16位微处理机

【英】伊恩R·惠德沃思 著

李三立 译

中国铁道出版社

16位微处理机

〔英〕 伊恩R. 惠德沃思 著　　李三立 译

中国铁道出版社
1988年·北京

内 容 简 介

本书从微处理机的硬件、软件、系统结构及应用等方面，对16位微处理机进行了综合的论述。

全书共分12章。其内容包括：准16位微处理机；早期16位微处理机；现代16位微处理机；接口技术；指令系统；汇编代码软件和开发；系统软件和操作系统；高级语言；多微处理机系统；应用以及未来的发展等。

本书可作为高等学校计算机专业高年级学生及研究生的教材，也可作为微计算机领域中科技人员的参考书。

16—Bit Microprocessors

Ian R. Whitworth

Granada Publishing Ltd, 1984

16位微处理机

〔英〕伊恩R. 惠德沃思著 李三立译

中国铁道出版社出版

责任编辑 郭 宇 封面设计 安 宏

新华书店总店科技发行所发行

各地新华书店经售

中国铁道出版社印刷厂印

开本：850×1168毫米 $\frac{1}{32}$ 印张：16.625 字数：402千

1988年2月第1版 第1次印刷

印数：0001—3,500册 定价：3.65元

译 者 的 话

微处理机技术的迅速发展，是体现当今科学技术活力的重要特征之一。随着微处理机性能的不断提高，它对人类生活影响的程度也在不断深化。

如果把七十年代说成是8位微处理机成长的年代，那么八十年代就是16位微处理机成长和日益成熟的年代。在数据处理、工业控制以至事务处理方面，16位微处理机的应用愈来愈趋于统治地位。随着社会信息处理能力需求的不断增长，16位微处理机的研制和开发已成为当今微处理机应用的主流。这种趋势无论是在国内还是在国外，都已经变得越来越明显。

16位微处理机与8位微处理机相比，不仅数据处理宽度延伸了一倍，更主要是它在体系结构、指令系统及设计思想等方面所做的重要改进。因此，现代16位微处理机比8位微处理机的功能更强、速度更快、使用更灵活、外部结构更优越。

现在已经有很多书籍对8位微处理机做了详尽的分析，但是，有关16位微处理机的书籍还不多，为此，我们特翻译此书献给读者。本书与其它同类书籍比较，具有一定的特色。它根据不同厂商生产的16位微处理机产品，对它们的体系结构、接口技术、指令系统、汇编语言与开发、系统软件、多微处理机系统及应用等方面进行了综合分析，并着重论述了它们存在的主要优缺点。这同其它专一论述同一厂商产品的书籍相比，形成了鲜明的对照。因此，可以这样说，本书是从横的方向（即综合介绍了不同厂商的产品）来论述的，而其它书则是从纵的方向（即只介绍同一厂商的产品）来论述的。这种横向的论述可以使读者摆脱厂商提供的用户手册的约束，并站在较高的位置上来全面分析和了解16位微处理机的

结构与设计方面的特点，从而能加深理解。

本书可以作为高等院校计算机专业高年级学生和研究生的教材，也可作为微计算机领域中科技人员的参考书。

本书涉及的内容相当广泛，从硬件到软件，从结构到应用。但是，这也给翻译工作带来一定的难度。译者各方面水平都很有限，在翻译工作中一定会有很多错误，希望读者指正。

译 者

1986年5月

前 言

微处理机世界正日益变得复杂化，每天都宣布有新的进展。尽管有这种复杂性，微处理机仍从这种或那种方式，愈来愈广泛地渗透到技术先进国家人们生活的所有方面。甚至不发达国家也感受到微处理机的冲击。在微处理机诞生这短短的十二年期间，半导体公司已经经历了几代微处理机设计的演变。起初，他们生产的器件看成是逻辑系统的替代品，或者看成是微控制器。但是，现在他们生产的最高级产品在速度和功能方面都可以与小型计算机领域内的高档机相竞争。

