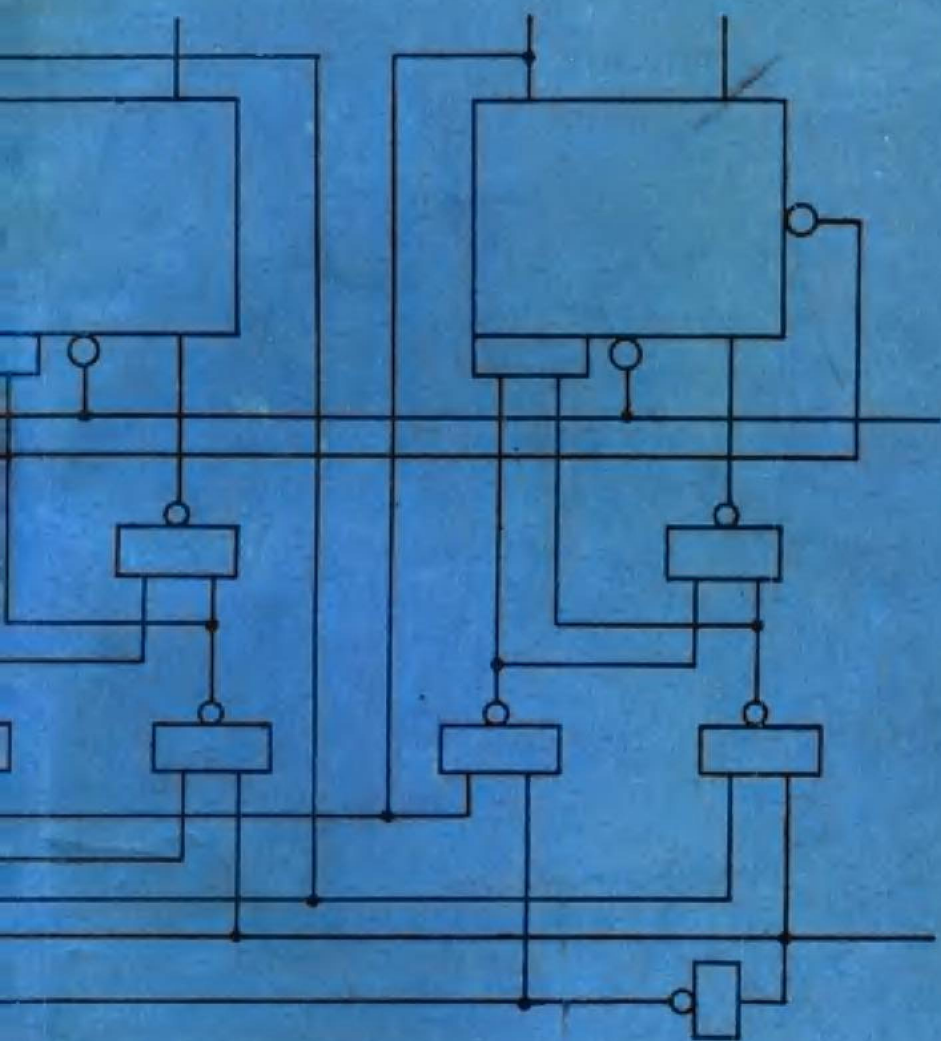




1207-X-AZD
SHIXOZGJ-CHAZG

王永祥 编著

陕西科学技术出版社

数控机床逻辑线路 上

数控机床逻辑线路

(上、下册)

王永祥 编著

*

陕西科学技术出版社出版

(西安北大街131号)

陕西省新华书店发行 交大印刷厂印刷

开本 787×1092 1/16 印张 41.75 插页, 字数: 1,026,000

印数 1—2,200

1981年7月第1版 1981年7月第1次印刷

统一书号: 15202.34 定价: 5.30元

编 者 的 话

《数控机床逻辑线路》一书的编写，既希望初学数控技术的人能够入门，并通过本书的学习掌握这门知识，又希望从事数控机床、数字电路设计与维护的工程技术人員能从本书得到补益。

逻辑代数是数字电路的理论基础。实践表明，为了掌握数控技术，学好逻辑代数是必要的。本书深入地讨论了这部分内容，介绍了“补割”规则，并用这一规则总结了一些有用的结论，设计出一些最简逻辑线路。

数控装置一般都由基本逻辑线路组成，知道基本逻辑线路的来龙去脉是重要的。时序线路设计一章介绍了其他文献尚未见到的“多周期计数器修正设计法”，读者可从中得到一些启示。

