

光机电一体化丛书

光机电一体化系统 接口技术

方建军 编著



化学工业出版社

TH-29
220
1. 接

光机电一体化丛书

光机电一体化系统接口技术

方建军 编著



化学工业出版社

· 北京 ·

本书从接口的角度论述光机电一体化技术。根据系统输入/输出关系,将接口划分为机械接口、软件接口、电气接口、机电接口和人机接口五个部分。在讨论每种接口时,不仅注重新的接口器件和接口方法的介绍,还以丰富的实例来说明如何进行接口设计。

本书可供从事光机电一体化技术开发的工程技术人员使用,也可作为工业自动化、机械设计制造及其自动化等专业高等院校学生的教学参考书。

图书在版编目(CIP)数据

光机电一体化系统接口技术/方建军编著. —北京:
化学工业出版社, 2007. 1
(光机电一体化丛书)
ISBN 978-7-5025-9889-1

I. 光… II. 方… III. 光电技术-机电-体化-接口
IV. TH-39

中国版本图书馆 CIP 数据核字 (2007) 第 002637 号

责任编辑:任文斗
责任校对:吴静

文字编辑:钱诚
装帧设计:潘峰

出版发行:化学工业出版社(北京市东城区青年湖南街13号 邮政编码100011)

印刷:北京永鑫印刷有限责任公司

装订:三河市万龙印装有限公司

787mm×1092mm 1/16 印张16¼ 字数400千字 2007年6月北京第1版第1次印刷

购书咨询:010-64518888(传真:010-64519686) 售后服务:010-64518899

网 址: <http://www.cip.com.cn>

凡购买本书,如有缺损质量问题,本社销售中心负责调换。

定 价: 34.00 元

版权所有 违者必究

《光机电一体化丛书》编辑委员会

主任 林 宋

副主任 王生则 赵丁选 罗 均

委 员 (排名不分先后)

林 宋	王生则	赵丁选	罗 均	胥信平	黎 放
胡于进	何 勇	谢少荣	高国富	崔桂芝	殷际英
方建军	郭瑜茹	徐盛林	文秀兰	周洪江	刘杰生
蒋 蓁	王 琦	杨野平	王东军	尚国清	叶天朝
戴 荣	刘 勇	裴晓黎			

序

光机电一体化是激光技术、微电子技术、计算机技术、信息技术与机械技术的相互交叉与融合，是诸多高新技术产业和高新技术装备的基础。它包括产品和技术两方面：光机电一体化产品是集光学、机械、微电子、自动控制和通信技术于一体的高科技产品，具有很高的功能和附加值；光机电一体化技术是指其技术原理和使光机电一体化产品得以实现、使用和发展的技术。

目前，国际上产业结构的调整使得各个行业不断融合和协调发展。作为光学、机械与电子相结合的复合产业，光机电一体化以其特有的技术带动性、融合性和普适性，受到了国内外科技界、企业界和政府部门的特别关注，在提升传统产业的过程中，它以其高度的创新性、渗透性和增值性，成为未来制造业的支柱，被誉为 21 世纪最具魅力的朝阳产业。我国已经将发展光机电一体化技术列为重点高新技术发展项目。

随着光机电一体化技术的不断发展，各个行业的技术人员对其兴趣和需求也与日俱增。《光机电一体化丛书》第一批（共 9 册）的出版，受到了广大读者的欢迎。为满足读者的进一步需求，我们联合北方工业大学、上海交通大学、东华大学、华中科技大学、海军工程大学、北京机械工程学院、中国船舶工业集团船舶系统工程部、上海大学、吉林大学、江汉大学、河南理工大学等高校的教师及科研部门的工程技术人员编写《光机电一体化丛书》第二批（共 21 册），拟在 2005 年初开始陆续出版发行，主要内容为光机电一体化技术在测试传感、驱动控制、激光加工、精密加工、机器人等方面的应用，以满足科研单位、企业和高等院校的科研及生产和教学的需求，为有关工程技术人员在开发光机电一体化产品时，提供有价值的参考素材。

本丛书的基本特点是：①内容新颖，力求及时地反映光机电一体化技术在国内外的最新进展和作者的有关研究成果；②系统全面，丛书分门别类地归纳总结了光机电一体化技术的基本理论和在国民经济各个领域的应用实例，重点介绍了光机电一体化技术的工程应用方法和实现方法；③深入浅出，每本书重点突出，注重理论联系实际，既有一定的理论深度，又具有很强的实用性，力求满足不同层次读者的需求，适合工程技术人员阅读和高校机械类专业教学的需要。

