

新编考研辅导丛书

Microcomputer Principle & Application

微型计算机原理及应用辅导

李伯成 编著

重点·难点·考点

典型例题解析

精选习题与解答

模拟试题与答案

西安电子科技大学出版社

<http://www.xduph.com>



新编考研辅导丛书

微型计算机原理及应用辅导

李伯成 编著

西安电子科技大学出版社

内 容 提 要

本书在介绍《微型计算机原理及应用》课程内容及重点、难点的基础上,通过对例题和综合性习题的分析,详细阐述了微型机的基本原理、基本概念及一些实际应用,特别注意到对基本概念进行综合应用,以及解决具体工程问题的描述。

本书除了作为报考硕士研究生人员的辅导书外,对在校师生及从事微型机应用的工程技术人员也有较好的参考价值。

图书在版编目(CIP)数据

微型计算机原理及应用辅导/李伯成编著. —西安:西安电子科技大学出版社,2000.9
(新编考研辅导丛书)

ISBN 7-5606-0915-5

I. 微... II. 李... III. 微型计算机-研究生-入学考试-自学参考资料 IV. TP36

中国版本图书馆 CIP 数据核字(2000)第 41480 号

责任编辑 李惠萍 陈宇光

出版发行 西安电子科技大学出版社(西安市太白南路2号)

电 话 (029)8227828

邮 编 710071

<http://www.xduph.com>

E-mail: xdupfxb@pub.xaonline.com

经 销 新华书店

印 刷 西安兰翔印刷厂

版 次 2000年9月第1版 2001年3月第2次印刷

开 本 787毫米×960毫米 1/16 印张 13.5

字 数 264千字

印 数 4 001~12 000册

定 价 17.00元

ISBN 7-5606-0915-5/TP·0845

*** 如有印装问题可调换 ***

本书封面贴有西安电子科技大学出版社的激光防伪标志,无标志者不得销售。

《新编考研辅导丛书》编审委员会

主任委员：	傅丰林	副校长	教授
副主任委员：	焦李成	博士生导师	教授
委 员：	刘三阳	博士生导师	教授
	曾兴雯		教授
	孙肖子		教授
	李伯成		教授
	张永瑞		教授

序

人类走过了又一个千年之交。世界正在发生深刻变化。这一变化是20世纪以来科学技术革命不断深入的必然结果，她已经成为推动社会发展与文明进步的革命性力量。人类走过了农业经济时代、工业经济时代，正在进入知识经济时代。

自1978年国家恢复招收研究生和1980年建立学位制度至今，研究生教育已经走过了20多年的历程，她是我国教育结构中最高层次的教育，肩负着为国家现代化培养高素质、高层次创造性人才的重任，是我国增强综合国力、增强国际竞争力的重要支撑力量。研究生教育的改革和发展，直接关系到21世纪我国第三步战略目标的实现。

西安电子科技大学是一所有70年历史的教育部直属的重点高等学校，也是国家“211”重点建设高校，同时又是国家首批具有硕士、博士授予权的单位之一。现有在校生15 000多人，其中研究生2000余人。学校建有研究生院等10个学院，有3个国家重点学科和27个省部级重点学科；同时建有3个国家重点实验室和16个省部级重点实验室，在“通信与信息系统”、“信号与信息处理”、“电路与系统”、“微电子与固体电子学”、“电磁场与微波技术”和“密码学”等领域设有“长江计划”特聘教授岗位。近年来，西安电子科技大学研究生教育得到了迅速的发展，年招生已超过1000人，招生质量和培养质量在省内名列前茅。毕业生遍布国内外，受到了广泛赞誉。

当前，研究生教育面临新的挑战，同时给研究生教育的发展带来了新的机遇。如何选拔优秀人才是一项长期的研究课题。西安电子科技大学出版社组织我校长期在教学科研第一线、在国内有一定知名度的教授编写了这套考研辅导丛书，并从重点、难点、考点、典型例题分析及自测题等方面进行有剖析、对比总结性的阐述，有助于考生在有限的时间内复习所学内容，并有新的提高和启发。