本书的后半部分对数控机床的工作原理做了详细的介绍。对“时差判据”插补方式做了探讨。愿与同志讨论。

在编写本书过程中，陕西省五机局新技术推广站给予大力支持，曾组织专人校稿制图。西安交通大学工企教研室和计算机控制教研室，东风仪表厂，庆华电气厂，华山机械厂，光辉机械厂，昆崙机械厂，秦川机械厂等单位给予不少帮助。王浣尘、薛钧义、申忠如、林魁明等同志对本书编写给予关怀和支持。张子炎、李秦庆、李新三位同志校阅全稿，在此一并致谢。

由于水平所限，错误之处请读者批评。

目 录

第一章 二进制数与码

第一节	基数与计数制	1
第二节	二进制数向十进制数的转换	4
第三节	十进制数向二进制数的转换	5
第四节	二进制数的计算	8
第五节	“二—十”进制数	11
第六节	机器中数的表示——码	14
第七节	补码加法、减法运算	19
第八节	补码运算定理的证明	21

第二章 逻辑代数与门电路

第一节	逻辑代数与初等代数的简单比较	23
第二节	非函数	30
第三节	与函数	33
第四节	或函数	39
第五节	复合逻辑函数	44
第六节	逻辑代数基本定理	47
第七节	唯一性证明	48
第八节	导出公式的证明(一)	50
第九节	导出公式的证明(二)	51
第十节	摩根定理及重要的有条件等价式	56
第十一节	重要规则	61
第十二节	函数的标准形	67
第十三节	函数的公式法化简	74
第十四节	函数的真值图化简	81
第十五节	晶体管——逻辑“与非”门电路(TTL 电路)	93

第三章 组合逻辑电路设计

第一节	组合逻辑电路设计的一般程式	101
第二节	全加器的设计	102
第三节	全加全减器的设计	111
第四节	判偶逻辑电路设计	113
第五节	译码器设计(一)	118
第六节	约束条件与函数化简	121

第七节	译码器设计(二)·····	127
第八节	循环码译码器·····	132
第九节	数字显示电路·····	138
第十节	控制带及译码器·····	143
第十一节	组合逻辑设计的简单问题·····	154

第四章 组合逻辑电路分析

第一节	直接写出函数表达式的方法·····	156
第二节	利用“关门”“开门”概念的读图方法·····	160
第三节	抑或函数和斥或函数的读图方法·····	162
第四节	单路输出译码器禁止项的逆推·····	163

第五章 脉冲电路与触发器

第一节	电容器的一般知识与微分、积分电路·····	171
第二节	振荡器·····	181
第三节	单稳态触发器的设计·····	186
第四节	基本 $R-S$ 触发器(双稳态触发器)的设计·····	191
第五节	三稳态乃至多稳态触发器的设计·····	194
第六节	同步触发器的设计·····	195
第七节	维持阻塞 D 触发器的设计·····	199
第八节	维持阻塞 T 触发器·····	203
第九节	主从 $J-K$ 触发器·····	205

第六章 时序逻辑线路设计

第一节	时序线路概述·····	211
第二节	寄存器的设计·····	215
第三节	计数器设计概述·····	223
第四节	同步计数器设计(一)——变迁表设计法·····	225
第五节	同步计数器设计(二)——循环码计数器的观察设计法·····	241
第六节	同步计数器设计(三)——多周期计数器的综合设计法·····	259
第七节	同步计数器设计(四)——多周期同步计数器的置数设计法·····	290
第八节	纯 2^n 进制最简异步计数器的设计·····	298
第九节	异步十进制计算器的设计·····	312
第十节	特殊计数流程与广义状态函数·····	330
第十一节	多周期异步计数器的修正设计法·····	336
第十二节	多周期异步计数器的置数设计法·····	348
第十三节	时标脉冲发生器的设计·····	352
第十四节	内插分频器的设计·····	355

第七章 基本时序线路分析·竞争

第一节	基本时序线路识别简介·····	361
-----	-----------------	-----

第二节	同步计数器分析·····	363
第三节	异步计数器分析·····	368
第四节	脉冲竞争概述·····	371
第五节	组合逻辑电路竞争的判别与消除(一)·····	374
第六节	组合逻辑电路竞争的判别与消除(二)·····	377
第七节	环形振荡器的过渡过程·····	384
第八节	$R-S$ 触发器的竞争与消除·····	388
第九节	同步 $R-S$ 触发器的竞争与消除·····	390
第十节	时序线路的竞争判别与消除·····	393

第八章 数控机床输入部分综述

第一节	输入代码·····	400
第二节	光电机自动输入方式·····	410
第三节	手置输入与刀具补偿输入·····	413
第四节	奇偶校验·····	415
第五节	输入寄存器·····	417
第六节	程序号显示及预选停机·····	417
第七节	P 寄存器·····	419
第八节	输入控制器所需时序信号·····	421