由于本丛书涉及内容广泛，相关技术发展迅速，加之作者水平有限，时间紧迫，书中错误和不妥之处在所难免，恳请专家、学者和读者不吝赐教。

《光机电一体化丛书》编辑委员会

2004 年 10 月于北京

前 言

自 20 世纪 70 年代以来,机电一体化技术方兴未艾。机电一体化产品在军事、工农业生产 and 日常生活中随处可见,占据重要的地位。机电一体化技术是多学科知识交叉融合而形成的边缘学科。它的最大特点就是不断吸收、消化、融合各种高新技术的最新成果,吐故纳新,动态发展。光机电一体化在机电一体化的基础上,更多地融合了现代光学技术。从光机电一体化的技术体系上看,它涉及到现代光学、精密机械、微电子、信息处理与控制 and 计算机等多种学科,是一个庞大的技术体系。光机电一体化产品是综合运用多种高新技术的最新成果而生产出来的高附加值产品。随着技术的发展和进步,各单元技术越来越先进和完善。是不是单元技术先进了,总体集成后的技术就一定先进呢?经过多年的实践,人们发现仅有先进的单元技术并不能使整个系统最优化。光机电一体化是使机电产品最优化设计的方法学。这个概念是从设计哲学 and 设计理念上概括光机电一体化,是对过去机电一体化产品设计三步骤的否定。为了探讨机电一体化设计方法学,各国学者提出了不同的理论和看法,但都不具有可操作性。模块化、标准化是现代技术发展的重要特征。对于一个复杂的光机电一体化系统而言,通常不是从零开始设计,即从一个个具体的零部件开始设计,而是按照选用模块、设计模块,然后总体集成的思路来设计。因此,从某种意义上讲,光机电一体化就是解决接口的技术。接口技术就是使不同子系统之间的信息、能量能顺畅地流动,使不同子系统的功能相互补充,相得益彰。从目前情况看,除了极少的文章探讨了机电接口技术,还没有比较系统的接口技术方面的著作。鉴于接口技术的重要性和存在的空白,本人萌发了编写本书的想法。本书在写作思路上有如下特点。

① 注意实用性 接口技术是一个实用性很强的技术,本书在介绍接口时,尽可能地多列举实例,帮助读者理解接口方法。对于不太实用的知识,并没有为了系统性而保留下来,坚决摒弃。如电机驱动电路都采用集成电路,不再介绍用分立元件搭建电路。

② 知识新、信息量大 接口技术涉及的面广,有丰富的知识和信息需要掌握。本书注重介绍最新的知识和技术,让读者了解接口技术发展动态。例如在第 6 章人机接口就介绍了生物特征识别技术;在第 2 章软件接口技术中介绍了 WDM 开发技术等。

③ 面向初、中级水平的读者,尽量做到语言简练、内容具体、通俗易懂。

本书共分 6 章,分别介绍软件接口、机械接口、电气接口、机电接口和人机接口。每章都有丰富的实例,向读者介绍接口技术的具体知识和方法,同时也介绍一些最新技术。作者将自己多年教学和科研经验融入书中,力图写出受读者欢迎的科技图书。但由于知识和经验有限,疏漏之处在所难免,敬请读者批评指正。

本书写作过程中,研究生常永超和窦海斌帮助整理文字和图表,在此表示感谢。同时还要感谢 Microchip 公司的网站,它提供了 PIC 系列单片机的丰富资料。

方建军

2006.12.20

目 录

第 1 章 绪论	1
1.1 什么是接口	1
1.2 接口的分类	2
1.3 接口技术的发展趋势	3
第 2 章 软件接口	5
2.1 混合语言编程	5
2.1.1 汇编语言和高级语言混合编程	5
2.1.2 嵌入式系统中 C 语言与汇编语言混合编程	9
2.1.3 动态链接库.....	11
2.1.4 高级语言之间的混合编程.....	14
2.2 设备驱动程序.....	19
2.3 软件接口综合应用实例.....	28
第 3 章 机械接口	32
3.1 机械零接口设计.....	32
3.1.1 刚性联轴器.....	32
3.1.2 无弹性元件的挠性联轴器.....	33
3.1.3 弹性联轴器.....	35
3.1.4 伺服机构用联轴器.....	38
3.1.5 联轴器选型与应用实例.....	41
3.2 机械被动接口设计.....	45
3.2.1 精密齿轮传动.....	45
3.2.2 谐波齿轮传动.....	49
3.2.3 滚珠丝杠.....	54
第 4 章 电气接口	60
4.1 电气接口类型与功能.....	60
4.1.1 输入/输出信号的种类	60
4.1.2 光机电一体化系统的输入/输出通道	61
4.2 开关和数字量输入输出接口.....	61
4.2.1 开关量输入输出接口.....	61
4.2.2 数字量输入输出接口.....	65
4.3 模拟量输入输出接口.....	77
4.3.1 模拟量输入接口.....	77
4.3.2 模拟量输出接口.....	91

