基于工控机和 DSP 的关节机器人控制器研究

摘 要

工业机器人随着数字信号处理技术和计算机技术的发展得到越来越广泛的应用。在六自由度机器人群体中，关节型机器人以工作范围大、动作灵活、结构紧凑、能抓取靠近机座的物体等特点备受设计者和使用者的青睐。本课题设计的六自由度关节型机器人既可以用于实际生产，也可以用于教学和科研。

首先，本文介绍了工业机器人技术和机器人控制器技术的应用和发展，给出了本课题研究的机器人对象。随后本文对机器人的机构可行方案进行了充分论证，在此基础上运用 D-H 方法进行了机器人的运动学分析并采用关节空间法详细规划了机器人的运动轨迹。

其后，本文探讨了机器人运动学正逆解求解方法，详细地讨论了其中的 PID 算法。通过对控制方案进行比较和对芯片进行选型，确定了基于 PCI 总线的 IPC（工控机）和 DSP 两级机器人伺服控制方案。DSP 运动控制卡以 TI 公司的浮点型信号处理器 TMS320LF4207 作为控制核心，用 CPLD 设计伺服电机的码盘信号接收电路，用数模转换器 DAC7725 提供伺服电机的转速给定值，构成了伺服电机的位置闭环系统。上位机实现机器人参数给定、命令控制、人机接口、上层路径规划等功能。PCI 总线选用 PLX 公司的从模式芯片 PCI9052 作为桥接芯片。编写了针对 PCI9052 的 WDM 驱动程序，实现了工控机和 DSP 运动控制卡的正常通讯。

在本文的最后，设计了关节调试程序，至此成功地搭建了 6 关节机器人控制系统的平台，为以后进一步的的科研和应用打下了的基础。

关键词： 机器人，运动控制，DSP，TMS320LF4207，PCI9052

RESEARCH ON THE CONTROLLER OF JOINT ROBOT BASED ON INDUSTRIAL COMPUTER AND DSP

ABSTRACT

With the development of DSP and computer technology,robot has got great application. In the group of six degrees of freedom robots, the articulated robot is cared by designer and user for its broad work range, flexible movement,compact structure, catching the object near the machine plinth. The sixdegrees of freedom robot designed in the article can be used not only in the practical production but also in the teaching and science research.

Firstly,the thesis introduces industrial robots technology and the development and application of robot controller,in the following the study object is given clearly.After demonstrating the feasibility of robot mechanics, this thesis analyses the robot's motion with D-H methodology, and planes the robot motion path detailedly with the method of joint space.

Then the paper discusses method of robot's kinematics,inverse kinematics and robot's control arithmetics mostly about PID control. Based on the compare of control theory and chip selection,we select the two-level robot's control system based on IPC and DSP as our system.Servo Control Card uses TMS320LF4207 as control core, uses CPLD to design reversible counter, and uses DAC7725 as DA output chip, they compose the close-loop control system of servo motor.Industrial PC fulfills robot's settings of parameter, controls of command,operational interface,complicated control arithmetic and so on. Subordinate mode chip PCI9052 is used for PCI bus connecting chip.WDM driver is programmed for PCI9052 and Protocol designed for communication between IPC and DSP Control Card.

In the last part,the debugging Procedures are designed and up to now we succeed in makeing up of robot's control system for prepare of advanced study and application.

KEYWORD: robot，motion control，DSP，TMS320LF4207，PCI9052

原创性声明及关于学位论文使用授权的声明

原 创 性 声 明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的科研成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律责任由本人承担。

论文作者签名：宋海波　　日　期：2009　年5月

关于学位论文使用授权的声明

本人完全了解陕西科技大学有关保留、使用学位论文的规定，同意学校保留或向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅；本人授权陕西科技大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或其他复制手段保存论文和汇编本学位论文。

（保密论文在解密后应遵守此规定）

论文作者签名：宋海波　导师签名：　日　期：2009年5月

1 绪论

1.1 引言

机器人技术是综合了计算机控制论机构学信息和传感技术人工智能仿生学等多学科而形成的高新技术，是当代研究十分活跃应用日益广泛的领域，机器人应用情况是一个国家工业自动化水平的重要标志，工业机器人在现代工业上起着极其重要的作用，对我国的工业现代化的发展也有着无比重要的作用，当前机器人技术飞速发展学习和研究机器人工作的原理和运用也成为各部门教学的重要问题。

机器人技术是建立在机械、微电子、控制、人工智能技术等基础上的一门机电一体化的高新技术。机器人诞生于1954年，美国的George Devol首先把远程控制器的杆结构与数控铣床的伺服轴结合起来，研制出了第一台通用机械手。这种机械手可以通过让其沿一系列点运动，将运动位置以数字形式存贮起来，动作执行时，使伺服系统驱动机械手各关节轴来再现这些位置，从而让机械手完成一些简单的工作。正是由于这个机械手具有了编程示教再现功能，因此许多人把它作为现代工业机器人产生的标志。在随后的生产应用中，为了适应各种不同用途需要，相继出现了直角坐标、关节坐标、极坐标等许多种不同结构的机器人。与第一代示教再现型机器人不同，第二代机器人是具有感觉功能的适应控制机器人。这种机器人带有传感器，能感知环境和对象的情况，以控制自身动作的变化。在机器人视觉、触觉研究的同时，机器人其他感觉功能的开发也已经开始，对机器人的接近觉、听觉、滑觉以及会话等功能都进行了研究，并取得了一定的成果。这种有感知功能的机器人已被广泛应用于工业生产中的汽车制造、电子、机械等行业从事焊接、油漆、装配、包装、零件加工、搬运等专业性工作。目前，这类工业机器人占总数的70%以上，全球正在工作的工业机器人共有74万，我国有 3000 台左右智能机器人被人们称为第三代机器人，它是能利用感觉和识别功能做决策行动的机器人。从七十年代后期开始，人们对智能机器人的规划生成系统（问题解决系统，路径搜索等），环境的理解系统（感觉智能，包括视觉，触觉等），知识获得与利用系统（知识工程和专家系统等），人—机接口系统，运动系统等问题展开了更加广泛、深入地研究。智能机器人的研究是一个艰巨而又广泛的问题，随着研究的不断深入，相关科学的飞速发展（如第五代智能计算机），具有智能的机器人在不久的将来一定会出现。

我国的机器人研究始于二十世纪70年代。经过近20年努力，特别是经过“七五”攻关、“863”计划，取得了一批重要成果，掌握了机器人控制系统硬件设计、软件设计、机器人语言等技术。综上所述，我国机器人发展已跨过了起步阶段，走上了进步和发展的道路。今后的任务是把机器人技术推广到更多的工业自动化生产领域和其他更广泛的应用领域，大力开展跨过区域交流合作，与国际接轨，早日跻身于世界先进行列。

1.2 机器人控制器的应用和发展

在各种作业中，机器人的运动主要表现为沿着预定的路径移动，并保持预定的姿态，在运动中或在给定的某些位置上完成作业规定的操作，这些都是在机器人控制器的控制下完成。机器人控制器相当于机器人的“大脑”，是机器人的核心和灵魂，设计的成功与否最终影响到机器人的优劣。机器人控制器由一组硬件和软件组成，是机器人系统不可分割的一部分，与机器人系统的工作原理、机器人的结构和功能、机器人的精度都有密切的关系。早期由于计算机技术比较落后，很多先进的控制方法无法在机器人上完全实现。九十年代以来，随着计算机技术的成熟并且引入机器人领域，机器人迅速地发展起来。机器人控制器已经和计算机技术紧密结合在一起。以计算机为核心的机器人控制器的发展经历了下面几个阶段：

1）基于通用微型处理器。这种控制器由8088、8031等为核心部件，加上存储器、编码器信号处理电路及D/A转换电路等，其位置环控制算法由事先编好的程序固化在存储器中。这种方案采用元器件较多，可靠性低，体积比较大，而且控制参数不易更改，软硬件设计工作量较大。

2）基于专用微控制器。采用专门生产用于伺服控制的芯片，如LM628、HCPL1100、PCL833等，完成速度曲线规划、PID伺服控制算法、编码器信号的处理等多种功能。电机位置、速度、加速度、PID参数等需要经常更改的参数存储在芯片的RAM区，但由于运算速度的限制，复杂的控制算法和功能很难在其中实现。近年来，专用控制芯片的运算速度越来越快，功能越来越强大，这种方案目前使用还较广泛。

3）基于通用数字信号处理器（DSP）。近年来，DSP在伺服控制系统中的运用越来越广泛，为设计高精度机器人开辟了道路。DSP的高速运算能力可以实现各种复杂的控制算法，而它本身独特的硬件结构可实现快速的位置捕捉等功能。在伺服控制领域中，比较常用的DSP有TI公司C24xx和C28x系列AD公司的ADSP21xx系列和AT&T公司DSP16x系列等。典型的DSP多轴运动控制卡是DeltaTau公司开发的开放结构运动控制器PMAC，它以Motorala的DSP56K系列DSP为核心支持多种总线插槽方式，一块卡可以同时控制8-32轴，可以并联运行。随着DSP价格的下降，采用DSP的机器人控制器应用日趋广泛，DSP的高速运算能力和计算机的强大资源相结合，可以开发出高性能的机器人，本课题就是采用IPC和DSP的两级控制方案设计机器人控制器。

1.3 开放式机器人控制系统的方案

1.3.1 开放式控制系统的特征

开放式控制系统是随着计算机控制技术的发展而产生的，其具有以下几点特征

1）模块化：开放式系统由一系列的功能模块组成。

2）可移植性：开放式系统的可移植性是指在不同的控制器或硬件平台上运行相同的系统组件的能力。

3）可互操作性：系统组件之间可以相互协作共同工作，总线式设计是实现可互操作性的关键。

4）可扩展性：开放式系统的功能可以方便的进行扩展。

5）可互换性：指开放式系统可以选择其它同类产品的能力。即可实现成本高低产品的组合。

6）可派生性：可以对某些组件进行升级，从而使整个系统的功能增加。

1.3.2 开放式机器人控制系统的优点

采用模块化技术，开发机器人系统的过程中可以使用经过测试、性能良好的子系统模块。功能模块的复用可以提高系统的质量和安全性能，保证控制器能满足要求，不会产生意想不到的错误，使机器人系统安全性得到可靠的保证。开放的控制系统能使系统中的硬件和软件结构很容易集成。更易实现平台、操作系统和用户接口的标准化。开放式机器人控制器的设计可以由用户更换或修改，用户可以根据需要进行机器人控制器的改型。开放式机器人系统的应用范围更广。

1.3.3 基于 PC 的开放式控制系统的实现模式

开放式结构的控制器的发展趋势是以通用PC为技术，采用面向对象的模块化的设计方法来构造系统，目前基于PC的实现模式主要有以下几种：

1）单PC的控制模式以PC为核心，配制实时的操作系统。存在实时性、标准统一性及系统稳定性的问题。

2）PC+PC的控制模式 PC+PC的控制模式采用两极CPU的结构，一级CPU完成系统管理工作。二级CPU完成全部关节位置的数字控制。这类的控制模式的两个CPU总线之间没有任何联系，是一个松耦合关系。

3）PC+DSP运动控制器的模式PC+DSP运动控制器的模式采用了以DSP芯片的多轴运动控制技术。DSP运动控制器的特点在于它的集成性、兼容性和高速性。PC机处理非实时的部分，实时的运动控制由插入PC的采用DSP芯片来承担。这种控制器灵活性好、功能稳定，是一种较为实用的控制模式。

1.4 本课题的研究对象和内容

本课题的研究对象是一个6关节机器人手臂，由底座主体、大臂、小臂和手腕等几个部分组成，本课题完成的主要内容有如下几个方面：

1）机器人本体结构的选择。

2）机器人运动学分析和相关控制算法。

3）机器人硬件选型及配置。

4）设计工控机和DSP运动控制卡的通讯协议。

5）完成了机器人单关节和双关节的软硬件调试。

本课题设计中使用的集成开发环境和软件资源有：Protel99SE，TICC4.0，MaxplusII，VisualC++6.0，Windows2000DDK，DriverStudio2.7。工控机操作系统采用 Microsoft公司的Windows2000。

2 机器人结构设计

2.1 机器人的工作要求

该机器人的特点是工作范围大，动作灵活，通用性强，结构较紧凑，能抓取靠近机座的物体。协作单位根据其用途和特点提出如下技术参数：

自由度数目：6个

坐标形式：垂直关节型

额定负荷质量（含末端执行器）：25kg

最大活动半径：1.50m

本体重量：≤180kg

各关节回转范围和最大工作转速见表2-1。

表 2-1 各关节回转范围和最大工作转速

Table2-1 Each joint gyration range and maximal run rotate speed

	回转范围（°）	最大工作转速		
		r/min	Rad/s	°/s
腰部回转	±150	28	2.64	150
大臂回转	-45～+120	54	3.66	210
小臂同转	±55	64	6.45	300
腕部摆动	±105	56	5.38	300
腕部俯仰	±90	54	5.45	300
腕部回转	±180	78	7.32	420

2.2 机器人结构方案的对比分析及选择

根据本课题的要求，参考国内外工业机器人的典型结构，初步对各个回转关节的结构单独分析。

2.2.1 腰部回转关节

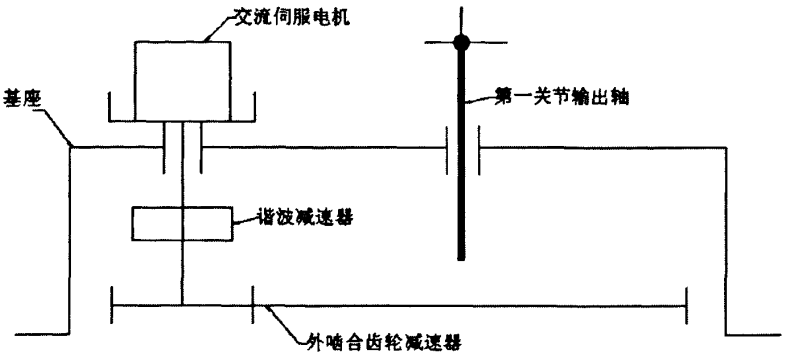


图2-1 腰部回转示意图

Fig2-1 Waist gyration sketch map no.1

方案：如图2-1所示，电机安装在底座上面，其输出轴先经谐波减速器减速，再经一对齿轮减速后，由第一关节输出轴带动整个腰部在基座上回转。

2.2.2 大臂和小臂回转示意图：

大臂和小臂回转都是通过谐波减速器减速后直接带动来实现的。力学方面更合理，布局美观，适合于较大负荷的机器人。

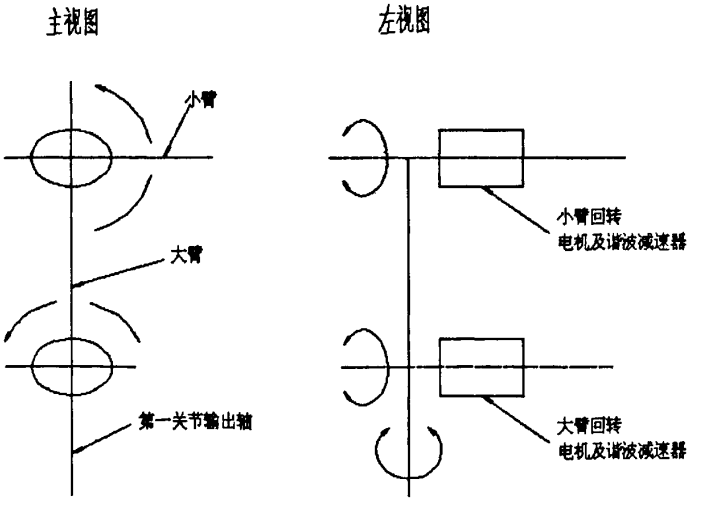


图2-2 大臂和小臂回转示意图

Fig2-2 Big arm and little arm gyration

2.2.3 腕部活动关节

如图2-3所示，步进电机先经过同步带传动减速，再通过锥齿轮传动换向后带动腕部摆动。对于腕部俯仰和腕部回转关节，均采用通用的结构，步进电机通过同步带传动减速后输出。

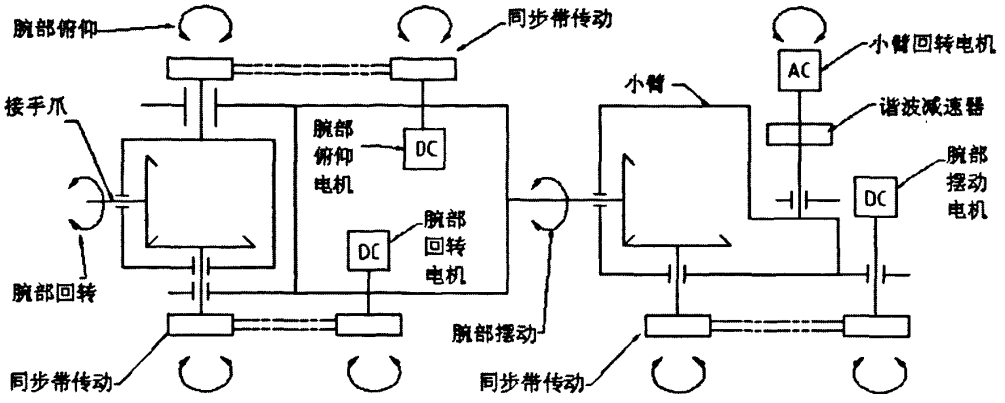


图2-3 腕部及小臂回转示意图

Fig2-3 Wrist and little arm gyration sketch map

2.3 机器人驱动方案的对比分析及选择：

1）步进电机：步进电机可直接实现数字控制，控制结构简单，控制性能好，而且成本低廉；通常不需要反馈就能对位置和速度进行控制；位置误差不会积累；步进电机具有自锁能力（变磁阻式）和保持转矩（永磁式）的能力，有利于控制系统的定位。但步进电机基本上不具有过载能力，功率偏大者，体积较大，并且其空间分辨率较低；功率较小者，只适于传动功率不大的关节或小型机器人。

2）交流伺服电机：交流伺服电机结构较简单，体积较小，运行可靠，使用维修方便，价格比直流伺服电机便宜，但高于步进电机。随着可关断晶闸管GTO，大功率晶闸管GTR和场效应管MOSFET等电子器件、脉冲调宽技术和计算机控制技术的发展，交流伺服电机在调速性能方面可以与直流电机媲美。

3）直流伺服电机：直流伺服电机具有良好的调速特性，较大的启动力矩，相对功率大及快速响应等特点，并且控制技术成熟。但其结构复杂，体积偏大，成本较高，而且需要外围转换电路与微机配合实现数字控制。若使用直流伺服电机，还要考虑电刷放电对实际工作的影响。

4）液压伺服马达：液压伺服马达具有较大的功率体积比，运动比较平稳，定位精度较高，负载能力也比较大，能够抓住重负载而不产生滑动，从体积、重量及要求的驱动功率这几项关键技术考虑，不失为一个合适的选择方案。但是，其费用较高，而且其液

压系统经常出现漏油现象，维护不方便。综合分析后，决定采用混合式步进电机和交流伺服电机混合驱动。腰部、大臂和小臂要求动态特性好、传动功率较大，采用交流伺服电机驱动；腕部所需传动功率较小，采用混合式步进电机驱动。

2.4 本章小结

本章根据该机器人的工作要求，从机器人本体结构和分析入手，初步确定了该机器人的整体结构方案，依据分析计算结果绘制出该机器人的工程图，如图2-4所示。

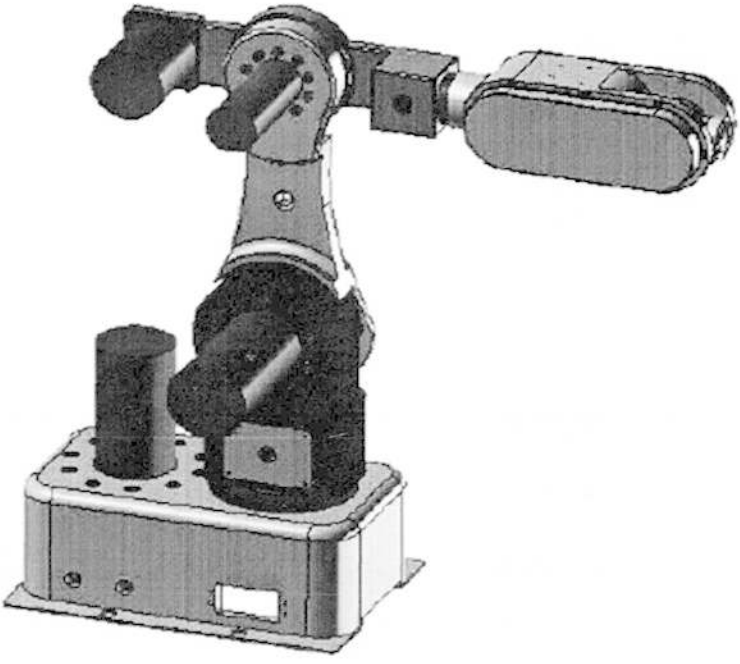


图2-4 装配图
Fig2-4 assembly drawing

3 机器人运动学分析及运动控制技术研究

3.1 关节机器人运动学

3.1.1 机器人运动方程

机器人机构是由一系列关节连接起来的连杆机构组成。把构件坐标系嵌入机器人的每一个连杆机构中，可以方便的描述一个连杆与下一个连杆之间的关系。齐次变换是描述这些坐标系之间的相对位置和方向的一种通用方法，把齐次变换记为A矩阵。一个A矩阵仅仅是描述连杆构件坐标系之间相对平移和旋转的齐次坐标变换。

