第 5 章　基于 PC 运动控制板卡的交流伺服运动控制系统

从 20 世纪 70 年代后期开始，随着微处理器技术、大功率高性能半导体功率器件技术和电机永磁材料制造工艺的发展及其性能价格比的日益提高，交流伺服技术——交流伺服电机和交流伺服控制系统逐渐成为主导产品。交流伺服驱动技术已经成为工业领域实现自动化的基础技术之一。伺服系统以其出色的性能满足了各种产品制造厂家的要求，从而能够对产品的加工过程、加工工艺和综合性能进行改造。

基于 PC 的伺服运动控制系统是一种开放式结构，它可以充分利用 PC 的资源和第三方软件资源完成用户应用程序的开发，将生成的应用程序指令通过 PC 并行总线传送给运动控制器，运动控制器则根据来自上位计算机的应用程序命令，按照设定的运动模式，向驱动器发出运动指令，完成相应的实时运动规划，达到工业生产的目的。

5.1　预备知识

5.1.1　伺服运动控制系统的组成

在机电一体化设备上伺服系统的使用更加广泛，几乎工业生产的所有领域都成为伺服系统的应用对象。表 5-1 列出了伺服系统的主要应用领域。

表 5-1　伺服系统的主要应用领域

加工机械	FA 机械	医疗机械	机器人	自动组装线	半导体制造	部件组装	纺织机械
加工中心 铣床 车床 磨床 数控机床	食品加工 食品包装 自动仓库 搬运机械 印刷机械 挤压成型机	CT 设备 人造器官	弧焊机器人 点焊机器人 搬运机器人 喷涂机器人	卷线机 自动生产线	晶片机械 清洗设备 CVI（化学气相沉积）设备	芯片安装 插装机 焊接机	编织机 纺织设备

由于伺服驱动产品在工业生产中的应用十分广泛，市场上的相关产品种类有普通电机、变频电机、伺服电机、变频器、伺服控制器、运动控制器、单轴控制器、多轴控制器、可编程控制器、上位控制单元、车间级和厂级监控工作站等。

1. 伺服运动控制系统的硬件组成

基于 PC 的伺服运动控制系统一般由上位计算机、运动控制器、驱动器、反馈元件和伺服电机等组成，如图 5-1 所示。

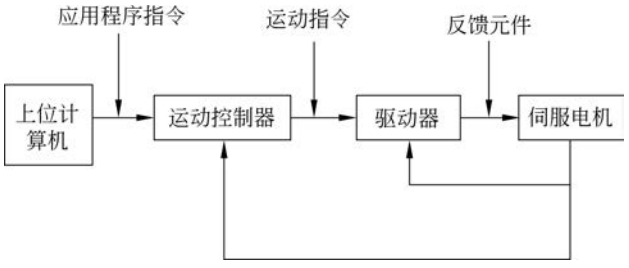


图 5-1 计算机控制系统组成框图

上位计算机：基于 PC 的伺服运动控制系统说到底是一个多 CPU 系统，其中，由 CPU 们各自完成系统分配给自己的任务，并在系统信息上传下递的交互中，实现系统资源的共享。这种系统结构上的“集中管理，分散控制”的思想，可以极大地增强系统的稳定性和可靠性。在结构上，伺服运动控制器需要内置于 PC 之中，即要使用 PC 的环境、电源等资源；在应用上，PC 上的调度程序是建立在操作系统之上的，操作系统的性能也会直接影响伺服运动控制器的工作。因此，上位计算机的选择直接会影响到伺服运动控制系统的性能。

运动控制器：随着生产过程对运动控制系统的速度和精度要求愈来愈高，传统的运动控制系统已难以取得满意的控制效果。由于 DSP(数字信号处理器)具有运算速度快，支持复杂的运动算法，可以满足高精度运动控制要求的特点，以 DSP 为核心的多轴运动控制卡已广泛地应用在运动控制系统之中。

驱动器：交流伺服系统驱动器目前已数字化，内部常采用 DSP，并运用 IGBT PWM 控制方式，可以支持脉冲与模拟量两种输入方式。

伺服电机：常用的有单相与三相供电电压两种，现有产品额定功率可以为 30～4000W。

交流伺服运动控制系统模型形式很多，自动控制工作者必须根据被控对象的需求和所要求的控制指标合理地选用某种控制模型，可详见本书第 3 章。

2. 伺服运动控制系统的软件组成

上位计算机软件：包括系统软件和应用软件。系统软件一般指操作系统和第三方软件资源等，它带有一定的通用性，由计算机制造厂提供。应用软件主要指用户调度程序，它具有专用性，是根据需要由用户开发的对机械传动装置的位置、速度进行实时的控制管理，使运动部件按照预期的轨迹和规定的运动参数完成相应动作的软件。

运动控制器软件：置于伺服运动控制器之中。它是实现伺服运动控制系统的核心，按照一定的控制算法和数学模型而编制的专用程序。

显然，从应用的角度看，自动控制工作者应把主要精力放在上位计算机轨迹运动规划和运动控制器软件的编程上，而系统软件是否丰富、是否能适应用户的要求，可以作为选择计算机的根据之一。

5.1.2 操作系统

操作系统(Operation System，OS)是一组计算机程序的集合，用来有效地控制和管理计算机的硬件和软件资源，即合理地对资源进行调度，并为用户提供方便的应用接口。

对于在 PC 上开发应用软件的用户来说，其工作完全是建立在操作系统之上的，是直接

面对操作系统核心的编程，因此，对于操作系统软件结构的理解至关重要。

为了更好地了解操作系统的功能，首先说明以下几个基本概念：

任务——一个程序分段，这个分段被操作系统当作一个基本工作单元来调度。任务是在系统运行前已设计好的。

进程——任务在作业环境中的一次运行过程，它是动态过程。有些操作系统把任务和进程同等看待，认为任务是一个动态过程，即执行任务体的动态过程。

线程——20 世纪 80 年代中期，人们提出了比进程更小的能独立运行和调度的基本单位即线程，并以此来提高程序并发执行的程度。近些年，线程的概念已广泛应用，在 Windows 中就使用了线程的概念。

多用户和多任务——多个用户通过各自的终端使用同一台主机、共享同一个操作系统及各种系统资源；每个用户的应用程序可被设计成不同的任务，这些任务可以并发执行。多用户及多任务系统可以提高系统的吞吐量，更有效地利用系统资源。

5.1.3　实时多任务操作系统(iRMX)

1. iRMX 操作系统结构

iRMX 是美国 Intel 公司发行的集中式实时多任务操作系统。它的内部同时有多道程序运行。每道程序各有不同的优先级。操作系统按事件触发使程序运行。多个事件发生时，系统按优先级高低确定哪道程序在此时此刻占有 CPU，以保证优先级高的事件实时信息及时被采集。

iRMX 操作系统是模块化分层结构的系统。它由多个子系统组成。iRMX 子系统包括内核程序 iRMK(intel Real-time Kernel)、核心程序(nucleus)、基本 I/O 子系统简称 BIOS(basic I/O subsystem)、扩展 I/O 子系统简称 EIOS(Extended I/O Subsystem)、人机接口简称 HI(Human Interface)。

从图 5-2 可以看出，分层式操作系统，都是以它前一层子系统为基础的扩展。可以直接访问前一层数据代码，也允许访问最内层功能。iRMX 系统的人机接口基于扩展 I/O 子系统，扩展 I/O 基于基本 I/O 子系统，而基本 I/O 完全依赖于核。除了基本 I/O 子系统外，其他层子系统既依赖于前一级又可与核发生直接联系；用户程序可与任何一级子系统相关联，尤其是可以直接利用核提供的强大功能。从内核向外扩展，操作系统每扩展一层，与用户就更接近一步，用户对系统的使用方法就简化一步。

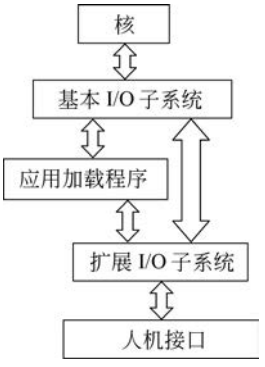


图 5-2　系统结构示意图

2. iRMX 操作系统的核

iRMX 操作系统的核是系统的心脏。系统的实时性，任务调度等重要功能与性能都是基于核实现的。核的主要功能如下：

目标管理——控制系统资源访问，实现任务间的通信。

任务调度——按基于优先级抢占方式调度任务。

中断管理——按基于中断优先级原理，响应外部中断请求，实现中断处理。

iRMX 操作系统中设定了 13 种目标。其中的每一类目标，实际上都是系统内部定义的

数据结构,每一类结构都有它自己的特征。将数据结构提供给各子系统,用来控制资源访问、任务调度、中断响应。核心程序通过对目标管理,从而实现了对各子系统的管理。基于目标的程序设计实际上是利用系统提供的数据结构,实现目标之间的相互联系与相互作用。每种目标都对应着一系列建立目标和管理目标的相关系统调用。系统调用是 iRMX 操作系统提供给用户的编程接口,用于实现用户与系统内部的联系。顺序执行的单任务系统,实际上是系统目标少,目标之间缺少联系的系统。

iRMX 的核心程序中用到两类处理程序:异常处理程序和中断处理程序。

中断处理程序在实时系统中十分重要。对应于每个中断源,应有一个中断处理程序。当该中断源接收到一个外部事件发出的信号时.就会向系统申请 CPU,转去执行中断处理程序。如果中断处理复杂,占用的 CPU 时间较长,或者需要用多种系统调用,这是不允许的,这时 iRMX 要求用户安排一个任务,专由中断处理程序来激活,该任务称为中断任务。中断任务的优先级为 0～127,比所有非中断任务优先级高。执行中断处理程序和中断任务统称为中断处理,它会屏蔽优先级较低的中断。因此,在应用中设计中断服务程序时,要尽量优化程序。

任务是 iRMX 操作系统中唯一可活动的目标。活动是指目标的某些特征是动态变化的。iRMX 中任务的主要特征是任务优先级与任务状态。

iRMX 系统中,任务优先级用 0～255 中的一个整型数表示,数值越小,任务优先级越高,0 为最高优先级。多个任务竞争处理器时,优先级高的任务优先获得 CPU。

iRMX 系统将 0～127 号任务安排与中断级相对应,如果某外部中断长期被屏蔽,对应的任务号可由其他非中断任务使用。

任务优先级是在任务建立时指定的,用户应根据应用系统内不同的工作要求来安排任务优先级。系统运行中,任务优先级不能动态修改。

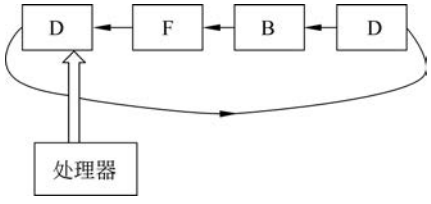


图 5-3 同优先级任务

可以给两个以上任务安排相同的优先级。相同优先级的任务同时申请 CPU 时,系统会安排给每个任务相同的时间片轮流执行。时间片的长短,要在系统配置(CPU)时设定。在一时间片内执行不完的指令将在下一轮时间片内才能继续执行。按这种方式,相同优先级的任务形成了一循环队列,如图 5-3 所示。

iRMX 操作系统对任务优先级范围做了以下安排:

(1) 0～127:操作系统安排用于外部中断服务,即外部中断发生后由中断处理程序激活,处理与外部事件相关的任务。该任务运行时会屏蔽中断优先级较低的外部中断。其中,0～16 由操作系统安排用于系统硬件的中断处理,应用中不能将这些中断级屏蔽。

(2) 128～255:用于用户建立的应用任务,如处理日常工作任务、通信任务等。通常将同优先级的任务安排在 200 左右,按时间片轮询。

任务一旦建立,就处于以下 5 种状态之一:

(1) 运行态(running),即任务正在执行。

(2) 就绪态(ready),即任务运行条件准备就绪,在等待 CPU 执行。就绪态队列中优先级最高的任务将首先获得 CPU。

(3) 睡眠态(sleep),任务暂停执行,“睡眠”一定时间后再继续执行。

(4) 挂起态(suspend),任务被挂起,暂不允许执行,直至有其他任务或某种信息到达为其解挂。

(5) 睡眠挂起态(asleep-suspended),任务睡眠后被挂起,要到两个条件都能满足即睡眠时间到,被其他任务或消息解挂后才能运行。

任务状态之间的转换关系可用图 5-4 表示。由图可见,运行中的任务可以转换成就绪态、睡眠态或挂起态。状态转换的原因可能不相同,任务执行完成后进入就绪态,任务运行中被更高优先级任务中断也可进入就绪态。任务运行中由系统调用实行睡眠态或挂起态的转换;而睡眠态、挂起态的任务被时间唤醒或被解挂都不能直接进入运行态。必须先进入就绪态,在就绪态任务队列中按优先级排队等候 CPU。只有就绪态队列中优先级最高的任务才能进入运行态。一个正在睡眠的任务可能由其他任务挂起,成为睡眠挂起态,如果给定的睡眠时间到了,它还未被解挂,则进入挂起态;如果它被解挂了,但睡眠时间未到,则仍处于睡眠态继续“睡眠”。

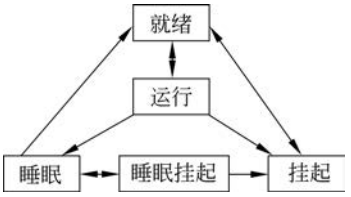


图 5-4　任务状态转换

任务的建立,只是有了独立的运行模块,用通信机制为纽带,把任务联系起来,才能形成一套完整的体系。就像城市中的马路、街道及电话线路把一幢幢建筑物联系起来一样。没有任务间的通信联络及相互关系,多任务就没有意义。

iRMX 操作系统为任务提供了灵活的通信方法,主要概括为两种:同步的方法和互斥的方法。任务通信时,一个任务发送信息,必然有一个任务接收信息,分别称为发送任务或发送方及接收任务或接收方。接收任务只有获得所需的数据后才能继续执行,这种方式称为同步通信。互斥是指任何时候只能有一个任务访问某数据。

这里,只介绍邮箱通信。

使用邮箱通信时,发送任务发送一个信息到邮箱,另一任务从邮箱中接收这一信息。该信息可以是 128 字节的数据或者是一个目标的 TOKEN。通过邮箱发送的目标最常用的是数据段的 TOKEN,这个段中存放着接收方所需要的数据。邮箱常用于同步任务通信,即接收任务只有获得所需的数据后才能继续执行,否则总是在等待。

一个邮箱可由多个任务使用,每个邮箱有两个队列:

(1) 等待接收信息的任务队列;

(2) 已经被送出的目标(或信息)队列,即该队列中的目标暂未被接收任务取走。实际上,任何时候这两个队列中总有一个是空的:或者任务在等待目标信息到来,这时目标信息队列为空;或者是目标信息已到达,在等任务接收,这时任务队列为空。

因为目标信息队列是先进先出(FIFO)队列,目标信息一经发出,在等待的任务立即将其取走,即先进入队列的目标先被取走,使信息队列为空。若此时没有接收任务在等待,则任务队列为空。

任务在等待接收信息期间,处于睡眠状态或睡眠挂起状态。一旦接收到所需的信息,则从睡眠状态转换成就绪状态或从睡眠挂起状态转换为挂起状态。

如果接收任务优先级高于发送任务优先级,则当它接收到信息后会抢占 CPU 而立即投入运行,发送任务被中断。

前面已提到信息队列是先进先出的,但任务队列可以有两种:

① 先进先出队列;

② 基于优先级的队列。

先进先出队列即等待接收信息的任务按发出系统调用"RECEIVE $ …"的时间先后排队;基于优先级的队列即按接收任务的优先级排队。优先级最高的任务总是在队列的最前面。邮箱建立时必须规定自己的任务队列类型,此后不能动态改变。

3. iRMX 操作系统的中断

中断及中断处理,是实时操作系统的中心议题。通过中断,才能将外部随机发生的事件通知 CPU,从而触发一个中断处理过程。iRMX 进行中断处理的过程如图 5-5 所示。

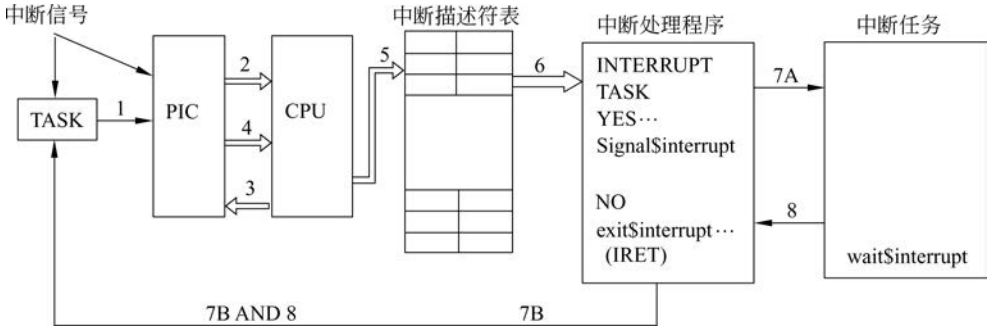


图 5-5 中断处理的过程

图 5-5 中各数字标号代表的含义如下:

1——PIC 可编程中断控制器(Programmable Interrupt Controller)接收到了一个中断;

2——PIC 通知 CPU;

3——CPU 向 PIC 回送一个应答信号;

4——PIC 向 CPU 发送中断号;

5——CPU 从中断描述符表(IDT)中获得相应中断级的中断处理程序;

6——将控制权发送给中断处理程序;

7A——如果系统中有中断任务,则激活并执行中断任务;

7B——如果系统中没有中断任务,则将控制返回到被中断的任务;

8——执行完中断任务后返回,将控制交于被中断任务。

下面详细讨论与中断有关的一些概念,包括中断处理、中断级、中断描述符表等。

当前计算机都使用可编程中断控制器实现外部事件对 CPU 的中断。PC 中主要使用 8259A 可编程中断控制器(PIC)。一个 8259A 可以管理 8 个外部中断源。一台主机中至少有一个中断控制器直接与 CPU 相连,它称为主 PIC,其所管理的中断源称为主中断源。每个中断源可以直接连接外部设备,也可用于与另一个 8259A 中断控制器相连,从而扩展出 8 个中断源。这个扩展的 8259A 称为从中断控制器,从中断控制器提供的中断源称为从中断源。一般,主机系统时钟总是接主 PIC 的主中断源而不接从中断源。因此,iRMX 操作系统可以管理其他 7 级主中断源扩展出的 56 个中断源。但是,与一般计算机系统一样,预配置的 iRMX 操作系统只配置了一个从中断控制器,该系统中共有 16 个中断源。

每个中断源都对应着一个号，称为中断级，中断级既表明了该中断源的优先级，同时又是每个中断源的名称。不同的操作系统中该名称是不同的。在 iRMX 操作系统中，主 PIC 的中断级号为 M0～M7；从 PIC 中断级号为 X0～X7，其中 X 表示主级序号，根据从 PIC 与第几级主 PIC 相连而决定。如与 M7 相连的从 PIC 的中断级号为 70H～77H，其中第 7 级中断源对应的中断级号为 77H。

中断级号越小，中断优先级越高。多个中断源同时申请 CPU 时，优先响应当前优先级最高的中断，即处理器调用相应的中断处理程序。中断处理程序的地址是按优先级定位在称为中断描述符表的内存中。

中断描述符表最多可有 256 个入口，每个入口包含着一个中断处理程序的物理地址，即程序第一条执行指令的地址。系统占用的中断源的处理程序地址在配置系统时由系统管理员给定；应用程序中所包含的外部中断处理程序由程序员作为系统调用 SET $ INTERRUPT 的参数给定。

IDT 不暂定位在内存哪块地址中，其格式及入口的相对定位都是确定的。一般地，0～16 主要由 SDM 和系统异常处理程序产生的内部中断占用；17～55 由系统保留作为扩展用；56～127 由外部中断使用。

iRMX 系统中每一级中断都可以屏蔽，即被禁止，使 CPU 不响应外部设备发出的中断请求，但系统时钟中断(M0)不屏蔽。

禁止中断的方法有两种：一种是由核心程序中断管理调度决定的；另一种是通过系统调用实现的。

基于优先级的抢占式调度方法决定，优先级较高的中断处理程序正在执行时，将自动禁止较低级的外部中断，直至高优先级的处理程序执行完毕打开较低级的中断(有效)，并释放处理器。更高优先级的中断源发出信号请求时，可以中断当前运行任务而抢占 CPU，并禁止低级中断。

iRMX 系统中有两种中断服务方式：中断处理程序方式和中断任务方式。选用哪种方式主要取决于中断服务的内容，也可以将一个中断的服务内容分成两部分，分别由中断处理程序和中断任务处理。中断处理程序只能做以下几个有限的系统调用：ENTER $ INTERRUPT、EXIT $ INTERRUPT、GET $ LEVEL、DISABLE、SIGNAL $ INTERRUPT。如果这些调用不能满足服务要求，则应该安排一个中断任务来进一步服务。

中断处理程序运行时，中断信号位是 1，表示禁止本级中断，直到退出中断处理程序或激活一个中断任务，使中断信号复位。因此，当中断处理工作内容较多时，应激活一个中断任务，以便尽早退出中断处理，及时打开被禁止的中断级。中断处理程序与中断任务的相互关系及流程可用图 5-6 表示。

中断任务的优先级由核心程序根据对应的中断级指定，比一般任务优先级高。此外，它与其他任务一样，有自己的资源、按优先级调度。

5.1.4　操作系统对运动控制器的影响

早期 PC 的操作系统 MS-DOS 是单用户的操作系统，某一时刻只能有一个任务处于活动状态，不具备并发功能。它的单任务弱实时性主要表现在 DOS 内核的“不可重入”，即操作系统核心不可在任意点处中断后，使其代码为另一程序重新使用。例如，若 CPU 当前正

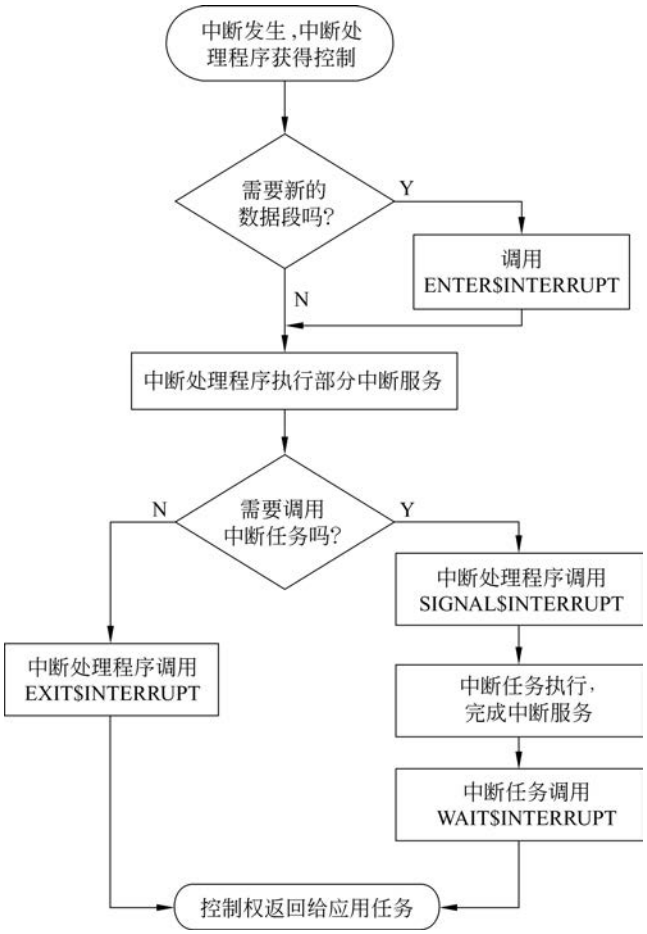


图 5-6 中断处理程序与中断任务的相互关系及流程

在运行的代码是 DOS 的系统功能调用(INT 21H),此时又出现了一个中断请求,则 CPU 暂停原代码的执行,转到中断例程,如果该例程也要调用 INT 21H,就会扰乱 DOS 内部堆栈,引起系统瘫痪或不可预料的行为。要想避免这种情况,一是将中断例程中调用 DOS 的代码暂缓执行,这显然不能满足实时性要求;二是中断例程只能用汇编语言编写,并且不能调用 DOS 功能,这使软件开发费时费力,编程的难度和工作量增加。由此可见,这种硬件中断方式弊多利少,并且难以保证可靠性。

Windows 是当今流行的多任务操作系统,其多任务处理可分为两种模式:协同式多任务和抢先式多任务处理模式。使用协同式多任务处理模式,CPU 的分配直接由应用程序决定,所以系统性能的好坏取决于应用程序设计的好坏。而运行抢先式多任务处理模式,各进程占用 CPU 的时间由系统调度程序决定,当调度程序检测出有比当前任务优先级更高的程序事件后,暂停当前任务并将 CPU 时间分配给优先级更高的进程。所有的 Win16 应用程序都在同一台虚拟机上运行,各程序之间采用的是协同式多任务处理;每个 Win32 应用程序和 MS-DOS 应用程序都具有自己专用的虚拟机,其进程按抢先式多任务处理模式运行。

Windows 中调度的任务有两种状态：运行状态和等待状态。正在运行的任务处于运行状态，当该任务把 CPU 控制权交给其他任务后，就被置为等待状态。为了便于任务调度和保存，每个任务运行的参数，Windows 在装载应用程序时，由 Load Module()函数创建一任务数据库(简称 TDB)，该任务数据库在内存中以链表的形式存在，TDB 链表中的节点记录了每个任务切换时的堆栈指针、中断处理程序地址及此任务对应的模块句柄和实例句柄等。

Windows 就是根据 TDB 链表中存放的各个任务的数据信息来完成任务调度的。对 Windows 而言，CPU 属于临界资源，在某一时刻只有一个任务独占 CPU。为了合理分配 CPU，提高系统的性能，Windows 根据任务是否存在等待事件进行任务调度。如果当前任务没有事件可供处理，那么就应该把控制权交给其他具有等待事件的任务。为了记录每个任务的等待事件个数，在 TDB 链表中，为每个任务建立一个事件计数器(TDB 偏移 6 处的值)，系统调度程序就是根据该计数器来进行任务调度。我们可以通过 Windows 中未公开的核心函数 Post Event()把指定任务的事件计数器值增 1，人为地伪造一个事件，引起相应的任务被唤醒。当有多个任务都具有等待事件时，Windows 采用的最高优先级(HPF)算法进行调度。为此 Windows 定义了任务的优先权值：范围是－32～＋15，任务的优先权值越小，它的优先级就越高，其任务节点在 TDB 链表中的位置也越靠前。而在实际应用中，绝大多数 Windows 应用程序的优先值都是 0，若优先权值相同，则按先来先服务的原则进行。任务的优先权值只对具有等待事件的任务生效，如果一个任务没有等待事件，即使优先权再高，也不会被调度。

若当前任务所分配的时间片用完或当前任务再无等待事件，任务调度程序就要释放当前任务的控制权，把控制权交给已选中的可调度任务。但在 Windows API 中并没直接公开这样一个函数，许多具有释放控制权功能的函数都被隐藏在 Get Message()或 Peek Message()这样的消息函数中，当应用程序在消息环中没有消息可供接收时，它就会把控制权交给别的任务，以防止当前任务进入死循环。当前任务释放控制权以后，如果没有一个任务被调度程序选中，则调度程序就会使 Windows 进入系统空闲状态，相应的电源管理软件就会使整个系统处于低能耗的睡眠状态，直至有任务被唤醒而重新开始正常运转。

由此，我们看到 Windows 操作系统不能实现“确定性时间访问”这个计算机控制的本质要求。

iRMX 操作系统将所要完成的任务划分为两类：前台任务与后台任务。前台任务是必须及时完成的任务，而后台任务则可以暂缓执行。前台任务主要是数据采集与滤波，数据格式转换与显示，控制程序的运算，控制量的输出，执行机构的状态监测，报警信息显示等。后台任务主要是数据备份，报表打印，键盘命令的查询与识别，在线修正控制参数，数据曲线的图形显示等。

任务的调度是保证前台任务及时完成，必要时顺延后台任务。当定时周期到时，首先完成前台工作，在空余的时间内处理后台工作。如果时间不够，将后台工作推迟在下一个周期的空闲时间完成。用这样的方式实现了计算机控制的本质要求：“确定性时间访问”。

在基于 PC 的伺服运动控制系统中，虽然上位计算机不直接参与“控制”，但上位计算机却在为多轴联动协调与规划，如果这个协调与规划不能与系统采样周期同步，则会使系统运动控制的速度减慢，直至产生意想不到的后果。因此，对于要求较为苛刻的被控对象，上位计算机应选用集中式实时多任务操作系统。

5.1.5 伺服运动控制对控制系统的要求

伺服运动控制是对机械运动部件的位置、速度等进行实时控制管理,使其按照预期的运动轨迹和规定的运动参数进行运动。如针对雕刻机、激光加工机械和 PCB 钻铣床等行业的专用伺服运动控制器、图像伺服控制专用运动控制器、力伺服专用运动控制器等,伺服运动控制对控制系统的要求可以分成如下几种形式:

(1) 点位运动控制。这种运动的特点是仅对终点位置有要求,与运动的中间过程即运动轨迹无关。在加速运动时,为了使系统能够快速加速到设定速度,往往提高系统增益和加大加速度,在减速的末段常采用 S 曲线减速的控制策略。为了防止系统到位后振动,规划到位后,又会适当减小系统的增益。所以,点位运动控制器往往具有在线可变控制参数和可变加减速曲线的能力。

(2) 连续轨迹(轮廓控制)运动控制。应用在数控系统、切割系统的运动轮廓控制中。相应要解决的问题是如何使系统在高速运动的情况下,既保证系统加工的轮廓精度,又保证刀具沿轮廓运动时切向速度的恒定。对小线段加工时,有多段程序预处理功能。

(3) 同步运动控制。指多个轴之间的运动协调控制,可以是多个轴在运动全程中进行同步,也可以是在运动过程中的局部有速度同步,主要应用在需要有电子齿轮箱和电子凸轮功能的系统控制中。工业上有印染、印刷、造纸、轧钢、同步剪切等行业。相应的运动控制器的控制算法常采用自适应前馈控制,通过自动调节控制量的幅值和相位,来保证在输入端加一个与干扰幅值相等、相位相反的控制作用,以抑制周期干扰,保证系统的同步控制。

5.2 PC 与伺服运动控制器的信息交换

PC 通过其内部(并行)总线(Backplane Busses,又称母板总线)与外部交换信息,并行总线安装在主机箱内,它是一块印制电路板,板上有若干条插线槽,插有 CPU 主板、显示卡、存储器和 I/O 电路的印制板等,伺服运动控制器就插在此母板的扩展槽中。总线支持各插件板之间的通信,总线一般为 ISA 和 PCI 总线。母板与插件板都有规定的尺寸与形状,固定在主机箱上。

5.2.1 ISA 总线与 PCI 总线

1. ISA 总线

ISA(Industry Standard Architecture)总线,又称 AT 总线,是由 Intel 公司、IEEE 和 EISA 集团联合开发的与 IBM-PC/AT 原装机总线意义相近的系统总线,它具有 16 位数据宽度,最高工作频率为 8MHz,数据传输速率达到 16MB/s,地址线 24 条,可寻访 16M 字节地址单元。它是在早期的 62 线 PC 总线基础上再扩展一个 36 线插槽形成的。分成 62 线和 36 线两段,共计 98 线。其 62 线插槽的引脚排列及定义与 PC 总线兼容。

ISA 总线信号分为总线基本信号、总线访问信号及总线控制信号。总线基本信号主要用来提供基本定时时钟、系统复位、电源和地信号。总线访问信号主要用来提供对总线目标模块访问的地址、数据、访问应答控制信号。总线控制信号的主要功能是提供中断、DMA

处理时的请求及响应信号以及扩展模块主控状态的确定信号 ISA。16 位总线在 ISA 8 位总线基础上把数据线由 8 位扩充到 16 位，把地址线由 20 位扩充到 24 位；并扩充了中断请求信号、DMA 请求与响应信号；还增加了 16 位数据访问的控制信号等。

