§ 4.4.2 BCD码转十进制译码器

BCD-to-Decimal Decoders

功能: 将 BCD 码转换成十进制码

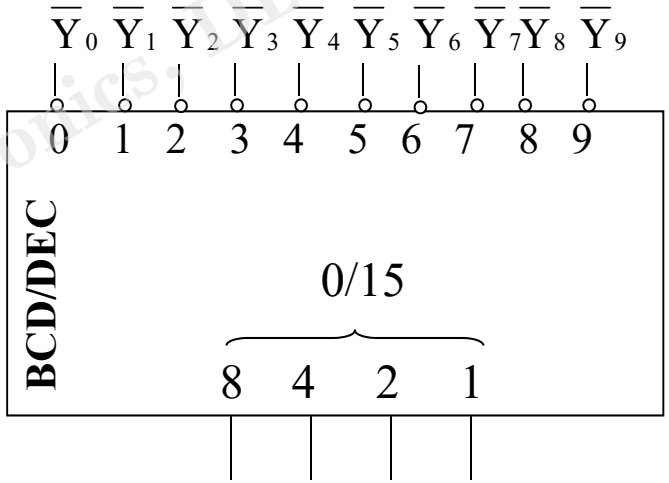
4–10线译码器 IC 7442

注意 :

输出 : 低电平有效

输入 : 有效输入 0000-1001

无效输入 1010-1111



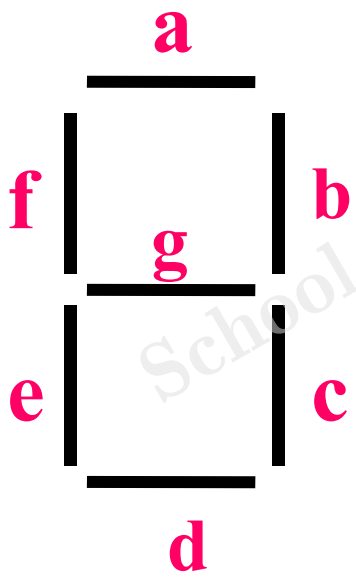
输入 BCD 码

输入数码是几，第几号输出就是唯一的低电平0

§ 4.4.3 显示译码器 (/驱动器)

Display Decoder (/Driver)

1. 7段数码管



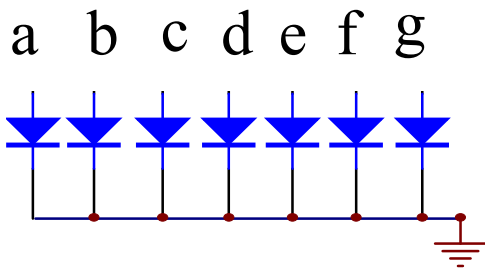
7 段数码管显示器是常见的显示器

数码管由7段发光管构成

LED：light emitting diode

LCD： liquid crystal display

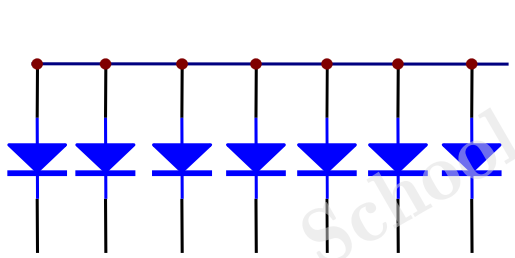
连接方式不同分成共阴极和共阳极两种



BS201A

共阴极

二极管 ⟶ 逻辑高 ⟶ 亮



BS201B

共阳极

二极管 ⟶ 逻辑低 ⟶ 亮

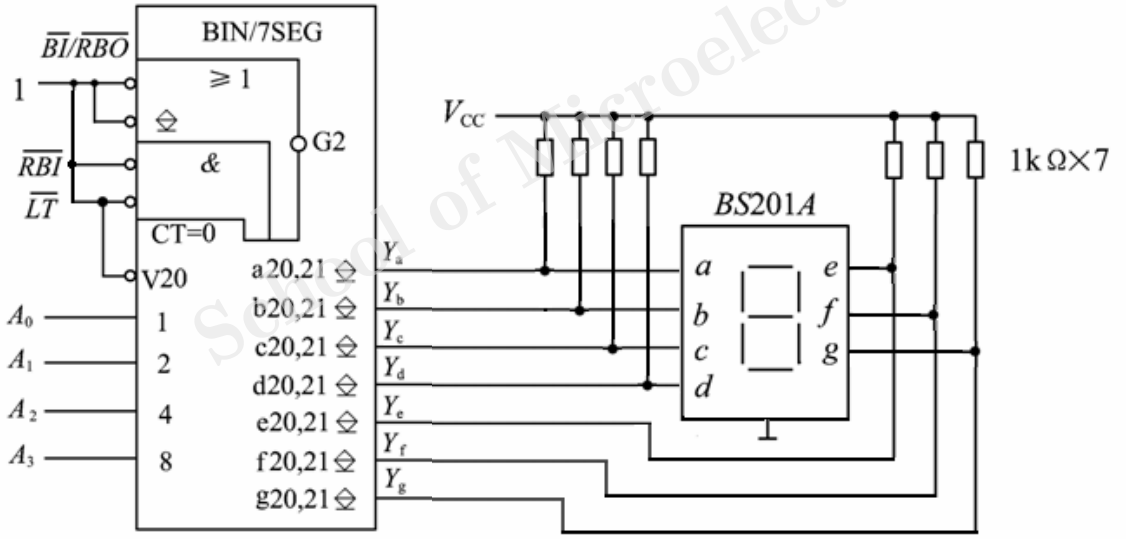
2. 显示译码器

要显示0 - 9 十个数字，需要用译码器来驱动

显示译码器 / 驱动器 7448

输入 4 线　　4 位二进制数 / 8421 BCD 码

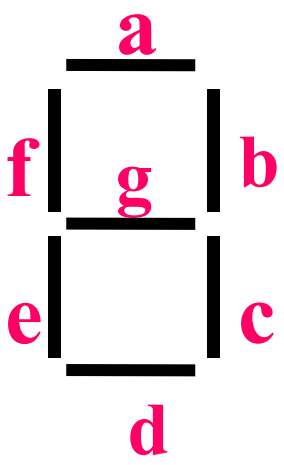
输出 7 线　⟹　驱动 7-段数码管



输出高有效，
驱动共阴极管

不一定只有一个输出端高（或低）有效

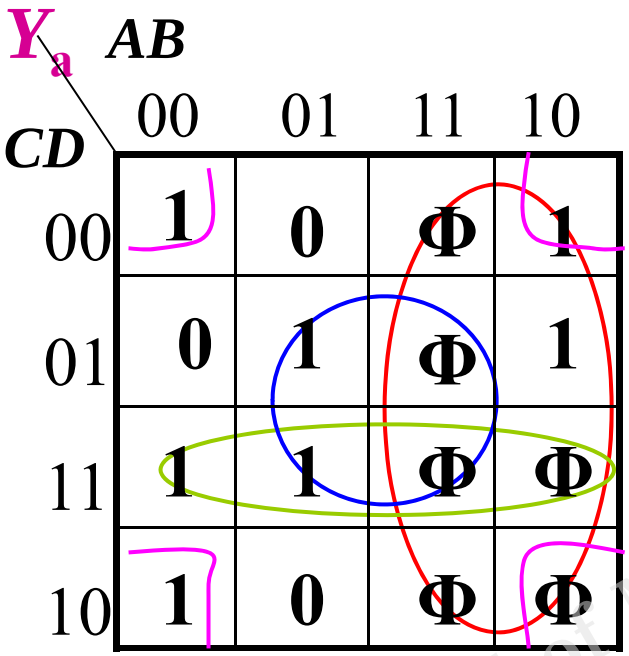
显示译码器内部电路设计



A B C D	a b c d e f g
0 0 0 0	1 1 1 1 1 1 0
0 0 0 1	0 1 1 0 0 0 0
0 0 1 0	1 1 0 1 1 0 1
0 0 1 1	1 1 1 1 0 0 1
0 1 0 0	0 1 1 0 0 1 1
0 1 0 1	1 0 1 1 0 1 1
0 1 1 0	0 0 1 1 1 1 1
0 1 1 1	1 1 1 0 0 0 0
1 0 0 0	1 1 1 1 1 1 1
1 0 0 1	1 1 1 0 0 1 1

