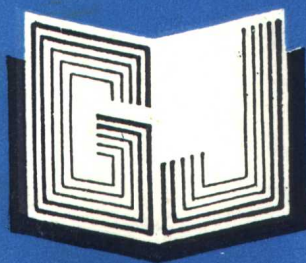


高等学校教材
微机在汽轮机中的应用



高等学校教材

微机在汽轮机中的应用

华中理工大学 陈 旭
东南大学 周 曦 合编
华中理工大学 黄树红

TK267-08



高等学校教材

微机在汽轮机中的应用

华中理工大学 陈 旭
东南大学 周 曦 合编
华中理工大学 黄树红

水利电力出版社

(京)新登字115号

1R

内 容 提 要

本书主要内容有微型计算机的基本结构及工作过程, Z 80的指令系统, 模拟量输入输出通道结构及接口技术, 过程通道的抗干扰措施, 汽轮机运行参数的微机巡回检测, 微机对汽轮机常规性能的计算和监视, 微机在汽轮机优化运行中的应用, 汽轮机设备的状态监测与故障诊断, 大型汽轮发电机组的计算机监控系统等。

本书可作为电厂热能动力工程专业本科生教材, 也可供从事火力发电厂运行的工程技术人员参考。

高等学校教材

微机在汽轮机中的应用

清华大学 陈旭

东南大学 周曦 合编

西安理工大学 黄树红

机械工业出版社

(北京三里河路6号)

新华书店北京发行所发行·各地新华书店经营

北京市地矿局印刷厂印刷

*

787×1092毫米 16开本 10.5印张 238千字

1995年5月第一版 1995年5月北京第一次印刷

印数0001—1520册

ISBN7-120-02329-2/TK·326

定价6.10元

前 言

本书是根据1989年高等学校热能动力类专业教材编审委员会会议安排和1991年召开的高等学校热能动力类专业教材编审会议审查通过的《微机在汽轮机中的应用》教材编写大纲编写的。

全书共分三章，主要内容有微型计算机基本结构与工作原理，Z80的指令系统，模拟量输入输出通道结构与接口技术，微机应用于生产过程的抗干扰措施，汽轮机运行参数的微机巡回检测，汽轮机常规性能的计算与监视，微机在汽轮机优化运行中的应用，汽轮机设备的状态监测与故障诊断，大型汽轮发电机组的微机监控系统介绍等。

目前，在火力发电厂特别在大型机组中，已越来越多地利用微机对生产过程参数和设备状态进行实时监视、保护和控制，来替代传统的监控方法。本书针对这一情况，介绍有关微机的基本原理和在汽轮机运行中应用微机的基本方法，使热能动力类专业学生对微机应用具备一定基本知识，以期对从事汽轮机运行方面的技术工作有所裨益。

本书第一章和第三章第二、三节由东南大学周曦同志编写；第二章和第三章第一节由华中理工大学陈旭同志编写；第三章第四、五节由华中理工大学黄树红同志编写。全书由陈旭同志负责统编工作。

由于水平限制，书中难免有缺点和错误，请读者批评指正。

编 者

1994年7月1日

目 录

前 言

第一章 微型计算机的基本原理.....	1
第一节 微型计算机的结构及工作过程	1
第二节 CPU的结构及Z80CPU简介	5
第三节 存贮器	12
第四节 Z80的指令系统.....	15
第五节 输入和输出.....	28
第六节 中断	34
第七节 常用芯片介绍.....	44
第二章 模拟量输入输出通道.....	53
第一节 模拟量输入输出通道的组成和功用.....	53
第二节 多路开关和采样保持器	54
第三节 D/A与A/D转换技术	58
第四节 A/D转换器与CPU的接口	66
第五节 模拟量输入与输出通道的设计	73
第六节 过程通道中的干扰及抗干扰措施	75
第七节 数字滤波	86
第八节 标度变换.....	90
第九节 显示器件和数字显示	93
第三章 微机在汽轮机运行中的应用	100
第一节 汽轮机运行参数的微机巡回检测.....	100
第二节 微机对汽轮机常规性能的计算与监视	110
第三节 微机在汽轮机优化运行中的应用.....	123
第四节 汽轮机设备的状态监测与故障诊断	136
第五节 大型汽轮发电机组的计算机监控系统	152

第一章 微型计算机的基本原理

随着集成电路技术的飞速发展,70年代初诞生了一代新型的电子计算机——微型计算机。它利用大规模集成电路,把计算机的中央处理单元(简称CPU),即计算机的运算器和控制器集成在一个芯片上,该芯片即称为微处理器。微处理器与存储器、输入输出电路及系统总线一起,组成了微型计算机。以微型计算机为主体,辅以外围设备(显示器、打印机、键盘、磁盘、绘图仪等)、系统软件和电源,就组成了功能齐全的微型计算机系统。微型计算机和微型计算机系统一般统称为微型机或微机。

微型机的生产从4位、8位到16位已形成系列。它以利用大规模和超大规模集成电路为技术特征,大大缩小了计算机的体积,同时也降低了成本。目前,微型机由于其体积小、性能价格比高而在科研、生产的各个领域得到广泛的应用。在火力发电厂,随着机组容量和参数的提高,热力系统和辅助设备也趋于复杂,运行中需要监视和调节的参数大大增加,电力生产的特殊性又决定了对这些机组运行的安全性、经济性有更高的要求。如仅采用传统的由人工监视、操作的方法,就很难适应这一特点。因此,微型机由于其较强的信息处理功能及较快的运算速度,参与电厂运行、管理已成为一种必然趋势。

