

数字电路基础知识

- 1、逻辑门电路（何为门）
- 2、真值表
- 3、卡诺图
- 4、3 线-8 线译码器的应用
- 5、555 集成芯片的应用

一. 逻辑门电路（何为门）

在逻辑代数中，最基本的逻辑运算有与、或、非三种。每种逻辑运算代表一种函数关系，这种函数关系可用逻辑符号写成逻辑表达式来描述，也可用文字来描述，还可用表格或图形的方式来描述。

最基本的逻辑关系有三种：与逻辑关系、或逻辑关系、非逻辑关系。

实现基本逻辑运算和常用复合逻辑运算的单元电路称为**逻辑门电路**。例如：实现“与”运算的电路称为**与逻辑门**，简称**与门**；实现“与非”运算的电路称为**与非门**。逻辑门电路是设计数字系统的最小单元。

1.1.1 与门

“与”运算是一种二元运算，它定义了两个变量 A 和 B 的一种函数关系。用语句来描述它，这就是：当且仅当变量 A 和 B 都为 1 时，函数 F 为 1；或者可用另一种方式来描述它，这就是：只要变量 A 或 B 中有一个为 0，则函数 F 为 0。“与”运算又称为**逻辑乘**运算，也叫**逻辑积**运算。

“与”运算的逻辑表达式为：

$$F = A \cdot B$$

式中，乘号“ \cdot ”表示与运算，在不至于引起混淆的前提下，乘号“ \cdot ”经常被省略。该式可读作： F 等于 A 乘 B ，也可读作： F 等于 A 与 B 。

表 2-1b “与”运算真值表

A	B	$F = A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	1

由“与”运算关系的真值表可知“与”逻辑的运算规律为：

$$\begin{aligned} 0 \cdot 0 &= 0 \\ 0 \cdot 1 &= 1 \cdot 0 = 0 \\ 1 \cdot 1 &= 1 \end{aligned}$$

简单地记为：有 0 出 0，全 1 出 1。

由此可推出其一般形式为：

$$A \cdot 0 = 0$$

$$A \cdot 1 = A$$

$$A \cdot A = A$$

实现“与”逻辑运算功能的电路称为“与门”。每个与门有两个或两个以上的输入端和一个输出端，图 2-2 是两输入端与门的逻辑符号。在实际应用中，制造工艺限制了与门电路的输入变量数目，所以实际与门电路的输入个数是有限的。其它门电路中同样如此。

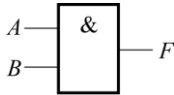


图 2-2 与门的逻辑符号

1.1.2 或门

“或”运算是另一种二元运算，它定义了变量 A 、 B 与函数 F 的另一种关系。用语句来描述它，这就是：只要变量 A 和 B 中任何一个为 1，则函数 F 为 1；或者说：当且仅当变量 A 和 B 均为 0 时，函数 F 才为 0。“或”运算又称为逻辑加，也叫逻辑和。其运算符号为“+”。

“或”运算的逻辑表达式为：

$$F = A + B$$

式中，加号“+”表示“或”运算。该式可读作： F 等于 A 加 B ，也可读作： F 等于 A 或 B 。

表 2-2b “或”运算真值表

A	B	$F = A + B$
0	0	0
0	1	1
1	0	1
1	1	1

由“或”运算关系的真值表可知“或”逻辑的运算规律为：

$$0 + 0 = 0$$

$$0 + 1 = 1 + 0 = 1$$

$$1 + 1 = 1$$

简单地记为：有 1 出 1，全 0 出 0。

由此可推出其一般形式为：

$$A + 0 = A$$

$$A + 1 = 1$$

$$A + A = A$$

实现“或”逻辑运算功能的电路称为“或门”。每个或门有两个或两个以上的输入端和一个输出端，图 2-7 是两输入端或门的逻辑符号。

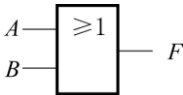


图 2-7 或门的逻辑符号

1.1.3 非门

逻辑“非”运算是一元运算，它定义了一个变量（记为 A）的函数关系。用语句来描述之，这就是：当 $A=1$ 时，则函数 $F=0$ ；反之，当 $A=0$ 时，则函数 $F=1$ 。非运算亦称为“反”运算，也叫逻辑否定。

“非”运算的逻辑表达式为：

$$F = \overline{A}$$

式中，字母上方的横线“ $\overline{}$ ”表示“非”运算。该式可读作： F 等于 A 非，或 F 等于 A 反。

表 2-3b “非”运算真值表

A	$F = \overline{A}$
0	1
1	0

由“非”运算关系的真值表可知“非”逻辑的运算规律为：

$$\overline{0} = 1$$

$$\overline{1} = 0$$

简单地记为：有 0 出 1，有 1 出 0。

由此可推出其一般形式为：

$$\overline{\overline{A}} = A$$

$$A + \overline{A} = 1$$

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

实现“非”逻辑运算功能的电路称为“非门”。非门也叫反相器。每个非门有一个输入端和一个输出端。图 2-12 是非门的逻辑符号。

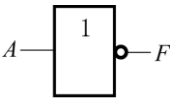


图 2-12 非门的逻辑符号

1.2.1 与非门

“与”运算后再进行“非”运算的复合运算称为“与非”运算，实现“与非”运算的逻辑电路称为与非门。一个与非门有两个或两个以上的输入端和一个输出端，两输入端与非门的逻辑符号如图 2-15 所示。

其输出与输入之间的逻辑关系表达式为：

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

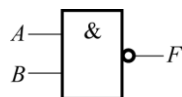


图 2-15 与非门的逻辑符号

与非门的真值表如表 2-4 所示。

表 2-4 “与非”门真值表

A	B	$F = \overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

1.2.2 或非门

“或”运算后再进行“非”运算的复合运算称为“或非”运算，实现“或非”运算的逻辑电路称为**或非门**。或非门也是一种通用逻辑门。一个或非门有两个或两个以上的输入端和一个输出端，两输入端或非门的逻辑符号如图 2-18 所示。

