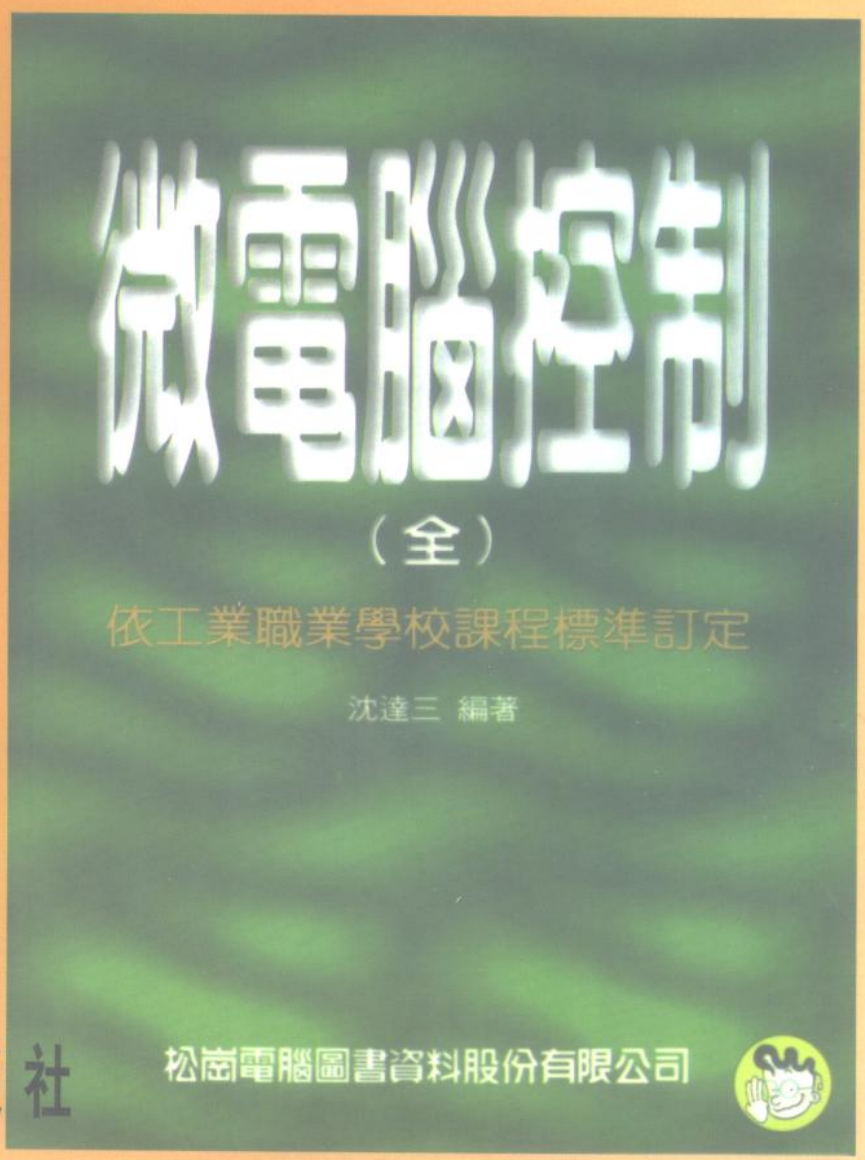
天大松岗系列丛书

微电脑控制

沈达三 编著



天津大学出版社

微电脑控制

（台湾工业职业学校教材）

沈达三　编著

天津大学出版社
松岗电脑图书资料股份有限公司

内 容 提 要

本书是台湾工业职业学校的教材。全书共分五章，第一章介绍微电脑的结构及主要组成部件的功能；第二章介绍微机的应用软件及硬件；第三章介绍微电脑常用的输入、输出结构与应用；第四章介绍微电脑界面电路的应用，数/模，模/数转换与应用；第五章介绍微电脑的应用，主要有屏幕显示、绘图控制、声音控制、计时电脑控制、打印控制、存储器电路控制等。另外，还以步进电机与直流电机的控制为例，具体说明微电脑应用的设计过程、元件选取、软件调试等全过程。

微电脑控制

沈达三 编著

天津大学出版社出版
（天津大学内）
邮编：300072
高等教育出版社印刷厂印刷
新华书店天津发行所发行

开本：787×1092 毫米 1/16 印张：11½ 字数：287 千
1996 年 8 月第 1 版 1996 年 8 月第 1 次印刷
印数：1—5000
ISBN 7-5618-0865-8
TP·79 定价：24.00 元

编辑大意

一、本书系依照1986年2月台湾教育部门有关工业职业学校电机电子专业“微电脑控制”课程标准编写而成，可供工业职业学校电子类、电机类、控制类、制冷类等专业第三学年第一学期每周三学时之教学使用。