我们相信此套丛书的出版对我国工科电子信息类研究生教育的发展会起到积极的促进作用。

西安电子科技大学研究生院

博士生导师 焦李成

2000年7月

前 言

《微型计算机原理及应用》是一门重要的计算机专业基础课,侧重于基本原理和工程实践。为了使读者更好地掌握本课程的内容,特编写本书。

在本书中,我们按主要的知识模块划分各章。在每一章中分别介绍本模块中的内容及其重点、难点、典型例题。考虑到该课是一门工程实践性很强的课程,在书中专门增加了一章,仔细分析一些常用的工程问题。读者将会看到,这些问题的解决应用了本书多个章节的内容,即在综合应用基本概念的基础上才能获得问题的解决。这也是学好这门课的主要目的。

现在各种微型机系列较多,不可能也没有必要样样都学。只要认真地学好其中一种,真正掌握它的基本概念和内容实质,对今后理解和掌握其他未学过的微型机将会有很大的帮助。编写本书的目的也就是使读者更好地掌握《微型计算机原理及应用》书中所阐述的内容。

本书编写时,力求简洁明了、深入浅出。同时,希望能将近几年作者在科研和教学实践中的经验与体会反映在书中,使书中的内容更加充实并具有实际应用价值。因此,本书除了作为考研辅导书之外,还可作为本科生学习《微型计算机原理及应用》课的参考用书,也可以作为从事微机应用的技术人员的参考书。

尽管我们力图将书写好,由于时间紧迫和水平有限,不妥之处仍难免,请读者予以指正。

作 者

1999. 12.

目 录

第 1 章 微型计算机概述	1
1. 1 主要内容	1
1. 1. 1 微型计算机的组成	1
1. 1. 2 微型计算机的工作过程	2
1. 1. 3 8088 CPU 的引线及内部寄存器	3
1. 1. 4 8088 CPU 的工作时序	7
1. 1. 5 系统总线的形成	8
1. 1. 6 关于其他 CPU	9
1. 2 重点及难点	10
1. 2. 1 微型机的组成及工作	10
1. 2. 2 从 8088 CPU 引线到系统总线形成	11
1. 2. 3 内部寄存器及内存地址的形成	11
1. 2. 4 时序的概念	11
1. 3 例题分析	12
1. 4 自测题	14
第 2 章 指令系统与汇编语言程序设计	16
2. 1 主要内容	16
2. 1. 1 寻址方式	16
2. 1. 2 8088 CPU 的指令系统	22
2. 1. 3 汇编语言简单说明	35
2. 1. 4 基本程序设计方法	40
2. 1. 5 汇编语言源程序的结构与开发	48
2. 2 重点及难点	50
2. 2. 1 8088 CPU 的寻址方式	50
2. 2. 2 8088 CPU 的指令系统	51
2. 2. 3 汇编语言及其程序设计	54
2. 3 例题分析	54
2. 4 自测题	63
第 3 章 微型机内部存储器	65
3. 1 主要内容	65
3. 1. 1 半导体存储器的分类及主要技术指标	65

3. 1. 2 随机读写存储器(RAM)	67
3. 1. 3 只读存储器(ROM)	74
3. 2 重点及难点	79
3. 2. 1 静态存储器(SRAM)	79
3. 2. 2 EPROM	81
3. 2. 3 EEPROM	81
3. 3 例题分析	82
3. 4 自测题	84
第4章 微型计算机的基本输入输出技术	86
4. 1 主要内容	86
4. 1. 1 微型计算机中最常用的内存与接口的编址方式	86
4. 1. 2 无条件传送	87
4. 1. 3 查询方式	91
4. 1. 4 中断方式	93
4. 1. 5 DMA 方式	107
4. 2 重点及难点	116
4. 2. 1 无条件传送	116
4. 2. 2 查询方式传送	117
4. 2. 3 中断方式传送	117
4. 2. 4 DMA 方式传送	120
4. 3 例题分析	121
4. 4 自测题	124
第5章 接口芯片及其应用	126
5. 1 主要内容	126
5. 1. 1 简单的接口芯片及其应用	126
5. 1. 2 可编程并行接口 8255	129
5. 1. 3 可编程定时器 8253	141
5. 1. 4 可编程串行通信接口 8250	147
5. 2 重点及难点	157
5. 2. 1 简单接口	157
5. 2. 2 可编程并行接口 8255	157
5. 2. 3 可编程定时器 8253	158
5. 2. 4 可编程串行通信接口 8250	159
5. 3 例题分析	159
5. 4 自测题	168
第6章 综合应用例题分析	170