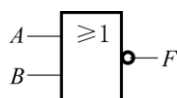


图 2-18 或非门的逻辑符号

输出与输入之间的逻辑关系表达式为：

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

或非门的真值表如表 2-5 所示。

表 2-5 “或非”门真值表

A	B	$F = \overline{A + B}$
0	0	1
0	1	0
1	0	0
1	1	0

1.2.3 异或门

在集成逻辑门中，“异或”逻辑主要为二输入变量门，对三输入或更多输入变量的逻辑，都可以由二输入门导出。所以，常见的“异或”逻辑是二输入变量的情况。

对于二输入变量的“异或”逻辑，当两个输入端取值不同时，输出为“1”；当两个输入

端取值相同时，输出端为“0”。实现“异或”逻辑运算的逻辑电路称为**异或门**。如图 2-21

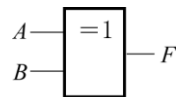


图 2-21 二输入**异或**门的逻辑符号

所示为二输入异或门的逻辑符号。

相应的逻辑表达式为：

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

其真值表如表 2-6 所示。

表 2-6 二输入“异或”门真值表

<i>A</i>	<i>B</i>	$F = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

1.2.4 同或门

“异或”运算之后再进行“非”运算，则称为“**同或**”运算。实现“同或”运算的电路称为**同或门**。同或门的逻辑符号如图 2-24 所示。

二变量同或运算的逻辑表达式为：

$$F = A \odot B = \overline{A \oplus B} = \overline{\overline{A}B} + \overline{A\overline{B}}$$

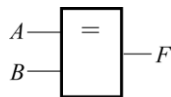


图 2-24 同或门的逻辑符号

其真值表如表 2-7 所示。

表 2-7 二变量“同或”门真值表

<i>A</i>	<i>B</i>	$F = A \odot B$
0	0	1
0	1	0
1	0	0
1	1	1

常用逻辑电路逻辑符号对照表

电路名称	国标符号	惯用符号	国外符号
与 门			
或 门			
非 门			
与非 门			
或非 门			
与或非门			
异或门			
同或门			

二. 真值表

真值表定义：表征逻辑事件输入和输出之间全部可能状态的表格。在表中通常以 1 表示真，0 表示假。真值表是在逻辑中使用的一类数学表，用来确定一个表达式是否为真或有效。

完全真值表的作法

三个步骤：

- 1、找出已给命题公式的所有变项，并竖行列出这些变项的所有真值组合；

2、根据命题公式的结构，由繁到简的依次横行列出，一次只引进一个连接词，直至列出该公式本身；

3、依据基本真值表，有变项的真值逐步计算出每个部分的真值，最后列出整个公式得真值。

如何根据真值表写出逻辑函数的表达式

第一种方法:以真值表内输出端“1”为准

第一步:从真值表内找输出端为“1”的各行,把每行的输入变量写成乘积形式;遇到“0”的输入变量上加非号。

第二步:把各乘积项相加,即得逻辑函数的表达式。

[例 1]已知某逻辑函数的真值表如表 1 表示,试写该函数的表达式并化简。

解:根据上述提示的方法有:

第一步:将输出端为“1”的各行写成乘积项,即:第四行:BC;第六行:AC;第七行:AB;第八行:ABC。

第二步:将各乘积项相加,即得逻辑函数表达式,并化简:

$$Y = \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC = BC + A\bar{B}C + AB\bar{C} = C(B + A\bar{B}) + AB\bar{C} = BC + AC + AB\bar{C} = BC + AC + AB。$$

第二种方法:以真值表内输出端“0”为准

第一步:从真值表内找输出端为“0”的各行,把每行的输入变量写成求和的形式,遇到“1”的输入变量上加非号。

第二步:把各求和项相乘,即得逻辑函数表达式。

[例2]已知某逻辑函数真值表如表2 所示,试根据此表写出函数表达式并化简。

解:

第一步:将输出端为“0”的各行写成求和形式,即:第二行:A+;第三行: +B。

第二步:将各求和项相乘即得函数表达式,并化简:Y=(A+)(+B)=AB+=A⊙B

注:在具体使用两种方法时,应观察输出端是“1”多还是“0”多,以少的为准写函数表达式(这样最简单),若输出端“1”与“0”出现的次数一样多,一般以“1”为准运算较为简单。

表 1

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

表 2

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

表 3

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

[例3]已知某函数真值表如表3 所示,试根据此表写出函数表达式并化简。

解:采用第一种方法:以输出端“1”为准时:

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

采用第二种方法:以输出端“0”为准时:

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

显然:第二种方法较第一种运算量大且烦琐一些。

三. 卡诺图

1.逻辑变量的最小项及其性质

1.1 最小项定义：

设有 n 个变量，若 m 为包含全部 n 个变量的乘积项（每个变量必须而且只能以原变量或反变量的形式出现一次）则称 m 为该组变量的最小项。

如：A、B、C 是三个逻辑变量，有以下八个乘积项

$$\overline{A}\overline{B}\overline{C} \quad \overline{A}\overline{B}C \quad \overline{A}B\overline{C} \quad \overline{A}BC \quad A\overline{B}\overline{C} \quad A\overline{B}C \quad AB\overline{C} \quad ABC$$

1.2 特点

- 1)每个最小项均含有三个因子（ n 个变量则含 n 个因子）
- (2)每个变量均为原变量或反变量的形式在乘积项中出现一次
- (3) n 个变量有 2^n 个最小项

1.3 最小项的编号

最小项常用 m_i 表示，下标 i 即为编号。在最小项中，原变量 $\rightarrow 1$ 、反变量 $\rightarrow 0$ ，所对应的十进制数即为 i 值。

以三变量为例

最小项	$\overline{A}\overline{B}\overline{C}$	$\overline{A}\overline{B}C$	$\overline{A}B\overline{C}$	$\overline{A}BC$	$A\overline{B}\overline{C}$	$A\overline{B}C$	$AB\overline{C}$	ABC
二进制数	000	001	010	011	100	101	110	111
十进制数	0	1	2	3	4	5	6	7
编号	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7

或定义为：使最小项为“1”的变量取值组合所对应的十进制数

注意

最小项的编号与变量的高、低位顺序有关

对于乘积项 ABC ，若 A 为高位 $\rightarrow m_3$

若 C 为高位 $\rightarrow m_6$

1.4 最小相的性质

A、B、C 三变量的最小项

A B C	m_0 $\overline{A}\overline{B}\overline{C}$	m_1 $\overline{A}\overline{B}C$	m_2 $\overline{A}B\overline{C}$	m_3 $\overline{A}BC$	m_4 $A\overline{B}\overline{C}$	m_5 $A\overline{B}C$	m_6 $AB\overline{C}$	m_7 ABC	$F = \sum_{i=0}^{2^n-1} m_i$
0 0 0	1	0	0	0	0	0	0	0	1
0 0 1	0	1	0	0	0	0	0	0	1
0 1 0	0	0	1	0	0	0	0	0	1
0 1 1	0	0	0	1	0	0	0	0	1
1 0 0	0	0	0	0	1	0	0	0	1
1 0 1	0	0	0	0	0	1	0	0	1
1 1 0	0	0	0	0	0	0	1	0	1
1 1 1	0	0	0	0	0	0	0	1	1