二、本书以IBM PC或其兼容机种为主。全书共分五章：第一章介绍微电脑结构，使学生对微电脑系统结构有初步的认识；第二章介绍软件及硬件，其中讨论到DEBUG程序及EDLIN程序，让学生学会以汇编语言建立文件并加以执行；第三章介绍微电脑I/O结构的分析与应用，举出实际的I/O地址解码电路及I/O结构，并提出两种最常使用的I/O型式：(1)可编程式I/O，(2)中断式I/O；第四章介绍微电脑界面电路之应用，内容有A/D和D/A转换与其应用，并辅以多功能界面卡来完成温度控制系统的设计；第五章介绍微电脑的应用实例，包含屏幕显示、绘图控制、声音控制、计时电路控制、打印机控制、家电控制、存储器电路之控制及工业上的应用(有步进电机与直流电机的控制系统实例)。

三、本书的特色为：

1. 各章节的安排力求前后呼应，以最直接的方式引导读者立即进入状况。

2. 提供足够的范例和练习，让读者可由模仿中学到控制的实务及理论。

3. 所有程序皆以IBM PC XT能执行者为原则。

4. 在程序的编写方面，先以简短的小程序设计来满足初学者的需求，再逐步加入巨集指令、模组外调用及LIB程序库的建立，使读者能扩展程序编写的能力。

5. 不少范例所讨论的控制系统都是笔者多年从事微电脑课程教学的结晶，相信必能帮助初学者更迅速地掌握其控制法则及技巧。

四、本书虽经再三校对，恐有疏漏，尚祈诸位施教老师能惠予赐教，不胜感激。

编者　谨识

目　录

第一章　微电脑结构

1-1　微电脑系统结构

微电脑系统必须具备以下单元：

(1)中央处理单元(简称 CPU)

(2)输入单元

(3)输出单元

(4)存储单位

其中的中央处理单元可算是系统的指挥中心。微电脑就以中央处理单元(CPU)的积体电路(IC)编号为其主要的名称，例如八位元的 Z-80，6502 及 16 位元的 8086/8088、80286、Z800 等等，目前已推出 64 位元的 80586。CPU 可控制其它单元的动作。

输入单元包括所有数据输入的设施，它能将外界数据传送到系统内部(CPU 或内存储器)。常用的输入设备有：

(1)键盘和开关

(2)光笔、标鼠、扫描器

(3)磁盘驱动器

(4)读卡机

(5)电传打字机

(6)磁带机

(7)模/数转换器(简称 ADC)。

输出单元则是用来把电脑内部数据传送到外界的部门，所谓内部数据可以涵盖 CPU 内的数据或存储器的数据。常用的输出设备有：

(1)指示器(LED、LCD 或指示灯)

(2)电传打字机

(3)打印机与绘图机

(4)屏幕(CRT)

(5)数/模转换器(简称 DAC)

存储器可分两类：一类是固定于主机板上的 IC(ROM 及 RAM)；另一类则是辅助存储器或次存储器，它是可以任意添增或移去的磁盘或磁带。ROM 和 RAM 是连接在电路上的主存储器。ROM 内常存着系统的操作程序，例如 IBM 的开机及启动系统程序以及所有微电脑产品的监督系统程序(monitor)，都是由厂商预存在 ROM 中且永不消失的程序，也唯有它才能使系统正常操作。

图 1-1 说明了微电脑的基本结构。

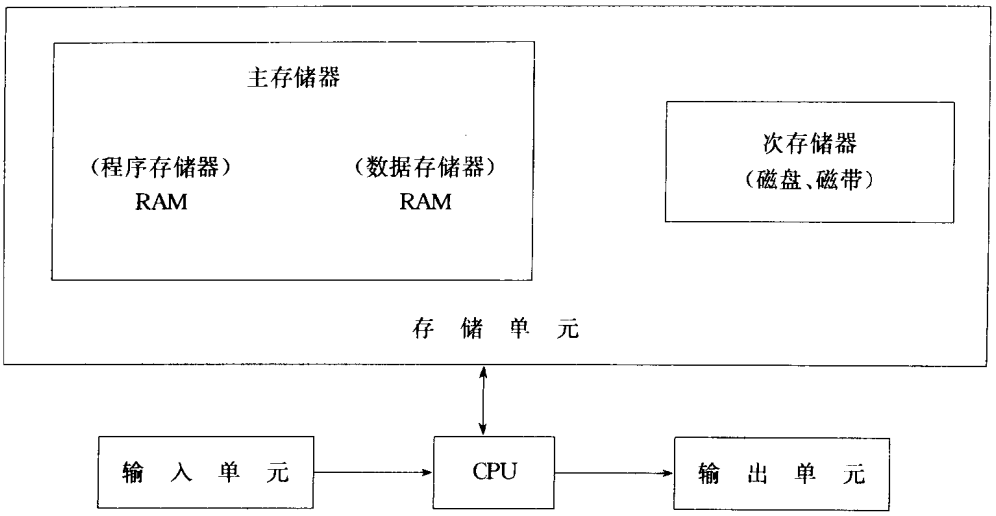


图1-1 微电脑的基本结构

常用的磁盘驱动器又分硬式磁盘与软式磁盘两大类，目前还有光盘(容量更大)。硬式磁盘容量有10MB、20MB及40MB为较普遍者，软磁盘则以320KB和1.2MB两者最常用。至于磁盘尺寸则有8英寸、5 $\frac{1}{4}$及3 $\frac{1}{2}$英寸之分，其中以5 $\frac{1}{4}$英寸最普遍。

