电子工程技术丛书

# 自动测试系统与虚拟仪器
# 原理·开发·应用

刘思久　张礼勇　编著

電子工業出版社
Publishing House of Electronics Industry
北京·BEIJING

## 内 容 简 介

本书在阐明自动测试系统和虚拟仪器基本概念的基础上,深入讨论了数据采集器、GPIB 和 VXI 仪用总线、基于 RS-232 接口和通信网络的远程测试系统、基于 USB 的虚拟仪器、PCB 电子功能模件的自动测试、典型软件算法和典型应用系统等诸多内容。本书力求简明地阐述有关原理和规范的要点,再以应用电路和编程示例说明自动测试系统和虚拟仪器的设计思想和实现方法。

本书可供从事自动测试工作的科技人员和大专院校相关专业师生参考,也可作为教材直接使用。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有,侵权必究。

**图书在版编目(CIP)数据**

自动测试系统与虚拟仪器原理·开发·应用/刘思久,张礼勇编著.—北京:电子工业出版社,2008.8
(电子工程技术丛书)
ISBN 978-7-121-09132-2

Ⅰ.自… Ⅱ.①刘… ②张… Ⅲ.①自动检测系统 ②智能仪器 Ⅳ.TP274 TP216

中国版本图书馆 CIP 数据核字(2009)第 104946 号

策划编辑:张 榕
责任编辑:周宏敏
印 刷:
装 订:
出版发行:电子工业出版社
北京市海淀区万寿路 173 信箱 邮编 100036
开 本:787×1 092 1/16 印张:19.5 字数:499 千字
印 次:2009 年 8 月第 1 次印刷
印 数:4 000 册 定价:39.00 元

凡所购买电子工业出版社图书有缺损问题,请向购买书店调换。若书店售缺,请与本社发行部联系,联系及邮购电话:(010)88254888。

质量投诉请发邮件至 zlts@phei.com.cn,盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线:(010)88258888。

# 前　言

现代社会快速发展所提出的各种测试任务,已不再满足于眼观手记的传统形式,也不再可能都去为之研制专用的传统仪器了。以计算机为核心、若干通用设备共同完成测试任务的自动测试系统应运而生,并将虚拟仪器的理念变为现实。本书试图能够比较全面地阐明自动测试系统和虚拟仪器的基本概念、工作原理、典型形式和实现方法,期望能对读者从事这一领域的工作有所帮助。这里说明几点想法,以便读者了解本书写作的思路。

本书作者曾长期从事自动测试领域的科技攻关,亲身经历了自动测试系统的结构体系从大型 CAMAC 机架到分散的 GPIB 台式仪器,再发展为 VXI 和 PXI 系统的卡式仪器机箱,又回归成 USB 装置和台式合成仪器,乃至 LXI 网络形式和虚拟仪器结构的技术演变过程。深感自动测试系统与相关标准、规范的紧密联系,以致所做工作常常陷入乏味的规范的研究,所编教材也难免会有看似规范的解释文本。然而,规范要求准确和全面,因此难免冗长烦琐,还要照顾不同观点。且其为了避免争议,文字上不惜重复,并不适于普通读者的入门学习。本书则以经验总结和学习指导为目的,在阐述诸如 GPIB、VXI、RS232、TCP/IP、USB、SCPI、IEEE488.2 和 IEEE1149 等有关内容时,先尽量简明地指出规范要点,然后主要通过示例讨论来说明其基本原理和实现方法,期望能够得到读者的欢迎。

自动测试技术的发展过程中 PC 最具活力,诸如用户界面、分析计算、总线接口、网络通信、数据库和多媒体等虚拟仪器的重要功能无不受益于 PC 的丰富资源。因此本书不仅对诸如频谱分析、数字滤波器、时间序列建模和统计分析等软件算法进行了专门的讨论,而且对自动测试系统和虚拟仪器各主要部分的讨论最终也都归结为 PC 软件的建立。常常通过一段编程代码来表述系统的工作过程,说明程控命令的使用,体现设计的思想。希望读者能够举一反三,在此基础上继续实践和创新。

虚拟仪器说到底还是仪器,因此必然涉及硬件测试电路的设计。由于仪器的大多数功能已被 PC 软件所取代,所需开发的硬件电路只是最核心、最必要的测试功能,故其实现的困难也会相对减少。考虑最典型的仪器功能就是数据采集器,因此本书将深入讨论其基本组成结构、核心采样量化元件的原理和特点、主要通道电路(前置放大器、隔离器、滤波器、通道开关等)的设计方法,以及微控制器总线、监控方式和实时操作系统应用等实际问题。

本书选择 PCB 电子功能模件作为自动测试的典型应用领域,全面阐述 ATE 的路内测试、功能测试、总线测试、边界扫描测试的基本原理和实现方法。同时,也以大型旋转机械的状态监测与故障诊断和小型心电监视为例,分别说明自动测试系统和虚拟仪器的广泛应用前景。

本书涉及的内容较广,均在自动测试与虚拟仪器这一主题下体现各自的作用,展示了实际工程技术中如何运用基础知识解决实际问题的方法和规律。本书也将有关参考文献分列在各章之后,以便读者对照使用。

鉴于作者的工作范围和知识水平所限,书中存在缺点和错误在所难免,恳切欢迎读者的批评指正。

# 目　　录


**第 1 章　绪论** …… 1
1.1　自动测试系统的基本概况 …… 1
1.1.1　自动测试系统和通用接口的基本概念 …… 1
1.1.2　GPIB 接口的基本概况 …… 2
1.1.3　VXI 仪用总线的基本概况 …… 4
1.1.4　VPP 软件规范概况 …… 5
1.2　虚拟仪器的基本概念 …… 6
1.2.1　虚拟仪器的主要功能 …… 6
1.2.2　USB 接口的基本概况 …… 7
1.2.3　虚拟仪器软件开发工具 …… 9
1.3　自动测试技术的发展和本书的主要内容 …… 11
1.3.1　自动测试系统的发展过程 …… 11
1.3.2　PXI 和 LXI 总线系统 …… 12
1.3.3　自动测试的应用领域 …… 13
1.3.4　自动测试技术的发展规律 …… 14
1.3.5　测试工程师的责任和本书的主要内容 …… 16
本章小结 …… 17
参考文献 …… 17
**第 2 章　数据采集器** …… 18
2.1　数据采集器的基本概念 …… 18
2.1.1　数据采集器的基本结构 …… 18
2.1.2　A/D 转换器的量化功能 …… 19
2.1.3　A/D 转换器的基本工作原理和控制方式 …… 20
2.1.4　采样保持器的工作原理 …… 23
2.1.5　A/D 转换器的主要性能指标和选择原则 …… 24
2.2　数据采集器的通道调理电路 …… 25
2.2.1　缓冲放大器与仪器放大器 …… 26
2.2.2　隔离放大器 …… 27
2.2.3　滤波器的基本概念 …… 31
2.2.4　滤波器电路设计的归一化实现 …… 33
2.2.5　Bach 滤波器的设计方法 …… 35
2.2.6　开关电容滤波器 SCF …… 36
2.2.7　多路通道开关 …… 39

2.3 数据采集器的操作控制 …… 42
2.3.1 典型的嵌入式微控制器 …… 42
2.3.2 接口芯片的总线操作 …… 43
2.3.3 $I^2C$ 总线的数据传输 …… 44
2.3.4 SPI 接口的操作 …… 46
2.3.5 LVDS 总线差分接口 …… 48
2.3.6 数据采集器定时采样的控制方式 …… 48
2.3.7 数据采集器中的实时操作系统 …… 50
2.4 数据采集过程中的采样问题讨论 …… 52
2.4.1 采样定理和混叠现象 …… 52
2.4.2 抗混叠滤波器及其有关参数的选择 …… 54
2.4.3 周期信号的 *M/N* 采样方法 …… 57
本章小结 …… 58
参考文献 …… 58
**第 3 章 GPIB 通用接口总线** …… 60
3.1 GPIB 通用接口总线的基本原理 …… 60
3.1.1 GPIB 通用接口总线的基本概念 …… 60
3.1.2 GPIB 规范的要点说明 …… 62
3.2 可程控仪器的 GPIB 接口开发 …… 68
3.2.1 可程控仪器的 GPIB 接口电路 …… 68
3.2.2 接口芯片的功能寄存器 …… 70
3.2.3 可程控仪器的基本工作流程 …… 75
3.3 GPIB 总线控制器 …… 79
3.3.1 GPIB 总线控制器的工作原理 …… 79
3.3.2 总线控制器命令函数的设计和使用 …… 82
3.3.3 基于 GPIB 接口的微机光谱分析系统 …… 85
3.4 IEEE488.2 和 SCPI 规范要点 …… 89
3.4.1 ANSI/IEEE488.2 规范的要点介绍 …… 89
3.4.2 可程控仪器标准命令 SCPI 规范的要点说明 …… 93
3.4.3 可程控仪器编程控制的示例 …… 94
本章小结 …… 97
参考文献 …… 97
**第 4 章 VXI 仪用总线系统** …… 99
4.1 VXI 总线的基本概念 …… 99
4.1.1 VME 总线的基本结构特点 …… 99
4.1.2 针对测试扩展的 VXI 总线结构 …… 102
4.1.3 VXI 机箱和模件的机械结构 …… 103
4.1.4 VXI 总线的电气特性和电磁兼容性要求 …… 104
4.1.5 VXI 总线的系统配置 …… 105

4.2 VXI 总线系统的工作原理 …… 107
4.2.1 逻辑器件和地址空间 …… 107
4.2.2 主模块与从模块的数据传输方式 …… 108
4.2.3 VXI 器件的接口寄存器组 …… 109
4.2.4 消息基器件之间的通信协议 …… 110
4.2.5 VXI 器件的初始化过程 …… 111
4.2.6 VXI 资源管理器与 0 槽控制器 …… 112
4.3 VXI 系统主机的工作方式和应用示例 …… 113
4.3.1 GPIB/VXI 的总线控制方式 …… 113
4.3.2 嵌入式主机的 VXI 系统工作方式 …… 116
4.3.3 仪器驱动程序 …… 117
4.3.4 用于地震模拟试验的 VXI 自动测试系统 …… 118
4.4 VXI 总线即插即用规范 …… 119
4.4.1 自下而上的 VISA 结构 …… 120
4.4.2 VISA 的 I/O 控制函数集 …… 121
4.4.3 仪器驱动程序的接口模型 …… 123
4.4.4 VPP 软面板 …… 124
4.4.5 VISA 资源描述和资源管理器 …… 124
4.5 软件开发平台 LabWindows/CVI …… 126
4.5.1 编程环境和编程方式 …… 126
4.5.2 编程示例 …… 129
4.5.3 IVI 可互换仪器驱动程序的概念 …… 132
4.5.4 独立发布软件的生成 …… 132
本章小结 …… 133
参考文献 …… 134
**第 5 章 测试系统的远程操作** …… 135
5.1 串行通信和 RS-232 协议 …… 135
5.1.1 串行通信的基本概念和分类 …… 135
5.1.2 异步串行通信的 UART 协议 …… 136
5.1.3 RS-232 接口标准 …… 136
5.1.4 信号电平的转换 …… 139
5.2 RS-232 对调制解调器的控制 …… 139
5.2.1 公用电话系统的通信过程 …… 139
5.2.2 信号调制技术 …… 140
5.2.3 调制解调器及其控制方式 …… 141
5.2.4 智能调制解调器的管理 …… 142
5.3 自动测试系统中 PC 主机的串口操作 …… 144
5.3.1 微机之间的 RS-232 串口连接方法 …… 145

5.3.2 20mA 电流环接口与现场总线 485 接口 …… 146
5.3.3 Windows 环境下 PC 串口的 VB 编程 …… 147
5.3.4 XMODEM 通信协议 …… 148
5.3.5 循环冗余校验码 CRC 算法 …… 149
5.3.6 强震观测系统远程监控的实例分析 …… 152
5.4 基于 TCP/IP 的远程测控功能 …… 155
5.4.1 TCP/IP 网络通信协议 …… 155
5.4.2 测控系统对网络通信的基本要求 …… 156
5.4.3 远程实时监控的端口操作实现方法 …… 158
5.4.4 数据传输的电子邮件实现方法 …… 160
5.4.5 文件传输的 FTP 实现方法 …… 161
5.4.6 远程操作的工作示例 …… 162
5.4.7 嵌入式系统中的网络接口与应用方法 …… 162
5.5 用于远程测试的无线通信 …… 165
5.5.1 GSM 和 SMS 手机短信业务的基本概况 …… 165
5.5.2 GSM 模块的应用方法 …… 166
5.5.3 串口编程操作的要点 …… 167
5.5.4 无线通信网络 ZigBee …… 168
本章小结 …… 171
参考文献 …… 171
**第 6 章 基于 USB 的虚拟仪器** …… 173
6.1 USB 的数据通信原理和协议要点 …… 173
6.1.1 USB 系统底层的树形结构 …… 173
6.1.2 差分信号与总线状态 …… 174
6.1.3 传输事务和信息包的基本结构 …… 175
6.1.4 数据传输的帧结构和传输类型 …… 176
6.1.5 控制传输的工作过程 …… 178
6.1.6 USB 系统工作的两个阶段 …… 179
6.1.7 USB 系统的开发 …… 180
6.2 USB 设备接口控制器的开发 …… 181
6.2.1 USB 通用接口模块的基本结构 …… 181
6.2.2 核心接口芯片 USBN9604 的操作原理 …… 183
6.2.3 USB 设备接口控制的程序固件 …… 185
6.2.4 通用接口模块的使用方法 …… 187
6.3 USB 主机操作软件的开发 …… 188
6.3.1 USB 设备驱动程序 …… 189
6.3.2 人机接口设备 HID 协议 …… 189

6.3.3 在 VB 中调用 Windows 的 API 函数 …… 190
6.3.4 在 PC 主机中建立支持 USB 设备的通用方式 …… 192
6.3.5 USB 操作类函数的实现方法 …… 193
6.4 基于 USB 的信源/数采装置 …… 195
6.4.1 信源/数采设备的电路设计 …… 195
6.4.2 设备控制固件程序的建立 …… 197
6.4.3 不同工作模式的虚拟仪器应用 …… 198
6.5 支持 USB 接口的 50MHz 高速数据采集器 …… 200
6.5.1 高速数据采集器的系统结构 …… 200
6.5.2 核心器件的原理和时序 …… 201
6.5.3 控制电路的 CPLD 实现 …… 202
6.5.4 USB 接口和虚拟仪器模式 …… 204
6.6 高速数据传输和同步问题的探讨 …… 205
6.6.1 影响 USB2.0 数据传输速度的因素分析 …… 205
6.6.2 USB2.0 设备控制器的高速工作模式 …… 206
6.6.3 建立 UXI 测试总线的构想 …… 208
本章小结 …… 210
参考文献 …… 211
**第 7 章 PCB 电子功能模件的自动测试** …… 212
7.1 PCB 电子功能模件测试的基本概念 …… 212
7.1.1 PCB 电子功能模件测试的基本方法 …… 212
7.1.2 传统 ATE 设备的基本情况 …… 213
7.1.3 针床和夹具结构 …… 215
7.1.4 PCB 功能模件测试要求的新动向 …… 216
7.2 模拟电路的路内测试 …… 217
7.2.1 PCB 模拟器件路内测试的基本原理 …… 217
7.2.2 阻抗测试的相关分析法 …… 218
7.2.3 保护地的概念和三线法路内测试 …… 219
7.2.4 三线法路内测试的误差分析 …… 220
7.2.5 六线测试的原理 …… 221
7.2.6 非线性器件的测试方法 …… 223
7.2.7 虚拟仪器的实现方案 …… 223
7.2.8 通用可程控仪器的使用 …… 225
7.3 PCB 模件的功能测试 …… 227
7.3.1 测试设备的设计思想 …… 227
7.3.2 面向微处理器的总线测试 …… 228
7.3.3 逻辑分析功能的实现方法 …… 229

7.3.4 特征分析功能及其软件实现 …… 232
7.3.5 功能测试系统的主操作界面 …… 234
7.4 边界扫描测试与 IEEE1149 规范 …… 236
7.4.1 边界扫描测试的基本原理和基本单元结构 …… 236
7.4.2 边界扫描测试控制器及其访问端口 …… 237
7.4.3 TAP 控制器的状态控制 …… 238
7.4.4 边界扫描测试的操作指令及工作方式 …… 239
7.4.5 边界扫描的描述语言 BSDL …… 242
7.4.6 边界扫描在板级测试中的连接方式 …… 243
7.4.7 边界扫描测试系统的虚拟仪器示例 …… 244
7.4.8 边界扫描虚拟仪器的上层软件 …… 247
7.4.9 混合信号测试及总线标准 …… 248
本章小结 …… 252
参考文献 …… 252
**第 8 章 典型软件算法和典型应用系统** …… 254
8.1 频谱分析 …… 254
8.1.1 频谱分析的基本概念 …… 254
8.1.2 离散傅里叶变换 …… 256
8.1.3 应用 DFT 进行频谱分析 …… 256
8.1.4 FFT 的基本原理和程序实现 …… 258
8.2 数字滤波器设计 …… 260
8.2.1 数字滤波器的等效离散化设计方法 …… 260
8.2.2 数字滤波器的 FIR 和 IIR 形式 …… 262
8.2.3 典型一阶 IIR 滤波器的参数确定 …… 262
8.2.4 四点加权的 FIR 数字滤波器 …… 263
8.2.5 数字滤波器的使用 …… 264
8.3 时间序列的建模分析 …… 265
8.3.1 最小二乘的参数估计算法(LS 算法) …… 265
8.3.2 时间序列的 ARMAX 模型 …… 268
8.3.3 最优激励信号的选择 …… 269
8.4 基于数理统计的可信度分析方法 …… 271
8.4.1 最小二乘估计参数的分布情况 …… 271
8.4.2 方差的估计 …… 273
8.4.3 估计参数的置信区间 …… 274
8.5 大型旋转机械的状态监测与故障诊断 …… 276
8.5.1 200MW 和 300MW 汽轮发电机组转子的状态监测 …… 277
8.5.2 传感器和监测仪表的配置 …… 279

8.5.3 数据采集器的工作特点 …… 281
8.5.4 计算机监测分析软件 …… 282
8.5.5 故障诊断的模糊推理与专家系统简介 …… 285
8.6 用于心电监测的虚拟仪器开发 …… 288
8.6.1 心电测试原理 …… 288
8.6.2 虚拟仪器的结构设计 …… 289
8.6.3 基于 SOC 的 USB 设备控制器 …… 290
8.6.4 前置放大电路的构建 …… 290
8.6.5 PC 应用软件的开发 …… 293
本章小结 …… 295
参考文献 …… 295
**参考复习题** …… 296

# 第1章 绪 论

科技的进步和社会的发展对测试技术提出了更多、更新的要求，眼观手记、人工操作的传统仪器在很多场合已被计算机管理的自动测试所取代。同时面对各行各业、各种各样的测试任务，人们不再可能都去为之研制相应的专用测试仪器了。于是，一方面由多台通用设备组成，在计算机的统一指挥下，共同完成一项测试任务的自动测试系统应运而生；另一方面，由于计算机技术和数字信号处理技术的迅速发展，使得已经提出多年的“虚拟仪器”理念正在成为现实。本章将首先围绕自动测试系统和虚拟仪器的基本概念进行讨论，阐述计算机在自动测试技术中的核心作用。进而探讨自动测试技术的基本规律，说明其发展的历史过程、应用现状和发展方向，以及测试工程师的主要任务和本书的主要内容。

## 1.1 自动测试系统的基本概况

### 1.1.1 自动测试系统和通用接口的基本概念

与传统的专用测试仪器不同，自动测试系统强调在计算机的控制下，由若干可程控的通用设备共同完成测试任务。这里最常用的通用设备是各种形式的可程控信号源、可程控数据采集器(包括数字表、示波器、频谱分析仪等)、可程控多路通道开关和可程控输出设备(打印机、绘图仪等)。

如图 1-1 所示，这是一个用于检查放大器线性度的的自动测试系统。测试激励信号由程控信号源发出，接入被测设备的输入端。程控数字万用表在多路通道开关的切换作用下，依次测量被测设备的输入端和输出端，然后用计算机算出结果(增益系数)，再由程控绘图仪画出误差关系曲线。这里采用比较法，同时测量被测设备的输入和输出是为了充分发挥数字万用表的精度优势，从而降低对信号源本身精度的要求。注意，尽管该系统中的程控信号源、数字万用表、多路通道开关、绘图仪和微计算机仍然分别起到激励、测量、通道、记录和数据处理的传统作用，然而当它们之间能够在一个系统控制器(通常为一台 PC)的指挥下实现相互通信后，系统就形成一个能够完成新任务的有机整体。

设备之间的通信需要协议，要对通信网络的构型、信号传递的方式、接口的电气/机械和功能特性做出明确的规定。然而在通用接口技术出现以前，这一切工作都需要系统的组建者自己完成。系统组建者要为系统中各组成设备设计专用的接口来适应计算机控制的要求，为此往往需要较高的技术和较长的时间，以致接口问题一度成为组建自动测试系统的主要障碍。于是人们设想建立一种通用的标准化接口，试图让全世界所有的厂家都按照同一种标准设计可程控设备的接口。这样，当用户组建自己的测试系统时，只需用一根标准的无源电缆

将带有这种接口的设备与计算机连接起来。

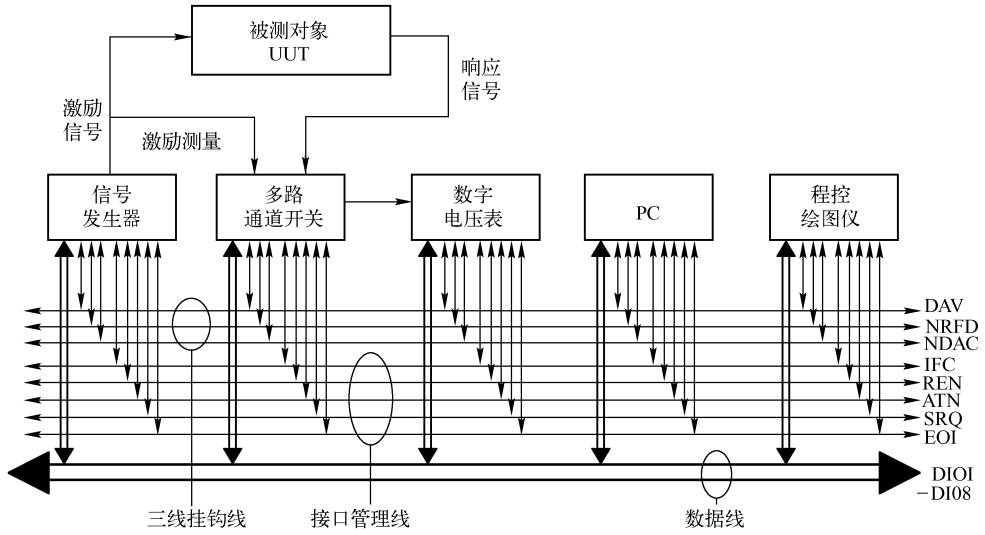



图 1-1　典型自动测试系统的示意框图

接口系统的这一革命是由当时美国的 HP 公司率先提出的，因其基本构型为总线式结构，故称 HPIB（Hewlett-Packard Interface Bus）。这项技术在欧洲推广后，又被称为 GPIB（General Purpose Interface Bus），即通用目的接口总线 。美国电气与电子工程师学会（IEEE）在 1975 年通过了正式文件，将其命名为 IEEE488，国际电工委员会（IEC）表决后形成 IEC-625 文件。从此这项技术就成为组建自动测试系统的国际标准，目前国际上几乎所有的仪器厂家都为自己的仪器配上了这种通用接口。以这种接口总线为基础组建自动测试系统，人们可以不分厂家、国别，只是根据测试的需要在全世界范围选用最合适的设备，然后直接用一条标准电缆将它们依次连接起来，就可以编写测试软件，灵活地完成各种复杂的测试任务。

### 1.1.2　GPIB 接口的基本概况

GPIB 接口技术的实质是为自动测试系统中各设备之间相互通信建立一个协议。在 GPIB 总线的 16 根信号线中，有 8 根信号线作为数据线，以串行字节的方式传递信息，被称为多线消息。而另外 8 根信号线各起各的作用，被称为单线消息。其中 3 根线为握手线（HandShake），即在每一次多线消息传递的过程中，都要通过它们做一次“三线握手”的工作，以确保多线消息准确无误、毫不含糊地完成异步通信的任务。每个 GPIB 设备使用一个 0-30 的地址作为设备的基本标识方法，这个地址通常由设备后面板上一个按 8421 编码的指形拨动开关设置。从图 1-2 给出的某 GPIB 仪器的后面板中，可以看到 GPIB 接口插座的样式和地址开关的形式。

GPIB 系统通常包括若干具有听、讲接口功能的设备，作为控制器的计算机还需要接口支持控者功能（也称控功能）。其典型的工作过程是由控制器的系统控者功能初始化总线上所有的设备，并使它们进入“远地控制”状态，由控制器的责任控者功能任命总线的一个仪器设备为讲者，另一个或另几个仪器设备为听者，然后使进入讲者作用状态和听者作用状态的设备在“三线握手”的保证下，一个字节一个字节地通过数据总线传递仪器消息。目前，大多数

GPIB 测试系统的控制器是在 PC 中配置必要的 GPIB 接口卡和相应软件实现的，通过总线中的一条专用信号线的单线状态将其多线消息的命令分为接口消息和器件消息，前者可以用来配置接口功能，如任命讲者、听者，后者则可具体实现测试数据的传输。

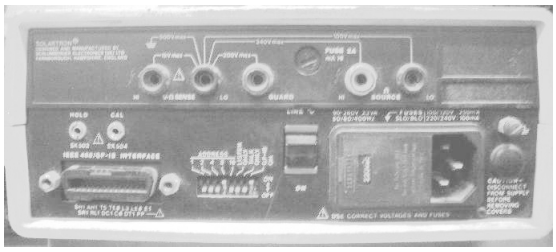
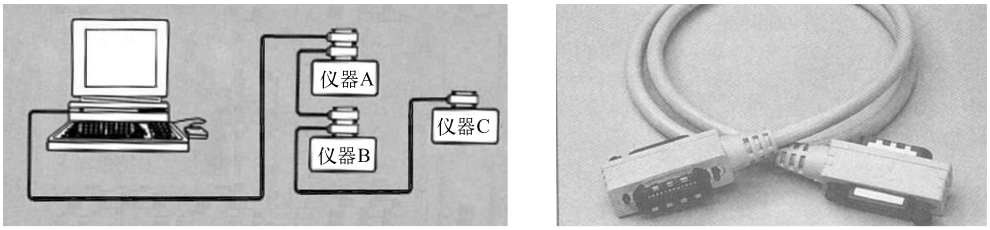


图 1-2　GPIB 设备后面板中的接口插座和地址开关

GPIB 接口采用一种猪脊型的 24 芯专用连接器，对于 16 根信号线以外的其他连接线被用做接地线和屏蔽绞线。连接器装在连接电缆的两端，由一个插头和一个插座背靠背地组装而成。因此当把其插头插入仪器的接口插座时，也同时自然地将另一个插座提供给其他连接电缆。图 1-3 给出了连接器的外形和连接形式。




(a) 系统连接方法　　(b) 连接器与连接电缆

图 1-3　GPIB 总线系统的连接

为了确保通信的可靠性，同时使连接在 GPIB 总线上的仪器被限制在 15 台以下，通信距离限制在 20m 内，目前通信速度的上限可达 1MB/s。

从 IEEE 于 1975 年、IEC 于 1977 年公布有关 GPIB 的标准以后，经过十几年的实际推广，得到了全世界范围各国主要仪器制造业的认同和响应。以 GPIB 总线接口为基础建立的测试系统遍布各行各业的各个领域，这种发展给广大测试工作者组建系统带来极大的方便，也使他们逐步熟悉了 GPIB 技术，从而提出了更高的要求。这些要求集中表现在对编码、格式、通信协议、程控命令方面的标准化。在这种背景下，IEEE 在 1989 年公布的 IEEE488.1 和 IEEE488.2 两个文件以及 SCPI 规范，及时地解决了这一问题。IEEE488.1 的名称是“对于可程控仪器的标准数字接口”，IEEE488.2 的名称是“使用可程控仪器 IEEE 标准数字接口时的编码、格式、规程和通用指令”。前者的内容与 IEEE488 的内容基本相同，仍为通用接口在电气、机械和功能上的规定；后者则主要针对仪器消息的传递，对消息交换的控制操作、消息描述的语法、通用指令的设置、通用状态的描述、同步执行的实现和仪器地址的自动组态等方面，在原先文件的基础上进行了较大的修订和扩充。同时 SCPI 还对各类仪器的程控命令做了详尽的规定，从而使新的 GPIB 系统不仅在接口功能的工作方式上得到了统一，而且在仪器控制的工作方式上也得到了统一。

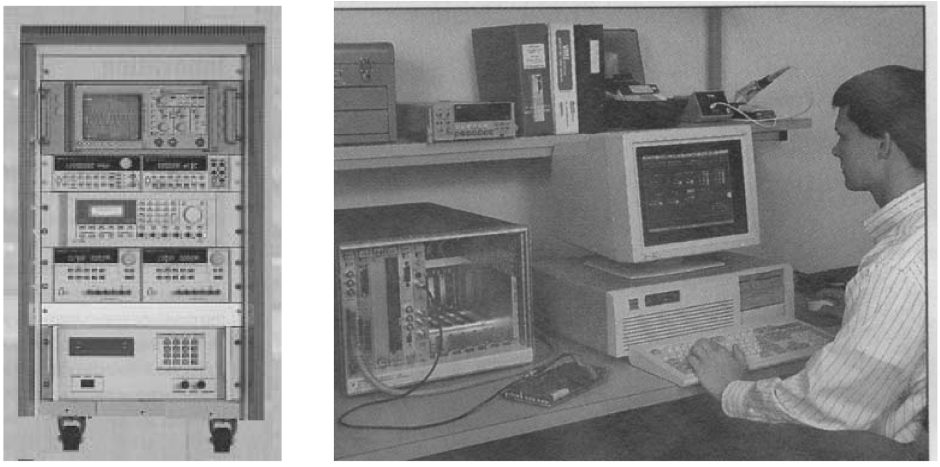
GPIB 接口总线技术是相当成功的，30 年经久不衰，至今仍在自动测试领域发挥着重要的作用。但该技术也存在两个比较明显的缺点：其一是受 8 位数据线的限制，在一些场合显得传输速率不够快；其二是系统由分开的台式仪器设备组成，占用了较大的工作空间。

### 1.1.3 VXI 仪用总线的基本概况

针对 GPIB 在工作速度和占用空间等性能上的限制，特别是军事上对测试系统的更高要求，促使人们开始进一步的探讨。1987 年，惠普、泰克、雷卡、Colorado Data 和 Wavetek 等 5 家著名测试和仪器公司的工程技术代表组成了一个联合体，推出一种在 VME 总线基础上扩展而成的模件化仪器的公用系统结构，命名为 VXI。经过几度修改后，于 1992 年被美国电气与电子工程师学会批准，成为 IEEE1155 标准。从此符合 VXI 总线规范标准的仪器及系统被称为 VXI 仪器及 VXI 仪器系统。1993 年又进一步制定了 VXI 即插即用(VXI Plug & Play，VPP)标准，并成立了 VPP 系统联盟。1996 年，VXI 总线系统实现了全球标准化，VXI 市场也形成了一定的规模。VXI 模件和系统的销售额开始以每年递增 30%～40%的高速度发展，这种情况远远超过了同时期其他仪器的增长率。

VXI 总线是 VMEbus eXtensions for Instrumentation 的缩写，即 VME 总线在仪器领域的扩展。VME 总线是一种在工业控制领域中获得广泛应用的高性能微机总线，其 32 位字长、高达 40MB/s 的传输速率和多种数据传输协议显然能够为高速、高精度、高可靠性的现代测试任务提供很好的硬件基础。同时，针对自动测试系统仪器模件之间同步、触发以及电磁兼容和供电电源的特殊要求，VXI 增加了 10MHz 和 100MHz 的时钟线、用于判别卡式仪器的模件识别线、8 条并行的 TTL 触发线和 2～6 条并行的 ECL 触发线、用于模块间准确定时的“星形触发”线、50 欧终端模拟相加线、12～36 条连接相邻模件的本地总线，并提供了±2V、+5V、±12V、±24V 和−5.2V 的完整电源功能。

如图 1−4 所示，VXI 在结构上与 GPIB 台式仪器及叠层机架方式不同，VXI 采用卡式仪器配置机箱的方式。完整的 VXI 测试系统由一个或多个 VXI 子系统组成，每个子系统由配置了电源、0 槽控制器和有关卡式仪器模件的 VXI 机箱组成。VXI 总线仪器没有传统意义的操



(a) GPIB台式仪器　　(b) VXI卡式仪器机箱

图 1−4　VXI 系统的典型结构

作面板，对 VXI 仪器的操作与显示，通常都需借助 PC 主机完成。VXI 机箱与 PC 主机之间的数据通信可采用不同方式，如 IEEE488、RS-232C、IEEE1394、USB 等标准总线，以及诸如 MXI 等专用总线。系统组成包括主计算机、机箱、卡式仪器模件、软件等。

VXI 总线的主要特点表现在以下几个方面：

（1）性能优越。系统传输速率快，VXI 总线背板数据传输速率达 40MB/s 以上，增扩的本地总线则达 100MB/s。而且不同等级器件优先权中断的使用，更能高效地利用数据总线。VXI 总线采用公用电源、公用冷却和高效的 CPU 管理，结构设计紧凑，缩小了测试系统的体积。机箱及模件设计遵循军用标准，一般模件的平均无故障时间在 $10^4$h 以上。

（2）系统配置灵活方便。国际上有几百个厂家共生产配套产品几千种，数字多用表、信号源、示波器等传统仪器均有相应的 VXI 总线模块化产品。机箱可通过 MXI 扩充，并可灵活选用外置 PC 或嵌入式主机，容易满足测试系统的更新要求。VXI 系统采用虚拟仪器面板，易于满足用户的不同要求。

（3）软件标准化程度高、开发环境好。VXI 总线系统不但可以兼容 IEEE488.2 的可程控仪器接口操作方式和 SCPI 的标准程控命令，而且还可直接使用诸如语言编程开发软件 LabWindows/CVI 和图形编程的开发软件 VEE、Labview 等软件开发工具。

（4）系统维护简单、升级容易。自检后可及时发现故障模件，故可自行处理。用 VXI 组建较小系统时最初投资可能较高，但扩展系统功能时所增模件的投入要比通常增添传统仪器的开销少得多。

### 1.1.4　VPP 软件规范概况

为了使 VXI 仪器易于使用，并在系统级上使 VXI 总线系统成为一个真正开放的系统结构，1992 年 9 月，NI、泰克等公司组成了一个开放式测试系统联盟组织（OMS），着手解决虚拟仪器软件结构统一定义与规范的问题。1993 年 9 月，在 OMS 组织的基础上，NI、泰克、雷卡等 5 家公司联合成立了 VXI 总线即插即用（VPP）系统联盟，通过定义和推行一些标准化准则和操作规程，来解决 VXI 总线规范中尚未包含的系统级问题。VPP 系统联盟内部由生产厂家联合会与用户协会两大部分组成，生产厂家联合会成员共同承担组织费用，而用户则可免费申请加入。联盟通过 Internet 网络论坛、E-mail 通信与学术研讨会等多种形式，进行系统开发与应用的经验交流与意见反馈，进一步维护和完善 VPP 规范。

VPP 规范的目的是为了解决不同生产厂家的 VXI 产品在系统中可能出现的易操作性与互操作性问题，并为最终用户提供进行系统维护与再开发的能力。VPP 不仅着眼于 VXI 仪器硬件模块与软件模块的设计，更注重于结构化、模块化的虚拟仪器系统设计。系统中总线的 I/O 接口控制软件、仪器驱动程序与应用程序自下而上地构成了虚拟仪器系统的软件结构：I/O 接口软件驻留在主计算机中，作为仪器驱动程序的底层函数，支持对总线的操作；每个仪器模件都有自己的仪器驱动程序，用于完成特定仪器的各种功能，它是应用程序实现仪器控制的桥梁；系统应用程序直接面对用户，通过提供直观友好的测控操作界面、丰富的数据分析与处理功能，完成各种自动测试任务。VPP 规范强调了用户在仪器系统开发与维护过程中的重要作用，VXI 即插即用规范不仅对仪器生产厂家开放，也对用户开放，它不仅是虚拟仪器系统的设计指导规范，也是虚拟仪器系统的应用指导规范。VPP 系统的核心是一个统一的 I/O 接口软件（VISA）规范，不仅适用于 VXI 卡式仪器，也适用于 GPIB 台式仪器、RS-232

串行接口支持的多种设备。这样，就为不同接口在同一平台中运行提供了统一的基础，在VISA 基础上编写的仪器驱动程序和软面板程序也都成为统一格式的标准软件模块。

## 1.2 虚拟仪器的基本概念

### 1.2.1 虚拟仪器的主要功能

什么是虚拟仪器？有人说："虚拟仪器就是计算机。"也有人说："虚拟仪器就是软件。"在我们看来，所谓虚拟仪器，就是要尽可能地采用通用计算机及其软件的功能来取代原先专用仪器的设备功能。对虚拟仪器来说，仪器与计算机之间的界限模糊了。由于充分发挥了通用计算机的功能，节省了大量的专用硬件设备和软件开发成本，从而成为自动测试技术的一个重要发展方向。

注意，这里强调的是通用计算机及其软件。因为即使是传统的仪器设备，实际上也都含有一个、甚至多个微处理器，也都需要各种微处理器程序软件(有时称为固件)的支持。然而，那样的计算机和软件是专用的，是在特定用途下专门开发并实现特定功能的仪器设备，是不能兼做他用的。因此从整体上看，它们不是虚拟仪器，而是传统意义下实实在在的仪器设备。

那么，什么是仪器设备的基本功能呢？又有哪些仪器的功能可以被通用计算机及其软件实现呢？

首先，作为传统的仪器设备，需要面板、机箱和电源。这里，面板用于人机交互，其显示功能和操作功能分别与人的眼和手作用相当。常见的面板显示元件包括指示灯、发光二极管、数码管、指针仪表、字符及图形显示器等；常用的操作元件则包括各种按钮、按键、扳键、旋钮、拉杆、多路转换开关等。而在虚拟仪器系统中，借助于通用的 PC 和 Windows 操作系统下强大的图形功能，这些功能可以得到更好和更方便的实现。在图 1-5 中，可以看到这种随心所欲的面板安排。

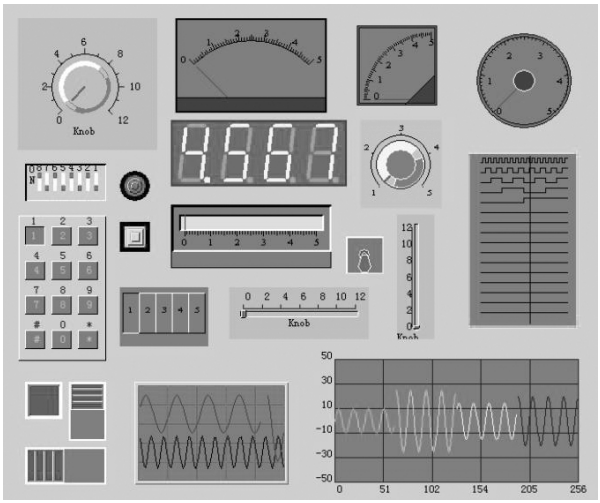



图 1-5 用于人机对话的虚拟仪器控件

其次，作为现代仪器，不可避免地需要各种数据处理功能。例如，需要产生诸如脉冲、阶跃、斜波、方波、正弦波、三角波、随机波等激励信号，对测量结果则往往需要进行频谱分析、小波分析和包括高通、低通、带通和带阻等功能的各种数字滤波，也需要实现包括相关分析、回归分析、方差分析等统计功能，以及曲线拟合、插值、积分、微分、卷积、求逆、分解等数值计算。显然，通用计算机进行这些工作的能力要比传统仪器中的微处理器强得多。

再次，近年来许多高档仪器在原先专门仪器的基础上加入了一些与信息技术相关的辅助功能，如语音报数、提示及报警的音响功能，图文并茂的操作指导功能，测试数据转储和网络通信的功能等。很明显，通用 PC 强大的多媒体功能、数据库功能、电子表格功能、TCP/IP 网络通信功能够更方便和更圆满地处理这类问题。

最后，也是最为重要的仪器功能是其本身的输出激励信号和测量输入信号的功能。其核心是不同字长分辨率、不同转换速度和不同抗干扰能力的模数转换器和数模转换器、相应的通道调理电路，以及专门的传感器设备。显然，通用计算机通常是不具备这些功能的。因此，要么需要在计算机内加装专门为此设计的功能插件板，要么需要通过某种接口设备与那些具有这种基本测试功能的设备连接起来。

在当今国际市场上，GPIB 通用接口总线系统和 VXI 仪用总线系统是两种比较常见的选择。前者建立在 GPIB 接口已为所有仪器厂家接受的基础上，以至于用户可以在世界范围内不分厂家国别，从测试的角度出发选择最理想的台式仪器来组建系统。后者以 VXI 模件的形式出现在专用的 VXI 机箱中，仪器之间的总线连接是通过背板实现的，因此极大地提高了自动测试系统的工作速度。同时取消了传统面板，共享了机箱电源，从而减少了公用开销和所占空间。然而，由于这两类系统的成本较高，在很大程度上限制了它们在民用和中小企业中的应用。对于计算机内加装插件卡的虚拟仪器系统，其优势在于它的价位普遍较低，因而引起很多用户的注意。但是也确实存在不少问题，如加卡、换卡需要打开机箱，设备独立性差；机箱内部结构限制及后面板出线方式容易引起电磁兼容性问题；台式 PC 内部总线升级过快，难以稳定标准；笔记本 PC 空间条件下仪器的工作环境更差等。因此，人们一直希望找到一种更适于虚拟仪器系统使用的接口总线。现在看来，当前广泛用于 PC 主机与其外设相互连接的通用串行总线 USB 就是一种不错的选择。

### 1.2.2　USB 接口的基本概况

通用串行总线 USB(Universal Serial Bus)最初是由 Intel、DEC、IBM、康柏、微软、NEC、北方电信等 7 家著名计算机和通信公司共同推出的新一代接口标准，其目的在于解决为数众多的 PC 外设与有限的 PC 插槽之间日益增加的矛盾，于是设想用一条 USB 总线代替 PC 主板上多种复杂的接口。现在，所有 PC 都已配备了至少两个 USB 接口，Windows 98、Windows 2000 和 Windows XP 操作系统中也都提供了支持 USB 通信的驱动程序。特别是后来 USB 组织又增加了惠普、朗讯和飞利浦 3 个重要的新成员，从而为建立基于 USB 技术的自动测试系统奠定了基础。

USB 总线设备是如何使用呢？它又有哪些基本特点呢？

USB 接口采用精心设计的 4 芯连结器，其中除了两侧的地线和电源线外，中间两根信号线采用一种特别的差分电平编码方式来实现数字信号的高速传输。传输通过 3~5m 的总线电缆将 USB 设备放到远离 PC、靠近被测对象的位置，从而可以提高系统的测试精度和电磁兼

容性。如图 1-6 所示，USB 系统在主机内部提供一个根集线器，其后可以直接连接到各个 USB 设备，也可通过一级、一级的集线器以树形方式扩充连接到多达 127 个设备。

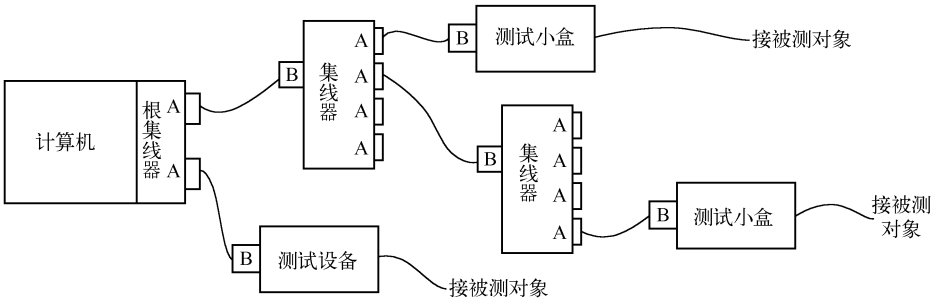



图 1-6　USB 虚拟仪器系统的树形连接结构

图 1-7 为在一台笔记本电脑上连接 USB 集线器和 USB 数采设备的典型情况。这台笔记本电脑有两个 USB 外接口，其中一个直接连接到一个 USB 鼠标，而另一个则连接到一个 USB 集线器，然后通过这个集线器再分别连接到扫描仪和一个数据采集器模块。

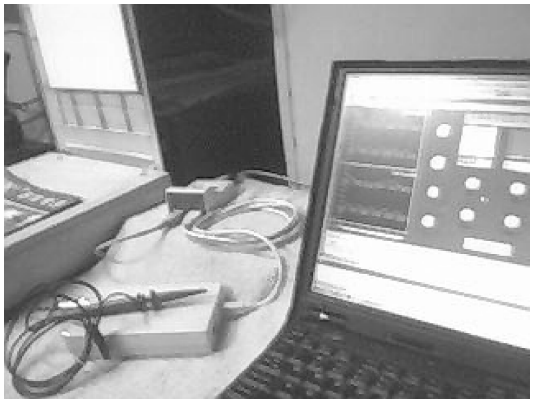


图 1-7　USB 数据采集器实物照片

USB 接口的连接可以带电操作(热插拔)，并且具有即插即用的功能。这种带电操作和即插即用的功能对虚拟仪器的工作具有明显的实用意义，因为当代的微机操作系统，诸如 Windows 系统的启动初始化过程十分费时。若无此功能，则对大多数用户来说都将是无法忍受的。

USB 集线器本身能为每个连接到它的设备提供 500mA 电流的 5V 电源，因此对于那些已经配备了根集线器的 PC 可以直接支持一些小型测试设备的工作。

数据传输的快速性也是 USB 的主要特点，在 1998 年推出的协议 USB1.1 中，规定的通信速率达到 12Mbps，而在 USB2.0 规范中，最高的通信速率可以达到 480Mbps。这样的速率大约高于普通串口的 100 倍，足以满足大多数自动测试的实际要求。

作为虚拟仪器系统中测试设备与 PC 主机之间的通信接口技术，USB 的上述特点使其与其他计算机外设接口相比，占有十分明显的优势。表 1-1 就 USB 与微机的其他几种主要接口在人们普遍关心的一些指标上进行了比较。

表 1-1　常用 PC 接口比较

| 接口名称 | 通信方式 | 连接设备数 | 传输距离 | 传输速率 | 典型应用 |
|---|---|---|---|---|---|
| RS-232（串口） | 串行异步 | 点对点 | 15m | 9600~115Kbps | 调制解调器 |
| Centronics（打印口） | 并行 | 点对点 | 3~10m | 8MB/s | 打印机 |
| IrDA(红外) | 串行异步 | 点对点 | 2m | 115Kbps | 打印机 |
| IEEE-488(GP-IB) | 并行 | 15 | 20m | 20K~1MB/s | 可程控仪器 |
| IEEE-1394(火线) | 串行 | 64 | 5m | 400Mbps 以上 | 数码相机 |
| 以太网卡 | 串行 | 1024 | 500m | 10/100Mbps | 计算机之间 |
| RS-485 | 串行异步 | 32 | 1000m | 10Mbps | 工控设备 |
| USB | 串行异步 | 127 | 3~5m | 1.5/12~480Mbps | 计算机外设 |

总之，面向未来的各种测试任务，基于 USB 总线通信的虚拟仪器提供了一种十分理想的工作方式，即将专门处理测试任务的硬件电路集中在一个离开主机、靠近测点的功能小盒(测试设备)中，而将现代仪器中的大多数工作任务留给 PC 主机的软件完成。由于这种方式充分发挥了通用计算机在人机对话、分析计算、多媒体、数据库、网络通信等方面的巨大能力，往往产生事半功倍的效果。

### 1.2.3　虚拟仪器软件开发工具

当今世界上最流行、也是公认比较成熟的自动测试系统软件开发工具，当数美国 NI 公司的 LabWindows/CVI、LabView、ComponentWorks 和惠普公司的 HP VEE 等。

LabWindows/CVI 是一个完整的软件设计平台，具有编辑、编译、连接、调试、求助等功能，它以 C 语言为基础(C for Virtual Instrument)。同时它具有交互式功能，即能够像 BASIC 那样解释性地执行代码，又能交互式地编写程序。当单击库函数菜单的 Liberary 选项时会得到一个库函数的树形索引，沿着树枝往下就会找到用户需要的函数，这时出现的画面可让用户添加函数所需的参数，确认后就会自动生成调用这一函数的代码并加到源程序中，使用户不必记住那些繁杂的函数名称和各种参数的细节含义。从自动测试的角度看，LabWindows/CVI 函数库主要提供了以下几方面的功能：

（1）用户界面的设计，通过在面板上加控件的方法实现。

（2）总线接口控制，实际接口驱动库函数随硬件控制卡提供。

（3）数据分析、计算的函数集合。

（4）通过动态数据交换 DDE 方式同数据库等其他软件联系，也支持 TCP/IP 网络功能。

（5）支持仪器驱动器和仪器类驱动器。

LabView 的面向自动测试的功能与 LabWindows/CVI 相似，但其开发应用系统的方法很不相同。如图 1-8 所示 LabView 的程序不是以文本方式描述，而是所谓的“图形编程”，用户界面上的每一个控制都对应于框图中的一个元件。通过连线把这些元件连接起来，以此表明信号的流程和走向，用户也能加入一些逻辑或算术运算的元件或者滤波器等信号处理的元件，从而使测得的结果能够被分析和计算。此外，对诸如条件分支、循环、跳转等程序控制语句也采用图形编程的方法，即用图形框把判断条件、循环体框起来，再加以不同的特性说

明。有人认为这种编程方法可使从来没有编程经验的用户避免学习编程语言的麻烦，并认为是编程语言发展的趋势。但也有人认为，如果完全没有编程经验，要很好地理解那些条件、循环、嵌套过程以及处理出/入口参数也不比学习一种语言更容易。特别是对大型的应用程序，尽管可以分层设计，但考虑总体设计和分块调试都还存在一些问题。惠普公司推出的HP VEE 同 NI 的 LabView 很相似，它也是一种基于图形的编程语言，它所提供的函数库的功能和种类也同 NI 的软件相似。

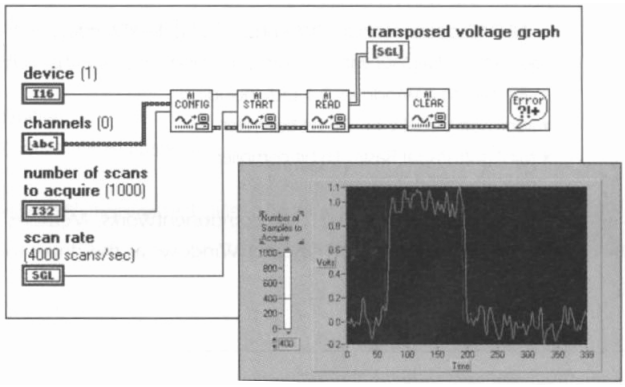



图 1-8　图形编程软件的开发工具

ComponentWorks 是 NI 公司推出的另一种虚拟仪器工具，与上述软件不同，它不是一个独立的集成开发环境，而是提供一系列支持测试功能的控件集合，这些控件可以很方便地加入到 VB 等语言开发环境中，这样用户可以用熟悉的语言来开发自己的自动测试系统。

ATS98 是在国家 95 科技攻关项目支持下建立的一种开发工具，它以 Microsoft Windows 下的 ActiveX 技术为基础，通过标准的 VB 控件、Windows DLL 动态连接库函数和其他 VB 模块，组成方便用户使用的各种虚拟仪器功能，并通过一个编程指导器将其和 VB 开发环境有机地结合起来。ATS98 软件工具的使用分为 3 个层次，首先对从事自动测试工作的工程师，可以利用这种软件工具提供的用户接口虚拟仪器控件迅速建立起测试系统的计算机可控操作界面；利用其总线接口虚拟仪器库函数操作 GPIB 和 VXI 仪器的硬件功能；利用其分析计算虚拟仪器库函数实现数据的处理；同时也可在其虚拟仪器辅助函数的支持下，实现诸如多媒体操作、数据库/电子表格处理、网络通信等 PC 系统功能的自然调用。从而极大地提高了软件开发的效率和程序运行水平。其次，对一般仪器使用者，可以利用这种软件工具的典型虚拟仪器面板，像使用普通仪器那样，在计算机屏幕上，通过操作鼠标器控制诸如数字仪表、示波器、信号发生器、频谱分析仪、逻辑分析仪等虚拟仪器面板，读取测试结果，并可直接存盘、打印、联网，实现资源共享。最后，对专门行业的测试任务，可以利用这种软件工具的上层专业框架功能，建立诸如数据采集、系统辨识、决策处理等以自动测试与信息处理为基础的自动化专业系统。这里，数据采集器是现代测试工作的基础，大多数测试任务都可以利用它来实现；系统辨识机将测试工作由对独立参数的量值测量发展到对动态过程的特性测试；基于知识的决策器则把测试进一步与专家系统结合起来。

## 1.3　自动测试技术的发展和本书的主要内容

### 1.3.1　自动测试系统的发展过程

通常可将自动测试系统的发展过程分为三代。第一代自动测试系统的产生最初只是针对传统专用仪器的单机、单参数的手工测试。随着产品的开发和生产中测试项目和测试范围的不断增加，测试的参数数量、工作速度和精度要求也不断提高，特别是一些要求反复测试、高度可靠的场合，利用计算机技术辅助测试可以严格地执行测试规范的要求，也才有可能定时、重复地完成测试过程，从而形成第一代自动测试系统的基本形式。这类系统最初出现在军用设备的检修维护中，面对大量连接电缆的通/断检查，乏味的人工测试很难保证测试质量。而后用于大型的重要工业生产过程的巡回监测，进而扩展到一般领域和科学实验的数据采集和分析。这类系统对于产品质量的监督和生产过程的安全具有明显的促进作用，从而广泛应用在生产过程中。第一代自动测试系统的结构通常以计算机主机为核心，集中配置各种与测试设备互联的接口装置。这些接口一般需要自行开发专用的电子设备，因此存在开发周期长、研制费用高、工作生命期短的缺点。

第二代自动测试系统的主要特点是采用标准的接口和总线，最具有代表性的是 CAMAC 系统和 GPIB 总线接口系统。CAMAC(Computer Automatic Measurement and Control)系统的规模比较大，用于宇航、核物理、重要的工业自动化等领域中测试项目繁多、测试内容变化频繁的场合。它采用集中形式的标准机箱，配置标准的各种测试插件设备。插件采用 20mm 的标准宽度和 285.25mm 的标准高度。根据不同插件的复杂程度，可以选择不同的单宽和双宽方式。这些测试插件名目齐备，从标准的数据采集器、信号发生器、通路开关，到当时核工业专用的各种测试设备，应有尽有。可以说，CAMAC 系统开创了自动测试标准接口系统的先河，成为将这种技术真正用于实际的成功范例。然而，CAMAC 系统对于更广泛的民用需要常常显得过于复杂和庞大，于是随着微处理器技术的日益普及，应用更为广泛的 GPIB 总线接口系统应运而生。与 CAMAC 系统作为计算机配套设备的形式不同，GPIB 总线接口附属于台式仪器本身。作为国际标准，国际上几乎所有的仪器厂家都为自己的仪器配置了这种通用接口。因此，以这种接口总线为基础组建自动测试系统时除了标准的专用电缆外，并不需要其他额外的设备。GPIB 系统的一个重要特点是：每台仪器仍然保持自己本身的工作方式和功能特性，系统中的计算机只是通过接口总线管理和协调它们的工作。

第三代自动测试系统的基本思想是要进一步发挥计算机的作用，使系统中的主计算机不但作为系统的管理者和控制者，而且成为测试功能的一个主要组成部分。这样，就有可能通过极为有限的测试硬件设备(如信号源、数采器、开关等)，结合计算机的计算功能、存储功能、显示功能、操作功能、通信功能以及其他扩展功能的作用，来代替各式各样的传统测试仪器。这就是虚拟仪器的思想，由于这种思想已经并正在成为现实，从而明显地改变着仪器仪表行业的面貌。

VXI 可以看做第三代自动测试系统总线的代表。作为国际标准，VXI 继承了 GPIB 可以不分厂家、国别地在世界范围选择适当测试设备的重要特点。同时，采用新一代背板总线连接的机箱结构，大大提高了系统的工作速度，同时减少了占用空间。更重要的是：作为虚拟

仪器，提高了软件的规范化和应用水平。VXI 总线即插即用规范的制定，为虚拟仪器系统的结构组建奠定了技术支持与指导，从而使组建一个统一的、开放的虚拟仪器系统成为可能。

第三代自动测试系统总线的发展在 VXI 出现后并没有停滞，针对 VXI 结构过于复杂、功能过于完善、售价过于高昂的问题，PXI 出现了，基于 USB 的虚拟仪器出现了，支持各种通信协议的远程控制的自动测试系统也出现了，网络仪器的概念开始进入工程界。

### 1.3.2 PXI 和 LXI 总线系统

尽管 VXI 总线在性能上以无与伦比的优越性受到很多用户（特别是军方用户）的高度评价，然而由于其结构过于复杂、功能过于完善、因此造成一些中小企业的用户感到售价过高而难以承受。为此，1997 年 NI（National Instruments）公司推出了一种称为 PXI（PCI eXtensions for Instrumentation ）的测试系统规范。这个规范的要点是用 PC 系统中流行的 PCI 总线来代替结构复杂、用于工业控制的 VME 总线，因此能把原先为数众多、物美价廉的 PC 总线模件直接用于测控系统，也能极为方便地得到那些人们十分熟悉的 PC 操作系统和各种应用软件的支持。同时各种用于测试的仪器扩展功能也做了一些简化（相对 VXI），但仍能提供 10MHz 的系统时钟、8 条非常灵活的触发总线、星形触发线和本地总线的支持。另外，PXI 系统继承了 VXI 的软件工作方式，也支持 VISA 标准和诸如 LabWindows/CVI、Lab View 等面向测试的软件开发工具，从而使原先在 GPIB 和 VXI 系统开发中建立的工作基础和积累的工作经验都可以在新的 PXI 系统开发中继续发挥作用。

图 1-9 是一个 PXI 系统的外形照片，它看上去与 VXI 系统很相似，也是采用标准主机箱并配置标准卡式模件，也是通过主机的面板进行操作。与 VXI 系统相比，PXI 系统的速度更快、体积更小、成本较低，但配置功能、电磁兼容性能和可靠性指标则不如 VXI 规范那样严格。PXI 系统的主要问题在于还没有为所有仪器开发商广泛认可，因此也还做不到像 GPIB 和 VXI 那样可以不分厂家、国别地根据测试需要自由选购到相应设备的程度。

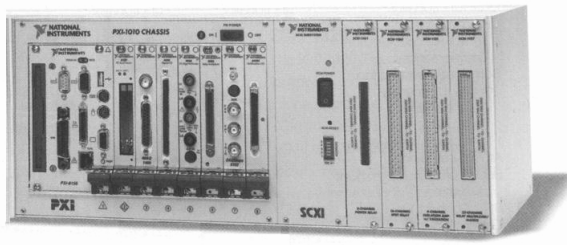



图 1-9 PXI 系统的外部结构

2004 年 9 月，Agilent 公司（原 HP 公司）和 VXI Technology 公司为自动测试系统推出了一种基于 LAN 的模块化测试平台标准 LXI（LAN eXtensions for Instrumentation）。从自动测试技术、微电子技术和计算机技术的发展情况来看，LXI 的提出是开放式标准化测试系统发展中合乎逻辑的结果，是成熟的以太网技术在测试自动化领域应用的拓展。其具体的设想是将非常成熟的以太网技术引入自动测试系统，以替代传统的测试总线技术。LAN 最明显的作用是替代 GPIB，它将保证系统中所有 LXI 单元能够在同一个网络中和平共处，LXI 支持 IEEE802.3 和 TCP/IP，试图为用户提供一个新的自动测试系统的构架，具有更快的速度和更为简单的编程方式。

国际 LXI 联盟是一个非盈利性的联盟，旨在测试测量自动化领域 LXI 标准的建立和发展。目前已包括绝大多数国际著名测试测量公司作为其会员，如 R&S、安立、泰克、吉时利等。安捷伦科技在北京宣布，随着 LXI 联盟最新技术规范的发布，将陆续推出多种符合 LXI 标准的测量产品。LXI 总线不受带宽、软件和计算机背板体系结构的限制，而且能利用日益增长的以太网吞吐能力，是一种适用于测试系统的新一代基于 LAN 的模块化平台标准。图 1-10 为 LXI 系统的结构和典型产品。

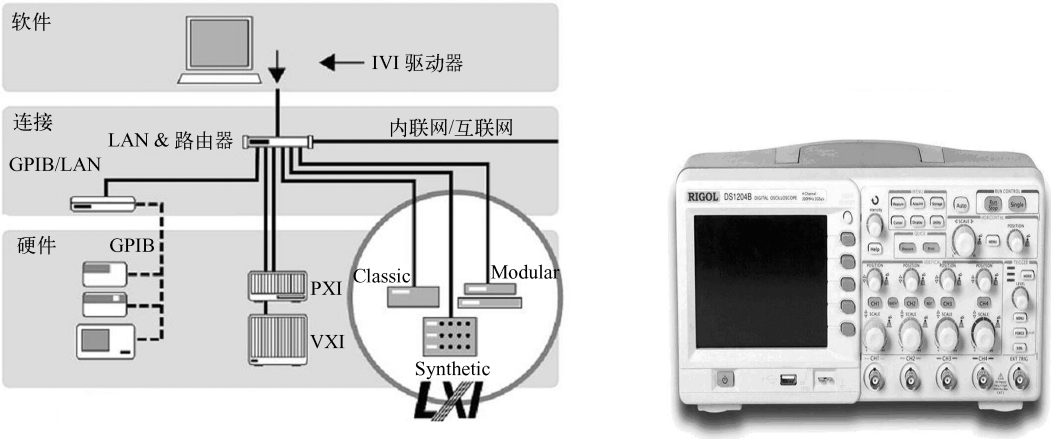



图 1-10　LXI 系统的结构和典型产品

LXI 目前已经得到较为普遍的支持，其主要原因如下：

（1）以太网、标准 PC 和软件技术非常成熟，且众多计算机厂家不断地加大研发及投入力度，使其性能更加完善，能够满足测试行业的广泛应用。

（2）LXI 是一种基于网络的仪器接口规范，为实现各种终端设备之间的同步控制，采用了以太网的精密时间协议（Precision Time Protocol，PTP）。这个协议定义了一个在测量和控制网络中，与网络交流、本地计算和分配对象有关的精确同步时钟，并被接受为 IEEE1588，这个网络同步标准可以在实验室环境中得到纳秒级的时钟同步误差，其标准网络接口也已普及，使 LXI 技术的前景十分看好。

（3）国际 LXI 联盟在众多系统集成商和工程师中的调查表明，人们普遍希望利用通用的 PC 接口和总线，而不是昂贵的测试测量的专用接口总线。

### 1.3.3　自动测试的应用领域

随着技术的进步和社会的发展，自动测试在军/民两个方面都得到了广泛的应用。在军事上，越来越多的武器装备配置了自动化和信息化设备，而设备中的电子装置的比例更是越来越高。这些设备的可靠性至关重要，在战场上一旦出现问题，轻则贻误战机，重则带来毁灭性的后果。以现代军用飞机为例，航空电子设备的性能和质量已经成为作战效能的决定因素。美国空军的统计数据表明：F15 战斗机的航空电子设备的故障率已占整个飞机故障率的一半，而其维护工时则超过整个飞机所用工时的三分之一。美国军方一直认为：测试是设备维护的重要组成部分，并于 20 世纪 70 年代提出了测试性概念，20 世纪 80 年代颁布了“电子系统及设备测试性大纲”（军用标准 MIL-STD-2165），20 世纪 90 年代则推行“综合诊断”军用

标准 MIL-STD-1814 的要求。同时大力发展 ATE(自动测试设备)和 METS(活动电子测试设备)等技术，以降低维修成本和人员要求，缩短维修时间。这些措施的结果，使每架飞机 20 年的使用保障费用由原先出厂价的 3.3 倍减少到 0.9 倍。这样看来，自动测试已经成为确保军用飞机作战适用性和降低使用保障费用的一项关键技术。

可靠性通常表现在两个方面：一是产品本身的质量可靠性，如平均无故障时间指标；二是出现故障后要求能够迅速查明原因，及时处理，即可维护性。军用电子设备数量巨大，要求处理故障迅速，而工作环境和空间又极其有限，而且很多场合甚至要求单兵作战，操作人员的专业知识和工作经验也很难达到理想的要求，这些都是不能不采用自动测试方法的基本原因。有关资料显示，某些产业固定资产的三分之一是仪器仪表和计算机，很多重要科研项目的仪器开支超过总研制经费的一半。为了适应军用系统对快速故障诊断的迫切要求，“可测性设计”方法在产品设计阶段预先为自动测试建立各种必要的条件，从而使自动测试的作用事半功倍。

再看民用领域，提高产品质量和确保生产安全始终是企业的两项基本工作。在冶金、电力、石化、轻工、建材等连续生产的过程中，每时每刻需要检测各种工艺流程的工作状态，从而确保各种工艺参数和质量参数。为了确定设备维护周期和大修的时机，还要检测和处理各种有关的安全参数和能耗参数，集数据采集、系统辨识和专家系统为一体的自动测试技术能够很好地解决这些问题。机电产品量大面广，针对五花八门的产品，从测试思想、测试方法，到测试的工作量都要求采用自动测试技术方能满足需要。在环保、气象、水文、地震的长期监测中，各种支持网络通信功能的自动化测试系统更是比比皆是。随着人民生活水平的提高，供电、供水、供气、供暖的计量逐步要求实现自动化管理，自动测试将信息获取和处理的结果直接转入信息的应用。医疗仪器、食品检验、室内装修则关系千家万户，自动测试将更能确保测试结果，克服测试过程中的人为因素。在高科技领域，新的测试内容和测试要求层出不穷，专用的仪器设备开发往往赶不上新产品的推出，有时只是为了某种新思想和新概念的研究并不值得都去开发新的专用仪器，如通信行业，对于新的载波调制方法、新的协议分析，虚拟仪器则有更大的市场需求。

需要指出，自动测试以及相关仪器仪表的重要性，不仅体现在其行业本身的产值，更重要的是它对国民经济发展的辐射作用和影响力。有关资料显示：美国仪器仪表行业只占国民经济的 4%，但对国民经济发展的影响面却高达 66%。随着知识经济、信息时代的到来，科技发展的步伐加快，必然要求测试更好、更快地满足需要，自动测试技术应在科学研究和经济发展中发挥更大的作用。

### 1.3.4 自动测试技术的发展规律

回顾自动测试系统的发展过程，可以发现其发展的某些规律：

(1) 自动测试的发展与经济和技术的发展同步，经济、技术发展越快，需要测试的范围就越宽，对自动测试的要求就越高。著名科学家门捷列夫指出“科学是从测量开始的”，反过来测量又将极大地促进科学的发展。由于越来越多的被测对象已经不再是能够直接度量的物理量，而是作为某些过程或系统的参数，因此单纯的测量(Measurement)变为复杂的测试(Testing)。即需要对被测对象施加某种激励，同时测取某种响应，然后通过计算来得到这种被测对象的参数结果。激励和响应在时间上可能是一个过程(时间序列)，空间上可能是多个

变量。测试的方法也不止一种，可能需要重复进行和交叉验证。这样比较复杂的测试过程当然需要依靠自动测试系统，而且随着测试领域和测试工作量的急速扩大，自然没有可能完全依靠专用的传统仪器了。

（2）像许多技术的发展一样，自动测试的发展最初来自军事上的需要。自动测试发展到了一定程度后，也如同其他一些新技术一样，会突破原先军事任务的狭隘范围，并在更广阔的天地得到更广泛的发展。作为现代自动测试研究工作的开端，美国军方在 20 世纪 50 年代中期，曾针对军用电子装备的维护问题，提出了一种所谓“万能”自动测试设备的概念。其目标为：实际上不必依靠测试技术文件，由非熟练人员进行全自动操作，并以计算机速度完成测试。这种设想并未达到预期的要求，尽管实现了计算机控制，但测试系统的响应远非计算机的速度；对仪器直接操作者的要求降低了，但却需要增加一些新的技术支持者；常规的文件资料被计算机软件代替了，可又多了一些新的软件资料。总之，贪大求全导致系统复杂，不仅使其价格昂贵，而且性能的提高反而受到某些限制。不过尽管如此，这种思想仍然主导着自动测试的发展，其原因就在于军事任务复杂情况的实际需要和不惜代价的实际条件。然而，当自动测试进入民用领域后，人们就不能不采取一些新的对策了。生产过程中遇到的被测对象常常是大量相同或相似的产品，人们对被测对象往往拥有完全的控制权。民用自动测试系统往往可以针对被测对象最关键的环节，选择最有利的时机，进行有限目标的有效测试。这时所采用的测试策略、测试指标、测试方法、测试设备也都有更多的选择，自动测试技术可能具有更为广阔的应用前景。

（3）谈到具体自动测试系统的结构，真可以说是“分久必合，合久必分”。先是集中管理方式的大型接口结构的 CAMAC 系统发展为分散到各普通台式仪器设备中的 GPIB 接口总线系统，以后又发展为 VXI 和 PXI 系统的集中式总线背板机箱，再到 USB 小盒和分布式的网络设备，直到 LXI 系统和合成仪器，并以网络代替总线。自动测试系统发展的历史表明，集中与分散、全面与专业、性能与成本这几对基本矛盾始终影响着设计者的权衡取舍。没有固定的模式，只有实事求是、与时俱进的要求。然而，标准化、小型化、系统化、智能化、软件化的趋势也是十分明显的，能否不分厂家国别、按测试任务的需要选择测试设备是标准化工作是否成功的重要标志，尽量采用通用计算机及其软件来代替专用设备的虚拟仪器思想也确实是自动测试技术在今后一段时间的努力方向。

（4）信息技术对自动测试系统发展所起的关键作用值得注意。自动测试不仅与信息的获取（各种传感器）密切相关，也与信息的传递联系紧密。从 GPIB 和 VXI 等专门用于测试系统的接口和总线，到虚拟仪器、网络仪器概念的形成，处处分享着通信技术的最新成果。同时，自动测试结果早已不再满足于人的眼观手记，信息的处理和存储已经成为自动测试的基本组成部分。加上故障诊断、预报预测等与测试密切相关的信息技术，自动测试也已直接影响到信息的应用。这样，自动测试的发展一方面应当紧密联系实际被测对象的实际需要，利用现代物理、化学、生物科学成果不断扩展测试的范围和提高测试的精度。另一方面就是要紧跟信息技术，从信息的获取，到信息的传递、信息的存储、信息的处理，乃至信息的应用。总之，就是要紧紧抓住传感器和数据采集核心技术的同时，不断地从通信技术、计算机技术和软件技术的发展中汲取推动力量。

### 1.3.5 测试工程师的责任和本书的主要内容

鉴于自动测试技术在现代企业生产和科学研究过程中的重要作用，测试工程师的工作已经成为一项专门的业务领域。测试工程师需要根据科研和生产的需要，了解测试要求、选择测试内容，然后制定测试计划、建立测试方案和策略，选购、维护和校验测试仪器和设备、编写测试大纲和测试流程，同时针对具体的测试任务，组建测试系统、编写测试软件，有时还要开发测试设备。测试进行过程中要检查测试质量，测试完成后要审查测试报告并提出意见。自动测试和虚拟仪器技术的发展，为测试自动化提供了更宽阔的发展空间。本书将集中讨论自动测试系统和虚拟仪器的基本原理和开发技术，力图为测试工程师在这一领域开展工作建立必要的基础。

本书第 1 章首先介绍基于计算机的自动测试系统与虚拟仪器的基本概念和主要特点，分析其发展过程，讨论其发展方向，阐述其在信息技术中的地位，以及其与控制技术、通信技术和计算机技术的关系，进而指出测试工程师的基本任务和本书的主要内容。

数据采集器是基于数字信号的计算机系统获取实际物理对象中连续被测物理量的主要手段，采样和量化是其中最重要的环节。本书将在第 2 章讨论不同形式的 A/D 转换技术，阐述放大器、隔离器、滤波器、多路开关等通道调理电路的基本原理，以及微处理器控制的硬件接口和软件结构，分析采样过程、采样方式和采样参数的影响。

为了组建自动测试系统时能够不分厂家、国别地自由选择需要的设备，需要掌握诸如 GPIB 通用接口总线和 VXI 仪用总线系统等专门用于自动测试系统的国际标准。本书将在第 3 章和第 4 章就它们的工作原理、协议要点、开发方法和应用示例展开讨论，并对诸如 IEEE488.2、SCPI、VISA 标准的基本内容和要点进行简明的介绍和解释，同时以 LabWindows/CVI 和 VB 为例说明使用常用软件编程工具的方法。

鉴于现代自动测试系统中常将测试设备配置到空间中分布在不同位置的被测对象附近，因此连接这些测试设备的通信方式及其相关的通信协议和通信设备就显得至关重要。本书将在第 5 章从最基础的 RS-232 串行通信及其在电信网中的调制解调技术谈起，介绍其原理、协议和设备。并结合具体的自动测试系统示例说明其应用方法。然后通过 TCP/IP 和无线通信技术的概况介绍，说明实现远程网络仪器的某些方法。

虚拟仪器的基本思想是强调通用计算机及其软件在测试系统中的作用。本书将在第 6 章重点介绍 USB 作为 PC 与外设连接的主要接口，开发虚拟仪器装置的通用方法。同时，展示应用该方式开发的一些典型仪器设备，进而说明这种虚拟仪器的发展方向。

作为自动测试系统的典型应用形式，本书第 7 章讨论 PCB 电子功能模件的测试技术。首先介绍 ATE 的基本概念，然后分别讨论路内测试、功能测试、总线测试和边界扫描测试等具体技术。

针对计算机测试系统中软件开发和系统组建的实际需要。本书将在第 8 章讨论离散傅里叶变换和 FFT 算法、数字滤波器设计方法和应用示例、基于时间序列建模的系统辨识原理和数理统计的应用方法。并结合大型旋转机械的状态监测与故障诊断系统和小型心电监测系统，介绍自动测试系统和虚拟仪器的典型开发过程，内容涉及传感器配置、锁相环倍频、专家系统、信号处理、系统辨识、操作界面等有关技术。

## 本 章 小 结

自动测试系统的结构从大型机架的 CAMAC 系统到台式仪器设备中的 GPIB 接口，再发展为 VXI 和 PXI 系统的集中机箱中的背板总线，又回归成 USB 测试小盒和台式合成仪器。总线连接还向网络连接发展，形成 LXI 技术。软件则从总线的 I/O 控制、设备驱动程序发展到虚拟仪器结构。这些事实充分体现了集中与分散、通用与专业、性能与成本、硬件与软件等基本矛盾在自动测试系统发展过程中的主导作用，也反映了人们对标准化、小型化、网络化、智能化、软件化的期望。自动测试不仅要满足信息获取的基本需要，还要跟上信息技术的发展，适应信息传递、信息存储、信息处理和信息应用的新方式，满足人们日益提高的信息需要。

## 参 考 文 献

[1] 张世箕，杨安禄等. 自动测试系统(上). 中国仪器仪表学会，电磁测量与信息处理分会资料，1980.
[2] 张毅刚，彭喜元，姜守达，付平. 自动测试系统. 哈尔滨工业大学出版社.2001.
[3] 赵会兵. 虚拟仪器技术规范与系统集成. 北京：清华大学出版社.2003.
[4] Lab View Quick Start Guide. National Instruments Corporation.1996.
[5] 王大衍，胡柏顺. 加速发展我国现代仪器事业，迎接 21 世纪挑战.现代科学仪器[J]，2000.
[6] 殷纯永.仪器科学发展与技术发展建议. 中国机械工程[J]，2003.11(3).
[7] 张礼勇，刘思久. 机电行业自动测试系统的一种典型实现模式. 仪器仪表学报，2001(01).
[8] 刘思久，张礼勇. 虚拟仪器与 USB 设备开发.电测与仪表，2003(01).
[9] 刘思久，张礼勇，冯玉贵. 面向测试的软件工具. 中国电子报，1997.
[10] 刘思久，陈祥立，张礼勇等. 面向用户接口的虚拟仪器控件. 电测与仪表，1997(12).
[11] 叶关山. PXI 总线平台的比较研究.科学技术与工程，2008.8.
[12] 高明. 解读推出 LXI 的思路和 LXI 的特点. 国外电子测量技术，2005.6.
[13] 诸玲珍. LXI 构筑电子测试新平台. 中国电子报，2005.10.
[14] 张礼勇. 总线式自动测试系统. 电测与仪表，1994.8.
[15] 张礼勇，程玉润. GP-IB 接口讲座(七讲连载).电测与仪表，1980.7.
[16] 张成名，王建华，蔡金燕.自动测试系统的通用性研究.微计算机信息，2007.
[17] 曹成俊，张宏伟.自动测试系统中的总线技术.现代电子技术，2008.
[18] 朱治国，郑建荣，刘小平，顾秀兰.虚拟仪器及其常用开发软件.现代仪器，2004.
[19] 仇志平，李树军.Labwindows/CVI 虚拟仪器软件在测试领域中的应用.计算机工程与设计，2007.11.

# 第 2 章　数据采集器

数据采集(简称数采)器是组成自动测试系统和虚拟仪器的基础设备，数据采集技术是一般数字化测量技术在时间、空间和功能上的扩展。一方面通过对被测物理量持续的定时采样测量，形成一组反映数字波形的时间序列数据；另一方面对多个模拟量通道同时测量，取得对实际环境及其相互关系的全面了解。为了满足实际应用的各种不同需要，采集的数据结果往往还需要通过某种方式进行存储、传输和处理，形成数据采集系统(Data Acquisition System，DAS)。本章将介绍数据采集器的基本结构，阐述具体通道设备，说明嵌入式系统的控制方式，并讨论采样过程中的一些理论问题。

## 2.1　数据采集器的基本概念

本节从分析数据采集器的基本结构开始，讨论数字化测量中核心的量化功能及其误差，重点阐述 A/D 转换器与采样保持器的工作原理、主要性能、操作方法和选用原则。

### 2.1.1　数据采集器的基本结构

图 2-1 是一个典型的数据采集器的基本结构框图。在将表达外界被测物理量的模拟电压信号转换成数字信号的过程中，采样保持器和 A/D 转换器是最核心的器件，用来直接完成采样和量化的工作过程。采样保持器将连续的被测信号电压采样成离散信号并保持稳定，A/D 转换器则将这个稳定的模拟信号量化为数码形式的数字信号。尽管现代的 A/D 转换器常常将采样保持器放到其芯片内部，然而在某些场合也需要将它们分开配置。数据采集器通常采取等间隔定时采样的方法，采样时钟常常通过内部微处理器的定时/计数器直接产生，也可通过连接外部信号来实现某种方式的同步测量。

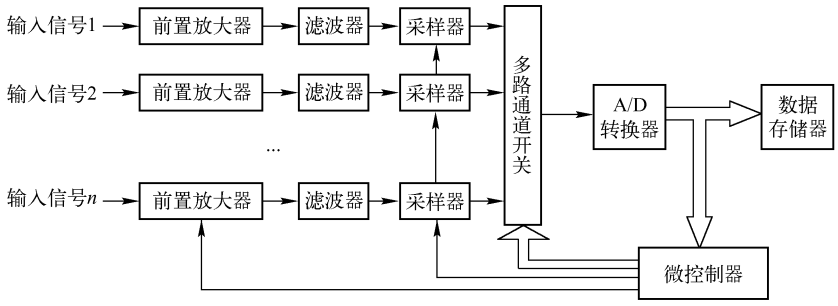



图 2-1　典型的数据采集器的基本结构框图

数据采集器的通道调理电路是指那些为被测模拟信号在进入 A/D 转换器前用来进行各种预处理的电路，包括多路通道开关、前置放大器、滤波器等。这里，通过依次切换多路通道开关，可以将 A/D 转换的输入依次连接到不同的数据输入通道，以此实现用一个 A/D 转换器对多个被测信号的测量。前置放大器可将来自传感器的外部输入信号电平转换成 A/D 转换器所要求的电平工作范围，并实现阻抗匹配；前置放大器也可做成隔离放大器来隔离外界环境与 A/D 转换器之间电的联系，常常用于高压、医疗和对可靠性要求较高的场合。滤波器的作用主要用来抑制各种噪声，特别是在对动态信号的采样操作之前可以起到抗混叠的作用(详见 2.4 节)。整个系统的结构有时会根据不同的实际需要进行一些调整，如为了适应各数采通道的不同条件，各通道可分别完成电平转换、缓冲放大、隔离放大和滤波后再由通道开关进入 A/D 转换器。反之，对标准化的被测对象则可先进行通道切换，再进行放大、隔离和滤波操作，从而减少调理电路元件的数量。又如通常可以利用微处理器程序将 A/D 转换器的测量结果直接读入其累加器或其他工作寄存器，然后再进行存储和处理。而有时却由于要求的数采速度很快，微处理器的工作速度不够，便将整个数采过程分为两个阶段，先将高速 A/D 转换的结果直接存入数据存储器，然后再由微处理器将数据逐个读出进行处理。

目前，由于上述这些基本电路环节大多都已做成集成电路芯片，甚至提供标准的操作接口，因此数据采集器通常总是需要微处理器通过读/写它们内置的各种专用寄存器来了解其状态、控制其动作并传输数据。同时，数据采集器得到的测试结果也往往需要通过微处理器及其通信接口进行进一步数据处理(如数字滤波、统计分析等)、数据存储(如 SD 存储体、U 盘等)、数据传输(如连接主机、网络)和数据的应用(如构成显示仪表或产生对其他设备的操作动作)，也可能需要接受上层主机或网络系统的监控。

### 2.1.2　A/D 转换器的量化功能

A/D 转换器是数据采集器的核心器件，量化则是 A/D 转换器的基本功能。数字信号可以用数字量序列来描述，其中每个数字量用有限位的二进制码来表示。A/D 转换器常常是单极性工作方式，如 0~5V 的量程范围。这时可采用直接二进制方式编码，这种编码以单位 1 表示满量程范围 FSR，用二进制数字表示 FSR 的分数部分。即对 0~FSR 之间的任何实际值，均可用 0~1 之间的数值表示，并写成二进制码 $a_1\ a_2\ a_3\cdots a_n$ 的形式，且与十进制小数 $N$ 有下列关系：

$$N = a_1 \cdot 2^{-1} + a_2 \cdot 2^{-2} + \cdots + a_n \cdot 2^{-n} = \sum_{i=1}^{n} a_i \cdot 2^{-i} \tag{2-1}$$

式中，$n$ 为二进制码的字长，$a_i$ 取值 0 或 1。如

$$0.1101\text{B} = 1\cdot 2^{-1} + 1\cdot 2^{-2} + 0\cdot 2^{-3} + 1\cdot 2^{-4} = 0.5 + 0.25 + 0 + 0.0625 = 0.8125\text{D}$$

式中，B 表示二进制数，D 表示十进制数。通常将二进制小数点省去，直接写成

$$1101\text{B} = 0.8125\text{D}$$

上式所表示的实际值为 0.8125·FSR。

同理可以得到：0100B = 0.25D，0001B = 0.0625D 等数值关系，表示的实际值分别为 0.25 倍和 0.0625 倍的满量程值。

考虑一般情况，若一个量化器件的满量程范围为 FSR，二进制码的字长为 $n$，则有如下一些量化的术语和概念：

量化电平　　$Q=\mathrm{FSR}/2^n$　　(2-2)

最低位权值　　$\mathrm{LSB}=\mathrm{FSR}/2^n=Q$　　(2-3)

最高位权值　　$\mathrm{MSB}=\mathrm{FSR}/2$　　(2-4)

量化器所能表示的最大码为　　$\mathrm{FSR}\ (1-2^{-n})$　　(2-5)

例如，某 A/D 转换器的满量程范围 FSR 为 5V，字长 $n$ 为 8，则有量化电平 Q=0.0195V (5V/256)；LSB=0.0195V；MSB=2.5V(FSR 的一半)。且最小码 00000001 的对应值为 0.0195V(FSR/$2^n$)；最大码 11111111 的对应值为 4.98V(5V×255/256)。

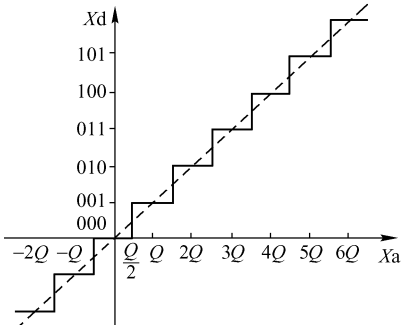



图 2-2　量化误差示意图

由于字长 $n$ 总是有限的，因此量化总是存在误差。显然 $n$ 增加时 $Q$ 减少，导致量化误差减小。如图 2-2 所示，量化误差属随机性质，被视为量化噪声，量化噪声具有在$-Q/2$ 到$+Q/2$ 之间均匀分布的特性，其最大值为量化电平的一半。量化噪声的方差也很重要，它反映噪声的平均功率，可计算其值为 $Q^2/12$。显然，A/D 转换器的字长越长，量化引起的误差越小。常见 A/D 转换器字长为 8、10、12、16、20 和 24 等，对于 10 位字长的 A/D 转换器，其分辨率可以做到满量程信号的 1‰。

### 2.1.3　A/D 转换器的基本工作原理和控制方式

A/D 转换器的工作原理分为双积分式、逐次比较式、并行闪烁式和 Δ/∑式等几种常用的类型。

双积分式 A/D 转换器的工作原理如图 2-3 所示，控制逻辑电路将 A/D 转换分为 $t_1$ 和 $t_2$ 两个积分过程，即未知的被测输入信号 $V_X$ 在固定的 $t_1$ 区间将积分电容 C 充电到一定数值，再由已知的直流参考电压 $V_{ref}$ 在 $t_2$ 区间反向积分(恒流放电)到初始值。

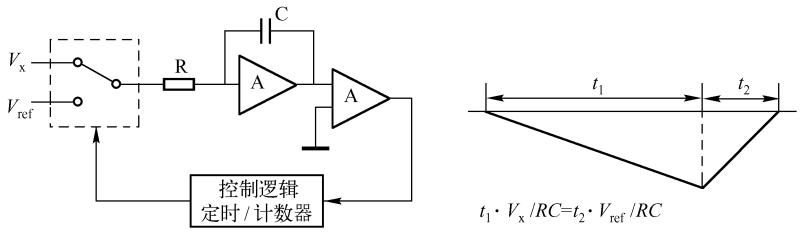



图 2-3　双积分式 A/D 转换器的工作原理

输入信号电压和参考电压之间的关系为

$$\frac{1}{RC}\int_0^{t_1} V_x(t)\cdot \mathrm{d}t=\frac{1}{RC}\int_0^{t_2} V_{ref}\mathrm{d}t=\frac{1}{RC}V_{ref}\cdot t_2 \qquad (2-6)$$

从而得到

$$V_{xAve}=\frac{1}{t_1}\int_0^{t_1} V_x(t)\cdot \mathrm{d}t=\frac{1}{t_1}V_{ref}\cdot t_2=V_{ref}\cdot\frac{n_2}{n_1} \qquad (2-7)$$

式中，$n_1$ 和 $n_2$ 分别为测量两次积分时间 $t_1$ 和 $t_2$ 的时钟计数值。由于积分式 A/D 转换器实际上是在测取被测信号的平均值，因此对于随机干扰具有很好的抑制能力，可以得到很高的精

度，但所需的转换时间较长。

并行闪烁式 A/D 转换器原理如图 2-4 所示，参考电压被电阻分压后直接产生各级标准电压，它们同时与被测信号 $V_x$ 比较，低于 $V_x$ 者为 1，反之为 0。然后通过数字编码电路将这 $2^n$ 个表示比较结果的逻辑信号归纳成 $n$ 位二进制数码。

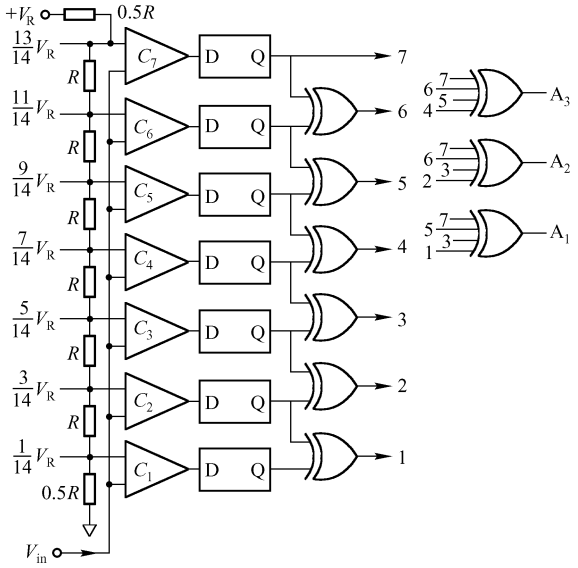



图 2-4　并行闪烁式 A/D 转换器原理

并行闪烁式的整个 A/D 转换过程只受比较器本身和编码电路的影响，因此可以得到很高的转换速度。但 $n$ 位字长需要 $2n$ 个比较器，故此种 A/D 的成本会随其分辨率的提高而急剧增加，而且所有比较器都在相当的功耗下同时工作，因此这种 A/D 一般只是用于高速、低分辨率的场合。

逐次比较式的 A/D 转换器最为常见，如图 2-5 所示。这种 A/D 转换器通过内部数据寄存器连接的一个 D/A 转换器产生逐次改变的比较电压量，数据寄存器各位自高至低依次置 1 试探，对应 D/A 转换器输出的比较电压量则按参考电压的 1/2、1/4、…、$1/2^n$ 的大小逐次改变增量后与被测信号电压进行比较试探，相当于天平秤重过程中逐次添加从大到小的比较砝码。在逐次比较的每个工作节拍下，信号电压大于比较电压时保持数据寄存器试探位为 1 的取值，反之将该位置 0 使比较电压量恢复到试探前的状态，相当于拿掉刚加的砝码。最终数据寄存器的码值即为 A/D 转换的输出结果。

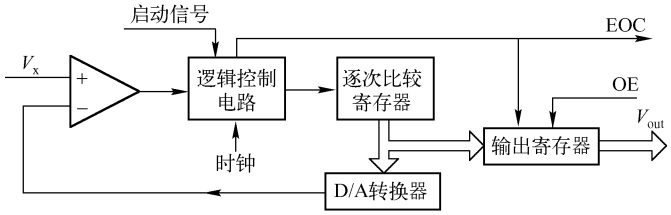



图 2-5　逐次比较式 A/D 转换器原理

由于逐次比较需要进行 $n$ 次，所以整个 A/D 转换的速度不是很快，但转换器所用元件的数量较少，能以较低的成本获得较高的分辨率，其性能价格比最好。

Δ/∑式 A/D 转换器是一种较新的方式，工作原理与前面介绍的几种情况不同。如图 2-6 所示，虚线框内是 Δ/∑调制器，其 1 位字长的 A/D 转换器以 $K$ 倍信号频率 $f_m$ 的采样速率 $f_s$ 将 B 处的模拟信号转换成 C 处由 1 和 0 构成的串行数字位流，再通过 1 位 D/A 转换器得到 D 处的模拟反馈信号。先不考虑这些 1 位转换器带来的量化误差，则 D 处模拟反馈信号表达了前次 B 处的测试结果。若将实际被测模拟输入信号 $V_{in}$ 与 D 处模拟反馈信号的差所形成的 A 处信号看做输入信号的增量，则其位于 B 处的积分量代表的自然是信号本身。根据反馈控制理论(前向通路增益足够大时单位反馈系统的输出信号将跟随输入信号)，考虑积分器对直流信号的增益可视为无穷大，则 B 处的输出电压平均值(或 C 处数字信号的平均数)就应等于输入信号的平均电压值。这样，直流信号的测取就可以通过非常简单的模拟电路(比较器、开关、积分器、一位 D/A 及模拟求和)构成的 Δ/∑型 A/D 转换器，得到 C 处数据流后计算平均数来实现。

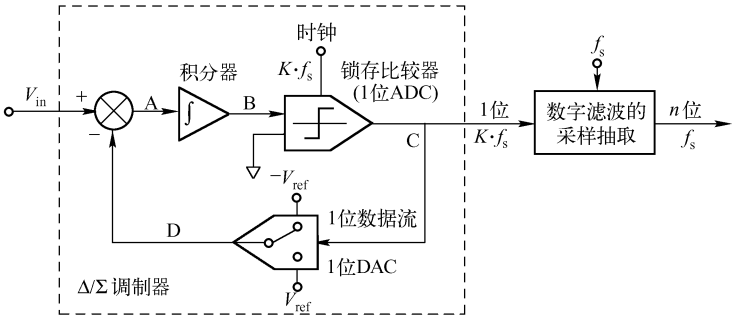



图 2-6　Δ/∑式 A/D 转换器原理

对于更一般的情况，一个常规 $n$ 位 A/D 或 D/A 转换器的量化误差作用可以看做加入了一个取值在 $Q/2$ 至 $-Q/2$ 之间、方差为 $Q^2/12$、均匀分布在 $0 \sim f_s/2$ 奈奎斯特频带范围内的随机噪声信号($Q$ 为 LSB 权重，$f_s$ 为采样频率)，且可采用模拟低通滤波器滤除 $f_s/2$ 以上的高频信号。

如图 2-7 所示，1 位量化器产生的量化噪声自然很大，但采用 $K$ 倍的过采样速率时，奈奎斯特频率增大了 $K$ 倍，整个量化噪声分布在 $0 \sim K \cdot f_s/2$ 之间，其单位频带下的噪声有效值降为原来的 $1/K$。由于针对被测信号的系统通带频率仍为 $f_m$，如构建滤波器滤除 $f_m$ 至 $K \cdot f_s/2$ 之间的无用信号，就能明显提高信噪比，实现用低分辨率器件实现高分辨率 A/D 转换的效果。根据采样定理，用于抗混叠的模拟低通滤波器只需滤除 $K \cdot f_s/2$ 以上的频率分量，而滤除 $f_m$ 至 $K \cdot f_s/2$ 之间无用信号的任务则可交给 A/D 转换之后的数字滤波器完成。

总之，Δ/∑式 A/D 转换器以很低的量化分辨率和很高的采样速率将模拟信号数字化，利用过采样技术、噪声整形和数字滤波技术增加有效分辨率。近年来 Δ/∑式 A/D 转换器发展很快，分辨率高达 24 位的产品在低成本、高分辨率的低频(直流到音频)信号处理场合得到了广泛应用，其设计思想也颇具启迪意义。

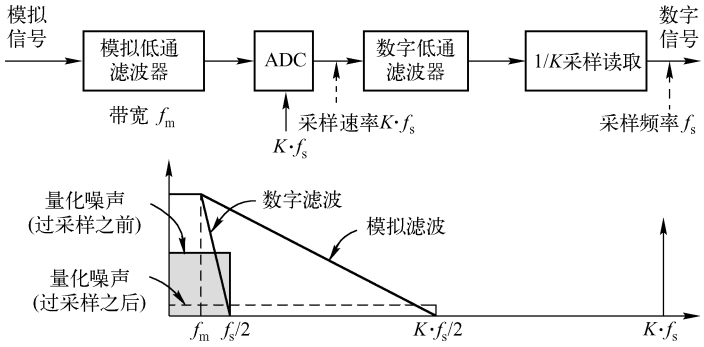



图 2-7　过采样抑制量化噪声原理

通常 A/D 转换器的控制方式可用图 2-8 表示。这里除了正确地选择适当的电源电压外，一定要尽可能地确保参考电压信号的准确和稳定。因为根据 A/D 转换器的工作原理，所有工作过程和最终结果都是以这个电压信号作为参考量与被测信号进行比较得出的。启动输入信号 Start 用于通知 A/D 转换的进行，通常为一个脉冲信号。状态输出信号 State 则表示一次转换已经完成，所提供的数据有效。输出选通 OE 用于选通数据输出的三态门。

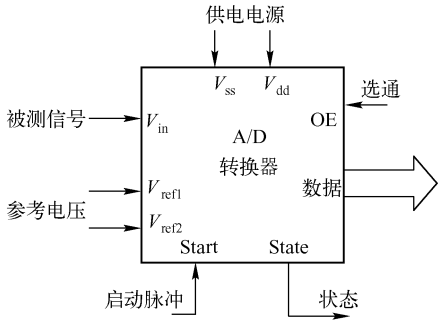



图 2-8　A/D 转换器的控制方式

一种微处理器操作的典型过程为：①通过写端口产生的脉冲信号启动 A/D 转换器；②通过读端口反复查询转换状态；③通过读数据信号选通 OE 读出数据。当然，启动信号也可来自其他外部或内部的同步信号，转换结束的状态信号也可采用触发中断的方式来要求微处理器响应。这里表述的数据输出方式是典型的并行数据接口，而近年来随着数字技术的发展，一些 A/D 转换器配置了串行数据接口。由于串行接口的接线简单，因此颇受工程师的欢迎。

### 2.1.4　采样保持器的工作原理

考虑 A/D 转换器总是需要一定时间进行量化和编码操作，因此如果直接使用 A/D 转换器进行数据采集，就应保证在 A/D 转换过程中被测信号不变。假定输入为满量程的正弦信号 $V_{FSC}\cdot\sin\omega t$，则考虑信号变化最大的过零处在 A/D 转换时间 $\Delta T$ 内可能出现的信号变化应为

$$\Delta V=\Delta T\cdot\left.\frac{\mathrm{d}V}{\mathrm{d}t}\right|_{t=0}=\Delta T\cdot V_{FSC}\cdot\omega \tag{2-8}$$

这样，若将此信号变化限制在半个 LSB 内，则可得到允许的数采速度为

$$f_{max}=\frac{1}{2\pi}\omega_{max}=\frac{1}{2\pi}\cdot\frac{\Delta V}{\Delta T\cdot V_{FSC}}=\frac{1}{2\pi\cdot\Delta T}\cdot\frac{1}{2^{n+1}} \tag{2-9}$$

式中，$n$ 为 A/D 转换器的字长，显然数采精度的要求极大地限制了数采的速度。

解决上述矛盾的一个方法是使用采样保持器，图 2-9 是采样保持器的工作原理图。采样

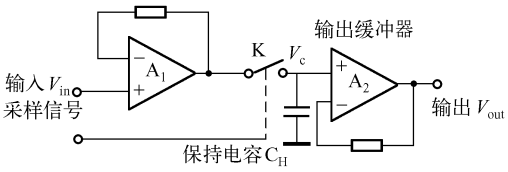



图 2-9　采样保持器的工作原理

保持器的工作分采样和保持两个阶段，开关 K 闭合时跟踪捕捉被测信号，开关 K 关断时则通过电容 $C_H$ 将捕捉到的被测信号电平稳定地保持。实际应用中采用的集成电路元件通常已将采样开关和输入/输出缓冲器做在芯片内部，而保持电容本身需选择具有低泄漏特性的聚苯乙烯、聚丙烯等介质的电容器。

图 2-10 用于描述采样保持器工作的动态特性。可以看出实线表示的输出信号在采样阶段紧紧地跟随着虚线表示的被测输入信号，而当切换到保持阶段后则能通过一段稳定的孔径时间后维持捕捉到的被测信号电平。由于采样保持器的孔径时间比 A/D 转换器的转换时间小得多，因此这种先采样、后量化的方法可以大大提高数据采集器的工作频率。

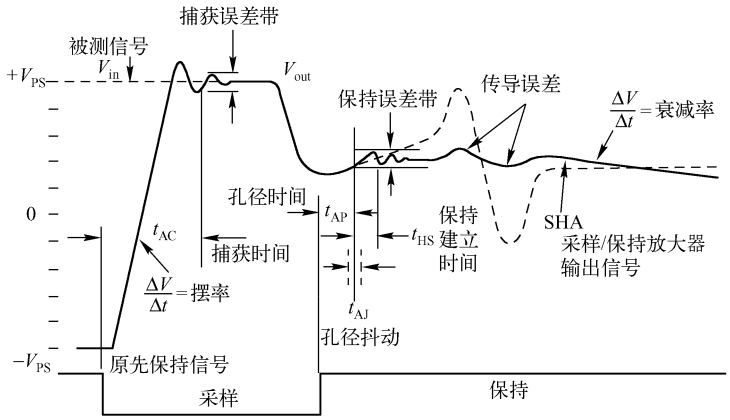



图 2-10　采样保持器的动态响应工作原理

### 2.1.5　A/D 转换器的主要性能指标和选择原则

数据采集器的核心性能是测量精度和工作速度，相应 A/D 转换器的主要性能则是其分辨率(字长)和转换时间。

由于字长直接决定量化噪声，每增加 1 位字长意味着减少 6dB 的量化噪声，10 位分辨率的 A/D 转换器的量化噪声应为满量程值的-60dB(1/1000)，因此为了提高数采系统的精度，首先必须确保 A/D 转换器的字长。进而为了真正实现这个上限指标所能达到的精度，还应考虑诸如增益误差、偏置误差、线性误差、温度漂移等诸多因素的影响。另外，其他外围元件和相关条件的配合也很重要，如参考电压不能长期稳定，或温度特性不好，就不可能获得较高的测量精度。但有时数据采集器要求采用高分辨率的 A/D 转换器并不完全是精度的需要，而是考虑数据采集信号的动态范围。因为通常所说的精度常常是指满量程范围下的相对误差，如某 8 位字长的 A/D 构成的数据采集器满量程范围为±5V，则对于峰值为 5V 左右的正弦波信号来说，1%的精度要求是可以达到的。但如果同时考虑峰值为 1V 左右的正弦波输入信号，原来的精度就无法实现了，因此也可以考虑提高 A/D 转换器字长的方法来解决这个问题。

数据采集器工作速度的上限由 A/D 转换器的转换时间决定，在此基础上还应考虑采样保持电路的孔径时间、A/D 转换器的操作时间等因素，以及本章后面还要专门讨论的采样信号处理问题。一般说来，数据采集器的采样速度决定了所选 A/D 转换器的类型：双积分式 A/D 转换器价格便宜、精度高、抗干扰性能好，但速度低；并行闪烁式 A/D 转换器恰好相反，价格贵、精度低、功耗大，但速度很高；逐次比较式的 A/D 转换器应用范围宽，字长 8 位、10 位、12 位、16 位的产品比比皆是，速度从每秒几十千次采样到几兆次采样都有，而且性价比好。Δ/∑式 A/D 转换器成本低，精度可以做得很高，但有时需要配置专门的数字滤波器，且由于滤波器需要较长的稳定时间，故难以采用多路通道开关来实现快速的巡回检测。

除此之外，电源的供电方式、参考电压的精度要求、操作接口的工作方式，以及温升、功耗、封装和体积也都是选择 A/D 转换器需要考虑的因素。

表 2-1 给出了一些 A/D 转换器的性能指标和典型应用方式。由于自动测试过程中，对被测对象进行数据采集的同时，常常需要提供一定的测试条件，如测量过程需要改变测试现场的温度，测试阻抗时需要激励电压等，故将常用的几种 D/A 转换器及典型参考电源也一并列出，以供读者参考。

**表 2-1　几种典型的 A/D 和 D/A 转换器的主要性能指标**

| 型　号 | 原　　理 | 字长 | 通道数 | 速度 | 接口方式 | 封装 | 备　　注 |
|---|---|---|---|---|---|---|---|
| AD1674 | 逐次比较 A/D | 12 | 1 | 1MHz | 并行 | 28 脚 | 数据采集 |
| AD7665 | 逐次比较 A/D | 16 | 2 | 500kHz | 并行，SPI | 48 脚 | 数据采集 |
| ADS805 | 并行闪烁 A/D | 12 | 1 | 20MHz | 并行 | 28 脚 | 高速采集 |
| AD9058 | 并行闪烁 A/D | 8 | 1 | 50MHz | 并行 | 44 脚 | 高速采集 |
| AD9228 | 并行闪烁 A/D | 8 | 2 | 65MHz | 并行 | 48 脚 | 高速采集 |
| AD9252 | 并行闪烁 A/D | 14 | 8 | 50MHz | 串行，LVDS | 64 脚 | 多路高速采集 |
| ICL7135 | 双积分 A/D | 4.5D | 1 | 50Hz | 并行 | 28 脚 | 数字万用表 |
| AD7705 | Δ/∑　A/D | 16 | 2 | 500Hz | SPI | 16 脚 | 慢变测量 |
| ADE7755 | Δ/∑　A/D | 16 | 2 | 500Hz | 脉冲输出 | 24 脚 | 电度表 |
| CS5550 | Δ/∑　A/D | 20 | 1 | 1kHz(典型) | SPI | 24 脚 | 地震测量 |
| LTC2400 | Δ/∑　A/D | 24 | 1 | 8Hz | SPI | 8 脚 | 电子秤 |
| ADS1110 | Δ/∑　A/D | 16 | 1 | 240Hz | $I^2C$ | 6 脚 | 便携仪器 |
| AD568 | 双极性输出 D/A | 12 | 1 | 28MHz | 并行 | 24 脚 | 波形发生 |
| DAC2932 | 电流型输出 D/A | 12 | 2 | 40MHz | 并行 | 48 脚 | 波形发生 |
| MAX536 | 电压型输出 D/A | 12 | 4 | 300kHz | SPI | 16 脚 | 测控 |
| MAX520 | 电压型输出 D/A | 8 | 4 | 50kHz | $I^2C$ | 16 脚 | 测控 |
| AD780 | 高精度参考源 | ±1mV | 1 | 2.5~3V 引脚编程 | | 8 脚 | 测控 |
| REF3030 | 高输出参考源 | 0.2% | 1 | — | | 3 脚 | 测控 |

## 2.2　数据采集器的通道调理电路

数据采集器的通道调理电路是连接其核心采样-量化功能与外界实际应用环境的纽带，本节将针对缓冲放大器、仪器放大器、隔离放大器、滤波器、多路通道开关等有关电路，讨论其工作原理和应用要点。

### 2.2.1　缓冲放大器与仪器放大器

通道调理电路中缓冲放大器最基本的要求是电平转换和阻抗匹配。A/D 转换器的输入信号范围通常要求单极性的有限电压，而连接外界传感器的数据采集器输入却可能需要提供更宽的电平选择，而且常有双极性输入的要求。在图 2-11 所示的缓冲放大器中，要求满足±10V 的输入信号动态范围，输出信号则要求连接到量程为 0~2V 的 A/D 转换器。

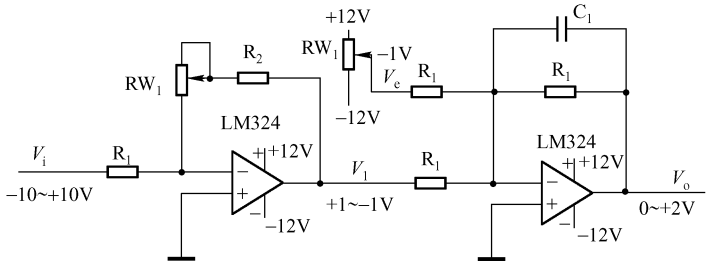



图 2-11　用于电平转换和增益调整的通道调理电路

这里，缓冲放大器的主要器件是两个运算放大器(简称运放)，接成反相输入形式是为了更容易实现放大增益和偏置电平的调整。其输入信号与输出信号的关系为

$$V_o = \frac{R_2}{R_1} \cdot V_i - V_e \tag{2-10}$$

为了确保测量电路不会影响被测电路的工作状态，应使通道调理电路具有足够高的输入阻抗。由于反相输入方式形成的“虚地”作用使缓冲放大器的输入阻抗可以简单地看做输入电阻 $R_1$，因此为了进一步提高输入阻抗特性和共模信号抑制特性，通常采用一种称为仪器放大器的缓冲输入电路。

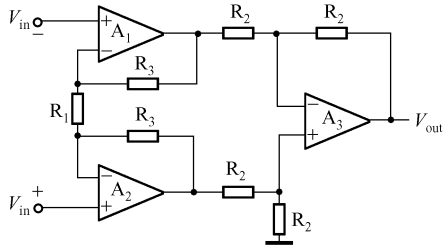



图 2-12　仪用放大器电路

如图 2-12 所示，在这种专用缓冲放大器中，前面两个运放采用正向输入方式，从而可以得到很大的输入阻抗特性。同时，两个运放接成差分方式结构可以十分有效地提高电路抑制共模干扰的能力。另外，为了保持电路的对称性，改变增益最简单的方法是改变 $R_1$ 和 $R_3$ 的值，即增益

$$K = 1 + \frac{2 \cdot R_3}{R_1} \tag{2-11}$$

AD612/614 是实现这种仪用放大器的一种典型芯片，其内部原理如图 2-13 所示。这种芯片可以通过改变引脚 1 与引脚 2 之间的电阻阻值

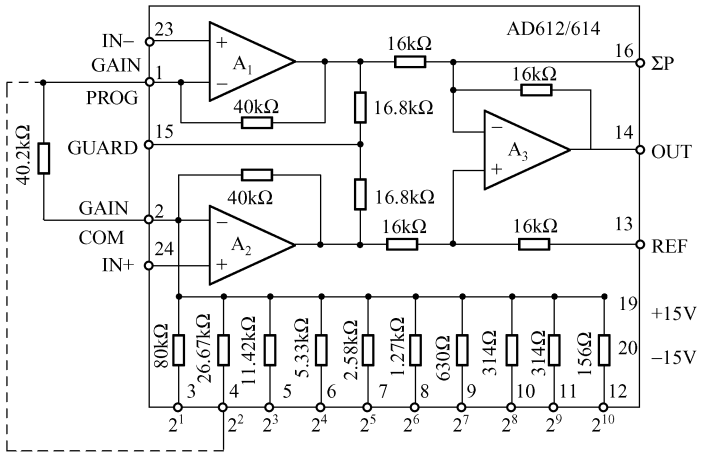



图 2-13　仪用放大器 AD612/614 电路原理

来调整放大器的增益。由于芯片内置高精度的薄膜电阻采用了激光自动修刻工艺，且将电阻阻值做了精心安排，故将引脚 1 分别连接到引脚 3～10 后就可分别形成 $2^1 \sim 2^8$ 的增益关系。而若得到 $2^9$ 和 $2^{10}$的增益关系，则需分别连接引脚 1、引脚 10、引脚 11 和引脚 1、引脚 10、引脚 11、引脚 12。此外，引脚 15 用于驱动屏蔽，即将其驱动后连接到输入电缆的屏蔽层可以提高共模抑制比，减少噪声的影响。引脚 13 作为参考信号，可为输出提供希望的任意直流电平偏置。从引脚 16 引入输出反馈信号，可将输出引线也包括在输出级的控制闭环内，从而从根本上提高放大器带动负载的能力。

### 2.2.2　隔离放大器

直流放大器本身的共模电压通常受电源电压的限制，一般不会超过±15V。然而在数据采集器实际工作的复杂环境中，由于用电设备绝缘不良、分布电容及静电感应等因素的影响，以及测量原理本身的需要，被测信号就有可能被加入一个很高的共模电压。为了避免元件在这种情况下发生损坏，同时也为了保证被采集数据的测量精度不受共模电压的影响，采用隔离放大器是一种安全可靠和简单有效的方法。

隔离放大器的作用是将输入信号、输出信号和供电电源三者相互之间电的联系统统隔断，而同时又要将被测输入信号正确地传递到输出，故具体设计和选用隔离放大器时需要综合考虑如下一些性能指标：

（1）传递增益的线性度(直接影响数采的分辨率)。

（2）传递信号的频响带宽限制(直接影响数采的速度)。

（3）隔离电压的幅度限制(关系到安全性)。

（4）共模抑制比(关系到数采的精度)。

（5）泄漏电流(或等效泄漏电阻与泄漏电容)。

通常采取的隔离方式有两种，即光电耦合方式和电磁耦合方式(变压器耦合方式)。现在这两种方式都可直接选用市场上提供的集成电路产品，成本都不很高，使用也很方便。

### 1. 光电耦合式隔离放大器

图 2-14 是一个光电耦合式隔离放大器的电路原理图，其核心芯片采用美国 EMENS 公司开发的 IL300 线性光电耦合器，由一个红外发光二极管和两个做在一起且彼此隔离的红外接收二级管组成。这种结构使两个接收管同时接收同一个 LED 管发出的光通量，且具有相同的温漂和时漂特性。这样若利用其中一个接收管输出，而利用另一个接收管反馈，就可大大提高整个隔离器的线性度和温漂、时漂指标。

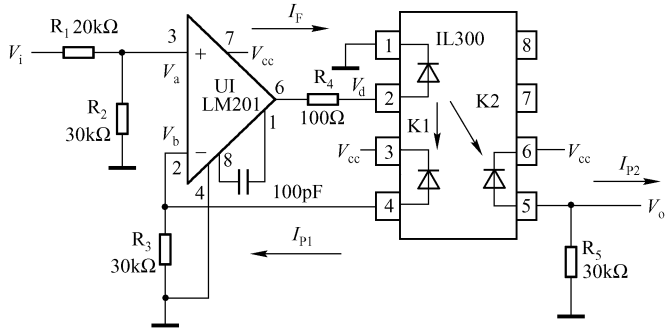



图 2-14　光电耦合式隔离放大器的电路原理

IL300 的基本特性指标见表 2-2，其红外发射管的驱动电流 $I_F$ 的工作范围为 0.5～40mA，但其最佳线性区是在 $I_F$ 为 5～20mA 时取得，并伴随 1.25～1.5V 的管压降 $V_d$，此时它的光能量输出受温度影响的变化是-0.5%/℃。IL300 红外接收二极管以电流源的方式工作，输出电流取决于 LED 发光管发出的光通量大小以及负载电阻的情况，其开路电压约为 0.5V。

**表 2-2　IL300 的基本特性指标**

| | | |
|---|---|---|
| 反馈增益 | $K_1=I_{P1}/I_F=$ 0.0036～0.011 | 典型值为 0.007 |
| 前向增益 | $K_2=I_{P2}/I_F=$0.0036～0.011 | 典型值为 0.007 |
| 传递增益 | $K_3=K_2/K_1=$0.56～1.65 | 典型值为 1.0 |
| 传递增益线性度 | $\Delta K_3=\pm$0.25% | 信号带宽　$BW_{-3dB}=200kHz$ |
| 隔离电压 | WTV=4420V | 共模抑制比　CMRR=130dB |
| 漏电阻 | $R_{10}=100G\Omega$ | 漏电容　$C_{10}=1pF$ |

图 2-15 是针对图 2-14 所示电路建立的系统框图，依据反馈控制原理，可得系统闭环的输出/输入关系为

$$V_o=K_2\cdot R_5\cdot\frac{K/R_4}{1+K_1\cdot R_3\cdot K/R_4}\cdot V_a-K_2\cdot R_5\cdot\frac{1/R_4}{1+K_1\cdot R_3\cdot K/R_4}\cdot V_d \tag{2-12}$$

显然，只要运算放大器 U1 的开环增益 $K$ 足够大，并适当调整外接电阻 $R_3$ 和 $R_5$，就能使输出信号 $V_o$ 很好地跟踪输入信号 $V_i$ 的变化，并有效地消除参数干扰引起的 $V_d$ 作用。

**例 2-1**　设某数据采集器用于监视 0～5V 的直流信号，且要求输出为 0～3V，为隔离其

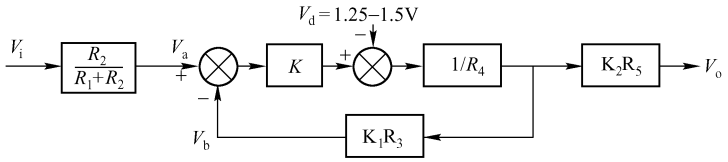



图 2-15　光电隔离器的系统框图

输入与输出之间的电联系，采用图 2-14 所示的隔离放大器电路，试求图中各元件的参数。

**解：**如图 2-14 所示，LED 的驱动电流 $I_F$ 由运算放大器 U1 控制，它在 LED 内部产生的光通量分别经 $K_1$ 和 $K_2$ 通道传输，被光电管接收后产生电流 $I_{F1}$ 和 $I_{F2}$，同时在电阻 $R_3$ 和 $R_5$ 上产生电压降 $V_b$ 和 $V_o$。前者由于运放 U1 的高放大倍数及高输入阻抗下 $I_F$ 的反馈作用，迫使 $V_a=V_b$，即

$$V_a=V_b=I_{P1}\cdot R_3=K_1\cdot I_F\cdot R_3$$

而后者则经 $K_2$ 通道输出，并有输出电压

$$V_0=I_{P2}R_5=K_2I_FR_5=K_2\frac{V_a}{K_1R_3}R_5=K_3\frac{R_5}{R_3}V_a=K_3\frac{R_5}{R_3}\cdot\frac{R_2}{R_1+R_2}V_i \tag{2-13}$$

根据上述基本关系，参数的确定可以归纳为如下计算过程。

1）确定反馈电阻 $R_3$

由 $I_F=5\sim20$mA 的 LED 最佳线性范围，选择驱动电流。取 $I_F=14$mA，则有 $I_{P1}=K_1\cdot I_F=0.007\cdot14\text{mA}=100\mu\text{A}$。因此，若取 $V_a=3$V，即电源电压的 60%左右，则 $R_3=\frac{V_b}{I_{P1}}=\frac{V_a}{I_{P1}}=\frac{3\text{V}}{100\mu\text{A}}=30\text{k}\Omega$。

2）确定限流电阻 $R_4$

由 LED 正常压降为 1.25～1.5V 的参数条件，试取 $R_4=100\Omega$，这意味着 U1 的输出需要 $1.5\text{V}+I_F\cdot R_4=1.5\text{V}+14\text{mA}\cdot100\Omega=2.9\text{V}$。这正是通常运放输出的一般工作范围，故认为试取合适。

3）确定输出电阻 $R_5$

由式(2-13)，$V_0=K_3\frac{R_5}{R_3}V_a$，可得 $R_5=\frac{R_3}{K_3}\frac{V_o}{V_a}=\frac{30\text{k}\Omega}{1}\cdot\frac{3\text{V}}{3\text{V}}=30\text{k}\Omega$

4）确定输入分压电阻 $R_1$ 和 $R_2$

由图可见，$V_a=\frac{R_2}{R_1+R_2}V_i$，即 $\frac{V_i}{V_a}=1+\frac{R_1}{R_2}$。当 $V_i=5$V 和 $V_a=3$V 时，可取 $R_1=20\text{k}\Omega$ 和 $R_2=30\text{k}\Omega$。

IL300 手册给出的参考测试曲线如图 2-16 所示，可以看出其线性度和频响带宽对一般数据采集的要求来说都是相当不错的。

### 2. 变压器耦合式隔离放大器

电磁耦合式隔离放大器(或称变压器耦合式隔离器)是隔离放大器的另一种主要形式，其历史比光电耦合式隔离器更长。一般说来，光电耦合式隔离器的频响较高，适应信号变化速度较快的场合；而变压器耦合式隔离放大器的线性度较高且范围较宽，适应于信号精度要求

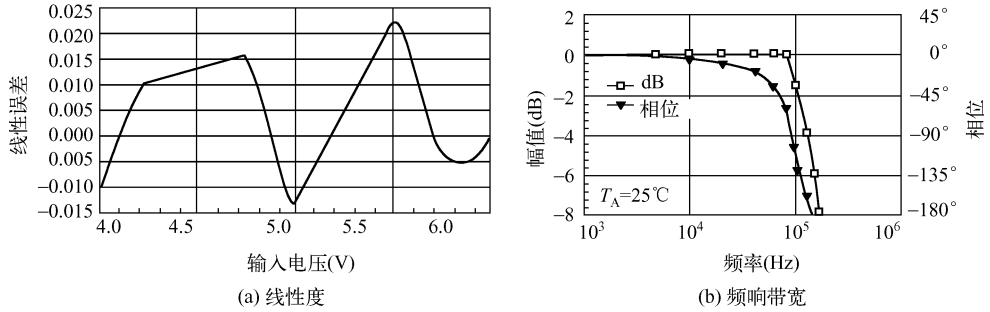



图 2-16　IL300 手册的参考测试结果

较高的场合。

如图 2-17 所示，某典型变压器耦合式隔离放大器可分成彼此电位相互隔离的 3 个模块，即输入、输出和电源模块。它的核心元件是一个磁耦合变压器，其原端绕组被电源模块中的一个 150kHz 的功率振荡器所激励。这个 150kHz 的非对称方波信号通过变压器磁路耦合到调制器和两个解调器上，并作为它们实现调制和解调任务的载波工作信号。在输入模块中，电源本身也是通过这个变压器逆变这个方波信号得到的。输入模块中调制器的调制信号是经过阻容滤波和运放 A1 放大的输入信号；而该模块内一个解调器的输出也加到 A1 的输入端作为其反馈信号。这样的反馈作用与前面分析的光电耦合式隔离放大器的情况非常相似，可以大大改善整个隔离器的线性度和时漂、温漂特性。在输出模块中另一个解调器则将变压器耦合过来的调制信号解调后通过滤波和运放 A2 输出。输入模块的增益 $G_1$ 和输出模块的增益 $G_2$ 都可以通过外接电路来加以调节。$G_1$ 取决于 $R_a$ 和 $COM_{IN}$ 两端所加的反馈分压电阻，$G_2$ 取决于引脚 FEED-BACK 在 Vout 和 $Com_{out}$ 两端的分压关系。整个隔离器的增益 $G=G_1G_2$。

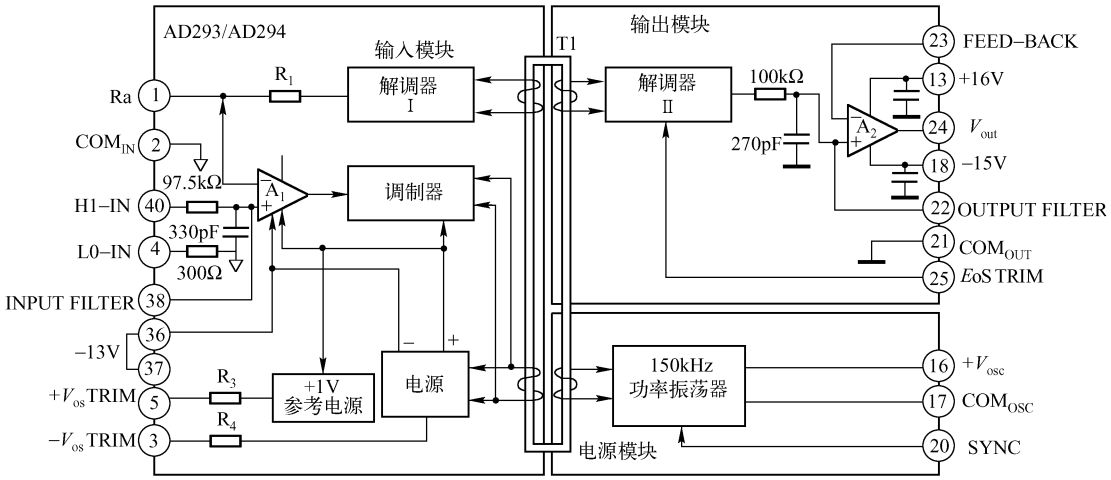



图 2-17　变压器耦合式隔离放大器原理示意图

AD294 是 Analog 公司一种低价位、高可靠性的隔离器产品。它封装在一个 40 引脚的芯片内，消除了传统绕线磁变压器的弊病，从而使性能得到很大的提高。

具体技术指标如下：

输入模块增益　$G_1=1\sim100\text{V/V}$　　增益线性度　$\Delta G=0.1\%\sim0.05\%$

输出模块增益　$G_2=1\sim10\text{V/V}$　　隔离电压　WTV = 2500V

总增益　$G=G_1G_2=1\sim1000\text{V/V}$　　共模抑制比 CMRR = 100dB

输入电压范围　RV = ±10V　　最大漏电流 $I_{IO}=2\mu\text{A}$

信号带宽范围　$BW_{-3dB}=2.5\text{kHz}$（小信号），200Hz(大信号)

AD294 的基本连接方式如图 2-18 所示。

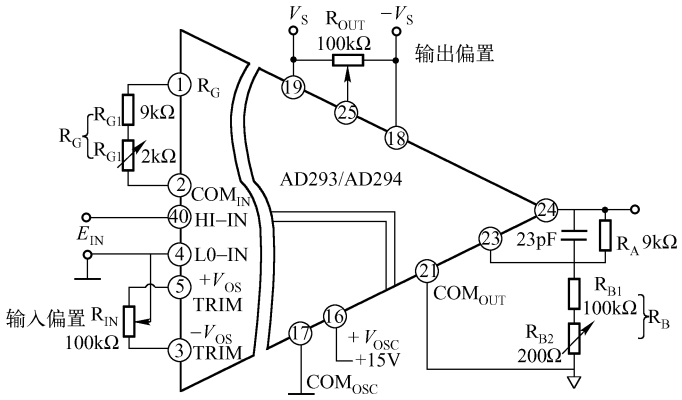



图 2-18　AD294 应用连接图

其中输入模块增益和输出模块增益分别为

$$G_1=1+\frac{100\text{k}\Omega}{R_G},\qquad R_G>1\text{k}\Omega$$

$$G_2=1+\frac{R_A}{R_B},\qquad 1\leqslant G_2\leqslant10$$

为了说明增益设置方法，可让 $G_1=G_2=10$，则应取

$$R_G=\frac{100\text{k}\Omega}{G_1-1}=\frac{100\text{k}\Omega}{10-1}=10.1\text{k}\Omega$$，取 $R_{G1}=9\text{k}\Omega$，$R_{G2}$为 0~2kΩ 可变电阻。

$$R_B=\frac{R_A}{G_2-1}=\frac{9\text{k}\Omega}{10-1}=1\text{k}\Omega$$，取 $R_A=9\text{k}\Omega$，$R_{B1}=900\Omega$，$R_{B2}$为 0~200Ω 可变电阻。

### 2.2.3　滤波器的基本概念

滤波器用于信号的处理，它可以选通输入信号的某些频率分量，同时抑制另外一些频率分量。一般称被选通的频段为通带，被抑制的频段为阻带。根据不同的通带和阻带要求，滤波器可分为低通、高通、带通和带阻等几种不同的类型，如图 2-19 所示。

在数据采集器的通道中，滤波器主要用来消除信号中的有害成分。这些有害成分主要是噪声，可能来自外部环境，如从传感器连接长线耦合的电磁干扰；也可能来自数采器内部器件，如电子放大器内部热噪声。噪声信号通常是随机信号，而且其频率范围往往高出正常有用信号的带宽范围很多，因此可以使用低通滤波器进行有效的抑制。对于

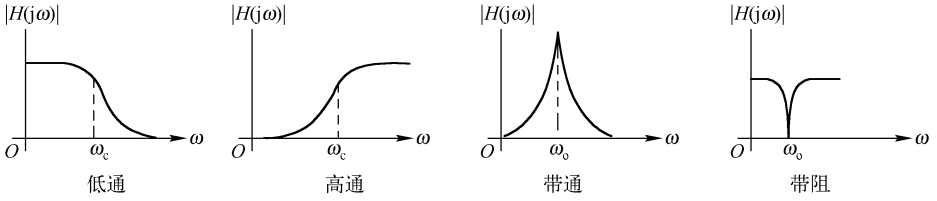



图 2-19 用于低通、高通、带通和带阻的滤波器特性

某些特殊的干扰，如 50Hz 电源交流噪声，则可用相应的带阻滤波器处理。对动态信号的数据采集来说，还有一个采样引起的“混叠”问题(将在 2.4 节详细讨论)，也需要低通滤波器的作用。

理想低通滤波器要求通带增益恒为 1，阻带增益为 0，通带与阻带之间的过渡带则要求越窄越好。然而，用线性电子器件构成的滤波器不可能如此理想。一般说来滤波器传递函数的阶次越高，则幅频特性可以做得越陡，但相应的相频滞后就会越大，通带和阻带的频段内也有可能出现波动。目前，对滤波器的多年研究已经形成了一些成熟的设计模型，如巴特沃斯(Batterworth)滤波器、切比雪夫(Chebyshev)滤波器等。

如图 2-20 所示，巴特沃斯滤波器通带内幅频变化平稳、单调，但过渡带较宽；切比雪夫滤波器尽管通带存在波动，但过渡带陡得多。另外，还有其他一些典型的滤波器模型，如反切比雪夫滤波器(阻带波动)、椭圆滤波器(通带、阻带均有波动，但过渡带更窄)和贝塞尔滤波器(线性的相频特性)等。

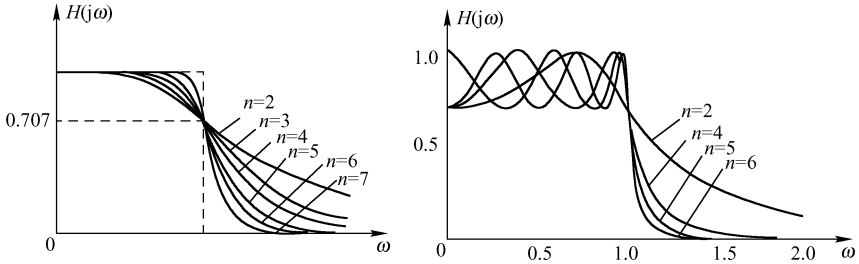



图 2-20 巴特沃斯滤波器与切比雪夫滤波器的幅频特性

滤波器的频率特性取决于其传递函数，传递函数可以写成算子 $s$ 的分式多项式：

$$G(s)=\frac{V_o(s)}{V_i(s)}=\frac{b_0+b_1 s+\cdots b_{n_b}s^{n_b}}{1+a_1 s+a_2 s^2+\cdots+a_{n_a}s^{n_a}}$$

用 $j\omega$ 代替式中算子 $s$，就可以得到相应的滤波器频率特性。

对二阶滤波器，传递函数的特征多项式可以描述成

$$p(s)=s^2+Bs+C \tag{2-14}$$

且可进一步定义极偶频率(Pole-Pair Frequency)为

$$\omega_0=\sqrt{C} \tag{2-15}$$

定义品质因数(Pole-Pair Quality)为

$$Q=1/2\zeta=\sqrt{C}/B \tag{2-16}$$

$Q$ 值越大，表明阻尼系数越小，则谐振幅度越高。当 $Q>10$ 时，滤波器需要采用特殊的电路方能实现。

在表 2-3 和表 2-4 中，不加推导地直接给出了不同阶次条件下巴特沃斯和切比雪夫这两种滤波器的幅频参数和特征多项式，以便设计滤波器时参考。

**表 2-3　巴特沃斯滤波器的幅频特性和传递函数特征多项式**

| 阶　次 | $\lvert H(j\omega)\rvert$ | $H(s)$的特征多项式（归一化后） |
|---|---|---|
| 1 | $(1+\omega^2)^{-1/2}$ | $s+1$ |
| 2 | $(1+\omega^4)^{-1/2}$ | $s^2+1.414s+1$ |
| 3 | $(1+\omega^6)^{-1/2}$ | $s^3+2s^2+2s+1$ |
| 4 | $(1+\omega^8)^{-1/2}$ | $s^4+2.6131s^3+3.4142s^2+2.6131s+1$ |
| 5 | $(1+\omega^{10})^{-1/2}$ | $s^5+3.2361s^4+5.2361s^3+5.2361s^2+3.2361s+1$ |

**表 2-4　切比雪夫滤波器的通带波动系数和传递函数特征多项式**

| 阶　次 | 波 动 系 数 | $H(s)$的特征多项式 |
|---|---|---|
| 2 | 1dB | $s^2+1.0977s+1.1025$ |
| 2 | 3dB | $s^2+0.6449\ s+0.7079$ |
| 3 | 1dB | $(s^2+0.4942s+0.9942)(s+0.4942)$ |
| 3 | 3dB | $(s^2+0.2986s+0.8392)(s+0.2986)$ |
| 4 | 1dB | $(s^2+0.2791s+0.9865)(s^2+0.6737s+0.2794)$ |
| 4 | 3dB | $(s^2+0.1703s+0.9031)(s^2+0.4112s+0.1960)$ |
| 5 | 1dB | $(s^2+0.1789s+0.9883)(s^2+0.4684s+0.4293)(s+0.2895)$ |
| 5 | 3dB | $(s^2+0.1097s+0.9360)(s^2+0.2873s+0.3770)(s+0.1775)$ |

滤波器电路分为无源滤波和有源滤波两类。前者是指仅用电阻、电容和电感等无源元件实现的滤波器电路；而后者则是以运算放大器等有源元件为主，配以部分阻容元件构成的滤波器电路。近年来流行的开关电容型滤波器也属后一类。有源滤波器的主要优点在于它能够不使用电感(电感很难集成在硅片上，且做不大)，也不需要很大容量的电容；缺点是需要电源供电，而且需要考虑反馈引起的稳定性问题。滤波器的设计已有多年的研究历史和丰富的文献资料，各种滤波器集成电路也早已进入市场。这里将为读者提供基本的设计方法，以便使读者能够针对数据采集的基本要求，利用通用运算放大器、阻容元件及其他有关芯片，自行设计实际滤波器电路。下面就滤波器实现的几个具体问题加以讨论。

