

微 计 算 机 引 论

大 连 工 学 院

中 譯 本 序

本书是美国 Adam Osborne and Associates 公司 1975 年出版的 An Introduction to Microcomputer 一书的中译本。全书详尽而系统地介绍了微型计算机的基础知识包括数字计算机的理论基础、微型计算机各部分的构造、原理、工作特点等。面向应用不侧重工艺，原理与典型机种介绍并重（系统介绍了国外流行的几种微计算机），是一本有价值的参考书，鉴于目前国内还较少系统讲述微计算机的书，特全文译出，供科研、生、使用单位与高等学校的同志参考。

本书由王耀奎、桂世和、李克春三同志译校。

译文中错漏之处在所难免，欢迎读者批评指正。

大连工学院无线电系

系统工程研究室

1979 年 3 月

目 录

第 一 章 什么是微计算机	(1)
计算机的发展	(2)
微计算机的起源	(3)
关于本书	(5)
第 二 章 若干基本概念	(6)
数 制	(6)
十进制数	(6)
二进制数	(6)
数制间的转换	(7)
其它数制	(9)
二进制算术	(10)
二进制加法	(10)
二进制减法	(10)
二进制乘法	(12)
二进制除法	(12)
布尔代数和计算机逻辑	(13)
“或”运算	(13)
“与”运算	(13)
“异或”运算	(14)
“非”运算	(14)
组合逻辑运算	(15)
摩根定理	(15)
第 三 章 微计算机构造	(17)
存储器组成	(17)
存储器字	(18)
字节	(19)
存储器地址	(19)
存储字内容的解释	(24)
独立的纯二进制数据	(24)
译出的二进制数据	(25)
字符码	(33)
指令码	(34)
第 四 章 微计算机中央处理单元	(38)
CPU 寄存器	(38)

CPU 寄存器的使用	(39)
算术逻辑单元	(45)
控制单元	(45)
状态标志	(46)
指令的执行	(49)
指令定时	(49)
指令周期	(50)
一条指令要做的事情	(55)
微程序设计和控制单元	(60)
微程序设计和微计算机指令系统	(63)
微程序设计的经济性	(72)
第 五 章 CPU外部逻辑	(73)
程序和数据存储器	(73)
只读存储器	(73)
读写存储器	(76)
微计算机系统外部数据传输 (输入/输出)	(79)
程序控制 I/O	(79)
中断 I/O	(82)
微计算机对中断的响应	(86)
中断设备选择码	(90)
中断优先级	(90)
直接存储器存取 (DMA)	(96)
周期挪用 DMA	(98)
多路外部设备 DMA	(103)
同时 DMA	(107)
同时 DMA 与周期挪用 DMA 的比较	(109)
外部系统总线	(109)
串行输入/输出	(109)
串行到并行的转换	(110)
字符装配与分解	(111)
实时逻辑	(112)
第 六 章 微计算机程序设计	(114)
程序设计语言概念	(114)
源程序	(115)
目的程序	(116)
目的程序的建立	(116)
程序储存介质	(117)
汇编语言	(117)
汇编语言语法	(118)

汇编命令.....	(123)
存储器寻址.....	(125)
微计算机存储器寻址.....	(125)
隐式存储器寻址.....	(126)
直接存储器寻址.....	(126)
直接寻址与隐式寻址的比较.....	(127)
直接存储器寻址的种类.....	(127)
页面直接寻址.....	(130)
微计算机中的直接存储器寻址.....	(135)
自动加 1 和自动减 1	(140)
堆栈.....	(140)
存储器栈.....	(141)
级联堆栈.....	(142)
堆栈的使用.....	(142)
嵌套子程序和堆栈的应用.....	(144)
间接寻址.....	(145)
分页计算机的间接寻址.....	(145)
程序相对间接寻址.....	(147)
间接寻址——小型计算机与微型机的比较.....	(147)
变址寻址.....	(148)
微计算机变址寻址.....	(151)
一个指令系统.....	(153)
CPU 结构.....	(153)
状态标志.....	(155)
寻址方式.....	(155)
指令描述.....	(156)
输入/输出指令.....	(156)
存储器访问指令.....	(159)
辅助存储器访问(存储器访问操作)指令.....	(165)
装立即数指令, 转移和转移到子程序指令.....	(170)
立即数操作指令.....	(173)
条件转移指令.....	(175)
寄存器——寄存器传送指令.....	(180)
寄存器——寄存器操作指令.....	(182)
寄存器操作指令.....	(186)
堆栈指令.....	(194)
参数传递指令.....	(196)
中断指令.....	(198)
状态指令.....	(201)

