单片机应用技术选编(10)

何立民　主编

北京航空航天大学出版社

内 容 简 介

《单片机应用技术选编》(10)选编了国内 60 种科技期刊中有关单片机与嵌入式系统应用类文章共 640 篇,全文部分 148 篇,摘要部分 492 篇。全文部分内容有:专题论述;综合应用;软件技术;网络、通信与数据传输;新器件与新技术;总线技术及应用;可靠性及安全性技术;DSP 及其应用技术;HDL 与可编程器件技术;综合应用。摘要部分的文章,都提供了内容提要和出处,以供选用参考。

本书编选的内容都是近年来单片机与嵌入式系统应用的一些热点技术,有助于减少单片机与嵌入式系统应用中的一些重复劳动,提高嵌入式系统应用水平,是从事单片机与嵌入式系统应用开发技术人员案头的重要参考资料。

图书在版编目(CIP)数据

单片机应用技术选编. 10/何立民主编. —北京:北京航空航天大学出版社,2004. 3

ISBN 7-81077-332-1

Ⅰ. 单… Ⅱ. 何… Ⅲ. 单片微型计算机—文集

Ⅳ. TP368. 1-53

中国版本图书馆 CIP 数据核字(2003)第 035818 号

单片机应用技术选编(10)

主　　编　何立民

责任编辑　曾昭奇

北京航空航天大学出版社出版发行

北京市海淀区学院路 37 号,邮编 100083　发行部电话(010)82317024

http://www. buaapress. cn. net　E-mail:bhpress@263. net

河北省涿州市新华印刷厂印装　各地书店经销

*

开本:787×1092　1/16　印张:53. 75　字数:1376 千字

2004 年 3 月第 1 版　2004 年 3 月第 1 次印刷　印数:5 000 册

ISBN 7-81077-332-1　定价:75. 00 元

序　言

由于《单片机应用技术选编》(9)的拖延,致使《单片机应用技术选编》(10)与《单片机应用技术选编》(9)几乎同期完成。所选文章跨越 3 年,正好是 21 世纪的头 3 年,是单片机及其相关技术转型的关键时期。如果说《单片机应用技术选编》(9)带有较多转型前的印记,那么《单片机应用技术选编》(10)(后面简称《选编》(10))则体现出许多转型特点。

1. 32 位嵌入式系统会有较大的发展

由于网络、通信、数据传输技术的大力发展,8 位单片机内部已无法满足要求,32 位嵌入式系统会成为网络、通信、数据传输领域中的重要角色。应用在控制领域的 8 位单片机,由于嵌入对象的广泛性与多样性,以及嵌入式系统的初级阶段而呈现出百花齐放的局面。与之相比,32 位嵌入式系统会形成数量不多的处理器支持局面。许多半导体厂家不再自行开发自己的 32 位嵌入式微处理器,而是基于一些通用的嵌入式微处理器核,来构成 32 位微控制器。因此给 32 位嵌入式系统的应用开发带来极大的方便。其中 ARM 嵌入式微处理器的应用值得注视。

2. 基于 DSP 的嵌入式系统应用比重上升

在网络、通信、数据传输以及人工智能的应用中,数据处理的应用需求上升,DSP 芯片价位大幅度下降,导致以 DSP 芯片为主角的嵌入式系统应用比重大幅度上升。为满足日益增长的,以 DSP 为中心的嵌入式系统应用市场需求,许多原来的单片机厂家进军 DSP 市场,推出各具特色的 DSP 单片机。原有的 DSP 厂家除不断提高 DSP 处理能力外,还加强了 DSP 芯片的控制功能和应用系统要求的外围电路集成。在 DSP 的嵌入式应用中也出现了 DSP＋MCU 的芯片体系或 DSP＋MCU 的双机应用结构。

3. CPLD 会成为嵌入式系统应用的一个重要方面

长期以来,在嵌入式系统应用中,PLD 处于配角地位。随着 CPLD 技的发展,芯片价位大幅度下降,软件工具的不断普及,CPLD 在嵌入式系统应用中的比重会不断加大。这一趋势会因为 CPLD 芯片向可配置的 SoPC 或 PSoC 发展而加速。同时,由于 CPLD 与 IC 设计有良好的界面,在嵌入式系统向 SoC 发展中,无疑地处于一个极为有利的地位。不少厂家在推动 CPLD 应用中,提供许多 IP 核供应用时选择,因此,CPLD 会逐渐形成嵌入式系统应用的一个重要方面。

4. 嵌入式应用系统设计向 IC 设计贴近

长期以来,嵌入式应用系统的 ASIC 都是由 IC 厂家承担。近年来,由于 IC 生产线加工能力过剩和多项目晶圆(MPW)服务的支持,使 ASIC 流片费用大幅度下降。IC 设计的普及、IP 库的完善以及 IC 设计与应用系统设计的友好界面,使传统的嵌入式应用系统设计向 IC 设计贴近。因此,基于 ASIC 的嵌入式应用系统设计,会逐渐成为产品设计人员的一种重要方法。目前不少嵌入式系统产品设计研发公司都开始建立 IC 设计部门。

5. 基于集成开发环境的产品研发

早期,通常都是基于单片机裸片的硬件系统设计、制作,基于指令系统、汇编语言、开发装置的应用程序设计产品的开发模式。一个新单片机从推出到广泛应用于产品中,周期很长,应用水平不高。目前,许多单片机厂家在推出新单片机时,逐渐采用为用户提供集成开发环境的厂家软、硬件平台的方法。随着单片机更加 SoC 化,这种趋势更加明显。在这种厂家平台支持下,用户可直接进入产品目标系统的应用程序设计。通常厂家提供的集成开发环境平台包括用户板(试验、评估系统)、驱动程序、仿真器(模拟、在线)、编译器、操作系统等用于产品开发的基础软、硬件资源。这样,可使用户将主要精力集中于用户产品本身的技术开发,既大大地缩短了新单片机的推广周期、用户产品研发周期,同时,也大大地提高了产品研发水平与质量。

《选编》(10)收集了 2002 年 60 种期刊的 640 篇文章,可以明显看出 DSP 和 PLD 技术的发展,故编成独立章节。由于篇幅所限,选择了 148 篇全文发表,其余的文章提供了摘要。

与每集出版相同,对于全文发表的文章作者,我们都要发函,与作者联系,并在本书出版后付给相应的稿酬。对于个别未能取得联系的作者,请见到本书后迅速与我们联系。同时感谢马海珍女士为联系《选编》(10)中文章作者所做的大量工作。

联系人:马海珍

通信地址:(100083)北京航空航天大学出版社

联系电话:(010)82317022

主编　何立民

2003.5

第一章

专题论述

1.1 嵌入式系统的技术发展和我们的机遇

北京航天信息股份有限公司 (100080) 魏庆福
福州大学 (350000) 郑文波

一、概 述

1. 嵌入式系统的定义和市场前景

嵌入式系统是指以应用为中心,以计算机技术为基础,软件硬件可剪裁,适应应用系统对功能、可靠性、成本、体积、功耗严格要求的专用计算机系统。它主要由嵌入式微处理器、外围硬件设备、嵌入式操作系统以及用户应用软件等部分组成,用于实现对其他设备的控制、监视和管理等功能。它通常嵌入在主要设备中运行。

PC 机主要应用在办公室自动化领域,而嵌入式系统已经广泛渗透到人们的工作、生活中,如家用电器、手持通信设备、信息终端、仪器仪表、汽车、航天航空、军事装备、制造工业、过程控制等。今天,嵌入式系统带来的工业年产值已超过 1 万亿美元。美国著名未来学家尼葛洛庞帝 1999 年 1 月访华时曾预言,4 ~ 5 年后嵌入式智能 (电脑) 工具将是 PC 和因特网之后最伟大的发明。据统计,嵌入式处理器的数量占分散处理器的 94%,而 PC 机用的处理器只占 6%。汽车大王福特公司的高级经理曾称:"福特出售的'计算能力'已超过了 IBM!"用市场观点来看,PC 已经从高速增长进入到平稳发展时期,其年增长率由 20 世纪 90 年代中期的 35% 逐年下降,单纯由 PC 机带领电子产业蒸蒸日上的时代已经成为历史。根据 PC 时代的概念,美国 Business week 杂志提出了"后 PC 时代"概念。

根据美国嵌入式系统专业杂志 RTC 报道,21 世纪初的十年中,全球嵌入式系统市场需求量具有比 PC 市场大 10 至 100 倍的商机。1998 年在芝加哥举办的嵌入式系统会议上,与会专家一致认为,21 世纪嵌入式系统将无所不在。它将为人类生产带来革命性的发展,实现"PCs Everywhere"的生活梦想。

2. 嵌入式系统的几个发展阶段

嵌入式系统的出现至今已经有 30 多年的历史。近几年来,计算机、通信、消费电子的一体化趋势日益明显,嵌入式技术已成为一个研究热点。纵观嵌入式技术的发展过程,大致经历了 4 个阶段。

第一阶段是以单芯片为核心的可编程控制器形式的系统,具有与监测、伺服、指示设备相配合的功能。这类系统大部分应用于一些专业性强的工业控制系统中,一般没有操作系统的支持,只通过汇编语言编程对系统进行直接控制。这一阶段系统的主要特点是:系统结构和功能相对单一、处理效率较低、存储容量较小、几乎没有用户接口。由于这种嵌入式系统使用简单、价格低,以前在国内工业领域应用较为普遍。但是远不能适应高效的、需要大容量存储的现代工业控制和新兴信息家电等领域的需求。

第二阶段是以嵌入式 CPU 为基础、以简单操作系统为核心的嵌入式系统。主要特点是：CPU 种类繁多，通用性比较弱；系统开销小，效率高；操作系统达到一定的兼容性和扩展性；应用软件较专业化，用户界面不够友好。

第三阶段是以嵌入式操作系统为标志的嵌入式系统。主要特点是：嵌入式操作系统能运行于各种不同类型的微处理器上，兼容性好；操作系统内核小、效率高，并且具有高度的模块化和扩展性；具备文件和目录管理、多任务、网络支持、图形窗口以及用户界面等功能；具有大量的应用程序接口 API，开发应用程序较简单；嵌入式应用软件丰富。

第四阶段是以 Internet 为标志的嵌入式系统。这是一个正在迅速发展的阶段。目前大多数嵌入式系统还孤立于 Internet 之外，但随着 Internet 的发展以及 Internet 技术与信息家电、工业控制技术结合日益密切，嵌入式设备与 Internet 的结合将代表嵌入式系统的未来。

综上所述，嵌入式系统技术日益完善，32 位微处理器在该系统中占主导地位，嵌入式操作系统已经从简单走向成熟。它与网络、Internet 的结合日益密切，因而，嵌入式系统应用将日益广泛。

3. 嵌入式系统的技术特点

· 嵌入式系统是集软件、硬件于一体的高可靠性系统　嵌入式系统麻雀虽小，五脏俱全。软件除操作系统外，还需有完成嵌入式系统功能的应用软件；硬件除了 CPU 外，还需有外围电路支持；微处理器、微控制器、DSP 已构成嵌入式系统硬件的基础。

· 嵌入式系统是资源开销小的高性能价格比系统　嵌入式系统的发展离不开应用，应用的共同要求是系统资源开销小，由于嵌入式系统技术日益完善，各种高性能嵌入式应用系统层出不穷。它已是资源开销小的高性能价格比的一类应用系统。为了满足系统资源开销小、高性能、高可靠性的要求，大多使用 Flash Memory。

· 嵌入式系统是功能强大、使用灵活方便的系统　嵌入式系统应用的广泛性，要求该系统通常是无键盘、无需编程的应用系统，使用它应如同使用家用电器一样方便。

4. 嵌入式系统的发展趋势

· 低功耗嵌入式系统　为满足高可靠性要求，低功耗的系统将应运而生。

· Java 虚拟机与嵌入式 Java　开发嵌入式系统希望有一个方便的、跨平台的语言与工具，Java 正是用 Java 虚拟机实现 Java 程序独立于各机种的平台。经过努力，一个支持嵌入式系统开发的、足够小、足够快、又有足够确定性的嵌入式 Java 程序包已经出现，Java 虚拟机与嵌入式 Java 将成为开发嵌入式系统的有力工具。

· 嵌入式系统的多媒体化和网络化　随着多媒体技术的发展，视频、音频信息的处理水平越来越高，为嵌入式系统的多媒体化创造了良好的条件，嵌入式系统的多媒体化将变成现实。它在网络环境中的应用已是不可抗拒的潮流，并将占领网络接入设备的主导地位。

· 嵌入式系统的智能化　嵌入式系统与人工智能、模式识别技术的结合，将开发出各种更具人性化、智能化的嵌入式系统。

二、嵌入式系统的核心硬件

嵌入式系统的核心部件是各种类型的嵌入式处理器。据不完全统计，全世界嵌入式处理器的品种已有上千种之多。其中，我们最为熟悉的是 8051 和 68H 结构的产品。实际上，几十年来，各种 4、8、16 和 32 位的处理器在嵌入式系统中都有广泛应用。嵌入式系统的处理器可

以分为两大类：一类是采用通用计算机的 CPU 为处理器，如 X86 系列；另一类为微控制器和 DSP，微控制器具有单片化、体积小、功耗低、可靠性高、芯片上的外设资源丰富等特点，成为嵌入式系统的主流器件。

当前，嵌入式系统处理器的发展趋势主要采用 32 位嵌入式 CPU，其主流系列有 ARM（包括 Intel 公司的 Strong ARM 和 XScale）、MIPS 和 SH 三大系列。

嵌入式系统 CPU 的另一类型为 DSP。当前，DSP 处理器的典型结构是单片化嵌入式 DSP，如 TI 公司的 TMS320 系列；另一类是在通用 CPU 或单片系统中增加 DSP 协处理器，如 Intel 公司的 MCS－296 等。还有一种类型是选用嵌入式单片系统 SoC（System on a Chip）。

其中，特别要指出，RISC 技术为计算机体系结构带来了一次重大的变革。简单的、固定长度的、单周期执行指令的 RISC 计算系统，与传统、复杂、可变长度指令并行执行的 CISC 计算机系统相比较，在相同的条件下，RISC 技术的速度快 2～5 倍，具有巨大的性价比优势。RISC 技术推动着计算机体系结构从封闭的 CISC 向开放的结构发展。因此，世界上各大 CPU 芯片制造厂商争相开发生产 RISC 芯片，目前的典型结构为 ARM 系列、MIPS 和 SH，32 位字长，最高时钟速率可达 400 MHz。多种嵌入式实时操作系统大都支持上述 RISC 处理器。

国际上有一种新的趋向，可以购买 IP 知识产权核模块，即在现有的 IC 电路模块的设计基础上，可根据需求将多个 IP 模块组合起来或经修改，形成自己的新的设计。由此可见，半导体芯片的设计现已不难了。其中，尤以 ARM 的应用最为典型。各半导体厂商大多可生产 ARM 的衍生产品。

三、嵌入式实时操作系统

嵌入式系统的概念出现在 20 世纪 70 年代。当时，大部分嵌入式系统的软件都是由汇编语言编纂而成，只能适用于某种特定的微处理器和应用，当然，对于某些简单的应用是足够了。自 1981 年出现了世界上第一个商业嵌入式实时操作系统（VRTX32）至今，嵌入式实时操作系统已有 20 多年的发展历程。20 世纪 80 年代的产品还只支持 16 位的微处理器，只有内核，以二进制代码为主，当时的产品如 VRTX、PSOS 等，主要用于军事和电信设备。进入 20 世纪 90 年代，现代操作系统的设计思想，如微内核设计技术和模块化的设计思想已开始渗入其中，特别是互联网日渐风行，用户都要求有网络（即支持浏览器）和图形功能，并能方便使用大量现有的软件代码，希望支持标准的 API，如 Posix、Win32 等，并希望其开发环境与大家熟悉的 Unix、Windows 一致，出现了几十种产品，代表性的有 VxWorks、QNX、Nucleus 和WinCE 等。

20 世纪 90 年代后，嵌入式实时操作系统已在嵌入式系统中确立了主导地位，在技术上具有如下突出的特征。

⑴ 因新的处理器越来越多，嵌入式实时操作系统的设计更易于移植，以便在短时间内支持多种微处理器。

⑵ 开放源代码之风盛行，相当多的厂家在出售实时操作系统时，就附加了源代码并含生产版权。

⑶ 电信设备、控制系统要求高的可靠性，迫使这些厂家下功夫提高性能，VxWorks 等对支持高可用性和热切换性都下了一番功夫。

⑷ 嵌入式 Linux 在消费电子设备中得以广泛应用，Linux 得到了相当广泛厂商的支持和投资。RT－Linux 产品也取得了很大的进展。由 32 位嵌入式 CPU 与嵌入式 RT－Linux 相结

合,在家用电器、工业控制和军工装备方面将大有可为。

嵌入式实时操作系统典型产品如下:

· VxWorks WindRiver 公司的高性能可扩展的实时操作系统,具有嵌入实时应用中最新一代的开发和执行环境,支持多种处理器和开发平台,并有多种开发工具,是目前世界上应用最广泛的产品。

· PSOS ISI 公司研发的产品推出时间较早,因此比较成熟,可以支持多种处理器,曾是国际上应用最广泛的产品,主要应用领域是远程通信、航天、信息家电和工业控制。但该公司已被 WindRiver 公司兼并,并将推出 VxWorks 与 PSOS 合二为一的产品。

· VRTX 是国际上最早推出的实时系统之一,比较成熟。其特点是内核紧凑,在模块化方面比原系统有重大的改善。

· Nucleus 的特点是约 95% 的代码用 C 语言编写,方便移植,同时,可提供 Web 支持、网络、图形、文件系统等模块。可全部提供源代码,用户只需通过 DLL 动态连接便可进行任务级调试。

· Lynx 与 Unix 兼容,符合 Posix 标准。是专为要求快速确定相应的、复杂的实时应用设计的,能为在任何一个 Unix 平台上的应用提供相当高程度源级水平上的兼容。

· Windows CE 是微软公司嵌入式实时应用系统,支持众多的硬件平台,其最主要特点是拥有与桌上型 Windows 家族一致的程序开发界面。因此,桌上型 Windows 家族上开发的程序就能在 Win CE 上运行。但嵌入式操作系统追求高效、节省,Win CE 在这方面是笨拙的,它占用内存过大,应用程序庞大。

· RT－Linux 是一种提供源代码、开放式自由软件,具有嵌入式操作系统的很多特色,突出的优势是适用多种 CPU 和多种硬件平台,性能稳定,裁剪性好,开发和使用都很容易。它是发展未来嵌入式设备的绝佳资源。国内也有不少单位在 RT－Linux 方面做了大量卓有成效的工作,具有广泛的应用前景。

此外,后 PC 时代的众多产品,如手持设备等,并不需要强实时性,Palm OS、Java OS 等应运而生。而 Qssl 公司拥有的 QNX 是一种限于 X86 平台的可提供集成化开发环境的实时操作系统。OS/9 在 DVD 等产品中则有广泛的应用。

四、发展和应用我国自主的嵌入式系统技术

1. 嵌入式系统为我们提供了创新空间

⑴ IT 产业中主要是以 IC 和软件为核心技术。核心技术的重点是:

· IC 中以 CPU 最为复杂,所有硬件都受其控制;

· 软件中,OS 是软件的“平台”,OFFICE 是关键应用,数据库也非常重要;

· CPU、OS、OFFICE、数据库等技术是 IT 核心技术的制高点。

⑵ PC 机产业

目前,PC 机的架构为 Wintel 所控制、垄断,即由 Intel 的 CPU + 微软的 Windows 主宰了产业。在该领域,我们没有主动权,没有创新空间,无能为力,充其量只是组装机和搞计算机系统集成等。中国软件企业规模太小,软件产品和软件出口很少。专家估计,10 ~ 20 年内难以突破!

⑶ 嵌入式系统

需求千变万化,没有统一的架构,软硬件需要各种各样的组合,技术密集,市场容量大。我们有无限的创新空间!

2. 发展自主 OS 的有利时机

当前,我们面临发展自主嵌入式操作系统的有利时机,主要是:

· 计算环境从以 PC 为中心转变为以网络为中心,涌现出各种新的信息设备(IA),不必与 PC 兼容;

· 应用软件逐步从 PC 移到网上,不必与 Windows 兼容;

· 浏览器取代 Windows 成为主要的用户界面;

· 基于 Linux 的 OS 很适合各种 IA;

· 跨平台语言 Java 被普遍采用。

综上所述,在 PC 上要脱离 Windows 很难,离开了 PC,特别是在网上,就可以有所作为。其中,看好 Linux!

目前,国内已有多家公司推出基于 Linux 的自主 OS,在服务器领域,特别是在网站上 Linux 已被广泛应用,如北京市电子政务项目。在桌面计算领域,如果 Office 和浏览器二大关键应用成熟,也将能迅速推广。在 IA 领域,各种嵌入式 Linux 有很好的前景;Linux 将成为我国未来的主要 OS 之一。

例如,中软公司推出了"中软实时嵌入式 Linux 操作系统",并在国家的新一代开放式数控系统运行平台开发项目中得到应用。该系统保留了常规 Linux 内核,重新设计了一个新的实时内核。以最小代价提供了强实时性,并可以提供几乎所有常规 Linux 操作系统的功能,如中文图形环境(X-Windows)、TCP/IP 网络和丰富的编程环境资源等。特别是它对外部事件可以做出微秒级响应,能够提供精确的实时时钟控制,实时硬中断和软中断的灵活设置,实时任务和线程的并发操作和同步机制。而且还能提供足够的实时浮点计算功能、很高的 I/O 处理能力,对对称多处理器环境的支持。

我们与福州大学郑文波教授合作开发成功具有 GPL 知识产权的一种嵌入式实时操作系统。它由 RT-Linux 核心、Linuxs 核心、加载模块等组成,采用弹性组合设计,可根据需要灵活多样地配置 500 Kb~2 Mb 不同规模的实时应用系统。RT-Linux 可广泛应用于工业控制、智能仪器仪表、数据网关、路由器、浏览器、通信设备、信息家电、娱乐设备等。

3. 开发嵌入式 CPU 的有利条件

近年来,我国在开发嵌入式 CPU 方面有所长进,有一系列有利条件。

(1) 市 场

嵌入式 CPU 可应用于各种领域,包括各种信息家电、网络设备、工业仪器仪表等。其市场容量将远远超过 PC 产业。

(2) 技 术

· 可允许多种结构,可有自己的创新结构。

· 有 Linux 和其他 OS 作为支持,不必依赖 Windows。

· 复杂性较低(百万晶体管的量级),开发周期短。

· 不必追求性能指标,工艺要求低。

例如,北京中芯微系统公司开发成功高性能嵌入式处理器方舟一号,2001 年 4 月鉴定、发布,并已产业化。其主要技术特点如下:

· 32 位 RISC CPU,时钟频率 166 MHz,200MIPS,2x8 KB cache,片上集成 SDRAM 控制器、PCI 总线控制器、中断处理器、RTC、TIMER、Watchdog 等,且功耗低;

· 自行设计的 32 位 RISC 结构,全新的指令体系结构;

· 拥有自主知识产权;

· 高集成度的 SOC 设计;

· 可裁减,良好的性能价格比。

32 位嵌入式 CPU 有广泛的支持并已大量产业化,有很好的应用前景。它的研制成功,配以国产的嵌入式实时操作系统 RT - Linux,将有可能作为国产嵌入式系统的一种很好的选择,可在信息终端、家用电器、工业控制、军工装备中应用,可以在 IA 领域用自己的核心技术有所作为,摆脱依赖 Wintel 的局面。

4. 启 示

· 嵌入式系统技术可以不受控制,有创新空间。

· 嵌入式 CPU + 嵌入式 OS + 关键应用 = 嵌入式系统的核心技术。

· 我国已逐步掌握信息设备的核心技术。

参 考 文 献

1 郑文波. 嵌入式系统的关键技术与产业化发展方向 [J]. 科技交流月刊,2000 (10)

2 郑文波,邱书毅. 实时操作系统 RT - Linux 及其应用. 自动化科学与技术 [M],电子工业出版社,2000

3 梁合庆. 当今嵌入式系统综述与新的投资机遇 [J]. 测控技术,2000 (4)

4 李小群,等. 浅析嵌入式系统中的浏览器 [J],测控技术,2000 (4)

5 孙玉轩,等. 嵌入式计算机系统在智能仪器中的应用 [J]. 测控技术,2000 (4)

选自《自动化博览》双月刊,2002 年第 4 期

1.2 一种新的电路设计和实现方法——进化硬件

深圳大学信息工程学院 EDA 中心 涂 磊 朱明程

一、引 言

当前的电子系统芯片主要是靠在片上集成更多功能模块的方法来实现更强的功能,随着芯片规模的增大,其技术成本会相应增加,而单位时间的资源利用率却不断下降。因此,需要新的电路设计实现方法和技术来解决这些问题,将系统设计从传统的片面追求大规模、高密度转向提高资源利用率、促进系统功能构造智能化的方向发展。另一方面,在电路设计、自动控制、容错系统、模式识别与人工智能、机器人、太空和深海探索等诸多领域,都迫切需要一种能够根据环境变化或人为的指导而有机地改变自身功能的具有自组织、自适应能力的新型电路系统。自从 30 多年前冯·诺依曼提出研制具有自繁殖与自修复能力的机器的设想以来,人们一直都没有停止过在这方面的探索。但在很长的时期内,由于相关理论和技术条件的限制一直未能实现。

随着进化计算方法的发展,特别是遗传算法等可用于电路进化的算法的出现和不断完善,为具有"进化"能力的电路的出现提供了相关的理论基础。而集成电路技术的飞速发展,特别是近年来大规模、高密度的现场可编程门阵列 (FPGA) 等可编程逻辑器件 (PLD) 的出现,又为进化电路提供了物质基础。于是进化硬件 (Evolvable Hardware,EHW) 作为一种新的系统设计方法和电路实现形式应运而生。

1992 年,日本的 Hugo de Garis 和瑞士联邦工学院的科学家提出了进化硬件的概念,从此关于硬件实现可进化电路的研究和探讨成为数字系统设计领域的热点之一。1995 年举行了第一届进化硬件国际研讨会 (EVOLVE '95),此后国际上每年都会举行许多相关的会议,如美国航空航天局和国防部举办的进化硬件研讨会等等。国际上为此成立了许多专门的研究机构,如日本的可进化系统实验室、瑞士联邦工学院的逻辑系统实验室、美国杨伯翰大学 (BYU) 的可重构计算实验室等。近年来,这方面的研究更是发展迅速,在许多关于数字逻辑、集成电路、计算机、电子设计等的学术期刊和网站上,不断地有相关论文和学术成果发表。进化硬件也已经开始被推向了相关的应用领域。

二、进化硬件及其分类

EHW 是进化计算技术在系统内部结构的设计、调节、实时自适应等方面的应用,关于其确切的含义还存在一些争论,但总的来说通常有两种范畴的定义。狭义地讲,EHW 是指那些使用进化算法来指导自身结构发生变化的硬件系统,主要包括在线自适应与容错系统;在广义上,它可以包括将进化算法与硬件功能设计相结合的许多工作,因此,还可以包括电路的进化设计。依据现有的主要研究方法和手段,可用如下的公式来简单定义进化硬件:

$$EHW = EAs + PLDs$$

即：　　进化硬件 = 进化算法 + 可编程逻辑器件

根据不同的特征可以对 EHW 进行不同的分类。从进化算法上来看，目前，最为普遍的是采用遗传算法的 EHW，此外，还有采用遗传规划等方法实现的 EHW；根据进化算法适应度评估的实现方法，可分为外部进化 EHW 和内部进化 EHW；根据进化电路的实现方法又可分为门级 EHW 和功能级（或称函数级）EHW；另外，从研究方向上看，还可分为基于新的器件特征和结构的 EHW 和基于现有可编程逻辑器件平台的 EHW 等等。

三、进化算法

为了实现“进化”，可进化系统必须使用一种进化算法来指导自身配置的改变。进化计算主要有几大分支：遗传算法（GA）、进化规划（EP）、进化策略（ES）和遗传规划（GP）。这几个分支在算法实现方面有一些差别，但都借鉴了生物进化的思想。

目前，在 EHW 领域使用的最为普遍的是遗传算法（GA）。基本的遗传算法 SGA 原理如图 1.2－1 所示。基于 GA 的 EHW 通常把 PLD 的有关配置位串作为 GA 的染色体，利用 GA 搜寻更好的硬件结构，用所得到的好的染色体对 PLD 进行配置，从而得到更加满足要求或更加适应环境的电路，实现电路的“进化”。其原理如图 1.2－2 所示。

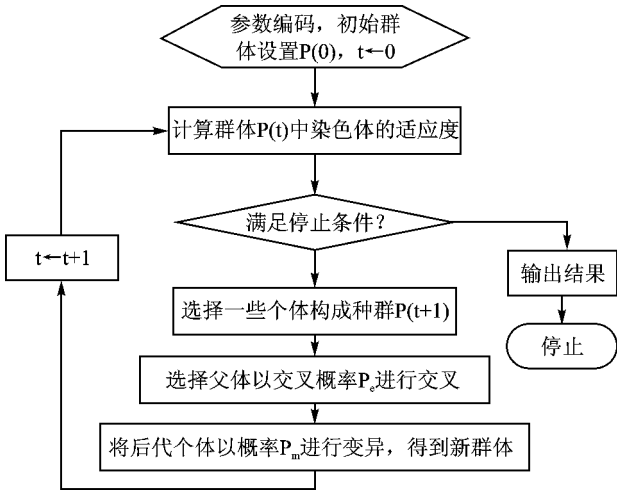


图 1.2－1　SGA 原理

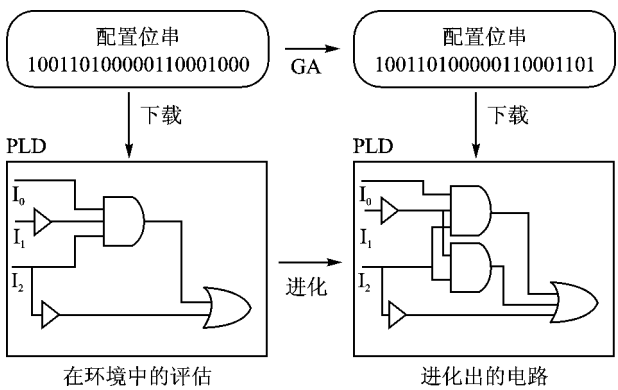


图 1.2－2　进化硬件原理

对于基于 GA 的进化硬件来说,主要的瓶颈在于其遗传操作过程中的计算量。针对这个问题,人们在 SGA 的基础上不断地进行着一些改进,如针对硬件实现将赌轮选择策略变为竞争方式、采用一致交叉策略等。另一种途径是从染色体编码方法等方面入手,如采用变长染色体遗传算法 (VGA)、函数变换法、基于最小项表达式的染色体编码等,这些方法从不同程度上提高了进化的速度,增加了所能实现的电路的规模。

四、进化硬件的 FPGA 实现

当前的 FPGA 器件已具有大规模、高速度、在线可重构等特点。因此,成为了进化硬件设计和研究所采用的主要平台。部分 FPGA 所具有的可部分重构、快速重构、安全配置等特点更是在进化硬件的设计实现中得到广泛利用。

1. 利用现有 FPGA 实现的进化硬件设计

Iwata 等人曾成功地开发出一种进化硬件模式识别系统,其组成如图 1.2-3 所示。它由一块 EHW 板 (包含 4 块 Xilinx 4025 FPGA)、1 个 DOS 机和 1 个用来绘制模式图案的输入板组成。由 DOS 机实施 VGA 并将染色体发送到 EHW 板,板上的 EPGA 被重构并实现模式识别器的功能。实验结果表明,这种系统与 ANN (人工神经网络) 一样具有较高的噪声模式识别能力,并且结果更容易理解。另外,通过使用 VGA,可以得到能够处理更多输入而规模更小的电路。

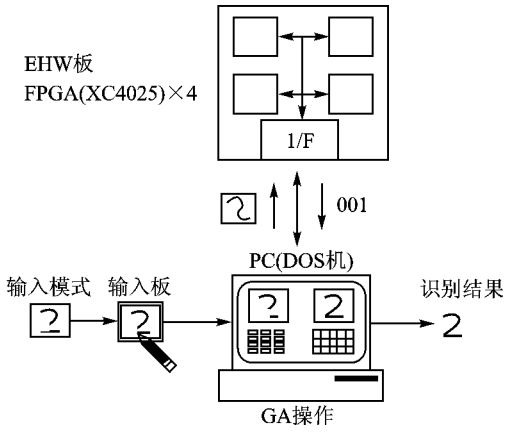


图 1.2-3 EHW 模式识别系统

Xilinx 公司生产的 XC6200 系列 FPGA 具有可进行随机安全配置、部分可重构和可快速重构、内部结构及配置数据格式公开等优点,曾被广泛用于进化电路特别是内部进化 EHW 研究。但该系列现已停产,现有的主流 FPGA 都不具备上述优点。但随着一些软件工具 (如 Xilinx 的 JBits API) 的出现,使得在一些具有与 XC6200 器件相似特性的器件 (如 Virtex) 上进行类似设计和研究成为可能。Hollingworth 等人就通过搭建类似 XC6200 元胞的方法,并结合 JBits API 的使用,在 Virtex 器件上实现了安全的内部进化。他们使用的元胞结构如图 1.2-4 所示。利用 JBits 的 Java 保护措施可以防止破坏性的配置,并且通过 JBits 还可以对系统进行动态的部分重构。利用 JBits 修改配置位流进行部分重构的原理如图 1.2-5 所示。另外,还有人从胚胎电子学的角度出发,利用 Virtex 等器件实现了诸如 MUXTREE 胚胎元胞模型等新的容错进化电路设计。

2. 功能级 (函数级) 进化硬件的实现

目前大多数的 EHW 所进行的是门级进化,但门级进化存在着染色体较长、进化较慢、实现的电路规模小等缺点,妨碍了它的进一步应用。而从门级进化延伸出来的功能级进化的思想则有望导致进化更快、规模更大、功能更强的更具有实用价值的进化硬件的出现。Higuchi 等人提出的功能级进化硬件模型 (F2PGA) 就是从较高级的硬件功能层次上对硬件进行遗传进化的。这种模型由可编程浮点处理单元 (FPU) 和互连开关阵列组成。FPU 可实现加/减法

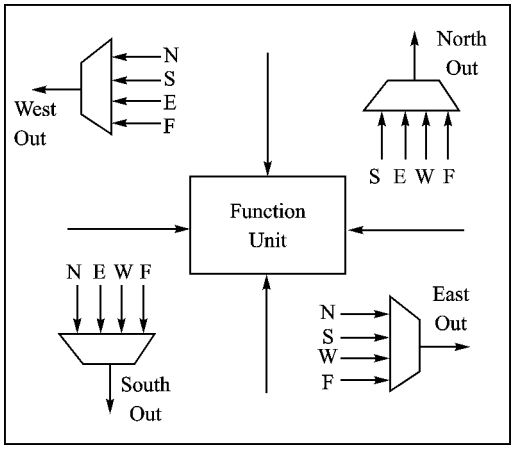


图 1.2－4 在 Virtex 上实现的 XC6200 元胞

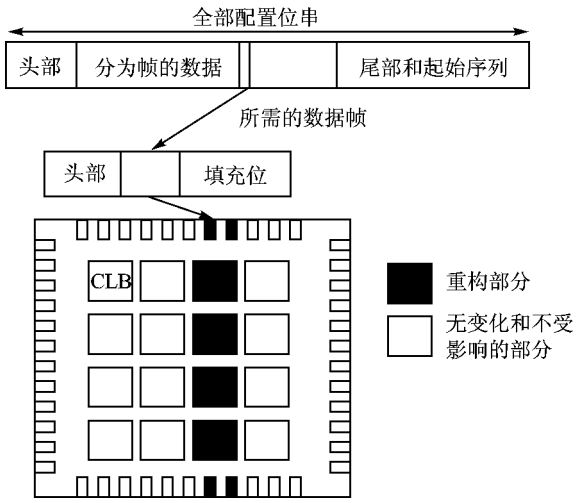


图 1.2－5 系统运行中的动态部分重构

器、正/余弦产生器、乘/除法器、比较（if－then）等功能。通过将 FPU 序号、FPU 功能以及输入操作数等编码为染色体，使用 VGA 进行进化，可以很快得到合乎要求并且规模较大的电路。这种 F^2PGA 模型的结构原理如图 1.2－6 所示。

五、讨论与展望

目前 EHW 研究还存在着许多有待解决的问题。比如，如何进一步提高进化电路的规模和效率？如何得到满足要求的最简化电路？如何保证进化结果的可实现性和长期稳定性？如何有效地实现内部进化？如何提高电路的鲁棒性？如何对自适应进化得到的电路进行有效的分析等等。

这些问题的解决有待于在染色体表达、适应硬件的快速进化算法、简单而有效的功能级进化模型、新的 FPGA 结构、生物进化基本原理及其有效性等方面再作更进一步的研究。

可以预见，随着对这些问题的深入研究和解决，进化硬件作为一种新的电路设计和实现方

法,必将体现出其广阔的应用前景和深远的理论价值。

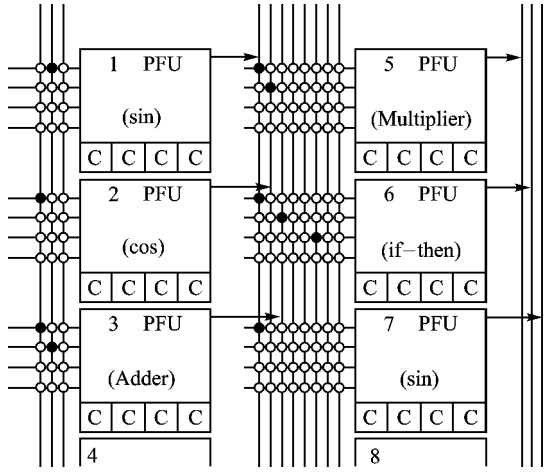


图 1. 2 − 6 **局部** F^2PGA **结构模型**

选自《无线电工程》月刊,2002 年第 9 期

1.3 从 8/16 位机到 32 位机的系统设计

摩托罗拉(中国)电子有限公司 肖踞雄

一、总体概述

在我国现阶段,以 51 系列为代表的 8/16 位处理器在信号采集、系统监控等低端领域仍得到广泛的应用。以 PowerPC、X86、ARM 等为代表的 32 位机由于其设计相对复杂、系统成本较高,因而主要应用于网络路由交换、掌上 PDA 等中高档的设备中。但随着信息技术和电子技术的不断发展,32 位机系统将逐渐成为嵌入式系统设计的主流。由于现阶段应用范围的局限性,市场上关于 32 位机系统设计的书刊并不多。本文就多年的 8/16 位机及 32 位机的实践工作,详细比较了两种不同档次的系统在设计过程中的区别。

二、8/16 位机与 32 位机的比较

现阶段 8/16 位系统与 32 位系统在系统配置及开发上的区别如表 1.3-1 所列。

表 1.3-1 8/16 位系统与 32 位系统配置比较

比较项目		8/16 位系统	32 位系统
硬件特性	ROM/B	一般为几 K 到几十 K	一般为几 M 到几十 M
	RAM/B	一般为几 K 到几十 K	一般为几 M 到几十 M
软件代码量/B		一般为几 K 到几十 K	一般为几 M 到几十 M
操作系统		无需操作系统	一般需要操作系统
软件运行		一般在 ROM 中	一般在 RAM 中
系统 MIPS		几到几十	几十到几百
开发流程及系统复杂性		简单	复杂
开发工作量/人月		几个	几十个
系统应用范围		多用于大系统前台的监控、小型的测试仪表以及小数据量的处理设备等	用于更加复杂的设备,比如图像语音的信号分析处理设备、信息网络交换设备等

三、8/16 位机与 32 位机系统结构及设计过程的比较

1. 一个 8/16 位机采样系统实例

下面是一个由 80C51 构成的带有 LCD 显示、键盘输入、RS232 通信接口的 8 位单片机系统。

(1) 硬件结构

图 1.3-1 所示类型系统主频最高可达 20 MHz,可扩展 64 KB 的外部 ROM 和 RAM。

UART 通信口波特率最高可达 115 200 bps。A/D 采样一般为 8 位或 16 位，采样速度要求不高。键盘一般为从 I/O 口扩展的 4×4/8×8 按键，支持的按键数目有限。扩展的 LCD 显示接口一般为静态的液晶显示模块，且显示的点阵有限，常见的不超过 16×128 点阵。硬件 PCB 板一般为双层板。

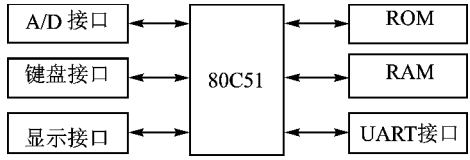


图 1.3－1 8 位采样系统硬件结构图

⑵ 软件结构及流程

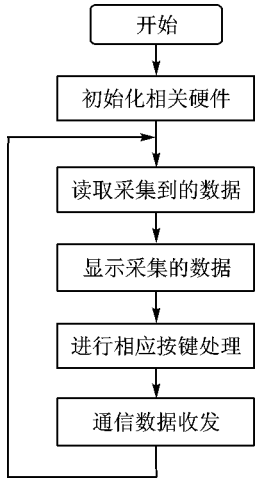


图 1.3－2 8 位采样系统软件流程

该类型系统的软件代码一般为几 KB 到十几 KB，常用汇编语言或 C 语言来实现。软件主体是一个大的死循环，用来处理所有信息，如图 1.3－2 所示。该系统软件首先初始化基本的系统接口，然后进入一个循环处理程序。在如上系统中，循环处理程序中应包含下列模块：从 A/D 口采集数据、检测并处理键盘按键、处理通信口数据、更新显示内容。各模块依照排列的顺序循环运行。这样就会存在一个问题：如果其中任一模块在运行过程中被阻塞住，就会影响其他模块的运行。比如通信口数据传输时，若系统被阻塞在等待接收远程的数据过程中，则 LCD 的显示内容将不会被更新，不会得到新的 A/D 采样数据，键盘操作也不能被及时处理。因而，该类型系统如果在处理模块比较多时，系统软件设计需要较高的技巧，并且可靠性很难得到保障。

⑶ 软件运行时序图

图 1.3－3 软件运行时序图说明了系统运行时各处理模块的调度情况。各模块均按特定顺序依次循环地运行。

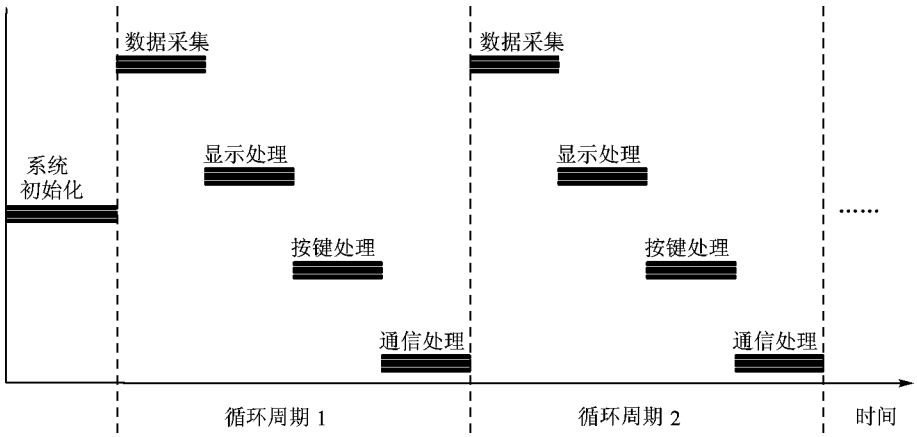


图 1.3－3 8 位采样系统软件时序图

⑷ 系统开发流程

8位采样系统开发流程如图1.3-4所示。

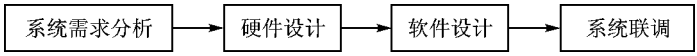


图1.3-4 8位采样系统开发流程

由于该类型系统工作量比较小,开发周期短,并且可参考的资料丰富,一般设计人员为1~2人。系统的开发过程是从硬件到软件顺序的设计。软件的设计与硬件的调试一般同时进行即可。

2. 一个32位机网络访问设备的系统实例

下面是一个由Motorola PowerPC 860处理器构成的网络访问设备,同样扩展了LCD显示、键盘输入、RS232通信接口、以太网接口等。

⑴ 硬件结构

图1.3-5所示类型系统主频一般为几十到几百MHz,通常需扩展几到几十MB外部ROM和RAM。外部ROM和RAM的扩展一般都需要数据及地址总线的驱动。网络接口一般为100/10 Mbps以太网,2 Mbps的HDLC等,UART通信口波特率可支持到1 Mbps。键盘接口一般支持标准PC键盘。显示接口一般为PC机的显示器或点阵数目比较多的LCD。硬件PCB板一般为6~8层板。在硬件的设计和制作过程中,应花更大的精力考虑系统的电气特性,以提高系统的稳定性和抗干扰能力。

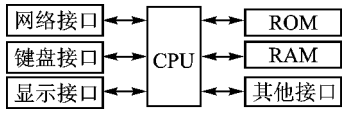


图1.3-5 32位采样系统硬件结构图

⑵ 软件结构

图1.3-6所示32位采样系统的软件一般包括:BSP、硬件相关驱动程序、操作系统及应用程序。软件设计一般使用C/C++语言来完成。

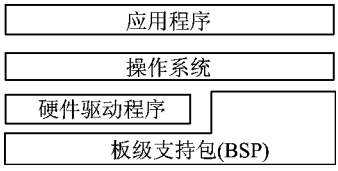


图1.3-6 32位采样系统软件结构图

BSP:BSP一般包含系统硬件板的初始化、操作系统的配置等工作。其与硬件关系密切。不同的硬件电路板都有其各自不同的BSP。

驱动程序:驱动程序主要用来连接操作系统与BSP和硬件平台。BSP中一般已经包含部分硬件的相关驱动程序,也可把所有的硬件驱动程序放在BSP中。但为了增加系统的可移植性,一般把与操作系统关系特别密切的驱动程序不放在BSP中,这主要取决于操作系统的特性。

操作系统:操作系统中一般包含系统资源管理(如内存、外设等)、各进程/线程的调度、各进程/线程之间的通信等。

应用程序:应用程序主要指应用软件部分,比如系统的网络应用程序、相关的图形菜单界面等。

⑶ 软件主流程

嵌入式操作系统一般都为多任务实时操作系统。操作系统为每个应用进程维护了一个消息队列。操作系统接收外部/内部事件并发送给相应的应用进程。每个应用进程都维护一个

自己的消息循环和消息队列,并不断地处理接收到的消息。操作系统同时又提供了各进程/线程的通信机制,比如消息发送/接收、资源的互斥访问、线程的同步等。

如图 1.3-7 所示,软件主流程一般为:分配和初始化所需系统资源、创建各应用模块的运行线程、创建消息循环。

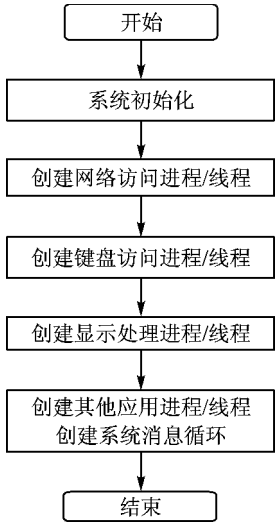


图 1.3-7 32 位采样系统软件流程图

基于多任务操作系统的软件设计,各模块的运行由操作系统来调度完成。软件设计人员不需自己来创建各模块的运行调度机制;但该类型软件设计需要着重考虑各任务模块的同步、互斥和重入等问题。

⑷ 软件运行时序图

图 1.3-8 所示软件运行时序图说明该类系统运行时各处理模块的调度。从宏观上看,各模块在同时运行,互不影响。比如,显示处理不会因为等待键盘输入而不能更新显示内容,各进程/线程之间可用消息及共享内存等机制来进行通信,以完成同步和互斥等操作。

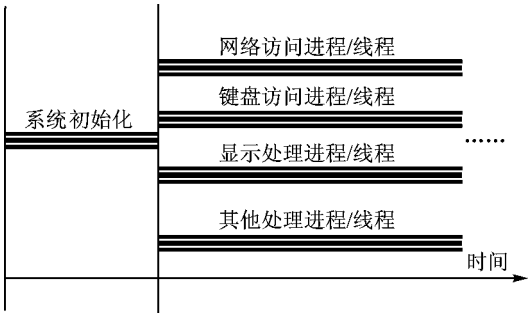


图 1.3-8 32 位采样系统软件时序图

⑸ 系统开发流程

图 1.3-9 所示 32 位机系统设计工作量一般比较大,开发周期长,并且可参考的资料不多,需要多个分别从事不同类型工作的设计人员协同工作。项目的管理和工作的协调在系统

开发过程中尤为重要。为了缩短开发周期,软硬件开发一般同时进行。软件开发人员要求有较强的软件设计能力,对硬件能力要求不高;而硬件开发人员要求有较强的硬件设计能力,对软件能力要求不高。对于 BSP 制作人员,要求有较强的软硬件设计能力。硬件和操作系统的选择都需要根据特定的系统需求来决定。

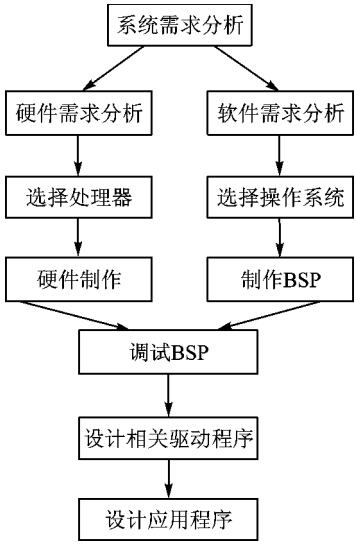


图 1.3-9 32 位采样系统开发流程

四、对开发人员的要求

随着电子技术的发展和 32 位嵌入式系统应用的普及,一个熟悉 8/16 位单片机开发的人员需要为转向 32 位嵌入式系统开发做好相应的准备工作。根据笔者的经验,一个 32 位嵌入式系统的开发人员应了解以下方面的内容:

⑴ 高级语言设计(C、Java 等);

⑵ 中高档处理器(PowerPC、X86、ARM 等)的结构;

⑶ 多任务操作系统资源管理、任务调度与同步等内容;

⑷ 软件设计的基本数据结构等知识。

选自《单片机与嵌入式系统应用》月刊,2002 年第 7 期

1.4 混合 SoC 设计

清华大学电子工程系 (100084) 高 泰 周祖成

一、引 言

随着集成电路工艺的飞速发展和电子系统厂商更广泛地参与微电子产品的设计,传统的电子设计方法和 EDA 工具受到了不断的挑战。一方面,电路设计复杂性的增加和尽快推出产品的压力要求,采用基于 IP、基于重用和基于模块的设计方法;另一方面,深亚微米技术带来的物理特性 (如信号完整性、功耗、电磁干扰),使得底层细节必须引起前所未有的重视,如图 1.4-1 所示。

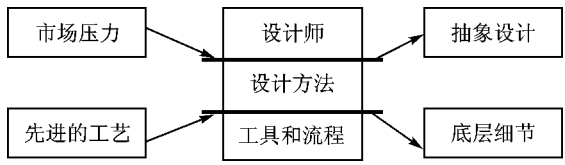


图 1.4-1 设计驱动以及设计方法的趋势

SoC 设计涉及高层 (抽象) 和底层 (细节) 两个方面。它通过适当地处理两者关系,保证高层设计能顺利地连接到底层。通常,SoC 设计包括系统级设计、电路级设计、物理实现、物理验证和最终验证。另外,在 SoC 的整个设计过程中,还要求正确使用约束并完成测试方法的生成。

二、基本概念

传统的 SoC 系统源于 ASIC,典型结构如图 1.4-2 所示。这种 SoC 系统主要是数字部分,包括处理器、存储器、外部接口和相应的嵌入式软件。而混合 SoC 源于消费类 IC,典型结构如图 1.4-3 所示。其最大的特点是包括了复杂的模拟和混合电路部分。由于涉及模拟与数字接口的多次信号反馈,并缺乏准确的物理参数,因此混合 SoC 系统需要发展新的设计方法,而不是在原有设计方法基础上的补充。

图 1.4-2 传统 SoC 示例

图 1.4－3 混合 SoC 示例

三、设计流程

混合 SoC 设计通常采用自顶向下的设计方法，主要包括系统级设计、线路级设计、物理实现、物理验证和最终验证。其优点在于全局性的性能优化，提高设计日程的可预测性，便于协调各级设计人员的工作。其缺点在于需要严格的设计流程和设计人员对 MS-HDL (Mixed-signal Hardware Description Language) 的掌握。

1. 系统级设计

系统级设计一般是由系统工程师来完成。其主要目标是寻找合适的算法和结构，从而在满足所需功能的前提下实现最小成本。该阶段设计通常使用各类系统级仿真工具 (如 Matlab、SPW) 来仿真和评估算法。当算法确定后，系统工程师必须将其映射为特定的结构，以利于线路设计和各模块作为整体验证。

SoC 设计中，由于算法设计工具和结构设计工具的不一致性，算法向结构的映射存在着设计的不连续性。在数字设计领域，已经出现了如 SPW、VCC 的设计工具，可以提供算法到结构的转换。而在混合 SoC 设计中，类似的工具尚未出现。MS－HDL (如 Verilog－AMS、VHDL－AMS) 以其精确的接口模型和支持混合级仿真，将逐渐被广泛用于混合系统级设计。

2. 线路级设计

线路级设计主要涉及仿真和综合两大部分。由于混合 SoC 采用自顶向下的设计方法，混合仿真愈加显得重要。基于行为级模型和晶体管级模型的混合仿真，可以避免传统的 HDL 模型仿真，从而减少仿真成本和提早发现模块接口的错误。

在数字设计领域，综合工具已经实现了较高的自动化。在混合和模拟领域，虽然自动化综合得到了广泛的研究，但各种算法缺乏普遍性，仅能对特定的模拟线路有较好的综合效果。其中，CMU 设计的 ACACIA 系统很具有代表性。我们认为，尽管通用的模拟综合工具很难在短期实现，但专用的模拟综合工具将在未来几年内迅速发展，从而简化模拟和混合线路的设计。

3. 物理实现

物理实现主要包括两方面的内容，即块编辑和块装配。过去几年，针对块编辑已经开发出了不少成功的商用工具、方法和流程，但在块装配方面却处在研究阶段。在混合 SoC 设计中，由于数/模交互接口、IP 重用、功耗管理和信号完整性，块编辑和块装配需要同时进行。保证块编辑和块装配完整性的一个关键部分是布局布线器。然而，今天的商用布局布线工具尚不能完全满足混合 SoC 的设计。可重用性和交互操作性将是未来混合 SoC 物理实现工具的重要特点。

4. 物理验证

近年来,物理验证技术在 EDA 界取得了很大的发展。许多商用工具 (如 Cadence 的 Assura) 已经实现了分层次的物理验证和参数提取,并可和块编辑工具进行良好的集成。在混合 SoC 设计中,物理验证和物理实现的结合将更为紧密。因为基于约束驱动的布局布线工具的应用,衬底耦合噪声验证技术将会在混合 SoC 设计中广泛应用。同时,物理验证工具还将会支持许多新出现的后布局布线技术,如 OPC 和亚波长光刻技术。

5. 最终验证

最终验证是通过物理验证工具从最终布线中提取线路网表 (包括寄生参数) 进行仿真。在数字设计中,通常采用时序仿真器,如 Synopsys 公司的 TimeMill。由于时序仿真器较线路仿真器 (如 SPICE) 而言是牺牲精确性而提高仿真速度,所以通常的时序仿真器不适合模拟线路的最终验证。另外,由于混合 SoC 的规模较大,最终验证通常采用自底向上的验证方式。这种方式通常是提取单个模块的参数,创建相应的宏模块,最后将各个宏模块结合后在高层的仿真器 (如 AMS simulator) 中验证。

四、混合 SoC 设计面临的主要问题和解决方法

在混合 SoC 设计中,特别需要考虑的问题包括模拟部分的工作频率、约束管理、混合测试、设计重用和描述方式。

1. RF 单元

混合 SoC 设计中,RF 模块的加入会对整个设计产生极大的影响。首先,RF 线路通常工作在 1 ~ 5 GHz 的频率上,携带 RF 信号的线路必须短且避免相互干扰。在混合 SoC 的布局、布线、封装过程中,应充分考虑 RF 线路的这个特点。第二,RF 线路易受数字模块的信号干扰。目前,接收机输入的信号可以小到 1 μV,任何通过电源、交叉连接或封装引起的耦合干扰都会影响接收机的灵敏度。因此,建立准确的模型进行预测和仿真是非常重要的。第三,因为 RF 模块的发射部分将对低频信号进行高频调制,其线路仿真的开销很大。一种可能的解决办法是用 RF 仿真器提取 RF 模块的宏模型,并在 AMS 仿真器中进行高效的评估。

2. 约束条件管理

混合 SoC 设计中,各种约束的设定和管理对保证设计过程的连续性非常关键。理想的约束管理系统应该具有以下特征:(1) 能够以一致的方式处理不同类型的约束条件,包括整体设计约束、电气特性约束及物理特性约束。(2) 能够提供各类约束条件间的转换,如通过噪声约束产生交叉耦合限定。(3) 能够进行一致性检查,从而验证约束条件的有效性并尽可能早地避免不合适的约束。(4) 能够提供约束条件的分析和验证工具,以测试和评估约束的性能。

3. 混合测试

混合 SoC 设计中,统一的测试一直是重要的研究方向。传统上,模拟部分和数字部分是分开进行测试的。这种测试方法不仅缺乏系统性,也影响最终的产品成本和推出时间。我们认为,开发支持 MS - HDL 语言的、统一数/模接口的测试工具将能很好地解决混合测试问题。

4. 设计重用

随着设计需求的不断增长,设计者开发产品的复杂度也呈指数增长,于是人们通过设计重用来减轻设计的难度并提高设计效率。设计重用的最终目标是建立一个包含软硬件模块的资源库,该库的模块包含各个层次 (物理层和系统层等) 的描述。当工艺技术进步时,物理层描

述可能被修改,但系统层的描述仍然有效,这样才能真正达到设计重用的目标。

混合 SoC 设计中,尽管像 VSIA (虚拟插件接口联盟) 这样的组织已经制定了很多重用 IP 的标准,但仍有很多问题有待解决,如提高数字模块重用的设计方法、IP 接口验证和质量认证、模拟模块行为级描述与约束条件的联系等。

5. 描述方式

混合 SoC 设计中,采用适当的描述方式是非常重要的。MS - HDL 语言的推出,不仅有利于混合 SoC 在单一的设计流程中实现,而且将事件驱动 (event-driven) 的思想引入了模拟部分的仿真,从而提高混合设计的自动化程度。目前,MS - HDL 分为 Verilog - AMS 和 VHDL - AMS 两种,其商用版本已被 Cadence 等 EDA 厂商所推出。

Verilog - AMS 是 Verilog - HDL 和 Verilog - A 的超级。它结合 Verilog - HDL 的事件驱动模型和 Verilog - A 的连续时间模型,可以有效地建立混合信号行为级模型。它还提供自动接口元件插入,以实现不同类型端口的相互连接。另外,它还支持实值事件驱动网络和交叉寄生参数反向注释。

VHDL - AMS 是在 VHDL 的事件驱动模型的基础上增加了连续时间模型。它支持建立混合信号行为级模型,但不支持数字端口和模拟端口的直接连接,不支持自动接口元件插入,这使得其对混合级仿真的支持存在一定的问题。

五、设计案例分析

蓝牙技术是一种无线数据与语音通信开放性的全球规范。它以低成本的短距离无线连接为基础,为固定与移动设备通信环境建立一个特别连接。由于市场价格的压力,单芯片实现蓝牙标准中的规定功能成为了必然。

单芯片蓝牙 SoC 的主要系统框图如图 1.4 - 4 所示。整个系统包括微处理器、存储器、RF 电路、复杂的模拟与数字接口,体现了混合 SoC 的全部特点。该 SoC 的核心是嵌入式微处理器,用于实现所有 Bluetooth 协议中软件层和部分物理层的功能,并保证 SoC 的应用灵活性。数字基带处理器主要用于实现 Bluetooth 物理层协议中与时间有关的部分。RF 部分主要用于 SoC 系统与外界的无线通信。另外 SoC 还包含了多种音频、数据接口。

图 1.4 - 4 单芯片蓝牙 SoC 系统框图

采用基于自顶向下的混合 SoC 设计,欧洲的一家公司已经实现了类似于图 1.4 - 4 的单片 SoC,其实际设计周期缩短了 18 个月。

本文首先提出了混合 SoC 的基本概念,介绍了通用的设计方法和流程,重点阐述了混合 SoC 设计面临的问题,并提出了可能的解决办法。最后,简单介绍了一个成功的混合 SoC 系统,以加深对混合 SoC 概念、设计方法的理解。

参 考 文 献

1 焦影,周祖成. PBD－SOC 实现的一条重要途径. 电子产品世界,2001,2

2 周祖成. 专用集成电路和集成系统自动化设计方法. 中国集成电路大全,北京 :国防工业出版社,1997

3 Kundert K. Design of mixed-signal systems-on-a-chip. IEEE Trans, On Computed-Aided Design of Integrated Circuits and Systems. 2000,19 (12)

4 Frank. A fully-integrated single-chip SoC for bluetooth. 2001 IEEE International Solid-state Circuits Conference

5 Verilog-AMS [Online], Available :http ://www. ovi. org

6 Virtual Socket Interface Alliance Official Web Pages [Online]. Available :http ://www. vsia. com

7 Wakabayashi, okamoto. C-based SoC design flow and EDA tools and ASIC system vendor perspective. On Computed Aided Design of Integrated Circuits and Systems IEEE Trans,2000,19 (12)

选自《半导体技术》月刊,2002 年第 2 期

1.5 AT24 系列存储器数据串并转换接口的 IP 核设计

武汉华中师范大学物理系 (430079)
谭文虎 彭新生 刘守印 黄光明

一、I^2C 总线的基本概念

I^2C 总线协议是 Philips 公司推出的总线协议。它是多主机总线,通过 2 根线 (SDA—a serial data line, SCL—a serial clock line) 与连接到总线上的器件之间传送信息,根据地址识别每个器件。例如,微控制器、LCD 驱动器、存储器、键盘,连接的器件可以工作在发送和 (或) 接收状态。很显然,LCD 驱动器等一些器件只能是接收器,而存储器可以发送和接收数据。对于 AT24 系列存储器来说,器件的地址是通过把地址输入端 A0、A1、A2 进行硬件连接来确定的。

图 1.5-1 是典型的 I^2C 总线结构。SDA 和 SCL 都是双向线,通过上拉电阻接正电源。当总线空闲时,这两根线处于高电平状态,连到总线的器件的输出级必须是开漏极或集电极开路,以具有线"与"的功能。设备与总线的接口电路如图 1.5-2 所示。

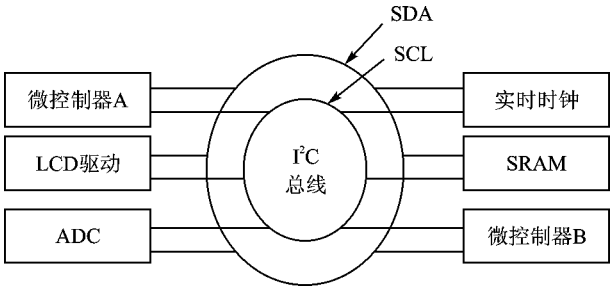


图 1.5-1 典型的 I^2C 总线结构

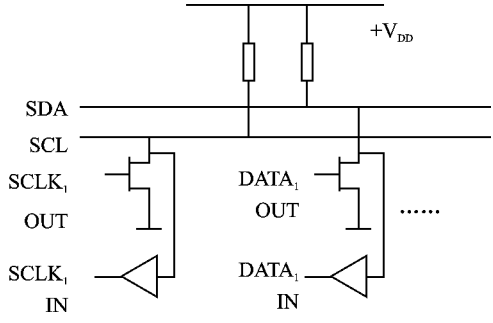


图 1.5-2 I^2C 总线的器件连接

二、I^2C 总线的数据传输

在 I^2C 总线的数据传输过程中，定义了开始和停止信号。如图 1.5－3 所示，SCL 保持“高”，SDA 由“高”变为“低”为开始信号；SCL 保持“高”，SDA 由“低”变为“高”为停止信号。开始（S）和停止（P）信号由主器件产生。在时钟高电平期间上的数据必须保持稳定，如图 1.5－4所示，只有在时钟线 SCL 的时钟低电平期间 SDA 线上高电平或低电平才能变化。

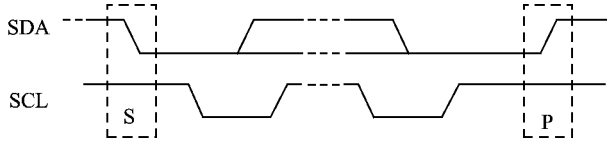


图 1.5－3 总线的开始/停止信号

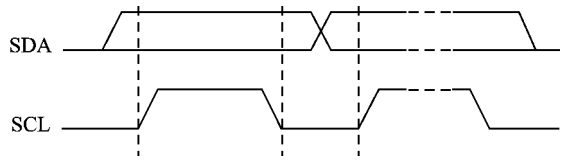


图 1.5－4 I^2C 总线上的位传送

到 SDA 线上的每个字节必须是 8 位长度，每次传输的字节数是不受限制的，每个字节后面必须跟一个响应位。如果一个接收器件在完成其他功能前（如一个内部中断）不能接收另一个数据的完整字节时，可以使时钟保持低电平，以促使发送器进入等待状态。当接收器准备好接收下一个数据字节并释放 SCL 线时，数据传输继续进行。图 1.5－5 表示出了 I^2C 总线上的数据传送时序。

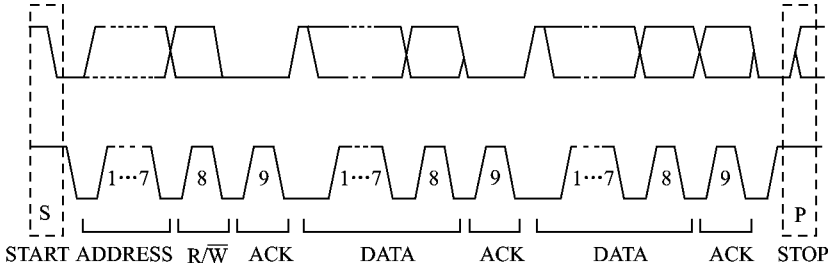


图 1.5－5 总线上的数据传输

三、IP 核的设计

1. IP 核设计与软件实现的比较

在 I^2C 总线的应用中，实现微机与 AT24 系列存储器之间的通信，可以把微机的通用 I/O 口作为 I^2C 总线的接口，通过汇编由软件控制实现数据的传输。由于软件在操作上时间的原因，速度总要受到限制。并且汇编控制也很难作为一个统一的标准在应用中推广。通过 IP 核设计，我们可以在硬件上实现数据串并转换的目的。工作的速度只与存储器本身的特性有关，克服了软件在此方面的不足。

2. 系统设计方案

该系统主要由 I^2C 串行移位寄存器（SSR）、数据缓冲寄存器（IDBR）、控制寄存器（ICR）、状态寄存器（ISR）、从地址寄存器（ICCR）、SCL 产生器（SCL Generator）及其他总线组成。图 1.5－6 为其基本内部结构。

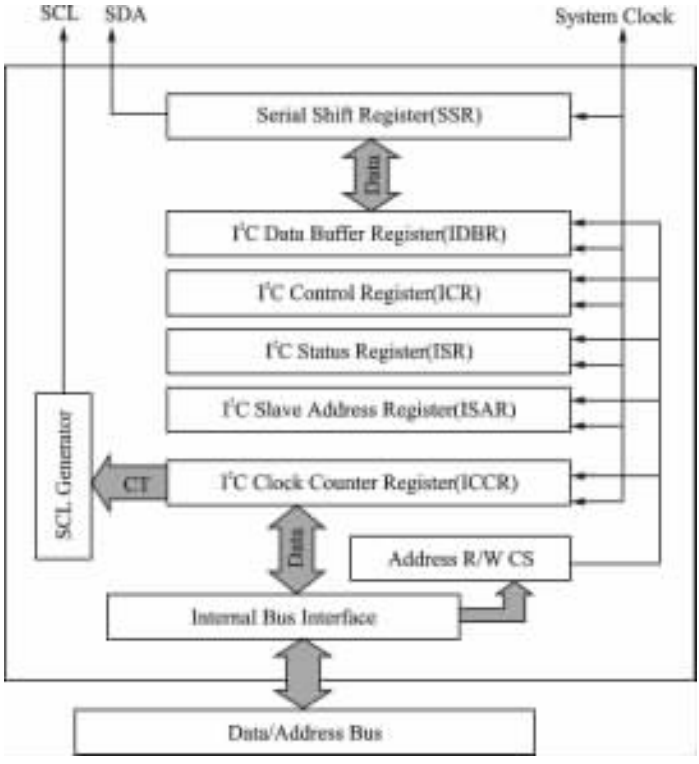


图 1.5－6 系统设计的内部结构

在该系统中，SSR 把并行数据变为串行数据，传输给存储器，或者把存储器的串行数据变为并行数据，传输给处理器；IDBR 把并口来的数据或把被转换成并行的数据暂时装载起来；ICR 控制着整个系统的读/写、数据的转换等操作；ISR 则监视着系统的状态。

3. 数据的通信格式

如果主控制器（CPU）要从存储器读数据或者写（0 表示写）数据到存储器，则需经过接口转换。SDA 上的信号传输要遵循一定的格式。在主控制器（CPU）给存储器写数据时，把设备地址、字节地址和数据送给接口，接口完成与存储器之间的数据交换。格式如下：

SDA

S	设备地址	R/W	A	字节地址	A	数据	A	P
开始		0					停止	

其中确认（A）是存储器传送给接口的信号，其余的如开始（S）、设备地址等信号是接口产生的信号。

主控制器从接口读数据时，会把设备地址、字节地址和读信号告诉接口，接口通过与存储

器进行数据交换,把数据读出来,送给主控制器。数据格式如下:

SDA

S	设备地址	R/W	A	字节地址	A	S	设备地址	R/W	A	数 据	P
开始		0				开始		1			停止

其中确认(A)和数据是存储器产生的,其余的如开始(S)、设备地址、停止(P)等信号是接口产生的。

4. IP 核的 VHDL 设计

IP 核的 VHDL 设计从低到高共 5 个模块。这几个模块分别为头地址移位寄存器模块、数据移位寄存器模块、计数器模块、控制模块和外围综合模块。

头地址移位寄存器是用来装载写入(读出)设备地址,在控制模块的控制下,把设备地址移位到串行数据线 SDA 上。数据移位寄存器是用来装载写入/读出的数据、字节地址,并在控制模块的控制下,把写入的数据、字节地址移位到 SDA 上,或者把从 SDA 读出的串行数据变为并行数据,以传送给主控制器。在该 IP 核设计中,需要对移位的数据字节进行计数,计数器模块实现该功能。控制模块主要通过以刚提到的 3 个模块为基础,实现了数据的单向传输,也就是把双向的数据线分成二根单向的数据线来传输数据。而外围综合模块则把二根单向的数据线综合成一根双向的数据线 SDA,实现了接口的串并转换功能。

5. VHDL 的实现与仿真

硬件描述语言 VHDL(Very—high Speed IC Hard—ware Description Language)是一种用于电路设计的高层次描述语言,具有行为级、寄存器传输级和门级等多层次描述,并具有简单、易读、易修改和与工艺无关等优点。本设计采用 MAX + plus Ⅱ 9.5 作为综合工具,对设计的 VHDL 程序进行调试和波形仿真。

在调试中,MAX + plus Ⅱ 生成所需要的 I^2C 接口模块,如图 1.5-7 所示,表示了整个接口的外部结构。

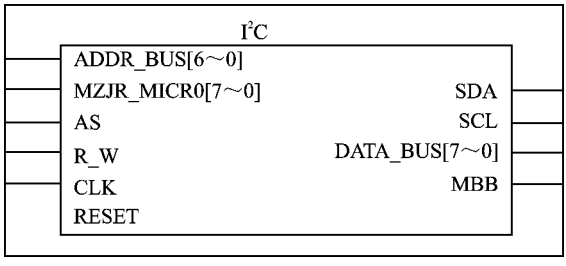


图 1.5-7 存储器接口的模块

其中各个管脚的意义如下:

ADDR_BUS[6~0]	7 位地址总线
MZJR_MICRO[7~0]	8 位字节地址总线
DATA_BUS[7~0]	8 位数据总线
AS	读/写准备信号
R_W	读/写信号(0 为写)

CLK	系统时钟
RESET	复位
MBB	系统忙信号
SDA	串行数据线
SCL	串行时钟线

在仿真中,选择 EPF10K10LC84—3 作为下载芯片来实现模拟仿真。当向存储器写数据时,串行时钟线和数据线得到如图 1.5-8 所示的仿真波形。

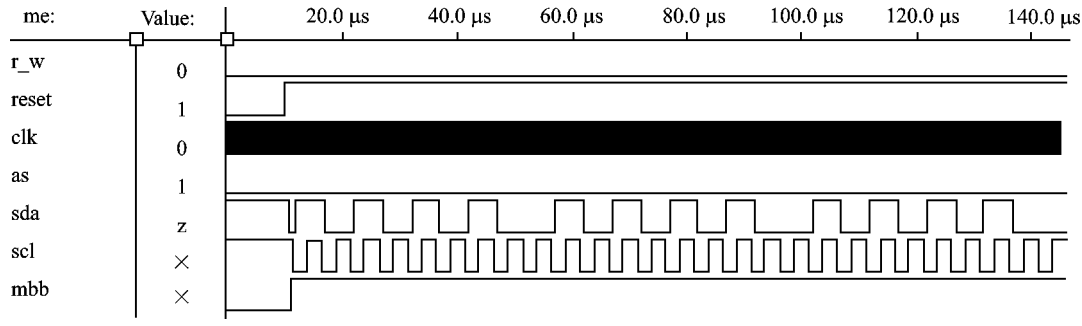


图 1.5-8 写数据时,SDA、SCL 上的数据传输

当从芯片中读数据时,串行数据线和时钟线上得到的仿真波形如图 1.5-9 所示。

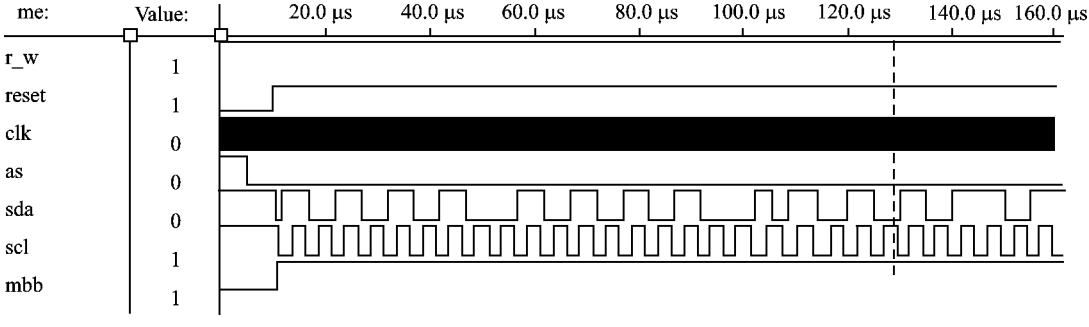


图 1.5-9 读数据时,SDA、SCL 上的数据传输

以上介绍了基于 I^2C 总线协议的 AT24 系列存储器数据串并转换接口的 VHDL 设计,该接口是针对 8 位微处理器而设计的。基于 FPGA 技术的基础上,把软件仿真、编译成功的程序,经 JTAG 电缆下载到以上指定的芯片上,用 89C51 与设计的接口进行数据通信,通过硬件验证,能实现它应具备的功能,可在通信系统中得到运用。

参 考 文 献

1 于宏军,赵冬梅. 智能(IC)卡技术全书. 北京:电子工业出版社,1996
2 何立民. I^2C 总线应用系统设计. 北京:北京航空航天大学出版社,1995
3 王志华,邓仰东. 数字集成系统的结构化设计与高层次综合. 北京:清华大学出版社,2000

选自《现代电子技术》月刊,2002 年第 6 期

1.6 低能耗嵌入式系统的设计

福建厦门大学计算机科学系 (361005)
许海燕 卢 伟 杨晨晖 吴 芸

一、背 景

为什么低能耗如此重要？无论是在军事上还是在商业贸易上的应用，嵌入式系统一般是由电池来供给电能的，而且大多数嵌入式设备都有体积和质量大小的约束。减少电能消耗不仅能延长电池的寿命，降低用户更换电池的次数，而且能带来提高系统性能与降低系统开销的好处，甚至能起到保护环境的作用。在便携式设备中，CPU 消耗的电能越少，电池的寿命就越长，同时，散发的热量少了，所需的散热器就越少，从而可减少该设备的花费和体积，进一步达到简化系统，使产品尽快进入市场的功效。

二、措 施

为了满足低能耗这一特性，必须在设计的每一部分及每一阶段减少能量消耗。首先，先描述一下本系统设计依据的一个能量等式[12]。

$$P = V^2 \cdot f \cdot C + Pstatic \quad (1)$$

式中，$V^2 \cdot f \cdot C$ 是动态能量消耗，大体上是由设计者控制的；Pstatic 与集成电路的静态电流有关，取决于硬模的特性、温度及供给的电压；V 为电压；f 是操作频率；C 是电容负载。从分析式 (1) 可看出，能量消耗与电压有平方的关系。因此，只需轻微降低电压就可显著地减少能量消耗。同样，电能消耗与操作频率成比例，当操作频率逐渐向 0 逼近时，能量消耗的动态部分也向 0 靠近。正是由于这个原因，选择处理器的速度最好与相应的应用所需的速度相当。否则如果采用的频率太高将造成极大的浪费，电池的寿命也将大大缩短。与此同时，CPU 时钟速度的良好管理也能节省不少能量。如果为了适应一个计算敏感性进程而把你的 CPU 时钟固定在一个很高的值上，当系统在执行另一个敏感性不强的进程时将损失大量的能量。最后，电容及元件的选择对电能消耗也有影响。

综上所述，可引入以下 4 个主要的优化措施。

1. 动态电源管理

动态电源管理 (Dynamic Power Management，DPM) 技术有选择地把闲置的系统成分置于低能状态，从而有效地利用电能。DPM 是建立在假设系统及其成分的工作负载各不相同，且工作负载的变化能较准确地被预测出来的基础上的。一个电源管理系统包含一个电源管理者 (Power Manager)，它能够基于对工作负载的观察来完成控制策略。该策略可采用不同方法来实现，如计时器、硬件控制器或软件控制。

一个电源管理系统可以用一个电源状态机来模拟，状态机的每个状态表示电能消耗情况及相应的系统性能。状态之间的转换需要一定的能量及延迟。一般来说，如果在一个状态上

耗能低,系统性能也较低,转换的延迟时间也较长。例如,一个硬盘系统可以表示为 3 个状态:活动态,此时硬盘可被读写;闲置态,可立即转换到活动态;睡眠态,此时转换到活动态性能损失较大。

电源管理者所采取的策略分为以下三种:预测性策略、适应性策略、随机性策略。其中预测性策略利用以前的工作负载情况来预测将来的闲置周期。预测技术的最简单的一个形式是超时设定(Timeouts)策略。超时设定策略的缺点在于当系统成分等待超时设定期满的过程也会消耗电能。有些预测性的停止策略对超时设定进行了改进,即当一个新闲置周期开始时就把系统成分转换到低耗能状态。为了处理非静态工作负载,一些适应性技术相继出现。这些技术主要是通过启发式的方法来调整超时时间、预测性及适应性策略军事启发性的。基于随机性模型的策略可取得最佳结果,结果的质量取决于所作的假设。测量结果表明这种方法可大大降低电能的消耗量。

2. 动态电压缩放

动态电压缩放(Dynamic Voltage Scaling,DVS)就是允许电压调节历程(Voltage Scheduler Routine)在运行时改变 CPU 的操作电压。电压调节历程首先分析系统状态,然后决定最佳的目标电压。文献[10]是通过实时期限调节来解决电压调整的问题。其中,进程被定义成 3 个参数组成的三元组:Si,Ci,Di。Si 是进程的开始时间,Ci 是所需的计算资源,Di 是进程的期限。开始时间与期限是固定不变的,而计算资源需求则被定义为当处理器全速运行时的执行时间。电压调节历程将根据以上的三元组来决定目标电压。

另一个减少能耗的渠道是降低时钟频率。虽然这种方法可减少电能消耗,但并不能显著地改变能量消耗,因为能量消耗与执行时间成正比。在降低电压的同时改变时钟频率可改变 CPU 的单操作耗能量,减少完成特定数量工作所需的能量。

3. 低耗能硬件设计

前面两种技术主要着眼于提高系统能源的利用率,即动态电源管理和动态电压缩放只能改进系统的使用。我们有必要在这个基础上,采用特定的方法来实现系统硬件和软件的节能设计。首先,介绍硬件的节能设计方法。可采用现成的产品,如附带 MMX™的 166 MHz 和 266 MHz 低耗能奔腾处理器。在奔腾处理器家族中这些处理器耗能量少,它们为嵌入式系统设计提供了简化而低开销的热管理方案。当核心电压为 1.9 V,I/O 电压为 2.5 V,系统在 266 MHz 时的最大电能消耗仅仅是 7.6 W。

除了处理器外,我们还需要一些数字逻辑来将处理器与其他子系统组合在一起。现代 CMOS 离散逻辑家庭拥有很低的动态电能消耗和几乎可忽略的静态消耗。目前,有不少可选的逻辑家族,但它们往往是能量消耗与速度不可两全齐美。74VHC 系列是个不错的选择。它不仅耗能少,而且能满足一般的微处理器应用。

4. 低耗能软件设计

在嵌入式系统的设计中,人们往往会认为只有硬件消耗能量。然而,这就好比我们假设在开车时只有汽车消耗汽油,而不是司机。在基于微处理器、微控制器的系统中,软件起到了应到硬件活动的主导作用。也就是说,软件对系统的能量消耗有很大的影响。直到最近,还没有有效且精确的方法可用来评估软件设计对能量消耗所起的效应,我们就无法对软件进行优化,进而减少电能消耗。

引起 CPU 电能消耗的众多因素中,至少有两个受软件的影响极大。在嵌入式系统的软件

设计中,我们主要考虑这两个因素:存储系统与系统总线。

⑴ 存储系统

由于几个原因,对存储器的存取耗能量高:①对存储单元的读写就是一个耗能操作。②把数据结构映射到多种内存条将影响多字并行装载的可能度,其中并行装载在提高系统性能的同时,还会起到节能的作用。程序的存取模式对系统的缓存性能影响极大,不恰当的存取模式会导致缓存失败,对存储器的访问也会相应增加。代码压缩可减少存取的指令数,降低缓存失败的可能性,也就减少了存储器的动作。这主要是由于访问缓存比存取主存耗能更少,而且缓存要比主存小得多。

在此引入了代码压缩的概念,文献[11]中描述了一种创新的算法,在指令压缩时,它为指令分配特定的代码,从而减少总线上的传输量,减少耗能量。在保证压缩性能和解码速度的前提下,该算法努力做到压缩率与总线耗能量之间的平衡,并指出高压缩率并不一定就会有最低的耗能量。采用这种方法,总线的耗能量将减少35%,且不会带来任何额外的硬件开销。

⑵ 系统总线

根据与每一总线相连接的模块数及总线路径的长度,在一个指令处理系统中的总线一般具有较高的装载容量。在这些总线上的转换动作,在很大程度上依赖于软件。指令总线上的转换是由将要执行的操作指令的顺序决定的。与此类似,地址总线上的转换是由数据和指令存取的顺序决定的。这两个效果可在编译时计算出来。但在编译时对数据相关的转换进行预测就要难得多,因为程序的输入数据多数是在执行时才给出的。

三、结 论

虽然人们在低能耗嵌入式系统设计的研究上已经投入了相当多的努力,但该领域至今还未完全成熟,仍然面临着许多难解的问题。随着嵌入式系统应用领域的不断扩大,嵌入式系统将面临更大的挑战,技术难题会接二连三地出现。但我们深信:没有困难就没有进步。

参 考 文 献

1 Johan Pouwelse, et al. Dynamic Voltage Scaling on a Low-Power Microprocessor [C]. 7th Int. Conf on Mobile Computing and Networking (Mobicom), Rome, Italy, July 2001

2 Johan Pouwelse, et al. Energy Priorits Scheduling for Variable Voltage Processors [C]. Int. Symposium on Low Power Electronics and Design (ISLPED01). Huntington Beach, CA, Aug 2001

3 J Pouwelse, et al. Power-aware Video Decoding [C]. Picture Coding Symposium 2001, Seoul, Korea, April 2001

4 Robert P Dick, et al. Power Analysis of Embedded Operating Systems [C]. Proceedings of the 37th conference on Design automation, New York, NY, USA, Pages:312 - 315 Series-Proceeding-Artile, ISBN:1 - 58113 - 187 - 9,2000

5 Michael J Schulte, James E Stine, Redued Power Dissipation Through Truncated Multiplication [J] IEEE Alessandro Volta Memorial Workshop on Low-Power Design, 4 - 5 March 1999

6 Johan Pouwelse, et al. A Feasible Low-Power Augmented-Reality Terminal [C]. lnt Workshop on Augmented Reality (IWAR 99) San Francisco, Califomia, October 1999

1.7 嵌入式应用中的零功耗系统设计

北京航空航天大学(100083) 何立民

从闭着眼睛走路说起

嵌入式应用系统中,普遍存在功耗浪费现象。如果将人比作一个嵌入式应用系统,人在行走时,系统处于连续运行状态,眼睛负责观察前方路况。通常在人行走的全过程中,眼睛都处于连续工作状态。然而,在实际行走中,并不要求对前方路况信息连续捕捉。假如眼睛对前方路况捕捉时间小于 0.5 s,人体盲目行走每米的横向偏差为 0.05 m,当路面允许最大横向偏差不大于 1 m 时,人行走在 20 m 范围内可不需要眼睛捕捉新的路况信息。这样,人便可以闭上眼睛走路,只在每行走 20 m 的周期中,将眼睁开 0.5 s 即可。当行走速度为 1 m/s 时,行走过程中眼睛的有效工作率仅为 0.5 s/20 s = 2.5%。由此看来,通常行走时,眼睛的"功耗"有 97.5% 都浪费了。

一、零功耗系统设计的基本概念

1. 系统中的理想功耗

一个电子系统要运行就会有功耗。如果系统运行时没有任何功耗浪费,那么它的功耗就是系统的理想功耗。

在一个嵌入式应用系统中,由于普遍存在 CPU 高速运行功能和有限任务处理要求的巨大差异,会形成系统在时间与空间上巨大的无效操作。如果在系统运行中,所有时间、空间上的无效操作都没有功耗,那么系统便处于理想功耗运行之下。

2. 应用系统中的有效操作时空占空比

如果将系统运行中,所有时间、空间上的有效操作和无效操作采用时空占空比来量化描述,那么,有效操作占空比定义为 :有效操作与系统全部运行操作之比。在一个具体应用系统中,有效操作的时空占空比有 :宏观时域占空比、宏观区域占空比、微观时域占空比和微观区域占空比。以下以一个嵌入式应用系统——热流量计为例来描述这 4 个占空比的概念。

(1) 有效操作的宏观时域占空比 T_{dc}

T_{dc} 定义为系统运行时域上有效操作时间 OP_{act} 与全部运行时间 OP_{tot} 之比。由于嵌入式应用中 CPU 的高速运行与有限任务操作的差异,常常会形成有效操作高谐小量的时域占空比现象。例如,在热流量计中,要采集、处理的物理参数有热水的入口温度、出口温度和流量计数值。由于这些参数的大惯量特征,在满足采集精度要求下,一次采集循环周期为 10 min,然而系统完成一次采集、处理、存储、送显示的时间只需 2 s,如图 1.7 - 1 所示。那么,该系统的有效操作时间 OP_{act} 为 2 s,全部操作循环时间 OP_{tot} 为 600 s,系统宏观有效操作时域占空比为

$$T_{dc} = \frac{OP_{act}}{OP_{tot}} = \frac{2}{600} \approx 0.33\%$$

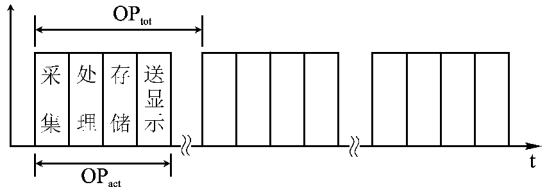


图 1.7－1

⑵ 有效操作的宏观区域占空比 S_{dc}

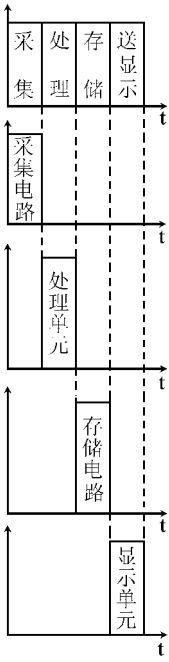


图 1.7－2

有效操作宏观区域占空比定义为：系统运行时，有效操作区域 S_{act} 与系统全部区域 S_{tot} 之比。由于系统运行时，并不是所有电路单元都处于有效操作状态，特别是在单 CPU 系统中，所有功能单元都是在 CPU 的轮流控制下运行，致使系统的各部分电路轮流进入有效操作状态。例如，在热流量计中，在有效操作时域 OP_{act} 中，除 CPU 外，采集、处理、存储、送显示的 4 个主体操作是轮流进行的，如图 1.7－2所示。如果按等区域原则最粗略地估算，可以算出该系统宏观有效操作的区域占空比为

$$S_{dc} = \frac{S_{act}}{S_{tot}} = \frac{1}{4} = 25\%$$

在系统硬件设计中，如果有意识地按任务进程，对系统电路进行粗略的划分，形成相对独立任务运行空间，这样便可较准确地计算出 S_{dc} 值。

⑶ 有效操作的微观时空占空比

在数字系统中，进入有效操作状态的一个完整电路中，也不是每一时刻、每一电路单元都处于有效操作状态，同样可以估算出微观有效操作的时域占空比和区域占空比。

① 有效操作的微观区域占空比 μS_{dc}

μS_{dc} 定义为：有效操作电路单元中，平均有效操作区域 A_{act} 与全部电路单元区域 A_{tot} 之比。例如，热流量计在执行数据存储任务，对 EEPROM 进行存储操作时，EEPROM 的 3 个操作区域，即输入缓冲电路、转换控制电路和 EEPROM 阵列轮流进入有效操作状态。设这三个区域有效操作功耗相等，那么，热流量计在数据存储时，存储器 EEPROM 的微观有效操作区域占空比为

$$\mu S_{dc} = \frac{A_{act}}{A_{tot}} = \frac{1}{3} \approx 33\%$$

② 有效操作的微观时域占空比 μT_{dc}

系统中，所有处于有效操作的电路，真正的有效操作只表现为“0”、“1”状态的变化操作。因此，电路有效操作的微观时域占空比 μT_{dc} 定义为：电路的动态时间 AT_{act} 与全部时间 AT_{tot} 之比。例如，在热流量计的数据采集任务中，频率测量的逻辑控制电路要根据温频传感器输出的信号脉冲，实现频率测量控制。这些操作控制都出现在脉冲的变化沿。设温频传感器输出的信号脉冲频率为 20 kHz，测控逻辑状态变化时间小于 100 ns，可以估算出，在数据采集任务中，

频率测量控制逻辑电路有效操作的微观时域占空比为

$$\mu T_{dc} = \frac{AT_{act}}{AT_{tot}} = \frac{100\ ns}{1/20\ kHz} = \frac{100\ ns}{50 \times 1\ 000\ ns} = \frac{1}{500} = 0.2\ \%$$

3. 高谐小量时空占空比与零功耗设计

(1) 实际系统中高谐小量的时空占空比

在嵌入式应用系统中,CPU 高速处理能力与实际任务操作状态以及系统中的微观静、动态的巨大差异,导致大量无谓等待状态,形成有效操作的时、空占空比现象。上述 4 类占空比现象,在许多嵌入式应用系统中都会存在,而且这 4 类占空比形成乘积效应。按照上述估算,热流量计总体有效操作的时空占空比 OP_{dc} 为

$$OP_{dc} = T_{dc} \times S_{dc} \times \mu T_{dc} \times \mu S_{dc} = \frac{2}{600} \times \frac{1}{4} \times \frac{1}{3} \times \frac{1}{500} = \frac{1}{1\ 800\ 000} \approx 0$$

从这里揭示了一个惊人的现状,即在一个嵌入式应用系统中,有效操作只是全部运行操作的高谐小量。这一特点是嵌入式系统零功耗设计的基础。零功耗系统按照有效操作时空占空比实行精细的功耗管理,非有效操作期间没有功耗,从而使系统功耗与原来相比达到趋于零的效果。早期提出零功耗概念,并实现零功耗设计的器件有 AMD 公司的 Flash 存储器 Am29SL800B。早先 Am28F800B 的功耗量级为 100 时,改进工艺并降低电压后的 Am29SL800B 为 20,而实现零功耗管理的 Am29SL800B 的功耗则小于 0.1。可见零功耗系统设计在降低系统功耗中的潜力。

(2) 零功耗系统设计基本要求

在不少实际的嵌入式应用系统中,虽然有效操作时空占空比不会是热流量计那样显著的高谐小量,但一般都会有 0.1% 的量级。如果能按照系统有效操作时空占空比实施精细的功耗管理,使无效操作期间没有功耗,就可实现系统的零功耗。

零功耗是一个工程概念。零功耗系统是指该系统中没有任何功耗浪费。因此,零功耗系统设计的基本要求如下:

① 系统中所有的电路单元都具有功耗管理功能,即该电路单元在非有效操作期间都能被关断 (没有功耗)。

② 系统具有按有效操作时空占空比实施精细功耗管理的能力,能做到“多干多吃、少干少吃、不干不吃、谁干谁吃”的系统功耗分配。

③ 对于系统无法企及的微观有效操作时空占空比的功耗管理,要求由电路静、动特性来满足功耗分配,即电路动态过程有功耗,电路静态时没有功耗。

二、零功耗系统设计的技术基础

零功耗系统设计的核心技术,是按系统中有效操作时空占空比来实现按需分配的功耗管理。不仅实现宏观有效操作时空占空比的功耗管理,还要实现微观有效操作时空占空比的功耗管理。因此,实现零功耗管理必须有相应的技术基础,这就是 CMOS 工艺的电路基础、嵌入式系统实时的智能化控制以及具有功耗管理功能的外围器件。这些技术基础可以满足零功耗系统设计的 3 个基本要求。

1. CMOS 工艺的电路基础

数字电路从 TTL 工艺转向 CMOS 工艺,对电路功耗特性产生最大影响的是静动态 (静态

是“0”、“1”的恒定状态,动态是“0”、“1”的跳变状态) 功耗特性的根本差异。正是这一差异诞生了电路系统功耗管理的概念与技术。图 1.7-3 是 TTL 电路和 CMOS 电路静动态功耗特性。图 1.7-3 (a) 为 TTL 功耗特性,图 1.7-3 (b) 为 CMOS 电路功耗特性。TTL 电路为电流注入型电路,静动态电流相近 ;而 CMOS 电路为压控型电路,只在动态下才消耗电流,静态电流为泄漏电流,理想情况下静态电流为零。根据数字电路的有效操作态只表现为电路的动态情况,那么,只有 CMOS 电路才能提供按有效操作时空占空比实施功耗管理,而且指出了 CMOS 电路功耗管理的基本原则就是系统的最大静态化设计。对于功耗管理无法企及的微观时空占空比,CMOS 电路静、动态特性能自动保证非有效操作时的极微功耗 (电路泄漏形成的功耗) 状态。

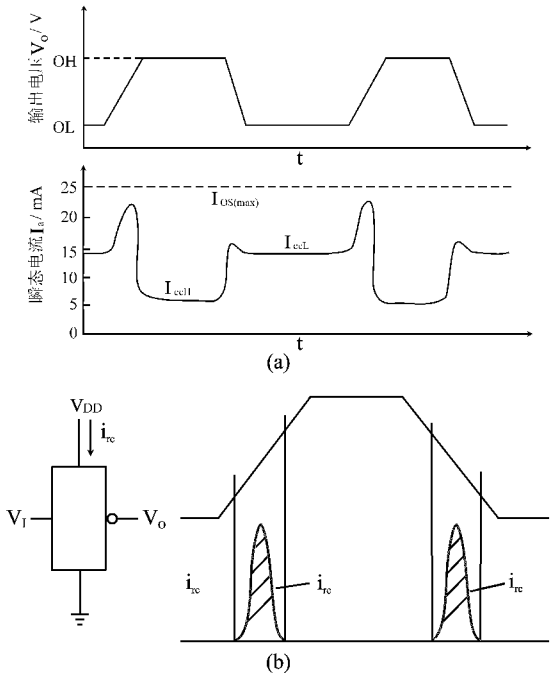


图 1.7-3

2. 嵌入式系统的实时功耗管理能力

嵌入式系统实时功耗管理能力,表现在能保证按照系统有效操作时空占空比来实现系统时空的最大静态化运行。其中核心的技术是系统中时钟与信号流的控制与调度。在系统无效操作的时间和区域上,终止或进入时钟运行,禁止开关、脉冲信号进入。

3. 外围器件功耗管理功能的保证

零功耗系统中所有的器件,包括处理器及外围器件,都必须具备功耗管理功能。目前,CMOS 的各类微处理器都具备有十分完善的低功耗模式。CMOS 外围器件中,有一部分具有自动的零功耗管理,不必微处理器的介入 ;许多 CMOS 外围器件都具有外部引脚控制或编程控制的功耗管理功能。

4. 电源管理的辅助技术

由于 CMOS 电路的静动态功耗特性,CMOS 电路的功耗管理遵循供电状态下的最大静态化原则。无论系统中的主器件还是外围器件的功耗管理都与指令控制相匹配,不必顾虑功耗转换的过渡过程。但当系统中不可避免地出现一些非 CMOS 功耗特性电路 (如传感器供电电

路)或一些模拟电路时,这些电路的功耗管理则须依靠电源供电管理方式。即这些电路退出有效操作时,关闭电源;待进入有效操作前开启供电线路。由于电路的时间常数,这些电路电源达到额定工作值或者进而启动时钟工作时,会有一个过渡期,不能即开即用,会给应用管理程序设计带来问题。

当前,嵌入式应用系统已走向全面 CMOS 化,嵌入式处理器中提供了由指令管理的多种低功耗模式,外围器件设置有许多低功耗控制功能,加上具有可局部关断功能的分布式供电体系以及电源总线开关等,为零功耗系统设计提供了十分现实的基础。

三、零功耗系统设计基本内容

按照最大静态化设计的基本原则,零功耗系统设计必须有最小量有效操作时空占空比的任务规划,设计出相应的硬件支持电路,并实现按有效操作时空占空比的功耗管理软件支持。因此,零功耗系统设计贯穿了应用系统设计的全过程。

1. 最小量有效操作时空占空比的任务规划

理论上讲,每个嵌入式系统都具有高谐小量的有效操作时空占空比;但若不认真将有效操作与无谓等待精细区分,而将有效操作与无效操作混在一起,就不可能实现系统的最大静态化管理。

⑴ 断续运行系统最小时空占空比的任务安排

对于可断续运行的系统,无论任务集中还是分散,都要努力寻求有效操作最小量的时空占空比。例如,热流量计中确定了采集、处理、存储、送显示 4 个任务时间 T_{OP} 后,任务的循环周期 T_{tot} 将决定宏观时域占空比的大小,即 $T_{dc} = T_{OP} / T_{tot}$。$T_{tot}$ 受温度变化率及测量精度的限制。在确知热水温度变化率和温度采集精度要求下,使 T_{tot} 最大来获得最小的有效操作时域占空比。

⑵ 连续运行系统的非连续化

将连续运行系统中的某些连续运行任务分离出来,实行非连续化,这样可以使连续系统的主体任务实现有效操作的占空比。例如,热流量计实际上是一个连续运行系统,因为它要不停顿地采集流量传感器的流量脉冲 Q_P。如果把流量脉冲采用极微功耗,独立的计数器不停地计数,热流量计只在数据采集任务中顺便读取计数器的计数值即可实现热流量计主体的最小量时域占空比。

⑶ 系统中各项操作任务相关区域的最小化与独立化

为保证系统能获取最小有效操作的宏观区域占空比,并据此实现区域的功耗管理,必须将每个操作任务限定在一个独立的最小区域内,使不同操作任务的电路相对独立。例如,时钟、信号通道可单独关闭;采用电源管理的区域设置单独的电源总线开关或采用 I/O 驱动供电等。

2. 系统硬件设计中的功耗管理电路设计

⑴ 满足宏观时空占空比功耗管理的独立电路设计。当按照最大限度宏观时空占空比来管理电路时,必须将这些电路设计成能独立实现静态化或实时关闭的电路单元和相应的管控电路。

⑵ 选择满足零功耗管理的外围器件。选择能自动实现零功耗管理的器件或可功耗管理的外围器件。

⑶ 最小值守电路设计。设计微功耗、高可靠性的开机值守、唤醒值守或运行值守电路。

⑷ 用电管理电路设计。在许多情况下,对于分时多区操作的独立电路单元,采用分布式

带关断功能的供电电路来实现功耗管理是十分有效的。例如,热流量计在采集完温度传感器的输出后立即将传感器电源关闭。

3. 功耗管理的应用软件设计

零功耗系统完全是在 CPU 的控制下完成功耗管理的,因此,它是依据总体设计要求,在系统硬件支持下,通过功耗管理的应用软件实现的。应用软件要遵循系统有效操作的时空占空比来及时关闭或唤醒相应的电路单元。

⑴ MCU、处理器、SOC 本身的零功耗管理。它包括内核的零功耗管理和核外功能单元的零功耗管理。

⑵ 外围器件的零功耗管理。它包括外围器件的功耗管理或电源供电管理。

四、零功耗系统与最小功耗系统设计

零功耗系统是基于功耗管理的低功耗系统,但只有零功耗系统设计并不能实现系统的最小功耗。因为在实际系统中,有效操作时系统的功耗过大以及非有效操作时系统的功耗远不为零,都会影响实际系统的最小功耗水平;而降低系统有效操作和非有效操作时空中的功耗水平,属于传统的低功耗设计技术。它是根据电路功耗特性参数来实现满足低功耗设计要求,在很多情况下并没有功耗管理的参与。例如,根据 CMOS 电路动态功耗特性,其动态功耗与供电电压、变换频率、负载电容等参数有关。降低系统供电电压,降低时钟频率,减少硬件电路设计制作时的分布电容等,这样可以减少有效操作电路中的功耗水平;减少 CMOS 电路的静态泄漏电流的措施,则可降低非有效操作时空电路上的功耗。只有充分实施了传统的低功耗设计和零功耗设计,才能获得系统的最小功耗。

五、结束语

⑴ 零功耗系统是一种工程概念。在这种系统中没有功耗浪费,所必需的系统功耗为传统电路功耗的高谐小量。

⑵ 零功耗系统设计是基于 CMOS 数字电路静、动态功耗特性的最大静态化的功耗管理设计。

⑶ 在嵌入式应用系统中,按系统有效操作的时空占空比,实现按需供给的功耗管理能最有效地、大幅度地降低系统功耗。

⑷ 对系统实现低功耗设计与零功耗设计可实现系统的最小功耗——微功耗。

⑸ 系统的微功耗以及便携化,使系统供电变得十分灵活与多样化,从而使传统的系统电源设计转向系统供电设计。

选自《单片机与嵌入式系统应用》月刊,2002 年第 1 期

1.8 数字指纹协议的研究与发展

上海交通大学计算机科学与工程系 (200030) 颜 浩 陈克非

一、引 言

随着电子商务的发展,以无形的数字格式作为多媒体产品的发布形式将成为 Internet 上的主流方式。但由于软件产品的易拷贝性,对软件产品 (如 JPEG、GIF 图像,应用程序,各种文档等) 的盗版就变得异常容易。从目前的技术研究来说,在开放环境下要严格防止非法使用者的侵权拷贝是不太可行的。主要的研究方向集中在如何有效地追查非法拷贝源,即发布商能够追查到这一份产品拷贝的原始购买者以向这个非法盗版者提起诉讼 (正版购买者应保护自己的产品拷贝不被其他人盗取)。这一类相关技术中比较主要的有数字水印技术、数字指纹协议体系等。

数字水印技术研究主要关注一个算法在数字产品 (主要针对数字图像) 中如何嵌入一串信息 (水印) 并能够完整地检测恢复。这一信息可以是发布商的名称,也可以是任意的有特定意义的 BIT 串。水印嵌入的要求是不可见性、鲁棒性,还要求能抵抗一定的主动攻击。

仅有水印嵌入检测算法是不足以实现版权的保护,我们需要一个完整的保护协议来规范商家和购买者之间的交易行为。既要保护商家的版权不受侵害,也要保护诚实的购买者的合法权益,因此,人们提出了数字指纹协议的概念。

数字指纹协议有比较大的适应性,在一定的前提下可以应用于各种类型的软件产品 (图像、文档等)。其基本思想是类似人类手指指纹的作用,在分发给每个软件产品购买者的产品拷贝中加入惟一的"指纹",当产品生产者发现侵权行为后,通过这一"指纹"来跟踪产品非法拷贝的源头,对盗版者进行指控,从而达到版权保护和威慑的作用。

用指纹协议来进行版权保护的思想最早是由 Neal R. Wagner 在 1983 年所提出的[1],后人在此基础上从不同的方面提出了各种实际的方案。1994 年 Chor 等提出了可以应用于付费电视广播系统中的 Traitor-tracing 协议[2],是指纹协议研究中的一个重要里程碑,以后的众多协议研究都沿用了其中的数学思想。1995 年 Boneh 等提出了另一种指纹协议[3],主要解决标记——"指纹"分发问题,这个协议可以和数字水印技术结合起来,从而在更多的应用环境中使用。1996 年 Pfitzmann 等提出了非对称的指纹协议的概念和基本协议体系[4],如同加密体系从对称模式发展到分对称模式一样,从另一个安全角度将指纹协议带入了一个新的研究方向。在此之后,人们又不断提出了新的指纹协议体系,如匿名指纹协议[5]、门限 Traitor-tracing 协议[6] 等。本文将对一些典型的指纹协议作基本的介绍和分析,并对数字指纹协议未来的研究发展方向和应用前景进行展望。

二、基本 Traitor-tracing 指纹协议的体系结构

Traitor-tracing 协议是由 Benny、Chor 等于 1994 年提出的,在详细描述这个协议之前举一

个典型的应用例子。在付费电视广播系统中,发行商(Data supplier)向他的所有合法用户广播加密的电视节目,如对每一个节目块(Block)使用不同的对称加密密钥,为了使合法用户能够解密所有节目块,发行商给每个合法用户分发一个惟一的个人密钥(为了能和非对称密码方案中的私钥 private key 加以区别,我们在这里使用个人密钥这个词),用户使用他的个人密钥解密出节目块中的对称加密密钥,从而解密出节目块中的内容。这个个人密钥就可以理解为指纹协议中的"指纹"。

这里有几个假设,任何用户都需要使用一个有效的个人密钥来获取节目的内容,而不能直接把节目的明文广播给非法的用户(这个做法可能是经济代价上不可行的);发行商不可能用每个合法用户的个人密钥直接加密节目或者说向每个合法用户广播完全不同的加密节目内容,因为这样广播代价太大而不可行;节目必须被分成许许多多的小块(Block),每个块使用不同的对称密钥加密(每个对称密钥本身的大小相对于 Block 的大小可以忽略),如果节目使用同一密钥加密,则一个合法用户只要用他的个人密钥解密出对称密钥发给一个非法用户则后者可以一直解密所有的节目。所以任何盗版者只能把自己的个人密钥发给非法用户使用或通过几个共谋盗版者之间通过比较他们各自的个人密钥来拼出一个不同的合法个人密钥发布。

考虑安全性,当某个想发布盗版的合法用户 Alice(Traitor)把他的个人密钥非法告诉一个非法用户 Bob(pirateuser),使 Bob 也能够解密出各个节目块中的内容,从而构成侵权行为。当发行商发现 Bob 的侵权行为后,他可以得到 Bob 所使用的个人密钥,发现这个密钥的原始所有者,从而抓到盗版源头 Alice,达到保护版权的目的。在后面的描述中将说明 Traitor-tracing 协议如何保证在 Traitor 共谋的情况下,发行商也可以至少追查出一个共谋盗版者,或者说这个概率是很大的。

更一般地,我们把 Traitor-tracing 描述成如下协议模型。

1. 协议中可能的角色

· 数据发布者(Data supplier 或 Merchant,下文中都使用后者):提供数字产品的商家。

· 合法用户(User 或 Buyer,下文中都是用后者):购买数据发布者产品的正版用户。

· 盗版者(Traitor):向非法用户提供个人密钥(指纹)的一个合法用户。

· 共谋者(Traitors):一组通过比较各自个人密钥进行侵权活动的盗版者。

· 非法使用者(Pirate user):使用 traitors 提供的个人密钥获得发布者产品的非正版用户。

· 仲裁者(Judge):仲裁 merchant 对 traitor(s)的诉讼。

2. 协议中的对象

· 商品(Item):数据发布者出售的一个产品(如一部电影),合法用户的一个个人密钥在一个同一个商品中都是有效的,每个 Item 被分成多个会话数据块。

· 会话密钥(Session key):对称密码体制的密钥(如流密码中的密钥)。

· 会话数据块(Session block):数据发布者向外广播(公布)的一段产品数据,每个会话数据块包含两部分:

· 加密块(Cipher block):由会话密钥加密的有效数据(明文),每个会话数据块中使用的会话密钥都不相同;

· 使能块(Enabling block):用分段并且对称加密的方法保存对应加密块使用的会话密钥。

· 个人密钥(Personal key):每个合法用户用来解密会话密钥的一个密钥集合。

3. 协议参数

- coll_size :协议可以保证安全的最大的共谋者的个数;
- L :会话密钥的分段个数,个人密钥的集合大小;
- b :一个任意字母表的大小,如字母表为 {1,2,3,…,b} ;
- N :Buyer 的最大数量。

协议可以简略地描述成如下的过程(见图 1.8-1)。可分成 4 个子协议。

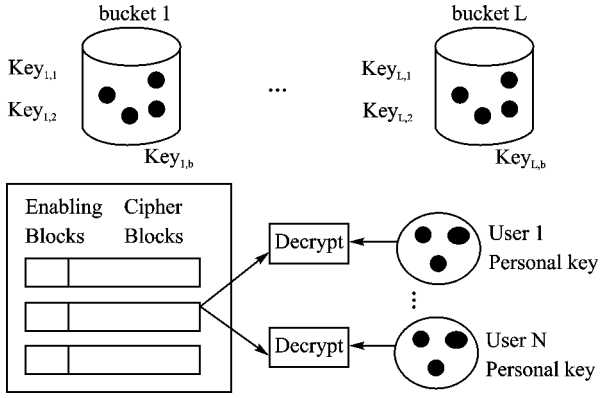


图 1.8-1

密钥初始化子协议 由 Merchant 执行。对于一个 Item,Merchant 随机选择 L 个对称密钥集合,每个集合有 b 个密钥,每个集合可以形象地比作一个密钥桶(Bucket),这 L × b 个对称密钥可以按字母顺序表示为 $key_{L,b}$。对于这个 Item 中每个会话数据块,选择一个会话密钥 S。

密钥指纹子协议 对每个 Buyer,Merchant 统一随机选择一个长为 L 的码字,其中每一位都是字母表 {1,2,3,…,b} 中的某个字母。这样一个码字就和 L 个密钥桶中 L 个密钥对应起来,这 L 个密钥构成一个 Buyer 的 Personal key。如码字为 {7,3,4,…,9} ,Personal key = $\{key_{1,7}, key_{2,3}, key_{3,4}, \cdots, key_{L,9}\}$。Merchant 保存每个 Buyer 的 Personal key,把 personal key 发给对应的 Buyer,personal key 对其他人是保密的。(码字的可能空间为 $b^L \gg N$。)

会话发送子协议 Merchant 广播(发布) Item 中每个会话数据块,每个会话数据块中 cipher block 使用这个块对应的 session key 加密。session key 被等长地分成 L 段 $S_1 \cdots S_i \cdots S_L$,其中第 i 段使用第 i 个密钥桶中的 b 个密钥进行加密 (i = 1,…,L)。这 L × b 个加密值按次序排列构成这个会话数据块的 Enabling block,随加密块一起发送。每个 Buyer 收到一个会话数据块后,使用自己的 personal key 从 Enabling block 中解密出对应的 $S_1, \cdots, S_L$,组成完整的 session key S,从而解密出对应加密块中的有效数据。

盗版跟踪子协议 对一组 coll_size 个共谋 traitors 来说,为了使一个 pirate user 可以成功解密所有 session block 中的数据,必须提供一个有效的 personal key。首先他们不会提供其中单个人的 personal key,否则 Merchant 直接从检测出的这个 personal key 查到 traitor,那么他们就要对 L 段中每个 S_i,从他们的 coll_size 个 personal key 中对应的 coll_size 个 $key_{i,j}$ 中选择一个作为解密 S_i 的密钥,使最终共谋产生的 personal key 是有效的,但是和 traitors 中的任意一个 personal key 都不同,来阻止追查。但是,该协议通过如下算法使 merchant 仍能确定其中的一

个 traitor：当事后查到了一个 pirate user，merchant 可以得到一个该 pirate user 使用的 personal key K（实际是每个密钥桶中对应一个 $key_{i,j}$）。对于这个 K 中 L 个 $key_{i,j}$（$i = \{1,\cdots,L\}$），merchant 对每个其 personal key 中对应位置等于 $key_{i,j}$的 Buyer 作一次标记，如 $K = \{key_{1,7}, key_{2,3}, key_{3,4}, \cdots, key_{L,9}\}$，某个 Buyer 的 personal key = $\{key_{1,7}, key_{2,8}, key_{3,4}, \cdots, key_{L,6}\}$（假设…位置中的对应 key 都不相同），则这个 Buyer 被标记两次（第 1、3 两个密钥桶）。最后被标记次数最多的 Buyer 就被认为是 traitors 之一。（实际上这个 Buyer 至少被标记 L/coll_size 次，并且共谋所伪造的 personal key 等于一个合法 Buyer 的 personal key 的概率根据参数设置是可忽略的，以保证共谋不能陷害一个诚实的 Buyer，这个算法的证明以及共谋不可陷害的证明在文献 [2] 中进行了详细的证明。）

三、非对称 Traitor-tracing 协议

Pfitzmann 等将非对称的概念应用于 Trait-tracing 协议，以改善协议的安全性和公平性。简单来说，在基本的 Trait-tracing 协议中，Merchant 和 Buyer 都知道卖给 Buyer 的个人密钥（Personal key），那么在发现盗版提起诉讼时，Buyer 完全可以声称 Merchant 发现的 Personal key 是 Merchant 的一个不诚实的员工所流传出去的，从而使仲裁无法确认谁是 traitor（对称性）。

非对称的概念就是要使标识 Buyer 的“指纹”只有 Buyer 自己知道（自己选择生成其个人密钥的码字），而 Merchant 在商品交易时不知道，但可以确认。在 tracing 时 Merchant 可以得到足够的证据（Proof）来指认相关的 traitor。

非对称的概念也被用于基于码字安全的指纹协议中。

非对称 Traitor-tracing 协议的基础仍基于上文简述的基本 Traitor-tracing 协议，包括参与的角色和对象。为了实现非对称性，需要增加 3 个密码学体系：1 个公钥签名模式、1 个承诺模式和 1 个安全 2-party 计算。在这里我们不讨论这些模式的概念和实现，假设都已存在。

协议参数：

σ：一个适当的概率参数以表示安全的系数；

coll_size：协议可以保证安全的最大的共谋者的个数；

L：会话密钥的分段个数，个人密钥的集合大小（这里我们选定 $L = 64 \times coll\_size \times (\sigma + \log_2(N))$）；

b：一个任意字母表的大小，如字母表为 $\{1,2,3,\cdots,b\}$（这里我们选定 $b = 48 \times coll\_size$）；

N：Buyer 的最大数量。

协议分成 6 个子协议：

Buyer 签名密钥生成　每个购买者（Buyer）生成一对签名模式使用的公私钥（sk_B，pk_B）并公布其 pk_B。

密钥初始化子协议　和基本（对称）协议中的对应子协议相同，merchant 选择 L 个密钥桶（Bucket），每个桶有 b 个对称密钥。

密钥指纹子协议　Buyer 选择一个长为 L 的码字 $word_B$（这一步在基本协议中是由 merchant 完成的）。Buyer 对 $word_B$ 生成一个承诺 com_B，并把这个承诺发送给 merchant，用来揭示承诺的信息称为 $open_B$。Buyer 使用自己的签名私钥 sk_B 对如下消息进行签名：msg_B = (text, com_B)，生成 sig_B。其中 text 是一个字符串，只要足以说明签名消息的含义。Merchant 使用 pk_B 验证 sig_B 是对 msg_B 的一个有效的签名。

接下来执行安全 2 - party 计算：

输入：

Buyer：$word_B$ 和 $open_B$

Merchant：在密钥初始化子协议中生成的 L × b 个对称密钥，一个随机选择的大小为 L/2 的集合 Set_B（{1，…，L} 的子集），com_B

计算：

验证 $open_B$ 可以打开承诺 com_B，否则协议失败

验证 Set_B 的大小最多为 L/2 个元素

$word_B$ 中符号对应集合 Set_B 中元素的位置组成的有序符号集称为 $halfword\_trace_B$，其余部分称为 $halfword\_evid_B$

输出：

Buyer：由 $word_B$ 对应生成的个人密钥（Personal key）

Merchant：$halfword\_trace_B$（这样，Merchant 只知道 $word_B$ 中的一半符号而不能猜出整个 $word_B$，仍符合我们的非对称性）

这样在执行完安全 2 - party 计算后，Merchant 得到的交易记录 $record_M = (id_B, text, com_B, sig_B, Set_B, halfword\_trace_B)$，Buyer 得到的交易记录 $record_B = (text, word_B, open_B)$。

会话发送子协议　和基本协议中对应的子协议相同。

盗版跟踪子协议　如同基本协议中一样，Merchant 发现一个 pirate user，得到一个个人密钥，也就相当于找到一个长为 L 的码字 $word_{found}$，它搜索所有交易记录（Set_B，$halfword\_trace_B$），找到一个记录，其 $halfword\_trace_B$ 和 $word_{found}$ 至少有 L/（4 × coll_size）个对应位置上符号是相同的。他取出 $msg_B = (text, com_B)$ 和 sig_B 准备向仲裁者控告此 Buyer。

仲裁子协议　Merchant 组成的证据为 $proof = (msg_B, sig_B, word_{accuse})$，其中 $word_{accuse}$ 长为 L 的码字，码字中对应 Set_B 中元素的位置上由 $halfword\_trace_B$ 中的对应符号组成，码字中其他位置的符号由 $word_{found}$ 中的对应位置符号组成。

Buyer 提供 $open_B$。

仲裁者首先验证 sig_B 签名的有效性和承诺的有效性（揭示 $word_B$）。

然后验证 $word_{accuse}$ 和 $word_B$ 是否至少在 L/2 + L/（16 × coll_size）个对应位置上的符号相同。成立则指控成功，否则 Buyer 即可否认指控。

为了使 $word_B$ 始终对 Merchant 保密，可以用零知识证明来给出 $word_B$。

四、其他主要的指纹协议

上面介绍的两个协议是典型的对称和非对称的数字指纹协议，除了这些以外，对称性的数字指纹协议还有如基于码字（标记）安全的指纹协议[3]，与这个协议相对应的非对称协议[4,7,8]，以及现在研究比较多的匿名数字指纹协议等[5,9]。其他和这方面研究相关的论文有文献[10～20]。下面简单介绍一下这些协议的概念。

1. 基于码字（标记）安全的指纹协议

这种指纹协议主要考虑如何构造一种安全的码字（码字是嵌入在如图像之类的软件产品中的），来防止多人的共谋攻击。其中应用了纠错码的许多方法。协议本身并没有规定码字是如何嵌入产品的，但提出了一个 Marking Assumption 来限制嵌入方法的功能。这个协议可

以和比较强的数字水印嵌入算法一同工作来保证产品的安全性。

2. 非对称的基于码字安全的指纹协议

如同非对称的 Traitor-tracing 协议,为了保证 Buyer 的安全性,把非对称的概念也引入了码字安全的指纹协议。其构造类似于非对称的 Traitor-tracing 协议。

3. 匿名指纹协议

这是对指纹协议在非对称以后的一大改进,其目的是保持 Buyer 在进行交易时能不对 Merchant 泄露其真实身份,就如同我们平时在商店购买商品时不需要对售货员出示身份证一样。但是,在发现产品被非法散播时仍需能检测出盗版的 Traitor。匿名指纹协议中引入了一个注册中心的角色(RC),Buyer 使用在注册中心登记的身份而不是其真实身份同 Merchant 进行交易(这里的身份可能就是某种数字签名体系中的公钥),只要 Buyer 是诚实的,Merchant 就不会从指纹协议的过程中知道 Buyer 的身份,否则在有 RC 参与的子协议中进行仲裁。

五、对主要数字指纹协议的分析和展望

从指纹协议的发展来看,早期的对称性的数字指纹协议提出了基本的指纹协议体系结构,奠定了数学基础,是后来所有协议的基石。这些协议研究的重点是使用类似纠错码或其他方法来构造惟一的"指纹",并使用概率论的工具来证明协议的安全性。它们主要关心的安全问题是防止 N 个人的共谋攻击,这类协议中最具有代表性和可行性的就是前面重点介绍的基本 Traitor-tracing 协议。它不依赖其他技术的研究发展情况,协议的提出就是针对广播加密节目问题的,使其应用性非常强。而基于码字安全的指纹协议适应性比较强,它不指定"指纹"嵌入方法,只要满足它所要求的 Marking Assumption。但是如数字水印等相关信息隐藏方法还不能从理论上证明足够的鲁棒性,所以这种指纹协议在应用上会有比较大的障碍。对称性数字指纹协议的缺点,我们总结为两条:一是考虑的安全因素不够多,仅仅针对共谋攻击,这也是以后非对称协议提出的原因之一;二是随着共谋者 N 的增加,协议执行需要的数据量会大大增加,系统假设的 N 不能太大,就这一点来说,如何在保证安全性的条件下降低协议的复杂性是一个重要的研究方向,这很大一部分依赖于纠错码技术的发展。

非对称数字指纹协议的提出使指纹协议的研究迈上了一个新的台阶,只有购买者自己知道"指纹"的内容,避免了因双方都保有"指纹"从而无法仲裁的问题,既保证了盗版者的不可抵赖性,又保证了诚实的购买者不会被不诚实的商家或其工作人员所陷害。由于非对称数字指纹协议使用了安全两方计算等密码学技术,从执行效率来说会很大程度依赖于这些技术的效率,因此选择合适的相关技术方案是非对称指纹协议在应用时需要研究的问题。

而匿名协议则是目前的研究特点,使指纹协议更贴近于现实中的商品交易过程。匿名性已经成为继非对称性之后对数字指纹协议的基本安全要求。由于安全要求的增加,指纹协议使用的密码技术也随之增加,使协议的复杂性更是大大增加,这很不利于实际应用。我们认为如何简化数字指纹协议,提出技术依赖性比较小,执行速度快的匿名协议是使数字指纹协议实用化的一个重要的研究方向。

六、结 论

数字指纹协议为解决电子交易中的盗版问题提出了可行的密码学体系,从各种安全性角度解决了不同的安全问题。数字指纹协议的提出,使数字产品的无形交易过程变得更安全可

靠。越来越多的数学密码学工具被用来实现更安全的非对称性和匿名性。数字指纹协议有着广阔的应用前景,从数字图像、数字电影电视节目、数字音乐,到应用程序、数字文档,各种数字产品的交易都可以和数字指纹协议结合起来以保证产品版权的安全性。因此,除了理论的研究,如何将指纹协议结合到各种现实的电子交易应用系统中去也是非常需要解决的问题,毕竟理论研究的目的是为了现实的应用。

参 考 文 献

1 Wagner N R. Fingerprinting;1983 Symposium on Seurity and Privacy, IEEE, Oakland, California, 18~22

2 Chor B, Fiat A, Naor M. Tracing Traitors;Crypto 94, LNCS 839, Springer-Verlag, Berlin, 1994, 257~270

3 Boneh D, Shaw J. Collusion-Secure Fingerprinting for Digital Data;Crypto 95, LNCS 963, Springer-Verlag, Berlin, 1995, 452~465

4 Pfitzmann B, Schunter M. Asymmetric Fingerprinting;Eurocrypt 96, LNCS 1070, Springer-Verlag, Berlin, 1996, 84~95

5 Pfitzmann B, Waidner M. Anonymous Fingerprinting;Eurocrypt 97, LNCS1233, Springer-Verlag, Berlin, 1997, 88~102

6 Naor M, Pinkas B. Threshold Traitor Tracing;Crypto 98, LNCS 1462, Springer-Verlag, Berlin, 1998, 502~517

7 Pfitzmann B, Waidner M. Asymmetric Fingerprinting for Larger Collusions;accepted for 4th ACM Conference on Computer and Communications Security, 1997

8 Bieh1 I, Meyer B. Protocols for Collusion-Secure Asymmetric Fingerprint-ing;STACS 97, LNCS 1200, Springer-Verlag, Berlin, 1997,399~412

9 Domingo-Ferrer J. Anonymous Fingerprinting of Electronic Information with Automatic Identification of Redistributors;Electronics Letters 34/13, 1998, 1303~1304

10 Blakley G R, Meadows C, Purdy G B. Fingerprinting Long Forgiving Messages;Crypto 85, LNCS 218, Springer-Verlag, Berlin, 1986,180~189

11 Garonni G. Assuring Ownership Rights for Digital Images. In:Proc. VIS 95;Vieweg, Wiesbaden, 1995, 251~263

12 Canetti R. Studies in secure multiparty computation and applications:[Ph. D. dissertation]. MIT, 1999

13 Cox I J, Kilian J, Leighton T, et al. Secure spread spectrum water marking for multimedia. IEEE Transactions on Image Processing,1997,6(12):1673~1687

14 Fiat A, Naor M. Broadcast Encryption;Crypto 93, LNCS 773, Springer-Verlag, Berlin, 1994,480~491

15 Lai X, Massey J. Hash functions based on block ciphers, Eurocrypt 92, Springer-Veralg, 1992,55~70

16 Ostrovsky R. An efficient software protection scheme;Crypto 89, LNCS 435, Springer-Verlag, Heidelberg, 1990,610~611

17 Petitlas F A P, Anderson R J, Kuhn M G. Information hiding-A survey. Proc. of the IEEE, 1999,87(7):1062~1078

18 Roe M. Performance of block ciphers and hash functions-one year later, FSE 94. In:Proc. of Second Intl. Workshop on Fast Software Encryption, Springer-Verlag, 1994, 359~362

19 Salvail L. Quantum bit commitment from a physical assumption, Crypto 98, Springer-Veralg, Berlin, 1998, 338~353

20 Yao A C. Protocols for secure computations, 23rd STOC. In:Proc. of the IEEE 23rd Annual Symposium on the Foundation of Computer Science, 1982, 160~164

选自《计算机科学》月刊，2002 年第 12 期

1.9 指纹识别控制系统设计

武汉大学电气工程学院(430072) 刘黎明 史 进
中发国际资产评估有限责任公司(100021) 刘慧环

指纹识别技术是把现场采集到的指纹特征数据与数据库中的指纹特征数据进行同认定,从中找出完全相同的指纹特征数据,从而确定主体身份的过程算法、技术设备及管理运行等方面的技术总和。指纹技术不仅广泛用于司法部门的鉴定和军事与工业自动化高级控制系统中,而且扩充到商业、服务业及个人住宅等领域。本文以酒店的门控系统为例,对其进行说明。

一、系统简介

本设计的 FDA01 指纹识别模块采用 CMOS 光学采集技术,能准确地捕捉干湿手指指纹,形成高对比度、无畸变的指纹图像。该采集镜具有高度抗静电和抗物理破坏能力,一般性的人为破坏如划痕、撞击及重压等不影响其正常工作。指纹识别系统软件采用国际领先的指纹匹配算法,能快速提取指纹特征数据,并准确比对。在运算过程中,指纹特征数据为加密状态,可有效避免通过其他途径伪造、窃取和复制指纹特征数据。本系统将 FDA01 指纹认别模块和单片机控制系统结合,用来控制指纹锁,有进出记录的备份、实时时钟、开门记录查询功能;可滚动存储 250 条开门记录。低功耗设计延长电池使用寿命,亦可选用外接电源。指纹的录入和删除通过手持机和管理权限来实现,操作简单;RS-232 通信使得该锁非常方便与手持机和 PC 机连接;使用安全级别可调,能满足不同安全级别的应用;具有很强的安全性和可靠性,1:N指纹比对开门,连续 5 次指纹对比失败,门锁控制系统控制门锁自锁 5 min。

