微计算机设计教程

C.A.Ogdin 著

微型电脑编辑部

一九八一

·微型电脑丛书·

微计算机设计教程

〔美〕C.A.Ogdin著

微型电脑编辑部

Carol Anne Ogdin

Microcomputer Design Course

据*EDN* Vol.21,No.23 译出，又据 *Microcomputer Design*, Prentice Hall Inc.1978 增订

内容简介

本书在介绍微计算机概念、结构、软件特征的基础上，系统地介绍了将芯片组装成微计算机系统及配以软件的方法。它与一般教程的不同之处是：它出自具有丰富经验的、从事具体设计工作的专家之手，所谈多是具体的设计步骤、技巧及经验，内容很是丰富，实用。本教程在EDN杂志发表后，被列为美国商务出版奖的候选佳作。

·微型电脑丛书①·

微计算机设计教程

周继武　译

黄　泰　校

*

微型电脑编辑部

（湖南株洲电子研究所）

1981年1月第二版

定价：1.50元

《全国微计算机用户录》

刊印说明

微计算机于七十年代问世之后，国外已迅速推广、普及，被称为电子学的一场革命。我国微计算机发展的高潮也即将到来，并将大大促进我国的四个现代化。

为此，中国科学院声学研究所于一九八〇年倡导成立《全国微计算机用户协会》，以便推广微计算机研究与应用的新成果、新经验，沟通微计算机应用的协作，提供引进国外微计算机技术与装备的情报，报导协会成员单位举办微计算机技术培训班、学习班的消息，以及内部进行资料交流和组织编译有关的资料。这一倡导已获得热烈的支持。作为该协会的活动之一，《微型电脑》编辑部拟于八一年中刊印一册《全国微计算机用户录》，刊印项目是：单位名称，通信地址，电话号码，现有（指生产/研制/进口）机种及配套情况，软件配套情况，研究与应用状况，今后计划，可提供资料，等。

举凡与微计算机的研制、生产、教学、应用有关的研究所、工厂、大专院校（不管是否已参加“全国微计算机用户协会”），均欢迎刊入。**欲刊入者请于八一年五月底之前按上述项目来函（勿超过三百字）湖南株洲电子研究所《微型电脑》编辑部。**凡愿参加“全国微计算机用户协会”者，亦请注明。

本通讯录还收集有国外及港澳的微计算机公司、商行的产品系列型号、技术指标，以供参考。

《全国微计算机用户录》初定每两年重版一次。第一册定于八一年七月出版。刊入者赠书一册。

中国科学院声学所《微计算机应用》
湖南株洲电子所《微型电脑》编辑部

一九八一年一月

目　录

序

本书与一般教程有所不同。它假定你就要设计微计算机，预料到你会有哪些问题。它把其他设计师成功地设计微计算机和微处理器的经验告诉你，指导你如何把这些经验用于你的日常工作。这里几乎没有什么理论，只谈实际技术和指导，重点是如何使用微处理器，而不是详细说明它们的原理与制造。

电子与计算机的出版界已出版过许多关于微计算机的书籍，为何又要出这本教程？迄今很少有书是介绍具体的设计者的实际经验的，唯有此书是个明显的例外。而且，此书是经过“实地检验”过的——其三分之二的内容曾在EDN杂志上发表过，数百名读者来信表示赞许，并被推荐为Jesse Neale奖（商务出版奖）的候选书。

然而，本书的真正目的并不只是重印那些极其成功的文章，而是要给这不断进展的领域增添更适用的内容。我的目的是给那些已经习惯于常规的数字逻辑系统的设计师们指明如何采用微型机来求得更佳的设计效果。大多数读者都已很好地掌握了TTL和CMOS技术了，但他们却从来接触过计算机和程序设计。在本书中我着力说明，熟练的设计师们用于数字逻辑设计中的那些原理，在设计用微计算机来解决有趣而困难的问题时仍然是适用的；而且，我认为，软件也不过是数字设计的另一个分支，尽管它被新的符号和词汇裹成另一番