(1)对于变量的任意一组取值组合，只有一个最小项的值为 1

(2)对于变量的任意一组取值组合，任意两个最小项的积为 0

(3)对于变量的任意一组取值组合，所有最小项之和(或)为 1

2.逻辑函数最小项表达式

如 $F(A, B, C, D)$

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

$$= m_0 + m_1 + m_5 + m_8$$

$$= \sum m(0, 1, 5, 8)$$

由一般逻辑式→最小项表达式方法

1.用摩根定律去掉非号(多个变量上)直至只在一个变量上有非号为止

2.用分配律去除括号，直至得到一个与或表达式

3.配项得到最小项表达式

例 1

求函数 $F(A, B, C) = \overline{A+B+ABC}$ 的最小项表达式

$$\text{解: } F(A, B, C) = \overline{A+B+ABC}$$

$$= \overline{A} \cdot \overline{B} + \overline{A} \overline{B} C$$

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

$$= \overline{A}BC + \overline{A}B\overline{C} + \overline{A}\overline{B}C$$

$$= m_3 + m_2 + m_1$$

$$= \sum m(1, 2, 3)$$

例2

$$\begin{aligned} L(ABC) &= \overline{(\overline{AB} + \overline{AB} + \overline{C})\overline{AB}} \\ &= \overline{\overline{AB} + \overline{AB} + \overline{C} + AB} \\ &= \overline{AB + \overline{AB} \cdot C + AB} \\ &= (\overline{AB} + \overline{AB}) \cdot \overline{C} + \overline{AB} \\ &= \overline{ABC} + \overline{ABC} + \overline{AB}(C + \overline{C}) \\ &= \overline{ABC} + \overline{ABC} + \overline{ABC} + \overline{ABC} \\ &= m_3 + m_5 + m_7 + m_6 \\ &= \sum m(3, 5, 7, 6) \end{aligned}$$

结论：任一个逻辑函数都可化成为唯一的最小项表达式

最小项表达式的一种图形表示 —— 卡诺图

可利用卡诺图对逻辑函数进行化简

3.用卡诺图表示逻辑函数

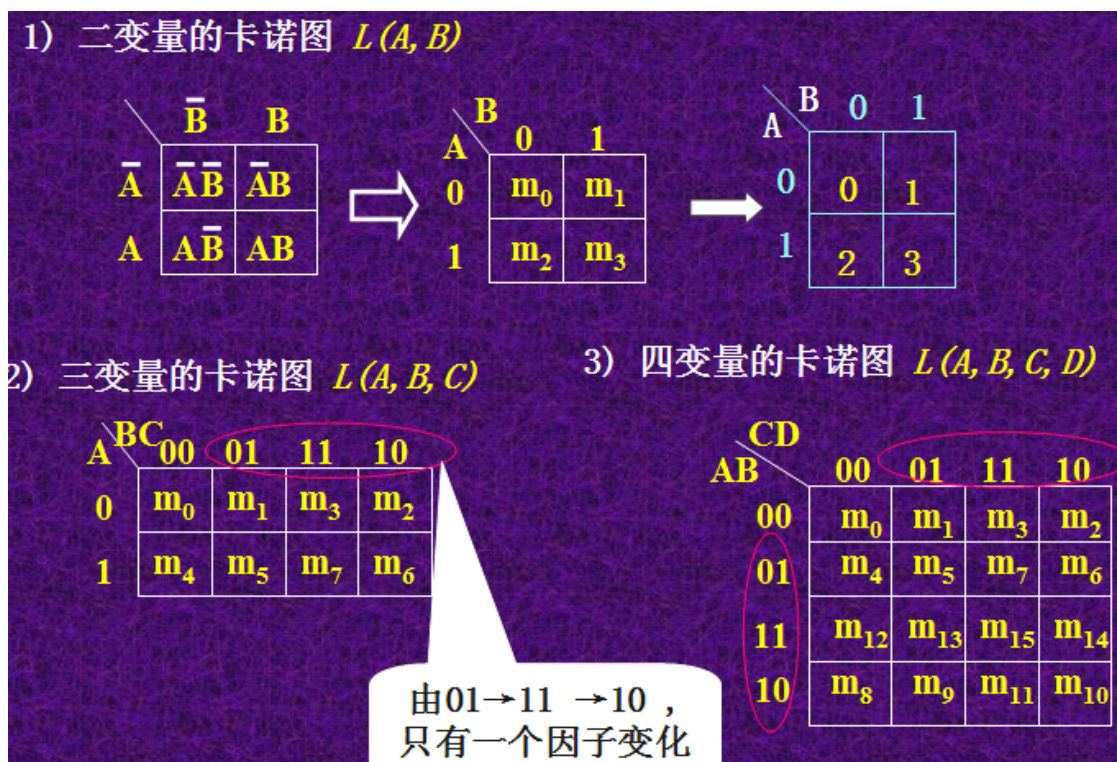
3.1 n 变量的卡诺图

将 n 个逻辑变量的 2^n 个最小项分别用一个小方块来表示，并按照逻辑上相邻的小方块在几何位置上也相邻的规则排列成的一个方格图形。

逻辑上相邻：两个最小项只有一个变量不同。例

$$\overline{A}BC\overline{D} \text{ 与 } \overline{A}BCD$$

3.2 n 变量卡诺图的具体画法：



注：变量卡诺图画法不唯一。但必须满足循环邻接的原则。

即 逻辑上邻接的最小项几何位置也邻接。

3.3 n 变量卡诺图的特点：

n 个变量函数的卡诺图有 2^n 个小方格，分别对应 2^n 个最小项；

图中行、列两组变量取值按循环码规律排列，使几何相邻的最小项之间具有逻辑相邻性。

3.4 逻辑函数的卡诺图画法

(1) 已知逻辑表达式

i) 逻辑表达式化成最小项表达式

ii) 画变量卡诺图

iii) 在最小项表达式中包含的最小项对应的小方块中填“1”；其余填入“0”