1-2 总线

微电脑系统的总线有以下三类：

(1)数据总线

(2)地址总线

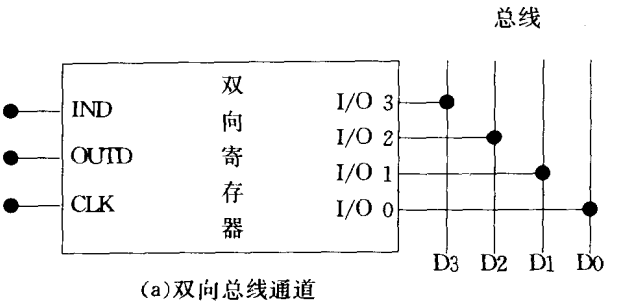
(3)控制总线

其中数据的传送是属于双向性质者，故称之为双向总线通道。通常所说的总线只不过是系统的公用通道(PC板中的连接线)。微电脑系统中有四个主要单元，各单元间的数据传送都由数据总体管理。CPU内有许多寄存器，内存中有更多的寄存器(RAM和ROM)，它们都要连接到数据总线上，以便互相传送数据。图1-2即为双向总线通道的电路与简化图形。

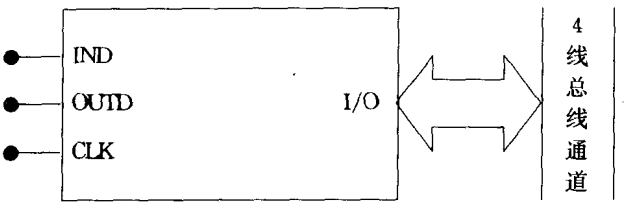
若要控制寄存器与总线的数据流向，只须以IND和OUTD两功能端来定义：若是IND即表示数据由总线传给寄存器；反之若是OUTD，则表示数据是由寄存器传出去。总线通道的宽度(根数)即表示微电脑的种类，例如8位元电脑的数据总线有8条线，16位元则有16条线。

地址总线是用来定义输入单元、输出单元或存储器的地址的。微电脑的每个I/O设备都有其特定地址，而主存储器则由0号地址起，定义到数十K或百万号。以8位元CPU而言，可寻址到64K，而16位元CPU则可寻址到1M(一百万)，$2^{16}=65536(64K)$，$2^{20}=1M(1K=1024)$。地址总线是供CPU选择I/O设备或主存储器地址用的。被选上的地址便可与数据总线沟通，每个地址都有其暂存器或I/O设备。未被选上的地址，其数据是无法进出数据总线的，只有被选上的才可进出数据总线。

图1-3即为256×8存储器的结构图，其中的地址总线有8条(A0～A7)，故可以选到$2^8=$



(a)双向总线通道



(b)简化图形

图1-2 数据总线

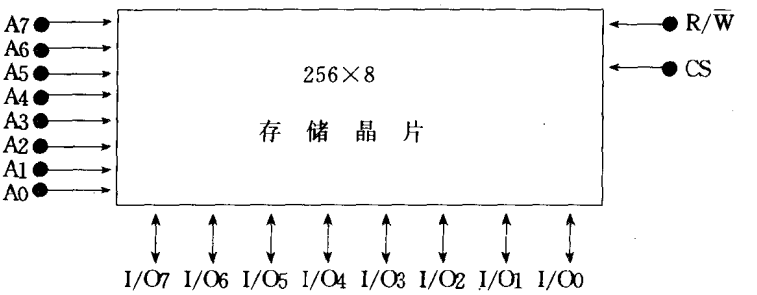


图1-3 256×8存储晶片

256个暂存器(地址)。I/O线有8条,故可提供8条数据线至数据总线。至于数据的流向(输入或输出)则由R/$\overline{W}$控制。地址总线是单向性质的,只由CPU发出,传至各存储器或I/O地址。

控制总线是CPU控制各单元的信号线,各单元也可以经由控制总线向CPU传达信号,例如要求中断、数据备妥等信号。

1-3 存储器

存储器分主存储器和次存储器两类。次存储器是主机板外部的辅助存储器(磁盘、磁带),主存储器则以主机板内的RAM和ROM为主。

RAM是可读写的存储器,当您打入程序或数据时,便是将指令或数据存入RAM中,它必须在开机的情况下进出数据,只要电源一关则所有数据皆消失。ROM是只读存储器,它的数据是以固化的方式载入,此后数据即永存于存储器中。一般微电脑系统皆备有一段ROM区,专门存放系统控制程序,其结构如图1-3所示,只是I/O端全改为单向输出的OUT信号而已,

