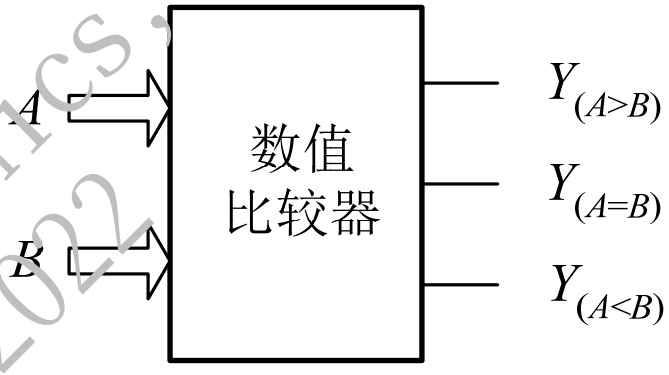
§ 4.6 比较器 Comparators

功能：比较两个无符号二进制数的大小



§ 4.6.1 一位比较器 One Bit Comparator

输入： A, B

输出：比较结果

- $L\ (A>B)$ Large
- $S\ (A<B)$ Small
- $E\ (A=B)$ Equal

高电平有效

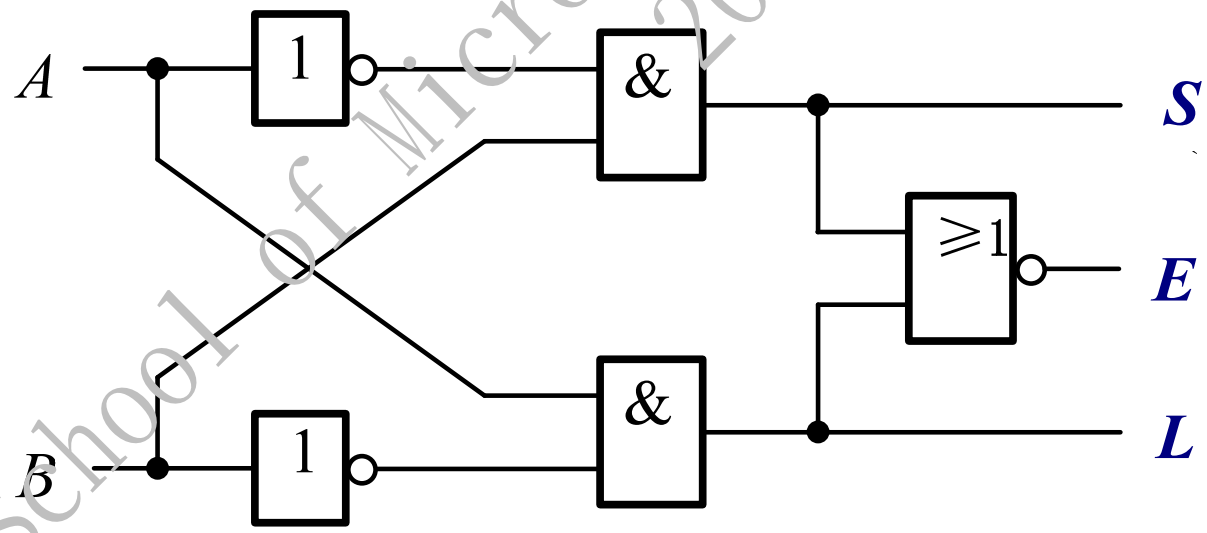
Yes 1

No 0

真值表

A	B	L	S	E
0	0	0	0	1
0	1	0	1	0
1	0	1	0	0
1	1	0	0	1

$$\begin{cases} L = A\overline{B} \\ S = \overline{A}B \\ E = AB + \overline{A}\cdot\overline{B} \\ \quad = A \odot B \end{cases}$$



逻辑电路图

§ 4.6.2 四位比较器 Four Bits Comparator

8 输入 $\begin{cases} A: A_3A_2A_1A_0 \\ B: B_3B_2B_1B_0 \end{cases}$

3 输出 $\begin{cases} L\ (A>B) \\ S\ (A<B) \\ E\ (A=B) \end{cases}$

从最高位开始比较

中规模4位比较器芯片

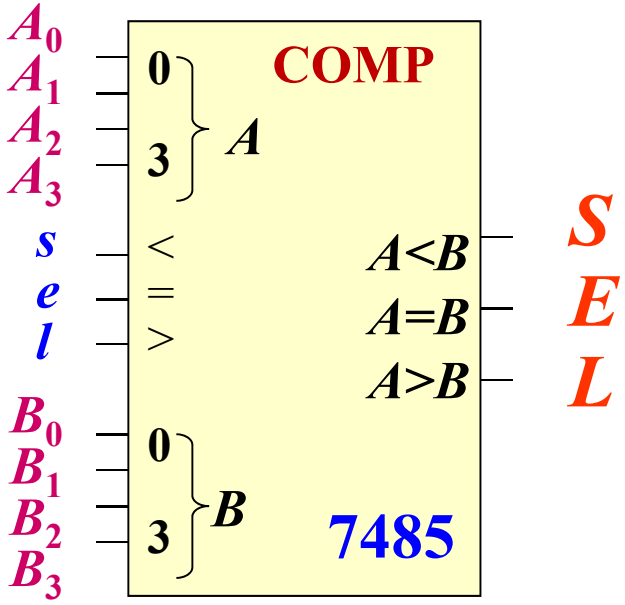
74LS85符号及逻辑功能

3个级联输入端

$\begin{cases} l\ (A>B) \\ s\ (A<B) \\ e\ (A=B) \end{cases}$

来自低位的比较结果

IEEE



比较输入				级联输入			输出		
$A_3\ B_3$	$A_2\ B_2$	$A_1\ B_1$	$A_0\ B_0$	$I_{(A>B)}$	$I_{(A<B)}$	$I_{(A=B)}$	$Y_{(A>B)}$	$Y_{(A<B)}$	$Y_{(A=B)}$
$A_3>B_3$	×	×	×	×	×	×	1	0	0
$A_3<B_3$	×	×	×	×	×	×	0	1	0
$A_3=B_3$	$A_2>B_2$	×	×	×	×	×	1	0	0
$A_3=B_3$	$A_2<B_2$	×	×	×	×	×	0	1	0
$A_3=B_3$	$A_2=B_2$	$A_1>B_1$	×	×	×	×	1	0	0
$A_3=B_3$	$A_2=B_2$	$A_1<B_1$	×	×	×	×	0	1	0
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0>B_0$	×	×	×	1	0	0
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0<B_0$	×	×	×	0	1	0
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	1	0	0	1	0	0
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	0	1	0	0	1	0
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	0	0	1	0	0	1