❖ 这样，任何一个逻辑函数就等于其卡诺图中

填“1”的那些最小项之和

例 1：把函数化成最小项表达式，再画卡诺图。

$$\begin{aligned} Y &= \overline{A}\overline{B}CD + \overline{A}BCD + AC \\ &= \overline{A}\overline{B}CD + \overline{A}(B + \overline{B})CD + A(B + \overline{B})C \\ &= \overline{A}\overline{B}CD + \overline{A}BCD + \overline{A}BCD + ABC(D + \overline{D}) \\ &\quad + \overline{A}BC(D + \overline{D}) \\ &= \overline{A}\overline{B}CD + \overline{A}BCD + \overline{A}BCD + ABCD + \\ &\quad \overline{A}BCD + \overline{A}BCD + \overline{A}BCD \\ &= \sum m_i (i = 3, 7, 9, 10, 11, 14, 15) \end{aligned}$$

4. 用卡诺图化简逻辑函数

CD	00	01	11	10
AB	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

1.卡诺图化简的依据：循环邻接性

1) 相邻两个最小项求和时,两项并一项并消去一个因子

如:

$$m_1 + m_9 = \overline{A}BCD + A\overline{B}CD = \overline{B}CD$$

$$m_4 + m_6 = \overline{A}BC\overline{D} + A\overline{B}C\overline{D} = \overline{B}C\overline{D}$$

2) 相邻四个最小项求和时,四项并一项并消去两个因子

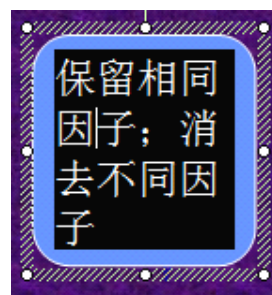
如:

$$\begin{aligned} m_0 + m_2 + m_8 + m_{10} \\ = \overline{A}B\overline{C}\overline{D} + \overline{A}BC\overline{D} + A\overline{B}\overline{C}\overline{D} + A\overline{B}C\overline{D} = \overline{C}\overline{D} \end{aligned}$$

3) 相邻八个最小项求和时,八项并一项并消去三个因子

如:

$$m_0 + m_2 + m_4 + m_6 + m_8 + m_{10} + m_{12} + m_{14} = \overline{D}$$



2.用卡诺图化简逻辑函数的方法和步骤

1) 将相邻的值为“1”的小方块画成若干个包围圈

i) 每个包围圈中必须含有 2^n 个小方块 ($n=0,1,2, \dots$)

ii) 小方块可重复被包围, 但每个包围圈中必须含有其他包围圈没有的新小方块

iii) 不能漏掉任何值为 1 的小方块

iv) 包围圈所含的小方块数目要尽可能多

v) 包围圈数目要尽可能少, 画包围圈的顺序由大→小

2) 将每个包围圈中的最小项合并成一项→乘积项 (留下相同因子, 消去不同因子)

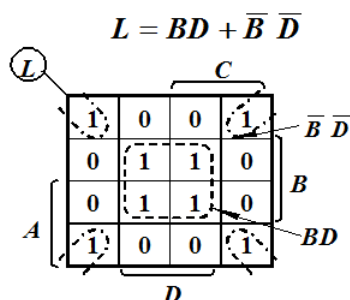
3) 对各个包围圈合并成的乘积项求逻辑和

例2.2.4 :用卡诺图法化简下列逻辑函数

$$L(A, B, C, D) = \sum m(0, 2, 5, 7, 8, 10, 13, 15)$$

解: (1) 由 L 画出卡诺图

(2) 画包围圈合并最小项, 得最简与-或表达式



例 化简 $L = \overline{A}\overline{B}\overline{C} + \overline{A}B + BC + AC$

$L = \overline{A}\overline{C} + \overline{A}B + AC$

$L = \overline{A}\overline{C} + BC + AC$

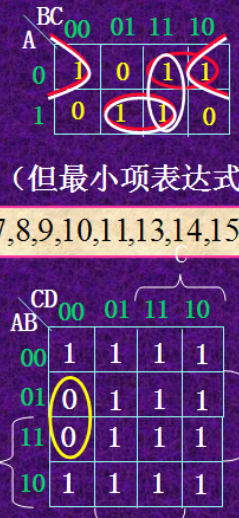
结论： 逻辑函数最简与或式不是唯一的（但最小项表达式唯一）

例2.2.6 $L(A, B, C, D) = \Sigma m(0,1,2,3,5,6,7,8,9,10,11,13,14,15)$

$\therefore \overline{L} = B\overline{C}\overline{D}$

$\therefore L = \overline{\overline{L}} = \overline{B\overline{C}\overline{D}} = \overline{B} + C + D$

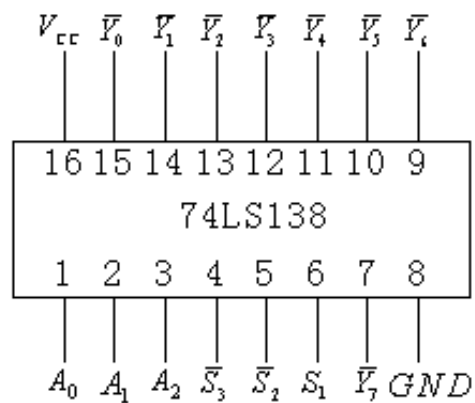
结论： 含0较少时，化包围0的小圆圈，并项得反函数。再求原函数。



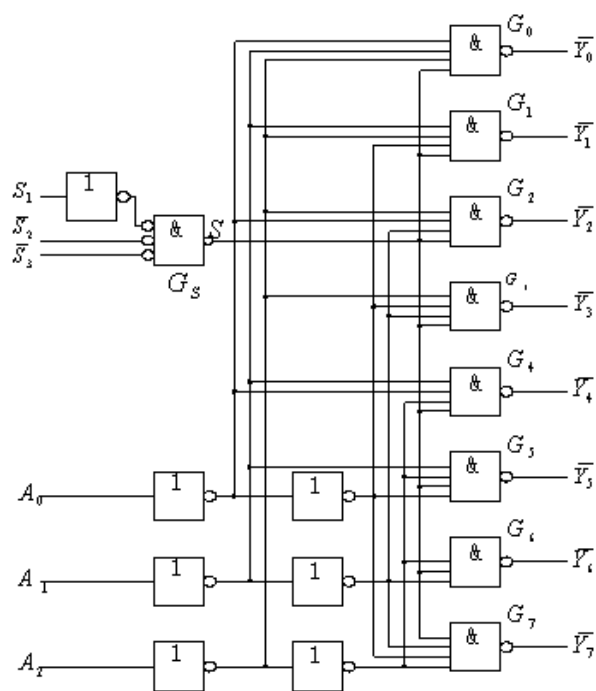
The Karnaugh map for $L(A, B, C, D)$ is a 4x4 grid with rows labeled AB (00, 01, 11, 10) and columns labeled CD (00, 01, 11, 10). The values are: Row 00: 1, 0, 1, 1; Row 01: 0, 1, 1, 1; Row 11: 0, 1, 1, 1; Row 10: 1, 1, 1, 1. The 0s are at (00,01), (01,00), (11,00), and (10,00). The 1s are at (00,00), (00,11), (01,01), (01,11), (01,10), (11,01), (11,11), (11,10), (10,01), (10,11), (10,10), and (10,00).