设An描述第n个连杆相对于第n－1连杆的位姿，对于六自由度机器人，则第六个连杆相对于基座的位姿可用下式表示：

$$T_6 = A_1 A_2 A_3 A_4 A_5 A_6 \tag{3-1}$$

3.1.2 构件坐标系的确定

为了描述连杆之间的数学关系，Denavit和Hertenberg提出了为关节链中的杆件建立主附体坐标系的矩阵方法，即D-H法。连杆串联型机器人是由一系列连接在一起的连杆构成的。需要用两个参数来描述一个连杆，即公共法线距离an和垂直于an所在平面内两轴的夹角αn；需要另外两个参数来表示相邻两连杆的关系，即两连杆的相对位置dn和两连杆法线的夹角θn，如下图3-1所示。除第一个和最后一个连杆外，每个连杆两端的轴线各有一条法线，分别为前、后相邻连杆的公共法线。这两条法线之间的距离为dn。我们称an为连杆长度，αn为连杆扭角，dn为两连杆距离，θn为两连杆夹角。机器人机械手上坐标系的配置取决于机械手连杆连接的类型。其连接方式有两种——转动关节和棱柱联轴节。对于转动关节，θn为关节变量。连杆n的坐标系原点位于关节n和关节n+1的公共法线与关节n+1轴线的交点上。如果两相邻连杆的轴线相交于一点，那么原点就在这一交点上。如果两轴线相互平行，那么就选择原点使对下一连杆的距离dn为零。连杆n的z轴与关节n+1的轴线在一直线上，而x轴则在连杆n和n+1的公共法线上，其方向从n指向n+1，见图3-1。

当两关节轴线相交时，x轴的方向与两矢量的交积zn×zn平行或反向平行。x轴的方向总是沿公共法线从转轴n指向n+1。当两轴xn和xn平行且同向时，第n个转动关节的θn为零，y轴的方向按右手定则确定。

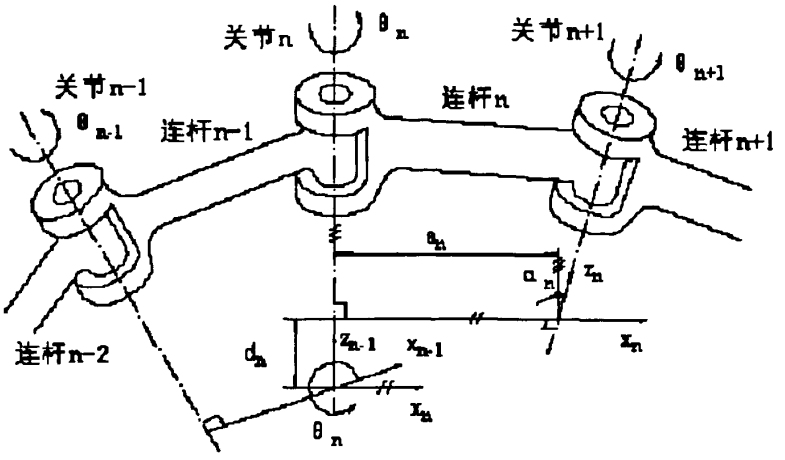


图3-1 机器人转动关节结构参数

Fig3-1 Structure parameter of robot rotation joint

3.1.3 构件参数的确定

根据D—H构件坐标系表示法，构件本身的结构参数 a_n、α_n 和相对位置参数 d_n、θ_n 可由以下的方法确定：

1）θ_n 为绕 Z_{n-1} 轴（按右手定则）由 X_{n-1} 轴到 X_n 轴的关节角；

2）d_n 为从第n-1坐标系的原点到 Z_{n-1} 轴和 X_n 轴的交点沿 Z_{n-1} 轴的距离；

3）α_n 为从 Z_{n-1} 和 X_n 的交点到第n坐标系原点沿 X_n 轴的偏移距离；

4）a_n 为绕 X_n 轴（按右手定则）由 Z_{n-1} 轴到 Z_n 轴的偏转角。

3.1.4 变换矩阵的建立

这种关系可由表示连杆n对连杆n－1相对位置的四个齐次变换来描述，称为An矩阵，此关系式为：

$$A_n = Rot(z,\theta_n)Trans(0,0,d_n)Trans(a_n,0,0)Rot(x,\alpha_n) \tag{3-2}$$

展开上式得（旋转关节）

$$A_n = \begin{bmatrix} \cos\theta_n & -\sin\theta_n\cos\alpha_n & \sin\theta_n\sin\alpha_n & a_n\cos\theta_n \\ \sin\theta_n & \cos\theta_n\cos\alpha_n & -\cos\theta_n\sin\alpha_n & a_n\sin\theta_n \\ 0 & \sin\alpha_n & \cos\alpha_n & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3-3}$$

当机械手各连杆的坐标系被规定以后，就能够列出各连杆的常量参数。这样，A矩阵就成为关节变量 θ 的函数（对于旋转关节）或变量d的函数（对于棱柱联轴节）。一旦求得这些数据之后，就能够确定式（3-1）中n个变换矩阵Ai（i=1,2……n）的值（机器人

的关节数为n）。

3.1.5 建立正运动学数学模型

对机器人操作机进行运动分析的目的是为了对机器人进行运动控制，同时也是对机器人进行动力分析的基础。运动分析分为正、逆两类问题，正向运动学主要解决机器人运动方程的建立及手部位姿的求解问题，是分析轨迹规划的基础。首先根据该机器人实体参数，采用D-H法建立其齐次坐标系，参数见表（3-1）及其变换矩阵。

表 3-1 连杆参数

Table3-1 Link parameters

杆件	转角 θ_i	偏距 d_i mm	扭角 α_i	杆长 mm
1	$\theta_1(0)$	d_1=0	$\alpha_1=0$	L1=0
2	$\theta_2(90)$	d_2=0	$\alpha_2=0$	L2=210
3	$\theta_3(0)$	d_3=0	$\alpha_3=90$	L3=0
4	$\theta_4(0)$	d_4=340	$\alpha_4=-9$	14=0
5	$\theta_5(0)$	d_5=0	$\alpha_5=90$	L5=0
6	$\theta_6(0)$	d_6=0	$\alpha_6=0$	L6=0

$$A_1=\begin{bmatrix} C_1 & 0 & S_1 & 0 \\ S_1 & 0 & -C_1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},\qquad A_2=\begin{bmatrix} C_2 & -S_2 & 0 & l_2C_2 \\ S_2 & C_2 & 0 & l_2C_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$A_3=\begin{bmatrix} C_3 & 0 & S_3 & 0 \\ S_3 & 0 & -C_3 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},\quad A_4=\begin{bmatrix} C_4 & 0 & -S_4 & 0 \\ S_4 & 0 & C_4 & 0 \\ 0 & -1 & 0 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$A_5=\begin{bmatrix} C_5 & 0 & S_5 & 0 \\ S_5 & 0 & -C_5 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},\quad A_6=\begin{bmatrix} C_6 & -S_6 & 0 & 0 \\ S_6 & C_6 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_6=A_1A_2A_3A_4A_5A_6=\begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3-4}$$

式中 n，o，a，p——机器人手部位姿的四个列向量。(3-4)为正运动学方程。

3.2 机器人的轨迹

3.2.1 机器人轨迹规划方法

机器人轨迹规划是在机械手运动学和动力学基础上，讨论在关节空间和笛卡儿空间（直角坐标空间）中机器人运动的轨迹规划和轨迹生成方法，根据作业任务的要求，计算出预期的运动轨迹。轨迹规划既可在关节空间也可在直角坐标空间中进行。

3.2.2 关节空间法

关节空间法是将关节变量表示为时间的函数，此方法不必在直角坐标系中描述两个路径点之间的路径形状，计算简单，容易。再者，由于关节空间和直角坐标系之间并不是连续的对应关系，因而不会发生机构的奇异性问题。在关节空间中进行轨迹规划，用户需要根据作业要求给定机器人手臂在各个结点的位姿，通过逆运算反解出关节变量，对关节变量进行插值，得到关节轨迹。但是关节轨迹要满足一组约束条件，如在各个结点（起始点、提起点、下放点和终止点）上的位姿、速度和加速度的要求，使关节位置、速度和加速度在整个时间间隔内连续等。同时，还必须给出两个路径点之间的运动时间。轨迹规划的流程如图3-2所示，

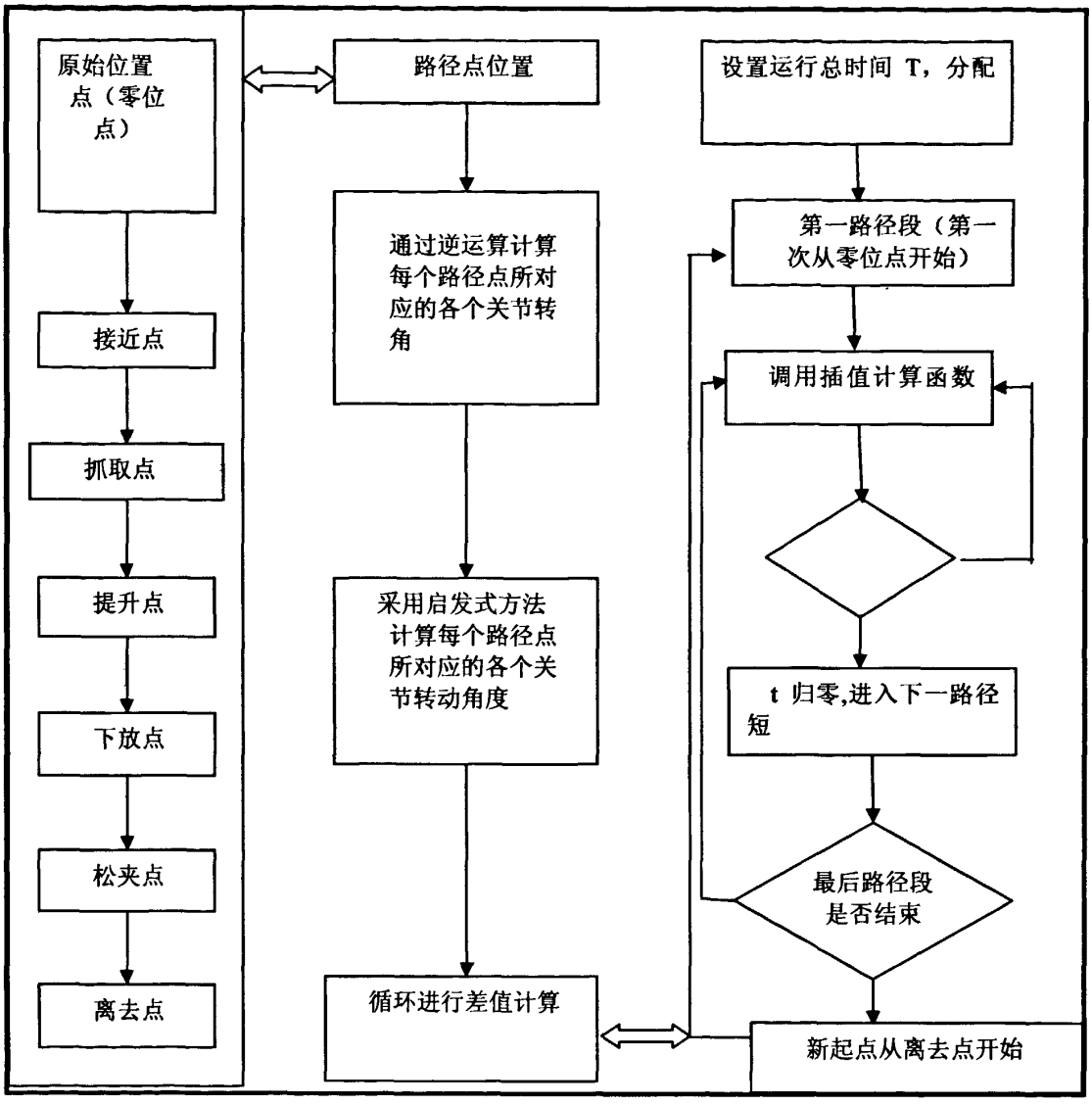


图3-2轨迹规划流程图

Fig3-2Flow chart of trajectory planning

大致步骤如下：

第一步：由于已知 ${}^{S}_{T}\mathrm{T}={}^{S}_{G}\mathrm{T}\,{}^{G}_{T}\mathrm{T}={}^{S}_{G}\mathrm{T}$ （运动中{T}应和{G}重合），利用 ${}^{B}_{W}T={}^{B}_{S}T\,{}^{S}_{T}T\,{}^{W}_{T}T^{-1}$ 对所有结点 $P_0, P_1, P_2, P_3, P_4\ P_5$ 依次求解相应的手臂变换矩阵：${}^{0}_{6}T_0$，${}^{0}_{6}T_1$ ${}^{0}_{6}T_2$，${}^{0}_{6}T_3$，${}^{0}_{6}T_4$，${}^{0}_{6}T_5$ 由 ${}^{0}_{6}T_i={}^{B}_{W}T={}^{0}_{6}T$，可得出腕部矢量n,o,a,p的各个分量的具体表达式。

$$q_i=\{\theta_{1i},\theta_{2i},\theta_{3i},\theta_{4i},\theta_{5i},\theta_{6i}\} \tag{3-5}$$

第二步：将各结点的手臂变换矩阵，用逆运动学求解相应的关节矢量

q_0,q_1,q_2,q_3,q_4,q_5 其中，第一脚标数表示关节序号，第二脚标数i=0,1,2,3,4,5表示路径点序号。

第三步：采用启发式方法计算各个路径点的速度对第j关节而言 θ_{j0} 为起始点、θ_{jn} 为终止点、$\theta_{j(i-1)},\theta_{ji},\theta_{i(i+1)}$ 为路径点，则相邻两结点连线的斜率可用下式表示：

$$K_i = \frac{\theta_{ji} - \theta_{j(i-1)}}{t_i - t_{i-1}} \tag{3-6}$$

第四步：对某一关节的各个结点，进行三次多项式插值，$\theta(t)$ 在时刻 t_0=0的值是起始关节角度 θ_0，在终端时刻 t_f 的值是终止关节角度 θ_f。设三次多项式：

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \tag{3-7}$$

则运动轨迹上的关节速度和加速度：

$$\dot{\theta}(t) = a_1 + 2a_2 t + 3a_3 t^2 \tag{3-8}$$

$$\ddot{\theta}(t) = 2a_2 + 6a_3 t \tag{3-9}$$

由位置约束条件：

$$\left.\begin{aligned}\theta(0) &= \theta_0 \\ \theta(t_f) &= \theta_f\end{aligned}\right\} \tag{3-10}$$

由速度约束条件：

$$\left.\begin{aligned}\dot{\theta}(0) &= \dot{\theta}_0 \\ \dot{\theta}(t_f) &= \theta_f\end{aligned}\right\} \tag{3-11}$$

得出，插值函数可表示为：$\theta(t) = \theta_0 + \dot{\theta}_0 t +$

$$\left[\frac{3}{t_f^2}(\theta_f - \theta_0) - \frac{2}{t_f}\dot{\theta}_0 - \frac{1}{t_f}\dot{\theta}_f\right]t^2 + \left[-\frac{2}{t_f^3}(\theta_f - \theta_0) + \frac{1}{t_f^2}(\dot{\theta}_0 + \dot{\theta}_f)\right]t^3 \tag{3-12}$$

第五步：关节空间轨迹的实时生成轨迹生成器只需随 t 的变化不断按插值函数 $\theta(t)$ 计算 θ，当达到路径段的终点时，调用新路径段的的三次样条系数，重新赋 t 为零，继续生成轨迹。

3.3 运动控制技术概述

由于运动系统能够实现对运动轨迹、运动速度、定位精度以及重复定位精度的精确控制要求，在各类控制工程中有着广泛应用前景，因此运动控制系统目前已成为控制科学应用领域中一个很有意义的研究方向。在运动控制系统中，按机械运动的轨迹分类，可分为点位、直线、轮廓控制等。现代数控机床及机器人绝大多数具有两个坐标或两个坐标以上联动的功能，如 6 轴（或自由度、维）控制的机械手，其运动可以给定在空间

的任意方向。

3.4 交流伺服系统的控制技术

如图 3-3 所示为交流伺服系统的控制原理框图。全数字式交流伺服系统包含了位置环、速度环和电流环。其中电流环一般是在伺服驱动器内部实现的，而速度环与位置环则是伺服运动控制器要实现的主要功能。

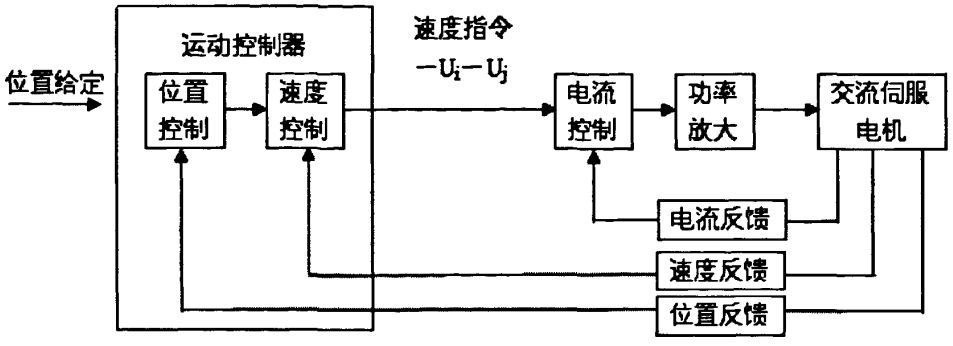


图 3-3 交流伺服系统的控制原理图

Fig3-3 Principle of AC servo control system

这种伺服系统可在驱动器中由参数设置为位置、速度和转矩三种控制模式，现分述如下。

3.4.1 位置控制

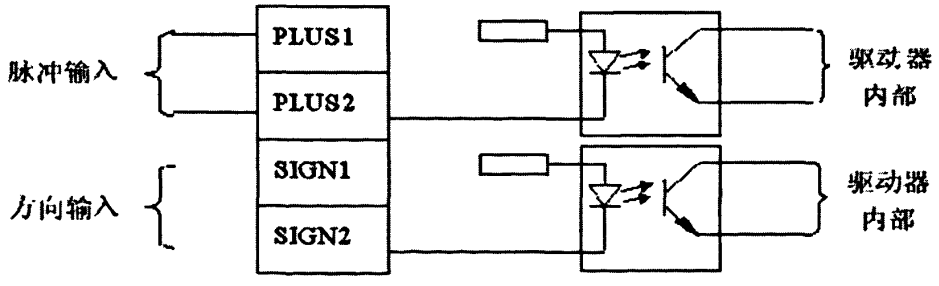


图 3-4 交流伺服系统位置控制接口电路

Fig3-4 Interface circuit of AC servo position control

当伺服系统处于位置控制时，控制系统给伺服驱动器的信号是脉冲和方向信号。这一点和步进电机的控制方式类似。其接口电路如图 3-4 所示。指令脉冲的输入方式可分为以下三种：

1）正交脉冲指令

如图 3-5 所示，频率相同但相位相差 90° 的 A、B 两相脉冲分别从 PLUS1、PLUS2 和 SIGN1、SIGN2 送入伺服驱动器。A、B 两相脉冲的频率控制电机转速；脉冲数控制电机的角位移。电机每转所需指令脉冲数可由驱动器内部参数设置。两相脉冲的超前或滞后关系控制电机旋转方向。亦即，若 A 相脉冲超前 B 相脉冲 90° ，则电机旋转方向为 CCW，若 B 相脉冲超前 A 相脉冲 90° ，则电机旋转方向为 CW。一般增量式旋转编码器的输出信号就是这种频率相同但相位相差 90° 的两路脉冲信号，可用这种信号直接控制电机的转速和方向，而不需增加方向识别电路。

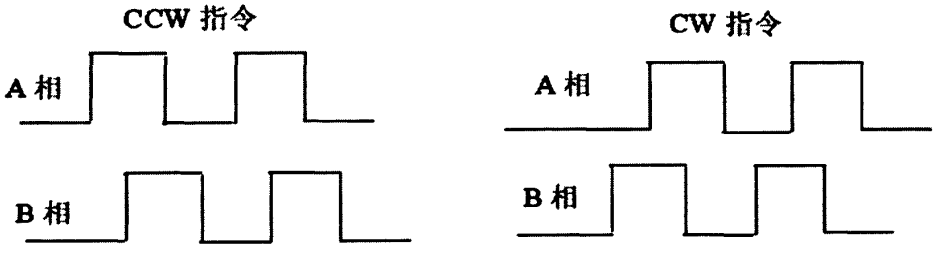


图 3-5 正交脉冲指令波形图

Fig3-5 Quadrature pulse instruction oscillogram

2）CW/CCW 脉冲指令

即单脉冲工作方式。脉冲信号通过 PLUS＋、PLUS－进入驱动器，则电机按 CW 方向旋转。若通过 SIGN＋、SIGN－进入驱动器，则电机按 CCW 方向旋转。脉冲频率控制电机的旋转速度，脉冲数控制电机的角位移。

3）脉冲+方向指令

脉冲信号从 PLUS1、PLUS2 进入驱动器，脉冲频率控制电机转速；脉冲数控制电机的角位移。方向信号从 SIGN1、SIGN2 进入驱动器，高低电平控制电机的转向。

3.4.2 速度控制

在速度控制模式中，上位控制系统通过 SPD、GND 引脚给伺服驱动器输入一个－10V～＋10V 的模拟电压，即可控制电机实现从负向最大转速到正向最大转速之间的速度变化。如图 3-6 所示，电机转速 n 和指令输入电压 v 之间是呈线性关系的.

