第一章　大容量存储设备

对一台计算机来讲，保存信息的能力是和它的微处理器一样重要。在处理过程中计算机必须能及时存取程序指令和数据。当计算机被关闭的时候，程序和数据必须存入计算机里，否则，一个操作者可能不得不用手工操作输入每一个指令和数据项目作为指令执行。正如你设想的那样，如果没有一定样式的存储机构，我们所知道的计算机是绝不可能存在的(如图 1.1)。

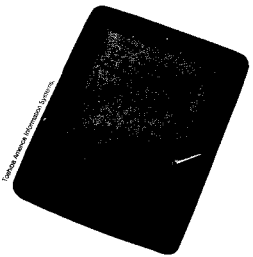

图 1.1　A Toshiba T100X Dynapad 笔记本式电脑

自从最早的计算机问世以来，对于更多、更块的存储器无止境的需求已经导致了几个高效存储器系列的出现，每一个这样的存储器设备都具有保存大量信息的能力。所有这些就是众所周知的大容量存储器设备。今天，随着对存储器要求很高的应用程序和操作系统例如 Windows 或者 OS/2 的出现，对于存储量大、存储速度快的存储器的要求比以往更急迫。在以后的许多年里，对于这种存储器的更大需求将很可能是今后计算机发展的驱动力。这本书就是一本有关研究以及维修驱动器和存储系统的专著。

现代大容量存储器系统存在着三项主要的技术：电子、磁性和光学。电子的大容量存储器使用半导体设备(主要是集成电路)。磁性大容量存储器利用了磁性材料、机械和电子微妙的相互关系来容纳存储的数据。光学的大容量存储器则将电子、机械与光学材料及其原理结

合起来存取巨大数量的信号。这三种大容量存储器技术各有着自身独特的优势，同样也具有其自身固有的局限性。它们中的每一种在计算机的应用上都占有重要的地位。通过对这部书课程的学习读者将详细地学到有关这些技术的知识。

然而，在我们转移视线进入大容量存储设备的细节之前，读者应该大体地知道一些有关计算机的知识。如果读者已经具备了大量有关计算机的基础知识，那么就可以轻松自如地略过这些部分。但是对于那些计算机基础技术懂得较少的读者，下面的材料能够给您提供一个对当代 IBM 兼容计算机了解的机会。并且使您对大容量存储设备在整个计算机中适用的那些地方，有一个较好的了解。

1.1 计算机基础

为了了解一部计算机，读者必须掌握两个重要的概念：数字逻辑和数字系统。这两个概念对于已有的每一个计算机系统的操作来说都绝对是最根本的，因为它定义了通过电子电路系统对信息进行解释和表达的方法。

让我们从头开始吧。

1.1.1 数字逻辑

现化数字逻辑的起源可以追溯到 1854 年。乔治·布尔(George Boole)发展了一个通过符号而非单词的替代来达到逻辑推论的新的思维方式；这种符号逻辑就是布尔逻辑或布尔代数。布尔逻辑的有趣特征是输入和输出状态只能被表达为正确或者是错误(是或者不是)。然而在十九世纪中叶，布尔的概念几乎没有任何实际的应用，在将近一百年之后它们才形成了电子逻辑设备的基础。

随着电子管的出现，用电子电路的形式来实现布尔逻辑成为可能，这种电路能自动解决布尔所设想的加、乘关系的问题。因为逻辑电路系统只处理两个状态(开/关或者 1/0)，它们被称为二进制逻辑电路系统。当电子管被诸如二极管和晶体管之类的半导体元件替代后，以布尔原则为基础的另外的逻辑函数(例如 NAND)，NOR，INVERTER，BUFFER，XOR，XNOR)出现了。每一个逻辑函数都用非常标准的电子电路系统来实现，因此他们被称为逻辑门。由于使用逻辑电路系统进行数学计算，所以逻辑也被称为数字逻辑。当分离的逻辑电路被最终组合为集成电路时，逻辑门概念出现，并一直被持续使用到现在。在第一章读者可以看见一幅画有八个主要逻辑门的图表。本书中，布尔逻辑、二进制逻辑和数字逻辑是相同的。

一个处理二进制逻辑的电子电路必须在逻辑状态和电信号之间有一个直接关系。因为逻辑电路执行操作是以存在于每个输入的电压信号为基础的，所以这个关系是很关键的。正像读者设想的，不正确的信号电压可能导致错误逻辑输出。二进制真(或者开着的)状态通常表示电压的存在，而二进制假(或者关闭的)状态则表示为没有电压。这一般称作常规或者高有效逻辑。然而在一些情况下，这个高有效约定被倒了过来。没有电压表示"开"，有电压的则表示"关"，这被称为低有效逻辑。低有效信号用在标号之上加负号(－)或者撇号(')来表