比较输入				级联输入			输出		
$A_3\ B_3$	$A_2\ B_2$	$A_1\ B_1$	$A_0\ B_0$	$I_{(A>B)}$	$I_{(A<B)}$	$I_{(A=B)}$	$Y_{(A>B)}$	$Y_{(A<B)}$	$Y_{(A=B)}$
$A_3>B_3$	×	×	×	×	×	×	1	0	0
$A_3<B_3$	×	×	×	×	×	×	0	1	0
$A_3=B_3$	$A_2>B_2$	×	×	×	×	×	1	0	0
$A_3=B_3$	$A_2<B_2$	×	×	×	×	×	0	1	0
$A_3=B_3$	$A_2=B_2$	$A_1>B_1$	×	×	×	×	1	0	0
$A_3=B_3$	$A_2=B_2$	$A_1<B_1$	×	×	×	×	0	1	0
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0>B_0$	×	×	×	1	0	0
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0<B_0$	×	×	×	0	1	0
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	1	0	0	1	0	0
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	0	1	0	0	1	0
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	0	0	1	0	0	1

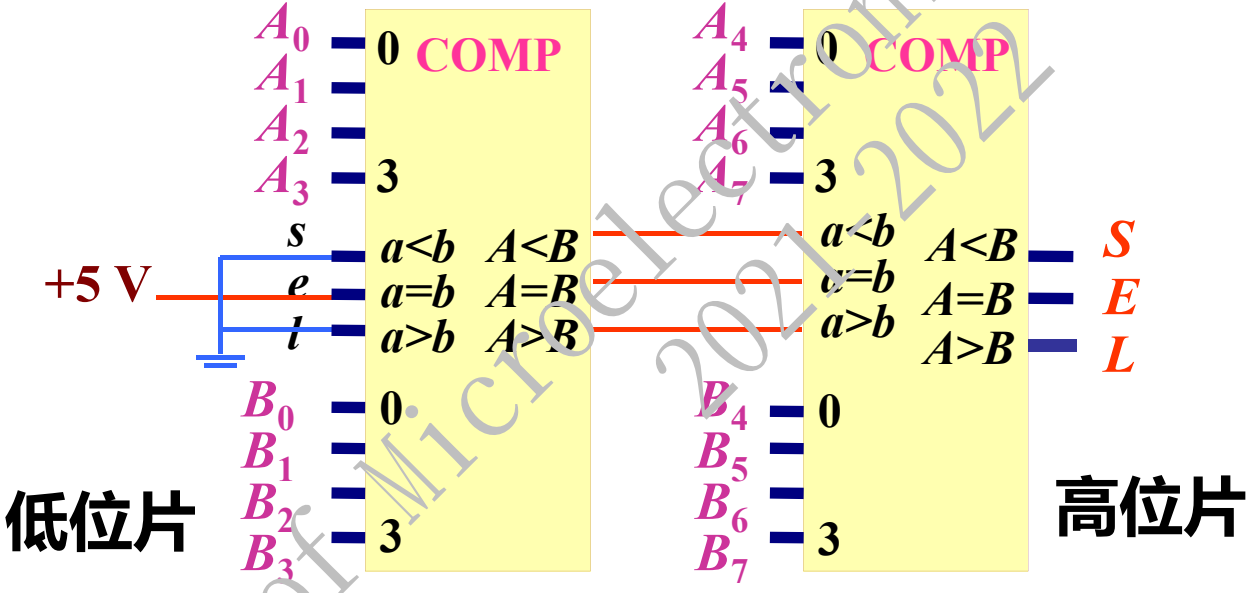
输出

$$\begin{cases} E = E_3E_2E_1E_0e \\ L = L_3 + E_3L_2 + E_3E_2L_1 + E_3E_2E_1L_0 + E_3E_2E_1E_0l \\ S = S_3 + E_3S_2 + E_3E_2S_1 + E_3E_2E_1S_0 + E_3E_2E_1E_0s \end{cases}$$

§ 4.6.3 比较器级联扩展

Cascading Comparators

2片7485 连成一个8位数值比较器

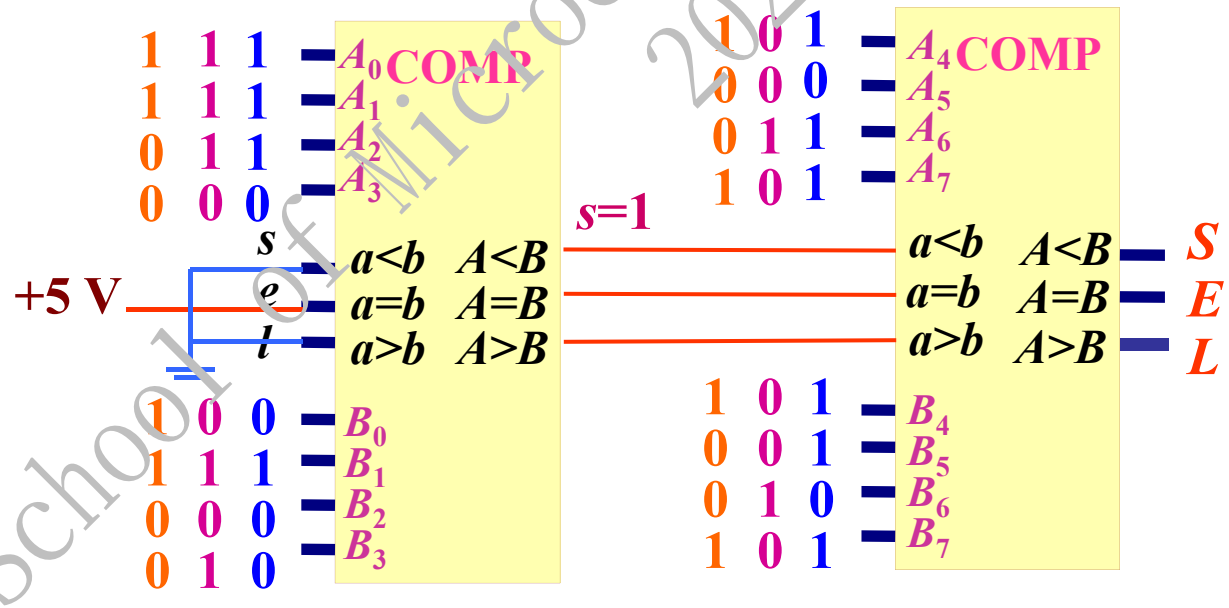


- 先用高位片，若高位片比出结果（$A>B$ or $A<B$), 则与级联输入状态无关
- 若高位片相等($A=B$)，再看级联输入，即看低位片比较结果 *l s e*
- 若低位仍相等，则$A=B$

例：比较 $\begin{cases} A=1\,1\,0\,1\,0\,1\,1\,1 \\ B=1\,0\,1\,1\,0\,0\,1\,0 \end{cases}$ **输出** $(S,E,L)=(0,0,1)$

$\begin{cases} A=0\,1\,0\,0\,0\,1\,1\,1 \\ B=0\,1\,0\,0\,1\,0\,1\,0 \end{cases}$ **输出** $(S,E,L)=(1,0,0)$

$\begin{cases} A=1\,0\,0\,1\,0\,0\,1\,1 \\ B=1\,0\,0\,1\,0\,0\,1\,1 \end{cases}$ **输出** $(S,E,L)=(s,e,l)=(0,1,0)$



§ 4.7 加法器 Adders

Adders are important not only in computer, but in many types of digital systems in which numerical data are processed.