停机指令.....	(202)
指令系统一览表.....	(202)
第 七 章 几种实际微计算机	(206)
对各种微计算机叙述的深度.....	(207)
微计算机研制系统.....	(208)
<i>Fairchild—F8</i>	(208)
<i>Fairchild 3850 CPU</i>	(208)
F8 可编程序 寄存器.....	(210)
F8 存储器寻址方式.....	(212)
F8 状态 标志.....	(214)
F8 CPU 引出 端和信号.....	(214)
<i>Fairchild 3851</i> 程序存储单元 (PSU)	(216)
<i>Fairchild 3852</i> 动态存储器接口 (DMI)	(217)
<i>Fairchild 3854</i> 直接存储器存取 (DMA)	(220)
<i>Fairchild 3853</i> 静态存储器接口 (SMI).....	(220)
F8 中断处 理简述.....	(221)
F8 指令系 统.....	(221)
基准程序.....	(221)
F8 指令 系统一览表.....	(223)
<i>National Semiconductor PACE SC/MP</i>	(225)
PACE 微计 算机系统.....	(225)
PACE 可编 程序寄存器.....	(228)
PACE 寻址 方式.....	(229)
PACE 状态和 控制标志.....	(232)
PACE CPU 引出 端和信号.....	(233)
PACE 中断 处理.....	(234)
PACE 直接存 储器存取.....	(235)
PACE 指令 系统.....	(235)
基准程序.....	(237)
PACE 微计 算机指令 系统一览表.....	(238)
SC/MP 微计 算机系统.....	(240)
SC/MP 可编 程序寄存器.....	(243)
SC/MP 寻址 方式.....	(243)
SC/MP 状态寄 存器.....	(244)
SC/MP 中断处 理.....	(244)
SC/MP 直接存 储器存取.....	(245)
SC/MP CPU 引出 端和 信号排列.....	(245)
SC/MP 指令 系统.....	(247)
基准程序.....	(248)

指令系统一览表	(249)
<i>Intel-8080</i>	(251)
8080 CPU	(254)
8080可编程序寄存器	(254)
8080寻址方式	(255)
8080状态寄存器	(255)
8080 CPU 引出端和信号	(256)
<i>Intel</i> 8224 时钟发生器与驱动器	(258)
<i>Intel</i> 8228 系统控制器	(259)
<i>Intel</i> 8255 可编程序外围接口 (PPI)	(259)
<i>Intel</i> 8251 I/O 通讯接口 (USART)	(261)
8080 中断处理	(262)
8080 直接存储器存取	(264)
8080 指令系统	(264)
基准程序	(267)
<i>Motorola M6800</i>	(268)
M6800 CPU	(270)
M6800 可编程序寄存器	(270)
M6800 存储器寻址方式	(271)
M6800 状态标志	(272)
M6800 CPU 引出端和信号	(272)
M6800 外围接口适配器 (PIA)	(274)
MC6850 异步通信接口适配器 (ACIA)	(275)
M6800 中断处理	(276)
M6800 直接存储器存取	(277)
M6800 指令系统	(278)
基准程序	(278)
M6800 指令系统一览表	(279)
<i>Rockwell PPS-8</i>	(282)
PPS-8 CPU	(282)
PPS-8 可编程序寄存器	(284)
PPS-8 存储器寻址方式	(285)
PPS-8 状态标志	(288)
PPS-8 CPU 引出端和信号	(289)
PPS-8 时钟信号发生器	(290)
PPS-8 只读存储器 (ROM)	(291)
PPS-8 随机存取存储器装置 (RAM)	(292)
PPS-8 通用输入/输出设备 (GPI/O)	(293)
PPS-8 并行数据控制器 (PDC)	(294)