模样。

本书按逻辑进展由浅入深分为十一章。

第一章介绍了微计算机的基本概念，为后面的内容提供基础。从基本原理讲起，读者容易理解后面各章所介绍的设计细节。计算机实际上只不过是个无固有用途的数字逻辑系统，我们就以所有数字设计师都已了解的原理为本书的起点。

第二章说明微计算机基本部分的连接。数据总线，地址总线及控制总线是所有计算机设计的核心，这里予以总的介绍。总线事件的定时也很重要，因此与时钟一并讨论。

使用微计算机一般需要些实际动手的经验。第三章介绍了设计师为了获得基本经验而要用的学习工具的一些特征。

从第四章起，是介绍硬件(即与电路有关的)和软件(程序设计)方面的内容。软件的首要方面是指令在计算机里如何工作，这在第四章里讨论；该章还说明了各种常用的微型机如何操作以及本书出版后肯定会出现的新的微处理器的有关情况。这里重点讨论了组成小程序的硬件指令的使用。

计算机少不了存贮器，第五章讨论存贮器。但讨论的只是实际设计中使用的存贮器，而不是所有的存贮器。重点是如何用现有的便宜的存贮器件来贮存数据和程序以及缓冲，地址译码等实际问题。

存贮器的利用是个软件和程序设计问题，第六章论述这个问题。根据我二十余年的程序设计经验，我认为最好的设计是在程序设计之前就确定好存贮器的使用模式。该章详细说明了一些特殊的设计问题的解决办法，帮助读者重温其他应用领域的概念。

第七章讨论输入输出的设计。虽然微计算机使许多数字电路设计师的技巧过时了，输入输出电路却可以难住最有创见的设计师。在说明了微计算机和外部设备之间执行输入输出数据传输的各种可能的方式之后，再介绍一些一般人感到为难的问题的设计方法。

输入输出硬件需要软件来使它工作，这种软件放在第八章里讨论。它表明，一个具有多重输入输出，每个输入输出装置均是实时操作的复杂的计算机系统所要求的软件也是可以有条有理地设计出来的。中断的处理及时间相关事件的控制专门作为另一套技术方案加以讨论。

硬件和软件在微计算机系统中都是必不可少的部分，两者之间的权衡取舍很是重要，第九章就讨论这种权衡。通过比较几个分别采用硬件和软件来执行同一功能的例子，让读者借鉴一下其他设计师是如何解决这种问题的。

第十章介绍软件研制所需的工具的特征，涉及汇编程序，编译程序，编辑程序及其相关的语言。

最后，第十一章，是微计算机的设计规划。这一章贯通了典型的设计规划步骤，说明了需要的资源，各阶段潜在的问题及解决办法。

这些篇章所介绍的都是象我这样的设计师日常所用的设计技巧，我希望它们对你也会很实用。一旦你理解了本书的思路，我相信你就一定能轻松愉快地得出结构合理，没有不必要的复杂性的设计方案来。

Carol Anne Ogdin

第一章 微处理器与微计算机系统的定义

本章介绍了通用/贮存程序/数字计算机（微型、小型、大型）的基本部分

任何数字系统，不管是通用计算机还是专用电路，其最基本的部件是寄存器——能容纳一般用数字表示的值的物理器件。寄存器的最简单形式可以仅仅是代表信息的单线。

比较常见的寄存器由并行闩锁组成，它所代表的数字的范围是0到2N—1，其中N为并行闩锁数。当寄存器要记录输入，时钟就发出信号，末次闩锁的数据就在输出端出现。

寄存器无所不为

所有数字系统可以设计成一组内部连接的寄存器，通过逐步地、选择性地将一个寄存器的内容传向另一寄存器，获得所需要的系统性能。

数据经由某种寄存器时也可以变换。就最简单的情况来说，可以看看并行移位寄存器。不管是由多组的多路转换器组成的，还是仅仅由几根导线组成的，数据置入这种寄存器，读出之前都经修改。修改的内容随应用的不同而不同。