二、硬件系统设计

指纹锁控制系统主要由单片机 AT89C52、FDA01 指纹识别模块、可编程通信接口、串行的存储器、串行时钟、低电检测和稳压、看门狗监视、通信控制电路等组成。指纹登记是事先通过验证记录在 PC 机中,包括许多合法的 ID 号,根据不同的需要先通过电平转换器 MAX232E 转换成 TTL 电平后,送入指纹锁控制系统中,并将 ID 号存放在串行 E^2PROM 中,验证时,从中取出相应的信息与指纹识别模块回送信息相比较,确定是否为合法用户。控制系统原理如图 1.9-1 所示。

1. FDA01 指纹识别模块

FDA01 指纹识别模块用来完成指纹识别。它将指纹图像转成数据,存储在其中,并且与手持机和 PC 机相连,将指纹特征数据送入其中,用来与数据库的指纹特征数据进行比较。它主要由以下部分组成。

(1) 处理单元:FDA01 处理单元包括一个 32 位处理器和高性能指令和数据缓存。

(2) FLASH 记忆本:FDA01 脱机处理系统的内置处理器有 1 MB 的快闪记忆体,用于指纹特征数据的存储。

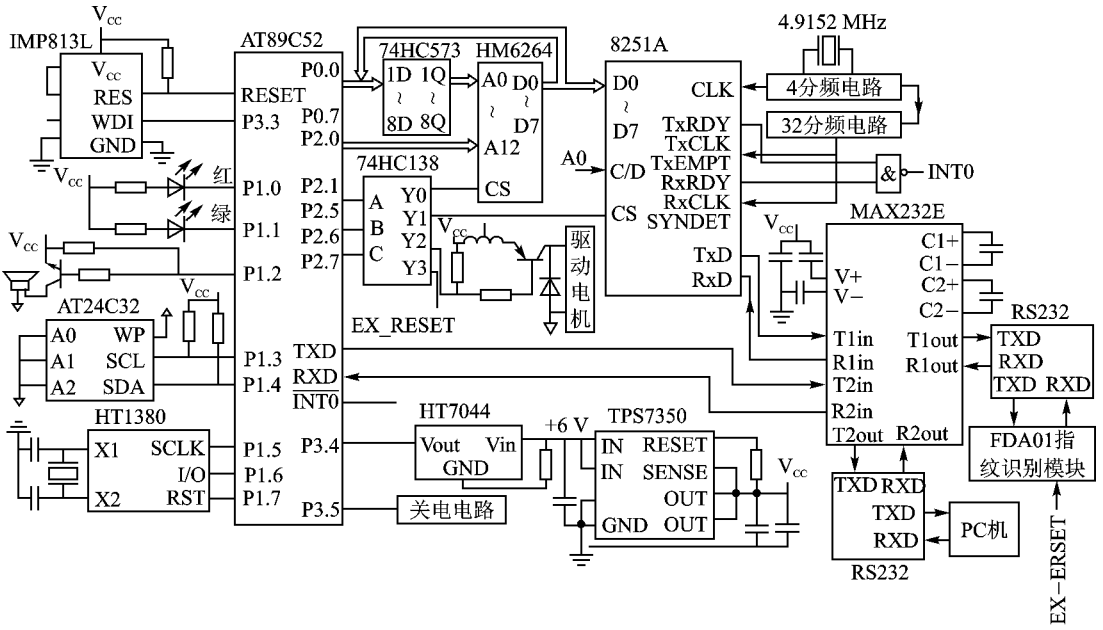


图 1.9-1 指纹锁控制系统原理框图

(3) RAM 如数据被采用,数据将转移到动态存储器中,正在运行的程序及数据也在动态存储器中。FDA01 的动态存储器容量为 1 MB。

(4) PLD (可编程逻辑控制设备):使用简单的逻辑电路控制光学采集器。

(5) 指纹采集单元:CMOS 指纹采集传感器通过光学指纹采集头采集指纹信息。

(6) 电源:5 V 直流电源输入。

2. 串行时钟电路

串行时钟电路的核心器件是串行时钟芯片 HT1380。它是 HOLTEK 公司推出的一款带秒、分、时、日、星期、月、年的串行时钟保持芯片,具有低功耗工作方式,并用若干寄存器存储对应信息,一个 32.768 kHz 的晶振校准时钟。为了使用最小引脚,HT1380 使用一个 I/O 口与 AT89C52 相连,仅用 RST 复位、SCLK 串行时钟、I/O 口数据线就可传送 1 字节或 8 字节的字符组。其性价比极高,工作电压为 2.0 ~ 5.5 V;最大输入串行时钟:2.0 V 时为 500 kHz,5.0 V 时为 2 MHz;工作电流:2.0 V 时至少 300 nA,5.0 V 时至少 1 A;它与 TTL 兼容,采用串行 I/O 口传送数据,可单字节和多字节传送(字符组方式),所有寄存器都以 BCD 码格式存储。这里用 HT1380 给控制系统提供准确的时间,来记录开门时间等。其中校准时间由 PC 机通过串口送入,使控制系统时钟与 PC 机时钟同步。

3. 看门狗监视电路

为防止程序进入死循环,设计中增加了外部硬件看门狗定时器 IMP813L,当其内部看门狗定时器超过 1.6 s 时,$\overline{WDO}$拉至低电平,直到看门狗被清零才变为等电平。此外,当 V_{CC}低于复位门限时,$\overline{WDO}$保持低电平,和 RESET 不同,$\overline{WDO}$没有最小脉冲宽度,只要 V_{CC}超过复位门限,$\overline{WDO}$就变为高电平而没有延迟。将$\overline{WDO}$和$\overline{WR}$连接可使看门狗超时产生复位。IMP813L 内的看门狗定时器监控的工作。如在 1.6 s 内未检测到其 μP/μC 工作,内部定时器将使看门狗输出$\overline{WDO}$处于低电平状态,$\overline{WDO}$将保持低电平直到 WDI 检测到 μP/μC 的工作。

三、软件设计

本文针对酒店管理系统中关键部门的门锁控制设计系统软件，主要包括主程序和 8251 串行通信中断处理程序。主流程如图 1.9－2 所示，通信中断处理程序如图 1.9－3 所示。

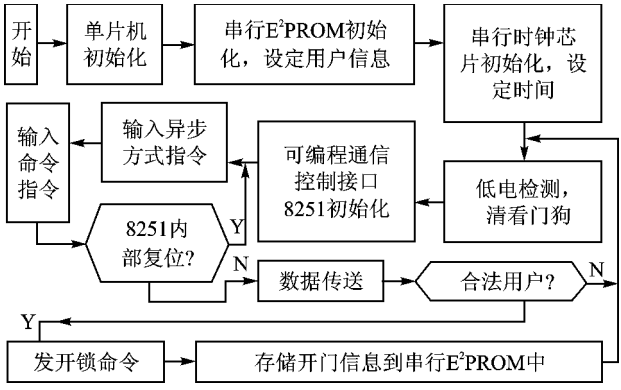


图 1.9－2　指纹门锁控制系统主流程图

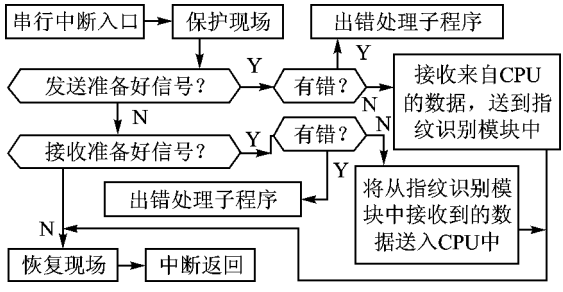


图 1.9－3　8251 中断服务程序流程图

四、结束语

采用先进的 FDA01 脱机指纹识别模块和单片机控制系统相结合的指纹锁控制系统，取代了传统的钥匙、密码或 IC 卡等门锁控制系统，避免了 IC 卡伪造、密码破译、钥匙丢失和盗用等弊端。是进门方便、快捷的高安全性智能化控制系统。随着指纹识别技术的进一步发展，其应用领域也将越来越广。

参 考 文 献

1　何立民. 单片机应用系统设计. 北京 北京航空航天大学出版社，1991

2　李华. MCS－51 系列单片机实用接口技术. 北京 北京航空航天大学出版社，1993

3　何立民. I²C 总线应用系统设计. 北京 北京航空航天大学出版社，1995

选自《电工技术》月刊，2002 年第 11 期

1.10 条形码的计算机编码与识别

北京理工大学机电工程学院 (100081) 何 军 康景利

一、条码技术现状

条形码技术起源于 20 世纪 40 年代,近几年来发展迅速,在国际上得到广泛的应用。比较有代表性的一维条码包括 UPC—E、EAN—8、EAN—13 等。随着信息工业的发展,在传统条码(一维条码)的基础上发展了二维条码技术,20 世纪 90 年代初首先被美军运用于军队管理、后勤保障和作战指挥。而后迅速从军事应用扩展到社会、经济和科技领域。目前,根据二维条码实现原理、结构形状的差异,可分为堆积式或层排式 (Stacked BarCode) 和棋盘式或矩阵式二维条码两大类型。堆积式二维条码编码设计、校验原理等方面继承了一维条码的特点,所以应用比较广泛,有代表性的包括 Code 49、PDF417、Code 16K 等。本文所讲的二维条码主要是指堆积式二维条码。

二、条形码的计算机编码及绘制

条形码的编码就是将可读信息转换成计算机用于绘制条码的码值,然后加上必要的附加信息,对于一维条码来说是起始符、中间分隔符、错误纠正码词和结束符等,对于二维条码来说还包括左层指示符及右层指示符。在 VC 环境下,绘制条码可以采用绘制直线的方法完成。

1. 一维条码的计算机编码

以 EAN—8 码为例,一个 EAN—8 码包含 7 位数据和一个校验数字,结构如图 1.10-1 所示。

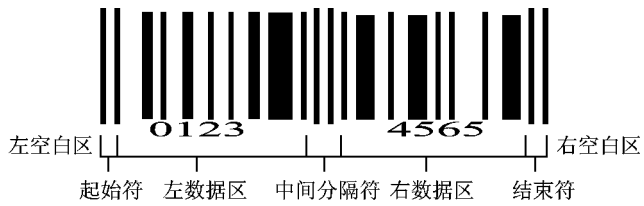


图 1.10-1 一维条码 EAN—8 结构示意图

首先根据可读字符 (0123456) 计算出校验字符 (5),校验字符的计算方法请参考文献 [1]。然后将得到的完整可读字符 (01234565) 转换成绘制条码用的码词,码词由 0 和 1 组成,0 表示一个单位模块宽的空,1 表示一个单位模块宽的条,比如左数据区的数字 3 对应码词 0111101。可读字符和码词之间具有一对一的对应关系,转换过程可以通过查询数组来完成,程序片段如下:

```
for (i = 0 ;i < 4 ;i + + ) {
    temp = EAN_8_L_ODD [m_strNum. GetAt (i) - 48] ;          // 得到左数据区的码词
```

```
        strCode + = temp;
    }
    for (i = 4; i < 8; i + +) {
        temp = EAN_8_R_EVEN [m_strNum. GetAt (i) - 48];          // 得到右数据区的码词
        strCode + = temp;
      }
```

然后根据得到的码词字符串 strCode 绘制条码，注意还必须在开始加上起始符（码词为101）、中间间隔符（01010）和结束符（101）。绘制条码的函数部分代码如下：

```
void CBar View::Draw1 DimBarCode (CString str)    // str 为包含码词的字符串
{…… (略)
char c;                                           // 0 或者 1
for (int i = 0; i < str. GetLength (); i + +) {
    char c = str. GetAt (i);
    if (c = = '1') {                              // 如果遇到 1 的话则画单位模块宽的条
            dc. MoveTo (m_CurPos);                // m_CurPos 为表示当前绘图位置的 Cpoint 对象
            dc. LineTo (m_CurPos. x, m_CurPos. y + m_intHeight);
            m_CurPos. x + = m_intWidth;           // m_int Width 为单位模块宽度
        }
        else if (c = = '0') {                     // 如果遇到 0 的话则留单位模块宽的空
            m_CurPos. x + = m_int Width;
        }
}
…… (略)
}
```

2. 二维条码的计算机编码

以 PDF417 为例，PDF417 是一种多层、可变长度、具有高容量和纠错能力的连续型二维条码。结构如图 1.10－2 所示。每一个 PDF417 条码可以包括 3～90 层。每一层由起始符/终止符、左/右层指示符及 1～30 个符号字符组成。每一个符号字符由 17 个模块构成，其中包括 4 个条和 4 个空，每一个条、空由 1～6 个模块组成。由于层数及每一层的符号字符数是可变的，故 PDF417 符号的高宽比可以变化以适应不同的需要。

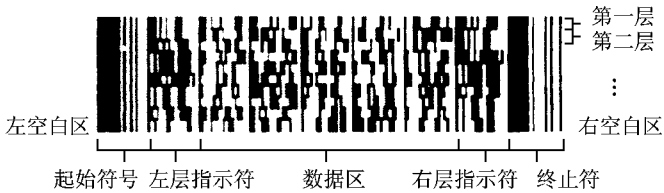


图 1.10－2 二维条码 PDF417 结构示意图

对于堆积式二维条码，编码方法从根本上与一维条码有相似之处，但也存在很大不同。对于 PDF417 条码，它有三种编码方式：文本组合模式、字节组合模式和数字组合模式。其中文本组合模式将两个字符以 30 为基数组合成一个码词，在表达文本时效率比较高；字节组合模

式通过基 256 到基 900 的转换,将字节序列转换成码词序列 数字组合模式可以将约 3 个数字位用一个码词表示,在表示数字时效率较高。

PDF417 的符号字符集分为 3 个族。每一个族中均有以不同条、空形式表示的 929 个符号字符 (其值为 0 ~928)。3 个族的逻辑族号为 0,3 及 6。也就是说每个码词对应 3 种符号字符。每一层仅用 3 族中的一族符号字符来表示数据。因此,同一族的符号字符每三层出现一次。族号与层号的关系如下:

$$族号 = [(层号 - 1) \bmod 3] \times 3$$

PDF417 的层指示符计算方法如下:

左层指示符 :当 Ci = 0 时,Li = 30 × INT (层号/3) + INT [(层数 - 1) /3]

当 Ci = 3 时,Li = 30 × INT (层号/3) + 错误纠正等级 ×3 + (层数 - 1) mod 3

当 Ci = 6 时,Li = 30 × NT (层号/3) + 每层列数 - 1

其中 Ci 代表每层的族号,Li 代表第 i 层左层指示符

右层指示符 :当 Ci = 0 时,Ri = 30 × INT (层号/3) + 每层列数 - 1

当 Ci = 3 时,Ri = 30 × INT (层号/3) + INT [(层数 - 1) /3]

当 Ci = 6 时,Ri = 30 × INT (层号/3) + 错误纠正等级 ×3 + (层数 - 1) mod 3

其中 Ci 也代表每层的族号,Ri 代表第 i 层右层指示符

例如一个 PDF417 符号有 3 层,每层 3 个字符,错误纠正等级为 1,则左/右层指示符分别为 (0,5,2) 和 (2,0,5)。

二维条码编码最重要的还是将数据 (汉字或 ASCII 码) 转换成符号字符的过程,这一过程主要采用文本组合模式。可以将数据与文本模式下的值的对应关系保存在一个数据库的一张数据表中,这样在转换时可以通过 SQL 语言查询数据库来实现,在 Microsoft VC + + 中使用 ODBC 连接数据库速度是很快的。下面是通过 ODBC 接口查询并保存字符串 m_strText 中各个字符在文本模式下对应的值的关键语句:

```
if (! m_CharCodeSet. IsOpen ()) m_CharCodeSet. Open (CRecordset::snapshot,"CharCode")
                                                    // 打开数据库,数据库名为 Charcode. mdb
for (i = 0 ;i < nChar ;i + + ) {                    // 依次查询字符串 m_strText 中的所有字符
    c = m_strText. GetAt (i) ;
    m_CharCodeSet. m_strFilter. Format ("Alpha = %d or Lower = %d or Mixed = %d or Punctuation =
    %d"), (int) c, (int) c, (int) c, (int) c) ;     // 生成查询字符型变量 c 对应的值的 SQL 语句
    m_CharCodeSet. Requery () ;                     // 查询并生成结果记录
    (略) …                                          // 处理并保存查询结果
}
```

当获得所有数据在文本模式下的值后,通过将这些值两两组合在一起就可以生成 PDF417 码词。组合公式为:

$$码词 = 30 \times H + L$$

式中 H、L 分别表示字符对中的高位和低位字符值。

下一个步骤是通过这些数据码词以及选择的错误纠正等级计算纠错码词。纠错码词的计算方法包括如下几个步骤。

第一步 :建立符号数据多项式,该多项式的表达式是:

$$d(x) = d_{n-1}x^{n-1} + d_{n-2}x^{n-2} + \cdots + d_1x + d_0$$

式中多项式的系数由数据码字区的码词组成。其中包括符号长度码词、数据码词和填充码词。每一数据码词(d_i,i = 0,…,n - 1)在条码中的排列位置如下：

起始符	L_0	d_{n-1}	d_{n-2}				R_0	终止符
	L_1						R_1	
	L_{m-2}			C_{k-2}		d_0	C_{k-1} R_{m-2}	
	L_{m-1}			C_0			C_1 R_{m-1}	

第二步 建立纠错码词的生成多项式,具有 k 个错误纠正码词的生成多项式为

$$g(x) = (x - 3)(x - 3^2)\cdots(x - 3^k) = x^k + g_{k-1}x^{k-1} + \cdots + g_1x + g_0$$

计算生成多项式的系数 gi 的程序如下：

```
g[0] = -3;                    // g[i]为生成多项式的系数gi,k为错误纠正码词的个数
g[1] =1;
    for (int =2;i<=k;i++) {
      for (int j = i - 1;j>=1;j--) {
        g[j] = -g[j] * (int) pow (3,i) +g[j - 1];
      }
        g[0] = -g[0] * (int) pow (3,i);
        g[i] =1;
```

第三步 生成错误纠正码词,实现的程序比较长,请参考文献[1]中求纠错码词的伪程序。得到纠错码词之后,根据码词数量选择条码的行数和列数,注意尽量使填充码词数量少,以使条码的效率最高。然后可以根据数据码词、左/右行指示符、纠错码词以及各行的族号查询各个码词对应的符号字符,并根据这些符号字符绘制条形码,这一过程可以仿照一维条码的绘制方法。

三、条形码的计算机译码

条码的传统译码方式通常是通过专用的条码识读设备进行的,有时候识读设备由于环境以及操作上的失误会导致译码错误,并且识读设备并不是通用的,某一种识读设备只能识读一种或几种条形码,所以存在一定的兼容性问题。如果能够用计算机实现条形码的软件译码,将会节约硬件设备的开销,并且可以根据需要增加可译码的条码种类。

1. 一维条码的计算机译码

一维条码的识读设备种类比较多,常用的是光笔式识读设备,即用光笔划过条码区域,根据条形码上条、空反射率的不同产生信号,然后对信号进行处理产生可读信息。在这里,我们先用扫描仪或者摄像头将条形码保存为位图,然后用软件对位图进行处理达到译码的目的。软件的编程环境是 Microsoft VC + +,在 MFC 的 CDC 类中有一个 GetPixel () 函数。它的原型如下：

```
COLORREF GetPixel (POINT point) const;
```

该函数的参数是屏幕上某一点的逻辑坐标,返回值是该点的颜色信息。颜色信息是用 RGB 值

表示的。这样如果从条码的左边开始取点,依次判断各个点的颜色是白色还是黑色,这一过程跟识读设备扫描过程比较类似。在扫描过程中,如果遇到点从白色变为黑色,则表示检测到条 如果点的颜色从黑色变为白色,则表示检测到空。下面是实现这一过程的程序:

```
CClientDC dc (this);                                    // 获得客户区设备
OnPrepareDC (&dc);                                      // 设置坐标影射信息
BOOL bSearchingBar = TRUE;                              // 标志变量,如果为真则表示正在搜索条
int y = 100;                                            // 扫描线的纵坐标,可以根据条形码的高度设定
CArray < int, int > intArray;                           // CArray 类,用于保存找到的颜色变化点的横坐标
for (int x = 10;x < = 800;x + +) {                      // 扫描线横坐标从 10 到 800
    if (bSearchingBar) {                                // 扫描颜色不是白色的点
      if (dc. GetPixel (x,y) ! = RGB (255,255,255))
{                                                       // 判断是否是误扫描,如果扫描两点之间间隔小于
                                                        // 单位模块宽的一半,则为
                                                        //误扫描,应该排除扫描到的条或空
    if ( (intArray. GetSize () >0) &&
      ((x - intArray. GetAt (intArray. GetUpperBound
    ())) < = m_int Width/2)) {
      intArray. RemoveAt (intArray. GetUpperBound ());
      bSearchingBar = false;
      continue;
        }
        intArray. Add (x);
        bSearchingBar = false;
      }
    }
else {
    if (dc. GetPixel (x,y) = = RGB (255,255,255)) // 颜色为白色则表示扫描到空
          {
          if ( (intArray. GetSize () >0) &&
              ((x - intArray. GetAt (intArray.
          Get Upper Bound ())) < = m_int Width/2))
          {
          intArray. RemoveAt (int Array. GetUpper Bound ());
          bSearchingBar = false;
          continue;
          }
          intArray. Add (x);
          bSearchingBar = true;
      }
    }
}
```

这样经过扫描后得到的 intArray 变量保存了各个颜色变化点的横坐标,然后再将相邻两

点的横坐标相减就得到各个条、空的宽度。最后根据条、空组合和可读字符的一一对应关系查询得到可读字符，注意在查询前应该排除表示附加信息的条与空（起始符、中间间隔符和结束符）。

2. 二维条码的计算机译码

二维条码的计算机译码与采用专用译码器译码相比有一个优点，采用译码器译码则必须保证扫描线的倾斜角度不能太大（见图 1.10－3），否则不能译码。相反，采用计算机软件译码的话则不会存在这类问题。

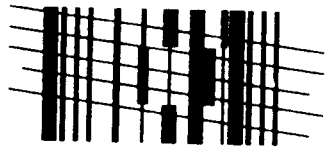


图 1.10－3　采用译码器译码，则必须保证扫描线倾角不能太大

二维条码在扫描原理上和一维条码是一样的，不过由于二维条码由多层组成，每一层都可以看做一个或几个独立的一维条码，所以在扫描的时候应该根据所含层数扫描多次。然后将各个层的扫描结果组合起来，排除掉附加信息后再进行查询，实现译码。由于篇幅有限，这里就不再给出完整的实现程序了。

通过上面的介绍可以看出，利用计算机进行条形码（尤其是二维条形码）的编码和译码，具有速度快、精度高以及节约成本等优点。实践证明，利用计算机译码在实际中具有很大的实用价值，应该得到大力的推广。

参 考 文 献

1　中华人民共和国国家标准 GB/T 17172 - 1997 [S]. 北京：中国标准出版社，1998

2　DAVID J. KRUGLINSKI，SCOT WINGO，Programming VisualC + + [M]. USA：Microsoft Press，1995

选自《计算机测量与控制》月刊，2002 年第 4 期

1.11 蓝牙技术综述

杭州浙江工业大学机电学院 (310014)
胡新华 杨继隆 姜 伟 殷进军

一、引 言

Bluetooth 无线技术是一个低功耗、短距离、双向无线通信的全球规范。它的设立是为实现“全球性互联”的概念。装备有 Bluetooth 无线技术功能的设备之间可以实现无缝互操作。一个微型的 Bluetooth 模块可以嵌入到任何电子设备中,使得产品可以直接通信或通过完善的全球网络通信。Bluetooth 无线技术使得数据、图像、声音、视频和远程遥控指令可以无线地传输到世界任何地方。Bluetooth 把用户从线缆的束缚中解放出来,超越了需要依赖其他技术实现无线通信的传统观念。Bluetooth 技术的应用实现了音频产品、个人通信设备和计算机的跨越式应用。

蓝牙的英文名称是 Bluetooth,是 Ericsson 移动通信公司在 1994 年开始启动的,当时想解决移动电话与它的外围配件,如免提耳机、台式设备间的无线连接问题,后来逐渐发展成一种短距离无线链路的概念。1998 年 5 月,爱立信、因特尔、东芝、诺基亚和 IBM 等 5 家公司组成的“蓝牙专门兴趣小组”(Bluetooth Special Interest Group—SIG) 把蓝牙无线技术的理念正式推向社会,使其成为无线技术的全球规范。Bluetooth 取自中世纪欧洲丹麦的一个开国皇帝 Harald Bluetooth 的名字,他为统一四分五裂的瑞典、丹麦、芬兰立下了汗马功劳。瑞典爱立信公司为这种即将成为全球通用的无线技术命此名,也许大有一统天下的含义。

Bluetooth 工作在 2.4 GHz 附近的 ISM (Industrial, Science and Medicine) 频段,此频率附近的频段在世界范围内都是公开、免费的,无需特许就可使用。不过,不同的国家和地区对实际可供使用的频段都有各自的规定。一般而言,在北美和欧洲有 79 个频段可供使用,而日本、法国和西班牙只有 23 个频段可供使用,每个频段间隔均为 1 MHz。当功率为 0 dBm 时,通信距离可达 10 m;当功率提升至 20 dBm,通信距离则可高达 100 m。

Bluetooth 采用时分多址 (TDMA) 的技术。数据经分组打包,才经由长度达 625 μs 时隙发送。Bluetooth 支持两类型链路,即同步面向连接 (Synchronous Connection—Oriented, SCO) 和非同步非连接 (Asynchronous Connection—Less, ACL)。SCO 包在预定时隙传送,主要用来传送声音;与此对应,ACL 则可在任意时隙传送,主要功用是传送分组的数据。每个声音通路都支持 64 kb/s 的同步链路,而非同步数据通路则可支持 721 kb/s 的不对称链路或是 432.6 kb/s 的对称链路。

为了使其能在嘈杂的无线环境中工作,使链路安全可靠,蓝牙无线技术采用快速跳频扩频技术 (Frequency Hopping Spread Spectrum),单时隙分组的跳频速率为每秒 1 600 次,而多时隙分组的跳频速率则略有降低;但在建立链接 (包括寻呼模式和查询模式) 时,跳频速率可提高至每秒 3200 次。由于使用了这样的高跳频速率,Bluetooth 系统具有足够的抗干扰能力。

Bluetooth 系统支持点对点和点对多点的无线连接。当一组设备通过 Bluetooth 链接起来后,就构成了微微网(Piconet);多个独立而非同步的微微网组成的群组则构成分散网(Scatternet)。任何一个设备都可以主动引发无线链接并成为主单元(Master);而在微微网链接期间,其余所有非主单元设备则成为从单元(Slaves)。主单元的时钟和跳频序列进而成为其他设备用以保持同步链接的依据。一般而言,一个主单元可支持多达 7 个同步链接,与 7 个从单元作数据交换,此时,这些附从被称为活跃从单元;其余的惰性从单元(Parked Slaves)虽仍与微微网同步,但并没有参与数据传输。事实上,微微网或分散网中的任何 Bluetooth 设备都可能在数据传输中作为主单元或从单元。

蓝牙技术是一种短距离无线连接技术,其目的是取代目前许多专用的电缆设备。主要优点有:可以随时随地用无线接口来代替有线电缆连接;具有很强的移植性,可应用于多种通信场合;功耗低;蓝牙集成电路应用简单。

二、蓝牙系统组成

蓝牙系统一般由无线部分、链路控制部分、链路管理支持部分和主终端接口组成。如图 1.11-1 所示。

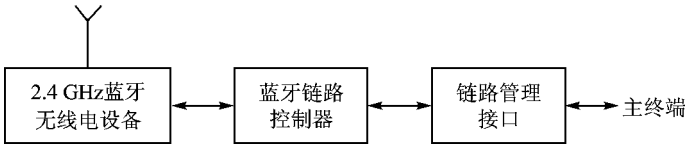


图 11.1-1 系统组成

三、蓝牙协议栈(Bluetooth Protocol Stack)

蓝牙技术是一个开放性系统(OSI),其主要目的就是使符合该规范的各种设备能互通,这就要求本地设备和远端设备使用相同的协议。当然,不同的应用,其使用的协议栈也可能不同。但是,它们都必须使用蓝牙技术协议规范中的物理层和数据链路层。完整的蓝牙协议如图 1.11-2 所示。

当然,不是任何应用都必须使用所有协议,可以只采用部分协议,例如语音通信时,就只需经过基带协议(Baseband)就行,而不必通过 L2CAP 层。除了上述协议层外,规范还定义了主控制器接口(HCI),为基带控制器、连接管理器、硬件状态和控制寄存器等提供命令接口。主控制器接口可以位于 L2CAP 的下层,也可以位于 L2CAP 的上层。这些协议又可分为蓝牙指定协议和非指定协议,这样区分主要在蓝牙指定协议的基础上尽可能地采用和借鉴现有的各种高层协议(也就是非指定协议),使得现有的各种应用能移植到蓝牙上来,充分利用兼容蓝牙技术规范的软硬件系统。蓝牙核心协议都是蓝牙指定协议,绝大多数蓝牙设备都需要这些协议。而 RECOMM 和 TCS—binary 协议是 SIG 分别在 ETSITS07.10 和 ITU—Recommendation Q.931 协议的基础上制定的。选用协议主要是各种已经广泛使用的高层协议。总之,电缆替代协议、电话控制协议和选用协议在核心协议的基础上构成了面向应用的协议。下面,就各个协议做一个简单的介绍。

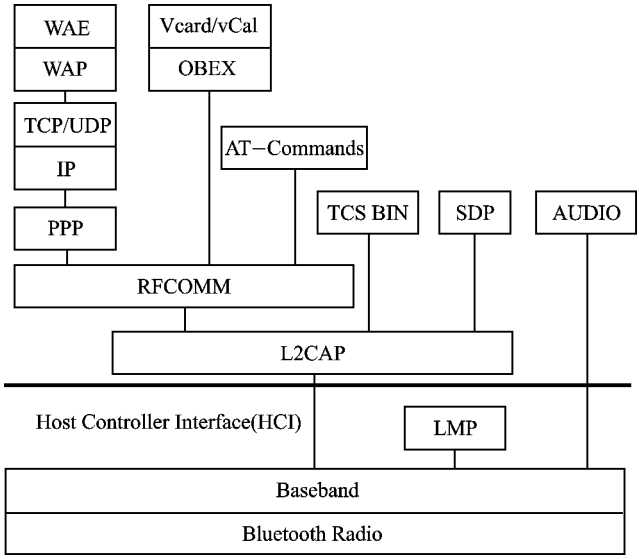


图 1.11－2 蓝牙协议栈

1. 蓝牙核心协议

(1) 基带协议

基带协议 (Baseband) 就是确保各个蓝牙设备之间的物理射频连接,以形成微微网。蓝牙的射频系统是一个跳频扩频系统,其任一分组在指定时隙、指定频率上发送,它使用查询 (Inquiry) 和寻呼 (Paging) 进程同步不同蓝牙设备间的发送频率和时钟。

它可为基带数据分组提供两种物理连接方式:同步面向连接 (Synchronous Connection－Oriented,SCO) 和异步非连接 (Asynchronous Connectionless, ACL)。SCO 既能传输语音分组 (采用 CVSD 编码),也能传输数据分组;而 ACL 只能传输数据分组。所有的语音和数据分组都附有不同级别的前向纠错或循环冗余校验编码,并可进行加密,以保证传输可靠。此外对于不同的数据类型都会分配一个特殊的信道,可以传递连接管理信息和控制信息等。

(2) 连接管理协议

连接管理协议 (Link Manager Protocol,LMP) 负责蓝牙设备间连接的建立和控制,包括控制和协商基带分组的大小。它还用于连接的发起、交换、核实,进行身份认证和加密等安全措施。另外,它还可以控制无线设备部分的电源模式和工作周期,以及微微网内各设备的连接状态。

(3) 逻辑链路控制和适配协议

逻辑链路控制和适配协议 (Logical Link Control and Adaptation Protocol, L2CAP) 是基带的上层协议,可以认为它是与 LMP 并行工作的,它们的区别在于当数据不经过 LMP 时,L2CAP 直接为上层服务。L2CAP 采用多路复用技术、分段、重组技术及组的概念。虽然基带协议提供了 SCO 和 ACL 两种连接类型,但是 L2CAP 只支持 ACL,并允许高层协议以 64 KB 的速度收发数据分组。

(4) 服务发现协议

服务发现协议 (Service Discovery Protocol,SDP) 是蓝牙技术框架中非常重要的一个部分,是所有用户模式的基础。使用 SDP,可以查询到设备信息和服务类型,之后,蓝牙设备之间的

连接才能建立。

2. 电缆替代协议(RFCOMM)

RFCOMM是基于ETSI 07.10规范的串行线仿真协议。它在蓝牙基带协议上仿真RS-232控制和数据信号,为使用串行线传送机制的上层协议(如OBEX)提供服务。

3. 电话控制协议(Telephony Control Protocols,TCS)

二元电话控制协议是面向比特的协议。它定义了蓝牙设备建立语音和数据呼叫的控制信令,定义了处理蓝牙TCS设备群的移动管理进程。在ITU-TV.250和ET300 916(GSM 07.07)的基础上,SIG定义了控制多用户模式下,移动电话、调制解调器和可用于传真机业务的AT命令集电话控制协议(AT-Commands)。

4. 选用协议(Adopted Protocols)

(1) 点对点协议(PPP)

PPP是IETF(Internet Engineering Task Force)制定的,在蓝牙技术中,它运行于RFCOMM之上,完成点对点的连接。

(2) UDP/TCP/IP

UDP/TCP/IP也是由IFTF制定的,是互联网通信的基本协议,在蓝牙设备中使用这些协议,是为了与互联网连接的设备进行通信。

(3) 对象交换协议(OBEX)

OBEX是IrOBEX的简写,是红外数据协会(IrDA)制定的会话层协议,采用简单和自发的方式来交换对象。它提供的基本功能类似于HTTP,在假定传输层可靠的基础上,采用客户机—服务器模式,而独立于传输机制和传输应用程序接口(API)。另外,OBEX专门提供了一个文件夹列表对象,用于浏览远端设备上的文件夹内容。电子名片交换格式(vCard)和电子日历及日程交换格式(vCal)都是因特网邮件协会(Internet Mail Consortium)开发的开放性规范。这些规范只是定义了数据传输格式,而没有定义传输机制。SIG采用这些已经定义好的规范,是为了进一步促进个人信息的交互。

(3) 无线应用协议(WAP)

WAP是无线应用协会论坛制订的,它融合了各种广域网络技术,其目的是将互联网内容和电话传送的业务传送到数字蜂窝电话或其他无线终端,选用WAP,可以充分利用无线应用环境(WAE)开发的高层应用软件。

四、蓝牙的硬件实现

目前多数厂家供应的蓝牙核心模块由三部分组成:基带处理器、无线收发模块和flash。配备少量的外围元件如音频编译码器(CODEC)、主机控制接口(HCI)、电源电路等即可构成蓝牙产品,例如由ERICSSON微电子公司推出的POK 101 007蓝牙模块。典型的蓝牙模块框图如图1.11-3所示。

利用蓝牙芯片供应商提供的蓝牙模块,可以很方便地实现一个应用。如图1.11-4所示为采用UART接口的蓝牙系统框架。图中利用UART作为数据通信接口,而PCM作为语音通信接口。

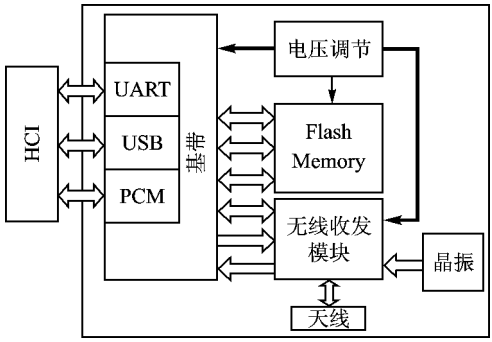
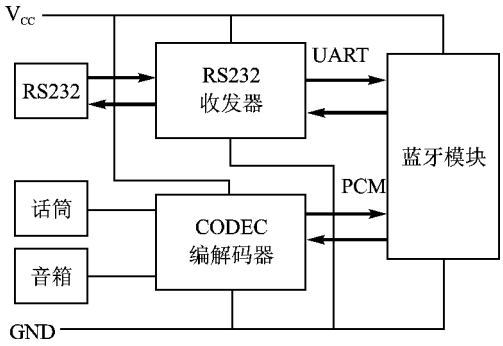


图 1.11－3 蓝牙模块框图



1.11－4 采用 UART 接口的蓝牙系统框架图

五、蓝牙技术的应用前景

Internet 和移动通信的迅速发展，使人们对各种数据源和网络服务的需求迅速增长。蓝牙作为一个全球开放性无线电标准，通过把各种语言和数据设备用无线链路连接起来，使人们能够随时随地实现个人区域内语音和数据的交换与传输，随着技术的发展和完善，蓝牙必将对人们的生活和工作产生重大影响。

最近的报告显示："1999 年，蓝牙技术产品的全球销售量几乎为零，2000 年猛增至 3 670 万美元，到 2001 年最少将达 1.26 亿美元。2002 年，全球使用蓝牙技术的外围设备将达 1.5 亿台，使用蓝牙技术的笔记本电脑将达 2 500 万部。2003 年，全球 90% 以上的笔记本电脑将使用蓝牙。2005 年，全球将推出 6.7 亿台使用蓝牙的信息家电。而届时全部蓝牙产品将达到 40 亿件。"在中国，蓝牙技术也正受到越来越多的关注。包括联想在内，中国的海尔、TCL、科龙等一批精英企业都加盟 SIG。家电业巨头海尔公司新近成立了研究中心，将蓝牙产品的开发放在重要位置。"E 海尔、E 家庭"就是基于蓝牙推出的新概念。

值得注意的是，目前市场上已有一种名为"802.11b"(或称 Wi－Fi) 的无线网络技术，国外一些大公司已经投入大量的人力、财力和物力进行研究、推广和应用，而且蓝牙的性能和它相比并不占绝对优势。市场上还有一种传输速度达 4 Mb/s 的红外 IR 无线传输技术，最近采用 16 Mb/s 的扩展方案也已经批准，它的传输速度不仅大大超过蓝牙，而且其传输距离也大大超过蓝牙，鹿死谁手，还难预料。更令人担忧的是，目前蓝牙芯片价格约在 15～20 美元，距蓝牙

市场的起飞价5美元还有一段距离。这个价格对大多数相对低廉的外设来说是一笔不菲的成本。蓝牙在目前还不是一种低成本的通信方式。最早支持蓝牙技术的Intel公司,现在也对蓝牙的前途产生了怀疑。他们已经决定在新型无线互联网设备中采用另外一种被称为“HomeRF”的技术,以留条后路。无论如何,蓝牙是美丽的,它具有独特的技术优势,而且受到电信行业尤其是电话公司的追捧,现在的关键就是需要使其更加成熟并迅速地推广。

参 考 文 献

1 http://www.bluetooth.com

2 何荣森,王宏宝,张跃. 蓝牙技术及其硬件设计. 电子技术,2001 (4)

3 曹冲. 蓝牙技术的发展现状和应用前景. 无线电工程,2001 (3)

4 Nathan J. Muller Bluetooth Demystified. 北京:人民邮电出版社

5 陈慧,潘志浩. 蓝牙技术要点及设计方案介绍. 无线电工程,2001 (9)

6 http://www.ericsson.com/microelectronics

7 Specifications of the Bluetooth System Version 1.1

8 金纯,许光辰,孙睿. 蓝牙技术. 北京:电子工业出版社

选自《现代电子技术》月刊,2002年第5期

1.12 蓝牙通信过程解析与研究

上海华东师范大学电子系(200062) 张 凌 姚 萌

蓝牙技术是基于 WPAN(Wireless Personal Area Network)的无线网络连接技术,是以短程无线电收、发技术为固定与移动设备通信环境建立一个短程无线电的特别连接。它建立一个通用的无线电空中接口(Radio-Air-Interface)及制定控制软件的公开标准,使无线通信技术和计算机技术紧密结合,使不同厂家生产的便携式设备在没有电线或电缆相互连接的情况下在近距离范围内具有互用、互操作的性能(Interoperability),代替固定与移动通信设备之间的电缆。利用 Ericsson 蓝牙开发包(Ericsson Bluetooth Development Kit),可以快速开发出建立在蓝牙通信技术之上的应用,加速产品开发的进度。

一、蓝牙系统模块分析

从软件和硬件来划分,蓝牙协议体系结构可分为底层硬件模块、中间协议层(软件模块)和高端应用层三大部分。链路管理层(LM)、基带层(BB)和射频层(RF)属于蓝牙的硬件模块。中间协议层包括逻辑链路控制和适配协议(L2CAP)、服务发现协议(SDP)、串口仿真协议(RFCOMM)和电话通信协议(TCS)。蓝牙协议栈的最上部是高端应用层,它对应于各种应用模型的 Profile,是 Profile 的一部分。

蓝牙协议体系结构框架如图 1.12-1 所示。

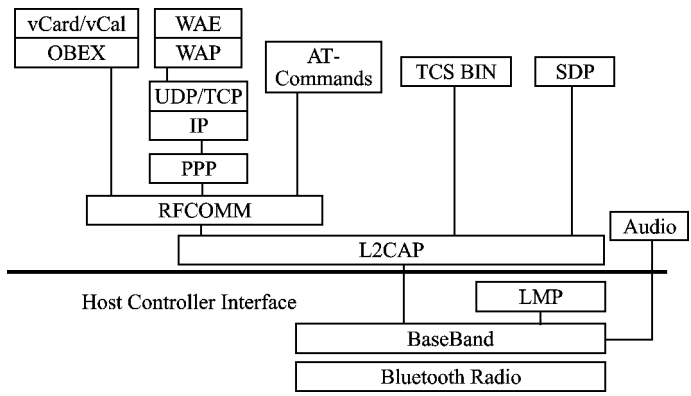


图 1.12-1 蓝牙协议体系结构框架

主域控制器接口 HCI(Host Controller Interface)是蓝牙协议中软硬件之间的接口。它提供了一个调用下层基带(BaseBand)、链路控制层(Link Manager)、状态和控制寄存器等统一的命令接口。HCI 协议以上的协议软件实体运行在主机上,而 HCI 以下的功能由蓝牙设备来完成,两者之间通过传输层进行交互。HCI 提供对基带控制器和链路管理器的命令接口,以及对硬件状态和控制注册成员的访问。该接口还提供对蓝牙基带的统一访问模式。

二、EBDK 硬件构架

Radio 模块是蓝牙硬件的射频模拟部分，包括射频发射器和射频接收器，以跳频技术实现频率扩展，进行 ISM 频段（即工业、科学、医学频段）频率信号的发送和接收。基带模块则对物理信道进行管理。链路控制模块进行通信链路的建立、鉴权。

EBDK 上的硬件分布构架如图 1.12－2 所示。

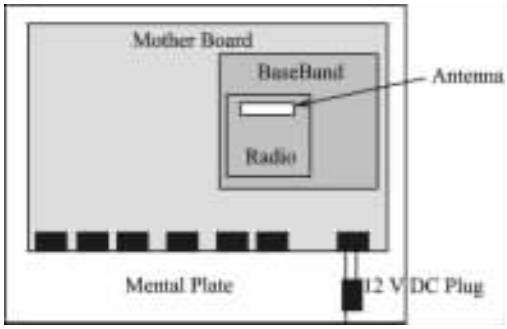


图 1.12－2 EBDK 硬件架构

三、软件功能解析

软件分为两部分：运行在宿主 PC 上的主机软件（Host Software）和在蓝牙基带设备上运行的 ROM 程序。EBDK 主机软件在 Win 98，NT PC 上运行，通过 RS232 或 USB 连接到 EBDK。其结构示意图如图 1.12－3 所示。

1. 主机软件功能机制解析

主机软件有两个主线程：一个执行主应用程序和传输数据包；另一个处理接收界面信息。如图 1.12－4 所示，左边的圆圈代表主应用程序线程，右边的圆圈代表接收器（Receiver）线程。接收器线程采用 Microsoft 定义的通信事件，一旦接收到一个通信事件（通常是接收缓冲区有一序列字符），就会产生一个 Windows 消息，同时将接收到的字符序列送到包组装器进行数据包的组装。主应用程序处理 Windows 消息队列，发现有输入字符的消息后，就进行数据包的组装；或者接收到用户界面的变化，对 Windows 的控制或输入信息转换成的数据流进行处理，进行数据包的整合。发送到包组装器中的数据必须重新组合，因为通常收到的数据不是一个完整的数据包。主程序判别出数据包的类型，然后进行图形用户界面的更新或发送数据包的相应处理。

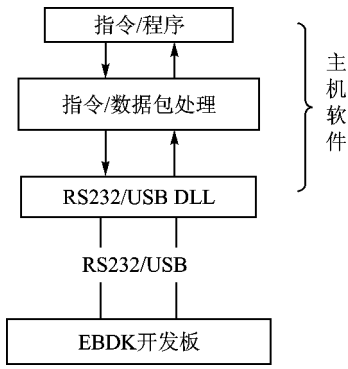


图 1.12－3 宿主 PC 软件与 EBDK 连接方式

2. 通信过程解析

以 EBDK 中的 Demo 程序为例，其中的通信过程流程如图 1.12－5 所示。

EBDK 主机软件结构为分层模块化形式，每一层都进行了封装，其他层只有通过接口才能访问。

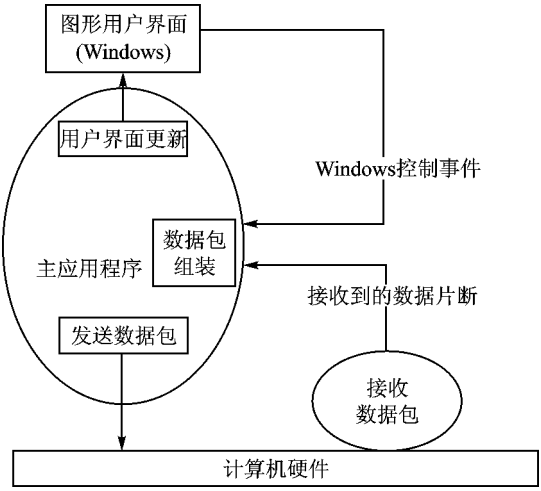


图 11.2－4 **主机软件结构**

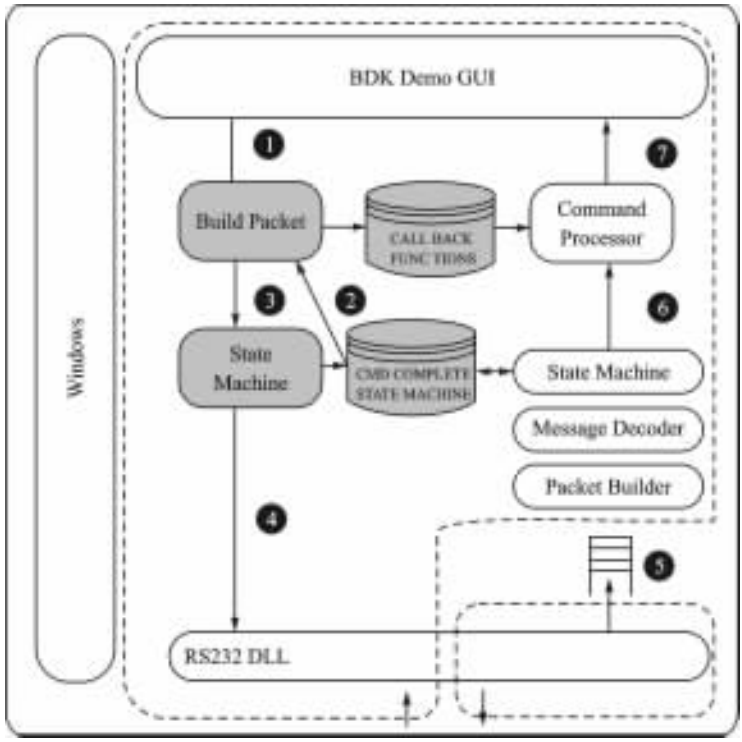


图 11.2－5 **通信流程**

如果用户应用程序发送了一序列数据，数据在到达 RS232 **之前，首先会进行包的组装，生成通信协议格式的数据包。在发送数据包之前还必须检测指令完成状态机（**Command Complete State Machine**）。指令完成状态机指示从发送前一条指令之后系统是否仍在等待指令完成（**Command Complete**）事件。如果状态是空闲（**IDLE**），则发送数据包 如果状态是等待指令完成（**WAITING_FOR_CMD_COMP**），用户程序则被通知稍候再试。**

在接收方向上,执行的操作与之相反。接收来的数据首先经过包组装器,组装成标准的数据包格式,然后解开包,抽取包中的数据,进行指令解码,获得数据中包含的指令。在指令被发送到指令处理器之前,仍然要检测指令完成(Command Complete)事件,以确定前一条指令是否已经被处理。之后,才被指令处理器处理,处理的结果反馈到应用程序进行界面的修改或通信状态的改变处理。

四、简单指令通信示例

以一个简单的示例来说明数据流动的全过程。

从 EBDK 的通信界面发送一条指令:

cmd cclk 0x2345

并按回车,将产生一个 Windows 事件调用字符串分析器对以空格分界的字符串进行标记和处理,生成一条消息字符序列。这个字符序列是对应指令的 HCI 位字符流。这个字符序列被封装到数据包中,经过通信过程(详见三.2)中的一系列过程发送给 RS232 DLL。RS232 DLL 再立即将它放到 COMM 口上。携带 HCI 指令的电信号从 PC 的 COM 或 USB 口出发,沿着 RS232 或 USB 线,到达 EBDK 的 COM 或 USB 口,穿过 RS232 缓冲区 Buffer(缓冲区)以及 Mother Board 进入基带连接器(BaseBand Connectors)和基带设备(BaseBand Device)。基带的 ROM 程序(Firmware)将消息解码并执行。在这种情况下,它生成一个数据帧,在空中通过电磁波传输。这个帧通过与 Radio Module 的接口,以串行序列的形式从射频天线发送出去。

射频信号被附近的一个 EBDK 获取,它的射频模块(Radio Module)将射频调制信号转换为一个数字信号,再以串行形式送到基带设备(BaseBand Device)。基带控制和 ROM 程序(Firmware)将数据包解码确认是否是发送给自己的信号,然后数据包被转换为一个 HCI 数据包(Data Packet)或事件包(Event Packet),再由基带设备将该包通过 RS232(或 USB)线送到 PC。PC 的 CPU 收到一个中断指示收到一些字节数据,这些数据被送到缓冲器中,然后被 RS232 包组装器(Packet Builder)进行处理。应用程序收到一个消息指明有一个数据包要处理,最后根据抽取出来的指令,调用相应的函数进行操作。这样就完成了一个消息通信的过程。

五、通信的编程实现

要以程序编程实现上述的过程,就需要在程序中设置上述操作中的软件接口动作机制。在 C++中以一个通信类可以实现上述的基本操作动作。一般需要的接口和解释如下介绍,即一个基本的蓝牙通信对象应该具有以下的基本操作接口。

发送消息

IssueCommand:发送指令

TransmitRawData:发送数据

接收消息

CommandCompleteSuccess:指令成功完成

CommandPending:指令延迟

GotRxPacket:收到数据包

ReadyForData:准备接收数据

DisconnectionOccurred :发生连接中断

GotBTAddress :获得蓝牙通信地址

而在实际通信操作中,要分为通信客户端和通信服务器端。因此,在基本的通信对象的基础上要派生出 Client 和 Server 对象,分别就具体的操作进行处理。

另外,需要一个包组装器专门处理数据包 CPacketBuilder :

On_WMC_RXD_DATA :处理接收到数据之后的动作

PacketComplete :包接收完毕的动作

InvalidPacket :判断包是否有效

在 VC + + 中,利用蓝牙工程模板可以自动生成,因此蓝牙通信包软件的开发具有易于实现和易与其他软件捆绑的特点。

本文解析了基于 EBDK 的蓝牙通信的整个过程以及软件实现机制,可以作为蓝牙通信应用开发的借鉴。具备了 Ericsson 蓝牙开发工具包之后,由于蓝牙软件包具有捆绑性与易实现性,可以在此基础上直接进行应用的开发,也可以从蓝牙通信的软件实现机制上理出应用的实现思路。

参 考 文 献

1 Palowireless. Com, Bluetooth Resource Center. Bluetooth Tutorial-Specifications [EB/OL]. http ://www. palowireless. com/infotooth/tutorial. asp,2001 - 11 - 10

2 金纯,许光辰,孙睿. 蓝牙技术 [M]. 北京 :电子工业出版社,2001

3 Ericsson Microelectronics AB. BSK Technical Documentation [DB/CD]. Ericsson Ltd,2000

选自《计算机应用研究》月刊,2002 年第 9 期

1.13 蓝牙模块基带电路的接口技术

哈尔滨工业大学电子与通信工程系 关宇东 王文星

一、引 言

蓝牙技术是一种新兴的短程无线通信技术。它使电子设备能够方便地互连起来。目前，蓝牙 SIG (Special Interest Group) 制定了蓝牙标准，其目的是实现 1 Mb/s (有效传输速率是 721 bit/s)、有效传输距离为 10 m，当发射功率加大时，可实现 100 m 左右的无线通信。蓝牙技术成为人类社会通往自由通信的有力手段，已成为 21 世纪新兴通信的热点，从国内一些刊物上发表的文章可以看到许多关于蓝牙协议的文章。本文讨论的是在蓝牙技术开发中难度较大的硬件电路，并分析了蓝牙硬件的工作原理。

二、蓝牙模块的组成

本文所介绍的蓝牙模块由蓝牙基带设备 ROP 101 1112/C、射频模块 PBA313 01/2 所组成。这两个模块之间形成了蓝牙模块的内部接口。基带设备负责跳频和蓝牙数据及信息帧的传输，包括对纠错编码的支持，对 SCO 和 ACL 包的组织、流控等，射频模块负责收发数据。外部模块主要有电源模块、晶体振荡模块、天线等。基带设备与外围模块的接口实现了蓝牙模块的主要功能，几个主要的接口有通用异步收发接口 UART、异步收发串行接口、PCM 语音接口、基带部分与 FLASH ROM 的接口，还有目前比较流行的 USB 接口。图 1.13－1 所示为典型的应用框图。

PBA313 01/2 是短程蓝牙通信系统的前端。它采用的封装形式是 BGA 封装，与基带的接口在图 1.13－1 中已经说明，主要由串行接口、数据接口、输入控制、输出控制组成。对于完整的操作，蓝牙射频必须连接到基带控制或能模拟基带功能的设备上。基带功能从射频控制器的寄存器被读出或写入，这些寄存器用于设定频率，实现调整和控制等功能。基带和射频模块之间的通信在串行接口 (Serial interface) 上实现，PBA313 01/2 也需要一个外部天线和 13 MHz 的晶体时钟输入，天线通过一个 50 Ω 的接口连接。电源分成两部分：一部分用于提供给压控振荡器 (VCO)，另一部分提供给蓝牙射频。射频模块主要由 7 个操作子块组成。图1.13－2说明了各个子块之间的互操作性，Radio ASIC 是一个利用外差接收的射频控制交换芯片，具有 3 MHz 的中频；另外，有大量的其他电路块集成在这个控制器里，如合成器、串行接口、晶振振荡器、可跳频的低功耗振荡器 (LPO)、电源复位电路 (POR)、分频电路以及可编程控制逻辑寄存器。VCO－tank 产生所需的射频频率，环路滤波器滤波所需的调整电压以改变 VCO－tank 的选择信息。接收平衡转换器用于转换接收的信号从不平衡状态改变为平衡状态。发送平衡转换器用于将发送信号有平衡状态转换为不平衡状态，以便于天线交换。天线开关既可以是从天线滤波器到接收端口，也可以是从射频控制部分输出端口到天线滤波器。天线开关的控制由 TX_ON 端口的设置完成的。天线滤波器去除接收模式时不想要的信号和减少发送模式时

的谐波信号。

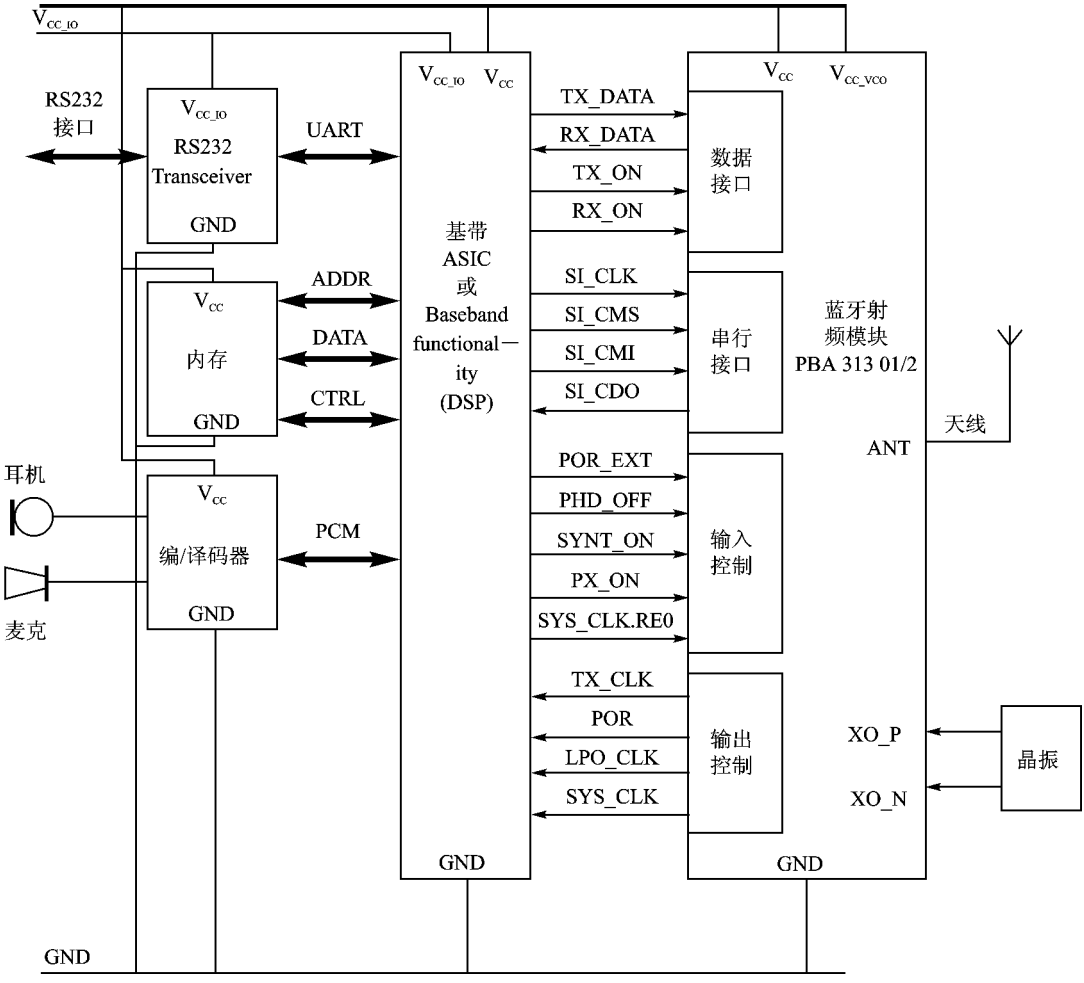


图 1. 13－1 **蓝牙模块框图**

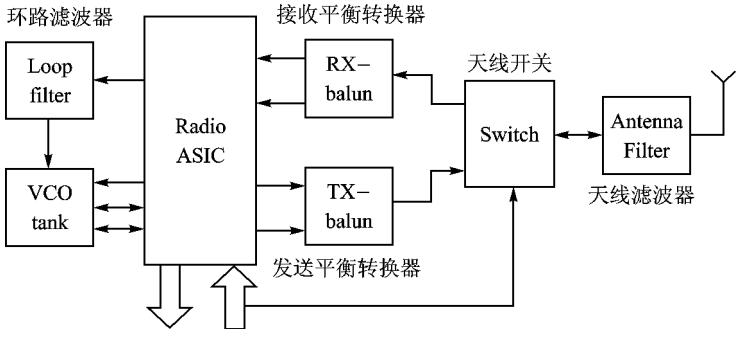


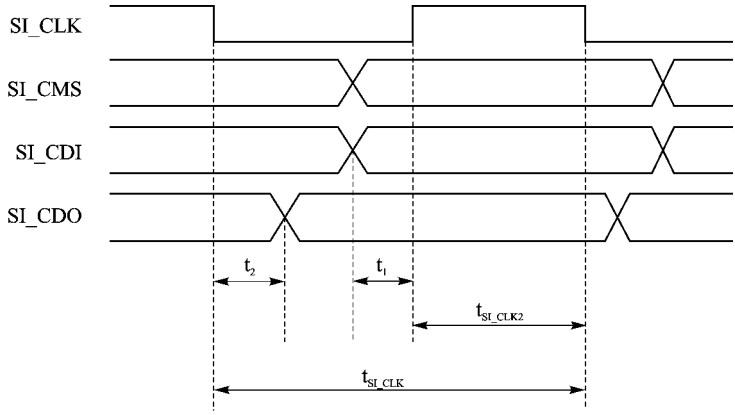
图 1. 13－2 **射频模块内部块图**

三、射频串行接口

射频串行接口(Serial Interface)是基于边界扫描体系结构(IEEE Std 1149.1)之上的一种协议。在串行接口和基带电路之间的互连是由4个一比特的信号组成的控制数据输入(SI_CDI)、控制模式选择(SI_CMS)、控制时钟(SI_CLK),以及控制数据输出(SI_CDO)。串行数据SI_CDI传输数据到PBA313 01/2,而SI_CDO则由射频传输数据到基带。SI_CMS和SI_CLK控制PBA313 01/2的串行接口。

1. 状态转换

从一种状态到另一种状态之间的过渡是由SI_CLK(4 MHz)上升沿时SI_CMS的输入值来决定的,SI_CMS及SI_CDI的值将在SI_CLK的下降沿变化,输出SI_CDO也将在SI_CLK的下降沿变化。一个指令寄存器(IR,Instruction Register)的扫描周期开始于状态信息下传时,即捕获IR。时序逻辑如图1.13-3所示。串行接口在13 MHz的系统时钟SYS_CLK及POR_EXT为高电平时操作有效。



	Min	Typ	Max	Unit
t_{SI_CLK}	200	250		ns
t_{SI_CLK2}	76	76		ns
t_1	20			ns
t_2			20	ns

图1.13-3 射频串行接口时序图

指令寄存器扫描是一般的读/写指令,这个指令选择一个特定的寄存器来读出或写入。它的位码定义为01YYYY,YYYY是要读写的寄存器地址。还有两种扫描指令:数据寄存器(Data Register)扫描,作用是把8位数据移入到寄存器中;旁路寄存器(BY Pass Register)扫描,在一个SI_CLK延迟下用于把SI_CDI值移入到SI_CDO中。表1.13-1列出了主要寄存器及说明、读写状态和地址。例如,在信道寄存器中写77时,选择的是第77条信道,VCO-tank的值将是2.477 GHz。一般的操作时序是:执行指令寄存器扫描,指出信道寄存器,寄存器地址的值是通过SI_CDI输入的,顺序输入010010(Chan地址);执行数据寄存器扫描,通过SI_CDI写入新的信道值01001101(77)。

表 1.13－1 寄存器及说明、读写状态和地址

Register	#bits	R or W	IR value	Comments
Chan	8	R&W	010010 = 18	Bit7 = 0 = > bit6 - bit0 :传输信道 Bit7 = 1 = > bti6 - bit0 :接收信道
LP0－hi	1	W	010100 = 20	微调低功耗振荡器,LPO - hi = MSB LPO - lo = LSB
LP0－lo	8	W	010101 - 21	
Ctrl	7	W	010110 = 22	初始逻辑控制 Bit6 - 3 = LPO - trim Bit2 = 基带是否准备好 Bit1 - 0 没有被用
Power Control	5	W	011000 = 24	外部电源控制 Bit7 - 5 :用于使能外部阻抗 Bit4 - 0 没有被用
Enable	2	W	011001 = 25	Bit1 = 1 = > 使能 DAFC (dynamic automatic frequency) 算法 Bit0 = 1 = > 翻转 TX_DATA 位

2. 启动序列

启动序列是从电源复位 (Power－On－Reset,POR_EXT) 开始的,将使射频控制器中的所有寄存器复位。LPO_CLK 开始振动,同时,蓝牙射频等待基带电路去控制其寄存器 (设置 Ctrl 中的第二位),现在由基带电路控制蓝牙射频,LPO 寄存器将被写入 LPO 频率,它的值射为 3.2 kHz。蓝牙射频和基带目前已经被初始化为通常所说的准确模式阶段,然后配置使能寄存器使 PX_ON、DAFC、TX_DATA 控制有效,因此,10111111 (191) 将被写入 Enable 寄存器。

3. 带电路的硬件接口

目前,大多数厂家所提供的蓝牙方案在硬件上都分为三部分 :无线收发单元、链路控制单元以及链路管理单元和 I/O 单元。只要和内含蓝牙软件协议栈的 Flash ROM 配合,即可向数据和语音设备提供全兼容的蓝牙接口。具体硬件电路的设计如图 1.13－4 所示,蓝牙基带芯片是一个由 96 个引脚组成的 8mm×8mm 的集成电路,采用的是 FPBGA 封装。由物理层 DSP 硬件引擎、猝发状态控制器、微处理器、内存管理单元等部分组成。该硬件引擎用专门的硬件电路来实现,能够在高速运算的同时维持低功耗。它所实现的功能包括 :前向纠错、帧头错误控制、加密、数据白化以及访问码比较等。此外,还能兼容 A 律、μ 律、线性和 CVSD (连续可变斜律增量调制) 四种语音格式的数据。它的比较典型的应用是在移动电话、个人数字助理、调制解调器、以及局域网上。Flash ROM 采用的是 Intel 公司的 28F800B3T120,与蓝牙基带的接口主要由地址总线 (EXT_DATA [15..0]) 、数据总线 (EXT_ADR [19..1]) 、读写以及片选线构成。蓝牙基带负责链路层的管理,因此它主要装载了链路层管理程序。

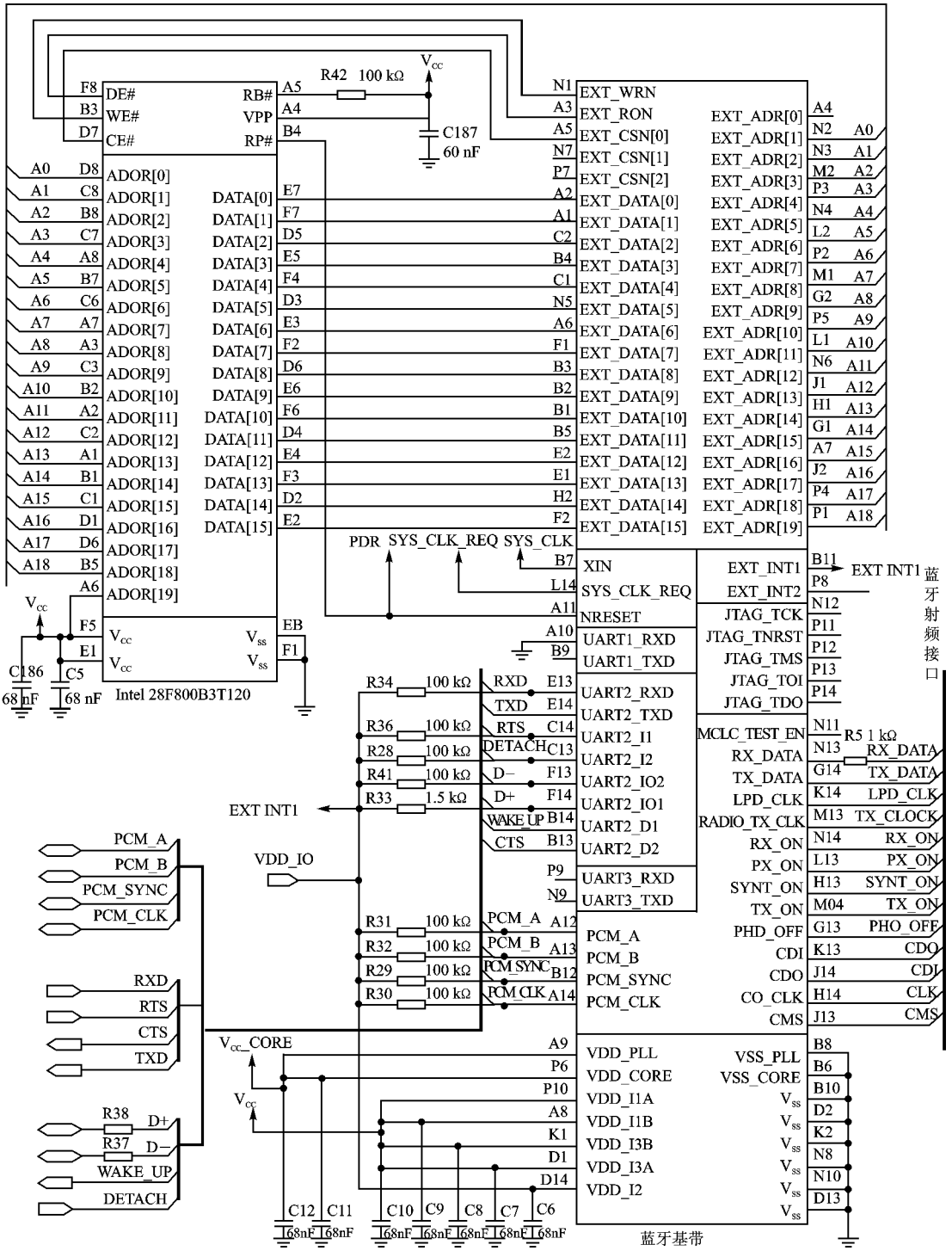


图 1. 13－4　基带与外围接口电路图

四、硬件电路时序

1. 系统启动时序

启动过程时序如图 1.13－5 所示，EXT_RESET 由主机产生连接到蓝牙芯片组，它至少在 4 个时钟周期之后。然后，射频模块产生 1 个内部复位，4 个时钟过后，POR 信号变高，同时，射频模块等待基带来通过设置控制寄存器中的状态位来对其控制。寄存器通过链路管理软件初始化，LPO 频率通过调整相应的寄存器利用连续相近算法被调整至 3.2 kHz，这时的状态被称为休眠状态。

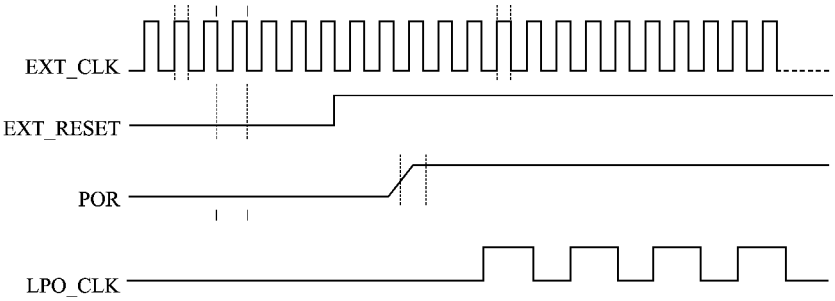


图 1.13－5 系统启动时序图

2. 基带总线的时序

基带总线的信号如下：Sysclk，csn，wrn，a［n: 2］，dwr［n: 0］，drd［n: 0］，int。其中，Sysclk 为系统时钟，所有的传输都发生在一个系统周期里。csn 是一个片选信号，每个外围块都是从中央地址译码接收 csn，它在上升沿有效，保持到系统时钟的下一个上升沿。wrn 是系统写选通，为低时，表示出现写操作，从 dwr 总线来的数据将被写入到地址寄存器；为高时，表示出现读操作，数据由地址寄存器写入到 drd 总线上。a［n: 2］是地址总线，地址在系统时钟变为上升沿后有效，保持到下一个系统时钟上升沿。dwr［n: 0］是数据写总线，在一个写周期里，它被送入数据；同样，drd［n: 0］是数据读总线，数据从这个总线上读出，系统写总线的写周期时序如图 1.13－6 所示。Int 是外围模块的中断输出。基带与主机相连的 UART、PCM 接口参照图 1.13－4 所示。

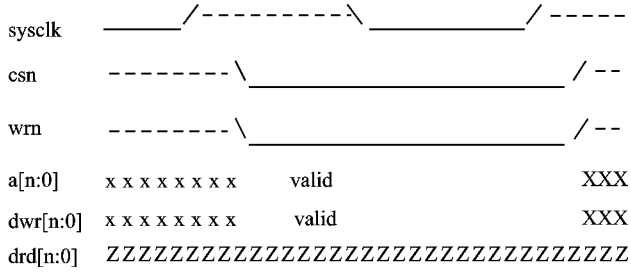


图 1.13－6 系统总线写周期

五、结束语

以上介绍了蓝牙模块的硬件组成,详细讨论了它的内部射频接口,叙述了基带模块与射频模块之间的数据通过射频串行口的传输过程,以及内部状态寄存器的工作原理。最后,介绍了基带的启动过程,与外围主机及接口的连接方式、具体实现,以及总线的读写逻辑时序。蓝牙仍然是一项发展中的技术,其应用目前正处于起步阶段,还有大量应用技术细节需要解决。总之,文中所讨论的蓝牙硬件接口应用希望能给蓝牙开发人员起到一定的帮助。

参 考 文 献

1 Ericsson, Nokia, IBM, etc. Specification of the Bluetooth System. Version 1.0B. December lst, 1999

2 Paul Buegess. The Bluetooth Radio Specification. Bluetooth Deverloper Conference Building a World Without Wires. December 8,1999 (http ∶//www.bluetooth.com)

3 http ∶//bluetooth.com/developer/core_10_b.pdf, Specification of the Bluetooth System—Core, Version 1.0b

选自《无线电工程》月刊,2002 年第 9 期

1.14 蓝牙 HCI 层数据通信的实现

重庆邮电学院自动化研究所(400065) 王 浩 王 平

蓝牙是无线数据和语音传输的开放式标准。蓝牙采用跳频技术,工作在 2.4 GHz 的 ISM 频段,在采用不对称的信道时单向最高传输速率可达 723.2 kb/s,可以同时支持数据与语音通信。蓝牙支持点到点和点到多点的连接,可采用无线方式将若干蓝牙设备连成一个微微网(Piconet),多个微微网又可互连成分散网(Scatternet),形成灵活的多重微微网的拓扑结构,从而实现各类设备之间的快速通信。

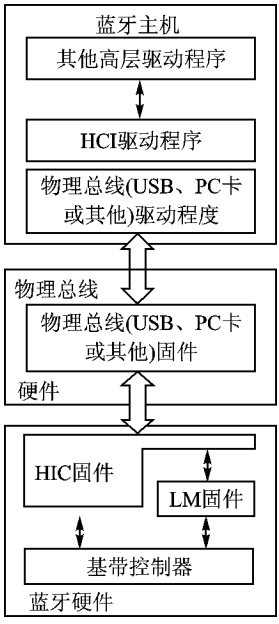


图 1.14-1 较低软件层概述图

一、HCI 概述

主机控制器接口(HCI)提供了一种访问蓝牙硬件能力的通用接口,图 1.14-1 是蓝牙软件层次的简单结构。HCI 固件通过访问基带命令、链路管理器命令、硬件状态寄存器、控制寄存器以及事件寄存器实现对蓝牙硬件的 HCI 命令。该接口还提供对蓝牙基带的统一访问模式。

在主机系统的 HCI 驱动程序和蓝牙的硬件 HCI 固件之间可能存在几个层次。这些中间层次,又称为主机控制器传输层,提供传输数据的能力。该层的目标是透明化,主机控制器驱动程序不关心它是在 USB 上还是 PC 卡上,USB 和 PC 卡对主机控制器驱动程序发送到主机控制器的数据不能进行处理。这样,主机控制器接口和主机控制器可以进行升级,升级不会对传输层有任何影响。

二、HCI 命令和事件

HCI 提供访问蓝牙硬件的统一指令方式。HCI 链路指令使主机能够控制到其他蓝牙设备的链路层链接。这些指令通过链路控制器(LM)与远程蓝牙设备交换 LMP 指令。具体细节请参见"链路管理器协议"。

HCI 策略指令用于本地或远程 LM。这些策略指令向主机提供影响 LM 管理微微网的方法。

主控制器和基带、信息和状态指令可为主机提供访问主控制器中不同注册表的能力。

执行 HCI 指令将耗费不同时间。因此,指令结果将以事件的形式返回给主机。例如,对于大多数 HCI 指令,主控制器在该指令完成时将生成命令完成事件。该事件分组包括已完成 HCI 指令的返回参数。为了在 HCI 传输层上侦测出错信息,必须定义在主控制器收到命令和发出应答之间的应答时间。由于最大应答时间取决于所采用的 HCI 传输层,因此推荐采用 1 秒的缺省值。这个事件值也取决于在指令队列中未处理指令的数量。

主控制器传输层提供 HCI 信息的透明传输。该传输机制为主机提供向主控制器发送 HCI 指令、HCI 数据和 SCO 数据,以及从主控制器接收 HCI 事件、ACL 数据和 SCO 数据的能力。由于主控制器传输层提供 HCI 信息的透明传输,HCI 规范对于主机和主控制器间交换的指令、事件、数据的格式进行了定义。

三、HCI RS232 传输层

HCI RS232 传输层的目的在于在蓝牙主机和蓝牙主机控制器之间的物理 RS232 接口上使用蓝牙 HCI,如图 1.14-2 所示。

图 1.14-2 HCI RS232 传输层

通过 RS232 传输层可发出四种 HCI 分组,包括:HCI 指令分组、HCI 事件分组、HCI ACL 数据分组和 HCI SCO 数据分组(参见"主控制器接口功能规范")。HCI 指令分组仅能用于发送到蓝牙主控制器,HCI 事件分组仅能有蓝牙主控制器发送,HCI ACL/SCO 数据分组则可由蓝牙主控制器自由发送和接收。

但是,主控制器接口不能区分四种 HCI 类型。因此,如果通过一通用物理接口发出一个 HCI 分组,则 HCI 分组指示器必须根据表 1.14-1 执行操作。

表 1.14-1 HCI RS232 分组头

HCI 分组类型	HCI 分组指示器
HCI 指令分组	0x01
HCI ACL 数据分组	0x02
HCI SCO 数据分组	0x03
HCI 事件分组	0x04
错误消息分组	0x05
协商分组	0x06

除上述四种 HCI 分组类型外,还有两种分组类型用于支持动态协商和错误报告。接收端使用错误消息分组(0x05)将错误报告给发送端。协商分组(0x06)则用于协商通信设置和协议。

四、蓝牙 HCI 层数据通信的硬件实现

1. 本系统硬件采用爱立信 ROK 101 008 蓝牙模块

特性:

- 遵从蓝牙 1.1 规范
- 射频输出为 class2 级
- 蓝牙 1.1 认证
- FCC&ETSTI 认证
- 支持

——USB 接口

——UART 接口

——PCM 语音接口

· I^2C 接口

· 点对多点操作

2. 硬件实现的原理图

硬件实现原理图如图 1.14－3 所示。

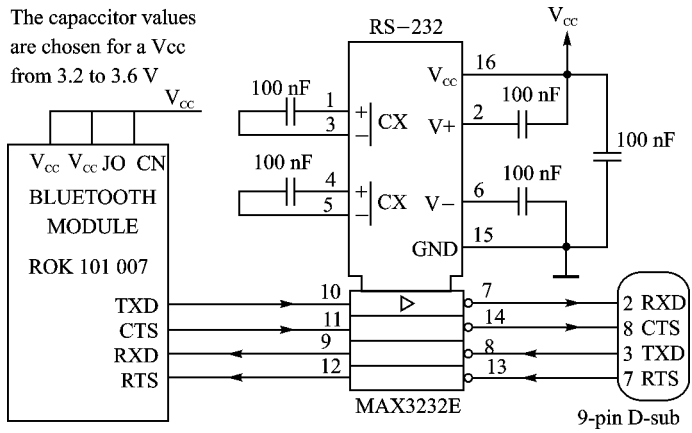


图 1.14－3 硬件实现的原理图

五、蓝牙 HCI 层数据通信的软件部分的实现

操作系统采用 windows 98/2000，开发平台采用 Microsoft Visual C++ 6.0；HCI RS232 传输层采用 MSComm 控件来实现软件部分实现的关键技术。

1. 打开串口和设置串口参数

在 CWorkDlg::OnInitDialog() 函数中打开串口，添加代码如下：

```
BOOL CWorkDlg::OnInitDialog()
{
if (m_mscomm.GetPortOpen())
   m_mscomm.SetPortOpen(FALSE);
   m_mscomm.SetCommPort(1);                    // 选择 com1
   if(!m_mscomm.GetPortOpen())
   m_mscomm.SetPortOpen(TRUE);                 // 打开串口
else
   AfxMessageBox("cannot open serial port");
   m_mscomm.SetSettings("57600,n,8,1");        // 波特率为 57600 (这是该蓝牙模块默认的设置),
                                               // 无校验,8 个数据位,一个停止位
   m_mscomm.SetInputMode(1);
   m_mscomm.SetRThreshold(1);
                                               // 参数 1 表示每当串口接收缓冲区中有多于或等于
```

```
                                        // 1 个字符时将引发一个接收数据的 OnComm 事件
  m_mscomm.SetInputLen (0) ;            // 设置当前接收区数据长度为 0
  m_mscomm.GetInput () ;                // 先预读缓冲区以清除残留数据
}
```

2. **蓝牙模块的初始化**

采用定时器发送蓝牙 HCI **指令,在** CWorkDlg::OnButtonInitialize () **函数中实现,代码如下:**

```
void CWorkDlg::OnButtonInitialize ()
{
  // TODO:Add your control notification handler code here
  SetTimer (1,500,NULL) ;               // 设置定时器,时间为 500 ms
}
void CWorkDlg::OnTimer (UINT nIDEvent)  // WM_TIMER 消息处理函数
{
// TODO:Add you message handler code here and/or call default
// The settings for getting into scan mode could be according to the below suggested script list
Reset
Read Buffer Size
Set Event Filter
Write Scan Enable:(Scan Enable 0x03)
Write Voice Setting:(Voice Channel Setting 0x0060)
Write Authentication Enable:(Authentication Enable 0x00)
Set Event Filter:(Connection Setup Filter:Connections from all devices, Auto Accept 0x02)
Write Connection Accept Timeout:(Connection Accept Timeout 0x2000)
Write Page Timeout:(Page Timeout 0x3000)
Read BDADDR
m_mscomm.SetOutput (COle Variant (m_strTXData)) ;// 发送数据
CDialog::On Timer (nIDEvent) ;
}
```

3. **串口事件消息处理函数**

CWorkDlg::OnOnCommMscomm1 () **的实现,代码如下:**

```
void CWorkDlg::OnOnCommMscomm1 ()
{
// TODO:Add your control notification handler code here VARIANT variant_inp;
  COleSafeArray safearray_inp;
  LONG len,k;
  BYTE rxdata [2048];                   // 设置 BYTE 数组 An 8 - bit
integerthat is not signed.
CString strtemp,str, stradd;

if (m_mscomm.GetCommEvent () = =2)      // 事件值为 2 表示接收缓冲区内有字符
{  // 以下你可以根据自己的通信协议加入处理代码
  safearray_inp.Clear () ;
```

```
variant_inp = m_mscomm. GetInput () ;          // 读缓冲区
safearray_inp = variant_inp ;                  // VARIANT 型变量转换为 ColeSafeArray 型变量
len = safearray-inp_inp. GetOneDimSize () ;    // 得到有效数据长度

for (k = 0 ;k < len ;k + +)
safearray-Get-Element (& ;rxdatatk) ;          // 转换为 BYTE 型数组
for (k = 0 ;k < len ;k + +)                    // 将数组转换为 CString 型变量
 {
BYTE bt = * (rxdata + k) ;
strtemp. Format ("%02X" ,bt) ;                 // 将字符送入临时变量
strtemp 存放
str + = strtemp ;
}
……
}
```

4. 转换为 UNICODE 类型

要能输出汉字,就需要把数据类型转化为 UNICODE 类型,在 CWorkDlg::ConvertUnicode () 函数中实现,其代码如下:

```
CString CWorkDlg::ConvertUnicode (CString string, int&position)
{
  ……
  在头文件中添加 #include" afxpriv. h"
  BYTE sumup
  CString strconv ;
  strconv + = (unsigned char) sumup ;
  USES_CONVERSION ;
  m_editaccept + = A2W (strconv) ;
  ……
}
```

该系统不但可以进行数据通信,还可以进行文件传输,HCI 指令测试等。在此基础上,还可以实现蓝牙的其他核心协议 L2CAP、RFCOMM 和 SDP,以及其他高层协议,如 WAP 等。由于蓝牙芯片价格低廉、应用简单,具有十分广阔的应用前景。

参 考 文 献

1 BluetoothTM. Specification Volume 2 Specification of the Bluetooth System Profiles. Version 1. 1 February 22, 2001

2 ROK101008 Bluetooth Module. Ericsson Microelectronics AB. September 2001

3 张禄林,雷春娟,朗晓虹. 蓝牙协议及其实现. 北京:人民邮电出版社,2001

4 金纯,许光辰,孙睿. 蓝牙技术. 北京:电子工业出版社,2001

5 David J. Kruglinski Inside Visual C + + . 北京:希望电子出版社,2000

1.15　蓝牙技术硬件实现模式分析

广州大学信息与机电工程学院(510405)　邹艳碧　张　为
广州大学理学院(510405)　吴智量

蓝牙技术是一项新兴的技术。它的主要目的是在全世界建立一个短距离的无线通信标准。它使用2.4～2.5 GHz的ISM(Industrion Scientifc Medical)频段来传送话音和数据。运用成熟、实用、先进的无线技术来代替电缆,它提供了低成本、低功耗的无线接口,使所有固定和移动设备通过微微网PAN(Personal Area Network)连接起来。诸如:计算机系统、家庭影院系统、无绳电话系统、通信设备等,相互通信,实现资源共享。蓝牙技术支持多种电子设备之间的短距离无线通信,这种通信不需要任何线缆,亦不需要用户直接手工干涉;海当一个嵌入了蓝牙技术的设备发觉另一同样嵌入蓝牙技术的设备,它们就能自动同步,相互通信,实现资源共享。

一、蓝牙的结构体系

蓝牙协议栈的体系结构如图1.15－1所示。它是由底层硬件模块、中间层和高端应用层三大部分组成。

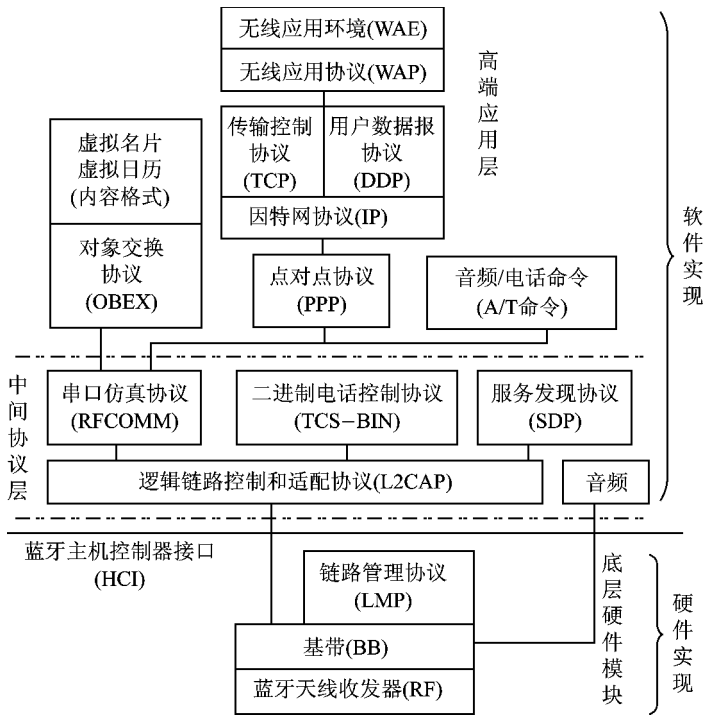


图1.15－1　蓝牙协议栈体系结构

1. 蓝牙的底层模块

底层模块是蓝牙技术的核心模块,所有嵌入蓝牙技术的设备都必须包括底层模块。它主要由链路管理层 LMP (Link Manager Protocol)、基带层 BB (Base Band) 和射频 RF (Rodio Frequency) 组成。其功能是:无线连接层 (RF) 通过 2.4 GHz 无需申请的 ISM 频段,实现数据流的过滤和传输;它主要定义了工作在此频段的蓝牙接收机应满足的要求,基带层 (BB) 提供了两种不同的物理链路 (同步面向连接链路 SCO Synchronous Connection Oriented 和异步无连接链路 ACL Asynchronous Connection Less),负责跳频和蓝牙数据及信息帧的传输,且对所有类型的数据包提供了不同层次的前向纠错码 FEC (Frequency Error Correction) 或循环冗余度差错校验 CRC (Cyclic Redundancy Check);LMP 层负责两个或多个设备链路的建立和拆除及链路的安全和控制,如鉴权和加密、控制和协商基带包的大小等,它为上层软件模块提供了不同的访问入口;蓝牙主机控制器接口 HCI (Host Controller Interface) 由基带控制器、连接管理器、控制和事件寄存器等组成。它是蓝牙协议中软硬件之间的接口,提供了一个调用下层 BB、LM、状态和控制寄存器等硬件的统一命令,上、下两个模块接口之间的消息和数据的传递必须通过 HCI 的解释才能进行。HCI 层以上的协议软件实体运行在主机上,而 HCI 以下的功能由蓝牙设备来完成,二者之间通过传输层进行交互。

2. 中间协议层

中间协议层由逻辑链路控制与适配协议 L2CAP (Logical Link Control and Adaptation Protocol)、服务发现协议 SDP (Service Discovery Protocol)、串口仿真协议或称线缆替换协议 (RFCOM) 和二进制电话控制协议 TCS (Telephony Control protocol Spectocol) 组成。L2CAP 是蓝牙协议栈的核心组成部分,也是其他协议实现的基础。它位于基带之上,向上层提供面向连接和无连接的数据服务。它主要完成数据的拆装、服务质量控制、协议的复用、分组的分割和重组 (Segmentation And Reassembly) 及组提取等功能。L2CAP 允许高达 64 KB 的数据分组。SDP 是一个基于客户/服务器结构的协议。它工作在 L2CAP 层之上,为上层应用程序提供一种机制来发现可用的服务及其属性,而服务的属性包括服务的类型及该服务所需的机制或协议信息。RFCOMM 是一个仿真有线链路的无线数据仿真协议,符合 ETSI 标准的 TS 07.10 串口仿真协议。它在蓝牙基带上仿真 RS-232 的控制和数据信号,为原先使用串行连接的上层业务提供传送能力。TCS 是一个基于 ITU-T Q.931 建议的采用面向比特的协议,它定义了用于蓝牙设备之间建立语音和数据呼叫的控制信令 (Call Control Signalling),并负责处理蓝牙设备组的移动管理过程。

3. 高端应用层

高端应用层位于蓝牙协议栈的最上部分。一个完整的蓝牙协议栈按其功能又可划分为四层:核心协议层 (BB、LMP、LCAP、SDP)、线缆替换协议层 (RFCOMM)、电话控制协议层 (TCS-BIN)、选用协议层 (PPP、TCP、TP、UDP、OBEX、IrMC、WAP、WAE)。而高端应用层是由选用协议层组成。选用协议层中的 PPP (Point-to-Point Protocol) 是点到点协议,由封装、链路控制协议、网络控制协议组成,定义了串行点到点链路应当如何传输因特网协议数据,它主要用于 LAN 接入、拨号网络及传真等应用规范;TCP/IP (传输控制协议/网络层协议)、UDP (User Datagram Protocol 对象交换协议) 是三种已有的协议,它定义了因特网与网络相关的通信及其他类型计算机设备和外围设备之间的通信。蓝牙采用或共享这些已有的协议去实现与连接因特网的设备通信,这样,既可提高效率,又可在一定程度上保证蓝牙技术和其他通信技术的互操

作性；OBEX（O bject Exchange Protocol）是对象交换协议，它支持设备间的数据交换，采用客户/服务器模式提供与 HTTP（超文本传输协议）相同的基本功能。该协议作为一个开放性标准还定义了可用于交换的电子商务卡、个人日程表、消息和便条等格式；WAP（Wireless Application Protocol）是无线应用协议，它的目的是要在数字蜂窝电话和其他小型无线设备上实现因特网业务。它支持移动电话浏览网页、收取电子邮件和其他基于因特网的协议。WAE（Wireless Application Environment）是无线应用环境，它提供用于 WAP 电话和个人数字助理 PDA 所需的各种应用软件。

二、蓝牙硬件的实现

蓝牙的技术规范除了包括协议部分外还包括蓝牙的应用部分（即应用模型）。在实现蓝牙的时候，一般是将蓝牙分成两部分来考虑：其一是软件实现部分，它位于 HCI 的上面，包括蓝牙协议栈上层的 L2CAP、RFCOMM、SDP 和 TCS 以及蓝牙的一些应用；其二是硬件实现部分，它位于 HCI 的下面，亦即上面提到的底层硬件模块，这已在图 1.15－1 中标示出。下面讨论蓝牙硬件模块的结构与性能。

蓝牙硬件模块由蓝牙协议栈的无线收发器（RF）、基带控制器（BB）和链路管理层（LMP）组成。目前大多数生产厂家都是利用片上系统技术 SOC（System－On－Chip）将这三层功能模块集中嵌在同一块芯片上。图 1.15－2 为单芯片蓝牙硬件模块结构图。它由微处理器（CPU）、无线收发器（RF）、基带控制器（BB）、静态随机存储器（SRAM）、闪存（Flash 程序存储器）、通用异步收发器（UAST）、通用串行接口（USB）、语音编/解码器（CODEC）及蓝牙测试模块组成。下面分别叙述各部分的组成及功能。

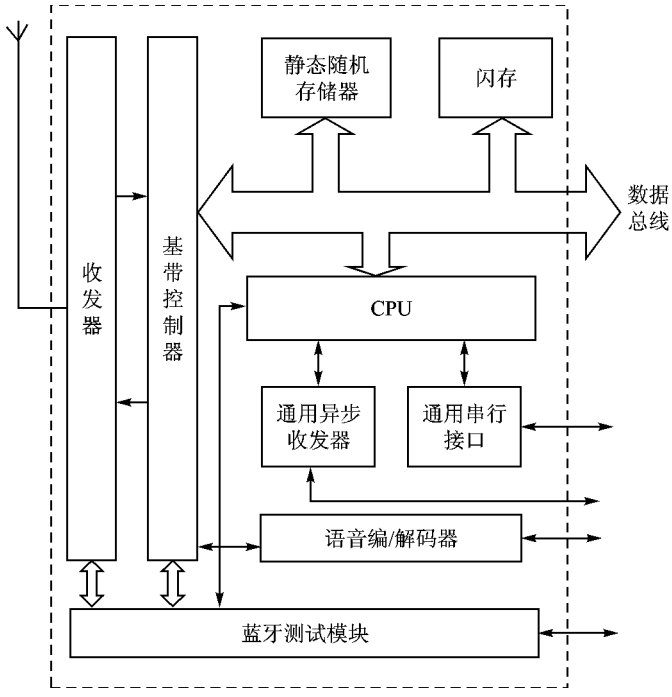


图 1.15－2 单芯片蓝牙硬件模块结构

1. 蓝牙基带控制器

蓝牙基带控制器是蓝牙硬件模块的关键模块。它主要由链路控制序列发生器、可编程序列发生器、内部语音处理器、共享 RAM 仲裁器及定时链管理、加密/解密处理等功能单元组成。其主要功能 :在微处理器模块控制下,实现蓝牙基带部分的所有实时处理功能,包括负责对接收的 bit 流进行符号定时提取和恢复 ;分组头及净荷的循环冗余度校验 (CRC) ;分组头及净荷的前向纠错码 (FEC) 处理和发送处理 ;加密和解密处理等。且能提供从基带控制器到其他芯片的接口 (诸如数据路径 RAM 客户接口、微处理器接口、脉码调制接口 (PCM) 等。

2. 无线收发器模块

无线收发器是蓝牙设备的核心,任何蓝牙设备都要有无线收发器。它与用于广播的普通无线收发器的不同之处在于体积小、功率小 (目前生产的蓝牙无线收发器的最大输出功率只有 100 mW、2. 5 mW、1 mW 三种) 。它由锁相环、发送模块和接收模块等组成。发送部分包括一个倍频器,且直接使用压控振荡器调制 (VCO) ;接收部分包括混频器、中频器放大器、鉴频器以及低噪音放大器等。无线收发器的主要功能是调制/解调、帧定时恢复和跳频功能同时完成发送和接收操作。发送操作包括载波的产生、载波调制、功率控制及自动增益控制 AGC ;接收操作包括频率调谐至正确的载波频率及信号强度控制等。

3. 微处理器 (CPU)

CPU 负责蓝牙比特流调制和解调后的所有比特级处理,且还负责控制收发器和专用的语言编码和解码器。

4. Flash 存储器和 SRAM

Flash 存储器用于存放基带和链路管理层中的所有软件部分。SRAM 作为 CPU 的运行空间,在工作时把 Flash 中的软件调到 SRAM 中。

5. 语音编/解码器 CODEC (Coder Decoder)

语音编/解码器 CODEC 由 ADC (数模转换器) 、模数转换口 (ADC) 、数字接口、编码模块等组成。主要功能 :提供语音编码和解码功能,提供 CVSD (Continuous Variable Slope Delta Modulation) 即连续可变斜率增量调制及对数 PCM (Pulse Coded Modulation) 即脉码调制两种编码方式。

6. 蓝牙测试模块

蓝牙测试模块是由 DUT (Device Under Test) 即被测试模块与测试设备及计量设备组成。一般测试设备和被测试设备构成一个微微网,测试设备是主节点,DUT 是从节点。测试设备对整个测试过程进行控制,其主要功能提供无线层和基带层的认证和一致性规范,同时还管理产品的生产和售后测试。

7. 通用异步收发器 UART (Universal Asynchronous Receiver Transunitter) 和通用串行接口 USB (Universal Serial Bus)

功能 :提供到 HCI (Host Confroller Interface) 即主机控制器接口传输层的物理连接,是高层与物理模块进行通信的通道。

三、TR0700 单芯片介绍

TR0700 单芯片是 Transilica 公司的蓝牙产品,其结构如图 1. 15 - 3 所示。它把无线收发器与基带都集成到一块 CMOS 芯片上,替代传统的串行语音和通用串行接口电缆,为语音和数

据业务提供无线连接。

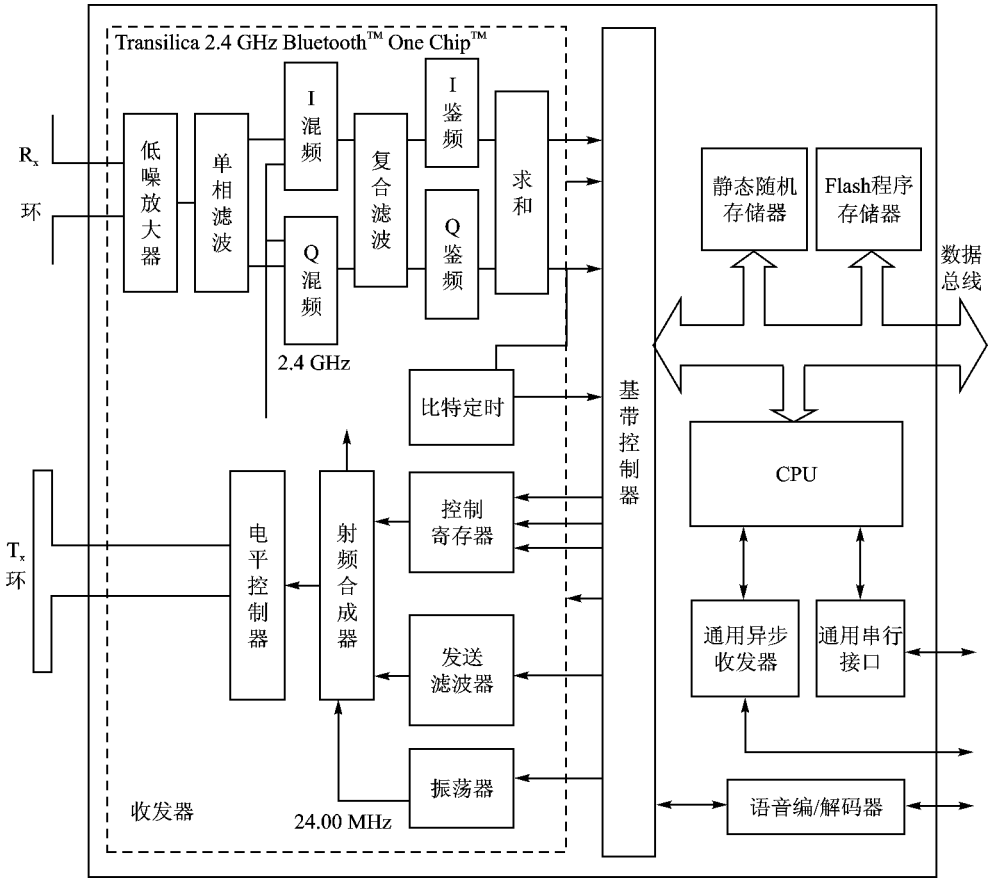


图 1.15－3 TR0700 原理框图

1. 结构及工作原理

TR0700 单芯片由收发器、基带、语音编/解码器（CODEC）、带有 4 个可配置的 8 位接口的 8051 微处理器、两个串行口和双高性能的通用异步收发器（UART）、4 KB 的静态随机存储器（SRAM）、64 KB 的 Flash 程序存储器等组成。

收发器由低噪放大器（LNA）、电平控制器（PA）、混频器、鉴频器、控制寄存器、发送滤波器、振荡器等组成。其工作原理是：来自接收天线上的信号经低噪放大器（LNA）放大后，送至多级滤波器；多级滤波器具有预选择功能，可把 LAN 的输出信号限制在 2.4 GHz 的 ISM 频段内，去除负频率成分，输出适合进行下变频处理的信号。I、Q 混频器把蓝牙频段的信号移频至低中频（IF）传输的调制信号。复合滤波器负责从下变频信号中滤除无用信号和噪声。鉴频器使用过采样技术从 IF 信号中取出蓝牙低调制指数信号；发送器由发送滤波器、频率合成器、功率放大器、振荡器、天线等组成。其工作原理是：发送滤波器是一个高斯数字滤波器，可对发送环 T_X 输入的数据进行数字过滤；振荡器的功能是驱动一个外部的晶体振荡器或者接受一个外部的时钟信号，向频率合成器提供一个低噪声的参考频率。功率放大器的主要功能是对频率合成器的输出功率放大到 1 mW 左右，且对频率合成器起缓冲作用，减少负载变化对合成器的影响；发送天线，当使用差分输入的 LNA 时，可以是一个低噪声的平衡双极天线；8051 微处

理器是一个 8 位的微处理器,主要功能是管理和实现蓝牙协议栈。它具有一个增强的指令集、二级数据指针、扩展的 SRAM 和双 UART。在 TR0700 中对一些重复性的操作诸如分组的组装和拆解、加密、地址编码/解码、纠错和同步等都由硬件来实现,这样能降低处理器的开销,有效地提高响应性能。TR0700 除了 8051 微处理器本身所带有的一些特殊功能寄存器 (SFR) 外,还定义了一些新的特殊功能寄存器 (SFR)。它还引入了一些特殊的中断,如一个带有特殊保护的外部中断 INT3 等。TR0700 的基带操作有三种模式可供选择:数据/地址、端口、测试。

2. 基本功能及应用

TR0700 单芯片的基本功能是:具有 10 m 的传输距离及 1 Mbps 的数据速率;支持 79 跳系统及支持点到点、点到多点连接,既可以是主节点又可以是小节点;支持 GAP、TCS、手机、intercom 剖面和串行口等;支持 Hold、Sniff 和 Park 功率节省模式;对 LC、LM、L2CAP、SDP、RFCOMM 等蓝牙协议栈能完全实现;对于 SCO 链路支持 HV1、HV2、HV3 数据分组;对于 ACL 分组支持 DM1、DM3、DM5、DH1、DH3、DH5 和 AUX1 数据分组;具有用于测试和 Flash 内存升级的 JTAG 接口。TR0700 单芯片的主要应用有:用于电信方面的峰房和无绳电话、调制解调器、手持设备、互联设备、小型监视器;用于计算机方面有键盘、鼠标、控制杆、扫描仪、监视器、打印机、桌面、笔记本计算机等;用于消费类的 PDA、耳机、监视系统、游戏控制器和数字相机等。

蓝牙技术作为一个开放的无线应用标准,能通过无线连接方式将一定范围内的固定或移动设备连接起来,使人们能够更方便更快速地进行语音和数据的交换,这无疑将会成为未来无线通信领域的一个重要的研究方向。本文所描述的蓝牙技术硬件实现模式分析,只是蓝牙核心技术中的一小部分,随着蓝牙技术的不断完善与产品的成功开发,可以肯定,蓝牙技术将会逐渐进入我们的工作和生活,成为不可缺少的一部分。

参 考 文 献

1 Bluetooth overview. http:hhwww. bluetooth,2000. 12

2 Bluetooth profiles. http:hhwww. palowireless. com, 2000. 12

3 Bluetooth tutorial. http:hhwww. bluetooth. 2000. 12

4 http:hhwww. research. com

5 李纯,周开波译. 蓝牙技术起跳 [M]. 北京:电子工业出版社,2002

选自《电子技术应用》月刊,2002 年第 11 期

1.16 Bluetooth 技术与相关器件

南京东南大学无线电系(210096) 徐平平

一、前 言

Bluetooth 技术是当前引人注目的热点课题之一。Bluetooth 技术是一种低功率短距离,支持点到点,点到多点的话音、数据业务的无线连接技术标准的代称。它由 Ericsson、Nokia、Inter、IBM 和 Toshiba 等世界级大公司提出、制定与推广。“蓝牙”一词取自公元 10 世纪统一丹麦的国王哈拉德二世(Harald)的绰号——“蓝牙(Bluetooth)”。Ericsson 为这种即将成为全球通用的无线技术命此名,也许有一统天下的含义。从 1998 年 5 月提出以来,推出了 1.0A、1.0B 标准,即将推出 2.0 标准。Bluetooth 技术作为有线网络的延伸,实现无缝覆盖。Bluetooth 技术的出现极大地推动了个人局域网(Personal Area Network,PAN)技术的发展,IEEE802.15 小组负责研究基于 Bluetooth 的 PAN 技术。

二、Bluetooth 技术概貌

蓝牙技术的实质内容是要建立通用的无线电空中接口及其控制软件的公开标准,使通信和计算机进一步结合,使不同厂家生产的便携式设备在没有电线或电缆相互连接的情况下,能在近距离范围内具有互用、互操作的性能。蓝牙技术可以使蜂窝电话系统、无绳电话系统、无线局域网和因特网等现有网络增添新的功能,使各类计算机、传真机、打印机乃至各种室内电子、信息和电器设备增添廉价的无线传输和组网的功能,方便地形成个人网络。表 1.16-1 给出了 Bluetooth 1.0 标准的摘要。该标准有详细的产品设计规范,并有产品互操作性要求。有了这一完整的技术规范,全球的技术开发商就可以开始采用 Bluetooth 技术的产品开发和互操作测试,从而使 Bluetooth 技术向产品化实用化方向迈进了一大步。

Bluetooth 技术具有以下特征:

- 工作在 2.4 GHz ISM 频段,频段全球统一;

表 1.16-1 Bluetooth 1.0 标准摘要

范 围	10 m(100 m 加功率放大器)
总比特率	1 Mbps
最大同步速率	432.6 Kbps(对称数据)
最大异步速率	723.3 Kbps/57.6k(非对称数据)
每个链路的最多语言信道数	3 个 64 Kbps
基带协议	电路交换与分组交换的结合
最多连接终端数	8

续表 1.16－1

范 围	10 m(100 m 加功率放大器)
无线频率	2.4 GHz(开放频带)
跳频速率	1 600 bps
“监听”信息周期	1.28 s

· 通信距离为 10 cm～100 m,1 mW 的发射功率可以覆盖 10 m 的范围；

· 采用跳频扩频通信技术克服干扰,跳频速率为 1 600 hops/s；

· 采用全向天线,不受视距的限制；

· 支持点到点、点到多点的话音和数据业务；

· 数据传输速率为 1 M symbol/s/piconet；

· 有多级的低功耗模式,适合于消费类电子应用；

· 内建多级的安全与鉴权体制。

Bluetooth 技术的主要应用范围：

· 话音/数据的接入。这是将一台计算设备通过完全的无线链路连接到一个通信设备上,完成与广域通信网络的互连；

· 外围设备互连。外围设备互连是指将各种外围设备通过 Bluetooth 链路连接到主机。

· 个人局域网 PAN(Personal Area Network)。PAN 的注意力集中在个人网络的构造上,完成信息的共享和交换。

“蓝牙”技术使得现代一些轻易携带的移动通信设备和电脑设备,不必借助电缆就能联网,并且能够实现无线上因特网,其实际应用范围还可以拓展到各种家电产品、消费电子产品和汽车等信息家电,组成一个巨大的无线通信网络。

三、Bluetooth 技术实现解决方案

Bluetooth 技术的实现主要介绍 Philips Semiconductous 的套片,该公司以 System－on－Silicon 的设计理念,不仅进行单个 IC 器件的开发,重点考虑总体系统的功能,以系统最优为目标开展研究开发设计。图 1.16－1 的 Bluetooth 模型框图就是一个 Bluetooth 系统解决方案。

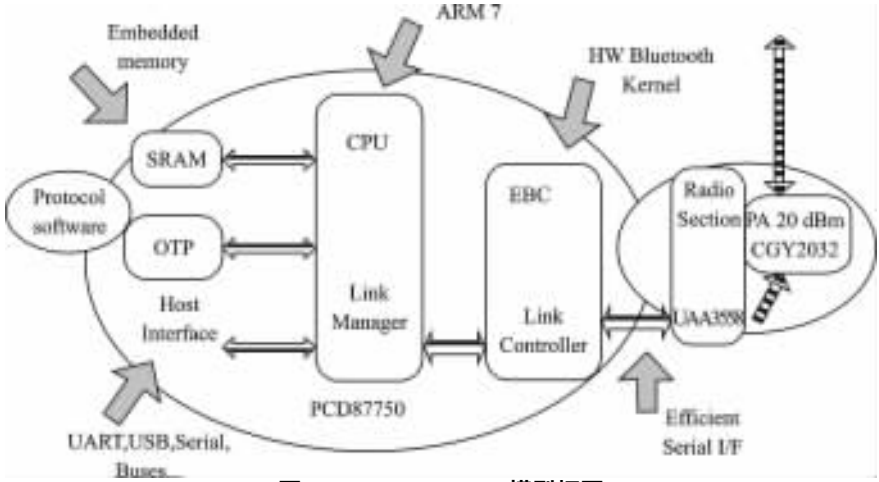


图 1.16－1 Bluetooth 模型框图

该方案中用 UAA3558 单片 RF 收发器件,称之为 NZIF (Near Zero IF) 方式。接收部分的信道滤波器用 SAW 声表面波滤波器,不用传统的石英晶体滤波器,大大降低了成本费用。器件内置了 VCO、PLL,降低了外围电路的复杂度。图 1.16－2 给出了 UAA3558 的电路图。

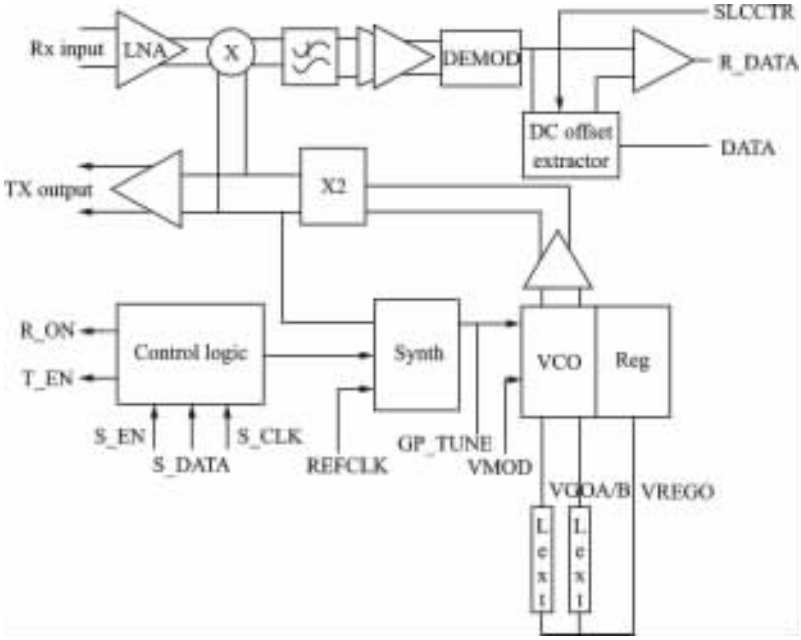


图 1.16－2 UAA3558 电路图

UAA3558 由来自基带芯片的 3 根串行接口控制,如此简单的 RF 芯片并没有以牺牲性能为代价,接收灵敏度 - 90 dBm,输出功率 4 dBm,跳频速率 160/s,对应的高速 PLL 充分满足了必要的性能。更进一步的特点是设置了发送模式、接收模式、降功率待机模式。可控制降功率待机模式的功耗电流在 60 μA 以下。器件的外型为 BCC32pin,5 mm×5 mm。若需要 20 dBm 输出功率可以另外加功率放大器 PA。

与 UAA3558 RF 芯片对应的基带芯片 VWS26003VLSI 的电路方框图如图 1.16－3 所示。这是可编程的芯片,和 UAA3558 的接口要一块专用 ASIC。器件外型是 LFBGA96pin 封装 (8 mm×8 mm)。

PCD87750 是 Philips 开发的 OTP (One Time Programmable) 芯片,可以直接和 RF 接口。图 1.16－4 是 PCD87750 的电路方框图,这是 LFBGA108pin (8 mm×8 mm) 封装。

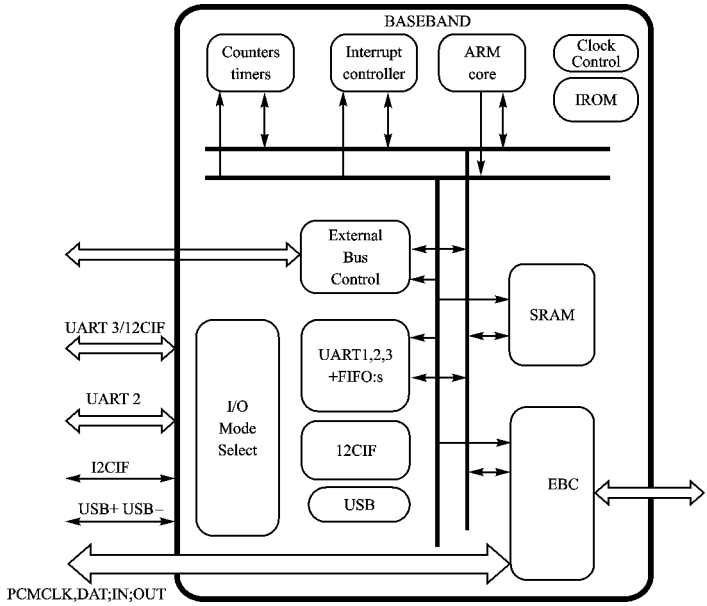


图 1. 16 − 3 VWS26003 **方框图**

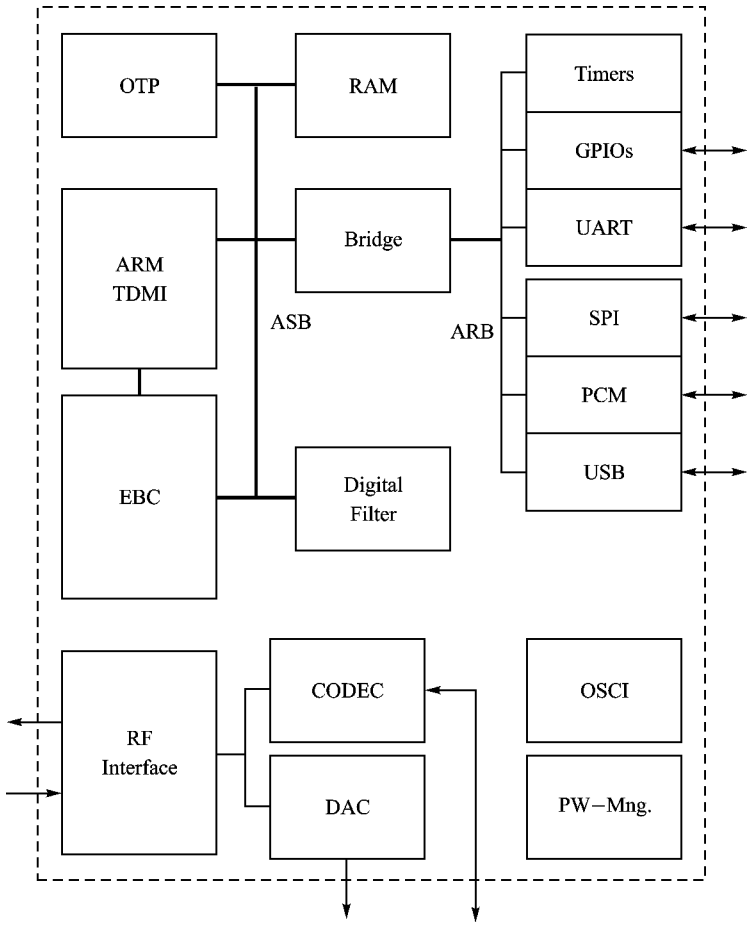


图 1. 16 − 4 PCD87750 **方框图**

四、今后的开发动向

Bluetooth 技术能否在今后迅速普及,Bluetooth 市场潜力受到肯定,但是在日内瓦的 Bluetooth 技术研讨会上,多数人对 Bluetooth 整体解决方案的价格表示担忧,降到 5 美元难度很大。目前问世的具有 Bluetooth 功能的产品单硬件成本价格就在 20 ~ 25 美元之间,过高的成本意味着该技术仍然仅限于应用在膝上型电脑等高端产品上。显然,降低费用成本是一个非常重要的因素,Philips 的 Bluetooth 芯片组已经将成本降了一半 (10 美元),最终计划目标是 5 美元。图 1.16 - 5 是该公司的 Bluetooth 产品成本售价计划图。

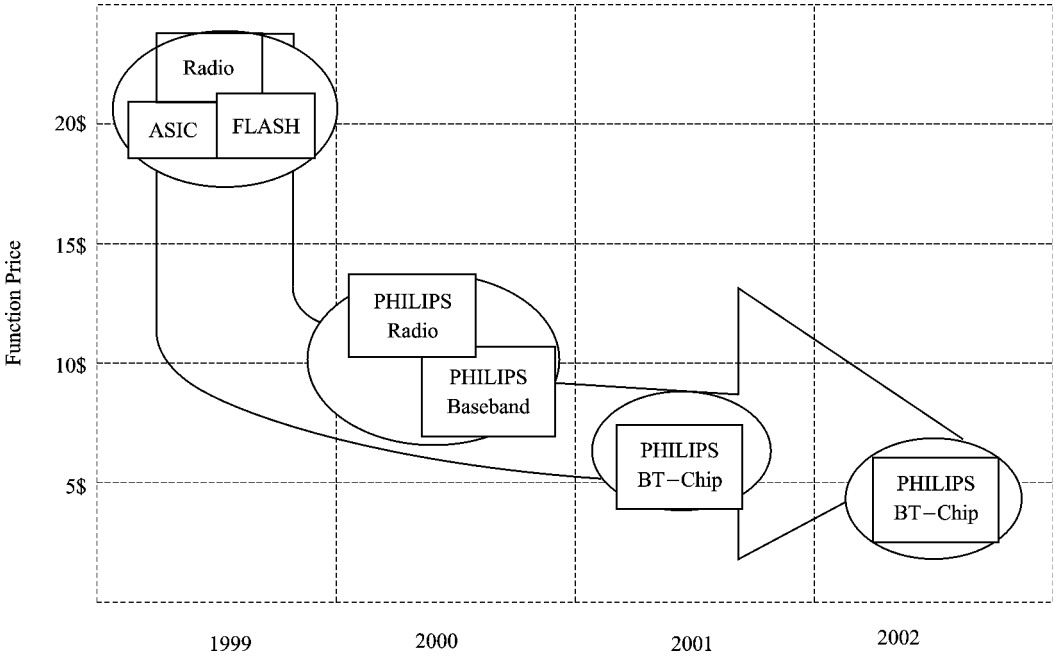


图 1.16 - 5 Bluetooth Cost Trading

尽管 Bluetooth 是一种对未来移动计算技术有深远影响的技术,但是发展道路不会一帆风顺。Microsoft 公司至今仍在对标准的内容进行评估,以决定在其操作系统中是否支持 Bluetooth 技术。但人们越来越清晰地看到,这一代表着电信与计算机业合作的 Bluetooth 技术在未来随时随地的、无处不在的计算环境中发挥不可低估的作用。“蓝牙”这颗小牙能否占据“无线商机”,我们拭目以待,未来会更精彩。

参 考 文 献

1 多田敏宏. Bluetooth 技术と对应デ バイスの介绍. 日本电子技术, Vol. 3. 2000

2 http / / :www. bluetooth. com

选自《集成电路应用》月刊,2002 年第 4 期

1.17 基于蓝牙技术的无线收发芯片 nRF401

绍兴水联智能仪表有限公司 张寅刚

一、芯片特点

nRF401 是北欧集成电路公司最新推出的无线收发一体的芯片,采用蓝牙核心技术设计,工作于 433/434 MHz 的 ISM(Industrial Scientific Medical)频段。nRF401 将很多功能和外围部件协议集成在芯片内部,是目前业界惟一一个可以直接接单片机串口进行数据传输的无线收发芯片,经 RS232 电平转换后也可接至计算机串口。

nRF401 系列无线收发芯片的显著特点是外围元件极少(约 10 个,其他产品如 TRF6900 外围元件达 50 个),没有调试部件,便于开发与生产。nRF401 采用便宜且易于获得的 4 MHz 晶体作为 PLL 频率基准源,无需外接昂贵的变容二极管,而其他竞争产品大多需要外接变容二极管。其他无线收发芯片一般需要进行曼彻斯特编码后才能传输,大大增加了软件的工作量和产品开发的难度,而 nRF401 系列独特的技术可以直接传送单片机串口数据。采用 DSS(直接数字合成) + PLL(锁相环稳频)频率合成技术,频率稳定性好,具有较强的抗干扰能力。由于发射功率低,接收灵敏度高,因此使用时无需申请许可证,能满足无线手持设备与嵌入系统的要求,广泛应用于保安系统、无线远程抄表、家庭自动化、遥控、遥测、非接触智能卡、无线通信等领域。nRF401 与 A/D、D/A 器件配合使用,将有关信号转换成数字信号后还可以传输语音甚至图像信号。

二、内部结构与功能参数

1. 电气特性

nRF401 的发射功率最大 10 mW,工作电压 2.7 ~ 5 V,发射电流 8 ~ 30 mA(与发射功率有关),接收电流约 10 mA,待机电流 8μA,灵敏度 - 105 dBm,封装 20 脚贴片,速率最高 19.2 Kb/s,工作频段 433/434 MHz,有 2 个信道,调制方式 FSK,适合工业控制场所。

nRF401 参数指标如表 1.17 - 1 所列。

表 1.17 - 1

项 目	参数指标	条 件
信道#1/信道#2 频率/MHz	433.92/434.33	—
调制方式	FSK	—
频率差/kHz	±15	—
最大发射功率/dBm	10	400 Ω,3 V
灵敏度/dBm	- 105	400 Ω,BR = 20 Kb/s,BER < 10
最大传输速率/Kb/s	20	—

续表 1.17－1

项 目	参数指标	条 件
工作电压/V	2.7～5.25	—
接收电流/mA	250	—
发射电流/mA	8～30	10 dBm 输出功率
备用状态电流/μA	8	—

注：信号功率的单位为瓦（W）、毫瓦（mW）。为了方便，功率有时也用电平来表示。设某信号功率为 P（W），P（W）与单位信号功率（1W 或者 1 mW）比的对数称为该信号功率的绝对电平或功率的绝对值，用符号 P 来表示，单位为分贝瓦（dBW）、分贝毫瓦（dBm）等。P（dBW）＝10 lg［P（W）/1W］，P（dBm）＝10 lg［P（W）×1000/1 mW］

2. 内部结构与外围器件

nRF401 的内部结构与外围器件如图 1.17－1 所示。

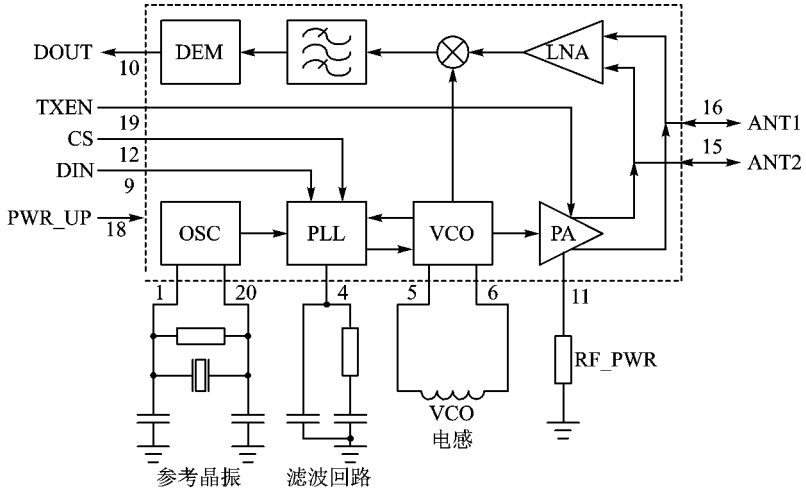


图 1.17－1

从引脚 9 输入的数字信号经 PLL、VCO 完成频率合成与调制。射频振荡器的中心频率由外部参考晶振决定，外接滤波回路为 VCO 提供反馈电压，用于稳频。调制后的射频信号经 PA 功率放大后从天线端输出。

从天线接收到的射频信号经低噪高增益放大器 LNA 放大，再经滤波、频率解调后转换成数字信号，从引脚 10 输出。

3. 引脚功能

nRF401 为 20 引脚、SSOIC 封装。引脚功能如表 1.17－2 所列。

表 1.17－2

引脚	名称	引脚功能
1	XC1	晶振输入
2	V_{DD}	电源(DC:+3～-5 V)
3	V_{SS}	接地(0 V)
4	FIFT1	环型过滤器输入
5	VCO1	VCO 外部感应器输入
6	VCO2	VCO 外部感应器输入
7	V_{SS}	接地
8	V_{DD}	电源(DC:+3～-5 V)
9	DIN	数据输入
10	DOUT	数据输出
11	RF_PWR	发射功率设定,输入
12	CS	信道选择输入:CS="0"为 433.92 MHz(信道#1) CS="1"为 434.33 MHz(信道#2)
13	V_{DD}	电源(DC:+3～-5 V)
14	V_{SS}	接地
15	ANT2	天线端
16	ANT1	天线端
17	V_{SS}	接地
18	PWR_UP	电源开关输入:PWR_UP="1"为上电(工作模式) PWR_UP="2"为下电(备用模式)
19	TXEN	发射使能:TXEN="1"为发射模式 TXEN="0"为接收模式
20	XC2	晶振输出

三、应用设计

1. 天线与射频输出能量

ANT1 和 ANT2 引脚在 nRF401 处于接收模式时为片内 LNA(Low Noise Amplifer)提供 RF 输入,在芯片发射模式时从内部 PA(Power Amplifer)输出 RF 信号。nRF401 连接微带天线,推荐负载为 400 Ω。

图 1.17－2 所示为印刷板上带微带天线的典型应用电路。其中输出级(PA)包含 2 个集电极开路的三极管,对 PA 的电源供应必须通过集电极负载。如图 1.17－2 在 ANT1/ANT2 引脚连接的微带天线,电源必须加在天线环的中心;也可以用单端天线或 50 Ω 测试仪器,通过一个差动匹配网络连到芯片上。

连接在 RF_PWR 脚与 VSS 脚之间的外部偏置电阻 R3 用来调节输出放大器(PA)的增益,从而调节射频输出能量。在 400Ω 的负载下,发射功率最大能达到 +10 dBm。