PPS-8 直接存储器存取控制器 (DMAC)	(295)
PPS-8 串行数据控制器 (SDC)	(296)
PPS-8 中断处理	(297)
PPS-8 指令系统	(297)
基准程序	(299)
PPS-8 指令系统一览表	(300)
Signeties 2650	(304)
2650 CPU 逻辑	(304)
2650 可编程寄存器	(305)
2650 存储器寻址方式	(306)
2650 状态标志	(310)
2650 CPU 引出端和信号	(311)
存储器对 2650 微计算机的连接	(315)
I/O 设备对 2650 微计算机的连接	(315)
2650 中断处理	(316)
2650 直接存储器存取	(317)
2650 指令系统	(317)
2650 基准程序	(318)
2650 指令系统一览表	(319)
第八章 微计算机的选择	(322)
使用微计算机时所涉及的设计步骤	(323)
微计算机研制硬件	(325)
微计算机系统软件	(326)
一个经济性举例	(330)
展望	(331)

附 录 A

标准字符码	(333)
-------------	---------

第一章 什么是微计算机

微计算机，也叫微处理机，是一种逻辑装置。确切地说是一种做在单个芯片上的各种各样的逻辑装置；由于微计算机的出现，使得逻辑设计不再和以前一个样了。

图 1—1 绘出了一个微计算机。微计算机的逻辑是做在一个芯片上，安装在双列直插式插件 (DIP) 上的。我们把 DIP 看作是一个逻辑装置，与硅片不同它是个逻辑芯片。

顾名思义，微计算机也是一种数字计算机。事实上，微计算机与其它计算机是很相似的。从旧的评价计算机的方法——指令系统、寻址方式到运行速度来看，某些微计算机同其它计算机没有很大差别，所以区分这两种产品之间的不同，只好注意到它们表现出来的任何一点特点。

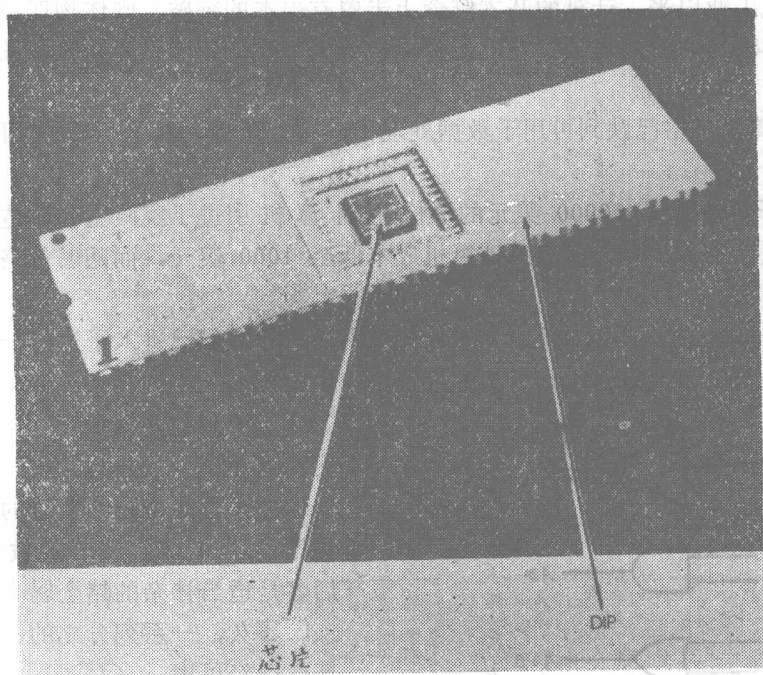


图 1—1. 微计算机芯片和 DIP

然而微计算机是一种新的和不同的产品。因为对微计算机用户来说，指令系统、寻址方式和运行速度等是属第二位的，所以评价微计算机不用旧的比较计算机的方法，而评价芯片上逻辑分布和微计算机装置的价格却是首要的；正是这些比较使得微计算机作为一种不同的产品而从其它类型计算机中分离开来。

