + A\overline{B}C$



电路原理图



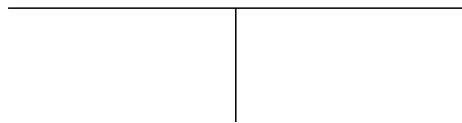
- 描述了数字电路的内部元件及其相互连接
- 输入在原理图的左边或顶部
- 输出在原理图的右边或底部
- 门必须从左至右
- 最好直线无拐角

电路原理图

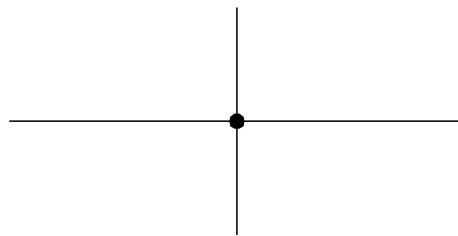


- 线总是在T型接头连接
- 俩线交接处有点
- 俩线交接处无点

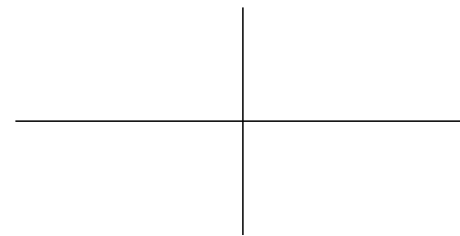
wires connect
at a T junction



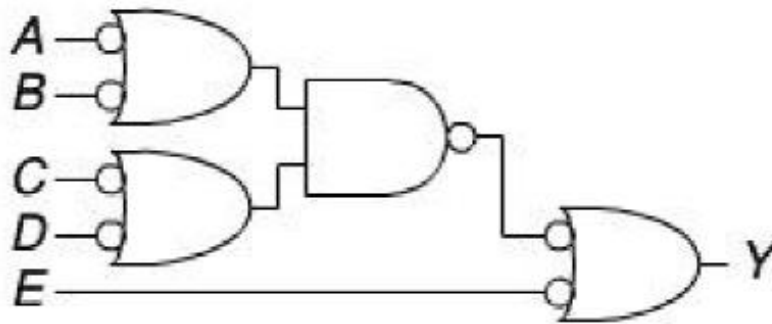
wires connect
at a dot



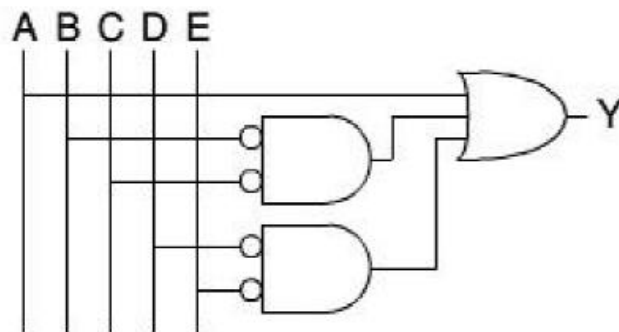
wires crossing
without a dot do
not connect



$$Y = (\bar{A} + \bar{B})(\bar{C} + \bar{D}) + \bar{E}$$



$$Y = A + \bar{B}\bar{C} + \bar{D}\bar{E}$$



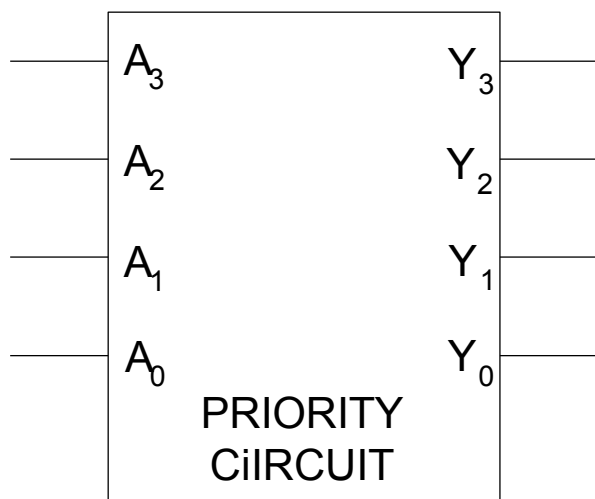
多输出电路



• Example: 优先电路

校长、院长、系主任、班主任
都要使用会议室

$A_3, A_2, A_1, A_0 \rightarrow Y_3, Y_2, Y_1, Y_0$



A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0				
0	0	0	1				
0	0	1	0				
0	0	1	1				
0	1	0	0				
0	1	0	1				
0	1	1	0				
0	1	1	1				
1	0	0	0				
1	0	0	1				
1	0	1	0				
1	0	1	1				
1	1	0	0				
1	1	0	1				
1	1	1	0				
1	1	1	1				

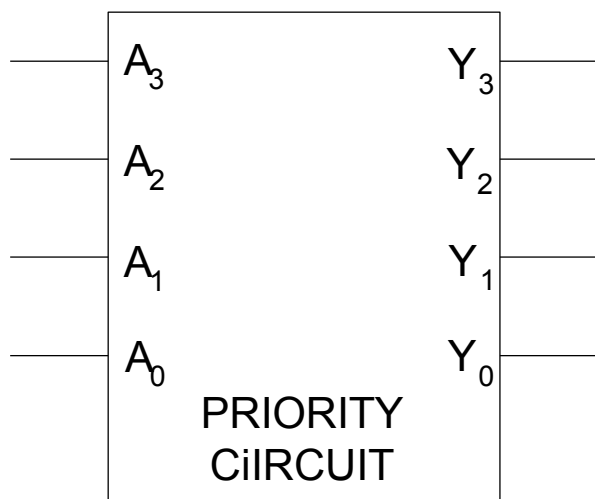
多输出电路



• Example: 优先电路

校长、院长、系主任、班主任
都要使用会议室

$A_3, A_2, A_1, A_0 \rightarrow Y_3, Y_2, Y_1, Y_0$

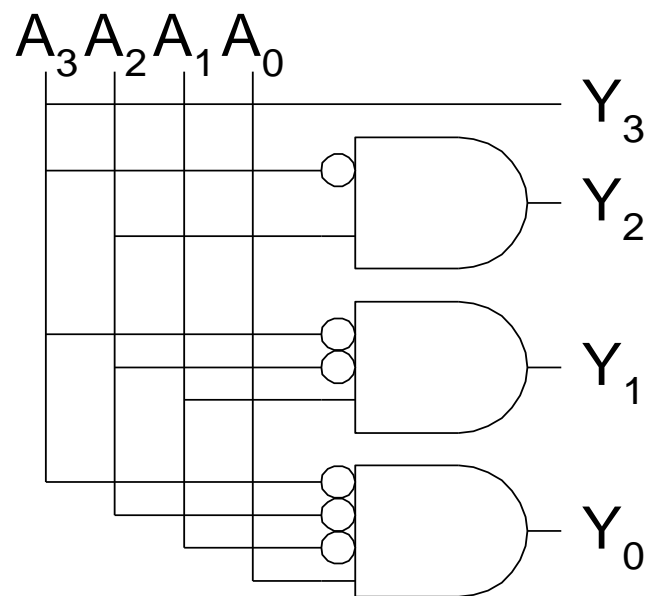


A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	0
0	1	0	0	0	1	0	0
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	0
1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	0
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	1	0	0	0
1	1	1	1	1	0	0	0

优先电路硬件实现



A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	0
0	1	0	0	0	1	0	0
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	0
1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	0
1	0	1	0	1	0	0	0
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	1	0	0	0

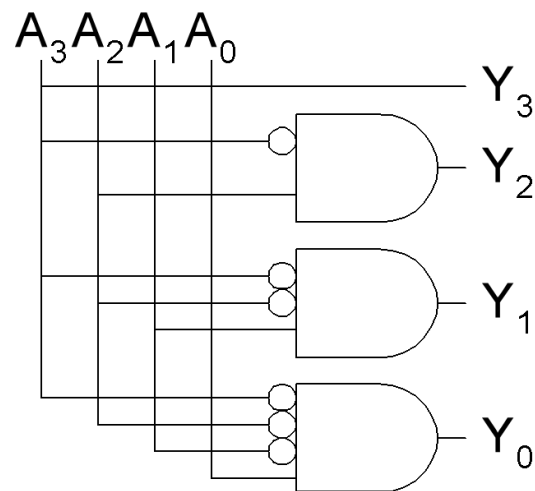


不予考虑 X

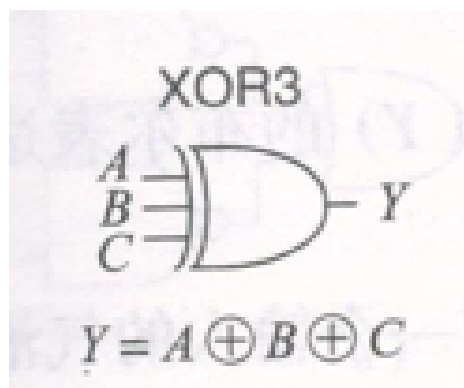


A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	0
0	1	0	0	0	1	0	0
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	0
1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	0
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	1	0	0	0

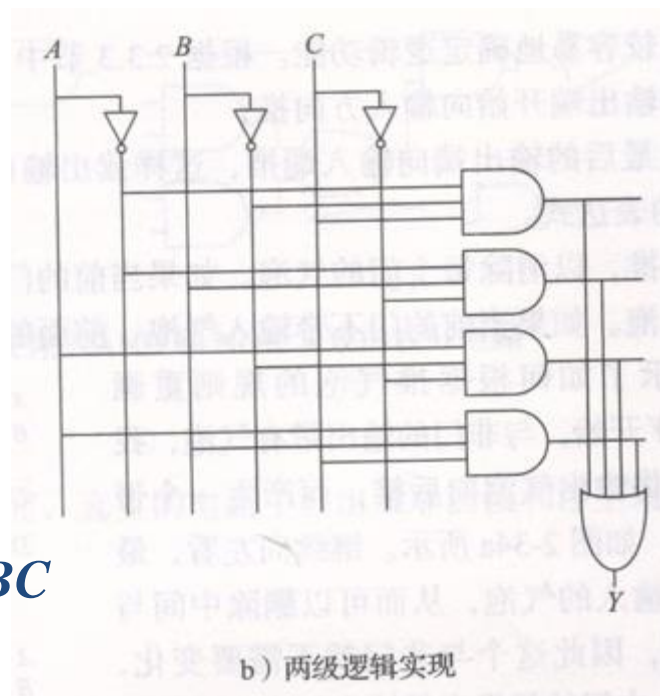
A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	X	0	0	1	0
0	1	X	X	0	1	0	0
1	X	X	X	1	0	0	0



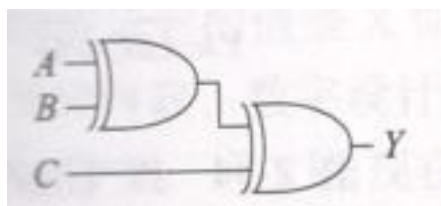
多级组合逻辑



A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1



$$Y = \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC$$



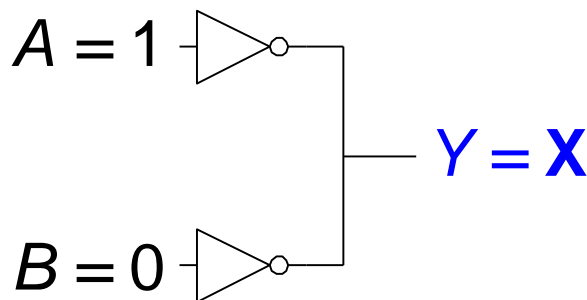
$$Y = (A \oplus B) \oplus C$$

标准：门数量最少、速度最快、功耗最低；CMOS电路中用与非门、或非门取代与门、或门更高效。

竞争值：X



- **竞争Contention:** 电路同时被 1 或 0 驱动，节点Y同时被高电平和低电平驱动，真实电压 $0 \text{---} V_{dd}$
 - 可能随电压、温度、时间、噪音而改变
 - 会造成过度的功耗