在写这本书的时候，几乎所有以微处理机为核心的产品都是围绕着4位或8位微处理机的中央处理部件(CPU)或者具有片上存储器和接口的单片计算机来设计的。这些器件技术相当成熟，它们是经过实践验证的设计，是由丰富的知识和应用经验所支持的。从专业工程师的观点看，8位微处理机的知识——硬件和系统设计、软件设计和维护以及应用几乎都已有成规。的确，刚刚毕业的工程师很可能在他的学院或大学课程的实践环节中已经使用过8位微处理机，而很多老一些从事实际工作的工程师也会通过制造厂家培训课程而通晓8位微处理机，并把这些器件设计成有用的产品，从而取得实际经验。对于任何希望了解8位微处理机的人来说，有很多基础教科书可以作为这个领域中很好的入门书。因此，我们先假定大家已经具有8位微处理机的知识。为了节省篇幅，这本书不想从零开始。第一章是复习的一章，它用来帮助读者回忆8位微处理机的概念和术语。本书的其余部分专门写那些也许大家还不熟悉的16位微处理机思想，以及其它相应的进展。

16位微处理机的设计远远不是具有两倍字长的8位机。虽

然某些早期的 16 位微处理机只是由于其字长而得到速度方面的优点，除此以外，并未提供其它更多的特点。但是，现代的 16 位微处理机则完全与其相应的 8 位机不同，它在设计上提供了相当高级的指令系统，硬件上支持操作系统和高级语言，它还为总线结构接口提供了方便，这将构成多处理机装置的基础。所有这些都将简化系统设计者的任务。起初 8 位机的设计是体现了这样一种观点，即从硬件逻辑设计者的立场考虑以满足在一个处理机中哪些是需要的。然而，现代 16 位机的设计却反映了整体系统设计中软件的重要性。此外，还注意到了系统设计者所发现到的 8 位机的局限性，并把由此提出需要具备的那些特点也结合进去了。很多数字设计工程师觉得从逻辑设计过渡到 8 位机系统是很容易的，因为这两者彼此相近，但是，过渡到现代 16 位机就不那么容易了，因为这两个领域并不相似。16 位机的术语看来很陌生。例如，异常事故、特权级别、存储管理、虚拟存储、协处理机、信标，以及很多其它新名词。本书的主要目的是阐明这些新特点，并且给读者有关它们是如何工作和应用的概念，以及有关它们的优点和局限性方面的论述。本书也讨论一些通信功能的知识，在不断扩展的信息技术的新领域中，通信功能已经变得日益重要，16 位机结合这种通信功能，构成了各种类型的信息处理设备。

作者对于在准备此书过程中给予帮助和鼓励的各位同事表示感谢。特别应感谢 C. J. Harris 教授，RMCS 的电子和电气工程系主任和电子部主任，以及 P. C. J. Hill 教授，感谢他们的气度和鼓励，也应感谢 Graham Turner 所进行的有益的讨论，感谢 A Hare 太太帮助打印部分书稿，并且感谢 RMCS 的执行院长 F. Hartley 博士，感谢他允许出版此书。

目 录

1 引 论

1.1 引 言

微处理机自七十年代早期开发以来，它不仅使电子学发生变革，而且使制造工业，甚至娱乐和家庭用品都发生了变革。其演变的速度是惊人的。只不过十年多的时间，微处理机器件已经经历了几代的演变。从 4 位的 Intel4004 开始，到现在微处理机已经站在小型计算机和大型计算机领域的边界线上了。凡是具有数字集成电路的半导体制造厂家，没有一家不想研制自己的微处理机或者支持其它厂家设计第二来源器件。其结果是，在当前工业标准的 8 位微处理机领域内，可供选择的产品有多种多样。与此同时，大家都竞相生产下一代的微处理机，即将要达到工业标准程度的 16 位微处理机。下面可以排出一个粗略的微处理机演进年（见表 1.1）。