本书的目的不仅描述微计算机是什么，此外也说明了评价它的方法为什么必须和早先用来比较计算机的方法不同。

本书假定读者还不大了解计算机的工作，因此从初步原理开始讲述计算机的概念。

微计算机和其它所有计算机有共同的祖先，但为了讲得透彻些，将从简短的计算机发展历史开始并溯源至微计算机的由来。

计 算 机 的 发 展

今天最小的微计算机和最大的主计算机有共同的祖先——UNIVAC—1，它是用电子管建成于 1950 年，有一个房间大，而运算能力小于多数现代微计算机。

UNIVAC—1 及以后的电子管计算机都是用于很有限的不计成本的场合，通常用来解决那些非用它不可的数学问题。

电子管计算机的逻辑不特别适应于科学应用；它的逻辑是由于构成它的双稳态逻辑装置（所有数字计算机的构造单元）自然而然的归结。

所有设计计算机的概念都可归根于查·贝其，他在 1833 年提出的概念在今天制造的所有数字计算机上面都可以找到，只有少数变化。第二和第三章中将叙述这些基本概念——即那些由二进制数构成计算逻辑的概念，而不论计算机怎么使用。

自有计算机工业以来，计算的基本概念上未曾有根本的突破。固体物理学的进展成了变革计算机工业的动力。新的电子工艺已使计算机价格下降得如此迅速而致不几年整个市场都被计算机席卷。

60 年代计算机价格已跌到可用于数据处理了，这意味着进入了计算机可用于一般任务的年代。

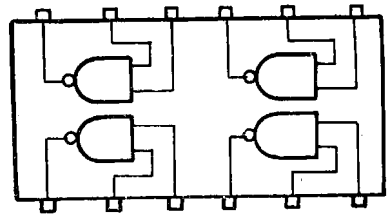
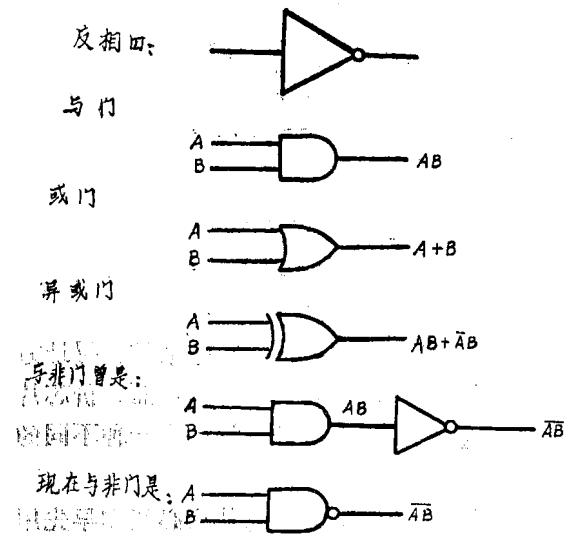
1965 年，PDP-8 以 50000 美元的价格把计算机引进了实验室和制造车间的生产线上；小型计算机诞生了。今天，小型计算机成本低至 1000 美元，而随着价格的降低它们的使用范围也扩展了。

但是，微计算机价格却在 5 元到 250 元范围内，——我们已进入了这样一个纪元：计算机可用来控制洗衣机、烘箱、或许是某种批量销售的消费产品中的一个部件。

所谈到的固体物理学的进展是什么呢？

电子管是个较为粗笨的器件，内部结构昂贵。50 年代后期它被晶体管取代。后者是适当掺杂的锗金属小片。

不久，一系列分立的、低成本的基本部件已经可用：（左图）



四个“与非”门已能做在一个芯片内（成本相当于，甚至低于一个单独的“与非”门），提

供了一种 4×2 输入端正与非缓冲器：（上页图）

后来由一个芯片上四个门变成十个、然后是一百、一千，今天相当于上万个门逻辑可装在单片硅芯上，而什么地方到头还无法预料甚至是望不见的。

在一个芯片上有一定数量的门叫做集成电路。如果一个芯片上有相当于 100 到 1000 个门逻辑认为是中规模集成电路 (MSI)，在某些不严格的定义中，在一个芯片上有超过 1000 个逻辑门就叫做大规模集成电路 (LSI)。