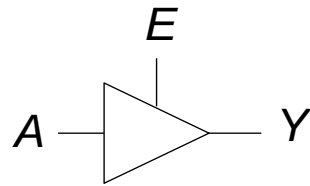
- **Warnings:**
 - 竞争经常表明是一个bug
 - **X** is used for “**don't care**” and **contention** - look at the context to tell them apart.
 - 真值表中的X 和 电路中的X 不同！

浮空值: Z 即没有被高电平 也无低电平驱动

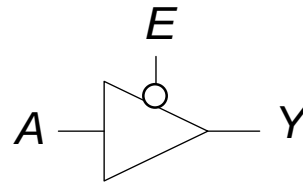


- 浮空、高阻态、高Z态（输入没连电压源）
- 高阻态的输出可能是 0, 1, or somewhere in between,

– Tristate Buffer（三态缓冲器）



E	A	Y
0	0	Z
0	1	Z
1	0	0
1	1	1



E	A	Y
1	0	Z
1	1	Z
0	0	0
0	1	1

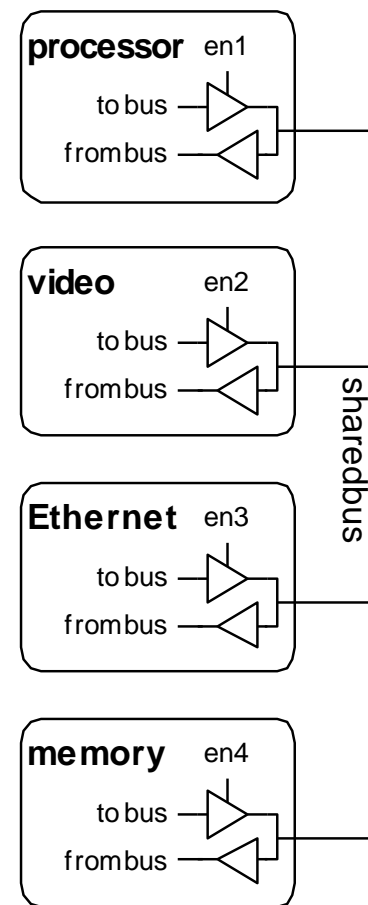
三态总线



浮动节点用于三态总线

- 许多不同的驱动程序
- Exactly one is active at once

三态缓冲器经常在连接多个芯片的总线上使用，每个芯片可以通过三态缓冲器与共享存储器总线连接，某时刻只允许一个芯片使能其它芯片的输出必须为高阻态（Floating）
点到点链路的现代计算机用得比较少了



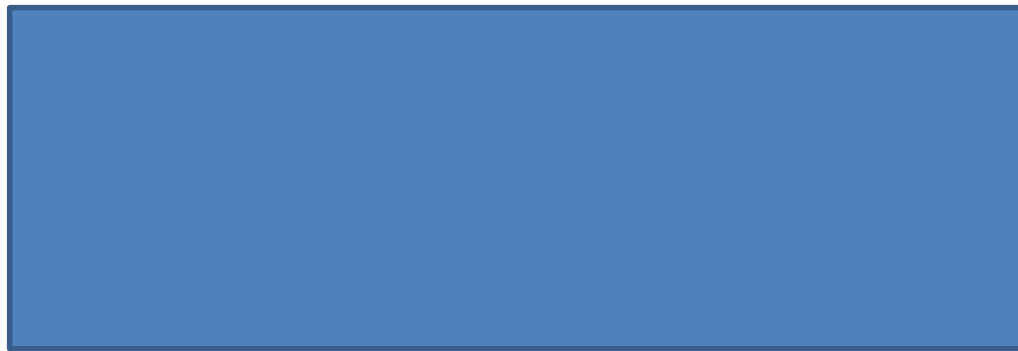
布尔代数化简 课堂练习



$$(a) Y=AC+\bar{A}\bar{B}C$$

$$(b) Y=\bar{A}\bar{B}+\bar{A}B\bar{C}+(\bar{A}+\bar{C})$$

$$(c) Y=\bar{A}\bar{B}\bar{C}\bar{D}+A\bar{B}\bar{C}+A\bar{B}C\bar{D}+ABD+\bar{A}\bar{B}C\bar{D}+B\bar{C}D+\bar{A}$$



卡诺图



- 布尔表达式可以最小化
- 卡诺图K-map: 图形化简化布尔表达式
- 对处理最多4个变量的问题非常好
- 理论依据: $PA + P\bar{A} = P$ (卡诺图中相邻格子)
- 卡诺图每一个方格与真值表一行中的一个输出Y相对应
- 卡诺图的每一个方格子代表了一个最小项

卡诺图



A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

		AB			
		00	01	11	10
C	0	1	0	0	0
	1	1	0	0	0

		AB			
		00	01	11	10
C	0	$\bar{A}\bar{B}\bar{C}$	$\bar{A}B\bar{C}$	$AB\bar{C}$	$A\bar{B}\bar{C}$
	1	$\bar{A}\bar{B}C$	$\bar{A}BC$	ABC	$A\bar{B}C$

先把真值表的与或项（值为1的最小项相加）写出来，再对比和卡诺图化简的效果。

输入卡诺图---格雷码



A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Y C	AB			
	00	01	11	10
0	1	0	0	0
1	1	0	0	0

Y C	AB			
	00	01	11	10
0	$\bar{A}\bar{B}\bar{C}$	$\bar{A}B\bar{C}$	$AB\bar{C}$	$A\bar{B}\bar{C}$
1	$\bar{A}\bar{B}C$	$\bar{A}BC$	ABC	$A\bar{B}C$

**格雷码相邻项只有一个变量不同，
与传统不同：00 01 10 11**

卡诺图是“环绕的”，最右边的方格可以有效地和最左边的方格相邻，也只有一个变量不同。“卷成圆柱体”“圆”

卡诺图---画图原理



- 框出“1”
- 在同一个圈中，同时包含了真值形式和取反形式的某个变量可以从蕴含项中删除

A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

		AB			
		00	01	11	10
C	0	1	0	0	0
	1	1	0	0	0

$$Y = \overline{A}\overline{B}$$

3输入卡诺图



Truth Table

A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

K-Map

Y C		AB			
		00	01	11	10
0					
1					

Y C		AB			
		00	01	11	10
0	$\bar{A}\bar{B}C$	$\bar{A}BC$	ABC	$AB\bar{C}$	
1	$\bar{A}\bar{B}\bar{C}$	$\bar{A}B\bar{C}$	ABC	$A\bar{B}\bar{C}$	

卡诺图中的定义



- **Complement补码**: variable with a bar over it
 $\bar{A}, \bar{B}, \bar{C}$
- **Literal项**: variable or its complement
 $\bar{A}, A, \bar{B}, B, C, \bar{C}$
- **Implicant蕴涵项**: product of literals
 $A\bar{B}C, \bar{A}C, BC$
- **Prime implicant主蕴涵项**: implicant corresponding to the largest circle in a K-map
- ($\bar{A}\bar{B}\bar{C}, \bar{A}\bar{B}C, \bar{A}B\bar{C}, \bar{A}BC, A\bar{B}\bar{C}, A\bar{B}C, AB\bar{C}, ABC, \bar{A}\bar{B}$)

卡诺图化简逻辑



布尔表达式化简 == 卡诺图化简

- 用最少的圈来圈 1，圈中所有方格都是1；
- 每个圈必须是矩形，边长为1,2,4,8；
- 每个圈必须尽可能大；
- 圈可以环绕卡诺图的边界；
- 一个为1的方格可以被多次圈中。

4输入卡诺图



<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>Y</i>
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

<i>Y</i>	<i>AB</i>			
<i>CD</i>	00	01	11	10
00				
01				
11				
10				

4输入卡诺图



<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>Y</i>
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

<i>Y</i>	<i>AB</i>			
<i>CD</i>	00	01	11	10
00	1	0	0	1
01	0	1	0	1
11	1	1	0	0
10	1	1	0	1

4输入卡诺图

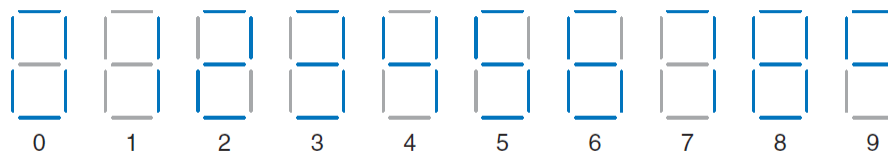
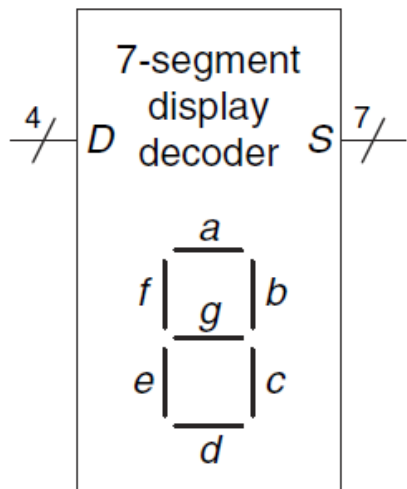


A	B	C	D	Y
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

Y	AB			
CD	00	01	11	10
00	1	0	0	1
01	0	1	0	1
11	1	1	0	0
10	1	1	0	1

$$Y = \bar{A}C + \bar{A}BD + A\bar{B}\bar{C} + \bar{B}\bar{D}$$

7段数码管显示译码器 --- 很常见的例子



$D_{3:0}$	S_a	S_b	S_c	S_d	S_e	S_f	S_g
0000	1	1	1	1	1	1	0
0001	0	1	1	0	0	0	0
0010	1	1	0	1	1	0	1
0011	1	1	1	1	0	0	1
0100	0	1	1	0	0	1	1
0101	1	0	1	1	0	1	1
0110	1	0	1	1	1	1	1
0111	1	1	1	0	0	0	0
1000	1	1	1	1	1	1	1
1001	1	1	1	0	0	1	1
others	0	0	0	0	0	0	0

通过四位数据输入，产生七位输出来控制发光二极管显示数字0-9

思考一下实现过程!