ISA 总线引脚图如图 5-7 所示，“#”表示低电平有效，否则表示高电平有效。

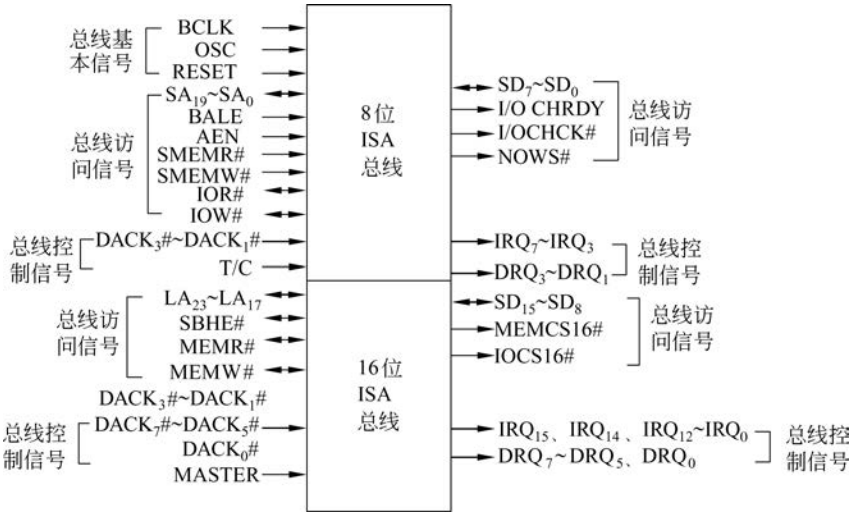


图 5-7　ISA 总线引脚图

2. PCI 总线

图 5-8 为 PCI(Peripheral Component Interconnect)总线引线图，图中“#”表示低电平有效，没有此符号表示高电平有效。

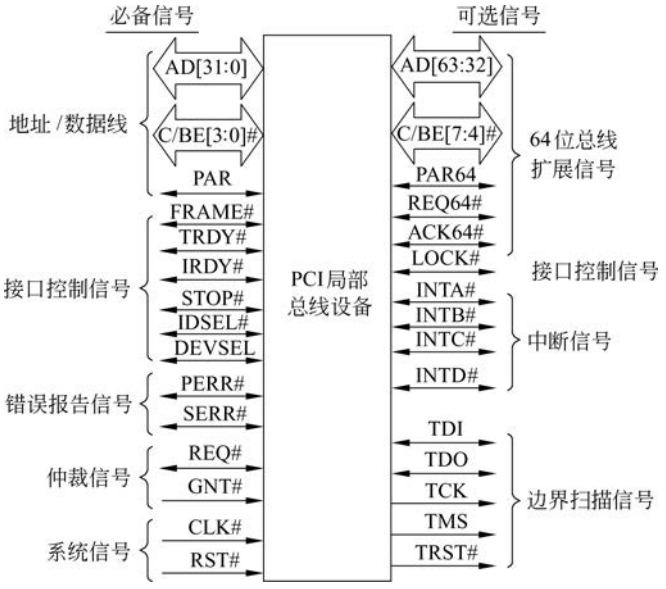


图 5-8　PCI 总线引线图

1) 高速的实现

PCI 总线的地址总线与数据总线是分时复用的,这样做的好处是:一方面可以节省接插件的引脚数,另一方面便于实现突发数据传输。在进行数据传输时,由一个 PCI 设备作发起者(主控,Initiator 或 Master),而另一个 PCI 设备作目标(从设备,Target 或 Slave)。总线上的所有时序的产生与控制,都由 Master 来发起。PCI 总线在同一时刻只能供一对设备完成传输,这就要求有一个仲裁机构(Arbiter),来决定谁有权利拿到总线的主控权。32 位 PCI 系统的引脚按功能分有以下几类:

(1) 系统控制:CLK,PCI 时钟,上升沿有效,RST,Reset 信号;

(2) 传输控制:FRAME#,标志传输开始与结束;

(3) IRDY#:Master 可以传输数据的标志;

(4) DEVSEL#:当 Slave 发现自己被寻址时置低应答;

(5) TRDY#:Slave 可以转输数据的标志;

(6) STOP#:Slave 主动结束传输数据的信号;

(7) IDSEL:在即插即用系统启动时用于选中板卡的信号;

(8) 地址/数据总线:AD[31:0],地址/数据分时复用总线;

(9) C/BE[3:0]#:命令/字节使能信号 PAR,奇偶校验信号。

当 PCI 总线进行操作时,Master 先置 REQ#,当得到 Arbiter 的许可时(GNT#),会将 FRAME#置低,并在 AD 总线上放置 Slave 地址,同时 C/BE#放置命令信号,说明接下来的传输类型。所有 PCI 总线上的设备都需对此地址译码,被选中的设备要置 DEVSEL#以声明自己被选中。然后当 IRDY#与 TRDY#都置低时,可以传输数据。当 Master 数据传输结束前,将 FRAME#置高以标明只剩最后一组数据要传输,并在传完数据后放开 IRDY#以释放总线控制权。

可以看出,PCI 总线的传输是很高效的,发出一组地址后,理想状态下可以连续发数据,峰值速率为 132MB/s。从数据宽度上看,PCI 总线有 32 位、64 位之分;从总线速度上看,有 33MHz、66MHz 两种。目前流行的是 64 位、64MHz 芯片,其数据传输速率可以达到 200MB/s 的连续传输。

2) 即插即用的实现

所谓即插即用,是指当板卡插入系统时,系统会自动对板卡所需资源进行分配,如基地址、中断号等,并自动寻找相应的驱动程序,而不像旧的 ISA 板卡,需要进行复杂的手动配置。实际的实现远比说起来要复杂。在 PCI 板卡中,有一组寄存器,叫“配置空间”,用来存放基地址与内存地址,以及中断等信息。以内存地址为例,当上电时,板卡从 ROM 里读取固定的值放到寄存器中,对应内存的地方放置的是需要分配的内存字节数等信息。操作系统根据这个信息分配内存,并在分配成功后在相应的寄存器中填入内存的起始地址,这样就不必手工设置开关来分配内存或基地址了,对于中断的分配也与此类似。

3) 中断共享的实现

ISA 卡的一个重要局限在于中断是独占的,而计算机的中断号只有 16 个,系统又用掉了一些,这样当有多块 ISA 卡要用中断时就会有问题了。PCI 总线的中断共享由硬件与软件两部分组成。硬件上,采用电平触发的办法:中断信号在系统一侧用电阻接高,而要产生

中断的板卡上利用三极管的集电极将信号拉低，这样不管有几块板产生中断，中断信号都是低，而只有当所有板卡的中断都得到处理后，中断信号才会回复高电平。

软件上，采用中断链的方法：假设系统启动时，发现板卡 A 用了中断 7，就会将中断 7 对应的内存区指向 A 卡对应的中断服务程序入口 ISR_A，然后系统发现板卡 B 也用中断 7，这时就会将中断 7 对应的内存区指向 ISR_B，同时将 ISR_B 的指针指向 ISR_A。以此类推，就会形成一个中断链。而当有中断发生时，系统跳转到中断 7 对应的内存，也就是 ISR_B。这时 ISR_B 就要检查是不是 B 卡的中断，如果是，要处理，并将板卡上的拉低电路放开；如果不是，则呼叫 ISR_A。这样就完成了中断的共享，如图 5-9 所示。

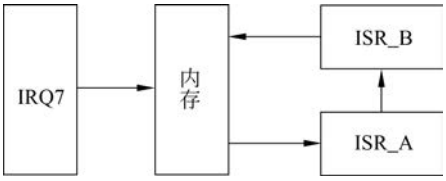


图 5-9　中断共享的软件实现

4）可靠性的实现

PCI 总线与原先 PC 常用的 ISA 总线相比，增加了奇偶校验错(PERR)、系统错(SERR)、从设备结束(STOP)等控制信号及超时处理等可靠性措施，使数据传输的可靠性大为增加。

5）PCI 接口电路对逻辑器件的要求

PCI 总线是 CMOS 总线，在传输信号消失后，稳态电流是很小的，大多数电流消耗在上拉电阻上。PCI 总线基于反射波信号而不是入射波信号。总线无终端的特性导致传输线终端的反射波与入射波叠加后达到所需的电平值。PCI 规范要求集成电路的引脚有上拉电阻以防止振荡或在输入缓冲器上的功率消耗。除此之外要求芯片的输入引脚有箝位特性，PCI 对复杂可编程逻辑器件的要求如下：芯片引脚具有上拉电阻；专用输入和 I/O 引脚具有箝位功能。物理器驱动 TDAT 和 TADD，而其他处理器均采样 TADD 线，确认 TADD 地址线上的地址是否与本地址一致。如果一致，进行读操作，将接收的值从 TRSR 传递到 TRCV，并产生接收中断请求，表明 TRCV 已接收到一个有效数据，CPU 可以来读取。

5.2.2　双口 RAM

要实现 PC 和伺服运动控制器之间的数据传送，必须在 PC 和伺服运动控制器的 DSP 之间建立起双向的数据交换通道。

透过 ISA 总线实现数据交换有以下两种方法：

(1) 静态数据传送，采用并行接口器件 8255 等或锁存器如 74LS373 等构成一字节深度的 FIFO，这种方法较为简单，但一次传输的数据量小，只适应数据量小、速度要求不高的场合；

(2) 共用伺服运动控制器上的外部数据存储器，此时可直接采用双口 RAM 或通用 RAM 加上一些控制逻辑组成的双口 RAM 电路。

双口 RAM 是一种性能优良的快速通信器件，适用于多 CPU 分布式系统及高速数字系统的场合。它提供了两路完全独立的端口，每个端口都有完整的地址、数据、控制线。对器件两边的使用者而言，它与一般 RAM 并无大的区别，只有在两边同时读写同一地址单元时，才发生争用现象。利用双口 RAM 提供的指示信号，采用适当的通信规则，可以避免争用，实现快速数据交换。

5.2.3 IDT71321 应用举例

以在上位 PC 与单片机 8031 之间选用 2K×8 位的带中断请求信号 $\overline{\text{INT}}$ 和忙信号 $\overline{\text{BUSY}}$ 的 IDT71321 为例。

上位 PC 对接口的寻址方式有存储器寻址方式和 I/O 寻址方式。

存储器寻址方式可以有较大的地址空间,可达 32K 甚至 64K,且指令丰富,可以实现快速传送。但其选在高端内存块 UMB(640K～1M 范围),由于 UMB 内存分配已比较拥挤,选择不当就会引起地址冲突导致死机。

当双口 RAM 的容量为几 K 时可以使用 I/O 寻址方式。I/O 寻址方式使用专门的指令,指令明确,但一次只能传送一个字节或字。这里采用 I/O 寻址方式,使用的控制线为 $\overline{\text{IOW}}$ 和 $\overline{\text{IOR}}$。PC 的 I/O 寻址范围为 64K,有 16 根地址线。主板和 OEM 生产商生产的 I/O 扩展卡通常只对地址线 A_0～A_9 译码,即只定义了其中的 1K 寻址空间。当 I/O 指令的地址高 6 位不同而低 10 位相同时会选中同一个 I/O 地址。且 PC 内部由于软盘驱动器、串行口、并行口等用掉了一些空间,剩下的空间并不连续,且远远不足双口 RAM 所需要的 2K 地址空间,因此必须通过高 6 位地址来扩展地址空间。

扩展地址的方法如图 5-10 所示,将 A_{15}～A_0 和 A_{10L}～A_{0L} 非对应连接,将可以任意编址的 A_{15}～A_{10} 和 A_4～A_0 共 11 根地址线组成 2K 的地址空间,与 IDT71321 的 A_{10L}～A_{0L} 一一相连,将剩下的 A_9～A_5 作为基地址码,经数据译码器接地址拨码开关,这样可灵活地选择空闲的 I/O 端口地址。显然这样构成的地址空间在 PC 中是不连续的,各个小段相互隔开。当 PC 侧访问双口 RAM 时,应先将物理地址换码,以得到正确的 I/O 地址。换码公式是:设双口 RAM 地址为 X,则 PC 侧地址为 400H×N_1+340H+N_2,其中 $N_1=X/20$H(取整),$N_2=X\%20$H(求余)。一般 330H～360H 是用户可以自由使用的一块 I/O 空间,这里所加的 340H 是为了避免与计算机系统占用的资源相冲突。这时 PC 侧空闲的 A_9～A_5 译码为 11010。

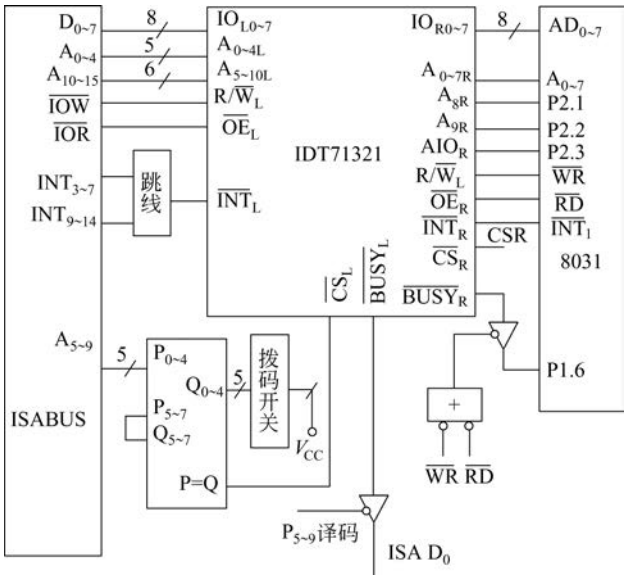


图 5-10 集成双口 RAM 的数据交换电路

双口 RAM 通信方式的关键是处理好争用现象，避免因此产生的读写错误。IDT71321 提供了中断判优和硬件判优两种方式，分别利用 $\overline{INT}$ 和 $\overline{BUSY}$ 引脚信号。本节同时采用这两种方式，以提高系统灵活性。在硬件电路图 5-10 中，IDT71321 内含硬件判优电路，两边端口都有 $\overline{BUSY}$ 引脚。当两端口争用同一地址单元时，由片内硬件电路，根据两边的地址、片选、读写信号到达的先后顺序，裁决哪个端口有使用权。IDT71321 的 $\overline{BUSY}$ 仲裁逻辑真值表如表 5-2 所列。如左端口优先使用，则自动将右端口的 $\overline{BUSY}_R$ 信号拉为低电平，通知右侧暂停读写操作。

表 5-2　IDT71321 的 $\overline{BUSY}$ 仲裁逻辑真值表

左　端　口					右　端　口					功　能	注：
R/W_L	CE_L	OE_L	$A_{10L}\sim A_{0L}$	$\overline{INT}_L$	R/W_R	CE_R	OE_R	$A_{10R}\sim A_{0R}$	$\overline{INT}_R$		
L	L	×	7FF	×	×	×	×	×	L	置右 $\overline{INT}_R$ 标志	H：高电平
×	×	×	×	×	×	L	L	7FF	H	清右 $\overline{INT}_R$ 标志	L：低电平
×	×	×	×	L	L	L	×	7FE	×	置左 $\overline{INT}_L$ 标志	×：未定
×	L	L	7FE	H	×	×	×	×	×	清左 $\overline{INT}_L$ 标志	

由此可知 $\overline{BUSY}$ 信号可直接接至支持插入等待时序的 CPU 如 80196 的 $\overline{READY}$ 引脚，此时无须软件支持。在本例中，由于 8031 没有 $\overline{READY}$ 信号，当 8031 发出读写 IDT71321 命令时，锁存 $\overline{BUSY}_R$ 信号，读 P1.6 口的值就可判断刚才对 IDT71321 读写时是否存在冲突。当 P1.6 的值为 1 时，刚才询问不存在冲突，当 P1.6 的值为 0 时，刚才询问存在冲突，那么就要重发读写 IDT71321 的命令。由于 ISA 总线没有 $\overline{READY}$ 信号，也没有通用的 I/O 引脚，故考虑将来自 IDT71321 的 $\overline{BUSY}_L$ 信号接至数据线的最低位 D_0。因除了查询 $\overline{BUSY}_L$ 引脚电平时外，$\overline{BUSY}_L$ 不应接到 D_0，这时应采用三态门 74LS125。$\overline{BUSY}_L$ 接三态门输入端，三态门输出端接 ISA 总线数据线 D_0 位，其门控信号由 $A_9\sim A_5$ 产生，故占用 ISA 总线一个 I/O 端口。应该选用一个空闲的端口号，这里可选用 330H（双口 RAM 侧，PC 侧须换码），即将 $A_9\sim A_5$ 译码为 11001。具体电路如图 5-11 所示。

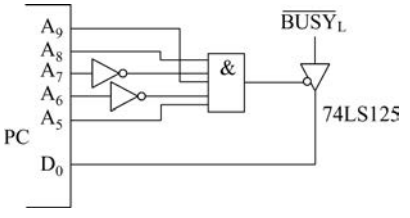


图 5-11　PC 侧硬件仲裁电路

两个中断引脚 $\overline{INT}$ 专为用于端口和端口之间的通信。当用户使用中断功能时，每个端口各分配一个固定的双口 RAM 单元，称为信箱或消息中心。左端为 7FEH，右端为 7FFH。其时序逻辑是当右端写 7FEH 单元时，左 $\overline{INT}_L$ 变低触发中断，左端口读 7FEH 时，将 $\overline{INT}_L$ 置 1 清中断。左端情况类似，只是 7FFH 单元操作。当使用中断判优时，右端口不应写 7FFH 单元，左端口不应写 7FEH 单元，两单元中按约定置入用户握手信息，在不采用中断方案时两单元是普通双口 RAM。可见中断判优方式需要软件配合。ISA 总线提供了 3～7、9～14 共 11 个中断，$\overline{INT}_L$ 可经过跳线与上述 11 个中断相连，这样上位 PC 可以灵活地选择空闲的中断。$\overline{INT}_R$ 直接与 8031 的 $\overline{INT}_1$ 相连。8031 的另一个中断可用于来自 CAN 控制器的中断。当接收到 CAN 控制器有中断信号时，8031 查询 CAN 控制器的中断寄存器以判断是哪一类中断，进行相应的响应。

双口 RAM 主要承担上位计算机和 8031 之间的数据交换任务,其软件设计也包括两部分:

(1) PC 端的应用程序接口(API)函数,负责完成 PC 端与双口 RAM 之间的通信。主要包括向双口 RAM 发送控制命令、数据命令和请求数据命令,还将接收到的数据进行后处理如分析、显示、报表等。