第 5 章 机电接口	95
5.1 光电隔离功率放大器件	95
5.2 常用的电力电子器件	96
5.2.1 可关断晶闸管 GTO 的特性及参数	97
5.2.2 功率晶体管 GTR 的特性与应用	101
5.2.3 功率场效应管 MOSFET 的特性与应用	106
5.2.4 绝缘栅双极晶体管 IGBT 的特性与应用	111
5.2.5 智能功率集成电路和智能功率模块	115
5.3 步进电动机驱动与控制	117
5.3.1 步进电动机的结构与工作原理	117
5.3.2 步进电动机的驱动电路	120
5.3.3 步进电动机驱动与控制实例	132
5.4 直流伺服电动机驱动与控制	136
5.4.1 直流伺服电动机的 PWM 驱动原理	137
5.4.2 直流伺服电动机集成驱动电路	138
5.4.3 直流伺服电动机驱动与控制实例	142
5.5 交流伺服电动机驱动与控制	154
5.5.1 交流异步电动机变频调速原理	155
5.5.2 SPWM 波发生器芯片 SA4828	158
第 6 章 人机接口技术	164
6.1 输入接口	165
6.1.1 键盘接口设计	165
6.1.2 拨码盘接口设计	176
6.1.3 触摸屏接口设计	178
6.1.4 生物特征识别输入	183
6.2 输出接口	192
6.2.1 LED 显示器的接口设计与编程	192
6.2.2 LCD 显示器的接口设计与编程	198
6.2.3 热敏打印机的接口设计与编程	206
6.3 人机接口应用综合实例	233
附录 1 CBD 系列外插管埋入式双螺母垫片预紧滚珠丝杠副	242
附录 2 XB1 系列谐波齿轮减速器额定数值	246
附录 3 滚珠丝杠副选用系数	248
参考文献	249

第 1 章 绪 论

1.1 什么是接口

光机电一体化系统是集光学技术、机械技术、计算机技术、控制技术、检测传感技术、伺服驱动技术和系统集成技术等于一体的多学科融合的边缘技术。各种单元技术相互渗透、相互影响,使光机电一体化技术的内涵和外延得到不断丰富和拓展。从最初的机械电子化发展到机电一体化,进而发展成光机电一体化和微光机电一体化。光机电一体化技术向着智能化、模块化、网络化、集成化方向迅猛发展。光机电一体化产品的柔性更强,智能化程度更高,系统也更加复杂。从 20 世纪 70 年代机电一体化技术诞生以来,各种机电一体化单元技术发展很快,技术也更加成熟和完善。经过多年的实践,人们发现仅有先进的单元技术并不能使整个系统最优化。因为光机电一体化系统是多个子系统的复合体。各子系统的集成并不是简单的叠加,而是需要运用系统工程的理论和方法将各子系统有机地融合。每个子系统都有自己的特征,子系统之间存在必然的内在联系,但这种联系并不十分明显。人们在设计机电一体化系统或产品时,往往采用从机械系统设计开始的三个步骤完成。当机械系统设计完成后,依次设计动力系统和微电子系统,最后设计控制器和控制算法。这种顺序设计的思路,往往将机械设计的问题带到电子系统和控制系统设计中。例如,在精密数控机床中,机械设计工程师从节约成本的角度考虑,选用普通滚珠丝杠而不是精密滚珠丝杠作为传动件,希望通过闭环控制系统来补偿机械系统产生的误差。从机械设计角度看,节约了成本。但是,要保证数控机床的定位精度和运行平稳性,控制工程师需要采用高精度的光栅作为位置传感器来构成闭环控制系统,从控制算法上补偿系统误差,保障系统的定位精度。从综合的角度考虑,系统的成本可能更高,甚至导致系统不能正常工作。机械设计工程师和控制工程师本着“各人自扫门前雪”的态度来设计单元技术,最后的系统集成是随机的,谈不上优化。

光机电一体化系统是多系统的集成和融合,应采用系统工程的方法进行设计。有人提出,机电一体化是一种用于机电产品最优化设计的方法学,从设计哲学上说明机电一体化不仅仅是技术的集合体,也是一种设计理念和设计思想。系统工程学的方法要求改变传统的串行设计思路,转而采用并行方法进行设计。在预设计阶段,对各子系统进行综合分析,提出子系统的功能和子系统之间的接口方案。在进行设计的过程中,不同子系统的设计师互相交换信息,适时调整设计方案,向系统最优化方向发展。要实施并行设计思想,一个很重要的因素是子系统设计人员要有“共同语言”,互相能进行技术交流。这个共同语言就是光机电一体化。也就是说,子系统设计人员都具备光机电一体化技术知识,只不过是“术有专攻”而已。德国在这方面有独到的见解,强调培养人才的跨学科思维能力,促进光机电一体化技术各学科的相互渗透。