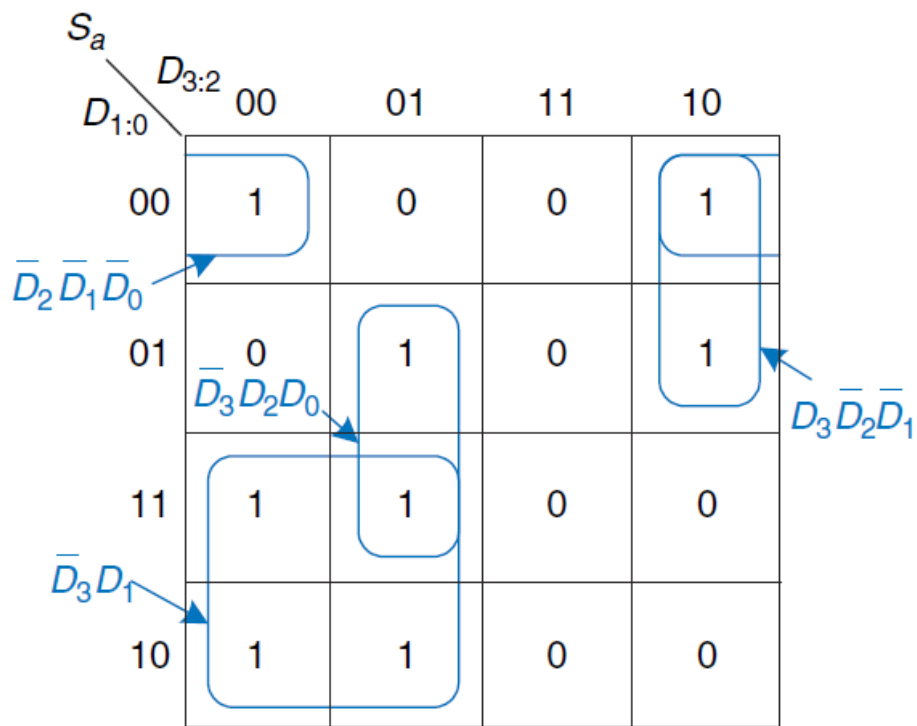


Truth table for function S_a :

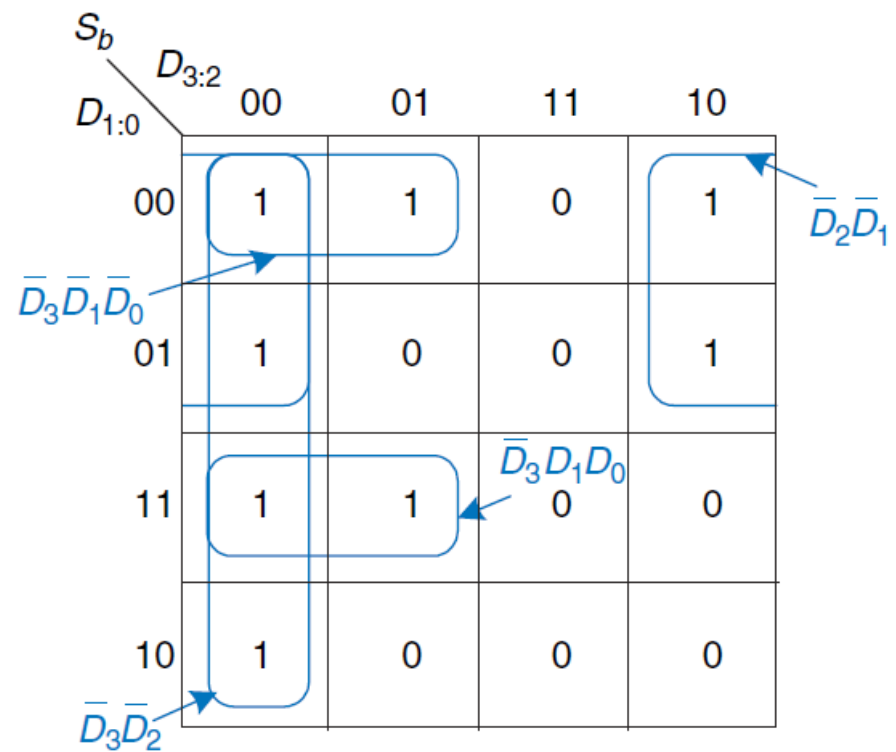
S_a	$D_{3:2}$			
$D_{1:0}$	00	01	11	10
00	1	0	0	1
01	0	1	0	1
11	1	1	0	0
10	1	1	0	0

Truth table for function S_b :

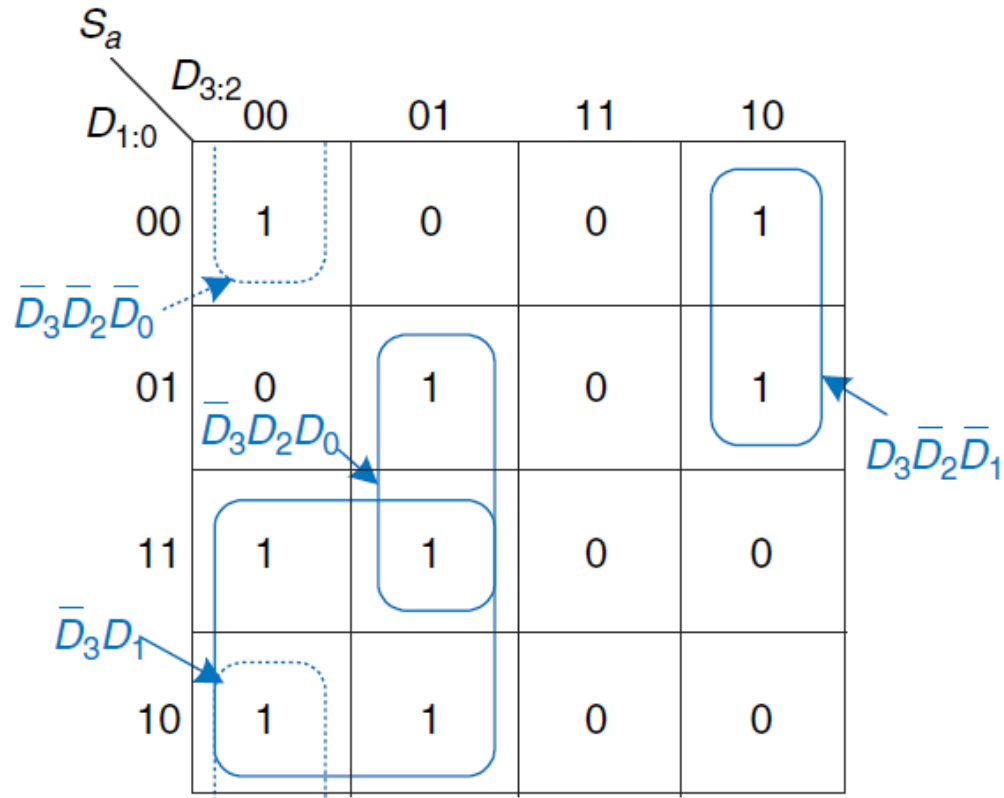
S_b	$D_{3:2}$			
$D_{1:0}$	00	01	11	10
00	1	1	0	1
01	1	0	0	1
11	1	1	0	0
10	1	0	0	0



$$S_a = \bar{D}_3D_1 + \bar{D}_3D_2D_0 + D_3\bar{D}_2\bar{D}_1 + \bar{D}_2\bar{D}_1\bar{D}_0$$



$$S_b = \bar{D}_3\bar{D}_2 + \bar{D}_2\bar{D}_1 + \bar{D}_3D_1D_0 + \bar{D}_3\bar{D}_1\bar{D}_0$$



$$S_a = \bar{D}_3D_1 + \bar{D}_3D_2D_0 + D_3\bar{D}_2\bar{D}_1 + \bar{D}_3\bar{D}_2\bar{D}_0$$

无关项的卡诺图



A	B	C	D	Y
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	X
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

Y		AB			
		00	01	11	10
CD	00				
	01				
	11				
	10				

- 有帮助可以圈起来，
- 没帮助也可以不圈起来

无关项的卡诺图



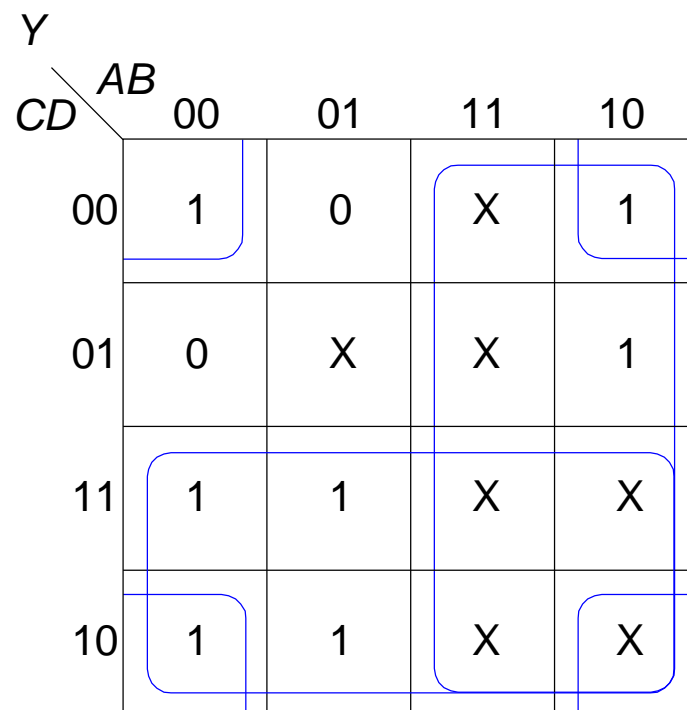
A	B	C	D	Y
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	X
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

Y CD \ AB		AB			
		00	01	11	10
00	00	1	0	X	1
	01	0	X	X	1
11	11	1	1	X	X
	10	1	1	X	X

无关项的卡诺图



A	B	C	D	Y
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	X
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X



$$Y = A + \bar{B}\bar{D} + C$$

无关项的卡诺图



布尔代数和卡诺图是两种逻辑简化方法
最终目标都是找到开销最低的特定逻辑函数

现代工程实践使用 逻辑综合(Logic Synthesis)代替人工方法.

大规模电路优化-- EDA软件综合

(Synopsis DesignCompiler)

小规模 电路--人工方法优化

第二章 组合逻辑设计



- 介绍（组合逻辑电路，）
- 布尔表达式（最小项、最大项、与或 或与）
- 布尔代数（真值表、布尔表达式、电路、化简）
- 从逻辑到门
- 多级组合逻辑
- X和Z
- 卡诺图
- 组合逻辑模块
- 时序

Application Software	
Operating Systems	
Architecture	
Micro-architecture	
Logic	
Digital Circuits	
Analog Circuits	
Devices	
Physics	