(2) 8031 端的程序设计,负责 8031 与双口 RAM 之间的通信。主要包括接收来自 PC 的命令,将由 8031 接收的被控对象的数据、状态信息送到双口 RAM。

5.2.4 基于 PC ISA 的运动控制板卡

带有标准 ISA 总线的基于 DSP 和 FPGA 的运动控制卡,可提供 4 轴闭环伺服控制,8 路光电隔离限位开关信号输入,4 路光电隔离原点开关信号输入,16 路光电隔离通用信号输入,16 路光电隔离通用信号输出,其功能框图如图 5-12 所示。其中,DSP 为 TI 公司的 TMS320L F2407A,FPGA 为 AL TERA 公司的 EP1 K30QC208,DAC 为 ADI 公司的 AD1866。

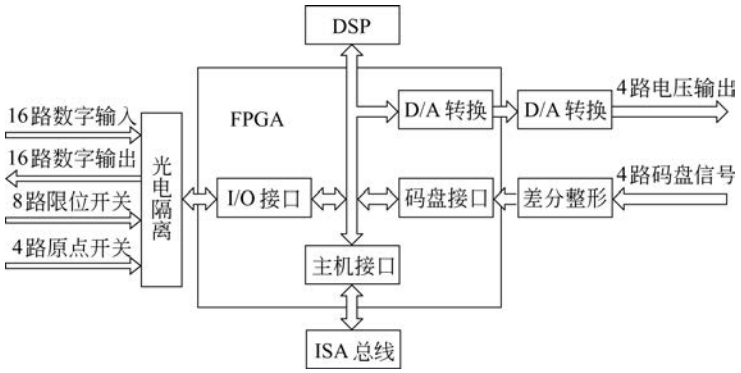


图 5-12 运动控制卡功能图

1. DSP 模块

TMS320L F2407A 是 TI 公司的 16 位定点 DSP,针对工业控制而开发。其主要特点是:采用静态 CMOS 技术,供电电压为 3.3V,最高 40MIPS 的指令执行速度。基于 TMS320C2xx 的 CPU 内核,与 TMS320 系列 DSP 代码相兼容。片内 32KB 的 Flash 程序存储器,115KB 的数据/程序 RAM,544KB 的双口 RAM,2K 的单口 RAM。两个事件管理器模块 EVA 和 EVB,每个模块包括 2 个 16 位通用定时器、8 个 16 位的 PWM 通道、3 个捕获单元及 16 通道 A/D 转换器。可扩展外部存储器空间为 192K。其中,64KB 的程序存储器空间,64KB 的数据存储器空间,64KB 的 I/O 空间。具有 SCI、SPI、CAN 接口等功能。

由于 TMS320L F2407A 具有多总线、多处理单元、流水线、硬件乘法器等结构,使其具有了高速数据处理和逻辑控制能力,能够较好地完成较为复杂的控制算法。在此,其主要任务是完成位置控制和速度控制的 PID 调节。位置控制分为 S 曲线和 T 曲线模式。当进行位置控制时,DSP 根据 PC 传过来的指令,按照 S 曲线或 T 曲线模式,将每个伺服周期内产生理论的位置、速度、加速度、加加速度与根据码盘反馈回来的信号而确定的位置、速度、

加速度、加加速度进行比较，从而进行 PID 调节。DSP 通过其 I/O 地址空间对 FPGA 内部的各模块进行统一编址，要访问某一模块，只要对相应的 I/O 地址进行读操作或写操作即可。

2. FPGA 模块

EP1 K30QC208 是 ALTERA 公司的 ACEX1 K 系列 FPGA 中的一款，采用可重构的 CMOS SRAM 工艺，把连续的快速通道(Fast Track)与独特的嵌入式阵列(EAB)相结合，同时也结合了众多可编程器件的优点来完成普通门阵列的宏功能，可提供可编程单芯片系统(SOPC)集成。其特点是 30 000 个典型可用门，1728 个逻辑单元，6 个嵌入式阵列块，总共 24 576 位的内部 RAM(每个 EAB 有 4096 位)，171 个用户 I/O 引脚，4 个专用输入引脚，2 个全局时钟输入引脚。支持多电压接口，允许输入/输出引脚电压 3.3V 和 5.0V，支持内部三态。内带 JTAG 边界扫描测试电路等特点。EP1 K30QC208 主要完成 4 个模块的逻辑功能，即主机接口模块，码盘接口模块，D/A 接口模块，I/O 接口模块。单片大容量 FPGA 的使用，一方面使整个系统得到了集成，提高了系统的可靠性，另一方面也解决了因 TMS320L F2407A 为 3.3V 供电不能与 5V 器件相兼容而导致的逻辑电平不兼容的问题。下面分别对 4 个模块进行介绍。

1) 主机接口模块

主机接口模块的主要任务是完成 PC 和 DSP 之间的数据交换。这里，把 PC 和 DSP 之间传送的指令分为两部分：指令代码和指令参数。指令代码代表指令的具体含义，指令参数是指令所带有的参数。在主机接口模块中用指令代码寄存器和指令参数寄存器分别存放指令代码和指令参数，PC 和 DSP 均可对其进行读或写。二者通信的基本原理是：PC 向 DSP 写参数。PC 先检查 BUSY 信号，若 BUSY 为 0，则不能写，若 BUSY 为 1，则 PC 可向指令代码寄存器写入指令代码。然后，BUSY 被置 0，同时产生中断信号向 DSP 申请中断。DSP 响应中断，检查到是指令代码中断，则读取指令代码，再根据指令代码确定 PC 欲写入的指令参数的位数，从而初始化相应的指令参数寄存器。然后，把 BUSY 置 1，同时清除中断信号。

PC 向指令参数寄存器写入指令参数的基本过程与写入指令代码相同。至此，DSP 得到了一个完整的指令，即可执行相应的功能。

PC 从 DSP 读参数的过程和上述过程基本相同，区别在于 DSP 向指令参数寄存器写入指令参数，而 PC 从指令参数寄存器中读取指令参数。

2) 码盘接口模块

经差分整形后的每路码盘的 A、B、Z 三相信号进入码盘接口模块。由于 4 路码盘信号性质相同，故这里只对一路码盘进行说明。对 A、B、Z 三相信号分别进行滤波以消除干扰信号的影响；对 A、B 两相信号进行 4 倍频，同时进行鉴相确定出 DIR；根据 DIR 对 4 倍频的脉冲进行加计数或减计数。可由 Z 相信号或原点信号对计数器进行瞬时捕捉，将捕捉到的值存入位置捕捉寄存器。计数器和位置捕捉寄存器均为 32 位，DSP 可对其进行读取或清零。

3) D/A 接口模块

由于 AD1866 是串行输入，同时又因 D/A 转换模块提供 4 路电压输出，故 D/A 接口模块的功能有：把来自 DSP 的 16 位并行数据转换成相应的 16 位串行数据；根据译码电路产

生 4 路 DAC 中某一路的片选信号。

4) I/O 接口模块

I/O 接口模块的主要功能是对 I/O 进行统一管理,包括 16 路通用数字输入、16 路通用数字输出信号、8 路限位开关信号和 4 路原点信号以及 FPGA 内部的 4 位 0 信号。同时,原点信号也作为码盘信号位置捕捉的激发信号之一(另一信号为码盘的 Z 相信号。)

3. D/A 转换模块

由两片 AD1866 和一片 OP497 构成 D/A 转换模块,提供 4 路模拟电压输出,电压输出范围均为±10V。

4. 其他模块

差分整形模块将 4 路增量式码盘信号(每路均为 A、B、Z 三相的差分信号)处理后变为 A、B、Z 三相的单端信号,再进入 FPGA 的码盘接口模块进行相应的处理和应用。

光电隔离模块将 I/O 信号进行隔离后与 FPGA 的 I/O 接口模块相连,以消除内部系统和外部系统之间的影响。

5.2.5 基于 PC PCI 的运动控制卡

1. PCI 局部总线接口设计

PCI 总线既能满足当今技术的要求,又能满足未来发展的需要,是业界公认的最具有发展前途的总线。在 PCI 应用系统中,目标设备至少需要 47 条信号线,主控设备至少需要 49 条信号线,利用这些信号便可以处理数据、地址,实现接口控制、仲裁及系统功能,具体内容请参阅相关文献。但 PCI 总线协议复杂、信号很多,且这些信号无法直接与通用目标设备接通,必须首先根据协议对信号进行转换完成接口设计。

第一步的工作是将 PCI 总线信号转变为目标设备(运动控制卡)所需要或可以接收的接口信号。

PCI 的工作是从总线主控器宣布 FRAME ＃开始的,以后 PCI 总线上的全部其他设备便成为作业目标器,并对主控器放在总线上的地址和命令进行锁定,总线上的每个设备都对地址进行解码,以判定是否为一个匹配的地址。如果设备检测到匹配地址时,便宣布 DEVSEL ＃信号,以申请作业。这时目标器准备好向主控器传送数据或从主控器接收数据。主控器和目标器都在宣布解除 IRDY ＃、TRDY ＃后,传送数据,由主控器宣布解除 FRAME ＃以表示作业完成。目标器可以宣布 STOP ＃信号而引发作业提前中断。在全部数据传送完成后总线转入等待状态。

在本系统中,通过使用 I/O 端口完成双口存储器的数据读写,因此使得系统的实现大大简化,用到的信号也较少。为了易于说明,现将各部分的设计用原理图给出,在利用 FPGA 实现时,全部用硬件描述语言来编写,更容易实现。

2. PCI 接口运动控制卡设计

运动控制器硬件系统结构如图 5-13 所示。

运动控制卡硬件系统主要包括:

(1) 微处理器系统(处理器、辅助电路、板载存储器),完成系统的管理协调,位置控制。

(2) PCI 通信接口包括 PCI 接口逻辑,高速双端口存储器。

(3) 反馈处理主要包括数字滤波、鉴相、倍频、计数等,获取各轴实际位置和手摇脉冲发

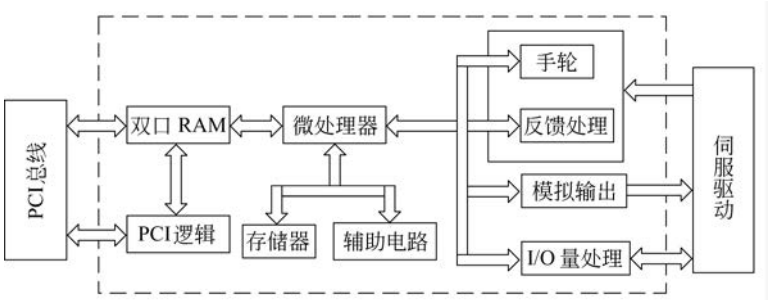


图 5-13　运动控制器硬件系统结构

生器的处理。

所以模拟量输出提供 8 根实轴驱动模块的 8 路－10～＋10V 模拟信号，下面主要讨论位控卡 PCI 通信接口部分的设计，其他部分各种资料文献介绍较多，这里不再赘述。I/O 量处理模块负责对 I/O 量的输入/输出和逻辑处理。

3. 运动控制器 PCI 通信接口设计

运动控制器 PCI 通信接口设计如图 5-14 所示。

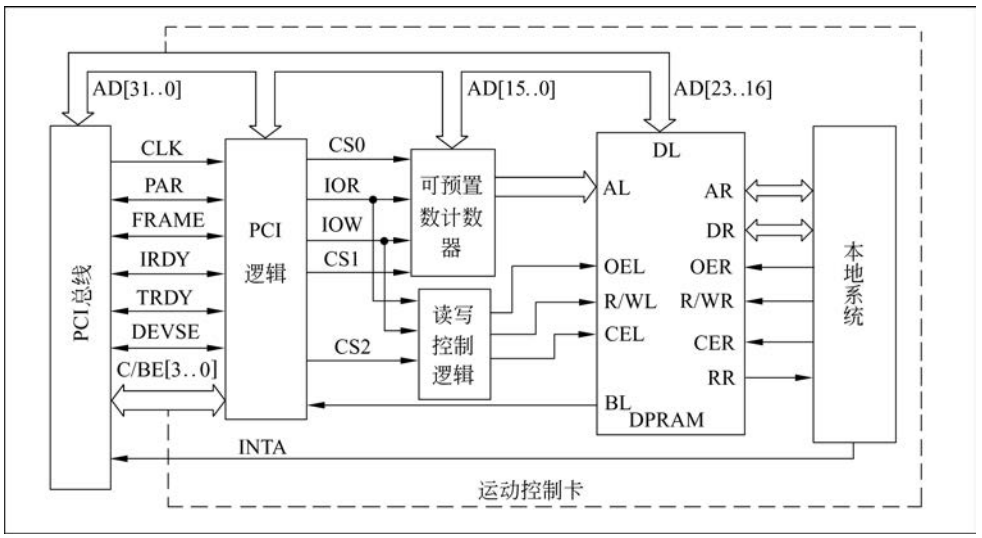


图 5-14　运动控制器 PCI 通信接口设计

PCI 通信接口中包含 4 部分：

(1) PCI 逻辑完成 5.1 节的 PCI 总线接口信号的实现，提供了 PCI 总线信号转变为目标设备(运动控制卡)所需要或可以使用的接口信号，这些信号可以供上位机通过少量 I/O 端口对下位机的存储器进行读写操作，本设计只采用了三个 I/O 地址单元，两个是预置数端口，另一个是数据端口。地址可以通过前面的 IOCS、BCS、IOW、IOR、A0 和 A1 的组合实现。

(2) 可预置数计数器占用 2 个 I/O 端口，数据位数根据不同容量的 DPRAM 的地址线

位数决定。由上位机在访问存储器之前将存储器中的数据区起始地址送预置数端口地址,计数器的计数输出与 DPRAM 的一侧地址线相连,当上位机对数据端口执行一次读写操作,计数器中的计数值就自动加一,指向 DPRAM 的下一个存储单元。预置数端口的地址译码与 C/BE 信号的配合,使得该端口从 AD[15～0]取数。

(3) 读写控制逻辑实现对 DPRAM 的读写使能和时序的控制。读写控制逻辑中的 CS2 非常重要,其信号的译码输入中有 A0、A1 信号,因此保证了与 C/BE 信号的配合,使对 DPRAM 读写的数据位为 AD[23～16]。

(4) 双端口随机存储器(DPRAM)有两套完整的总线结构,提供内部仲裁允许两套总线同时读写,完成数据在两个异步系统中的共享。

通信接口的数据传递由上位机对预置数端口预置起始地址开始,随后上位机对数据端口的反复读写就可以与运动控制卡成批交换数据了。

5.3 伺服运动控制系统的采样周期

5.3.1 信息变换原理

计算机进行运算和处理的是数字信号,而伺服运动控制系统是连续系统,连续系统中的给定量、反馈量及被控对象都是连续型的时间函数,把计算机引入连续系统,造成了信息表示形式与运算形式不同,为了设计与分析计算控制系统,就要对两种信息进行变换。为了对控制系统进行分析与运算,需把伺服运动控制系统变换成能够进行数学运算的结构,如图 5-15 所示。

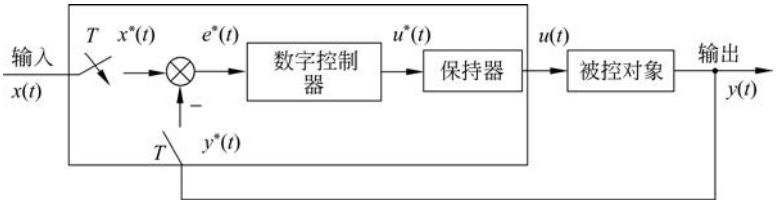


图 5-15 数字控制系统方框图

这里假设 A/D 转换有足够的精度,因此由 A/D 转换器形成的量化误差在数学上是可以不计的,这样可以把采样器和 A/D 转换器用周期为 T 的理想采样开关代替。该采样开关在不同采样时刻的输出脉冲强度(又称脉冲冲量),表示 A/D 转换在这一时刻的采样值。这样采样函数可以用 $x^*(t)$,$y^*(t)$及 $e^*(t)$表示,* 号表示离散化的意思。数字计算机用一个等效的数字控制器来表示,令等效的数字控制输出的脉冲强度,对应于计算机的数字量输出。计算机的输出通道 D/A 转换器的作用是把数字量转化成模拟量,D/A 转换器在精度足够高的情况下(通常也是满足的),数学上可用零阶保持器来代替。如图 5-15 所示为由计算机作为控制器的计算机控制系统,在数学上可以等效成为一个典型的离散控制系统。在上述假定下,分析和研究离散控制系统的方法可以直接应用于数字控制系统。

5.3.2　采样过程及采样函数的数学表示

计算机控制系统中，把一个连续模拟信号，经采样开关后，变成了采样信号，即离散模拟信号，采样信号再经过量化过程才变成数字信号。如图 5-16 所示，图(a)是采样开关，每隔一定时间(例如 T 秒)，开关闭合短暂时间(例如 τ 秒)，对模拟信号进行采样，得到时间上离散数值序列为

$$f^*(t)=\{f(0T),f(T),f(2T),\cdots,f(KT),\cdots\}$$

式中，T 为采样周期，$0T$，T，$2T$，…为采样时刻，$f(KT)$表示采样 K 时刻的数值，由于实际系统，$t\leqslant 0$ 时，$f(t)=0$，所以从 $t=0$ 开始采样是合理的。