本章将以8位字长的微型机Z80为例,介绍微型计算机的基本知识。

第一节 微型计算机的结构及工作过程

一、微型计算机系统的构成

微型机的核心是中央处理单元(CPU),即微处理器。微处理器通过外部总线,与存储器、输入输出接口电路和外围设备一起,组成一个实用的微型计算机系统。见图1-1。这些设备统称为计算机的硬件。在这些硬件中,常把CPU和存储器合在一起称为微型机的主机,而把输入输出设备统称为微型机的外围设备。

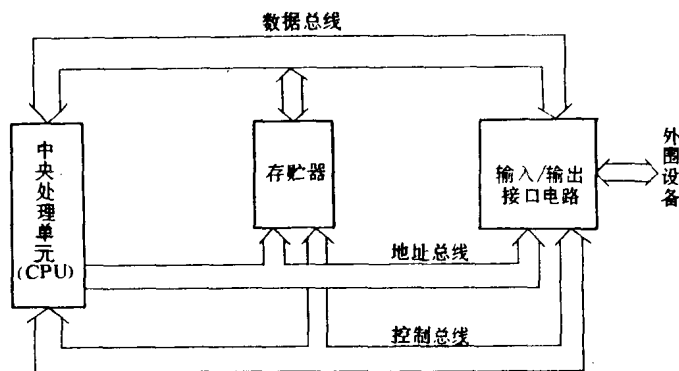


图 1-1 微型计算机结构图

CPU的主要功能是：根据事先给定的程序，为整个计算机系统的协调工作下达命令，这由CPU中的控制器完成；进行具体的数值和逻辑运算，这由CPU中的算术逻辑单元完成；保留运算结果、储存运算所用的数据及为了使机器能自动进行运算而编制的各种命令，这些任务由CPU中的寄存器完成。

CPU中的寄存器能存贮的信息是非常有限的（至多只有几十个字节），因此，要将程序、原始数据、中间结果等全部贮存起来，就必须使用存贮器。存贮器通常又可分为内存和外存两部分。内存容量小（目前微型机常用的为32K或64K字节），但存贮速度快，常用的有磁心或半导体存贮器，目前绝大部分微型机使用半导体存贮器。外存容量大，但存取速度慢，常用的有硬磁盘、软磁盘及磁带等。

要使计算机完成一个具体任务，就必须输入原始的数据与命令，所以必须有输入设备；而中间数据或计算结果需要输出，就要有输出设备。通常的输入输出设备有：键盘、纸带读入机、CRT显示器、打印机、绘图仪等，这些设备统称为微型机的外部设备，它们必须通过专门的接口电路才能接入微型机系统。

二、微型计算机的总线结构

总线是微型计算机系统各部件之间所有标准信号的总交换线。

计算机在工作时，有两种信息在流动：一种是数据，另一种是控制命令。各种原始数据、中间结果、程序等，由输入设备输入至运算器，再存于存贮器中，在运算处理过程中，数据从存贮器读入运算器进行运算，运算的结果要存入存贮器中，或最后由运算器经输出设备输出。而人给机器的各种命令（即程序）也以数据的形式由存贮器送入控制器，由控制器经译码后变为各种控制信号，来控制输入输出装置的启停、运算器的运算和处理、存贮器的读或写等。CPU与这些设备间的信息传递，都是通过外部总线来完成的。

根据传递信息的不同，外部总线分为地址总线、控制总线和双向数据总线。地址总线通常为16位，用 $A_{15}-A_0$ 表示，因此可通过地址总线寻址的内存单元为 $2^{16}=64K$ ，输入输出接口也是通过地址总线的低8位来寻址的，故可寻址 $2^8=256$ 个外设接口。数据总线的宽度等于计算机的位数，对8位机而言，它就是8位的。

微处理器的总线结构，见图1-1所示。

微处理器的内部也采用总线结构。微处理器的结构受到大规模集成电路工艺的约束。首先，为了提高生产的成品率，降低成本，微处理器只能生产在较小的芯片上；其次，由于引出端越多，测试设备就相应复杂，成本也越高，因此为了降低成本，微处理器封装引出端就受到限制，目前八位机都采用40个管脚。有限的管脚数量就约束了控制总线的数量。以上约束就严格地规定了通用微处理器的内部结构及其同外部的连接方式。外部如上所述采用三总线结构。内部为了减少连线所占面积，采用内部单总线结构，即把内部所有单元电路都挂在内部总线上，分时使用总线。这样，虽然使微处理器的运算速度减慢，但简化了内部结构。见图1-2。