集成电路

集成电路有意义的方面在于单片的成本乃是其物理尺寸的函数——而不是在一个片上装置了多少逻辑的函数。因此，片子内逻辑越复杂，制成的计算机就越便宜。

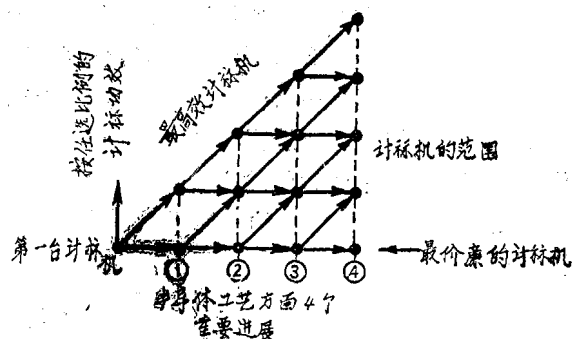
关于使计算机惊人地精缩方面有二点须予以搞清楚：

(1) 是不是整个计算机都能精缩？如果不是，那么哪一部分是保留原样的？

(2) 既然微计算机这么便宜，那为什么没有淘汰掉其它计算机？

首先，不能精缩整个计算机，只是电子部分可以精缩，人工接口即控制盘和开关被保留下来，它们是作为接收数据输入和产生人们可读形式结果的手段。一旦计算机变成单个逻辑装置，那么计算机的所有其它部分就没有必要了。

微计算机决不会抵消其它计算机，因为当计算机用于处理或解决科学问题时，有一个要使计算机更高效的苛刻的经济要求。于是随着每一个固体电子学工艺的重要进展，会出现两种新产品：昨天的计算机更小型些，和更高效的现代计算机：



随着时间的消逝，最便宜的计算机和最高效的计算机的能力之间有了明显的扩大。于是在 1965 年形成了第一个随便的划分，是在小计算机和大计算机之间。我们不想确定什么是小计算机和大计算机的不同。一个小计算机之所以是小计算机是因为产品的制造者叫它做小计算机。

第二个随便的划分是 1970 年在小计算机和微计算机之间形成。但这时产品之间的差别是易于确定的：

微计算机是预备作为一个大逻辑系统中的部件而出售的一个或很少几个逻辑装置。

相反，所有其它计算机是作为执行计算机程序的工具，每一程序瞬时地确定计算机系统的功能。

微计算机的起源

由于这是一本关于微计算机的书，让我们考察一下第一台实际微计算机的产生经过。

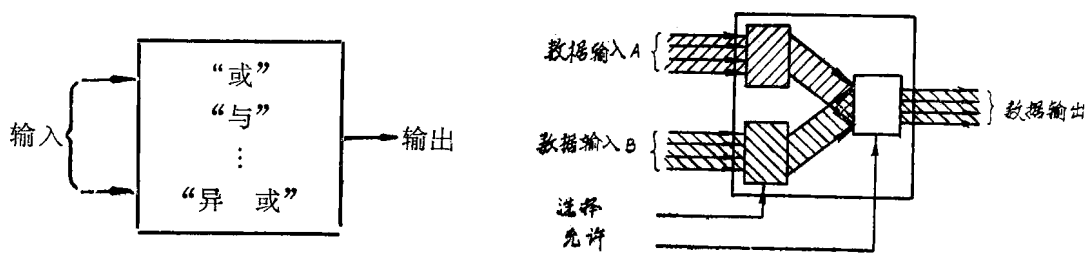
得克萨斯的圣·安东尼数据公司是个“智能终端”和小型计算机系统的制造厂商，1969 年该公司工程师设计了一个很初级的计算机，并与英特尔公司和得克萨斯仪器公司订立合同在单个逻辑芯片上实现其设计。英特尔公司制成了，但其产品的指令执行时间约比数据公司规定的慢十倍；因此数据公司拒绝购买，而用现行逻辑元件制造他们自己的产品。