### 2.2.4　滤波器电路设计的归一化实现

从频率响应反推传递函数，再构建实现电路并不是一个直截了当的过程。首先有一个物理上可否实现的问题，在直接推出的电路中，阻容参数可能会出现负值。其次还有一个推出的结果是否唯一的问题，即同一问题可能推出多种结果。工程上常用的办法是将各种典型滤波器的频率响应、传递函数和电路实现一一列出，编成手册供用户选用。然而，对一种特定形式的滤波器，截止频率不同就可能有无数种参数配置，因此需要采用归一化的方法。即在

手册中只给出在归一化条件下的参数，例如在截止频率 $\omega_{cn}=1$ 和负载电阻 $R_{Ln}=1\Omega$ 的情况下电路中的各参数取值。而当实际需要一个截止频率为 $\omega_c$ 的滤波器接在负载电阻为 $R_L$ 的环境下时，则可根据需要先按归一化条件（如 $\omega_{cn}$ 和 $R_{Ln}$）选取电路及其元件的参数，然后根据实际要求的 $\omega_c$ 和 $R_L$，按一定规则解算这些归一化参数，最终得到各元件的实际参数。

设 $\omega_{cn}$ 和 $R_{Ln}$ 分别为归一化条件的截止频率和负载电阻，归一化实现电路中的电阻、电容和电感元件分别是 $R_{in}$，$C_{jn}$ 和 $L_{kn}$（这里下标 $i$、$j$、$k$ 是任意正整数，分别用以区别实现电路中存在的多个电阻、电容和电感元件）。先计算归一化解算系数 $K_L$ 和 $K_\omega$，即

$$\begin{aligned} K_L &= R_L / R_{Ln} \\ K_\omega &= \omega_c / \omega_{cn} \end{aligned} \tag{2-17}$$

然后，就可以按照下面介绍的一组公式解算归一化电路中的所有元件，即

$$\begin{aligned} R_i &= R_{in} \cdot K_L \\ C_j &= C_{jn} / K_L K_\omega \\ L_k &= L_{kn} \cdot K_L / K_\omega \end{aligned} \tag{2-18}$$

式中，$\omega_c$ 和 $R_L$ 是实际电路要求的截止频率和负载电阻，$R_i$、$C_j$ 和 $L_k$ 分别是实际电路中各电阻、电容和电感元件的参数。

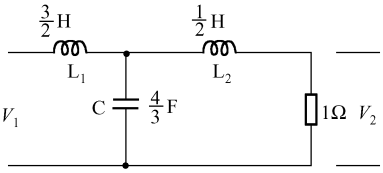



图 2-21　某标准低通滤波器原理图及其参数配置

**例 2-2**　已知为截止频率 $\omega_{cn}=1$ 和负载电阻 $R_{Ln}=1\Omega$ 所设计的标准低通滤波器如图 2-21 所示，图中所表明的各元件参数均为归一化参数，试求对截止频率 10kHz 和负载电阻 $R_L=600\Omega$ 的实际滤波器要求下的实际电路元件参数。

**解**：先计算两个归一化解算系数，依据式(2-17) 可得

$$K_L=\frac{R_L}{R_{Ln}}=\frac{600\Omega}{1\Omega}=600,\ K_\omega=\frac{\omega_c}{\omega_{cn}}=\frac{2\pi\cdot 10\,000\text{rad/s}}{1\text{rad/s}}=6.28\cdot 10^4$$

按规则，最终可得

$$L_1=\frac{K_L}{K_\omega}\cdot\frac{3}{2}\text{H}=\frac{600}{6.28\cdot 10^4}\cdot 1.5\text{H}=14.3\text{mH}$$

$$L_2=\frac{K_L}{K_\omega}\cdot\frac{1}{2}\text{H}=\frac{600}{6.28\cdot 10^4}\cdot 0.5\text{H}=4.8\text{mH}$$

$$C=\frac{1}{K_L\cdot K_\omega}\cdot\frac{4}{3}\text{F}=\frac{4/3}{600\cdot 6.28\cdot 10^4}\text{F}=0.0354\mu\text{F}$$

**例 2-3**　试求图 2-22 中阻容网络滤波器在截止频率 $\omega_c=2\pi\cdot 10\text{krad/s}$ 和负载电阻 $R_L=600\Omega$，且直流增益 $K>0.75$ 的情况下的电路元件参数。

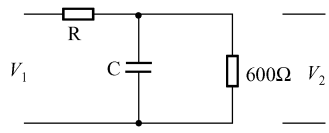



图 2-22　低通阻容网络滤波器

**解**：先看归一化条件 $\omega_{cn}=1$ 和 $R_{Ln}=1\Omega$ 下的情况，为满足直流增益和截止频率的要求，可有下列变量关系：

$$K=\frac{R_L}{R_L+R_n}=0.75 \text{ 和 } \omega_{cn}=\frac{1}{C_n\cdot(R_L\cdot R_n)/(R_L+R_n)}=1$$

于是，可取 $R_n=1/3\Omega$，$C_n=4\mathrm{F}$。上述结果实际上也可通过查表得到。

再由要求的 $\omega_c$ 和 $R_L$，依据式(2-17)可得归一化解算系数为

$$K_L=\frac{R_L}{R_{Ln}}=\frac{600\Omega}{1\Omega}=600,K_\omega=\frac{\omega_c}{\omega_{cn}}=\frac{2\pi\cdot 10\,000\mathrm{rad/s}}{1\mathrm{rad/s}}=6.28\cdot 10^4$$

这样按式(2-18)所示规则，最终可得 $R=R_nK_L=200\Omega$，$C=C_n/K_LK_\omega=0.1\mu\mathrm{F}$。

### 2.2.5　Bach 滤波器的设计方法

早在 20 世纪 40 年代，Bach 滤波器就以电子管放大器的形式出现。尽管从减少使用运算放大器数目的角度来考虑，Bach 方法并非最有效的方法，然而由于它十分容易地满足滤波器任意数目极点的配置需要，因此在数据采集器中实现巴特沃斯或切比雪夫滤波器都十分方便。图 2-23 给出了 1~4 阶低通滤波器的实际实现电路，更高阶的要求可以通过简单地串联组合来完成。

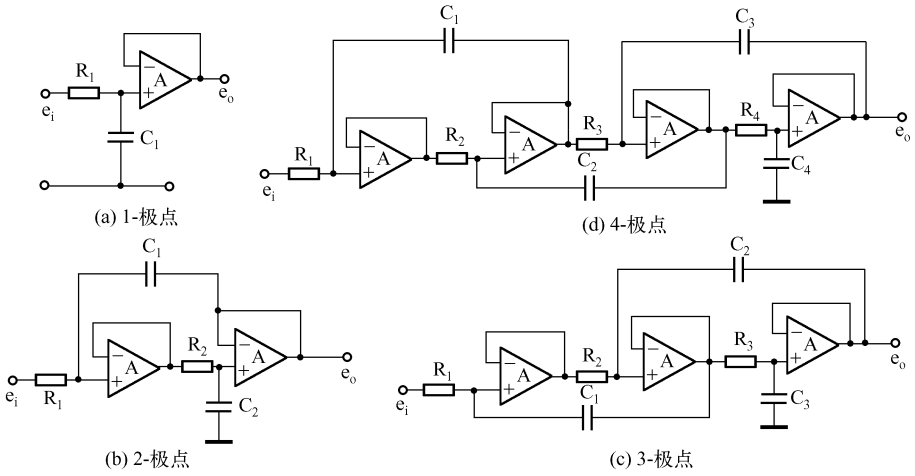



图 2-23　1~4 阶低通滤波器的 Bach 实现方法

如果定义参数 $\omega_1=1/R_1C_1$、$\omega_2=1/R_2C_2$、$\omega_3=1/R_3C_3$、$\omega_4=1/R_4C_4$，则图 2-23 中各阶低通滤波器输入与输出之间的传递函数就可清晰地写为

$$T_1(s)=\frac{\omega_1}{s+\omega_1}$$

$$T_2(s)=\frac{\omega_1\omega_2}{s^2+\omega_1 s+\omega_1\omega_2}$$

$$T_3(s)=\frac{\omega_1\omega_2\omega_3}{s^3+\omega_2 s^2+\omega_1\omega_2 s+\omega_1\omega_2\omega_3}$$

$$T_4(s)=\frac{\omega_1\omega_2\omega_3\omega_4}{s^4+\omega_1 s^3+\omega_1\omega_2 s^2+\omega_1\omega_2\omega_3 s+\omega_1\omega_2\omega_3\omega_4}\tag{2-19}$$

选择较大的 $R_1$ 值可以提高滤波器的输入阻抗，而滤波器的输出阻抗取决于末级输出阻抗，通常可以做得很小。但实际应用中确有必要时，也可再增加一个输出级来获得更好的负载能力。对上述滤波器实现，电容 C 的值不需要特别大，就能实现截止频率为几赫兹的低频

滤波器。下面举例说明这种方法的应用。

**例 2-4** 试设计一个巴特沃斯低通滤波器，要求截止频率在 200Hz，且其后高频段的渐近衰减率为-18dB/倍频(-60dB/十倍频程)。信号源内阻为 100Ω，负载电阻为 10kΩ。

**解：** 对巴特沃斯滤波器，截止频率后的幅频衰减率为 6*n*dB/倍频程。这里，参数 $n$ 是指滤波器传递函数的阶次，需要 3 阶滤波器才能达到衰减率为-18dB/倍频的要求。同时，根据表 2-2 给出的巴特沃斯低通滤波器的归一化传递函数，可以直截了当地推出解算后用于转折频率在 $\omega_c$ 处的滤波器传递函数公式：

$$T_1(s)=\frac{K_1}{s+\omega_c}$$

$$T_2(s)=\frac{K_2}{s^2+1.414\omega_c s+\omega_c^2}$$

$$T_3(s)=\frac{K_3}{s^3+2\omega_c s^2+2\omega_c^2 s+\omega_c^3}$$

$$T_4(s)=\frac{K_4}{s^4+2.6131\omega_c s^3+3.4142\omega_c^2 s^2+2.6131\omega_c^3 s+\omega_c^4} \tag{2-20}$$

将式(2-20)与式(2-19)进行比较，并选用 3 阶滤波器，可得

$$T_3(s)=\frac{K_3}{s^3+2\omega_c s^2+2\omega_c^2 s+\omega_c^3}=\frac{\omega_1\omega_2\omega_3}{s^3+\omega_1 s^2+\omega_1\omega_2 s+\omega_1\omega_2\omega_3}$$

从而解得

$$\omega_2=2\omega_c=2.2\pi\cdot 200=8\pi\cdot 10^2$$

$$\omega_2=2\omega_c^2/\omega_1=\omega_c=4\pi\cdot 10^2$$

$$\omega_3=\omega_c^3/\omega_1\omega_2=\omega_c/2=\pi\cdot 10^2$$

为了取得与信号源电阻(100Ω)相比相对较高的输入阻抗，取 $R_1=10\text{k}\Omega$。同时为简便起见，也取 $R_2=R_3=10\text{k}\Omega$，则可得到

$$C_1=1/R_1\omega_1=0.04\mu\text{F}\qquad C_2=1/R_2\omega_2=0.08\mu\text{F}\qquad C_3=1/R_3\omega_3=0.16\mu\text{F}$$

### 2.2.6 开关电容滤波器 SCF

开关电容滤波器利用器件内部模拟开关的通断过程和电容充放电过程来模拟一个阻值与开关切换周期成比例的等效电阻。这个等效电阻与做在同一芯片内的运算放大器及电容器配合，共同构成一个积分常数与开关切换周期成比例的积分器。以这种积分器为核心，配接外部反馈电阻，可形成一种滤波中心频率与开关切换频率成比例的滤波器。这种滤波器通常用于低频场合，但典型产品也能达到 50kHz 左右，而使用 BIT-CMOS 技术则可提高到 500kHz 以上。开关电容滤波器易于实现集成电路化，而且易于实现程控，其中心频率调节方便、准确，故在数据采集器中得到十分广泛的应用。

开关电容积分器的工作原理如图 2-24 所示，其中图 2-24(a)是一个普通积分器，由于运放反向输入端的“虚地”作用，其输出电压 $V_o$ 等于积分电容两端的电压，考虑输入电流与运放反馈回路电容的充电电流相等，则有

$$V_o(t)=-\frac{1}{C}\int I_2(t)\,\mathrm{d}t=-\frac{1}{C}\int I_1(t)\,\mathrm{d}t=-\frac{1}{RC}\int V_i(t)\,\mathrm{d}t \tag{2-21}$$

从而形成积分作用，且积分时间常数 $\tau = RC$。

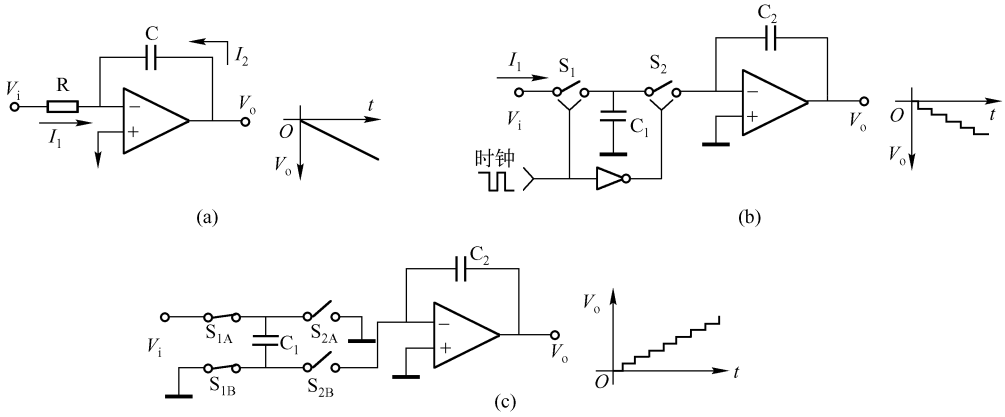



图 2-24　开关电容积分器的工作原理

再看图 2-24(b)，开关电容两端电路被电子开关 $S_1$ 和 $S_2$ 交替切换。在开关切换控制时钟信号 $f_{clk}$ 的前半周期 $T$，$S_1$ 接通、$S_2$ 关断时，输入信号 $V_i$ 对开关电容 $C_1$ 充电直至其电量 $Q = C_1 V_i$。在时钟信号的后半周期 $T$，$S_1$ 关断、$S_2$ 接通时，在 $C_1$ 上的电荷通过电容 $C_2$ 和运算放大器放电直至其电量为 0。于是在开关切换的整个周期通过 $S_1$ 和 $S_2$ 的平均电流为

$$I_1 = \frac{Q_{c1}}{T} = \frac{C_1 \cdot V_i}{T} = C_1 \cdot f_{clk} \cdot V_i$$

而开关电容 $C_1$ 和开关 $S_1$、$S_2$ 的作用则等效于图 2-24(a)中电阻 R 的作用，其等效电阻为

$$R = \frac{V_i}{I_1} = \frac{V_i}{C_1 \cdot f_{clk} \cdot V_i} = \frac{1}{C_1 \cdot f_{clk}}$$

因此积分器的积分时间常数为

$$\tau = RC_2 = \frac{C_2}{C_1 \cdot f_{clk}}$$

它只取决于开关切换控制时钟 $f_{clk}$ 的频率和电容 $C_1$、$C_2$ 的比值。由于 $C_1$、$C_2$ 可以做在同一芯片上，故易于稳定；而控制时钟可以外接，使得调整十分方便。图 2-24(c)是一个非反向积分滤波器的开关电容式的实现方法。

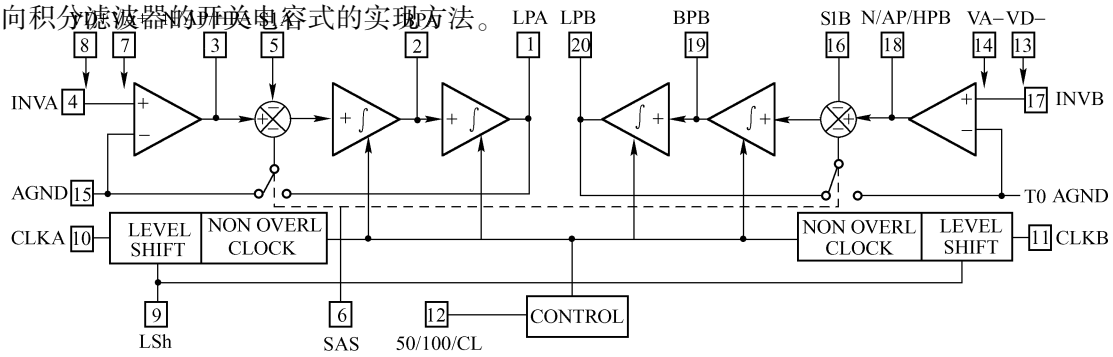



图 2-25　MF10 开关电容积分器内部结构

MF10 是一种典型的开关电容滤波器芯片，它由两个完全独立的 CMOS 二阶通用有源滤波单元组成，每个单元与一个外接时钟 $f_c$ 和若干个反馈电阻组合，可以实现诸如巴特沃斯、切比雪夫等各种标准滤波器功能。图 2-25 是 MF10 的结构框图，图中 A、B 两组完全相同，均以两级非反向的开关电容积分器为主体。引脚 INV、N/AP/HP、BP 和 LP 作为输入、输出和反馈接线端口，可根据不同工作方式连接反馈电阻。引脚 S、S1 用于确定不同的工作方式。引脚 CLK 是时钟输入端，由引脚 LSh 控制时钟电平。仅当单电源+10V 下 CLK 为 CMOS 电平时，LSh 接+5V，而其他情况接地。引脚 50/100/CL 用来调整外接时钟频率 $f_{clk}$ 与滤波器中心频率 $f_0$ 之间的关系，当其为高电平时，$f_0=f_{clk}/100$；而当其为低电平时，$f_0=f_{clk}/100$。

MF10 有 6 种工作方式，其中方式 3 为通用状态变量反馈滤波器，是为通用目的推荐的一种工作方式。在这种方式下，滤波器中心频率与时钟频率之间的关系可用外接反馈电阻加以调整，其工作原理及其相应的传递函数方框图如图 2-26 所示。

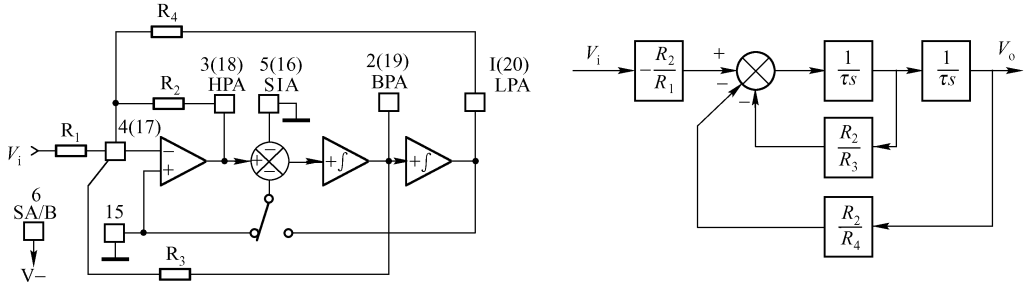



图 2-26　MF10 的工作方式 3 及其等效传递函数方框图

整理后可以得到闭环传递函数为

$$\frac{V_o(S)}{V_i(S)}=\frac{-R_2/R_1}{\tau^2 s^2+(R_2/R_3)\cdot\tau s+R_2/R_4}=\frac{-R_4/R_1}{(R_4/R_2)\cdot\tau^2 s^2+(R_4/R_3)\cdot\tau s^2+1} \tag{2-22}$$

这样，可以算出极偶角频率为

$$\omega_0=\frac{1}{\tau}\cdot\sqrt{\frac{R_2}{R_4}}=\frac{C_1}{C_2}\cdot f_{clk}\cdot\sqrt{\frac{R_2}{R_4}} \tag{2-23}$$

由于手册上已经给出实际中心频率、品质因数和通带增益的计算公式如下：

$$f_0=\frac{1}{50\text{ 或 }100}\cdot\sqrt{\frac{R_2}{R_4}}\cdot f_{clk} \tag{2-24}$$

$$Q=\frac{R_3}{\sqrt{R_2\cdot R_4}} \tag{2-25}$$

$$H_{OLP}=R_4/R_1 \tag{2-26}$$

故可认为其内部已经设定 $C_1/C_2=2\pi/(50\text{ 或 }100)$。

实际应用时，可以将上述 2 阶基本单元串联起来使用。下面举例说明如何应用 MF10 来实现一个低通滤波器。

**例 2-5**　假设系统需要一个 4 阶切比雪夫低通滤波器，要求波动系数 RW = 1dB，直流增

益为 1，截止频率为 1000kHz，考虑用两个 MF10 的 2 阶开关电容滤波器串联完成。

**解**：由表 2-3 可以直接查到组成一个 4 阶切比雪夫低通滤波器的两个归一化了的 2 阶传递函数的特征多项式。当波动系数为 1dB 时，它们分别是 $s^2+0.2791s+0.9865$ 和 $s^2+0.6737s+0.2794$。通常为了确保谐振频率附近的稳定，总是将 $Q$ 值较低的环节放在前级，从而经解算归一化过程的计算结果如下。

第一级环节：

$$f_{0A}=\sqrt{0.2794}\cdot 1000=529\text{Hz}$$

$$Q_A=\sqrt{0.2794}/0.6737=0.785$$

$$H_{0A}=1$$

第二级环节：

$$f_{0B}=\sqrt{0.9865}\cdot 1000=993\text{Hz}$$

$$Q_B=\sqrt{0.9865}/0.2791=3.559$$

$$H_{0B}=1$$

选择 MF10 的工作方式 3，并将时钟频率 $f_{clk}$ 选为 100kHz，引脚 50/100/CL 接低电平，此时需要用反馈电阻来调整两个 2 阶环节本身的中心频率。

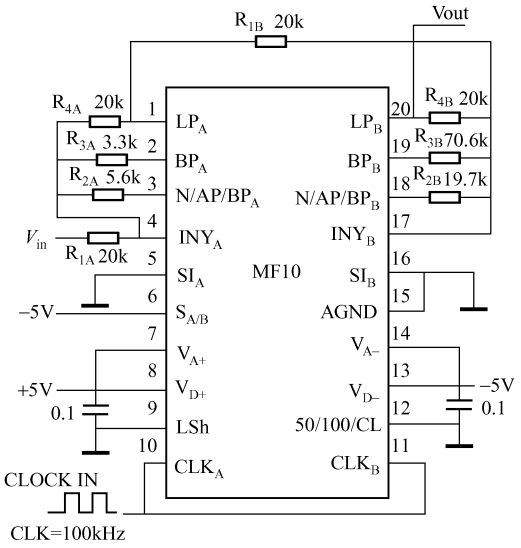



图 2-27　MF10 的外部接线

取输入电阻 $R_{1A}=20\text{k}\Omega$，考虑 $H_{0A}=1$，可得

$$R_{4A}=R_{1A}\cdot H_{0A}=20\text{k}\Omega\cdot 1=20\text{k}\Omega$$

$$R_{2A}=R_{4A}\cdot(100\cdot f_{0A}/f_{clk})^2$$

$$=20\text{k}\Omega\cdot(100\cdot 529/100\text{k}\Omega)^2$$

$$=5.6\text{k}\Omega$$

$$R_{3A}=Q_A\cdot\sqrt{R_{2A}\cdot R_{4A}}$$

$$=0.785\cdot\sqrt{20\text{k}\Omega\cdot 5.6\text{k}\Omega}=8.3\text{k}\Omega$$

同理，可得

$R_{1B}=R_{4B}=20\text{k}\Omega$，$R_{2B}=19.7\text{k}\Omega$，$R_{3B}=70.6\text{k}\Omega$

实际电路接线如图 2-27 所示。

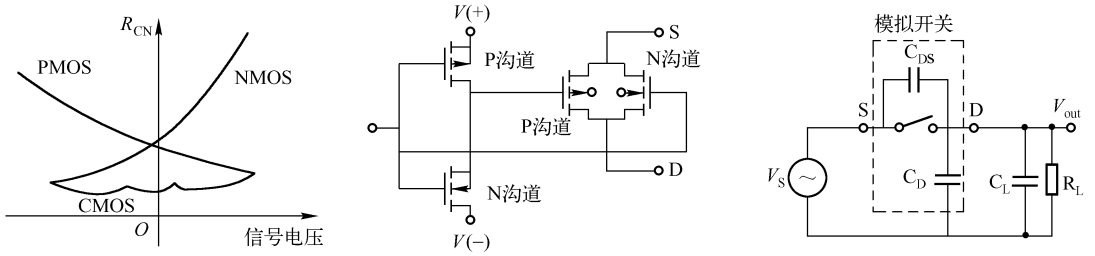
## 2.2.7　多路通道开关

数据采集器选择切换多个模拟通道的开关主要有机械触点的继电器和模拟电子开关两类。两者相比，前者的特点是通/断电阻特性好，接通电阻小(典型参数 $R_{ON}$ 小于 0.1Ω)、关断电阻高，缺点是工作速度低、体积较大。

干簧继电器作为前者的代表，曾被广泛应用在早期巡回监测一类的测试系统中。它的接通电阻 $R_{ON}<0.1\Omega$，关断电阻 $R_{OFF}>10^{10}\Omega$，击穿电压可达上千伏。同时其可靠耐用，额定电流负载下可重复工作 $10^{10}$ 次。然而，干簧继电器的簧片在与铜线接触时会产生一个大约 200μV/℃的温差电势，而且开关的速度被限制在 250 次/秒左右。

与此相对应的是近年来电子技术的快速发展，使模拟电子开关应用得越来越多。其中最

常使用的是 CMOSFET 开关，它具有功耗小、速度快的特点。如图 2-28 所示，由于并联的 P 沟道和 N 沟道器件提供对称的电阻，故当输入信号变化时仍能保持导通电阻的稳定。特别是半导体工艺可以把开关的驱动电路和译码控制电路、甚至 A/D 转换器都集成到一个芯片中，因此使用起来十分方便。




(a) 电阻特性　　(b) 内部结构　　(c) 等效电路

图 2-28　CMOSFET 开关的特性

图 2-28 也给出了模拟电子开关的等效电路，影响信号传输的主要参数是开关的导通电阻 $R_{ON}$和泄漏电流 $I_{OFF}$。显然，当开关负载阻抗较大、信号源内阻较小时，导通电阻 $R_{ON}$的分压作用和泄漏电流 $I_{OFF}$的分流作用都较小。模拟开关工作在高频切换场合时，由于极间电容和分布电容的作用将影响模拟开关的通/断时间和通/断质量，从而也会限制多路开关的切换速度。实际模拟多路开关做成不同的形式，表 2-5 给出了部分常用模拟电子多路开关芯片的主要性能指标。

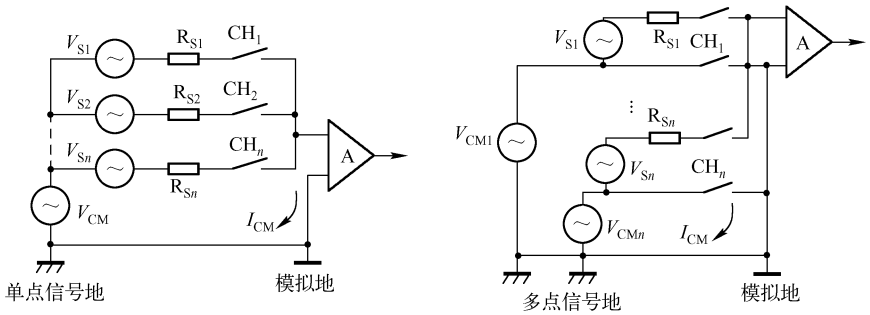
**表 2-5　常用模拟电子多路开关性能**

| 型　号 | $R_{ON}$ | $I_{OFF}$ | $T_{ON}$ | $T_{OFF}$ | 开关数 | $V_{EE}-V_{CC}$ | $I_C$ |
|---|---|---|---|---|---|---|---|
| CD4051B | 200~300 | 20nA | 50ns | 40ns | 双向单 8 选 1 | ±7.5V | ≤30mA |
| CD4052B | 200~300 | 20nA | 40ns | 40ns | 双向双 4 选 1 | ±7.5V | ≤30mA |
| CD4053B | 200~300 | 20nA | 40ns | 40ns | 双向三 2 选 1 | ±7.5V | ≤30mA |
| CD4066 | 125 | 20nA | 40ns | 40ns | 双向四 1 选 1 | ±7.5V | ≤30mA |
| CD4067B | 200~300 | 20nA | 40ns | 40ns | 双向单 16 选 1 | ±7.5V | ≤30mA |
| CD4097B | 200~300 | 20nA | 40ns | 40ns | 双向双 8 选 1 | ±7.5V | ≤30mA |
| CD7501D1 | 170~300 | 0.2nA | 150ns | 0.8μs | 单向单 8 选 1 | ±15V | ≤35mA |
| CD7502D1 | 170~300 | 0.2nA | 150ns | 0.8μs | 单向双 4 选 1 | ±15V | ≤35mA |
| CD7506D1 | 350~400 | 0.05nA | 100ns | 0.8μs | 单向单 16 选 1 | ±15V | ≤35mA |
| CD7507D1 | 350~400 | 0.05nA | 100ns | 0.8μs | 单向双 8 选 1 | ±15V | ≤35mA |
| CD7510D1 | 75 | 0.5nA | 180ns | 350ns | 单向四 1 选 2 | ±15V | 50mA |
| CD7512D1 | 75 | 5nA | 180ns | 350ns | 单向双 1 选 2 | ±15V | 50mA |
| DG408 | 100 | 5nA | 150ns | 150ns | 双向单 8 选 1 | ±20V | 30mA |
| DG409 | 100 | 5nA | 150ns | 150ns | 双向双 4 选 1 | ±20V | 30mA |
| MAX333 | 140 | 5nA | 50ns | 460ns | 单向四 2 选 1 | ±18V | 20mA |

只有正确选择多路开关的种类，注意多路开关与相关电路的合理搭配与协调，保证各电路单元合适的工作状态，才能充分发挥多路开关的性能，甚至弥补某些性能指标的欠缺。如目前多路开关元件的通/断切换方式大多为“先断后通”，这种方式显然适用于数采通道的信号切换，可以避免“先通后断”造成两个通道之间短接的现象。然而，在程控增益放大器中用多路开关来改变反馈电阻(调整放大器增益)时，“先断后通”的多路开关就可

能导致放大器出现瞬间开环，此时极高的开环增益容易破坏电路的正常工作，甚至损坏元器件。

模拟多路开关的基本配置是单端方式，但有时将两个模拟多路开关分别接到差动放大器的两个输入端，以发挥差动放大器的共模抑制能力，不过此时芯片为系统提供的连接通道数会减少一半。图 2-29 给出了这样两种不同的连接方式。




(a) 信号共地时接单端输入方式　　(b) 多信号地时采用差分方式

图 2-29　模拟通道开关的使用方式

多路开关的切换速度与其自身的结构、工作条件及外电路情况都有关系，如多路开关的平均传输延迟时间随电源电压升高而减小。

电子模拟开关导通时的等效电阻自然希望较低，但在数据采集器中，只要后置电路的输入阻抗足够大，则不会产生开关压降而影响信号的传递。传输信号的信号源内阻 $R_S$ 对多路开关的切换时间也有重要影响。对高内阻的信号源(一些传感器就是如此)，宜用阻抗变换器将阻抗变低后再接入多路开关，减小 $R_S$ 可减小多路开关关断漏电流造成的误差。

当系统需要的信号通道数较多时，宜采用图 2-30 所示的两级连接方式。若系统共需要 64 个信号通道，则可将这 64 个通道分成 8 组，各组分别接至 8 个二级开关，信号由二级开关输出。这样可使开关并联输出的总电容大大减小，电路的时间常数降低，开关速度提高。此外，这种连接方式还可使多路开关的总关断漏电流下降(设每个开关的关断漏电流相同)，从而减小关断漏电流造成的误差。对上述两种作用，通道数越多效果越显著。

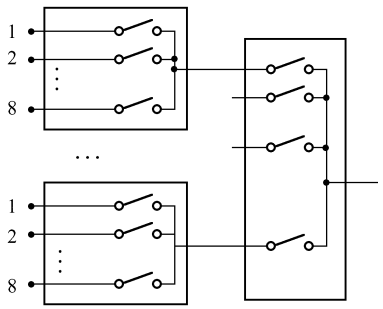



图 2-30　通道开关的两级连接方式

选择和使用多路开关时，考虑的重点是满足系统对信号传输精度和传输速率的要求，同时还必须全面了解多路开关的特性，否则可能出现难以预料的问题。如 CMOS 多路开关在电源切断时是断开的，而结型 FET 多路开关在电源切断时是接通的。若未注意到这一点，就可能因电源的通/断而损坏相关芯片。又如，受芯片种类或应用场合的限制，实际电路中往往有多余的通道。由于多路开关的内部电路相互联系，所以多余的通道可能产生干扰，必要时应进行适当处理，如将所有多余通道的输入端接地。

## 2.3　数据采集器的操作控制

数据采集器的完整功能是由其核心 A/D 转换电路、通道调理电路和操作控制系统构成的，通常总是利用某种形式的微处理器控制器对其数据采集电路进行控制操作。控制器可以采用 PC 中的 CPU 直接控制其插槽中的数采模件，也可以通过微处理器芯片、单片机、DSP、SOC 和 SOPC 等方式组成的嵌入式系统来完成数据采集任务。测试得到的数据可由控制器本身直接处理和应用，形成特定功能的仪器和设备；也可经过某种通信方式将采集到的数据传输到其他设备，构成功能更为广泛的各种应用系统。本节将以单片机组成的嵌入式系统为例，具体说明数据采集器的基本控制方式、工作原理和设计方法。

### 2.3.1　典型的嵌入式微控制器

嵌入式系统的核心控制器可以选择微处理器、单片机、DSP 和 SOC 等不同方式。在微处理器方式下 CPU 通过配置的程序存储器、数据存储器、定时/计数器、各种串/并行端口和总线接口等外部器件，形成操作 A/D 转换器、控制通道调理电路和传输测试数据的能力。单片机则尽可能地将上述功能做到一块芯片中，因此更为开发者和使用者所喜好。DSP 具有很强的数据处理能力，适用于那些要求更高的专业应用场合。SOC 则可做成模拟电路和数字电路混合的单片结构，进一步满足电子电路集成化趋势的要求。

单片机的种类很多，选择时除了需要考虑其性能指标必须满足数采控制的需要、工作必须可靠外，还应特别注意其应用的广泛性和普及性。因为单片机的控制作用主要通过编写程序固件(Firmware)来实现，因此必须建立一个好的开发环境。其中包括支持某种编程语言的编译器、支持基本功能的函数库、执行代码的联机下载机制，以及进行单步和断点调试的硬件开发设备和相应的系统软件。购置和掌握这样的设备和软件需要花费相当的成本和时间，因此希望其应用面能够更宽一些，持续的时间也能更长一些。同时，一种应用广泛的机型，资料自然丰富，交流的机会也多。实际上，尽管单片机的生产厂家和型号很多，但其内核在国内市场上流行的却只有少数几种，其中最典型的是 51 系列、ARM 和 PIC。

51 系列是指以 Intel-8051CPU 结构和指令系统为内核的单片机，包括早期的 8031(片内不设程序存储器)、8051(片内提供固化 ROM 程序存储器)、87c51(片内提供可擦写的 EPROM 程序存储器)、89c51(片内提供可电擦写的 $E^2$PROM 程序存储器)等，也包括近年来在此基础上新开发的、具有某些专用功能的一些单片机，如 Cygnal 公司开发的 F0xx、F2xx、F3xx 系列(包括 A/D、D/A、Flash、4KB 数据存储器、$I^2C$、SPI 等接口功能)。又如 Cypress 公司开发的 EZUSB-FX2 提供了片内 USB2.0 设备接口，可以直接实现底层数采设备与 PC 主机之间的快速通信。

ARM 是指 Advanced RISC Microprocessor 的缩写(高级 RISC 微处理器)，这是英国的一家靠转让知识产权运营的设计公司，本身并不生产芯片，而由合作伙伴的公司来生产各具特色的芯片。ARM 产品内核耗电极少、功能很强，特有的 16/32 位双指令集效率很高，而且在全世界范围有众多的合作伙伴，目前已成为移动通信、手持计算、多媒体数字消费等嵌入式解决方案的 RISC 标准。ARM 还提供了嵌入式 ICE-RT 逻辑和嵌入式跟踪核，用以辅助器件的调试。嵌入式跟踪宏单元(ETM，Embedded Trace Macro Cell)被设计成驻留在 ARM 处理器

内，用以监控内部总线，并在不妨碍内核速度的条件下跟踪指令和数据的访问。另外，ARM单片机本身的集成度也很高，拥有大量的程序存储器和数据存储器，往往作为高端产品嵌入式控制器的一种优先选择。

PIC是指Microchip公司开发的一类微控制器系列。这类微控制器采用精简指令集计算机结构，存储方式基于哈佛双总线概念，完全分开的数据和指令传输总线可以实现多位字宽访问，从而使其指令集内的多数指令成为单字指令。取指和执行采用指令流水线结构，具有高速、低功耗、高性价比的特点。同时，PIC还有一些十分引人注意的特别之处：一是型号特别多，可针对各种特殊要求分别提供特殊的I/O功能；二是输出引脚驱动能力很强，可以提供20~25mA的电流直接驱动小型继电器；三是一次性编程OTP(One Time Program)技术使芯片的使用成本很低。因此，人们常常把PIC当做一个“万能专用芯片”来使用，即在本来需要某种特殊功能电路时，就选择一个具有相应功能基础的PIC，然后按具体需要编程后再一次性地下载固化代码。

### 2.3.2 接口芯片的总线操作

在嵌入式系统中，微控制器对接口芯片的操作通常是通过读/写片内寄存器来实现的。通过读其状态寄存器了解工作现状，通过写其控制寄存器操作动作，通过读/写其数据寄存器交换数据。控制器访问数采接口芯片主要采用传统三总线方式、非多路复用方式、多路复用方式和串行接口等不同形式。

传统的三总线结构以端口地址直接区分寄存器，通过数据总线读/写寄存器内容，根据不同的读/写指令操作读/写控制线，完成读/写操作的访问周期。由于其访问内部寄存器的多少直接关系芯片引脚所占用的地址线数目，故使用逐渐减少。

非多路复用方式的系统结构和典型时序如图2-31所示，此刻CPU仍以三总线方式操作使用控制线$\overline{CS}$、$\overline{RD}$、$\overline{WR}$，地址线A0，以及双向数据总线D0~D7。但地址线A0=1时，芯片所接收的数据总线内容只是用来指定所要访问的芯片寄存器内部地址；而当A0=0时，才去具体读/写所指定的芯片寄存器。因此，增加芯片寄存器数目，只需增加芯片内部寻址寄存器的能力，而不需要增加外部地址线。

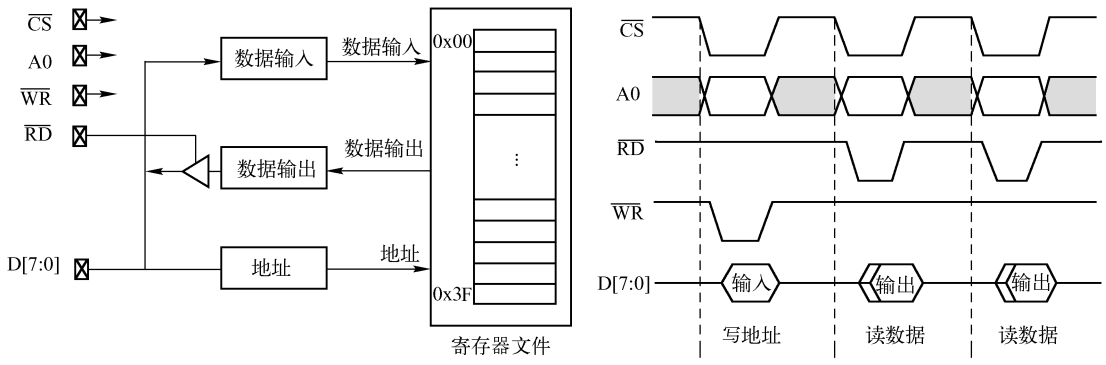



图2-31 非多路复用方式的系统结构和典型时序

多路复用方式的基本电路接法和控制时序如图2-32所示，此时地址-数据复用线AD[7:0]符合单片机8051核的标准操作时序。当地址锁存使能信号ALE有效时，地址信息

被打入地址锁存器内，然后数据随控制信号$\overline{WR}$、$\overline{RD}$的设置被写入或读出。

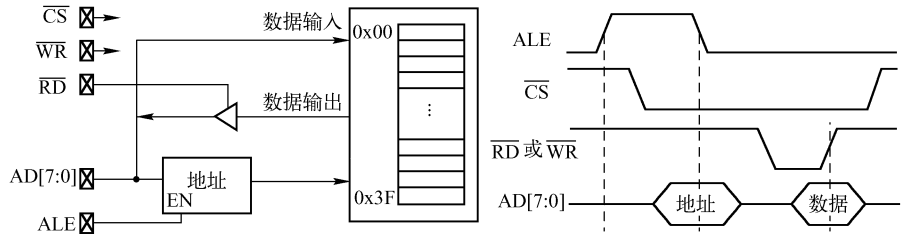



图 2-32　多路复用方式的基本电路接法和控制时序

应当指出，虽然不同单片机的指令系统不同，但实际编程目前均以 C 语言为主，只是个别场合需要特殊处理时(如中断处理)，才使用针对不同指令系统的汇编语言。因此在嵌入式数采的开发过程中，不同单片机系统对于实际编程固件的程序设计并无很大影响。需要注意的只是指令的执行时间和具体执行读/写指令的总线时序配合，如 ARM 机的速度比 8051 机快很多。因此对同一接口芯片的 I/O 操作，在 8051 机下工作得很好的 C 程序，改换 ARM 机后就有可能出现某些时序问题，从而需要增加某些延迟来调整和匹配控制脉冲的宽度。

与传统的并行传输方式不同，近年来许多用于数据采集工作的芯片(如 A/D 转换器)允许微控制器通过串行接口的方式进行访问。其中最常见的串行通信方式是 SPI 总线和 $I^2C$ 总线，下面将结合有关数据采集电路的功能予以说明。

### 2.3.3　$I^2C$ 总线的数据传输

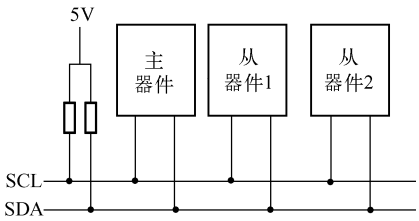



图 2-33　$I^2C$ 总线的构型

$I^2C$ 总线以半双工同步传输的方式工作。它只有两根信号线、同步时钟线 SCL 和串行数据线 SDA。$I^2C$ 总线上数据传送的最高速率分为低速(100kbps)、快速(400kbps)、高速(3.4Mbps)3 种模式。如图 2-33 所示，$I^2C$ 总线必须配置上拉电阻，故当总线空闲时器件输出呈高电平。$I^2C$ 总线上器件的输出级应以三态门或集电极开路门的方式支持“线与”功能，即一个器件的低电平输出必然压倒另一器件的高电平输出。

如图 2-34 所示，SDA 线上传送的数据只在时钟 SCL 线处于高电平期间才有效，而且此刻 SDA 线必须稳定。数据从高到低或从低到高的变化只有在时钟 SCL 线处于低电平期间才能发生。如果在 SCL 线为高电平时 SDA 线发生由高到低的跳变，则被 $I^2C$ 总线定义为数据传输过程的起始信号；而当 SCL 线为高电平时 SDA 线发生由低到高的跳变将被定义为数据传输的结束(停止)信号。

$I^2C$ 总线将器件分为主器件和从器件两类。只有主器件才能发出控制通信过程的启动命令、结束命令和寻址命令。SCL 线上的时钟同步信号主要由主器件产生，但在某些条件下也能被从器件影响以实现某种同步关系。如果从器件未能准备好接收传输字节，可将时钟 SCL 线暂时拉低，迫使主器件处于等待状态，直到从器件准备好后再释放 SCL 线。$I^2C$ 总线上进行通信的基本过程如下：

（1）主器件发出起始命令(起始信号)。

（2）主器件发出寻址命令，选择准备与之交换数据的从器件。

（3）主器件发送数据给接受寻址的从器件，或接收从器件发来的数据。

（4）主器件发出结束命令(结束信号)。

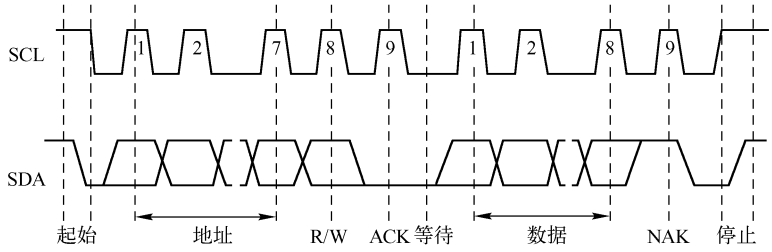



图 2-34　$I^2C$ 总线数据的传输时序

$I^2C$ 总线的数据传送以字节为单位，字节传送时先传高位，后传低位。每个字节后面跟随一个响应(认可)位，响应位也对应于 SCL 线的一个时钟。发送器件在此时钟周期内释放 SDA 线，等待接收器件用 SDA 线的低电平发来认可响应信号。主器件在得不到从器件认可响应信号时应及时产生一个结束信号，使数据传送过程异常结束。$I^2C$ 数据传送的基本格式如图 2-35 所示。主器件在起始信号产生后开始发送寻址命令，包括从器件的 7 位地址码 SLA 和它的读/写标志位 W/R(第 8 位)。每次正常的数据传送过程总是由主器件发出起始信号 S 开始，而由主器件发出结束信号 P 而终止。

主器件发送器工作方式

| S | SLA | W | A | 数据字节 | A | 数据字节 | A | P |
|---|---|---|---|---|---|---|---|---|

主器件接收器工作方式

| S | SLA | R | A | 数据字节 | A | 数据字节 | N | P |
|---|---|---|---|---|---|---|---|---|

图 2-35　$I^2C$ 总线上的一些典型数据传送格式

在主器件接收方式的工作过程中，从器件接收主器件的寻址，并发回认可响应位信号 A 后，主器件发送器就变成主器件接收器，从器件接收器则变成从器件发送器。此刻寻址字节的认可位仍由从器件产生，而以后主器件接收数据过程中的认可位(除最后一个不发外)则由主器件接收器产生。若主器件准备停止接收，则对传送的最后一个数据字节不发确认信号，以使在主器件本身产生结束信号的同时保持释放 SDA 线的状态。

通常在传送过程中方向改变时，需要重发起始信号并重新寻址。但在组合方式中，主器件可以直接产生下一个起始信号并重新寻址从器件，而不需要中间再发出一个停止信号。

$I^2C$ 的器件地址由固有部分和可选部分组成。前者由 $I^2C$ 器件的本身功能类型决定，作为基地址部分，具体规定由 $I^2C$ 总线的有关委员会决定；后者则作为子地址部分，可通过 $I^2C$ 器件引脚电平的设置来确定。

当微处理器作为从器件连到 $I^2C$ 总线时，如果片内提供硬件 $I^2C$ 总线接口，微处理器可用响应接口中断的方式工作。如果没有 $I^2C$ 硬件器件担任接口功能，则必须通过软件不断地

查询监视总线状态。为使微处理器能以相对低的速率查询总线，可在主器件发出起始信号之后、数据传送之前加发启动字节(0000 0001)。这样，总线上的微处理器从器件平时可以用较低的速率采样 SDA 线，直至检测到启动字节中 7 个“0”中的 1 个“0”后，再切换到较高速率采样来捕捉总线上发出的第二个起始信号 Sr。第二个起始信号仅用于微机从器件的同步。而对普通 $I^2C$ 硬件芯片构成的从器件，收到第二个起始信号时则会自动略去这个启动字节的作用。主器件在发出启动字节后也要在 SCL 线上产生相当于认可信号的时序，但这仅仅是操作格式的需要，并不要求从器件在 SDA 线上做出响应。

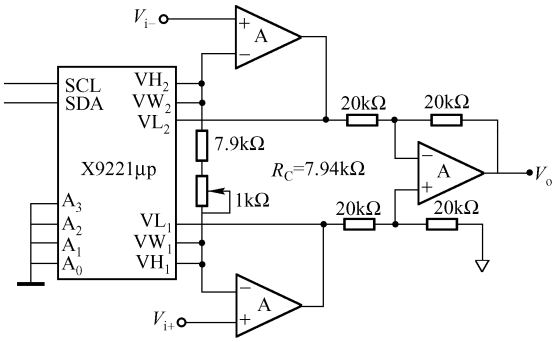



图 2-36　$I^2C$ 控制的电子电位器

图 2-36 是数采中使用 $I^2C$ 总线器件的一个典型示例，芯片 X9221 是美国 Xicor 公司开发的一个电子电位器，其中引脚 $VH_x$、$VL_x$、$VW_x$ 分别是电位器的高端、低端和滑动端。图中接法通过调整反馈电阻来控制放大倍数，程控命令则通过芯片内置的一个 $I^2C$ 接口输入。这个芯片的电位器阻值是 50kΩ，数字调整分为 0~63 个等级，步进值为 794Ω。按图中的电路设计，放大器程控增益关系为 $A = 1 + 0.2N$，这里 $N$ 表示滑动端位置。对应 $N$=0~63，电路的程控增益范围为 $A = 1 \sim 12.7$。

X9221 作为从器件，其高 4 位地址为 0101(由器件类型决定)，低 4 位地址由 A3~A0 输入端状态确定。在送到 X9221 的下一个字节中，高 4 位为指令，低 4 位用来指出待操作的电位器及其辅助寄存器。每个电位器有 1 个易失性存储器 WCR，上电时自动装入 $R_0$ 的值，用来选择电阻阵列的 64 选 1 开关。每个电位器有 4 个数据寄存器，可被主机直接读出或写入。数据可以在数据寄存器和 WCR 之间传输，改变这些寄存器需要 10ms 时间。X9221 的电阻阵列之间可以串联。在三字节指令中，其数据字节的低 6 位用来定义滑动端位置，而高位 CM=0 时正常工作，CM=1 时与相邻电位器串联。同时，高位 DW=0 时使能滑动端，DW=1 时禁止滑动端(电平浮空)。

### 2.3.4　SPI 接口的操作

SPI(Serial Peripheral Interface)串行外设接口由 Motorola 公司提出，用于实现主器件和从器件之间的全双工通信。SPI 支持在同一个总线上将多个从器件连接到一个主器件，并允许在主器件和所选从器件之间进行数据传输。同一总线上也可以有多个主器件，系统提供了冲突检测功能。SPI 接口通常使用主出从入(MOSI)、主入从出(MISO)、串行时钟(SCK)、从选择($NSS^*$)等 4 根信号线。只有主器件才能控制 SCK，也只有主器件才能启动数据传输。通常主器件是一个微控制器，主方式下对 SPI 接口的数据寄存器写入一个字节就启动一次数据传输。于是在 MOSI 线上立即串行移出这个字节数据(先高位后低位)，并为 SCK 建立相应的时钟信号。时钟线仅在数据传输时才被激活，传输结束后又回到三态高阻电平。选择线 $NSS^*$ 用于从器件的使能控制，低电平有效，可以直接接地以保持长期有效。每次数据传输均由主器件在 SCK 线上发出的 8 个脉冲作为从器件的同步时钟。SPI 的从器件接口通常也是一

个独立的移位寄存器，它以 SCK 时钟为时基，顺序接收 8 位数据。

SPI 主器件和从器件之间连接的基本方法如图 2-37(a)所示，从器件常用其 NSS* 作为其片选使能信号。从器件的这种片选使能信号除了直接接收主器件专用的 NSS* 信号外，也可以利用主器件上一些普通的 I/O 端口线来完成同样的工作。

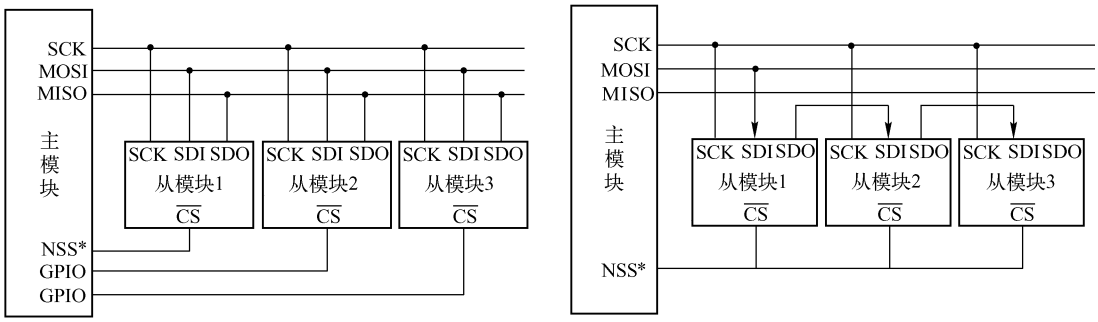



图 2-37　SPI 的两种总线连接方式

SPI 主器件和从器件之间连接的另一种方法如图 2-37(b)所示，此时各从器件接口都是简单的 8 位移位寄存器，每个从器件具有相同的时钟和片选信号，3 个从器件组成一个 24 位的从器件整体，高位在前，低位在后。主机需要连续发送一个 24 位数据，从高到低完整地传送到 3 个从器件中。

另外，在全双工操作方式下，SPI 主器件通过 MOSI 线向从器件发送数据时，也可以通过 MISO 线同时接收来自被寻址从器件发出的数据内容，并自动更换主器件数据寄存器中的内容。这样，完成一个字节数据传输后产生的中断标志位既可作为发送完成的标志，也可作为接收数据准备好的标志。SPI 不需要为传输数据特别指定数据流动方向，因为主机用两根分立的引脚同时接收和发送数据。图 2-38 描述了某种 SPI 主器件和 SPI 从器件之间的全双工操作方式。当 SPI 传送发生时，一个 8 位字节通过 MOSI 引脚移位输出，同时另一个 8 位字节从 MISO 引脚输入。这个过程相当于主器件的 8 位移位寄存器和从器件电路联合组成了一个 16 位循环移位寄存器。每次传送发生时，这个 16 位寄存器就循环移动 8 位，从而完成主器件和外设从器件之间的数据交换。

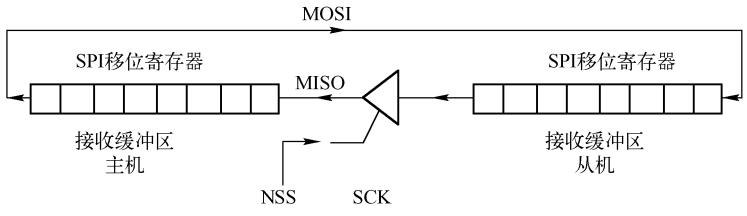



图 2-38　SPI 主器件和从器件之间的全双工操作

在数据采集器中，带有 SPI 接口 A/D、D/A 转换器、存储器等器件一般作为从器件出现，而单片机控制器则通常作为主器件。实际上 $I^2C$ 和 SPI 规范均支持多个主器件的情况，且需讨论它们之间的仲裁问题。考虑这种情况在数据采集器中并不常见，故这里不再赘述。

### 2.3.5　LVDS 总线差分接口

LVDS(Low Voltage Differential Signal)即低电压差分信号，LVDS 接口又称 RS-644 总线接口，是 20 世纪 90 年代才出现的一种数据传输和接口技术。LVDS 的典型工作原理如图 2-39 所示，其驱动器由驱动差分对的电流源组成，电流通常为 3.5mA。LVDS 接收器具有很高的输入阻抗，因此驱动器输出的大部分电流都流过 100Ω 的匹配电阻，并在接收器的输入端产生大约 350mV 的电压。当驱动器翻转时，它改变流经电阻的电流方向，由此产生有效的逻辑“1”和逻辑“0”状态。

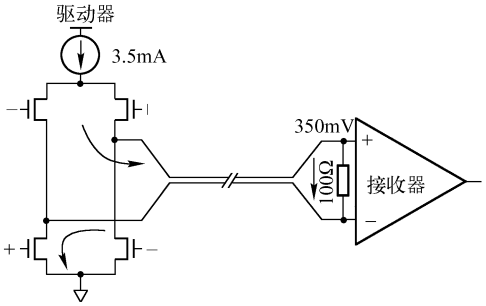



图 2-39　LVDS 的典型工作原理

LVDS 技术定义在两个标准中：ANSI/TIA/EIA644（1995 年 11 月通过）和 IEEE P1596.3（1996 年 3 月通过）。这两个标准都着重定义了 LVDS 的电特性，其中包括：

（1）低摆幅（约为 350mV）的低电流驱动模式意味着可实现高速传输，ANSI/TIA/EIA644 建议 655Mbps 的最大速率和 1.923Gbps 的理论极限速率。

（2）恒流源电流驱动把输出电流限制到 3.5mA 左右，使跳变期间的尖峰干扰很小，因而产生的功耗也很小。这将允许集成电路密度进一步提高，从而提高了 PCB 的效能，降低了成本。

（3）具有相对较慢的边缘速率（$dV/dt$ 约为 1V/ns），同时采用差分传输形式，使其信号噪声和 EMI 都大为减小，因而具有较强的抗干扰能力。

LVDS 具有高速、超低功耗、低噪声和低成本的优良特性，其应用模式分单向点对点、双向点对点、多分支形式、多点结构等多种形式。2001 年推出多点低压差分信号（MLVDS）的国际标准 ANSI/TIA/EIA8992001，目前已有一些 MLVDS 器件面市。LVDS 技术的应用领域也日益广泛，普遍用于高速系统内部的背板互连和电缆传输。图 2-40 为一种高速 A/D 转换器 AD9252 的引脚方式，AD9252 可同时进行 8 个通道、14 位字长、50MHz 采样速率的高速数采。各通道可串行输出转换结果，通过 LVDS 差分驱动器送入 FPGA 中进行存储和处理。

### 2.3.6　数据采集器定时采样的控制方式

数据采集器典型的控制任务是定时采样，定时采样有两种主要方式：一种方式是通过单片机系统的定时/计数器产生定时中断，并将数据采集器的工作分为前台和后台两个过程同时运行。另一种方式是依托嵌入式实时操作系统，将数采过程、数据存储、数据通信等功能分别形成不同的实时任务，在操作系统的统一调配下同时运行。

如图 2-41 所示，数据采集器采用定时中断的工作方式时，前台为一个顺序执行、循环操作的主程序。由于系统初始化过程已为定时计数器设置了定时时间，于是每隔这样一段时间就会产生一个中断，并强迫完成中断服务子程序。通常这个定时中断事件的优先级最高，甚至系统只安排这样一个中断，所以这个定时采样中断的服务子程序只会打断其他程序的运

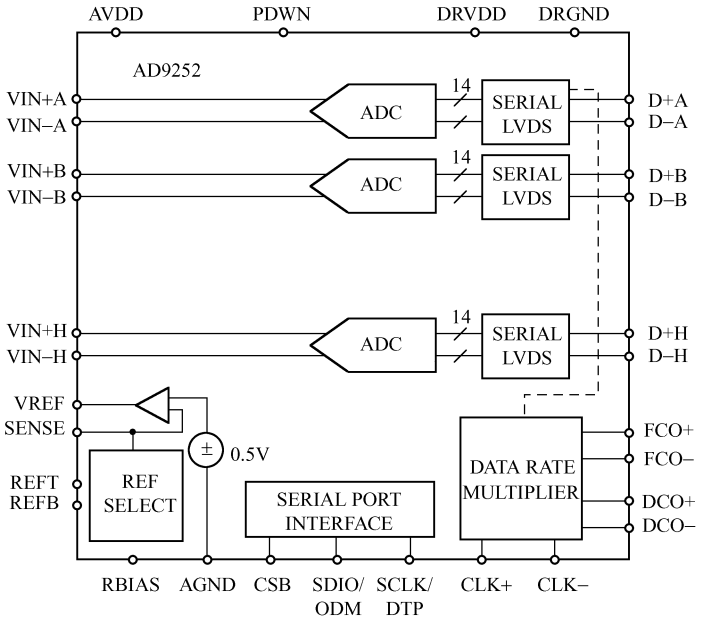



图 2-40　高速 A/D 转换器 AD9252 的引脚方式

行，而不会被其他程序所打断。这里，中断服务子程序的任务就是数据采集器中必须及时完成的 A/D 转换操作，即实现启动转换、监测状态、读出数据后将其依次暂存到缓存数组中的完整过程。可以认为这个后台工作过程根本不受前台主程序的影响，只是依据定时中断完成 A/D 转换，并将结果写入数据数组。另外，对于前台主程序处理数据采集的结果，当数据数组达到一定数目(nData)后，即可进行处理(如计算平均值、进行频谱分析等)，也可将计算的结果通过某种通信方式上传到其他设备，或者直接进行显示。当然，数据采集的有关参数(如采样时间)也可以通过主程序设置，其方式可以通过本地键盘，也可以利用通信接口从系统主机下载得到。

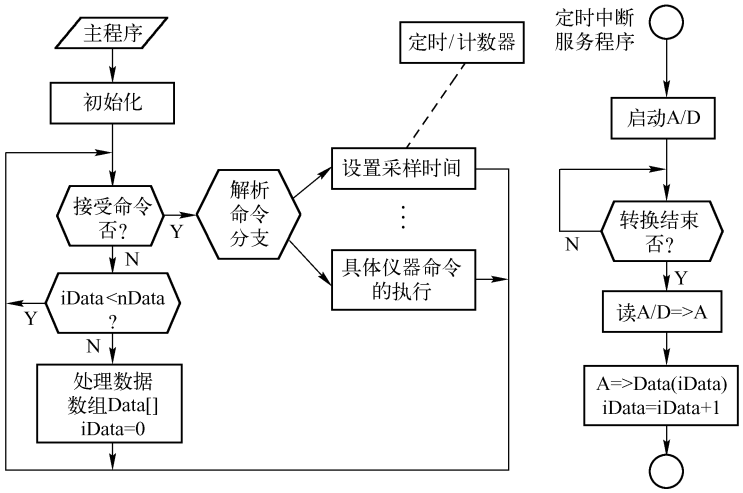



图 2-41　数据采集器的定时中断控制方式

应当指出，主程序中的各种工作都可以被中断子程序中的数据采集工作打断，这是满足数据采集器定时采样基本需要的关键措施，其结果好像数采已在后台自动进行，前台只需要注意命令的执行。但是如果还有更多的中断源的话，系统控制就会变得十分复杂，处理起来就会产生一些困难，此种情况下使用操作系统可能更方便一些。

### 2.3.7　数据采集器中的实时操作系统

随着数据采集器总体性能的提高，人们不再满足简单的数据显示，数据存储、数据通信、数据处理和数据应用的不同方式使数据采集系统更新换代变得很快，因此建立嵌入式实时操作系统就引起了人们的注意。计算机操作系统作为应用系统与机器本身的“桥梁”，可以避免直接操作机器带来的烦琐工作，转而通过调用操作系统相对简明的接口机制来处理诸如任务调度、内存管理、外设驱动等基础工作，从而不仅可以确保这些底层工作更加可靠，而且可以将主要精力和程序设计集中到上层应用任务本身。

实时操作系统中的实时概念是针对批量处理的工作方式，前者能够在一个任务的执行过程中及时地响应某种实时要求，即可以暂时中断本身任务后去执行另一个任务，而后还可回到原先任务中断处继续进行原先的工作。后者则只能在一个任务完成后，再去顺序地执行另一个任务。使用实时操作系统后，应用程序的各个部分由相对独立的任务组成，处理它们之间本来需要精心策划的复杂关系变为操作系统下选择相对简单的调度策略和建立依靠标志的通信方式。

μC/OS 作为一个实时操作系统，其主要功能是实现任务调度和任务间通信。在 μC/OS 中，通常将一个复杂的应用程序分解为若干个相对独立的任务，而任务的主体实际上是一段运行在内存中完全占有 CPU 资源的程序代码。这段代码以函数形式出现(但没有返回参数)，通常循环执行特定的功能，并拥有自己的任务控制块(OSTCB)和堆栈空间。μC/OS 可以管理最多 64 个任务，每个任务都必须唯一地配置不同的任务号作为标识，且任务号从小到大按优先级排列，这意味着每个任务的优先级均不相同。

如图 2-42 所示，每个任务均有 5 种主要的工作状态，即休眠态、就绪态、运行态、等待态和延时态，且每个时刻最多只有一个任务处于运行状态，即当某个任务运行时，其他任务就只能处于非运行的其他状态。

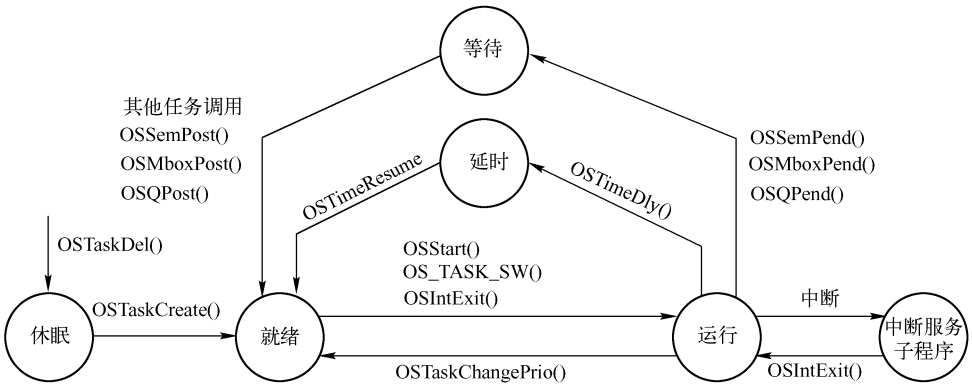



图 2-42　μC/OS 系统每个任务调度的工作状态图

由于 μC/OS 的任务调度是按照抢先式多任务系统设计的，因此总是让所有处于就绪状态的任务中优先级最高的那个任务前去挑战当前正在运行的那个任务，再在它们之间选择优先级高者作为调度后的运行者(进入或保持运行态)。任务的建立可以通过函数 OS_Create 实现，且新任务建立后会立刻进入就绪态。如果中断源打开，正在运行的任务也可响应中断进入中断服务程序，且中断返回后则会进行一次任务调度，重新选择运行任务。

上述工作过程说明具有最高优先级且正运行的任务只有自行退出运行态，才能让其他任务有机会得到 CPU 的控制权。任务主动退出运行状态的一种方法是它本身调用延时函数 OSTimeDly 使该任务进入延时态。μC/OS 有一个基本时钟节拍，相当于系统的“脉搏”，延时则根据“脉搏”的计数进行。在每次“脉搏”发生时都要进行检查，使已经达到延时要求的任务立刻回到它的就绪状态，并立刻进行一次新的任务调度。如果这个已经回到就绪态的任务此刻在所有处于就绪态的诸任务中具有最高优先级，并且这个优先级也高于现行运行任务的优先级，则这个任务就会重新进入运行态，恢复其对 CPU 的控制权，并在原先程序代码调用延迟函数的断点处继续运行。

任务主动退出运行状态的另一种方法是通过调用等待事件发生的函数(如 OSSemPend、OSMboxPend、OSQPend)进入等待态。这个等待的事件显然需要其他任务来触发产生。也就是说，只有其他任务在运行过程中通过调用能够影响状态事件的某些函数(如 OSSemPost、OSMboxPost、OSQPost)并且满足相应等待状态要求后，已经处于等待状态的那个任务才能回到就绪态，才有可能在新的调度中重新进入占据 CPU 的运行态。这实际上是在进行任务间的通信，μC/OS 中任务间通信方式包括信号量、邮箱和消息队列等不同形式，它们的实现都基于事件标志，只是使用和定义的方法不一样。图 2-43 所示是两个任务通过事件进行通信的示意图。

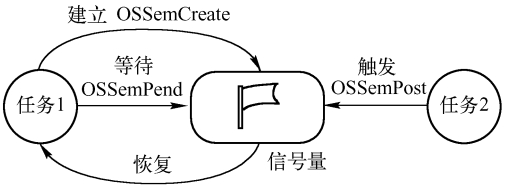



图 2-43 两个任务通过事件进行通信的示意图

在事先通过调用 μC/OS 函数(OSSemCreate)建立信号量标志事件后，优先级较高且正在运行的任务 1 本身可以通过调用 μC/OS 函数 OSSemPend 将自己挂起(从运行态退到等待态)，直至此刻正在运行的任务 2 通过其本身调用 μC/OS 函数 OSSemPost 满足信号量要求，任务 1 将返回就绪态，并在系统调度处理后回到运行态。由于任务 1 和任务 2 相互之间并没有直接进行任何程序调用，它们之间的协调动作都是调用操作系统 μC/OS 的函数，通过信号量标志的建立、改变和监测间接实现的。这样，在处理多个任务的复杂关系时就能避免程序的相互调用和中断的复杂嵌套，而将其转变为信号量标志在各任务中分别处理的过程。

在数据采集系统中，往往需要同时建立定时采样、数据处理、上传显示等不同任务。显然，定时采样任务在时序要求上最为严格，因此应当安排为最高优先级。同时，仅当其完成采样、A/D 转换、将测得数据存入缓存区并通过调用延时函数 OSTimeDly 自行进入等待态后，其他任务方能考虑进入运行。而一旦延迟的“脉搏”节拍计数达到要求，即经历指定的采样周期后，则在操作系统的机制下又必须立即自动恢复运行状态，从而确保定时采样的时间间隔要求。数据处理任务的优先级则应低于上传显示任务的优先级，这是因为前者并不总是需要实时完成，而后者则随时需要应对用户的要求。不过后者的执行也是有条件的，即只有

已经完成了数据处理任务进行上传显示任务时才能得到当前需要的数据，否则只能得到以前处理过的数据。因此，可由优先级较高的上传显示任务在其运行期间，通过调用等待事件发生函数 OSSemPend，建立并清空数据处理完成的信号量标志后进入等待状态。于是优先级较低的数据处理任务便可趁机进入运行态工作，直至数据处理过程完成后通过调用函数 OSSemPost 设置数据处理完成的信号量标志，从而使上传显示任务退出等待状态，恢复运行状态，执行上传显示任务。注意，数据处理任务与上传显示任务的运行均在采样任务处于延迟的采样间隔期间插空完成。

上述分析表明，使用操作系统的主要目的在于简化编程的复杂性，便于系统的扩充和更新，从而提高软件的开发效率，但不会提高系统的运行效率。

## 2.4　数据采集过程中的采样问题讨论

数据采集过程中被测物理量原本都是连续的时间信号，经过等间隔时间的采样，得到一组离散的采样序列。那么数据采集器得到的这组离散采样序列是否还能保持原先连续时间信号所含的全部信息呢？这与采样间隔究竟有何关系？对此又应采取什么对策？这些将是本节要讨论的主要问题。

### 2.4.1　采样定理和混叠现象

香农(Shannon)采样定理认为：只要被采原始连续信号本身的带宽有限，而采样频率又大于或等于信号所含频率分量中最高频率的两倍，理论上就可以根据采样信号恢复出被采样的原始连续信号。

如图 2-44 所示，设已知一连续信号 $f(t)$ 具有有限带宽频谱 $F(j\omega)$，现以采样周期 $T$ 对其采样，得到相应的采样函数 $f^*(t)$。考虑载波信号

$$\delta_T(t) = \sum_{k=-\infty}^{\infty} \delta(t - kT) \tag{2-27}$$

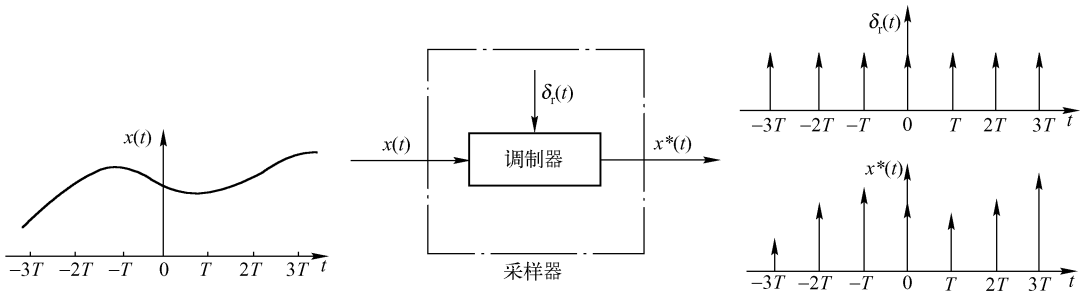



图 2-44　采样过程

是一个周期函数(周期为 $T$)，因此必然可以将其展成基频为 $\omega_s = 2\pi/T$ 的傅里叶级数。写成指数形式，即有

$$\delta_T(t) = \sum_{n=0}^{\infty} \dot{C}_n \cdot e^{jn\omega_s t} \tag{2-28}$$

这里系数

$$\begin{aligned}\dot{C}_n &= \frac{1}{T}\cdot\int_{-T/2}^{T/2} f(t)\cdot \mathrm{e}^{-\mathrm{j}n\omega_s}\mathrm{d}t \\ &= \frac{1}{T}\int_{-T/2}^{T/2}\sum_{k=-\infty}^{k=\infty}\delta(t-kT)\cdot \mathrm{e}^{-\mathrm{j}n\omega_s t}\mathrm{d}t = \frac{1}{T}\cdot \mathrm{e}^{-\mathrm{j}n\omega_s t}\Big|_{t=0} = \frac{1}{T}\end{aligned} \tag{2-29}$$

于是，采样信号将表达成下列形式：

$$f^*(t) = \delta_{\mathrm{T}}(t)\cdot f(t) = \frac{1}{T}\sum_{n=-\infty}^{\infty} f(t)\mathrm{e}^{\mathrm{j}n\omega_s t} \tag{2-30}$$

由原始连续信号对应的频谱关系：

$$f(t)\Leftrightarrow F(\mathrm{j}\omega)$$

根据傅里叶变换的性质：

$$f(t)\cdot \mathrm{e}^{\mathrm{j}n\omega_s t}\Leftrightarrow F(\mathrm{j}\omega-\mathrm{j}\omega_s)$$

于是可以发现采样后新的离散信号的频谱特性关系为

$$f^*(t) = \frac{1}{T}\sum_{n=-\infty}^{\infty} f(t)\mathrm{e}^{\mathrm{j}n\omega_s t}\Leftrightarrow \frac{1}{T}\sum_{n=-\infty}^{\infty} F(\mathrm{j}\omega - \mathrm{j}n\omega_s) = F^*(\mathrm{j}\omega)$$

即有

$$F^*(\mathrm{j}\omega) = \frac{1}{T}F(\mathrm{j}\omega)+\frac{1}{T}F(\mathrm{j}\omega-\mathrm{j}\omega_s)+\frac{1}{T}F(\mathrm{j}\omega+\mathrm{j}\omega_s)+\frac{1}{T}F(\mathrm{j}\omega-\mathrm{j}2\omega_s)+\cdots \tag{2-31}$$

可见，采样后得到的采样函数$f^*(t)$的频谱不仅保留了原先原始连续信号$f(t)$的全部频谱分量$F(\mathrm{j}\omega)$，还包括一系列附加的频谱分量$F(\mathrm{j}\omega-n\omega_s)$，$n=\pm1$，$\pm2$，…，且这些附加的频谱分量具有与$F(\mathrm{j}\omega)$相似的分布形状。

如图2-45(a)所示，信号$f(t)$的带宽有限，且其频谱分量的最高频率为$\omega_m$。当采样角频率$\omega_s\geqslant 2\omega_m$时，则如图2-45(b)所示，可采用一个理想的低通滤波器滤掉$F^*(\mathrm{j}\omega)$中$\omega>\omega_m$的全部频谱分量，从而将$F^*(\mathrm{j}\omega)$恢复成$F(\mathrm{j}\omega)/T$，表明$f^*(t)$能够恢复成原先的$f(t)$。但若

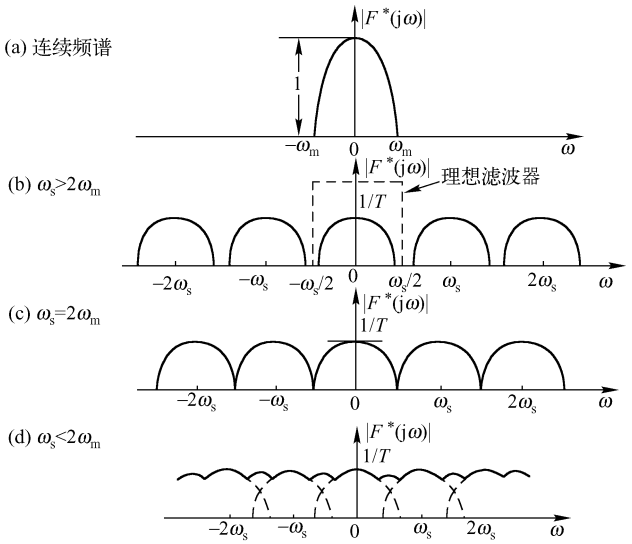



图2-45　采样定理的图示分析

取 $\omega_s<2\omega_m$，则如图 2-45(d)所示，在 $F^*(j\omega)$ 中小于 $\omega_m$ 的频段内不仅包括 $F(j\omega)$ 的频谱分量作用，而且还有从 $F(j\omega+jn\omega_s)$ 混入的频谱分量影响。这样，既使能够滤掉 $\omega>\omega_m$ 频段的频谱分量，剩下 $\omega<\omega_m$ 的频谱分量部分也不再是原来 $F(j\omega)$ 的单独作用，而是把原来高频部分的频谱分量混叠进来。当这种在信号处理中称为“混叠”的现象发生时，$F^*(j\omega)$ 就不再能够无失真地恢复成 $F(j\omega)$ 了，从而意味着 $f^*(t)$ 不能无失真地恢复成原先的 $f(t)$ 了。

因此采样定理可重述为：对一个有限频谱带宽为 $\omega_m$ 的连续信号 $f(t)$ 采样，当采样频率满足 $\omega_s\geqslant 2\omega_m$ 的条件时，采样函数 $f^*(t)$ 能够无失真地保留原先连续信号 $f(t)$ 所含的全部信息。

应当指出，采样定理中 $\omega_s\geqslant 2\omega_m$ 只是避免“混叠”的必要条件，实际滤波器低通与高阻之间的过渡特性不可能做得很陡，因此为了避免“混叠”，需要采用更高的采样频率。

### 2.4.2　抗混叠滤波器及其有关参数的选择

工程上大多数实际信号的频谱总是具有低通特征，即信号能量主要分布在一定的带宽范围内，否则就不可能实际计算 $\omega_s\geqslant 2\omega_m$。但有时被采信号的频率特性在带宽之外并不立刻衰减，而是带有一条长长的尾巴。这时为了合理地选择采样频率(避免过高)，一般要在被测信号采样前使其先通过一个称为“抗混叠滤波器”的低通滤波器，将 $\omega>\omega_m$ 的高频尾巴预先切掉。即确保信号的有限带宽条件后，再以 $\omega_m$ 为基础选择采样频率。

抗混叠滤波器将实际被测信号的高频尾巴切掉后，自然会带来信号波形畸变的负面影响。但损失的只是不很重要的高频成分，保护的则是信号的低频主要成分，使其免受混叠信号的污染。因此，抗混叠滤波器的积极作用是十分明显的。

抗混叠滤波器本质上是一个低通滤波器，这里以一个巴特沃斯滤波器为例，说明其有关参数的选择方法。这种滤波器的频率特性可以描述为

$$|H(j\omega)|=\left|\left[1+\left(\frac{\omega}{\omega_0}\right)^{2n}\right]^{-1/2}\right| \tag{2-32}$$

式中，$n$ 为滤波器的阶次，$\omega_0$ 为-3dB 频点。即当 $\omega=\omega_0$ 时，$|H(j\omega)|=0.707$，其对数分贝值为 $20\lg|H(j\omega_0)|=-3\text{dB}$；且当 $\omega\ll\omega_0$ 时，$|H(j\omega)|\approx 1$；而当 $\omega\gg\omega_0$ 时，$|H(j\omega)|=(\omega/\omega_0)^{-n}$。上述特征可参见图 2-20(a)。

需要选择的有关参数包括抗混叠滤波器的阶数和转折频率，以及采样系统的采样频率，理论上的一种具体方法如下：

(1) 根据数采性能要求和实际条件初选 A/D 转换器的字长和滤波器的阶数 $n$。

(2) 选择抗混叠低通滤波器的-3dB 频点 $\omega_0$，使信号保留频带 $\omega_m$ 内的所有频率分量的衰减不大于 A/D 转换器的半个 LSB。

(3) 选择采样频率 $\omega_s$，使任何混叠到信号保留频带 $\omega_m$ 内的频率分量均小于 A/D 转换器的半个 LSB。

为了说明上述设计过程，可以假定一个被测信号的所有频率分量都具有 $V_{span}/2$ 的幅度，这里 $V_{span}$ 是数据采集器的输入信号动态范围。由于 LSB 取决于数采 A/D 转换器的字长 $N$，即 $1\text{LSB}=V_{span}/2^N$，因此为了满足步骤(2)的要求，则有

$$\frac{V_{span}}{2}\cdot(1-|H(j\omega_m)|)=\frac{1}{2}\text{LSB}=\frac{1}{2}\cdot\frac{V_{span}}{2^N} \tag{2-33}$$

消去式(2-33)两端的 $V_{span}$，代入式(2-32)，可得

$$1-\left|\left[1+\left(\frac{\omega_m}{\omega_0}\right)^{2n}\right]^{-1/2}\right|=\frac{1}{2^N}$$

进而整理成

$$\left(\frac{\omega_m}{\omega_0}\right)^{2n}=\left(\frac{2^N}{2^N-1}\right)^2-1=\frac{2^{N+1}-1}{2^{2N}-2^{N+1}+1}$$

从而得到在 A/D 转换器字长为 $N$、滤波器阶数为 $n$ 的条件下，滤波器转折频率 $\omega_0$ 与要求信号保留频带 $\omega_m$ 之间的关系为

$$\omega_0=\omega_m\cdot\left(\frac{2^{N+1}-1}{2^{2N}-2^{N+1}+1}\right)^{-1/2n}\approx\omega_m\cdot 2^{(N-1)/2n} \tag{2-34}$$

可见，低通滤波器的阶次 $n$ 越低、A/D 转换器的字长 $N$ 越大时，滤波器的转折频率 $\omega_0$ 就要越大，方能保持原先信号不被衰减。

另外，为了进一步根据步骤(3)确定数据采集所应采用的采样频率，可以借助图 2-46。图中，$\omega_s$ 是采样角频率，曲线 A 是滤波器本身的频率特性，曲线 B 是采样引起的附加频谱。要求曲线 B 在 $\omega_m$ 处产生的混叠频率分量的幅值小于半个 LSB，等价于选择 $\omega_s$，使曲线 A 在 $\omega_s-\omega_m$ 处的频率分量幅值小于半个 LSB。即要求

$$|H(j\omega)|=\left|\left[1+\left(\frac{\omega_s-\omega_m}{\omega_0}\right)^{2n}\right]^{-1/2}\right|=\frac{1}{2}LSB=\frac{1}{2^{N+1}} \tag{2-35}$$

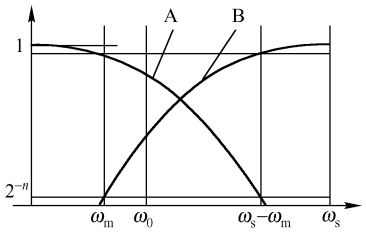


图 2-46　抗混叠滤波器的参数关系

考虑到式(2-35)中

$$\left(\frac{\omega_s-\omega_m}{\omega_0}\right)^{2n}\gg 1 \tag{2-36}$$

的通常情况，可得采样频率 $\omega_s$ 与滤波器转折频率 $\omega_0$ 的近似关系为

$$\omega_s=\omega_m+2^{(N+1)/n}\cdot\omega_0 \tag{2-37}$$

**例 2-6**　某数据采集器拟用于分析 5kHz 带宽的信号。现采用 8 位字长的 A/D 转换器测量信号，并用 4 阶巴特沃斯低通滤波器来避免混叠。试求滤波器的转折频率和系统的采样频率。

**解**：由前述式(2-34)，代入 $N=8$、$n=4$ 和 $f_m=5000$ 后，可得滤波器转折频率：

$$\omega_0=2\pi\cdot 5000\cdot 2^{(8-1)/(2\cdot 4)}=2\pi\cdot 9150\text{s}^{-1}$$

再由式(2-37)，求得系统采样频率为

$$\omega_s=2\pi\cdot 5000+2^{(8+1)/4}\cdot 2\pi\cdot 9150$$

$$f_s=\frac{\omega_s}{2\pi}=5000+4.75\cdot 9150=48.5\text{kHz}$$

对比采样定理，可见由此算出的采样频率会很高。由于上述计算是在假设被测信号在其带宽内的各频率分量都为 $V_{span}/2$ 的情况下，而工程上实际波形信号的高端频谱幅值分量

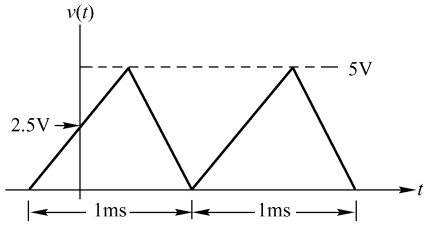



图 2-47　被测三角波信号

往往只有其低端频谱幅值分量的几分之一到几十分之一，因此上述方法无疑是十分保守的。下面将通过一个例子，说明确定抗混叠滤波器转折频率和采样频率的一种更具体的方法。

**例 2-7**　已知被测三角波信号如图 2-47 所示，数据采集器采用 8 位字长的 A/D 转换器，选择 4 阶巴特沃斯低通滤波器来避免混叠。现在讨论如何确定滤波器的转折频率和系统的采样频率。

**解**：(1)利用傅里叶级数展开三角波输入信号：

$$v(t)=2.5+\frac{20}{\pi^2}\sum_{n=1}^{\infty}\frac{(-1)^{n+1}}{n^2}\cdot\sin(2n-1)\omega t$$

$$=2.5+2.0264\cdot\sin\omega t-0.2252\cdot\sin3\omega t+0.0811\cdot\sin5\omega t-0.0414\cdot\sin7\omega t$$
$$+0.0250\cdot\sin9\omega t-0.0167\cdot\sin11\omega t+\cdots$$

(2) 确定信号保留频带。

根据输入信号的傅里叶级数展开表达式系数，可以根据需要决定保留频带。本例将信号保留频带设为 5kHz，即保留级数中前 4 项，舍去 7 倍频以上谐波分量。注意 7 倍频谐波分量的系数为 0.0414，与基波分量系数 2.0264 相比有 50 倍之差。

对于保留的 0kHz、1kHz、3kHz 和 5kHz 的频率分量，应确保其在低通滤波器作用下能够保持不大于半个 LSB 的衰减程度。

(3) 确定 4 阶巴特沃斯低通滤波器的转折频率。

为使滤波器的幅频特性在频率等于 1kHz、3kHz、5kHz 处的衰减均不超标，试取滤波器转折频率 $f_0=6$kHz。于是，根据 4 阶巴特沃斯低通滤波器的幅频特性

$$|H(j\omega)|=\frac{1}{\sqrt{1+(\omega/\omega_0)^8}}=\frac{1}{\sqrt{1+(f/f_0)^8}}$$

可以计算出各个频率分量下的信号输出幅值(如表 2-6 所示)。

**表 2-6　4 阶巴特沃斯低通滤波器的频域衰减情况**

| 频率 $f$ | $\lvert H(jf)\rvert$ | 输入信号分量幅值 | 滤波器输出幅值 | 滤波器衰减 |
|---|---|---|---|---|
| 0 | 1.0000 | 2.5000V | 2.5000V | 0.0000V |
| 1kHz | 0.9999 | 2.0264V | 2.0263V | 0.0001V |
| 3kHz | 0.9980 | 0.2252V | 0.2247V | 0.0005V |
| 5kHz | 0.9007 | 0.0811V | 0.0730V | 0.0081V |
| 7kHz | 0.4749 | 0.0414V | 0.0196V | 0.0218V |
| 9kHz | 0.1938 | 0.0250V | 0.0048V | 0.0202V |
| 11kHz | 0.0882 | 0.0167V | 0.0014V | 0.0153V |

考虑，$1\text{LSB}=5\text{V}/2^8=0.0196\text{V}$，$(1/2)\text{LSB}=0.0098\text{V}$，可以确认 5kHz 带宽要求下滤波器衰减均在 1/2 LSB(0.0098V)以下。

(4) 确定采样频率。

采样频率的确定是基于确保混叠到信号保留频带(或诸如滤波器带宽等其他人为定义的

频带)的频率分量幅值小于 1/2 LSB(这里为 0.0098V)的要求。为此观察表 2-6，可见 9kHz 是满足这一要求的最低频率分量。因此，必须同时确保不满足这一要求的 7kHz 频率分量不要混叠到信号保留频带的范围。于是，采样频率应当至少大于 12kHz(即 7kHz+5kHz)。这里，实际采样频率 $f_s = 14\text{kHz}$。此时尽管 9kHz、11kHz、13kHz、… 以及其他更高次谐波会混叠到信号保留频带的频谱范围内，但由于其幅值小于 1/2 LSB 而影响很小，而低于 7kHz 的频率分量则根本没有混叠到滤波器频带范围。

### 2.4.3 周期信号的 *M/N* 采样方法

对被测周期信号进行采样的方式可以分为实时采样和等效时间采样两类。前者是按被测信号本身的先后发生过程等周期地依次采样获取数据，最为简单和直观；后者则指在信号周期重现的条件下通过适当的设计，用较低的速率进行采样后，通过一定的算法将采样的数据重新排列组合，形成一个采样密度较高的波形数据。*M/N* 采样法就是这样一种实现等效时间采样功能的具体算法。

如图 2-48 所示，设测量主周期为 $P$，使其包括 $N$ 个采样周期，同时也包括 $M$ 个信号整周期，即

$$P = M \cdot T_m = N \cdot T_s \tag{2-38}$$

式中，$T_m$ 和 $T_s$ 分别表示信号整周期和采样周期，$M$ 和 $N$ 要求互为质数。

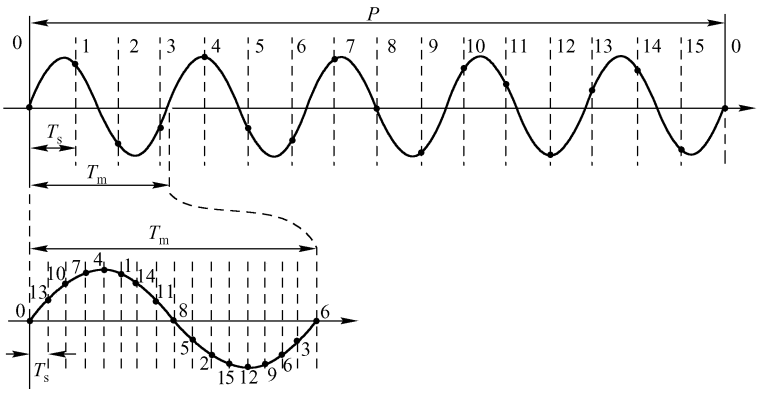



图 2-48 *M/N* 采样法示意图

为了体现波形的细节，可以重新安排实际采集数据的采样排列次序，具体方法如下。

对一组已经确定了的 $M$ 和 $N$，必然存在一个整数 $I$，使得

$$(M \cdot I) \mod N = 1 \tag{2-39}$$

这里，mod 表示取模。如 9 mod 8 = 1，17 mod 7 = 3 等。要求重新安排次序后，前后相邻的两个采样点应为原先实际采样序列中间隔等于 $I$ 的两个采样点。

现在，用图 2-48 作为使用 *M/N* 采样法的一个例子。选择 $M = 5$ 和 $N = 16$ 后，再根据式(2-39)来求取 $I$。试取 $I = 13$，并代入 $M = 5$ 和 $N = 16$ 后，可得 $5 \times 13 \mod 16 = 1$，表明试取成功。这样由 $I = 13$，根据前述规则可依次确定重新排序后的主波形序列如下：

0+13 = 13，(13+13) mod 16 = 10，(10+13) mod 16 = 7，(7+13) mod 16 = 4，(4+13) mod 16 = 1，(1+13) mod 16 = 14，…

由此取得的有效波形序列的采样周期称为有效采样周期 $T_p$，它是实际采样周期 $T_s$ 的 $1/M$。相应的有效采样频率 $f_p=1/T_p$，则为实际采样频率 $f_s$ 的 $M$ 倍。因此只要 $M$ 足够大，就可大大降低对 A/D 转换器转换速率的要求，许多高速数字示波器和高速数字化仪常常利用这种工作原理。由于在这种情况下用于决定混叠信号频带的采样频率限制仍基于有效采样频率 $f_p$，而不是实际采样频率 $f_s$，故这些仪器的通道电路仍然需要满足和原先要求同样高的频率响应要求。

如图 2-49 所示，还有一种称为“Beat-Frequency”的特殊形式。

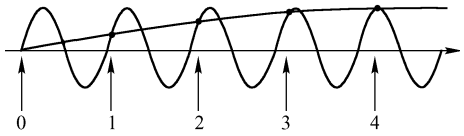



图 2-49　Beat-Frequency 采样

取 $M=N+1$，根据式(2-39)，必有 $I=1$，因为 $(N+1)\times 1$ 对 $N$ 取模等于 1，因此不必计算采样序列就可直接得到有效波形，并有 $T_p=T_s/(N+1)$ 和 $f_p=f_s\cdot(N+1)$ 的关系。

## 本 章 小 结

数据采集器是虚拟仪器中最典型的设备，其核心是 A/D 转换器，主要指标为分辨率和转换速率。为了获得较高的测量精度，需要确保参考电源具有很好的稳定特性和温度特性。通道调理电路作为其与外界应用环境联系的纽带，通常包括放大器、隔离器、滤波器、多路通道开关等典型器件，在系统设计和实际运行过程中地位重要。通常数采的控制总是需要某种形式的微控制器及总线接口的支持，以便同时完成数据的采集、存储、处理和通信等功能，因此需要建立定时采样和实时操作系统的概念。采样定理揭示了数据采集过程中信息混叠的现象，为采样频率和抗混叠滤波器的设计提供了依据。为了实现等效时间采样，可以采用 $M/N$ 算法。

## 参 考 文 献

[1] Gayle F.Miner, David J.Comer. Physical Data Acquisition For Digital Processing. Prentice-Hall Inc., 1992.

[2] Tien Lang Tran. Electronics of Measuring System. Practical Implementation of Analogue and Digital Techniques. John Wiley & Sons Inc., 1987.

[3] DSP-Based Testing of Analog and Mixed Signal Circuits. IEEE Catalog Number EH0258-4.

[4] 沈兰荪. 数据采集技术. 中国科技大学出版社, 1990.

[5] 应怀樵. 波形和频谱分析与随机数据处理. 北京：中国铁道出版社, 1985.

[6] 赵新民, 王祁. 智能仪器设计基础. 哈尔滨：哈尔滨工业大学出版社, 1999.

[7] 魏庆福等. STD 总线工业控制机的设计与应用. 北京：科学出版社, 1991.

[8] 赵新民.智能仪器原理及设计.哈尔滨：哈尔滨工业大学出版社, 1990.

[9] 冯勇. 现代计算机控制系统. 哈尔滨：哈尔滨工业大学出版社, 1996.

[10] Jean J.Labrosse. μC/OS-Ⅱ The Real-Time Kernel. Labrosse, 1992.

[11] 刘思久，孙钊. 一种支持网络功能的嵌入式 ARM 控制器.电测与仪表，2006(8).
[12] 易劲刚，傅尚新，瞿笛横，张礼勇. 智能数据采集系统. 电测与仪表，1996(8).
[13] 郭松林，张礼勇，林海军. Σ-Δ 型 A/D 转换器的原理级分析.电测与仪表，2002(11).
[14] 闫景富，李淑秋. LVDS 和 CML 电平在高速串行连接中的应用. 微计算机应用，2008.8.
[15] 邵华，刘亚斌，徐志跃. 实时高速 LVDS 串行数据采集系统的设计. 计算机测量与控制，2008.
[16] 王玲. 滤波器的研究与仿真. 工业控制计算机，2008.

# 第 3 章　GPIB 通用接口总线

GPIB 是通用目的接口总线(General Purpors Interface Bus)的英文缩写。它自 1975 年由美国电气与电子工程师学会批准命名为 IEEE-488 标准、1978 年经国际电工委员会表决形成 IEC-625 规范后，30 年来经久不衰，已被全世界各仪器厂家所接受。世界上几乎所有的台式仪器均能配置 GPIB 接口总线，而且这种接口总线也是 VXI 主机箱与 PC 的典型连接方式，因此对于测试工程师组建自动测试系统来说，掌握这门技术无疑是很有意义的。

## 3.1　GPIB 通用接口总线的基本原理

GPIB 接口技术的实质是在自动测试系统中各设备之间建立一个能够相互通信的协议。本节首先介绍接口总线各信号线的消息含义，从而阐明这种接口总线的基本概念，然后通过分析规范要点来说明其工作原理。

### 3.1.1　GPIB 通用接口总线的基本概念

参看 1.1 节及图 1-1，在 GPIB 总线的 16 根信号线中，有 8 根信号线用于仪器间通信的数据线，它们并行组成字节传递信息，被称为多线消息。而另外 8 根信号线分别传递一种信息，被称为单线消息。其中，DAV、NRFD、NDAC 3 根线为握手线(HandShake)，即在每一次多线消息传递的过程中，都要通过它们建立的“三线握手”机制来确保多线消息毫不含糊的异步通信。另外 5 根信号线 IFC、REN、ATN、SRQ 和 EOI 为接口管理线。这些信号线的简要说明见表 3-1。

表 3-1　GPIB 接口总线信号线简介

| | |
|---|---|
| DIO1-DIO8 | 双向数据线，共 8 根，用于传递字节形式的多线消息 |
| NRFD | 握手线，用于接收方表示尚未做好接收数据的准备 |
| DAV | 握手线，用于发送方表明发送的数据有效 |
| NDAC | 握手线，用于接收方表示尚未完成了一个多线消息的接收 |
| IFC | 控制线，用于控制器复位总线上所有设备的接口 |
| REN | 控制线，用于控制器允许总线上所有设备进入远地控制状态 |
| ATN | 控制线，用于控制器指定当前数据总线上传递的多线消息为接口消息，还是仪器消息 |
| EOI | 控制线，用于数据发送方表示一组多线仪器消息结束，也用于控制器执行并行点名 |
| SRQ | 控制线，用于总线上设备向控制器请求服务 |

需要强调的是，GPIB 总线上传递的信息能够分为两类，一类是为完成测试任务所需要的实质性信息，如设定测试仪器工作条件的程控命令以及获得测试结果的测量数据，称为仪器

消息；另一类则是为了完成上述仪器消息的传递而使总线上各设备接口处于适当状态所需要的接口管理信息，如指出设备通信的听、讲地址，称为接口消息。由于接口消息和仪器消息都可利用数据线在“三线挂钩”机制下传送多线消息，故专门采用一个单线消息 ATN 来加以区分。当 ATN = 1 时，数据线上传递的多线消息为接口消息；反之，ATN = 0 时则为仪器消息的传递。

GPIB 接口通常有 3 种最基本的工作角色——讲者、听者和控者。而控者又分为负责控者和系统控者两种功能。只有接口状态进入负责控者作用的设备才能控制 ATN 线，并在 ATN = 1 的状态下发多线接口消息来控制总线上其他设备的接口状态。在同一时刻，总线上只能有一个设备进入负责控者作用状态，但负责控者是可以转移的，即当前处于负责控者作用状态的设备可以放弃负责控者的地位，把控制权转移给总线上的其他设备，而使自己的负责控者功能处于空闲状态。与此不同，总线上只能有一个固定不变的设备具有系统控者功能，其作用是控制单线消息 IFC 和 REN。当 IFC = 1 时，就强迫命令总线上所有设备的接口状态复位，使其进行相应的初始化操作，并称为接口清除。而发 REN = 1，则允许系统中各设备进入远地控制工作方式，称为远地功能。这里远地控制是指仪器接受来自 GPIB 总线的系统控制，而本地控制则是指仪器接受其本身的面板控制。

GPIB 总线中控者发出的多线接口消息由表 3-2 列出，它们可以分为通令、专令(或称指令)、址令(或称地址)和副令几类。通令要求总线上所有的设备都接收，而专令则只要求已被寻址到的指定设备接收。严格地说，址令是通令的一种，专门用于设备的寻址处理，其发出时总线上所有设备都要接收。副令是依附于通令、专令或指令的子命令。

**表 3-2 GPIB 通用接口总线的多线接口消息**

| | 名 称 | 代 号 | 编 码 |
|---|---|---|---|
| 通令 | 本地开关封锁 | LLO | X 0 0 1 0 0 0 1 |
| | 全部器件清除 | DCL | X 0 0 1 0 1 0 0 |
| | 允许串行点名 | SPE | X 0 0 1 1 0 0 0 |
| | 禁止串行点名 | SPD | X 0 0 1 1 0 0 1 |
| | 取消并行点名组态 | PPU | X 0 0 1 0 1 0 1 |
| 专令 | 执行群触发 | GET | X 0 0 0 1 0 0 0 |
| | 进入本地 | GTL | X 0 0 0 0 0 0 1 |
| | 并行点名组态 | PPC | X 0 0 0 0 1 0 1 |
| | 选择器件清除 | SDC | X 0 0 0 0 1 0 0 |
| | 取控 | TCT | X 0 0 0 1 0 0 1 |
| 址令 | 听寻址 | LAD(MLA) | X 0 1 $L_5$ $L_4L_3L_2L_1$ |
| | 讲寻址 | TAD(MTA) | X 1 0 $T_5$ $T_4T_3T_2T_1$ |
| 副令 | 副地址 | SAD | X 1 1 $S_5$ $S_4S_3S_2S_1$ |
| | 禁止并行点名 | PPD | X 1 1 1 $D_4D_3D_2D_1$ |
| | 允许并行点名 | PPE | X 1 1 0 S $P_3P_2P_1$ |

为了接受寻址，每个设备必须有一个取值为 0~30 的地址，实际仪器中的设备地址通常通过其后面板上的一个地址拨码开关设置。设备将收到的址令与其自身设定的地址比较：相符时表明寻址成功，该设备将接受其后总线控制器发来的专令；反之则对其后发来的专令不

予理会。如通过地址拨码开关将设备地址设置为 3，则当总线控制器发 LAD 址令，即在 ATN = 1 的条件下发出多线消息×0100011 时，就启动了设备的听功能，并在随后 ATN = 0 的条件下进入听者工作方式。

GPIB 接口的讲者和听者工作方式分别表达了接口在 ATN = 0 的情况下发送和接收仪器消息的能力。GPIB 系统工作的典型过程是由系统中控制器(常为计算机)的系统控者功能通过依次发布单线消息 IFC 和 REN，首先初始化总线上的所有设备，并使它们处于“远地可能”的状态。然后再由控制器的负责控者功能在发布单线消息ATN = 1 的条件下，利用多线接口消息任命总线上一个设备为讲者，另一个或另几个设备为听者。接着就在发布单线消息 ATN = 0 的条件下，使进入讲者作用状态和听者作用状态的设备在“三线挂钩”的配合下，一字节、一字节地通过数据总线传递多线仪器消息。同时讲者设备也可以通过发送单线消息 EOI 来向控者设备报告一组仪器消息传递的结束。仪器消息的通信常常采用 ASCII 编码。

设备可以利用发送单线消息 SRQ 来请求控制器中断现行工作为其服务。由于 GPIB 采用低电平有效的负逻辑机制，因此总线中只要有一个设备发出 SRQ = 1(低电平)，则总线就呈 SRQ = 1 状态而激活控制器。控制器在接收到服务请求后，往往采用“串行点名”的办法来判断发出 SRQ 的请求者及其设备状态。即将总线上各设备依次任命为讲者，然后根据它们分别发回反映各自设备状态的“状态字节”来判断服务请求的性质，并进行相应的处理。控制器检查总线上各设备工作状态的另一个方法是 “并行点名”，即通过预先将每条数据线分别配置给不同的设备，然后在单线消息 ATN = 1 和 EOI = 1 的条件下，通过同时读取 8 条数据线来获取反映各设备状态的“状态比特组”。显然，“并行点名”的过程较快，但一次最多只能检查 8 个设备的状态。

### 3.1.2　GPIB 规范的要点说明

为了更好地确切说明 GPIB 接口系统，实际规范是从机械、电气、功能 3 个方面分别予以明确阐述的。机械要素规定了接口连接器采用一种“猪脊”式的 24 芯专用连接器(如图 3-1 所示)，并规定了各引脚的信号定义。而对于 16 根信号线以外的其他连接线，则被用做接地线和屏蔽线。连接器由一个插头和一个插座背靠背地组装在连接电缆的两端，当把其插头插入仪器的接口插座时，又自然地将另一个插座提供给其他连接电缆的连接插头。

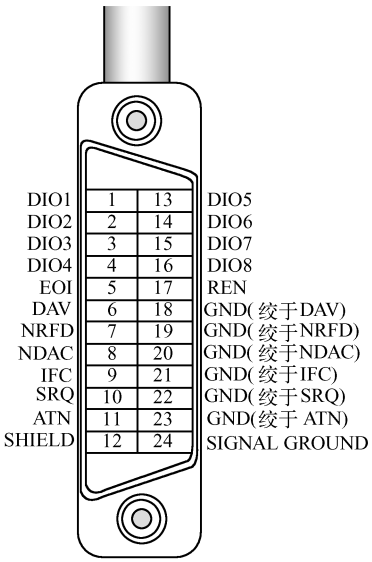



图 3-1　GP-IB 总线连接器

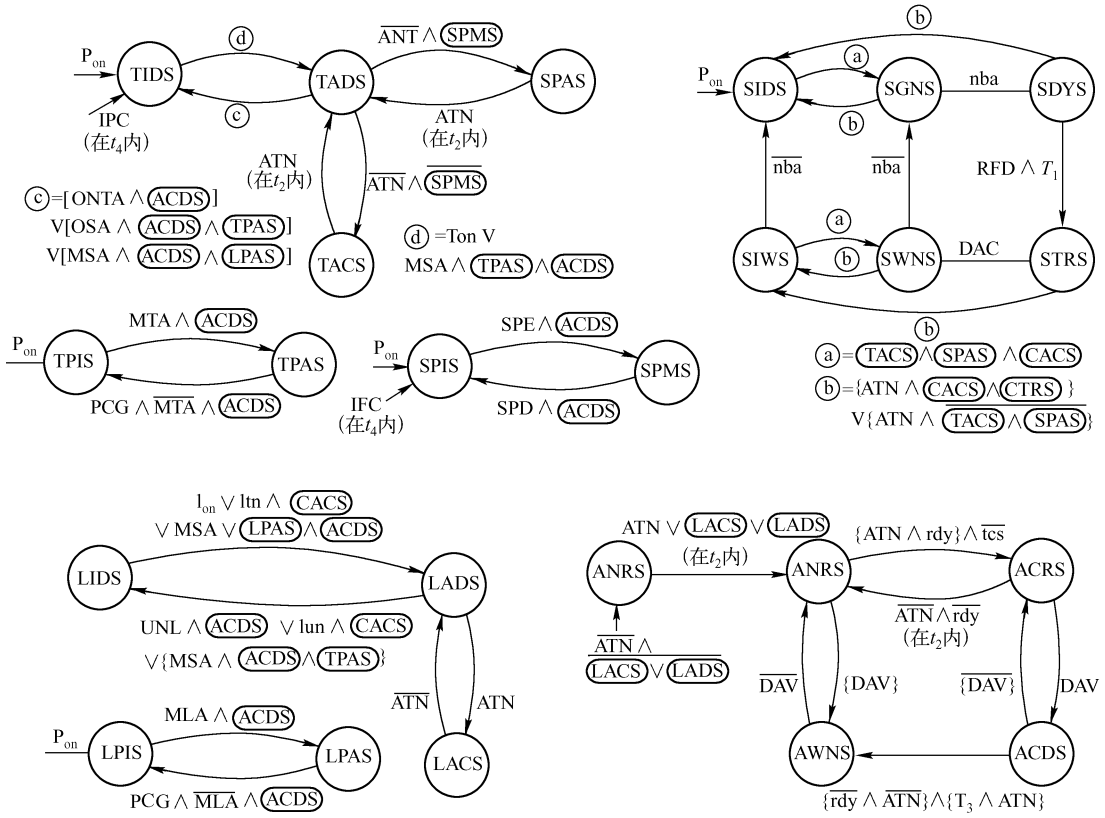
电气要素规定了信号线为负逻辑的 TTL 电平，即以低于+0.8V 为逻辑真“1”，以高于+2.0V 为逻辑假“0”，低电平拉电流的负载能力为 48mA。考虑到总线负载的影响，同时接在总线上的仪器被限制在 15 台以下。通信距离限制在 20m 内，通信速度理论上最高可达 1MB/s，实际应用中受器件和环境的限制则可能只有几十 KB/s。

功能要素通过功能状态图对 GPIB 接口的 10 种功能予以确切描述。这 10 种功能是讲功能(T，TE)，听功能(L，LE)，控功能(C)，源握手功能(SH)，受握手功能(AH)，

服务请求功能(SR)，并行点名功能(PP)，远地/本地功能(RL)，仪器触发功能(DT)和仪器清除功能(DC)。通常被控的程控仪器需要支持除控功能以外的其他9种接口功能，而作为控制器的计算机则需要支持控功能，以及听、讲功能和源、受握手功能。GPIB规范还规定了各种功能的子集，以使各仪器接口的功能配置情况能够一目了然。如并行点名功能分PP1和PP2两种情况，前者可以完成并行点名的全部功能，后者则由于删去了一些状态，不能通过副令PPD和PPE进行远地组态并行点名报告。当需要在仪器接口说明中表达后一种情况时，只须标明PP2即可。

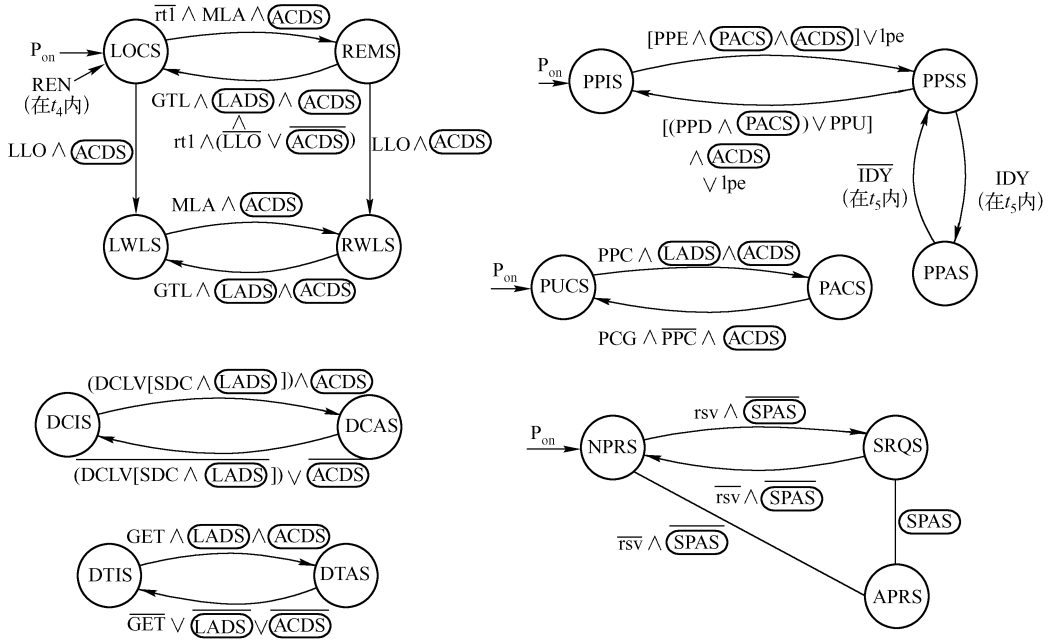
下面结合如图3-2所示规范的功能状态关系具体阐述GP-IB的各功能工作原理。

GPIB功能状态图分为若干组，每组内部存在若干状态，并用圆圈将缩写的状态名称圈起，状态之间的变迁条件直接写到其带有箭头指向的连线旁边(有时为了避免表达式过长在图中表达不便而将其用符号代替，再在注释中说明)。大写字母组成的标识符表示来自远地控者的单线或多线接口消息，小写字母组成的标识符表示来自本地仪器的内部接口消息，而变迁条件中用封闭线圈起的大写标识符表示设备接口本身其他功能组的接口状态。变迁过程至少需用的时间可用 $T_x$ 表示(这里 $x$ 为数字标号)，表示至多需要的时间则可用 $t_x$ 表示。变迁条件通常需要这些基本因素构成的与、或、非逻辑组合。下面结合如图3-2所示规范的功能状态关系具体阐述GPIB各接口功能的工作原理。

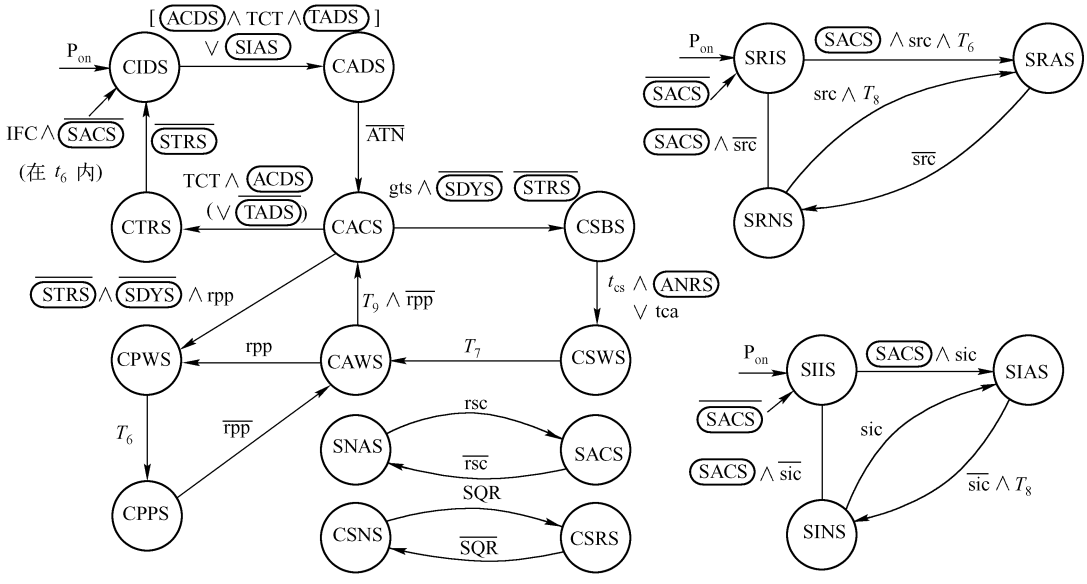



(a) 讲功能、听功能、源挂钩功能和受挂钩功能

图3-2　GPIB功能状态图




(b) 远地功能、仪器清除和触发功能、并行点名功能、服务请求功能




(c) 控功能

图 3-2　GPIB 功能状态图(续)

### 1. 三线握手技术与接口的源握手和受握手功能

不论是接口消息还是仪器消息，只要是多线消息，数据线上的传输都要通过三线握手来完成，为此设备接口中就要配置源握手功能和受握手功能。源握手功能的源方可以是控者，

也可以是讲者。在控功能处于控者作用态 CACS 时，要求能在 ATN = 1 的条件配合下发多线接口消息，于是启动源握手功能。同样，在讲功能处于讲者作用态 TACS 或串行点名作用态 SPAS 时，要求能在 ATN = 0 的条件配合下发多线仪器消息或状态报告，此刻也要启动源握手功能。如图 3-2(a)所示，源握手功能在上述 3 种情况下启动源握手(即从空闲态 SIDS 进入产生态 SGNS)后，只要本地消息 nba(数据字节准备好)为真，则源功能就会进入 SDYS 态准备发送数据并监视 NRFD 线，等待受者发来准备好接收数据的信息。

再看受握手功能，受方在总线 ATN = 1 下接收接口消息时，或听功能被寻址到听者作用状态 LACS 且有本地消息 rdy 准备接收数据时，受握手功能就从受者未准备好状态 ANRS 进入受者准备好状态 ACRS，发 NRFD 为假(即 RFD = 1)，准备接收数据。同时监视 DAV 线，等待源方发来的表明数据有效的信息。源方在测定 NRFD 线为假(RFD = 1)，即确认受者已经准备好后，则经 $T_1$ 时间将状态转移至源传送状态 STRS，并发 DAV = 1，宣布数据有效。与此同时，监视 NDAC 线，等待受者收到数据。受者得到 DAV = 1 便进入接受数据态 ACDS，在收到仪器消息或经 $T_3$ 时间收到接口消息后进入 AWNS，并发 NDAC 为假(DAC = 1)，表明已经接收到数据。源方在得到 DAC = 1，确认所发数据已被接收后进入等待态 SWNS，发出 DAV = 0，结束这一字节的传送。而受方也在收到 DAV = 0 的信息后回到 ANRS 态，结束这一字节的接收。三线握手的时序过程如图 3-3 所示，注意信号均为低电平有效。

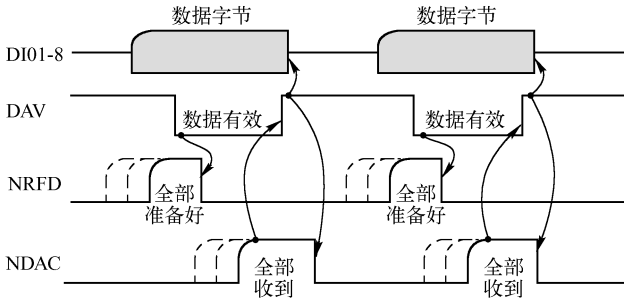



图 3-3 三线握手的时序过程

### 2. 听功能和讲功能的寻址过程

GPIB 总线系统中各仪器被挂在同一组总线上，那么仪器是如何分别接收针对各自的特定指令呢？显然只有事先寻址，寻址命令要求所有仪器都必须接收。参见表 3-2，接口消息中寻址命令的低 5 位(第 0~4 位)是真正的地址号，第 5 位是听地址标志，第 6 位是讲地址标志。当控制器发出的接口消息中的地址码与仪器设备本身所设地址匹配一致时，即可认为寻址成功(状态图中标为 MLA 或 MTA)，从而启动仪器接口的听功能或讲功能。从图 3-2(a)中则表现为从听者空闲态 LIDS 或讲者空闲态 TIDS 转移到听者受命态 LADS 或讲者受命态 TADS。这样，当控者发送 ATN = 0 时，接口再由 LADS 或 TADS 转移到 LACS(听者作用状态)或 TACS(讲者作用状态)。由于系统中可以同时存在多个听者，却只能同时存在一个讲者，因此当控者发布一个讲地址时，相当于自动撤销了原先那个讲地址。因为一个设备的听、讲功能不可同时工作，因此任命一个仪器的讲功能时，也就自动地清除了它的听功能，反之亦然。多线接口消息 UNL 是一个特殊的通令，可让所有设备听者功能退回到空闲状态(故称不听址令)。另外，设备本身也可通过本地消息 lon(只讲)或 ton(只听)强迫接口直接进入听者受命态 LADS 或讲者受命态 TADS，而用本地消息 lun 退出听者受命态 LADS。

注意，如图 3-2(a)所示的状态图体现了扩充寻址的概念。在这种寻址方式下，若要接口被寻址后从听者空闲态 LIDS 进入听者受命态 LADS，或从讲者空闲态 TIDS 进入讲者受命

态 TADS，均要求接口收到自己的听地址或讲地址（MLA 和 MTA），分别从空闲态 LPIS 和 TPIS 进入作用态 LPAS 和 TPAS 后，还要紧跟着收到自己的副地址（状态图中以 MSA 表示）。副地址的格式如表 3-2 所示，其字节的第 5 位和第 6 位均为 1。因此有 31 个副地址可以选择，故总共可以有 961 个寻址对象。

### 3. 远地-本地功能

如图 3-2(b)所示，接口的远地-本地功能共有 4 个状态，其中两个远地状态，两个本地状态。所谓远地状态，是指仪器接收并执行接口总线发来的各种程控命令，而本地状态则指仪器执行本身面板的操作控制。本地消息 rtl 通常是指仪器面板上的一个本地开关，用来手动设定仪器返回本地控制状态。在第一组本地-远地状态 LOCS 和 REMS 之间，使 REMS 转移到 LOCS 可以通过本地消息 rtl 的作用实现，也可以通过接收总线控制器发来的指令 GTL 达到同一目的。然而要从 LOCS 转移到 REMS，则只有在本地消息 rtl 为假的条件下接收听寻址才能实现。通令 LLO 用于使远地-本地状态 LOCS 或 REMS 转移成具有封锁本地切换开关作用的远地-本地状态 LWLS 或 RWLS。这时本地消息 rtl 对切换回本地状态不起作用，返回本地只能采用听寻址下接收指令 GTL 的方法，进入远地也只能采用听寻址的方法。注意，单线消息 REN（远地可能）是进入远地的必要条件。无论接口处于何种状态，只要 REN 为假，则总线上所有设备均将回到本地状态 LOCS。另外,状态变迁条件中的 ACDS 表明对受握手作用状态的要求。

### 4. 仪器清除功能和仪器触发功能

仪器清除通常是指仪器本身的复位功能，这与接口清除的概念不同。在如图 3-2(b)所示的状态图中可以看出，仪器接口在接收通令 DCL（全部仪器清除）或接收指令 SDC（选择仪器清除）后，就使仪器清除功能从空闲态 DCIS 变迁到作用态 DCAS，从而产生复位信号提供给仪器本身。

仪器触发可以用于要求几个仪器同时动作的场合，以便同步完成某一任务。如图 3-2(b)所示，事先已由听寻址进入 LADS 态的仪器接口在接收群触发执行指令 GET 时，就会使仪器接口的触发功能从触发空闲态 DTIS 变迁到触发作用态 DTAS，并产生触发信号提供给仪器本身。

### 5. 服务请求功能

服务请求是由被控仪器主动发起的。如图 3-2(b)所示，当本地消息 rsv 发生时，接口从服务请求空闲态 NPRS 进入服务请求作用态 SRQS，将 SRQ 线设置成真。控制器查出总线上的 SRQ 信号后，不知其来自何处，于是发动串行点名过程。先发通令 SPE，允许串行点名，接着任命自己为听者，再逐个对系统中各仪器接口依次进行讲寻址，且在每一次讲寻址后再发 ATN=0，使仪器接口的讲功能进入串行点名作用状态 SPAS，并发回状态报告。同时由于此刻接收串行点名的仪器接口已能断定控制器响应了服务请求，故该接口的服务请求功能就会转移到 APRS 态，自动撤销它在 SRQ 线上的服务请求。注意，只有当本地消息 rsv 撤销时，接口才能重新回到空闲态 NPRS。

### 6. 并行点名功能

并行点名是将总线上每一台仪器都分配一根数据线，每根线代表各自仪器的状态。这样

8 根数据线就能同时查询 8 台仪器的接口状态。如图 3-2(b)所示，并行点名过程可分为组态和执行两个阶段。组态有两种：远地组态的 PP1 子集或本地组态的 PP2 子集。在 PP1 子集中，接口功能在接收并行点名组态指令 PPC 后，并行点名寻址组态功能就由其未作用态 PUCS 进入作用态 PACS。如又收到副令 PPE，就进入并行点名预备态 PPSS，并用 PPE 中提供的组态信息分配自己的并行点名线。如表 3-1 所示，PPE 编码字节的低 3 位表示并行点名分配的总线编号，第 4 位则表示并行点名响应线的电平指示。PP2 与 PP1 的区别在于其组态信息由本地信息 lpe 提供并直接进入 PPSS 态。并行点名的执行是由总线上的 IDY 信号(ATN = 1 与 EOI = 1 同时出现)产生的，即只要总线上同时出现 ATN = 1 和 EOI = 1，则设备的并行点名功能就从预备态 PPSS 变迁到工作态 PPAS，于是接口执行并行点名，通过分配的数据线将仪器状态发送出去。

### 7. 控功能

控功能是控者所需要的特殊功能，实质上就是管理单线接口消息和多线接口消息的功能。如图 3-2(c)所示，控功能的状态图分为 3 组，分别描述系统控者的接口清除功能、远地控制功能，以及负责控者的各种功能。

只有当控制器为系统控者时，才能管理 IFC 和 REN 线，而本地消息 rsc 则用于设置系统控者，即完成系统控制功能从 SNAS 态到 SACS 态的切换。本地消息 sic 和 sre 分别用于发送 IFC 和 REN，在状态图上由它们分别和 SACS 状态的“与”操作来表达其动作必由系统控者完成的条件。

负责控者的核心状态是负责控者作用态 CACS，此时可在 ATN = 1 和 EOI = 0 下发送多线接口消息。同时，CACS 态可以通过接收本地消息 rpp 转移到并行点名执行态 CPPS 去发送 IDY(ATN = EOI = 1)，接收数据线上的并行点名响应。也可以通过本地消息 gts(转预备)转移到负责控者预备态 CSBS，控功能在此状态下发出 ATN = 0 后，总线上已经任命为讲者和听者的设备之间就会传送仪器消息(如向数字电压表发送量程设置的程控命令或接收测量数据)，而此刻并不需要控者参与。在这些仪器消息的传输完成后，接口可通过本地消息 tcs(同步取控)或 tca(异步取控)重新取控转移到同步等待态 CSWS，再经 $T_7$ 和 $T_9$ 时间重新进入控制作用态 CACS。

负责控者是可以转移的，这需要当前负责控者在 ATN = 1 的条件下发送接口消息，将欲成为新的负责控者的器件寻址成讲者受命态后发多线接口消息 TCT 来实现。然后让ATN = 0，完成自身从负责控者的退出，而原先处于控者空闲态的那个新控者器件则在其被寻址进入讲者受命状态下，收到指令 TCT 后就会转移到控者受命状态 CADS，且一旦发现总线上 ATN = 0，则进入控者作用状态 CACS，成为新的负责控者。由于总线上只能有一个设备处于讲者状态，因此只要当前负责控者设备发出新负责控者的讲地址，其本身讲功能也一定会退出，且 TCT 作为通令也一定会被其本身接受，这样的条件就使原负责控者从 CACS 态转移到 CTRS 态，进而在源握手功能离开传递主状态 STRS 时控者功能回到空闲态 CIDS。

GPIB 规范规定用来指导仪器厂家分别开发符合标准的接口功能，但并不限制具体电路的设计。实现上述接口功能要素(即功能状态图描述的各种状态关系及其变迁条件)的具体方法显然是多种多样的，可以借助软件实现，也可以采用硬件电路实现，更多的是采用专用接口芯片的方法实现。当所用接口芯片与标准规定的功能要素存在个别变化时，也常采用修改系统状态图的方式来表述。

## 3.2　可程控仪器的 GPIB 接口开发

可程控仪器内部通常都有一个微处理器，用以控制仪器的工作过程和程控动作，因此大多数 GPIB 接口芯片也都是基于微处理器操作的方式。典型产品有 Intel 公司的 8291/8292/8293 套片、Motorola 公司的 MC68488/3447 套片、TI 公司的 TM9914、NI 公司的 NI4882 等。由于这些芯片的设计思想和操作方式往往大同小异，因此下面仅以 Intel8291A/8293 一组套片为例来说明可程控仪器的 GPIB 接口实现方法。

### 3.2.1　可程控仪器的 GPIB 接口电路

图 3-4 是以 Intel 8291 为核心的 GPIB 接口电路，8291 作为 GPIB 听-讲功能适配器，具有 GPIB 总线接口规范中除控功能以外的其他全部 9 种接口功能，即具有 SH1、AH1、TE1、LE1、RL1、SR1、DC1、DT1 和 PP2。

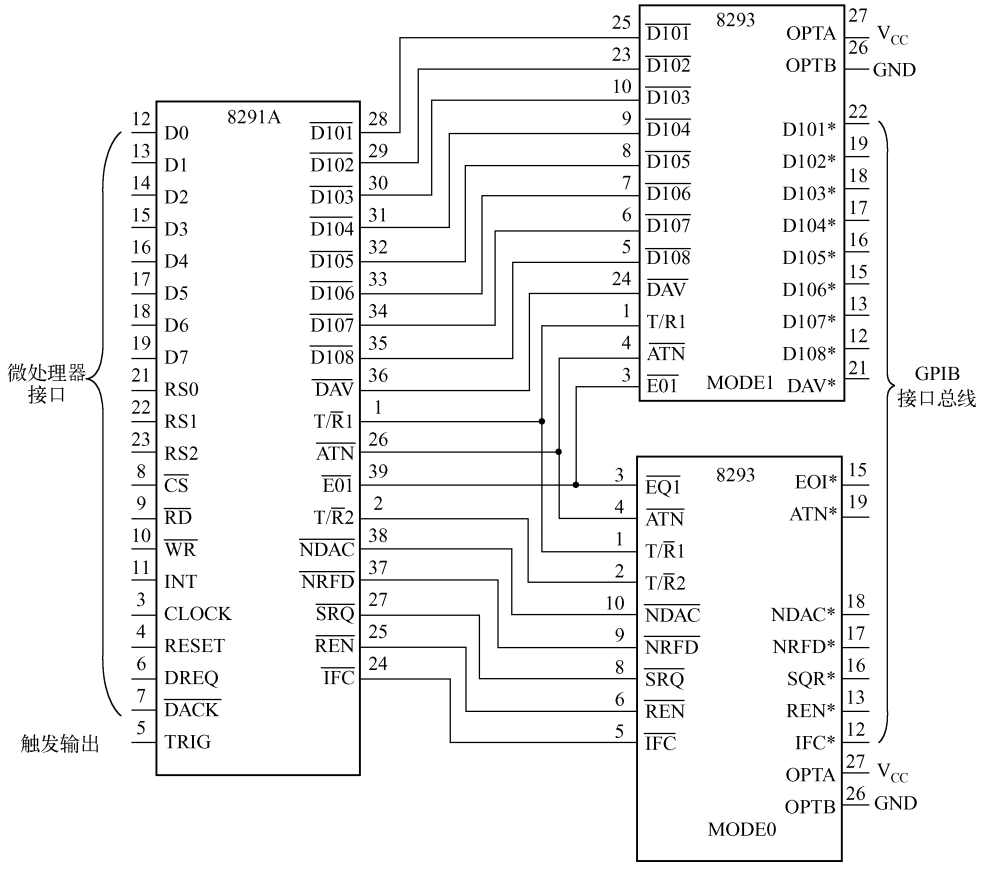



图 3-4　可程控仪器的 GPIB 接口电路原理图

在 8291 的 40 个外部引脚中，包括与 GPIB 总线对应的 16 根信号线，将它们通过总线收发器 8293 接入 GPIB 总线后，这组套片就能自动地识别总线上发来的命令，并在程序固件的支持下实现接口功能。8293 是一种专门用于 GPIB 接口总线的双向收发器，起到非反向的缓

冲输入和驱动输出作用，并满足 GPIB 接口总线规定的电气特性。8293 可以提供 4 种不同的工作模式，分别用于可程控仪器和总线控制器中不同信号线的输入和输出。这样做的好处在于减少芯片的种类，以一种芯片满足不同场合的要求。8293 收发器能以三态门和 OC 门(集电极开路门)两种形式输出，它们均有 48mA 拉电流的驱动能力。三态门输出的速度比 OC 门快，芯片本身提供接口所需要的终端负载。

如图 3-5 所示，8293 中数据传送的方向是由引脚 $T/\overline{R}_1$ 控制的。站在仪器接口的角度，当 8291 源握手功能工作时，数据线 DIO1～DIO8 以及握手控制线 DAV 为发送方向，而握手控制线 NRFD 和 NDAC 为接收方向，此时 $T/\overline{R}_1$ 为真；反之，当 8291 的受者握手功能作用时，$T/\overline{R}_1$ 为假，数据传送被配置成接收方向。作为可程控仪器，8291 总是接收总线控制器发来的接口命令，发出产生的状态报告，因此控制线 IFC、REN、ATN 恒为输入，而控制线 SRQ 恒为输出。需要注意的是控制线 EOI，它具有结束标志 END 和并行点名识别 IDN 的双重作用，为此 8291 专门设计了输出引脚 $T/\overline{R}_2$ 来确定 8293 对控制线 EOI 的驱动作用方向。只有当讲功能作用时，$T/\overline{R}_2$ 信号才为真，并使 8293 对控制线 EOI 的驱动作用方向为输出。而当 ATN = 1 或听功能作用时，$T/\overline{R}_2$ 信号均为假，使 8293 对 EOI 的作用为输入，等待接收并行点名执行命令或数据传送结束消息。