§ 4.7.1 半加器 Half Adder

功能: 实现两个一位二进制数相加，不考虑低位进位

2 输入: A, B　　**2 输出**: S (sum)　C_o (carry out)

$$\begin{array}{rr} & A \\ + & B \\ \hline C_o & S \end{array}$$

A	B	S	C_o
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

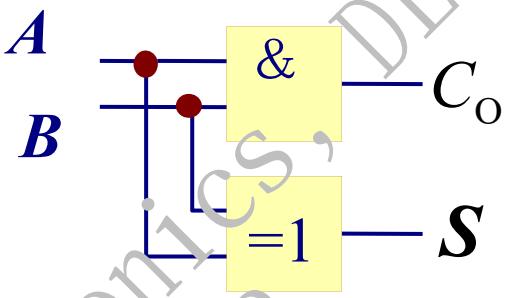
$$S = A \oplus B$$

$$C_o = AB$$

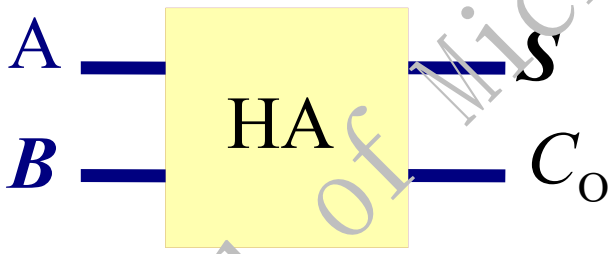
$S = A \oplus B$

$C_O = AB$

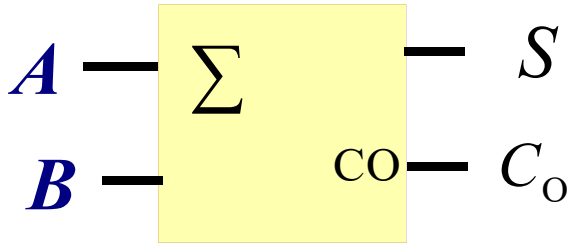
电路



符号:



IEEE



思考：一位半加器能构成N位的加法器吗?

$A=A_4A_3A_2A_1A_0$　　0 1 1 1 1　（15）$_{10}$

$A=B_4B_3B_2B_1B_0$　　0 1 0 1 0　（10）$_{10}$

1 1 1 0 ←——— 产生的进位

$S=S_4S_3S_2S_1S_0$　　1 1 0 0 1　（25）$_{10}$

结论：半加器没有考虑来自低位的进位，无法构成N位的加法器

§ 4.7.2 全加器 Full Adder

3 输入：A, B, C_i (来自低位的进位)

2 输出： S, C_{i+1} (向高位的进位)

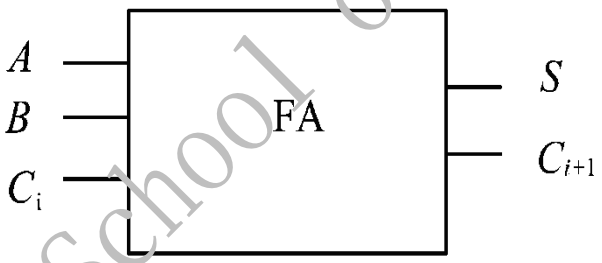
$$S = A \oplus B \oplus C_i$$ 奇数个1, 本位和为1

$$C_{i+1} = AB + AC_i + BC_i$$

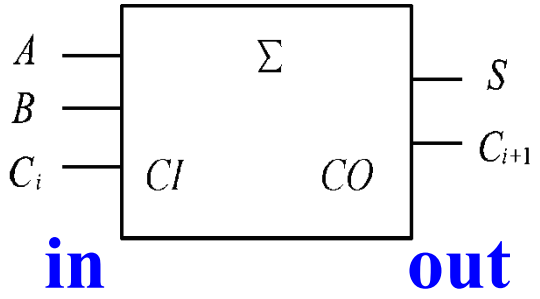
任何两个数为1, 进位

A B C_i	S C_{i+1}
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

符号



IEEE



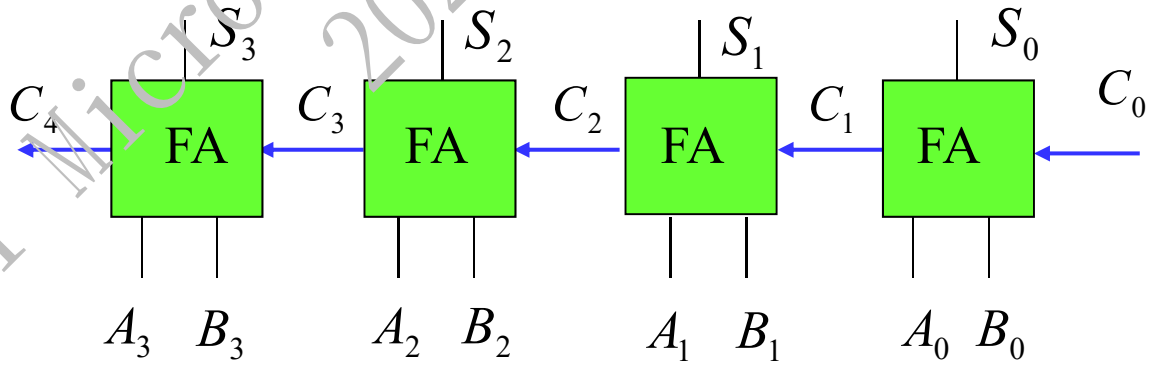
§ 4.7.3 并行加法器 Parallel Adder

多位二进制数相加时，每位一个全加器，加法器并行

并行加法器中进位方式 { **串行(脉冲)进位** / **超前进位**

串行进位（Ripple carry）

$$\begin{array}{r} A_3A_2A_1A_0 \\ +\quad B_3B_2B_1B_0 \\ \hline C_4S_3S_2S_1S_0 \end{array}$$