附录 西安电子科技大学计算机学院本课程近几年研究生入学考试试题	191
A 一九九六年考试试题	191
B 一九九七年考试试题	193
C 一九九八年考试试题	195
D 一九九九年考试试题	197
E 二〇〇〇年考试试题	199
F 二〇〇一年考试试题	203
主要参考资料	205

第 1 章 微型计算机概述

本章首先介绍微型计算机的概念框图,说明微型计算机的主要组成部分及各部分的功能与作用,并说明微型机的简单工作过程,以期从概念上对微型机建立初步的认识。然后,再从 CPU 的外部引线及内部寄存器开始,说明 CPU 的外特性及其工作时序。最后,叙述在此 CPU 的基础上,如何构成微型机的系统总线。

在本章及后面的章节中,要用到一些预备知识,例如数制(二进制、十进制、十六进制等)及其运算、符号数的表示方法、常见的编码(例如 BCD 码、ASCII 码等)以及数字电路和模拟电路的基本知识等。我们认为读者均已学习并掌握了这些知识,在本书中只是应用,不再做任何解释。

1.1 主要内容

1.1.1 微型计算机的组成

微型计算机是由硬件和软件两大部分构成的,下面分别加以说明。

1. 微型计算机的硬件概念框图

从概念上讲,微型计算机都可以认为是由如图 1.1 所示的几部分组成。

从图 1.1 中可以看出,微型计算机硬件以 CPU 为核心,利用 CPU 的信号形成系统总线,并在系统总线上连接内存和接口。其中,内存用来存放程序和数据;接口用于将外设和计算机连接在一起,可利用接口将数据传送到外设,也可将外设的数据通过接口传送到微型机中。CPU 发出的地址信号、控制信号以及数据信号都是通过系统总线传送的。

由此可见,微型计算机硬件主要由 CPU、系统总线、内存和接口这样几个重要部分组成。如果将这些部分集成在一块集成电路芯片中,就构成了人们常说的单片微型计算机。

2. 软件

图 1.1 所示的仅是微型机的硬件,只有硬件的计算机称为“裸机”,它是不能工作的。

为了使微型机完成某种功能,就需要为它配备软件。在微型机中通常配有操作系统、应用程序、用户程序等软件。软件是微型计算机系统的重要组成部分。

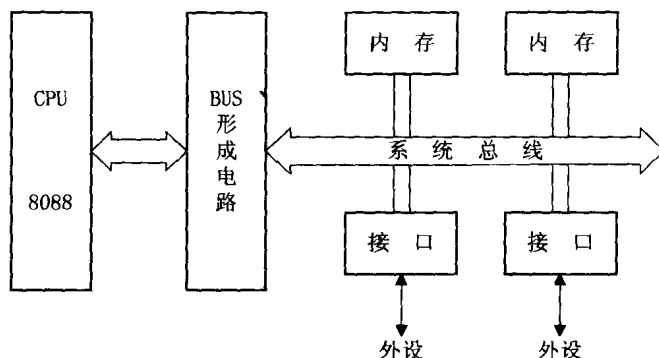


图 1.1 微型计算机概念组成框图

1.1.2 微型计算机的工作过程

微型计算机通过执行程序来实现各种功能。程序是完成某种功能所需指令的集合，而指令则是命令 CPU 完成某种最简单的操作的命令。

通常，我们总是将一个复杂的任务分解成很多最简单的操作，这些最简单的操作可由一条条指令来实现，而这些指令的集合就构成了完成该任务的软件——程序。

由于 CPU 只能识别二进制代码，因此程序最终也必须用二进制代码来表示。将程序代码放到内存中，而后让 CPU 执行这段程序。CPU 从内存中逐条地将指令读到 CPU 内部并识别和执行它。CPU 逐条顺序地执行这些指令，当指令执行完毕，程序的功能也就实现了。