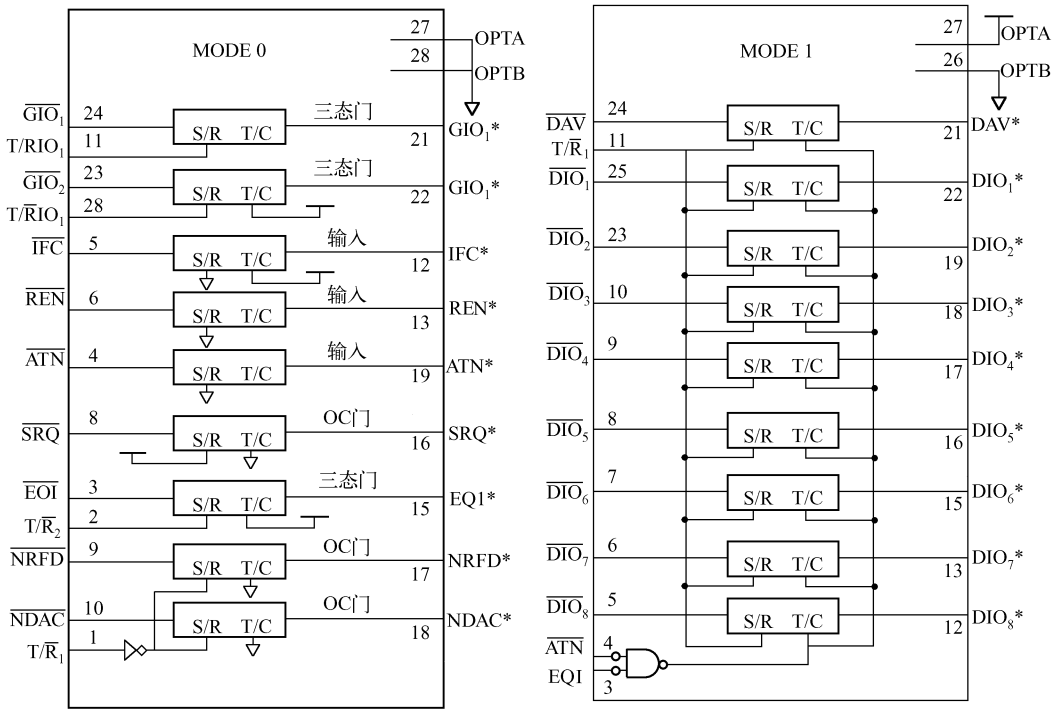



图 3-5 Intel 8293 的引脚接法

8291 的操作只能通过读/写其内部寄存器实现，为此需要将 8291 的 D0～D7 接到微处理器的数据总线；将寄存器选择线 RS0～RS2 接微处理器地址总线的低 3 位，将片选线 CS 接地

址译码器的输出；同时将读/写控制线 $RD^*$ 和 $WR^*$、DMA 请求线 DREQ 及其响应线$\overline{DACK}$、时钟线 CLK、复位线 RESET、中断申请线 INT 分别连接相应的微机控制总线。注意，不用 DMA 方式工作时，$\overline{DACK}$应接高电平。

信号线 TRIG 是响应 GPIB 群触发命令的专用输出线，平常为低电平。在 8291 响应 GPIB 群触发命令时，输出一个宽度为 1μs 的触发脉冲。这样对 GPIB 总线上发来的触发命令，可程控仪器可以通过这个触发脉冲输出信号直接响应接口的仪器触发功能，而不需要通过微处理器的处理过程，从而提高了设备响应触发命令的速度。

### 3.2.2　接口芯片的功能寄存器

8291 内部设置了 16 个功能寄存器，其中 8 个为读寄存器，另外 8 个为写寄存器，分别用于传输数据、状态监视和控制设置。其各位作用见表 3-3 至表 3-5。

**表 3-3　8291 的寄存器设置**

| 只读寄存器 | | | | | | | | 寄存器选择码 | 只写寄存器 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R0R 数据输入寄存器 | | | | | | | | | R0W 数据输出寄存器 | | | | | | | |
| DI7 | DI6 | DI5 | DI4 | DI3 | DI2 | DI1 | DI0 | 000 | DO7 | DO6 | DO5 | DO4 | DO3 | DO2 | DO1 | DO0 |
| R1R 中断状态寄存器 1 | | | | | | | | | R1W 中断允许寄存器 1 | | | | | | | |
| CPT | APT | GET | END | DEC | ERR | BO | BI | 001 | CPT | APT | GET | END | DEC | ERR | BO | BI |
| R2R 中断状态寄存器 2 | | | | | | | | | R2W 中断允许寄存器 2 | | | | | | | |
| INT | SPAS | LLO | REM | SPASC | LLOC | REMC | ADSC | 010 | 0 | 0 | DMA0 | DMA1 | SPASC | LLOC | REMC | ADSC |
| R3R 串行点名状态寄存器 | | | | | | | | | R3W 串行点名模式寄存器 | | | | | | | |
| S8 | SRQS | S6 | S5 | S4 | S3 | S2 | S1 | 011 | S8 | rsv | S6 | S5 | S4 | S3 | S2 | S1 |
| R4R 地址状态寄存器 | | | | | | | | | R4W 地址方式寄存器 | | | | | | | |
| ton | lon | EOI | LPAS | TPAS | LA | TA | MJMN | 100 | T0 | L0 | 0 | 0 | 0 | 0 | ADM1 | ADM0 |
| R5R 命令传递寄存器 | | | | | | | | | R5W 辅助模式寄存器 | | | | | | | |
| CPT7 | CPT6 | CPD5 | CPT4 | CPT3 | CPT2 | CPT1 | CPT0 | 101 | CNT2 | CNT1 | CNT0 | COM4 | COM3 | COM2 | COM1 | COM0 |
| R6R 地址状态寄存器 1 | | | | | | | | | R6W 地址设置寄存器 | | | | | | | |
| X | DT0 | DL0 | AD5 | AD4 | AD3 | AD2 | AD1 | 110 | ARS | DT | DL | AD5 | AD4 | AD3 | AD2 | AD1 |
| R7R 地址状态寄存器 2 | | | | | | | | | R7W 结束符寄存器 | | | | | | | |
| X | DT1 | DT2 | AD5 | AD4 | AD3 | AD2 | AD1 | 111 | EC7 | EC6 | EC5 | EC4 | EC3 | EC2 | EC1 | EC0 |

**表 3-4　8291 中断状态寄存器的各位设置**

| 控 制 位 | 意　义 | 控 制 位 | 意　义 |
|---|---|---|---|
| CPT | 表示收到一个未定义的接口命令 | INT | 表示芯片中断线状态为高 |
| APT | 表示收到一个副地址 | SPAS | 表示接口已被启动并行点名 |
| GET | 表示收到一个群触发命令 | LLO | 表示接口处于本地封锁状态 |
| END | 表示收到 EOI 单线消息或一个结束符 | REM | 表示接口处于远地控制状态 |
| DEC | 表示收到一个仪器清除命令 | SPASC | 表示串行点名作用状态发生改变 |

| | | | |
|---|---|---|---|
| ERR | 表示接口发生错误，总线上无听者作用 | LLOC | 表示本地封锁状态发生改变 |
| BO | 表示一字节的数据已经发出 | REMC | 表示远地/本地方式发生变化 |
| BI | 表示一字节的数据已经接收 | ADSC | 表示寻址状态发生变化 |

**表 3-5 8291 辅助模式寄存器的各位设置**

| 控制 | 命 令 字 | 命 令 说 明 | 位 | 意 义 |
|---|---|---|---|---|
| 000 | 0 CCCC | 执行辅助命令 | | |
| | 0 0000 | 如同电源复位 | $A_0$ | RFD/DAV 闭锁 |
| | 0 0010 | 初始化命令，如同硬件 RESET | $A_1$ | 出现 END 时 RFD/DAV 闭锁 |
| | 0 0011 | 握手结束 | $A_2$ | 收到 EOS 时产生 END 中断 |
| | 0 0100 | 执行群触发，如同 GET 作用 | $A_3$ | 发送 EOS 时输出 EOI |
| | 0 1101 | 进入本地(rtl=1) | $A_4$ | 完整 8 位输出 EOS |
| | 0 0110 | 发送 EOI | | |
| | 0 0111 | 副地址无效 | $B_0$ | 允许未定义命令传送 |
| | 0 1111 | 副地址有效 | $B_1$ | 发送状态字节时输出 EOI |
| | 0 1001 | 执行并行点名置位(ist=1) | $B_2$ | 允许高速数据传输 |
| | 0 0001 | 执行并行点名复位(ist=0) | $B_3$ | 设置 INT 低电平有效 |
| 001 | 0 FFFF | 设置分频系数 | | |
| 100 | $A_4A_3A_2A_1A_0$ | 设置辅助寄存器 A | U | 允许并行点名 |
| 101 | $0\ B_3B_2B_1B_0$ | 设置辅助寄存器 B | S | 设置并行点名标志 |
| 011 | $U\ S\ P_3P_2P_1$ | 设置并行点名组态 | $P_{123}$ | 组态并行点名线 |

### 1. 中断寄存器组

中断寄存器组包括两个中断状态寄存器(R1R，R2R)和两个中断允许寄存器(R1W，R2W)。前者用于指示中断源或事件发生的性质，后者用于设置中断允许条件。注意，无论中断允许位是否置位，中断状态寄存器中的状态位均依对应事件的出现而置位。当微处理器读取该寄存器或执行本地消息 pon 时，中断状态寄存器被清除复位。如果在微处理器读取某中断状态寄存器时又出现了另一个事件，则该事件被保留，直至中断状态寄存器被清除后再重新设置该事件标志。

状态位 ERR 为 1 说明接口总线出现了错误，如 8291 作为讲者发送数据时总线上没有听者。当接口接收通令 DCL 或指令 SDC 并进入仪器清除作用状态 DCAS 时，中断状态寄存器 R1R 的状态位 DEC 置位。这样微处理器就能够根据对此状态位的识别，将程序转入仪器本身的初始化过程，实现仪器清除功能。当接口接收 GET 命令时，中断状态寄存器 R1R 的状态位 GET 置位，于是在确认这个状态后就可转入用于仪器触发的函数过程。事实上，8291 在将 GET 置位的同时，还从其芯片引脚 TRIG 发出一个脉冲信号，以便直接通过硬件电路达到触发仪器本身的功能，从而消除了程序运行的时延。

中断状态寄存器 R2R 的高 4 位用于反映接口的一些特定状态，它们并不自动产生中断。通常通过低 4 位状态变化产生中断，再通过高 4 位判断具体状态值。接口状态变迁时产生中断是十分有用的，接口的远地-本地状态的切换就是通过微处理器响应 REMC 中断后再检查

REM 状态的方法来实现针对面板的操作响应。对本地开关封锁的处理也是一样，先由 LLOC 引起的中断发现 LLO 的状态发生变化，再检查 LLO 状态位来决定对仪器面板远地-本地开关的处理。

### 2. 地址寄存器组与听、讲功能

8291 提供了 5 种不同的寻址方式，可由设置地址方式寄存器的内容来实现，具体参数如表 3-6 所示。

**表 3-6　地址寄存器设置表**

| 地址方式寄存器内容 | 方　式 | 说　明 |
|---|---|---|
| 0 0 0 0　0 0 0 1 | 方式 1 | 两个主地址，分别存于 R6R，R7R |
| 0 0 0 0　0 0 1 0 | 方式 2 | 一个主地址(R6R)，一个副地址(R7R) |
| 0 0 0 0　0 0 1 1 | 方式 3 | 两个主地址(R6R，R7R)，两个副地址(R5R) |
| 1 0 0 0　0 0 0 0 | 只讲(ton) | |
| 0 1 0 0　0 0 0 0 | 只听(lon) | |
| 1 1 0 0　0 0 0 0 | 自讲 | 对自己讲 |

方式 1 为仪器提供了两个地址，分别保存在表 3-3 的地址寄存器 0 和地址寄存器 1 中，其值通过地址设置寄存器 R6W 写入。R6W 的 ARS 位决定写入的是地址寄存器 0 还是地址寄存器 1。DT 位和 DL 位分别用于禁止讲功能和禁止听功能。通常一个仪器接口仅需一个地址，故只使用地址寄存器 0，且禁止第二个主地址的讲、听功能。方式 2 执行扩展的讲/听寻址功能，采用这种工作方式时必须将仪器的主地址装入地址寄存器 0，将副地址装入地址寄存器 1，之后 8291 能自动处理扩展的讲、听寻址操作过程。方式 3 提供两组扩展的地址，将它们的主地址分别装入两个地址寄存器(可通过 R6R 和 R7R 检查)，此刻 8291 本身并无直接功能来识别副地址命令，需在收到主地址命令进入讲者主寻址状态 TPAS 或听者主寻址状态 LPAS 时，将副地址命令放入命令传递寄存器 R5R 中，并置位中断状态寄存器的 APT 位。微处理器响应 APT 中断后就可利用软件方式从命令传递寄存器 R5R 中读取副地址，再把副地址有效标志 OFH 或副地址无效标志 07H 写入辅助模式寄存器中来完成寻址过程。只讲方式或只听方式可在任意时刻设置。置地址方式寄存器 R4W 的 LO 位为 1 相当于产生本地消息 lon；置 TO 位为 1 相当于产生本地消息 ton。

8291 自动地实现接口总线的听/讲寻址过程，用地址状态寄存器 R4R 的各状态位表达接口的寻址状态，其中 ton 和 lon 标志分别表明接口处于只讲或只听寻址方式。LPAS 或 TPAS 为真表示已经收到听主地址或讲主地址，且正等待副地址的到来。状态位 LA 为真表明接口处于 LACS 或 LADS 状态，状态位 TA 为真说明 8291 已经处于 TACS 或 TADS 状态了。在寻址方式 1 和寻址方式 3 的工作过程中，状态位 MJMN = 0/1 用于指出上述听/讲状态所属的具体地址寄存器号。

### 3. 辅助模式寄存器

辅助模式寄存器包括 3 个控制位(CNT0～2)和 5 个命令位(COM0～4)，用于执行来自微处理器的辅助命令、将控制信息装入 8291 内隐藏的辅助寄存器 A 或 B、预置内部时钟计数器

以产生 SH 功能中的 $T_1$ 延时、允许或禁止并行点名及实现其本地组态。这些任务的编码格式见表 3-3。

8291 中的内部定时计数器用来决定 DIO*x* 线上的数据所需要的稳定时间，也就是 GPIB 通用接口总线规范中所规定的时间 $T_1$。总线规范要求讲者把数据字节送到 DIO*x* 数据线后，经过 $T_1$ 时间才把 DAV 线置 1。$T_1$ 是决定 GPIB 数据传输速率的主要因素，当采用集电极开路收发器时 $T_1$ 为 2μs。由于 8291 时钟频率范围为 1~8MHz，所以对不同的时钟频率，内部定时计数器的分频系数也应不同，具体分频系数 $N_f$ 的计算如下：

$$T_1 = \frac{2N_f}{f_c} + t_{SYNC} \quad (\mu s)$$

式中，$f_c$ 为时钟频率（单位为 MHz）；$N_f$ 为分频系数；$t_{SYNC}$ 为同步误差，对于 50% 占空比的时钟，$t_{SYNC}$ 小于脉冲周期的一半。如果 $N_f = 2$，$f_c = 2MHz$，则 $T_1$ 为 2μs。$N_f$ 确定后按二进制格式要求将分频系数写入辅助模式寄存器，便可完成对内部定时计数器的预置。

RFD/DAV 握手信号闭锁是指听者操作时 RFD 被强制封锁为假，讲者操作时 DAV 被强制封锁为假，直到执行结束握手的辅助命令。注意，如果使 A0 = A1 = 1，则会形成一种特殊的“连续 AH 握手循环”方式，这种方式仅用在 8291 和其他芯片构成的控者条件下使 AH 功能状态连续循环，即当 AH 处于 ANRS 状态时自动产生本地消息 rdy。这样，8291 的 AH 功能就可作为控者的 AH 功能，使控者执行本地消息 tcs 时 8291 脱离“连续 AH 循环”方式，让通用接口悬挂在 ANRS 状态。同时产生 BI 中断，表明现在可以采取控制。若希望在数据块结束时执行，可使 8291 处于“连续 AH 循环”，当接收到数据块结束 EOI 或 EOS 时产生闭锁，让接口悬挂在 ANRS 状态，然后采取控制。

#### 4. 数据输入/输出寄存器与握手功能

中断状态寄存器中的 BO 和 BI 位用来控制数据的传输过程。BO 置位表示 8291 已被寻址为讲者，源握手功能正在等待发送数据，此时可将新的待发数据字节写入数据输出寄存器。一旦数据写入数据输出寄存器，BO 复位，同时 8291 开始握手操作，并把数据送到 GPIB 通用接口总线上。此刻若 8291 仍为讲地址作用且 ATN 仍为 0，则 BO 将再次置位。同样，当 8291 被寻址为听者，受握手工作。从通用接口总线接收到的数据字节进入 8291 后，BI 置位；当微处理器读取数据输入寄存器时，BI 被复位。另外，当微处理机读取中断状态寄存器 1 或执行本地消息 pon 时，BO 和 BI 也被复位。8291 的输入和输出寄存器的操作互不影响。

8291 在 DMA 工作方式下，输入或输出数据字节时所产生的 BI 和 BO 状态可以通过 DREQ 引脚发出，作为 DMA 的请求信号。中断允许寄存器 2 中的 DMAI 和 DMAO 是 DREQ 引脚产生 DMA 请求信号的允许条件。DMAI 允许 BI 出现时在 8291 的 DREQ 引脚产生 DMA 申请，DMAO 允许当 BO 出现时在 DREQ 引脚上产生 DMA 申请。当读中断状态寄存器 1 时，DREQ 不受影响。每当数据字节写入数据输出寄存器，或者从输入寄存器读取字节之后，DREQ 被复位。如果 8291 没有使用 DMA 方式，BI 和 BO 状态也可以直接从 DREQ 引脚获取，这样就不必读取中断状态寄存器而直接为 BI 和 BO 中断服务了。

读取输入寄存器的内容之前，RFD 一直为假。8291 每接收一个数据，就自动进行一次握手操作。若 8291 处在 RFD/DAV 闭锁模式，则握手操作暂停，直至微处理器发出释放闭锁命令为止。利用这种模式，可以将同一个数据连续读取多次，或者在对接收到的数据处理完之

前一直闭锁总线上的讲者。当 8291 为讲者使用 RFD/DAV 闭锁模式时数据被保持，直到出现释放命令。

实际接收数据时，8291 发 NRFD 为假，然后检查 DAV 线，直到 DAV 为真时将数据线上的数据锁入数据输入寄存器，产生 BI 标志，并通过 INT 引脚发出中断申请。微处理器响应中断，读取中断状态寄存器，弄清中断性质为 BI 后，读取数据输入寄存器，取得数据字节。8291 在数据锁入数据寄存器后就发 NDAC 为假，待 DAV 也为假后，NDAC 再变为真，然后 8291 置 NRFD 为假，准备接收下一数据字节。发送数据时，数据写入数据输出寄存器后，8291 就将数据送到接口总线的数据线上。8291 检查 NRFD 线，如果准备好就发 DAV 信号。8291 接收到 NDAC 为假后，便使 DAV 信号为假。8291 产生 BO 中断，表明数据输出寄存器已空，可以输出下一字节。

#### 5. 数据块结束(EOS)寄存器

微处理器可将一个作为发送或接收数据块结束标志的字节写入 EOS 寄存器 R7W，简称 EOS 结束符。若辅助寄存器 A 中的 A4 置位，EOS 为 8 位有效，反之为 7 位有效。若 8291 处于讲者作用状态，且辅助寄存器 A 中的 A3 置位，表示允许发送 EOS 字符的同时输出 EOI，即当数据输出寄存器中锁存的字节与 EOS 寄存器中所存的字节一致时，伴随该字节的发送置 EOI 线为真。当 8291 处于听者作用状态，且通过数据输入寄存器收到的数据字节与 EOS 寄存器内容一致或 EOI 线为真时，便使中断寄存器中 END 位置 1，表明接收的一组多字节仪器消息结束。

#### 6. 串行点名寄存器与服务请求功能的实现

串行点名模式寄存器 R3W 中的 6 位相当于请求服务的本地消息 rsv，将其置位则使 8291 的 SRQ 线输出为真，向系统提出服务请求。当服务请求被响应、8291 收到 SPE 消息并被寻址成讲者作用状态时，就会自动地将串行点名模式寄存器 R3W 中建立的状态字节发往接口总线。串行点名状态寄存器 R3R 被用来检查已存入串行点名模式寄存器中的状态字节，当接口的状态字节已经发出，串行点名已经实现时，R3R 中的 SRQS 位就会自动被清除，由此表明 8291 发出的服务请求已被接受，相应的串行点名也已经结束。但是需要注意串行点名模式寄存器 R3W 中的 rsv 位必须通过微处理器重写才能清除。

#### 7. 命令传递寄存器及其在并行点名远地组态中的应用

命令传递寄存器用于锁存接口总线发来的 8291 未定义的多线接口消息码。当允许 CPT 功能作用(即辅助寄存器 B 中 B0=1)时，任何 8291 不能识别的多线接口消息都被看做未定义命令而被锁存在命令传递寄存器中，同时闭锁握手过程并产生 CPT 中断。于是微处理器可读取命令传递寄存器内容，利用软件识别来命令，进行相应处理后再通过辅助命令恢复握手过程。如前所述，8291 工作在寻址方式 3 时，副地址就是通过 CPT 中断并读取命令传递寄存器来处理的。8291 的这种功能很有意义，使其在 GPIB 标准扩充接口命令后仍能适应新的情况。下面仅以并行点名中的远地组态过程为例，说明这种功能的应用方法。

8291 本身的并行点名功能采用 PP2 子集，即只有本地组态点名功能，其方法是在辅助模式寄存器中写入格式为 $011USP_3P_2P_1$ 的编码命令。编码中的位 U 相当于通用接口规

范中定义的本地消息 lpe(本地并行点名可能)，即在 U = 1 时允许响应并行点名。$P_3P_2P_1$ 用于分配自己的并行点名响应线。位 S 是判别位，只要并行点名标志(本地消息 ist)与位 S 一致，就在接口总线执行并行点名时将并行点名消息 PPR = ist 在分配的数据线上发送出去。而本地消息 ist 的形成是由执行辅助命令 01H 或 09H 来实现的。由于 8291 仅能实现本地组态，所以有关并行点名远地组态的接口消息 PPC、PPD、PPE 对其来说都是未定义的命令。因此为了实现并行点名的远地组态就需要利用 8291 的命令传递寄存器，并以微处理器程序配合如下：

(1) 事先设置辅助寄存器 B0，允许未定义命令进入命令传递寄存器并产生 CPT 中断，同时闭锁握手过程。

(2) 在接口闭锁握手功能期间，微处理器读取命令传递寄存器消息并加以识别处理，然后发出 VSCMD 辅助命令，释放握手。

(3) 接收 PPC 消息后再接收第二个未定义命令(PPE 或 PPD 消息)。

(4) 根据软件识别出多线接口消息 PPE 或 PPD，将其内容以并行点名控制信息 $011USP_3P_2P_1$ 的格式写入辅助模式寄存器，然后发出释放握手的辅助命令。

### 3.2.3 可程控仪器的基本工作流程

现代的可程控仪器控制几乎都以微处理器为核心，图 3-6 给出了这种控制方式的典型工作流程。如图 3-6 所示的初始化部分包括微处理器系统本身的初始化和 GPIB 接口电路的初始化工作。前者包括堆栈的设置、中断向量的建立和变量初值的确定，也包括键盘和显示设备的初始化；后者则专为芯片 8291 的初始化。

在初始化后，程序流程进入一个循环体。在这个循环过程中，一方面通过扫描键盘查看和接收本地操作人员的面板操作信息，并执行相应要求的仪器动作。另一方面，通过调用管理 GPIB 接口工作的函数 GP_RunInt( )来响应接口芯片的各种状态处理要求、获取系统发来的远地信息并产生内部约定的功能标志。然后，再根据这些标志分别处理诸如远地-本地切换、本地开关封锁、仪器触发、仪器清除等任务。对远地发来的程控命令则通常先将其依次逐个地暂存在一个命令缓冲区中，见到结束符后再对已经放入缓冲区中的命令进行整体解释，并转变为仪器的动作。另外，在执行仪器功能的过程中，可以根据需要为测量结果建立输出数据，为串行点名准备状态报告，为并行点名准备响应状态线电平，并作为对主机仪器程控命令或有关接口命令的响应信息发出。

为了进一步说明接口功能的具体实现方法，这里也节选了几段典型的函数代码(见程序清单 3-1)，并可参照注释理解其具体含义。

**程序清单 3-1**

```
Hrst(int madr)                          //接口初始化
{
    _outp(P91W5, 0x02);                 //复位
    _outp(P91W6, 0x1F & madr);          //设地址
    _outp(P91W6, 0xe0);                 //第 2 地址禁止讲、听 DT=1, DL=1
    _outp(P91W4, 0x02);                 //方式 2, 只有主地址
    _outp(P91W5, 0x20 |fc);             //设置频率
```

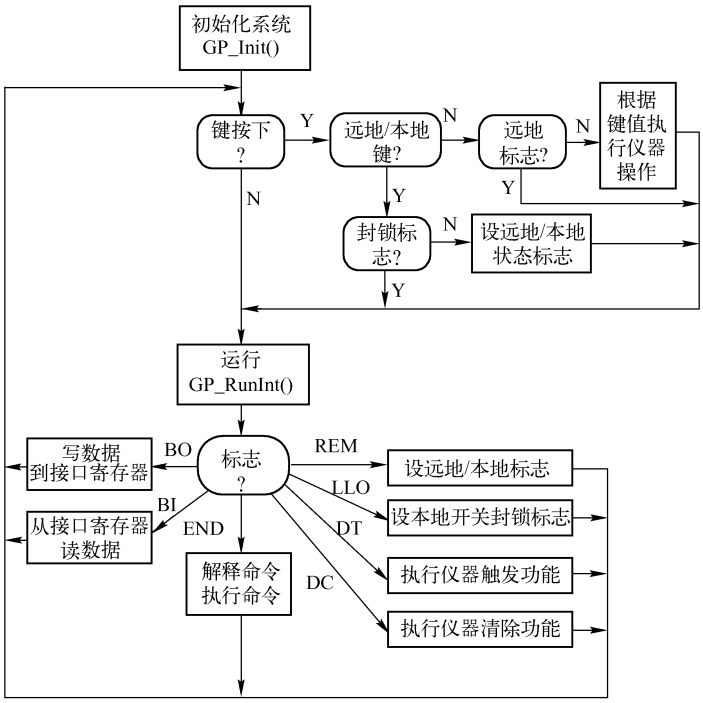



图 3-6　GPIB 可程控仪器的基本控制流程

```
    _outp(P91W5, 0x00);                          //pon
    return 0;
}
Hint(short far *pw)                              //读接口状态
{
    char b1, b2, b4, b5, short w=0, static short fppc;
    b1=_inp(P91R1); b2=_inp(P91R2); b4=_inp(P91R4);
    if(0x80&b2)                                  //CPT=1, 发现未定义的接口命令
    {
        b5=_inp(P91R5);                          //读命令传递寄存器
        switch(b5)
        {
          case TCT: w|=GP_F_TCT; break;                  //读控制权转移命令
          case PPC: fppc=1; break;                       //读并行点名组态命令
          case PPU: fppc=0; _outp(P91W5, 0x60); break;   //读并行点名组态取消命令
        }
        _outp(P91W5, 0x03);              //辅助命令: 握手结束
    }
    if(0x40&b1)                          //APT=1, 收到一个副令/
      if(fppc)
      {
            b5=_inp(P91R5);              //取并行点名副令
            switch(0x60&b5)
            {
                case PPD:
                case PPE: _outp(P91W5, b5); break;     //辅助命令: 写 PPE
```

```
            }
            _outp(P91W5, 0x03); fppc=0;                    //辅助命令：握手结束
        }
    if(0x10&b1) _outp(P91W3, 0);                           //复位串行点名状态位
    if(0x10&b2) w|=GP_F_REM;
    if(0x20&b2) w|=GP_F_LLC;
    if(0x08&b1) w|=GP_F_DCS;
    if(0x20&b1) w|=GP_F_DTS;
    if(0x02&b4) w|=GP_F_TAS;
    if(0x04&b4) w|=GP_F_LAS;
    if(0x10&b1) w|=GP_F_END;
    w=(w|wst); *pw=w;     return w;
}
Hwr (char *data, int n)                                    //发送仪器消息
{
    long l;     short b, j;
    if(!(0x02&_inp(P91R4)))    return GP_ER_TA;
    for(j=0; j<n; j++)
    {
        _outp(P91W0, data[j]);
        for(l=0L; l<nTime; l++)
            if( b=(0x02&_inp(P91R1)) )break;
        if(!b)    return GP_ER_TMO;
    }
    return 0;
}
Hrd (char *data, int n)                //接收仪器消息
{
    long l;     short b=0, j;
    if(!(0x04&_inp(P91R4)))    return GP_ER_LA;
    for(j=0; j<n; j++)
    {
        for(l=0; l<nTime; l++)
            if( b=(0x11&_inp(P91R1)) ) break;
        if(l>=nTime) return GP_ER_TMO;
        if( 0x10&b ) break;
        if( 0x01&b ) data[j]=_inp(P91R0);
    }
    data[j]='\0';     return j;
}
Heos(int w)                            //设置结束方式
{
    char b=0x90;     _outp(P91W7, w&0xff);
    if(w&GP_F_REOS) b|=0x04;
    if(w&GP_F_SEOS) b|=0x08;
    _outp(P91W5, b); return 0;
```

```
}
Htmo(int tmo)                          //设置握手限时
{
    if((tmo<16)&&(tmo>0))    nTime=n100US*tabTime[tmo];
    return 0;
}
Hstb(int bst, int ist)                 //设接口状态字节
{
    _outp(P91W3, (char)bst);           //串行点名状态位
    if(ist) _outp(P91W5, 0x09);        //并行点名状态 ist=1;
    else _outp(P91W5, 0x01);           //ist=0;
    return 0;
}
```

### 1. 初始化过程函数 Hrst

通过将 02H 写入辅助命令寄存器(相当于加脉冲到 8291 的 RESET 引脚)进入初始化状态，于是将 8291 的中断状态寄存器清除(不包括中断允许寄存器)，辅助寄存器 A 和 B 清除，串行点名标志清除，地址状态寄存器中的 EOI 位清除，内部计数器的 NF 置为 8，产生 rdy 本地消息。然后在 8291 内部寄存器中写入其他初始化控制信息，如设置地址工作方式、设置主/副地址、屏蔽中断允许寄存器、设置 EOI 发出方式和输入及输出的结束字符。最终通过写辅助命令“立即执行 pon”(OOH⇒R5W)解除初始化状态。

### 2. 响应接口消息函数 Hint

当接口状态发生变化、需要微处理器固件程序进行处理时可以通过中断方式直接转入此函数，也可在主程序中通过循环调用此函数主动查询。此函数的主要作用是根据 8291 引起 INT 置位的几种接口消息，分别设置不同的标志，从而为主程序转入不同的分支处理建立必要的标志。如接受听、讲寻址后进入相应状态的标志可由 R4R 的状态位 LA、RA 直接得到，建立仪器清除和仪器触发的标志也可由 R1R 的状态位 DEC 和 GET 直接判断，而远地-本地状态标志可在 R2R 的 REMC 置 1 时判断 REM 位，本地开关封锁的状态建立则可在 LLOC 置 1 时判断 LLO 位。另外，该函数也被用来辅助接口硬件电路完成一些新的功能，如当 R1R 的 CPT 置 1 时通过读取命令传递寄存器 R5R 来判断接口命令 PPC 或 PPU，从而设置并行点名组态标志。也在 R1R 的 APT 置 1 时通过 R5R 读取的副令 PPD 或 PPE，得到要求的并行点名组态信息，再利用 8291 的 PP2 功能通过辅助命令实现组态。

### 3. 数据输入和数据输出函数 Hrd/Hwr

中断状态位 BI=1 时表明接口已经收到一字节的数据，于是从数据寄存器 P91R0 读取后暂存到命令缓冲区，同时将该命令缓冲区的指针加 1 后继续运行。若所收字符为设定的结束符或伴随 EOI 信号，则被认定为命令结束，于是设定 END 标志后退出，让主程序解释和执行命令缓冲区中的程控命令。BO 出现表明允许数据输出，通常是把输出数据缓冲区中指针指向的数据字节通过写入 8291 的数据寄存器 P91W0 输出，同时将指针加 1，直至指针等于数

据字长时，伴随 EOI 发结束符。结束符内容和 EOI 的设置可由函数 Hend 建立。在函数 Hrd/Hwr 读/写数据的过程中，均需循环检查相应的接口握手状态，为了防止出现故障时陷入握手状态始终不能满足的死锁，建立了定数进行循环查询等待的超时退出机制，而超时限制的定数可由函数 Htmo 建立。

#### 4. 串行点名和并行点名的状态建立函数 Hstb

将状态报告直接写入串行点名寄存器 R7W，当 rst 位为真时，自动发出服务请求，且当串行点名查到时自动讲出这个状态报告。同时可由 8291 辅助命令 01H(或 09H)写入 R5W 产生并行点名响应状态的本地消息 ist。

#### 5. 本地方式设置函数 Hrtl

函数 Hrtl 用于产生本地消息 rtl，它由 0dH 写入 R5W 执行辅助命令实现。

## 3.3 GPIB 总线控制器

在 GPIB 自动测试系统中，各仪器接口通常由仪器厂家提供，而总线控制器则常采用在 PC 中配置一块具有控功能的 GPIB 接口卡和相应的支持软件来实现。可见对于组建自动测试系统来说，掌握总线控制器的工作方式和使用方法颇具意义。

### 3.3.1 GPIB 总线控制器的工作原理

GPIB 总线控制器的基本任务本质上就是管理 GPIB 的单线接口消息和多线接口消息，接口规范中称为控功能。同时，由于总线控制器还有发送程控命令、读取状态报告和测试结果这样一些处理仪器消息的任务，且发送接口消息本身也需要三线握手，故除了实现接口控功能外，还需要实现源握手功能、受握手功能、讲功能和听功能。而那些在被控仪器的接口中要求执行本地-远地操作的功能、实现仪器清除和仪器触发及响应并行点名和发出服务请求的功能则不再需要。对控制器来说，与设备上述要求对应的是发出远地-本地操作命令，发出器件清除和器件触发命令，发起并行点名和响应服务请求，它们均包括在管理单线和多线接口消息的控功能中。

在如图 3-7 所示的总线控制器工作原理图中，与 3.2 节介绍的情况相似，8291 仍然起到实现听、讲功能和源、受握手功能的作用，而增加的专用芯片 8292 则用来实现 GPIB 接口控功能。8292 实际上是一个微控制器，通过其数据线接口与计算机系统总线交换信息。然后一方面直接管理控功能中的单线消息，另一方面则仍然利用 8291 来实现多线接口消息的管理。由于接口消息是在 ATN = 1 的条件下完成多线消息的传递，而 8291 实现的听/讲功能是在 ATN = 0 的条件下传输多线的仪器消息，因此需要采用一种特殊的方法予以解决。8293 使用工作模式 2 和工作模式 3，以适应控制器条件下总线收发逻辑的处理。由于 8292 的硬件功能并不复杂，因为都是通过 PC 端口来控制单线消息，而且所实现的状态转换也都不是那些频繁发生的情况，因此实际电路常常可以采用一个 GAL 实现的读/写端口寄存器组来代替原先 Intel 公司提供的 8292 器件。下面仅就与此有关的一些处理方法做些分析。

### 1. 单线消息的管理

总线控制器的最常见形式是其处于既是系统控者又是负责控者的情况。在如图 3-7 所示

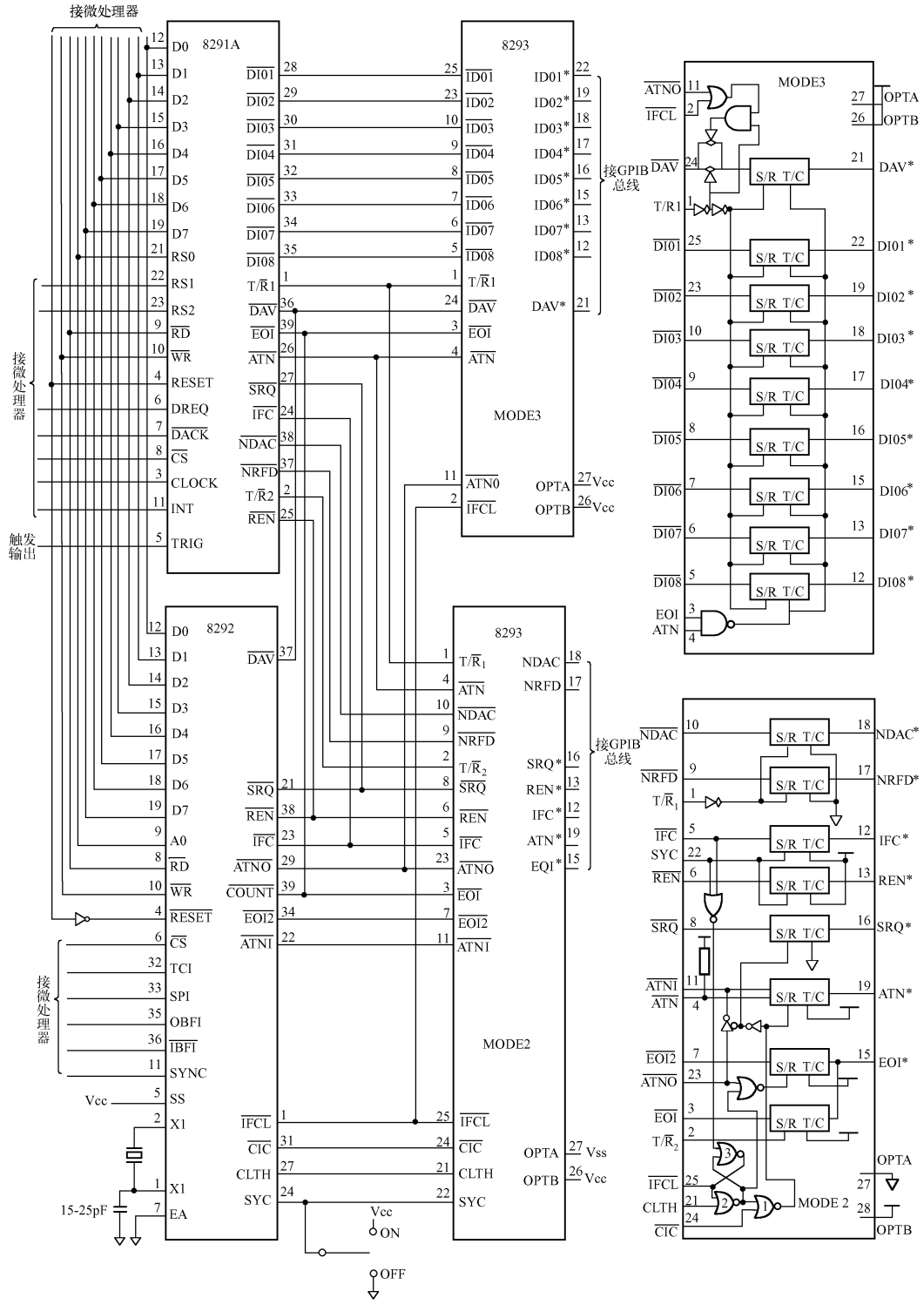



图 3-7　GPIB 总线控制器的电路原理图

的总线控制器的电路原理图中，系统控者设置开关信号 SYC 在上电时为真，决定控功能为系统控者方式。在这种情况下，8293 将其对 IFC 和 REN 的收发器设置成输出形式，且通过 8292 的端口输出实现对 IFC 和 REN 线的控制。当图中取控信号$\overline{\text{CIC}}$为低时，负责控者功能启动，此时通过 8293 内部的门电路将 ATN 线的收发器设置成输出方式，将 SRQ 线的收发器设置成输入方式。同时，8293 还为 EOI 线增加了一个输出形式的收发器，以满足执行并行点名时发送 ATN 和 EOI 为真的控功能需要，ATN 和 EOI 线的控制经 8292/8293 的引脚$\overline{\text{ATNO}}$和$\overline{\text{EIO2}}$实现，以满足负责控者所需要的 3 种状态：

（1）负责控者作用状态 CACS。ATN = 1，EOI = 0，发送多线接口消息。

（2）并行点名执行状态 CPPS。ATN = 1，EOI = 1，接收并行点名数据响应。

（3）控者预备状态 CSBS。ATN = 0，EOI 被动，发送或接收多线仪器消息。

### 2. 多线接口消息的产生和发送方法

针对多线接口消息需要使用数据线及三线握手，可以利用 8291 本身的源握手功能，将 8291 设置成只讲工作方式，即将 8291 的地址方式寄存器 R5W 中的 ton 置位。由于启动负责控者功能时 8293 的$\overline{\text{CIC}}$ = 0，因此将其内部 ATN 收发器设为输出方向，而接到 8291 的 ATN 线引脚由 8293 内部的上拉电阻设置成高电平（$\overline{\text{ATN}}$ = 1），从而使 8291 的讲功能从讲者受命状态进入到讲者作用状态（TADS→TACS），并启动其源握手功能传送多线消息。而另一方面，8293 真正输出到 GPIB 总线上的 ATN 信号却来自 8292 引脚信号$\overline{\text{ATNO}}$，因此达到了借助 8291 在 ATN = 0 的条件下发送多线仪器消息的能力，实现了总线上 ATN = 1 的情况下发送多线接口消息的目的。

### 3. 负责控者的控制权转移

负责控者的控制权是可以转移的。通过对总线上准备接管负责控者权的器件进行讲寻址后，再发多线接口消息 TCT 就可以通过设置 8293 的引脚$\overline{\text{CIC}}$ = 1，使其内部门 1 输出为 0，于是通过反向器和一个三态门将到$\overline{\text{ATNI}}$的通路封锁。同时也将 8293 中对 ATN 的收发器设置成输入方式，对 SRQ 的收发器设置成输出方式，为接收总线接口消息和发出服务请求（如果需要）创造条件。另外，随后也应使 8292/8293 的 ATNO 和 EOI2 引脚电平变高，使其对 ATN 线和 EOI 线的主动作用变为被动作用，从而放弃了对它们的控制。上述过程在接口总线规范中被描述成控功能从控者作用状态到控者控闲状态的转移（CACS→CIDS）。反过来，控功能也可从控者空闲态转移到控者作用态（CIDS→CACS）。接口在控者空闲状态下可以通过 8291 接收当前负责控者发来的讲寻址命令，又可以通过 8291 的命令传递寄存器，利用软件识别出 8291 本身未定义的取控指令 TCT 后，便知道总线上当前控者欲将控制权转移给自己。于是就通过 8292/8293 的引脚信号 ATNI 监视 ATN 状态，直至发现总线上 ATN 信号为假，表明当前控者已经放弃了对总线的控制时，通过 8292 将 8293 的引脚$\overline{\text{CIC}}$和$\overline{\text{ATNO}}$置 0，从而重新获得控制权，成为当前的负责控者。

### 4. 非系统控者工作时的接口清除

当控制器不是系统控者时，则可能接收到系统控者发来的要求接口清除的单线消息 IFC，

此时一方面通过 8291 的相应引脚将其复位，另一方面则由 8293 内部门 3，通过触发接口清除触发器来产生 IFCL 信号，再通过内部门 2 将内部门 1 输出变低，使 ATN 线的收发器为输入方式、SRQ 线的收发器为输出方式，以接收总线上的接口命令。同时利用 8293 的内部逻辑将 DAV 通路打开，使 8291 能够接收多线接口消息。另外，IFCL 被 8292 或相应 GAL 读入识别后将控功能的接口清除，再通过 8292 或相应 GAL 电路向 8293 的引脚 CITH 发出脉冲，结束 IFC 的接口清除作用。

### 3.3.2 总线控制器命令函数的设计和使用

GPIB 总线控制器是面向广大仪器工作者的，然而要让一个普通的仪器工作者熟悉整个 GPIB 接口总线的工作细节实在是太苛求了。因此要求提供一组便于广大仪器工作者使用、面向 GPIB 功能的高级函数。这些函数应当简明易懂，只要了解 GPIB 总线技术的基本概念就可编写测试程序，组建测试系统和实现测试任务，从而达到使那些不懂 GPIB 技术细节的人也能使用 GPIB 技术的目的。同时，采用这种 GPIB 总线控制方式，可使面向用户的库函数调用与实现接口功能的实际电路分开，从而能在保持原有应用接口函数调用格式的前提下不断采用新的 GPIB 芯片开发新的 GPIB 控制器。

GPIB 总线控制器命令函数的配置应根据 IEEE488.2(将在 3.4 节专门阐述)对控者的消息交换要求和协议。同时考虑各类用户使用的方便，可将其做成 Windows 的 DLL 动态连接库形式，以便应用程序调用。表 3-7 给出了用 ATS98 软件工具开发的一组 GPIB 总线控制类函数，可供 VB 环境下以类模块的方式调用。

表 3-7 总线控制库函数 I4882 的主要函数

| | |
|---|---|
| 建立仪器 | SetDevice( dev%( ), ByVal maddr%, ByVal saddr% ) |
| 建立仪器组 | SetDevList( devlist%( ), devs%( ) ) |
| 初始化 | Init( ByVal iMode%, ByVal iTmo% ) |
| 接口初始化 | SendIFC( ) |
| 仪器进入远地 | Remote( devs%( ) ) |
| 仪器返回本地 | Local( devs%( ) ) |
| 本地开关封锁 | SendLLoc( ) |
| 进入远地且封锁 | SendRWLS( devs%( ) ) |
| 仪器清除 | DevClear( devs%( ) ) |
| 仪器触发 | Trigger( devs%( ) ) |
| 发送数据 | Send( devs%( ), data( ) as byte, ByVal n%, ByVal eos% ) |
| 接收数据 | Recv( dev%( ), data( ) as byte, ByVal n%, ByVal eos% ) |
| 等待服务请求 | WaitSRQ( ByVal ntime%) |
| 串行点名服务 | FindSRQ( devs%( ), idev%, stb% ) |
| 串行点名所有仪器状态 | AllSPoll( devs%( ), stblist% ) |
| 串行点名某仪器状态 | ReadStatusByte( dev%( ), stb% ) |
| 并行点名执行 | PPollExec( Byval ppb as byte ) |
| 并行点名组态 | PPollConfig( devs%( ), ByVal iline%, ByVal Sense% ) |
| 并行点名组态取消 | PPollUnConfig( ) |
| 控制转移 | PassCtrl( ByVal dev% ) |

| 控制请求 | RequestCtrl( ByVal iTmo% ) |
|---|---|

在测试程序中，首先需要为系统中每一个仪器定义一个标识名，这是通过函数 SetDevice 完成的。这个函数的参数要求给出仪器的主地址和副地址，它们的有效取值分别为 0~30 和 0~31，副地址参数取值-1 时表示非扩展地址工作方式。由于有些函数功能(如仪器触发 Trigger)需要同时操作一组仪器，因此可以用函数 SetDevList 来定义一个仪器组，该函数中第一个参数表示仪器组的标识名，第二个参数则为正要加入此仪器组的仪器标识名。由于仪器和仪器组的标识名采用相同的数据结构，因此它们都可以作为设备方便地用在总线控制的各种函数中。函数 Init 用于 GPIB 接口控制器本身的初始化，它将计算机接口卡本身的 GPIB 主地址强行设为 0 且不用扩展地址。函数的参数 iTmo 则表示限时退出(Timeout)的等级，即当数据传输的握手过程超过一定时限时，能够自动退出握手过程，用于防止异步通信中握手失败引起的死循环。上述命令用于系统控制的准备工作，并不真正操作 GPIB 总线。

函数 SendIFC 用于总线初始化，其实质工作是对 GPIB 总线发送单线接口消息 IFC。函数 SendRWLS 用于使其参数代表的仪器或仪器组中的所有仪器进入远地控制，并使设在面板上的本地/远地方式转换开关不起作用。这由发送单线消息 REN=1 后发送各仪器的听地址、再发送本地开关封锁通令 LLO 来实现。函数 DevClear 和函数 Trigger 分别用于对参数所表示的仪器或仪器组进行仪器本身的初始化操作和同步触发操作，其内部实现是由对参数变量中各仪器听寻址后再发专令 SDC 或 GET 完成的。

函数 Send 和函数 Recv 为自动测试系统提供了两个最重要的功能，即发送数据和接收数据。函数中第一个参数为有关发送(或接收)数据的仪器标识名，第二个参数为存放数据的缓冲区变量，第三个参数为欲发送或接收的字节数，最后一个参数为结束符(参数低字节)及结束方式(参数高字节)。当发送数据达到规定的字节数后，则根据结束符的存在与否(0 表示无结束符)决定是否再发一个结束字节。另外，结束方式为非 0 取值时，则在发送最后一字节的同时自动地伴随发送一个单线消息 EOI。与此相似，当一字节一字节地接收数据到规定的字节数时，或当接收的字节为规定的非 0 结束符，以及收到总线上 EOI 信号时，都将自动结束接收过程。

函数 TestSRQ 用于检查服务请求线 SRQ 的状态，当 SRQ 为真时返回值为 1，说明总线上存在某台设备要求控制器为其服务。然而，由于各 GPIB 设备共用一条 SRQ 线，因此还需要使用串行点名的方式找出具体请求服务的设备及其具体状态信息，函数 FindRqs 用于完成传统的串行点名仪器状态的功能，查询按照各仪器在仪器组中的位置依次进行，查得请求服务的仪器位置序号及其状态字节分别由参数变量 idev 和参数变量 stb 给出。

并行点名的过程分组态和执行两个阶段，前者可以通过调用组态函数 PollConfig 为其参数 dev 表示的仪器分配数据线和作用电平(由参数 iline 和 sense 分别定义)，也可通过并行点名组态取消函数 PPollUnConfig 将所设置的组态配置取消。并行点名的具体实现通过函数 PPollExec 完成，其内部作用是在 ATN=1 和 EOI=1 下直接读取数据线，从而同时取回各仪器(多至 8 台)的工作状态，并通过参数变量 ppb 上传。

函数 PassCtrl 用于将控制权转移到其参数 dev 定义的设备，转移后的计算机作为普通仪器设备，可以接收当前控制器发出的接口命令。函数 RequestCtrl 则为失去控制权的计算机提供请求重新获取控制权的一种方法。若在其指定时间(由参数 iTmo 定义)内能够接到当前控者的响应，则会返回成功标志。

为了说明面向总线接口的虚拟仪器功能，下面将组建一个最简单的自动测试系统，并以此为例展示各函数的使用方法。如图 3-8 所示，通过一台具有 GPIB 接口的数字万用表周期性地测量电网电压，以了解一天 24 小时中试验室电网电压的变化情况。这里数字万用表是通用仪器，希望它以 2000V 交流电压数字表的方式工作，且每分钟测量一次，再将测量结果形成计算机报告。

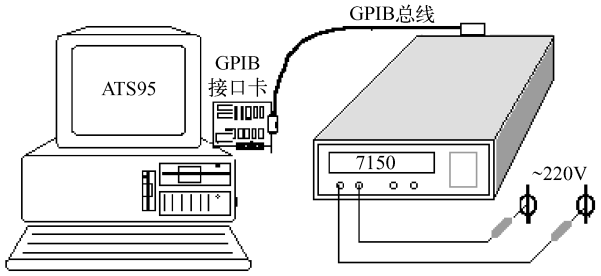



图 3-8 一个最简单的自动测试系统

由于测量连线不准，因此需要解决的问题在于如何通过计算机控制仪器的功能、量程和触发时刻，以及如何从仪器中读回测量结果和了解仪器的工作状态(是否发生故障)。程序清单 3-2 给出的代码说明如何使用 ATS98 面向总线接口的虚拟仪器功能函数来解决这个问题的方法。

**程序清单 3-2**

```
Dim Gp2 as New Lab49_ATS98.Cs4882, s7150() As Integer, GrpA() As Integer
Private Sub Command1_Click()
    GP2.Init 1, 13              '初始化计算机接口卡为系统控者，TMO 等级为 13.
    GP2.SetDevice s7150, 8, -1 '建立 GPIB 仪器 Solatron 7150，主地址为 8，无副地址
    GP2.SetDevList GrpA, s7150'建立 GPIB 仪器组 GrpA，仅包括一台仪器 s7150
    GP2.SendIFC                 '发送 GPIB 接口清除命令，初始化各仪器的 GPIB 接口
    GP2.DevClrList GrpA         '发送仪器清除命令，初始化 s7150
    Sleep 2000                  '仪器初始化需要一定的时间
    GP2.SendRWLS GrpA           '进入远地并封锁面板的本地／远地切换开关
    Timer1.Interval=60000       '建立定时器，确定定时时间为 60s
    Timer1.Enabled=1            '启动定时器
End Sub
Private Sub Timer1_Timer()      '定时服务程序
    GP2.Send s7150,"M1R4U7", 6, LF+GP_EOI  '发送程控命令，GP_EOI 为结束方式设置常数
    GP2.Trigger GrpA                       '触发仪器组中的各仪器，启动测量过程
    GP2.Recv s7150, sData$ , 15, CR        '从 s7150 中接收测量结果
    t$ =Now                                '取得当前时间
    AppendTextRow "c:\ats98\a1.dat", t$ +" "+sData$   '调用专用数据处理函数，存储数据
     If GP2.TestSRQ=1 Then                            '检查服务请求线 SRQ
        GP2.FineRqs GrpA, idev, stb                   '执行串行点名
        If (idev=1) and (stb=65) Then TreatErr   '出现服务请求，则转入相应处理
程序
    End If
End Sub
```

仪器的程控命令通常需要参考其用户手册，如发送的程控命令串“M1R4U7”是根据数字表 7150 的规定，其中 M1 表示交流电压测量方式，R4 表示 0~2000V 的测量范围，U7 表示读数时的结束符为 ASCII 码 CR(十进制数 13)。本例仪器组中只有一台 s7150，由其手册规定的状态字节的第 6 位为请求服务标志位，第 0 位为故障标志位。这样当状态字节为 65 时，表示 s7150 发生故障，应转入相应的故障处理程序。

### 3.3.3　基于 GPIB 接口的微机光谱分析系统

计算机技术的快速发展，使得前些年生产的一些贵重仪器设备的数据分析和存储方式难以维持。这里使用光谱分析仪分析光通信产品时就遇到了这种情况，原先专门配套的软盘已经买不到了，配有专用接口的计算机也已更新换代，而光谱分析的主机功能却依然十分可贵。由于这些早年生产的仪器通常都配备了 GPIB 通用接口(通常标为 IEEE488)，而且支持这种总线标准的通用控制卡及其软件工具虽不断更新却始终存在，因此下面分析光纤连接器的实际需要，介绍如何利用在 PC 中加装 PCI-GPIB 接口板来存取光谱分析仪器的测试数据，并在微机上实现相应的分析功能，显然具有相当普遍的实用意义。

光谱分析仪是研究光通信中光学插接件、光源、光隔离器和光放大器等高科技产品特性的重要工具。光通信产品的开发和研制要求较高测量精度和稳定工作的光学测量仪器，并且需对所测数据进行实时存储和分析。在如图 3-9 所示的微机控制光纤光谱分析系统中，光源是由激光器 DFB 产生的，通过光纤、被测组件和光探头，连接到光谱分析仪。通过分析光谱线的强度，谱线分布来确定光纤产品中的光学特性。通用光谱分析仪虽具备以上基本功能，但数据分析结果只能通过打印机输出，或以专用软盘保存，因此限制了数据的应用范围。为此通过配置于 PC 的 PCI-GPIB 接口卡将微机和光谱分析仪有机地联系起来，可以实现对光谱分析仪的远程控制、数据的实时存储，并利用通用的 VB 语言建立示波功能，实现数据曲线的回放和比较，从而为分析光纤连接的状态提供了可靠的依据。

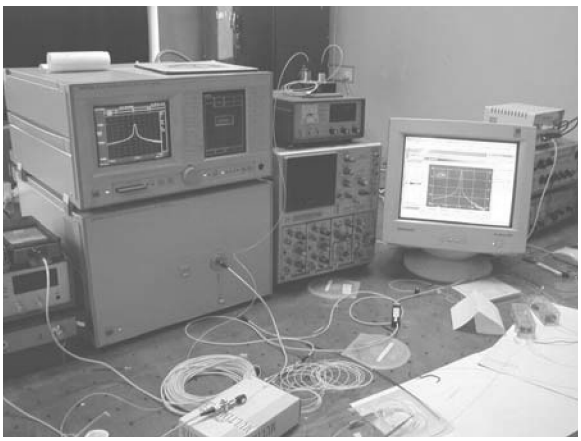


图 3-9　光纤连接器的光谱分析系统

虚拟仪器的示波功能可以直接取代光谱分析仪的显示。如图 3-10 所示，横轴为光谱的波长[单位是 μm(微米)]，纵轴为光强[单位是 dB(分贝)]，波形曲线则直接表示光谱分析扫频测试的结果。显示数据通常是通过单击菜单中的运行选项，调用 GPIB 接口函数得到的。

得到的数据首先显示在窗口左侧的列表框中，然后在右侧的示波屏中形成曲线。列表框中下面的按钮可以用来改变列表框的宽度，只在需要时才显示全部数据内容。同时可以通过单击菜单中文件选项下的子项将数据转储成文件，并在以后任何时候通过其他选项调回文件进行回放和比较分析。回放时，可通过窗口面板下部的按钮分别控制曲线在水平和垂直两个方向的幅度增益和位置移动。A 和 B 两个通道分别用两种颜色绘出曲线，并在坐标网格的两侧分别标度。两个通道的标尺采用同一组按钮进行调整，通过单击刻度外侧部位进行切换，图中显示屏下部所示的 ChB 标记表明当前正在操作的是 B 通道。而显示的两条曲线则分别来自上传的实时数据和先前转储的数据文件，并表现出两个不同条件下的光纤通信器件的光谱特性。另外，这个示波窗口还专门设计了一个独特的红色游标，当用鼠标单击或拖动它时，就将其指示的当前水平和垂直位置的准确数据值用数字形式在屏幕上部显示出来。这样的示波功能既用曲线说明了趋势，又用数字表现了精度。示波功能被建立在一个 VB 的类对象结构中，只要简单地调用几条所属函数就可将全部示波功能体现在指定的图形控件对象下，因此具有普遍的意义。

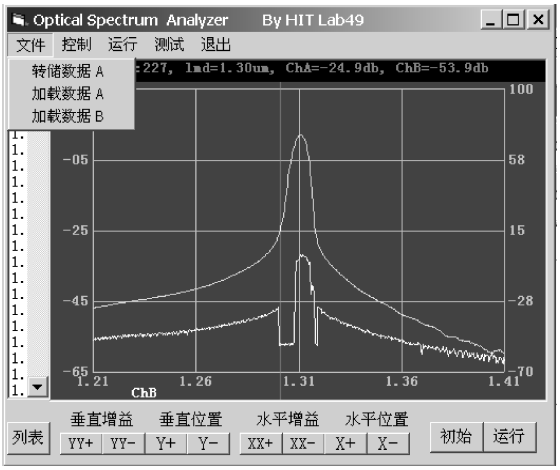



图 3-10 示波功能窗口

GPIB 接口功能的实现简明易懂，只要了解 GPIB 总线技术的基本概念就可编写测试程序。现以程序清单 3-3 所示的两段核心程序为基础，说明组建此类测试系统的方法。在此测试程序中，首先为系统中的仪器定义标识名，通过函数 SetDevice 完成。这个函数的参数表明仪器的主地址为 11，没有副地址(-1 表示非扩展地址工作方式)。函数 Init 用于接口控制器本身的初始化，它将接口卡本身的 GPIB 主地址设为 0，而且不用扩展地址，参数 12 表示 TMO(Timeout)的等级，用于防止异步通信中握手失败时引起死循环。函数 SendIFC 和函数 SendRWLS 的调用使总线初始化后仪器进入远地控制。

**程序清单 3-3**

```
Dim GP2 As new Lab49_ATS98.Cs4882Ni, s$ , sDev1() as Integer
GP2.Init 12: GP2.SetDevice sDev1, 11, -1        '初始化，设置仪器(主地址 11)
GP2.SendIFC: GP2.SendRWLS sDev1                 '远地
GP2.DevClear sDev1                              '仪器清除
GP2.Send sDev1, "STA?", 4, Nlend                '查询扫频起始波长
```

```
GP2.Recv sDev1, s$, Nlend: w1#=Val(Mid(s$, 5))
GP2.Send sDev1, "STO?", 4, Nlend                    '查询扫频停止波长
GP2.Recv sDev1, s$, 15, Nlend: w2#=Val(Mid(s$, 5)): dw#=(w2#-w1#) /500
GP2.Send sDev1, "DMA?", 4, Nlend                    '要求读取缓冲区内容
For k% =0 To nData
    GP2.Recv sDev1, s$, 15, Nlend: yData1(k%)=Val(s$)
Next k%
```

发送和接收数据的函数 Send 和 Recv 分别用来发送程控命令和读取数据。这里发送的程控命令串“STA?”和“STO?”是要求光谱分析仪提供扫频的起始和终止波长，随后通过接收函数读回其值。而命令“DMA?”则要求读取放在缓冲区中的数据，且根据光谱分析仪的设置，规定的数据长度为 nData=501。

实际系统采用的 GPIB 接口卡能够采用市场上任何可以采用的产品，如采用 NI 公司的 PCI-GPIB 接口卡(外形见图 3-11)，自然可以得到相应的配套动态连接库文件 Gpib-32. dll，也可得到用于声明这些函数和定义有关常数和变量的 VB 文件(Vbib-32. bas 和 Niglobal. bas)。为了更加方便地建立和调试所需要的应用程序，可以进一步将这些接口函数包装成 VB 类函数形式，从而使应用程序能够用统一的 VB 类函数通过不同接口卡来控制和处理 GPIB 设备。程序清单 3-4 给出了一些常用类函数的具体建立过程，以使读者了解这种类函数包装的实现方法。

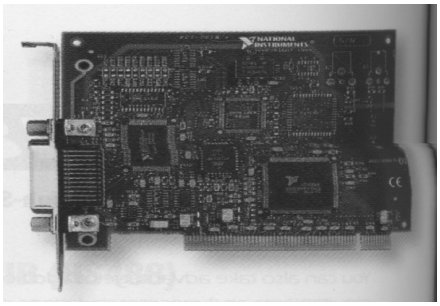


图 3-11　NI 公司的 PCI-GPIB 接口卡

**程序清单 3-4**

```
'设备及其设备表的设置
Function SetDev(dev() As Integer, madr As Integer, sadr As Integer)  '设置设备变量
    ReDim dev(1): dev(0)=madr+(sadr Or &H60) * 256: dev(1)=-1
End Function
Function SetDevList(devs() As Integer, dev() As Integer)  '设置设备变量列表
    n% =UBound(devs): ReDim Preserve devs(n% +1)
    devs(n%)=dev(0): devs(n% +1)=-1
End Function
Function SetTmo(itmo As Integer) As Integer                '设置超时处理的等级
    If (GPIBglobalsRegistered=0) Then Call RegisterGPIBGlobals
    ibtmo 0, itmo
End Function
'接口系统操作
Function SendIFC()                                         '接口清除
    Call SendIFC32(0): Call copy_ibvars: SendIFC=iberr
End Function
Function SetRWLS(devs As String)                           '进入远地并封锁本地开关
    Call SetRWLS32(0, devs(0))
```

```
    Call copy_ibvars: SetRWLS = iberr
End Function
Function EnableRemote(devs As String)                    '进入远地
    Call EnableRemote32(0, devs(0))
    Call copy_ibvars: EnableRemote = iberr
End Function
Function EnableLocal(devs As String) As Integer          '返回本地
    Call EnableLocal32(0, devs(0))
    Call copy_ibvars: EnableLocal = iberr
End Function
Function SendLLO() As Integer                            '本地封锁
    Call SendLLO32(0)
    Call copy_ibvars: SendLLO = iberr
End Function
Function DevClear(devs() As Integer)                     '仪器清除和触发
    If UBound(devs) > 1 Then
        Call DevClearList32(0, devs(0))
    Else
        Call DevClear32(0, devs(0))
    End If
    Call copy_ibvars: DevClear = iberr
End Function
Function Trigger(devs() As Integer)                      '仪器清除和触发
    If UBound(devs) > 2 Then
        Call TriggerList32(0, devs(0))
    Else
        Call Trigger32(0, devs(0))
    End If
    Call copy_ibvars: Trigger = iberr
End Function
'发送和接收数据
Function Send(devs() As Integer, buf() As Byte, n as Integer, ByVal term As In-
teger) as integer
    Dim term1 as integer
    If (term And &H100 > 0) Then
        If term And &HFF Then term1 = &H1 Else term1 = &H2
    End If
    If UBound(devs) > 2 Then
        Call SendList32(0, devs(0), ByVal buf, n, term1)
    Else
        Call Send32(0, devs(0), ByVal buf, n, term1)
    End If
    Call copy_ibvars: Send = iberr
```

```
End Function
Function Recv(devs() As Integer, buf() As Byte, n as Integer, ByVal term As In-
teger) As Integer
    Call Receive32(0, devs(0), buf, n, term and &H100)
    Call copy_ibvars: Recv=iberr
End Function
```

这里，类函数的实现无非就是按照前面介绍 ATS98 总线控制类函数的格式要求，把支持其他类型接口板的驱动功能函数包装一下而已。当然，所用的参数也须进行适当的转换。值得注意的是 ATS98 对仪器设备与仪器设备组的标识方式相同，均为整型数组，因此用户调用操作函数时不必区分这两种情况，可由其库函数内部自行区别处理。具体标识设备数组的结尾标志为数值-1，其基本元素的低字节表示设备的主地址，而高字节表示设备的副地址(且其第 5 位和第 6 位均为 1)，这样设备数组中的每个整数就完整表达了设备的主地址和副地址。另外，由于 NI 公司的 PCI-GPIB 接口卡只有无 EOI(&H0)、回车符伴随 EOI(&H1)和末尾符伴随 EOI(&H2)等几种结束数据发送的方式，以及用接收结尾符伴随 EOI(&H100)来判断结束数据接收过程的方式，因此与原先类函数的衔接也需要进行一些简化处理。

## 3.4　IEEE488.2 和 SCPI 规范要点

随着 GPIB 在世界范围的普及，1989 年 IEEE 发布的 IEEE488.1 和 IEEE488.2 两个文件及同期发布的 SCPI 文件，进一步促进了自动测试系统的标准化进程，本节将以有限的篇幅对此情况予以简述。

### 3.4.1　ANSI/IEEE488.2 规范的要点介绍

ANSI/IEEE488.2 的内容分为一般概念、仪器和控者 3 个部分，其中一般概念是指规范的适用范围、目的、文字约定、参考文献、系统术语、消息传递的层次和方式等内容；器件部分规定了仪器对选择接口功能子集的要求、消息交换的控制协议、程控命令和状态报告的语法格式、公用命令的设置、地址分配功能及仪器文档等要求；控制器部分给出了详细的控制序列要求和相应的协议要求。这里仅对测试工作者最为关心的一些问题予以说明。

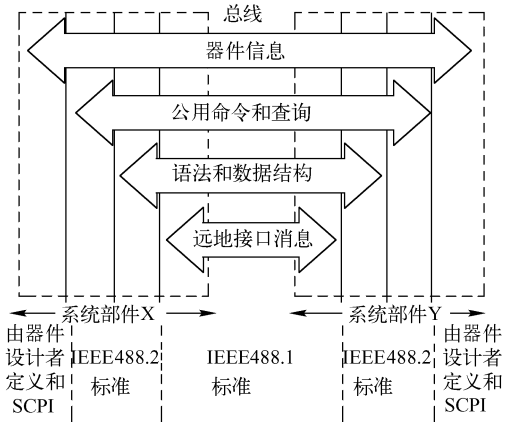



图 3-12　消息交换控制接口模块的功能关系

#### 1. 消息交换控制协议

IEEE488.2 引入了通信协议的概念。控者和被控设备之间的仪器消息交换被归纳为程序消息和响应消息，程序消息又分普通命令和查询命令(也称查询，以问号结尾)两种。前者使接收命令的仪器产生某种操作；后者则不但进行操作，还要返回响应消息。这里程序消息和响应消息都是作为仪器消息的方式出现的，即 ATN = 0 下的多线消息。如

图 3-12 所示，消息的交换分为 A、B、C 和 D 类 4 个层次。其中，A 层为接口功能层，用于接口电路的实际连接，其基础就是前面讨论的 IEEE488.1；B 层和 C 层分别为消息传递层和公用功能层，具体实现 IEEE488.2 标准定义的语法和数据结构，以及公用命令和查询的具体内容；D 层为器件消息功能层，是指那些以仪器本身操作信息为内容的信息传递。

消息交换接口的逻辑模式如图 3-13 所示，可以根据这个模式直接设计可程控仪器的接口管理方式。这里，I/O 控制单元根据 IEEE488.1 总线上的多线消息可以产生发往输入缓存区的数据有效字节 DAB、传输结束信息 END 和触发命令 GET。其中触发命令也可以直接产生仪器的本地触发信号 get，通过触发控制单元直接操作仪器功能，从而减少语法分析器中软件处理命令过程引起的延迟。同理，为了让仪器能够立即响应仪器清除命令 DCL 和 SDC，在 I/O 控制单元产生的仪器清除信息 dcas 也不存入输入缓存器。输入缓存器要求采用先进先出的数据结构，根据语法分析器单元的分析能力和执行控制单元的处理速度，依次解析输入缓存器中的内容，并根据解析的要素执行不同的控制动作，最终将处理后得到的有关仪器信息送给器件功能。另一方面，I/O 控制单元根据状态报告单元发来的状态字节 STB 及 reqt、reqf、ist 信号可以响应串行点名查询和并行点名查询，向总线发出数据有效字节 DAB 和 END 消息。

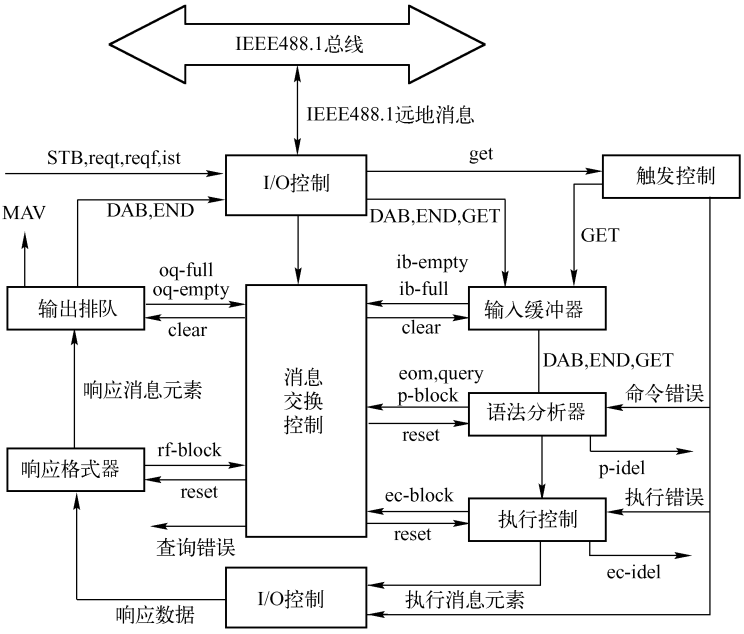



图 3-13 器件消息交换接口的逻辑模式

为了简明起见，图中没有标出数据请求 brq 和数据有效 bav 等握手信号的作用。实际要求 bav 为真、输入缓存器满标志 ib-full 为假时 I/O 控制单元通过受者握手功能将总线上的 DAB 数据字节和伴随的 END 消息依次读入输入缓存器，输入缓存器的空、满状态可以通过 ib_empty 和 ib_full 通知消息交换控制，而消息交换控制也可通过 clear 信号来清除输入缓存器。语法分析器从输入缓存器接收的 DAB 数据字节、END 和 GET 消息被分离成标准定义的

各种语法元素进行分析，将有效的语法元素转换成内部表示的消息送至执行控制单元，也将无效语法的部分作为错误命令交给状态报告单元处理。当语法分析器识别到消息结束和要求查询的语法元素后产生 eom 消息和 query 消息，而当已经分析出的消息元素正在等待执行时则应通过建立的阻塞消息(p-block)及时通知消息交换控制器。每当仪器清除 dcas 和上电信号 Pon 发生时，语法分析器和执行控制单元都要复位，并将下一个接收到的数据作为程序消息的首字节。当执行控制单元收到足以开始一个器件操作的消息后就将这个可执行的消息元素送至仪器设备本身的器件功能，执行过程发生错误时则应向状态报告单元报告。当执行控制单元等待响应格式器和器件功能单元处理先前的查询请求时，要通过信号 ec-block 为真的信号通知执行控制单元。通过语法分析得到的询问请求元素以可执行消息的方式送至器件后，由此产生的响应数据经响应格式器被处理成响应消息元素。然后作为 DAB 并与伴随的 END 消息一起放在输出队列中暂存，直至当字节请求信号 brq 为真，输出队列空标志 q_empty 为假时，I/O 控制单元通过源握手功能将它们发到总线。同时只要输出队列中有数据，向状态报告单元发出的消息有效信号 MAV 就为真。

消息交换控制协议为采用确切的消息语法结构提供了可靠的基础，IEEE488.2 在语法上要求消息接收时比其发送时更灵活，从而体现出器件之间能够“严格地讲、宽容地听”。

### 2. 程控命令集的设计和公用命令的使用

鉴于某些程控命令对各种仪器都有用，IEEE488.2 定义了公用命令，以便用户设计仪器程控命令时选用。其中 13 条为仪器必备的命令，还有 21 条为任选的命令。为了与专用命令相区分，公用命令以 * 号开头。表 3-8 给出了这些命令的简要说明。

**表 3-8　IEEE488.2 通用命令表**

| | | | | |
|---|---|---|---|---|
| 内部操作同步 | * RST | 复位 | 必备 | 执行器件复位 |
| | * TST? | 自检查询 | 必备 | 内部自检，返回是否正确 |
| | * CAL? | 校准查询 | 任选 | 指示器件完成自校的响应 |
| | * LRN? | 识别器件设置查询 | 任选 | 获取响应消息单元组成的序列 |
| | * OPC，* OPC? | 标示操作完成 | 必备 | 置位 OPC；事件输出队列放“1” |
| | * WAI | 等待操作完成 | 必备 | 停止执行命令或查询，直至仪器操作完成 |
| 状态和事件 | * CLS | 上电清除 | 必备 | 包括状态数据结构，进入空闲状态 |
| | * PSC，* PSC? | 上电状态清除 | 必备 | 包括服务请求、标准事件、并行点名等使能 |
| | * SRE，* SRE? | 服务请求使能 | 必备 | 设置和查询服务请求使能寄存器内容 |
| | * STB? | 读状态字节 | 必备 | 查询状态字节，包括 MSS(主综合状态)位 |
| | * ESE，* ESE? | 标准事件使能 | 必备 | 设置和查询标准事件状态使能寄存器内容 |
| | * ESR? | 读标准事件 | 必备 | 查询标准事件状态寄存器当前内容 |
| 控者 | * PCB | 传递控制权 | 非 C0 必备 | 发给欲接受控制权的设备 |
| 并行点名 | * IST? | 并行点名状态查询 | PP1 必选 | 读 ist 当前状态 |
| | * PRE，PRE? | 并行点名使能 | PP1 必选 | 设置和查询并行点名使能寄存器内容 |

| | | | | |
|---|---|---|---|---|
| 触发 | * TRG | 触发动作 | DT1 任选 | 实现群触发 GET 功能 |
| | * DDT，* DDT? | 定义触发 | DT1 任选 | 定义和查询一个群触发时执行的序列 |

续表

| | | | | |
|---|---|---|---|---|
| 系统数据 | * IDN? | 识别查询 | 必备 | 包括生产厂家、型号、序列号、固件版本等 |
| | * OPT? | 选项识别查询 | 任选 | 用于识别系统接口中的可读选项 |
| | * PUD，* PUD? | 保护用户数据 | 任选 | 包括校准日期、使用时间、环境、编号等 |
| | * RDT，* RDT? | 源说明传送 | 任选 | 存储和读取一条资源描述 |
| 组态 | * AAD | 分配地址 | 任选 | 应结合地址设置协议 |
| | * DLF | 取消听功能 | 任选 | 取消地址配置仪器的听功能，以便控者搜索 |
| 存储 | * RCL | 恢复命令 | 任选 | 恢复本地存储器中的器件设置 |
| | * SAV | 存储命令 | 任选 | 将当前设置保存到本地存储器中 |
| 宏命令 | * DMC | 宏定义 | 任选 | 允许为宏标记指定若干程序信息单元的序列 |
| | * EMC，EMC? | 宏使能 | 任选 | 设置和查询宏扩展内容 |
| | * PMC | 宏清除 | 任选 | 删除所有 * DMC 定义的宏 |
| | * GMC?，* LMC? | 宏内容获取、识别 | 任选 | 从仪器查询宏定义和宏记号 |

### 3. 用于串行点名的通用状态消息

IEEE488.2 建立了一个分层的状态消息模型，如图 3-14 所示。

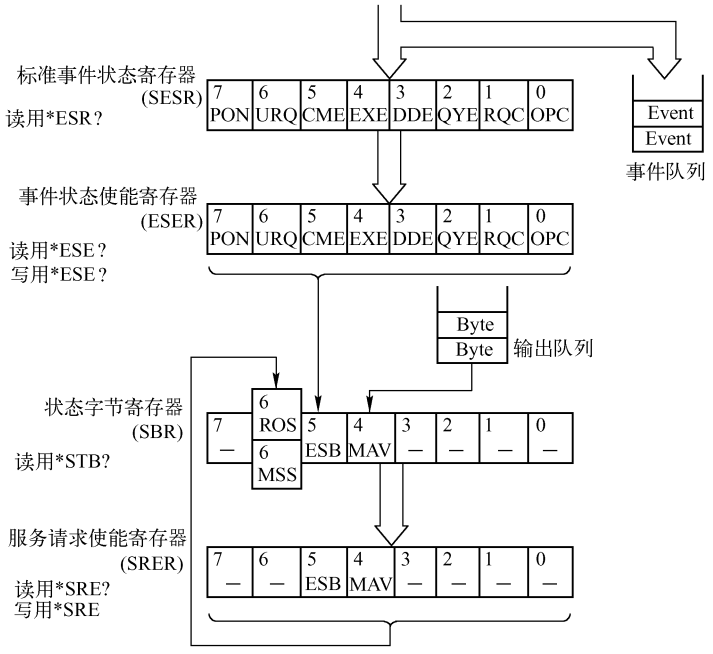



图 3-14　状态寄存器的设置和操作

当诸如上电(PON)、用户请求(URQ)、命令出错(CME)、执行出错(EXE)、设备出错(DDE)、队列出错(QYE)、请求控制(RQC)、操作完成(OPC)等标准事件发生时，就会将标准事件状态寄存器(SESR)中的相应状态位置 1，并依据通用命令 * ESR? 将整字节作为其响应输出。

如果在事件状态使能寄存器(ESER)中的某控制位置 1，则会在其对应事件发生时将状态字节寄存器(SBR)中的 ESB 位置 1。这里，ESER 可用通用命令 * ESE 和 * ESE? 分别进行设置和查询。状态字节寄存器中的另一个状态位 MAV 置 1 表示另有状态信息已经写入输出队列，故可通过正常的多线仪器消息依次读出。

如果服务请求使能寄存器(SRER)中对应的允许状态位置 1，则应产生相应的服务请求操作，并将状态字节寄存器中的 RQS 位置 1。这里 SRER 寄存器的读/写可通过执行通用命令 SRE? 和 SRE 分别实现。

#### 4. 系统的同步技术

IEEE488.2 的信息交换采用数据缓冲区，程控命令可以连续地传送到仪器中缓存，然后再逐条执行，这样就产生一个确保系统控制器操作程序与仪器实际动作之间同步关系的问题，也就是说，可能先前发给甲设备的命令还未执行，测试条件还未建立起来，随后发给乙仪器的测试要求已经开始，从而由于同步问题造成测试错误。

IEEE488.2 常使用 3 个公用命令来处理这类同步问题。其中公用命令 *WAI 的执行要求仪器暂时处于等待状态，直至完成先前所有已经进行的命令操作之后方能结束，这样就能确保发出 *WAI 命令之后新的命令操作与先前所应执行的那些操作之间的先后关系。公用命令 *OPC 则采用另一种方法，即要求仪器在其完成当前操作任务后就将标准事件状态寄存器中的 OPC 位置 1，并发出服务请求，这样总线控制器收到服务请求后确认仪器已经完成了预设任务，再去安排下一步动作。公用命令 *OPC? 是一条查询命令，它要求仪器在完成一项命令操作任务后将一个 ASCII 字符“1”写入输出队列中，这样通过读取仪器输出响应队列的内容就能判断仪器内部是否已经执行到位。

#### 5. 系统的自动地址分配

尽管大多数 GP-IB 仪器采用后面板上的地址编码开关，或通过前面板菜单式选项手动建立，但有时由于环境限制，也有可能需要采用远地自动分配地址的方式。IEEE488.2 提出了一种自动组态的规程，可由控者远地识别总线上每个具有自动组态功能的器件，并为其动态安排一个地址。

### 3.4.2 可程控仪器标准命令 SCPI 规范的要点说明

SCPI(Standard Commands for Programmable Instruments)是一个用于选择可程控仪器命令的设计指南。全文分为引言、语法和格式、命令说明、数据交换格式等 4 个主要部分。本质上讲，SCPI 是一种新的描述仪器控制的命令语言，从远地程控观点试图促进仪器之间的一致性。

作为程控命令组分类和实现兼容性的一种方法，SCPI 提出了可程控仪器的通用模型。模型由若干分层的功能命令集组成，为仪器程控提供了一致性的编程环境。这种一致性表现在纵向和横向两个方面：纵向一致性意指相同类型、不同时期生产的仪器具有相同的控制方法(如型号不同的示波器具有相同的时基、触发和增益设置功能，可用同样的程控命令加以控制)；横向一致性则指不同类型的仪器可以使用相同命令完成相同的具体功能(如示波器和函数发生器的触发功能相同，使用的触发命令也应是同样的)。图 3-15 给出了这种可程控仪器通用模型的基本结构形式。

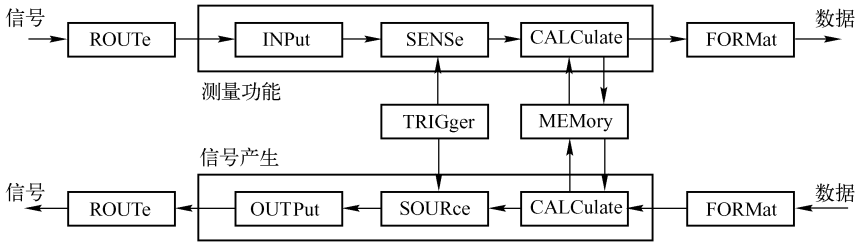



图 3-15　可程控仪器通用模型

这个模型体现了 SCPI 分类仪器功能的思想和方法，仪器主要是由测量和信号源两大功能组成，前者把物理信号转变为数据，后者则把数据转变为物理信号。其主要子系统为输入通道(INPut)、检测(SENSe)、计算(CALCulate)、触发(TRIGger)、存储(MEMory)、格式化(FORMat)、输出通道(OUTPut)、源(SOURce)，以及测量(MEASure)、显示(DISPlay)等各种不同功能。而对具体的子系统，则又有更细的描述模型。如对测量仪器来说，可以分为各种不同的测量内容，如电压(VOLTage)、电流(CURRent)、频率(FREQuency)等，还可完成信号的调整、转换、计算等附加工作。这样根据通用仪器模型，组建自动测试系统的编程人员就不需要十分费力地记忆所用仪器的操作命令，而只要根据要求首先找到 SCPI 系统的功能块，然后沿着树形结构的命令分支，逐次查找适合自己要求的操作命令。如欲进行周期电压测量时，可以从测量功能开始，再查测量内容，选择电压后再看类型，最终得到的操作命令有如 MEASure:VOLTage:PERiod 的典型形式。

SCPI 的语法形式是以关键字为单元，以冒号作为连接词，按树形层次结构组成描述仪器功能的复合词。这里组成关键字的原英文单词字母数超过 4 时用前 4 个字母作为关键字；如果关键字以元音字母结尾，则去掉这个元音字母(如 Power 取关键字 POW)；对多个单词组成的表达方式，可使用每个单词中的第一个字母和最后一个单词的全部字母作为关键字(如 Direct Sequence 取关键字 DSEQ)。为了便于阅读，可采用一种长型的助记符形式，即前面大写字母表示关键字部分，后面用小写字母表示其余帮助记忆的部分(如 VOLTage、TRIGger 等)。SCPI 的参数格式分数值参数、布尔参数、开关参数等几种形式，其中数值参数使用 ASCII 码表达，布尔参数取 0、1 或 ON、OFF，而开关参数则根据需要可以有多个不同的取值。

实际 SCPI 规范给出 IEEE488.2 的 13 个必备命令和 400 多个选择命令供仪器设计者选用，对仪器操作的编程标准化起到了很好的作用，而且 SCPI 允许不断地补充和完善，需要认真学习和利用。

### 3.4.3　可程控仪器编程控制的示例

目前，国际上大多数正规仪器厂家开发的可程控仪器都已利用 IEEE488.2 和 SCPI 规范的设计思想来配置自己的程控命令，尽管也有一些具体的仪器功能采用厂家自行扩展的命令，但依据上述两种规范的思路来参阅编程手册，仍然能够体会到其标准一致性对编程控制的明显好处，下面将以泰克公司的通用示波器 TDS220 为例来说明此类设备的具体程控方法。

如图 3-16 所示，使用计算机程控这种示波器时可首先通过面板设置，建立其 GPIB 地址，然后就可使用该地址建立如下一段初始化程序，并通过返回的设备识别字来确认 GPIB 系统的连接状态。

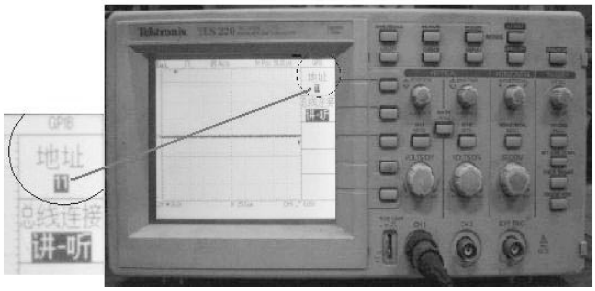


图 3-16 泰克示波器 TDS220 的程控设置

```
'TEK-TDS220 型示波器的控制程序
Dim Gp2 As New Lab49_ATS98.Cs4882Ni
Dim Dev() As Integer      '设备变量
Dim trmS As Integer, trmR As Integer
Dim s$, st%
s$ =String(80, 0): trmS =&H10A: trmR =&H100
 Gp2.SetTmo 9              '建立超时处理的等级
 Gp2.SetDev Dev, 11, -1    '建立仪器(主地址为 11)
 Gp2.SendIFC               '发送接口清除,实际操作总线的首条命令
 Gp2.EnableRemote Dev      '发送远地使能,从此仪器操作由面板转为总线,应显示标志
 st% =RunCmd("*CLS; *WAI", 0, s$)      '发送命令,清除仪器现行的状态标志
 st% =RunCmd("*IDN?", 10, s$)  '查询仪器的识别字串,并可判断接口收发功能是否正常
```

这里,函数 RunCmd 是考虑采用 IEEE488.2 和 SCPI 规范后,大多数仪器的程控功能都可以通过仪器消息的传送来实现,且可以归纳为发送命令-延迟等待-接受响应的处理模式,因此可以建立这种通用命令处理的功能函数。具体实现方法如下,其 3 个参数分别用于表达控制命令、延迟时间(ms)和响应结果。

```
Function RunCmd(sCmd As String, twait As Integer, sRsp As String) As Integer
'运行接口命令
    Dim st%
    st% =Gp2.Send(Dev, sCmd, Len(sCmd), trmS)          '发送命令
    If twait > 0 Then
        Sleep twait                                    '主动延迟
        st% =Gp2.Recv(Dev, sRsp, Len(sRsp), trmR)      '接受响应
        If sRsp="" Then st% =-2
    End If
    RunCmd=st%
End Function
```

示波器的程控功能主要包括面板参数设置、数据采集触发方式和波形数据读取等几个方面。下列程序示例用于面板参数的设置,首先选通信号通道 1 和 2,然后设置水平和垂直通道,参数 SCAL 用于定义每格表示的时间和电压,POS 用于定义零点位置,AC 和 DC 分别表示交流和直流耦合方式。参数 DAT 下的子参数 START 和 STOP 分别用于设置数据采集的始末值,子参数 WID 和 ENC 则分别用于设置波形数据的点数和波形数据的格式,RPB 表示一种正整数的二进制格式。为了确保设置这些参数的命令能够切实执行,这段程序最后使用了

*WAI，要求仪器等待前步命令操作成功执行后再进行后步命令的操作。

```
s$ =String(20, 0)
st% =RunCmd("SEL:CH1 ON; CH2 ON", 0, s$)                              '双踪测试
st% =RunCmd("HOR:MAIN:SCAL " & CStr(xx /10), 0, s$)                   '设参数 xx
st% =RunCmd("CH1:SCAL " & CStr(yy1) & "; POS 0; COUP DC", 0, s$)      '设参数 yy1
st% =RunCmd("CH2:SCAL " & CStr(yy2) & "; POS 0; COUP AC", 0, s$)      '设参数 yy2
st% =RunCmd("DAT:STAR 1; STOP 2500", 0, s$)                           '采数始末值
st% =RunCmd("DAT:WID 1; ENC RPB", 0, s$)                              '编码数
st% =RunCmd("*WAI", 0, s$)                                            '等待完成
```

示波器数据采集的方式有两种。当命令 ACQ:STOPA 取值为 RUNST 时，运行和停止均由用户在前面板上操作 Run/Stop 按钮决定，相当于“连续”方式，且命令 ACQ:STATE 等价于 Run/Stop 按钮；而当参数 ACQ:STOPA 取值 SEQ 时，数采的波形数据达到要求的数量后就会自动停止数采过程，相当于“单次”状态。

由于示波器内部量程的分挡机制，程控设置的参数并不一定都能被实际设备完全接受。而且，用户有时也希望能够以手动方式进行一些参数的调整，因此通常需要计算机读取被测波形数据的同时，也要读回仪器当前实际配置的一些关键参数，以便完成数据的定量计算。这些关键参数包括：采样时间，以及各通道的增益、零点和偏移量，它们可从分别执行查询命令 WFMP:XIN?、WFMP:CHx:YMUL?、WFMP:CHx:YZERO? 和 WFMP:CHx:YOFF? 后的响应数据中得到。另外，读回的响应字串一般需要根据仪器特定的格式去掉说明部分，方能最终获取有用的参数数据，这里是由函数 TakePmVal 完成的。下面节选的一段 VB 程序代码用来说明从示波器读取波形数据的实际过程。

```
s$ =String(40, 0): s1$ =String(2500, 0)
st% =RunCmd("ACQ:STOPA SEQ; *WAI ", 0, s$)                            '设数采单次方式
ts =TakePmVal("WFMP:XIN?")                                            '读取采样时间
n1% =TakePmVal("DAT:STAR?"): n2% =TakePmVal("DAT:STOP?")              '读取起始和停止索引
n% =n2% -n1% +1: If n% < 2 Then GoTo l2                               '计算数据长度
ReDim t(n%), d1(n%), d2(n%)
For k% =0 To n% -1: t(k%)=ts * k% : Next k%
For i% =1 To 2
    st% =RunCmd("DAT:SOU CH" & Format(i% , "0"), 0, s$)               '设数据源
    If RunCmd("CURV?", 100, s1$) < 0 Then n% =-1: GoTo l2             '读数据
    st% =RunCmd("*WAI", 0, s$)                                        '同步等待
    s$ ="WFMP:CH" & Format(i% , "0")                                  '设数据源
    yy#=TakePmVal(s$ & ":YMUL?")                                      '读 Y 轴参数：v/格
    y1#=TakePmVal(s$ & ":YOFF?")                                      '读 Y 轴参数：偏移格数
    y0#=TakePmVal(s$ & ":YZERO?")                                     '读 Y 轴参数：偏移电平
    delta% =Log(n%)+3+1: bData=StrConv(s1$, vbFromUnicode)            '格式转换
    For k% =0 To n% -1
        If i% =1 Then
            d1(k%)=(bData(k% +delta%)-y1#) * yy#-y0#
        Else
            d2(k%)=(bData(k% +delta%)-y1#) * yy#-y0#
        End If
    Next k%
```

```
Next i%
st% =RunCmd("ACQ:STOPA RUNST; STATE RUN", 0, s$)      '设连续数采方式, 开始运行
l2: …

Function TakePmVal(sCmd As String) As Variant          '提取格式参数
    Dim i1%, i2%, s$, st%
    s$ =String(40, 0)
    st% =RunCmd(sCmd, 50, s$): i1% =InStr(1, s$, ""): i2% =InStr(1, s$, Chr(10))
    TakePmVal =Val(Mid(s$, i1% +1, i2% -i1% -1))
End Function
```

由于数字示波器对所读取波形数据也能进行一些数据处理, 因此可以通过 SCPI 的一些标准命令直接得到诸如均值、峰-峰值、频率、前沿时间等被测波形的特征量, 如:

```
st% =RunCmd("MEASU:IMM:SOU CH0; TYPE FREQ, 0, s$)      '试图读取波形频率
s$ =TakePmVal("MEASU:IMM:VAL?")
```

由上可见, 尽管采用 SCPI 规范的 TDS220 通用示波器的程控命令有时显得有些烦琐和复杂, 但由于其配置全面和层次清晰, 可将掌握的程控技能举一反三地用于其他可程控仪器的开发工作中。这样, 即使某些可程控仪器的编程手册没有提供足够的例程, 用户也能根据以往的经验完成 GPIB 仪器系统程控代码的编写和调试。同时, 采用 SCPI 规范的仪器程控命令并不局限于 GPIB, 也同样适用于为该仪器配置的其他接口。

## 本 章 小 结

GPIB 通用接口总线提供了一种 8 位并行数据传输的异步通信方式, 为在全球范围不同厂家的台式仪器之间组建自动测试系统提供了一种十分有效和可靠的方法。有关规范用状态变迁图的方式完整地描述了 GPIB 接口的功能要素, 基于微处理器的专用接口套片则为实际开发 GPIB 接口提供了必要的硬件支持, 在此基础上实现仪器控制的程序固件便成为可程控仪器开发的一项关键工作。同时, 以通用 PC 为核心的总线控制器作为 GPIB 系统中操作各测试仪器的主要工具, 掌握其应用程序软件的开发方法具有更为实际的意义。另外, IEEE488.2 和 SCPI 是 GPIB 和自动测试系统技术进一步发展的产物, 了解并掌握其核心思想将有助于可程控仪器的标准化设计, 也有助于实际自动测试系统的软件开发。

## 参 考 文 献


[1] IEEE std 488.1-1987, IEEE standard digital interface programmable instrumentation. (ANSI)

[2] IEEE std 488.2-1992, IEEE standard codes, formats, protocol, and common commands for use with IEEE std 488.1 digital interface programmable instrumentation. (ANSI)

[3] Standard Commands for Programmable Instruments(SCPI), Version 1999.0.SCPI.Consortium.

[4] 张礼勇, 程玉润. IEC625 通用接口及其应用.计量出版社, 1985.

[5] 孙续.自动测试系统与可程控仪器. 电子工业出版社, 1994.

[6] 恽纪昌. 自动测试通用接口系统. 机械工业出版社, 1990.

[7] Intel Microprocessor and Peripherals Handbook. 1983.
[8] 刘思久, 冯玉贵. GPIB 总线控制器与总线分析仪. 1986 全国自动化学会应用年会, 1986.
[9] 刘思久, 陈祥立, 黄华, 张礼勇. 面向 GPIB 总线接口的虚拟仪器功能. 电测与仪表, 1998(04).
[10] 刘思久, 励强华, 周飞. 基于 GPIB 接口总线的微机光谱分析系统. NI 公司第二届基于计算机的测量和自动化应用方案有奖征文, 2001.
[11] 国家"九五"科技攻关 97-772-01-08 研究报告: 面向自动测试语言编程软件开发工具.ATS98, 2000.
[12] 7150 Remote Control Operating Manual. Solartron Instruments, 1983.
[13] Programmer Manual for TDS 200-Series Digital Oscilloscope. Tektronix, 2000.
[14] 刘思久, 全宏波, 冯玉贵, 张礼勇. USB-GPIB 总线控制器.全国第二届总线技术与测控系统工程学术报告会, 2001.
[15] 郭顺喜, 刘思久, 栾黎明. GBASIC—— 一种具有 IEC625 总线控制器功能的扩充 BASIC 语言的建立和实现. 电测与仪表, 1985.7.
[16] 莫畏. 一种基于 GPIB 的自动测试系统的研究与实现. 电脑开发与应用, 2008.4.

# 第 4 章　VXI 仪用总线系统

VXI 是一种完全开放，适于多渠道供货的模件式仪用总线规范。它是继 GPIB 总线之后，为适应测量仪器从分离台式和机架式结构向更紧凑的模件式机箱结构发展而推出的一种新型总线标准。VXI 总线集中了智能仪器、个人仪器和自动测试系统发展的很多经验，不仅测试速度大为提高，系统性能全面优于 GPIB 总线，而且总体尺寸大大缩小。同时，虚拟仪器的通用软件结构得以体现，适应了自动测试系统向标准化、自动化、模块化、小型化和软件化发展的需要。与 GPIB 系统相比，VXI 作为新一代仪器接口总线，特别适于军用装备和各种高性能要求的场合。

## 4.1　VXI 总线的基本概念

VXI 总线是 VME(Versa Bus Modular European)总线在仪器领域的扩展，本节将从 VME 总线的基本结构开始讨论，并在此基础上阐述 VXI 扩充的仪器功能，进而说明 VXI 机箱和模件的机械结构、电源配置和电磁兼容性等具体要求和特点。

### 4.1.1　VME 总线的基本结构特点

VME 总线是基于 Motorola 公司 32 位微处理器 68000 的一种总线标准，并被美国电气与电子工程师学会和国际电工委员会分别接受为 IEEE1014 和 IEC821，其总线结构如图 4-1 所示。

系统中的 VME 模件根据功能可分为系统控制模件、CPU 模件、存储器模件和 I/O 模件 4 种类型。每个 VME 系统必须配备系统控制模件作为整个系统的控制单元，其他 3 种模件的配置由系统设计者根据需要来选用。每种模件都由背板接口逻辑电路和若干功能模块组成，前者用于满足 VME 总线规范的电气特性(如信号线阻抗、信号传输速度、终端参数等)，后者则用来实现某种具体的应用功能。VME 的信号线可分为数据传输总线(DTB)、仲裁总线、优先中断总线和公用总线等 4 组。

数据传输总线用于 CPU 板上主模块与存储器板或 I/O 板上从模块之间数据、地址及有关控制信号的传送。主模块通过地址修改线 AM0～AM5 提供的编码，可以选择使用不同数目的地址线，如 64KB 寻址范围的短地址(A02～A15)、64MB 寻址范围的标准地址(A02～A23)及 4GB 寻址范围的扩展地址。32 根数据线(D0～D31)在一个 DTB 周期内可同时传送1～4 字节的数据。用数据选通线 $DS0^{*}$～$DS1^{*}$、地址线 A01 和字长选择线 LWORD 配合指定不同种类的数据传输类型，如单字节奇地址或偶地址数据传送、双字节或四字节数据传送及多字节的数据块传输等。VME 采用应答挂钩的握手方式传递数据，这与一般 PC 总线定时控制的

方式不同，主模块通过 AS、DS0 和 DS1 线向从模块发出数据有效的控制信号后，需要被寻址的从模块通过 DTACK 线做出响应。当主模块发生寻址错误，或从器件本身发生故障时，均可引起主模块启动的 DTB 超时。于是，可以由此产生总线错误信号 BERR*，并中止当前 DTB 周期。由于所有 DTB 周期都是异步进行的，因此数据传输速度取决于从模块的响应速度。

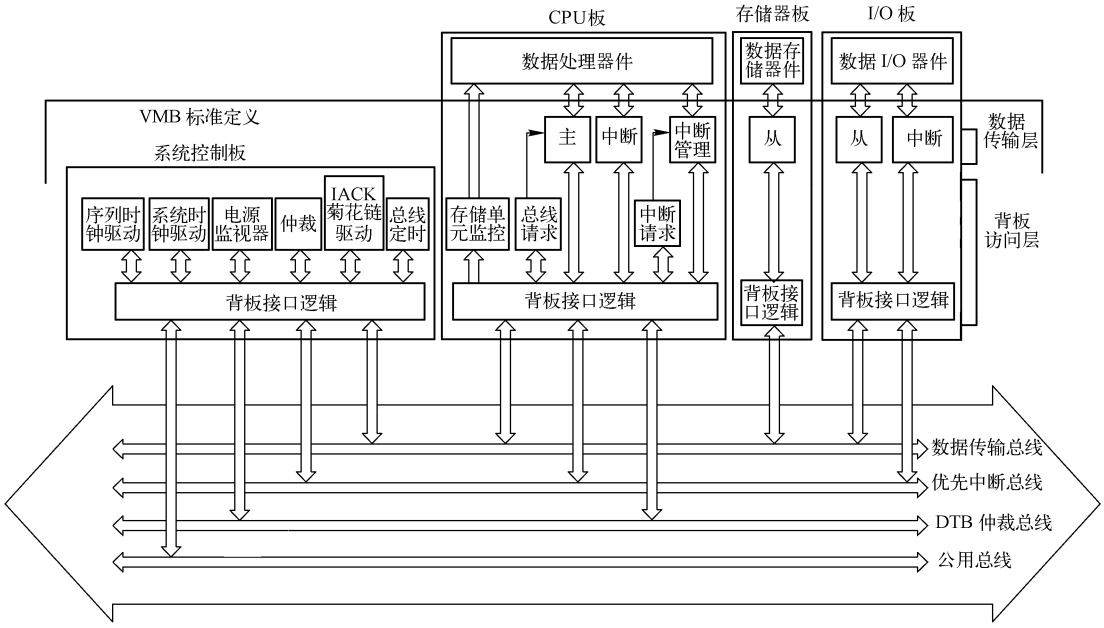



图 4-1 VME 总线的系统结构

VME 标准支持多主模块的分布式微机系统。每个主模块都能使用 VME 系统总线，但却不能出现两个模块同时驱动总线的情况，因此当多个主模块通过 DTB 仲裁总线申请 DTB 使用权时，就需要仲裁机制进行协调和安排。VME 标准规定了优先仲裁、循环仲裁和单级仲裁 3 种方法。优先仲裁将 4 根总线请求线 BR0~BR3 分为不同的优先级(BR3 最高，BR0 最低)，并依次考虑授权；循环仲裁不分优先级，按总线请求线 BR3、BR2、BR1、BR0、BR3、BR2、…的循环顺序平等分配 DTB 的使用权；单级仲裁法则只接收 BR3 线上的请求信号。

当模件板上的主模块或中断管理模块要求使用 DTB 时，其请求模块驱动 BRx 中的一根线向仲裁模块发出 DTB 请求信号。系统控制板上收到请求信号后的仲裁模块把仲裁结果通过如图 4-2 所示的菊花链通知被授权使用 DTB 的总线请求模块。

同一链路上各主模块的优先权由对应当前授权输入/输出线(BGxIN/BGxOUT)组成的菊花链分别决定，距最左面的 1 号槽越近的插件优先权越高。每个发出总线请求 BRx 信号的主模块将其 BGxOUT 线设为“高”，以禁止其右端其他主模块的总线请求。同时监视 BGxIN 线，一旦该线变“真”，则说明其 DTB 请求得到允许，从而驱动 BBSY* 线，表示 DTB 已被占用。更高级别的 DTB 请求发生时，仲裁模块可以通过 BCLR* 线通知当前占用 DTB 的主模块，使其在现行 DTB 周期结束后释放 BBSY* 线，以便系统重新组织仲裁。

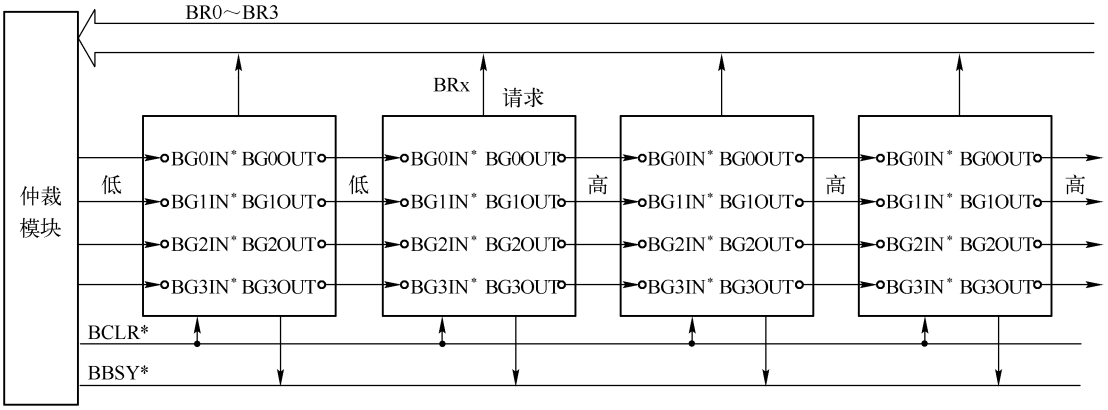



图 4-2　总线仲裁的菊花链结构

优先中断总线用于 VME 系统中断模块和中断管理模块之间进行中断请求和中断认可的操作。VME 支持的中断系统可以是一个具有中断管理模块的单控制器系统，也可以是具有多个中断管理模块的分布式系统。在单控制器系统中，所有中断模块的中断请求都由一个中断管理模块接收，全部中断服务程序由一个 CPU 板的微处理器来执行。分布式系统中的中断管理模块通常分布在不同的 CPU 板上，每块 CPU 板上的微处理器只为其辖域内的中断请求服务。中断模块通过中断请求线向中断管理模块发出中断申请，其中 IRQ7 的优先级最高，IRQ1 最低。在单控制器系统中，中断管理模块监视全部 7 条 IRQx 线；在分布系统中，每个中断管理模块只监视分配给它的部分 IRQ 线。

如图 4-3 所示，与优先中断总线有关的功能模块还有 IACK 菊花链驱动模块。当中断管理模块在其所监视的 IRQ 线上收到中断请求信号时，首先通过模件板内的 DTB 请求模块申请 DTB 的使用权，得到允许后则启动中断认可周期。在这个周期里，中断管理模块使 IACK* 线有效。IACK* 线与 1 号槽的 IACKIN* 相连，IACK* 菊花链驱动模块从该槽的 IACKOUT* 线发出负脉冲，激励菊花链路。同时中断管理模块在地址线 A01～A03 上送出信息，指明现行中断认可周期所对应的 IRQ 线。中断模块则应监视所在槽的 IACKIN* 信号线和 A01～A03 地址线，当其 IACKIN* 线上出现脉冲下降沿且地址线指示它发出的 IRQ 编号时，该中断模块就在数据线上发出 1～4 字节的 STATUS/ID 信息，表明由它来响应现行的中断认可周期。中断

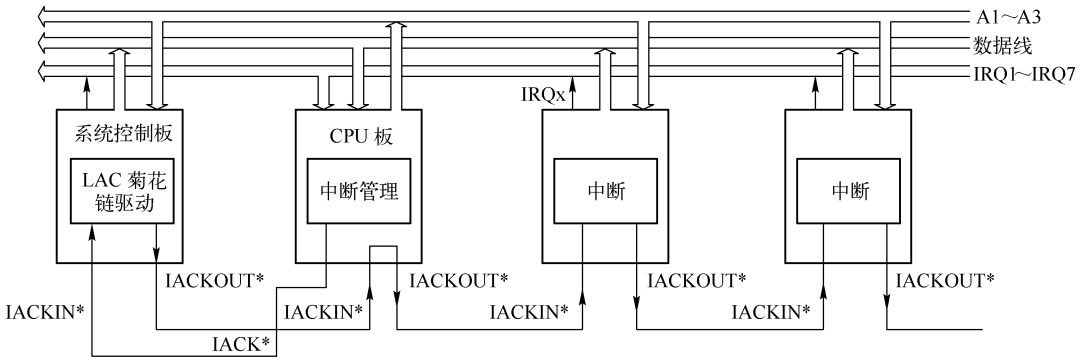



图 4-3　VME 的中断应答链路结构

管理模块根据收到的 STATUS/ID 信息转而执行相应的中断服务程序。正在响应中断认可周期的中断模块则使所在槽的 IACKOUT* 线无效，以防后面其他中断模块响应这个周期。

公用总线为 VME 系统提供了系统时钟及系统初始化和故障诊断等功能。与公用总线有关的功能模块全部安排在系统控制板上，包括系统时钟驱动模块、序列时钟驱动模块和电源监视模块等。系统时钟驱动模块驱动 SYSCLK 线，为 VME 系统提供一个独立的、占空比为 50% 的 16MHz 时钟信号作为系统操作的时间基准。序列时钟驱动模块驱动 SERCLK 线，可为一种称为 IEC823 的扩展总线系统提供特定的周期波形信号。电源监视模块负责监视系统的交流电源故障和系统面板上的复位按钮，当系统电源出现故障时驱动 ACFAIL 线向系统发出报警信号，而当操作人员按下复位按钮时驱动 SYSRESET 线，使整个系统进入初始状态。系统复位后，总线上的所有插件都要进行自检，并将自检结果通过 SYSFAIL 线传送给系统控制板。

### 4.1.2 针对测试扩展的 VXI 总线结构

如图 4-4 所示，VXI 总线以 VME 为基础，同时针对自动测试系统仪器模块之间同步、触发及电磁兼容和供电电源的特殊要求，增加了时钟总线、模块识别总线、触发总线、本地总线、模拟相加总线、星形总线和配电总线等扩展功能。

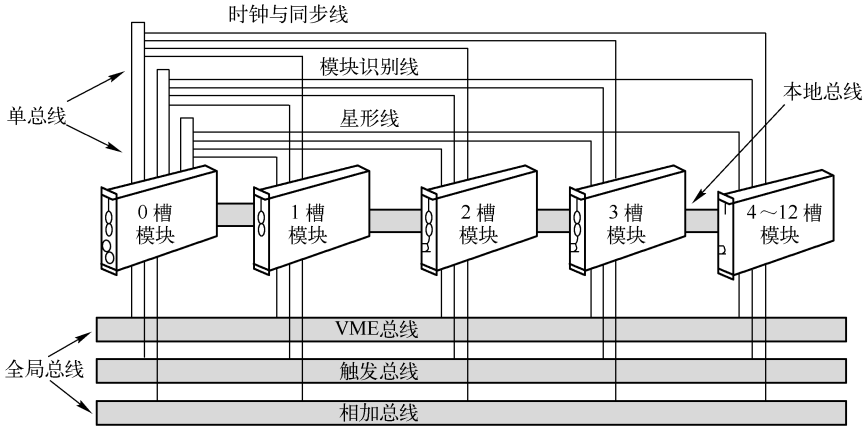



图 4-4 VXI 总线结构

与 VME 的定义不同，VXI 机箱最左面的模件插槽称为 0 号槽。通过 CLK10 时钟线提供 10MHz 的精确时钟，在规定的工作温度下应当具有 ±0.01% 的精度，其波形占空比为 50% ±5%。背板馈送阻抗设计成 50Ω，从 0 号槽到任何槽位的模件，CLK10 时钟线的延迟不超过 8ns。触发线为标准的总线形式，其中 6 条集电极开路的 TTL 驱动线用于模块之间的通信，且它们可以组合成单线组、双线组和四线组等不同形式。另外还有两条 ECL 触发线用于模块间的定时资源，阻抗为 50Ω。

模块识别线(MODID)用来检测模件的存在和位置，如图 4-5 所示，当这些源于 0 槽模件的 MODIDxx 线的电平被插入其他槽位模件上的 825Ω 接地电阻拉低时，不但插槽旁的指示灯可以由此指示模件的连接状态，0 槽模件本身也可通过 MODIDxx 了解模块的连接情况。另一方面 VXI 的 0 槽模件还可以通过读取其 MODID00 引线与背板上 825Ω 接地电阻的连接，来确

认该模件是否已经正确地插在 0 号槽位置，高电平状态则表示该模件已被错误地插到其他槽位上了。

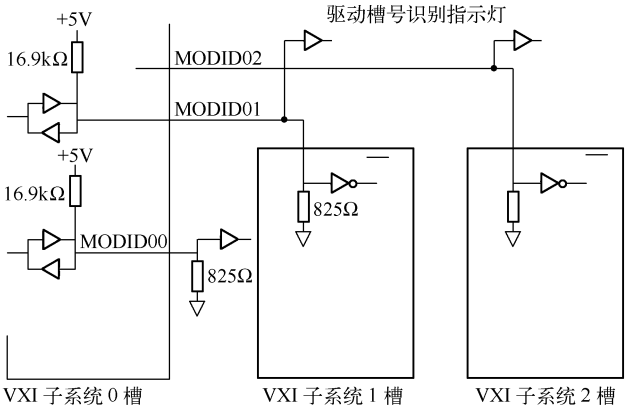
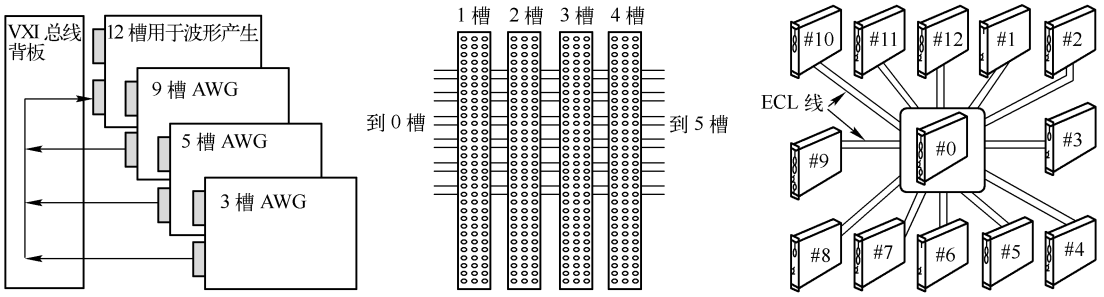



图 4-5　MODID 线的驱动

如图 4-6 所示，VXI 的模拟相加总线上的信号可以被总线上任何模件驱动和接收，各自模件输出的模拟信号相加求和后可在这条总线上形成复杂的波形信号。本地总线 LBUS 为相邻模件之间提供了更为直接的连接关系，星形总线 STARX 和 STARY 则为 0 槽模件和其他模件之间建立了一个通过交叉矩阵开关直接连接的渠道。




(a) VXI 的相加总线　　(b) 本地总线　　(c) 星形总线

图 4-6　VXI 的相加总线、本地总线和星形总线结构

### 4.1.3　VXI 机箱和模件的机械结构

VXI 总线系统的机械结构也以 VME 总线为基础，系统主机箱和模件结构如图 4-7 所示。模件从主机箱前端插入，通过模件连接器插头接入后背板上相应槽位的插座。机箱后部装有电源和冷却系统。

VXI 总线规范规定了 4 种模件尺寸，其中包括与 VME 总线标准完全相同的 A、B 两种尺寸，也包括为适应仪器需要而扩展的 C、D 两种较大的尺寸。如图 4-8 所示，A 型模件只有一个 P1 连接器，B 型和 C 型模件带有 P1 和 P2 两个连接器，D 型模件除 P1、P2 连接器之外还带有 P3 连接器。3 种连接器都具有 3 列 96 引脚，VXI 机箱的插槽间隔宽度由 VME 标准的 0.8in(英寸) 增加至 1.2in，以便容纳较大的元件和进行屏蔽处理。VXI 总线规范要求所有模

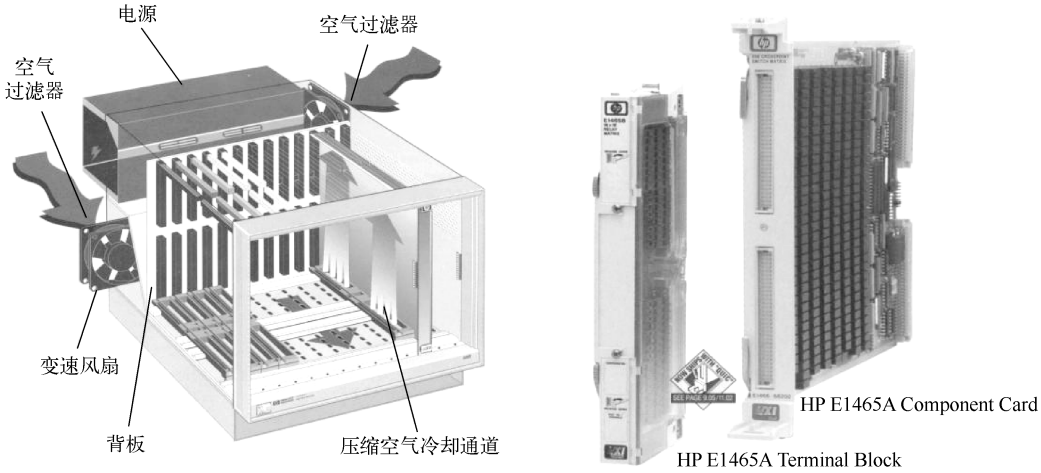



图 4-7 VXI 主机箱及模件结构

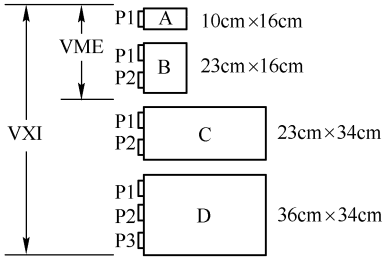



图 4-8 VXI 模件尺寸

件均采用前面板接线方式，模件采用空气冷却，气流方向从机箱的底部向上，要求在没有插入模件的空槽处安装填板，以防该处漏风破坏机箱内部正常的通风气流。规范对屏蔽和非屏蔽的 C 型和 D 型尺寸的模件分别规定了板的翘变形、引线长度、屏蔽高度、允许高度等尺寸要求，并考虑到 D 型尺寸模件有 3 个连接器，插入和拉出时模件受力较大，因此要求设置压入/弹出机构。C 型机箱应用最广，C 型模件和 B 型模件均可插入。

VXI 总线规范规定了对主机箱和模块的冷却要求，要求机箱厂家说明冷却性能，给出气流速度和有效压力。规范要求所有模件都将冷却要求包括在产品说明书中(如周围最高温度 55℃，温升 10℃，压力约为 66.66Pa，风量 1.5Vs 等)。还有一种智能主机箱(如 Tektronix 公司开发的 VXI1410)本身就是一个 VXI 设备，其内部风机和安装在各个插槽处的温度传感器构成了完整的温度控制系统。风机根据温升的情况调整转速或启停间隔，超过调节范围时还会自动报警。同时，VXI 机箱和模件对湿度、震动和冲击等环境试验也都提出了相应的要求。

### 4.1.4 VXI 总线的电气特性和电磁兼容性要求

VXI 总线系统的电源类型如表 4-1 所示，其中+5VDC 是 VXI 总线系统的主电源，为了实现总线上的数据传输，必须提供这个电压。±12V 主要用于串行通信接口驱动电路及模件内部的模拟电路芯片，-5.2V 和-2V 用于 ECL 器件，系统所需其他负电源可由电压调节器从-12V 电源中产生。备用+5V 用于断电时保护存储器中的内容、维护系统时钟等工作。直流负载纹波/噪声是由电源产生的周期性和随机偏差(PARD)，它是在从 0~10MHz 的带宽下测得的峰-峰电压。感生纹波/噪声是存在于后背板电源引脚上附加的峰-峰纹波，是由工作模件注入电流引起的。

表 4-1　电源电压配置

| 电 压 标 示 | 允许的变化范围 | 直流负载纹波/噪声 | 感生的纹波/噪声 |
|---|---|---|---|
| +5V | +0. 25V/-0. 125V | 50mV | 50mV |
| +12V | +0. 60V/-0. 36V | 50mV | 50mV |
| -12V | -0. 60V/+0. 36V | 50mV | 50mV |
| +24V | +1. 20V/-0. 72V | 150mV | 150mV |
| -24V | -1. 20V/+0. 72V | 150mV | 150mV |
| -5. 2V | -0. 260V/+0. 156V | 50mV | 50mV |
| -2V | -0. 10V/+0. 10V | 50mV | 50mV |
| +5V 备用 | +0. 25V/-0. 125V | 50mV | 50mV |

为保证系统中各插板模件之间能可靠、高速地传输信息，VXI 依据 VME 标准对系统电源的特性与分配、电信号的电平、总线驱动器和接收器、信号线的连接、终端网络、线路阻抗等都做了严格规定，背板采用多层 PCB(8 层以上)，将各种电源和地线分层配置，TTL 和 ECL 电路的信号连接也分层安排。

为了保证 VXI 总线模块和系统在电磁干扰的情况下能够正常工作，VXI 总线规范对模件的电磁兼容问题做了特别详细的规定，其中包括电导传输、电导敏感度、辐射传输和辐射敏感度等，也对它们的测试方法提出了建议。对远场辐射传输，要求主机箱和模块说明选择的标准在指定场合的适用性，如 FCC、VDE 或 MIL 规范。每个模件的允许辐射不应超过总辐射量的平均数，即在 13 个模件的主机箱中每个模件的辐射作用不应大于总允许辐射量的 1/13。模件对模件的干扰是近场辐射问题，这些场在模件表面几厘米的范围内作用。模件对模件的干扰也是近场敏感度问题，规范规定了不同尺寸的模件测试方法。因为电场传输容易通过屏蔽技术得到抑制，所以规范对其未做过多的阐述。

### 4.1.5　VXI 总线的系统配置

VXI 系统可以包括一个或多个子系统，每个子系统通常为一个 VXI 机箱，每个机箱最多包括 13 个模件。资源管理器和其他系统公共资源都放在 0 槽模件上，如 VXI 总线中央定时的产生、VME 总线要求的系统控制功能等。其余槽位则可最多配置 12 个普通仪器模件，并分享 0 槽模件提供的各种公共资源。每个模件可以含有一个器件或多个器件，一个器件也可占据多个模件位置。

一个 VXI 总线系统最多可以控制 256 个器件，每个器件都有自己的地址。VXI 总线中的器件有 4 种，它们是寄存器基器件、消息基器件、存储器件和扩展器件。每个 VXI 器件都有一组组态寄存器，系统通过这组寄存器可以识别该器件的类别、型号、厂家及地址空间和存储器要求等。寄存器基器件通过配置某些专用寄存器对其进行直接访问，消息基器件则提供专门的通信寄存器，可以通过命令实现更高级的通信能力。当然，为了实现命令的分析和解释，消息基器件通常都是微处理器控制的器件。规范对存储器件没有具体的定义，ROM 和 RAM 模件都是存储器器件的实例。扩展器件是经过改进后用于 VXI 总线上的 VME 器件。

VXI 总线系统需要主控计算机参与系统工作，可以为每个 VXI 机箱组成的子系统配备一个外部 PC，也可由一个 PC 管理多个机箱形成的多个 VXI 子系统，还可以将计算机放到机箱中，形成嵌入式工作方式。VXI 子系统与外部计算机的连接需要在 0 槽模件上建立与外部计算机的通信接口，最传统的接口形式是 GPIB 通用接口总线，且用 RS-232 接口来实现其监控和调试的辅助功能。

为了提高数据传输的速度，VXI 也常采用一种称为 MXI 的专用连接总线，它采用 62 芯 D 型连接器，连接形式如图 4-9 所示。这种并行总线理论上的数据传输速度可达 20MB/s，实际接口能力约为 8MB/s。随着计算机技术的发展，IEEE1394、USB 等现代串行总线也先后得到了实际的应用。

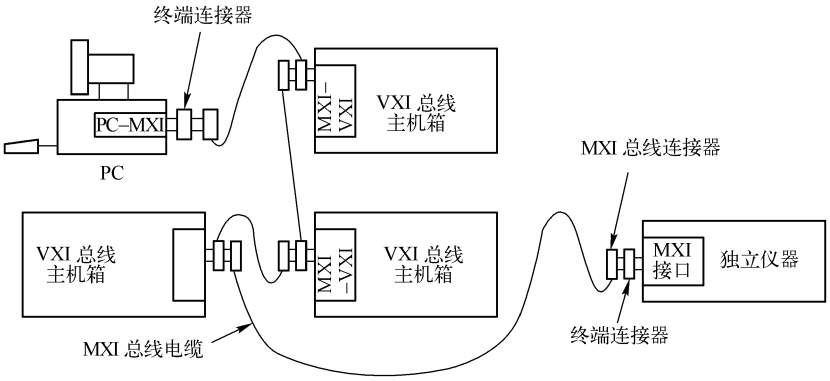



图 4-9　VXI 机箱外接 PC 的 MXI 方式

VXI 总线规范定义了命令者/从者层次。命令者作为总线的控制者，可在需要时得到其对总线的控制权。命令者下面可以有多个从者器件，可以对从者器件进行基于寄存器或基于消息的通信操作。从者能在命令者的控制下向命令者发回数据信号和中断信号。如图 4-10 所示，VXI 系统可以有多种配置方式，通过单个命令者器件的 CPU 控制全部从者器件的方式、多个命令者方式、分层命令者方式等。在多层结构中一个 VXI 器件可以是命令者，又可以是从者，且以分层的通信协议组为基础。

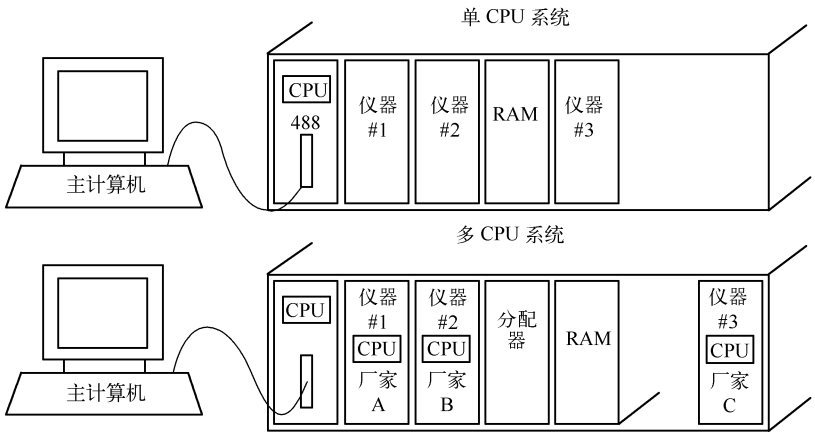



图 4-10　VXI 系统的配置方式

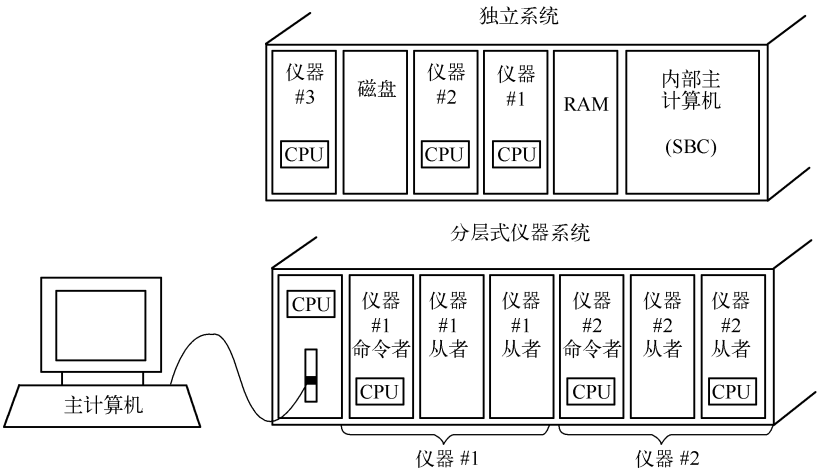



图 4-10　VXI 系统的配置方式(续)

总之，VXI 总线允许在一个主机箱中安装来自不同厂家的卡式仪器、接口板或计算机，用户可按需要选择性能和价格适当的模件和机箱。VXI 系统中各器件可以共享电源、时钟、机架、冷却等硬件资源，从而降低了成本和减小了体积，使多个仪器模件工作在一个紧密联系的环境中，从而获得严密的时间配合，形成高性能的仪器系统。

## 4.2　VXI 总线系统的工作原理

VXI 系统中各器件之间的数据传输是 VXI 的基本功能，本节将重点解释其工作原理，阐述逻辑地址、主从模块、接口寄存器组、通信协议、初始化过程等基本概念，以及资源管理和 0 槽控制器的基本功能。

### 4.2.1　逻辑器件和地址空间

寻址是总线式系统数据通信的基本方法，VXI 系统中每个器件均需要有一个访问地址，称为逻辑地址。在系统上电启动前，逻辑地址可由器件上的编码开关在 0~254 范围内手动选择。但当编码开关放到 255 的位置时意味着手动选择逻辑地址无效，系统中的 0 槽控制器将通过组态的方式为该器件重新分配一个适当的动态逻辑地址。

每个 VXI 器件都有一组接口寄存器，被安排在一个称为 A16 的地址空间之中，以便通过系统的地址总线来进行统一的寻址访问。每个 VXI 器件占据 A16 地址空间的 64 字节，且其基地址可由逻辑地址唯一确定。其计算公式为

$$\text{基址} = \text{逻辑地址} \times 64 + 49\,152$$

例如，某 VXI 器件的逻辑地址为 2，则其占据的 A16 地址空间的绝对地址是 0xC080~0xC0BF(49 280~49 344)。这样，由 256 个逻辑地址建立的 256 组接口寄存器就在系统统一寻址的 A16 空间中占据了 16KB 的地址空间，其绝对地址是 0xC000~0xFFFF(49 152~65 535)。注意，这里所说的每个 VXI 器件的 64 字节是作为 VXI 器件的接口寄存器，这是该器件接受其他器件访问的主要手段，也是 VXI 系统工作的主要方式。

与此对应，系统中还包括一块称为 A24 和一块称为 A32 的地址空间，它们的空间绝对地址范围分别是 00000016 ~ FFFFFF16 和 0000000016 ~ FFFFFFFF16，并应能够分别访问 20000016~DFFFFF16 和 2000000016~DFFFFFFF16 的可用空间。如果某个器件在 A16 中分配的空间不能满足其卡式仪器功能的需要，则可要求在 A24 或 A32 地址空间中分配更多的寄存器或存储器单元，且将其基地址信息放在 64 字节中的某个指定位置。

### 4.2.2 主模块与从模块的数据传输方式

VXI 系统各测试仪器模块之间的数据传输是通过器件的主模块功能访问其他器件从模块所支持的接口配置寄存器组实现的。这里，主模块功能是指其 CPU 具有访问其他器件的从模块寄存器组和共享存储器的能力，而从模块功能则指器件能够提供符合 VXI 标准的接口寄存器组和存储器，并以这个标准寄存器组内容的存取为核心操作仪器动作、反映仪器状态、交换测试数据。因此，从模块功能及其配置寄存器组集中体现了 VXI 的工作方式，故 VXI 总线规范以很大的篇幅阐明从模块功能，以及接口对其配置寄存器组的操作要求。

如图 4-11 所示是一个消息基 VXI 接口的实现框图，这里接口工作的核心安排在 A16 地

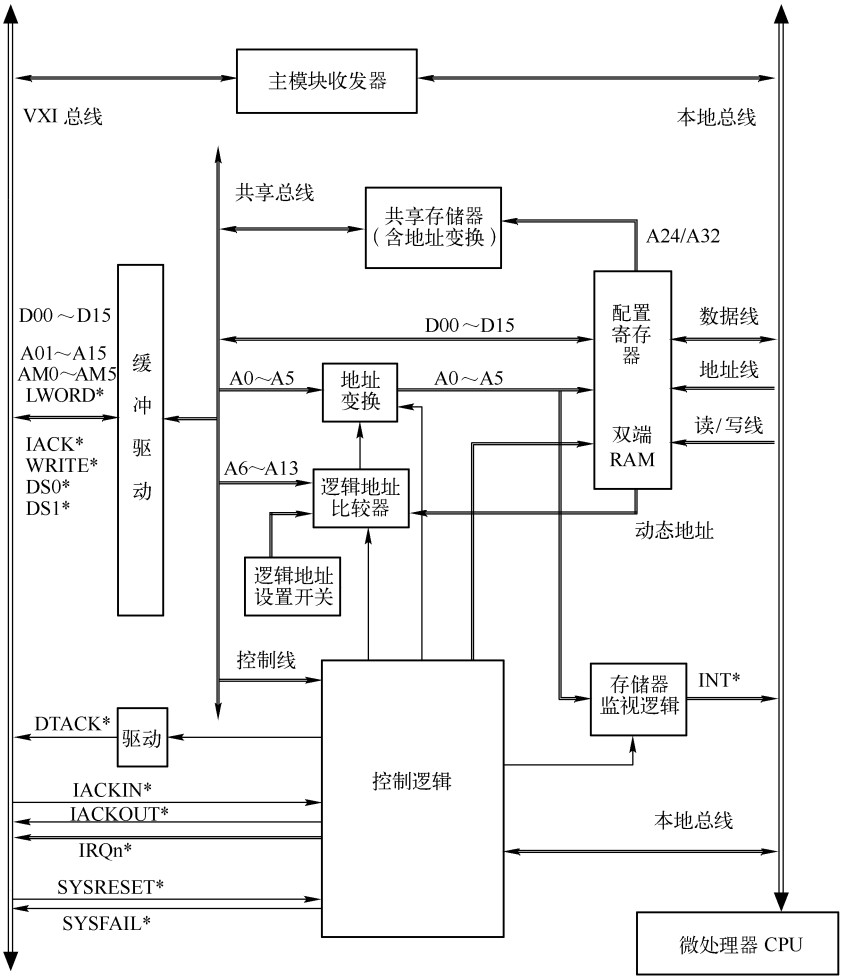



图 4-11 实现 VXI 主-从模块之间数据通信的示意图

址空间中的那组配置寄存器。共享总线是实现 VXI 和本地总线之间交换信息的一种方法，VXI 总线与本地总线分别通过缓冲器连接共享总线，使它们能在控制逻辑的管理下轮番使用共享总线来访问共享存储器和配置寄存器，从而达到 VXI 总线与本地总线之间交换信息的目的。配置寄存器组采用双端口 RAM 是实现这种信息交换的另一种方法，此时配置寄存器组在接受 VXI 总线上其他模件通过共享总线访问的同时，也可直接接受模件本身通过本地总线的访问。