很多工程师已经很熟悉 8 位微处理机了，无论对于通用 CPU 或者对于单片微控制器都是如此。这样使得从电子工程到微处理机应用的过渡进行得相当顺利，使吸收硬件技术的过程也相当顺利。然而，要自如地运用（大多数情况）汇编语言软件也许要花费稍微多一点时间。半导体制造厂家对子他们器件的支持程度，起初是很有限的，后来增长了，在开发系统设计和软件方面进行了巨大的投资。工业界已经认识到了 8 位微处理机所提供的机会，现在普通工程师看来已经很好地具备了有关 8 位微处理机的专业知识，但是很多人有点被“新型”16 位 CPU 的表面上的复杂性和不熟悉其特点所吓倒。从 8 位到 16 位微处理机应用进行过渡的问题并不是微不足道的。这里，在硬件、软件、语言、操作系统以及网

微处理机演进年表 表1.1

年代	说 明
1971	第一台4位微处理机诞生
1972	第一台8位微处理机诞生
1973～1975	采用当前工业标准的8位CPU及接口，紫外线可擦洗PROM，4K动态RAM
1976～1977	第一台8位单片计算机诞生
1978～1980	采用16位现代化微处理机，16K静态RAM，动态RAM可达64K，以及更大的EROMM
1980至今	宣布32位微处理机，第二代8位微控制器，若干第二代16位微处理机，电气可擦洗PROM(E^2PROM)

络方面都有很多新的概念。本书的目的就是要介绍这些概念。

下面将讨论一些典型的8位微处理机，以及支持它们的器件和软件。这部分篇幅是为了重温一下读者8位机的知识，同时也是为了与构成本书主体的16位机系统做一个比较。

1.2 8位微处理机硬件

8位微处理机领域的特点是它们具有相对的一致性。虽然可提供很多类型的8位微处理机，它们在速度和某些特性方面有差别，但是，除了一两个例外以外，它们的结构和设计思想都是相似的。当然，“工业标准”微处理机都是面向寄存器的，都具有相同的寻址范围，而且都具有大致相似的指令系统。一个典型的8位微处理机的框图如图1.1所示。

8位算术运算和逻辑部件（ALU）对于程序员可访问的寄存器或累加器中所持有的变量进行操作，然后把结果送回到一个寄存器中，并且根据算术运算逻辑指令的结果，把组合在一个8位条件码寄存器中的某位标志进行置数。时序则是由一个外面产生的时钟信号驱动的控制部件所执行。控制部件从一个指令译码器提取信号，而这个指令译码器是连接在8位指令寄存器上的。此外，还

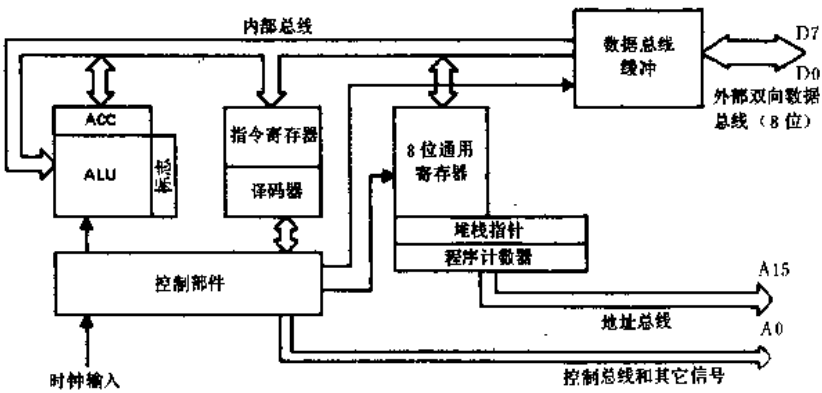

图 1.1　8 位 CPU 结构

可以提供一组保持数据和操作数的程序员可访问的通用寄存器。微处理机肯定还将具有某些具备专门功能的 16 位寄存器。尤其是，有一个程序计数器(或指令指针)，它是由控制部件自动增量来跟踪指令操作代码。程序计数器有一个堆栈指针，它是作为一个地址寄存器来支持堆栈的。堆栈是在读/写存储器中的一种数据结构，它是作为后进先出(LIFO)缓冲来控制的。此外还有间接地址寄存器，或许还有变址寄存器。所有这些功能块是用一个内部并行的 8 位数据总线通信的，这个总线采用时分多路转换。