组合逻辑模块

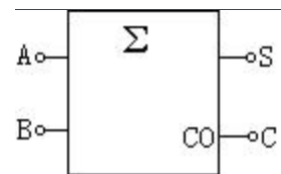


- 半加器
- 全加器
- 优先电路
- 7段译码器
- 选择器
- 译码器

半加器（一位） --- （自己设计）



半加器电路是指对两个输入数据位相加，输出一个结果位和进位，**没有进位输入的加法器电路**。是实现两个一位二进制数的加法运算电路。



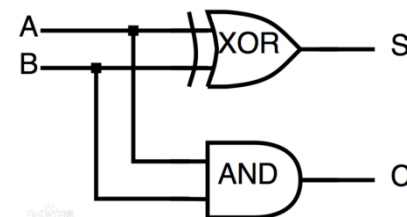
A和B是相加的两个数，S是半加和数，C是进位数。所谓半加就是不考虑进位的加法

被加数A	加数B	和数S	进位数C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

表中：

$$S = \bar{A}B + A\bar{B} \quad S = A \oplus B$$

$$C = AB$$



全加器（一位）



半加器是实现**两个一位二进制码相加**的电路，因此只能用于两个二进制码**最低位**的相加。能计算低位进位的两个一位二进制码的相加电路，即为全加器。

全加器英语名称为full-adder，是用门电路实现两个二进制数相加并求出和的组合线路，称为一位全加器。**一位全加器可以处理低位进位，并输出本位加法进位。**

多个一位全加器进行级联可以得到**多位全加器**。

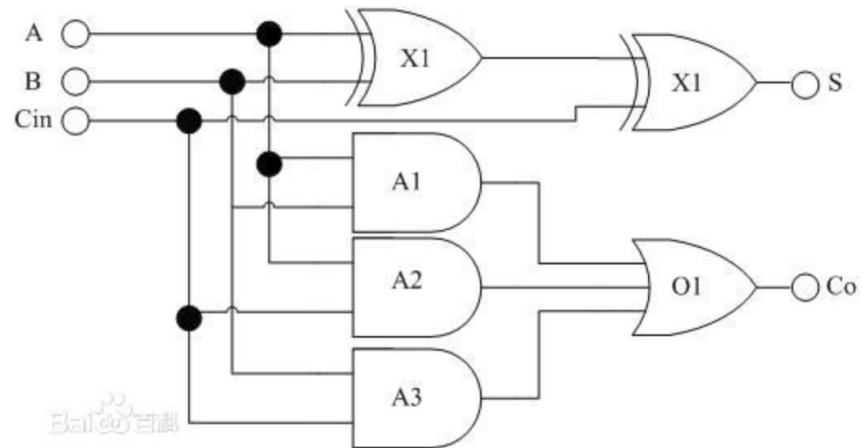
输入			输出	
C _{i-1}	A _i	B _i	S _i	C _i
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$S_i = A_i \oplus B_i \oplus C_{i-1}$$

$$C_i = A_i B_i + C_{i-1} (A_i + B_i)$$

第二个表达式也可用一个异或门来代替或门对其中两个输入信号进行求和。

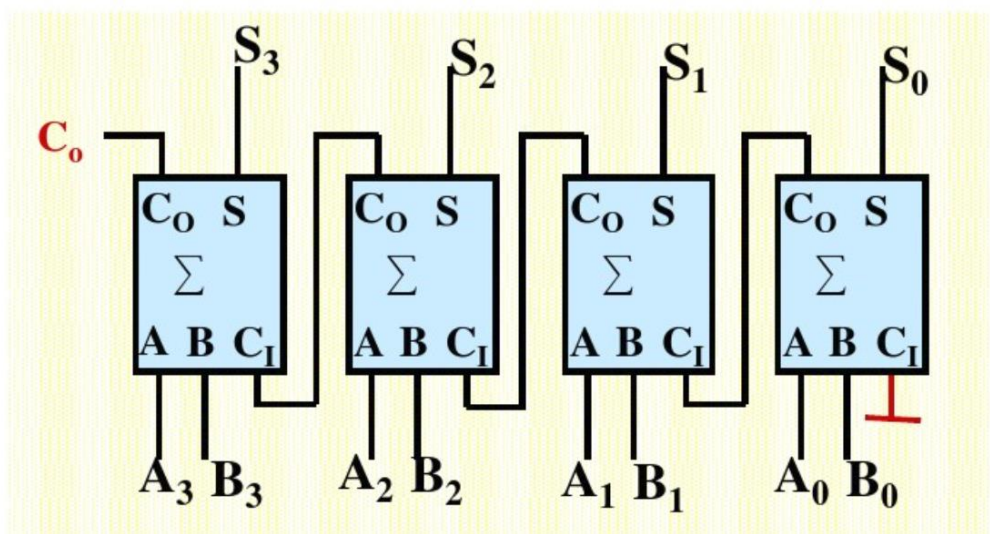
$$C_i = A_i B_i + C_{i-1} (A_i \oplus B_i)$$



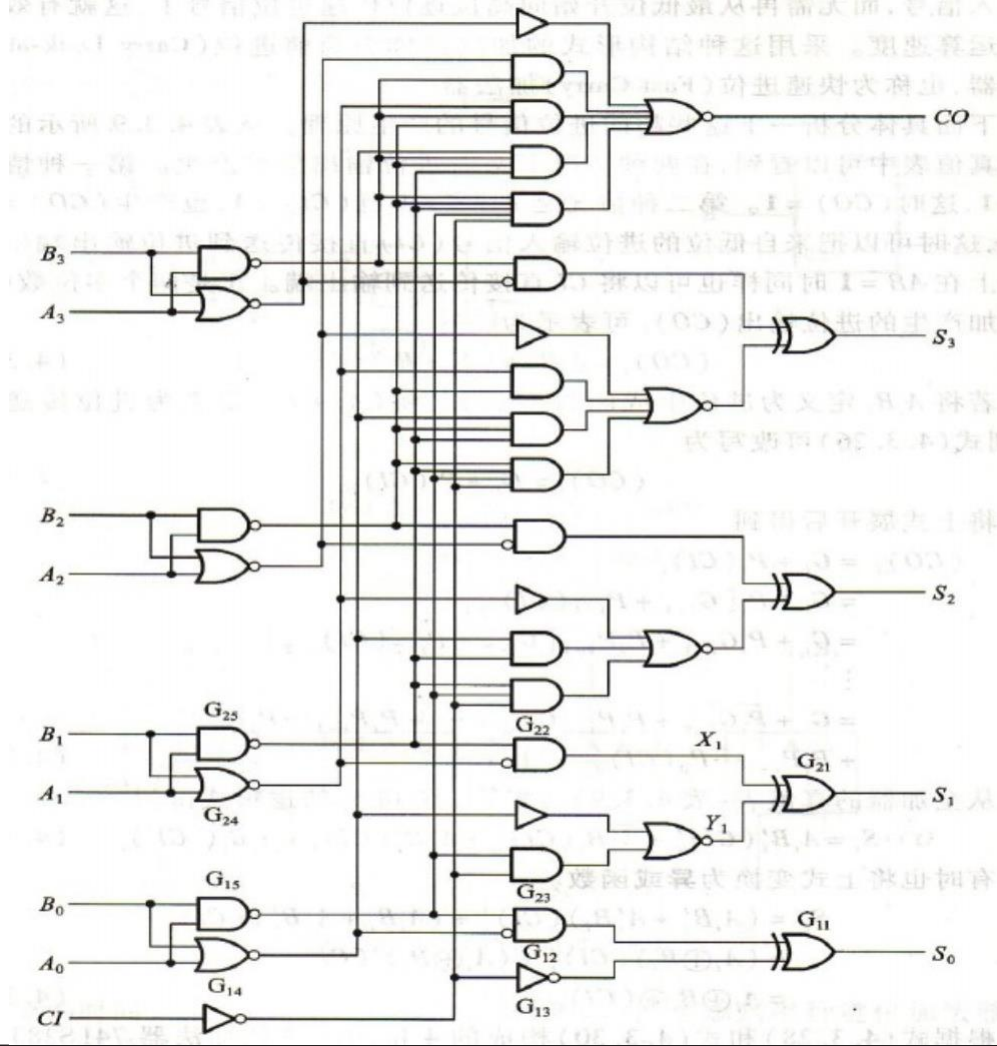
多位加法器 (怎样构建)



要实现32位的二进制加法，一种自然的想法就是将1位的二进制加法重复32次（即**逐位进位加法器**）。这样做无疑是可行且易行的，但由于每一位的CIN都是由前一位的COUT提供的，所以第2位必须在第1位计算出结果后，**才能开始计算**；第3位必须在第2位计算出结果后，才能开始计算，等等。而最后的第32位必须在前31位全部计算出结果后，才能开始计算。这样的方法，使得实现32位的二进制加法所需的时间是**实现1位的二进制加法的时间的32倍**。



4位全加器



4位全加器systemverilog实现



主模块:

```
module full_adder4(a,b,cin,s,cout);  
input[3:0] a,b;  
input      cin;  
output[3:0] s;  
output      cout;  
wire[3:0]    s;  
assign {cout,s} = a + b + cin;  
endmodule
```

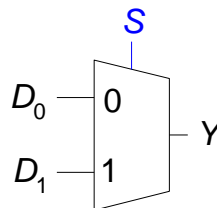
测试模块:

```
module test_full_adder4;  
reg[3:0]    a,b; reg      cin;  
wire[3:0]   s;  
initial begin  
            a = 4'h5; b=4'hb; cin=1'b1;  
            #10 a = 4'ha; b=4'hc; cin=1'b0;  
            #10 a = 4'h8; b=4'h6; cin=1'b0;  
            #10 a = 4'h4; b=4'hf; cin=1'b1;  
            #10 $stop; end  
full_adder4  n1(a, b, cin, s, cout);  
endmodule
```

选择器 (Mux)



- 最常用组合逻辑电路，多个输入选一个输出
- $\log_2 N$ -bit 选择输入作为控制信号
- Example: **2:1 Mux**
- 内部电路?自己实现



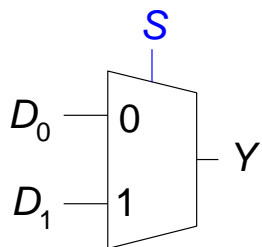
S	D ₁	D ₀	Y	S	Y
0	0	0	0	0	D ₀
0	0	1	1	1	D ₁
0	1	0	0		
0	1	1	1		
1	0	0	0		
1	0	1	0		
1	1	0	1		
1	1	1	1		

选择器的实现---自己实现



• 逻辑门

- 与或式

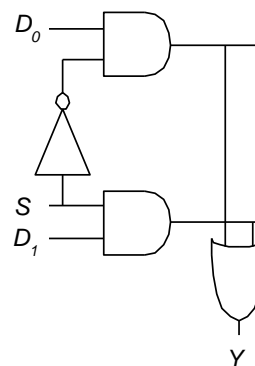


S	D ₁	D ₀	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

S	Y
0	D ₀
1	D ₁

Y	D ₀ D ₁	00	01	11	10
S	0	0	0	1	1
1	0	1	1	0	

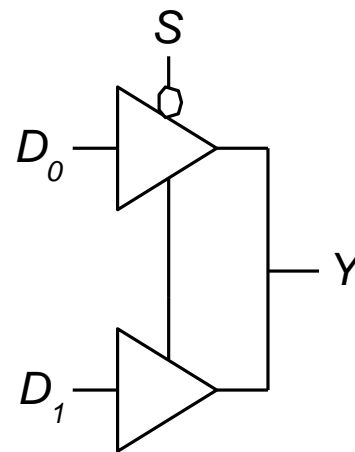
$$Y = D_0 \bar{S} + D_1 S$$



选择器的实现



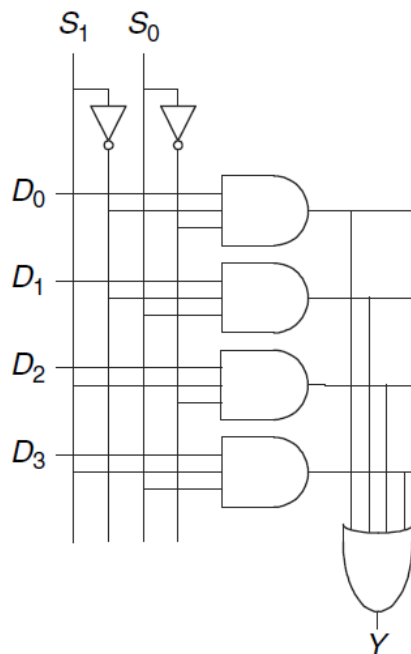
- 三态
- 选择器可以由三态缓冲器构成
 - 三态缓冲器使能信号有效使得在任何时刻仅有一个三态缓冲器有效
 - $S=0$, D_0 有效
 - $S=1$, D_1 有效