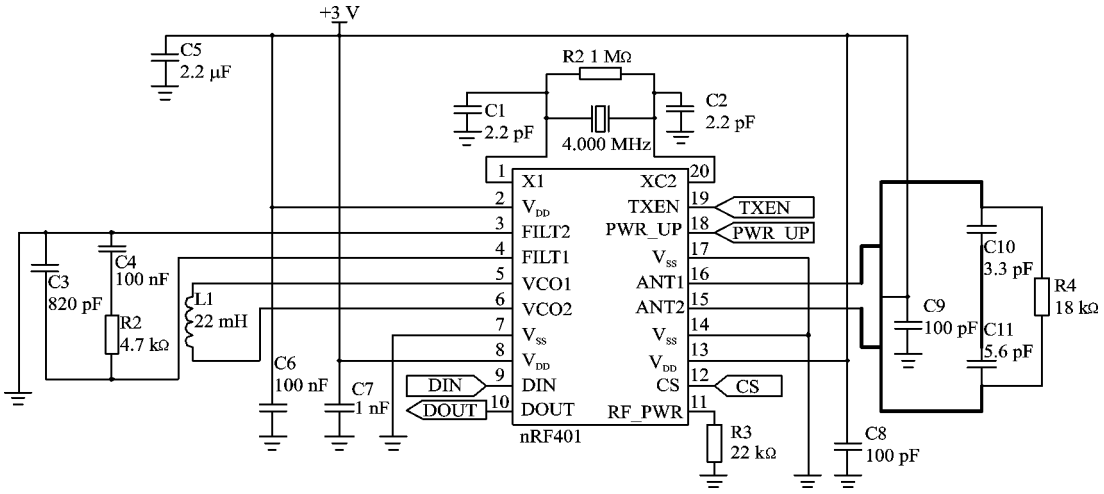


图 1.17－2

2. PLL 环路滤波器、VCO 电感与参考晶振的选择

频率合成器的二阶锁相环路需外接一个单端的相位滞后滤波器，推荐的器件参数为：

$$C3 = 820\ \text{pF},\ C4 = 15\ \text{nF},\ R2 = 4.7\ \text{k}\Omega$$

片上的压控振荡器（Voltage Controlled Oscillator）需要在 VCO1 与 VCO2 引脚间接一个 22 mH 的电感。该电感必须为高质量的片型电感，$Q > 45$（在 433 MHz，最大偏差不超过 ±2% 条件下）。

为使晶体振荡器节能高效地工作，需要选择适当的器件作为晶体衰减和电容负载。晶体参数如下：

晶体本振频率：$f = 4.000\ 0\ \text{MHz}$

晶体等效电容：$Co5 \leqslant 5\ \text{pF}$

晶体等效串联电阻：$R_{ES} \leqslant 150\ \Omega$

总电容负载，包括 PCB 配线的电容：$C_L \leqslant 14\ \text{pF}$

因此，如图 1.17－2 所示晶振中的电容负载为：

$$C_L = \frac{C1' \times C2'}{C1' + C2'}$$

其中 $C1' = C1 + C_{PCB1}$，$C2' = C2 + C_{PCB2}$

芯片和微控制器共用一个晶振时，参考晶振的引线必须远离全速率数据与控制信号线。

3. PCB 布线和抗干扰设计

PCB 板的设计关系到整个系统的射频性能。推荐使用至少 2 层的 PCB 板，并且包含一个接地层。nRF401 的电源必须经过滤波并且与数字电路的电源隔离，最好与小电容并接 1 只表面贴装的大电容，避免使用长电源线。VDD 电源与旁路电容应尽量靠近模块。VSS 端可靠接地。

全速率数据和控制信号线不能靠近 PLL 滤波器的元件或外部 VCO 电感。

VCO 电感的布置也很重要。VCO 电感处于最佳位置时，PLL 滤波回路端的电压为 1.1 + 0.2 V，能在 FILT1（pin4）引脚测量到。对于 0603 型的电感，其中点与 VCO1、VCO2 中点

间的距离应为 5.4 mm。

4. 无线通信距离的计算

通信距离与发射功率、接收灵敏度和工作频率有关。下面的公式说明在自由空间下电波传播的损耗：

$$Los = 32.44 + 20 \lg d + 20 \lg f$$

式中：Los 是传播损耗，单位为 dB；d 是传输距离，单位是 km；f 是工作频率，单位是 MHz。

由上式可见，自由空间中电波传播损耗（亦称衰减）只与工作频率 f 和传播距离 d 有关，当 f 或 d 增大 1 倍时，Los 将分别增加 6 dB。

一个工作频率为 433.92 MHz，发射功率为 +10 dBm（10 mW），接收灵敏度为 - 105 dBm 的系统在自由空间的传播距离为：

① 由发射功率 +10 dBm，接收灵敏度为 - 105 dBm，有

$$Los = 115 \text{ dB}$$

② 由 Los、f 计算得出 d = 9.7 km

这是理想状况下的传输距离，实际应用中是会低于该值的。这是因为无线通信要受到各种外界因素的影响，如大气、阻挡物、多径等造成的损耗。将上述损耗的参考值计入上式，即可计算出近似通信距离。

假定大气、遮挡等造成的损耗为 25 dB，可以计算得出通信距离为：d = 1.7 km。

参 考 文 献

1 张肃文. 高频电子线路. 北京：高等教育出版社，1979

选自《单片机与嵌入式系统应用》月刊，2002 年第 2 期

1. 18　蓝牙收发芯片 RF2968 的原理及应用

南华大学　黄智伟　王　彦　廖金盛

一、概　述

RF2968 是为低成本的蓝牙应用而设计的单片收发集成电路,RF 频率范围 2 400 ~ 2 500 MHz,RF 信道 79 个,步长 1 MHz,数据速率 1 MHz,频偏 140 ~ 175 kHz,输出功率 4 dBm,接收灵敏度 - 85 dBm,电源电压 3 V,发射消耗电流 59 mA,接收电流消耗 49 mA,休眠模式电流消耗 250 μA。芯片提供给全功能的 FSK 收发功能,中频和解调部分不需要滤波器或鉴频器,具有镜像抑制前端、集成振荡器电路、可高度编程的合成器等电路。自动校准的接收和发射 IF 电路能优化连接的性能,并消除人为的变化。RF2968 可应用在蓝牙 GSM/GPRS/EDGE 蜂窝电话、无绳电话、蓝牙无线局域网、电池供电的便携设备等系统中。

二、引脚功能

集成电路采用 32 脚的塑料 LCC 形式封装,各引脚功能如下。

VCC1 :给 VCO (压控振荡器) 倍频和 LO (本机振荡器) 放大器电路提供电压。

VCC2 :给 RX (接收) 混频器、TXPA (发射功率放大器) 和 LNA (低噪声放大器) 偏置电路提供电压。

TXOUT :发射机输出。当发射机工作时,TXOUT 输出阻抗是 50 Ω ;当发射机不工作时,TXOUT 为高阻态。因为这个引脚是直流偏置,所以需外接 1 个耦合电容。

RXIN :接收机输入。当接收机工作时,RXIN 输入阻抗是低阻态 ;接收机不工作时,RXIN 为高阻态。芯片内用 1 个内部串联电感来调节输入阻抗。

VCC3 :给 RX 输入级 (LNA) 提供电压。

VCC4 :给 TX 混频器、LO 放大器、LNA 和 RX 混频器的偏置电路提供电压。

LPO :低功耗模式的低频时钟输出。在休眠模式中,这个引脚能给基带提供一个 3. 2 kHz 或 32 kHz、占空比为 50 % 的时钟。在其他工作方式没有输出。

DVDDH :给 RX IF VGA (接收中频电压增益放大器) 电路提供电压。

IRE F :外部接 1 个精密电阻以产生恒定的基准电流。

VCC5 :给模拟中频电路提供电压。

D1 :这是为时钟恢复电路提供的电荷泵输出。外接 1 个 RC 网络到地以确定 PLL 的带宽。

BPKTCTL :在发射模式时,这个脚作为启动 PA 级的选通脉冲 ;在接收模式时,基带控制器可以有选择地使用这个引脚来给同步字的检测发信号。

BDATA1 :输入信号到发射机/接收机的数据输出。输入的数据是速率为 1 MHz 的没有被滤波的数据。这个引脚是双向的,根据发射和接收模式转换为数据输入或数据输出。

RECCLK :恢复时钟输出。

RECDATA :恢复数据输出。

BXTLEN :功率控制电路的一部分,用来接通/关断芯片的"休眠"模式。在电路从"OFF"状态上电之后,当低功耗时钟不工作时,BR CLK 被 BXTLEN 的状态控制(上电期间,BRCLK 先与 BXTLEN 激活且被设为高电平,以进入空闲状态)。

BRCLK :基准时钟输出。这是由晶振决定的基准时钟,频率范围为 10 ~ 40 MHz,典型值为 13 MHz。电路上电时,BRCLK 在基带控制器将 BXTLEN 设为高电平之前激活。电路进入空闲状态后,当低功耗时钟不工作时,BRCLK 由 BXTLEN 的状态控制。

OSC O :与 19 脚相同。

OSC I :OSC 脚可通过负反馈的方式来产生基准时钟。在 OSC I 到 OSC O 之间连接 1 个并联的晶振和电阻,以提供反馈通道和确定谐振频率。每一个 OSC 脚都接 1 个旁路电容来提供合适的晶振负载。如果用 1 个外部的基准频率,那就要通过 1 个隔直电容来连接到 OSC I,并且用 1 个 470 kΩ 的电阻将 OSC O 和 OSC I 连接起来。

BnDEN :锁存输入到串行端口的数据。数据在 BnDEN 的上升沿被锁存。

BDDATA :串行数据通道。读/写数据通过这个引脚送入/输出到芯片上的移位寄存器。读取的数据在 BDCLK 的上升沿被传送,写数据在 BDCLK 的下降沿被传送。

BDCLK :串行端口的输入时钟。这个引脚被用来将时钟信号输入到串行端口。要使得跳变频率的编程时间最短时,建议使用 10 ~ 20 MHz 的 BRCLK 频率。

BnPWR :芯片电源控制电路的一部分,用来控制芯片从"OFF"状态到电源接通状态。

PLLGND :RF 合成器、晶体振荡器和串行端口的接地端。

VCC6 :RF 合成器、晶体振荡器和串行端口的电源端。

DO :RF PLL 的充电泵输出。外接 1 个 RC 网络到地以确定 PLL 带宽。要使得合成器的设置时间和相位噪声最小,可采用双重的环路带宽方案。在频率检测的开始时期,使用 1 个宽环路带宽。在检测频率结束时,用 RSHUNT 来转换到窄环路带宽,并提供改进的 VCO 相位噪声。带宽转换的时间由 PLL Del 位设置。

RSHUNT :通过将 2 个外部串联电阻的中点分路到 V_{REG},使环路滤波器从窄带转换到宽带。

RESNTR - :用来给 VCO 提供直流电压以及调节 VCO 的中心频率。在 RESNTR - 和 RESNTR + 之间需 2 个电感来跟内部电容形成谐振。在设计印制板时,应该考虑从 RESNTR 脚到电感器的感抗。可以在 RESNTR 脚之间加 1 个小电容来确定 VCO 的频率范围。

RESNTR + :见引脚 28。

VREG :电压调节输出(2.2 V)。需 1 个旁路电容连接到地。通过与 28 脚和 29 脚相连的回路给 VCO 提供偏置。

IFDGND 数字中频电路接地端。

VCC7 数字中频电路电源电压。

三、内部结构

RF2968 是专为蓝牙的应用而设计,工作在 2.4 GHz 频段的收发机。符合蓝牙无线电规范 1.1 版本功率等级二(+4 dBm)或等级三(0 dBm)要求。对功率等级一(+20 dBm)的应用,RF2968 可以和功率放大器搭配使用,如 RF2172。RF2968 的内部框图如图 1.18 - 1 所示。芯

片内包含有发射器、接收器、VCO、时钟、数据总线、芯片控制逻辑等电路。

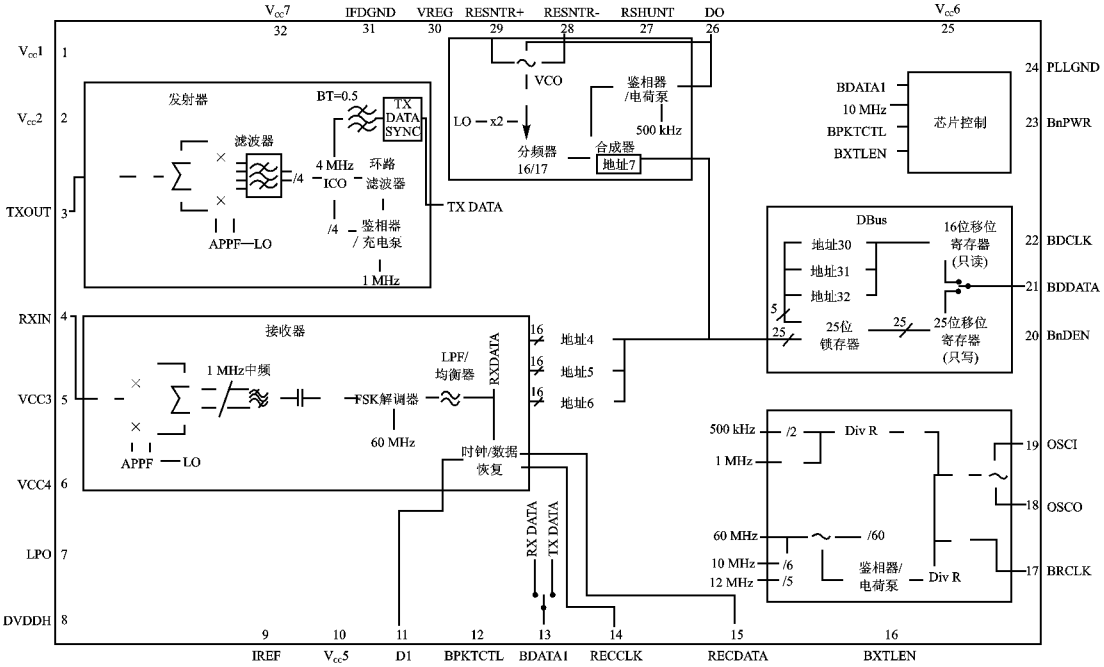


图 1.18－1 RF2968 内部框图

由于芯片内集成了中频滤波器，RF2968 只需最少的外部器件，避免外接如中频 SAW 滤波器和对称–不对称变换器等器件。接收机输入和发射机输出的高阻状态可省去外部接收机/发射机转换开关。RF2968 和天线、RF 带通滤波器、基带控制器连接，可以实现完整的蓝牙解决方案。除 RF 信号处理外，RF2968 同样能完成数据调制的基带控制、直流补偿、数据和时钟恢复功能。

RF2968 发射机输出在内部匹配到 50 Ω，需要 1 个 AC 耦合电容。接收机的低噪声放大器输入在内部匹配 50 Ω 阻抗到前端滤波器。接收机和发射机在 TXOUT 和 RXIN 间连接 1 个耦合电容，共用 1 个前端滤波器。此外，发射通道可通过外部的放大器放大到 + 20 dBm，接通 RF2968 的发射增益控制和接收信号强度指示，可使蓝牙工作在功率等级一。RSSI 数据经串联端口输入，超过 - 20 ~ 80 dBm 的功率范围时提供 1 dB 的分辨率。发射增益控制在 4 dB 步阶内调制，可经串联端口设置。

基带数据经 BDATA1 脚送到发射机。BDATA1 脚是双向传输引脚，在发射模式作为输入端，接收模式作为输出端。RF2968 实现基带数据的高斯滤波、FSK 调制中频电流控制的晶体振荡器 (ICO) 和中频 IF 上变频到 RF 信道频率。

片内压控振荡器 (VCO) 产生的频率为本振 (LO) 频率的一半，再通过倍频到精确的本振频率。在 RESNTR + 和 RESNTR - 间的 2 个外部回路电感设置 VCO 的调节范围，电压从片内调节器输给 VCO，调节器通过 1 个滤波网络连接在 2 个回路电感的中间。由于蓝牙快速跳频的需要，环路滤波器 (连接到 DO 和 RSHUNT) 特别重要，它们决定 VCO 的跳变和设置时间。所以，极力推荐使用应用电路图中提供的元件值。

RF2968 可以使用 10 MHz、11 MHz、12 MHz、13 MHz 或 20 MHz 的基准时钟频率，并能支持

这些频率的 2 倍基准时钟。时钟可由外部基准时钟通过隔直电容直接送到 OSC1 脚。如果没有外部基准时钟,可以用晶振和 2 个电容组成基准振荡电路。无论是外部或内部产生的基准频率,使用 1 个连接在 OSC1 和 OSC2 之间的电阻来提供合适的偏置。基准频率的频率公差须为 20×10^{-6} 或更好,以保证最大允许的系统频率偏差保持在 RF2968 的解调带宽之内。LPO 脚用 3.2 kHz 或 32 kHz 的低功率方式时钟给休眠模式下的基带设备提供低频时钟。考虑到最小的休眠模式功率消耗,并灵活选择基准时钟频率,可选用 12 MHz 的基准时钟。

接收机用低中频结构,使得外部元件最少。RF 信号向下变频到 1 MHz,使中频滤波器可以植入到芯片中。解调数据在 BDATA1 脚输出,进一步的数据处理用基带 PLL 数据和时钟恢复电路完成。D1 是基带 PLL 环路滤波器的连接脚。同步数据和时钟在 REDATA 和 RECCLK 脚输出。如果基带设备用 RF2968 做时钟恢复,D1 环路滤波器可以略去不用。

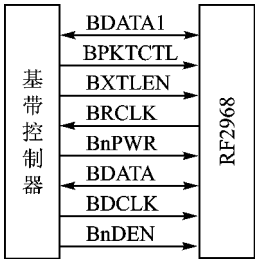


图 1.18－2 基带控制器与 RF2968 接口

四、应 用

RF2968 射频收发机作为蓝牙系统的物理层(PHY),支持在物理层和基带设备之间的 Blue RF(蓝牙射频)接口。

RF2968 和基带间有 2 个接口。串行接口提供控制数据交换的通道,双向接口提供调制解调、定时和芯片功率控制信号的通道。基带控制器与 RF2968 接口如图 1.18－2 所示。

控制数据通过 DBUS 串行接口协议的方式在 RF2968 和基带控制器之间交换。BDCLK、BDDATA 和 BnDEN 都是符合串行接口的信号。基带控制器是主控设备,它启动所有到 RF2968 寄存器存取操作。RF2968 数据寄存器可被编程,或者根据具体命令格式和地址被检索。数据包首先传送 MSB。串行数据包的格式如表 1.18－1 所列。

表 1.18－1 串行数据包格式

域	位 数	注 释
设备地址	3[A7: A5]	物理层“101”
读/写	1[R/W]	“1”为读,“0”为写
寄存器地址	5[A4: A0]	32 个寄存器的最大值
数据	16[D15: D0]	RF2968 在写模式编程,在读模式返回寄存器的内容

“写”周期,基带控制器在 BDCLK 下降沿驱动数据包的每一位,RF2968 在数据寄存器设为高状态后,在 BDCLK 第 1 个下降沿到来时被移位寄存器的内容更新,如图 1.18－3 所示。

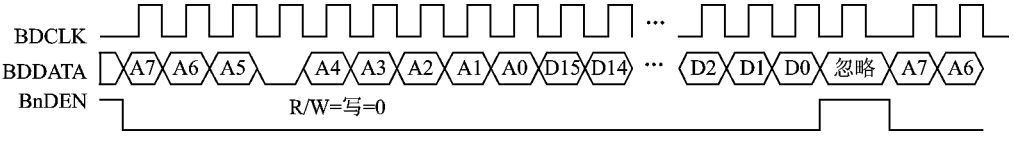


图 1.18－3 DBUS 写编程图

在读操作中,基带控制器发出设备地址、READ 位(R/W = 1)和寄存器地址给 RF2968,再

跟 1 个持续半个时钟周期的翻转位。这个翻转位允许 RF2968 在 BDCLK 的上升沿通过 BDDATA 驱动它的请求信号。数据位传输后,基带控制器驱动 BnDEN 为高电平,在第 1 个 BDCLK 脉冲的下降沿到来时重新控制 BDDATA,如图 1.18－4 所示。

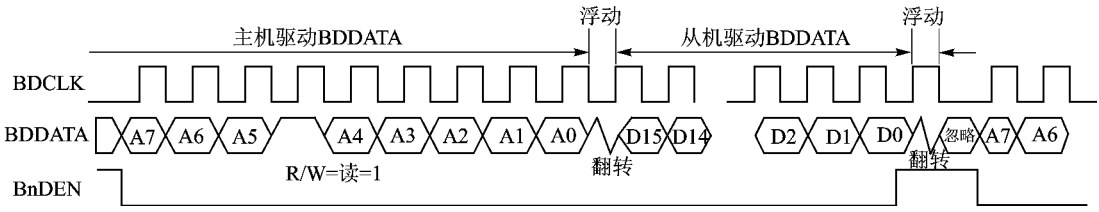


图 1.18－4 DBUS 读编程图

寄存器地址域可寻址 32 个寄存器,RF2968 仅提供 3～7 和 30、31 的寄存器地址。通过设置寄存器的数据可实现不同的功能。

双向接口完成数据交换、定时和状态机控制。所有双向同步(定时)来自 BRCLK,BRCLK 由 RF2968 产生。RF2968 使用 BRCLK 的下降沿。图 1.18－5 给出当数据从 RF2968 传给基带控制器时的通用定时。

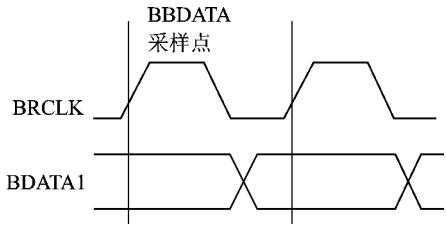


图 1.18－5 RF2968 写入到基带控制器时的通用定时

RF2968 的芯片控制电路控制芯片内其他电路的掉电和复位状态,把设备设置为所需要的发射、接收或功率节省模式。芯片的控制输入经双向接口从基带控制器(BNPWR、BXTLEN、BPKTCTL、BDATA1)输入,也可从 DBUS 模块(RXEN、TXEN)输出端的寄存器输入。基带控制器和 RF2968 内的状态机维持在控制双向数据线方向的状态。基带控制器控制 RF2968 内的状态机,并保证数据争用不会在复位和正常工作期间发生。RF2968 常用的状态有:

OFF 状态——所有电路掉电且复位,设置数据丢失。

IDLE 状态——待机模式。数据被读入到控制寄存器中,振荡器保持工作,所有其他电路掉电。

SLEEP 状态——芯片通常从 IDLE 模式进入这种模式。此时,所有电路掉电,但不复位,因此数据得以保留。电路同样可从其他模式进入 SLEEP 模式,但 TXEN 和 RXEN 状态不变,以便 TX 和 RX 电路保持导通。

TX DATA 状态——数据在这种模式发射(合成器稳定,数据信道同步)。

RX DATA 状态——接收的数据经 BDATA1(不同步)和 REDATA(和 RECCLK 同步)发送到基带电路。

RF2968 的一个典型的应用电路(GSM 电话)如图 1.18－6 所示。

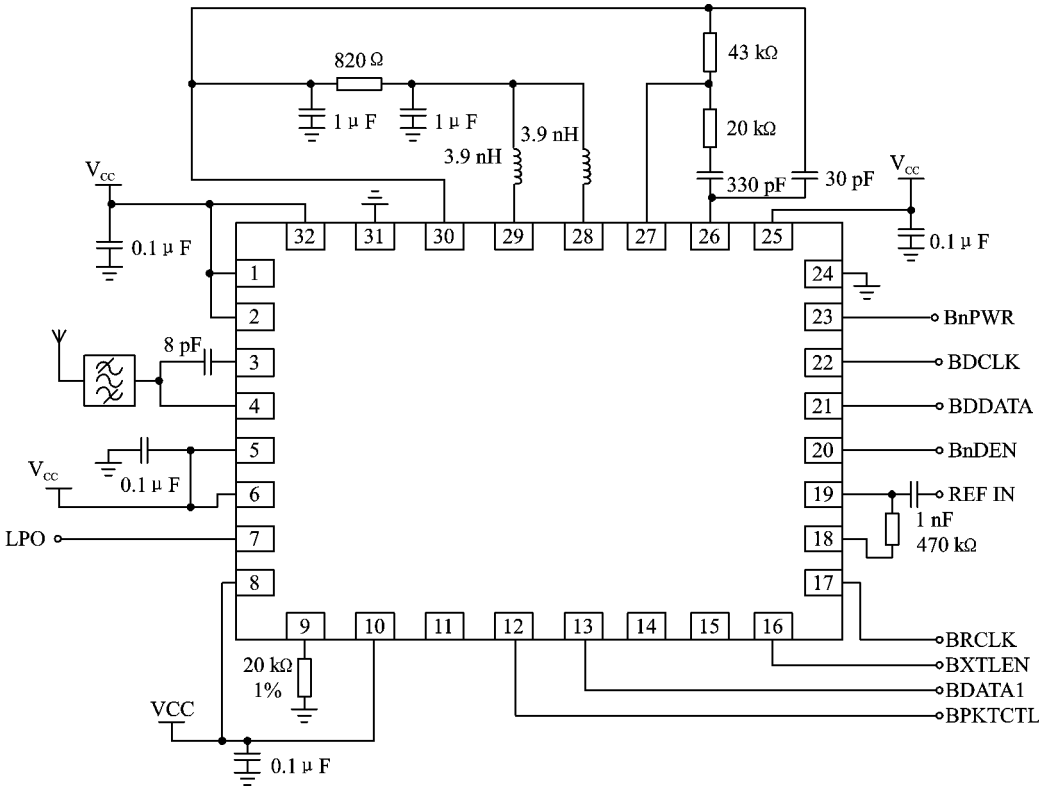


图 1. 18 - 6 RF2968 **应用电路** (GSM **电话**)

参 考 文 献

1 RF Micro Devices Inc. BLUETOOTH TRANSCEIVER RF2968. http ∶//www. rfmd. com,2002 - 01 - 30

选自《单片机与嵌入式系统应用》月刊,2002 **年第** 11 **期**

1.19 nRF™ 系列单片机无线收发器的应用设计

科汇(亚太)有限公司 韩鸣岗

一、器件特性

NORDIC 公司最新推出的 nRF™系列(nRF401/nRF403/nRF903)无线收发一体芯片,在单个芯片中集成了高频发射/接收、PLL 合成、FSK/GMSK 调制等电路,可多信道切换,直接串行数字接口,且无须曼彻斯特编码,是目前集成度最高的无线数传产品之一。其外围元件少,功耗低,便于设计生产。可广泛用于遥控遥测、无线抄表系统、交通管理系统、自动报警安防监控系统、汽车防盗监控、无线键盘/鼠标/游戏杆/PDA、小区/家庭/办公室/工厂无线网络、遥控玩具/机器人等。nRF401/403/903 的特性如表 1.19-1 所列。

表 1.19-1 nRF401/403/903 特性

器 件	nRF401	nRF403	nRF903
频率带宽/MHz	433/434	315/433	433~928
调制方式	FSK	FSK	GMSK/GFSK
频移/kHz	±15	±15	±19.2
信道数量	2	2	170
外围元件数量	15+天线匹配网络	15+天线匹配网络	15+天线匹配网络
工作电压/V	2.7~5.25	2.7~3.6	2.7~3.3
待机(掉电)功耗/μA	8	8	200/(1)
接收功耗/mA	11	11	18.5
发射功耗/mA @ dBm	8@-10,26@10	8@-10,26@10	16@-8,31.5@10
最大发射功率/dBm @ 50Ω	10	10	10
最高数据波特率/kbps	20	20	76.8
灵敏度/dBm@误码率	-105 @ BER 10^{-3}	-105 @ BER 10^{-3}	-100 @ BER 10^{-3}
工作环境温度/	-25~85	-25~85	-40~85
封 装	20 SSOIC	20 SSOIC	32 TQFP

二、应用设计评估

在开始系统设计前应对整体方案进行以下几点初步评估。

1. 电磁兼容性

使用无线产品要遵守所在国家的无线电管理规定(如欧洲 ESTI、美国 FCC),对频段、发射功率、占空比、信道带宽等指标会有限制。nRF™系列无线收发器使用 ISM(Industrial, Scientif-

ic,Medical) 国际通用频段,发射功率不大于 10 mW,故可在绝大多数国家使用。但设计者可能要求在 nRF 器件输出级放大,则需考虑是否符合当地规定。

2. 设计收发距离

RF 无线链路在可视范围的自由空间中传播的能量损耗 (FSL),由下式计算

$$FSL\ (dB) = 20\log\left(\frac{\lambda}{4\cdot\pi\cdot R}\right) \tag{1}$$

式中 λ——RF 信号的波长;

R——接收端到发射端的距离。

如图 1.19-1 所示,RF 信号在可视范围内传播的功率分布等式为:

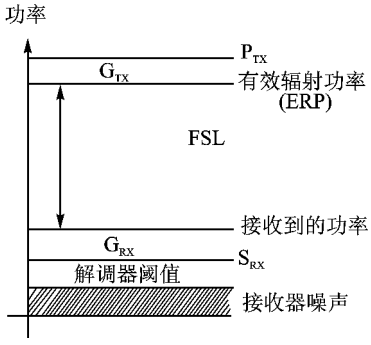


图 1.19-1 RF 传播功率分布

$$P_{TX} + G_{TX} + FSL + G_{RX} = S_{RX} \tag{2}$$

式中 S_{RX}——接收器灵敏度 (dBm);

G_{RX}——接收器 (RX) 天线增益 (dB);

G_{TX}——发送器 (TX) 天线增益 (dB);

P_{TX}——发送器 RF 输出功率 (dBm)。

以 433.93 MHz 的 nRF401 系统为例,其波长 λ = 0.69 m,又设

S_{RX} = - 105 dBm

P_{TX} = 10 dBm

G_A = 发射/接收天线总增益 $G_{TX} + G_{RX}$ = - 26.3 dB

由公式 (1) 和公式 (2) 可计算出理论传输距离 R≈1 500 m

但在实际应用中还应考虑两端天线阻抗匹配电路的损耗,传输路径中由于障碍物、多径传输引起的损耗,以及 PCB 板材质量、产品外壳和人体等的影响。其中的一些因素与应用环境密切相关。一般经验,在室外空旷地带传送距离约为理论计算距离的 1/2,室内传送距离约为理论距离的 1/10。

由图 1.19-2 可看出 RF 频率每增加一倍,损耗将增加约 6 dB,即引起传送距离下降一倍。所以如果设计者要求的主要指标是传送距离的话,可以选择较低的频率。采用中继传输也是一种有效增加通信距离的方法。

3. 其他因素

由于无线传输容易产生干扰也易被其他噪声干扰,故设计者还必须认真考虑应用场合及安装位置,不要一味追求远距离。对于电池供电,可采用占空工作方式省电。

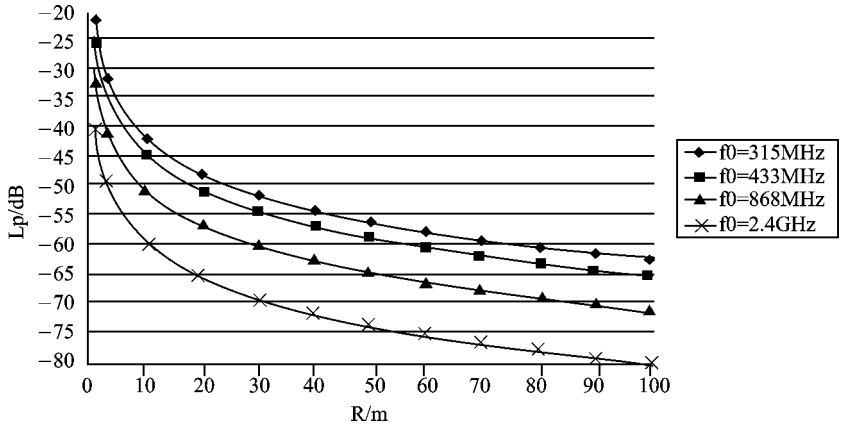


图 1.19－2 自由空间传播损耗–传播距离– RF 频率

三、外围元件的选择

nRF™收发器的外围包括 RF 前端(即天线匹配网络)及锁相环 PLL 电路(包括晶体振荡器、VCO 电感和环路滤波电路)。外围元件是实现 nRF™器件功能的关键要素。这些元件的品质和精度直接影响器件的整体性能。

1. 晶 体

RF 器件通常比典型的微控制器对晶体的要求更高。晶体必须具备:

- f_0 :晶体振荡频率;
- C0 :晶体的并行等效电容;
- ESR :晶体等效串联阻抗;
- C1 :晶体负载电容;
- 室温下绝对频率精度容差;
- 稳定度 :频率的温度稳定性。

对于 nRF 品件,晶体总的容差(绝对容差和稳定度的和)将直接影响器件之间的通信效果。如果发送端和接收端器件的频率偏差很大,接收器将会把所发送的信号当作噪声滤掉。图 1.19－3 所示为晶体引起的频率误差与信号辐射强度的关系示意图,当频率误差超过 35×10^{-6} 时,信号强度会迅速下降。

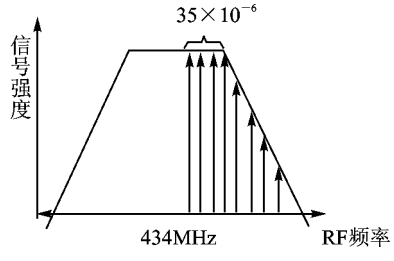


图 1.19－3 频率误差对信号强度的影响

为节省成本和 PCB 面积,通常可考虑多个器件共用晶体。如 1.19－4 所示,nRF40x 和微控制器共用同一个晶体。

2. VCO 电感

nRF™器件的片内压控振荡器(VCO)产生器件工作的所有 RF 频率,因此是一个非常关键

的部分。VCO 的额定工作频率由公式 $f = \frac{1}{\sqrt{LC}}$ 确定。

由于 VCO 的片内部分将 VCO1 和 VCO2 引脚之处的整个电路视作它的 VCO 电感，因此，LC 电路将包括以下几个部分：

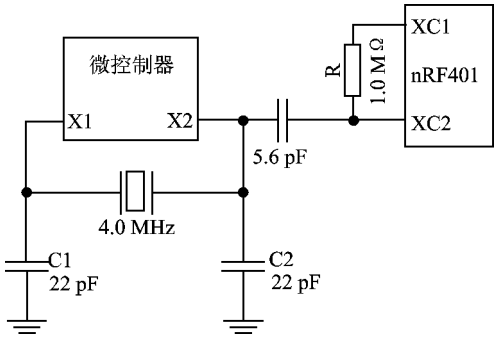


图 1.19－4 nRF401 与微控制器共用基准晶体

- 片内电路；
- 外部 VCO 电感；
- 连接芯片和 VCO 电感的线路电感；
- VCO 电感到地线层线路的寄生电容。

可见电感的取值和放置位置是相当重要的。VCO 电感的品质因数 Q 值对 VCO 内部的相位噪声和电压摆幅也起着决定作用，如果 Q 值太低，VCO 甚至不能起振。对于 nRF™收发器，要求 Q＞40～45，电感量的精度应控制在 2% 之内。

3. 环路滤波电路

环路滤波电路用以保证提供给 VCO 一个稳定的控制电压。因此，屏蔽外部噪声是十分重要的，尤其是低频（＜50 kHz）高幅值数字信号对滤波器的干扰如果不能被有效滤除，将直接影响 VCO 的稳定工作。因此，不要在环路滤波器的附近布置数字线路，而是在其周围布置地线平面加以屏蔽，使数字信号线尽量远离。同时，为获得良好的环路滤波性能，连接到地的滤波元件应直接连接到地线层，而不要通过走线或公共过孔。环路滤波器的电压应在各器件数据手册所述的极限指标范围内，nRF401/403 应在引脚 4、而 nRF903 应在引脚 3 上测量到电压为 1.1±0.2 V

4. 天 线

当器件处于接收模式时，引脚 ANT1 和 ANT2 提供 LNA（低噪声放大器）的 RF 输入，而在发送模式时，提供 PA（功率放大器）的 RF 输出。天线的连接可采用差分平衡方式。建议天线端口的负载阻抗取为 400 Ω，比起标准的 50 Ω，增大阻抗可以提高发射功率/耗用电流的比率。图 1.19－5 和图 1.19－6 分别为采用在板差分环形天线的 nRF401/403 和 nRF903 的典型应用电路。在板天线直接用 PCB 线路作天线，成本低，方向性较好，对人体不敏感，但增益较低。环形天线通常应用于相对较窄的带宽，有助于抑制强的不需要的信号以免干扰接收器。对于 50 Ω 的单端天线或测试仪器可以用如图 1.19－7 和图 1.19－8 所示的差分到单端匹配网络进行连接。

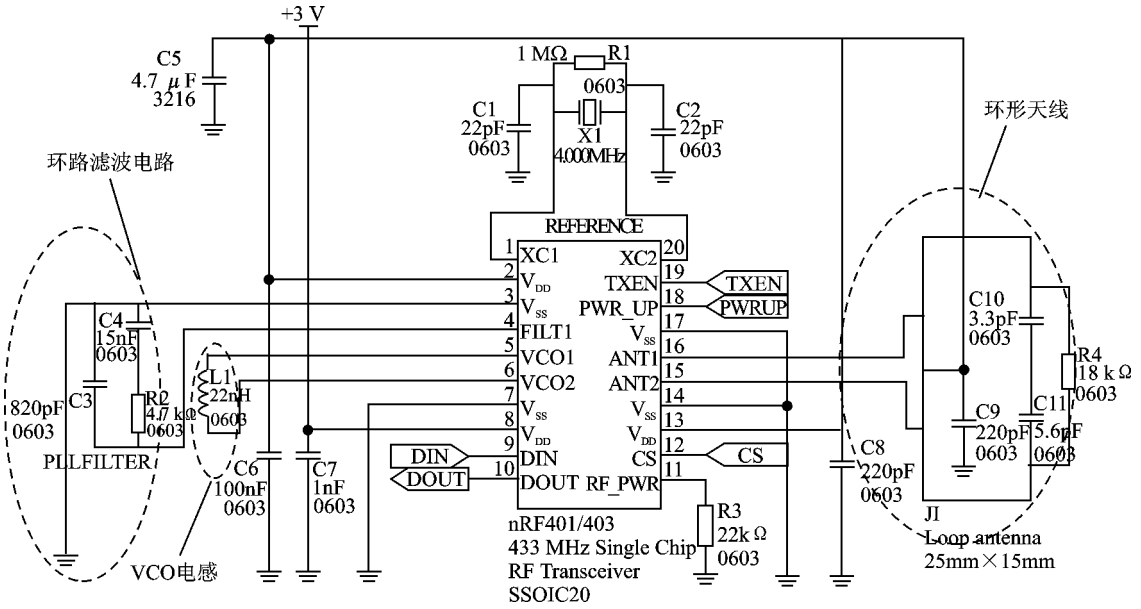


图 1.19－5 nRF401/403 应用电路

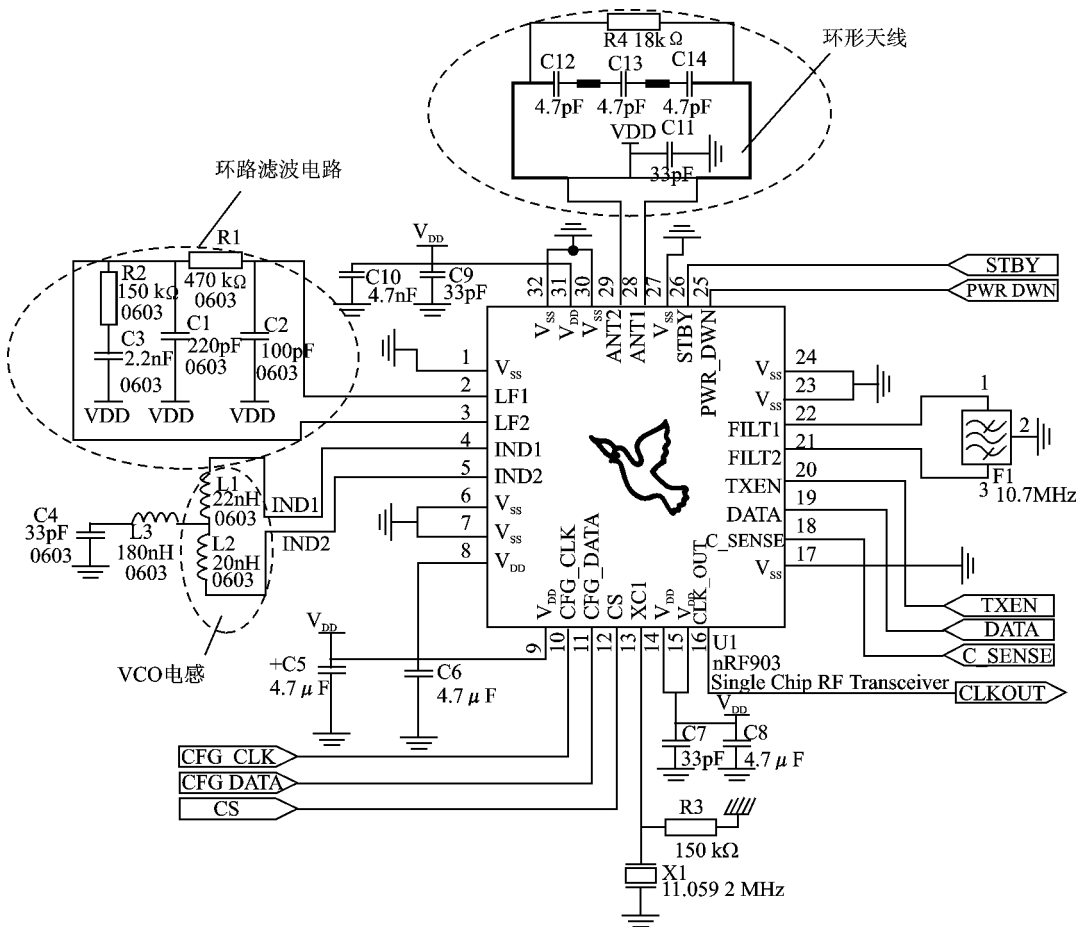


图 1.19－6 nRF903 应用电路

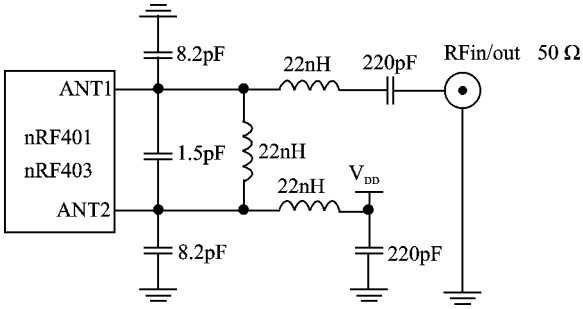


图 1. 19－7 nRF401/403 **单端匹配网络** (@433MHz)

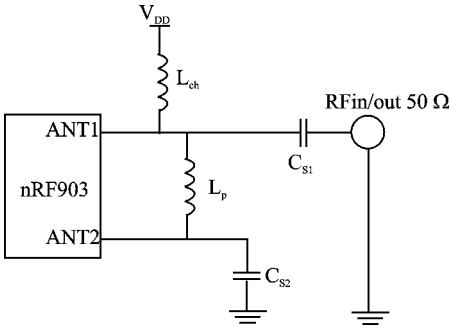


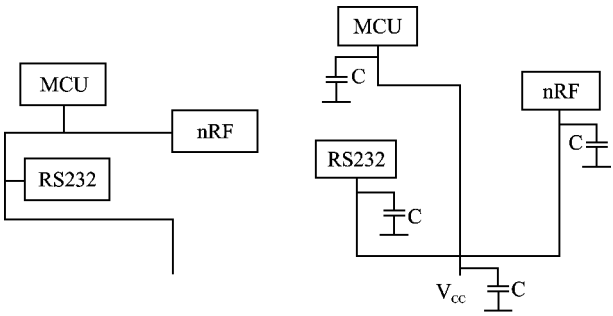
图 1. 19－8 nRF903 **单端匹配网络**

四、硬件设计要点

即使模拟 (射频) 和数字 (微控制器) 电路各自工作正常,但将两者放在同一块电路板上却有可能使问题变得复杂。因此,各电路模块在板上的布局十分重要。同时,电源和地线的排布也是至关重要的。

1. **电源噪声**

射频电路对于电源噪声相当敏感,尤其是对毛刺电压和高频谐波。因此在包含 RF **电路的** PCB **板上,电源布线必须比在普通数字电路板上布线更加仔细。如图** 1. 19－9 (b) **星形布线使数字部分和** RF **部分有各自的电源线路,并且应在靠近集成电路电源引脚处分别去耦,这是一个隔开来自数字部分和来自** RF **部分电源噪声的有效方法。**



(a) 不正确布线 (b) 星形布线

图 1. 19－9 **布线示意图**

2. 地线布局

由于在 RF 频段,即使一根很短的导线也会如电感一样,精略估算,每毫米长度导线的电感量约为 1 nH。如果不采用地线层,大多数地线将会较长,电路的设计特性将无法保证。因此,设计有 RF 元件的 PCB 时应该采用一个可靠的地线层,使所有的器件容易去耦。在表面贴装的 PCB 上,所有信号走线可在元件安装面的同一面,地线层则在其反面。理想的地线层应覆盖整个 PCB (但要注意天线方向)。如果采用两层以上的 PCB,地线层应放置在邻近信号层的层上 (如元件面的下一层)。另外最好将信号布线层的空余部分也用地线平面填充。这些地线平面必须通过很多过孔与主地线层面连接。所有对地线层的连接必须尽量短。接地过孔应放置在 (或非常接近) 元件的焊盘处。决不要让两个地信号共用一个接地过孔,这可能导致由于过孔连接阻抗在两个焊盘之间产生串扰。

综上所述,nRF™系列单片无线收发器件为无线数据传输的应用提供了理想的选择。在设计中遵循上述设计要点,结合具体应用,可获得最佳的通信效果。

选自《无线电工程》月刊,2002 年第 8 期

1.20 基于蓝牙技术的家庭网络

南昌华东交通大学 (330013) 张利华
北京航空航天大学 (100083) 张其善

一、概 述

家居的安全、舒适、方便,一直是人们追求的目标。随着计算机技术、通信技术、网络技术、控制技术、微电子技术、信息技术的迅速发展,国际互联网、个人电脑和以网络家电为代表的各种数字化家用设备逐渐进入家庭,家庭的内部联网和外部接入成为人们日益关注的问题。家庭网络是通过建立统一的高速网络结构和平台,以灵活的接入方式来实现高可靠性、低成本、灵活性强的音频、视频及远程教育软件、交互控制等信息的传递,为家庭住户提供各种信息服务。家庭网络需达到以下几方面要求。

⑴ 兼容性强。使用通用的接口模块,通用式的遥控器,有充分扩展能力。

⑵ 可操作性好。组网灵活方便,安装方式简单,使用简便,固定和移动相结合,不需要专门人员管理,不需要进行专门的培训。

⑶ 功能强大、短距离。采用嵌入式操作系统,实现远程控制和监测,将家庭内的电器设备、照明灯光、安全防护、环境监测等设备连成一体,通过统一的网络总线式结构和控制平台,实现对这些设备集中监控管理,一般距离不会超过 100 m。

⑷ 价格低廉。特别是在现有家居基础上构建的,不能对家居结构、装修和环境美观造成破坏。

⑸ 高可靠性和安全性。

现有的一些短距离连接技术,如 802.11b、HomeRF、HomePNA、IrDA 技术等,由于各自的局限性不能很好地满足要求。蓝牙技术是一种开放的短距离、低成本的无线连接技术,能够在嵌入蓝牙模块的设备之间自动联络、确认,并进行通信,能够满足构建家庭网络的要求。

二、蓝牙技术

蓝牙 (Bluetooth) 技术是由特殊利益集团 (SIG) 发起的一种短距离 (10 cm ~ 10 m,增加放大单元后可达 100 m 以上) 无线通信连接技术。它以低成本的近距离无线连接为基础,为固定与移动设备通信环境建立一个特别短程无线连接。设计初衷就是将移动电话与笔记本电脑、掌上电脑以及各种数字化的外部设备用无线方式连接起来,进而形成一种个人周围领域的网络,使得在其可达到的范围之内,各种信息化的移动便携设备都能实现无缝地共享资源。实质上是一种替代电缆的技术,通过建立通用型空中无线信号传输接口及其操控软件的国际公开标准,使不同厂家生产的便携式设备在没有电线或电缆相互连接的情况下,能在近距离范围内具有相同操作的性能。

1. 蓝牙网络

蓝牙支持点对点和点对多点通信,有两种网络拓扑结构:微微网和分布式网络。

(1) 微微网(Piconet)

蓝牙最基本的网络组成是微微网。微微网由主设备单元(Master unit)和从设备单元(Slave unit)两部分组成。Master 是指处于主地位的蓝牙设备单元,负责提供时钟同步信号和跳频序列;Slave 是指处于从地位的蓝牙设备单元,它受控同步于主设备。结构如图 1.20-1 和图 1.20-2 所示。

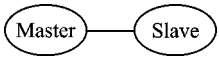


图 1.20-1 单 Slave

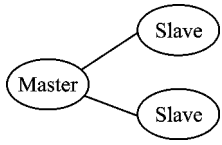


图 1.20-2 多 Slave

一个微微网可包括 8 个蓝牙设备:1 个 Master 和最多 7 个 Slave。在一个微微网中,所有的用户均用一个跳频序列,微微网之间由不同的跳频序列来区分,不同的信道有不同的主设备,从而有不同的跳频序列和相位。其中,主设备的 ID(48 bit 设备地址码 BD_ADDR)决定跳频序列,系统的时钟决定跳频序列的相位;对于从设备,系统的时钟加上一个偏移和主设备的时钟同步。

(2) 分散网(Scatternet)

蓝牙允许多个微微网共存一个区域,在一个区域的多个 Piconet 组成分散网。结构如图 1.20-3 所示。

所有的 Piconet 共享 80 MHz 带宽,每一个 Piconet 用 1 MHz 带宽。蓝牙采用在时隙上连接的基于数据包的传输方式,一个设备可以加入多个 Piconet,但在某一具体时刻只能加入一个 Piconet。而且,蓝牙设备跳到另一个 Piconet 时可以改变身份,即主设备变成从设备。

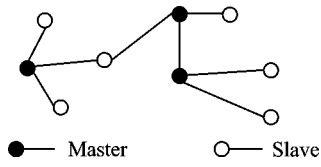


图 1.20-3 分散网

2. 蓝牙系统特征及主要参数

(1) 使用 2.4 GHz ISM 频带。采用跳频扩谱技术,每秒 1 600 跳,在建立连接的时候为 3 200 跳;1 mW 发射功率,通信距离为 10 cm~10 m,增加发射功率可达 100 m,典型的家庭范围。

(2) 传输特性。1 M 带宽,有效通信速率为 721 kbps,支持 64 kbps 的实时语音传输,语音编码采用 PCM 和 CVSD;仅传输语音时,支持三路全双工;当语音和数据同时传输或者仅传输数据时,支持 433.9 kbps 对称全双工或 723.2/57.6 kbps 的非对称双工通信。

(3) 采用时分双工(TDD),FSK 调制,利用 CRC、FEC 及 ARQ,保证通信的可靠性。

(4) 惟一的 48bit BDADDR.。

(5) 通信协议采用分层结构,支持语音/数据访问点、外设连接、个人网络(PAN)等三大范畴的应用。

(6) 安全机制。在通信链路采用跳频技术,应用层采用 PIN(个人标识码)进行单双向认证,链路层采用认证、加密和密钥管理来进行安全控制。

(7) 连接类型和数据包。支持两种类型的连接,异步无连接类型(传输数据包)ACL 和同步面向连接类型(传送语音)SCO 采用基于数据包的格式如图 1.20-4 所示。

72bit 接入码	54bit 包头	0~2745 数据

图 1.20-4 数据包格式

三、蓝牙家庭网络

正是由于蓝牙技术的低成本、低功耗、低复杂性、高速率、高可靠性、强互操作性和兼容性等特点,使得基于蓝牙技术的家庭网络能为人们所接受。蓝牙家庭网络是网络家电和其他设备通过嵌入蓝牙模块(如 USB 卡、UART 卡、PCM 卡),利用无线方式连在一起,使之相互通信;同时利用具有路由功能的蓝牙无线接入点 BLAP 和外部网络相连,构成家庭式网络系统或家庭局域网,提供集中的或异地的音频、视频通信、计算机控制和管理等,使信息在家庭内以及与外部之间充分流通和共享。

1. 蓝牙家庭网络功能

蓝牙家庭网络主要有以下功能。

(1) 安全服务。紧急情况报警、医疗求助信息传送、家庭情况监控和安全对讲等。

(2) 能源管理。家庭水、电、气和供暖定时开关等。

(3) 家庭环境控制。自动或远程控制家庭的温度、湿度、光照和空气流通等。

(4) 数据通信。电话/传真通信,互联网接入,远程教育,远程医疗,视频传送,点播、分配和交互通信等。

(5) 智能控制。信息家电智能化、家电设备的进程和网络控制、三表自动计量传输和家庭远程控制等。

2. 蓝牙家庭网络结构

根据外部接入网的不同,蓝牙家庭网络的拓扑结构有以下两种形式。

(1) 直接接入方式 家庭蓝牙设备(BD,Bluetooh device)组成多个 Piconet,多个 Piconet 组成户内网络(Scatternet),然后通过蓝牙家庭网关(BLAP)直接与公共服务网络相接。

(2) 间接接入方式 户内 Scatternet 通过 BLAP 与智能小区局域网相接,然后接入公共服务网络,此时 Scatternet 充当智能小区局域网的子网。

3. 蓝牙家庭网关

蓝牙家庭网络的主要组成包括:蓝牙家庭网关(蓝牙无线接入点 BLAP)、嵌入蓝牙模块的终端设备、提供多样化输入输出的蓝牙外围设备和软件系统。其中,蓝牙家庭网关是基于蓝牙的 LAN 访问协议,是整个家庭网络的核心,用于连接家庭 Scatternet 和公共网络。家庭 Scatternet 通过 BLAP 将公共网络的功能和应用延伸和扩展到家庭。BLAP 一端通过网口与公共网络相连,另一端通过蓝牙与家庭 Scatternet 中的蓝牙设备相连,实现两者之间的信息流通和共享。

(1) 蓝牙局域网接入的系统模型

蓝牙设备以 PPP 方式访问局域网,蓝牙局域网接入应用的系统模型如图 1.20-5 所示。

模型中用到的协议栈包括:基带(baseband)、链路管理协议(LMP)、逻辑链路控制和适应协议(L2CAP)、服务发现协议(SDP,service discovery protocol)、串口仿真协议(RFCOMM)和点对点协议(PPP)。ME 是一个管理实体(master entity),负责协调在初始化、配置和链路管理过

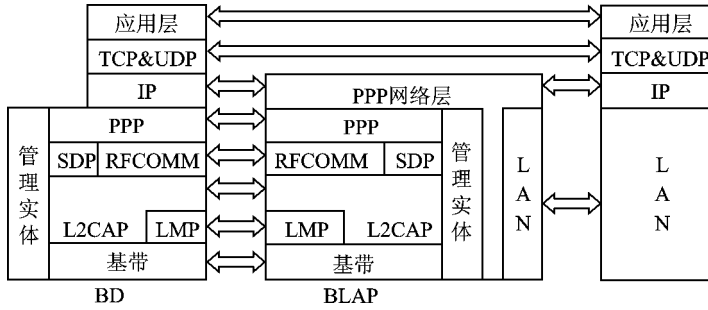


图 1.12－5 蓝牙局域网接入系统模型

程中所涉及到的一系列操作。BLAP 负责家庭 Scatternet 和外部网络间的 IP 转发。其中，①应用层：负责 BLAP 的初始化，进行参数配置；BD 通过 SDP 建立 LAN 连接，以及连接失败时通过保存以前的 BLAP、服务、Link Keys、用户名和口令，重新加快或自动建立连接；断开连接。②PPP 层：进行服务搜索；启动 RFCOMM，建立 PPP 连接；断开和关闭 PPP 连接。③RFCOMM：提供 L2CAP 之上的串行口仿真，通常用于传输 PPP 分组，提供 PPP 流量控制。④SDP：为应用程序提供发现服务并确定服务属性的机制。⑤L2CAP：向高层协议提供面向连接或无连接的数据服务，包括协议复用、打包、拆包、提供信号信道和服务质量控制等。⑥LMP：用于建立链路、安全（认证、数据加密）和控制。⑦BaseBand：负责跳频和蓝牙数据及信息帧的传输。

（2）BLAP 的硬件结构

BLAP 主要包括蓝牙部分（Bluetooth module、天线、放大模块）、MCU、语音编解码电路、接口电路和其他辅助电路（液晶显示、键盘和电源），其组成如图 1.20－6 所示。

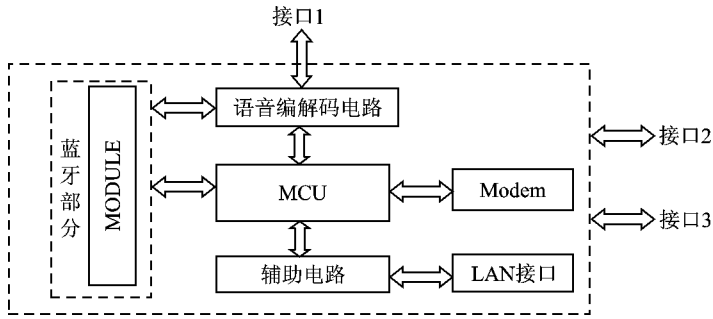


图 1.12－6 BLAP 硬件结构

接口 1 是与公共电话网的接口，接口 2 是与公共电话网和 Internet 的接口，接口 3 是与外部 LAN 的接口。系统接口还可包括 R232、USB、UART 和 IIC 等。通过这些接口，BLAP 可以和外界公共服务网络进行连接，进行数据通信；同时通过蓝牙部分可以和户内 Scatternet 建立无线连接，完成蓝牙家庭网络中各种不同的通信协议之间的互相转换和信息共享、并管理和控制网络中的设备和外设。

显然，BLAP 硬件结构实际上也是设备和外设蓝牙接口卡硬件的一般结构。

4. 蓝牙家庭网络通信过程

蓝牙家庭网络内的 BD 或 BLAP 上电后，进行初始化，同时通过“通行字鉴别”核实持有者身份，产生连接密钥用来作设备认证和数据加密。核实后，首先处于旁观模式（Standby），在无

线范围内,通过应用程序发现并选择合适的提供 PPP/RFCOMM/L2CAP 服务的其他 BD 或 BLAP,建立通信链路过程如下。

(1) 建立基带物理链路。首先通过查询形成一个完整的覆盖范围内的设备情况表,包括标识蓝牙设备跳频序列的惟一的 48bit BD_ADDR 和相位,然后通过寻呼形成 Piconet 。在查询过程中,主设备在 32 个偶数信道发送包含自身 BD_ADDR 的 Inquire 指令,同时在反向回复信道上监听回音,1.25 s 后再间隔 16 个偶数信道发送包含自身 BD_ADDR 的 Inquire 指令 ;被询设备每隔 1.25 s 就在 32 个回复信道上的一个进行扫描,并停留 10 ms,一直到被询设备扫描的信道和主询设备发布指令的信道重合,而后被询设备就会用 FHS 包发送自己的 BD_ADDR 和相位,如此反复,形成一个完整的覆盖范围内的设备情况表。寻呼过程是类似的,主设备用所需设备的 BD_ADDR 寻呼这个设备,被呼设备用自己的 BD_ADDR 响应,一直到在某信道重合,此后,主设备发一个 FHS 包 (包含 BD_ADDR 和相位) 给寻呼设备,从设备加入主设备的 Piconet。一旦某个设备加入 Piconet,它被分配一 3 bit 的 AMA (主动成员地址) ,其他设备可以用其访问该设备。此时,所有设备处于活动状态。

(2) 服务发现过程。在物理链路建立之后,主设备运行服务发现协议客户端部分,从设备运行服务发现协议服务器端部分。在主设备端,主设备发出服务发现请求 ;在从设备端,服务器程序将它的各项服务属性进行注册,并可以采用 3 种途径 (服务类型搜索、服务属性搜索、服务扫描) 从服务记录数据库当中寻找服务。服务发现协议数据单元先利用 L2CAP 的面向连接服务,然后通过基带的 ACL 发送出去。

(3) 建立逻辑链路。RFCOMM/L2CAP 连接成功后,启动 PPP 认证机制,要求用户输入用户名和密码,建立 PPP 连接,可以进行通信。

在整个过程中,认证和加密由 LMP 实现,且 BD 或 BLAP 可随时中断连接。

5. 蓝牙家庭网络的设备应用模式

蓝牙家庭网络的设备可以灵活进行连接通信,有以下几种模式。

(1) 点对点对等应用模式

家庭 Scatternet 内部外设和终端设备、终端设备之间两两单独进行单双向信息交换时使用这种模式。如图 1.20 -7 所示。

连接的双方处于活动状态 (Active) ,其他则处于旁观状态 (Standby) 。

(2) 集线器应用模式

家庭 Scatternet 内部多个终端、多个外设、多个外设和终端同时和某个终端、外设连接进行单双向信息交换时使用这种模式。如图 1.20 -8 所示。

连接的多方处于活动状态,为维持低功率状态,除传输数据的两方外,其他处于三种节能状态即停等 (park) 、保持 (hold) 和呼吸 (sniff) 状态中的一种 ;另外的设备和外设处于旁观状态。

(3) 无线蜂窝漫游应用模式

当设备或外设从一个家庭 Scatternet 移动漫游到另一个 Scatternet、本地 Scatternet 的设备或外设与异地 Scatternet 进行信息流通时使用这种模式。如图 1.20 -9 所示。

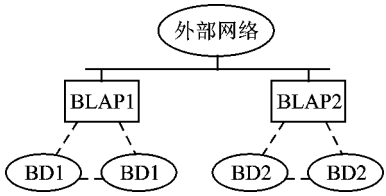


图 1.20－7 点对点对等应用模式

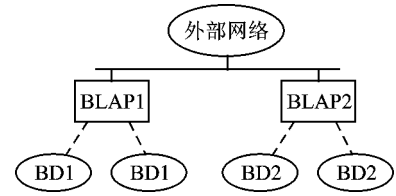


图 1.20－8 集线器应用模式

(4) 无线接入点应用模式

当设备或外设通过 BLAP 接入外部网络进行信息交换、用户通过外部网络对家庭进行监控时使用这种模式。

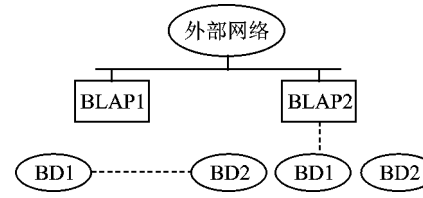


图 1.20－9 无线蜂窝漫游应用模式

用蓝牙技术构建家庭网络，具有低成本、多向性、安装简单、使用灵活、稳定安全等优点。随着蓝牙芯片价格的进一步下降，蓝牙网络家电产品和网络设备的出现，蓝牙家庭网络将实现人们家居智能化梦想。

参 考 文 献

1 Specification of the Bluetooth System，Volume 1. Version 1.1，Bluetooth SIG，Feb. 2001

2 Specification of the Bluetooth System，Volume 2. Version 1.1，Bluetooth SIG，Feb. 2001

3 肖晴. 家庭网络系统及其实现. 电子技术，2001 (4)：51～64

选自《电子技术》月刊，2002 年第 7 期

第二章

综合应用

2.1 嵌入式系统的超时控制及其应用

西安理工大学 同向前

嵌入式系统多为实时控制系统,工作流程常受到各种外部事件的同步制约,下一步工作的开展可能需要等待某些外部预期事件的发生。譬如,预定时间内的 I/O 外设的同步响应、串行通信中对方对通信联络信号的及时回应等。然而,在实际工作中,由于设备故障或外界干扰等意外情况,使得等待的预期事件没有发生,此时,测控系统将会永远处于等待状态,工作流程无法向下进行,导致生产故障。为防止此种情况的发生,应在测控程序的适当位置引入超时控制。

超时控制过去主要应用于 PC 机串行通信程序中,嵌入式系统中未见论述。实际上,超时控制的概念在嵌入式系统中有着更为广泛的应用。

一、超时控制的程序结构

超时控制实际上是设置一个定时/计数器,在进入外部事件等待之前,先设定定时值并启动超时定时/计数器。在程序循环等待的同时,超时定时器也在不断计时。在预定时间到来之前,若等待的事件如期发生了,则系统正常退出循环等待状态并进入下一个工作流程继续执行 否则,系统发生超时,并将因超时而退出循环等待状态且进入超时处理流程。有无超时控制时的程序流程结构比较如图 2.1-1 所示。

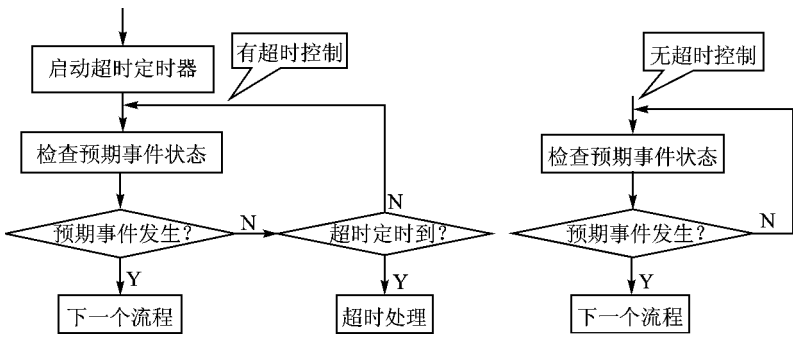


图 2.1-1 有无超时控制时的程序结构比较

二、超时控制的实现方法

1. 基于软件计数器的实现方法

由前述可知,超时控制用于循环等待结构。设循环体程序的执行时间为 T,则 N 次循环所需时间为 NT。可见,控制程序循环次数即可完成超时控制。下面给出 C 语言参考程序,其中,状态变量 Status 反映预期事件是否发生。在程序 Status_Check () 中,当检查到预期事件已发生时,置状态变量 Status 为 ON。

```
    Status = OFF;
    for (i = 0; i < 1000; i++)                    /* 超时控制 */
     {Status_Check();                             /* 预期事件状态检查 */
       if (Status = = ON) break;                  /* 预期事件发生,退出循环 */
     }
if (Status = = OFF) Overtime_error();             /* 超时处理 */
```

软件实现方法虽然简单,但超时时间不易准确控制。对于要准确控制超时时间的场合,可采用下述基于硬件定时器的实现方法。

2. 基于硬件定时器的实现方法

对于单片机系统而言,可采用单片机内部定时器作为超时定时器,定时器工作于中断方式。当定时时间到(即发生超时)时,定时器申请中断,在中断服务程序中置位超时标志。在预期事件循环等待的循环条件检验中,一方面检查预期事件是否发生,另一方面查看超时标志以判断等待是否超时。当两者之一为真时,系统退出循环等待状态。

当超时定时值超出单片机片内定时器的定时范围或系统需要多个不同定时值的超时控制时,可使超时控制用的片内定时器定时在一个小而固定的基本定时值 T(譬如 500 μs)上,通过设置不同的定时中断次数计数值来满足长超时或不同超时控制的要求。以 8051 单片机系统为例,引入超时控制的 C 语言参考程序如下。其中,采用 8051 片内定时器 T0 作为超时定时器。Status 含义同上,另设中断次数计数变量 Tnum 和超时标志 Overtime_Flag。

```
/* 超时启动子程序 */
Overtime_Start()
{ TMOD = TMOD&0xF0;
   TMOD = TMOD|0x02;                              /* 定时器 T0 工作于方式 2 */
   TL0 = 0x06;
   TH0 = 0x06;                                    /* 6 MHz 下定时时间为 500μs */
   TR0 = 1;                                       /* 启动定时器 T0 */
   ET0 = 1;                                       /* 允许定时器 T0 中断 */
   EA = 1;                                        /* 8051 开中断 */
   Overtime_Flag = OFF;                           /* 清除超时标志 */
}
/* 定时器 T0 中断服务子程序 */
void Timer0 (void) interrupt 1
{ Tnum--;                                         /* 计时(中断次数计数) */
   if (Tnum = = 0)                                /* 超时? */
      {ET0 = 0;                                   /* T0 关中断 */
       TR0 = 0;                                   /* 关闭定时器 T0 */
       Overtime_Flag = ON;                        /* 置位超时标志 */
      }
/* 超时控制程序段 */
Tnum = 2000;                                      /* 超时时间设定,1 s */
Overtime_Start();                                 /* 启动超时定时器 */
Status = OFF;
```

```
do {Status_Check () ;                          /* 预期事件状态检查 */
    } while ((Status = = OFF) && (Overtime_Flag = = OFF)) ;
if (Status = = OFF) Overtime_error () ;        /* 超时处理 */
…
}
```

三、超时控制的应用

超时控制在嵌入式系统设计中具有广泛的用途,许多与时间有关的操作流程均可借助超时控制方法来实现,如预期事件的延时等待、设备自检、延时和定时控制等。下面举例说明超时控制方法的应用(有关变量和函数的定义请见前述)。

1. 串口通信

在串口通信中,当单片机根据约定要从串口接收一个字符时,常处于等待接收状态。退出等待的条件应是接收有效或超时,或者说,继续循环的条件是接收无效且未超时。设本机发出请求命令后 1 s 内接收不到对方的回应信号,则认为通信链路出错。以 8051 单片机为例,采用基于硬件定时器的超时控制方法,C 语言参考程序如下。

```
Tnum = 2000 ;                                  /* 超时时间设定 1s */
Overtime_Start () ;                            /* 启动超时定时器 */
do {Status = SCON&0x01 ;                       /* 串口接收状态检查 */
    } while ((Status ! = 0x01) && (Overtime_Flag = = OFF)) ;
if (Status ! = 0x01) Overtime_error () ;       /* 超时处理 */
```

2. RAM 自检

设某单片机系统的 8000H ~ 9FFFH 存储器空间为 RAM 区,系统运行之初先对 RAM 进行自检,自检通过则继续向下,否则进入故障状态 error () 。自检方法为逐个字节写入 55H,再读出进行比较,二者一致为正常,否则重新读写一次。若连续 5 次读写出错,则认为 RAM 存在故障。显然,这是一种基于软件计数器的超时控制,C 语言参考程序如下。

```
ram_ptr = 0x8000 ;
for (n = 0 ;n < 0x2000 ;n + + ) {
    for (i = 0 ;i < 5 ;i + + ) {               /* 读写次数控制 */
        * ram_ptr = 0x55 ;
        value = * ram_ptr ;
        if (value = = 0x55) break ;            /* 读写正确,退出循环 */
    }
    if (value ! = 0x55) error () ;             /* 故障处理 */
    ram_ptr + + ;
}
```

3. 液晶显示器的背光控制

许多液晶显示器设有背光电路。当环境光线昏暗时,打开背光以利观察和操作,待操作结束后再关闭背光以节约电量和延长寿命。然而,操作者往往观察时打开背光而离去时忘记关闭背光,为此,可以通过超时控制来自动关闭背光。超时控制的规则是:每当键盘在一定时间

内没有人按动时,认为操作者已经离去,自动关闭背光。

```
Tnum = 120000;                          /* 定时时间设定,1min */
Overtime_Start();                       /* 启动超时定时器 */
Overtime_Flag = OFF;
for(;;){                                /* 运行主循环 */
…
Key_Detect();                           /* 键盘检测,若无键按下则置键值变量 Key 为 0FFH */
if(key != 0xFF)                         /* 每当有键按下,超时控制重新启动 */
    {Tnum = 120000;
    Overtime_Start();
    Overtime_Flag = OFF;
    }
if(Overtime_Flag = ON) Close_Light();   /* 超时,自动关闭背光 */
…
}
```

4. 超时控制与延时的统一

```
Tnum = 10;                              /* 延时时间设定,5ms */
Overtime_Start();                       /* 启动定时器 */
while(Overtime_Flag != ON);             /* 延时等待 */
```

四、结束语

在微机测控软件中合理运用超时控制策略,不仅可以提高测控软件的容错性能,还可以以简洁的方式实现各种与时间有关的程序流程控制。本文论述了嵌入式系统中超时控制的程序结构及其应用,并给出了便于阅读理解的单片机 C 语言参考程序。当然,也可根据需要用汇编语言来编写超时控制程序。

参 考 文 献

1 马忠梅,籍顺心,张凯,等. 单片机的 C 语言应用程序设计. 修订版. 北京 北京航空航天大学出版社,1999

选自《单片机与嵌入式系统应用》月刊,2002 年第 7 期

2.2 多路读写的 SDRAM 接口设计

杭州浙江大学信息与通信研究所 (310027)
赵丕凤 徐元欣 赵 亮 李式巨

存储器是大容量数据处理电路的重要组成部分。随着数据处理技术的进一步发展,对于存储器的容量和性能提出了越来越高的要求。同步动态随机存储器 SDRAM (Synchronous Dynamic Random Access Memory) 因其容量大、读写速度快、支持突发式读写及相对低廉的价格而得到了广泛的应用。SDRAM 的控制比较复杂,其接口电路设计是关键。

本文首先介绍 SDRAM 的主要控制信号和基本命令,然后介绍接口电路对 SDRAM 的主要操作路径及操作过程,应用于解复用的 SDRAM 接口电路的设计方法,最后给出了实现结果。

一、SDRAM 的主要控制信号和基本命令

SDRAM 的主要控制信号为:

- $\overline{CS}$:片选使能信号,低电平有效;
- $\overline{RAS}$:行地址选通信号,低电平有效;
- $\overline{CAS}$:列地址选通信号,低电平有效;
- $\overline{WE}$:写使能信号,低电平有效。

SDRAM 的基本命令及主要控制信号见表 2.2-1 所列。

表 2.2-1 SDRAM 基本操作及控制信号

命令名称	$\overline{CS}$	$\overline{RAS}$	$\overline{CAS}$	$\overline{WE}$
命令禁止 (NOP:Command inhibit)	H	X	X	X
空操作 (NOP:No operation)	L	H	H	H
激活操作 (ACT:Select bank and active row)	L	L	H	H
读操作 (READ:Select bank and column, and start READ burst)	L	H	L	H
写操作 (WRITE:Select bank and column, and start WRITE burst)	L	H	L	L
突发操作停止 (BTR:Burst terminate)	L	H	H	L
预充电 (PRE:Deactive row in bank or banks)	L	L	H	L
自动刷新或自我刷新 (REF:Auto refresh or self refresh)	L	L	L	H
配置模式寄存器 (LMR:Load mode register)	L	L	L	L

所有的操作控制信号、输入输出数据都与外部时钟同步。

二、接口电路对 SDRAM 的主要操作路径及操作过程

一个完备的 SDRAM 接口很复杂。由于本文的 SDRAM 接口应用于解复用，处理的事件相对来说比较简单，因而可以简化设计而不影响性能。接口电路对 SDRAM 的主要操作可分为：初始化操作、读操作、写操作、自动刷新操作。

1. 初始化操作

SDRAM 上电一段时间后，经过初始化操作才可以进入正常工作过程。初始化主要完成预充电、自动刷新和模式寄存器的配置。操作过程如图 2.2－1 所示。

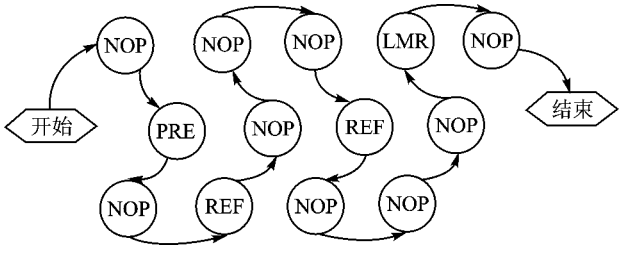


图 2.2－1 初始化操作过程

2. 读写操作

读写操作主要完成与 SDRAM 的数据交换。读操作过程如图 2.2－2 所示，写操作过程如图 2.2－3 所示。

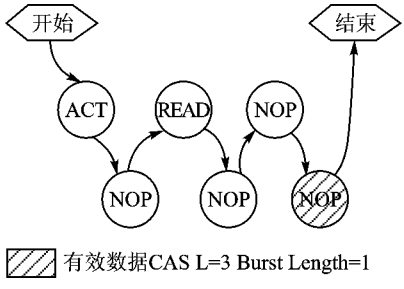


图 2.2－2 读操作过程

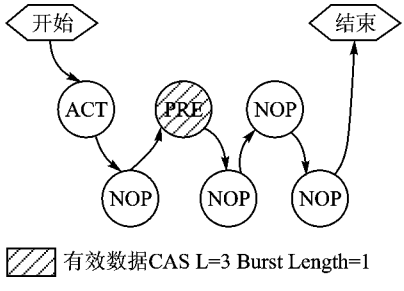


图 2.2－3 写操作过程

3. 刷新操作

动态存储器(Dynamic RAM)都存在刷新问题。这里主要采用自动刷新方式,每隔一段时间向 SDRAM 发一条刷新命令。刷新过程如图 2.2-4 所示。

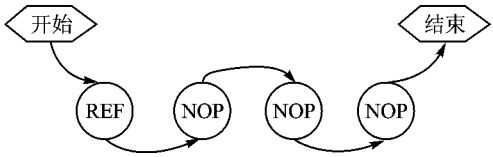


图 2.2-4 刷新操作过程

三、接口电路的设计

1. 解复用电路

本解复用电路主要完成将 1 路高速数据流解复用为 4 路数据流,其结构框图如图 2.2-5 所示。1 路数据流进入解复用器后,经过 SDRAM 缓冲,解复用为 4 路数据流。

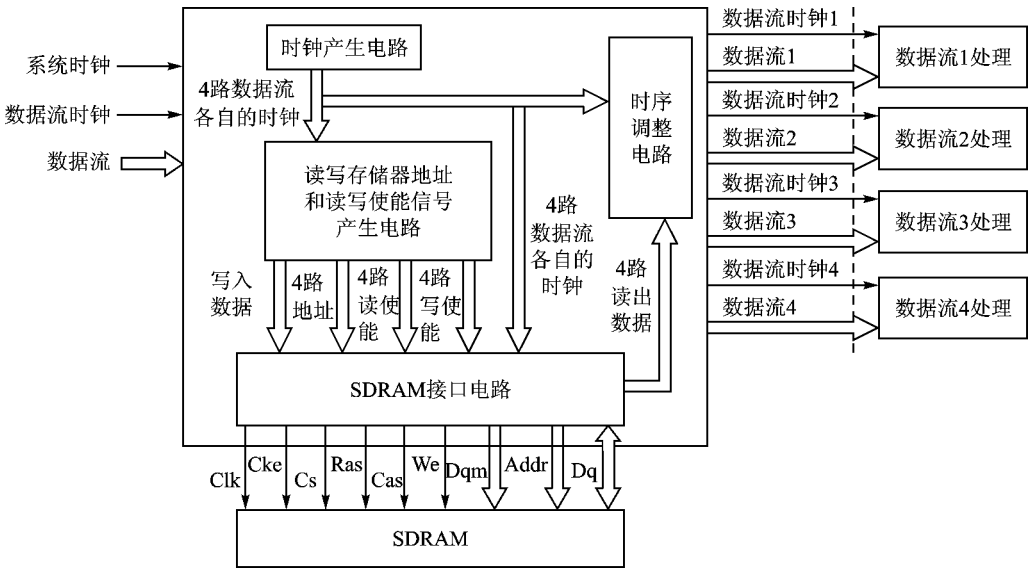


图 2.2-5 解复用电路结构图

由于要解复用为 4 路数据流,为了充分利用时隙,满足高速的要求,采用 4 个 bank 的 SDRAM,各路数据缓冲对应不同的 bank。为简化设计,数据流 1 的缓冲区定为 bank0,数据流 2 的缓冲区定为 bank 1,数据流 3 的缓冲区定为 bank 2,数据流 4 的缓冲区定为 bank 3。对于每路数据实际上是以高速率集中写入,然后以低速率均匀读出。

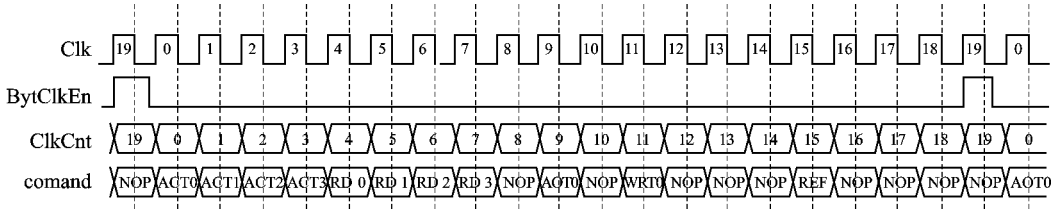
由于进行的是解复用,因此写入的数据只有 1 路,但是有可能 4 路数据同时都要读出。所以对于 4 路数据流,其读写地址和读写使能信号是分开的。

2. SDRAM 接口电路的时序控制

高速数据流的速率为 3 MB/s,采用的系统时钟为 20 倍的字节时钟。送入 SDRAM 的时钟为 60 MHz 系统时钟。在一个字节时钟内对 SDRAM 的操作最多有 5 次(1 次读,4 次写),而且

为了满足刷新的要求,每个字节时钟进行一次刷新操作。根据 SDRAM 的时序要求,这样的操作是难以实现的。因而要通过多 bank 操作,尽量做到时分复用来实现。图 2.2－6 给出了在一个字节时钟周期内数据流 1 进行读写操作,其他 3 路数据进行读操作的命令排序时序图。可以看出通过多 bank 操作,时分复用,在 20 个系统时钟节拍内所需的读写操作命令刚好很紧凑地排开。

一个字节时钟内对 SDRAM 读写操作是随机的,这与数据流的复用比例有关。为了满足时序,根据上面的说明,需要把一个字节时钟周期内对 SDRAM 的命令合理排序,然后按照排好的顺序执行命令。这样就需要把一个字节时钟周期内对 SDRAM 的操作进行缓存,然后在下一个字节时钟周期内进行排序、与 SDRAM 命令相对应、将命令译码产生相应的控制信号线,完成操作。缓存排序过程如图 2.2－7 所示。



注:1. Clk 为送给 SDRAM 的时钟;BytClkEn 为字节时钟使能;
ClkCnt 根据 Clk 的时钟节拍计数;command 为要对 SDRAM 执行的命令。
2. 读写命令都是带 auto－precharge 的;CAS L = 1,Burst Length = 1。
3. ACT 0 表示对 bank 0 进行激活,其他操作类推。

图 2.2－6 SDRAM 命令排序时序图

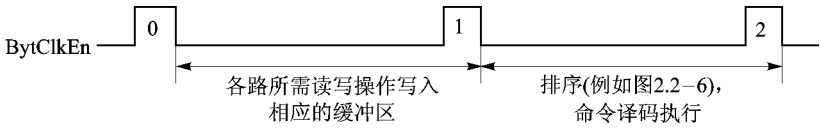


图 2.2－7 缓存排序流水时序图

注:命令排序的结果如图 2.2－6 所示。

3. SDRAM 接口电路

SDRAM 接口电路中需要专门操作缓冲区存储一个字节时钟周期内的操作,以备下一字节时钟的排序。为了方便处理,对每路数据的缓冲操作内容(或读或写)放在一个缓冲区。由于数据流的连续性,排序的同时仍然会有操作要求,因此每路的操作内容缓冲区分为两块。对一块缓冲区写入时,读出另一块缓冲区中的操作内容,进行排序、译码、执行。根据字节时钟切换对缓冲区的读写,从而避免冲突。对于从 SDRAM 读出的数据,每路数据写入相应的读出数据缓冲区。同样每路的读出数据缓冲区也分为两块,根据字节时钟切换读写。

由于一个字节时钟周期内,每路所需的操作最多有 2 次,每路的操作内容缓冲区只需两个单元(每个单元存储了此次的读写使能信号、写入数据、地址)即可。对于读出数据缓冲区,由于一个字节时钟每路数据最多执行一次读操作,所以读出数据缓冲区只需要一个字节。这两类缓冲区容量都小,因此全部用寄存器来实现,控制简单。

整个接口电路的结构框图如图 2.2－8 所示。

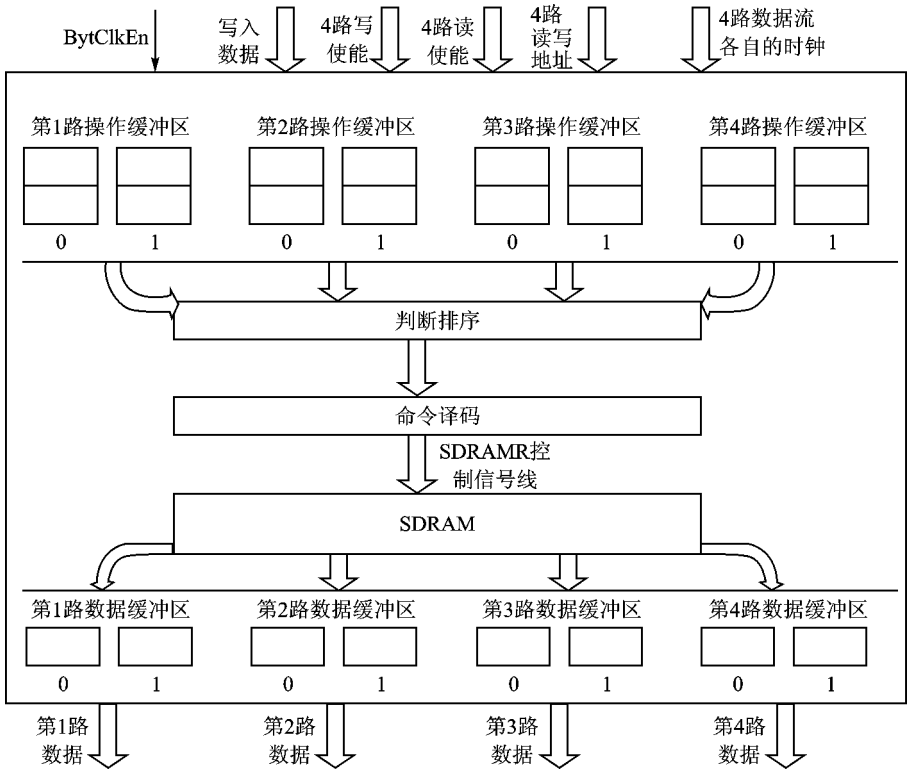


图 2.2－8 SDRAM **接口电路结构图**

四、SDRAM 接口的实现结果

针对 MT48LC8M8A2 **的** SDRAM,**采用同步设计方法,用** Verilog HDL **硬件描述语言建立模型,接口电路已经调试通过,规模为** 2100 **门** (NAND)。**整个解复用电路也已经调试通过。**

参 考 文 献

1 MICRON. SDRAM Data Sheets. http：//www. micron. com/

2 Tomasz Szymanski. SDRAM controller for real time digital image processing systems. CADSM 2001 proceeding

3 **孙 玉. 数字复接技术 (修订本). 北京：人民邮电出版社,**1991

选自《电子技术应用》月刊,2002 **年第** 9 **期**

2.3 SDRAM 视频存储控制器的设计与实现

合肥中国科学技术大学电子科学与技术系 (230026)
罗玉平 施业斌 尹社广 陈海涛

在实时视频处理领域中,大容量存储器的设计始终是重点和难点,而同步动态随机存储器(SDRAM)以其价格低廉、容量大等优点成为普遍采用的方法。SDRAM 控制器的设计是整个存储器设计的主要环节。本文介绍如图 2.3-1 所示的 3D DCT(3 维离散余弦变换)视频压缩系统。它是将视频 A/D 的原始视频数据存入 SDRAM 中,由 3D DCT 运算核从 SDRAM 中取出数据进行压缩处理后再送入传输控制口或发送至视频 D/A。在分析了 Xilinx 公司的 IP Core [1] 后,对其进行修改,并增加由 FIFO 组成的数据缓冲接口,使其更适用于实时视频数据存储、传输。本文将对 SDRAM 接口的存储控制模块进行详细的描述。

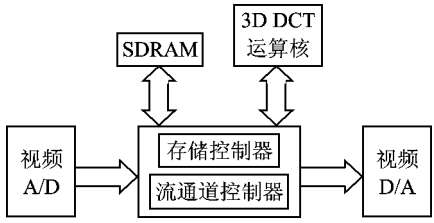


图 2.3-1 3D DCT 视频压缩系统原理图

一、SDRAM 基本操作原理

SDRAM 中的操作主要包括:设置模式寄存器、读写和刷新等操作。以 Micron 公司的 MT48LC1M16A1S-7 (512 KB × 16 × 2) 为例,简要介绍 SDRAM 的操作。SDRAM 操作过程:(1) 在 VDD、VDDQ 电源管脚加电及时钟 CLK 稳定后,采用 Command Inhibit 或者 Nop 命令做 100 μs 延迟。(2) 对所有的区 (Bank) 预充电 (Precharge),SDRAM 进入等待 (Idle) 状态。(3) 2 个周期的自动刷新 (Auto refresh)、设置模式寄存器 (Load MR)。

预充电命令用于取消激活 (Active) 命令所指定行的激活状态。预充电命令中的 A_{10} 决定取消的是某个 Bank 中的行或是所有 Bank 中的行。可以在读写命令中将预充电使能,在读写的突发 (Burst) 结束时执行一个预充电命令,也可以自行给出预充电命令。预充电命令与下一条命令之间的间隔时间为 t_{RP} = 21 ns。在进行读写操作之前,需要通过 BA 和地址线设置 SDRAM 中的模式寄存器以指定操作的模式,包括:突发长度 (Burst Length)、突发类型 (Burst Type)、CAS 延时 (CAS Latency)、写突发模式 (Write Burst Mode)、操作模式 (Operation Mode)。其格式如图 2.3-2 所示。

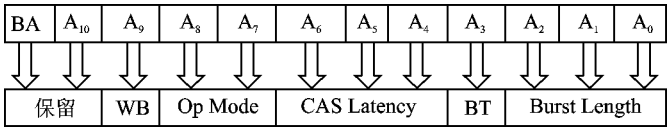


图 2.3-2 SDRAM 模式寄存器定义

图 2.3-2 中,BA 和 A_{10} 为保留位置零。A_9 为定义写突发模式,"0"为突发长度可编程,

“1”为单个访问。突发长度可编程时将 $A_2A_1A_0$ 的值作为突发长度应用到读和写操作中。单个访问时只将 $A_2A_1A_0$ 值应用到读操作中，写为单个写。A_8A_7 为“00”定义标准操作，否则为保留状态。$A_6A_5A_4$ 定义 CAS 延时（值为 1、2、3 个时钟周期）。A_3 定义顺序访问或间隔访问。$A_2A_1A_0$ 定义突发长度为 1、2、4、8 个时钟周期。$t_{MRD}=2t_{Clk}$，即 LoadMR 整个过程需要 2 个时钟周期。

SDRAM 的读操作只有突发读（Burst Read）一种，如图 2.3－3 所示。在时钟 T_0 处给出 Active 命令（由 RAS、CAS、CS 组合成）、行地址（A_{10} A_9……A_0）、Bank 选择（BA），即给出行激活命令。经过 $t_{RCD}=20$ ns 后，由于系统时钟为 81 MHz（6×13.5 MHz），所以在 2 个时钟后即时钟 T_2 处给出读命令和列地址（A_7 A_6……A_0）、Bank 选择。而读出的数据是在读命令之后的第 n 个 CAS Length（CAS 长度 = 1、2、3，图 2.3－3 中设定值为 2）时钟处出现在数据线上。SDRAM 的写操作有突发写和单个写二种。突发写操作时序同读类似，只是写入时的数据同写命令、列地址同时出现在时钟 T_2 处，如图 2.3－4 所示。其中 SDRAM 的命令（Command）编码如表 2.3－1 所列。

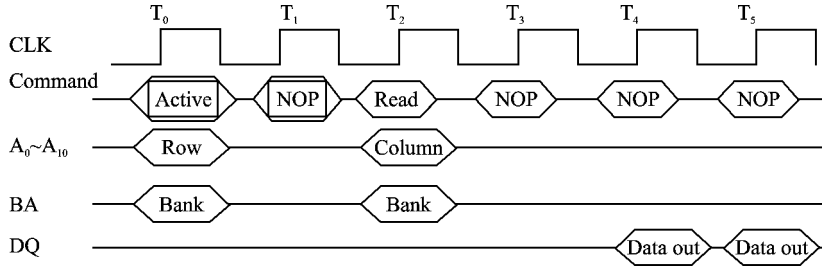


图 2.3－3 SDRAM 读操作时序

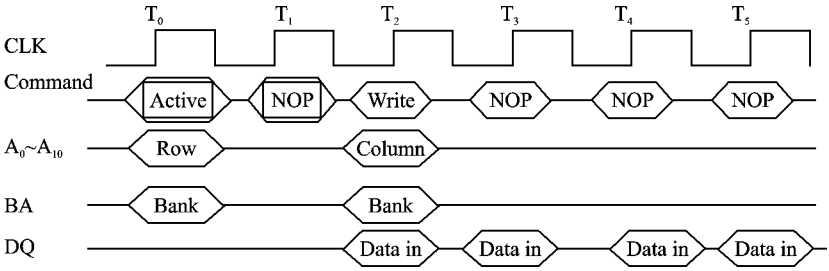


图 2.3－4 SDRAM 写操作时序

表 2.3－1 SDRAM 操作命令真值表

命 令	#CS	#RAS	#CAS	#WE	Addr
Active	L	L	H	H	Bank/Row
Read	L	H	L	H	Bank/Col
Write	L	H	L	L	Bank/Col
Precharge	L	L	H	L	Code
LoadMR	L	L	L	L	Op－Code
Refresh	L	L	L	H	X
Nop	L	H	H	H	X
Com Inhibit（Nop）	H	X	X	X	X

SDRAM视频存储控制器主要由系统缓冲接口和SDRAM操作控制器2个模块组成，如图2.3－5所示。

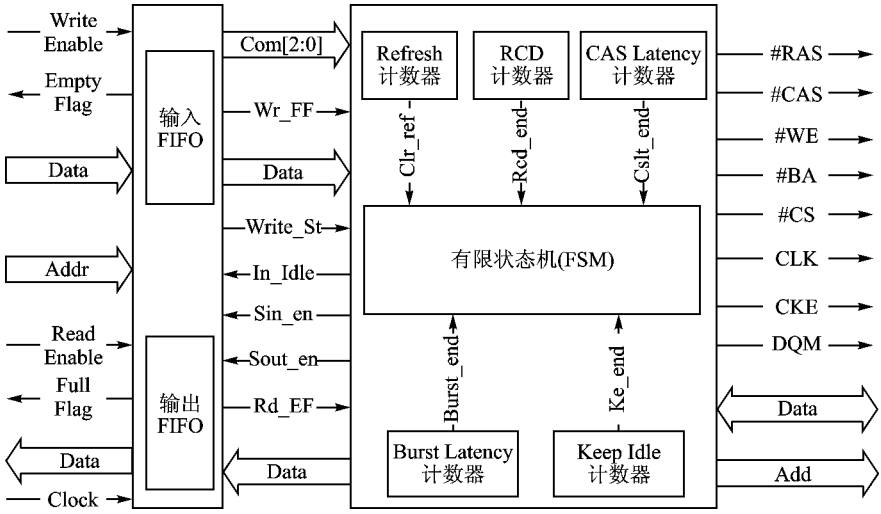


图2.3－5 系统结构原理图

二、SDRAM控制状态机的设计

SDRAM控制状态机如图2.3－6所示。在SDRAM控制器的内部，采用状态机来实现数据读写、设置模式寄存器和刷新等操作的命令译码，产生#RAS、#CAS、#WE、#CS、DOM等信号。

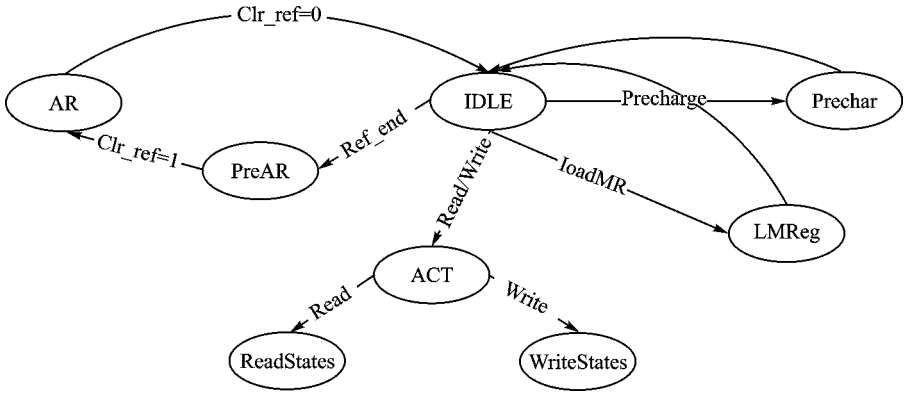


图2.3－6 SDRAM控制状态机

在Micron的SDRAM芯片中，t_{REF} = 64 ms，即需要在64 ms内完成2048行的刷新操作，也就是在31.25 μs内对1行进行1次刷新，并由Refresh计数器定时（T < 31.25 μs）启动刷新命令。状态机处于IDLE状态时，若发现计数器输出信号Ref_end = '1'或发现接收的命令Com ="111"，则转入PreAR，将Clr_ref置'1'（启动刷新后的KeepIdle计数器）。下一时钟则进入AR状态，将Clr_ref清零，并发出Refresh命令至SDRAM。KeepIdle计数器计数结束后才可以从Idle状态转入其他状态，以满足t_{RC} = 70 ns的时序要求。图2.3－6中ACT和Read、Write的状态更加复杂，如图2.3－7所示。

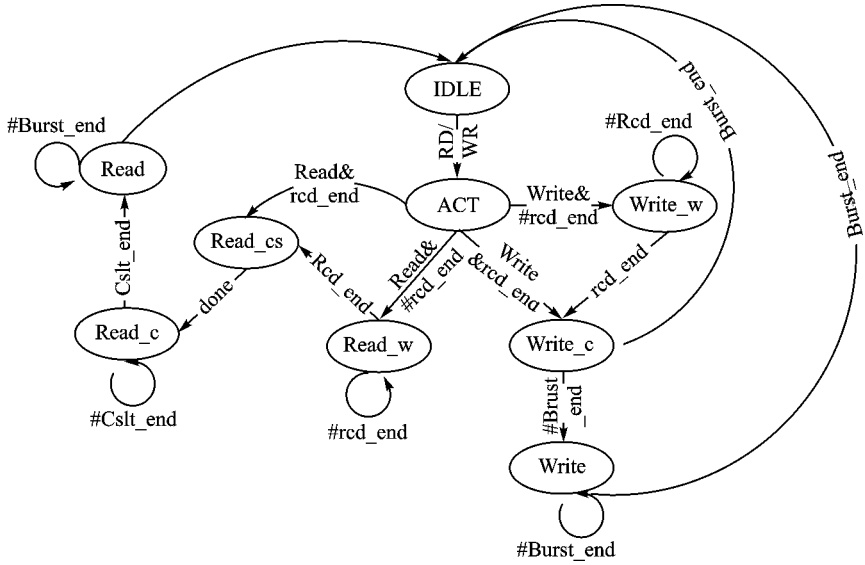


图 2.3－7 SDRAM **读写状态转换图**

由于采用高效的 One－hot **状态机设计方法,可方便地由状态机译出** c_act、c_prech、c_load_mr **等信号。根据表** 2.3－1 **所列的命令译码,对这些信号进行编码后得到** SDRAM **接口的控制信号** :RAS、CAS、DQM、WE、CS **和与** FIFO **接口的信号** In_Idle、Sin_en、Sout_en。**其** VHDL **代码实现如下:**

```
#ras <= not (c_act or c_prech or c_load_mr or c_ar);
#cas <= not (c_read_cs or c_write_c or c_load_mr or c_ar);
#we <= not (c_write_c or c_prech or c_load_mr);
In_Idle <= c_idle and ki_end;
sin_en <= c_write_c or c_write;
sout_en <= c_read_cs or c_read_c or c_read;
```

三、系统缓冲接口设计

对外所提供的系统接口,包括 2 **个** 16×16 bit **的同步** FIFO (**输入** FIFO、**输出** FIFO) **的读写逻辑,以及输入** SDRAM **的地址总线** S_Addr [22: 0]、**视频数据输入与输出的** 2 **条数据总线** Din [15: 0] **和** Dout [15: 0]。

地址总线中 S_Addr [22] **定义** CS1、CS2,Addr [21] **定义** Bank0 **或** Bank1,S_Addr [20: 10] **的** 11 bit **为** RAS **地址,**S_Addr [9: 2] **的** 8 bit **为** CAS **地址。由于在** SDRAM **接口部分** Addr **采用行列地址复用,故需要产生复用控制信号** Add_mux。**它译自** SDRAM **内部状态机,定义为** Add_mux <= C_idle or C_act or C_loadMR or C_ar or C_pre_ar。**当出现上述状态中的一个时,** Add_mux = '1',**则** Addr **提供的是行地址;否则在读或写状态中,**Add_mux = '0',Add **提供的是列地址。双向数据总线** Data **是由内部** 2 **条单向总线** Sd_data_i **和** Sd_data_o **结合** Spartan Ⅱ **的** IOBUF **实现的。在此需要从** SDRAM **状态机中产生** IOBUF **的控制信号** Sd_data_tri,**由** Sd_data_tri <= not (C_write or C_Write_c) **实现。**

如图2.3-5所示,该部分的输入FIFO、输出FIFO接收和发送视频数据。同样利用状态机产生WRITE_PRE[1:0]来完成2个FIFO与SDRAM的读写交替操作,同时产生输入至SDRAM的命令COM[2:0]。COM[2:0]总线的编码如表2.3-2所列。读写交替操作:

```
Flug <= Wr_ff&Rd_ef;
with Flug select
write_pre <= "01"when "10",
             "10"when "01",
             "01"when "11",
             "00"when others;
```

表2.3-2 系统缓冲接口命令的定义

COM[2:0]	命令操作
000	Idle
100	Read/Write
101	Load MR
110	Precharge
111	Refresh

系统缓冲接口与SDRAM控制器进行通信的信号主要有:COM[2:0]、WRITE_ST[1:0]、SIN_EN、SOUT_EN、In_Idle。前2个信号由系统缓冲接口发出,分别提供SDRAM控制器操作命令和读写操作。WRITE_ST[1:0]是由WRITE_PRE[1:0]延时产生,SIN_EN、SOUT_EN是由SDRAM控制器在读写状态下反方向提供给系统接口,执行输入FIFO和输出FIFO的读和写操作。

系统缓冲接口模块的状态机设计如图2.3-8所示。初始化之后IN_IDLE='1'是状态转移的条件,产生COM总线输入至SDRAM控制器状态机,决定SDRAM控制器的动作。

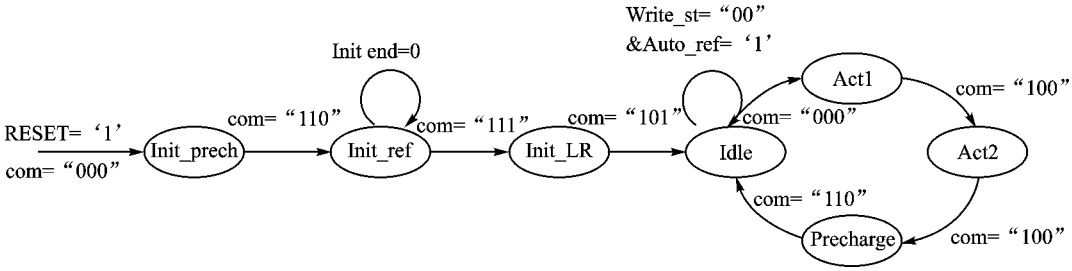


图2.3-8 系统缓冲接口模块的状态机

初始化操作包括1次Precharge操作、2次Auto Refesh循环(Init_end为初始化刷新循环计数器输出),然后是模式寄存器的设置。图2.3-8中的Act1、Act2是连续2次的SDRAM写操作或2次读操作,这是因为输入FIFO和输出FIFO的深度均为16,而SDRAM的Burst长度为8(最大值)。为了保持2次操作的读写一致性,需将WRITE_PRE[1:0]作如下处理:

```
when Act1 => if (In_idle = '1' and (not (write_pre = "00"))) then
    com <= "100";
    write_st <= write_pre;
    temp <= write_pre;next_state <= Act2;
      ……
when Act2 => if (In_idle = '1') then
    com <= "100";
    write_st <= temp;……
```

3D DCT压缩系统中视频SDRAM存储控制器的一种实现及应用方案,其VHDL代码已在

Xilinx 公司的 FPGA SPARTANIT XC2S100 中通过了 ModelSim 的后期仿真、Xilinx ISE 的综合及布局布线。该控制模块共占用 1 个 DLL (延时锁定环路)、494 CLB (可配置逻辑单元),整个 FPGA 的速度可达到 103 MHz,并在所设计的视频 A/D、D/A 接口板中通过了验证,完全可满足输入视频数据的存取操作要求。在其他应用场合中,SDRAM 可以同其他逻辑模块集成在一块 FPGA 中,可大大地节约系统的面积、有效降低开发成本、增加设计的灵活性。

参 考 文 献

1 Xilinx, Inc. Synthesizable High Performance SDRAM Controller. xapp134 (v3.1),2000

2 Micron Technology Inc. MT48LC1M16A1S data sheet. 1999

3 Xilinx, Inc. ISE 4 User Guide. 2001

4 侯伯亨. VHDL 硬件描述语言与数字逻辑电路设计. 西安:西安电子科技大学出版社,1999

选自《微型机与应用》月刊,2002 年第 9 期

2.4 集成多路模拟开关的应用技巧

河南信阳师范学院应用电子技术研究所 (464000) 周胜海

集成多路模拟开关 (以下简称多路开关) 是自动数据采集、程控增益放大等重要技术领域的常用器件,其实际使用性能的优劣对系统的精度和可靠性有重要影响。关于多路开关的应用技术,一些文献上的介绍有两点不足:一是对器件自身介绍较多,而对器件与相关电路的合理搭配与协调介绍较少;二是原则性的东西介绍较多,而操作性的东西介绍较少。研究表明: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,保证各电路单元有合适的工作状态,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,收到预期的效果。本文从应用的角度出发,研究多路开关的应用技巧。目前市场上的多路开关以 CMOS 电路为主,故以下的讨论除特别说明外,均针对这类产品。

一、"先断后通"与"先通后断"的选择

目前市场上的多路开关的通断切换方式大多为"先断后通" (Break - Before - Make)。

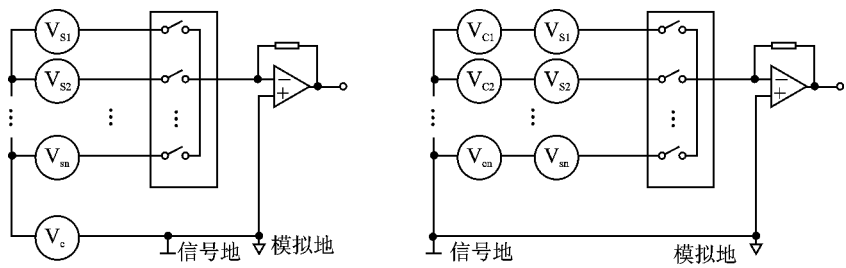
在自动数据采集中,应选用"先断后通"的多路开关。否则,就会发生两个通道短接的现象,严重时会损坏信号源或多路开关自身。

然而,在程控增益放大器中,若用多路开关来改变集成运算放大器的反馈电阻,以改变放大器的增益,就不宜选用"先断后通"的多路开关。否则,放大器就会出现开环状态。放大器的开环增益极高,易破坏电路的正常工作,甚至损坏元器件,一般应予避免。

二、选择合适的传输信号输入方式

传输信号一般有单端输入和差动输入两种方式,分别适用于不同的场合。

单端输入方式如图 2.4-1 所示,即把所有信号源一端接同一信号地,信号地与 ADC 等的模拟地相接,各信号源的另一端分别接多路开关。图中 V_s 为传输信号,V_c 为系统中的共模干扰信号。



(a) 信号地与模拟地共点　　(b) 信号地与模拟地不共点

图 2.4-1 单端输入方式

图 2.4－1 (a) 接法的优点是无需减少一半通道数,也可保证系统的共模抑制能力;缺点是仅适用于所有传输信号均参考一个公共电位,且各信号源均置于同样的噪声环境下,否则会引入附加的差模干扰。

图 2.4－1 (b) 接法适用于所有传输信号相对于系统模拟公共地的测量,且信号电平明显大于系统中的共模干扰。其优点是可得到最多的通道数,缺点是系统基本失去了共模抑制能力。

差动输入方式如图 2.4－2 所示,即把所有信号源的两端分别接至多路开关的输入端。其优点是抗共模干扰的能力强,缺点是实际通道数只有单端输入方式的一半。当传输信号的信噪比较低时,必须使用差动输入方式。

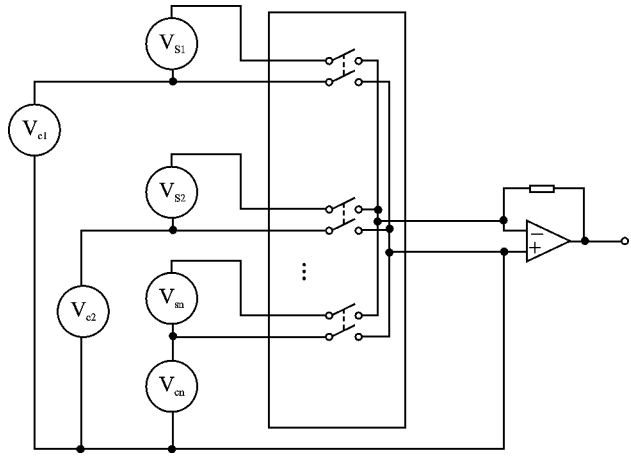


图 2.4－2 差动输入方式

三、减小异通电阻的影响

多路开关的导通电阻 R_{ON} (一般为数 10 Ω 至 1 kΩ 左右) 比机械开关的接触电阻 (一般为 mΩ 量级) 大得多,对自动数据采集的信号传输精度或程控增益放大的增益影响较明显,而且 R_{ON}通常随电源电压高低、传输信号的幅度等的变化而变化,因而其影响难以进行后期修正。实践中一般是设法减小 R_{ON}来降低其影响。

以 CD4051 为例,测试发现[1]:CD4051 的 R_{ON}随电源电压和输入模拟电压的变化而变化。当 $V_{DD}=5$ V、$V_{EE}=0$ V 时,$R_{ON}\approx 280$ Ω,且随 V_i 的变化突变;当 $V_{DD}>10$ V、$V_{EE}=0$ V 时,$R_{ON}\approx 100$ Ω,且随 V_i 的变化缓变。可见,适当提高 CD4051 的 V_{DD}有利于减小 R_{ON}的影响。必须注意:提高 V_{DD}的同时,应相应提高选通控制端 A、B、C 的输入逻辑电平。例如:取 $V_{DD}=12$ V ($V_{EE}=0$ V),可采用电源电压上拉箝位的方法,上拉电阻的阻值取 1.5 kΩ 以上,使选通控制端信号的有效高电平不低于 6 V。这样,既保证 CD4051 理想导通 (R_{ON}小),又实现了 CMOS 电平与 TTL 电平的转换 (μP 一般为 TTL 电平)。

可见,根据具体情况,适当提高多路开关的电源电压,是降低其 R_{ON}影响的一种有效措施。此外,适当提高电源电压,还可以同时减小导通电阻路差 ΔR_{ON}和加快开关速度。

四、消除抖动引起的误差

和机械开关类似,多路开关在通道切换时也存在抖动过程,会出现瞬变现象。若此时采集多路开关的输出信号,就可能引入很大的误差。例如[2],某计算机自动数据采集与处理系统采集三个模拟量:水泵转速、流量、压力。三个模拟量对应的 TTL 电平分别为:1.5454 V,1.5698 V、2.9394 V。采集系统从通道 1、2、3 分别对这三个模拟量连续采集 10 次,采集结果位于 1.8554 ~ 1.8603、1.5625 ~ 1.5673、1.62207 ~ 1.62695 之间,其中 1、3 通道的误差很大。研究发现,这种误差是由于系统在多路开关通断切换未稳定下来就采集数据造成的。

消除抖动的常用方法有两种:一是用硬件电路来实现(硬件方法),即用 RC 滤波器滤除抖动;另一种是用软件延时的方法来解决(软件方法)。在有 μP 的系统中,软件方法较硬件方法更显优势。如上例中,只要在原 Quick - BASIC 数据采集程序中加入一循环语句来适当延时,则采集结果位于 1.5454 ~ 1.5478、1.5698 ~ 1.5722、2.9394 ~ 2.9418 之间,采集精度明显提高,采集结果正常。

五、提高切换速度

多路开关的切换速度与其自身的结构、工作条件以及外电路的情况都有关系。在实践中应注意以下几点:

所有的多路开关的平均传输延迟时间 t_{pd} 均随 V_{DD} 的升高而减小。以 CD4051 为例[3],当 $V_{DD}=5$ V 时,$t_{pd}=720$ ns;当 $V_{DD}=10$ V 时,$t_{pd}=320$ ns;当 $V_{DD}=15$ V 时,$t_{pd}=240$ ns。可见,适当提高多路开关的电源电压,可加快其开关速度。

传输信号的信号源内阻 R_s 对多路开关的切换时间有重要影响。分析表明:在其他条件不变的情况下,切换时间近似与 R_s 成正比,即 R_s 越小,开关的动作就越快。所以,对高内阻的信号源(一些传感器就是如此),宜用阻抗变换器(如电压跟随器),将阻抗变低后再接入多路开关。此外,减小 R_s 还可同时减小多路开关的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 2.4 - 3 所示的两级联接方式。在图 2.4 - 3 中,假设系统共需要 32 个信号通道,将这 32 个通道分成 4 组,各组分别接至 4 个二级开关,信号由二级开关输出。设每个开关的输出电容为 C_o,则输出总电容由 32 C_o 降至大约 12 C_o,电路的时间常数减小,开关速度提高。此外,这种联接方式还可以使多路开关的总关断漏电流由 31 I_z 降至大约 10 I_z(设每个开关的关断漏电流为 I_z),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。当然,这种联接方式需要的开关数相对多些,选通控制也相对复杂些,因而主要用于信号通道数较多的场合。

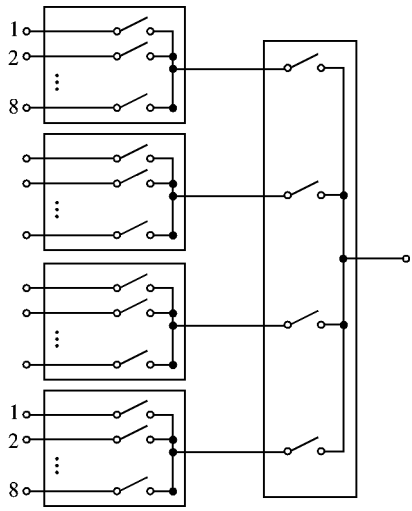


图 2.4 - 3 两级联接方式

目前市场上的多路开关以 RCA、AD、SILICONIX、MOTOROLA、MAXIN 等公司的产品为多见,种

类繁多,性能、价格差异较大(详见有关公司的相关产品数据手册)。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度的要求,同时还必须注意以下两点。

第一,全面了解多路开关的特性,否则可能出现难以预料的问题。例如: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有关芯片。

第二,多路开关只有与相关电路合理搭配、协调工作,才能充分发挥其性能,甚至弥补某些性能的欠缺。否则,片面追求多路开关的高性能,忽视与相关电路的搭配与协调,不但会造成成本与性能指标的浪费,而且往往收不到预期的效果。

此外,受芯片种类或应用场合的限制,在实践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信号,必要时应作适当处理。例如[4]:测试多路开关 CC4097 和 CC4067 时发现,所有多余通道的输入端都必须接地,否则将产生干扰信号。

有关多路开关的基本应用技术,许多文献上都有介绍[5],本文不予赘述。

参 考 文 献

1 周胜海,王栋臣,马建中. 可变增益放大器的实现方法. 仪表技术与传感器,2001 (7):32 ~ 34
2 马明建,周长城. 数据采集与处理技术. 西安:西安交通大学出版社,1999
3 王福瑞. 单片微机测控系统设计大全. 北京:北京航空航天大学出版社,1998
4 房国良. 模拟开关组合应用设计原理. 电子技术应用,1997,23 (11):15 ~ 16
5 郝鸿安. CMOS 模拟集成电路实用电路集. 上海:上海科学普及出版社,1996

选自《电子技术应用》月刊,2002 年第 4 期

2.5 合理选择 DC－DC 转换器

Maxim 公司北京办事处 魏 智

一、概 述

DC－DC 转换器的主要功能是提供稳压，即能够将不稳定的输入电压转换成稳定的输出电压，输出电压不随输入电压或输出电流的变化而改变。这类电路包括 线性稳压器、开关型稳压器、稳压型电荷泵。通常，选择电源芯片的目的是从不稳定的电源获得稳定的输出电压，有些情况下则是将一个稳定的电源电压转换成另一个不同电压的输出。这种情况下，电源调节器只是用于改变电压的电平，无需稳压。

许多应用中直接利用电池为便携式产品供电，这种直接供电方式存在诸多问题。首先，便携式产品中的许多电路所要求的供电电压范围较窄，特别是对于微处理器、存储器、高速器件等。电池在有效使用期内输出电压极可能超出这些芯片所要求的供电范围，如果增加一个稳压器便能够保证电路工作的可靠性。其次，电池的内阻也是一个潜在问题，便携式产品常常存在不同的工作状态，工作电流是变化的。如果直接从电池吸取变化的电流，由于电池内阻的存在会导致电池电压发生变化。如果引入一个稳压器就可以在负载电流变化时保持电源电压的稳定性，因为稳压器的输出电阻要远远低于电池的串联电阻。除了强调开关型稳压器、线性稳压器、电荷泵的稳压功能外，它们还常常用于产生不同幅度的电源电压，即所谓 DC 到 DC 的转换。从理论上讲，上述三种类型的电源芯片都属于 DC－DC 转换器，而在许多场合仅用 DC－DC转换器代表开关型稳压器。

三种电源转换器中，线性稳压器的局限性表现在其输出电压只能低于输入电压 开关电源最为通用，能够提供升压、降压以及极性相反的输出 电荷泵同样可提供升压、降压、极性相反的输出，但其输出电流有限。另外，变压器耦合转换器也常常用来提供输出电压与输入电压不同的变换，个别情况下要求输出电压与输入电压相同，此时，采用变压器耦合电路可以提供电源隔离，如图 2.5－1 所示。选择隔离电源的主要原因是安全，例如，在通过电极与病人接触的供电电路需要与墙上电源插座隔离开，大多数消费类产品都需要提供与交流电的隔离。

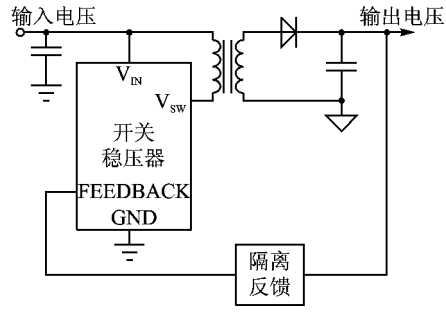


图 2.5－1 变压器耦合开关稳压器提供输出与输入的隔离

二、线性稳压器

线性稳压器,如图 2.5－2 所示,利用一个有源调整元件 (双极型晶体管或 MOSFET) 将输入电压降低至稳定的输出电压,其中低压差 (LDO) 线性稳压器在最近几年的应用已经十分广泛。压差指维持稳定输出所需要的最小电压差 (输入和输出之间)。压差在 1 V 以内的调节器称为 LDO,但现在典型的调节器只有 100～300 mV 的压差。线性稳压器的输入电流接近于输出电流,可以用输出电压与输入电压的比估算它的效率。如果输出电压非常接近输入电压,线性稳压器就可提供较高的转换效率 如果输入电压高出输出电压很多,或者它在很宽的范围内变动,就很难获得比较高的转换效率。线性稳压器具有小尺寸、低成本、低噪声等特点,而且使用非常简单,因此,在电压、电流条件适当的情况下应尽可能选用线性稳压器。除此之外,LDO 调节器还可作为一道屏障来隔离开关调节器产生的噪声。在此用途中,LDO 调节器的低压差特性有利于改善电路的总体效率。由于线性稳压器只能提供低于输入的电压,如果所需要的电压高于输入电压或与输入电压极性相反,则选用开关电源或电荷泵。为了延长电池的工作时间、降低系统的热耗散,常常需要考虑电源的效率。线性稳压器的效率通常低于开关型稳压器,这一点也是值得注意的地方。

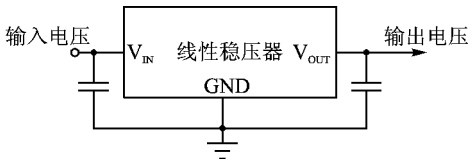


图 2.5－2 结构简单的线性稳压器

三、开关电源

开关电源不具备线性稳压器的优点,与线性稳压器相比,它占用更大的线路板面积 (不考虑线性稳压的散热片)、成本也更高、所产生的噪声较大。而在最近几年,开关电源的普及率却大大提高了,其主要原因是开关电源在不同输入电压、不同负载电流条件下能够提供很高的转换效率,升压、降压型开关电源效率可达 96%。当然,降压型开关电源的转换效率可能更高一些,反激型开关电源的效率也可达到 90%。当需要升压、降压或反极性电压转换、负载电流高于 125 mA 时,最好选用开关型稳压源。当然,用电荷泵同样可实现上述转换,但该类芯片所能提供的负载电流有一定的局限性,尽管市场上也存在负载电流达到几百 mA 的电荷泵,但当负载电流大于 125 mA 时,采用电荷泵结构成本将非常高。

开关稳压器的取名主要源于内部的开关功率管。当与电感器连接时,通过控制电感的充、放电实现电压的转换。如图 2.5－3 所示,当开关功率管断开时,有较高的电压施加在开关管上,但此时没有电流流过开关管,因此,没有功率消耗 ;当开关管处于导通状态时,管子的导通电阻非常小、产生的压降也极小,尽管有电流流过开关管,但功耗很小 ;当功率开关从断开状态过渡到导通状态,或从导通状态过渡到断开状态时,功率开关上会存在一定的压降、同时也有一定的电流通过,此时,功率开关产生较大的电流损耗。为降低功耗,需减小功率开关的切换时间。而在快速切换的瞬间会有大电流流过这些电路,在连线上产生很大的 dV/dt 和 di/dt 信号,因此,开关电源的线路板布局要求很高。尤其是对于高密度、采用多层电路板的开关电源,元件的布局和走线对于电路的正常工作具有重要的影响。不过,只要在关键回路的布局方面多加注意,就可避免上述问题。大多数芯片制造商针对其开关电源芯片提供相应的评估板,用户可以参考评估板的线路板布局进行设计。

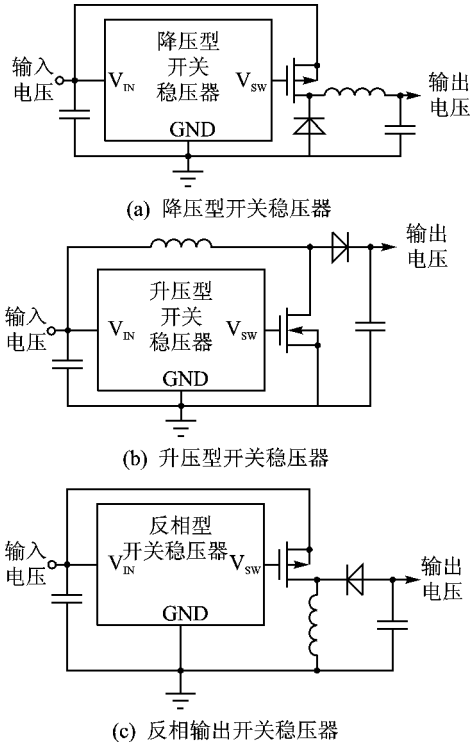


图 2.5－3 开关型稳压器的三种结构

另外，对开关电源外围器件的选择也有一定的要求，这对于控制 EMI 至关重要。没有电感或变压器设计经验的电路设计者倾向于选择商品化的变压器和电感，选择时须注意选用高磁导率的磁芯，以便使磁场局限于磁芯中而不向周围空间散射，另外，由于高磁导率介质不能储存很多能量，为了缩小电感尺寸，常常采用带有气隙的高磁导率磁芯。由于采用带气隙的磁芯会增加电感周围空间的磁辐射，因此，还须采取一些其他措施，如在磁芯外部套装铁氧体屏蔽罩等。在选择旁路电容时，要确保其在可能引起问题的频率范围有足够低的阻抗，通常选用具有低 ESR、ESL 的陶瓷电容用于高频旁路。容量较高的陶瓷电容其电容值会随电压、温度的改变发生较大的变化，可利用陶瓷电容和钽电解电容相并联，既能满足低 ESR、ESL 的要求，又能满足对电容值的需求。

四、电荷泵

电荷泵在三种电源转换电路中知名度最低。它具有和开关电源相同的功能，但不需要电感元件，利用外部电容器可以实现升压、降压、电压逆转变换。这种电路在一定条件下为设计人员带来很大的便利，因为电感的货源较少，标准规格和尺寸较少、体积较大、电磁干扰多且成本较高，同时，对布线敏感，这些都会造成具体设计中的不便。

基本电荷泵结构可以利用模拟开关组成，也可以利用二极管分立元件组成，如图 2.5－4 所示。图 2.5－4 中，C_1 用于传送电荷，C_2 用于保持电荷。在该电路的基础上还可以根据需要扩充修改，如增加稳压、减少噪声、获得更高的输出电压等。基于电容的电荷泵包括稳压型和非稳压型两种。非稳压型电荷泵输出电压随输入电压而改变，输出电阻为一固定值，负载电流

增大时输出电压会相应跌落。在反相模式下,非稳压型电荷泵输出电压与输入电压大小相同、极性相反;在倍压模式下,输出电压是输入电压的 2 倍。这是非稳压型电荷泵的两种常用模式。稳压型电荷泵的输出电压与输入电压之间的关系不像非稳压型电荷泵那样严格。例如,它可以从 3.3 V 的输入电压获得稳定的 5 V 输出,并且,当负载电流在一定范围内变化时输出仍保持稳定。

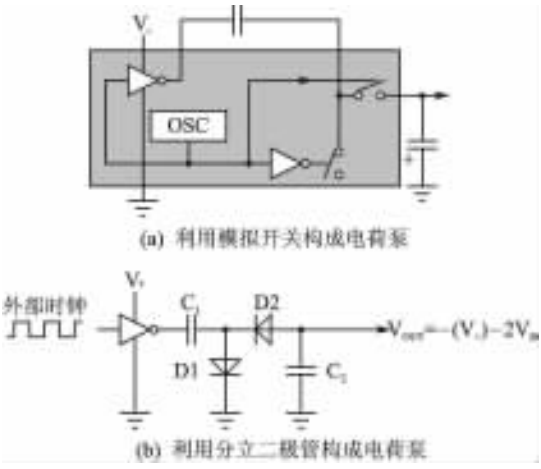


图 2.5－4 电荷泵提供倍压或反相输出

电荷泵在控制切换与之相连接的电容的过程中,会伴随出现一定的噪声,但由于电荷泵用于轻载条件下,而且电路中没有电感,不存在磁性噪声。另外,当电荷泵中断流过电容器的电流时也不会产生峰值电压,所以,电荷泵的噪声低于开关电源所产生的噪声。许多基于电容的电压转换器提供极低的工作电流,这对于工作电流较低的便携式产品十分有利。电荷泵芯片的电源电流总的来说与其工作频率成正比。工作频率较低时,电流消耗较小,但相应的代价是负载驱动能力较低,输出纹波较大,而且所需外部元件较大。选择外部电容时,应该保证输出电压、电流和纹波水平在所允许的范围内。当然,这些参数还与电荷泵的工作频率、开关导通电阻等因素有关。大容量的输出电容可降低输出纹波;电容等效串联电阻越小,输出纹波越小,而且还可降低输出阻抗。

选自《单片机与嵌入式系统应用》月刊,2002 年第 4 期

2.6 单片机定时器中断时间误差的分析及补偿

湖南株洲职业技术学院(412000) 聂 毅

单片机内部一般有若干个定时器,如 8051 单片机内部有定时器 0 和定时器 1。在定时器计数溢出时,便向 CPU 发出中断请求。当 CPU 正在执行某指令或某中断服务程序时,响应定时器溢出中断往往延迟一段时间。这种延时虽对单片机低频控制系统影响甚微,但对单片机高频控制系统的实时控制精度却有较大的影响,有时还可能造成控制事故。为扩大单片机的应用范围,本文介绍它的定时器溢出中断与 CPU 响应中断的时间误差、补偿误差的方法和实例。

一、误差原因、大小及特点

产生单片机定时器溢出中断与 CPU 响应中断的时间误差有两个原因。一是定时器溢出中断信号时,CPU 正在执行某指令;二是定时器溢出中断信号时,CPU 正在执行某中断服务程序。