要设计系统整体最优化的光机电一体化产品,需要对机电系统有一个合理的划分。对机电系统模型的看法主要有以下几种。

① 五块论 德国 Darmstadt 大学的 Rolf Isermann 认为,机电系统是由控制功能、动力功能、传感检测功能、操作功能、结构功能等五大功能模块组成。五块论加深了人们对机电

一体化系统的认识，使系统设计有章可循；但也很容易使设计者陷入条块分割、各自为政的局面，不利于机电一体化系统的融合与集成。

② 三环论 丹麦理工大学的 Jacob Baur 等人提出机电一体化系统是由机械、电子、软件三个相关圆环组成，以此表示机电一体化系统的组成和相互关联。三环论中电子含义比较含糊；软件是指实现信息处理和控制的程序，同样无法使人们对真实系统作科学和明确的组成划分。三环论纯属于理论层面的探讨，没有实际的指导意义。

③ 两个系统 挪威科技大学的 Bassam A Hussein 提出将机电一体化系统划分为物理系统与控制系统两大子系统。物理系统包括各种驱动装置、执行机构、传感器等；控制系统包括软件、硬件。两子系统论从功能划分来看，比前一种看法更趋于完善，有利于对控制系统的深入研究。但是物理系统比较笼统，划分不是很细，使系统设计无从下手。

④ 国内有人提出将机电一体化系统划分为广义执行机构子系统、传感检测子系统、信息处理及控制子系统等三个子系统。这种提法仍不能清晰地界定各子系统，不利于直接用于系统设计。

从机电一体化的本质上讲，机电一体化就是接口技术。随着各种单元技术的发展和完善，子系统之间的接口显得越来越重要。接口技术就是要研究各子系统之间的连接问题，使信息、能量在各子系统之间顺利流动，达到系统整体上的最优化。图 1-1 表示光机电一体化系统的组成和接口，各子系统之间通过不同的接口实现连接。对接口要求，不仅仅是连接各个子系统，更重要的是通过接口将各子系统的特性和优点有机地融合在一起，扬长避短，从而实现各子系统单独都无法实现的新功能，达到系统最优化。因此，系统性能在很大程度上取决于接口性能。各子系统之间的接口性能成为判断系统整体性能好坏的决定因素。接口技术的研究对象正是机电一体化系统中的各种接口。光机电一体化系统的接口包括硬件接口和软件接口两大类。硬件接口主要在子系统之间或人与光机电一体化系统之间建立物理连接，为信息和能量的输入/输出、传递和转换提供物理通道。软件主要是提供系统信息交互、转换、调整的方法和过程，协调和综合光机电一体化组成技术，使各子系统集成并融合为一个整体，实现新的功能。对接口的深入研究，对光机电一体化技术的发展至关重要。