当置入寄存器的数据“左”移一个位置，最高位如何处置？将最高位馈回最低位谓之循环操作。另一种设计是，最高位移出、消失，此时仍有相异的设计，或者将最低位清零，或者，有时候，让它保留原始输入值。须知这个寄存器的最简单的例子也有多种多样的微妙变化，这种种微妙变化

使微处理器与微计算机也随之有根本的差异。

所有计算机“都一样”

每个计算机由五个必不可少的基本部分组成（图1）。但是，并非凡具这五个部分的系统就都是计算机。计算器也具有这五个部分，但却根本不成为计算机。

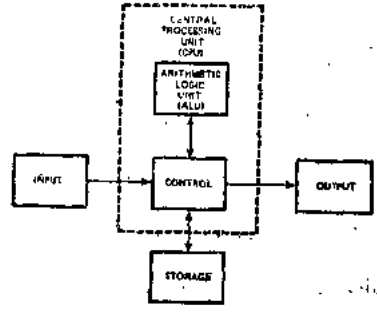

图1 所有微计算机都包括这五个基本部分，其CPU（虚线框内的）一般是个单片微处理器

运算器（ALU）是计算机的数据变换部分。它的输入数据按特定方案改变。存贮器为下一步使用贮存中间值，可把它看作是一个庞大的算草本。输入输出是接受有待处理的数据和为使用提供处理过的数据的部件。没有输出，计算机根本没用。没有输入，计算机则只有一遍又一遍地重复单一定序（这一点在某些定序控制应用中还是有用的）。最后，上述四个部分在控制部件的安排下运行。控制器决定计算机内哪个寄存器按哪种顺序执行传输。所以，控制部件和特性决定了计算机的全部特性（及结构）。

顺序性有什么用处呢？业已证明（1936年），只要给足够的时间，寄存器之间的适当的传输顺序可以解出有收敛解的任何问题。这也即是意味着，这对每一项新应用来说都无需作专门的硬件设计。这正是数字计算机的力量和前途所在：它是一种通用的机器，能解决许多问题，这些问题如果——需要专门的硬件，那就会太昂贵无法实行。

计算机用户也发现计算机不必单纯伴随数字工作，这些数也可以代表非数字数据。比如，数41可代表A，42代表B，等等，现在，每个寄存器里都存有既可解释为数字（如41）也可解释为符号（如“A”）的数据。这一点，说明了数据与信息的基本区别。数字41是数据，它可以代表某个操作需执行的次数，或一个电阻值，或者字母A。当纯数字附上了函义，数据就成了信息。所以，信息是有函义的数据。计算机处理数据，人处理信息。许多有关理解计算机能与不能的问题都集中在这个原则上。

寄存器的传输确定了微计算机

微计算机由寄存器组成，其结构特征取决于寄存器之间可完成的不同种类的传输。同样，计算机的子系统也由寄存器组成。

例如，存贮器即是数个（一般都是同样的）寄存器组合而成的（图2）。每个寄存器可分别选择。这样，不管内容和目的如何，存贮器都可看作是一个可选择的寄存器组。这

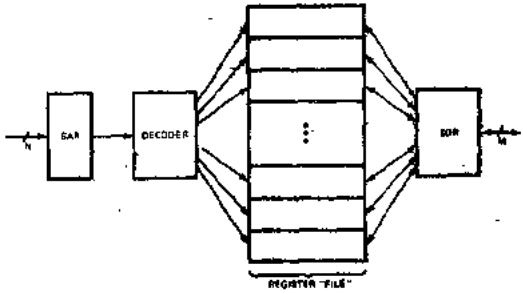

图2任何随机可寻址存贮体的基本结构必须有分别选择每个寄存器并将其内容传输到微计算机的其他部件的方法。

些寄存器是通过另一个寄存器选择的。

在访问存贮体的特定寄存器时，标示该寄存器的数字地址被放入存贮器地址寄存器(SAR)。SAR一般有N位输入容量，由此产生2N个中的一个输出。这一译码功能不过就是个复合寄存器。在从存贮体读出时，所选的寄存器里的数据送入存贮器(SDR)。