示。

在图 1.2 的第一个例子中，标号"Error"表示一个高有效信号。当存在逻辑"1"的信号电压时表明逻辑状态是真。当存在逻辑"0"的信号电压时表明逻辑状态是假。在第二个例子中，标号记为"Error"所表示的是一个低有效信号。在这里，逻辑 1 的信号电压被认为是假的。表示不存在误差条件。逻辑 0 信号电压表示真状态，表示存在误差条件。因为通常出现一个"0"逻辑信号比出现一个"1"逻辑信号更快，因而使用了低有效逻辑。也通常有效信号能够在硬件的水平上帮助提高逻辑系统的性能。

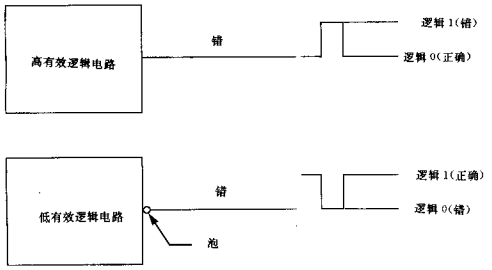

图 1.2 高有效和低有效逻辑的示例

当检测故障时，用户需要关心的仅仅是在测试期间认清高有效信号和低有效信号之间的差别。在一些简图和大幅的图解中，低有效信号通过电路输出上的小圈(称为包)来突出表示。

1.1.2 二进制数码

读者一定是很熟悉十进制数码系统的。这个数码系统是由从"0"到"9"的十个数码组成的。事实上通过使用位值的基本规则来组合数字，用户能够表示任何数量。二进制逻辑也与此相同。

二进制逻辑使用两个元素而非十个。把"真"看作逻辑"1"，把"假"看作逻辑"0"。这些二进制数字(或者二进制数的位)也能够组合起来去代表任何数量。十进制位存在从"0"到"9"十个数，而二进制位只有"0"和"1"两个。

在十进制数字系统中，一个单独的数字就能表达十个离散的数量(从 0 到 9)。当要表达的数量超过一个数位所能表达的范围时，数字将进位到下一个较高的位值上。因为在十进制系统中存在着十个数字所以十进制位值以十的乘方为基础中。它们是个位、十位、百位、千位

等等。举个例子，"35"这个数就是由十位上的一个"3"和个位上的一个"5"构成的。也就是说这个数可以分解为(3×10)+(5×1)。"256"这个数是由百位上的 2，十位上的 5 和个位上的 6 组成的。可以被分解为(2×100)+(5×10)+(6×1)。读者可能在低年级时就潜意识地做过这种分解。

因为二进制数字系统仅仅有两个数字而非十个数字，所以每一个位值是以"2"的乘方为基础的。这样就构造了如图 1.3 所示的位值系统。图 1.3 使用了"1"位、"2"位、"4"位、"8"位、"16"位、"32"位、"64"位等等。二进制数字 100110(发音为 one、zero、zero、one、one、zero)就由"32"位、"4"位、"2"位上的"1"组成。当将二进制数转换为等价的十进位数时，你得把相关连的每一个值为"1"的位值加起来。100110 这个数与十进制 38 是等价的。能够用二进制数表示出来的数的个数等于 2^n。在这里 n 可利用的位的个数。例如 4 个位能表示 2^4 或者 16 个数。8 个位能表示 2^8 或 256 个数等等。

2^7 128s	2^6 64s	2^5 32s	2^4 16s	2^3 8s	2^2 4s	2^1 2s	2^0 1s	
		1	0	0	1	1	0	二进制数
		1	×	32	=	32		
		0	×	16	=	0		
		0	×	8	=	0		
		1	×	4	=	1		
		1	×	2	=	2		
		0	×	1	=	0		
						38		等价的十进制数

100110(二进制)=38(十进制)

图 1.3 二进制数构成的例子

1.1.3 十进制数的二进制代码(BCD)

十进制数的二进制代码(BCD)是一种通用的用等价的二进制数代表十进制数字的技术。例如十进制数"25"在 BCD 码中被表示为 00100101(在标准二进制中，十进制数字"25"表示为二进制数字 11001)。因为每一个十进制的数字能被直接地用它的四位等价的二进制数字代替，所以 BCD 提供了从 BCD 到十进制的快速和方便的转换方法，反之亦然。在历史上人们已经发现可使用像姆指旋转开关和原数据等可变状态输入设备来驱动七段 LEDs。现今，BCD 的典型应用是在光盘系统中对数据进行编码。这些将在本书的后面介绍。