下面我们用一个最简单的例子来说明程序的执行过程。若要计算

$$5+7=?$$

则可将这种计算写成如下的程序(第一列为各指令的二进制代码)：

B005	MOV	AL, 05H
B307	MOV	BL, 07H
02C3	ADD	AL, BL
E63E	OUT	3EH, AL
F4	HLT	

在上面的程序中，前两条指令分别将 5 和 7 传送到寄存器 AL 和 BL 中，第三条指令将 AL 中的 5 与 BL 中的 7 相加，并将相加的和放到 AL 中，至此就求得了两数之和。最后是一条停机指令，命令 CPU 停止执行指令，原地待命。

如将上面程序左侧的指令代码放入内存中，然后命令 CPU 开始执行这个程序。CPU 首先从内存中读出 B0H，然后再从内存的下一个地址读出 05H，并把它放到 AL 中。这样第一条指令就执行完了。接着 CPU 以同样的方式依次执行下面的指令，第三条指令实现了两数相加并将结果放在 AL 中。执行第四条指令时，CPU 首先从内存中读出该指令的两个操作码 E6H 和 3EH，然后执行该指令，将 AL 中的内容写到地址为 3EH 的接口上。直到 CPU 从内存中取出停机指令 HLT 的指令操作码 F4H，执行该指令 CPU 就处于停机状态。

1.1.3 8088 CPU 的引线及内部寄存器

本书以 8088 CPU 为例进行说明，因为 8088 CPU 在过去的 PC 机中曾得到广泛地应用且具备一般 CPU 的共性。同时，在过去的教材中，都是在讲述了 8088 CPU 之后再简单介绍后来出现的 80386、80486 和奔腾。由于受教学时间的限制，有关保护模式方面的内容通常只作简单介绍，因此，在这里我们主要对 8088 CPU 的外部引线及内部寄存器进行说明。

1. 8088 CPU 的外部引线

8088 CPU 是一块具有 40 个引脚的集成电路芯片，其各引线定义如图 1.2 所示。

(1) 最小模式下的引线信号

在 8088 CPU 的引线信号中有一条 $MN/\overline{MX}$ 信号线，这是一条输入信号线。当该信号为高电平时，规定 8088 CPU 工作在最小模式之下。所谓最小模式，就是构成的微型机中只有一个 8088 CPU。而当该信号线为低电平时，规定 8088 CPU 工作在最大模式之下，即在构成的微型计算机中，存在着多个 CPU，除了 8088 CPU 之外还有其他 CPU（例如 8087、8089 等）。

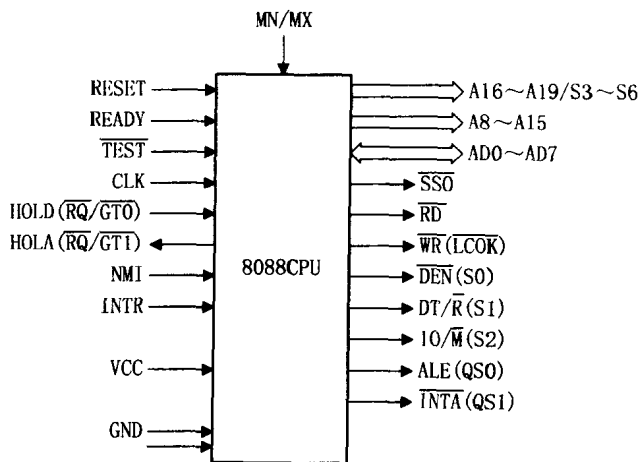


图 1.2 8088 CPU 引脚图

在最小模式下，A16~A19/S3~S6 这 4 条信号线以及 AD0~

AD7 这 8 条信号线采用分时复用方式，即某一时刻这些线上送出地址，而另一时刻则送出状态或用于传送数据。信号线 A8~A15 这 8 条线则只用于送出地址信号。由于以上这 20 条信号线从 CPU 输出，最后都经过三态门，因此，它们在特殊情况下会呈现第三态（即高阻状态）。