第九章 运算器

第一节	两数求和的串行运算原理·····	424
第二节	十翻二运算·····	426
第三节	变补器的设计·····	430
第四节	移位寄存器奇偶校验逻辑线路设计·····	434
第五节	运动学与插补方式·····	436
第六节	逐点比较法直线插补原理·····	440
第七节	逐点比较法圆弧插补原理·····	449
第八节	逐点比较法抛物线插补原理·····	455
第九节	逐点比较法插补运算中用到的基本逻辑部件的一般设计方法·····	461
第十节	脉冲速率乘法器插补原理(MIT 分配方式)·····	466
第十一节	数字积分法插补原理·····	470
第十二节	数字积分器直线插补原理·····	479
第十三节	数字积分器二次曲线插补原理·····	482
第十四节	数字积分器正弦曲线及指数曲线插补原理·····	495
第十五节	时差法插补原理·····	504
第十六节	进给速度控制·····	529

第十章 输出控制器

第一节	输出控制器概述·····	535
-----	--------------	-----

第二节	升降速线路的一般介绍	536
第三节	同步器的设计	538
第四节	升降速线路中的可逆计算器	545
第五节	T型解码网络	550
第六节	滤波器及变频振荡器	552
第七节	全部数字化的升降速线路	554
第八节	步进电机	559
第九节	步进电机相励磁顺序控制器	562
第十节	步进电机相励磁功率放大电路简介	564
第十一节	加补偿的开环控制系统	568
第十二节	数字闭环控制系统简介	570
第十三节	检测器概述	576
第十四节	旋转变压器及数控闭环系统	577
第十五节	感应同步器及数控闭环系统	582
第十六节	感应同步器定尺励磁鉴相式位移数字测量装置	598
第十七节	光栅检测系统	608
第十八节	磁尺检测系统	611

第十一章 CJSK-1 型车床简易数控机

第一节	概述	615
第二节	鼓轮程序输入机	619
第三节	主轴转速控制	621
第四节	进给速度控制	623
第五节	换刀控制	626
第六节	位置检测器——光电数码盘	627
第七节	比较器	633
第八节	超程保护及其他保护环节	636
第九节	自动程序转换控制电路	637
第十节	起停电路及操作过程	641

附 录

附图 1		643
附图 2		644
附录 1	2 的幂一览表	645
附录 2	逻辑代数基本公式	646
附录 3	二进制逻辑电路图图形符号(SJ 1223-77)	648
附录 4	几种插补曲线的误差计算	650

第一章 二进制数与码

数控装置与电子数字计算机进行两种演算：逻辑演算与数字演算。这两种演算是本书研究的主要内容。有关逻辑演算部分待后详述。这里首先介绍数字演算。

数字演算有两部分内容，其一是计数方法，其二是计算方法。数控装置与计算机主要采用“二进制”数计数与计算。什么是二进制数？它有什么特点？下面作一全面介绍。

第一节 基数与计数制

一、基数 N

十进制数的计数符号共有十个：0, 1, 2, 3, 4, 5, 6, 7, 8, 9。我们说十进制数的基数为十。基数的大小不同，表示计数符号的多少不同，基数为十时，称为十进制数。

类推：二进制的基数为二，计数符号有 0, 1 两个。

八进制的基数为八，计数符号有 0, 1, 2, 3, 4, 5, 6, 7 共八个。

N 进制的基数为 N ，计数符号共有 N 个。

可见，“基数”揭示了不同计数制的本质。为了区分各种计数制的数，在数的右下方加下标，下标的数为基数，如： $[1011]_2$ 就表示二进制数， $[a]_N$ 表示 N 进制数。

由于二进制数的计数符号只有 0 和 1，也就是说只要用两个截然不同的状态表示 0 和 1 就可以计数了。而十进制数的计数符号有十个，要用十个截然不同的状态才能表示十进制数。显然，用物理器件的十个稳定状态表示十进制数困难会很大。因此数控装置与计算机都用二进制数进行计数与计算。

二、“权”与基数的关系

十进制数的某些计数符号从左向右排成一行，譬如“11201.1”，这个数在不同位置上用到相同的计数符号“1”，但是所代表的数值大小不同。可见，某一符号代表的数值大小和该符号位置有关。一个十进制数，抽象地写成

$$a_n a_{n-1} \cdots a_2 a_1 a_0 a_{-1} a_{-2} \cdots a_{-m}$$

a_i 表示十进制数的第 i 位。注意，平时我们说的“个位数”叫做第 0 位数。 a_{i+1} 叫做 a_i 的“相邻高位”，而 a_{i-1} 对于 a_i 而言，叫做“相邻低位”。就是说，某一位数的左边一位叫做该位的相邻高位，某一位数的右面一位叫做该位的相邻低位。 a_i 可以是 0 到 9 中某一个符号。对“11201.1”这个十进制数，各位所代表的数值大小为：

