第一章

计算机基础知识

现代计算机是在微电子学高速发展与计算数学日臻完善的基础上形成的，可以说现代计算机是微电子学与计算数学相结合的产物。微电子学的基本电路元件及其逐步向大规模发展的集成电路是现代计算机的硬件基础，而计算数学的数值计算方法与数据结构则是现代计算机的软件基础。

微电子学与计算数学发展至今已是内容繁多、体系纷纭，已有不少专著分别研究阐述。本章只是简要地阐述计算机中最基本的电路元件及最主要的数学知识。对于已学过这些知识的读者，本章将起到复习和系统化的作用。对于未曾接触过这些内容的读者，本章的内容是必要的入门知识，因为这些都是今后各章的基础。本章的目的也是为了使本书能够自成系统，读者不必依赖于更多的参考书刊。

1.1 数　制

数制是人们利用符号来计数的科学方法。数制可以有很多种，但在计算机的设计与使用上常使用的则为十进制、二进制、八进制和十六进制。

1.1.1 数制的基与权

数制所使用的数码的个数称为基；数制每一位所具有的值称为权。

十进制(Decimal System)十进制的基为“10”，即它所使用的数码为 0,1,2,3,4,5,6,7,8,9，共有 10 个。十进制各位的权是以 10 为底的幂，如下面这个数：

5	2	3	7	9	1
10^5	10^4	10^3	10^2	10^1	10^0
十万	万	千	百	十	个

其各位的权为个、十、百、千、万、十万，即以 10 为底的 0 幂、1 幂、2 幂、3 幂……等。故有时为了说话简便而顺次称其各位为 0 权位、1 权位、2 权位、3 权位等。

二进制(Binary System)二进制的基为“2”，即其使用的数码为 0,1，共两个。

二进制各位的权是以 2 为底的幂，如下面这个数：

二进制	1	1	0	1	1	1
	2^5	2^4	2^3	2^2	2^1	2^0
十进制	32	16	8	4	2	1

其各位的权为1,2,4,8……,即以2为底的0次幂、1次幂、2次幂、3次幂……等。故有时也顺次称其各位为0权位、1权位、2权位……等。

八进制(Octave System)八进制的基为"8",即其数码共有8个:0,1,2,3,4,5,6,7。八进制的权为以8为底的幂,有时也顺次称其各位为0权位、1权位、2权位等。

十六进制(Hexadecimal System)十六进制的基为"16",即其数码共有16个:0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F。十六进制的权为以16为底的幂,有时也称其各位的权为0权、1权、2权等。

在微型计算机中这些数制都是常用到的,但在本书后面的内容中,二进制和十六进制更为常用,希初学者注意。

1.1.2 为什么要用二进制

电路通常只有两种稳态:导通与阻塞,饱和与截止,高电位与低电位等。具有两个稳态的电路称为二值电路。因此,用二值电路来计数时,只能代表两个数码:0和1。如以1代表高电位,则0代表低电位,所以,采用二进制,就可以利用电路进行计数工作。而用电路来组成计算机,则有运算迅速、电路简便、成本低廉等优点。

1.1.3 为什么要用十六进制

用十六进制既可简化书写,又便于记忆。如下列一些等值的数:

$$1000_{(2)}=8_{(16)}(\text{即 }8)$$
$$1111_{(2)}=F_{(16)}(\text{即 }15)$$
$$11\ 0000_{(2)}=30_{(16)}(\text{即 }48)$$
$$1111\ 1001_{(2)}=F9_{(16)}(\text{即 }249)$$

可以看出用十六进制,可以写得短些,也更易于记忆。试看一下,是记$F9_{(16)}$容易呢?还是记1111 1001容易呢?尤其是,当二进制位数很多时,更可看到十六进制的优点了。如:

$$1010\ 1101\ 1000\ 0101_{(2)}=AD85_{(16)}$$

显然,记$AD85_{(16)}$要比记16位的二进制数容易得多了。

上面书写的意义:在数字后面加上(2)和(16)是指二进制和十六进制。同理如写(8)和(10)则表示为八进制和十进制。也有用字母符号来表示这些数制的:

B——二进制,H——十六进制,D——十进制,O——八进制。

通常如上下文可以理解所写的数是什么进位时,就不必附加数制符号。

1.1.4 数制的转换方法

由于我们习惯用十进制计数,在研究问题或讨论解题的过程时,总是用十进制来考虑和书写的。当考虑成熟后,要把问题变成计算机能够"看得懂"的形式时,就得把问题中的所有十进制数转换成二进制代码。这就需要用到"十进制数转换成二进制数的方法"。在计算机运算完毕得到二进制数的结果时,又需要用到"二进制数转换为十进制数的方法",才能把运算结果用十进制形式显示出来。

1. 十进制数转换成二进制数的方法

一般可用下列方法来求一个十进制数的二进制代码:

用 2 除该十进制数可得商数及余数，则此余数为二进制代码的最小有效位(LSB)之值。

再用 2 除该商数，又可得商数和余数，则此余数为 LSB 左邻的二进制数代码。

用同样的方法继续用 2 除下去，就可得到该十进制数的二进制代码。

【例】 求 13 的二进制代码。其过程如下：

0	1	
2) 1	1	↓（由上往下读，可从左至右写出二进制代码）
2) 3	0	
2) 6	1	
2) 13		

结果为：1101

上面是十进制整数转换成二进制数的“除 2 取余法”。

如果十进制小数，要转换成二进制小数，则要采取“乘 2 取整法”：

一个十进制的小数乘以 2 之后可能有进位使整数位为 1（当该小数大于 0.5 时），也可能没有进位，其整数位仍为零（当该小数小于 0.5 时）。这些整数位的结果即为二进制的小数位结果。举例如下：

【例】 求十进制数 0.625 的二进制数。

其进行步骤可以用乘法的竖式算法：

$$\begin{array}{r} 0.625 \\ \times\quad 2 \\ \hline 1.25 \end{array}$$

整数部为 1，即二进制小数后第一位为 1。

$$\begin{array}{r} 0.25 \\ \times\quad 2 \\ \hline 0.50 \end{array}$$

整数部为 0，即二进制小数后第二位为 0。

$$\begin{array}{r} 0.50 \\ \times\quad 2 \\ \hline 1.00 \end{array}$$

整数部为 1，即二进制小数后第三位为 0。

至此就不用再算下去了。如果小数位不是 0.00，则还得继续乘下去，直至变成 0.00 为止。因此，一个十进制小数在转换为二进制小数时有可能无法准确地转换。如十进制数 0.1 转换为二进制数时为 0.0001100110…。因此，只能近似地以 0.00011001 来表示之。

2. 二进制数转换成十进制数的方法

这可以由二进制数各位的权乘以各位的数(0 或 1)再加起来就得到十进制数。

【例】

	1	0	1	0	1	1
权：	2^5	2^4	2^3	2^2	2^1	2^0
乘积：	32	0	8	0	2	1
累加：			43			
结果：		$43_{(10)}$				

二进制小数转换为十进制时也可用同样的方法，不过二进制数小数的各位的权是 2^{-1}，2^{-2}，2^{-3}……例如二进制数 0.101 转换为十进制数的方法如下：

	0	. 1	0	1	
权：	2^0	2^{-1}	2^{-2}	2^{-3}	
乘积：	0	0.5	0	0.125	
累加：		0.625			
结果：	$0.625_{(10)}$				

由上述可得出两点注意事项：

(1) 一个二进制数可以准确地转换为十进制数，而一个带小数的十进制数不一定能够准确地用二进制来表示；

(2) 带小数的十进制数在转换为二进制数时，以小数点为分界，整数和小数要分别转换。

此外，还有其他各种数制之间的转换，其方法和上述的差不多，都可以从数制的定义中找到转换的方法。

1.2 逻辑电路

逻辑电路由其三种基本门电路(或称判定元素)所组成。图 1-1 是这几个门电路的名称、符号及表达式。

在这三个基本元素的基础上，还可发展成如图 1-2 那样的更复杂的逻辑电路。

最后一个，叫做缓冲器(Buffer)。实为两个非门串联以达到改变输出电阻的目的。如 A 点左边电路的输出电阻很高的话，则经过这个缓冲门之后，在 Y 点处的输出电阻就可以变得低许多倍。这就能够提高带负载的能力。

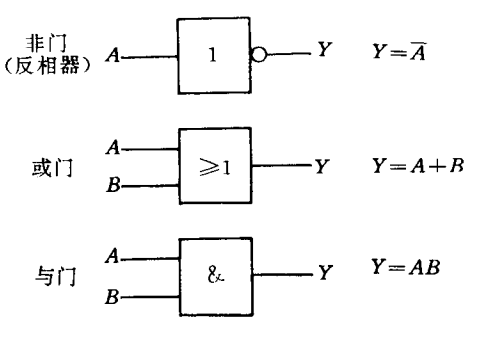


图 1-1 三个基本门电路

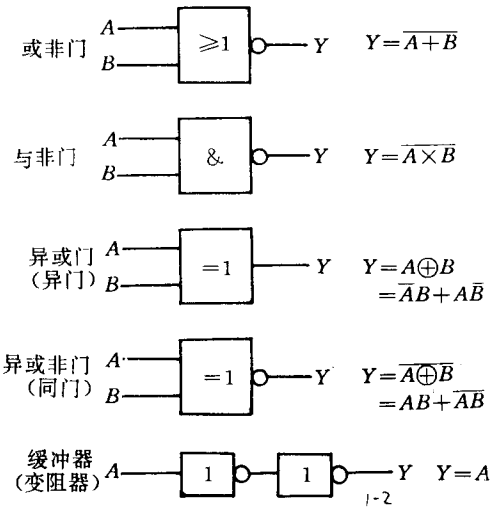


图 1-2 其它的门电路

1.3 布尔代数

布尔代数亦称开关代数或逻辑代数，也和一般代数一样，可以写成下面这样的表达式：

$$Y=f(A,B,C,D)$$

但它有两个特点：

第一，其中的变量 $A,B,C,D\cdots$ 等均只有两种可能的数值：0 或 1。布尔代数变量的数值并无大小之意，只代表事物的两个不同性质，如用于开关上，则：