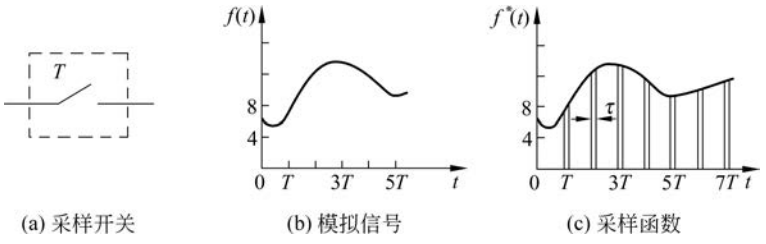


图 5-16　信息的转换过程

如果采样周期 T 比采样开关闭合时间 τ 大得多，即 $\tau\ll T$，而且 τ 比起被控对象的时间常数也非常小，那么认为 $\tau\to 0$。这样为了数学上的分析方便，因为以后要用到的 Z 变换与脉冲传递函数在数学上只能处理脉冲序列，因此引入了脉冲采样器的概念，脉冲采样器工作过程如图 5-17 所示。

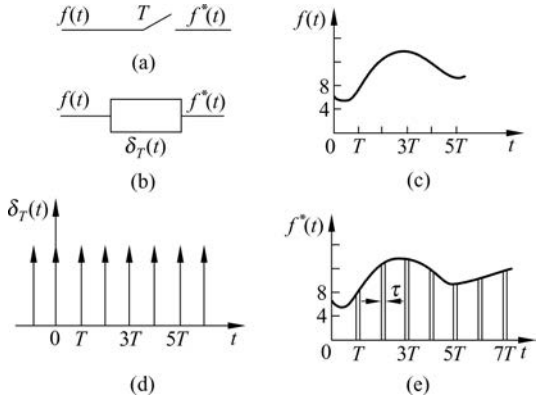


图 5-17　$f(t)$经脉冲采样器的调制过程

如图 5-17 所示，给脉冲采样器输入一个连续函数 $f(t)$，经脉冲采样器调制后输出一采样函数 $f^*(t)$ $\left($图中 $\delta_T(t)=\sum_{k=0}^{\infty}\delta(t-kT)$ 称为单位理想脉冲序列，它是一个以 T 为周期的周期函数$\right)$。采样函数表达式为

$$f^*(t)=f(t)\sum_{k=0}^{\infty}\delta(t-kT) \tag{5-1}$$

式中,k 为整数;T 为采样周期;$\delta(t)$为理想单位脉冲;$\delta(t-kT)$为 $t=kT$ 时刻的理想单位脉冲,定义为

$$\delta(t-kT)=\begin{cases}\infty, & t=kT\\ 0, & t\neq kT\end{cases} \tag{5-2}$$

且冲量为 1,即

$$\int_0^{\infty}\delta(t-kT)\mathrm{d}t=1 \tag{5-3}$$

式(5-2)中,当 $t\neq kT$ 时,$\delta(t-kT)=0$,因此 $f(t)$在 $t\neq kT$ 时的取值大小就没有意义了,所以式(5-1)可以改写为

$$f^*(t)=\sum_{k=0}^{\infty}f(kT)\delta(t-kT) \tag{5-4}$$

这就是理想脉冲采样函数的数学表达式。此式的物理意义可以这样理解:采样函数 $f^*(t)$为一脉冲序列,它是两个函数的乘积,其中 $\delta(t-kT)$仅表示脉冲存在的时刻,冲量为 1,而脉冲的大小由采样时刻的函数值 $f(kT)$决定。

需要指出,具有无穷大幅值和时间为零的理想单位脉冲纯属数学上的假设,而不会在实际的物理系统中产生。因此,在实际应用中,对理想单位脉冲来说,只有讲它的面积,即冲量或强度才有意义,用式(5-3)表示。

式(5-4)中,$f(kT)$是采样值,可以看作是级数求和式里对脉冲序列 $\delta(t-kT)$的加权系数,即 $f(kT)$是 $\delta(t-kT)$在 kT 时刻的脉冲冲量值,或称为脉冲强度。

5.3.3 采样函数的频谱分析及采样定理

采样函数的一般表达式为

$$f^*(t)=f(t)\sum_{k=-\infty}^{+\infty}\delta(t-kT) \tag{5-5}$$

又因为 $\sum_{k=-\infty}^{+\infty}\delta(t-kT)=\delta_T(t)$,$\delta_T(t)$是周期函数,可以展成傅里叶级数,它的复数形式为

$$\delta_T(t)=\sum_{k=-\infty}^{+\infty}C_k\mathrm{e}^{\mathrm{j}k\omega_s t} \tag{5-6}$$

式中,$\omega_s=\dfrac{2\pi}{T}$为采样角频率;C_k 为傅里叶系数,由下式给出:

$$C_k=\frac{1}{T}\sum_{-\frac{T}{2}}^{\frac{T}{2}}\delta(t)\mathrm{e}^{-\mathrm{j}k\omega_s t}\mathrm{d}t$$

因为 $\delta_T(t)$在 $t=0$ 时积分值为 1,所以可得

$$C_k=\frac{1}{T}$$

将 C_k 代入式(5-6),得

$$\delta_T(t)=\frac{1}{T}\sum_{k=-\infty}^{+\infty}\mathrm{e}^{\mathrm{j}k\omega_s t} \tag{5-7}$$

将式(5-7)代入式(5-5),得

$$f^*(t)=\frac{1}{T}\sum_{k=-\infty}^{+\infty}f(t)\mathrm{e}^{\mathrm{j}k\omega_s t} \tag{5-8}$$

且定义 $F(s)$ 是 $f(t)$ 的拉氏变换式$\left[F^*(s)=\int_0^{\infty}f(t)\mathrm{e}^{-st}\mathrm{d}t\right]$则采样函数 $f^*(t)$的拉氏变换式为

$$F^*(s)=\int_0^{\infty}f^*(t)\mathrm{e}^{-st}\mathrm{d}t=\int_0^{\infty}\frac{1}{T}\sum_{k=-\infty}^{+\infty}f(t)\mathrm{e}^{\mathrm{j}k\omega_s t}\mathrm{e}^{-st}\mathrm{d}t$$

所以

$$F^*(s)=\frac{1}{T}\sum_{k=-\infty}^{+\infty}F(s+\mathrm{j}k\omega_s) \tag{5-9}$$

它是采样函数 $f^*(t)$拉氏变换式的一种表达式。可见,采样函数的拉氏变换式 $F^*(s)$是以 ω_s 为周期的周期函数。若令 $s=\mathrm{j}\omega$,直接求得采样函数的傅里叶变换式,即

$$F^*(\mathrm{j}\omega)=\frac{1}{T}\sum_{k=-\infty}^{+\infty}F(\mathrm{j}\omega+\mathrm{j}k\omega_s) \tag{5-10}$$

式(5-10)建立了采样函数频谱与连续函数频谱之间的关系,$F(\mathrm{j}\omega)$为原连续函数 $f(t)$的频谱,$F^*(\mathrm{j}\omega)$为采样函数 $f^*(t)$的频谱,如图 5-18 所示。图 5-18(a)表示了连续函数 $f(t)$的频谱 $F(\mathrm{j}\omega)$是孤立的,非周期频谱,只有在$-\omega_{max}\sim+\omega_{max}$ 有频谱,其外$|F(\mathrm{j}\omega)|=0$,而采样函数 $f^*(t)$的频谱 $F^*(\mathrm{j}\omega)$是采样频率 ω_s 的周期函数,其中 $k=0$ 叫主频谱,除了主频谱外,$F^*(\mathrm{j}\omega)$尚包括$|k|>0$ 的无穷多个附加的高频频谱。

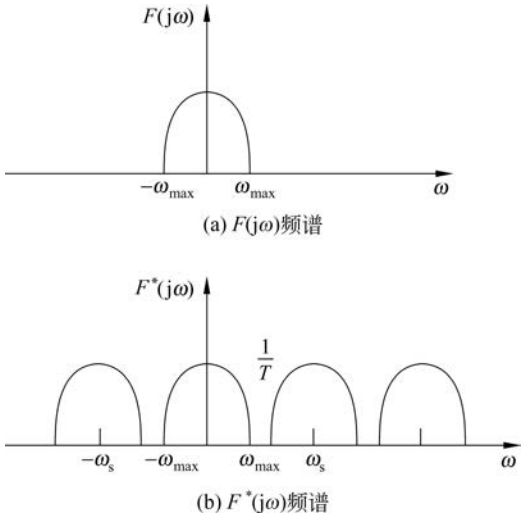


图 5-18　频谱图

频率域内的周期 ω_s 与时间域内的采样角频率$\frac{2\pi}{T}$相等,即关系为

$$\omega_s=\frac{2\pi}{T} \tag{5-11}$$

显然采样周期 T 的选择会影响 $f^*(t)$的频谱、采样定理所要解决的问题是,采样周期选多大,才能将采样信号较少失真地恢复为原连续信号。

当 $\omega_s \geqslant 2\omega_{max}$ 时,即 $T \leqslant \frac{\pi}{\omega_{max}}$时,由式(5-11)知,如图 5-19(a)所示采样信号 $f^*(t)$的频谱是由无穷多个孤立频谱组成的离散频谱。其中主频谱就是原连续函数 $f(t)$的频谱,只是幅值是原来的$\frac{1}{T}$,其他与$|k|>0$ 所对应的频谱都是由于采样过程而产生的高频频谱,如果将 $f^*(t)$经过一个频带宽大于 ω_{max} 而小于 ω_s 的理想滤波器 $W(j\omega)$,滤波器输出就是原连续函数的频谱,说明当 $\omega_s \geqslant 2\omega_{max}$ 时,采样函数 $f^*(t)$能恢复出不失真的原连续信号,这是我们希望得到的。

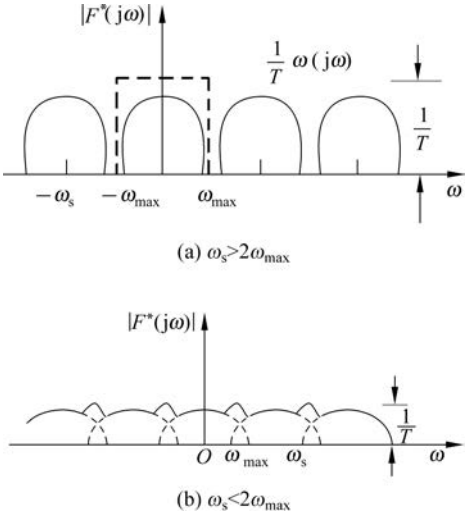


图 5-19 采样信号频谱的两种情况

而当 $\omega_s < 2\omega_{max}$ 时,即 $T > \frac{\pi}{\omega_{max}}$,如图 5-19(b)所示,采样函数 $f^*(t)$的频谱已变成连续频谱,重叠后的频谱中没有哪部分与原连续函数频谱 $F(j\omega)$相似,这样,采样信号 $f^*(t)$就不能通过低通滤波方法不失真地恢复原连续信号了。

因此,对采样周期就要有个限制。为了不失真地由采样函数恢复原连续函数,则要求

$$\omega_s \geqslant 2\omega_{max} \tag{5-12}$$

即

$$T \leqslant \frac{\pi}{\omega_{max}} \tag{5-13}$$

这就是香农(Shannon)采样定理,它给出了采样周期的上限。

5.3.4 采样周期 T 对运动控制器的影响

采样周期 T 的选择是实现计算机控制系统的一个关键问题,采样周期选择不合适,会导致系统动态品质恶化,甚至导致系统不稳定,前功尽弃。但是采样周期的选择至今没有一

个统一的公式，至于香农采样定理只给出了理论指导原则，实际应用还有些问题，主要是系统的数学模型不好精确地测量，系统的最高角频率 ω_{max} 不好确定，况且采样周期的选择与很多因素有关，比较明显的因素如下：

(1) 控制系统的动态品质指标。

(2) 被控对象的动态特性。

(3) 扰动信号的频谱。

(4) 控制算法与计算机性能等。

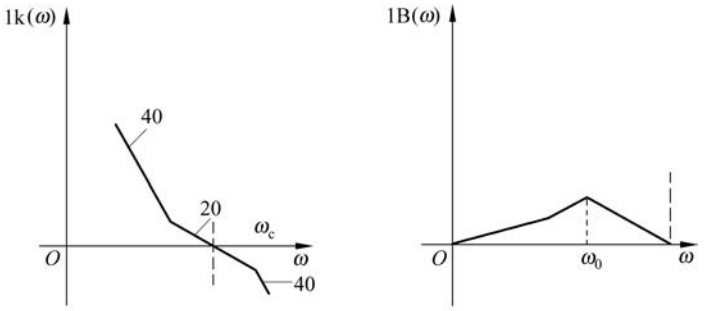
目前采样周期的选择是在一般理论指导下，结合实际对象进行初步选择，然后再在实践中通过实验来确定的。对于伺服运动系统，要求其响应快，抗干扰能力强，采样周期可以根据动态品质指标来选择。假如系统的预期开环频率特性如图 5-20(a)所示，系统的预期闭环频率特性如图 5-20(b)所示。在一般情况下，闭环系统的频率特性具有低通滤波器的功能，当控制系统输入信号频率为 ω_0(谐振频率)时，幅值将会快速衰减，反馈理论告诉我们，ω_0 是很接近它的开环频率特性的截止频率 ω_c，因此可以认为 $\omega_c \approx \omega_0$，这样，我们对被研究的控制系统的频率特性可以这样认为，通过它的控制信号的最高分量是 ω_c，超过 ω_c 的分量被大大地衰减掉了，根据经验，用计算机来实现模拟校正环节功能时，选择采样角频率为

$$\omega_s \approx 10\omega_c \tag{5-14}$$

或

$$T \approx \frac{\pi}{5\omega_c} \tag{5-15}$$

可见，式(5-14)与式(5-15)是式(5-12)和式(5-13)的具体体现。



(a) 系统的预期开环频率特性　　(b) 系统的预期闭环频率特性

图 5-20　频率法分析系统

按式(5-15)选择采样周期 T，则不仅不能产生采样信号的频谱混叠现象，而且对系统的预期校正会得到满意的结果。

在伺服运动系统中，也可以根据系统上升时间而设定采样周期，即保证上升时间内有 2～4 次采样。设 T_r 为上升时间，N_r 为上升时间采样次数，则经验公式为

$$N_r = \frac{T_r}{T} = 2 \sim 4 \tag{5-16}$$

经验表明，伺服运动系统采样周期太大，会产生失真，采样周期太小又会引进测量误差，为了达到好的滤波效果，许多伺服运动系统将控制器输出和编码器输入的周期分开，其中控

制器输出周期可为 200μs,编码器输入采样周期可为 25μs。

5.4 基于 PC 与基于 PLC 运动控制器的比较

在伺服运动控制中,基于 PC 与基于 PLC 的伺服运动控制器经常被人们选用。一般以为,基于 PC 与基于 PLC 的伺服运动控制器因其性能、重点不同而应用于不同的场合,PC 的实时性明显优于 PLC,有些场合仅能使用基于 PC 的伺服运动控制器。工控机与 PLC 性能比较一览表如表 5-3 所示。

表 5-3 工控机与 PLC 性能比较一览表

性 能	分 类	
	工控机 IPC	PLC
操作性	对于硬件的一些编程较为复杂,但可以借助软件功能扩充	采用面向用户的指令,因此编程方便,主要以梯形图为主,程序有通用性
可靠性	工控机能在粉尘、烟雾、高低温、潮湿、振动、腐蚀环境中工作。系统的故障率低,同时其可维修性好,抗干扰性不如 PLC	PLC 采用的 CPU 都是生产厂家专门设计的工业级专用处理器,抗干扰性特别是抗电源干扰能力也较高。系统软件为生产厂家所提供,具有很高的可靠性
移植性	受其自身限制,可移植性质较差	适合多种工业现场,有较好的移植性
工作方式	多为中断处理	串行通道顺序控制,循环扫描
实时性	由于 PC 的采用,实时性明显优于 PLC	PLC 大多都是晶体管输出类型的,这种输出类型的输出口驱动电流不大,决定了 PLC 实时性能不是很高,要受每步扫描时间的限制
复杂控制	充分利用计算机资源,IPC 可用于运动过程、运动轨迹都比较复杂,且柔性比较强的机器	虽说有的 PLC 已经有直线插补、圆弧插补功能,但由于其本身限制,对于诸如伺服电机高速高精度多轴联动,高速插补等复杂动作不太容易实现
适用范围	工控机在中规模小范围自动化工程中有很高的性价比	主要适用于运动过程比较简单、运动轨迹固定的设备,如送料设备、自动焊机等 适合低成本自动化项目和作为大型 DCS 系统的 I/O 站
接线	接线涉及板卡,所以接线较为复杂,而且容易出错,在工业现场中较为不方便	接线较为简单,配置 I/O 模块非常清晰
扩展性	IPC 由于采用底板 CPU 卡结构,因而具有强大的输入/输出功能,能在与运动卡相接的同时,也与工业现场的各种其他外设、板卡等相连,以完成各种任务	要增加一个功能只要增加相应的模块和修正对应的程序,比较方便
故障诊断	利用看门狗技术能在系统出现故障时迅速报警,并在无人干预的情况下,使系统自动恢复运行。但很难诊断出故障元件及原因,使维修周期增加	本身有很强的自诊断功能,一旦系统出现故障,根据自诊断很容易诊断出故障元件,减少了出事故后的故障恢复时间