第 4 位	第 3 位	第 2 位	第 1 位	第 0 位	第 (-1) 位
1	1	2	0	1	1
1×10^4	1×10^3	2×10^2	0×10^1	1×10^0	1×10^{-1}

即 $11201.1 = 1 \times 10^4 + 1 \times 10^3 + 2 \times 10^2 + 0 \times 10^1 + 1 \times 10^0 + 1 \times 10^{-1}$

写成竖式

$$\begin{array}{r}
 1\ 0\ 0\ 0\ 0 \\
 1\ 0\ 0\ 0 \\
 2\ 0\ 0 \\
 0\ 0 \\
 1 \\
 + \quad 0.1 \\
 \hline
 1\ 1\ 2\ 0\ 1.1
 \end{array}$$

我们把 10^4 叫做第 4 位的“权”， 10^0 叫做第 0 位的“权”， 10^{-1} 叫做第 (-1) 位的“权”。写成通式：第 n 位的“权”为 10^n ，第 $(-m)$ 位的“权”为 10^{-m} 。

考察十进制数的“权”，以指数形式出现，底为基数 10，指数恰为数的位置序号。可见，“权”既与位置有关，又与基数有关。就是说“权”是基数和位置的函数。

由基数为十的十进制数第 n 位之“权”是 10^n ，类推得到：

基数为二的二进制数第 n 位之“权”是 2^n

基数为八的八进制数第 n 位之“权”是 8^n

基数为 N 的 N 进制数第 n 位之“权”是 N^n

例如：二进制数 101101.1101 按权展开的和式是

$$\begin{array}{cccccccccccc}
 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\
 \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\
 1 \times 2^5 & + 0 \times 2^4 & + 1 \times 2^3 & + 1 \times 2^2 & + 0 \times 2^1 & + 1 \times 2^0 & + 1 \times 2^{-1} & + 1 \times 2^{-2} & + 0 \times 2^{-3} & + 1 \times 2^{-4}
 \end{array}$$

三、相邻数位之权的关系

十进制数第 n 位之权为 10^n ，第 $(n-1)$ 位之权为 10^{n-1} ，相邻两位权的比值为 10，即有

$$\frac{10^n}{10^{n-1}} = 10, \text{ 说明相邻两位权的比值为十进制数的基数十。}$$

类推：二进制数第 n 位之权 2^n 是相邻低位之权 2^{n-1} 的 2 倍，即， $\frac{2^n}{2^{n-1}} = 2$

八进制数第 n 位之权 8^n 是相邻低位之权 8^{n-1} 的 8 倍，即， $\frac{8^n}{8^{n-1}} = 8$

N 进制数第 n 位之权 N^n 是相邻低位之权 N^{n-1} 的 N 倍，即， $\frac{N^n}{N^{n-1}} = \frac{N^{n-1} \cdot N}{N^{n-1}} = N$

例如：二进制数 1111.1111 各位之权如下所示：

$$\begin{array}{ccccccc}
 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
 \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\
 8 & 4 & 2 & 1 & 0.5 & 0.25 & 0.125
 \end{array}$$

四、运算规则

1. 加法运算规则

试计算 $[183]_8 + [529]_8 = ?$

列成竖式

$$\begin{array}{r}
 1\ 8\ 3 \\
 + \quad 5\ 2\ 9 \\
 \hline
 7\ 1\ 2
 \end{array}$$