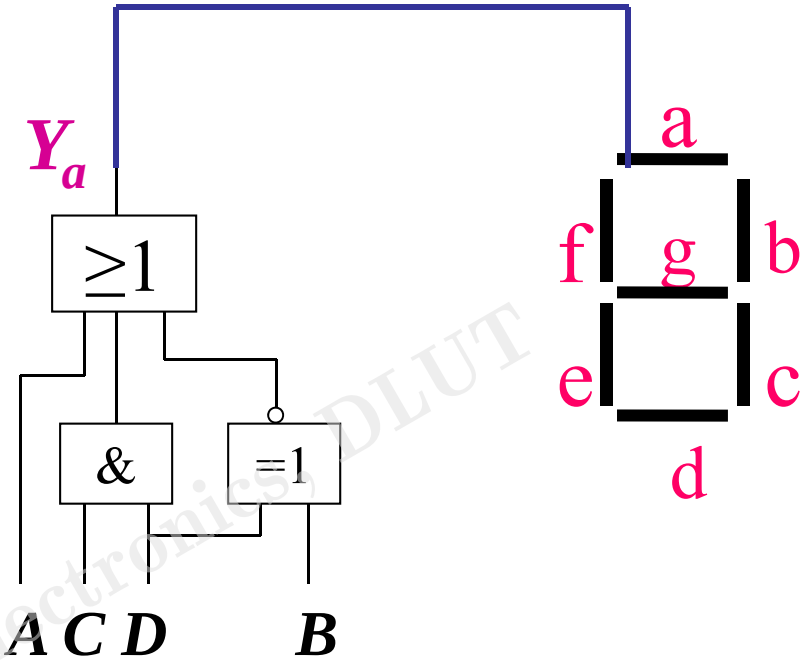
Display

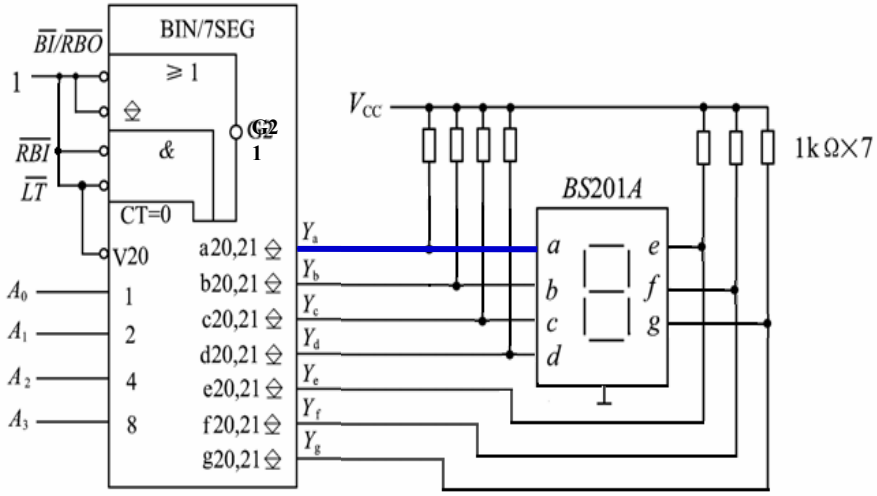
分别做7个卡诺图



$$Y_a = A + \overline{B} \cdot \overline{D} + BD + CD$$

$$= A + CD + \overline{B \oplus D}$$





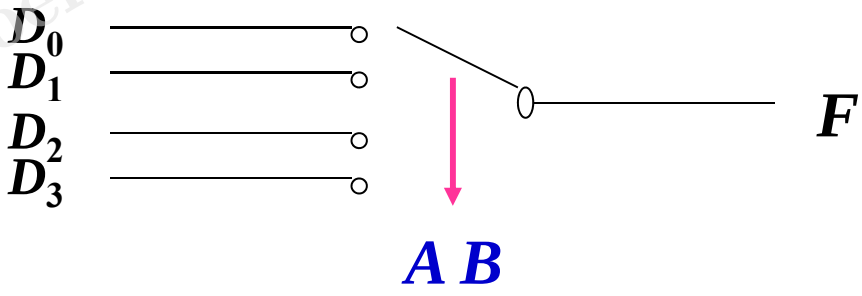
§ 4.5 多路（数据）选择器 MUX

Multiplexers (Data Selectors)

MUX 功能：在多路输入数据中选择一路进行输出

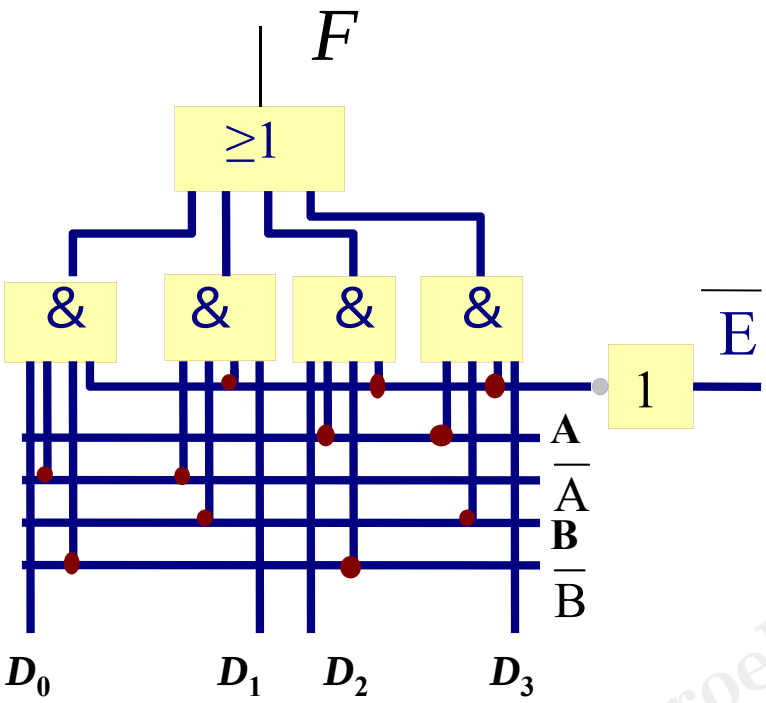
1. 4 线-1线 MUX

相当于4个数据 D_0, D_1, D_2, D_3中选一个，由开关A B控制。



A B：控制输入 (地址输入)