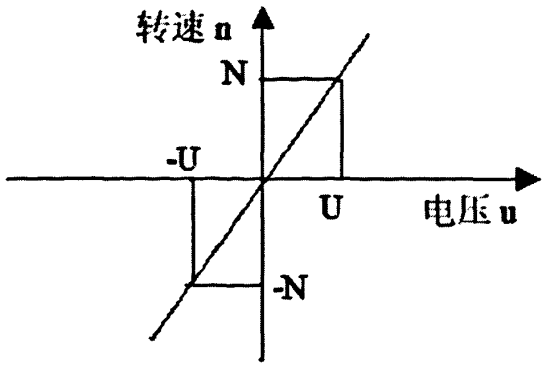


图 3-6 交流伺服电机转速同电压的关系

Fig3-6 Relation of rotation rate and voltage of AC servo

速度指令除了可以由外部模拟电压来输入外，还可以在驱动器内部用四个参数设置四种内部速度。通过驱动器的两个开关输入信号的四种状态组合选择其中一种。驱动器可由内部参数对外部速度指令进行零漂调整。本控制卡就是采用速度控制方式对伺服驱动器进行控制的。

3.4.3 转矩控制

通过外加－10V～＋10V 的电压，即可控制电机的转矩。和速度控制相似，电机的额定转矩和输入电压之间呈线性关系，直线斜率可用驱动器内部参数设置。伺服电机工作在转矩控制模式时，应限制其最大转速，以免驱动器产生过速报警。

3.5 控制算法的实现

3.5.1 数字 PID 算法

按偏差的比例、积分和微分进行控制的调节器，简称为 PID 调节器，是连续系统中技术成熟且应用广泛的一种调节器。它的结构简单，不一定需要系统的确切数学校型，参数易于调整，在长期应用中已积累了丰富经验。将它移植到计算机控制系统，通过软件予以实现，对丁大多数控制对象都能获得满意的控制效果，所以人们常常采用数字 PID 调节算法．并根据经验和实验，在线整定参数，具有很强的灵活性和适应性：随着微机控制技术的发展，PID 控制算法已能在微机上简单地实现—由于软件设计的灵活性，数字 PID 算法可以很容易得到修正而比模拟 PID 调节器的性能更完善。因此，数字 PID 控制算法是电机微机控制中常用的一种基本控制算法。

a 模拟 PID 调节器

在连续控制系统中，模拟 PID 调节器是一种线性调节器。模拟 PID 调节器的框图如图 3-7 所示，图中 w 是设定值，y 为系统输出，$e=w-y$ 构成控制偏差，为 PID 控制器的输入，u 为 PID 控制器的输出，也是被控对象的输入。模拟 PID 调分器的控制规律为：

$$u = K_P\left(e + \frac{1}{T_i}\int_0^T edt + T_D\frac{de}{dt}\right) + u_0 \tag{3-13}$$

式中 K_p—比例系数；T_i—积分常数；T_D—微分常数，模拟 PID 控制器中比例调节器的作用是对于偏差作出瞬间快速反应。偏差一旦产生。调节器立即产生控制作用使控制量向着减小偏差的方向变化，控制作用的强、弱取决于比例系数 K。见式(3-13)表明，只有当偏差存在时，第一项才有控制量输出，因此对于大部分控制对象，如直流电机的电枢电压凋速，要加上适当的和转速及机械负载有关的控制常量 u0，否则，单用比例调节器会产生静态误差(静差)。

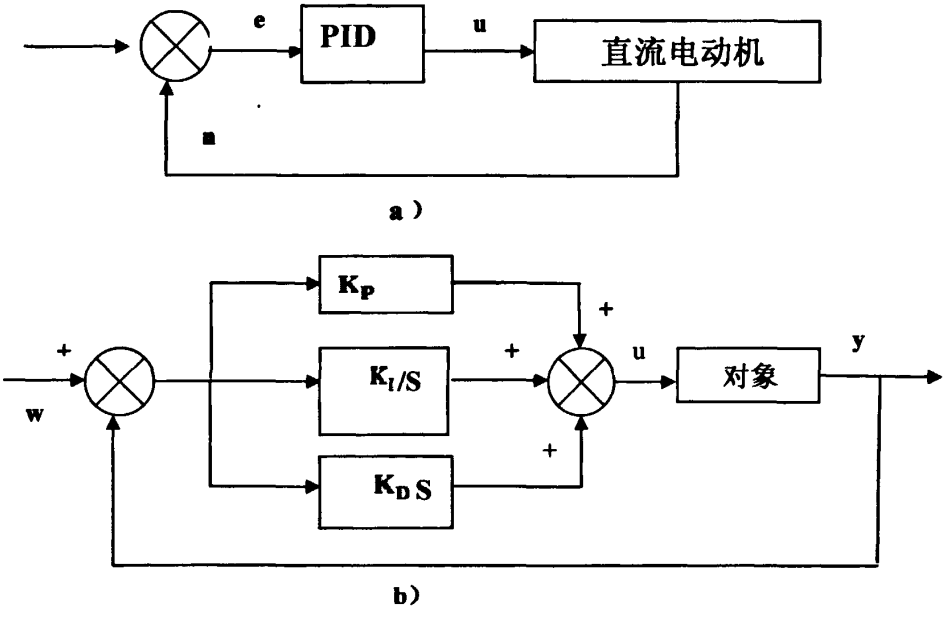


图 3-7 模拟 PID 控制

Fig3-7 Simulation PID control

积分调节器的作用是把偏差累积的结果，作为它的输出，在调节过程中，只要偏差存在，积分器的输出就会不断增大，直至偏差 e=0，输出 u 才可能维持某一常量，使系统在设定值 w 不变的条件下趋于稳态。因此，即使不加上适当的控制常量 u_0，有了它就能消除系统输出的静差。积分调节作用虽然可以消除静差，但会降低系统的响应速度增加系统输出的超调。式(3-13)的第二项表明，积分常数 T_I 越大，积分的累积作用越弱，反之则积分作用越强。T_I 必须根据控制的具体要求来选定。增大 T_I 将减慢消除静差过程，但可减少超调，提高稳定性。实际的控制系统除了希望消除静差。还要求加快调节过程，有必要在偏差出现的瞬间偏差变化的瞬间，不但对偏差量作出即时反应(即比例调节作用)，而且根据偏差的变化或者偏差的变化趋向预先给出适当的控制量。为了达到这

一目的，可以在 PI 调节器的基础上加微分调节器，这样就得到式(3-13)所示的 PID 调节器。微分调节器的作用是阻止偏差的变化，偏差变化越快，微分调节器的输出也越大。因此微分作用的加入将有助于减小超调，克服振荡，使系统趋于稳定。它加快了系统跟踪的速度。适当选样微分常数 T^D 的大小，对恰当地实现上述微分作用是至关重要的。

传统的模拟 PID 调节器是用硬件(如电子元件、气动装置、液压器件等)实现它的调节规律的。自从微机进入控制领域以来，只要对式(3-13)所示模拟 PID 控制规律，作适当的近似变换，以适合微机运算，就可以用软件来实现 PID 调节。这样就提供了更大的灵活性，从而在微机控制中得到广泛的应用。下面我们首先讨论如何将 PID 控制规律离散化。

b PID 控制规律的离散化方法

计算机控制是一种采样控制，它只能根据采样时刻的偏差值计算控制量，进行离散控制，而不能像模拟控制器那样能连续输出控制量，实现连续控制；因此式(3-13)中的积分和微分项在微机中不能直接准确计算，只能用数值计算的方法逼近。如果 T 为采样周期，用离散采样时刻点 iT 表示连续时间 t，以和式代替积分，以增量代替微分，可作如下近似变换：

$$t=iT\ (i=0,1,2,\ldots\ldots) \tag{3-14}$$

$$\int_0^t e(t)dt = T\sum_{j=0}^{i} e_j \tag{3-15}$$

$$\frac{de(t)}{dt} = \frac{e_i - e_{i-1}}{T} \tag{3-16}$$

将式(3-16)带入式(3-13)，可得离散（数值）型控制的近似计算公式：

$$u_i = k_p\left[e_i + \frac{T}{T_I}\sum_{j=0}^{i} e_j + \frac{T_D}{T}(e_i - e_{i-1})\right] + u_0 \tag{3-17}$$

u_i—第 i 个采样时刻的输出值；

e^i—第 i 个采样时刻的系统输出偏差，也即 PID 调节器的输入值；

e^{i-1}—第 i-1 个采样时刻的系统输出偏差；

u^0—开始进行 PID 控制时原始的控制值；

如果采样周期 T 取得足够小，上述近似计算相当准确，被控制过程与连续控制过程十分接近。式(3-17)表示的控制算法是直接定义计算的。它直接给出了全部控制量的大小，所以称为直接全量式(或称位置式)PID 控制算法。有的系统采用步进电机等增量型执行机构，执行机构需要的控制量，不是位置量的绝对数值，而是其增量值，则应采用增量式 PI 算法。增量的 PID 算法可在式(3-17)的基础上导出，只要先写出调节器在第 i-1 个采样时刻的输出值 u^{i-1}：

$$u_{i-1}=K_P\left[e_{i-1}+\frac{T}{T_I}\sum_{j-0}^{i-1}e_j+\frac{T_D}{T}(e_{i-1}-e_{i-2})\right]+u_0 \quad (3\text{-}18)$$

再将式(3-17)减去式(3-18)，稍加整理，就可以导出下面的增量式 PID 控制算法公式：

$$\Delta u_i=u_i-u_{i-1}=K_P\left[e_i-e_{i-1}+\frac{T}{T_I}e_i+\frac{T_D}{T}(e_i-2e_{i-1}+e_{i-2})\right] \quad (3\text{-}19)$$

电机控制是实时性很强的快速控制，微机采用的控制算法，要尽可能缩短计算时间，如果按式(3-19)计算Δui，每次要做 3 次加法，2 次减法，4 次乘法和 2 次除法。为了提高运算速度，可将该式稍加整理，改写成另外一种形式：

$$\Delta u_i=K_P\left[e_i-e_{i-1}+\frac{T}{T_I}e_i+\frac{T_D}{T}(e_i-2e_{i-1}+e_{i-2})\right]=K_P\Delta e_i+K_I e_i+K_D\Delta^2 e_i \quad (3\text{-}20)$$

总之，增量式算法只需要保留现时以前三个时刻的偏差值即可。与位置式算法相比，增量式 PID 算法的计算工作量小得多。鉴于上述优点，增量式 PID 算法比位置式应用更为广泛，它不仅适用于增量式控制，也用于位置式控制。所以本设计也采用增量式 PID 算法来设计软件，其流程图如图 3-8 所示

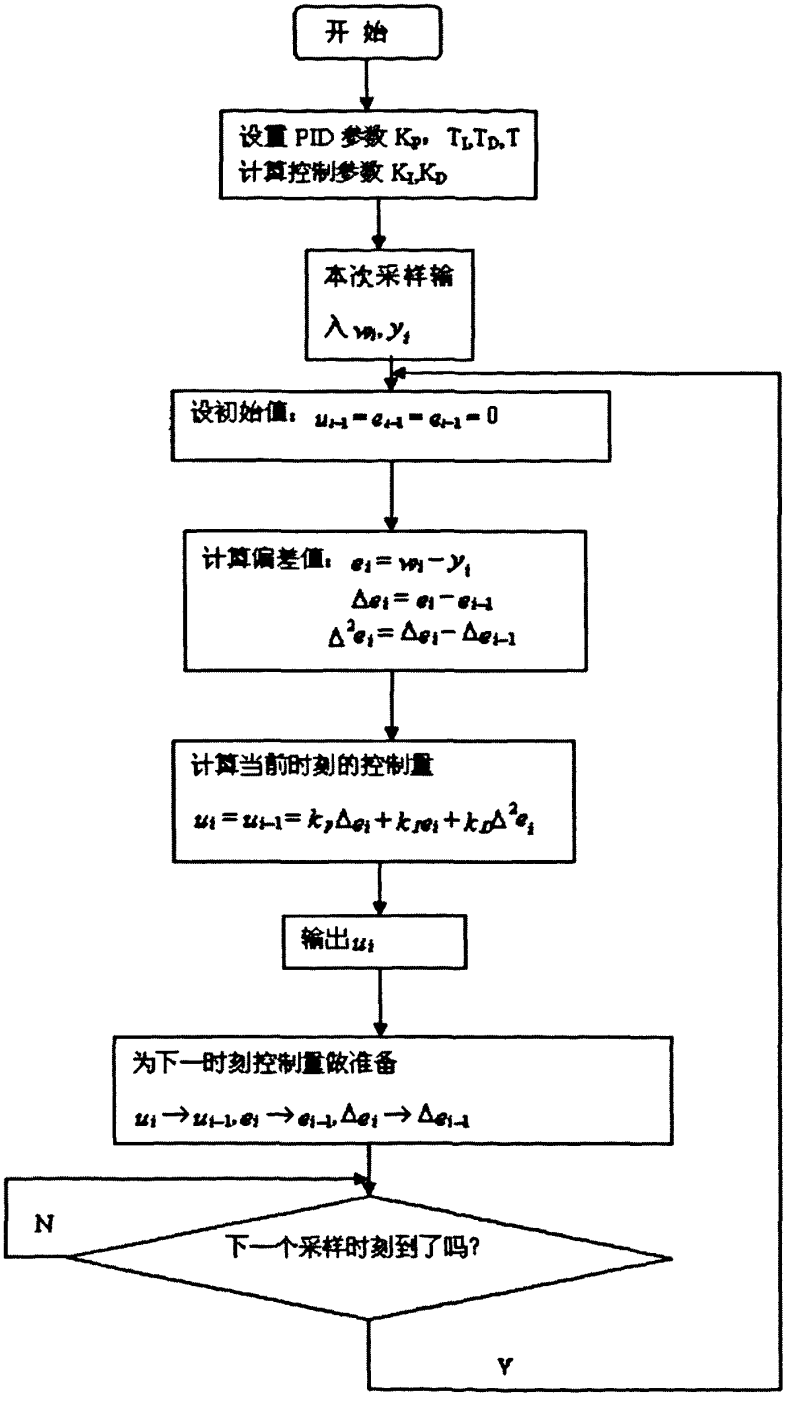


图 3-8 PID 控制算法流程图

Fig3-8 The flow chart of PID control algorithm

3.5.2 直线插补算法

该模块以数字积分法为基本原理，将直线插补算法做成一个函数，进行直线和圆弧的插补运算时，计算机根据插补运算的结果循环调用插补函数来实现。DDA 直线插补软件实现的思路是模拟硬件的逻辑关系完成。相应流程图如图 3-9 所示。流程图中 JVX，JVY 分别寄存 X 坐标和 Y 坐标的终点值，J_{RX} 和 J_{RY} 分别寄存 X 坐标和 Y 坐标的累加余数，J_M 为累加次数。

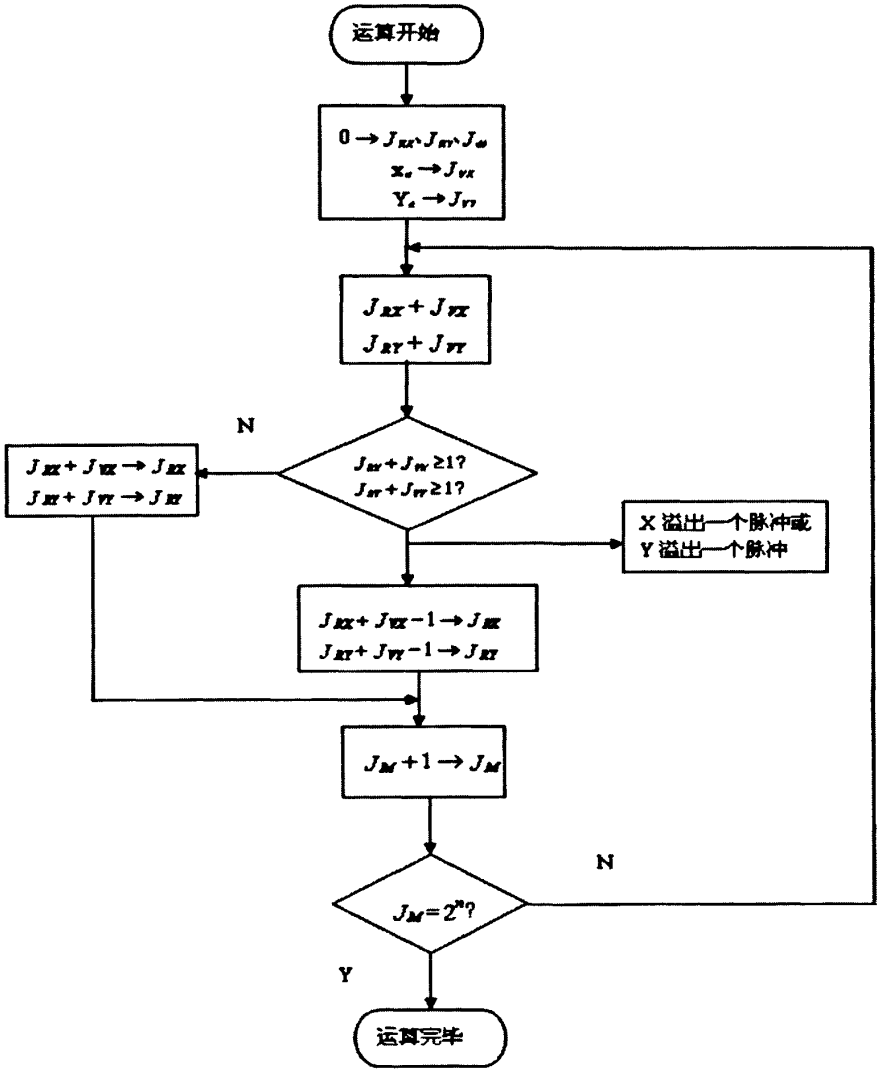


图 3-9 直线插补流程图

Fig3-9 The flow chart of line interpolation

3.6 PID 模糊融合控制

在计算 PID 控制中，使用的是数字 PID 控制器。PID 参数对闭环系统特性有着不同的影响，理解各个参数的特性有助于手工调整 PID 参数的过程。一般而言，比例增益 KP 增大，上升时间 tr 减小，稳态误差减小但不会完全消除；积分增益 KI 的加入，可以消除稳态误差，但 KI 过大有可能使系统的动态特性变坏；微分增益 KD 可以改善系统的动态特性，减小超调量 δ，并提高系统的稳定性。

3.6.1 模糊控制的基本原理

在控制工程中，有一些复杂的被控对象难以用一般的规律来描述，以致不可能建立数学模型。对于这类不具有数学模型的被控对象(或过程)，应用传统控制理论，包括现代控制理论很难取得满意的控制效果。然而，这类被控对象(或过程)在人的手动操作下却往往能正常运行。以手动控制策略为基础，并在此基础上实现模糊控制。在模糊控制中，模糊控制器的作用在于通过计算机，根据由精确量转化来的模糊输入信息，按照总结手动控制策略取得的语言控制规则进行模糊推理，给出模糊输出判决，并再将其转化为精确量，作为反馈送到被控对象(或过程)的控制作用。可见，模糊控制体现了模糊集合理论，语言变量及模糊推理在不具有数学模型，而控制策略只有以语言形式定性描述的复杂被控过程中的有效应用模糊控制的核心部分是模糊控制器，模糊控制器的规律由计算机的程序实现。实现模糊控制算法过程描述如下：微机经中断采样获取被控制量的精确量，然后将此精确量与给定值比较得到偏差信号。一般选偏差信号 e 和偏差变化率 ec 作为模糊控制器的输入量，对它们进行模糊化变成模糊量。偏差 e 和偏差变化率 ec 的模糊量可用相应的模糊语言表示，得到偏差 e 和偏差变化率 ec 模糊语言集合的一个子集 E 和 EC，再由 E，EC 和模糊控制规则 R(模糊算子)根据推理的合成规则进行模糊决策，得到模糊控制量 U，再将 U 解模糊，便可得到精确的控制量 u。要设计一个模糊控制器以实现语言控制，必须解决以下问题：

1）精确量的模糊化，把语言变量的语言值化为适当论域上的模糊子集；

2）模糊控制算法的设计，通过一组模糊条件语句构成模糊控制规则，并计算模糊控制规则决定的模糊关系；

3）输出信息的模糊判决，并完成由模糊量到精确量的转化。

本课题的机器人控制可看作是两个单关节的转角位置控制，在关节转角位置控制中也是采用双输入单输出的模糊控制器，其误差就是交流伺服电机的指令位置和电机实际输出位置之差。这类模糊控制器的控制规则通常是由模糊语句：if E and EC then U 来表达，这是模糊控制器中最常用的一种规则，它反映非线性比例加微分(PD)控制规律。控制规则是设计模糊控制器的关键。

3.6.2 控制方法融合的基本思想

在现有控制方法中选择用于融合的基本控制方法，选择原则为：在所有控制方法所对应的控制全域中，选择尽可能少的控制方法并使这些控制方法所对应的控制子域的并集尽可能接近控制全域；这些控制方法之间要存在很强的互补性，例如将 PID 控制和鲁棒控制相结合既能发挥 PID 控制稳态性能好的特点，又能保证系统在发生过程摄动和模型失配时的稳定性。编制基本控制方法各自独立的控制模块，运用数据融合技术，如参数估计、自适应加权、综合推理等，构建具有多源多类信息数据挖掘及各种综合分析和决策处理能力的控制融合的基本框架。借助控制性能评价，通过自学习自适应推理机进一步调整匹配可行决策使融合的正确决策达到最佳。多传感器信息源进行数据融合，即将来自多传感器或多源的信息和数据模仿专家的综合信息处理能力进行智能化处理，从而得出更为准确可信的结论，此为原始融合。然后各种控制方法分别独立地对原始融合后的数据进行处理得到各自的输出，根据原始融合及不同的控制目标要求和系统所处的状态将各种控制方法所得的输出进行特征提取、加权、综合推理等，实现系统的二次融合，得到优化的决策输出作用于被控对象。并利用控制性能评价体系反馈的信息来评价控制融合的效果，在实验的基础上完善已形成的知识库达到自学习自适应地对控制融合进行寻优的目的，机器人模糊 PID 融合控制虽然融合的成功应用研究已很广泛，但是融合理论本身却至今仍未形成基本的理论框架和广义有效的融合模型及算法。目前对融合问题的研究都是根据问题的种类，各自建立直观融合准则，并在此基础上形成所谓的最佳融合方案。因此融合方法是具体的，是灵活的，没有统一的固定模式。本课题的控制融合即根据具体情况采用数据融合的方法来将几种控制方法的输出进行融合，以期得到较为理想的效果。对于机器人这样复杂的控制对象，采用单独的 PID 控制，其动态响应差，存在明显的滞后，并且存在参数调整不方便，抗干扰能力差等缺陷。模糊控制是一种语言控制，不依赖于被控对象的数学模型、算法就能够直接从操作者中归纳、优化得到，抗干扰能力强、鲁棒性好，更重要的是其具有良好的快速响应。但模糊控制是一种非线性控制，存在着静态误差。