多路选择器的实现

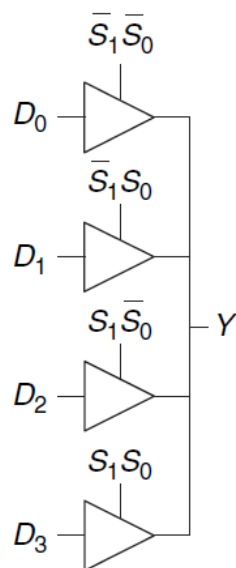


• 4:1 多路选择器的实现



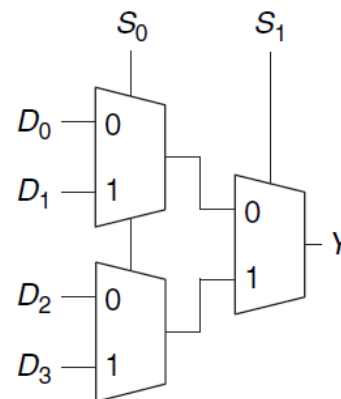
(a)

两级逻辑



(b)

三态缓冲器



(c)

层次结构

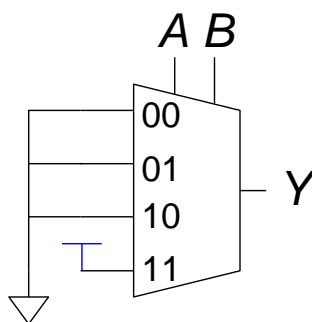
选择器逻辑



使用选择器实现其他逻辑，类似基本门单元

<i>A</i>	<i>B</i>	<i>Y</i>
0	0	0
0	1	0
1	0	0
1	1	1

$$Y = AB$$



用一个4:1选择器实现2输入与门（能否化简？）

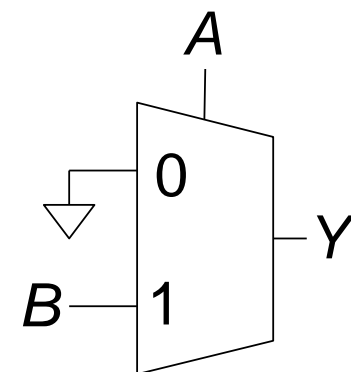
选择器逻辑



降低 mux 的规模 (化简)

$$Y = AB$$

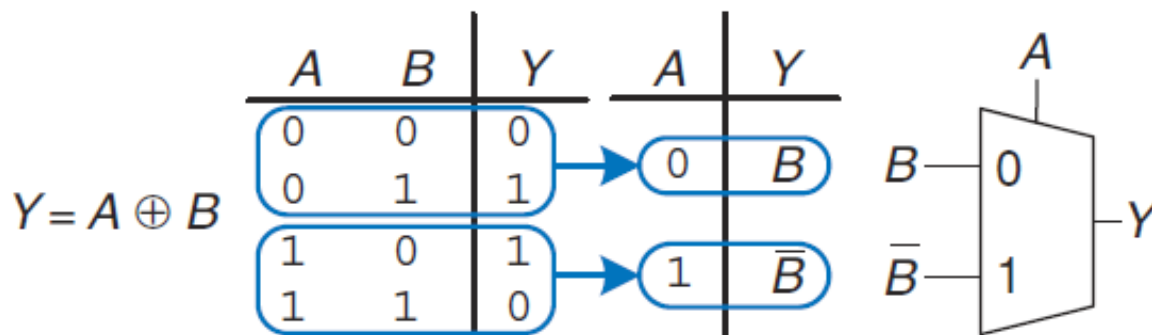
A	B	Y	A	Y
0	0	0	0	0
0	1	0		
1	0	0	1	B
1	1	1		



选择器逻辑



降低 mux 的规模 (化简)



使用选择器实现逻辑 1



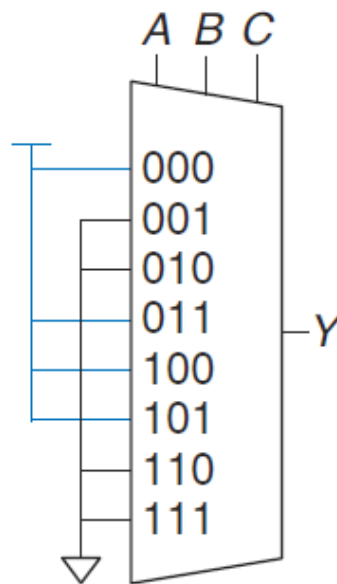
例子：某同学需实现逻辑 $Y = A\bar{B} + \bar{B}\bar{C} + \bar{A}BC$

但发现目前只有8:1复用器，再无其它逻辑器件（真值表）

A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

$$Y = A\bar{B} + \bar{B}\bar{C} + \bar{A}BC$$

(a)



(b)

使用选择器实现逻辑 2

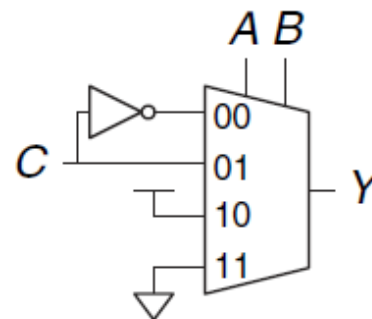


例子：某同学需实现逻辑 $Y = A\bar{B} + \bar{B}\bar{C} + \bar{A}BC$

如果只有4:1选择器和一个非门呢，再无其它逻辑器件

A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

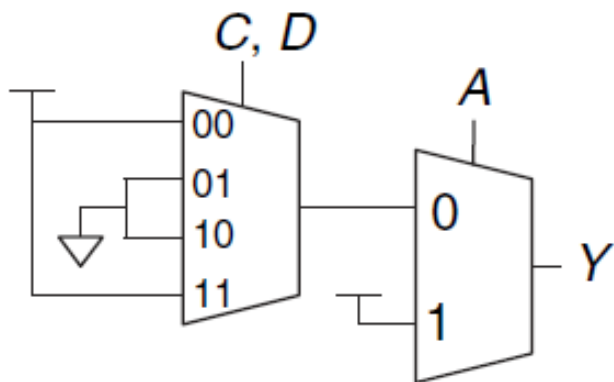
A	B	Y
0	0	\bar{C}
0	1	C
1	0	1
1	1	0



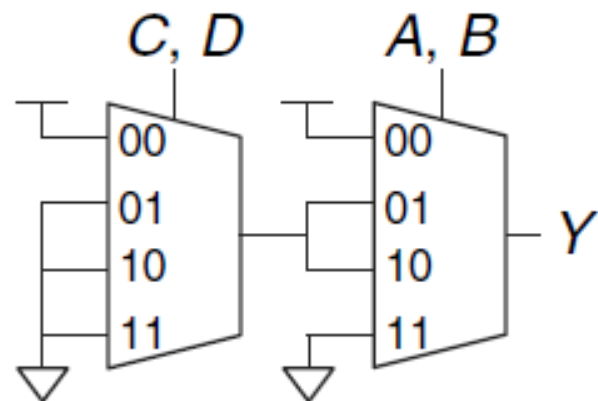
使用选择器实现逻辑 3



写出选择器的表达式



$$Y = A + \overline{C \oplus D} = A + CD + \overline{CD}$$

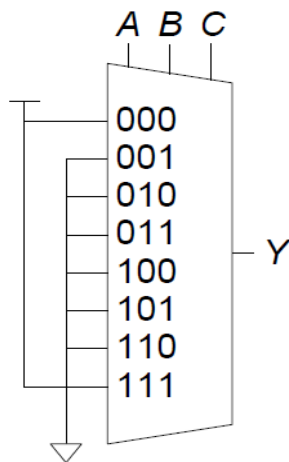


$$Y = \overline{CD}(A \oplus B) + \overline{AB} = \overline{A}CD + \overline{B}CD + \overline{AB}$$

使用选择器实现逻辑 4

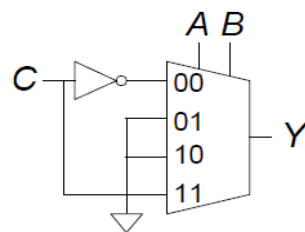


A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1



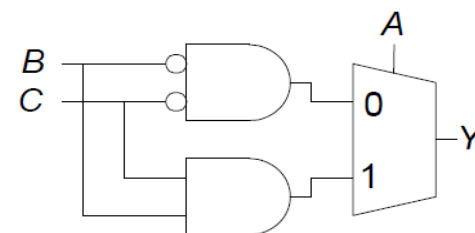
8:1选择器实现

A	B	Y
0	0	\overline{C}
0	1	0
1	0	0
1	1	C



4:1选择器实现

A	Y
0	\overline{BC}
1	BC



2:1选择器实现

选择器systemverilog实现



主模块：

```
module mux4(a,b,c,y);  
input a,b,c;  
output y;  
wire y;  
assign y= ((~a)&(~b)&(~c)) | (a&b&c);  
endmodule
```

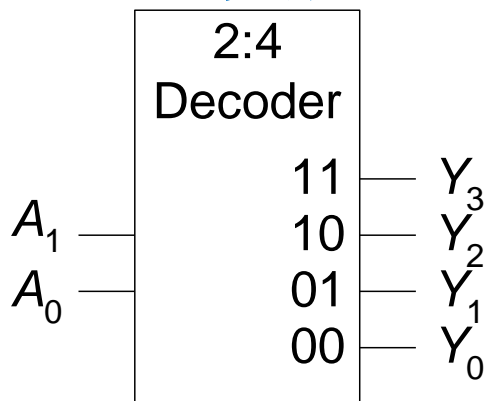
测试模块：

```
module test_mux4;  
reg a,b,c;  
initial begin  
a = 1'b0; b=1'b0; c=1'b0;  
#10 a = 1'b1; b=1'b1; c=1'b0;  
#10 a = 1'b0; b=1'b1; c=1'b0;  
#10 a = 1'b1; b=1'b1; c=1'b1;  
#10 $stop;  
end  
mux4 n1(a, b, c, y);  
endmodule
```