0 代表关(断路)或低电位

1 代表开(通路)或高电位

如用于逻辑推理，则：

0 代表错误(伪)

1 代表正确(真)

第二，函数 f 只有三种基本方式："或"运算，"与"运算及"反"运算。下面分别讲述这三种运算的规律。

1.3.1 "或"运算($Y=A+B$)

由于 A 与 B 只有 0 或 1 的可能取值，所以可以将其各种可能结果列写如下：

$$\left.\begin{array}{l}Y=0+0=0\rightarrow Y=0\\ Y=0+1=1\\ Y=1+0=1\\ Y=1+1=1\end{array}\right\}\rightarrow Y=1$$

第四个式子与一般的代数加法不符，这是因为 Y 也只能有两种数值：0 或 1。

上面四个式子可归纳成两句话，两者皆伪者则结果必伪，有一为真者则结果必真。这个结论也可推广至多变量：$A,B,C,D\cdots$，各变量全伪者则结果必伪，有一为真者则结果必真，写成表达式如下：

设 $Y=A+B+C+D+\cdots$

则

$$\left.\begin{array}{l}Y=0+0+\cdots+0=0\rightarrow Y=0\\ Y=1+0+\cdots+0=1\\ Y=0+1+\cdots+0=1\\ \cdots\cdots\\ Y=1+1+1\cdots+1=1\end{array}\right\}\rightarrow Y=1$$

这意味着，在多输入的"或"门电路中，只要其中一个输入为 1，则其输出必为 1。或者说只有全部输入均为 0 时，输出才为 0。

或运算有时也称为"逻辑或"。当 A 和 B 为多位二进制数时，如

$$A=A_1A_2A_3\cdots A_n$$

$$B=B_1B_2B_3\cdots B_n$$

则进行"逻辑或"运算时，各对应位分别进行"或"运算：

$$\begin{aligned}Y&=A+B\\&=(A_1+B_1)(A_2+B_2)(A_3+B_3)\cdots(A_n+B_n)\end{aligned}$$

【例】 设 $A=10101$

$B=11011$

则

$$\begin{aligned}Y&=A+B\\&=(1+1)(0+1)(1+0)(0+1)(1+1)\end{aligned}$$

$=11111$

写成竖式则为：

$$\begin{array}{r} 1\ 0\ 1\ 0\ 1 \\ +)\ 1\ 1\ 0\ 1\ 1 \\ \hline 1\ 1\ 1\ 1\ 1 \end{array}$$

注意，1“或”1 等于 1，是没有进位的。

1.3.2 “与”运算($Y=A\times B$)

根据 A 和 B 的可能取值(0 或 1)可以列写出下列各种可能的运算结果：

$$\left.\begin{array}{l} Y=0\times 0=0 \\ Y=1\times 0=0 \\ Y=0\times 1=0 \end{array}\right\}\rightarrow Y=0$$

$$Y=1\times 1=1\rightarrow Y=1$$

这种运算结果也可归纳成两句话：二者为真者结果必真，有一为伪者结果必伪。同样，这个结论也可推广至多变量：各变量均为真者结果必真，有一为伪者结果必伪。写成表达式如下：

设 $Y=A\times B\times C\times D\times\cdots$

则

$$\left.\begin{array}{l} Y=0\times 0\times\cdots\times 0=0 \\ Y=1\times 0\times\cdots\times 0=0 \\ Y=0\times 1\times\cdots\times 0=0 \\ \cdots \end{array}\right\}\rightarrow Y=0$$

$$Y=1\times 1\times 1\cdots\times 1=1\rightarrow Y=1$$

这意味着，在多输入“与”门电路中，只要其中一个输入为 0，则输出必为 0，或者说，只有全部输入均为 1 时，输出才为 1。

与运算有时也称为“逻辑与”。当 A 和 B 为多位二进制数时，如

$$A=A_1A_2A_3\cdots A_n$$

$$B=B_1B_2B_3\cdots B_n$$

则进行“逻辑与”运算时，各对应位分别进行“与”运算：

$$\begin{aligned} Y &= A\times B \\ &= (A_1\times B_1)(A_2\times B_2)(A_3\times B_3)\cdots(A_n\times B_n) \end{aligned}$$

【例】 设 $A=11001010$

$B=00001111$

则 $Y=A\times B$

$=(1\times 0)(1\times 0)(0\times 0)(0\times 0)(1\times 1)(0\times 1)(1\times 1)(0\times 1)$

$=00001010$

写成竖式则为

$$\begin{array}{r} 1\ 1\ 0\ 0\ \ 1\ 0\ 1\ 0 \\ \times)\ 0\ 0\ 0\ 0\ \ 1\ 1\ 1\ 1 \\ \hline 0\ 0\ 0\ 0\ \ 1\ 0\ 1\ 0 \end{array}$$