并行输入，串行进位: 结构简单, 速度慢

为提高运算速度，采用超前进位方法

超前进位 （Carry look-ahead）

分析全加器输出

$$S_i = A_i \oplus B_i \oplus C_i$$

$$C_{i+1} = \overline{A}_i B_i C_i + A_i \overline{B}_i C_i + A_i B_i \overline{C}_i + A_i B_i C_i$$

$$= A_i B_i + (A_i \oplus B_i) C_i$$

定义

$$\begin{cases} G_i = A_i B_i \\ P_i = A_i \oplus B_i \end{cases}$$

产生变量 (Carry generation)

传输变量（Carry propagation ）

全加器输出

$$\begin{cases} S_i = P_i \oplus C_i \\ C_{i+1} = G_i + P_i C_i \end{cases}$$

进位:

$$C_{i+1} = G_i + P_i C_i$$

$$C_1 = G_0 + P_0 C_0$$

$$C_2 = G_1 + P_1 C_1 = G_1 + P_1 G_0 + P_1 P_0 C_0$$

$$C_3 = G_2 + P_2 C_2 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0$$

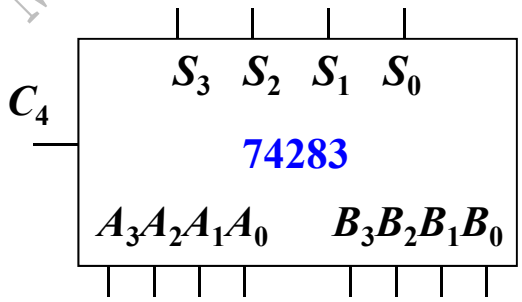
$$\begin{cases} G_i = A_i B_i \\ P_i = A_i \oplus B_i \end{cases}$$

∵ C_0 =0，C_i 只与G, P 有关，即只与 A, B 有关，可以并行产生

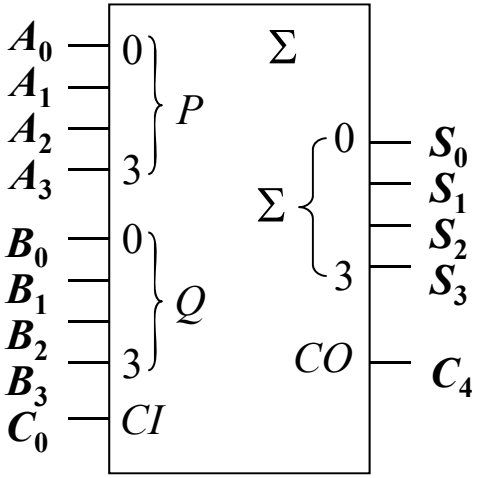
相当于四个全加器同时计算，提高速度

超前进位加法器74283:

$$\begin{array}{rr} & A_3\ A_2\ A_1\ A_0 \\ +) & B_3\ B_2\ B_1\ B_0 \\ \hline C_4 & S_3\ S_2\ S_1\ S_0 \end{array}$$



惯用符号

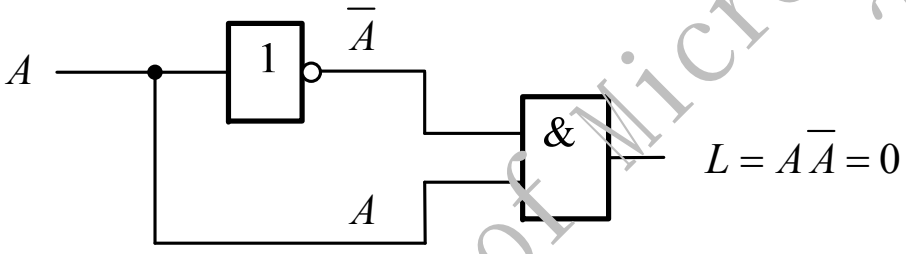


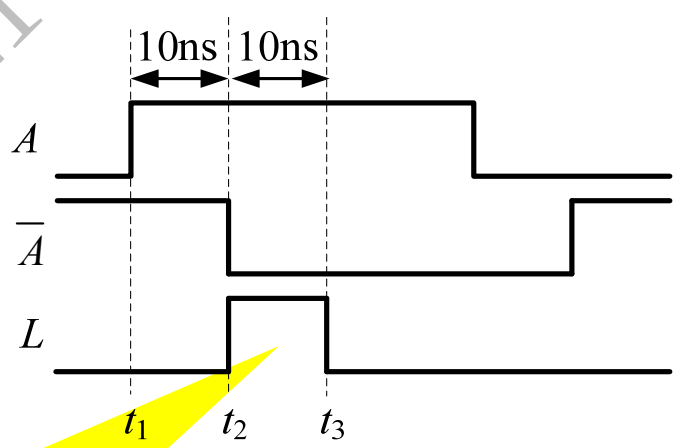
国际标准符号

§ 4.8 组合逻辑电路的竞争冒险

Race-Hazard of Combinational Logic

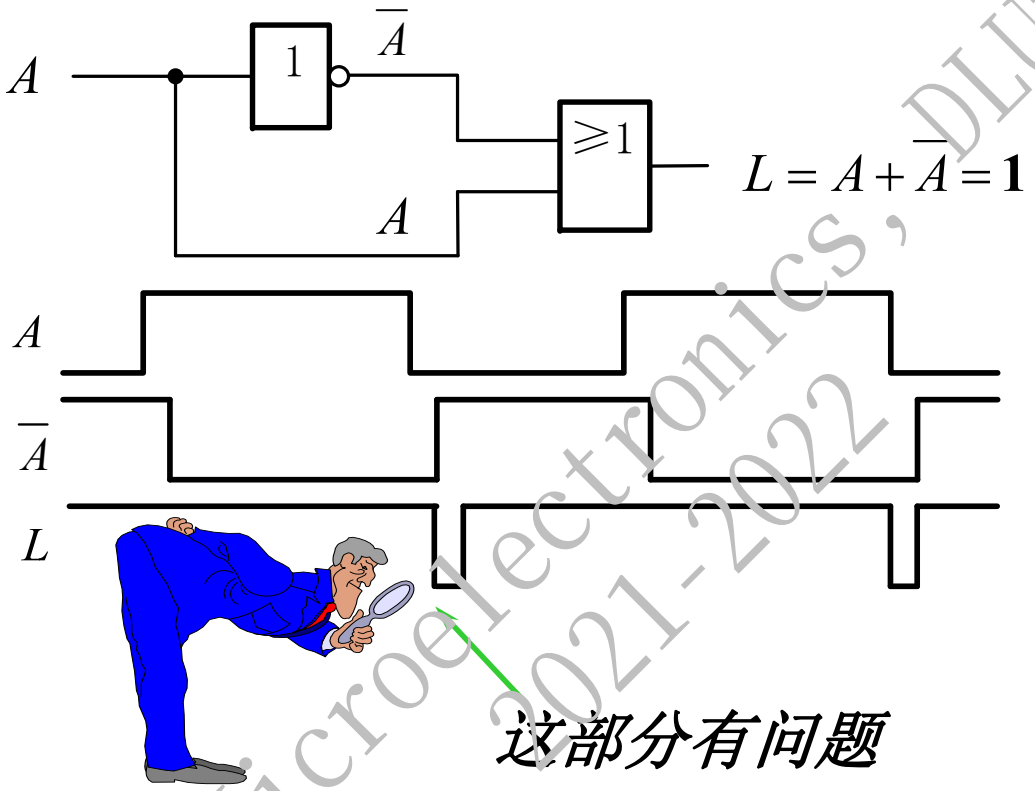
前面讨论电路是在理想的情况下进行的。为了保证工作的可靠性，要**考虑输入信号逻辑电平发生变化的瞬间电路的工作情况。**