续表

性　能	分　类	
	工控机 IPC	PLC
多种控制	可以充分发挥计算机的多任务性特点，从而非常好地实现各种控制	尽管现代 PLC 可以实现过程控制，运动控制，顺序控制等，但是它的最大优点还是逻辑控制
可带电机	工控机可以通过增加运动卡来增加控制电机数目	其控制的电机数目受 PLC 自身设计的限制，往往只能带动少数几个电机
控制精度	可以充分发挥自行开发软件的优势，通过改进算法，来实现更精确的运动控制	因 PLC 设计时并不是以运动控制为主要目的，所以其控制精度不高
价格	相对便宜	相对昂贵
占地	体积相对较大	相对较小
主流厂家	上海康泰克、北京康拓和研华、艾讯等	A-B、Siemens 和 Modicon 等
市场占有率	35%左右	30%左右

5.5　基于 PC 的伺服运动控制系统设计分析

图 5-21 是基于 PC 的伺服运动控制系统结构框图，按物理结构可以划分为上位控制装置、交流伺服电机及其驱动器和机械传动机构 3 部分。

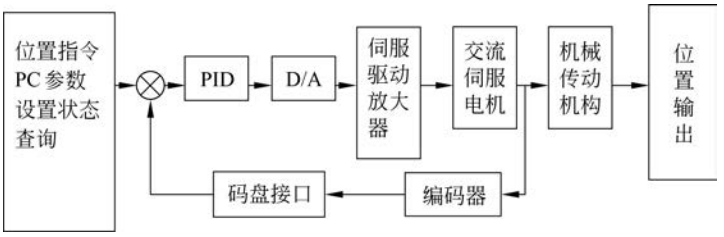


图 5-21　基于 PC 的伺服运动控制系统结构框图

5.5.1　上位计算机的选择

在伺服运动控制过程中，上位计算机实现运动规划、多轴插补、伺服控制滤波等数据运算和实时控制管理。

一般来说，在工业现场需要使用基于 PC 的伺服运动控制系统时，常采用专门为工业现场设计的计算机（工控机）作为上位计算机。这种计算机机箱采用钢结构，有较高的防磁、防尘、防冲击的能力并且便于维护，机箱内有专用底板，一般底板上有多个 ISA 和 PCI 总线插槽，机箱内还设有专门电源，电源有较强的抗干扰能力。

当工业现场条件恶劣，可以考虑使用 CNC 控制器。CNC 控制器除了满足商业利益的封闭性外，以其小巧、专用更能适应工业现场，但是 CNC 控制器本质上与工控机是相同的，都是一种加固的增强型工业个人计算机结构，是一种密闭的基于 PC 的伺服运动控制系统，它可以作为一个模块在工业环境中可靠运行。

实践表明,如果基于 PC 的伺服运动控制系统仅仅运行于实验室环境,使用一般的个人 PC 也能够胜任其工作。注意,这里所说的“实验室环境”主要指电源、温度、湿度、外部干扰等。这也是我们在开发时,经常使用一般个人 PC 的原因。

PC 操作系统也是选择上位计算机的一个重要内容。一般的,在工业现场,应该选择实时多任务或单任务操作系统。但是如果使用了 Windows 这样的非实时多任务操作系统,若不开启运行多个“任务”,也能够达到正常“运行”的目的,这也是在开发时,经常使用 Windows 的原因。

5.5.2 运动控制器板卡的设计分析

采用 DSP 为核心,结合 FPGA/CPLD 逻辑可编程器件的灵活性完成运动控制的硬件架构组成的运动控制器是目前伺服运动控制系统的主流。

首先,DSP 以其不适应频繁中断,擅长深度处理而著称,而运动控制器恰恰又是中断源少,控制算法复杂为特点。基于以上的理由,在伺服运动控制器中,DSP 已取代单片机。其次,FPGA/CPLD 逻辑可编程器件的灵活性使其在多 CPU 架构的联系中发挥了越来越重要的不可替代的作用。基于 PC 板卡的伺服运动控制系统的结构限制了运动控制器的物理尺寸,其中复杂的逻辑和时序电路的转换又需要多种 TTL 集成电路,器件越多,故障点就会越多,会给系统带来多种不稳定因素。FPGA/CPLD 逻辑可编程器件的应用,刚好为问题提供了解决方案。

现在的运动控制器板卡有两种输出模式:脉冲输出模式和模拟量输出模式。其中,脉冲输出模式的硬件电路比较简单,但输出有滞后效应;模拟量输出模式的硬件电路比较复杂,但输出无滞后效应。

大多数用于运动控制的 DSP 设计都具有许多共同的特征,常见的 DSP 算法可以归结为以下几类:

(1) 信号滤波;

(2) 信号变换(时域到频域的变换);

(3) 卷积(信号混合);

(4) 相关(信号比较)。

下面的数学表达式描述了数字滤波的过程:

$$y(i)=\sum_{k=0}^{k}b(k)x(i-k)-\sum_{m=1}^{m}a(m)y(i-m)$$

该过程需要大量的加法、减法和乘法运算。$b(k)$和 $a(m)$分别是存储器中的两个系数表,它们分别与过去的输入 $x(i-k)$和输出 $y(i-m)$样值表相乘。每当一个新的样值到来,将其放入输入样值表的上端,并删掉表中最老的样值。这个过程表明:DSP 器件应该设计成能够高效访问和管理其存储器,同时能够处理数据端口的输入和输出的数据流。图 5-22 所示是滤波器系数和样值。

另一个常见的 DSP 算法是快速傅里叶变换(FFT)。FFT 常用计算中用到的“点数”来表示,例如,128 或 256 点 FFT。点数越大,输出的 FFT 的频域分辨率越高,缺点是随着点数的增加,运算量也相应增加。下式可用来估算每个 FFT 输出结果所需的乘法次数:乘法次数对于 $N=1024$ 点 FFT,需要大约 5120 次乘法。实际上,1024 点 FFT 通常作为一个标

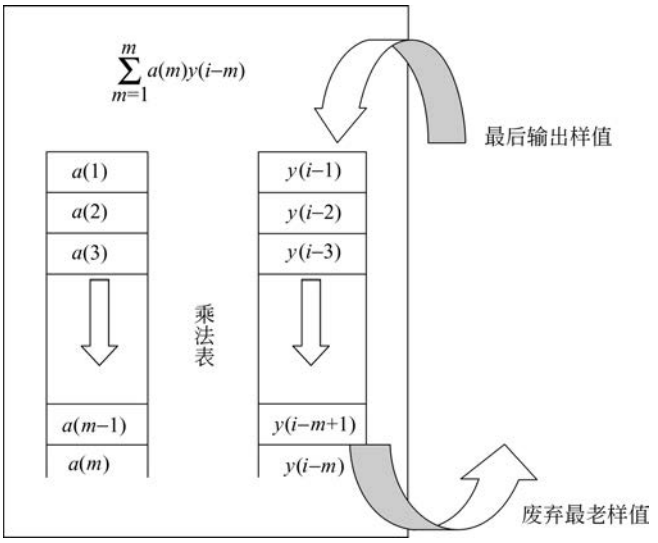


图 5-22　滤波器系数和样值

准来表示、比较不同 DSP 的性能。通用 FFT 可以用下面的关系式表示：

$$\text{乘法次数} = \frac{N}{2}\log_2 N$$

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{nK}$$

式中，$W_N^{nK} = \exp\left(-\frac{\mathrm{j}2\pi nK}{N}\right)$。

相关、卷积和功率谱都是与傅氏变换相关的函数。这些信号处理函数都能用 FFT 大致相同的关系式来实现。很明显这些算法都必须执行大量的乘法运算。大多数 DSP 器件具有乘累加运算(MAC)硬件，简化了这个过程。

5.5.3　驱动器、反馈元件的设计分析

交流伺服系统驱动器内部常采用 DSP 控制，内嵌频率解析功能，可检测出机械的共振点，具有共振抑制和控制功能，弥补机械的刚性不足，实现高速度定位，一般均采用 IGBT PWM 控制方式。

5.5.4　伺服电机的设计分析

伺服电机可匹配多种编码器，编码器模式与需求有关。如系统开始被控对象可复位时，常选用增量编码器，而当系统开始被控对象不可复位时，则常选用绝对编码器。

伺服电机的功率多为中容量(4000W 以下，三相供电电压)和小容量(750W 以下，单相供电电压)，特别是交流伺服电机防护等级高，环境适应性强。

5.6 基于 PC 的伺服运动控制系统举例

5.6.1 PMAC 开放式运动控制卡在数控系统中的应用

PMAC(Programmable Multi-Axes Controller)是美国 Delta Tau 公司于 20 世纪 90 年代推出的开放式多轴运动控制器,它提供运动控制、离散控制、内务处理、同主机的交互等数控的基本功能。

1. PMAC 简介

PMAC 内部使用了一片 Motorola DSP 56003 数字信号处理芯片,它的速度、分辨率、带宽等指标远优于一般的控制器。伺服控制包括 PID 加 Notch 和速度、加速度前馈控制,其伺服周期单轴可达 60μs,2 轴联动为 110μs。产品的种类可从 2～32 轴联动。甚至连接 MACRO 现场总线的高速环网,直接进行生产线的联动控制。与同类产品相比,PMAC 的特性给系统集成者和最终用户提供了更大的柔性。它允许同一控制软件在 3 种不同总线(PC-XT/AT,VME,STD)上运行,由此提供了多平台的支持特性,并且每轴可以分别配置成不同的伺服类型和多种反馈类型。

2. PMAC 的分类

PMAC 卡按控制电机的控制信号来分,有 1 型卡和 2 型卡。1 型卡输出±10V 模拟量,主要用速度方式控制伺服电机。2 型卡输出 PWM 数字量信号,可直接变为 PULSE+DIR 信号,来控制步进电机和位置控制方式的伺服电机。

PMAC 卡按控制轴数来分,有如下几种:

(1) 2 轴卡:MINI PMAC PCI;

(2) 4 轴卡:PMAC PCI Lite,PMAC2 PCI Lite,PMAC2A-PC/104 及 Clipper;

(3) 8 轴卡:PMAC-PCI,PMAC2-PCI 和 PMAC2A-PC/104 及 Clipper;

(4) 32 轴卡:TURBO PMAC 和 TURBO PMAC2。

PMAC 卡按通信总线形式分,有 ISA 总线、PCI 总线、PCI04 总线、网口和 VME 总线。PMAC 各种轴数的 1 型和 2 型卡,都有上述的计算机总线方式供选择,PMAC 除上述板卡形式外,还可以提供集成的系统级产品,有 UMAC、IMAC400、IMAC800、IMAC Flex、ADVANTAGE400、ADVANTAGE900 等,具体分类可以参考北京泰诺德公司网站。

3. 与各种产品的匹配

(1) 与不同伺服系统的连接:伺服接口有模拟式和数字式两种,能连接模拟、数字伺服驱动器,交、直流、直流无刷伺服电机伺服驱动器及步进电机驱动器。

(2) 与不同检测元件的连接:测速发电机、光电编码器、光栅、旋转变压器等。

(3) PLC 功能的实现:内装式软件化的 PLC,使用类似 Basic 的程序,可扩展到 2048 点 I/O。

(4) 界面功能的实现:按用户的需求定制。

(5) 与 IPC 的通信:PMAC 提供了 3 种通信手段——串行方式、并行方式和双口 RAM 方式。采用双口 RAM 方式可使 PMAC 与 IPC 进行高速通信,串行方式能使 PMAC 脱机运行。

(6) CNC 系统的配置：PMAC 以计算机标准插卡的形式与计算机系统共同构成 CNC 系统，它可以用 PC-XT&AT，VME，STD32 或者 PCI 总线形式与计算机相连。

4. 应用实例

随着 PC-NC 型数控技术的发展，具有模块化拓扑结构的集成电路在构筑数控系统硬件平台时得到广泛采用，使得系统硬件的可重构性和可重用性有了很大改观。而为了实现开放式数控系统平台，还必须建立一种通用的、组件化的软件模型，以提高这些软件的可重构能力与可重用性，真正达到开放性。COM 技术能很好地构筑组建化的软件模型，因此，可以利用 COM 技术来实现开放式数控系统平台。开放式数控系统软件平台如图 5-23 所示。

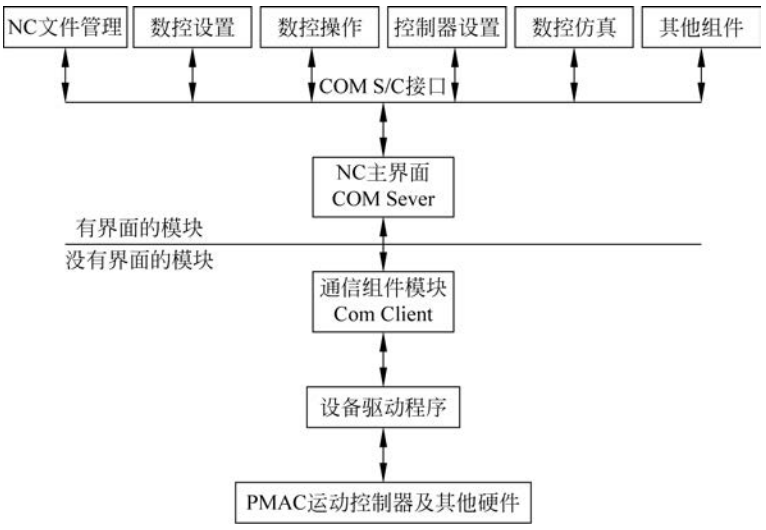


图 5-23　开放式数控系统软件平台

开放式数控平台的模块划分及模块间的关系如图 5-23 所示。NC 主界面程序运行时，运用插件技术将其他模块的 DLL 载入系统，系统被初始化，就能正常运行了。其中 COM Sever 是 NC 主界面程序，其他模块都是 COM Client，各个上层应用模块通过 COM 接口双向传输数据。用户可以根据自己的需求定制自己的程序，将不需要的模块删除，添加所需的模块，从而充分体现该数控平台的开放性。

笔者自行研制的微小零件加工机床其机床结构如图 5-24 所示，该机床具备铣、车、钻和镗等微细切削加工能力，机床共分为 X、Y、Z 和 C 共 4 轴，可实现 4 轴联动。同时机床在配上车削主轴后，X 轴与 C 轴可以组成 R-8 型车床，可以进行非球面车削。机床 X 和 Y 向导轨的运动分辨率为 1Fpm，C 轴旋转工作台转动分辨率为 2s。在铣削加工时 C 轴可以作为分度台使用，这样 C 轴既可在铣削时作为分度转台使用，又可在车削时实现球面和非球面加工。另外，利用伺服转台机构还可以充分利用其高精度、高频响的优点，并可以保持机床加工过程中的对称结构。由于微细切削加工不同于普通的切削加工，因此我们采用了基于 PMAC 的 IPC-NC 型开放式数控系统，针对微细加工的特点，对一些功能

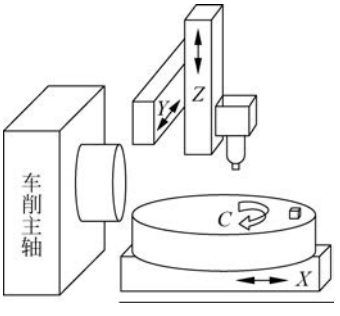


图 5-24　微型加工机床结构简图

模块的功能进行了改进。其硬件结构如图 5-25 所示,在通用的 IPC 插槽中插入 PMAC 卡,运用组件对象模型(COM)技术,建立了基于 PMAC 的开放式数控系统软件平台。微机上的操作系统为 Windows 98,采用 VC 作为开发语言,所开发的人机界面提供给用户一个操作接口,用户可以通过鼠标单击来完成相应的操作。该系统能实现的功能模块有 NC 文件管理、数控设置、数控操作、控制器设置、数控仿真、通信等。