四. 3 线-8 线译码器的应用

逻辑原理图及功能表：



用与非门组成的 3 线-8 线译码器 74LS138



3 线-8 线译码器 74LS138 的功能表

输 入					输 出							
S_1	$\overline{S_2} + \overline{S_3}$	A_2	A_1	A_0	$\overline{Y_0}$	$\overline{Y_1}$	$\overline{Y_2}$	$\overline{Y_3}$	$\overline{Y_4}$	$\overline{Y_5}$	$\overline{Y_6}$	$\overline{Y_7}$
0	X	X	X	X	1	1	1	1	1	1	1	1
X	1	X	X	X	1	1	1	1	1	1	1	1
1	0	0	0	0	0	1	1	1	1	1	1	1
1	0	0	0	1	1	0	1	1	1	1	1	1
1	0	0	1	0	1	1	0	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1	1	1
1	0	1	0	0	1	1	1	1	0	1	1	1
1	0	1	0	1	1	1	1	1	1	0	1	1
1	0	1	1	0	1	1	1	1	1	1	0	1
1	0	1	1	1	1	1	1	1	1	1	1	0

无论从逻辑图还是功能表我们都可以看到 74LS138 的八个输出引脚，任何时刻要么全为高电平 1——芯片处于不工作状态，要么只有一个为低电平 0，其余 7 个输出引脚全为高电平 1。如果出现两个输出引脚同时为 0 的情况，说明该芯片已经损坏。

当附加控制门的输入为高电平（S=1）时，可由逻辑图写出

$$\begin{cases} \bar{Y}_0 = \overline{A_2 A_1 A_0} = \bar{m}_0 \\ \bar{Y}_1 = \overline{A_2 A_1 A_0} = \bar{m}_1 \\ \bar{Y}_2 = \overline{A_2 A_1 A_0} = \bar{m}_2 \\ \bar{Y}_3 = \overline{A_2 A_1 A_0} = \bar{m}_3 \\ \bar{Y}_4 = \overline{A_2 A_1 A_0} = \bar{m}_4 \\ \bar{Y}_5 = \overline{A_2 A_1 A_0} = \bar{m}_5 \\ \bar{Y}_6 = \overline{A_2 A_1 A_0} = \bar{m}_6 \\ \bar{Y}_7 = \overline{A_2 A_1 A_0} = \bar{m}_7 \end{cases} \quad (3.3.7)$$

由上式可以看出，同时又是这三个变量的全部最小项的译码输出，所以也把这种译码器叫做最小项译码器。

74LS138 有三个附加的控制端，当 $s_1=1, s_2'+s_3'=0$ 时，GS 输出为高电平（ $S=1$ ），译码器处于工作状态。否则，译码器被禁止，所有的输出端被封锁在高电平，如表 3.3.5 所示。这三个控制端也叫做“片选”输入端，利用片选的作用可以将多篇连接起来以扩展译码器的功能。

带控制输入端的译码器又是一个完整的数据分配器。在图 3.3.8 电路中如果把 S_1 作为“数据”输入端（同时令 $S_2=S_3=0$ ），而将 $A_2A_1A_0$ 作为“地址”输入端，那么从 S_1 送来的数据只能通过 $A_2A_1A_0$ 所指定的一根输出线送出去。这就不难理解为什么把 $A_2A_1A_0$ 叫做地址输入了。例如当 $A_2A_1A_0=101$ 时，门 G_s 的输入端除了接至 G_s 输出端的一个以外全是高电平，因此 S_1 的数据以反码的形式从 Y_5 输出，而不会被送到其他任何一个输出端上。

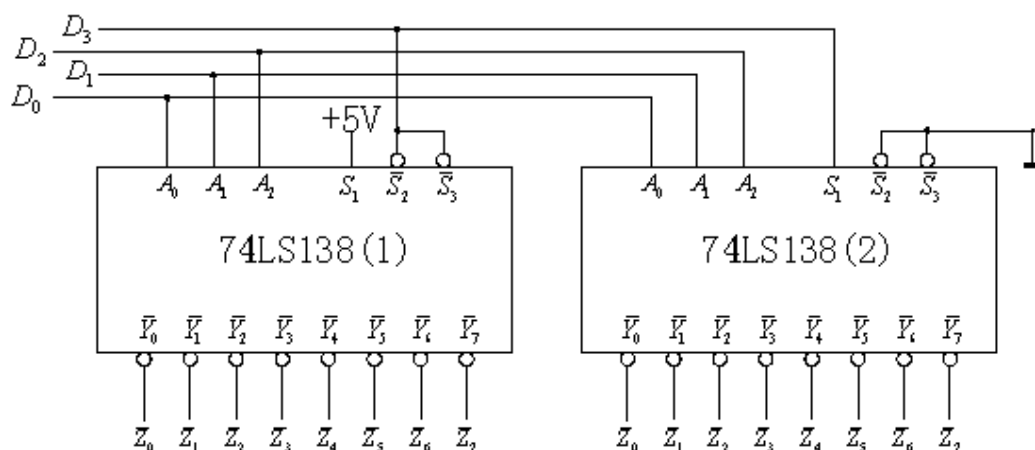
【例 3.3.2】试用两片 3 线-8 线译码器 74LS138 组成 4 线-16 线译码器，将输入的 4 位二进制代码译成 16 个独立的低电平信号。

解：由图 3.3.8 可见，74LS138 仅有 3 个地址输入端。如果想对 4 位二进制代码，只能利用一个附加控制端（其中的一个）作为第四个地址输入端。取第（2）片的 s_1 端作为它的第四个地址输入端，如图 3.3.9 所示，于是得到两片 74LS138 的输出分别为

$$\begin{cases} \bar{Z}_0 = \overline{\bar{D}_3 \bar{D}_2 \bar{D}_1 \bar{D}_0} \\ \bar{Z}_1 = \overline{\bar{D}_3 \bar{D}_2 \bar{D}_1 D_0} \\ \vdots \\ \bar{Z}_7 = \overline{\bar{D}_3 D_2 D_1 D_0} \end{cases} \quad (3.3.8)$$

$$\begin{cases} \bar{Z}_8 = \overline{D_3 \bar{D}_2 \bar{D}_1 \bar{D}_0} \\ \bar{Z}_9 = \overline{D_3 \bar{D}_2 \bar{D}_1 D_0} \\ \vdots \\ \bar{Z}_{15} = \overline{D_3 D_2 D_1 D_0} \end{cases} \quad (3.3.9)$$