n 位地址线可以控制 2^n 个数据输入



$\overline{E}$	A B	F
0	0 0	D_0
0	0 1	D_1
0	1 0	D_2
0	1 1	D_3
1	φ φ	0

A B任取一值时，只有一个与门输出1(D)，其他为0，或之后为F

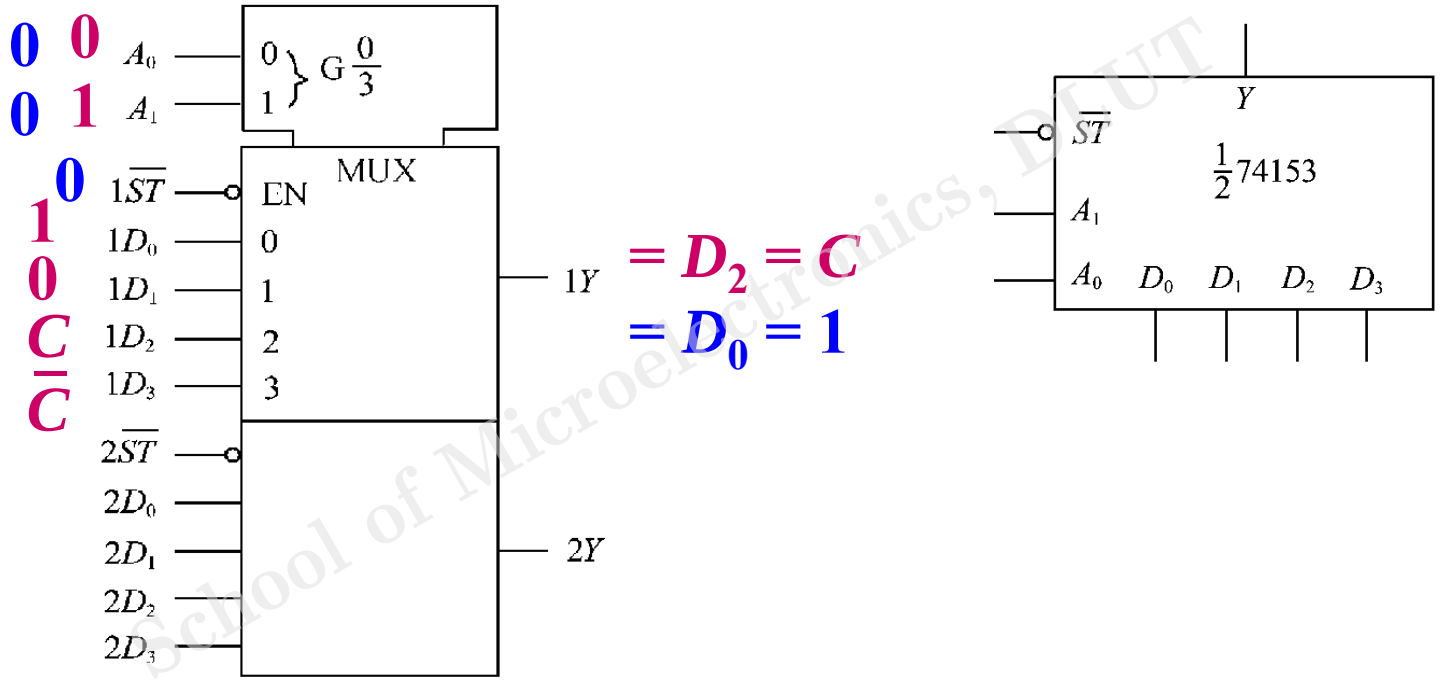
$\overline{E}$为使能端。在 $\overline{E}=0$ 的条件下，

控制码是几，就把第几号数据送到唯一的输出端

Decoder + Data lines + OR gate

MSI 4 - 1 MUX 74153 (一芯片上有 2 个 4 - 1 MUX)

符号

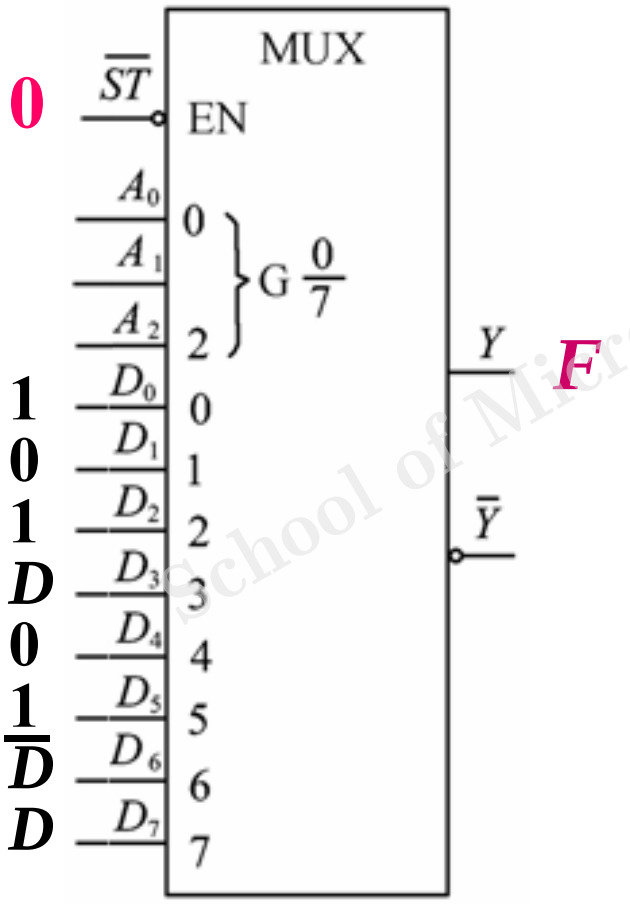


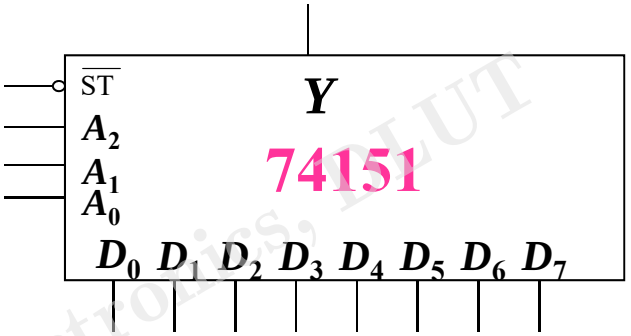
$\overline{ST}$ Select Transform 选通端，低电平有效

$A_1 A_0$：地址线 (控制输入)

2. 8线-1线 MUX　　74151 (MSI)

3 位地址线: $A_2A_1A_0$；　8 条数据线: D_0 - D_7





地址输入

A_2	A_1	A_0	F
1	0	0	0
0	1	1	D
0	0	0	1
1	1	0	$\overline{D}$

3. MUX实现逻辑函数

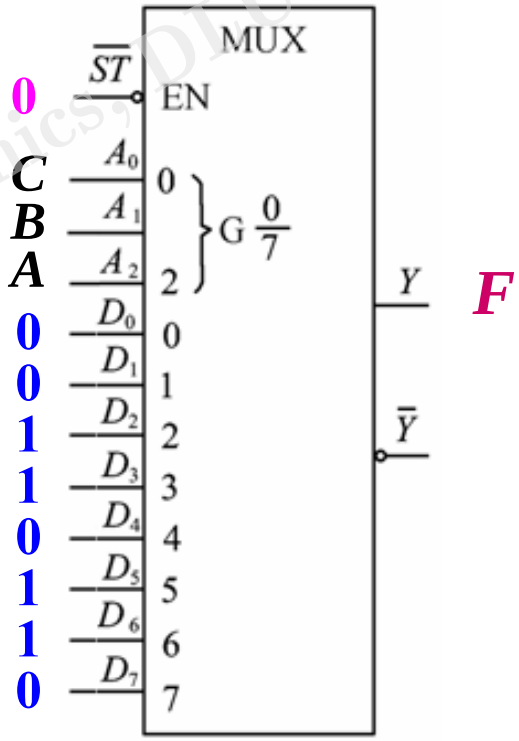
例 1: 用MUX 实现函数

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

解: 3 变量

选择 74151 (8-1 MUX)