三、微型计算机的软件

计算机要完成各种计算任务，除了具备一定的硬件外，还必须有软件的配合，首先是指令和程序。

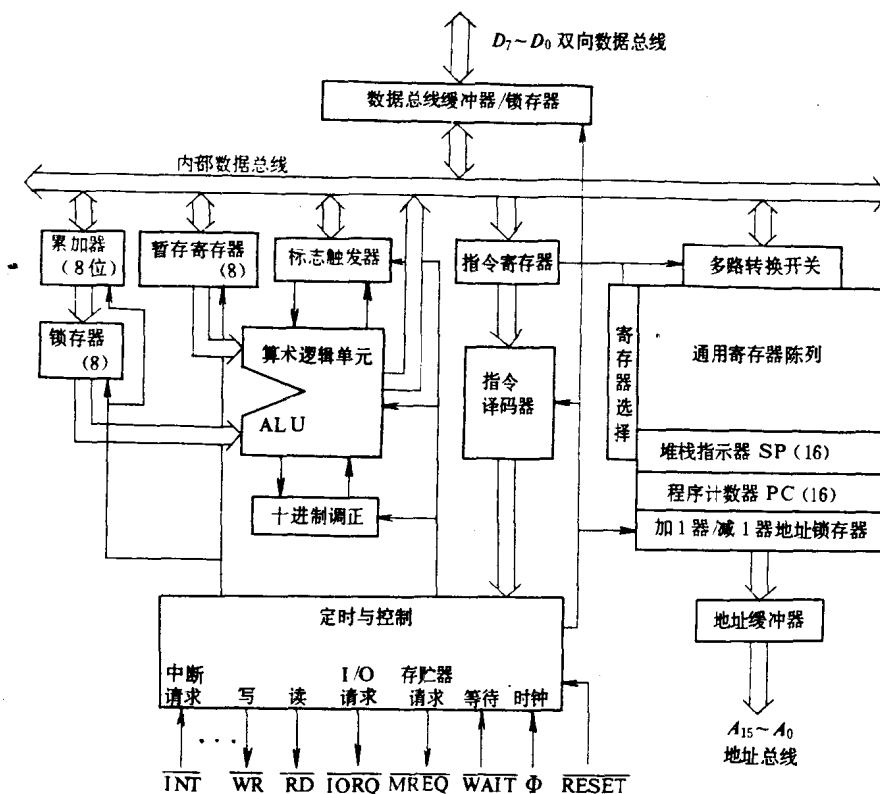


图 1-2 CPU内部结构方框图

1. 指令及指令系统

程序是由一条条指令组成的。针对具体问题，人们把解决这一问题的步骤分解成计算机能执行的一步步具体操作，相应于每步操作，就对计算机给出一条命令，即指令，一系列的指令就构成了一个完整的程序。

一个计算机能执行什么样的操作，能做多少操作，这是由设计计算机时所规定的指令系统决定的。一条指令对应着一种基本操作。计算机所能执行的全部指令，就是计算机的指令系统。这是每种类型的计算机所固有的。

指令通常分成操作码和操作数两大部分，操作码表示计算机执行什么类型的操作、操作数部分指定参加操作的数的本身或操作数所在的地址。因为计算机只能识别二进制数码，所以计算机的所有指令，都只能以二进制编码的形式来表示。如在Z 80中加法指令的编码为87H，向存储器存数指令的编码为77H，这就是通常所说的指令的机器码。本章所要介绍的Z80微型机为8位字长，即一个字节为8位二进制数，当指令用一个字节不能充分表示各种操作码和操作数时，就用两个或两个以上的字节。所以，按照字节数，指令可分为单字节、双字节、3字节及4字节指令。

2. 汇编语言及汇编程序

用户为解决自己的问题用某种语言所编的程序，称为源程序。如果用户用指令的机器码来编制源程序，就称这个源程序是用机器语言编写的。

由于机器码是由一连串的 0 和 1 组成的, 特征不明显, 不易理解, 且很容易出错, 因此用机器语言编程是一种十分困难十分繁琐的工作。因而, 人们就用一些助记符 (通常是指令功能的英文缩写) 来代替操作码。如 Z 80 中, 数的传送指令用助记符 LD 表示, 加法用 ADD 表示等。这样, 每条指令都有明显的特征, 易于理解和记忆, 也不易出错。这种用助记符表示的指令称为汇编语言。用户可用汇编语言编写源程序, 而把汇编语言翻译成机器语言 (机器语言的程序称为目标程序) 的工作, 由计算机中的一个专门程序来完成, 该程序即称为汇编程序。汇编程序把汇编语言翻译成机器语言的过程, 称为汇编。

3. 高级语言及系统软件

与机器语言相比, 汇编语言较易理解、掌握, 但它仍与机器指令是一一对应的, 程序的语句数仍然很多, 编程仍然是一件十分繁琐、复杂的工作, 而且必须对机器的指令系统十分熟悉, 还不能脱离具体的机器。因而用汇编语言编写的源程序不能在不同的机器上通用。

为了使用户可以离开具体的机器编写程序, 也为了使程序中所用的语句与实际问题的更接近, 使程序具有更强的通用性, 就出现了各种高级语言, 如: BASIC, FORTRAN, PASCAL 等。高级语言易于理解和掌握, 用它编程序也大为方便, 工作量大大减少。但是, 计算机在运算时, 首先必须把用高级语言编写的源程序翻译成用机器指令表示的目标程序, 然后计算机才能执行。这样就需要有各种执行这种翻译任务的解释程序 (例如对 BASIC) 或编译程序 (如 FORTRAN 等)。

随着计算机发展得速度更快、容量更大, 以及计算机应用的普及和推广, 计算机的操作方式也由手工过渡到自动运行, 于是就出现了操作系统。它能控制计算机中的所有资源 (CPU、存储器、输入输出设备以及计算机中的各种软件), 使多道程序能自动地成批运行, 且充分发挥各种资源的最大效能。

这些为了使用和管理计算机所编制的软件, 统称为系统软件。它是由机器的设计者提供的。系统软件包括:

- (1) 各种语言的汇编、解释或编译程序;
- (2) 机器的监控管理程序、调试程序、故障检查和诊断程序;
- (3) 程序库, 为了扩大计算机的功能, 便于用户使用, 机器中设置了各种标准子程序, 这些子程序的总和就形成了程序库;
- (4) 操作系统。

4. 应用软件

用户利用计算机以及它所提供的各种系统软件, 编制解决用户实际问题的程序, 这些程序就称为应用软件。应用软件也可逐步标准化、模块化。科学研究或工程技术中解决各种典型问题的应用程序的组合就称为软件包。

四、微型计算机的工作过程

为了使计算机完成某项任务, 用户通过外围设备把程序和原始数据输入, 经输入输出电路和数据总线送入存储器。存储器中存放指令的区域称为程序存储区域, 存放数据的区域称为数据存储区域。这两者往往是分开的。

程序是由一系列指令组成的，因此机器在执行程序时也是按指令逐条执行的。完成一条指令所需的时间称为指令周期。指令周期分成两个阶段：第一个阶段，先从存储器中把指令取出，这一过程称为取指，然后对该指令进行译码，说明该指令完成何种操作及指出操作数的地址，接着，根据这个地址取出操作数；第二阶段，按照操作码指定的操作类型，对操作数进行操作，这是执行指令的阶段。

下面，结合图1-2，简要介绍微机执行一条指令的工作过程。

程序计数器指出当前指令的地址，地址总线根据该地址向存储器指出当前指令所在的地址。根据该地址从存储器中取出的指令，经过双向数据总线、内部数据总线进入指令寄存器。指令的操作码部分经指令译码器译码后，控制电路为完成此指令发出相应的控制信号。指令的地址码部分被直接（或经过修改后）送入地址锁存器，然后，通过地址总线向存储器指出操作数的地址。CPU按操作数地址从数据存贮区中取出操作数，再经过数据总线将操作数送入暂存寄存器。该操作数被送到算术逻辑单元ALU的一个输入端，ALU的另一个输入端到累加器A。在ALU中，对其两个输入端引来的数执行算术或逻辑运算，其操作结果再通过内部数据总线送回累加器，同时，还把与操作结果有关的一些特殊信息，送入标志寄存器，这些内容将对下一条指令如何执行产生影响。至此，执行一条指令的过程便结束了。

在以上讨论中，操作数是从存储器中取出的。在一般微处理器中，设置了由多个通用数据寄存器组成的寄存器阵，这样，有些操作数便可从寄存器中取出。这样比从存储器去取操作数节省时间，从而缩短整个指令的执行时间。

第二节 CPU的结构及Z80CPU简介

一、CPU的一般结构

微处理器的内部由寄存器、算术逻辑单元ALU及控制器等组成。

（一）寄存器

寄存器是微处理器的重要组成部分，它既可以存放各种数据和地址，也可以控制程序执行的次序。CPU中的寄存器可以互相连接，也可以通过内部总线与外部总线相连，再通过外部总线与外面的存储器及输入输出设备相联系。

寄存器和内部总线的宽度（即位数）是一致的，它也和计算机其它部分的位数相同。我们称4位、8位和16位的微处理器，一般就是对这个位数而言的。但有时也可以两两连成寄存器对使用，这样它的位数就是内部总线数的两倍。

微处理器中的寄存器通常有以下几种类型：①累加器A；②指令寄存器IR；③存储地址寄存器，包括程序计数器、堆栈指示器、变址寄存器等；④通用寄存器。

下面分别介绍累加器、指令寄存器及存储地址寄存器，其余在Z80CPU的内部结构中介绍。

1. 累加器A

累加器A是算术逻辑单元ALU的关键部件。ALU执行算术运算时，它的一个输入数

一般是从累加器来的,其结果又送回累加器。微处理器也用累加器执行逻辑操作、移位以及其它指令动作。微处理器有一个或多个累加器。增加累加器的数目,可以节省累加器传送数据的时间,提高CPU的运算速度,但相应的也会增加一些成本,在指令中还必须指明其中某个的数码。

2. 指令寄存器IR

IR用来保存要执行的指令的内容,一直到指令被译出为止。它是指令译码器、定时和各种控制信号的产生电路。它把用户程序中的指令逐条译出,然后按一定的时序发出相应的控制信号,因此,它相当于微处理器的控制器。

IR的宽度一般与微处理器的基本指令字长相同,对于8位机,IR的字长即为8位。对于双字节和3字节指令,分别要经两次、三次译码。有的微处理器有多个指令寄存器,其中一个存放当前执行的指令,其余的存放以后要顺序执行的几条指令。这就可使CPU在执行当前指令的同时,去摄取下一步要执行的指令,即取指和执行的时间可重叠起来,从而可大大提高处理效率。这种处理过程叫“流水线操作”。

3. 存贮地址寄存器MAR

它用以存放存贮器的地址值。一般微处理器的MAR为两个字长,对于8位机,即为16位,它可以指出 $2^{16} = 64K$ 存贮空间。存贮器地址既可在指令中直接给出,也可以通过其它途径间接给出。MAR的输出与地址总线连接,它的输入由内部总线供给。对于8位的寄存器,要分两次传送数据,一次送低8位地址,另一次送高8位地址。实际上,程序计数器PC、堆栈指示器SP、变址寄存器IX等都属于地址寄存器。