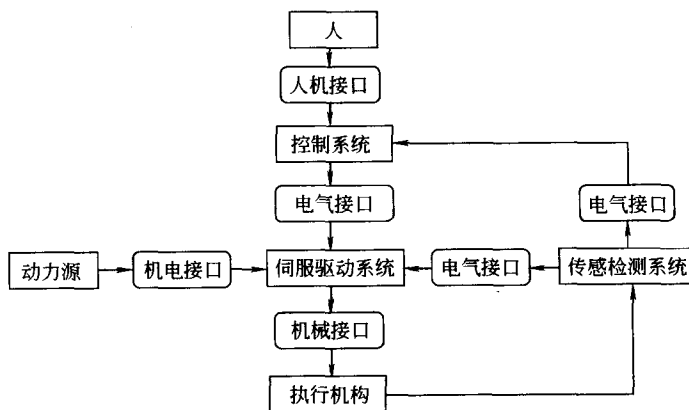


图 1-1 光机电一体化系统的组成与接口

1.2 接口的分类

光机电一体化系统的接口根据输入/输出的关系，可划分为人机接口、机电接口、机械

接口、电气接口和软件接口（信息接口）。

人机接口解决人与光机电一体化系统之间的接口。通过人机接口，可以向系统输入操作指令，系统的运行状态也可以通过输出器件显示出来，便于操作者监视系统的运行状态，控制其运行过程。人机接口是双向互动的。接口硬件包括输入/输出设备，主要有键盘、拨码开关、按钮、触摸屏、LED、LCD显示屏、打印机等。高级人机接口还包括语音识别、指纹识别、虹膜识别、面部识别等。当然，人机接口也应包括产品与环境的关系，产品与人的关系。产品与环境的关系体现在产品报废后的可拆卸、可回收、无污染；在产品生存期内，对环境的友好性。产品与人的关系体现在产品的宜人性。随着人们生活水平的提高，审美情趣也越来越高。光机电一体化系统或产品除了要满足系统的性能要求外，还应具有宜人的外观和文化底蕴。在激烈的市场竞争下，具有相同功能的产品，其外观设计成为决定消费者是否购买的重要因素。换句话说，人机接口性能的好坏决定产品的市场价值。因此，光机电一体化产品不仅要注重技术方面的接口，还应注意非技术因素的接口。由于产品的宜人性设计属于工业设计的范畴，本书对这部分接口内容没有涉及。

机电接口主要解决动力源与驱动装置之间的能量转换接口。光机电一体化系统中的驱动装置有各种交、直流电动机、液压执行装置、气动执行装置和其他新型执行装置。驱动这些装置需要用到机电接口。

电气接口主要解决控制系统与驱动系统、驱动系统与传感器、传感器与控制系统之间的接口。也就是说，电气接口将传感器输出的信号进行转换，输入到控制系统中；将控制系统输出的数字量信号转换为驱动系统所需要的信号。这些信号的转换都是通过软件进行的。

软件接口主要解决不同总线、不同程序之间的信息格式转换。例如，ISA总线与USB总线的转换、Windows下的设备驱动程序、不同语言之间的混合编程等。

机械接口技术解决执行机构和驱动系统之间的接口。机械接口有零接口和被动接口两种。零接口直接将执行机构和驱动系统连接，不进行能量和运动的变换。例如用联轴器将电动机轴和执行机构直接连接。被动接口要进行能量的转换和运动的变换。例如用滚珠丝杠将直线运动转化为旋转运动或将旋转运动变换为直线运动，用减速器进行减速增矩。

1.3 接口技术的发展趋势

随着光机电一体化技术的发展，系统的智能化程度越来越高，系统也变得越来越复杂。不仅系统之间的接口变得复杂，系统内部之间的接口也很复杂。从国内外光机电一体化的发展现状和高新技术的发展动态来看，光机电一体化接口技术将朝着以下方向发展。

① 标准化 光机电一体化系统的发展趋势之一是模块化。模块化的必然要求是标准化，使不同的模块具有一致的接口规范。接口在保持一致性的前提下，还要保证能量、信息的顺畅流动。

② 智能化 智能化是光机电一体化与传统机械自动化的根本区别，也是接口技术发展的主要方向。接口不再简单地进行能量和信息的传递，而是向着信息采集、处理的智能化方向发展。

③ 微型化 微光机电一体化是光机电一体化技术发展的趋势之一，它是机械技术和电子技术在纳米尺度上相融合的产物。微光机电一体化的系统构成同光机电一体化有很大的不

同，其接口也趋向于微型化。

④ 人性化 未来的光机电一体化更加注重与人的关系，接口除了满足技术要求外，还要注重人的心理感受，提高产品与人和环境的友好性，提高产品的可靠性和安全性，将人与系统有机地融合在一起，形成新的光机电一体化系统。

⑤ 网络化 网络使时间和空间距离缩短，未来的接口技术在网络化方面应有较大的作为。

第 2 章 软件接口

软件是光机电一体化系统的灵魂，是系统不同部件之间联结的桥梁和纽带。随着光机电一体化技术的发展，光机电一体化系统的智能化程度越来越高，大量新的智能控制算法得到应用。这些算法都是用软件来实现的。光机电一体化技术发展趋势之一就是模块化，不同模块可能用不同的软件来实现，因此模块之间的接口就涉及到软件接口。即使在同一个模块内，也有可能用不同的软件开发工具进行开发，软件之间也存在接口问题，如在 C 语言程序中嵌入汇编程序，以提高运算速度。用 VB 语言开发软件界面，用 C 语言编写运算、控制程序，通过动态链接库将两者连接起来，这也是软件接口。只要有软件的地方，就存在软件接口问题。

2.1 混合语言编程

2.1.1 汇编语言和高级语言混合编程

在所有的编程语言中，汇编语言产生的代码效率最高，能够直接访问硬件，但其编程难度大、程序维护困难、程序可移植性差。C 语言是介于高级语言和低级语言之间的“中级语言”，代码效率高、编程容易、程序可读性和可移植性强，因此 C 语言是一种十分流行的编程语言。当对运行速度和定时时间要求很苛刻时，最佳的解决方案是利用汇编语言和 C 语言的混合编程。对速度要求很高的处理模块用汇编语言编写，而其余模块用 C 语言完成。在编写硬件驱动程序、工业实时控制以及大量计算方面，利用 C 语言与汇编语言混合编程可以提高程序的效率。混合编程使用两种语言相互调用，进行参数传递，共享数据结构和数据信息。由于不同语言在程序格式、函数调用、寄存器使用等方面都有不同的规定，所以混合编程要解决两者的接口和参数传递。