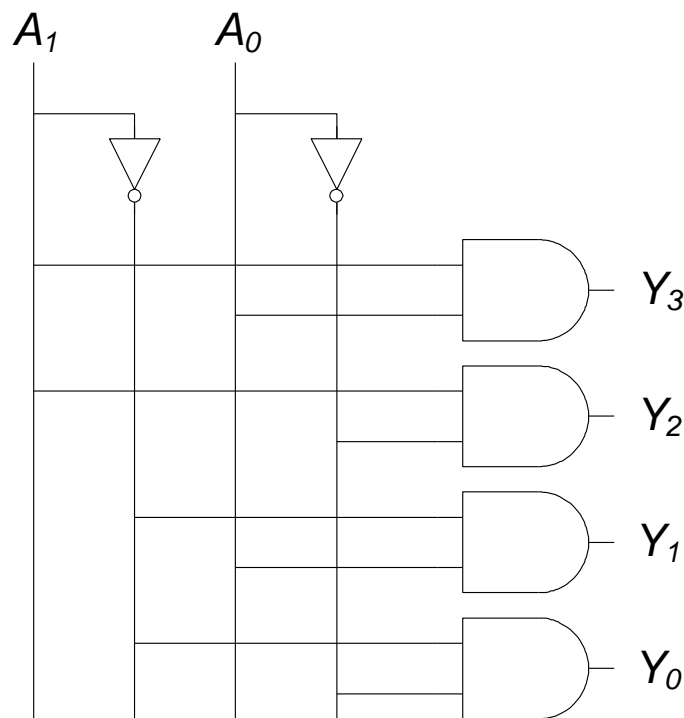

译码器



- N 输入, 2^N 输出, 每一个输出取决于输入组合
- One-hot (独热) 输出: 每次只有一位为高



A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

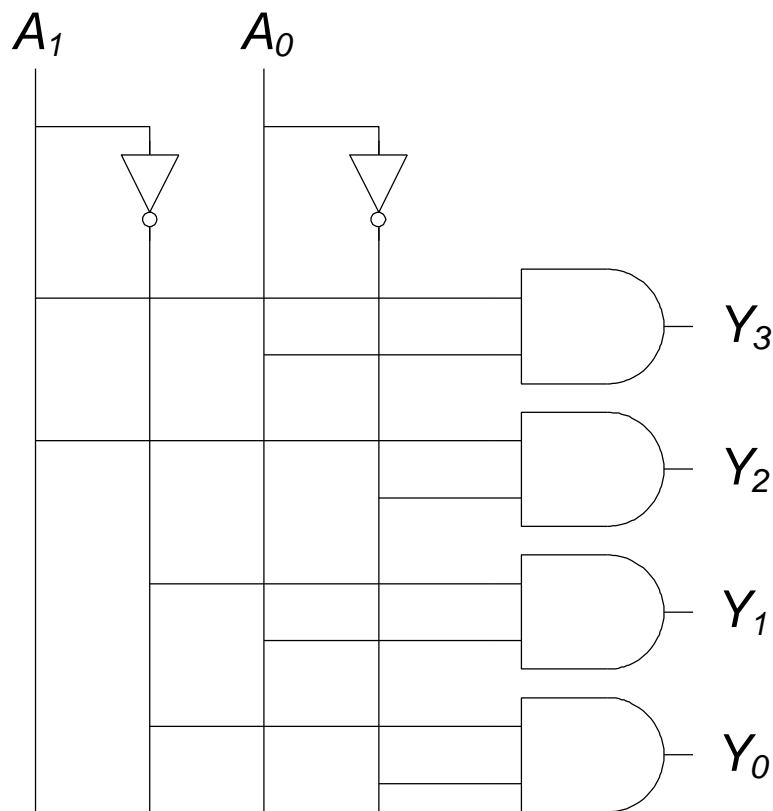


译码器实现



一个 $N:2^N$ 的译码器可以由 2^N 个 N 输入与门通过接收所有输入的值形式或取反形式的各种组合来构成，译码器的每一个输出代表一个最小项

A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0



译码器systemverilog实现



主模块：

```
module decode4(a1,a0,y3,y2,y1,y0);  
input  a1,a0;  
output y3,y2,y1,y0;  
wire   y3,y2,y1,y0;  
assign y0= ~a1 & ~a0;  
assign y1= ~a1 & a0;  
assign y2= a1  & ~a0;  
assign y3= a1 & a0;  
endmodule
```

测试模块：

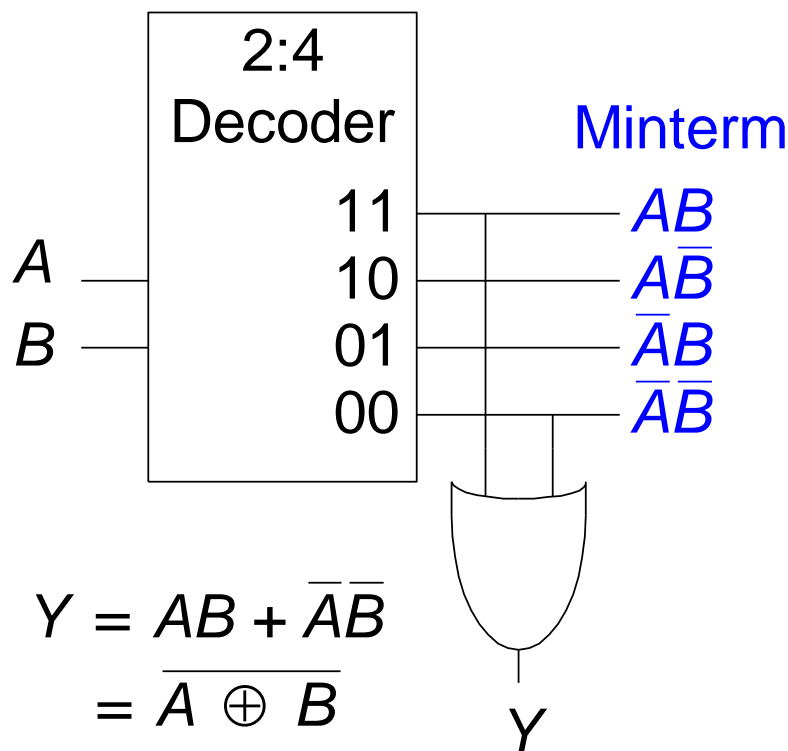
```
module test_decode4;  
reg    a1,a0;  
initial begin  
        a1 = 1'b0; a0=1'b0;  
#10    a1 = 1'b0; a0=1'b1;  
#10    a1 = 1'b1; a0=1'b0;  
#10    a1 = 1'b1; a0=1'b1;  
#10    $stop;  
end  
decode4  n1(a1, a0, y3,y2,y1, y0);  
endmodule
```

译码器实现逻辑



译码器可以和 **或门** 组合来实现逻辑。如实现2输入XNOR（异或非）。
因为译码器的**每一个输出都代码一个最小项**，所以函数将以所有最小项的**或**来实现。（与或表达式）
($A'B+AB'$)










a	b	$a \oplus b$
1	0	1
1	1	0
0	0	0
0	1	1



第二章 组合逻辑电路



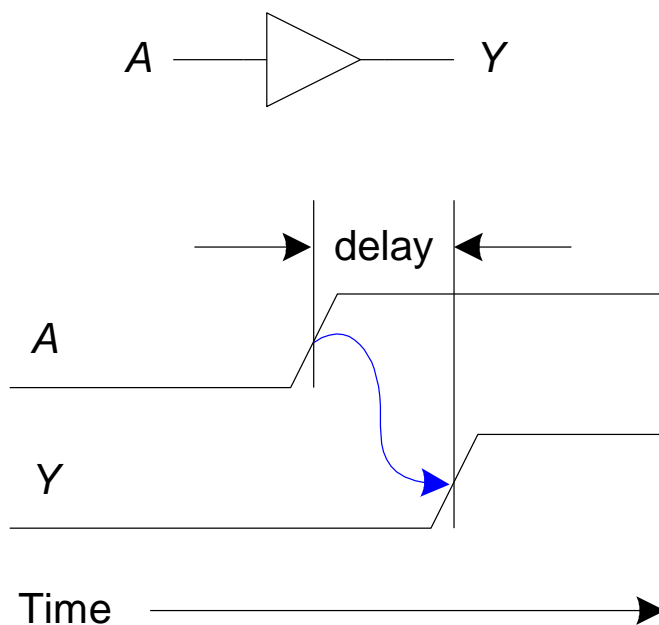
- 介绍（组合逻辑电路，）
- 布尔表达式（最小项、最大项、与或 或与
- 布尔代数（真值表、布尔表达式、电路、
- 从逻辑到门
- 多级组合逻辑
- X和Z
- 卡诺图
- 组合逻辑块（半加器，全加器，优先电路，7段译码器，复用器，译码器）
- 时序

Application Software	 >"hello world!"
Operating Systems	
Architecture	
Micro-architecture	
Logic	
Digital Circuits	
Analog Circuits	
Devices	
Physics	

时序



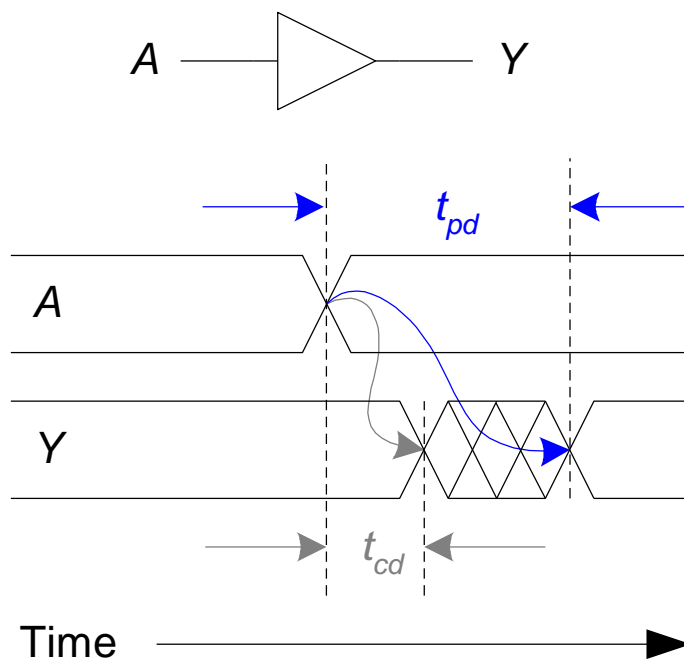
- 延迟: 输入与输出直接的时差 (原因 电容 电路)
- 学习时序是为了去构建更快的电路



传输延迟 & 最小延迟



- **传输延迟:** $t_{pd} = \max$ delay from input to output (所有) : 当输入改变直到一个或多个输出达到它们的最终值所经历最长时间
- **最小延迟:** $t_{cd} = \min$ delay from input to output (任意一个) : 当一个输入发送变化直到任何一个输出开始改变的最短时间

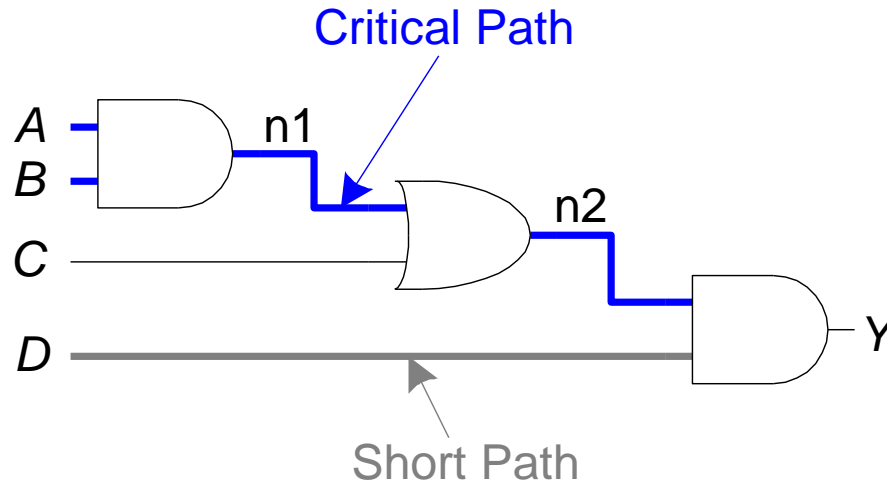


传输延迟& 最小延迟



- 原因
 - 电容/电阻引起
 - 电信号传播也需要时间
- 为什么 t_{pd} and t_{cd} 可能不同:
 - 上升、下降沿不同
 - 多个输入输出之间延迟不同
 - 温度影响：电路冷会变快，热越慢