结果英特尔公司留下了一个类似计算机的逻辑装置，其研制费用已被支付，面临生产出

售还是搁置起来的抉择。他们还是选择了出售，命名为 *Intel-8008*，微计算机就此诞生了。

尽管所设计的 *Intel-8008* 只用于完成简单数据处理（计算机的传统工作），却开创了一个崭新的商品市场。一个标志着脱离了“可程序”的逻辑装置。让我们来解释这个概念。

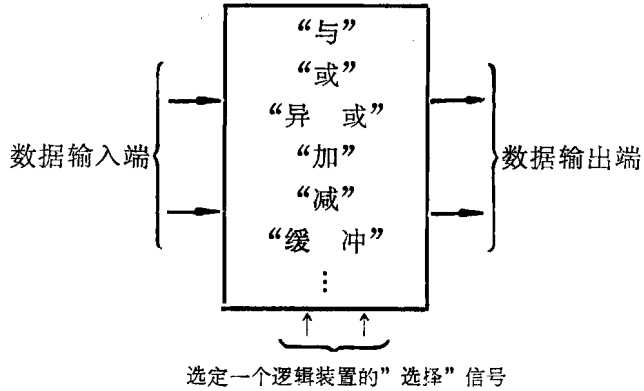
在某些逻辑部件的分类中，或许有上万个不同的逻辑单元。简单逻辑门可描述为：



输入数据按照转移函数规则传送到数据输出端。进一步研究一个更为有意义的逻辑装置，一个 4- 位 2- 输入端的缓冲多路转接器：

这个缓冲多路转接器中有二个有意义的概念：第一，数据以 4- 位为单元处理；第二，有二个非数据信号：“选择”和“启动”。“选择”决定哪一个输入数据转变为数据输出。“启动”决定它什么时候输出。

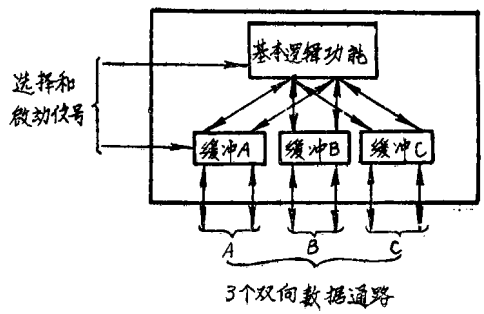
如果一个 *LSI* 片容纳相当于几千个逻辑门，如何简化在单个通用芯片上的逻辑分类呢？如图：



上面所描绘的通用片子有许多不必要的逻辑重复，列归在分类中的上万个片子之任一个，可以从少量的基本逻辑功能——“与”、“或”、“异或”、“加”、“减”——加上一些缓冲器、“选择”和“启动”来合成：

这个基本逻辑装置可以合成任意独立逻辑装置或任意独立逻辑装置序列。

这就是微计算机的概念。



关 于 本 书

本书目的是使读者对什么是微计算机以及它和其它计算机产品的不同有个透彻了解。由于本书认为读者以前未曾接触过计算机，所以基本概念包括得较详尽，并从基本概念着手来构成微计算机系统的基本部件。

书中着重指出微计算机与小计算机之间的不同。

本书不讨论制造逻辑芯片的各种工艺，因为对用户来说工艺本质通常是不太重要的。应用上可能只要几个关键参数诸如可供消耗的电功率或能容许的运行速度；当然用不同的工艺会影响功率消耗、运行速度和其它极限因素的，但这些因素的极限在那里，可以简单地看产品说明书选择合理的微计算机就是了。至于产品是用 *N-MOS* 工艺制造的还是 *C-MOS* 工艺制造的对于理解什么是微计算机或者怎样去使用它并无明显的难易差别。

第二章 一些基本概念

在微计算机和许多其它计算机之间没有什么根本的不同，其理由是所有计算机产品都基于一些基本计算概念——即在二进制数基础上的基本逻辑概念。

二进制数只有二个数码：0 或 1，而没有其它数码。

由于二进制数码可以用任意一个双稳态器件来表示，这就使得它十分有用。任何可以有“通”或“断”、

“高”或“低”状态的事物都可以用 0 来表示一个状态而 1 表示另一个状态。图 2—1 说明了双稳态器件的状态，而这就是你了解微计算机所需知道的全部物理性质所在。