(1) Turbo C 语言和汇编语言的混合编程

C 语言和汇编语言混合编程一般有以下三种方法。

- ① 对 C 语言程序编译后形成的汇编程序进行手工修改和优化。
- ② 直接在 C 语言程序中嵌入汇编语句。
- ③ 分别编写 C 语言程序和汇编程序，独立编译成目标代码模块后再进行链接。第一种方法对程序编写者的要求较高，一般很少采用。下面主要就后两种方法进行讨论。

在 C 语言中嵌入汇编语言的方法有两种形式。第一种就是在汇编语句的前面加上关键字 `asm`，例如，

```
asm mov ax, 10      /* 把 10 给到 ax 寄存器 */
```

第二种是用一个大括号将一组汇编语句包围起来，前面加上关键字 `asm`，即

```
asm {  
    mov ah, 2  
    mov bh, 0  
    mov dl, 20  
    mov dh, 10
```

```
int 10h
```

```
}
```

C语言和汇编语言混合编程的第三种方法是最常用的，即分别编写C语言程序和汇编程序，独立编译成目标代码模块后再进行链接。具体的步骤如下。

- ① 在TC集成环境中编写文件名后缀为.c的程序，如sum.c。
- ② 在TC集成环境中用编辑器编写文件名后缀为add.asm的汇编程序。
- ③ 用宏汇编MASM汇编add.asm，即masm add.asm，生成目标文件add.obj。
- ④ 在TC集成环境下建立一个工程文件sum.prj，将sum.c和add.asm添加到工程文档中，如

```
sum.c
```

```
add.asm
```

- ⑤ 编译链接前，在OPTION-LINKER选项中关闭CASE-SENSITIVE LINK，使其大小写不敏感，对工程文件编译链接，生成可执行文件sum.exe。

[实例1] 编写两个整数相加的程序，汇编程序作为C语言程序的外部调用函数。

汇编程序

```
_dataseg segment byte public 'code'
assume cs: _dataseg
public _sum ; 声明 _sum 为外部调用函数
_max proc near
    push bp
    mov bp, sp
    mov ax, [bp+4] ; 取出函数的第一个参数
    add ax, [bp+6] ; 取出第二个参数并施加加法运算
    pop bp ; 恢复 bp 寄存器的内容
    ret ; 返回主程序
_sum endp
_dataseg ends
end
```

C语言程序

```
extern "C" int sum (int a, int b); /* 声明汇编程序为外部调用函数 */
void main () {
    int x=5, y=20;
    int add;
    printf ("%d+ %d=%d/n", x, y, sum (x, y)); /* 调用汇编程序求和 */
}
```

由于C语言遵守C约定，其他语言、包括汇编语言，都是使用Pascal约定。两者的区别如下。

- ① C约定生成的汇编代码中，符号严格按C语言程序中的大小写生成，而Pascal约定则将符号均转换成大写，因此在汇编程序中所有变量和函数名都使用小写，不用大小写混用的变量命名。

② C 约定生成的符号均自动加下划线，而 Pascal 约定则不加下划线，因此在汇编程序中的函数用下划线。

③ 调用函数时，C 约定是将函数参数从右往左倒序压栈，Pascal 约定则是从左到右压栈。

④ C 语言程序和汇编程序互相要调用函数时，被调用函数在 C 语言程序中用 extern 进行声明，汇编程序中函数名前加下划线“_”，汇编程序中用 public 定义相应的子程序为公用函数，能够被 C 语言程序调用。

混合编程中，参数传递是一个很重要的问题。一般参数传递都是通过数据结构——堆栈实现的，堆栈的特性就是“先入后出”。汇编程序从 C 语言程序中获得参数时，要求说明是何种语言调用、调用的参数及其类型。主程序调用子程序时，参数传递的次序与它们出现的次序是相反的，从后到前向堆栈中压入参数。汇编程序获得参数时要用到寄存器 BP，因此首先要保存 BP。如果子程序中有自己的数据段，也要保存原有的 DS。传递完参数后，C 语言程序还将保存 (CS, IP)，其顺序是 CS 在前，IP 在后。这个过程是 C 语言自动进行的，不需要编程者干预。在实例 1 中，传递参数 $x=5$ 和 $y=20$ 的堆栈情况如图 2-1 所示。