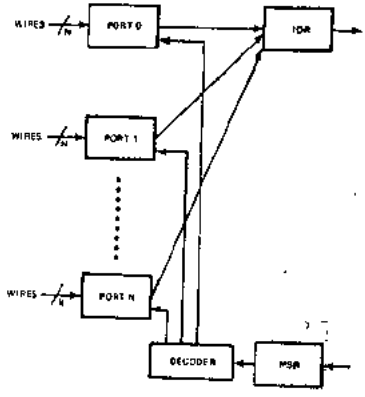

图3 输入口如存贮器一样选择。存贮器与入口寄存贮器唯一区别是数据如何存放，这对选择过贮是无关紧要贮。出口与此相反。

那么，从一般意义上说，计算机的存贮体包括寄存器（存贮单元）的一个阵列和一对连接寄存器（SAR和SDR）。SDR的宽度为存贮体的字长。

输入输出都可看作是存贮器系统的扩充，主要差别只是选中的寄存器对于外界的可达性。为了输入数据（见图3），某些入口选择寄存器（IPSR）——另一种译码器——只选中一个寄存器，然后将单一的输入数据寄存器（LDR）的内容送往计算机的控制部分。数据的输出与此过程相反。

运算器（ALU）是另一组寄存器，数据在寄存器之间传输时，它执行所有的内部运算。在外部，当数据送到运算器的输入寄存器，需执行的特定功能（一般用二进制数表示）同时置入运算器的功能寄存器(图4)。运算器一般对成对的寄存器内容进行运算，将一个数加到另一个数上需要二

个数据值，这两个值应可以同时访问。

常与运算器相连的是一个单寄存器，谓之累加器。每次运算，两个运算数中的一个常常就放在累加器中。累加器是存贮运算结果的寄存器。累加器的大小决定计算机的字长。8位微计算机具有8位宽的累加器。一般说来以并行方式处理累加器的内容，但也不总是如此。

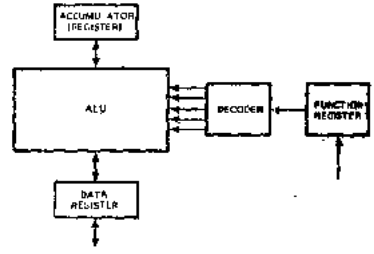

图4虽然只不过是一组寄存器组，运算器却同时使用其中几个寄存器处理数据，其位宽及寄存器传送种类可表明微计算机的能力。

控制部件指挥微计算机

计算机的控制部件将所有这些各不相同的寄存器连成一体，执行需要的一组任务。由于所有可能发生的寄存器传送方式都在这个中心部件的控制之下，所以必须有某种方法来指明进行何种传送。这种“指明”就是所谓“指令”。计算机能接受的所有可能的指令的表称为指令集。

典型的指令可以选择一个特定的输入寄存器，然后将其内容传向特定的存贮寄存器。这个寄存器传送包含一系列较小的寄存器传送，它们不在程序的直接控制之下。计算机的控制部件决定这些内寄存器传送，或微指令。

一般控制部件自身有一个小的只读存贮器，对指令集中的每个指令它包含一个或几个字，这些“微”字依次组成为了微观地控制各个内寄存器传送所需要的特定的内信号，以使指令得以完成。如果这个只读存贮器可以修改，计算机就

具有微处理器的微可编程序性。大部分微处理器只能在工厂里编制微程序。

计算机的指令顺序决定某特定问题的具体细节。这些指令放在某个存贮体中。目前大部分微计算机将指令存贮在存放数据的同一存贮体中。

程序计数器（PC）寄存器（图5）选择要从存贮器中取出的特定指令并执行之。为了取指令，PC的内容首先需送入存贮地址寄存器。当寄存器选择的物理过程发生时（称为访问时间），程序计数器就增1，准备在完成第一个指令时取邻接的下一个指令。在存贮体产生所选择的寄存器的内容之后，控制部件将存贮器据数寄存器的内容输入指令寄存器。