1.1.4 十六进制数

十进制系统使用十个数字、二进制系统用两个数字，相应的十六进制系统使用十六个数字。除了使用字符"0"到"9"被使用之外，还使用了代表着数值从"10"到"15"的字母 A、B、C、D、E 和 F。用来形成十六进制数的位值技术是与十进制或者二进制的位值形成方法相同的。但是现在每一个位值是十六的乘方而不是十或者二的乘方。图 1.4 说明了两个十六进制数的例子。在第一个例子中，十六进制的数字 32 是由"16"位上的 3 和"1"位上的 C(12)构成的。这就是说其等价的十进制数是(3×16)+(12×1)或者 60。第二个例子是由在第"16"位上的 F(15)和在第"1"位上的 F(15)构成的。十六进制的 FF 代表了十进制的 255。

```
16⁴          16³          16²          16¹
65,536s      4096s        256s         16s          1s

3C(十六进制)                          FF(十六进制)
 3  x        16   =    48              15  x 15  =    240
12  x         1   =    12              15  x  1  =     15
                      60  十进制                    255   十进制
3C = 0011 1100 二进制                 FF = 1111 1111 二进制
```

图 1.4　十六进制数构成的例子

非常有趣的是每一个十六进制字符恰好对应四个二进制的位。例如十六进制的数字 5 在二进制数字中表示为 0101。而十六进制中的 F 也就等于二进制的中的 1111。这种关系使得数字在十六进位与二进位之间的转换很容易。这也是十六进制记法在计算机领域被广泛接受的原因。一个有十六个数位的二进制地址能够仅仅用四个十六进制字符表达出来。

1.1.5　八进制数

八进制数是一种用八个数字的数字系统，表示数量的方式。字符从"0"到"7"被用来表示这八个数字。然而八进制的位值利用的是 8 的乘方而不是 2、10 或者 16 的乘方。图 1.5 显示一个典型的八进制数是怎样形成的。一旦一个实际的数从"0"数到"7"，接下来的便是从"10"到 17"、"20"到"27"等等。八进制数字 170 是由"64"位上的 1、"8"位上的 7、"1"位上的 0 构成的。所以它与十进制的 120 等价。

```
8⁴          8³          8²          8¹          8⁰
4096s       512s        64s         8s          1s

170(八进制)             1           7           0

1   ×    64   =    64
7   ×     8   =    56
0   ×     1   =     0
                   120  (十进制)
170(八进制) = 001 111 000 (二进制)
```

图 1.5　八进制数构成的例子

读者应该注意到，仅仅用三个二进制数字就能表示每一个八进制数字。八进制数字 1 等于二进制数字 001，八进制数字 4 与二进制数字 100 是相等的，而八进制数字 7 则等于二进制 111。这个关系也使在八进制和二进制之间的数码转换非常方便。像十六进制一样，八进制系统也是为了促进快速的前后转换而发展起来的。

表 1.1 阐述了十进制、十六进制、八进制、二进制数码之间的关系。

表 1.1　十进制、二进数、十六进制和八进制数码之间的关系

十进制	二进制	十六进制	八进制
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3

（续表）

十进制	二进制	十六进制	八进制
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	8	10
9	1001	9	11
10	1010	A	12
11	1011	B	13
12	1100	C	14
13	1101	D	15
14	1110	E	16
15	1111	F	17

1.2 基本计算机

从最广泛的意义上讲，一部数字计算机是使用布尔逻辑原则进行输入、储存、操作和输出二进制信号的电子设备，仅此而已。已经制造出的每一部计算机都可以从这个基本的概念中找到自己的基础所在。为了符合这样一个概括性的定义，一部计算机必须至少有如图 1.6 所讲到的四个主要功能要素。它们是：处理器、储存器、输入设备和输出设备。尽管一部现代全功能计算机具有许多特殊的功能，每一个功能都大体地能被划分为这四类要素中的一种。

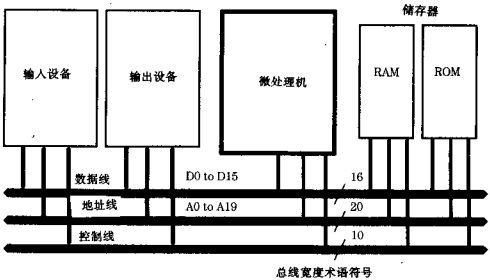

图 1.6 简单计算机系统图表

1.2.1 中央处理器

中央处理器(也叫CPU,或者微处理机)是每个现代计算机系统的"心脏"。正是CPU才使得计算机具有强大的处理能力。不可思议地是这个中央处理器只有三种操作功能:算术、逻辑和控制,仅此而已。算术操作是让CPU去进行加、减、乘、除的运算。逻辑操作是让CPU对不同数据进行比较和条件决定。控制操作是使CPU存取端口和计算机内部任何地方的数据,并且把信息从一个地方传递到另一个地方。甚至在复杂的CPU中有一百多独特、高效的指令,这些指令也能够被列入这三个种类。