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



一个 MUX 只能实现一个逻辑函数

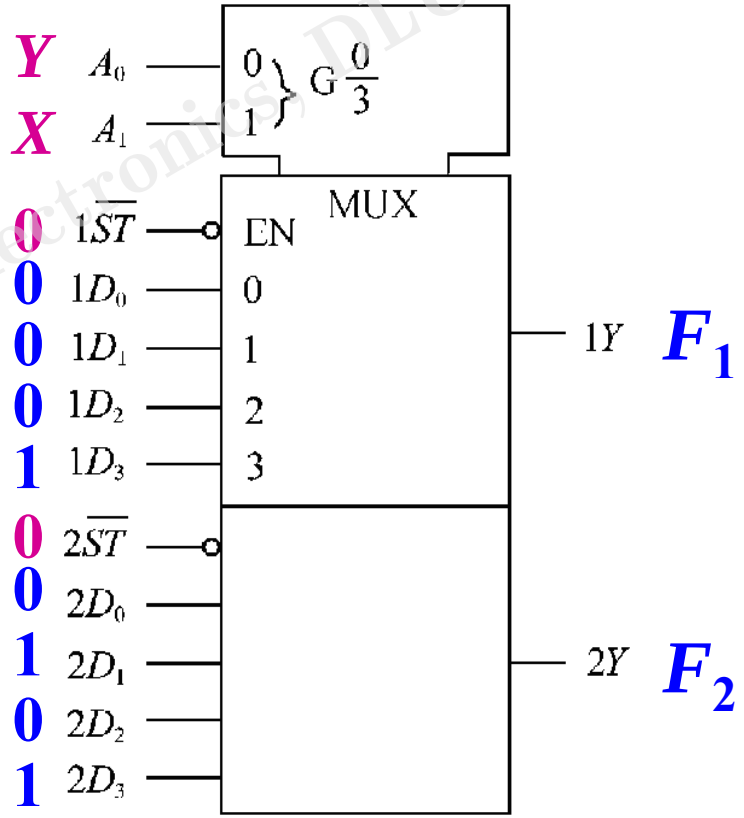
例 2： 用双4选1 MUX 74153 实现下列函数

$$F_1(X,Y) = X(\overline{X}+Y) = XY = m_3$$

$$F_2(X,Y) = \prod(0,2)$$

解:

标准形式



例 3: 用一片 74151 实现下列函数

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

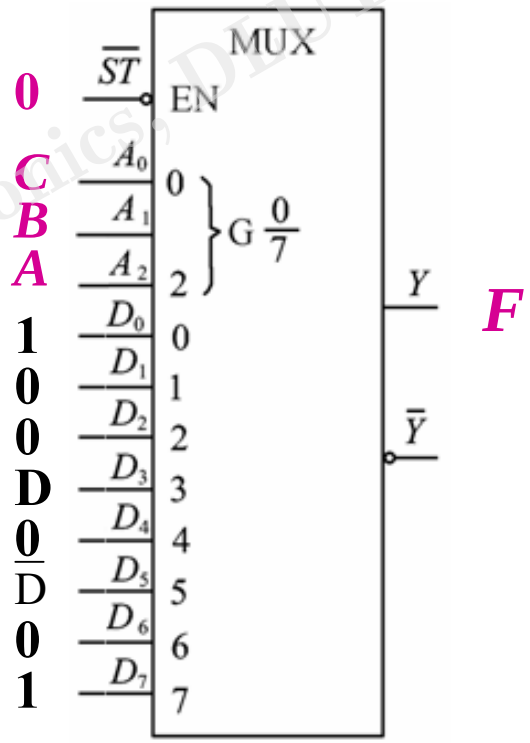
解:

74151　3变量

D **→VEM**

F ***AB*** / ***C***	00	01	11	10
0	**1**	**0**	**0**	**0**
1	**0**	**D**	**D+** $\overline{\text{D}}$	$\overline{\text{D}}$

1



例 4. 用一片4-1 MUX实现

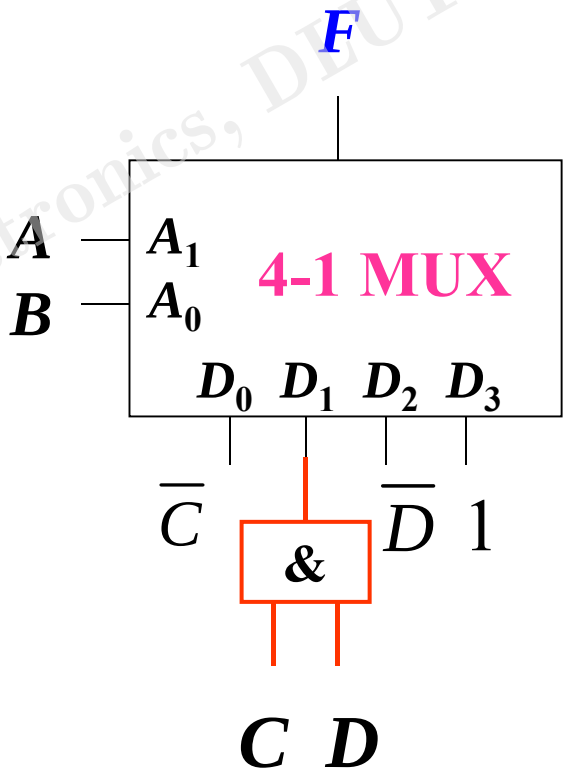
$$F(A,B,C,D)=\overline{A}BCD+AB+\overline{A}\cdot\overline{B}\cdot\overline{C}+A\overline{B}\cdot\overline{D}$$

解：

4-1 MUX　2 变量

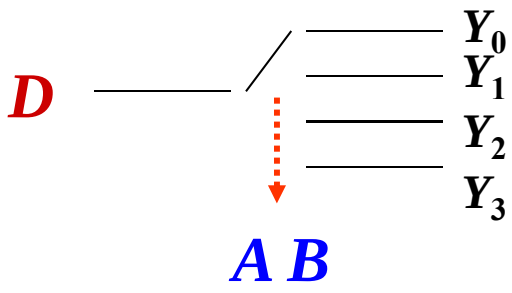
2 变量 *C*、*D* →VEM

F	*A*	0	1
B			
0		$\overline{C}$	$\overline{D}$
1		***CD***	**1**



数据分配器 Demultiplexers （DEMUX)

(Data Distributors)