在此模式下, CPU 送出用于内存或接口的控制信号: $\overline{RD}$ 、 $\overline{WR}$ 、 $IO/\overline{M}$ 信号。当 $IO/\overline{M}$ 为低电平时, 用于读写内存, 即对内存寻址; 而当它为高电平时用于对接口的寻址。当将数据写入内存或接口时 $\overline{WR}$ 为低电平(有效), 而当从内存或接口读出数据时 $\overline{RD}$ 为低电平(有效)。这三个信号为三态输出。

信号线 ALE、 $\overline{DEN}$ 和 $DT/\overline{R}$ 这三个信号是用于形成系统总线的。其中, 当地址线上出现有效的地址时需锁存这些地址, ALE 就用作锁存脉冲。 $\overline{DEN}$ 为数据允许信号, 用它来控制数据总线驱动器, 当 $\overline{DEN}$ 有效时使数据总线驱动器有效工作。信号 $DT/\overline{R}$ 用于控制数据总线驱动器的信号传送方向, 当 CPU 读数据时, 信号流向 CPU; 而当 CPU 写数据时, 方向指向系统总线。这三个信号为三态输出信号。

信号 INTR 和 $\overline{INTA}$ 分别为(可屏蔽)中断请求和中断响应信号。有关中断的概念以及在微机中的具体应用是十分重要的。所谓中断是指当由于微机内部或外部某个事件发生, 迫使 CPU 暂时中断正在执行的程序, 转向对所发生事件的处理, 处理结束后又回到被中断的程序接着中断前的状态继续向下执行的过程。INTR 是一个外部中断请求的输入端, INTR 的请求是可以屏蔽的, 即当标志寄存器中 IF 的状态为 0 时, CPU 不会响应该请求, 也就是说它被屏蔽了。而当 $IF=1$ 时, CPU 才有可能(还有其他条件)对它作出响应。CPU 对 INTR 的响应包括许多内容, 其中就有从 $\overline{INTA}$ 引线信号端输出两个负脉冲。该信号为三态输出信号。

引线信号 NMI 为非屏蔽中断请求, 顾名思义该中断不可屏蔽, 即不受中断屏蔽位 IF 的影响。不管 IF 是 0 还是 1, 只要满足其他条件 CPU 就会对它作出响应。当 NMI 为上升沿时, 向 CPU 提出请求。

保持请求信号 HOLD 和保持响应信号 HLDA 是一对为完成 DMA 传送而设置的信号, 保持请求 HOLD 为高电平有效的输入请求信号。当外设与内存需要快速交换数据时, 可以通过 DMAC 在 CPU 的 HOLD 输入端加上一个高电平的请求信号, CPU 会在一个总线周期结束时作出响应。CPU 的响应包括两个方面: 一方面是从引线 HLDA 上送出高电平加到 DMAC 上, 告诉 DMAC 其 DMA 请求 CPU 已经响应。另一方面 CPU 放弃了对系统总线的控制权, 将系统总线的控制权交给 DMAC。DMAC 就可以利用系统总线实现 DMA 传送。有关 DMA 传送的详细内容将在后面的章节中加以说明。

在 CPU 的引线中 READY 信号是一个高电平有效的输入信号。当该信号为高电平时, CPU 正常地执行指令, 而当其输入为低电平时 CPU 原地踏步, 不再向下执行程序。直到 READY 输入为高电平时 CPU 才继续向下执行。

输入信号 $\overline{TEST}$ 称为测试信号, 当 CPU 执行 WAIT 指令时对 $\overline{TEST}$ 输入电平进行测试。若该输入为高电平, 则 CPU 在此指令上原地踏步, 直到此输入变为低电平, CPU 才继续向下执行。

复位信号 RESET 是一个重要的输入信号。当它输入为高电平时, CPU 将被复位。复位后 DS、SS、ES、IP、F 等内部寄存器均清零, CS 寄存器为 FFFFH。因此, 当 RESET 变低启动时, CPU 的启动入口地址为 FFFF0H。在复位时 CPU 的三态输出信号 AD0~AD7、A8~A15、A16~A19/S3~S6、 $\overline{DEN}$ 、 $\overline{RD}$ 、 $\overline{WR}$ 、DT/ $\overline{R}$ 、IO/ $\overline{M}$ 、 $\overline{INTA}$ 均处于高阻状态(浮空); 而其他输出信号均为无效状态。