在输出端产生尖峰干扰

由门电路的传输延时引起的问题



竞争：从输入到输出的途径不同，延时时间不同，到达输出端的时间不同，这种现象为竞争

冒险：输入信号变化导致逻辑电路产生错误输出，称为冒险或险象

§ 4.8.1 冒险现象的识别

1. 代数法

$$F = AB + \bar{A}C$$

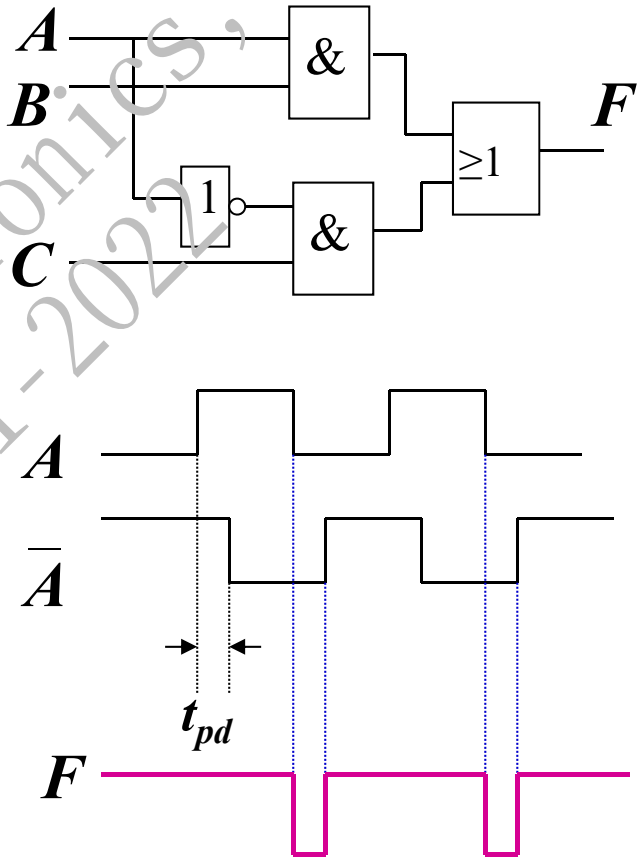
*A*到达终点的途径不同

当 $B = C = 1$ $F = A + \bar{A} = 1$

F 应该总是高电平

传播延时 *F* – 窄的负脉冲

“0”型冒险 (hazard)



毛刺 (glitch)