CPU的最大的优势不在于它内部的设计。事实上,CPU使用比指定执行某一特殊功能的流水线集成电路慢得多的通用的内部设计。CPU执行按任何方式排列的指令的能力即活性弥补了它在执行速度上的不足。指引处理器操作的指令被称为程序。程序存放在CPU外的大容量储存器内(典型地在内存芯片库里)。CPU从储存器内的程序中存取它的指令和数据,然后执行每一个指令所要求的操作。这样,通过改变储存的指令和数据的类型同一CPU能执行一组完全不同的操作。尽管当今的计算机利用了一系列高效集成电路来高水平地控制驱动器、显示器、和键盘,实际使那些部分的工作得以配合起来的程序指令仍然得用CPU来执行。

1.2.2 存储器

CPU根据程序中包含的指令工作。然而CPU仅仅是一个处理工具,除了极少一些内部寄存器用来储存一些暂时信息外,微处理器无法保存它将要执行的程序。这个限制要求程序和任何与其有关的程序数据存储在CPU外,因而CPU只能简单地存取储存的程序并按其指令执行。计算机使用固态内存集成电路来保存当前的程序信息。暂时的随机存取存储器(RAM)存有当前的应用程序(例如文字处理或电子数据表),也容纳计算机的操作系统(例如DOS、Windows,或者os/2)。永久的只读存储器(ROM)存有计算机的上电自测(POST)程序和它的基本输入和输出系统(BIOS)。

在过去的几年里,内存芯片在速度和容量方面得到了迅速的发展。现在许多商业化的计算机装配有2Mb至4Mb的随机存取存储器作为标准配置。接下来的几年这些数字还可能翻倍。由于存储器变得很廉价,固态存储器集成电路已经成为大容量存储器家族的一个重要分支。在这章的后面读者将了解更多关于存储器集成电路的知识。第五章详细地介绍了内存设备、第六章将讨论独立固态存储器中存储器集成电路的强大的新应用。

1.2.3 输入和输出设备

尽管CPU加上存储器从技术上说已能组成能工作的计算机,然而,这样的计算机几乎没有任何实用价值,因为这样的设备不与外部世界进行交互。为了使计算机有用,计算机必须能够从外部世界得到信息或者传递信息给外部世界。在我们的意义中,CPU和存储器外部的任何组成部分或机构都可以称为"外部世界"。

输入设备包括了许多能向 CPU 和内存提供信息的电路和机电系统。这样，CPU 就能利用输入的信息来适应外部世界的任何变化。键盘是最普通、了解最多的输入设备之一。当读者压下一个键时，键盘电路系统产生一个数码值。CPU 识别到有键按下就中断当前的活动去获得这个键的数据。

甚至当计算机能读入和处理来自外部世界的数据时，若计算机不能将其处理结果以一种一致的有意义的形式提供给外部世界，它仍然不是完美的。我们可以认为输出设备是任何能接收计算机提供的数据的电路系统或机电设备。

计算机监视器和平面显示屏是两种典型的输出设备。CPU 将代表命令、正文和图形的数码输出给视频控制芯片，视频控制芯片储存和处理这些数码，并生成形成图形显示所需的信息。

许多其它的电路系统和机电设备兼具输入设备和输出设备的能力。串行通信端口是一种在程序控制下使数据流入和流出计算机的通用的输入或输出(I/O)电路。许多输入、输出设备属于这本书所提到的大容量存储系统。它们是软盘驱动器、硬盘驱动器、CD－ROM、磁带驱动器和固态存储卡。所有这些设备(除了 CD－ROM 之外)都能从 CPU 中接收输出数据，也能向 CPU 提供输入数据。

1.2.4 总线

当看到图 1.6 所示的简单计算机图表时，读者能够发现经典计算机的每一部分是通过使用叫做总线的三类主要的信号集相互联系在一起的。

值得指出的是，尽管总线可能只有一根线，它也能表示出所有在它上面一起通过的单独的信号集。总线里的确切行数由被使用的特定的微处理器和存储器排列决定。线的个数通常用在线上标注斜线来表示，或者用标号标示(例如从 D0 到 D7 表示一个 8 个二进制位的总线。)

在计算机里有三类主要的总线：地址总线、数据总线和控制总线。地址总线由 CPU 专门控制，被放置在地址总线上的二进制信息定义了计算机中能被 CPU 读入或写出的信息的精确位置。微处理器通常提供多于 20 条的地址线(从 A0 到 A19)，但是高级的地址技术可以使许多更新的 CPU 访问超过十亿的地址。

数据总线从地址总线指定的唯一位置中传送二进制信息或将二进制信息传送到该位置。这些二进制信息可能是 CPU 所需的指令、计算或比较的结果、子程序、转移的目标地址，在任一特定时刻，CPU 使用数据线输入或者输出数据。