激励 CPU 工作的是 CPU 的时钟, 时钟的每一个周期激发 CPU 内部硬件完成某些细微工作。这些细微工作积累起来就完成了总线周期的工作。一个或多个总线周期工作的积累就完成了一条指令的执行。

在最小模式下尚有一个状态输出信号 SSO。它和前面提到的输出信号 DT/ $\overline{R}$ 和 IO/ $\overline{M}$ 结合在一起, 三个信号的不同状态的编码表示 CPU 的不同的工作状态。

(2) 最大模式下的引线

当 MN/ $\overline{MX}$ 输入端接上低电平时, CPU 将工作在最大模式之下。

最大模式下 8088 CPU 的引线信号大部分与最小模式下的信号相同, 只有下面的 10 条引线信号的名称及功能发生了改变, 它们是:

SSO 在最大模式下总为高电平。

$\overline{RD}$ 信号不再使用。

$\overline{DEN}$ 、DT/ $\overline{R}$ 、IO/ $\overline{M}$ 这三个信号重新定义为 S0、S1、S2。这三个信号的不同状态的编码可以表示 CPU 的不同的工作状态。为了在最大模式下利用这些状态, Intel 公司专门研制了总线控制器 8288, 利用该芯片对 S0、S1 和 S2 三个状态信号进行译码, 产生一系列的总线控制信号, 例如 $\overline{INTA}$ 、 $\overline{IOR}$ 、 $\overline{IOW}$ 、 $\overline{MEMR}$ 、 $\overline{MEMW}$ 等等。

信号 ALE 和 $\overline{INTA}$ 被重新定义为队列状态信号 QS0 和 QS1。这两个信号状态的不同编码用来表示 CPU 内部四个字节的指令预取队列的状态。

保持请求和保持响应信号 HOLD 和 HLDA 重新定义为 $\overline{RQ}/\overline{GT0}$ 和 $\overline{RQ}/\overline{GT1}$ 。功能也由最小模式时的一个请求和一个响应变为二个请求和二一个响应, 即 $\overline{RQ}/\overline{GT0}$ 和 $\overline{RQ}/\overline{GT1}$ 每一条引线即可完成请求, 在其上加一个宽度为一个时钟周期宽度的负脉冲, 而 CPU 响应请求会在同一引线上送出一个时钟周期宽度的负脉冲, 而当 DMA 传送结束时, DMAC 在同一引线上输入一个时钟周期宽度的负脉冲表示传送结束。可见, 这种功能 $\overline{RQ}/\overline{GT0}$ 可以实现, $\overline{RQ}/\overline{GT1}$ 也同样可以实现。

最大模式下, 将 $\overline{WR}$ 信号重新命名为 $\overline{LOCK}$ 。它在 CPU 执行带有“LOCK”前缀指令时输出为有效的低电平, 用以封锁总线。

2. 8088 CPU 的内部寄存器

8088 CPU 是一片大规模集成电路芯片, 其内部结构十分复杂。我们不可能也没有必要弄清楚其内部的细节, 只要知道与我们应用有关的内容。这种思路将适用于本书后面用

到的各种内存和接口芯片，以至于将来的工作中。

(1) 编程中所用到的内部寄存器

在后面学习的过程中，以及指令系统和汇编语言中都会频繁地用到这些寄存器。其中通用寄存器为：16 位的 AX、BX、CX、DX、BP、SP、SI、DI。这 8 个寄存器中前 4 个为数据寄存器，主要用于传送数据和进行算术及逻辑运算。同时，它们还具备其他功能，例如 BX 可存放偏移地址，作为地址指针使用；CX 在循环指令和串操作指令中用作计数单元；DX 在 IO 指令中可存放接口地址，实现接口地址的寄存器间接寻址。AX、BX、CX 和 DX 这 4 个 16 位的寄存器还可以拆成两个独立的 8 位寄存器单独使用，用 AH, AL; BH, BL; CH, CL 和 DH, DL，分别表示一个字的高 8 位和低 8 位。这 8 个独立的 8 位寄存器也是通用的，均可以实现传送、算术逻辑运算及位移、循环等操作。在执行多次循环或位移指令时，必须用 CL 作为循环或位移的次数计数器。