图 3.3.9 用两片 74LS138 接成的 4 线—16 线译码器

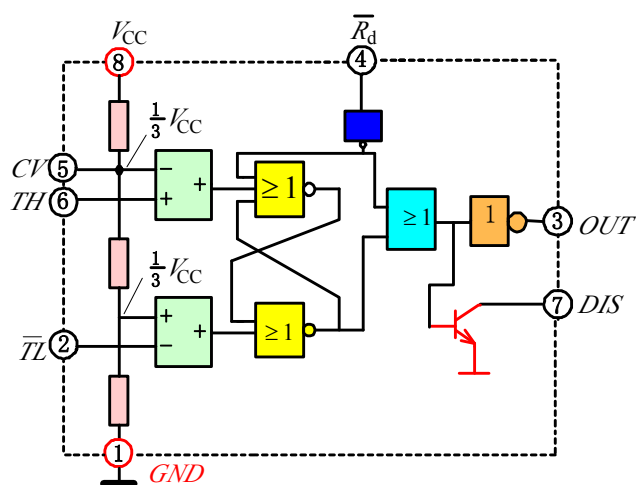


五. 555 集成芯片的应用

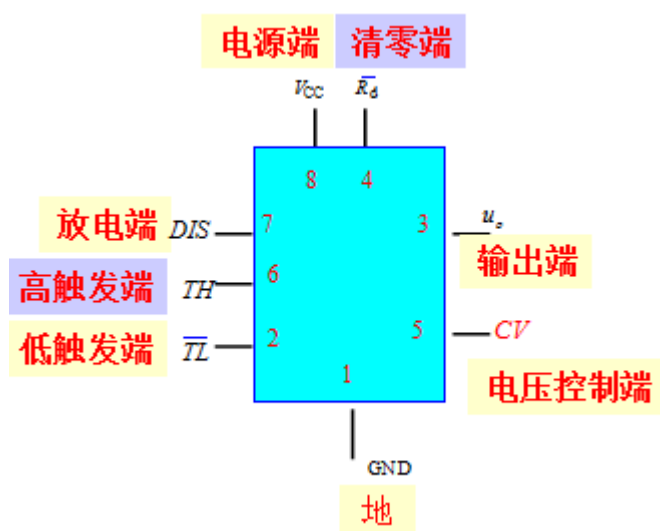
555 定时器简介：

555 定时器（时基电路）是一种用途广泛的模拟数字混合集成电路。1972 年由西格尼蒂斯公司（Signetics）研制；设计新颖、构思奇巧，备受电子专业设计人员和电子爱好者青睐；它可以构成单稳态触发器、多谐振荡器、施密特触发器和压控振荡器等多种应用电路。

555 定时器的工作原理



555 定时器电路框图



555 定时器符号图

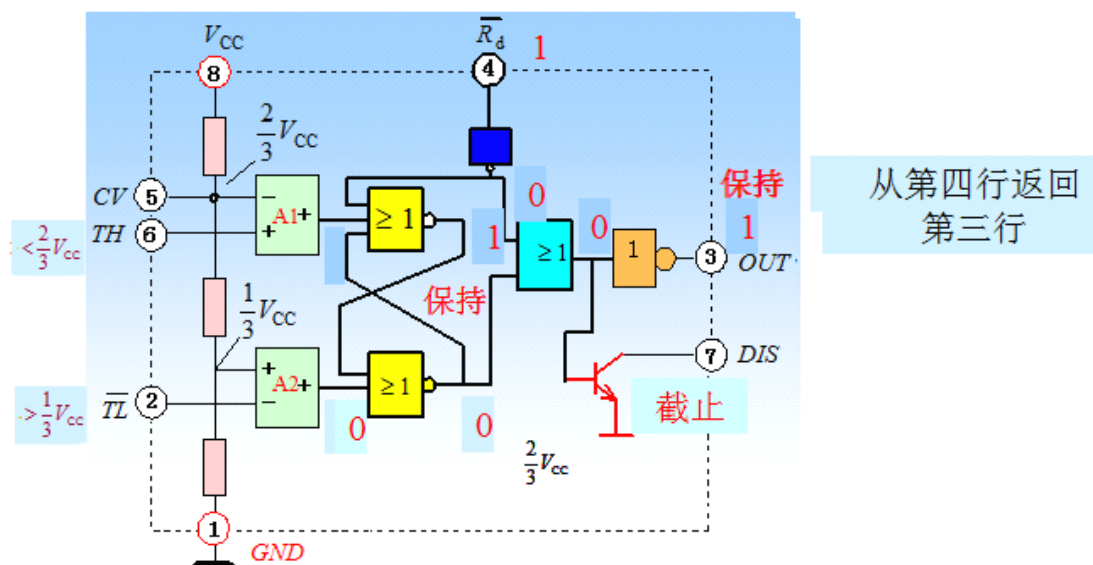


表 5-1 555 定时器功能表

TH	TL	$\overline{R_d}$	OUT	DIS
\times	\times	L	L	通
$> \frac{2}{3}V_{cc}$	$> \frac{1}{3}V_{cc}$	H	L	通
$< \frac{2}{3}V_{cc}$	$> \frac{1}{3}V_{cc}$	H	保持	保持
$< \frac{2}{3}V_{cc}$	$< \frac{1}{3}V_{cc}$	H	H	断

清零

回差现象

从 555 定时器的功能表可以看出：

1. 555 定时器有两个阈值（Threshold）电平，分别是 $\frac{1}{3}V_{CC}$ 和 $\frac{2}{3}V_{CC}$ ；
2. 输出端为低电平时三极管 TD 导通，7 脚输出低电平；输出端为高电平时三极管 TD 截止，如果 7 脚接一个上拉电阻，7 脚输出为高电平。所以当 7 脚接一个上拉电阻时，输出状态与 3 脚相同。