控制总线传送用以引导系统操作的数字信号集，在不同的计算机模型机制造技术中，控制信号可能有相当大的区别。所以要想定义一个适用于所有计算机的总线是非常困难的。控制信号的数值和目的依赖于使用的 CPU 以及系统的总体复杂度。一种普通的的控制是 CPU 产生的读、写(R/W)信息。如果读、写指示为读，那么 CPU 输入地址总线所指定的特定位置中数据总线的内容。如果读/写指示为写，那么 CPU 就将数据输给由地址总线指定的地址。大多数电子大容量存储设备通过它的主总线与计算机接口，所以总线在这本书中是极其重要的。

1.3 母板和总线的结构

使计算机能工作所需的电路装配在固定于台式机里的印刷电路板上。在单一印制电路板上装上一完整的计算机是很简单的一件事。现在笔记本式电脑和膝上机就是这样被制造出来的。实际上，早期计算机如 Commodore 的 VIC－20，用的就是单板设计。

然而，当 IBM 在二十世纪八十年代早期设计他们的个人计算机时，他们通过在母板上只提供关键核心部件来节省费用。核心部件包括微处理器、数学微处理器、选择插座、基本内存(现在许多台式系统中超过 4Mb)BIOS 存储器(只读存储器)和提供时钟、系统控制和信号转换的支持电路。其它计算机功能例如显示控制器、驱动控制器和通讯电路必须以插入式扩展电路板型式附加给计算机。

1.3.1 有源底板

如图 1.7 所示典型的台式系统中 PC 机主系统板(也叫母板或底板)上装有核心元件。将主要的信号总线提供给一系列大的卡边连接器。每一个卡边连接器能连接一个标准尺寸的扩展板。当把新功能插入计算机以提高其性能时，它直接与主系统总线相连接。

这种方式被称为有源底板系统。术语"有源"是指有源的半导体元件在 PC 板的底板上工作。

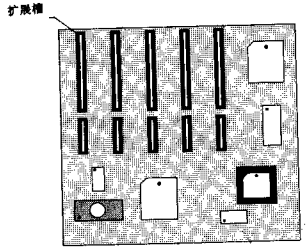

图 1.7 有源底板母底板

1.3.2 无源底板

有源底板系统的缺点是它们的体积太大。大面积的母插板区域(或者不动产)用来容纳有源元件和扩展板连接器。由于许多计算机系统努力减小计算机总外壳的尺寸(或足迹),因此有源底板常常是太大了。另一个问题在于有源底板的核心元件是固定的。每一件东西都可以改变,唯独核心元件例外的,为了克服有源底板系统所固有的限制,计算机设计者引进了无源底板体系。

无源底板不使用有源元件。这就导致了如图 1.8 所示的简单的,更小的 PC 电路板的出现。中央处理单元、基本输入/输出系统、核心内存、数学协处理器和配套电路系统都从母板中移去而被放置在自己的插入板上。中央处理单元板占据自己的扩展槽。读者只需简单地卸下旧的中央处理单元板换上新的中央处理单元板就能更新或者维修计算机的核心功能。其他的扩展板例如视频控制系统、驱动控制系统都是与有源底板系统完全相同的。

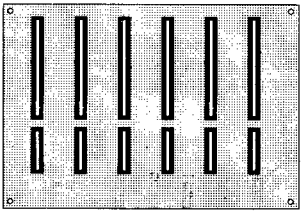

图 1.8 无源底板母板

用户应该熟悉 PC 总线。因为在排除大容量存储器故障的时候,用户可能将不得不处理扩展电路板及计算机底板。这本书给读者介绍三类流行的 IBM PC 总线。它们是 PX/XT 总线、PC/AT 总线和 PC 微通道总线。

1.3.3 PC/XT 总线

原始的 IBM PC/XT 的底板总线使用如图 1.9 所示的标有从 A1 到 A31 和 B1 到 B31 共 62 个引脚的卡边连接器。它有三个地线,五个电源线,20 个地址线、八个数据线、十个中断线和十六个控制信号。表 1.2 描述了每个引线的标号。

图 1.9 一个 IBM PC/XT 总线连接器的输出引线图表

表 1.2 PC/XT 总线扩展板引脚

B1	地	A1	I/O 通道检查
B2	复位驱动	A2	数据 7
B3	+5V 直流电压	A3	数据 6
B4	中断请求 2	A4	数据 5
B5	−5V 直流电压	A5	数据 4
B6	DMA 请求 2	A6	数据 3
B7	−12V 直流电压	A7	数据 2
B8	片选(只用于 XT)	A8	数据 1
B9	+12V 直流电压	A9	数据 0
B10	地	A10	I/O 通道就绪
B11	存储器写	A11	地址
B12	存储器读	A12	地址 19
B13	I/O 写	A13	地址 18
B14	I/O 读	A14	地址 17
B15	DMA 应答 3	A15	地址 16
B16	DMA 请求 3	A16	地址 15
B17	DMA 应答 1	A17	地址 14
B18	DMA 请求 1	A18	地址 13
B19	DMA 应答 0	A19	地址 12
B20	时钟	A20	地址 11
B21	中断请求 7	A21	地址 10
B22	中断请求 6	A22	地址 9
B23	中断请求 5	A23	地址 8
B24	中断请求 4	A24	地址 7
B25	中断请求 3	A25	地址 6
B26	中断请求 2	A26	地址 5
B27	计数终止	A27	地址 4
B28	地址锁存	A28	地址 3
B29	+12V 直流电压	B29	地址 2
B30	振荡器	A30	地址 1
B31	地	A31	地址 0