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

当 $B = C = 0$

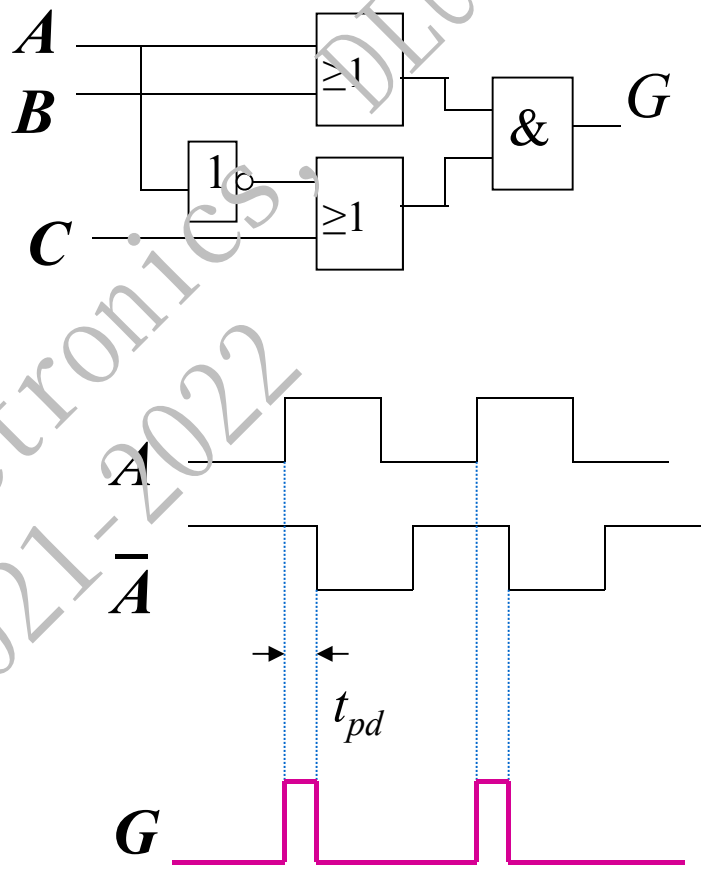
$G = A \cdot \overline{A} = 0$

G 应该总是低电平

传播延时

G – 窄的正脉冲

“1”型冒险

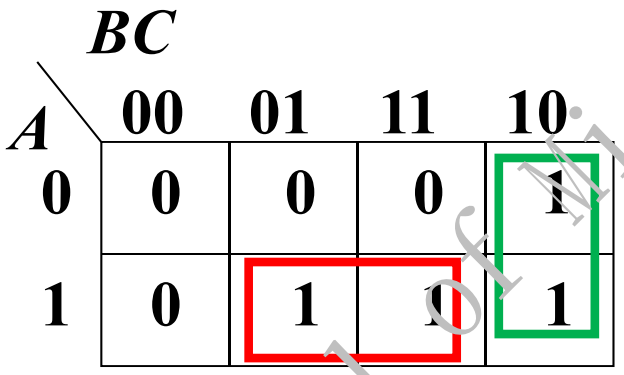


当函数表达式可以化成： $F = A + \overline{A}$ $F = A \cdot \overline{A}$

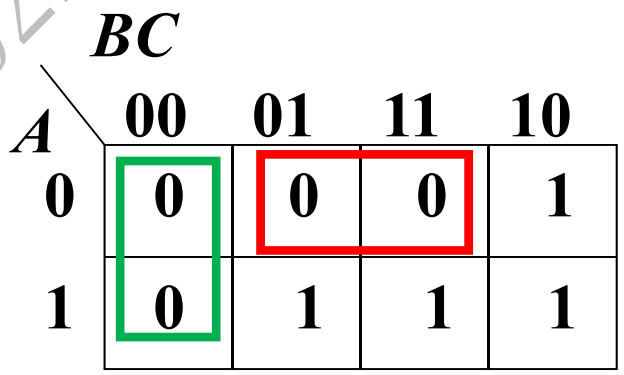
即含有互补变量，A变量变化可能引起冒险

2. 卡诺图法

如函数卡诺图上有包围圈相切，且相切处又无其他圈包含，则可能有险象



$$F = B\overline{C} + AC$$



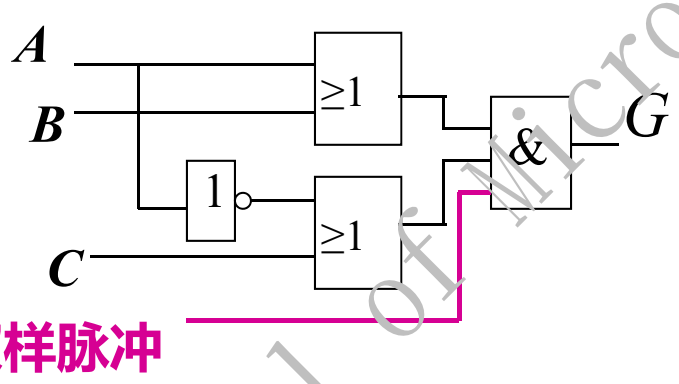
$$F = (B + C)(A + \overline{C})$$

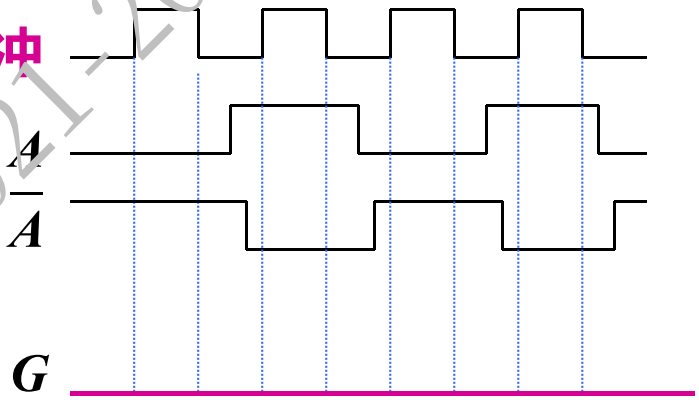
§4.8.2 冒险现象的消除

1. 接入滤波电容

竞争冒险引起的脉冲一般很窄(纳秒级)，在输出端并接一个滤波电容，将其滤掉。

2. 引入取样脉冲





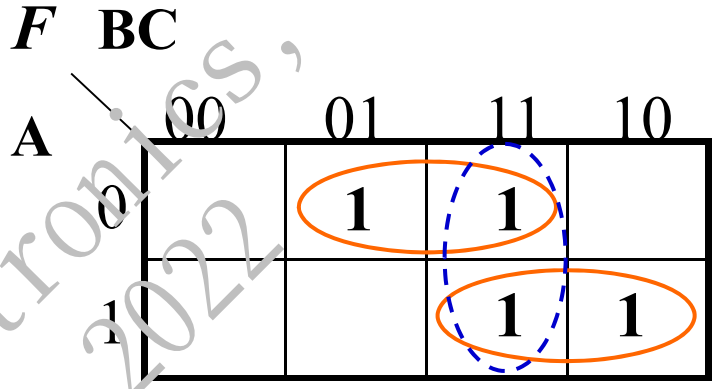
- 在输出端接取样脉冲，仅在输出处于稳定值时出现
- 取样脉冲为0期间，输出端信息无效

3. 修改设计方案

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

$$F = A \cdot \overline{A}$$

冒险



卡诺图两圈相切处增加一个圈（冗余），就能消除冒险

两个圈相切，存在冒险

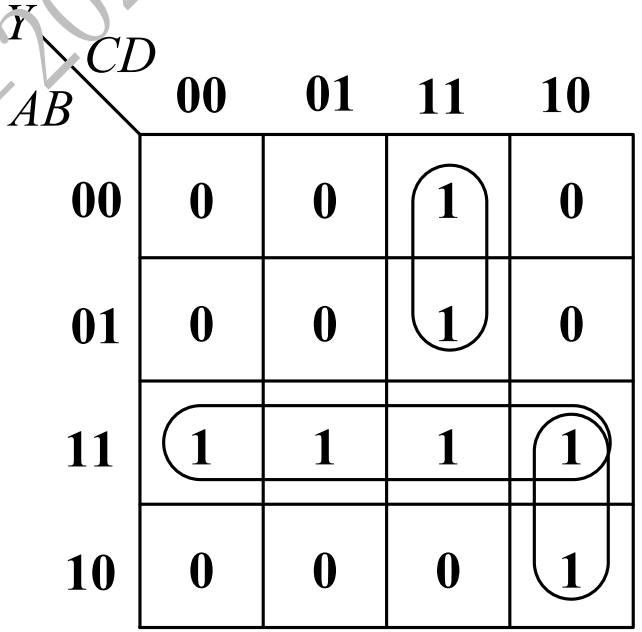
$$F = AB + \overline{A}C \Longrightarrow F = AB + \overline{A}C + BC$$