另外 4 个通用寄存器 BP、SP、SI、DI 主要用作地址指针，SI 和 DI 还可以作为通用寄存器使用，用于传送、算术、逻辑、循环、位移等各种指令作为源或目的寄存器之用。

(2) 段寄存器

在 8088 CPU 中有 4 个特殊的 16 位寄存器——段寄存器 CS、DS、SS 和 ES。它们的作用只是为了在只有 16 位内部寄存器的 8088 CPU 中能够形成 20 位地址，以寻址 1MB 的内存空间。在 8088 CPU 内部形成 20 位的物理地址是这样进行的：

$$20 \text{ 位的物理地址} = \text{段寄存器的内容} \times 16 + 16 \text{ 位的偏移地址} \quad (1-1)$$

(3) 专用寄存器

在 8088 CPU 中有两个专用的 16 位寄存器：指令指针 IP 和标志寄存器 F。

在 CPU 执行指令的过程中每从内存中取出一个指令字节，指令指针 IP 的内容自动加 1，并指向下一个要读出的指令字节。这是因为 CPU 读指令字节的内存物理地址为：

$$20 \text{ 位读指令字节的物理地址} = CS \times 16 + IP \quad (1-2)$$

标志寄存器 F 也是一个 16 位的寄存器，其各位标志如图 1.3 所示。

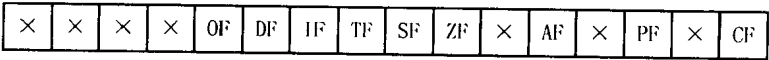


图 1.3 标志寄存器 F 的各位标志

标志寄存器中的每一位标志均表示 CPU 执行指令的状态或表示某种功能。读者必须记住标志寄存器各位标志的功能。由于在每一种 CPU 中均有类似的寄存器，因此，学好 8088 CPU 的标志寄存器对今后理解其他 CPU 也是有用的。

1.1.4 8088 CPU 的工作时序

1. 指令周期与周期

前面已经提到,由指令集合而成的程序放在内存中,CPU 从内存中将指令逐条读出并执行。我们将 CPU 完整地执行一条指令所花的时间叫做一个指令周期。在后面的章节中我们可以看到,有的指令很简单,执行时间就比较短,而有的指令很复杂,其执行时间比较长,但我们都称为一个指令周期,只不过时间长短不同而已。

如果再细分,一个指令周期还可以分成若干个总线周期,即一条指令是由若干个总线周期来完成的。那么什么是总线周期呢?8088 CPU 通过总线对外部(存储器或接口)进行一次访问所需的时间称为一个总线周期。这里主要是指 8088 CPU 将一个字节写入一个内存单元或写入一个接口地址或者 8088 CPU 由内存或接口读出一个字节到 CPU。尽管中断响应及 DMA 传送也有它们的总线周期,但在本章里我们主要说明 CPU 读内存中的指令字节、读内存数据字节、读接口字节、将字节写入内存和将字节写入接口这 5 种总线周期。

现在我们已经明确,一条指令可以由若干个总线周期来完成。同时,要进一步明确,在正常情况下,上述 5 个总线周期均由 4 个时钟周期来完成。时钟周期就是加在 CPU 芯片上 CLK 时钟信号的周期。

至此,可以看到一条指令的执行时间可以用总线周期来说明,还可以用时钟周期来度量。这就是指令周期、总线周期和时钟周期之间的关系。

2. 总线周期的工作时序

在正常情况下,前面提到的 5 个总线周期均由 4 个时钟周期来完成。为了便于说明问题,我们首先以 CPU 向内存写入一个字节的总线周期来简要说明。该总线周期从第一个时钟周期 T1 开始,在 T1 时刻 CPU 从 A16~A19/S3~S6 这 4 条引线上送出 A16~A19、从 A8~A15 这 8 条线上送出 A8~A15、从 AD0~AD7 这 8 条线上送出 A0~A7。可见,在这个时钟周期里,CPU 从它的 20 条引线上送出了 20 位地址信号 A0~A19,而且时钟 T1 之后,A16~A19 及 AD0~AD7 上的信号将变为其他信号。因此,CPU 在 T1 周期里送出 ALE 地址锁存信号,我们可以用这个信号将 A0~A19 锁存在锁存器中,使地址信号在整个总线周期里保持不变。当然,地址信号 A8~A15 可以不锁存,因为这些信号在整个总线周期里保持不变。在此 T1 期间 CPU 由 $\overline{\text{IO}}/\overline{\text{M}}$ 送出低电平并在整个总线周期中一直维持低电平不变,表示该总线周期是一个寻址内存的总线周期。