本文采用了传统的 PID 控制和模糊控制相结合的一种新型控制策略——模糊 PID 融合控制。在系统偏差较大时侧重于模糊控制，提高系统的响应；在系统偏差较小时侧重于 PID 控制，可以减小稳态误差，提高系统的静态性能。针对于此两方法的这一具体情况我们采取了自适应加权的融合方法来体现在控制的不同阶段各方法所占比重不同。机器人的轨迹控制是一个非线性、时变、几个控制量间强耦合的复杂系统，因而选用模糊控制与 PID 控制相结合的方法，既保持 PID 控制的优点，又具有模糊控制的特点，其中加权因子 α_1 和 α_2 取值范围为[0，1]。通过实时地改变加权因子 α_1 和 α_2 的值，可实现对 PID 控制和模糊控制的加权程度的调整，充分发挥模糊控制和 PID 控制各自的优点，

避免各自的缺点。在具体的操作过程中，可根据偏差的值来决定加权因子的大小，当偏差较大时，偏重于模糊控制，则加权因子 α i 取较小的值；偏差较小时，偏重于 PID 控制，加权因子 α i 取较大的值。加权因子的具体值是在实验的基础上确定的。

3.7 本章小结

本章根据该机器人本体结构的实际尺寸，通过对其进行运动学分析，建立了正、逆向运动学数学模型，进行了轨迹规划，阐述了交流伺服控制技术，而后介绍了传统的数字 PID 控制和模糊控制的基本原理，并将其应用于机器人的轨迹跟踪控制。然后通过分析两种控制方法的优缺点看到存在很强的互补性，提出了 PID 模糊融合控制，PID 控制有好的稳定性，及消除稳态误差的能力，但其动态响应差，存在较大的滞后。模糊控制响应速度快，对误差变化反应灵敏，可以快速地减小误差，但其消除系统稳态误差的能力比较差，存在控制上的盲区和死区，难以达到较高的精度。而 PID 模糊融合控制有机地将两者结合起来，实现优势互补。除了具有良好的静态性能外，还同时兼备不错的动态性能。

4 TMS320LF2407 运动控制卡硬件电路设计

4.1 运动控制卡硬件电路的总体设计思想

运动控制卡以美国 TI（Texas Instrument）公司的 16 位定点 DSPTMS320LF2407 为核心微处理器，它集成了编码器信号采集和处理电路，D/A 输出电路，扩展存储器电路和 DSP-PC 机通信电路。PC 机把粗插补的数据通过 DSP-PC 机通讯接口双口 RAM 传递给轴控制卡系统，DSP 通过对光电编码器反馈信号处理电路的结果分析，计算出与给定位置的误差值，再通过软件 PID 算法调节器获得位置控制量，计算出速度控制量，产生的输出信号经 D/A 转换将模拟电压量送伺服放大器，通过对伺服电机的控制实现对位置的闭环控制。控制卡上，使用 Altera 公司的可编程逻辑器件 EPM7128 实现数字逻辑电路设计包括地址译码、编码器处理电路，降低了板卡的设计尺寸，增加了电路板的可靠性和设计灵活性。它的在线可编程特性可以使得数字逻辑设计的硬件设计如同软件设计一样简便。系统总体框图如 4-1。

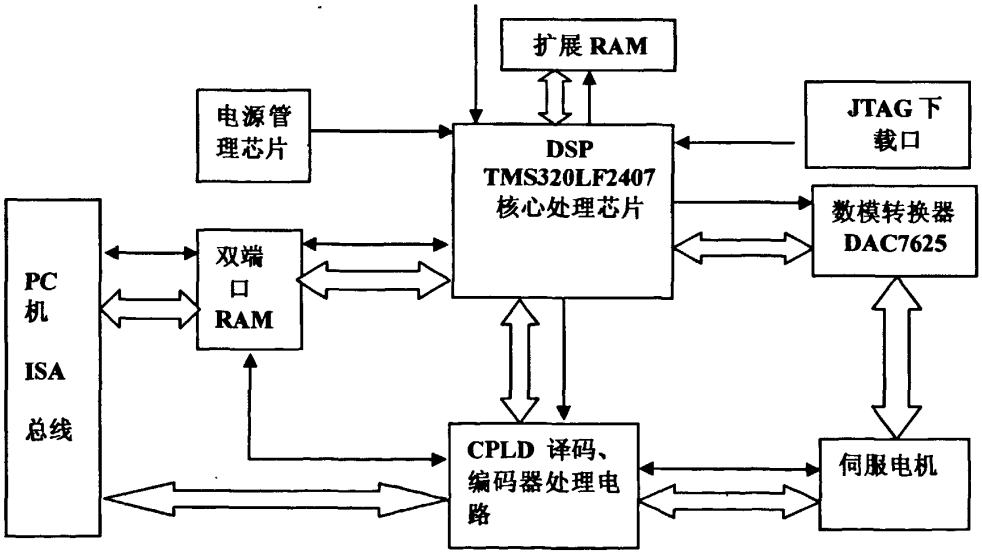


图 4-1 DSP 系统原理图

Fig4-1 The principle of system based on DSP

4.2 DSP 芯片的特点和功能分析

4.2.1DSP 芯片的特点

定点 DSP 主要特点可以概括如下：

1）哈佛结构。

程序存储器和数据存储器相互分开各占独立的空间，允许取指令和执行指令全部重叠进行；可以直接在程序和数据空间之间进行信息传送，减少访问冲突，从而获得高速运算能力。

2）采用管道式设计加快执行速度。

即采用流水线技术，取指令、译码和执行指令等操作可以重叠进行。

3）在每一时钟周期中执行多个操作。

DSP 的每一条指令都是自动安排空间、编址和取数，支持硬件乘法器，使得乘法能用单周期指令完成。这样可以提高执行速度（通常 DSP 的指令周期是纳秒级）。

4）具有专门的硬件支持复杂的 DSP 编址。

5）特殊的 DSP 指令。

6）面向寄存器和累加器。

DSP 所使用的是专用的寄存器，许多 DSP 还有大的累加器，可在异常情况下对数据溢出进行处理。

4.2.2 面向电机控制的 DSP 芯片

随着微处理器、微控制器和数字信号处理器（DSP）的性价比的提高，它们越来越多地在运动控制系统中得到应用。现在，工程师们可以利用不断发展的控制技术和高性能的处理器设计数字控制系统，这种控制理论和实现手段的并行发展使得许多复杂而有效的控制算法能迅速地用到实际生产过程中。尤其是 DSP 器件的优越性能，以及各 DSP 厂商不断推出针对运动控制的 DSP 芯片，使得 DSP 成为复杂运动控制系统（如机器人运动控制）和高性能运动控制器中的首选器件。作为 TMS320C2xx 产品族中的一员，LF2407 是一个高性价比的专门为电机及运动控制系统设计的 DSP 器件，其结构如图 4-2 所示，它在单片处理器中集成了高性能的 TMS320CxLPDSP 内核（运算能力为 30MIPS）、为电机控制而优化的事件管理器及 PWM 输出接口、可同时完成采样的双工 A/D 转换器、并行的电机电流读数转换与通讯、快闪存储器（Flash）或 ROM 程序存储器，还包括同步、异步串行外设接口、比较单元、通用定时器及与光电编码器接口的编码单元等片上资源。这种高度集成的单片数字控制方案比现存的多片方案具有更高的电机驱动能力及更低的系统费用。

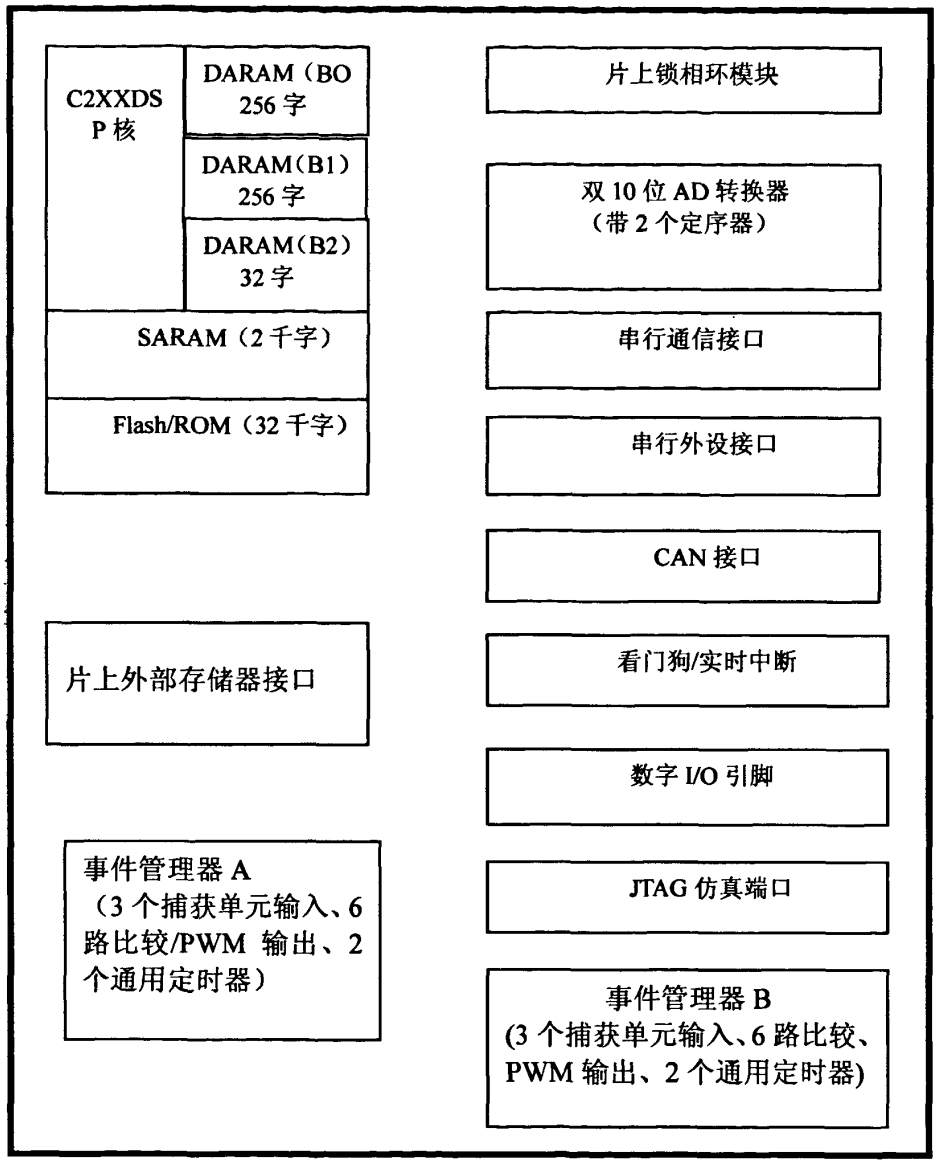


图 4-2 TMS320C2xx 系列 DSP 的结构图

Fig4-2 Block diagram of TMS320Lx24xseries of DSP

TMS320LF2407 的资源如下：

1）采用高静态 CMOS 技术，使得供电电压降为 3.3V，减小了控制器的功耗；30MIPS 的执行速度使得指令周期缩短到 33ns(30MHz)，从而提高了控制器的实时控制能力。

2）基于 TMS320C2xxDSP 的 CPU 核保证了 TMS320LF240X 系列 DSP 代码和 TMS320 系列 DSP 代码兼容。

3）片内高达 32K 字的 FLASH 程序存储器，高达 1.5K 字的数据/程序 RAM，544 字双口 RAM（DARAM）和 2K 字的单口 RAM（SARAM）。

4）两个事件管理器 EVA 和 EVB，每个包括：两个 16 位通用定时器；8 个 16 位的脉冲调制（PWM）通道。它们能够实现：三相反相控制；PWM 的对称和非对称波形；当外部引脚 PDPINTx 出现低电平时快速关闭 PWM 通道；可编程的 PWM 死区控制以防止上下桥臂同时输出触发脉冲；3 个捕获单元；片内光电编码器接口电路；16 通道 A/D 转换器。事件管理器模块适用于控制交流感应电机、无刷直流电机、开关磁阻电机、步进电机、多级电机和逆变器。

5）扩展的外部存储器（LF2407）总共 192K 字，64K 字程序存储器；64K 字数据存储器；64K 字 I/O 寻址空间。

6）看门狗定时器模块（WDT）。

7）10 位 A/D 转换器最小转换时间为 500ns，可选择由下列两个事件管理器来触发的两个 8 通道输入 A/D 转换器或一个 16 通道输入的 A/D 转换器。

8）控制局域网络（CAN）2.0B 模块。

9）串行通信接口（SCI）。

10）16 位的串行外设接口模块（SPI）。

11）基于锁相环的时钟发生器。

12）高达 40 个可单独编程或复用的通用输入/输出引脚（GPIO）。

13）5 个外部中断（电机驱动保护、复位和两个可屏蔽中断）。

14）电源管理器包括 3 种低功耗模式，并且能独立将外设器件转入低功耗模式。

4.3 芯片各个功能模块分析

4.3.1CPLD 计数模块

a EPM7128 介绍

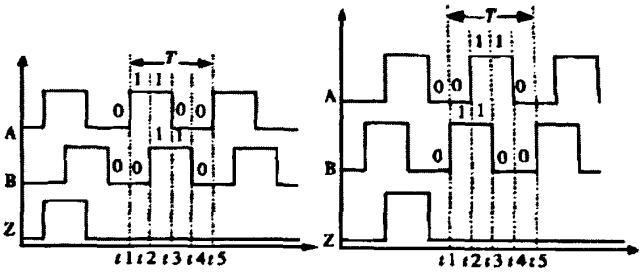
CPLD 是 ComplexPLD 的简称，是一种较复杂的 PLD 逻辑元件。CPLD 具有高整合性的特点，故其有性能提升，可靠度增加，PCB 面积减少及成本下降等优点。CPLD 元件的基本结构由许多个逻辑方块（Logic Blocks）组合而成，而各个逻辑方块均相似于一个简单的 PLD 元件。逻辑方块间的相互关系则由可编程的连线进行架构。经过几十年的发展，许多公司都开发出了 CPLD 可编程逻辑器件，门类较为齐全，比较著名的 CPLD 制造厂商有 Altera、Lattice、Xilinx 等。

我们使用 Altera 公司提供的 MAX+Plus II 集成开发环境编写 CPLD 程序。MAX+Plus II 提供了一种与结构无关的设计环境，使 Altera 的 CPLD 系列设计者能方便地进行设计输入、快速处理和器件编程配置，提供电路图、文本和波形等设计输入方式，执行编译和逻辑综合，仿真和定时分析，以及器件编程等工作。MAX+Plus II 具有 EDIF、VHDL、VerilogHDL 等网表接口，便于为工作在工业计算机和工作站上的各种工具提供附加的设计输入与仿真支持。MAX+Plus II 的所有平台都包含 300 多个 74 系列宏函数和

VHDL(Altera 硬件描述语言)，支持状态机、布尔方程式、组合逻辑和真值表输入多种方法。MAX+Plus II 还提供高度自动化的编译、自动多器件的划分、定时仿真和分析、自动错误定位、器件编程和验证以及综合的在线求助系统。

b 光电码盘原理

为了测量机器人各关节运动的速度和位置，在每个关节的伺服电机上都装有一个 n 位的光电码盘，伺服电机每转过一圈，光电码盘就会输出 2n 个脉冲，DSP 运动控制卡读取光电码盘脉冲，就可以计算出机器人各关节的位置和速度。光电码盘有相对式码盘和绝对式码盘之分。相对式码盘输出两个相位差为 90° 的正交信号 A 和 B，以及零位脉冲信号 Z，A、B 两相信号的脉冲数标志码盘轴所转过的角度，A、B 之间的相位关系标志码盘的转向，当 A 相超前 B 相 90° 时，标志码盘正转，见图 4-3（a）当 B 相超前 A 相 90° 时，码盘反转，见图 4-3（b）。



(a)码盘正转时输出信号　　(b) 码盘反转时输出信号

图 4-3 码盘的正交脉冲输出

Fig4-3 Quadrature encoder pulse output

绝对式码盘输出码盘位置的绝对值，只需要读取这些绝对值，就可以知道码盘的当前位置和速度。本系统使用的是 16 位绝对式光电码盘，和伺服驱动器进行连接，伺服驱动器既可以输出绝对码盘信号，也可以将绝对码盘信号转化为相对码盘信号输出，本设计中采用相对式码盘方式。光电码盘的脉冲周期 T 对应的码盘角位移固定为 θ，故其量化误差为 θ/2，如果能够将 A 或 B 信号四倍频，则计数脉冲的周期将减小到 T/4，量化误差下降为 θ/8，从而使光电码盘的角位移测量精度提高 4 倍。观察图 4-3 可以发现，在脉冲周期 T 内，A、B 两相信号共产生了四次变化，分别为 10、11、01、00。利用 A、B 两方波信号之间相位关系，能够产生四倍频信号。四倍频后的码盘信号，需经计数器计数后，才能转化为机器人关节的相对位置，计数过程一般有两种实现方法：第一种方法是由可编程计数器或微处理器内部定时计数器实现计数；第二种方法是由可逆计数器实现对正反向脉冲的计数。当需控制的电机数量少时，前一方案附加元件少，结构简单，较为容易实现，当需控制的电机数量较多时，则采用后一种方案，利用复杂可编程逻辑器件 CPLD 实现。

4.3.2DSP-PC 机通讯电路

DSP 与 PC 机之间需要高速地传送大量的数据，为了提高两者之间的通讯速度,选用双口 RAM 来提高通讯速度，双口 RAM 作为一种特殊的 RAM 芯片，在高速数据采集处理系统中得到广泛的应用。它具有两个独立的端口，各自均有一套独立的数据总线、地址总线和控制总线，允许两个端口独立地对存储器中的任何单元进行存取操作。当两个端口同时对存储器中的同一单元进行存取操作时，可由内部仲裁逻辑决定优先权。

这里是选用美国 IDT 公司生产的双口 RAM 芯片 IDT7132 它是 2K×8bit 高速静态双口 RAM，存取速度为 35ns（民用），IDT7132 在数字信号处理领域应用已经比较普通。它的时序与 DSP 的时序相配合，特别适用于 DSP 与 PC 机之间的大量数据高速双向传送。该芯片均提供两个带有自身的控制、地址和 I/O 引脚的独立端口，它允许独立地读写存储器中的任何单元。IDT7132 带有片内硬件端口仲裁电路，可以允许双机同步地读或写存储器中的任何单元，同时保证数据的完整性。它的竞争原则是：

1）左右两端口的地址信号同时到达，那么谁的 CE 片选信号先到，慢的一方 BUSY 线下拉，直到快的一方访问完毕。

2）左右两端口的片选信号同时到达，那么谁的访问地址信号先到，慢的一方 BUSY 线下拉，直到快的一方访问完毕，DSP 与 PC 机的通讯电路如图 4-4 所示。采用双口 RAM 作为 TMS320LF2407 与 PC 主机之间的通讯接口不但可以简化通讯接口电路的设计，而且提高了数据交换的速度，增强了控制卡的实时性。