来自 VXI 总线地址线中的逻辑地址信息应与地址开关设定的模件地址比较，以产生设备选通信号，并作为总的使能条件选通接口的从模块功能。由此共享总线直连的数据线、通过地址变换形成的地址线及控制逻辑作用下产生的读/写线就构成了配置寄存器组的实际访问条件。对采用动态逻辑地址的情况，配置寄存器组中的内容将代替地址开关来设定模件的逻辑地址。寄存器监视逻辑可以监视共享总线对该寄存器组中某些重要寄存器的访问，并能以中断方式直接通知本地 CPU。这样，在共享总线更新某寄存器内容后本地 CPU 就能及时读出更新后的信息，或在共享总线读出其内容后及时写入新的信息。共享存储器的工作原理与配置寄存器组的情况相似，也须进行地址变换，其基址信息来源于配置寄存器组。VXI 总线数据传输的挂钩、仲裁、中断和初始化控制功能均通过控制逻辑完成，有些直接操作相应电路，有些则通过本地总线与模件的 CPU 交换信息。

相比之下，主模块中操作 VXI 总线的功能则显得比较单纯，只须根据 VME 总线数据传输的协议要求，完成 VXI 总线的读/写操作。

### 4.2.3　VXI 器件的接口寄存器组

VXI 器件依照其功能类型可以分为存储器件、扩展器件、消息基器件和寄存器基器件 4 种，它们的接口操作寄存器组的配置不同。然而所有器件都包括最基本的一组组态寄存器。

现在需要说明的不是这些寄存器内部各工作位的具体格式，而是需要明确这些标准接口寄存器所能表示的操作功能究竟有哪些内容。如图 4-12 所示，在一个 VXI 器件的组态寄存器组中，逻辑地址分配寄存器用于写入动态配置的逻辑地址，而 ID 寄存器则用于读出该器件的基本特征，其中包括生产厂家识别码、地址空间码和器件类型码。厂家识别码是从 4095 向下分配的，如 HP、Wavetek、Tektronix、Colorado Data System 和 Racal-Dana 等公司的识别码分别为 4095～4091；地址空间码(a=01，10，11)用来说明器件所需使用的地址空间，A16 方式(a=11)意味着器件只使用 A16 地址空间中分配给该器件的 64 个地址，而 A16/A24(a=01)或 A16/A32(a=10)工作方式则表明器件还要求使用 A24 或 A32 中的地址空间；器件类型码用 00～11 分别表示存储器件、扩展器件、消息基器件和寄存器基器件。

器件类型寄存器用于读出器件的型号编码和器件要求的存储空间码(m=0～15)，型号编码是识别器件的产品型号，而存储空间码 m 与地址空间码 a 结合，就可以计算实际存储空间，计算公式为 $256^a+2^{23-m}$。对 A16(a=11)工作方式，存储空间码没有意义，此时型号编码扩展到寄存器的全部 16 位。状态寄存器和控制寄存器则分别用于读出器件工作的状态和写入控制器件的命令。其状态位 A24/A32、MODID*、Ready 和 Passed 均用于表示系统初始化过程中各阶段的状态，而控制位 A24/A32、Sysfail 禁止和 Reset 则用来接受初始化过程中来自系统的操作。复位方式有两种，硬件复位 SRST(总线信号 $SysReset^*=1$)和软件复位(将控制寄存器的 Reset 位置 1)。偏移寄存器是一个读/写寄存器，可以用来读出该器件存储器在

A24 或 A32 地址空间中的操作基址，也可以通过写入该寄存器来设定新的基址。

| | |
|---|---|
| 00 | ID/ 逻辑地址分配 |
| 02 | 器件类型 |
| 04 | 状态 / 控制 |
| 06 | 偏移 |
| 08 | 协议 |
| 0A | 响应 |
| 0C | 数据高 |
| 0E | [ 数据低 ] |
| 10 | [A24 指针高 ] |
| 12 | [A24 指针低 ] |
| 14 | [A32 指针高 ] |
| 16 | [A32 指针低 ] |
| 18<br>⋮<br>1F | VXI 系统保留寄存器 |
| 20<br>⋮<br>3F | 与器件相关的寄存器 |

ID 寄存器

| 15 ～ 14 | 13 ～ 12 | 11 ～ 0 |
|---|---|---|
| 器件类型 | 地址空间 | 生产厂家识别符 |

器件类型寄存器

| 15 ～ 12 | 11 ～ 0 |
|---|---|
| 存储空间要求 | 型号编码 |

状态寄存器

| 15 | 14 | 13 ～ 4 | 3 | 2 | 1 ～ 0 |
|---|---|---|---|---|---|
| A24/A32 允许 | MODID× | | Ready | Passed | |

控制寄存器

| 15 | 14 ～ 2 | 1 | 0 |
|---|---|---|---|
| A24/A32 允许 | 与器件相关 | sysfail 禁止 | 复位 |

协议寄存器

| 15 | 14 | 13 | 12 | 11 | 10 | 9 ～ 4 | 3 ～ 0 |
|---|---|---|---|---|---|---|---|
| CMDR* | 信号寄存器不存在 | 无主模块 | 能中断 | FHS* | 共享存储 | 保留 | 与器件相关 |

响应寄存器

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 ～ 0 |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 保留 | DOR | DIR | Err* | 读准备好 | 写准备好 | FHS 激活 | 封锁 | 与器件相关 |

图 4-12　VXI 寄存器组配置

与器件相关的寄存器及其各组成位是由 VXI 模件设计者自己定义的。通过访问这些寄存器，可以直接操作 VXI 器件的各种功能。例如，这些寄存器可以用来作为多路开关工作状态的映像图，从而直接实现对多路开关的监控操作。

对 VXI 消息基器件来说，除了需要配置上述基本的配置寄存器组和与器件相关的寄存器组外，还需要增加一组通信寄存器。VXI 总线系统中与消息基器件的通信操作将主要通过读/写其数据寄存器来完成，同时响应寄存器用来反映数据传输的各种状态，以确保数据传送过程的正确执行。另外，可由读取协议寄存器的内容得到器件所支持的协议及其通信能力。而寄存器基的器件则不支持上述这些通信方式，与其交换信息只能通过直接读/写与器件直接有关的那些寄存器来完成。

对于 VXI 共享存储器器件，将会用到 A24 指针寄存器与 A32 指针寄存器。

### 4.2.4　消息基器件之间的通信协议

VXI 系统最基本的通信方式是消息基命令者器件与消息基从者器件之间的通信，最基本的通信协议是字串行协议，在此基础上可以建立更高级的串行协议和共享存储器协议。

字串行协议中传输数据通过对从者的配置寄存器组中的数据低位寄存器(16 位)进行读/写操作，并将所有写操作解释为命令。字串行协议可以采用正常传输和快速握手两种方式，正常传输方式要求将命令写入从模块数据寄存器前必须首先确认从者器件中响应寄存器的“写准备好”位有效，而由从模块数据寄存器读出响应信息前必须确保从者响应寄存器的“读准备好”位有效，以此实现数据的同步传输。快速传输方式下的从模块器件总是处于“读准备好”和“写准备好”状态，同步仅受从模块的硬件挂钩信号 DTACK* 和 BERR* 的限定。从模件器件对快速握手方式的支持情况可由从模件协议寄存器的 FHS* 位激活状态决定。当字串行

协议的一致性错误发生时，从者响应寄存器中的 Err* 位置 0，这时可以通过“读协议错误”命令将错误的信息报告给命令者。

字节传输协议是命令者和从者之间传输 8 位数据的另一种方式，在这种协议下传输数据的过程如下。

发送数据(命令者→从者)的工作过程为：①命令者确认从者响应寄存器的 DIR=1；②命令者向从者数据寄存器发送“字节可用”命令；③命令者向从者数据寄存器依次发送字节数据。

接收数据(从者→命令者)的工作过程为：①命令者确认从者响应寄存器的 DIR=1；②命令者向从者数据寄存器发送“请求字节”命令；③命令者确认从者响应寄存器的 DIR=0；④命令者通过从者数据寄存器依次读取字节数据。

### 4.2.5　VXI 器件的初始化过程

图 4-13 和表 4-2 明确给出了初始化过程中各工作状态与标准寄存器的关系。开机上电后 0 槽控制器通过拉低 SysReset* 线，发出的系统有效复位信号(记为 SRST)迫使系统中各 VXI 器件接口进入硬复位状态。这时各器件状态寄存器的 Passed 位和 Ready 位、控制寄存器的 Reset 位和 SysFail 禁止位都被清零。同时，由于 SysFail 线被拉低，从而点亮 0 槽控制器面板上的故障指示灯。

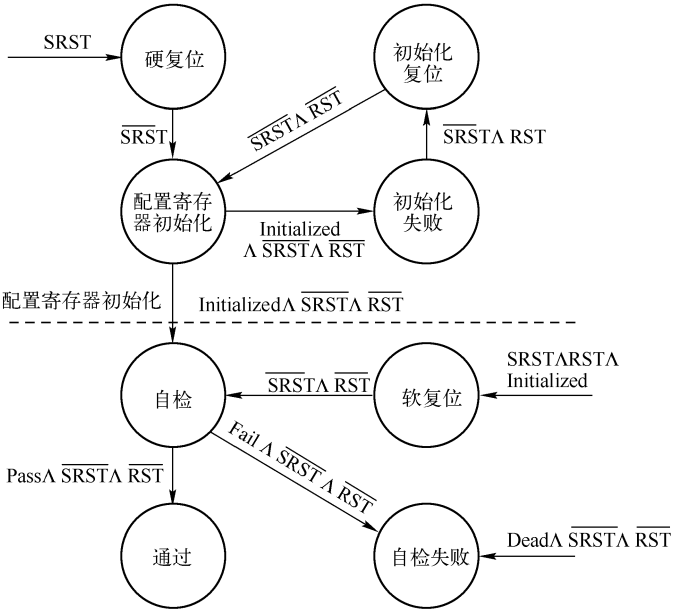



图 4-13　VXI 启动过程

**表 4-2　控制寄存器表达的状态**

| 状态 | Passed 位 | Ready 位 | SysFail 线 | SysFail Inhibit 位 | 灯 |
|---|---|---|---|---|---|
| 硬复位 | (0) | (0/1) | 0 | 0 | 亮 |
| 配置 | (0) | (1) | 0 | 0 | 亮 |
| 初始化失败 | 0 | 1 | 0 | 0 | 亮 |
| 自检 | 0 | 0 | 0 | 0 | 亮 |
| 失败 | 0 | 0 | 0 | 0 | 亮 |
| 通过 | 1 | 0/1 | 1 | 0 | 灭 |
| 软复位 | 0 | 0 | 0 | 1 | 灭 |
| 初始化复位 | 0 | 1 | 1 | 1 | 灭 |

当复位信号结束(SysReset* 线变为高电平，记为 SRST*)后，正常工作的各 VXI 器件接口将相继进入配置寄存器初始化状态，包括逻辑地址、ID、状态、控制和偏移寄存器的设置。此时 SysFail 保持有效，故障指示灯继续点亮，直至其状态寄存器状态位 Ready 变为 1，仍然禁止访问 VXI 器件。当配置成功后，器件在内部信号 Initialized 作用下从配置状态进入自检状态，全部自检过程(包括配置寄存器初始化)应在 SysReset* 线变高或状态位 Ready 由 1 变 0

后的 4.9s 内完成。若器件成功地完成自检，则进入通过状态。此时应使状态寄存器的状态位 Ready 和 Passed 均为 1，允许 VXI 总线访问它的组态寄存器。同时释放 SysFail 线、熄灭故障指示灯，表明它已准备好与 VXI 总线通信。如果某器件不能完成所有组态寄存器的初始化，则产生本地信号 SysFail，进入初始化失败状态。于是在 Ready 位为 1 的情况下保持 Passed 位为 0，表明器件初始化失败。如果器件没有通过自检，则产生本地信号 Fail，进入自检失败状态，在 Passed=0 和 Ready=0 下保持 4.9s 后表明自检失败，于是驱动 SysFail 线有效，点亮故障指示灯。在初始化失败状态或自检失败状态下，可以通过向控制寄存器中 Reset 位写 1 的方法(记为 RST)进入软复位状态或自检复位状态，从而再次转入配置寄存器初始化状态或自检状态。另外，一个 VXI 主模块也可以通过设置自检失败器件控制寄存器中的 SysFail 禁止位来封锁 SysFail 线。

### 4.2.6 VXI 资源管理器与 0 槽控制器

VXI 总线的资源管理器是一个具有命令者能力的消息基器件，它的逻辑地址为 0。系统上电时，完成诸如自检管理、器件识别与组态、A24 和 A32 地址的映像组态、命令者/从者分层(系统分层)、分配 IRQ 线(IRQ 中断线)及启动正常操作(启动服务)等 6 项任务。

#### 1. 自检管理

系统需要首先等待所有器件完成自检，并在禁止 SysFail 线有效时，强制自检失败的器件进入软复位状态。也可用一些与器件相关的方法来完成失效器件的诊断测试。

#### 2. 器件识别与组态

对 A16 中全部 256 个可能存在的器件依次进行寻址访问，如果在访问某一逻辑地址的器件状态寄存器时产生总线错误，则表明该器件并不存在，从而依次记下所有成功读回状态的逻辑地址，即可达到器件识别的目的。对于逻辑地址为 255 的器件，需要利用与机箱插槽位置一一对应的 MODID 标识线进行动态组态，重新分配逻辑地址。

#### 3. A24 和 A32 地址的映像组态

读取每一器件对 A16/A24 和 A16/A32 的地址空间要求，根据不同的规则(如器件类别和存储器特性)为提出要求的器件分配一个基地址偏移量，并通过偏移量的合理安排，保证各器件的地址空间不会重叠。

#### 4. 系统分层

资源管理器建立系统分层结构的方法如下：

(1) 检查每一个基于消息的器件协议寄存器中 CMDR(控者)位，找出所有命令者。

(2) 用读从者区域(Read Servant Area)命令，读取每一命令者拥有的从者区域大小。

(3) 确定命令者/从者层次。

(4) 利用授予器件(Grant Device)命令给命令者分配从者。

VXI 总线命令者/从者地址映像算法实际上不对层次的数目进行限制，它只是基于每个 VXI 总线器件的逻辑地址以及每一个命令者的从者区域大小。从者的区域大小是一个 8 位数

值(0~255)，由用户或生产厂家选定，以非易失的方式保存在每一个命令者器件中。每个命令者所属的从者区域从命令者的逻辑地址之后开始，所包括的数量由该命令者的从者区域大小确定。如果某一器件是在某一已给定的命令者的从者区域中，并且它不属该命令者从者区域里其他器件的从者区域，则此器件就是那个命令者的一个从者，可由资源管理分配给那个命令者。

#### 5. IRQ 中断线

VXI 总线系统的 7 根 IRQ 中断线可由资源管理器通过命令分配给系统中各中断管理器和中断器。每一根 IRQ 线仅可分给一个中断管理器和若干个中断器使用。对于有跨接线的器件，像对 PI(可编程中断器)和 PH(可编程中断管理器)一样，选择 IRQ 线的分配要求用户提供一些中断组态信息。资源管理器给 PH 分配 IRQ 线，先用读协议命令确定哪个基于消息的器件具有 PI 能力。首先，使用所提供的中断分配信息；其次，根据命令者的 IRQ 线分配情况，给每一个有 PI 能力的从者分配一根 IRQ 线；最后，用读中断器(Read Interrupter)、读中断器线(Read Interrupter Line)以及分配中断器线(Assign Interrupter Line)命令，为每个 PI 指定一根 IRQ 线并对其 IRQ 分配进行编程。

#### 6. 启动服务

分配 IRQ 线后资源管理器发出开始正常操作的命令，伴随顶层位置“1”，以递增逻辑地址的顺序传送给所有最高级别的命令者。至此，资源管理器上电过程完成，可进入运行操作方式，要求每一最高级别的命令者对其从者进行初始化，它应发出开始正常操作命令并伴随顶层位清零，传送给它所有的消息基从者。

VXI 总线 0 槽位器件为系统中其他槽位的器件提供公用资源，包括模块识别 MODID，时钟 CLK10、CLK100，也包括 SYNC100、STARX 及 STARY 等信号。由于通常将此类特殊的系统功能和服务放在 VXI 箱内的 0 槽器件，因此也称 0 槽控制器。

## 4.3　VXI 系统主机的工作方式和应用示例

由于 VXI 模件没有传统的操作面板，因此 VXI 系统的操作和显示总是需要利用计算机主机通过虚拟仪器的方式来实现。本节将分别以 GPIB-VXI/C 控制器方式和嵌入式主机方式说明最基础的计算机主机软件的支持方法，并进一步通过示例指出 VXI 系统的一般应用方式。

### 4.3.1　GPIB/VXI 的总线控制方式

计算机对 VXI 卡式仪器系统的控制有几种不同形式。鉴于自动测试发展的历史原因，依靠插在 VXI 机箱 0 槽位的 GPIB-VXI/C 控制器，通过 GPIB 总线从外部 PC 控制 VXI 系统的方法最为传统。尽管这种方式的数据传输速度受到 GPIB 总线的瓶颈限制，但由于标准化程度高，能够适应计算机设备的更新换代，因此仍为很多测控系统的组建者所偏爱。

GPIB-VXI/C 控制器方式通常将整个 VXI 机箱看做一个子系统，并为 0 槽控制器本身分配一个 GPIB 主地址，再为机箱中每个 VXI 模件各自分配一个副地址。然后，就可以利用

VXI 字串行协议，通过主机中 GPIB 控制器与 VXI 机箱中的各个模块进行数据通信了。这里，GPIB-VXI/C 控制器相当于一个翻译机，它将来自 GPIB 总线的主机命令解释分析，将表达仪器的 GPIB 副地址映射到 VXI 的逻辑地址，然后作为 VXI 的主模块访问相应的 VXI 设备。由于 IEEE488.2 的推广，GPIB-VXI/C 控制器通常也能提供足够的程控命令组，直接通过解释主机发来的命令字串，实现对机箱内部 VXI 仪器的读/写操作。下面以一个 VXI 数字万用表的虚拟仪器建立过程为例，说明 VXI 系统应用的基本方法。

如图 4-14 所示，这里组成的 VXI 系统选用 Tektronix 公司开发的智能机箱 VXI1410，使用的 VXI 卡式仪器是 Wavetek 1362，0 槽控制器采用 NI 公司的 GPIB-VXI/C 控制器，主机接口卡采用 NI-4882。

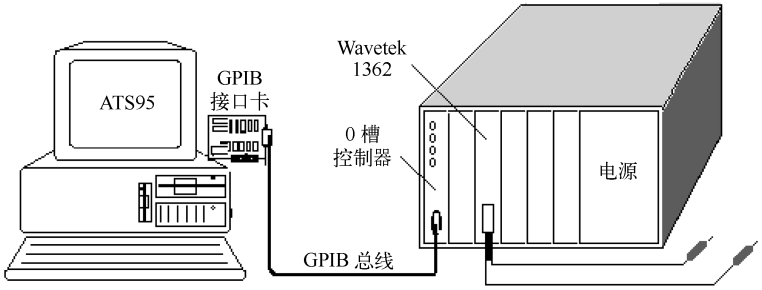



图 4-14　VXI 虚拟仪器示例

如图 4-15 所示为 VB 开发环境下的数字万用表虚拟仪器面板，在 VB 编程状态分别单击各按钮可进入相应的编程代码。程序清单 4-1 给出了这些响应代码的主要部分，通过注释和后面的说明可使读者了解和掌握其基本设计思想和具体实施方法。




图 4-15　VXI 数字万用表虚拟仪器面板

**程序清单 4-1**

```
Dim w1362 As Integer,sCmd1 As String * 30,sCmd2 As String * 30,sData As String
 * 30
Private Sub Form_Load()                                    '加载过程代码
    sta% =VXI_Setup(1,GP_F_EOI,10)
    w1362 = 1                                             '设定仪器的 VXI 逻辑地址
End Sub
Private Sub CmdReset_Click()                              '复位键响应代码
    sta% =VXI_Reset()                                     'VXI 系统复位
    DelaySec 5
    sta% =VXI_SendStr(w1362,"*RST",4)
    DelayMs 1500                                          '仪器复位
```

```
End Sub
Private Sub Cmd1V_Click()                          '1V 键响应代码
    sCmd1 = "INP:STAT 1"
    sCmd2 = "MEAS:VOLT:DC? 1E0,1E-6"
    Label8.Caption = "V":Led1.Value = ""           '建立 1V 电压量程字串命令
End Sub
Private Sub Cmd1A_Click()                          '1A 键响应代码
    sCmd1 = "INP:STAT 1"
    sCmd2 = "MEAS:CURR:DC? 1E2,1E-6"
    Label8.Caption = "A":Led1.Value = ""
End Sub                                            '建立 1A 电流量程字串命令
Private Sub CmdTrig_Click()                        '触发测量键响应代码
    n1% = Len(Trim(sCmd1)):n2% = Len(Trim(sCmd2))
    sta% = VXI_SendStr(w1362,sCmd1,n1% )           '发送程控命令字串
    sta% = VXI_SendStr(w1362,sCmd2,n2% )
    DelayMs 1500
    sta% = VXI_RecvStr(w1362,sData,15)             '接收测量结果字串
    Label7.Caption = Now                           '显示测量时刻
    s$ = Mid(sData,1,5):Led1.Value = s$            '数码显示测量结果
    Led1.Value.Refresh
    SoundData(s$ )                                 '音响报数测量结果
End Sub
Private Sub KNOB1_Click()                          '旋钮转动响应代码
    Timer1.Interval = Knob1.Number
End Sub
Private Sub Timer1_Timer()                         '定时中断响应代码
    CmdTrig_Click()
End Sub
Private Sub CmdCont_Click()                        '连续测量键响应代码
    Timer1.Enabled = 1
End Sub
Private Sub CmdStop_Click()                        '终止测量键响应代码
    Timer1.Enabled = 0
End Sub
```

由于系统采用 VXI-GPIB/C 的形式，因此加载程序后，需要首先通过函数 VXI_Setup 建立主计算机与 VXI 机箱系统的联系。

函数 VXI_Setup 的内部只是预先保存 VXI-GPIB/C 控制器的 GPIB 地址、建立限时操作的等级以及设置读/写操作的结束方式，并未开始实际通信。函数 VXI_Reset 内部更是直截了当，就是通过 GPIB 总线向 GPIB-VXI/C 控制器发送程控命令 *RST 和 SysReset，从而由 0 槽控制器完成对 VXI 总线的复位操作。实际对消息基的 VXI 仪器的操作主要采用 VXI_SendStr 和 VXI_RecvStr 两个函数，它们的内部实现方法无非是通过 GPIB 总线向 GPIB-VXI/C 控制器发送程控命令，其命令字串的最终形式如下。

- 发送：WSstr《逻辑地址》，“《字符串数据》”
- 接收：WSstr?《逻辑地址》，“《取数长度》”

其中，关键字后操作仪器的逻辑地址由函数的首参数直接转换成字串，待发送的字符串数据

则紧跟逗号分隔符后且夹在两个双引号之间，接收命令需含取数长度(也以双引号括起的字符串形式出现在逗号分隔符后)，并随后通过 GPIB 总线取回响应数据字串。

VXI 卡式仪器 Wavetek 1362 的逻辑地址应根据其编码开关的设置建立(本例取值为 1)，也可以通过 VXI 0 槽控制器中的资源管理器读取。从程序清单中可以看出，单击复位键将分别实现 VXI 系统和 Wavetek 1362 的复位；万用表的操作分触发和连续两种工作方式。由于程控命令中包括了触发命令，因此在本示例中单击各功能选择键时，仅将程控命令写到一个字符串变量中，而在单击触发键时再将该字串变量作为程控命令，调用函数 VXI_SendStr 来完成对 VXI 仪器的实际操作；连续测量工作方式是通过单击连续测量键启动一个定时器，在定时器产生的定时事件中实现仪器的每次测量。定时器的定时周期由一个以旋钮形式出现的模拟量设置器建立，每当用户操作旋钮使其设定值改变时都会产生一个事件，故可在这个事件的响应代码中安排对定时器周期的设置，从而达到改变旋钮的指针位置就改变采样周期的目的。这个旋钮的虚拟仪器功能是自行开发的控件，在其标识刻度处单击鼠标可使指针立刻转到该位置，而在旋钮左、右的下部非刻度盘处单击则使旋钮步进动作，还可按住旋钮旋动，控件属性 Value 的值对应最终旋钮的位置。读取测量结果由函数 VXI_RecvStr 完成，其值既可通过自行开发的 LED 控件显示，也可通过语音报数函数 SoundData 直接播出报数声音。

在 GPIB-VXI/C 控制器方式下，VXI 0 槽控制器与 VXI 设备之间的通信仍为 VXI 总线高速方式，也同样可以直接使用 VXI 系统强大的触发、同步功能。但 0 槽控制器与主机之间的 GPIB 连接方式不仅导致主机与 VXI 设备之间的速度变慢，而且 VXI 触发、同步功能对主机的作用也受到明显限制。因此在对这方面有较高要求的场合，可能需要采用嵌入式主机的工作方式。

### 4.3.2 嵌入式主机的 VXI 系统工作方式

由于 PC 发展迅速，因此置于 0 槽位置且直接支持 0 槽控制器功能的嵌入式主机在计算机通用性能指标上通常无法与市场上的主流 PC 媲美，但由于其直接控制和管理 VXI 总线，不仅整体的实际运行速度更快、体积更紧凑，而且处理触发、中断和同步的功能更为协调，支持软件的开发也更为直接，特别适于开发专用的测试系统。实际实施时常常以支持 PC104 的嵌入式 PC 为核心，从而既能充分利用 PC 系统极为优越的软件开发环境，又能简单地实现嵌入式系统的结构需要。

这里介绍一些用于直接控制 VXI 总线工作的函数形式，以此说明 VXI 总线实际工作的方式和过程：

```
发送数据字符串  VXI_SendStr(la% ,data$ ,n% )
接收数据字符串  VXI_RecvStr(la% ,data$ ,n% )
写仪器寄存器  VXI_WrReg(Byval la% ,Byval bias% ,Byval data% )
读仪器寄存器  VXI_RdReg(Byval la% ,Byval bias% ,data% )
写系统 A16 存储区  VXI_WrA16(Byval adr16% ,data% ,Byval n% )
读系统 A16 存储区  VXI_RdA16(Byval adr16% ,data% ,Byval n% )
写系统 A24 存储区  VXI_WrA24(Byval adr24,data% ,Byval n% )
读系统 A24 存储区  VXI_RdA24(Byval adr24,data% ,Byval n% )
```

这些函数可以直接读/写 VXI 系统的寄存器和存储器区，参数 la%和 bias%分别为被访问

的 VXI 模件的逻辑地址和寄存器偏移位置，参数 adr16 和 adr24 则分别为 A16 和 A24 的访问地址，n%为传送的字数，参数 data 为传送数据的数组变量。

另外，VXI 操作也需要一些实现 0 槽控制器本地功能的重要函数，包括资源管理、触发、中断、组态等。这里 0 槽控制器的信息管理系统可根据下列函数的不同要求，分别将相应的信息通过字串变量参数 data 返回。当参数 la 取值为 1～254 时，仅返回相应逻辑地址的 VXI 器件信息，否则返回所有 VXI 器件的信息。

```
读取 0 槽主要信息　VXI_GetCtrl0(data$)
读取资源管理信息　VXI_GetRm(la%,data$)
读取设备组态信息　VXI_GetCfg(la%,data$)
读取设备静态信息　VXI_GetDev(la%,data$)
设置触发线　VXI_SetTrigMap(TrigSource%,TrigDist%,TrigMode%)
执行触发　VXI_ExecTrig(TrigLine%)
设置中断　VXI_SetIntMap(IntSource%,sta%)
等待服务请求　VXI_WaitInt(tmo)
```

在有关触发功能的函数中，参数 TrigSource、TrigDist、TrigLine 均以约定的二进制位形式描述 VXI 触发线，包括 TTL 触发线 0～7、ECL 触发线 0～1、TIC 计数器和 TIC TICK 定时器，以及 0 槽本身提供的本地功能，参数 TrigMode 则用于规定触发的方式。

### 4.3.3　仪器驱动程序

仪器驱动程序也是一些操作函数，但它与前面介绍的 VXI 总线操作函数处于不同的层次，其内部可能需要多次调用总线操作函数和其他底层代码来支持仪器的某项特殊功能，其外部为应用软件提供支持直接操作某种特定仪器的高层函数功能。用户可以直接利用仪器驱动程序提供的功能函数组建自动测试系统，也可以利用这些功能函数建立更符合自己需要的虚拟仪器。下面是某数据采集器驱动程序支持的一些函数形式，以便从其函数设置和参数配置的情况看出这个数据采集器的操作方法：

```
初始化函数　Daq2000_Init(la%,query%,reset%,tmo%)
系统复位函数　Daq2000_SysReset()
复位函数　Daq2000_Reset(la%)
取状态函数　Daq2000_GetSTB(la%,state$)
触发函数　Daq2000_Trigger(la%)
寄存器读函数　Daq2000_ReadReg(la%,offset%,data%)
寄存器写函数　Daq2000_WriteReg(la%,offset%,data%)
块读函数　Daq2000_ReadData(la%,offset%,len%,data()
滤波设置函数　Daq2000_Filter(Modu1%,Modu2%,Filter%)
通道设置函数　Daq2000_SetCh(la%,Modu1%,Modu2%,Am1%,Am2%,num%,Time&,
Pnts%)
通道信息设置函数　Daq2000_SetChInfo(num%,ID$,name$,type$,amp%,unit$,sensi
#)
```

这里显示的几条函数分别用于实现数据采集器的初始化、系统组态、触发、测量等功能，其中初始化函数首先建立计算机与 VXI 之间的连接，并根据参数来决定是否复位和识别数据采集器模块，以及设置限时等级。系统复位函数用来复位整个 VXI 系统，复位函数则用来复

位数据采集器本身，使之回到默认设置状态。取状态函数用来读取数据采集器的状态寄存器，以得到数据采集器当前的工作情况。触发函数用来向数据采集器发送一个触发命令来启动数据采集过程。寄存器读/写函数提供了直接对数据采集器寄存器的读/写操作。块读函数用于以数据块的形式从数据采集器读取测试结果。滤波函数用来设置两个调理模块的滤波功能。通道设置函数用来设置数据采集器和调理模块的各种参数，如放大倍数、通道数、采样间隔及采集点数等。通道信息设置函数与上面的通道设置函数不同，用来设置数据采集器和调理模块的文本信息，与硬件无关，只是用来对系统进行标识和说明，其参数包含每个通道的所有信息，如通道号、标识符、通道名称、传感器类型、放大倍数、单位、灵敏度等。

参数 la 为分配给该数据采集器的逻辑地址(0~255)，它必须与仪器的硬件设置保持一致，参数 query 决定初始化时是否向数据采集器发送仪器识别查询命令，参数 reset 为复位标志，决定初始化时是否向数据采集器发送复位命令，参数 tmo 为限时等级。参数 state 为数据采集器 FIFO 存储器的当前状态(如数据空、半满、全满、不空等情况)；参数 offset 为偏移地址，读/写目标寄存器的偏移地址或数据采集器 AD 转换器的端口地址；参数 data 为读出/写入目标寄存器的数据，len 表示一次性读出数据块的长度，单位为字(Word)。data 是一个整型数组，用来返回数据采集器的测量结果，其大小由通道数和 len 决定。

### 4.3.4 用于地震模拟试验的 VXI 自动测试系统

地震灾害是最严重的自然灾害之一，提高结构或建筑物的抗震能力、减轻震灾威胁是减灾工作的一个重点。作为地震工程研究的重要手段，某地震工程实验室的 5m×5m 地震模拟振动台最大荷载达 30t，满载加速度为 1g，可模拟实际记录的地震波实现台面同时在水平及垂直等 3 个方向的平动和绕 3 个主坐标轴的扭振。地震模拟试验通过强震观测记录的真实地震波激励振动台，观测置于地震台上的结构模型是按比例缩小的真实建筑物的破坏情况，据此研究地震破坏机理，改进抗震结构。

由于地震工程试验要求测试项目多，可靠性高，参数复杂，试验难度大，特别是实验内容千变万化，因此采用 VXI 总线的自动测试是十分合理的选择。系统结构如图 4-16 所示，该数据采集系统采用 GPIB-VXI 的外置主计算机方式，由标准台式 PC、GPIB 接口卡、GPIB 电缆、VXI 主机箱、GPIB-VXI/C 型 0 槽控制器、VXI 数据采集器以及多通道 VXI 调理模块等组成，可实现 64 通道数据采集任务。系统采用 GPIB-VXI/C 控制器 VXI9510 作为 VXI 0 槽控制器，开发的数据采集器 VXI9521 是寄存器基 VXI 仪器，其 A/D 转换分辨率为 12 位，精度为 0.3%，双通道 A/D 分别配有 32KB 的 FIFO(first in first out，先进先出)存储器。同时，对应每路 A/D 提供了 16 通道的多路开关控制信号，供与 VXI9521 相连的调理模块使用。同时还给出了一个 b1/b2 板选信号，在带两个调理模块时用来选择板 1 或板 2，以便该 VXI9521 与调理模块配合，实现 64 路模拟信号的数据采集。VXI9521 具有定时触发转换功能，其触发周期为 200μs~130ms。此外还具有软件触发和外触发功能。多路通道信号调理器 VXI9530 是根据地震工程试验的特点而专门设计的，它是寄存器基的 C 尺寸 VXI 仪器模块，每一个调理模块可实现 32 通道的信号调理。该模块支持信号隔直、可程控多路通道选择、可程控增益放大(放大倍数为 0.5、1、2、4、8、16、32、64、128，可选，线性度为 0.2%)、程控可选择低通滤波器，模拟电源及模拟信号与 VXI 总线隔离。实

验时将各关键部位传感器得到的振动信号送入专用的前置放大器(该放大器由输入电路、增益调节电路、滤波电路和输出电路构成)，其输出的信号再通过 VXI 信号调理模块，由 VXI 数据采集模件进行处理。

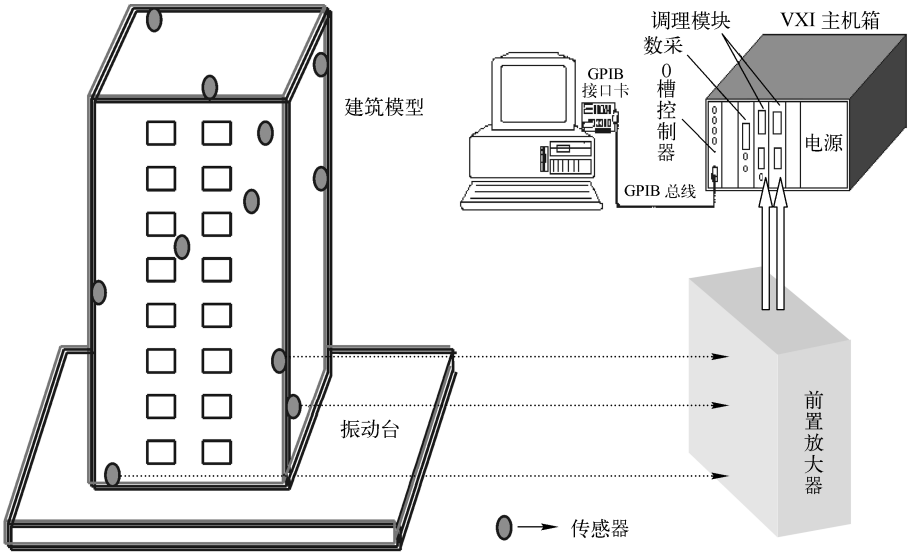



图 4-16　基于 VXI 的数据采集系统结构图

图 4-17 为本采集器的操作主面板，用户可以直接调用此操作面板来完成自己的数据采集任务，显示一些建筑结构模型在地震振动试验台上对地震波的响应曲线。

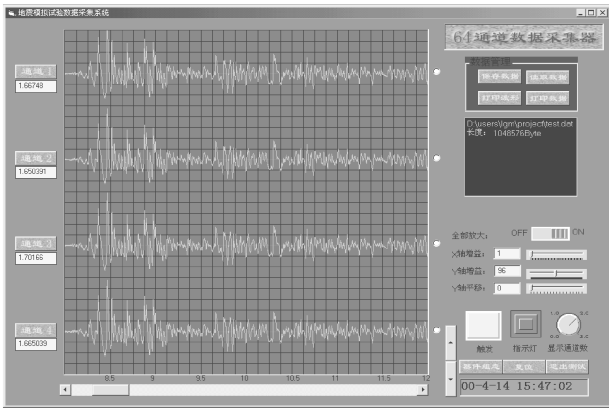



图 4-17　通用数据采集器操作面板

## 4.4　VXI 总线即插即用规范

为使 VXI 仪器模块更易使用，并使 VXI 总线系统具有一个真正开放的系统结构，1993 年，在 VXI 总线联合会的基础上进一步成立了 VXI 总线“即插即用”联盟，目的在于定义和推行一些标准化准则和操作规程，解决在 VXI 总线规范中尚未包含的系统及软件结构问题。这个

VXI“即插即用”规范称为 VPP(VXI Plug & Play),它补充和发展了 VXI 总线规范。

### 4.4.1 自下而上的 VISA 结构

对仪器系统 I/O 接口软件的研究由来已久,NI 公司的 NI-488 就是业内常用的 I/O 接口软件。这类传统的 I/O 接口软件采用了自顶向下的设计方法,首先列出需要 I/O 接口软件控制的各种仪器类型,然后列出了它们的各种功能,再对其中相同功能进行归纳,最后将统一后的功能函数与那些无法归纳的专用仪器功能合并在一起形成自顶向下结构的 I/O 接口软件库,以此实现不同类型仪器的互操作性与兼容性。然而,这种构成方法对于诸如中断处理、内存映射、接口配置、硬件触发等属于仪器器件特有的操作,难以建立起统一的 I/O 函数,对 VXI 消息基器件与寄存器基器件的操作也无法得到统一的方式。于是,自顶向下的设计方法在实现不同仪器类型之间软件的统一性问题上遇到了障碍。

为了实现仪器系统 I/O 接口软件的统一,VPP 提出了一种自底向上的解决方案。VISA 是虚拟仪器软件结构(Virtual Instrument Software Architecture)的缩写,核心就是一个 I/O 接口函数库及其规范,一般称为 VISA 库。与自顶向下的构成方式相比,VISA 的构成模型是从仪器操作本身开始的,它实现的统一性深入到操作功能之中,而不是停留于仪器类型之上。VISA 的结构模型如图 4-18 所示,自底向上形成了一个金字塔结构,最底层为资源管理器,其上依次为控制总线接口的 I/O 资源层、实现仪器驱动程序的仪器资源层和建立虚拟仪器的用户定义资源层,金字塔的顶部是用户利用 VISA 资源实现的应用程序。

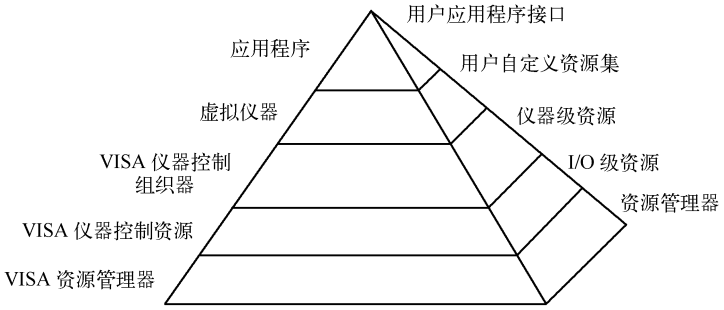



图 4-18 VISA 结构模型图

#### 1. VISA 资源管理器

VISA 资源管理器用于管理、控制与分配 VISA 资源的操作功能,各种操作功能包括资源寻址、资源创建与删除、资源属性的读取与修改、操作激活、事件报告、并发与存取控制、默认值设置等。

#### 2. 输入/输出(I/O)接口软件

输入/输出(I/O)接口软件存在于仪器与仪器驱动程序之间,是一个完成对仪器内部寄存器单元进行直接存取数据的 I/O 操作、对 VXI 总线背板与器件进行测试与控制、并为仪器与仪器驱动程序提供信息传递的底层软件层,是实现开放的、统一的虚拟仪器系统的基础与核心。在 VPP 系统规范中,详细规定了虚拟仪器系统输入/输出(I/O)接口软件的特点、组成、

内部结构与实现方法，并将符合 VPP 规范的虚拟仪器系统输入/输出接口软件定义为 VISA 软件。

#### 3. 仪器驱动程序

仪器驱动程序实质是为用户提供了一组用于仪器操作的功能函数集，每个仪器模块均有自己的仪器驱动程序，应用程序对仪器的操作一般是通过仪器驱动程序来实现的。对于应用程序设计人员来说，一旦有了仪器驱动程序，就可以在不了解仪器内部具体操作过程的情况下进行虚拟仪器系统的设计工作。仪器驱动程序成为连接上层应用软件与底层输入/输出软件的纽带和桥梁，这在 4.3 节对 VXI 主机软件的讨论中已经可以体会到。然而，以前仪器供应厂家为仪器模件配套提供的仪器驱动程序都是一个“黑匣子”，用户只能见到仪器驱动程序引出函数的原型，而无法获得源程序。用户即使已经发现供应厂家提供的仪器驱动程序不能完全符合使用要求，也无法对其进行修改。现在，VPP 规范明确定义了仪器驱动程序的组成结构与实现方式，也明确规定仪器生产厂家在提供仪器模块的同时必须提供仪器驱动程序的源程序文件和动态连接库(DLL)文件。事实上，由于 VPP 下仪器驱动程序是通过调用计算机 I/O 软件提供的 VISA 函数库来实现的，因此仪器驱动程序之间不仅可以相互参考，而且源程序本身也不难理解，用户可以对仪器功能进行扩展，从而掌握仪器使用的主动权。

#### 4. 用户定义资源层

用户定义资源层充分体现了 VISA 的灵活性和扩展性，人们可在前面两层的基础上通过增添某些数据分析和信号处理、设计更符合需要的人机对话界面来建立物理上原先并不存在的虚拟仪器。这样的虚拟仪器可以作为更理想的仪器设备资源提供给应用程序，也可作为普通的实用仪器独立运行。

#### 5. 应用软件开发环境

应用软件开发环境可因开发人员的喜好不同而有不同的选择，目前虚拟仪器系统的应用软件开发环境主要有两种：一种是基于传统的语言编程平台，如 NI 公司的 LabWindows、VB 下配置支持虚拟仪器功能的组件；另一种是基于图形化工程环境的平台，如 HP 公司的 HP VEE、NI 公司的 LabView 等。这些软件开发平台的提出，可以大大减轻系统开发人员的负担，使其能将主要精力集中投入到系统设计中，而不再忙于具体代码的实现细节。另外，根据计算机系统和操作系统的不同，VPP 联盟规定了多种系统框架。其中一些框架直接以操作系统命名，组建 VXI 测试系统时可以直接选择某种合适的系统框架后，再在此框架协议的指导下进一步选择必要的组成设备和软件。

### 4.4.2　VISA 的 I/O 控制函数集

基于自底向上结构模型的 VISA 创造了一个统一形式的 I/O 控制函数集，它是所有现存的 I/O 接口软件的功能超集，在形式上与其他 I/O 接口软件十分相似。这样，一方面对于初学者或是简单任务的设计者来说，VISA 提供了简单易用的控制函数集；另一方面对于复杂系统的组建者来说，VISA 提供了仪器间的兼容性与互操作性。下面通过调用 VISA 库函数分别

对 GPIB 与 VXI 消息基仪器进行简单读/写操作的例程，与 4.3 节调用普通 I/O 接口软件库进行比较，说明 VISA 的工作特点。

**例 4-1** 用 VISA 库函数实现对 GPIB 仪器仪器的读/写操作。

```
#include<visa.h>;#include<visatype.h>;
int main(void)
{
    ViSession DefaultRM,inst;                          //通道句柄
    ViChar d[LENGTH];                                  //响应返回变量
    ViUInt32 retCount;                                 //实际传送字节变量
    Vistatus st;                                       //返回状态变量
    st=viOpenDefaultRM(&DefaultRM);                    //打开默认的资源管理器
    st=viOpen(DefaultRM,"GPIB::1::INSTR",0,0,& inst);  //打开主地址为1的GPIB器件
    st=viWrite(inst,"*IDN?\n",6,&retCount);            //发送查询命令
    st=viRead(inst,d,LENGTH,&retCount);                //读回响应值
    st=viClose(inst);                                  //关闭器件
    st=viClose(DefaultRM);                             //关闭默认的资源管理器
    return 0;
}
```

**例 4-2** 用 VISA 库函数实现 VXI 仪器的读/写操作。

```
#include<visa.h>;#include<visatype.h>;
int main(void)
{
    ViSession viDefaultRM,inst;                        //通道句柄
    ViChar d[LENGTH];                                  //响应返回变量
    ViUInt32 retCount;                                 //实际传送字节变量
    Vistatus st;                                       //返回状态变量
    ViUInt16 devID;                                    //返回状态变量
    st=viOpenDefaultRM(&DefaultRM);                    //打开默认的资源管理器
    st=viOpen(DefaultRM,"VXI0::2::INSTR",0,0,& inst);  //打开逻辑地址为2的VXI器件
    st=viIn16(inst,"VI_A16_SPACE,0,&devID);            //读器件ID
    st=viOut16(inst,"VI_A16_SPACE,0,0x1234);           //写存储器操作
    st=viClose(inst);                                  //关闭器件
    st=viClose(viDefaultRM);                           //关闭默认的资源管理器
    return 0;
}
```

这里声明的数据类型均为 VISA 的数据类型，而 VISA 的数据类型与编程语言数据类型的对应关系包含在特定文件中(如 C 语言形式的头文件 visatype.h)。由于程序中没有涉及具体语言的数据类型，故程序本身具有良好的兼容性与可移植性，各种编程语言调用 VISA 的数据类型与操作函数的格式相差很少，无非是在开启部分进行消息基仪器的初始化，建立仪器与 VISA 库的通信关系；然后完成对消息基仪器的命令传送和读取响应数据；最后在仪器操作结束时需关闭仪器与 VISA 库的联系。这里对 GPIB 仪器初始化与对 VXI 消息基仪器的初始化相比，调用函数 ViOpen()的形式完全一致，差别只是输入参数中仪器的类型与地址。调用函数 viClose()则更无不同。而对 GPIB 仪器和 VXI 仪器的具体 I/O 操作，也只是需要带入初始化过程中返回的句柄作为仪器的标识参数，所调用的 VISA 函数也是同样的。

由此可见，VISA 库函数的调用与其他 I/O 接口软件库函数的调用在形式上并无太多不同，学习功能强大的 VISA 软件库并不比一般的 I/O 接口软件库任务繁重。同时，与其他 I/O 接口软件相比，VISA 体现了多种结构与类型的统一性，为虚拟仪器系统软件结构提供了坚实的基础。

VISA 的 I/O 接口软件库的源程序具有唯一性，通过提供不同的 API 接口，适用于不同的操作系统与编程语言环境。在不同的编程语言环境之中调用 VISA 库，均需在应用程序头部引入说明文件。在 C 语言环境下，VISA 资源说明文件为 visatype. h 和 visa. h 文件。这样，同一 VISA 源程序通过不同的引入接口与文件说明，就可以适应不同的工作环境。同时，VISA 资源描述格式不仅适用于现行 VISA 库包含的所有资源，也为 VISA 库将来的资源扩充定义了一个标准格式。

### 4.4.3　仪器驱动程序的接口模型

VXI 总线即插即用仪器驱动程序的作用与普通设备驱动程序并无不同，无非是通过调用底层的总线控制函数来实现特定仪器的仪器功能，形成支持上层应用软件的仪器驱动函数。但 VPP 将其内部实现的结构和外部调用的方式也纳入了标准化，图 4-19 给出了 VPP 通用仪器驱动程序的结构模型，它由 5 个主要部分组成。

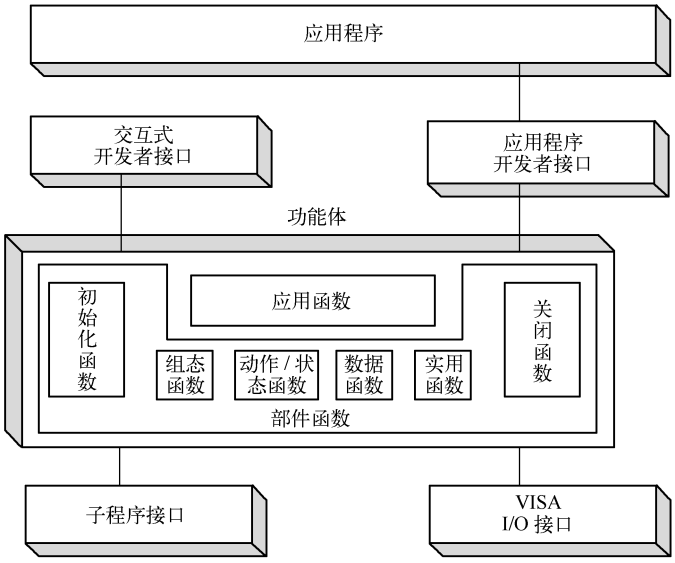



图 4-19　仪器驱动程序外部接口模型

功能体为仪器驱动程序的实际代码，是仪器驱动程序的主体。在 VPP 规范中，定义了两种源代码形式，一种为语言代码形式，主要是 C 语言形式；另一种是在图形化开发环境下采用的 G 语言形式。功能体内部包含部件函数和应用函数两层。作为底层的部件函数，它们是控制仪器特定功能的一些软件模块，分别实现初始化和关闭连接的功能，也包括组态、动作/状态、数据处理，以及其他各种实用函数功能。作为上层的应用函数，它们用来完成面向应用的具体仪器操作

I/O 接口解决了仪器驱动程序与底层仪器总线控制之间的通信问题。在 VPP 系统中，仪

器驱动程序通过调用 VISA 这种通用的 I/O 接口软件库，实现不同类型的仪器驱动程序在同一软件平台上的运行与调试。子程序接口提供了调用其他软件模块及软件库函数的机制(如数值计算、字串处理等)。应用程序开发者接口描述了从高层软件程序调用驱动程序的机制。对于不同的应用程序开发环境，仪器驱动程序将提供不同的软件接口，以使应用程序可以方便地调用仪器驱动程序中定义的所有功能函数。交互式开发者接口用于帮助软件开发人员理解每个仪器驱动程序函数的功能，也有助于理解应用程序开发者接口的函数功能。这个接口通常是一个图形化的功能面板，用户可以在这个图形接口上操作仪器的各种控制功能，改变每一个功能调用的参数值。VPP 规范要求所有仪器驱动程序都必须包括相应的功能面板文件(.fp 文件)。

### 4.4.4 VPP 软面板

作为一台独立的仪器，必须要有一个与用户交换信息的人机交互面板，既要通过面板上的按钮、开关、旋钮等操作器件来控制和管理仪器，又要通过面板上的指针表头、数码管、示波屏等显示器件来指示测试结果。对于安放在 VXI 机箱中的 VXI 仪器，其有限宽度的实际面板不可能配置这些必要的功能。于是，作为虚拟仪器的一个重要特点，软面板借助计算机强大的屏幕显示功能全面地代替了传统仪器的实物操作面板。VPP 为此做出专门的规定(VPP7)，对与 VXI 设备配套的仪器软面板的设计也提出了规范化的要求，并将符合这种要求的软面板称为 VPP 软面板。

VPP 软面板作为相应仪器的配套软件，需要以独立的可执行文件方式运行。这样，只要有计算机操作系统和 VISA 函数库的支持就可以独立运行，实现对仪器的独立操作。VPP 软面板通常由主面板和副面板组成，前者作为主要的用户界面，应当始终打开；后者则由前者调用，可随时退回主面板。主面板的标题部分可选择明显的字体、字号和颜色，标题应完整地描述仪器，如 VXI 设备的名称和型号；也应表现主要的工作状态，如 VXI 设备的逻辑地址、所在机箱中的槽位号和连接状态等。VPP 软面板需要一个连接指示器，指明其与相应 VXI 设备的连接状态。例如连接成功时可选择绿色指示，否则保持背景底色。同时，主面板上也需要包含 VXI plug&play 图标和形象地表达仪器型号和开发者的图标，且应注意必须支持版本说明和退出工作这两个最基本的操作按钮。另外，在主面板的应用区应根据仪器设备本身的功能需要，配置各种操作和显示组件来实现用户和设备之间的人机对话。

VPP 软面板与自动测试系统的应用程序分享整个计算机屏幕，在不影响自动测试系统应用软件协调各 VXI 设备，完成复杂自动测试任务的同时，也允许用户通过软面板单独使用每个 VXI 设备本身。为此要求 VPP 软面板能够随意改变窗口的位置和大小，并能在自动测试系统的应用程序与各 VXI 设备的 VPP 软面板之间方便地切换。

VXI 仪器的 VPP 软面板应由 VXI 仪器开发商配套提供。依靠这个软面板，用户可以直接测试 VXI 仪器本身的操作功能和通信功能，也可以像使用普通的传统独立仪器那样直接进行测试工作，还可以用来了解各种有关的仪器信息资料，从而提高组建自动测试系统的工作效率。

### 4.4.5 VISA 资源描述和资源管理器

定义在资源类下的各种资源是 VISA 中最基本的概念，VISA 资源由属性集、事件集与操

作集 3 个要素组成。规范对各种 VISA 资源的描述均按资源概述、资源属性表及属性描述、资源事件集、资源操作集的顺序依次进行。

这里资源概述应简要说明资源所要实现的功能，资源属性表列出了资源所有属性，并在属性描述中按属性符号名(用于标识属性)、存取特性(包括读/写特性和局部/全局特性)、数据类型、取值范围做出了详细说明。读/写特性可选择 R/W(可读可写)与 RO(只读)，局部/全局特性可选择 Local 或 Global，VISA 的数据类型独立于编程语言定义，在具体调用 VISA 函数进行实际操作时再映射到具体的数据类型。

资源事件集用于定义资源事件。作为一种运行机制，VISA 事件用于在某种条件下通知应用程序一种行为的发生，如 GPIB 器件的 SRQ 请求、VXI 的 SysFail 线有效响应等硬件状态，以及其他资源行为的起始与结束、操作状态转换和错误发生等。VISA 定义了两种事件处理机制：一种是事件排队方式，它先允许特定事件的处理机制作用，然后在一定时刻去查询事件是否已经发生；另一种事件处理方式为回调方式，即中断方式，它先安装回调句柄，然后在程序发生事件时处理机制自动转入相应的回调子程序。

资源操作集包括一系列 VISA 资源操作，而对每个操作的描述包括名称(操作函数的原型描述)、目标(相应的操作功能)、参数表(包括参数名、输入/输出方向、参数数据类型及参数描述)、返回状态值、功能描述、可参考的其他相关操作和操作实现时应满足的一些条件与备注。

VISA 资源类包括仪器控制资源(INSTR)、存储器访问资源(MEMACC)、GPIB 总线接口资源(INTFC)、VXI 主机箱背板资源(BACKPLANE)、从者器件侧资源(SERVANT)和套接字资源(SOCKET)等，其中 VXI 系统必须支持的 INSTR 资源包括基本 I/O 服务、格式化 I/O 服务、存储器 I/O 服务和共享存储器服务等几个方面。这里特别强调的基本 I/O 服务包括读服务、写服务、触发服务、状态/请求服务和器件清除服务，依靠这些服务资源可以方便地实现自动测试的各种基本要求。

从例 4-1 和例 4-2 所示的例程中可以看出，VISA 资源管理器本身也是资源，它从 VISA 资源模板中获取接口，并与所有在其内部注册的资源进行链接，从而为应用程序提供资源查找和访问服务。应用程序通过 viOpenDefaultRM( ) 函数访问默认的资源管理器，获取通话标识后即可通过 viOpen( ) 函数建立与特定资源的访问链路。viOpen( ) 函数的关键参数是需指出表达访问资源的描述字符串，除了例程中给出的注释外，下列更一般的示例对于直观理解描述字符串的形式也很有意义：

```
GPIB-VXI0::3::INSTR        在 GPIB-VXI 控制的 VXI 系统中,逻辑地址为 3 的 VXI 器件
GPIB::1::0::INSTR                GPIB 设备,其主地址为 1,副地址为 0
USB0::1689::832::c012378::0::INSTR USB 连接的 Tek 函数发生器 AFG3021 的仪器资源
TCPIP0:1.2.3.4::999::SOCKET    指定地址端口的 TCP/IP 访问
```

为了便于资源的查找，VISA 资源管理器提供了如下两个服务函数：

```
viFindRsrc(sesn,expr,findList,retcnt,instrDesc)
viFindNext(findList,instrDesc)
```

这里，前者可用正则表达式作为参数 expr，从参数 instrDesc 中查找与其匹配的第一个资源并放在参数 findList 指示的列表中；后者则可给出资源列表中的下一个资源。正则表达式

采用“?”、“*”等通配符来指出待查资源的形式，如表达式 GPIB-VXI0::?::*INSTR 表示 GPIB-VXI0 所连 VXI 模件的所有 INSTR 资源。

各种 VISA 资源的处理和应用最终都是利用 VISA 软件提供的函数来完成的，由于 VISA 软件的标准化结构，在此基础上学习和使用都很方便。

## 4.5 软件开发平台 LabWindows/CVI

LabWindows/CVI 由 NI 公司推出，是一个面向自动测试和虚拟仪器的语言编程软件开发平台。它以 C 语言为基础，故称 CVI(C for Virtual Instrument)。这个软件开发平台一方面提供了丰富的虚拟仪器面板控件，另一方面提供了十分全面的功能函数库(包括面向总线接口、仪器驱动、分析计算和各种 PC 通用功能)，从而使自动测试系统的软件开发更为简便。本节将首先介绍 LabWindows/CVI 的编程环境和编程方式，然后以一个 VXI 虚拟仪器的应用程序开发为例，重点说明利用此编程软件开发平台控制 VXI 仪器的实现方法。

### 4.5.1 编程环境和编程方式

开发环境的主界面如图 4-20 所示。在 LabWindows/CVI 编程环境下开发一个应用程序，需要首先建立一个工程项目文件(.prj)，它将指出构成该项目的用户界面文件(.uir)和 C 语言编写的源文件(.c)，其中 C 源文件中还可包含某些头文件(.h)，用于定义系统的某些公用数据结构和库函数原型。于是，在相应开发窗口的支持下建立和调试这些组成文件就成为开发过程的主要任务，整个开发过程的有关参数被自动存放在文件(.cws)中。

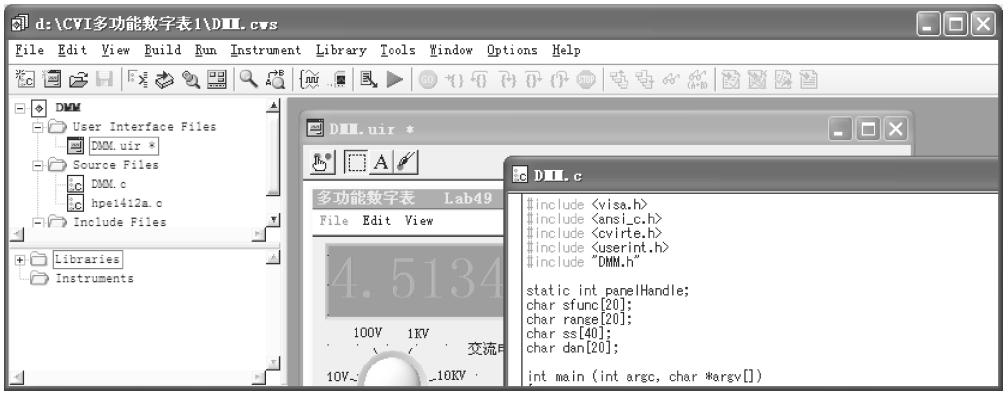



图 4-20 LabWindows/CVI 的工程文件窗口

#### 1. 工程项目的建立

LabWindows/CVI 启动后将在其开发平台界面左上角的项目管理器展示(.prj)文件的内容，并可通过主菜单选项打开或新建一个 .prj 文件。如选择 File→New→project(*.prj)，弹出一个空的用户界面编辑窗口，且默认的项目文件名为 Untitled.prj；也可通过主菜单选项 Edit→Add File To Project 向一个打开的 .prj 文件添加(或删除)组成工程的各种文件。

### 2. 用户界面的建立

通常用于自动测试系统的应用程序开发总是从用户界面开始，可通过下拉菜单选择 File→New→User Interface( * uir)得到一个名为 Untitled. uir 的用户界面新文件。或在文件列表窗口中直接单击需要编辑的已有的 . uir 文件，便可弹出如图 4-21 所示的用户界面编辑窗口。在此窗口下通过单击选项 Create→Panel 可创建一个空面板，然后双击其空白处，弹出面板(Panel)属性对话框后便可根据需要修改各属性内容，如将 Panel Title 换成新的标题名，并单击 OK 按钮退出。

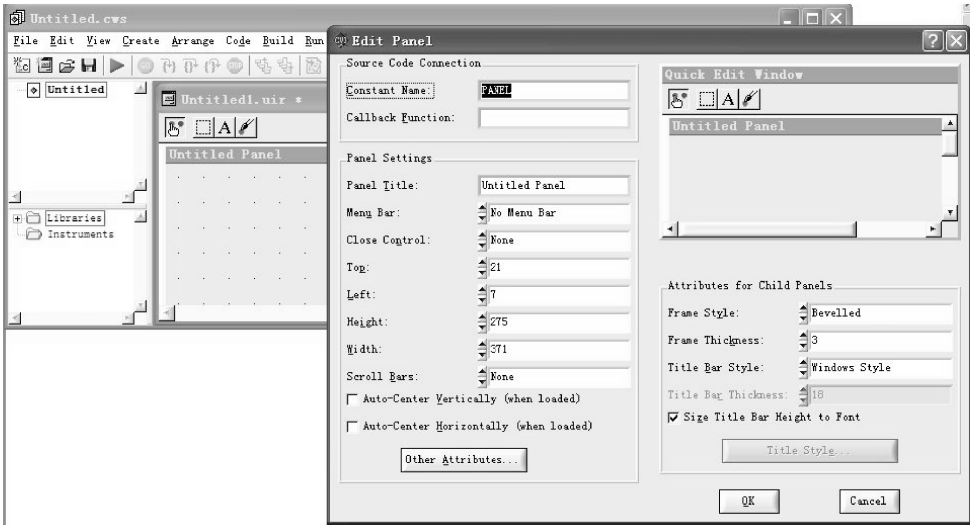



图 4-21　用户面板的建立和配置

为在面板 Panel 上添加控件和菜单，可在所选 Panel 的空白处单击鼠标右键，即可弹出如图 4-22 所示的控件库选择窗口，进而展示该库所包含的各种控件供用户选用。LabWindows/CVI 提供了很多用于虚拟仪器面板的控件(从按钮到仪表盘、开关、旋钮等)。

如果在控件库选择菜单及其子菜单中选择 Command Button→Square Command Button 选项，就会在面板上建立一个按钮控件。这样，可用鼠标选中这个控件后将其拖到面板中希望的位置，并设置该控件的大小。也可双击该控件，弹出编辑对话框来配置其参数。如在编辑对话框中可以通过 Source Code Connection→Constant Name 来设置该按钮的资源定义名(如“COMMANDBUTTON”)，通过 Source Code Connection→Callback Function 来建立该按钮所对应的回调函数名(如“CCallback”)，通过 Lablel Apperance→Label 来设置该按钮的标示内容(如“开始测量”)。最后单击 OK 按钮，完成控件的配置过程。

又如在面板上单击鼠标右键，再在随之出现的快捷菜单中选择 Ring→Knob 控件，然后双击出现在面板上的控件便会弹出控件的属性编辑对话框。将控件的属性 Constant Name 设为 R1，将 Callback Function 设为 R1_Callback，然后确认退出。

如图 4-23 所示，在用户窗口的面板上完成所有控件的布局和配置后，可以通过下拉主菜单 Code→Generate→All Code 自动产生程序清单中某些重要的代码结构，如主函数、回调函数等。

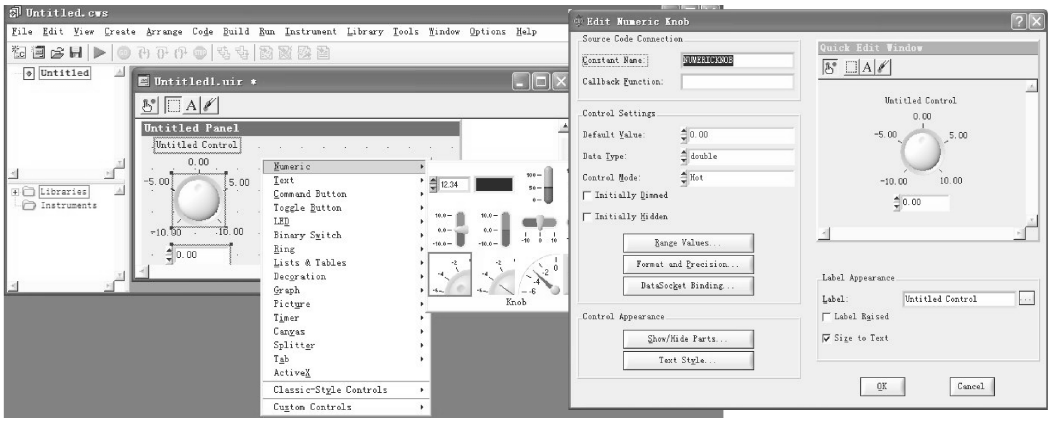



图 4-22 控件的选用和配置

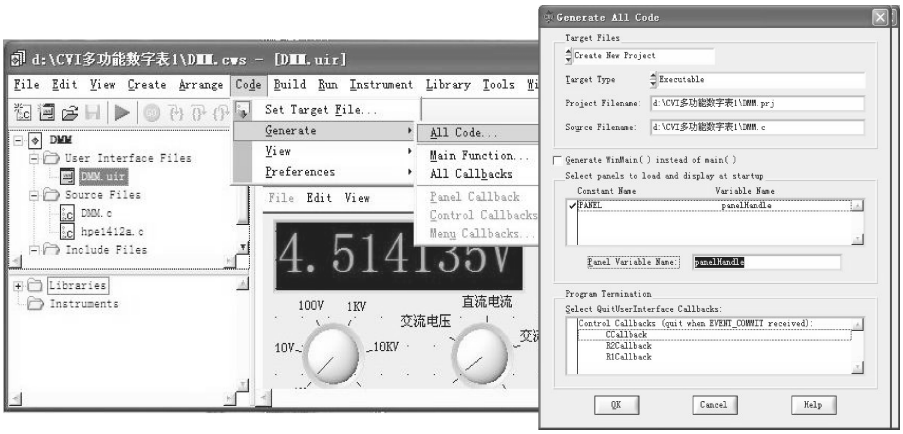



图 4-23 代码自动生成与回调函数响应事件窗口

### 3. 源代码编辑

在项目文件管理窗口下选择 File→New→Source(*.c)…，或用鼠标在文件列表框中直接单击需要编辑的 .c 文件，可弹出如图 4-24 所示的源程序代码窗口。

为了支持针对仪器的功能，LabWindows/CVI 主要通过 VXI 库函数、GPIB 库函数、RS-232 库函数及 VISA 库函数来实现其对仪器总线的操作；同时也可以直接调用仪器驱动程序中的各种功能函数。LabWindows/CVI 也支持用于信号分析和数据处理的函数调用，以及诸如 TCP/IP、DDE 等现代 PC 的各种通用服务功能。LabWindows/CVI 能使开发者在程序开发过程中方便地分类查找并通过窗口选择需要的参数，从而摆脱记忆烦琐的函数名称与参数的麻烦。加载 VXI 模件的仪器驱动程序可通过菜单选项 Instrument→Load 实现，并通过菜单选项 Edit→Add Files to Project→Instrument(*.fp)将该仪器驱动程序的函数面板文件(.fp)加入到项目文件列表栏中。进而通过双击打开这个函数面板选择树窗口，选择适当的函数来操作 VXI 设备。如在主窗口下选择菜单选项 Library→User Interface，就可以弹出相应函数库的函数选择窗口，选择需要的函数后会自动弹出对应该函数的函数面板窗口，该窗口支持这个函

数本身及其参数的在线帮助说明，便于用户选择参数时参考。

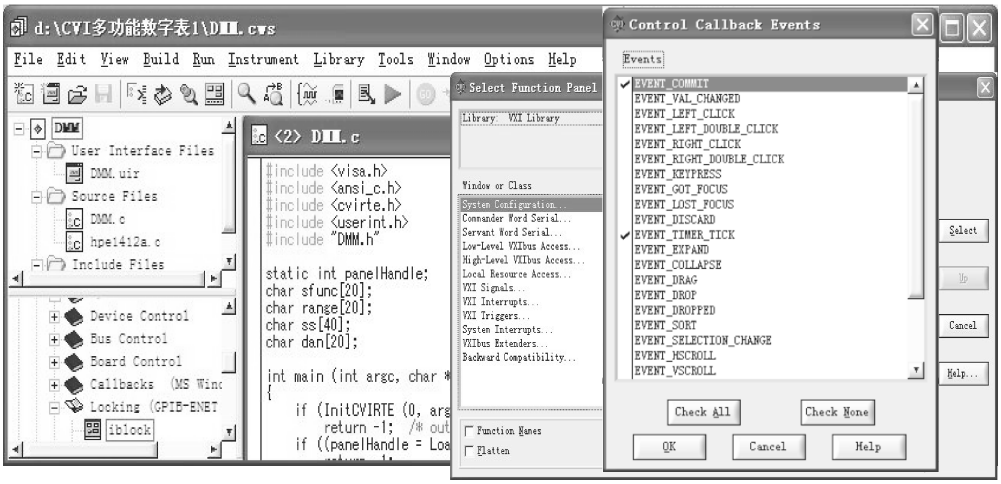



图 4-24　源程序代码窗口和函数调用窗口

#### 4. 其他功能

除了通过 Edit 菜单中的选项为源代码窗口提供文本删除、复制等编辑功能外，也可用 View 菜单中的选项查看和调用各种库函数的结构和内容，用 Tools 菜单中的选项创建仪器驱动程序。同时，可通过 Windows、Option 和 Help 等菜单中的选项设置各种窗口参数、编译条件和参数、各类文件和目录的路径、开发环境本身的各种参数，以及有关帮助信息的获取。在具体研究开发过程中，在选中对象后单击快捷键 F1 可以调出有关说明资料和调用方法，这对深入理解 CVI 至关重要，应不断实践体会。另外，可利用 Instument 菜单中的选项来加载、卸载、编辑仪器文件(.fp)，用 Build 菜单下的选项来实现编译、链接、生成软件包等功能，用 Run 菜单中的选项来完成启动、单步、断点等运行调试功能。这些功能的使用方法与采用普通 C 语言开发系统的做法基本相同，这里不再赘述。

### 4.5.2　编程示例

为了解释如何使用 LabWindows/CVI 开发测试系统软件，这里组建的系统包括 Agilent 的 VXI 机箱 E1421B、VXI 零槽控制器 E8491B、消息基的多功能数字表 E1412，PC 内的接口卡为 Texas Instruments OHCI Compliant IEEE 1394 Host Controller，软件使用 NI 公司的 LabWindows/CVI 开发环境。

为此首先根据前面介绍的 CVI 基本开发过程，依次建立 dmm.prj、dmm.uir 和 dmm.c 文件。这里，dmm.uir 中只建立一个面板(Panel)，其中文本框控件 Tx1 用于显示测试结果；旋钮控件 R1 用于选择仪器功能方式(如 DC 电压、AC 电压、DC 电流、AC 电流和电阻)；旋钮控件 R2 用于选择量程；按钮控件 C1 用于触发操作。

考虑 dmm.c 程序的主函数可由开发平台自动生成，因此所需开发的实际程序代码就是如何响应两个旋钮控件和一个按钮控件的回调函数。这里在获悉用户通过旋钮表达其对“功能模式”和“量程范围”的要求后并未立即设置仪器，而只是将其记录下来，在控件回调函数

的运行过程中再形成相应的命令字串。而当“启动测量”按钮的动作发生时才真正执行 VXI 仪器动作，即在该按钮控件的响应回调函数中添加操作 VXI 设备的程序代码。具体程序如程序清单 4-2 所示，下面将着重说明如何选择功能和量程，如何启动测试过程，如何读取测试数据并显示结果。

**程序清单 4-2　多功能数字仪表 DMM 的程序清单**

```
#include<visa.h>
#include<ansi_c.h>
#include<cvirte.h>
#include<userint.h>
#include "DMM.h"
static int panelHandle;char sfunc[20],range[20],ss[40],dan[20];
int main(int argc,char * argv[]){
     if(InitCVIRTE(0,argv,0) == 0)  return -1;  /* out of memory * /
     if((panelHandle = LoadPanel(0,"DMM.uir",PANEL))< 0)  return -1;
     DisplayPanel(panelHandle);RunUserInterface();DiscardPanel(panelHan-
dle);
     return 0;
}
int CVICALLBACK CCallback(int panel,int control,int event,void * callbackData,
       int eventData1,int eventData2){
   char buf[512] = {0};ViStatus err;ViSession defaultRM;ViSession dmm;
   switch(event){
     case EVENT_COMMIT:                          //注释①
         viOpenDefaultRM(&defaultRM);
         viOpen(defaultRM,"VXI0::8::INSTR",VI_NULL,VI_NULL,&dmm);
         viSetAttribute(dmm,VI_ATTR_TMO_VALUE,10000);
         err = viPrintf(dmm," * RST \n");
         sprintf(ss,"% s% sDEF \n",sfunc,range);viPrintf(dmm,ss);
         viScanf(dmm,"% t",buf);SetCtrlVal(panel,PANEL_STRING,strcat(buf,
dan));
         viClose(dmm);viClose(defaultRM); break;
  } return 0;
}
int CVICALLBACK R2Callback(int panel,int control,int event,
         void * callbackData,int eventData1,int eventData2){
   int i;
   switch(event){
     case EVENT_COMMIT:
          GetCtrlVal(panel,PANEL_RINGDIAL_2,&i);sprintf(range,"% d",i);
break;
         //…
   } return 0;
}
int CVICALLBACK R1Callback(int panel,int control,int event,void * callback-
Data,
       int eventData1,int eventData2){
   int i;
```

```
    switch(event){
      case EVENT_COMMIT:
          GetCtrlVal(panel,PANEL_RINGDIAL,&i);
          switch(i){
              case 1: sprintf(sfunc,"MEAS: VOLT: DC?");sprintf(dan,"V");break;
              case 2: sprintf(sfunc,"MEAS: VOLT: AC?");sprintf(dan,"V");break;
              case 3: sprintf(sfunc,"MEAS: CURR: DC?");sprintf(dan,"A");break;
              //…
          } break;
       //…
} return 0;
```

由于一般 VXI 设备均支持 VPP，因此通常对 VXI 设备的操作并不需要直接调用 VXI 总线控制函数，更不需要直接访问主机的 GPIB 接口，所做的工作只是在响应“启动测量”按钮 C1 的回调函数中，集中使用 VISA 函数来操作总线。即如注释①部分所示，先打开 Visa 资源管理器，再打开 Visa 的总线设备接口，然后依次发送复位命令、程控命令和触发命令，随后等待和接收返回的数据，并显示到面板。最后，关闭 Visa 设备和 Visa 资源管理器。这里，Visa 函数直截了当，所发命令也已标准化(SCPI)，因此成功编程的关键在于正确指出仪器的资源描述字符串，这里是"VXI0::8::INSTR"，可通过 VXI 逻辑地址计算得到，也可直接通过资源管理器搜索获取。

代码编译后，可以通过主界面菜单选项 Run→Debug 来运行调试，弹出如图 4-25 所示的虚拟仪器界面，单击“开始测量”按钮后就会看到 String 控件显示的测试结果。




图 4-25　VXI 虚拟仪器操作界面

按照 VPP 的规定，VXI 设备出售时应提供相应的仪器驱动程序，且应提供源代码和虚拟仪器面板，因此组建 VXI 系统时也可不直接调用面向总线的 Visa 函数，而是调用设备驱动程序所提供的面向仪器的功能函数。显然，这种仪器操作函数更贴近应用，使用起来更加简便。

为了使用 HP 公司的模件 E1412a 的仪器驱动程序，应用程序应首先连接其面板文件 E1412a.fp，并将模件的连接状态、机箱槽号和逻辑地址显示在用户界面中。作为示例，这里仅将程序清单 4-2 中具体操作 VXI 设备的一段 Visa 程序换成下面用仪器驱动函数实现的等价代码：

```
hpe1412a_init("VXI0::8::INSTR",0,0,dmm);          //初始化,得到仪器句柄
```

```
hpe1412a_confDcVolt(dmm,atoi(range),0,2,0.00001,VI_ON,VI_ON)); //DC 电压
hpe1412a_confTrigger(dmm,1,10,0,100,1);        //触发测试
hpe1412a_readMeas(dmm,buf);                    //读回数据
hpe1412a_close(dmm);                           //关闭仪器
```

其中，参数 dmm 为仪器句柄，通过初始化函数获取后即可作为访问该仪器的标识进行各种操作了，即按配置(功能和量程)、触发、读数的顺序分别调用相应的函数。

### 4.5.3 IVI 可互换仪器驱动程序的概念

IVI 的目标是要建立一种可互换式的仪器驱动程序框架结构，从而允许系统更换同类仪器时无须更换应用程序和驱动软件，从而节省测试系统开发、维护的开销。

IVI 将通用仪器分为 IviDmm(万用表)、IviScope(示波器)、IviGen(波形发生器)、IviSwitch(通道开关)、IviPower(程控电源)等 5 类驱动程序，并在此基础上分别定义了它们的标准应用接口(API 函数)，用户可以通过这个 API 函数开发独立于具体型号仪器的测试应用程序。API 函数可以分为基本操作和扩展操作两类，前者描述了仪器的基本属性和通用操作，后者则用来描述特定型号仪器的专用属性和功能。如在数字万用表类规范 IviDmm 中，定义了测量直/交流电压、电流、电阻、温度、频率、电量等通用功能，基本操作包括初始化、设置功能、量程、精度、触发参数及返回测量值等，而扩展功能则可能包括自动量程设置、自动校零、多点扫描等。

IVI 提供的仿真驱动程序可供用户在没有实际具体设备的情况下开发含有该类仪器功能的应用程序，并在执行用户测试程序的过程中返回仿真得到的测试数据结果。

在应用 IVI 技术组建自动测试系统时，对不同型号的具体仪器应配置特定的仪器驱动程序，IVI 仪器类驱动程序通过特定型号仪器驱动程序与特定型号的仪器通信，这样实际更换仪器时只需更换特定仪器驱动程序，而不需要改变测试应用程序对仪器类驱动程序的调用。通常像 LabWindows/CVI 这样的仪器开发平台提供的 IVI 驱动程序库会为一些著名公司的通用仪器提供 IVI 仪器驱动程序，而对不在 IVI 驱动仪器库中的仪器，可以通过 LabWindows/CVI 的 IVI 驱动程序开发工具建立 IVI 驱动程序，然后加入所用的 IVI 驱动仪器库中，并修改 IVI. INI 文件。

### 4.5.4 独立发布软件的生成

在 LabWindows/CVI 环境下生成的可执行程序需要相应的 LabWindows/CVI 驱动模块支持才能运行。为使开发的应用程序能在并未安装 LabWindows/CVI 的计算机上运行，可以利用 LabWindows/CVI 提供的软件发布功能，生成实际应用软件的安装程序，即将应用程序的可执行代码本身与执行该代码所需要的 LabWindows/CVI 驱动模块打包后同时发布，从而形成能够独立运行的完整应用软件包。

在 LabWindows/CVI 的项目文件管理窗口下单击菜单选项 Build→Create Distribution Kit，就可以弹出发布程序的创建窗口，开发安装发布程序。也可先生成应用工程的可执行文件，然后通过主界面菜单 Build→Distributions→Manage Distributions…，弹出 Manage Distributions 对话框。如图 4-26 所示，单击“New…”按钮后便弹出 New Distribution 对话框，而且会自动生成“Name”和“Settings file”设置。

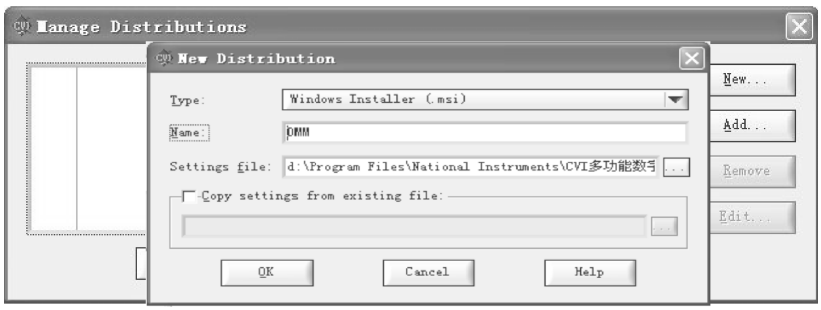



图 4-26　分发对话框

单击“OK”按钮，弹出“Edit Installer”对话框。如图 4-27 所示，在 Application Name 处填入生成应用软件的名称，在 Output Directory 处填入软件包的存储路径，其余可选默认值。单击“OK”按钮(在图 4-27 中未显示出来)便回到分发对话框。再单击“OK”按钮，回到主界面后选择菜单选项 Build→Distributions→Build DMM 便弹出一个 Build Distribution DMM 对话框，开始软件包生成的运行过程，直至完成后单击“Close”按钮，已建立的安装文件便可独立发布了。

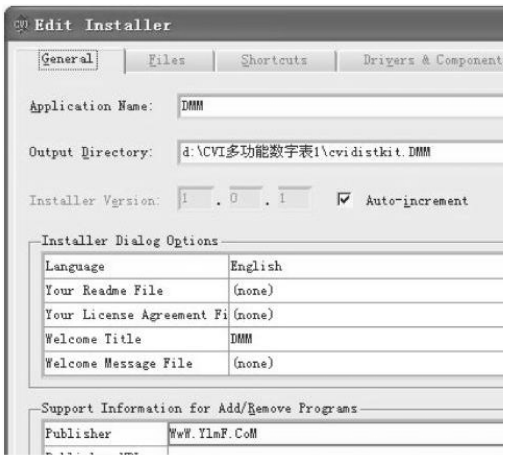



图 4-27　安装对话框

## 本 章 小 结

VXI 系统以 VME 总线为基础，针对仪器系统对同步和触发等功能的专门需要，扩展形成了新一代仪用总线标准，体现了仪器系统高速化和小型化的发展趋势。数据传输作为 VXI 的基本功能，是了解 VXI 实际工作过程的关键。“即插即用”规范 VPP 解决了 VXI 总线规范中尚未包含的系统及软件结构的问题，其中 VISA 提供了统一的标准 I/O 接口，在此基础上建立的设备驱动程序则与设备本身配套成为支持设备操作的基本方式，专门的软件开发平台和 IVI 技术为自动测试系统的开发提供了更为方便的环境。在学习 VXI 技术的过程中，需要特别注意在 VXI 总线控制、VXI 0 槽控制器、VXI/GPIB-C 转换、主机软件的资源管理、I/O 控制的 VISA 函数库、设备驱动程序、虚拟仪器和应用系统软件之间建立清晰的层次关系。

# 参 考 文 献

[1] 马怀俭, 孙愚蒙等. VXI 总线讲座. 电测与仪表, 1993 增刊.
[2] 陈光踽. VXI 总线测试平台技术. 电子科技大学出版社, 1996.
[3] IEEE Standard For VMEbus Extensions For Instrumentation.VXIbus-IEEEstd-1155, 1993.
[4] IEEE Standard For VPP. 2001.
[5] 林茂六, 王丽, 付平. VXI 总线雷达自动测试系统集成和模块设计范例. 哈尔滨工业大学出版社, 2000.
[6] 张毅刚, 彭喜元, 姜守达, 付平. 自动测试系统. 哈尔滨工业大学出版社, 2001.
[7] 赵会兵. 虚拟仪器技术规范与系统集成. 清华大学出版社, 2003.
[8] 宋宇峰等. Lab Windows/CVI 逐步深入与开发实例. 机械工业出版社, 2003.
[9] 冯玉贵, 刘思久. VXI 主模块接口逻辑实现. 全国第二届总线技术与控制系统工程学术报告会, 2001.
[10] 刘思久, 卢光明等. ATS95 中的 VXI 资源管理操作. 电测与仪表, 2000(02).
[11] 刘思久, 卢光明等. VXI 虚拟仪器的建立和 ATS95 编程工具. 电测与仪表, 1999(09).
[12] 李行善, 左毅. 自动测试系统的集成技术. 电子工业出版社, 2006.
[13] 林海军, 郭松林, 张礼勇, 徐文德.基于 VXI 总线的地震工程模拟实验自动测试系统.电测与仪表, 2002(09).
[14] 王滨, 张礼勇, 陈德鸿.VXI 总线多功能数据采集模块的研究.电测与仪表, 2001(01).
[15] 王彦丽, 童子权, 张礼勇等.四通道 VXI 高速数据采集模块仪器驱动程序的设计.哈尔滨理工大学学报, 2002(01).
[16] 曾杰, 田书林.VXI 总线中数据快速传输的实现.中国测试技术, 2007.7.
[17] 黄星, 栗冰.基于虚拟仪器技术的 VXI 总线测控系统研究与设计. 航天控制, 2006.4.

# 第 5 章　测试系统的远程操作

自动测试技术的发展已从早期的实验室技术走向实际现场，特别是现代信息技术的高速发展给仪器之间的远距离通信提供了十分有利的条件，远程监测和控制已经成为自动测试系统的一项基本功能。根据远程监控系统的实际需要，传感器和数据采集设备可以分散安置在远离主机或靠近被测对象附近，而显示和存储数据的主机则可放在远方用户的身旁或手中。本章将从串行通信的基本原理开始讨论，通过 RS-232 接口、调制解调器设备、网络传输和移动通信业务等实用技术的阐述，说明如何利用 PC 和嵌入式设备建立远程自动测试系统的方法。

## 5.1　串行通信和 RS-232 协议

串行通信使用单根或双根信号线传送数据，减少了通信电缆的路数，不仅节省了线路的硬件开销，也简化了处理电磁干扰的复杂程度。RS-232 接口是应用最广泛的一种串行通信接口标准，其协议直接支持的调制解调器工作方式可以利用现存的通信媒体(如电话网络、电报信道)实现两地之间的远程通信，使测试系统能够顺利地扩展到远程操作的领域。同时，由于自动化测控领域相当多的设备和仪器都已直接或间接(如通过 RS-485、RS-422 转换器)地配备了这种接口，因此掌握这种经济实用的通信技术对于组建自动测试系统和实现虚拟仪器设备都是很有意义的。

### 5.1.1　串行通信的基本概念和分类

串行通信信号线配置分为单工、全双工和半双工几种形式。这里，单工是指通信数据只能沿一个方向传送，而全双工可以在两个方向同时完成数据的传送，半双工通过开关切换线路来实现收发过程的轮流工作。全双工配置的成本最高，但功能最强，是实际应用中主要采用的形式。

串行通信可分为异步和同步两种基本工作方式。异步通信方式面向字符传送，被传送的字符由一组二进制数据位(bit)构成，并由附加的起始位和停止位将其打包。当接收设备收到起始位信息后开始定时采样传输线上的信号，依次接收发送设备发来的二进制数据位、校验位和停止位，然后进入等待状态继续监视传输线，直至遇到下一个字符的起始位时再启动同样的过程。由于传送一个字符的累计时间有限，这种一个个字符的再同步方式允许发送器时钟频率与接收器时钟频率之间存在一些误差，从而降低了对硬件设备的要求。串行同步通信方式是面向成块数据传送的，其同步信号在局部范围传送时通常可由另一根专用的时钟线提供。同步规程可以面向字符，但更多的情况是直接面向比特(位)流。由于这种方式消除了

为每个字符所附加的同步位，因此工作效率(有效数据位与总传送数据位之比)要比异步方式高，速度也更快。本书前面介绍的 $I^2C$ 和 SPI 总线属于同步通信方式，而本节将要讨论的 UART 协议和 RS-232 标准则属于异步通信方式。

信号传输方式分成基波传输和载波传输两种。基波传输方式又称基带传输，即按信号的原样直接传输，显然要求线路带宽高于信号带宽；载波传输则可通过调制解调器传输高于线路带宽的信号。数据传输率是指单位时间内传送的信息量，以每秒传输的二进制位数(bps)表示，它与术语波特率容易发生混叠。波特率指单位时间内通信线路状态(电信号)的变化频率，以波特(baud)为单位。仅当采用基波传输方式，或一个调制时间间隔内仅调制一个信号时，这两个概念在数值上才是相同的，而现代通信技术则常使数据传输率高于波特率。

### 5.1.2 异步串行通信的 UART 协议

UART(Universal Asynchronous Receiver Transmitter)指通用异步串行通信收发器。进行异步通信的通信双方必须遵守异步通信控制规程，也称异步通信协议，其协议格式如图 5-1 所示。

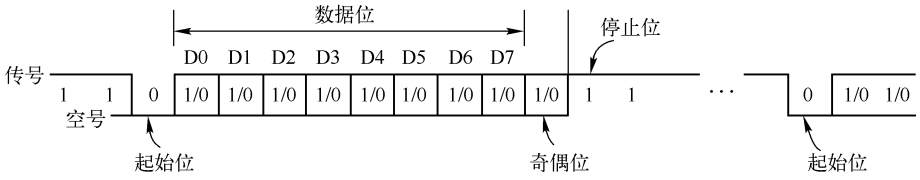



图 5-1 UART 异步通信协议格式

协议格式中逻辑“1”称为传号(MARK)，逻辑“0”称为空号(SPACE)。一个字符单位除包括表示字符本身的数据位(长度为 5~8 位，可选，顺序由低到高)外，还包括起始位(1 位，取值恒为 0)、奇偶校验位(可选有/无)、停止位(长度为 1 位和 2 位，可选，取值为 1)。传送每一个字符都必须以起始位开始，以停止位结束，这个过程称为一帧。异步通信的一帧传输经历以下步骤：

(1) 空闲传输：发送方连续发送传号，处于逻辑“1”状态，表明无数据传输。

(2) 起始传输：送方在任何时刻将信号由 1 变 0，并持续 1 位时间表明启动传输。与此同时，接收方收到这个信息后，等待接收随后同步到来的数据。

(3) 数据传输：自低位到高位连续发送或接收规定的数据位序列。

(4) 奇偶传输：使全部数据位与奇偶位中信号为 1 的数目满足奇偶特性的要求。

(5) 停止传输：停止位的长度可选择 1 位、1.5 位和 2 位。

异步通信的实质是指字符发送的随机性。接收方收到一个起始位后才去与发送方同步，此后双方在约定传输速率下完成一个字符的同步传输。由于一个字符帧最多只有 12 位(起始位+数据位+校验位+停止位)，因此通信双方的时钟频率偏差只要不是过大，就不会累积到错位的程度，而下一个字符起始位到来时又将重新建立同步。

### 5.1.3 RS-232 接口标准

RS-232 是串行通信中应用最广泛的一种接口标准。它被收录在美国电子工业协会工程

部的建议标准第 232 号修改版 C 中，故也称为 EIA RS-232C。该标准与国际电报电话咨询委员会(CCITT)制定的串行接口标准 V.24 基本相同。RS-232 标准的正规名称是“数据终端设备和数据通信设备之间串行二进制数据交换的接口”。通常将通信线路终端一侧的计算机或应用设备称为 DTE(Data Terminal Equipment)，即数字终端设备；而把连接通信线路一侧的调制解调器称为 DCE(Data Communication Equipment)，即数据通信设备。下面就这种接口的机械特性、电气特性和功能描述分别加以说明。

### 1. 接口的机械特性

RS-232 标准对连接器本身的物理特性并没有任何定义，最初以 DB-25 连接器广泛用于电子设备中，而后则以 DB-9 连接器广泛用于 PC 的外设连接。其中，DTE 采用插针方式(Male)的插座形式，而 DCE 采用插孔方式(Female)的插座形式，设备接口如图 5-2 所示。

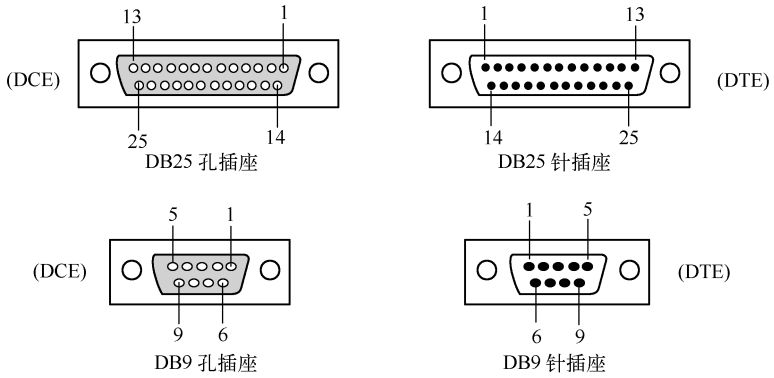



图 5-2　RS-232C 串行异步通信接口的连接器

### 2. 接口电气信号特性

RS-232 标准规定了双极性逻辑电平，仅有两个引脚表示数据信息，其余均是控制信息，它们具有不同的逻辑称呼和逻辑电平，如表 5-1 所示。

表 5-1　RS-232 逻辑电平定义

| 方　向 | 数据信号 | 控制信号 | 电平幅值 |
|---|---|---|---|
| 输出<br>DTE→DCE | 空号(逻辑 0)<br>传号(逻辑 1) | ON(逻辑 1)<br>OFF(逻辑 0) | $V_H$ = +5 ~ +15V<br>$V_L$ = -5 ~ -15V |
| 输入<br>DCE→DTE | 空号(逻辑 0)<br>传号(逻辑 1) | ON(逻辑 1)<br>OFF(逻辑 0) | $V_H$ = +3 ~ +15V<br>$V_L$ = -3 ~ -15V |

RS-232 标准允许信号传输速度在 0 ~ 20 000bps 范围内，同时连接电缆的长度不超过 50ft(约 15m)。实际传输速度与通信距离关系很大，对于短距离 PC 的外设接口常常可以达到 100kbps。RS-232 标准要求接口能够承受任何两个引脚的短接(其短路电流不应超过 0.5A)，用以确保本身及所连接设备不至损坏。

### 3. 接口的功能描述

必须强调，RS-232C 处理的是数据终端设备 DTE 与数据通信设备 DCE 之间的通信协议。对同一信号而言，DTE 侧的输出即 DCE 侧的输入(反之亦然)。通常，信号的名称和意义都是从数据终端设备 DTE 的角度予以说明的。表 5-2 列出了接口实际用于异步通信的信号线定义和功能说明。

表 5-2 用于异步通信的信号线定义和功能说明

| 信号名称 | 助记符 | DB-9 引脚 | DB-25 引脚 | 传输方向 | 功能简介 |
|---|---|---|---|---|---|
| 发送数据 | TXD | 3 | 2 | DTE→DCE | DTE 发送数据 |
| 接收数据 | RXD | 2 | 3 | DTE←DCE | DTE 接收数据 |
| 终端设备就绪 | DTR | 4 | 20 | DTE→DCE | DTE 准备就绪 |
| 通信设备就绪 | DSR | 6 | 6 | DTE←DCE | DCE 准备就绪 |
| 请求发送 | RTS | 7 | 4 | DTE→DCE | DTE 请求切换到发送方式 |
| 清除发送 | CTS | 8 | 5 | DTE←DCE | DCE 已经切换到发送方式 |
| 载波检测 | DCD | 1 | 8 | DTE←DCE | DCE 已收到载波信号 |
| 振铃指示 | RI | 9 | 22 | DTE←DCE | DCE 已经接通通信线路 |
| 信号地 | GND | 5 | 7 | — | 信号地 |
| 速率选择 | DSRD | — | 23 | DTE↔DCE | 双向通知选择较高的速率 |
| 保护地 | — | — | 1 | — | 接设备外壳，安全地 |

TXD 和 RXD 是数据传输线。TXD 用于终端设备 DTE 对通信设备 DCE 发送数据，在未发数据时和字符传送的间隔期保持传号状态(逻辑 1)。仅当信号 RTS(终端设备请求发送)、CTS(通信设备清除发送)、DTR(终端设备就绪)、DSR(通信设备就绪)均为有效状态时，DTE 才能发送数据。RXD 专门用于接收 DCE 发来的数据，它不依赖任何控制信号，仅当无载波信号时或在半双工条件下从发送方式到接收方式的切换过程中才处于固定的传号状态。DCD 为载波检测信号，又称 RLSD(接收线路信号检测)。当通信设备 DCE 收到远程载波信号时 DCD=ON，且在线路接通期间一直保持。DTR 为数据终端设备就绪的指示信号，DTR=ON 表明 DTE 已做好操作准备，允许 DCE 接通通信线路，而 DTR=OFF 则要求 DCE 切断与通信线路的联系。DSR 为数据通信设备就绪的指示信号，当 DCE 已经接通通信线路时，DSR=ON，指示 DTE 可以通过 DCE 与远程的另一终端通信。

半双工控制线 RTS 用于 DTE 请求发送方式工作，而 CTS 线用于 DCE 接收 DTE 的发送请求后通知 DTE 清除发送请求的应答。当 DTE 要求发送数据时 RTS=ON，而在 DCE 完成通信线路切换，发回 CTS=ON 后 DTE 才能开始发送数据。这种 RTS/CTS 请求应答的握手方式只用于半双工系统，而在全双工系统中可将 RTS 和 CTS 常接 ON 状态或短接到载波检测信号 DCD 上。

RI 用于指示通信设备 DCE 的振铃状态。振铃期间 RI=ON，振铃间隔期间 RI=OFF。双向信号 DSRD 用于选择数据信号速率，终端设备 DTE 发 DSRD=ON，命令通信设备 DCE 使用较高的传输速率；通信设备 DCE 发 DSRD=ON，表明已建立所选择的数据传输速率。

### 5.1.4　信号电平的转换

RS-232 接口的 EIA 电平为±5～±15V，与 TTL 电平不同，需要配置收发器。传统的 RS-232 收发器需要±12V 的供电电源(如 MC1488/MC1489、SN75150/SN75154 等)，这对于只有+5V 的单片机系统不够方便。新式收发器芯片内部常常配置一种称为充电-泵原理的电源转换器，可直接使用 5V 电源产生 EIA 信号电平，具体型号有 ADM232、MAX232 等，如图 5-3 所示。

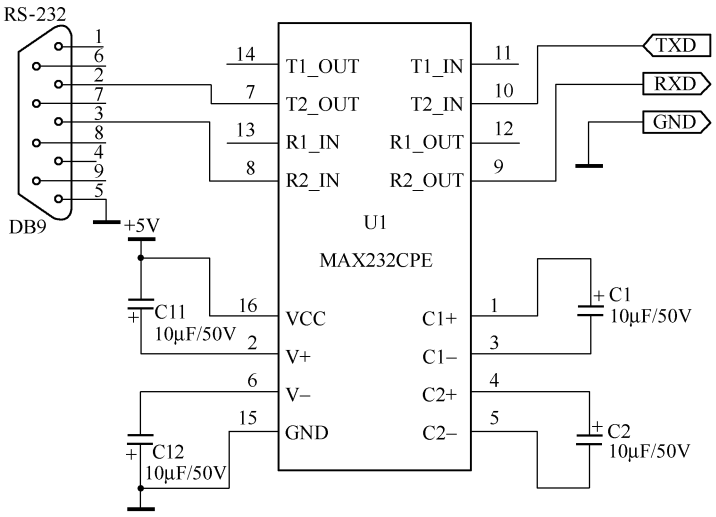



图 5-3　用于 RS-232 的电平转换电路

## 5.2　RS-232 对调制解调器的控制

公用电话系统的历史悠久，所建立的通信网络是一种十分宝贵的资源，因此远程通信理所当然地将其作为首选媒体之一。其特点在于经济、方便，通过调制解调器(Mode & Demode)的方式，可在世界范围将数字信息传送到电话网络所能达到的任何地方。

### 5.2.1　公用电话系统的通信过程

尽管目前公用电话系统的交换机已为计算机化的程控设备，但用如图 5-4 所示的电路仍能说明其工作原理。

在公用电话系统中，其源方摘机拨号到受方振铃应答的过程如下：源方拿起听筒，则电话机电路接通。此时直流电流流过电话机，使交换台得到该路话机接通的信息，便发出频率为 400Hz 的连续正弦波的拨号音(Dial Tone)，源方应在听到拨号音后才开始拨号。拨号分脉冲方式(PULSE)或双音频方式(TONE)。脉冲方式是以直流通路通断(周期为 100ms)形成的脉冲次数来表示所拨的号码，而双音频方式则以不同的两个频率组合来表示所拨号码。交换台收到拨号信息后，首先判断被拨号的话机工作状态。若其正在通话，则发忙音信号(Busy Tone)来通知源方，说明占线；反之，应将振铃信号发往被拨话机，同时向源方发候机音信号

(Touch Tone)，说明其正在试图与受方沟通。若受方用户在振铃的呼叫中拿起话机(Hold Handel)，则会由于直流电路的接通使交换台判明受方线路也已接通。于是交换台撤销发给源方的候机音，并将话路交给源方，从此源方用户与受方用户之间的话路连接就建立起来。

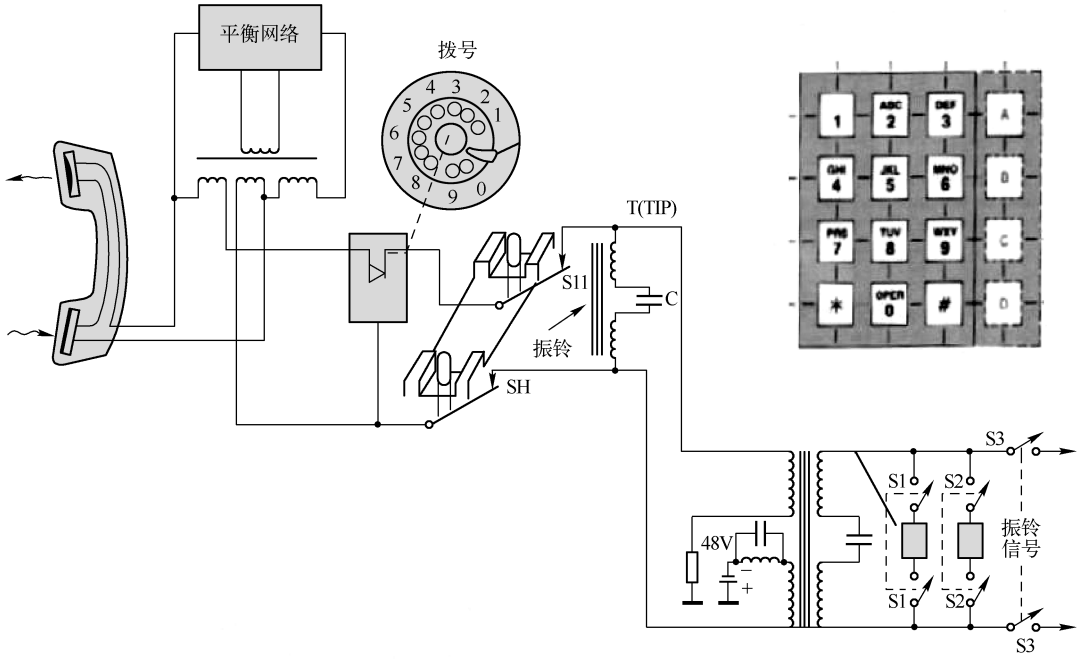



图 5-4　公用电话网的基本工作原理

### 5.2.2　信号调制技术

如图 5-5 所示，由于通信线路的带宽有限，若将计算机输出的数字信号直接接入通信线路，就会发生信号畸变。线路的距离越长，信号的频率越高，波形畸变就越严重，从而引起数据信息的传输出错。

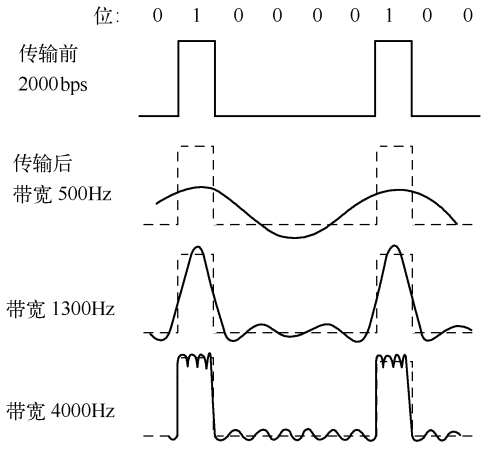



图 5-5　线路带宽对信号传送的影响

电话网络的传统用途为通话，其传递声音信号的频率范围为 300～3000Hz。因此为使数字信号在电话线上传输，需要先将数字信号调制成这个频段范围内的正弦波形，再由通信的接收方将这个正弦信号解调成数字信号。

如图 5-6 所示，3 种常用的调制方法为幅移键控(ASK)、频移键控(FSK)和相移键控(PSK)，分别以两种不同幅值、不同频率或不同相位的正弦波表示数字信号的 1 和 0。由于图中这 3 种调制波均在一个调制单位时间内调制一个信号，故数据信号传输率与波特率相同。

图 5-7 给出另一类调制方法，可根据两位二进制数据形成调相波形的 4 种状态(称为四相调制)，使一个波形周期可以传递两位数据，这样实际得到的数据信号传输率就可以等于线路波特率的 2 倍，达到在 3kHz 频率带宽的电话线上取得更高数据传输率的目的。据此原理，现代调制解调器均已具有工作在 9600bps 以上的传输速率。

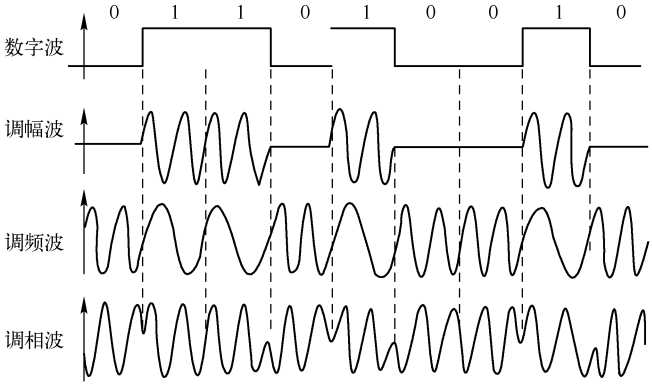



图 5-6　调制解调器工作方式示意图

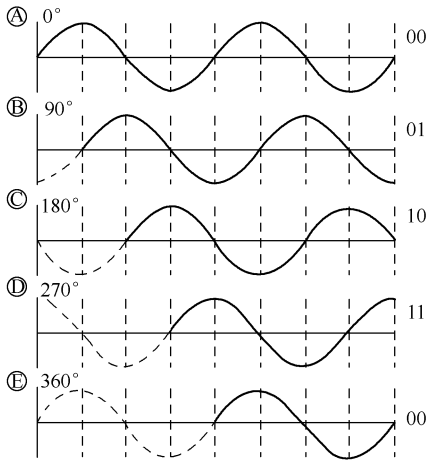



图 5-7　DPSK 调制解调器原理图

### 5.2.3　调制解调器及其控制方式

调制解调器(Modem)是一个包含调制和解调功能的合成装置。作为 DCE，它把 DTE 发送来的二进制数字信号变换成音频信号送到线路，同时将从线路接收的音频信号变换为二进制数字信号并传送给 DTE。

如图 5-8 所示，一个全双工 FSK 型的 Modem 允许的频带内划分出源信道和应答信道，如载波中心频率分别为 1170Hz 和 2125Hz，并使传号与空号的载波频率相差 200Hz。这样，两个信道之间载波的频谱没有重叠，从而减少了全双工通信中的相互干扰。

下面通过一台作为 DTE 的微机与一台作为 DCE 的 Modem 之间应答呼叫的全过程，说明 RS-232 管理 Modem 的标准方法。配置的 Modem 是全双工方式，不需要 RTS/CTS 握手信号，即 DTE 侧保持 RTS=ON，DCE 侧保持 CTS=ON，其余控制信号在初始化时均处于 OFF，工作过程如下。

(1) 空闲等待状态：微机 DTE 持续监视调制解调器 DCE 发出的振铃指示信号线 RI，DCE 等待响应 DTE 的数据终端设备信号 DTR。当这两个信号均为 OFF 时，信息交换处于空

闲状态。

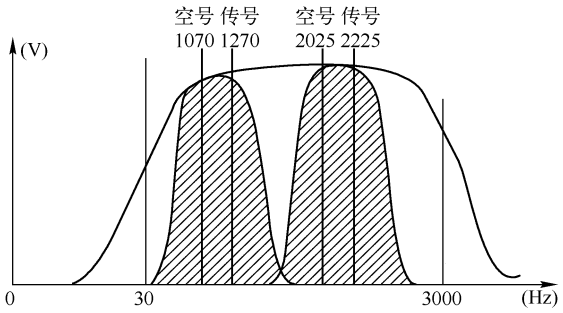



图 5-8　某调制解调器信道的带宽分配

（2）振铃指示接通：DCE 收到远方振铃呼叫后，在振铃作用期间对 DTE 发 RI=ON。

（3）数据终端设备就绪接通：DTE 检测到振铃指示信号 RI 后，便对 RI=ON/OFF 变换的次数进行计数。达到预定的数目后，DTE 发出数据终端就绪信号 DTR=ON。

（4）数据通信设备就绪接通：DCE 收到 DTR=ON 后迫使其 Modem 功能进入摘机状态，即在两秒后对电话网发送应答载波信号，并向 DTE 发数据通信设备就绪信号 DSR=ON。

（5）载波检测接通：应答载波信号被源 Modem 识别后，源 Modem 也在电话线上发出自己的载波信号，应答 Modem 收到这个载波信号后，便向该方微机 DTE 发出载波检测有效信号 DCD=ON。

（6）数据发送和接收同时进行：微机 DTE 收到 DSR=ON 后，随即开始检测载波 DCD=ON 是否出现，以证实数据通道链是否建立起来。在数据通道链建立期间，开始了全双工通信，DTE 通过 TD 信号线发送数据，通过 RD 信号线接收数据。

（7）通信结束撤除数据通道链：数据传输完毕，微机 DTE 撤销数据终端设备就绪信号，即使 DTR=OFF，DCE 收到这个信息后便关闭它的载波源，于是对方远程 DCE 也将停止发送载波信号，因为收不到载波信号 DCE 则向 DTE 发 DCD=OFF 信号，同时发 DSR=OFF 信号，表明对微机 DTE 发出的 DTR=OFF 信号予以响应。链接撤除后，双方 RS-232 接口又都返回状态(1)，准备应答或发出下一个呼叫。

### 5.2.4　智能调制解调器的管理

计算机与传统 Modem 相互之间的联系是由 RS-232 接口的信号电平管理的。与此不同，微机管理智能调制解调器(Smart Modem)由一些以 ASCII 字符串形式出现的接口命令和状态响应实现。这些控制信息与实际需要远程交换的数据信息在形式上并无不同，都是采用 START/STOP 的异步数据传输格式，只是采用不同的解释处理而已。这样在实际应用中，只靠 RS-232 接口的数据发送信号 TD、数据接收信号 RD 以及公用地信号，就可通过发送命令字串和接收响应字串来实现微机对 Modem 的控制。

由于命令字串方式的灵活性，一些原先由微机程序实现的接口功能便改由智能调制解调器本身来承担。例如，传统方式下为了响应远方的呼叫，其微机需要不停地监视 RS-232 的振铃监视线 RI，计算 RI 线上脉冲信号出现的次数，直至达到预设的振铃数后，微机才利用数控终端就绪信号 DTR 去通知 Modem 摘机、发载波信号与远地对方握手，而且终端微机还要进入一个检查 RS-232 接口载波检测信号 DCD 的循环过程。现在使用智能调制解调器的终端微

机很少再做这些工作，微机可通过发送命令 ATA 要求智能调制解调器本身自动应答远方来的呼叫。利用命令 ST=n 预设振铃次数。这样，智能调制解调器可以自动接收呼叫、摘机、握手，直至联机成功给终端微机发送响应信息字串 Connect，或联机失败发字串信息 Carried tone not found(载波未收到)。当然，收到远方发来的振铃呼叫时也会发出响应信号字串 Ring 来通知终端微机。

智能调制解调器的工作状态可以分为两种情况："在线状态"和"命令状态"。前者将来自 RS-232 接口的信息作为数据直接送往远程通信的另一端，后者则将信息作为命令字串来管理自己的操作动作。命令串由其引导符开头，由结束符结尾。通常，智能调制解调器用 AT(Attention 的缩写，意指"注意")作为引导符，用 CR 作为结束符。上电启动时默认选择"命令状态"，以后可用 O 命令切换到"在线状态"，用"+++"返回"命令状态"。表 5-3 给出了一些常用的 AT 命令，并说明其使用格式和操作方法。

**表 5-3　常用 AT 命令和主要寄存器设置**

| | | |
|---|---|---|
| Z—复位，同上电过程，将所有开关设置成默认状态后，执行自检 | | |
| H—H0 挂机，H1 摘机 | A—自动应答 | W—等待第二次拨号音 |
| D—拨号 0~9，#，* | P—脉冲方式拨号 | T—双音频方式拨号 |
| ，—暂停 | ；—拨号后返回命令方式 | |
| E—E0 命令无回声，E1 命令回声 | | |
| Q—Q0 响应发回，Q1 响应不发回 | V—V0 数字响应，V1 字串响应 | |
| Sr? —读寄存器 r 的值，r=0~27 | Sr=n—设寄存器 r 的值 n，n=0~255 | |

主要寄存器

S0—自动应答振铃数(0~255)，默认值 0 表示非自动应答方式
S6—拨号音等待时间(0~255s)，默认值 2
S7—远地载波等待时间(0~60s)，默认值 30
S8—命令字串中逗号表示的暂停时间(0~255s)，默认值 2

响应信息

| | | | | | |
|---|---|---|---|---|---|
| 0 | OK | 命令实现 | 1 | Connect | 连接成功 |
| 2 | RING | 振铃 | 3 | NO CARRIER | 无载波信号 |
| 6 | NO DIALTONE | 无拨号音 | 7 | BUSY | 忙音 |
| 4 | ERROR | 错误发生 | 8 | NO ANSWER | 检测不到应答 |

典型智能调制解调器的面板设置如图 5-9 所示，各指示灯的意义解释如下。

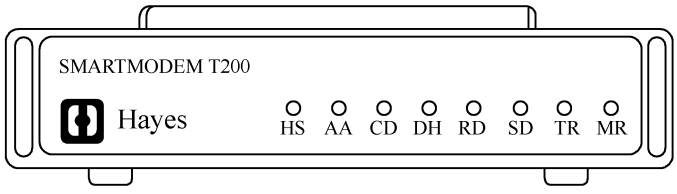



图 5-9　典型智能调制解调器的面板设置示意图

- HS(High Speed)：高速传递。
- AA(Auto Answer)：自动应答方式，在振铃到来时指示灯闪烁。
- CD(Carrier Detect)：载波检测，指示灯亮时表明已与远地接通数据通信线路。
- OH(Off Hook)：摘机。
- RD(Receive Data)：Modem 从 RS-232 接口接收数据。

- SD(Send Data)：Modem 向 RS-232 接口发送数据。
- TR(Terminal Ready)：终端就绪，RS-232 接口的 DTR=ON。
- MR(Modem Ready)：上电灯亮。

智能调制解调器的工作过程也分主动拨号呼叫联机、被动应答连接及联机后数据的双向通信过程，这里分述如下。

(1) 拨号呼叫过程：AT DT86413651 是一条典型的自动拨号命令，其中 T 表示双音频式拨号，若将 T 改为 P，则表示采用脉冲式的拨号，后面的 86413651 为被呼叫的远方 Modem 的电话号码。Modem 上电后 MR 灯亮；命令发送后 Modem 首先摘机，因此 OH 灯亮；接着按命令拨号，拨通后则自动与远方 Modem 交换载波信号，完成握手过程。事实上，现代 Modem 的这一握手过程是很复杂的，它包括机器自动调整波特率，能实现不同协议的选择。然而这些都由通信双方智能调制解调器之间自行安排完成，并不需要终端进行干预，直至最后“握手”成功，对终端发回响应“Connect”，CD 灯亮，表明数据链路已经建立，Modem 进入数据传递方式。或者“握手”失败，向终端发出“No Carrier”(无载波信号)，表明未能建立数据链路。于是挂机，OH 灯也随之灭掉。另外，很可能电话线路根本未通，根据终端响应“No Dial Tone”说明从一开始就未收到拨号音。也可能收到终端响应“Busy”表明拨号遇到忙音，自然也将话路挂断，OH 灯灭。由于 RD 灯和 SD 灯反映数据线的状态，因此在命令发送和响应传输的过程中，RD 灯和 SD 灯会相应地出现闪烁。

(2) 应答过程：可由命令 AT S0=4 将智能调制解调器设置成自动应答方式，此时 AA 灯亮。当远方进行呼叫、振铃响起时 AA 灯就会闪烁，同时智能调制解调器自动计数，直至 4 次振铃响过后，Modem 做摘机动作。然后与远方交换载波，进行“握手”操作，这时 OH 灯亮，如握手成功则 CD 灯亮，否则挂机且 OH 灯灭，同时发出反映相应连接状态的响应信号，其后过程与前述拨号过程情况完全相同。

(3) 数据交换过程：“握手”成功，数据链路建立，智能调制解调器自动转入数据交换的在线工作方式，以后通过 RS-232C 接口传递的数据就会直接通往远方的 Modem，并进而送给通信的另一方终端用户。若想通过终端设备使智能调制解调器从数据传递方式主动返回命令方式，可由终端设备向通信设备发出特定字串“+++”。

对智能调制解调器工作过程和工作状态的了解，不仅可以借助前面板上的指示灯，也可通过声音判断，如交换载波进行“握手”操作时，智能调制解调器发出的载波频率和对方返回的载波频率并不相同，发出的声音差别也很明显。智能调制解调器应用很广，熟练掌握其工作原理对于实际组建远程监控系统是很有帮助的。

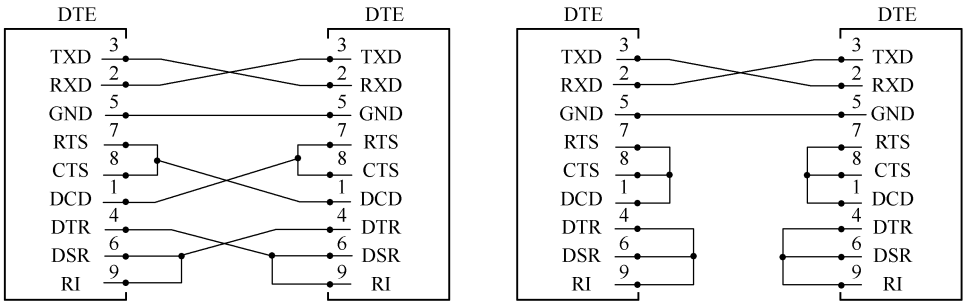
## 5.3　自动测试系统中 PC 主机的串口操作

在自动测试系统和虚拟仪器中，PC 的主导作用显而易见。PC 与外界测试设备连接的传统方式就是 RS-232C 串口，它不仅最早被人们所熟悉，而且使用也最方便。因为只要使用这种串口，一般用户就能利用通用语言编程，直接操作 PC 所连接的外部设备。本节将首先讨论自动测试系统中 PC 串口的硬件连接电路，然后介绍 Windows 操作系统下使用 VB 编程语言操作串口的基本方法和控制仪器的实用程序，以及软件中有关通信协议的一些具体问题，最后以一个远程监控的测试系统实例说明这类系统的开发方法。

### 5.3.1 微机之间的 RS-232 串口连接方法

由于 RS-232C 接口原本是基于 Modem 的串行通信过程的，因此其通信标准实质上被定义为终端设备与 Modem 之间点对点的通信接口协议，即 DTE 和 DCE 之间的接口协议。若实际应用要求两台微机之间直接连接通信，那么人们可能会问：把微机(包括嵌入式设备)的串行口看成是 DTE 呢？还是 DCE？哪一台是 DTE？哪一台又是 DCE 呢？做法合理吗？还有其他的办法吗？下面介绍的 NULL 接口提供了一种标准的解决方法。

图 5-10(a)所示的连接方式称为正规连接，因为它完全遵照 RS-232C 标准定义的控制 Modem 的规则完成各引脚的逻辑功能。双方的 DTE 仍以为与自己一侧的 DCE 在通信，其实双方 DTE 均跳过了 DCE，而实现透明的异步通信。




(a) 正规连接　　(b) 简单连接

图 5-10　NULL 方式的连接方法

此种连接也称双交叉环回接口。其连接要点如下：

(1) 一方 DSR(通信设备就绪)与 RI(振铃指示)相接并与对方的 DTR(数据终端就绪)互连。这意味着当一方的数据终端就绪(DTR 有效)，则另一方的 RI 也就有效，于是呼叫产生并应答响应，同时又使 DSR 有效，即数据通信设备就绪。换言之，只要双方 DTE 准备就绪，便立即使双方 DCE 准备就绪，尽管实际上双方 DCE 并不存在。

(2) 一方的 RTS(请求发送)与 CTS(清除发送)相接并与对方的 DCD(载波检测)互连。这意味着，任何一方提出发送请求(RTS 有效)均可立即得到允许发送(CTS 有效)的响应，同时使对方的 DCD 有效，表明数据通道链已经建立。

(3) 双方的 TXD(发送数据)与 RXD(接收数据)互连。这意味着只要上述两项均准备就绪，双方即可进行双全工传输。

图 5-10(b)所示的连接也称“环回三线接口”。只要任何一方自身请示发送(RTS)有效和数据终端就绪(DTR)有效，即可建立发送和接收的条件要求。如果双方 DTE 在通信中根本不要 RS-232 接口的控制信号，那么只要将双方的 TXD(引脚 2)与 RXD(引脚 3)互连，地线(引脚 5)相接，即是一种最简单的 3 线连接方式了。

可程控仪器作为数据终端设备，理应按 DTE 方式配置接口，这样，既可通过标准 RS-232 接口连接调制解调器进行远程通信，也可通过上述 NULL 方式直接接受计算机的控制。

### 5.3.2　20mA 电流环接口与现场总线 485 接口

PC 技术的发展使 RS-232 串口功能早已配置到主板上的大规模集成电路芯片中，因此 PC 使用者已经没有必要和可能再去研究其 UART 的具体实现电路。然而，一些 PC 配置的 RS-232 接口除了支持 EIA 电平的接口电路外，还支持 20mA 的电流接口电路。为此，在 DB-25 型连接器中加入了 4 个电流信号。图 5-11 是采用电流环接口的示例，依靠电流传输信号的电路不仅可以有效克服长线传输过程中线阻压降的不良影响，而且光电耦合方式也能隔离通信双方两台计算机之间电的联系，从而提高通信的质量和可靠性。

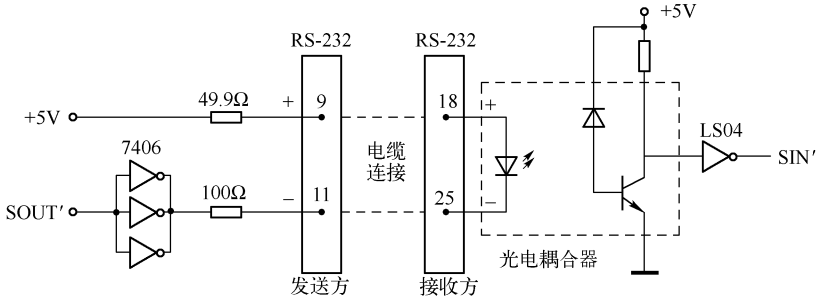



图 5-11　电流环通信接口电路

RS-232 接口的主要特点在于支持大多数通信设备和自动化设备，且 PC 主机的编程环境十分理想。但 RS-232 接口采取点对点的通信方式，因此不能直接用于要求连接多台设备的工业测控领域。RS-485 总线则采用半双工的总线通信方式，可对总线上各设备广播寻址后分别通信，因此在工业现场得到更广泛的应用。

图 5-12 所示为 RS-485 的收发器接口电路，通过芯片 MAX481 可将 TTL 电平的 UART 工作方式转换为 RS-485 的 2 线差分方式，其中$\overline{RE}$/DE 线为控制信号，可用来管理半双工通信方式下的通信方向。对于 PC 配置的 RS-232 接口，可将图 5-12 所示电路与图 5-3 所示电路结合，不难构建 RS-232/RS-485 的转换电路。

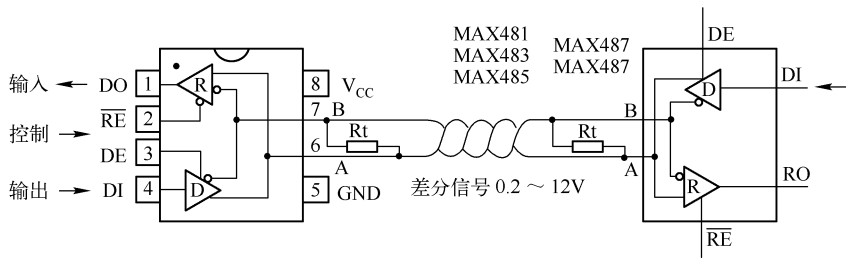



图 5-12　RS-485 的收发器接口电路

与 RS-232 接口相比，RS-485 的差分电路方式能从地线干扰中分离有效的通信信号，区别出 0.2V 以上的电位差。因此，可以免受对地参考系统的电位波动和共模电磁干扰的影响，从而可以大大减少逻辑信号高、低电平之间的电压摆幅要求，明显提高一定通信范围内的传输速率，或在相同传输速率下的通信距离。

### 5.3.3 Windows 环境下 PC 串口的 VB 编程

Windows 操作系统下的 VB、VC 等通用编程语言开发环境中都为串口的开发提供了十分方便的编程资源。在 VB 环境下开发需要进行串口通信的自动测试系统时，可以使用一个名为 MsComm 的控件。通常 VB 开启时，MsComm 控件不会自动出现在工具箱内，因此需要在主菜单的工程项目(Project)处下拉选择组件(Components)的选项，出现对话框后再下拉列表，选择 Microsoft Comm Control 6.0 选项，单击确认后就可在工具箱中看到 MsComm 这个控件的图标(形如一个电话机)。实际编程时先将这个控件加入到程序表单中，如命名为 Form1. MsComm1，于是在这个控件的属性表处就可以看到它的各种属性，应用程序通过读、写这些属性，就可达到操作串口的目的。常用的控件属性说明如表 5-4 所示。

表 5-4 Comm 控件常用属性

| 名 称 | 类 型 | 说 明 |
|---|---|---|
| CommPort | Integer | 设定或返回通信连接端口代号，范围为 1~16 |
| Settings | String | 表示数据传送的波特率、校验方式、数据位数和停止位数的设置 |
| PortOpen | Bullen | 表示端口状态，数据传输时 PortOpen 为真 |
| OutPut | String | 将一个字符串或字节数组写入输出缓冲区 |
| InPut | String | 以先进先出(FIFO)的方式从输入缓冲区中读出数据 |
| InputLen | Integer | 指定用 Input 方法从输入缓冲区读取的字节数，取 0 值时表示所有数据 |
| InputMode | Integer | 读取数据的方式：0—ComInputModeText(字符)<br>1—ComInputModeBinary(二进制) |
| InBufferCount | Integer | 已从串口接收到输入缓冲区等待读取的字节数 |
| InBufferSize | Integer | 输入缓冲区本身的最大字节数 |
| Rthreshold | Integer | 触发接收事件的字节数，为 0 则表示不产生接收事件 |

如 PC 配置多个串口，控件则可通过 CommPort 属性选用其中之一。也可建立多个串口控件，以便同时操作多个串口的通信活动。控件的 Settings 属性用于波特率(300、600、1200、2400、4800、9600)、奇偶校验方式(o/e/n)、数据位(7/8)和结束位(1/2)的设置。如表 5-5 所示，MsComm 控件有多个事件，但最重要的事件是 ComEvReceive 事件。每当串口接收到外界发来的串行数据时，就会自动地将其依次存入系统内部的数据缓冲区，同时启动事件 ComEvReceive。这样在 VB 中响应 ComEvReceive 事件的程序代码处，就可以安排接收代码来读取已经收到缓冲区中的数据。在 MsComm 控件下读取数据一般要使用方法 Input，读出的数据会从缓存区自动清除，以防再读时出错。

表 5-5 Comm 控件事件

| 名 称 | 类 型 | 说 明 |
|---|---|---|
| ComEvReceive | 2 | 收到 Rthreshold 字符后产生，直到用 Input 属性从接收缓冲区读出数据 |
| ComEvCTS | 3 | CTS 线的状态发生变化 |
| ComEvDSR | 4 | DSR 线的状态发生变化 |
| ComEvCD | 5 | CD 线的状态发生变化 |
| ComEvRing | 6 | RI 线上检测到振铃信号 |
| ComEvEOF | 7 | 接收到字符 EOF(26) |

通过程序清单 5-1 提供的程序示例，可以清晰地看出 VB 中通过 MsComm 控件实现串口操作的具体方法。即先指定串口设备，再配置通信参数。然后打开串口，就可发送命令字串。

**程序清单 5-1　串口操作的典型示例。**

```
MsComm1.CommPort = 1                '设通信端口为 COM1
MsComm1.Settings = "9600,n,8,1"     '设 9600 波特率、无奇偶校验、8 位数据位和 1 位停止位
MsComm1.PortOpen = True             '打开端口
MsComm1.Output = "*CLS;*WAI" & Chr(10)      '发清除标志的命令
MsComm1.Output = "*IDN?" & Chr(10)          '发查询命令:识别设备类型信息
Sleep 500:sId$ = MsComm1.Input              '将串口收到的信息读入字串变量 sId$

MsComm1.Output = "SEL:CH1 ON;AUTOS EXEC" & Chr(10)      '发命令设置通道
MsComm1.Output = "HOR:MAIN:SCAL 2.5E-3" & CStr(10)      '发命令设主时基为 2.5ms/格
MsComm1.Output = "CH1:SCAL 100E-3;POS 0" & Chr(10)      '发命令设通道 1 增益为 100mV/格
MsComm1.Output = "ACQ:STOPA RUNST;STATE RUN" & Chr(10)  '发命令启动正常运行模式
MsComm1.Output = "MEASU:IMM:SOU CH1;TYPE PK2" & Chr(10) '发命令要求读取峰-峰值
Sleep 500:Data$ = MsComm1.Input             '将串口收到的数据读入字串变量 Data$
```

这里，程序所操作的可程控仪器就是本书第 3 章所述的数字示波器 TDS200，可通过仪器面板按键 UTILITY，进入屏幕菜单后依次选择“选件”、“RS-232 设置”，直至进入与 MsComm 控件 Setting 属性对应的串口设置选项。尽管 GPIB 接口和 RS-232 接口不同，但上层使用的 SCPI 程控命令相同，可对照程序清单研究体会。首先通过“ *CLS; *WAI”复位，然后便可通过通用命令“ * IDN?”来查询仪器识别字串。这条命令不涉及仪器参数设置，故能直接看出 RS-232 接口操作的基本功能正常与否。下面的命令分别用于示波器本身通道功能的配置、增益和偏移等参数的设置，建立数据采集模式并启动测试过程，最后的 MEASU 命令表示从通道 1 读取测试波形峰-峰值的具体要求。

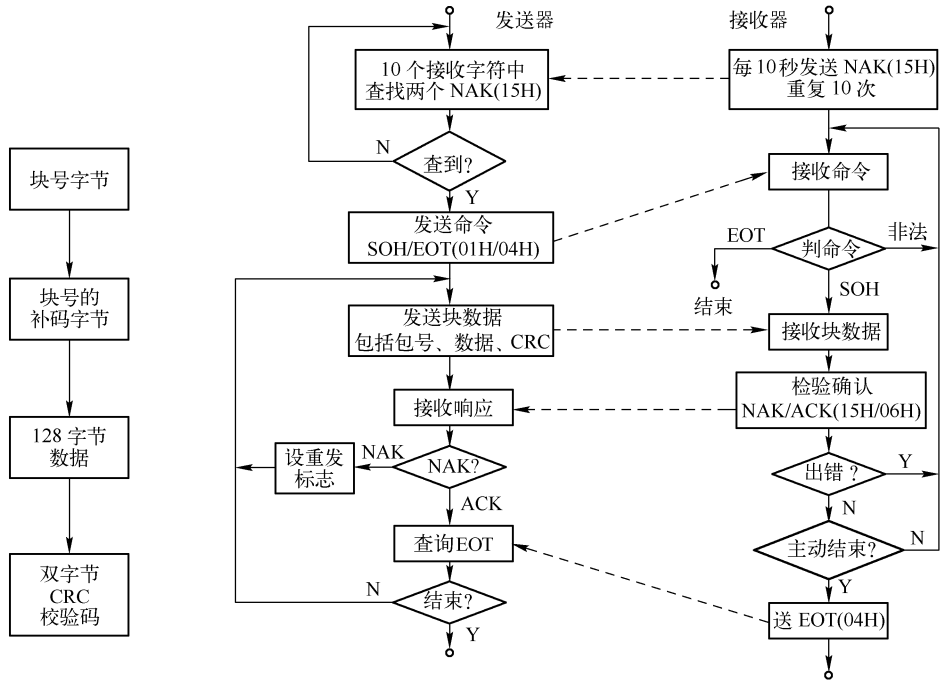
注意，实际串口的数据接收过程是由 PC 操作系统底层 I/O 驱动程序自动完成的，而不管程序是否包含接收函数方法 Input。方法 Input 只是读取已经存入输入数据缓冲区中的收到的数据，因此程序发出要求提供峰-峰值的命令后 可能需要稍加延迟，或等待接收事件产生后再去调用 Input 方法读取数据。