1. CPU 正在执行某指令时的误差及大小

由于 CPU 正在执行某指令,因此它不能及时响应定时器的溢出中断。当 CPU 执行此指令后再响应中断所延迟的最长时间为该指令的指令周期,即误差的最大值为执行该指令所需的时间。由于各指令都有对应的指令周期,因此这种误差将因 CPU 正在执行指令的不同而不同。如定时器溢出中断时,CPU 正在执行指令 MOV A,Rn,其最大误差为 1 个机器周期。而执行指令 MOV Rn, direct 时,其最大误差为 2 个机器周期。当 CPU 正在执行乘法,或除法指令时,最大时间误差可达 4 个机器周期。在 8051 单片机指令系统中,多数指令的指令周期为 1 ~2 个机器周期,因此最大时间误差一般为 1 ~2 个机器周期。若振荡器振荡频率为 f_{osc},CPU 正在执行指令的机器周期数为 C_i,则最大时间误差为 $\Delta t_{max1} = 12/f_{osc} \times C_i$ (μs)。例如 $f_{osc} = 12$ MHz,CPU 正在执行乘法指令 ($C_i = 4$),此时的最大时间误差为:

$$\Delta t_{max1} = 12/f_{osc} \times C_i = 12/(12 \times 10^6) \times 4 = 4 \times 10^{-6}\ (s) = 4\ (\mu s)$$

2. CPU 正在执行某中断服务程序时的误差及大小

定时器溢出中断信号时,若 CPU 正在执行同级或高优先级中断服务程序,则它仍需继续执行这些程序,不能及时响应定时器的溢出中断请求,其延迟时间由中断转移指令周期 T_1、中断服务程序执行时间 T_2、中断返回指令的指令周期 T_3 及中断返回原断点后执行下一条指令周期 T_4 (如乘法指令) 组成。中断转移指令和中断返回指令的指令周期都分别为 2 个机器周期。中断服务程序的执行时间为该程序所含指令的指令周期的总和。因此,最大时间误差 Δt_{max2} 为:

$$\Delta t_{max2} = (T_1 + T_2 + T_3 + T_4)\ 12/f_{osc} = (2 + T_2 + 2 + 4)\ 12/f_{osc} = 12\ (T_2 + 8)/f_{osc}$$

若设 $f_{osc} = 12$ MHz,则最大时间误差为:

$\Delta t_{max2} = 12 (T_2 + 8) / f_{osc} = 12 (T_2 + 8) /12 \times 10^6 = (T_2 + 8) \times 10^{-6}$ (s) $= T_2 + 8$ (μs)

由于上式中 T_2 一般大于 8,因此,这种时间误差一般取决于正在执行的中断服务程序。当 CPU 正在执行中断返回指令 RETI、或正在读写 IE 或 IP 指令时,这种误差在 5 个机器周期内。

3. 误差非固定性特点

定时器溢出中断与 CPU 响应中断的时间误差具有非固定性特点。即这种误差因 CPU 正在执行指令的不同而有相当大的差异。如 CPU 正在执行某中断服务程序,这种误差将远远大于执行一条指令时的误差。后者误差可能是前者误差的几倍、几十倍、甚至更大。如同样只执行一条指令,这种误差也有较大的差别。如执行乘法指令 MUL AB 比执行 MOV A, Rn 指令的时间误差增加了 3 个机器周期。这种误差的非固定性不仅给误差分析带来不便,同时也给误差补偿带来困难。

二、误差补偿方法

由于定时器产生溢出中断与 CPU 响应中断请求的时间误差具有非固定性,因此,这种误差很难用常规方法补偿。为此,本文介绍一种新方法。现介绍该方法的基本思路、定时器新初值及应用情况。

1. 基本思路

为使定时器溢出中断与 CPU 响应中断实现同步,该方法针对中断响应与中断请求的时间误差,对定时器原有的计数初值进行修改,以延长定时器计数时间,从而补偿误差。在该方法中,当定时器溢出中断得到响应后,即停止定时器的计数,并读出计数值。该计数值是定时器溢出后,重新从 OOH 开始每个机器周期继续加 1 所计的值。然后,将这个值与定时器的停止计数时间求和。若在定时器原计数初值中减去这个和形成新计数初值,则定时器能在新计数初值下使溢出中断与 CPU 响应中断实现同步,从而达到误差的补偿要求。

2. 定时器新计数初值

若定时器为计数方式,操作方式为 1,则计数器初值 $X_0 = 2^{16} - t_0 \times f_{osc} /12$。式中 f_{osc} 为振荡器的振荡频率;t_0 为需要定时的时间,也为中断的间隔时间;X_0 为定时器原计数初值。在对定时器溢出中断与 CPU 响应中断时间误差进行补偿时,定时器的新计数初值 X_1 为:

$$X_1 = 2^{16} - t_3 \times f_{osc} /12$$

$$t_3 = t_0 + t_1 + t_2$$

式中 t_0 为中断间隔时间;t_1 为定时器停止计数时间,该时间为定时器停止计数到重新启动计数之间所有程序指令周期数的总和;t_2 为定时器溢出中断后,重新从 OOH 开始直至计数器停止时计的值。在误差补偿中,若将定时器计数初值 X_1 取代 X_0,则可使定时器下次的溢出中断与 CPU 响应中断实现同步。

3. 实 例

要求补偿定时器每 1 ms 产生一次溢出中断时的中断响应延迟的误差。若振荡器振荡频率 $f_{osc} = 12$ MHz,定时器工作在计数方式,工作模式为 1,则补偿中断响应时间误差时的定时器新初值 X_1 为:

$$X_1 = 2^{16} - t_3 \times f_{osc} /12 = 2^{16} - (t_0 + t_1) - t_2 = 2^{16} - (1\,000 + 13) - t_2$$

误差补偿程序为:

```
……
0    CLR  EA                    ;关 CPU 中断
1    CLR  TRi                   ;停止定时器计数
2    MOV  R0,  #OOH             ;R0 清零
3    MOV  R0,  #LOW (216)       ;定时器最大计数值的低 8 位送 R0
4    MOV  A,   R0
5    SUBB A,   #LOW (1000 + 13) ;216 的低 8 位减去 (t0 + t1) 的低 8 位送累加器 A
6    SUBB A,   TLi              ;216 的低 8 位减去 (t0 + t1 + t2) 的低 8 位送 TLi
7    MOV  TLi, A
8    MOV  R0,  #OOH             ;R0 清零
9    MOV  R0,  #HIGH (216)      ;216 的高 8 位送 R0
10   MOV  A,   R0
11   SUBB A,   #HIGH (1000 + 13);216 的高 8 位减去 (t0 + t1) 的高 8 位送 A
12   SUBB A,   THi              ;216 的高 8 位减去 (t0 + t1 + t2) 的高 8 位送 A
13   MOV  THi, A
14   SETB TRi                   ;重新启动定时器
……
```

在上式和上段程序中,由于 f_{osc} = 12 MHz,中断间隔时间为 1 ms,因此 t_0 的机器周期数为 1 000。由于第 1 条指令到第 14 条指令的指令周期的机器周期数之和为 13,因此,t_1 为 13 个机器周期。CPU 虽在执行第一条指令 CLR TRi 后停止定时器计数,但在 TLi、THi 中分别保存了 t_2 的低位数据和高位数据。

由于本文介绍的误差补偿方法能对定时器溢出中断与 CPU 响应中断的非固定性时间误差进行有效补偿,因此,该方法对于提高高频控制系统实时控制精度和扩大单片机应用范围都有较高的实用价值。

参 考 文 献

1 曹巧媛. 单片机原理及应用. 北京:电子工业出版社,1997
2 汪吉鹏等. 微机原理及接口. 北京:高等教育出版社,2001

选自《微计算机信息》月刊,2002 年第 4 期

2.7 单片机无线串行接口电路设计

南华大学 黄智伟 朱卫华

一、概 述

单片机无线串行接口电路由 MICRF102 单片发射器芯片、MICRF007 单片接收器芯片组成,工作在 300～440 MHz ISM 频段;具有 ASK 调制和解调能力,抗干扰能力强,适合工业控制应用;采用 PLL 频率合成技术,频率稳定性好;接收灵敏度高达 - 96 dBm,最大发射功率达 - 2.5 dBm;数据速率可达 2 Kb/s;低工作电压:4.75～5.5 V;功耗低,接收时电流 3 mA,发射时电流 7.75 mA,接收待机状态仅为 0.5 μA,发射待机状态仅为 1.0 μA;可用于单片机之间的串行数据无线传输,也可在单片机数据采集、遥测遥控等系统中应用。

二、电路组成及工作原理

1. 无线发射电路

无线发射电路如图 2.7－1 所示,电路以 MICRF102 为核心。MICRF102 是 Micrel 公司推出的一个单片 UHF ASK 发射器,采用 SOP(M)－8 封装,芯片内包含有:由基准振荡器、相位检波器、分频器、带通滤波器、压控振荡器构成的合成器,发射偏置控制,RF 功率放大器,天线调谐控制和变容二极管等电路,是一个真正的"数据输入－无线输出"的单片无线发射器件。UHF 合成器产生载频和正交信号输出。输入相位信号(I)用来驱动 RF 功率放大器。天线调谐正交信号(Q)用来比较天线信号相位。天线调谐控制部分检测天线通道中发射信号的相位和控制变容二极管的电容,以调谐天线,实现天线自动调谐。功率放大器输出受发射偏置控制单元控制。ASK/OOK 调制,提供低功耗模式,数据传输速率为 20 Kb/s。

使用中应注意的问题是:(1) REFOSC(引脚 4)是基准振荡端,连接晶振到地,或采用 AC 耦合方式输入峰－峰值为 0.5 V 的时钟脉冲。发射频率是基准振荡器频率的 32 倍:基准振荡频率 ×32 = 发射频率。如果使用外接时钟信号,须采用 AC 耦合方式,输入信号幅度峰－峰值为 200～500 mV。(2) MICRF102 使用差分输出去驱动天线负载。功率放大器输出级包含有一个变容二极管,它自动与天线的电感调谐,以保证谐振在发射频率上。典型的 PCB 导线天线的电感与回路的尺寸、天线导线的宽度、PCB 铜泊的厚度和接地板的位置有关。设计时一般选择变容二极管的电容值为 6.5 pF。天线电感 L 由公式 $L = 1/(4\pi^2 f^2 C)$ 计算。(3) 功率放大器的输出功率与 PC 端(引脚 1)上的电压有关。正常工作时,该引脚端上的电压被设置在 0.2～0.4 V 之间。PC 端上的电压上升,输出功率加大;但是,如果 PC 端上的电压超过0.4 V,功率放大器被限流,输出功率不再增加。减少 PC 端的电压可降低电源功率消耗,同时也会减少 RF 输出功率。(4) STBY 端(引脚 5)是待机模式控制。接 V_{DD}为发射方式,接 V_{SS}为待机模式。(5) MICRF102 芯片对电源纹波敏感,正确地电源旁路是必需的,一般使用4.7 μF、0.1 μF、100 pF 3 个电容并联在 V_{DD}和 V_{SS}之间。

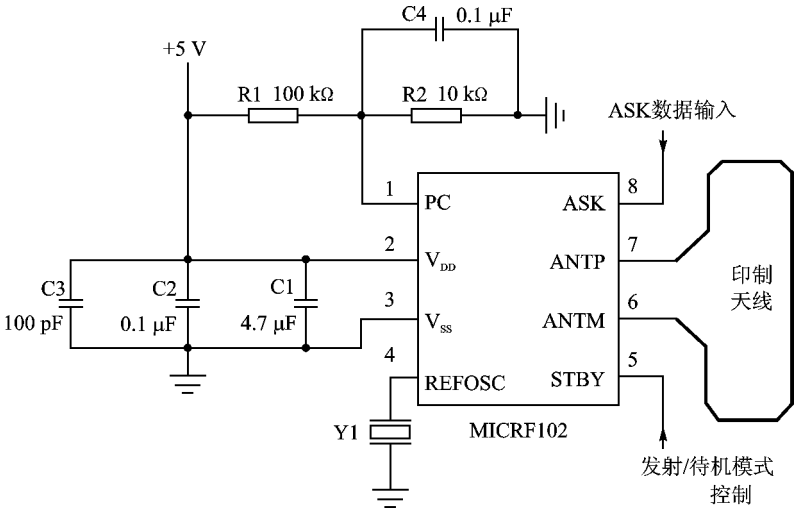


图 2.7－1 无线发射电路

2. 无线接收电路

无线接收电路如图 2.7－2 所示，电路以 MICRF007 为核心。MICRF007 是 Micrel 公司推出的单片 UHF ASK/OOK (导通-关断键控) 超外差无线电接收芯片。MICRF007 采用 SOP (M) －8 封装，芯片内电路可分为 UHF 下变换器、OOK 解调器和基准控制三部分。UHF 下变换器包含 RF 放大器、混频器、中频放大器、带通滤波器、峰值检波器、合成器、AGC 控制电路；OOK 解调器包含低通滤波器、比较器；基准控制电路包含基准振荡器和控制逻辑电路。仅需外接 2 个电容器 C_{AGC} 和 C_{TH}，1 个晶振以及电源去耦电容即可构成 1 个 UHF ASK 接收器，所有的 RF 和 IF 调谐都在芯片内自动完成，是一个真正“无线输出-数据输入”的单片器件。

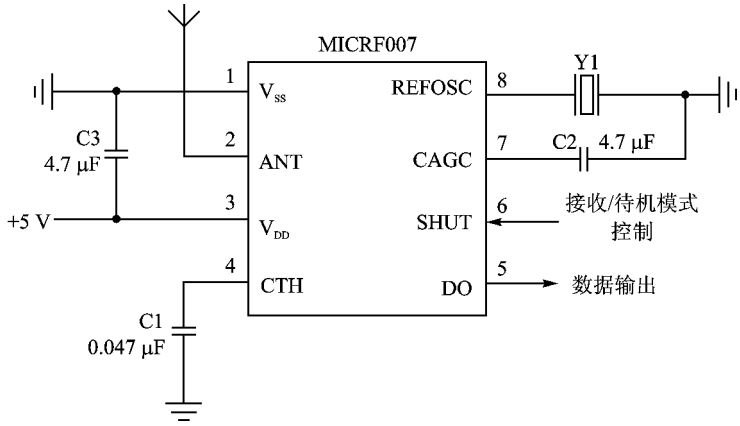


图 2.7－2 无线接收电路

MICRF007 是标准的窄 RF 带宽的超外差接收器，窄带宽接收器对 RF 干扰信号不敏感。RF 中心频率由完全集成的 PLL/VCO 频率合成器控制，与基准振荡器外接晶振有关。中频带通滤波器的带宽为 430 kHz，基带解调器的低通滤波器带宽为 2.1 kHz。接收数字 ASK 信号，接收器数据传输率为 2 Kb/s。

使用中应注意的是：(1) MICRF007 是一个窄带宽接收器，要求发射电路必须使用 SAW 或晶振稳频。(2) 如果接收器处于高噪声环境，在天线 ANT 端和 V_{SS} 之间可以连接一个固定数值的带通网络，以提供接收选择性和输入过载保护。(3) 基准振荡器可通过 REFOSC 端 (引脚 8) 外接晶振或输入时钟信号。基准振荡器的频率 f_T 是外接晶振频率的 64.5 倍。对于超外差接收器本机振荡频率 f_{LO} 和发射频率 f_{TX} 的差值必须等于中频的中心频率。因此，发射器的频率 f_{TX} (即接收器接收频率)、基准振荡器频率 f_T 和本机振荡器频率 f_{LO} 的关系为：$f_T = f_{LO}/64.5$，$f_{LO} = f_{TX} \pm (1.064 f_{TX}/390)$。(4) SHUT 端 (引脚 6) 控制接收器使能，当 SHUT 端电压 V_{SHUT} 为高电平时，芯片进入低功耗待机模式，电流消耗仅为 0.5 μA；当 V_{SHUT} 为低电平 (下拉到地) 时，芯片使能，为接收状态。(5) CTH 端 (引脚 4) 上的解调信号的直流值作为比较器的基准阈值。CAGC 端 (引脚 7) 外接电容 C_2 可增加输入动态范围。(6) MICRF007 芯片对电源纹波敏感，正确地电源旁路是必需的。一般使用 4.7 μF、0.1 μF、100 pF 3 个电容并联在 V_{DD} 和 V_{SS} 之间。

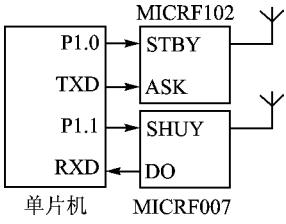


图 2.7-3 与单片机的接口电路

3. 单片机串行接口电路

无线收发电路可以直接与常用的单片机如 8051、68HC05、PIC16C5X 等连接，实现单片机与单片机之间的串行数据无线传输，连接电路如图 2.7-3 所示。

三、结束语

实验表明：所设计的单片机串行接口无线收发电路结构简单、工作可靠，可方便地在单片机与单片机之间，构成一个点对点、一点对多点的无线串行数据传输通道。

使用中应注意的问题是：①在发射模式下，通信速率最高为 2 Kb/s；发送数据之前须将电路置于发射模式 (MICRF102 的第 5 脚 STBY = 1)；接收模式转换为发射模式的转换时间至少 5 ms；可以发送任意长度的数据；发送结束后应将电路置于接收模式 (MICRF007 的第 6 脚 SHUT = 0)；发射模式转换为接收模式的转换时间至少 5 ms。②在待机模式 (MICRF102 的 STBY = 0，MICRF007 的 SHUT = 1) 下，电路不发射/接收数据。设计串行通信程序应考虑：双方通信的协议，有效数据识别标志，数据的检错、纠错和校验。

参 考 文 献

1 Micrel Inc. QwikRadio™ UHF ASK Transmitter www.micrel.com. 2001.8

2 Micrel Inc. QwikRadio™ Low Power UHF Receiver www.micrel.com. 2001.8

选自《单片机与嵌入式系统应用》月刊，2002 年第 2 期

2.8 单片机控制 Modem 的两种硬件接口方法

武汉大学电气工程学院 (430072) 宣 扬 张俊敏

本文将基于 80C196KC 单片机介绍如何用单片机实现与 Modem 的硬件接口以支持软件的操作。

一、Modem 硬件接口标准

Modem 硬件接口符合 EIA 的 RS-232C 串行接口标准,其连接器有 DB-9 和 DB-25 两种,通常使用的信号线只有 9 根,可分为 3 类。在此我们必须明确本文中的数据终端即指单片机,数据设备指的是 Modem。各信号线定义如下。

1. 联络控制信号线

DSR (数据设备就绪):有效时 (ON 状态) 表示 Modem 处于可使用状态;DTR (数据终端就绪):有效时 (ON 状态) 表示数据终端可用。

注意,这两个信号有效,只表示设备本身可用,并不表示通信线路可开始通信。

RTS (请求发送):有效时 (ON 状态) 表示数据终端向 Modem 请求发送数据;CTS (允许发送):有效时 (ON 状态) 表示 Modem 已准备好接收数据终端发来的数据。

这两个信号是用于半双工 Modem 系统中发送和接收方式之间的切换。全双工中因配置了双向通道,故无须 RTS/CTS,可置其为高电平。

RI (振铃指示) ——当 Modem 收到交换台送来的振铃呼叫信号时,使该信号有效 (ON 状态),通知数据终端已被呼叫。

2. 数据发送和接收线

TD (串行数据发送):通过此线数据终端将串行数据发到 Modem;RD (串行数据接收):通过此线数据终端接收 Modem 发来的串行数据。

3. 地 线

PG,SG:设备保护地信号地,无方向。

上述控制信号线何时有效,何时无效的顺序表示了接口信号的传送过程。由此可见,用单片机控制 Modem 关键就是如何和 Modem 进行数据传输和如何对联络控制信号线进行控制和响应。

二、直接连接法

1. 控制线路

直接连接法接线图如图 2.8-1 所示。目前的 Modem 均为全双工方式,故可不接 RTS、CTS 两线。Modem 的 DTR 脚接单片机的输出口 P1.0,DSR、DCD、RI 脚接单片机的 HSI1、HIS2、HSI3 脚。单片机的串行口的收、发数据线接 Modem 的发、收数据线。因 Modem 的引脚电平为 RS-232 标准的,单片机引脚是 TTL 电平的,所以这里的连接不是直接连接。它们之间的连线 (地线

除外) 必须经过 TTL/232 电平转换电路,这里用两片 MAX232 芯片即可实现。

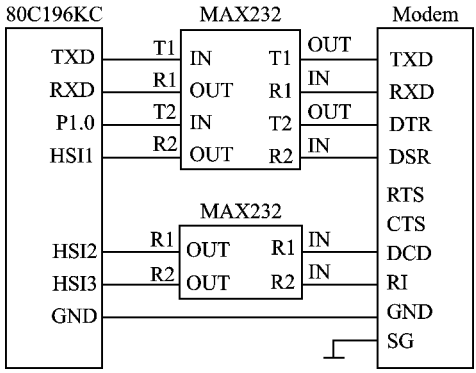


图 2.8－1 直接连接法接线图

2. 控制方法

(1) 数据的接收与发送:对数据的接收和发送直接操作单片机的串行口。须注意,Modem 接收 AT 命令时不用校验位,可置串口工作方式为 1。信道建立时可改变串口工作方式。

(2) 联络控制信号线的控制与响应:DSR、DCD、RI 接单片机的高速输入口 (HIS 口)。HSI 口可及时记录引脚上的电平发生了怎样的变化、变化发生时间及当前引脚电平状态并能产生中断及时通知 CPU,所以非常适合检测由 Modem 发来的状态信号。DSR、DCD 信号均要求其状态一 改变 (正跳变和负跳变) 即被检测出,而 RI 信号要求在振铃后端 (负跳变) 被检测出,故应设置 HSI1、HSI2 口工作方式为每跳变记录,而 HSI3 口为负跳变记录。可在 HIS 中断程序中处理 Modem 状态变化。DTR 信号由单片机发出,通知 Modem 数据终端就绪。

(3) 中断控制。这里涉及到单片机的串行中断 (数据的收、发) 和 HIS 中断 (Modem 状态变化),可利用单片机的中断屏蔽寄存器使能或禁止两种中断。

三、扩展 8250 芯片控制法

INS8250 芯片是一通用异步通信控制器,用其控制 Modem 的方便之处在于其具有 Modem 的控制功能和完整的状态报告功能。8250 芯片在众多微机接口书中有详细介绍,读者可阅读参考文献 [2]。这里只介绍 8250 与 80C196KC 单片机的硬件连接。

1. 控制线路

图 2.8－2 所示为 8250 与单片机数据线、地址线、控制线等连接的具体情况。这里值得注意的是 8250 的 XTAL1 脚,其输入的是芯片的基准时钟:这里接 80C196KC 单片机的 CLKOUT 脚。CLKOUT 脚输出的脉冲频率为单片机晶振的 2 分频。若单片机晶振取 8 MHz,则 CLKOUT 输出 4 MHz 频率的脉冲,8250 的基准时钟即为 4 MHz。8250 产生的波特率是由下式决定的:除数寄存器 = 基准时钟/(16 × 波特率),所以基准时钟取 4 MHz 会产生波特率误差。例如,想得到 9600 的波特率,取除数寄存器为 26,由上式可计算出实际波特率为 9 615,误差为 (9 600 - 9 615)/9 600 = - 0.1%。若串行线路两端均为上述电路,可不考虑波特率误差;若线路不同则应考虑波特率的误差问题,一般在 11 位帧传输中误差应小于 2.25%。8250 芯片有专有的 RS－232 信号线,但信号是 TTL 电平的,所以须经过 TTL/232 的电平转换电路即可直接连接 Modem 接口中的相应的信号线。

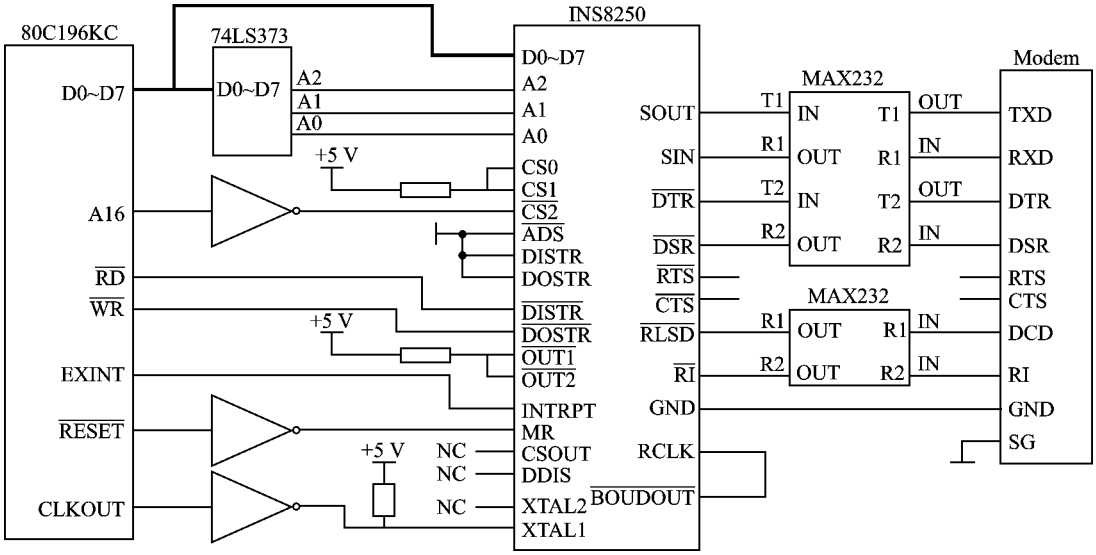


图 2.8－2 扩展 8250 芯片控制 Modem 的接线图

2. 控制方法

8250 控制 Modem 的方法很简单。8250 内部有 10 个可访问的寄存器，只需对这些寄存器进行读写即可进行数据传输和对 Modem 控制、获取 Modem 状态。因为 8250 芯片只有 3 根地址线 A0～A2，只能产生 8 个寄存器地址，所以这里的 10 个寄存器有共用的情况。这里专门设置了一除数寄存器访问允许位 DBAL（线路控制寄存器中的最高位）以增加可访问寄存器的个数。表 2.8－1 列出如何选通 8250 内部各寄存器。

表 2.8－1 8250 内部寄存器及选择方法

CS2	DBAL	A2 A1 A0	被访问的存储器
0	0	000	收发缓冲器
0	0	001	中断允许寄存器
0	*	019	中断识别寄存器
0	*	011	线路控制寄存器
0	*	100	Modem 控制寄存器
0	*	101	线路状态寄存器
0	*	110	Modem 状态寄存器
0	1	000	除数锁存器（低）
0	1	001	除数锁存器（高）
1	*	* * *	禁止读写

8250 可产生四种中断：发送中断、接收中断、接收出错中断、Modem 状态变化中断。可通过对其中断允许寄存器设置来使能或禁止各种中断。再由图 2.8－2 中 8250 的中断线 IN-

TRPT 接单片机的外中断脚 EXINT。这样通过使能或屏蔽单片机的外中断可控制 8250 是否可向单片机申请中断。

四、结束语

上述两种实现方法,方法一控制线路简单,方法二控制方法容易。总体看来,这两种方法均具有硬件电路简单、成本低、控制方便、可靠性高等优点,有一定的实用价值。我们在开发远程集中抄表系统中,这种方法在用单片机系统实现的集中器模块中得到了很好的应用。

参 考 文 献

1 刘乐善. 微型计算机接口技术原理及应用. 武汉 华中理工大学出版社

2 孙涵芳. Inter 16 位单片机. 北京 北京航空航天大学出版社

选自《工业控制计算机》月刊,2002 年第 9 期

2.9 使用 PWM 得到精密的输出电压

江南大学 朱立强

近年来,许多单片机生产厂家,如 Atmel、Analog Divices、Intel、Philips、Dallas、Maxim 等等,纷纷推出了新型的高速单片机。它们的指令执行周期仅是原来的 1/3 ~ 1/十几,并在单片机中集成了 EEPROM、WDT、A/D 转换器和 D/A 转换器,大大地提高了单片机的性能,方便了用户。然而,许多单片机中的 D/A 转换器的输出都采用了脉宽调制 (PWM) 的形式。PWM 十分适用于开关电源、可控硅等器件的控制,也可使用于 LCD 亮度控制、音频输出等不需要输出精确电压的场合。由于 PWM 没有基准电压,它的输出脉冲的幅度不是很恒定,这就限制了 PWM 的使用范围。在要求输出精密控制电压的场合,如精密可调电压源、电机变频器等等,就无法使用 PWM。

然而,只需使用 2 片廉价的集成电路就可以把幅度不恒定的 PWM 输出转换成精密的 PWM 输出电压。

一、电路原理

使用三端精密基准电源和模拟开关得到电压精密的 PWM 脉冲的电路原理如图 2.9-1 所示。D1 为 TL431 三端基准电压集成电路,U1 采用单刀双掷的模拟开关 MAX4544;电阻 R1、R2、R3 根据具体的需要而定。当然,也可以采用其他型号的集成电路。

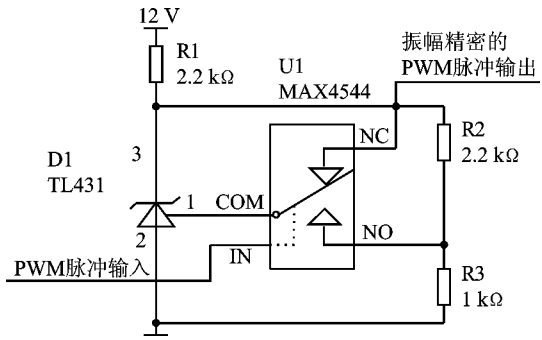


图 2.9-1

当 PWM 脉冲为高电平 (逻辑 1) 时,U1 的 COM 端掷向常闭端 (NC),TL431 的调整脚与正电压脚相连,输出电压值为 2.5 V。当 PWM 脉冲为低电平 (逻辑 0) 时,U1 的 COM 端掷向常开端 (NO),TL431 的输出电压经过 R2、R3 分压后送到调整脚,此时输出电压值等于 [(R2 + R3) /R3] ×2.5 V。本例中输出电压等于 8 V。这样,当 U1 的 IN 脚输入 PWM 信号时,电路相应地输出高电平为 8 V,低电平为 2.5 V 的 PWM 脉冲,其振幅为 8 V - 2.5 V = 5.5 V。如果需要输出低电平为零的 PWM 信号,则再加上 1 个差分放大器就可解决。

在对于精密度的要求不是很高的场合,可以采用更简单的方法。图 2.9-2 为使用精密稳

压二极管对 PWM 脉冲进行稳压限幅的电路图。在图 2.9－2 中,PWM 信号经过高速运算放大器 U1 放大成为 ±12 V 的输出电压,在经过 R1 的限流和 D1 的稳压后,得到 ±6.5 V 的 PWM 脉冲输出。

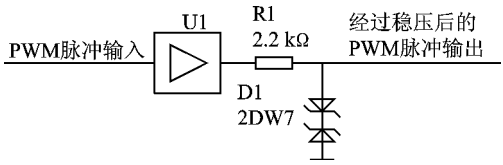


图 2.9－2

二、误差分析

图 2.9－1 中,只要基准电源选取恰当,基准电源本身的误差完全可以忽略。除此之外,误差的来源主要有以下几个方面。

1. 模拟开关的导通电阻引起的误差

模拟开关导通时有一定的导通电阻。TL431 调整脚输入电流通过模拟开关时就会形成电压降,产生误差。MAX4544 的导通电阻为 35 Ω,而 TL431 的调整脚输入电流则在 4 μA 以下。由此而导致基准电压的误差小于 140 μV,为 2.5 V 的 0.000 056,相当于二进制 14 位的精度。

2. 开关延迟时间引入的误差

开关延迟时间将会引起脉冲占空比的变化,从而导致 PWM 输出脉冲产生误差。MAX4544 的导通时间为 30 ns,关断时间为 25 ns。计算可知,当 PWM 频率为 10 kHz 时,由此产生的误差最大为 0.000 3,相当于 12 位的精度。如果 PWM 的频率选得较低,则开关延迟时间的影响相应减小。例如选取 1 kHz 时,引入误差为 0.000 03,相当于 15 位的精度。

上述两项中真正影响输出电源精度的是这些参数随温度和时间的漂移。由于这两项参数本身的绝对值非常小,可以推知它们的漂移更小。

从以上的分析可知,由于附加电路引入的误差完全能够满足 PWM 的精度需求。

图 2.9－2 电路中,引起误差的原因主要有 3 个方面。

⑴ 稳压二极管的动态电阻引入的误差

稳压二极管的动态电阻比较大,一般在几十 Ω 左右(工作电流 5～10 mA 时),而运算放大器的驱动能力比较小,只能使稳压二极管工作在较小的工作电流下。另外,稳压二极管小电流工作时的动态电阻更大,更容易引起电压变化。

⑵ 稳压二极管温度漂移引入的误差

2DW7(2DW230～236)内部结构可以认为是 2 个稳压二极管对接串联而成的。其中 1 个二极管的正向电压降(具有负温度系数)对另 1 个稳压二极管的温度漂移(具有正温度系数)进行补偿,得到很低的温度系数。然而,当 2DW7 反向应用时,其温度漂移就不能得到恰当的补偿,从而导致负脉冲部分的温度系数较高。

⑶ 运算放大器引入的误差

运算放大器的输入失调电压的漂移可直接导致脉冲振幅的误差,而转换速率(SR)过低,将导致脉冲方波波形的失真,继而引起电压的误差。失调电压温度漂移低并且转换速率高的运算放大器的价格将会很高。

但是，对于 8～10 位的 PWM 而言，该电路已经能够满足要求。对于要求更低的场合，可以用 2 个廉价的稳压二极管对接来代替 2DW7。

三、应用实例

利用单片机的 PWM 输出，在图 2.9－1 的基础上增加 RC 滤波电路和 1 级运算放大器，得到0～10 V直流输出电压，作为变频器的控制信号，取得了良好的效果。图 2.9－3 所示为使用 PWM 输出控制变频器的实例。

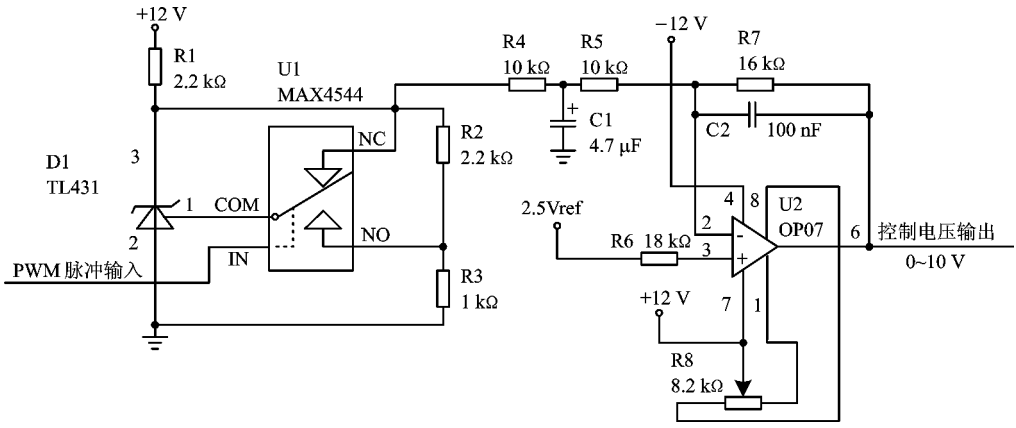


图 2.9－3

选自《单片机与嵌入式系统应用》月刊，2002 年第 11 期

2.10 测控系统前向通道的误差分析及标定

沈阳电力高等专科学校(110036) 杨秀敏
东北大学机械工程与自动化学院(110006) 张 镭
辽宁省信息产业厅科技与产品处(110032) 张 震

一、引 言

在过程控制及各种智能仪器仪表中,多采用计算机进行数据处理及实时控制,由于被检测及被控制的对象往往是一些连续变化的模拟量(如温度、压力、速度等),总要有与被测对象相联系的前向通道。一般讲,在数据处理过程中,软件的计算误差极小,完全可以忽略不计,数据处理的精度主要决定于采样精度,即最终的精度主要取决于前向通道的精度。因此,对于任何测控系统,其前向通道的误差分析都是必须的。

前向通道设计时要考虑到传感器或敏感元件选择、通道结构、信号调节与滤波、A/D 转换、电源配置等,如图 2.10－1 所示。因此,前向通道的精度也主要取决于这些环节的各部分精度,这些环节的误差均将反映在组合信号中。本文以图 2.10－1 电路为例,通过对测控系统的前向通道各环节的误差进行分析,给出前向通道精度的分析方法。

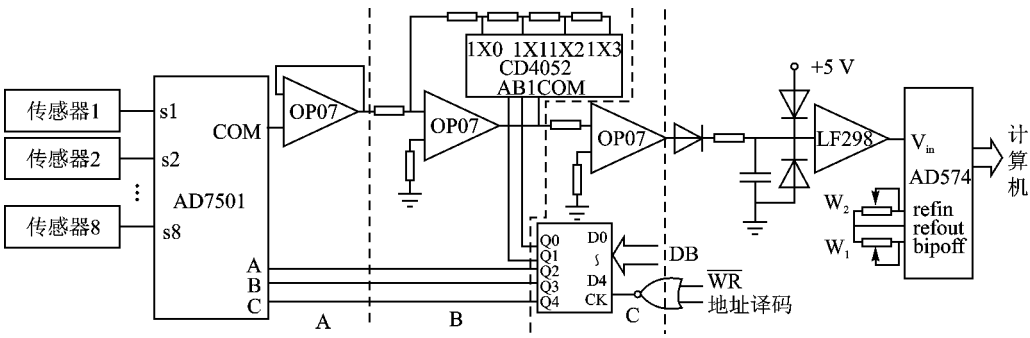


图 2.10－1 多路信号输入的前向通道结构

二、前向通道精度的误差分析

用来描述前向通道数据采集和转换系统精度的最常用方法是计算系统中各环节的均方根误差。由下式表示:

$$\varepsilon_{前} = \pm \sqrt{\varepsilon_1^2 + \varepsilon_2^2 + \varepsilon_3^2 + \varepsilon_4^2}$$

式中 ε_1——模拟多路选择器误差;

ε_2——输入放大器误差;

ε_3——采样/保持电路误差；

ε_4——A/D 转换器误差。

前向通道中信号源阻抗、信号带宽、A/D 转换器分辨率和系统的通过率都会影响这些误差的计算，为简化起见，在下述假定下进行误差计算。

⑴ 忽略孔径误差，即认为孔径误差小于 1/10LSB；

⑵ 信号源阻抗小于 100 Ω；

⑶ 信号幅值在 A/D 转换器最大量程范围内；

⑷ 系统通过率小于系统的最大通过率。

在正常情况下，前向通道的总误差应该小于或等于 A/D 转换器的量化误差。

在小信号的高速、高精度采样时，系统中的采样/保持电路十分重要，用户有必要详细研究采样/保持电路设计精度的诸问题。

1. 多路开关 AD7501 误差分析

多路开关产生误差的原因有二。

⑴ 漏电流造成的信号衰减误差 ε_{11}：多路开关中，未接通的通道有漏电流，该漏电流通过未接通的通道开关、接通的通道开关及信号源内阻形成回路，从而在信号源上产生了压降，信号被衰减。

⑵ 接通电阻造成的信号衰减误差 ε_{12}：在多路开关中，接通的那一路，开关本身的“接通电阻”使输入模拟信号在该电阻上产生压降，则信号被衰减。

因此，多路开关电路可能存在的总误差为：

$$\varepsilon_1 = |\varepsilon_{11}| + |\varepsilon_{12}|$$

2. 放大滤波电路的误差分析

图 2.10－1 电路相当于三级比例放大电路，需要分区计算各部分误差，然后求和。其中一个区的误差各因素如下（具体计算参看文献 [2]）：

⑴ 由于开环差模放大倍数和差模输入电阻为有限值所造成的误差 ε_{Ar}；

⑵ 由共模抑制比为有限值产生的误差 ε_K；

⑶ 由输入偏置电流、失调电压、失调电流和电源电压抑制比等参数影响产生的误差 ε_{IO}；

⑷ 由运放电路中电阻的阻值误差使比例放大电路产生的误差 ε_R。

则该区的综合误差为：

$$\varepsilon_A = |\varepsilon_{Ar}| + |\varepsilon_K| + |\varepsilon_{IO}| + |\varepsilon_R|$$

整个放大滤波电路的总误差可取各区误差之和为：

$$\varepsilon_2 = |\varepsilon_A| + |\varepsilon_B| + |\varepsilon_C|$$

3. 采样/保持电路误差分析

采样/保持器的四个瞬变阶段是：采样、由采样瞬变到保持、保持、由保持瞬变到采样。

采样瞬间产生的误差主要是偏移误差和增益误差：

⑴ 偏移误差 ε_{31}：采样瞬间虽然输入信号是 0，但仍有信号输出，这是由放大器的偏移电压造成的。这一误差可通过调节外部电位器来补偿。

⑵ 增益误差 ε_{32}：输出信号不等于输入信号，但与输入信号成比例。这是由放大器的增益误差造成的。通常，增益误差可通过调节系统的增益来补偿。

⑶ 孔径误差 ε_{33}：控制信号由“1”跳变到“0”时输出信号不是立即保持下来而是继续变

化,这就造成了误差。造成孔径误差的原因是由于孔径时间和充电误差。

孔径时间,从指令信号给出瞬间到模拟开关有效切断,存储电容器停止充电为止所经历的时间称作孔径时间。由于孔径时间的存在,存储电容器存储的电荷量就不是指令信号发出瞬间所要充的电荷量(或者说要保持的电压值),而是出现了误差,即理想保持电压与实际保持电压之差。孔径时间的大小很大程度上取决于开关的性能和某些不定因素的影响。

充电误差,保持周期开始瞬间,在开关断开时,由于开关驱动电路(如使用场效应管)极间电容的电荷变成存储电容器的电荷而产生的一种误差。结果输出信号有一跳跃。

最大孔径误差为:

$$\varepsilon_{33} = 2\pi f t_A$$

其中 f 为信号频率,t_A 为孔径时间。

⑷ 在保持时间内,保持电压下降 ε_{34}:在保持状态下,由于保持电容的漏电流会使保持电压不断地衰减或上升,保持电压的变化速率为:

$$dV_o/dt = I_D/C_H$$

式中 I_D 为保持阶段流入或流出保持电容 C_H 的总泄露电流。增大电容 C_H 值可以减少变化速率,但会增加捕获信号的辅助时间。

因此,采样/保持电路可能存在的总误差为:

$$\varepsilon_3 = |\varepsilon_{31}| + |\varepsilon_{32}| + |\varepsilon_{33}| + |\varepsilon_{34}|$$

4. A/D 转换电路的误差分析

⑴ 失调误差 ε_{41}:主要由转换电路中比较器的输入失调电压、输入失调电流和输出漏电流产生的,当输入信号为零时,输出信号不为零的误差。

⑵ 增益误差 ε_{42}:主要由参考电压、增益误差及电阻容差引起的,当输入信号为零时,输出信号不为零的误差。

一般讲,调整前增益误差和失调误差都较大,所以失调和增益调整必不可少。经仔细调整,可使这两项误差减小到 ±0.024% 以下。

⑶ 量化误差 ε_{43}:决定于转换电路的转换特性。量化误差和分辨率一般讲是统一的,量化误差是由于有限数字对模拟数字进行离散取值(量化)而引起的误差,理论上为一个单位的分辨率:$\pm\frac{1}{2}$LSB。

⑷ 非线性误差 ε_{44}:指在整个量程范围内任一数字量所对应的模拟输入量的实际值与理论值之差。

⑸ 温度漂移误差 ε_{45}:指 A/D 转换电路受环境温度影响的程度,一般用每摄氏度温度变化所产生的相对误差作为指标。

⑹ 电源波动误差 ε_{46}:A/D 变换电路中包含有运算放大器,有的还带有参考电源。因此供电电源的变化会直接影响 A/D 转换精度。A/D 变换电路对电源变化的灵敏度一般用相对误差表示,但实际工作中也常用绝对误差,即最低有效位的变化来表示。

所以,A/D 转换电路可能存在的总误差为:

$$\varepsilon_4 = |\varepsilon_{41}| + |\varepsilon_{42}| + |\varepsilon_{43}| + |\varepsilon_{44}| + |\varepsilon_{45}| + |\varepsilon_{46}|$$

三、前向通道精度的精度标定

1. 模数转换电路的调整与标定

当图 2.10－1 中的 AD574 模数转换电路采用双极性输入，接通电源，在模拟信号输入端加上 $-\frac{1}{2}$FSR $+\frac{1}{2}$LSB 的电压（在 -5 V 和 +5 V 量程中，此电压值为 $-5+\frac{1}{2}\times\frac{10}{1\,096}=-4.998\,8$ V），控制 A/D 转换输出，调节电位器 W_1，使输出数字在 0000 0000 0000 与 0000 0000 0001 之间等几率变动，则第一个阶跃点核准。

把模拟电压输入改为 $+\frac{1}{2}$FSR $+1\frac{1}{2}$LSB 的电压（在 -5 V 和 +5 V 量程中，此电压值为 $+5-\frac{3}{2}\times\frac{10}{4\,096}=-4.996\,3$ V）控制 A/D 转换输出，调节电位器 W_2，使输出数字在 1111 1111 1111 与 1111 1111 1110 之间等几率变动，则最后一个阶跃点核准。

在执行调整时，转换器应首先经过适当时间的预热，使组件内部达到热平衡。改变 AD574 的输入电压值，将输入值及所读取的经 AD574 转换后的数字量相互对应做进一步的标定，一般来讲，只要调整适当，AD574 转换的转换精度很高，线性度很好，不须补偿和修正。对 -5 V ~ +5 V 双极性输入的 AD574 模数转换电路，输入信号 v_i 与输出数字量 y 之间的理论关系为：

$$y=(v_i+5)\times\frac{4\,095}{10}$$

2. 采样/保持电路的调整与标定

图 2.10－1 中采样/保持电路采用的是 LF398，其 2 脚为输入偏移调整端，在 2 脚外接一精密可调电位器，通过调整应保证采样保持电路的电压跟随性很好，一般不须补偿和修正。在理想情况下，输出值应等于输入值：$v_{oi}=v_i$。

3. 对滤波放大电路的调整和标定

如图 2.10－1 所示，滤波放大电路的中心元件选用的是 OP07，其 1 脚和 8 脚是调零端，接一 10 kΩ 的精密微调电位器，通过调整该电位器，可使 OP07 的输出零点对应于输入零点。

采用适当频率范围的信号发生器作为滤波电路的信号输入端，用示波器分别观察输入与输出信号波形，可以测得滤波电路的实际滤波效果，并取得相应的标定数据。用电位差计作为整个滤波放大电路的信号输入源，数字电压表采集输出端电压信号，可取得滤波放大电路的标定数据。

在对前向通道的各部分电路进行调整和标定之后，还须对整个前向通道系统进行整体标定。

四、标定数据分析

各通道标定数据需经线性回归分析，模型为：

$$Y=ax+b$$

其中 a、b 值可由最小二乘法获得。在回归分析中，由标准差可以预报各通道和系统在正态分布情况下的值域，标准差 σ 由有限测量数据的标准差计算公式求得：

$$\sigma=\sqrt{\frac{\sum(y_i-\bar{y}_i)^2}{N-2}}$$

式中 y_i——输出信号测量值；

$\overline{y_i}$——回归方程计算值；

N——测量次数。

上述标定过程为静态标定，可以满足大多数测量条件的误差估计。如工作条件为高速数据采集系统，且输入为阶跃信号等，应考虑动态标定。

五、结束语

由于数据处理过程中软件的计算误差极小，完全可以忽略，因此测量系统的精度主要取决于前向通道的精度。在测量系统的设计中，应将对测量系统的综合精度要求在前向通道的各个环节上进行分配，以确定 A/D 转换器的精度及分辨率、采样保持电路的选型以及其他集成电路的型号、电阻、电容的精度。因此，根据本文给出的方法，不但可以对已设计完成的测控系统前向通道各环节的误差进行定量的分析，而且可以直接用于测控系统前向通道各器件参数的选择等初始设计。

参 考 文 献

1 何立民编著. MCS－51 单片机应用系统设计. 系统配置与接口技术. 北京：北京航空航天大学出版社，1990

2 童诗白主编. 模拟电子技术基础（第二版）. 北京：高等教育出版社，1988

3 杨秀敏. AD574 与单片机的接口技术. 沈阳电力高等专科学校学报，1993（4）：1～4

选自《微处理机》月刊，2002 年第 1 期

2.11 如何认识和提高 ADC 的精度

西北工业大学电子工程系 (710072) 李文峰 王永生
广东省南海建筑设计院有限公司 (528200) 何敏丽

在将模拟量转换成数字量的过程中,模/数转换器 (ADC) 是一核心器件。它把采集到的采样模拟信号量化和编码后,转换成数字信号并输出。ADC 的精度是整个电路和系统精度至关重要的部分,因此有必要认识一下 ADC 的精度,并且了解如何提高 ADC 的精度。

一、如何认识 ADC 的精度

1. 精度和分辨率的关系

ADC 的精度和分辨率是两个不同的概念。精度指转换后所得结果相对与实际值的准确度;分辨率是指转换器所能分辨的模拟信号的最小变化值。ADC 分辨率的高低取决与位数的多少。设 ADC 的位数为 n (一般有 8、10、12、14、16 位等),满量程电压为 FSR,ADC 分辨率与位数之间的关系如表 2.11-1 所列。

表 2.11-1 ADC 分辨率与位数之间的关系 (设 FSR 为 10 V)

位数	码数	相对分辨率 (1LSB)	分辨率 (1LSB) /mV
8	256	0.391%	39.1
10	1 024	0.097%	9.77
12	4 096	0.024%	2.44
14	16 384	0.006%	0.61
16	65 536	0.001 5%	0.15

一般来讲,分辨率越高,转换误差越小;但影响精度的因素较多,分辨率很高的 ADC,可能并不一定具有很高的精度。

影响 ADC 精度的因素之一就是量化误差,这是无法减小的,一般在 ±1LSB/2 范围内;另外还包括偏移 (失调) 误差、增益误差、非线性误差、电源误差等。

2. 偏移 (失调) 误差

偏移误差是指使最低有效位成"1"状态时的实际输入电压与理论输入电压之差。这一差值电压称作偏移电压,一般以满量程电压值的百分数表示。由于偏移电压的存在,如图 2.11-1 (a) 所示,ADC 的传输特性曲线在横轴方向有一个相应的位移,而误差特性则有一个纵向位移,如图 2.11-1 (b) 所示。在一定温度下,偏移电压可以通过外部电路予以抵消。但当温度变化时,偏移电压又将出现,这主要是输入失调电压及温漂造成的。一般来说,温度变化较大时,要补偿这一误差是困难的。

3. 增益误差

增益误差是指满量程输出数码时,实际模拟输入电压与理想模拟输入电压之差。它使传

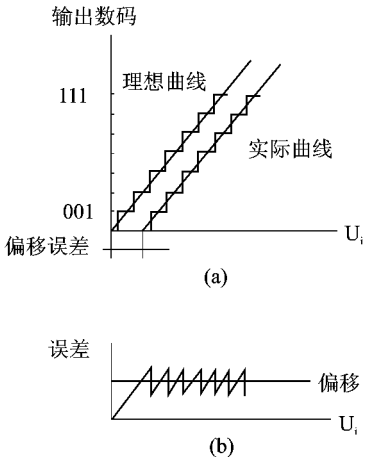


图 2.11－1 偏移误差

输特性曲线绕坐标原点偏离理想特性曲线一定的角度，如图 2.11－2 所示。它的数值一般用满量程电压的百分数表示。

ADC 的理想传输函数的关系式为

$$U_n = U_{REF}(2^{-1}a_1 + 2^{-2}a_2 + \cdots + 2^{-n}a_n) \quad (1)$$

式中，U_n 是没有量化误差时标准模拟电压。由于存在着增益误差，式 (1) 将变为

$$U_n = KU_{REF}(2^{-1}a_1 + 2^{-2}a_2 + \cdots + 2^{-n}a_n) \quad (2)$$

式中的 K 为增益误差因子。当 K = 1 时，没有增益误差；当 K > 1 时，传输特性的台阶变窄，在输入模拟信号达到满量程值之前，数字输出就已“饱和”(即全“1”状态)；当 K < 1 时，传输特性的台阶变宽，输入模拟信号已超满量程值时，数字输出还未达到全“1”状态输出。

在一定温度下，可通过外部电路的调整使 K = 1，从而消除增益误差。但当温度变化时，增益误差又将出现。

4. 线性误差

线性误差是指在没有增益误差和偏移误差的条件下，实际传输特性曲线与理想特性曲线之差，如图 2.11－3 所示。线性误差通常不大于 ±1LSB/2。因为线性误差是由 ADC 特性随模拟输入信号幅值变化而引起的，因此，线性误差是不能进行补偿的，且当温度变化时，数值还会增加。

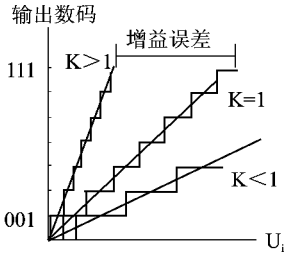


图 2.11－2 增益误差

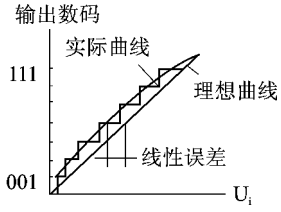


图 2.11－3 线性误差

5. 电源电压变化误差

电源电压由于时间和温度的影响而改变，可能为 ±1%。因而电源的精度对整个系统的精度产生影响，电源电压变化引起的转换误差在高精度要求场合不可忽略。

二、ADC 各项误差分析

现以美国 Analog Devices 公司生产的 AD678 为例分析 ADC 的各项误差。

AD678 是一个高档、多功能的 ADC，内部含有采样/保持器、高精度参考电源、内部时钟和三态缓冲数据输出等部件。只需很少的外部元件就可以构成完整的数据采集系统一次 A/D 转换仅需要 5 μs 进入 www.analog.com 主页在 product 网页可以查到关于 AD678 的资料，如表 2.11－2 所列 (摘录)。

表 2.11－2 ADC678 性能（V_{CC} = ±12 V±5%，V_{EE} = - 12 V±5%，V_{DD} = +5 V±10%）

参 数	AD678J/A/S			AD678K/B/T			单位
	最小	典型	最大	最小	典型	最大	
温度范围							
J,K 级（商用）	0		+70	0		+70	
A,B 级（工业用）	- 40		+85	- 40		+85	
S,T 级（军用）	- 55		+125	- 55		+125	
精度							
分辨率	12			12			位
非线性（INL）		±1			±0.7		LSB
单极性偏移误差（@ +25 ）[1]		±4			±2	±3	LSB
双极性偏移误差（@ +25 ）[1]		±4			±3	±5	LSB
增益强差（@ +25 ）[1,2]		±4			±3	±6	LSB
温漂							
单/双极性偏移温漂误差							
J,K 级		±2			±2	±4	LSB
A,B 级		±4			±3	±4	LSB
S,T 级		±5			±4	±5	LSB
增益[3]							
J,K 级		±4			±4	±6	LSB
A,B 级		±7			±5	±9	LSB
S,T 级		±10			±8	±10	LSB
增益[4]							
J,K 级		±2			±2	±4	LSB
A,B 级		±4			±3	±4	LSB
S,T 级		±6			±5	±6	LSB
内部基准电源							
输出电压[5]	4.98		5.02	4.98		5.02	V
负载							
单极性			+1.5			+1.5	mA
双极性			+0.5			+0.5	mA
电源							
V_{CC} = +12 V±5%		±2				±2	LSB
V_{EE} = - 12 V±5%		±2				±2	LSB
V_{DD} = +5 V±10%		±2				±2	LSB
功率		560	745		560	745	mW

注：1 可调至零；2 包括内部基准电源误差；3 包括内部基准电源温漂；4 排除内部基准电源温漂；5 最大负载下。

表中的温漂范围在 T_{MA} 到 T_{MIN} 之间，设室温为 25 器件的工作温度范围为 0 ~ +50 ，单极性，满量程电压为 10 V，大体上可对 AD678J/K 各项误差做一估算。如表 2.11－3 所列。

表 2.11－3 ADC678 的各项误差

序号	误差源	性 能	误差/%
1	量化误差（ε_1）	1LSB	0.024
2	线性误差（ε_2）	1LSB	0.024
3	线性温漂误差（ε_3）	<1LSB/2（估计）	0.012
4	偏移失调误差（ε_4）	4LSB	0.096
5	偏移温漂误差（ε_5）	2LSB	0.048
6	增益误差（ε_6）	4LSB	0.096
7	增益温漂误差（ε_7）	6LSB	0.144
8	电源电压变化误差（ε_8）	6LSB	0.144

$$\varepsilon_{总静态误差} = (\varepsilon_1^2 + \varepsilon_2^2 + \varepsilon_4^2 + \varepsilon_6^2 + \varepsilon_8^2)^{1/2} \quad (3)$$

$$\approx 0.201\% \text{（均方根值）}$$

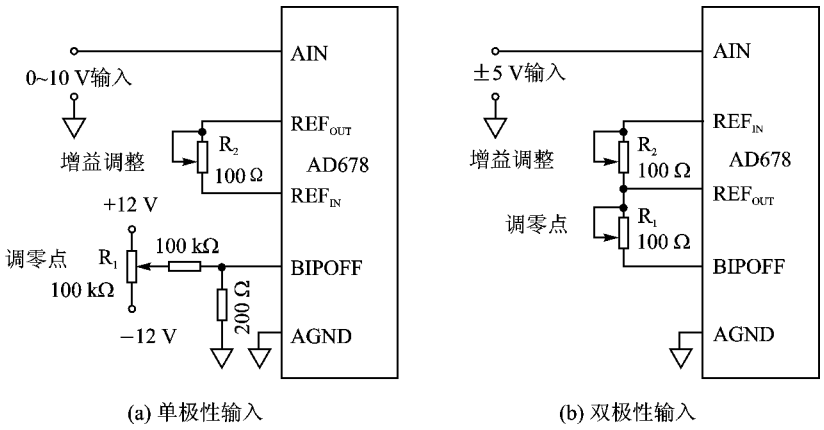
$$\varepsilon_{总误差} = (\varepsilon_1^2 + \varepsilon_2^2 + \varepsilon_3^2 + \varepsilon_4^2 + \varepsilon_5^2 + \varepsilon_6^2 + \varepsilon_7^2 + \varepsilon_8^2)^{1/2} \quad (4)$$

$$\approx 0.252\% \text{（最坏情况）}$$

三、如何提高 ADC 的精度

1. 通过外部电路调整和校准 ADC 的零点和增益

为了提高 ADC 的精度，在静态条件下，可以将偏移误差和增益误差调整至接近于零。具体外部调整电路依 ADC 型号而定，仍以 AD678 为例，调整电路如图 2.11－4 所示。



(a) 单极性输入　　(b) 双极性输入

图 2.11－4　AD678 外部调整电路

AD678 的引脚 REF_{IN} 和 REF_{OUT} 是用来调节增益的。引脚 REF_{IN} 与权电阻网络相连，通过 100 Ω 电位器来调节各级权电流。对电位器的要求是 1% 的精度和低温度系数（100×10^{-6}/　），金属陶瓷电位器即可满足要求。引脚 BIPOFF 可用来调零点，在模拟输入最小的情况下，单极性接近 0 V，双极性接近 - FSR/2。调增益在最大值附近进行。为防止两个电位器互相影响，应先调零点，再调增益。

2. 用软件来消除 ADC 的偏移温漂误差

在用计算机对模拟信号采集情况下，首先对未加载的传感器进行检测，并将所有零电平信号 U_{ZOi} 读入计算机内存中相应的单元，然后才开始采样程序的执行。在采样程序中，将采集到的数据与零电平相减，从而基本上消除偏移温漂误差。

3. 采用高精度稳定电源供给，减少电源电压变化误差

基准电压是提供 ADC 转换时的参考电压，是保证转换精度的基本条件。在要求较高精度时，基准电压要考虑单独用高精度稳定电源供给。此外，外加模拟电源和数字电源也要尽量采用稳定性高（电源电压敏感度 < 0.002%）、受温度变化小的电源。

四、结 论

采取上述措施后,再来分析以下 AD678 的各项误差,见表 2.11-4,并且与表 2.11-3 做一比较。

表 2.11-4 ADC678 的各项误差(提高精度后)

序 号	误差源	性 能	误差%
1	量化误差 (ε'_1)	1LSB	0.024
2	线性误差 (ε'_2)	1LSB	0.024
3	线性温漂误差 (ε'_3)	<1LSB/2 (估计)	0.012
4	偏移失调误差 (ε'_4)	→0	→0
5	偏移温漂误差 (ε'_5)	→0	→0
6	增益误差 (ε'_6)	→0	→0
7	增益温漂误差 (ε'_7)	2LSB	0.048
8	电源电压变化误差 (ε'_8)	0.002%	0.002

于是

$$\varepsilon^{*}_{\text{总静态误差}} = [(\varepsilon'_1)^2 + (\varepsilon'_2)^2 + (\varepsilon'_8)^2]^{\frac{1}{2}} \tag{5}$$

$$\approx 0.034\%\ (\text{均方根值}) < \varepsilon_{\text{总静态误差}}$$

$$\approx 0.201\%\ (\text{均方根值})$$

$$\varepsilon^{*}_{\text{总误差}} = [(\varepsilon'_1)^2 + (\varepsilon'_2)^2 + (\varepsilon'_3)^2 + (\varepsilon'_7)^2 + (\varepsilon'_8)^2]^{\frac{1}{2}} \tag{6}$$

$$\approx 0.060\%\ (\text{最坏情况}) < \varepsilon_{\text{总误差}} \approx 0.252\%\ (\text{最坏情况})$$

比较表 2.11-3 和表 2.11-4,不难看出:在采取了部分提高精度的措施后,ADC 精度有了比较大的提高。实际上,所有误差都恰好在同一级的可能性极小,而静态误差则又可能过于乐观,因为毕竟误差源的数目考虑较少。但是,无论如何,ADC 的精度介于 0.04% ~0.252% 之间。

参 考 文 献

1 马明建,周长城. 数据采集与处理技术 [M]. 西安:西安交通大学出版社,1998

选自《测控技术》月刊,2002 年第 8 期

2.12 提高 ADC 分辨率的硬件和软件措施

天津大学精密仪器与光电子工程学院 (300072)
李素芳 李 刚 孙景发

为了适应计算机、通信和多媒体技术的飞速发展以及高新技术领域的数字化进程不断加快,模数转换器 (ADC) 不仅在工艺、结构、性能上都有了很大的进步,并且正在朝着低功耗、高速、高分辨率的方向发展。在实际应用中,例如数字音响系统,多媒体,地震勘探仪,声纳,电子测量等领域往往需要高分辨率、高精度的模数转换器,这样提高 ADC 的分辨率和精度已经成为 ADC 发展的必然趋势。为此,我们通过一些具体实用的硬件和软件等方法,可以在现有较低分辨率 ADC 的基础上,有效地实现高分辨率 ADC。

一、提高 ADC 分辨率的硬件方法

输入信号为 V_{in},参考电压为 V_{ref},以 8 位 ADC 为例,则输入的数字量为

$$D = (2^8 - 1) V_{in}/V_{ref} \quad (1)$$

若参考电压为 5 V,则此时分辨率为 $5V/2^8 = 19.53$ mV。

1. 对输入信号的调理的方法

ADC 对输入信号的转换是在整个信号幅度范围内的线性量化,在实际测量中对小信号的量化关系到 ADC 的分辨率问题,对于 8 位 ADC 其分辨率为 19.53 mV,此时,理论上 ADC 所能分辨的最小信号不应小于 19.53 mV。若采用程控放大技术,可实现在大信号状态下用小放大倍数,在小信号状态下用大放大倍数,例如放大倍数为 1、10、200、500、1000 等,这样使 ADC 的动态范围扩大了 60 dB,理论上,分辨率也可以提高到 0.01953 mV (实际上达不到,因为噪声的限制),这样针对小信号实现了分辨率的提高,对大信号效果不明显。

对于多通道 ADC,可以利用多通道选通的方法来提高 ADC 的分辨率,即将输入信号分几个区间,然后分别放大转换。例如一个 4 通道的 8 位 ADC,对应关系见表 2.12-1,根据输入信号的大小系统自动选择 4 档中合适的档位进行差分放大处理,产生低 8 位的数字信号,而高 2 位可通过系统控制自动产生,并计算组合成为 10 位数字信号,这样分辨率提高了 2 位,由原来的 19.53 mV 提高到了 4.88 mV,原理如图 2.12-1 所示。

表 2.12-1 输入信号与档位选择关系

地址码	ADC 通道	输入信号范围/V	高 2 位数字信号
00	IN0	0 ~ 1.25	00
01	IN1	1.25 ~ 2.5	01
10	IN2	2.5 ~ 3.75	10
11	IN3	3.75 ~ 5	11

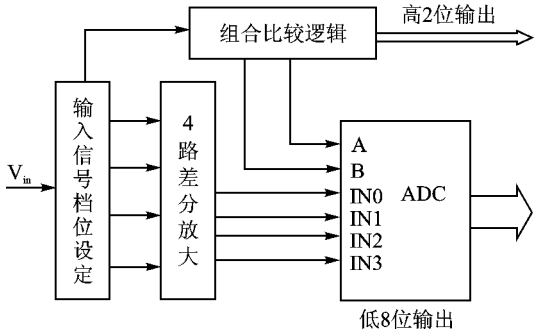


图 2.12－1

2. 处理参考电压法

由公式(1)可以看出,ADC 的分辨率和参考电压有着直接的关系,要保证参考电压的直流精度与要求的 ADC 分辨率相匹配,例如若要求 ADC 有 0.1% 的分辨率,则参考电压的直流精度必须小于 0.05%。这样我们应该尽量采用标准的高精度基准电压模块或采用二级稳压措施。

通过参考电压的直接降压可以提高 ADC 的分辨率,例如将 5 V 的参考电压降至 1 V,则分辨率由原来的 19.53 mV 提高到 3.91 mV;但是与此同时输入信号的范围也在减小为 $0 \leqslant V_{in} \leqslant V_{ref}$。因此直接采用参考电压的降压法不妥,这样提出了用参考电压的匹配下降法解决这一问题:第一步,首先在正常的参考电压下进行 A/D 转换,取 8 位转换结果的前 4 位(可多可少)为输出的高位,同时通过 DAC 将 8 数字信号转换成模拟信号和 V_{in} 相减,即取到低位模拟信号;第二步,将这一低位模拟信号送到原 ADC,同时将参考电压匹配下降到 5V/16 = 321.5 mV,此时 ADC 的输出得到低 8 位的数字信号,而后将这两部分的结果结合起来得到 12 位的数字信号。这样分辨率由原来的 8 位提高到了 12 位,这一方法用硬件实现比较复杂,但是,采用单片机实现则复杂程度会大大降低。

3. 微处理器控制下的粗精结合法

这一方法是利用已有的 8 位 ADC 和 DAC,第一阶段是粗化量化。首先在 CPU 的控制下给 DAC 赋值(补偿信号),使其接近与输入信号 V_{in},此时作为转换后的高 8 位输出 M,同时再把数字输出进行 D/A 转换,恢复成模拟电压。第二阶段是进一步的细化量化。由差动放大器对输入信号 V_{in} 和 DAC 输入的补偿信号 V_f 进行差动放大,放大后信号输入 ADC 进行低 8 位的 A/D 转换输出数字量 E。整个过程均在 CPU 控制之下,CPU 根据 ADC 所得到的结果控制 DAC 提供补偿信号,并且 CPU 可以根据提供给 DAC 的补偿值和 ADC 所得到的结果,理论上通过计算可以得到 16 位的数字信号。原理如图 2.12－2 所示。

这样理论上最后的数字输出为:

$$D = M + E/A$$

其中 A 为放大器的放大倍数。实际上,由于各种噪声的存在,使得最终的转换结果达不到 16 位,这样我们还需进行一定的误差纠正措施,利用一个高精度的 16 位 DAC 进行校正,将每一数字位对应的误差计算出来,由 CPU 统一管理,当 CPU 控制 16 位转换结果时,将误差从转换结果中去除,这样可以将分辨率接近 16 位。

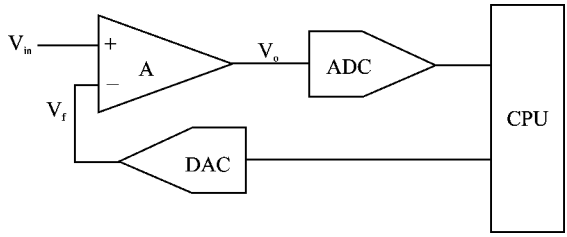


图 2.12－2

二、提高 ADC 分辨率的软件方法

噪声是限制分辨率提高的主要因素,如果能将噪声有效地得到抑制,则分辨率在理论上是可以提高的。下面介绍一种消除噪声提高分辨率的新方法。

利用大量数据的平均值,从极快的转换器噪声中找回失去的分辨率,这种方法不是在任何情况下都适用的。关键在于噪声的性质,如果噪声对称的叠加在信号上,而且数据信号与噪声信号之和是在 ADC 的电压和频率范围之内,此时利用平均值的方法可以将噪声去除。然而,要在实际的 ADC 中找到周期性交流噪声的注入点,在求平均值和数字化的输出上,不产生直流误差是很困难的,因而不能保证电源噪声和 EMI 能够成功的用平均值的方法得到消除,最好还是使用滤波和屏蔽的办法。对于 ADC 自身产生的噪声,用求平均值的方法只能获得人为的结果,处理后的信号可能变得很清晰,但这种人为的产物与输入信号没有人为关系,这样可以看出利用求大量数据平均值的方法不能去除所有的噪声。针对最常见最严重的串模干扰,即来自电网的工频干扰,或是为提高 ADC 分辨率而加入的被测信号的正弦扰动,用软件的方法即可提高 ADC 的分辨率,这一方法称之为平均求和算法。

ADC 通过采样将连续波形 Y'(t) 信号转换为时序信号 Y'(n),其输出信号的量化误差为:

$$e(n) = Y(n) - Y'(n)$$

称为 ADC 的量化误差。对于动态范围为 2D(峰—峰值)、转换位数为 m 位的 ADC 来说,一个量化单位对应的模拟量大小为:

$$q = \frac{2D}{2^m} = 2^{-(m-1)}D$$

把被测量 y'(n) = A 看作是一随机变量,则 y'(n) 是平稳非遍历的,因而采样后的量化误差 e(n) 也是平稳非遍历的随机信号,而且 e(n) 满足均匀分布,即

$$E[e(n)] = 0 \quad D[e(n)] = q^2/12 \tag{2}$$

但其数学期望 E[e(n)]0 与各次采样的平均值 $\frac{1}{N}\sum_{n=0}^{N-1} e(n)$ 之间并无确定关系,即 $\frac{1}{N}\sum_{n=0}^{N-1} e(n)$ 未必趋于 0,所以此时不能用统计算法减小误差。

今设被测信号 A 被一随机相位正弦波污染:

$$y'(n) = A + B\sin\left(\frac{2\pi}{N}n + \alpha\right)$$

其中 α 是正弦波的初相角,也是一个随机变量,并且在此前提下 y'(n) 是平稳遍历的随机信号,相应的量化误差 e(n) 也是平稳遍历的随机信号,并满足

$$E[y'(n)] = \lim_{N\to}\frac{1}{N}\sum_{n=0}^{N-1} y'(n)$$

$$E[e(n)] = \lim_{N\to}\frac{1}{N}\sum_{n=0}^{N-1} e(n)$$

当 N 相当大时,上式可写为:

$$E[y'(n)] \approx \frac{1}{N}\sum_{n=0}^{N-1} y'(n)$$

$$E[e(n)] \approx \frac{1}{N}\sum_{n=0}^{N-1} e(n)$$

由于在正弦的一个周期中 $\sum_{n=0}^{N-1}\sin\left(\frac{2\pi}{N}n+\alpha\right)=0$,故而:

$$E[y'(n)] \approx \frac{1}{N}\sum_{n=0}^{N-1} y'(n)$$

$$= \frac{1}{N}\sum_{n=0}^{N-1}\left[A + B\sin\left(\frac{2\pi}{N}n+\alpha\right)\right] = A$$

而 $E[e(n)]=0$,从而有

$$\frac{1}{N}\sum_{n=0}^{N-1} y(n) = \frac{1}{N}\sum_{n=0}^{N-1}\left[A + B\sin\left(\frac{2\pi}{N}n+\alpha\right) + e(n)\right] \approx A$$

这说明在随机相应信号的干扰下,多次采样的平均值收敛于被测量 A,量化误差得到了抑制。

串模干扰是工业工程中常见的干扰信号,所以上述结论亦适用于串模干扰信号,此时相应的量化误差的方差为:

$$D[e(n)] = \frac{1}{N^2}NA\cdot NA + E\left\{\frac{1}{N^2}\sum_{n=0}^{N-1}\sum_{n=0}^{N-1} e(n)\,e(k)\right\} - A^2$$

$$= E\left\{\frac{1}{N^2}\sum_{n=0}^{N-1} e^2(n) + \frac{1}{N^2}\sum_{n=0}^{N-1}\sum_{n=0}^{N-1} e(n)\,e(k)\right\}$$

$$= \frac{1}{N^2}\sum_{n=0}^{N-1} E[e^2(n)] + \frac{1}{N^2}\sum_{n=0}^{N-1}\sum_{n=0}^{N-1} E[e(n)\,e(k)]$$

$$= \frac{1}{N^2}\cdot N\cdot D[e(n)] = D[e(n)] \quad (n\neq k) \tag{3}$$

由于量化误差为不相关的随机变量,因此 $E[e(n)\,e(k)]=0$ $(n\neq k)$,比较式 (2) 和式 (3),但存在串模干扰时量化误差减小到 $\frac{1}{N}D[e(n)]$。若令 $N=1024=2^{10}$,$m=8\text{bit}$,则此时 $\frac{1}{N}D[e(n)]=\frac{q/2^5}{12}$,即采用 8 位的 ADC,利用 1024 个采样数据抑制串模干扰,其测量结果可获得相当于 $8+5=13$ 位的 ADC 可达到的分辨率,这样通过抑制串模干扰的平均求和算法提高了 ADC 的分辨率。

四、小 结

以上所提到的硬件和软件的方法可以在现有低分辨率 ADC 的基础上节约成本、提高分辨率，因而可应用于高精度、高分辨率的转换系统中。若将软件和硬件结合起来，则能够更好的提高分辨率满足用户的需要。

参 考 文 献

1 高光天主编. 模数转换器应用技术. 北京 科学出版社，2001

2 潘荣贵等. 提高模数转换器精度的措施综述. 现代电子工程，1998. 4

3 李维国. 抑制工频串模干扰的平均求和算法对 ADC 分辨率的提高作用. 山东建筑工程学院学报，1996. 11

4 Bertocco M，Narcuzzi C. Accuracy performances of ADC parameter estimators，Conference Record－IEEE Instrumentation and Measurement Technology Conference V2 May 19 - 21 1997 Sponsored by：IEEE IEEE p 985 - 988 1997

5 Bader，Bonnie，Multiplexer－scaling ADC input for higher accuracy and resolution，Electronic Design 44 24B Nov 18 1996 p 79

选自《计量技术》月刊，2002 第 11 期

2.13 智能温度传感器的发展趋势

石家庄河北科技大学电子信息工程系 (050054) 沙占友

现代信息技术的三大基础是信息采集 (即传感器技术)、信息传输 (通信技术) 和信息处理 (计算机技术)。传感器属于信息技术的前沿尖端产品,尤其是温度传感器被广泛用于工农业生产、科学研究和生活等领域,数量高居各种传感器之首。近百年来,温度传感器的发展大致经历了以下三个阶段:(1) 传统的分立式温度传感器 (含敏感元件);(2) 模拟集成温度传感器/控制器;(3) 智能温度传感器。目前,国际上新型温度传感器正从模拟式向数字式、由集成化向智能化、网络化的方向发展。

一、集成温度传感器的产品分类

1. 模拟集成温度传感器

集成传感器是采用硅半导体集成工艺而制成的,因此亦称硅传感器或单片集成温度传感器。模拟集成温度传感器是在 20 世纪 80 年代问世的。它是将温度传感器集成在一个芯片上、可完成温度测量及模拟信号输出功能的专用 IC。模拟集成温度传感器的主要特点是功能单一 (仅测量温度)、测温误差小、价格低、响应速度快、传输距离远、体积小、微功耗等,适合远距离测温、控温,不需要进行非线性校准,外围电路简单。它是目前在国内外应用最为普遍的一种集成传感器,典型产品有 AD590、AD592、TMP17、LM135 等。

2. 模拟集成温度控制器

模拟集成温度控制器主要包括温控开关、可编程温度控制器,典型产品有 LM56、AD22105 和 MAX6509。某些增强型集成温度控制器 (例如 TC652/653) 中还包含了 A/D 转换器以及固化好的程序,这与智能温度传感器有某些相似之处。但它自成系统,工作时并不受微处理器的控制,这是二者的主要区别。

3. 智能温度传感器

智能温度传感器 (亦称数字温度传感器) 是在 20 世纪 90 年代中期问世的。它是微电子技术、计算机技术和自动测试技术 (ATE) 的结晶。目前,国际上已开发出多种智能温度传感器系列产品。智能温度传感器内部都包含温度传感器、A/D 转换器、信号处理器、存储器 (或寄存器) 和接口电路。有的产品还带多路选择器、中央控制器 (CPU)、随机存取存储器 (RAM) 和只读存储器 (ROM)。智能温度传感器的特点是能输出温度数据及相关的温度控制量,适配各种微控制器 (MCU);并且它是在硬件的基础上通过软件来实现测试功能的,其智能化程度也取决于软件的开发水平。

二、智能温度传感器发展的新趋势

进入 21 世纪后,智能温度传感器正朝着高精度、多功能、总线标准化、高可靠性及安全性、开发虚拟传感器和网络传感器、研制单片测温系统等高科技的方向迅速发展。

1. 提高测温精度和分辨力

在20世纪90年代中期最早推出的智能温度传感器,采用的是8位A/D转换器,其测温精度较低,分辨力只能达到1 。目前,国外已相继推出多种高精度、高分辨力的智能温度传感器,所用的是9~12位A/D转换器,分辨力一般可达0.5~0.062 5 。由美国DALLAS半导体公司新研制的DS1624型高分辨力智能温度传感器,能输出13位二进制数据,其分辨力高达0.031 25 ,测温精度为±0.2 。为了提高多通道智能温度传感器的转换速率,也有的芯片采用高速逐次逼近式A/D转换器。以AD7817型5通道智能温度传感器为例,它对本地传感器、每一路远程传感器的转换时间分别仅为27 μs、9 μs。

2. 增加测试功能

新型智能温度传感器的测试功能也在不断增强。例如,DS1629型单线智能温度传感器增加了实时日历时钟(RTC),使其功能更加完善。DS1624还增加了存储功能,利用芯片内部256字节的E^2PROM存储器,可存储用户的短信息。另外,智能温度传感器正从单通道向多通道的方向发展,这就为研制和开发多路温度测控系统创造了良好条件。

智能温度传感器都具有多种工作模式可供选择,主要包括单次转换模式、连续转换模式、待机模式,有的还增加了低温极限扩展模式,操作非常简便。对某些智能温度传感器而言,主机(外部微处理器或单片机)还可通过相应的寄存器来设定其A/D转换速率(典型产品为MAX6654)、分辨力及最大转换时间(典型产品为DS1624)。

智能温度控制器是在智能温度传感器的基础上发展而成的。典型产品有DS1620、DS1623、TCN75、LM76、MAX6625。智能温度控制器适配各种微控制器,构成智能化温控系统;它们还可以脱离微控制器单独工作,自行构成一个温控仪。

3. 总线技术的标准化与规范化

目前,智能温度传感器的总线技术也实现了标准化、规范化,所采用的总线主要有单线(1-Wire)总线、I^2C总线、SMBus总线和SPI总线。温度传感器作为从机可通过专用总线接口与主机进行通信。

4. 可靠性及安全性设计

传统的A/D转换器大多采用积分式或逐次比较式转换技术,其噪声容限低,抑制混叠噪声及量化噪声的能力比较差。新型智能温度传感器(例如TMP03/04、LM74、LM83)普遍采用了高性能的∑-Δ式A/D转换器。它能以很高的采样速率和很低的采样分辨力将模拟信号转换成数字信号,再利用过采样、噪声整形和数字滤波技术,来提高有效分辨力。∑-Δ式A/D转换器不仅能滤除量化噪声,而且对外围元件的精度要求低;由于采用了数字反馈方式,因此比较器的失调电压及零点漂移都不会影响温度的转换精度。这种智能温度传感器兼有抑制串模干扰能力强、分辨力高、线性度好、成本低等优点。

为了避免在温控系统受到噪声干扰时产生误动作,在AD7416/7417/7817、LM75/76、MAX6625/6626等智能温度传感器的内部,都设置了一个可编程的"故障排队(fault queue)"计数器,专用于设定允许被测温度值超过上、下限的次数。仅当被测温度连续超过上限或低于下限的次数达到或超过所设定的次数n(n=1~4)时,才能触发中断端。若故障次数不满足上述条件或故障不是连续发生的,故障计数器就复位而不会触发中断端。这意味着假定n=3时,那么偶然受到一次或两次噪声干扰,都不会影响温控系统的正常工作。

LM76型智能温度传感器增加了温度窗口比较器,非常适合设计一个符合ACPI(Advanced

Configuration and Power Interface,即“先进配置与电源接口”)规范的温控系统。这种系统具有完善的过热保护功能,可用来监控笔记本电脑和服务器中 CPU 及主电路的温度。微处理器最高可承受的工作温度规定为 t_H,台式计算机一般为 75 ,高档笔记本电脑的专用 CPU 可达 100 。一旦 CPU 或主电路的温度超出所设定的上、下限时,INT 端立即使主机产生中断,再通过电源控制器发出信号,迅速将主电源关断起到保护作用。此外,当温度超过 CPU 的极限温度时,严重超温报警输出端(T_CRIT_A)也能直接关断主电源,并且该端还可通过独立的硬件关断电路来切断主电源,以防主电源控制失灵。上述三重安全性保护措施已成为国际上设计温控系统的新观念。

为防止因人体静电放电(ESD)而损坏芯片,一些智能温度传感器还增加了 ESD 保护电路,一般可承受 1 000 ~ 4 000 V 的静电放电电压。通常是将人体等效于由 100 pF 电容和 1.2 kΩ电阻串联而成的电路模型,当人体放电时,TCN75 型智能温度传感器的串行接口端、中断/比较器信号输出端和地址输入端均可承受 1 000 V 的静电放电电压。LM83 型智能温度传感器则可承受 4 000 V 的静电放电电压。

最新开发的智能温度传感器(例如 MAX6654、LM83)还增加了传感器故障检测功能,能自动检测外部晶体管温度传感器(亦称远程传感器)的开路或短路故障。MAX6654 还具有选择“寄生阻抗抵消”(Parasitic Resistance Cancellation,英文缩写为 PRC)模式,能抵消远程传感器引线阻抗所引起的测温误差,即使引线阻抗达到 100 Ω,也不会影响测量精度。远程传感器引线可采用普通双绞线或者带屏蔽层的双绞线。

5. 虚拟温度传感器和网络温度传感器

(1) 虚拟传感器

虚拟传感器是基于传感器硬件和计算机平台,并通过软件开发而成的。利用软件可完成传感器的标定及校准,以实现最佳性能指标。最近,美国 B&K 公司已开发出一种基于软件设置的 TEDS 型虚拟传感器,其主要特点是每只传感器都有惟一的产品序列号并且附带一张软盘,软盘上存储着对该传感器进行标定的有关数据。使用时,传感器通过数据采集器接至计算机,首先从计算机输入该传感器的产品序列号,再从软盘上读出有关数据,然后自动完成对传感器的检查、传感器参数的读取、传感器设置和记录工作。

(2) 网络温度传感器

网络温度传感器是包含数字传感器、网络接口和处理单元的新一代智能传感器。数字传感器首先将被测温度转换成数字量,再送给微控制器作数据处理。最后将测量结果传输给网络,以便实现各传感器之间、传感器与执行器之间、传感器与系统之间的数据交换及资源共享,在更换传感器时无须进行标定和校准,可做到“即插即用(Plug & Play)”,这样就极大地方便了用户。

6. 单片测温系统

单片系统(System On Chip)是 21 世纪一项高新科技产品。它是在芯片上集成一个系统或子系统,其集成度将高达 $10^8 \sim 10^9$ 元件/片,这将给 IC 产业及 IC 应用带来划时代的进步。半导体工业协会(SIA)对单片系统集成所作的预测见表 2.13 - 1。目前,国际上一些著名的 IC 厂家已开始研制单片测温系统,相信在不久的将来即可面市。

表 2.13－1 单片系统集成的发展预测

年 份	2001	2004	2007	2010
最小线宽/μm	0.18	0.13	0.1	0.07
包含晶体管数量/片	1.3×10^8	2.5×10^8	5×10^8	9×10^8
成本/(晶体管/毫美分)	0.2	0.1	0.05	0.02
芯片尺寸/mm^2	750	900	1 100	1 400
电源电压/V	1.8	1.5	1.2	0.9
芯片 I/O 数	2 000	2 600	3 600	4 800

参 考 文 献

1 美国 ADI、NSC、HARRIS、Telcom、MAXIM、DALLAS 等公司产品资料,1998～2000

2 沙占友. A Study of the Control System with Intelligent Temperature Sensors. ICEMI'第四届国际电子测量学术会议论文集,电子测量与仪器学报第 13 卷,1999.8,ISTP 收录

3 沙占友. 智能化温度传感器的原理与应用. 集成电路通信,2001 (3)

选自《电子技术应用》月刊,2002 年第 5 期

2.14 温度传感器的选择策略

陈永信

温度传感技术被广泛应用于消费类电子产品、玩具、家用电子产品、工业测量以及最为重要的个人计算机应用中。传统上热敏电阻是最常见的温度传感元件,而 IC 温度传感器的厂商也在同样的应用领域中推出了 IC 传感器。本文通过对美国国家半导体的一系列 IC 温度传感器的介绍和相对的热敏电阻性能测试的比较,为选择温度传感产品的设计人员提供了指导。

一、热敏电阻的传统优势

1. 价 格

毫无疑问,在大多数情况下,就单个组件进行比较,热敏电阻的价格低于 IC 温度传感器。但为了将热敏电阻的阻值转换成电压值,需要一个精度为 1% 的上拉电阻,以便获得准确的读数。如果需要以数字方式读出热敏电阻的值,必须再加用带有不同接口 (如 I^2C 或 SPI) 的模数转换器 (ADC)。对于模拟输出的 IC 温度传感器,单独一颗芯片即可读出电压值,不需外加器件。而对于需要数字化的设计来说,我们能很容易地找到具备不同输出接口的单芯片数字 IC 温度传感器。以整体系统价格来说,IC 温度传感器并不一定较高。

另一方面,随着工艺的改进,美国国家半导体公司最近宣布了全球最低成本的模拟温度传感器 LM19,其价格可与热敏电阻相媲美。

2. 各种各样的封装

如果就传感器无法安装在电路板上的情况而论,热敏电阻具有优势,但仅限于这种情况。如果传感器需要安装在电路板上,则没有差别,甚至当采用具有良好导热性能的 LLP 封装的 IC 温度传感器,如 LM20 或 LM74 时,则能获得更准确的读数。

3. 精 度

在这点上的争论取决于它的用途。在小范围内测量温度,例如体温计,热敏电阻具备输出微调能力,配合精确的外加线路,可以得到精确的读数。由于安装在电路板上受到的限制,IC 温度传感器可能因无法直接碰触测量物,精度会有所影响。但是如果在一个允许在电路板上测量温度,而且范围较大的应用领域时,IC 温度传感器比热敏电阻更精确。

另一方面,使用热敏电阻时为了达到一致的精度,需要对每批热敏电阻或每颗热敏电阻进行调校。然而 IC 温度传感器在出厂时即完成测试,保持了系统生产时的稳定性。

二、温度读取误差的来源

1. 读取网络和功耗

热敏电阻是一个电阻随着温度变化的器件。通常设计人员需要建立一个电阻网络以把电阻变化转化成电压变化。

如图 2.14-1 所示,位于上部的电阻是读取数据的第一个误差来源。如果你想读取一个

准确的结果，那么，上部电阻的精度必须不低于 1%。同样，上部电阻的阻值将影响整个网络的线性度和功耗。

与热敏电阻相比较，IC 温度传感器更易于使用。IC 温度传感器中有三个引脚，分别是 V_{cc}、GND 和 V_{out}。一般而言，V_{out} 与温度呈线性关系，而 CMOS 工艺能最大程度地降低功耗。再者，IC 温度传感器一般都可承受一定范围的 V_{cc} 变化而不影响输出值。例如图 2.14－2 的 LM20 可稳定地在 2.4～5.5 V 的 V_{cc} 下工作。

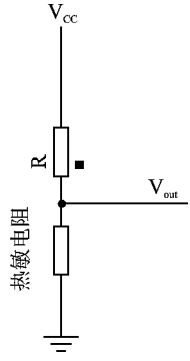


图 2.14－1 将电阻变化转化成电压变化的电阻网络

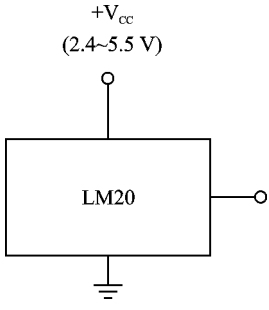


图 2.14－2 LM20 IC 温度传感器

2. 线性度

如读取网络中所示，线性度是造成温度读取误差的另一个因素，特别是当温度读数的范围较宽时。因而热敏电阻在完成数字化过程后总是需要一个查阅表，或增加一些模拟电路，以对线性度进行补偿，这再次增加了成本。

IC 温度传感器以线性方式指示温度，而且不需要进行线性度补偿。

图 2.14－3 是热敏电阻与 IC 温度传感器响应性能的示例。

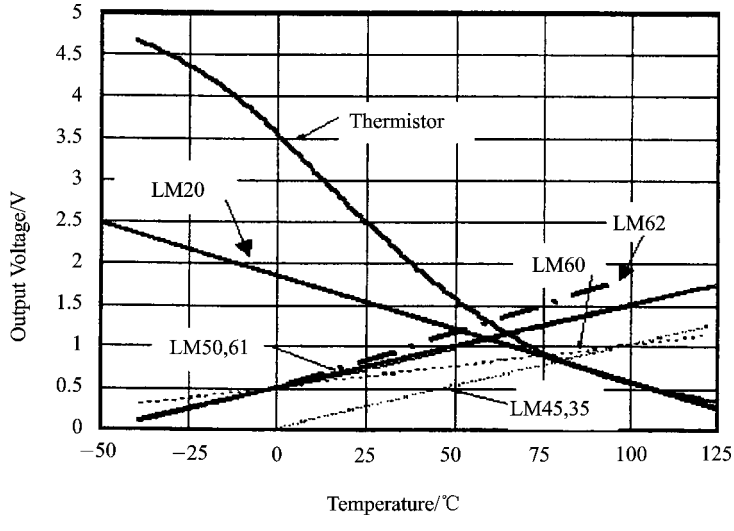


图 2.14－3 热敏电阻与 IC 温度传感器的输出电压–温度曲线图

3. 数字化误差

为了读出数字世界中的温度值,我们需要用模数转换器将模拟输出转换为数字量。在这一过程中,模数转换器的线性误差、量化误差、偏移误差、PSRR 误差与温度的关系也将造成整体系统的温度读取误差。各种误差与温度的关系如图 2.14－4 所示。

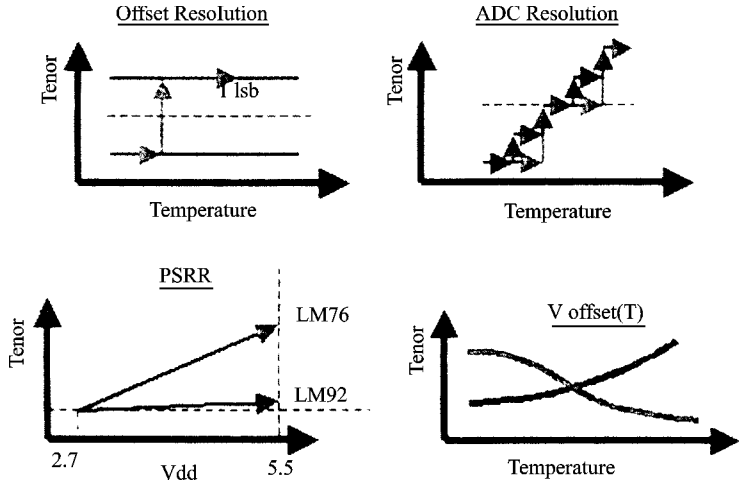


图 2.14－4 各种误差与温度的关系图

因此,如果设计人员认为最终系统的全部参数和误差都依赖于多个因素,那么设计工作将大为复杂。我们可能还需要在生产现场进行系统校准。

IC 数字温度传感器将传感器和数字化功能集成在一块芯片内。这种芯片的数字输出级在最后出厂前都经过测试,以排除一切误差来源。整块芯片的精度得到了保证,因而非常容易即行采用。例如,LM92 的技术规格为 30 时的精度为 0.33 。这时,你可以立即从 I^2C 总线获得准确的数字读数而无须校准!

三、热敏电阻与 LM20 的对比实验

1. 设 置

为比较热敏电阻和 IC 温度传感器 LM20 的性能,特别进行了一项实验。两者的设置尽可能做到相同,以便进行直接比较。实验电路如图 2.14－5 所示。

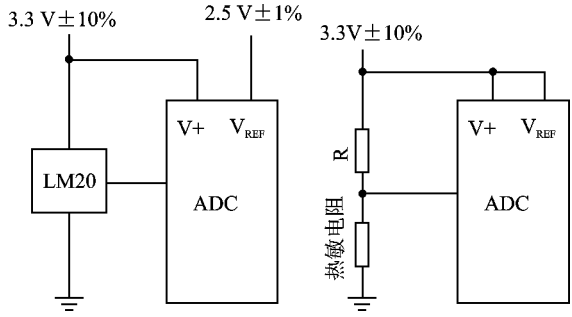


图 2.14－5 IC 温度传感器 LM20 与热敏电阻性能比较的实验电路

LM20 是美国国家半导体公司推出的一款新型模拟温度传感器，采用 SC－70 或 Micro－SMD 封装，体积小巧，尤其适用于手机。另外，采用 TO－92 封装的 LM19 也具有相同的性能。在此使用的热敏电阻是 Murata NTH5G10P/16P33B103F。

2. 热敏电阻的非线性及功耗

首先，我们需要确定热敏电阻设置中的上拉电阻。如前所述，电阻值的选取可能产生不同的线性度和功耗。下面是选取阻值分别为 4.7 kΩ 和 97.6 kΩ 时的结果。

如图 2.14－6 所示，在 97.6 kΩ 的上拉电阻下，我们可以观测到严重的非线性情况。而且温度较高范围内斜率的下降将降低 ADC 的分辨率。微小的电压变化即代表大幅度的温度差异而无从观察。

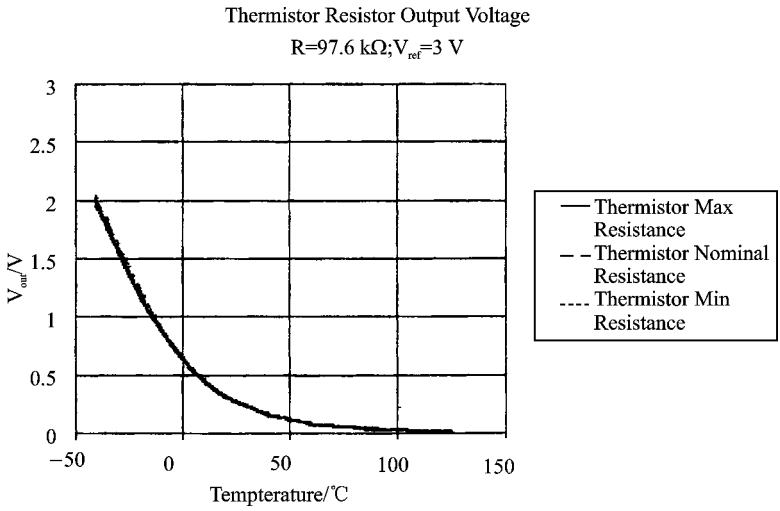


图 2.14－6 上拉电阻为 97.6 kΩ 时，输出电压—温度曲线图

如图 2.14－7 所示，随着上拉电阻减至 4.7 kΩ，热敏电阻设置确实产生了较好的线性度，但同时也增大了功耗。功耗的增加导致热敏电阻网络的自热，从而引入了读取误差。而且系统连续使用时间越长，自热效应造成的误差将越严重。

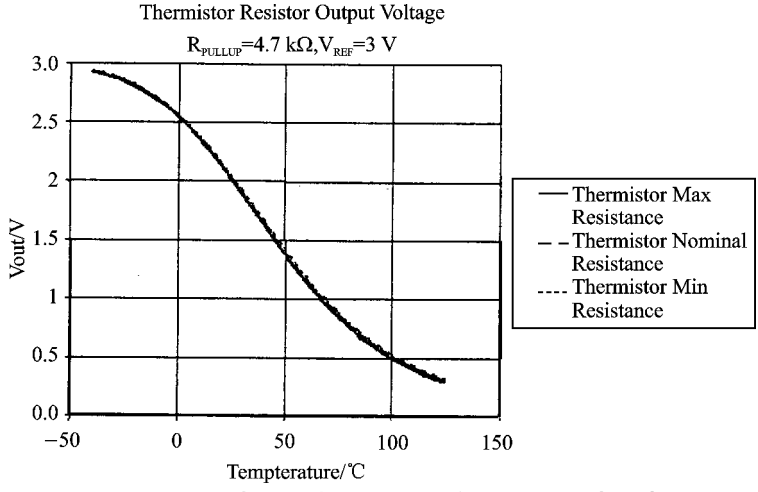


图 2.14－7 上拉电阻为 47 kΩ 时，输出电压—温度曲线图

迄今为止,设计中出现的热敏电阻的非线性与自热问题仍不能同时得到解决。但用 IC 温度传感器就能真正地同时获得高线性度与低自热的性能。

3. 实验结果

在这些设置条件下,热敏电阻与 IC 温度传感器的特性图如图 2.14－8 所示。

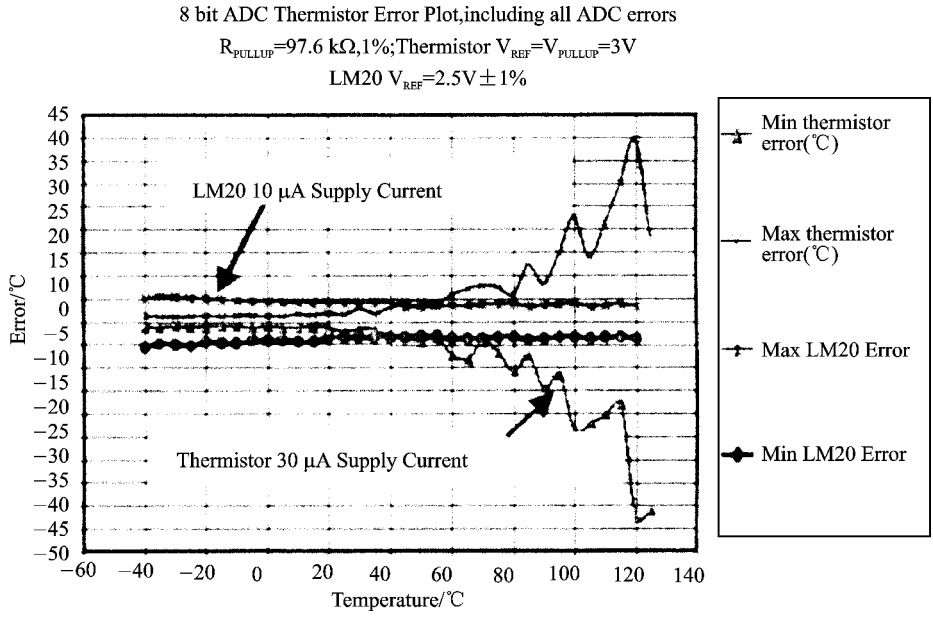


图 2.14－8 热敏电阻与 IC 温度传感器的特性曲线图

从图中可以明显地看出 在较低温度范围内,两种设置都具有大致相同的误差,甚至热敏电阻的读数精度更高一些。但在 50 以上的较高温度范围内,热敏电阻的读数精度降低,温度越高,精度越低。

在大多数情况下,温度读数被用于温度过高时的系统保护,这时热敏电阻发出假警报的机会将会大增。

另一方面,IC 温度传感器消耗的电流仅相当于热敏电阻的三分之一(10 μA: 30 μA)。

四、如何作出正确的选择

根据以上结果,下面提供了一些判断准则以供设计人员正确地选择 IC 温度传感器。

- 当测量的温度范围较广时;
- 当系统的总成本非常重要时;
- 当设计时间必须最短时;
- 当空间非常宝贵时;
- 当需要较低的供电电流时。

热敏电阻的选用准则为:

- 系统精度不很重要时;
- 感测元件必须远离 PCB 时;
- 系统无需外加器件来读取温度值时。

2.15 单线数字温度传感器 DS18B20 数据校验与纠错

齐齐哈尔大学信息学院 (161006) 岳云峰 马春光
哈尔滨信息产业部电子第四十九研究所 (150001) 李怀盛 李晓青

一、前 言

DS18B20 单线数字温度传感器是新一代温度传感器。它具有微型化、低功耗的特征,可直接将测得的结果以串行数字信号输出,易与微处理器连接。由于片内 ROM 中有惟一的 64 位序列号做为地址,所以可将多片 DS18B20 以总线的方式连接在一条线上,这为需要多点同时进行测温的场合带来了很大的方便。BS18B20 的测温范围为 - 55 ~125 ,输出 9 ~12 位可通过编程设定。 - 10 到85 时精度可达0.5 ,温度转换时间从93.75 ms 到750 ms,这是可以满足大多数应用场合的。DS18B20 有 5 条关于 ROM 的操作指令、6 条关于存储器的操作指令。

在进行多点测温时,敏感元件与数据采集系统一般要有一定的距离,不可避免地要遇到电磁干扰、信号衰减等问题,使数据发生错误。如果在数据的传输过程中系统具有一定的容错能力,在纠错范围内,就可以对错误的数据进行纠正,提高抗干扰能力和加大传输距离;当错误超出纠错范围时,也可以识别出错误的数据进行重新采集,从而提高了所采集的数据的可信度。检验与纠错可以用硬件或软件进行处理。DS18B20 在设计时已经为用户提供了用于检验与纠错的循环冗余校验码 (cyclic redundancy code, CRC)。然而关于对 DS18B20 中数据的检验与纠错的文章却较为少见。下边将就用软件对 DS18B20 中数据的检验与纠错进行详细的讨论,并给出了用查表法进行校验及纠错的算法以及实现这一算法的过程。

二、DS18B20 的工作原理与测温系统的组成

1. DS18B20 的工作原理

DS18B20 的内部结构如图 2.15 - 1 所示。它是由温度敏感元件、高温限值触发器 (TH)、低温限值触发器 (TL)、数据缓冲存储器、存储控制器、配置寄存器、64 位激光 ROM 与单线数据口及 8 位 CRC 发生器等组成。

在图 2.15 - 1 中两个二极管与一个电容器组成寄生电源,通过电源检测电路来决定 DS18B20 的供电方式。64 位激光 ROM 的数据结构如图 2.15 - 2 (a) 所示。DS18B20 的产品系列号是一个固定值 (28H);48 位的元件序号为一不重复的惟一编号,在进行多点单总线测温通信时,做为元件的地址与系统所发出的匹配命令中的地址相匹配[1],来选择测温点;64 位 ROM 中的 8 位 CRC 码为循环冗余校验码,它是由产品序列号、元件序号通过 CRC 发生器产生的。

由温度存储器的低位字节、高位字节,低温报警触发器 TL,高温报警触发器 TH,配置寄存器和 CRC 字节组成了 DS18B20 的数据存储器,共 9 个字节,结构如图 2.15 - 2 (b) 所示。其中

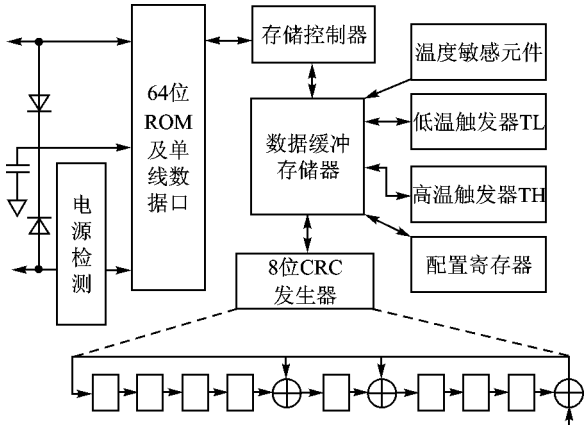


图 2.15－1 DS18B20 的内部结构

温度存储器的低位字节、高位字节是以补码形式存放，两个字节所对应的 16 位二进制数中最低 4 位是温度值的小数部分，最高 5 位是符号扩展，0 表示正数，1 表示负数值，其余为整数部分。例如，16 进制温度值 018CH 对应的二进制数为 0000000110001100，温度值是 24.75 ；FF5EH 对应的二进制数为 1111111101011110，温度值 - 10.125 。图 2.15－2 (b) 中的 CRC 是通过 CRC 发生器产生的。CRC 发生器产生的逻辑电路是由移位寄存器和异或门组成，也称为除法逻辑电路。CRC 发生器产生的逻辑电路对应的表达式是 $X^8 + X^5 + X^4 + 1$ (对应的二进制数为 100110001)，称为生成多项式，记为 g(x)。实际应用中就是通过这种除法逻辑电路对一组数据进行校验与纠错。如果速度允许，也常使用生成多项式 g(x) 通过软件方法去进行校验与纠错[2]。

8 位 CRC 码	48 位 ROM 元件序列号	8 位产品系列码 (28H)

(a) 64 位激光 ROM 序列号数据结构

CRC	保留	保留	保留	配置寄存器	低温限值 TL	高温限值 TH	温度高字节	温度低字节

(b) 9 字节数据存储器结构

图 2.15－2 64 位激光 ROM 与 9 字节存储器的数据结构

2. 多点测温系统的组成

图 2.15－3 为多点测温系统硬件组成原理图。其中使用 AT89C51 单片机的 P1.0 ~ P1.4 共 5 条数据线做 5 路数据采集口。每一路连接 10 个 DS18B20 传感器，使用外电源供电方式，由三芯屏蔽电缆连接。显示部分由 6 位共阳极 LED 数码管组成，最高 2 位显示测温点的地址，其余 4 位显示温度值 (包括一位小数位)。数码管的段驱动直接由 AT89C51 的 P0 口承担，位驱动由 P2 口的 P2.2 ~ P2.7 通过 6 支 PNP 型三极管 9012 完成。为了防止出现死机现象，使用了看门狗专用电路 MAX813L，由 AT89C51 的 P1.5 定时进行触发。

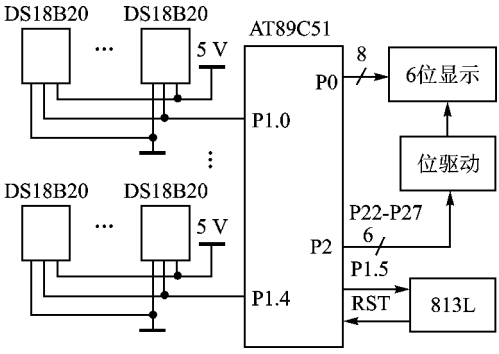


图 2.15－3 多点测温系统的组成

三、用 CRC 码的校验与纠错

CRC 校验的算法与编码的算法是相同的[3]，校验时要将 CRC 作为数据一同进行计算。下边对其进行讨论并给出查表算法。

1. 直接算法

用生成多项式 g(x)直接进行校验的方法是将图 2.15－2(b)中的数据，低字节低位放在前(左)。然后用 g(x)去做异或除法[3]。得到的余数若为 0，则表示数据正确；余数不为 0，则表示数据有错。通过余数便可知道结果是否正确。DS18B20 的 CRC 码是可以纠正一位错误的。

2. 改进算法

直接算法在实际应用中由于要求必须将数据低字节低位放在前，并且要将几个字节做连续的整体处理，所以处理起来显得很不方便。由于循环次数较多而需要的时间也长，下边给出一种改进的算法，可以较方便的进行处理。该算法是对数据进行逐字节处理，处理顺序是由低字节到高字节。

这里首先将 CRC 单元赋 0，取一个字节数据，做异或并将结果存入暂存单元 AY，然后字节左环移。如果 AY 的最低位是 1，则 CRC 与十六进制数 18H 做异或且左移；否则 CRC 只左移。再将 AY 的最低位移入 CRC 的最高位。做完一个字节之后，将 CRC 的结果与下一字节作同样处理直至最后一个字节处理完。具体算法见图 2.15－4。

3. 从改进算法派生的查表算法

从改进算法中看到这种方法是一个字节一个字节进行处理的，由于一个字节的值从 0 到 255 只有 256 个整数，这样利用改进算法的核心算法将 256 个值的结果事先计算出来(表 2.15－1 就是计算的结果)，然后再进行查表。

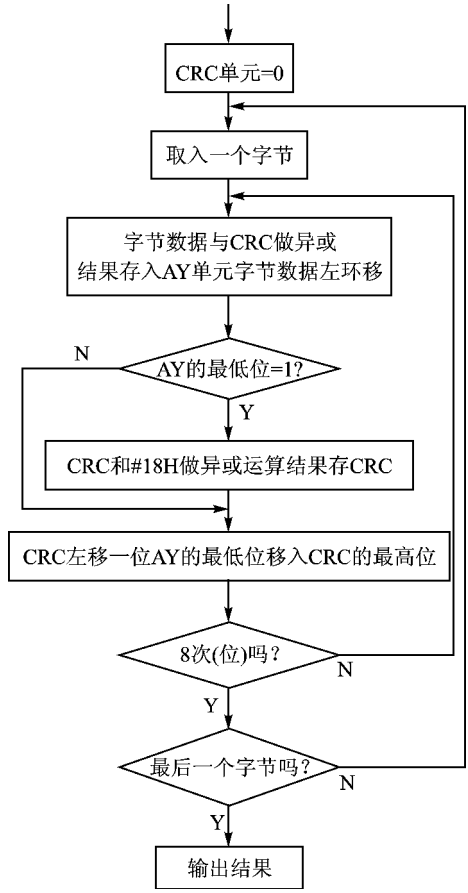


图 2.15－4 改进算法

表 2.15－1 改进算法对 256 个值的计算结果

行	0	1	2	3	4	5	6	7	8	9
0	0	94	188	226	97	63	221	131	194	156
1	126	32	163	253	31	65	157	195	33	127
2	252	162	64	30	92	1	227	189	62	96
3	130	220	35	125	159	193	66	28	254	160
4	225	191	93	3	128	222	60	98	190	224
5	2	92	223	129	99	61	124	34	192	158
6	29	67	161	255	70	24	250	164	39	121
7	155	197	132	218	56	102	229	187	89	7
8	219	133	103	57	186	228	6	88	25	71
9	165	251	120	38	196	154	101	59	217	135
10	4	90	184	230	167	249	27	69	198	152
11	122	36	248	166	68	26	153	199	37	123
12	58	100	134	216	91	5	231	185	140	210
13	48	110	237	179	81	15	78	16	242	172
14	47	113	147	205	17	79	173	243	112	46
15	204	146	211	141	111	49	178	236	14	80

续表 2.15－1

行	0	1	2	3	4	5	6	7	8	9
16	175	241	19	77	206	144	114	44	109	51
17	209	143	12	82	176	238	50	108	142	208
18	83	13	239	177	240	174	76	18	145	207
19	45	115	202	148	118	40	171	245	23	73
20	8	86	180	234	105	55	213	139	87	9
21	235	181	54	104	138	212	149	203	41	119
22	244	170	72	22	233	183	85	11	135	214
23	52	106	43	117	151	201	74	20	246	168
24	116	42	200	150	21	75	169	247	182	232
25	10	84	215	137	107	53				

该方法首先将 CRC 单元赋 0，然后将数据字节与 CRC 做异或，再以异或的结果做为序号查得的值送入 CRC 单元，CRC 再与下一字节数数据做异或，如此下去，直到最后一字节。这时若结果为 0，则表示数据正确无误 若不等于 0，则表示数据有误，可根据结果进行纠错或重新采集数据。由于这时只循环 9 次，显然这种方法要比前两种方法快。

为了说明问题，取一组实际采集的数据（见表 2.15－2）。

表 2.15－2 数据采集样本

位 号	64 位序列号（16 进制）	采集的数据（16 进制）	温度/
1	69 00 00 00 16 59 29 28	76 10 08 FF 7F 46 4B 01 88	24.500 0
2	55 00 00 00 16 4C 06 28	2E 10 04 FF 7F 46 4B 01 8C	24.750 0
3	13 00 00 00 16 41 D4 28	F8 10 02 FF 7F 46 4B 02 09	32.562 5
4	59 00 00 00 16 4F 47 28	02 10 02 FF 7F 46 4B 01 8E	25.875 0
5	83 00 00 00 15 F4 D3 28	0B 10 0B FF 7F46 4B 01 95	25.312 5
6	83 00 00 00 16 57 37 28	AA 10 02 FF 7F 46 4B 01 9E	25.875 0
7	1C 00 00 00 16 52 8A 28	8A 10 0B FF 7F 46 4B 02 05	32.312 5
8	0C 00 00 00 16 47 E6 28	C7 10 01 FF 7F 46 4B 01 B0	27.000 0
9	0A 00 00 00 16 54 71 28	13 10 01 FF 7F 46 4B 00 2F	2.937 5
10	EB 00 00 00 16 49 17 28	AD 10 08 FF 7F 46 4B 01 48	20.500 0

表 2.15－2 中取第 2 组数据放在数组中，数据中 A（0）＝8CH，A（1）＝01H，A（2）＝4BH，A（3）＝46H，A（4）＝7FH，A（5）＝FFH，A（6）＝04H，A（7）＝10H，A（8）＝2EH。其中 A（0）、A（1）分别为温度值的低位和高位，A（8）为 CRC。它们对应的十进制数为 140，1，75，70，127，255，4，16，46。图 2.15－5 为该算法的实际处理过程。

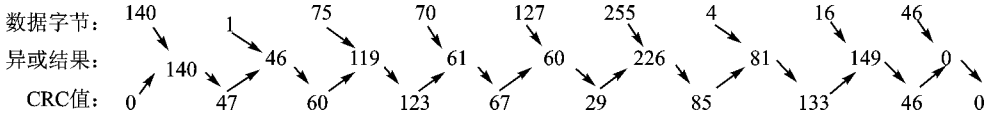


图 2.15－5 查表算法

在图 2.15-5 中 140 XOR 0=140 从表 1 查得 47,1 XOR47=46 从表 2.15-1 查得 60,这样一直到最后一个字节。

4. 纠 错

如果利用上述的算法最后得到的结果不为 0,则意味着数据有错。DS18B20 的编码是可以纠正一位错误的,当错误多于一个时就要重新采集数据,本文给出一种利用查表进行纠错的方法。该方法是首先计算出一位错误代码的样图,如表 2.15-3 所列。

表 2.15-3 一位错误代码样图

下标	D7	D6	D5	D4	D3	D2	D1	D0
0	173	218	109	186	93	162	81	164
1	146	73	168	84	42	21	134	67
2	151	199	239	251	241	255	122	61
3	234	117	182	91	161	220	110	55
4	203	233	248	124	62	31	131	205
5	217	224	112	56	28	14	7	143
6	98	49	148	74	37	158	79	171
7	47	155	193	236	118	59	145	196
8	140	70	35	157	194	97	188	94

然后根据查表得到这个余数在表 2.15-3 中的位置值(这里设为 N,N 的值从 0 到 71),通过 N 的值就可以计算出错误在数据中的位置。计算方法是,将 N 被 8 除,商的整数部分为错误字节的位置,用 7 减去余数则得到错误位在该字节中的位号,只须对错误位取反即可纠正错误。例如上例中的 A(1)变成了 11H(00010001),利用查表法得到的余数是 84,从表 2.15-3 中查得 N=11,用 11 除以 8 得 1 余 3,则说明 A(1)的 D4 出现错误。

四、结束语

由于使用了校验与纠错方法,增加了数据的传输距离与可信度。在没使用校验与纠错方法时,线路大于 20 m 时,显示的温度值常出现大幅度的跳动,在使用校验与纠错方法后,线路在 50 m 时仍十分稳定。文中给出的查表算法使得编程变得更简单快捷。在纠错处理中也使用了查表的方法加快了处理速度,实际应用效果良好。

参 考 文 献

1 何立民. 单片机应用技术选编(8)[M]. 北京:北京航空航天大学出版社,2000
2 樊昌信. 通信原理[M]. 北京:国防工业出版社,2001
3 王新梅. 纠错码与差错控制[M]. 北京:人民邮电出版社,1989

选自《传感器技术》月刊,2002 年第 7 期

2.16 TMP03/04 型数字温度传感器的工作原理

河北科技大学 沙占友 王晓君 孟志永

TMP03 和 TMP04 是美国模拟器件公司(AD)生产的串行比率输出式数字温度传感器,适配 80C31、80C51 型单片机(μC)或数字信号处理器(DSP)构成测温系统。二者主要区别是 TMP03 为集电极开路输出,而 TMP04 为互补型 MOS 场效应管输出,其输出电平与 CMOS/TTL 电路兼容。TMP03/04 既可以检测温度,也可以通过单片机实现温度控制功能,适用于远程温度检测、微机或电子设备的温度监视器及工业过程控制等领域。

一、TMP03/04 性能特点

⑴ TMP03/04 带串行接口,其输出为经过调制的串行数据。解码后高、低电平持续时间的比率(t_1/t_2)与温度成比例关系。利用微处理器的定时/计数器接口,很容易计算出摄氏温度或华氏温度值。

⑵ 芯片内部有一个 Σ-Δ 数字调制器,内含输入采样器、模拟求和器、积分器、比较器和 1 位数/模转换器(DAC)。Σ-Δ 调制器配上数字滤波器后,即构成了 Σ-Δ 式 A/D 转换器。它具有分辨力高、线性度好、成本低、抑制混叠噪声和量化噪声的能力强等显著优点,特别适用于微传感器系统。

⑶ 属于三端器件,其外围电路非常简单,数据输出端(D_{OUT})能直接连到单片机的输入口。若经过光耦合器隔离后,还适合检测远程温度。测温范围一般为 -25 ~ +100 ℃,测温精度为 ±1.5 ℃(典型值)。使用时不需要校准。

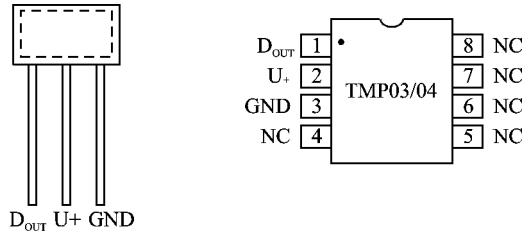
⑷ 低电压供电,微功耗。电源电压范围是 +4.5 ~ +7 V。采用 +5 V 供电时,电源电流不超过 1.3 mA。最大功耗仅为 6.5 mW。

⑸ TMP03 和 TMP04 的数字输出的电路结构不同。TMP03 的输出级采用一只集电极开路的 NPN 型晶体管作为大电流驱动器,其输出电流可达 5 mA;TMP04 的输出级则采用互补型 MOSFET 电路,其输出电平与 CMOS/TTL 电路兼容。

二、TMP03/04 工作原理

TMP03/04 有三种封装形式:TO-92、SO-8 和 RU-8,引脚排列如图 2.16-1 所示。其中,U_+ 接电源的正极,GND 为公共地,D_{OUT} 为串行数据输出端。

TMP03/04 的内部框图如图 2.16-2 所示。主要包括 4 大部分:① 基准电压源和温度传感器。其中,基准电压源的输出电压接至 1 位的 DAC(图中未画),温度传感器输出的与热力学温度成正比的 U_{PTAT} 电压,接到求和器的一个输入端。② Σ-Δ 调制器,内含模拟求和器(亦称加法器)、积分器、比较器(亦称量化器)和 1 位数/模转换器(1bit DAC)。③ 数字滤波器。④ 高速时钟振荡器。由模拟求和器、积分器、比较器和 1 位 DAC 构成一个闭环系统,比较器还起到负反馈作用。它能根据输入温度信号的变化情况,来改变比较器输出信号的占空因数,



(a) TO−92封装　　(b) SO−8和RU−8封装

图 2.16−1　TMP03/04 的引脚排列

通过负反馈电路使积分器输出电压 U_{OUT}为最低。上述电路也属于电荷平衡式转换器，经过多次快速比较之后，输出的数字量就与被测温度成比例关系。

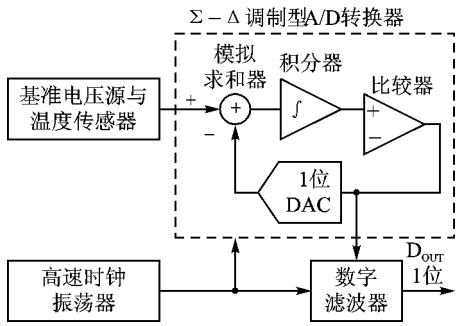


图 2.16−2　TMP03/04 型数字温度传感器的内部框图

1. **Σ−Δ 式 A/D 转换器**

近年来，随着超大规模集成电路 (VLSI) 技术的发展，采用 VLSI 工艺制成的高性能 Σ−Δ 式 A/D 转换器，不仅已成为数字通信、数字音响等领域的主流产品，还被用于新型数字温度传感器中。Σ−Δ 式 A/D 转换器由 Σ−Δ 数字调制器和数字滤波器组成。Σ−Δ 式 A/D 转换器以很高的采样速率和很低的采样分辨力 (1 位)，将模拟信号转换成数字信号，再使用过采样、噪声整形和数字滤波等方法来提高有效分辨力。一阶 Σ−Δ 式 A/D 转换器中的模拟电路非常简单，只需 1 个积分器、1 个模拟求和器、1 个比较器。

Σ−Δ 式 A/D 转换器采用了“过采样”(Oversampling) 技术。设采样频率为 f_S，过采样倍数为 K。若用 Kf_S 的采样速率对输入信号进行过采样，则量化噪声的频谱就位于 DC (直流) ~ $Kf_S/2$ 之间。通过对量化噪声的频谱整形，还可使绝大部分噪声位于 $f_S/2 \sim Kf_S/2$ 之间，仅有很少一部分留在 DC (直流) ~ $f_S/2$ 范围内。再利用数字滤波器滤掉绝大部分噪声，只保留有用的信号。这样，不仅提高了信噪比，而且能用低分辨力 A/D 转换器来达到高分辨力的指标。当比较器的采样频率远高于模拟输入信号频率时，就称之为“过采样”。利用过采样技术可将已转换为数字量的输入信号频谱 (低频段) 与量化噪声频谱 (高频段) 分离开。这样再通过数字低通滤波器，就很容易滤除量化噪声及混叠噪声，获得高信噪比、高分辨力的数字信号。

综上所述，Σ−Δ 式 A/D 转换器兼有积分式 A/D 转换器和反馈比较式 A/D 转换器的优点：对串模干扰的抑制能力很强，而对外围元件的精度要求较低；由于采用了数字反馈方式，因此比较器的失调电压及零点漂移不会影响转换精度。此外，Σ−Δ 式 A/D 转换器还能滤除量

化噪声并且达到高分辨力指标,这更是其显著特点。

2. TMP03/04 的测温原理

TMP03/04 是将温度传感器、Σ－Δ 调制器和数字滤波器集成到同一芯片上而制成的。由于 VLSI 的工艺和技术日益成熟,因此能大大降低器件的成本,开发出高性价比的数字温度传感器。TMP03/04 的分辨力为 12 位,还可配带 16 位计数器的微处理器。内部比较器的调制输出是经过电路编码的串行数据,在通过 μP 解码后,即可获得摄氏(或华氏)温度数据。采用电路编码技术的优点是便于单线传输数据,并且不依赖于时钟,能避免引入时钟误差。

TMP03/04 的工作原理是将被测温度的模拟量转换成数字量,并且把数字化信号编码成时间比率(t_1/t_2)的形式。t_1 和 t_2 在时间上是连续的,用同一个时钟即可获得二者的比率。因此温度仅与时间比率有关,而与时钟频率无关,即使时钟频率发生波动,也会在解码过程中被数字滤波器滤掉。

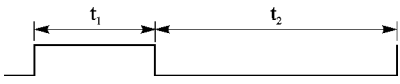


图 2.16－3 TMP03/04 的输出波形

TMP03/04 的输出信号为矩形波,当 θ = +25 时,矩形波的标称频率为 35 Hz。输出波形如图 2.16－3 所示。图中的 t_1、t_2 分别代表 1 位数据波形中的高、低电平持续时间。被测温度与 t_1/t_2 的比率有关:

$$\theta = 235 - \frac{400t_1}{t_2} \quad (1)$$

$$\theta = 455 - \frac{720t_1}{t_2} \quad (2)$$

式(1)被测温度的单位为摄氏度(℃),式(2)被测温度的单位为华氏度(°F)。

举例说明,当 θ = 0 ℃时,t_1/t_2 = 58.8 %,代入(1)式中计算出 θ = 235 - 235.2 ≈ 0 ℃。实际上,只需将 D_{OUT} 信号加至微处理器的定时/计数器接口,利用软件很容易计算出温度值。定时/计数器最大有效计数值与量化误差的对应关系如表 2.16－1 所列。不难看出,μP 的计数值越大,时钟频率越高,量化误差就愈小。

表 2.16－1 最大有效计数值与量化误差的对应关系

μP 中的计数器		最高温度 θ_{max}/℃	最高计数频率 f_{CPmax}/kHz	+25 ℃时的量化误差	+77 °F 时的量化误差
位数	最大计数值 N_{max}				
12	4 096	+125	94	0.284	0.512
13	8 192	+125	188	0.142	0.256
14	16 384	+125	376	0.071	0.128

三、TMP03/04 校准方法及使用要点

1. TMP03/04 的校准方法

TMP03/04 在出厂前已对精度和线性度进行了激光修正,一般情况下无须再进行校准。若用户确实需要,也可以进行单点校准。具体方法是在 +25 ℃的室温下,分别记录实际温度值 θ_A 和温度传感器的测量值 θ_B,再根据式(3)求出 TMP03 的偏移常数 k:

$$k = 235 + (\theta_A - \theta_B) \quad (3)$$

对于 TMP04,应将式 (3) 中的常数改为 455。

2. 定时/计数器的优化设计

使用定时/计数器 (以下简称计数器) 时,可按下述原则来计算 t_2、f_{CPmax}和量化误差 γ。

(1) 计算 t_2

t_1 是固定的,其标称值为 10 ms,最大值 $t_{1\,max} \leqslant 10\ (1 + 20\ \%)$ ms = 12 ms。t_2 随被测温度而变化,最大值 $t_{2\,max}$ = 44 ms,对应于最高温度 θ_{max} = 125 ℃。

对于其他温度 θ,可用下式计算出 t_2 值:

$$t_2 = \frac{400 t_{1\,max}}{235 - \theta} \tag{4}$$

(2) 计算 $f_{CP\,max}$

所选最高计数频率必须合适,才能防止计数器在 t_2 时间内溢出。令最大计数值为 N_{max},计算 $f_{CP\,max}$的公式如下:

$$f_{CP\,max} = N_{max} / t_{2max} \tag{5}$$

用 12 位计数器计数时,从表 12.6 - 1 中查出 N_{max} = 4 096,代入式 (5) 中计算出 $f_{CP\,max}$ = 4 096/44 ms = 94 kHz。

(3) 计算量化误差 γ

使用 12 位计数器,再给计数频率留出 5 % 的余量,实选 $f_{CP} = f_{CP\,max}\ (100\ \% - 5\ \%) \approx 90$ kHz。在 θ = +25 ℃时,计算量化误差的计算公式为

$$\gamma = 400 \times \left(\frac{N_1}{N_2} - \frac{N_1 - 1}{N_2 - 2} \right) \tag{6}$$

式中

$$N_1 = t_{1\,max} \cdot f_{CP} = 12\ \text{ms} \times 90\ \text{kHz} = 1\,080$$

$$N_2 = t_{2\,max} \cdot f_{CP} = 44\ \text{ms} \times 90\ \text{kHz} = 3\,960$$

代入式 (6) 中计算出 γ = 0.073 ℃。但是,当 θ 超过 +25 ℃时,γ 值会增大。另外,计数器的位数愈高,测温精度也愈高。考虑到 TMP03/04 内部噪声所产生的量化误差约为 0.1 ℃,因此 +25 ℃时的总量化误差不会超过 0.3 ℃。