首先，个位相加 $3+9=12$ ，“本位计二，逢十进一”。

所谓“逢十进一”表示：十进制某位数计数到基数十，则向相邻高位进位。本位的十等于相邻高位的一。

随之，计算第1位： $8+2+1=11$ “本位计一，向上进一”，即“逢十进一”。

最后，计算第2位： $1+5+1=7$

得到 $[183]_+ + [529]_+ = [712]_+$

类推：二进制数加法运算，某位的各个数相加，计满基数二，则“逢二进一”。也就是本位数二等于相邻高位的一。

八进制数加法运算：某位的各个数相加，计满基数八，则“逢八进一”。也就是本位数的八等于相邻高位的一。

N 进制数加法运算：“逢 N 进一”。

例如，两二进制数求和：

$$[11011]_2 + [11101]_2 = [111000]_2$$

列成竖式

$$\begin{array}{r} 11011 \\ + 11101 \\ \hline 111000 \end{array}$$

2. 减法计算规则

试计算 $[401]_+ - [189]_+ = [212]_+$

列成竖式

$$\begin{array}{r} 401 \\ - 189 \\ \hline 212 \end{array}$$

首先，计算第0位：1减9不够减，向第1位“借一为十”变成 $(10+1)-9=11-9=2$ ，就是说相邻高位的一等于本位的十。

再算第1位：被减数为0，要减去相邻低位的借位1，再减去本位的8，不够减，向相邻高位“借一为十”得 $10-1-8=1$ 。

最后再算第2位： $4-1-1=2$ 。

求得 $[401]_+ - [189]_+ = [212]_+$

类推：二进制数本位相减，如不够减，向相邻高位“借一为二”。

八进制数相减，“借一为八”。

N 进制数相减，“借一为 N ”。

例如： $[10110]_2 - [10011]_2 = [11]_2$

列成竖式

$$\begin{array}{r} 10110 \\ - 10011 \\ \hline 00011 \end{array}$$

3. 扩大(缩小)基数倍的乘法(除法)运算规则

一个十进制数扩大(缩小)基数倍，即十倍，小数点向右(左)移一位。

例如： $[10.78]_+ \times [10]_+ = [107.8]_+$

$$[10.78]_+ \div [10]_+ = [1.078]_+$$

类推：一个二进制数扩大(缩小)二倍，小数点向右(左)移一位。

一个 N 进制数扩大(缩小) N 倍，小数点向右(左)移一位。

数 1 扩大二倍, 在二进制数中表为 $[10]_2$, 数 1 扩大四倍, 则为 $[100]_2$, 数 1 扩大八倍变为 $[1000]_2$ 。 $[10]_2$ 、 $[100]_2$ 、 $[1000]_2$ 分别为十进制数的 2、4、8。

十进制数的 24 在二进制中表为 $[11000]_2$, 24 缩小八倍为 3, $[11000]_2$ 缩小八倍小数点向左移三位, 变为 $[11]_2$, 这正是十进制数 3 的二进制表示法。

各种计数制列表 1-1。

表 1-1 各种计数制的特点和运算规则

各种计数制 计数特点及运算规则		十进制数	二进制数	八进制数	N 进制数
1	基数 N	10	2	8	N
2	第 n 位之权 N^n	10^n	2^n	8^n	N^n
3	相邻高位与本位权之比 $N^n/N^{n-1}=N$	10	2	8	N
4	加法运算, 本位相加, 计满基数, 则逢 N 进一	逢 10 进 1	逢 2 进 1	逢 8 进 1	逢 N 进 1
5	减法运算, 本位相减, 向相邻高位借位, 借一为 N	借 1 为 10	借 1 为 2	借 1 为 8	借 1 为 N
6	小数点向右移动一位扩大 N 倍	扩大 10 倍	扩大 2 倍	扩大 8 倍	扩大 N 倍
7	小数点向左移动一位缩小 N 倍	缩小 10 倍	缩小 2 倍	缩小 8 倍	缩小 N 倍

由表可见: 不同的基数产生出各种不同的计数制。只要知道了基数 N , 就可推知其计数规律和运算规则。从我们熟悉的十进制计数规律出发, 不难推出其他计数制的规律, 并用于各种数制的计算。

第二节 二进制数向十进制数的转换

由二进制数的计数符号 0 和 1 以及小数点“.”可构成一切非负的二进制数, 如 1011.101。

现在我们要问: “1011.101”的值是多大? 这个问题实质是二进制数向十进制数的转换问题。

两种不同数制的数的相互转换基于一个基本原则: 等值。也就是说, 不能因为数制不同使得某一数改变其值的大小。

我们对十进制数是非常熟悉的。其他计数制的基数和权都用十进制数予以表示。一个二进制数 1011.101 只要按权展开就会得到与之等值的十进制数。具体做法是

$$\begin{aligned}
[1011.101]_{-} &= 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} \\
&= 8 + 0 + 2 + 1 + \frac{1}{2} + 0 + \frac{1}{8} \\
&= [11.625]_{+}
\end{aligned}$$

二进制数的各位之权都是 2 的整数幂，为了快速准确地进行换算，熟悉这些“权”的值极有必要，尤其是

$$\begin{array}{cccc}
2^0=1 & 2^1=2 & 2^2=4 & 2^3=8 \\
2^4=16 & 2^5=32 & 2^6=64 & 2^{10}=1024
\end{array}$$

更有记住的必要。2 的其他方幂读者可查阅附录 1。

下面再举一例：

例1 试求 $[1111111111]_{-}=[?]_{+}$

解1 各位按权展开

$$\begin{aligned}
& \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \\
&= 2^9 + 2^8 + 2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 \\
&= 512 + 256 + 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 \\
&= [1023]_{+}
\end{aligned}$$

$$\begin{aligned}
\text{解2} \quad [1111111111]_{-} &= [1111111111]_{-} + 1 - 1 \\
&= [1000000000]_{-} - 1 \\
&= 2^{10} - 1 = 1024 - 1 = 1023
\end{aligned}$$