且无 R/$\overline{W}$ 控制线。ROM 的数据写入方式有四：

(1)固化式程序化 ROM：它在出厂前即被制作完成，永远无法改变内容，适用于大量生产。

(2)可编程式 ROM：又称 PROM，内部有极细小的连线，使用者可选择性地将其烧断，此后数据永远无法改变。

(3)可拭除可编程式 ROM：又称 EPROM，程序可用高压方式烧录进去，不要时也可用紫外光拭除之，因此 EPROM 上方有一透明窗口供紫外光照射用。

(4)电压变化式 ROM：又称 EEPROM，可用适当电压清除数据。

依操作方式之不同，存储器又可分静态及动态两类。静态 RAM 系由正反器组成，每个正反器代表一个位元，它的电路(内部)比动态 RAM 复杂，通常要十个电晶体才能完成一个位元，因此容量比动态 RAM 小(动态 RAM 一位元只需三个电晶体)。

在电路连接上，静态 RAM 不必多加定时充电电路，而动态 RAM 则必须外加定时充电电路。因为动态 RAM 系由电容充电的方式来存放数据，每隔一段时间即会放电而使电压下降，故必须在电压尚未降至临界电位前再将其电位提升。这种周期性的充电动作即为“数据更新”(refresh)，所以动态 RAM 要另加一个自动充电电路，如此一来也就增加了电路的复杂性。

1-4 存储器电路

微电脑系统中使用到许多存储元件，以 IBM PC 而言，它的 CPU 可寻址到 $2^{20}=1M$ 的空间。因此在装配存储器电路时，不可能只用一个 RAM 或 ROM，必须选择合适的 RAM 和 ROM，再进行电路的设计。

存储器元件的规格皆以下列方式表示：

〔容量〕×〔宽度〕

例如，1K×1 即代表容量 1K 宽 1 位元的存储元件。如果以它来设计 1K×8 的存储电路，则需并联 8 只 IC(图 1-4)，此时地址线共有 10 条(A0～A9)，必须同时加至各 IC 的地址输入端。而每个 IC 的 I/O 输出则依序接至数据总线上(D0～D7)。R/$\overline{W}$ 要连接在一起，接至 CPU 的 R/$\overline{W}$ 控制端。

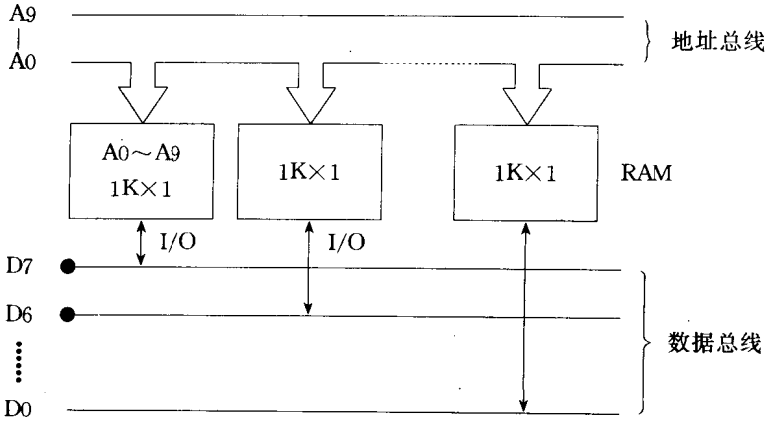


图1-4 增加字节宽度

图 1-4 可以将字节的宽度扩增至 8 位元(原 IC 只输出一位元)。若要扩充容量，则须增加

地址总线的线数。例如要将图 1-4 的容量(原为 1K)扩充至 2K,则可多加一条地址线 A10,再以 A10 来选择存储区块(图 1-4 简画为一区块),图 1-5 即为容量扩充的简化图。

ROM 电路结构与 RAM 一样,只是少掉 R/$\overline{W}$ 控制线而已。动态 RAM 则需外加充电电路,以便做存储器刷新的工作。图 1-4 和图 1-5 都省略了 R/$\overline{W}$ 的连接,也省略了电源的连接,读者不妨自行加入。