图表半导体振荡器信号(引线 B30)是一直接由晶体振荡器形成的为性能时钟。1431818 —MHz 时钟精确度是中央处理单元所需频率的三倍,是电视和监视器收到彩色信号时钟的

四倍。通过它，半导体振荡器不仅能驱动 CPU 而且也能驱动显示系统。任何需要与中央处理单元同步的扩展电路系统都能使用半导体振荡器信息。时钟信号(B20 脚)是由三等份振荡器信号得到的频率为 4.77－MHz 的矩形波。它直接用来驱动中央处理单元。它也能用来操作这个系统中的其它扩展电路板。

输入/输出通道检查(引线 A1)是附属于总线系统的内存和设备的检查信号。当信号是逻辑"1"时，奇偶校验是正确的，中央处理单元也就知道处理正在正常地进行，当奇偶校验发生错误时，信号就显示为"0"。这有效地加快了计算机的处理。复位驱动线(引线 32)上出现脉冲时，整个系统就重新初始化(也就是众所周知的热启动)。

当中央处理单元将地址放在它的地址总线(引线 A12－A31)上时，地址锁存选通信号(ALE，引线 B28)上就出现一个短形脉冲来指示这个地址是有效的。外部电路系统使用 ALE 来获取地址，与此同时，中央处理单元进行其它的工作。在存储器读数据期间中央处理单元激活存储器读命令线(引脚 B12)，将数据从特定的地址输入到数据线(引脚 A2 到 A9)上。相反地，在写操作期间，CPU 激活存储器写命令线(引脚 B11)这导致数据被输出到特定的地址。如果中央处理单元是对输入/输出口而不是存储器进行操作，那么中央处理单元就能通过激活输入/输出读出引线(引脚 B14)从输入/输出地址端口输入数据，或者通过激活输入/输出写命令(引脚 B13)将数据输给输入/输出地址端口。因为微处理器存取输入/输出数据的速度比输入\输出设备速度快，所以输入/输出通道就绪线(引线 A10)使得中央处理单元一直等到输入/输出电路赶上时间为止。

IBM 总线体系也允许在 DMA 控制下的数据传输。DMA 传输比通过微处理器传输快得多，但它必须控制地址和数据总线。地址锁存信号通知 DMA 控制器何时控制系统并进行数据传输，在说明了需转移的字节数后，控制器对每一传输的字节记数。当所有的字节都被转移完时，终止计数信号(引脚 B27)就显示出所有数据的转移已经完成了。

需要存储器直接存取的设备通过指定一条 DMA 请求线来提请求。存储器直接存取的第一请求级是最高的体系统先级，存储器直接存取的第三请求级是最低体系优先级。当接受存储器直接存取请求时，存储器直接存取应答线(引脚 B4、B17、B26 和 B15)来证实这一请求。只有第 1、2、3 条的存储器直接存取应答线能被系统用作普通用途。存储器直接存取应答线 0(引脚 B19)是用来处理刷新存储器操作的最优先级应答信号。

另一种获得微处理器中断的方式是激活中央处理单元中断线(引脚 B4、B25、B24、B23、B22 和 B21)中的一条。有六条总线可以使用的中断线(被称为中断线 2 到中断线 7)，其中中断线 2 是最高优先中断线，中断线 7 是最低中断线。中断线"0"是系统时钟中断线，中断线"1"专用于键盘服务。当一中断程序完成之后，中央处理单元又返回到它的原先的程序。

尽管 IBM PC/XT 及其所有的兼容计算机已经过时好几年了，二十世纪八十年代早期卖出的许多 XT 仍将在今后的一些年里在家庭、办公室、学校中使用。由于这个原因，读者必须熟悉 XT 总线体系。

1.3.4 PC/AT 总线

XT 计算机的缺陷没有多长时间就显露出来了，XT 计算机的存储器的容量太有限了。它的 Intel 8088 微处理器只能支持一个"8"位的数据总线。它的系统服务(例如中断线，直接

内存存取)无法满足简单处理之外的所有应用的要求。计算机设计者不得不更新XT而不抛弃已制作了许多扩展产品的电路板。IBM的下一代计算机称为PC/AT。AT计算是以"16"位Intel 80286微处理器为核心的,但同时保留了几乎所有8位XT扩展产品的兼容性。