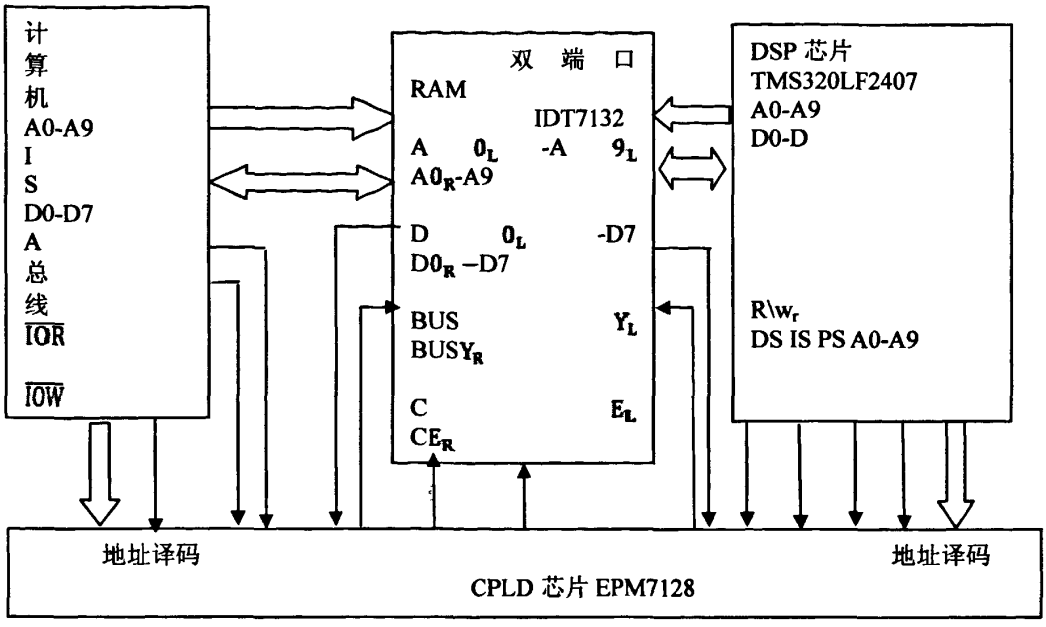


图 4-4 DSP 与 PC 机通讯电路图

Fig4-4 Communication diagram between DSPand PC

4.3.3 扩展 RAM 电路

TMS320LF2407 有 1.5K 字的数据/程序 RAM，544 字双口 RAM（DARAM）和 2K 字的单口 RAM（SARAM），但是考虑到该轴卡所需的程序储存空间和数据储存空间较大在 DSP 外部用两片 CY7C1021 芯片扩展成 128Kx16bit 的片外存储器，其中 64k 作为数据存储器，数据存储空间用来保存采集的实时数据，处理过的数据和运算过程中的许多中间变量，当 TMS320LF2407 访问片内数据存储器时，外部存储器信号 $\overline{DS}$ 和 $\overline{STRB}$ 处于高阻状态，$\overline{DS}$ 信号直接和外部 RAM 的 $\overline{CS}$ 连接，当 TMS320LF2407 访问外部数据存储器时，一个有效的 $\overline{DS}$ 信号表明外部总线正被数据存储器占用，$\overline{DS}$ 处于低电平。另一个芯片 64k 作为程序存储器，程序存储器空间保存应用程序代码，也可以保存表信息和操作常数，当 TMS320LF2407 访问片内程序存储器时，外部存储器信号 $\overline{PS}$ 和 $\overline{STRB}$ 处于高阻状态，$\overline{PS}$ 信号直接和外部 RAM 的 $\overline{CS}$ 连接，一个有效的 $\overline{PS}$ 信号表明外部总线正被程序存储器占用，$\overline{PS}$ 处于低电平。对于存储器接口，选择具有快速存取时间的外部存储器非常重要。如果没有快速的存储器，或者不考虑速度问题，可使用 READY 信号和片内等待状态发生器来产生等待状态，满足快速器件 DSP 与慢速器件之间的接口问题。而对于本系统，终端部分的数据处理量比较大和数据处理速度要快，因此需要比较快速的存储器接口。

4.3.4DSP 接口电路

a 时钟信号

TMS320LF2407 上集成有 PLL（锁相环）时钟模块，可以为 DSP 芯片及外设提供所需的时钟信号。PLL 可提供 4 种时钟信号：CPUCLK 供 CPU 内核、片内寄存器、片内设备使用；SYSCLK，一般为 CPUCLK 的 1/2 或 1/4；ACLK，供模拟模块使用，一般为 1MHZ；WDCLK，供 WATCHDOG 计数/实时中断模块使用，一般为 16KHZ。PLL 模块可以按照 4、2、1.33、1、0.8、0.66、0.57、0.5 的倍率进行倍频或分频。这里的时钟信号由外部提供，由于 TMS320LF2407 的速度达到 33ns，所以选择 15MHZ 的有源晶振，从 XTALI/CLXIN 输入，经 PLL 和 PLL2 倍频成 30MHZ 信号供 DSP 使用，管脚 XTAL2 悬空。

b 测试接口

在 TMS320LF2407 上设置有符合 IEEEll49 标准的 JTAG（Joint Test ActionGroup）标准测试接口及相应的控制器，不但能控制和观察多处理器系统中的每个处理器的运行，测试每块芯片，还可以通过这个接口来装入程序。利用 PC 机和 TI 提供的仿真器，接通 JTAG 接口，就可以在 PC 上运行软件去控制 DSP，和 JTAG 测试口同时工作的还有一个分析模块，它支持断点设置和程序存储器、数据存储器、DMA 的访问，程序的单步运行和跟踪，以及程序的分支和外部中断计数等。

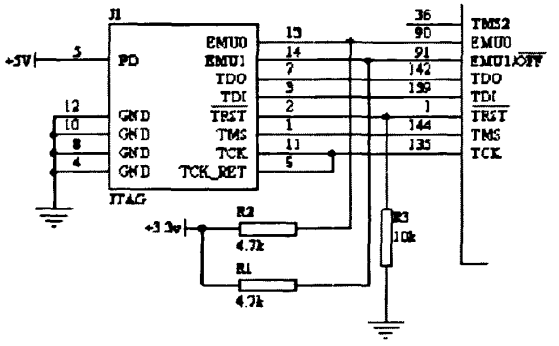


图 4-5 JTAG 接口电路图

Fig4-5 JTAG interface circuit diagram

为了方便调试，在控制卡上设置了 JTAG 接口。它是一个 14 针的接口，可以同 TI 的仿真器相连。JTAG 接口与 TMS320F2407 的连接如图 4-5 所示，EMU0 和 EMU1 信号加上拉电阻的目的是为了保证信号上升时间小于 10us。

c 外部 IO 信号处理

通过管脚引入的硬件中断，包括轴限位中断和编码器 INDEX 信号中断。每个控制轴有正反方向的两个限位开关，产生两个限位信号，4 个轴共 8 个限位信号：limit1+、limit1-、limit2+、limit2-、limit3+、limit3-、limit4+、limit4-，其中“＋”表示正限位，“-”表示负限位。这几个信号通过 CPLD 的相与之后接到 DSP 的中断管脚 $\overline{\text{NINT1}}$，同时这些信号通过光耦电路接入 DSP 的 IO 口。当某一轴运动到限位开关处时，就会触发 DSP 的外部中信号 $\overline{\text{NINT1}}$，然后 DSP 就可以判定哪个限位开关到位了。8 个限位开关分

别接到 DSP 的 IO 口复用控制位 IOPC2-IOPC7、IOPB4-IOPB5,只要设置 MCRB（地址：7092H）、MCRC（地址：7094H）数字 I/O 控制寄存器的功能配置 MCRB.4-MCRB.5、MCRC.2-MCRC.7 为“0”，使这些复用管脚处于 I\O 功能。

用户操作时，有可能错误的将高压信号输入接口与轴卡相连接；外界设备变化比较多，也能产生非正常高压信号。为防止外界高压信号损伤轴控制卡，保证轴控制卡的可靠性和安全性，输入信号都将采用光耦隔离。通过光信号传递避免将高压大电流信号引入控制卡。光耦器件选用 HP 公司的高速光耦 6N137，它的平均输入驱动电流为 50mA，平均输出驱动电流为 5mA，可直接驱动 TTL 电路；耐受 3000V 高压和 5kV/s 的瞬时高压。编码器的 INDEX 信号处理同上面相类似。每个轴能产生一个 INDEX 信号，4 个轴有 4 个 INDEX 信号。这 4 个信号通过逻辑与门产生一个中断信号，接到$\overline{\text{XINT2}}$,同时接到 DSP 的 I/OF 口，供中断产生时 DSP 读入。

d IO 地址译码

4 路编码器反馈信号处理模块和 4 路数模转换模块的片选信号由 DSP 提供。TMS320LF2407 支持 64K 的 I/O 空间。$\overline{\text{IS}}$、R / W、WE、A2、A3、A4 参与译码,由 EPM7128 实现译码，为外设提供控制信号。DAC 处模块 DAC0～DAC3 占用的 I/O 空间分别为 00H～04H，编码器信号处理模块 ENCONDER0～ENCODER3 占用的 I/O 空间分别为 08H、0CH、10H、14H。IO 译码电路由 EPM7128 实现如图 4-6 所示：

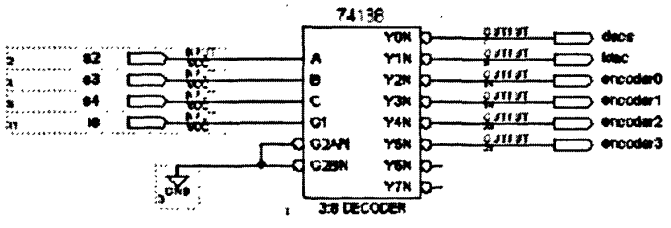


图 4-6 IO 译码电路

Fig4-6 Circuit diagram of IO encoding

4.3.5 数/模转换输出电路

数模转换电路的核心芯片采用 BURR-BROWN 公司生产的 12 位四路电压输出的数模转换芯片 DAC7625，它接收 12 位并行的输个内部输入寄存器的反馈模式，它有低功耗 12mw。DAC7625 能在单极电源＋5V 或双极电源＋5V 至－5V 下工作。每个 DAC 的低功耗、小体积使 DAC7625 特别适合于闭环伺服控制系统。其中 MC1403 是一个基准电源，它提供了一个+2.5V 的基准电压作为供给 DAC7625 的参考电压，如图 4-7 所示。

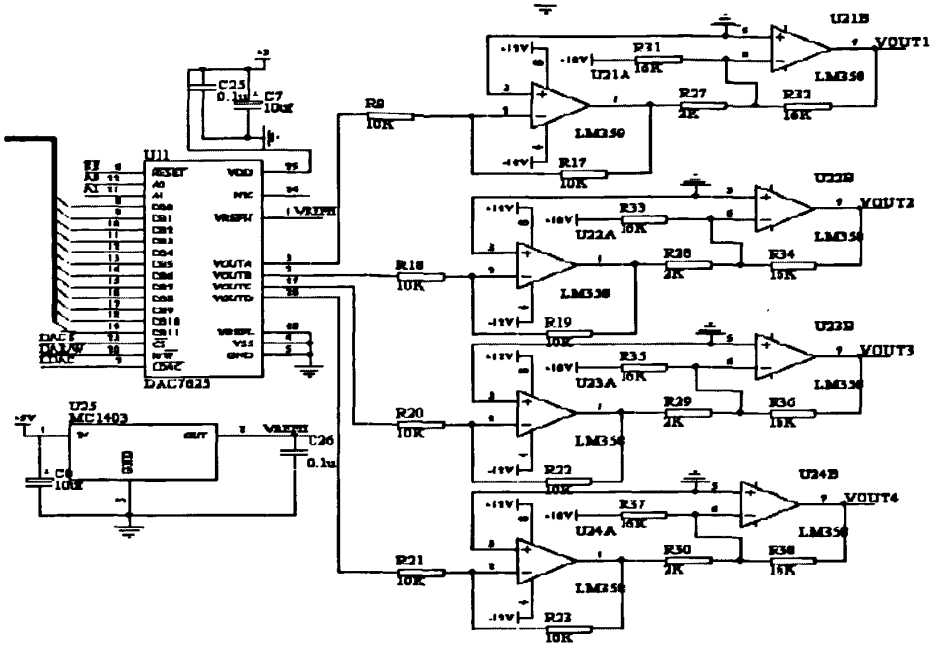


图 4-7 D/A 转换电路

Fig4-7 D/A conversion circuit

表 4-1 DAC 的 I/O 地址

Table4-1 DAC I/O addresses

I/O 地址	通道
0x0000	1
0x0001	2
0x0002	3
0x0003	4
0x0004	转移

本设计采用单极电源方式，DAC7625 的 Vrefh 接+2.5V，Vrefl 接地，Vdd 接+5V，Vss 接地，输入范围 0～+2.5V，DAC 的片选信号由 IS、STRB、A2、A3 经过译码得到，DAC 的四个通道在 DSP 中的 IO 地址为：0000H，0001H，0002H，0003H，DAC 更新的地址为 0004H，如表 4-1 所示。由于 DSP 只能接收 3.3V 的信号，而 DAC7625 是 5V 供电，DAC7625 与 DSP 之间通过 74LVC245 的电压转换芯片传输数据，DAC7625 的四

个通道信号经过 LM358 房大器的两级运放电路后能够得到-10V 到+10V 的电压，由于输入的电压范围是 0～+2.5V，经过运算放大器的第一级运放后得到的电压范围是 0～-2.5V，然后经过运算放大器的第二级运放后得到的电压范围是-10V～+10V，并与 000H－FFFH 的数字量相对应。

4.4 本章小结

本章详细讨论了机器人运动控制系统的硬件电路的总体设计思想。将运动系统分为 CPLD 计数模块、DSP-PC 机通讯电路模块，扩展 RAM 电路模块，DSP 接口电路模块，数/模转换输出电路模块等 5 大模块，讨论了芯片的选型，各芯片的工作原理、DSP 芯片 TMS320LF2407 在电机控制方面的特点和功能，自此，我们成功搭建了机器人控制的硬件平台为进一步的硬件调试和软件设计打下了良好的基础。

5 软件设计及应用

5.1 控制软件设计的思路

因 DSP 其强大的计算能力，我们在设计软件时，在实时性一定能够满足的条件下，只需要考虑运动控制的精度，因此在 DSP 的 FlashRom 里面固化了大量的算法来提高运动控制的精度，同时也提高了运动控制卡的开放性，软件还提供各种动态联接库函数，以方便用户直接利用高级语言编程，对轴卡直接可以编程,实现用户想要实现的功能。本文软件设计时采用模块化设计，方便修改和调用，函数有初始化函数，轴中断处理函数，固化的算法有电机闭环控制所需要的 PID 算法，直线插补算法和圆弧插补算法，下文将以每个模块化设计的实现加以阐述。

动态链接库函数主要包括如下几种：

1）SYSINIT(VOID)　　初始化函数

2）READ_ENCODER(REF1,REF2,REF3...)　　读取码盘数

3）PID_CONTROLLER(REF1,REF2,REF3...)　　PID 控制函数

4）LINE_MOVE(REF1,REF2,REF3...)　　直线插补函数

5）CIRCLE_MOVE(REF1,REF2,REF3...)　　圆弧插补函数

6）INTTER(REF1,REF2,REF3...)　　中断控制函数

STSINIT(VOID)函数，负责对运动控制单元进行初始化，特别是对 DSP 的中断，时钟，I/O 口，定时器等进行初始化，同时清除各种状态。在使用过程中，可以用于自动寻找机器零点。READ_ENCODER(REF1,REF2,REF3…)函数，读取码盘反馈的计数为程序判断机器当前位置、速度、加速度提供依据，PID_CONTROLLER(REF1,REF2,REF3…)函数给 DSP 中的固化算法提供具体的参数，如微分增益、积分增益、比例增益等等。LINE_MOVE(REF1,REF2,REF3…)函数给据用户提供的直线起点和终点实现插补过程。该函数和圆弧插补函数联合使用完成平面的任何连续图形作业，同时函数内部嵌入了加、减速等控制。

5.2 对函数进行初始化

DSP 在调用各个寄存器之前必须对 DSP 进行初始化，目的是定义 DSP 的各个口的功能，内容包括对存储器映像寄存器、中断结构、方式控制、寄存器控制、辅助寄存器和辅助寄存器指针，以及数据存储器页指针等的初始化和宏定义。行初始设定以确定工作方式、工作时钟、看门狗、等待状态发生器状态、可屏蔽中断设置等。

代码如下：

```
void SYSINIT( )
{
```

```
asm(" SETC INTM ");                    /*关总中断*/
asm(" CLRC SXM ");                     /*抑制符号扩展*/
asm(" CLRC OVM ");                     /*累加器中结果正常溢出*/
asm(" CLRC CNF ");                     /*B0 区被配置为数据空间*/
SCSR1=0x83FE;                          /*时钟 2 倍频，CLKIN=15M,CLKOUT=30M*/
WDCR=0x00E8;                           /*不使能 WDT*/
IMR=0x0000;                            /*屏蔽所有 CPU 中断*/
IFR=0xFFFF;                            /*清全部中断标志*/
MCRA=0xCFFF;                           /*配置 IOPB4,IOPB5,XINT1*/
MCRB=0xFF03;                           /*配置 IOPC2-IOPC7, XINT2*/
MCRC=0x03FF;                           /*配置 IOPF2-6*/
PBDATDIR=0x0000;                       /*设置 IOPB4,5 输入低电平有效*/
PCDATDIR=0x0000;                       /*设置 IOPC2-7 输入低电平有效*/
PFDATDIR=0x0000;                       /*设置 IOPF2,3 输入低电平有效*/
ACTRA=0x0666;                          /*PWM6,4,2 低有效，PWM5,3,1 高有效*/
ACTRB=0x0666;                          /*PWM8 低有效，PWM7 高有效*/
DBTCONA=0x02E7;                  /*使能死区定时器 3，2，1，死区周期为 3，9 分频*/
DBTCONB=0x02E7;                  /*使能死区定时器 6，5，4，死区周期为 3，9 分频*/
CMPR1=48;
CMPR2=48;
CMPR3=48;
CMPR4=48;
T1PR=95;
T3PR=95;
T2PR=0xFFFF;
T4PR=0xFFFF;
T1CNT=0;
T2CNT=1000;
T3CNT=0;
T4CNT=1000;
GPTCONA=0x0006;
GPTCONB=0x00                     /*禁止定时器比较输出，T4 低有效，T3 高有效*/
EVAIMRA=0x0000;                  /*禁止定时器 1 的中断，禁止比较单元 1-3 的中断，
```

```
                                    禁止功率驱动保护中断*/
    EVAIMRB=0x0000;                 /*禁止定时器 2 的中断*/
    EVAIMRC=0x0004;                 /*使能 CAP3 中断，禁止 CAP1,CAP2 中断*/
    EVBIMRA=0x0080;                 /*使能定时器 3 的周期中断，禁止比较单元 4-6 的中
                                      断，禁止功率驱动保护中断*/
    EVBIMRB=0x0000;                 /*禁止定时器 4 的中断*/
    EVBIMRC=0x0004;                 /*使能 CAP6 中断，禁止 CAP4,CAP5 中断*/
    EVAIFRA=0xFFFF;
    EVAIFRB=0xFFFF;
    EVAIFRC=0xFFFF;
    EVBIFRA=0xFFFF;
    EVBIFRB=0xFFFF;
    EVBIFRC=0xFFFF;
   COMCONA=0xA600;                  /*使能比较操作，使能比较输出，下溢或周期匹配时重
                                      装载比较和方式控制存器*/
    COMCONB=0xA600;                 /*禁止比较操作，禁止比较输出，下溢或周期匹配
                                      时重装载比较和方式控制寄存器*/
    T1CON=0x1442;                   /*连续增计数模式，预分频为 16，定时器计数使能，使
                                      用内部时钟*/
    T3CON=0x1442;
    T2CON=0x1872;                  /*定向增减计数模式，使能定时器操作，使用正交编码脉
                                      冲电路作为时钟输入*/
    T4CON=0x1872;
    CAPCONA=0x7404;                /*使能 QEP1,2,CAP3(T1,检测上升沿)*/
    CAPCONB=0x7404;                /*使能 QEP3,4,CAP6(T3,检测上升沿),禁止 CAP6*/
    XINT1CR=0x8001;                /*使能外部中断 XINT1*/
    XINT2CR=0x8001;                /*使能外部中断 XINT2*/
    IMR=0x000A;                    /*开 CPU 中断 INT2,开 INT4*/
    asm(" CLRC INTM ");            /*开总中断*/
    }
```

5.3 WDM 驱动程序

5.3.1WDM 构架

在Windows操作系统中，程序要么执行在用户模式，要么执行在内核模式。执行于

用户模式的程序不能直接访问硬件设备，它需要调用Win32API函数，Win32子系统模块通过调用平台相关的系统服务接口实现API，而相关的系统服务则调用内核模式支持的例程，从而达到访问硬件的目的。对于应用程序提出的请求，内核模式中存在有相应的服务例程，它们之间是通过I/O请求包（IRP）建立通信，I/O管理器将不同的IRP传递给不同的驱动程序例程，从而完成应用程序的请求，如图5-1所示。

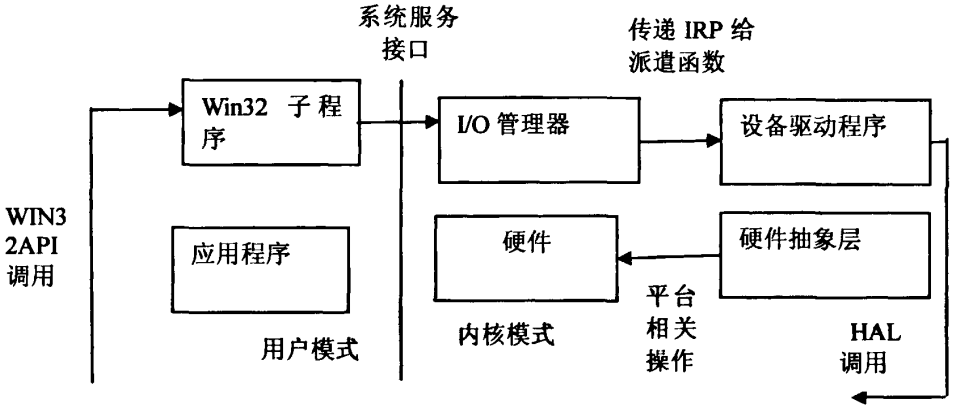


图 5-1 服务请求过程

Figure5-1 Service request process

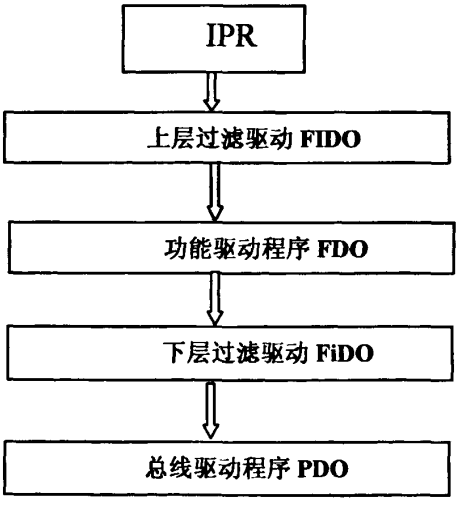


图5-2 WDM 驱动层次结构

Figure5-2 The structure of WDM driver

一个WDM驱动程序包含有多个例程，对于不同的IRP请求，它会调用不同的例程完成该IRP操作。其中有五个例程是必需的，即DriverEntry，AddDevice，DispatchPnP，DispatchPower，DispatchWMI。另外，还可能需要有StartIO例程对IRP进行排队，中断服