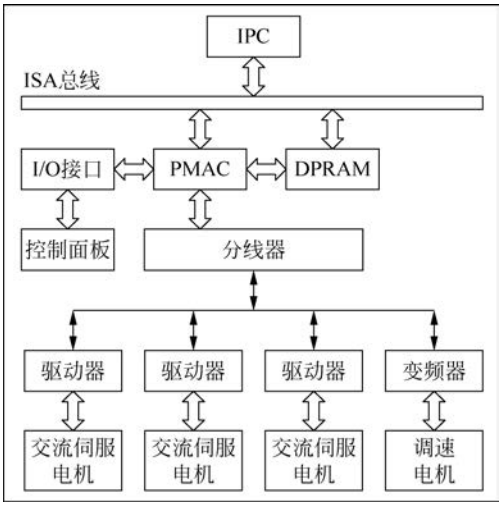


图 5-25 开放式数控系统原理框图

(1) NC 文件管理模块。管理数控加工文件输入、编辑、输出、修改数控加工文件,同时还具有语法检查功能。

(2) 数控设置模块。此模块主要完成系统参数设定,刀具参数的设置,同时还进行系统诊断。

(3) 数控操作模块。可以切换到手动操作、MDI 操作和自动操作,实时加工时各种数据的数字、图形综合显示窗口。显示的机床坐标包括绝对坐标显示,相对坐标显示与剩余距离显示;图形显示有加工轨迹图形与加工过程图像显示。

(4) 控制器设置模块。该模块的主要功能是对 PMAC 运动控制器的资源进行配置并且驱动运动控制器以供数控系统使用,并且将已经加载进了系统的 PMAC 运动控制器及其资源显示出来以供用户选用。开机时检测 PMAC 运动控制器设备是否存在,如果不存在,用户可以通过设置模块添加设备及设置相应的端口。对已经存在设备的系统,用户可以通过设置模块配置运动控制器设备。对不需要的设备可以通过设置模块将其删除。如果有多卡与主机相连通信,那么设置不同卡的端口以及编号,以便主机可以区分不同的卡,分别或同时与每一个卡通信。

(5) 数控仿真模块。通常在进行正式的 NC 加工之前,需要检验 NC 代码的正确性。对加工切削过程进行模拟仿真,校验其加工代码的正确性十分必要。为此,开发一个能调试和仿真数控指令程序的模块,该模块完全模拟零件的对刀和加工的整个过程,能在任意视图上真实逼真地显示刀具切削的动态过程,检测刀具、刀夹、主轴与工件、夹具、机床是否正常工作。

(6) 通信模块。主要完成 IPC 与运动控制器 PMAC 的通信任务,是上层界面与运动控制器的桥梁,其他组件模块都要通过它与运动控制器交换数据。在对以上关键技术研究的基础上,在自研的微小零件加工机床上成功地实现了三维微细零件的切削加工。

图 5-26 是应用 COM 技术开发的开放式数控系统的软面板,与数控机床的硬操作面板功能一样。它的接口是通用和开放的,可以与数控设置和 NC 文件管理等组件模块通信,通过简单的按钮操作就可以进行数控操作。

图 5-26　开放式数控系统的软面板

5.6.2　Trio 运动控制卡

Trio 从 1987 年创立之初起,一直致力于运动控制器的设计研发及应用研究工作。Trio 运动控制器的设计理念是:为客户提供满足各类现场应用要求的高品质的运动控制器。

1. Trio 运动控制器的特性说明

(1) 独立性:Trio 运动控制器从设计之初,即按照独立运行的理念为依托来设计控制器。每一款控制器均可以独立进行编程,无须外部计算机而独立脱机运行。

(2) 可靠性:目前在世界上,有超过 100 000 台各类电机由 Trio 运动控制器进行控制运转,没有发生一例安全事故。由 Trio 运动控制器组成的各个系统安全可靠地运行,涵盖了几乎工业自动化领域的各个行业。

(3) 安全性:Trio 运动控制器是一种嵌入式系统,其有自身独立的操作系统和运行环境,该环境与外界彻底隔离,从原理上讲就没有遭到外界计算机病毒攻击的可能性。

(4) 开放性：提供几乎所有的各类通信接口形式，可以与各类伺服驱动器、伺服电机连接，与各类计算机系统连接以及触摸屏连接等。

(5) 实时性：Trio 运动控制器特有的嵌入式开发系统，可以为客户提供最底层的开发编程环境，可以为客户提供最为实时的响应特性，提高生产效率。

(6) 高精度：在脉冲(步进)方式控制时，可以提供最高 2MHz 的脉冲输出频率，作为伺服(模拟量)方式控制时，可以最高接收 6MHz 的反馈输入脉冲。并且所有轴的每个伺服运算周期可到达 0.25ms。

2. 与台湾产的运动控制卡的功能及性能比较

1) 控制器的结构和原理

台湾地区产的各类运动控制卡的原理如图 5-27 所示。

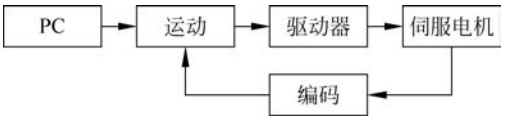


图 5-27 台湾地区产的各类运动控制卡的原理图

该图描述了目前市场上绝大多数运动控制卡的一个基本框图，运动控制卡作为一种接口卡插在计算机 PCI 插槽中，同时各个厂商为其运动控制卡提供专用的各类 PC 系统下的驱动和接口程序，运动控制卡作为一个计算机系统与实际伺服系统的一个接口单元，实时接收来自计算机的指令来进行运动过程的处理。用户在使用过程中，按照厂商提供的接口函数实时给运动控制卡发送各类指令。该类系统的实现需要计算机系统时刻参与运动过程的处理，简单的如限位处理，复杂的如多轴配合运动等。这类运动控制卡作为一种接口卡，预制了各种运动指令，但由于其没有自身的逻辑处理能力和编程能力，只能依靠上位计算机来做逻辑控制，而由于计算机目前多采用 Windows 系统，不能实现实时操作而且还需处理其他各种运算统计工作，此外运动控制还不同于简单的逻辑控制，因此造成各个子系统间耦合性过强，整体系统可靠性差，而且难于维护，例如急停处理，由于其要由 PC 进行处理，因此一旦 PC 系统出现故障，则急停信号就会失效，造成系统安全性能下降。

2) Trio 运动控制器体系结构

Trio 运动控制器的体系结构如图 5-28 所示。

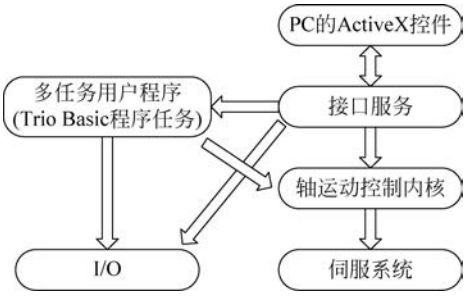


图 5-28 Trio 运动控制器体系结构

Trio 运动控制器采用独立式的设计方式和理念，本身控制器就是一个可以执行多任务程序的嵌入式系统。用户可以根据自身需要用 Trio Basic 语言进行程序开发，整个系统可以脱离任何外界 PC 系统独立运行。同时借助 PCI 总线或各类串口通信方式与 PC 可进行实时通信。此外控制器本身就具有 I/O 端口，加扩展后可直接扩展到 272 点，通过自身内部程序编写即可实现最为实时可靠的处理。由于运动控制并不是简单的指令传送问题，其还涉及更为复杂的逻辑及调度问题，所以可以用 Trio Basic 编写出各类程序模块，将运动问题全部交给控制器自己来做，PC 通过设置各个状态位实现对运动过程的调度。与其他控制卡相比，Trio 运动控制器更为灵活，即可以按照原有控制卡的模式进行工作外，还可以进行自身编程，将原来复杂的程序结构进行分解简化，可以大大降低程序的复杂程度，提高系统整体的可靠性。

3）控制方式

控制方式上，Trio 运动控制器可以根据客户需要将各个轴进行独立配置，如前面两个轴为脉冲方式，后面 6 个轴为伺服方式等。而其他大多数轴卡只能有一种模式，而且每张轴卡所支持轴的数量也仅有 4 个而已。Trio 公司目前单个 CPU 支持的轴数最多可达 24 轴，并且已经开发出 64 轴的运动控制器。PCI 方式的运动控制器，目前标准产品为 8 轴，根据需要已经开发出 14 轴的 PCI 方式的运动控制器。此外 Trio 运动控制器形式多样，既有插在 PC 中的 PCI 接口卡，又有独立式的运动控制器，所有的运动控制器中的软件都互相通用，用户可以随时改变自己系统的形式，而不需要改变系统结构。

4）运动程序的编程

Trio 运动控制器通过 Motion Perfect2 软件采用 Trio Basic 语法进行嵌入式程序开发，程序简单易懂。而且为 PC 提供 ActiveX 控件，上位机可以与控制器进行直接的数据交互和运动指令传送。用户可以根据需要将运动相关的所有过程直接编写到控制卡内，与 PC 只是简单交互对应数据和状态而已，整个系统结构清晰，稳定可靠。而且可大大简化上位机程序开发的复杂程度，节约开发周期。此外由于运动控制过程独立于其他计算机运算过程，系统效率较高，并且在上位机系统出现故障时，不会影响运动过程，提高了系统的安全性（例如紧急停止信号直接由控制器实现，不需要计算机的干预）。

其他大多数运动控制卡没有嵌入式开发接口，只提供 DLL 动态链接库，用户必须通过 PC 程序去单独地触发一个个运动指令，从形式上讲该类控制卡只是一种程序接口卡而已，用户必须将所有的运动过程在 PC 中进行处理，这样做的效率低下而且程序结构过于复杂，不利于系统的稳定运行和维护。

5）I/O 控制

Trio 运动控制器本身就带有 I/O 端口，以 PCI208 为例，其本体即带有 10 个输出、20 个输入接口，而且还可以直接连接扩展模块。用户可以将所有的 I/O 处理直接放在运动控制卡上实现，不但包括限位处理而且还有其他功能性 I/O 都可以在控制器中直接通过编程实现。

其他大多数控制卡没有 I/O 能力，需要在计算机中插接另外的 I/O 处理卡件，所有的逻辑（有时包括限位信号）都需要通过 PC 控制，这样做的效率过低，而且系统不够稳定。

3. 应用

1）Trio 公司的 PCI208 运动控制卡

英国翠欧运动技术公司（Trio Motion Technology Ltd）成立于 1987 年，是一家致力于运动控制领域内技术研发及产品生产的公司。其数字运动控制器产品分为独立型控制器和

基于 PC 的 PCI 线式控制器两大类。

PCI208 是翠欧(Trio)公司基于 PC 的 PCI 线控制的控制卡的典型代表,该运动控制卡采用了独立的 120MHz 的 DSP 微处理器技术,可以控制 8 个轴的伺服或步进电机,并提供 1、2 个 CAN 点线通道和独立的 I/O 扩展。通过增加一个扩展连括器,PCI208 还能额外提供 10V 的电压输出。同时在实际工作中,它还能够独立于计算机运行。

PCI208 设计为主-子板结构,主板即为 PCI208,提供了运动控制卡的基本功能,当主板不能满足复杂应用的需要时,可以通过使用与 PCI208 配套的 P182,P183,P184,P185 等扩展组件即子板,增强 PCI208 的功能。PCI208 通过名为 Breakout Board 的子板(P181)与其他子板建立连接。

软件编程方面 PCI208 使用也非常方便。由于翠欧(Trio)公司已提供了高度封装的 Active X 控件,使用者可以很轻易地在 PC 上用 VB、VC、Delphi 等高级语言进行二次开发。通过控件提供的连接、运动、变量、输入/输出以及过程控制等命令完成与 PCI208 的通信,实现要求的运动控制。

PCI208 本身也是可编程的,使用的编程语言为 Trio BASIC,这是一种类 BASIC 的面向过程语言。在 Windows 系统下,Trio 公司还提供了一个集成开发环境 Motion Perfect 2,该软件不仅可以对运动控制器编程、离线开发,还能执行在线控制和监测等功能。

通过对控制器编程,PCI208 最多能同时提供多达 8 个任务的多任务运行。值得强调的是,使用者应该尽可能地充分利用控制卡的多任务特性,提高资源的利用率。

2) PCI208 在数控领域的应用

在数控系统中,PC 与运动控制卡通信最多、最频繁的是传递 G 代码数据。G 代码中包含着大量的信息。下面以一种简化的 G 代码模型为例,介绍如何利用 PCI208 的多任务特性完成上位机与运动控制卡的数据交换。

(1) 上位机与运动控制卡框架结构。

在运动控制卡内存在 3 个进程:控制卡主进程、解释程序进程以及实时状态进程。

上位机主程序发送加工命令和参数到运动控制卡主进程,然后由控制卡主进程开启解释程序进程和实时状态进程。实时状态进程向上位机主程序返回当前加工位置坐标等信息,上位机主程序则根据这些信息,启动传输数据例程向运动控制卡的 TABLE 数据表传送数据。解释程序进程则不断地从 TABLE 数据表取出数据,控制电机运动,完成数控加工,图 5-29 为程序结构模块。

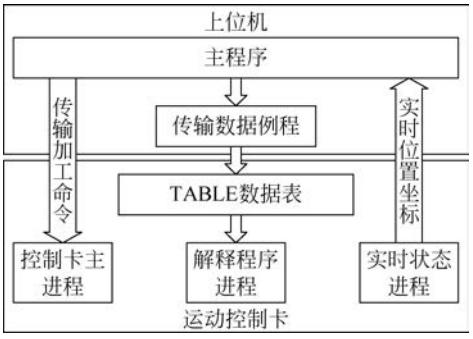


图 5-29 程序结构模块

(2) TABLE 数据表与解释程序。

PCI208 的 TABLE 数据表如表 5-4 所示，根据任务需要，将 TABLE 表示以 6 个存储空间为基础分段。在每段中，第 1 个存储位置存放插补方式 C 值，并将第 2,3,4,5,6 位置存放数据 X、Y、Z、I、J(根据插补方式的不同，有些位置的值为 0)。在控制卡内部有一个专门运行解释程序(见图 5-30)的线程，不断地从 TABLE 中读取插补方式和坐标点信息，以控制插补运动。

表 5-4　TABLE 数据表

Address	Value	Address	Value
1	G=1	8	X=112.374
2	X=103.902	9	Y=178.828
3	Y=209.159	10	0
4	Z=50.0	11	I=-22.543
5	0	12	J=-81.031
6	0	13	…
7	G=2		

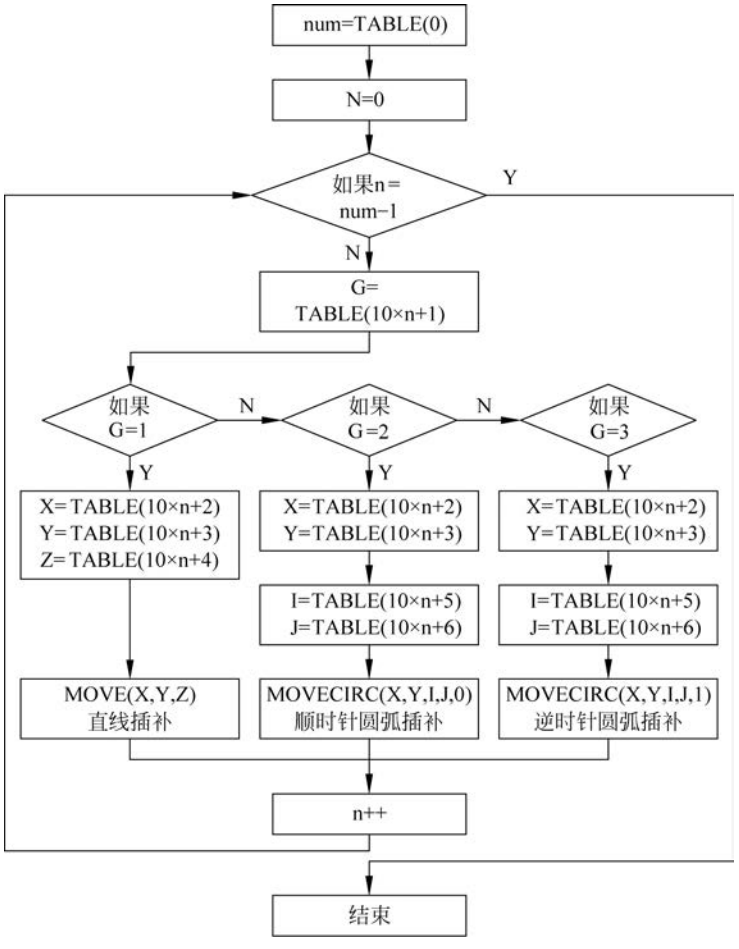


图 5-30　解释程序流程图

当然,随着问题复杂性的不同,还可以在每段中添加其他的数据成员,以控制更复杂的运动。

5.6.3 PCIMC-3A 及 PCIMC-3B 控制卡

1. PCIMC-3A 控制卡

PCIMC-3A 型计算机运动控制卡是维宏科技公司专门为 NC STUDIO™ 运动控制系统(该系统是上海维宏科技有限公司自主开发、自有版权的运动控制系统)设计的配套板卡。该卡插在 PC PCI 槽内,通过它实现机床运动控制。

