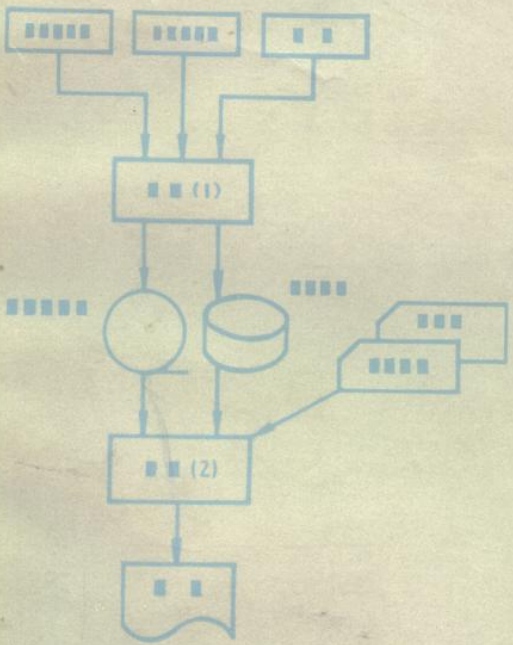
微程序设计

〔日〕萩原 宏 著



科学出版社

微程序设计

〔日〕萩原 宏著

費志浩 译

科学出版社

内 容 简 介

本书深入浅出地叙述了微程序设计的各种方式和主要应用．全书共九章，首先介绍了微程序设计的发展过程和一般概念，接着讲述微程序控制的各种方式以及微程序的变更技术和编制技术，并在此基础上分别介绍了微程序在仿真、故障诊断、语言处理、系统程序及数值计算中的应用．章末和书末附有丰富的参考文献．附录中列出了一些采用微程序控制方式的计算机实例．

本书可供从事计算机硬、软件设计的科技人员及大专院校有关专业的师生阅读和参考．

萩原 宏

マイクロプログラミング

産業圖書株式會社 1977

微程序设计

〔日〕萩原 宏 著

费志浩 译

责任编辑 李淑兰 那莉莉

科学出版社出版

北京朝阳门内大街137号

中国科学院印刷厂印刷

新华书店北京发行所发行 各地新华书店经售

*

1982年6月第一版 开本：787×1092 1/32

1982年6月第一次印刷 印张：10 3/4

印数：0001—9,550 字数：242,000

统一书号：15031·401

本社书号：2554·15—8

定价：1.65元

序　言

自微程序设计问世以来，已过去了四分之一个世纪．现在，微程序设计技术已经确立，并被广泛地应用于各个方面．

最初，微程序设计方式是作为系统地组织和设计计算机控制部分的一种手段而提出来的．当时，这种技术仅仅作为硬件的一种设计方式而加以发展．

在微程序设计方式中，处理器的控制逻辑以程序的形式集中地存放在控制存贮器里，这样易于变更和修正，所以它具有一个显著的优点，就是能给处理器的硬件带来灵活性．特别是随着可改写控制存贮器这一技术的出现，这个优点就更为突出，它引起了各方面的关注，不仅仅硬件技术人员，软件技术人员和计算机系统的用户也对此产生了兴趣．固件日益受到重视，微程序获得了广泛的应用．这就是说，微程序设计已经不只是硬件的设计手段，它还被用作仿真和故障诊断技术，以及应用于系统程序和各种应用程序．原先由软件完成的部分功能正在逐步地变为由固件实现，微程序设计技术不仅对硬件，而且对软件和各种应用产生了极大的影响．

因此，在微程序控制方式计算机的硬件设计中，必须充分考虑在软件方面利用微程序，同时在软件设计中，应利用微程序这一技术编制出高效率和高性能的程序．换言之，在设计计算机系统时，希望在硬件、固件和软件这三者之间获得最佳的功能分配．因此，不仅对硬件技术人员，而且对软件技术人员或用户来说，微程序设计已经成为一项重要的技术．

本书从这种观点出发，叙述了微程序设计的各种方式和主要应用．并假定读者已具有计算机硬件和软件的基本知识．

本书力求写得通俗易懂，使得没有高深专业知识的读者也能充分理解.

微程序设计的方式依计算机的不同而千差万别，所以本书不以特定的计算机为对象，而尽量概括作一般的叙述；同时，对各个问题分别举出具体例子来加以说明.

本书的第1章，扼要地回顾了微程序设计的发展过程，介绍了有关微程序设计的一般概念.

第2章综述了微程序控制的各种方式. 第3章和第4章分别讲述有关微程序变更的技术和微程序的编制技术.

从第5章起的以后各章介绍了微程序设计的主要应用，即仿真、故障诊断、程序设计语言的处理以及在系统程序和数值计算中的应用. 在介绍过程中，结合具体实例作了说明.