无竞争冒险

例1：已知逻辑函数真值表，用二输入与非门实现该逻辑函数，并分析竞争冒险现象

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

1. 逻辑函数化简

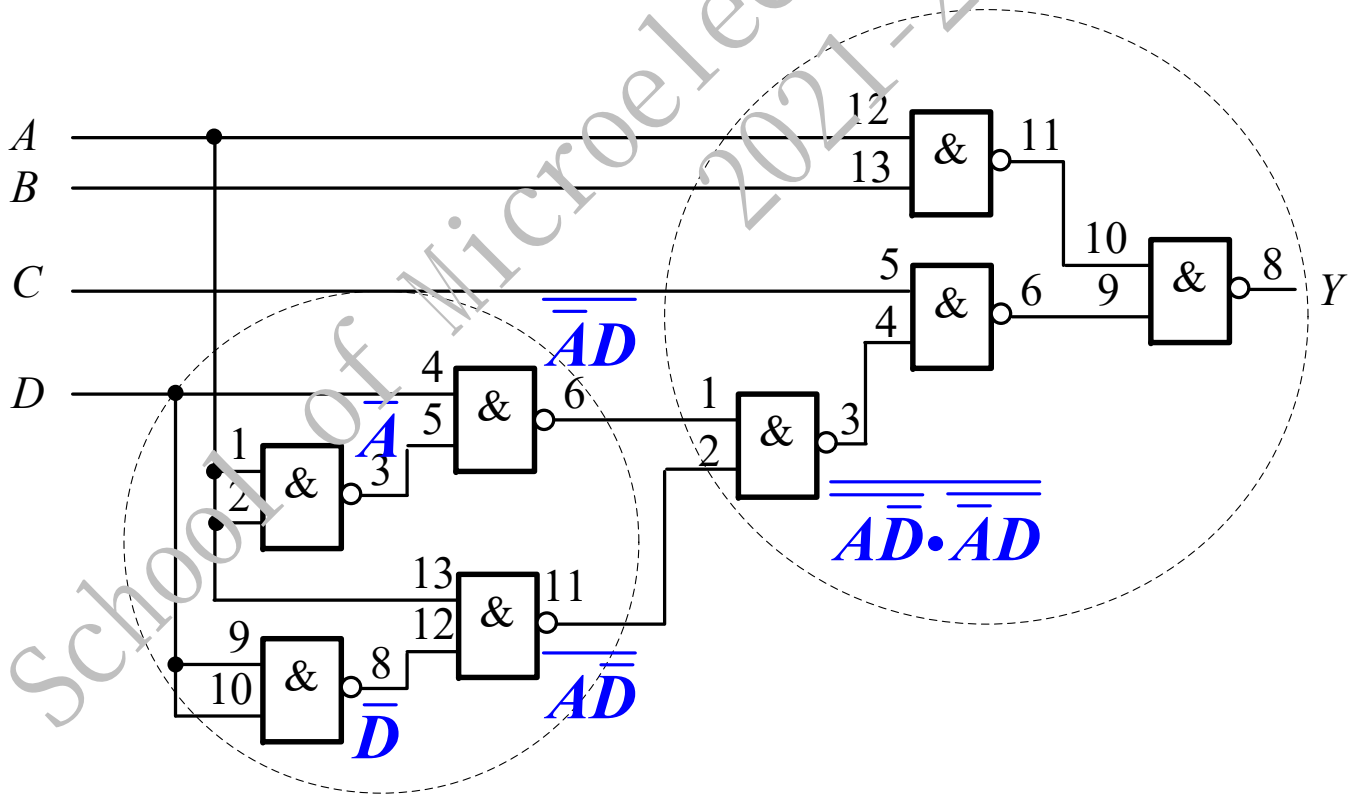


$$Y = AB + \overline{A}CD + AC\overline{D}$$

2. 与非门实现

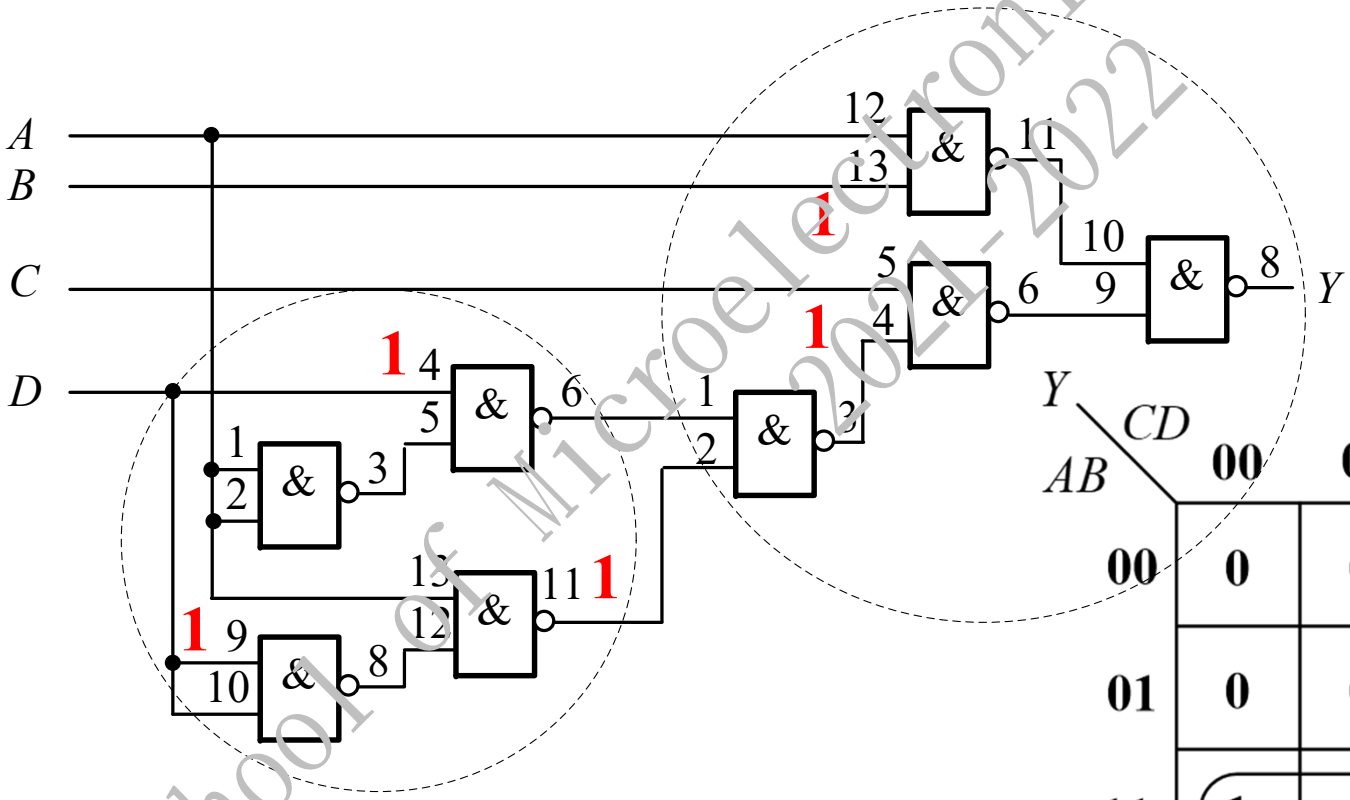
$$Y = AB + \overline{A}CD + AC\overline{D} = \overline{\overline{AB + \overline{A}CD + AC\overline{D}}} = \overline{\overline{AB}\ \overline{C(\overline{A}D + A\overline{D})}}$$

$$= \overline{\overline{AB}\ \overline{C(\overline{A}D + A\overline{D})}} = \overline{\overline{AB}\ \overline{C\overline{A}D\ A\overline{D}}}$$