关键路径 (Long) & 最短 Paths



Critical (Long) Path (每个之和) : $t_{pd} = 2t_{pd_AND} + t_{pd_OR}$

它是一条最长的路径，也是最慢的路径，这条路径是关键路径，因为它限制了电路运行的速度。

Short Path (最小的之和) : $t_{cd} = t_{cd_AND}$

从某个输入到输出的最短的路径，即最快路径

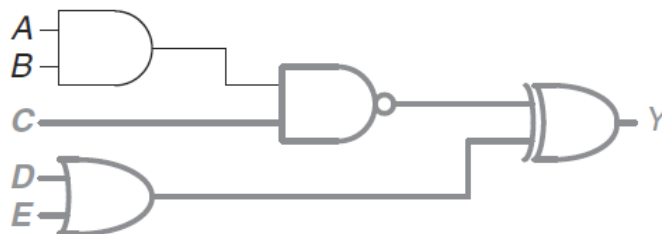
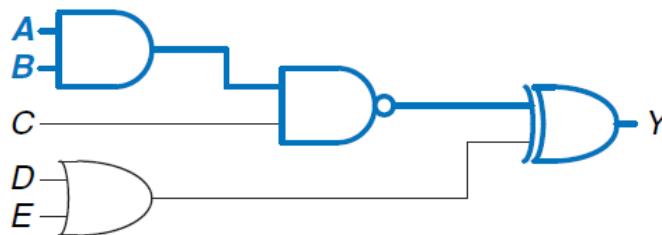
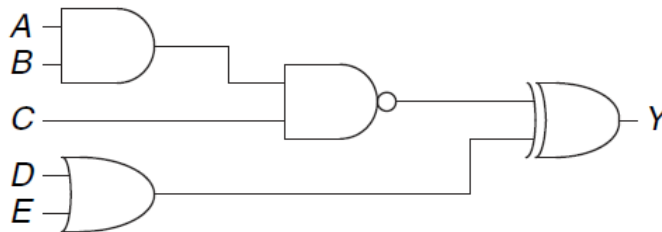
计算题



已知每个门的传输延迟和
最小延迟分别是100ps, 60ps

求电路1的传输延迟和最小延迟

关键路径是从A或B通过三个门到
达输出Y



计算题

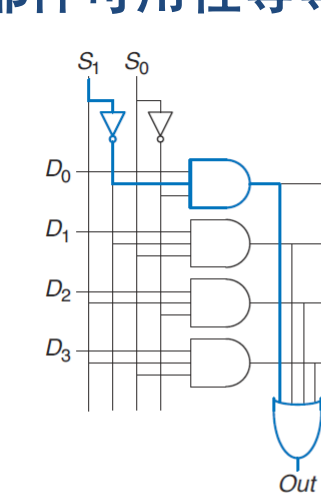


求选择器的最坏情况延迟？

如果数据输入在控制输入前到达？ 3

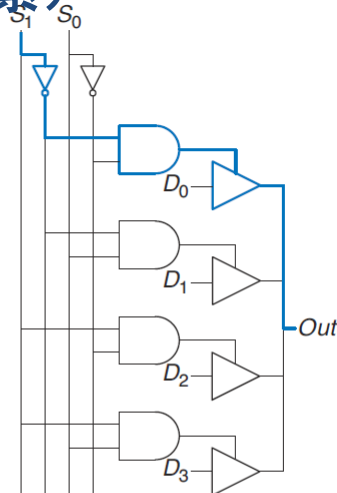
如果控制信号在数据信号之前到达？ 2（原则总时间最短！但还要看实际情况中的功耗、成本、部件可用性等等因素）

Gate	t_{pd} (ps)
NOT	30
2-input AND	60
3-input AND	80
4-input OR	90
tristate (A to Y)	50
tristate (enable to Y)	35



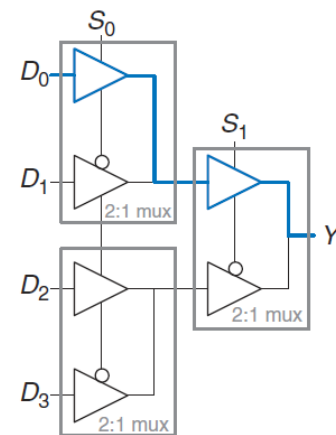
$$\begin{aligned}
 t_{pd_sy} &= t_{pd_INV} + t_{pd_AND3} + t_{pd_OR4} \\
 &= 30 \text{ ps} + 80 \text{ ps} + 90 \text{ ps} \\
 &= \mathbf{200 \text{ ps}} \\
 t_{pd_dy} &= t_{pd_AND3} + t_{pd_OR4} \\
 &= \mathbf{170 \text{ ps}}
 \end{aligned}$$

(a)



$$\begin{aligned}
 t_{pd_sy} &= t_{pd_INV} + t_{pd_AND2} + t_{pd_TRI_sy} \\
 &= 30 \text{ ps} + 60 \text{ ps} + 35 \text{ ps} \\
 &= \mathbf{125 \text{ ps}} \\
 t_{pd_dy} &= t_{pd_TRI_ay} \\
 &= \mathbf{50 \text{ ps}}
 \end{aligned}$$

(b)



$$\begin{aligned}
 t_{pd_s0y} &= t_{pd_TRLSY} + t_{pd_TRI_AY} = \mathbf{85 \text{ ns}} \\
 t_{pd_dy} &= 2 t_{pd_TRI_AY} = \mathbf{100 \text{ ns}}
 \end{aligned}$$

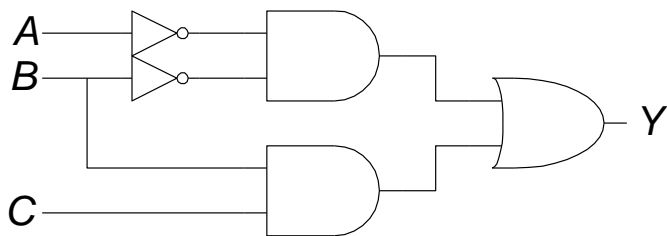


- 当一个输入信号的改变可能会导致多个输出信号改变 (或者叫冒险hazard)

毛刺例子

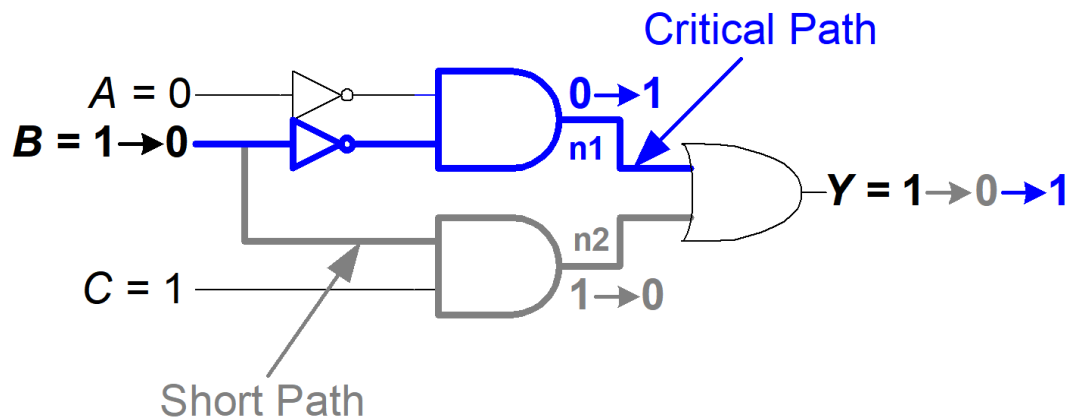


- 当 $A = 0, C = 1, B$ falls 从1变0?

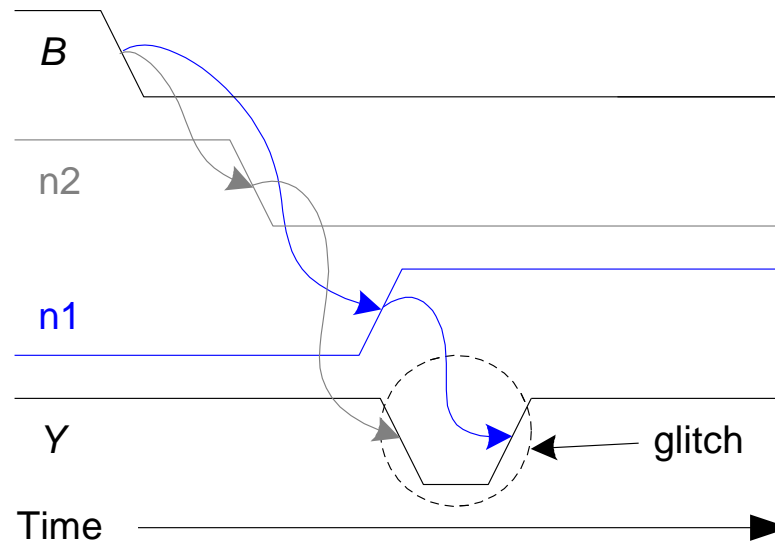
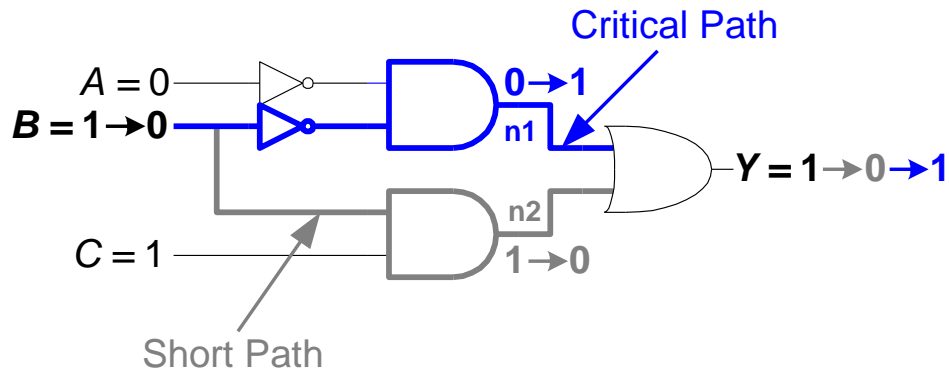


Y		AB			
		00	01	11	10
C	0	1	0	0	0
	1	1	1	1	0

$$Y = \bar{A}\bar{B} + BC$$



毛刺例子 (cont.)