(二) 算术逻辑单元ALU

它是执行算术和逻辑运算的装置。它以累加器A的一个内容为操作数,另一个操作数由内部数据总线供给,它可以是寄存器中的内容,也可以是由数据总线供给的由内存读出的内容,它们通过暂时寄存器与ALU相连。ALU有两个输出端:一个用来输出操作结果(通常通过内部总线送至累加器A中),另一个用来输出表示操作结果特殊信息的标志位,送入至标志触发器中。参见图1-2。

ALU执行的典型操作有:①加法;②减法;③逻辑“与”;④逻辑“或”;⑤逻辑“异或”;⑥求补;⑦增量(加1);⑧减量(减1);⑨清零;⑩移位及循环移位。

与ALU相连的标志触发器,其作用是存贮ALU输出的操作结果的特殊状态,这种状态作为一种条件,常用于判断是否控制程序转移等。一般微计算机的标志有以下几种:

(1) 进位(C)标志。当两个8位数在ALU中相加时,若其结果产生进位,则将这个进位存贮在这个进位标志位中。该标志位可由计算机来检验。

(2) 零(Z)标志位。当ALU操作结果为零时,则在零标志位Z中置“1”。这对于算术运算或逻辑运算都可适用。Z标志位还可用于检查输入输出数据在传送中是否出错。

(3) 符号(S)标志位。符号标志位S有时称为负数标志位N。在2的补码表示法中,累加器的最高位表示数的符号。若这位是“1”,则表示是负数,同时使符号标志位成“1”。检验标志位S,就可知道操作结果的正负。

(4) 辅助进位 (AC) 标志位。AC 标志位用在二进制 (BCD) 操作中。一个 8 位字节 (第 0 位至第 7 位) 可表示两个 BCD 数字。当两个数相加时, 其结果的低位 BCD 的数字可能有进位 (即在 b_3 与 b_4 之间), 这个中间的进位使 AC 置 “1”。AC 标志位主要用在十进制的运算中。

(5) 奇偶校验 (P) 标志位。P 标志位用来检查数据传送是否有错。奇偶校验分奇数校验和偶数校验两种。偶数校验的原理是: 把所传送的数的最高位留出来, 对 8 位数来说, 就是把第 8 位留出来, 其余 7 位作为传送的数据, 并计数 7 位数中 “1” 的个数。如果 “1” 的个数为奇数, 则在第 8 位中置 “1”, 否则置 “0”, 以满足 8 位数中 “1” 的个数为偶数。对于这种偶数校验方法, 在数据传送过程中, 如能保持 “1” 的个数为偶数, 则 P 标志位置 “1”; 如发现 “1” 的个数为奇数, 则表示传送有错, P 置 “0”。奇数校验方式的原理与上述相同, 只是在第 8 位置 “1” 或 “0” 时, 应满足 8 个数位中 “1” 的个数为奇数。

二、Z80CPU的内部结构

Z80CPU 是一种 8 位的 40 条引脚的内部单总线结构的微处理器, 它的结构如图 1-3 所示。

(一) CPU 内部寄存器组

Z80 的内部寄存器组分成两种类型: 一种是专用寄存器, 另一种是通用寄存器。现分别介绍如下。

1. 程序计数器 PC

这是个 16 位计数器, 它总是包含着下一条要执行的指令的 16 位地址。CPU 总是把 PC 中的内容作为地址, 从内存中取出指令, 加以译码和执行。通常指令是顺序执行的, 所以在一般情况下, 当取出一条指令 (更确切地说为一个指令字节) 后, PC 就自动加 1, 只有在执行转移指令、子程序调用指令或中断时, 才把要转向的地址赋给 PC。

2. 堆栈指针 SP

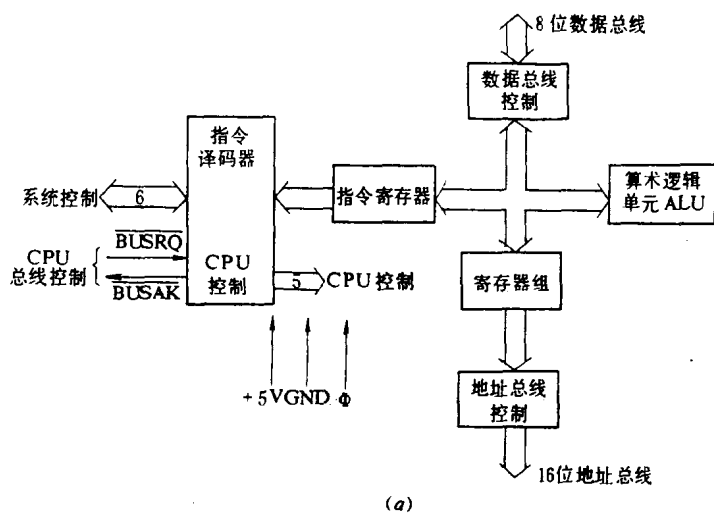
堆栈是在 CPU 外部存储器中的一个按照后进先出的原则组织的存储区域。堆栈指针包含着 16 位地址, 它始终指向堆栈的顶部。利用 PUSH 和 POP 指令, 可把内部寄存器对的内容推入堆栈 (推入到 SP 所指的单元), 或把堆栈的内容 (SP 所指的单元中内容) 弹出到内部寄存器中去。利用堆栈, 可以实现子程序嵌套、多级中断等。关于堆栈将在本章第四节中较详细地介绍。

3. 变址寄存器 IX 和 IY

这是两个独立的 16 位的寄存器。通常它们各自包含着一个 16 位的基地址, 由它加上指令中给定的偏移量, 形成操作数的有效地址。这种称为变址寻址的寻址方式使许多程序大大简化, 特别是在用到数据表格的场合。