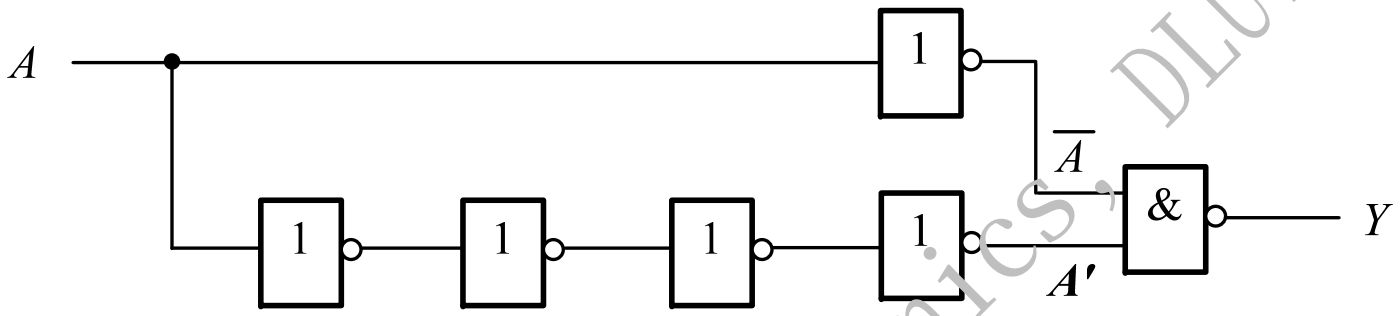
3. 什么时候产生竞争冒险?

当*BCD*=111时 $Y = AB + \overline{A}CD + AC\overline{D} = A + \overline{A}$



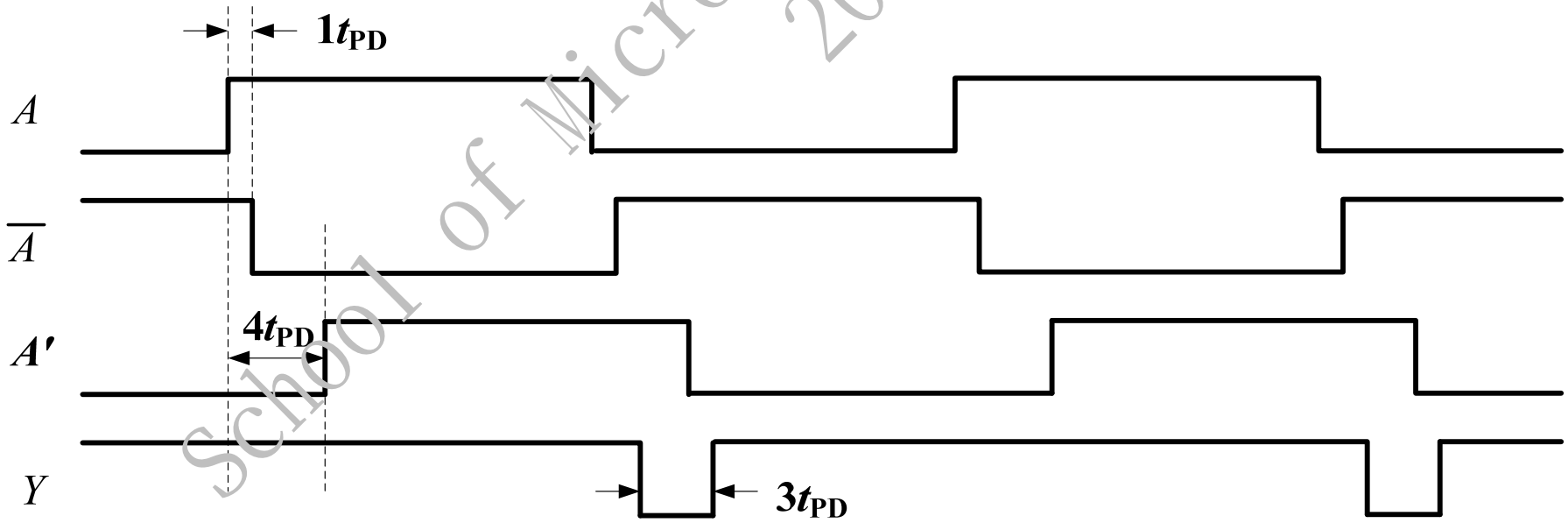
Y	CD=00	CD=01	CD=11	CD=10
AB=00	0	0	1	0
AB=01	0	0	1	0
AB=11	1	1	1	1
AB=10	0	0	0	1

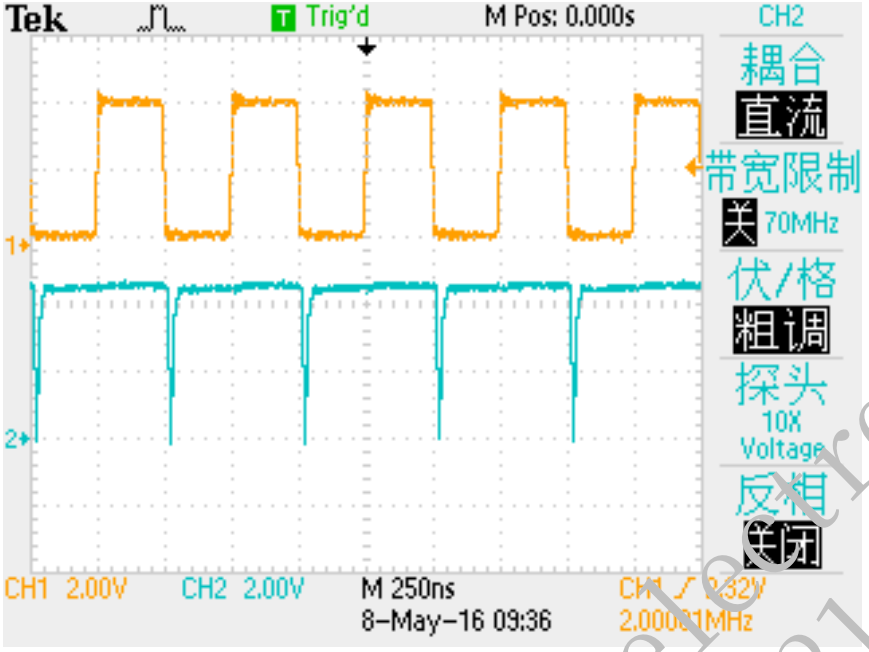
$Y = AB + \overline{A}CD + AC\overline{D}$



等效电路

假设二输入与非门延迟时间为t_{pD}，负向窄脉冲宽度为$3t_{pD}$





实测波形

4. 如何消除竞争冒险?

$$Y = AB + \overline{A}CD + AC\overline{D}$$

$$Y = AB + \overline{A}CD + AC\overline{D} + BCD$$

电路中增加相应与非门

Y (AB \ CD)	00	01	11	10
00	0	0	1	0
01	0	0	1	0
11	1	1	1	1
10	0	0	0	1

本章总结

- 掌握组合逻辑电路的基本概念
- 掌握组合逻辑电路的分析方法
- 掌握组合逻辑电路的常规设计方法
- 掌握常用译码器、多路选择器电路及集成IC
- 掌握加法器、比较器
- 掌握理解组合逻辑电路的竞争冒险以及消除方法