### 5.3.4　XMODEM 通信协议

XMODEM 是数据传输的一个通信协议，早在 8 位计算机时代就已广泛应用于各种场合。如图 5-13(a)所示，XMODEM 定义了一个数据块，数据块起始于一个块号字节，接着是块数据字节数目的补码，然后传送 128 字节的数据内容，最后是由这 128 个数据字节计算得到的 16 位循环冗余检验码 CRC。

图 5-13(b)所示为描述数据发送器和数据接收器之间的数据传输过程。一旦建立了通信联系，数据接收器就发出一系列 NAK 字符(15H)，向数据发送器表达自己的状态。若数据发送器所接收的 10 个字符中包括 2 个连续的 NAK，就认为接收器已经准备好，于是发送控制符 SOH(01H)来提醒接收器准备接收后面的数据。当接收方收到 132 字节(包括 2 字节的块号及其补码、128 字节数据和 2 字节 CRC 校验字)后，需要首先检验块号的正确性。将块号字节与补码字节加在一起，其结果应为零。同时，接收器应按 CRC 算法对接收到的数据块

进行 CRC 校验，若出错，就发回一个 NAK 字节，要求发送器重新传送前次发送的数据块；若没有检测到任何错误，接收器就给发送器发回 ACK(06H)字符并等待发送下一块数据。XMODEM 协议以一个 EOF 信号表示文件传送的结束。通信联系建立后，若发送器和接收器在约 1 分钟内仍没有完成正确的同步交换，通信就会终止。接收器可在任何时刻通过发出一个 CAN 字符来终止发送器继续发送数据。




(a) 协议块格式　　(b) 协议的通信流程

图 5-13　XMODEM 通信协议

### 5.3.5　循环冗余校验码 CRC 算法

循环冗余校验 CRC(Cyclic Redundancy Check)的编码和解码方法具有较强的检错和纠错能力，因此被广泛用于差错控制过程。CRC 校验的基本思想是在发送端所要传送的 $n$ 位二进制信息码序列后，按一定规则产生一个 $r$ 位的校验码(CRC 码)，并依次发出。接收端根据信息码和 CRC 码之间所遵循的规则进行检验计算，以确定传输是否出错。在代数编码理论中，可将一个码组表示为一个多项式，码组中各码元则为二进制的多项式系数。如序列 1100101 表示为 $1\cdot x^6+1\cdot x^5+0\cdot x^4+0\cdot x^3+1\cdot x^2+0\cdot x+1$，即 $x^6+x^5+x^2+1$。

设 $P(x)$ 为编码前原始信息序列的多项式(其最高次幂为 $n-1$)，$G(x)$ 为体现校验算法的生成多项式(其最高次幂为 $r$)，$R(x)$ 为 CRC 多项式，$T(x)$ 为编码后带 CRC 校验项的总序列多项式。发送方编码时将 $P(x)$ 乘以 $x^r$(对应信息码序列左移 $r$ 位)，再除以 $G(x)$ 得余式 $R(x)$，其表达式为 $T(x)=x^r P(x)+R(x)$；接收方解码时将收到的总序列多项式 $T(x)$ 除以 $G(x)$，若余数非 0 则表明传输过程有误。

举例来说，设 $P(x)=x^3+x^2$，即信息码序列为 1100；生成多项式为 $G(x)=x^3+x+1$，记为 1011。计算 CRC 的过程为

$$\frac{x^r \cdot P(x)}{G(x)}=\frac{x^3 \cdot (x^3+x^2)}{x^3+x+1}=\frac{x^6+x^5}{x^3+x+1}=(x^3+x^2+x)+\frac{x}{x^3+x+1}$$

即有余式 $R(x)=x$。考虑 $G(x)$ 的最高次幂为 3，则余项 $R(x)=x$ 的最高次幂则应为 2，对应的 CRC 为 010。而对接收方 $T(x)=x^6+x^5+x$ 来说，余项为 0。具体多项式长除过程如图 5-14 所示，注意二进制系数项相减时 $-x^i=x^i$。

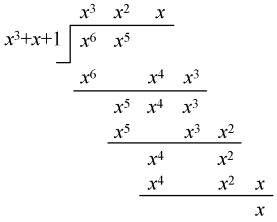



图 5-14　二进制长除过程

生成多项式 $G(x)$ 可具有不同形式，以此体现不同的算法。例如：

CRC-16　$G(x)=x^{16}+x^{12}+x^2+1$　简记为：1005

CRC-ITU（即 CRC-CCITT）　$G(x)=x^{16}+x^{12}+x^5+1$　简记为：1021

CRC-32　$G(x)=x^{32}+x^{26}+x^{23}+\cdots+x^2+x+1$　简记为：04C11DB7

由于 G(x)的最高幂次项系数固定为 1，故简记式中可将最前面的 1 统一去掉。

二进制系数的多项式除法可用除法电路实现。图 5-15 以实现 CRC-CCITT 算法为例，用 16 位移位寄存器的状态表达 CRC 的计算过程。

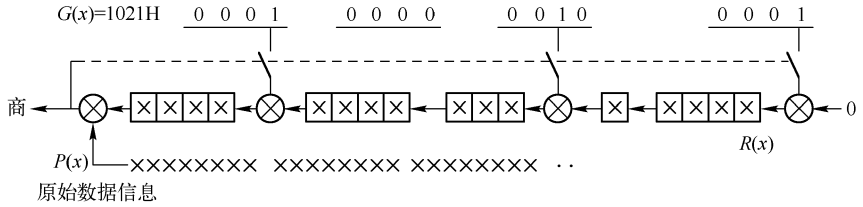



图 5-15　CRC-CCITT 协议的生成多项式（1021H）

各寄存器初始状态为“全 0”，原始数据信息的逐位输入与寄存器移位同时进行，当输入的信息比特位与移出的寄存器最高比特位之间所进行的异或运算结果为真时，则用 $G(x)$ 的简记式所对应的 16 位编码（1021h=0001 0000 0010 0001）与此刻移位寄存器本身内容进行异或操作后更新该寄存器。随着信息比特序列的逐位输入，上述操作过程循环进行，移位寄存器中将始终保持对应 $R(x)$ 的 CRC 计算结果。

由于上述比特型算法需要逐位进行运算，故效率较低。考虑数字通信过程一般是对一帧数据进行 CRC 校验，而字节是帧的基本单位，因此可采用一种按字节查表的快速算法。若事先针对输入信息字节中可能出现的内容（共 256 种），将它们对应的 CRC 分别计算出来并放在一个转换表中，则编码计算 CRC 时就可直接从表中查找。该算法基于这样一个事实：计算传输某字节后的 CRC 码，仅应依据该传输字节的本身取值和当前寄存器所存 CRC 码的取值。这样分析之后，可将数据字节与 CRC 码高 8 位进行“异或”运算后的结果作为查询转换表的索引值，再以查表得到的 16 位二进制码与 CRC 码左移 8 位后的结果进行“异或”运算，所得结果即为更新后的 CRC 取值。

通信数据发送时，将针对全部数据序列计算得到的 CRC 放在数据之后发出；数据接收方则在 CRC 初值设为全“0”的条件下，对接收数据依次执行同样的 CRC 算法。若全部传输内

容(数据+CRC 码)在接收方再次进行 CRC 计算后为 0，则校验通过，否则校验失败。

具体代码见程序清单 5-2，针对数据序列的每个字节 data，均可通过逐次调用函数 UpdateCRC 不断更新 CRC 码。CRC 转换表的建立则可通过事先调用函数 SetTbCRC 完成，即将取值为 0~255 的字节内容依次进行上述 CRC 计算，从而得到 256 个 16 位的 CRC 计算值作为 CRC 转换表的全部内容。

**程序清单 5-2　用于处理 CRC-CCITT 校验算法的 VB 程序。**

```
Dim tbCrc(256)As Long                    'CRC-ITU 的快查表
Public Function UpdateCRC(ByVal crc As long,ByVal data As byte)as Long
    Dim idx%
    idx% =(crc /256)Xor data             'CRC 码右移 8 位后与数据异或作为索引
    UpdateCRC=(256& * (crc and &HFF&))Xor tbCrc(idx% )  '查表取数后再与左移
                                                        8 位的 CRC 码进行异或操作
End Sub
Public Sub SetTbCRC(Gx As Long)          '置参数 Gx=&H1021,可建立 CRC-ITU 的快查表
    Dim k% ,i% ,c&,d&
    For k% =0 To 255
        c&=0:d&=(k% *256)And &HFFFF&
        For i% =1 To 8
            c&=(c&+c&)And &H1FFFF:d&=(d&+d&)And &H1FFFF
            If(c& Xor d&)and &H10000 Then c&=(c& Xor Gx)And &H1FFFF
        Next i%
        tbCrc(k% )=c& And &HFFFF&
    Next k%
End Sub
```

通过 SetTbCRC 函数建立的 CRC 转换表应具有表 5-6 的数据形式。实际程序设计时也可据此建立常数表格，以供函数直接调用。

**表 5-6　CRC 计算转换表**

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0000 | 1021 | 2042 | 3063 | 4084 | 50A5 | 60C6 | 70E7 | 8108 | 9129 | A14A | B16B | C18C | D1AD | E1CE | F1EF |
| 1231 | 0210 | 3273 | 2252 | 52B5 | 4294 | 72F7 | 62D6 | 9339 | 8318 | B37B | A35A | D3BD | C39C | F3FF | E3DE |
| 2462 | 3443 | 0420 | 1401 | 64E6 | 74C7 | 44A4 | 5485 | A56A | B54B | 8528 | 9509 | E5EE | F5CF | C5AC | D58D |
| 3653 | 2672 | 1611 | 0630 | 76D7 | 66F6 | 5695 | 46B4 | B75B | A77A | 9719 | 8738 | F7DF | E7FE | D79D | C7BC |
| 48C4 | 58E5 | 6886 | 78A7 | 0840 | 1861 | 2802 | 3823 | C9CC | D9ED | E98E | F9AF | 8948 | 9969 | A90A | B92B |
| 5AF5 | 4AD4 | 7AB7 | 6A96 | 1A71 | 0A50 | 3A33 | 2A12 | DBFD | CBDC | FBBF | EB9E | 9B79 | 8B58 | BB3B | AB1A |
| 6CA6 | 7C87 | 4CE4 | 5CC5 | 2C22 | 3C03 | 0C60 | 1C41 | EDAE | FD8F | CDEC | DDCD | AD2A | BD0B | 8D68 | 9D49 |
| 7E97 | 6EB6 | 5ED5 | 4EF4 | 3E13 | 2E32 | 1E51 | 0E70 | FF9F | EFBE | DFDD | CFFC | BF1B | AF3A | 9F59 | 8F78 |
| 9188 | 81A9 | B1CA | A1EB | D10C | C12D | F14E | E16F | 1080 | 00A1 | 30C2 | 20E3 | 5004 | 4025 | 7046 | 6067 |
| 83B9 | 9398 | A3FB | B3DA | C33D | D31C | E37F | F35E | 02B1 | 1290 | 22F3 | 32D2 | 4235 | 5214 | 6277 | 7256 |
| B5EA | A5CB | 95A8 | 8589 | F56E | E54F | D52C | C50D | 34E2 | 24C3 | 14A0 | 0481 | 7466 | 6447 | 5424 | 4405 |
| A7DB | B7FA | 8799 | 97B8 | E75F | F77E | C71D | D73C | 26D3 | 36F2 | 0691 | 16B0 | 6657 | 7676 | 4615 | 5634 |
| D94C | C96D | F90E | E92F | 99C8 | 89E9 | B98A | A9AB | 5844 | 4865 | 7806 | 6827 | 18C0 | 08E1 | 3882 | 28A3 |
| CB7D | DB5C | EB3F | FB1E | 8BF9 | 9BD8 | ABBB | BB9A | 4A75 | 5A54 | 6A37 | 7A16 | 0AF1 | 1AD0 | 2AB3 | 3A92 |
| FD2E | ED0F | DD6C | CD4D | BDAA | AD8B | 9DE8 | 8DC9 | 7C26 | 6C07 | 5C64 | 4C45 | 3CA2 | 2C83 | 1CE0 | 0CC1 |
| EF1F | FF3E | CF5D | DF7C | AF9B | BFBA | 8FD9 | 9FF8 | 6E17 | 7E36 | 4E55 | 5E74 | 2E93 | 3EB2 | 0ED1 | 1EF0 |

### 5.3.6　强震观测系统远程监控的实例分析

地震是最严重的自然灾害之一，观测和记录每一次地震的数据具有重要的意义。强震观测仪由加速度传感器和记录仪两部分组成，配置在位于地震活动区的观测台站，长期监测该处地壳的活动情况，根据其东西、南北和垂直 3 个自由度的震动情况和某种强震特征条件来触发记录仪，并在记录仪的存储体上记录强震波形后，再通过某种远程通信方式传送到远在监控中心的计算机中进行进一步处理。这里在简述测试背景后，将重点介绍数据远程通信和仪器远程监控的方法。

如图 5-16 所示，强震观测仪通过三分量加速度计采集强震信号，经过各路前置放大器、滤波器、采样保持器和 A/D 转换器变为数字数据后暂存在缓存区中；然后通过数字信号处理计算各通道数据峰-峰值后，根据地震波的特点进行多票表决来确定触发动作(即按多种判据综合考虑来确认是否为地震引起的强震信号)，触发后则以文件的形式将记录数据保存在 4MB 的 CMOS 非易失 RAM 中，每 256B 数据为 1 页。4MB 总存储容量中包括 16KB 的目录区，实际数据存储区为 16 320 页。仪器内部有一个 4.096MHz 晶振分频产生的 1ms 定时脉冲信号，作为系统的时基信号。由于地震烈度和震中的确定不但依据多个台站的震动数据本身，还要准确知道台站的位置和数据产生的时间，因此系统内部提供 GPS(Global Position System，全球定位系统)模块来校时。考虑野外长期工作的需要，设备密封且内充正压氮气，并配备具有复充能力的电池供电系统。

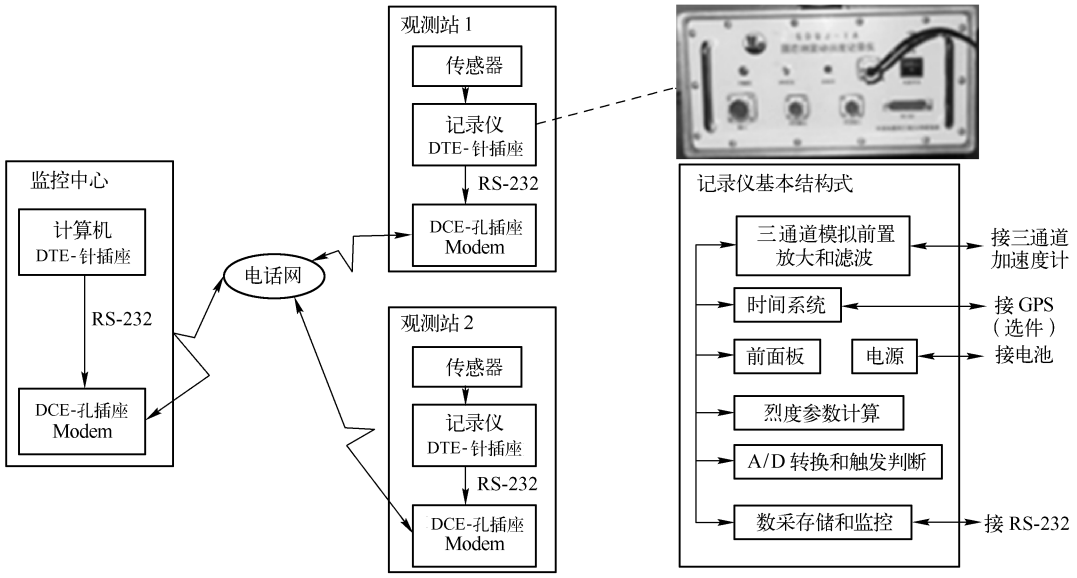



图 5-16　强震观测仪系统组成

一个强震事件发生后，仪器将通过调制解调器(Modem)和电话网络自动拨号，向监测中心通报与此次强震有关的各种信息，包括仪器编号、数据特征、事件时间和所在地理位置等参数，并根据上位机的监控命令进一步上传地震本身的波形记录数据，也可根据监控命令改变控制方式和参数。记录仪参数设置的项目很多，包括数据采样周期、数据文件长度、采样通道数、通道增益、滤波参数、定时参数、触发器决策的总票数及触发算法的选择等，也包括

诸如远程文件列表、文件删除、文件恢复、上传文件、头文件内容等多种功能命令。为了方便操作，记录仪设有多个工作窗，每个工作窗可以预先配置不同的参数，这样只需执行选窗命令即可迅速完成一组参数的设置。

监控操作的主界面如图 5-17 所示，上部的显示框用于当前状态的指示，下部的显示框用于显示发送的命令内容和接收的原始信息，中部的波形显示器主要用于观察地震波形。主菜单由多个下拉式子菜单组成，如状态、参数、时间、存储、数据、设备、通信、系统、帮助等。每个子菜单含有多个选项，用于执行更具体的功能或命令。主窗体标题下面的工具条则由某些工具按钮组成，提供常用命令的快速访问。

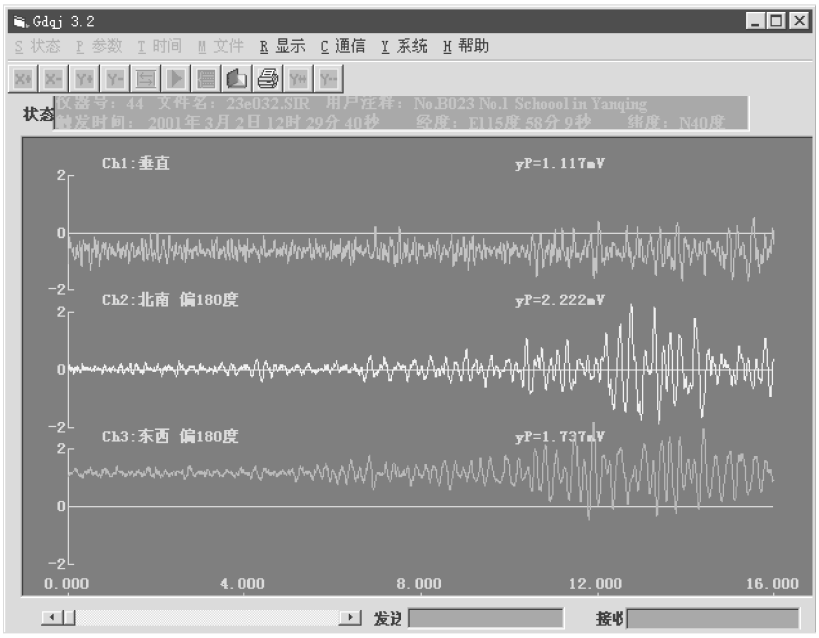



图 5-17　强震观测仪的上位机监控界面

在主窗体菜单的状态栏下可以通过短状态快速查看仪器的实际状态，所设条件监控、强迫监控、返回数采、功能检查、实时监视等选项则可分别用于各种状态的切换；参数栏下分采样参数、文件参数、窗参数、存储参数、装载参数、密码参数和默认设置等若干项，用于各类参数的分别查看和编辑；时间栏下包括定时的显示、设置和同步方式的确定；文件栏下的选项用于浏览文件目录、读事件摘要、删除文件、恢复文件、上传文件等功能；显示栏下则可选择数据波形、触发事件、转储文本和参数副本的显示；通信栏体现对通信串口和调制解调器的设置，如选择端口、波特率、奇偶校验方式、数据与停止位数等。为了使显示图形的操作方便，窗体上方设有调整坐标的各种快捷键。

监控中心主机采用 Modem 利用电话线与强震观测记录仪的通信有两种方式：一是主机拨号上网呼叫仪器；二是主计算机等待和应答仪器的呼叫。通常地震发生时，可由仪器呼叫主计算机，建立通信联系后就将该事件与烈度有关的主要参数发至主机。同时，主计算机可以随时选择拨号上网，在给出电话号码、叫通仪器后发送口令，并经仪器确认后建立通信联系，进而可以继续获取更详细的地震波形记录数据。仪器收到命令后，通常总要反馈相应的

信息。由于一般监控命令比较短，故可采用 ASCII 字符串方式以利于程序调试时直接观察。但强震观测记录数据可能很长(如记录时间有时高达 20 分钟，每秒采样 200 次下 24 位多个通道的数据量也很可观)，为提高效率采用二进制直接编码。远距离传输难免会出现通信错误，因此需要使用 XMODEM 通信协议。由于通信分包进行，出现通信问题时只需重发出错包。实践结果表明，线路的通信质量一般还是不错的，因此可以采用较大的字节数目(1024B)的包结构来提高通信效率。读取强震观测记录时，强震记录仪一方是发送方，而监控系统一方则是接收方。这里节选作为监控系统一方的 PC 接收数据过程的 VB 程序清单 5-3，以使读者了解实际编程方法。

**程序清单 5-3　接收强震记录的 VB 程序节选。**

```
    … 串口已配置,使用 8 位字节传输模式;记录文件已经打开,使用#1 的缓冲区通道
n% = 0:s$ = "TX " & s1$ & vbCr: ShowNotes s$                    '显示发送命令
SioTxString MSComm1,s$ :ShowNotes "准备,接收数据文件…"发送文件传送命令
For i% = 0 To 15                                                '试图建立通信
  SioTxString MSComm1,Chr(NAK):ShowNotes " NAK=" & NAK :sleep 1000
  size% = SioRxData(MSComm1,sb,1,1000)                          '收信息到 sb(0)
  If size% >0 Then Exit For
Next i%
If i% >= 15 Then ShowNotes = "无响应,请检查有无文件!!":GoTo lb1
j% = 1                                                          '成功建立通信
Do
    For i% = 0 To 2                                             '三次
        c = sb(0)
        Select Case c
            Case SOH:n1% = 128:s$ = "小包"
            Case STX:n1% = 1024:s$ = "大包"
            Case EOT:SioTxString MSComm1,Chr(ACK):ShowNotes
                                                    "文件结束标志":Exit Do
            Case Else:ShowNotes "非要求命令,请确认监控状态!!!":GoTo lb1
        End Select
        If SioRxData(MSComm1,sa,2,1000)< 0 Then ShowNotes
                                                    "读取包号超时":GoTo lb1
        If SioRxData(MSComm1,ss1,n1% ,2000)< 0 Then ShowNotes
                                                    "读包超时":GoTo lb1
        If SioRxData(MSComm1,sc,2,1000)< 0 Then ShowNotes
                                                    "未读到 CRC":GoTo lb1
        If((sb(0)+sb(1))<>0)or(sb(0)<>(j% Mod 256))Then ShowNotes
                                                    "包号错":GoTo lb1
        crc = 0
        For ik% = 0 To n1% - 1:crc = UpdateCRC(crc,ss1(ik% )):Next ik%
        For ik% = 0 To 1:crc = UpdateCRC(crc,sc(ik% )):Next ik%
        If crc = 0 Then                                         'CRC 校验是否通过
            Put #1,,ss1:                                        '写入文件记录
            SioTxString MSComm1,Chr(ACK):ShowNotes "ACK"'发 ACK
            n% = SioRxData(MSComm1,sb,1,10000)                  '收信息到 sb(0)
            If n% < 0 Then ShowNotes "未收到下一个命令符!":GoTo lb1
            Exit For
```

```
            Else
                SioTxString MSComm1,Chr(NAK):ShowNotes "CRC 校验出错,请求重发"'发 NAK
                n% =SioRxData(MSComm1,sb,1,10000)        '收信息到 sb(0)
                If n% < 0 Then ShowNotes "未收到下一个命令符 !":GoTo lb1
            End If
        Next i%
        If i% >=3 Then ShowNotes "已试三次,退出":GoTo lb1
        j% =j% +1
    Loop
    ShowNotes "数据接收结束"
    lb1:Close #1:MSComm1.InputMode=comInputModeText   '恢复串口字符模式
```

这里，串口输出和输入分别使用子函数 SioTxString 和函数 SioRxData，前者简单使用 MsComm 控件中的 Output 方法，后者则在规定的时间内从数据输入缓存区中读取规定字节数目的内容后返回。

## 5.4 基于 TCP/IP 的远程测控功能

我们一直强调：所谓虚拟仪器就是要充分发挥通用计算机及其软件在测试系统中的作用，这种作用当然也应包括为虚拟仪器用户提供远程联网的通信功能。不过这里所说的联网功能不再通过操作系统本身提供的各种操作界面，也不再需要人机对话式的操作过程，而应通过编程操作将其嵌入到自动测试系统和虚拟仪器当中，在应用系统运行过程中无人干预地使测试数据在网络中自动传递。同时，这种针对测试的联网通信还要求调用简捷，使广大测试工作者只要了解网络通信最基础的概念，就能使用网络完成测试任务所需要的远程通信工作。

### 5.4.1 TCP/IP 网络通信协议

TCP/IP 是目前远程网络通信中广泛使用的一种协议，其四层结构如图 5-18 所示。在 TCP/IP 中，IP 网络层以下用于处理网络中两台机器之间物理连接的点对点通信，而传输层则用来处理最终信源机与信宿机之间信息通道的端对端连接。传输层屏蔽了下层的通信细节（如网络拓扑结构、物理层协议等），它使应用进程看似两个传输层实体之间有一条端到端的逻辑通信信道。

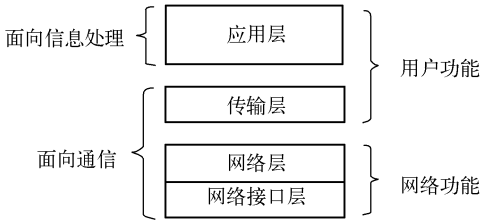



图 5-18 TCP/IP 协议四层体系结构

TCP/IP 提供给用户使用的是应用层和传输层。应用层协议直接为用户的应用进程提供服务，它的具体内容就是规定应用进程在通信时所遵循的协议，如实现网页的 HTTP 协议、

用于电子邮件的 SMTP 和 POP3 协议、用于文件传输服务的 FTP 协议就工作在这一层。TCP/IP 传输层的主要作用是建立网络中主机应用进程之间的通信，常采用两种协议：TCP(传输控制协议)和 UDP(用户数据报协议)。它们均需利用 IP 层提供的服务，分别提供端对端的服务。

TCP 提供的是全双工、可靠、面向连接的字节流传输数据服务。在传送数据前必须先建立连接，数据传输后则需释放连接。TCP 的连接和建立都采用客户/服务器(C/S)模式。主动发起连接请求的应用进程称为客户(Client)，而被动等待连接建立的应用进程称为服务器(Server)。连接建立采取 3 次握手的方式，客户端先向服务器端发出连接请求，然后服务器端发出确认，最后客户端发送对信息的确认，从此连接建立起来。TCP 要求在传输数据时对接收的报文段进行校验，出现差错时将丢弃该报文段，并要求发送端没有收到接收端确认时重发该报文段，从而保证 TCP 在通信过程中可靠的数据交付。

用户数据报协议 UDP 是传输层经常使用的另一个协议。UDP 通过 IP 在机器之间传送报文，但没有提供可靠的连接传输服务。传输过程中 UDP 报文有可能会出现丢失、重复及乱序等现象，因此要求使用 UDP 的应用程序承担可靠性方面的全部工作。UDP 在传送数据之前不需要建立连接，远地主机的传输层在收到 UDP 报文后也不需要给出任何确认，这样 UDP 几乎是直接建立在 IP 协议之上，只是为面对应用进程的传输层提供了端对端服务。虽然 UDP 不提供可靠交付，但在某些情况下(底层通信子网工作相当可靠时)，则是一种效率更高的工作方式。

TCP 和 UDP 同时在传输层使用的事实说明：计算机网络中的效率与可靠性同样重要，对两者的选择应综合考虑底层线路的工作条件和应用内容的实际需求，在局域网下用 UDP 可能效率更高，在 Internet 下使用 TCP 则更可靠。前者更适于一次传输交换少量报文的情况(如交易性应用)，后者则更适于一次传输交换大量报文的情况(如文件传输)。

在内核支持 TCP/IP 协议的操作系统中，一种称为 Socket(套接字)的机制被用来实现对内核网络功能的编程调用。Socket 面向客户-服务器方式，对服务器一方，需要拥有一个固定的 IP 地址和一个固定的 Socket 端口(即要求使用一个包括协议名、本地地址和本地端口号的描述形式)，这样客户一方才能向它发出连接请求和信息请求，完整的连接描述应包括协议名、本地地址、本地端口号、远地地址和远地端口号。在完成 Socket 连接后，客户就可通过操作系统提供的读、写功能对远程服务器收、发数据了。

### 5.4.2　测控系统对网络通信的基本要求

为了说明虚拟仪器联网通信的工作方式，有必要分析一下测控系统对通信的基本要求。

首先，作为测控设备的虚拟仪器可能需要与远程的其他用户连接，接受其他用户的控制，并将测试结果数据实时地传送回去。这种情况常常被看做是实时监控，希望通信双方能够互连数据通道，像计算机测控系统中惯用的总线 I/O 操作那样完成网络通信任务。为此可能需要直接在 TCP/IP 的传输层通过 Socket 的通信机制进行。

其次，有时测试结果数据和参数设定数据并不需要立刻进行传输。如一天的电能测量，需要在虚拟仪器主机本身计算后累积形成数据，再提供给其他用户。此时的电子邮件也是一种可行的方案，因为这种方式不需要通信双方同时开机。数据提供者在其开机时通过电子邮件将信息上传到为其服务的 ISP，再通过网络发往接收者的 ISP，直至接收者开机时从其 ISP

下载这一信息，其间并不需要通信双方的直接连接。

另外，有时测控系统要求传输的数据量很大，如图像、资料、长时间的波形数据等，且这样的数据在发送端往往已经以文件形式存放，故在 TCP/IP 网络中采用 FTP 文件传输的服务方式就很合适。

最后需要强调说明的是：无论是何种情况，对要求通信的双方，均希望能以函数调用的方式来自动实现其收、发功能，均应尽量避免传统的对话操作，以便将其通信功能嵌入到独立开发和运行的测试系统软件中。为此，在开发虚拟仪器联网通信功能时，可以根据测控功能的实际需要和网络通信的方式限制，定义有关类库函数的形式和参数。在诸如 VB 一类的开发工具中，类变量一旦被定义，就能在编程界面下自动给出该类所包含的所有函数及其参数的名称和数据类型。因此，只要精心设计这些函数和参数的名称，调用者即使不用说明书也能猜出这些函数的功能及其参数的意义。表 5-7 列出了为此目的开发的一些典型类函数。表中函数及其参数经过选择只保留最必要的设置，通常运行时有关参数采用默认值，同时也为这些参数另行设计了参数设置函数(但通常不必使用)，从而既方便了普通测试工作者联网通信的一般要求，也能满足少数专业的特殊需要。

**表 5-7 用于虚拟仪器网络通信的典型类函数**

| 端口操作方式的网络通信类 CommBySocket | |
|---|---|
| 连接控件 | SetObj(winsock As Object) |
| 建立服务器 | SetupServer(ip$ , port&) |
| 建立客户连接 | Connect(ip$ , port&) |
| 取消客户连接 | DisConnect( ) |
| 发送数据 | SendData(txt$ ) |
| 接收数据 | RecvData(txt$ ) |
| 电子邮件方式的网络通信类 CommByEMail | |
| 连接控件 | SetObj(MAPISession As Object, MAPIMessages As Object) |
| 上传邮件 | SendMail(addr$ , sbj$ , txt$ ) |
| 下载邮件 | RecvMail(addr$ , sbj$ , txt$ ) |
| FTP 文件传输方式的网络通信类 CommByFTP | |
| 连接控件 | SetObj(Inet As Object) |
| 上存文件 | SaveFile(url$ , port%, usr$ , psw$ , dir$ , fl$ ) |
| 下载文件 | LoadFile(url$ , port%, usr$ , psw$ , dir$ , fl$ ) |

上述函数调用前应事先在应用程序的 Form 中建立所需要的控件，然后通过函数 SetObj 将其与类变量对象链接起来。

在端口操作方式的 CommBySocket 类中，函数 SetupServer 用于建立服务器，其参数 ip$ 和 port& 分别用来表示待建本地服务器的 IP 地址和端口号。函数 Connect 用于客户建立与服务器的网络连接，参数 ip$ 和 port& 分别表示欲连接的远地服务器的 IP 地址和端口号。函数 DisConnect 则用于释放已连接的网络。由于进行读、写数据操作时系统连接已经建立，因此函数 SendData 和函数 RecvData 只需设置读、写操作的数据变量及其长度。另外，故障发生时这些函数的返回值为小于 0 的错误识别码。

在电子邮件方式的 CommByEMail 类中，SendMail 和 RecvMail 分别为实现发送和接收信息的基本功能函数。发送过程当然需要指出发送的地址、主题和发送的内容，因此函数 SendMail 中的参数也就设置这样几项。对接收函数 RecvMail，要求只将符合对方地址和主题

(参数 addr$ 和 sbj$ )的内容接收到字串变量 txt$ 之中。

由于虚拟仪器并不需要浏览 FTP 服务器的目录，因此针对 FTP 文件传输的 CommByFTP 类中可以只用 SaveFile 和 LoadFile 两个函数来上传和下载指定的文件。这里，参数 url$ 用于表示访问网络中 FTP 服务器指定信息的位置，如“FTP：//ftp. microsoft. com/movie”。Port& 为该服务器开放的端口(通常为 21)，参数 usr$ 和 psw$ 分别为访问该服务器所需要的用户名和密码，如果该服务器提供匿名访问，则可将这两个参数设为空字符。若连接出现由于用户名或密码错误而导致无权访问等故障，则返回值小于 0。参数 dir$ 表示传输文件的本地目录，参数 fl$ 表示上传或下载的文件名。

### 5.4.3　远程实时监控的端口操作实现方法

为了实现端口操作方式建立的网络通信类函数 CommBySocket 的各种功能，需要使用一个 VB 自身提供的控件 Winsock。由于这类控件相对操作系统内核的功能调用具有简单方便的特点且由编程工具直接提供，安全可靠，故可作为开发虚拟仪器联网通信的工作基础。使用 Winsock 控件不必了解 TCP 协议的细节，也不需要直接调用底层 Winsock API 函数，只要适当设置控件属性、处理控件事件和调用控件方法，就可编写客户或服务器程序，方便地使用 TCP 和 UDP 支持的网络功能。表 5-8 给出了 Winsock 控件的部分属性、方法和事件的简要说明。网络通信类函数 CommBySocket 的具体源程序见程序清单 5-4。

表 5-8　Winsock 控件的部分属性、方法和事件

| | | |
|---|---|---|
| 属性 | LocalHostName | 返回本地计算机名 |
| | Protocol | 使用的通信协议，0—TCP(默认)，1—UDP |
| | RemoteHost | 返回或设置远地计算机(通过计算机名) |
| | RemoteHostIP | 返回或设置远地计算机的 IP 地址 |
| | RemostePort | 返回或设置要连接的远程计算机端口号 |
| | LocalIP | 返回或设置本地计算机的 IP 地址 |
| | LocalPort | 返回或设置用到的本地端口 |
| | State | 返回控件的状态 |
| 方法 | Accept | 仅适用于 TCP 服务器应用程序，接受新连接 |
| | Bind | 指定用于 TCP 连接的 LocalPort 和 LocalIP |
| | Close | 关闭 TCP 连接或侦听套接字 |
| | Listen | 创建套接字并将其设置为侦听模式，该方法仅适用于 TCP 连接 |
| | GetData | 获取当前的数据块并将其存储在变体类型的变量中 |
| | SendData | 将数据发送给远程计算机 |
| 事件 | ConnectRequest | 连接请求时触发 |
| | Connect | 建立连接时触发 |
| | Close | 关闭连接时触发 |
| | DataArrival | 数据到达时触发 |
| | Erro | 发生错误时触发 |

**程序清单 5-4 CommBySocket. cls**

```
Dim o As Object, flg as integer
Public Function SetObj(winsock As Object)                    '建立控件连接
    Set o=winsock
End Function
Public Function SetServer(ip$ ,port&)as Integer              '设置服务器
  flg=0:SetServer=0
  If o.State<>0 Then SetServer=-3:Exit Function
    o.Protocol=0:o.LocalPort=port&:o.Listen                  '设定 TCP 协议
    Do Until o.State=2                                       '处理事件
        DoEvents:If flg<>0 Then SetSever=-2:Exit Function
    Loop
End
Public Function Connect(ip$ ,port&)as Integer                '建立客户连接
    flg=0:o.RemoteHost=ip:o.RemotePort=port
    o.LocalPort=0:o.Connect
    Do Until o.State=7
        DoEvents:If flg<>0 Then Connect=-1:Exit Function
    Loop
    Connect=1
End Function
Public Sub Disconnect()                                      '取消客户连接
    o.Close
End Sub
Public Function Send(txt$)As Integer                         '发送数据
    flg=0:Send=0:If o.State<>7 Then Send=-1:Exit Function
    If flg<>0 Then Send=-2:Exit Function
    o.SendData txt$
End Function
Public Function Recv(txt$)As Integer                         '接收数据
    flg=0:Recv=0:If o.State<>7 Then Recv=-1:Exit Function
    If flg<>0 Then Recv=-2:Exit Function
    o.GetData txt$
End Function
Private Sub sock_ConnectionRequest(ByVal requested&)
    If sock.State<>sckClosed Then sock.Close
    o.Accept requestID
End Sub
```

在函数 SetSever 中，首先通过控件的 State 属性检查网络的连接情况，若已连接其他用户，则退出，否则即可开始设置控件的属性。设定 Protocol=0 表明选择通信协议为 TCP。通过控件属性 LocalPort 的设定建立本机服务器的监听端口，然后通过调用控件的 Listen 方法使服务器端进入侦听状态。此时如果有客户端发起连接，则会激发 Winsock 控件的 Connection-Request 事件。作为对该事件的响应，应通过控件的 Close 方法关闭当前的侦听，再通过控件的 Accept 方法接受连接，并在连接完成后通过函数返回值报告结果。函数 Connect 用于客户端连接，通过控件属性 RemoteHost、RemotePort 设置欲连接的远地服务器的 IP 地址和开放的端口号，然后通过调用控件的 Connect 方法发起连接。函数 Send 和 Recv 首先通过控件属性

State 监测当前网络的连接状态，确认连接后再分别通过控件的 SendData 和 GetData 方法来发送和接收数据。函数 Disconnect 通过控件的 Close 过程来释放网络的连接。

### 5.4.4　数据传输的电子邮件实现方法

VB 提供的消息应用程序接口(MAPI)可将用于电子邮件的工具软件和消息服务的机制连接起来，因此可以通过使用 MAPI 程序和 Microsoft Outlook Express 邮件代理软件来实现虚拟仪器的网络通信功能。MAPI 由两个控件组成，MAPISession 控件用于建立一个 MAPI 会话，而 MAPIMessages 控件为用户提供可执行的消息交换功能。这里巧妙地使用这两个控件，从而达到通过应用电子邮件代理软件完成虚拟仪器远程网络通信功能的目的。具体实现过程如程序清单 5-5 所示。

**程序清单 5-5　CommByEmail. cls**

```
Dim o1 As Object:Dim o2 As Object
Public Sub SetO(oMAPISession as object,oMAPIMessages as object)
    Set o1=oMAPISession:Set o2=oMAPIMessages
End Sub
Public Function SendMail(addr$,sbj$,txt$)     '上传邮件:设置地址、主题、内容
  If o1.SessionID>0 Then o1.SignOff
  o1.DownLoadMail=False:o1.SignOn:o2.SessionID=o1.SessionID:o2.Compose
  o2.RecipDisplayName = addr$:o2.MsgSubject = sbj$:o2.MsgNoteText = txt$:
o2.Send:o1.SignOff
End Function
Public Function RecvMail(addr$,sbj$,txt$)     '下载邮件:设置地址、主题、内容变量
    If o1.SessionID>0 Then o1.SignOff
    o1.SignOn:o2.SessionID=o1.SessionID:o2.FetchUnreadOnly=True:o2.Fetch
    o2.MsgIndex=0:Recv=-1
  Do Until o2.MsgIndex>o2.MsgCount - 1
        DoEvents
        If o2.MsgOrigAddress=addr And o2.MsgSubject=sbj Then
          txt=o2.MsgNoteText:Recv=0:Exit Do
        End If
        o2.MsgIndex=o2.MsgIndex+1
    Loop
    o1.SignOff
End Function
```

函数 SendMail 和 RecvMail 都要求首先通过 MAPISession 控件的属性 SessionID 判断是否控件正在进行会话，关闭已经存在的会话，开启新的会话。对 Send 函数，需要通过设置 MAPISession 控件的 DownLoadMail 属性来避免邮件代理进程启动后首先执行检查并接收邮件的过程。然后，通过控件的 SignOn 方法来开启与邮件代理软件的会话。同时将得到的会话句柄赋值给 MAPISession 的 SessionID 属性，使其获得邮件代理的进程。进而通过执行消息交换来调用邮件代理软件的功能，即首先通过 MAPIMessages 控件的 Compose 方法来清空发送缓冲区并设置邮件索引，然后分别设置属性 RecipDisplayName、MsgSubject 和 MsgNoteText 来指定邮件的收件人地址、主题和正文内容。接着通过 Send 方法来执行邮件发送功能。最后通过控件 MAPISession 的 SignOff 方法关闭当前的会话。

对 Recv 函数，应当首先通过设置控件 MAPIMessages 的 FetchUnreadOnly 属性来确保缓冲区接收的邮件仅为未读过的邮件，然后通过控件的 Fetch 方法来获得邮件。再在循环语句中通过匹配属性 MsgOrigAddress 和 MsgSubject 来检索指定发送人和主题的邮件，并将找到的邮件内容转存到函数的参数中。

### 5.4.5 文件传输的 FTP 实现方法

VB 自带的 Inet 控件可以用来实现网络通信中最为广泛应用的 HTTP 协议和 FTP 协议。如程序清单 5-6 所示，两个函数功能(DownLoad 和 UpLoad)都要求首先设定控件的 Protocol 属性来选用 FTP 协议。然后分别设定控件的属性 url、RemotePort 、UserName 、Password 来指定所要连接 FTP 服务器的资源位置、开放的端口和用户名及密码，而且都需要调用控件的 Execute 方法来执行上传和下载任务。这里，Execute 方法的第一个参数可以省略，第二个参数字串中的 PUT(或 GET)分别表示上传或下载，后面跟随本地文件的目录和文件名，并用空格隔开。执行过程中通过控件的 StillExecuting 属性来判断文件传送是否结束，网络通信发生故障后会触发控件事件 StateChanged，需要在此控件事件的响应程序中通过调用类函数来设置标志 flg，从而引起 UpLoad 函数的退出。退出的实际执行是通过调用控件的 Cancel 方法和 Excute 方法来释放此次 FTP 连接的。

**程序清单 5-6 CommByFTP. cls**

```
Dim o As Object, flg as integer
Public Sub Seto(oInet as object)
  Set o=oInet
End Sub
Public Function UpLoad(url$ ,port&,usr$ ,psw$ ,dir$ ,fl$ )
                                '上传函数:路径,端口,用户,口令,目录,文件
  o.Protocol=2:o.url=url:o.RemotePort=port:o.UserName=usr:o.Password=psw
  o.Execute ,"PUT " & dir & fl & " " & fl
  Do While Inet.StillExecuting
    DoEvents:If flg< 0 Then UpLoad=-1:Exit Do
  Loop
  o.Cancel:o.Execute ,"CLOSE"
End Function
Public Function DownLoad(url$ ,port&,usr$ ,psw$ ,dir$ ,fl$ )
                                '下载函数:路径,端口,用户,口令,目录,文件
   o.Protocol = 2: Inet.url = url $ : o.RemotePort = port: o.UserName = usr:
o.Password=psw
  o.Execute ,"GET " & " " & dir$ & " " & fl$
  Do While Inet.StillExecuting
    DoEvents:If flg< 0 Then DownLoad=-1:Exit Do
  Loop
  o.Cancel:o.Execute ,"CLOSE"
End Function
Private Sub InetStateChanged(ByVal State As Integer)
                                '应为控件的 StateChanged 事件调用
     If State=11 Then flg=-1  '传递控件的 StateChanged 事件
End Sub
```

### 5.4.6　远程操作的工作示例

某虚拟仪器测试系统用于测试激光器噪声与内部温度的关系曲线，测试现场的虚拟仪器与试图分析这条曲线的研究人员相距千里，为此虚拟仪器提供两项内容：

（1）现场测试工作一方通过调用函数

```
Public s1 as New CommBySocket, s2 as New CommByFTP
s1.SetServer("202.118.235.182",500)
```

将自己设置成服务器方式，并进行网络侦听。网络连接后通过调用函数

```
s1.Recv(cmd$)
```

每秒钟读取一次来自远方的命令（放在字串参数 cmd$ 中），并根据命令进行测试参数的设置。然后测取一条特性曲线，存入文件 d:\data\050501.dat 后通过调用函数

```
s2.Upload("FTP://ftp.share.com/Data",21,"user1","4949","D:\Data","050501.dat")
```

上传到网络中的 FTP 服务器中。

（2）远方研究人员处的虚拟仪器测试系统面板与测试现场的几乎相同，只是操作命令并不发给本机的底层仪器执行功能，而是通过如下程序代码

```
Public c1 as New CommBySocket,c2 as New CommByFTP
c1.Connect("202.118.235.182",500)
```

连接到远方计算机，然后通过调用函数

```
c1.Send(cmd$)
```

将命令 cmd$ 的内容发往远地作为服务器的虚拟仪器测试设备，再通过函数

```
c2.DownLoad("FTP://ftp.share.com/Data",21,"user1","4949","E:\Data","050501.
dat")
```

从 FTP 服务器取回测得的曲线数据。

上述简短的函数代码为仪器远程通信指出一种利用网络功能的基本方法，实际构建软件时还应注意一些具体问题。例如，在用户窗体中需要建立相应的控件，并在窗体响应控件事件的代码中调用库函数程序等。

### 5.4.7　嵌入式系统中的网络接口与应用方法

TCP/IP 网络的推广不仅普及了 PC 上网，而且也使嵌入式系统上网呈现出快速发展的趋势。人们很自然地想到建立一个支持 TCP/IP 网络功能的通用接口模块或微控制器模块，预先实现远程网络传输的底层操作管理。这样，仪器系统只要将这个模块接入自己的嵌入式系统，就可以通过十分简便的函数调用实现测试信息的远程网络传输。这里介绍的 AS312 模块是万博公司开发的一种嵌入式应用产品，其特点是体积小（36mm×26mm×15mm），使用方便（不需要任何外围器件），提供的 UART 串行接口采用 TTL 电平，可以直接连接嵌入式系统中的微控制器。AS312 采用 11 脚单排插针式封装，其引线管脚定义如图 5-19 所示，其中引脚

9~12 连接以太网的 RJ-45 插头，引脚 3 和引脚 4 分别连接串口接收和发送。

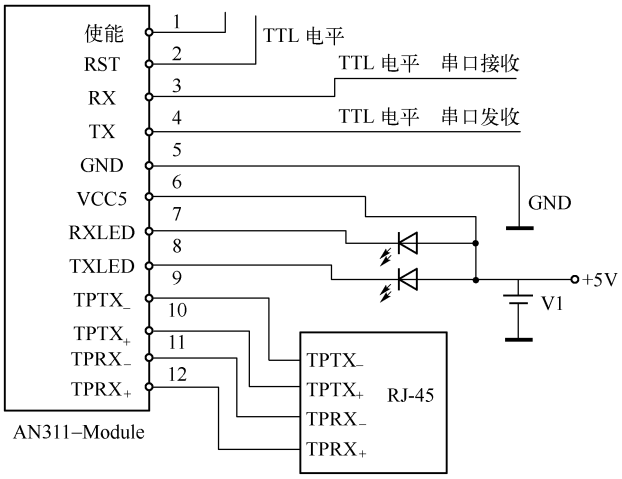



图 5-19　嵌入式网络模块的连接

AS312 的操作也很简单，按图连接并接入网络后即可发现发光二极管闪动，与其连接的 PC 就会产生网络已连接的提示，表明网络工作正常，进而可以通过串口进行数据通信：PC 下行传输数据时可从网上向 AS312 的 07CFh(1999)端口直接发送数据包，AS312 收到数据包后会根据先前允许其发往串口的配置(为默认情况)，将这些数据通过串口发给模块所在的底层测试功能模块。上行传输则可由测试系统的微控制器直接向 AS312 串口发送数据，AS312 会把数据通过 UDP 协议发送到网上指定主机的指定端口上。除了通过串口收、发数据，也可以通过串口设置参数，其通信格式如表 5-9 所示。设置过程应在 AS312 串口通信模式与用户设定一致的情况下进行(默认值为波特率 1200，偶校验，8 位数据位，1 位起始位，1 位停止位)，设置完成后 AS312 也将通过串口返回设置的参数状态。

**表 5-9　AS312 嵌入式网络模块操作命令格式**

| 标　志　码 | 标　识　码 | 设 置 数 据 | 说　　明 |
|---|---|---|---|
| BE EB CD<br>加<br>S(53h) | I(49h) | 0A. 0D. 52. F9 | AS312 的 IP 地址，0A. 0D. 52. F9 (10. 13. 82. 249) |
| | M(4dh) | 00. 00. 00. 22. 33. 33 | 设置 AS312 的 MAC 地址 00. 00. 00. 22. 33. 33 |
| | P(50h) | 56 78 | 设置 AS312 的访问密码为 56，78 |
| | T(54h) | FF DC<br>20 | 设置 AS312 的串口定时器值<br>串口校验方式(00—偶，10—奇，20—无) |
| | G(47h) | 0A 0D 52 02<br>FF FF FF 00 | 设置网关 IP 为 10. 13. 82. 2<br>子网掩码为 255. 255. 255. 0 |
| | S(53h) | 0A 0D 52 0D<br>07 E0<br>02 | 设置远端服务器 IP 为 10. 13. 82. 13<br>接收端口<br>AS312 接收限制标志 |
| | R(52h) | | 读取 AS312 中的配置信息 |

串口发送 R 命令后，AS312 将返回 31 字节的状态数据，其格式如表 5-10 所示。如返回值为：BE EB CD 0A 0D 52 F9 00 00 00 22 33 44 FE E0 20 0A 0D 52 01 FF FF FF 00 0A 0D 52 0D 07 CF 02，则表明 AS312 的 IP 是(10. 13. 82. 249)，MAC 是 00 00 00 22 33 44，AS312 的串口定时器值是 FEE0h(表示串口速率为 1200bps)，串口无校验(空格 20)，AS312 的网关设置是(10. 13. 82. 1)，子网掩码设置是(255. 25. 255. 0)，服务器 IP 是(10. 13. 82. 13)，端口是 07CFh，AS312 没有接收限制(02 标志)。

**表 5-10　AS312 嵌入式网络模块状态返回的数据格式**

| 起始码 | IP | MAC | 定时器值 | 校验方式 | 网关 | 子网掩码 | 服务器 IP | 服务器端口 | 接收标志 |
|---|---|---|---|---|---|---|---|---|---|
| BE，EB，CD | 4 | 6 | 2 | 1 | 4 | 4 | 4 | 2 | 1 |

简单的网络仪器结构框图如图 5-20 所示，其中控制器模块以 ARM 机 LPC2104 为核心，直接通过串口操作网络模块 AS312，再由 RJ-45 连接外部网络(局域网或因特网)。

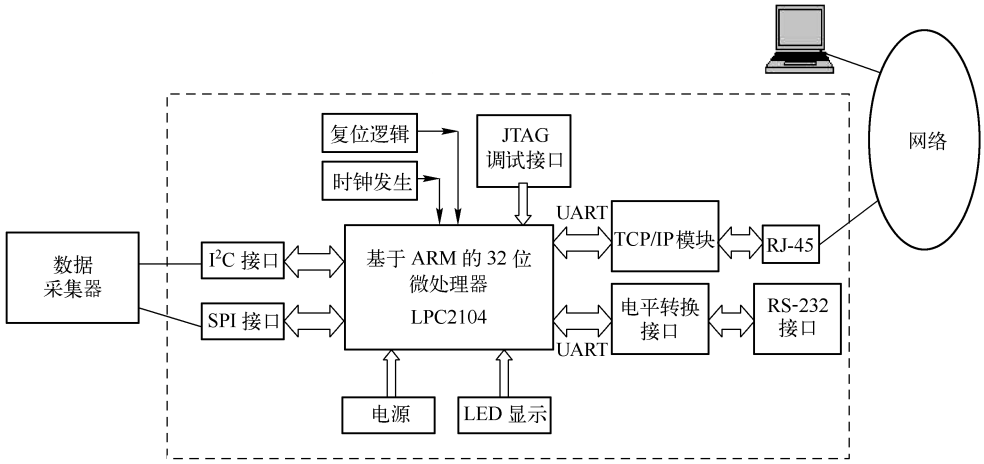



图 5-20　网络仪器结构框图

LPC2104 的功耗很小，功能却很强，可以通过 JTAG 接口进行程序调试和故障检测，其供电电源分正常和待机两组，后者可以保持片内计时日历的正常运转。数据采集功能的控制和数据传输分别由 $I^2C$ 和 SPI 串行接口实现，前者的半双工方式灵活，便于控制通道调理电路中的多个器件；后者速度较快，可与多种 A/D 转换器接口直接连接。控制器的实际外形结构由图 5-21 给出，具体实现数据采集的 A/D 转换和通道调理电路可安放在小盒结构右侧的电

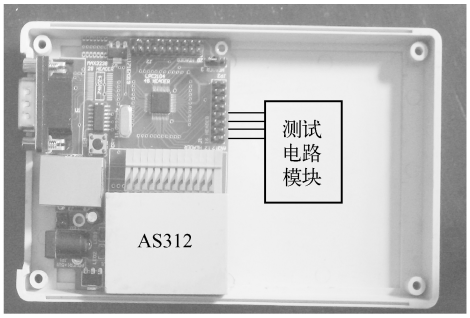



图 5-21　含网络接口的嵌入式微控制器模块配置方式

路模块中，模块之间可由扁平电缆连接。

程序清单 5-7 给出了这种控制器实现网络数据传输的基本函数，其中初始化函数 Init 用于配置模块本身的参数，函数 Send 可将数据发送到网络中指定的远程计算机上，函数 Recv 则可通过模块的串口直接得到来自网络接收的数据。SioR 和 SioW 为固件中串口操作的底层函数。应用程序只需调用清单中的 3 个基本函数，即可实现测试数据的网络传输。

**程序清单 5-7 对 AS312 的控制函数**

```
Init(char * sMac,char * sIP,char * sNG){
  s1[0]=0BEh;s1[1]=0EBh;s1[2]=0CDh;
  s=strcmp(s1,"SM");SioW(strcat(s,sMac));
  s=strcmp(s1,"SI");SioW(strcat(s,sIP));
  s=strcmp(s1,"SG");SioW(strcat(s,sNG));
}
Send(char * ip_ep_flg ,char * d){
  s=strcmp(s1,"SS");SioW(strcat(s,ip_ep_flg));
  SioW(d);
}
Recv(char * d){
  SioW(d);
}
```

## 5.5 用于远程测试的无线通信

随着无线通信技术的发展，各种无线通信设备及其网络业务在人们的工作和生活中正在发挥着越来越重要的作用，因此在建立远程操作的自动测试系统时，自然应当考虑这种无线通信的工作方式。本节将以十分有限的篇幅，在介绍有关无线通信背景的基础上，说明其在自动测试中的作用和应用。

### 5.5.1 GSM 和 SMS 手机短信业务的基本概况

数字蜂窝移动通信 GSM（Global System for Mobile Communication）工作在 900MHz 和 1800MHz，是国内目前覆盖面最广、用户最多的移动通信系统，而且随着 GSM 网络基站越建越密，在城市中已基本做到无缝覆盖。依托于该通信网，无须专门架设通信平台，也无须为频点资源的申请而费心。GSM 移动通信系统提供了多种业务，包括话音业务、短消息业务、数据业务等。针对远程操作的自动测试系统的要求，GSM 网络提供的服务中最引人注意的是 SMS 和 GPRS。短消息服务 SMS（Short Message Service）以短消息业务中心作为短消息的存储转发站，可实现移动终端间的点对点短消息通信；通用分组无线业务 GPRS（General Packet Radio Service）采用分组交换技术，可以更有效地利用资源，信道上提供 TCP/IP 连接，可使用户永远在线并按流量计费，从而降低了服务成本。

GSM 话音业务和数据业务需要通过拨号建立连接，且一旦连接后就开始计费，故很不经济。而 GSM 的短消息业务不需要建立拨号连接，只需把要发的信息加上目的地址发送到短消息中心，再由短消息中心转发到最终目标。短消息每次限制在 160 字节以内，这对

很多需要远程传输的数据信息监控要求(如气象、水文、环保仪器的定时定点在线监测，供电、用水、煤气等基础能源供应的自动抄表)是完全能够满足的。短消息还有很多其他优点，如短消息发送占用的是信令通道，当移动接收台占线而无法通话的情况下仍可接收短消息。当移动接收台关机或不在服务区时，短消息中心会保存该条短消息并建立等待标志，一旦用户开机或返回服务区，GSM 系统会及时通知短消息中心重发，使用户即刻收到该条短消息。

能够应用上述 GSM 系统进行远程监测的工作很多，如图 5-22 所示的油井管理远程监控就是一个典型示例。采油队通常每天都要巡井检查抽油机的运行情况，记录电机运行、电能消耗、井口流量等重要参数。这些数据的种类不多，数据量不大，巡检的频度要求也不高，但由于有些油区的油井分布偏远、分散，因此建立自动监测系统很具实际意义。实际系统的建立应分为数据采集和远程通信两部分，且尽可能地使它们相对独立。这样，原先有关仪器功能本身的电路设计依然不变，可用典型的嵌入式数据采集器实现；而远程通信的模块则可选用现成商品，它们之间的连接方式往往就是采用本章开头介绍的 UART 协议和 RS-232 接口。

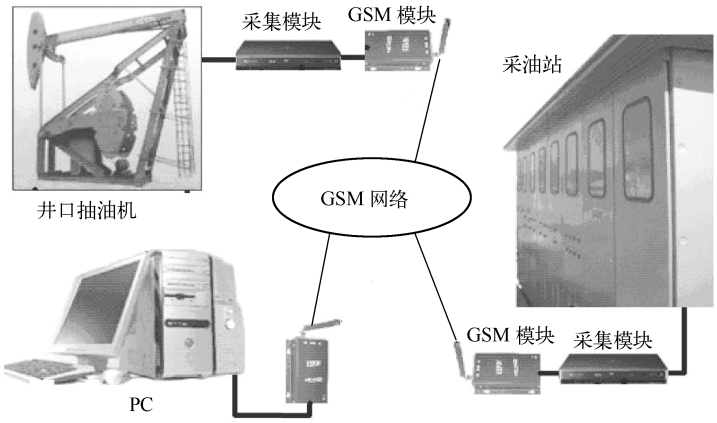



图 5-22　油井监测的无线系统

### 5.5.2　GSM 模块的应用方法

短消息业务按其实现方式可以分为点到点短消息业务和小区广播短消息业务。点到点短消息业务是通过 MO(MS 发起短消息业务)和 MT(MS 终止短消息业务)将一条短消息从一个实体发送到指定目的地址的业务，被发送的信息经过编码后的消息长度为 140 字节，即一条短消息可以包含 160 个英文字母(7 位编码)或 80 个中文汉字(Unicode 编码)。小区广播短消息业务是指通过发送信息的基站向指定区域中所有短消息用户发送短消息的业务。

可供选择的 GSM 模块种类很多，它们都有短消息功能，但是考虑到系统的可升级性，可以选用支持 GPRS 功能的 GSM 模块。市场上流行的 GSM 模块有西门子公司的 MC35、摩托罗拉公司的 G18 和索尼爱立信移动通信公司的 GM47 等。GPRS 模块在发射时的瞬间电流很大(2A 左右)，因此配置电源时一定要留有余地，以免电压波动太大，影响 GPRS 模块工作的稳定性。在 GPRS 模块工作时，一旦电源跌落到它的下限，模块就会自动关机。G18 可直接连

接+5V 电源供电，最为方便。MC35 的供电电压为 3.3~4.8V，可采用+5V 电源串接二极管的方式供电。GM47 的供电范围较窄，仅为 3.4~4.0V。GM47 和 MC35 的外形尺寸差不多，G18 则较大，但其内部集成了一个 SIM 卡座。

模块连接的基本方式如图 5-23 所示。MC35 模块的电源必须留有余地，因为其波动过大会引起模块自动关机。单片机对 MC35 模块的控制和数据通信均通过串口实现，但 MC35 模块采用 CMOS 电平，故与采用 TTL 电平的单片机之间需要进行电平转换。MC35 模块在启动时需要一个启动时序电路来控制其 IGT* 引脚，正常工作后则会通过 PD* 引脚输出方波，紧急情况还可用 PD* 引脚复位。为了调试方便，系统应配置一个状态指示灯，可利用 MC35 的 SYNC 引脚进行控制。每当 MC35 模块扫描网络或收、发短信息时，指示灯就会闪烁。

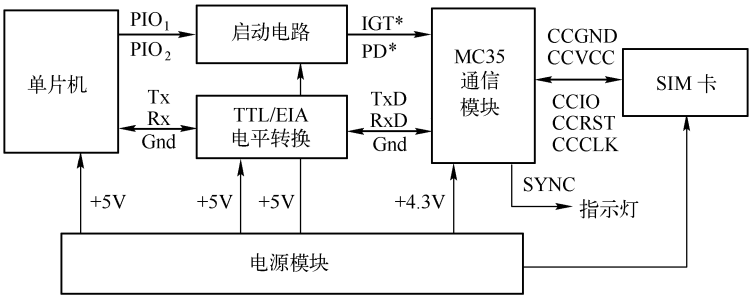



图 5-23 基于 MC35 的无线通信网络仪器结构框图

采用上述模块，可以构建一个外形独立的通信模块设备，也可直接选用适当的手机，通过配置的数据线连接到 PC 主机的串口或 USB 接口，然后编写软件通过接口直接操作通信模块或手机，上报数据和状态。注意，尽管有时厂家所配数据线连接 PC 一端的接口是 USB，但实际配置相应设备驱动程序后仍可用 VB 开发环境下的串口工作方式进行操作。

### 5.5.3 串口编程操作的要点

短消息可以分为 Block、Text 和 PDU 3 种模式。其中，Block 模式需要厂家提供驱动支持；PDU 模式发送 SMS 的消息正文需要十六进制编码，移动台发送短消息的帧包括头标记、消息参考号、目的地址、协议表示、有效时间、用户数据等元素，而 PDU 模式接收需要进行解码处理；Text 模式最为方便，虽然它不支持中文，但用于远程控制的测试系统传输测试数据也并不需要使用中文。

多年来 AT 命令被广泛用于调制解调器的操作，后来为了通过终端设备(TE)或计算机来操作移动通信文本消息，又出现了一种称为 SMS Block Mode 的协议。几年后，诺基亚、爱立信、摩托罗拉和 HP 共同为 GSM 研制了一整套 AT 命令，其中也包含对短信息 SMS 的控制。AT 命令在此基础上演化，形成 GSM07.05 标准。计算机和 GSM 引擎之间采用这种 AT 指令实现它们之间的通信，AT 指令集的命令格式帧都以字符 AT 开头，以<CR>结束，且绝大多数应答语句的开头和结尾都采用<CR><LF>，即回车换行符。表 5-11 列出了与发送和接收短消息直接相关的一些 AT 命令。

表 5-11　计算机和 GSM 引擎之间通信的 AT 指令

| AT 指令 | 功　能 |
|---|---|
| AT+CMGC | Send an SMS command(发出一条短消息命令) |
| AT+CMGD | Delete SMS message(删除 SIM 卡内存的短消息) |
| AT+CMGF | Select SMS message formats(选择短消息信息格式) |
| AT+CMGL | List SMS message from preferred store(列出 SIM 卡中的短消息) |
| AT+CMGW | Write SMS message to memory(向 SIM 内存中写待发的短消息) |
| AT+CNMI | New SMS message indications(显示新收到的短消息) |
| AT+CMSS | Send SMS message to memory(从 SIN/M 内存中发送短消息) |
| AT+CPMS | Preferred SMS message storage(选择短消息内存) |
| AT+CSCA | SMS service center address(短消息中心地址) |
| AT+CSCB | Select cell broadcast message(选择蜂窝广播消息) |
| AT+CSMP | Set SMS text mode parameter(设置短消息文本模式参数) |
| AT+CSMS | Select Message Service(选择短消息服务) |
| AT+CMGR | Read SMS message(读短消息) |
| AT+CMGS | Send SMS message(发送短消息) |

为了成功地发送和接收短消息，必须在单片机和 GSM 模块之间建立通信链路，下面以 MC35 模块操作为例，说明单片机使用 AT 指令操作 GSM 模块的方法。

(1) 在初始化过程中单片机通过启动电路启动 MC35 模块后通过发送 ATE 命令来确认，如果 MC35 模块确实启动，则返回信息“OK”。

(2) 发送 AT 命令“AT+CMGF=1”，设置 MC35 模块为文本方式。

(3) 发送 AT 命令“AT+CSCA=+8613800531500(短消息中心号)”，设置短消息中心，若设置正确，则模块返回“OK”。

(4) 发送 AT 命令“AT+CMGS=13022445115(SIM 卡号码)”，建立通信号码并返回信息“>”后，再依次发送有关测试数据的短消息。

(5) 通过发送 AT 命令“AT+CMGL=ALL”来查询短消息命令，当确有短消息接收时模块返回如下格式：

+CMGL:<序号>,<短消息状态>,<SIM 卡号码>,,<日期时间><CR><LF><短信内容><CR><LF>

注：其中连续出现的两个逗号之间可为某些特定模块配置专门的参数。

(6) 删除短消息。

通过 AT 命令“AT+CMGD=<Index>”后，即可删除序号为 Index 的短消息。

### 5.5.4　无线通信网络 ZigBee

ZigBee 是一组基于 IEEE 802.15.4 无线通信标准的组网、安全和应用软件方面的技术标准。与其他无线通信标准(如 802.11 或 802.16)不同，802.15.4 以 250kbps 的最大传输速率

承载有限的数据流量。ZigBee 则为一种新的，具有低功耗、低成本、低复杂度、短时延、高容量、高安全的近程无线网络技术，其名源于蜂群使用的通信方式（蜜蜂通过跳 ZigZag 形状的舞蹈来表达发现的信息）。IEEE 802.15.4 小组与 ZigBee 联盟分别制订硬件与软件标准，两者的角色分工如同 IEEE 802.11 小组与 WiFi 的关系。802.15.4 仅定义网络通信实体层和介质访问层，ZigBee 则在 802.15.4 标准的基础上为不同产品之间共享资源提供了应用的平台。

ZigBee 在工作模式下传输数据量小，因此信号的收/发时间很短；另一方面，其在待机模式下休眠，且从休眠转入工作状态只需 15ms，节点连接进入网络只需 30ms（蓝牙和 WiFi 均需 3s 以上），这样就使节点设备非常省电，2 节 5 号电池可支持工作 6~24 个月（与此比较，蓝牙为数周、WiFi 为数小时）；ZigBee 协议紧凑简单（不到蓝牙的 1/10），从而降低了对通信控制器的要求，以 8051 的 8 位微控制器测算，全功能的主节点代码需 32KB、子功能节点少至 4KB，而且 ZigBee 免协议专利费，因此实际成本明显较低。

ZigBee 工作在 IEEE802.15.4 的频段是免费开放的，采用 ZigBee 技术的产品可以在 2.4GHz 上提供 250kbps（16 个信道）、在 915MHz 上提供 40kbp/s（10 个信道）、在 868MHz 上提供 20kbps（1 个信道）的传输速率。为了避免在开放频段下工作受其他无线设备的干扰，各个频段均采用直接序列扩频技术。在 2.4GHz 频段采用较高阶的 QPSK 调制技术以达到 250kbps 的传输速率，并降低工作时间，以减少功率消耗；而在 915MHz 和 868MHz 频段，则采用 BPSK 调制技术，由于频率相对较低，无线传播损失较少，传输距离较远，且因室内无绳电话的频段现已转到 2.4GHz，干扰可能反而比较少。

ZigBee 可采用星状和网状网络结构，也允许两者组合形成丛集树状。一个主节点最多可管理 254 个子节点，同时主节点还可由上一层网络节点管理，最终可组成多达 65 000 个节点的大网。根据节点的角色不同，可分为全功能设备（FFD，Full-Function Device）与精简功能设备（RFD，Reduced-Function Device）。RFD 电路较为简单且存储体容量较小，只能传送数据给 FFD 或从 FFD 接收数据。FFD 节点具备控制器的功能，能够提供数据交换，且要求它支持所有 49 个基本参数。ZigBee 提供了三级安全模式，包括无安全设定、接入控制清单（ACL）来防止非法获取数据，以及采用属于高级加密标准（AES128）的对称密码来保护数据，防止攻击者冒充合法器件。ZigBee 以 250kbps 的通信速率工作，能够满足测试系统低速率数据传输的应用需求。传输范围（指相邻节点间的距离）一般为 10~100m，在增加 RF 发射功率后可增加到 1~3km。如果通过路由和节点间通信的接力，传输距离可以更远。

目前 ZigBee 技术提供的主要方式有 3 种：

（1）ZigBee-RF 器件+MCU，如 TI 公司开发的 CC2420+MSP430、FREESCLAE 的 MC13XX+GT60、MICROCHIP 的 MJ2440+PIC MCU 等；

（2）单芯片集成 SoC，如 TI 开发的 CC2430/CC2431（用 8051 内核），FREESCALE 的 MC1321X、EM250 等；

（3）单芯片内置 ZigBee 协议栈+外挂芯片，如 JENNIC SOC+EEPROM、EMBER260+MCU 等。

图 5-24 是一种使用 ZigBee 器件 CC2420 的电路原理图，无线通信的所有硬件功能均由专用芯片 CC2420 完成，单片机采用美国 Silicon Lab 公司的 C8051F121，固件需依靠该公司提供的协议栈函数库。

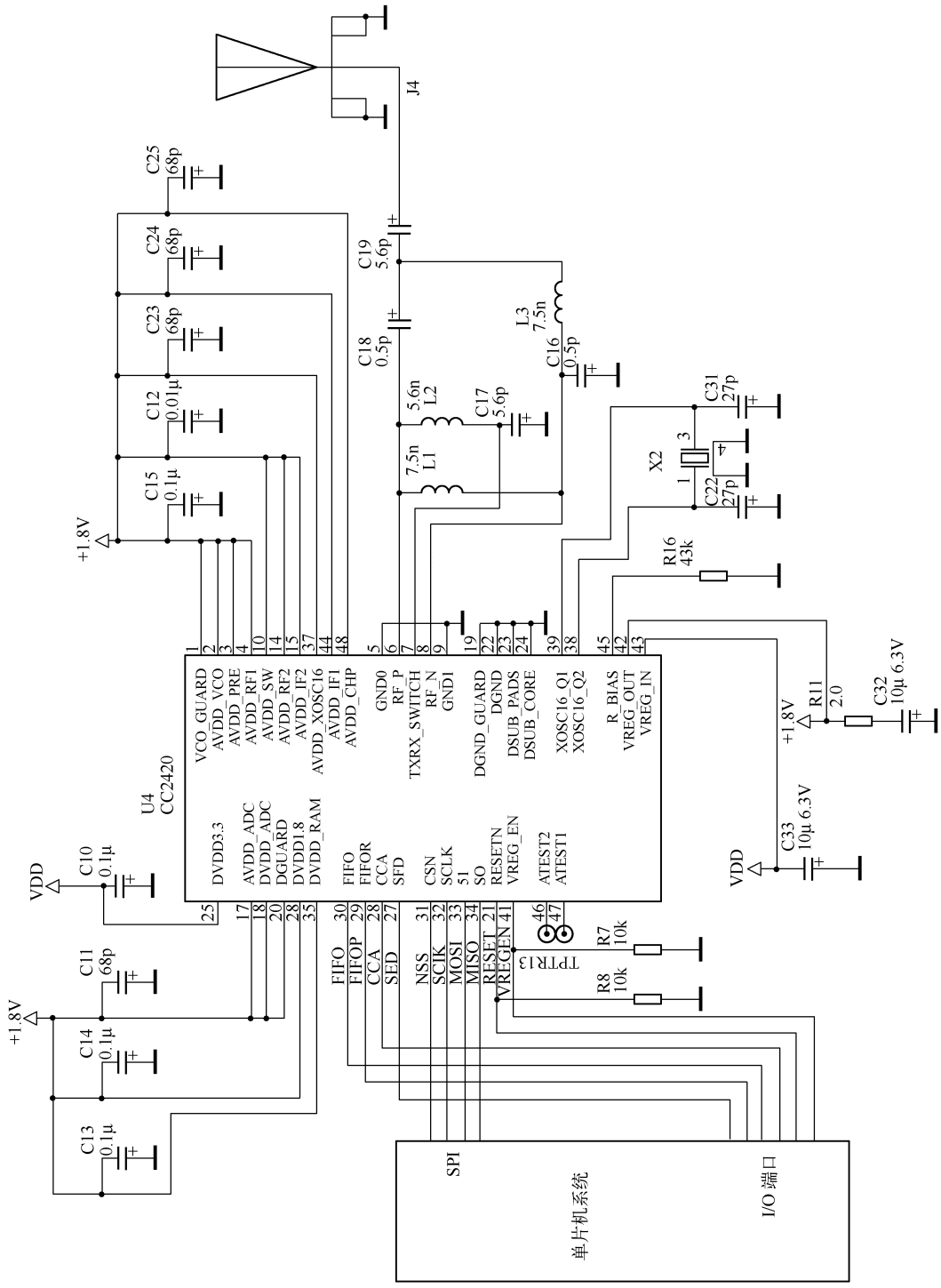



图 5-24　基于 CC2420 的无线通信模块电路原理图

CC2420 的 4 线 SPI 接口与 C8051F121 的 SPI 端口直接相连(前者作为从器件，后者作为主器件)，CC2420 的其他 6 条控制线分别连接到 C8051F121 的 P1 端口。其中帧起始界定信号 SFD 被连接到 C8051F121 的 P1.0，这个引脚被配置为输入捕捉方式下 PCA 的一个输入捕捉引脚 CEX0，用来戳记输入帧的时间，并与标识帧同步。先进先出的缓存器 FIFO 的入口信号 FIFOP 被连接到 C8051F121 的 P1.2，它被配置成外部中断信号 INT0*，用来启动一帧数据的接收过程。上述电路的 PCB 布局和走线需要很好地考虑，一方面要使单片机电路和无线电路之间能够很好地隔离，另一方面则要确保 RF 本身的高频性能。PCB 分 4 层，RF 部分的布线要特别小心，建议严格根据芯片 CC2420EM 参考设计资料进行复制。ZigBee 技术的核心是 ZigBee/802.15.4 的 C51 源代码，这些代码和 ZigBee 芯片配合，完成数据包装收/发、校验、各种网络拓扑、路由计算等复杂的功能。从技术开发的角度看，选用熟悉的单片机，采用熟悉的硬件结构、指令系统和开发环境，自然便于掌握 ZigBee 的代码编程，以集中精力应对复杂的无线通信协议和超高频的硬件环境。

测控系统应用 ZigBee 技术的前景十分广泛，针对那些频繁移动、出线困难的场合(如运动的机械装置、人体生理的指标监测等)，当其与传感器结合后常常能在更为复杂的环境中为自动测试系统提供更全面的原始数据。

## 本章小结

UART 是通用的异步串行通信协议，作为很多器件进行控制和数据交换的基本方式。RS-232 是数据终端设备(DTE)与数据通信设备(DCE)之间的通信协议，应用在测试系统时需要特别注意仪器设备和计算机在协议中的角色。鉴于 PC 串口在 VB 控件的支持下可以直接进行编程操作的特点，以及各种网络、有线和无线通信设备对 UART 和 RS-232 串口的支持能力，使这种工作方式不仅能够直接处理串口和调制解调器通信的一般问题，也在解决现场总线、网络传输和无线通信等其他远程测控系统的应用过程中发挥了重要的作用。鉴于 PC 成熟的联网技术，虚拟仪器自然应当充分利用，而且需要将其嵌入到测试系统之中。同时，为在远程条件下确保通信的可靠性，也应对数据传输过程中的编码校验算法和通信协议予以足够重视。

## 参考文献

[1] Jordan, Larry E. Communications and Networking for the IBM PC. 1983.
[2] Joe Campbell. Programmer's Guide to Serial Communication. Howard W.Sanms & Company, 1988.
[3] 张载鸿. 微型机(PC 系列)接口控制教程. 清华大学出版社, 1992.
[4] Stephen J. Bigelow. Understanding Telephone Electronics. Hayden Books, 1991.
[5] Visual Basic Component Tools Guide. Microsoft Corporation, 1997.
[6] 周明天, 汪文勇. TCP/IP 网络原理与技术. 清华大学出版社, 1993.
[7] 谢希仁. 计算机网络(第 3 版). 大连理工大学出版社, 2000.
[8] 刘思久, 王芳林. 虚拟仪器的远程联网通信功能开发. 国外电子测量技术, 2005(8).
[9] 刘宁远. 一种适于 VB 编程环境的串口通信服务器. 电测与仪表, 2002(12).
[10] Computer System Interface (iSCSI) Cyclic Redundancy Check (CRC)/Checksum Considerations. RFC

3385. 2002.
[11] 李文仲, 段朝等. ZigBee 无线网络技术入门与实战. 北京航空航天大学出版社, 2007.
[12] 张海霞, 曹宇辉. RS-232 串口多方控制的实现. 现代电力电子技术, 2007.
[13] 刘伟, 崔永锋, 吴相林. TCP/IP 协议在远程单片机控制系统的设计. 单片机接口技术, 2007.
[14] 张爱科. 基于 TCP/IP 的工业设备联网设计方案. 微计算机信息, 2008.
[15] 卢海峰, 江朝元, 阳小光. 基于串口通信的在线监测系统关键技术研究. 仪器仪表学报, 2006(z3).
[16] 张占新, 孟如, 孟庆洪, 张伟芹. 基于 MC35 的无线数据采集系统. 测控自动化, 2005(7-1).

# 第 6 章　基于 USB 的虚拟仪器

随着现代仪器中含有的信息功能越来越多，仪器与计算机之间的界限变得越来越模糊，于是成本相对低廉、能够充分发挥通用计算机强大功能的虚拟仪器便越来越快地成为现实。PC 的计算功能、存储功能、图形功能、数据处理功能，以及多媒体、数据库、电子表格、TCP/IP 网络通信等通用功能可以极为方便地解决诸如面板操作、信息存储、数据处理、联网通信、报警报数、图示指导和机箱电源等现代仪器的各类需求，只有仪器本身最本质的信号激励功能和信号测量功能需要配置专门的数据采集设备。

鉴于本书在第 1 章已经阐明 USB 作为 PC 主机与数采外设之间一种理想接口的基本概念，本章将进一步分析 USB 的工作原理和协议要点，具体讨论 USB 的接口电路和操作程序，同时结合实例说明 USB 虚拟仪器的开发方法。

## 6.1　USB 的数据通信原理和协议要点

USB 作为目前 PC 连接外设的主要方式，其性能优越、使用方便，但其工作原理和协议内容都比较复杂，需要专门讨论。

### 6.1.1　USB 系统底层的树形结构

尽管 USB 具有总线方式的通信功能，但 USB 系统在物理构造上却是有层次性的树形结构，USB 主机内部支持一个根集线器，可以直接连接各 USB 设备，也可通过一级级的集线器(HUB)级联后再连接到 USB 测试设备(如第 1 章图 1-7 所示)。

USB 集线器不是一个简单的连接器，也不只是主机和设备之间上传、下载的通道，它还直接担负着检测外部设备的连接和断开，并向主机逐级报告的重要任务。如图 6-1 所示，当 USB 设备与 USB 集线器(Hub)相接时，Hub 中原先被下拉电阻拉成低电平的信号线就会被设备中的上拉电阻(全速方式时接 D+线，低速方式时接 D-线)拉成高电平，从而发现某个设备正待加入系统，并层层上报直至主机。主机由此了解到新连接设备的位置后，就会通过层层集线器发布命令来迫使该设备进入复位状态，于是该设备将自己的地址设成一个编号为 0 的特殊地址(称为默认地址)，并暂时通过这个默认地址接收主机的请求命令，开始一个称为“列举”(Enumeration)的初始化组态过程。

为了在逻辑意义上实现主机对各设备自由寻址的总线传输目的，USB 仍然需要为每个设备确定一个独立的设备地址(Address)。由于传输方向和传输内容的类型不同，USB 进一步为每个设备定义了若干端点(End Point)，这些端点实际上就是设备接口中一些先进先出的数据缓冲区通道。这样，主机就有可能通过对设备地址和端点的寻址，直接访问不同设备的不

同数据缓冲区，建立主机与设备端点之间的通信联系。

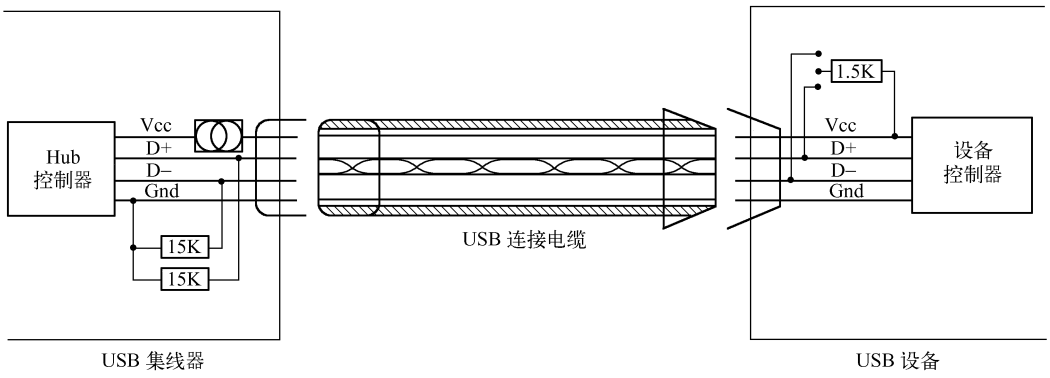



图 6-1 USB 集线器和 USB 设备之间的连接

USB 的总线传输以半双工的方式工作，节点之间的通信采用异步方式，它没有专门的时钟信号线，而是将时钟信号内置于传输信号之中，依靠有规律的信号变换来维持接收器与发送器之间的同步。

### 6.1.2 差分信号与总线状态

USB 总线底层物理信号的传输方式与微计算机外设常用的其他总线不同，它采用一种特殊的电平逻辑规定。简单地说，当信号线 D+的电平高于信号线 D−的电平时，定义差分驱动器的输出信号为“1”；反之，当信号线 D−的电平高于信号线 D+的电平时，定义差分驱动器的输出信号为“0”。严格地说，USB 总线差分驱动器输出数据信号不仅要求两根信号线之间的电位差大于 0.2V，而且还要求其绝对输出电位不能低于 2V。

如图 6-2 所示，USB 总线信号的有效电平规定并非只有差分方式，也存在一种让 D+与 D−两根信号线的绝对电平同时为低的“单端 0”(SE0)状态。

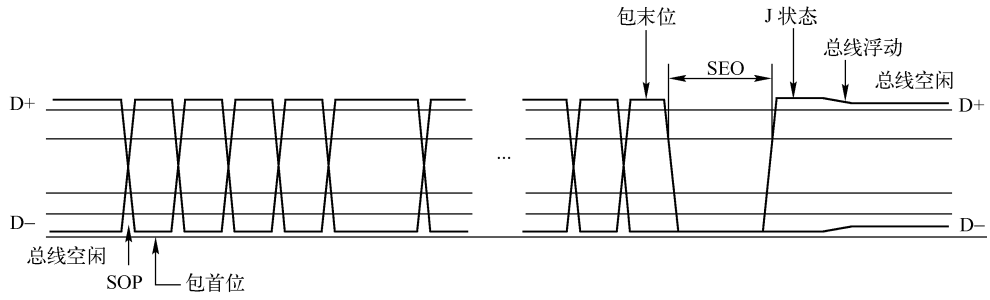



图 6-2 USB 差动驱动器输出信号

USB 差分驱动输出电平状态所表示的意义在全速方式与低速方式下并不相同(读者不妨先只考虑全速方式)。为使进一步的数据编码不受速度方式的影响，可以将上述差分驱动输出的具体“1”、“0”状态用通用的“J”、“K”状态表示，即

D+高、D−低的“1”差分驱动输出表示：全速的“J”、低速的“K”

D−高、D+低的“0”差分驱动输出表示：全速的“K”、低速的“J”

USB 数据传输采用一种称为不归零变换(NRZI)的编码方法，对实际传输信号(不是指差分驱动输出)中的每个“0”信号，编码规定为差分驱动输出的 J-K 切换状态，而信号“1”编码规定为差分驱动输出的 J-K 延续状态。这样的安排在遇到连续多个“1”组成的数据序列时，可能会因总线波形长时间不变，而使接收器容易产生“失步”。于是，USB 规范又规定每当连续传输 6 个数据“1”后，需要强制性地加入一个额外的数据“0”来迫使差分驱动输出的 J-K 状态跳变一次，从而确保信号的同步作用不会丢失。为了帮助读者理解，可以仔细体会下面所列的这种编码的若干示例：

01010111000111111101 => (J)KKJJKKKKJKJJJJJJJKKJJ

101001111110111111101 => (J)JKKJKKKKKKKJKKKKKKKJJKK

这里，下画线处的 J-K 状态切换是为产生同步而强制加入的额外信号，而非对应需要传输的数据。由此右边实际传输的 J-K 状态位数可能多于左边需要传输的数据位数。

除了基本的数据信号外，USB 总线还需要设定一些特殊的控制状态：如空闲态 Idel(源端无作用，受端则与 J 态相似)、启动传输 SOP(从空闲态到 K 态)、结束传输 EOP(SE0 态下保持 2 位时间后转入 J 态并持续 1 位时间)、连接通信(空闲态下持续 2ms)、断开连接(SE0 态下持续 2.5μs)、复位操作(SE0 态下持续 10ms)、挂起恢复(进入 K 态)等。

USB 上述这些编码和解码通常由传输方和接收方的专用接口电路完成，因此了解这些过程只是为了便于观测和理解总线上实际物理信号的信息意义，而对直接使用 USB 信号接口芯片(支持收/发功能)的系统设计并非十分重要。

### 6.1.3　传输事务和信息包的基本结构

USB 的数据传输过程比一般串口通信复杂得多，实际规范将一个完整的传输过程(transfer)分解成若干个传输事务，而组成传输事务的基本数据结构是一些类型不同的信息包。

如图 6-3 所示，信息包由起始段、信息段和结束段 3 部分组成。所有信息包都从同步(SYNC)字段开始，同步字段从空闲状态(J)开始，以“KJKJKJKK”的二进制位序列紧跟。通过这个二进制位串序列，输入电路的同步机制将本地时钟调整到与输入数据同步的状态。同步字段里的最后 2 位是同步字段结束的记号，同时指出信息实质内容的开始。包信息实质内容中的第 1 个字节总是包标识符(PID)，用来标识包的类型，并据此解释后面的信息内容。信息包的最后部分是包结束标识符，即在两位传输时间内以信号线 D+和 D-均降为低电平所表示的 SE0 状态来指明包的结束。

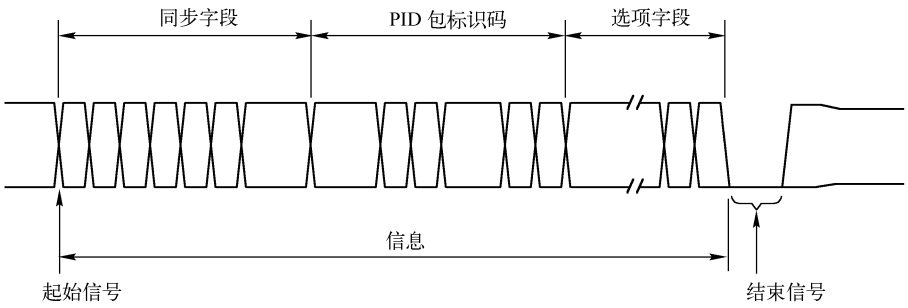



图 6-3　信息包的组成波形

如图 6-4 所示，USB1.1 规定的信息包依据其 PID 分为 4 类 10 种。PID 字节由 4 位标识信息和 4 位补码补充位组成，PID 的这种冗余结构使接收器能够对其进行纠错检测。

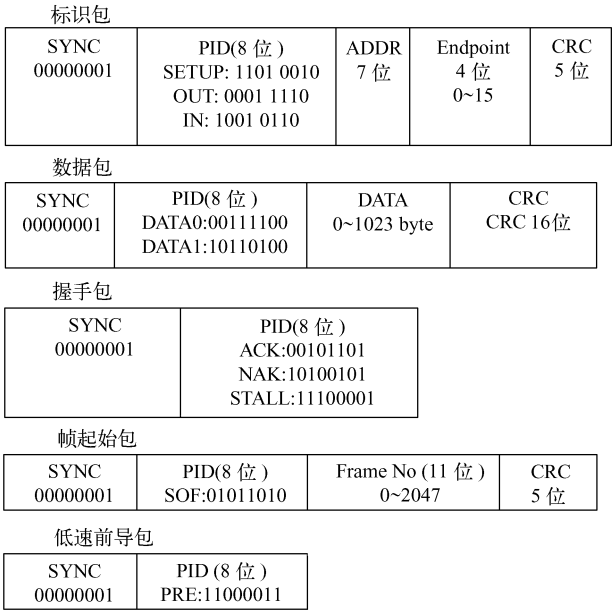



图 6-4 USB 的包结构

标识包分为 Setup 包、Out 包和 In 包 3 种情况，分别用于表达传输的目的。在 USB 数据传输过程中，由于所有传输共享一个数据通道，因此需要主机通过标识包对有关传输对象进行寻址。主机首先通过总线，以广播方式发出一个寻址信息块(包括标识包中 7 位设备地址码和 4 位端点号)，其后只有与此寻址信息匹配的设备和端点方与主机建立相应的数据传输通道，包内的 5 位 CRC 校验码用于提高包传输的可靠性。数据包可分为 Data0 包或 Data1 包两种情况。在多个传输事务连续传输的过程中，要求依次出现的数据包，其 PID 在 Data0 和 Data1 之间中交替切换，以此作为校验传输是否正确的又一种措施。握手包可分为 ACK、NAK 和 STALL 3 种包，分别表示数据传输接收方对当前数据传输的认可、否认和因功能未备而无法完成的不同响应。另外，还有两种包类型，它们的 PID 分别是 SOF 和 PRE，分别用于帧起始标志和指示低速的前导标志。

传输事务是实现传输的基本结构，通常总是由标识包、数据包和握手包组成。标识包总是由 USB 主机发往外部设备，当一个传输事务的标识包以 IN 包方式出现时，表示后面跟随的数据包将从外设传向主机，再后面的握手包将从主机传回外设，这样的传输事务称为 IN 事务。同理，在 OUT 事务中用 OUT 包作为标识包，表达了其后的数据包将从主机传往外设，随后相应的握手包则从外设传回主机。标识包为 SETUP 包的 SETUP 事务是一种用于系统管理的特殊事务，其数据包总是从主机发往设备，而数据包的内容则是标准的 USB 系统命令，如要求读取 USB 设备描述符、为设备分配设置 USB 地址等。

### 6.1.4 数据传输的帧结构和传输类型

USB 数据传输的过程始终是在主机控制下完成的，是主机把时间分成 1ms 间隔的传输

帧，并在每帧的开始时刻将称为 SOF 的帧起始包广播发出。同时，也是主机把每个传输帧的时间区间，以传输事务为单位一一分配给各个传输任务。由于系统往往包括多个 USB 设备，且每个设备也往往不止包括一种传输任务，因此同一个传输帧内往往需要包括不同设备的不同传输任务所包含的不同传输事务。同时，传输任务通常由多个事务组成，因此组成同一传输任务的多个事务有可能需要放在前后不同的多个传输帧中进行传输。

如图 6-5 所示，系统中 USB 设备由其设备地址码(Device Address)来识别，而同一设备的不同传输任务则由设备接口的不同端点(Endpoint)来操作。为了实现对各 USB 设备的实时控制，USB 主机应在每一个传输帧中，首先安排那些要求实时操作的接口管理信息和有实时、同步要求的数据传输信息；然后再在剩余的时区内继续安排那些与时间没有直接对应关系的其他数据传输信息。这一过程称为分配带宽。

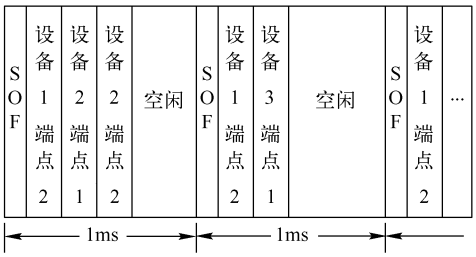



图 6-5　USB 数据传输的帧结构

总线上的数据传输使用控制传输、中断传输、批量传输和同步传输 4 种传输类型，每种传输都有其特定的格式和适合其特殊用途的传输协议。USB 设备的不同传输方式是通过对设备端点进行不同的配置来实现的，且配置一旦选定，传送方式也就确定了。

控制传输主要用于执行诸如读取设备描述信息、选择列举设备、配置地址等 USB 标准接口命令，是 USB 设备必须支持的一种标准传输方式。控制传输也能用于执行一些非标准的 USB 类命令，但它的传输效率不高，因此对一些特定要求的数据传输可能需要采用其他一些传输方式。

批量传输适于那些数据量较大、希望传输速率快、但实时性并不十分重要的场合，例如，向打印机输出一个文件或从一个扫描仪接收数据。此时如果总线正忙于其他必须进行的传输，批量传输就可以停下来等待，直至总线空闲时再继续进行批量传输。

中断传输的适用特点与批量传输正好相反，它适于那些传输速率不是很高、但需要主机能够给予较快响应的场合。例如，鼠标或键盘可以通过中断传输方式与主机进行实时通信。USB 的每个传输帧要为每个中断端点安排一次中断事务，从而确保该端点 1ms 的响应时间。

同步传输用于那些要求严格按照同一速度传输数据的场合，如一个要求实时播放的录音或摄像设备。同步传输不支持自动检查传输错误，并实现重发纠错的功能。

表 6-1 总结了上述各种传输方式的特点和用途(表中括号内的数据为低速模式下的情况)。使用 USB 处理计算机各种外围设备时，可以根据传输速率、反应时间和错误纠正的不同要求，选择不同的传输方式。

**表 6-1　USB 传输类型及其特点**

| 传输类型 | 控　制 | 批　量 | 中　断 | 同　步 |
|---|---|---|---|---|
| 典型用途 | 设备配置 | 扫描仪、打印机、存储设备 | 鼠标、键盘 | 音像 |
| 必备性 | 是 | 否 | 否 | 否 |
| 每毫秒传输字节数(全速) | 832 (13×64) | 1216 (19×64) | 64 | 1023 |

续表

| 传输类型 | 控　制 | 批　量 | 中　断 | 同　步 |
|---|---|---|---|---|
| 每毫秒传输字节数(低速) | 24(3×8) | 无 | 0.8(0.1×8) | 无 |
| 数据流方向 | 双向 | 单向 | 单向 | 单向 |
| 保留的带宽(最大) | 10% | 没有 | 90%(同步和中断合并) | |
| 进行纠错 | 是 | 是 | 是 | 否 |
| 确保传输率 | 否 | 否 | 否 | 是 |
| 确保响应时间 | 否 | 否 | 是 | 是 |

### 6.1.5 控制传输的工作过程

在 USB 的传输过程中，控制传输方式主要用于主机读取和选择列举设备的配置，以及其他有关的系统设置，所有 USB 设备都必须支持控制传输方式。控制传输包括配置、数据和状态 3 个阶段(Stage)。其中，配置阶段由一个 Setup 传输事务构成，Setup 事务包括一个 Setup 包(作为标识包)、一个数据包(此时所传的数据为 USB 标准或非标准的请求命令)和一个握手包组成。数据阶段则由若干个普通的数据传输事务组成，用以传送应用任务真正需要传输的数据。这里，所谓普通的数据传输事务是指由 IN 包或 OUT 包作为标识包的传输事务，标识包后跟随数据包和握手包。状态阶段也使用一个数据传输事务，通常它由数据阶段最后一个数据传输事务的数据接收者发出，作为在此层次上数据接收者对数据传输过程的一种回应方式。

USB 传输过程中所用的数据结构层次如图 6-6 所示，除了控制传输需要特殊的 Setup 事务外，其他中断、批量和同步传输只是相对简单地重复使用 IN 事务和 OUT 事务。

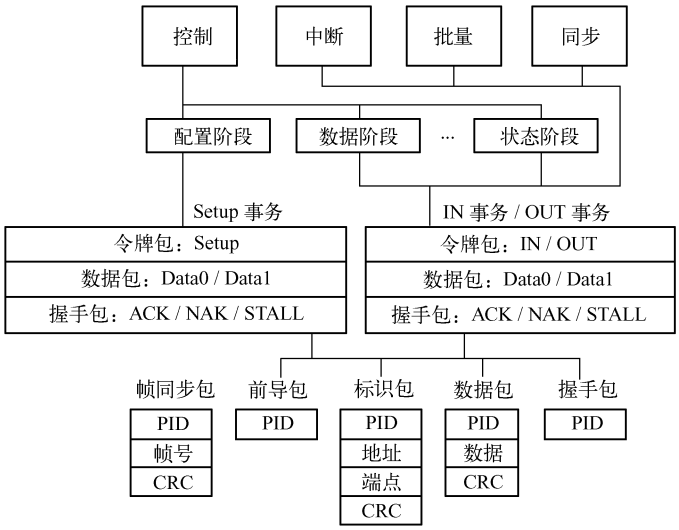



图 6-6　USB 信息传输过程及其数据结构

### 6.1.6　USB 系统工作的两个阶段

USB 通信分为列举(Enumerate)阶段和应用阶段。列举阶段是指进行实际应用数据信息交换之前为主机了解设备所做的准备工作，以及根据设备要求进行系统配置的过程；而应用阶段则指主机和设备之间为实际应用所进行的数据交换过程。

列举阶段主要发生在开机初始化的过程和设备突然插拔的过程中，此刻设备先将自己的地址暂时设成编号为 0 的默认地址，并通过该地址接收主机的请求命令。然后再由主机根据整个系统的组成情况为该设备分配一个正式的地址，并同样通过请求命令的方式下载给这个设备。此后主机对该设备的通信均将以此设备地址为基础，通过一系列请求命令进一步了解该设备的能力和特性，并选择该设备的配置方式。而设备则通过接收和执行来自主机的一系列标准的请求命令，最终完成该设备的配置任务。

列举阶段的主要工作是主机通过发布请求命令，从被列举的设备处读取一系列有关设备信息的描述符。其中最重要的信息可能要属设备生产厂家和产品的识别码了，主机往往据此确定所需要的设备驱动程序。这样，为了完成列举，主机系统需要两个文件：一个是设备驱动程序本身，它可能包含在 Windows 系统内，也可能由外围设备厂家提供；另一个则是识别设备驱动程序位置和文件名的 INF 文件。对设备的“列举”过程是一个个设备分别完成的，主机将得到的信息保存在一个系统数据库中，以备以后应用程序使用。当列举完成时，Windows 把新设备加到控制面板的设备管理器上(如图 6-7 所示)。当用户断开外围器件时，Windows 也会从列表中自动删除这个设备。

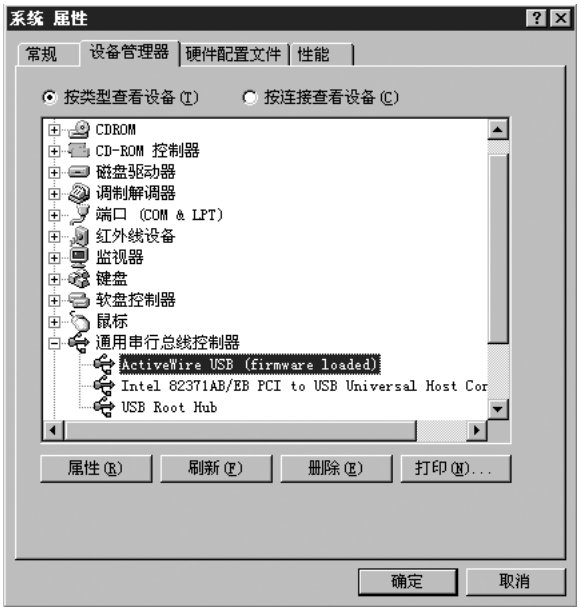



图 6-7　Windows 的控制面板

总之，USB 在列举阶段主机与设备之间的信息交换依靠 Setup 事务发出的命令和从设备读回的描述符。标准设备命令包括：读取和设置描述符(GetDescriptor 和 SetDescriptor)、设置地址(SetAddress)、设置和查询配置(SetConfiguration 和 GetConfiguration)、设置和查询接口

(SetInterface 和 GetInterface)、清除和设置特性(ClearFeature 和 SetFeature)、读取状态(GetStatus)和同步帧(SynchFrame)等项。

每个 USB 设备均应提供一个设备描述符,用于描述设备的整体信息,如设备说明的版本号、设备归类的标识码和子类标识码、协议码、端点 0 的最大包长、厂商标志、产品标志、设备发行号、字串索引(描述厂商、产品和设备序列号信息)、配置的数目等。主机收到设备描述符后,应继续取回该设备的配置描述符、接口描述符和端点描述符,字串描述符则可有可无。

每个 USB 设备至少有一个配置描述符,用来描述设备的特征和功能,这里主要是指设备应用电源和远程唤醒系统的能力,以及该配置所支持的接口数目。远程唤醒的作用是让挂起的 USB 设备告诉主机其要求通信的愿望。如果 USB 设备在 3ms 内没有总线活动,则设备必须进入挂起状态。如果已经挂起的设备上某一个事件需要主机动作,支持远程唤醒且已激活这种特征的设备就可请求主机恢复通信。配置描述符的 MaxPower 域指出设备所需总线电流的大小,其值以 2mA 为单位,可用 1 字节表达最大值为 500mA 的请求电流。例如,当设备需要 200mA 的供电电流时,可取 MaxPower = 100。若主机认定请求的电流不能得到,它将拒绝配置这个设备。

尽管大多数 USB 设备仅用一种配置和一个接口就可能满足实际要求,但多重用途和模式的设备也可以有多重配置和多个接口。接口描述符总是作为配置描述符的一部分,端点描述符则应跟在所属接口描述符后,其内容包括端点号和端点传输方向、端点支持的传输类型(00—控制,01—同步,10—批量,11—中断)、端点数据包的最大字节数、中断和同步端点的轮询频率(全速设备的中断传输取值在 1~255 之间,同步传输取值则恒为 1,而对批量传输和控制传输此值无用)。主机依据端点描述符来分配各端点的带宽。

在应用阶段,主机通过应用程序和外部设备交换通信数据。由于此时主机和设备已经交换了列举信息,并且设备驱动程序已经加载,即数据通道已经建好,因此工作过程变得相当简单。在主机侧,应用程序可以用标准的 Windows API 函数对指定设备读/写。而在设备侧,每当通过配置的端点收到来自主机的数据后就自动地将其存入一个先进先出的输入缓冲区,并产生中断事件,通知设备去处理;同时设备只需把要上传的数据放在另一个 USB 端口的输出缓冲区内,即可在主机需要时通过 IN 事务自动送往主机。

### 6.1.7 USB 系统的开发

USB 系统主要由具有 USB 接口的主机系统、支持 USB 的系统软件、基于 USB 接口的设备和形成系统扩展的集线器等几个基本部分组成。图 6-8 展示了 USB 通信模型之间基本的信息流与互联关系,其中主机与设备都被划分成不同的层次。底层是处理传送和接收 USB 信息的总线接口功能;中层用于处理与设备端点之间的数据通道关系,可被看做数据源与数据接收端之间信息交换;上层则由应用功能决定。

在基于 USB 的虚拟仪器系统中,主机往往采用 PC,其 USB 总线接口及其 USB 总线的支持软件通常已包含在操作系统中。这样,实际 USB 系统的开发,除了主机中的应用软件和设备的仪器电路,主要集中在设备接口的硬件电路与固件程序上。同时,为了有效地操作 USB 设备,有时也需要考虑主机端的 USB 设备驱动程序。

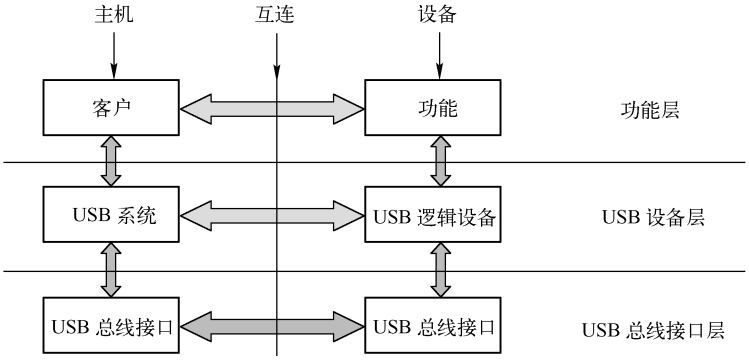



图 6-8　USB 主机-设备通信模型层次关系图

## 6.2　USB 设备接口控制器的开发

实际 USB 设备的接口功能通常被集成到专用的芯片中，这类接口芯片可以分为两类，一是利用通用的 USB 接口芯片实现通用的接口电路，然后通过电路板内的串行或并行总线与微处理器相连。这样的芯片包括美国国家半导体公司的 USBN9604、NEC 公司的 NEC2888、飞利浦公司的 PDIUSBD11/12、郎讯公司的 US820/825、ScanLogic 公司的 SL11 等。另一种方法是采用具有 USB 接口功能的单片机，如 Intel 公司的 8x930A/931A、Cypress 公司的 CY7C63xxx 系列和 EZ-USB、Motorola 公司的 68HC05BD9A、SiLab 的 F320 等。前者通用性强，可作为组件嵌入到不同的仪器控制器系统中；后者效率高，但受所用微处理器特性的限制，且需要专门的开发设备。

### 6.2.1　USB 通用接口模块的基本结构

这里介绍的通用 USB 接口微控制器模块由一个 USB 专用接口芯片 USBN9604 和一个低功耗的单片机 Cygnal8051-F231 组成，图 6-9 为此模块的工作原理图。USBN9604 功能齐备，与 USB 规范 1.1 兼容，对 USB 总线串行信号的接收和解码、标识包的解释、格式的检查、传输错误的处理，均由接口芯片本身的硬件实现。

模块采用了 4 引脚的 24MHz 晶振，产生的时钟在芯片内部倍频，以满足系统 48MHz 的过采样要求。芯片引脚$\overline{\text{RESET}}$是低电平复位，$\overline{\text{DACK}}$引脚在没有使用$\overline{\text{DMA}}$模式时接高，$\overline{\text{RD}}$引脚在 MICROWIRE 模式时也接高。工作过程不用的引脚通过一个电阻接地，以确保较低的功耗。INTR 是中断输出引脚，根据不同微处理器中断信号的电平特性要求，可以编程设置相应的中断输出模式。接口状态的检查、控制操作的实现、命令和数据的传输则由微处理器通过读/写芯片中各 USB 接口寄存器来完成。为此芯片提供了 3 种接口总线的工作模式：并行非多路复用模式、并行多路复用模式和串行 MICROWIRE 模式，以适应其与不同特点的微控制器相连。引脚 MODE0 和 MODE1 用来选择工作模式，这里，通用 USB 设备接口控制器模块选用 MICROWIRE 模式，通过 MICROWIRE 接口与主频 20MHz 的微处理器 C8051F231 连接。而微处理器的其他并/串口则可用来连接各种专用的测控电路。

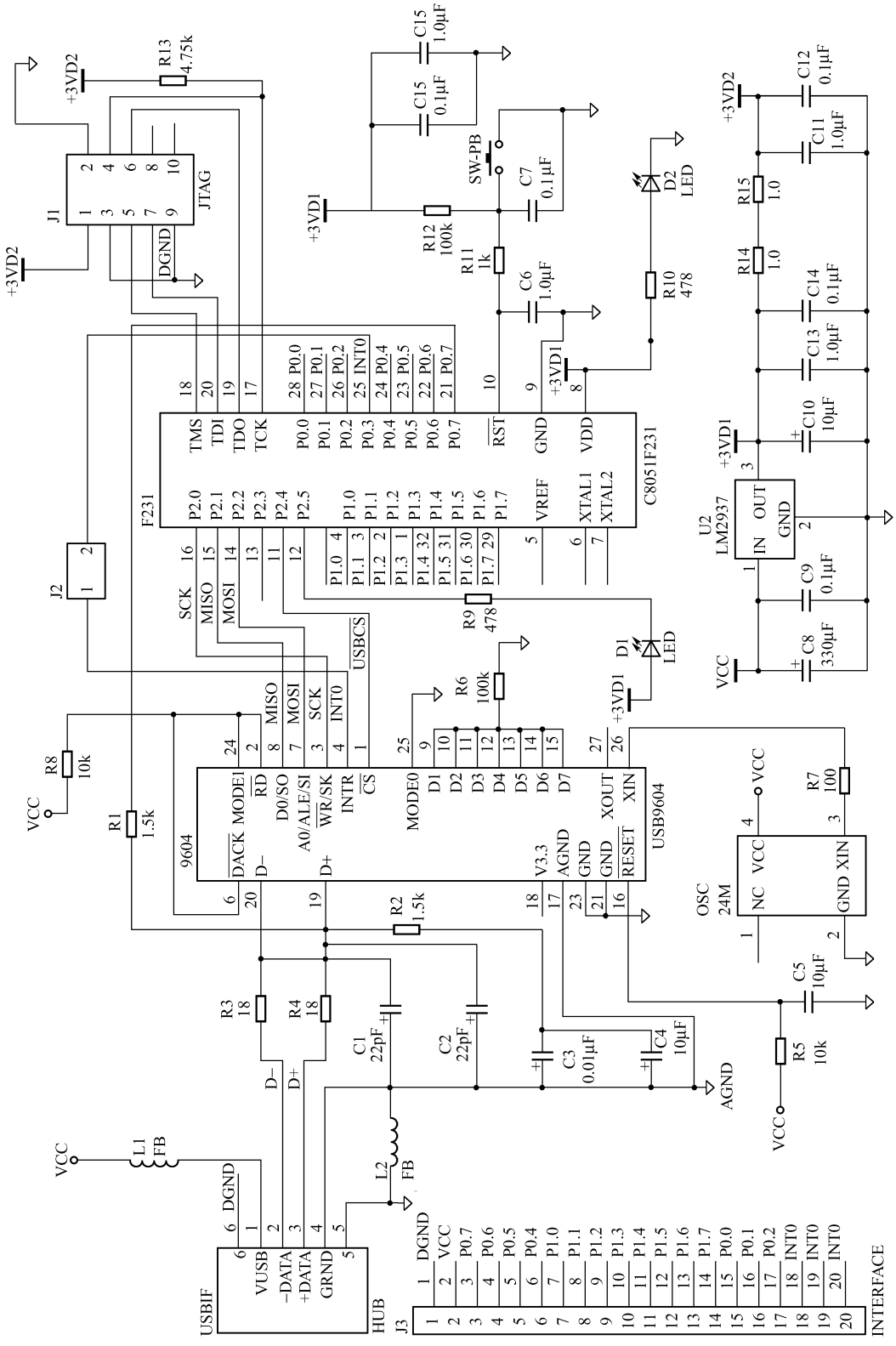



图 6–9 通用 USB 设备控制器电路原理图

### 6.2.2　核心接口芯片 USBN9604 的操作原理

与大多数基于微处理器控制的外围接口芯片一样，USBN9604 的操作只能通过微处理器读/写其内部寄存器实现。这里将 USBN9604 的主要操作寄存器一揽子地归纳列出(如表 6-2 所示)，以便给读者一个全局的概念。

表 6-2　USBN9604 主要寄存器操作功能一览表

| 功能 | 名称 | 地址 | 位 7 | 位 6 | 位 5 | 位 4 | 位 3 | 位 2 | 位 1 | 位 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 主控 | MCNTRL | 00 | INTOC[1:0] | | — | — | NAT | VGE | — | SRST |
| 时钟配置 | CCONF | 01 | CODIS | — | — | — | CLKDIV[3:0] | | | |
| 版本识别 | RID | 03 | — | — | — | — | REVID[3:0] | | | |
| 功能地址 | FAR | 04 | AD_EN | AD[6:0] | | | | | | |
| 节点状态 | NFSR | 05 | — | — | — | — | — | — | NFS[0:1] | |
| 主事件 | MAEV/MSK | 06/07 | INTR | RX_EV | ULD | NAK | FRAME | TX_EV | ALT | WARN |
| 变迁事件 | ALTEV/MSK | 08/09 | RESUME | RESET | SD5 | SD3 | EOP | — | — | — |
| 传输事件 | TXEV/MSK | 0A/0B | TXFIFO3 | TXFIFO2 | TXFIFO1 | FIFO0 | TXFIFO3 | TXFIFO2 | TXFIFO1 | FIFO0 |
| 接收事件 | RXEV/MSK | 0C/0D | RXFIFO3 | RXFIFO2 | RXFIFO1 | FIFO0 | RXFIFO3 | RXFIFO2 | RXFIFO1 | FIFO0 |
| NAK 事件 | NAKEV/MSK | 0E/0F | OUT[3:0] | | | | IN[3:0] | | | |
| 报警事件 | FWEV/MSK | 10 | RXWARN[3:1] | | | — | TXWARN[3:1] | | | — |
| 帧号高节 | FNH | 12 | MF | UL | RFC | — | — | FN[10:8] | | |
| 帧号低节 | FNL | 13 | FN[7:0] | | | | | | | |
| DMA 控制 | DMACTRL | 14 | DEN | — | — | — | DMOD | DSRC[2:0] | | |
| 端点控制 | EPC0 | 20 | STALL | DEF | — | — | EP[3:0] | | | |
| 传送数据 | TXD0 | 21 | TXFD | | | | | | | |
| 传送状态 | TXS0 | 22 | — | ACK_STAT | TX_DONE | TCOUNT[4:0] | | | | |
| 传送命令 | TXC0 | 23 | — | — | — | IGN_IN | FLUSH | TOGGLE | - | TX_EN |
| 接收数据 | RXD0 | 25 | RXFD | | | | | | | |
| 接收状态 | RXS0 | 26 | RX_ERR | SETUP | TOGGLE | RX_LAST | RCOUNT[3:0] | | | |
| 接收命令 | RXC0 | 27 | — | — | — | — | FLUSH | IGN_Setup | IGN_OUT | RX_EN |
| 端点控制 | EPC1-6 | 20 | STALL | — | ISO | EP_EN | EP[3:0] | | | |
| 传送数据 | TXD1-3 | 21 | TXFD | | | | | | | |
| 传送状态 | TXS1-3 | 22 | TX_URUN | ACK_STAT | TX_DONE | TCOUNT[4:0] | | | | |
| 传送命令 | TXC1-3 | 23 | IGN_ISOM | TFM1[1:0] | | — | FLUSH | TOGGLE | LAST | TX_EN |
| 接收数据 | RXD1-3 | 25 | RXFD | | | | | | | |
| 接收状态 | RXS1-3 | 26 | RX_ERR | SETUP | TOGGLE | RX_LAST | RCOUNT[3:0] | | | |
| 接收命令 | RXC1-3 | 27 | — | RFWL[1:0] | | — | FLUSH | IGN_SETUP | — | RX_EN |

主控寄存器 MCNTRL 中控制位 SRST 用于软件复位，控制位 VGE 用于使能内部 3.3V 电压调节器，控制位 NAT 置位时表明所在节点正作为 USB 总线上等待检测的设备，控制域 INTOC[1:0]用于设置芯片中断输出脚的电平特性。时钟配置寄存器 CCONF 中的 4 位外部时钟

分频器 CLKDIV[3:0]用于设置芯片引脚 CLKOUT 输出的时钟频率，其计算方法为：频率 = 48MHz/(CLKDIV+1)；而控制位 CODIS 则用于禁止时钟输出。版本识别寄存器 RID 中 4 位的只读控制域 REVID[3:0]用于表示芯片的版本。功能地址寄存器 FAR 中的地址有效设置位 AD_EN 为 1 时，采用控制域 AD[6:0]编码作为设备地址来识别接收的 USB 信息。

接口芯片分为复位、操作、挂起和恢复等几个操作状态，导致一种操作状态向另一种操作状态的变迁事件均由状态变迁事件寄存器 ALTEV 提出报告，包括复位事件 RESET(上行发来的 USB 呈 2.5μs 的 SEO 状态)、恢复事件 RESUME(挂起状态下检测到 USB 线上的恢复信号)、包结束信号 EOP、挂起事件 SD3 和 SD5(上行口检测空闲 3ms 和 5ms)。所有状态转变都由固件程序对节点状态寄存器 NFSR 中控制域 NFS[0:1]的设置操作来实现，具体设置操作过程应依据图 6-10 所示的端点功能状态变迁图。

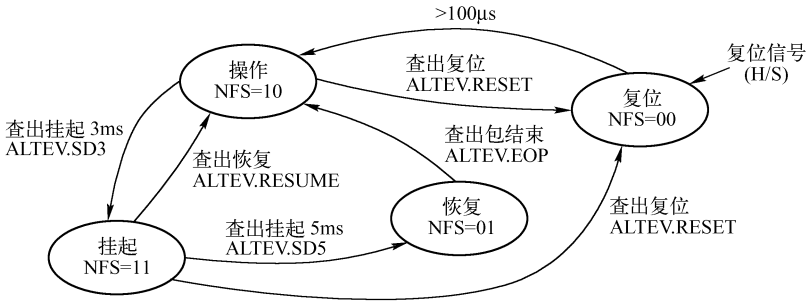



图 6-10 端点功能状态变迁

端点操作态表明设备处于正常工作状态，若在 3ms 内没有任何检测到的总线活动，挂起事件就会发生，此刻可通过设置 NFS 将节点置于挂起状态。在挂起状态，接口收发器采用一种特殊的省电工作模式，所有寄存器状态和 FIFO 缓冲区保持不变。这样一旦固件程序检测到能够唤醒总线的事件，从挂起状态中恢复到操作状态时就不需要附加的工作。挂起 5ms 后，允许固件通过改写 NFS 转入恢复态，并在查出包结束事件后回到操作态(当然也要通过固件改写 NFS)。作为对复位事件的响应，固件应通过改写 NFS 进入复位态，其后也可通过改写 NFS 进入操作态，但至少需在复位态停留 100μs。

USBN9604 使用 7 个先进先出的队列缓冲器 FIFO，包括 1 个称为控制端点 0 的双向 FIFO0、3 个用于接收的 FIFO 和 3 个用于发送的 FIFO。FIFO0 只有 8 字节，而其他 FIFO 是 64 字节。每个 FIFO 都可支持一个 USB 端点编程。端点 0 控制寄存器 EPC0 中 4 位控制域 EP[3:0]用来表示端点地址，控制位 DEF 为默认地址标志，置 1 时设备只对默认地址做出反应，而不管地址寄存器 FAR 中 AD 和端点 0 控制寄存器 EPC0 中 EP 域的内容如何。USB 复位后应把 0x80 写到功能地址寄存器 FAR 中，把 0x00 写到端点 0 的控制寄存器 EPC0 中。当一个 IN 标识包传向这个端点时，DEF 位自动清零，进行从默认地址到分配地址的转换。执行寻址命令时，必须把 0x40 写到 EPC0，再把分配的功能地址写入 FAR，然后用一个 0 长度的数据包来完成寻址控制序列的状态阶段。FAR 的控制位 STALL 用于建立 Stall 握手包(表示端口停用)。

图 6-11 所示的状态图显示了端点 0 的基本操作。如图所示，接口通常处于空闲状态，发送数据时通过写数据到端点数据寄存器 TXD0 进入 TX 满状态后，再置端点命令寄存器

TXC0 的控制位 TX-EN 位进入 TX 等待状态。在此状态下若收到的 IN 标识包就会进入 TX 发送状态，将已经写到 FIFO 中的数据直接发送出去，并在数据传输完成后回到空闲状态。而当发生错误时，则会继续保留 FIFO 中的数据，并在下一个 IN 标识包到来时重发送这些数据。如果在 FIFO 中没有数据的情况下置位 TX-EN，则可实现一个 0 长度的数据发送。已经写入 FIFO 的内容也可以通过程序固件刷新，从而用新的数据来响应下一个 IN 标识包。为了响应 OUT 标识包，也必须首先刷新 FIFO 中的原来内容，且当收到 OUT 标识包时，只有在没有错误的状态下才会通知程序固件去接收 FIFO 中的数据。

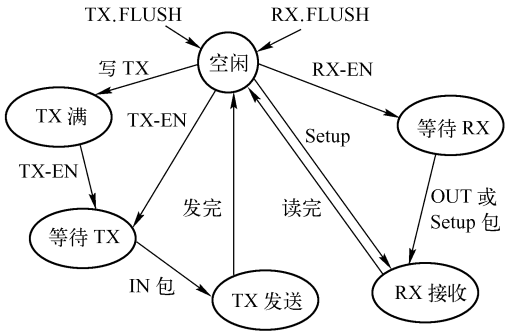



图 6-11　端点 0 的基本操作

其他 6 个端点(EPC1～EPC6)的工作与端点 EPC0 相似，但或收或发仅有单向功能，且端点控制寄存器没有设置默认地址的 DEF 位，同时增加了允许端点作用的 EP_EN 位和执行同步传输方式的 ISO 位。USB 设备的每个端点都是一个可寻址的实体，一旦收到指向该端点的 IN 包，端点通道就从指定的 FIFO 缓冲区向主机传输数据包；对已被停用的端点将自动送出 STALL 握手包；如果端点可用但无数据，则自动产生一个 NAK 握手包。同样，如果检测到一个 OUT 标识包，端点将自动接收随后来自主机的数据包，并将其存储到端点的 FIFO 缓冲区中。假如端点停用则自动发 STALL 包，而若端点可用但无可用缓冲区存储空间，便自动返回 NAK 握手包。

### 6.2.3　USB 设备接口控制的程序固件

整个固件可由上层应用主程序和下层 USB 接口控制程序组成，其中接口控制程序由初始化、USB 中断服务和数据传送 3 部分组成。接口初始化过程通过主控寄存器 MCNTRL 复位 USBN9604 接口芯片，然后由各事件屏蔽寄存器 xxMSK 和 MAMSK 来设置针对各中断事件的屏蔽、允许或禁止条件，再由 FAR、EP0 设置默认地址，由 MCNTRL 设置芯片中断输出引脚的电平特性、使能电压调节输出和接口连接功能，并通过 RXC0 和 NFSR 激活端点 0 的接收功能，进入正常操作状态。初始化完成后 USBN9604 的引脚 CLK 应能输出方波，从而可以根据计算公式核查其频率，以判断设备接口的工作状况。

主程序初始化后进入主循环，等待由 USBN9604 引发的所有事件，并可通过单片机的 INT0 进入中断服务程序。如图 6-12 所示，USB 中断服务子程序首先读取各事件寄存器，查询发生的事件类型。事件包括变迁、发送、接收、NAK、报警等类，其中事件 RX0 的处理程序是整个固件中“最忙”的，当 USBN9604 接收到总线发来的 Setup 包或 OUT 包时都会引发这个事件。处理时首先读取端点 0 的接收状态寄存器 RXS0，判断是否为 Setup 包。确认后继续读取后面的数据包，并在判断控制请求类型(标准请求或类请求)后分别处理。反之，对 OUT 包引发的 RX0 事件，则应将其后的数据包接收到指定的缓冲区，并设置任务请求标志后，再为控制传输状态阶段的 IN 事务做出响应。端点 0 发送数据包后会引发 TX0 事件，处理时应通过端点 0 的发送状态寄存器判定 TX0 是否发送成功，失败时应将端点 0 转入接收状态。

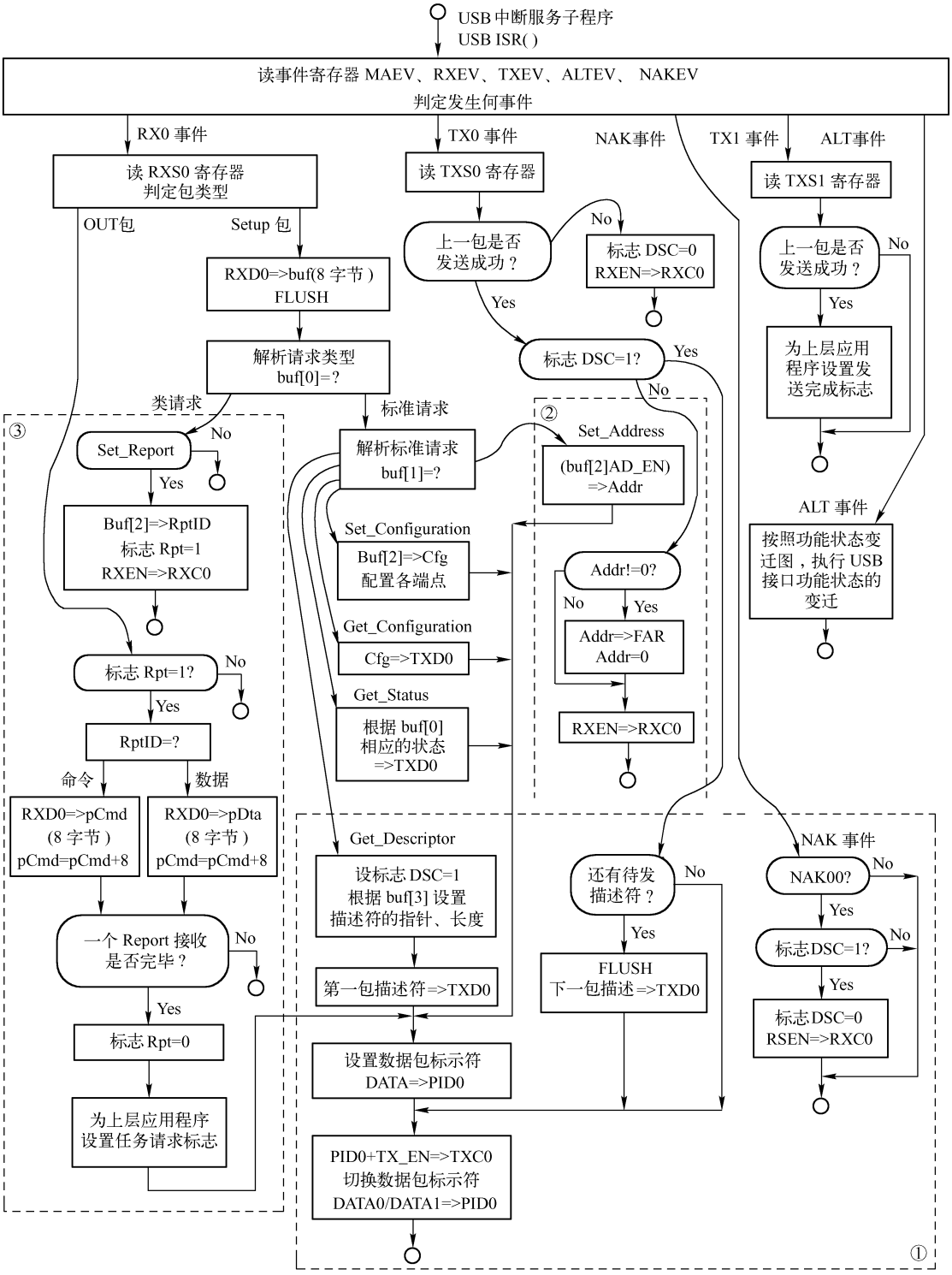



图 6-12 USB 接口控制流程

注意，图 6-12 中用虚线框标出了 USB 通信中需要特别注意的 3 项重要工作。框①描述了读取 USB 设备描述符的过程，即首先在控制传输的 Setup 阶段接收标准请求 Get_Descriptor 后设置描述符发送标志、描述符的指针和长度，并将第一包描述符内容送到端点 0 的 FIFO 缓冲区；同时设置 Data1 数据包标志后，端点 0 就等待 IN 包到来时将数据发到总线。每包描述符数据由 IN 包引导成功发送后都会产生 TX0 事件，固件应在 TX0 事件的处理程序中判断是否还需继续发送下一包描述符，直到发出所有的描述符内容。若某个 IN 包到来时发现端点 0 的 FIFO 发送缓冲区已无数据内容，则会引发 NAK 事件。于是，由 NAK 事件的处理程序清除描述符发送标志，并使端点 0 转入接收状态，结束读取描述符的过程。注意，所有描述符的发送都是在控制传输的读数据阶段完成的，第一个数据包为 Data1 类型，其后的数据包依次在包类型 Data0/Data1 之间切换。

图 6-12 中的框②列出了 USB 设备设置地址的过程。首先通过标准请求命令 Set_Address 接收并保存分配的功能地址，然后通过发送零长度的 Data1 数据包作为完成控制传输的状态阶段。发送数据包将引发 TX0 事件，故在 TX0 事件处理过程中确认数据包发送成功后便通过 FAR 寄存器设置所分配的地址。注意，设置设备地址所用控制传输没有数据阶段，此处并没有采用设置 DEF 位，也能确保控制传输成功完成后才实际建立分配的地址。

图中框③展现了 USB 接口模块接收数据的过程。首先接口模块接收到 Set_Report 类请求，于是设置传送标志 Rpt、Report 的类型 RptID，并将端点 0 设置到接收状态。然后在收到 OUT 包后根据 RptID 从端点 0 的 FIFO 缓冲区中将收到的数据读到指定的数据变量中。一个 Report 可包括多个数据包，需在接收完所有数据包后为固件主程序建立一个标志，同时发回一个零长度的 Data1 数据包，作为最终状态阶段 IN 事务作用的体现。

### 6.2.4　通用接口模块的使用方法

图 6-13 是这个 USB 通用接口模块的实物照片，图中模块占据小盒右端，空出的左端则可以用来安放设备的硬件应用电路，并通过所示模件左面的双列插座连接起来。

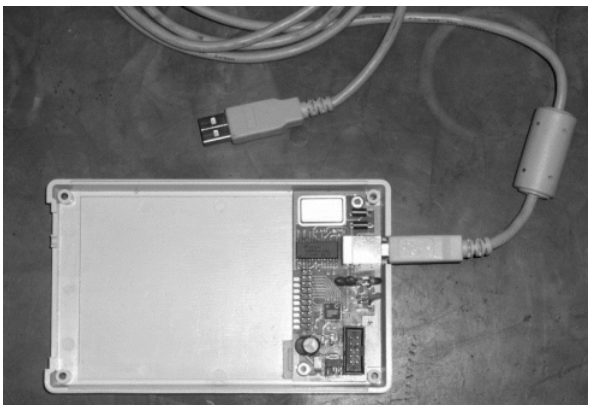


图 6-13　USB 通用接口模块的实物照片

USB 通用接口模块提供两种使用形式：一是作为带 USB 接口的微控制器，可将应用程序作为这个微控制器的主程序，并通过调用所开发的 USB 接口函数与主机建立通信联系；二是本身作为一个独立的、以 USB 方式与主机连接的智能接口，并支持另一个微处理器系统通过

普通串/并行接口访问自己。

为了便于理解和使用，全部程序固件采用严格的分层模块化结构，这里仅以一段应用程序（如程序清单 6-1 所示）为例说明前者使用方式的工作过程。

用户所需了解的只有 5 个接口函数，它们的原型声明均被包含在一个称为 usb. h 的头文件中。在应用系统的主程序调用函数 USB_Init 实现 USB 接口初始化，并利用参数设置命令缓冲区与数据缓冲区后，只需要通过循环调用（也可中断调用）USB 服务子程序 USB_ISR，即可实现 USB 接口全部例行操作管理的工作。同时可通过调用函数 USB_GetFlg 发现是否存在主机发来的新任务，并在确认后通过解析命令缓冲区和数据缓冲区的内容执行相应的应用任务。也可根据需要，通过调用函数 USB_Resp 来回送主机状态信息或数据信息（其参数为待传信息字符串的指针和长度）。任务完成后应通过调用 USB_SetFlg 清除 USB 任务标志，以便进行新一轮操作。