其他计数制的数向十进制数转换也是本着等值变换的原则按权展开进行计算的。

例2 $[7065]_{\text{八}}=[?]_{+}$

$$\begin{aligned}
\text{解} \quad [7065]_{\text{八}} &= 7 \times 8^3 + 0 \times 8^2 + 6 \times 8^1 + 5 \times 8^0 \\
&= 7 \times 512 + 0 + 6 \times 8 + 5 \times 1 \\
&= [3637]_{+}
\end{aligned}$$

例3 $[3401]_{\text{五}}=[?]_{+}$

$$\begin{aligned}
\text{解} \quad [3401]_{\text{五}} &= 3 \times 5^3 + 4 \times 5^2 + 0 \times 5^1 + 1 \times 5^0 \\
&= [476]_{+}
\end{aligned}$$

复 习 题

1. $[11101.11011]_{-}=[?]_{+}$
2. $[11111.11111]_{-}=[?]_{+}$
3. $[653]_{\text{八}}=[?]_{+}=[?]_{-}$
4. $[3024]_{\text{八}}=[?]_{+}=[?]_{-}$

第三节 十进制数向二进制数的转换

已知一个十进制数，如何换算成与它等值的二进制数？分两种情况予以介绍。

一、十进制整数向二进制整数的转换

以具体数字转换为例说明如下：

设有十进制数 $[57]_{+}$ ，待求的二进制数为 $[a_n a_{n-1} \cdots a_1 a_0]_{-}$ 我们把待求的二进制数按权

展开为 $a_n 2^n + a_{n-1} 2^{n-1} + a_{n-2} 2^{n-2} + \dots + a_1 2^1 + a_0 2^0$, 写成等式

$$57 = a_n 2^n + a_{n-1} 2^{n-1} + a_{n-2} 2^{n-2} + \dots + a_1 2^1 + a_0 2^0$$

由上式看出, 各个未知数 a_i 及未知数的个数 $(n+1)$ 都是不知道的, 由一个方程式求解多个未知数是不可能的。已知的约束条件之一是: 待求的二进制数为“整数”, 如果两边用二进制数的基数相除, 如第一节所述, 右边的二进制数的小数点将向左移一位, 从而把 a_0 从二进制整数中分离出来。利用两数相等时, 整数部分和小数部分分别对应相等的关系, 即可求出 a_0 , 具体计算步骤如下:

$$57 = 2(a_n 2^{n-1} + a_{n-1} 2^{n-2} + \dots + a_1 2^0) + a_0$$

等式两边都用 2 (二进制数的基数) 除

$$\frac{57}{2} + \frac{1}{2} = (a_n 2^{n-1} + a_{n-1} 2^{n-2} + \dots + a_1) + \frac{a_0}{2}$$

即

$$28 + \frac{1}{2} = (a_n \cdot 2^{n-1} + a_{n-1} 2^{n-2} + \dots + a_1) + \frac{a_0}{2}$$

由

$$\begin{cases} 28 = a_n 2^{n-1} + a_{n-1} 2^{n-2} + \dots + a_1 \\ \frac{1}{2} = \frac{a_0}{2} \end{cases}$$

(两数相等, 则整数部分和小数部分分别对应相等)。

求得 $a_0 = 1$, 而 a_0 正是用 2 去除 57 所得的第一个余数。

对等式

$$28 = 2(a_n 2^{n-2} + a_{n-1} 2^{n-3} + \dots + a_2) + a_1$$

两边再除以 2, 把 a_1 分离出来

$$\frac{28}{2} = 14 + \frac{0}{2} = (a_n 2^{n-2} + a_{n-1} 2^{n-3} + \dots + a_2) + \frac{a_1}{2}$$

由

$$\begin{cases} 14 = a_n 2^{n-2} + a_{n-1} 2^{n-3} + \dots + a_2 \\ \frac{0}{2} = \frac{a_1}{2} \end{cases}$$

求得

$$a_1 = 0$$

以此类推, 有如下转换方法:

把十进制数作为被除数, 以二进制数基数 2 为除数, 作连除运算, 直到最后商数为 0, 余数逆序列便是待求的二进制数。见竖式操作:

十进制数		余数
÷ 2	57	1 a_0
÷ 2	28	0 a_1
÷ 2	14	0 a_2
÷ 2	7	1 a_3
÷ 2	3	1 a_4
÷ 2	1	1 a_5
	0	