各种具体计算机的控制方式和结构未在本书正文中叙述，而是作为附录，简扼地列出了KT试验型计算机的控制方式及国内外一些商用计算机的硬件结构和微指令形式，以帮助读者了解微程序控制方式的具体内容，加深对正文的理解.

在撰写本书的过程中，参考了许多国内外资料，总结了著者的研究成果，并结合自己的看法作了论述. 但是，微程序设计是涉及计算机硬件的技术，其详细内容很少公开发表，有不少地方还不十分清楚，希望今后有机会时再行补充. 另外，有些关于细节的说明过于冗长，而篇幅有限，故予略去，请原谅.

本书之所以能完稿，主要依靠在计算机研究方面，特别是微程序设计领域的导师和前辈的指导，在此谨表衷心感谢. 此外，向为本书提供宝贵资料的有关人士，向给予本书执笔机会的高桥秀俊和石田晴久两先生以及承担编辑工作的产业图书编辑部，深致谢意.

萩原　宏

1977年3月

目　次

1. 绪　论

1.1　微程序设计的概念

1951 年,英国剑桥大学的 M. V. 威尔克斯(Wilkes)提出了微程序设计的概念[1,2],他把这种概念作为系统地确定和设计计算机控制部分的方法.

存贮程序方式的计算机由五个功能部分组成,如图 1.1 所示,通常将运算和控制这两部分合称为中央处理机(CPU).

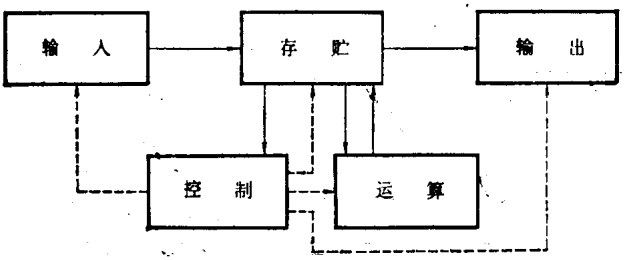


图 1.1　计算机的各个功能部分

控制部分的结构如图 1.2 所示. 机器按照指令计数器所指定的地址,依次读出存放在主存中的各条指令,并送往指令寄存器. 译码器解释指令的操作码,然后由控制电路依次发出为执行该指令所必需的控制信号,这些控制信号对计算机的各个部分进行控制,以完成程序规定的操作.

假定运算器的结构如图 1.3 所示,已存放在指令寄存器中的指令是加法指令(该指令地址部所指定的主存地址单元

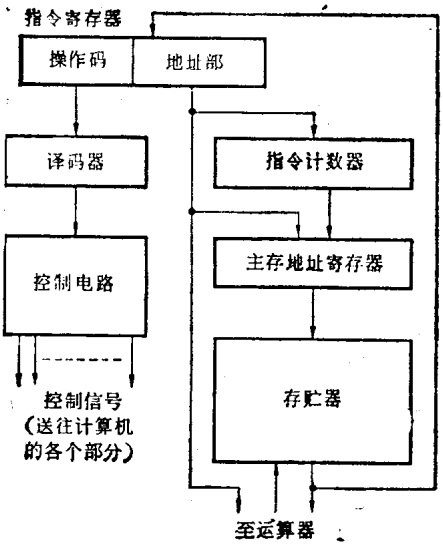


图 1.2　控制部分结构简图

的内容与寄存器 A 的内容相加，得到的和送入寄存器 B)．为了执行这条指令，必须完成下列操作：

a．把指令寄存器地址部的内容传送到主存地址寄存器．

b．从存贮器读出数据(READ 信号)．

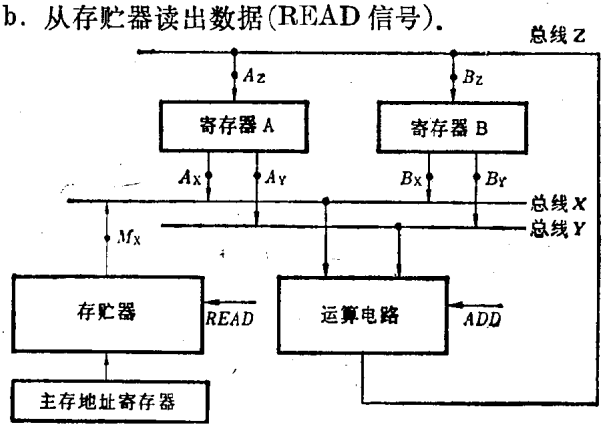


图 1.3　部分运算器的示意图

e. 将读出的数据送到总线 X(开门电路 M_X).

d. 将寄存器 A 的内容送到总线 Y(开门电路 A_Y).

e. 命令运算器做加法(ADD 信号).

f. 运算的结果经总线 Z 送往寄存器 B(开门电路 B_Z).