**程序清单 6-1　USB 接口模块的应用程序**

```
#include < usb.h >
unsigned char sCmd[16];                    //定义命令接收缓冲区
unsigned char Data[64];                    //定义数据接收缓冲区
main( )
{
    unsigned char idCmd;                   //定义任务标志变量
    USB_Init(sCmd, Data);                  //USB 接口模块初始化
    While(1)
    {
        USB_ISR();                         //调用 USB 中断服务子程序
        If( USB_GetFlg(0x01) )             //判定有无任务
        {
          //解析命令缓冲区 sCmd 内容，得标志 idCmd
          switch( idCmd )
          {
              case 1:                      //执行任务 1
                    ⋮
                  USB_Resp("OK", 2);       //返回 OK
                  break;
              case 2:
                      ⋮
               ⋮
          }
          USB_SetFlg(0x01, 0);             //清任务标志
      }
    }
}
```

所述 USB 通用接口模块为 USB 设备与主机之间的通信提供了一种理想的基础功能，以此模块为基础已经成功地开发出诸如信号源、数据采集器、逻辑分析仪、多路通道扫描器、接口转换器等多种虚拟仪器。

## 6.3　USB 主机操作软件的开发

USB 主机的软件编程与通常的应用程序设计很不相同，因为 PC 中的 USB 接口是由 PC

主板配置的，而且不同的主板厂家有不同的设计，无法通过直接控制 I/O 端口来操作 USB 总线。因此，需要按照操作系统下应用程序控制外设的一般方式，即通过设备驱动程序来实现应用程序对 USB 设备的操作。

### 6.3.1　USB 设备驱动程序

图 6-14 显示了 Windows 操作系统在 USB 通信中各功能模块的层次关系。

尽管 Windows 已经提供了用于底层总线操作的驱动程序，但与此类底层驱动程序接口的是一种称为 I/O 请求包（IRP）的数据结构，而 Windows 为应用程序提供的接口方式却是 API 函数，因此，必须在其间建立一个驱动程序来沟通 USB 底层总线的 I/O 操作与 Windows 应用程序之间的联系。VB、VC/C++和 Delphi 等通用编程语言编写的应用程序可以调用 API 函数 ReadFile、WriteFile 或 DeviceIOControl，但在 Windows 操作系统下这些函数只有在设备驱动程序的支持下才有意义。下面是解决这个问题的几种方式：

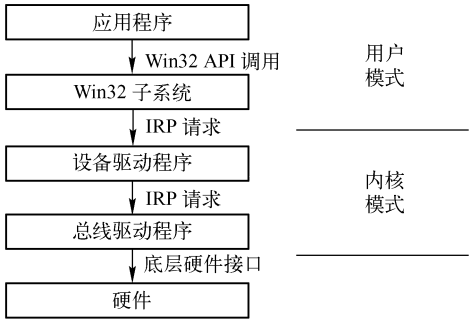



图 6-14　Windows 使用分层驱动器模型

（1）通常在 VB 等环境下编程，使用其本身提供的一些支持外设操作的控件，往往可以得到事半功倍的效果。但微软公司和 VB 本身目前并没有提供操作 USB 设备的控件。

（2）采用包含在 Windows DDK 中，用于 USB 批量传输方式的一个可与各种设备进行通信的通用设备驱动器 bulkusb. sys。

（3）对那些可以归于标准类的设备，如磁盘驱动器、打印机、键盘、调制解调器和鼠标等，可以使用 Windows 类设备的通用驱动程序。第一个被 Windows 支持的 USB 外围设备类是人机接口设备（HID）。

（4）可利用一些专用 USB 接口芯片的开发 Pc Windows 系统下运行的厂家为用户建立的示范软件，在这些芯片的应用资料中，往往提供驱动程序和有关的 inf 文件。

（5）假如所控制的设备具有一些特殊要求，提供商往往可以提供一个称为微驱动程序（mini-driver）的补充驱动程序，以增加通用驱动程序的功能。

（6）对某些具有特别用途的专用设备，可能需要自己建立一个满足专门要求的设备驱动程序。由于设备驱动程序是为执行 Windows API 函数而由 Windows 内部调用的代码，它的编程和调试都将涉及系统内核，因此开发和调试均有一定难度，也需要一些特定的工具。除了要求微软公司的 VC/C++编译器和编程调试环境外，还需要 Windows DDK（设备开发工具，可从微软网站下载）。也有第三方软件开发商提供的一些专用开发工具，但获取和掌握可靠工作的开发工具也不容易。

### 6.3.2　人机接口设备 HID 协议

人机接口设备 HID（Handle Interface Device）是指那些直接和人进行交互操作的设备，典型的人机接口设备包括键盘、鼠标、游戏操纵杆等。但在 USB 系统中使用 HID 协议并不一定就是为了操作这类人机接口设备。只要外设所需要的通信功能可以在 HID 协议下得到充分支持，就

可以采用这个协议来实现主机和设备之间的数据通信。HID 协议颇受人们关注的一个重要原因是计算机操作系统 Windows 直接支持这个协议。通过调用 Windows 的 API 函数，人们就可以对支持 HID 协议的 USB 设备直接进行操作，而不需要编写专门的设备驱动程序。这样对于初次进入 USB 系统开发的人们来说实在是一件好事。然而当真要采用这个协议来控制设备前，还是应当先看看 HID 协议的主要功能，以便确定它是否真的适于所要完成的任务。

使用 HID 协议的设备可以随时向主机传送信息，如用户可能随时按动键盘或移动鼠标，因此支持 HID 协议的主机将会不断地、周期性地查询设备来获取新的数据。设备在每次主机轮询时，把准备好的数据传送给主机。HID 协议通过报告(Report)的形式实现设备与主机之间的数据交换，通常是由主机通过在控制传输和中断传输中发送和请求报告来完成。报告具有特定的结构，实际格式很灵活，用户可以自己定义报文的格式，并用来处理各种类型的数据。报告最多包含 255 字节，发送和接收报告可以在多个事务中进行。其有关函数可在 Windows DDK 中 WDM 文档部分的 WDM_HID 和 USB_Client_Drivers 下找到。

HID 协议只使用两种传输方式：控制传输和中断传输。控制传输主要用于列举过程，中断传输的速度不是很快，也不能保证同步传送率。对一个全速设备，中断传输每毫秒最多只能有一个事务处理，且每个事务处理能够传送的最大数据量是 64 字节，这样中断传输的最高速度是 64KB/s。而对低速设备，每 10ms 只能有一个事务处理，且由于此时每个事务处理至多只能传输 8 字节，因此传输速度只有 800B/s。然而，即便如此，使用人机接口设备方式的简便性使它在很多场合仍被人们广泛接受。

### 6.3.3 在 VB 中调用 Windows 的 API 函数

Windows 的 API 函数是其核心功能与应用程序接口的主要方式，通过动态连接库形式的 DLL 文件提供给所有用户。事实上，VB、VC 等各种通用高级编程语言也都是通过内部调用 API 函数来具体实现其为用户提供的各种接口功能。如第 5 章介绍了 VB 支持串口通信的 Comm 控件，本质上也是将相关的 API 函数包装起来，从而使用户可以在 VB 环境下直接使用。但由于 VB 没有操作 USB 的控件，因此与 HID 进行通信也不像打开一个普通串口、设置几个参数后读/写数据那样简单。

在 VB 中调用 API 函数前需要预先声明，声明具有特定的格式，VB 文档中有一个名为 win32api.txt 的文件，其中包括大量的 API 函数声明，因此可以根据实际需要进行摘录。实际上，VB 开发环境还提供了一个专门检索和复制 API 函数声明的选项，实行起来更为简单。API 函数的调用方式是按 C 语言格式给出的，有时与 VB 的数据类型不能完全匹配(如指针和字串)，需要进行一些特别的处理。另外，有些 API 函数的参数需要比较复杂的结构，因此也需要用 VB 的方式构建相应的结构类型。这里，程序清单 6-2 给出了包含使用 HID 所需调用的 API 函数声明及其相关结构定义的 .BAS 文件，在此文件支持下就可以直接调用有关的 API 函数了。

**程序清单 6-2 USB 人机接口系统调用模块**

```
' HidApiCalls.bas
' 用户定义的类型
Public Type Tp_Guid                    '设备类的唯一码
    Data1 As Long: Data2 As Integer: Data3 As Integer: Data4(7) As Byte
End Type
Public Type Tp_DeviceInfoData          '设备接口数据结构
```

```
    cbsize As Long: ClassGuid As Tp_Guid: DevInst As Long: Reserved As Long
End Type
Public Type Tp_DeviceInterfaceData '空的结构指针
    cbsize As Long: ClassGuid As Tp_Guid: Flags As Long: Reserved As Long
End Type
Public Type Tp_DeviceInterfaceDetail  '设备接口细节(含路径数据结构)
    cbsize As Long: DevicePath As Byte
End Type
Public Type Tp_HidAttributes        '设备 HID 属性结构
    Size As Long: VendorID As Integer: ProductID As Integer: VersionNumber
As Integer
End Type
'API 函数
Public Declare Function CreateFile Lib "kernel32" Alias "CreateFileA" _
  (ByVal lpFileName As String, ByVal dwDesiredAccess As Long, ByVal dwShare-
Mode As Long, _
    ByRef lpSecurityAttributes As Long, ByVal dwCreationDisposition As Long, _
       ByVal dwFlagsAndAttributes As Long, ByVal hTemplateFile As Long)
As Long
Public Declare Function CloseHandle Lib "kernel32" (ByVal hObject As Long)
As Long
Public Declare Function ReadFile Lib "kernel32" _
  (ByVal hFile As Long, ByRef lpBuffer As Byte, ByVal nNumberOfBytesToRead As
Long, _
    ByRef lpNumberOfBytesRead As Long, ByVal lpOverlapped As Long) As Long
Public Declare Function WriteFile Lib "kernel32" _
  (ByVal hFile As Long, ByRef lpBuffer As Byte, ByVal nNumberOfBytesToWrite
As Long, _
     ByRef lpNumberOfBytesWritten As Long, ByVal lpOverlapped As Long)
As Long
Public Declare Function HidD_GetHidGuid Lib "HID.dll" (ByRef HidGuid As Tp_
Guid) As Long
Public Declare Function HidD_GetAttributes Lib "HID.dll" _
    (ByVal HidDeviceObject As Long, ByRef Attributes As Tp_HidAttributes) As Long
Public Declare Function SetupDiEnumDeviceInterfaces Lib "setupapi.dll" _
  (ByVal DeviceInfoSet As Long, ByVal DeviceInfoData As Long, ByRef Interface-
ClassGuid As Tp_Guid, _
    ByVal MemberIndex As Long, ByRef DeviceInterfaceData As Tp_DeviceInter-
faceData) As Long
Public Declare Function SetupDiGetClassDevs Lib "setupapi.dll" Alias "Setup-
DiGetClassDevsA" _
  (ByRef ClassGuid As Tp_Guid, ByVal Enumerator As String, ByVal HwndParent
As Long, ByVal Flags As Long) As Long
Public Declare Function SetupDiGetDeviceInterfaceDetail Lib "
setupapi.dll" Alias
  "SetupDiGetDeviceInterfaceDetailA" _
  (ByVal DeviceInfoSet As Long, ByRef DeviceInterfaceData As Tp_DeviceInter-
faceData,
  ByVal DeviceInterfaceDetailData As Long,ByVal DeviceInterfaceDetailData-
Size As Long,_
    ByRef RequiredSize As Long,ByVal DeviceInfoData As Long)As Long
```

### 6.3.4 在 PC 主机中建立支持 USB 设备的通用方式

主机 PC 对 USB 的操作首先是在一个 USB 设备插入系统时对设备进行“列举”。这一工作完全由 Windows 操作系统自动完成，并不需要编写任何应用程序。由于 Windows 提供的驱动程序 hid. sys 已经支持与 HID 设备通信所需要的功能，因此也不需要安装其他的驱动程序。列举成功后，通过调用 Windows API 函数可以直接建立应用程序与设备之间的通信通道，并发送和接收数据。在应用程序和人机接口设备交换数据之前，必须首先确认所需要的人机接口设备已经连接到系统(根据该设备的提供商和产品号)，然后建立操作该设备的操作句柄，以后就可根据该句柄，通过 HID 报文的方式和设备进行通信了。

为了便于用户使用，建立一个面向 USB 设备操作的虚拟仪器控制库函数是很有必要的。这里介绍一个类名为 Cs_MsUsb 的 USB 类函数功能，并置于一个名为 LAB49_Usb05. dll 的动态连接库中。类功能仅由下列几条函数组成，即使不懂 USB 具体技术的用户，只要掌握下面几条函数，即可自行编程利用 USB 完成 PC 主机与 USB 外设之间的通信任务。

```
打开 USB 设备：OpenUsbDevice(vid As String, pid As String) As Boolean
下发数据：WrReport(Data() As Byte, Byval n as Variant, ByVal rid As Variant)
As Integer
上传数据：RdReport(Data() As Byte, Byval n as Variant, Byval rid as Variant)
As Integer
关闭 USB 设备：CloseUsbDevice()
```

这里函数 OpenUsbDevice 用于打开一个 USB 设备通道，参数 vid 用来表示所选设备的开发者代码，而参数 pid 用来表示所选设备的产品号，返回值则表明是否成功(TRUE/FALSE)；函数 WrReport 和函数 RdReport 分别用来下发数据和上传数据，其中参数 Data 为传输数据的数组，参数 n 为数据的字节数，参数 rid 用来表示传输报告的标识符；函数 CloseUsbDevice 用于关闭连接的 USB 设备通道。

下面用一个实际工作的简例来说明这种工作方式：

```
Dim u as new Lab49_USB05.Cs_MsUsb
If u.OpenUsbDevice("0400", "C35B") < 0 Then Exit Sub
If u.WrReport( d1, 64, 1 ) < 0 Then Exit Sub
If u.RdReport( d2, 64, 1 ) < 0 Then Exit Sub
⋮
u.CloseUsbDevice()
```

这里，变量 u 被首先定义为一个 USB 控制类变量，只要先通过 VB 环境下项目菜单的参考选项设置参考项，即通过浏览硬盘文件目录路径,并选择动态连接库文件 Lab49_USB05. dll 后，即可在应用程序中使用该动态连接库下的 USB 控制类 Cs_MsUsb 来定义类变量，并进而使用该类变量的函数来操作 USB 设备。对函数 OpenUsbDevice 的调用打开了一个开发者代码 VID 为 0400、产品代码为 C35B 的 USB 设备通道。当返回值非负时，表明与该设备的通道已被成功打开，于是就可通过调用函数 WrReport 和函数 RdReport 下发和上传数据，即将数组变量 d1 中的 64 字节数据通过报告 1 下发到 USB 设备，再回收 64 字节的数据到数组变量 d2 中，最后通过函数 CloseUsbDevice 关闭连接的 USB 设备通道。

### 6.3.5　USB 操作类函数的实现方法

USB 通用类函数内部最终通过调用 Windows 的 API 函数 ReadFile、WriteFile 或 DeviceIOControl 实现其对 USB 设备的操作，因此必须打开相应的驱动程序通道。也就是说，必须获取 USB 总线操作的句柄，这个句柄可以通过调用 Windows 的 API 函数 CreateFile 得到，但函数 CreateFile 需要一个设备路径方式的字符串参数来导向，因此通用函数的关键就是如何取得这个设备路径字符串。程序清单 6-3 节选了这个函数的实现代码。

**程序清单 6-3　HID 实现的 USB 通用类函数代码**

```
'Cs_MsUsb USB 类函数模块
Dim hDev As Long                                    '设备句柄(Hid)
'开通 USB 设备通道：搜索系统设备列表，得到希望的设备句柄
Public Function OpenUsbDevice ( sVid As String, sPid As String, sVes As String) As Integer
    Dim Guid As Tp_Guid, DevData As Tp_DeviceInterfaceData, DevAttr As Tp_HidAttributes
     Dim hPnp&, iDev&, nDt&, nRt&, s $, Buf ( ) As Byte, st As Boolean, p&, vid% , pid%
    If HidD_GetHidGuid(Guid) = False Then OpenUsbDevice = -1: Exit Function '取 HID 标识符
    hPnp& = SetupDiGetClassDevs(Guid, vbNullString, 0, &H12) '取所有 HID 设备的数据结构
    If (hPnp& = 0) Then OpenUsbDevice = -2: Exit Function '-1
    iDev& = 0: DevData.cbsize = LenB(DevData) '查取 HID 设备表中的所有设备名
    Do While SetupDiEnumDeviceInterfaces(hPnp&, 0, Guid, iDev&, DevData) <> 0 '取一设备
        st = SetupDiGetDeviceInterfaceDetail(hPnp&, DevData, 0, 0, nRt&, 0)
        nDt& = nRt&: ReDim Buf(nDt&): Buf(0) = 5: Buf(4) = 0: p& = VarPtr(Buf(0))
        st = SetupDiGetDeviceInterfaceDetail(hPnp&, DevData, p&, nDt&, nRt&, 0) '取指针
        s$ = CStr(Buf) '转换数组数据到字串
        s$ = StrConv(s$, vbUnicode)          '转换为 Unicode.
        s$ = Right$(s$, Len(s$) - 4)         '滤去前 4 字节
        hDev = CreateFile(s$, &HC0000000, 3, DevSecu, 3, 0, 0) '找出查找的设备
        If hDev <> -1 Then
            DevAttr.Size = LenB(DevAttr)
            If HidD_GetAttributes(hDev, DevAttr) <> -1 Then
                vid% = Val("&h" & sVid): pid% = Val("&h" & sPid)
                    If ( DevAttr.VendorID = vid% ) And ( DevAttr.ProductID = pid% ) Then
                    OpenUsbDevice = 0: Exit Function
                End If
            End If
            st = CloseHandle(hDev)
        End If
        iDev& = iDev& + 1
    Loop
    OpenUsbDevice = -3
End Function
```

```
Public Function CloseUsbDevice() As Integer    '关断 USB 设备通道
    CloseUsbDevice=CloseHandle(hDev)
End Function
```

如程序清单所示，为了通过调用 Windows API 函数 SetupDiGetClassDevs 得到包含全部列举到的 HID 设备的信息数组结构，需要一个针对 HID 类的 GUID 作为参数，它是一个 128 位且能唯一表示对象的代码。HID 类的 GUID 被定义在 hidclass.h 文件中，因此可直接读取(4D1E55B2-F16F-11CF-88CB001111000030)。这里采取调用另一个 Windows API 函数 HidD_GetHidGuid 的方法，通过参数返回 Guid 的结构变量。函数 SetupDiGetClassDevs 的参数 &H12 用来告诉函数只去查找当前已经连接和已经列举且属于 HID 类成员的接口设备，其返回值为系统存储所有 HID 设备信息结构的数组地址句柄(放在一个长整型变量 hPnp& 中)，可以直接作为下一步调用其他 Windows API 函数的参数。

为了从上述包含所有 HID 设备信息的结构数组中找出需要访问的接口设备，需要建立一个程序循环体来依次取得各接口设备的信息。而在每个循环周期内，需要首先调用 Windows API 函数 SetupDiEnumDeviceInterfaces，取得反映一个设备信息的结构变量 DeviceData，然后再以其作为主要参数，通过调用另一个 Windows API 函数 SetupDiGetDeviceInterfaceDetail，取得包含设备路径名的结构参数。这个参数的结构包括两项，前者为一个长整型变量，后者则为表示设备路径名的字符串变量。考虑 VB 和 C 对字符串的不同存储方式，这里采用一个暂存的字节数组变量来存储这个结构体，然后利用 VarPtr 操作符来获取这个变量的起始地址 p&。这样，将 p& 作为函数的第 3 个参数，返回后就可以根据长整型变量在 VB 中占 4 个字节，从这个存储变量中挑出所需要的设备路径。

注意，程序清单中函数 SetupDiGetDeviceInterfaceDetail 调用了两次，这是因为其第 3 个参数作为包含设备路径名的结构变量，必须填写结构变量的长度作为函数的第 4 个参数，方能返回需要的结果。但首次调用时不知此长度，故可先预调一次这个函数。尽管不能成功得到结果，但却能从其第 5 个参数中得到应取长度，于是可在第 2 次调用这个函数时将其作为第 4 个参数，从而可使函数调用得到满意的结果。

有了设备路径，就可将其作为调用 Windows API 函数 CreateFile 的参数，得到一个操作句柄 hDev。由此句柄，通过调用函数 HidD_GetAttributes，取回设备的厂家和产品识别码(VID 和 PID)，当其与指定的要求匹配时，就用此句柄来进行以后的操作。最终控制 USB 总线的所有功能都是通过使用 ReadFile 和 WriteFile 两条 API 函数来读、写数据报告的。这两条函数的参数相似，包括从函数 CreateFile 返回的句柄、通信用的数组变量首地址、数据报告的字节数、成功读出或写入设备数据的字节数等。注意参数数组变量中的第一个字节为数据报告的标识，然后才是数据本身。函数的具体实现方法可见程序清单 6-4。

**程序清单 6-4　USB 数据发送和接收的通用函数代码**

```
'写入 USB 设备读 USB 设备：发送数据(最多 254 字节)，并自动加入 ReportID(=0)
Public Function WrReport(Data() As Byte, Byval n as Variant, ByValrid As Variant) As Integer
    Dim buf() As Byte, i as integer, nReturn&
    ReDim buf(n + 1)
    If n > 254 Then ErrorExit ("Maximum ByteCount for WriteUSBdevice is 254")
    buf(0)=rid
```

```
    For i=1 To n: buf(i)=data(i-1): Next I:
    If WriteFile(pHidHandle&, buf(0), n+1, nReturn, 0) Then
        WrReport=nReturn
    Else
        WrReport=-1
    End If
End Function
Public Function RdReport(data() As Byte, Byval n as Variant, Byval rid as Va-
riant) As Integer
    Dim i As Integer, nReturn As Long
    If n>254 Then ErrorExit ("Maximum ByteCount for ReadUSBdevice is 254")
    If n<1 Then ErrorExit ("Minimum ByteCount for ReadUSBdevice is 1")
    If ReadFile(pHidHandle&, buf(0), n+1, nReturn, 0) Then
        For i=1 To nReturn-1: data(i-1)=buf(i): Next i
        RdReport=nReturn
    Else
        RdReport=-1
    End If
End Function
```

## 6.4　基于 USB 的信源/数采装置

信号源和数采器是测控系统最基本的组成部件，以 PC 内插卡形式构建的自动测控系统已被广泛使用。然而，当 PC 主机普遍配置 USB 接口后，开发基于 USB 接口的数据采集器就成为更自然的需求。

### 6.4.1　信源/数采设备的电路设计

图 6-15 为开发的信源/数采模块的原理图，其核心是一块专用的 USB 接口芯片 USBN9604 和一个高集成的低功耗单片机 Cygnal8051-F007，同时配置适当的模拟通道调理电路和外部接口。

F007 主频 25MHz，支持 32KB 的内部 Flush ROM 和 2KB RAM。片内不但提供 UART、$I^2C$、SPI 等多种总线接口，而且带有 2 路 12 位的 D/A 转换器和 4 路 12 位的 A/D 转换器(转换速度 100kHz，并配置可编程的放大器和比较器)。Cygnal 公司的 F007 芯片有两个相互独立的 64KB 程序存储空间和 64KB 数据存储空间，其中片内有一个 32KB 的 FLASH ROM、一个 256B 的内部数据存储区和一个 2058B 的 RAM 位于外部数据存储空间中，它们可通过 MOVX 指令来寻址。F007 的另一个特点是可为有限的引脚选择接口类型，端口 P0、P1、P2 的每一个引脚既可以作为普通的 I/O 端口，也可依次编程设置为 $I^2C$(SMBus)、SPI 和 UART、捕获和比较模块、16 位定时/计数阵列、通用的 16 位定时/计数器、电压比较器、外部中断资源等。这些资源可以通过优先连接编码器灵活设定，I/O 端口单元电路在端口配置寄存器中可被配置成推挽式或开漏式。芯片供电电压范围为 2.7~3.6V，典型供电电流为 10mA。

USBN9604 的控制采用 MICROWIRE 接口，鉴于 F007 上的 SPI 接口与其兼容，这里将 F007 设置成 SPI 主设备，此时最大数据传输速率为系统时钟的一半。激励信号和采集信号的

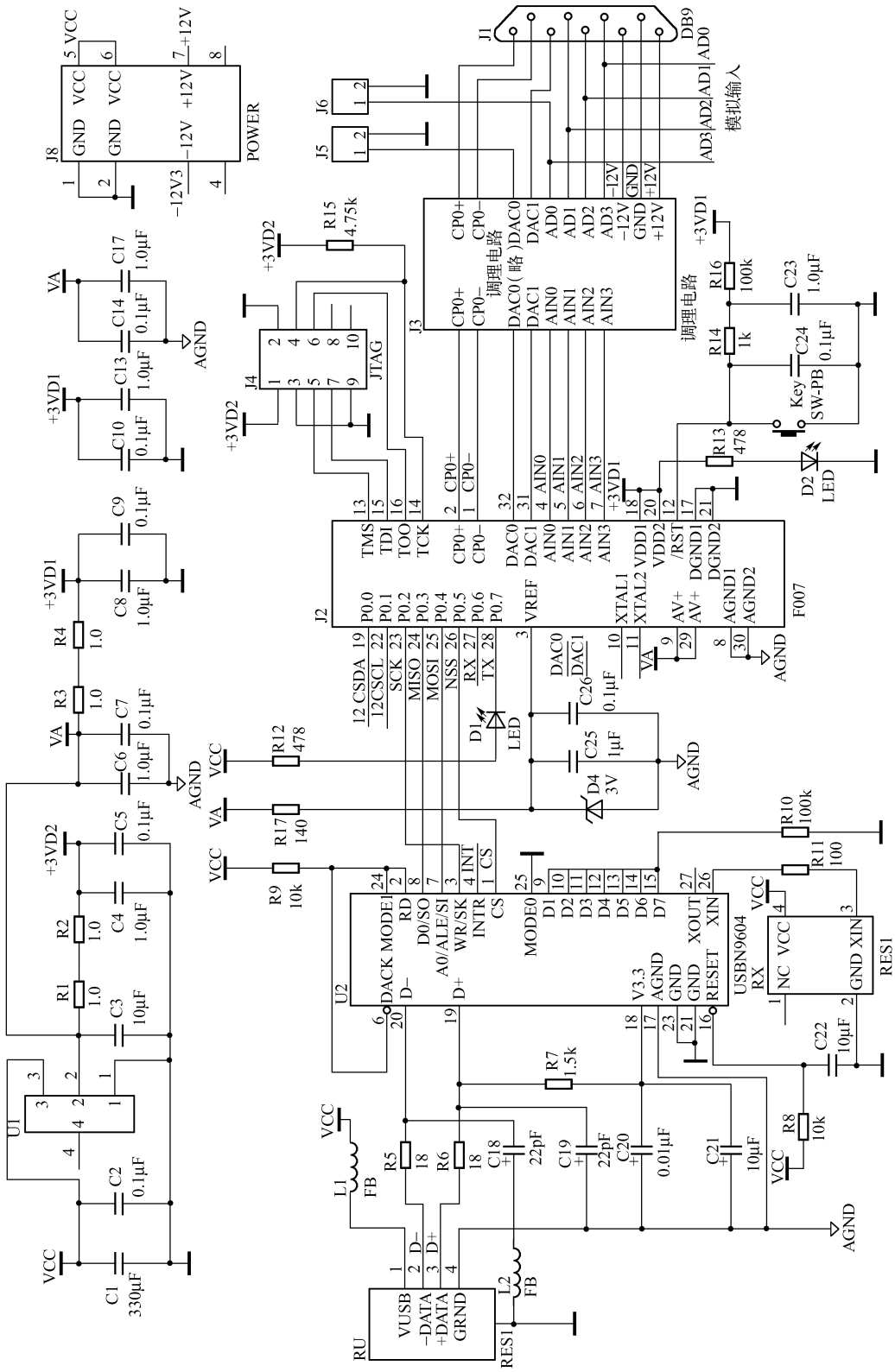



图 6－15　支持 USB 通信的信源 / 数采设备的电路原理图

幅值范围均可设为±10V，由调理电路将转换器的输入和输出电平信号变换到需要的范围。为了使用 USB 总线提供的 5V/500mA 电源，设计应尽量减少设备的功耗(这里功耗为 210mA)。主电源 5V 通过磁珠直接从总线电源获取，调理电路的模拟电源±12V 通过 DC/DC 隔离变换模块得到，F007 要求的数字电路和模拟电路电源则分别通过 REG1117-3.3 稳压芯片并经 RC 滤波后形成。

如图 6-16 所示，在模块设备的一端以十分紧凑的布局实现了支持 USB 接口通用信源/数采的全部核心功能，而将信号隔离、电平转换等通道功能单独设计并配置在模块的另一端。设备模块的中部配有外部接口 16 线插座，以便作为通用模块连接不同场合应用对象所需要的外围电路。探头接口既可通过同轴电缆插座直接连接，也可通过一个 9 芯插座接出(可进一步串接变换隔离模块)。

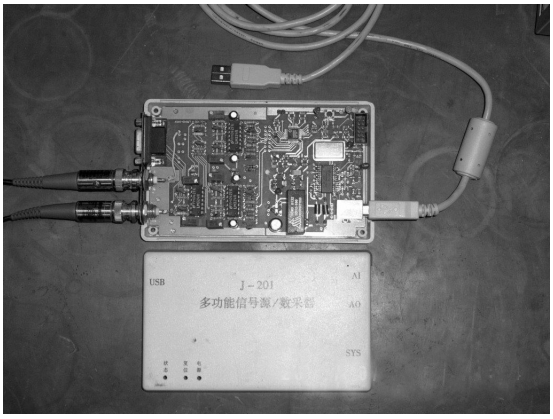



图 6-16　USB 信源/数采设备实物照片

### 6.4.2　设备控制固件程序的建立

设备控制固件编程根据分层模块化的设计原则，首先建立通常系统所需的基本底层功能函数，包括系统初始化(设置堆栈、定义 I/O 端口、输出方式、设置各状态变量初值)；SPI、$I^2C$、UART 接口的初始化和读、写操作；定时计数器 Timer 的初始化、使能设置、参数配置；A/D、D/A 的初始化和读、写操作，以及通道、增益、触发比较器设置；中断响应、数字捕获、看门狗设置、I/O 读/写等。

实际设备固件分为几个独立的程序模块，其中 F007.c 为微处理器接口功能的底层驱动模块；9604.c 将全部 USB 功能归纳为初始化、中断操作、状态读/写和上传数据等 5 个基本功能函数，其具体应用方法和主要工作流程已在本章前面详细讨论。Ut51.c 是通用函数库。这里将集中讨论系统主模块 J201.c 的设计思想和具体工作流程。

如图 6-17 所示，主流程首先在函数 USB_Init 的参数中指定命令缓冲区和数据缓冲区，然后就可通过在主程序循环过程中反复调用函数 USB_ISR 完成有关 USB 通信的全部监控工作。当通过函数 USB_GetFlg 查明存在新的设备命令时，则可从命令缓冲区读取命令本身及其参数，并予以执行。然后可以根据具体需要通过函数 USB_Resp 上传设备的响应数据，也可以直接从数据缓冲区读取下载的数据块，最后再通过函数 USB_ClrFlg 将状态标志复位。

这里，命令 Run 是来自主机的一个重要命令。设备收到该命令后，将其参数 nk 作为将

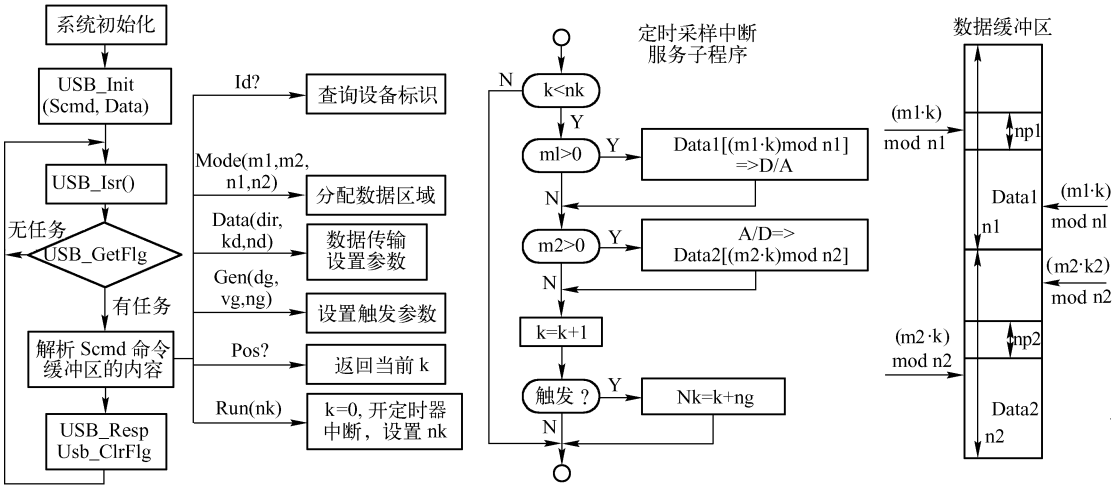



图 6-17 USB 信源/数采器的程序控制流程

要采样的总数，并将采样序列指针清零(k=0)，同时打开中断。于是根据先前默认的定时计数器设置，系统实现等间隔时间的采样中断(只此一个中断源或最高优先级)。在中断处理过程中，当采样总数未达到要求(k<nk)时，按照先前设置的通道要求依次进行各相应通道的D/A 或 A/D 转换，同时完成与数据缓冲区的内容交换，再将指针步进调整(k=k+1) 后退出中断服务,返回主循环。另一方面，主机可以通过命令 Pos? 了解采样进行的情况(返回 k 值)，进而可以通过命令将 A/D 转换后存入的数据读回主机。值得强调的是：采样中断进行的 D/A 和 A/D 过程与上位机通信传输数据应当看成独立的并行工作，设备控制器只提供基本的执行机制而不加以协调，主机通过发送各种命令来协调整个设备的工作,这样就把对不同测试设备的开发工作尽可能多地推向主机软件。

**表 6-3 信源/数采设备基本命令表**

| |
|---|
| Rst |
| Id? |
| Tst? <d> |
| Mode<m1>,<m2>,<n1>,<n2> |
| Smp<t1>,<t2> |
| Gain<i> |
| Data<dir>,<kd>,<nd> |
| Pos? |
| Gen<dg>,<vg>,<ng> |
| Run<nk> |

实际设备的基本命令如表 6-3 所示，包括通用的复位命令 Rst、设备标识名查询 Id 和自检命令 Tst。命令 Mode 用于数据区域分配，其中参数 m1 和 m2 分别表示 D/A 和 A/D 的通道数，参数 n1 和 n2 则为通道所占数据区的大小；命令 Smp 用于采样时间设置，参数 t1 和 t2 分别为定时基数(0~4096μs)和定时倍数(0~255)；命令 Gain 用于 A/D 增益设置，参数为增益选择；命令 Gen 用于触发，参数分别表示触发方向和触发电平，以及触发后的采样数；命令 Pos 要求返回当前采样的序列数；命令 Data 用于数据传输，3 个参数分别表示传输方向、数据初始地址和数据长度；命令 Run 用于启动，参数表示要求运行后的采样总数。表中命令名后的问号表示要求返回响应数据。可以看出，这些命令设计得比较贴近底层的执行功能，而将其组合起来更接近使用功能的操作序列留给上层主机完成,从而体现尽可能用上层主机的软件功能来代替下层设备硬件和固件工作这一虚拟仪器的基本设计思想。

### 6.4.3 不同工作模式的虚拟仪器应用

从虚拟仪器的应用考虑，信源/数采器有 3 种主要工作形式:一是作为双路输出的信号发

生器；二是作为双路输入的示波器；三是单入单出的系统辨识机或数字控制器。

从主机与设备通信的角度看，存在间歇式集中传输模式和持续传输模式两种。间歇模式适于电子测试，如信号发生器只需事先将一个周期的数据下载到设备内部的数据存储器中，则在设备中断采样期间就会依次通过 D/A 转换形成连续的信号输出。由于 F007 片内的数据存储器有 2KB，则两个通道 12 位数据可以预置 500 个数据。同理，作为双踪示波器的数采设备也是在预定时间的采样中断期间进行 A/D 转换，直至得到全部数据后再一同传至上位 PC 中。由于这种间歇式的工作方法在数采阶段只是中断采样取数，并无上传数据的任务，因此可以实现较高的采样速率。但也由于采到的数据需要全部暂存在缓冲区中等待数传阶段统一上传，因而数据存储深度受到缓冲区内存空间的限制。

对于持续传输模式，在机电设备和其他更广范围的应用系统中也很常见。如系统辨识机的数据很长且往往采用递推处理方式，因此通常要求边采边传。又如心电监视仪 24 小时工作，希望 PC 主机能够实时地持续取数，就像 PC 直接操作 A/D 转换器一样。完成这种任务需要注意两个问题，一方面要正确地选择存取数据的位置。仍以图 6-17 所示工作过程为例进行讨论。显然，对信源，下载数据序列 k1 应大于 D/A 采样序列 k，而上传数据序列 k2 应小于 A/D 采样序列 k。考虑到实际数据缓存采用有限区间循环存储的方式，所得关系式应该改为：$k<k1<k+n1/m1$，$k>k2>k-n2/m2$。于是，可以得到如下一组简明算法：

当 $k1 < (k + n1/m1 - np5/m1)$ 时，即可下载信源数据(np1 字节)。

当 $k2 < (k - np2/m2)$时，即可上传一组数采数据(np2 字节)。

而若下载数据过快 $k1>k+n1/m1$，或上传数据过慢使 $k2<k+n2/m2$，均会造成循环存贮的错误复盖。

因此，还应考虑确保通信一包数据的时间小于采样这一包数据所用的时间。好在这类任务的过程往往较慢，否则难以实现任务的实时完成。

这里所述的 USB 通用信源-数采设备，以及信号发生器、数字示波器和系统辨识机的应用方式具有十分普遍的意义，许多专用的测试工作都可从中得到借鉴，只需在软件方面再做一些更专业的工作。图 6-18 展示了与此相应的虚拟仪器操作界面。

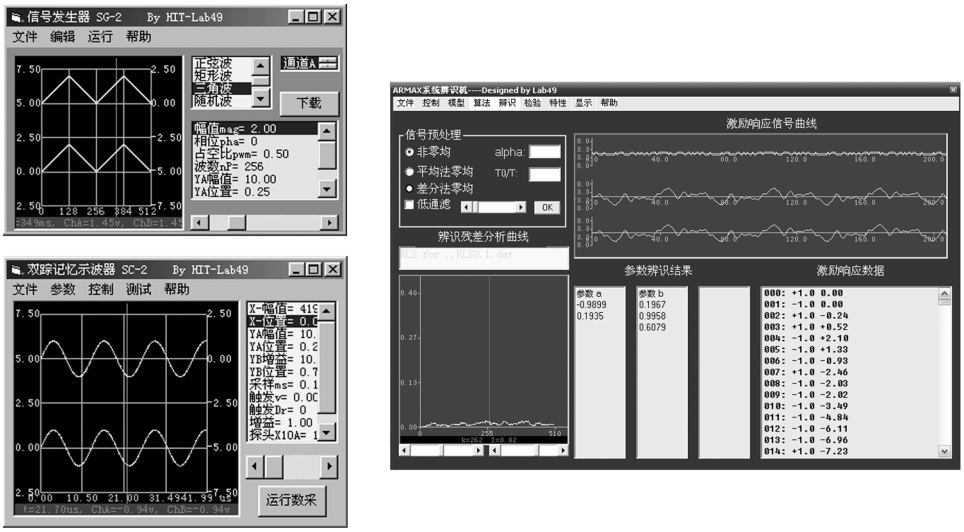



图 6-18　虚拟仪器形式的操作界面

# 6.5 支持 USB 接口的 50MHz 高速数据采集器

本节以数字示波仪器为典型应用背景，通过介绍其设计思想、核心结构、工作原理、实施要点和使用方法，试图说明仪器的开发思路。示波仪器主要用来观测动态波形，以便了解被测信号的变化情况。示波仪器往往更加看重带宽而放宽精度要求。这里介绍的数据采集器采用 8 位字长精度、50MHz 采样频率、双通道同步触发作为基本指标，并将 USB 总线作为设备与主机之间通信的手段，通过 PC 主机软件建立虚拟仪器面板，实现示波仪器功能的波形显示和控制操作。

## 6.5.1 高速数据采集器的系统结构

高速数据采集器的基本原理结构如图 6-19 所示，其中通过高速 A/D 转换器采样得到的数据被直接存入先入先出的 FIFO 缓存器中，这里 A/D 和 FIFO 是高速数据采集的核心器件，FIFO 的速度应比 A/D 转换器的速度更快一些。由于这两个器件均工作在 50MHz 以上的时钟频率，单片机无法对它们进行实时控制，因此需要采用大规模可编程逻辑器件 CPLD 实现时序逻辑来控制和管理数据采集过程。然而，在数据已被采入缓存器后进一步将其通过 USB 总线逐个传送到主机软件处理的过程则不需要那么快。于是，整个数据采集器仍由单片机管理。它一方面通过操作 USB 接口和外部 PC 主机连接，响应主机命令，上传采集的数据；另

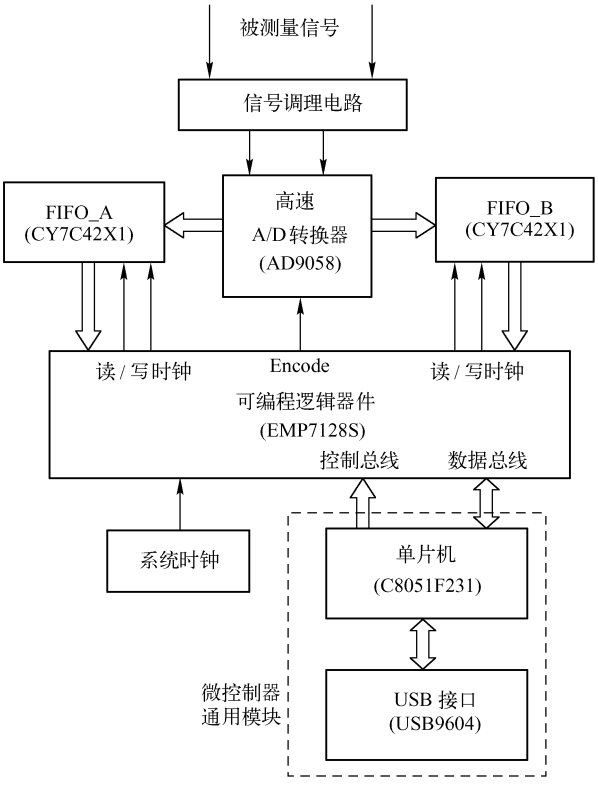



图 6-19 USB 高速数采的系统结构

一方面通过访问可编程逻辑器件中的寄存器来控制数据采集的工作过程。这里，数据采集器在功能上可以划分为采集和通信两个部分。前者为了最大限度地利用 A/D 和 FIFO 的高速度，尽量以最短的布线、最直接的硬件方式实现数采的时序控制；后者则充分发挥单片机操作灵活的特点，尽量以程序来实现 PC 主机的控制意图，并可将单片机与 USB 接口组成通用的模块。另外，充分利用 CPLD 的高速度和高集成度，以及在线编程和在线仿真的特性来完成数据采集器的时序电路和逻辑电路，不仅能够大大提高系统的整体性能，而且也能加速设备开发和调试的工作效率。

### 6.5.2　核心器件的原理和时序

为了同时高速采样两路模拟信号，核心器件采用 AD 公司生产的 AD9058。它内部包含两个完全独立的 A/D 转换器，最高采样频率达 50MHz。图 6-20 所示为高速数采的应用电路，其中 AD9058 采用自身提供的内部参考电压。以运算放大器 AD8032 为核心的调理电路将-5~+5V 电平范围的模拟输入信号变为 AD9058 所要求的 0~2V 转换电平。为了简化通道调理电路，AD9058 的 0V 模拟输入信号对应外部+5V 输入信号，2V 模拟输入信号对应外部-5V 输入信号，这种反相关系将在上位机对采集数据的处理中得到校正。注意，在 AD9058 的信号输入端和 AD8032 的输出端之间需要串接一个 5Ω 的电阻，其作用是在不降低输入信号带宽的前提下，防止 AD9058 的输入端电容(大约 10pF)影响运放 AD8032 的稳定性，并为 AD9058 内部的比较器提供适当的阻抗匹配。在地和负电源之间连接一个反相偏置的二极管，是用于防止电源上电顺序出现+$V_s$ 先于-$V_s$ 情况时损坏器件，以及+$V_s$ 过流时产生锁定效应。

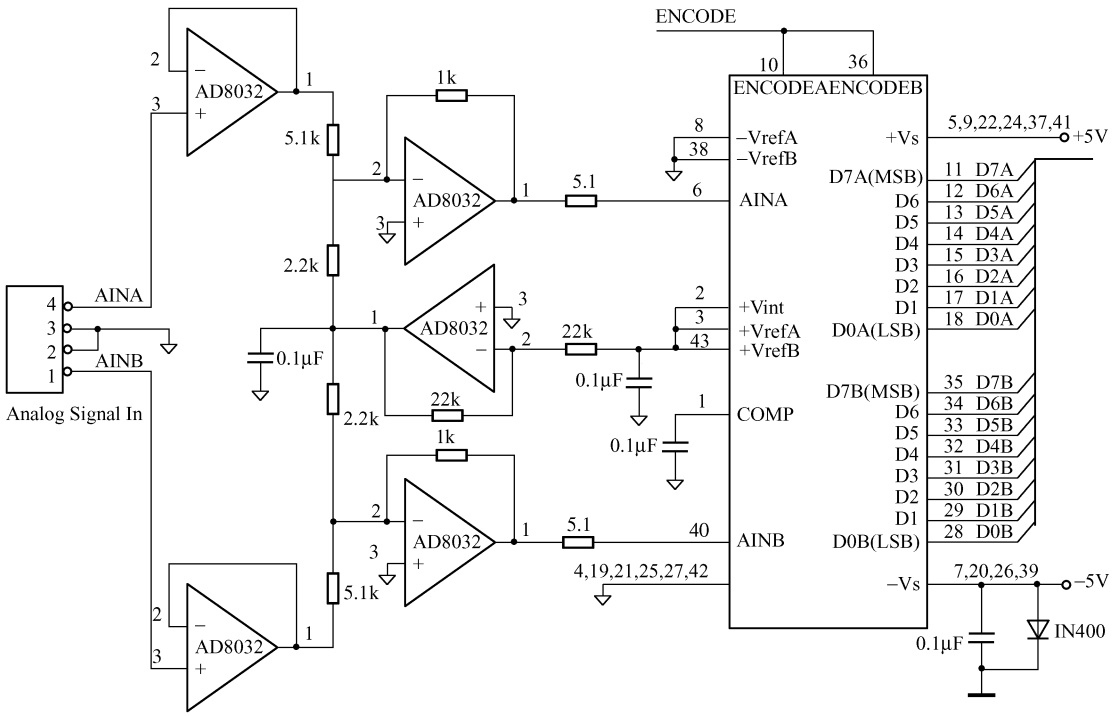



图 6-20　高速模数转换电路

AD9058 在 ENCODE 时钟下完成模数转换的时序关系如图 6-21 所示，其中孔径时间 $tB_{AB}$（0.8ns）可以忽略。AD9058 在 ENCODE 时钟下对模拟输入信号进行第 $N$ 次采样后需经数据传播延迟时间 $tB_{PDB}$（12ns）的延迟，模数转换的有效数据才会稳定地呈现在数据线上，而且一直保持到 ENCODE 时钟第（$N$+1）次采样后 $tB_{VB}$（8ns）时段。

芯片 CY7C44x1 是 CYPRESS 公司生产的高速低功耗先进先出存储器（FIFO），分为 10ns/15ns/25ns 等不同速度等级，最高工作频率可达 100MHz，字长为 9 位，且有引脚相互兼容的不同型号（存储容量分别为 64B/256B/512B/1KB/2KB/4KB/8KB）。CY7C44X1 具有完全独立的数据输入/输出端口和读/写控制电路，因此读、写过程可以同步进行，也可异步进行。这里采用 15ns 的芯片，满足数采频率 50MHz 的应用要求。AD9058 在 ENCODE 时钟下进行 A/D 转换，CY7C44X1 在 WCLK 时钟下存储 AD9058 转换的数据。为了确保 AD9058 转换的数据能够存入 CY7C44X1，WCLK 时钟由 ENCODE 时钟反相后延迟 $t_D$ = 4ns 形成，并在 CPLD 中实现。

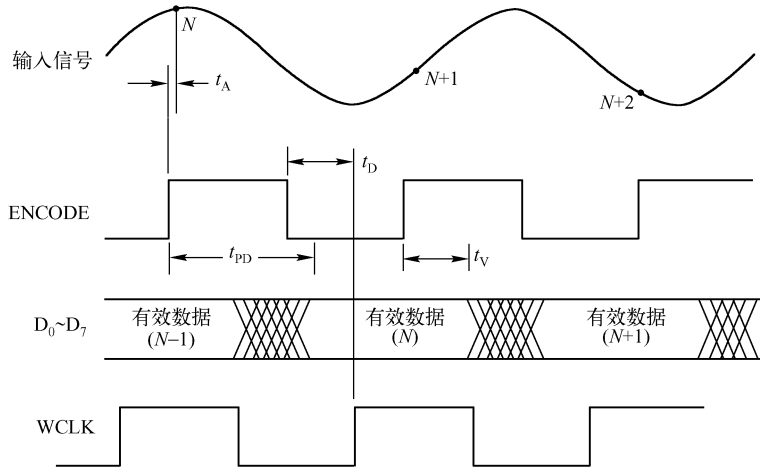



图 6-21 数据同步采集存储时序图

### 6.5.3 控制电路的 CPLD 实现

设计的 CPLD 内部电路及其外围连接如图 6-22 所示。电路主要包含了通信、时序控制、核心状态控制、追忆比例控制和触发控制等 5 个模块。

通信模块用来负责 CPLD 与单片机之间的通信，通过双向的 8 位数据线 $D_0$～$D_7$、地址线 Addr0～Addr2、读/写控制线 W/R 和锁存/时钟复用线 ALE/Rclk 与单片机相连。该模块接受单片机对其内部参数的设置、工作状态的控制，也将内部设置的参数、工作的状态和从 FIFO 提取的数据回送单片机。时序控制模块受核心状态控制模块的控制，产生采集数据、读取数据所需的 A/D 转换与 FIFO 读/写的时钟脉冲。时序信号产生逻辑电路在数据采集模式下，可以根据时钟设置寄存器的设置值改变数据采集频率，分为 50MHz、25MHz、10MHz、1MHz、100kHz 或 10kHz 的采样时钟信号；而在读取数据模式下则根据锁存时钟复用线 ALE/Rclk 产生 FIFO 的读数时钟信号。

核心状态控制模块经过通信模块接受微控制器对其内部控制寄存器进行参数设置，控制

整个系统的工作状态。其内部包括一个有限同步状态机，该状态机有 6 个代表系统不同工作阶段的合法状态和 2 个与系统工作阶段无关的非法状态，如图 6-23 所示。

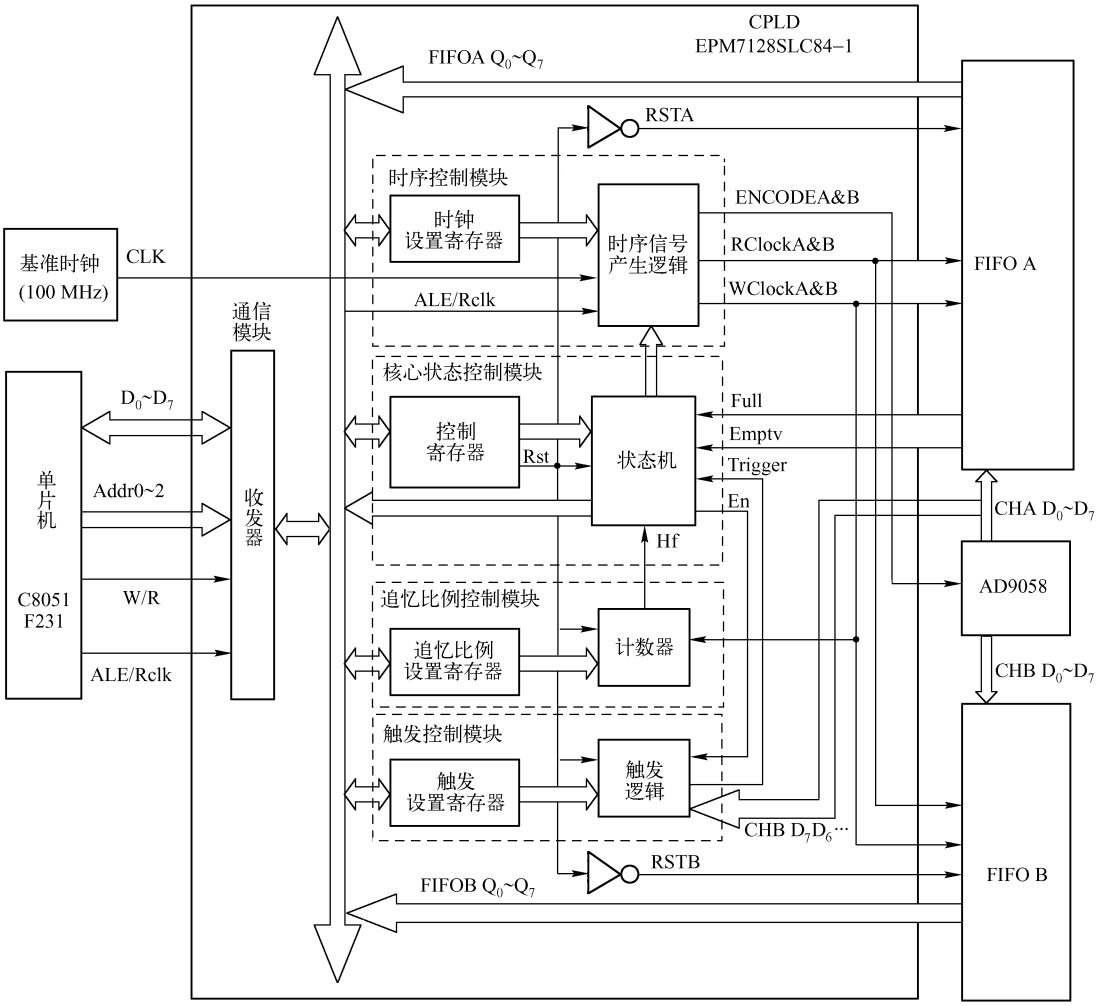



图 6-22　CPLD 内部结构及其外围连接

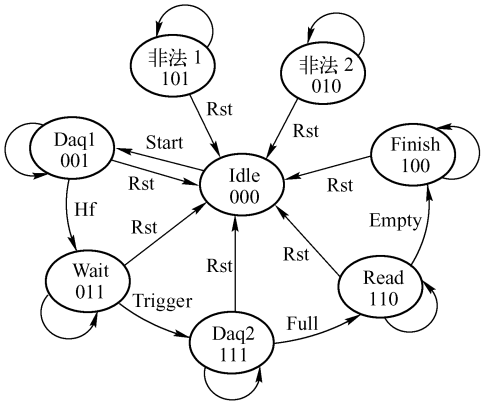



图 6-23　状态机状态变迁图

状态机在任何时候收到控制寄存器的 Rst 复位信号都将进入 Idle 空闲状态，此时追忆比例控制模块、触发控制模块和 FIFO 均复位到数据采集的初始等待状态。一旦状态机收到控制寄存器的 Start 信号则进入 Daq1 数据采集状态，系统开始进行数据采集，并把采集的数据存储到 FIFO 中，此时 FIFO 可谓“只进不出”。当采集一定数据后，状态机收到追忆比例控制模块的 Hf 信号(半满)，就会进入 Wait 等待触发状态，发出 En 信号来使能触发控制模块，等待触发信号的到来。同时一面继续采集数据送入 FIFO，一面从 FIFO 的另一端读出数据，形成“边进边出”的平衡情况。一旦收到由触发控制模块传来的 Trigger 触发信号，状态机将进入 Daq2 数据采集状态，此时“只进不出”，即不再从 FIFO 中读出数据，只是继续读入数据，直至 FIFO 存满后其满信号 Full 使状态机进入 Read 读状态。在此读数据状态，单片机通过控制信号 ALE/Rclk 从 FIFO 中读取数据，并进而送给上位机。当 FIFO 中的数据被读空时，其 FIFO 空信号 Empty 使状态机进入 Finish 数据采集结束状态，完成一轮数据采集的工作过程。

追忆比例控制模块用于设置触发信号前、后系统所存采集数据的比例，其核心是一个计数器。它根据追忆比例设置寄存器的取值来设定计数器的溢出值，在数据采集开始后计数器对采样次数进行计数，当计数达到溢出值时输出 Hf 信号，切换状态机状态。由于 FIFO 的存储总量一定，计数器的不同溢出值就对应不同的触发追忆比例。如计数溢出值取 FIFO 容量的一半时，最终 FIFO 保存的数据将依触发时刻为界，形成前后各取一半的方式。

触发控制模块根据触发设置寄存器的内容完成触发功能。在数据采集过程中一旦其内部的触发逻辑被选通，就对 A 通道 A/D 转换的数据高位进行实时监测，直到满足触发条件，输出触发信号 Trigger，切换状态机的状态。采集器是对-5~5V 范围的模拟信号进行 A/D 转换，因此 A/D 转换得到的高位编码就对应一定范围电平内的模拟信号。

### 6.5.4 USB 接口和虚拟仪器模式

虚拟仪器的最终控制者是 PC，然而由于 PC 不能直接控制设备的模数转换器，所以需要通过 USB 把控制命令发送给设备，再由设备中的单片机执行控制命令，通过设置 CPLD 内部的寄存器来改变系统的采样频率、追忆比例和触发方式，从而通过 CPLD 内部各控制寄存器间接地控制数据采集过程。具有 USB 接口的通用微控制器核心模块可以直接采用本章前面介绍的通用 USB 设备接口控制器。

设备的实物照片如图 6-24 所示。

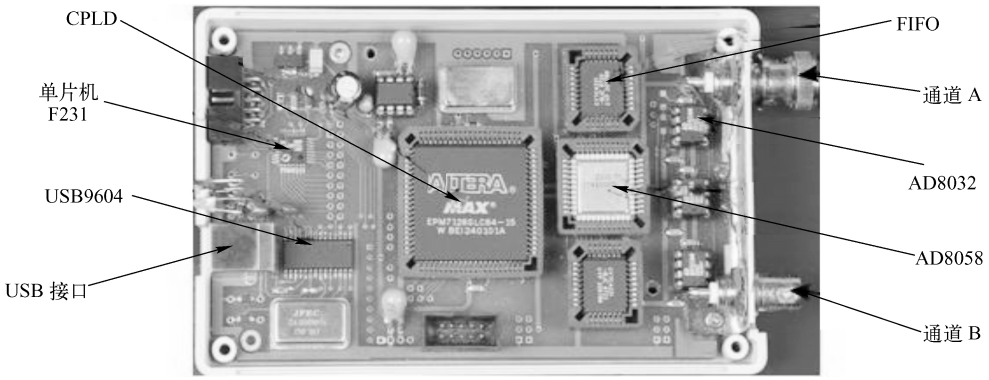



图 6-24 50MHz 数采设备实物照片

本节介绍的 USB 高速数据采集器提供了一种虚拟仪器典型模式，即以单片机和 USB 接口构成一个通用设备控制器，再以 CPLD 的时序逻辑形成执行各种命令的控制信号来操作专用的测控电路。对不同测试要求往往只需在硬件电路方面改变专用的测控部分。如将本示例中的 A/D 转换器改为数字输入缓冲器，即可形成一个逻辑分析仪的硬件结构，有关内容可参见第 7 章。

在这种虚拟仪器模式中，由于采用 CPLD 实现的硬件电路操作 A/D 转换器，实际数据采集的速度可以很快。但采用并行闪烁式高速 A/D 转换器的功耗随采样速度提高而增大，因此受 USB 供电 5V/500mA 的限制，需要特别注意选用低功耗的电子器件。

主机上层软件采用 VB 语言开发，设备驱动程序可以利用 Windows 支持的 USB 人机接口设备(HID)的驱动程序。对 HID 来说，一个全速设备中断方式下每个事务能够传送的最大数据量是 64B，这样对每毫秒 1 帧的中断传输方式来说，数据采集器向上传送数据可以达到 64KB/s 的速度。对于高速采集的数据已经存储在 FIFO 器件中的本示例来说，满足主机显示的要求是没有问题的。至于数字示波器面板的开发，则可直接参照图 6-18 所示的方式。

## 6.6　高速数据传输和同步问题的探讨

尽管 USB2.0 规范的最高传输速度已经达到 480Mbps，但很多 USB2.0 设备在实际工作时的数据传输速度却与此相差甚远，许多因素都有可能成为影响速度的瓶颈，因此很有必要对此进行探讨。同时，同步触发是自动测试系统区别于其他系统的重要特征，探讨主机同步触发若干仪器的问题是将 USB 发展成为测试总线的必然要求。

### 6.6.1　影响 USB2.0 数据传输速度的因素分析

在 USB 数据通信过程中，总线上传输的内容并不只是真正有用的测试数据信息，还要包括诸如同步信号、类型标识、校验码、握手信号等各种实现通信协议的信息，因此实际数据传输的速率根本没有可能达到总线传输的 480Mbps 极限速度。对不同的传输类型，存在不同的协议开销。如在 USB1.1 协议下规定的每毫秒传输 1 帧的结构中，对一个设备的中断传输只能进行一次，考虑中断传输的数据包为 64B，故可算出这种传输的最大速度只有64KB/s。对 USB2.0 的情况，由于采用了微帧结构，每帧分为 8 个微帧，且中断传输在每个微帧下可以传输 3 个数据包，而每包的数据也增加到 1024 字节，故可以算出 USB2.0 的中断传输的最大速度提高到 8×3×1024/ms=24MB/s。尽管这一指标与 USB1.1 的 64KB/s 相比已有明显提高，但却仍与 480Mbps(60MB/s)相差甚远。

USB2.0 中最能体现高速传输特点的应属批量传输类型，其 53.24MB/s 的理论传输速度上限可以说比较接近 60MB/s 的总线传输速度极限。因此如果仅从获取最高数据传输的目标出发，应当选用批量传输工作方式。USB 协议规定，控制传输应确保在低/全速时能够使用 10%的带宽，高速时能够使用 20%的带宽。而批量传输并没有保留任何带宽，即批量传输只有在控制传输和其他传输不需要使用其带宽的情况下，方能使用剩下的带宽。因此，尽管总线闲置时批量传输可以在一段时间里尽快传输大量的数据，但总线忙时批量传输可能工作得很慢。通常 PC 主机可能同时使用诸如鼠标、键盘、摄像头、打印机和扫描仪等多种 USB 设

备，它们分别采用不同的传输方式。可以设想，如果只有一个设备以批量传输方式独占系统的全部带宽，显然速度会很快(接近 53MB/s)。但如果新插入的若干设备需要使用控制传输全部 20%的保留带宽，那么先前设备批量传输的可用带宽就会下降到原先的 80%，传输速度可能变为 42MB/s 以下。作为一种更极端的情况是批量传输设备插入前，已有设备以控制传输方式完全占有了 20%的保留带宽，其他设备也以中断或者同步传输方式共同占据了剩下的 80%带宽，那么批量传输设备就会因为没有保留带宽只能处于等待状态了。

USB 的使用环境也是需要考虑的因素，如果系统处于一个电磁环境非常复杂的使用场合，不可避免地会受到干扰而产生传输错误。尽管这种错误在大多数通信协议(控制、中断、批量传输)的管理下并不会影响通信的最终结果，但由此引发的重发、等待等纠错工作则会明显影响数据传输速度，甚至发生堵塞的现象。因此，USB2.0 设备和 PC 主机间最好采用带磁环的 USB2.0 专用连接线，且其长度最好限制在 3~5m，这样受到的电磁干扰和噪声影响较小，纠错重发的概率也较小，数据传输的速度也就显得较快。

PC 主板的数据吞吐量也与 USB2.0 主控制器在主板芯片组结构中的位置有关，当 USB2.0 主控制器挂在 PCI 总线时就会受到 PCI 总线带宽的限制，因此较新的主板将 USB2.0 控制器与 CPU 模块直接连接，从而消除 PCI 总线的带宽瓶颈。

目前 PC 上大规模使用的 Windows 操作系统是非实时操作系统，在数据传输时，操作系统不能实时响应数据传输的请求。通常情况下设备驱动程序需要频繁产生中断，而设备驱动程序的系统优先级并不高，当计算机负荷较重或有其他优先级较高的中断发生时，它的中断得不到处理而需要等待。如果设备驱动程序向 USB 总线传递 URB 时设置一个较大的缓冲区，便可减少中断的频率从而缩短等待时间，提高数据传输的速度。此外，调整驱动程序中诸如命令排队策略、改变请求包大小等措施，也能对 USB 的数据传输速度产生一定的影响。

由于 USB2.0 的速度实在太快，设备控制器中一般单片机的数据读/写处理方式常常成为限制数据传输速度的瓶颈。因此为了实现 USB 设备与 PC 主机之间的高速传输，必须采用专门的接口电路和固件程序。

### 6.6.2 USB2.0 设备控制器的高速工作模式

EZ-USB FX2(CY7C68013)是一种支持 USB2.0 的微控制器，FX2 内部集成了 USB2.0 收发器、智能串行接口引擎(SIE)、增强型 8051 内核、8.5KB 的 RAM、4KB 的 FIFO 存储器、I/O 端口、$I^2C$ 总线接口、8/16 位数据总线和通用可编程接口(GPIF)等功能。

为了适应 USB2.0 高速数据传输的需要，EZ-USB FX2 采用了一种特殊的数据传输模式。如图 6-25 所示，其内部机制可以直接实现端点 FIFO 与外部的数据交换(沿标记线②所示的数据传输路线)，避免了传统 CPU 8051 内核固件程序访问 USB 接口端点 FIFO 后再访问外设端口通道的做法(沿标记线①所示的数据传输路线)，从而很好地解决了普通微处理器转发方式造成的带宽瓶颈。

EZ-USB FX2 还有一些非常独到的特点，如其固件可以放到主机中随时修改，每当设备列举时则自动下载到系统。同时，电路结构中 4KB 的端点 FIFO 可以通过固件程序配置成多重缓冲的形式，故可有效地提高 USB 带宽的性能、平滑带宽的抖动并减少通信等待的时间。系统具体控制方式分为从机模式和主机模式两种。从机模式是指利用外部信号控制端点 FIFO 与外设之间的数据传输；主机模式则指利用 FX2 内部集成的通用可编程接口 GPIF2

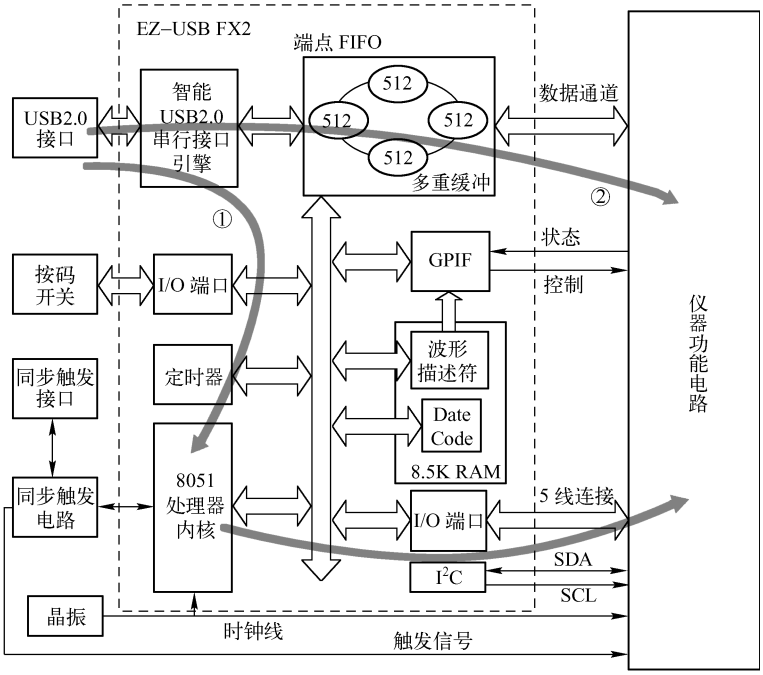



图 6-25　USB 设备控制芯片的结构与数据传输模式

(General Programmable Interface)产生时序逻辑信号来控制与外设的数据传输。GPIF 可包含 8 个时序状态，支持控制输出线 CTL、状态输入线 RDY 和地址信号 ADR，构造的各种时序逻辑信号可以有效地支持 16 位数据总线的双向操作。描述 GPIF 波形的数组称为波形描述符，可以在设备初始化时同固件程序一起下载，GPIF 将根据其表达的时序逻辑关系执行相应的控制作用。FX2 的固件程序是设备运行的核心，担负着处理 USB 驱动程序的请求，执行应用程序的控制指令，完成数据的读/写操作，以及根据需要启动 GPIF 周期，实现端点 FIFO 与外界的数据交换等功能。但固件程序可不经常参与数据传输的具体过程，通常仅起检测、判断的作用，决定 GPIF 的工作在何时开始，何时停止。图 6-26 展示了一种基于 EZ-USB FX2 的数据采集器，右侧为通用的 USB 设备控制器，通过 ATA 扁平电缆与左侧双路 12 位 20MHz 采样的数据采集模块连接。

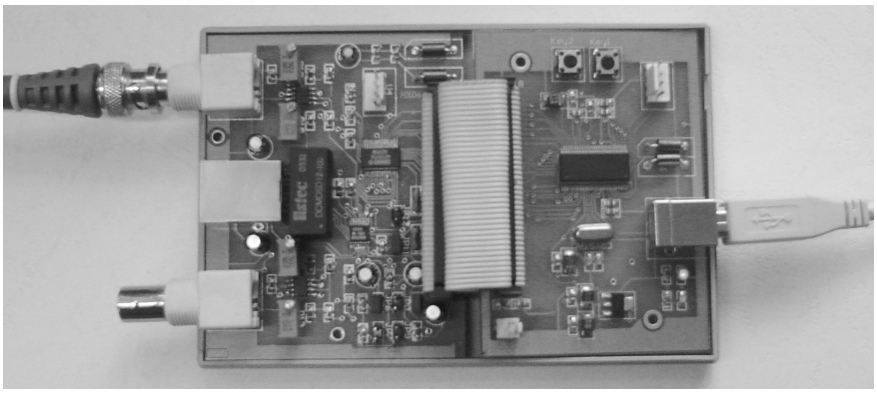


图 6-26　基于 EZ-USB FX2 的 USB 数据采集器

### 6.6.3 建立 UXI 测试总线的构想

正如 VXI 作为 VME 总线的仪用扩展、PXI 作为 PCI 总线的仪用扩展、LXI 作为 LAN 网络的仪用扩展，这里选择面向连接 PC 外设的 USB 作为基本通信总线，通过为其扩展仪用功能构建了所谓的 UXI(USB eXtension for Instruments)系统。鉴于 USB 采用的半双工方式必须通过标识包规定数据传输的对象地址和方向，故无法像 GPIB 总线发送群触发命令 GET 那样将触发动作信息同时送给系统中的各个仪器。这里将通过两个方法来解决这个问题；一是直接扩展一条触发线，专门用来实现仪器之间的触发同步；二是利用 USB 帧起始包广播发出的机制，使用原有 USB 接口，达到各设备同时执行主机触发命令的目的。

#### 1. UXI 仪用扩展触发线

借鉴 VXI 总线在 VME 总线的基础上增加专用触发线的方法，在 USB 基础上增加专用触发线是很自然的。至于触发线的数目，与 VXI 针对复杂军用系统设置多根触发线的方式不同，考虑设计的精巧和系统的成本，只增加一根专用触发线，配合 USB 数据和命令的传输，也能实现多种触发动作。这里讨论的 UXI-H1 接口模块用来完成 PC 与测试设备之间的通信和系统触发功能，包括 USB 接口电路、微控制器以及触发电路和连接端口，并与完成测试功能的电路模块连接，其系统结构如图 6-27 所示。UXI-H1 完全保持了 USB 接口的通用性，只有当选择外接触发线同步方式时，才通过结构精巧的 T 形连接器，用一根触发总线连接所需的仪器。

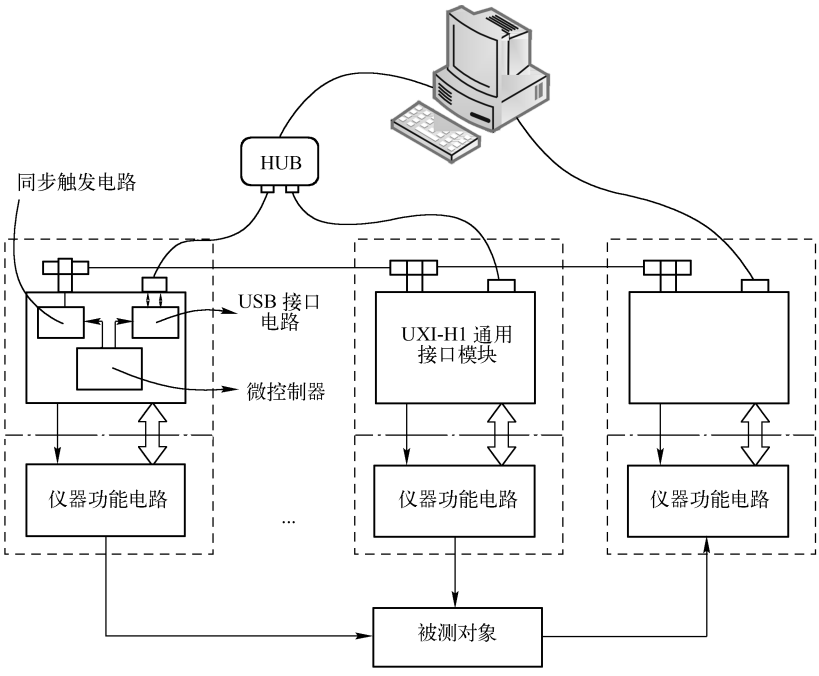



图 6-27 UXI-H1 系统总体结构图

采用外接触发线方式同步时，由主机设置一台仪器为发送者。其余则为接收者。当所有仪器准备就绪后，由发送者向触发线发送同步信号，使整个测试系统同步动作。UXI-H1 完

整的同步触发功能状态转换关系如图 6-28 所示，共有 8 种状态。

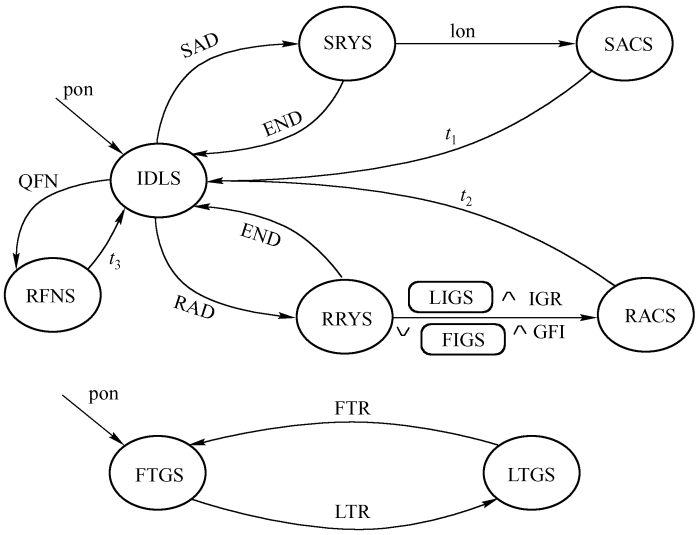



图 6-28　扩展的 UXI 触发功能状态图

系统上电复位后停留在空闲态(IDLS)，当收到来自 USB 的发送寻址命令 SAD 后则可进入发送就绪态(SRYS)，或当接受寻址命令 RAD 后进入接收就绪态(RRYS)。当然也可以通过命令 END 返回空闲态。处于发送就绪态的设备一旦收到设备的本地信号 lon(如按键、外触发等),则进入发送作用态(SACS)，向 UXI 触发线发出触发脉冲，并在 $t_1$ 时间内返回空闲态。若要再次触发，则需要主机通过 USB 重新发送命令，从而避免多次触发的误动作。而处于接收就绪态的设备当收到触发线上的触发脉冲时进入接收作用态(RACS)，启动仪器部分的触发动作，并在 $t_2$ 时间内返回空闲态。实际上，系统上电复位处在帧包方式态(FTGS)，要进入单线触发方式态(LTGS)，主机需要发送 LTR 命令。

采用 USB 命令和触发信号配合的方式，只需要一根触发线就可以根据实际测试要求触发不同的仪器。即使一台仪器有多项独立功能需要分别触发，也可以首先用 USB 命令使仪器的不同功能部件处于就绪态，然后通过触发脉冲使仪器动作。由于接口模块既可发送又可接收触发脉冲，故触发电路应为双向工作方式。实际电路采用低电平有效的负逻辑，确保总线上的“线或”关系，即任何设备均可向总线发出负脉冲的触发信号。

### 2. 基于 USB 帧信号的群触发机制

借鉴 GPIB 系统不设专用同步线，而采用群触发消息实现同步的方法，如果设法让所有设备通过 USB 本身的通信功能同时收到“群触发”消息，就可避免增加任何连线，直接利用 USB 实现自动测试系统的同步操作。

主机通过 USB 发送的下行数据本来是以广播方式发出的，但 USB 接口芯片只响应那些与令牌包中所含地址和端点相符的传输事务。基于标准化和兼容性的考虑，人们显然不愿改变这种 USB 接口芯片和集线器的原有工作方式。然而传输事务的组成除了标识包、数据包和握手包外，还有一个帧起始包 SOF。由于 SOF 在 USB 正常通信过程中不需要程序干预，以至它的存在常常被忽略。这个 SOF 是被所有设备同时收到的，且包含一组作为标志的帧序列号

(11 位)。一般 USB 接口芯片(如 USBN9604、EZ-USB FX2 等)都有一个反映帧起始包的状态事件及其寄存器标志位。这个事件也可以产生中断，因此如果精心设计接口模块的通信协议，使所有就绪设备在某个指定帧起始包到达时触发仪器动作，就可以实现类似 GPIB 的群触发机制。

如图 6-29 所示，在帧包方式态(FTGS)，所有仪器都只能作为群触发信号的接收者。PC 主机通过 USB 依次向各台仪器发送的接收寻址命令 RAD 中应包含指定的触发帧序号。这样,接口模块收到该命令后就进入接收就绪态，并当指定序号的 SOF 到来时触发仪器动作。为了确定适当的帧序号以使所有设备都能够做好触发准备，主机首先向一台仪器发送查询帧序号命令 QFN，仪器收到此命令进入报告帧序号态(RFNS)，向主机发送当前帧序号 FN1。然后主机依次向各台仪器发送就绪命令，其中包含触发 SOF 的帧序号 FN，FN 在 FN1 的基础上加上了合理的延迟。注意，为了确保命令都能够及时发送，所有命令都应采用中断传输方式发送。具体过程如图 6-29 所示。

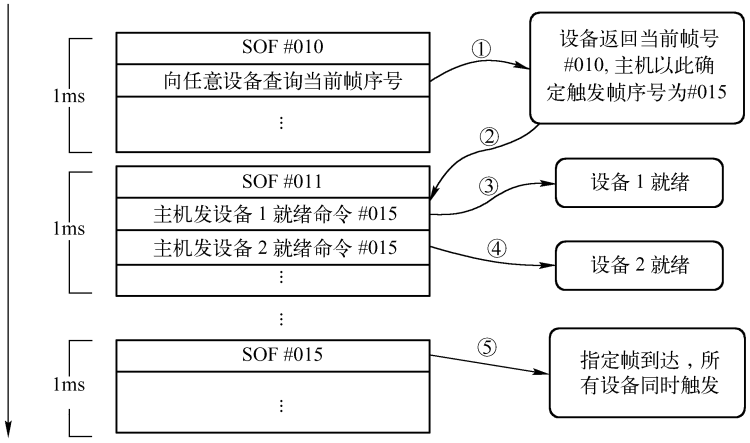



图 6-29 帧同步方式示意图

由于各设备内部微处理器处理的速度可能会有快有慢，为确保触发时刻，可以建立一个精确时钟控制的计数器，然后针对具体设备接口的操作延时进行相应的补偿。这样只要各设备确实能够同时收到同步触发信号，就可以克服处理过程的差别，最终达到对多个仪器进行同步触发操作的目的。

## 本 章 小 结

USB 总线的协议比较复杂，从物理层信号、信息包交换、传输方式都具有鲜明的特色。基于微处理器的 USB 设备控制器采用专用的 USB 接口芯片和精心设计的固件程序，将其通信功能表现为几个简明易用的功能函数，确实可以作为 USB 仪器的通用核心模块。基于 HID 类驱动程序的 VB 类函数则可直接用来实现 PC 主机虚拟仪器主体对测试设备的监控管理和数据交换。本章所述信源-数采器和 50MHz 高速数据采集器为开发 USB 基础测试设备、组建虚拟仪器系统提供了有益的参考示例。另外，本章对提高 USB 工作速度和同步触发能力所进行的探讨也将有助于 USB 发展成为名符其实的自动测试总线。

# 参 考 文 献


[1] Jan Axelson USB Complete. Lakeview Reseach, 1999.

[2] John Hyde.USB Design by Example. Intel Univrsity Press, 1999.

[3] Jan Axelson.USB Complete. Lakeview Research,1999.

[4] Compaq .HP .etc. Universal Series Bus Specification Revision2.0. April 27, 2000.

[5] Cypress Semiconductor .EZ-USB FX2 Technical Reference Manual.May,2001.

[6] 李华.USB 仪器将成为测量仪器的主流. 国外电子测量,2006.1.

[7] 刘思久, 赵蔚. 实现 USB2.0 高速数据传输的问题探讨. 计算机工程与应用,2007.2.

[8] 刘思久,李致远. 建立 UXI 测试总线的设想. 仪器仪表学报,2006(S3).

[9] 刘思久,张礼勇. 虚拟仪器与基于 USB 总线的测试设备. 电测与仪表,2003.1.

[10] 张礼勇,刘思久. 虚拟仪器与 USB 方式的探讨和实践.电子测量仪器学报,2003.2.

[11] 刘思久,汪秀丰等.支持 USB 接口的 50MHz 高速数据采集器. 电子测量技术, 2004.2.

[12] 刘思久,罗艳等.USB 设备的通用控制器.天津大学学报(增刊), 2005.

[13] 刘思久,董景, 罗艳. 用于虚拟仪器的 USB 信源/数采设备开发. 试验技术与试验机, 2004(Z2).

[14] 刘思久,滕岩峰, 冯玉贵, 张礼勇. 基于 USB 总线的测试系统开发[J]. 电测与仪表, 2002.5.

[15] 刘思久,全宏波等. USB-GPIB 总线控制器.全国第二届总线技术与测控学术报告会, 2001.

[16] 刘思久,张礼勇. UXI 虚拟仪器系统的开发平台研究.中国仪器仪表学会电磁测量仪器分会,2008.

[17] 滕岩峰,周雪峰, 刘思久.基于 USBN9604 的通用 USB 设备接口的研究与开发.微型机与应用,2003.6.

[18] 吴东艳,张礼勇等. 基于 USB 总线的数据采集/信号发生器的设计.哈尔滨理工大学学报,2005.1.

[19] 张邦涛,张礼勇. 基于 USB2.0 的数据采集器和波形发生器的设计.哈尔滨理工大学学报, 2007.2.

[20] 刘煜坤, 张礼勇. USB 设备接口的研究与开发. 哈尔滨理工大学学报, 2003.1.

[21] 张旭辉, 张礼勇等. 基于 USB 接口的电力线载波通信模块的设计. 电站系统工程, 2006.1.

# 第 7 章　PCB 电子功能模件的自动测试

装好元件的印制电路功能板(也称 PCB 电子功能模件)是电子产品与机电设备的重要组成部分，其质量直接影响整个系统的工作状态。有关测试理论和实践经验表明：在生产过程中对 PCB 功能模件进行多次质量检测，尽早发现问题和尽早采取措施，可以最大限度地降低生产成本，提高最终产品的质量。为此，从 20 世纪 70 年代开始，在此领域已经逐步形成了一个 ATE(Automatic Testing Equipment)产业，专门进行 PCB 电子功能模件的自动测试。本章将从 PCB 电子功能模件的各种测试方法和自动测试设备的基本结构开始讨论，重点阐述路内测试、功能测试和边界扫描测试的工作原理和实用技术，说明虚拟仪器在此领域的现实作用。

## 7.1　PCB 电子功能模件测试的基本概念

### 7.1.1　PCB 电子功能模件测试的基本方法

PCB 测试工作主要包括：通断测试、路内测试、功能测试和基于微处理器的总线测试，以及近年来新发展的边界扫描测试和红外热像测试等不同方式。

通断测试(Connect Test)主要是指在安装和焊接元器件前对印制电路板本身连线通断情况的测试，这时主要检测的是电路走线有无短路、断路故障，金属化孔的连接是否良好等。这是保证印制电路板正常工作的基础，它可作为 PCB 本身加工过程的最后一道工序，也可作为采用 PCB 的电子功能模件生产线的第一道工序。通常这种测试方式与 PCB 电子功能模件的测试分属两个不同的领域。

路内测试(In-Circuit Test)的主要任务是测试已经安装部分或全部元器件的 PCB 电子功能模件有无元件插错(如电阻、电容、电感、集成电路等元器件的型号、参数是否正确)、插漏、插反(如二极管、三极管的引脚是否接错，电解电容的极性是否接反，IC 的方向是否正确等)的错误现象，同时也可以检测焊接点有无漏焊、虚焊，以及所装元件本身质量等问题。路内测试的基本方法是通过专用针床中的一组测试探针，一方面给被测元件输入激励信号，另一方面从被测元件引出响应信号，以此检查元件在电路中的状态，并推断其参数。由于被测元件被连接在整个电路中，其他元件会对被测元件的测试产生不可忽略的影响，因此在测试时需要采用专门的技术将被测元件与其他元件的作用隔离开来，从而确保测试精度，这是路内测试是否成功的关键。由于路内测试是对 PCB 上的各个元器件分别进行测量，所以通用性很好，测试程序编制简单，可以在不知道 PCB 整体功能的情况下进行。路内测试也可在分阶段安装元件的模件生产过程中进行，这样有利于及早发现故障，并及时下线处理，从而降

低生产成本。

功能测试(Functional Test)用来检查 PCB 功能模件是否能够完成既定的功能要求，同时也能对一些可调参数的元件(如电位器、可调电感或电容)进行参数值的调整。进行功能测试时，需要使用一系列激励源，如直流信号、方波信号、正弦波信号或随机信号，建立模件的运行条件，以便检测模件的功能特性并找出故障。由于功能测试是针对 PCB 整体功能进行的，因此自然需要全面了解被测模件电路的实际功能要求及运行条件。同时，因为功能测试通常是在生产过程的最后阶段进行，所以能够最直观地反映 PCB 功能模件的实际使用质量，测试结果也是用户最为关心的。

功能测试和路内测试不能相互替代。路内测试用于确保 PCB 功能模件中各元器件本身功能及其安装的正确性，功能测试则用于确保 PCB 功能模件整体性能的正确性。测试理论和实践经验表明：通常可将路内测试和功能测试的各自目标设定为覆盖 70%～80%的故障范围时，对每块 PCB 功能模件依次进行这两种测试，就可得到满意的总体测试效果和更高的总故障覆盖范围。

总线测试(Bus Test)专门用于那些以微处理器为核心的 PCB 功能模件。在微处理器总线结构下，挂在总线上的器件运行必然直接影响总线的状态，因此如果采用适当的测试程序对挂在总线上的接口器件进行充分的 I/O 操作，则有可能将这类器件的测试简化为对总线状态的分析。在总线测试时，通常使用特征分析仪和逻辑分析仪。前者通过对总线上的数据序列进行特定的移位反馈记录，得到不重复的特征字，从而快速判定总线工作是否正常；后者则可对总线传输数据进行严格的时序分析，从而找出故障的具体根源。

边界扫描测试(Boundary-Scan Test)的主要思想是在芯片引脚和芯片内部逻辑电路之间增加边界扫描单元电路，这些边界扫描单元依次连接成串行扫描链，通过串行扫描的方式设定和读取芯片的引脚状态来实现对芯片本身逻辑功能和器件之间连线的测试和诊断。边界扫描测试需要被测电路的支持，即在其设计过程中必须经过可测性设计。这种测试方法只需连接标准接口的少数引脚，大大减化了测试设备和探头装置。

红外热像测试(Infra-red Thermal Imaging Test)的基本思路是：当一个元件的参数发生变化时，元件上的电压和电流必有变化，导致它消耗的功率及其表面温度产生变化。如果对印制电路板(PCB)进行红外摄像，通过计算机来寻找温度具有显著差异(与无故障标准模件的情况比较)的部位，即可判断故障元件的位置。现代的红外摄像仪的灵敏度相当高(0.1℃以下)，完全可以达到测试所需的要求。其不足之处在于投资过高，且某些故障不可测量(如电容短路等)。

### 7.1.2　传统 ATE 设备的基本情况

传统的大型 ATE 测试设备往往包揽上述多种测试方法，通过一台微计算机选择测试方式，并用软件操纵测试设备，分别完成不同的测试内容。

如图 7-1 所示，测试前需要熟悉被测电子模件的电路原理图，标示出图中所有感兴趣的节点作为测点。对功能测试，应切实保证被测电路正常工作所需要的电源和其他必要的外部条件，确定施加外部信号和激励信号的位置，同时选择适当的信号参数，如为典型模拟信号选择频率、幅值、相位，为开关信号选择通、断状态，为任意波信号选择波形数据，为多通道的数字信号设计测试向量。另外,需要考虑测量相应的响应信号，并规定期望的标称值和上、

下允许误差。由于功能测试的项目可能很多，每一项还可能包括若干测试步骤，而且每一个测试步骤所选择的测点位置、信号源、测量设备和测试要求都可能不同，因此需要编制一个完整的测试程序。当然，测试程序通常并不需要用户使用编程语言来产生具体代码，而是通过一些特定的人机对话和填表方式建立一个测试数据表单。这个测试数据表单包含上述所有测试内容要求的全部信息，依据这个测试表单，自动测试系统就可以指挥各种测试设备依次进行各项测试工作，并将测试数据与期望数据比较后显示测试结果了。对于路内测试(也称在线测试)，由于测试目标是要测试电子模件中的各个元件，因此电路原理图中几乎所有的节点都要进行标识。此时通常也是通过人机对话的方式建立一个以被测元件为索引的列表数据结构，包括各元件的标识名、额定值、允许误差、所在节点位置，以及测试过程所需施加的电压、电流、频率、保护点等参数数据。也是依据这个测试数据清单，就可以通过自动测试系统的硬件和软件完成测试任务了。

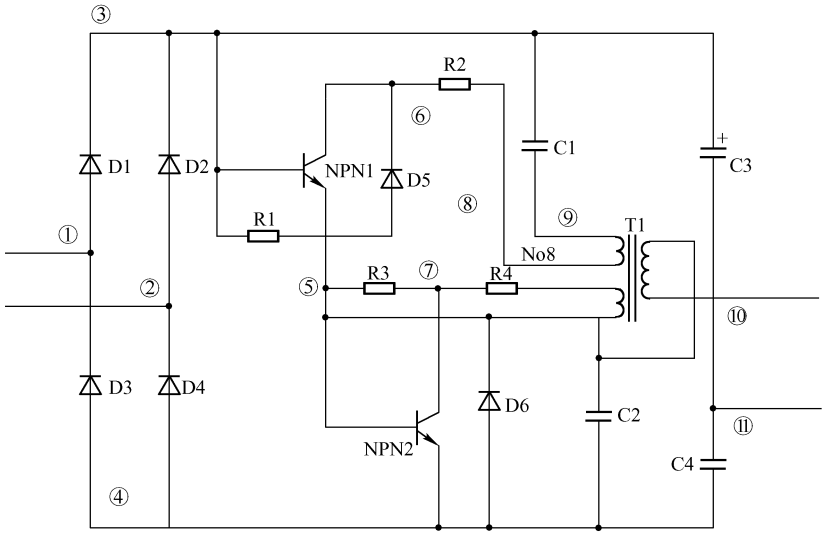



图 7-1 标识被测电路原理图中的节点



图 7-2 罗马尼亚 FD5450 PCB 测试系统

早期的 ATE 系统大多由 GPIB 设备组成，人们可以根据测试内容的需要，选择适当精度和量程的通用测试仪器，组成机架式结构(如图 7-2 所示)。后来随着技术的发展，具体的形式发生了许多变化，台式仪器的机架变成了卡式仪器的机箱，乃至虚拟仪器。此外还出现了一些专门设计的设备，总体体积大大缩小。

在 PCB 模件自动测试系统的开发方面，国外厂家大都将其设计目标定位于“通用型”和“傻瓜式”操作方式，开发了诸多的标准测试系统，如 HP 公司的 306X、FairChild

公司的 207、AGILENT 公司的 3070、GENRAD 公司的 2284I 系列等，典型系统的外形结构可参见图 7-3 和图 7-4。这些系统虽然功能强大，但价格昂贵，有时效率也不很高 。因此，并非所有厂家都完全依赖这种标准测试设备来解决自己的电子模件测试，而是在部分装备这种设备的同时，也自行开发或委托开发一些专用的测试系统，以满足生产过程多产品、多层次、多道工艺和需要多台设备同时高效工作的实际要求。



图 7-3　美国 AGILENT 公司 3070 系统



图 7-4　美国 GENRAD 公司 2284I 系统

### 7.1.3　针床和夹具结构

传统的 PCB 测试一般是将被测 PCB 固定在针床上，如图 7-5 所示，激励信号由触针施加到被测 PCB 的测试点上，相应的响应信号也是通过被测 PCB 测点上的触针取得，并通过多路通道开关连接到测试设备。针床本身是一块钻满定位孔的绝缘板(称为针床体)，这些定位孔的位置与被测电路节点在 PCB 模件上的具体位置一一对齐，先将导电外套安装在定位孔中，并通过尾部引出接线，然后将触针配件以紧配合的方式固定在外套中，从而使触针配件横向对准 PCB 的测点。标准的触针配件本身由内导套、弹簧和针头组成，在其内部弹簧的压力下将触针纵向推往接触测点。触针配件具有不同类型(其针头形状不同)，可以整体从外套

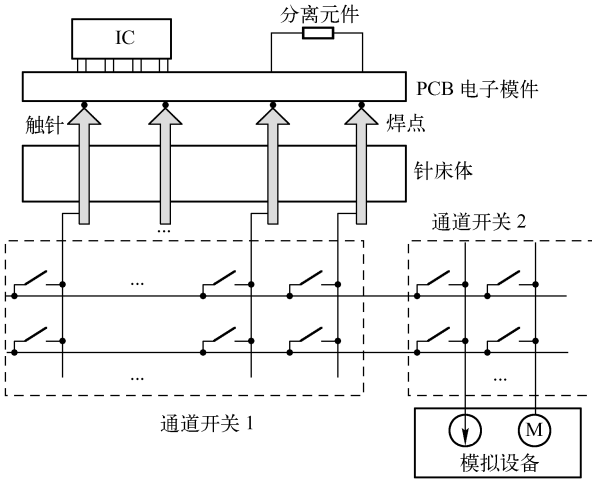



图 7-5　PCB 模件测试的针床连接示意图

中拔出更换。为了确保触针与被测 PCB 模件的节点接触良好，应根据不同的测试对象，正确选择具有适当针头形状的内针体。图 7-6 给出了触针配件的内部结构和一些针头的形状。

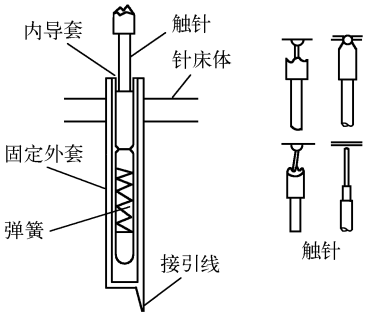



图 7-6 触针内针体结构和针头形状

针床体被固定在针床座上构成了针床的基本结构，针床需要配置装卸 PCB 功能模件的机构，真空吸附和机械锁紧是两种常用的方法。前者利用设备在针床体内形成的真空，可将 PCB 功能模件吸向针床体，从而使触针探头接通被测电路；后者则利用机械锁紧机构的机械动作直接将被测 PCB 模件压向针床体，也同样能让触针接通被测电路。

另外，对双列直插式芯片结构有一种专用的夹具(如图 7-7 所示)，可以同时接触芯片的全部引脚，在故障诊断和故障排除的一些场合也经常采用。

由于针床结构要求为每种被测模件配置专门的针床，因此增加了测试的成本。为此出现了一种称为“飞针”的方法，即以快速定位的机械手结构动态连接被测点，然而这种技术要求较高，快速性、接触质量和成本控制都不容易。

图 7-8 所示是红外热像的实际影像，显然这种方法也不需要针床，具有非接触测量的优点。

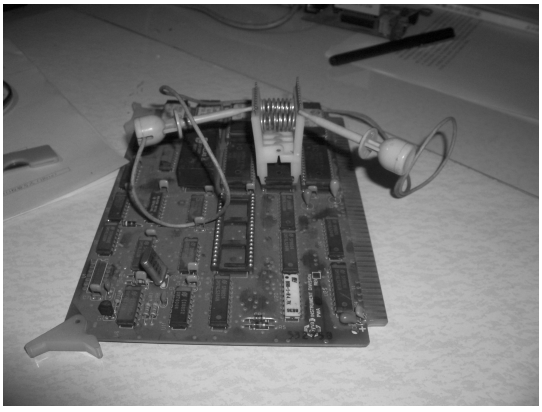


图 7-7 双列直插式芯片专用夹具

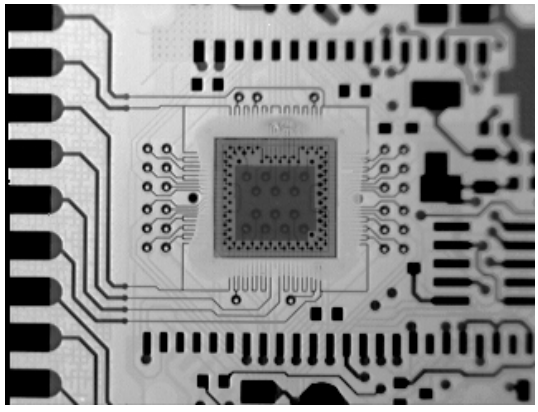


图 7-8 红外热像图

### 7.1.4 PCB 功能模件测试要求的新动向

对生产线上 PCB 电子功能模件的自动测试任务来说，通常被测产品的设计问题已经解决，于是板上每个元件本身功能是否正常就成为这个模件测试的主要问题。由于“路内测试”在技术上是通用的，因此生产厂家常常直接采用标准的 ATE 设备来完成这项工作。标准设备通常价格昂贵，其特点是“大而全”，包括各种模拟元件、数字元件、存储器、甚至微处理器的测试资料，能够自动生成测试程序，操作简单，使用方便，可以称为“傻瓜式”工作模式。但这种设计思路导致开发工作需要耗费极大的人力、费用和时间，这是其价格昂贵的主要原因。我们注意到近年来 PCB 功能模件本身出现的一些新变化，故对 PCB 功能模件自动测试

设备的开发需要考虑一些新的问题。

首先，当代 PCB 功能模件常以 1~2 块专用的大规模芯片为核心，配接一些外围的阻容元件构成，而通用的中、小规模逻辑电路芯片则被各种可编程逻辑器件所代替。此时传统标准 ATE 设备大而全的元件库就变得意义不大了，而真正需要测试的专用芯片却根本来不及事先包括在标准的测试数据库中，因此所谓的“大而全”也就不能真“全”了。

其次，电力电子技术的发展使许多原来并不使用 PCB 的机电设备也变成以印制电路板为依托的 PCB 功能模件。如汽车、发动机的控制电路、电动机的软启动和节能设备、电话机、直流电源、空调机的控制、洗衣机的控制、电饭锅的控制、日光灯镇流器等都已广泛使用 PCB 功能模件。在此类 PCB 功能模件中，属于电力电子的模拟器件占有很大的比重，要求配置设备的功率和电流往往较大，这是传统标准 ATE 设备所不能完全涵盖的，因此需要更灵活地配备专用设备。

再次，随着电子产品小型化的发展，芯片集成程度的提高和表面贴装技术的普及，使双列直插式芯片和针床结构式的探头连接方式受到很大限制，导致仅用一组标准接口的“边界扫描测试”成为更有效的方法，因此往往需要结合被测 PCB 功能模件本身的可测性设计建立适当的测点、测试向量和测试策略。

另外，由于电子功能模件内部广泛使用了 $I^2C$、SPI 等串行通信方式，传统的“总线测试”也发生了一些变化，特征分析和逻辑分析的概念得以扩展，而通道数目却不多。

最后，除了这些适用于生产线上的测试设备外，对因某种“问题”下线的 PCB 功能模件还需要提供一些专门用于维修与故障诊断的测试设备(包括专家系统)。实际情况表明，此时故障维修的工作效率往往大大落后于生产线上测试的工作效率，因此亟待改进。

综上所述，尽管目前“大而全”的标准 ATE 系统仍在国际市场上占据统治地位，但结合我国国情和市场需求，考虑近年来 PCB 功能模件本身出现的变化，在虚拟仪器基础上建立新型 PCB 功能模件自动测试系统则更有吸引力。

## 7.2　模拟电路的路内测试

PCB 上的模拟元件包括电阻器、电容器、电感器、二极管、三极管、场效应管和运算放大器等。电阻器可以直接测量其阻值，电感器和电容器则可以通过测量其阻抗来推算其电感量和电容量，二极管则要了解其正、反向特性曲线，故本质上说，对这些元件的测试就是建立其伏-安特性曲线，只是有的是直流特性，有的是交流特性；有的是线性关系，有的是非线性关系。三极管和运放的测试略有不同，前者可用其电流放大倍数来判断器件是否合格，后者则只需考虑闭环条件下的放大作用。由于被测模拟元件与周围元件相互连接，已经形成了一个复杂的网络，因此测试的关键就在于如何将被测元件从周围电路网络中分离出来，使其周围连接的元件不再影响被测元件本身参数的确定。

### 7.2.1　PCB 模拟器件路内测试的基本原理

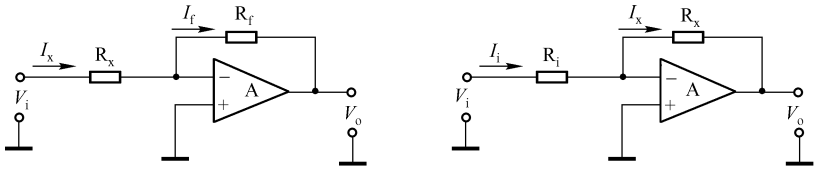
测试元件伏-安特性的常用方法是恒流源法和恒压源法。前者对被测元件施加已知电流，测取其电压响应。后者则对被测元件施加已知电压，然后测取电流响应。最终都是通过伏-安特性关系，或者算出线性阻抗，或者找出伏-安特性的特征。

如图 7-9(a)所示，$R_x$ 是待测元件，根据运算放大器的“虚地”概念，放大器的反向输入端接近地电位，因此被测元件两端施加的电压可以看做所加激励电压 $V_i$，且由于运放高输入阻抗的原因，可有 $I_f=I_x$，运放输出端电压就可以看做这个电流与反馈电阻的乘积。于是可以得出

$$R_x=-R_f\cdot V_i/V_o \tag{7-1}$$

的简明关系。即对被测元器件施加已知电压，通过测量运放输出端电压就可算出 $R_x$ 的实际阻值。恒流源法测试原理则如图 7-9(b)所示，具体实施时可将已知恒压源加载到一个高精度、已知阻值的输入电阻 $R_f$ 上，从而产生希望的恒定电流 $I_f$。考虑 $I_f=I_x$，这个电流就是激励被测电阻的电流 $I_x$，于是通过测取运放输出端电压，就可算出

$$R_x=-R_f\cdot V_o/V_i \tag{7-2}$$




(a) 恒压源法　　(b) 恒流源法

图 7-9　测试原理的恒压源法与恒流源法

恒流源法的好处在于作为测量结果的输出电压 $V_o$ 与 $R_x$ 成正比，不必进行倒数计算。但采用恒流源法有可能损害被测板上的其他器件，如标称 10kΩ 的电阻错焊成 100kΩ 电阻时，就有可能在实际被测电阻 $R_x$ 上产生相比原先高出 10 倍的电压，从而可能损害与之连接的其他器件。另一方面，在恒压源法构成的测试电路中，反馈回路简洁(与被测电路无关)，容易确保稳定。

### 7.2.2　阻抗测试的相关分析法

将上述为测试电阻推出的式(7-1)和式(7-2)推广到电感或电容的测试，则只需用阻抗 $Z_x$ 代替电阻 $R_x$，同时将激励信号改做交流信号即可。不过由于此时 $V_o/V_i$ 成为一个复数，应当包括幅值和相位的测量。为此引入相关辨识的方法，其基本思想是要避免直接在时域进行幅值和相位的测量，而是通过被测对象输出信号与其激励输入信号的一对正交分量的相关计算得出。

如图 7-10 所示，设被测线性系统的频率特性为

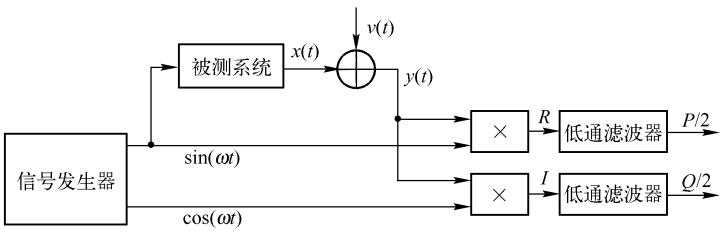



图 7-10　频率特性的相关测试法原理

$$G(\mathrm{j}\omega)=|A(\omega)|\angle\phi(\omega)=P(\omega)+\mathrm{j}Q(\omega) \tag{7-3}$$

则其理想输出信号为

$$\begin{aligned}x(t)&=A(\omega)\cdot\sin[\omega t+\phi(\omega)]\\&=A(\omega)\cdot\cos\phi(\omega)\cdot\sin\omega t+A(\omega)\cdot\sin\phi(\omega)\cdot\cos\omega t\end{aligned}$$

假设干扰信号 $v(t)$ 的均值为零，且与 $v(t)$ 相互独立，则被测电路的实际输出信号为

$$y(t)=x(t)+v(t)=P(\omega)\cdot\sin(\omega t)+Q(\omega)\cdot\cos(\omega t)+v(t)$$

依据如图 7-10 所示的相关测试方法，可得

$$\begin{aligned}R(\omega)&=y(t)\cdot\sin(\omega t)=P(\omega)\cdot\sin^2(\omega t)+Q(\omega)\cdot\cos(\omega t)\cdot\sin(\omega t)+v(t)\cdot\sin(\omega t)\\&=P(\omega)\cdot[1-\cos(2\omega t)]/2+Q(\omega)\cdot\sin(2\omega t)/2+v(t)\cdot\sin(\omega t)\\&=P(\omega)/2-P(\omega)\cdot\cos(2\omega t)/2+Q(\omega)\cdot\sin(2\omega t)/2+v(t)\cdot\sin(\omega t)\end{aligned}$$

从而推出 $G(\mathrm{j}\omega)$ 的实部分量计算方法如下：

$$P(\omega)=2\times R(\omega)\text{的直流分量} \tag{7-4}$$

同理，可知

$$\begin{aligned}I(\omega)&=y(t)\cdot\cos(\omega t)=P(\omega)\cdot\sin(\omega t)\cdot\cos(\omega t)+Q(\omega)\cdot\cos^2(\omega t)+v(t)\cdot\cos(\omega t)\\&=P(\omega)\cdot\sin(2\omega t)/2+Q(\omega)\cdot[1+\cos(2\omega t)]/2+v(t)\cdot\cos(\omega t)\\&=P(\omega)\cdot\sin(2\omega t)/2+Q(\omega)/2+Q(\omega)\cdot\cos(2\omega t)/2+v(t)\cdot\cos(\omega t)\end{aligned}$$

从而推出 $G(\mathrm{j}\omega)$ 的虚部分量计算方法如下：

$$Q(\omega)=2\times I(\omega)\text{的直流分量} \tag{7-5}$$

于是，根据式(7-3)，可得

$$\frac{V_o(\omega)}{V_i(\omega)}=G(\mathrm{j}\omega)=P(\omega)+\mathrm{j}Q(\omega)=|P^2(\omega)+Q^2(\omega)|\angle\tan^{-1}\frac{Q(\omega)}{P(\omega)} \tag{7-6}$$

考虑图 7-9(a)和式(7-1)，可以推出恒压测量方法中被测阻抗的计算方法如下：

$$Z_x(\omega)=-R_f\frac{1}{V_o(\omega)/V_i(\omega)} \tag{7-7}$$

显然，阻抗 $Z_x$ 也是测试频率的函数，且可写成以下复数形式：

$$Z_x=R+\mathrm{j}X=R+\mathrm{j}\omega L+\frac{1}{\mathrm{j}\omega C} \tag{7-8}$$

为激励源选择不同的正弦波角频率，根据复数量计算的幅值相角关系列出联立方程组，即可解出相应被测阻抗的电阻、电感和电容分量。这里，$\omega$ 表示激励输入电压的角频率，$R$、$L$ 和 $C$ 分别表示被测阻抗中的电阻、感抗和容抗分量。

### 7.2.3　保护地的概念和三线法路内测试

由于在线测试过程中被测元件与其他元件连在一起，因此必然会对测量产生影响。如图 7-11所示，对于被测电阻 $R_1$ 来说，存在几条与其并联的通道，如 $C_1$-$Q_1$-$R_3$-$C_2$、$L_1$-$R_4$-$R_3$-$C_2$、$C_1$-$Q_1$-$R_2$ 等，因此直接测试得到的阻值将是它们的并联值，误差很大。

为了把被测元件 $R_1$ 从周围电路中分离出来，可按恒压法将激励源 S 加在测点①、将测试运放的反向输入点(虚地)$I$ 连接测点②的同时，通过测试触针将电路节点③和节点⑦连接到测试运放的输入地 G。这样可得如图 7-12 所示的三线测试等效原理图。图中 $Z_x$ 是被测元件阻抗，$Z_1$ 表示 $Z_x$ 左端与保护地之间所有元件的等效阻抗(图 7-11 中节点③、⑦与节点①之

间的所有元件的等效作用)，$Z_2$ 表示 $Z_x$ 右端与保护地之间所有元件的等效元件(图 7-11中节点③、⑦与节点②之间的所有元件的等效作用)。测试设备由输入电压信号源 $V_i$、测试运算放大器及反馈电阻 $R_f$ 组成。由于测试电路包括加载激励电压信号的 S 触针、测取电流响应信号的 I 触针和用于保护接地的 G 触针，因此这种方法称为“三线测试法”。

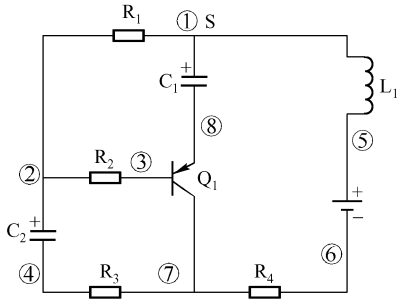



图 7-11　在线测试的隔离问题

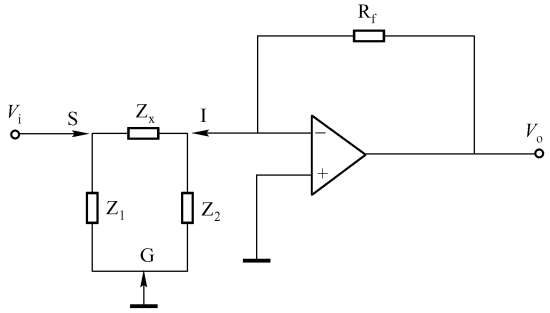
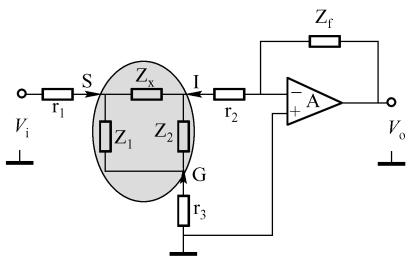



图 7-12　三线测试原理

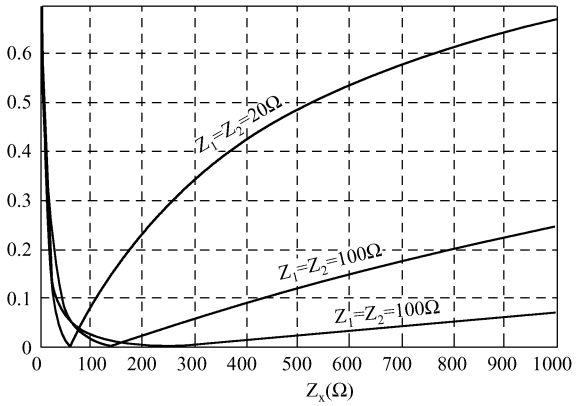
若假定激励源与运放都是理想器件，且不考虑引线和触针本身的连接电阻，则尽管 $Z_1$ 上有电流流过，却不会影响 $V_i$ 对 S 端的电压设定值。同时根据运放反向输入端的“虚地”原理，I 点电位近似为零，这样 $Z_2$ 上将没有电流通过，造成流过 $Z_x$ 的电流与流过 $R_f$ 的电流相等，因此仍然能够保持测试电路中 $R_x=-R_f\cdot V_i/V_o$ 的理想关系。故仍可根据已知输入信号 $V_i$ 和设定反馈电阻 $R_f$，以及测量得到的运放输出电压 $V_o$，算出被测元件 $Z_x$ 的阻抗参数。然而，由于实际测试过程中触针通常可能存在 0.5~1Ω 的电阻，激励源存在内阻，因此在信号源内阻及 S 触针电阻上存在压降，使实际加在 $Z_x$ 左端的电压低于 $V_i$。而且 G 触针上也有电流流过，G 点的电位并不为零，I 点的电位也不为零，故 $Z_2$ 上将有电流通过，造成流过 $Z_x$ 的电流并不完全流过 $R_f$，从而使 $Z_1$ 和 $Z_2$ 的存在引起误差。

### 7.2.4　三线法路内测试的误差分析

在如图 7-13(a)所示的三线测试法中，假定被测电阻为 $Z_x$，与其相邻并联电路的等效电阻分别为 $Z_1$ 和 $Z_2$，触针电阻 $r_1=r_2=r_3=1\Omega$。




(a) 考虑连线电阻的测试电路




(b) 误差曲线

图 7-13　三线法测试电路的误差分析

对 $Z_1$、$Z_2$ 和 $Z_x$ 组成的网络进行“三角/星”变换后，再串接 $r_1$、$r_2$ 和 $r_3$，然后再做“星/三角”变换，可得 $Z_x$ 的等效测试值为

$$Z_m = r_1 + r_2 + Z_x \frac{Z_1 \cdot Z_2 + r_1 \cdot Z_2 + r_2 \cdot Z_1 + r_3 \cdot (Z_1 + Z_2)}{Z_1 \cdot Z_2 + r_3 \cdot (Z_1 + Z_2 + Z_x)} + \frac{r_1 r_2 \cdot (Z_1 + Z_2 + Z_x)}{Z_1 Z_2 + r_3 \cdot (Z_1 + Z_2 + Z_x)} \tag{7-9}$$

相应的测试误差为

$$E_m = \frac{Z_m - Z_x}{Z_x} = \frac{r_1 + r_2}{Z_x} + \frac{r_1 Z_2 + r_2 Z_1 - r_3 Z_x}{Z_1 Z_2 + r_3 (Z_1 + Z_2 + Z_x)} + \frac{r_1 r_2 \cdot (Z_1 + Z_2 + Z_x)}{Z_x [Z_1 Z_2 + r_3 (Z_1 + Z_2 + Z_x)]} \tag{7-10}$$

因为 $r_1$ 和 $r_2$ 的阻抗与电路其他部分阻抗相比很小，所以式(7-9)与式(7-10)中最后一项可以忽略不计，而在计算误差的式(7-10)中，第一项是单独测量 $Z_x$ 本身的误差项，第二项才是由于接地触针影响所产生的误差项。变换此项，可得

$$E_m = \frac{r_1 Z_2 + r_2 Z_1 - r_3 Z_x}{Z_1 Z_2 + r_3 \cdot (Z_1 + Z_2 + Z_x)} = \frac{\dfrac{r_1}{Z_1} + \dfrac{r_2}{Z_2} - \dfrac{r_3 Z_x}{Z_1 Z_2}}{1 + \left( \dfrac{r_3}{Z_1} + \dfrac{r_3}{Z_2} + \dfrac{r_3 Z_x}{Z_1 Z_2} \right)} \tag{7-11}$$

如果$\frac{r_3 Z_x}{Z_1 Z_2}$的值很小，那么误差的大小与 $Z_x$ 的值无关。而若$\frac{r_3 Z_x}{Z_1 Z_2} \gg 1$，则误差的值接近100%。这里需要注意的是如果$\frac{r_1}{Z_1} + \frac{r_2}{Z_2} = \frac{r_3 Z_x}{Z_1 Z_2}$，则会导致误差项 $E_m$ 为 0，所以此误差项的大小并非单独由 $Z_x$ 决定，而是 $Z_1$、$Z_2$ 及接地电阻共同作用的结果。

图 7-13(b)表明当触针电阻 $r_1 = r_2 = r_3 = 1\Omega$ 时，$Z_1 = Z_2 = 20\Omega$、$50\Omega$、$100\Omega$ 时的测试误差与被测电阻 $Z_x$ 之间的关系。注意，误差曲线的最低点实际上是一种误差相互抵消的结果，并不意味如此可以真正获得高精度。从图中可以看出随 $Z_x$ 值的增加，误差值随之增大，特别是 $Z_1$、$Z_2$ 的阻抗值较小(小于 20Ω)时，一般电阻的误差可达 30%～100%。即便 $Z_1$、$Z_2$ 值达到一定的水平(例如均大于 100Ω)，在测试低值电阻(例如 10Ω 以下)和高值电阻(10kΩ 以上)时，误差仍然不小。

导致误差的主要原因是与被测元件相连的触针电阻造成的测试误差和接地触针造成的测试电流从 $Z_1$ 和 $Z_2$ 分流的影响，为了尽可能消除这种影响，可以采用另一种称为六线测试的方法。

### 7.2.5　六线测试的原理

六线测试法在三线测试法的基础上增加 3 根触针(A、B、L)，其原理如图 7-14 所示。

首先通过测取输入节点处的实际电压 $V_1$ 作为激励电压的计算值，从而直接消除了 $Z_1$ 的分流、激励源接线电阻 $r_1$ 及本身内阻压降的影响。且由于数据采集器前置调理电路的输入阻抗很高，电压测量在接线电阻 $r_4$ 上就没有压降($V_r = V_1$)。其次，为了消除 $Z_2$ 的分流，将运放的接地端改接到保护节点 $V_3$，从而使 $Z_2$ 两端的电位差降为零，这是因为高增益深反馈的测试运放输入电压“虚短”的原因。再者，将反馈电阻直接连接到被测阻抗节点处，就使连接电阻 $r_2$ 上的反馈电流减小为运放输入电流(可视为 0)，而且此时流过 $R_f$ 的电流可以通过测量电压 $V_o$ 和 $V_f$ 精确地计算出来。基于上述分析，最终采用的计算公式为

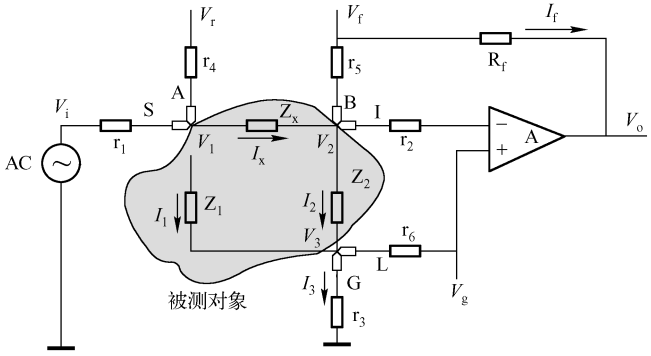



图 7-14 六线测试的工作原理图

$$Z_m = \frac{V_r - V_g}{(V_f - V_o)/R_f} \tag{7-12}$$

式中，$V_o$ 为运放输出电压，$V_f$、$V_r$、$V_g$ 分别为新增加 3 根测试连线处的测量电压。分子表示实际测得的激励电压，而分母则为实际测得的响应电流。

六线法考虑全面，只要测量电路的输入阻抗足够高，测量得到的 $V_r$、$V_g$、$V_f$ 和 $V_o$ 就能准确地体现电位 $V_1$、$V_3$ 和电流 $I_f$ 的实际值。这样，尽管测试电路使用的运放并非理想(输入电阻和放大倍数不可能无穷大)，只要选择参数时注意下列要点(这里不做具体的推导和论证)，就能获得较好的测试精度：

(1) 考虑 A/D 转换器的测量精度，需使测量电压 $V_r$ 和 $V_o$ 的取值范围在 0.1~10V 之间，以确保适当的线性范围，故应根据被测元件有关阻抗 $Z_x$ 的标称值来确定激励电压 $V_i$ 和反馈电阻 $R_f$。

(2) 应选择放大倍数 $A$ 高，输入阻抗 $R_{in}$ 高，输入偏置电流 $I_0$ 小的测试运放元件。通常 $A>10^5$、$R_{in}>2M\Omega$ 的运算放大器是很普通的。这样，若能使 $I_0/I_f$ 小，则计算误差就小。建议选择 $I_0<50nA$。

(3) 尽管被测电路的等效阻抗 $Z_2$ 难以限制，但只要选择间隔不能过大，误差就能较小。

表 7-1 给出了典型参数条件下测试误差的计算结果，可见合理选择测试电压和反馈电阻后六线测试便可达到满意的精度。

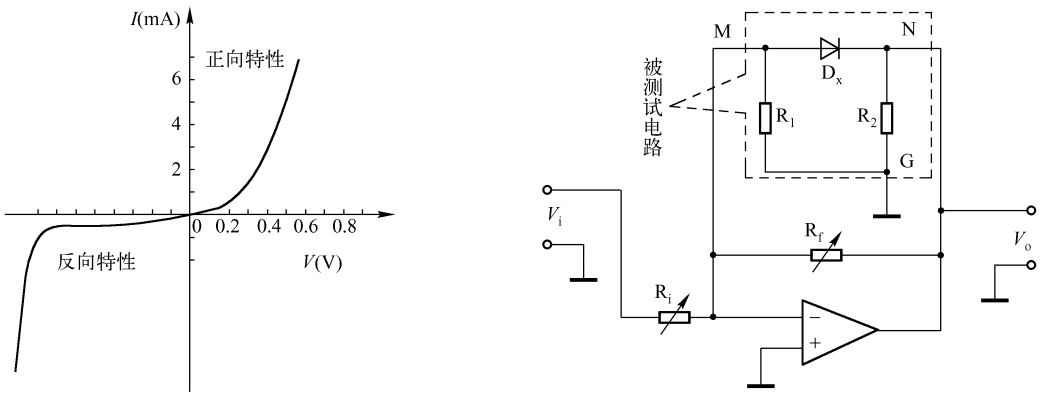
**表 7-1 六线测试的一些典型参数示例**

| $Z_x(\Omega)$ | $Z_1(\Omega)$ | $Z_2(\Omega)$ | $V_i(V)$ | $R_f(\Omega)$ | 误差 |
|---|---|---|---|---|---|
| 100 | 大于 10 | 大于 10 | 1 | 100 | 小于 1% |
| 1000 | 大于 10 | 大于 10 | 10 | 1000 | 小于 2% |
| 10 000 | 大于 10 | 大于 10 | 10 | 10 000 | 小于 2% |
| 100 000 | 大于 10 | 大于 100 | 10 | 100 000 | 小于 2% |

注意，路内测试的目的是通过在线估计元件的参数来诊断故障，不是测量元件的精度。因此，2%的测试误差就已能满足要求了。

### 7.2.6　非线性器件的测试方法

非线性器件的测试需要注意激励信号的施加方式。以二极管或稳压管为例，观察如图 7-15(a)所示的 V-I 特性曲线。可见正向导通电压和反向击穿电压附近的等效动态电阻极小，电压的微小变化会造成电流的突变，因此很容易损坏信号源设备和被测二极管本身，故宜采用恒流法测试结构。




(a)V-I 特性曲线　　(b) 测试电路

图 7-15　非线性器件(二极管/稳压管)的路内测试

如图 7-15(b)所示，用虚线框起的被测电路中 M 点电位近似为 0(根据运放的虚地效应)，因此流过 $R_1$ 的电流很小，于是可知流过被测二极管的电流为

$$I_D = \frac{V_i}{R_i} + \frac{V_o}{R_f} \tag{2-13}$$

同时，被测二极管的端电压就是测试设备的输出电压 $V_o$，因此可以直接建立二极管的特性曲线。这里反馈通道的 $R_f$ 用于被测电路开路时避免运放饱和所采取的理论措施，实际可以通过非线性的钳位电路限制此刻被测节点处的最大输出电压。

### 7.2.7　虚拟仪器的实现方案

测试系统的开发采用虚拟仪器的设计思想，由计算机软件实现测试所需要的上层操作功能和各种分析、计算功能。图 7-16 为 PCB 路内测试设备的结构框图，其硬件部分由核心控制模块、测试电路模块、多路开关及针床组成。

测试电路模块使用 A、S、B、I、L、G、M 和 N 等 8 条测试总线与多路开关的一端相连，再将多路开关的另一端与被测 PCB 针床相接。用户通过计算机操作界面向测试设备发送命令，经过设备的核心控制模块解释后，由 D/A 转换器产生模拟激励信号，再经测试电路模块，根据被测模拟元件的类型发送到相应的测试总线。同时，根据当前所测元件在被测 PCB 上的具体位置，动作多路开关，将测试总线连接到针床上相应测点。从测点得到的响应信号经过多路开关和测试总线传入测试电路模块，经信号调理后由 A/D 转换器形成数字数据后上传计算机。计算机处理这些数据后，再以合适的形式显示测试结果。

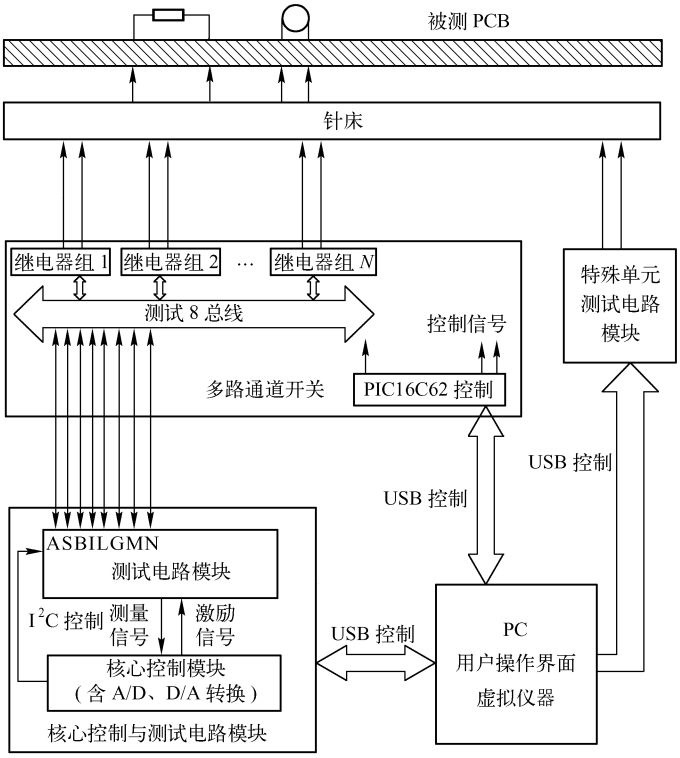



图 7-16 PCB 路内测试的一种实现方法

核心控制器由微控制器和 USB 总线接口构成，一方面通过 USB 接口接收 PC 的控制命令、回传测取的工作状态和数据；另一方面通过 $I^2C$ 总线对测试电路模块协调控制，收发与它们有关的命令、状态和数据信息。A/D、D/A 转换器是产生激励信号和采集测量信号的基础手段，它可以利用核心控制模块内部芯片的功能直接实现，也可以单独构建，还可以直接使用可程控的通用信号发生器和数字示波器。

测试电路模块根据被测元件的不同，提供了不同的测试电路单元，但均通过 8 条测试总线与外部相连。测试电路模块将调理后的激励信号送至测试电路单元，再通过测试总线施加于测点；又将测试电路单元得到的响应信号调理后送 A/D 转换。测试单元的选择和参数的调整由单片机 PIC 控制器实现，PIC 单片机通过 $I^2C$ 总线接受核心控制模块的控制，成为测试电路模块中能够接受命令的通用接口。

8 条测试总线分为 A-S、B-I、L-G 和 M-N 等 4 组，开关矩阵中每 4 个双刀开关继电器组成的继电器组使每个与针床相连的测点均可自由选择与测试模块的相应功能相接。这样一块包含 240 个继电器的模件可提供 60 对测点，当电路板上测试点较多时，可增加继电器板的数量。

测试操作界面如图 7-17 所示，主界面一目了然地显示出所有被测元件的列表。为了指出各被测元件在电路图中的节点位置，用户在使用本文所述的测试系统时需要首先在电路原理图中标出所需要的测试节点，并据此配置多路开关的通道、进行针床内部的连线。然后根据被测电路元件的实际参数和结构确定测试策略，形成如图底部所示的测试表单中的各项测试要求。

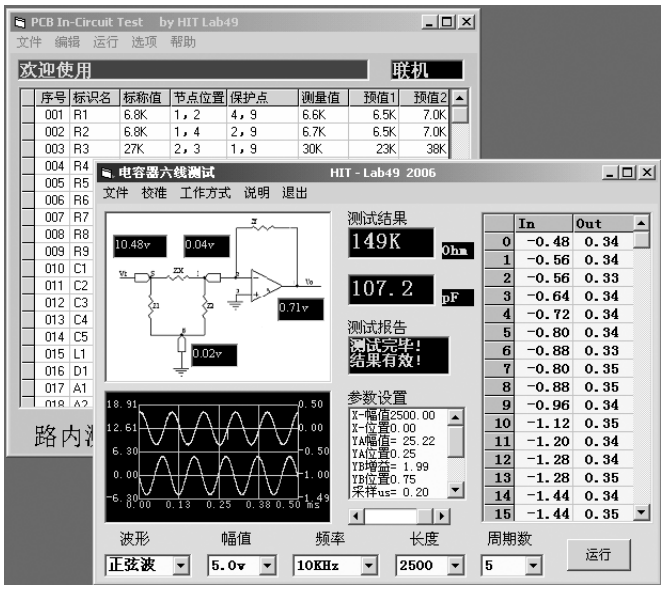



图 7-17　测试操作界面

由于操作主窗口测试列表的各行记录依次包括所有被测元件的标识名、标称值、节点位置、保护点位置、实际测量值、预期值等信息。如对列表第 1 行指出被测元件为电路中的电阻 $R_1$，标称值为 6.8kΩ，节点①和节点②分别连接测试总线 S 和 A，节点④和节点⑨设为保护点等。测试系统可据此逐条分析后依次进行测试，并与预值比较。测试过程可选择全速、低速和单步几种不同形式，并在单步方式下弹出相应类型的元件测试子窗口，展示具体测试过程。如在图 17-7 所示针对电容器测试过程的子窗口中，指出所加激励信号的正弦波形、电压幅值 5V 和频率 10kHz 等参数，以及作为中间测试变量的容抗 $X_c = 149\text{k}\Omega$ 和最终测试结果 $C = 107.2\text{pF}$。

PCB 路内测试不需要了解被测电路的工作原理，具有很强的通用性。基于 USB 构建的虚拟仪器充分发挥了通用计算机的计算功能和图示功能，简化了硬件设备，降低了开发成本。

### 7.2.8　通用可程控仪器的使用

使用可程控仪器作为测试设备时必须选用合适的接口，这里以美国泰克(Tektronix)公司的 TDS-1001B 示波器和 AFG-3021 任意波形发生器为例(见图 7-18)，说明在 VB 通用编程环境下使用 USB 接口开发应用程序操作仪器的实际方法。

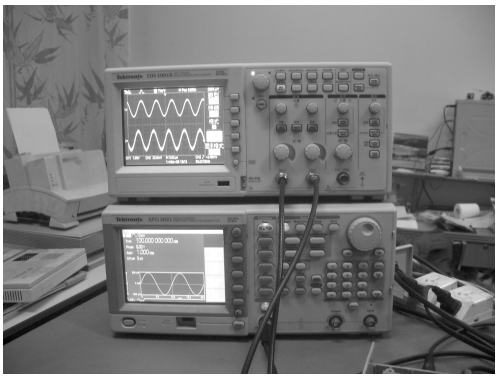


图 7-18　泰克公司生产的可程控仪器

仪器开发商提供的 TekVISA 软件包括测试仪器资源管理器和基于 VISA 标准的驱动程序。仪器资源管理器可以搜索接入计算机的各种资源，并显示每个仪器资源的描述字符串及其属性。驱动程序是以 VISA32.dll 命名的动态连接库文件。这样，应用程序借助 VISA32.bas 和