↑
余数逆序

求得

$$a_5 a_4 a_3 a_2 a_1 a_0 = 111001$$

得到

$$[57]_{10} = [111001]_2$$

上面“连除求余”的方法也能类推到其他计数制数的转换，如 $[385]_{10} = [?]_8$ ，列算式如下：

		余数	
÷8	3 8 5	a_0 1	余数 逆序 ↑
÷8	4 8	a_1 0	
÷8	6	a_2 6	
	0		

求得

$$[385]_{10} = [601]_8$$

二、十进制小数向二进制小数的转换

仍以实际例子加以说明。

已知 $[0.8125]_{10}$ ，求与它等值的二进制数。

把待求的二进制小数按权展开为

$$a_{-1}2^{-1} + a_{-2}2^{-2} + \dots + a_{-m}2^{-m}$$

一个二进制数扩大两倍，即扩大“二进制数的基数倍”，小数点向右移一位，从而把 a_{-1} 分离出来。利用两数相等时整数部分和小数部分分别对应相等的关系，即可把 a_{-1} 求出。

对上式两边扩大2倍，得等式

$$1.625 = a_{-1} + (a_{-2}2^{-1} + \dots + a_{-m}2^{-m+1})$$

也即

$$1 + 0.625 = a_{-1} + (a_{-2}2^{-1} + \dots + a_{-m}2^{-m+1})$$

求得

$$\begin{cases} a_{-1} = 1 \\ 0.625 = a_{-2}2^{-1} + \dots + a_{-m}2^{-m+1} \end{cases}$$

对小数等式两边再乘2，可以把 a_{-2} 分离出来，以此类推，写成下面的连乘转换方法：把十进数作为被乘数，用二进制数的基数2连乘，得到的整数序列就是待求的二进制小数，见竖式操作：

×2	0.8 1 2 5	进 位	
×2	1.6 2 5 0	1	a_{-1}
×2	1.2 5 0 0	1	a_{-2}
×2	0.5 0 0 0	0	a_{-3}
	1.0 0 0 0	1	a_{-4}

整数
序列
↓

注意，在连乘时凡整数均被丢掉不参于连乘运算，求得

$$[0.8125]_{10} = [0.1101]_2$$

十进制小数向二进制转换，有时会无穷尽地进行下去，根据计算误差的要求，可以舍去后面的数位。

如果一个十进制数，既有整数又有小数，可以分成两步求出等值的二进制整数和二进制

小数。

$$\begin{aligned}\text{如: } [57.8125]_+ &= [111001]_- + [0.1101]_- \\ &= [111001.1101]_-\end{aligned}$$

复 习 题

1. $[216]_+ = [?]_-$
2. $[0.0385]_+ = [?]_-$
3. $[104.625]_+ = [?]_-$
- *4. $[618]_+ = [?]_-$
- *5. $[425]_+ = [?]_-$

* 号题选作

第四节 二进制数的计算

一、加法运算规则

我们知道：一位的十进制数加法运算如果不考虑交换律则有 100 条规则：

$$\begin{array}{l} 0+0=0, 0+1=1, 0+2=2, \dots, 0+9=9 \\ 1+0=1, 1+1=2, 1+2=3, \dots, 1+9=1*0 \\ \vdots \\ 9+0=9, 9+1=1*0, 9+2=1*1, \dots, 9+9=1*8 \end{array} \quad \begin{array}{c} 0 \\ \downarrow \\ \text{进位} \quad \text{本位} \end{array}$$

多位数相加时，按以上 100 条规则实行本位相加，计满十则向相邻高位进一。

一位的二进制数加法运算若不考虑交换律只有 4 条规则：

$$\begin{array}{l} 0+0=0, 0+1=1, 1+0=1, 1+1=1*0 \\ \downarrow \quad \downarrow \\ \text{进位} \quad \text{本位} \end{array}$$

多位的二进制数相加，按“本位相加，逢二进一”的原则进行。

例1 $[10110]_- + [11011]_- = [110001]_-$

这由竖式

$$\begin{array}{r} 10110 \\ + 11011 \\ \hline 110001 \end{array}$$

得到。

二、减法运算规则

一位的十进制数减法运算有 100 条规则，多位的十进制数相减按“本位相减，借一有十”的原则进行。

一位的二进制数减法运算有 4 条规则：

$$\begin{array}{l} 0-0=0, 0-1=1*1, 1-0=1, 1-1=0 \\ \downarrow \quad \downarrow \\ \text{借位} \quad \text{本位} \end{array}$$

多位的二进制数相减也按“本位相减，借一有二”的原则进行。

例2 $[1100]_- - [0111]_- = [0101]_-$

这由竖式

$$\begin{array}{r} \dot{1} \dot{1} \dot{0} 0 \\ - 0 1 1 1 \\ \hline 0 1 0 1 \end{array}$$