务例程（ISR）和延迟过程调用例程（DPC）处理硬件中断等，WDM驱动程序开发者可以根据自己开发驱动程序的要求选择所需的例程。WDM驱动程序采用图5-2所示的层次结构。层次结构中有一个由设备对象构成的堆栈，设备对象是系统为帮助软件管理硬件而创建的数据结构，一个物理硬件可以有多个这样的数据结构。处于堆栈最底层的设备对象称为物理设备对象(Physical Device Object)，简称为PDO。在设备对象堆栈的中间一个对象称为功能设备对象(Functional Device Object)，简称为FDO。在FDO的上面和下面还会有一些过滤器设备对象(Filter Device Object)，简称为FiDO，主要用于监视和修改IRP流。IRP先到达上层过滤驱动程序，做适当处理或不做处理，传递给功能驱动程序，添加需要的功能后，传递给下层过滤驱动程序，做适当处理或不做处理，最后传递给总线驱动程序，完成整个堆栈。

5.3.2 使用 DriverStudio2.7 开发 WDM 驱动程序

DriverStudio2.7是一套微软Windows平台下设备驱动程序的开发、调试和测试的工具包，DriverStudio2.7版本中包含了许多工具模块：DriverAgent、VToolsD、DriverWorks、DriverNetWorks、SoftICE等驱动程序开发调试工具，其中的DriverWorks以面对对象（OOP）的方式，将WDM和NT环境下的驱动程序编写所需要的与内核和硬件访问相关的API函数封装成类，简化了驱动程序开发的难度，DriverWorks嵌入到VC++6.0中使用，方便了驱动程序的开发。在DriverStudio2.7安装之前，先要安装Windows2000DDK，其内部封装了驱动程序开发所需的API函数。利用 DriverWorks的向导DriverWizard可以很方便地搭建驱动程序的框架，步骤如下：

1）第一步选择驱动程序工程的名字和所在的目录位置。

2）第二步选择驱动程序的类型。如果操作系统是 WindowsXP、Windows2000、Windows me或Windows98，选择WDM Driver，如果操作系统是 WindowsNT4.0，选择NT4.0 Driver。本驱动程序选择WDM Driver。

3）第三步选择驱动程序模型对象，分为WDM Function Driver和WDM Filter Driver两种。本驱动程序选择WDM Function Driver。

4）在第四步选择接口类型。本驱动程序选择PCI，并在Vendor ID和Device ID中分别输入厂商号和设备号，还需填入PCI Subsystem ID 和PCI Revision ID，这个要和PCI9052的配置完全相同。

5）第五步定义驱动程序的设备类，代表了驱动程序对应的硬件设备。例如可以设置设备类名字为PCI9052。

6）第六步选择需要添加的驱动程序访问函数，选择DeviceControl（和应用程序进行交换数据）和ClearUp（驱动程序停止或卸载时清除IRP例程）。

7）第七步选择IRP队列的排队方法，它决定了驱动程序检查设备的方式。如果选择

DriverManaged，则驱动程序会自动检查设备对启动特定的操作请求是否有效，选用DriverManaged比较灵活，但编程较复杂，如果选择SystemManaged，则所有的IRP排队都由系统完成。本驱动程序选择SystemManaged。

8）第八步设置设备启动时从注册表中装载的标识参数。当设备启动执行DriverEntry例程时，将从注册表中装载所设置的标识参数。

9）第九步是最关键的一步。首先在Resources中添加资源，在name中输入变量名，在PCIBaseAddress中输入0～5的序列号，0～5和PCI 配置寄存器的BAR0～BAR5相对应。Interface 属性页选择 GUID，在Buffer属性页中选择读写方式，用于描述与I/O操作相关的数据缓冲区，选择 Direct I/O方式。设置中断对话框中，name栏写入中断服务程序的名称，选中创建中断服务程序ISR和中断延迟调用DPC等，本驱动程序不需要。

10）在第十步加入与应用程序或者其它驱动程序通信的I/O控制代码参量。本驱动程序加入两个变量MEMREAD_IOCTL和MEMWRITE_IOCTL（也可以加入更加丰富的函数，比如读字节，读字，读双字，写字节，写字，写双字等函数），完成与应用程序的读写操作，在对应驱动程序函数中加入相应的读写代码。选择读写方式有Buffered、In_Direct、Out_Direct和Neither 四种选择，由于传输的数据量较小，选择Buffered方式。

11）第十一步选择是否生成一个Win32 Console应用程序，还可以选择Debug代码等一些选项。本设计选择默认，加一个Win32 Console应用程序。点击完成按钮，生成工程项目。在工程项目中会添加两个类：PCIDriver和PCIDriverDevice类。PCIDriver提供设备驱动程序的基本框架，负责驱动程序的初始化，并负责将IRP分发到目标设备对象，PCIDriverDevice支持即插即用和电源管理，主要处理IRP_MJ_PNP和IRP_MJ_POWER IRP。我们设计驱动程序只需要修改PCIDriverDevice类中的相应函数即可。PCI9052接在工控机上以后，如果PnP管理器检测到相应的硬件设备，就会发送IRP_MN_START_DEVICE的IRP给驱动程序处理。由于PCI总线为PnP总线，PCI设备的硬件资源都是由PCI总线配置机构动态分配的。驱动程序必须先将该IRP交给PCI总线驱动程序处理，取得PCI设备的配置信息和分配的资源后，再分发和处理该IRP。IRP_MN_START_DEVICE在驱动程序中对应的处理函数一般为OnStartDevice。如果在向导中设置好了PCI设备的资源，就不必再处理了，向导已把资源处理完毕。PCI的地址可以映射到工控机的IO空间或Memory空间，可以通过IO方式或Memory方式来访问PCI的配置空间和局部空间，有两个类封装了对其访问的操作：KIORange和KMemoryRange。KIORange类实现了对IO空间映射的访问，KMemoryRange类实现了对内存空间映射的访问，这两个类除了其中的构造函数不同之外，其他成员函数完全相同，主要的成员函数有：初始化函数 Initialize（）；从端口读字节、单字和双字函数 inb（）、inw（）、ind（）；向端口写字节、单字和双字函数outb（）、outw（）、outd（）。EEPROM配置

中，将板卡的局部空间映射到了计算机Memory空间，下面用Memory方式实现对PCI9052局部空间的访问。DriverStudio向导生成PCI驱动程序框架PCIDriverDevice类的OnStartDevice函数中，对PCI资源进行初始化。代码如下：

```
PCM_RESOURCE_LIST pResListRaw = I.AllocatedResources();          原始资源表指针
PCM_RESOURCE_LIST pResListTranslated = I.TranslatedResources();翻译资源表指针
status = m_IoPortRange2.Initialize(
pResListTranslated,
pResListRaw,
PciConfig.BaseAddressIndexToOrdinal(2)
);
if (!NT_SUCCESS(status))
{
Invalidate();
return status;
}
```

初始化完毕后，在PCIDriverDevice成员函数 DeviceControl中进行对端口的访问。DriverStudio向导生成框架中，步骤十加入应用程序和驱动程序通信的控制代码参量，DeviceControl中封装了对应代码：

```
switch (I.IoctlCode())
{
case MEMREAD_IOCTL:
status = MEMREAD_IOCTL_Handler(I);
break;
case MEMWRITE_IOCTL:
status = MEMWRITE_IOCTL_Handler(I);
break;
default:
status = STATUS_INVALID_PARAMETER;
break;
}
```

通过读取从应用程序中传过来的IRP中的控制代码变量，选择与之对应的入口进行处理。MEMREAD_IOCTL和MEMWRITE_IOCTL是定义的两个控制代码变量，分别为对端口的读操作和写操作，处理函数分别为：MEMREAD_IOCTL_Handler（）和

MEMWRITE_IOCTL_Handler（），我们用这两个函数实现对PCI局部空间进行双字（4个字节）的读写，下边是两个函数的实现代码。读双字函数MEMREAD_IOCTL_Handler（）实现：

```
PDWORD pInBuffer = (PDWORD)I.IoctlBuffer();  输入缓冲区指针，传来应用程序的参数读取的地
                                             址和数据个数
PDWORD pOutBuffer = (PDWORD)I.IoctlBuffer(); 输缓冲区指针
*pOutBuffer = m_MemoryRange2.ind(*pInBuffer); 执行读操作
I.Information() = 4;返回接收到的字节数
```

写双字函数 MEMWRITE_IOCTL_Handler（）实现：

```
PDWORD pInBuffer = (PDWORD) I.IoctlBuffer();  输入缓冲区指针,
DWORD offset;                                 从输入缓冲区获得的要写入的数据偏移地址
offset=*pInBuffer;
data=*(pInBuffer+1);                          从输入缓冲区获得的要写入的数据
m_MemoryRange2.outd(offset,data);             执行写操作
I.Information() = 4;                          返回接收到的字节数
```

5.4 软件开放式功能的实现

5.4.1 动态链接库（DLL）

a 动态链接函数库基本概念

DLL 是动态链接函数库（Dynamic Link Libraries）的简称，是组成 Windows 系统的最重要的元素。Windows 将构成其系统的大部分程序代码、数据以及经常用到的资源以二进制文件（类似于可执行文件）的形式，存储在磁盘里。DLL 就是允许用户的应用程序共享代码和资源的可执行模块，这种分享能力存在于应用程序模块之间或不同的应用程序之间，同一个动态链接库里的同一个函数可同时被不同应用程序调用。Windows 系统核心的绝大部分代码、数据、资源及大量的设备驱动程序等均以动态链接库的形式存在，DLL 实质上是 Windows 环境下的一个特殊的可执行模块，它内含一系列的函数或资源，供一般的应用程序或动态链接库调用。它有不同的文件后缀，如：.EXE，.DLL，DRV，.SYS 和.FON 等。DLL 和 Windows。应用程序最大的区别是它不能作为单独任务执行，而应由 Windows 应用程序调用。简单的说，DLL 是 Windows 和 Windows 应用程序可以使用的资源库，这些资源是共享的。

b 动态链接库与静态链接库的区别

DLL 节约内存、减少交换文件、节约磁盘空间、容易升级、提供售后服务支持及提供扩展MFC类库的机制、支持多种语言程序设计和容易创建国标化版本。DOS下的LINK程序主要任务是把源程序中所使用的库函数代码从库中提取出来加入到链接生成的可执

行文件中，该文件包含了运行时所需要的全部代码，这种链接方式称为静态链接。而在Windows这样的多任务环境下，应用系统共享内存资源。如果也采用静态链接方式，当多个程序都调用相同的函数时，每个应用程序的链接都要将函数拷贝给应用程序，结果应用程序运行时，在内存生成同一函数的多个拷贝，浪费了宝贵的内存资源。而动态链接是所调用的函数代码并没有被链接程序拷贝到应用程序的可执行文件中去，而是仅仅在其中加入了所调用函数的描述信息（往往是一些定位信息）。仅当应用程序被装入内存开始运行时，在Windows的管理下，才在应用程序与相应的DLL之间建立链接关系。当要执行所调用的DLL中的函数时，根据链接时产生的重定位信息，Windows转去执行DLL中相应的函数代码，而不是简单的复制或提取。

c 使用DLL的优势

使用DLL，特别是在大应用程序中使用DLL，有3个明显的优点，这些优点是静态链接库所没有的。一是明显减少应用程序的磁盘空间；二是减少应用程序所需的内存；三是修改应用程序更容易，四是这种函数库可以被Windows下的多种编程语言调用，使系统具有更强的适应性和柔性。

5.4.2 动态链接库的设计

下面重点介绍Windows下动态链接库的设计方法，实践证明动态链接库的正确创建是开发的关键。采用VC++集成开发环境，通过以下步骤完成动态链接库的创建：

1）建立动态链接库C/C++源文件（实现部分）动态链接库源文件包含两个基本部分：LibMain函数，它是Windows的主入口，负责初始化DLL，当DLL第一次装入时，Windows将调用它。同时它负责终止结束等控制，从而使用户能够根据应用程序的需要分配和释放内存，无需象16位那样由WEP()函数来完成清理工作。

2）用户定义输出的库函数和仅供内部调用的辅助函数。

3）建立动态链接库资源描述文件（可选）。

4）建立动态链接库模块定义文件（DEF）模块定义文件向LINK提供定义库属性的输入信息。它包括DLL标志、DLL信息和版权声明、库代码段属性、库数据段属性、局部堆属性、导出函数等。

5）动态链编生成动态链接库。

5.4.3 用动态链接库实现开放式功能

硬件设备的控制，实质上也就是从外设上读取设备状态，把控制信息传送到外部设备。在微机和系统控制接口插卡之间实现数据传送和交换的过程，这其中包括很多复杂的对底层控制电路的操作及读写，对于一个专业的接口开发人员来说，这项工作可能不算太难，但对于一个用户来说，这些操作将会非常复杂和繁琐，而且容易出错。在Windows中，动态链接库中的程序段可常驻内存，而且和多种Windows编程语言可方便实现接口，供其调用，这种结构还具有很多其它程序不具有的特点。为此，我们将所有的控制按照

功能编制成动态链接库，供用户二次开发或调整系统功能时直接调用，大大的方便了用户，实现了开放式功能。为了使用方便,我们将完成每个具有独立控制功能的硬件操作编制成一个库函数,使系统具有很强的柔性。每个库函数主要包括两部分：源代码编写和输出声明。源代码主要是实现具体的功能代码，而输出声明则通过在模块定义文件中使用 EXPORT 宏来声明该函数，是编译器将其编译为可被调用的动态函数。以下是我们编制的对一轴伺服电机控制的初始化动态链接库。

源代码文件 CONTORLX.CPP 及模块定义文件 CONTORLX.DEF 如下:

```
源代码文件: CONTORLX.CPP
#include "stdafx.h"
#include "contorlx.h"
#include "conio.h"
#include "math.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif// Note!部分省略
void _declspec(dllexport) Motivesystem()
{ _outp(0x26B,0x80);//电机控制输出
_outp(0x26a,0x0);//全部置零，禁止电机转动}
CONTORLX.DEF 中声明如下：
LIBRARY              "contorlx"
DESCRIPTION 'contorlx Windows Dynamic Link Library'
EXPORTS
Explicit exports can go here
DllCanUnloadNow PRIVATE
DllGetClassObject PRIVATE
DllRegisterServer PRIVATE
Motivesystem
```

链接 DLL 到应用程序中主要有两种方式：隐式链接和显式链接。在此只介绍隐式链接。通过隐式链接，使用 DLL 的可执行程序链接到 DLL 导入库（.LIB）中。当装载使用 DLL 的可执行程序时，操作系统装载 DLL。隐式链接 DLL 时，可执行程序必须从 DLL 提供者获取以下内容：

包含导出函数或 C++类声明的头文件(.H)；导入库文件(.LIB)；实际的 DLL(.DLL 文件)；在可执行程序的每个使用导出函数的源文件中必须用#include 语句并包含有导出函数（或者 C++）的头文件。从编程角度来讲，调用导出函数和调用其它函数完全一样。建立可执行程序时，必须与导入库链接。如果在 Developer Studio 中，则必须在“Project Setting”对话框的“Link”选项卡的“Object/Library Modules”文本框中指定导入库的名字。

5.5 上下位机通讯模块

5.5.1 上下位机通讯技术

上下位机控制系统设计的关键是上位机和下位机之间的数据通讯，因此，为了保证主控计算机与多轴运动控制器之间有效，可靠的进行通讯,在控制器上外扩一块双端口,专门用于通讯,起缓冲作用，称其为通讯缓冲区，在这个缓冲区的内部，划分了个子块，分别用作函数，实时指令，系统状态,系统参数和资料缓冲区，在通讯过程中，主机将信息送入通讯缓冲区，而则从该缓冲区中取出被传递的信息，或者相反，由写入信息，而由读出信息，其中函数缓冲区和资料缓冲区用来传递与数控机床或机器人有关的函数参数和用作特性和动态显示的资料实时指令，系统状态，系统参数缓冲区则分别用来传递用户的实时控制指令，系统当前工作状态以及系统当前的结构和控制参数等信息。在通讯过程中对缓冲区的读写操作必须严格按所规定的数据结构和通讯协议来进行。

5.5.2 上位机与下位机的通讯

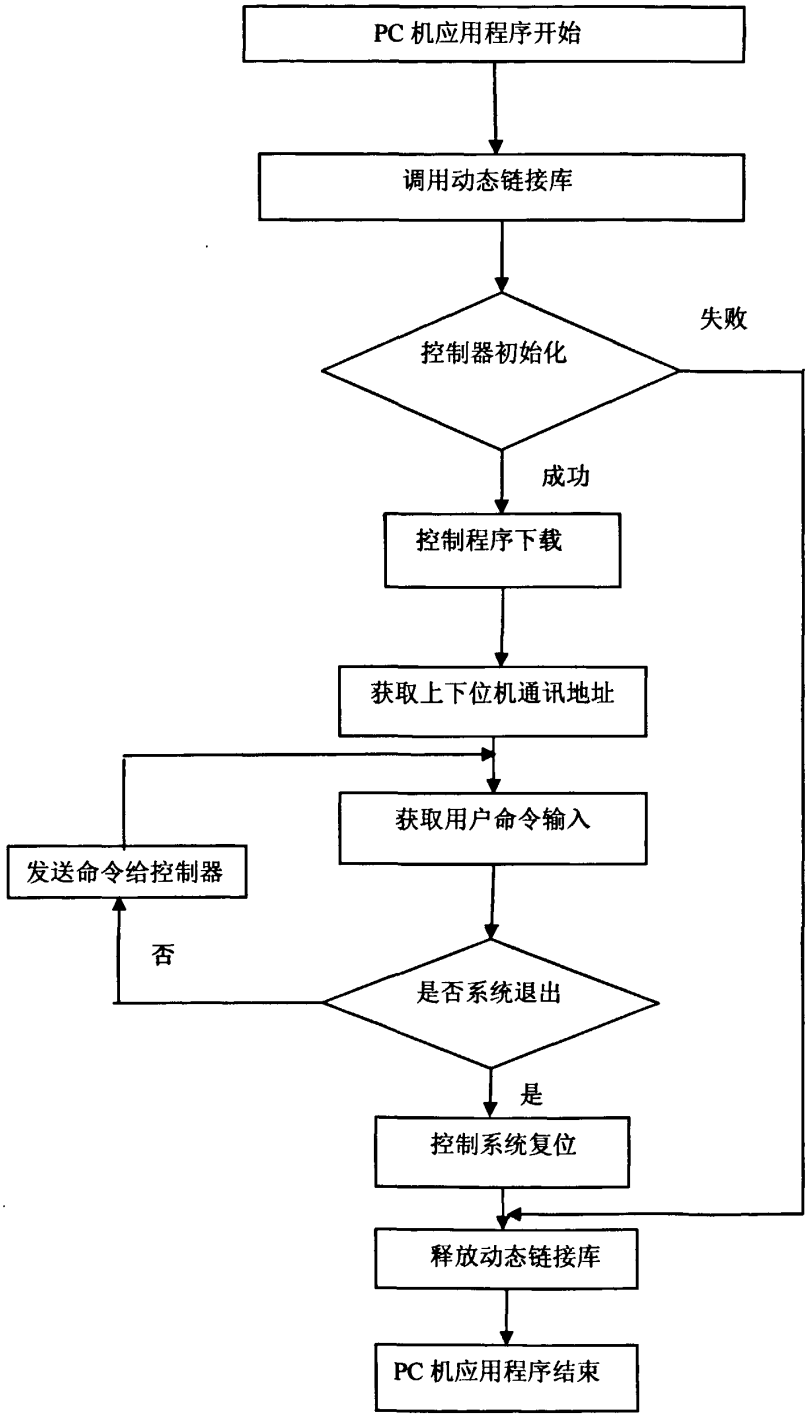


图 5-3 上位机通讯流程图

Figure5-3 The chart of PC communication flow

上位机向下位机发送指令和地址，并按要求与下位机进行通讯，同时要求下位机发送数据，进行数据处理和管理而下位机中断服务程序完成的任务是接受上位机发送的地址和指令，同时向上位机传送数据，当数据传送完后发送结束信号。如图 5-3，图 5-4。