微处理机通常放在 40 线封装中，这些引线所提供的信号通常包含有分开的并行的地址和数据总线，其宽度相应地分别为 16 位和 8 位。16 位地址总线宽度允许有 64K 字节的寻址范围。这一点很方便，因为地址是一个数据字的两倍宽度。当然，任何保存在存储器中的地址值占据两个字节。为了控制数据在总线上传送，CPU 控制部件产生一系列的控制信号。通常实行的总线控制是为了同步传送的，它是由 CPU 系统时钟专门引出的信号时序所控制的。在总线上数据传送需要两种不同的信息：方向信息(从存储器传送或者接口到 CPU，可认为是读操作，而那些方向相反的传

送则是写操作)；时序或选通信息(它控制什么时候一个选中地址的存储单元或设备应把其数据放在总线上，或者从总线接受数据)。微处理机可以使用方向与时序信息结合在一起的总线信号，或者也可以使用分开的方向与时序信号。这两种可能性如图 1.2 所示。那些通用类型处理机使用的可归并为 Intel8080(简称 8080 类型，其读操作与写操作，如图 1.2(a)所示。读操作一开始，先由 CPU 发出存储器或输入/输出单元的地址，后面加入 $\overline{RD}$ 控制线。作为对 $\overline{RD}$ 信号的响应，该设备将把数据放在数据总线上，CPU 将从数据总线上接受数据，接着是释放 $\overline{RD}$ 并除去地址。设备的访问时间必须配合 CPU 的总线时序。写操作开始与读操作一样，也是 CPU 先发一个地址，后面是 CPU 把数据放在数据总线上。当地址和数据稳定时，CPU 加入 $\overline{WR}$，选中地址的设备用它锁定从总线来的数据。

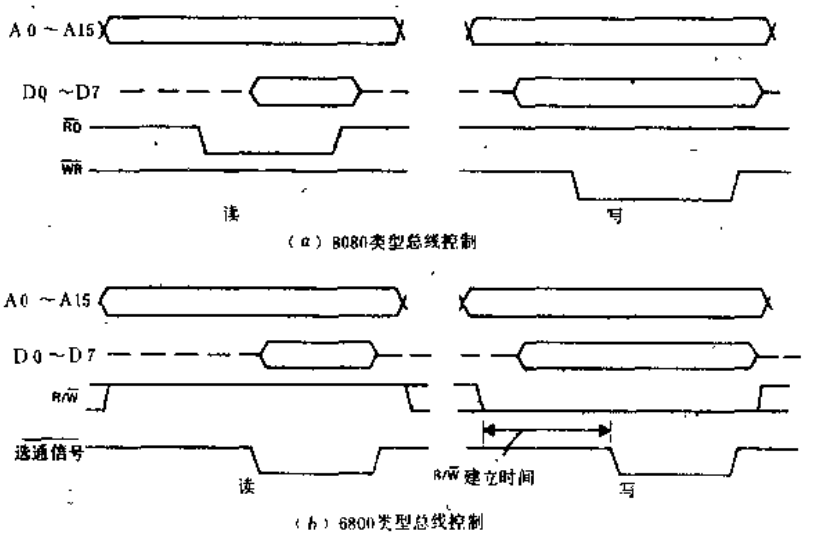

图 1.2 总线控制信号

另一类是 Motorola 6800(简称 6800)类型微处理机使用的控

制信号，如图 1.2(b)所示。读操作开始时，CPU 发出一个地址，同时把 R/$\overline{W}$ 控制信号变高。当地址与 R/$\overline{W}$ 信号稳定，CPU 加入选通信号 E。选中地址的设备对 E 信号加入所做的响应是，把它的数据放在数据总线上，该数据被 CPU 所接受。写操作开始时，先由 CPU 发出一个地址，与此同时把 R/$\overline{W}$ 线变成低电平，把数据放在数据总线上，一旦地址、数据和 R/$\overline{W}$ 信号稳定，后面加上选通信号 E。E 可以被选中地址的设备用作为一个选通信号，从总线上把数据锁定。对于 8080 类型的微处理机，任何设备必须去响应这个方向信息，如同是响应这时间的选通信号。而对于 6800 类型微处理机，方向信息（R/$\overline{W}$）相对于时间选通信号的“建立”时间，如同是对于地址信息一样。