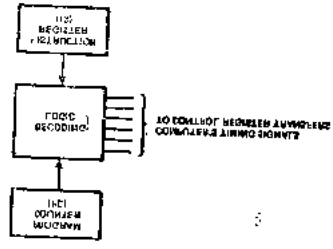

图5 微计算机的“心脏”，控制部分，决定必须发生的动作（寄存器传送）。它根据用户送入的指令序列及先前操作的结果作出决定。

指令寄存器可以是拥有大量的“随机”逻辑的复杂类型，也可以是对指令进行译码的只读存贮器装置的存贮地址寄存器。不论哪一种，结果都一样：控制信号的序列有选择地将某些寄存器的内容传向另一些寄存器。指令的结尾即下一步取（指）和执行操作的开始。如此连续进行。

将所有不同的部件连接起来，即成图6所示的一种假想的计算机的总结构。当然，这里我们过于简单化了。比如，许多控制信号必须与某一主钟同步发生。但是，我们不妨将

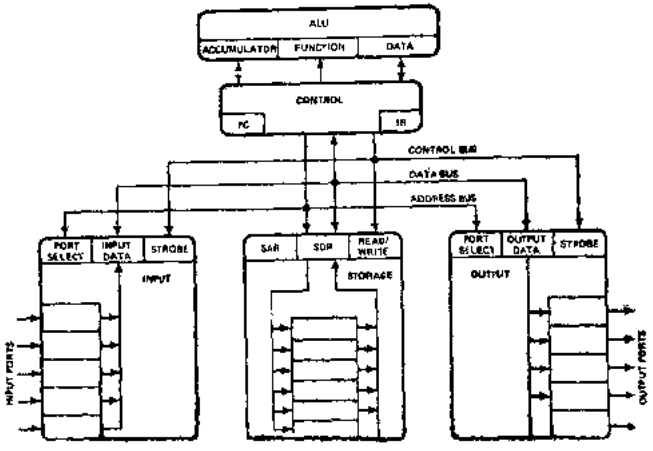

图 6 所有计算机都可表示为寄存器的组合。这个假想的计算机比较简单，没有特殊的控制信号（1 位宽寄存器）和增加效率和速度并使计算机互有区别的特殊寄存器。

这些控制信号作为 1 位宽的寄存器进行传送，正如其他的寄存器传送一样。

另一方面，还可在这种计算机上增添更多的具有特殊功能的寄存器。比如，控制器就可以包括一组“局部”寄存器，用来寄存中间数据而无须涉及存贮器。这种寄存器的好处在于速度（因为不包含存贮器访问时间）和效率（局部寄存器可用指令中的很少几位来选择而不用存贮地址寄存器那么宽的地址）。

就是总线也可看作寄存器

计算机部件的内部连接几乎总要采用“总线”概念。总线，实际上就是一种具有被所有使用元件公认的单一格式的

USER INSTRUCTION	MICROINSTRUCTION OPERATIONS (REGISTER TRANSFERS)	COMMENT
LOAD "17" INTO ACCUMULATOR	SAR ← PC	ADDRESS THE INSTRUCTION
	PC ← PC + 1	INCREMENT PROGRAM COUNTER
	IR ← SDR	FETCH THE INSTRUCTION
	SAR ← PC	FETCH THE SECOND WORD OF THE INSTRUCTION (CONTAINS "17")
	PC ← PC + 1	
	ACC ← SDR	PUT DATUM INTO ACCUMULATOR
ADD CONTENTS OF LOCATION 81 TO ACCUMULATOR	SAR ← PC	ADDRESS THE INSTRUCTION
	PC ← PC + 1	INCREMENT PROGRAM COUNTER
	IR ← SDR	FETCH THE INSTRUCTION
	SAR ← PC	FETCH THE SECOND WORD OF THE INSTRUCTION (CONTAINS "81")
	PC ← PC + 1	
	SAR ← SDR	ADDRESS THE LOCATION HOLDING THE DATA
	ALU ← SDR	GET THE DATA
	FUNCTION ← ADD	START THE ALU IN OPERATION
STORE THE RESULTS NOW IN ACCUMULATOR INTO LOCATION 70	SAR ← PC	ADDRESS THE INSTRUCTION
	PC ← PC + 1	INCREMENT PROGRAM COUNTER
	IR ← SDR	FETCH THE INSTRUCTION
	SAR ← PC	FETCH THE SECOND WORD OF THE INSTRUCTION (CONTAINS "70")
	SAR ← SDR	ADDRESS THE LOCATION TO RECEIVE THE RESULT
	SDR ← ACC	PUT DATA INTO STORAGE