便于记忆：2脚— \overline{S} （低电平置位）；6脚—R（高电平复位）；

555 定时器的典型应用电路

单稳态触发器

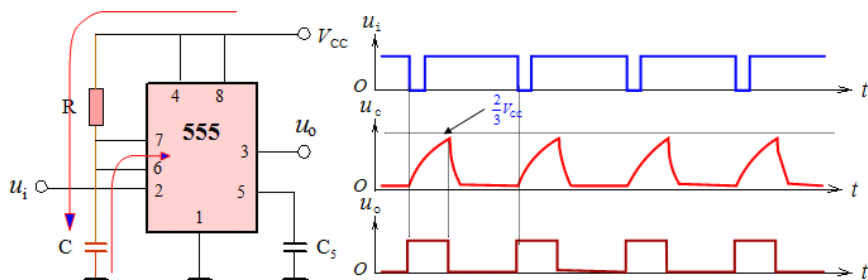


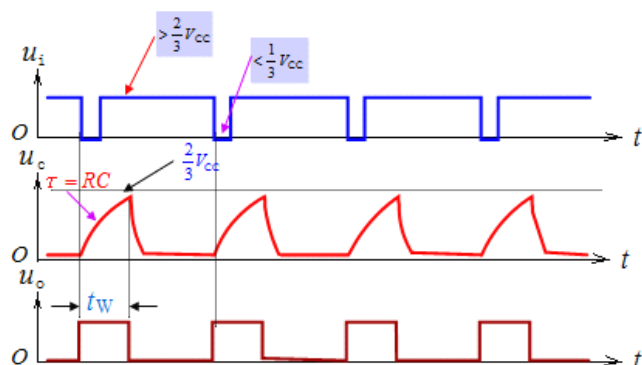
图5-2-4 单稳态触发器电路图

图5-2-5 单稳态触发器的波形图

这里要注意R的取值不能太小。（为什么？）

若R太小，当放电管导通时，灌入放电管的电流太大，会损坏放电管。

单稳态触发器暂稳态时间的计算



根据 u_c 的波形, 由过渡过程公式即可计算出暂稳态时间 t_w , t_w 电容C从0V充电到 $2/3 V_{CC}$ 的时间, 根据三要素方程:

$$u_c(t) = u_c(\infty) + [u_c(0) - u_c(\infty)]e^{-\frac{t}{\tau}}$$

为此需要确定三要素:

$u_c(0) = 0V$ 、 $u_c(\infty) = V_{CC}$ 、 $\tau = RC$, 当 $t = t_w$ 时, $u_c(t_w) = 2/3 V_{CC}$ 代入公式。于是可解出

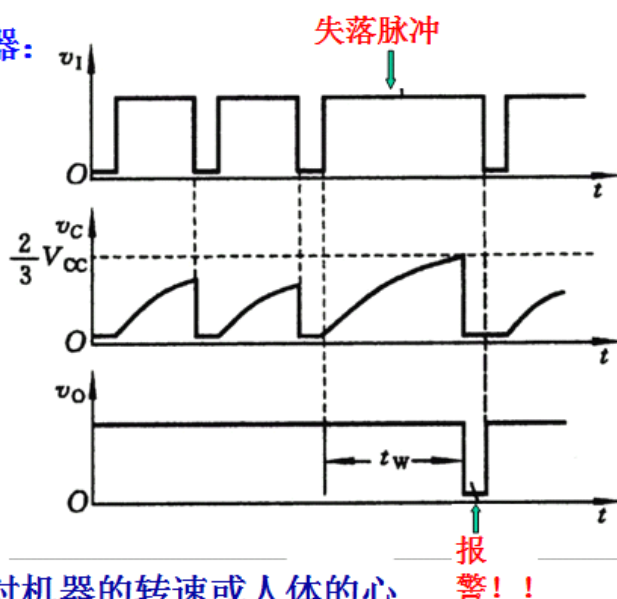
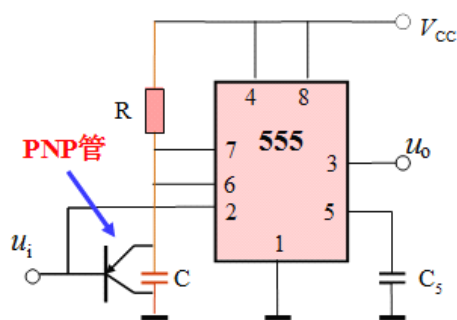
$$t_w = RC \ln 3 = 1.1RC$$

暂稳态的持续时间由RC确定!!!

注意: 触发输入信号的逻辑电平, 在无触发时是高电平, 必须大于 $2/3 V_{CC}$, 低电平必须小于 $1/3 V_{CC}$, 否则触发无效。

◇单稳态触发器的应用

★可重复触发的单稳态触发器:



该电路有何作用?

可作为失落脉冲检出电路, 对机器的转速或人体的心律(呼吸)进行监视, 当机器的转速降到一定限度或人体的心律不齐时就发出报警信号。

◆ 555定时器构成多谐振荡器 (Astable Multivibrator)

555定时器构成多谐振荡器构成的多谐振荡器如图5-2-9所示。它是将两个触发端2脚和6脚合并在一起，放电端7脚接于两电阻之间。

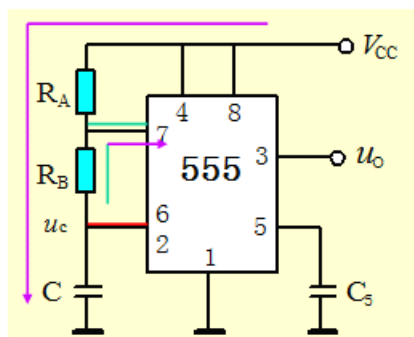


图5-2-9 多谐振荡器电路图

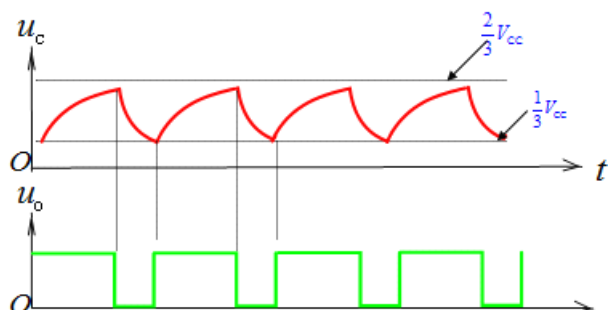
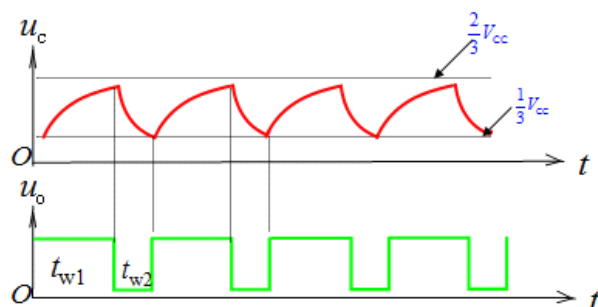