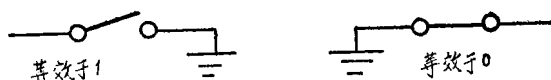


图 2—1. 用双稳态器件表示的二进制数

数 制

一台计数不能超过 1 的计算机不会是一台很有用的机器。幸而，二进制数能够用来表示任意数制的数字，正如同十进制能够表示超过 9 的任意数值一样。让我们研究实际上都有那些数制。

十 进 制 数

当一个十进制数具有一个以上数位时，你是否研究过每位数字实际上表示什么吗？一个二位数“11”表示 10 加上 1：

$$11 = 1 \times 10 + 1$$

类此，数 83 实际表示 8 个 10 加上 3：

$$83 = 8 \times 10 + 3$$

数 2347 实际表示 2 个 1000，加上 3 个 100，加上 4 个 10，再加上 7：

$$2347 = 2 \times 1000 + 3 \times 100 + 7$$

关于十进制没有什么独到或特别之处，事实上人们都有十个手指和十个脚趾，这无疑形成普遍使用十个数码的基础。但是，其它进制的数码也一样使用得很好。

二 进 制 数

因为十进制数字只有 0 到 9 的数码。十必须用“10”来表示。即用 1 乘上基数（这时为十）加上 0。用字母“B”表示基数为十，我们得到：

$$10 = 1 \times B + 0$$

在二进制中，“B”不代表十，而代表 2，所以在二进制中， $10 = \text{十进制的 } 2$ 。

$$10 = 1 \times 2 + 0$$

同样地，在二进制中的 11 代表十进制的 3，

$$11 = 1 \times 2 + 1$$

一般说，假设在任一计数制中，可用符号 d_i 、 d_j 、 d_k 等等符号表示它的数码，如果 B 表示基数，那么，任一个数都能用一个等式表示如下：

$$d_i d_j d_k d_l = d_i \times B^3 + d_j \times B^2 + d_k B + d_l$$

例如，假定有十进制数 ($B=10$) 和二进制数 ($B=2$)

$$2174_{10} = 2 \times 10^3 + 1 \times 10^2 + 7 \times 10^1 + 4$$

$$d_i d_j d_k d_l = d_i B^3 + d_j B^2 + d_k B + d_l$$

$$1011_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2 + 1$$

数制间的转换

从一种数制转换到另一种数制是容易的。下面我们只讨论十进制与二进制间的转换，先研究二进制到十进制转换

二进制数转换到十进制数

$$1011 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1$$

$$2^3 = 8 \text{ 和 } 2^2 = 4 \quad \text{所以,}$$

$$1011 = 1 \times 8 + 0 \times 4 + 1 \times 2 + 1$$

$$= 8 + 0 + 2 + 1$$

$$= 11 \text{ (十进制数)}$$

从十进制转换到二进制的一种简单的方法就是连续用除，记下每次的余数，这个余数就是用二进制表示的数。例如，将十进制数 11 转换二进制，其过程表示如下：

十进制数转换到二进制数

	商	余数
$\frac{11}{2}$	= 5	+ 1
$\frac{5}{2}$	= 2	+ 1
$\frac{2}{2}$	= 1	+ 0
$\frac{1}{2}$	= 0	+ 1

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$
 1 0 1 1

于是 $11_{10}=1011_2$

下标 10 和 2 用来区别是十进制还是二进制。

从二进制小数转换到十进制小数的展开式表示如下：

分数的转换

$$d_i d_j d_k d_l \cdots \text{等等} = (d_i \times B^{-1}) + (d_j \times B^{-2}) + (d_k \times B^{-3}) \\ + (d_l \times B^{-4}) \cdots \text{等等}。$$

d_i 、 d_j 、 d_k 等等表示数码，而“ B ”表示基数。例如，将 0.1011_2 转换到十进制数，其过程表示如下：

$$0.1011_2 = (1 \times 2^{-1}) + (0 \times 2^{-2}) + (1 \times 2^{-3}) + (1 \times 2^{-4})$$