A	B	Y_0	Y_1	Y_2	Y_3
0	0	D	0	0	0
0	1	0	D	0	0
1	0	0	0	D	0
1	1	0	0	0	D

控制数码是几，就把输入数据送到第几路输出端

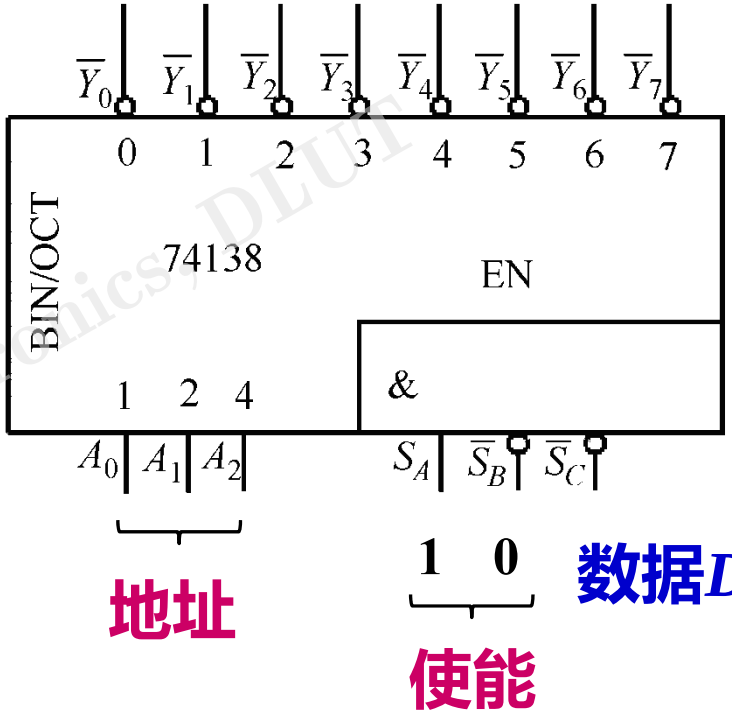
1-4 DEMUX　1-8 DEMUX　1-16 DEMUX

Decoder +Data　常用译码器实现数据分配

例： 将3-8译码器74138改装成1-8DEMUX

一个使能端$\overline{S_c}$作为数据输入

$A_2\ A_1\ A_0$作为地址输入



如$A_2\ A_1\ A_0$=110, $\overline{Y_6}$为输出端

$\overline{S_c} = D = 0$, 译码器工作, $\overline{Y_6} = 0$

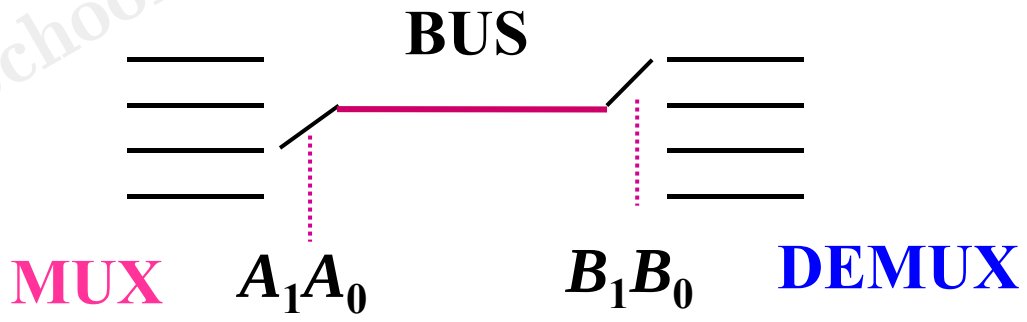
$\overline{S_c} = D = 1$, 译码器被锁住, $\overline{Y_6} = 1$