AT总线仅仅在最初的"62"引线卡边连接器上附加了一个第二连接器而非重新设计XT总线。这一辅助连接器就是如图1.10所示的标有C1至C18和D1至D18的36引线的卡边连接器。AT增加了五个中断线、八个数据线(D8到D15)、四个存储器直接存取请求/应答线、四个额外地址(A20到A23)和一些附加的控制线。表1.3列出了每一AT引线的标号。AT总线的易适应性使之被IEEE采用作为工业标准,所以它也被称为工业标准结构(ISA)总线。

图1.10 IBM PC/AT总线连接器引脚图

表1.3 PC/AT总线扩展板引脚

B1	地	A1	I/O通道检查
B2	复位驱动	A2	数据7
B3	+5V直流电压	A3	数据6
B4	中断请求2	A4	数据5
B5	-5V直流电压	A5	数据4
B6	DMA请求2	A6	数据3
B7	-12V直流电压	A7	数据2
B8	+12V直流电压	A8	数据1
B9	+12V直流电压	A9	数据0
B10	地	A10	I/O通道就绪
B11	实存储器写	A11	地址使能
B12	实存储器读	A12	地址19
B13	I/O写	A13	地址18
B14	I/O读	A14	地址17
B15	DMA应答3	A15	地址16
B16	DMA请求3	A16	地址15
B17	DMA应答1	A17	地址14
B18	DMA请求1	A18	地址13
B19	刷新	A19	地址12
B20	时钟	A20	地址11
B21	中断请求7	A21	地址10
B22	中断请求6	A22	地址9
B23	中断请求5	A23	地址8

（续表）

B24	中断请求 4	A24	地址 7
B25	中断请求 3	A25	地址 6
B26	中断请求 2	A26	地址 5
B27	计数终止	A27	地址 4
B28	地址锁存	A28	地址 3
B29	+12V 直流电压	B29	地址 2
B30	振荡器	A30	地址 1
B31	地	A31	地址 0
D1	存储器 16 位片选	C1	系统总线高便能
D2	I/O 16 位片选	C2	未锁存地址 23
D3	中断请求 10	C3	未锁存地址 22
D4	中断请求 11	C4	未锁存地址 21
D5	中断请求 12	C5	未锁存地址 20
D6	中断请求 13	C6	未锁存地址 19
D7	中断请求 14	C7	未锁存地址 18
D8	DMA 应答 0	C8	未锁存地址 17
D9	DMA 请求 0	C9	存储器读
D10	DMA 应答 5	C10	存储器写
D11	DMA 请求 5	C11	数据 8
D12	DMA 应答 6	C12	数据 9
D13	DMA 请求 6	C13	数据 10
D14	DMA 应答 7	C14	数据 11
D15	DMA 请求 7	C15	数据 12
D16	+5V 直流电压	C16	数据 13
D17	主	C17	数据 14
D18	地	C18	数据 15

因为 AT 总线允许在同一计算机内同时使用 8 位和 16 位扩展电路板，所以系统总线高锁存(引脚 C)必须被激活以进行 16 位数据转移。扩展电路板使用 16 位存储器片选或 16 位输入/输出系统片选来要求数据传送或传出存储器及输入/输出端口。零等待状态信号可以通过排除总线周期的等待状态使扩展电路板实质性地加速总线操作。

在 16 位区域内取址是由辅助存储器读(引脚 C0)和存储器写(引脚 C10)信号来实现的，任何在 1Mb 取址范围内的取址使用原 62 引线连接器上的存储器读/写线，而任何在 1Mb 之上的取址使用附加读或写的信号。

AT 总线也允许其它微处理器共享有限总线(与存储器直接存取相似)。新的中央处理单元能够通过激活主线(引脚 D17)几毫秒控制秒系统，当原中央处理单元必须收回总线控制权时，激活刷新信号以警告"访问"中央处理单元放弃控制以便原始中央处理单元能够刷新它的局部存储器。

1.3.5 PC/微通道总线

1987 年研制成的 IBM 微通道总线是对计算机扩展槽容量的重新考虑。此时，计算机扩

展槽容量已被扩展到32位数据处理领域。微通道扩展电路板在物理上比全槽的AT电路板小,它使用更小,更紧致的连接器。但是不同之处远远不止在物理性能方面。微通道总线的设计目的就是比AT总线更快、更有效。它带有32位数据,32位地址成线使之具有4Gb内存寻址能力,一个内部的从50位到10kHz信号的固定声频频道和一内部的视频图形阵列(VGA)卡,微通道总线面向高性能的计算而研制出来的。微通道总线还被设计为能处理总线仲裁。总线仲裁是各个不同的中央处理单元控制总线的能力。图1.11阐述典型微通道卡槽引脚。每个引脚都在表1.4中作了标示。