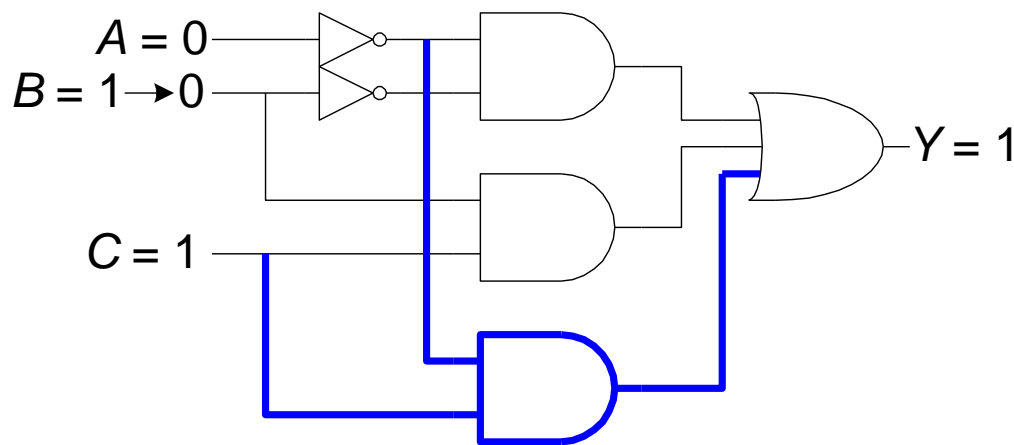
修正 (增加电路, 蕴涵项覆盖)



Y		AB			
		00	01	11	10
C	0	1	0	0	0
	1	1	1	1	0

$\bar{A}C$

$$Y = \bar{A}\bar{B} + BC + \bar{A}C$$



总结



- 1、数字电路是一个带离散电压值输入和输出的模块；
- 2、组合电路功能可以通过真值表或布尔表达式确定；用与或式、或与式表示；
- 3、与或式：一个或多个蕴含项（各个项的与）进行或；
- 4、布尔表达式用布尔代数（定理、公理）来化简，也可以根据卡诺图K-map化简；
- 5、逻辑门可以组合成更大规模的电路，如选择器、译码器、加法器等；
- 6、组合电路时序包含传输延迟和最小延迟，存在关键路径（传输延迟最大的路径）。

作业题



**课后题：2.1(课堂已做)， 2.17， 2.33，
2.41， 2.45**

以上5道本章典型的题需要书面做。

其它课后题大家自己边复习本章内容边练习。熟能生巧，大家抽时间练习。

课后题



2.1

Exercise 2.1

(a) $Y = \bar{A}\bar{B} + A\bar{B} + AB$

(b) $Y = \bar{A}\bar{B}\bar{C} + ABC$

(c) $Y = \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + A\bar{B}\bar{C} + A\bar{B}C + ABC$

(d)

$$Y = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + \bar{A}B\bar{C}\bar{D} + \bar{A}B\bar{C}D + A\bar{B}\bar{C}\bar{D} + A\bar{B}C\bar{D} + ABC\bar{D}$$

(e)

$$Y = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + \bar{A}B\bar{C}\bar{D} + \bar{A}B\bar{C}D + A\bar{B}\bar{C}\bar{D} + A\bar{B}C\bar{D} + ABC\bar{D} + ABCD$$

课后题

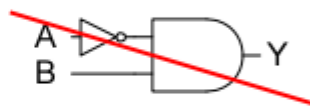


2.17

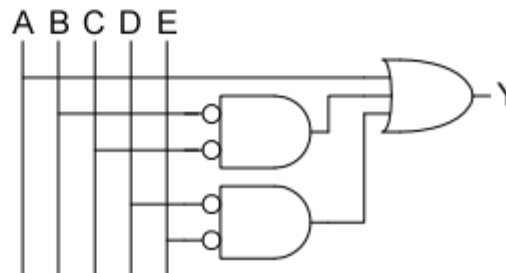
(a) $Y = B + \overline{A}\overline{C}$



~~(b) $Y = \overline{A}B$~~ 0



(c) $Y = A + \overline{B}\overline{C} + \overline{D}\overline{E}$



课后题



$$2.17(c) Y = ABC + ABD + ABE + ACD + ACE + (A + D + E) + \overline{BCD} + \overline{BCE} + \overline{BDE} + \overline{CDE}$$

$$Y = ABC + AB(D + E) + AC(D + E) + \overline{A}\overline{D}\overline{E} + \overline{B}\overline{C}(D + E) + (\overline{B} + \overline{C})\overline{D}\overline{E}$$

$$Y = ABC + (D + E)(\overline{AB} + \overline{AC} + \overline{BC}) + (\overline{A} + \overline{B} + \overline{C})\overline{D}\overline{E}$$

$$Y = ABC + \overline{D}\overline{E}(\overline{AB} + \overline{BC}) + \overline{ABC}\overline{D}\overline{E}$$

$$Y = \overline{ABC} + \overline{D}\overline{E}(A + \overline{BC}) + \overline{ABC}\overline{D}\overline{E}$$

$$Y = \overline{ABC} + \overline{D}\overline{E} + \overline{D}\overline{E}(A + \overline{BC})$$

$$Y = \overline{ABC} + \overline{D}\overline{E} + \overline{A} + \overline{BC}$$

$$Y = A + \overline{BC} + \overline{D}\overline{E}$$

$$\overline{PA} + A = \overline{P} + A$$

$$\overline{PA} + \overline{A} = \overline{P} + \overline{A}$$

$$2.33 E = \overline{S}\overline{A} + SH$$

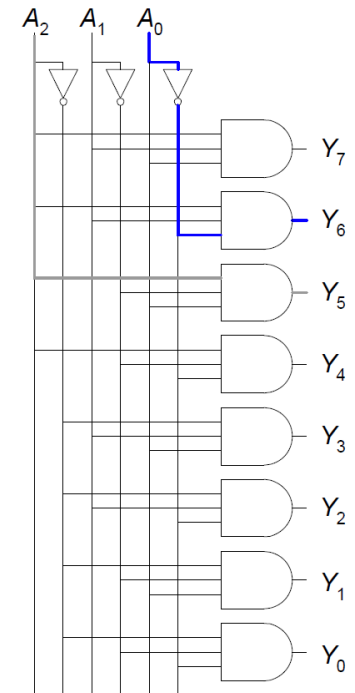
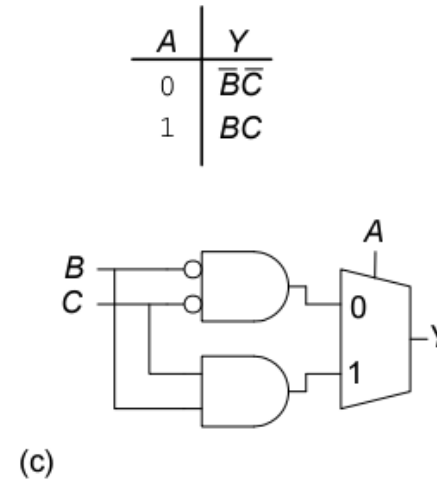
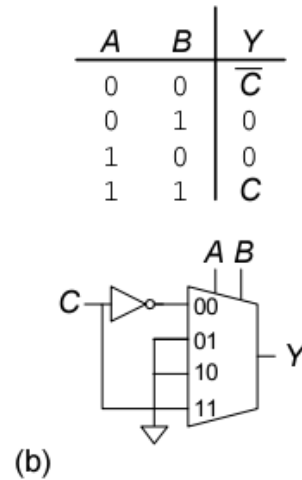
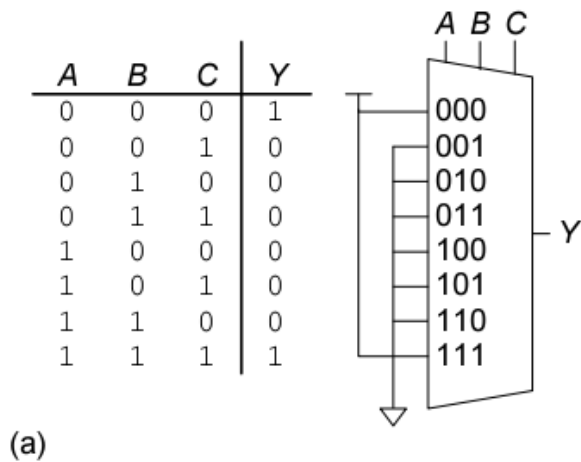
课后题



2.45

$$\begin{aligned}
 t_{pd} &= t_{pd_NOT} + t_{pd_AND3} \\
 &= 15 \text{ ps} + 40 \text{ ps} \\
 &= 55 \text{ ps} \\
 t_{cd} &= t_{cd_AND3} \\
 &= 30 \text{ ps}
 \end{aligned}$$

2.41



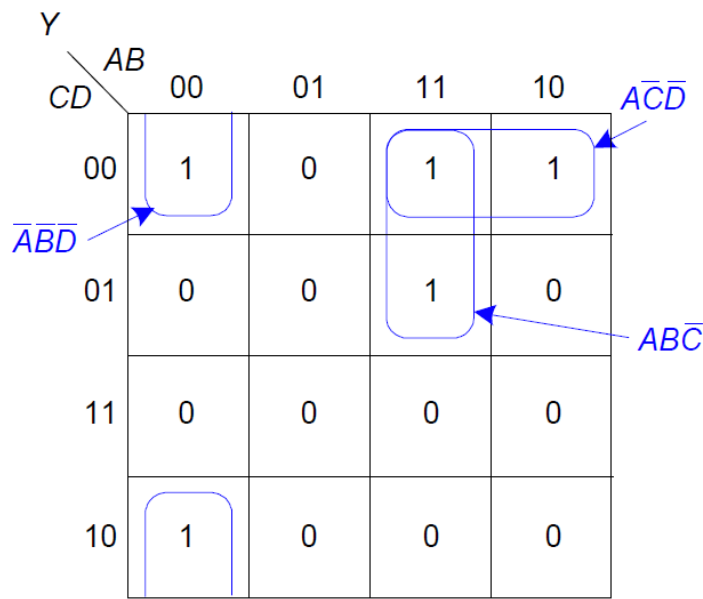


本章结束

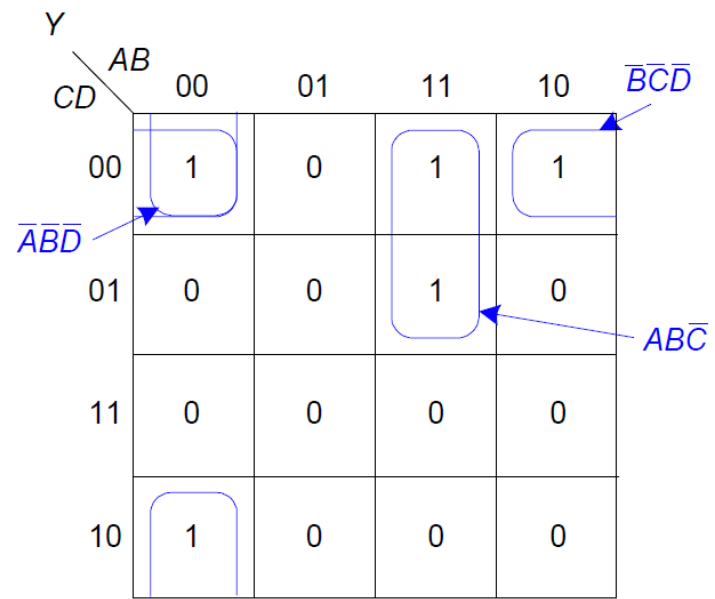
课后题



2.21



$$Y = \bar{A}\bar{B}\bar{D} + A\bar{B}\bar{C} + \bar{A}\bar{C}\bar{D}$$



$$Y = \bar{A}\bar{B}\bar{D} + A\bar{B}\bar{C} + \bar{B}\bar{C}\bar{D}$$

课后题



2.24

$$Y = \bar{A}D + \bar{A}\bar{B}C + A\bar{C}D + ABCD$$
$$Z = A\bar{C}D + BD$$

课后题



2.35

Decimal Value	A_3	A_2	A_1	A_0	D	P
0	0	0	0	0	0	0
1	0	0	0	1	0	0
2	0	0	1	0	0	1
3	0	0	1	1	1	1
4	0	1	0	0	0	0
5	0	1	0	1	0	1
6	0	1	1	0	1	0
7	0	1	1	1	0	1
8	1	0	0	0	0	0
9	1	0	0	1	1	0
10	1	0	1	0	0	0
11	1	0	1	1	0	1
12	1	1	0	0	1	0
13	1	1	0	1	0	1
14	1	1	1	0	0	0
15	1	1	1	1	1	0