VppType. bas 两个文件的支持，就可以在 VB 通用编程环境下，通过调用 Visa 库函数来操作可程控仪器了。

如程序清单 7-1 所示，函数代码 GetFrqMag( )用于可程控仪器幅频特性测试的操作，作为识别阻容器件的基础。先将频率和幅值(参数变量 vFrq 和 vAmp)转换成信号发生器的程控命令发出，从而产生指定频率和幅值的正弦波信号。然后，通过对数字示波器发布程控命令，设置双通道方式分别测量被测电路的实际输入激励信号和输出响应信号，从而可以算出其幅频增益。在不同频率、幅值参数下重复调用函数 GetFrqMag( )，即可获取幅频响应曲线。

**程序清单 7-1 用于幅频特性测试的可程控仪器操作**

```
Const rsAfg As String="USB0::1689::832::C012378::0::INSTR"'AFG-3021 寻址表达字串
Const rsOsc As String="USB0::1689::866::C051196::0::INSTR"'TDS-1001B 寻址表达字串
Dim sesn As Long, vi_Osc As Long, vi_Afg As Long
Function GetFrqMag(ByVal vFrq As Variant, ByVal vAmp As Variant) As Single
Dim s$, rt&, d1#, d2#, n1%, n2%, tm$
    viOpenDefaultRM sesn: viOpen sesn, rs_Afg, 0, 0, vi_Afg '信号发生器设置
    s$="FUNC SIN; FREQ" & CStr(vFrq) & "HZ; VOLT" & CStr(vAmp /2) & "VPP; "
    s$=s$& "PHAS 0DEG; VOLT: OFFS 0": viWrite vi_Afg, s$, Len(s$), rt&
    s$="OUTP ON": viWrite vi_Afg, s$, Len(s$), rt&
    viClose vi_Afg: viClose sesn
    viOpenDefaultRM sesn: viOpen sesn, rs_Osc, 0, 0, vi_Osc '示波器设置
    s$="SEL: CH1 ON; CH2 ON; REFA ON; REFB ON": viWrite vi_Osc, s$, Len(s$), rt&
    s$="AUTOS EXEC": viWrite vi_Osc, s$, Len(s$), rt& 'Sleep 1000
    s$="*WAI": viWrite vi_Osc, s$, Len(s$), rt&
    s$="HOR: MAIN: SCAL" & CStr(0.2 /vFrq): viWrite vi_Osc, s$, Len(s$), rt&
    s$="ACQ: STOPA SEQ": viWrite vi_Osc, s$, Len(s$), rt&: Sleep 1000
    tm$=Chr(10)
    s$="MEASU: IMM: SOU CH1; TYPE PK2": viWrite vi_Osc, s$, Len(s$), rt&: Sleep 1000
    s$="MEASU: IMM: VAL?": viWrite vi_Osc, s$, Len(s$), rt&: Sleep 1000
    ReDim s$(40): viRead vi, s$, rCnt, rt&: s$=Left(s$, rt&)
    n1%=InStr(1, s$, " ")+1: n2%=InStr(n1%, s$, tm$)-n1%: d1#=Mid(s$, n1%, n2%)
    s$="MEASU: IMM: SOU CH2; TYPE PK2": viWrite vi_Osc, s$, Len(s$), rt&: Sleep 1000
    s$="MEASU: IMM: VAL?": viWrite vi_Osc, s$, Len(s$), rt&: Sleep 1000
    ReDim s$(40): viRead vi, s$, rCnt, rt&: s$=Left(s$, rt&)
    n1%=InStr(1, s$, " ")+1: n2%=InStr(n1%, s$, tm$)-n1%: d2#=Mid(s$, n1%, n2%)
    s$="ACQ: STOPA RUNST; STATE RUN": viWrite vi_Osc, s$, Len(s$), rt&
'恢复正常运行
    viClose vi_Osc: viClose sesn_Osc
    GetFrqMag=d2#/d1#
End
```

在 VISA 支持下使用 SCPI 命令编程，可以不受开发软件平台、总线接口和仪器类型的限制。尽管本节采用基于 USB 接口的可程控仪器，却完全可以利用本书先前介绍的有关 GPIB、

VXI 仪器的编程操作经验，而且也自然可在今后其他系统中运用这些经验。唯一需要注意的是：必须首先建立每个仪器资源的描述字串，通常可通过仪器面板的操作直接得到，也可通过设备厂家提供的资源管理器搜寻，还可根据有关资料进行分析推断。

## 7.3　PCB 模件的功能测试

与 PCB 功能模件的在线测试不同，其功能测试通常并不采用通用的 ATE 设备。这是因为功能测试需要结合被测功能模件本身的工作原理，严格执行具体产品的行业规定，同时也需要从用户使用的角度考虑测试流程，并尊重其心理需求。这样，测试过程更强调最终功能的实现，这种有针对性的测试工作往往效率更高。

### 7.3.1　测试设备的设计思想

这里介绍的 PCB 功能自动测试系统中采用 USB 虚拟仪器技术，尽可能地通过软件和通用计算机固有的硬件资源来实现传统的测试系统功能，而测试系统中的硬件设备只是用来完成那些通用计算机所不具备的功能。

PCB 功能测试系统如图 7-19 所示，包括上位机 PC 系统、PCB 功能测试仪和连接被测 PCB(UUT)的针床结构。上位机系统由一台通用的 PC 和数据库等相关软件构成，它是整个功能测试系统的控制中心。PC 中的软件通过 USB 总线统筹控制着整个系统的测试过程(包括激励的施加、信号的采集、故障的分析等)，同时它还提供友好的人机交互界面，数据库用来记录用户录入的测试条件和测试信息，保存测试结果。

PCB 功能测试仪是整个测试系统中需要针对功能测试所需开发的主要硬件设备，包括由微控制器和 USB 接口构成的核心控制器、电源、模拟通道模块、数字测试模块和多路开关模块，用来实现测试系统中激励信号的输出功能和响应信号的采集功能。核心控制器一方面通过 USB 接口与系统上位机进行通信，接收上位机的控制命令、激励数据，并回传系统有关的工作状态和测得的响应数据；另一方面，通过 $I^2C$ 总线控制其下层的数字测试模块、多路开关模块。此外，由于 C8051F007 内部集成了 4 路 12 位的 A/D 转换器和 2 路 12 位的 D/A 转换器，可以产生模拟激励信号，也可以采集模拟响应信号。模拟通道模块用于信号调理，它将 C8051F007 的 4 路 A/D 通道扩充成 18 路 A/D 通道以适应多点测试的需要。数字测试模块用来完成系统中有关的数字测试，该数字模块实现 48 路通用的双向数字测试通道和一组串行测试通道。多路开关模块有 20 路开关通道，主要作用是施加各种测试条件，也用于施加电源或将其他系统外部的信号施加到被测对象上，从而建立相应的测试环境。针床结构通过针床上的触针和连线将被测 PCB 上各个测点通过接口插座引出，并通过扁平电缆与 PCB 功能测试仪连接起来。

系统提供了两种扩充方式。一是开发相应的测试模块，通过 $I^2C$ 总线嵌入到功能测试仪内部。二是选用基于 USB 总线的测试设备，通过 USB 总线将其融入系统。这种外部测试设备也可以不支持 USB 接口，如某些测试场合中需要的大功率电源，可以通过针床结构或多路通道开关将其直接接入被测对象。

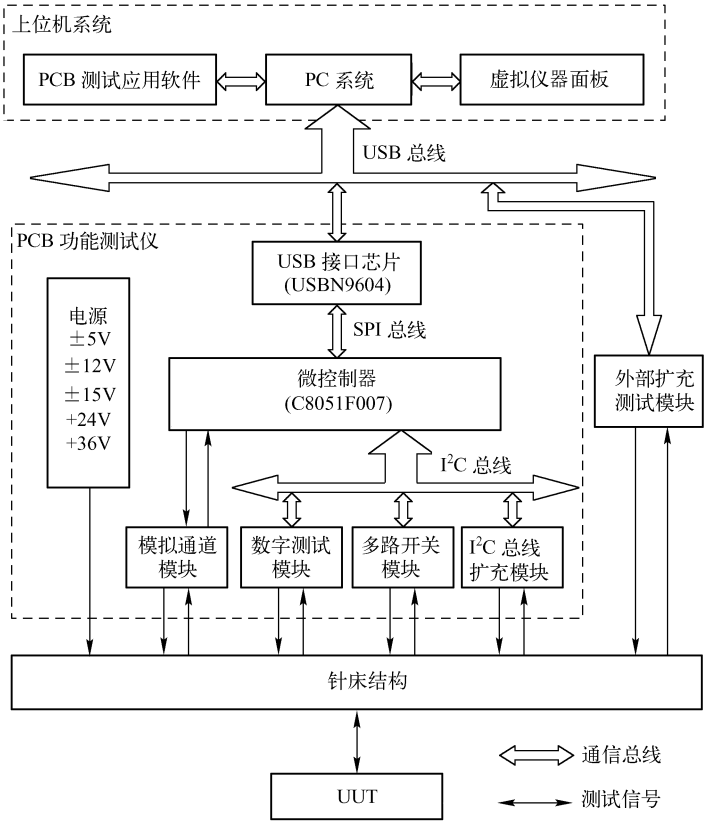



图 7-19 测试系统结构框图

鉴于模拟电路功能测试所用的一些具体测试设备与前面路内测试的内容有些重叠，下面只重点介绍数字电路的功能测试。

### 7.3.2 面向微处理器的总线测试

数字信号的信息量大、工作频率高，常伴有多路同步的要求，也包括某些偶然出现的单次信息。数字系统测试的基本要求是跟踪和分析状态数据流，监测电路的逻辑电平和时序状态并加以分析诊断。

数字测试系统包括数字测试模块、逻辑分析仪、特征分析仪和边界扫描测试仪。数字测试模块用来完成数字激励信号的施加和数字响应信号的采集；逻辑分析仪能够对特定激励信号和相应的响应信号进行时序和状态的分析；边界扫描仪主要通过 JTAG 接口实现，将在本章后面详细讨论；特征分析可由软件实现，通过获取数据特征字判断系统工作状态。基于虚拟仪器的设计思想，面板显示和操作、数据处理和计算、结果分析和数据保存都由 PC 主机完成；数字测试模块能够提供 6 组 48 路并行的双向输入/输出测试通道，每个通道既可作为激励输入通道，也可作为响应采集通道。在微控制器的控制下，一方面通过 USB 总线接收上层的控制命令和激励数据、回传采集的响应数据和数字测试模块有关的工作状态；另一方面通过复杂可编程逻辑器件 EPM7128S84 内部的控制逻辑配合双向总线收发器 74FCT543 实现数

字激励信号输出和数字响应信号采集的基础功能。

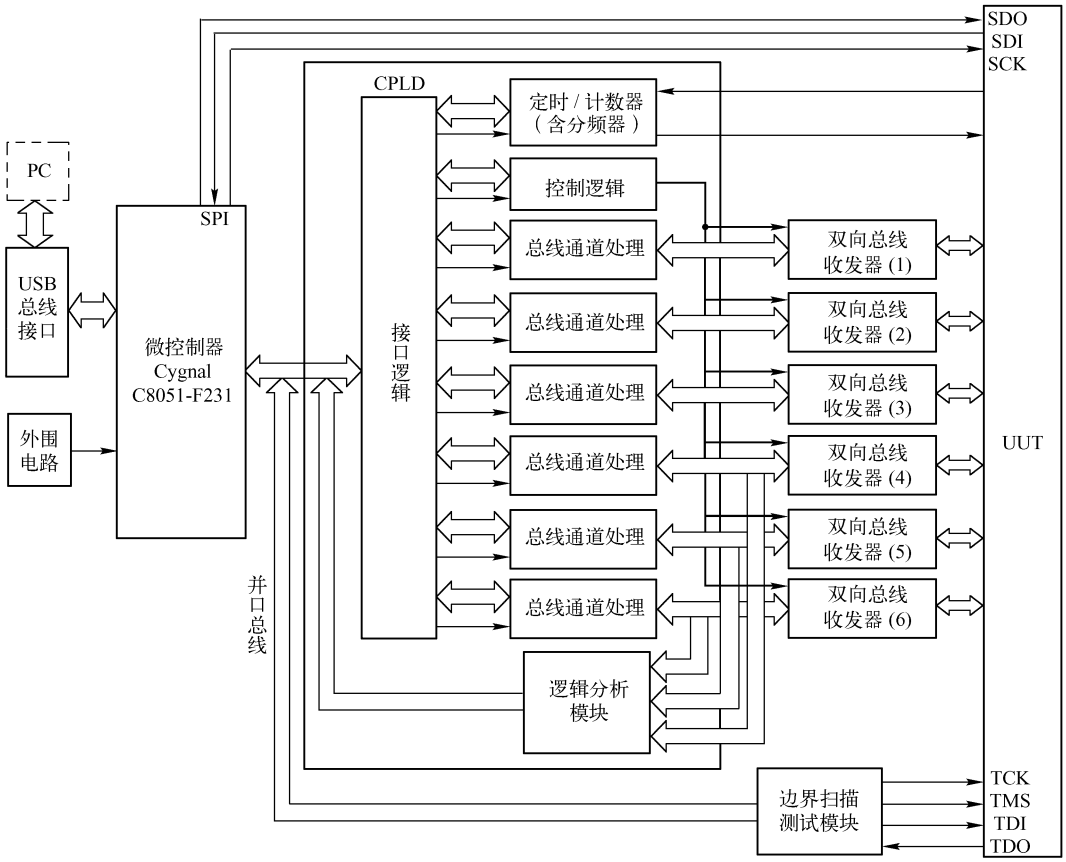



图 7-20　数字测试系统的结构示意图

### 7.3.3　逻辑分析功能的实现方法

逻辑分析仪的核心作用是要严格地按照指定的工作时钟节拍，将数字电路中多路被测通道的时序信号依次写入一个多通道的数据存储器，然后在需要时依次从中读出显示。

写入数据存储器的过程必须通过高速的硬件电路来完成，从而确保被测电路高速的时序状态能被精确地跟踪和记录。数据存储的空间并非越深越好，因为采样过长不仅造成电路结构复杂、设备成本升高、通信负担加重，而且显示和分析时也会增加从浩大存储数据中找到所需内容的难度。为此，在数据采集阶段就进行选择时序记录区段的“触发”功能就成为逻辑分析仪中的一个关键问题。实际工作中触发信号可能来自被测信号中某种特定的数据状态，也可能来自多路地址或其他控制信号，还可能是它们的组合。通过触发获取的数据块通常在时区上需要跨越触发点，即最终提供显示和分析的被采数据应当包括触发前的若干数据和触发后的若干数据。

逻辑分析仪的基本工作原理及基本电路结构如图 7-21 所示。这里，先入先出(FIFO)的数据存储器是一种理想的数据存储器件，只要给定读/写时钟便能依次存写或读取数据，从而省去了一般存储器需要管理多路地址总线的麻烦。且 FIFO 能根据当前存储内容所占空间

的状态，给出满、半满和空信号，从而为建立控制流程带来很大方便。

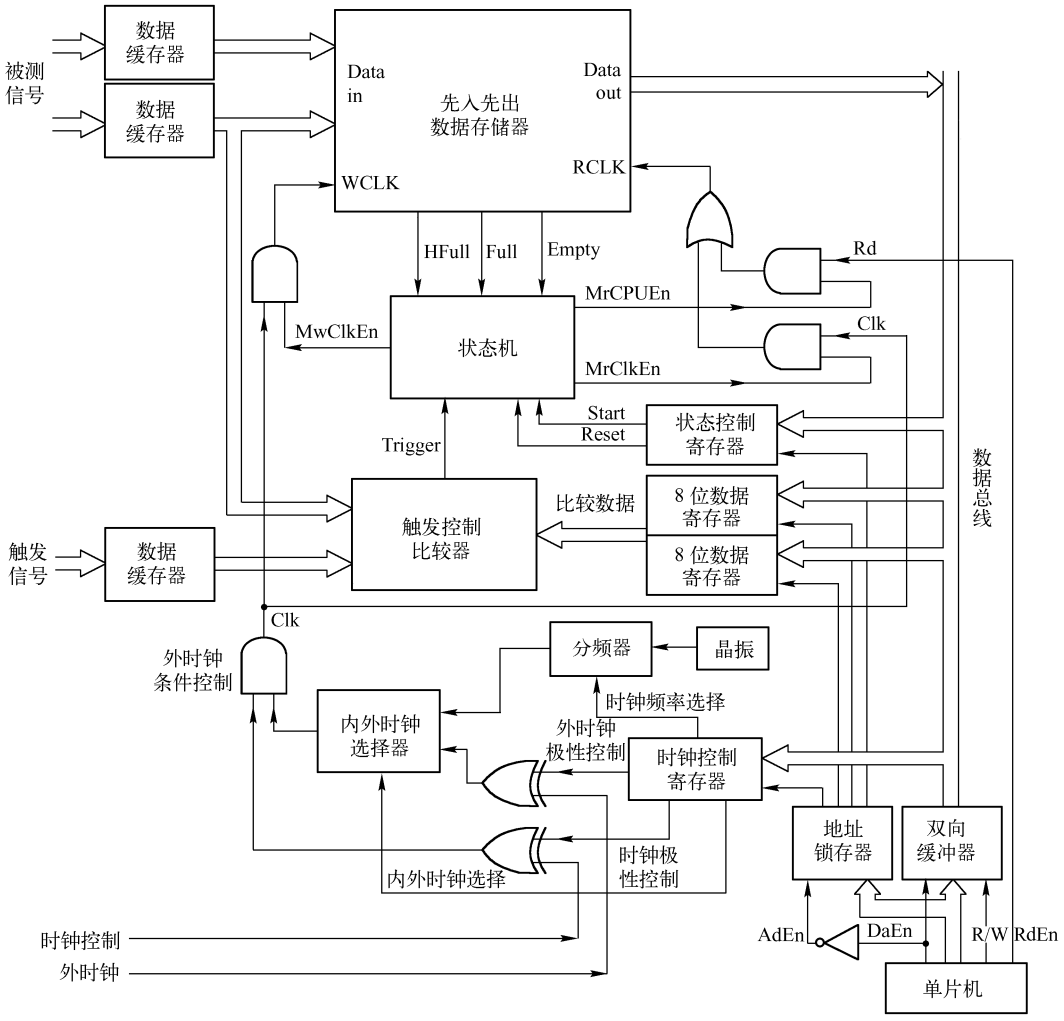



图 7-21　逻辑分析仪原理框图

逻辑分析仪的整个工作过程分为 5 个不同的工作阶段，可用一个工作状态机来实现这一任务。如图 7-22 所示，状态机的每个不同状态对数据存储器发出不同的选通控制信号，状态机的状态变迁条件则源于逻辑分析仪电路的工作信号。一旦系统上电，或单片机发出复位信号 Reset，系统状态机就进入空闲状态，逻辑分析仪等待工作。此时数据存储器的读/写选通信号均被清除(即 MwClkEn = 0，MrClkEn = 0，MrCpuEn = 0)。当单片机发出启动信号 Start 后，系统状态机变迁到预采数状态，逻辑分析仪开始工作。此刻数据存储器的写选通有效(MwClkEn = 1)，从而使被测数据通道的时序状态被逻辑分析仪的时钟脉冲逐次打入数据存储器中。同时，计数器进行同步计数，直至半满标志信号 HFull 产生，状态机进入触发前采数状态。此后数据存储器的时钟读、写选通均被打开(MwClkEn = 1，MrClkEn = 1)，于是出现依次写入一个新数据的同时依次读空一个老数据，使数据存储器始终保持一个半满左右的有效存储空间。当触发信号 Trigger 到来时，状态机进入触发后采数状态。此刻 FIFO 存储器在

继续打开写选通信号的同时，禁止读出数据（MwClkEn = 1，MrClkEn = 0），于是数据只进不出，直到数据存储器的满标志信号 Full 产生，状态机变迁到读数状态。这时数据存储器中已存储的数据应包括触发前采集的半满长度数据信号和触发后的另一半数据。在读数状态，为单片机读取数据的选通信号有效（MrCPUEn = 1），而读数脉冲则来自单片机本身，单片机可以从 FIFO 中从容地逐个读取数据。当数据存储器的数据被全部读空后，一个空标志信号 Empty 将状态机拉回到空闲状态，一个完整的工作周期结束。注意，上述工作流程一经单片机启动，系统就完全在硬件电路的直接作用下独立完成高速的数字数据采集任务，然后再等待单片机读数。

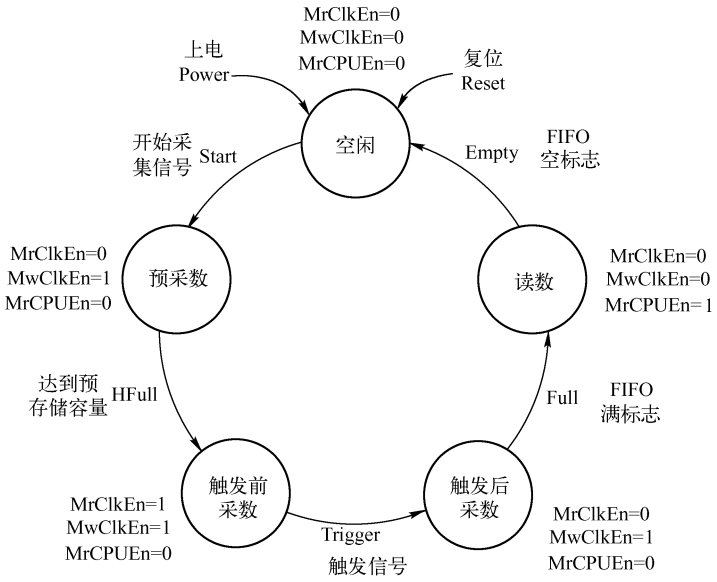



图 7-22　状态机状态转换图

单片机是整个设备的核心控制器。它通过一组数据总线和 3 个控制信号操作设备的工作过程，并能观察设备的工作状态。AdEn/DaEn 是地址/数据选通信号，用以区分单片机数据线上操作的当前信号是地址还是数据。当 DaEn 有效时，R/W 信号控制数据总线上的数据方向（读出或写入）。Rd 信号在设备进入读数状态后有效，将被测数据读入单片机。

触发控制比较器将一组被测信号和触发信号同单片机设定的触发条件进行比较，当比较结果一致时，发出触发动作信号 Trigger。

设备支持内、外两种工作时钟。内时钟由晶振通过分频电路产生，具体工作频率可由单片机设定分频系数来实现，外时钟信号可直接接入逻辑分析仪设备，并由单片机控制其有效电平的极性。系统还允许接入某种针对时钟工作条件的选择信号，以实现对工作时钟进行附加条件的限制，而这种附加的时钟控制条件也可选择其有效电平的极性。

逻辑分析仪功能模块部分的核心部件是一个可编程逻辑器件 CPLD，主要工作电路都在 CPLD 中实现。CPLD 的高速特性确保其在很高的工作频率下能对系统状态的变化做出快速反应，而 CPLD 的可编程特性则使其容易实现与单片机之间的接口以及各种通信规约。这里选用的 EPM7128S84 是 ALTERA 公司的 CMOS 产品，特别适于高速、复杂的组合逻辑和时序电路的实现，其引脚到引脚间的延迟最快可小于 5ns，计数频率可达 175MHz，提供 64 个可编

程的通用 I/O 口引脚，从而使这样一个可编程逻辑器件能够实现传统设计中众多小规模集成芯片所完成的功能。

高速数据缓存器(74FCT541T)用来建立 CPLD 与输入信号之间的通道，因为 CPLD 是电路中最主要也是最娇气的部分，所以对其进行适当的保护很有必要。数据存储器(FIFO)选用 CY7C44X1 系列，它的最高存储速率可达 100MHz，FIFO 存储深度可达双组 8KB，成为高速数据采集通道和单片机常速通信传输之间的理想缓存设备。

### 7.3.4 特征分析功能及其软件实现

特征分析的基本思路是在总线测试过程中，将被测电路的某组数据响应序列压缩处理成一组特征状态字，然后根据其与预期特征状态字的异同来进行故障分析或诊断。被测电路预期的标准特征字可以预先通过完好的标准电路实测得到，也可以通过对电路功能进行仿真试验得到。图 7-23 体现了一种 16 位移位寄存器的组成的特征分析仪工作原理，可以直接用来构建实际的特征分析系统。

Data 数据进入特征分析仪后，要经过 ((B6 ⊕ B8)⊕(B11 ⊕ B15))⊕ Data 的异或操作后移入 16 位移位寄存器。这里 B6、B8、B11、B15 是移位寄存器第 6、8、11 和 15 位的输出量，选择它们作为反馈变量对输入数据进行特征处理，就构成了特征分析仪工作的核心电路。同时，16 位移位寄存器的各输出位按 8421 编码组成相应的特征字，并通过数码管显示出来。

为了控制特征分析仪工作，C1、C2、C3 分别控制了特征分析仪的 Start 信号、Stop 信号及 Clock 信号的极性。当其为 0 时信号高电平有效；为 1 时信号低电平有效。信号$\overline{R}$可将时钟锁住，使特征分析仪工作暂停；信号$\overline{L}$可同时将移位寄存器和数码管清零，使特征分析仪恢复初始工作状态。

特征分析仪对总线上的数据序列进行特定的移位反馈记录，通过选择适当的反馈方式得到数据的特征字。通常使用的 16 位寄存器特征分析仪的特征字由两个字节组成，因此对每一路被测信号来说表达的是 $2^{16}$种时序状态的组合。然而对采样深度为 1024 的数据应有 $1\times2\times3\cdots\times2^{10}$种时序状态的组合。显然，直接用特征字描述测试序列时会出现不同时序信号产生相同特征字的可能。由于正确地设计特征分析仪的反馈电路能够大大减小出现这种情况的概率，因此需要合理选择特征分析仪反馈方式，实际推导过程是一项专门的基础研究工作，这里将直接使用其研究结果。为了提高特征分析仪的故障侦出率，可以通过增加线性反馈移位寄存器的位数来实现，但两个字节(4 个字符)的特征字比较简洁、便于观测和记忆，增加字符个数反而会影响检测者对特征字的习惯反应。

在总线测试中实际运用特征分析时往往需要多路通道，因此最终使用的移位寄存器单元往往会很多。而在虚拟仪器中，长达几千个字节的完整时序信号已经上传到主机。那么，通过软件算法同样可以实现图 7-23 的功能，而且显然是一种事半功倍的方法。

如图 7-24 所示，待分析的数据保存在数组变量 Data 中，若数组元素为长整型，则表明可同时在 32 通道分别进行 b=b ⊕((B6 ⊕ B8)⊕(B11 ⊕ B15)) 的逐位运算后移入 16 位移位寄存器 B 中，移位寄存器 B 中的 B6、B8、B11、B15 位需要反馈回运算器中参与运算。最终通过计算机面板显示程序建立的虚拟仪器数码管分别显示 16 位移位寄存器 B 中所存的 B15~B12、B11~B8 位、B7~B4 和 B3~B0 各位，作为被分析数据序列的特征值。

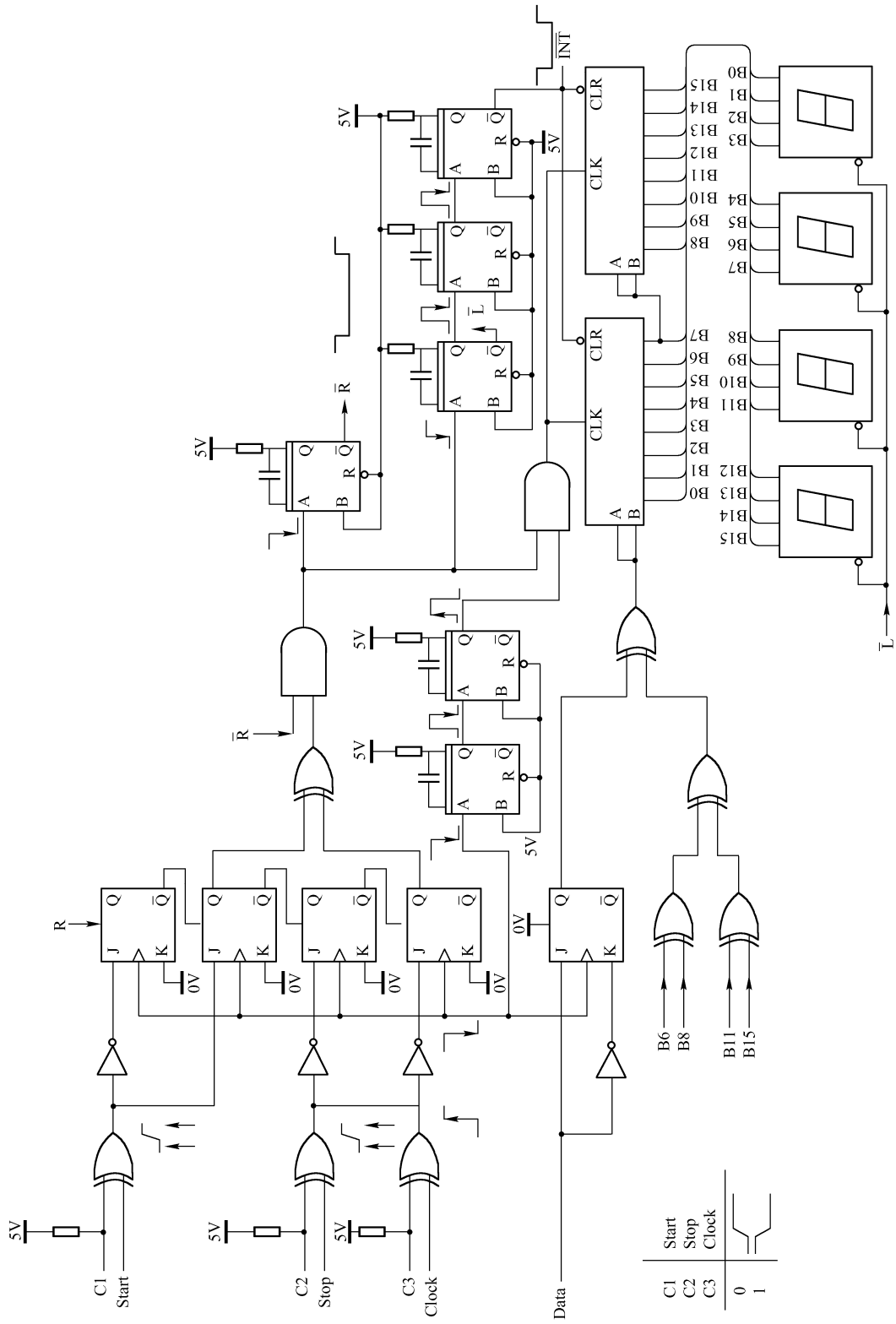



图 7-23　特征分析仪原理

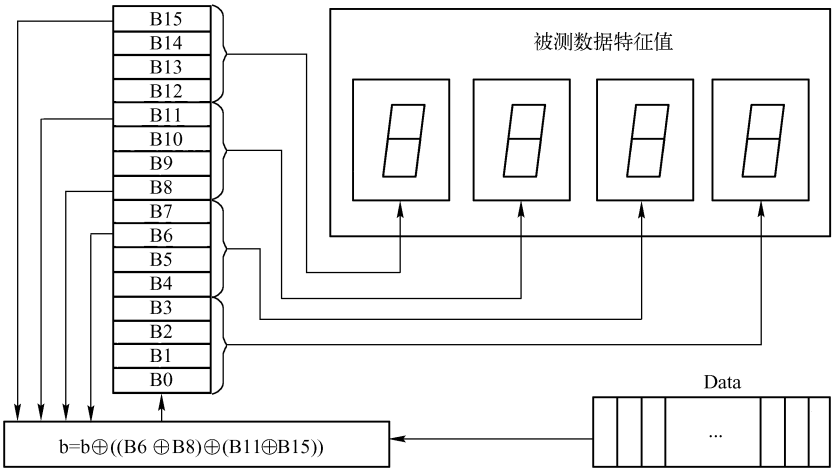



图 7-24 特征分析仪工作流程原理图

实际特征分析的核心程序如程序清单 7-2 所示，其逻辑关系简单，可用 VB 语言直接实现。这里，数据变量和特征变量均为 byte 类型数组，因此表示 8 个通道的处理过程。

**程序清单 7-2 软件实现的 8 通道特征分析核心程序**

```
Public Sub signature (n As Variant, Data() As Byte, b() As Byte)
    For j = 0 To n
        For i = 0 To 15
            b(i + 1) = b(i)
        Next i
        b(0) = Data(j) Xor ( ( b(6) Xor b(8) ) Xor ( b(11) Xor b(15) ) )
    Next j
End Sub
```

### 7.3.5 功能测试系统的主操作界面

功能测试系统的主操作界面如图 7-25 所示。系统首先从前期建立的测试数据库表中提取测试信息到主控操作界面左部的测试信息表中，然后由测试程序逐条解释测试信息表中的测试项目和参数，执行相应的测试操作，并将有关的测试结果记录在测试信息表的相应栏目。同时，有关测试过程也显示在操作界面右部的各个虚拟仪器面板中。

测试过程完成后，系统将所有的故障信息和排除故障的参考措施打印成清单，与有故障的被测 PCB 一同交付技工处理。执行过程也可采用单步工作方式，以便反复进行测试过程中的某一步骤，通过主操作界面右部的数字表、信号发生器、示波器、逻辑分析仪、特征分析仪等虚拟仪器窗口仔细观察被测电路的细节。

测试数据库表如表 7-2 所示，包括序号、测试项名称、测点位置、类型、通道、实值、预测值 1、预值 2 等项。序号是测试条目在整个测试数据库表中的顺序位置，测试项名称表达了测试项目的内容，位置是施加激励信号或提取响应信号的测点标号。类型指示所采用的测试设备，如类型值为“K”时表明是多路开关操作，即根据实值项的值是“ON”还是“OFF”，将通道项所对应的开关闭合或断开；类型值为“AD”时表明是采集模拟响应信号的操作，即从通道值所对应的模拟通道处采集响应信号。如果所采集的实际信号值在预值 1 和预值 2 范围

内，则表明该条测试通过。

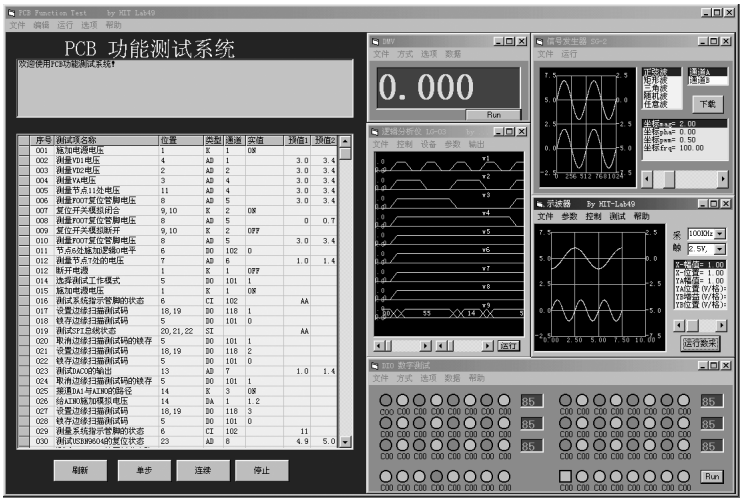



图 7-25　主机软件操作界面

**表 7-2　被测 PCB 的测试信息**

| 序号 | 测试项名称 | 位置 | 类型 | 通道 | 实值 | 预值 1 | 预值 2 |
|---|---|---|---|---|---|---|---|
| 001 | 选择测试工作模式 | 5 | DO | 101 | 1 | | |
| 002 | 施加电源电压 | 1 | K | 1 | ON | | |
| 003 | 接通 AD15 与节点 4 间的路径 | 4 | K | 2 | ON | | |
| 004 | 测量 VD1 电压 | 4 | AD | 15 | 3. 65 | 3. 0 | 4. 0 |
| 005 | 断开 AD15 与节点 4 间的路径 | 4 | K | 2 | OFF | | |
| 006 | 接通 AD15 与节点 2 间的路径 | 2 | K | 3 | ON | | |
| 007 | 测量 VD2 电压 | 2 | AD | 15 | 3. 24 | 3. 0 | 3. 4 |
| 008 | 断开 AD15 与节点 2 间的路径 | 2 | K | 3 | OFF | | |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |

测试数据库表的建立反映了对该被测对象的测试策略，通常第一步总是测试电源部分，在给整个被测板施加外部电源后，就应首先查看稳压电路中各个输出节点的电压值是否正常。第二步则可能测试被测板的电源指示功能和状态指示功能。第三步为被测板上微处理器的外围复位电路功能。前三步的成功表明被测电路的基本运行条件已经建立起来，在此基础上方能针对被测板本身的具体功能进行逐项测试。

基于虚拟仪器的 PCB 功能测试系统具有开发快、成本低的特点，常常可以根据不同被测 PCB 功能模件的具体要求开发有针对性的测试设备。这样，测试的效率往往较高，提供的设备数量也可能较多，容易满足生产的实际需要。图 7-26 为用于 PCB 功能测试的某虚拟仪器系统的实例照片。



图 7-26 基于虚拟仪器的 PCB 功能测试系统

## 7.4 边界扫描测试与 IEEE1149 规范

鉴于大规模集成电路(VLSI)的迅速发展和 PCB 表面贴装工艺(SMT)的广泛使用，印制电路板(PCB)功能模件的密度越来越高，使其在测试中适于针床和夹具的引脚越来越少，给电路板在线检测和故障排除带来很大困难。边界扫描测试方法在芯片内部构建测试电路，相当于为其配置了施加激励和观测响应的虚拟探头，其测试设备简单、测试功能强，已形成的 IEEE 1149.1 边界扫描标准开辟了数字电路测试的新途径，而规范 IEEE 1149.4 则进一步为模拟电路的边界扫描测试展示出诱人的前景。

### 7.4.1 边界扫描测试的基本原理和基本单元结构

边界扫描测试的核心思想是在数字电路芯片的每个 I/O 引脚和芯片内核逻辑功能电路之间插入一个边界扫描单元(Boundary Scan Cell，BSC)，同时在芯片中附加一些测试控制逻辑来管理这些边界扫描单元。如图 7-27 所示，边界扫描单元 BSC 在器件正常运行状态下相当于直接横向连通，不会影响器件的正常工作；而器件在测试状态下则通过 BSC 竖向连接构成的串行移位扫描链，实现从器件接口移位输入激励信号、移位输出监测响应信号。

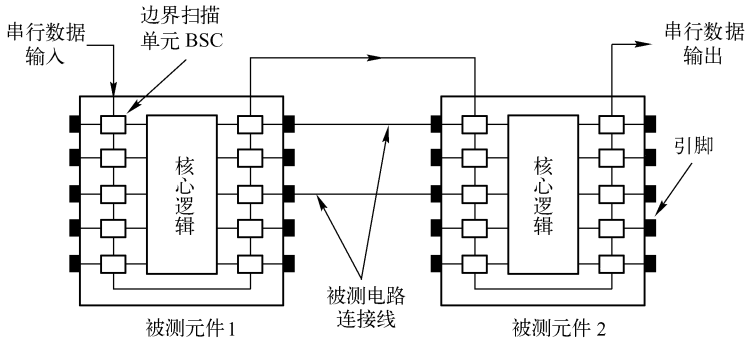



图 7-27 边界扫描测试的原理示意图

BSC 的内部结构如图 7-28 所示，其标准的数据输入端 DI（Data In）和输出端 DO（Data Output）可以接到芯片输入引脚与内部核心逻辑之间，也可接到内部核心逻辑与芯片输出引脚之间。

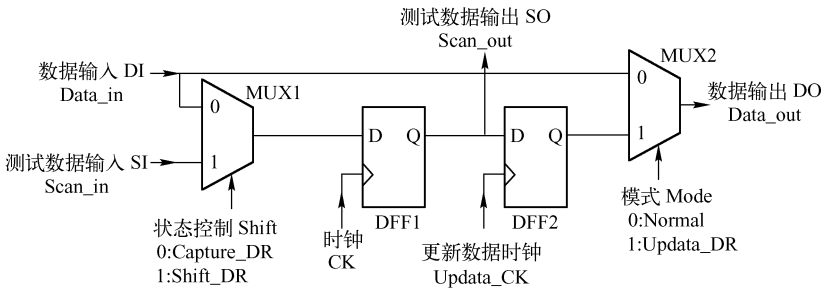



图 7-28　边界扫描单元 BSC 的内部结构

作为前者的连接方式，为了检测连接 DI 的芯片输入引脚状态，应让操作 MUX1 的控制信号 Shift = 0（Capture_DR 模式），然后在时钟信号的配合下将 DI 信号捕捉到 DFF1 中，再通过扫描链的串行移位功能将测得数据从测试数据输出端 SO 逐次移出；作为后者的连接方式，为了将指定的测试向量加载到连接 DO 的芯片输出引脚，可使 MUX1 的控制信号 Shift = 1（Shift_DR 模式），并通过时钟将其从测试数据输入端 SI 移入 DFF1 暂存，再由更新时钟信号将该测试向量移入 DFF2，并通过置 MUX2 的控制信号 Mode = 1（Update_DR 方式），将数据从 DO 发出。不论何种连接方式，倘若设置 MUX2 的控制信号 Mode = 0（Normal 模式），则由于芯片引脚与芯片电路直接相通，均不影响芯片电路的正常工作。

### 7.4.2　边界扫描测试控制器及其访问端口

根据 IEEE 1149.1 的规定，JTAG 控制器的电路结构如图 7-29 所示，其组成主体包括测试访问端口（Test Access Port，TAP）、指令寄存器（Instruction Register，IR）和若干测试数据寄存器（SR，Register）。

测试访问端口 TAP 采用四线方式，包括接口信号线 TDI、TDO、TCK 和 TMS。如果希望配置复位信号线，则端口扩展为包括接口信号线 TRST 的五线接口方式。这里，接口信号线 TDI（Test Data In）和 TDO（Test Data Out）主要用于测试数据和指令的输入和输出，在测试时钟（Test Reset，TCK）的配合下实现数据寄存器或 SR 指令寄存器 IR 的串行移位传输。测试模式选择（Test-Mode Select，TMS）信号在测试时钟 TCK 的配合下，产生特定的控制信号来控制整个测试过程的状态，复位线 TRST 则可强制初始化系统的工作状态。

IEEE 1149 规定了 3 种最主要的测试数据寄存器，包括边界扫描寄存器、器件标识寄存器和旁路寄存器。具体是何种数据寄存器被接入当前连接 TDI 和 TDO 的串行移位数据链取决于特定指令的执行，数据寄存器的一些具体操作动作也要依赖于指令的执行。与数据的串行移位传输方式相似，指令也是通过 TDI 线在 TCK 的配合下串行移位传入芯片内部的指令寄存器后译码执行的。

边界扫描寄存器 BSR 由置于芯片引脚与内核功能电路之间的一组边界扫描单元 BSC 构成，它们首尾相连，且首端接 TDI 线，末端接 TDO 线，在测试时钟 TCK 的作用下，形成移位

寄存器链。从而将 TDI 串行输入的数据加载到各 BSC，将各 BSC 捕捉的状态通过 TDO 串行输出。

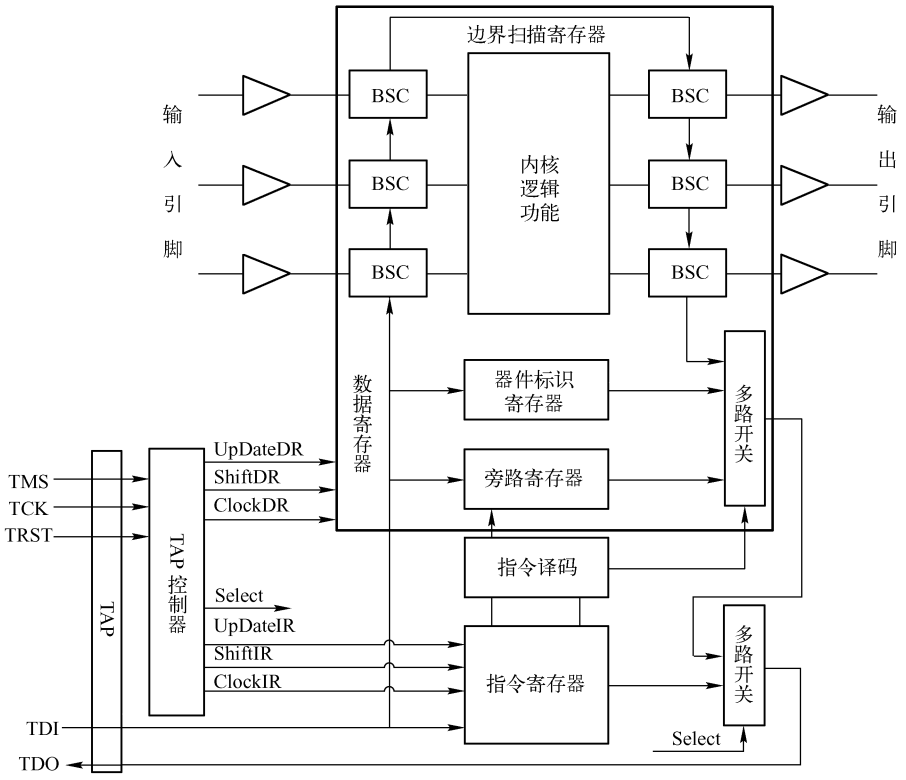



图 7-29 边界扫描器件的内部结构

器件识别寄存器 IR(ID-Code Register)有 32 位，如图 7-30 所示，其中 31~28 位是版本号，27~12 位是器件序列号，11~1 位是厂家标识，第 0 位恒为 1。通过分析 TDO 移位输出的结果，可以检查是否已将正确的器件安装在 PCB 的正确位置，也可获取这些元器件的厂家信息。

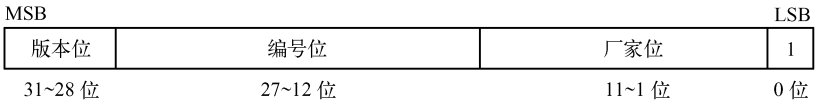



图 7-30 器件标识寄存器的编码格式

旁路寄存器(Bypass Register,BR)只有 1 位，它提供了一条从 TDI 到 TDO 之间的最短通道，用以将不参加串行边界扫描的器件旁路掉，以减少不必要的扫描时间。

### 7.4.3 TAP 控制器的状态控制

IEEE 1149.1 规定：JTAG 接口的 TMS 和 TDI 信号在 TCK 信号的上升沿被采样，TDO 信号在 TCK 信号的下降沿产生变化。图 7-31 为 TAP 控制器有限状态机的状态变迁图，图中右部是指令寄存器的工作分支，左部是数据寄存器的工作分支。TAP 控制器的状态由 TMS 控

制，状态变迁的条件“0”和“1”均指信号 TMS 的逻辑电平，依据这些变迁条件，可使 TAP 进入所希望的任何状态。

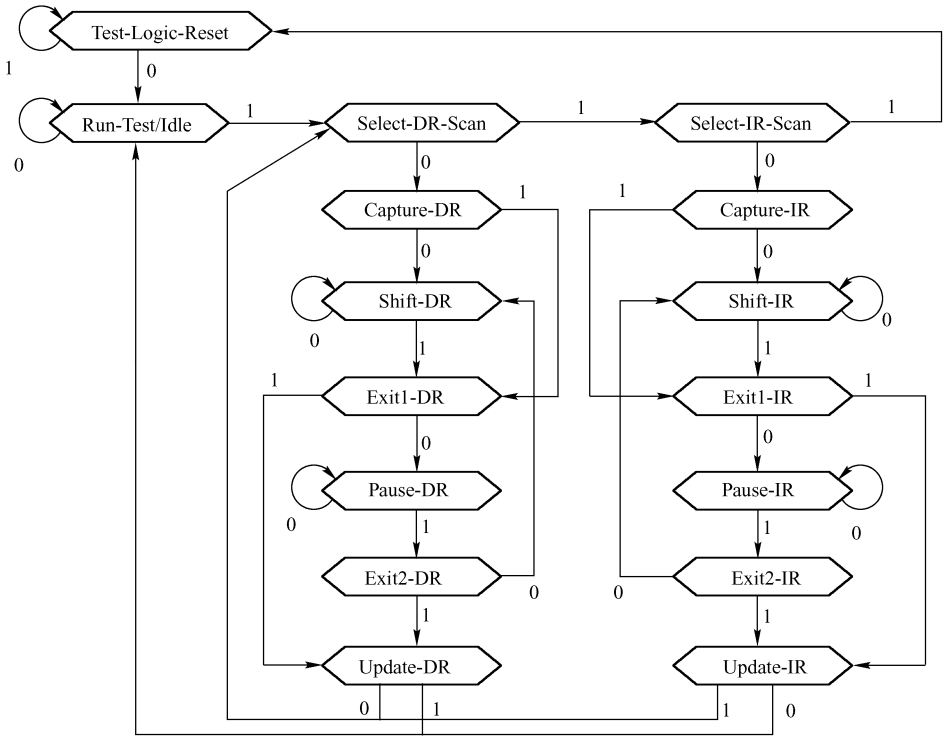



图 7-31　TAP 控制器状态机

TMS 保持高电平时，任何状态下均会在 5 个 TCK 时钟周期内回到复位状态(Test-Logic-Reset,TLR)。芯片加电后设定 TMS 为高电平，这样 5 个 TCK 后 TAP 控制器就进入 TLR 状态，产生复位作用，使测试电路不影响芯片本身的工作。测试时，TAP 控制器在 TMS 和 TCK 的作用下跳出 TLR 状态，并可通过 TMS=01 或 TMS=011 分别进入选择数据寄存器扫描状态(S-DR-Scan)或选择指令寄存器扫描(S-IR-Scan)状态。然后，根据图中所示 TMS 的设置要求，可以依次进入图中其他状态。对于数据寄存器的操作，可在 Capture-DR 状态把监测数据并行捕捉到 BSR 后，再在 Shift-DR 状态从 TDO 移位传出。也可同时在 Shift-DR 状态从 TDI 移位传入激励数据，再进入 Update-DR 状态将此数据加载输出。对指令寄存器操作的情况相似，其主要工作状态是在 Shift-IR 状态传送指令，在 Update-IR 状态加载指令。可通过 TMS=11100 从 Idel 状态进入 Shift-IR 状态，再通过 TMS=11 从 Shift-IR 状态进入 Update-IR 状态。

### 7.4.4　边界扫描测试的操作指令及工作方式

尽管 TAP 状态机为边界扫描过程规定了明确的节拍，但实际动作的执行仍然需要依据相应的指令来完成。IEEE 1149.1 标准中定义了一些边界扫描测试必备的指令，而另一些指令则是可选的，同时也允许定义其他专用的指令来扩展测试的功能，这里列出最常使用的几种标准指令，并通过图 7-32 展示其工作通道。

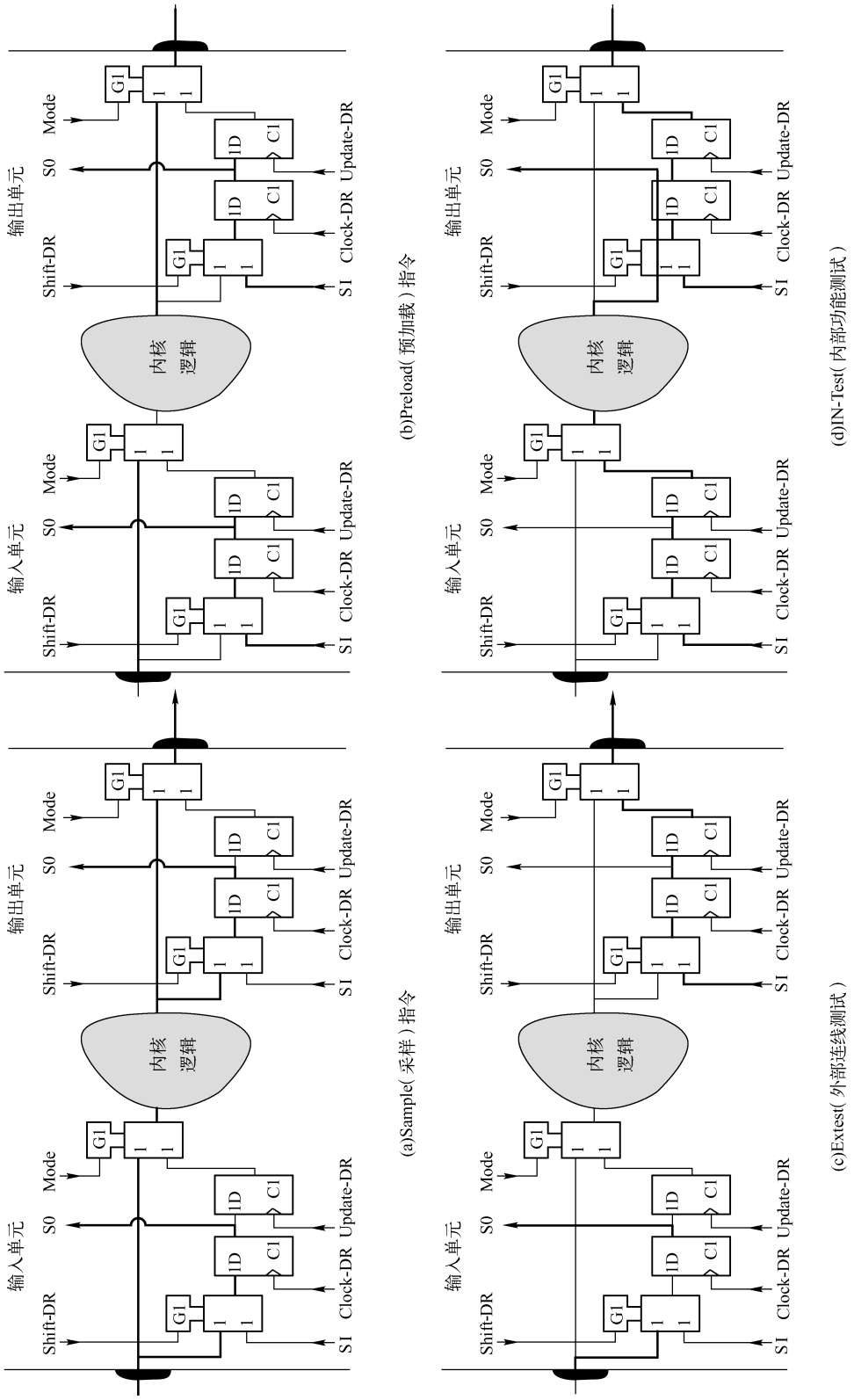



图 7-32 典型边界扫描指令的工作方式

**IDCode**：将 ID 寄存器连接到 TDI 与 TDO 之间，为读取器件标识提供必要的条件。

**Bypass**：将旁路寄存器连接到 TDI 与 TDO 之间，为数据和指令的快速通过提供途径。

**Sample/Preload**：将边界扫描寄存器连接到 TDI 与 TDO 之间，允许在 Capture-DR 状态下从 BSC 的 DI 口读数，在 Shift-DR 状态下从 BSC 的 SI 下载预设数据，为采样信号和下载数据提供必要的条件。

**Extest**：外部连线测试，允许在 Updata-DR 状态下向芯片输出引脚所连 BSC 的 DO 口输出数据，在 Capture-DR 状态下从与芯片输入引脚相连 BSC 的 DI 口读取数据，从而为测试 PCB 内芯片引脚之间的互连状态提供必要的通道条件。

**InTest**：内部功能测试，允许在 Updata-DR 状态下向与芯片内核电路输入端所连 BSC 的 DO 口输出数据，在 Capture-DR 状态下从与芯片内核电路输出端相连 BSC 的 DI 口读取数据，从而为测试芯片内部逻辑功能提供必要的通道条件。

下面简要介绍边界扫描测试的几种主要工作方式。

### 1. 外部连线的测试

断路和短路是 PCB 模件连线经常出现的故障，可利用连线一端连接的芯片输出引脚加入激励信号，从连线的另一端连接的芯片输入引脚测取响应来判断连线的故障。将加入各激励测点的信号数据依行排列，多次测试则形成多列的测试矩阵 $\boldsymbol{T}$；同样，将各测点测得的状态数据依行排列，多次测试形成响应矩阵 $\boldsymbol{R}$。这样，一种常用的测试策略是将测试激励矩阵选做如下形式(其中 $n$ 为激励测点的数目)。

$$\boldsymbol{T}_{n\times(n+1)}=\begin{pmatrix}0 & 0 & \cdots & 0 & 1\\ 0 & 0 & \cdots & 1 & 1\\ \vdots & \vdots & \iddots & \vdots & \vdots\\ 0 & 1 & \cdots & 1 & 1\end{pmatrix}$$

若以图 7-33 所示被测电路为例，则有

$$\boldsymbol{T}=\begin{pmatrix}0 & 0 & 0 & 1\\ 0 & 0 & 1 & 1\\ 0 & 1 & 1 & 1\end{pmatrix}$$

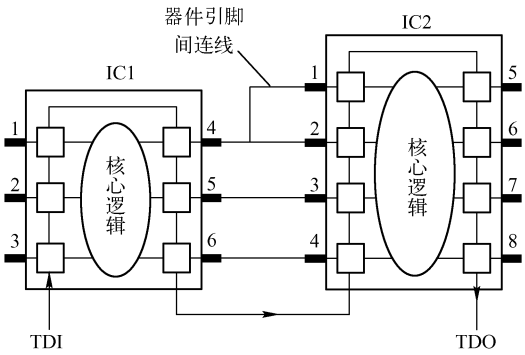



图 7-33　外部测试方式

分析相应响应矩阵的数据就可以判断连接的故障。考虑可能发生的下列几种情况：

$$\boldsymbol{R}_1=\begin{pmatrix}0&0&0&1\\0&0&0&1\\0&0&1&1\\0&1&1&1\end{pmatrix}\qquad \boldsymbol{R}_2=\begin{pmatrix}0&0&0&1\\0&0&0&1\\1&1&1&1\\0&1&1&1\end{pmatrix}\qquad \boldsymbol{R}_3=\begin{pmatrix}0&0&0&1\\0&0&1&1\\0&0&1&1\\0&1&1&1\end{pmatrix}$$

显然，$\boldsymbol{R}_1$ 的各行完全响应了 $\boldsymbol{T}$ 中对应连接行的激励，因此没有故障。另外，根据 $\boldsymbol{R}_2$ 第3行出现的全1，与 $\boldsymbol{T}$ 对照很容易判断是 IC2 的引脚3开路。同理，根据 $\boldsymbol{R}_3$ 的第2行与第3行出现的相同响应，可以判断 IC2 的引脚2与引脚3之间发生短路。

具体操作的第1步可先加载并执行预载指令 Preload，再将测试激励矩阵 $\boldsymbol{T}$ 的第1列移入边界扫描寄存器；第2步加载并执行 Extest 指令后，一面移入 $\boldsymbol{T}$ 的其他列作为新的激励数据，一面顶出响应数据作为 $\boldsymbol{R}$ 的列向量；依次进行第2步操作过程，直至得到完整的响应矩阵 $\boldsymbol{R}$。显然，最后一次移入的数据没有意义，只是为了顶出前次激励作用产生的响应。

#### 2. 内部功能的测试

为了测试电路板上集成电路芯片的内部故障，可先加载 Preload 指令，然后加载测试向量；接着加载 Intest 指令，一面加载新的测试向量，一面读取对前次测试向量的响应向量；依次输入所有测试向量，读回全部响应结果。这样，根据输入测试向量和输出响应结果，即可对电路板上各芯片的内部工作状态做出测试分析，找出可能存在的故障。

#### 3. 工作状态的监测

为了监测正常工作状态下各芯片引脚处的实际工作状态，可以通过取样指令 Sample 对芯片引脚上的状态信号进行“快拍”。即将这些正常工作的信号状态直接捕捉到各引脚的 BSC 中，然后可通过移位操作将其从 DTO 口串行输出。

#### 4. 检查元件的安装位置

通过标识代码指令 IDCode 检查各集成电路元件中标识寄存器的内容，根据得到的芯片序列号和生产厂家信息来判断元件安装的正确性。

#### 5. 微处理器调试功能的实现

对于微处理器一类可编程元件，常常利用 JTAG 接口配置各种专用的特别指令，从而使得边界扫描技术的应用得以拓展和延伸。如通过读/写片内的 FLASH 存储器，实现下载程序、读取数据，以及换码设置软断点、进行单步操作等系统调试功能。

### 7.4.5 边界扫描的描述语言 BSDL

需要指出，各器件的指令编码并不完全统一，如 Bypass 的指令编码被规定为{111…11}的形式，Extest 指令被编码为{000…0}的形式，但其长度却随器件的指令寄存器而异。Sample/Preload 指令则完全由器件开发者自主编码，因此厂家通常需要通过一种称为 BSDL 的标准描述语言将芯片中有关边界扫描的内容以文件方式表达出来。

边界扫描描述语言 BSDL（Boundary Scan Description Language）是超高速集成电路硬件描

述语言(VHDL)的一个子集，它为自动测图产生(ATPG)工具提供特定被测电路的相关信息，形成标准的测试逻辑，并与软件工具结合起来帮助用户分析测试结果和诊断故障。每一个边界扫描器件都应附有相应的 BSDL 文件。BSDL 使用 BNF 范式来描述句法，由整体(Entity)、组件(Package)和组件本体(Package Body)3 个主要部分组成。下面以德州仪器公司的 8 路缓冲器 sn74bct8244a 为例介绍 BSDL 语言的基本语法。

```
attribute INSTRUCTION_LENGTH of sn74bct8244a : entity is 8;        定义所有指令码长度为 8 位
attribute INSTRUCTION_OPCODE of sn74bct8244a : entity is           具体操作指令的二进制描述
        "EXTEST (00000000, 10000000)," &
        "BYPASS (11111111, 10000100, 00000101, 10001000, 00000001)," &
        "SAMPLE (00000010, 10000010)," &
        "INTEST (00000011, 10000011)," &
         ⋮
attribute BOUNDARY_CELLS of sn74bct8244a : entity is "BC_1";       BSC 单元类型
attribute BOUNDARY_LENGTH of sn74bct8244a : entity is 18;          BSC 单元数
attribute BOUNDARY_REGISTER of sn74bct8244a : entity is
—num(cell, port, function, safe [ccell, disval, rstl])             按参数对位解释
         "0 (BC_1, Y2(4), output3, X, 16, 1, Z)," &
        "1 (BC_1, Y2(3), output3, X, 16, 1, Z)," &
         ⋮
        "8 (BC_1, A2(4), input, X)," &
        "9 (BC_1, A2(3), input, X)," &
         ⋮
        "17 (BC_1, OE_NEG1, input, X)," &
        "18 (BC_1, *, control, 1)";
```

其中，num 为 BSC 单元号，最靠近 TDO 脚者为 0 号；cell 定义 BSC 类型；port 为相连器件引脚标号；function 给出引脚的 IO 功能；safe 为安全子段(不常用)，其后方括号内的参数表示可选用的 3 个子段。

由上述描述可以得到有关芯片控制的基本信息，如所有指令码长度为 8 位，外部测试指令码为“00000000”或“10000000”，扫描单元数为 18 等。

### 7.4.6　边界扫描在板级测试中的连接方式

在待测电路板上，支持边界扫描测试电路的芯片采用某种板级扩展方式组成一条或多条边界扫描链，通过电路板的 JTAG 端口与测试设备相连。如图 7-34 所示，板级边界扫描链可以采用不同的方式，各自具有不同的特点，应当根据不同情况和测试要求进行选用。

图 7-34(a)所示的单端串行方式是最简单、最广泛使用的扫描链路方式，但若板上连接有缺陷或移去某一器件，就会破坏边界扫描链路，使系统测试无法继续进行。图 7-34(b)所示的并行链路则可克服这个缺陷，此时 TDI 和 TMS 信号均独立分支，数据可以独立进入系统的不同部分。但由于 TDO 是共享的，数据的流出须由 TMS 控制进行多路转换，显得有些麻烦。在图 7-34(c)所示的串-并联结构中，TMS 和 TCK 是共享的，TDI 和 TDO 通道是独立的，允许数据独立地流入和流出系统各个部分。

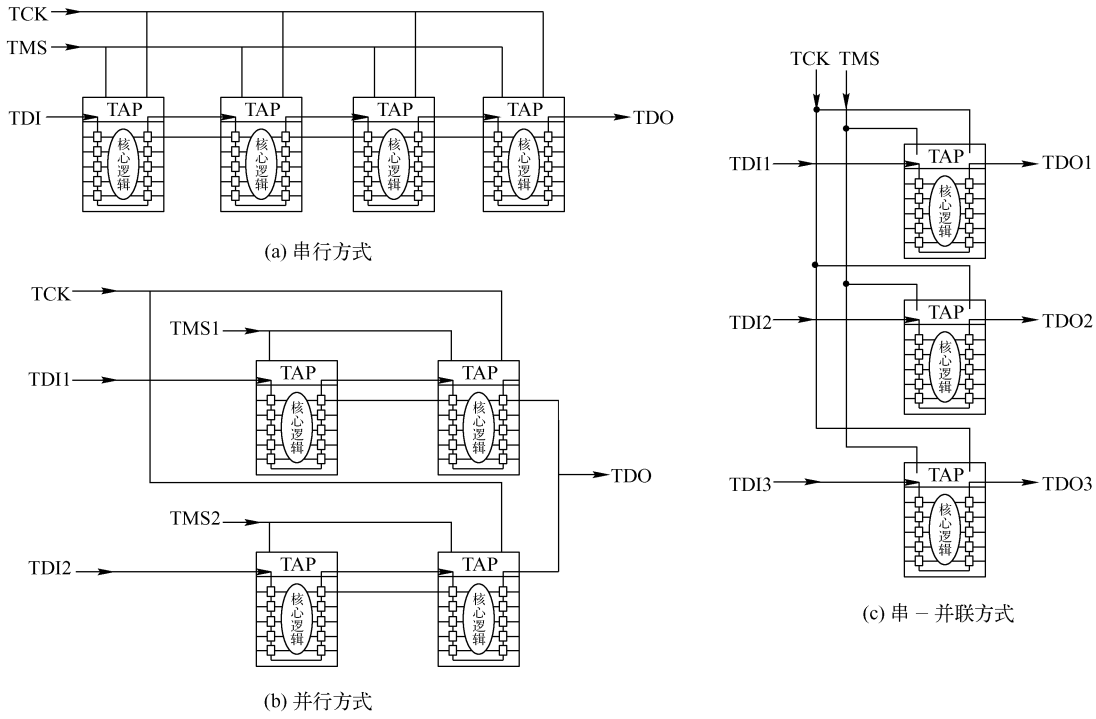



图 7-34 TAP 的连接方式

### 7.4.7 边界扫描测试系统的虚拟仪器示例

用于边界扫描测试的虚拟仪器示例系统由 3 部分组成：

1）PC 主机作为用户操作的主界面，用来描述测试对象和资源、生成测试矢量、分析测试响应，最终显示测试结果；

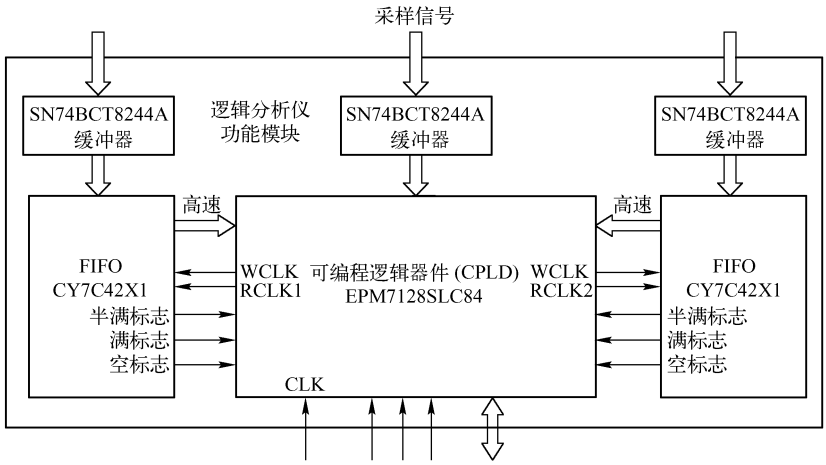
2）控制器直接控制被测对象的 JTAG 接口，实现标准的 IEEE 1149.1 测试命令和测试数据的传递；

3）被测系统必须经过可测性设计，使电路支持边界扫描。

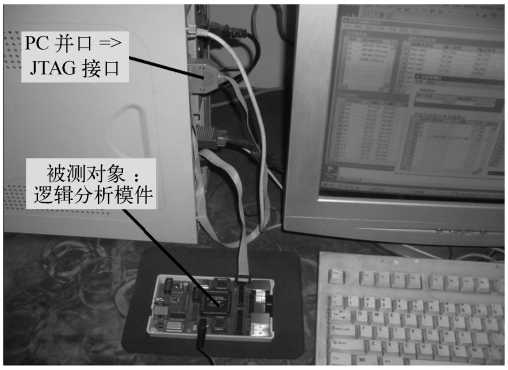
如图 7-35 所示，示例系统选择一个逻辑分析模件作为被测对象，其核心元件是 1 个大规模可编程逻辑器件(CPLD)，配合 3 个数据缓冲器和 2 个高速数据存储器(FIFO)来实现其设备功能。CPLD 的型号是 EPM7128SL84，已经支持边界扫描；而不支持边界扫描的数据缓冲器则需更换，这里采用了 TI 公司的 SN74BCT8244A。由于很难找到支持 JTAG 边界扫描的 FIFO 存储器，因此可以利用电路中 FIFO 夹在 CPLD 与数据缓冲器之间的结构条件，以及其逻辑简洁、控制简单的特点，通过对 FIFO 的某些信号加以控制，使保持在缓冲器输出引脚上的测试数据可以“穿过”FIFO 存储器，在 CPLD 的输入引脚捕获，从而构建出测试非边界扫描器件的“簇测试”形式。最终，将支持 JTAG 边界扫描的 BS 器件通过 TDI 和 TDO 首尾串行相连形成边界扫描链，并增设一个 JTAG 测试插座，将扫描链中总的数据输入输出信号 TDI 和 TDO，连同 TCK 和 TMS，构成访问被测电路的完整接口。

如图 7-36 所示，被测系统通过一条作为 JTAG 控制器电路的下载电缆与 PC 并口互连，就可利用所开发的软件实现对该电路板进行边界扫描测试。下载电缆接口的电路原理如

图 7-36所示，接口一方为与 PC 并头(打印口)相接的 25 针插头，另一方为与被测电路板 JTAG 端口相接的 10 芯插头。




(a) 被测系统结构




(b) 测试系统连接的实物照片

图 7-35　边界扫描虚拟仪器的典型示例

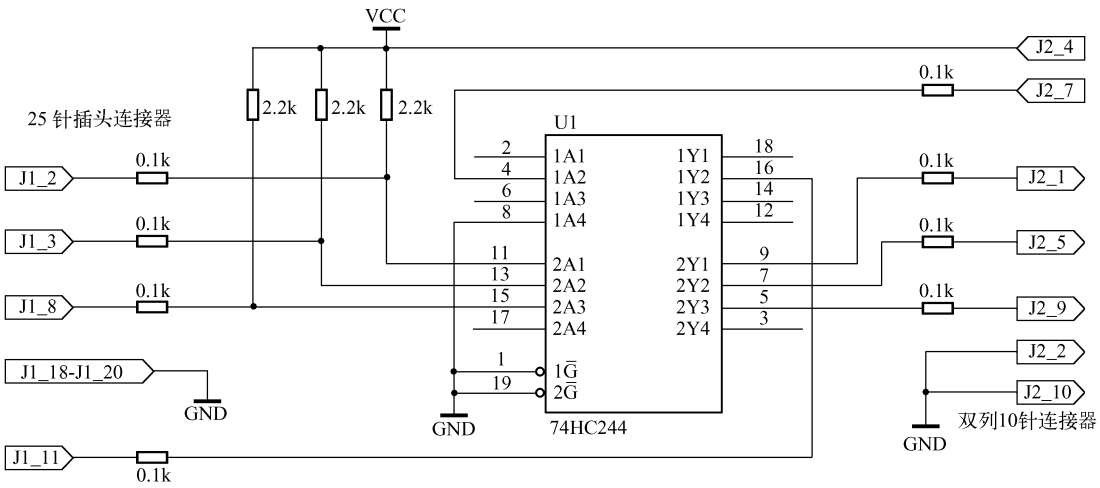


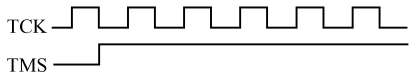

图 7-36　下载电缆原理图

并口寄存器分配 LPT1 的数据端口(0378H)作为输出，状态端口(0379H)作为输入，通过对这些 I/O 端口的读、写操作来模拟 JTAG 时序。由于 NT 环境下禁止应用系统直接操作外设，故 I/O 读/写需要驱动程序的支持。实现模拟 JTAG 信号时序的 VB 函数如程序清单 7-2 所示，其中函数 WrLpt 和 RdLpt 为直接读、写并口的驱动函数。

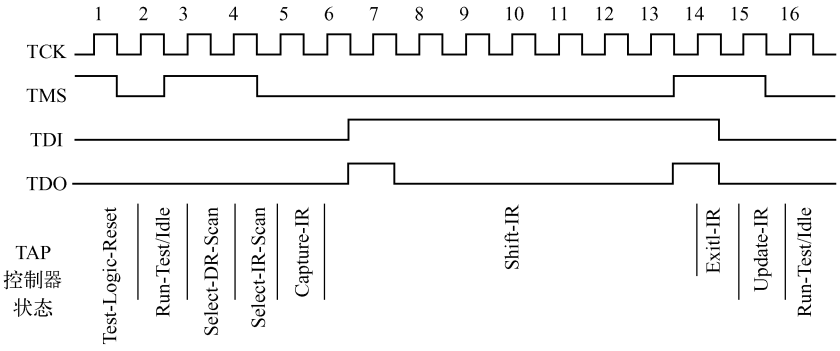
**程序清单 7-2　JTAG 控制器的接口信号**

```
Function RunJTAG(TDI$, TMS$) As String
  Dim b1% , b2% , s$, i% , d% , d1%
  n% =Len(TDI$): s$=""          '取 TDI 长度
  For i% =1 To n%
    If Mid(TDI$, i% , 1)="1" Then b1% =64 Else b1% =0
    If Mid(TMS$, i% , 1)="1" Then b2% =2 Else b2% =0
    WrLpt &H378, b1% + b2%     'TCK 作用
    WrLpt &H378, b1% + b2% + 1
    WrLpt &H378, b1% + b2%
    d% =RdLpt(&H379)    '读 TDO
    d1% =d% And 128: If d1% =0 Then s$=s$& "1" Else s$=s$& "0"
  Next i%
  RunJTAG=s$
End Function
```

为了说明上述采用并口电路及软件程序配合产生的 JTAG 信号是否符合协议标准，可以直接观察 TCK、TMS、TDI、TDO 的信号波形。如为了产生复位函数，可以调用函数 RunJTAG("xxxxx", "11111")，从而得到如图 7-37(a)给出 TMS 连续 5 个周期保持为高的实际波形。又如，为了读取芯片 SN74BCT8244A 的 IR 初值，需根据规范要求先为 TMS 发送时序信号“01100”，使 TAP 控制器从复位状态开始，先后通过 Run-Test/Idle、Select-DR-Scan、Select-IR-Scan、Captyre-IR 的过渡进入 Shift-IR 状态，然后在此状态下保持 8 个时钟，以便在 TCK 的




(a) 复位过程




(b) 读 IR 初值

图 7-37　JTAG 模拟信号检验

配合下从 TDO 移出芯片 SN74BCT8249A 的 IR 初值(8 位)。所调用的函数为 s$ = RunJTAG(string(5+8, "0"), "0110000000000"), 其中 string 函数用来产生多位特定字符的字串。函数返回结果为"10000001"(81H), 相应波形如图 7-37(b)所示。

### 7.4.8　边界扫描虚拟仪器的上层软件

边界扫描测试的上层软件主要包括人机界面、被测电路的描述、测试代码和测试向量的生成与下载、测试响应的分析和故障诊断, 这里仅以外部互连测试为例说明测试软件数据流的加工和处理过程。

如图 7-38 所示, 对被测电路的描述由扫描链路表文件(.lbs)、器件描述表文件(.bsm)和电路节点的网络连接表文件(.lex)完成。扫描链路表用以指出被测电路所含各 BS 器件的标识名称、所属的器件类型, 以及它们之间相互连接的次序。专门为某种器件建立的器件描述表则描述了该种器件所含的引脚、对应的 BSC 单元号及其属性, 器件的 BS 指令寄存器长度、具体编码、指令集, 以及其他相关信息, 是实现边界扫描测试的基础。电路节点的网络连接表则用于表达被测电路的拓扑结构, 集中说明电路中各元件之间的相互连接情况, 是进行互连测试的必要条件。由于扫描链路表和电路节点的网络连接表所表达的信息均与被测电路直接相关, 因此必须由系统使用者来建立。然而, 使用者可以通过系统功能将被测电路 PCB 设计的 Protel 网络表文件(.net)直接转换成电路节点连接表文件(.lex)。同时, 也可以通过系统功能将 BS 器件厂商提供的 BSDL 文件(.bsd)转换为本系统所用的某种器件描述表文件(.bsm)。

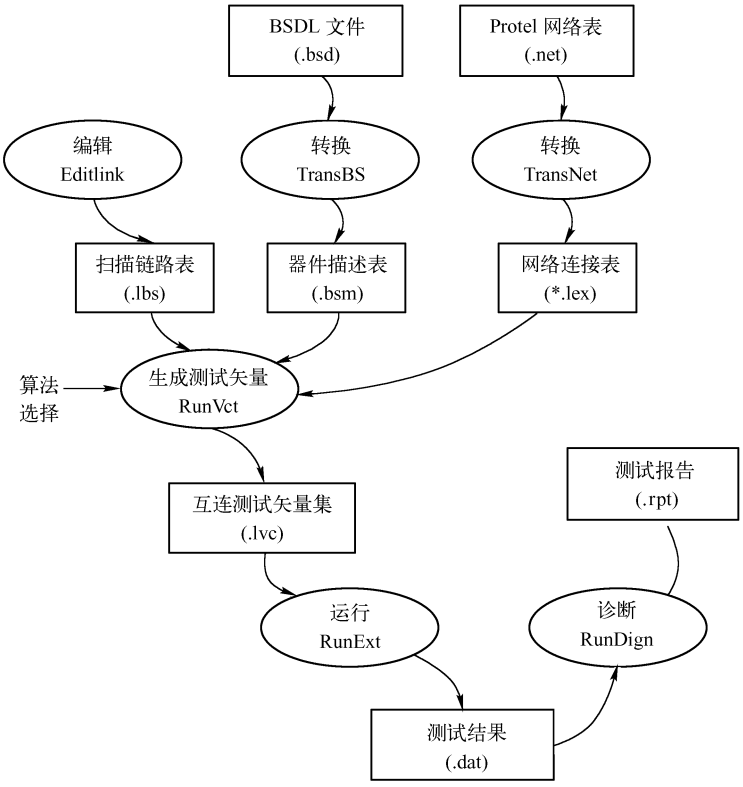



图 7-38　测试软件数据流程

测试向量的生成要依据具体的测试目的。对外部互连测试，只有通过 .lex 文件才能获取电路本身的互连描述，只有通过 .lbs 文件和相关的 .bsm 文件，才能获取对所建边界扫描链路的具体描述。据此，再根据不同的算法就可生成互连测试的向量集文件(.lvc)。不同的测试算法生成的测试向量集不一样，测试运行的时间也不一样，产生的抗误判和抗混叠的能力也不一样。走步 1(Walk-1)算法是一种相当保守却相当可靠的方法，其基本思想是将加到各输入点的初始测试向量设为 1，0，0，…，0 的形式，然后让这个 1 逐次移位(故称走步 1)到其他输入点，从而通过多次测试形成对角线的输入矩阵。它能够检测开路、短路和呆滞故障，且能较快地定位，因此常被用做系统的基础算法，也作为与其他算法相互比较的参照。

系统的操作主界面应当能够一目了然地体现当前被测电路的特征，考虑扫描链路表、电路节点连接表及簇测试文件是描述被测对象之间相互区别的基本信息，故应作为主界面的主要内容。如图 7-39 所示，主操作界面还有一组下拉式的菜单选项，分别用于数据文件的保存、加载、编辑、转换，同时也可通过一些选项进入完备性测试、功能测试、互连测试、簇测试的子工作界面。




图 7-39 系统人机交互界面

功能测试和互连测试是测试的主要内容，分别用于检查芯片内部的逻辑功能和外部电路的连接状况。完备性测试用于测试前的准备工作，以确保本身扫描链路的完备性。簇测试则针对非边界扫描器件的存在，让用户根据相邻器件可能利用的测点条件，灵活选择测试向量和测试步骤。

### 7.4.9 混合信号测试及总线标准

针对数字电路的边界扫描体系结构及测试访问端口 IEEE 1149.1 标准，无论是从原理上还是实践上来讲，都是相当成功的。计算机依托 JTAG 接口管理被测功能模件中各芯片内部

的边界扫描电路，不但可以解决 PCB 电路的互连测试，也可以直接用于芯片内部逻辑功能的测试，还常常扩展到芯片内部的各种调试功能，如程序加载、设置断点等。然而，当今 PCB 功能模件中有相当多的电路属于数字信号和模拟信号混合的情况，自然希望也能建立一种可以处理模拟信号的虚拟探针，因此产生了混合信号边界扫描测试的概念和要求。1999 年 6 月，混合信号测试总线标准 IEEE 1149.4 得以发布。

如图 7-40 所示，IEEE 1149.4 混合信号测试总线在原有 IEEE 1149.1 测试总线的基础上，扩展了模拟测试访问端口 ATAP(Analog Test Access Port)的两个引脚 AT1 和 AT2，而在芯片内部增加了模拟边界扫描单元 ABM(Analog Boundary Module)和测试总线接口电路 TBIC(Test Bus Interface Circuit)，以及两条连接所有 ABM 的模拟总线 AB1 和 AB2。

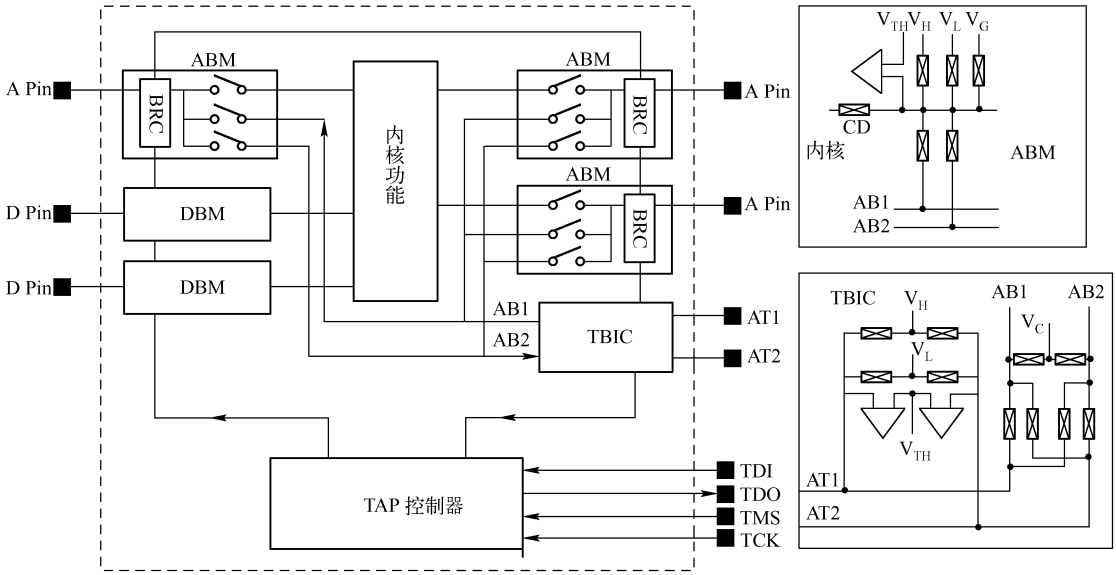



图 7-40　IEEE 1149.4 混合信号边界扫描的结构原理

ABM 和 TBIC 的主要作用是通道转换，ABM 中用 6 个模拟开关实现模拟测点与 AB1、AB2、芯片内核功能、参考地电位 $V_G$，以及高、低逻辑电平($V_H$ 和 $V_L$)之间的连接。TBIC 则用 10 个开关实现模拟测试端口 AT1、AT2 与内部总线 AB1、AB2 以及逻辑电平之间的任意切换，以适应各种测试模式的不同要求。ABM 和 TBIC 中这些内部开关的控制由其内部对应的边界扫描寄存器 BRC 完成，而 BRC 则由边界扫描链中的 4 个 BSC 单元构成。BRC 根据 IEEE 1149.1 规定的方式接受 TAP 的控制，通过边界扫描链路从 TDI 串行移位输入数据和指令，实现开关的各种规定模式。这样，可以通过 TBIC 的开关将 AT1 和 AT2 分别连接芯片内部总线 AB1 和 AB2，再经 ABM 就可连接芯片的任意引脚，从而实现建立模拟信号虚拟探针的设想。通常可将 AT1 外接测试系统的模拟信号源，AT2 外接模拟量测量仪器，则可向芯片任意引脚处发送模拟激励信号，并从芯片任意引脚处采集模拟响应信号。

由于 IEEE 1149.4 标准在 ABM 中提供了用于激励数字电路的高电平 $V_H$、低电平 $V_L$，使模拟引脚也能仿真 IEEE 1149.1 标准的数字状态，并能以 $V_{TH}$ 作为门阀电平来捕获数字响应信号。因此，可以说 IEEE 1149.4 是 IEEE 1149.1 的超集，不但支持模拟信号的互连测试、参数测试和功能测试，也支持原先数字信号的测试功能。

如图 4-41 所示，IEEE 1149.4 将 IC 间的直接连线定义为简单互连方式，将 IC 间为无源元件组成的阻抗网络定义为扩展互连方式，而将 IC 间为有源网络的情况定义为差分互连方式(这种情况极少，一般有源元件都已集成到 IC 之中)。

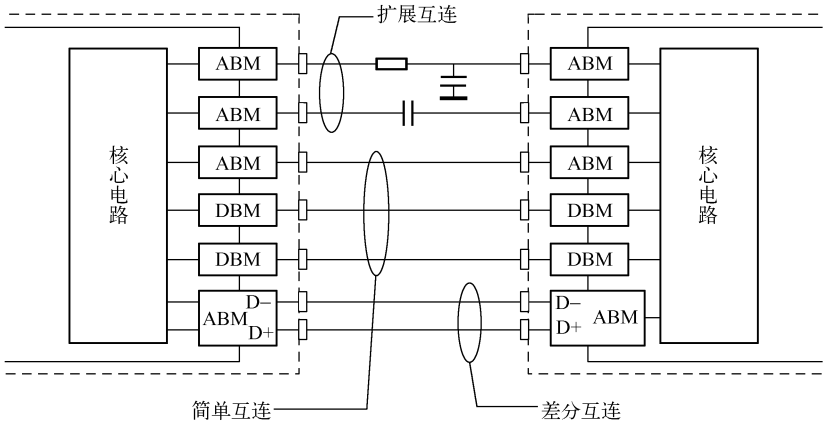
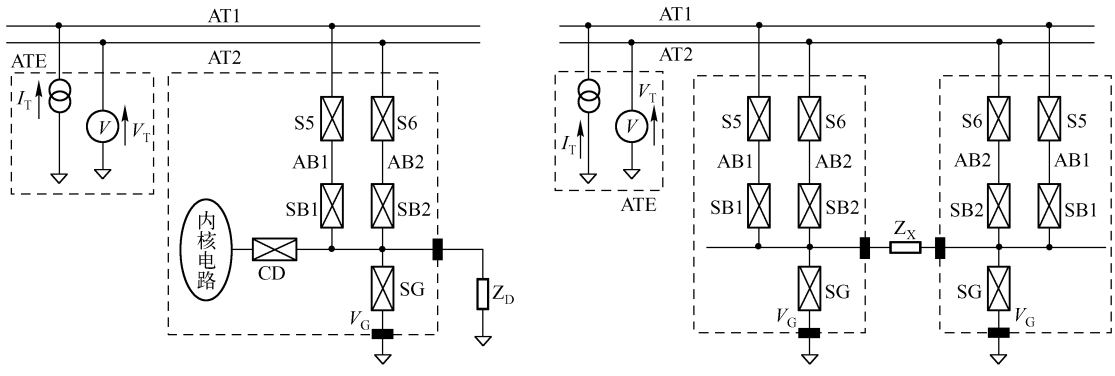



图 7-41　扩展互连测试的原理

简单互连测试主要用来测试器件连线间的开路、短路和网络间桥接等故障。测试时先用 Sample/Preload 指令给输出型的边界扫描单元预置测试激励，执行 Extest 指令之后捕获该连接线上输入型边界扫描单元的测试响应，然后通过对预置激励信号和测得的响应信号进行比较，即可分析出该简单互连线是否发生了短路、断路、桥接等故障。由于模拟引脚与数字引脚兼容，可在控制逻辑的作用下将 $V_H$ 或 $V_L$ 施加到输出型的模拟引脚上作为预置测试激励(相当于数字引脚加入高电平或低电平激励)。执行 Extest 指令之后，将该连接线上的输入型模拟引脚上捕获的模拟量电压与 $V_{TH}$ 门阀电平比较，便产生数字化的“0”或“1”值，作为测试响应信息，交给边界扫描链移出。这样，就可将模拟电路的简单互连测试转化为数字电路的互连测试。

扩展互连测试是针对引脚之间由电阻、电容、电感等无源元件或由它们组成网络的参数测试，即要求测出该无源网络中 R、L、C 的元件值。如图 7-42(a)所示，$Z_D$ 跨接在被测电路中某芯片模拟引脚与地之间，对它的测试可分步进行。




(a) 单端接地的情况　　(b) 两端互连的情况

图 7-42　扩展互连测试的原理

ATE 测试设备先将一个已知的模拟量测试激励电流 $I_T$ 通过测试线 AT1，经芯片 IC 中 TBIC 的开关 S5 接入其内部总线 AB1，再经 ABM 的开关 SB1 施加到被测电路 $Z_D$ 的上端点。同时将此端点对地的电压信号通过 ABM 的 SB2、内总线 AB2、TBIC 的开关 S6、测试线 AT2，由 ATE 测出其值，记为 $V_T$。这样，由 $V_T$ 除以 $I_T$ 就可算出 $Z_D$ 的阻值。而采用不同频率的交流模拟激励电流信号，则可通过计算求得电感或电容的元件值。如果 $Z_D$ 是由多个电阻、电感或电容组成的单端口网络，那也只需施加一组不同频率的激励电流，然后测量 $Z_D$ 上对应频率的电压降就能得到一组联立方程，解此方程组即可求出各元件值。AT1 采用电流源激励，可以有效地克服开关通道中开关本身压降的影响；而 AT2 电压测量通道中尽管存在开关内阻，但因电压测量设备的高输入阻抗特性，故通道中并无输入电流及其引起的压降存在。

如图 7-42(b)所示，对于被测电路元件两端分别连接两个芯片引脚的情况，可仍将 AT1 作为电流源信号激励，但将 $Z_x$ 的另一引脚通过内部开关连接到参考电位 $V_G$(通常为地电平)，然后通过 AT2 分别测量两引脚处的电平，从而算出互连的阻抗。

基于虚拟仪器的边界扫描测试设备相对简单，包括混合信号的测试也只是增加了一组模拟信号激励源和数据采集器。图 7-43 给出一个 USB 虚拟仪器的设计思想，其中单片机 F320 直接连接 PC 主机，同时可利用其 I/O 端口直接产生 JTAG 数字信号，并在少量集成电路组成的模拟测试电路和通道调理电路的配合下实现对模拟信号 AT1 和 AT2 的激励和监测操作。

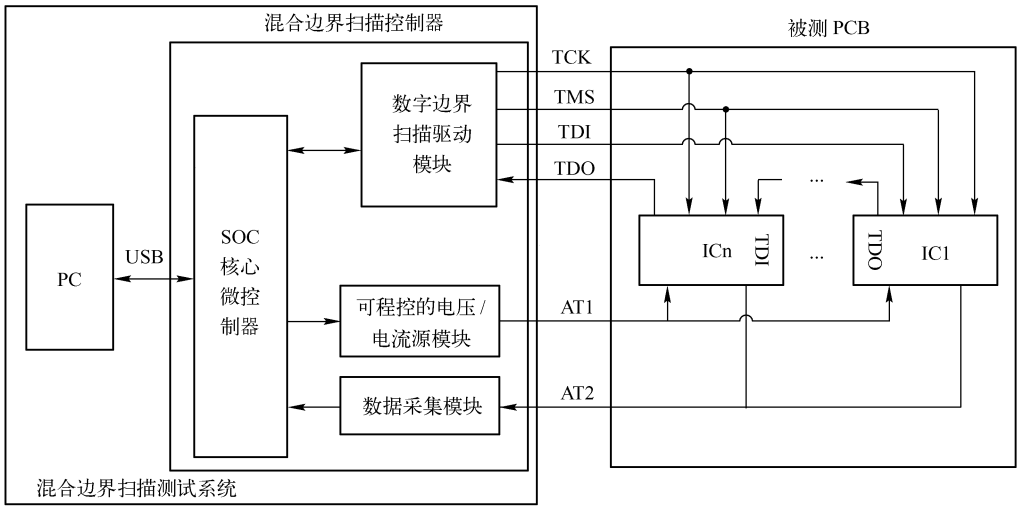



图 7-43　混合信号边界扫描测试的虚拟仪器方式

目前市场上支持 IEEE 1149.4 标准的芯片还很少。在此介绍 NS 公司生产的一种典型元件 STA400，以此说明混合边界扫描测试的具体应用情况。如图 7-44 所示，其内核功能为 2 组 2 选 1(或 1 组 4 选 1)的多路通道选择器，共 20 个引脚。模拟信号的边界扫描功能包含 11 个 ABM 单元和 1 个 TBIC 单元，其 BRC 控制由扫描链路上的 48 个 BSC 组成，能够提供多个模拟测试的“虚拟探针”。可执行 Extest、Probe、Bypass 等指令功能。

实际试验结果表明：AT1 和 AT2 可加载(监控)的直流电压为 -480mV ~ 5V；最大测试电流为 500μA。当从 AT1 处施加频率为 40kHz 的正弦波输入信号时，在片内 ABM 中串连开关的作用下，从 AT2 处观测的输出响应信号没有明显的衰减和相移。随着信号频率的增加，由

片内开关的等效电路产生的阻容效应增大，致使系统的衰减和相移加大。例如，输入频率为100kHz时相移可达4°左右；输入频率为1MHz时，AT2观测的输出响应信号约有-3dB的衰减。即芯片幅频带宽约为1MHz，考虑相移要求后通常也可工作在30kHz的频率范围。

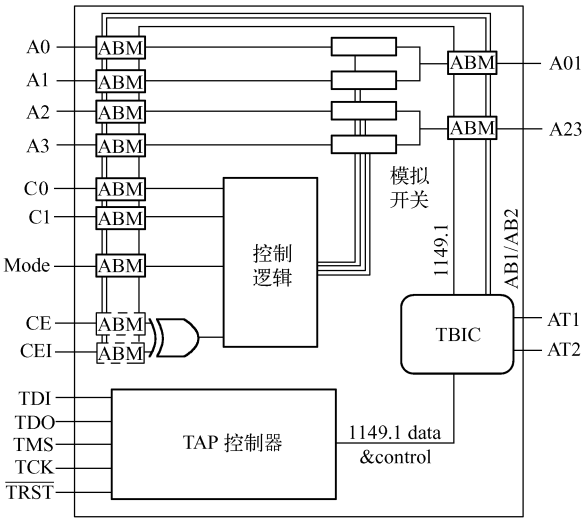



图 7-44 STA400的结构原理

## 本章小结

用于PCB电子功能模件的ATE技术是自动测试技术实际应用的重要领域。在大量电力电子器件采用PCB结构的现实情况下，传统的路内测试仍然很有意义。功能测试通过数字/模拟混合的测试设备功能，可以直接检查被测产品的质量，实用效果明显。边界扫描方法的硬件测试设备十分简单，且不仅能够测试芯片内部的工作状况，还能检查板级电路中各元件之间的外部互连状况，已经成为PCB电子功能模件自动测试的发展方向之一。随着虚拟仪器设计思想的普及和计算机信息处理能力的持续提高，PCB电子功能模件的ATE技术将会更多、更灵活地使用更先进的各种算法来提高测试效率、降低测试成本，从而更好、更全面地体现自动测试系统和虚拟仪器的优越性。

## 参考文献

[1] 杨士元.模拟系统的故障诊断与可靠性设计. 清华大学出版社，1993.
[2] 陈光踽,潘中良. 可测性设计技术. 电子工业出版社，1997.
[3] IEEE Standard Test Access Port and Boundary-Scan Architecture. IEEE Std 1149.1-2001.
[4] IEEE Standard for a Mixed Signal Test Bus IEEE 1149.4-1999. IEEE Computer Society.
[5] 刘思久，张礼勇. PCB功能模件的测试. 中国仪器仪表学会电测分会年会报告，2004.9.
[6] 刘思久,汪秀丰，董景. 基于虚拟仪器方式的PCB功能测试. 电测与仪表. 2004.1.
[7] 刘思久,于小秋. 阻抗元件在线测试的误差分析. 电测与仪表，2004.7.
[8] 刘思久,陈意，郑春平. PCB路内测试系统的开发. 电测与仪表，2007.12.

[9]　刘思久,罗艳. 虚拟逻辑分析仪. 电子测量技术 2005 年增刊,p1.
[10]　刘思久,于德伟, 罗艳. 基于虚拟仪器的 PCB 数字功能模件的测试. 哈尔滨理工大学学报, 2005.6.
[11]　刘思久,罗艳, 郑春平, 于德伟. 用于边界扫描测试的虚拟仪器开发. 仪器仪表学报, 2007(S1).
[12]　刘思久,郑春平, 黄辉. 用于数字电路调试的边界扫描测试仪开发. 仪器仪表学报(增刊),2007.11.
[13]　王让定,叶富乐 杜呈透.基于边界扫描技术的 TAP 接口研究.计算机工程, 2003.3.
[14]　管剑波,崔向东等. JTAG 边界扫描技术在高速路由器硬件测试中的应用.计算机工程, 2001.8.
[15]　王孜,刘洪民, 吴德馨. 边界扫描测试电路的设计. 微电子学(1) , 2003.
[16]　方葛丰, 布乃红, 宋斌, 邹芳宁. 边界扫描与电路板测试技术. 电子设计应用, 2003.4.
[17]　Harry BleekerPeter, Van, Den Eijnden, Fran de Jang. Boundary-scan Test-A Practical Approach. Kluwer Academic Publishers, 1993.
[18]　刘家欣, 肖大雏, 王彬如. 边界扫描在 PCB 缺陷测试中的应用. 计算机测量与控制,2004.5.
[19]　Larry Cicchinelli. Signal-Board Logic Analyzer. Ciruit Cellar,2004.10.
[20]　郭学仁.边界扫描可测性设计及其 VHDL 描述. 桂林电子工业学院学报,1995.4.
[21]　李正光,雷家. 混合信号总线测试——总线与封装. 2005.11.

# 第 8 章　典型软件算法和典型应用系统

自动测试系统和虚拟仪器的应用最终是通过计算机软件体现的，与用户交互的界面是软件，测试过程的总体安排是软件，主要的分析计算功能也是通过软件实现的。本章将针对自动测试应用系统的一些典型要求，从频谱分析、数字滤波器设计、基于时间序列的建模和常用数理统计等典型分析计算问题入手，介绍软件开发的基本方式和技巧，并结合典型自动测试系统和虚拟仪器的工作特点，重点阐述面向测试的计算机应用软件开发及相关的系统设计方法。

## 8.1　频 谱 分 析

在自动测试系统和虚拟仪器的工作过程中，常常需要通过计算机来计算信号的频谱。测试工程师不但需要了解频谱分析的基本概念和计算方法，还应熟悉数值计算的技巧，切实掌握计算结果与频谱特性参数之间的关系。

### 8.1.1　频谱分析的基本概念

对周期函数 $f(t)=f(t+T_1)$，只要满足狄里赫利条件，必能展成傅里叶级数：

$$f(t)=\frac{a_0}{2}+\sum_{n=0}^{\infty}a_n\cos(n\omega_1 t)+\sum_{n=0}^{\infty}b_n\sin(n\omega_1 t) \tag{8-1}$$

式中：

$$a_n=\frac{2}{T_1}\int_{-T_1/2}^{T_1/2}f(t)\cdot\cos(n\omega_1 t)\,\mathrm{d}t$$

$$b_n=\frac{2}{T_1}\int_{-T_1/2}^{T_1/2}f(t)\cdot\sin(n\omega_1 t)\,\mathrm{d}t \tag{8-2}$$

由此看出周期函数信号 $f(t)$ 可以分解为一系列正弦波信号分量，且基波角频率 $\omega_1=2\pi/T_1$，而高次谐波的角频率则为基波角频率的整数倍。

上述傅里叶级数也可以写成以下指数形式：

$$f(t)=\sum_{n=-\infty}^{+\infty}\dot{C}_n\cdot\mathrm{e}^{\mathrm{j}n\omega_1 t}$$

$$\dot{C}_n=\frac{1}{T_1}\int_{-T_1/2}^{T_1/2}f(t)\cdot\mathrm{e}^{-\mathrm{j}n\omega_1 t}\,\mathrm{d}t \tag{8-3}$$

式中，复系数 $\dot{C}_n=c_n\cdot\mathrm{e}^{-\mathrm{j}\theta}$ 可以分解为幅值和相角两部分。

根据复指数函数与三角函数关系的欧拉公式：

$$e^{-jn\omega_1 t}=\cos(n\omega_1 t)-j\sin(n\omega_1 t) \tag{8-4}$$

将式(8-4)代入式(8-3)，并考虑式(8-2)，可以推出复系数$\dot{C}_n$的实部和虚部：

$$\begin{aligned}\mathrm{Re}(\dot{C}_n)&=\frac{1}{T_1}\int_{-T_1/2}^{T_1/2}f(t)\cdot\cos(n\omega_1 t)\,\mathrm{d}t=\frac{a_n}{2}\\ \mathrm{Im}(\dot{C}_n)&=-\frac{1}{T_1}\int_{-T_1/2}^{T_1/2}f(t)\cdot\sin(n\omega_1 t)\,\mathrm{d}t=-\frac{b_n}{2}\end{aligned} \tag{8-5}$$

并有幅值和相角关系如下：

$$\begin{aligned}c_n&=\frac{1}{2}\sqrt{a_n^2+b_n^2}\\ \theta_n&=\tan^{-1}(b_n/a_n)\end{aligned} \tag{8-6}$$

即

$$f(t)=\sum_{n=-\infty}^{+\infty}c_n\cos(n\omega_1 t-\theta_n)=\sum_{n=-\infty}^{+\infty}c_n\sin(n\omega_1 t+\theta_n) \tag{8-7}$$

式(8-7)表明周期函数信号$f(t)$可以看做由一系列幅值为$c_n$、相角为$\theta_n$且角频率正、负相反的正弦波信号组成。所表达的频率特性为一系列离散的频率分量，称为线谱。

对非周期信号，可以看做周期$T_1\to\infty$时的周期函数。讨论以下极限情况：

$$f(t)=\lim_{T_1\to\infty}\sum_{k=-\infty}^{+\infty}\left[\frac{1}{T_1}\int_{-T_1/2}^{T_1/2}f(\tau)\cdot e^{-jk\omega_1\tau}\mathrm{d}\tau\right]\cdot e^{jk\omega_1 t} \tag{8-8}$$

由于此刻$\omega_1=\frac{2\pi}{T_1}=\Delta\omega\to\mathrm{d}\omega$，$k\omega_1=k\Delta\omega\to\omega$，求和变为以下积分：

$$\begin{aligned}f(t)&=\lim_{\Delta\omega\to0}\left\{\sum_{k=-\infty}^{+\infty}\left[\frac{\Delta\omega}{2\pi}\int_{T_1/2}^{-T_1/2}f(\tau)\cdot e^{-jk\Delta\omega\tau}\mathrm{d}\tau\right]\cdot e^{jk\Delta\omega t}\right\}\\ &=\frac{1}{2\pi}\int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty}f(\tau)\cdot e^{-j\omega\tau}\mathrm{d}\tau\right]\cdot e^{j\omega t}\mathrm{d}\omega\end{aligned} \tag{8-9}$$

于是可以得到傅里叶变换(积分)对：

$$\begin{aligned}f(t)&=\frac{1}{2\pi}\int_{-\infty}^{\infty}F(j\omega)\cdot e^{j\omega t}\mathrm{d}\omega\\ F(j\omega)&=\int_{-\infty}^{\infty}f(t)\cdot e^{-j\omega t}\mathrm{d}t\end{aligned} \tag{8-10}$$

对比式(8-10)和式(8-3)，可以看出非周期信号的频谱特性已经由线谱变为连续谱了。同时还应注意到，连续频谱的量纲也与线谱不同，线谱中表达频率分量的$\dot{C}_n$相当于连续谱中表达谱密度的$F(j\omega)$与$\mathrm{d}f=\mathrm{d}\omega/2\pi$的乘积。

借助数值积分(如矩形法)，在采样间隔为$T$、信号周期$T_1$内采样$N$次的情况下，可以得到下面适于计算机实现的频谱计算形式：

$$\begin{aligned}\dot{C}_n&=\frac{1}{N\cdot T}\sum_{k=0}^{N-1}f(k)\cdot\exp(-jn\omega_1\cdot kT)\cdot T=\frac{1}{N}\sum_{k=0}^{N-1}f(k)\cdot\exp(-jn2\pi k/N)\\ &=\frac{1}{N}\sum_{k=0}^{N-1}f(k)\cdot\left[\cos\left(n\frac{2\pi k}{N}\right)-j\sin\left(n\frac{2\pi k}{N}\right)\right]\end{aligned} \tag{8-11}$$

$$F(j\omega)=\sum_{k=0}^{N-1}f(k)\cdot\exp(-jn\Delta\omega\cdot kT)\cdot T=\sum_{k=0}^{N-1}f(k)\cdot\exp(-jn2\pi k/N)\cdot T$$
$$=\sum_{k=0}^{N-1}f(k)\cdot\left[\cos\left(n\frac{2\pi k}{N}\right)-j\sin\left(n\frac{2\pi k}{N}\right)\right]\cdot T \tag{8-12}$$

式中，$F(j\omega)$中的角频率$\omega$应为$\Delta\omega=2\pi/NT$的整数倍($n=0, 1, 2, \cdots$)。

### 8.1.2 离散傅里叶变换

离散傅里叶变换本质上是将采样得到的有限长度时间序列进行周期延拓后，形成离散时域序列到离散频域序列的对应关系，并得到如下简明形式：

$$f(k)=\frac{1}{N}\sum_{n=0}^{N-1}F(n)\cdot e^{jn2\pi k/N} \tag{8-13}$$

$$F(n)=\sum_{k=0}^{N-1}f(k)\cdot e^{-jn2\pi k/N} \tag{8-14}$$

这里，称式(8-13)为离散傅里叶变换的DFT算法，而称式(8-14)为离散傅里叶反变换的IDFT算法。对比式(8-13)和式(8-14)给出的DFT算法和IDFT算法，可以看出两者的形式很相似。考虑

$$f(k)=\frac{1}{N}\sum_{n=0}^{N-1}F(n)\cdot e^{jn2\pi k/N}=\frac{1}{N}\sum_{n=0}^{N-1}F(n)\cdot e^{-jn2\pi(N-k)/N} \tag{8-15}$$

令$p=N-k$，则有

$$f(N-p)=\frac{1}{N}\sum_{n=0}^{N-1}F(n)\cdot e^{-jn2\pi p/N} \tag{8-16}$$

故进行离散傅里叶反变换的IDFT算法，可以先直接使用DFT算法，然后将所得到的时间序列颠倒过来，再统统除以一个常数$N$即可，从而得出如下VB实现的IDFT程序代码。程序清单8-1是一个用VB语言编写的DFT程序，可以直接用来求取参数iw所指的频谱分量。

**程序清单8-1 DFT的函数实现**

```
Public Function RunDFT(ByVal nk As Integer, d() As Single, ByVal iw As
Variant, _
     vMag As Single, vPha As Single) As Integer '角频率=iw*6.28/(nk*采样间隔)
    Dim i%, k%, vr#, vi#
    vr#=0: vi#=0: dwt#=(PI+PI)*iw/nk  '常数PI=3.1415926
    For k%=0 To nk-1
        vr#=vr#+d(k%)*Cos(k%*dwt#)
        vi#=vi#-d(k%)*Sin(k%*dwt#)
    Next k%
    vMag=Sqrt(vi#*vi#+vr#*vr#)
    vPha=Atan2(vi#, vr#)  'Atan2*考虑参数象限后的三角函数
End Function
```

### 8.1.3 应用DFT进行频谱分析

DFT的算法直截了当，如用程序清单8-1给出的函数，就可以方便地将作为输入时域信号的数组数据转换成作为频域输出的数组数据。然而，需要特别强调的是：DFT的输入数组

数据表达的是实际时域信号截取后的采样序列，输出数据则为转换到频域后的采样值，其实际频谱特性的解释必须考虑信号的性质。如对周期信号 $f(t)$ 的采样序列 $f(k)$，通过 DFT 算法得到的 $F(n)$ 必须再除以一个常数 $N$，方为这个周期函数的谐波倍频线谱分量 $\dot{C}_n$；而对非周期函数的采样序列 $f(k)$，通过 DFT 算法得到的 $F(n)$ 则必须再乘以一个常数 $T$（采样时间间隔），方为这个非周期函数的连续频谱密度 $F(j\omega)$ 的频域采样值。这种简明的关系必须切实理解和掌握，方能实际用于频谱分析。

图 8-1 为离散傅里叶变换的示意图。如图所示，时域信号 $x(t)$ 的频谱为 $X(f)$，$x(t)$ 的采样信号为 $x^*(t)$，其频谱 $X^*(f)$ 被以 $1/T$ 的频域周期拓展。为了避免采样时间过大而引起频域“混叠”的问题（本书第 2 章已专门阐述），必须遵守采样频率高于信号带宽 2 倍的采样定理。

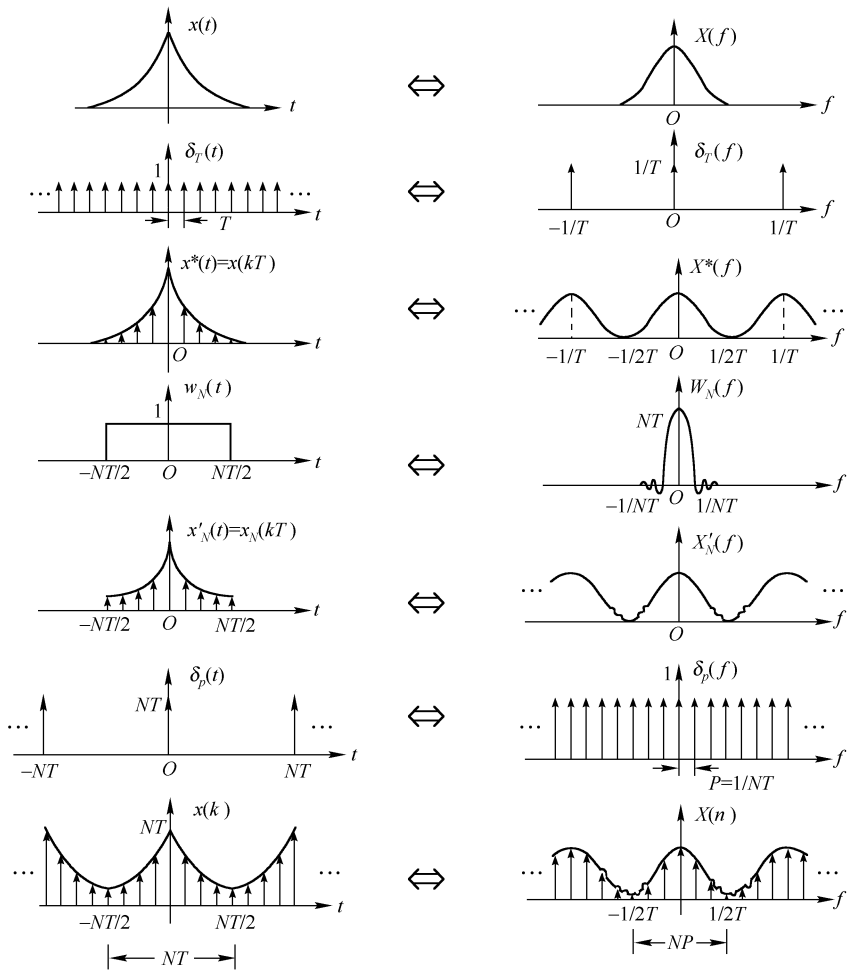



图 8-1　离散傅里叶变换的图解说明

另一个值得注意的问题是频域“泄露”。因为对时域信号采样时只取了 $N$ 点，相当于加了一个数据窗。如果只是简单地截断 $N$ 点后的数据，这种数据窗便是一个高度为 1、宽度为 $NT$ 的矩形窗 $w_N(t)$。其对应的频域函数 $W_N(f)$ 在与 $X^*(f)$ 主频进行卷积计算时，其非 $\delta$ 函数波形的边瓣就会引起频率“泄露”，表现为图中 $X_N^*(f)$ 波形呈现的皱纹。尽管 $w_N(t)$ 可以采用

诸如海明窗、海宁窗等特殊设计的窗函数来代替矩形窗以减少泄露的影响，但若采用整周期的采样，即 $N$ 个时域采样点刚好覆盖 $x(t)$ 信号周期的整数倍，则由于 $W_N(f)$ 在其频域采样点处（$P=1/NT$ 的整数倍）为 0 而使泄露作用消除。

### 8.1.4 FFT 的基本原理和程序实现

快速傅里叶变换（FFT）是实现离散傅里叶变换（DFT）的一类快速算法，正是由于 FFT 明显提高了计算效率，才使其得到了广泛的应用。具体的 FFT 算法很多，这里以一种称为 Cooley-Tukey 的实用算法为例，说明 FFT 的实现方法。FFT 要求采样数据的点数 $N$ 为 2 的整数次幂，因此对 $N=2^r$ 个数据的 DFT 公式可以写成：

$$F(n)=\sum_{k=0}^{N-1}f(k)\cdot\exp(-\mathrm{j}\cdot 2\pi kn/2^r)=\sum_{k=0}^{N-1}f(k)\cdot w^{k\cdot n} \tag{8-17}$$

这里定义因子

$$w=\exp(-\mathrm{j}\cdot 2\pi/2^r) \tag{8-18}$$

将 $k$ 和 $n$ 分别用二进制码 $k_r\cdots k_2k_1$ 和 $n_1n_2\cdots n_r$ 表示，并考虑 $w^p=w^{p+2^r}$ 的周期性，可得

$$w^{k\cdot n}=w^{k_1\cdot(2^{r-1}n_1+2^{r-2}n_2+\cdots+n_r)}\cdot w^{2\cdot k_2(2^{r-2}\cdot n_2+\cdots+n_r)}\cdots w^{2^{r-1}\cdot k_r\cdot n_r}$$

这样，式（8-17）的 DFT 算法可以转换成如下 $r$ 重求和的形式：

$$F(n_1n_2\cdots n_r)=\sum_{k_1=0}^{1}w^{k_1\cdot(2^{r-1}n_1+2^{r-2}n_2+\cdots+n_r)}\sum_{k_2=0}^{1}w^{2\cdot k_2(2^{r-2}n_2+\cdots+n_r)}\cdots\sum_{k_r=0}^{1}w^{2^{r-1}\cdot k_r\cdot n_r}f(k_r\cdots k_2k_1) \tag{8-19}$$

尽管从式（8-17）到式（8-19）的推导过程并不困难，但利用图 8-2 所示的信号流图则可以更容易理解 FFT 的具体计算过程，直接看出式（8-17）的内涵和本质。

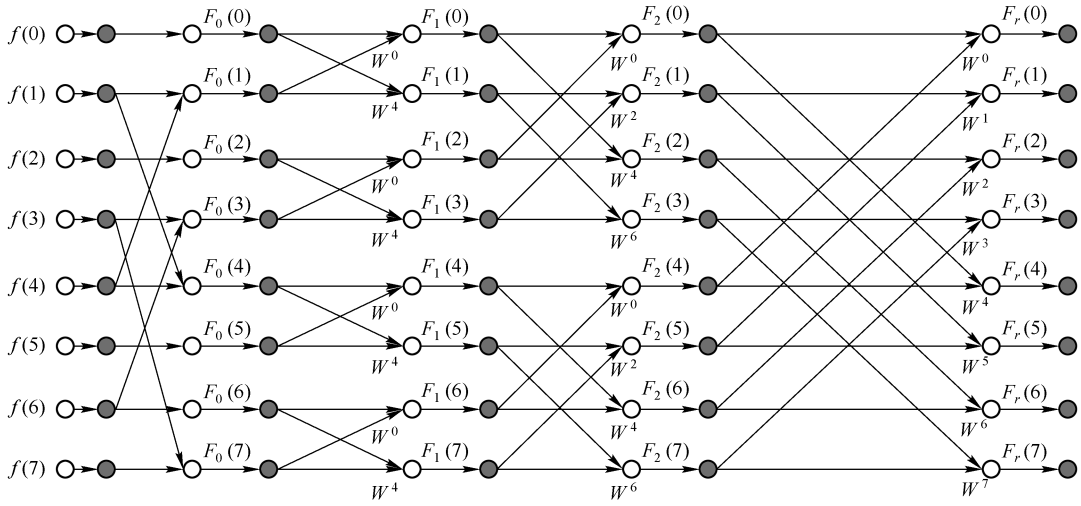



图 8-2 FFT 算法信号流图

图 8-2 中左端表明算法需要首先整序，将表示时序信号 $f(k)$ 的数列转换到数列 $F_0(n)$，且其 $k$-$n$ 关系就是将二进制码翻转过来，如 110 变为 011、100 变为 001。然后，依次进行 $F_0\rightarrow F_1$、$F_1\rightarrow F_2\cdots$的数列变换，直至得到 $F_r$，即为 FFT 的最终结果。

数列变换过程则由对偶的 4 个节点成对构成，如 $F_0(0)$、$F_0(1)$、$F_1(0)$和 $F_1(1)$，又如 $F_1(5)$、$F_1(7)$、$F_2(5)$和 $F_2(7)$。对偶节点中两数之间的序号差 $n$ 与信号流图的数列序号 $m$ 有关，即有 $n=2^{m-1}$。且对偶节点相乘支路的 $w$ 的幂次则相差 $2^{r-1}$，如图 8-2 中$r=3$，则对偶节点两支路的 $w$ 幂次相差 $2^{3-1}=4$。根据式(8-18)对 $w$ 的定义可知，偶节点上相乘因子相互对称(正、负相反)，即

$$w^{p+2^{r-1}}=w^p\cdot[\exp(-j\cdot 2\pi/2^r)]^{2^{r-1}}=w^p\cdot\exp(-j\pi)=-w^p \tag{8-20}$$

因此对每组对偶节点的计算可以只算一次 $w^p$，而后分别加、减到对偶节点的两处。FFT 算法的另一个特点是利用 $w^p$ 计算的周期性，考虑 $w^p=w^{p+p_1}$和 $p_1=2^r=2n$，故每计算一次 $w^p$，可用于所有相隔 $2n$ 点的相乘因子计算。

程序清单 8-2 是根据上述 FFT 算法实现的 VB 程序代码。函数参数 nr 以 2 的幂次指出时域输入信号数组数据 dt( )的维数，dr( )和 di( )分别为频域输出数据的实部和虚部数组。

**程序清单 8-2　FFT 函数**

```
Public Function RunFFT(ByVal nr As Integer, dt() As Single, dr() As Single, di
() As Single)
    Dim k% , i% , k1% , k2% , b% , n% , dr1#, di1#, PI#, a#, dwt#, nk% , nk1% , ndwt#
    nk% =1: For i% =1 To nr: nk% =nk% +nk% : Next i%
    Redim dr(nk%), di(nk%)
    For k% =0 To nk-1                                          ″倒序
        b% =1: k1% =0
        While b% < nk
            k1% =k1% +k1% : If k% And b% Then k1% =k1% +1
            b% =b% +b% :
        Wend
        dr(k1%)=dt(k%): di(k%)=0
    Next k%
    n% =1: PI#=3.1415926: dwt#=(PI#+PI#) /nk% : nk1% =nk%
    Do                                                         '计算各列
        a#=0: nk1% =nk1% /2: ndwt#=dwt# * nk1%
        For k% =0 To n% -1                                     '计算各项
            k1% =k% : c#=Cos(a#): s#=Sin(a#): a#=a#+ndwt#
            Do While k1% < nk%                                 '计算每项各对
                k2% =k1% +n%
                dr1#=dr(k2%) * c#-di(k2%) * s#: di1#=di(k2%) * c#+dr(k2%) * s#
                dr(k2%)=dr(k1%)-dr1#: di(k2%)=di(k1%)-di1#
                dr(k1%)=dr(k1%)+dr1#: di(k1%)=di(k1%)+di1#
                k1% =k2% +n%
            Loop
        Next k%
        n% =n% +n%
    Loop While (n% < nk%)
End Function
```

**例 8-1**　对图 8-3 所示 1kHz 的矩形波周期信号进行频谱分析，要求分析上至 8 倍和下至 1/4 倍基波的谐波分量，并指出 3 次谐波对应的数组元素。

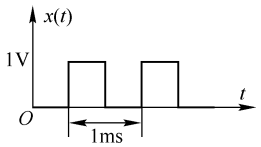


图 8-3　矩形波信号波形

**解：**按采样定理可选采样频率 16kHz，实取 32kHz。考虑采样 4 个信号整周期，则实选采样总长度 $N=128$。这样，将这 128 个数据存入数组 d( )后，便可使用如下程序实现频谱分析。其中 3 次谐波对应的输出数据数组元素应为 dm(12)。

```
RunFFT 7, d, dr, di          '2 的 7 次方为 128,d()为已知时域数据数组
N=128
For k% =0 To N/2-1
    dm(k%)=Sqr(dr(k%) * dr(k%)+di(k%) * di(k%)) /N
Next k%
```

注意，对周期信号调用 DFT 或 FFT 算法函数后必须再除以常数 $N$，才能获得正确结果。

针对 $N$ 个数据的 FFT 算法需要的基本运算为 $N\cdot r=N\cdot\log_2 N$ 次，而对应 DFT 算法则为 $N^2$ 次。当 $N=1024$ 时，两者相差高达 100 倍。

## 8.2　数字滤波器设计

为了消除干扰和噪声的影响，自动测试系统和虚拟仪器常常通过计算机软件程序和微处理器固件程序来构建数字滤波器。实际上，正确使用数字滤波器往往可以提高系统的测试精度，因为将量化噪声过滤掉，就等于提高了信号的分辨率。本节将重点讨论数字滤波器的工作原理、具体设计方法，以及使用数字滤波器的技巧和从实际工程项目中节选出的程序代码等问题。

### 8.2.1　数字滤波器的等效离散化设计方法

由第 2 章有关阐述可知：一般滤波器特性是由其频率特性描述的，进而可以写成描述其输入/输出关系的传递函数，并由有源或无源电路实现；虚拟仪器中的数字滤波器通常由计算机程序实现，算法遵循与原始传递函数等价的等效离散化传递函数。

通常离散传递函数 $G(z)$ 总是算子 $z^{-1}$（对应时域一拍延迟）的分式多项式，如

$$G(z)=\frac{Y(z)}{U(z)}=\frac{b_0+b_1z^{-1}+b_2z^{-2}+\cdots+b_mz^{-m}}{1+a_1z^{-1}+a_2z^{-2}+\cdots+a_nz^{-n}} \tag{8-21}$$

将其分式方程交叉相乘后，很容易转化为相应的计算机算法：

$$y(k)=-a_1\cdot y(k-1)-a_2\cdot y(k-2)-\cdots-a_n\cdot y(k-n)+$$
$$b_0\cdot u(k)+b_1\cdot u(k-1)+\cdots+b_m\cdot u(k-m) \tag{8-22}$$

这样，数字滤波器输出的当前值 $y(k)$ 可以通过简单的线性组合[illegible]值 $y(k-i)$，以及作为滤波器输入信号的当前值 $u(k)$ 与过去值 $u(k-j)$ 计[illegible]计算过程直截了当，使等效离散化过程成为数字滤波器设计的关键，其[illegible]效离散化方法是双线性法（Tustin，梯形积分）。

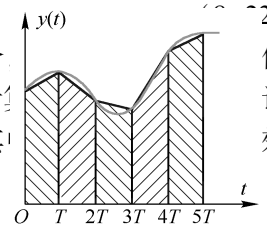


图 8-4　双线性变换原理

双线性法以数值积分为基础，从一个积分或微分的作用入手，可以找出一个与已知连续传递函数所对应的等效离散化传递函数。

如图 8-4 所示，为了等效普通模拟电路的积分器作用，双线性法用输入信号采样点之间的梯形面积作为积分增量，得到数字积分器的积分输出 $y(k)$ 与其输入 $x(k)$ 之间的关系为

$$y(k)=y(k-1)+\frac{T}{2}\cdot[x(k)+x(k-1)]$$

于是相应 $z$ 变换的输入、输出关系成为

$$Y(z)=z^{-1}\cdot Y(z)+\frac{T}{2}\cdot(1+z^{-1})\cdot X(z)$$

从而得到积分器 $1/s$ 的等效离散化传递函数为

$$\frac{Y(z)}{X(z)}=\frac{T}{2}\cdot\frac{1+z^{-1}}{1-z^{-1}} \tag{8-23}$$

相应的拉普拉斯算子 $s$ 则可表示为

$$s=\frac{2}{T}\cdot\frac{1-z^{-1}}{1+z^{-1}} \tag{8-24}$$

实际应用时，只要将普通滤波器连续传递函数中的所有算子 $s$ 均用式(8-24)替换，就可建立起离散条件下的等效离散化传递函数。下面以一个普通一阶滤波器为例来说明这种实现等效离散化的方法。

**例 8-2**　已知待实现的滤波器传递函数为 $G(s)=\dfrac{1}{\tau\cdot s+1}$，$\tau=1$，求等效数字滤波器。

**解**：根据双线性方法，考虑滤波器时间常数 $\tau$，设采样时间 $T=0.5\text{s}$，则有

$$G_{\text{d}}(z)=\frac{1}{\dfrac{2\tau}{T}\cdot\dfrac{1-z^{-1}}{1+z^{-1}}+1}=\frac{T}{2\tau+T}\,\frac{1+z^{-1}}{1-\dfrac{2\tau-T}{2\tau+T}z^{-1}}=\frac{0.2\cdot(1+z^{-1})}{1-0.6z^{-1}}$$

相应输出 $y(k)$ 与输入 $u(k)$ 的算法关系为

$$y(k)=0.6\cdot y(k-1)+0.2\cdot[u(k)+u(k-1)]$$

这里将 $G(s)$ 的等效离散传递函数用 $G_{\text{d}}(z)$ 表示而不用 $G(z)$ 表示是为了强调等效离散化过程并非对 $G(s)$ 直接进行 $z$ 变换。

根据 $z$ 变换定义的算子 $z=\text{e}^{sT}$ 的概念，并令 $s=\text{j}\omega$。考虑双线性变换，可得

$$\text{j}\omega_{\text{A}}=s=\frac{2}{T}\,\frac{1-z^{-1}}{1+z^{-1}}=\frac{2}{T}\cdot\frac{1-\text{e}^{-\text{j}\omega T}}{1+\text{e}^{-\text{j}\omega T}}=\frac{2}{T}\cdot\frac{\text{e}^{-\text{j}\omega T/2}\cdot(\text{e}^{\text{j}\omega T/2}-\text{e}^{-\text{j}\omega T/2})}{\text{e}^{-\text{j}\omega T/2}\cdot(\text{e}^{\text{j}\omega T/2}+\text{e}^{-\text{j}\omega T/2})}$$

$$=\text{j}\omega_{\text{D}}\cdot\frac{\text{tg}(\omega_{\text{D}}T/2)}{\omega_{\text{D}}T/2}>\text{j}\omega_{\text{D}}$$

式中，$\omega_{\text{A}}$ 和 $\omega_{\text{D}}$ 分别表示离散化变换前后的对应频点，可见双线性变换将会产生频率扭曲现象，即变换前模拟环节的频点 $\omega_{\text{A}}$ 的特性将会映射到变换后离散环节的 $\omega_{\text{D}}$ 处，具体数据如表 8-1 所示。

**表 8-1　双线性变换的频率扭曲数据**

| $\omega/\omega_{\text{s}}$ | 1/3 | 1/4 | 1/5 | 1/10 | 1/20 |
|---|---|---|---|---|---|
| $\omega T/2=\pi\cdot(\omega/\omega_{\text{s}})$ | 1.047 | 0.785 | 0.628 | 0.325 | 0.158 |
| $\text{tg}(\omega T/2)$ | 1.732 | 1.000 | 0.726 | 0.325 | 0.158 |

| | | | | | |
|---|---|---|---|---|---|
| $\mathrm{tg}(\omega T/2)/(\omega T/2)$ | 1.65 | 1.27 | 1.15 | 1.00 | 1.00 |

数字滤波器的频率特性可表达为

$$G_{\mathrm{d}}(\mathrm{j}\omega_{\mathrm{D}}) = G(\mathrm{j}\omega_{\mathrm{A}})$$

针对例 8-2 的参数，图 8-5 给出了其模拟滤波器传递函数 $G(s)$ 和等效离散化传递函数 $G_{\mathrm{d}}(z)$ 的频率特性对比。由 $T=0.5\mathrm{s}$ 可知采样频率 $\omega_{\mathrm{s}}=2\pi f_{\mathrm{s}}=12.5/s$。图中可见转折频率附近的幅频特性并无明显的频率扭曲现象，因此只要采样频率足够高，并不一定需要对此频率扭曲现象进行修正。尽管在 $\omega_{\mathrm{s}}/2$ 后幅频、相频曲线都出现跳变，但考虑采样定理，工作信号自然应当限制在此带宽之内。

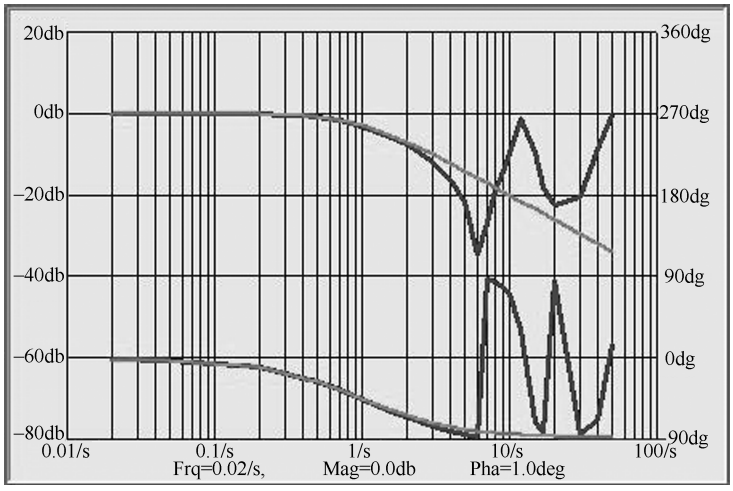



图 8-5 双线性变换前、后的频率特性对比

## 8.2.2 数字滤波器的 FIR 和 IIR 形式

数字滤波器分有限冲激响应的 FIR 和无限冲激响应的 IIR 两种方式，其 $z$ 传递函数表达式分别为

$$G(z)=\frac{Y(z)}{U(z)}=b_0+b_1z^{-1}+b_2z^{-2}+\cdots+b_mz^{-m} \tag{8-25}$$

$$G(z)=\frac{Y(z)}{U(z)}=\frac{b_0+b_1z^{-1}+b_2z^{-2}+\cdots+b_mz^{-m}}{1+a_1z^{-1}+a_2z^{-2}+\cdots+a_nz^{-n}} \tag{8-26}$$

从式(8-25)描述的传递函数结构可以看出：FIR 数字滤波器的传递函数的分式多项式中没有分母部分。也就是说，滤波器的输出信号仅与滤波器的输入信号以及输入信号的过去值有关，意味着滤波器对冲激信号的响应将只会持续有限拍，故滤波器系统本身总是稳定的。相反，式(8-26)描述的 IIR 数字滤波器的传递函数中存在分母多项式。也就是说，滤波器的输出信号不仅与滤波器的输入信号以及输入信号的过去值有关，还与其输出信号本身的过去值有关，因此产生无限持续的冲激响应。由于 IIR 滤波器 $z$ 传递函数可能存在单位圆外的极点，因此系统明显存在稳定性问题。

若型如式(8-26)所示的 IIR 数字滤波器稳定，则对其传递函数的分式多项式进行长除计

算，必然可以得到如式(8-25)所示的 FIR 数字滤波器。一般来说，IIR 数字滤波器构建简单，往往使用很低阶次的 IIR 就可以顶替很高阶次 FIR 的滤波作用。但 IIR 滤波器产生的相移是非线性的。

### 8.2.3　典型一阶 IIR 滤波器的参数确定

最典型的一阶低通滤波器传递函数为

$$\frac{Y(s)}{U(s)}=\frac{1}{\tau\cdot s+1} \tag{8-27}$$

采用双线性方法进行等效离散化后，可得相应离散系统的 $z$ 传递函数为

$$\frac{Y(z)}{U(z)}=\frac{b_0\cdot(1+z^{-1})}{1-a_1\cdot z^{-1}} \tag{8-28}$$

这里只有两个参数，参数 $a_1=\exp(-aT)$，$b_0=\exp(-T/\tau)$。显然，为了确保滤波器系统的稳定性，要求作为极点的 $a_1$ 位于单位圆内，即 $|a_1|<1$；同时，滤波器的输出也不宜出现上、下跳动的现象，因此要求极点 $a_1>0$；另外，低通滤波器的直流增益应取 1，因此根据 $z$ 传递函数的特性，可得

$$\lim_{z\to 1}\frac{Y(z)}{U(z)}=\lim_{z\to 1}\frac{b_0\cdot(1+z^{-1})}{1-a_1\cdot z^{-1}}=1$$

可得

$$a_1+2\cdot b_0=1 \quad\Rightarrow\quad a_1=1-2\cdot b_0$$

从对应的数字滤波器算法

$$y(k)=a_1\cdot y(k-1)+b_0\cdot[u(k)+u(k-1)] \tag{8-29}$$

可以直观地看出滤波器的关键参数 $a_1$ 的作用：$a_1$ 增大看重滤波，$a_1$ 减小则看重现实。极端情况下若 $a_1=1$，$y(k)=y(k-1)$，输出则完全不顾当前的观测结果而直接采用过去值；反之，若 $a_1=0$，$y(k)=u(k)$，输出则直接采用当前的观测结果而无滤波作用。

通常可按表 8-2 选择参数，即

$$a_1=\mathrm{e}^{-T/\tau} \tag{8-30}$$

其中，$T$ 为被测系统的采样时间，$\tau$ 为一阶滤波器的时间常数。

表 8-2　一阶滤波器参数的选择

| 参数 $a_1$ | 0.5 | 0.6 | 0.8 | 0.9 | 0.99 |
|---|---|---|---|---|---|
| 时间常数 $\tau$ | $T$ | $2.5T$ | $5T$ | $9T$ | $99T$ |

最终，一阶 IIR 数字滤波器只有一个参数 $a_1$，其实现算法如下：

$$y(k)=a_1\cdot y(k-1)+0.5\cdot(1-a_1)[u(k)+u(k-1)] \tag{8-31}$$

### 8.2.4　四点加权的 FIR 数字滤波器

四点加权的 FIR 数字滤波器基本算法为

$$y(k)=A_1\cdot u(k)+A_2\cdot u(k-1)+A_3\cdot u(k-2)+A_4\cdot u(k-3) \tag{8-32}$$

式中，$y(k)$ 和 $u(k)$ 分别为滤波器的输出和输入信号，显然计算只需要提供输入信号的当前

值和过去值，等价的 $z$ 传递函数为

$$G(z)=\frac{Y(z)}{U(z)}=A_1+A_2\cdot z^{-1}+A_3\cdot z^{-1}+A_4\cdot z^{-3} \tag{8-33}$$

为了确保直流信号分量不变，即要求直流增益为 1，则有滤波器系数关系如下：

$$\sum_{i=1}^{4}A_i=1 \tag{8-34}$$

如果取平均值，即只保留直流分量，则要求各系数取值为

$$A_1=A_2=A_3=A_4=0.25 \tag{8-35}$$

如果滤波结果看重当前作用，则可将靠近当前时刻数据的系数加大，如取

$$A_1=0.4,\ A_2=0.3,\ A_3=0.2,\ A_4=0.1 \tag{8-36}$$

另一方面，微分效果可用下列差分实现：

$$y(t)=\frac{\mathrm{d}}{\mathrm{d}t}u(t)\Rightarrow\frac{u(k)-u(k-1)}{T}$$

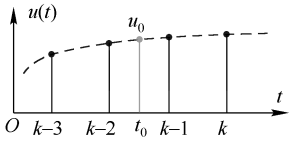


图 8-6 微分平滑原理

但考虑差分作用过于敏感，实际微分算法常采用一种具有平滑作用的典型算法。如图 8-6 所示，可采用对 $t_0$ 时刻的取值 $u_0$ 与其周围多点采样值之间分别进行差分计算后再取平均的方法。

$$\begin{aligned}y(k)&=\frac{1}{4}\cdot\left[\frac{u(k)-u_0}{1.5\cdot T}+\frac{u(k-1)-u_0}{0.5\cdot T}+\frac{u_0-u(k-2)}{0.5\cdot T}+\frac{u_0-u(k-3)}{1.5\cdot T}\right]\\&=\frac{1}{6\cdot T}[u(k)+3\cdot u(k-1)-3\cdot u(k-2)-u(k-3)]\end{aligned} \tag{8-37}$$

这种算法相当于四点加权 FIR 滤波器选取如下新的系数：

$$A_1=\frac{1}{6\cdot T},\ A_2=\frac{1}{2\cdot T},\ A_3=-\frac{1}{2\cdot T},\ A_4=-\frac{1}{6\cdot T} \tag{8-38}$$

比较式(8-35)、式(8-36)和式(8-38)，可以看出：同样的 FIR 滤波器，只要适当改变构成系数的数值，就能实现完全不同的滤波效果。这里式(8-35)实现的滤波器是典型的积分作用，属于低通滤波器；而式(8-38)实现的滤波器则是典型的微分作用，属于高通滤波器的范畴。

### 8.2.5 数字滤波器的使用

根据采样定理，如果采样频率低于被测信号的带宽或伴随的噪音频率带宽，就有可能发生频率分量“混叠”的情况，此时不但输出得不到信号的高频分量，而且低频分量也会由于“混叠”的作用产生畸变。按照采样定理，采样频率应为信号频率带宽的两倍以上，但夹杂噪声的实际信号频谱通常拖着一个长长的尾巴。对这种没有明显带宽的情况，需要在被测信号进入采样前接入一个模拟电路实现的低通滤波器来滤除高频分量，使被测信号变为有限带宽后再进行采样。

可能有人会问：“既然总要配置一个硬件滤波器电路，为什么还要使用数字滤波器呢?”