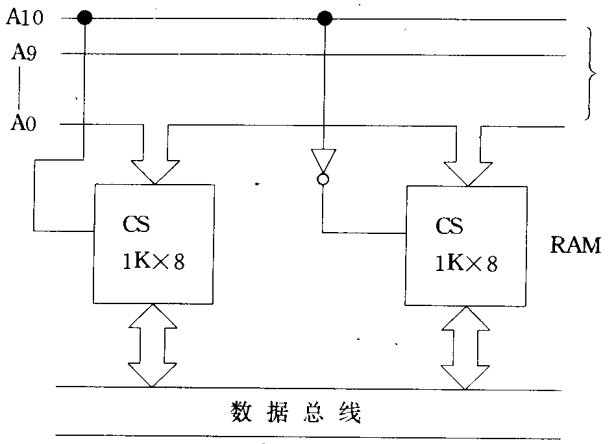


图1-5 扩充容量

8086/8088CPU 的地址可定到 1M bytes。也就是说,它能从 0 到 0FFFFFH 的地址中选到任何一个地址。由于 8088 基本上是以 8 位元的型式来处理数据(数据总线只有 8 条线)的,所以在传送 16 位元数据时,必须分成两次完成,这是为了让 8088 也能在 8 位元的系统中工作而设计的。至于 8086 虽有 16 条数据线,可以一次传送 16 位元数据,但是由于存储器电路结构的安排,有时候也要分两次才能传送 16 位元数据。图 1-6 为 8086 地址空间的安排情形,它将存储器地址的所有奇数地址集中在一块,偶数地址集中在另一块,而以 BHE 信号来选奇数地址,以 A0(信号地址线的最近位元)信号来选偶数地址。

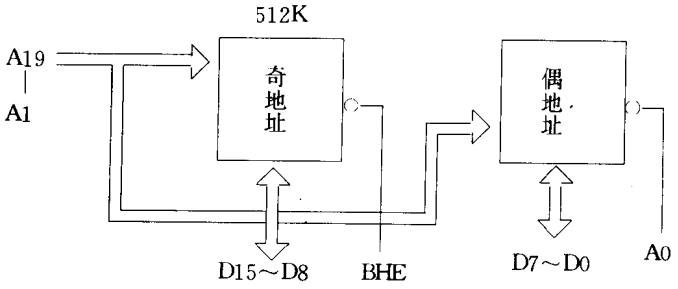


图1-6 8086地址空间安排

所谓奇数地址是指地址码为 1、3、5……的地址,偶数地址则是 0、2、4……的地址。数据进出对每一地址而言只有 8 位元,所以奇地址的数据接 D15～D8,偶地址的数据线接 D7～D0,二者合成才为 16 位元数据。

严格来说,8086/8088 是 8 位元 CPU,因为在它的存储器地址上,每个地址只能存 8 位元数据。因此,其指令也是 8 位元指令,存储器空间的安排也是为了迁就 CPU 的数据进出,其中

每块区间 512K，例如图 1-6 中的 BHE(Bank High Enable)为 0，表示奇地址有效；若同时 $A_0=0$，则偶地址也有效，如此便可读写两个地址的数据(共 16 位元)，然而这只适合在数据由偶地址存起的时候。反之，由奇地址存起，则必须分两次才能读写一个字节组(Word)，亦即先令 BHE 有效，读写奇地址内容，此时 $A_0=1$，偶地址区间无法存取数据，必须等第二次存取的时候，才能读写偶地址内容。

如果进出存储器的数据为 8 位元形式，则不管数据是存在偶地址或奇地址，皆可一次存取完毕，唯有在 16 位元数据存取时，才会因数据所在地址(奇或偶地址)而影响传送速率。

8088 CPU 由于数据线只有 8 条，每次只能存取 8 位元数据，因此不管数据位于奇地址或偶地址，其传送速率不受影响，故不须使用 BHE 信号线(8088 CPU 没有 BHE)，所以每个字元组(两个 byte)数据必须两次存取，速率则比不上 8086 快。

8086/8088 CPU 把 1M 的存储器空间看成数个区段(Segment)，每个区段有 64K 的空间，必须以区段寄存器来寻址。由于 CPU 内部的寄存器皆为 16 位元型式，所以要将区段寄存器的值右边多填 4 个 0，形成 20 位元型式($2^{20}=1M$)，如此才可寻址到 1M 的空间。实际地址必须由区段寄存器和偏移(offset)地址做下述加法而得：

```
    区段寄存器
+       偏移地址
-------------------
  实际位址(20 位元)
```

8086/8088 的区段寄存器和偏移地址通常按下列组合搭配而成：

CS：IP

SS：SP

DS：SI

ES：DI

其中 CS 代表码段(code segment)；SS 代表堆栈段(stack segment)；DS 代表数据段(data segment)；ES 代表额外段(extra segment)。

程序写作时至少使用一个区段(码段)，最多可用 4 个区段，因此最大的程序容量可达 256K。至于各区段的定义，可依程序设计需要而定，段地址也可相重叠，或是分开来安置。

早期 8 位元 CPU 的程序设计是不分区段的，它的总存储器空间只有 64K，数据和程序要写在同一个连续区域，只将堆栈分开来放在另一块固定的区域。这种处理方式对于小型程序(只有一个程序在存储体中)并无影响，然而在同时处理多个程序时，就会出现困难。此时最好将程序和数据分开来放置。如果要将 8086/8088 的各区段安放在同一地址(CS=DS=ES)也是可以的，此时的程序空间只有 64K，和 8 位元 CPU 一样。

1-5 微电脑的控制单元

中央处理单元(CPU)一般都做成 40pin 的包装，它必须能提供 16 条地址线、8 条或 16 条数据线和若干条控制线。由于 CPU 必须以外加时钟脉冲信号来作为指令的处理时序，因此也