1) 控制卡的结构

PCIMC-3A 型计算机运动控制卡外形如图 5-31 所示,该卡尺寸为 160mm×120mm。

LED 为一发光二极管用做状态指示。系统上电启动时,LED 闪烁发光。启动后持续发光。当 NC STUDIO™ 运动控制软件启动后,LED 熄灭。DB15 为 15 芯(针)插座,通过电缆与机床通信。底端为插脚,插在 PC PCI 槽内。

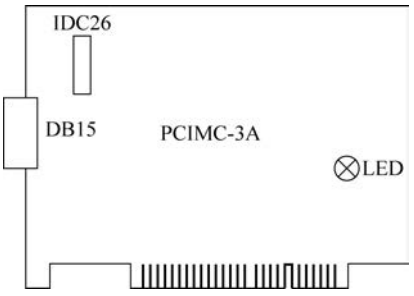


图 5-31 PCIMC-3A 型计算机运动控制卡外形图

2) 控制卡的安装

关闭主机电源,打开机箱盖,将运动控制卡插入任何一个空的 PCI 槽内,安装时,用手轻按运动控制卡两侧,确保运动控制卡牢固插入 PCI 槽,然后,旋紧固定螺钉,盖好机箱盖。

3) 控制卡与驱动系统的连接

NC STUDIO 的机械运动控制信号通过插在计算机 PCI 扩展槽上的运动控制卡实现 NC STUDIO 软件系统与安装在机床电气箱的进给电机驱动系统的通信。PCIMC 运动控制卡与电机驱动系统连接之前,应先将机床与电气箱安装就位,用专用的 15 芯电缆将运动控制卡上的 15 芯插座与电气箱上的 15 芯插座连接,这样 NC STUDIO 运动控制卡与电机驱动系统的连接就完成了。

PCIMC-3A 控制卡与机床连接线插头定义如表 5-5 和表 5-6 所示。

表 5-5 PCIMC-3A 控制卡引脚定义(DB15RA/M,针)

引脚	功能	信号方向	描 述
1	对刀	输入	对刀信号为低电平时,表示刀已接触对刀块;对刀信号悬空或高电平时,表示无效
2	ZP	输出	低电平时,驱动器内的输入光耦导通,否则截止
3	YP	输出	低电平时,驱动器内的输入光耦导通,否则截止
4	XP	输出	低电平时,驱动器内的输入光耦导通,否则截止
5	主轴频率	输出	可驱动 F/V 模块的输入光耦主轴转速为控制主轴转速的频率信号,当频率越高时,转速越高,反之转速越低
6	Z 限位	输入	低电平表示限位到

续表

引脚	功能	信号方向	描　　述
7	Y 限位	输入	低电平表示限位到
8	X 限位	输入	低电平表示限位到
9	XD	输出	低电平时,驱动器内的输入光耦导通,否则截止
10	YD	输出	低电平时,驱动器内的输入光耦导通,否则截止
11	ZD	输出	低电平时,驱动器内的输入光耦导通,否则截止
12	冷却控制	输出	输出 TTL 信号,高电平有效并启动外部设备,低电平时无效并关闭外部设备
13	主轴控制	输出	输出 TTL 信号,高电平有效并启动外部设备,低电平时无效并关闭外部设备
14	GND		
15	＋5V	输出	输出电流容量不小于 500mA

表 5-6　PCIMC-3A 控制卡引脚定义(DB25RA/F,孔,通过 IDC26 插座引出)

PCIMC-3A, PCB J2(DB25)引脚	功 能 描 述
J2-1	GND
J2-2	＋5V
J2-3	×1 挡
J2-4	Y－　Y 负方向移动
J2-5	×10 挡
J2-6	Z－　Z 负方向移动
J2-7	×100 挡
J2-8	Z 轴精定位
NC	
J2-10	LED1 ×1 指示灯
J2-14	GND
J2-15	＋5V
J2-16	Z＋　Z 正方向移动
J2-17	X＋　X 正方向移动
J2-18	Y＋　Y 正方向移动
J2-19	X－　X 负方向移动
J2-20	X 轴精定位
J3-21	Y 轴精定位
J2-22	LED2　×10 指示灯
J2-23	LED3　×100 指示灯

注：NC 表示未定义,J2 插口用于扩展手持盒等设备。

4) 电气接线示意图

(1) 接法一。

为了控制主轴和冷却电机,这里采用固态继电器作为控制元件。主轴的固态继电器输出控制主轴变频器的启停,冷却的固态继电器输出控制冷却电机的启停,如图 5-32 所示。

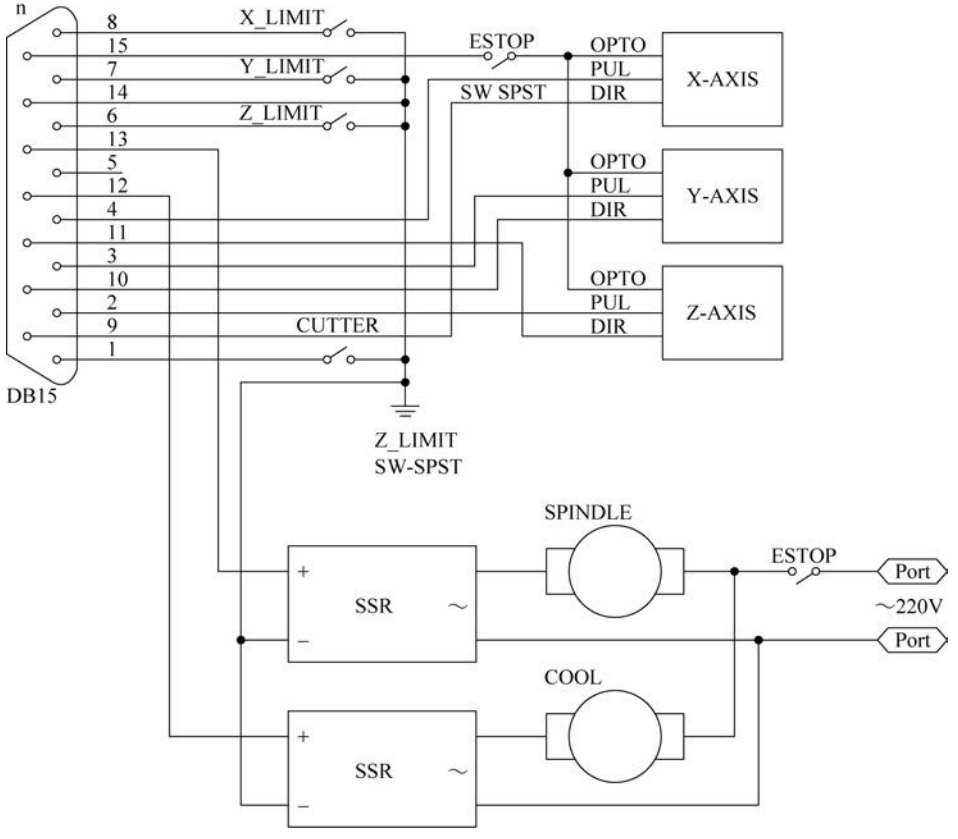


图 5-32 接法一

(2) 接法二。

当需要主轴速度控制时,可选用 MPG-3AX 板,该板可用于连接手摇脉冲发生器,同时还有 0～10V 模拟量输出,可实现主轴转速的调速控制,如图 5-33 所示。

2. PCIMC-3B 控制卡

PCIMC-3B 型计算机运动控制卡是维宏科技公司专门为 NC STUDIO™ 运动控制系统(该系统是上海维宏科技有限公司自主开发、自有版权的运动控制系统)设计的配套板卡。该卡插在 PC 的 PCI 槽内,通过它实现机床运动控制。

1) 控制卡的结构

PCIMC-3B 型计算机运动控制卡外形如图 5-34 所示,该卡尺寸为 160mm×120mm。

LED 为一发光二极管用做状态指示。系统上电启动时,LED 闪烁发光。启动后持续发光。当 NC STUDIO™ 运动控制软件启动后,LED 熄灭。

DB15 为 15 芯(针)插座,通过电缆与机床通信。

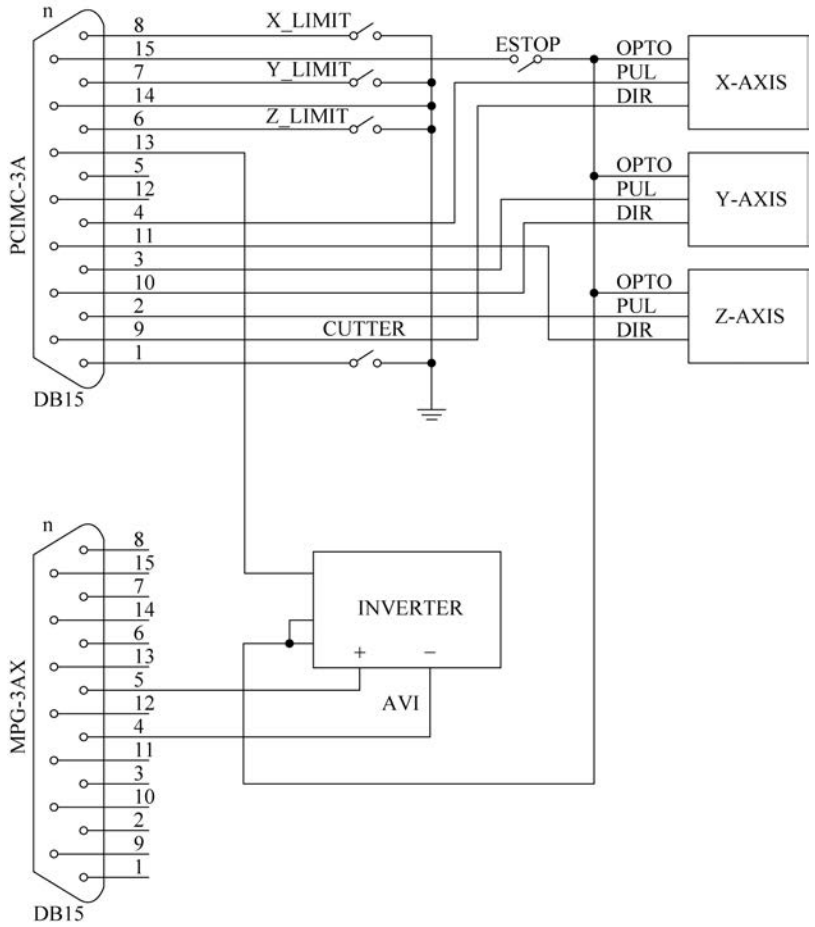


图 5-33　接法二

底端为插脚，插在 PC PCI 槽内。

2）控制卡的安装

关闭主机电源，打开机箱盖，将运动控制卡插入任何一个空的 PCI 槽内，安装时，用手轻按运动控制卡两侧，确保运动控制卡牢固插入 PCI 槽，然后，旋紧固定螺钉，盖好机箱盖。

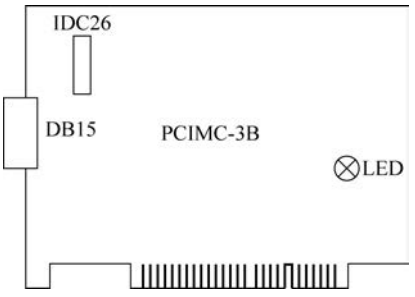


图 5-34　PCIMC-3B 型计算机运动控制卡外形图

3）控制卡与驱动系统的连接

NC STUDIO 的机械运动控制信号通过插在计算机 PCI 扩展槽上的运动控制卡实现 NC STUDIO 软件系统与安装在机床电气箱的进给电机驱动系统的通信。PCIMC 运动控制卡与电机驱动系统连接之前，应先将机床与电气箱安装就位，用专用的 15 芯电缆将运动控制卡上的 15 芯插座与电气箱上的 15 芯插座连接，这样 NC STUDIO 运动控制卡与电机驱动系统的连接就完成了。

PCIMC-3B 控制卡与机床连接线插头定义如表 5-7 和表 5-8 所示。

表 5-7 PCIMC-3B 控制卡引脚插头定义(DB15RA/M,针)

引脚	功能	信号方向	描 述
1	对刀	输入	对刀信号为低电平时,表示刀已接触对刀块;对刀信号悬空或高电平时,表示无效
2	ZP	输出	低电平时,驱动器内的输入光耦导通,否则截止
3	YP	输出	低电平时,驱动器内的输入光耦导通,否则截止
4	XP	输出	低电平时,驱动器内的输入光耦导通,否则截止
5	主轴高速	输出	OC 门输出
6	Z 零点	输入	低电平表示零点到
7	Y 零点	输入	低电平表示零点到
8	X 零点	输入	低电平表示零点到
9	XD	输出	低电平时,驱动器内的输入光耦导通,否则截止
10	YD	输出	低电平时,驱动器内的输入光耦导通,否则截止
11	ZD	输出	低电平时,驱动器内的输入光耦导通,否则截止
12	主轴低速	输出	OC 门输出
13	主轴中速	输出	OC 门输出
14	GND		
15	+5V	输出	输出电流容量不小于 500mA

表 5-8 PCIMC-3B 引脚定义(DB25RA/F,孔,通过 IDC26 插座引出)

PCIMC-3B, PCB J2(DB25)引脚	功 能 描 述
J2-1	GND
J2-2	+5V
J2-3	×1 挡
J2-4	Y− Y 负方向移动
J2-5	×10 挡
J2-6	Z− Z 负方向移动
J2-7	×100 挡
J2-8	Z 轴精定位
NC	
J2-10	LED1 ×1 指示灯
J2-14	GND
J2-15	+5V
J2-16	Z+ Z 正方向移动
J2-17	X+ X 正方向移动
J2-18	Y+ Y 正方向移动
J2-19	X− X 负方向移动
J2-20	X 轴精定位
J3-21	Y 轴精定位
J2-22	LED2 ×10 指示灯
J2-23	LED3 ×100 指示灯

注:NC 表示未定义,J2 插口用于扩展手持盒等设备。

4）电气接线示意图

为了控制主轴电机的转速，控制卡输出 3 个 OC 门信号，分别可控制主轴高速、中速、低速旋转。这里要求变频器带分挡控制。

如果选择 DZJ-3 转接板，则电气接线更简单。3 个轴 6 个方向的限位都用常闭开关，并将它们串联后，一端连到 XW1，另一端连到 XW2；限位释放按钮（常开，按下时接通）并接到 XW1 和 XW2；紧停开关（常闭）两端分别连到 ES1 和 ES2。SPL、SPM、SPH 用于控制变频器实现主轴转速的分档控制。3 个轴的零点信号和对刀信号直接连到该转接板，如果使用普通行程开关，则这些开关都必须有一端与 GND 相连；如果使用光电开关、霍尔开关或接近开关，转接板可为其提供 5V 电源，如图 5-35 所示。

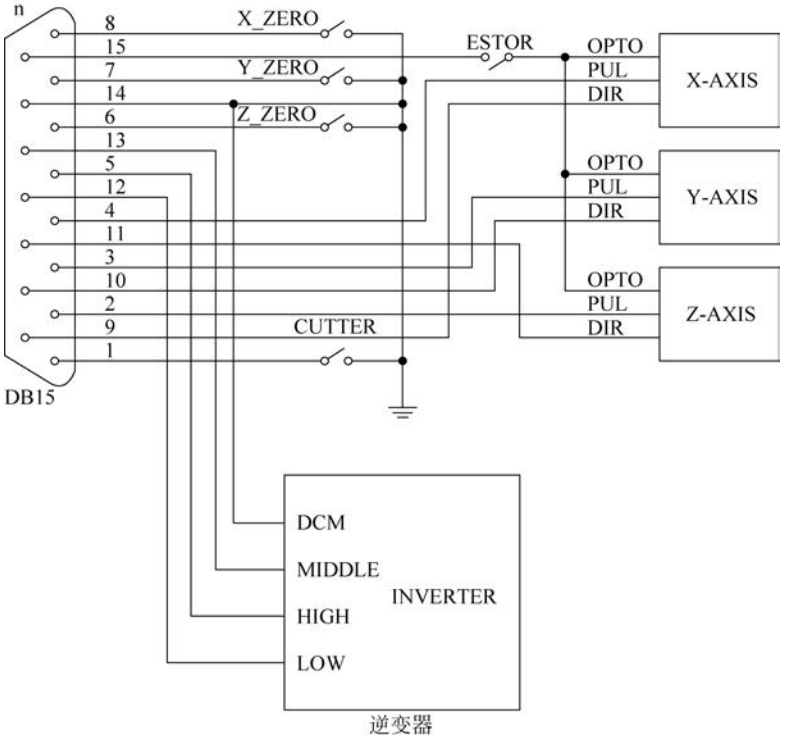


图 5-35　电气接线示意图

习题

5-1　伺服系统对控制系统的要求是什么？简述基于 PC 与 PLC 运动控制器的区别。

5-2　试画出基于 PC 的运动控制系统结构框图。

5-3　PCI 总线与 ISA 总线在功能优缺点方面的比较分析。

5-4　描述实时多任务操作系统（iRMX），并分析其组成，画出系统结构示意图。