4. 中断页地址寄存器 I

这是一个 8 位的寄存器。当 CPU 用中断方式与外设交换信息时, 若外设有中断请求, 而 CPU 也允许和响应了中断, 则程序就要转向中断服务程序。Z80 共有三种中断方式, 它的中断方式 2, 允许有 128 个中断服务程序的入口, 这些入口地址形成了一个表格, 这个表



主寄存器组		辅助寄存器组	
累加器 A	标志寄存器 F	累加器 A'	标志寄存器 F'
B	C	B'	C'
D	E	D'	E'
H	L	H'	L'

} 通用寄存器

中断矢量寄存器 I	存储器刷新寄存器 R
变址寄存器 IX	
变址寄存器 IY	
堆栈指针 SP	
程序计数器 PC	

} 专用寄存器

(b)

图 1-3 Z80CPU的内部结构

(a) CPU内部简化框图； (b) Z80CPU内部寄存器组

格存放在内存的哪一页（若以256个字节为一页，则64K内存就分为256页），就由I寄存器中的内容确定（由用户预先给定）。所以，I寄存器中的内容，是中断服务程序入口地址表的页地址（即高8位）。

5. 存储器刷新寄存器R

为了增加半导体存储器的集成度，在大容量的存储器中广泛采用了MOS动态存储器，它利用寄生电容来存储信息。为了防止因电容漏电而使信息丢失，就要定期（一般为2ms）对动态存储器进行刷新。Z80是利用取指周期的后两个T状态（此时CPU对指令进行译码和内部操作，不使用存储器）来刷新的，每一个取指周期对一部分存储器进行刷新，以保证在2ms内对整个64K内存都刷新一遍。每次刷新的内存单元（每次刷新内存中的一行而非一个单元）的地址由R寄存器（7位）提供。而在每刷新一行后，R的内容自动加1，以指向下一行。实际上R为一个计数器。

6. 累加器A和状态标志寄存器F

Z 80中有两个累加器A和与它相连的状态标志寄存器F。在进行算术和逻辑操作时,累加器A中的内容必为一个操作数,且操作的结果放在A中。另外,算术和逻辑操作结果的一些特征,如操作结果是否为零、是否有进位等都寄存在标志寄存器中。A和F都是8位的寄存器。

程序员可用简单的交换指令来选择两个累加器中的任一个工作。在复位状态后是累加器A工作。

7. 中断控制触发器IFF₁和IFF₂

Z 80CPU中有两个用以控制CPU开中断或关中断的触发器IFF₁和IFF₂。它们的作用在本章第六节中介绍。

8. 通用寄存器组

Z 80中有两组一样的通用寄存器,每组都有6个8位的寄存器,分别是B、C、D、E、H、L和B'、C'、D'、E'、H'、L'。它们可以分别作为6个8位的寄存器使用,也可以两个连起来形成BC、DE、HL 3对16位的寄存器对。它们主要用于寄存参与运算的8位数据,或操作数的16位地址。

在工作时,只有一组寄存器参与操作,但可以用一个简单的交换指令,来选择另一组寄存器。

(二) 算术和逻辑单元ALU

CPU的8位算术和逻辑指令在ALU中执行。ALU通过内部数据总线与内部寄存器和外部数据总线交换信息。ALU所能完成的功能有加、减、逻辑“与”、逻辑“或”、逻辑“异或”、比较、左或右的移位或循环、加1、减1、位操作等。

(三) 指令寄存器IR和CPU控制

这部分相当于控制器的功能。从存储器取出来的指令,送到指令寄存器IR中,然后由指令译码器译码,通过定时和控制电路,在规定的时刻发出完成各种操作所需要的全部内部控制信息,以及发出所需要的对CPU外部的控制信号。

CPU的动作需要精确定时。基本定时脉冲由外部振荡器产生,接入CPU的 Φ 端。两个定时脉冲之间的间隔称为一个时钟周期,也称T周期。CPU实现某种规定的基本操作所需的时间,称为机器周期或M周期,一个机器周期一般由3~4个时钟周期组成。一条指令从取出到执行完毕所需的时间,称为指令周期。CPU执行不同指令所需的机器周期及时钟周期(它反映了CPU完成指令所需的时间)是不同的,一个指令周期一般由1~6个机器周期组成。不管什么指令,它的第一个机器周期M₁都是取指令周期。Z 80的定时波形如图1-4所示。