由此可见，用“0”去和一个数位相“与”，就是将其“抹掉”而成为“0”，用“1”去和一个数位相“与”，就是将此数位“保存”下来。这种方法在计算机的程序设计中经常会用到而称为“屏蔽”。上面的 B 数(0000 1111)则称为“屏蔽字”，它将 A 数的高 4 位给屏蔽起来而都变成 0 了。

1.3.3 反运算($Y=\overline{A}$)

如果一件事物的性质为 A，则其经过“反”运算之后，其性质必与 A 相反，用表达式表示为：

$$Y=\overline{A}$$

这实际上也是反相器的性质。所以在电路实现上，反相器是反运算的基本元件。

反运算也称为“逻辑非”或“逻辑反”。

当 A 为多位数时，如

$$A=A_1A_2A_3\cdots A_n$$

则其“逻辑反”为 $Y=\overline{A_1}\,\overline{A_2}\,\overline{A_3}\cdots\overline{A_n}$

【例】 设 $A=1101\quad 0000$

则 $Y=0010\quad 1111$

1.3.4 布尔代数的基本运算规律

1. 恒等式

$A\cdot 0=0\qquad A\cdot 1=A\qquad A\cdot A=A$

$A+0=A\qquad A+1=1\qquad A+A=A$

$A+\overline{A}=1\qquad A\cdot\overline{A}=0\qquad \overline{\overline{A}}=A$

2. 运算规律

与普通代数一样，布尔代数也有交换律、结合律、分配律，而且它们与普通代数的规律完全相同。

交换律 $A\cdot B=B\cdot A$

$A+B=B+A$

结合律 $(AB)C=A(BC)=ABC$

$(A+B)+C=A+(B+C)=A+B+C$

分配律 $A(B+C)=AB+AC$

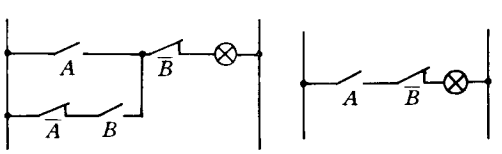
$(A+B)(C+D)=AC+AD+BC+BD$

我们利用这些运算规律及恒等式，就可以化简很多逻辑关系式。

【例 1】 $A+AB=A(1+B)=A$

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

【例 2】 如果原设计继电器线路如图 1-3a，现用逻辑关系，化简线路。



(a) (b)

图 1-3 用布尔代数化简继电器线路

首先我们可以把图 1-3a 中触点(如同开关)和灯的关系用布尔代数表达出来：

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

其中 A 与 $\overline{A}$ 是同一继电器的常开与常闭触点。一般我们把常开触点定为变量 A,B，则常闭触点相应的为 $\overline{A},\overline{B}$。

下面我们来用布尔代数知识简化：

$$\begin{aligned} Y &(A+\overline{A}B)\cdot\overline{B} \\ &=A\overline{B}+\overline{A}B\cdot\overline{B} \\ &=A\overline{B}+0 \\ &=A\overline{B} \end{aligned}$$

从而我们可以用图 1-3 中 b 电路，代替原设计电路 a。电路大大简化了，但能起到同样的作用。

1.3.5 摩根定理

在电路设计时，人们手边有时没有“与”门，而只有“或”门和“非”门。或者只有“与”门和“非”门，没有“或”门。利用摩根定理，可以帮你解决元件互换问题。

摩根定理式为：

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

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

推广到多变量：

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

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

此定理我们可以用表格法验证。

从表 1-1 中可以看出变量在各种可能取值的情况下，式$\overline{A+B}$的值始终和 $\overline{A}\cdot\overline{B}$ 相同。所以证明了$\overline{A+B}=\overline{A}\cdot\overline{B}$，同理也证明了$\overline{A\cdot B}=\overline{A}+\overline{B}$。

表 1-1 摩根定理验证

二变量 A	B	$\overline{A+B}$	$\overline{A}\cdot\overline{B}$	$\overline{A\cdot B}$	$\overline{A}+\overline{B}$
0	0	1	1	1	1
0	1	0	0	1	1
1	0	0	0	1	1
1	1	0	0	0	0
变量可能取值		相等		相等	

至于多变量的摩根定理，用相同的方法同样可以得到证明。

这个定理可以用一句话，“头上切一刀，下面变个号”来记忆使用。

【例】

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

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

1.3.6 真值表及布尔代数式的关系

当人们遇到一个因果问题时，常常把各种因素全部考虑进去，然后研究结果。真值表也就

是这种考虑问题的方法的一种表格形式。

例如考虑两个一位的二进制数 A 和 B 相加，其本位的和 S 及向高一位进位 C 的结果如何？

全面考虑两个一位二进制数，可能出现四种情况：或 $A=0,B=0$；或 $A=0,B=1$；或 $A=1,B=0$；或 $A=1,B=1$。（一般 n 个因素可有 $2n$ 种情况）。这实质是两个一位数（可零可 1）的排列。我们可以把它们列入表内，如表 1-2 左边部分所示。然后，对每一种情况进行分析。当 A 和 B 都为 0 时，S 为 0，进位 C 也为 0；当 A 为 0 且 B 为 1 时，S 为 1，进位 C 为 0；当 A 为 1 且 B 为 0 时，S 为 1，进位 C 为 0；当 A 为 1 且 B 也为 1 时，由于 S 是一位数所以得 0，而有进位 $C=1$。填入表 1-2，如表右部分。