和这些同步总线控制信号一样，有一个负向响应信号，READY(8080)或 $\overline{WAIT}$ (Z80)，可以使那些比 CPU 总线周期时序所包括的访问时间更长的存储器接口设备也能工作。这个响应信号并不是一个握手应答信号，因为它只是在选中地址的设备不能做出响应时才加入。如果设备已经能够及时地做出响应，则不提供正向响应信号。一般 CPU 对于这种 READY 或者 $\overline{WAIT}$ 信号是这么响应的，即在读或写总线周期中插入空闲或等待状态时钟周期，同时维持地址和控制信号稳定，这实际上是把总线周期延长整数个 CPU 时钟周期，直到 REAY 或 $\overline{WAIT}$ 信号去除为止。为了区分存储器操作与输入/输出操作，可选择两种可能的方案：第一种，在概念上说可能是“最干净”的，它对两种操作根本不做区别，所以所有输入/输出设备必须占据存储地址空间，相对应的是所有总线周期都由存储器访问指令产生。这种存储器对应输入/输出方案也意味着，CPU 指令系统没有必要包含专门写明的输入/输出指令。第二种处理输入/输出的方案是提供专门写明的输入/输出控制信号是 IO/$\overline{M}$ (8085)，或者是分开的 $\overline{IORD}$ 与 $\overline{IOW}$ (8080)。在这种情况下，输入/输出设备具有它们自己的地址空间

(I/O 地址空间)，它是用不同的控制信号来区分存储器地址空间的。输入/输出地址这样可以与存储器地址重叠起来，而输入/输出设备则需要有专门的指令，通常其助记符称为 IN 和 OUT。

所有 CPU 操作是由 CPU 时钟控制的，而每个基本总线周期或者机器周期(如读、写、输入/输出写或取指令等)是由若干个时钟周期组成的。每条指令至少由一个机器周期(如取指令)所组成，而存储器访问指令则要 n 个机器周期(如取指令、数据读或写等)。诸如 Z80 和 8080 微处理机要花若干个周期去执行每个机器周期，而对于 6800 类型微处理机，时钟周期与机器周期数目是一样的(例如，存储器读恰好只要一个时钟周期)。为了说明这种类型的时序，Z80 的一条典型指令，即输出(OUT)指令如图 1.3 所示。这需要三个机器周期(取指令，从存储单元读输入/输出端口地址〔8 位〕，这是在包含 OUT 操作码〔操作代码〕的单元后面的，然后把 A〔累加器〕寄存器内容传送给该设备)。注意这条特殊的 OUT 指令自动把一个 WAIT 状态插入到写机器周期。这种特点可以使速度稍慢的设备一起工作，而不需要任何外部产生 $\overline{\text{WAIT}}$ 信号的逻辑。这种可以自动延伸输入/输出读或写周期的方案是 Z80 微处理机所独有的。在很多 8 位微处理机中，还提供一个硬件信号(可能是一个编码状态信号的一个部分，与其它信号多路转换使用)去标明这个取指令周期的性质。它可以在仿真、跟踪和调试以及特殊总线周期(如中断响应)中使用。

总线请求与允准信号比较简单，它主要是为存储器直接访问(DMA)总线而设置的。典型情况下，从一个设备(如一个 DMA 控制器)发出一个 HOLD 或 $\overline{\text{BUSREQ}}$，它将使得 CPU 完成它当前的机器周期(或者可能是指令周期)，然后停止其操作，进入一个空闲状态，在这状态中它将只是维持任何内部数据和状态信息的刷新。一旦 CPU 完成这个周期，它将交出外面的总线(地址、数据和控制)，并把总线浮空为三态高阻状态(所有 CPU 内部总线驱动器有三种逻辑状态：平常“0”接近“0V，“1”在 5V(CPU 情况下最小为