之所以不单独采用硬件技术有两个主要原因：

第一，高频截止滤波器仅仅需要小容量的电容器和电阻器(体积较小、成本较低)。但

低频段作用的滤波器则需要体积较大(且较昂贵的)电容器。因此，在采用抗混叠低通滤波器滤除高频分量后，就可以用软件的方法更为经济和方便地实现 Nyguist 频率内的滤波作用(如第 2 章讨论过的 Δ-Σ 式 A/D 转换器)。

第二，滤波器的时间常数有时需要适当调整，且对不同设备的不同应用场合，往往需要不同的时间常数。就软件实现的数字滤波器来说，调整时间常数与调节增益都十分容易，只要在式(8-22)描述的算法中适当改变各项的系数即可。但对硬件滤波器电路来说，靠移动磁心改变电感，或改变电容器和电阻器的值来改变时间常数就都困难得多。

使用数字滤波器的一个常见问题是没有正确地为其建立初值，有时则误将初值设为 0。正确的处理方法是取信号的平均值作为初值，使之接近于真实值，以使滤波器在上电时不会面临一个过大幅度的阶跃函数。常用的方法是在系统上电时读取传感器信号作为初始化的平均值。有时也将多次测量读数取平均值作为初值，但除非系统噪声很大，否则没必要这样做。因为设置初值只是为了接近实际值，以避免引起过大的波动，故这个初值不必精确。

注意，滤波器会在系统中引入滞后，当需要强调系统动态特性时这一点尤为重要，因此不要使用过多的滤波器使其动态特性变坏。另外，数字滤波器编程也有一些技巧，如采用分数化整数算法代替浮点运算可大大减少计算的时间。

## 8.3　时间序列的建模分析

自动测试系统工作任务从单纯的物理量直接测量发展到通过激励-响应过程辨识参数的间接测试，其测试对象也从静态参数发展到动态参数，通过建模并辨识其参数的方法越来越多地出现在自动测试过程中。本节将从最小二乘的参数估计算法开始，以时间序列的建模过程为例，说明解决这类辨识问题的一般方法。

### 8.3.1　最小二乘的参数估计算法(LS 算法)

对 $n$ 维线性代数方程

$$y=\theta_1\cdot x_1+\theta_2\cdot x_2+\cdots+\theta_n\cdot x_n \tag{8-39}$$

可用 $n$ 个参数 $\theta_1$、$\theta_2$、$\theta_n$ 简明地描述输出变量 $y$ 和 $n$ 个输入变量 $x_1$、$x_2\cdots x_n$ 之间的线性关系，并可用如图 8-7所示的权系数模型表示。

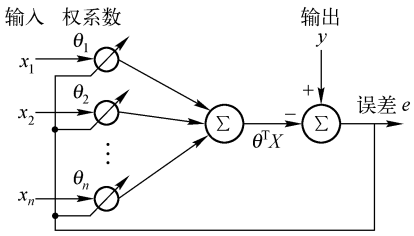



图 8-7　线性权系数模型

考虑进行 $N$ 次观测实验可以得到如下 $N$ 个方程组成的方程组，即

$$y(i)=\theta_1\cdot x_1(i)+\theta_2\cdot x_2(i)+\cdots+\theta_n\cdot x_n(i)$$
$$i=1,\ 2,\ \cdots,\ N$$

如果模型的结构是正确的，测量所得到的数据又不包括任何噪声，那么只要让 $N=n$，求取参数的问题就可简化为求解 $n$ 元一次线性方程组的简单过程。然而，由于事实上量测必然存在噪声，而模型本身和参数估计都不可能准确，因此实际模型可以改写成下列最小二乘法的标准方程：

$$y(k)=\boldsymbol{\varphi}_{(k)}^{\mathrm{T}}\cdot\boldsymbol{\theta}+\varepsilon(k) \tag{8-40}$$

式中，$\boldsymbol{\theta}=[\theta_1, \theta_2, \cdots, \theta_n]^{\mathrm{T}}$ 为 $n$ 维参数向量，$\boldsymbol{\varphi}_{(k)}^{\mathrm{T}}=[x_1, x_2, \cdots, x_n]$ 为 $n$ 维观测向量，$\varepsilon(k)$ 被称为残差，通常被假设为零均值的随机变量。

将描述 $N$ 次试验的 $N$ 个线性方程写成矩阵形式，即

$$\begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(N) \end{bmatrix} = \begin{bmatrix} x_1(1) & x_2(1) & \cdots & x_n(1) \\ x_1(2) & x_2(2) & \cdots & x_n(2) \\ & & \vdots & \\ x_1(N) & x_2(N) & \cdots & x_n(N) \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix} + \begin{bmatrix} \varepsilon(1) \\ \varepsilon(2) \\ \vdots \\ \varepsilon(N) \end{bmatrix} \tag{8-41}$$

这里，设 $N\times n$ 维观测矩阵如下：

$$\boldsymbol{\Phi}=\begin{bmatrix} \boldsymbol{\varphi}^{\mathrm{T}}(1) \\ \boldsymbol{\varphi}^{\mathrm{T}}(2) \\ \vdots \\ \boldsymbol{\varphi}^{\mathrm{T}}(N) \end{bmatrix} = \begin{bmatrix} x_1(1) & x_2(1) & \cdots & x_n(1) \\ x_1(2) & x_2(2) & \cdots & x_n(2) \\ & & \vdots & \\ x_1(N) & x_2(N) & \cdots & x_n(N) \end{bmatrix}$$

$N$ 维输出向量为

$$\boldsymbol{y}=[y(1), y(2), \cdots, y(N)]^{\mathrm{T}}$$

$N$ 维误差向量为

$$\boldsymbol{\varepsilon}=[\varepsilon(1), \varepsilon(2), \cdots, \varepsilon(N)]^{\mathrm{T}}$$

则式(8-41)可以写成紧凑的矩阵方程形式：

$$\boldsymbol{y}=\boldsymbol{\Phi}\cdot\boldsymbol{\theta}+\boldsymbol{\varepsilon} \tag{8-42}$$

由于最小二乘估计以残差平方和最小为目标函数，即要求目标函数

$$J=\sum_{k=1}^{N}\varepsilon^2(k)=\boldsymbol{\varepsilon}^{\mathrm{T}}\cdot\boldsymbol{\varepsilon}=(\boldsymbol{y}-\boldsymbol{\Phi}\cdot\boldsymbol{\theta})^{\mathrm{T}}\cdot(\boldsymbol{y}-\boldsymbol{\Phi}\cdot\theta) \tag{8-43}$$

取最小。于是，参数 $\boldsymbol{\theta}$ 的最小二乘估计值 $\hat{\boldsymbol{\theta}}_{\mathrm{LS}}$ 可采用目标函数 $\boldsymbol{J}$ 对参数求导后取极值的方法推出，即利用 $\boldsymbol{\theta}=\boldsymbol{\theta}_{\mathrm{LS}}$ 时 $\dfrac{\partial J}{\partial\theta}=0$ 的条件，导出：

$$\frac{\partial J}{\partial\theta}=-2\boldsymbol{\Phi}^{\mathrm{T}}\cdot y+2\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\cdot\theta=0$$

从而得到最小二乘解为

$$\hat{\boldsymbol{\theta}}_{\mathrm{LS}}=[\boldsymbol{\Phi}^{\mathrm{T}}\cdot\boldsymbol{\Phi}]^{-1}\cdot\boldsymbol{\Phi}^{\mathrm{T}}\cdot y \tag{8-44}$$

注意，这里观测矩阵 $\boldsymbol{\Phi}$ 的维数为 $(N\times n)$，$\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}$ 则为 $(n\times n)$ 的方阵。$(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi})^{-1}$ 也称 $\boldsymbol{\Phi}$ 的伪逆，且只要其存在，就可以解出 $\hat{\boldsymbol{\theta}}_{\mathrm{LS}}$，实现辨识参数的目的。然而，$\boldsymbol{\Phi}$ 的伪逆存在条件与组成 $\boldsymbol{\Phi}$ 阵的观测数据有关。

上述最小二乘基本算法是一种批量处理的方法。每当得到一些新的数据，都需要从头进行一遍完整的计算，因此需要保留全部数据，从而占用了很多内存容量。同时，由于算法中存在矩阵的求逆运算，在参数数目 $n$ 较大时可能出现一些计算上的问题。另外，上述算法是在假定 $N$ 次测量过程中被估计参数不变，因此不必区分数据的新旧和先后。而对时变系统就有必要更看重最近新得到的数据，使其比旧数据具有更高的加权份量，故可取新的算法目标函数为

$$J_w=\sum_{i=1}^{k}\lambda^{k-i}\varepsilon^2(i)=\boldsymbol{e}^{\mathrm{T}}\cdot W\cdot\boldsymbol{e},\qquad 0<\lambda<0 \tag{8-45}$$

这样，在已经得到第 $k$ 次实验数据时，可以推导出如下具有数据渐消记忆功能的最小二乘估计的递推算法：

$$\gamma(k)=[\lambda+\boldsymbol{\varphi}^{\mathrm{T}}(k)\cdot P(k-1)\cdot\boldsymbol{\varphi}(k)]^{-1}$$
$$\hat{\boldsymbol{\theta}}(k)=\hat{\boldsymbol{\theta}}(k-1)+\gamma(k)\cdot P(k-1)\cdot\boldsymbol{\varphi}(k)\cdot[y(k)-\boldsymbol{\varphi}^{\mathrm{T}}(k)\cdot\hat{\boldsymbol{\theta}}(k-1)]$$
$$P(k)=\frac{1}{\lambda}\cdot[P(k-1)-\gamma(k)\cdot P(k-1)\cdot\boldsymbol{\varphi}(k)\cdot\boldsymbol{\varphi}^{\mathrm{T}}(k)\cdot P(k-1)] \tag{8-46}$$

以下几点需要注意：

（1）估计公式很有道理，$k$ 时刻所得到的参数估计 $\hat{\boldsymbol{\theta}}(k)$ 等于前一次参数估计值 $\hat{\boldsymbol{\theta}}(k-1)$ 与后面修正项之和，而这个修正项由三部分组成：其中被称为新息的预报误差 $y(k)-\boldsymbol{\varphi}^{\mathrm{T}}(k)\cdot\hat{\theta}(k-1)$ 是 $k$ 时刻的测量值与基于 $k-1$ 时刻估计参数得到的预报值之差，它直接决定修正项的大小，当其为 0 时修正项也为 0，因为此时实测值已与预报值相等，表明参数估计已经完成；修正增益 $\gamma(k)$ 用于反映修正程度的大小，由于它是一个标量，因此并不需要进行矩阵求逆的计算；$P(k-1)\phi(k)$ 为几维向量，用于分配修正项在各个参数方向上的作用。

（2）$P(k)$ 是进行第 $k$ 次递推计算时对估计误差的协方差的直接测度，$P(k)$ 取值逐步减少，意味参数误差的协方差阵减小，表明估计趋于更加有效，因此对参数的修正作用也应减少。

（3）如何确定参数初值 $\theta(0)$ 和 $P(0)$ 呢？直接的方法是先取 $m$ 组数据，用成批处理的方法算出 $\theta(m)$ 和 $P(m)$ 后作为初值。另一种常用的方法则是先取初值 $P(0)=\alpha I$，其中 $\alpha$ 为一个很大的正数，如 $\alpha=10^6\sim10^{10}$，则递推 $N$ 步后即可得到实际的 $P(N)$。可以证明：当 $\alpha$ 很大时，$P(N)=[\boldsymbol{\Phi}^{\mathrm{T}}(N)\boldsymbol{\Phi}(N)]^{-1}$。

（4）遗忘因子 $\lambda$ 用于表明过去数据被遗忘的速度。$\lambda$ 取值越小，则算法跟踪参数变化的能力就越强，但同时受到噪声干扰的影响也越严重。通常，$\lambda$ 取值在 0.95～0.99 之间，如 $\lambda=1$，则渐消记忆估计算法退化为普通最小二乘估计。

根据渐消估计的最小二乘参数估计的递推公式[式(8-46)]，可以编写相应的通用函数来实现 RLS 的递推最小二乘参数估计功能(如程序清单 8-3 所示)。

**程序清单 8-3　递推最小二乘的参数估计程序**

```
Public Function RunRLS(ByVal n As Integer, ByVal y As Variant, x() As Double, _
                       q() As Double, P() As Double, ByVal lamda As Variant)
                       As Variant
    Dim xpx#, xpx1#, e#, px#(), i% , j% , k%
    ReDim px#(n): lamda=1#: xpx#=lamda: k% =0
    For i% =0 To n-1
        px#(i%)=0
        For j% =0 To n-1
            px(i%)=px(i%)+P(k%) * x(j%): k% =k% +1
        Next j%
        xpx#=xpx#+x(i%) * px(i%)
    Next i%
    xpx1#=1# /xpx#: e#=y
    For i% =0 To n-1: e#=e#-x(i%) * q(i%): Next i%
```

```
    k% =0
    For i% =0 To n-1
        q(i%)=q(i%)+px(i%) * xpx1# * e#
        For j% =0 To n-1
            If j% >=i% Then
                P(k%)=P(k%)-xpx1 * px#(i%) * px#(j%)
            Else
                P(k%)=P(i% +j% * n)
            End If
            k% =k% +1
        Next j%
    Next i%
    RunRLS=e#
End Function
```

其中，q( )为待求参数值，x( )和 y 为当前时刻的输入和输出值，P( )为中间变量，r 为遗忘因子。给定变量 q、x、y、P、r 的当前值，即可求出对应参数 q 的值和下一时刻中间变量 P 的值。

### 8.3.2 时间序列的 ARMAX 模型

动态系统的一个主要特征是其输出信号不仅取决于系统输入的当前值，也取决于系统输入和输出的过去值。通常随机型差分方程被称为可控的自回归滑动平均模型，并被记为 ARMAX(Auto-Regression Moving Average with External-input)模型。

$$y(k)+a_1y(k-1)+\cdots+a_{n_a}y(k-n_a)=b_0u(k)+b_1u(k-1)+\cdots+b_{n_b}u(k-n_b)+c_0e(k)+c_1e(k-1)+\cdots+c_{n_c}e(k-n_c) \tag{8-47}$$

这里，$u(k)$、$y(k)$分别表示输入和输出在采样时刻 $k$ 时的数值，$e(k)$为白噪声误差。若将自回归滑动平均部分的 $e(k)$统归为有色噪声 $\varepsilon(k)$，模型将退化为 ARX 模型，但仍可写成预报误差的形式：

$$y(k)=-a_1y(k-1)\cdots-a_{n_a}y(k-n_a)+b_0u(k)\cdots+b_{n_b}u(k-n_b)+\varepsilon(k) \tag{8-48}$$

如令

$$\varphi^{\mathrm{T}}(k)=[-y(k-1),\cdots,-y(k-n_a),u(k),u(k-1),u(k-n_b)] \tag{8-49}$$

$$\theta=[a_1,a_2,\cdots a_{n_a},b_0,b_1,b_{n_b}]^{\mathrm{T}}$$

则可直接套用基本的最小二乘模型[式(8-40)]和式(8-42)，并用公式(8-44)和式(8-46)来辨识模型参数 $a_i$ 和 $b_j$。

另外，ARX 模型中输入变量与输出变量之间的关系也可表达为如下 $z$ 传递函数的形式：

$$G(z)=\frac{Y(z)}{U(z)}=\frac{b_0+b_1z^{-1}+\cdots+b_{nb}z^{-nb}}{1+a_1z^{-1}+a_2z^{-2}+\cdots+a_{na}z^{-na}} \tag{8-50}$$

**例 8-3** 尽管实际系统大都为连续系统，但用离散模型来描述往往更为方便。已知系统模型如图 8-8 所示，输入、输出数据已分别放在 $N$ 维数组 $u( )$和 $y( )$中，$d(k)$为随机噪声，试以最小二乘辨识方法辨识一阶模型的参数 $b_1$

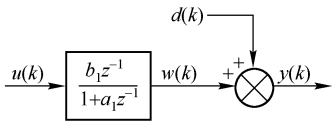



图 8-8 ARX 时序模型

和 $a_1$。

**解**：将此典型 ARX 模型按程序清单 8-4 循环调用 RunRLS 后，最终数组 q( ) 中的内容就是参数 $a_1$、$b_1$ 的辨识结果。

**程序清单 8-4　ARX 时序模型的辨识程序**

```
n% =na+nb+1-nk: Redim P(n% * n%), q(n%)              'n 维参数, P, q 清零
for i% =0 to n% -1: P(i% *n% +i%)=100000#: next i%  '设 P 阵对角元素
q(0)=0 : q (na+1-nk)=b1                              '参数初值 b1 可取直流增益
For k% =0 To N-1                                     '循环处理
    For i% =0 To na-1: x(i%)=-y(k% -1-i%): Next i%       '建立 RLS 数据
    For i% =0 To nb-nk: x(i% +na)=u(k% -i% -nk): Next i%
    Call RunRLS(n% , y(k%), x, 1#, P, q)             '调用 RLS 算法
Next k%
a1=q(0): b1=q(na+1-nk)                               '提取参数辨识结果
```

这里取参数个数 na=1，nb=1；可以再取 nk=1 来避免对已设参数 b0=0 进行不必要的辨识计算。显然，只要设置适当的 na、nb 和 nk，以及相应维数的初值 P 和 q，即可利用程序清单 8-4 所示的代码进行针对更高阶次模型的参数辨识。

### 8.3.3　最优激励信号的选择

考虑到辨识系统的动态特性，应当选择能够对系统产生有效激励作用的输入信号。一般说来希望其频谱(或功率谱)足够宽，足以激励被测系统的频谱模态。$m$ 序列是一种伪随机二位式序列，其相关函数近似 $\delta$ 函数，因而对应的频率特性具有近似白噪声特征的广谱特性。

$m$ 序列广泛用于系统辨识的另一个重要原因是其二进制输出容易生成，可用一种具有异或逻辑反馈的移位寄存器组产生。其工作原理如图 8-9 所示，各触发器依次相接形成基本的移位功能，首级触发器的 D 端输入状态则取自移位寄存器组中某些指定寄存器位的输出反馈信号，并经异或逻辑处理后确定。注意，移位寄存器状态不能出现全 0 的情况，因此 $r$ 级移位寄存器产生的 $m$ 序列长度不能超过 $2^r-1$。究竟需要什么样的反馈逻辑，从哪级寄存器反馈？在此不做细致的推导，而由表 8-3 给出高至 10 级的各种移位寄存器反馈方式，以此提供产生 $m$ 序列的反馈方法。

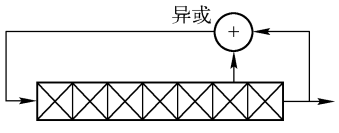



图 8-9　产生 $m$ 序列的工作原理

**表 8-3　产生 $m$ 序列的反馈方法**

| |
|---|
| 4 级反馈选择:(4,3),(4,1) |
| 5 级反馈选择:(5,3),(5,2) |
| 6 级反馈选择:(6,5),(6,1) |
| 7 级反馈选择:(7,6),(7,1) |
| 8 级反馈选择:(8,7,2,1) |
| 9 级反馈选择:(9,4),(9,5) |
| 10 级反馈选择:(10,3),(10,7) |

$m$ 序列也可通过计算机程序直接得到，程序清单 8-5 给出了产生 7 级(127 位长)$m$ 序列的 VB 程序代码，结合表 8-3，不难生成其他长度的 $m$ 序列。

**程序清单 8-5　生成 *m* 序列的 VB 程序**

```
nr=7: i1%=6: i2%=7
w$=String(nr,"1"): nk=2^nr - 1: ReDim d(nk)
For k%=0 To nk-1
    If Mid(w$,i1%,1)=Mid(w$,i2%,1) Then s1$="0" Else s1$="1"
    If s1$="0" Then d(k%)=a Else d(k%)=-a      '输出对称电平
    w$=s1$ & Mid(w$,1,nr-1)                    '相当于寄存器移位后再设置首位状态
Next k%
```

这里，字串 w$ 相当建立一组初值状态为 1 的寄存器，循环体则实现寄存器组的移位动作，首寄存器状态取决于第 i1 位和第 i2 位的状态，它们之间的异或关系可用其状态是否相同来表示，产生的标志 s1$ 作为 w$ 首字符的取值。程序最终产生幅值为±1 的 *m* 序列。

通常生成的 *m* 序列本身幅值不高，看似杂乱而没有规律。但若其相关函数接近 $\delta$ 函数，则对应的功率谱就会接近常数，作为输入激励信号就很理想。实际 *m* 序列的相关函数可表示为

$$R_{uu}(\tau)=\begin{cases}a^2\left(1-\dfrac{N_p+1}{N_p}\cdot\dfrac{|\tau|}{\Delta}\right) & -\Delta\leqslant\tau\leqslant\Delta\\ -\dfrac{a^2}{N_p} & \Delta\leqslant\tau\leqslant(N_p-1)\Delta\end{cases}$$

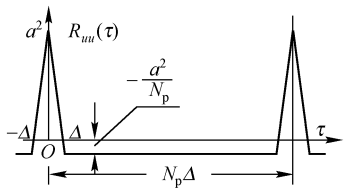



图 8-10　*m* 序列的相关分析

如图 8-10 所示，尽管其相关函数并非真正的 $\delta$ 函数，但也确实相似。特别是当 $N_p$ 增加、$\Delta$ 减小时，波形脉冲变窄、周期拉长、非脉冲段取值趋于 0，整体作用也更接近 $\delta$ 函数。

由于 $R_{uu}(\tau)$ 是一个周期函数，故能展成傅里叶级数，从而可以直接算出实际 *m* 序列的功率谱密度的线谱函数：

$$a_n=\frac{4}{N_p\Delta}\int_0^{N_p\Delta/2}R_{uu}(\tau)\cdot\cos(n\omega_0\tau)\,\mathrm{d}\tau=\frac{N_p+1}{N_p^2}a^2\left(\frac{\sin(n\omega_0\Delta/2)}{n\omega_0\Delta/2}\right)\qquad n=1,2,\cdots$$

$$\frac{a_0}{2}=\frac{2}{N_p\Delta}\int_0^{N_p\Delta/2}R_{uu}(\tau)\,\mathrm{d}\tau=\frac{2}{N_p\Delta}\left[-\frac{a^2}{N_p}\cdot\frac{N_p\Delta}{2}+\frac{N_p+1}{N_p}a^2\cdot\left(\Delta-\frac{\Delta^2}{2\Delta}\right)\right]=\frac{a^2}{N_p^2}$$

(8-51)

这里只关心频谱曲线的相对态势，以确定带宽与 $\Delta$ 的关系。图 8-11 表明-3dB 带宽的频点可以大致定位于

$$\omega_{-3\mathrm{dB}}=\frac{N_p}{3}\omega_0=\frac{N_p}{3}\cdot\frac{2\pi}{N_p\Delta}=\frac{1}{3}\,\frac{2\pi}{\Delta}$$

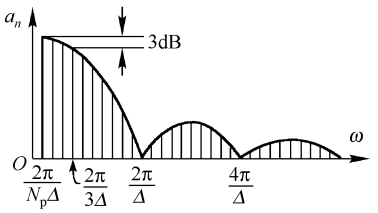



图 8-11　*m* 序列的功率谱密度

因为根据式(8-49)，此处与低频分量相比，衰减度为

$$\left(\frac{\sin(n\omega_0\Delta/2)}{n\omega_0\Delta/2}\right)^2=\left(\frac{\sin(\pi/3)}{\pi/2}\right)^2=\left(\frac{\sqrt{3}/2}{\pi/3}\right)^2$$
$$=0.684\approx0.707$$

综上所述，实际采用 *m* 序列作为系统辨识输入信号时，序列的时钟周期可取

$$\Delta<\frac{1}{3}\cdot\frac{2\pi}{\omega-3\mathrm{dB}}=\frac{1}{3\cdot f_M}\qquad(8-52)$$

又因 $m$ 序列的周期应大于系统过渡过程时间 $T_s$，可取

$$N_p\Delta=(1.2\sim1.5)\cdot T_s \tag{8-53}$$

至于输入信号的二进制电平幅度，则应在不超过系统动态线性范围和工艺允许的条件下，尽量取大一些以提高测量的精度。

**例 8-4**　已知某过渡过程时间 $T_s=0.2s$，截止频率 $f_M=1kHz$，试选择作为辨识激励信号的 $m$ 序列参数(时钟周期 $\Delta$ 和信号周期数 $N_p$)。

**解**：可取 $\Delta=0.3ms$，$N_p=1024$。这样，$N_p\Delta=1024\times0.3ms=0.3s$。

系统辨识输入信号的核心要求是希望其自相关函数接近 $\delta$ 函数，从而信号的功率谱密度成为广谱形式，这样的输入信号不止是 $m$ 序列，诸如白噪声信号、$m3$ 序列和逆重 $m$ 序列等其他一些信号序列也都经常采用。由于逆重 $m$ 序列克服了 $m$ 序列本身存在直流成分的缺点，实现起来也很方便，因此实际辨识中使用得更多一些。建立逆重 $m$ 序列时，应先产生一个普通的 $m$ 序列，再生成一个周期为 2 倍采样的时域方波，然后根据“异或”逻辑方式逐位计算，得到的就是逆重 $m$ 序列。

$u_1(t)=m$ 序列　　1 1 0 1 0 0 1 1 1 0 1 0 0 1

$u_2(t)=$ 方波　　⊕1 0 1 0 1 0 1 0 1 0 1 0 1 0

$u(t)=u_1(t)\oplus u_2(t)$　　0 1 1 1 1 0 0 1 0 0 0 0 1 1

可以证明，逆重 $m$ 序列的自相关函数表达式为

$$R_{uu}(\tau)=\begin{cases}a^2\cdot\left(1-\dfrac{N_p-1}{N_p}\cdot\dfrac{|\tau|}{\Delta}\right) & -\Delta\leqslant\tau\leqslant\Delta\\[2ex](-1)^k\dfrac{a^2}{N_p}\left(\dfrac{2|\tau|}{\Delta}-1\right) & (k-1)\Delta\leqslant\tau\leqslant k\Delta\\[2ex]a^2\cdot\left(\dfrac{N_p-1}{N_p}\cdot\dfrac{|\tau|}{\Delta}-1\right) & N-\Delta\leqslant\tau\leqslant N+\Delta\end{cases}$$

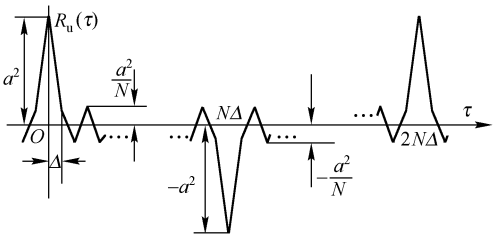



图 8-12　逆重 $m$ 序列的自相关函数波形

其自相关函数波形由图 8-12 给出，可见它没有直流分量，且更接近于 $\delta$ 函数。

总之，利用上述逆重 $m$ 序列生成系统激励信号 $u(k)$后，就可调整幅值后施加到被测系统，并同时测取系统响应。随后套用式(5-48)~式(5-50)建立 ARX 模型，再按最小二乘算法辨识参数。具体电路则可直接利用 6.4 节讨论的数据采集系统。

## 8.4　基于数理统计的可信度分析方法

在自动测试系统和虚拟仪器的工作过程中，常常需要对测试结果进行可信度分析，本节将通过对系统辨识中最小二乘法辨识参数的可信度分析，说明使用数理统计理论的具体方法。

### 8.4.1　最小二乘估计参数的分布情况

对线性方程

$$y=a\cdot x+b+\varepsilon,\ \varepsilon\sim N(0,\sigma^2) \tag{8-54}$$

按式(8-41)~式(8-44)给出的最小二乘估计方法，可由模型方程

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_N & 1 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_N \end{bmatrix}$$

得到参数估计公式：

$$\begin{bmatrix} \hat{a} \\ \hat{b} \end{bmatrix} = \left( \begin{bmatrix} x_1 & x_2 & \cdots & x_N \\ 1 & 1 & \cdots & 1 \end{bmatrix} \cdot \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_N & 1 \end{bmatrix} \right)^{-1} \cdot \begin{bmatrix} x_1 & x_2 & \cdots & x_N \\ 1 & 1 & \cdots & 1 \end{bmatrix} \cdot \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

$$= \frac{1}{N \cdot \sum_{i=1}^{N} x_i^2 - \left( \sum_{i=1}^{N} x_i \right)^2} \cdot \begin{bmatrix} N & \sum_{i=1}^{N} x_i \\ \sum_{i=1}^{N} x_i & \sum_{i=1}^{N} x_i^2 \end{bmatrix} \cdot \begin{bmatrix} \sum_{i=1}^{N} x_i y_i \\ \sum_{i=1}^{N} y_i \end{bmatrix} = \frac{1}{\sum_{i=1}^{N} x_i^2 - n\bar{x}^2} \cdot \begin{bmatrix} \sum_{i=1}^{N} x_i y_i - n\overline{xy} \\ \sum_{i=1}^{N} x_i y_i \cdot (\bar{x} - \bar{y}) \end{bmatrix}$$

即

$$\hat{a} = \frac{\sum_{i=1}^{N} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{N} (x_i - \bar{x})^2} = \frac{\sum_{i=1}^{N} (x_i - \bar{x}) \cdot y_i}{\sum_{i=1}^{N} (x_i - \bar{x})^2} \tag{8-55}$$

$$\hat{b} = \bar{y} - \hat{a} \cdot \bar{x} \tag{8-56}$$

以及 $y$ 的预报值

$$\tilde{y} = \hat{a}x + \hat{b} = \hat{a}x + \bar{y} - \hat{a}\,\bar{x} = \bar{y} + \hat{a} \cdot (x - \bar{x}) \tag{8-57}$$

现在进一步分析所得结果的置信度。由式(8-55)可知 $\hat{a}$ 是样本数据 $y_j$ 的线性组合，可表示为

$$\hat{a} = \sum_{j=1}^{N} [c_j \cdot y_j]$$

式中，$c_j = \dfrac{x_j - \bar{x}}{\sum_{i=1}^{N} (x_i - \bar{x})^2}$，且有 $\sum_{j=1}^{N} c_j = 0$，$\sum_{j=1}^{N} c_j x_j = 1$。

根据式(8-54)的假设，由于 $\varepsilon_i$ 服从零均值的正态分布 $N(0, \sigma^2)$，则有 $y_i$ 服从 $N(a+b \cdot x_i, \sigma^2)$ 的正态分布，于是 $\hat{a}$ 作为 $N$ 个独立随机变量 $y_i$ 的线性组合自然也是正态分布，而且可以求取其数字特征参数(均值和方差)如下：

$$E[\hat{a}] = E\left[ \sum_{j=1}^{N} c_j y_j \right] = E\left[ \sum_{j=1}^{N} c_j (a x_j + b + \varepsilon_j) \right]$$

$$= a \cdot E\Big[\sum_{j=1}^{N} c_j x_j\Big] + b \cdot E\Big[\sum_{j=1}^{N} c_j\Big] + E\Big[\sum_{j=1}^{N} \varepsilon_j\Big] = a$$

可见数 $\hat{a}$ 是 $a$ 的无偏估计，再看方差

$$D[\hat{a}] = D\Big[\sum_{j=1}^{N} c_j y_j\Big] = \sum_{j=1}^{N} c_j^2 D(y_j) = \sum_{j=1}^{N} \left[\frac{x_j - \bar{x}}{\sum_{i=1}^{N}(x_i - \bar{x})^2}\right]^2 \cdot \sigma^2 = \frac{\sigma^2}{\sum_{i=1}^{N}(x_i - \bar{x})^2}$$

这样，对具有正态分布的估计参数 $\hat{a}$ 来说，得到均值和方差就可以建立完整的概率分布描述：

$$\hat{a} \sim N\Big[b,\ \sigma^2 / \sum_{i=1}^{N}(x_i - \bar{x})^2\Big] \tag{8-58}$$

于是，根据标准正态分布的对称性质和双侧 $100\alpha$ 百分点位的定义，可得

$$\mathrm{Prob}\left[\frac{|\hat{a} - a|}{\sigma}\sqrt{\sum_{i=1}^{N}(x_i - \bar{x})^2} > z_{\alpha/2}\right] = \alpha$$

$$\mathrm{Prob}\left[|\hat{a} - a| < z_{\alpha/2} \cdot \sigma / \sqrt{\sum_{i=1}^{N}(x_i - \bar{x})^2}\right] = 1 - \alpha$$

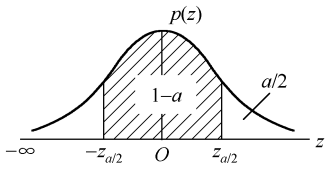



图 8-13　正态分布

如图 8-13 所示，参数 $a$ 在 $(1-\alpha)\times 100\%$ 的置信度下的置信区间为

$$\hat{a} \pm z_{\alpha/2} \cdot \sigma / \sqrt{\sum_{i=1}^{N}(x_i - \bar{x})^2} \tag{8-59}$$

另一方面，由 $\hat{b}=\bar{y}-\hat{a}\cdot\bar{x}$，可知 $\hat{b}$ 也是正态分布，也可求取其数字特征参数：

$$E(\hat{b}) = E[\bar{y}-\hat{a}\cdot\bar{x}] = E[\bar{y}]-\bar{x}\cdot E[\hat{a}] = a\bar{x}+b-\bar{x}\cdot a = b \quad \Rightarrow \hat{b} \text{ 是 } b \text{ 的无偏估计}$$

$$D(\hat{b}) = D[\bar{y} - \hat{a}\cdot\bar{x}] = D[\bar{y}] + \bar{x}^2 \cdot D[\hat{a}] - 2\bar{x}\cdot \mathrm{Cov}(\bar{y}, \hat{a})$$

$$= \frac{\sigma^2}{N} + \bar{x}^2 \cdot \frac{\sigma^2}{\sum_{i=1}^{N}(x_i - \bar{x})^2} - 2\bar{x}\cdot \sum_{j=1}^{n}\left[\frac{x_j - \bar{x}}{\sum_{i=1}^{N}(x_i - \bar{x})^2} \cdot \mathrm{Cov}(\bar{y}, y_j)\right]$$

$$= \sigma^2 \cdot \left[\frac{1}{N} + \frac{\bar{x}^2}{\sum_{i=1}^{N}(x_i - \bar{x})^2}\right]$$

于是最终得到参数 $\hat{b}$ 的分布特性如下：

$$\hat{b} \sim N\left\{b,\ \sigma^2 \cdot \left[\frac{1}{N} + \frac{\bar{x}^2}{\sum_{i=1}^{N}(x_i - \bar{x})^2}\right]\right\} \tag{8-60}$$

这样，从上述分析可以看出估计参数的波动大小（$\hat{a}$ 和 $\hat{b}$ 的方差）不仅取决于观测值的方差 $\sigma^2$，而且还与变量 $x_i$ 的离散程度有关。因此，如果 $x_i$ 是可控的变量，则在安排试验时应考虑以下几点：

（1）$x_i$ 的取值越分散，$\hat{a}$ 和 $\hat{b}$ 的方差越小。

（2）$\bar{x}=0$，可使 $\hat{b}$ 的方差达到最小。

(3) 加大试验次数 $n$，可减少 $\hat{b}$ 的方差。

### 8.4.2 方差的估计

上述推出估计参数 $\hat{a}$ 和 $\hat{b}$ 的方差表达式中的 $\sigma^2$ 是样本偏差 $\varepsilon_i$ 和样本数据 $y_i$ 的方差，实际上是无法得到其真值的，因此也需要采用参数估计的方法。这里 $\sigma^2$ 的估计借助于残差平方和 SSD，可令

$$\hat{\sigma}^2=\frac{\mathrm{SSD}}{n-2}$$

其中，

$$\mathrm{SSD}=\sum_{i=1}^{N}(y_i-\tilde{y}_i)^2=\sum_{i=1}^{N}(y_i-\hat{a}+\hat{b}x_i)^2$$

这样，

$$\hat{\sigma}^2=\frac{\mathrm{SSD}}{N-2}=\frac{1}{N-2}\sum_{i=1}^{N}(y_i-\tilde{y})^2=\frac{1}{N-2}\sum_{i=1}^{N}(y_i-\hat{a}x_i-\hat{b})^2 \tag{8-61}$$

可以证明，$\hat{\sigma}^2$ 是 $\sigma^2$ 的无偏估计，因此有理由用计算得到的估计值 $\hat{\sigma}^2$ 来代替真值 $\sigma^2$。

### 8.4.3 估计参数的置信区间

在分析了 $\hat{a}$、$\hat{b}$ 和 $\hat{\sigma}^2$ 等估计量的统计特性后，就可以讨论模型结构的假设是否合理、分析关键参数估计值的置信区间有多大了。

对一组同分布的正态随机变量 $Z\sim N(0_N,\sigma^2 I_N)$，其中 $z_i\sim N(0,\sigma^2)$，则由 $Z/\sigma^2$ 服从标准正态分布的事实可知，其平方和服从 Chi 平方分布(如图 8-14 所示)，即

$$\frac{\mathrm{SSD}}{\sigma^2}\sim\chi^2(N-2)$$

这里分布的自由度为 $N-2$，是由计算其数学期望

$$E\left[\frac{\mathrm{SSD}}{\sigma^2}\right]=\frac{1}{\sigma^2}\sum_{i=3}^{n}E[z_i^2]=\frac{1}{\sigma^2}\sum_{i=3}^{n}\{D(z_i)+[E(z_i)]^2\}=N-2$$

得到的。进而考虑式(8-61)，可得

$$\frac{(N-2)\hat{\sigma}^2}{\sigma^2}\sim\chi^2(N-2) \tag{8-62}$$

再由式(8-58)$\hat{b}$ 的正态分布函数，可以推出

$$(\hat{a}-a)/\sqrt{\frac{\sigma^2}{\sum_{i=1}^{n}(x_i-\bar{x})^2}}\sim N(0,1) \tag{8-63}$$

于是，根据 $t$ 分布的定义可知，式(8-62)和式(8-63)组合形成的一个统计量可以具有 $t$ 分布函数特性(如图 8-15 所示)。

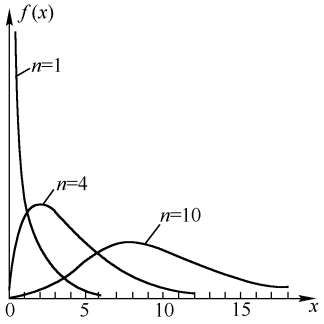



图 8-14　Chi 平方分布函数

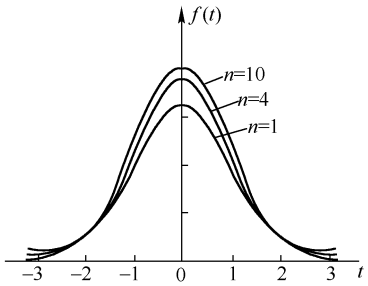



图 8-15　$t$ 分布函数

$$\frac{(\hat{a}-a)\Big/\sqrt{\dfrac{\sigma^2}{\sum_{i=1}^{n}(x_i-\bar{x})^2}}}{\sqrt{\dfrac{(N-2)\hat{\sigma}^2}{\sigma^2}\Big/(N-2)}}\sim t(n-2)$$

即

$$\frac{\hat{a}-a}{\hat{\sigma}}\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\sim t(N-2) \tag{8-64}$$

与前面利用服从正态分布推出置信区间的方法相似，可得

$$\mathrm{Prob}\left[\,|\hat{a}-a|>t_{\alpha/2}(N-2)\cdot\hat{\sigma}\Big/\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\,\right]$$

可见参数 $a$ 在($\alpha\times100\%$)置信度下的置信区间为

$$\hat{a}\pm t_{\alpha/2}(N-2)\cdot\hat{\sigma}\Big/\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2} \tag{8-65}$$

然而，此刻区间中所含的参数已经没有不可计算的 $\sigma^2$ 了。

**例 8-5**　已知某放大器的参数测试，测得一一对应的输入信号电压 $x$ 和输出信号电压 $y$ 如表 8-4 左面两列所示。求 $y$ 关于 $x$ 的线性回归方程参数(放大倍数 $a$ 和偏置电压 $b$)，说明回归效果是否显著，以及置信度 $1-\alpha=95\%$ 时的预测区间。

**表 8-4　例 8-5 的数据及其计算列表**

| $x$ | $y$ | $x^2$ | $y^2$ | $xy$ |
|---|---|---|---|---|
| 0 | 3.024 | 0.000 | 9.145 | 0.000 |
| 0.5 | 4.035 | 0.250 | 16.281 | 2.017 |
| 1.0 | 5.078 | 1.000 | 25.786 | 5.078 |
| 1.5 | 5.945 | 2.250 | 35.343 | 8.918 |
| 2.0 | 7.006 | 4.000 | 49.084 | 14.012 |
| 2.5 | 8.003 | 6.250 | 64.048 | 20.008 |
| 3.0 | 9.038 | 9.000 | 81.685 | 27.114 |
| 3.5 | 9.936 | 12.250 | 98.724 | 34.776 |

| | | | | |
|---|---|---|---|---|
| 4.0 | 11.022 | 16.000 | 121.484 | 44.088 |
| 4.5 | 11.921 | 20.250 | 142.110 | 53.644 |
| $\Sigma=22.5$ | 75.01 | 71.250 | 643.691 | 209.655 |

**解**：先将所算中间结果 $x^2$、$y^2$、$xy$ 填入表中右面三列，再分别计算 $\Sigma x$、$\Sigma y$、$\Sigma x^2$、$\Sigma y^2$、$\Sigma xy$，并将结果填入表中最后一行。由样本数 $N=10$，可得

$$\bar{x}=\frac{1}{N}\cdot\sum_{i=1}^{N}x_i=\frac{22.5}{10}=2.25,\bar{y}=\frac{1}{N}\cdot\sum_{i=1}^{N}y_i=\frac{75.01}{10}=7.5$$

$$\sum_{i=1}^{N}(x_i-\bar{x})^2=\sum_{i=1}^{N}x_i^2-\frac{1}{N}\Big(\sum_{i=1}^{N}x_i\Big)^2=71.25-\frac{22.5^2}{10}=20.625$$

$$\sum_{i=1}^{N}(y_i-\bar{y})^2=\sum_{i=1}^{N}y_i^2-\frac{1}{N}\Big(\sum_{i=1}^{N}y_i\Big)^2=643.691-\frac{75.01^2}{10}=81.071$$

$$\sum_{i=1}^{N}(x_i-\bar{x})\cdot y_i=\sum_{i=1}^{N}x_iy_i-\frac{1}{N}\Big(\sum_{i=1}^{N}x_i\Big)\Big(\sum_{i=1}^{N}y_i\Big)=209.655-\frac{22.5\times75.01}{10}=40.887$$

根据式(8-54)，可得

$$\hat{a}=\sum_{i=1}^{N}(x_i-\bar{x})y_i/\sum_{i=1}^{N}(x_i-\bar{x})^2=\frac{40.887}{20.625}=1.982$$

$$\hat{b}=\bar{y}-\hat{a}\cdot\bar{x}=7.5-1.98\cdot2.25=3.04$$

且有回归直线

$$\tilde{y}=\hat{a}\cdot x+\hat{b}=1.98\cdot x+3.04$$

假设 $H_0$：$a=0$，表示 $x$ 和 $y$ 之间并非线性关系。于是，根据式(8-64)，可以建立符合 $t$ 分布的统计量：

$$\frac{|\hat{a}|}{\hat{\sigma}}\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\sim t(n-2)$$

即

$$\mathrm{Prob}\left[\frac{|\hat{a}|}{\hat{\sigma}}\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}>t_{\alpha/2}(n-2)\right]=\alpha$$

若取 $\alpha=0.05$，则当假设 $H_0$ 成立时统计量

$$\frac{|\hat{a}|}{\hat{\sigma}}\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}$$

大于 $t_{\alpha/2}(n-1)$ 的概率只有 5%。现在算得

$$\hat{\sigma}^2=\frac{\mathrm{SSD}}{N-2}=\frac{1}{N-2}\sum_{i=1}^{N}(y_i-\tilde{y})^2=\frac{1}{N-2}\Big[\sum_{i=1}^{N}(y_i-\bar{y})^2-a^2\sum_{i=1}^{N}(x_i-\bar{x})^2\Big]$$

$$=\frac{1}{10-2}(81.071-1.982^2\times20.625)=0.0021$$

则使统计量

$$\frac{|\hat{a}|}{\hat{\sigma}}\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}=1.98\cdot\sqrt{\frac{20.625}{0.0021}}=196.385$$

真的大于

$$t_{\alpha/2}(N-2)=t_{0.025}(8)=2.306$$

因此，鉴于居然出现了小概率事件，假设 $H_0$ 应被拒绝，故 $a=0$ 不成立，表明线性回归的效果是显著有效的。另外，若回归效果不显著有效，则无非出于如下原因：

（1）除 $x$ 外还有其他不可忽略的因素影响 $y$。

（2）$x$ 与 $y$ 之间不是线性关系。

（3）$x$ 与 $y$ 无关。

根据式(8-65)，可计算参数 $a$(放大倍数)的置信区间，如图 8-16所示。

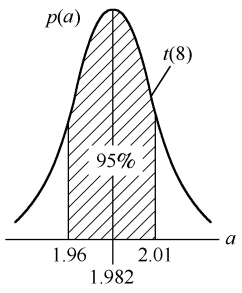



图 8-16　$a$ 的置信区间

$$a=\hat{a}\pm t_{\alpha/2}(n-2)\cdot\hat{\sigma}/\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}$$

$$=1.982\pm 2.306\cdot\sqrt{0.0021/20.625}=1.982\pm 0.0233$$

即在可信度 95%的条件下，$a$ 的置信区间为 1.96~2.01。

本节以估计参数的可信度分析为例，所使用的数理统计方法自然可以用于自动测试过程中其他数据的分析和处理。

## 8.5　大型旋转机械的状态监测与故障诊断

生产设备的运行是否正常、其工作状态是否安全，始终都是企业生产的头等大事。大型旋转机械，诸如发电行业的大型汽轮发电机组、石化行业的大型透平压缩机组、冶金行业的大型轧机和风机，均为企业中最关键的生产设备，因此其状态监测和故障诊断具有更为重要的地位。随着传感器、电子学、数字信号处理、人工智能等现代信息科学技术的发展，以计算机为核心的自动测试系统已经逐步代替了眼观耳听的原始方法，实时监测系统工作状态并诊断故障，已经成为十分明显的普遍趋势。本节将在简明介绍 200MW 与 300MW 汽轮发电机组的状态监测和故障诊断系统的基础上，说明此类自动测试系统的设计方法。

### 8.5.1　200MW 和 300MW 汽轮发电机组转子的状态监测

200MW 和 300MW 汽轮发电机组是火力发电厂的重要设备，经过多年运行后故障发生的可能性正在逐步增加。这些机组的主轴转子在 3000 转/分下高速运行，始终伴随明显的振动。由于这些振动信号的变化常常体现系统初期故障的特征，且其监测具有实时、无损和相对简便的特点，因此以机组振动信号为主进行全面、实时的自动测试，就成为大型机组状态监测和故障诊断最普遍和最有效的方法。不仅可以比较方便地了解系统的工作状态，预防某些重大事故的发生，也为系统大修机制从定期计划方式向实际状态方式的过渡提供了依据，从而可以明显减少不必要的停机时间，直接获取重大经济效益。图 8-17 为汽轮发电机组的现场照片。

振动监测系统的总体框图如图 8-18 所示，分传感器、监测仪表、数据采集器、计算机监测分析、故障诊断专家系统等几个部分。

作为典型的大型旋转机械系统，这种基于振动信号的汽轮机组状态监测包括轴振、瓦振、偏心、对中、串轴、差涨、壳涨，以及转速、温度、阀位等多种类型物理量的测量，测量参

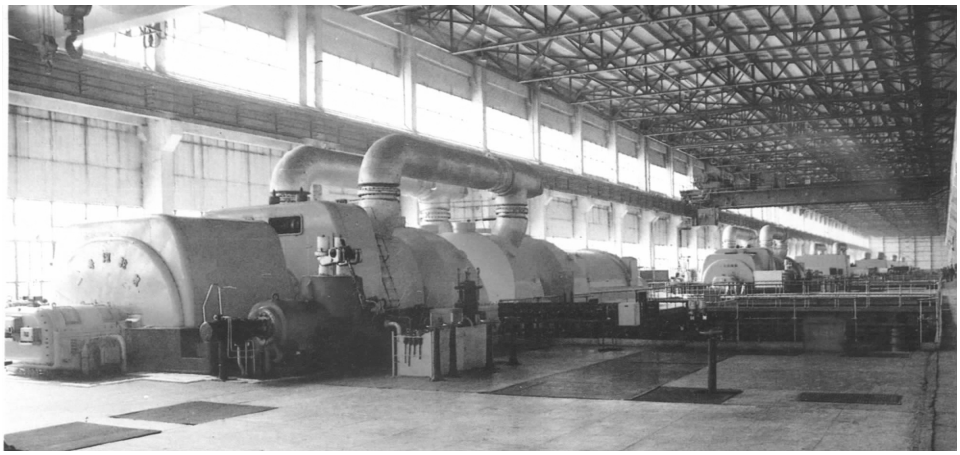


图 8-17 汽轮发电机组的现场照片

数则分动态和静态两种形式。

轴振主要是指主轴转子在多组支撑轴瓦附近的径向振动情况，要求展示它们的时基波形、频谱特性和轴心轨迹状况，通常以正交方向安装在轴瓦附近的各对电涡流传感器直接监测其位移情况；瓦振则指轴瓦本身的振动情况，通常用安装在轴瓦处的加速度传感器进行监测。轴振和瓦振的监测以动态测量为主，需要记录其振动波形，并通过对原始波形数据进行处理，得到各种必要的特征参数和图示曲线。轴振测量的直流分量能表示轴的偏心位置，用来监测径向轴承的碰摩故障。偏心度可表示转轴静态时的挠度，对其监视可防止弯轴事故。对中表示轴系各转子之间的连接情况，对中不良会造成较大的振动。

串轴是指转子的轴向位移，这是系统最重要的测量参数。其目的是确保转子止推环与静子止推轴承之间的间隙，以避免轴向碰摩一类灾难性事故的发生。为了监视轴向位置，通常需要安装两个电涡流传感器，以提高测量的可靠性。

差涨是指转子与静子在高温工质作用下受热膨胀比率不同时所引起的轴向相对位移变化。如果机组启动时差涨过大，就可能发生轴向碰摩的事故。壳涨是指以地基作为固定基准，机壳本身的膨胀。对一些大型机组，除了需要监视差涨外，还应监视壳涨。

监测系统还包括转速、温度等测量项目。机组运行过程(特别是启/停机过程)，各监测参数与转速的关系常常能够反映系统实际工作状态和故障特征，因此需要建立反映主轴实时转速的测速功能。温度测量主要针对轴瓦中巴氏合金衬套的温度，它们往往直接标志系统润滑条件的工作状态，因此也需要配置专门的传感器。

机组转子的振动测量以振幅和频率测量为主，并有相应的工业标准。但为了配合故障诊断和平衡校验的需要，必须对转子键相位记号提供测量的方法，以满足相位测试的要求。这里，系统通过电涡流传感器对转子端头齿盘的测量，同时获取了转子的频率和相位。由于高速运转的转子在多处轴瓦的支撑下产生的振动会形成一定的振型和模态，因此不同转速、不同位置处振动的幅值、相角以及共振频率等信息，都为故障诊断的分析提供了十分丰富和重要的依据。

传感器的测量信号通常需要适配器电路将其调理到标准的电压或电流，然后通过电缆直接传送到远离机组的操作室监测仪表处进行展示。通常监测仪表与传感器成套配置，并且构

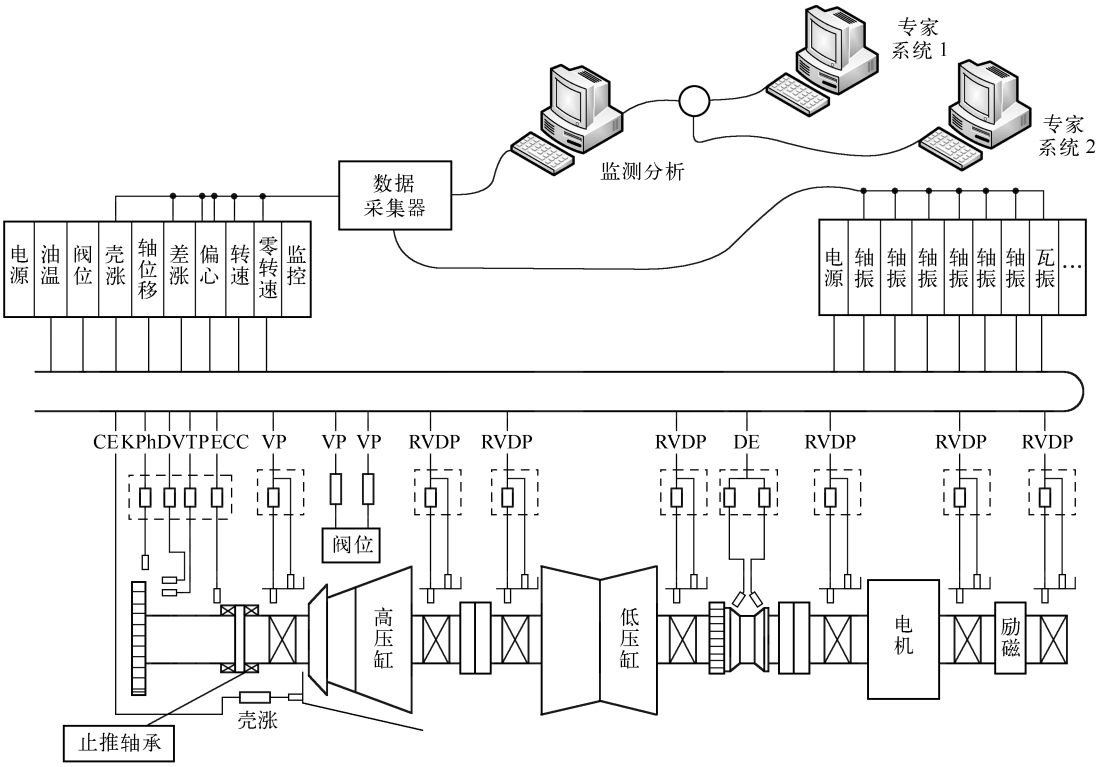



图 8-18　汽轮发电机组状态监测系统的配置示意图

成独立的监测仪表体系。监测仪表除了支持面板显示外，还可支持各种信号的数据接口。

进一步建立的计算机监测和故障诊断系统则需专门配置数据采集器和计算机，以及用于显示、打印和网络连接的终端设备和通信设备，这些设备可以分别配制在控制室操作屏后和主操作台上。数据采集器的输入信号可与监测仪表共用相同的电缆信号，也可取自监测仪表处理后的标准信号。

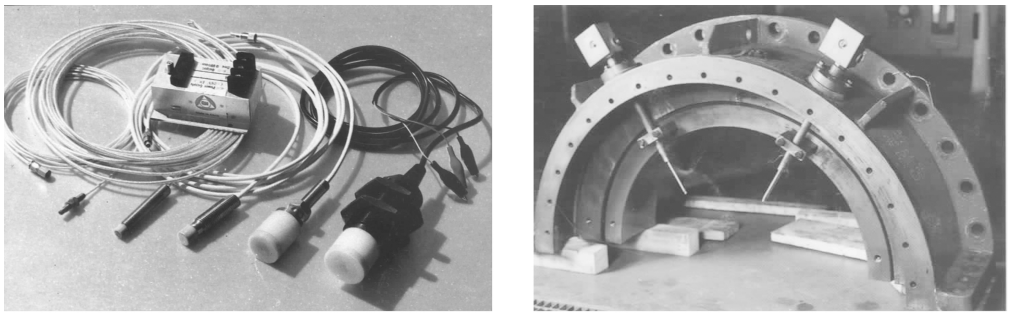
计算机系统的监控分析软件为操作者和工程师提供了现场实时监控操作的完整界面，并配置了便于分析诊断的各种显示图表，从功能上完全可以看做代替监测仪表的虚拟仪器。然而，出于可靠性考虑，通常电厂仍将保留监测仪表，将其作为传感器系统的原配基准，依靠其实现最为重要的保护功能。而将不断更新的计算机监测诊断系统作为日常监测和实时操作的主要手段，配置在操作台处的微计算机中。

考虑旋转机械故障诊断系统颇为复杂的机理和需要不同专家经验的实际情况，采用专家系统的诊断方式，可以利用计算机网络技术实现位于不同位置的计算机终端之间的资源共享，使不同专家都能利用现场提供的数据，也使现场可以得到不同专家的分析结果和技术支持。

### 8.5.2　传感器和监测仪表的配置

通常振动信号的测量可以直接选用位置传感器来测量振动的位移特征值(峰-峰值或有效值)，也可以选用速度传感器和加速度传感器，并通过适当的积分运算来间接得到振动的位移特征值。速度传感器体积相对较大，需要一定的空间安装，其因自身有源而信号较强，但频率范围有限，适于测量中频参数。加速度传感器一般采用压电结构，体积小，安装方便，

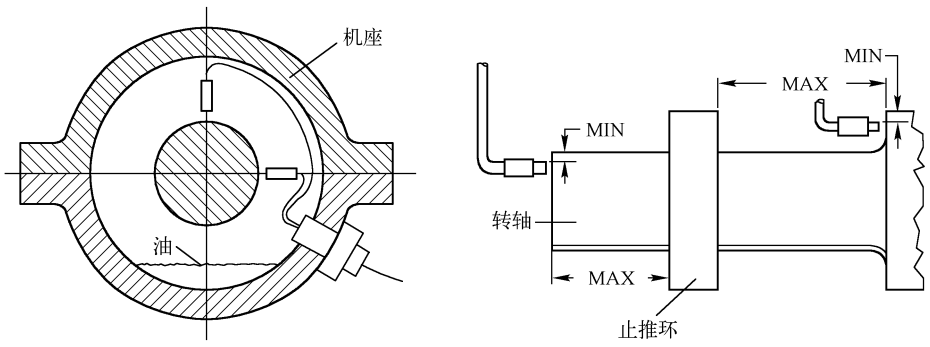
可通过螺扣直接固定在被测物体的壳体上，但有时可能过于灵敏，容易引起噪声，且输出信号需要积分处理。位移传感器尽管可以采用差动变压器、磁电、电容、磁阻等多种方式，但最常用的是电涡流传感器。图 8-19 展示了电涡流传感器的实物外观及其在机组中的安装方式。



(a) 电涡流传感器系列产品　　(b) 电涡流传感器的安装

图 8-19　电涡流传感器及其安装

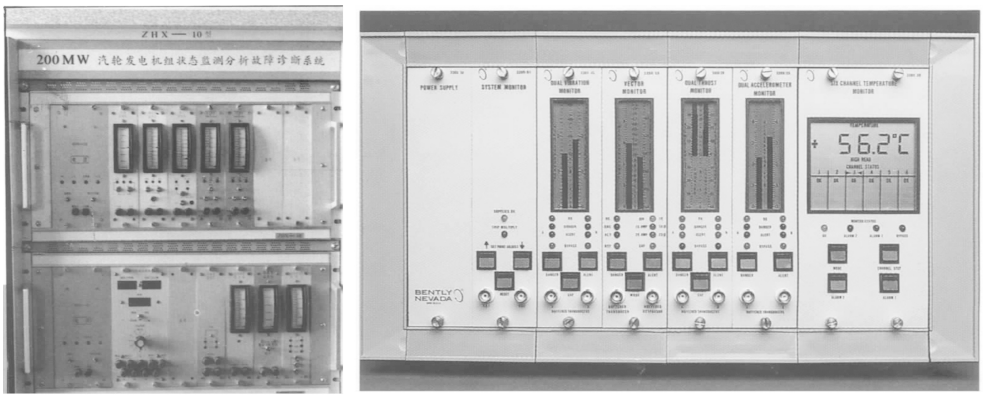
电涡流传感器灵敏度高、线性范围大、频率特性宽、体积小、抗干扰能力强，在大型旋转机械的振动信号监测系统中应用最广。电涡流传感器通过探头线圈发出电磁场，在被测导体表面产生电涡流回路，当探头与被测导体位置变化时就引起其内部谐振参数变化，从而将探头与导体之间的位置信息以输出电压的形式表现出来(通过适配器电路)。电涡流传感器为非接触式传感器，便于安装。考虑汽轮发电机组转子状态监测的具体需要，图 8-20 展示了电涡流传感器在径向和轴向测量时的不同安装方法。




(a) 径向正交安装　　(b) 轴向安装

图 8-20　电涡流传感器及其安装

振动传感器需要专门的监测仪表配合方能实际应用。早期的监测仪表采用典型的模拟电路，通过模拟表头的指针直接展示测试结果(如图 8-21(a)所示)。而新型的监测仪表则已数字化，但考虑到使用者的操作习惯，仍以模拟光柱的高低表示监测数据的大小(如图 8-21(b)所示)，同时配置预警、危险等标志。另外，监测仪表应有统一的状态监视机制，以确保其本身功能的完备性。




(a) 模拟型监测仪表　　(b) 数字型监测仪表

图 8-21　监测仪表的外形结构

监测仪表是系统的眼睛，必须稳定可靠，且应独立配置和运行，不可简单地用数字采集系统代替。监控仪表还可设置保护动作的触发操作条件，可根据其对测试数据的处理结果产生不同的输出信号，直接触发系统的保护设备。

监测仪表的开发通常采用背板机箱的方式，以便统一提供电源、冷却和各种通信接口。机箱中每个插槽模件对应一台仪表，用于相关通道进行独立的监测。这种方式与 VXI 和 PXI 的工作方式十分相似，自然可以考虑采用自动测试系统的典型模式实现。只是监测仪表与传感器、机组系统联系紧密，且责任重大，因此实际应用需要得到有关部门的认可。监测仪表也有采用分立式结构的便携设备，可用少量传感器独立进行故障分析和诊断。

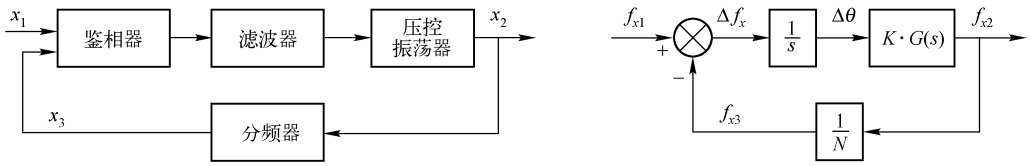
### 8.5.3　数据采集器的工作特点

汽轮发电机组转子正常运行的转速为 3000 转/分，相当于 50Hz 的基波信号。根据采样定理，考虑 8 倍的高次谐波和 8 组轴振测量需要的 16 通道动态测试，数据采集器可参考例 8-1的原理来计算采样频率：

$$f_s > 2 \cdot f_b = 2 \cdot 50 \cdot 8 \cdot 16 = 128\text{kHz} \tag{8-66}$$

尽管这样的指标要求对普通数据采集器来说并不高，但考虑实际转子的转速并非常数，特别是在启/停机时更是大幅度持续变化，这样普通数据采集器的定时采样机制就不能确保对输入信号的整周期采样。当使用傅里叶变换进行频谱分析时，就会产生频谱“泄漏”的影响。为了避免这类问题，旋转机械状态监测和故障诊断系统的数据采集器需要特别设计，确保实际采样频率随转子主频的变化而变化。为此，需要保持对输入信号的整周期采样，同时信号输入通道的抗混叠低通滤波器带宽特性也应随主频的变化而改变。为了适应上述特点，这里采用一种锁相环倍频的方法，图 8-22 是实现这一方案的原理框图。

图中锁相环倍频电路的输入信号为被测转子信号的同步脉冲信号，对正弦周期信号常采用过零检测的方法来获取，这里则直接使用现场专门配置的键相位信号。鉴相器用于识别被测信号与反馈信号的相位差，其输出通过滤波处理后控制压控振荡器的输入。压控振荡器是一个电压-频率的转换装置，根据其输入电压的大小来确定输出频率的高低。在锁相环进入锁定状态后，鉴相器输出呈一常数，压控振荡器的输入电压和输出频率均保持不变，即鉴相器测得的相位差 $\Delta\theta$ 被锁定为常数，于是相位差的微分(频率 $\Delta f$)为 0，因此可知被测信号与




(a) 锁相环原理　　(b) 闭环控制框图

图 8-22　锁相环倍频的原理框图

反馈信号的频率相等($f_{x1}=f_{x3}$)。由于反馈信号是由锁相环输出信号分频得到的($f_{x3}=f_{x2}/N$)，因此必有锁相环输出信号与其输入信号之间的整数倍频关系($f_{x2}=f_{x2}\cdot N$)。即转子每转一周，必有 $N$ 组等间隔采样时间的测量数据。

锁相环电路的稳态特性以其跟踪某种典型输入信号的误差来表示，系统的型则指开环传递函数中所含积分环节的数量。由于从频率到相位已经包含了一个积分环节，因此系统至少是Ⅰ型系统，若滤波器再含 1 个积分环节，则系统将成为Ⅱ型环节。型值越高，则系统的稳态无差度越高，但可能带来稳定性问题。锁相环电路的稳定性和动态响应本质上应由其闭环传递函数特征方程的根决定，但可通过开环频率特性进行分析和设计，故需妥善配置滤波器，同时也需考虑鉴相器和压控振荡器的线性工作区范围。

实际电路通常采用锁相环集成芯片，厂家往往给出芯片外部阻容元件的选择公式，配合调试一般也能得到满意的结果。图 8-23 是一种典型锁相环电路的原理图，包括由锁相环集成电路 MC14046B、分频器 CD4020 构成的锁相环倍频跟踪电路，以及分别由电压比较器 LM339 和单稳触发器 CD4528 构成的输入、输出电路。

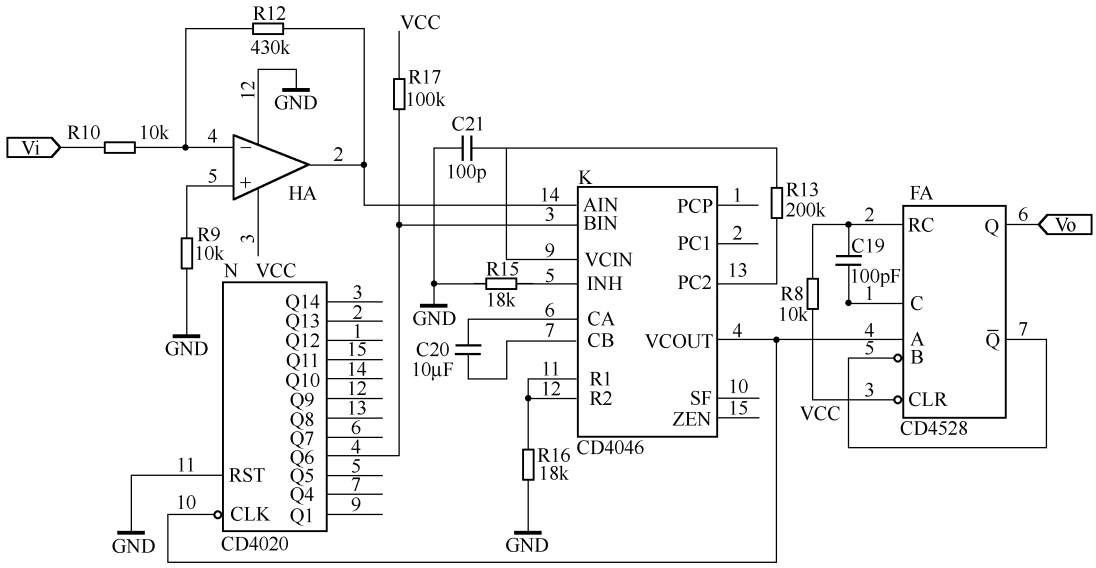



图 8-23　锁相环倍频电路

这里，核心集成电路芯片 CD4046/MC14046B 包括鉴相器和 VCO 压控振荡器两部分，但封装时共地。根据有关文献的分析，这种结构在其 VCO 和鉴相器之间产生干扰，使 VCO 输入处混入输入信号与反馈信号之间的比较频率成分，当输出频率较高时就会增加比较频率中

的寄生成分。为此，实际电路可用两个 CD4046 芯片分别作为鉴相器和 VCO 工作，从而使系统频谱特性得到明显改善。

锁相环倍频电路输出的脉冲信号可以用于开关电容滤波器的时钟信号，从而可以实现其系统监测通道滤波时间常数随机组转子转速变化而自动调整的自适应工作方式。

### 8.5.4　计算机监测分析软件

计算机监测分析软件作为典型的自动测试系统软件，一方面担负对前端各数据采集器的监控操作和数据采集任务，另一方面则要完成对读回的各机组原始数据进行分析、整理、存储和显示。同时，也是计算机诊断专家系统的现场数据提供者。它的工作是承上启下，将整个系统联系在一起。作为面向用户的窗口，监测分析软件不仅关系整个监测过程的质量，也直接影响进一步诊断的可靠运行，因此确定合理的设计原则无疑十分重要。

监测分析软件通常可以分为前台和后台两个过程。前者通过从内存或数据文件中提取监测分析所需要的各类数据，通过图形或表格的形式显示在屏幕上供用户监视和分析；后者则定时与数据采集器进行通信，并将通信传来的采集数据存放在监测计算机的数据缓冲区或文件中。前、后台任务相互独立，在 Windows 操作系统下可以作为两个应用程序独立运行。另外，计算机监测分析软件也将分析、整理后的数据形成新的标准化数据，提供给更上层的诊断专家系统使用。

来自数据采集器的数据分静态参数和动态参数两类，前者用于监测状态，每隔 1~4s 进行一次，以保证系统的实时性(保护动作则由监测仪表直接处理)。动态参数主要用于分析和故障诊断，定时收回可程控的数据采集记录。每个记录包括 16 个整周期波形，滤波前后各 8 个，每周期采样 32 点，共计 512 点，可以从中获取具有丰富频谱的原始振动信息。

监测分析过程通过系统、机组和测点 3 级主菜单操作完成，系统级主要为用户提供整个系统的运行状态、报警记录和事件记录等报告。机组级针对一个指定机组为用户提供能够监测的全部内容，测点级则针对某一测点进行更为细致的分析。

用户总是通过计算机软件的操作界面来使用系统。考虑机组级主操作界面的使用频率最高，希望其包括的信息量大，能够一目了然地使用户了解机组的总体状况和当前的实时状态。如图 8-24 所示，系统主窗体布局就是按照这个原则，将机组转子结构和传感器配置按轴瓦位置形象地展示出来，同时也将各主要振动参数一一列出，而将其他相对次要的分析内容放在子窗口中，通过下拉菜单选项进行切换。

如图 8-25 和图 8-26 所示，多通道轴心轨迹和多通道仪表式棒图显示反映整个机组的状态，将其显示在同一屏上可以相互比对，从中发现一些问题，如轴弯曲等现象。轴心轨迹的显示还分滤波和非滤波两种情况：前者观察得更为清晰，后者则包括更为丰富的信息。

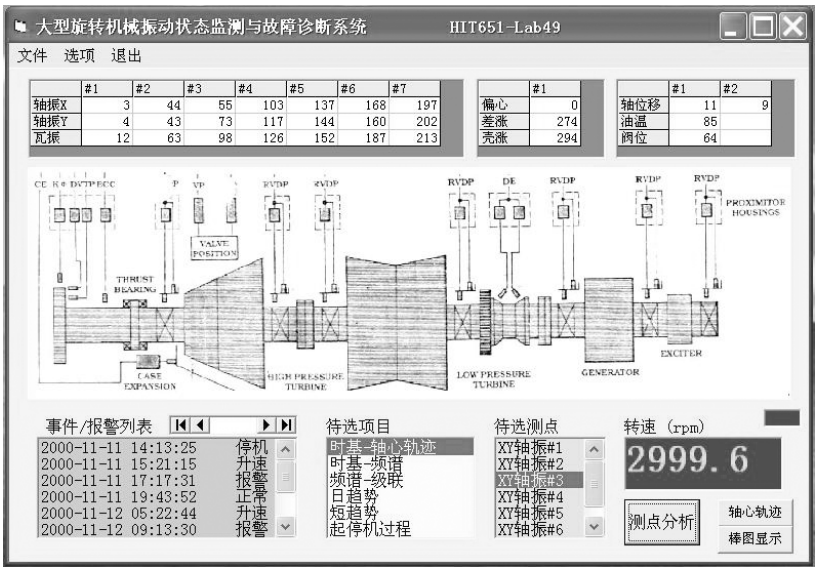



图 8-24　系统主操作界面

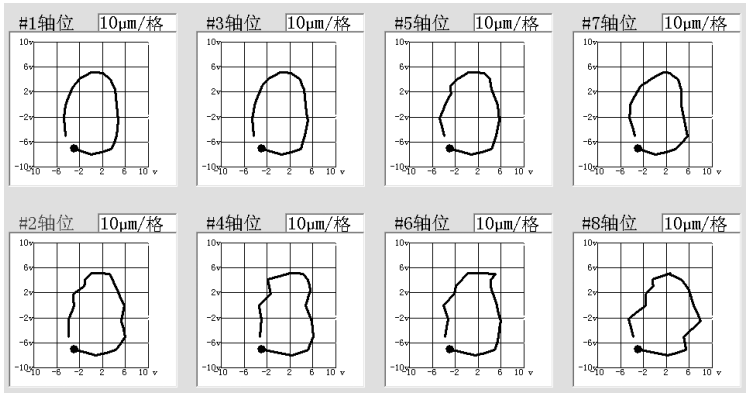



图 8-25　轴心轨迹的显示

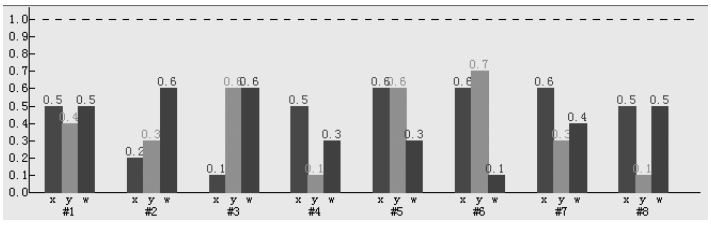



图 8-26　系统静态参数的棒图显示

各测点的测试内容则更为丰富，诸如时基波形图、频谱分析图、不同时间级联的三维频谱分析图，以不同时段为周期的趋势图，还有反映启/停机过程的瞬时伯德图、极坐标图和不同转速级联的三维频谱分析图等。有经验的专业工作者可以从中获取很多有用的信息，为诊断故障提供重要线索，也为大修停机提供依据。图 8-27～图 8-32 给出了这些分析图表的若干示例，它们的建立和应用常常会用到本章前面几节介绍的数据处理方法，如频谱分析的

FFT 算法、振动趋势的建模预测、统计数据的置信区间分析等。

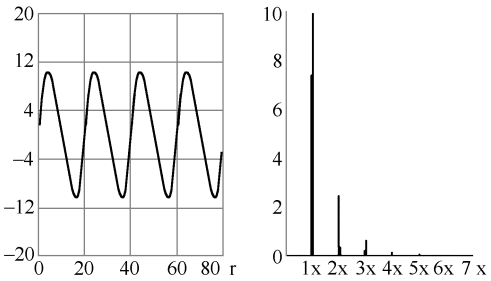



图 8-27　时基波形和频谱分析图

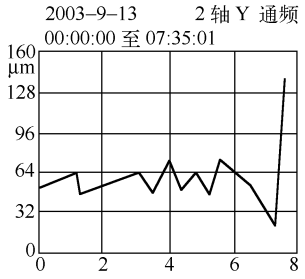



图 8-28　趋势分析图

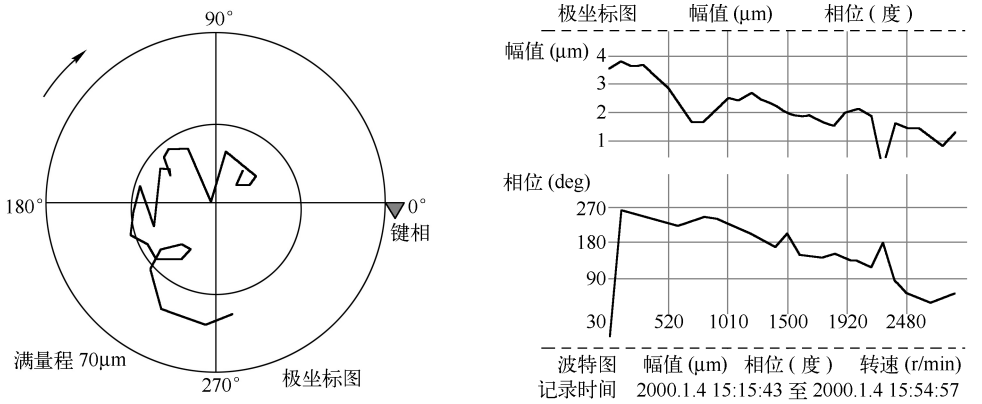



图 8-29　启/停机过程的伯德图和极坐标图

通常自动测试系统和虚拟仪器的核心均为 PC 主机，因此自然应当充分发挥 PC 强大的数据存储、网络通信、多媒体、通用接口和软件开发等各种固有功能。随着 PC 配置的硬盘容量不断增大，对系统原先规定的数据存储文件的数量限制已不复存在。这些数据文件包括来自所有传感器通道的原始静态/动态数据，以及经过信号处理和分析整理后的图表文件，还包括反映现场实际工作状态的各种报警和事件报告。将历史记录的数据以数据库的形式存放也是很好的方式，可以利用数据库丰富的查询功能，比较容易地从繁杂的记录中找出需要时段和特征的原始数据。

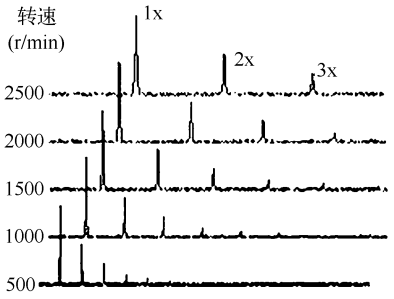



图 8-30 启/停机过程的三维级联图

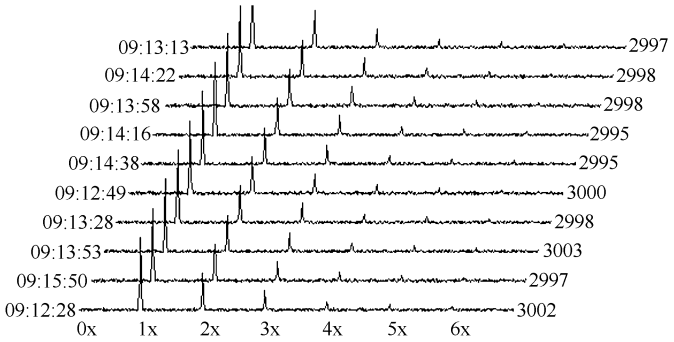



图 8-31 不同时刻的频谱瀑布图

| 时间 | 机组号 | 通道号 | 事件 |
|---|---|---|---|
| 2002年12月10日15时12分23秒 | 三号机 | #1瓦Y | 停机状态 |
| 2002年12月10日17时18分04秒 | 四号机 | #2瓦Y | 降速状态 |
| 2002年12月12日10时19分07秒 | 三号机 | #4瓦Y | 降速状态 |
| 2002年12月12日10时23分02秒 | 三号机 | #8轴X | 超过报警值 |
| 2002年12月12日14时55分03秒 | 三号机 | #1轴X | 正常状态 |
| 2002年12月12日14时56分31秒 | 三号机 | #1轴X | 升速状态 |
| 2002年12月12日15时02分22秒 | 三号机 | #1轴X | 正常状态 |
| 2002年12月12日15时05分40秒 | 三号机 | #1轴X | 升速状态 |
| 2002年12月12日15时05分48秒 | 三号机 | #1轴X | 超过报警值 |

图 8-32 系统报警事件列表

鉴于 PC 可在网络条件下运行来自各方的分析软件，这里讨论的计算机监测分析软件的数据存储方式没有可能和必要普遍支持它们的直接运行，但必须提供文本文件，以便为用户自行编写文本转换程序提供必要的条件。

另外，为了克服监测分析软件和诊断专家系统的调试过程中缺少实际数据来验证算法和实时运行的可靠性，通常需要建立仿真调试程序，根据实际情况定时产生一些合成的数据来仿真模拟仪表静态数据、动态波形、特征参数以及启/停机过程曲线等，这样既可以脱离实际现场环境独立进行软件的调试，也有利于软件本身可靠性的试验和考核。另外，当系统调整某些硬件设备，如监测仪表、数据采集器时，往往希望能够直接观察原始数据，以便进行直观的比对，因此也需要提供相应的专用调试工具。组态软件专门用于编辑系统的组态参数，可以灵活地改变系统的配置、运行参数等选项，从而在不重新编译的条件下，为不同现场建立不同工作条件和显示内容的运行系统。

### 8.5.5 故障诊断的模糊推理与专家系统简介

模糊推理作为故障诊断的重要方法，其基本思想如图 8-33 所示。可将故障本身看做病症，而将监测得到的信息看做病兆。从机理上讲，病症与病兆之间的相互关系可以通过一个模糊模型矩阵来描述，矩阵的元素即为这种对应模糊关系的隶属度。但从故障诊断的角度看，则是一个从病兆反推病症的逆过程。

如果用条件概率来建立模糊关系的隶属度，则模糊关系矩阵 $R$ 的元素 $p(B/A_i)$ 可看做以病症 $A_i$ 为原因引发病兆 $B$ 现象的概率，这样若以 $p(A_i)$ 表示病症 $A_i$ 本身出现的概率，以 $p(B)$ 表示病兆本身出现的概率，则根据求取逆概率的 Bayes 公式：

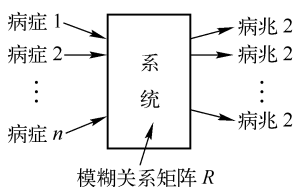



图 8-33　故障诊断的模糊推理

$$p(A_i/B)=\frac{p(A_i)\cdot p(B/A_i)}{p(B)}=\frac{p(A_i)\cdot p(B/A_i)}{\sum_j p(A_j)\cdot p(B/A_j)}$$

就可得到病兆 $B$ 发生条件下病症 $A_i$ 可能发生的概率 $p(A_i/B)$。这样，考虑几种单故障发生的实际情况，可取 $i=1, 2, \cdots, n$，并以取得最大概率值 $p(A_i/B)$ 的病症 $A_i$ 作为故障诊断的结论。

故障诊断的真正难点在于 $R$ 本身的建立也不容易，对诸如汽轮发电机组这样复杂的系统，单纯的机理分析恐怕不能解决问题，试验的风险又过大。因此，很多时候往往需要借助经验，如转子不平衡会引起振动变量的一倍频分量增高、不对中则导致二倍频分量变大等。

进一步讲，若把模糊矩阵看成规则，则整个故障诊断系统就可借助专家系统来解决。

广义的专家系统是指通过计算机保存和推广专家的经验知识，使一般操作者也能像专家那样处理日常事务和决策，从而取代专家重复性脑力劳动的计算机软件。但通常信息领域所说的专家系统则特指那些基于知识库的决策机制，其与传统应用程序的基本差别在于将推理与知识分开的运行方式。

传统应用程序以“数据结构+算法”的模式解决问题，本质上已将知识体现在程序代码之中。这种方法的问题在于知识被程序代码固定，不能随时改变。如果需要改变或添加新的知识，只能通过程序员重新编写程序，因此不能适应复杂的诊断技术要求不断补充知识的场合。与此相反，专家系统中只有推理机程序需要根据一定的知识结构事先建好，而知识库作为领域知识的集合，以数据文件的形式存在，允许用户独立地进行补充、修改和更新。而且对于具有模糊推理机制的推理机，还允许相互矛盾的知识同时加入知识库，共同作用完成决策。

知识库中的知识必须以规范的形式表示。最常用的知识表示方式是产生式规则，具有与人类思想方式接近的特点，其基本形式如下：

IF<条件 1><条件 2>…THEN<结论 1><结论 2>…

当条件满足时，规则就激发结论。规则的条件部分可以是多个条件事实的逻辑组合，而规则的结论部分也可以是多个结论事实。

通常，一个以规则为基础、以问题求解为中心的专家系统主要包括下述几个组成部分，各部分之间的相互关系如图 8-34 所示。

专家系统的推理过程有正向推理、反向推理和正反混合推理之分。正向推理从加载用户输入的初始事实到运行事实库开始，在知识库中寻找能与之匹配的规则，找到后则将该规则

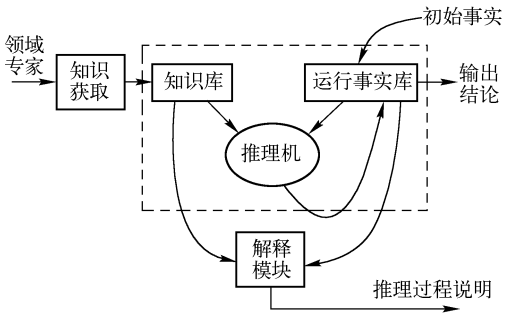



图 8-34 专家系统的结构

的结论部分作为新的事实，加入运行事实库中再继续推理，直至运行事实库中的内容全部处理完毕，便从推出的事实中选择结论输出。反向推理则从规则的结论出发，先选定一个目标结论，然后在知识库中查找能导出该目标结论的规则集，若其中的某条规则条件与运行事实库中的内容匹配，则激发该规则并作为新的目标，继续推理过程，直到运行事实库中不再具有可激发的内容，所存的条件事实即为整个推理过程的结论。由于最终存在的这些条件事实可能不止一个，故实际工作应根据可信度指标进行推理结论的取舍。故障诊断属于从病兆反推病症的过程，因此在建立病症到病兆的规则后，用反向推理方法解决问题较为方便。正反向混合推理集中了正向和反向的推理特点，它首先根据原始信息数据通过反向推理帮助系统提出假设性结论，然后运用正向推理进一步寻找支持该假设的证据。

实际专家系统在推理完成后，通常需要提供解释机制，说明层层推理的具体过程，使用户对结论的产生得到可以信赖的依据。解释接口负责回答用户提出的各种问题，就推理路线、推理结论给出必要的解释，使用户了解推理过程，并以用户容易理解的外部表示形式显示给用户。

知识获取模块将专家的经验知识转化为计算机可利用的知识库规则，同时也负责知识库规则的修改、删除和更新，并对知识库的完整性和一致性进行维护。知识获取模块使领域专家可以直接修改知识库而不必了解知识库中知识的计算机表示方法、知识库结构实现等细节问题。另外，为了方便知识库中规则的输入，专家系统应有相应的规则输入机制，通常采用字典方式，可由用户首先自行定义专用的短语，并在其后能够借助计算机鼠标和键盘方便地选取。知识(规则)在知识库中以代码的方式存储，而以文本的方式在终端上显示。

专家系统运行后，初始事实和结论也可以通过计算机程序动态地输入和输出，从而实现诊断与应用系统的交互作用，这种工作方式特别适用于自动测试与虚拟仪器的应用。

与传统程序相比，专家系统的优越性体现在：1）系统的可维护性好，易于修改和扩充；2）更加适合处理模糊性的、经验性的问题；3）能解释所得结论的过程。

与人类专家相比，专家系统的优越性体现在：1）使用费用低廉；2）不受外界环境和情绪的影响；3）可安装在任何地方，尤其是不适于人工作业的恶劣环境；4）格式化知识表达

便于人们更系统地总结经验，也便于对这些知识的修改、完善和使用。

专家系统按照解决问题的性质可分为很多类型，目前自动测试领域主要采用诊断型和设计型专家系统。测试领域的控制系统软件一般由实时数据采集、故障诊断、人机接口、通信、解释等多个子系统构成，基于知识库的决策器往往只是整个测试系统的一部分，对诸如测试树的生成、故障块的判断等问题起作用。我们并不期望专家系统处于主控地位，而只是力图发挥其适于推理的决策机制，建立“嵌入式”专家系统。这样，用户可在一个应用程序中嵌入多个专家系统，形成多专家系统协同工作的局面。

实际旋转机械故障诊断专家系统的建立需要领域专家的参与，而且必然涉及很多汽轮机组和转子动力学的专业知识，故此处不再深入讨论。以上介绍只是希望说明自动测试与虚拟仪器在处理诸如故障诊断一类复杂问题时，不仅应当充分利用计算机强大的分析计算功能，也应利用计算机处理规则推理的专家系统功能。

实际专家系统的开发可以借助某些专家系统开发工具(如 Clips 等一些软件内核)采取调用动态连接库函数的方法来实现推理过程。可将专家系统机制嵌入到应用程序中，使其仅在自动测试过程中最适于其发挥作用的地方运行工作，并不代替其他适于传统软件的工作方式。

## 8.6　用于心电监测的虚拟仪器开发

心脑血管疾病是最常见的疾病之一，心电图是十分重要的诊断心脏疾病的手段。针对心脏病人需要长期观察病情和监护的实际需要，本节根据虚拟仪器思想设计的心电测试系统具有功能灵活、成本低廉的特点，可以借助一台普通 PC 了解自身心电的异常情况，展示了虚拟仪器在日常生活中可能产生的广泛影响。

### 8.6.1　心电测试原理

每分钟 60 次左右的心脏收缩由右心房上窦房结产生的微小电肌脉动所控制。常见的心电频率在 100Hz 以内，能量主要集中在 17Hz 附近，幅度小于 5mV。心电电极阻抗较大，通常在几百千欧以上。心脏收缩前，首先发生电位变化。心电变化由心脏的起搏点——窦房结开始，经传导系统传至心室，最后到达心肌，引起肌肉的收缩。心脏兴奋活动的综合性电位变化可通过体液传播到人体的表面。如果在体表放置两个电极，分别用导线连接到心电图机(即精密的电流计)两端，便可按照心脏脉动的时序，将体表两点间的电位差记录下来，形成一条连续的曲线，这就是心电图。

心电图可以反映心脏综合性电位变化的产生、传导和消失过程，正常心电图包括 P、QRS 和 T 三个波形，如图 8-35 所示，其中，P 波表示心房去极化，QRS 波群表示心室去极化，T 波表示心室复极化。所谓心搏的一个周期，是由窦房结发出电脉动以渐进波的方式传导至左、右心房，造成左、右心房的收缩(如图中 P 部所示)；电脉动传到房室结后停滞约 0.1s(如图中 P-Q 过渡部所示)，以便让血液充分流至心室；接着电脉动经由传导组织继续传

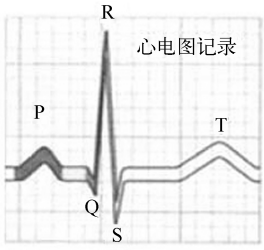


图 8-35　心搏的记录

至左、右心室(如图中 Q 部所示),造成左、右心室收缩(如图中 R 部所示);在此一连串的电活动之后心脏暂时静止,心室等待再极化以恢复其负电状态(如图中 T 部所示),从而完成一次心搏。心室去极化与再极化现象分别为图中 Q、R、S 与 T 部,而心房却仅有去极化的 P 部,因为心房再极化现象波形小,且多半淹没在 Q、R、S 的复合波中,不易察觉。

专业心电图仪中用心电导联来连接人体和测试电路。临床常用的导联方式有肢体导联和胸前导联,肢体导联又有标准导联和加压单极肢体导联之分。为了记录心电,将探测电极安置于相隔一定距离的体表两点构成一个导联,两点的连线代表连轴方向。临床中广泛使用标准十二导联系统,分别记为Ⅰ、Ⅱ、Ⅲ三个标准导联,aVR、aVL、aVF 三个加压导联,以及 V1-V6 六个胸极导联。其中双极导联Ⅰ、Ⅱ、Ⅲ主要反映左、右手,以及左腿中任意两电极间的电压差;其他单极导联由无关电极和探查电极组成,用于测取某点电位。除了标准的十二导联心电图机外,心电监护仪仅在胸前位置采集模拟信号,虽然难以分析心肌缺血(如心绞痛和急性心梗),但使用简便,且能监护心律失常(如房颤、室颤、室速等),因此也在临床上得到广泛的应用。



图 8-36 一次性心电极贴片

为使心电测试系统简便可行,这里采用三导联右腿驱动。使用长为 1m 的三芯屏蔽线。在导线一端配上与心电极贴片匹配的摁扣,这样即可方便地进行心电极贴片的更换,又能满足接触卫生和仪器精度的要求。这种一次性心电极贴片如图 8-36 所示。

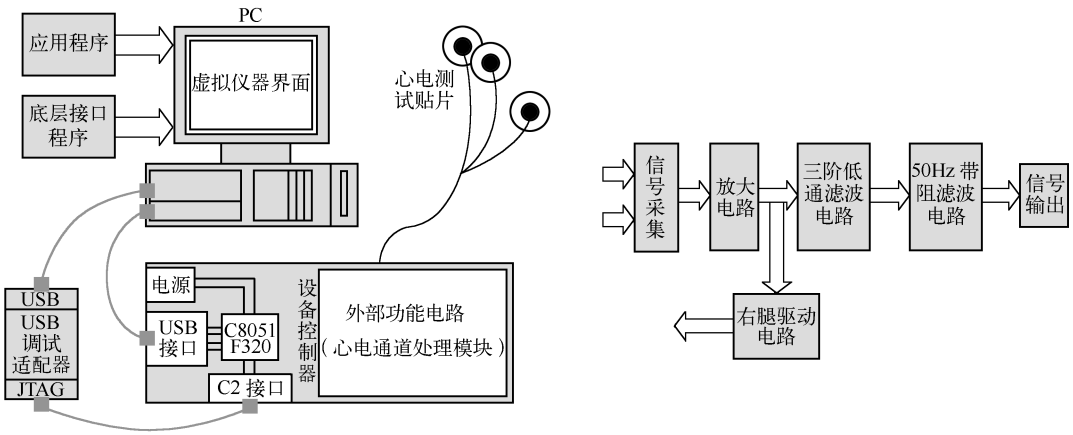
### 8.6.2 虚拟仪器的结构设计

心电监测系统遵循虚拟仪器的设计思想,将仪器的操作、波形的展示、数据的存储和通信都交给 PC 完成,而将前置放大及其通道处理电路做成一个硬件模块,通过基于 SOC 单片机的 USB 设备控制器实现模拟信号的数据采集,再将数据送到 PC 处理。系统总体结构如图 8-37(a)所示,其中通道处理模块如图 8-37(b)所示,由前置放大器、共模信号抑制电路(右腿驱动电路)、低通滤波器和工频陷波电路等几个部分构成。

### 8.6.3 基于 SOC 的 USB 设备控制器

USB 设备控制器以 Silab 公司的 SOC 芯片 C8051-F320 为核心,能够通过 USB 实现与主机的直接通信。SOC 体积小、功能强,能够实现定时中断、A/D 转换、PWM 输出、并口读/写,以及 $I^2C$、SPI、UART 传输等多种接口操作。Silab 公司提供了用于编写固件程序的开发环境,可通过 C2 型调试适配器将固件程序下载到芯片 F320 中,并能进行非侵入式的全速在线调试。芯片 F320 内含可编程振荡器,频率为 12MHz,精度为±1.5%,可以作为系统时钟。F320 中还包含一个 3.3V 的稳压器,可为外部设备提供稳定电源。

如图 8-38 所示,芯片 F320 的外部引脚 VBUS、GND、D+与 D-直接连接 USB 接口,可实现 SOC 与 PC 主机之间的 USB 通信。REGIN 为 5V 电源输入,VDD 为 3.3V 稳压输出。C2 型调试适配器的接口为一个 10 针插座,可由 VDD 供电,C2CK 与 C2D 则为调试适配器的接口信号。为了指示 SOC 的工作状态,可配置发光二极管与 F320 的 P0.4 引脚连接,并




(a) 总体结构　　(b) 通道组成框图

图 8-37　心电测试系统结构

在固件程序代码的关键处通过设置端口电平来控制发光二极管的亮灭时间，从而显示工作过程。

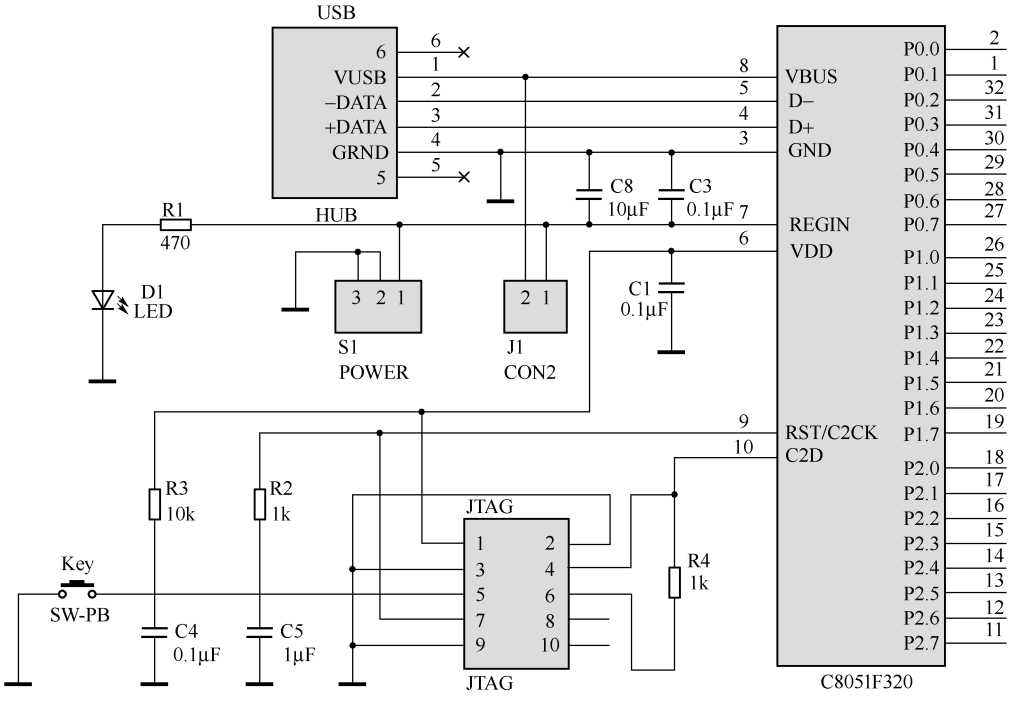



图 8-38　SOC 为核心的微控制器模块

F320 的引脚大多可以用于 A/D 转换和 I/O 操作，有些还有特殊的接口功能，这些信号连同电源 REGIN 与 GND 一起从一个 20 针插座引出，从而形成可以实现各种不同应用功能的通用接口。这里，心电前置通道处理电路就是以此方式与 USB 设备控制器相连的。

### 8.6.4 前置放大电路的构建

心电前置通道电路的主要特点是信号十分微弱，且在获取生物电信号的同时存在很强的干扰，包括电极移动引起的基线漂移(一般小于 1Hz)、交流电源的工频干扰(50Hz)、肌电效应引起的干扰(200Hz 以上)等。其中，电源工频干扰主要以共模形式出现，幅值可达几伏甚至几十伏。为此心电前置通道电路需要配置专门的前置放大器和滤波器。

前置放大器主要解决人机接口问题，如图 8-39 所示，应能满足下述要求：

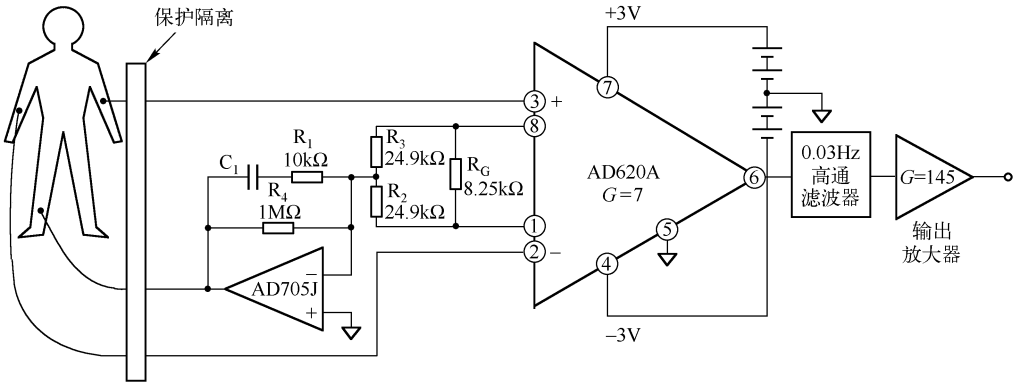



图 8-39 前置放大电路

(1) 高输入阻抗。被提取的心电信号是内阻高且不稳定的微弱信号，为了减少信号源内阻的影响，必须提高放大器输入阻抗。通常信号源内阻为 100kΩ 左右，故放大器输入阻抗应大于 1MΩ。

(2) 高共模抑制比 CMRR。人体所携带的工频干扰以及所测参数以外生理作用的干扰均为共模干扰，故前置级须采用共模抑制比高的差动放大形式。

(3) 低噪声、低漂移，应对信号源影响小，输出稳定。

(4) 右腿体表驱动电路是为克服 50Hz 共模干扰、提高共模抑制比而专门设计的。其做法是从前置放大级中取出共模电压，经驱动电路倒相放大后再施加到右腿体表，故称右腿驱动。通常心电检测时空间电场在人体产生的共模干扰非常严重，而使用右腿驱动电路可以形成以人体为相加点的共模电压并联负反馈，从而能够有效地抑制共模干扰。

这里选择低功耗、高精度的仪表放大器 AD620 作为前置放大器的核心芯片。AD620 只用一个外部电阻就能实现放大倍数为 1～1000 的增益调整。AD620 电源范围宽(±2.3V 至 ±18V)、体积小、功耗低(最大供电电流仅为 1.3mA)。图 8-40 给出了 AD620 的外部引脚图和内部原理图。

由图 8-40 可见，三极管 $Q_1$ 和 $Q_2$ 提供双极差分输入。通过 $Q_1$-$A_1$-$R_1$ 和 $Q_2$-$A_2$-$R_2$ 的环路反馈，保持 $Q_1$、$Q_2$ 集电极电流为常量，输入电压相当于加在外接电阻 $R_G$ 的两端，从输入到 $A_1$、$A_2$ 输出的差分放大倍数为 $G=(R_1+R_2)/R_G+1$。由 $A_3$ 组成的单位增益减法器消除了共模成分，并产生了一个与 REF 引脚电位有关的单路输出。内部增益电阻 $R_1$ 和 $R_2$ 被精确地确定为 24.7kΩ，使放大器增益可由 $R_G$ 确定：

$$G=\frac{49.4\text{k}\Omega}{R_G}+1 \tag{8-67}$$

倘若取 $R_G=100\Omega$，则放大倍数 $G=500$。

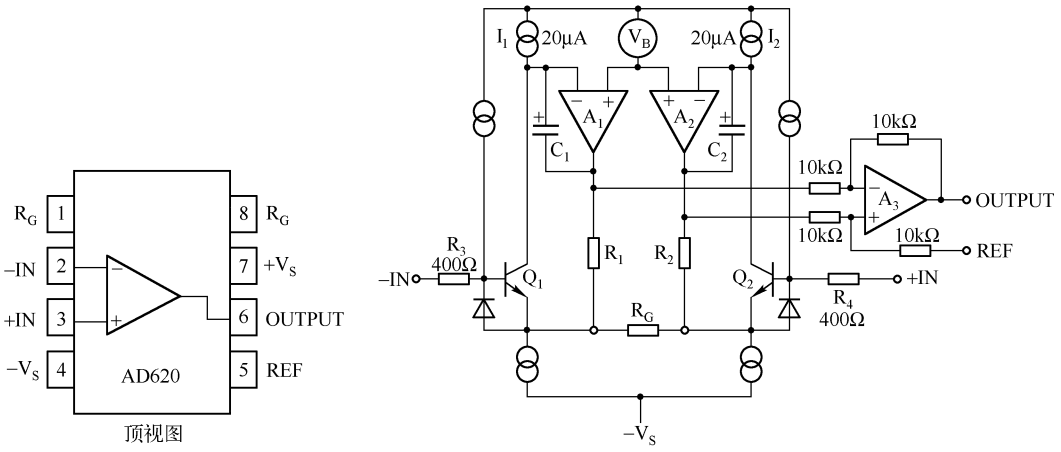



图 8-40　AD620 引脚图和内部结构

为了滤除频率高于 100Hz 的干扰信号，这里采用本书第 2 章所述的 Bach 方法建立一个三阶巴特沃斯低通滤波器。为了对付工频 50Hz 的干扰，则可采用 RC 双 T 原理的带阻滤波器(也称陷波电路)。具体电路如图 8-41 所示。

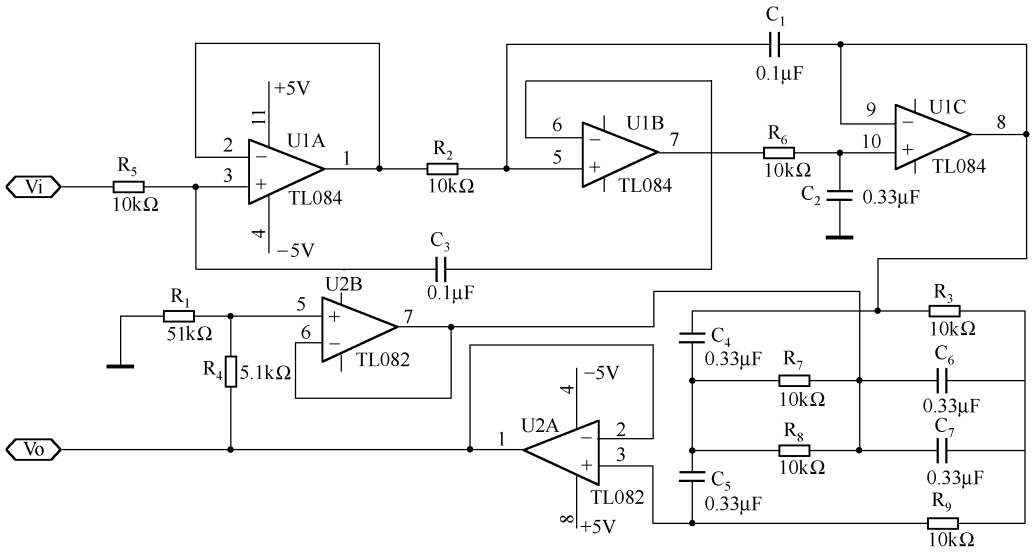



图 8-41　心电监测通道的滤波器电路

若取 Bach 低通滤波器的截止频率为 $\omega_c=100\text{Hz}$，则按 2.2.6 节所述，可取

$$R_2=R_3=10\text{k}\Omega, R_1=10\text{k}\Omega, C_1=C_2=0.1\mu\text{F}, C_3=0.33\mu\text{F}$$

而对 RC 双 T 带阻滤波器，其特征参数包括中心频率 $\omega_0$ 和品质因数 $Q_0$。

根据有关文献给出的一种参数配置方法：$\omega_0$ 取决于不对称系数 $\alpha$ 和电路中 $R_1$、$C_1$ 的值。

而 $Q_0$ 只由系数 $\alpha$ 决定。

当 $\alpha=1$ 时(对称网络)，可取 $C_1=C_2=C$、$C_3=2C$、$R_1=R_2=R$、$R_3=R/2$，此时电路的 $Q=0.25$。

当 $\alpha>1$ 时(不对称网络)，通频带更窄，陷频特性更陡，选频特性更理想。这里，取 $C_5=C_6=C_7=C_8=0.33\mu F$，$R_{11}=R_{12}=R_{13}=R_{14}=10k\Omega$，$R_{15}=5.1k\Omega$，$R_{16}=51k\Omega$。由于电容值偏小，故用可变电阻 $R_{13}$进行微调。

### 8.6.5　PC 应用软件的开发

心电测试系统的基本任务是展示心电图。同时，也可通过 PC 主机直接存储、分析、网络传输心电数据，以求在更高层次应用心电测试得到的数据信息。心电显示的主操作界面如图 8-42 所示，下面仅以心电波形的动态显示实现过程为例，说明虚拟仪器的面板开发方法。

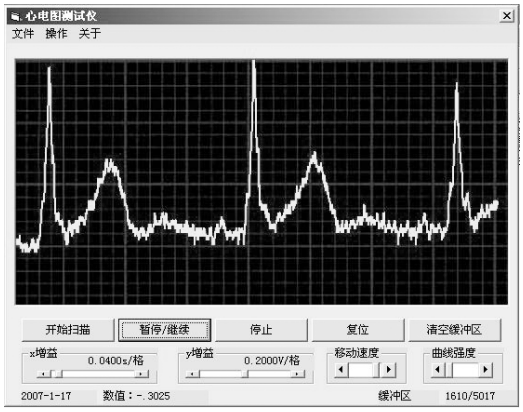



图 8-42　心电信号波形的显示

为在 VB 环境下实现心电波形的动态显示，可利用 Picture 控件建立一个限制可视范围的窗口。再在其中安排两个体现心电波形的 Picture 控件首尾相接，一面同时向左步进移动，一面将新的数据增绘到右面控件图片向左移动后在窗口右端展现的空位。当左侧控件图片完全移出可视窗口后可清除其图片波形内容，并将其置于另一控件图片的右侧。如此循环，使原波形向左移动，新数据则从右端加入显示过程，从而形成心电波形自左至右的连续展示。

心电波形显示过程中所用的参数被设置为 X 增益、Y 增益、扫描频率和曲线强度等选项，并用面板中几个滚动条控件进行设置。其中 X 增益用以确定波形时间轴的显示尺度，Y 增益表示垂直方向坐标显示的电压幅度，并将每格表示的时间间隔和电压值分别显示在对应的滚动条上方。扫描频率是指从数据存储变量中提取数据的速率。通常可使其与数据采集功能读取心电记录的过程同步；也可加快速度以便能够较快地寻找和浏览已有的历史记录；或放慢速度，甚至停止移动，以便更仔细地分析所关注的局部数据。曲线强度只是通过简单地调整显示波形的宽度，用以突出图片显示的曲线部分。

心电波形动态显示的具体程序代码如程序清单 8-6 所示，这里将其核心部分做成通用的类函数，可为其他相似目的的动态数据显示任务所调用。

**程序清单 8-6　动态数据的波形显示**

```
'CS49_CvMove 类
```

```
Dim Picture1 As PictureBox, Picture2(0 To 1) As PictureBox
Dim rDpy As Single, rDpx As Single, xSp As Integer
Dim Px As Long, iP1 As Integer, iP0 As Integer
Public Sub Init (oPic1 As PictureBox, oPic2 As PictureBox, oPic3 As
PictureBox) '初始化
    Set Picture1 = oPic1: Set Picture2(0) = oPic2: Set Picture2(1) = oPic3
    SetPm 500, 40, 40: Px = 0: iP1 = 1: iP0 = 0
    Picture2(iP0).Left = Picture1.Width-Picture2(iP0).Width-500
End Sub
Public Sub SetPm(xSpace As Integer, dx As Single, dy As Single)
    rDpx = dx: rDpy = dy: xSp = xSpace
End Sub
Public Sub SetLineWidth(Lwid As Integer)
    Picture2(0).DrawWidth = Lwid: Picture2(1).DrawWidth = Lwid
End Sub
Public Sub Run(k As Long, data() As Single) '运行
    Dim j%
    Picture2(iP1).PSet (Px, rDpy * data(k)+Picture1.Height /2), QBColor(11)
    Picture2(iP1).Line (Px, rDpy * data(k)+Picture1.Height /2) _
           -(Px+rDpx, rDpy * data(k+1)+Picture1.Height /2), QBColor(11)
    Px = Px+rDpx
    If (Picture2(iP0).Width+Picture2(iP0).Left) < -xSp Then
       Picture2(iP0).Left = (Picture2(iP1).Left+Picture2(iP1).Width)
    End If
    If (Picture2(iP1).Left+Picture2(iP1).Width <= Picture1.Width-xSp) Then
       j% = iP1: iP1 = iP0: iP0 = j% : Px = 0
    End If
    If (Picture2(iP0).Left+Picture2(iP0).Width) >= -xSp Then
       Picture2(iP0).Left = Picture2(iP0).Left-rDpx
       Picture2(iP1).Left = Picture2(iP1).Left-rDpx
    End If
End Sub
```

应用程序需要首先通过下列程序代码：

```
dim oCv as New CS49_CvMove, k as long
oCv.Init Picture1, Picture2, Picture3
```

定义 CS49_CvMove 类变量 oCv，并将用户窗体(如 Form1)中自建的控件 Picture1、Picture2 和 Picture3 传递到 oCv 类中。同时，可在 Form1 中建立一个 Timer 控件(如 Timer1)，并在其事件响应函数中添加如下代码：

```
oCv.Run k, Data: k=k+1
```

这样，就可不断地从数组变量 Data 中取出 k 点的心电数据并动态显示波形。并通过调节 T=Timer1. Interval 属性来改变波形显示的扫描频率，也可通过调用类函数 SetPm 和 SetLineWidth 来改变其他显示参数。图 8-43 则给出类函数的整体工

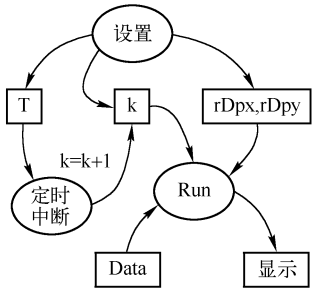



图 8-43　类函数的调用关系

作方式。

## 本 章 小 结

本章以频谱分析、数字滤波器、时间序列建模和统计量置信区间的分析为题，阐述了 DFT、FFT 算法的基本原理和实现方法，介绍了 FIR 和 IIR 的基本概念和典型参数的配置，指出了最小二乘递推算法及其在参数辨识中的应用，也给出了诸如 $m$ 序列等实现激励信号的典型方法，结合对估计参数的统计分析，说明数理统计方法在测试系统中的应用，从而体现了测试软件在自动测试系统和虚拟仪器中事半功倍的重要作用。

同时，本章以大型旋转机械状态监测系统和小型心电测试系统的组建过程为例，分别指出了自动测试系统和虚拟仪器组建的特点和设计方法。前者需要发挥已有测试设备的专业作用，并在网络环境中多方软件的配合下，实现确保产品质量和设备安全的重要目的；后者则需要充分发挥 PC 的主体作用，并在灵活设计的自制电路配合下，可以方便地实现人们的想象力。这些典型示例表明，从大型工业的骨干核心设备到日常生活的灵活创新活动，自动测试系统和虚拟仪器都担负着十分重要的作用。

## 参 考 文 献


[1] 曰井支朗．信号分析．科学出版社，2001.

[2] 王广雄．自动控制系统设计．宇航出版社，1992.

[3] 应怀樵．波形和频谱分析与随机处理．中国铁道出版社，1985.

[4] 熊光楞．控制系统数字仿真．清华大学出版社，1982.

[5] 王秀峰，卢桂章．系统建模与辨识．电子工业出版社，2004.

[6] 吴翔，李永乐，胡庆军．应用数理统计．国防科技大学出版社，1995.

[7] 黄文虎，夏松波，刘瑞岩等．设备故障诊断原理、技术及应用．科学出版社，1996.

[8] Joseph Giarratano and Gary Riley. Expert Systems. PWS. Boston，1994.

[9] 王雷，张礼勇．旋转机械故障监测诊断及专家系统发展动态．电测与仪表，1990. 7.

[10] 蒋东翔，张礼勇，徐世昌，黄文虎．基于知识的传感器故障在线诊断系统．电测与仪表，1993. 11.

[11] 冯玉贵，刘思久．多机组故障诊断系统．电测与仪表，1995. 12.

[12] 张礼勇．国家“八五”科技攻关项目 85-720-07 课题研制报告．1996.

[13] 刘思久．国家“九五”科技攻关专题 97-772-01-08 研制报告．2000.

[14] 刘思久，李桃等．专家系统基本概念及其在自动测试中的应用．电测与仪表，2000. 9.

[15] 刘思久，孙延峰，滕岩峰．自动测试中的系统辨识机．电测与仪表，2000. 12.

[16] 远坂俊昭著．何希才译．锁相环(PLL)电路设计与应用．科学出版社，2006.

[17] 张华强，乔晓林，郑载满．家用心电图示仪及远程诊断系统．黑龙江自动化与应用，1999. 5.

[18] 汪克仁．双 T 网络有源滤波器性能分析．全国滤波专题讨论会文集，1983：85-89.

[19] 李建平．C8051F320 在 USB 设计中的应用．国外电子元器件，2005：54-56.

[20] USB Firmware Programmer’s Guide. Silicon Labs. 2003：1-239.

# 参考复习题

## 第 1 章　绪论

1. 何谓自动测试系统？何谓通用接口技术？
2. 何谓虚拟仪器？试述虚拟仪器中计算机的作用。
3. 试述三代自动测试系统的发展过程，指出自动测试系统的发展方向。
4. 指出自动测试系统中的几种接口总线形式，说明各自特点。
5. 试述自动测试技术的应用领域和意义。
6. 说明测试技术在信息领域中的地位。
7. 试述测试工程师的工作范围。
8. 测试与测量有何不同？
9. 组建自动测试系统时采用 GPIB 和 VXI 技术有什么好处？指出它们各自的特点。
10. 指出 USB 作为通用 PC 与测试电路设备之间连接方式的主要特点。
11. 举例说明什么情况下需要用自动测试系统来代替专用仪器？什么情况下不需要？

## 第 2 章　数据采集器

1. 试述数据采集的基本组成部分，并以框图方式说明各部分的联系。
2. 试述 A/D 转换器的几种典型类型，指出它们的工作特点和应用场合。
3. 试述 A/D 转换器的一般控制方法，给出微处理器控制的流程。
4. 已知模拟输入信号最大值为±10V，试参考图 2-11 所示电路，设计电平转换和增益调整的前置电路来适应 0~5V 的 A/D 转换器输入，并按 1Hz 带宽设计一阶滤波器，确定各阻容参数。
5. 已知仪用放大器选择 AD612(电路如图 2-13 所示)，若模拟输入为 0~0.2V 信号，要求0~2V输出，试确定适当的电路连接方式和参数。
6. 试为模拟电压为 0~5V 的输入通道选择设计光电隔离器电路(参考图 2-14)，要求 0~2V输出。确定电路各参数，并解释工作过程。
7. 试设计 Bach 滤波器电路，以实现截止频率为 1kHz、高频衰减率为-60dB/十倍频程的巴特沃斯低通滤波器。注意输入信号源内阻为 100Ω，等效负载电阻为 10kΩ。
8. 试设计四通道的通道扫描电路，信号频率带宽为 10kHz，电压范围为±5V。
9. 某数据采集器拟用于分析 100Hz 带宽的信号，采用 10 位字长的 A/D 转换器和四阶巴特沃斯低通滤波器来避免混叠。试求滤波器的转折频率和系统采样频率。
10. 通常正弦信号波形的展示需要在其周期内采样多次，如何利用 M/N 采样法以 100kHz 的 A/D 转换器测取 100kHz 的平稳正弦波信号波形？
11. 试设计一个具有 8 通道切换、分辨率 10 位、采样速度大于 10kHz 的数据采集器，试

画出原理框图，选择元件参数，并给出微处理器控制的流程。

## 第 3 章　GPIB 通用接口总线

1. 试述 GPIB 的发展过程，说明有关协议标准的名称和主要作用。
2. 试述 GPIB 的基本概念和组成结构、总线信号的定义和作用。
3. 解释 GPIB 多线消息的“三线握手”工作过程。
4. 试述 GPIB 规范的机械、电气和功能特性的要点。
5. 对照简化的 GPIB 接口状态图，解释远地-本地的基本概念。
6. 对照简化的 GPIB 接口状态图，解释讲功能的基本概念。
7. 解释图 3-13 所示 IEEE 488.2 的接口控制流程。
8. 解释图 3-14 所示状态消息的处理方式。
9. 举例说明 SCPI 的目的和要点。
10. 如何通过总线控制器对 TEK220 示波器设置量程、控制动作和读取波形数据。

## 第 4 章　VXI 仪用总线系统

1. 试述 VXI 的基本组成结构。
2. 试述图 4-11 所示 VXI 的数据传输工作过程。
3. 试述 VXI 0 槽控制器的功能，指出其与主计算机的连接方式。
4. 解释消息基仪器和寄存器仪器的差别。
5. 解释 VISA 的意义，试给出语句示例。
6. 对照图 4-18 所示的 VISA 结构模型图，说明各部工作方式和层次关系。
7. 对照图 4-19 所示的仪器驱动器外部接口模型，试述其工作原理。
8. 试述 LabWindows/CVI 语言的编程要点，解释程序清单 4-2 的要点。
9. 简述 IVI 规范的作用。

## 第 5 章　测试系统的远程操作

1. 何谓 RS232-C 接口？指出标准的由来和协议的要点。

2. 图示如何通过调制解调器连接位于两地的 PC 主机和微处理器控制下嵌入式测试设备之间的通信。指出 DTE 和 DCE 的角色配置，以及接口插座的形式。

3. 何谓 DTE 和 DCE？当测试系统中两台设备(如 PC)试图用串口直接通信时如何考虑它们的相连和定义？试图示解决的方法。

4. 举例说明如何在 PC 主机中利用 VB 程序控制串口连接的远程仪器。
5. 何谓 Xmodem 协议？指出选择数据块长度的原则。
6. 试述 CRC 校验的原理，写出函数代码并指出调用方法。
7. 试述如何通过 PC 将被测数据发至网上指定 IP 的计算机。
8. 如何利用手机短信业务传输测试数据？
9. 简述 RS232 与 RS485 之间的区别，指出其应用的不同场合，试画出转换电路框图。
10. 试编写通过 RS232 串口操作 TEK 示波器 TDS220 读取波形的测试程序。

## 第 6 章　基于 USB 的虚拟仪器

1. 试述 USB 的 4 种传输方式及其典型应用领域，并说明可能用于虚拟仪器的方式。
2. 试述 USB 的 10 种包结构，说明控制方式的工作过程。

3. 试述 USB 的列举过程，说明虚拟仪器如何实现即插即用。

4. 指出开发 USB 设备的主要工作内容。

5. 借助图 6-11，解释 USBN9604 接口芯片端点 0 的操作过程。

6. 试述 USB 设备的典型固件结构，指出如何调用 usb. c 的通用功能。

7. HID 是用于人机对话的工作方式，试解释用于 PC 主机的 VB 程序。

8. 借助图 6-17，解释 USB 信源/数采器的控制流程。

9. 借助图 6-19，指出 USB 高速数据采集器的工作原理和特点。

10. 试述影响 USB 高速传输的各种因素。

11. 试述设想 UXI 概念的目的和要点。

## 第 7 章　PCB 电子功能模件的自动测试

1. 试述 PCB 电子功能模件测试的基本方式，指出功能测试与路内可否相互代替？

2. 解释路内测试的恒压法与恒流法，说明各自的特点。

3. 解释阻容元件路内测试的三线测试法和六线测试法。

4. 试画出二极管和稳压管的路内测试框图，说明其工作原理。

5. 试画出虚拟仪器实现的逻辑分析仪原理框图(可简化)，说明基本工作原理。

6. 试画出特征分析仪的原理框图(可简化)，说明其基本工作原理。

7. 试画出边界扫描测试的示意图，解释其基本工作原理。

8. 试述 JTAG 接口的基本结构形式，给出一种虚拟仪器实现的电路方案。

9. 借助图 7-31，说明 IEEE 1149. 1 中如何从 Shift-IR 状态转移到 Shift-DR 状态。

10. 借助图 7-40，说明 IEEE 1149. 4 对混合信号扩展互连测试的原理。

11. 试举例说明 TEK 信号发生器的程控命令，并给出实现扫频信号的编程示例。

## 第 8 章　典型软件算法和典型应用系统

1. 编写 DFT 和 FFT 程序，并用示例比较算法的执行速度。

2. 指出离散傅里叶变换与实际频谱分析的关系。

3. 举例说明数字滤波器的等效离散化方法。

4. 指出 FIR 与 IIR 的主要区别，指出各自的特点。

5. 针对 10kHz 的干扰噪声，设计一阶 IIR 数字滤波器，指出关键参数。

6. 试设计一个四阶的 FIR 数字滤波器，指出可以通过选择不同参数来达到不同的滤波目的。

7. 试述参数估计中最小二乘算法的递推计算公式，解释其各部分含义。

8. 解释 ARMAX 模型的各部意义，说明时间序列建模的参数估计算法。

9. 指出汽轮机组主轴转子监测与故障诊断系统的总体结构和传感器配置情况。

10. 指出旋转机械监测系统数据采集器的主要特点，解释其数据采集器采用等周期采样方式的原因。试画出锁相环实现电路框图，并阐述倍频原理。

11. 试述模糊诊断方法的原理，指出专家系统用于诊断的基本方法。

12. 简述心电测试原理，图示其实现框图。

13. 试用 VB 编写绘制图形曲线的实用程序。

# 反侵权盗版声明

电子工业出版社依法对本作品享有专有出版权。任何未经权利人书面许可,复制、销售或通过信息网络传播本作品的行为;歪曲、篡改、剽窃本作品的行为,均违反《中华人民共和国著作权法》,其行为人应承担相应的民事责任和行政责任,构成犯罪的,将被依法追究刑事责任。

为了维护市场秩序,保护权利人的合法权益,我社将依法查处和打击侵权盗版的单位和个人。欢迎社会各界人士积极举报侵权盗版行为,本社将奖励举报有功人员,并保证举报人的信息不被泄露。

举报电话:(010)88254396;(010)88258888
传　　真:(010)88254397
E-mail:dbqq@phei.com.cn
通信地址:北京市万寿路173信箱
　　电子工业出版社总编办公室
邮　　编:100036

# 《自动测试系统与虚拟仪器原理·开发·应用》

# 读者调查表

尊敬的读者：

欢迎您参加读者调查活动，对我们的图书提出真诚的意见，您的建议将是我们创造精品的动力源泉。为方便大家，我们提供了两种填写调查表的方式：

1. 您可以登录 http://yydz.phei.com.cn，进入右上角的读书栏目，填好本调查表后直接反馈给我们。
2. 您可以填写下表后寄给我们（北京海淀区万寿路 173 信箱应用电子技术图书事业部　邮编：100036）。

姓名：_____　性别：□ 男 □ 女　年龄：_____　职业：_____

电话（寻呼）：_____　E-mail：_____

传真：_____　通信地址：_____

邮编：_____

1. 影响您购买本书的因素（可多选）：

□封面封底　□价格　□内容简介、前言和目录

□出版物名声　□作者名声　□书评广告　□其他_____

2. 您对本书的满意度：

| | | | | | |
|---|---|---|---|---|---|
| 从技术角度 | □很满意 | □比较满意 | □一般 | □较不满意 | □不满意 |
| 从文字角度 | □很满意 | □比较满意 | □一般 | □较不满意 | □不满意 |
| 从版式角度 | □很满意 | □比较满意 | □一般 | □较不满意 | □不满意 |
| 从封面角度 | □很满意 | □比较满意 | □一般 | □较不满意 | □不满意 |

3. 您最喜欢书中的哪篇（或章、节）？请说明理由。

_____

_____

4. 您最不喜欢书中的哪篇（或章、节）？请说明理由。

_____

_____

5. 您希望本书在哪些方面进行改进？

_____

_____

6. 您感兴趣或希望增加的图书选题有：

_____

_____

邮寄地址：北京市万寿路 173 信箱电子技术分社　张榕　收　　邮编：100036

电　　话：（010）88254455　　E-mail：zr@phei.com.cn