NOTE:
Y ← X MEANS PLACE CONTENTS OF REGISTER X INTO REGISTER Y

SAR STORAGE ADDRESS REGISTER
SDR STORAGE DATA REGISTER
ACC ACCUMULATOR REGISTER
ALU ALU OPERAND INPUT REGISTER
FUNCTION ALU FUNCTION REGISTER
PC PROGRAM COUNTER OF CONTROL SECTION
IR INSTRUCTION REGISTER OF CONTROL SECTION

表 1

注：Y←X意指将寄存器的内容置入寄存器Y，SAD：存贮地址寄存器；SDR：存贮数据寄存器；ACC：累加器；ALU：ALD操作输入寄存器；FUNCTION：ALU功能寄存器；PC：控制部分程序计数器；IR：控制部分指令寄存器。

抽象寄存器。大部分微计算机具有三种总线：地址总线，数据总线和控制总线。CPU一般发出其他系统元件使用的地址和控制总线信号，而CPU与其他元件之间的数据交换采用的是双向数据向线。当然，也可以将双向总线组织成二条单向总线，但是单一的双向总线可减少数据传输所需的插脚数。

计算机的字长可以指存贮器中的可寻址单元的宽度，也可指累加器或数据总线的宽度。由于计算机元件之间的联系的重要性，如果字长的这三种定义不同，则一般以数据总线

的宽度为主。

分析提供的是哪种寄存器和存器之寄间传输数据所采用的方式，即可最清楚地看出各种计算机之间的差异。计算机的花样越少，计算机就愈易于理解。但同时，只有很少几种寄存器传输方式的简单的计算机，其效率也差。反之，具有各种不同的专用的寄存器的复杂的计算机往往又难以理解。但是，一旦理解了，这些计算机的效果就非常好。因为它们完成同样的工作只需很少的指令。

较大程序中的一个很小的指令组的实例（见表1）可以说明所有这些原理。给定的程序（即说明做什么的指令的汇集）是很简单的，它只说："在存贮器61号单元的寄存器的内容上加17，将结果存入存贮器70号单元"，这一般程序由三个指令构成，在别的计算机可能多于三个，也可能不到三个。

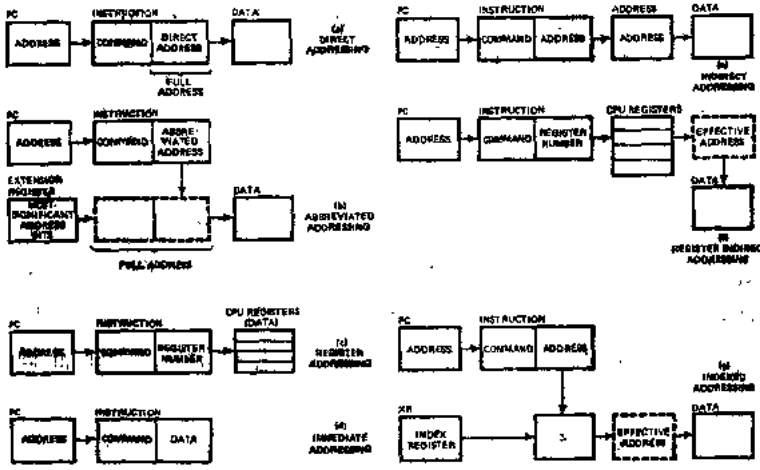

图7 每个微计算机并不具备所有的寻址方式，寻址方式的不全意味着为了执行本来是很简短的任务，微计算机却必须使用较多的指令。