参 考 文 献

1 美国 AD 公司产品资料,2000

2 沙占友,等. A Study of the Control System with Intelligent Temperature Sensors. ICEMI'第四届国际电子测量学术会议论文集. 电子测量与仪器学报,1999 年 13 卷增刊,1999.8 (被 ISTP 收录)

3 沙占友. 由 DS1820 组成的单线数字温度计原理与应用. 电测与仪表,1999 (2)

选自《单片机与嵌入式系统应用》月刊,2002 年第 3 期

2.17 TMP03/04 型数字温度传感器的应用

河北科技大学 沙占友 睢丙东 王彦朋

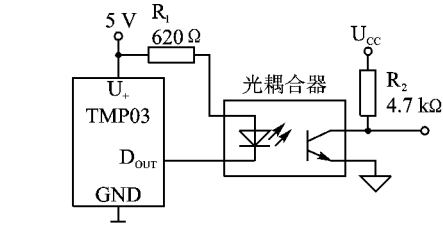
一、远程测温电路

远程测量温度时,传输线上存在着较高的共模电压,须用光耦合器(以下简称光耦)对输出端进行隔离。三种光耦隔离电路分别如图 2.17－1(a)、(b)、(c)所示。

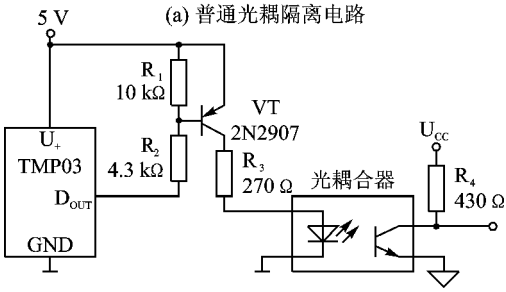
2.17－1(a)图为普通光耦隔离电路。TMP03 能够承受 5 mA 以下的灌电流,可直接与光耦中 LED 发光二极管的阴极连通。此时上拉电阻 R_1 还起到限流作用,防止 LED 过流损坏。取 R_1 = 620 Ω,当 D_{OUT}端呈低电平时,灌电流小于 1 mA,LED 上的正向压降仅为 1 V 左右。需要注意的是,光耦合器的开启和关闭时间必须完全相同,否则会导致被传输的数据发生错误。某些达林顿型光耦合器(例如 4N32)的开启时间远大于关闭时间,不能在此使用。

2.17－1(b)图中增加了 1 只 2N2907 型 PNP 管,目的是给 LED 提供较大的工作电流。

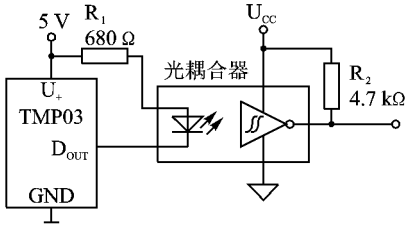
2.17－1(c)图中采用施密特整形输出的光耦合器,能降低传输噪声。



(a) 普通光耦隔离电路



(b) 加PNP管的光耦隔离电路



(c) 具有整形作用的光耦隔离电路

图 2.17－1 三种光耦隔离电路

进行远程测温时，TMP03/04 远优于模拟输出式温度传感器。这是因为它输出的是数字信号，比模拟信号的抗干扰能力强。远程传输信号时不得影响 t_1/t_2 的比率，必要时还可加 1 片 ADM485 型 RS－485 差分线驱动器，电路如图 2.17－2 所示。该电路能准确传输 1 200 m 远的温度信号。ADM485 中的发射器和接收器所造成的滞后时间误差仅为 5 ns，不会影响 t_1、t_2 值。在 RS－485 总线上可以接 32 片 ADM485。

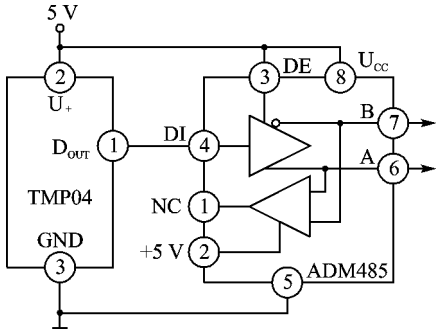


图 2.17－2　加外部缓冲器的远程测温电路

二、TMP03/04 与 80C51 单片机的接口电路

TMP03/04 配上单片机后，很容易用微机中的计数器分别测得 t_1、t_2，再用软件计算出温度值。因为 TMP03/04 是用时间比率（t_1/t_2）来计量温度的，所以并不要求微机的计数频率十分精确，但计数频率应足够高。

TMP04 与 80C51 单片机的接口电路如图 2.17－3 所示。该接口电路非常简单，在温度传感器与单片机之间仅需单线连接，TMP04 的输出数据可直接加到 80C51 的 P1.0 口。若按传统的串行通信输入协议，至少需要 3 条线（数据线、时钟线和片选信号线），甚至更多。这是因为 TMP03/04 的输出数据仅代表 t_1 与 t_2 的比率，这与通常讲的二进制数据完全不同。

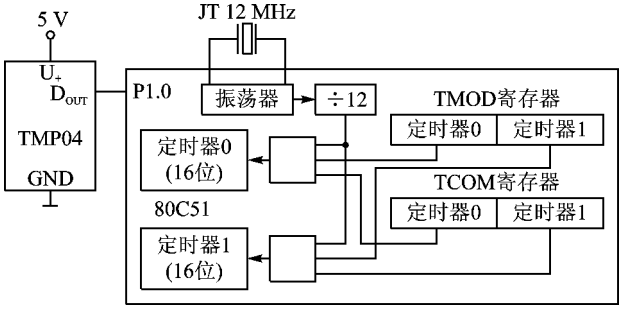


图 2.17－3　TMP04 与 80C51 单片机的接口电路

80C51 内部有 2 个 16 位计数器，并且是由 2 个专用寄存器分别控制的。TMOD 寄存器亦称定时器模式寄存器，专门控制定时器 0 和定时器 1 的工作模式，决定哪个定时器工作。TCOM 寄存器则用来控制定时器 0 和定时器 1 的计数起止时间。这样很容易分别测出 t_1、t_2。80C51 采用 12 MHz 晶振频率时，经内部 12 分频器后获得 1 MHz 的计数频率，计数周期为 1 μs。

接口电路的程序清单如下：

```
;TMP04 与 8051 的接口测试
;使用定时器 0 和 1 来测量占空比
;这个程序分三步
;① 清除定时寄存器,然后等待 P1.0 输入脚上出现 0－1 的跳变 (此引脚与 TMP04 的输出相连)
;② 一旦 P1.0 变为高,定时器 0 开始启动。然后程序开始循环检测 P1.0
;③ 一旦 P1.0 变为低,定时器 0 停止,同时定时器 1 启动。程序又再次循环检测 P1.0,直到 P1.0 变低,定时器 1 停止,从而 TMP04 的 t1 和 t2 的数值就会存储在特殊功能寄存器 8AH 到 8DH (TL0 到 TH1) 中
;主控程序
$ MOD51
$ TITLE (TMP04 Interface, Using T0 and T1)
$ PAGEWIDTH (80)
$ DEBUG
$ OBJECT
;变量声明
PORT1 DATA 90H                          ;端口 1
;TCON DATA 88H                          ;定时器控制
;TMOD DATA89H                           ;定时器模式
;TH0 DATA 8CH                           ;定时器 0 高字节
;TH1 DATA 8DH                           ;定时器 1 高字节
;TL0 DATA 8AH                           ;定时器 0 低字节
;TL1 DATA 8BH                           ;定时器 1 低字节
ORG 100H                                ;任意的起始地址
READ_TMP04：MOV A,#00                   ;先清除寄存器
MOV TH0,A
MOV TH1,A
MOV TL0,A ;
MOV TL1,A ;
WAIT_LO：JB PORT1.0,WAIT_LO             ;等待 TMP04 输出变高
MOV A,#11H                              ;准备好启动定时器 0
MOV TMOD,A
WAIT_HI：JNB PORT1.0,WAIT_HI            ;等待输出变高
;在 TMP04 输出变高期间定时器 0 运行
SETB TCON.4
WAITTIMER0：JB PORT1.0,WAITTIMER0
CLR TCON.4 ;
;在 TMP04 输出变低期间定时器 1 运行
SETB TCON.6                             ;启动定时器 1
WAITTIMER1：JNB PORT1.0,WAITTIMER1
CLR TCON.6                              ;停止定时器 1
MOV A,#0H                               ;准备停止定时器
MOV TMOD,A
```

```
RET
END
```

上述程序用来监视 TMP04 的输出,并且开启或关闭相应的计数器来测量 t_1 与 t_2 的比率。当 TMP04 输出为高电平(对应于 t_1)时,由定时器 0 进行记录;当输出呈低电平(对应于 t_2)时,由定时器 1 记录。记录结果存储在特殊功能寄存器 SFR 的 08AH ~ 08DH 单元中。

在调用读 TMP04 的程序时,计数寄存器就被清零。程序首先将计数器置成 16 位模式,然后等待 TMP04 输出为高电平。当 P1.0 口输入高电平时,定时器 0 开始工作,程序用来监视输入口计数器进行累加计数。一旦 TMP04 输出为低电平,定时器 0 就停止计数,而定时器 1 开始计数,直到 TMP04 的输出又变成高电平为止。执行完上述子程序之后,定时器 0 中的 t_1 值和定时器 1 中的 t_2 值,分别存入各自的专用寄存器中。最后由软件计算出被测温度 t_o。由于 80C51 与 TMP04 属于异步操作,因此从 TMP04 的输出电平发生跳变、到定时器开始工作之间,必须加上延迟时间。这个延迟时间从 0μs 开始,到识别转换指令为止。控制 80C51 跳转到某位置的指令,需要 24 个时钟周期,即 24/12 MHz = 2 μs。在 +25 时,由 2 μs 延迟时间所造成的测温误差约为 ±0.15 ,一般可忽略不计。

三、TMP04 与数字信号处理器 ADSP-2101 的接口电路

数字信号处理器(DSP)属于 CMOS 超大规模集成电路(VLSI),其集成度可达 2 000 万个元件/片,运算速度高达 2 亿次/s,电源电压为 1.5 ~ 5 V。TMP04 很容易和 ADSP-2100 系列 DSP 芯片相匹配,典型电路如图 2.17-4 所示。TMP04 的 D_{OUT} 端接 ADSP-2101 的输入端 FI。ADSP-2101 内部只有 1 个计数器,但其运行速度极快,采用了分时计数的办法完全可代替 80C51 中 2 个计数器的功能。ADSP-2101 完成 1 个指令仅需 1 个时钟周期,相比之下 80C51 需要 12 个时钟周期。因此,采用相同的时钟频率时,ADSP-2101 可获得更精确的结果。

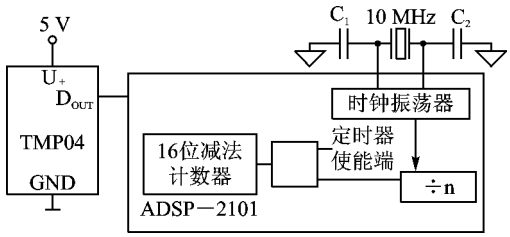


图 2.17-4 TMP04 与数字信号处理器 ADSP-2101 的接口电路

ADSP-2101 中的定时器可视为一个可编程减法计数器。使用软件编程时,计数频率按照时钟的预分频系数进行分频。将(N-1)存入预分频系数寄存器中,则晶振频率将被 N 分频。例如,将 4 存入寄存器时,就对 10 MHz 晶振频率进行 5 分频,得到 2 MHz 的计数频率。在程序开始时,首先设定寄存器的初始值为 0FFFH。ADSP-2101 开始监视 FI 端的输入电平,直到出现一个下降沿时启动计数器。当 TMP04 输出变高时,计数器停止计数,计数值也从 0FFFH 减到实际值,并存储下来。然后计数器又重新置数,再次做减计数,直到 TMP04 输出为低。这样就先后测出了 t_2、t_1 值。

TMP03/04 用来监视电子设备中高速 μP、DSP 芯片的温度时,将 TMP03/04 装在下边并尽量贴近 μP 或 DSP 芯片,即可测出芯片的表面温度。对于 TO-92 封装的 TMP03/04,直接固定

在印制版(PCB)上,就能检测流过印制板的空气温度。

参 考 文 献

1 美国 AD 公司产品资料,2000
2 沙占友,等. 新编实用数字化测量技术. 北京:国防工业出版社,1998
3 沙占友. 由 DS1820 组成的单线数字温度计原理与应用. 电测与仪表,1999(2)

选自《单片机与嵌入式系统应用》月刊,2002 年第 4 期

2.18 谐振式水晶温度传感器的现状和发展预测

哈尔滨信息产业部电子第四十九研究所 (150001) 谢胜秋 宋国庆

一、引 言

温度是一种重要的计量参数,工农业、国防事业、科研和日常生活都离不开对温度的测量。随着科技的飞速发展,对温度的测量要求愈来愈高,不仅要求准确度高、响应速度快,而且要求低功耗、适宜远传和便于计算机配合使用。

目前使用的 PN 结半导体传感器、陶瓷热敏电阻器、铂 (或铜) 电阻器、热电偶等都是把温度信号转换为电阻、电压或电流信号进行检测。它们皆为模拟式传感器,其缺点是转换效率低、功耗大、换算复杂、抗电磁干扰能力差,不便于进行数字信号处理。

谐振式水晶温度传感器 (以下简称为 RQTS) 是一种新型数字传感器[1]。它一问世就受到人们的关注。随着水晶加工工艺的成熟和微型计算机的发展使 RQTS 有了更快的发展。它的主要特点是:

⑴ 分辨力可达 $10^{-3}\sim10^{-6}$ 。灵敏度大大高于绝大多数温度传感器和温度计,其中包括铂电阻温度计[2]。

⑵ 准确度高,可以作为温度传递标准。“现广泛应用的铂电阻温度计的非线性大,在 0 ~100 的范围内达 0.55 ,而水晶温度传感器却不超过 0.07 ,”可制作 0.01 级温度计[2]。

⑶ 长期稳定性佳,工作 12 年,其准确度仍可保持为 0.002 ~0.02 [2]。

⑷ 输出是频率信号,没有铂电阻传感器的接线电阻误差,可以远传[2~4]。

⑸ 由于 RQTS 的谐振 Q 值 (品质因数) 高达 2×10^{6},其谐振频率和热敏特性完全由 QTS 决定,外电路的温度漂移、时漂极小。因此没有铂电阻温度计的温度漂移现象。

⑹ 传感器振荡频率受电源影响较小,因此利用普遍稳压电源就可以获得精密的、高稳定特性[2~4]。

⑺ 晶体的各向异性决定了传感器可通过切割方位角的调整,改善其频率-温度特性。目前已找到一些线性频率-温度特性的水晶切型。以后如采用新温标,只要对水晶温度传感器标准切型的切割方位角进行小的调整,其频率-温度特性曲线相对新的温标仍然具有良好的线性[5,6]。这是绝大多数温度传感器和温度计 (包括铂电阻式) 做不到的。

⑻ 绝大多数温度测控装置需要传感器与控制、显示器分离,并且温度信息能够在任意长度、廉价的电线中不失真地传输,不受任何电磁干扰,也不干扰别的信号。铂电阻和热电偶传感器必须采用高准确度模拟电路变为 4 ~20 mA 信号或进行 A/D 变换变为数字信号远传。这不仅降低了准确度,而且 A/D 变换器的温度漂移又带来误差。QTS 的输出信号是频率,可直接远传,并且导线长短对准确度无影响[2]。

目前国际上问世的 QTS 的种类繁多,约有 28 种以上,但是按其芯片结构大致可分为两大

类：①透镜式（厚度切变模式）；②音叉式 QTS。

按频率–温度特性分又可分二大类：①线性 QTS；②非线性 QTS。

二、测温机制和传感器的结构

在某一温度时，厚度切变式石英谐振器的固有谐振频率 f 可表示为[3]

$$f = \frac{n}{2d}\sqrt{\frac{C_{ij}}{\rho}}$$

式中：n 为泛音次数；ρ 为水晶密度；d 为水晶厚度；C_{ij} 为水晶弹性常数。d，ρ 和 C_{ij} 是温度的函数。在某一温度下，弯曲振动模式工作音叉的谐振频率 f[4] 为

$$f = \frac{a^2}{4\pi\sqrt{3\rho S'_{22}}}\cdot\frac{\omega^2}{L^2}\cdot\frac{1}{\sqrt{1+\frac{1}{3}\left(1+\frac{S'_{44}}{K\cdot S'_{22}}\right)\left(\frac{a}{2}\cdot\frac{\omega}{L}\right)}}$$

式中：a 是常数，它是 $\cos a\cdot\cosh a+1=0$ 方程的根；K 是修正系数；ρ 是水晶的密度；S'_{22}，S'_{44} 是水晶的柔顺常数（取决于切型）；ω 是臂宽；L 是臂长。L，ω，ρ 是温度的函数。

扭曲振动模式工作音叉的谐振频率 f[3] 为

$$f = \frac{1}{L\sqrt{\rho S'_{55}}}\cdot\frac{d/\omega}{\sqrt{1-(d/\omega)^2}}\cdot\sqrt{1-\frac{192}{\lambda\cdot\pi^2}\sum_{n=0}^{a}\frac{\operatorname{tg}\frac{2n+1}{2}\pi\lambda}{(2n+1)}}$$

式中 参数 $\lambda=\frac{\omega}{d}\sqrt{\frac{S'_{55}}{S'_{66}}}$；$S'_{55}$，$S'_{66}$ 是水晶的柔顺常数。λ，S'_{55}，S'_{66} 是温度的函数。

RQTS 的谐振频率 f 与温度 t 的关系可表示[2]为

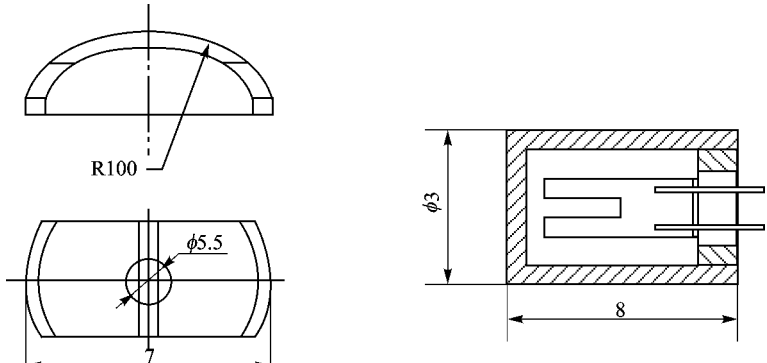
$$f_t = f_{t_0}\left[1+A_0(t-t_0)+B_0(t-t_0)^2+C_0(t-t_0)^3\right] \tag{1}$$

式中：f_{t_0} 为在 t_0 时的频率；f_t 为在 t 时的频率，t_0 是任意的参考温度；A_0，B_0，C_0 分别是 1，2，3 阶温度系数，它主要由石英晶体切型决定。若忽略 3 次项，则可求得温度 t 为

$$t = \frac{-A_0+\sqrt{A_0^2-4B_0\left(\frac{f_{t_0}-f_t}{f_{t_0}}\right)}}{2B_0}+t_0$$

显然，测得传感器的输出频率，就可以知道被测体的温度。利用单片机，代入 A_0、B_0、t_0、f_{t_0} 后，当测出 f_t 就可以求得 t，即可求出被测的温度。

通常，透镜式和音叉式 QTS 的结构如图 2.18－1 所示。



三、透镜式 QTS 的发展

最先问世的透镜式 QTS 是 AC 切型水晶(它的方位角为 Y × /L + 31°),QTS 是由 Flynn 研制成功的[2],其一阶频率温度系数为 20×10^{-6}/ ;后来是 Wade 和 Sludsky 研制成功的 Y 切型,工作频率为 10 MHz,其一阶频率-温度系数高达 100×10^{-6}/ ,传感器分辨力可达万分之几度[2]。另一个比较成功的例子是 Smith 和 Spencer 研制的 Ys 切型 QTS,它采用 5 MHz 频率 3 次泛音工作。由于采用惯用的高精密晶体加工工艺,Q 值高达 3×10^{6},其一阶频率-温度系数为 80×10^{-6}/ ,短期稳定度和长期稳定度皆高达 10^{-10},因此传感器的技术指标很高[2],其温度分辨力高达 4×10^{-6} 成为国际最高水平[2]。至今美、俄、日仍旧生产此类 QTS。

上述的温度传感器的一阶温度系数很大,分辨力很高,但是它们的最主要的缺点是二阶温度系数较大,换言之,频率温度特性的线性度较差。因此很多学者致力于线性温度系数的 QTS 的研制。

美国 HP 公司的 Hammond 等人分别计算了式(1) $B_0 = 0$ 和 $C_0 = 0$ 晶体方位角的轨迹,然后找出这两个轨迹的交点,并把此方位角命名为 LC 切型(即 Linear Coefficiend 的缩写),至今美、俄、日、法等国都生产此种切型的传感器。美国 HP 公司采用 LC 切型生产的水晶温度计的指标如下:测温范围为 - 80 ~ 250 、分辨力为 0.000 1 、准确度为 ± 0.02 、稳定度为 ± 0.01 /年,响应时间可达 1 s[6]。

可是此种切型的可操作性欠佳。因为在 $\varphi' = 8.73°$ 和 $\theta = 13°$ 两个定向角附近很难找到一个相近的切割面,并且对 X 射线产生很强反射的标准原子面。因此对操作者的工艺技术要求甚高,其生产效率和成品率皆低[6,7]。

俄国学者 ЯРСЛАВСИЙ 提出的 ПЯ 切型,既保留了 LC 切型的优点,又提高了其可操作性。他们利用 ПЯ 切型研制的传感器指标如下:f = 5 MHz,工作温度范围为 - 60 ~ 120 ,准确度为 0.05 ,温度灵敏度为 180 Hz/ 。1984 年日本的中泽光男教授发明了 NL 切型,采用 NL 切型的温度传感器[2]是一种超线性频率温度特性 QTS。

NL 切型族的主要参数如下:

NL_1:$A_0 = 63.5 \times 10^{-6}$/ ,比 LC 切型高近 1 倍;

$B_0 = -18.1 \times 10^{-9}$/( )^2;

$C_0 = -35.9 \times 10^{-11}$/( )^3,非线性度为 0.003 7%。

NL_2:$A_0 = 17.2 \times 10^{-6}$/ ,是切型的 43%,线性度优于 LC 切型,热敏晶体另一个重要因子——电容比为 733,比 LC 切型大 4 倍,因此其频率稳定性更佳。

NL_1 切型传感器比 LC 的一阶温度系数大得多,灵敏度更高。中泽研制的传感器[2]工作频率为 4 025 kHz,Q 值为 55 000,$A_0 = 65.2 \times 10^{-6}$/ (实测值)。1987 年中泽教授又研究出一种采用超线性并且具有应力补偿功能的 NLSC 切型[3],解决了动态温度传感器和高精密水晶温度传感器的一个关键技术,使得透镜式水晶温度传感器更加成熟和完美。现在日本、俄国、保加利亚等正在生产采用此种切型的温度传感器[3,8]。

四、谐振式音叉水晶温度传感器

透镜式水晶温度传感器的主要缺点是:

(1) 工作频率过高,为 10 ~ 28 MHz,所需外围电路成本高,并且需要复杂的杂波辐射抑制

措施。

⑵ 传感器的体积较大。通常采用 T0 -5 型或 49U 型外壳封装。例如美国和日本产品的尺寸为 φ10 mm × 100 mm 或 13.5 mm × 11 mm × 4.7 mm,法国和俄罗斯有的厂家产品甚至达到 φ18 mm × 37 mm。因此,响应速度慢。

⑶ 厚度切变模式温度传感器的动态电阻值为 20 ~ 40 Ω,工作电流大,功耗大,约为 2 mW 以上。

⑷ 传感器的水晶片与电极(金或银膜)的接触面积较大(通常 φ≥7.0 mm),由于两者膨胀系数之差将导致热应力的产生,而厚度切变模式对应力又比较敏感,故易产生迟滞误差,甚至有时达到 0.02 以上。

⑸ 传感器的工作频率位于短波波段,因此必须以同轴电缆作馈线。这不仅提高了成本而且装配复杂,特别给温度探头同步供电和传送信号带来很大的困难。

⑹ 厚度切变模式温度传感器的外形尺寸大,不适宜测量"点温",更不适宜精确地测量特殊形状物体的表面温度。

简言之,厚度切变模式水晶温度传感器的振动模式和工作机制已经限制了温度测量范围的扩展、响应速度的提高和电功耗的降低[8]。

谐振式音叉水晶温度传感器克服了上述缺点。目前国际上公布的传感器主要有两种:①弯曲振动模式工作的音叉 RQTS;②扭曲振动模式工作的音叉 RQTS。前者灵敏度高,但频率温度线性度差;后者的线性度好,但是灵敏度低。

日本植田先生研制的弯曲振动模式音叉 RQTS 采用 φ = -40.23°切型[2~4]。它的一阶温度系数为 -47.6 × 10^{-6}/ ,二阶温度系数为 -187 × 10^{-9}/ ,工作频率为 40 kHz,工作温度范围为 -269 ~ 250 ,外形尺寸为 φ3 mm × 8 mm,测温准确度为 0.05 ,响应时间为 0.9s[2~4]。

瑞士的 Dinger 研制出采用扭曲模式工作的音叉 QTS[3],它的切割角 θ = 92°,一阶温度系数为 20 × 10^{-6}/ ,二阶温度系数为零,频率温度特性为直线,工作频率为 262.114 kHz,动态电阻为 15 kΩ,动态电容为 0.3 pF,外形尺寸为 φ1.5 mm × 5.3 mm。这是目前国际上外形尺寸最小的一种。

五、今后发展预测

水晶温度传感器发展方向主要有:

⑴ 材料改性(利用电清洗法等提高水晶 Q 值,提高传感器测温极限以及进一步改善其长期稳定性)。

⑵ 寻找新的单转角切型。目前发现的优良热敏切型大都是双转角切的。其加工工艺复杂,难度大,成品率低,不适宜普通工厂批量生产。

⑶ 逐渐采用化学刻蚀制作水晶传感器,可是目前此法尚不成熟,仅能对 Z 轴附近切型进行化学刻蚀,很多优良的热敏切型无法利用化学刻蚀法制作。因此,通常利用刻蚀法制作的水晶温度传感器技术指标不如机械法的高,例如 Q 值,分辨力、压电活力等,但是刻蚀法加工的物理尺寸较小。因此,水晶微机械加工技术的突破是水晶传感器发展的关键。采用离子束刻蚀是一种新方法,它具有化学刻蚀和机械加工法两种工艺的优点,颇有前途[5,6]。

⑷ 多功能化,单片式水晶温度传感器、压力传感器、单片式温度-流量传感器国外正在研制。

(5) 美国和日本正在研制采用复合式(弯曲和扭曲振动)的音叉传感器[3,4]。它具有两者的优点,灵敏度高、准确度好、线性度佳、体积小、功耗低。

总之,水晶温度传感器的发展前景广阔。国外现已应用在精密温度测量仪、井下石油温度计、原子反应堆、数字体温计、海图制作、热线式流量计、热量仪等,作为精密温度测量或控制用。

参 考 文 献

1 徐军,李欣,林冬辉,等. 适宜动态检测的新型谐振式水晶温度传感器[J]. 传感器技术,2000,19(2):35~37

2 马洛夫 BB. 压电谐振传感器[M]. 翁善臣译. 北京:国防工业出版社,1984

3 МАЛОВ В В. ПЪЕЗО-РЕЗОНАНСНЪ ДАТЧИКИ [M]. МОСКВА:ЭНЕРГОАТОМИЗДАТ,1989. 19~48,102~127

4 МОСТЯЕВВ В,ИДЮЖИКОВ А. ТЕХНОЛОГИЯ ПЪЕЗО И АКУСТО-ЭЛЕКТРОННЫХ УСТРОЙСТВ [M]. МОСКВА:ЯГУАР,1993,39~223

5 刘永,恒文. 实用校准刻度的石英测温技术[J]. 仪表工业,1992,(1):25~27

6 林江,范加玲,张滨华,等. 分辨率高达0.00051的新型水晶谐振式温度传感器[J]. 传感器技术,1995,(4):19~22

7 Hjord K, Thornell G, Spohr R. Quardz micromechning by lidhgraplic control of ion drack edching [J]. Appl. Phys. Ledd., 1996,69(22):3435~3436

8 王玉娟,董晓辉,叶东生,等. 一种音叉式水晶温度传感器[J]. 传感器技术,1999,18(1):31~34

选自《传感器技术》月刊,2002 年第 2 期

2.19 石英晶体温度传感器的应用

湖南大学应用物理系 (410082)
陈小林 王祝盈 谢 中 翦知渐 皮承宪

石英谐振器的振荡频率随温度而变化。采取特殊的切割方向,可以使这种变化加强,再把这种变化控制成线性或接近线性关系,就可以制成一种高灵敏度测温传感器。

根据不同的频率和切型,石英晶体温度传感器的温度灵敏度 C_t 可以在 20 ~ 2 850 Hz/ 范围内变动。对这一变化进行频率测量,可以使温度分辨力达到 1×10^{-4} ,这是一般温度计很难办到的。这样高的温度分辨力对于某些理化性能实验研究至关重要。

一、石英晶体温度传感器的频率–温度特性

实验表明,在 - 200 ~ 200 温度范围内,任何一种石英谐振器的频率–温度特性 f (t) ,可以足够精确地表示成展开式 [1]

$$f(t) = f_0\left[1 + \sum_{n=1}^{3} \frac{1}{n!} \frac{\partial^n f}{f_0 \partial t^n}\bigg|_{t=t_0}\right](t - t_0) \tag{1}$$

式中: f_0 为 t_0 时的晶体谐振频率;

系数 $\frac{1}{n!} \frac{\partial^n f}{f_0 \partial t^n}\Big|_{t=t_0} = T_f^{(n)}$ 称为 n 阶频率温度系数。

由此,将式 (1) 改写为

$$f(t) = f_0 [1 + T_f^{(1)} (t - t_0) + T_f^{(2)} (t - t_0)^2 + T_f^{(3)} (t - t_0)^3]$$

式中: $T_f^{(1)}$ 、$T_f^{(2)}$ 、$T_f^{(3)}$ 分别是一阶、二阶、三阶频率温度系数。

对于温度传感器而言,重要的是灵敏度要高,温度–频率转换的直线性要好。这就归结为寻找一些特别切型的振动模式,使得

$$T_f^{(1)} = \max, \text{且 } T_f^{(1)} \gg T_f^{(2)}, T_f^{(1)} \gg T_f^{(3)} \tag{2}$$

或

$$T_f^{(2)} = T_f^{(3)} = 0 \tag{3}$$

满足式 (2) 的谐振器就是目前通用的旋转 Y 切型 YXL/ +5°,工作在厚度剪切振动模式。频温系数约为 95.6×10^{-6} / ,且对切割时的取向偏差不敏感,这对元件的互换性是很有利的。满足式 (3) 的谐振器是一种双转角切型 YXWL/8° 44' /18° 的 LC 切型,频温系数 48×10^{-6} / ,只是 Y 切型的一半,但是线性度极好。

研制成功的石英晶体温度计采用旋转 Y 切型 YXL/ +5°,10 MHz,J3 型电阻焊接封装石英晶体谐振器。平均频率温度系数实测值为 971 Hz/ ,内充干燥氮气以加快热响应速度,工作温度范围为 - 30 ~ 120 ,年老化率约为 3×10^{-6},热时间常数约为 4 s。

二、石英晶体温度传感器频率–温度转换的非线性及处理

在 $t_0 \pm \Delta t$ 的温度范围内,非线性度定义为

$$N_f = \left[\frac{\partial f}{\partial t}\bigg|_{t_0+\Delta t} - \frac{\partial f}{\partial t}\bigg|_{t_0-\Delta t} \right] \Big/ \frac{\partial f}{\partial t}\bigg|_{t_0}$$

即

$$N_f = \frac{2T_f^{(2)}}{T_f^{(1)}} \cdot \Delta t$$

常见石英温度传感器的主要切型、各阶温度系数及温度–频率非线性度如表 2.19 – 1 所列[1]。

表 2.19 – 1 常见石英温度传感器的频率温度系数

温度系数 / 切 型	$T_f^{(1)}$ / ($\times 10^{-6}$)	$T_f^{(2)}$ / ($\times 10^{-9}$)	$T_f^{(3)}$ / ($\times 10^{-12}$)	N% ($\Delta t = \pm 50$)
YXL/0° (Y 切)	92.5	57.5	5.8	12.5
YXL/5° (旋转 Y 切)	95.6	63.8	35.0	13.4
YXL/31° (AC 切)	20.0	23.0	118.0	23.0
YXWL/11°10'/9°24' (LC 切)	33.78 ± 0.12	0 ± 0.14	0 ± 0.23	0 ± 0.08
YXWL/18°54'/9°45'45"	34.02 ± 0.19	1.87 ± 0.87		0.98

由表 2.19 – 1 可知,旋转双转角 YXWL 切型 (即 LC 切) 的温度传感器直线性最好,用它制成的温度计几乎没有非线性误差,在许多要求很高的场合适用。普通 Y 切型的温度传感器非线性较大,当测温范围比较宽时,需采用适当的方法以减少非线性导致的测温误差。最常用的方法之一是分段线性修正。将温度测量范围根据所要求的测量准确度分成若干段。实测每温度段的一阶频率–温度系数 $T_{f1}^{(1)}$,$T_{f2}^{(1)}$,$T_{f3}^{(1)}$ …。然后分别计算每段的频率–温度转换。这种方法,当分段数较多时,实测定标及软件编制都是较麻烦的。

采用一种简单的迭代法处理这个问题[2]。以一般的三次函数为例

$$f = f_0 \left[1 + T_f^{(1)} (t - t_0) + T_f^{(2)} (t - t_0)^2 + T_f^{(3)} (t - t_0)^3\right]$$

已知系数 $T_f^{(1)}$,$T_f^{(2)}$,$T_f^{(3)}$,及常数 f_0,测得 f 后,要用单片机实时解出温度 t,是不太容易的。采用迭代,取初值

$$t' = t_0 + \frac{f - f_0}{T_f^{(1)}}$$

再计算频率修正值

$$f' = T_f^{(2)} (t' - t_0)^2 + T_f^{(3)} (t' - t_0)^3$$

进行修正,得

$$t'' = t_0 + \frac{f - f_0 f'}{T_f^{(1)}}$$

这就是经一次迭代修正后的温度值。实际验算表明,在 ($t_0 \pm 10$) 范围内,非线性误差可以控制在 0.001 在 ($t_0 \pm 20$) 范围内,非线性误差可以控制在 0.01 在 ($t_0 \pm 30$) 范围内,非线性误差可以控制在 0.02 。并且 t_0 是任意设定的,这样亦可辅以分段法来满足更高

的测量要求。这种算法简单,在单片机上不难实现。采用旋转 Y 切温度传感器,经上述方法修正,全温度范围内(- 30 ~ 120)的非线性误差小于 0. 01 。

三、石英温度传感器短期频率不确定度与长期频率不确定度对温度测量的影响

在频率测量中,频率不确定度为

$$S_f = \frac{\Delta f}{f}$$

式中:f 是工作频率;Δf 是频率的起伏或漂移。当然,给出 S_f 时应注明测量的取样时间间隔或其他条件。如 S_f 已知,则最小可测量的温度增量为

$$\Delta t_{min} = \frac{S_f}{T_f^{(1)}}$$

因为石英晶体谐振器的 Q 值很高($10^4 \sim 10^6$),用其构成石英晶体振荡器的短期频率稳定度是很好的[3]。一般电阻焊 J3 封装的中等准确度石英谐振器做成温度传感器,秒级频率不确定度可小于 10^{-10},则温度增量可以测至

$$\Delta t_{min} = 10^{-10}/10^{-4} = 10^{-6}$$

可见,石英温度传感器的短期频率不确定度不对石英测温技术带来麻烦。

长期频率不确定度是指石英晶体谐振器的老化效应。老化主要是由于石英晶体表面吸附物的释放和应力的缓慢消除而引起的。老化又可分为时效老化和循环老化。测温石英晶体谐振器的年老化率约为 3×10^{-6},且老化低于 1×10^{-8},则它工作一天对温度带来 $\Delta t = 10^{-4}$ 的误差,问题不大。循环老化的影响,可以采用预循环老化的措施加以减少。

在定标温度传感器之前,温度探头进行了老化处理:在 85 的恒温箱内放置 3 个月,通电老化 1 个月,24 h,0 ~ 100 温度循环老化 30 次,效果令人满意。

四、测温晶体过热误差的影响

测温晶体的过热误差,是指石英晶体谐振器等效电阻上的电功率被转换成热量,使晶体片相对于周围介质产生过热。随着激励能量的增加,过热上升,这种误差可以用下式计算:

$$\Delta t_n = K_n P$$

式中:K_n 是过热系数;P 是耗散功率。

对于大多数测温石英晶体,过热系数约为 0. 1 ~ 0. 15 /mW。所以,若激励谐振器的耗散功率为 1 mW,则产生的过热温升为 0. 1 ~ 0. 15 ,该温升在温度定标过程中可以消除掉。若同样大小的激励功率有 ±10% 的起伏,则产生的测量误差为 0. 01 ~ 0. 015 。由此可见,为了降低过热误差,首先必须提高谐振器激励电平的稳定性,同时也要降低激励电平。

五、仪器的主要技术指标及软硬件配置

石英晶体温度传感器的结构原理如图 2. 19 - 1 所示。

仪器达到的主要技术指标如下:

- 测温范围为 - 30 ~ 120 ;
- 温度分辨力为 0. 0001 ;

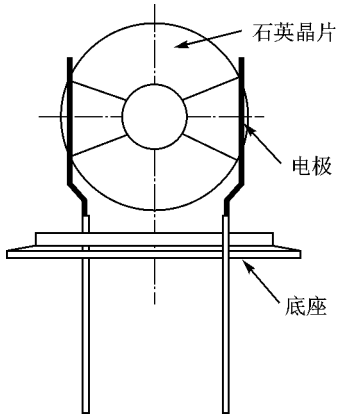


图 2.19－1 石英晶体温度传感器结构图

- 最大测温误差（- 30～120 范围内）为 0.2 ；
- 测温探头数为 8 个；
- 具有键盘设置、查询功能，定时测温打印功能；
- 功耗为 30 W；
- 16 位数码管显示。

图 2.19－2 是仪器的原理方框图。

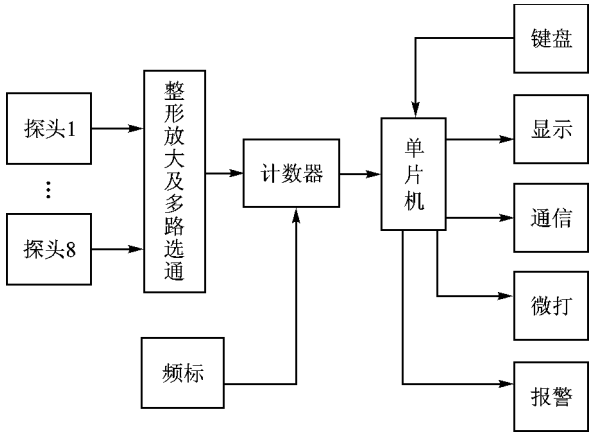


图 2.19－2 石英晶体温度计原理方框图

其中，温度探头主振电路采用一片 54LS04 六非门集成电路，其中一个门作为晶体振荡器，另三个门作整形电路输出。探头用 1.5 m 长的三芯屏蔽电缆与主机连接。为了隔离温度测试区对振荡电路的影响，晶体谐振器与振荡电路之间用 30 cm 长的聚四氟乙烯封装线连接。同时，晶体谐振器与保护套管之间用环氧树脂进行热隔离，故探头的热稳定性较好，感温速度快，热时间常数约为 5 s。单片机采用 8032 芯片。8032 的两个计数器用作测频计数的高位计数器，第三个计数器作为与系统机串行通信的波特率发生器。8279 作为键盘及显示接口，8155 作为扩展 I/O 接口并外带 PP40 打印机，实时钟采用 DS1216C，该芯片内嵌锂电池，同时带 8 KB 内存，数据保存 10 年以上，所有设置参数、计算参数都保存在该芯片内。

六、结束语

石英晶体温度传感器具有温度分辨力高、感温速度快、直接频率量输出等优点，在物质燃烧值的测定、物态相变研究、地震研究等领域有广泛的应用。研制成功的石英晶体温度计智能化程度高，性能稳定，具有八路温度探头，自带与系统机的接口及软件，价格低廉，在多通道高分辨力测温方面达到了新的水平，极具推广价值。

参 考 文 献

1 马波夫 B B. 压电谐振传感器 [M]. 翁差臣译. 北京：国防工业出版社，1994
2 邓建中，葛仁杰，程正兴. 计算方法 [M]. 西安：西安交通大学出版社，1995
3 赵声衡. 石英晶体振荡器 [M]. 长沙：湖南大学出版社，1997

选自《传感器技术》月刊，2002 年第 5 期

2.20　无线数字温度传感器的设计

湖南衡阳南华大学电气工程学院(421001)
黄智伟　朱荣辉　朱卫华

所设计的无线温度传感器是无线库房温湿度测量控制系统的一个部件,与有线方式比较具有可靠性高、安装容易、重复使用性和系统维护性好等优点。无线温度传感器由温度测量、无线收发和微控制器电路组成。

一、温度测量电路

温度测量电路如图 2.20－1 所示。

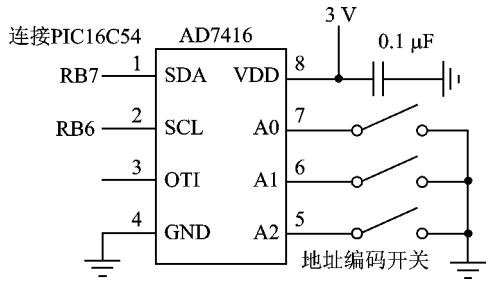


图 2.20－1　温度测量电路

采用单片温度监控系统集成电路 AD7416[1],其芯片内部包含有温度传感器和 10 位模数转换器,可将感应温度转换为 0.25　量化间隔的数字信号。测温范围 - 55 ~ 125　,分辨力为 0.25　,精度为 ±2　。

AD7416 采用 I^2C 串行总线和数据传输协议来实现与微控制器的数据传输,数据输入/输出线 SDA 以及时钟信号线 SCL 与微控制器的 RB7 和 RB6 相连。当 SCL 保持高电平时,SDA 从高电平到低电平的跳变为数据传输的开始信号,随后传送 AD7416 的地址信息和读/写控制位。其地址信息的格式为:1001 A2 A1 A0 R/W。读/写控制位为 1 时,表示对 AD7416 进行读操作;为 0 时,则表示进行写操作。当每个字节传送结束时,必须在收到接收数据一方的确认信号(ACK)后方可开始下一步的操作。然后在地址信息和读/写控制位之后传送片内寄存器地址和数据。最后,在 SCL 保持高电平的情况下,当 SDA 从低电平跳变到高电平时将终止数据的传输操作。地址编码开关用于传感器的编号。

AD7416 片内温度传感器可按预先设置的工作方式对环境温度进行实时测量,并将结果转化为数字量存入到温度值寄存器中(地址 00H),其环境温度与输出数据的关系如表 2.20－1所列。

表 2.20－1 环境温度与输出数据的关系

环境温度/	二进制数字输出
- 50	11 0011 1000
- 25	11 1001 1100
- 0.25	11 1111 1111
0	00 0000 0000
0.25	00 0000 0001
25	00 0110 0100
50	00 1100 1000
100	01 1001 0000
125	01 1111 0100

AD7416 预先设置的工作方式分自动测温方式和低功耗方式两种，本设计采用低功耗方式。当需要对环境温度进行测量时，通过 I^2C 串行接口总线来写入操作命令，此时，芯片将由睡眠状态转入测温状态。当温度量化转换结束后，芯片将重新转入睡眠状态。

AD7416 内部的配置寄存器（地址 01H）为 8 位读/写寄存器，数据 D7 ~ D5 始终设置为 000；D4 和 D3 用于设置故障排队长度，以防止测温系统在受到干扰时错误地触发过温指示器（OTI），故障排队长度可分别设置为 1、2、4 和 6 次；D2 用于设置 OTI 的输出极性。0 表示低电平输出，1 表示高电平输出；D1 用于设置 OTI 的工作方式，0 表示采用比较方式工作，1 表示采用中断方式工作。D0 用于设置工作方式，0 表示采用自动测温方式，1 表示采用低功耗方式。

AD7416 的感温器件在芯片内部，因此芯片表面要与被测物体紧密接触，这一点在结构设计时一定要注意。温度传感器校准方法十分简单，将单片温度监控系统集成电路 AD7416 置入高低温箱中，在微控制器的控制下，即可将所测温度值一一读出，将此值与标准值进行比较即可得到该传感器的技术指标。

二、无线收发电路

无线收发电路如图 2.20－2 所示。

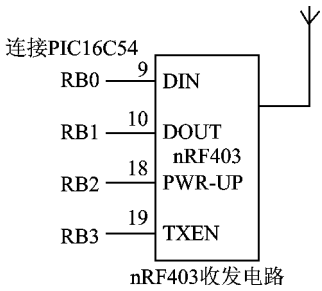


图 2.20－2 无线收发电路

采用 nRF403 单片射频收发芯片[2]，芯片内包含有发射功率放大器、低噪声接收放大器、晶体振荡器、锁相环、压控振荡器、混频器等电路。工作频率 433 MHz，FSK 调制解调，采用晶体振荡和 PLL 频率合成技术，接收灵敏度为 - 105 dBm，发射功率为 10 dBm，待机状态电流消耗仅 10 μA。

在接收模式中，射频输入信号被低噪声放大器放大，经由混频器变换，这个被变换的信号在送入解调器之前被放大和滤波，经解调器解调，解调后的数字信号在 DOUT 端输出。在发射模式中，压控振荡器的输出信号是直接送入到功率放大器，DIN 端输入的数字信号被频移键控后馈送到功率放大器输出。

芯片引脚 9 脚 DIN 输入数字信号，与微控制器的 RB0 相连，需要发射的数字信号通过 DIN 输入；10 脚 DOUT 输出数字信号，与微控制器的 RB1 相连，解调出来的信号经过 DOUT 输出进入微控制器；18 脚 PWR－UP 电源开关控制，与微控制器的 RB2 相连：PWR－UP =“1”为工作模式，PW-RUP =“0”为待机模式。待机模式电路进入待机（睡眠）状态，工作电流 8 μA，在待机（睡眠）状态电路不接收和发射数据。19 脚 TXEN 为发射允许控制，与微控制器的 RB3 相连：TXEN =“1”为发射模式；TXEN =“0”为接收模式。接收模式转换为发射模式的转换时

间至少 1 ms；发射模式转换为接收模式的转换时间至少 3 ms。

印制电路板（PCB）的设计直接关系到射频性能。PCB 使用 1.6 mm 厚的 FR－4 双面板，分元件面和底面。PCB 的底面有一个连续的接地面，射频电路的元件面以 nRF403 为中心，各元器件紧靠其周围，尽可能减少分布参数的影响。元件面的接地面保证元件充分的接地，大量的通孔连接元件面的接地面到底面的接地面。nRF403 采用 PCB 天线，在天线的下面没有接地面。射频电路的电源使用高性能的射频电容去耦，去耦电容尽可能地靠近 nRF403 的 VDD 端，在较大容量的表面安装的电容旁还应并联一个小数值的电容。射频电路的电源与控制电路的电源分离，nRF403 的 VSS 端直接连接到接地面。注意不能将数字信号或控制信号引入到 PLL 回路滤波器元件上。

三、微控制器控制电路[3～5]

微控制器选用 PIC16C54。系统采用 LP 低频低功耗晶体振荡方式，地址编码开关用于传感器的编号，RB 口分别与 AD7416 和 nRF403 的引脚相连，用于温度测量和无线收发控制。由于无线温度传感器采用电池供电，整个电路的低功耗控制十分重要。

无线库房温湿度测量控制系统采用查询方式对各点的温度进行测量，各传感器在未被查询时可处于低功耗睡眠状态。微控制器控制程序如图 2.20－3 所示。

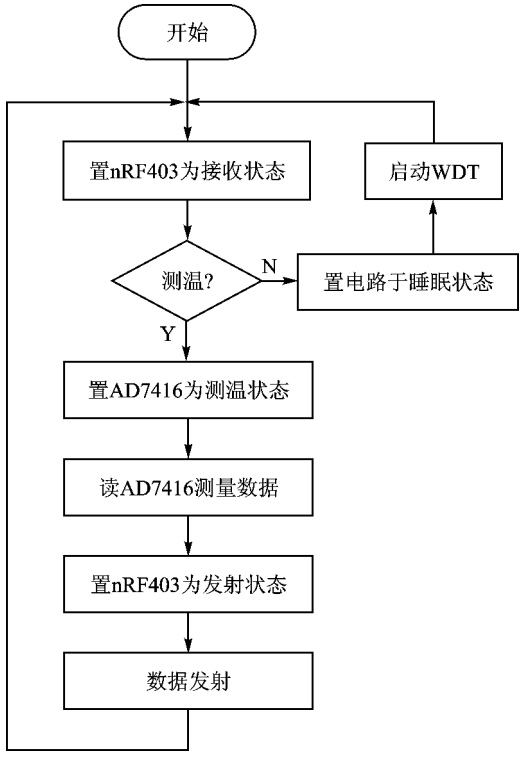


图 2.20－3 微控制器控制程序

应特别注意的是印制电路板设计时，妥善地处理数字电路与射频电路的元件、信号线与地线的布置十分重要。

四、结束语

所设计的无线数字温度传感器采用专用的集成电路,电路结构简单,工作稳定可靠。设计中充分地利用了各芯片的低功耗特性,有效地延长了电池的使用时间;无线数据传输使用方便灵活。在烟叶、粮食等仓库中应用效果良好。改进的设计是将温度和湿度的测量结合在一起构成无线数字温湿度传感器。

参 考 文 献

1 Analog Devices Inc. 10 - Bit Digital Temperature Sensors AD7416 Datasheet [DB/OL]. http://www.analog.com. 1999 - 01 - 01

2 Nordic VLSI ASA Inc. 315/433 MHz Single Chip RF Transceiver nRF403 Datasheet [DB/OL]. http://www.nvlsi.no. 2001 - 07 - 01

3 俞光昀. PIC 系列单片机开发应用技术 [M]. 北京:电子工业出版社,2000

4 黄智伟. 无线串行接口电路设计 [J]. 电测与仪表,2001,37 (7):30 ~ 33

5 黄智伟. PC 机无线串行接口电路及程序设计 [J]. 电子质量,2002,15 (3):106 ~ 109

选自《传感器技术》月刊,2002 年第 9 期

2.21 液晶屏温度响应特性及其温度控制

安徽机电学院电气工程系 (241000) 王元庆
芜湖电真空研究所第三研究室 (241000) 董 戴

一、引 言

液晶显示器由于其质量轻、体积小、有效显示面大等优点而日益受到各国军方的重视,其中有源阵列液晶显示器 (AM - LCD) 正越来越广泛地应用到军事领域各兵种的军械装备上。美国军方甚至决定,在今后的军用器械中不再采用 CRT 作为显示终端,而全部采用平板显示器件,特别是液晶显示器。美军主力战机 F22 飞行座舱的主多功能显示器 (PMFD)、辅多功能显示器 (SMFD)、正前方控制显示器 (UFCD) 共 5 处功能显示器均采用了液晶显示器。液晶显示器在军械中的应用将越来越普遍,这是不争的事实。

由于液晶的一些参数如粘度在低温条件下发生着很大的变化,导致液晶的反应速度下降,在极低的温度下 (- 25 以下),液晶甚至无法工作。液晶显示器应用于军械装备必须解决其这种低温下无法工作的问题,目前,多采用局部加热的手段,使液晶始终处在合适的工作温度下。我们为此作了一定的前期研究并取得了一定的成果。本文将在分析液晶屏温度响应特性的基础上,介绍利用具有控制量自修正能力的模糊控制技术对液晶屏温度加以控制的办法。

二、液晶屏的动态响应特性

设初始状态下,屏幕的内外表面初始温度为 t_i,屏幕包括液晶的温度均为 t_i。假设在 τ_0 时刻,屏的外表面 (即自然环境) 的温度出现一个阶跃变化至 t_n,并假定 t_n 大于 t_i。屏幕内部由外及内,温度将不断升高,随着时间的延长温度升高的情形将不断向内层发展,由于外界环境温度保持不变,经过很长一段时间以后,屏内层的温度变化梯度最终都相同,内壁各点的导入热量等于导出的热量,温度将不再发生变化,屏壁由非稳态过程过渡到了稳态过程。

屏壁的非稳态响应可以用导热微分方程加以描述:

$$\frac{\partial t}{\partial \tau} = \alpha \nabla^2 t \tag{1}$$

其中,导温系数

$$\alpha = \lambda / (\rho c) \tag{2}$$

ρ、c 分别为屏壁的材料 (玻璃) 密度和比热容。∇^2 为拉普拉斯算子:

$$\nabla^2 = \left(\frac{\partial^2 t}{\partial x^2} + \frac{\partial^2 t}{\partial y^2} + \frac{\partial^2 t}{\partial z^2} \right) \tag{3}$$

一般来说,液晶屏的外部温度场可以视为均匀的,屏内表面所处的环境同样可以视为具有均匀的温度分布。因此,对屏壁的温度响应分析可以简化为一维温度响应分析,微分方程 (1) 可以简化为:

$$\partial t/\partial \tau = \alpha(\partial^2 t/\partial x^2) \tag{4}$$

初始条件：
$$\tau = 0, t = t_i \tag{5}$$

边界条件：
$$x = 0, \partial t/\partial x = 0 \tag{6}$$

$$x = \delta, \partial t/\partial x = -\alpha(t_n - t) \tag{7}$$

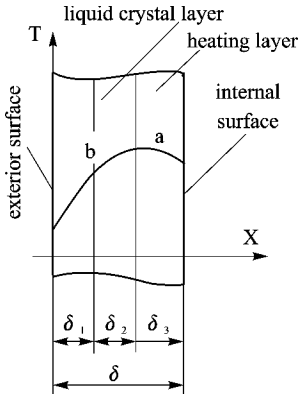


图 2.21－1 屏的横向坐标设定

其坐标系设置如图 2.21－1 所示，X 轴为屏的厚度方向。δ（=3.2 mm）为 LCD 屏内外表面之间的厚度，δ_1（=1.4 mm）为液晶盒前玻璃板厚度（含偏振片厚度），δ_2（=1.3 mm）为液晶盒后玻璃板厚度（含偏振片厚度），δ_3（=0.5 mm）为平板加热器厚度（基底材料也是玻璃）。为了简化微分方程的求解，首先将微分方程（4）式及初始条件和边界条件式（5）～（7）无量纲化，使微分方程式的解所包含的变量个数减少。

令过余温度 $\theta = t - t$，再引入无量纲过余温度 Θ 和无量纲坐标变量 X，并且

$$\Theta = \theta/\theta_i = (t - t)/(t_i - t) \tag{8}$$

$$X = x/\delta \tag{9}$$

同时定义无量纲时间 F_0：
$$F_0 = \alpha\tau/\delta \tag{10}$$

以及无量纲热阻 B_i：
$$B_i = \alpha\delta/\lambda \tag{11}$$

则微分方程（4）及初始条件和边界条件可简化为：

$$\partial\Theta/\partial F_0 = (\partial^2\Theta/\partial X^2) \tag{12}$$

$$F_0 = 0, \Theta = \Theta_i = 1 \tag{13}$$

$$X = 0, \partial\Theta/\partial X = 0 \tag{14}$$

$$X = 1, \partial\Theta/\partial X = -B_i\Theta_n \tag{15}$$

对液晶屏的非稳态温度响应的分析建立在微分方程（12）的解的基础上，结合初始条件式（13）～（15）求解可得微分方程式（12）的定值解为：

$$\Theta = 2\sum_{i=1}^{n} e^{-\mu_i^2 F_0} \frac{X\sin\mu_i\cos\mu_i}{\mu_i + \sin\mu_i\cos\mu_i} \tag{16}$$

式中 μ_i 由 $\mu/B_i = \mathrm{ctg}\mu$ 确定。

3 层结构的液晶屏其各层均是玻璃材料，玻璃的导温系数 α、导热系数 λ、材料密度 ρ、比热容 c 分别为 3.4×10^{-7}（m^2/s）、0.76（W/m·　）、2707（kg/m^3）、0.8（kJ/kg·　）。另外，屏壁厚 δ 为 3.1×10^{-3}m。

常压（$1.013\,25\times10^5$ Pa）下，干空气和干饱和水蒸气的换热系数 α 与温度的关系如图 2.21－2所示。

根据上述已知条件，以干空气的外界环境为例，可得图 2.21－3 所示的液晶盒温度阶跃响应特性曲线。图中，温度响应从约 6 s 之后开始，这主要是由于温度传感器（具有一阶动态特性）的滞后引起的。从图 2.21－3 可见，后表面温度达到前表面温度的 63.28% 所需要的时间（可定义为温度响应时间常数）约为 15.1 s，实际测量的时间与理论推导的时间十分接近，存在误差的客观原因可能是实际的液晶显示屏中液晶盒前后粘贴了约 0.1 mm 厚偏振片，偏振片（在聚乙烯基片上镀制的多层膜系）的热物理性质与玻璃不同。

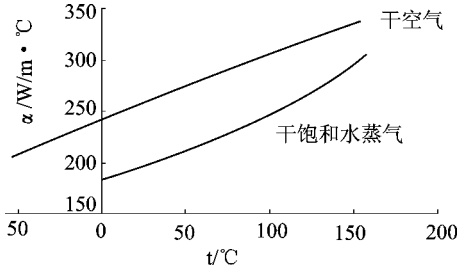


图 2.21－2 干空气和干饱和水蒸气的换热系数与温度的关系

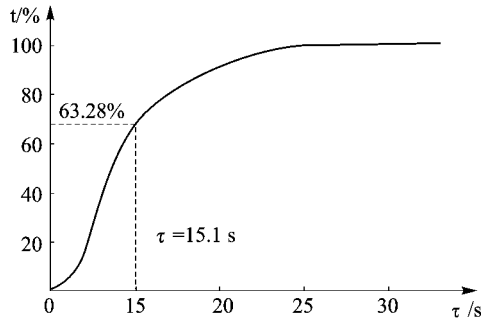


图 2.21－3 内表面温度响应实验结果

事实上，液晶显示器可能还具有本质的非线性特性、参数时变性，用式 (1) 描述液晶显示器温度响应模型是不完全准确的，准确确定其模型是比较困难的，因此采用常规的运算控制方法如 PID 控制方法不易达到好的控制效果。

三、模糊控制算法

模糊控制方法在许多工业控制场合均得到了良好的应用。在我们所设计的系统中，温度控制必须具有良好的稳定性、温度波动小、系统结构应尽可能小。不过由于被控制对象 (即液晶屏) 直接暴露在具有非线性扰动特点的外界工作环境下 (冷或热空气)，温度变化复杂 (正向温度冲击 ≤ + 20 /min、负向温度冲击 ≤ | - 5 | /min)，环境温度变化范围大 (- 55 ~ + 71)；由于温度测量的严重滞后特性，对温度控制带来不可预测的影响。因此，常规的模糊控制方法不能取得良好的控制效果。结合液晶显示器的实际工作背景，我们对温度控制方法作了进一步的研究，制定了独特的具有控制量修正功能的模糊控制方法解决方案，并取得了良好的效果。

具有控制量自修正能力的模糊控制器的结构如图 2.21－4 所示。控制器以屏温误差 e、误差变化率 ec、环境温度 T_a 和环境温度变化方向 ΔT_a 四个参量作为输入。其中屏温误差 e、误差变化率 ec 作为模糊推理的输入，为模糊推理提供基本输入值；环境温度 T_a、环境温度变化量 ΔT_a 作为控制量修正模块的输入，为控制量修正提供依据。如图 2.21－4 所示，模糊控制器按照模糊控制规则得出作为增量式控制输出量的初始值 F_0，初始控制输出量 F_0 输入控制量修正模块。将环境温度 T_a、环境温度变化量 ΔT_a 作为依据，控制量修正模块根据大量的试验经验得到的预备知识对初始控制量加以增量式修正。

控制量修正模糊控制方法其基本控制理论是模糊控制技术，而模糊控制的输出量受到特

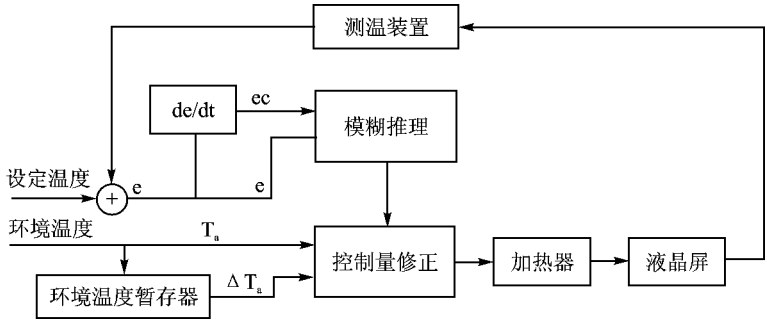


图 2.21－4 模糊控制器结构

定的修正,输出量可近似地看作是模糊控制与修正规则的与。

设按照模糊控制算法得到的控制量偏移值为 $\Delta U_F\ (T_{s2})$、控制量修正算法得到的控制量为 $U_L\ (T_a)$,则实际控制量 $U_O\ (T_{s1}, T_{s2}, T_a)$ 为:

$$U_O\ (T_{s1}, T_{s2}, T_a) = [U_{F0}\ (T_{s1}) + \Delta U_F\ (T_{s2})] \wedge U_L\ (T_a) \tag{17}$$

式中 $U_{F0}\ (T_{s1})$ ——上次输出的控制量;

T_{s1} ——上次测量的屏温度;

T_{s2} ——此次测量的屏温度;

T_a ——此次依据的环境温度。

符号"∧"表示与的逻辑关系,或者可看作并的数学集合运算。

从关系式(17)可以看出,模糊控制算法得到的控制量为偏移输出,控制量修正算法得到的控制量为位移输出式。在随后的内容中将详细介绍这两个控制的计算方法和过程。式(17)中的模糊控制输出量 $\Delta U_F\ (T_{s2})$ 按照当前液晶屏温度偏差值 ΔT_{s2}、液晶屏温度变化率 δ 两个因素决定。其中液晶屏温度值 T_{s2} 每 8 s 采集一次,同时计算液晶屏温度变化率 δ,每 1 min 刷新一次环境温度值 T_a,实际的模糊决策可以用模糊语言和模糊决策表两种形式表达。

根据从实际环节中总结出来的液晶屏温度控制经验,得到模糊决策规则如表 2.21－1 所列。

表 2.21－1 模糊控制规则表

U E / EC	> =4	3	2	1	0	- 1	- 2	- 3	- 4	< = - 5
1	- 3	- 2	- 1	- 0.5	0	0	0	0.5	1	1
0.5	- 2	- 1	- 1	0	0	0	0.5	1	1	2
0	- 2	- 1	- 1	0	0	0	1	1	2	3
- 0.5	- 1	- 1	- 0.5	0	0	0	1	1	2	2
- 1	- 1	- 0.5	0	0	0	0.5	1	2	3	4

四、温度控制效果实测

控制量修正模块的修正曲线由大量实验得出。它的作用是根据环境温度值以及环境温度变化量决定对输出量进行修正与否、修正大小和方向。如图 2.21－5 所示为修正量的范围与环境温度的关系曲线,控制量修正模块的修正强度由该曲线决定。

环境温度值以及变化量和变化方向都是十分有用的参量。环境温度值 T_a 每隔 8 s 测量一次，连续测量 8 次后求得平均值，并用计算的平均值替代原先的环境温度值作为新的环境温度值。这样确定环境温度值的办法具有以下优点：(1) 避免测量误差带来的对温度控制的不利影响；(2) 克服了环境温度不断波动、扰动对温度控制的干扰；(3) 减小了由于瞬间的环境冲击温度脉冲诱发的温度控制振荡，这一点已被实验所证明。

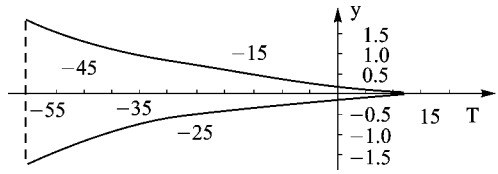
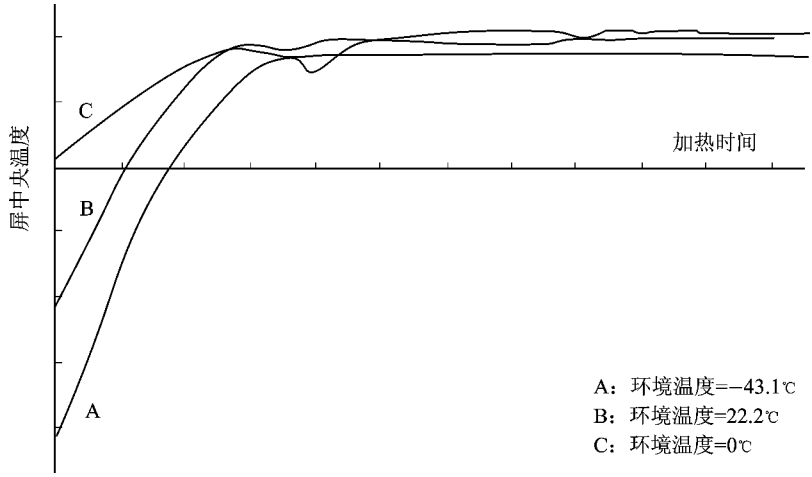
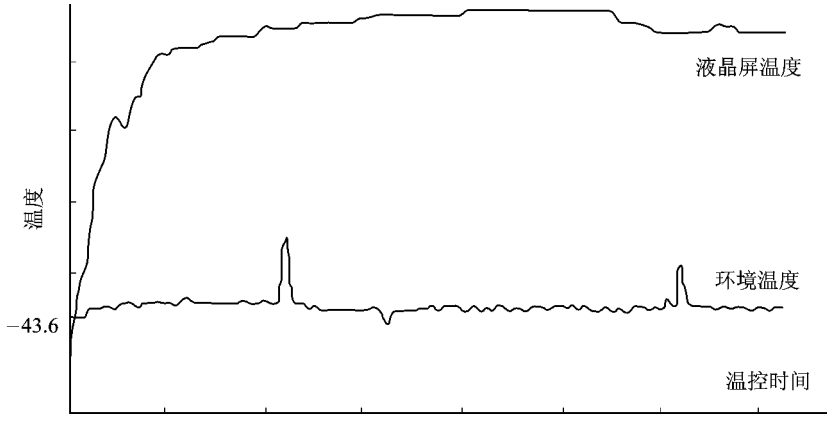


图 2.21－5　修正量的范围与环境温度的关系曲线

图 2.21－6 (a) 为 3 个环境温度下温度过渡过程实测曲线，图 2.21－6 (b) 为某一环境温度下温度的控制曲线，图中同时标出了环境温度及其变化时间及强度。图 2.21－6 (b) 中环境温度过渡较大，并伴有脉动，脉动形成的办法是有意打开低温箱，使试验箱内的温度出现快速上升。从图中不难看出，液晶屏的温度过渡过程和保持过程具有良好的稳定性，基本克服了环境温度变化对液晶屏温度控制所带来的影响问题。



(a) 温度过渡过程实测曲线



(b) 环境温度及其变化对温度控制的影响

图 2.21－6

五、结束语

利用模糊控制技术可以实现对液晶屏的液晶层温度的良好控制,使液晶屏在不同的环境温度下具有优良的图像显示性能,从而达到低温乃至超低温环境下显示器必须稳定工作的军用要求。本项目业已完成,并在我军某重点工程中得到良好的应用。

参 考 文 献

1 王元庆. 液晶屏非稳态温度响应特性的理论分析 [J]. 安徽机电学院学报,2000,15 (3):1 ~ 4

2 王元庆. 液晶显示屏的液晶层温度测量方法的研究 [J]. 仪器仪表学报,2001,22 (6):632 ~ 635

3 王元庆,等. 液晶显示屏温度特性的试验研究 [J]. 电子测量与仪器学报,2001,15 (4):52 ~ 55

4 FARRELLJF,WRIGHTJC. "High performance AMLCD designs for military/severe environment" [C] Proceedings of the 1995 2nd Int. Workshop on AML CD. 123 ~ 124

5 BUFORD, T. GEORGE. S. , ALTADONNA, LYNN P. ;BLOWERS. DAN. F - 15 air naviation mutiple indicator prototype development program [C] Proceedings of SPIE - The I. S. O. E, 1995, 2462:211 ~ 216

选自《计算机测量与控制》月刊,2002 年第 4 期

2.22 CPU 卡的接口特性、传输协议与读写程序设计

北京恒基伟业产品开发部 陈 峰
国瑞数码安全有限公司工程部 尹 寒

IC 卡的概念是 20 世纪 70 年代提出的。法国 BULL 公司首创 IC 卡产品,并将这项技术应用到金融、交通、医疗、身份证明等多个方面。IC 卡的核心是集成电路芯片,一般为 3 μm 以下的半导体技术制造。IC 卡具有写入数据和存储数据的能力。IC 可存储其中的内容,根据需要可以有条件地供外部读取,或供内部信息处理或校验用。

根据各种集成电路的不同,IC 卡可以分为以下三类:存储器卡、逻辑加密卡与 CPU 卡。其中,存储器卡仅有数据存储能力,没有安全措施;逻辑加密卡仅有几个字节的密码,卡中有一个错误计数器,如果指定次数验证密码失败,则卡中数据被自动锁死,该卡数据不能再更改;CPU 卡是这三类 IC 卡中最高级的卡,一般有 ROM、RAM 和 EEPROM 三种存储器。ROM 中存放的是程序,程序是为 IC 卡的 CPU 专门设计的,用来解释读写器终端送来的命令。IC 卡应用系统根据应用需要由终端送一系列命令到 CPU 卡,通过改变命令的内容和命令的顺序就可以满足不同的需要,因此有较高的灵活性;同时,因为 CPU 有计算功能,存储容量又大,可以进行比较复杂的加密/解密运算,极大提高了安全性。EEPROM 主要用来存放一些应用数据,其容量比逻辑加密卡大,可实现一卡多用,是目前最安全的卡类型。因此,CPU 卡是目前 IC 卡的重要发展方向之一。

一、CPU 卡的接口特性

1. 触点定义

触点的定义遵循 ISO7816－2 的规定,如图 2.22－1 所示。符号说明如表 2.22－1 所列。

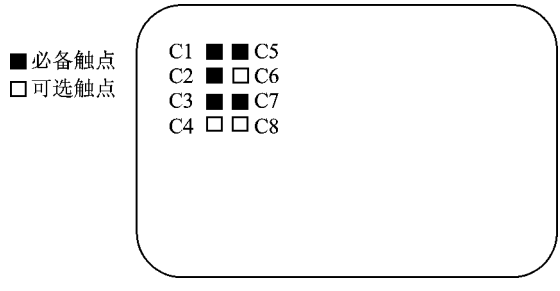


图 2.22－1 CPU 卡的触点

表 2.22－1 符号说明

符号	说明	符号	说明
C1	电源电压（V_{CC}）	C5	地（GND）
C2	复位信号（RST）	C6	不使用
C3	时钟信号（CLK）	C7	输入输出（I/O）
C4	不使用	C8	不使用

2. 字符帧

数据在 I/O 上以图 2.22－2 所示的字符帧方式传输。

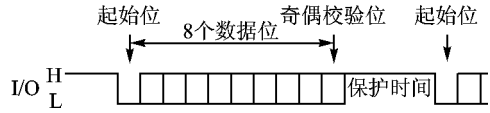


图 2.22－2 字符帧传输方式

每个位宽是 1 个 etu，etu = 372/f。在此处，f = 3.57 MHz。

起始位由接收端通过对 I/O 周期采样获得，采样周期应小于 0.2 etu。2 个连续字符起始位上升沿之间的间隔时间等于（10 ± 0.2）etu 加上 1 个保护时间（最少 2 个 etu）。在保护时间内，卡与终端都应处于接收模式（I/O 为高电平状态）。如果卡或终端作为接收方检测出奇偶错误，则 I/O 被置为低电平，以向发送方表明出现错误。

3. 卡操作

卡操作的步骤如下：

① 将卡插入终端接口设备，使两者的触点相接并激活触点；

② 将卡复位，建立卡与终端间的通信；

③ 执行操作；

④ 释放触点，并从接口设备取出卡片。

以下是除第③步（执行操作）以外，各步的时序要求。

（1）触点激活

时序如图 2.22－3 所示。

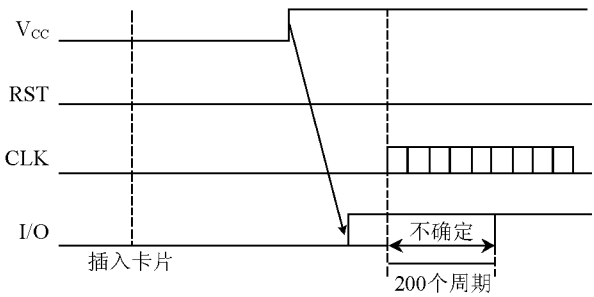


图 2.22－3 触点激活时序

⑵ 卡复位

卡利用低电平复位来完成异步复位应答，随着触点的激活，终端将进行一个冷复位并从卡获得复位应答。冷复位时序如图2.22－4所示。

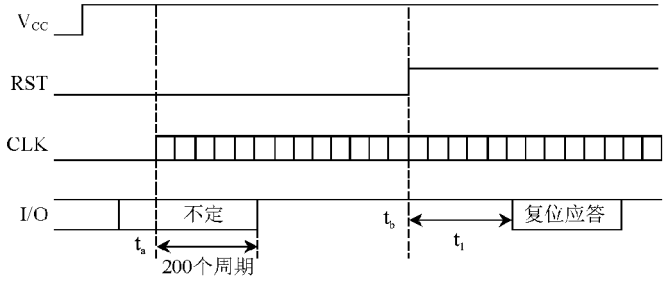


图2.22－4 冷复位时序

冷复位过程之后，如果收到的复位应答信号不满足标准的规定，终端将启动一个热复位并从卡获得复位应答。热复位时序如图2.22－5所示。

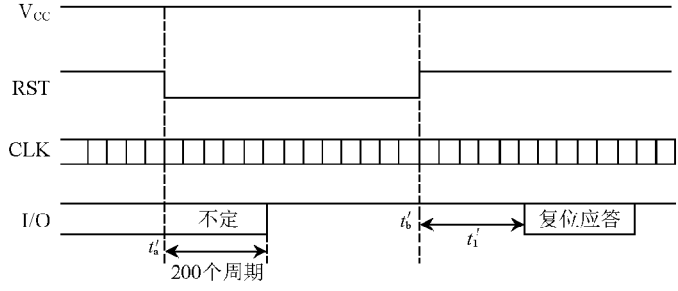


图2.22－5 热复位时序

在实际程序设计时，由Reset子程序实现触点激活和卡复位。

⑶ 触点释放时序

触点释放时序过程如图2.22－6所示。有关触点释放的子程序见本刊网站：www.dpj.com.cn。

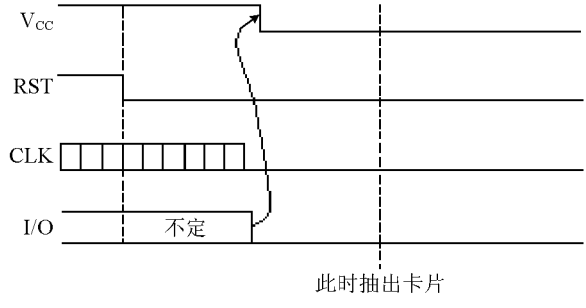


图2.22－6 触点释放时序

二、传输协议与卡命令处理程序

ISO7816－4 及中国金融集成电路(IC)卡规范所规定的异步半双工传输协议,是关于终端为实现传输控制和特殊控制而发出的命令的结构及其处理过程。包括了两种协议:字符传输协议(T = 0)和块传输协议(T = 1)。本文着重讨论字符传输协议(T = 0),它是 IC 卡推荐使用的通信协议。

1. 命 令

命令包含 1 个连续 4 字节的命令头,用 CLA、INS、P1 和 P2 以及 1 个可变长度的条件体来表示。

命令头定义如下:

· CLA 指令类别,除"FF"外的任何值。

· INS 在指令类别中的指令码,当最低位是"0",并且高位半字节既不是"6"也不是"9"时,INS 才有效。

· P1、P2 完成 INS 的参数字节。

条件体定义如下:

· Lc(发送数据长度)占 1 个字节,在命令中定义为发送数据的字节数,取值范围是 1～255。

· Data 为将要发送的命令数据域,字节数由 Lc 定义。

· Le(接收数据长度)占 1 个字节,指出命令响应中预期的数据最大字节数。Le 的取值范围是 0～255。如果 Le = 0,预期数据字节的最大长度是 256。

表 2. 22－2

情况	结 构
1	CLA INS P1 P2
2	CLA INS P1 P2 Le
3	CLA INS P1 P2 Lc Data
4	CLA INS P1 P2 Lc Data Le

可能的命令结构的 4 种情况定义如表 2. 22－2 所列。

命令全部由终端应用层(TAL)初始化。它通过终端传输层(TTL)向卡发送 1 个由 5 个字节组成的命令头,并等待一个过程字节。

2. 过程字节

卡收到命令后,紧接着返回一个过程字节给 TTL,指明下一步该作什么,如表 2. 22－3 所列。

表 2. 22－3

序号	过程字节值	步 骤
(1)	与 INS 字节相同	所有余下的数据将由 TTL 传送或 TTL 将准备接收来自卡所剩的数据
(2)	"60" "60X"或"9X",除"60"之外(过程字节或状态码 SW1)	TTL 将提供额外工作等待时间
(3)		TTL 将等待下一个过程字节或状态码 SW2

在(1)、(2)情况中,TTL 完成动作后将等待另一个过程字节。在(3)情况中,第二个过程字节或状态码(SW2)被收到后,TTL 将做以下事情:

· 如果过程字节为"61",TTL 将发送一个最大长度(P3)为"XX"的得到响应命令(GET

RESPONSE)给卡,“XX”为 SW2 的值。GET RESPONSE 命令仅适用于 T = 0 协议。命令报文的结构如表 2.22－4 所列。

· 如果过程字节为“6C”,TTL 将立即重发前一个命令的命令头给卡,它的 P3 值用“XX”代替。“XX”是 SW2 的值。

· 如果过程字节是“6X”(除“60”、“61”及“6C”之外)或“9X”,与前两者 TTL 自己处理不同,TTL 将通过命令响应返回状态码给上一层——终端应用层(TAL),由 TAL 处理,并等待下一个命令。

表 2.22－4

CLA	“0x”,x 指明通道号
INS	“C0”
P1	“00”
P2	“00”预期
Le	预期数据的最大长度

3. 卡命令处理程序流程图

图 2.22－7 是卡命令处理程序,即终端与卡的信息交互过程的流程图,具体程序见本刊网站。

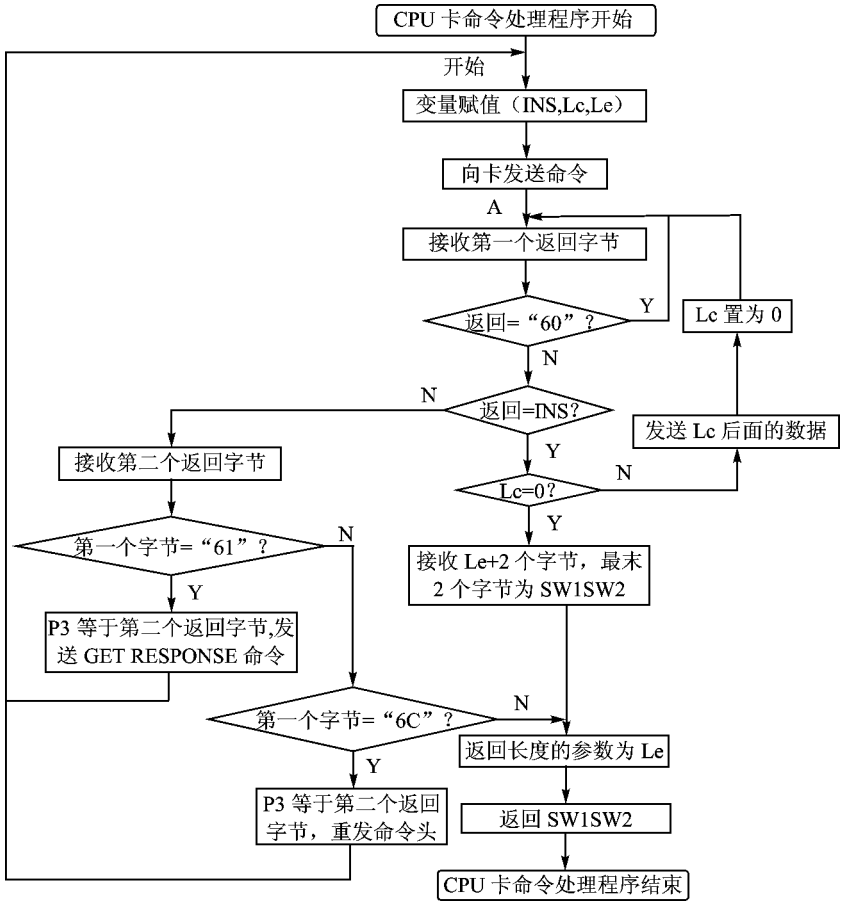


图 2.22－7 卡命令处理程序流程图

以下是引脚说明。

ECPU :决定卡的 CLK 触点上是否有 CLK 信号的引脚;

ICVCC 终端与卡的 V_{CC} 触点相接触的引脚；

ICIO 终端与卡的 I/O 触点相接触的引脚；

ICCLK 终端与卡的 CLK 触点相接触的引脚；

ICRST 终端与卡的 RST 触点相接触的引脚。

以下是程序中函数介绍。

· void isr_timer1 (void) interrupt 3：定时器 1 中断子程序，用于按位发送和接收数据字节；

· void Snd (void)：发送数据子程序，由定时器 1 实现；

· void Rcv (BYTE bytenr，BYTE * Buffer) 接收数据子程序 (参数含义分别是 接收数据字节数、接收数据的存放处)，由定时器 1 实现；

· 函数 1 void Reset (BYTE * len，BYTE * resp)：复位子程序 (参数含义分别是 返回复位响应数据的长度、复位响应数据)；

· 函数 2 void Power_off (void) 触点释放程序；

· 函数 3 WORD CPUC_Cmd (BYTE len，BYTE * comm，BYTE * lenr，BYTE * resp)：CPU 卡命令子程序 (参数含义分别是 发送命令长度、发送命令、返回字节数、返回数据)，函数返回状态字节 SW1SW2。

注 篇幅所限，读写程序代码及注释部分见本刊网站：www. dpj. com. cn

参 考 文 献

1 全国标准化技术委员. 中国金融集成电路 (IC) 卡规范 (V1. 0). 北京 中国金融出版社

2 中国华大集成电路设计中心. CIU9102 智能卡接口特性和通信规程

选自《单片机与嵌入式系统应用》月刊，2002 年第 3 期

2. 23　一种基于铁电存储器的双机串行通信技术

温州大学　陈　冲

随着电子技术的飞速发展,单片机也步入一个新的时代,越来越多的功能各异的单片机为我们的设计提供了许多新的方法与思路。对于某一些场合,比如:复杂的后台运算及通信与高实时性前台控制系统、软件资源消耗大的系统、功能强大的低功耗系统,加密系统等等,如果合理使用多种不同类型的单片机组合设计,可以得到极高灵活性与性能价格比,因此,多种异型单片机系统设计渐渐成为一种新的思路,但单片机之间的通信一直是困扰这种方法拓展的主要问题。本文将分析比较几种单片机之间的通信方式、难点,并提出一种解决方案。

一、几种常用单片机之间的通信方式

1. 采用硬件 UART 进行异步串行通信

这是一种占用口线少,有效、可靠的通信方式;但遗憾的是许多小型单片机没有硬件 UART,有些也只有 1 个 UART,如果系统还要与上位机通信的话,硬件资源是不够的。这种方法一般用于单片机有硬件 UART 且不需与外界进行串行通信或采用双 UART 单片机的场合。

2. 采用片内 SPI 接口或 I^2C 总线模块串行通信

SPI/I^2C 接口具有硬件简单、软件编程容易等特点,但目前大多数单片机不具备硬件 SPI/I^2C 模块。

3. 利用软件模拟 SPI/I^2C 模式通信

这种方式很难模拟从机模式,通信双方对每一位要做出响应,通信速率与软件资源的开销会形成一个很大的矛盾,处理不好会导致系统整体性能急剧下降。这种方法只能用于通信量极少的场合。

4. 口对口并行通信

利用单片机的口线直接相连,加上 1 ~ 2 条握手信号线。这种方式的特点是通信速度快,1 次可以传输 4 位或 8 位,甚至更多,但需要占用大量的口线,而且数据传递是准同步的。在一个单片机向另一个单片机传送 1 个字节以后,必须等到另一个单片机的接收响应信号后才能传送下一个数据。一般用于一些硬件口线比较富余的场合。

5. 利用双口 RAM 作为缓冲器通信

这种方式的最大特点就是通信速度快,两边都可以直接用读写存储器的指令直接操作;但这种方式需要大量的口线,而且双口 RAM 的价格很高,一般只用于一些对速度有特殊要求的场合。

从上面几种方案来看,各种方法对硬件都有很大的要求与限制,特别是难以在功能简单的单片机上实现,因此寻求一种简单、有效的,能在各种单片机之间通信的方法具有重要的意义。3、4 方案中,双方单片机必须对要传递的每一位或每一个字节做出响应,通信数据量较大时会耗费大量的软件资源,这在一些实时性要求高的地方是不允许的。针对这一问题,假设在单片

机之间增加 1 个数据缓冲器,大批数据先写入缓冲区,然后再让对方去取,各个单片机对数据缓冲器都是主控模式,这样必然会大大提高通信效率。谈到数据缓冲,我们马上会想到并行 RAM,但是并行 RAM 需要占用大量的口线(数据线 + 地址线 + 读写线 + 片选线 + 握手线),一般在 16 条以上。这是一个让人望而生畏的数字,而且会大大增加 PCB 面积并给布线带来一定的困难,极少有人采用这种方式。串行接口的 RAM 在市场上很少见,不但难以买到而且价格很高。移位寄存器也可以做数据缓冲器,但目前容量最大的也只有 128 位,因为是"先进先出"结构,所以不管传递数据多少,接收方必须移完整个寄存器,灵活性差而且大容量的移位寄存器也是少见难买的。一种被称为"铁电存储器"芯片的出现,给我们带来了解决方法。

二、利用铁电存储器作为数据缓冲器的通信方式

铁电存储器是美国 Ramtran 公司刚刚推出的一种新型非易失性存储器件,简称 FRAM。与普通 EEPROM、Flash-ROM 相比,它具有不需写入时间、读写次数无限,没有分页结构可以连续写入的优点,因此具有 RAM 与 EEPROM 的双重特性,而且价格相对较低。现在大多数的单片机系统配备串行 EEPROM(如 24CXX、93CXX 等)用来存储参数。如果用 1 片 FRAM 代替原有 EEPROM,使它既能存储参数,又能作串行数据通信的缓冲器。2 个(或多个)单片机与 1 片 FRAM 接成多主-从的 I^2C 总线方式,增加几条握手线,即可得到简单高效的通信硬件电路。在软件方面,只要解决好 I^2C 多主-从的控制冲突与通信协议问题,即可实现简单、高效、可靠的通信了。

三、实例(双单片机结构,多功能低功耗系统)

1. 硬 件

W78LE52 与 EMC78P458 组成一个电池供电、可远程通信的工业流量计。78P458 采用 32.768 kHz 晶振,工作电流低,不间断工作,实时采集传感器的脉冲及温度、压力等一些模拟量;W78LE52 采用 11.059 2 MHz 晶振,由于它的工作电流较大,采用间断工作,负责流量的非线性校正、参数输入、液晶显示、与上位机通信等功能,它的 UART 用于远程通信。通信接口部分线路如图 2.23-1 所示,2 个单片机共用 1 片 I^2C 接口的 FRAM(FM24CL16)组成二主一从的 I^2C 总线控制方式,W78LE52 的 P3.5、P3.2 分别与 78P458 的 P51、P50 连接作为握手信号线 A 与 B。我们把握手线 A(简称 A 线)定义为总线控制、指示线,主要用于获取总线控制权与判别总线是否"忙";握手线 B(简称 B 线)定义为通知线,主要用于通知对方取走数据。

2. I^2C 总线仲裁

由于我们采用的是二主一从的 I^2C 总线方式,因此防止 2 个主机同时去操作从机(防冲突)是一个非常重要的问题。带有硬件 I^2C 模块的器件一般是这样的,器件内部有 1 个总线仲裁器与总线超时定时器:当总线超时定时器超时后指示总线空闲,这时单片机可以发出获取总线命令,总线仲裁器通过一系列操作后确认获取总线成功或失败;超时定时器清零,以后的每一个 SCL 状态变化对总线上所有主机的超时定时器进行清零,以阻止它溢出,指示总线正处于"忙"状态,直到一个主机对总线控制结束不再产生 SCL 脉冲,超时定时器溢出,总线重新回到"空闲"状态。但是目前大多数单片机没有配备硬件 I^2C 模块,而且当 2 个主机的工作频率相差较大时,超时定时器定时值只能设为较大的值,这样也会影响总线的使用效率。下面介绍一种用软件模拟 I^2C 总线仲裁的方法(I^2C 读写操作程序的软件模拟十分多见,这里不再赘

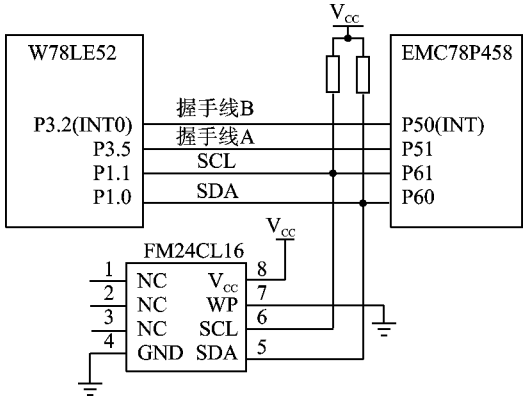


图 2.23－1

述）：用 1 条握手线 A，流程图如图 2.23－2 所示，当 A 线高电平时，指示总线空闲；当其中一个主机要获取总线控制权时，先查询总线是否空闲，“忙”则退出，空闲则向 A 线发送一个测试序列（如：1000101011001011），在每次发送位“1”后读取 A 线状态。如果读取状态为“0”，马上退出，说明有其他器件已经抢先获取总线；如果一个序列读取的 A 线状态都正确，则说明已成功获得总线控制权，这时要拉低 A 线以指示总线“忙”，直到读写控制结束重新拉高 A 线，使总线回到“空闲”状态。不同的主机采用不同的测试序列，或产生随机测试序列，测试序列长度可以选得长一些，这样可以增加仲裁的可靠性。

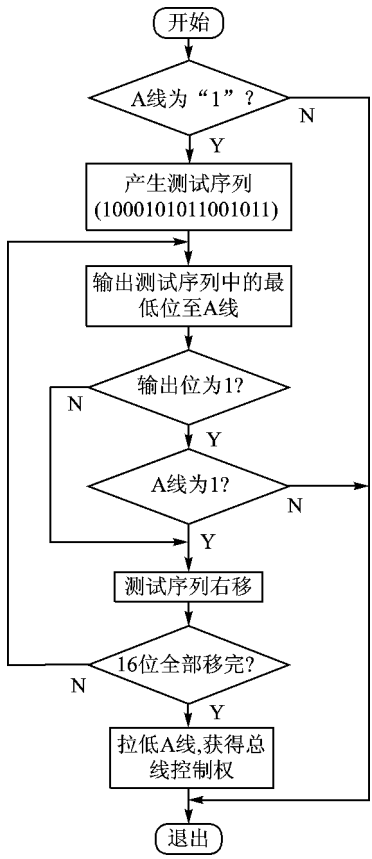


图 2.23－2

3. 通信协议

一个可靠通信体系，除了好的硬件电路外，通信协议也至关重要。在单片机系统 RAM 资源与执行速度都非常有限的情况下，一个简捷有效的协议是非常需要的。下面具体介绍一种比较适用于单片机通信的协议，数据以包的形式传送。数据包结构：

⑴ 包头——指示数据包的开始，有利于包完整性检测，有时可省略；

⑵ 地址——数据包要传送的目标地址，若只有双机通信或硬件区分地址可以省略；

⑶ 包长度——指示整个数据包的长度；

⑷ 命令——指示本数据包的作用；

⑸ 参数——需要传送的数据与参数；

⑹ 校验——验证数据包的正确性，可以是和校验、异或校验、CRC 校验等或者是它们的组合；

⑺ 包尾——指示数据包的结尾，有利于包完整性检测，有时可省略。

4. 通信流程

首先,要在 FRAM 里划分好各个区域,如各个单片机的参数区、数据接收区等。然后,单片机可以向另一个单片机发送数据包,发送完毕之后通过向握手线 B 发送 1 个脉冲通知对方取走数据;接收方读取数据并进行处理后,向 FRAM 内发送方的数据接收区写入回传数据或通信失败标志,再向握手线 B 发送 1 个脉冲回应发送方。表 2.23-3 是单片机 1 启动 1 次与单片机 2 之间的通信的例子。

表 2.23-3

步骤	单片机 1	单片机 2	A 线	B 线
1	总线空闲	总线空闲	1	1
2	获取总线控制权	其他操作	0	1
3	向 FRAM 内单片机 2 的数据接收区写入一包通信数据	其他操作	0	1
4	向 B 线发送 1 个负脉冲,通知单片机 2,启动超时定时器	其他操作	0	负脉冲
5	其他操作	响应来自 B 线的脉冲,读取 FRAM 内数据接收区的内容(无须获取总线操作)	0	1
6	其他操作	对数据进行处理后,向 FRAM 内单片机的数据接收区写入回传数据或通信失败标志	0	1
7	其他操作	向 B 线发送 1 个负脉冲,通知单片机 1	0	负脉冲
8	清超时定时器,读取数据内容。如果失败可以做重发或其他处理;如果成功则拉高 A 线,释放 I^2C 总线,1 次通信结束	其他操作	1	1
9	如果超时定时器溢出,说明单片机 2 没有响应单片机 1 的通知,可以做重发或故障处理			

如果需要单片机 2 发送的话,只需交换一下操作过程即可。

四、总 结

通过实践可知,以上方法是可行的。与其他方法相比具有以下优点:

⑴ 简单。占用单片机口线少(SCL、SDA、握手线 A、握手线 B)。

⑵ 通用。软件模拟 I^2C 主机方式,可以在任何种类的单片机之间通信。

⑶ 高效。由于采用数据缓冲,可以在不同时钟频率、不同速度的单片机之间通信;读写数据时,可以 I^2C 总线的最高速度进行,可以实现 1 次传送大量数据;在一个单片机向 FRAM 传送数据时,另一个单片机无须一一作出响应或等待,可以进行其他程序操作,提高软件工作效率。

⑷ 灵活。通信硬件接口对于各个单片机是对等的,通过软件配置,每个单片机既可以根据需要主动发送通信,也可以只响应其他单片机的呼叫。

⑸ 容易扩展。通过增加地址识别线,修改通信协议,即可做到多机通信。

以下是需要注意的地方:

⑴ 为了提高通信效率,握手线 B 最好使用中断端口,负脉冲宽度一定要满足速度较低单片机的中断信号要求。如果没有中断的话应增加 1 条口线,用改变端口状态的方法通知对方,等待对方查询,而不是负脉冲。

⑵ 向对方发送负脉冲时,应屏蔽自己的中断。

⑶ 由于参数与通信缓冲区同时设在同一片 FRAM 内,要避免对参数部分的误操作。一个较好的解决办法是把参数存放在地址的后半部分(A2 = 1),在进行通信操作时,把 FRAM 的 WP 引脚拉高(地址在后半部分的单元写保护),这样可以有效地防止通信时对参数区的误操作。

⑷ 由于 I^2C 总线在一个时间段内只有 1 个主机和 1 个从机,所以当 1 个单片机正在写通信数据时,另一个单片机是不能对 FRAM 进行操作的。如果需要实时、频繁地读取 FRAM 中参数的话,请预先将参数读入 RAM 单元使用或另外增加专门存放参数的芯片。

参考文献

1 Philips. THE I^2C-BUS SPECIFICATION

2 Ramtran. FM24C16 DataSheet

3 Joe Campbell. 串行通信 C 程序员指南. 北京:清华大学出版社,1995

选自《单片机与嵌入式系统应用》月刊,2002 年第 12 期

第三章

软件技术

3.1 面向应用的嵌入式操作系统

西安中兴通信研究所 (710065) 赵 宇 冯 锋

半导体工艺的发展和芯片设计水平的进步,使得单片机的性能大幅度地提高,因而应用范围不断扩大,可以同时监控的对象更多、系统规模更大并且更多地借助网络互联,软件当然随之愈来愈复杂。将实时操作系统 (RTOS) 引入单片机系统,使嵌入式软件开发从手工作业转变成工业化协作生产,是解决这一问题的途径。现在的 RTOS 功能更强,整体效率更高,适应的单片机更多,可移植性也更好。

近来嵌入式系统出现了一些新的特点,特别是嵌入式 DSP 技术和嵌入式 Internet 技术的出现,使得这些成熟的 RTOS 表现出了一定程度的不适应性。由于无处不在的数字化和无处不在的通信,嵌入式系统开发迫切要求面向应用的、可剪裁的 RTOS,我们称之为面向应用的嵌入式操作系统,本文以下简称 AEOS。

一、传统的 RTOS 和新的 AEOS

目前嵌入式领域采用的实时多任务操作系统 (RTOS) 是和应用的复杂化直接相关的。当单片机系统控制的外设和担负的任务不多时,一个主循环和几个顺序调用的子程序模块就可以满足要求,一般不需要 RTOS。如果有很多任务需要处理、要求对外部事件做出实时响应、任务之间存在信息传递甚至多个单片机协同工作,简单的程序设计方法就会面临许多问题,于是引入了 RTOS。

早期的 RTOS 只是一些小而简单的、带有一定专用性的实时监控程序,功能一般包括系统的初始化、实时时钟管理等,后来又逐步引入了任务调度和任务间协调功能。今天的 RTOS 已经变得非常强大,可以提供实时性很好的内核、多种任务通信机制、基于 TCP/IP 的网络组件、文件管理及 I/O 服务,提供了集编辑、编译、调试、仿真为一体的集成开发环境,支持用户使用 C/C++ 进行应用程序的开发。RTOS 是嵌入式软件编写从小生产进入工业化生产方式的必然产物,而 AEOS 是 RTOS 的进一步发展。

但是,AEOS 需要面对一些新的挑战。首先由于嵌入式 DSP 技术的发展,使得数字滤波、FFT、谱分析等方面的 DSP 算法大量进入嵌入式领域,而嵌入式系统为了实现更多的智能,例如智能逻辑消费类产品、生物信息识别终端、带有加解密算法的键盘、ADSL 接入、实时语音压缩和解压缩、虚拟现实显示等,也迫切要求 DSP 介入。DSP 处理器硬件经过单片化、EMC 改造、增加片上外设等途径变成了嵌入式 DSP 处理器。但是,嵌入式 DSP 仍然和 DSP 一样,具有明显不同于 RISC 和 CISC 的特点,其处理能力的提高不仅仅靠高速的时钟速率,而且需要并行的处理结构。DSP 处理器大多具有多个执行单元,每个执行单元都由算术逻辑运算单元、乘法器和累加器组成,为使这些单元能够有效地并行操作,一般需要直接使用汇编级编程工具而很少使用 C 语言。传统的 RTOS 在这样的系统上很难实现,要做到高效就更加困难。

另一个热门技术是嵌入式 Internet 系统。由于互联网和无线骨干网技术的飞速进展,使

得基于分组交换技术的通信性能、质量和可靠性得到稳步提高,互联网正在由服务型向应用型转变。过去孤立的 8/16 位单片机、通过专用网络连接的嵌入式设备,现在完全可以通过通用的 TCP/IP 网络共享信息,通过浏览器监测的控制。为此,许多单片机制造商推出了具有崭新体系结构的芯片,以保证 TCP/IP 协议栈可以在 8 位机上有效地实现,比如 Ubicom 公司的 IP2022 网络处理器,这些新体系结构带来的变化也是那些成熟的 RTOS 没有预料到的。

至少有以下三个问题使得传统的 RTOS 不能适应新的要求:

⑴ 任务调度数目固定。理论上,一个 RTOS 的实时性和多任务能力取决于它的任务调度机制。出于系统资源和核心效率的考虑,多数 RTOS 限定了系统中可被抢先调度的任务数目,用户不能增加,也不能减少。但实际应用的要求可能与这些假设存在很大差异。某些应用程序没有严格的实时要求,完全可以用一个轮循检测调度方法实现,却需要多个同时的 TCP 连接。另一个应用仅仅需要一二个 TCP 连接,但却有很多需要按优先级调度的任务以处理实时对象。RTOS 假定的任务数或者不能完全满足应用的要求,或者浪费了资源。

⑵ 内存开销太大。RTOS 需要的最小内存开销比起 Windows 一类的通用系统平台,可以说是微乎其微了,不过对于真正的无所不在的单片机来说,仍然属于庞然大物。多数系统需要数百 KB 的 RAM 和 ROM,典型的 RTOS 可能要求 4 MB 的 ROM 和 2 MB 的 RAM,这也是目前 RTOS 主要运行在 32 位系统上的原因。早期这个问题并不突出,因为有一段时间普遍认为那些曾在嵌入式系统和家电中担当过主力的 8/16 位芯片,在 Internet 时代应该下岗了,因为它们不能有效地处理 TCP/IP 协议。而高昂的开发和实现成本又限制了 32 位单片机的普及。随着新一代芯片纷纷推出,8/16 位单片机具有了新的计算能力,但其存储资源依然有限。在嵌入式系统中,单片机往往是成本的大头,而在一个单片机中,片上存储器的大小又是决定成本的关键。8/16 位系统要想在满足嵌入式系统功能要求的前提下保持最低的成本,相应的 RTOS 就必须大幅度地减少代码尺寸。

⑶ 中断禁止时间的影响。RTOS 通过核心态运行来实现其本身的功能,在核心态运行期间将屏蔽中断。中断只有在 RTOS 离开核心态时才能响应,这一过程所需的时间就是中断禁止时间。对于传统的嵌入式系统和单片机来说,只要保证各任务在规定的时间内完成,延时一段时间响应中断并不会产生什么问题。现在的情况有所不同,一个典型的例子就是 Ubicom 推出的 IP2022,它通过实现确定性的体系结构来保证可靠和有效地实现虚拟外设,这就是说,指令的执行时间是可以预测的,中断响应时间也是确定的(在内部时钟为 100 MHz 的 IP2022 中,片内中断响应时间是 30 ns)。如果 RTOS 的核心屏蔽了 IP2022 的中断,就会导致中断延时响应,从而引起输出抖动。在慢速的串行通信系统中,几个时钟周期的抖动完全可以忽略,不过在其他一些应用中,这种抖动可能是致命的。

总的来说,RTOS 强调的是系统的通用性、资源可配置性和整体效率,基本上没有考虑可伸缩性和可剪裁性。当然,也有一些 RTOS 允许我们进行一定程度的定制以满足前面提到的部分需求,不过新的 AEOS 已经提供了另外一种解决方案。

理想地,AEOS 应该是一种面向应用的、可剪裁的嵌入式实时操作系统,通过面向特定应用的简化型 API 来支持一种或一类嵌入式应用,而不再强调通用性;具有最小的内核处理集,系统开销小,运行效率高,可用于各种嵌入式系统和单片机;强调实时高性能而不再仅仅是整体效率;支持设备接入和无线接入扩展;具有 Internet 接入功能,提供 TCP/UDP/IP/PPP 协议等。

当然,强调 RTOS“小”,并不是简单地退回到 RTOS 的初级阶段,在强调实时运行内核“小”的同时,必须更加强调另一方面的“大”,这就是为用户提供更方便、更强大的基于 PC 主机的开发工具,包括集成开发环境、C 语言交叉编译器、调试工具和实时分析工具以及最重要的——操作系统配置工具。

从体系结构上看,AEOS 具有可伸缩、可剪裁、提供多层次功能、面向对象的系统体系结构。多层次构造有利于操作系统的功能伸缩 面向对象的系统功能划分有利于系统的剪裁和扩充新的功能。

可以把 AEOS 视为四层概念模型,即硬件接口层、内核层、系统层、系统服务接口层。硬件接口层主要提供与嵌入式硬件系统的接口。内核结构实现一个精小的内核,提供最基本的操作、如系统时钟、电源管理、程序装载与运行、进程线程调度、通信、内存管理等。系统层实现系统功能模块化和对象化,提供面向对象的系统资源管理功能,如 FFS Flash 内存管理、文件与目录管理、设备管理、TCP/IP 网络协议等。用户可以根据实际应用的需要选择特定的功能模块或部件。系统服务接口层提供基于系统功能的、面向应用的系统功能调用与服务接口(API)。

对 AEOS 发展前景的预测源于对嵌入式系统的市场分析。AEOS 不仅能在传统的工业自动化和商业管理领域,如智能工控设备、POS/ATM 机、IC 卡、智能交通设备等系统中有广泛的应用空间,而且在信息家电领域将更具应用潜力,比如机顶盒、网络电器等众多的消费类和医疗保健类设备。

二、AEOS 实例

面向应用的嵌入式操作系统已经引起相关行业的重视,出现了一些专门的研制开发公司。随着对应用高性能、低成本的不断追求,AEOS 将成为许多产品的核心。我们在实际项目开发中,使用了两种 AEOS 类产品,其鲜明的面向应用的特点给我们留下了深刻的印象。这两个 AEOS 一个针对 TI TMSC62xDSP,一个针对嵌入式 Internet IP2022。它们都是各自公司与器件配套开发的产品,具有强大的主机开发支持工具、灵活的可配置和可剪裁性。

1. Ubicom 的 IpModules

IpModules 是 Ubicom 公司为其新一代的 8 位互联网处理器 IP2022 开发的系统软件模块集。Ubicom 公司以前叫 Scenix,提供 SX 系列高速 8 位单片机和虚拟外设技术。新一代的 IP2022 具有 100 MHz 的工作频率,实现了确定性的指令执行和中断响应时间,利用预先构建的软件模块实现各种传统硬件外设功能,可以灵活、高效地实现 10BaseT 和 USB 接口,特别适合于 8 位单片机嵌入式 Internet 系统应用。我们使用 IP2022 单片机配合 IpModules 实现蓝牙协议的基带处理。

IpModules 包括核心操作系统服务(ipOS)、IP 协议栈(ipStack)、Ethernet 虚拟外设(ipEthernet)、I/O 服务(ipIO)、HTTP/1.1(超文本传输协议 1.1)和 Web 服务器(ipWeb)。

IpModules 使用图形化的工具对一个指定的应用项目进行配置。配置工具通过对预先构建软件模块的描述文件进行分析,改变模块配置点,满足应用程序的不同需要。例如,在一个 ipStack 中,用户可以选择支持哪些核心协议 在 ipOS 模块中,可以选择采用什么样的任务调度策略,是抢先的多任务调度还是简单的轮循检测。所有的模块使用 C 语言库提供,借助链接工具可以保证没有使用的程序和数据结构不会包含在目标代码中。

一旦确定了应用选择,配置工具就会生成若干文件,管理各种模块和用户应用程序的构

建。这样配置工具实际上就成了一个有效的高级编程环境,可以用来管理应用程序的全局结构。配置工具还有一个相关的特性是版本管理。在由配置工具管理的项目中,每个 IpModules 模块有一个独立的标识号,而每个模块的相近版本也有不同的版本号。这样,一个模块的多个版本可以交错地用于不同的项目之中。当一个模块升级时,用户可以选择使用这个模块的新版本还是保持老版本,于是处于软件维护阶段的模块可以和正在开发的新产品共存。

最终用户和第三方开发人员可以扩展 IpModules,当为特定的应用编写模块或者编写新的虚拟外设模块时,就可以借助已有的 IpModules,这样就提高了软件的可重用性。

IpModules 模块还可以实现方便的运行期间错误定位。所有的 IpModules 通过配置工具可以选择支持运行期间的"断言"。断言事件定义了一系列的预定动作,在项目开发过程中,最基本的预定动作就是将所有异常都记录到终端设备上。为此,需要实现一个高速串行接口连接主机和目标系统。当然,这个要求对于 IP2022 来说非常简单,只要利用虚拟外设技术实现一个 UART 就可以了,只不过占用了任意两个用户认为方便的 I/O 引脚(芯片有 50 多个通用 I/O 引脚),几百字节的代码空间,增加了微不足道的处理器负荷。

在一个存储资源相对受限的处理器上实现操作系统肯定要有一些权衡,许多 32 位系统的功能不可能在 8 位处理器上实现。比如 ipOS 的目标是最大程度地避免动态存储堆碎片,以保证可以返回尽量大的空闲存储器块给分配请求。这种策略比一些传统的策略要慢,但却可以为其他软件提供富有弹性的使用环境。

在片上资源有限的单片机上实现一个通用的 IP 协议栈不能采用常规的办法,如果按照 BSD Unix 实现进行移植,将占用大量的数据缓冲空间。初看起来好像必须简化 ipStack 模块,但简化又与通用性的需求相矛盾。解决这个问题的关键是充分理解 IP2022 的体系结构。在 100 MHz 的 IP2022 处理器中,程序代码可以直接在 FLASH 存储器中按 30 MIPS 的速率执行,也可以复制到程序 RAM 中按 100 MIPS 速率执行,这样高的速率就使得 C 语言函数调用可以全部利用堆栈传递参数,因为通过堆栈传递参数、寄存器传递参数以及引用嵌入在代码中的常数几种方法耗费的资源相当。所以,实现 TCP/IP 协议栈时,可以让数据同时通过协议栈的各个协议层,彻底避免各层之间的数据缓冲。如果对一个通用的 RTOS 在传统的处理器上使用同样的策略,得到相应的空间利用率自然没有问题,但吞吐量可能会变得非常低。这也从一个侧面反映了 AEOS 面向应用的特点。

2. TI 的 DSP/BIOS

DSP/BIOS 可运行于 TMS 320C5x 和 TMS 320C6xDSP 器件上。整个系统由三大部分组成,即实时库和应用程序接口、主机配置工具、插件(PlugIns)。实时库和应用程序接口提供应用程序直接调用的接口函数,RTOS 的基本功能由这一部分实现。主机配置工具用来静态定义应用系统组成,用户可以使用这个工具描述系统功能,实现操作系统的伸缩和剪裁。插件支持代码调试、主机数据传递和软件仪器等诸多功能,DSP 芯片上的 JTAG 边界扫描接口实现目标系统和 PC 主机的数据交换。

为了降低 CPU 的开销和减少系统存储器的占用,DSP/BIOS 同样采用了一些新的思想。首先,所有的 DSP/BIOS 对象包括任务、中断响应、事件记录、统计对象等等,都可以通过配置工具静态建立,这就省去了动态建立对象的代码空间和时间开销,当然,根据应用的需要,一些对象也可以动态建立。其次,应用程序接口 API 模块化,应用程序只需要调用相关部分的 API 函数,而未被引用的部分不会包含在目标代码中。第三,由于面向专门的器件和专用的目的,

大部分函数使用汇编语言实现,应用程序语句基本上是对这些函数的调用,所以应用程序可以用 C 语言编写。最后,目标 CPU 和主机插件之间的数据传输在后台空闲循环中完成,不影响应用程序的运行,重要的数据使用精简的结构在目标系统中缓冲,保证不会丢失实时事件的现场状态。

通过这些措施,DSP/BIOS 实现了高效、简洁的结构。实际上,DSP/BIOS 仅占用 6 KB,各种软件仪器对象仅仅耗用约 1% 的硬件资源。

DSP/BIOS 以线程为单位调度应用程序,应用可以使用多种线程类型。硬件中断线程(HWI)用于处理时间紧迫的外部、异步事件,它与硬件中断资源对应,可以剥夺任何其他线程的执行权。当然,用户编写的 HWI 线程应该尽量缩短执行时间,以保证中断得到及时和稳定的响应,不过这不是由 DSP/BIOS 强制的,而是完全由用户编码来自由控制的。在 HWI 中,可以通过 API 函数启动较低级别的其他线程。

软件中断线程 SWI 是一段相对独立的函数,优先级低于 HWI。它的主要作用是辅助 HWI 完成相应的任务,又避免 HWI 长期屏蔽中断,SWI 可以分成若干个相对优先级。每次退出 HWI 之后,将执行系统调度。如果有 SWI 的优先级高于当前运行的线程,则最高优先级的 SWI 调度运行,并且一直执行到结束,即使因缺少资源而等待,也不会引起重新调度。

一般 RTOS 中的任务对应 DSP/BIOS 中的 TSK 对象,其优先级低于 SWI。TSK 对象提供了完善的线程间通信和同步机制。每个任务具有不同的运行堆栈,当所需的条件和资源不能满足时,可以转入挂起状态,以调度其他任务执行。一个任务具有三种状态之一 运行,就绪和挂起,当然系统中同时只能有一个任务处于运行态。DSP/BIOS 提供灵活的线程间通信和同步机制,包括信号灯和邮箱。与外设之间的 I/O 数据传递则提供管道和 I/O 流两种机制,后者常用来为任务 TSK 提供硬件无关的输入输出设备驱动。

与一般的 RTOS 不同,DSP/BIOS 还有一类更低优先级的线程称为空闲线程 IDL,用于没有实时条件限制的事件处理。当系统空闲时,就会依次调用 IDL 中的各个函数。单独考察 IDL,实际上相当于轮循检测调度。

一个基于 DSP/BIOS 的应用程序就是由以上这些线程组成的,开发者只需要使用 C 语言或者汇编语言编写各个线程的代码,而线程的安排和各种对象的配置,都是通过图形化的主机配置工具来完成的。

DSP/BIOS 更具特色的是使用软件仪器来辅助系统的实时分析和现场测试。与传统的依靠断点的方式不同,DSP/BIOS 机制可以在系统实时运行的同时获得所需要数据,并在 PC 主机上对数据实时分析,然后以文本或图形方式显示。比如统计对象可以对任意变量进行实时采样,PC 主机每隔一定时间读取一次统计记录的原始数值,然后在 PC 机上进行统计计算。此外主机可以显示系统各个线程的执行时间、执行图和 CPU 负荷图,可以与 Excel、LabView 以及用户编写的具有 OLE 接口的程序交换数据。

三、结束语

随着嵌入式 DSP 和嵌入式 Internet 技术的广泛应用,对 RTOS 提出了更多更新的要求,与此对应的一些具有可伸缩性、可剪载性、强大主机开发支持环境以及面向应用特点的嵌入式实时操作系统产品已经出现。通过使用这些 AEOS,可以彻底改变嵌入式系统的开发面貌,使软件开发能够适应嵌入式 DSP 和嵌入式 Internet 的要求,使系统设计灵活,开发周期缩短,开发

成本降低,使最终的产品更加可靠和智能化。而且与传统的 RTOS 相比,价格也不贵。

参 考 文 献

1 UBICOM. IP2022 User s Manual [Z]. 2001,6

2 TI. TMS320C6000 DSP/BIOS User s Guide [Z]. 2000,5

3 Zilog. eZ80 Web server Programmer s Guide [Z]. 2001

选自《航空计算技术》双月刊,2002 年第 1 期

3.2 嵌入式实时操作系统及其应用

解放军理工大学 朱 巍

一、嵌入式操作系统简介

“嵌入式系统”是指将应用程序和操作系统与计算机硬件集成在一起的系统。简单地说，所谓嵌入式系统就是用户自己开发设计板子，板上有微处理器和各种芯片，其软件部分常常烧在 ROM 或 Flash 中，工作方式类似于 BIOS。这些专用的计算机系统是以嵌入式计算机为技术核心，围绕应用系统的功能、可靠性、成本、体积、功耗等严格要求来开发设计的。嵌入式系统一般由嵌入式微处理器、外围硬件设备、嵌入式操作系统以及特定的应用程序等 4 个部分组成，用于实现对其他设备的控制、监视或管理等功能。很多简单的嵌入式系统，如大多数单片机系统，是不需要操作系统的。但一些功能复杂的嵌入式系统，如机顶盒，有自己的操作系统。“嵌入式操作系统”即指嵌入式系统中采用的操作系统，其特点是微内核、可裁剪、低资源占用和低功耗。按照“嵌入”方式的不同，嵌入式系统可分为以下几种。

1. 整机式嵌入

一个带有专用接口的计算机系统嵌入到一个控制系统中，成为控制系统的核心部分。一般这种计算机系统功能完整而强大，完成系统中的核心的关键工作，也具有较完善的人机界面和外部设备。

2. 部件式嵌入

以部件式嵌入到一个控制设备中，完成某一处理功能，与设备的其他硬件偶合紧密、功能更专一。如雷达的数字处理部件，一般选用专用 CPU 或 DSP。

3. 芯片式嵌入

一个芯片是一个完整的专用计算机，具有完整的输入输出接口，完成专一的功能。如显示处理器、微波炉控制器等。一般为专门设计的芯片。随着微电子技术的发展，芯片式嵌入应用将越来越广泛。

二、实时操作系统简介

“实时操作系统”是相对“分时操作系统”而言的，我们日常接触的通用操作系统（如 Windows、Unix、Linux 等）都是分时操作系统。实时操作系统能及时（或即时）响应外部事件的请求，在规定的时间内完成对该事件的处理，并控制所有实时任务协调一致地运行。与分时系统相比，具有多路性、独立性、及时性、交互性、可靠性的特点。

分时操作系统的基本设计原则是：尽量缩短系统的平均响应时间并提高系统的吞吐率，在单位时间内为尽可能多的用户请求提供服务。由此可以看出，分时操作系统注重平均表现性能，不注重个体表现性能。如对于整个系统来说，注重所有任务的平均响应时间而不关心单个任务的响应时间；对于某个单个任务来说，注重每次执行的平均响应时间而不关心某次特定执

行的响应时间。通用操作系统中采用的很多策略和技巧都体现出了这种设计原则,如虚存管理机制中由于采用了 LRU 等页替换算法,使得大部分的访存需求能够快速地通过物理内存完成,只有很小一部分的访存需求需要通过调页完成,但从总体上来看,平均访存时间与不采用虚存技术相比没有很大的提高,同时又获得了虚存空间可以远大于物理内存容量等好处,因此虚存技术在通用操作系统中得到了十分广泛的应用。类似的例子还有很多,如 Unix 文件系统中文件存放位置的间接索引查询机制等,甚至硬件设计中的 Cache 技术以及 CPU 的动态分支预测技术等也都体现出了这种设计原则。

而对于实时操作系统,除了要满足应用的功能需求以外,更重要的是还要满足应用提出的实时性要求,而组成一个应用的众多实时任务对于实时性的要求是各不相同的。此外实时任务之间可能还会有一些复杂的关联和同步关系,如执行顺序限制、共享资源的互斥访问要求等,这就为系统实时性的保证带来了很大的困难。因此,实时操作系统所遵循的最重要的设计原则是 采用各种算法和策略,始终保证系统行为的可预测性 (predictability)。可预测性是指在系统运行的任何时刻,在任何情况下,实时操作系统的资源调配策略都能为争夺资源 (包括 CPU、内存、网络带宽等) 的多个实时任务合理地分配资源,使每个实时任务的实时性要求都能得到满足。与通用操作系统不同,实时操作系统注重的不是系统的平均表现,而是要求每个实时任务在最坏情况下都要满足其实时性要求。也就是说,实时操作系统注重的是个体表现,更准确地讲是个体最坏情况的表现。举例来说,如果实时操作系统采用标准的虚存技术,则一个实时任务执行的最坏情况是每次访存都需要调页,如此累计起来的该任务在最坏情况下的运行时间是不可预测的,因此该任务的实时性无法得到保证。

由于实时操作系统与通用操作系统的基本设计原则差别很大,因此在很多资源调度策略的选择上以及操作系统实现的方法上两者都具有较大的差异。

一个好的实时操作系统需要具备以下功能 (必须但非充分):

- 多任务和可抢占的;
- 任务具有优先级;
- 操作系统具备支持可预测的任务同步机制;
- 支持多任务间的通信;
- 操作系统具备消除优先级转置的机制;
- 存储器优化管理 (含 ROM 的管理);
- 操作系统的 (中断延迟、任务切换、驱动程序延迟等) 行为是可知的和可预测的。这是指在全负载的情形下,最坏反应时间可知;
- 实时时钟服务;
- 中断管理服务。

实时系统最关键的部分是实时多任务内核。它的基本功能包括多任务管理、定时器管理、存储器管理、资源管理、事件管理、系统管理、消息管理、队列管理、信号量管理等。这些管理功能是通过内核服务函数形式交给用户调用的,也就是实时操作系统的 API。

三、嵌入式实时操作系统

嵌入式操作系统并不一定是实时的。如 Windows CE,现在比较热的嵌入式 Linux 的大多数版本也都不是实时的。对于 Windows CE 和嵌入式 Linux 也有所谓“软实时”操作系统的说

法,但严格来讲它们都不能算是真正的实时操作系统。这主要是因为 Windows 和 Linux 最初都是按分时系统设计的,即使作成嵌入式的应用也无法实现真正的实时性。实时操作系统也并不一定要是嵌入式的,在微机上一样可以装实时操作系统,而实际上大多数实时操作系统也都支持 PC 机使用的 X86 芯片。非实时的嵌入式操作系统的应用领域如机顶盒、PDA、掌上电脑等,这类应用对实时性并没有特殊的要求。嵌入式实时操作系统是将嵌入式和实时性相结合的产物。由于其优良的特性,广泛应用于制造工业、通信、航空航天、军事武器装备等领域。它的主要特点如下:

· 响应时间快,并且有确定的硬实时性要求;

· 具有异步处理并发事件的能力;

· 具有快速启动、出错处理和自动复位功能;

· 嵌入式系统的应用软件与操作系统之间的界限模糊,往往是一体化设计的程序;

· 软件开发困难,要使用交叉的开发环境(即开发环境与运行环境不同,开发平台叫宿主系统,而嵌入式系统的运行系统叫目标系统)。

嵌入式系统的一般开发流程如图 3.2-1 所示。

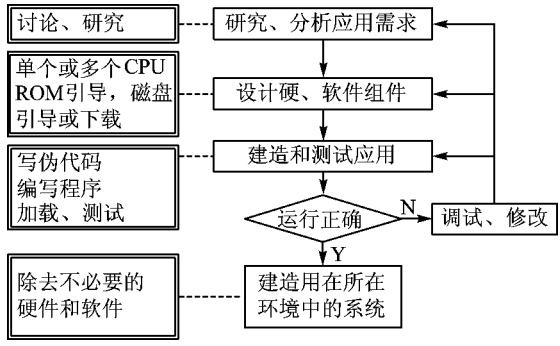


图 3.2-1 嵌入式系统的开发流程

与个人电脑的操作系统不同,嵌入式操作系统并没有被某个厂商所垄断。原因是相对 Unix 和 Windows 这样的操作系统而言,嵌入式实时操作系统的规模要小得多,系统更简单,开发也更为容易。目前嵌入式实时操作系统数量众多,很多大公司都有自己的实时操作系统。一些常见的嵌入式实时操作系统如表 3.2-1 所列。

表 3.2-1

操作系统名称	生产商
DeltaOS	北京科银京成技术有限公司
QNX	QNX SOFTWARE SYSTEMS LTD
VxWorks	Wind River Systems, Inc.
pSOSystem	Wind River Systems, Inc.
Hopen	北京凯思昊鹏软件工程技术有限公司
CMX	CMX Systems,Inc.
LynxOS	Lynx Real-Time Systems

续表 3.2－1

操作系统名称	生产商
Nucleus PLUS	Accelerated Technology Inc.
OS-9	Microware Systems Corp.
OSE	Enca OSE Systems
RTXC	Embedded Power Corporation
SuperTask&TRONTASK	US Software
VRTX	Microtec Research
I-TRON	(日本) TRON 协会-ITRON 技术委员会

还有一些 RTOS 是公开源代码和免费的，如 μC/OS。在众多的 RTOS 中，使用最广、最有名的当属美国 WindRiver 公司的 VxWorks。VxWorks 是美国市场销售量第一的实时操作系统。在美国的火星探测器、爱国者导弹中均使用了 VxWorks。目前在我国市场上该系统也占有较大分额。下面以 VxWorks 为例，简要介绍一下嵌入式实时操作系统的一般构成、开发环境和应用领域。

1. VxWorks 的一般构成

VxWorks 支持 32 位的 CPU，包括 Intel 公司的 x86、Motorola 公司的 68k 和 PowerPC、MIPS、ARM、Intel 公司的 i960、Hitachi 公司的 SH。和通用操作系统不同，嵌入式实时操作系统一般没有自开发能力，也就是说其开发环境是基于 Unix 或 Windows 这样的通用操作系统的。VxWorks 带有功能强大的集成开发环境 Torando，支持的平台包括 Windows 9x/NT/2000，Sun Solaris，SunOS，HP－UX。

VxWorks 的主要特性有：

· 可裁减微内核结构；

· 高效的任务管理——多任务，具有 256 个优先级，支持优先级排队和循环调度；

· 灵活的任务间通信方式，包括：信号量（二进制、计数、有优先级继承特性的互斥信号量）、消息队列、套接字（socket）、共享内存和信号（signals）；

· 微秒级的中断处理；

· 支持 POSIX 1003.1b 实时扩展标准；

· 支持多种物理介质及标准的、完整的 TCP/IP 网络协议栈（与 BSD 兼容）；

· 灵活的引导方式，支持从 ROM、Flash、本地盘（软盘或硬盘）或网络引导；

· 支持多处理器并行处理；

· 快速灵活的 I/O 系统；

· 支持 MS－DOS 和 RT－11 文件系统；

· 支持本地盘，Flash，CD－ROM 的使用；

· 完全符合 ANSI C 标准；

· 1 100 多个系统调用；

· 支持 Java 和 CORBA。

2. VxWorks 的开发环境

VxWorks 的开发环境 Torando 包括强大的开发和调试工具，尤其适用于面对大量问题的

嵌入式开发人员。这些工具包括 C 和 C + + 远程源级调试器、目标和工具管理、系统目标跟踪、内存使用分析和自动配置。另外,所有工具能很方便地同时运行,很容易增加和交互式开发。

WindRiver 公司还提供了一系列的可选开发组件,包括:

(1) BSP 开发包 (BSP Developers Kit)。BSP 开发包帮助开发人员把 VxWorks 移植到客户化硬件平台上。BSP 开发包的选项包括:测试工具、硬件设备的驱动程序库、BSP 模板。用户可以根据需要选择不同的选项。WindRiver 还提供 BSP 测试验证等咨询服务。

(2) VxVMI 是 VxWorks 的虚拟内存接口。在调试阶段和软件运行时都能提供强大的内存管理功能。它包括代码段和数据段保护功能,并包含对不同 CPU 结构的标准编程接口。

(3) VxMP 是 VxWorks 多处理器支持扩展包。它允许将任务分布在多个 CPU 上执行以提高效率。它透明的、高性能的设计使得在不同 CPU 上运行的任务可以通过现有的通信机制,如信号量、消息队列等进行同步和数据交换。

(4) Tornado 移植包。易于使用的 Tornado 移植包,允许把基于 VMEexec、pSOS 及其他嵌入式操作系统的应用程序移植到 VxWorks 上。

3. VxWorks 的应用

VxWorks 的应用领域包括数据网络、远程通信、医疗器械、消费电子、交通运输、计算机外围设备、数字图像、工业测量和控制、航空、多媒体等等。下面主要介绍一下它在网络和通信方面的应用。

VxWorks 是最早在其内核中加入 TCP/IP 网络协议的嵌入式实时操作系统,其内部包括 1 个 BSD4.4 兼容的实时 TCP/IP 协议栈,网络模块结构如图 3.2-2 所示。