这样，加法操作就结束了，但为了按程序一步步处理下去，还必须继续操作，即从存贮器读出程序的下一条指令.

计算机的指令就是通过组合上述基本操作(称之为微操作)而执行的，控制电路必须依次发出控制这种微操作的信号. 过去，人们一直采用硬联逻辑方式的控制电路(不采用微程序设计方式)，这种控制电路虽然也能按一定的时序输出控制信号，但它的设计和结构都很复杂.

因此，对于由同时发生的控制信号而执行的一组微操作，设想了一种微指令. 微指令就是把同时发生的控制信号的有关信息汇集成指令的形式，故计算机的指令操作可表达为这种微指令的序列. 计算机的程序由指令序列构成，与此相对应，计算机的指令也可看作由微指令序列构成，这种微指令序列即称为微程序. 将微程序存贮在适当的存贮器(称为控制存贮器，以下简称为控存)之后，即可依次读出微指令来控制微操作，以此执行计算机的指令. 这种控制方式叫做微程序控制方式或微程序设计方式. 也就是说，采用微程序设计方式的计算机，象图 1.4 所示的那样，用各个微程序去对应各条机器指令. 微程序存贮在控存里，当调用程序的指令时，就由相应的微程序进行控制，以此执行该指令.

微程序设计方式的控制部分结构如图 1.5 所示，控制器的设计相当简洁. 就是说，只要将计算机指令应执行的操作分解为微操作，再以微指令序列的形式将其编成微程序，然后存入控存就行了. 这样，计算机的控制逻辑就转化为存于控存的微程序，而控制部分的逻辑设计就变成以编制微程序为

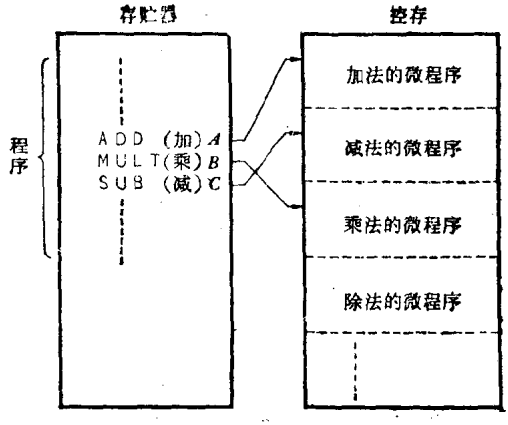


图 1.4 对应于机器指令的微程序

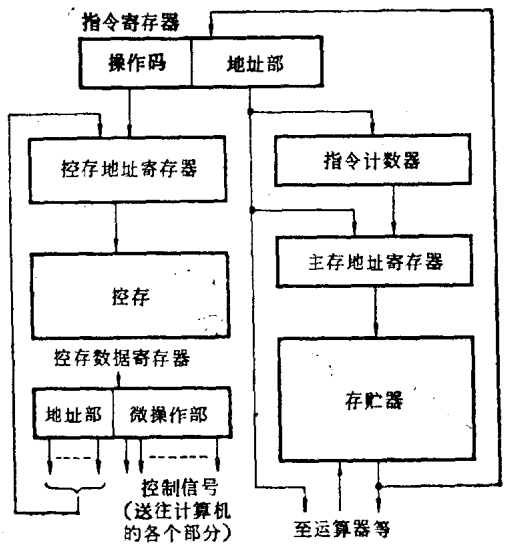


图 1.5 微程序方式的控制部分结构

中心的工作,使设计更加直观.

此外, 如果使用可改写内容的存贮器作为控存, 那末

只要改变控存中的微程序，便能简单地改变计算机的控制逻辑．因而，计算机的机器指令也可按不同的目的加以变更，于是开辟了仿真技术等各种各样的应用领域．

微程序设计最初是作为计算机控制部分的一种设计手段而提出的，后来在 IBM 360 系列的计算机中，被巧妙地用来实现该系列各机种之间体系结构的兼容．现在已很清楚，由于微程序设计方式的硬件能比较容易地变更和扩充功能，所以，可以利用这种灵活性来实现仿真，或有效地实施故障诊断等．利用微程序还能提高处理效率并且可构成适合于使用目的的系统等等．总之，在今天，微程序设计已不仅对硬件的设计，而且对包括软件在内的整个计算机系统都有着很大的影响．

1.2 微程序设计的发展过程

自微程序概念问世到现在，其发展过程可以分为三个时期．第一时期是 1951 年至 1964 年初，这段时间提出了微程序设计的各种设想，也试制了不少计算机．第二时期确认了微程序设计的优点，并以 IBM 360 系统为首，广泛地在商用计算机上应用．第三时期从进入七十年代开始，随半导体集成电路技术的不断进展，已能提供廉价的高速半导体存贮器，使可改写的控存实用化，同时也出现了运用微程序设计的各种应用领域．