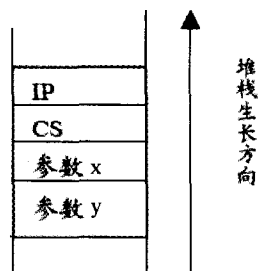


图 2-1 参数存储方式

当 C 语言程序需要从汇编程序中获得某个参数时，这个参数应该通过寄存器 AX, DX 来传递，具体使用哪些寄存器取决于参数的大小。char 型数据 (1 个字节)、short 型数据 (2 个字节) 放在寄存器 AX 中，int 型数据 (4 个字节) 放在寄存器 DX: AX 中，其中 DX 存放高位字，AX 存放低位字。BX 和 CX 寄存器可以任意使用。

(2) Visual C++ 语言和汇编语言的混合编程

基于 Windows 的图形界面具有友好的人机界面，成为当前软件开发的事实标准。Visual C++ 是微软推出的基于 Windows 环境的集成可视化编程工具。目前的最新版本是 Visual C++ 7.0。在可视化集成环境下，能实现软件代码的编辑、编译、链接和调试一体化，还提供 AppWizard、ClassWizard 等多种实用开发工具，使得在 Windows 下开发面向对象的应用程序变得更加轻松、快捷。C++ 语言相对其他高级语言已经有了很高的代码效率，但是如果直接访问内存、寄存器、I/O 端口等硬件，C++ 库函数和 WinAPI 要由编译器经过一系列转换后才能完成这些操作，速度比汇编代码慢几倍甚至十几倍。因此，在一些实时性要求高的应用中，考虑在 C++ 程序中嵌入汇编代码，充分发挥汇编代码内存需求小、能直接访问硬件端口、执行速度快的优势。同时保留了 VC++ 面向对象、可视化等 Windows 程序风格和 C++ 语言对数据的强大处理能力。实际上，VC++ 从最早版本起就一直支持与 MASM 汇编语言混合编程。

在 Visual C++ 中调用汇编程序的方法有两种，一种即在 Visual C++ 程序中直接嵌入汇编语句，称之为嵌入式汇编；另一种方法是用两种语言分别编写独立的程序模块，汇编程序源代码汇编产生目标代码 OBJ 文件，将 C++ 源程序和 OBJ 文件组建成工程文件，然后进行编译和链接，生成可执行文件。

① 嵌入式汇编 嵌入式汇编又称在线汇编。Visual C++ 提供嵌入式汇编功能，允许在 C++ 源程序中直接插入汇编语言指令的语句，可以直接访问 C++ 语言程序中定义的常量、

变量和函数，而不用考虑二者之间的接口，从而避免了汇编语言和 C++ 语言之间复杂的接口问题，提高了程序设计效率。嵌入的汇编代码由 VC++ 的在线汇编编译器统一完成，不需单独编译，使用相当方便。通过关键字 “_asm” 将 MASM 汇编语句加入到 VC++ 程序中。嵌入汇编的方法有两种，一种即在 VC++ 代码行中以 “_asm 汇编语句” 这样的单行形式嵌入，另一种是以 “_asm {…}” 这样的块状形式嵌入。例如

```
_asm {  
    _asm mov eax, 5h      mov eax, 5h  
    _asm mov ecx, 5h     mov ecx, 5h  
    _asm add eax, ecx    mov eax, ecx }  
}
```

② 模块调用法 C++ 程序模块和汇编模块可以互相调用。大多情况下，C 语言程序模块调用汇编模块。采用模块调用方式，要在变量和函数命名、函数调用、参数传递和返回值等方面进行约定。C++ 语言具有三种调用协议：_cdecl、_stdcall 和 _fastcall。MASM 支持的语言类型有：C、FORTRAN、PASCAL、STDCALL、SYSCALL 和 BASIC。C++ 语言与汇编语言混合编程，通常利用堆栈进行参数传递，调用协议决定利用堆栈的方法和命名约定。两者要保持一致，通常 Visual C++ 采用 _cdecl 调用协议，MASM 汇编语言采用 C 语言调用协议。Visual C++ 没有远、近调用之分，所有调用都是 32 位的偏移地址，所有的地址参数也都是 32 位偏移地址。参数返回时，小于或等于 32 位的数据扩展为 32 位，存放在 EAX 寄存器中返回；4~8 个字节的返回值存放在 EDI：EAX 寄存器对中返回，更大字节数据则将它们的地址指针存放在 EAX 中返回。对 Visual C++ 和汇编语言中使用的外部函数和变量应该进行声明，并且标识符应该一致。Visual C++ 语言对标识符区分大小写，而汇编语言不区分大小写。在 Visual C++ 语言程序中，采用 extern “C” {} 的方式对所调用的函数和变量加以说明，即

```
extern “C” {返回类型 调用协议 函数名称 (参数表);}  
extern “C” {变量类型 变量名称;}  
}
```