得到。

三、乘法运算规则

一位的十进制数乘法运算共有 100 条规则：

$$0 \times 0 = 0, 0 \times 1 = 0, 0 \times 2 = 0, \dots, 0 \times 9 = 0$$

$$1 \times 0 = 0, 1 \times 1 = 1, 1 \times 2 = 2, \dots, 1 \times 9 = 9$$

.....

$$9 \times 0 = 0, 9 \times 1 = 9, 9 \times 2 = 18, \dots, 9 \times 9 = 81$$

多位的十进制数乘法运算分解为三种操作：移位、相乘、相加。

例3 $93 \times 25 = 93 \times 20 + 93 \times 5 = 1860 + 465 = 2325$

这里，2 后续 0 可转换为 93 后续 0，列成竖式看得更清楚

$$\begin{array}{r} 93 \\ \times 25 \\ \hline 465 \\ + 1860 \\ \hline 2325 \end{array}$$

一位的二进制数乘法运算只有 4 条规则：

$$0 \times 0 = 0, 0 \times 1 = 0, 1 \times 0 = 0, 1 \times 1 = 1$$

多位的二进制数乘法运算可分解为两种操作：移位、相加。乘法运算被加法运算所代替。

例4 $[1011]_2 \times [101]_2 = [110111]_2$

这由竖式

$$\begin{array}{r} 1011 \\ \times 101 \\ \hline 1011 \\ 0000 \\ + 1011 \\ \hline 110111 \end{array}$$

得到。

四、除法运算规则

除法是乘法的逆运算。十进制数除法运算可分解为四种操作：比较、乘法、移位、相减。

二进制数除法运算只用三种操作“比较、移位、相减”即能实现，省去了乘法运算。

例5 $[110110]_2 \div [1001]_2 = [110]_2$

列成竖式

$$\begin{array}{r}
 110 \\
 1001 \overline{) 110110} \\
 \underline{- 1001} \\
 1001 \\
 \underline{- 1001} \\
 0
 \end{array}$$

得到上述结果。

五、交换律

十进制数加法、乘法运算满足交换律：

$$[a]_{10} + [b]_{10} = [b]_{10} + [a]_{10}$$

$$[a]_{10} \times [b]_{10} = [b]_{10} \times [a]_{10}$$

二进制数加、乘运算亦满足交换律：

$$[a]_2 + [b]_2 = [b]_2 + [a]_2$$

$$[a]_2 \times [b]_2 = [b]_2 \times [a]_2$$

六、结合律

十进制数加、乘运算满足结合律，二进制数亦是：

$$[a]_2 + [b]_2 + [c]_2 = \{[a]_2 + [b]_2\} + [c]_2 = [a]_2 + \{[b]_2 + [c]_2\}$$

$$[a]_2 \times [b]_2 \times [c]_2 = \{[a]_2 \times [b]_2\} \times [c]_2 = [a]_2 \times \{[b]_2 \times [c]_2\}$$

七、分配律

二进制数加(减)、乘(除)联合运算时满足分配律：

$$\{[a]_2 + [b]_2\} \times [c]_2 = [a]_2 \times [c]_2 + [b]_2 \times [c]_2$$

$$\{[a]_2 + [b]_2\} \div [c]_2 = [a]_2 \div [c]_2 + [b]_2 \div [c]_2 \quad (c \neq 0)$$

$$\{[a]_2 - [b]_2\} \times [c]_2 = [a]_2 \times [c]_2 - [b]_2 \times [c]_2$$

$$\{[a]_2 - [b]_2\} \div [c]_2 = [a]_2 \div [c]_2 - [b]_2 \div [c]_2 \quad (c \neq 0)$$

如果把 $[a]_2 - [b]_2$ 转换为 $[a]_2 + [-b]_2$ ，把 $[a]_2 \div [b]_2$ ($b \neq 0$) 转换为 $[a]_2 \times \left[\frac{1}{b}\right]_2$ ，

则可实施交换律、结合律。

由本节内容不难看出，使用二进制数进行数值演算规则少，便于运算。这是数控装置和计算机使用二进制数的又一重要原因。

复 习 题

1. $[1101001]_2 + [111000111]_2 = [?]_2 = [?]_{10}$

2. $[100000]_2 - [11110]_2 = [?]_2 = [?]_{10}$

3. 列一个竖式进行计算：

a) $[1111]_2 + [1010]_2 + [111011]_2 = [?]_2$

b) $[10000000]_2 - [111100]_2 - [110111]_2 = [?]_2$

4. $[1101]_2 \times [1011]_2 = [?]_2 = [?]_{10}$

5. $[1001111]_2 \div [111]_2 = [?]_2 = [?]_{10}$

*6. $[104]_{10} + [776]_{10} = [?]_{10} = [?]_{16}$

*7. $[7003]_{10} - [4777]_{10} = [?]_{10} = [?]_{16} = [?]_{17}$