要有时钟脉冲(CLOCK)输入端。

图 1-7 是 8 位元 CPU 结构图。

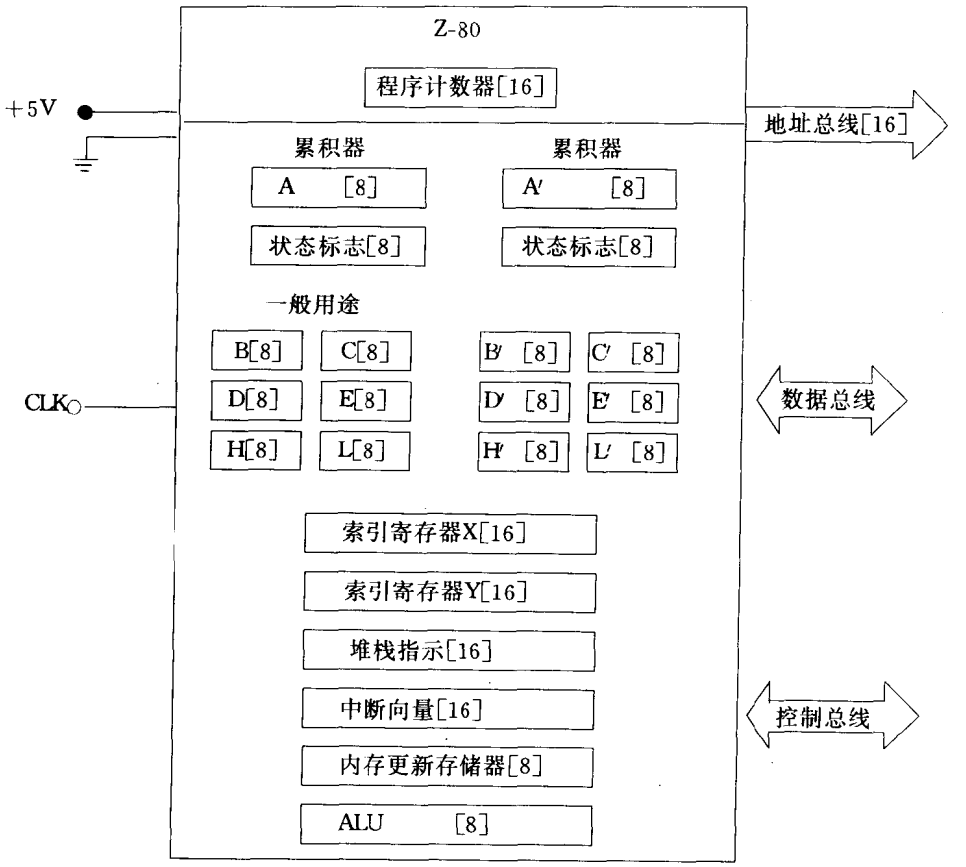


图1-7 8位元CPU结构图

由图 1-7 可算出 8 位元 CPU 的包装接脚数如下：

电源〔2〕+脉冲〔1〕+地址总线〔16〕+数据总线〔8〕+控制总线〔13〕=40

可见恰好是 CPU 的标准接脚规格(40pin)。如今 16 位元 CPU 也要包装成 40pin 的标准规格，且寻址能力为 1M(2^{20})，数据传送又是 16 位元型式(数据总线要用到 16 条线)，40pin 的包装势必无法满足其要求。为此 8086 与 8088 在设计时即采用多工操作的接脚方式，亦即将部分接脚的功能扩充为两种功能。例如地址线和数据线使用同一接脚，而有了所谓的 AD_0～AD_{15} 及 A_{16}/S_3、A_{17}/S_4、A_{18}/S_5、A_{19}/S_6 等双功能接脚出现。

8088 由于只传送 8 位元数据，所以只有 AD_0～AD_7 是地址数据共用线，而且 8088 也不需要 BHE 控制信号，所以将此信号端(34pin)改为 SSO 信号。另外 8086 的 M/$\overline{IO}$控制线和 8088 的 IO/$\overline{M}$ 控制线的控制电位恰好相反，这是为了要让 8088 能和 8 位元的 8086 CPU 相兼容而做的修正。