下面就按这三个时期，概要地回顾一下与微程序设计发展有关的主要情况．

(1) 第一时期(1951—1964)

威尔克斯在 1951 年提出了微程序设计概念[1]，这种概

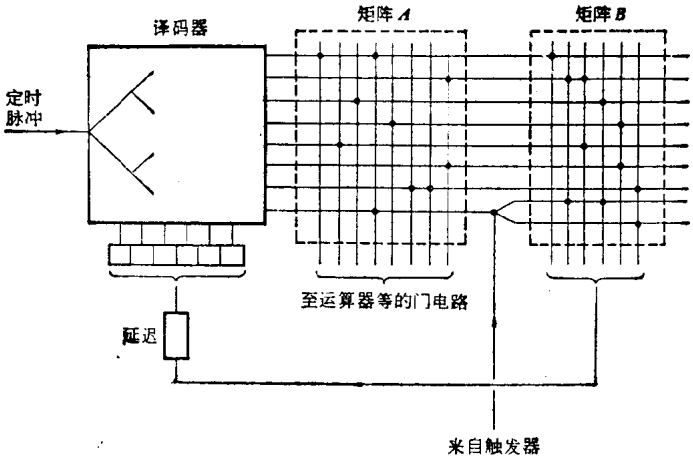


图 1.6　威尔克斯最早的微程序控制模型

念被作为有规则地设计计算机控制电路的方法. 在这篇论文中,他提出了如图 1.6 所示的模型,采用图中所示的由两个二极管矩阵组成的固定存贮器作为控存, 图中的每条水平线对应于一条微指令,矩阵 A 的垂直线是控制信号线,它连接到运算器一类的门电路. 两线交点上画有黑圆点的地方表示该处存在二极管的耦合. 此时,由译码器选择一条水平线,等一旦送入定时脉冲, 矩阵 A 仅在有二极管耦合的垂直线上有输出, 用其执行相应的微操作. 矩阵 B 的输出则经过一个延迟电路后送入地址寄存器, 以选择下一条微指令. 在矩阵 A 向右延伸的各条线中, 有一条在进入矩阵 B 之前产生分支. 此分支由触发器送来的信号控制, 用于选择后继微指令. 换言之, 通过控制触发器的条件, 可以变更后继微指令. 如有必要, 在译码器和矩阵 A 之间也可以有分支, 从而通过触发器的条件变更微操作本身.

根据这种想法, 美国剑桥大学制造了名为 EDSAC II 的计算机[3,4]. 这台计算机的控存不用二极管矩阵,而使用有选

择地穿有导线的铁氧体磁心矩阵. 当选定并驱动矩阵中的某一个磁心时,穿过该磁心的导线上产生感应电动势,其输出用作控制信号,而不穿过该磁心的导线上则没有输出.

威尔克斯等人对最初的模型作了修改, 提出了图 1.7 的方案[4]. 即矩阵 B 的输出送入寄存器 S, 它的内容再由脉冲 I_2 送往寄存器 R. 在送入输入脉冲 I_1 后,经过适当的时间再送入脉冲 I_2, 使得下一个输入脉冲送入译码器之前, 寄存器 R 能及时换上新的值.

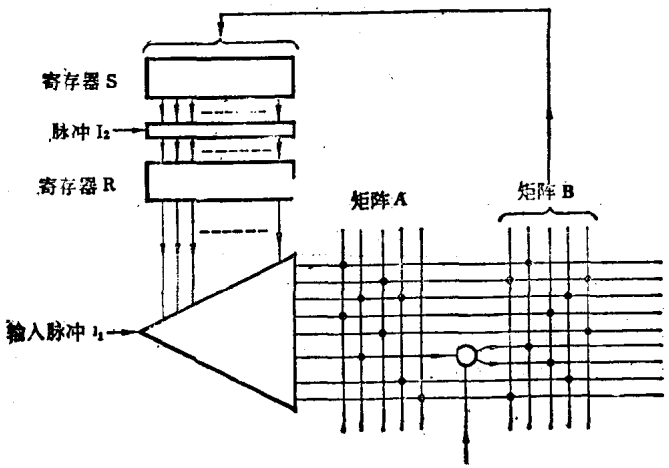


图 1.7 图 1.6 模型的修改方案

此外,译码器也可一分为二,结构如图 1.8 所示. 在两个译码器的输入端交替地加送脉冲,可以提高操作速度,但这样做的缺点是使程序变得复杂.

在威尔克斯等发表微程序设计的概念以后, 自 1955 年起,人们对微程序设计进行了许多探讨,并且论及微程序的改进[5,6]. 布兰肯贝克(Blankenbaker)提出了与威尔克斯不同的想法, 他在非常简单的计算机上实现了更为基本的逐位操作[7]. 范德波尔(van der Poel)也发表了一种简单的计算机