汇编语言程序中供外部使用的标识符用 PUBLIC 标识为外部使用。

汇编模块和 C 语言模块的编译、链接如下：首先将汇编源代码编译成 obj 文件，然后建立一个工程，包含 C 语言源代码文件 (.cpp) 和汇编目标文件 (.obj)，对工程文件进行编译、链接，生成可执行文件。需要注意的是，Visual C++ 与汇编语言混合编程的环境是 32 位的，因此汇编过程中存取堆栈的寄存器使用 32 位的 EBP，而不是 BP。

Windows2000 以上版本的操作系统，禁止直接对系统硬件的访问。要实现硬件访问和控制，只能通过编写设备驱动程序和使用 WinAPI 等手段。这些方法比较复杂，对程序员要求也相对较高。选择使用嵌入汇编的 VC++ 编程，既保持了 VC++ 的 Windows 程序风格，又具备灵活地对内存、总线、I/O 端口、寄存器的控制访问能力，是一种可以实现对硬件资源访问的方便可行的编程方式。下面的实例用汇编语言和 VC++ 混合编程实现硬件资源的访问。

[实例 2] 向采集设备发送脉冲并读取量化后的数据。

```
BOOL CSample:: ReadData () {  
    MemData= new char [1024 * 1024 * MemLen];  
    if (MemData==NULL) {  
        AfxMessageBox (“Failed to allocate memory!”); return FALSE; }  
}
```



```

        int i=MemLen;        // 全局变量值赋给 i
    ...
    // 汇编代码
    _asm {
        mov  ebx, MemData    // 将存储采集数据的数组首地址传给 ebx 寄存器
        mov  eax, i          // 引用 C++ 中定义的变量
        mov  ecx, 20         // 循环次数
        shl  eax, cl         // 引用 C++ 中定义的变量
        mov  ecx, eax
        mov  edx, PORT_NO    // 设定端口号, PORT_NO 由 #define 定义
begin:   in  al,  edx         // 从指定的端口处读取数据
        mov  [ebx], al       // 将数据存储到 ebx 指向的内存地址
        inc  ebx             // 内存地址向前移动 1 个字节
        loop begin          // 判断 ecx 的值, 是否退出循环
    }
    ...
    return TRUE;
}

```

2.1.2 嵌入式系统中 C 语言与汇编语言混合编程

在早期的嵌入式系统开发中, 普遍采用汇编语言编程。当程序比较复杂时, 用汇编语言编写程序的效率比较低, 程序开发周期长, 可读性差。嵌入式 C 语言及其编译器的推出, 使编程变得相对简单。用 C 语言开发程序, 具有开发周期短、可读性强、可移植性好和维护方便等优点, 因此得到普遍采用。但是, 嵌入式 C 语言程序反汇编生成的汇编代码往往较长, 运行速度相对较慢。对实时时钟系统、要求执行效率高的系统, 用 C 语言编程就不适合。汇编语言具有直接和硬件打交道、执行代码的效率高等特点, 可以做到 C 语言所不能做到的一些事情。在这些场合, 使用汇编语言就成了惟一的选择。结合 C 语言和汇编语言的混合编程, 在嵌入式系统的开发中也是常用的手段。

目前大多嵌入式系统中, C 语言和汇编语言混合编程的方式有两种情况: 一种是汇编程序部分和 C 语言程序部分为不同的模块或不同的文件, 通常由 C 语言程序调用汇编语言程序模块的变量和函数。另一种是嵌入式汇编, 即在 C 语言程序中嵌入一段汇编语言程序。

一般情况下, C 语言程序模块是程序的主体, 汇编语言程序模块通常是由用 C 语言实现效率不高的函数组成, 如常用的数学函数, 也可包含已经成熟的、没有必要再转化成 C 语言的汇编子程序。在 C 语言程序模块和汇编语言程序模块中, 都可能使用临时寄存器来存储中间计算结果或存储函数的返回值。C 语言编译器把从 1CH 到 23H (或 2BH) 的这一片低端寄存器单元作为 C 语言程序模块的临时寄存器单元。如果在汇编语言程序模块中也使用这一片区域作临时寄存器, 会增加编译器的负担, 降低代码优化程度。因此在汇编语言程序模块中单独开辟一片低端寄存器区作为其临时寄存器区。这片区域既可用于汇编语言程序模块内部的数据交互, 也可用于汇编语言程序模块与 C 语言程序模块之间的数据交互。

C 语言程序和汇编语言程序可以互相调用。当在 C 语言程序模块中调用汇编语言程序中的函数或变量时, 要事先用 extern 声明被调用函数或变量是外部的; 这些被调用的函数或