图 1-8 为 8086 的内部结构，PSW 为状态标志寄存器，ALU 为算数逻辑单元，指令区又称为指令流字节队列，是用来预存指令的。8088 只有 4 个 bytes 的指令流队列，8086 则有 6 个

bytes。图 1-9 为 8086 CPU 的详细结构说明。

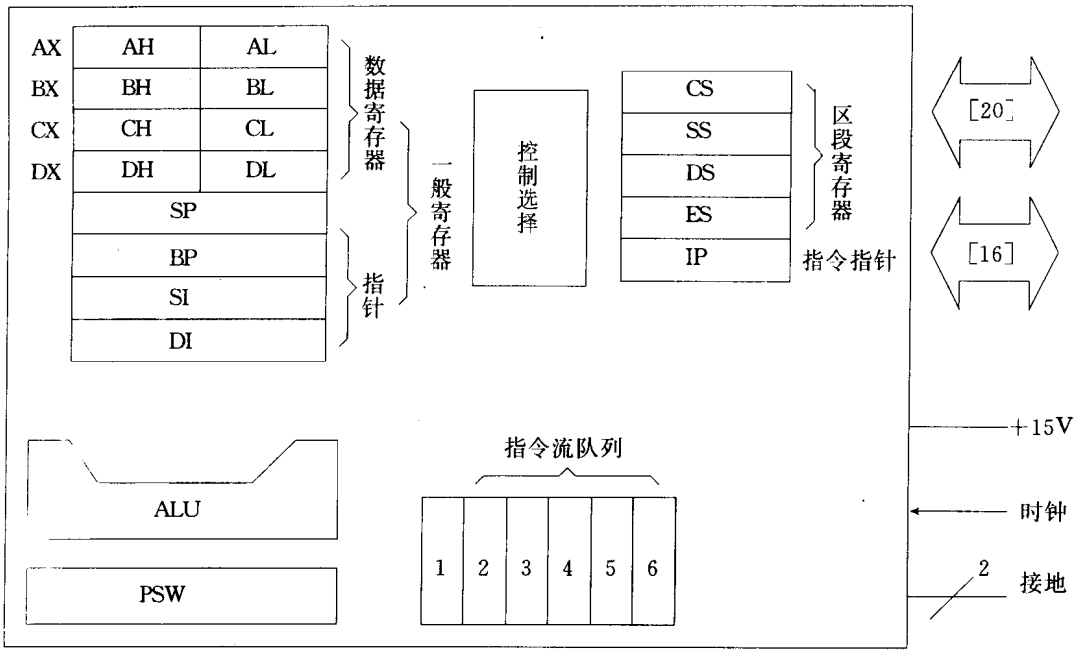


图1-8　8086内部结构

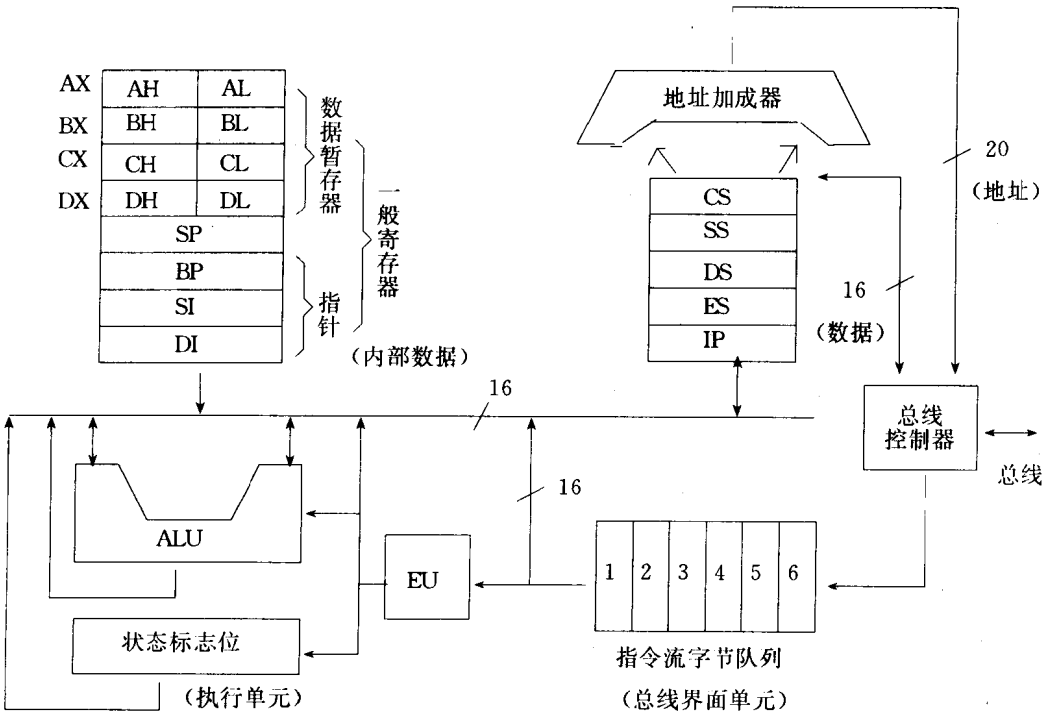


图1-9　8086内部详细结构

8086 的 CPU 内部基本上可分为两大部分：

(1)执行单元(EU)

(2)汇总线界面单元(BIU)

执行单元内有 8 个 16 位元的一般寄存器，可以配合 ALU 做算术逻辑运算，并以一个 16 位元的状态标志寄存器来存放目前的状态。

总线界面单元的主要任务是产生实际地址(20 位元)来和存储器沟通，以便存取指令码或数据。它内部有 4 个区段寄存器和一个指令指针(皆为 16 位元型式)由外面程序指令所能影响的寄存器就是 8 个一般寄存器和 4 个区段寄存器，以及一个状态标志。