表 1-2

	A	B	S	C
第一项	0	0	0	0
第二项	0	1	1	0
第三项	1	0	1	0
第四项	1	1	0	1

我们称表 1-2 为这个逻辑问题（二进制数相加的运算由于取值只能 1、0，所以可以转化为逻辑问题）的真值表。

从真值表上我们可以方便地分别写出 S 和 A、B 关系的布尔式子，以及 C 和 A、B 关系的布尔式子。

对于 C，因为只有 A 与 B 都为 1 时，它才为 1，所以一分析即可知 $C=A\cdot B$

对于 S，因为在表中第二项或第三项都可能为 1，而第二项要求 $A=0$ 与 $B=1$，在写式子时要求能使 S 为 1，显然只有写 $\overline{A}\times B=\overline{0}\times 1=1$。所以第二项布尔式子就是 $\overline{A}\cdot B$。对于第三项要求 $A=1$ 与 $B=0$，在写式子时要使 S 为 1，显然只有写 $A\times\overline{B}=1\times\overline{0}=1$。所以第三项布尔式子就是 $A\cdot\overline{B}$。从而我们可以写出 S 和 A、B 的关系式为 $S=\overline{A}B+A\overline{B}$。

这种从真值表写出布尔代数式的方法可以用下面两段话来描述。

(1) 写布尔代数式先看真值表中结果为 1 的项，有几项就有几个“或”项；

(2) 每一项各因素之间是“与”关系。写该项时每个因素都写上，然后添“反”。至于哪个因素 要加“反”（上横线）要看该因素在这项里是否是“0”状态，是“0”状态则加“反”，否则不加“反”。

写出布尔代数式后，反过来去检查写得对不对。例：表 1-2 中第一项 $A=0$ 和 $B=0$ 代入式 $S=\overline{A}B+A\overline{B}$，则 $S=\overline{0}\cdot 0+0\cdot\overline{0}=0$；表中第二项 $A=0$ 和 $B=1$ 代入式 $S=\overline{A}B+A\overline{B}$ 则 $S=\overline{0}\cdot 1+0\cdot\overline{1}=1+0=1$；依次类推地代入检查。如果四项都对，则式子 $S=\overline{A}B+A\overline{B}$ 确实代表了真值表中 S 和 A、B 之间的逻辑关系。

一般的说来，用真值表来描述问题，不仅全面，而且通过它来写布尔代数手续也很简便。

1.4 二进制数的运算及其加法电路

作为算术的基本运算，众所周知，是有四种算法：加、减、乘和除。在微型计算机中常常只有

加法电路，这是为了使硬件结构简单而成本较低。不过，只要有了加法电路，也能完成算术的四种基本运算。

1.4.1 二进制数的相加

两个二进制数相加的几个例子：

(a)

```
    1     A
+)  1     B
-------
  1 0     S
```

进位

(b)

```
    0 1   A
+)  1 0   B
---------
    1 1   S
```

(c)

```
    1 1   A
+)  1 1   B
---------
  1 1 0   S
```

进位

(d)

```
    1 1     C
    0 1 1   A
+)  0 1 1   B
-----------
  1 1 0     S
```

进位

例(a)中，加数 A 和被加数 B 都是一位数，其和 S 变成二位的，这是因为相加结果产生进位之故。

例(b)中，A 和 B 都是二位的，相加结果 S 也是二位的，因为相加结果不产生进位。

例(c)中，A 和 B 都是二位的，相加结果 S 是三位的，这也是产生了进位之故。

例(d)中，是例 c 的另一种写法，以便看出"进位"究竟是什么意义。第一位(或称 0 权位)是不可能有进位的，要求参与运算的就只有两个数 A_0 和 B_0，其结果为 S_0。第二位(或称 1 权位)就是三个数 A_1、B_1 及 C_1 参与运算了。其中 C_1 是由于第一位相加的结果产生的进位。此三个数相加的结果其总和为 $S_1=1$，同时又产生进位 C_2，送入下一位(第三位)。第三位(或称 2 权位)也是三个数 A_2、B_2 及 C_2 相加参加运算。由于 A_2 及 B_2 都是 0，所以 C_2 即等于第三位的相加结果 S_2。