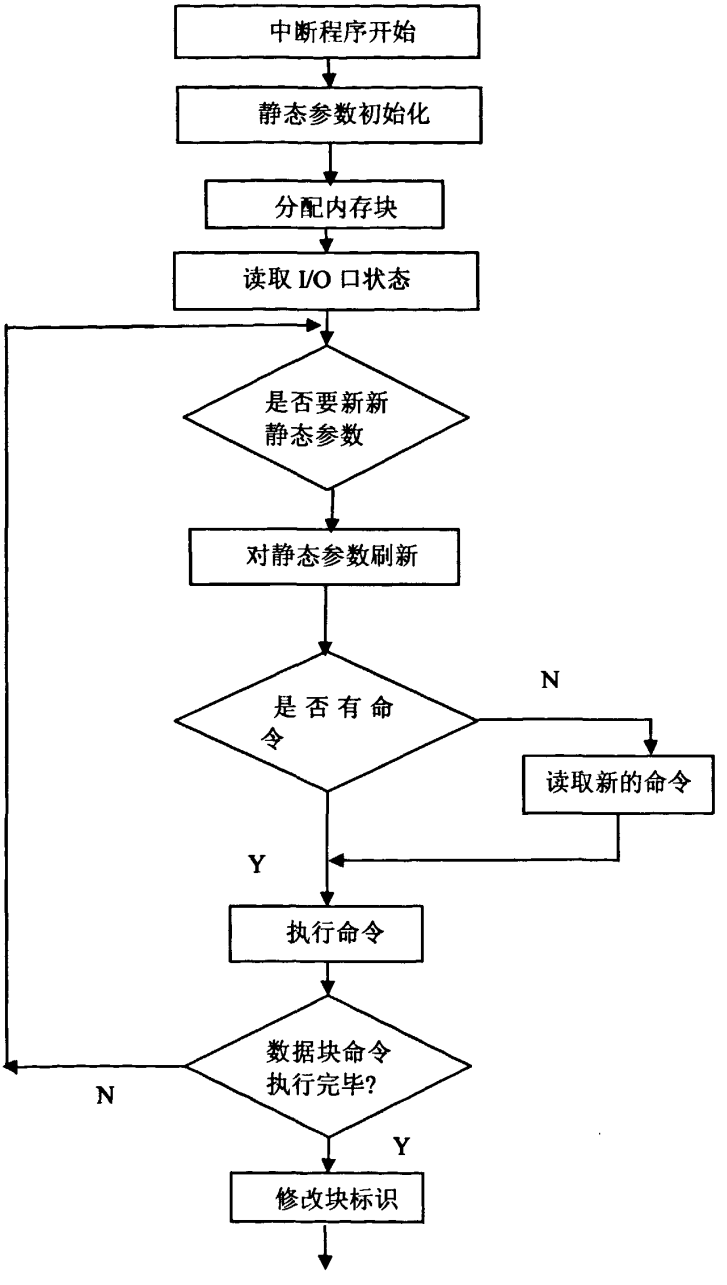


图 5-4 下位机通讯流程图

Figure5-4 Flow chart of digital communication

5.6 在系统可编程逻辑器件

5.6.1 可编程逻辑器件 ISP

ISP(In-system Programming在系统可编程)是lattice公司提出的能在产品设计，制造过程中的每个环节，甚至在产品卖给最终用户以后，具有对其器件，电路板或整个电子系统的逻辑和功能随时进行组态或重组能力的最新技术。它允许重组系统特性，而器件仍保持焊接在电路板上这个能力革新了板级设计，板级调试，系统制造和系统升级。

5.6.2 硬件描述语言 VHDL

VHDL 的英文全名是 Very-High-Speed Integrated Circuit Hardware Description Language

诞生于 1982 年主要用于描述数字系统的结构，行为，功能和接口除了含有许多具有硬件特征的语句外，VHDL 语言形式和描述风格与句法十分类似于一般的计算机高级语言应用 VHDL 进行工程设计的优点是多方面的，具体如下：

1）丰富的仿真语言和库函数，使得在任何大系统的设计早期即尚未完成，就能查验设计系统的功能可行性，随时可对设计进行仿真模拟即在远离门级的高层次上进行模拟，设设计这对整个工程设计的结构和功能的可行性做出决策。

2）VHDL 语言的行为描述能力和程序结构决定了它具有支持大规模设计的分解和已有设计的再利用功能。符合市场需求的大规模系统高效，高速的完成必须有多人甚至多个开发组共同并行工作才能实现。

3）VHDL 中设计实体的概念。程序库的概念为设计的分解和并行工作提供了有力的支持。VHDL 对于用完成的一个确定的设计，可以利用 EDA 工具进行逻辑综合和优化，并自动地把描述设计转变城门级网表根据不同的实现芯片这种方式突破了门级设计的瓶颈，极大的减少了电路设计的时间和可能发生的错误，降低了开发成本，应用工具的逻辑优化功能，可以自动地把一个综合后的设计变成一个更小，更高速的电路系统，反过来，设计者还可以容易的从综合和优化后的电路获得设计信息，返回去更新修改设计描述，使之更为完善。目前流行的一些支持的工具软件有 Altera 公司的 MUX+PLUS II 软件和 Xilinx 公司的 Foundation Series 集成开发的 EDA 工具 VHDL 程序就是这些平台上编辑，编译，综合，仿真，适配，配置，编程下载和硬件调试等。

本课题根据 TMS320LF4207 的读写时序图编写相应的 VHDL 逻辑程序，完成 DSP 和 ISP 的通讯，利用 MUX+PLUS II 软件进行文本设计，并进行编译和综合，程序主要说明 TMS320LF4207 和 ISP 之间通讯时锁存译码部分：adder 指 DSP16 位外部地址总线,is-dsp 指 DSP 的 I\O 空间选择信号，rd 指 DSP 读信号，wt 指 DSP 写信号，strb 指 DSP 的外部访问有效选通信号，oen 指锁存器使能信号，程序如下：

```
PROCESS(adder,is_dsp,rd)
```

```
    BEGIN
IF((adder= " 1111000011110000 " )AND(is_dsp= '0' )AND (rd= '0' ) )THEN
    data(7 DOWNTO 0)<=latch_in1;
    data(15 DOWNTO 8)<=latch_in2;
ELSE data(7 DOWNTO 0)<= " ZZZZZZZZ " ;
      Data(15 DOWNTO 8 )<= " ZZZZZZZZ "  ;
  END IF;
END PROCESS;
PROOCESS(oen1)
BEGIN
  IF(oen1EVENT AND oen1= '1' )THEN
      latch_in1<=dbin1;
  END IF
END PROCESS;
PROCESS(oen2)
BEGIN
  IF(oen2EVENT AND oen2= '1' )THEN
      latch_in2<=dbin2;
  END IF;
END PROCESS;
PROCESS(adder,is_dsp,rd)
BEGIN
  IF((adder= " 1111000011111000 " )AND(is_dsp= '0' )AND(rd= '0' ))THEN
      Data(7 DOWNTO 0)<=latch_in3;
  ELSE data(7 DOWNTO 0)<= " ZZZZZZZZ " ;
  END IF;
END PROCESS;
PROCESS(oen3)
BEGIN
  IF (oen3EVENT AND oen3= '1' ) THEN
      latch_in3<=dbin3;
  END IF;
END PROCESS;
```

```
PROCESS(adder,is_dsp,rd,ena)
BEGIN
  IF((adder= " 1111000011111010 " )AND(is_dsp= '0' )AND(rd= '0' ))THEN
     data(0)<=latch_in4;
  ELSE data(0)<= 'Z' ;
  END IF;
END PROCESS;
PROCESS(ena)
BEGIN
IF(enaEVENT AND ena= '1' )THEN
  latch_in4<=z0_mux411;
END IF;
END PROCESS;
PROCESS(adder,is_dsp)
BEGIN
  IF((adder= " 1111001011110000 " )AND(is_dsp= '0' ))THEN
     wr_enable1<=wr;
ELSE wr_enable1<= '1'
  END IF;
END PROCESS;
PROCESS(wr_enable1)
BEGIN
  IF((wr_enable= '0' )AND(strb= '0' ))THEN
     latch_out1<=data;
  END IF;
END PROCESS;
din1(15 DOWNTO 0)<=latch_out1;
PROCESS(adder,is_dsp)
BEGIN
  IF((adder= " 1111001011111000 " )AND(is_dsp= '0' ))THEN
     wr_enable2<=wr;
  ELSE wr_enable2<= '1' ;
  END IF;
```

```
END PROCESS;
PROCESS(wr_enable2)
BEGIN
  IF((wr_enable2=0)AND(strb=0))THEN
     latch_out2<=data(7 DOWNTO 0);
  END IF;
END PROCESS;
din1(23 DOWNTO 16)<=latch_out2;
PROCESS(adder,is_dsp)
BEHIN
  IF((adder= " 1111001111110000 " )AND(is_dsp= '0' ))THEN
     wr_enable3<=wr;
ELSE wr_enable3<= '1' ;
END IF;
END PROCESS,
PROCESS(wr_enable3)
BEGIN
  IF((wr_enable3= '0' )AND(strb= '0' ))THEN
     latch_out3<=data;
END IF;
END PROCESS,
din2(15 DOWNTO 0)<=latch_out3;
PROCESS(adder,is_dsp)
BEGIN
  IF((adder= " 1111001111111000 " )AND(is_dsp= '0' ))THEN
     wr_enable4<=wr;
ELSEwr_enable4<=1;
END IF;
END PROCESS
  PROCESS(wr_enable4)
  BEGIN
IF((wr_enable4= '0' )AND(strb= '0' ))THEN
latch_out4<=data(7 DOWNTO 0);
```

```
END IF;
  END PROCESS;
  din2(23 DOWNTO 16)<=latch_out4;
  PROCESS(adder,is_dsp)
  BEGIN
IF((adder= " 1111010011110000 " )AND(is_dsp= '0' ))THEN
   wr_enable5<=wr;
ELSE wr_enable5<= '1' ;
END IF;
END PROCESS;
  PROCESS(wr_enable5)
  BEGIN
IF((wr_enable5= '0' )AND(strb= '0' ))THEN
   latch_out5<=data;
END IF;
END PROCESS;
din3(15 DOWNTO 0)<=latch_out5;
PROCESS(adder,is_dsp)
BEGIN
IF((adder= " 1111010011111000 " )AND(is_dsp= '0' ))THEN
wr_enable6<=wr;
ELSE wr_enable6<= '1' ;
END IF;
END PROCESS;
PROCESS(wr_enable6)
BEGIN
   IF((wr_enable6= '0' )AND(strb= '0' ))THEN
      latch_out6<=data(7 DOWNTO 0));
   END IF
END PROCESS;
din3(23 DOWNTO 16)<=latch_out2;
PROCESS(adder,is_dsp)
BEGIN
```

```
IF((adder= " 1111010111110000 " )AND(is_dsp= '0' ))THEN
        wr_enable7<=wr;
ELSE wr_enable7<= '1' ;
END IF;
END PROCESS;
PROCESS(wr_enable7)
BEGIN
  IF((wr_enable7= '0' )AND(strb= '0' ))THEN
     latch_out7<=data;
  END IF;
END PROCESS;
din4(15 DOWNTO 0)<=latch_out7;
PROCESS(adder,is_dsp)
BEGIN
  IF((adder= " 1111010111111000 " )AND(is_dsp= '0' ))THEN
     wr_enable8<=wr;
  ELSE wr_enable8<= '1' ;
  END IF;
  END PROCESS;
PROCESS(wr_enable8)
BEGIN
IF((wr_enable8= '0' )AND(strb= '0' ))THEN
      latch_out8<=data(7 DOWNTO 0);
END IF;
 END PROCESS;
din4(23 DOWNTO 16)<=latch_out8;
```

经过以上处理后，DSP 取出位置反馈信号经倍频、辩向和计数的值，即位置实际值，即 DSP 与 ISP 进行通讯。

5.7 本章小结

本文以这一世界前沿方向为指导，在对 DSP 的运动控制特性及深入研究运动控制关键技术的基础上，设计了一种通用型运动控制卡及实现方法，主要包括以下内容：

1）编写了 WDM 驱动程序，实现了 PCI 板卡与工控机的通讯。

2）利用 C++语言进行程序设计，采用了流行的模块化设计方法和面向对象技术，

利用了 WINDOWS 操作平台下 DLL 动态链接库的核心技术，设计了相对独立、可操作性强的控制模块，通过动态链接库的动态链接，实现了开放式的功能。

3）利用 VHDL 语言实现 ISP 与 DSP 的通讯，程序主要说明 DSP 和 ISP 之间通讯时锁存译码部分。

基于 DSP 的运动控制卡的开发成功，由于运动控制卡的控制细节由 DSP 芯片来完成，无需占用 PC 机的资源，大大提高了系统的效率，同时也提高了控制卡的实时性，它实际上为其他机电一体化系统中运动控制的数字化解决方案提供了软、硬件开发平台。

6 机器人控制系统的调试

6.1 机器人调试要点

对机器人调试阶段时，我们把整个机器人先单独调试，在此基础上，逐渐增加调试的关节数。本章首先对机器人单关节调试，然后 2 个关节联合调试，最后对控制系统性能和控制余量作进一步的探讨。

机器人单关节控制在性能上有两个方面的要求：一是要有稳定平滑的瞬态响应，二是稳态位置跟踪误差和动态位置跟踪误差要小，以获得高精度的位置控制性能。本机器人单关节闭环控制系统如图 6-1 所示。

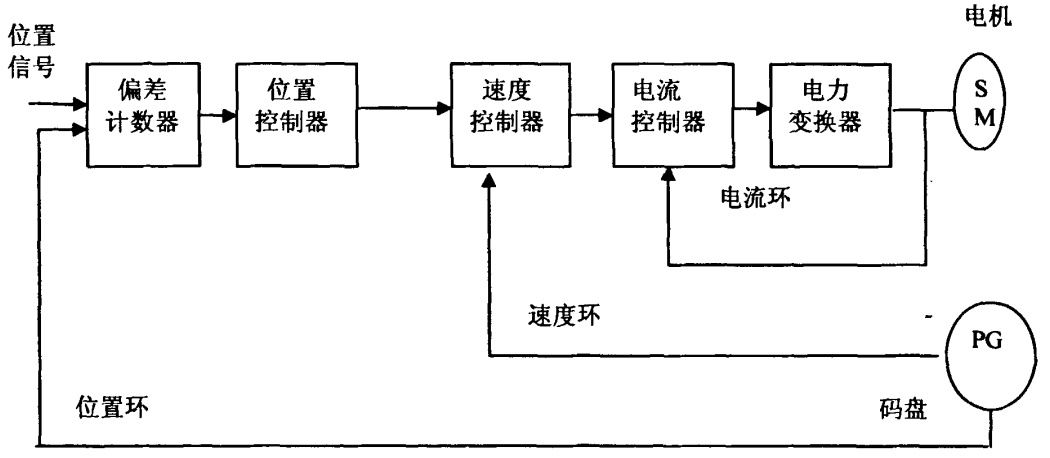


图 6-1 伺服电机位置闭环控制系统

Figure6-1 The location of the closed-loop servo motor control system

由三个环组成，最里面是电流环，中间是速度环，最外边是位置环，电流环和速度环在伺服驱动器中实现，位置环在 DSP 运动控制卡中实现。电流环的作用是改造内环控制对象的传递函数，提高系统的快速性，及时抑制电流环内部的干扰，限制最大电流，又使系统有足够大的加速转矩。电流环在伺服驱动器中实现，无参数需要修改。速度环的主要作用是增强系统负载抗扰动能力，抑制速度波动。速度环在伺服驱动器中实现，为了使伺服电机平滑运行和高速定位。除了可以调节速度环增益和速度环积分增益，伺服驱动器需要对速度环的参数进行调整来改善速度环的性能：模式开关、速度反馈补偿、增益切换、自动调谐、零钳位等。下面对这几种手段分别加以说明，我们按照需要使用其中的一种或多种功能，以缩短调整时间，提高控制性能。速度环增益调节在位置环增益调节中统一讨论。

6.1.1 模式开关

模式开关在速度控制中的作用是降低加、减速的超调。在伺服驱动器中，通过模式开关自动切换控制方式，设定自动切换的速度阀值，当前速度小于速度阀值时，采用 PI 调节，当前速度超过速度阀值时，采用 P 控制，如图 6-2 所示，Pn10D 表示速度阀值。

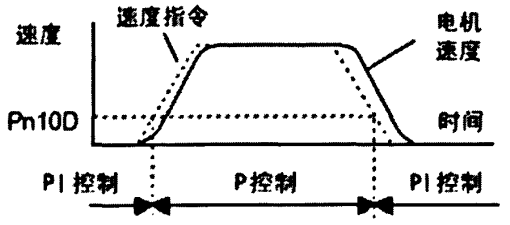


图 6-2 模式控制方式

Figure6-2 Control mode

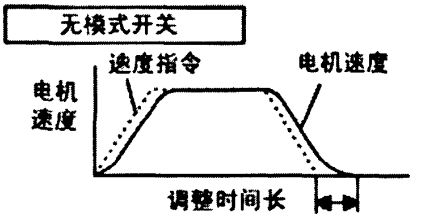


图 6-3 无模式开关下的速度控制

Figure6-3 No mode switch speed control

6.1.2 增益切换

在速度调节的不同时刻，可能要选用不同的增益，增益切换功能用于在线情况下自动调整速度环的增益。伺服驱动器内部可以设定速度环第一增益和第二增益，在增益选择功能有效的情况下，通过改变伺服驱动器增益切换数字量输入信号，可以自动在第一增益和第二增益之间来回切换。

6.1.3 自动调谐

伺服驱动器内部配备自动测量功能，对负载惯性动量的变动进行测量，使目标的速度环增益保持一定的控制功能。通过设定自动调谐功能，可以非常方便地完成各种增益的自动调节。伺服驱动器默认设置为伺服打开时进行一次自动调谐，运行过程中不再进行自动调谐，如果负载运行过程中惯性动量变动较大时，应该设置为常时自动调谐，即运行过程中不断地进行自动调谐。将“零钳位”功能置于“ON”后，即使速度指令的输入电压不为“0V”，也要使电机停止，使伺服处于锁定状态。“零位”功能是在伺服驱动器内部临时配置位置环，电机在该位置进行±1 个脉冲范围以内的钳位，即使在外力作用下转动，也会返回零钳位置。“零钳位”在本控制系统中的作用是当机器人到达了目标位置后，使用“零钳位”防止机器人偏离目标位置。

下面对位置闭环控制系统进行增益调整，首先介绍增益调整的基本原则和方法，然后给出了本系统实际调节时的调试结果。闭环控制系统有三个环（电流环、速度环和位置环），越是内侧的环，越要提高其响应性，如果不遵守这个这个原则，系统响应性将变差或产生震动。我们要遵守：电流环在伺服系统中已经确保了充分的响应性。速度环的响应性要高于位置环的响应性，表现在提高位置环增益时，首先要提高速度环增益，否则可能因为速度环响应性跟不上，而引起速度震动，反而延长调节时间。

不能将位置环增益提高到超出机械系统固有震动频率的范围。本系统为 6 关节机器

人，整体结构的刚性极低，固有震动频率只有 10-20Hz 左右，位置环增益也不能设得很高。

通过提高速度环和位置环的增益，可以提高系统的响应性，当响应性还不够时，可以使用前面提到的伺服驱动器的几种功能，也能起到改善系统响应性的作哦用，如模式开关、前馈控制、自动调谐等。但是这几种方法也不是万能的，要注意调整，否则可能出现相反的效果。

下面给出调整速度环增益和位置环增益的方法：

1）先将位置环增益设定一个比较小的值，即设定 Kp 较小，增大速度环增益，直到临界。

2）增大位置环增益，即增大 Kp，重复第 1 步，增大速度环增益，直到临界。

3）在 1，2 步调节的同时，设定速度环积分时间常数，过大会延长定位调整时间。

4）机械没有发生震动，尽量减小扭矩指令滤波器的值，发生震动，将该值调大，以减小震动。

5）选择合适的一组增益值，然后进行微调，以达到最佳效果。

6.2 单关节测试程序调试

工控机中用 VC++6.0 设计了单关节测试程序，对机器人的单关节进行调试，测试程序的界面如图 6-4 所示，DSP 中也编写了对应的控制程序。该测试程序可以进行速度控制和位置控制，界面中间可以显示单关节运动的速度和位置曲线，左边用于修改 DSP 中的控制参数，右边用于显示电机的转速、位置等运行数据，下边是对速度和位置坐标系进行翻页、放大和缩小等操作。应用这个测试程序对机器人单关节进行电机闭环调试，调整速度环增益和位置环增益参数，提高单关节的位置控制能力。

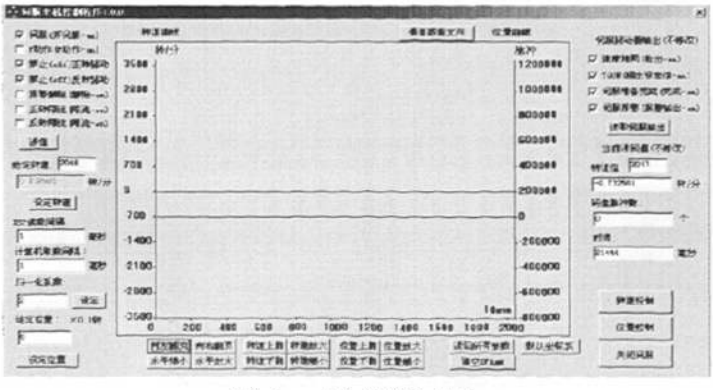


图 6-4 测试程序的界面

Figure6-4 Test procedure interface

图 6-5 中，归一化参数 Kp 为 0.05；转速上限为 3000 转/分，积分阀值为 10 圈，即当伺服电机运动到和给定位置相差 10 圈时，由 PD 控制转为 PID 控制，死区阀值为 20 个码盘脉冲，即当误差小于 20 个码盘脉冲时，不进行 PID 控制，速度给定为 0。伺服驱动器中，转速环增益为 40。