1-6 指令

8086 指令共分成六类：

(1)数据传送指令

(2)算术运算指令

(3)逻辑运算指令

(4)字串处理指令

(5)程序转移指令

(6)控制指令

数据传送可在寄存器间、寄存器与存储器间、暂存器与堆栈间、寄存器与 I/O 端口间进行。常用的指令有：

(1)MOV(传送字节或字)、LEA(装入有效地址)、LDS 及 LES(装入段寄存器)

(2)PUSH、POP(堆栈操作)

(3)XCHG(交换)

(4)IN、OUT(输入、输出)

(5)XLATB(查表)

算术指令只能在 CPU 内部的寄存器进行，它可以做 8 位元或 16 位元的加、减、乘、除运算。至于运算的数目可以是有符号数或无符号数，也可以是紧凑式 BCD 码或非紧凑式 BCD 码。所谓紧凑式是指 4 个位元代表一个 BCD 码(0～9)的数目，而非紧凑式则是 8 个位元代表一个 BCD 码，其中的高 4 位元被载入 0，通常是用来将数目转换为 ASCII 码用的，因为每个 ASCII 码要以 8 位元构成。

常用的算术指令有：

(1)ADD(加法)

(2)SUB(减法)

(3)MUL(乘法)

(4)DIV(除法)

(5)INC(加 1)

(6)DEC(减 1)

(7)NEG(求反)

(8)CMP(比较)

逻辑指令也是在CPU内部寄存器进行，主要指令有：

(1)AND(逻辑乘)

(2)OR(逻辑加)

(3)XOR(按位加)

(4)NOT(取反)

(5)TEST(测试位元)

(6)SHL、SHR(算术左移、算术右移)

(7)ROL、ROR(循环左移、循环右移)、RCL及RCR(通过进位的循环左移、通过进位的循环右移)

字串处理指令是将传送指令(MOV)的功能扩大而成的指令，可以处理到64K位元组的数据，主要指令有：

(1)MOVS(字串传送操作)

(2)CMPS(字串比较操作)

(3)SCAS(字串扫描操作)

(4)LODS(装入字串操作)

(5)STOS(保存字串操作)

程序转移指令是用在程序区段或程序指针(CS:IP)被更改的时候，程序的执行地址会发生变动，因为CPU是根据CS:IP的指引去存储器中取出指令来译码及执行的，主要的指令有：

(1)CALL(过程调用)

(2)JMP(无条件转移)

(3)LOOP(条件循环)

(4)INT(中断)

控制指令是用来改变CPU状态标志位的指令，主要指令有：

(1)CLC,STC(清除或设定进位标志位)

(2)CLD,STD(清除或设定方向标志位)

(3)CLI,STI(清除或设定中断标志位)

(4)HLT(处理器暂停，直到出现中断或复位信号才继续执行)

1-7 寻址模式

8086的寻址模式可分成两类：

(1)数据地址模式

(2)转移地址模式

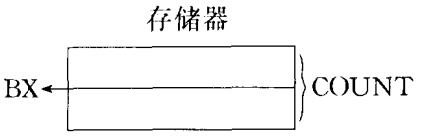
现说明数据地址寻址模式如下：

(1)立即寻址——运算数据即附在指令中。

例:MOV CX,100

(2)直接寻址——直接取得数据。

例:MOV BX,COUNT

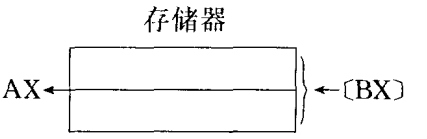


(3)寄存器寻址——暂存器内容即为运算数据。

例:MOV DS,AX

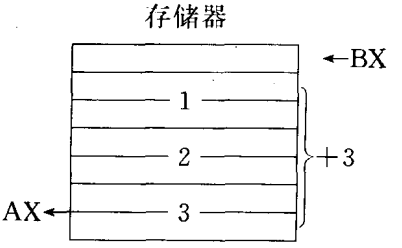
(4)寄存器间接寻址——数据的有效地址由寄存器取得。

例:MOV AX,〔BX〕



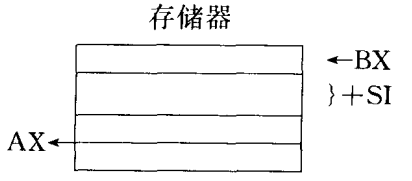
(5)寄存器相对寻址——数据地址由寄存器和偏移地址决定。

例:MOV AX,〔BX+3〕



(6)基址索引寻址——数据地址由基址及索引寄存器决定。

例:MOV AX,〔BX+SI〕



(7)基址相对索引寻址——数据地址由基址、索引寄存器及偏移地址决定。

例:MOV AX,〔BX+SU+3〕