图5-2-10 多谐振荡器的波形

多谐振荡器参数的计算



输出波形的振荡周期可用过渡过程公式计算：

$$t_{w1}: u_C(0) = V_{CC}/3 \text{ V},$$

$u_C(\infty) = V_{CC}$ 、 $\tau_1 = (R_A + R_B)C$ 、
当 $t = t_{w1}$ 时， $u_C(t_{w1}) = 2V_{CC}/3$ 代入三要素方程。于是可解出

$$t_{w1} = 0.7(R_A + R_B)C$$

$t_{w2}: u_C(0) = 2V_{CC}/3 \text{ V}$ 、 $u_C(\infty) = 0 \text{ V}$ 、 $\tau_1 = R_B C$ 、
当 $t = t_{w2}$ 时， $u_C(t_{w2}) = V_{CC}/3$ 代入公式。于是可解出

$$t_{w2} = 0.7R_B C$$

$$T = t_{w1} + t_{w2} = 0.7(R_A + 2R_B)C$$

$$f = \frac{1}{T} = \frac{1.44}{(R_A + 2R_B)C}$$

$$D = \frac{T_1}{T} \times 100\% = \frac{t_{w1}}{T} \times 100\%$$

占空比 (Duration Ratio)

对于图5-2-9所示的多谐振荡器，因 $t_{w1} > t_{w2}$ ，它的占空比大于50%，占空比不可调节。图5-2-12是一种占空比可调的电路，该电路因加入了二极管，使电容器的充电和放电回路不同，可以调节电位器使充、放电时间常数相同。如果 $R_A = R_B$ ，调节电位器可以获得50%的占空比。

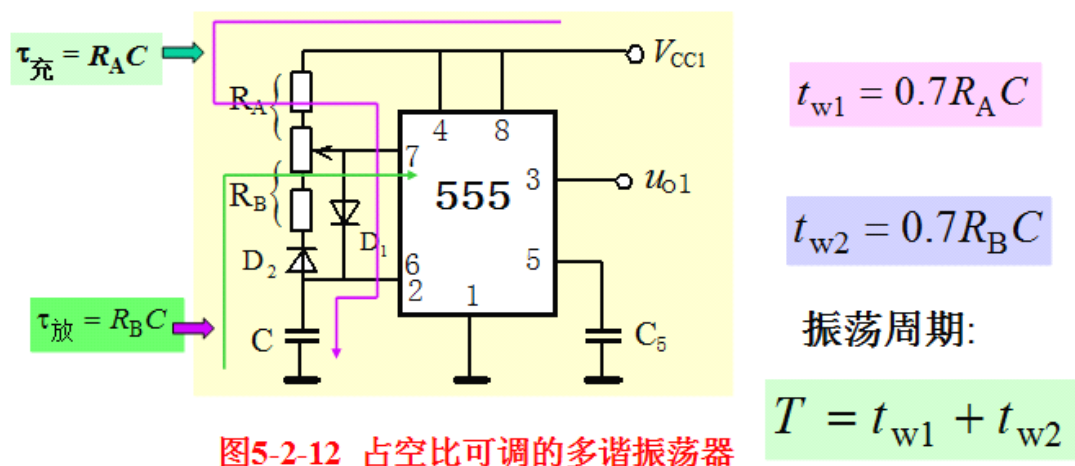


图5-2-12 占空比可调的多谐振荡器

◆ 555定时器构成施密特触发器 (Schmitt Trigger)

555定时器构成施密特触发器的电路图如图5-2-13所示，施密特触发器属于波形变换电路，该电路可以将正弦波、三角波、锯齿波变为脉冲信号。

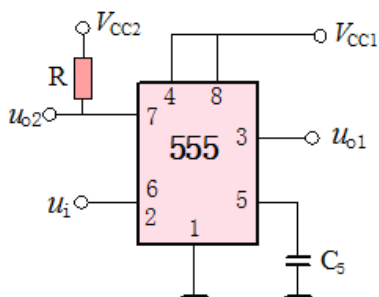


图5-2-13 施密特触发器电路图

施密特触发器的工作原理和多谐振荡器基本一致，无原则不同。只不过多谐振荡器是靠电容器的充放电去控制电路状态的翻转，而施密特触发器是靠外加电压信号去控制电路状态的翻转。所以，在施密特触发器中，外加信号的高电平必须大于 $\frac{2}{3}V_{cc}$ ，低电平必须小于 $\frac{1}{3}V_{cc}$ ，否则电路不能翻转。

由于施密特触发器无须放电端，所以利用放电端与输出端状态相一致的特点，从放电端加一上拉电阻后，可以获得与3脚相同的输出。但上拉电阻可以单独接另外一组电源，以获得与3脚输出不同的逻辑电平。

施密特触发器的输出波形如下：

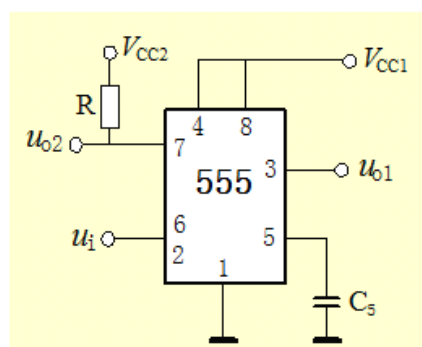


图5-2-13 施密特触发器电路图

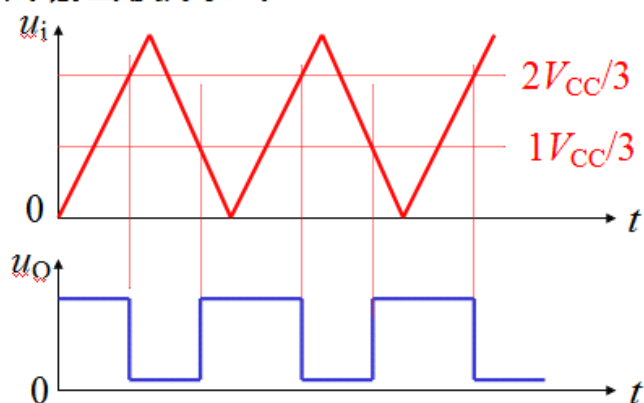


图5-2-14 施密特触发器的波形图

施密特触发器的主要用于对输入波形的整形。图5-2-14表示的是将三角波整形为方波,其它形状的输入波形也可以整形为方波。