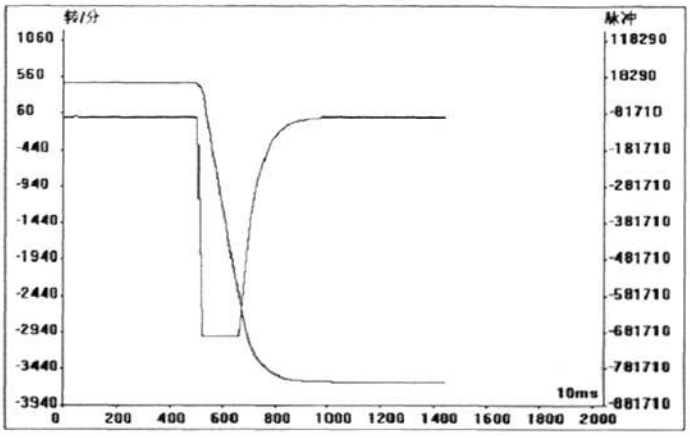


图 6-5 机器人单关节调试结果

Figure6-5 One single robot joint commissioning results

6.3 双关节测试

工控机中用 VC++6.0 设计了双关节测试程序对机器人双关节进行调试，测试程序界面如图 6-6 所示，

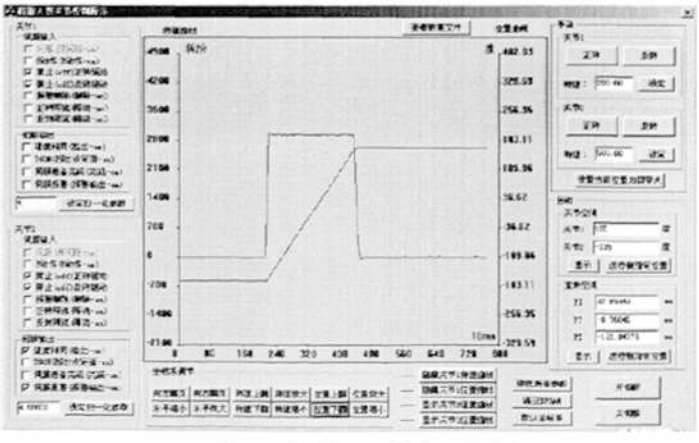


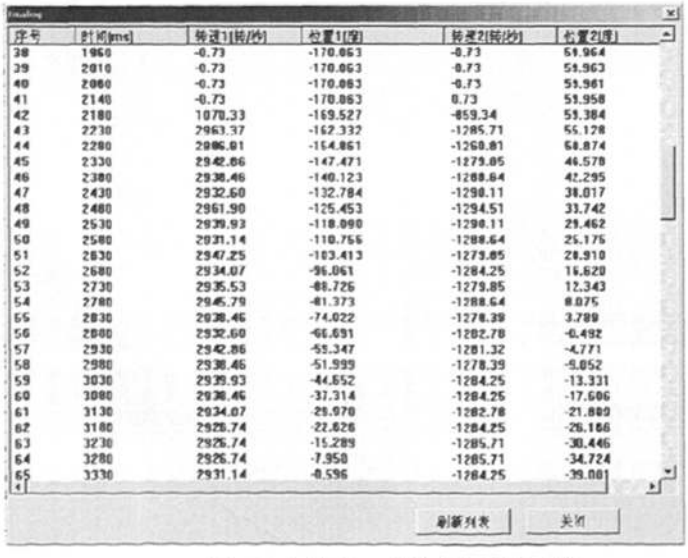
图 6-6 机器人双关节调试界面

Figure6-6 Two-joint robot debug interface

TMS320LF4207 中编写了对应的控制程序。该程序实现了机器人的点动控制、关节空间控制和直角坐标空间控制等功能。界面中间可以显示机器人双关节运动的速度和位置曲线，左边是伺服驱动器的数字量输入控制和双关节 PID 归一化参数设定，右边是双

关节的点动控制和位置控制，下面是对双关节的速度和位置坐标系进行翻页、放大和缩小的操作，顶部的查看数据文件按钮可以查看机器人运行时的速度和位置数据，如图 6-7 所示，这些数据就是绘制速度和位置曲线用到的数据。由于机器人上未安装机械零点，在程序中点动控制关节运行到参考零点，然后按下设置当前位置为回零点按钮，即将此位置设为参考零点位置。

机器人双关节调试结果 1 和调试结果 2 都是基本上同时开始运动和同时停止运动，不仅在视觉上运动比较协调，而且可以达到运动时间最小。在本位置闭环控制系统中，运动的加速时刻和减速时刻所占的时间较短，加速到最大速度 3000 转/分的时间大约为 80ms，可将关节运动近似成以最高转速运行的匀速运动，将两个关节中运动距离较长的关节最高转速设为 3000 转/分，运动距离较短的关节按运动距离的比例设定最高转速，基本上可以实现了双关节同时启动和同时停止。看出由于此时底座负载较小，速度和位置都不会产生超调现象，在双关节调试中，在底座关节的基础上又安装了一个大臂关节，底座负载提高了，而且重心出现了偏移，由曲线可以看出，随着负载惯性的提高，对底座关节同样的控制方法出现了不同的调试效果，速度和位置出现了超调现象。由此可以预见，如果整个机器人安装完成，底座负载的惯性将更加巨大，而且底座上面的负载是变动的（关节是活动的），负载的重心随着关节的运动不断的变化，单一的 PID 系数调节可能达不到预期的效果，可以考虑采用更加优化的控制算法，如模糊控制、参数自整定 PID 控制、神经网络控制等。



序号	时间[ms]	转速1[转/秒]	位置1[度]	转速2[转/秒]	位置2[度]
38	1960	-0.73	-170.063	-0.73	59.964
39	2010	-0.73	-170.063	-0.73	59.963
40	2060	-0.73	-170.063	-0.73	59.961
41	2110	-0.73	-170.063	0.73	59.958
42	2160	1070.33	-169.527	-859.34	59.384
43	2210	2963.37	-162.332	-1285.71	55.128
44	2260	2986.81	-154.861	-1260.81	50.874
45	2310	2942.86	-147.471	-1279.85	46.578
46	2360	2938.46	-140.123	-1288.64	42.295
47	2410	2932.60	-132.784	-1290.11	38.017
48	2460	2961.90	-125.453	-1294.51	33.742
49	2510	2939.93	-118.090	-1290.11	29.462
50	2560	2931.14	-110.756	-1288.64	25.175
51	2610	2947.25	-103.413	-1279.85	20.910
52	2660	2934.07	-96.061	-1284.25	16.620
53	2710	2935.53	-88.726	-1279.85	12.343
54	2760	2945.79	-81.373	-1288.64	8.075
55	2810	2938.46	-74.022	-1278.39	3.789
56	2860	2932.60	-66.691	-1282.78	-0.492
57	2910	2942.86	-59.347	-1281.32	-4.771
58	2960	2938.46	-51.999	-1278.39	-9.052
59	3010	2939.93	-44.652	-1284.25	-13.331
60	3060	2938.46	-37.314	-1284.25	-17.606
61	3110	2934.07	-29.970	-1282.78	-21.880
62	3160	2926.74	-22.626	-1284.25	-26.166
63	3210	2926.74	-15.289	-1285.71	-30.446
64	3260	2926.74	-7.950	-1285.71	-34.724
65	3310	2931.14	-0.596	-1284.25	-39.00

刷新列表　　关闭

图 6-7 机器人双关节运行数据列表

Figure6-7 Two-joint robot operating data list

6.4 本章小结

在机器人硬件平台和软件平台的基础上，我们开发了机器人单关节调试程序和双关节调试程序，对机器人的单关节和双关节进行了调试，给出了调试的结果。

7 全文总结与展望

7.1 全文总结

本课题以六关节机器人作为研究对象，对机器人运动控制系统的有关理论问题做了较深入的研究，主要探讨了六关节机器人正解、逆解的算法和伺服闭环控制系统的算法。在此理论基础上，设计了六关节机器人伺服控制系统，搭建了机器人控制系统的硬件平台，为以后实现机器人的全面控制提供了一个完整的平台。围绕六关节机器人设计的要求，本课题进行了以下方面的研究：

1）设计六关节机器人运动控制系统，实现了 IPC 和 DSP 的两级伺服控制方案。

2）对机器人运动学进行了分析和对伺服系统做了阐述。

3）提出了机器人硬件配置和对几大模块进行了详尽的说明。

4）提出了本控制系统软件设计思想。利用 C++语言进行程序设计，采用了流行的模块化设计方法和面向对象技术，利用了 WINDOWS 操作平台下 DLL 动态链接库的核心技术，设计了相对独立、可操作性强的控制模块，通过动态链接库的动态链接，实现了开放式的功能。

5）完成了机器人单关节和双关节的位置闭环控制，并达到了较好的控制效果。

6）工控机中使用 VC++6.0 编写了机器人单关节控制程序，DSP 中编写了相应的控制程序，完成了对机器人单关节调试，在此基础上，编写完成了机器人双关节控制程序，实现了机器人双关节的点动控制和位置协调控制。

7.2 展望

经过大量的实验，证明本文所提出的理论正确，设计的机器人结构合理，研究开发的各关键控制模块运行安全可靠，实现了预期的功能。机器人控制系统开放化、PC 化是世界机器人控制发展的大方向。基于 DSP 的运动控制卡的开发成功，由于运动控制卡的控制细节由 DSP 芯片来完成，无需占用 PC 机的资源，大大提高了系统的效率，同时也提高了控制卡的实时性，它实际上为其他机电一体化系统中运动控制的数字化解决方案提供了软、硬件开发平台。其涉及的知识结构复杂，知识面宽，由于研究时间有限，本文的研究还有待于进一步深化完善，尤其在机器人动态特性和仿真方面还有待于进一步研究。希望后来的机器人爱好者和研究者能够取得更辉煌的成果，特别希望他们充分利用现代化数据库技术及网络技术，现代化的芯片技术，在运动控制中发挥出 DSP 的潜力。

21 世纪将是计算机技术，通讯技术，光，机，电一体化全球化制造模式等高薪技术发展的新纪元高速，高效，高精度，高品质是人们追求的主要目标。因此,运动控制器只有不断地和先进的技术相结合，才能实现更高的要求，使我国在运动控制技术方面达到世界先进水平。

本文对所述工作内容及方法做了详细叙述，由于时间仓促，个人水平所限，文中免不了有错误疏漏之处，敬请原谅！

参考文献

[1] 罗永新. 基于 PC 工控机的机器人运动控制研究[D]南京：东南大学.2005

[2] 蔡自兴著 机器人学[M]北京：清华大学出版社. 2000

[3] 孙迪生，王炎著. 机器人控制技术[M]北京：机械工业出版社 1997

[4] 陶永华，尹怡欣，葛芦生. 新型 PID 控制及其应用[M].北京：机械工业出版社 1998

[5] 王耀南著. 机器人智能控制工程[M]北京：科学出版社. 2004

[6] 高晓辉，蔡鹤皋.光电码盘细分电路及接口设计[J]传感器世界 1997，16(6):37-39

[7] 付京孙 机器人学[M]中国科学技术出版社，1992：1～10

[8] 周伯英.工业机器人设计[M]机械工业出版社，1995：60～80

[9] 吴广玉，姜复兴 机器人工程导论[M]哈尔滨工业大学出版社，1990：2～12

[10] 蔡自兴 机器人学[M]清华大学出版社.2000： 2～4、46～82、265～280

[11] 范印越 机器人技术[M]电子工业出版社，1988：2～45

[12] 合田周平等 机器人技术[M]科学出版社，1983：10～12

[13] 求是科技.Visual C++ 6.0 程序设计与开发技术大全[M]北京：人民邮电出版社.2004

[14] 陈国辉，郑学仁. 基于 PCI 总线的 IP 仿真验证平台的 WDM 驱动程序设计[J]计算机工程与应用 2005

[14] 彭启琮，管庆等.DSP 集成开发环境－CCS 及 DSP/BIOS 的原理和应用[M]北京：电子工业出版社.2005

[15] TMS320LF24X DSP Reference Set[Z],Volume2:Mnemonic Instruction Set.Texas Instrument Inc,1996：11～15

[16] 翰安太，刘峙飞，黄海. DSP 控制器原理及其在运动控制系统中的应用[M] 清华大学出版社, 2003：27～86

[17] 张雄伟. DSP 芯片的原理与应用[M]电子工业出版社, 1997：26～38

[18] 蔡学江，王宏文. DSP 在电机控制方面的应用[J].基础自动化, 2000, 7(4)：40～41

[19] 刘江晖，熊清平，姚幸. 基于 DSP 和 ISP 的运动控制板开发[J]机电一体化，2000，(5)：24～26

[20] Compumotor & Digiplan, [Z]Step Motor & Servo Motor Systems andControls, 1995：126～128

[21] 李长峰，王明宇，尤波. 基于 DSP 芯片的机器人运动控制卡设计[J]哈尔滨理工大学报，2003，(5)

[22] 周志红. 四关节机器人伺服控制系统的研制[D]湖南：中南大学，2004

[23] 周霖等. 控制工程技术应用[M]北京：国防工业出版社 2005

[24] 武安河，郃鈱，于洪涛. Windows 2000/XP WDM 设备驱动程序开发[M]北京：电子工业出版社 2004

[25] 樊丁等 基于多轴运动控制卡的伺服控制系统研究[J]甘肃工业大学学报 2005，28(2)：5-8。

[26] 盛刚，唐厚军 基于 PCI 总线和 DSP 的运动伺服控制卡的设计和实现[J]工业控制计算机 2004，17

[27] 陈晓平等 Protel 99 SE－电子线路 CAD 应用教程[M]南京：东南大学出版社.2005

[28] 徐灏 机械设计手册（第 3 卷）[M]机械工业出版社，1993：220～260

[29] 袁宏杰，李传日，殷雪岩. TMS320C3X 的汇编语言和 C 语言及混合编程技术[J]. 测控技术 2000，19(6):32－34.

[30] 霍伟著. 机器人动力学与控制[M] 北京：高等教育出版社. 2005

[31] 王胡舰，吴瑞生，孙翔，李正平. 利用接口芯片 PCI9052 制作 PCI 总线接口卡详解[J].工业控制计算机 2004，17(14):25－27。

[32] 李志勇，罗家融，岳冬利 HT27 极向场多变量控制器 TMS320VC33 的数据交换设计和实现[J]电子器件

[33] Daivid lee [M]Space Memory on the TMS320C24x DSP. Texas Instrument , 2002：32～58

[34] TMS320LF2407A 16-bit fixed point DSP with Flash. [Z] Texas Instrument ,2001：121～127

[35] LSI circuit for AC motor speed control. [Z]Technical publication M820015, MullardLimited：2003：22～29

[36] 吴晓峰，杨旭雷，张浩. 双口 RAM 在 DSP 人机接口中的应用[J]微机与应用，2002，(10)：24～27

[37] 黄廷磊，李敏 Windows 环境下数据采集控制软件的研究[J]上海理工大学学报，1998，(4)：356～358

[38] 孟卓，孟庆鑫，王晓东，谭定忠. 双口 RAM IDT7130 在 DSRV 对接机械手控制系统中的应用 哈尔滨理工大学学报,2000, 5(3)：48～50

[39] 刘乐善 微型计算机接口技术原理及应用[M]华中理工大学出版社，1996：26～28

[40] 章玮等. 数字信号处理器(DSP)在电机控制中的应用[J]中小型电机，2001，(4)：56～60

[41] 宁改娣，杨拴科 DSP 控制器原理及应用[M]科学出版社，2002：10～12

[42] 高军礼等. 基于 DSP 技术的多轴运动控制器的应用研究[J]组合机床与自动化加工技术，2002(3)：40～42

[43] Texas Instruments. TMS3x Assembly Language Guide. June 1998

[44] 李文涛，丁美新 基于 PCI 总线的运动控制方案[J]中小型电机 2005，32（1）：67-70.

[45] 张芳兰 TMS320C2xx 用户指南[M]电子工业出版社，1999：28～30

[46] 高盛涛 基于工业 PC 的 6 轴伺服控制卡的研究和实现[D]中国科学院电工研究所硕士论文, 2001：6～9

[47] 王宝国等. 复杂可编程逻辑器件(CPLD)在 DSP 交流电机控制系统中的应用[J]电机与控制学报 2001,.5(1)：123～125

[48] 刘和平，严利平，张学锋，卓清锋 TMS320LF240XDSP 结构、原理及应用[M]北京航空航天大学出版社，2002：15～98

[49] 刘宝琴等 ALTERA 可编程逻辑器件及其应用[M]清华大学出版社，1995：104～137

[50] 虞鹤松等 利用动态链接库开发 Windows 环境下的实时工业控制软件[J]电气自动化，1998，(4)：28～35

[51] 杨延西，任海鹏，刘丁 交流伺服系统中 DSP 与上位机的通讯实现[J] 电气传动，2000，(6)：28～30

[52] Reference Texas Instrument,[Z]1998：76～79

[53] 潘新民 微型计算机控制技术[M]人民邮电出版社, 1988：231～328

[54] 刘和平，王维俊，江渝，邓力. TMS320LF240XDSP C 语言开发应用[M]北京航空航天大学出版社，2003：84～175

致谢

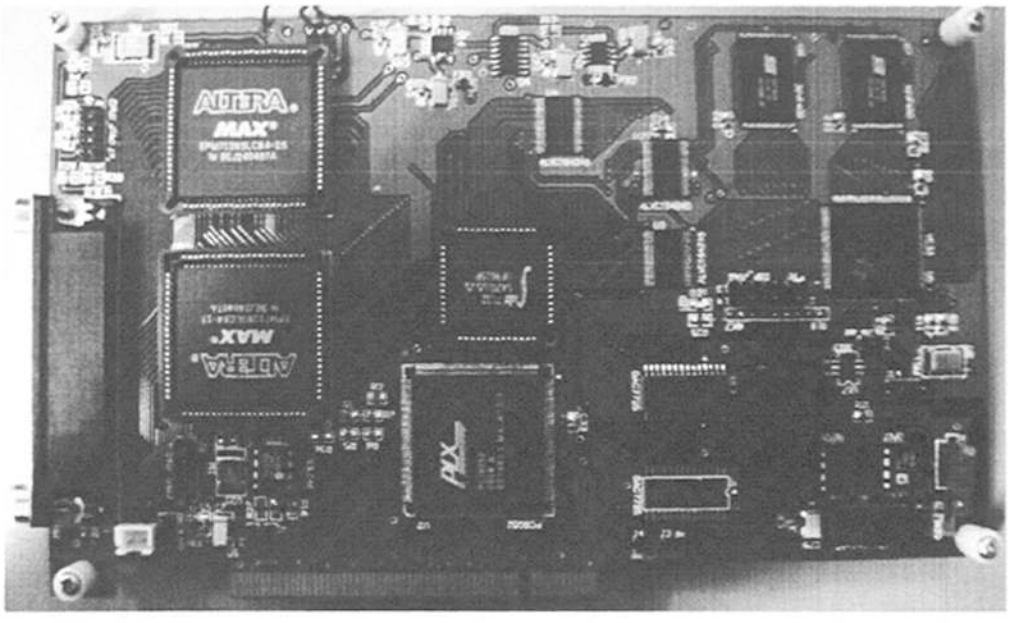
在整个攻读硕士学位的三年学习过程中，曹老师严谨的治学态度和以身作则、勤奋踏实的工作作风，敏捷的思维，渊博的知识均使我受益终身，在此表示我深深的谢意。

我的一点一滴的知识，都离不开老师的教诲，是启蒙老师为我开启了智慧宫殿的大门，中学老师使我确定了人生的目标，大学老师传授我更多的专业知识，在这里我感谢我所有的老师，我将在今后的工作中努力用所学知识服务社会，回报你们。

深深感谢我的父母以及一直关心着我的亲人和朋友们，对他们的关心、支持和鼓励，我将终生铭记在心。是他们用伟大的爱支持着我不断克服困难，不断获得成功。亲人的爱仅仅用语言是无法表达清楚的，在这里，我谨以此文献给他们。

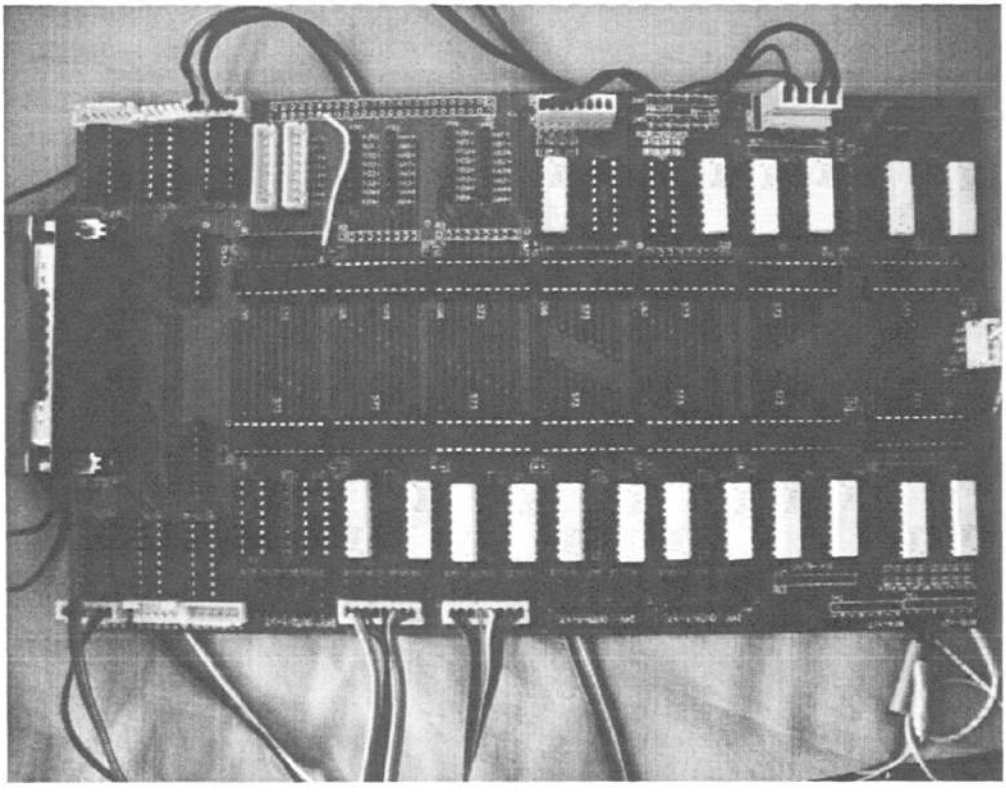
感谢研究生部的各位领导和老师在我攻读硕士研究生期间给予的各方面的关心和支持，感谢席文魁，薛永强，李宁等同学在我的课题中所给予的极大帮助，并感谢陕西科技大学的所有同学，是他们使我生活在一个朝气蓬勃，团结奋进的的大家庭里。

附录 1



DSP 运动控制卡

附录 2



伺服接口板

攻读学位期间发表的学术论文目录

[1]曹西京，宋海波，白雪峰，康兵 基于 Renishaw 的对数控机床定位精度和动态性能的研究[J]机床与液压，2009，03

[2]宋海波，曹西京 基于工控机 IPC 和 DSP 的机器人控制系统的设计与研究[J]陕西科技大学学报 2009，08