从以上几例的分析可得下列结论：

(1) 两个二进制数相加时，可以逐位相加。如二进制数可以写成：

$$A=A_3A_2A_1A_0$$

$$B=B_3B_2B_1B_0$$

则从最右边第一位(即 0 权位)开始，逐位相加，其结果可以写成：

$$S=S_3S_2S_1S_0$$

其中各位是分别求出的：

$$S_0=A_0+B_0\rightarrow\text{进位 }C_1$$

$$S_1=A_1+B_1+C_1\rightarrow\text{进位 }C_2$$

$$S_2=A_2+B_2+C_2\rightarrow\text{进位 }C_3$$

$$S_3=A_3+B_3+C_3\rightarrow\text{进位 }C_4$$

而最后所得的和就是：

$$C_4S_3S_2S_1S_0=A+B$$

(2) 右边第一位相加的电路要求：

输入量为两个，即 A_0 及 B_0

输出量为两个，即 S_0 及 C_1。

这样的一个二进制位相加的电路称为半加器(Half Adder)。

(3) 右边第二位开始，各位可以对应相加。各位对应相加时的电路要求：

输入量为三个，即 A_i、B_i、C_i。

输出量为两个，即 S_i、C_{i+1}。

其中 $i=1,2,3,\cdots,n$。这样的一个二进制位相加的电路称为全加器(Full Adder)。

1.4.2 半加器电路

要求：有两个输入端，以供两个代表数字(A_0、B_0)的电位输入；有两个输出端，用以输出总和 S_0 及进位 C_1。

这样的电路可能出现的状态可以用图 1-4 中的表表示出来。此表在布尔代数中称为真值表(True Table)。

考察一下 C_1 与 A_0 及 B_0 之关系，即可看出这是"与"的关系，即：

$$C_1 = A_0 \times B_0$$

再看一下 S_0 与 A_0 及 B_0 之关系，也可看出这是"异或"的关系，即：

$$S_0 = A_0 \oplus B_0$$
$$= A_0\bar{B}_0 + \bar{A}_0 B_0$$

即只有当 A_0 及 B_0 二者相异时，才起到或的作用；二者相同时，则其结果为 0。因此，可以用"与门"及"异或门"(或称"异门")来实现真值表的要求。图 1-4 就是这个真值表及半加器的电路图。

真值表

A_0	B_0	C_1	S_0
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0
		与门	异或门

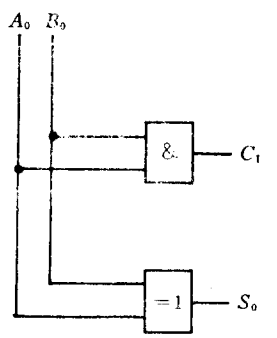


图 1-4 半加器的真值表及电路

1.4.3 全加器电路

要求：有三个输入端，以输入 A_i、B_i 和 C_i，有两个输出端，即 S_i 及 C_{i+1}。

其真值表可以写成如图 1-5 所示。由此表分析可见，其总和 S_i 可用"异或门"来实现，而其进位 C_{i+1} 则可以用三个"与门"及一个"或门"来实现，其电路图也画在图 1-5 中。

这里遇到了三个输入的"异或门"的问题。如何判断多输入的"异或门"的输入与输出的关系呢？判断法是：多输入 A、B、C、$+D$、…中为"1"的输入量的个数为零及偶数时，则输出为 0；为奇数时，则输出为 1。

真值表

A_i	B_i	C_i	C_{i+1}	S_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1
			先“与”后“或”	“异或”

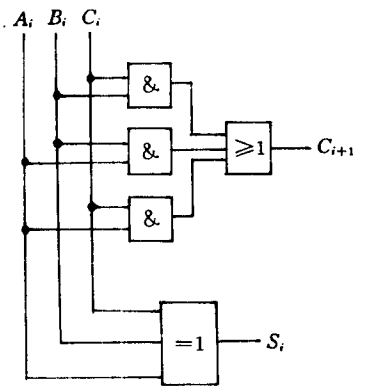


图 1-5　全加器的真值表及电路

1.4.4　半加器及全加器符号

图 1-6a 为半加器符号，图 1-6b 为全加器符号。

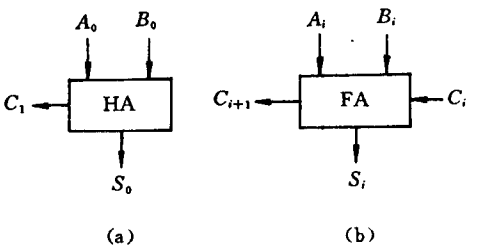


图 1-6　半加器及全加器的符号

1.4.5　二进制数的加法电路

设　$A=1010=10_{(10}$

　　$B=1011=11_{(10)}$

则可安排如下的加法电路(图 1-7)。

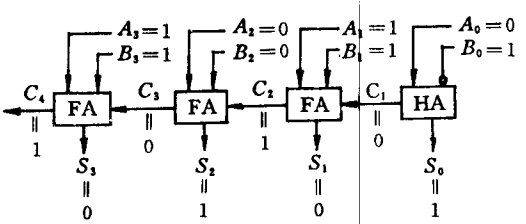


图 1-7　4 位的二进制加法电路

A 与 B 相加，写成竖式算法如下：

$$\begin{array}{l} A: 1\ \ 0\ \ 1\ \ 0 \\ B: 1\ \ 0\ \ 1\ \ 1\ \ (+ \\ \hline S: 10\ \ 1\ \ 0\ \ 1 \end{array}$$