MUX 应用

1）实现逻辑函数

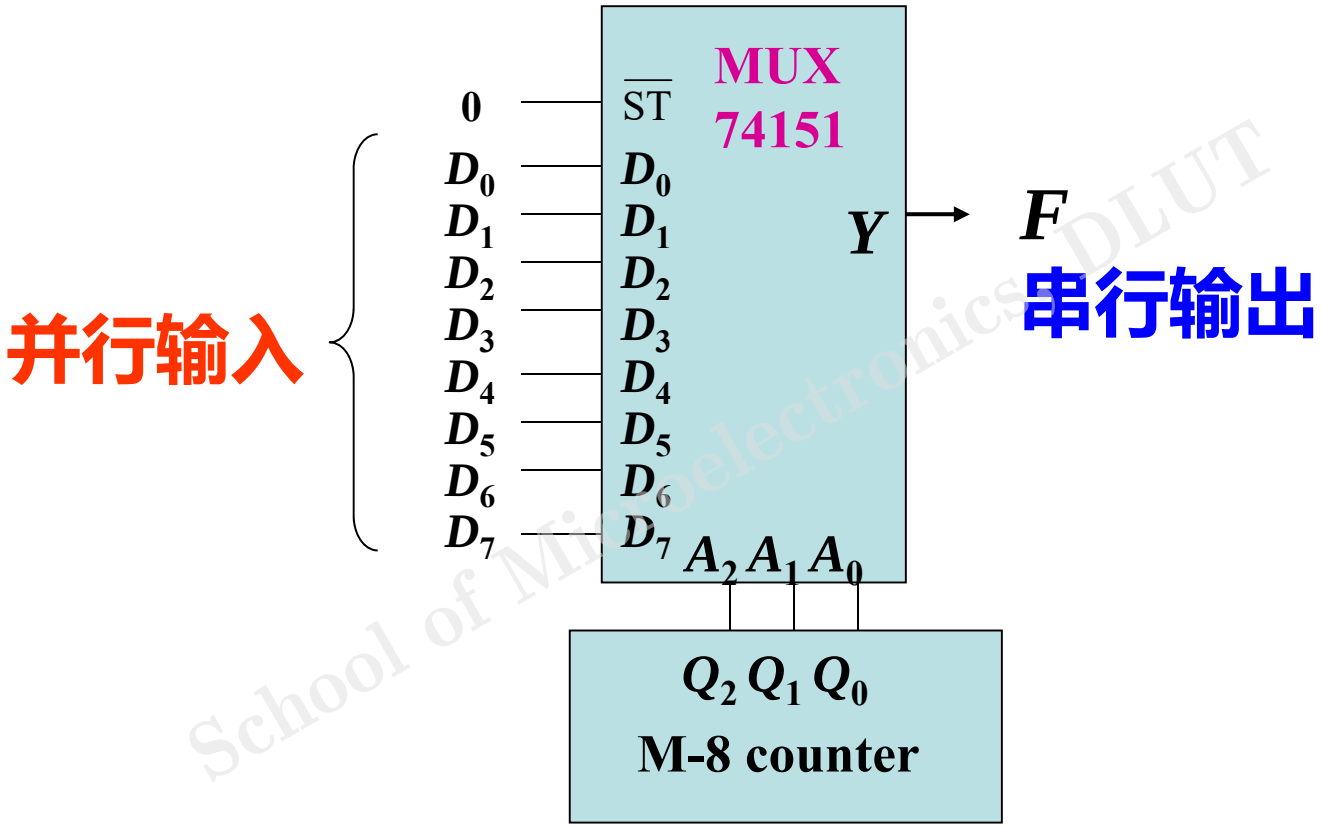
2）多路数字开关

实现路由选择，将 MUX 和 DEMUX 结合使用，实现时分多路数据通信。



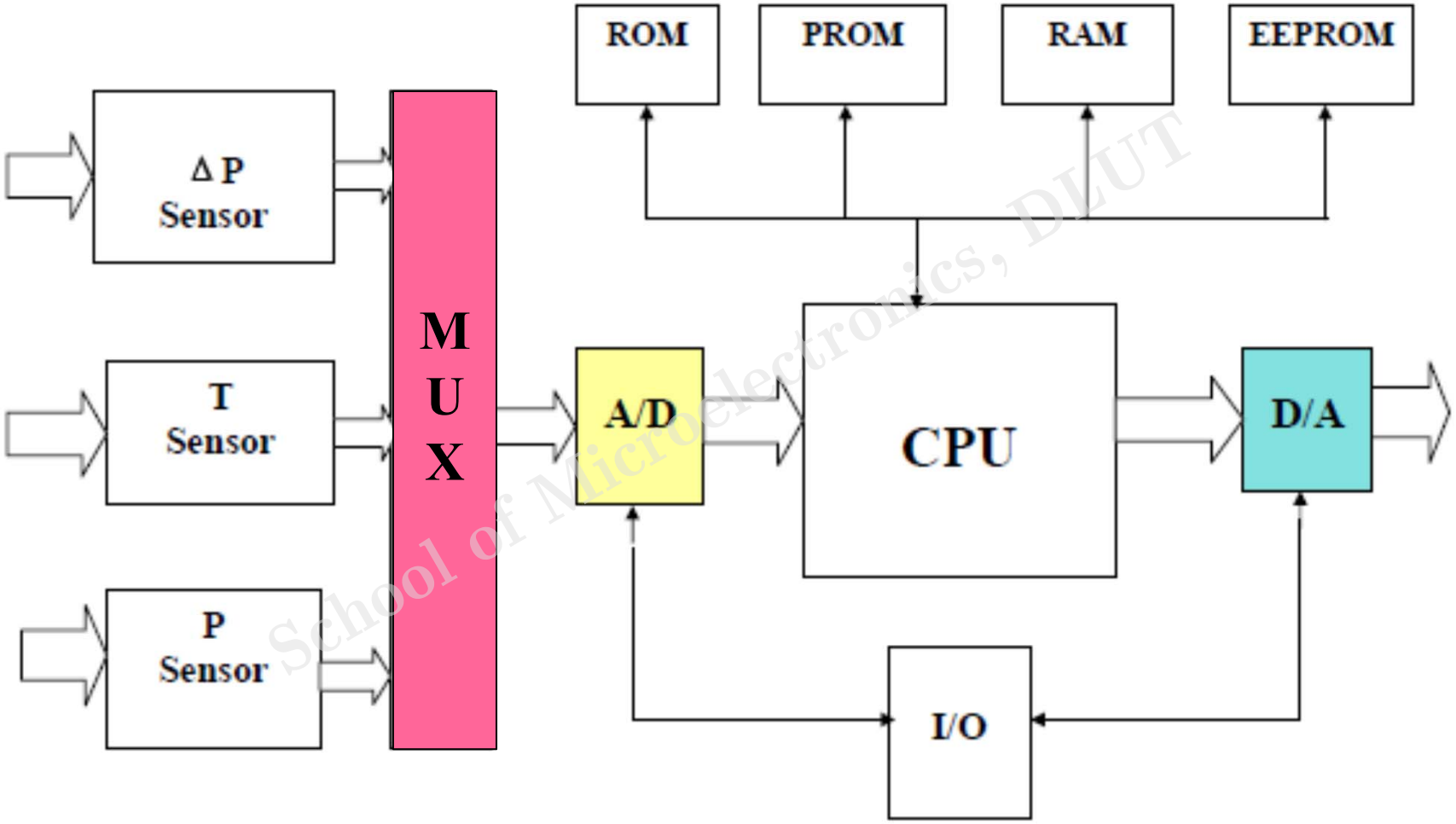
3）数据并行/串行转换

MUX 和计数器



计数器从000 ~ 111循环，使MUX 依次选择$D_0 \sim D_7$ 输出。

智能压力传感器框图



§ 4.6 比较器 Comparators

基本功能：比较两个二进制数的大小

§ 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

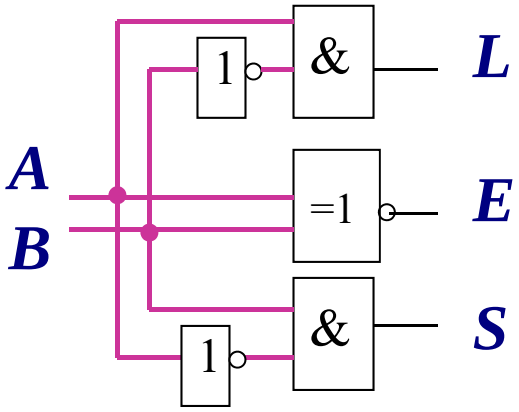
真值表

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

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

$$= A \odot B$$

Circuit



§ 4.6.2 四位比较器 Four Bits Comparator

8 输入 $\begin{cases} A: A_3 A_2 A_1 A_0 \\ B: B_3 B_2 B_1 B_0 \end{cases}$ 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 7485：

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

真值表：

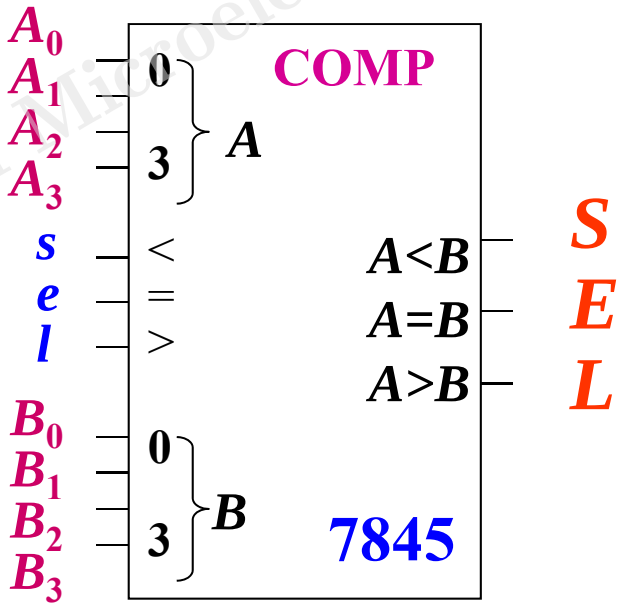
比较输入						级联输入			输出		
$A_3\ B_3$	$A_2\ B_2$	$A_1\ B_1$	$A_0\ B_0$			$l(A>B)$	$s(A<B)$	$e(A=B)$	$L(A>B)$	$S(A<B)$	$E(A=B)$
$A_3>B_3$	X	X	X			X	X	X	1	0	0
$A_3<B_3$	X	X	X			X	X	X	0	1	0
$A_3=B_3$	$A_2>B_2$	X	X			X	X	X	1	0	0
E_3	$A_2<B_2$	X	X			X	X	X	0	1	0
E_3	$A_2=B_2$	$A_1>B_1$	X			X	X	X	1	0	0
E_3	E_2	$A_1<B_1$	X			X	X	X	0	1	0
E_3	E_2	$A_1=B_1$	$A_0>B_0$			X	X	X	1	0	0
E_3	E_2	E_1	$A_0<B_0$			X	X	X	0	1	0
E_3	E_2	E_1	$A_0=B_0$			1	0	0	1	0	0
E_3	E_2	E_1	E_0			0	1	0	0	1	0
E_3	E_2	E_1	E_0			0	0	1	0	0	1

输出:

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

7845 符号:

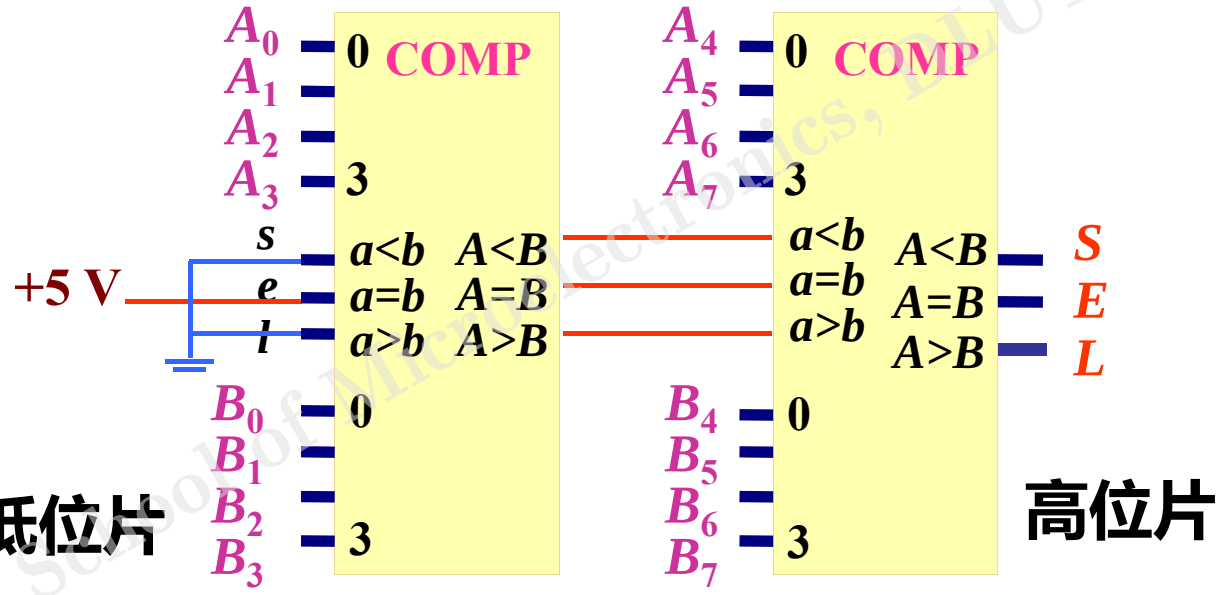
IEEE



§ 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

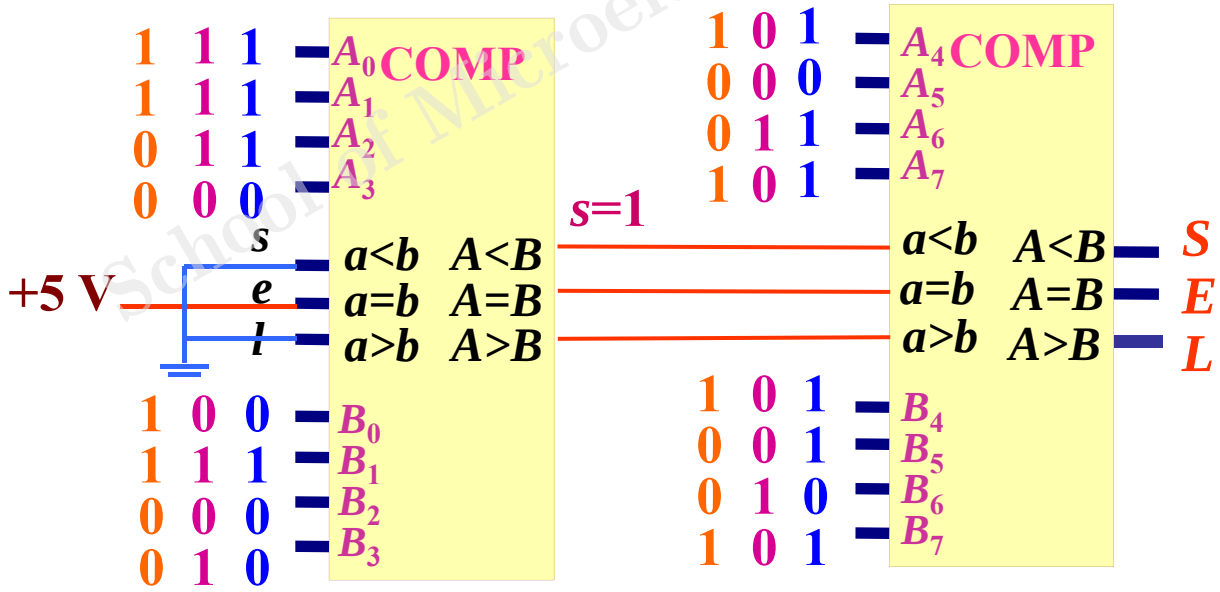


先用高位片，若高位片比出结果（$A>B$ or $A<B$), 则与级联输入状态无关；若高位片相等($A=B$)，再看级联输入，即看低位比较结果 l s e, 若低位仍相等，则$A=B$。

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

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

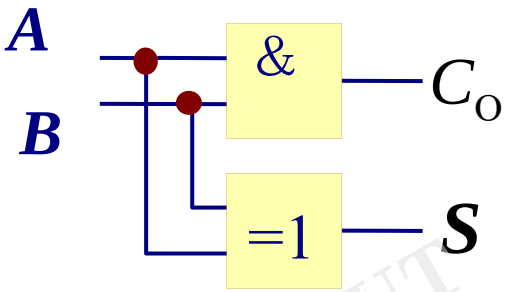
$$S = A \oplus B$$

$$C_o = AB$$

$S = A \oplus B$

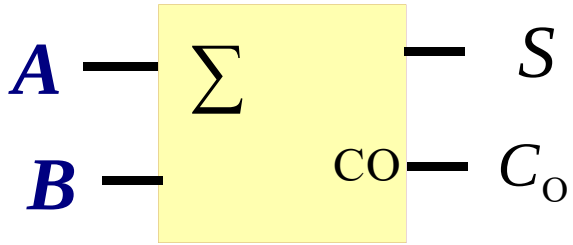
$C_O = AB$

电路



符号:

IEEE



§ 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)

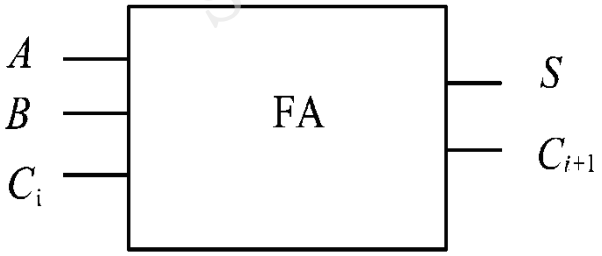
$$S = A \oplus B \oplus C_i$$ **奇数个1, 本位和为1**

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

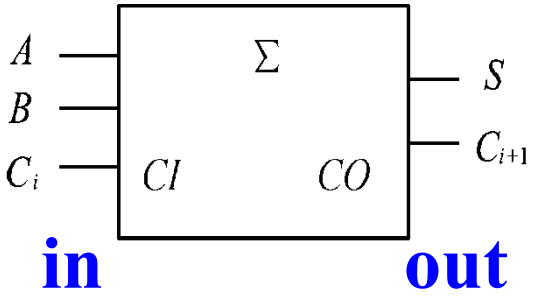
任何两个数为1, 进位

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



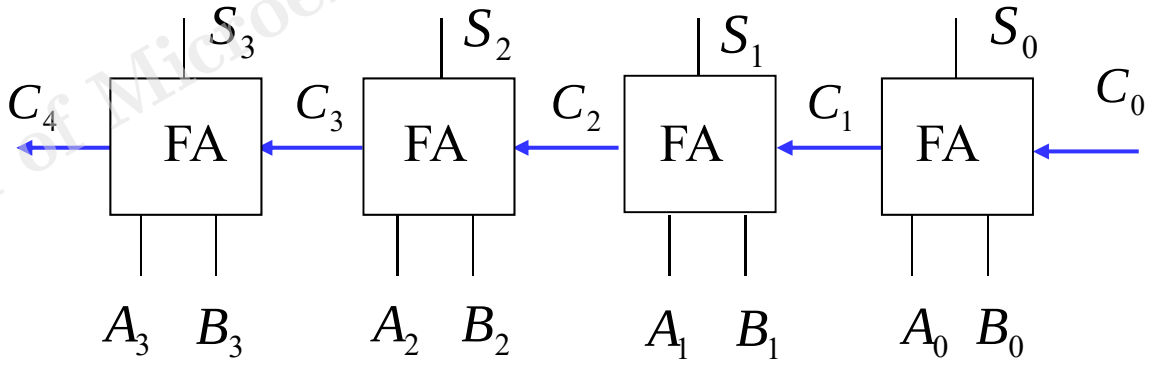
§ 4.7.3 并行加法器 Parallel Adder

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）：

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



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:

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

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

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

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)

输出写成

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

进位:

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

$$C_1 = G_0 + P_0 C_0$$

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

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

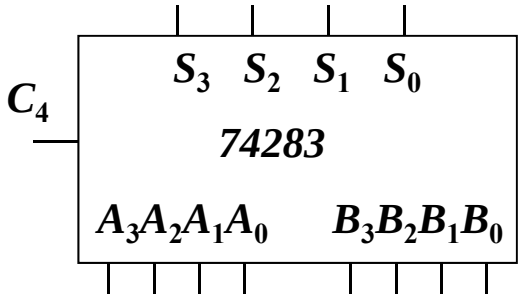
$$\begin{cases} G_i = A_i B_i \\ P_i = A_i \oplus B_i \end{cases}$$