其中：

$$2^{-1} = \frac{1}{2^1} = 0.5, \quad 2^{-2} = \frac{1}{2^2} = 0.25,$$

$$2^{-3} = \frac{1}{2^3} = 0.125, \quad 2^{-4} = \frac{1}{2^4} = 0.0625。$$

因此： $0.1011 = 0.5_{10} + 0 + 0.125_{10} + 0.0625_{10}$

$$= 0.6875_{10}$$

从十进制小数转换到二进制小数（例如将 0.6875_{10} 转换到等值的二进制数）应用如下的近似法：

$$\begin{array}{r} 0.6875 \\ \times \quad 2 \\ \hline 1.3750 \\ \downarrow \\ 1 \end{array}$$

$$\begin{array}{r} 0.3750 \\ \times \quad 2 \\ \hline 0.7500 \\ \downarrow \\ 0 \end{array}$$

$$\begin{array}{r} 0.7500 \\ \times \quad 2 \\ \hline 1.5000 \\ \downarrow \\ 1 \end{array}$$

$$\begin{array}{r} 0.5000 \\ \times \quad 2 \\ \hline 1.0000 \\ \downarrow \\ 1 \end{array}$$

遗憾的是，二进制小数转换到十进制小数不能总是精确的，正象分数不能精确的用十进制小数表示一样。因此，不能用 2^{-n} 项的和表示十进制小数，只能用二进制小数近似它。

假定有 0.42357_{10} ；等于这个数的二进制近似数产生如下：

$$\begin{array}{r} 0.42357 \\ \times \quad 2 \\ \hline 0.84714 \\ \downarrow \\ 0 \end{array}$$

$$\begin{array}{r} 0.84714 \\ \times \quad 2 \\ \hline 1.69428 \\ \downarrow \\ 1 \end{array}$$

$$\begin{array}{r} 0.69428 \\ \times \quad 2 \\ \hline 1.38856 \\ \downarrow \\ 1 \end{array}$$

$$\begin{array}{r} 0.38856 \\ \times \quad 2 \\ \hline 0.77712 \\ \downarrow \\ 0 \end{array}$$

$$\begin{array}{r} 0.77712 \\ \times \quad 2 \\ \hline 1.55424 \\ \downarrow \\ 1 \end{array}$$

答案是： $0.01101 \cdots_2$

让我们再往回转换比较一下：

$$0.01101_2 = 0 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3} + 0 \times 2^{-4} + 1 \times 2^{-5} \\ = 0 + 0.25 + 0.125 + 0 + 0.03125 \\ = 0.40625$$

$0.42357 - 0.40625$ 的差值等于 0.01732 ，这个误差是由于忽略了余数 0.55424 引起的。换句话说，忽略的余数 (0.55424) 乘以最小计算项 (0.03125) 得到总的误差：

$$0.55424 \times 0.03125 = 0.01732$$

其 他 数 制

因为二进制数表示起来很长，所以人们常常将它分成三位或四位一组。这时就构成基数为 8（八进制数）或基数为 16（十六进制数）的数字。表示在表 2—1 中：

计 数 制

表 2—1

十六进制	十 进 制	八 进 制	二 进 制
0	0	0	0000
1	1	1	0001
2	2	2	0010
3	3	3	0011
4	4	4	0100
5	5	5	0101
6	6	6	0110
7	7	7	0111
8	8	10	1000
9	9	11	1001
A	10	12	1010
B	11	13	1011
C	12	14	1100
D	13	15	1101
E	14	16	1110
F	15	17	1111

研究二进制数：

1 1 0 1 1 1 1 0 1 1 0 0

将上面二进制数分成三位一组，就转换到如下八进制形式的数，

八 进 制

110 111 101 100 = 6754₈

6 7 5 4

基数为 8（八进制）的数只包含 8 个数码：

0、1、2、3、4、5、6、7

十进制数的 8 等于八进制的 10。

将二进制数分成四位一组就转换成十六进制数：

十 六 进 制

1101 1110 1101 = DED₁₆

D E D