即其相加结果为 $S=10101$。

从加法电路，可看到同样的结果：

$$\begin{aligned} S &= C_4S_3S_2S_1S_0 \\ &= 10101 \end{aligned}$$

1.4.6 二进制数的减法运算

在微型计算机中，是没有专用的减法器的，而是将减法运算改变为加法运算。其原理是这样：将减数 B 变成其补码后，再去和被减数 A 相加，其和(如有进位的话，则舍去进位)就是两数之差。

补码是什么呢？对于二进制数来说，简言之，可用下式来表示：

$$\text{补码}=\text{反码}+1$$

这就是说，如有一个二进制数为 A，这就是原码，则其反码为 $\overline{A}$，于是补码 A' 可以写成：

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

补码并非只有二进制数才有。在十进制、十六进制等各种进制中都是存在的。如在十进制中原码为 6 的补码是 4，原码为 64 的补码是 36，原码为 642 的补码是 358 等等。

由此可见：原码＋补码的结果如下：

$$6+4=10$$

$$64+36=100$$

$$642+358=1000$$

即原码与补码互相补充而能得到一个进位数：

一位的原码加补码得到的是二位数 10；

二位的原码加补码得到的是三位数 100；

三位的原码加补码得到的是四位数 1000。

在做十进制减法时，也可以利用补码而将减法运算变成加法运算。例如 73－15，可利用 15 的补码 85 而使减法变成加法：73＋85＝158，把进位位 1 去掉，58 即为 73 与 15 之差。不过在十进制中用电路由原码求补码并不十分方便，所以没有人用这个规律去算减法。

在二进制中，原码每位变反，可得反码。如 10100 的反码为 01011，用二位电路很容易做到，而原码与反码相加正好差 1 而未有进位(无溢出)。如上例

原码：10100

反码：01011

原码＋反码＝11111

如果反码加 1 后再去与原码相加就得：

$$\text{原码}+(\text{反码}+1)=10100+01100$$

所以，在二进制中，常用反码加 1 的方法来获得补码。

这在计算机中很有方便之处，因为二进制电路由原码求反码是很容易的，这在下面就会看到。

有了补码，就可以将减法变成加法来运算了。请看下面的例子。

【例 1】 求 $Y=8_{(10)}-4_{(10)}=?$

解：因为 $A=8_{(10)}=1000_{(2)}$

$B=4_{(10)}=0100_{(2)}$

则 $B'=1011+1=1100_{(2)}$

于是 $Y=A-B$

$=A+B'$

$=1000+1100$

$=1\ 0100$

↑——进位，应舍去

$=0100_{(2)}=4_{(10)}$

【例 2】 求 $Y=F_{(H)}-A_{(H)}=?$ （即求 15 减 10 之差）

设 $A=F_{(H)}=1111_{(B)}=15_{(D)}$

$B=A_{(H)}=1010_{(B)}=10_{(D)}$

则 $B'=0101+1=0110_{(B)}$

∴ $Y=1111+0110$

$=1\ 0101$

$=$ ↑——进位，舍去

$=0101_{(B)}$ （结果为 5）

1.4.7 可控反相器及加法/减法电路

利用补码可将减法变为加法来运算，因此需要有这么一个电路，它能将原码变成反码，并使其最小位加 1。

图 1-8 的可控反相器就是为了使原码变为反码而设计的。这实际上是一个异或门（异门），两输入端的异或门的特点是：两者相同则输出为 0，两者不同则输出为 1。用真值表来表示这个关系，更容易看到其意义（表 1-3）。

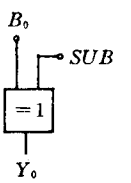


图 1-8 可控反相器

由此真值表可见，如将 SUB 端看作控制端，则当在 SUB 端加上低电位时，Y 端的电平就和 B_0 端的电平相同。在 SUB 端加上高电平，则 Y 端的电平和 B_0 端的电平相反。

表 1-3

SUB	B_0	Y	Y 与 B_0 的关系	
0	0	0	Y 与 B_0 相同	同相
	1	1	Y 与 B_0 相同	
1	0	1	Y 与 B_0 相反	反相
	1	0	Y 与 B_0 相反	

就是利用这个特点，在图 1-7 的四位二进制数加法电路上增加四个可控反相器并将最低位的半加器也改用全加器就可以得到如图 1-9 的四位二进制数加法器/减法器电路了，因为这