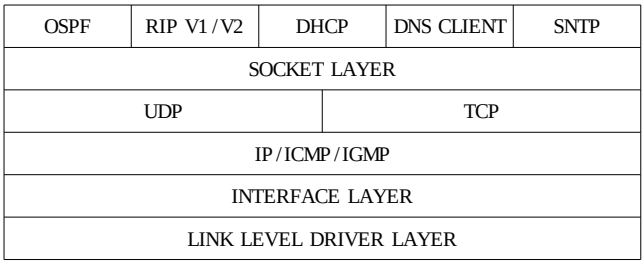


图 3.2-2 VxWorks 的网络模块

VxWorks 适合于高性能的网络交换设备到低价的网络接入设备,如 10M/100M 以太网交换机、广域网接入设备、ATM 交换机等。VxWorks 软件包是可调整的,使得开发者可以将其应用到从 IP 路由设备到完全 TCP/IP 的基于 SNMP 管理的应用系统中。VxWorks 协议栈提供本地交换机或远程接入路由器所需的最新路由技术,可被用于 gigabit 以太交换机或 DSL 接入复用器等。VxWorks 协议栈还支持 IP 多址广播、CIDR、DHCP、DNS、SNTP 等网络协议。由于 VxWorks 具有完备的协议栈、方便的编程接口和灵活的可裁减特性,在其基础上可以方便地开发自己的各层协议。在 VxWorks 基础上开发的第三方产品包括:OSI、SS7、ATM、Frame Relay、CORBA、ISDN、X.25、CMIP/GDMO、分布式网络管理、无线接入等。

四、结束语

美国著名未来学家尼葛洛庞帝 1999 年 1 月访华时预言,4 ~ 5 年后嵌入式智能(电脑)工具将是 PC 和因特网之后最伟大的发明。今天,嵌入式系统带来的工业年产值已超过了 1 万亿美元。在计算机网络和通信领域,嵌入式实时操作系统以其优良的性能必将发挥越来越大的作用。

参 考 文 献

1 汤子瀛,哲凤屏,汤小丹. 计算机操作系统. 西安:西安电子科技大学出版社,1999

2 孔祥营,柏桂枝. 嵌入式实时操作系统 VxWorks. 北京:中国电力出版社,2001

3 Wind River System Inc. Tornado User's Guide (Windows Version) 1.0. 1996 - 01

选自《单片机与嵌入式系统应用》月刊,2002 年第 8 期

3.3 Windows CE 在嵌入式工业控制系统中的应用思考

华北电力大学 吕跃刚 张新房 徐大平 柳亦兵

一、嵌入式系统

嵌入式系统(Embedded System)是指有特定功能或用途的计算机硬、软件的集合体,分为嵌入式软件系统和嵌入式硬件系统。在智能控制设备、便携式智能仪器等应用场合,出于对产品体积、成本等诸因素的考虑,往往要求将智能控制部分安装于设备内部,且占用的空间尽可能小,在这种情况下,处理器没有一般意义的硬盘,只有有限容量的内存及常用的 Flash 电子盘,这样的系统称为嵌入式系统。嵌入式系统的操作系统和功能软件集成于计算机硬件系统之中,也就是软件与硬件的一体化。嵌入式系统目的性或针对性很强,具有软件代码小、高度自动化、响应速度快等特点,这也是与通用计算机系统的最主要区别。嵌入式技术与实时性有着必然的联系。

二、从单片机的应用发展到嵌入式操作系统

嵌入式系统开始于 20 世纪 80 年代单片机的使用。单片机技术已经渗透到各个领域,且与人们的日常生活密不可分,给人们生活和工业生产带来极大方便。单片机的功能强大,从信号采集、处理到传输都能由单片机来完成。但是,随着网络时代的来临,许多电子设备需要联网和更智能化、更强的计算能力,比如音频、视频的数据采集、处理和传输 ;丰富的图形界面等。

单片机越来越不能满足应用对象的需求,开发工作也变得越来越复杂、庞大。随着微电子技术的进步,芯片的制造成本大大降低,而功能却大大增强,16 位和 32 位的嵌入式微处理器逐渐成为嵌入式系统设计的主流。但是,只有嵌入式微处理器是不够的,OEM(原始设备制造商)还需要有一个运行于嵌入式微处理器上的操作系统。嵌入式操作系统要有良好的可移植性,能够用在根据应用要求选择的微处理器中 软件开发工作变得规范,容易测试,可实现模块化编程,同时由多个人共同完成一个任务 解决已往开发产品存在的诸多不安全隐患。很多软件厂商迎合嵌入式系统发展的需要,推出了多种不同特点的嵌入式操作系统。例如 Microsoft 公司的 Windows CE、3COM 公司的 Palm OS、Symbian 公司的 EPOC、中科院凯思集团的 HOpen 以及 Linux 等。

三、Windows CE 3.0 实时操作系统及其性能分析

1. Windows CE

Windows CE 操作系统是微软为实现“信息随手可得”的设想而努力开发的成果。通过 Windows CE,微软提供了标准的开放式平台,极大地减少了硬件制造商(IHV)、软件开发商(SHV)以及最终将采纳新一代非 PC 技术解决方案的客户多方之间的矛盾。Windows CE 是一个功能强大的开放的 32 位实时嵌入式操作系统,适用于快速构建新一代内存少、体积小的智

能设备。例如工业控制器、手持式设备、智能电话、机顶盒和零售点设备等。目前的掌上电脑(PDA)、全球定位系统(GPS)、地理信息系统(GIS)、车载 PC(Auto PC),有很多采用 Windows CE 操作系统。

2. Windows CE 3.0 性能特点

Windows CE 是一个抢先式多任务并具有强大通信能力的嵌入式操作系统。它是一个全新的、可移植的、实时的、模块化的操作系统,具有流行的微软程序开发界面,提供许多快速开发嵌入式系统的工具。

(1) 新内核

Windows CE 看上去和 Windows 9X/NT 很像,但它不是这些操作系统的简化版,也不是从这些系统移植过来的。Windows CE 具有全新的内核和任务调度、内存管理策略。

(2) 可移植性

由于 Windows CE 操作系统几乎完全是用 C 语言编写的,所以可移植到众多的 32 位微处理器上;支持各种处理器家族,包括 x86、PowerPC、ARM、MIPS 和 SH 等系列。微软为每个支持的处理器家族提供完整的系统库。Windows CE 可以通过 OEM 适配层 OAL(OEM Adaptation Layer)适配到任何硬件平台。OAL 是驻留在 CE 内核和硬件之间的代码层。原始设备制造商使用这些代码把 CE 适配到自己的硬件上。OAL 链接 CE 的内核和定制的硬件。

(3) 实时性

Windows CE 2.1 及其以前的版本实时性能不强,但 Windows CE 3.0 及以后的版本实时性能得到明显改善。Windows CE 3.0 的实时性能主要通过以下技术实现:支持嵌套中断,高优先级的中断并不需要等待低优先级的中断服务例程(ISR)完成;256 个线程优先级,可以灵活调度嵌入式系统的任务;通过固定高优先级中断服务线程(IST)的最大调度延迟改善线程响应时间;使用 API 函数 CeSetThreadQuantum 和 CeGetThreadQuantum 修改操作系统中线程的线程量;中断服务子程序的响应时间非常短;支持信号量。在基于 Windows CE 的参考平台上,使用 Hitachi SH3 微处理器,系统可以在 2 ~ 5 μs 内启动一个中断服务例程(ISR),在90 ~ 170 μs 内启动相应的中断服务线程。如果考虑其他因素,如 CPU 类型、时钟频率、总线速度等的影响,许多实际的基于 Windows CE 平台的响应时间更短。

(4) 模块化

由于存储器资源在移动和嵌入式设备中非常有限,Windows CE 设计成一个模块化操作系统,设计者只需选择那些需要的模块以满足指定平台的存储器要求。Windows CE 的结构如图 3.3 - 1 所示,主要包括 4 个模块:内核(Kernel)、图形窗口事件子系统(GWES)、文件系统(Filesys)和通信模块(Communications)。Kernel 负责中断处理、进程和线程管理、虚拟内存管理和其他相关任务;GWES(Graphics Windowing and Events Subsystem)相当于桌面 Windows 的图形设备接口 GDI 和用户库;Filesys 用于永久存储,包括文件系统、注册表和数据库;Communications 模块负责与桌面 PC、其他 CE 设备和因特网的互联。每个模块又分成许多小组件。裁减 Windows CE 时,可以只选择那些需要的组件。

(5) Win32 兼容性

Windows CE 采用与 Windows 95/NT 相同的编程模型,它的 API 是 Win32 API 的一个子集,大约有 600 个 API 函数,可以实现所有的嵌入式应用。CE 只支持 UNICODE 码,CE API 删除了 Win32 API 中包含 ANSI 字符串参数的函数。CE 还支持当前流行的软件技术和运行库,

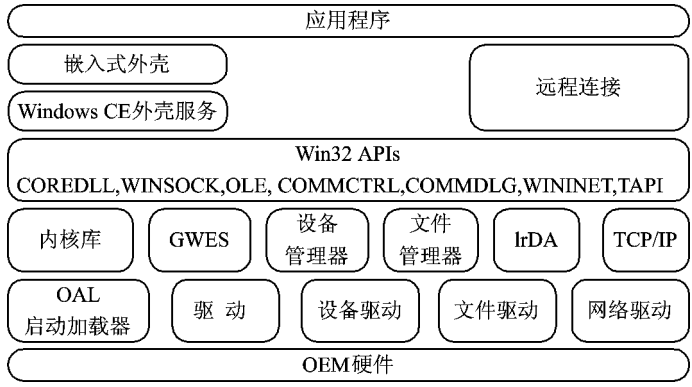


图 3.3－1 Windows CE 的基本结构

如 MFC (Microsoft Foundation Class)、ATL (Active Template Library)、EVC (Embedded Visual C++)、EVB (Embedded Visual Basic)。Win32 的兼容性可以容易地把现成的 Windows 应用程序移植到 Windows CE 中。目前有许多开发人员精通 Windows 编程技术,他们只需学习很少的知识就可以开发 Windows CE 应用程序。

PC 机技术的发展必然出现两极分化:一方面 PC 机功能将进一步加强,达到以前工作站和小型机水平;另一方面,面向普通消费者和特定用途的智能化电子设备将会大量涌现。后者将会普遍采用类似 Windows CE 的嵌入式操作系统。

四、Windows CE 在嵌入式控制系统中的应用分析

嵌入式操作系统是一种应用广泛的系统软件,工业控制是它的传统应用领域,在这一领域里已有一些比较成功的嵌入式操作系统。但是,随着应用对象的扩大和技术的进步,实际应用对工业控制系统的功能和性能提出了许多新的要求。例如,适应恶劣的工作环境,熟悉和友好的用户界面,统一的编程界面,强大的通信功能和多媒体功能等,这些嵌入式操作系统很难满足工业应用的新需要。由于 Windows CE 2.1 及以前版本的实时性较差,在工业控制领域应用较少,主要应用在移动式 (或便携式) 产品和信息家电领域。Windows CE 3.0 的出现极大地改善了它的实时性能,为 Windows CE 进入工业控制领域奠定了基础。

虽然 Windows CE 3.0 作为嵌入式系统平台在工业控制领域还未被广泛采用,但前景非常广阔。许多著名的工业控制器生产商已经开发出基于 Windows CE 3.0 的工业控制产品,如西门子 AG 公司的多功能操作面板 MP 系列,Cybectec 公司的变电站现代化平台 SMP (Substation Modernization Platform) 等。工业控制操作系统需要严格的实时处理功能,高可靠性,良好的开放性,对人机界面、开发环境、可操作性、成本等也有特别的要求。

1. 实时性

实时性是指能够在限定时间内执行完规定的功能,并对外部的异步事件作出反应的能力。实时性的强弱以完成规定功能和作出响应时间的长短来衡量。提高硬件的处理能力可以在一定程度上提高计算机控制系统的实时性,但是当硬件确定以后,控制系统的实时性能主要由操作系统来决定。无论从汽车制造到工业自动化,还是从电子通信到交通运输,Windows CE 3.0 均可为具备确定性响应能力的应用程序提供内建实时支持。

2. 可靠性

工业控制系统对可靠性要求很高,计算机控制系统发生故障或死机对于企业安全高效生产带来不利的影响。可靠性主要包含两个方面的含义:一是控制计算机本身要连续稳定运行,二是系统检查出故障后要有保持安全状态的能力。虽然软硬件抗干扰技术、热冗余技术可以在一定程度上提高工业控制系统的可靠性,但是操作系统的可靠性仍然影响着工业控制系统的运行。

稳定性方面,在实时控制操作系统中,一般要提供源代码或者提供许可证,由控制器生产商来保证系统的稳定性。控制器生产商根据应用需要定制 Windows CE 操作系统,经过一段时间的完善和测试以后投入使用。测试的方式和时间由生产商确定。通过这种方式定制的操作系统一般情况下可以稳定运行,但微软公司既没有保证 Windows CE 连续运行的时间,也不公开源代码。从这个意义上说,Windows CE 的稳定性受到质疑。此外,在工控设备中,因为产品缺陷而造成事故,厂商要承担赔偿责任。对操作系统也一样。因此,微软对这一问题的态度,也是 Windows CE 能否很好地应用到工业控制领域的因素之一。

从故障角度来看,实时操作系统在应用中是以内核模式工作的,应用的故障会立刻造成系统崩溃。Windows CE 内核具有内存管理功能,可以检查出应用造成的系统异常,抑制由于应用不正常直接破坏系统的危险性。所以 Windows CE 比一般的实时系统健壮。

3. 人机界面

不同对象对工业控制系统的人机界面 HMI (Human Machine Interface) 要求差别很大。在一般的实时嵌入式操作系统中,图形功能弱,虽然也有提供图形库的,但没有更强的功能。Window CE 不仅支持图形和窗口,具有多媒体功能,而且还可以利用丰富灵活的控件库在 Windows CE 环境下为嵌入式应用建立各种图形用户界面。Windows CE 支持 256 色,显示分辨率可以设定,支持触摸屏。因此,Windows CE 完全可以满足工业控制系统对人机界面的要求。

4. 开放性

Windows CE 具有良好的通信能力,广泛支持各种通信硬件、局域网连接以及拨号连接,并提供与 PC、内部网以及 Internet 的连接,包括用于应用级数据传输的设备至设备间的互连。Windows CE 具有良好的可扩展性,用户可根据实际需要定制合适的硬件,开发自己的模块和组件,集成到运行 Windows CE 的设备上。

5. 开发成本和开发环境

工业控制设备的生产批量小,开发环境所占比重大,所以易用、廉价的开发环境对控制设备生产商十分关键。Windows CE 的开发成本低,生成和调试工具方便易用。Windows CE Platform Builder 3.0 提供了迅速创建 Windows CE 嵌入式系统需要的全部软件工具。Platform Builder 主要包括 Windows CE Add－on Pack (插件包)、各种调试工具及 Embedded Visual Tools (由面向嵌入式系统开发而进行优化的 Embedded Visual Basic 和 Embedded Visual C++ 组成)。

五、结 论

到目前为止,工业控制系统中的自动化设备仍然受专用硬件或工业化 PC 平台的限制。专用硬件通常十分耐用,并能抗恶劣环境,但是只能用于单一的、特定的用途。虽然工业化 PC

比专用硬件更具有灵活性,但是由于振动、灰尘、潮湿、高温以及其他环境问题的影响,工业化 PC 平台经常会出现故障和数据丢失。Windows CE 操作系统是一个适合下一代互连工业自动化设备的理想小体积嵌入平台。由于采用 MSMQ (Microsoft Message Queuing) 这样的先进应用服务,使 Windows CE 实现与生产现场 IT 设施的全面集成成为可能。它还具有很强的实时性能,支持确定性的响应时间控制。Windows CE 能从闪存启动,从而避免了暴露在灰尘、高温和震动环境下,使它可以适应恶劣的生产环境。基于 Windows CE 的嵌入式控制系统提供统一的、可伸缩的解决方案,将专用硬件的耐用性与 PC 的灵活性结合在一起。因此 Windows CE 在工业控制领域有着很好的应用前景。

参 考 文 献

1 卢海峰. Windows CE. 电子科技. 2001 (11):32 ~ 34

2 微软公司. Microsoft Windows CE Device Driver Kit (设备驱动程序开发指南). 希望图书创作室译. 北京:北京希望电子出版社,1999

3 Hipson Peter D. Windows NT4 注册表专家指南. 朱友芹、王欣等译. 北京:电子工业出版社,1999

4 微软公司. Microsoft Windows CE Platform Builder 3.0 Library (Platform Builder 3.0 电子帮助文档)

选自《单片机与嵌入式系统应用》月刊,2002 年第 9 期

3.4 简易非抢先式实时多任务操作系统的设计与应用

合肥工业大学 彭良清

一、嵌入式实时操作系统的特点和产品

1. 嵌入式操作系统的特点

按 Andrew S. Tanenbaum[2] 的说法,使用操作系统平台来进行软件设计的目的之一是:操作系统隐藏了计算机硬件的细节,提供了比物理计算机更为容易使用的虚拟计算机,使应用软件设计更加容易,硬件适应性也更好。这一点嵌入式操作系统也不例外。

嵌入式操作系统和一般通用计算平台所使用的操作系统相比,前者主要的特点在于其代码效率高,并且具有实时性、高可靠性和可剪裁性,同时也更加简单,不需要文件操作等功能。嵌入式操作系统和在其之上的应用软件一般是存放在 ROM 或 Flash 存储器中的,不同于通用操作系统在磁盘上存放,因此,软件规模的大小十分重要。与 UNIX、Windows 等通用 OS 相比,嵌入式 OS 的规模要小得多。

嵌入式操作系统的另一个特点是专用性。嵌入式系统的处理器和应用要求各不相同,嵌入式软件(包括操作系统)往往也是专用和各异的,无须照顾各种情况,因此,嵌入式 OS 的设计比通用 OS 要简单得多,这使得我们能根据某一类应用的特点来设计自己的简易操作系统。

从嵌入式系统的硬件平台来看,一类是以 Intel 公司和 AMD 的 X86 系列处理器、MOTOROLA 公司的 68K 系列处理器为代表的系统。这类微处理器是 32 位的,可以支持较大容量的存储器,因此,可以运行功能较强和规模稍大的软件。另一类硬件平台是各种 8 位或 16 位字长的微控制器(单片机)系统。其特点是存储器极其有限,必须运行规模很小的软件,同时对软件的复杂性要求也低。在这种系统中的软件设计一般不使用 OS 平台,对硬件的依赖性也很强,在不同 MCU 之间的移植困难;另外一方面也使设计过程相对复杂,可维护性差。如果能在 OS 平台上开发应用软件,则可屏蔽硬件结构。因此,选择和设计一种适用于 MCU 平台的 OS 内核是必要的。

2. 商业 OS 软件介绍

表 3.4-1 中给出了几种常见的商业嵌入式实时多任务操作系统内核性能比较(不包括操作系统的 I/O 部分)。

表 3.4-1

名称	运行平台	规模/KB	价格/美元	源码提供	编程语言	公司网址
Vxwork	x86/68K	16/488	16 500	NO	C/C++	www. wrs. com
PSOS	x86/68K	40	17 000	NO	C/C++	www. isi. com
QNX	x86	32/64	收费	部分提供	C/C++	www. qnx. com
Lynx	x86/68K	240	10 000	YES	C/C++	www. Lynx. COM
VRTX	x86/68K	16/26	2 000	YES	C/C++	www. mentorg. com
RTXC	x86/68K/8051	不详	不详	NO	C	www. lineo. com
RTX51	8051	8+2	不详	NO	C51	www. keil. com

* 本表资料来源：http://www. embedinfo. com/rtos/rtosmain. asp

商用实时多任务操作系统一般均应用在中型系统中，即运行在前面所说的第一类硬件平台上，主要特点是存储器较大；另外，价格和技术上的花费也不是一般用户能够承受的。在微控制器（MCU）上可以使用和运行的商业 OS 内核很少，有 Keil 公司提供的运行在 8051 平台上的 RTX51，并不提供源代码。

3. MCU 应用系统中应用软件和 OS 平台的特点

从操作系统本身所具有的特点来说，可分为非抢先式和抢先式 OS 二种。后者的实时性好，但实现复杂，对硬件系统（存储器）的要求也较高，适合于系统相对复杂，对价格相对不是很敏感的系统。

对于一般的 MCU 应用系统来说，由于应用进程数量和功能均是可以预知的（比如对于一个数据采集仪表，应用进程可分为数据显示、数据采集、通信、数据处理等），数量极其有限（一般不超过 10 个），进程执行的功能也相对简单。因此，完全可以使用简单的非抢先式 OS 内核。只要保证在实际过程中，应用进程中没有死循环等待处理代码就可以保证 OS 对进程的正常调度了。

由于使用商业 OS 代价相对较大，并且从技术上同样要花时间去掌握，因此，我们的想法是设计一种简易的非抢先式实时操作系统，以满足一定程度上的通用性。以下讨论这种 OS 平台的设计和基于这种 OS 平台的应用软件设计方法。

二、非抢先式实时多任务操作系统的设计

1. 基本架构

操作系统的最基本功能有：

① 进程调度；

② 进程间通信；

③ 为应用进程提供定时功能（定时器和定时任务）；

④ 内存管理；

⑤ I/O 功能。

操作系统是为了对用户程序提供支持和管理。所谓进程实际上可以看作是一个 C 函数模块。以下分别讨论一个简易非抢先式多任务操作系统基本功能的实现方法，并说明利用中 OS 平台来设计应用程序的步骤。

2. 进程调度和进程间通信

对于应用进程函数，操作系统应该根据什么、在何时对其中的进程函数进行激活运行呢？

一般的抢先式操作系统是根据时间片来轮回调用各进程函数的。这种方式需要大量的堆栈内存,因为在进程任务切换过程中,需要保存其当时运行的状态;而切换往往是操作系统在进程未知的情况下,通过硬件中断服务例程实现的,应用进程并不知道自己何时被终止执行,所以,各个应用进程必须有各自的专用数据区,以免互相影响。

在本设计中,为简单起见,OS 内核采取非抢先式机制。在这种机制下,应用进程如果不主动退出执行,OS 内核将不能获得控制权,并将控制权交给其他进程,因此,也就不需要保存进程当前运行状态的堆栈存储器。非抢先式 OS 内核是在何种条件下调用一个应用进程,应用进程又是在何种条件下暂停本次执行的呢?我们先说明前者。

非抢先式内核是根据外部事件来对应用进程进行调度和激活的。这里的外部事件指以下方面:

① 其他进程发给本进程的数据;

② 各种 I/O 口产生状态信息数据;

③ 本进程的定时任务所产生的数据。

以上三种数据均可以存放在一个消息队列中,然后由 OS 内核来查阅该消息队列,对消息的去向进行分析后调用相应的进程;而没有外部事件时,应用进程将得不到执行。因此,进程调度和进程间通信实际上是连为一体的。根据这种思想,实现的 C 代码如下:

```
main()
{
int Recv_PID;                    /* 表示接收消息进程号的变量 */
MessageQueueTerm Message;
                                 /* 保存当前消息队列首部的消息位置变量 */
  while (1) {                    /* 嵌入式软件总是一个死循环 */
  Recv_PID = ReadMessageQueue (&Message);
                                 /* 扫描消息队列,并得到其中的进程号和消息内容 */
    DispatchProcess (Recv_PID, &Message);
                                 /* 根据进程号并查阅 PCB 表来调用该 PID 号对应的进程 */
  }
}
```

在这里的实现中,需要定义二个数据结构:一个是消息队列;一个是进程控制块(PCB)。前者用于存放各种进程所发出的消息,后者用于存放进程编号和进程代码地址的对照表,以便 OS 内核能根据此表和进程号 PID 调用对应的应用进程代码。

消息队列数据结构定义如下:

```
#define   MAX_ MESSAGE_LEN 48
                                 /* 单个消息的最大长度 */
#define MAX_MESSAGE_NUM 100
                                 /* 系统最多支持的消息个数 */
typedef struct {                 /* 消息队列中的单个数据项结构定义 */
  int Send_PID,                  /* 发送消息的进程号:由发送进程填写 */
    Recv_PID,                    /* 接收消息的进程号:由发送进程填写 */
    Link,                        /* 队列连接指针,表明后面一个消息位置(数组下标)指针由 OS 填写 */
```

```
    len;                    /* 有效消息内容长度 */
    char Message[MAX_ MESSAGE_LEN];
                            /* 消息内容,由发送进程填写 */
} MessageQueueTerm;         /* 消息队列数组变量定义,每个消息占用一个数组元素 */
MessageQueueTerm MyMessageQueue[MAX_MESSAGE_ NUM];
int QeueuHead, QeueuEnd; /* 消息队列的头尾指针定义 */
```

为简单起见,这里没有使用内存动态申请方式来得到消息列数据项的存储空间,而采取固定存储单元长度来表示队列数据相(单个消息),并且整个队列的长度也采取固定长度。在一些小型专用系统中,由于进程的可预知性,消息的长短和数量也是可预知的,并且由于是非抢先式任务调度(独占 CPU),存放到消息队列中的消息总是能立即得到执行。因此,可采取固定的和有限的队列长度设计。

OS 内核函数 ReadMessageQueue()在得到消息的内容和其中的接收进程号(PID)之后,就必须根据一个 PCB 表来得到对应的应用进程代码的入口地址。因此,必须定义一个数据结构来表示进程号和进程代码首地址的对应关系。定义如下:

```
#define MAX_APPPROC_NUM 10
                            /* 定义本系统中的最大应用进程数目 */
typedef struct {            /* PCB 表定义 */
  int PID;                  /* 进程号 */
  void (*AppProc)(void *Message);
                            /* 该进程号对应的进程代码首地址 */
} PCB;
```

假设现在的系统中有一个处理通信的应用进程,其主函数名称为 CommProc,进程号为 COMMPID,则可对 PCB 表按如下格式进行初始化:

```
#define COMMPID 1
void CommProc (void *Message);
PCB ProcTable[MAX_APPPROC_NUM] = {
  COMMPID, CommProc,
  }
```

这样,进程调用函数 DispatchProcess 的代码可写为

```
void DispatchProcess (int PID, MessageQueueTerm *Message)
{
int i;
  for (i = 0; i < MAX_APPPROC_NUM; i++)
    if (PID = = ProcTable[i].PID)
                            /* 根据 PID 查找对应的进程 */
              (*ProcTable[i].AppProc)((void*)Message);
                            /* 调用该进程 */
}
```

以上对进程调度和消息传递的具体实现过程进行了说明,另外必须说明的一点是,在主函

数开始时(死循环之前),还必须有关于 OS 本身和各应用进程初始化代码,见本文中的第二、6 小节。

3. 定时器任务和定时任务

(1) 定时器任务和定时任务的概念和区别

为应用进程提供定时任务和定时器任务触发功能是操作系统平台的基本功能。定时任务是指间隔为固定周期的需要执行的功能。例如一个数据采集系统中必须间隔 1s 对外部物理量进行数据采集,就是一种定时任务。定时器任务的概念像日常生活中的闹钟,是应用进程在某个特定时刻启动计时,等候固定周期后必须执行的任务。比如在通信程序设计中,发送方在发送数据后立即启动一个定时,并在约定的 3s 后仍然未收到对方的响应数据,则执行一个超时处理任务(函数)来报告通信出现了问题或者重新发送数据。在没有操作系统平台支持的软件设计中,以上功能一般是通过定时中断所产生的时基并结合软件的查询等待来实现的。现在讨论如何在 OS 平台上提供这种定时任务触发功能。

从以上叙述可知,如果在定时器任务被触发后,又重新启动计时,则定时器任务就成了周期性的定时任务了,因此,二者的设计可以综合起来考虑。为了通过 OS 来触发应用进程的定时任务和定时器任务,应用进程所做的工作包括:

① 启动定时器任务或定时任务的计时开始,确定定时周期长度;

② 编写定时器任务或定时任务处理函数。

对于 OS 来说,所完成的功能包括:

① 产生一个时间基准;

② 记录应用进程的启动计时时刻和时长,同时纪录应用进程启动的定时任务类型;

③ 不断判断定时的时间是否到达,如时间周期到,则触发定时器任务或定时任务函数;

④ 如是定时任务,则由系统自动再次启动定时。

(2) 定时(器)数据结构定义

在一个应用系统中,定时器任务或定时任务的数量可以是很多个,因此,先定义一个数据结构对每个定时(器)任务进行管理,同时定义一个能同时登记 30 个定时(器)任务的结构数组。

```
#define MAX_TIMER_NUM 30
typedef struct {
  int Timer_id,                    /* 定时器标识 ID */
      Type,                        /* 类型,1 表示是定时器,2 表示是定时任务,0 表示空 */
      Time,                        /* 定时时长(周期),时间单位见下文说明 */
      TimeBackup;                  /* 时长数值备份,用于定时器时长重置 */
    void (*TimerProc)(void);       /* 超时处理函数 */
} TIMER;
TIMER Timer_Table[MAX_TIMER_NUM];
                                   /* 定时任务登记表 */
```

在应用进程启动定时时,只要告诉 OS 有关类型、时长周期和超时处理函数名称等信息,OS 就负责将该定时器的相关信息登记在 Timer_Table[] 中,并负责不断判断定时时间间隔是否到达,如到达则启动定时。那么,首先应如何启动一个定时器呢?

(3) 定时(器)任务的启动(登记)

OS 提供一个定时(器)任务启动函数来供应用进程调用,现在我们知道,该启动函数的入口参数就是类型、时长周期和超时处理函数名称等。函数作用就是将这些信息登记在 Timer_Table[]中,则该函数的代码可写为

```
void TimerStartup (int Timer_id, int Type, int Time, void ( * TimerProc) (void))
{                                   /* 入口参数:定时器 ID 号、类型、时长,超时处理函数代码的首指针
*/
int i;
  for (i = 0;i < MAX_TIMER_NUM;i + +)
                                    /* 遍历定时器登记表 */
    if (Timer_Table [i]. Type = =0) {
                                    /* 如是空表项,则登记之 */
    Timer_Table [i]. Type          = Type;
    Timer_Table [i]. Timer_id      = Timer_id;
    Timer_Table [i]. Time          = time;
    Timer_Table [i]. TimeBackup    = time;
    Timer_Table [i]. TimerProc     = TimerProc
    }
}
```

这里由于定时器登记表是采用线形数组,表的处理则必须逐个遍历,在定时器数量大的情况下会影响速度,可使用双向链表、排序插入等方法改进。

例如,对于通信进程中,发送数据后需要启动一个接收响应超时定时器,时长为 3s,超时处理函数为:

```
void TimerComm (void);
```

则只要在通信进程中发送数据后加入以下代码启动定时器即可:

```
TimerStartup (TIMER_ACK_OVERTIME,1,3000/50, TimerComm);
```

下面讨论 OS 如何在设定的时长到达后启动这个超时处理函数?

(4) 产生定时时间基准和超时处理函数的触发

在定时器启动时,我们设置了时长,因此,OS 只要将此时长数值按一个时间基准进行减值,时长的数值减至 0 时,调用对应的超时处理函数就可以了。那么,如何产生这个时间基准呢?

对于嵌入式微处理器,一般均有片内的硬件定时器,时间基准必须使用硬件定时中断服务例程来得到。具体做法是由定时中断服务例程在间隔固定时间单位(比如 100 ms)后产生一个标志位,OS 查阅此标志,如标志有效,则将定时器表中所有的时长减去一个单位,同时将此标志清 0。为了保证定时器的定时精度,定时中断服务例程所产生时基标志的时间间隔单位应大大小于最小的定时时长单位,例如,时基标志间隔为 50 ms,最小时长单位定为 1 s。

定时中断服务例程的编写在不同的硬软件平台下有所区别,这里,以 PC 硬件平台、TC2.0 软件环境来给出定时中断服务例程的代码。我们知道在 PC 中,硬件定时的中断向量号是 8,硬件定时中断的时间间隔为 18.2 次/s,也就是 55 ms(在这里为计算方便,姑且按 50 ms 计)。

因此我们编写的中断服务例程函数如下：

```
int FlagOf55ms =0;
void interrupt Timer55ms(void);
void interrupt Timer55ms(void) {
        FlagOf55ms =1;
}
```

而在 main 函数的一开始，必须设置新的中断向量，即设置中断向量 0x08 指向我们编写的中断函数：

```
setvect(0x08, Timer55ms);
```

现在，已经设置了定时基准，下面在系统的主循环中来处理定时器表。在前面的 main 函数中加入代码：

```
main()
{
…
    setvect(0x1c, Timer55ms);
    while(1) {                          /*嵌入式软件总是一个死循环*/
    if (FlagOf55ms ==1) TimerProcess();
                                        /*如果一个时间单位到，则处理定时器表*/
…
    }
}
```

定时器表处理函数 TimerProcess 的作用是查阅定时器表，将表中的各定时器的时长数值递减，并触发时长数值为 0 的定时器超时处理函数。因此可写出代码如下：

```
void TimerProcess(void) {
  int i;
  for (i=0;i<MAX_TIMER_NUM;i++) {
    if (Timer_Table[i].Type ==0) continue;
    if (Timer_Table[i].Time ==0) {
        (*Timer_Table[i].TimerProc)();
                                        /* 调用超时处理函数*/
        if(Timer_Table[i].Type ==1)
                                        /*是定时任务，则重置时间常数*/
            Timer_Table[i].Time = Timebackup;
        else Timer_Table[i].Type ==0;
                                        /* 否则，清除定时器类型标志*/
    };
    Timer_Table[i].Time--;              /* 递减时长*/
    }
}
```

如果需要在定时器和进程之间进行数据传递,可以采用进程和进程之间同样的方式,以上代码需要加以适当修改。

4. 内存管理方法

对于一个小型嵌入式系统来说,内存管理的功能可以简单化,甚至不要。因为各应用进程对存储器的需求是知道的,可以各自使用固定和独立的存储器区域,而不需要通过 OS 来控制和分配。

5. 操作系统的 I/O 功能实现

嵌入式应用的操作系统的 I/O 功能与系统硬件平台相关。由于嵌入式应用硬件平台各不相同,实现的代码也是不同的。这里要说明的是,I/O 功能的编程可以同高层应用进程一样,使用操作系统内核提供的消息传递、进程调度和定时器功能。也就是说,将 I/O 操作软件当作应用进程来设计,具体方法不再详述。

6. 系统初始化和进程初始化处理

对于一个完整的操作系统,各种初始化代码是必须的。初始化代码是软件中仅在启动时执行一次的代码,其作用是为正常运行提供各种条件,包括硬件电路、寄存器和软件变量的各种初始设置。我们可将初始化代码分为系统初始化代码和应用进程初始化代码二部分,应用进程初始化代码需要通过 OS 内核来调用。为此,我们在进程控制表中增加进程初始化代码指针,并将结构类型 PCB 修改如下:

```
typedef struct {
    int PID;
    void (*AppProcInit)(void);
                                    /*该进程号对应的初始化代码首址*/
    void (*AppProc)(void *Message);
} PCB;
```

可见,非抢先式实时多任务操作系统的内核代码是短小的,也是非常容易实现的。笔者优化和完善调试后的 OS 内核代码的 PC 版本约为 12KB;C51 版本的 HEX 文件大小约为 4KB(C51 版本需要作一定修改,如中断函数、函数指针中的使用等);8051 系列的汇编版本大小不足 800 字节。利用这种 OS 平台来设计软件的好处主要是,软件的模块性好,层次分明,同时也使软件设计更加简单了。

三、基于 OS 平台的应用进程设计

1. 应用进程的设计方法

为了说明应用进程如何利用 OS 内核,我们设计了一个用于演示的应用程序,该应用程序的主要功能如下:

① 在屏幕的中间显示一个秒表。

② 根据键盘输入的数字,在对应的以 s(秒)为单位的时间后,在屏幕上显示一行字串:"Welcome To You."例如,输入数字 5,则间隔 5 s 后显示该字符串,同时显示:"2 秒后停止显示"。

为了实现该演示程序的设计,我们设计了 2 个应用进程:一个用于接收键盘输入的数字,另一个用于完成以上各种要求的数字和字符显示。键盘进程通过消息将键数字传递给显示进程,同时在键盘进程中设计一个定时任务,每 200 ms 读一次键盘;而在显示输出进程中设计 2

个定时器,一个用于在接收到键盘进程的键盘消息后启动,时长为 N (N 等于接收的键值),在其超时函数中完成显示字符串:"Welcome To You",并启动第二个定时器,时长为 2,在第二个定时器的超时处理函数中清除显示字符串。

完整的非抢先式实时多任务内核和应用进程举例的代码见本文附录。

2. 应用进程设计应该注意的问题

从上面的例子中,已经知道了利用 OS 内核来设计一个应用进程的方法和步骤,但必须注意的是,这种非抢先 OS 内核和抢先式 OS 内核有一个本质区别,就是系统在应用进程正在执行时不能暂停其执行。也就是说,这种多任务实际上是一种伪多任务,应用进程不能长时间独占 CPU,代码中不能有长时间等待代码,更不能存在无限循环和无条件等待代码,否则,其他进程将不能执行。应用进程在对一次外部事件处理完后应立即退出,比如,上例中的显示进程在收到键盘进程的键盘消息并完成本次的字符显示后,就停止执行了。

应用进程的代码设计最好采取有限状态机 (FSM) 方法。

四、总 结

本文给出了一种简易操作系统内核的设计方法,该设计在小型嵌入式系统中有很好的使用价值。笔者也在 C51 下进行了实现,其 HEX 格式文件代码约为 4 KB;同时,利用这种设计思想,也可将此设计用汇编语言实现 (实现的汇编语言代码约为 800 KB),并在汇编编程时加以利用。笔者在此系统上设计了多个应用系统,缩短了软件开发周期,增加了软件的可读性和可维护性,避免了使用商用实时操作系统在技术上和价格上的花费和硬件上的不适应性。这种设计在一些弱实时性应用、数据存储器容量有限的各种类型单片机系统中具有良好的适应性和使用价值。不仅保证了同一个应用软件在不同的 MCU 上的快速移值,也使得在同一 MCU 硬件平台上开发不同的应用系统更加容易 (因为不需要为每个应用软件编写硬件接口代码)。

附录 非抢先式多任务实时操作系统及其应用举例代码 (限于篇幅,此处省略。本刊网上会发布带有完整附录的全文)。

参 考 文 献

1 RTX51 FULL/TINY REAL - TIME OPERATING SYSTEM User's Guide. Keil Software Co.

2 Tanenbaum Andrew S. 现代操作系统. 陈向群译. 北京 机械工业出版社

选自《单片机与嵌入式系统应用》月刊,2002 年第 1 期

3.5 单片机程序设计中运用事件驱动机制

广西梧州师范高等专科学校物理与电子信息系 (542800) 蒋 翔

一、传统单片机程序开发的不足

在传统的单片机程序中,通常是以“过程”和“操作”为中心的结构,程序按规定的过程顺序地执行,与外设的连接一般采用中断方式,在中断服务程序中完成外设的全部处理工作,主程序一般只是初始化系统并等待中断的发生。这种结构成熟、易于理解,但有如下不足:

⑴ 受单片机性能的限制,容易造成系统对其他中断的响应变得迟缓,特别是对于中断源较多、中断处理耗时较多的系统 (如:LED 显示、键盘扫描等);

⑵ 中断服务程序过长,在中断服务期间系统无法响应同级的中断;

⑶ 可能导致代码重入,增大堆栈开销,造成难以预料的结果;

⑷ 程序调试时,花在各模块定时协调方面的时间、精力随系统的复杂程度大幅增加。

如果在编写单片机程序时,引入 Windows 程序中的事件驱动机制,把中断响应与事件处理程序分离,中断服务程序的任务只是产生一个中断产生的标志,而事件处理则由处理程序来完成,主程序则负责判断标志和调度处理程序。这样,可大幅缩短中断服务程序的长度,减少中断服务程序的耗时,提高系统对多中断的响应能力,从而较好地解决上述矛盾。

二、Windows 的事件驱动机制

在 Windows 系统中,程序的设计围绕事件驱动来进行。当对象有相关的事件发生时 (如按下鼠标键),对象产生一条特定的标识事件发生的消息,消息被送入消息队列,或不进入队列而直接发送给处理对象,主程序负责组织消息队列,将消息发送给相应的处理程序,使相应的处理程序执行相应的动作,做完相应的处理后将控制权交还给主程序。

在这种机制中,对象的请求仅仅是向队列中添加相应的消息,耗时的处理则被分离给处理函数。这种结构的程序中各功能模块界限分明,便于扩充,能充分利用 CPU 的处理能力,使系统对外界的响应准确而及时。

三、事件驱动的单片机程序设计

与 Windows 系统相比,单片机的资源非常有限,因此,单片机程序中的事件驱动机制只能采取一种简化的方式。当某个中断发生时,中断服务程序设置相应的标志,不同的标志代表不同的中断发生的消息,而主程序不断地判别这些标志,以决定启动哪一个处理函数。相应的处理函数被启动处理完相关的任务后,清除此标志,然后把控制权交还给主程序。采用这种机制,可合理地利用有限资源,使程序调试的工作量大幅下降。对于延时、定时处理 (如 LED 显示、键盘扫描等),更可方便地使用一定时器来完成延时、定时的任务,从而把 CPU 从这种耗时的任务中解放出来,确保系统对多中断有足够的响应能力。

本文以一 IC 卡读写机为例,说明事件驱动机制在单片机程序设计的具体应用。

1. 硬件结构

本系统以 ATMEL 公司的 89C51 为核心(见图 3.5-1)。89C51 价格低廉,性能较好,片内有 4 KB 的可擦写程序存储器,可满足本系统的要求。为简化硬件结构及系统能耗,键盘采用软件扫描的矩阵键盘。LED 显示采用段位动态扫描,在任一时刻 LED 中最多只有一段被点亮。具体是在位选信号送某位 LED 的公共极时,每隔一个时间片依次输出该位 LED 的段码(含小数点),输出完成一位后,再逐次输出下一位。从第一位至第 N 位 LED 依次分成 8×N 个时间片循环扫描显示。串口 UART 作为系统与外部数据通信的通道,IC 卡的读写由 MCU 模拟 I^2C 协议来实现。

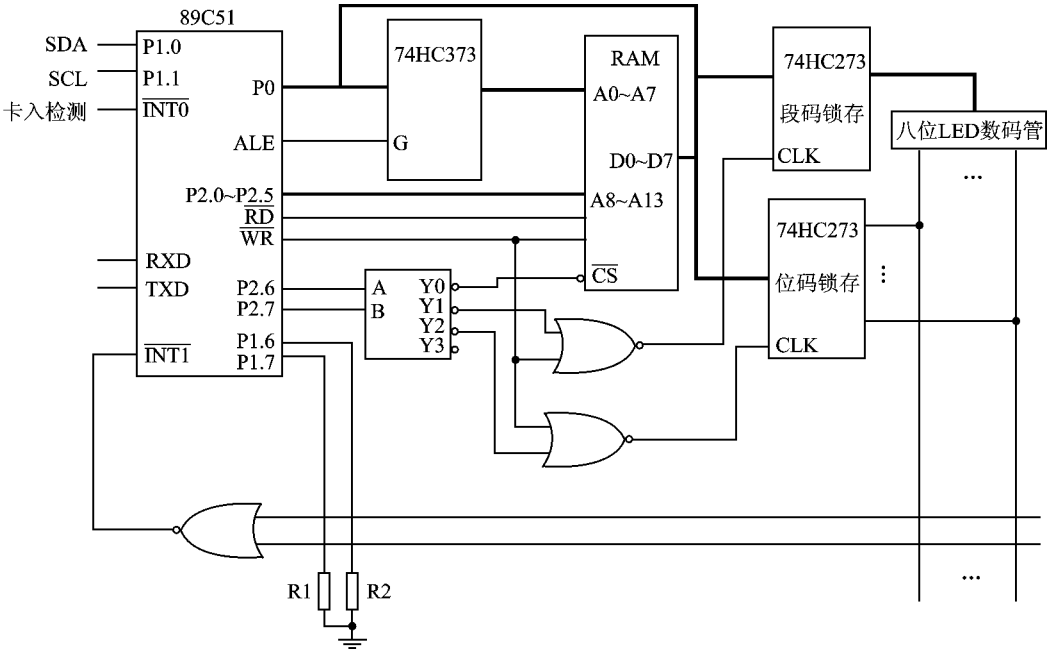


图 3.5-1 IC 卡读写机原理图

2. 事件驱动机制的单片机程序设计

(1) 中断申请标志

在系统中定义一个可位寻址的单元,在此把它命名为 Message_Flag,用来记录描述中断事件发生的情况。各位的定义如下:

D7	D6	D5	D4	D3	D2	D1	D0
键盘"确认"标志			串口事件发生标志		INT0 事件发生标志	T1 事件发生标志	T0 事件发生标志

* Message_Flag 中某位为 1 表示当前有相应的事件发生,为 0 则当前没有相应的事件发生。

(2) LED 显示的实现

显示模块结构见图 3.5-2。以定时器 T0 作为 LED 的动态扫描的定时基准,T0 的定时时间最大值 Tseg = 20 ms/(8×N)(其中 N 为 LED 位数),改变 Tseg 的值可改变显示的亮度。T0

每隔 Tseg 时间向 MCU 申请中断，在 T0 的中断服务程序中置位相应的标志位（Message_Flag 中的 D0 位）。主程序检测到此标志位被置位后，启动显示模块实现位段的显示输出。

⑶ 键盘输入的实现

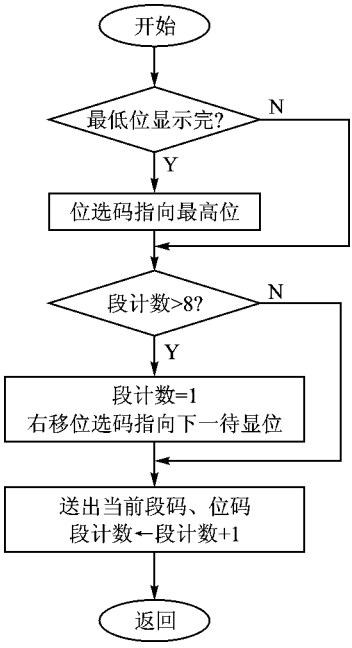


图 3.5－2 显示模块

键盘模块结构见图 3.5－3。在 LED 动态扫描期间，只有被点亮的 LED 相应的位选线维持大约 3 ms 的低电平，而在系统工作的绝大部分时间内 LED 的位选线（即键盘的列线）维持高电平。当有键被按下时，将把键盘的行线中某一根拉成高电平，经或非门后，向 MCU 申请 INT1 中断，在 INT1 的中断服务程序中启动定时时间为 20 ms 的定时器 T1。T1 的定时时间到后向 MCU 申请 T1 中断，在 T1 的中断服务程序中置位相应的中断申请标志（Message_Flag 中的 D1 位）。主程序检测到此标志位被置位后，启动键盘扫描模块实现键盘输入。键盘输入完成（用户按“确认”键），置位键盘输入确认标志（Message_Flag 中的 D7 位）。

⑷ IC 卡的读写

IC 卡的 SDA、SCL 经卡座分别通过 P1.0、P1.1 与 MCU 相连。当 IC 卡插入卡座时，座上的微动开关使 INT0 变为低电平，向 MCU 申请 INT0 中断。在 INT0 中断服务程序中置位相应的中断申请标志（Message_Flag 中的 D2 位），主程序检测到此标志位被置位后，启动 IC 卡的读模块，以软件模拟 I^2C 协议来实现读卡操作。在数据处理完成后，同样通过软件模拟 I^2C 协议来完成写卡的操作。

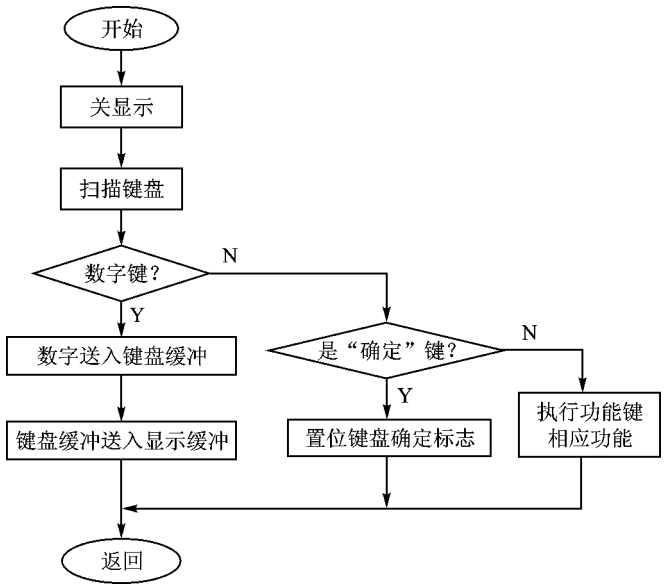


图 3.5－3 键盘模块

⑸ 串口通信

实际应用中可把 UART 转换成 RS232C 与 PC 相连或转换成 RS485 等其他协议组成单片机网。MCU 与外部的通信采用中断方式,在串口的中断服务程序中置位相应的中断申请标志(Message_Flag 中的 D4 位)。主程序检测到此标志位被置位后,启动串口通信模块,实现与外部的数据通信。

⑹ 主程序的设计

综上所述,主程序首先完成系统的初始化,然后循环检测各中断的中断申请标志,如有某标志被置位,则启动相应的处理模块完成相应的任务。程序结构如下(用 C51 编写):

```
unsigned bdata message_flag;
sbit t0_int = message_flag 0;
sbit t1_int = message_flag 1;
sbit int0_int = message_flag 2;
sbit uart_int = message_flag 4;
sbit kb_enter = message_flag 7;
unsigned char kb_buf [8];
unsigned char led_buf [8];
unsigned char ic_buf [8];
unsigned char num_buf [8];
void num_proc (void);                                   /*数据处理模块*/
void ledbuf_write (unsigned, unsigned int);             /*数据处理*/
void system_init (void);                                /*系统初始化*/
void uart_commune (void);                               /*串口通信模块*/
void led_display (void);                                /*LED 显示*/
void kb_scan (void);                                    /*键盘扫描*/
void ic_reader (void);                                  /*读 IC 卡*/
void ic_writer (void);                                  /*写 IC 卡*/
void set_timer (unsigned int time_len, unsigned char type, unsigned char id);
                                                        /*设置定时器*/
void t0_int_sever (void);                               /*定时器 T0 中断服务*/
void t1_int_sever (void);                               /*定时器 T1 中断服务*/
void int0_int_sever (void);                             /*INT0 中断服务*/
void int 1_int_sever (void);                            /*INT1 中断服务*/
void uart_int_sever (void);                             /*串口中断服务*/
void main (void)
{
    system_init 0;
while (1) {
    if (t0_int)        led_display 0;
    if (int0_int)      ic_reader 0;
    if (t1_int)        kb_scan 0;
    if (uart_int)      uart_commune 0;
    if (kb_enter) {
```

```
            num_proc();
            ic_writer();
            ledbuf_write(num_buf,8);
        }
    }
}
```

事件驱动的单片机程序设计是通过在中断服务程序中置位相应标志,把耗时的中断服务中的处理部分分离出来,中断返回后,再由主程序根据标志启动相应的处理模块。在任务处理完成后,清除相应的标志。由于中断服务程序短小,所以一般能实时地响应各种中断;而处理程序之间不会被相互调用,所以不会产生代码重入;各模块界限分明,给程序中各模块的统调带来很大的方便。

实践证明,运用事件驱动机制来组织单片机程序,即使对于要求定时准、耗时多的多中断、多模块系统,也可轻松地完成。

参 考 文 献

1 何立民. MCS-51 系列单片机应用系统配置与接口技术. 北京:北京航空航天大学出版社,1990
2 马忠梅等. 单片机的 C 语言应用程序设计. 北京:北京航空航天大学出版社,1997
3 黄维通. Visual C++面向对象与可视化程序设计. 北京:清华大学出版社,2000

选自《电子技术应用》月刊,2002 年第 7 期

3.6 实时操作系统 RT－LINUX 的原理及应用

湖南邵阳高等专科学校计电中心 (422004) 周红波

一、引 言

在 RT－LINUX 出现之前,在为实时应用选择系统平台的时候,人们大抵只有两种选择,要么使用 DOS 并自己编写所有必要的驱动程序,要么就得购买专用的实时系统。前者不仅费时费力,其性能也难以令人满意。而后者性能虽佳,其价格却高得让人难以接受。

RT－LINUX 的出现解决了这一问题,它为实时应用领域的研究与开发提供了一个物美价廉的完备的操作系统平台。凭着自身的技术特色,借助于 LINUX 的强大功能,RT－LINUX 下开发出的实时应用有着不俗的表现。

RT－LINUX 是在 LINUX 基础上改写的具有硬实时处理能力的实时操作系统。该系统主要是由美国新墨西哥州的矿业技术学院于 1996 年研究开发出来的。最初这个项目的目的有两个:一是设计出一个非私有的可以用来操控科学仪器和机器人的工具;二是利用 RT－LINUX 来进行实时和非实时操作系统设计的研究工作。

在 RT－LINUX 中,实时任务的运行和中断的处理都与原 LINUX 并存于同一台计算机上。它们可以在任何需要的时候获得处理机,而不必理会当时 LINUX 正在做什么。它所建立的是一个直接面对处理器的小内核,此内核独立于原 LINUX 的标准内核,并且拥有自己的调度程序。原 LINUX 内核在这个小内核的基础上以最低的优先权与其他实时任务分享处理器。

在一台普通的 X86 计算机上,RT－LINUX 的最坏响应时间可达 15 ms 以下,而周期性实时任务的周期可以达到在 25 ms 以内。这些性能指标是与硬件的性能相关的,也就是说,如果升级硬件,RT－LINUX 的性能也会随之提升。

这里需要说明的是,标准 LINUX 也为实时任务提供毫秒级的调度精度,但这是一种软实时,因为标准 LINUX 的设计初衷并未考虑提供亚毫秒的精度以及可靠的时限保证。

二、RT－LINUX 的原理

RT－LINUX 把 LINUX 的系统内核当作在一个小的实时系统上运行的一个任务来对待。事实上,LINUX 被当作实时系统的空闲任务,也就是说,只有当系统中没有任何实时任务需要运行的时候,才来运行它。LINUX 的任务再也不能够通过封锁中断来避免被抢占。实现这一目的的技术关键在于对中断控制硬件的软件仿真。当 LINUX 通知特定硬件机构禁止中断的时候,实时内核将截获这一请求,将其记录然后返回给 LINUX。LINUX 等于被“欺骗”,这样 LINUX 就不能真正地禁止硬件中断。所以不管当前 LINUX 处于什么状态,都不能延长实时系统的中断响应时间。

当一个中断产生时,RT－LINUX 将首先截获它并决定该如何处理。如果该中断有相应的实时处理程序,该程序就被调用。如果没有或者该程序表示与 LINUX 共同处理此中断,中断

就被挂起。然后在 LINUX 请求开放中断时,挂起的中断就被仿真,软件中断送到 LINUX 后,LINUX 的响应中断处理程序就被调用,同时硬件中断被重新开放。

不论 LINUX 处于何种状态,核心态也好,用户态也罢,或者当 LINUX 正在封锁中断或正在开放中断,或者陷于死锁状态,实时系统都能够响应外界的中断。

RT－LINUX 自带一个完全抢占、静态优先数的缺省调度程序,它把 LINUX 的任务设为最低的优先数。如果实时任务消耗掉全部的处理器时间,LINUX 的任务就得不到一点 CPU 时间来运行,系统看起来就好像挂起。

三、RT－LINUX 的实现

1. 中断仿真

在中断控制硬件与 LINUX 核心之间放置一个软件仿真层。具体做法是,在 LINUX 源码中出现 cli、sti 和 iret 的所有地方都用仿真宏 S_CLI、S_STI 和 S_IRET 来替换。所有的硬件中断就都被仿真器所截获。

当需要关中断时,就将仿真器中的一个变量置 0。不论何时若有中断发生,仿真器就检查这个变量。如果是 1 (LINUX 已开中断),就立即调用 LINUX 的中断处理程序;否则,LINUX 中断被禁止,中断处理程序不会被调用,而是在保存着所有挂起中断的信息的变量的相应位置 1。当 LINUX 重新开中断,所有挂起中断的处理程序都会被执行。这种仿真方式可以称之为“软中断”。

2. 实时任务

实时任务是在一个由核心控制的调度程序的调度下执行的用户定义的程序。

RT－LINUX 最初将实时任务设计成 ELF 格式的目标文件。这一设计方案的最大缺点就是性能比较差。原因在于:(1) 486 的缓存是虚拟的。所以每当页表目录的基址寄存器改变时,TLB (转换后备缓冲器) 就会失效。由于实时任务的上下文转换频繁,所以 TLB 的频繁失效就导致系统性能的严重下降。(2) 486 的保护级别就换耗时不少。比如,陷入更高级别时需要 71 个循环,而其他指令一般少于 10 个循环。

解决的办法就是使用可加载模组技术,所有的实时任务都同处于一个地址空间－内核地址空间,不仅避免了频繁的 TLB 失效,同时也消除了变换保护级别的消耗,而且任务转换也变得相当容易。

3. 进程调度

实时系统的进程调度的主要任务就是满足实时任务时间上的要求。调度算法的种类很多,没有一个策略是放之四海而皆准的,因此采用哪种算法要取决于具体应用。

RT－LINUX 采用的方法是允许用户编写自己的调度程序,并可以编译成模组的形式。这样就可以方便地试验不同的策略和算法对于某一特定应用的适合性,从中选出最优。RT－LINUX 自带的是一个基于优先数的抢占式调度程序。此调度程序将 LINUX 当作具有最低优先数的实时任务。因此,LINUX 只在实时系统无任何实时任务时才运行。在从 LINUX 切换到实时任务时,系统记下软中断的状态并禁止软中断。在切换回来时,再恢复软中断的状态。

4. 时 钟

调度程序需要精确的时钟才能准确操作。调度通常是在特定的时刻进行任务切换。时钟的偏差会引起预定调度的偏差,导致产生被称为任务发布抖动的现象。这是一种应该尽量避

免的不良现象。

RT－LINUX 的解决办法是，将 IBM PC 兼容机中的时钟芯片 Intel 8254 设置为中断开启终端计数模式。在这种模式下，精度可以达到 1 ms。这样在降低中断处理的影响的同时，获得了较高的时钟精度。

5. IPC

由于标准 LINUX 核心可以被实时任务在任意时刻抢占，所以实时任务无法安全地调用 LINUX 的程序。但是总要有一个信息交换的机制。

在 RT－LINUX 中所有的信息交换方式是 RT－FIF0（实时队列）。它与 LINUX 的管道非常相似，都是一个无结构的数据流。通过 RT－FIF0，LINUX 的进程之间，实时进程之间，以及 LINUX 的核心与实时进程之间可以交换信息。

对于一个普通的进程来说，RT－FIF0 就是一个特殊的字符文件。这些文件必须自建：

```
# for i in 0 1 2 3;do mknod/dev/rtf$i c 63 $i;done
```

四、RT－LINUX 的应用

编制基于 RT－LINUX 的实时应用程序有一个基本宗旨，那就是，编制出与时间直接相关的代码，因为这一部分通常包含与实时硬件的底层交流，对于时间要求苛刻。

鉴于这样的想法，一般把应用程序分成两块。一块是硬实时部分，以模组方式实现，运行时作为实时任务，具有比 LINUX 高的优先权，负责从硬件设备将实时应用中产生的或是需要处理的信息和数据取出，然后送入 RT－FIF0 中。另一块是用户部分，因其运行于用户空间而得名。这一部分编程的比重较大，使用通常的编程方式实现，运行时作为 LINUX 下的一个普通进程，负责从 RT－FIF0 的另一端取出信息和数据进行处理、分析、显示以及与其他 LINUX 进程交互。

综上所述，RT－LINUX 不仅仅是将分时系统改写为一个具有硬实时能力的实时操作系统的成功尝试，更具有学术价值的是，它提出了一种新的架构操作系统内核的方法，可以应用于以后的操作系统的设计中。

参 考 文 献

1 刘庆龙，陈宗海. LINUX 环境下基于 Agent 的自主机器仿真系统. 计算机应用，2001.5

2 Phil corfles. The Linux A－Z (M). Prentice Hall Europe. 1997

选自《微计算机信息》月刊，2002 年第 5 期

3.7 RT－Linux 的实时机制分析

广州华南理工大学计算机学院 (510640) 吴一民

一、引 言

实时操作系统是指系统中应用的输出结果不仅要保证逻辑上的正确性,而且要保证在规定的时间内能够得到正确结果。实时系统主要有两大类:一类称之为硬实时,其实时任务必须在规定的时限之前完成,否则将会产生严重的后果;另一类则称之为软实时,其偶尔的时限丢失并不会引起太大的问题。

传统上的实时系统基本上都是专用的操作系统,性能比较好,但价格昂贵。因此国外许多研究机构都在致力于 Linux 的实时化研究,而且已经取得了一定的成果,RT－Linux 就是其中一个典型的例子。

Linux 本身是一个通用性的操作系统,它着重强调的是系统的整体性能,提倡进程调度的公平性,尽管它也符合 POSIX1003.4 实时标准,但由于 Linux 内核的不可抢占性、定时粒度太粗以及缺乏有效的实时调度策略,其实时能力太弱。

RT－Linux 是美国新墨西哥州大学计算机系 M. Barabanov 和 V. Yodaiken 等人通过对 Linux 的改造而实现的一个硬实时 Linux。其设计者认为,任何实时应用都可分为实时部分和非实时部分两个部分。实时部分的处理逻辑一般比较简单,不需要复杂的人机界面设计,但要求对实时事件能够立即响应,作出必要的处理。因此,这部分实际上是一种反应型系统。而应用中时间要求不高的部分如用户界面等则可交由非实时部分处理。基于这种思想,RT－Linux 的设计者把单内核结构的 Linux 改造成为一个双内核操作系统,实时内核处理紧急的事件,而 Linux 原来的内核则作为实时内核的一个任务以最低优先级运行。实时内核和分时内核之间通过实时管道 FIFO (First Input Fist Output) 进行通信。同时,RT－Linux 也改进了传统 Linux 的定时机制,使其定时精度可达到微秒级,这样满足了硬实时的时间粒度要求。

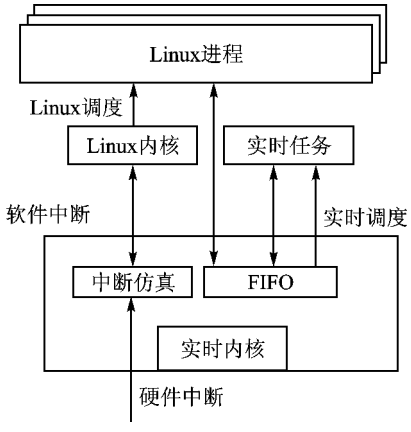


图 3.7－1 RT－Linux 的结构图

从图 3.7－1 中可以看出,RT－Linux 的实时内核主要由实时调度器、中断仿真层以及 FIFO 处理三部分组成。

二、中断仿真

RT－Linux 在 Linux 和中断控制器之间加入了一个中断控制的仿真层，作为其实时内核的一部分。非实时内核中所有的 cli、sti 和 iret 分别被替换为 S_CLI、S_STI 和 S_IRET 等几个宏。Linux 中的允许中断和禁止中断，实际上变为仅仅设置和清除了一个中断允许/禁止变量的值，而不是真正对 CPU 允许中断和禁止中断。当中断发生时，实时内核首先截获中断并进行一定的处理，然后再把中断传给 Linux 内核。

以下是 S_CLI、S_STI 和 S_IRET 的代码。

```
S_CLI：  movl     $ 0,SFIF

S_IRET：push      % ds
         push     % eax
         push     % edx                    //保存寄存器
         movl     $ KERNEL_DS,% edx
         mov      % dx, % ds
         cli
         Movl     SFREQ,% edx
         andl     SFMASK,% edx
         bsrl     % edx, % eax             //查找是否有被挂起的中断
         jz       not_found
         Movl     $ 0,SFIF                 //有被挂起的中断
         Sti
         Jump     SFIDT (,% eax,4)
         not_found：                       //无被挂起的中断
         movl     $ 1, SFIF
         sti                               //允许中断
         popl     % edx
         popl     % eax
         pop      % ds                     // 恢复现场
         iret                              //硬返回
S_STI：  pushfl
         pushl    $ KERNEL_CS              //保存当前的现场
         pushl    $ done_STI               //以准备以后检查是否有被挂起的中断
         S_IRET
         done_STI：
```

其中用到的关键变量意义为：

SFIF：中断允许/禁止标志；

SFREQ：记录被挂起的中断；

SFMASK：中断屏蔽变量。

当 Linux 试图屏蔽中断时，实际上运行的是宏 S_CLI。它对 SFIF 进行了标记，并没有真正屏蔽中断，也就是说实时内核仍可处理中断。当发生中断时，实时内核首先接受中断请求，如

果实时内核存在对应的中断服务程序,则转至该服务程序,由实时内核的中断服务程序处理该中断。否则查看变量 SFIF,如果允许中断,则将中断交给 Linux 中断服务程序,如果不允许中断则将记录在变量 SFREQ 中(挂起中断)并返回。

当 Linux 试图允许中断时,实际上运行的是宏 S_STI。它首先保存当前的现场,然后执行一个软返回 S_IRET。

S_IRET 这个宏的执行的时机有两个:一个在从 Linux 的中断返回时;另一个则是宏 S_STI 执行结束时。S_IRET 首先检查必要的寄存器以备处理被挂起的中断,然后检查是否有被挂起的中断。如果有,则依优先级的高低顺序依次处理;最后恢复现场执行硬返回。值得注意的是 S_IRET 在检查有无被挂起的中断时,会禁止中断,这段时间非常短暂,仅几条指令的执行时间,与 Linux 在执行系统调用时禁止中断的时间相比可以忽略不计,因此解决了因中断服务导致的实时调度延迟的不确定性。

三、调 度

在大多数实时系统中,调度器是比较复杂的。应用往往只能修改自己进程的调度参数而略微改变调度的行为,而调度算法本身并不能被修改。RT - Linux 则允许应用开发者自己重写其实时调度程序。在 RT - Linux V1.3 中,调度器实现了优先级调度,并支持周期任务。其实时任务控制块描述如下:

```
struct RTtask_struct
  {
    struct Rtthread tss;
    unsigned long kernel_stack;
    struct RTmm_info mm;
    int pid;                          //实时任务的任务号
    long state;                       //实时任务的状态
    int priority;                     //优先级
    Rtime resume_time;                //下次执行时间
    Rtime period;                     //周期
  }
```

其中实时任务的状态可以是以下几种:

RT_TASK_RUNNING:实时任务正在执行;

RT_TASK_STOPED:实时任务已经停止;

RT_TASK_RUNNABLE:实时任务处于就绪状态,一旦获得 CPU 的控制权即可立即执行;

RT_TASK_WAITING:实时任务被阻塞。

所有的实时任务控制块组成一个实时控制块数组,其描述如下,其中 RT_NR_TASKS 是一个常量,表示系统中所允许的最大的实时进程数目:

```
struct RTtask_struct RTtasks [RT_NR_TASKS];
char status [RT_NR_TASKS];
```

系统初始化时,这个数组的所有表项被初始化为空表项。每次实时调度器在执行时,首先将实时任务控制块数组扫描一遍,找出所有 resume_time 小于当前时间的进程,并将其状态改

为就绪。然后再次扫描实时任务控制块数组，在所有就绪实时任务中找出优先级最高的实时任务，将其投入运行。

从上述实时任务的调度分析中可以看出，RT－Linux 的实时任务控制块以及调度程度比较简单而精巧，反映了其设计者的一个主导思想，将实时应用可以分为两个域，即实时域和非实时域。实时域只对应用中与时间密切相关的部分进行处理，而大量的复杂处理则留给非实时域处理。

RT－Linux 中的有关实时任务的系统调用如下：

```
int RTload (const char * file) ;                    //装入一个实时任务，返回其 pid
int RTrun (int pid) ;                               //开始运行 pid 指定的实时任务
int RTkill (int pid) ;                              //杀死 pid 指定的实时任务
int RTset_params (Rtime * start, Rtime * period, int priority) ;
                                                    //改变实时任务的调度参数
int RTwait_start (Rtime * start, Rtime * period, int priority) ;
                                                    //挂起一个实时任务，直到其开始时间到来
int RTwait_period () ;                              //挂起一个实时任务，直到下一个周期开始
```

四、实时管道 FIFO

FIFO 是 RT－Linux 中实时任务与非实时任务通信的手段。FIFO 控制块的描述如下：

```
struct RTfifo
  {
    char * buf;                                     //队列缓冲
    char * eob;                                     //队列底
    char * head;                                    //队列头指针
    char * tail;                                    //队列尾指针
    int size;                                       //队列大小
  }
```

RT－Linux 中所有的 FIFO 形成一个数组，其中 RT_MAX_FIFO 是系统中 FIFO 的最大数目，其描述如下：

```
struct RTfifo fifoes [RT_MAX_FIFO] ;
char status [RT_MAX_FIFO] ;
```

FIFO 的空间由实时空间提供。实时任务对 FIFO 的操作是原子操作，因此不会被阻塞。而对 Linux 进程而言，FIFO 则是普通的字符设备。有关 FIFO 的实时系统调用主要有：

```
int RTfifo_create (unsigned int fifo, int size) ;   //创建一个 FIFO
int RTfifo_destroy (unsigned int fifo) ;            //销毁一个 FIFO
int RTfifo_read (unsigned int fifo, char * buf, int count) ;
                                                    //读数据，如果 FIFO 空则等待
int RTfifo_get (unsigned int fifo, char * buf, int count) ;
                                                    //读数据，如果 FIFO 空则返回 - 1
int RTfifo_write (unsigned int fifo, char * buf, int count) ;
```

```
                                        //写数据,如果 FIFO 满则等待
int RTfifo_put (unsigned int fifo, char * buf, int count);
                                        // 写数据,如果 FIFO 满则返回 - 1
```

从以上有关 FIFO 的实时调用可以看出,FIFO 不仅是 RT - Linux 中实时任务与非实时任务通信的手段,它与 UNIX 中的管道也很类似,可以作为实时任务和非实时进程的一个同步工具。

五、结束语

RT - Linux 较好地解决了硬实时问题,并且在一些领域获得了应用,我国有些企业也将其应用于实时控制场合。但是也存在一些问题,例如在实时任务之间以及实时任务与非实时进程之间的同步、互斥还比较粗糙 实时任务也不能访问 Linux 的非实时系统调用。

随着计算机应用技术的发展,如何使通用操作系统具有实时性,支持分布式实时多媒体应用,支持服务质量保证,将是操作系统研究领域的一个重要课题。

参 考 文 献

1 Barabanov M. A Linux based Real Time Operating System [D]. Master paper, New Mexico Institute of Technology, 1997, (6)

2 Barabanov M, Yodaiken V. Introducing Real Time Linux [J]. The Linux Journal, 1997, (34se)

3 Zhou Qing Guo, Huang Chun Hong Zhou Rong Jie. Remote Data Acquisition and Control System for Mössbauer Spectroscopy Based on RT - Linux [A]. Proceedings of RTSS2000 (The 21st IEEE Real Time Systems Symposium) [C]. Orlando, Florida, USA. 2000, 11

选自《计算机应用》月刊,2002 年第 12 期

3.8 基于 RT－Linux 系统的设备驱动程序开发与应用

福建泉州华侨大学先进制造技术研究所 (362011)
陈建春 吴 寒 刘雄伟

一、RT－Linux 系统简述

RT－Linux 是源代码开放的具有硬实时特性的多任务操作系统,由标准 Linux 操作系统及一个小的实时内核构成。与其他实时操作系统 (如 VxWorks、pdos) 相比,RT－Linux 具有优越的开放性、廉价性和通用性。

RT－Linux 的开发设计者为了减少工作量、保证兼容性及充分利用标准 Linux 内核的丰富资源,没有重写 Linux 内核,而是采用了一种独特的设计思想:把一个小的可抢占的实时内核植入标准 Linux 内核底层,对所有中断进行初始化处理。若中断是由实时任务产生,则交由实时内核处理,否则交由非实时内核处理。标准 Linux 核心作为实时核心的一个进程同用户的实时进程一起调度,但标准 Linux 核心进程的优先级最低,可被用户实时进程所抢占,这样用户程序的实时性完全由实时内核保证。

二、RTLinux 系统驱动程序开发

在 RT－Linux 系统中,输入输出设备被分为字符设备、块设备和网络设备 3 类。字符设备是指那些无缓冲区可以直接读写内存的设备,如串口设备等;块设备是指仅以块为单位进行读写的设备,如软盘、光盘等;网络设备主要是指那些具有网络功能接口的设备,如网卡等。

在 RT－Linux 系统中,所有设备都被看成文件,即设备文件。对设备文件进行操作就等于对设备进行操作。每个设备文件都有主设备号和从设备号。主设备号用来描述控制这个设备的驱动程序,从设备号用来区分同一个驱动程序控制的不同设备。

从用户角度看,既然每个设备都被当做文件,则同样可以用读写文件的方式对设备进行访问。设备驱动程序为用户程序操作设备提供接口函数。

设备驱动程序的本质是一组相关函数的集合,它的作用是屏蔽用户对设备进行操作的细节。虽然所有设备驱动程序之间都存在一定的差异,但也存在共同的特性。RT－Linux 系统的设备驱动程序的共性主要表现为以下几点:

⑴ 同属内核源代码,所有设备驱动程序都以内核模式运行。

⑵ 中断都由实时内核处理。

⑶ 主要功能相同,都是对设备进行初始化。把数据从内核发送到设备,从设备接收数据到内核,检测设备状态。

为了有效利用资源、增强实时性,设备驱动程序可设计为动态可加载模块。在 RT－Linux 系统中,用 insmod 模块加载,加载时自动调用驱动程序的入口函数 intinit_module (void)。该函数负责进行设备驱动程序的初始化工作,若返回“0”值,则表示初始化成功;若返回负数,则表

示初始化失败。另一个函数是 void cleanup_module (void),在模块被卸载时调用,负责清除设备驱动程序所占用的资源。需要注意的是,驱动程序以内核模式运行,直接对内存、端口等进行操作,非常容易导致死机甚至系统崩溃。所以使用系统资源时,应倍加小心。另外,用户空间程序是编译成可执行文件运行的,而实时内核空间程序则是编译成目标模块。如果要运行目标模块,就动态加载它,而且实时内核空间程序编译时必须加上 - D_KERNEL_ - DMODULE 编译选项。

下面以串口驱动程序开发为例,阐述基于 RT - Linux 系统的设备驱动程序开发的基本过程并对相关函数作简要说明。

串口驱动程序由 3 个部分组成。

(1) 初始化部分。包括中断申请、向系统注册字符设备驱动程序。这部分工作主要由驱动程序入口函数 int init_module (void) 完成。

(2) 向用户进 (线) 程提供 I/O 接口函数。由读设备函数、写设备函数等构成。

(3) 中断服务子程序。用来完成实际的读写数据过程。

RT - Linux 系统中,设备驱动程序利用实时文件操作结构 rtl_file_operations 与文件系统联系起来,为字符设备提供入口点,为用户进程提供接口函数。

```
static struct rtl_file_operations com_fops = {
NULL,
com_read,                          //读设备
com_write,                         //写设备
NULL,NULL,NULL,
com_open,                          //打开设备相关处理
com_realeased                      //释放设备相关处理
};
```

此结构体中定义了读、写、开、释放设备 4 个最重要的函数,这些函数名可自己定义。定义完结构体后,在驱动程序入口函数 int init_module (void) 中对设备进行初始化。初始化过程如下:

(1) 调用 check_region (portadr, 8) 函数,检测从 portadr (0x2F8 或 0x3F8) 起所要用的 8 个寄存器是否空闲。如果空闲,则用 request_region (portadr,8,name0) 申请这 8 个寄存器。portadr 表示用到的 I/O 端口的起始地址;name0 是设备名,驱动程序运行后,它将出现在文件 /proc/ioports 中。

(2) 确定中断号,调用 rtl_request_irq (irq,isr) 函数,注册中断号的中断处理函数。串口中断号 irq 一般为 4,isr 为中断处理函数名。

(3) 调用 rtl_hard_enable_irq (irq) 函数,使中断号有效。

(4) 调用 rtl_register_chrdev (RT_COM_MAJOR,name1,&com_fops) 函数,向系统注册字符设备驱动程序。RT_COM_MAJOR 为主设备号,串口主设备号一般为 4;name1 是设备名,驱动程序运行后,它将出现在 /proc/devices 中。

虽然 RTLinux 系统内核可用函数资源较少,但是 outb (char value, unsigned short port) 和 inb (unsigned short port) 函数就足已编写读、写串口函数。

中断服务子程序可以采用查询方式,也可以采用中断方式。如果采用查询方式,可以设置

一个周期线程，使这个线程以固定周期读或写串口。在本例中，中断服务子程序采用中断方式，这有利于提高系统资源的利用率。中断服务子程序的中断源分别是串口接收数据就绪及串口的发送缓冲器为空。因为设备在任一瞬间不可能同时进行读写二种操作，所以本例中采用了以下方法解决读写二种操作不可能同时存在的问题：中断允许寄存器的第 0 位（ERBFI 位）若为 1，则允许接收数据就绪时发出中断请求；若第 1 位（ETBEI 位）为 1，则允许发送缓冲器为空时发出中断请求。中断允许寄存器默认状态为只允许接收数据就绪时发出中断请求，即只可读数据。当要往串口写数据时，则要在中断允许寄存器写入控制字，使其第 1 位为 1、第 0 位为 0，中断识别寄存器就只识别发送缓冲器为空时引起的中断，同时通信线状态寄存器的第 5 位被置为 1。这样中断服务程序可判断通信线状态寄存器的第 5 位状态是否为 1，如是则往串口写数据。写完数据后，中断允许寄存器恢复为默认状态。

综上所述，驱动程序运行过程可归纳为：首先初始化串口；然后，若要从串口读数据，则驱动程序就先从串口读数据到内核缓冲区，再把数据从内核缓冲区送到用户空间；若要往串口写数据，则驱动程序就先把用户空间的数据送到内核缓冲区，再把数据从内核缓冲区送到串口。

驱动程序源代码完成后，可写一个 Makefile 文件，然后编译并加载驱动程序。

三、驱动程序的调用

在 RT－Linux 系统中，驱动程序可在用户空间调用，也可在实时内核空间调用。如果要在用户空间调用驱动程序，则在驱动程序成功地向系统注册后，用 mknod 命令把设备映射为一个特别文件。其他程序使用此设备时，只要对此特别文件进行操作即可。在本例中，驱动程序在实时内核空间调用。此时，不需要把设备映射为一个特别文件，只需在内核增加一个小实时内核模块。本例中驱动程序的调用需要二大模块：用户空间模块（testcom_app. c）和实时内核模块（testcom. c）。用户空间模块用于发出用户请求，处理或显示所获信息；实时内核模块用于接受用户请求，向用户输送信息，初始化串口和调用驱动程序读写串口。与在用户空间调用驱动程序相比，实时内核空间调用驱动程序中增加了一个小模块，使驱动程序调用的灵活性增强，方便了用户编程。例如可在这个小模块中确定是用端口 0 还是端口 1，还可以确定串口传输的波特率。如果要在用户空间调用驱动程序的方式中实现这些功能，一般只能通过修改驱动程序来实现。

在 RT－Linux 系统中，实时内核模块的实时任务以实时线程形式实现，用户空间模块的非实时任务以用户进（线）程实现。

实时线程之间的通信方式主要有 4 种：⑴ RTFIFO，即实时命名管道；⑵ 公用变量；⑶ 共享内存；⑷ 信号灯和互斥锁。目前，由于 RTFIFO 简单易用、稳定性高、传输数据量大，已成为最常用的通信方式。

用户进（线）程与实时线程之间的通信方式主要是 RTFIFO。本例中就是采用 RTFIFO 实现用户进（线）程与实时线程之间的通信。用户空间、实时内核空间、串口及外设之间的通信流程如图 3.8－1 所示。

下面简述该驱动程序的应用过程。

首先，在内核空间模块中创建 RTFIFO 管道，用于接收来自用户空间模块的数据或往用户空间模块发送数据。按 RTFIFO 管道所传输的数据功能进行分类，一般可分为数据管道和命令管道。数据管道用于传输程序进行运算的数据；命令管道用于传输起命令性作用的数据，如

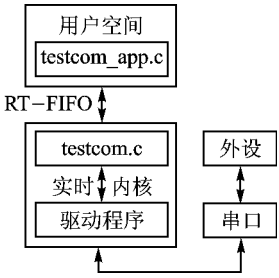


图 3.8-1 通信流程

挂起或激活一个线程或一个任务。

当用户空间模块要读串口或从内核空间获取数据信息时,可利用 read () 函数从 RTFIFO 管道中读取数据。而 RTFIFO 管道数据由内核空间程序提供。如果内核空间程序要定期获得串口数据,可通过设置线程、利用线程周期性地调用驱动程序的函数从串口读取数据,再利用 rtf_put () 函数将数据发送到 RTFIFO 管道。这种获取数据的方式就是上面所说的查询方式。

当用户空间程序要往串口或内核空间发送数据信息时,利用 write () 函数往 RTFIFO 管道写数据,内核空间程序读取 RTFIFO 管道数据,并对数据进行处理。若需要往串口写数据,可调用驱动程序函数实现。这样就实现了用户空间程序读写串口数据的目的,用户空间程序也就可通过串口与外设进行通信。

四、结束语

本文结合 RT-Linux 系统特征,阐述了开发及应用 RT-Linux 系统设备驱动程序的过程。虽然目前 RT-Linux 内核函数资源较少,但足以开发一个完整的设备驱动程序。作者曾用上述中断方式的驱动程序及查询方式的驱动程序分别与 PLC 通信。结果表明,使用中断方式的驱动程序接收数据更加准确,运行更加稳定。

参 考 文 献

1 陈莉君. Linux 操作系统内核分析. 北京:人民邮电出版社,2000

选自《微型机与应用》月刊,2002 年第 9 期

3.9 嵌入式实时操作系统 μC/OS－Ⅱ及其应用

西北工业大学 司利云 王铁勇 吴盘龙

早在20世纪60年代,就已经有人开始研究和开发嵌入式操作系统。但直到最近,它才在国内被越来越多的提及。其在通信、电子、自动化等需要实时处理的领域所日益显现的重要性吸引了人们越来越多的注意力。针对国内大部分用户使用的51系列的8位处理器,我们可以选择 μC/OS－Ⅱ。

μC/OS－Ⅱ是由 Labrosse 先生编写的一个开放式内核。其主要特点是源码公开的自由软件。这一点对于用户来说可谓利弊各半 好处在于,一方面它是免费的,另一方面用户可以根据自己的需要对它进行修改。坏处在于,它缺乏必要的支持。它没有功能强大的软件包,用户通常得自己编写驱动程序,特别当用户使用的是不太常用的单片机,还必须编写移植程序。

一、μC/OS－Ⅱ特点

⑴ μC/OS－Ⅱ是一个占先式的内核,即已经准备就绪的高优先级任务可以剥夺正在运行的低优先级任务的 CPU 使用权。这个特点使得它的实时性比非占先式的内核要好。通常我们都是在中断服务程序中使高优先级任务进入就绪态(例如发信号),这样退出中断服务程序后,将进行任务切换,高优先级任务将被执行。但是因为我们无法确定发生中断时程序到底执行到了什么地方,我们也就无法判断要经过多长时间数据处理程序才会执行,中断响应时间无法确定,系统的实时性不强。如果使用 μC/OS－Ⅱ的话,我们只要把数据处理程序的优先级设定的高一些,并在中断服务程序中使它进入就绪态,中断结束后数据处理程序就会被立即执行。这样我们可以把中断响应时间限制在一定的范围内。对于一些对中断响应时间有严格要求的系统,这是必不可少的。

⑵ μC/OS－Ⅱ和我们所知道的 Linux 等分时操作系统不同,不支持时间片轮转法。它是一个基于优先级的实时操作系统。每一个任务的优先级必须不同(分析它的源码会发现,μC/OS－Ⅱ把任务的优先级当作任务的标识来使用,如果优先级相同,任务将无法区分)。进入就绪态的优先级最高的任务首先得到 CPU 的使用权,只有等它交出 CPU 的使用权后,其他任务才可以被执行。所以它只能说是多任务,不能说是多进程,至少不是我们所熟悉的那种多进程。μC/OS－Ⅱ的这种特性是好是坏,主要看从什么角度来判断。显而易见,如果只考虑实时性,当然比分时系统好,它可以保证重要任务总是优先占有 CPU。但是在系统中,重要任务毕竟是有限的,这就使得划分其他任务的优先权变成了一个让人费神的问题。另外,有些任务交替执行反而对用户更有利。例如,用单片机控制两小块显示屏时,无论是编程者还是使用者肯定希望它们同时工作,而不是显示完一块显示屏的信息以后再显示另一块显示屏的信息。

⑶ μC/OS－Ⅱ对共享资源提供了保护机制。它是一个支持多任务的操作系统。我们可以把一个完整的程序划分成几个任务,不同的任务执行不同的功能。对于共享资源(比如串口),μC/OS－Ⅱ也提供了很好的解决办法,一般情况下使用的是信号量方法。我们创建一个

信号量并对它进行初始化,当一个任务需要使用一个共享资源时,它必须先申请得到这个信号量。在这个过程中即使有优先权更高的任务进入了就绪态,因为无法得到信号量,也不能使用该资源。在 μC/OS－Ⅱ中称为优先级反转。简单的说,就是高优先级任务必须等待低优先级任务的完成。在上述情况下,在两个任务之间发生优先级反转是无法避免的。所以在使用 μC/OS－Ⅱ时,必须对所开发的系统了解清楚才能选择对于某种共享资源是否使用信号量。

二、μC/OS－Ⅱ在单片机中的应用

⑴ 在单片机系统中嵌入 μC/OS－Ⅱ将增强系统的可靠性,并使得调试程序变得简单起来。我们经常遇到编完程序时,在调试过程中要不是程序跑飞了,要不就是陷入死循环。如果在系统中嵌入 μC/OS－Ⅱ,我们可以把整个程序分成许多任务,每个任务相对独立。然后在每个任务中设置超时函数,时间用完以后,任务必须交出 CPU 的使用权。即使一个任务发生问题,也不会影响其他任务的运行。这样既提高了系统的可靠性,同时也使得调试程序变得容易。需要指出的是,这里所说的容易是建立在开发人员对 μC/OS－Ⅱ有所了解并有实际操作经验的基础上的。

⑵ 在单片机系统中嵌入 μC/OS－Ⅱ将增加系统的开销,这在许多书籍和资料中都提到过。现在我们所使用的 51 系列单片机,其片内都带有一定的 RAM 和 ROM。对于一些简单的程序,如果采用传统的编程方法,已经不需要外扩存储器了。如果在其中嵌入 μC/OS－Ⅱ的话,在只需要使用任务调度、任务切换、信号量处理、延时或超时服务的情况下,也不需要外扩 ROM 了,但是外扩 RAM 是必须的。由于 UC/OS－Ⅱ是可裁减的操作系统,其所需要的 RAM 大小就依赖于我们对操作系统一些功能的选择。嵌入 μC/OS－Ⅱ以后总的 RAM 需求可以由如下表达式得出:

RAM 总需求 = 应用程序的 RAM 需求 + 内核数据区的 RAM 需求 +
(任务栈需求 + 最大中断嵌套栈需求) × 任务数

所幸的是 μC/OS－Ⅱ可以对每个任务分别定义堆栈空间的大小,我们可根据任务的实际需求来进行栈空间的分配。但不管怎么说,在 RAM 容量有限的情况下,我们还是应该注意一下对大型数组、数据结构和函数的使用,别忘了,函数的形参也是要推入堆栈的。

⑶ μC/OS－Ⅱ的移植也是一件需要值得注意的工作。如果我们手中没有现成的移植实例的话,我们就必须自己来编写移植代码。虽然只需要改动两个文件,但仍需要对相应的微处理器比较熟悉才行。最好参照已有的移植实例。另外,即使我们有移植实例,在编程前最好也要阅读一下,因为里面牵扯到堆栈操作。我们在编写中断服务程序时,把寄存器推入堆栈的顺序必须与移植代码中的顺序相对应。

⑷ 和其他一些著名的嵌入式操作系统不同,μC/OS－Ⅱ在单片机系统中的启动过程比较简单。μC/OS－Ⅱ的内核是和应用程序放在一起编译成一个文件的,我们只需要把这个文件转换成 HEX 格式,写入 ROM 中就可以了。上电后,它会像普通的单片机程序一样运行。

从以上的分析中我们不难看出,是否在单片机系统中嵌入 μC/OS－Ⅱ取决于使用者所要开发的项目。对于实时性,可靠性要求较强的项目,特别是大型项目,最好使用 μC/OS－Ⅱ;而对于一些简单的,成本要求低的项目,就没必要这么麻烦。

选自《电子产品世界》月刊,2002 年第 8 期

3.10 在 MOTOROLA 568XX 系列 DSP 上运行 μC/OS-Ⅱ

清华大学 薛 涛 龚光华 邵贝贝

一、μC/OS-Ⅱ与 MOTOROLA DSP 型单片机

本刊《学习园地》栏目曾分6期介绍了嵌入式实时操作系统 μC/OS-Ⅱ。该实时内核的源码以光盘的形式附在《μC/OS-Ⅱ——源码公开的实时嵌入式操作系统》一书中。这本书是美国人 Jean J. Labrosse 写的《MicroC/OS-Ⅱ The Real-Time Kernel》一书的中译本。μC/OS-Ⅱ已被移植到几乎所有能运行实时操作系统的单片机上,并用到了许许多多应用领域。μC/OS-Ⅱ也能方便地移植到数字信号处理器 DSP 上。由于 μC/OS-Ⅱ的商业价值得到越来越多的认可,故不再像 μC/OS 那样是完全免费的。目前,最新版本的 μC/OS-Ⅱ是 V2.51,在网上的标价是 55.95 美元;而上面提到的书以及相应的中译本中附的源码是 V2.08。由于该书中译本的出版,V2.08 的 μC/OS-Ⅱ的源码已在我国广泛流传。

MOTOROLA 的 DSP 虽然不如 TI 公司的 DSP 那么有名,但 MOTOROLA 的 DSP 有其不同于 TI 之 DSP 的独到之处。MOTOROLA 的 DSP 集成了更加丰富的 I/O 模块,强调片上系统(SOC)和单片化,从而更像 MOTOROLA 的单片机,只不过是将 CPU 换成 DSP。对于工业控制中运算量很大的应用,完全可以选用 MOTOROLA 的 DSP 来替代单片机。在这种意义上,完全可以把 MOTOROLA 的 DSP 看成运算速度特别快的单片机。在单片机开发中推荐使用 RTOS,在 MOTOROLA 的 DSP 上运行 RTOS 也就不足为怪了。由于 DSP 内核结构比单片机复杂得多,有的汇编指令可以同时并行地干两件甚至三件事,使得 DSP 的应用程序开发有一定的难度。好在 MOTOROLA 在推出 568XX 系列 DSP 的同时,还推出了两个强有力的开发工具,使得应用开发工程师还不大了解 DSP 内部结构时,就已经能上手开发应用程序。这两个重要的开发工具是:①Codewarrior 的 C 交叉编译器以及集成调试环境 IDE;②Embedded DSP SDK 软件包。

这两个工具对于开发 MOTOROLA DSP 应用的工程师来说,都不难得到。C 语言的交叉编译器是实现 RTOS 的基本工具。SDK 中除了各种 I/O 模块的驱动程序库之外,还提供了将最新版本 μC/OS-Ⅱ V2.51 移植到 DSP568XX 上去的源代码。遗憾的是,由于 μC/OS-Ⅱ的 2.51 版本要从国外购买,多数国内开发人员,手头上只有 μC/OS-Ⅱ的 2.08 版本。本文的目的在于,明示读者如何参照 Embedded DSP SDK V2.4 中给出的 DSP 移植源码,实现μC/OS-Ⅱ V2.08 在 DSP568XX 上的运行。

二、向 DSP56800 上移植 μC/OS-Ⅱ

μC/OS-Ⅱ的文件结构如图 3.10-1 所示。

移植 μC/OS-Ⅱ只要修改 3 个文件:

OS_CPU.H

OS_CPU_A.ASM

图 3.10－1 μC/OS－Ⅱ **的文件结构**

OS_CPU_C.C

OS_CPU.H **定义了关开中断的两个函数** OS_ENTER_CRITICAL() **和** OS_EXIT_CRITICAL()**。**

OS_CPU_A.ASM **完成任务栈的初始化,我们在任务栈结构与** DSP **中断机制一节中描述。**

OS_CPU_C.C **定义定时器相关函数** timerSleep (Word32 Ticks) **以及相应的接口空函数。这个文件在** Embedded DSP SDK V2.4 **中已经给出,无需修改。**

三、μC/OS－Ⅱ V2.51 新功能

比较新老版本的 μC/OS－Ⅱ V2.51 **首先增加了两个新的功能:一个是事件标志** Event_Flag() **,另一个是函数** mutex()**。** Event_Flag **对于应用程序中会用到先对几个不同事件条件做"与"、"或"操作,然后再实现任务调度的情况。函数** mutex() **在处理互斥条件相关的应用中更为方便。这两个函数在** 2.08 **版本的源码中没有。如果用户手头上没有最新版本的** μC/OS－Ⅱ**,仍可以利用** Embedded DSP SDK V2.4 **中提供的移植代码运行** μC/OS－Ⅱ V2.08**,办法是在相关移植文件中删除这两个函数。**

新老版本 μC/OS－Ⅱ **的另一个区别是,在处理关、开中断的方法上,**V2.08 **只定义了两种方法:方法** 1 **是用关中断、开中断指令定义进入和退出临界区函数;方法** 2 **用进入临界区前将当前中断的开关状态入栈的办法来处理这两个函数。这种方法在离开临界区以后要对栈指针做调整,而这种调整有时与所用的** C **编译器有关。**2001 **年第** 12 **期《学习园地》第** 6 **讲对这一方法可能带来的问题作了介绍。**

在新版本的 μC/OS－Ⅱ V2.51 **中,移植者** Lee Silverthorn **用了第三种方法。采用定义局部变量的方法来临时保存进入临界段代码之前中断的开关状态,离开临界区后恢复原来的状态。如果在临界段中发生任务切换,当离开定义该局部变量的那个函数时,该局部变量自动消失,不会引起栈指针的错乱。这种做法虽然比较稳妥,但对于每个需要调用临界段代码保护函数的函数,都要定义一个局部变量** CPU_SR**。** Embedded DSP SDK V2.4 **中使用方法** 3**。**

我们认为,DSP568XX **中断机制并不复杂,虽然中断有不同的优先级别,但并不像** MC68K **系列** CPU **那样允许开关某一级的中断,在** DSP568XX **上运行** μC/OS－Ⅱ **仍可以使用方法** 1**,即在文件** OS_ CPU.H **中定义方法** 1 **以后** μC/OS－Ⅱ **可以正常运行。定义方法如下:**

```
#define  OS_CRITICAL_METHOD      1
#if      OS_CRITICAL_METHOD  = = 1
#define  OS_ENTER_CRITICAL ()    {h
  asm (bfset   #0x0300,SR) ;asm (nop) }
#define  OS_EXIT_CRITICAL () {h
  asm (bfclr   #0x0200,SR) ;asm (nop) }
#endif
```

四、任务栈结构与 DSP 中断机制

由于 56800 是一个相当复杂的 DSP,它的算术逻辑单元 ALU 中有 9 个寄存器,地址产生单元有 7 个寄存器,程序控制单元中有各类寄存器 6 个。每个任务控制块中至少要有这 22 个寄存器的影像。在移植文件 OS_CPU_A. ASM 中,要给出每个任务的栈结构。此外,在 Codewarrior 的 568XXDSP 的 C 编译中,编译器把 0X30 到 0X3F 这 16 个 RAM 字(word)当作寄存器来用,任务切换时,总共入栈、出栈的寄存器数目相当于各有 38 个,如图 3.10－2 所示(注意,DSP568XX 的堆栈指针是递增的)。

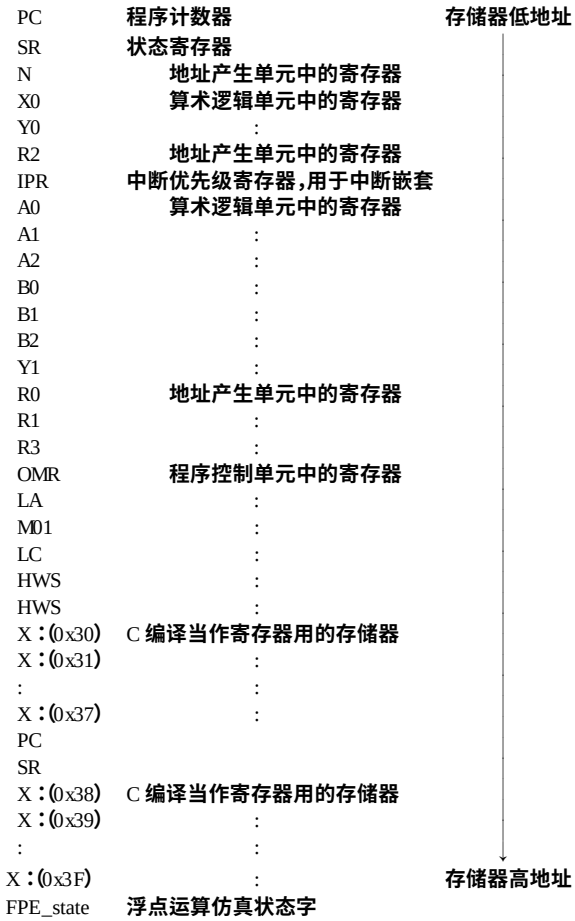


图 3.10－2 任务的栈结构

DSP 的中断机制类似 MOTOROLA 32 位单片机的 MC68K 中断机制，即在中断响应时，只保护当前程序计数器 PC 和条件码寄存器 CCR，不保护其他任何寄存器。这就要求用户在中断服务子程序中用到哪些寄存器就得保护哪些寄存器，以求最快的运算速度。

在 μC/OS－Ⅱ中，任务或事件可以引起任务切换，中断也可能引起任务切换。凡引起任务调度与切换的中断，必须保护上述全部 38 个寄存器，而不引起任务切换的中断则可根据 ISR 中用到的寄存器作局部保护。显然，这 38 个寄存器的入栈出栈是要花时间的。为了发挥 DSP 寄存器多的优势，Codewarrior 的 C 编译器在函数调用的参数传递上，采用了用寄存器传递参数而不使用传统的用堆栈传递参数的方法。这在一定程度上减小了栈操作的负荷，提高了速度。以下是用汇编定义的、可以在 C 中调用的任务栈初始化函数，见 OS_CPU_A. ASM。

asm OS_STK ＊OSTaskStkInit (void (＊task)(void ＊pd), void ＊pdata, OS_STK ＊ptos, INT16U opt)

调用此函数输入参数的格式是：

Y0	存放	opt
R2	指向	pd
R3	指向	pdata
x:(SP - 2)	指向	ptos

任务栈初始化函数：

```
FOSTaskStkInit:
    move   x:(SP - 2),R1                ;将 ptos 存入堆栈
    nop
    lea (R1) +
    move R2,X:(R1) +                    ;返回地址
    move # $ 0100,Y1
    move Y1,x:(R1) +                    ;建立新的 SR
    clr X0                              ;开中断
    rep #3
    move X0,x:(R1) +                    ;清零三个寄存器:N, X0, Y0
    move r3,x:(R1) +                    ;将 R2 指向 pdata
if SUPPORT_NESTED_INTERRUPTS = = 1
    move IPR,Y0                         ;保存 IPR
    move   Y0,X:(R1) +
endif
    rep #9
    move X0,x:(R1) +                    ;清零九个寄存器:
;A0, A1, A2, B0, B1, B2, Y1, R0, R1
    move r3,x:(R1) +                    ;设置 R3
    move omr,x:(R1) +                   ;设置 omr
    move   la,x:(R1) +                  ;设置 la
    move   m01,x:(R1) +                 ;设置 M01
    move   lc,x:(R1) +                  ;设置 lc
    move   X0,x:(R1) +                  ;保存模拟的硬件栈
```

```
move   X0,x:(R1)+                    ;
rep    #8                            ;保存 C 编译使用的临时寄存
                                     ;器地址在 0x30~0x37
move   X0,x:(R1)+                    ;
move   #DispatchRestore,R2           ;添加 jsr 语句,返回到 DispatchRestore
move R2,x:(R1)+                      ;
move Y1,x:(R1)+
rep #8                               ;保存 C 编译器使用的临时变
                                     ;量,地址在 0x38~0x3F
move X0,x:(R1)+                      ;
move X0,x:(R1)                       ;保存 FPE_state
move R1,R2                           ;返回((OS_STK *)stk);
rts
```

五、关于在 DSP 上运行 RTOS 的讨论

由于超大规模集成电路技术的发展,DSP 的成本逐年降低,DSP 正在替代单片机中的部分 CPU,组成以 DSP 为核心的单片机,特别是在 32 位单片机上 DSP 的影响越来越不可忽视。开发 DSP 应用的工程师,仍可以用单片机(MCU)的开发思路来开发 DSP 应用产品。以 32 位 MOTOROLA MCU CPU32 为例,典型的内部时钟频率是 25 MHz,此时的运算速度大约为 4.7 MIPS,而 568XX 系列 DSP 典型的内部时钟频率为 80 MHz,此时的运算速度大约为 40 MIPS,大约要快 1 个数量级。可以按照开发 MCU 的思路先运行 1 个 RTOS,如 μC/OS-Ⅱ,然后再开发应用程序。由于 DSP 较 CPU 复杂,任务切换时入栈出栈的寄存器比 CPU 多很多,虽然 Codewarrior 的 C 编译器在函数调用的参数传递上采用了用寄存器传递参数的方法,在一定程度上加快了程序的执行速度,但 C 编译把 16 个字的存储器当作寄存器用,如使用 RTOS 的话,入栈操作会使中断延时时间加长,相应的中断响应时间与 CPU,如 CPU32,大体在同一数量级上,并没有什么优势。将上述 22 个寄存器和 16 个存储器字入栈约需要 280 个指令周期。对于 40 MHz 的指令周期,要 7 μs 左右,这对于响应时间为数百 μs 或 ms 级的系统是没有问题的,但对于那些要求 1 μs、1 μs 地计算实时响应的控制系统,笔者认为要慎用 RTOS,即可以不使用 RTOS,在 SDK 中选用 No RTOS。这样做才能发挥 DSP 快速的优势。以 DSP 的思路开发单片机应用产品的做法,对于提高系统的响应速度是有利的。总之,在 DSP 上移植和运行 μC/OS-Ⅱ并不困难,而用还是不用 RTOS,完全取决于具体应用。

参 考 文 献

1 Labrosse Jean J. μC/OS-Ⅱ——源码公开的实时嵌入式操作系统. 邵贝贝译. 北京:中国电力出版社,2001

2 MOTOROLA Embedded SDK V2.4

选自《单片机与嵌入式系统应用》月刊,2002 年第 4 期

3.11 Franklin C51 浮点数与 A51 浮点数的相互转换、传递及其在混合编程中的应用

中央民族大学物理系 (100081) 马 岩
北京机械工业学院 (100085) 马力妮

一、引 言

Franklin C51 是用于开发 8051 系列单片机系统的工具软件。它允许用户使用不同的语言进行混合编程,其中使用汇编语言 A51 与 C51 进行混合编程是一种比较理想的技术组合,既有利于发挥 C51 强大的计算能力,又能发挥汇编语言 A51 对硬件极强的控制能力。然而,在实际应用中,这种混合编程技术组合却遇到了一些麻烦和困难,最主要的有以下三个方面。

⑴ FC51 与 A51 变量在 8051 内部资源分配上有冲突;

⑵ FC51 与 A51 函数的相互调用及参数的相互传递问题;

⑶ FC51 浮点数与 A51 浮点数之间数据格式不兼容问题。

对于前两点,已有大量文章论及并给出许多很好的解决方案,本文主要结合一个应用实例讨论第三方面的问题。

二、从实例看混合编程对 FC51 浮点数与 A51 浮点数间相互转换、传递的需求

在某型智能化数字流量计的研发过程中,有这样一个棘手问题:该流量计的样机监控程序完全采用 A51、按 FD—ASM51 浮点格式编制。主要分组态、计算、采集显示三大部分。在后来改型中,需要优化算法,系数精度要求提高到七位半。这意味着,至少计算部分不能再用 A51、按 FD—ASM51 浮点格式编程。如果推倒重来,全部改用 FC51 重写,除了大大延长开发周期外,最大的困难还在于:尽管用 FC51 编程,便于进行大规模复杂的、高精度、宽数域的浮点运算,可以很好解决流量计的计算模块高精度、宽数域的要求,但该流量计的组态模块要求通过若干按键、以浮点格式,实时完成对温度、压力、流量等量程参数的实时装订输入、切换,还要求其增、减偏移量键实现实时滚动式累进、显示(当按住该键不动时,输入偏移量要自动按浮点格式高速累加、累减,并实时滚动显示,偏移量有时只有 0.001),如果用 FC51 编程难度很大,也无完全把握。为了缩短开发周期,更重要的是为了能充分发挥 FC51 浮点数和 A51 浮点数特长,优势互补,提高开发的成功率,最佳方案应是:只用 FC51 编写优化后的计算模块,保留原有的、用 ASM51 汇编语言编写的组态、采集和显示模块,再通过混合编程技术实现升级。很显然要实现此方案,关键是要实现 FC51 浮点数与 A51 浮点数的相互转换并能在混合编程中实现相互传递。这样,FC51 浮点数与 A51 浮点数之间数据格式不兼容问题在这个混合编程的实例中凸现出来。

三、FC51 与 A51 做浮点数格式之间的相互转换

为解决两种浮点数之间数据格式不兼容这个关键技术问题，首先必须弄清这两种浮点数格式上的差异。浮点数的数学表达式为：$a * E^b$。其中 E 为底，a 为尾数，b 为阶码。E 通常取值为 2，b 长度一般为一字节。此时最大阶码为 127，浮点的精度由尾数的字长所决定，若尾数为 2 字节，可保证四位半有效数字，若尾数为 3 字节则可保证七位半有效数字。在实际运用中，不同浮点数格式各有差异。

1. A51 浮点数格式

A51 浮点数即 FD－ASM51 浮点数，它有三字节浮点数和四字节浮点数两种。以四字节浮点数为例（三字节浮点数仅比四字节浮点数少一字节尾数，其他格式完全相同），其格式如下：

地址＋0：S E E E E E E E　数符和阶码
地址＋1：MMMMMMMM　尾数高字节
地址＋2：MMMMMMMM　尾数中字节
地址＋3：MMMMMMMM　尾数低字节

其中　S——浮点数符号位；
　　0——表示正；
　　1——表示负；
　　E——浮点阶码，以补码表示，其中左起第一位为阶符；
　　M——浮点尾码，以原码表示。

2. FC51 浮点数格式

FC51 浮点数格式使用的是 IEEE－754 的标准，具有 24 位尾数精度，尾数的高位始终为“1”，因而不保存，其具体格式如下：

地址＋0：MMMMMMMM　尾数低字节
地址＋1：MMMMMMMM　尾数中字节
地址＋2：E MMMMMMM　尾数高字节
地址＋3：S E E E E E E E　数符和阶码

其中　S——浮点数符号位；
　　0——表示正；
　　1——表示负；
　　E——浮点阶码，共 8 位（分布在两字节中），偏移量为 127，左起第一位为阶符；
　　M——浮点尾码，共 24 位，以原码表示，最高位始终为“1”，缺省。

在分析清楚 FC51 浮点数和 A51 浮点数的格式结构上的差异后，可以利用 A51 汇编语言强大的逻辑运算和位操作能力，编写两种浮点数格式之间的相互转换子程序，架起 A51 与 FC51 之间浮点数相互转换、相互传递的桥梁，彻底消除由于两种浮点数格式不兼容对混合编程所造成的障碍。以下是两个用于这种转换的子程序并给出具体数据调试结果。

程序 1：IE－FD 子程序
功能：将 FC51 浮点数转换成 A51 浮点数
入　口：
R6：C51 浮点数符和阶码。

R2：C51 浮点尾数高字节。

R3：C51 浮点尾数中字节。

R4：C51 浮点数尾数低字节。

出 口：

R6：A51 浮点数符和阶码。

R2：A51 浮点尾数高位字节。

R3：A51 浮点尾数中位字节。

R4：A51 浮点尾数低位字节。

```
IEFD：  MOV     A,R6
        ORL     A,R2
        ORL     A,R3
        ORL     A,R4
        JZ      IEFD4
        MOV     B,R6
        MOV     A,R2
        MOV     C,ACC.7
        SETB    ACC.7
        MOV     R2,A
        MOV     A,R6
        RLC     A
        MOV     R6,A
        CJNE    R6,#7FH,IEFD0
IEFD0：JNC     IEFD1
        ADD     A,#02H
        ANL     A,#7FH
        LJMP    IEFD2
IEFD1：MOV     A,R6
        SUBB    A,#7EH
        MOV     R6,A
IEFD2：JNB     B.7,IEFD3
        SETB    ACC.7
        SJMP    IEFD4
IEFD3：CLR     ACC.7
IEFD4：MOV     R6,A
        RET
```

调试数据：

定点数	C51 浮点	A51 浮点
$+5.4210108 * 10^{-20}$	00H	41H
$(+1 * 2^{-64})$	00H	80H
	80H	00H
	1FH	00H
$-5.4210108 * 10^{-20}$	00H	C1H
$(-1 * 2^{-64})$	00H	80H
	80H	00H
	9FH	00H
$+9.22337 * 10^{+18}$	FFH	3FH
$(+1 * 2^{+63})$	FFH	FFH
	FFH	FFH
	5EH	FFH
$-9.22337 * 10^{+18}$	FFH	BFH
$(-1 * 2^{+63})$	FFH	FFH
	FFH	FFH
	DEH	FFH
	(IN)	(OUT)

程序 2：FD－IE 子程序

功能：将 A51 浮点数转换成 FC51 浮点数。

入　口：

R6：A51 浮点数数符和阶码。

R2：A51 浮点数尾数高位字节。

R3：A51 浮点数尾数中位字节。

R4：A51 浮点数尾数低位字节。

出　口：

R6：FC51 浮点数数符和阶码。

R2：FC51 浮点数尾数高位字节。

R3：FC51 浮点数尾数中位字节。

R4：FC51 浮点数尾数低位字节。

```
FDIE：  MOV     A,R6
        ORL     A,R2
        ORL     A,R3
        ORL     A,R4
        JZ      FDIE6
        MOV     B,R6
        MOV     A,R6
        CLR     ACC.7
        CJNE    A,#40H,FDIE0
FDIE0：JNC     FDIE1
        ADD     A,#7EH
        LJMP    FDIE2
FDIE1：CLR     C
```

```
        SUBB    A,#2H
FDIE2：CLR     C
        RRC     A
        MOV     R6,A
        MOV     A,R2
        JC      FDIE3
        CLR     ACC.7
        SJMP    FDIE4
FDIE3：SETB    ACC.7
FDIE4：MOV     R2,A
        MOV     A,R6
        JNB     B.7,FDIE5
        SETB    ACC.7
        SJMP    FDIE6
FDIE5：CLR     ACC.7
FDIE6：MOV     R6,A
        RET
```

调试数据：

定点数	C51 浮点	A51 浮点
$+5.4210108*10^{-20}$	41H	00H
$(+1*2^{-64})$	80H	00H
	00H	80H
	00H	1FH
$-5.4210108*10^{-20}$	C1H	00H
$(-1*2^{-64})$	80H	00H
	00H	80H
	00H	9FH
$+9.22337*10^{+18}$	3FH	FFH
$(+1*2^{+63})$	FFH	FFH
	FFH	FFH
	FFH	5EH
$-9.22337*10^{+18}$	BFH	FFH
$(-1*2^{+63})$	FFH	FFH
	FFH	FFH
	FFH	DEH
	(IN)	(OUT)

因最小阶码长度为 7 位（补码），故转换程序适用的数域和精度为：

最小浮点数：$-5.4210108*10^{-20} \sim +5.4210108*10^{-2}$（$-2^{-64} \sim +2^{-64}$）；

最大浮点数：$-9.22337*10^{18} \sim +9.22337*10^{18}$（$-2^{+63} \sim +2^{+63}$）。

四、在混合编程中如何实现 FC51 浮点数与 A51 浮点数之间的相互传递

首先，在 FC51 源程序中声明一个指向 8051 片内 RAM 区的浮点数据指针：float idata f；**同时在 A51 汇编程序中，在 8051 片内 RAM 区开辟一个 4 字节的缓冲区，如** 70H、71H、72H、

73H。

如果想把 FC51 源程序中的一个浮点数变量 q 传递到 A51 汇编程序中,并转换成 A51 浮点数,可采用以下步骤:

首先在 FC51 源程序中做如下编程:

```
f = 0x70;                    /* 将浮点指针指向 8051 片内缓冲区首址 */
*f = q;                      /* 将 FC51 浮点传递到片内缓冲区 */
```

随后在 A51 汇编程序中,按格式将 FC51 浮点数从首址为 70H 的 8051 片内缓冲区中取出并送入 IE-FD 子程序的入口 R6、R2、R3、R4 中,在调用 IE-FD 子程序后,就能在其出口 R6、R2、R3、R4 中得到转换后的 A51 浮点数。然后用户就可以调用 FD-ASM51 浮点运算子程序库,在 A51 汇编语言程序中进行所需的浮点运算了。

如果想把一个 A51 汇编程序的浮点数转换成 FC51 浮点数并传递到 FC51 源程序中的浮点变量 q 中,可采用以下步骤:

首先在 A51 汇编程序中将该浮点数按格式送入 FD-IE 子程序的入口 R6、R2、R3、R4 中,调用 FD-IE 子程序,则在其出口 R6、R2、R3、R4 中就能得到转换后的 FC51 浮点数,将此数按 FC51 浮点格式送到首址为 70H 的 8051 片内 RAM 缓冲区中,然后在 FC51 源程序中做如下编程:

```
f = 0x70;                    /* 将浮点指针指向 8051 片内缓冲区首址 */
q = *f;                      /* 将 FC51 浮点数传递到浮点数变量 q 中 */
```

即告完成。需要特别注意的是,在做这种转换前,A51 浮点数格式必须归格化。

五、结束语

在科学技术飞速发展的今天,在以 MCS-51 为代表的 8 位单片机的应用领域中,我们不但要对传统的实时性发出最新挑战,更要去面对运算规模和运算精度方面不断提出的更高要求。C51 抑或 A51 是我们每天都要面临的选择。如果仅仅选择 C51,从某种意义上来说,意味着在获得较高的运算功能和运算精度的同时却削弱了实时性。而如果仅仅选择 A51,则会得到恰恰相反的效果。由此看来,或许采取两条腿走路的方针,优势互补,即采用混合编程的方法不失为一种明智的选择。但如果不能彻底解决 FC51 和 A51 浮点数格式不兼容的问题,以至于在混合编中不能使这两种互不兼容的浮点数互相转换、融汇贯通,那么就会使我们的选择弄巧成拙。若果真如此,这种混合编程技术也就不会成为一种成熟的、完备的编程技术。正是基于这种理念,奉上此文,希望对您的工作有所帮助。

参 考 文 献

1 张友德,涂时亮. MCS-51 单片机实用子程序及其应用 [M]. 上海:复旦大学出版社,1988

2 周航慈. 单片机应用程序设计技术 [M]. 北京:北京航空航天大学出版社,1991

3 马忠梅,马岩等. 单片机的 C 语言应用程序设计 [M]. 北京:北京航空航天大学出版社,1997

选自《电测与仪表》月刊,2002 年第 8 期

第四章

网络、通信与数据传输

4.1 嵌入式系统以太网接口的设计

杭州浙江机电职业技术学院 (310012) 葛永明
杭州中程兴达智能信息技术公司 (310023) 林继宝

目前,以太网 (Ethernet) 协议已经非常广泛地应用于各种计算机网络,如办公局域网、工业控制网络、因特网等场合,并且还在不断地发展。基于以太网的新技术和联网设备不断出现,以太网已经成为事实上最常用的网络标准之一。

但是,基于以太网的嵌入式系统目前并不是很多。其原因除了嵌入式系统本身运行速度较慢、资源较少且不足以实现以太网的各种协议外,更重要是设计以太网的接口及协议相对比较复杂,使人望而却步。

本文将介绍以 8051 系列单片机系统为例的嵌入式系统与 10 Mbps 以太网控制器芯片 DM9008 的接口电路实现及编程方法。

一、以太网控制器 DM9008 简介

DM9008 是台湾 DAVICOM 公司生产的基于 ISA 总线的 10 M 超级以太网控制器芯片。它集成了介质访问控制子层 (MAC) 和物理层的功能,可以方便地设计基于 ISA 总线的系统,也可以比较简单地与通用单片机进行接口。

主要特点如下:

- 实现 IEEE 802.3 协议、10BASE - T、10BASE2 和 10BASE5 的单芯片解决方案;
- 集成 ISA 总线接口、8K × 16 SRAM、介质访问控制 (MAC)、编解码器 (ENDEC) 和 10BASE - T 收发器;
- 与 NOVELL NE2000 软件兼容;
- 可选 8 根中断申请线;
- 自动极性检测和纠正;
- 可选 8、16 位模式;
- 外部可编程 EEPROM;
- 单 5 V 电源低功耗 CMOS 设计;
- 100 脚 PQFP 封装。

由于该芯片功能较强,配置有较多的引脚,但在与一般单片机接口时只需要用到其中的一部分即可完成常用的功能。

二、与 8051 单片机系统的接口电路

下面介绍国内最常用的 8051 系列单片机与 DM9008 的接口电路,实现的网络接口采用无屏蔽双绞线 (UTP) RJ - 45 接口。

图 4.1 - 1 给出了 8051 单片机系统与 DM9008 网络控制器的接口电路框图。8051 单片机

系统所提供的接口信号线为：P0 口的 8 位数据总线 D0 ~ D7、5 根经过锁存的地址线 A0 ~ A4、读信号线 RD、写信号线 WR、经过译码产生的片选线 CS1 和经过反相后高电平有效地中断请求线 INT。这些信号线分别与 DM9008 的数据线低 8 位 SD0 ~ SD7、地址线低 5 位 SA0 ~ SA4、I/O 读信号线 IOR、I/O 写信号线 IOW、地址使能线 AEN 和 8 根中断请求线中的一根 IRQ12 相连。

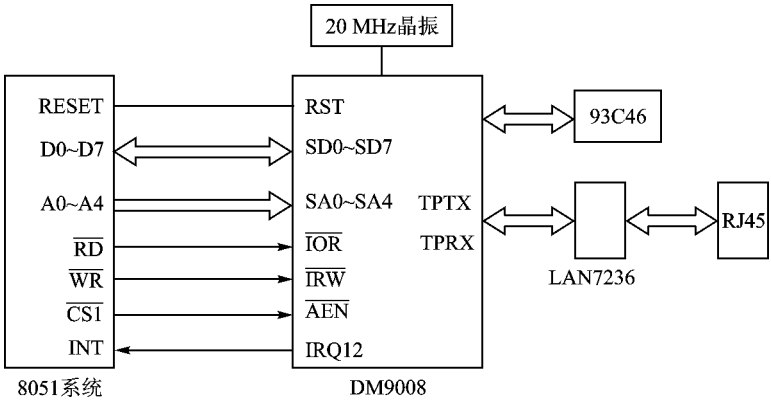


图 4.1－1 8051 单片机系统与 DM9008 网络控制器的接口电路框图

DM9008 的复位线 RST 与 8051 单片机的复位线同为高电平有效，故在系统上电时同时复位。时钟电路只需外接 1 个 20 MHz 的晶振及 2 个电容即可。

DM9008 有 16 根数据线，与 8051 单片机接口，只需用到低 8 位数据线，其他高 8 位数据线不用，IO16 接高电平或浮空，MD6/SLOT 接地。地址线有 SA0 ~ SA19 共 20 根，只用到低 5 位 SA0 ~ SA4 与单片机相连，SA5 ~ SA7 接地、SA8 ~ SA9 接高电平，其他高地址位全部接地。这样对于 DM9008 内部来说，I/O 的基地址为 300H。另外 BALE、SYSCLK 接地，SMEMR、MEMW、MEMR 浮空。DM9008 有 8 根中断请求线，可以选 1 根与 8051 系统的中断线相连，其他 7 根线均浮空，所选线在 EEPROM93C46 中指定。

EEPROM 93C46 是一个采用 4 线 SPI 串行接口的电可擦除存储器，容量为 64 ×16 位（即 128 个字节），主要保存 DM9008 的配置信息，如网络硬件地址、I/O 基地址、中断线选择等配置寄存器内容，在 DM9008 复位后读取 93C46 的内容并设置内部配置寄存器的值。如果 93C46 中内容不正确，DM9008 就无法正常工作。所以通常先通过编程器把配置数据写入 93C46，再把它焊在电路板上。DM9008 通过 EECS、EEDI、EEDO、EECK 与 93C46 相连。

由于本设计只使用 10BASE－T，即采用无屏蔽双绞线的 RJ45 接口，而 DM9008 已内置了 10BASE－T 的收发器，故接口部分电路比较简单，只需要外接 1 个隔离滤波器 LAN7236 即可。TPTX +、TPTX - 为发送线，TPRX +、TPRX - 为接收线，经隔离后的 4 根线与 RJ45 接口相连。

对于其他型号的单片机，如 16 位单片机 80C196KC 等，其接口电路只需稍作修改即可改为 16 位数据总线方式。

三、软件设计

DM9008 的编程包括初始化、发送、接收三部分功能。在接收和发送数据以前要进行必需的检测和初始化。DM9008 的初始化主要是设置所需的寄存器状态,建立网络接口收发的条件。

网络接口通过 2 个 DMA 操作来完成数据的接收和发送。本地 DMA 完成 DM9008 与其内部 FIFO 队列之间的数据传送,远程 DMA 完成 DM9008 与 CPU 之间的数据传送。

DM9008 可寻址的空间有 32 个,分别为 00H ~ 1FH。其中 00H ~ 0FH 是寄存器区,00H 作为命令寄存器,通过设置可选择 3 个页面,10H ~ 17H 为数据端口,18H ~ 1FH 为复位端口。

1. DM9008 的初始化

DM9008 的具体初始化过程如下(CPU 对 DM9008 的寻址需要加上基地址,为了描述方便,省略掉基地址直接用 DM9008 的内部地址描述寄存器地址):

⑴ 读入 1FH 端口数据,再写回该地址以启动 DM9008 工作。

⑵ 向命令寄存器 CR (00H) 写入 21H,选择寄存器页面 0,并进行软件复位。

⑶ 设置数据结构寄存器 DCR (0EH) 为 48H。

⑷ 设置方式状态寄存器 TCR (0DH) 为 02H。

⑸ 读出网络的物理地址:

① 设置远程 DMA 计数器 RBCR1 (0BH)、RBCR0 (0AH) 的值为 000 CH;

② 设置远程 DMA 地址 RSAR1 (09H)、RSAR0 (08H) 的值为 0000H;

③ 设置命令寄存器 CR (00H) 为远程 DMA 读,即 0AH;

④ 重复从数据端口 (10H) 读 6 个字节,这 6 个字节即网络物理地址;

⑤ 停止远程 DMA,设置 CR 为 21H,RBCR1、RBCR0 为 0000H。

⑹ 设置接收状态寄存器 RCR (0CH) 为 04H。

⑺ 划分缓冲区为接收缓冲区和发送缓冲区,并建立接收缓冲环。将 DM9008 内部 RAM 地址为 4000H ~ 4BFFH 设置为发送缓冲区,4C00H ~ 7FFFH 设置为接收缓冲区,即设置 PSTART (01H) 为 4CH,PSTOP (02H) 为 80H,BNRY (03H) 为 4CH。

⑻ 设置 CR 为 61H,选择页面 1。

⑼ 设置网卡地址寄存器,把 PAR0 (01H) ~ PAR5 (06H) 设置为前面读出的物理地址。

⑽ 设置当前页面寄存器 CURR (07H) 为 PSTART + 1,即 4DH。

⑾ 清除多址寄存器,即 MAR0 (08H) ~ MAR7 (0FH) 为 00H。

⑿ 设置 CR 为 21H,选择寄存器页面 0。

⒀ 清除中断状态寄存器 ISR (07H) 为 0FFH。

⒁ 设置中断屏蔽寄存器 IMR (0FH) 为 3BH,即接收中断允许、接收错误中断允许、发送错误中断允许、溢出中断允许、计数器溢出中断允许。

⒂ 设置发送设置寄存器 TCR (0DH) 为 00H。

⒃ 设置 CR 为 22H,芯片进入工作状态。

至此,DM9008 的初始化过程完成,DM9008 处于接收状态。只要网络上有可以接收的数据包,DM9008 自动将数据存入接收缓冲区并在收完后向 CPU 发中断申请。

2. 接收数据

DM9008 收到一个完整的以太网数据包后,向 CPU 发出中断请求,CPU 响应 DM9008 的中断申请后,进入中断服务程序并开始接收数据,具体过程如下。

⑴ 读出中断状态寄存器 ISR,并写回该寄存器。

⑵ 判断是否数据接收中断,如果不是,不执行以下步骤。

⑶ 设置 CR 为 22H,选择页面 0。

⑷ 设置远程 DMA 地址寄存器 RSAR1、RSAR0 为接收地址指针,该指针高位字节初始值位 PSTART + 1,低位字节为 0。

⑸ 设置远程 DMA 计数器 RBCR1、RBCR0 为 0004H。

⑹ 设置 CR 为远程读 0AH,读数据端口,读出 4 个字节,这 4 个字节第 1 个字节表示接收状态,第 2 个字节为下一包开始地址指针,第 3 ~ 4 个字节为本数据包的长度(高位字节在前)。

⑺ 设置 CR 为 22H,远程 DMA 完成。

⑻ 根据接收状态判断数据包是否接收正确,如果接收正确,启动远程 DMA,收取该数据包并进行处理。

⑼ 结束远程 DMA,设置下一次接收数据指针和接收边界指针。

3. 发送数据

数据的发送过程应包含三个步骤:数据包的封装;通过远程 DMA 将数据包送入 DM9008 的数据发送缓冲区;通过 DM9008 的本地 DMA 将数据送入 FIFO 进行发送。具体过程如下。

⑴ 数据包在发送前应该按规定的格式封装好,格式如下:

目的地址(6 字节)	源地址(6 字节)	协议(2 字节)	数据(不小于 46 字节)

⑵ 把上面的数据包通过远程 DMA 写送入 DM9008 的数据发送缓冲区。

① 设置 CR 为 22H,选择寄存器页面 0;

② 设置中断状态寄存器 ISR 为 40H,清除发送完成标志;

③ 设置远程 DMA 地址寄存器 RSAR1、RSAR0 为 4000H,即发送缓冲区开始地址;

④ 设置远程 DMA 字节计数寄存器 RBCR1、RBCR0 为发送数据包的长度;

⑤ 设置 CR 为 12H,设置命令寄存器为远程 DMA 写;

⑥ 往数据端口写入发送数据;

⑦ 查询中断状态寄存器 ISR,等待远程 DMA 完成;

⑧ 设置 CR 为 22H,设置 RBCR1、RBCR0 为 0,远程 DMA 停止;

⑨ 设置 ISR 为 40H,清除发送完成标志。

⑶ 启动本地 DMA,把数据发送出去。

① 设置发送字节计数器 TBCR1(06H)、TBCR0(05H)为发送数据包的长度;

② 设置发送页面起始地址 TPSR(04H)为 40H,即发送缓冲区开始地址高位字节;

③ 设置命令寄存器 CR 为 26H,启动发送。

4. 高层通信协议

上述发送、接收过程所完成的协议是 MAC 层和物理层的协议。要真正实现嵌入式系统与以太网上其他设备(如 PC 机)之间的通信,还需要在嵌入式系统中实现更高层的通信协议,如

TCP/IP 协议,这样 PC 机的程序员就可以使用 TCP/IP 协议透明地访问嵌入式系统的数据。

因此,上述以 8051 单片机系统为例的嵌入式系统的软件设计中除了实现收发数据的功能外,还需要实现 TCP/IP 协议及更高层的应用层协议才能真正实现整个系统的通信功能。由于 TCP/IP 协议的实现通常采用 C 语言,并且有现成的源程序,所以在用 8051 系列单片机编程时,可采用 C51 语言并参考 TCP/IP 标准的源程序来具体实现。有关这方面内容,可以查看有关 TCP/IP 协议方面的资料。

参 考 文 献

1 DAVICOM. DM9008 ISA/Plug & Play Super Ethernet Controller. DATASHEET, 2000

2 W. Richard Stevens 著. TCP/IP 详解. 范建华译,北京 机械工业出版社,2000

选自《电子技术应用》月刊,2002 年第 3 期

4.2 以太网在网络控制系统中的应用与发展趋势

上海同济大学电子与信息工程学院 (200092) 李炳宇 萧蕴诗

网络控制系统 (Network Control System, NCS) 又称为控制网络。分布式控制系统 (DCS)、工业以太网和现场总线控制系统 (FCS) 都属于网络控制系统。它们的出现标志着控制系统正向着网络化、集成化、分布化和节点智能化的方向发展。