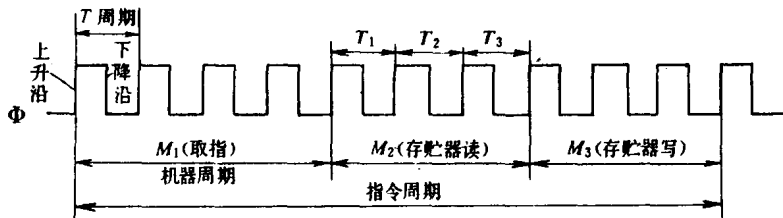


图 1-4 Z 80定时波形图

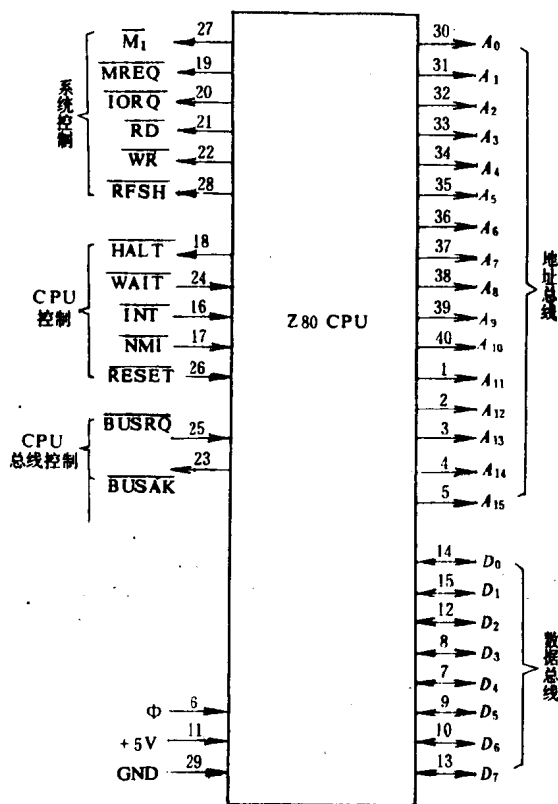


图 1-5 Z80CPU引脚图

三、Z80的引脚及其功能

Z80有40条引脚，如图1-5所示。

其中，有16条为地址总线 $A_{15} \sim A_0$ ；有8条为双向数据总线 $D_7 \sim D_0$ ；三条作为电源、接地和时钟信号线；另13条为控制信号线。这些引线的功能如下：

$A_0 \sim A_{15}$ 为地址总线，它是三态输出，高电平有效。 $A_0 \sim A_{15}$ 组成16位地址信息，用于寻址内存（最大到 $2^{16} = 64\text{K}$ 字节）以及输入输出装置。输入输出寻址用地址总线中的低8位，用户可直接选择 $2^8 = 256$ 个输入输出端口。 A_0 是最低有效位。在动态存储器刷新期间，低7位由R寄存器提供一个有效的刷新地址。

$D_0 \sim D_7$ 为三态输入/输出，高电平有效。 $D_0 \sim D_7$ 组成了一个8位的双向数据总线。系统内数据的传送——CPU与存储器之间、CPU与外设端口之间以及存储器与外设之间的数据都是通过这个数据总线传送的。

$\overline{M}_1$ 输出，低电平有效。 $\overline{M}_1$ 指示现行的机器周期是取操作码周期。当指令具有两个字节机器码时，则每次取操作码周期，都发出 $\overline{M}_1$ 信号。这样的两个字节的操作码（此处指指令的机器码）的第一个字节通常为CBH、DDH、EDH或FDH。 $\overline{M}_1$ 也与 $\overline{\text{IORQ}}$ 信号一起作为一中断响应周期。

MREQ，三态输出，低电平有效。这是存储器请求信号，它表示地址总线上保持着一个供存储器读或写操作的有效地址。

$\overline{\text{IORQ}}$ ，三态输出，低电平有效。这是输入/输出请求信号，它表示地址总线的低8位上保持着一个供I/O（输入/输出）读或写的有效的I/O端口地址。当中断被响应时，也产生一个 $\overline{\text{IORQ}}$ 信号，它与 $\overline{M}_1$ 一起表示中断响应，通知外设把中断矢量放到数据总线上。在 $\overline{M}_1$ 期间可以发生中断响应，而在 $\overline{M}_1$ 期间，绝不发生I/O操作。

RD，三态输出，低电平有效。这是读信号，它表示CPU要求从存储器或某个I/O装置读入数据。所寻址的I/O装置或存储器，应用这个信号把它们的数据门打开（即作为三态门的选通信号），使数据进入数据总线。

WR，三态输出，低电平有效。这是写信号，它指示CPU的数据总线上保持着有效的数据要存入（写入）所寻址的存储器或I/O装置。

RFSH，输出，低电平有效。这是动态存储器刷新信号，它表示地址总线的低7位保持有动态存储器的刷新地址，此时 A_7 是逻辑0，地址总线的高8位包含I寄存器的内容。这

个信号与 $\overline{\text{MREQ}}$ 信号一起用于刷新动态存储器。

$\overline{\text{HALT}}$, 输出, 低电平有效。这是暂停信号, 它表示CPU已经执行了一条HALT软件指令, CPU就进入了暂停状态。一直要到CPU接受到非屏蔽中断或者屏蔽中断, 操作才能重新开始。在暂停期间, CPU执行NOP(无操作)指令, 此时仅维持进行存储器刷新。

$\overline{\text{WAIT}}$, 输入, 低电平有效。这是要求等待信号, 它告诉CPU: 所寻址的存储器或I/O装置, 尚未准备好数据传送。只要这个信号有效, CPU就继续插入等待周期, 直至外设或存储器将数据准备好、该信号无效为止。这个信号能使高速CPU与慢速的存储器或I/O装置协调工作。必须注意的是, Z80CPU在此状态下不发生动态存储器的刷新。

$\overline{\text{INT}}$, 输入, 低电平有效。这是屏蔽中断请求信号, 它由I/O装置产生。它告诉CPU, 外设产生中断请求。此时, 如果在CPU内部的由软件控制的中断允许触发器 IFF_1 为“1”, 即为开中断状态, 且在没有总线请求的情况下, 即CPU在现行指令结束后响应中断。当CPU接受中断后, 在下一个指令周期的开始, 就送出一个中断响应信号(在 $\overline{\text{M}_1}$ 周期有 IORQ 信号)。CPU响应中断的方式, 在本章第七节介绍。