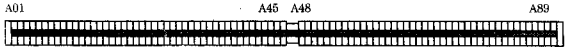

图1.11 一IBM微通道总线连接器的引脚图

表1.4 PC/微通道总线的扩展电路板引脚

A01	卡启动	B01	音频地
A02	Make 24	B02	音频信号
A03	地	B03	地
A04	地址11	B04	14.3MHz振荡器
A05	地址10	B05	地
A06	地址9	B06	地址23
A07	+5V直流电	B07	地址22
A08	地址8	B08	地址21
A09	地址7	B09	地
A10	地址6	B10	地址20
A11	+5V直流电	B11	地址19
A12	地址5	B12	地址18
A13	地址4	B13	地
A14	地址3	B14	地址17
A15	+5V直流电	B15	地址16
A16	地址2	B16	地址15
A17	地址1	B17	地
A18	地址0	B18	地址14
A19	+12V直流电	B19	地址13
A20	地址解码锁存	B20	地址12
A21	预空(preempt)	B21	地
A22	猝发(burst)	B22	中断9
A23	−12V直流电	B23	中断3
A24	仲裁0	B24	中断4
A25	仲裁1	B25	地

（续表）

A26	仲裁 2	B26	中断 5
A27	−12V 直流电	B27	中断 6
A28	仲裁 3	B28	中断 7
A29	仲裁同意	B29	地
A30	计数终止	B30	保留
A31	+5V 直流电	B31	保留
A32	状态位 0	B32	通道检查
A33	状态位 1	B33	地
A34	存储器 I/O	B34	命令
A35	+12V 直流电	B35	通道就绪回送
A36	卡就绪	B36	片选反馈
A37	数据线 0	B37	地
A38	数据线 2	B38	数据线 1
A39	+5V 直流电	B39	数据线 3
A40	数据线 5	B40	数据线 4
A41	数据线 6	B41	地
A42	数据线 7	B42	通道复位
A43	地	B43	保留
A44	数据大小 16 回送	B44	保留
A45	刷新	B45	地
A46	KEY(空位)	B46	KEY(空位)
A47	KEY(空位)	B47	KEY(空位)
A48	+5V 直流电	B48	数据线 8
A49	数据线 10	B49	数据线 9
A50	数据线 11	B50	地
A51	数据线 13	B51	数据线 12
A52	+12V 直流电	B52	数据线 14
A53	保留	B53	数据线 15
A54	状态字节高使能	B54	地
A55	卡数据大小 16	B55	中断 10
A56	+5V 直流电	B56	中断 11
A57	中断 14	B57	中断 12
A58	中断 15	B58	地
A59	保留	B59	保留
A60	保留	B60	保留
A61	地	B61	保留
A62	保留	B62	保留
A63	保留	B63	地
A64	保留	B64	数据线 16
A65	+12V 直流电	B65	数据线 17
A66	数据线 19	B66	数据线 18

（续表）

A67	数据线 20	B67	地
A68	数据线 21	B68	数据线 22
A69	+5V 直流电	B69	数据线 23
A70	数据线 24	B70	保留
A71	数据线 25	B71	地
A72	数据线 26	B72	数据线 27
A73	+5V 直流电	B73	数据线 28
A74	数据线 30	B74	数据 29
A75	数据线 31	B75	地
A76	保留	B76	字节使能 0
A77	+12V 直流电	B77	字节使能 1
A78	字节使能 3	B78	字节使能 2
A79	数据大小 32 回送	B79	地
A80	卡数据大小 32	B80	32 位数据线使能
A81	+12V 直流电	B81	地址 24
A82	地址 26	B82	地址 25
A83	地址 27	B83	地
A84	地址 28	384	地址 29
A85	+5V 直流电	B85	地址 30
A86	保留	B86	地址 31
A87	保留	B87	地
A88	保留	B88	保留
A89	地	B89	保留

尽管微通道体系提供了一些高于 AT 总线的性能优势，但是它没有被广泛地接受。因此这本书将不深入地探讨微通道总线。

1.4 大容量存储器综览

既然读者有了关于计算机概念和扩展总线结构的初步了解，这一节读者将会对本书中所涉及到的大容量存储设备有一个总的概念。

对于所有的计算机系统来讲，储存大量信息的能力是很关键的。存储使程序和数据成为计算机的工作部分。如果没有存储能力，计算机运行时程序指令和数据将不得不用人工输入，这可不是一个可行的选择。除了存储已存的程序和数据外，存储设备常成为程序操作结果的贮藏所。例如，读者能装载和操作一个单词处理器，然后将已创建的文本用作唯一标识的计算机文件保存在存储设备中。显然，正如我们了解的那样，计算机没有大容量存储器将毫无用处。

读者可能会想到有许多方式来贮藏数字信息。一些最早的方法是利用穿孔纸卡片或者磁带。然而穿孔纸储存在历史上已经消失很长时间了，其它技术已经被开发出来的以满足现代计算机的需要。这本书讨论了现在通用的三种主要的存储技术，它们是固态集成电路、磁