个电路既可以作为加法器电路(当 $SUB=0$),又可以作为减法器电路(当 $SUB=1$)。

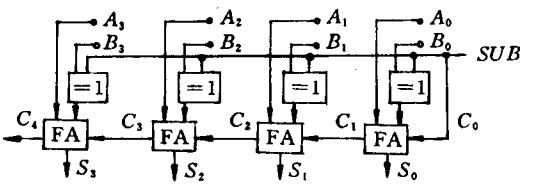


图 1-9　二进制补码加法器/减法器

如果我们有下面两个二进制数:

$$A=A_3A_2A_1A_0$$

$$B=B_3A_2A_1B_0$$

则可将此二数的各位分别送入该电路的对应端,于是

当 $SUB=0$ 时,电路作加法运算:$A+B$

当 $SUB=1$ 时,电路作减法运算:$A-B$

图 1-9 电路的原理如下:当 $SUB=0$ 时,各位的可控反相器的输出与 B 的各位同相,所以图 1-9 和图 1-7 的原理完全一样,各位均按位相加。结果 $S=S_3S_2S_1S_0$,而其和为:$C_4S=C_4S_3S_2S_1S_0$。

当 $SUB=1$ 时,各位的反相器的输出与 B 的各位反相。注意,最右边第一位(即 S_0 位)也是用全加器,其进位输入端与 SUB 端相连,因此其 $C_0=SUB=1$。所以此位的相加即为:

$$A_0+\overline{B}_0+1$$

其它各位也是:

$$A_1+\overline{B}_1+C_1$$

$$A_2+\overline{B}_2+C_2$$

$$A_3+\overline{B}_3+C_3$$

因此其总和输出 $S=S_3S_2S_1S_0$ 就是:

$$\begin{aligned}S&=A+\overline{B}+1\\&=A_3A_2A_1A_0+\overline{B}_3\overline{B}_2\overline{B}_1\overline{B}_0+1\\&=A+B'\\&=A-B\end{aligned}$$

当然,此时,C_4 如不等于 0 的话,则要被舍去。

习　题

1. 下列各二进制数相当于十进制数的多少?

(1) $1101_{(2)}$

(2) $11010_{(2)}$

(3) $110100_{(2)}$

(4) $10101001_{(2)}$

2. 第 1 题各二进制数相当于十六进制数的多少?

3. 试简述三个门电路的基本元素在电路中对电平的高低各起什么作用？
4. 布尔代数有哪两个特点？
5. 布尔代数的“或运算”结果可用哪两句话来归纳？其“与运算”结果又可归纳成哪两句话？
6. 什么叫原码、反码及补码？
7. 为什么需要半加器和全加器，它们之间的主要区别是什么？
8. 用补码法写出下列减法的步骤：
   (1) $1111_{(2)}-1010_{(2)}=?_{(2)}=?_{(10)}$
   (2) $1100_{(2)}-0011_{(2)}=?_{(2)}=?_{(10)}$
9. 做出 $101011_{(2)}+011110_{(2)}$ 的门电路图并求其相加的结果。
10. 做出第 9 题中二数相减的门电路图并求其相减的结果。

第二章

微型计算机的基本组成电路

任何一个复杂的电路系统都可以划分为若干电路，这些电路大都由一些典型的电路组成。微型计算机就是由若干典型电路通过精心设计而组成的，各个典型电路在整体电路系统中又称为基本电路部件。

本章就是对微型计算机中最常见的基本电路部件的名称及电路原理作一简单介绍。这些基本电路中最主要的是算术逻辑单元(Arithmatic Logical Unit，ALU)、触发器(Trigger)、寄存器(Register)、存储器(Memory)及总线结构等。在本章中，数据在这些部件之间的流通过程以及“控制字”的概念也将逐步地引出。所有这些内容都是组成微型计算机的硬件基础。

2.1 算术逻辑单元(ALU)

顾名思义，这个部件既能进行二进制数的四则运算，也能进行布尔代数的逻辑运算。

第一章已讲过，二进制数的运算电路只能算加法。增加可控反相器后，又能进行减法，所以上章最后介绍的二进制补码加法器/减法器就是最简单的算术部件。但是，只要利用适当的软件配合，乘法也可以变成加法来运算，除法也可变成减法来运算。

如果在这个基础上，增加一些门电路，也可使简单的 ALU 进行逻辑运算。所谓逻辑运算就是指“与”运算和“或”运算。为了不使初学者陷入复杂的电路分析之中，本教程不打算在逻辑运算问题上开展讨论。

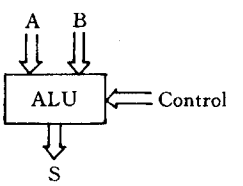


图 2-1 ALU 的符号

ALU 的符号一般画成图 2-1 那样。A 和 B 为两个二进制数，S 为其运算结果，Control 为控制信号，如上面的控制线端 *SUB*。

2.2 触发器(Trigger)

触发器是计算机的记忆装置的基本单元，也可说是记忆细胞。触发器可以组成寄存器，寄存器又可以组成存储器。寄存器和存储器统称为计算机的记忆装置。

微型计算机所用触发器一般用晶体管元件而不用磁性元件。这是因为晶体管元件可以制成大规模的集成电路，体积可以更小些。从晶体管电路基础中，我们已经知道触发器可以由两个晶体管组成的对称电路来构成，我们也知道触发电路中有所谓单稳态触发电路和双稳态触发电路，这里不打算重复这些电路的原理图和工作特点了。

下面简要地介绍一下 RS 触发器、D 触发器和 JK 触发器，因为这些类型的触发器是计算