$\overline{\text{NMI}}$, 输入, 负边沿触发。这是非屏蔽中断请求信号, 它比 $\overline{\text{INT}}$ 具有更高的优先权。当外设向CPU产生 $\overline{\text{NMI}}$ 信号后, 不管中断允许触发器 IFF_1 的状态如何, 只要没有总线请求信号, CPU就在现行指令结束后响应该请求。 $\overline{\text{NMI}}$ 自动地迫使Z80CPU转向内存中0066H单元, PC的内容自动保存在堆栈中, 保证在中断服务程序结束后自动返回至中断前的程序。而非屏蔽中断的服务程序, 就放在从0066H开始的内存单元中。

$\overline{\text{RESET}}$, 输入, 低电平有效。这是复位信号, 它迫使PC中的内容为0, 且对CPU进行初始化。CPU的初始化包括:

- (1) 使中断允许触发器 IFF_1 置为零;
- (2) 置I寄存器为00H;
- (3) 置R寄存器为00H;
- (4) 置中断方式0。

在复位期间, 地址总线和数据总线处于高阻状态, 且所有控制信号处于无效状态。不发生动态存储器刷新。

$\overline{\text{BUSRQ}}$, 输入, 低电平有效。这是总线请求信号, 它表示外设请求使用总线, 要求CPU把地址总线、数据总线和三态的输出控制信号处于高阻状态, 使别的装置能控制这些总线。这主要用于存储器与快速外设之间直接传送信息(DMA)。当 $\overline{\text{BUSRQ}}$ 有效时, CPU在现行的机器周期结束时立即响应, 使这些总线处于高阻状态。

$\overline{\text{BUSAk}}$, 输出, 低电平有效。这是总线响应信号, 它告诉总线请求装置, CPU的地址总线、数据总线和三态的控制总线已处于高阻状态, 外部装置现在可以用这些总线了。必须注意的是, 在 $\overline{\text{BUSAk}}$ 状态下, 不发生动态存储器的刷新。

Φ , 单相系统时钟。Z80时钟频率为2.5MHz, Z80A的时钟频率为4.0MHz。

第三节 存 贮 器

存储器是计算机极其重要的组成部分，它用来存放程序和数据。有了它，计算机才能具有记忆功能。显然，存储器的容量越大，则记忆的信息越多，计算机的功能就越强。

存储器由存储单元、地址译码器、控制器组成，它通过地址总线、数据总线及存储器

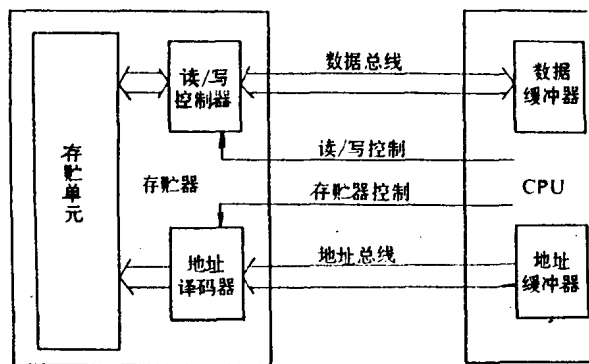


图 1-6 存储器结构及其与CPU连接示意图

读写控制线与CPU相连接，如图1-6所示。

在CPU中，地址缓冲寄存器的内容由地址总线传送到存储器中的译码器中，经译码后选中存储单元中的一个地址。CPU数据缓冲寄存器中的内容通过数据总线和存储器读写控制在CPU的读写信号控制下写入存储单元，或从存储单元中读出。

一、存储器的分类

微型计算机的存储器主要使用半导体存储器。从使用功能上分，目前所使用的半导体存储器可分为两类：随机存取存储器（RAM）和只读存储器（ROM）。

1. 随机存取存储器RAM

RAM中存储单元的内容，按需要可由CPU写入，也可以由CPU读出。因此，它主要用来存放各种用户程序、现场的输入或输出数据、中间计算结果、与外存交换信息及作堆栈使用。RAM又可分成静态RAM和动态RAM两种。

静态RAM是由触发器贮存信息的，只要有电源，信息就可保存下来。

动态RAM是利用MOS器件栅极电容上的电荷来保存信息的。由于电容存在漏电，所以保存的信息会因漏电而消失。因此，动态RAM中的内容每隔2ms就必须刷新一次，即把存储器单元中的内容读出来，再写回到原单元中去。故采用动态RAM的微型计算机必须有相应的动态存储器刷新电路。动态RAM由于组成存储单元的晶体管少，其存储密度比静态RAM高，再加上价格便宜，因此在微型计算机中被广泛采用。

2. 只读存储器ROM

在用户使用计算机时，ROM中的所有信息只能被读出，而不能被CPU写入。ROM没有挥发性，即在断电的情况下信息也不会丢失，故一般用来存放固定的程序，如微型机的管理和监控程序、汇编程序以及各种表格等。只读存储器又可分为ROM、可编程的只读存储器PROM和可擦除的可编程只读存储器EPROM三种。

ROM由制造厂家在生产时采用掩模工艺把信息固化在存储器中，一旦形成产品，信息就不能再改写了。因此信息稳定可靠，一般用于存放系统管理程序。

PROM又称熔丝式只读存储器。其存储的内容是由用户通过PROM写入器写入的，一旦写入就不能再改写了，它只能由用户写入一次。