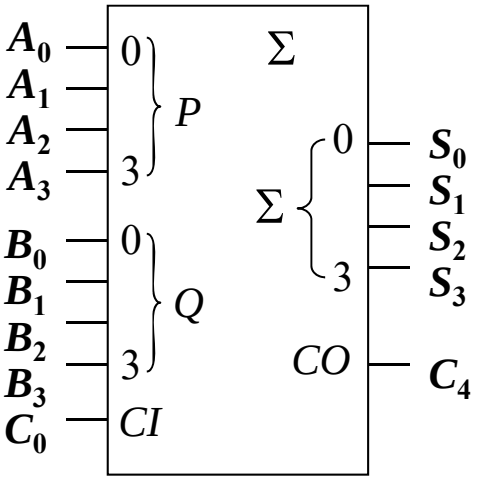
∵ C_0 =0，C_i 只与G, P 有关，即只与 A, B 有关，可以并行产生

相当于四个全加器同时计算，而不是第一个做完，得到 C_1 后，第二个再做。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

前面讨论电路是输入输出处于稳定的逻辑电平的情况。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。

由门电路的传输延时引起的问题

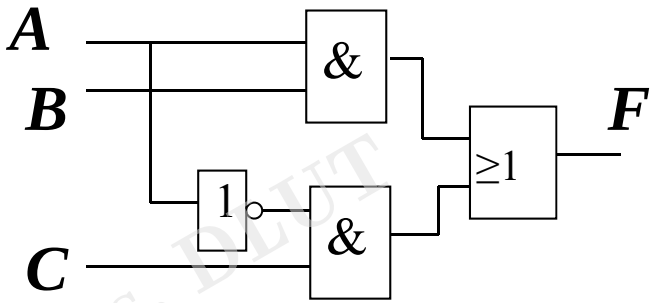
竞争：从输入到输出的途径不同，延时时间不同，到达输出端的时间不同，这种现象为竞争。

冒险：竞争结果导致逻辑电路产生错误输出，称为冒险或险象。

§ 4.8.1 竞争冒险的分类与判别

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

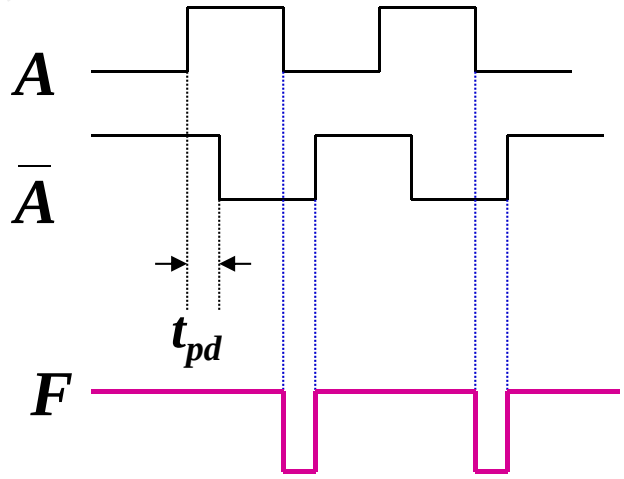
A到达终点的途径不同



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

冒险 (hazard) **毛刺 (glitch)**

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

当 $B = C = 0$

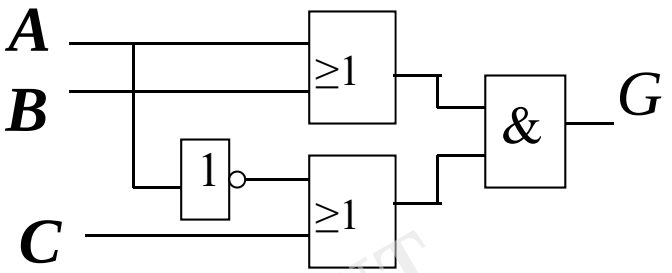
$G = A \cdot \overline{A} = 0$

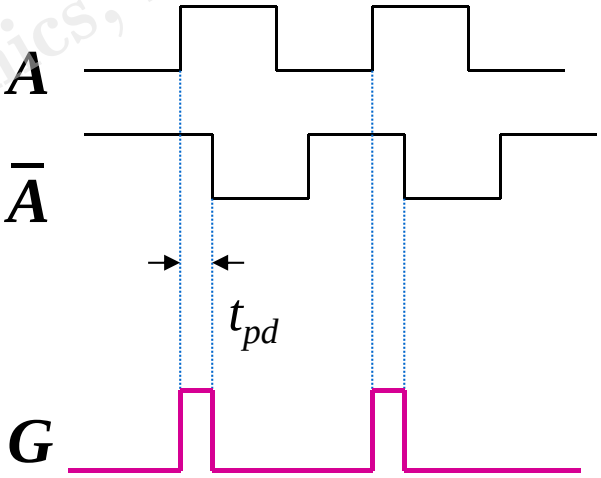
G 应该总是低电平

传播延时

G – 窄的正脉冲

冒险 (glitch)



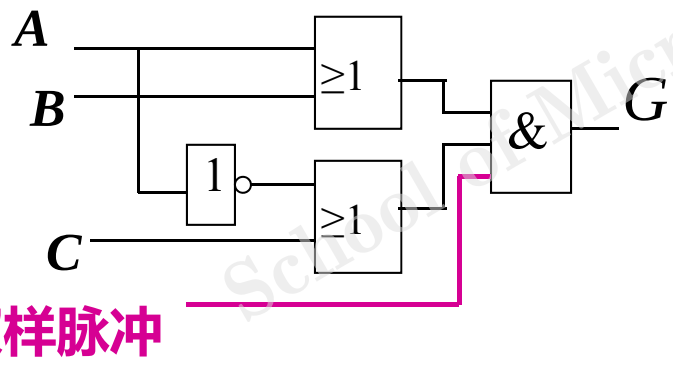


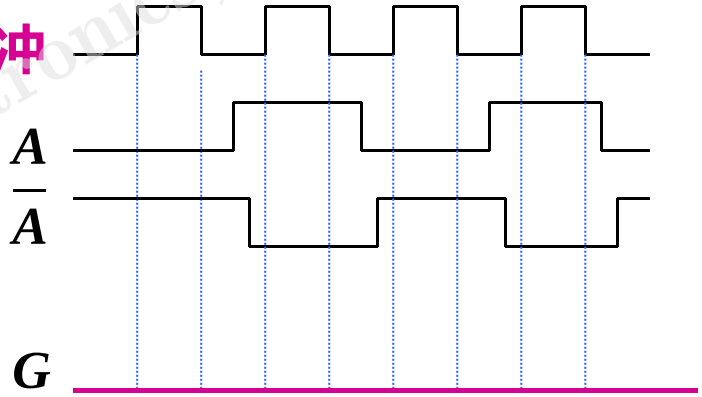
§4.8.2 竞争冒险消除方法

1. 接入滤波电容

竞争冒险引起的脉冲一般很窄(几十纳秒), 在输出端并接一个滤波电容，将其滤掉。

2. 引入取样脉冲





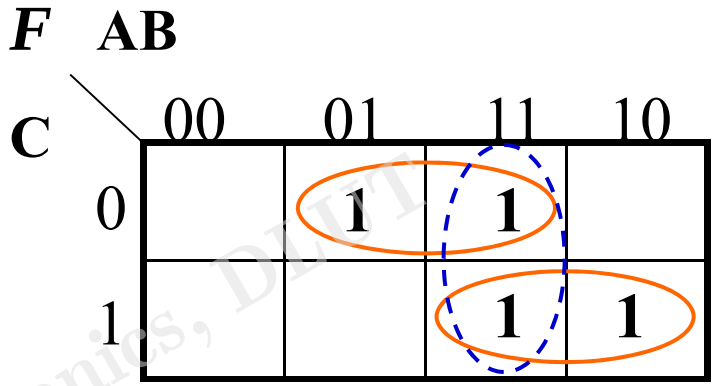
在输出端接取样脉冲，仅在输出处于稳定值时出现。取样脉冲为0期间，输出端信息无效。

3. 修改设计方案

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

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

冒险



引入冗余项，可以消除冒险

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

作业

4.2

4.5

4.6

4.14

4.18

4.21

4.26

4.28

4.30