在时钟周期 T2 里,CPU 将写入内存的数据从 AD0~AD7 上送出来,加到数据总线 D0~D7 上。同时 CPU 还会送出 $\overline{\text{WR}}$ 控制信号,在地址信号 A0~A19、 $\overline{\text{IO}}/\overline{\text{M}}$ 及 $\overline{\text{WR}}$ 的共同作用下,将 D0~D7 上的数据写入相应的内存单元中。写入内存的操作通常是在 $\overline{\text{WR}}$ 的后沿(其上升沿)来实现的。这时的地址、数据信号均已稳定,写操作的工作就很可靠。

以上就是在最小模式下正常的内存写入过程。若在实际应用中内存的写入时间要求较长而 CPU 提供的写入时间却较短(最长也只有 4 个时钟周期),则在这样短的时间里数据无法可靠地写入。为了解决这个问题,可以利用 CPU 的 READY 信号。当 CPU 的总线周期里的时钟周期 T3 开始时(下降沿),CPU 内部硬件测试 READY 信号的输入电平。若此时 READY 为低电平,则 CPU 在 T3 之后不执行 T4,而是插入一个等待时钟周期 TW。在 TW 的下降沿 CPU 继续检测 READY 输入电平,若其为低电平,则继续插入等待的时钟周期 TW。就这样一直插到 READY 为高电平时,则插入停止并执行总线周期的 T4。这样一来,一个写入内存的总线周期就可以由 4 个时钟周期延长为更多个时钟周期,以满足低速内存的要求。

以上我们仅以最小模式下将数据写入内存的总线周期为例说明 8088 CPU 的时序。对于其他读内存或读、写接口其思路是类似的,在此不再说明。

我们强调时序并不是时序图本身怎么重要,而是其思路及概念对理解本书后面的许多内容是很有用的。这是因为:一方面时序对理解指令的执行很有意义;另一方面时序对于内存和接口的连接及调试是极其有用的。这些问题在本书的后面的综合应用中详细加以讨论。

1.1.5 系统总线的形成

从前面的图 1.1 中可以看到,微型计算机是在 CPU 形成的系统总线上接上内存和接口来构成的。在后面的章节中,将逐一讨论内存和接口是如何接到系统总线上的。因此,弄清系统总线的信号怎么产生将是很重要的。在形成系统总线时,我们认为几种常用的数字集成电路(如 74LS244、245、373 等)是读者早已熟悉的。

1. 最小模式下系统总线的形成

在最小模式下,为了形成系统总线上的地址总线 A0~A19,采用 3 片具有三态输出的锁存器 74LS373,利用 CPU 送出的 ALE 信号将出现在时钟 T1 时刻的地址信号 A0~A19 锁存在 74LS373 的输出端,并一直保持到下一个总线周期锁存新的地址信号为止。显然,由于地址信号 A8~A15 在整个总线周期内保持其状态不变,因此,这 8 个信号可以不用锁存器,而用三态门(74LS244 或其他三态门)。

在形成地址总线信号时,应注意,若将来在系统总线上不进行 DMA 传送,则三态门输出控制端(74LS373 的 $\overline{OE}$ 及 74LS244 的 $\overline{E1}$ 、 $\overline{E2}$)均可接成永远有效,即可接地。若要在系统总线上进行 DMA 传送,则那些三态门的控制端就应当由 CPU 的 HLDA 输出信号或其他译码输出控制信号来控制。

数据总线形成电路可由双向三态门 74LS245 或其他类似的器件来实现。74LS245 的 A 边接 CPU 的 AD0~AD7,而其 B 边接出去即为数据总线 D0~D7。而 74LS245 的允许控制