现场总线控制系统是用开放的现场总线控制通信网络,将现场控制器和现场智能仪表设备互连的实时网络控制系统。开放性、分散化和低成本是现场总线最显著的特点。目前国际上有 40 多种现场总线。由于各大总线厂商都在积极参与和把持标准的制定工作,因此导致了在现有的产品结构和应用水平上,现场总线领域很难统一,不易广泛应用。由于以太网技术具有数字式互连、互操作性、开放性和高网络性能等特点,是最符合网络控制系统现场总线特点的技术,因此许多人认为现场总线将完全统一于以太网。但是以太网起初并不是为工业控制而设计的。虽然许多组织 (如 IEA) 和公司在不断改进其技术,但其能否完全应用于工业实时控制,还有许多争议。本文通过分析现场总线控制系统的发展现状、以太网的技术特点和控制网络在企业特别是制造业的应用特点,指出了网络控制系统的发展趋势和以太网在网络控制系统中的地位。

一、现场总线控制系统发展现状

分布式控制系统曾经在网络控制系统中占主导地位,它的核心思想是分散控制、集中操作、分级管理。虽然它在一定程度上克服了集中式数字控制系统对控制器处理能力和可靠性要求高的缺陷,但其一对一的结构导致布线量大、布线种类多、工程周期长、安装费用高和维护困难等问题。此外,由于采用单向模拟信号传输,因此可靠性和互操作性差,有时会使现场设备处于失控状态。而采用现场总线控制系统可以很好地解决分布式控制系统存在的上述问题。在现场总线技术诞生的初期,它的主要功能是将当时的可编程逻辑控制器 (PLC) 以一种较简洁的方式连接起来。随着计算机技术引入 PLC,计算机通信技术也被引入现场总线。PLC 功能的增强对现场总线提出了更高的要求。同时,计算机通信技术的引入也大大增强了现场总线的功能。现场总线发展迅速,目前已有 40 多种现场总线,如 Interbus、Bitbus、DeviceNet、MODbus、Arcent、P - Net、FIP、ISP 等。其中最有影响力的有 5 种:FF、Profibus、HART、CAN 和 LonWorks。2000 年 1 月 4 日,国际电工委员会 (IEC) 发布了 IEC61158 标准。新标准包括 8 种类型的现场总线标准,其具体定义如表 4.2 - 1 所列。

表 4.2－1 IEC61158 新标准定义的 8 种现场总线

类型 1	IEC 61158	IEC 技术报告 (即 FF H1)
类型 2	ControlNet	现场总线 (Rockwell 公司支持)
类型 3	Profibus	现场总线 (德国西门子公司支持)
类型 4	P－Net	现场总线 (丹麦 Process Data 公司支持)
类型 5	FF HSE	High Speed Ethernet (美国 Fisher-Rosemount 公司支持)
类型 6	SwiftNet	现场总线 (美国波音公司支持)
类型 7	WorldFIP	现场总线 (法国 Alsthom 公司支持)
类型 8	Interbus	现场总线 (德国 Phoenix Contact 公司支持)

现场总线都有各自不同的特色和市场定位,如 FF (Foundation Fieldbus) 主要应用于石油化工、工业过程控制中的仪表连接等;Profibus 主要用于 PLC 之间的通信,CAN (Controller Area Network) 主要用于汽车监控、开关量控制等。控制网络的层次结构如图 4.2－1 所示。该网络有 3 个层次:现场设备层、控制层和信息管理层。不同类型的现场总线在网络中的应用位置不尽相同,如 FF 的 H1 和 DeviceNet 主要用于控制网络的最底层,连接各种仪表、变送器、调节阀等现场设备;Profibus 的 DP 主要用于控制层的控制器、数据集中器等设备。

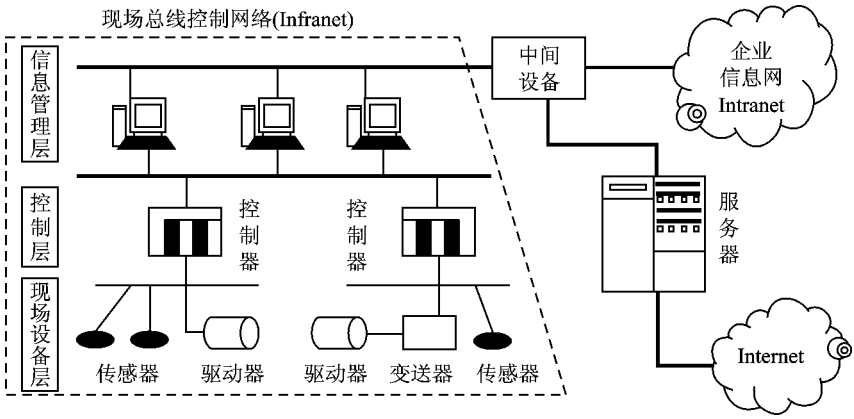


图 4.2－1 控制网络层次结构

网络控制系统要求用于控制领域的现场总线是具有高网络性能的数字式互连网络,具有互操作性和开放性。数字式互连是网络的基本特征。互操作性要求采用统一的标准和协议来实现。开放性指技术公开且被广泛应用。而现在多种现场总线标准并存已成定局。不同公司为了各自的利益,把持着标准的制定工作,使现场总线不易广泛应用。目前,以太网技术是最符合网络控制系统现场总线特点的技术,因此许多公司已经将以太网引入到工业控制网络中。

二、以太网的技术分析

1. 以太网技术基础

以太网是由 DEC、Intel 和 Xerox 3 家公司在 20 世纪 70 年代开发研制的。它采用的是载波监听/冲突检测 (CSMA/CD) 的多路访问协议。设计以太网的初衷是要使系统能够不局限于某一种传输介质,如同轴电缆、光纤、无线电波,因此取名为以太网。20 世纪 80 年代中期,

IEEE 在 DEC、Intel 和 Xerox 3 家公司开发的以太网基础上制订了 802.3LAN 标准。因此现在有 IEEE 的 802.3LAN 标准和 DIX Ⅱ以太网 2 个标准，但二者差别不大，其帧格式分别如图 4.2－2和图 4.2－3 所示。

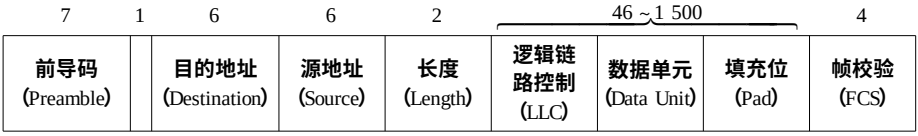


图 4.2－2 IEEE 802.3 数据帧格式

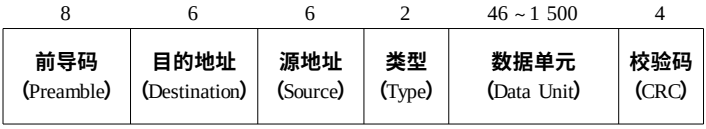


图 4.2－3 DIX Ethernet 数据帧格式

需要说明的是 在 IEEE 802.3 标准中，定义目的地址和源地址的长度是 2 或 6 个字节。因此帧的长度范围为 64～1 518 个字节。但在实际应用中，目的地址和源地址长度都使用 6 字节，因此帧的实际最小长度为 72 个字节。

2. 以太网的优势

控制技术与计算机及网络技术的紧密结合，促使传统控制系统向现场总线控制网络发展。Infranet 与 Internet 和 Intranet 互连实现了控制网络与数据信息网络的结合，丰富了网络的信息内容，降低了企业生产维护成本。例如，远程监控、诊断、维护和服务就是 Infranet－Intranet－Internet 互连体系结构的应用之一。以太网和 TCP/IP 在信息技术领域已经基本上占据主导地位。商用计算机领域的通信网络已逐步被以太网和 TCP/IP 协议垄断。因此，以太网和 TCP/IP 也随着控制技术与信息技术的结合进入到过程控制领域。其优势主要体现在以下方面。

⑴ 成本低，技术支持广泛。以太网已被使用多年，具有大量的软、硬件资源和开发设计经验。应用于以太网上层的 TCP/IP 协议也比较成熟。绝大多数网络产品均提供以太网接口，许多计算机操作系统也配备了 TCP/IP 协议。因此，以太网产品价格相对较低，有丰富的软、硬件支持，有大量从事开发和维护的技术人员。

⑵ 性能高，发展迅速。以太网发展速度，从 10 Mbps、100 Mbps 已发展到千兆以太网。1999 年以来，IEEE 802.3 HSSG (High Speed Study Group) 开始专门研究 10G 标准——802.3ae。由于高速以太网的帧格式不变，使之向下兼容，升级成本低，因此以太网是理想的高性能网络载体。

⑶ 实现网络的无缝连接。在以太网上使用 TCP/IP，其开放性可以使测控网和信息网统一起来。现场信号可以在企业的 Intranet 上及时发布和共享，还可以在 Internet/Intranet 的任何位置对现场智能设备进行在线控制、功能组态以及远程诊断等，实现网络控制系统真正意义上的开放性和互操作性。

⑷ 出现新技术和标准以弥补以太网在工业控制中的不足。工业控制对数据和命令传输的确定性、可预测性和同步性要求极为严格。而以太网采用 CSMA/CD，当发生碰撞时要重试，使数据和命令的传输具有不确定性 (nondeterministic)。针对这种情况，交换式以太网可以将

网络分段,降低甚至避免碰撞,而且提供全双工通信,使带宽增加一倍。高速以太网和交换式以太网的结合使得其在时间响应方面的性能大大提高。

采用以太网作为现场总线,可以使现场总线技术和计算机网络技术很好地融合,形成 2 种技术相互促进的局面。因此,许多人认为网络控制系统将完全统一于基于以太网的控制网络。

3. 以太网在工业控制中的不足

高速以太网的出现虽然缩短了响应时间,提高了网络的性能,但并没有完全解决其在工业控制中的问题,而且新技术的引进也带来了新的问题。

⑴ 没有根本解决不确定性。虽然目前以太网的传输速度大幅度提高,但它仅仅是统计意义上平均速度的提高,这对严格的实时控制系统来说是不可靠的。交换式以太网为避免碰撞,要求一个接口对应一台设备,这样导致交换机体积庞大。但随着接口数量的增加,交换机内部的接口映射阵列将极其复杂,难以实现。交换机的引入会增加网络的复杂程度,而且交换机不能保证在恶劣的工业环境中正常工作。即使增加以太网的带宽,系统的效率也不会随着带宽的提高而线性增长。例如,具有 6 个 16 bit I/O 的系统,其巡回时间是 1.9 ms,其中数据在网络上的传递时间为 0.69 ms,剩余的 1.21 ms 是软件延迟时间。由于软件延迟时间不随带宽变化,因此在 100 Mbps 的快速以太网上,该系统的巡回时间是 0.69/10 + 1.21 = 1.28 ms。可以看出网络速度提高到 1 000%,而巡回速度只提高了 33%。

⑵ 带宽利用率不高。如果建一个有 6 个 I/O 节点的网络,每个节点有 16 bit 的 I/O,则该网络的总通信量是 6 ×2 ×72 ×8 =6 912 bit (因为以太网实际帧长度为 72 个字节)。而 CAN 总线的有效帧长只有 64 bit,且没有最小数据包长度限制。相同的远程 I/O 系统将只需 6 ×2 ×64 =768 bti 的通信量,是以太网效率的 9 倍。即使以太网最小帧长度为 64 个字节也足以使只有 16 bit 的 I/O 不堪重负。且在这 64 个字节中,18 个字节是被以太网自己使用的,TCP/IP 的数据报头也要占用 40 个字节,因此用户只能使用剩下的 6 个字节来存放数据。

⑶ 不符合工业标准。以太网最初不是为工业应用而设计的,如 RJ45、UTP 和 HUB 都不能保证在恶劣的工业现场正常工作。要解决的环境适应性问题包括电磁环境适应性、气候环境适应性 (如耐温、防水、防尘)、机械环境适应性 (耐冲击、耐振动)。为此而制定新的工业标准,使用新的符合工业标准的连接头、电缆、集线器等。但这时以太网已经没有成本上的优势,其成本甚至会高于相同功能的现场总线。

⑷ 安全性不高。以太网虽然可以简化网络控制系统,将控制网络的 3 层结构用一根总线连接起来,使网络结构扁平化。但以太网不能给现场设备供电,没有冗余,不能及时恢复,一处故障可能会导致整个系统的瘫痪。与 Internet/Intranet 连接虽然可以实现控制网络与数据信息网络的接合,大大降低企业成本,但信息网络的故障可能会导致工业控制网络也不能正常工作,增大故障率,且黑客的存在使得控制网络的信息安全受到了威胁。

⑸ 在应用层没有统一的标准。以太网与工业现场控制相结合的一个重要出发点是以太网有良好的互联性,然而当前众厂商开发的工业以太网在上层协议,特别是应用层上未形成统一的标准。例如,若要将 Profibus、Modbus、Contro1Net、DeviceNet 协议转换到 TCP/IP 上,把多种不同的协议应用于同一网络,并让它们与同一主机在同一时间对话,则每种协议都要有一个 Driver。因此,供应商设计了不同的协议,并把它们统称为工业 Ethernet 网,但是它并未真正解决通用标准的问题。

三、发展趋势

当前,网络技术的迅速发展,给制造业带来了新的变化和重大影响。2000 年 11 月国际权威制造技术研究机构德国 Fraunhofer 研究院的院长 Warnecker H J 教授,在上海召开的第一届国际机械工程会议上作了题为"网络生产——制造业全球化的前景"的大会报告,强调了网络生产在制造业中发展的重大影响和作用,指出了制造全球化、制造敏捷化和制造虚拟化均离不开制造网络化的支撑环境。因此可以说制造网络化是现代制造业发展的主要趋势之一。

在企业外部,随着电子商务与 Internet 技术的发展,出现了分散网络化制造技术。该技术针对某一市场需要,利用 Internet 灵活而迅速地组织社会制造资源,把分散在不同地区的现有生产设备资源、智力资源和各种核心能力,迅速地组合成一种超越空间约束的、统一指挥的经营实体即网络化联盟企业,以快速推出高质量和低成本的新产品。在企业内部,工业自动化控制系统的网络结构越来越分散,系统越来越复杂,内部的连接更加高速化、紧密化。系统被分成了许多独立的"控制孤岛"对驱动器和用户接口的需求更多。而命令、数据要在不同的硬件平台上传递,要求软件与硬件无缝连接。产品模型的数字化已基本得到解决,但将模型直接用于加工和制造,还需要对模型的精度、力学结构等加工特性进行校验。由于庞大的三维模型数据和计算密度,要求不断提高带宽和加强深度计算与分布计算能力,而分布式实时控制的前提及其确定性是一种网络技术,其性能至少要高于现行的现场总线。基于这些需求,越来越多的企业开始尝试用以太网作为网络控制系统的总线,并使用 TCP/IP 协议。

在控制网络体系中,信息管理层负责对下层的监控管理和与外部网络的连接。对数据的延迟时间、同步及可确定性等要求没有工业控制那么严格,以太网技术完全可以满足其要求。因此,这一层基本已经统一到以太网。在控制器层,许多控制器都提供以太网接口,控制室的环境已大大改善,并且在一些应用中以太网已经向下延伸到控制层。现在许多公司开始尝试将以太网建到 I/O 级,推出了 Ethernet 智能 I/O 模块,配以各种各样的 I/O 模块,组成前端智能 I/O 系统,对外提供以太网通信接口(如 Modicon 和 Opto 22)。

为解决以太网存在的问题,Synergetic Micro Systems、Hirschmann、Grayhill、HMS Fieldbus System 和 Hilscher GmbH and Contemporary Controls 公司于 1999 年成立了"工业以太网协会"(Industrial Ethernet Association, IEA)。目前该协会已发展有十几个成员。IEA 的目标是解决以下问题。

⑴ 确定数字或模拟 I/O 打包到 TCP/IP 的方式。

⑵ 与复杂设备接口的标准。这些设备包括:驱动设备、运行控制器、操作员界面、PLC、条形码阅读器等。

⑶ 确定工业级的接头。

⑷ 采用"确定性"机制。

⑸ 定义工业 Ethernet 和企业 Intranet 的接口。

IEA 希望通过上述工作,使工业以太网成为真正的工业现场总线。但鉴于目前以太网在工业控制中的优点和缺点,它只能应用于控制网络的中上层。以太网的技术基础使之很难应用于控制网络底层简单的 I/O 和低位数的控制器。因此,网络控制系统将根据不同的现场情况和控制要求,兼顾经济效益,选择一种或几种总线和以太网来构建控制网络。在未来的几年里,随着网络技术、以太网技术和相关工业标准的不断完善以及智能 I/O、智能变送器等控制

网络底层设备的丰富,以太网将会越来越多地应用于工业控制并逐渐占据主导地位。大部分现场总线将由于以太网延伸到其应用领域及其他现场总线的残酷竞争而逐步被淘汰。Ethernet 将与少数几种适用于小规模生产系统的 bit 级总线标准长期共存,构成完整的网络控制应用体系。

四、结束语

网络控制系统的发展方兴未艾,其应用范围广,影响层面深。计算机网络、以太网以及控制领域中各种新技术的出现、发展和融合,将引发控制领域更深层的变革。人们可以期望高度开放、使用灵活方便、功能强大的新型控制系统的出现。

参 考 文 献

1 顾洪军. 网络控制系统的实时特性分析及数据传输技术. 计算机工程与应用,2001,37 (6)
2 吴誉. 现场总线技术展望及我国的发展策略. 测控技术,1999,18 (11)
3 Frazier H. The 802.3z Gigabit Ethernet Standard. IEEE Network, 1998
4 Allen D. Will Gigabit Ethernet WAN Services Make Us Forget About SONET? Network Magazine, 2001, (6)
5 Hulsebos R. Enter Ethernet, Exit Fieldbus? http://ethernet.industrial-networking.com/articles,2001
6 李健,刘飞. 基于网络的先进制造技术. 中国机械工程,2001,12 (2)
7 Bongaerts L. Hierarchy In Distributed Shop Floor Control. Computers In Industry,2000, (43)
8 张曙. 分散网络化制造. 北京 机械工业出版社,1999
9 单忠德. 基于分散网络化的铸件快速柔性制造. 中国机械工程,2001,12 (3)

选自《微型机与应用》月刊,2002 年第 11 期

4.3 IPv4 向 IPv6 的过渡

潘素玉 邹建华

一、引 言

基于 TCP/IP 协议簇的 Internet 在最近 20 年中得到了飞速的发展,Internet 的影响力波及到了社会的各个角落,深刻地影响着人类社会的文明进程。目前 Internet 上正在广泛使用的 IP 协议——IPv4 产生于 1974 年,Internet 经过近 30 年的发展,无论从规模、技术,还是用户的要求来说,IPv4 已经越来越不能满足实际使用和进一步发展的要求。IPv4 如今面临着如下的问题。

首先是地址空间的局限性,IPv4 提供了 32 位的地址空间,但随着 Internet 网络规模和数量迅速增长以及 IP 地址的实际利用率很低,IP 地址面临着很快就要耗尽的问题。其次是性能,IPv4 制定之初,主要目的在于为在异种网络间进行数据的可靠和高效传输探索最佳机制,从而实现不同计算机的互操作。在很大程度上,IPv4 实现了此目标,但这并不意味着 IPv4 可以继续实现这些目标,一些源自 20 年前甚至更早以前的设计需要进行改进。再次是安全性,很久以来,人们认为安全性议题在网络协议栈的低层并不重要,应用安全性的责任仍交给应用层,在许多情况下,IPv4 设计只为具备最少的安全性选项,但安全性现在已经成为 IP 的下一版本可以发挥作用的地方。最后是自动配置。对于 IPv4 节点的配置一直比较复杂,而网络管理员与用户则更喜欢"即插即用"。IP 主机移动性的增强也要求当主机在不同网络间移动和使用不同的网络接入点时能提供更好的配置支持。

同时,急速增长的网络用户和连接站点数、对高速网络的支持、提供包括图像和视频的混合数据流及更好的网络安全性能等诸多要求都需要 Internet 网络体系结构和承载技术做相应的改进。新的运行环境和网络应用要求增加新的功能,IPv6 由此应运而生。但是,由于 IPv4 是目前 Internet 中正在运行的非常成功的协议,有着广泛的应用基础,虽然人们公认 IPv6 是下一代的 IP 协议,但 IPv4 向 IPv6 的转变不可能一夜之间完成。它们之间的过渡需要一个较长的时期。

二、IPv4 向 IPv6 的过渡

实现 IPv4 向 IPv6 过渡的方法有 3 种,即隧道法、双栈法和翻译法。

1. 隧道法

在 IPv6 被全面采用之前,众多局部 IPv6 网络将不得不和 IPv4 骨干网络长期共存,连接这些孤立的"IPv6 岛"必须使用隧道技术,这也是 IPv4 向 IPv6 过渡初期最常用的技术。路由器将 IPv6 的数据分组封装为 IPv4,IPv4 分组的源地址和目的地址分别是隧道入口和出口的 IPv4 地址 到了隧道的出口,再将 IPv6 分组取出转发给目的站点。隧道技术只需要在隧道的入口和出口处进行修改,所以非常容易实现,只是存在不能实现 IPv4 主机与 IPv6 主机直接通信的

问题。在图 4.3－1 中,节点 A 和节点 B 都是只支持 IPv6 的节点,如果节点 A 要向 B 发送包,A 只是把 IPv6 头的目的地址设为 B 的 IPv6 地址,然后传递给路由器 X,X 对 IPv6 包进行封装,然后将 IPv4 头的目的地址设为路由器 Y 的 IPv4 地址;若路由器 Y 收到此 IPv4 包,首先拆包,如果发现被封装的 IPv6 包是发给节点 B 的,则将此包正确地转发给 B。

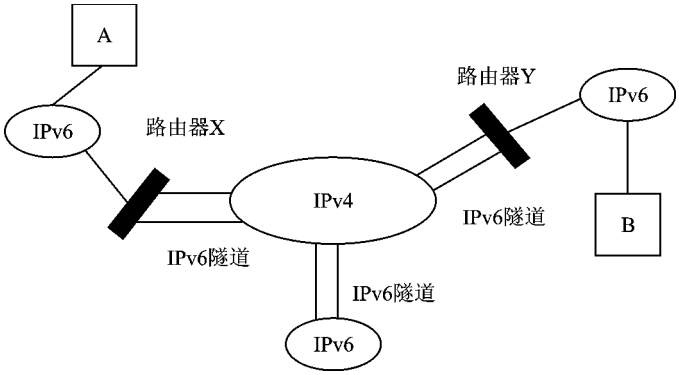


图 4.3－1 隧道连接

配置隧道 在配置隧道中,隧道的终点是一个中途的 IPv6/IPv4 型路由器。它将 IPv6 分组在隧道传输中加上的封装去掉,然后将分组传送到目的地。通过配置隧道法,路由器在 IPv4 网上建立了端到端的连接,这种连接可以传递 IPv6 包。

自动隧道 IPv6 分组直接经隧道发往最终目的地,隧道的终点是分组的终点。只有执行隧道功能的节点 IPv6 地址是 IPv4 兼容地址(IPv4 兼容地址是指在 128 位地址中,高阶的 96 位全部为 0,而最后的 32 位包含 IPv4 地址)时,自动隧道才是可行的。在为执行隧道功能的节点建立 IP 地址时,自动隧道方法无需进行配置。但通常情况下,为了使 IPv6 主机可以与所有的 IPv6 主机(IPv6 主机和使用 6 to 4 过渡的主机)通信,自动隧道要与配置隧道结合使用;当自动隧道允许远距离的主机进入 IPv6 网,为了完成端到端的通信,自动隧道必须与其他的方法结合使用。

6 to 4 机制 6 to 4 是一种自动构造隧道的技术。它实现转换的关键是激活 IPv6 路由器间的连通性。当缺少本地 IPv6 服务时,IPv6 的分组可以被封装到 IPv4 的分组中来实现连通,所以只需要一个全球惟一的 IPv4 地址便可使整个站点获得 IPv6 的连接。这种机制主要应用在与 IPv4 网连接的孤立的 IPv6 站点/主机。当没有 IPv6 支持时,这些孤立的站点/主机可以与其他的 IPv6 站点/主机通信。6 to 4 机制还可以使用在专用 IPv4 地址和只有一个路由地址的网中。

2. 双栈方法

很长时间内,IPv4 仍将存在,即使一些网络的部分已被升级到 IPv6,到那时,升级系统将需要保持与 IPv4 系统的互操作能力。任何情况下,同时支持 IPv4 和 IPv6 的系统都是必要的。

链路层接收到数据段,拆开并检查包。如果 IPv4/IPv6 头中的第一个字段,即 IP 包的版本是 4,该包就由 IPv4 栈来处理;若为 6,则由 IPv6 栈来处理。

最简单的双栈工作是只支持 IPv4 和 IPv6,但不支持隧道方式,如同用于访问 IPv4 网络服务器一样,同一应用也能访问本地 IPv6 网络。节点可以与任何 IPv4 节点或 IPv6 节点互操作,但只限于与其有连接能力的网络。如图 4.3－2 中,可以与主机 D 互操作的节点包括:A 网与

B 网的 IPv4 或 IPv6 节点,M 网中的所有 IPv4 节点;但不能与 C 网中的节点互操作,因为从 A 网到 C 网没有 IPv6 路径。

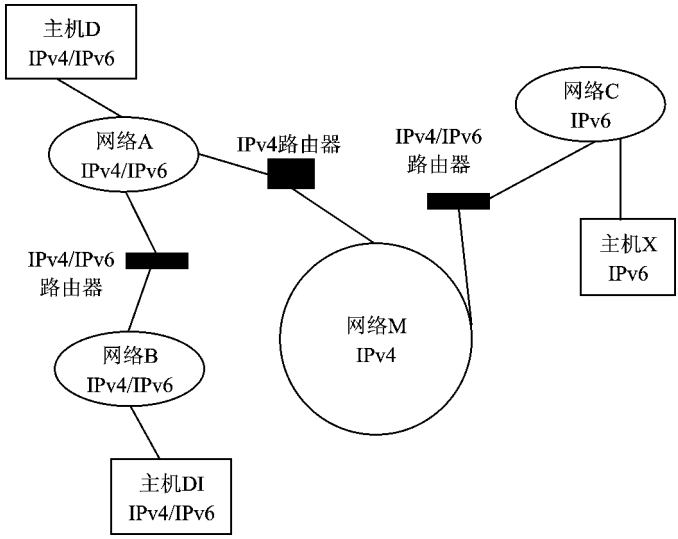


图 4.3-2 双栈节点路由器和网络的互换

支持隧道方式的双栈节点增加了在 IPv4 网络上进行互操作的能力,而无需额外的 IPv6 路由器。如果节点 D 能在 IPv4 上以隧道方式传递 IPv6 包,则它可以使用本地 IPv4 路由器将包发给网络。

3. 翻译法

翻译法是支持不同协议的主机可以实现通信。翻译法的缺点是常常会破坏端到端服务(如端到端的 IP 安全)。这一点和 IPv4 中的 NAT(网络地址转换)类似。同时,翻译器还会造成网络潜在的单点失效。因此,使用翻译法必须经过慎重考虑,而且对终端应该透明,否则翻译器就需要做相应的修改。它主要有以下几种方式。

报头转换 这个方法可以把 IPv4 的报头翻译成 IPv6 报头,反之亦然。这有点类似于 NAT 协议(IPv4 到 IPv6 报头转换)。尽管这种方法的转换速度很快,但它在应用上存在限制,如它不支持应用层上的转换。

网络地址翻译——协议翻译(NAT-PT) NAT 在 IPv4 环境下已经得到了广泛的应用,NAT-PT 就是在做 IPv4/IPv6 地址转换的同时在 IPv4 分组和 IPv6 分组之间进行报头和语义的翻译。对于一些内嵌地址信息的高层协议(如 FTP),NAT-PT 需要和应用层的网关协作来完成翻译。在 NAT-PT 的基础上利用端口信息,就可以实现了,这点同目前 IPv4 下有本质区别。

Socks 代理服务器法 利用 Socks 代理服务器可以建立 IPv4 和 IPv6 主机通信的一种方式。将代理服务器配置成支持 4、6 双协议栈的连接主机,通过利用 Socks 对 IPv4、IPv6 两个网络的服务器进行两者之间的通信。首先运行 Socks 代理服务器必须支持 IPv4/IPv6 双协议,也就是说服务器的协议栈中必须带有 IPv4 和 IPv6 的协议支持,可以同时访问 IPv6 子网和 IPv4 子网。只有这样它才能在内外网的 IPv4 与 IPv6 之间转化。若 Socks 代理是支持 IPv4/IPv6 双协议的服务器,IPv6 客户可通过此代理访问 IPv6 服务器,同时 IPv4 客户也可以通过此代理访

问 IPv4 服务器。如果进行能够交叉访问,则需要通过域名来访问。在许多情况下,往往只能给出 IP,例如很多 Web 页面中内嵌了 IP 地址,则需要客户端也予以显式的支持,也就是说 IPv6 的客户端软件必须能够识别 IPv4 的目的地址,而 IPv4 客户端则须能识别 IPv6 地址。在向 IPv6 升级的初期,在广大的 IPv4 的"海洋"中存在一系列的已经实现 IPv6 的网络"孤岛"可以利用 Socks 实现这些孤岛和广大互联网络节点之间的通信。如图 4.3-3,利用 Socks 实现 IPv6 主机访问 IPv4 网络。

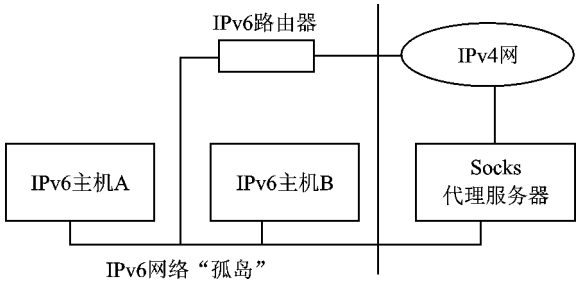


图 4.3-3 利用 Socks 实现 IPv4/IPv6 过渡

选自《现代通信》月刊,2002 年第 12 期

4.4 在嵌入式网络应用中实现 TCP/IP 协议

上海交通大学 (200030) 方捷磊 朱 杰

一、引 言

在网络应用日益普遍的今天,越来越多的嵌入式应用需要支持网络功能。比如,一些小型现场控制设备可能需要增加网络功能来实现远程控制的功能。TCP/IP 是目前一种被广泛采用的网络协议。只要那些设备上实现了 TCP/IP 协议并增加一个网络接口,就可方便地接入到现有的网络中。

考虑到嵌入式应用中硬件系统 (主要是 CPU 和存储器容量) 的多样性,完成特定功能的应用程序也各不相同,因而软件的设计最好易于被移值,尤其是应用程序与网络协议软件应具备一定的独立性。这样,移植到不同的应用系统中时,网络协议软件几乎不用改动,只要更换不同的应用程序即可。

TCP/IP 协议比较复杂,尤其在一些缺乏功能强大的操作系统支持的嵌入式设备上实现更非易事,为此设计了一个软件框架。根据这个框架我们可以在没有任何操作系统支持和存储空间有限的硬件上实现网络功能和应用程序。

整个设计方案是把不同部分的功能由不同的任务来实现。与具体应用相关的部分由一个或几个应用程序任务实现。网络协议软件由几个相关的网络任务来实现。由一个简单的任务管理器来统一管理和调度这些任务。网络任务和应用程序之间通过 API 来交互。这样就从最大程度上实现了模块化和层次分明的要求,因而更易于移植和扩充。

二、任务管理器的实现

任务管理器功能很精简,所以能用极少的代码实现,但它却能满足嵌入式网络协议软件和应用程序的需求。它包括两个关键的数据结构:任务控制块和信号灯。

系统中的每个任务都对应一个任务控制块,任务控制块中有两个变量 saved_sp 和 state。saved_sp 记录着任务当前的栈指针,state 记录着任务的当前状态。state = READY 表示任务可以被调度执行,state = WAITING 表示任务在等待某个信号灯。每个任务都有一个专用的栈。栈有两个作用:一是用来保存局部变量;二是保护中断现场或任务切换现场,就是在发生中断或任务切换时保存当前寄存器值,在中断返回或任务被调度时恢复寄存器的值。

任务间的同步是由信号灯来控制的。每个同步事件都有一个相关的信号灯,信号灯可以用来同步两个任务对。信号灯由两个变量 count 和 waiting_task 组成。count 是对事件的计数。count 大于 0,表示已有 count 个事件发生并等待处理;count 小于 0 表示有某个任务在等待事件的发生,此时 waiting_task 中保存着相应任务控制块的地址。

对信号灯有两个操作:sem_up 和 sem_down。sem_up 首先使 count 值加 1,然后看 count 是否为 0。若为 0,表示有任务在等待,就通过 waiting_task 中记录的任务控制块的地址把等待任

务的 state 设为 READY；否则返回。

sem_down 首先使 count 值减 1，然后判断是否小于 0。若小于 0 会使任务成为 WAITING 状态并引发任务管理器对任务的调度，其处理可参考图 4.4－1 所示的任务管理器的流程图；否则返回。

任务管理器类似于一个程序的主循环，每个任务类似于一个被主循环调用的子程序。任务管理器调度各任务的流程如图 4.4－1 所示。从图中可以看出，各任务之间是协作性的，而非抢先的。

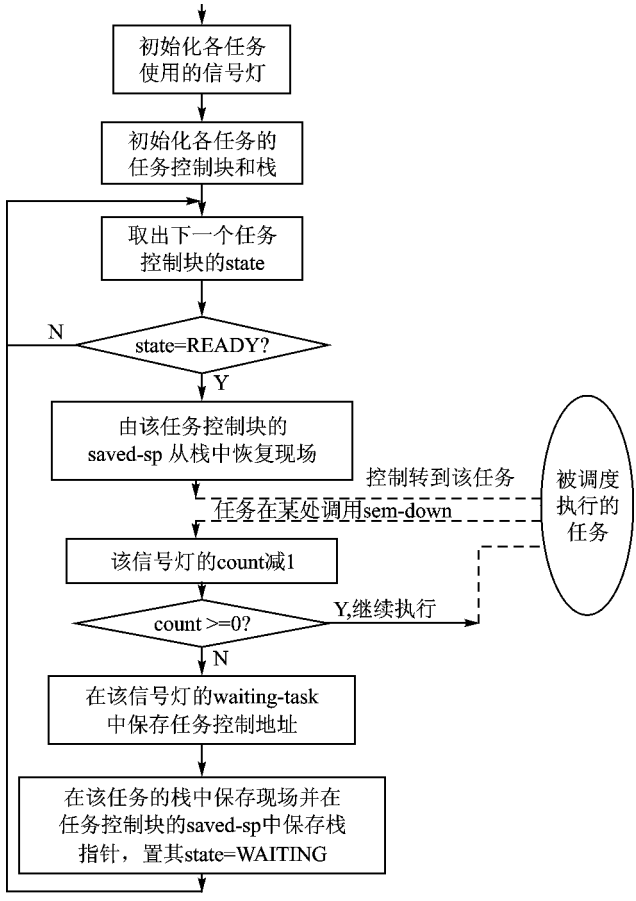


图 4.4－1 任务管理器各任务流程图

三、网络协议的实现

网络软件按组成协议划分为 ARP 模块、IP 模块、ICMP 模块、UDP 模块、TCP 模块。这些模块都根据嵌入式应用的需求作了裁剪，因而可以保证足够小的代码量。由一个输入任务来处理网络输入的数据。由一个定时器任务来处理超时事件。没有单独的网络输出任务，负责网络输出的例程在需要时由网络输入任务和应用程序任务来调用。

1. ARP 模块

ARP 模块完成的主要功能是完成目的 IP 地址和 MAC 地址的映射。同时，为了提高网络

传输速度和效率,避免在每次发送数据时都要发送 ARP 请求包来获得目的 MAC 地址,还要维护一个常用目的地址的 ARP 高速缓存。这些功能主要由 2 个例程来实现:arp_in 用来处理来自网络的 ARP 请求和 ARP 响应,维护 ARP 高速缓存;arp_find 用来为一个 IP 地址在 ARP 高速缓存中找到相应的 MAC 地址。

2. IP 模块

针对嵌入式应用的特点,该 IP 模块不实现路由器对 IP 模块要求的功能,也不支持多 IP 地址。这样大大简化了它的实现。它主要由 2 个处理输出和输入的例程 ip_send 和 ip_in 实现。ip_send 将要发送的数据封装到一个 IP 包中,然后查看发送的目的地址是否与本机在同一子网内。如果是则直接发送,否则把它转发给一个默认的路由器。ip_in 处理收到的 IP 包,如果它是一个分片,则将其组装。它会把完整的 IP 包根据包头中的 protocol 字段传递到相应的 ICMP 模块或上一层协议模块 UDP、TCP 处理。

3. ICMP 模块

许多 ICMP 消息类型是与路由器的功能相关的。该 ICMP 模块只实现对收到的 echo 请求消息产生一个 echo 应答,用以实现一定的网络诊断功能。

4. UDP 模块

UDP 协议比较简单,因而实现起来也较为方便。UDP 模块主要由两个处理输出和输入的例程 udp_send 和 udp_in 实现。udp_send 将来自应用程序的数据封装在 udp 包中,通过调用 IP 模块中的 ip_send 发送出去。udp_in 根据包头中 port 字段的指示将数据送给相应的应用程序。

5. TCP 模块

TCP 协议实现了一种面向连接的、可靠的数据传送机制。TCP 模块是所有模块中最复杂的。协议中规定了一种确认机制和窗口机制,由一个循环缓冲区来实现。协议中还规定了一个 TCP 状态机[2],由 TCP 模块中一个重要的数据结构——TCP 控制块和对应于各个状态的处理例程来实现。TCP 控制块中不仅记录了 TCP 状态机的当前状态,还记录了有关循环缓冲区的相关信息。

6. 与应用程序的接口

每个被应用程序使用的网络连接都通过一个网络接口控制块来管理。对于 UDP 接口来说比较简单,它只需记录本地端口号和对方的 IP 地址和端口号。对于 TCP 来说,还要保留一个 TCP 控制块的地址。另外,每个控制块都有一个用于同步应用程序从网络输入任务接收数据的信号灯。基本的 API 为发送 net_send 和接收 net_recv。对于 TCP 来说,还有用于建立主动连接、被动连接以及关闭连接的 API。所有这些 API 都以相应的网络接口控制块的地址为第一个输入参数。

7. 网络输入任务

网络输入任务有一个同步网络输入的信号灯。当网络上有数据包到来时,网络硬件会产生一个中断。中断驱动程序将数据包拷贝到输入缓冲区,并对信号灯输入执行 sem_up 操作。

当输入任务被调度执行时,它会从输入缓冲区取出一个数据包,调用各层网络模块的输入例程处理。如果数据顺着网络栈到达应用程序,输入任务会通过对网络接口控制块的输入信号灯执行 sem_up 操作告诉等待数据的应用程序。最后,输入任务对同步网络输入的信号灯执行 sem_down 操作,准备继续处理下一个网络输入。

四、超时和重发的处理

在 TCP/IP 协议中多处要用到超时和重发机制。这种机制对于确保两个或多个彼此独立的通信结点从通信错误或故障状态自动恢复到正常状态是非常有效的。但这也增加了软件结构的复杂性。因为对超时的处理往往是独立于正常程序流程的,也就是与正常的程序流程是异步的。

解决问题的关键是系统需要提供一种有效的定时器机制,用以完成对超时项的计时和当超时发生时将处理转到相关的处理程序。

要实现的 TCP/IP 协议软件中有 4 处要用到定时器。第一是在 ARP 高速缓存的维护中。被添加到 ARP 高速缓存中的表项在一段时间后要置为无效。第二是在等待对发出的 ARP 请求返回响应时,可能会在指定的超时时间内还未收到返回的响应。第三是在 IP 组装收到的分片时,由于部分分片在一定时间内没有收到而丢弃整个 IP 包。第四是在 TCP 等待接收方对数据段的确认时。如果在指定时间内还未收到对某个数据段的确认,需重新发送。

从上述可见,要实现的定时器具备以下特点:

⑴ 对定时的精度要求都不是很高,基本都是秒一级的精度。这样,我们完全可以稍滞后一些来处理定时器超时。如此,我们就可以不把超时处理放在时钟中断处理程序中。

⑵ 对同一类超时处理可以由同一处理程序来完成,只是传入到相应的处理程序中的参数不同而已。比如一个 ARP 高速缓存中的表项超时时,需要将其置为无效,可以统一用一个处理程序,参数中放入相应的表项地址即可。

首先,定义一定时器的数据结构,如图 4.4-2 所示。

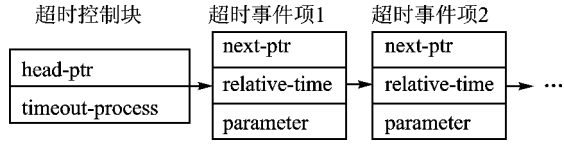


图 4.4-2 定时器的数据结构图

每一类超时都是由一个超时控制块和其所属的一个由超时事件项组成的链表来管理的。整个链表是按超时事件将要发生的时间顺序排列的,先发生的超时事件排列在前。超时控制块中的 head_ptr 用以指向一个超时事件项链表的首项,timeout_process 是超时事件发生时的处理程序的入口地址。在每个超时事件项中,next_ptr 指向链表中的下一项。relative_time 是本表项的超时事件相对于上一表项的超时事件发生的相对时间。所以某个表项表示的超时事件距离当前的时间是它以前所有表项(包括自身)中的 relative_time 的和。relative_time 的基本单位是 granularity。

定时器任务使用一个信号灯用来作同步。每当时钟中断服务程序计数到 granularity 个时钟中断,就给定时器任务使用的信号灯作 sem_up 操作。当定时器任务被调度执行时,它遍历每一个超时控制块,对每一个超时控制块作如图 4.4-3 的处理。最后对信号灯调用 sem_down。

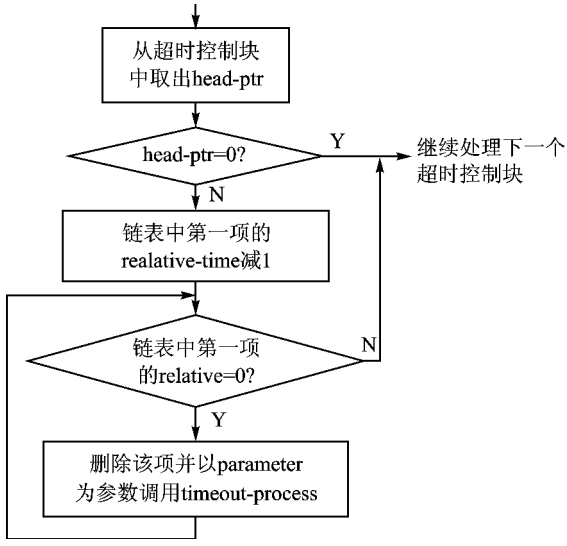


图 4.4-3 超时控制块的处理图

五、同其他实现方案的比较

现在要自行开发嵌入式 TCP/IP 协议软件有两种典型的实现的方法。一种是在 linux 的基础上修改。Linux 是一个源代码开放的软件平台,它实现了包括 TCP/IP 协议软件的诸多功能。但 Linux 一开始毕竟不是为嵌入式应用编写的。Linux 目前最小的容量是 500 KB 内核和 1.5 MB RAM。即便是网虎国际(Xlinux)公司开发的目前世界上最小的 Linux 内核 QUARK,内核也达到 100 KB,总容量近 2 MB。因而把它移值到某些对硬件成本有较苛刻要求的嵌入式应用场合不太合适。

另外一种是 Ubicom 公司专门为其 SENIX 单片机开发的 Internet 协议栈。但其存在以下缺点:对硬件环境有较大的依赖性,用汇编编写,并且软件结构缺乏灵活性,不适合移植到其他系统中;为了在有限的程序存储器上实现网络协议,作了大量简化,因而在性能和通用性上较差。

本设计方案兼顾了小容量和通用性的要求,它可以在多种硬件平台上实现,甚至可以移植到像 8051 系列那样被广泛使用且低成本的单片机上。

参 考 文 献

1 孟庆余. 电子数字计算机实时操作系统. 北京:国防工业出版社,1991
2 Douglas E COMER. TCP/IP 网络互连技术卷 1—原理,协议和体系结构. 北京:清华大学出版社,1998
3 Douglas E COMER. TCP/IP 网络互连技术卷 2—设计,实现和内部结构. 北京:清华大学出版社,1998

选自《微电子学与计算机》月刊,2002 年第 5 期

4.5 一种以太网与 8 位单片机的连接方法

华中科技大学 何锐波 赵英俊

随着计算机技术与网络技术的发展和普及,嵌入式系统在工业自动化、办公自动化和楼宇自动化等领域得到了日益广泛的应用。为了实现远程数据采集、远程监控等功能,网络化已成为新一代嵌入式系统发展的一个重要趋势。目前,以太网作为局域网的骨架,正从办公室逐步延伸进入车间和家庭。因此,研究嵌入式系统与以太网的接入方法,可为新一代网络化嵌入式系统的设计提供必要的基础,具有重要的现实意义和经济价值。

理论上将单片机接入以太网是完全可能的,现实的问题是,如何在单片机有限的硬、软件资源的限制之下来实现这一功能?考虑到这种限制条件,下面重点讨论一般性能的 8 位单片机与以太网的连接问题。

一、网络化系统的最小构成

一个 8 位单片机以太网接入系统的最小系统构成包括单片机(MCU)、以太网控制器和驱动程序。其系统结构如图 4.5-1 所示。下面从以太网控制器的选择、硬件设计和驱动程序编制三个方面进行说明。

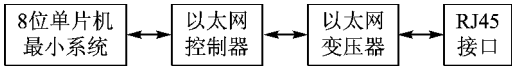


图 4.5-1 8 位单片机以太网接入系统的最小系统构成

1. 以太网控制器的选择

以太网的物理介质有多种标准,这里以 10Base-T 为例,讨论单片机与 IEEE 802.3/802.2 局域网的连接方法。到目前为止,几乎所有的计算机系统都是通过专用的以太网控制器接入以太网的。对于 8 位单片机系统,在选择以太网控制器时,要考虑以下几个因素:

首先,由于主机用的是 8 位单片机,因此要求所选的以太网控制器必须支持 8 位工作模式。实际上,只有部分基于 ISA 总线的以太网控制器才能满足此条件,所以基于 PCI 总线的以太网控制器不在考虑之列。其次,要考虑以太网控制器的片上缓存。最好选用具有足够片上缓存的以太网控制器,以简化系统设计。再次,应考虑以太网控制器与主机的数据交换形式。在 8 位模式下,有的以太网控制器可支持中断,有的则只支持查询。这些势必对系统设计产生一定的影响。最后,根据与 NE2000 的兼容性,这些基于 ISA 总线的以太网控制器还可以分成兼容和非兼容两类。前者一般生产时间较长,可供借鉴的资料较多;后者则多为近期产品,性能更佳。

根据上述分析,表 4.5-1 列出了目前市面上几种可供选用的以太网控制器和主要特性。

表4.5-1 几种以太网控制器的比较

生产厂商	型 号	片内缓存/bit	8位模式中断	NE2000兼容性	相对价格
National Semiconductor	DP8390[1]	无	是	是	—
Davicom	DM9008F[2]	8K×16	是	是	低
Realtek	RTL8019AS[3]	8K×16	是	是	低
Cirrus Logic	CS8900A[4]	4K×8	否	否	高

2. 硬件设计

如前所述,8位单片机系统中的以太网控制器必须工作在8位模式,因此,应根据所选以太网控制器的数据手册和具体的系统要求,进行相应的电路设计。

在8位工作模式下,以太网控制器只能通过I/O模式与8位单片机进行数据的读写,因此需要占用单片机一定的I/O端口。典型的NE2000兼容型以太网控制器本身具有地址译码功能,可以将控制器的高位地址线(SA_{19} ~ SA_5)固定接地或高电平以确定I/O端口基址,另用5根低位地址线(SA_4 ~ SA_0)完成I/O偏移寻址。因此,单片机与以太网控制器的基本接口信号线有16根:读写线2根、地址线5根、数据线8根和片选信号线1根。另外,可根据实际要求选择1根中断请求线。

实际上,为了缩短原型电路的研制时间,可以考虑选用相应的商品网卡与8位单片机系统组成实验平台。这时应注意将网卡设置成"Jumpless"模式,关闭"即插即用"功能,并禁用BootROM。

3. 驱动程序设计

8位单片机与NE2000兼容以太网控制器之间的通信,可用查询方式或中断方式来控制。采用中断方式,可以避免在主程序当中不断地对以太网控制器进行查询,使驱动程序成为完全独立的一个模块,能够很方便地与上层TCP/IP协议模块接口。对采用中断方式的驱动程序,稍作修改就可成为查询方式的驱动程序。下面只就中断方式进行讨论。

在中断方式下,以太网接入驱动程序主要由以太网控制器中断服务程序、接收数据模块和发送数据模块组成,其层次结构如图4.5-2所示。

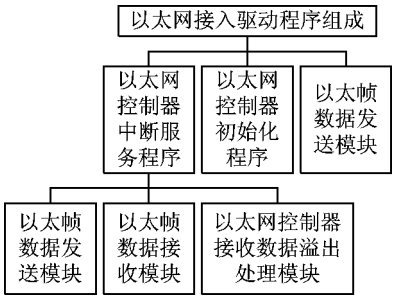


图4.5-2 以太网接入驱动程序组成

在以太网控制器中断服务程序中进行数据发送,可以进一步提高系统性能。

中断服务程序流程如图4.5-3所示。其主要功能是完成以太网帧的发送与接收,整个程序也就是围绕着这两个过程进行的。

数据接收模块与数据发送模块程序的具体编制方法请阅参考文献[5]。

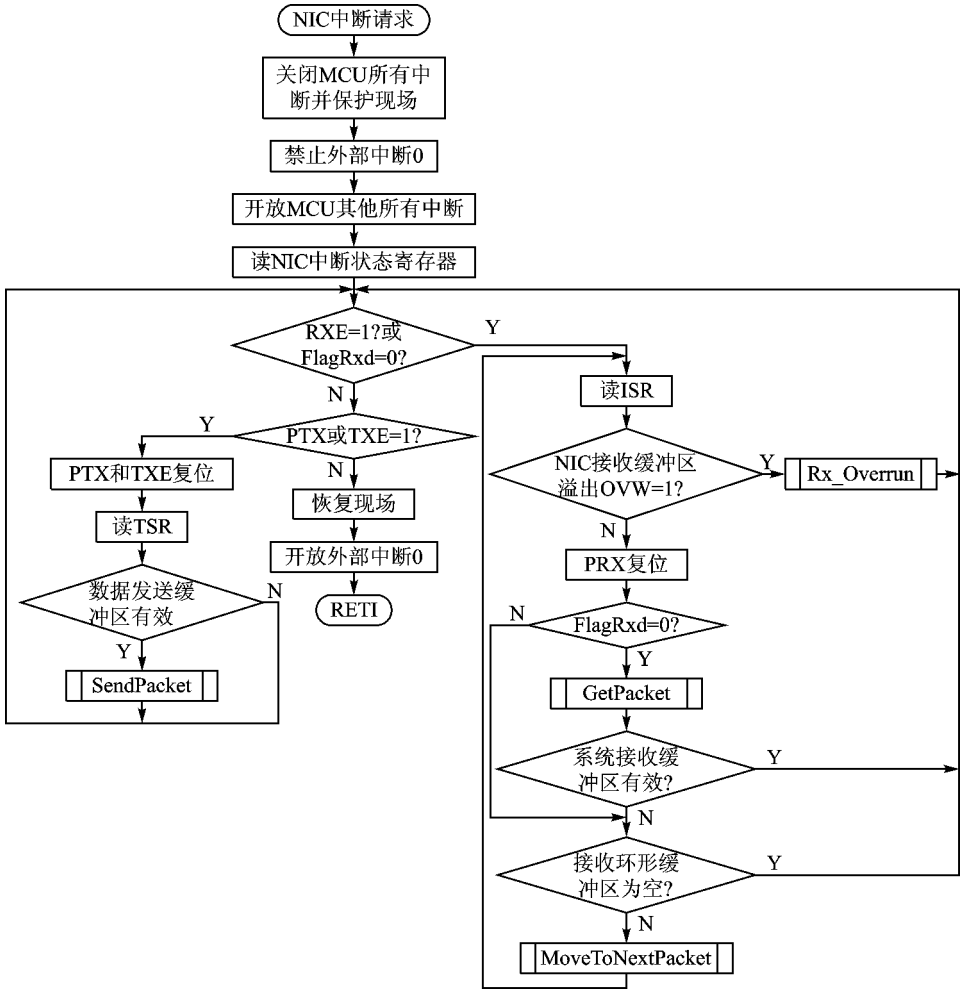


图4.5-3 中断服务程序流程图

二、实 例

下面结合一个嵌入式设备网关的具体设计,对在8位单片机上实现以太网接入时应当特别注意的一些问题作进一步说明。

1. 硬件构成

该嵌入式设备网关的硬件基本结构如图4.5-4所示。其主要硬件包括ISSI公司的8位单片机IS80C32、Davicom公司的10M以太网控制器DM9008F和Waferscale公司的可编程单片机外围器件PSD934F2等。它们组成了以太网接入模块。

图4.5-4表明,采用可编程单片机外围器件(PSD),能够大大简化单片机系统电路。1片PSD,就可以为单片机提供所需的译码电路、数据存储器和程序存储器等电路和元器件。本系统所采用的PSD934F2[6],片上集成有256 KB的主Flash存储器、32 KB的次Flash存储器、8 KB的SRAM、2000门Flash可编程逻辑电路(PLD)和27个可单独配置的I/O端口,同时具有

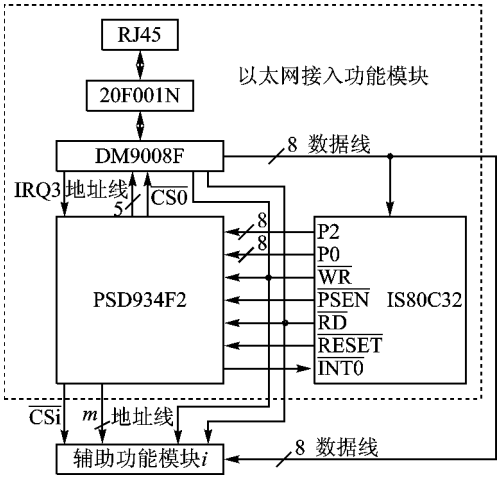


图4.5-4 嵌入式设备网关的硬件组成

电源管理功能、在线/在系统编程功能和程序代码保护功能。Flash存储器可擦写10 000次以上,PLD至少可擦写1 000次,数据可以保存15年。

DM9008F与PSD934F2之间的连线,只需连接DM9008F的低5位地址线、中断请求线和片选信号线即可。DM9008F剩余的高位地址线,可根据其指定的I/O基址来选择接地还是接高电平。由于IS80C32的外部中断触发是低电平有效,而DM9008F的中断请求是高电平有效,因此,需要把DM9008F的中断请求线IRQ3接到PSD934F2,经PSD934F2反相后再连接到IS80C32的$\overline{\text{INT0}}$。

采用PSD不但可以简化片外电路,而且还可以很方便地设置I/O端口地址和分配内存空间。这些都通过软件实现,便于系统功能的扩展。

由于DM9008F工作在8位模式,必须按用户手册把一些引脚接入地或高电平。BALE、SYSCLK、$\overline{\text{SMEMR}}$、$\overline{\text{MEMW}}$和$\overline{\text{MEMR}}$接高电平,AEN接片选信号线$\overline{\text{CS0}}$。注意将引脚70(MD6/SLOT)悬空,以保证DM9008F工作在8位模式。另外,由于不用EEPROM,所有设置都采用缺省值会方便些。如I/O基址用缺省的300H,SA_{10} ~ SA_{19}、SA_6和SA_7接地,SA_8和SA_9接高电平;中断请求线直接用缺省的IRQ3。

2. 软件编程

驱动程序的结构以上述程序流程图为基础,具体编程时应注意以下问题:

① 由于没有EEPROM,必须直接往寄存器写入MAC地址。

② 以太网控制器的初始化过程应严格参照DP8390数据手册[1]所给出的步骤进行。

③ 由于以太网控制器与单片机之间的数据读/写,都是通过Remote DMA完成的,因此在读/写的操作没有完成之前,不能进行新的读/写操作,否则会使以太网控制器停止工作。所以在发送数据时,即单片机向以太网控制器写数据,不能被以太网控制器中断服务程序中断。

④ 在接收数据过程中,当读取以太网控制器数据完成之后,不要查中断状态寄存器ISR的RDC位。尽管有些NE2000兼容以太网控制器的数据手册[5]要求查询该位状态,但实验表明,这种操作可能会造成以太网控制器长时间的停顿,甚至停止工作。

三、结束语

根据上述方法所完成的以太网接入系统，性能稳定，主机时钟频率为 11.059 2 MHz 时，其实际数据收发速率大约为 240 Kbps，能够满足基本的嵌入式设备网关的要求。上述连接方法的探讨，不仅为实现完整的 TCP/IP 协议提供了良好的基础，而且可作为后面设计性能更高的网络化嵌入式系统的借鉴。

参 考 文 献

1 DP8390D/NS32490D NIC Network Interface Controller Datasheet. National Semiconductor，1993

2 DM9008 ISA/Plag & Play Super Ethernet Controller Datasheet. Davicom Semiconductor，2000

3 RTL8019AS Realtek Full－Duplex Ethernet Controller with Plug and Play Function Datasheet. Realtek Semiconductor，1996

4 CS8900A Crystal LAN™ ISA Ethernet Controller Datasheet. Cirrus Logic，Inc. 1999

5 Application Note 874：Writing Drivers for the DP8390 NIC Family of Ethernet Controllers. National Semiconductor，1995

6 PSD9XX Family Datasheet. Waferscale Integration Inc. 2000

选自《单片机与嵌入式系统应用》月刊，2002 年第 7 期

4.6　RS485 总线通信避障及其多主发送的研究

同济大学电子与信息工程学院智能控制研究室 (200092)
吴军辉　林开颜　徐立鸿

在工业现场控制中,由于控制对象比较分散,往往采用集中管理、分散控制的集散式控制方式,具体地讲就是根据现场情况分解成多个数据采集级或直接控制级,根据控制过程的复杂性分解成多层递阶结构,即现场控制层、监视和综合管理层,这就构成了一个集散控制系统(DCS,Distributed Control System)。由于串行通信使用的传输线较少,在长距离通信时比较经济。在多种串行接口标准中,RS485 接口以其结构简单、通信速率高、传输距离远等诸多优点,在集散式工业控制系统中得到了广泛应用。

RS485 总线的数据通信大多是半双工的,在整个网络中任一时刻只能有一个节点成为主节点处于发送状态并向总线发送数据,其他所有节点都必须处于接收状态。如果有两个节点或两个以上节点同时向总线发送数据,将会导致所有发送方的数据发送失败。因此,解决其总线数据传输的冲突,即通信避障,就成了提高其工作可靠性、稳定性的关键和前提。

一般情况下,整个系统由一个主节点、多个辅节点构成。数据通信一般采用主节点轮巡各辅节点的方式,或总线控制权分时交由各个节点,从而各个节点分时成为总线的主节点。这种工作方式在很多工业控制场合有较大的局限性。诸如不能满足数据通信实时性的要求等。笔者对多主节点的通信作了一定的研究,实践表明应用了这些方法之后 RS485 总线数据通信的可靠性、稳定性有很大的提高,并较好地解决了系统数据通信实时性的要求。

一、RS485 总线接口的主要特性

RS485 接口大多连接成半双工通信方式[1],如图 4.6－1 所示。RS485 半双工通信的芯片有 SN75176、SN75276、SN75LBC184、MAX485、MAX1480、MAX1487、MAX3082、MAX1483 等。

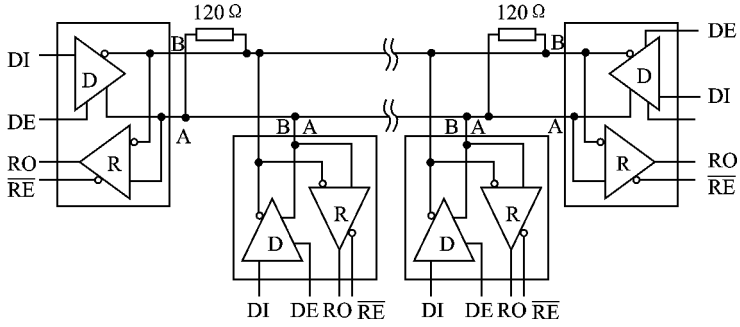


图 4.6－1　RS485 半双工通信方式

其主要特性如下：

① 传输方式：差分；

② 传输介质：双绞线；

③ 标准节点数：32；

④ 最远通信距离：1 200 m；

⑤ 共模电压最大、最小值：+12 V、- 7 V；

⑥ 差分输入范围：- 7 ~ +12 V；

⑦ 接收器输入灵敏度：±200 mV；

⑧ 接收器输入阻抗：≥12 kΩ。

以 TI 公司的 RS485 接口芯片 SN75176[2] 为例，其真值表如表 4.6－1 所列。收发器的逻辑框图如图 4.6－2 所示。

表 4.6－1 (a) RS485 发送器真值表

输入端	使能端	输出端	
D	DE	A	B
H	H	H	L
L	H	L	H
	L	Z	Z

表 4.6－1 (b) RS485 接收器真值表

差分输入	使能端	输入端
A - B	$\overline{RE}$	R
$U_{ID} \geq 0.2$ V	L	H
- 0.2 V < U_{ID} < 0.2 V	L	?
$U_{ID} \leq$ - 0.2 V	L	L
X	H	Z
输入端开路	L	H

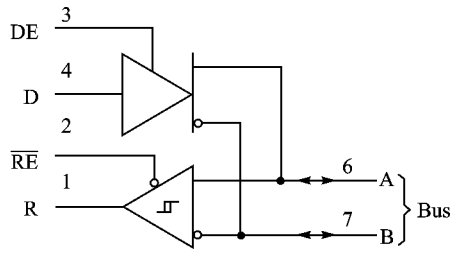


图 4.6－2 SN75176 逻辑框图

二、通信避障和多主发送的研究

1. 避免异常总线冲突

在 RS485 总线构筑的半双工通信系统中，在整个网络中任一时刻只能有一个节点处于发送状态并向总线发送数据，其他所有节点都必须处于接收状态。如果有 2 个节点或 2 个以上节点同时向总线发送数据，将会导致所有发送方的数据发送失败。因此，在系统的各个节点的硬件设计中应首先力求避免因异常情况而引起本节点向总线发送数据而导致总线数据冲突。

以 MCS51 系列的单片机为例，因其在系统复位时，I/O 口都输出高电平。如果把 I/O 口直接与 RS485 接口芯片的驱动器使能端 DE 端相连，会在 CPU 复位期间 DE 为高，从而使本节点处于发送状态。如果此时总线上有其他节点正在发送数据，则此次数据传输将被打断而告失败，甚至引起整个总线因某一节点的故障而通信阻塞，继而影响整个系统的正常运行。为了做到通信避障，考虑到系统工作的稳定性和可靠性，在每个节点的设计中应将控制 RS485 总线接口芯片的发送引脚设计成 DE 端的反逻辑。即控制引脚为逻辑“1”时，DE 端为“0”；控制

引脚为逻辑“0”时，DE 端为“1”。应用中，将 CPU 的 I/O 引脚 DEO（控制 RS485 总线接口芯片的发送引脚）反相和 CPU 复位引脚 RESET 反相相与驱动 DE 端，这样就可以使控制引脚为高或者异常复位时使 RS485 接口始终处于接收状态，从而从硬件上有效避免节点因异常情况而对整个系统造成的影响。这就为整个系统的通信避障和实现多主发送奠定了基础。其示例接口电路如图 4.6－3 所示。

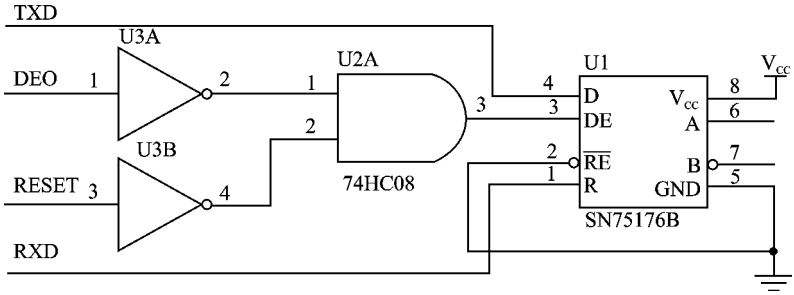


图 4.6－3 与 CPU 接口电路图

2. 通信避障和多主发送

在 RS485 总线构筑的半双工通信系统中，一般采用主从通信模式，即整个系统中只有一个节点为主节点，总线上的所有其他节点都是从节点。通信方式一般是主节点循环轮询（POLL）各个从节点。各个从节点都有自己的网络通信识别号，当主节点的轮询信息中包含自己的网络通信识别号时，此从节点则对此帧进行应答，其他节点则忽略此帧，不做任何处理。这种软件模式一般不会引起总线数据冲突，但其也存在着一些局限性。

⑴ 主节点通信负担较重。在系统工作的整个流程中，主节点都担负着循环轮询各个从节点的通信工作，其 CPU 通信负担较重。一定意义上讲，会对其完成其他系统工作造成一定的影响。

⑵ 整个系统“危险”集中。在整个系统中，主节点的工作可靠性和稳定性是整个系统稳定、可靠工作的前提和保证。一旦主节点发生故障，将导致整个系统的崩溃。

⑶ 系统通信效率较低。因为无论某一个从节点是否需要发送数据，或需要使用总线，都要等到主节点循环轮询到自身，从而使得系统的通信效率较低，总线的利用率也较低。

⑷ 系统实时性较差。因其通信的效率较低，使得从节点有实时性要求的帧信息得不到及时发送，从而使得系统的实时性能力较差。

针对这些局限性，笔者在 RS485 总线构筑的半双工通信系统中引入多主发送的通信方式，即在整个系统中，所有节点在总线空闲时都能成为总线上的主节点而向总线发送数据。采用这种工作方式后，系统中已经没有主、从节点之分，各个节点对总线的使用权限是平等的，从而有效避免了个别节点通信负担较重的情况。总线的利用率和系统的通信效率都得以大大提高，从而也使系统响应的实时性得到改善。而且即使系统中个别节点发生故障，也不会影响其他节点的正常通信和正常工作，这样使得系统的“危险”分散了，某种程度上来说增强了系统的工作可靠性和稳定性。

3. 多主发送的实现

当一个节点需要使用总线时，为了避免实现总线通信避障。在有数据需要发送的情况下侦听总线。在硬件接口上首先将 RS485 接口芯片的数据接收引脚通过一反相器接至 CPU 的

中断引脚 INT0。其电路如图 4.6－4 所示。当总线上有数据正在传输时，RS485 接口芯片的数据接收端表现为变化的高低电平。利用其产生的 CPU 下降沿中断，可以得知此时总线是否正“忙”，即总线上是否有节点正在通信。如果侦听到总线上在一段时间内（这段时间可以根据系统的通信协议的应答时间而定）“空闲”，则可以得到对总线的使用权限。如果恰巧此时有不少于 2 个以上的节点同时使用总线（可以通过发送与接收的数据是否一致侦测出），则同时让出总线进行避障。然后在一段随机延时后，再重新侦听。这样就能解决因为多主而带来的总线冲突的问题。在此基础上还可以定义各种消息的优先级，使高优先级的消息得以优先发送，从而进一步提高系统的实时性。

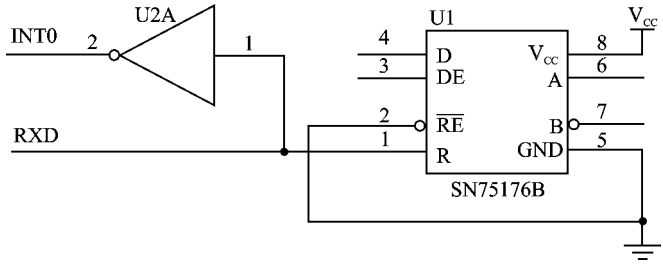


图 4.6－4 多主发送硬件电路

以 MCS51 系列单片机为例，给出侦听总线“忙”否的程序示例。以下是编程实现的 C 语言源代码，在 Keil C 6.20 [3] 调试通过。

```
void interrupt_timer0 (void) interrupt 1                    //中断 T0
{
if (need_test_bus = = true)
{
    counter + + ;
    if (counter > TIME_SET)
    {
      need_test_bus = false ;                               //侦听时间结束
    }
  }
}
void interrupt_int0 (void) interrupt 0                      //中断 INT0
{
    test_bus_flag = false ;                                 //总线有数据传输
}
void main (void)
{
    …
      //开始侦听总线
      test_bus_flag = true ;
      need_test_bus = true ;
      while (need_test_bus = = true)
```

```
{
}
//侦听时间结束
if (test_bus_flag = = true)
{//总线空闲可以发送数据
}
else
{//总线正忙,进行相关处理
}
…
}
```

三、结束语

测试及现场应用表明,采用了 RS485 总线避障和多主发送的通信方式后,整个系统的工作可靠性较高、稳定性较好,并且大大提高了系统的实时响应能力。这种方法也同样可以用于很多工业测控领域,或对实时性有较高要求的应用场合。

参 考 文 献

1 B&B Electronics. RS-422 and RS-485 Application Notes [EB/OL]. http://www.bb-elec.com/,2001

2 Texas Instruments. SN65ALS176, SN75ALS176, SN75ALS176A, SN75ALS176B DIFFERENTIAL BUS TRANSCEIVER [EB/OL]. http://www.TI.com/,1999

3 Keil Software. Cx51 Compiler Optimizing C Compiler and Library Reference for Classic and Extended 8051 Microcontrollers [EB/OL]. http://www.keil.com/,2001

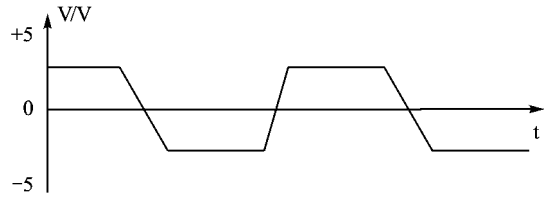
选自《测控技术》月刊,2002 年第 8 期

4.7 RS422/RS485 网络的无极性接线设计

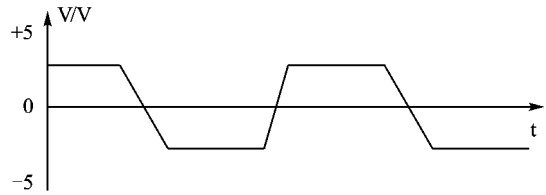
合肥工业大学 彭良清 洪占勇

一、问题提出

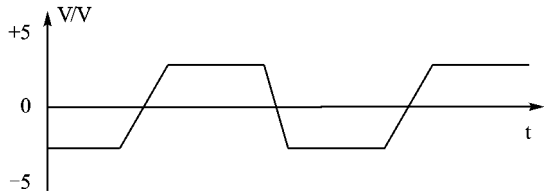
现在的很多测控系统是使用 RS422 或 RS485 总线互连的。RS422/485 总线信号是由 4 (2) 根有极性的差分信号来传输的,不能将其反接。当网络传输距离长或节点多时,在线路上的分续线盒也会很多,很容易将信号线在传输途中接反,从而造成信号无法正常传输。虽然可以查出故障点,但在分线盒很多时,也是一件很费时的事情。为了布线方便,分续线盒的数量往往大于总线上的模块数。对于室内系统,网络线路一般外加 PVC 线槽甚至暗埋于墙体内;对于室外系统,线路一般架空或地下走线,造成对线路反接问题的查找和修正很困难。另一方面,为了施工方便,也应允许在途中随意接线,不分极性。为此,需要各模块既能接收图 4.7-1 (b) 所示的正相信号,也能接收图 4.7-1 (c) 所示的极性可能反相的 RS422、RS485 信号。



(a) V_A-V_B发送端波形



(b) V_A-V_B同相波形(接收端)



(c) V_A-V_B反相波形(接收端)

图 4.7-1 传输线反接错误所引起的接收端信号波形改变

对于那些采取未经任何编码调制的基带信号来传输数据的 RS422/485 系统，图 4.7－1 中由于接线错误将造成收信方无法正确接收数据；但如对信号进行适当的调制后，即使途中出现接线错误，收方仍然能正确接收到数据，即在布线施工中可以无极性布线。

下面分别给出使用未调制信号和调制信号传输数据 2 种情况下的无极性接线设计方法。先讨论使用未编码调制信号的情况。

二、RS422 信号线的无极性接线设计

RS422 总线使用收发分开的信号线传输，各为 2 根信号线。为了使 RS422 接收器能够接收总线上传来的 2 种极性的信号，见图 4.7－1 (b) 和图 4.7－1 (c)，首先要检测到接线的错误，其次才是更正接线错误。这里希望通过网络模块电路来修正接线错误，而不是通过更正错误的传输线连接。

1. 人工修正方法

对于 MCU 的 UART 来说，无信号传送时，TX 引脚为“1”电平，因此，RS422 驱动器的 A 端会为高电平，B 端为低电平，此时，在接收模块的 A 端也应为固定高电平。图 4.7－2 电路中，在一个接收端和 GND 之间连接一个 LED，从而可以据此判断该端是和发送端 A 相连，还是和发送端 B 相连，然后通过 4 位拨码开关 SW 来人工调整模块总线和接收模块中 RS422 驱动器接收输入的连接。

人工修正方法需要发送模块在软件上进行配合。在调整时，发送方不能发送数据，也就是总线上的差动电压为固定的。这种方法，虽然有些麻烦，但在一些情况下，比起检查和修正线路来还是要简便一些。

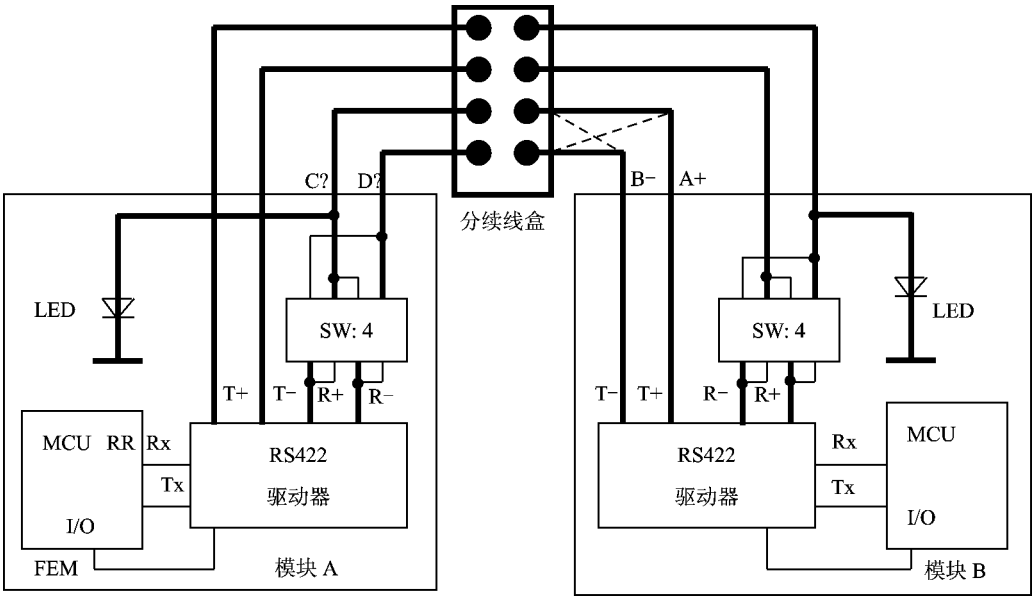


图 4.7－2 RS422 总线传输中的无极性接收电路

2. 自动修正方法

如果在总线输入端子和 RS422 驱动器之间用电磁继电器（或模拟开关）代替拨码开关

SW,就可以通过软件来自动控制总线 A、B 端的切换。检查是否存在错误接线也可通过软件进行,只要发送端发送一个和收方约定的固定内容数据。如果收方不能正确收到,则表明接线错误,就控制继电器切换总线连接;否则,不切换。必须注意的是,使用模拟开关时,应注意对线路阻抗和传输速度的影响。

三、RS485 信号线的无极性接线设计

RS485 总线中的发送和接收信号共用一对线,使用的驱动器可分为两类:一类是像 SN75176 之类的驱动器,在驱动器内部已经将 Rx 和 Tx 信号接到一起;另一类是使用 RS422 方式的驱动器,如发送和接收使用 2 个芯片,如 SN75177(接收驱动器)加 SN75178(发送驱动器),或者收发驱动器集成在 1 个芯片上,如 SNLBC75179、MAX488、MAX490 等,这种情况下,在线路板上将收发的同相端短接。对于后者(使用收发引脚独立的驱动器),无极性设计的方法仍然类同于 RS422 方式;对于前者,由于收发信号的同相端在驱动器内部已经短路,无法在接收驱动器增加电路,不能达到无极性信号传输的目的。

可见,在 RS485 网络中,模块必须使用独立收发引脚的驱动器时,才能增加无极性设计电路。

四、使用限制

以上方法只适合于点对多点的主从式 RS422/485 网络。对于 RS422 网络来说,在主模块中的接收驱动器不能加修正电路,而应调整到发送模块的发送端。因为在从模块发送而主模块接收的情况下,可能部分模块和主模块之间的连接正确,部分模块和主模块之间的连接错误。对于 RS485 网络来说,只要在从模块的驱动器接收端增加调整电路就可以了。

对于各模块平等通信的 RS422/485 网络来说,一个模块可能和其他模块之间的接线既有正确,又有错误,因此通过此方法来修正。

五、采取调制信号传输消除信号极性

使用以上 2 种(手动设置或软件自动配置)使模块可以接收任意极性信号的方法虽然可行,但仍然有一些麻烦:手动设置仍然会带来施工的不便,而自动配置会增加软件设计的复杂度,降低了可靠性。此外,以上方法也只适用于点对多点的主从通信网络,对于节点对等网络不能使用。

另外一种消除信号极性的方法就是在对信号编码调制后传送,使调制后的信号是无极性要求的。在数据传输领域,最常用的无极性信号调制方法是使用差分曼彻斯特编码,其波形如图 4.7-3 所示。

差分曼彻斯特编码信号的编码原则是:

- 在信号位中间总是将信号反相;
- 在信号位开始时不改变信号极性,表示逻辑“1”;
- 在信号位开始时改变信号极性,表示逻辑“0”。

由此可见,经差分曼彻斯特编码的信号,见图 4.7-3(b),经过由于接线错误变成反相的波形后,见图 4.7-3(c),仍然符合此定义,从而可以解调出原始数据信号。

为了在 RS422/485 网络中实现差分曼彻斯特编码,需要在 UART 和 RS422/485 芯片之间

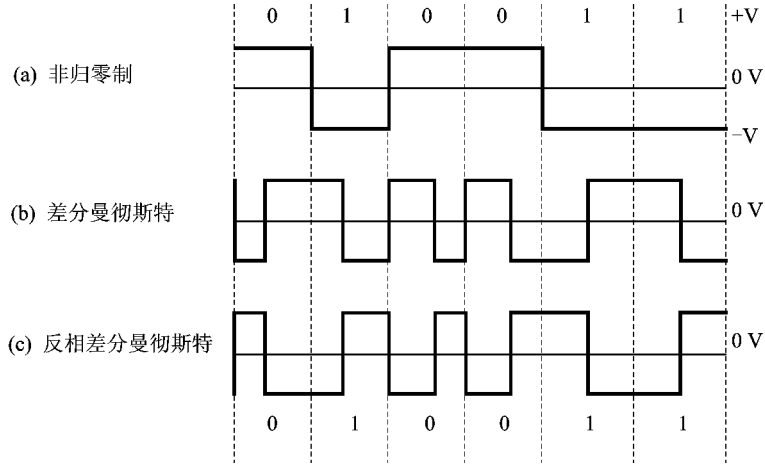


图 4.7－3 差分曼彻斯特编码信号及其反相信号和原始数据关系波形

增加编码电路。差分曼彻斯特编码属于自同步编码,因此需要时钟。对于工作于异步方式的 UART 来说,可以使用 GAL 器件完成编码和解码,但用于控制 UART 异步传输的波特率时钟和编码电路时钟必须使用同一时钟源。以下给出图 4.7－4 所示的实现框图,具体实现电路这里不再详细叙述。也可以使用专用芯片完成编码和解码,比如采用 Echelon 公司的 FTT - 10A 收发器。该收发器对信号进行差分曼彻斯特编码调制后传输(同时包含一个隔离变压器)。

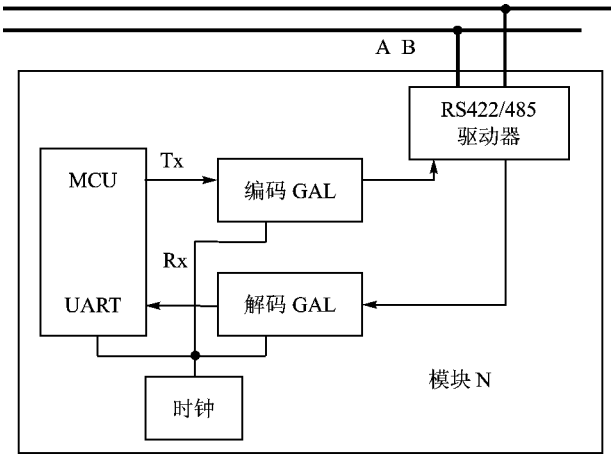


图 4.7－4 使用差分曼彻斯特编码产生无极性信号

六、直流供电的无极性接线设计

在 RS422/485 网络中,常采用集中 +5 V、+12 V 或 +24 V 直流对所有模块进行供电,如线路较长,一般使用 +24 V 电源,较短时使用 +5 V 或 +12 V 电源。同信号线一样,电源线也同样存在反接问题,基于同样目的和原因,模块也应能使用正相和反相接线 2 种情况的输入电源。和信号不同,2 根电源线虽然可能反接,线间的电位差始终是一个极性,要么为正,要么为负。因此,可以在模块电源输入处增加一个整流电桥,在电桥的输出端就始终能得到正极性的

+24 V 或 +12 V、+5 V 电压供自己使用了，如图 4.7－5 所示。

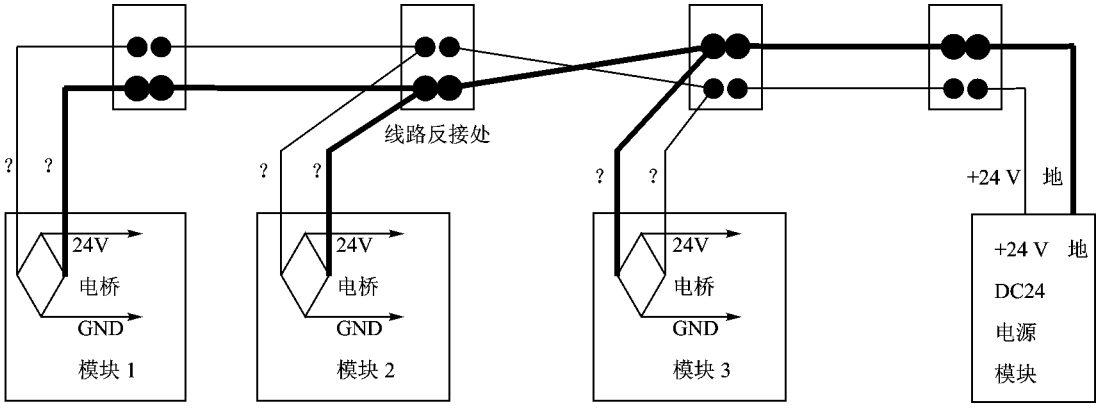


图 4.7－5 集中直流供电网络的无极性电源接收电路

参 考 文 献

1 Data Encoding. http ∶//www. cpe. ku. ac. th

2 阳宪惠主编. 现场总线技术及其应用. 北京 ∶清华大学出版社，1999

选自《单片机与嵌入式系统应用》月刊，2002 年第 6 期

4.8　RS485 与 USB 接口转换卡的设计与实现

浙江大学（310027）　孟祥育　毛雪珍

一、概　述

USB（通用串行总线）具有速度快、即插即用、易扩展、可总线供电以及多种传输模式等优点。但是 USB 的传输距离通常不超过几十米，这对工业应用而言是太短了。

目前，工业现场有大量采用 RS485 传输数据的采集设备。RS485 总线由于采用平衡传输技术，传输距离可以达到 1 200 m 以上，并且可以挂接多个设备。不足之处是传输速度慢、可靠性差、需要板卡的支持、成本高、安装麻烦等。RS485 的这些缺点恰好能被 USB 所弥补，而 USB 传输距离的限制又是 RS485 的优势所在。如果能将两者结合起来，优势互补，就能够产生一种快速、可靠、低成本的远距离数据采集系统。

二、PDIUSBD12 简介

PDIUSBD12 是 Philips 公司生产的一款 USB 接口芯片，它完全符合 USB1.1 规范。与 EZ－USB 等内部集成处理器的 USB 器件不同，PDIUSBD12 只实现 USB 接口，成本较低，外接普通 MCU（微控制器）进行控制，用户编程更方便，所以我们选用该芯片作为 USB 接口。

PDIUSBD12 的内部框图如图 4.8－1 所示。

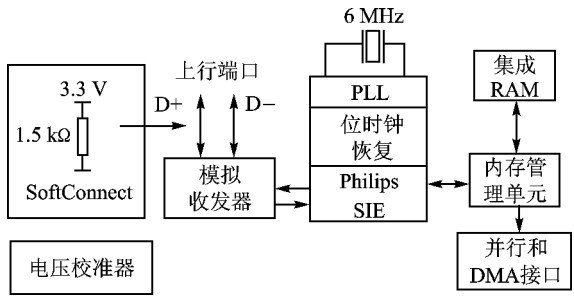


图 4.8－1　PDIUSBD12 内部结构框图

PDIUSBD12 主要包括如下模块：

⑴ 模拟收发器　集成的收发器直接通过终端电阻与 USB 电缆连接。

⑵ 电压校准器　集成的 3.3 V 电压校准器为模拟收发器供电，也提供连接到外部 1.5 kΩ 上拉电阻的输出电压。

⑶ PLL　集成的 6～48 MHz 倍频锁相环，允许使用 6 MHz 的晶振，电磁干扰也由于使用低频晶振而减小。

⑷ 位时钟恢复　位时钟恢复电路用 4 倍过采样原理从输入的 USB 数据流中恢复时钟，能跟踪 USB 规范中指出的信号抖动和频率漂移。

⑸ 全硬件的 SIE (串行接口引擎) Philips 公司的 SIE 完全实现 USB 协议层,采用全硬件实现,提高了处理速度。

⑹ SoftConnect 内部集成的 D + 上拉电阻与 VDD 的连接可由 MCU 来控制。这使得 MCU 可以在 USB 连接建立之前完成自身初始化,重新初始化 USB 总线连接也可以不用物理插拔来实现。

⑺ GoodLink GoodLink 为 D12 的调试提供了方便。D12 枚举时,LED 指示灯将立即闪亮;D12 被成功枚举并配置后,LED 指示灯将会始终亮;D12 在数据传输过程中,LED 将一闪一闪;传输成功后 LED 熄灭;挂起期间,LED 熄灭。

⑻ 存储器管理单元 (MMU) 和集成 RAM 缓冲 USB 数据传输和 MCU 间并行接口的速度差异,使得 MCU 可以以自己的速度读写 USB 包。

⑼ 并行和 DMA (直接存储器存取) 接口 并行接口容易使用、速度快并且能直接与主微控制器连接。同时,在主端点 (端点 2) 和局部共享存储器之间也可使用 DMA 传输。

三、转换卡的原理与应用

RS485 与 USB 转换卡主要面向远程数据采集传输系统,其原理如图 4.8 - 2 所示。在采集现场,数据采集设备将采集的数据利用 485 总线进行传输。在主机端,利用转换卡接收来自 485 总线的数据并通过 USB 接口传送至 PC 进行分析处理。而主机向设备发送数据的过程正好相反,主机向 USB 口发送数据,通过 485 ~ USB 转换卡转换为 485 协议向远端输送。

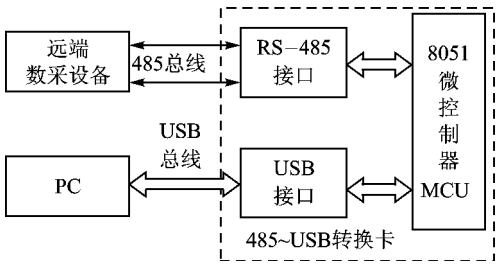


图 4.8 - 2 485 ~ USB 转换卡框架

转换卡的主要功能是完成信号电平转换和数据格式转换。RS485 和 USB 电平的转换主要依靠硬件,通过各自的接口芯片将信号转换为 TTL 电平。数据格式的转换则主要是软件方面的工作,MCU 读到某一个接口的数据,然后按照另一个接口协议中对数据帧的要求进行打包。设计的核心是通过 MCU 协调两个接口的数据收发,由于相当部分的工作已经由各个接口电路完成,所以设计工作量大为减少。

USB 总线是一种高速的串行总线,USB1.0 的速度已经达到 12 Mbit/s,远远高于 485 总线的数据传输速率。同时,USB 总线易于扩展,一个 USB 口理论上可以连接 127 个 USB 设备。图 4.8 - 3 就是充分利用 USB 的这些特性,基于 485 ~ USB 转换卡的分布式数据采集系统。这种数据采集系统适用于多点多信号量的远距离采集,其中 D 表示某一个信号量的采集设备。随着 USB2.0 的出台,USB 传输速度高达 480 Mbit/s,这种采集系统的优势将变得更加明显。

四、硬件设计

硬件部分主要是 MCU 与 485 和 USB 的接口电路设计,如图 4.8 - 4 所示。MCU 采用普通

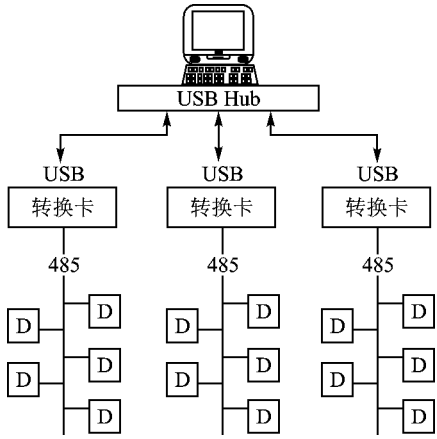


图 4.8－3 485～USB 转换卡组成分布式采集系统

8051，USB 接口芯片为 PDIUSBD12，485 接口芯片为 MAX485。

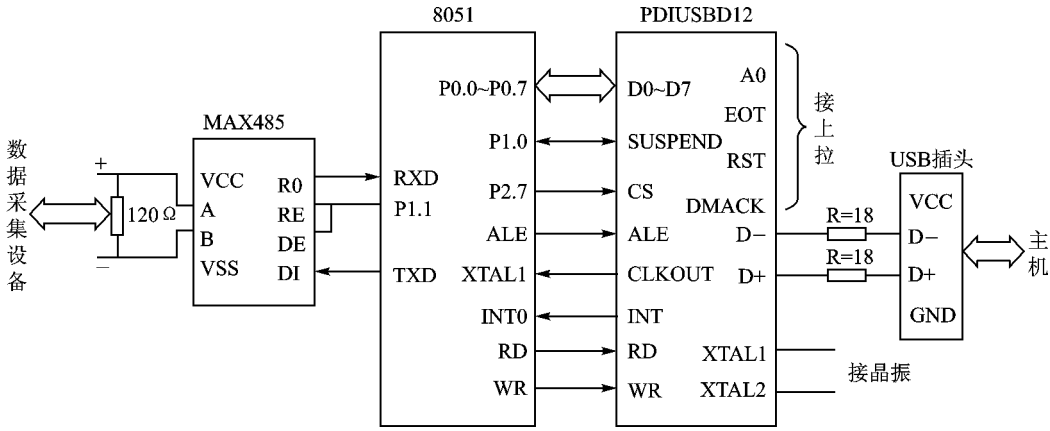


图 4.8－4 485～USB 转换卡接口电路

8051 和 D12 的连接有独立的地址数据总线方式和复用的地址数据总线方式。由于 D12 既要接收来自 8051 的命令又要跟 8051 进行数据通信，而它们使用共同的接口 D[0..7]。所以 D12 地址的含义是对命令和数据的选择，理解这一点对 D12 的正确使用非常重要。在第一种方式中，用 A0 脚表示地址位：A0 为 1 表示命令，A0 为 0 读写的是数据，将它与 8051 的一 IO 口相连。在发送命令（数据）前先对 A0 进行置 1（0），然后再把命令（数据）的内容送到数据总线上。此时，D12 的 ALE 脚未使用，可直接接地。图 4.8－4 中为第二种连接方式，命令和数据使用不同的地址，地址字节中仅 LSB 位具有实际意义。对偶数地址赋值表示送往 D12 的是读/写数据，对奇数地址赋值表示往 D12 写入一个命令。这种接法中，D12 的 ALE 与 8051 的 ALE 相连，ALE 的时序关系与 8051 跟一般存储器相连时相同，在下降沿对地址锁存。此时，A0 不使用，应接高电平。8051 的 P0 口直接与 D12 的数据总线相连，作为并行数据和命令传输通道。它的时钟可直接从 D12 的 CLKOUT 接入，而不需外接晶振。

五、软件设计

软件设计包括单片机软件（固件）设计和主机部分软件设计。单片机软件使用伟福公司

提供的 WAVE 仿真软件开发,并通过其仿真器进行在线调试。主机部分软件又包括驱动程序和应用程序两部分,分别使用 DDK 和 VC6.0 生成。另外,在开发过程中还用到 USB 总线数据通信查看工具,如 USBView、SniffUSB 以及驱动程序调试工具 SoftICE 等。

1. 单片机软件设计

下位机程序主要的工作是初始化 D12,通过 D12 进行数据传输以及按协议对数据进行格式转换。USB 单片机控制程序通常由三部分组成:第一,初始化部分,完成单片机和所有的外围电路(包括 D12)的初始化;第二,主循环部分,等待来自数据采集设备或上位机的数据,并启动数据格式转换程序,进行数据通信,是固件的主体部分;第三,中断服务程序,由上位机和数据采集设备触发,进行一些低工作量的实时处理(如置相应标志位),然后在主循环部分对数据作进一步的处理。以笔者的经验,如何成功初始化是初次使用 D12 的最大挑战,这一阶段主要是理解 D12 的工作方式以及进一步理解 USB 协议。初始化成功后,对 D12 的工作过程已相当熟悉,以后的开发就比较顺利了。

D12 的初始化过程如下:

⑴ 设置地址使能;

⑵ 设置端点(EndPoint)使能(这时候 LED 亮);

⑶ 软断开(Disconnect);

⑷ 延时(1 ~ 2 s);

⑸ 软连接(Connect,用 Set Mode 命令,此时 LED 灭);

⑹ 中断使能,等待中断;

⑺ 响应来自主机的 Setup 包,完成枚举。

步骤 3 ~ 5 就是利用 SoftConnect™技术,不必进行物理插拔而使主机初始化 USB 总线。虽然 USB 协议对枚举过程作了统一的规定,但是不同公司的芯片实现起来可能有所不同。USB 枚举的过程实际上就是主机和 USB 设备的一个握手过程:主机发送出包含某个一枚举请求的 Setup 包,USB 设备响应该请求并返回必要的信息。在主机得到 USB 通信所需要的所有 USB 设备的信息之后,枚举即告结束。

D12 构成的 USB 设备的枚举过程如下:

⑴ GetDeviceDescriptor 主机请求代码为 80 06 00 01 00 00 40 00,然后 8051 通过 D12 发送设备描述符,第一次只需发送设备描述的前 8 个字节,如:12 01 00 01 00 00 00 10。

⑵ SetAddress 主机请求代码为 00 05 02 00 00 00 00 00,说明主机设置其地址为 0x02,收到该请求后 D12 只需使能该地址(0x82),并对控制输入端写 0 长度的数据。

⑶ 读取全部 DeviceDescriptor 主机请求代码为 80 06 00 01 00 00 12 00,与 1 不同的是此时是读取全部设备描述符一,一般为 18 个字节,可以分为多次传输,D12 发送的前 8 字节与 1 相同,后 10 个字节为:71 04 88 88 00 01 00 00 00 01。其中,前两个字节是厂商 ID(VID),本例中的为 0x0471,即分配给 Philips 公司的 ID 号。后两个字节是设备 ID(PID),设计定义为 0x8888。VID 和 PID 决定了驱动程序的匹配,一定要与最后生成的主机驱动程序一致。

⑷ GetConfigDescriptor 主机请求代码为 80 06 00 02 00 00 09 00,根据 USB 协议的定义,第四字节的 0x02 表明该请求是一配置描述符请求。D12 发送 9 字节的配置描述符给主机,为 09 02 2e 00 01 01 00 60 01。

⑸ 读取全部 ConfigDescriptor 主机请求代码为 80 06 00 02 00 00 12 00,此时 D12 必须

把包括配置描述符、接口描述符、各端点 (D12 为四个) 的描述符在内的所有的配置情况分多次发送给主机。

(6) 如果以上步骤都正确,主机将找到新设备,提示安装驱动程序,否则找到未知设备,不可用。安装驱动程序后,以后的每次设备插入,枚举次序与以上步骤略有不同,之后会有 SetConfiguration、GetConfiguration 和 GetInterface 等调用。

以上各步 D12 对主机请求的响应要严格根据 USB 协议对每个字节的定义进行设置,具体设计中可能与给出的参考不同。发送正确的设备描述符给主机是成功进行枚举的必要条件。在具体编程时,如果把各描述符定义为存储在程序存储器的字符数组,把对各请求的响应定义为不同的函数,会给编程和调试带来极大的方便。

2. 主机方面软件设计

Windows 2000 提供了一些常见 USB 设备的驱动程序,但是要使 D12 构成的 USB 设备正常工作仍需要自己编写驱动程序。尽管系统已经提供了很多标准接口函数,但编制驱动程序仍然是 USB 开发中最困难的,通常采用 Windows DDK 来实现。目前有许多第三方软件厂商提供了各种各样的生成工具,像 Computeware 的 DriverStudio, Blue Waters 的 DriverWizard 等,它们能够很容易地在几分钟之内生成高质量的 USB 驱动程序。笔者即是采用 DriverStudio 软件生成 USB 驱动程序的框架,然后在此基础上添加用户代码,就可以生成适用于本设计的驱动程序。

当主机检测到有 USB 设备插入时,根据 PID 和 VID 查找相应的驱动程序。如果是第一次插入,系统会提示安装驱动程序,此时把驱动程序的 Inf 文件指定。通过操作系统的控制面板可以检查驱动是否安装成功,不成功时要检查 Inf 文档以及驱动程序代码,看是否存在错误。更深入的查错就要借助如 SoftICE 这样的工具进行跟踪。主机应用程序主要是用来进行数据显示和处理以及发出数据采集命令等,它跟下位机的通信通过驱动程序进行,可以用 VC 等高级编程工具来开发。

六、结束语

基于 PDIUSBD12 的 485 ~ USB 的转换卡电路简单、成本低,可以取代一般的 485 卡进行数据采集。由于采用 USB 接口与主机进行通信,具有安装方便、不受插槽数限制、不用外接电源、可多路采集等优点,为工业和科研数据采集提供了方便、廉价、有效的途径。

参 考 文 献

1 Don Anderson. USB 系统体系. 北京 :中国电力出版社,2001

2 Philips Corp. PDIUSBD12 Users Manual, 2001

3 王朔等. USB 接口器件 PDIUSBD12 的接口应用设计. 单片机与嵌入式系统应用,2002 (4)

4 刘丁等. USB 在数据采集系统中的应用. 电子技术应用,2002 (4)

选自《电子技术》月刊,2002 年第 12 期

4.9 低压电力线载波数据通信及其应用前景

湖南省计算技术研究所 (410012) 王书亮 苟新兵 卢新华 周 钢

一、引 言

电力线载波通信 (PLC) 并不是一项新技术,它出现于上世纪 20 年代。在我国的应用也已有半个多世纪的历史了。利用高压输电网进行载波通信早已被电力部门用于电网管理和话音通信,并有包括加工设备 (高频阻波器等)、结合设备 (耦合电容、结合滤波器等) 及载波机等在内的成套设备供应。

而利用低压输电线路组成数据通信网络,则是近十几年迅速发展的一门新技术。

电力供电网络四通八达,而低压电力线连接千家万户,构成最普及的网络。利用它进行数据通信,传递各种信息,不占用无线频道资源,亦无需铺设专用通信线路,省工、省钱、维护简单、无疑有着美好的前景。

这项技术越来越受到计算机与通信界专业人士的高度重视,并成为 IT 业的又一个新的研究热点。

二、低压电力线载波信道特点

在电力线上搭载高频 (或低频) 调制信号,不是一件轻而易举的事情。对于数据通信而言,电力线远不是一种理想的载体。

这是因为,对于低压电力线上的干扰特性,阻抗变化及信号衰减情况,很难找到一个较为明确的解析式或数学模型加以描述,这些对通信系统的设计提出了很高的要求,也是严重影响通信质量的主要技术障碍。

(1) 电力线上除有 50 Hz 正弦波交流电干扰外,还存在着不同来源的多种电磁干扰,如雷电干扰、电火花及可控硅通断时产生的干扰;负载特别是感性负载投入或断开瞬间的干扰,都会对电力线上所传输的信号产生严重影响。

电力线上存在的随机干扰具有极强的时变性。突发性的干扰可引起瞬间的高误码率,我们在通信实验中曾因实验楼近旁锅炉旁一台故障电机运转时所产生的电火花干扰,导致通信几近中断的情形。这种干扰往往是能量很大的脉冲干扰或脉冲干扰群,持续时间较短,但能量很集中,频谱也很宽。

电力线上周期性的干扰也有较高的强度,有时可达 10 V (峰-峰) 以上。图 4.9-1 显示了实验室内某一时刻电力线上周期性干扰波形。

产生这种周期性干扰的原因是由于许多用电设备会在工频交流电基波的某个固定相位释放干扰。

由于我国电器上网的电磁兼容性没有欧美控制的严格,因此,我国电力网上的干扰要比欧美严重得多,电力线上的通信环境也更加恶劣。我们曾购入国外某著名 IC 厂商开发的 PLC

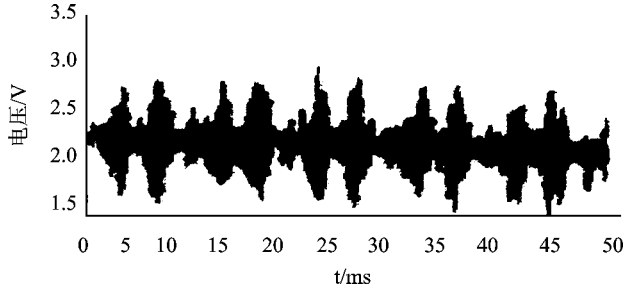


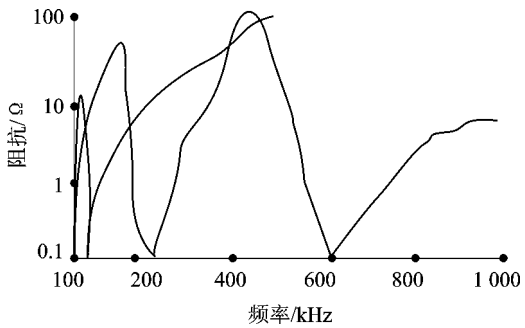
图 4.9－1　电力线上的干扰波形

芯片进行通信实验。结果发现能实现的通信距离比资料上所介绍的数据大打折扣，且误码率随通信距离的延长急速增大。

⑵ 电力线上阻抗依电源线种类的不同，电力负载的大小、数量、种类及接入时间的不同随意变化，导致网络特性阻抗显著波动，变化不可预测，这使得发射机功率放大器的输出阻抗和接收机的输入阻抗难以保持匹配，因而给电路设计带来很大困难。

线路空载时，点对点通信信号传输距离有可能达数千米，而在线路负荷极重时，则信号传输距离仅为数十米。

另外，研究表明，低压电力线路输入阻抗与所传输信号的频率密切相关，从图 4.9－2 看出，输入阻抗随频率的变化非常之大，且规律性不强。



两曲线是在同一个低压电力线网的不同地点测得的

图 4.9－2　输入阻抗-频率关系图

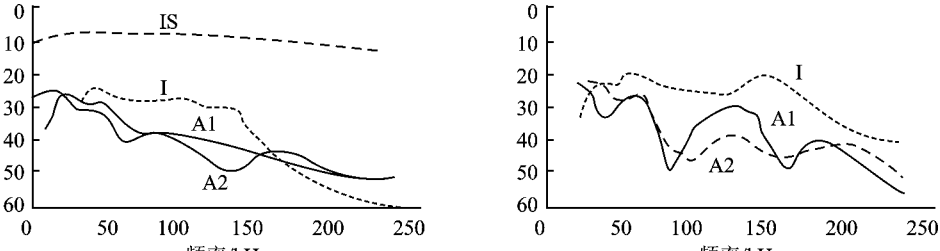
实践中发现高频信号传输中最大的障碍是线路中存在的电容。气体放电灯具上所并接的电容、电子设备及开关电源在电源输入端联接的滤波电容以及配电柜中挂接用以提高功率因素的电容器组等等，都能将高频信号旁路。如线路上一个 10 μF 电容，对于 100 kHz 信号的阻抗仅为 0.16 Ω，几近短路。压敏电阻在常态时也是一种电容，我们为了防雷，曾在挂在室外的载波通信装置进线端搭接压敏电阻，结果导致通信中断，不得不将其拆除。

另外，地下电缆中线间分布电容比架空明线大得多，200 kHz 高频信号经过 200 m 长的地下电缆后，幅值衰减到近乎零电位。

⑶ 信号衰减是低压电力线用作通信信道的又一个问题

首先，信号衰减与通信距离密切相关，信号传输距离越远，信号衰减就越厉害；一般为 40～100 db/km。其次，信号衰减与时间也有关系，在工业区，白天衰减比晚上大，而在居民区，

晚上用电高峰期衰减最大。图 4.9－3 所示为一个工业建筑内测出的信号衰减与频率之间的关系，可见衰减与信号频率也直接相关。



IS 曲线是在近距离（大约 10 m）同一相上测得的衰减曲线；I 曲线是在较长距离同一相上测得的曲线；A1 和 A2 曲线表示距离较长且发送机和接收机不在同一相上的信号衰减曲线。

图 4.9－3　衰减-频率变化曲线

另外，由于电力变压器对载波信号有阻隔作用，而同一变压器的不同相线间亦对信号传输形成障碍，因而，载波数据通信若要跨变压器、跨相传输则须在相间或变压器间搭设信号耦合装置。

三、载波数据通信实现方法

图 4.9－4 为实现低压电力线数据通信的载波发送与接收机的一般描述。

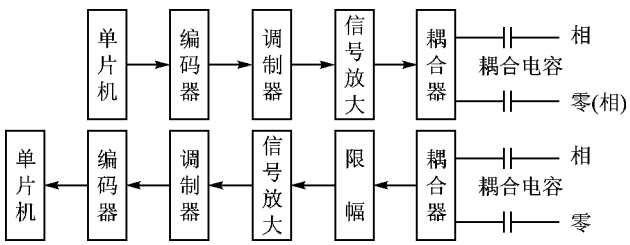


图 4.9－4　载波机框图

先将要传送的数字信号进行编码，然后将其对正弦载波进行调制，已调波经放大后由耦合器经耦合电容送入电力线路，这就完成了发送过程，接收过程正好与此相反。

现在，国内外多家公司已推出集成化的 PLC Modem，即在一块芯片上已集成载波机的主要功能模块，只需外接不多的几个元器件即构成一载波机。它通过串口与单片机或微机相联结。

搭载载波信息的电力线可以是一相一零，也可利用两根不同的相线。下面按框图中各部件的功能对通信实现方法分别进行介绍。

1. 信号编解码

对数字信号进行编码有多种方式，电力线载波数据通信常用的有 DTMF（双音多频）及 FSK（频移键控）等。

DTMF 编码借用电话机拨号机制，将待发送的每一个数字代码（0—F），在单片机控制下，由 DTMF 编码器（如 5087 芯片）转换成相应的每组两个音频信号，如代码 A，由一个 697 Hz 和一个 1 632 Hz 的一组双音频信号表示。接收机中的解码器（如 8870 芯片）则将收到的双音频

信号还原成相应的数字代码，我们曾在一个用于路灯监控系统的载波通信机中采用这种编码方式。实践证明这种编码方式实现容易，成本也低，但易受干扰出现错码和丢码。虽然采取三次重发的办法予以补救，亦不能保证 100% 的通信成功率。

FSK 编码即是将二进制码“0”和“1”分别用不同频率的等幅波表示，见图 4.9－5 示，例如传输速率定为 1 200 波特时，以 f_1 = 217.6 kHz 代表数字“0”，以 f_2 = 222.4 kHz 代表数字“1”。这种方式的电路实现也简单，但抗干扰能力较强，胜于前者。

2. 信号调制与解调

由于 AM（调幅）调制方式通信质量不高，数据通信中较少采用单纯的 AM 方式，多采用 FM（调频）方式。单边带调制（SSB）可视为对 AM 调制方式的一种改进，因其抗干扰能力强，且通信保密性好，也被用于载波数据通信。单边带调制原理简述如下。

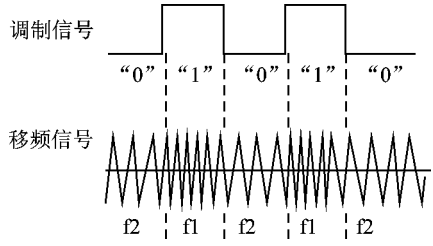


图 4.9－5 FSK 编码

设基带信号为一单一频率正弦波

$$m(t) = u_\Omega / u_c \cos 2\pi ft$$

则调幅波的数学表示式为

$$u_c(t) = u_c \cos 2\pi f_c + \frac{1}{2} u_\Omega \cos 2\pi (f_c + f) t + \frac{1}{2} u_\Omega \cos 2\pi (f_c - f) t$$

式中第 1 项为载波，后 2 项分别为上下两个边带，两个边带都包含了基带信号信息。而单边带调制正是采用滤波法或其他方法去掉了已调波中的一个边带。

单边带调制另一些特点是能节省频带和发送功率，但电路复杂，国外西门子公司的 ESB500 电力线载波机即采用单边带调制方式。

3. 载波频率选择

230 kHz 曾是英国规定的用于电力线载波通信载波频率标准，国内许多应用也常采用 200 kHz 上下的频带，频率过低则信号耦合较难，而频率太高则远距离传输衰减较为严重。另外，载频为 200 kHz 上下时，线路阻抗呈感性，且阻抗大小在 20 Ω 左右变化，通信情况较为稳定。

也有采用 1 kHz 以下的低频载波的。这使得数据传递信号带宽极窄，传输系统也较复杂。但优点是可获得较高信噪比，信号衰减也较小。

20 世纪 90 年代初，欧洲提出了 BS EN 50065 标准，对在低压电力线上的载波信号的频段、频带和电平等都作出了规定。该标准对频段的划分作了如下规定：

3～9 kHz——电力公司专用频段；

9～95 kHz——电力公司和经电力公司许可的用户使用的频段；

95～148.5 kHz——其他用户使用的频段。

我国电力线通信载波频率范围一般为 40～500 kHz。

适当提高发送功率及采用合适变比的耦合器，也对提高通信成功率起显著作用。

四、新技术与新器件

传统 PLC 速率低、信息传输距离短、通信成功率不高，限制了其应用范围。为了适应计算机网络及多媒体等大容量信息高速传输的需要，国内外许多开发机构都投入了大量的人力财力进行研究，近几年，用于低压电力线载波数据通信的新技术与新器件不断推出，形势喜人，下面根据笔者所掌握的一些资料介绍几种。

1. 扩频技术（扩展频谱，spread spectrum）

扩频技术发展于上世纪 50 年代中期，原用于军用无线电通信，采用这种技术可使通信系统具有很强的抗干扰能力及保密性能，同时，还具有误码率低、发射功率小的优点。上世纪 90 年代后半叶，一些开发机构将该技术应用于电力线载波数据通信获得成功。

扩频技术的基本理论根据信息论中的香农（shannon）公式：

$$C = W \log_2 (1 + S/N)$$

式中 C 为信道容量（bit/s），W 是信道带宽，N 为噪声功率，S 为信号功率。

香农公式表明一个信道无误差地传输信息的能力与信道中的信噪比以及用于传输信息的带宽之间的关系。

上式经换底后化为

$$C/W = 1.44S/N$$

这说明，对于一个给定的信道容量而言，既可以用增大信道带宽同时相应降低信噪比的办法达到，又可以用减少信号信道带宽同时相应增大信噪比的办法来实现。从理论上说，对于任意的信噪比，只要增加用于传输信息的带宽，就可以增加在信道中无误差地传输信息的速率。

扩频通信即是利用同欲传输的数据（信息）无关的码对信号扩展频谱，如直接序列扩频则是将一串伪随机码（其频率比信息码高得多）与信息进行模 2 加，使之占有远超过被传输信息所必需的最小带宽。在接收端则利用同一码进行相关接收和恢复数据。

采用扩频技术的美国 Intellon 公司 ssc p300 芯片，信息传输率最大可达 10 kbit/s，它的数据报文采取短帧格式以降低通信误码率，可在噪声范围较大的环境下稳定可靠地工作。类似的芯片还有 PLⅢ、PL2101 等等。

2. 数字配电线载波技术（DPL - Digital Power Line）

DPL 由英国 Norweb 通信公司与加拿大 Nortel 公司共同开发。它采用改进的因特网规约（IP）以及复杂的专用电子装置，来沿低压配电电缆网络传输 MHz 级的数字高频通信信号，同时监视导致信息失真的脉冲信号以及其他形式的干扰。由于数据传输中的数据流是被分切后封装起来进行传输，而不是以连续的信号流形式传输，因而数据传输不像声音通信那样易受干扰。此外，IP 能保证被噪声影响的封装可再传一次，提高了通信成功率；另一方面，由于数字化双工设备的采用，调制过程的应用及无线装置性能上的改善，可利用的通信带宽已增至原来的 100 倍。

3. 本地网络回路（PL^2 - Power Line Local 100P）

该项技术为韩国 Keyin 公司最新开发，据称是一种出色的廉价高速解决方案，解决了 PLC 应用中所遇到的技术障碍。

PL^2 采用了增强模拟前端处理（ENAFE，Enhanced Analog Front End）以及多电平频移键控（MESK）和正交调幅（QAM）等多项技术，有效地降低了噪声，提高了 PLC 传输速率。

他们推出 PL^2 输电线 Modem,使输电线通道最优化,尤其是找到一个最优化电力线通道的通信波长,使 PLC 传输速度可达 2 Mbit/s 以上,误码率可达 10^{-9}。10 Mbit/s 或更速率的 Modem 芯片组也将陆续商品化。

五、应用现状和前景

我国在低压电力线载波数据通信方面的应用尚处于起步阶段,主要停留在自动抄表、楼宇保安和某些过程控制领域。

在美国、电力系统通信已成为信息高速公路计划的一个重要组成部分。

美国 Hunt 技术公司的一项统计表明,自 1995 ~ 1997,该公司已经在 40 个州的 200 多个能源电力公司安装了超窄带载波数据通信系统,除抄表功能外,在能源管理中,如线路电压监测、停电情况监测等,也有不少的应用。

美国的 Lonworks 技术为 Lon 总线提供的智能化现场测控产品中,就包含有低压电力线载波数据通信设备——PLT 系列电力线收发器。该类收发器采用了数字信号处理 (DSP)、动态调整收发器灵敏度及短报文头纠错等技术,传输速率达 2 ~ 10 kb/s。

欧美还在尝试利用低压配电网载波通信来实现配电网自动化和负荷控制,并且用于工业控制和家庭自动化的同时,进而提供电话和视频传输等服务。欲将四通八达的配电线转化为信息高速公路的通道。

1997 年,在英国曼彻斯特的一所中学采用 Norweb 通信公司 DPL (Digitel Power Line) 技术,成功地将 12 台 PC 机通过配电线连至因特网,通信速率高达 1 Mbit/s。到 1998 年 Norweb 公司已为英国西北部的 2 000 居民和商户提供了电力线的商业因特网服务。

在 2001 年德国汉诺威信息技术大展上,德国 RWE 能源股份公司和瑞士的 ASCOM 公司推出了名为电力互联网 (Power Net) 的新技术。这种新的传输技术将能通过电源线路传输各种互联网数据信号,从而大大推进互联网的普及。将特制的 Modem 与普通电线插口相接,就可以 2 Mbit/s 的速度上网浏览。

前文提到的韩国 Keyin 公司将 ENAFE 技术成功应用于 2 Mbit/s 的 Modem,构成 DSP 环境下的实时数字式电话。几种高速 PLC 应用产品,包括:用户端数字式电力线 Modem,具有 10Base - T 的以太网支持;服务供应商端家庭连接器 (PL^2 - HC),安装在用户家用电表附近;电力线路由器,安装在用户变压器低压侧,都已商品化,采用该公司的 PL^2 解决方案,可取代计算机联网的综合布线系统,每一个网络用户计算机可直接通过低压电力线互联成网,并进入国际互联网。

在美国,目前思科、惠普、英特尔等 90 多家公司组成的一个开发集团,正在研制被称之为"家庭插接电力线装置"的设备。这一开发集团已为相关装置设定了统一标准。美国英特伦公司的做法是让每个家庭都使用转接器。用户先将电话线插入转换器中,然后把转换器插入墙上插座,便组成一个小型网络。

总之,对于低压电力线载波数据通信信道特征及其实现技术的研究和探索,导致了这一领域里新技术和新器件的不断涌现,极大地推动了它在质和量两个方面的发展,从而不断地拓宽着它的应用范围,为计算机和通信事业的发展增添了新的活力。

本文经曹文彬教授、欧阳湘达教授、向万新研究员审阅并提出修改意见,在此一并致谢。

参 考 文 献

1 沈允春. 扩谱技术 [M]. 国防工业出版社

2 张学漠. 电力线载波信道 [M]. 水电出版社

3 王钧铭. 通信技术 [M]. 电子科技大学出版社

4 Radford D. Spread - Spectrum Data Leap Through AC Power Wiring [J]. IEEE Spectrum,1996

5 邱玉春,徐平平. 低压电力线载波信道特性分析 [J]. 电力系统通信,1999,4

6 高锋,董亚波. 低压电力线载波通信中信号传输特性分析 [J]. 电力系统自动化,2000,4

7 宋永华,肖颖,张琪. 电力线载波技术重大突破——数字配电线载波技术及其应用 [J]. 电网技术,1999,2

8 张小伍,涂荣疆,王声琪. 新一代高速电力线载波通信技术 [J]. 2000,10

9 郑黎明,王书桓,姜相. 电力线载波中的超窄带通信技术 [J]. 电测与仪表,1999,7

10 电力线上网究竟是什么 [J]. 财经时报,2001,6,15

选自《计算技术与自动化》月刊,2002 年第 3 期

4.10 基于 LM1893 的电力线载波通信系统设计

广西工学院电子信息与控制工程系 (545006)
韩峻峰 彭 勇 张 巍 陈文辉

电力线载波通信以电力线路为传输通道,具有通道可靠性高、投资少、见效快、与电网建设同步等优点。系统所有设备可就近、就地接入电力线,无须重新布线破坏建筑结构及装潢,具备适用于早期无通信系统的旧建筑以及不占用其他资源的特点;同时能够巡检用户,及时发现事故隐患,利于查找事故原因。其缺点是仅限于单个变压器供电范围内的管理,超出变压器供电范围的设计复杂,成本较高。

随着人们生活水平的提高,智能大厦、智能小区已成为市场热点,作为一个综合性的系统工程,它包含许多小系统,各家各户、每一房间也存在铺设通信线路问题,例如消防报警系统、防盗报警系统等,把各报警点信息集中起来统一处理,采用电力线载波通信有其无法比拟的优越性,由于小区本身具有社区服务、保安中心,可在第一时间内获取相关报警信息以及定时巡检用户,根据需要可以采用电话联系方式与上级报警中心联系,完全可以解决电力线载波通信存在的不足。本文以智能小区安全报警网络系统的应用为例,对基于 LM1893 的电力线载波通信系统作一介绍。

一、系统简介

电力线载波通信即将载波通信信号调制在输电线路上,载波通道的衰减特性类似于架空明线,即频率越高其衰减越大,因而传输频段一般选在 40 ~ 500 kHz 范围内 (即为中频段),电力线与端局的耦合通过耦合电容进行。此外,为了阻止载波通信信号流入各种电力设备,配置了扼流线圈。智能小区通信报警系统结构如图 4.10 - 1 所示。

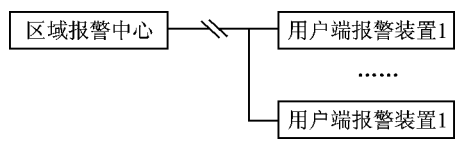


图 4.10 - 1 电力线载波通信系统结构简图

二、系统设计

区域控制中心硬件电路框图如图 4.10 - 2 所示。中心设备采用 MCS51 单片机为主控单元,通过电力线载波芯片 LM1893 实现与各用户端通信,传输报警和巡检信息。

LM1893 是美国 National Semiconductor 公司生产的高性能专用电力线载波通信芯片[5],可实现可靠的半双工电力线数据通信。LM1893 含有数据调制解调的全部功能,只要设计出控制单元及线路耦合变压器即可构成电力线载波通信系统。由于采用基于锁相环的解调器和针对脉冲噪声干扰设计的脉冲噪声滤波电路接收数据,使接收信号的动态范围很宽,灵敏度极高,且干扰小;发送数据时,运用基带数据对载波进行调制,调制后的载波信号经片内功率驱动器加到电力线上,进行半双工数据通信。

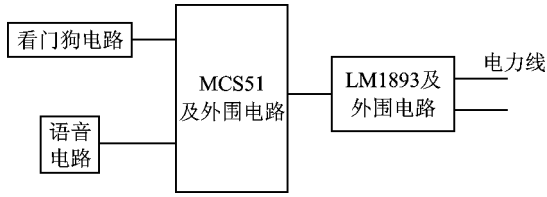


图 4.10－2 区域报警中心硬件电路框图

语音电路用于报警提示，包括语音接口电路、语音处理组件以及话筒扬声器等，芯片选用 ISD2500 系列，外围少，控制简单，可根据需要分组录放并预置播放次数。

看门狗电路选用 X25045 以防止数据丢失和 CPU 误动作。该芯片集成了看门狗定时器、电源监控电路与 EEPROM 三个部分，可编程操作，可靠性高，功耗低，内含 512 B 的 EEPROM，用于存储话机号码、系统设置等数据。

1. 通信接口设计

LM1893 与 MCS51 单片机接口的电路如图 4.10－3 所示，采用 MCS51 单片机为主控单元，通过 LM1893 芯片实现与各用户端的通信，传输报警和信息巡检。LM1893 [5] 电路的工作状态由发送/接收引出端 TX/$\overline{\text{RX}}$控制，当处于高电平发送状态时，由单片机送出的串行数据经 LM1893 的 17 脚送入内部调制器，产生控制电流，控制输出三角波的电流控制振荡器工作频

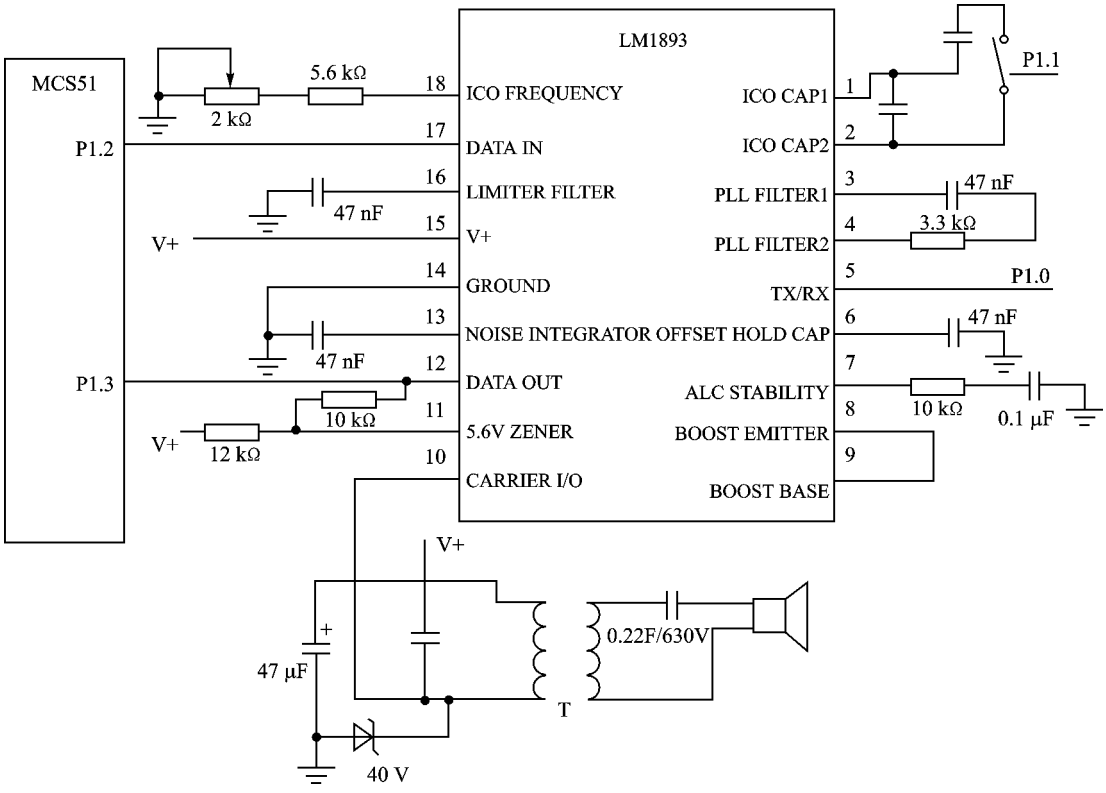


图 4.10－3 通信接口电路

率，其输出信号经正弦波成形器、自动电平控制电路及输出功率放大器后输出，经过 LC 选频网络以后，由高压耦合电容送到电力线上 当处于低电平接受状态时，从电力线上来的信号经高压耦合电容和变压器 T 组成的耦合电路从 LM1893 的 10 脚送入内部平衡限幅放大器，经锁相环路解调，得出包含载波的高次谐波、噪声及直流分量的数据信号，经三阶 RC 滤波器和偏移抵消电路后的信号再经过比较器进行整形和脉冲噪声滤波器滤出信号中的脉冲干扰，从数据输出端 12 脚输出数据信号。

当发生火警、盗警时，不仅要把报警信号及时准确发给中心主机，而且要对中心主机的巡检作出反应，为此，本文采用了两种载频来完成这两种通信任务，以解决相互干扰问题。LM1893 的 1、2 脚为电流控制振荡器的外接电容端，用于控制电流振荡器的振荡频率。因此，本文通过单片机的 P1.1 口根据通信种类控制载波电容 2 的接入与断开，实现两种载频的切换，确保通信数据可靠。LM1893 调制解调数据的输入、输出端分别与单片机的串行输出、输入端口相连接，单片机的 P1.0 口为发送/接收控制端。

LM1893 初始化设置在接收报警信号状态，发生警情后，区域中心单片机一旦检测到 P1.3 口负跳变信号，读取 LM1893 的报警信息，发出报警。单片机上电复位后，将 LM1893 置于发送巡检信号状态，接收用户端返回的巡检结果，若出现异常情况，则接通报警。

2. 通信软件设计

本系统利用单片机的多机通信功能通过串行工作方式实现区域报警中心与用户间的通信。主机通过不同的地址帧和数据帧信号来与多用户实现一对一的通信，程序框图如图 4.10－4所示。

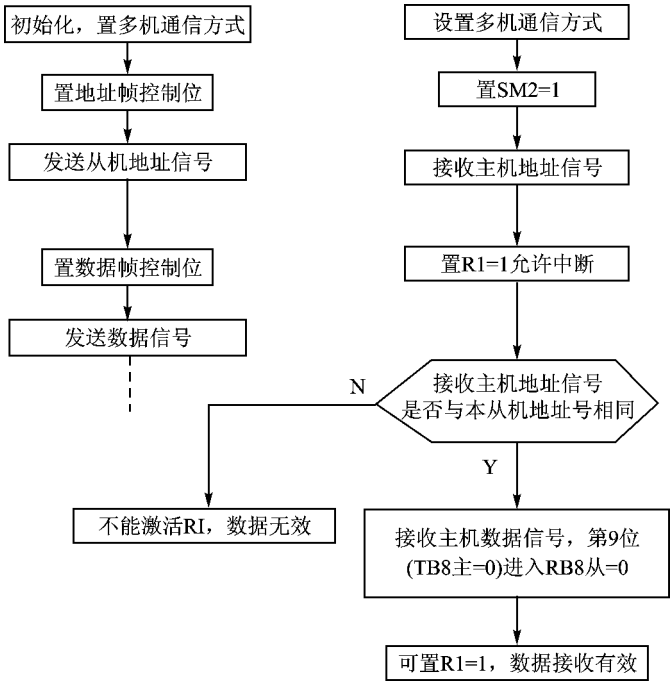


图 4.10－4　多机通信主、从机程序框图

3. 通信协议

为确保通信的准确可靠,在本报警系统中采用了如下通信协议。

(1) 主机发送的控制命令代码 (发送时 TB8 = 0) 为:

00H :要求从机接收数据块;

01H :要求主机发送数据块。

(2) 地址 FFH (发送时 TB8 = 1) 是对所有从机都起作用的命令,命令所有从机恢复到 SM2 = 1 状态,准备重新接收主机发送的地址。

(3) 从机状态字格式为:

DERR						TRDY	RRDY

其中

如果 DERR = 1,表示从机接收到非法命令;

如果 TRDY = 1,表示从机发送准备就绪;

如果 RRDY = 1,表示从机接收准备就绪。

(4) 发送数据块长度为 16 B。

本文设计的电力线载波通信系统由于采用了具备数据的调制解调功能和与电力线的连接功能,并且包含了滤出电力线所特有的脉冲噪声的滤波器的 LM1893 集成电路,可用于借助电力线构成通信网络,具有成本低、工作方式灵活、可靠,抗干扰强的特点,可广泛应用于智能小区、商业网点等单变压器供电的局域网通信。

参 考 文 献

1 孙光伟,李丽艳,牟英峰,刘红. 智能住宅电力线载波通信的研究 [J]. 低温建筑技术,2001 (2)

2 姚振东,卢强. 电力线载波通信实现数据及语音的传输 [J]. 成都信息工程学院学报,2001 (4)

3 屈励,姚虹,郭文华. 电力线载波通信火灾报警网络系统的研究开发 [J]. 安微机电学院学报,1998 (4)

4 李烈刚. LM1893 构成的电力线载波通信系统 [J]. 郑州大学学报 (自然科学版),2000 (2)

5 黄采伦,刘易凯. LM1893 电力线 MODEM 原理及应用 [J]. 国外电子元器件,1999 (1)

选自《电测与仪表》月刊,2002 年第 8 期

4.11 家庭无线信息网络解决方案

深圳桑达电子总公司 李达全

一、前 言

随着因特网爆炸式发展,PC 机在家庭中的日益普及以及数字化、智能化保安监控设备和网络家电的出现,人们越来越希望把 PC 机、各种家电、保安监控及三表抄表等设备连接起来,构成一个家庭信息网络,并接入互联网,旨在实现集中管理和控制、家庭信息资源共享、家庭成员同时上网,以满足不断发展的家庭办公、家庭娱乐、电子商务、网上教育、保安监控等的需要。家庭信息网络技术是当前受到广泛关注的热点技术,正蕴育着一个巨大的潜在市场。

为了加速我国家庭信息网络技术的发展,目前我国国家经贸委正组织清华同方、中兴通信、TCL、海尔、长虹、信息产业部电子三所等 11 个有影响的单位,成立家庭信息网络联合体,着手研究和制定我国家庭信息网络标准。

家庭信息网络与计算机网络不同,家庭信息网络内并非所有设备都要信息交换,所以不必采用复杂的分布式结构,一般采用以 PC 机(也有以数字彩电)为中心的集中式控制结构。对家庭信息网络的普遍要求是:价格低廉、方便操作(即插即用)、带宽够用、安全可靠。

家庭信息网络按传输信号的媒质分为:家庭有线信息网络和家庭无线信息网络。目前已经进入和即将进入市场的家庭有线信息网络有家庭电话线信息网络、家庭电力线信息网络等;家庭无线信息网络有无线局域网(WLAN)、HomeRF、蓝牙网络等。这些网络各有其特点和各自的适用场合,目前还没有哪种网络产品能成主导产品,蓝牙网络能否胜出,也还难说。家庭信息网络技术正在快速发展中。

二、基于无线局域网技术的家庭信息网络

无线局域网(WLAN)技术有两种标准,一种是美国 IEEE(电气与电子工程师协会)标准,一种是欧洲电信标准协会(ETSI)标准。它们新的代表性产品是 IEEE802.11b、IEEE802.11a 和 HiperLAN2。它们的主要性能指标见表 4.11-1 所列。

表 4.11-1 新的代表性产品主要性能指标

标准 / 指标	IEEE802.11b	IEEE802.11a	ETSL HiperLAN2	注
频率范围	2.4~2.4835 MHz	5.15~5.35 GHz 5.725~5.825 GHz	5.15~5.35 GHz 5.47~5.57 GHz	
调制方式	直接序列扩频	OFDM(正交频分复用) 52 个子载波	OFDM,52 个子载波	
最高传输速率	11 Mbps	54 Mbps	54 Mbps	

续表 4.11-1

标准 / 指标	IEEE802.11b	IEEE802.11a	ETSL HiperLAN2	注
访问层协议	GSMA/CA	CSMA/CA	TDMA/TDD	
发射功率	15 dbm	50 mW (5.15~5.25 GHz) 250 mW (5.25~5.35 GHz) 1W (5.725~5.825 GHz)	200 mW (5.15~5.25 GHz) 1 W (5.47~5.57 GHz)	
链接	无链接	无链接	基于链接	
加密	40 bit WEP	40 bit WEP	DES,3DES	
室内传输距离	30~150 m	30~150 m	最大 150 m (无障碍)	
售价	网桥 275 美元 网卡 99 美元 (家用) 适配器 56 美元	即将上市	即将上市	

两种标准的产品最初都选在 2.4 GHz ISM 频段,采用扩频通信技术,最高传输速率达 11 Mbps。为了向更高传输速率发展(目前已开始研究 108 Mbps),两种标准都把载频提高到 5 GHz 频段,调制方式采用 OFDM(正交频分复用)调制方式。

OFDM 技术实际上是一种多载波调制技术,其基本原理是在频域将给定信道分成许多个子载波信道,传输时将原始数据信号分解成多个比特流,每个比特流具有低得多的传输比特率,并且用这些比特率流去进行调制各个子载波。比特流对子载波的调制可选择多种方式,如 BPSK、160QAM、64QAM 等。由于每个子载波信道是相对平坦的,而且传输的是窄带信号,所以具有很强的抗多径干扰引起的频率选择性衰落的能力,同时,由于各子载波信道可以重叠,所以 OFDM 调制方式具有很高的频谱利用率。OFDM 在宽带应用环境下具有很突出的优势,正受到世界广泛的关注,OFDM 技术的不足之处是对频率偏移和相位噪声很敏感,所以对同步的要求很高。

由上面介绍可见,不管 802.00 还是 Hiper-LAN 产品,就其网络性能来说,均可满足和超过家庭信息网络指标要求,但价格太高,技术比较复杂,较适用于居住大型别墅或豪宅的富裕家庭。

为了满足一般小康家庭高速连网的需要,朗讯科技公司专门推出一种针对家庭和小办公室使用的住宅式网关。其主要技术性能与 802.11b 相同,操作更加简便、价格较低,可轻松实现家庭资源共享和家庭成员同时上网,有一定的市场前景。

三、基于家庭射频(HomeRF)技术的家庭信息网络

为利用射频技术构成家庭语音和数据网络,1997 年美国家用射频委员会成立了 HomeRF 工作组,成员包括 Inter、Microsoft、Motorola、Proxim 等。HomeRF 技术是数字无绳电话技术(DECT)和无线局域网(WLAN)技术相互融洽发展的产物。DECT 使用时分多路接入(TDMA)方式,特别适用于话音通信,而 WLAN 采用 GSMA/CA(载波监听多路访问/冲突避免)方式,特别适用于数据通信。二者融洽形成家庭射频信息网络的共享无线应用协议 SWAP(Shard Wireless Access Protocol),同时适用于话音和数据业务。SWAP 是 HomeRF 工作组提供的一个开放式家庭信息平台,各种可操作的家庭电子设备可通过它进行话音和数据通信,并能与公共电话网和互联网进行交互操作。

HomeRF 网络工作在 2.4 G ISM 频段;采用跳频(PH)扩频技术(每秒 50 跳);发射功率 100 mW,通信距离 150 英尺(45.75 米),典型的家庭庭园范围;每个网络最多可支持 127 个设备;数据速率 1 Mbps(2FSK)、2 Mbps(4FSK),将进一步扩展到 10 Mbps;与 TCP/IP 协议很好结合,支持广播、多点传送和分组,48 位 IP 地址,允许同一区域多个网络平行操作;可实现 6 路全双工话音通信,接近有线话音质量;数据可用 24 位的网络 JD 号和 MAC 层 56 位密钥加密,在微波炉开启情况下能很好工作。HomeRF 适用于小公司和大型别墅式家庭。

HomeRF 网络的拓朴结构是以连接点 CP 为网络核心星形结构。网络设备分为两类:一类为 TDMA 同步设备,如电话等;一类为 CSMA 异步设备,使用 TCP/IP 协议,如 PC 机和网络家电等。连接点与公众电话网相联,所有设备在连接点控制和管理下工作。

HomeRF 技术主要竞争对手是蓝牙技术。如果蓝牙芯片能实现预期的性能和价格目标,则 HomeRF 有可能被蓝牙取代。

四、基于蓝牙技术的家庭信息网络

蓝牙(Bluetooth)技术开发计划由爱立信公司在 1994 年提出,Intel、IBM、Motorola、Noria 等 8 家世界著名跨国大公司先后加盟该计划,组成了蓝牙特殊利益小组(SIG),旨在建立一种短距离的开放式无线电接口标准和建立控制软件的开放式标准,使全球不同厂家生产的设备,不通过线缆方便地实现近距离链接。目前该计划已得到全球 2 000 多个厂商的广泛支持。IEEE 标准化工作组成立了 IEEE802.15 工作组专门研究蓝牙技术标准及未来的发展。当前该工作组正在研究建立与蓝牙 1.0 版本相一致的标准,探讨蓝牙如何与无线局域网共存和蓝牙的传输速率向 10 ~ 20 Mbps 拓展的问题。

蓝牙技术实际上是一种短距离无线电通信技术,目的是提供一种低成本、方便高效及支持高质量话音和数据传输的无线通信网络。它采用 2.4 GHz ISM 频段和先进的快速跳频(1 600 跳/秒)、前向纠错(FEC)、短分组等技术,减少了同频干扰和随机噪声干扰的影响,采用移频键控(GFSK)和零中频技术,降低了设备的复杂性和成本。它的标称传输速率为 1 Mbps,传输距离为 10 ~ 100 m。蓝牙的安全体系可为蓝牙技术的近距离和全球应用提供安全保证。

利用蓝牙的一个跳频信道,可以构成一个微微网,称为 Piconet。一个微微网最多可由 8 个工作单元组成(1 主 7 从),从单元在主单元控制下工作,主单元承担接入互联网任务,是网络的核心。另外还可有 200 多个待机单元。一般情况下,一个微微网即是一个家庭信息网络,如果工作单元数量或复用范围不够,可以构建多个微微网,同一区域的相互联通的多个微微网称为散布网(Scatternet)。而微微网使用不同的跳频序列来区分。

与 HomeRF 相比,蓝牙产品的特点是:采用全部统一的开放式技术标准,价格更低(每个芯片目标为 5 美元);功耗更低和体积更小(目标 8mm × 8mm),便于嵌入移动终端中,跳频速度更快,抗干扰能力更强,安全更有保证。

目前蓝牙 SIG 正抓紧完善技术标准,解决蓝牙产品与同处 ISM 频段的无线局域网、微波炉等设备的共存问题和不同厂家的蓝牙产品的兼容问题,完善产品测试和认证规范,进一步提高传输速率和降低成本,扩大产品应用领域。蓝牙技术还在发展,其中蓝牙芯片的价格将成为蓝牙网络能否胜出的关键因素。

选自《无线电工程》月刊,2002 年第 3 期

4.12 基于 GSM 短消息接口的 MC3 一体化遥测系统

国防科技大学 张正红 胡小军 刘 东

GSM (Global System for Mobile Communication) 是全球移动通信系统的简称。在 GSM 中，惟一不需建立端到端通道的业务就是端消息业务 (SMS)，在移动设备处于点与点通信状态下，还能同时实现短消息业务。短消息只能传送一句话，这种通信是异步进行的。作为 GSM 系统，每一条短消息都是作为单独的时间来处理的，短消息的传送都是经过短消息服务中心进行周转的。

由于点对点短消息不需单独建立通信通道，因此费用较低。移动、联通用户内部发短信收费 0.1 元，移动与联通对发 0.15 元。每一条短消息可以传送 160 个 7 比特编码数据或 140 个 8 比特编码数据，或 70 个 UNICODE 码。因此，在一些对实时性和数据传输量和传输速度要求不是很高的测控系统中，可以利用 GSM 短消息接口进行数据和控制指令传输，这样就可以节约首期庞大的投资去建立无线通信网络。

一、MC3 一体化测控系统

MC3 一体化就是在一台测控仪器同时具有测量 (measurement)、数据采集 (collection)、通信 (communication)、控制 (control) 功能。现代独立仪器已不仅仅要求具有测量、数据采集并显示的功能，而且还要求具有通信和控制多种功能，同时还要体积小、功耗低，其结构示意如图 4.12－1 所示。MC3 一体化就是要改变过去那种利用各种总线技术将测量、数据采集、通信、控制等功能模块分散互连，而将这些功能模块做在 1 个仪器里面，或者做在 1 个电路板上，甚至于在芯片上集成。这样可以提高系统稳定性、减小系统所占空间、减小功耗、节约电能。MC3 一体化将会成为独立仪器发展的一个方向，尤其在遥测领域。

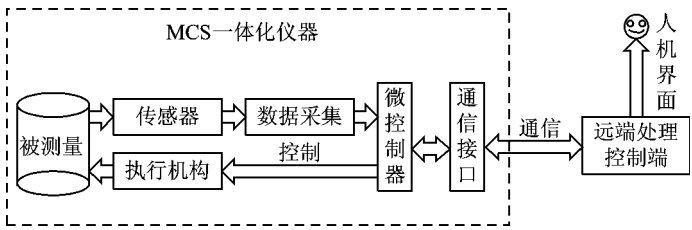


图 4.12－1 MCS 一体化遥测系统示意图

二、MC3 一体化遥测系统的硬件组成

基于 GSM 网络的 MC3 一体化遥测系统的测量和数据采集由 1 个温度传感器芯片 DS18B20 完成。DS18B20 是一种可组网数字式温度传感器。根据单总线独特的优点，它可以使用户轻松地组建传感器网络，并可使多点温度测量电路变得简单、可靠。可组网数字式温度

传感器 DS18B20 是 DS1820 的更新产品，它在电压、特性及封装方面都具有优势，给了用户更多的选择，让用户可以更方便地构建适合自己的测温系统。DS18B20 充分利用了单总线的独特优点，可以轻松地组建传感器网络，提高系统的抗干扰性，使系统设计更灵活、方便，而且适合于在恶劣的环境下进行现场温度测量。系统的通信部分是利用 1 个西门子手机，对温度数据进行传输，同时接收来自外界的控制指令。基本上所有的手机都提供了 1 个用户接口，这些接口的作用主要用于维修。

西门子手机短消息的发送和接收由微控制器 AT89C51 处理，并根据相应的处理，向远端移动用户发送相应的温度值，同时作为微控制器，接受远端的指令，识别、翻译并控制执行机构执行。执行机构由继电器控制 1 个直流风扇。系统的硬件框图如图 4.12－2 所示。

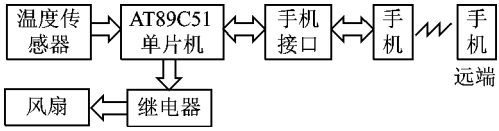


图 4.12－2　系统框图

三、MC3 一体化遥测系统的软件设计

系统开发的关键和主要难点是手机的短消息接口开发。手机短消息的开发主要包括手机短消息的用户数据区编码和解码、通信模式设定和联机测试、短消息的收发、收发数据的关键命令字的模式匹配等。手机接口开发主要是利用由爱立信、摩托罗拉和惠普共同提出的 AT 指令集。AT 指令是基于字符的命令结构，有 TEXT 模式和 PDU 模式，还有早期使用的 BLOCK 模式。BLOCK 模式是二进制流命令格式，具有很强的检错、纠错能力，主要用在通信链路不可靠的环境中。TEXT 模式是基于字符的，更具体地说是基于 ASCII 码的一种结构模式，每一条命令很容易读懂。PDU 模式也是基于字符的，准确地说，是基于十六进制的，数据和代码都经过编码了，所以无法直接读懂。PDU 模式在 GSM 移动设备中使用最为普遍。西门子 C35I 只支持 PDU 模式。不同厂家的 GSM 终端接口是会不相同的。其结构为 AT + 命令 = 参数。例如，读取手机上全部未读过的 SMS 消息，最简单的办法是用 AT + CMGL = 0 ；而用 AT + CMGL = 4 则可读取全部 SMS 消息，无论读过与否。图 4.12－3 给出本系统的软件流程图。

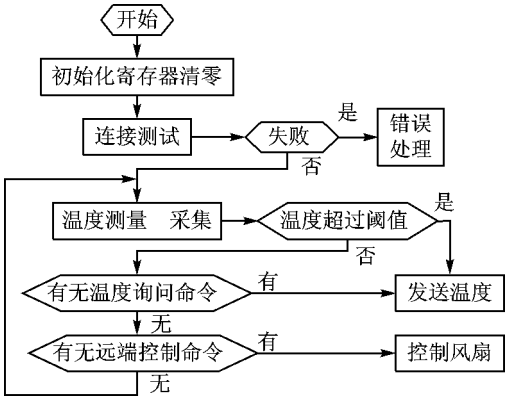


图 4.12－3　流程图

四、结束语

基于 GSM 短消息接口的 MC3 一体化遥测系统经测试运行,工作稳定可靠,远端手机可以是任何移动用户,但命令码可以只对特定人开放,所获取的数据经过编码外,还可以进行数据加密,确保系统的安全性。系统具有很强的可扩展性,能在无人值守、家用电器状况监控、车辆安全等方面有实际的应用;也可以在许多场合使用,如数据采集、商业零售、遥控遥测、全球定位、交通管制、汽车防盗和跟踪、电子零售、安保产品、移动银行、移动 ATM 取款机上得到应用。

参 考 文 献

1 孙孺石,等编著. GSM 数字移动工程. 北京:人民邮电出版社,1996

2 卢尔瑞,等. 移动通信工程. 北京:人民邮电出版社,1988

3 http://www.etsi.com

选自《单片机与嵌入式系统应用》月刊,2002 年第 11 期

4.13　基于短消息的自动抄表系统

西安邮电学院电信系（710061）　毛永毅
哈尔滨电信公司（150001）　张丽娟　岳慧芬

随着电子技术和通信的发展，自动抄表系统得以广泛应用，各类相关产品也层出不穷，人们对新产品也提出更高的要求。

目前的自动抄表系统一般分为两层结构：上层（管理中心与集中站之间）数据采集采用星型网络；底层（集中站和采集器或智能电能表之间）数据采集采用总线型结构，如 485 总线、仪表总线等。目前上层数据采集根据信道介质不同可分为光纤、电话线和无线等多种模式。

其中无线方式，对于范围广、布局分散的集中站进行数据通信是一种较好的方式。现在开发的一些自动抄表系统采用的是数据电台或自行研制的无线通信系统，使用这种通信系统方式，安装调试方便，主要缺点是需申请频点使用权；另外自己需对通信网络进行维护，这种专网通信系统的可靠性也没有公网（如 GSM 网）的高。本文提出了一种于短消息的自动抄表系统，该系统依托 GSM 网，采用短消息方式，通信质量可靠，成本低，同时还能对集中器进行远程监控，是一种理想的自动抄表解决方案。

一、系统的构成和功能

整个系统结构如图 4.13－1 所示，中心管理机利用 GM360 无线调制解调器，通过 GSM 无线通信网络与各集中站以 P87LPC767 单片机为核心的下位机构成星形结构网络，对各电力采集点实施电量采集和远程监控。集中站以 P87LPC767 单片机为核心的下位机模块通过

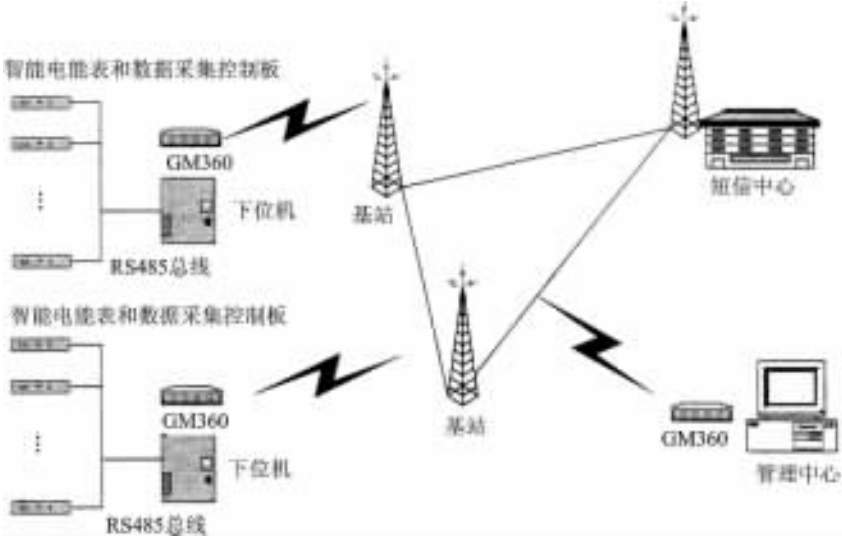


图 4.13－1　基于短消息的自动抄表系统结构

GM360 无线调制解调器利用短消息与管理中心进行通信，当中心发出采集指示后，P87LPC767 单片机通过 RS－485 总线按照三相电子式多功能电能表通信规范与所连接的电能表进行通信，采集电能表中存储的电力数据。然后，再通过短消息将数据发回管理中心。另外下位机和智能电能表还可配备以 89C2051 为核心的数据采集系统，对集中站和智能电能表周围的环境和电网情况进行数据采集，并将数据传送上位机。对于没有智能电能表的供电区域也可采用数据采集系统实施监测，但所采集的数据不能用作计量。当环境温度、湿度等以及电网的电压异常时，下位机即可通过短消息向管理中心或指定手机发回报警信息。管理中心经过权衡后，可决定是否切断该区域供电或派人维护。如需断电，则通过短消息向下位机发出指令，由下位机再传递给数据采集系统执行相应的操作控制供电回路。另外当用户欠费时，也可实施停电处理。

1. 管理机

监控中心的管理机为多媒体计算机、打印机、UPS 电源等构成的系统，通过 GM360 利用短消息与分散在不同位置的抄表机相互联系。管理机采用 VB 编制的管理软件完成信息的提取、发送及分析处理、显示和声光报警等功能。包括口令认证、口令设置、时间及日期设置与显示、下位机编号与名称设置、报警电话设置、退出系统、提取下位机数据以及控制电路通断等功能。

2. 下位机

下位机由 P87LPC767、74HC245、MAX485、MAX202、24WC32、PCF8563 和 DC－DC 电源模块等构成，通过 RS－485 接口接入智能电能表，通过 RS－232 接口接入 GM360 调制解调器，另外也可以通过 RS－485 接口接入数据采集板用于采集智能电能表周围环境，实施对智能电能表的相关监控。同时下位机具有电源监控模块可用于下位机的电源监控及蓄电池的充电控制。由于本系统的 CPUP87LPC767 采用 I^2C 总线外扩存储器和时钟芯片，因此本系统具有结构简单、成本低的特点。下位机硬件结构如图 4.13－2 所示，下位机程序框图见图4.13－3。

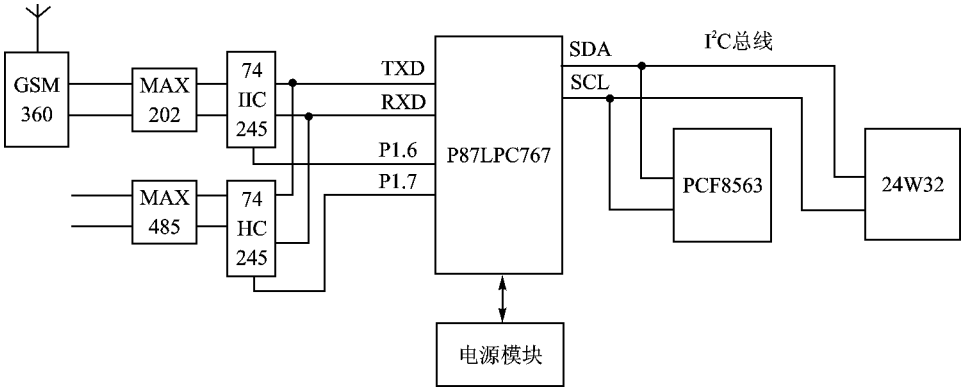


图 4.13－2 下位机硬件结构

(1) P87LPC767 结构和性能

P87LPC767 是 PHILIPS 半导体公司近年来推出的 51PLC 系列 OTP (一次编程) 单片机的一种。它在基本结构、汇编指令等方面与 80C51 系列兼容 然而，它采用 80C51 加速处理器结构，时钟频率可高达 20 MHz，20 MHz 下其吞吐能力相当于 40 MHz 的传统 C51。虽然只有 20 个引脚，但 I/O 口的功能丰富，有两个模拟比较器、Watchdog、I^2C 总线、四路 8 位 A/D，使用片

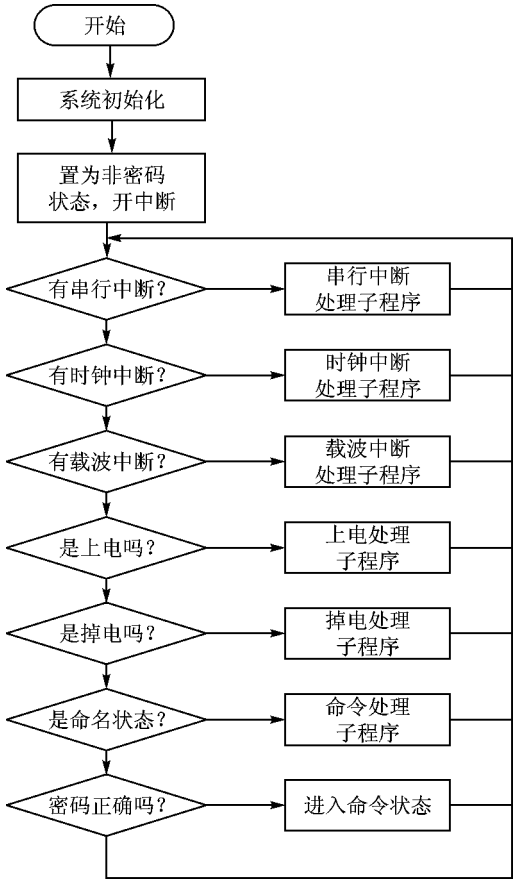


图 4.13－3 **下位机程序框图**

内上电复位时,不需外接元件。温度范围为 - 40 ~ 85 **(工业级)。内部有** 4K OTP **程序存储器,串行的** EPROM **允许在线编程。**

(2) RS485 **接口**

RS485 **接口采用** MAX485 **用于与智能电能表和数据采集系统通信,通过数据采集系统可完成对环境及电网的监控。**RS485 **接口,距离可以超过** 1 km**。每个下位机最多可带** 128 **个智能电能表和数据采集系统。**

(3) 掉电报警

下位机具有掉电报警功能,电源控制电路主要完成开关电源与蓄电池供电的切换、蓄电池的充电以及交流电断电与上电检测等功能。可以在交流掉电时,将掉电日期、时间以及保存在 24WC32 EEPROM **中的下位机编号、名称等数据报告给中心管理机或指定的手机。报警电话号码也存储在** 24WC32EEPROM **中。**

(4) 存储电路和时钟电路

存储电路采用 CATALYST **公司生产的基于** I^2C **总线的** 24WC32 EEPROM **芯片,它具有** 1.6 ~ 6.0 V **的全电压范围和** 100 **万次重写及擦除周期,可存储** 4 KB **数据。时钟电路采用基于** I^2C **总线的** PCF8563 **时钟芯片,也是由** CATALYST **公司所生产,它具有低工作电流,典型值为** 0.25 μA**,宽工作电压范围:**1.0 ~ 5.5 V**。同时,**PCF8563 **还具有世纪标志。**

3. 数据采集系统硬件结构

数据采集系统以 89C2051 单片机为核心与 8 路 AD 转换器 TLC0838 一起构成数据采集系统完成对非智能设备的监控。硬件结构框图如图 4.13－4 所示。每个数据采集系统有 8 路模拟量输入,加上电量传感器可采集电网电压、电流等数据。另外还有两路开关量输入、两路开关量输出接口,适合对一部智能电能表的监控;同时数据采集系统也可以实施温度及电网的监控。每个数据采集系统均采用 485 接口与下位机相连。

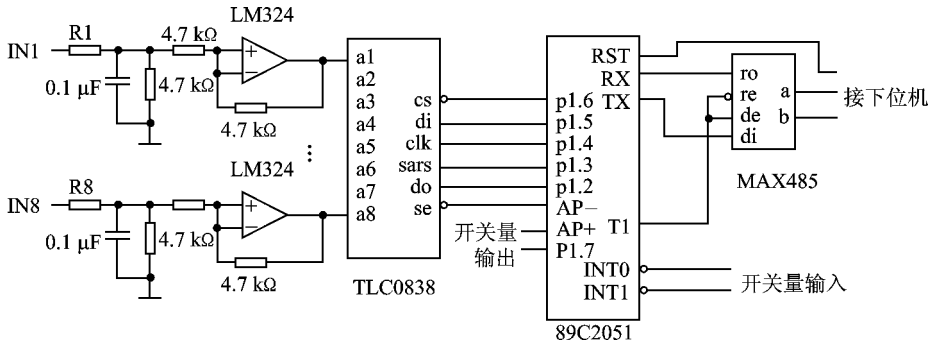


图 4.13－4 数据采集系统硬件结构图

为节省硬件成本,数据采集系统不设有看门狗电路和单独电源,数据采集系统复位信号和电源由下位机提供。另外由于数据采集系统所需处理的数据量较少,因此也不再单独外扩存储器。当数据采集系统程序跑飞时,下位机不能够与数据采集系统正确交换数据,此时,由下位机就对数据采集系统实施复位。数据采集系统程序框图如图 4.13－5 所示。

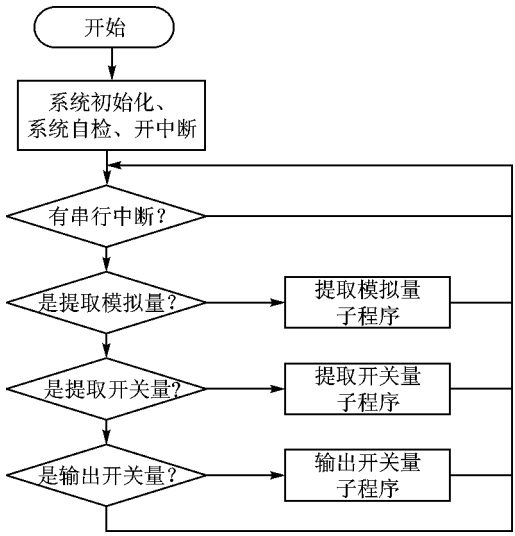


图 4.13－5 数据采集系统程序框图

(1) 模拟量采集

多路数据采集电路采用 TLC0838 完成的 A/D 转换,TLC0838 为 8 路 A/D 转换,前端接有运放 LM324 组成的缓冲电路。由于 LM324 运放输入电压范围为 0～4 V,所以电路之前加有两个分压电阻,用来扩展测量范围,具体输入范围由用户现场来确定。多路数据采集电路可以

根据电网电压数据,来监测电网电压是否异常。同时,还可以通过温度传感器和其他传感器对电力采集点环境因素进行监测。根据实际的电力采集点情况选取温度的测量范围为 - 10 ~ + 50 ,选用 AD590 作为传感器。

⑵ 开关量的输入输出

开关量输出电路可驱动继电器完成对设备电源、供电回路的开关控制,开关量输入电路可用于设备电源、供电回路等的开关状态的监测。

二、通信的问题

1. 通信方式

监控中心的管理机和监控机通过 GM360 接入 GSM 网,利用短消息业务传递信息。该方式优点是通信利用公网,免于维护,通信质量有保证,价格比较低廉,目前每条短消息费用为 0.15 元。但缺点是实时性稍差,每条短消息有效载荷仅为 140 字节,对业务量大的应用不太适合。另外,短消息是利用 GSM 网络的控制信道而不是语音信道进行数据的传递,用户发出的短消息首先被发送到短信息中心的服务器中,然后短信中心的服务器对所收到的短消息进行排队处理,按顺序再发送给相应的接收用户终端,如果接收用户关机或超出服务区不能正常通信时,则该条短消息进行一定的延时后重新发送,这样有可能会造成后发的短消息先到的情况。此外短消息中心服务器为每一个用户开设的缓存区一般较为有限,约 15 ~ 25 条,当接收缓存区存满而接收用户还不能正常通信时,将不再接收新的短消息,即发生短消息拥塞,造成短消息丢失。另外,短消息在短消息中心服务器中保留的时间也有一定的期限,一般为一天左右。为了保证下位机与中心管理机的数据交换,一定要使接收机可靠的与网络处于通信状态。GM360 模块所接收的短消息被保存在 SIM 卡中,普通 SIM 卡一般能存储 25 条短消息,因此,在使用过程当中应及时删除已处理过的短消息,以免造成短消息的丢失。在该系统中采用多点轮询的通信方式,即中心管理机按下位机地址呼叫,下位机接收后,将所需采集的数据送至中心管理机。另外,遇到下位机掉电时,还可由下位机利用蓄电池供电向管理中心或指定的手机进行三次报警。

2. GM360 调制解调器

⑴ 主要特点

① 体积小、重量轻,几何尺寸为:131.5 mm × 52.3 mm × 13.6 mm;

② 自带专用天线;

③ 支持标准 RS - 232 串行接口;

④ 内含 SIM 卡读卡器;

⑤ 模拟语音输入/输出;

⑥ 数字语音输入/输出;

⑦ 透明模式与非透明模式传输速率最大为 9 600 bps;

⑧ 短信息服务(SMS),支持 PDU 模式,符合 GSM 07.05 标准;

⑨ AT 命令集,符合 GSM 07.07、GSM 07.05 和 V25 标准;

⑩ 工作电压:5.6 ~ 10 V,标称值为 6 V;

⑪ 工作电流:待机 70 mA,工作时平均 450 mA;

⑫ 操作温度:- 20 ~ + 70 ;

⑬ 储存温度：- 40 ~ +85 。

⑵ 工作原理

GM360 是一个标准的 GSM 移动终端模块，市场价格目前在 1000 左右，如果配上液晶显示、键盘、话筒等外围设备也可成为普通手机。该模块提供了标准 RS232 接口，采用直流 6 V 供电，采用 AT 贺氏指令，符合 ETSI 标准 GSM07. 07 和 GSM07. 05，并内置微处理器和 GSM 模块结合到一起，具有定时轮询和远端控制功能。在 GM360 中插入 SIM 卡，连接天线，即可通过 RS232 接口与数据终端设备 DTE 相连并利用 GSM 网络和中心进行通信。GM360 仅支持异步通信模式，它支持的通信速率分别为 2400 bps、4800 bps、9600 bps。10bit 字结构，1 bit 起始位、8 bit 数据位和 1 bit 停止位。GM360 采用协议数据单元 (PDU) 可对短消息进行发送、接收和存储。PDU 相当于一个数据包，它构成短消息的信息组成，作为一种数据单元，它必须包括源/目的地址、保护时间、数据格式、协议类型和正文。PDU 的结构根据短消息是由终端发起或以终端为目的而不同。由终端发起时 PDU 的格式如下 (以字节为单位) 。

SMSC	PDU type	MR	DA	PID	DCS	VP	UDL	UD
1 - 12	1	1	2 - 12	1	1	1	1	0 - 140

以终端为目的时 PDU 的格式如下 (以字节为单位) 。

SMSC	PDU type	OA	PID	DCS	SCTS	UDL	UD
1 - 12	1	2 - 12	1	1	7	1	0 - 140

其中 SMSC 为短消息业务中心地址，DA/OA 为源/目的地址，PID 为协议识别，DCS 为数据编码，UDL 为用户数据长度，VP 为数据有效时间，MR 为指明发出信息，SCTS 指短消息到达业务中心的时间。PDU 结构中的数据必须以 16 进制发送且必须为大写。用户数据 UD 即有效载荷，有 8 bit 方式和 7 bit 方式两种，选用何种方式可由 DCS 中的设置决定。GM360 采用 AT 指令完成短消息的发送 (AT + CMGS) 、接收 (AT + CMGR) 、查询 (AT + CPMS ?) 和删除 (AT + CMGD) 等操作，具体操作可查询 GM360 用户手册。

本文所介绍的系统成本低、维护简便、功能全，可对电力采集点实施电量采集和远程监控，极大地提高服务质量、降低工作人员的工作强度。该系统在实际的应用中获得了成功。从实际应用效果来看，系统运行稳定、可靠，达到了预期的效果。

参 考 文 献

1 陈粤初等. 单片机应用系统设计与实践 [M] . 北京 :北京航空航天大学出版社，1992

2 王修才，刘祖望. 单片机接口技术 [M] . 上海 :复旦大学出版社，1995

3 周航慈等. PHILIPS 51 LPC 系列单片机原理及应用设计 [M] . 2001

4 GM360 DATASHEET [Z] ，1999

选自《电测与仪表》月刊，2002 年第 3 期

第五章

新器件与新技术

5.1 ARM 核嵌入式系统的开发平台 ADS

北方交通大学电子信息工程学院电子工程系 (100044)
李哲英 骆 丽 刘元盛

一、引 言

随着半导体技术和系统设计方法的发展,SoC 已成为重要的应用技术,也是半导体技术和应用电子技术新的研究领域。SoC 技术的核心是微处理器 IP 核模块的应用与开发[1]。目前 SoC 技术中广泛采用的 IP 核模块有 8 位/16 位/32 位 CPU 核 (如 ARM720T 内核),以及外围电路核,例如 USB 核、CAN 核、DSP 核、MODEM 核、FAX 核等。

从电子系统设计的角度看,SoC 实际上就是一种高集成度的嵌入式系统[2]。SoC 中的处理器也叫作嵌入式处理器。从应用系统设计上看,嵌入式处理器系统应用设计的关键是应用软件的设计。在工程实践中,由于把整个系统的核心和绝大部分外部电路全都集成在一个芯片中,并且在设计时系统可能还没有形成硬件器件,所以 SoC 嵌入式系统的开发工具就成为能否完成应用系统设计的关键。

由于嵌入式系统在硬件上具有与应用相关的特殊性,所以对嵌入式处理器系统的系统操作软件和应用软件的要求与通用计算机有所不同,要求嵌入式处理器系统的软件满足如下要求:

⑴ 程序应能保存在 ROM/PROM 中;

⑵ 软件代码具有较高的健壮性;

⑶ 系统和应用程序具有满足设计要求的实时性。

随着软件规模的上升和对实时性的提高,目前国际上嵌入式系统的主流操作系统是实时多任务操作系统 (RTOS)[3]。RTOS 是嵌入式应用软件的基础和开发平台,用户的其他应用程序都建立在 RTOS 之上。

本文将从 SoC 和嵌入式系统应用技术的角度,介绍 ARM720T 内核及其开发套件 ADSv1.2。ARM16/32 位 RISC 微处理器内核,是目前通信、控制和信息设备中广泛应用的一个嵌入式内核。随着嵌入式技术的发展,具有较高性能价格比、低功耗的 RISC 处理器内核 ARM,已经成为广泛采用的工业标准。特别是在便携式计算机和多媒体通信领域中,ARM 更是具有十分重要的地位。随着 ARM 技术的不断发展,ARM 已经成为实际上的工程标准嵌入式内核,成为工业领域 SoC 技术的重要分支。

二、ARM720T 嵌入式内核

随着电子技术的发展,特别是消费类电子产品的迅速发展,使得对单片机的能力 (如速度、存储容量等) 提出了更高的要求。从电子器件的结构上看,如果要提高单片机的能力以适应产品要求,会要求采用 32 位单片机。由电子技术可知,微处理器和单片机的开发技术有着

相当大的差别。这样就对设计者提出了一个如何开发 32 位的微处理器系统的问题。这无疑是一个十分严重的技术问题,会使系统的开发周期延长,开发成本增加。嵌入式系统和 SoC 技术提供了解决这个问题的最好方案,这也是嵌入式微处理器得以发展的重要原因。

ARM720T RISC 内核正是为满足这种技术发展的需要而出现的。它提供了低功耗、高集成度和高性能的嵌入式处理器,并在嵌入式系统中得到了广泛的应用。

1. 降低对内部存储器的需求

存储器容量是所有高集成度消费类电子系统设计中的一个十分重要、但又难以解决的问题。ARM720T 采用的是指令精简结构,即 RISC 处理器,大大地减少了对存储器的需求。同时,由于采用 32 位 CPU 和 16 位总线结构,所以可以提供更多的数据存储空间。与一般微处理器相比较,ARM720T 可以降低 35% ~40% 的内存需求。

2. 降低功率损耗

对以电池为电源的消费类电子产品来说,如何在增加功能的同时降低电路功耗,是一个具有挑战性的重要技术问题。根据数字电路的工作原理,系统功耗与速度成正比。因此,速度的提高必然会较大幅度地提高器件的功耗。ARM720T 内核采用高密度的集成工艺,能在大幅度增加系统功能的同时,延长电池的使用寿命。例如 ARM7TDMI 内核的电源电压为 1.8 V,并可以达到 3 600 MIPs/W。此外,ARM720T 还采用了优化工作方式,可以进一步降低功耗。

3. 高速缓冲存储器

ARM720T 中的高速缓冲存储器能有效地减少 CPU 内核对外部存储器的访问次数,并且允许外部存储器和总线以低于 CPU 内核的速度工作。这就在保持了系统性能的同时,降低了对外部存储器的要求,允许使用价格低廉的外部存储器。从系统功能上看,由于目前微处理器系统采用的是串行结构,所以高速缓冲存储器在系统需要多个部件共享存储器时,对提高系统性能具有十分重要的作用。高速缓冲存储器提供了微处理器全速工作时其他电路使用总线的条件。

4. 存储器管理单元 MMU

ARM720T 提供了完整的 MMU,这就允许开发中使用需要存储器管理的操作系统。MMU 允许每一个应用都具有独立的存储器环境,并提供相应的保护。因此,不同的应用进程之间不会相互干扰,也不会被操作系统所影响。

5. ARM720T 内核产品类型

表 5.1-1 列出了不同类型的 ARM720T 内核产品类型。从降低系统功耗和造价上看,不同的应用系统应当根据需要选择相应的内核。当系统比较小、对内存容量需求不大时,可以选用 ARM7TDMI 核作为 SoC 的内核。而当需要具有强化的 DSP 性能或 Java 功能时,就需要选择 ARM7EJ-S 型内核。如果系统需要运行 Windows CE、Palm OS 、Symbian OS、Linux 或其他实时操作系统 (RTOS) 时,应当选用 ARM720T macrocell 内核。

目前已经有 35 个集成电路设计和制造厂家采用 ARM7 内核,并设计出了许多直接面向用户的器件。这些器件采用 0.25 μm、0.18 μm 或 0.13 μm 工艺制造,把用户需要的电路与 ARM 核集成在一起,构成了完整的 SoC 器件,为用户提供了 SoC 系统的实现方法和技术。

表 5.1－1 ARM720T 内核产品类型

工艺	0.18 μm/1.8 V			0.13 μm/1.2 V		
ARM7 CPU 核	晶片面积 /mm^2	功耗时钟 /(mW/MHz)	频率 /MHz	晶片面积 /mm^2	功耗时钟 /(mW/MHz)	频 率 /MHz
ARM7TDMI	0.60	0.25	80～110	0.30	0.08	100～133
ARM7TDMI－S^+	0.65	0.40	80～110	0.35	0.10	100～133
ARM7EJ－S^+	0.95	0.45	80～110	0.45	0.18	100～133
ARM720T	3.50	0.65	60～80	1.80	0.20	100～120

三、ARM720T 开发软件 ADS

嵌入式系统和 SoC 系统的开发中，软件占有十分重要的地位。在单片机系统开发中，由于系统比较简单，所以用户应用系统的所有软件都由用户自己开发，并且工作量也不大。但对于嵌入式系统和 SoC 系统情况就不同了。嵌入式系统的内核一般都是 32 位的微处理器，并且系统的工作要求比较复杂，所以开发工作主要是要开发操作系统。因此，对用户来说，应用系统所面临的是一个非常巨大的开发工作。

为了便于用户开发和 SoC 技术的应用，嵌入式系统开发工具实际上就是一个完整的用户操作系统。使用开发工具开发用户系统时，用户不必关心操作系统，只要根据所使用操作系统提供的环境，开发具体的应用程序就可以了。由此可知，嵌入式系统和 SoC 系统的开发，实际上还包括操作系统的开发应用，并且最终用户系统也以所采用的操作系统为核心。

ARM720T 使用的应用系统开发软件叫作 ADSv1.2。ADSv1.2 提供了完整的用户开发环境和操作系统。

1. 代码生成工具

代码生成工具是一个用于 ARM 指令的 C 和嵌入式 C++编译器、汇编器和连接器，用于使用 C 和 C++语言编写的用户程序进行编译和连接，生成 ARM 内核的机器代码。代码生成工具是整个开发系统的核心。

ADS 的代码生成工具有如下特征：

⑴ 从速度和长度上对程序进行优化；

⑵ 可选择调试和优化的层次；

⑶ 可针对不同处理器进行程序优化；

⑷ 支持位置无关的代码和数据（PIC/PID）；

⑸ 目标文件和调试表格式采用 ELF 和 DWARF2 工业标准；

⑹ 编译指示器能指示程序和数据的精确位置；

⑺ 功能强大的 ARM 指令宏汇编；

⑻ 可以用符号从汇编程序直接访问 C++的类成员。

ADS 的 C 和嵌入式 C++（EC++）编译器（如图 5.1－1 所示）提供了良好的程序与 ARM 核之间的配合，因此可以有效地缩短程序的长度。此外，特殊的优化方法还提供了适合硬件内核的程序优化结果，这就进一步降低了程序的规模。

图 5.1－1 ADS 的 C＋＋编译器界面

2. Windows 下的集成开发环境

集成开发环境是专为 PC 机设计的开发软件,其编译系统以代码开发工具为核心。

ADSv1.2 是具有特色的 CodeWarrior (代码斗士) 集成开发环境 (IDE),是一种功能强大且易于使用的 ARM 应用系统开发工具。集成开发环境具有如下功能特点:

⑴ 良好的项目管理和建立系统;

⑵ 全集成多语言的文本编辑器;

⑶ 源程序和类浏览器;

⑷ 集成化的错误报告;

⑸ 根据用途区分的文件核文件夹;

⑹ 可由用户设置的界面;

⑺ 开放的 API 可以集成第三方提供的工具;

⑻ 源程序控制系统界面;

⑼ 先进的搜索引擎。

项目管理具有允许使用户管理复杂项目的 GUI,十分适合由网络组成的开发组。功能强大的内置编辑器提供了理想的程序书写环境。用户可设置的界面可以提供适合不同用户使用的项目设计环境,使用户编辑的程序更具专业化水平。项目管理主要特征是:

⑴ 可设置 C 和 C＋＋源程序的颜色;

⑵ 多层返回功能,使用户能十分方便地恢复改动过的程序;

⑶ 拖拽编辑功能;

⑷ 提供了各种不同的模板,大大地方便了用户重建新的项目;

⑸ 括号自动对应功能,减少了程序编辑错误。

此外,还有其他一些便于程序编辑的功能。

先进的搜索引擎提供了在源程序、项目和文件系统中搜索和替换的功能。对于编写大型应用系统 (项目) 来说,这是一个十分有用的功能,可大大地提高系统设计和调试速度。错误和警告对话可以迅速地指出编译错误及其位置,并允许用户直接修改错误。

值得指出的是,ADS 的 CodeWarrior IDE 只提供 PC 版本。

3. 调试器

ADSv1.2 包含了两个调试器：ARM extended Debugger (AXD) 和 ARM symbolic debugger (ARMSD)。

AXD 提供了图 5.1－2 所示的 AXD 图形界面 GUI，这实际上就是用户用来调试程序和系统硬件的界面。GUI 是一种基于 Win9x/NT4 风格的用户界面，提供了所有专业软件和系统开发人员所需要的功能，可以开发出高质量的 ARM 程序。AXD 包括：

图 5.1－2 调试器界面

⑴ 设置简单和复杂断点；
⑵ 设置观察点；
⑶ 支持硬件和软件目标；
⑷ 堆栈重卷；
⑸ 支持所有新的和已有的 ARM 核；
⑹ 改善后的窗口管理功能；
⑺ 改善后的数据显示、格式化和编辑功能；
⑻ 全集成命令行界面；
⑼ 保存调试期间的访问设置。

与 AXD 不同，ARMSD 是一个命令行调试器，包括有如下功能：

⑴ 设置简单和复杂的断点；
⑵ 设置观察点；
⑶ 支持硬件和软件目标；
⑷ 堆栈不能重卷；
⑸ 支持所有的 ARM。

此外，ARMSD 还提供了功能强大的程序片断、系统等的自动测试和验证功能。ADS 调试器的可扩展功能允许用户选择不同调试目标而不需要更换调试器。

4. 指令仿真器

指令仿真器是一个十分有用的仿真工具，可以在用户目标硬件系统或器件还没有制造好之前，用指令仿真器对所涉及的系统进行仿真，是一个加快系统开发速度的有用工具。指令仿

真器的功能如下：

⑴ 支持所有的ARM核，包括ARM9E和ARM10；

⑵ 提供基准程序；

⑶ 对中断和终止功能的方针；

⑷ 可由用户扩展对用户设置的外设仿真功能；

⑸ 能设置符合用户目标系统的存储器结构；

⑹ 定义了外部接口（实际上是一个文件）。这个外部接口允许用C或C++编写的目标硬件描述程序代替硬件系统进行仿真。可知，如果在硬件系统还没有制作好时，可以用C或C++编写一个描述硬件系统功能的程序模块，在仿真时，这个程序模块可以代替硬件。图5.1－3是外部接口设置的对话窗，通过这个对话窗，用户可以把设计好的C或C++硬件功能程序模块连接到指令仿真器中代替真实的硬件系统。

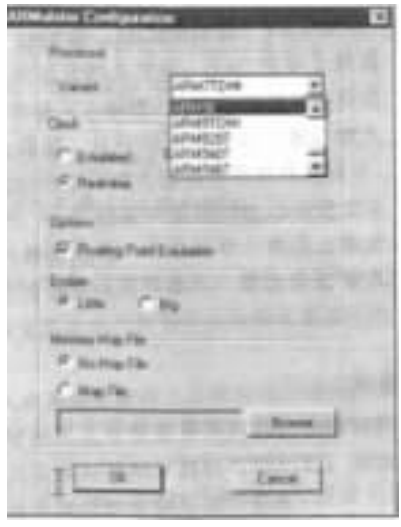


图5.1－3 外部文件接口设置对话窗

四、结束语

作为现代电子信息技术的重要实现技术，嵌入式系统和SoC技术已经成为应用系统的设计的基本技术。从应用系统设计上看，嵌入式处理器系统应用设计的关键是应用软件的设计，而软件的设计与所使用的IP内核直接有关。所以，在设计一个SoC系统需要掌握两个方面的技术，一个是所使用IP核的应用设计技术，另一个就是系统调试技术。

本文介绍了目前在通信、控制和消费电子领域中广泛使用的ARM720T内核及其开发套件ADSv1.2。ARM720T是一种RISC处理器内核，具有较高的性能价格比和低功耗特点，已经成为实际上的工程标准嵌入式内核，是工业领域SoC技术的重要分支。

参 考 文 献

1 李哲英等. 嵌入式系统与IP技术应用. 世界电子元器件，2001（11）. 17

2 骆丽等. 嵌入式系统中IP的协议专用ASIC器件电路设计. 单片机与嵌入式系统应用，2001，（9）. 17～20

3 邵贝贝等. 嵌入式RTOS讲座. 单片机与嵌入式系统应用，2001，7～9

选自《半导体技术》月刊，2002年第2期

5.2 大容量 Flash 型 AT91 系列 ARM 核微控制器

北京理工大学 徐英慧 马忠梅
Atmel 公司北京办事处 叶勇建

AT91FR40162 是美国 Atmel 公司生产的 AT91 系列微控制器中的一员,具有 ARM7TDMI 核、大容量 Flash 存储器以及片内 SRAM 和外围。这种微控制器的特点是高性能——32 位 RISC 体系结构、高密度——16 位指令集、低功耗以及实时性,扩充的 Flash 存储器还增加了开发者使用的灵活性。除此以外,大量的内部分组寄存器加速了对异常的处理过程,从而使其更适合于实时控制的应用。8 级基于向量的优先级中断控制器和外围数据控制器 PDC 大大增强了实时器件的性能。此器件适用于开发工业自动化系统、MP3、销售终端、GPS 接收机以及无线网络产品等对功耗敏感且要求具有实时性的产品。AT91FR40162 微控制器的特点是在一个 121 - ball BGA 封装中集成了 256 KB 的片内 SRAM 和 16 Mbit 的 Flash 存储器。它为许多计算密集的嵌入式控制应用领域提供了功能强大、使用灵活且性价比高的解决方案,同时还可以帮助用户减小 PCB 尺寸和系统成本。Flash 存储器可以通过 JTAG/ICE 接口或者厂家编写的 Flash Uploader 软件进行编程,从而使 AT91FR40162 适合于在系统可编程应用。

一、功能框图及产品特点

AT91FR40162 的功能框图如图 5.2 - 1 所示。

AT91FR40162 的主要特点是:ARM7TDMI 处理器核、256 KB 的片内 SRAM 和 1 024 K 字的 16 位 Flash 存储器、完全可编程的外部总线接口 EBI、具有 8 个优先级且可以独立屏蔽的向量中断控制器、32 个可编程的 I/O 口线、3 通道的 16 位定时器/计数器、2 个通用同步/异步收发器 USART、可编程的看门狗定时器、先进的省电特性、完全静态的操作、2.7 ~ 3.6 V 的 I/O 工作范围和 1.65 ~ 1.95 V 的内核工作范围、 - 40 ~ 85 的运行温度范围以及 121 - ball 10 mm × 10 mm × 1.2 mm BGA 封装(球的直径为 0.8 mm)。

二、体系结构

AT91FR40162 是由 Atmel 公司的 AT91R40008 ARM/Thumb 微控制器和 1 个 AT49BV1604A/1614A 16 Mbit Flash 存储器集成的 121 - ball BGA 封装器件。除了 Flash 存储器使能信号以外的所有地址、数据和控制信号都是内部互连的。

AT91R40008 体系结构包括 2 条主要总线:先进的系统总线 ASB 和先进的外围总线 APB。ASB 被设计为最佳性能,由存储控制器控制。ARM7TDMI 通过 ASB 与片内 32 位存储器、外部总线接口 EBI 和 AMBA 桥进行接口。AMBA 桥驱动 APB,APB 被设计用于访问片内外围并且进行了低功耗优化。

AT91FR40162 将 ARM7TDMI 处理器的 ICE 端口接到一些专用的引脚上,从而为目标调试提供了完整、低价且易用的调试解决方案。

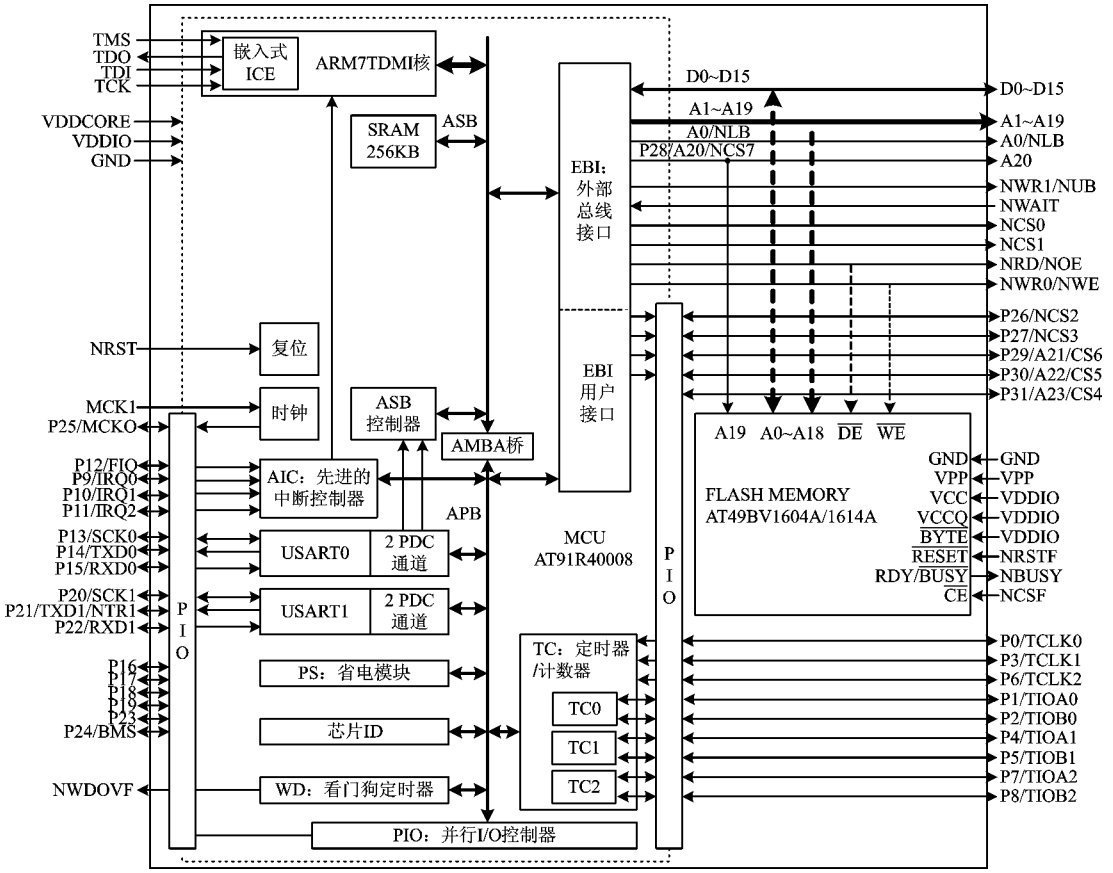


图 5.2－1 AT9FR40162 功能框图

1. **存储器**

AT91FR40162 嵌入了 256 KB 的内部 SRAM。这个内部存储器是单周期访问的，它直接与 32 位数据总线相连。这样通过使用微控制器的 ARM 指令集在 66 MHz 下可以提供 60 MIPS 的最高性能，同时降低了系统功耗。AT91FR40162 以拥有 1 个外部总线接口 EBI 为特性，它用于连接外部存储器和专用外围设备。EBI 支持 8 或 16 位器件并且可以使用 2 个 8 位器件来仿真 1 个 16 位器件。EBI 执行早读协议，与标准的存储器接口相比，能够提供更快的存储器访问速度。AT91FR40162 嵌入了 1 个由 1 024K 个 16 位字组成的 Flash 存储器，通过 EBI 可以访问它。Flash 的主要功能是作为程序存储器。1 条 16 位的 Thumb 指令可以在 1 个访问周期从 Flash 存储器被加载。分离的 MCU 和 Flash 复位输入（NRST 和 NRSTF）是为了得到最大的系统灵活性，方便用户自由地根据应用选择复位操作。AT91FR40162 集成了一个叫作 AT91 Flash Uploader 的驻留引导软件。AT91 Flash Uploader 软件能够向 Flash 存储器加载应用软件。

2. **外 围**

AT91FR40162 集成了多个外围，它们被分成 2 类：系统外围和用户外围。所有的片内外围都可以通过 AMBA 桥接受 32 位的访问。外围寄存器由控制寄存器、模式寄存器、数据寄存器、状态寄存器和使能/禁止/状态寄存器组成。外围数据控制器 PDC 在片内 USART 和片内或片外的存储器之间传输数据，并且无需处理器的介入。最重要的一点是，PDC 消除了数据

传输中断的额外开销,从而在不需要重新编程起始地址的情况下可以连续传输高达 64 KB 的数据。这样不仅增加了微控制器的性能,而且降低了功耗。

(1) 系统外围

外部总线接口 (EBI) 通过 1 条 8 位或 16 位数据总线控制外部存储器或外部设备,它通过 APB 被编程。每一条芯片选择口线有它自己的编程寄存器。省电模块 (PS) 实现空闲模式 (ARM7TDMI 核时钟停止直到有下一个中断),并且允许用户根据应用需求调整微控制器的功耗 (由独立的外围时钟控制)。先进的中断控制器 AIC 控制来自内部外围的中断源和 4 个外部中断口线 (包括 FIQ),从而为 ARM7TDMI 提供中断或/和外部中断请求。它通过集成 1 个 8 优先级中断控制器以及使用自动向量特性,降低了中断延迟时间。并行输入/输出控制器 PIO 控制高达 32 条 I/O 口线。它使用户可以选择特定的引脚作为片内外围的输入输出,或者作为通用的输入/输出信号。可以编程 PIO 控制器来检测每条线上的信号发生变化引起的中断。看门狗定时器 WD 用于防止当软件进入死锁陷阱时产生系统锁定。特殊功能 SF 模块集成了芯片 ID、复位状态和保护寄存器。

(2) 用户外围

2 个独立配置的 USART 可以高波特率进行同步或异步通信。每一个 USART 还包含 1 个超时寄存器和 1 个时间确保 (Time Guard) 寄存器,从而方便了 2 个专用外围数据控制器 PDC 通道的使用。

3 通道 16 位的定时器/计数器 TC 是高度可编程的,它支持捕获或波形模式。每一个 TC 通道可以被编程为测量或生成不同类型的波形,并且可以检测和控制 2 个输入/输出信号。TC 有 3 个外部时钟信号。

三、AT91FR40162 性能概述

AT91FR40162 主要由 AT91R40008 和 16 Mbit 的 Flash 存储器组成。因此 AT91FR40162 的许多性能与 AT91R40008 是一样的。

1. 电 源

AT91R40008 微控制器有 2 种类型的供电引脚:

(1) VDDCORE 引脚,它为内核提供电源 (包括 ARM7TDMI、嵌入的存储器和外围);

(2) VDDIO 引脚,它为 I/O 线提供电源。

一个独立的 I/O 电源允许灵活地进行调整以适应外部元件的信号电平。

2. 输入/输出

AT91FR40162 的 I/O 引脚所接受的电平以 VDDIO 的供电为限。复位以后,微控制器的外围 I/O 初始化成输入以提供给用户最大的灵活性。在每个应用阶段,微控制器的输入最好保持在有效的逻辑电平,以降低功耗。

3. 主时钟

如果 MCKI 引脚由 1 个外部源提供,那么 AT91FR40162 则为一个完全静态的设计,并工作于主时钟 MCK (Master Clock) 下。

主时钟还在引脚 MCKO 上作为器件的输出,这个引脚与一个通用 I/O 线复用。当 NRST 处于活动状态并且发生复位后,MCKO 有效并输出 MCK 信号的映像。必须编程 PIO 控制器来使用这个引脚作为标准 I/O 线。

4. 复 位

复位操作将使用户接口寄存器恢复其默认状态(在每一个外围的用户接口中定义),并强制 ARM7TDMI 从地址 0 执行对下一条指令的取指操作。除了程序计数器之外的其他 ARM7TDMI 寄存器则没有定义复位状态。

5. NRST 引脚

NRST 是低电平激活的输入引脚。它被异步激活,但是从复位退出是与 MCK 内部同步的。MCKI 上出现的信号必须在 NRST 上升沿之前最少 10 个时钟周期内有效,以确保操作的正确性。第 1 个取指操作出现在 NRST 上升沿之后的 80 个时钟周期。

6. 看门狗定时器复位

看门狗定时器可以被编程用来产生 1 个内部复位。在这种情况下的复位操作与激活 NRST 引脚有同样的效果,但是引脚 BMS 和 NTRI 不被采样。引导模式和三态模式不被更新。如果 NRST 引脚被激活并且看门狗定时器触发内部复位,那么 NRST 引脚具有优先权。

7. 仿真功能

(1) 三态模式

AT91FR40162 微控制器提供了 1 个用于调试目的的三态模式。它使能仿真探头与目标板连接,而不必先从目标板焊下器件。在三态模式下,AT91R40008 微控制器的所有输出引脚驱动器都是禁止的。在三态模式下,提供通过外部引脚对 Flash 的直接访问。这就使得在装配板子之前可以使用传统的 Flash 编程器对 Flash 进行编程。为了进入三态模式,NTRI 引脚必须在 NRST 上升沿之前的最后 10 个时钟周期内保持低电平。对于正常的操作,NTRI 引脚必须通过 1 个高达 400 kΩ 的电阻复位为高电平。NTRI 与 I/O 线 P21 和 USART1 串行数据传输线 TXD1 多路复用。

(2) JTAG/ICE 调试

JTAG/ICE 端口支持 ARM 标准的嵌入式在线仿真。引脚 TDI、TDO、TCK 和 TMS 专门用于这种调试功能,并且可以通过外部 ICE 接口与一个主机相连。在 ICE 调试模式下,ARM7TDMI 核以非 JTAG 芯片 ID 作为响应来标识微控制器。这一点不完全符合 IEEE1149.1 标准。

8. 存储控制器

ARM7TDMI 处理器的地址空间为 4 GB。存储控制器对内部的 32 位地址总线进行译码并定义了 3 种地址空间:位于最低 4 MB 的内部存储器空间;中间的地址空间为 EBI 控制的外部设备(存储器或外围)所保留;位于最高 4 MB 的内部外围空间。不论在哪一个地址空间,ARM7TDMI 只运行于小端模式。

(1) 内部存储器

AT91FR40162 微控制器集成了内部 SRAM。所有的内部存储器都是 32 位宽,并且支持字节(8 位)、半字(16 位)和字(32 位)访问,且这些访问都是单周期的。内部 SRAM 支持 Thumb 和 ARM 两种取指方式,并且内部存储器可以存储比 ARM 指令多 1 倍的 Thumb 指令。AT91FR40162 微控制器集成了 1 个 256 KB 的 SRAM 存储体。这个存储体映射到地址 0x0(重映射命令后),并且允许通过软件修改 0x0 到 0x20 之间的 ARM7TDMI 异常向量。把 SRAM 放置在片内并使用 32 位的数据总线带宽增加了微控制器的性能,降低了系统功耗。32 位总线与 16 位相比带宽增加,从而提高了使用 ARM 指令集和数据处理的效率,这是对 ARM7TDMI 先进性能的最佳使用。能够在 256 KB 的 SRAM 中动态地更新应用软件是 AT91FR40162 的另

一特性。

(2) 引导模式选择

ARM 复位向量在地址 0x0。NRST 口线被释放后,ARM7TDMI 执行存储于这个地址的指令。这就意味着复位之后这个地址必须映射到非易失的存储器中。NRST 上升沿之前的 10 个时钟周期期间 BMS 引脚上的输入用来选择引导存储器的类型,如表 5.2-1 所列。如果嵌入的 Flash 存储器用作引导存储器,BMS 输入必须被外部拉低,并且 NCS0 必须在外部与 NCS7 连接。

表 5.2-1 引导模式选择

BMS	引导存储器
1	NCS0 上的外部 8 位存储器
0	NCS0 上的外部 16 位存储器

引脚 BMS 与 I/O 线 P24 复用。复位以后,P24 可以与任何标准的 PIO 线一样被编程。

(3) Flash 存储器

16 Mbit 的 Flash 存储器由 1 048 576 个 16 位字组成。Flash 存储器通过 EBI 进行 16 位字寻址。它使用地址线 A1 ~ A20。

除了 Flash 存储器使能信号以外,所有的地址、数据和控制信号都是内部互连的。用户应当把 Flash 存储器使能信号 (NCSF) 和 1 个 EBI 上低电平有效的片选信号相连接。如果 Flash 存储器作为引导存储器使用,那么必须使用 NCS0 ;同时 BMS 必须被外部拉低,以使处理器在复位后可以执行正确的 16 位取指操作。

引导时,必须为 EBI 配置正确的标准等待状态数目。例如,当微控制器运行于 66 Hz 时,需要 5 个标准等待状态。

用户必须确保所有的 VDDIO、VDDCORE 和所有的 GND 引脚通过最短的路线连接到各自的电源。Flash 存储器在读模式下加电。命令序列用于将此器件置于其他的操作模式下,例如编程和擦除。

为了增加灵活性而提供的分离的 Flash 存储器复位输入引脚 (NRSTF),使能复位操作以适应具体应用。当这个输入为逻辑高电平时,存储器处于它的标准操作模式 ;当这个输入为逻辑低电平时,暂停当前的存储器操作并使它的输出置于高阻状态。

Flash 存储器的一个特性是它采用数据查询来检测一个编程周期的结束。当处于编程周期时,一个试图读取最后一个写入字的操作在 I/O7 上返回写入数据的补码值。开漏 NBUSY 输出引脚提供另一种检测编程周期或擦除周期是否结束的方法。当处于编程或擦除周期时,这个引脚被拉低,这个周期结束以后引脚恢复高电平。使用 1 个翻转位提供了检测编程或擦除周期是否结束的第 3 种方法。

Flash 存储器被分为 2 个存储区 (plane)。当从一个存储区执行读操作时,在另一个存储区可以同时执行编程或擦除功能。这一特性使得在执行读操作之前不需要系统等待一个编程或擦除周期的完成,从而增强了系统的性能。

为了方便擦除操作,Flash 存储器被分成了 39 个扇区。为了进一步增强器件灵活性,还提供了擦除挂起特性。这一特性使得擦除周期被挂起一段不确定的时间,并允许用户执行从同一个存储区内任何一个其他扇区的读数据操作或向其写数据的操作。如果所读的数据在另一个存储区,则不需要挂起擦除周期。

此器件具有保护存储在任一扇区中的数据不被破坏的能力。一旦对于某个扇区的数据保护被使能,那么,当输入电平处于地和 VDDIO 之间时,那个扇区的数据不能被改变。

1 个可选的 VPP 引脚用于增加编程/擦除次数。

以 6 字节命令序列进入的单脉冲编程模式(Single Pulse Program Mode)允许器件使用写控制线上的单脉冲直接被写入。通过器件掉电或者使 NRSTF 引脚为低电平并至少保持 50 ns,然后将其恢复为 VDDIO,退出单脉冲编程模式。

以下硬件特性保护 Flash 存储器,避免其不小心被编程:

· VDDIO 敏感——如果 VDDIO 低于 1.8 V(典型情况),编程功能被禁止;

· VDDIO 上电延迟——一旦 VDDIO 到达 VDDIO 的敏感电平,器件将自动地在编程之前超时 10 ms(典型情况);

· 编程禁止——保持 OE 为低、CE 为高或 WE 为高中的任意一个,将禁止编程周期;

· 噪声滤波——出现于 WE 或 CE 输入上的低于 15 ns(典型情况)的脉冲将不会启动编程周期。

四、AT91 Flash Uploader 软件

所有基于 Flash 的 AT91 器件都配备一个叫作 AT91 Flash Uploader 的预编程软件,它驻留在嵌入的 Flash 存储器的第 1 个扇区。Flash Uploader 允许通过串口向嵌入的 Flash 编程。Flash Uploader 可以使用任一个片内 USART。Flash Uploader 的运行环境如图 5.2-2 所示。

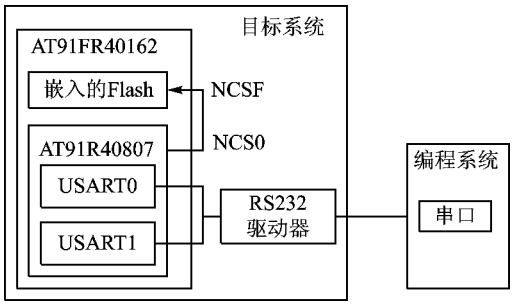


图 5.2-2 Flash Uploader 运行环境

1. Flash Uploader 操作

Flash Uploader 软件需要使用嵌入的 Flash 存储器作为引导存储器,还需要为 MCKI 提供 1 个有效的时钟。复位以后,Flash Uploader 立即复制软件本身到内部的 SRAM 并跳转到那里。随后的操作将只需要这个存储器资源。所执行的外部访问只是编程嵌入的 Flash 存储器。

开始以后,处理器初始化 2 个 USART 的 RXD 引脚的 PIO 输入变化中断。中断出现时,启动 1 个定时器/计数器通道;当在 RXD 线上检测到下一个输入变化时,则停止这个定时器/计数器通道。这就解释了如何测量第 1 个字符长度,以及如何通过计算器件主时钟和实际通信时的波特率速度之间的比率来初始化 USART。然后,编程系统便可以按照编程 Flash 器件所规定的协议发送命令和数据。为了最低限度地降低 Flash Uploader 被擦除和电源被关闭的危险,最后才由编程系统来擦除和编程 Flash 的第 1 个扇区。当 Flash Uploader 从第 1 个扇区被擦除时,如果新的最终应用程序还没有被编程并且目标系统的电源被关闭,那么,将导致一个不可恢复的错误,并且 AT91FR40162 不能再通过使用 Flash Uploader 被重新编程。

2. 编程系统

Atmel 公司提供了一个免费的 Host Loader 软件,运行于 Windows 95 或 Windows 98 操作系

统下的 IBM PC 兼容机上。可以从 Atmel 网站下载此软件,并且只需要 1 根串行电缆线来连接主机和目标机。可以选择使用 COM1 或 COM2 实行通信,同时串行链路的速度被限制为 115 200 baud。由于串行链路速度是配置瓶颈,因此,Flash 编程每 1 MB 需要持续 110 s。

通过使用一个快速的编程系统可以减少编程时间。AT91 评估板可以使一个串行链路运行到 500 K 位/s,并且可以匹配 Flash 所允许的最快编程速度。例如,当字编程成为瓶颈时,编程速度可以达到 40 s/MB。

四、结束语

随着后 PC 时代的进入,嵌入式系统已经无所不在。手机、汽车、航空航天以及军事装备等各个领域都能看到嵌入式系统的身影;而微控制器则正是嵌入式系统的核心部件。AT91FR40162 是 Atmel 公司 AT91 系列微控制器中的一种,这一系列的微控制器是低功耗、32 位性能和 16 位系统价位的最佳组合。AT91FR40162 在 AT91R40008 的基础上增加了 16 Mbit 的 Flash 存储器,Flash 存储器的可编程性为用户使用该芯片提供了更大的灵活性。AT91FR40162 为嵌入式应用提供了又一种可选的性价比较高的微控制器。

参 考 文 献

1 Atmel. AT91FR40162 Advance Information. 2001 - 11

2 Atmel. AT91X40 Series. 2002 - 05

选自《单片机与嵌入式系统应用》月刊,2002 年第 10 期

5.3 内嵌 UHF ASK/FSK 发射器的 8 位微控制器

南华大学 黄智伟 万碧根 李金龙

一、概 述

rfPIC12C509AF 是 Microchip 公司推出的单片集成内嵌射频无线数据发射器的 8 位 CMOS 微控制器。芯片具有高性能的 RISC 中央处理器,33 条 12 位字长的指令,8 位字长的数据;内置 4 MHz RC 振荡器,运行速度 1μs 指令周期;7 个特殊功能的硬件寄存器,2 级硬件堆栈,直接、间接和相对寻址方式;1024 × 12 bit 可编程 EPROM,41 字节数据 RAM;在线串行编程 (In-Circuit Serial Programming™,ICSP™),内部 RC 振荡器的频率可编程校准 (独立于发射器的石英晶体振荡器基准),8 位可编程定时器/计数器;上电复位,看门狗定时器,低功耗睡眠模式,可编程编码保护,5 个通用 I/O 等功能;工作电压 2.5 ~ 5.5 V,低功耗睡眠模式电流 0.2 ~ 4μA。内嵌的 UHF ASK/FSK 发射器,射频频率范围为 310 ~ 480 MHz,可调节的输出功率 +2 ~ - 12 dBm,ASK 数据发射速率 0 ~ 40 Kbps,FSK 数据发射速率 0 ~ 20 Kbps,PLL 锁相,集成的晶体振荡器和 VCO 电路仅需少量的外部元件。

可用于遥控无键入口 (RKE) 发射器、车库门开门器、遥测 (轮胎压力,水、电、气表,贵重物品跟踪)、无线安防系统、无线电遥控等领域。

二、引脚排列及功能

rfPIC12C509AF 采用 20 脚 SSOP 封装,各引脚功能如下。

(1) VDD:逻辑电路和 I/O 脚的正电压端。

(2) GP5/OSC1/CLKIN:双向 I/O 端口/石英振荡器输入/外部时钟输入 (GPIO 仅在内部 RC 模式,在其他振荡器模式下为 OSC1)。当 GPIO 时 TTL 输入,在外部 RC 振荡器模式时 ST 输入。

(3) GP4/OSC2:双向 I/O 端口/石英晶体振荡器输出。在石英晶体振荡器模式时连接晶振或谐振器。

(4) GP3/MCLR/VPP:输入端口/用户清除 (复位) 输入/编程电压输入。当构成$\overline{\text{MCLR}}$时,此脚是低电平有效,实现器件复位。在设备进入正常的运行和编程模式时,$\overline{\text{MCLR}}$、VPP 上的电压不能超过 VDD,并且能够通过软件编程改变引脚状态来唤醒睡眠状态。

(5) XTAL:发射器晶振,连接到考比慈 (COPITTS) 型晶体振荡器上。

(6) RFENIN:发射器和时钟输出使能,内部下拉。

(7) CLKOUT:时钟输出。

(8) PS/DATAASK:功率选择和 ASK 数据输入。

(9) VDDRF:发射器正电压端。

(10) ANT1:差分功率放大器的输出端连接到天线,集电极开路输出。

(11) ANT2 :差分功率放大器的输出端连接到天线,集电极开路输出。

(12) VSSRF :发射器接地参考端。

(13) NC :空脚。

(14) LF :连接外部回路滤波器。VCO 转换输入和充电泵输出的共用点。

(15) DATAFSK :FSK 的数据输入。

(16) FSKOUT :FSK 晶振的输出。

(17) GP2/T0CKI :双向 I/O 端口,能构成 T0CKI。

(18) GP1 :双向 I/O 端口/串口编程时钟,能通过软件编程改变引脚状态来唤醒睡眠状态。这个缓冲器在串口编程模式下为施密特触发器输入。

(19) GP0 :双向 I/O 端口/串口编程数据,能通过软件编程改变引脚状态来唤醒睡眠状态。这个缓冲器在串口编程模式下为施密特触发器输入。

(20) VSS :逻辑电路和 I/O 脚的参考地。

三、基本结构和特性

rfPIC12C509AF 内部结构包括一个完整的 8 位 CMOS 微控制器电路和发射器电路,以下介绍发射器电路。发射器电路方框图如图 5.3-1 所示。

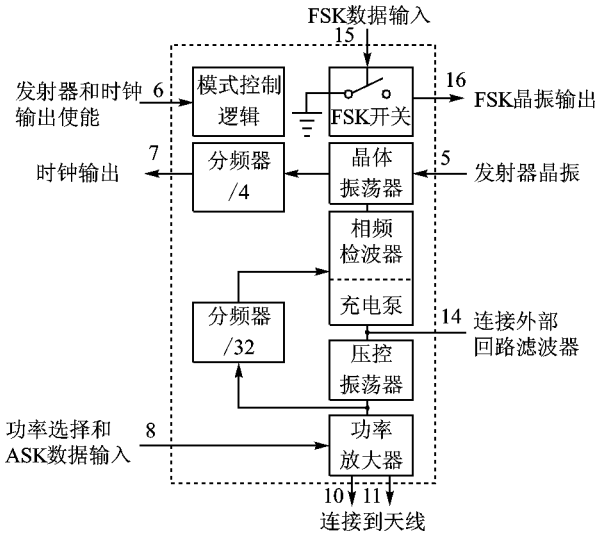


图 5.3-1 rfPIC12C509AF 发射器电路方框图

发射器是一个完整的集成 UHF ASK/FSK 发射电路,由石英晶体振荡器、锁相环电路(PLL)、集电极开路的输出功率可变放大器 PA(Power Amplifier)和模式控制逻辑(mode control logic)所组成。外接元件有旁路电容、晶振和 PLL 回路滤波器,能实现 ASK 和 FSK 的操作。

引脚 VDDRF 和 VSSRF 分别是发射器电路的电源供给端和接地端。这些电源脚与微控制器的电源供给脚 VDD 和 VSS 是相互独立的。

发射器的石英晶体振荡器是一个考比慈振荡器,提供 PLL 的基准频率,并且与 PIC 微控制器的振荡器是相互独立的。XTAL 脚上接外部振荡器或 AC 模拟基准信号。发射频率是由

晶振频率确定的,公式如下:

$$f_{transmit} = f_{XTAL} \times 32$$

考虑到发射频率的灵活选择,最终晶振频率可能不是标准值。晶振频率最小值为 9.65 ~ 15 MHz,负载电容 10 ~ 15 pF,并联电容 7 pF,等价串联阻抗 60 Ω。

rfPIC12C509AF 晶体振荡器实现 ASK 操作电路如图 5.3 - 2 所示。电容器 C1 取值 22 ~ 1 000 pF。

rfPIC12C509AF 晶体振荡器实现 FSK 操作电路如图 5.3 - 3 所示。电容 C1 和 C2 通过拖动晶振来实现 FSK 调制。当 DATAFSK = 1 时,FSKOUT 为高阻抗,只有 C1 对晶振起作用,发射频率为 f_{MAX} ;当 DATAFSK = 0 时,FSKOUT 与 VSSRF 接地,电容 C1 和 C2 并联,发射频率为 f_{MIN}。选择一组理想的 C1 和 C2 值来确定中心频率和频率偏差。电容 C1 确定 f_{MAX},而电容 C1 和 C2 的并联值确定 f_{MIN}。

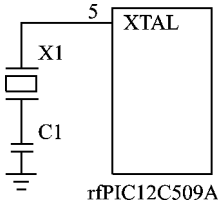


图 5.3 - 2 ASK 方式外接晶体振荡器与电容器

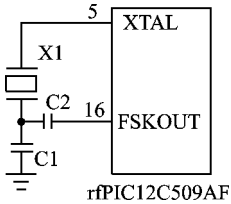


图 5.3 - 3 FSK 方式外接晶体振荡器与电容器

C1 取值 22 ~ 1 000 pF,C2 取值 47 ~ 1 000 pF。

发射器中心频率 (f_C) :$f_C = (f_{MAX} + f_{MIN})/2$

发射器频率偏差 :$\Delta f = (f_{MAX} - f_{MIN})/2$

石英晶体振荡器有 1 个四分频 (Divide by 4) 电路,此电路通过时钟输出 (CLKOUT) 引脚输出时钟。时钟输出信号可作为微控制器的输入或其他外围电路的稳定基准频率。注意千万不要将 CLKOUT 信号连接到 PIC 微控制器的 OSC1 输入端,因为 PIC 微控制器没有时钟信号就不能工作,此时发射器的振荡器也不能工作。这时 PIC 微控制器需要从外部引入时钟或经过内部 RC 振荡器产生时钟。当应用中需要稳定的基准频率时,可将 CLKOUT 脚连接到 GP2/T0CKI 输入上,并且使用 TIMER0 模块。为了使干扰信号尽可能小,应对 CLKOUT 有速率限制。CLKOUT 的电压幅值由在 CLKOUT 脚上的充电电容决定 (2VPP,5 pF)。

锁相环电路 (PLL) 由相频检波器 (phase frequency detector)、充电泵 (charge pump)、压控振荡器 VCO (Voltage Controlled Oscillator) 和固定的 32 分频器 (fixed divide by 32) 组成。引脚 LF 连接 1 个外部回路滤波器。这个回路滤波器控制 PLL 的动态范围和起始锁定时间。

PLL 的输出给功率放大器 (PA)。集电极开路输出的不同值可直接驱动闭环天线 (ANT1、ANT2) 或经过 1 个阻抗匹配网络或平衡 - 不平衡变换器改变成单端口输出。引脚 ANT1 和 ANT2 为集电极开路输出,必须通过负载上拉到 VDDRF。

PA 的差动输出应该匹配 1 个 1 kΩ 的负载电阻。当匹配不合理时会导致过度的干扰和谐波辐射。发射输出功率可以通过改变 PS/DATAASK 脚的电压调节成 +2 ~ - 12 dBm 中的 6 个等分值。

在 FSK 的操作中,PS/DATAASK 脚只能作为功率选择脚 (PS) 使用。1 个 20μA 的内部电

流源输出电流流入 PS/DATAASK 脚,通过电阻 R2 产生一个电压降,作为功率控制电压 (VPS) 控制发射输出功率。VPS 控制 PA 的偏置电流,高的发射功率需要较大的偏置电流。

为了实现 ASK 操作,PA/DATAASK 脚的功能是控制功率放大器 PA 导通或关断。分压网络上的 R1 和 R2 是为了确定 VPS,以达到选择发射器输出功率的目的。假如要得到最大发射器输出功率,可以把引脚 GP0 和 PA/DARAASK 直接连接起来。

逻辑控制模式引脚 RFENIN 控制着发射器的操作。当 RFENIN = 1 时,发射器和 CLKOUT 在工作模式;当 RFENIN = 0 时,发射器和 CLKOUT 进入待机模式。在待机模式时,发射机产生很小的电流。REFNIN 脚在内部有 1 个下拉电阻。

四、应用电路

一个 FSK 的应用电路如图 5.3 - 4 所示,工作频率 433.92 MHz,输出功率 + 2 dBm。电路可根据控制输入信号发射微控制器内的数据。

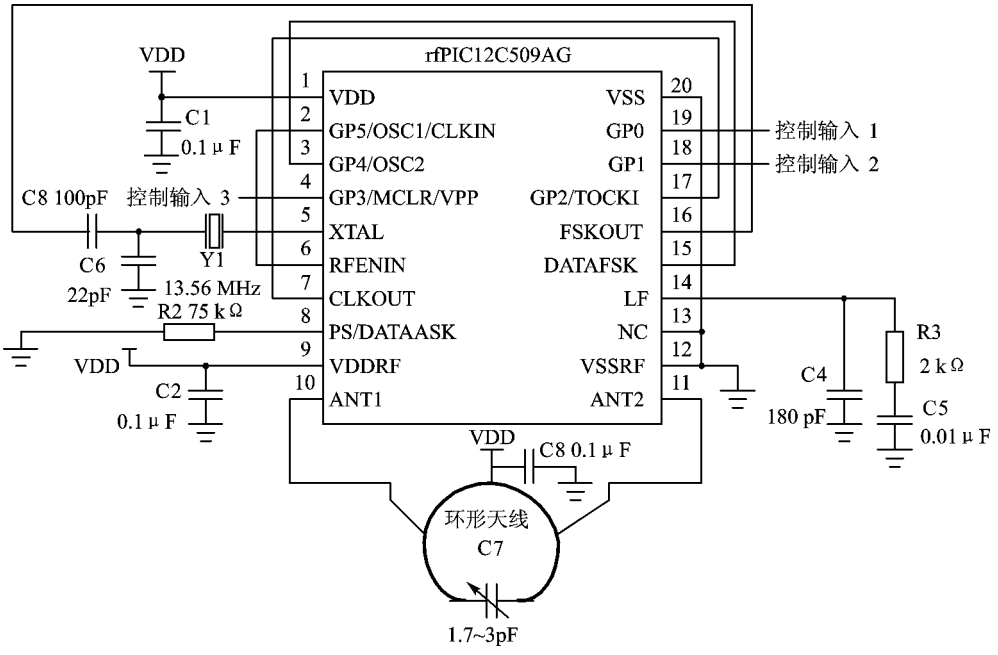


图 5.3 - 4 FSK 应用电路

设计印制电路板时应注意:需要提供 1 个低阻抗电源和最小噪声辐射的地线。要求使用双面 PCB 板,并把地线平面放在底层以减少无线电的辐射和串扰;旁路电容应尽量靠近每个电源引脚 VDD 和 VDDRF;用 1 个单独的 PCB 通孔连接 VSS 和 VSSRF,千万不要把 PCB 通孔与复合地线相连;为减少电路中的分布电容,应避免平行线路的出现;线路应越短越好;为防止耦合,应独立其各组成部分;使用接地线使各信号隔离;用地线来屏蔽时钟输出线,隔离 CLKOUT 信号和减少耦合;回路滤波器的组成部分尽可能地放在离 LF 脚近的地方,并保持线路尽可能短;发射天线可印制在 PCB 上。

5.4 专用单片机 C504－2E 在 SPWM 技术中的编程技巧

西安西北工业大学 (710068) 金 雍 汤 力 孙文福 孙国强

一、前 言

C504 系列单片机是德国 SIEMENS 公司推出的用于交/直流电机控制的专用单片机,是结构较新、性能强、编程简单、适应性强的新型复合单片机。C504－2E 内部具有 16k×8 空间的 OTP (One-tome programmable) 存储器,适于小批量或初次编程之用。典型的 C504 单片机其内部结构如图 5.4－1 所示。它的最大特点是:① CPU 及指令系统完全和 8051 兼容;② 具有片内 MASKROM 和 OTP 存储器,容量达 16 KB;③ 256 字节 RAM 和额外的 256 字节的 XRAM;④ 有一个功能强大的 CCU (Capture/Compare Unit) 单元,用来产生驱动三相电机 (交流、直流无刷电机) 的信号 控制脉宽的定时器 1 为 16 位,使脉宽精度超过一般同类单片机;⑤ 一组具有自校检的 10 比特 8 路 A/D 转换器;⑥ 具有两个优先级的 12 个中断源。因此,极适应于变频调速的电控系统。近年来,我们将 C504 成功地应用于变频空调电控系统。下面针对其编程方法和体会进行介绍。

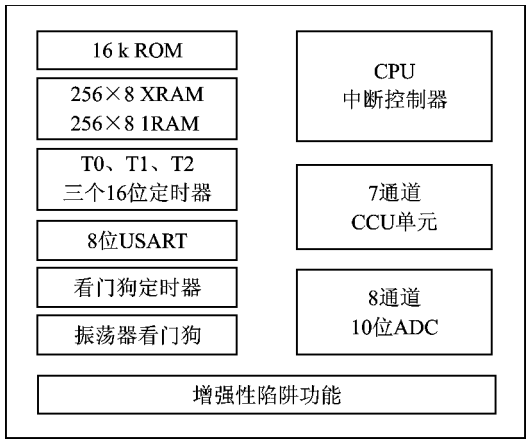


图 5.4－1 C504 内部结构

二、C504 中 CCU 工作原理

图 5.4－2 给出了 CCU 内部结构图,从图中可以看出,CCU 主要分为两部分。

图的上半部分为捕捉/比较单元 (CAP/COM)。其中 CCU I/O 在比较模式下,每个通道有两路极性相反的输出,分别控制一个桥臂的上下开关管。一共 6 路,控制完整三相全桥。比较定时器 1 输入时钟在 $f_{osc}/2 \sim f_{osc}/256$ 范围内任意设置。该定时器有两种工作模式 模式 0 和模式 1。模式 0 为正计数,并复位,可以形成单边调制 PWM 波形 ;模式 1 为正计数,然后倒计数,

可形成双边调制的 PWM 波形。在形成 PWM 波形时,输出极性可以用编程方法实现正/反极性转换,有利于和驱动器件的配合。图中偏置寄存器用于自动产生死区,其值可以任意设置。

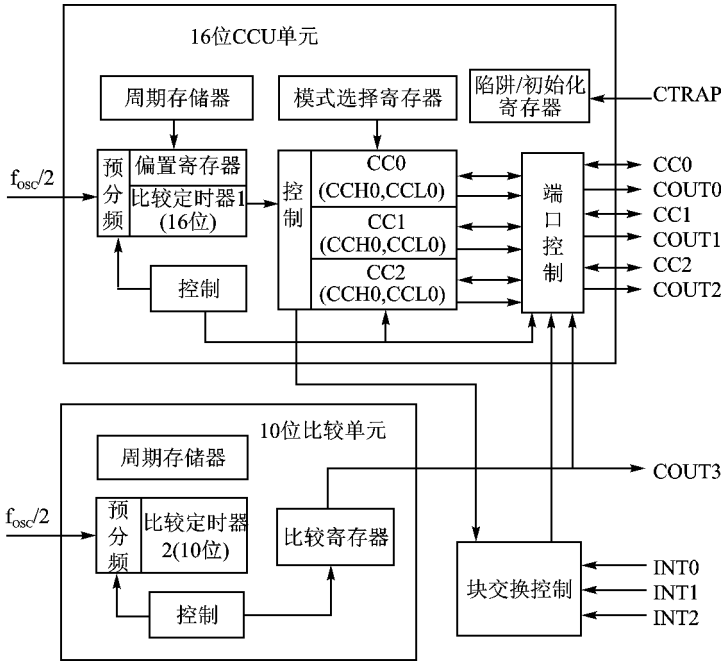


图 5.4－2 CCU 内部结构图

图 5.4－2 下半部分为 10 位比较单元,可以单独输出一路 PWM 信号,输出端口为 COUT3。比较定时器 2 的工作频率范围同比较定时器 1。该部分往往和前部分联合使用,形成多路 PWM 模式。在三相交流模式时,该部分不用。

下面,针对我们研制的三相交流变频空调控制系统,介绍 CCU 编程方法。

三、产生 SPWM 波形软件的编制

C504 的主要任务是产生频率可变的三相交流正弦波,且要在极短的限定时间内完成。

1. 比较定时器 1 中周期和偏移量的计算

在 CCU 中,PWM 周期是由比较定时器 1 设定的,要计算比较定时器 1 的值,首先必须确定 PWM 周期,并通过 CPU 存于周期存储器中,例如选定 $f_{PWM}=20$ kHz,也就是说每 50 μs 比较定时器 1 就要中断一次。其次,CCU 中形成 SPWM 中不同脉冲宽度的比较寄存器根据正弦波的需要定时更新内容,这些比较值在极短的时间内无法算出,要周期地从存储器中的正弦表中读出。该比较值为 8 bit,所以比较定时器 1 的周期值和偏移量不需超过 255 就能满足任意调制度的要求。比较定时器 1 的设定值可通过下式计算(此值存于周期寄存器中):

$$N=\frac{f_{osc}}{\text{预分频数}\times f_{PWM}\times 2}=\frac{40\ \text{MHz}}{4\times 20\ \text{kHz}\times 2}=250 \tag{1}$$

式中,预分频系数选 4 是为了提高对不同周期的适应范围;分母系数 2 是满足比较定时器 1 工作于模式 1 时,双向计数的原因。当预分频系数选 4 后,则计数器每隔 0.1 μs (1/10 M) 增加

1,若死区时间定为 1 μs 时,则偏移量为(此值存于偏置寄存器中):

$$\text{偏移量} = \frac{\text{死区时间} \times f_{osc}}{\text{预分频数}} = \frac{1\ \mu s \times 40\ MHz}{4} = 10 \tag{2}$$

至此,在 f_{PWM} = 20 kHz,死区时间为 1 μs 时的定时器 1 设定值 N(250)和为产生死区的偏移量(10)计算完毕。用同样的方法,可以设置任意周期值和死区值。

2. 关于正弦表读出指针的求解

为了在较短的时间内查到正弦表中相应的值,则正弦表一个周期的长度不应超过 256 位。因为指针(亦称角度变量)可用 8 bit 值实现,这对 8 位单片机 C504 来说是非常容易的事。为了产生正弦波,则角度变量需每中断一次其值增加 1,从而使正弦表指针也不断增加,并将表中的值读出,送入比较寄存器。用这种方法,可产生的最低频率达 1 Hz。根据上面的计算,角度变量每 50 μs(0.1 μs × 250 × 2)增加 1,并从正弦表读出一个脉宽量,这时输出的正弦波频率为:

$$f_{OUT} = \frac{1}{256 \times 50\ \mu s} = 78.125\ Hz \tag{3}$$

因之,当需要 f_{OUT} > 150 Hz 的场所,必须选择预分频系数为 2。同样,要使频率低于 78.125 Hz,就必须选择 16 bit 的角度指针变量,并且让其高 8 位所确定的值对应正弦表。例如,对于 16 bit 角度变量指针,若每增加 1 仍然是 50 μs,则计完 16 位需要的时间为 65 536 × 50 μs = 3.275 s,即最低频率为 0.305 Hz。这样就可以完全满足低频范围的要求。为了获得任意频率值,由于计数器每 50 μs 增加 1,因之必须改变 16 位计数器的计数增量值。下面引入表示计数增量的 Δ 变量:

$$\Delta = \frac{f_{OUT}}{0.305\ Hz} = f_{OUT} \times 65\ 536 \times 50\ \mu s = \frac{f_{OUT} \times 65\ 536}{20\ kHz} \tag{4}$$

定义 P 为正弦表指针,则:

$$P(k) = P(k-1) + \Delta \tag{5}$$

显然,随着 Δ 值的增大,可以使输出脉冲数减少,图 5.4－3 用图解的方法指出用上面介绍的方法产生正弦波的过程,以便理解。

3. 求解调制度的技巧

如果用 C504 中 8 位 CPU 完成乘法运算,需要许多机器周期,致使占机时间大大增加,这种情况在整个运算过程中不是都能接受的。进一步讲,如果要提高精度,就需要更强大的计算能力,这意味着单片机的升级和价格的升高。下面将介绍一种技巧算法,它是利用加法来完成正弦波的计算,从而可以节省大量乘法运算所需的时间。假设供给三相感应电动机的正弦波可用如下公式表示:

$$U = A \times \sin B \tag{6}$$

其中: B 对应角度变量的高位字节(即正弦表角度指针);

sin B 是正弦表中之值;

U 表示最后送到比较寄存器的值。

为了避免乘法运算,将式(6)利用三角函数变换,有:

$$U = A \times \sin B = \cos(\arccos A) \times \sin B = \cos A' \times \sin B$$

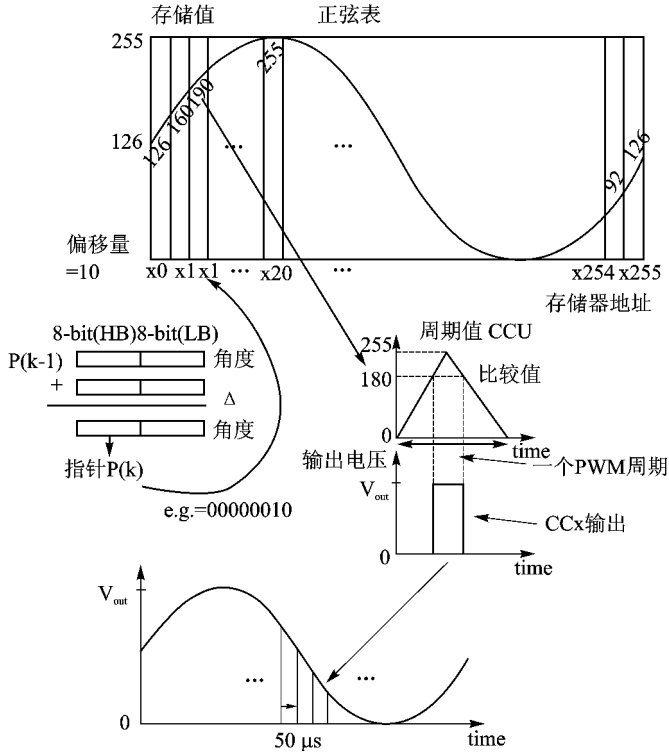


图 5.4－3 **产生正弦波计算过程的图示**

$$= \frac{1}{2}[\sin(B - A') + \sin(B + A')], \quad A' = \arccos A \tag{7}$$

通过式 (7) **可以看出,已经将乘法运算转化为加法运算。只要求出** B - A'、B + A',**然后分别查正弦表,把两次查表值相加,然后再除以** 2 **即可。**

这种算法的具体操作如下 在给定振幅 A **和角度的情况下,先按上面介绍的方法算出** Δ **和角度变量高位字节** (B **值) 然后,求得** A' (= arccosA,**从表中查得) ;最后算出** (B + A')、(B - A')。**这时,**B + A'、B - A' **就是新的正弦表指针。通过两个指针值,经查正弦表所得值相加,再被** 2 **除就得到最终结果** (U **值)。将** U **送入比较寄存器,就完成了一个** PWM **脉冲的计算。另外,为了省去每次除** 2 **的麻烦,可直接通过修正正弦表值的方法一次完成。**

关于振幅精度的一个值得考虑的问题是 :上面提到的 B **代表角度变量的高** 8 **位,取值范围在** 0 ~ 255 **之间,覆盖** 360°**范围,从而对应了整个正弦表。振幅量** A **包含了** 0 ~ 100% **的调整范围,而** arccos A **只对应** 0 ~ 90°**范围,是** 360°**的** 1/4。**由于正弦表范围为** 0 ~ 255 (8 **位字节),** A' **只能覆盖** 0 ~ 64,**这也就是说,其分辨度只能用** 6 bit **描述,从而使振幅分辨度只有** 100% /64 = 1.56%。**产生这种缺点的原因是由于正弦表的存储空间和振幅分辨度紧密相关的缘故。为了提高精度,就必须使正弦表的长度和指针的长度超过** 8 bit。**但是,要使指针长度超过** 8 bit,**对于** C504 **这样的** 8 **位单片机来说,比较困难,要较多的增加** CPU **的处理时间。另一方面,一般感应电动机对角度和分辨率没有过高的要求。过分地增加分辨度将会导致不必要的** CPU **负担。还不如把这个占机时间用来完成电流闭环控制,这是对于** 8 **位机在运算能力和分辨度之间的比较理想的折衷。**

4. 存储器正弦表的生成

由于正弦表四分之一周期的变化规律，完全代表了全周期的变化。因之，在文件 sine-src 中，给出了 0～90°的正弦表，它占用了 255 个字节。这意味着：全周期的正弦表占 1 024 字节，寻址需要 10 位的指针变量。在本文三．3 节中推荐的应用中，正弦表的全部长度为 256 字节，适用于 8 位指针变量。那么，怎样由 1 024 字节的正弦表产生一个 256 字节的正弦表呢？程序中的 init_sine_tbl 函数可通过每四个值中取一个方法建立此表，并定义为存储器正弦表。同时，该函数还会对存储器正弦表每个字节乘 1/2，并加上前文提到的偏移量。这样，存储器正弦表的值，也就是最后装入 PWM 比较器之值，只有 7bit 的分辨度。其原因由公式（7）可以清楚地看出，原来正弦表中的值用 8bit 描述，而现在经 init_sine_tb1 函数处理后变成 7bit 的原因，是由于乘系数 1/2 的结果。另外，如前文提到的，由于要保证死区时间，还需要设定偏移量，它的大小和死区大小一致。因之，最后生成的存储器正弦表由 256 字节组成，每个字节的取值范围为 10～125（其中 10 是偏移量），其分辨度是 7bit。

产生三相 120°相移正弦波的快速实现方法是：在存储器里建立三个存储器正弦表，每个表之间相差 120°（注意相序的正确性）。这样，同一个角度变量对应不同正弦表的三个不同取值。也就是说，CCU 中的三个比较存储器存有不同的值。

图 5.4－4 给出了利用一个反余弦表和三个正弦表形成三相正弦波的过程。每一次 SPWM 方波参数的形成，在每隔 50 μs（f_{PWM} = 20 kHz）的比较定时器 1 中断服务程序中完成，整个服务程序的运行时间约为 15 μs（占 CPU 工作量的 30%）。

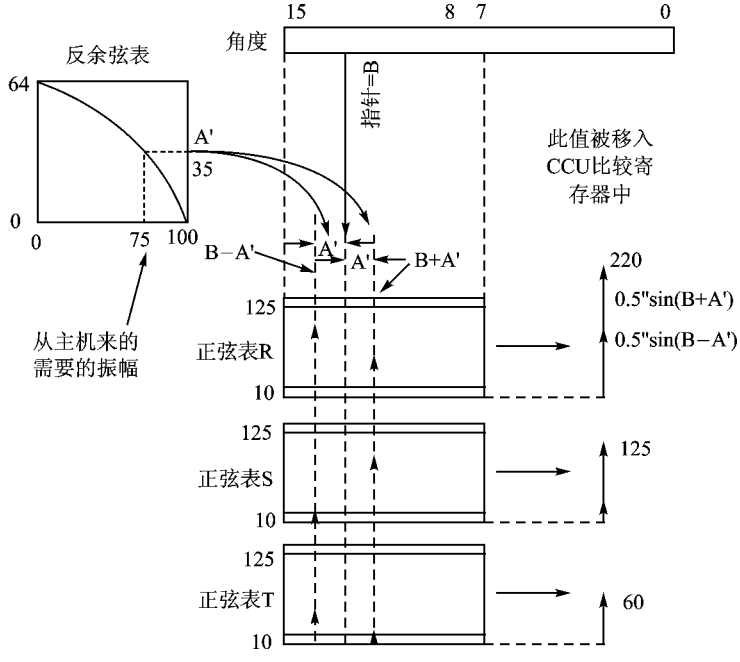


图 5.4－4 三相正弦波的形成

四、结 论

根据我们的体会,由于比较定时器 1 为 16 位计数器,因此只要正弦表设计合理,CPU 和 CCU 内部资源安排合理,可以使正弦波精度很高。我们将这种算法应用于新研制的变频空调电控系统,取得了理想的效果。特别在室外机部分,利用 C504 单片机中的 CCU 单元,不但产生了 0 ~ 150 Hz 的可变频率正弦波,而且方便地实现了 0 ~ 1.27 范围的调制度。波形可靠稳定,运行平稳,噪声低。

参 考 文 献

1 SIMENS 8 BIT MICROCONTROLLERS User s Manual. 11. 97

2 SIMENS C504 Starler Kit Version 1. 1 07. 98

3 三菱变频空调控制器方案一 KL05 简介,三菱电机 (香港) 有限公司等. 1999

4 KitCon-504, Hardware-Manaul, Version 1. 2 6/1997

选自《半导体技术》月刊,2002 年第 1 期

5.5 新型高精度时钟芯片 RTC－4553

宁光电工有限公司 赵四海

现在流行的串行时钟芯片很多，如 DS1302、DS1307、PCF8485 等。这些芯片接口简单、价格低廉、使用方便，被广泛地采用，但这些芯片都存在时钟精度不高，易受环境影响，出现时钟混乱等缺点。本文介绍一种 EPSON 公司最新推出的 RTC－4553 时钟芯片。该芯片采用内置晶振和独特的写数据方法，大大提高了时钟精度和可靠性。RTC－4553 配有串行通信接口，另有 30×4 bit SRAM，有 2000～2099 的百年日历，采用 14 脚 SOP 封装，电池耗电 2 μA，时钟误差 <3 min/年且无需调整，是仪器仪表高精度时钟的理想芯片。

一、内部结构及引脚

串行时钟芯片的内部结构如图 5.5－1 所示。它包含 I/O 控制器、移位控制器、命令及逻辑控制器，静态 RAM、实时时钟、计数器、晶振等部分。

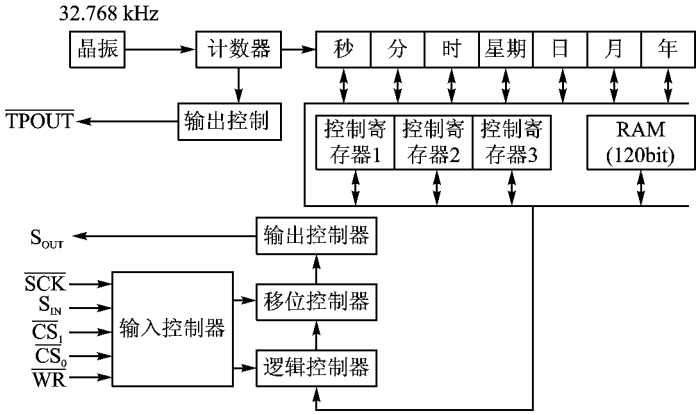


图 5.5－1 RTC－4553 内部结构框图

图 5.5－2 为 RTC－4553 的引脚图。$\overline{CS_0}$为片选脚，低电平选中；$\overline{WR}$为读写使能口，高为读，低为写；L1～L5 为工厂出厂调整精度和测试用，使用中悬空；CS_1 为芯片掉电检查口，可直接与系统电源连接，芯片测到该口为低时，自动进入低功耗状态；$\overline{SCK}$为时钟口，S_{IN}为数据输入口，S_{OUT}为数据输出口。另外，芯片还有 1 个时钟信号输出口 TP_{OUT}，该口可输出 1 024 Hz 或$\frac{1}{10}$ Hz 的信号，以供检测芯片的时钟精度所用。

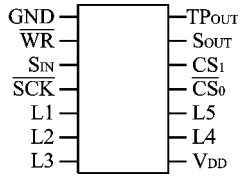


图 5.5－2 引脚图

二、功能及控制

1. 寄存器

RTC－4553 共有 46 ×4 bit 寄存器。这些寄存器分 3 页,第 1 页共 16 个,分别为时钟寄存器和控制寄存器,如表 5. 5－1 所列,用来存放秒、分、时、日、月、年、星期和 3 个特殊寄存器;第 2 页、第 3 页各有 15 个,共 30 个 SRAM 寄存器,页面的选择通过操作控制寄存器 3 的 MS1、MS0 位来实现。

表 5. 5－1

第 0 页		第 1 页		第 2 页	
地址 A3 A2 A1 A0	功能说明	地址 A3 A2 A1 A0	功能说明	地址 A3 A2 A1 A0	功能说明
0	个位秒	0		0	
1	十位秒	1		1	
2	个位分	2		2	
3	十位分	3		3	
4	个位时	4		4	
5	十位时	5		5	
6	星期	6	静	6	静
7	个位天	7	态	7	态
8	十位天	8	RAM	8	RAM
9	个位月	9	区	9	区
0A	十位月	0A		0A	
0B	个位年	0B		0B	
0C	十位年	0C		0C	
0D	控制寄存器 1	0D		0D	
0E	控制寄存器 2	0E		0E	
0F	控制寄存器 3				

控制寄存器 1:CNT1

TPS	—	CNTR	24/12

TPS——TP_{OUT}输出时钟选择位,1 输出 1 024 Hz,0 输出 1/10 Hz;

CNTR——时钟寄存器清零标志;

24/12——1 为 24 小时制,0 为 12 小时制。

控制寄存器 2:

BUSY	PONC	—	—

BUSY——有进位溢出;

PONC——初始上电检测,为 1 表示刚上电需校时。

控制寄存器 3:

—	—	MS1	MS0

MS1、MS0——页面选择位，00 和 01 指向 0 页，10 指向 1 页，11 指向 2 页。

2. 数据读出

在片选选中芯片，$\overline{WR}$置高时，芯片处于读出状态，随着$\overline{SCK}$脚上的时钟变化，内部寄存器的数据将出现在 S_{OUT}脚上。输入需要 8 个时钟，4 个用来输入地址 输出数据也需要 8 个时钟，包括 4 个地址位和 4 个数据位。数据在$\overline{SCK}$上升沿输入，在下降沿输出。寄存器的地址由 S_{IN}脚输入，页面由 MS0、MS1 决定。图 5.5－3 为读时序图。

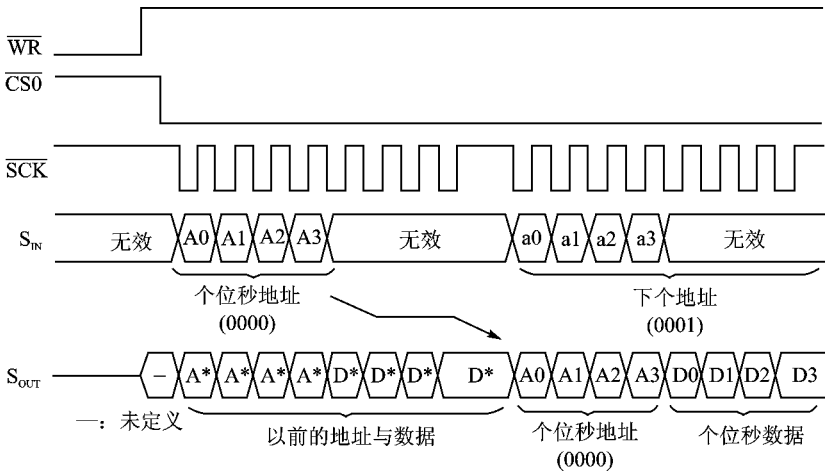


图 5.5－3 读时序

3. 数据写入

RTC－4553 采用特殊的写指令，对第 0 页的 0D～0FH 及第 1 页、第 2 页的寄存器的操作采用常规写法，地址后面的数据将原样写入寄存器中，而对时间寄存器写操作指令只能将内部的内容加 1，并自动完成转换。图 5.5－4 为时间寄存器写时序。芯片这种独特的设计，防止了时钟区数据被意外干扰出现非法数据的可能，这正是该芯片高可靠性的原因所在。

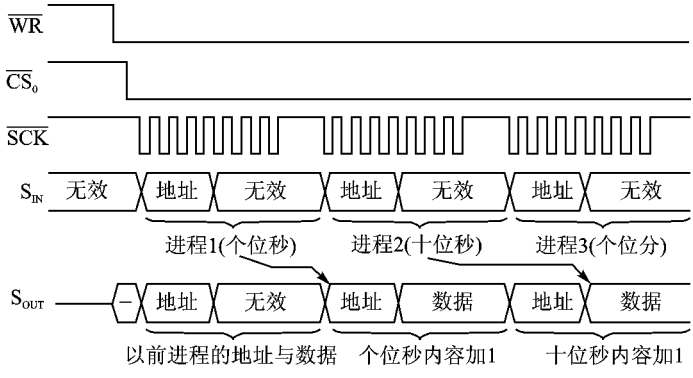


图 5.5－4 时间寄存器写时序

三、应 用

RTC－4553 采用串行通信,与单片机接口简单,在设计中 RAM 区可放置少量的停电后系统需要保存的数据。CS_1 也可与单片机的掉电检测口相连,以便能迅速进入低功耗状态。图 5.5－5 以 PIC 单片机为例,给出连接图。

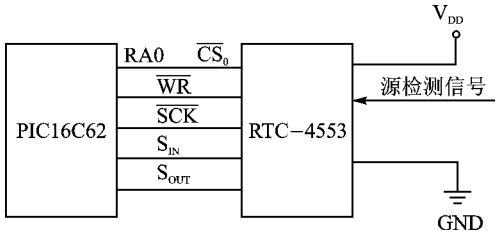


图 5.5－5 电路图

按图 5.5－5 给出单字节的读程序:

入口:FDE 的低 4 位存放读地址,W 的低 4 位存放读地址

```
        BSF RA,WR                ;读状态
        BCF RA,CS0               ;选芯片
        MOVLW 8
        MOVWF Count              ;准备发 8 位
LOOP:   BCF RA,SCK               ;SCK 低电平
        BCF RA,SIN
        BTFSS FDE,0              ;FDE 的 0 位为 1
                                 ;则 SIN 口为 1
        GOTO LLL                 ;否则 SIN 口输出 0
        BSF RA,SIN
LLL:
        RRF FDE,1                ;FDE 右移,准备发下一位
        BSF RA,SCK               ;SCK 高电平
        DECFS2 Count
        GOTO LOOP                ;读指令发完
        MOVLW 8                  ;准备接收数据
        MOVWF Count
LOOP1:
        BCF RA, SCK
        NOP
        BSF RA, SCK
        RRF W,0
        BCF W,0
        BTFSS RA, Sout           ;读判断
        GOTO LLL1
        BSF W,0
```

```
LLL1：
        DECFS2 Count
        GOTO LOOP1
        BCF RA，CS0                    结束，并芯片
```

参 考 文 献

1 窦振中. PIC 系列单片机原理和程序设计. 北京 北京航空航天大学出版社，1998

2 DATA SHEET OF REAL TIME CLOCK MODULE RTC - 4553

选自《单片机与嵌入式系统应用》月刊，2002 年第 11 期

5.6 A/D 芯片 TLC2543 与 Neuron 芯片的接口应用

武汉大学电子信息学院 (430072)
任清珍 杨显娇 黄天成 王志刚

现代工业过程控制对仪器仪表装置在速度、精度、成本等方面有了更高的要求,导致用数字信号传输技术代替传统 4 ~ 20 mA 直流电流信号、1 ~ 5 V 直流电压信号等统一的模拟信号传输技术成为发展趋势。现场总线就是用于现场仪器仪表与控制室之间的一种全数字化、双向、多站点的通信系统。LonWorks 现场总线是美国 Echelon 公司于 1991 年推出的局部操作网络 (local operating networks),也称为 Lon 总线,它适用于现场级的分布式控制系统。LonWorks 技术核心是神经元芯片,它不仅是 Lon 总线的通信处理器,同时也可以作为采集和控制的通用处理器。具有串行接口特性的 A/D 芯片 TLC2543 可以简化电路结构,节约 I/O 资源,用在现场总线式智能仪器仪表、工业测控系统中,很有实用价值。

一、接口硬件实现

1. 神经元芯片的神经元接线 I/O 模式

Lon Works 技术的核心是神经元芯片。神经元芯片主要包含 TMPN3150 和 TMP3120 两大系列。TMPN3150 支持外部存储器,适合更为复杂的应用;而 TMPN3120 则不支持外部存储器,它本身带 ROM。笔者选用 TMPN3150,该芯片内有 3 个微处理器,集网络通信、应用程序处理和多种 I/O 模式于一体。它与其他设备的互联是通过它的 11 个 I/O 口——IO0 ~ IO10。这些管脚可以根据不同的外部设备 I/O 的要求,灵活地配置输入输出方式。神经元芯片的 11 个 I/O 口有 34 种预编程设置模式,支持电平、脉冲、频率、编码等各种信号方式,可以定义为并行 I/O 对象、串行 I/O 对象、直接 I/O 对象、定时/记数器输入对象等。本文仅介绍其中一种 I/O 应用模式,即神经元接线 I/O 模式 (Neuronwire I/O mode)。神经元接线对象与外围设备实行双工同步数据传输,一次最多可以传递 255 位数据,可以工作在主模式和从模式。主模式和从模式的区别就是:在主模式下,Neuron 芯片输出时钟信号;在从模式下,Neuron 芯片接受时钟信号输入。在主模式下,引脚 IO8 是神经元芯片移位时钟信号输出端,引脚 IO9 是串行数据输出端,IO10 是串行数据输入端,引脚 IO0 ~ IO7 可以任选一个作为 A/D 芯片的片选信号线。当神经元芯片输入时钟为 10 MHz 时,输出移位时钟速率可以定为 1 Kb/s、10 Kb/s 或者 20 Kb/s。在从模式下,引脚 IO8 是移位时钟输入端,引脚 IO0 ~ IO7 不再用作片选信号线。

2. A/D 芯片 TLC2543 的基本特性

A/D 芯片 TLC2543 是 12 位开关电容逐次逼近式模数转换器,带有 SPI (serial peripheral interface) 接口。它改变以往许多 A/D 芯片并行输出、连线复杂的缺点,而是在 A/D 转换结果串行输出的同时,可串行输入下次 A/D 转换位控制字。芯片管脚如图 5.6 - 1 所示。每一个器件有 3 个输入控制端,片选 ($\overline{CS}$)、I/O 时钟 (I/O clock) 以及地址输入端 (Data Input)。它还可以通过一个串行的三态输出端与主处理器及其外围的串行口进行通信,输出转换结果。除

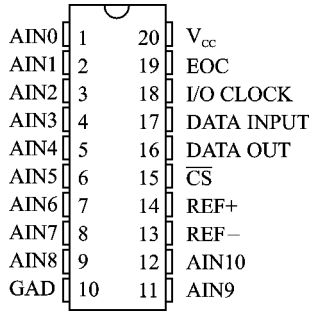


图 5.6－1 芯片管脚

了高速的转换器和通用的控制能力外，片内有 14 通道多路器可以在 11 个模拟输入通道（AIN0 ~ AIN10）或 3 个内部自测电压（self-test）任意选一个。转换结束时，EOC 输出端变高，指示转换完成。该芯片可以用控制字选择输出数据长度、输出数据顺序和输出数据格式（单极性或双极性）。下文详述 A/D 转换位控制字的内容。控制字位长为 8 位，其中 D_0 控制用来表示转换结果的二进制数据格式。D_0 置 0，转换结果为单极性（无符号二进制）数据；D_0 置 1，转换结果为双极性（有符号二进制）数据。D_1 选择转换后的数据是先输出 MSB，还是选输出 LSB。D_1 置 0，先输出 MSB；D_1 置 1，先输出 LSB。D_3D_2 选择数据长度，可以选择 8 位、12 位、16 位。其内部实际的转换结果总为 12 位长，当选择 8 位数据长传送时，内部的 4 个 LSB 被截去以提供一种更快的单字节传送。当选择 12 位数据长传送时，所有的位都被传送。当选择 16 位数据长时，4 个 LSB 填充位被补充到内部转换结果。在 LSB 导前方式下，4 个前导零被输出。在 MSB 导前方式下，最后 4 个输出为零。高四位（$D_7D_6D_5D_4$）选择模拟输入通道地址。通道选择字如表 5.6－1 和表 5.6－2 所列。A/D 芯片和神经元芯片 3150 的接线图如图 5.6－2 所示。输入通道只画了一路（IN1），其余 10 路输入接法一样。当输入为 1 ~ 5 V 电压信号时，跳线 J 断开；当输入为 4 ~ 20 mA 电流信号时，跳线 J 短接。

表 5.6－1 通道选择字

选择模拟输入通道						
通道号	AIN0	AIN1	AIN2	AIN3	AIN4	AIN5
$D_7D_6D_5D_4$	0000	0001	0010	0011	0100	0101
通道号	AIN6	AIN7	AIN8	AIN9	AIN10	
$D_7D_6D_5D_4$	0110	0111	1000	1001	1010	

表 5.6－2 自测电压选择

选择自测电压			
电压值	$(V_{ref+} - V_{ref-})/2$	V_{ref-}	V_{ref+}
$D_7D_6D_5D_4$	1011	1100	1110

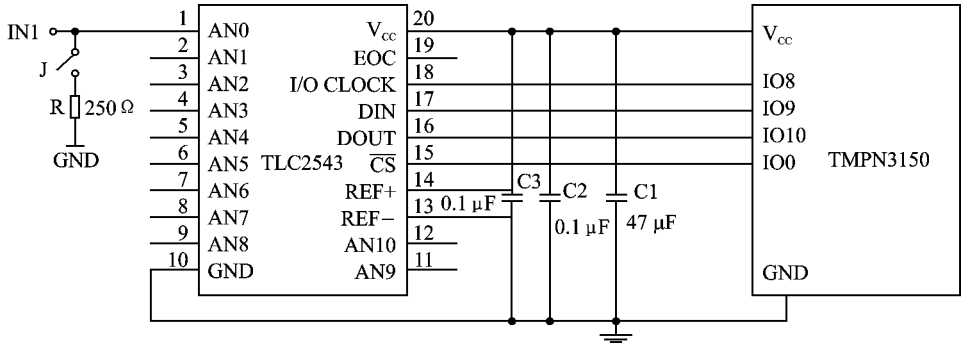


图 5.6－2 芯片 3150 接线图

二、系统软件实现

神经元芯片的编程用 Neuro C 语言，它以 ANSI C 为基础，专为神经元芯片而设计，同时加入通信、时间调度、分布数据对象和 I/O 功能。以下为 A/D 接口程序。

```
mtimer repeating t0;      //定义毫秒定时器
//////////////////////networks varials//////////////////////////
//定义标准的温度网络变量
network output SNVT_temp Result[11];
//////////////////////I/O Objects//////////////////////////////
//定义为 neurowire 主模式,A/D 片选信号由 IO0 输出,ioA2D 为用户自定义 IO 对象名称
IO_8 neurowire master select(IO_0) ioA2D;
//IO0 输出 A/D 片选信号,ioA2DSelect 为用户自定义的 IO 对象名称,初始化为 1
IO_0 output bit ioA2DSelect = 1;
#pragma ignore_notused ioA2DSelect
//////////////////////////function declare//////////////////////////
unsigned long analog_to_digital(unsigned long analog_addr);
//////////////////////////function definition//////////////////////
//读取选定通道(由 analog_addr 控制字决定选定的通道)的 A/D 转换结果值的子函数
unsigned long analog_to_digital (unsigned long analog_addr)
 {
   unsigned long adc_info;unsigned long digital_out;
   adc_info = (analog_addr<<8);
   io_in (ioA2D,&adc_info,12);
   digital_out = ((adc_info>>4) &0xff0) | (adc_info&0x00f);
   feturn digital_out;
}
////////////////////////////////////task/////////////////////////////////////////
//每隔 4 ms(即 T0 定时到)采集 4 路模拟通道的值
when (timer_expires (t0))
```

```
{
  unsigned long addr;
  unsigned long deduct;
  int i;
  addr = 0x00;
  deduct = analog_to_digital (addr);
  Result [10] = analog_to_digital (addr) * 10 + 2740;
  for (i = 1; i < 11; i + +)
  {
    addr = addr + 1;
    deduct = analog_to_digital (addr);
    Result [i - 1] = analog_to_digital (addr) * 10 + 2740;
  }
}
//系统复位时设置 T0 的定时时间为 4 ms
when (reset)
{
  t0 = 4;
}
```

在读取选定通道的 A/D 转换结果值的子函数中，io_in (ioA2D,&adc_info,12)，不仅向 A/D 芯片传递 adc_info 的值，即控制字的值，决定了选取通道号和其他信息，而且同时将 A/D 上次转换结果存放入 adc_info 所指向的存储单元，也即 adc_info 同时表示一个地址信息。实现了 A/D 转换结果串行输出的同时，可以串行输入下次 A/D 转换位控制字。Neuron C 的任务调度是事件驱动 (event driven) 的，当一个给定事件发生的条件为真时，与该事件关联的一段代码 (成为任务) 被执行，调度程序允许编程者定义事件，如输入管脚状态的改变、计时器的溢出等。事件通过 when 子句来定义。本程序通过判断定时器 T0 的溢出事件来执行数据采集任务。

参 考 文 献

1 Neuron C Reference Guide [Z], Echelon Company, 1999

2 Neuron Chip Data Book [Z], Echelon Company, 1999

3 阳宪惠，魏庆福等. 现场总线技术及其应用 [M]. 北京：清华大学出版社，1999

选自《测控技术》月刊，2002 年第 9 期

5.7 一种新型传感器接口 IC

石家庄中国人民解放军军械工程学院 (050003) 贾英江
石家庄市养路费稽征处 (050001) 贾向英

一、引 言

传统传感器对于敏感元件的校准和补偿是在模拟域,采用“模拟记忆”元件,如电位器、电容器和激光微调薄膜电阻等进行。这样的传感器常常使用热敏电阻、二极管或其他模拟技术来进行温度补偿。尽管不是十分合适,二极管间断点还是常被用来增强线性。所有这些方法都存在以下几个主要缺点:

① 补偿精度受限于元件的非线性误差;

② 补偿器件同样受温度漂移影响;

③ 激光微调和其他自动调整设备价格昂贵;

④ 手动校准(“旋钮调节”)导致高成本。

可编程数字器件的出现使得采用数字方式调整模拟系统成为可能,并可将独立的校正系数保存在不挥发数据存储器内(如 EEPROM)。对于传感器,这种电子调整发展为两种技术。

(1) 数字传感信号处理器 (DSSP)

DSSP 技术包括利用 ADC 对传感器信号的数字化转换,利用带有 EEPROM 的控制器在数字域进行校准和补偿,以及利用 DAC (如有必要) 将补偿结果变回到模拟信号。这种方法的优势在于 ADC 数字化后的处理是由处理器在数字域完成的,具有零漂移的特性。缺点在于软件的复杂性,要求一定的内存,以及提高 ADC 分辨率会降低动态范围等。

(2) 模拟传感信号处理器 (ASSP)

ASSP 通过调整传感器的激励以及数字调整放大器的偏移和增益,实现了模拟域对传感器进行校准和温度补偿,而不需要信号的量化。利用 DAC、EEPROM 和可数字调节的模拟器件,这种混合技术具有全模拟和全数字两种方法的优点,即在模拟域处理信号而用数字系统代替电位器的功能。为改善传感器的线性,ASSP 系统根据由原始传感器输出到 DAC 基准输入的反馈信号调节增益和偏置,这种技术省去了 DSSP 中采用的复杂的多项式曲线拟合。利用 DAC 将数字量与模拟电压 (DAC 基准输入) 相乘,这是 ASSP 电子调节系统的关键。本文简要介绍一种传感器接口 IC。

二、IC 简介

如图 5.7-1 所示,MAX1457 是一个高精度、混合信号、线性化前端器件。它包括一个 12 位 ADC,用来将传感器输出的温度量数字化并转换为地址信息提供片外的线性化 EEPROM。该 EEPROM 内部保存了 120 段曲线的数据,用来对失调和增益进行校正和线性化处理。

尽管没有内部 EEPROM,MAX1457 可直接寻址标准 EEPROM,例如 93C66。模拟信号通

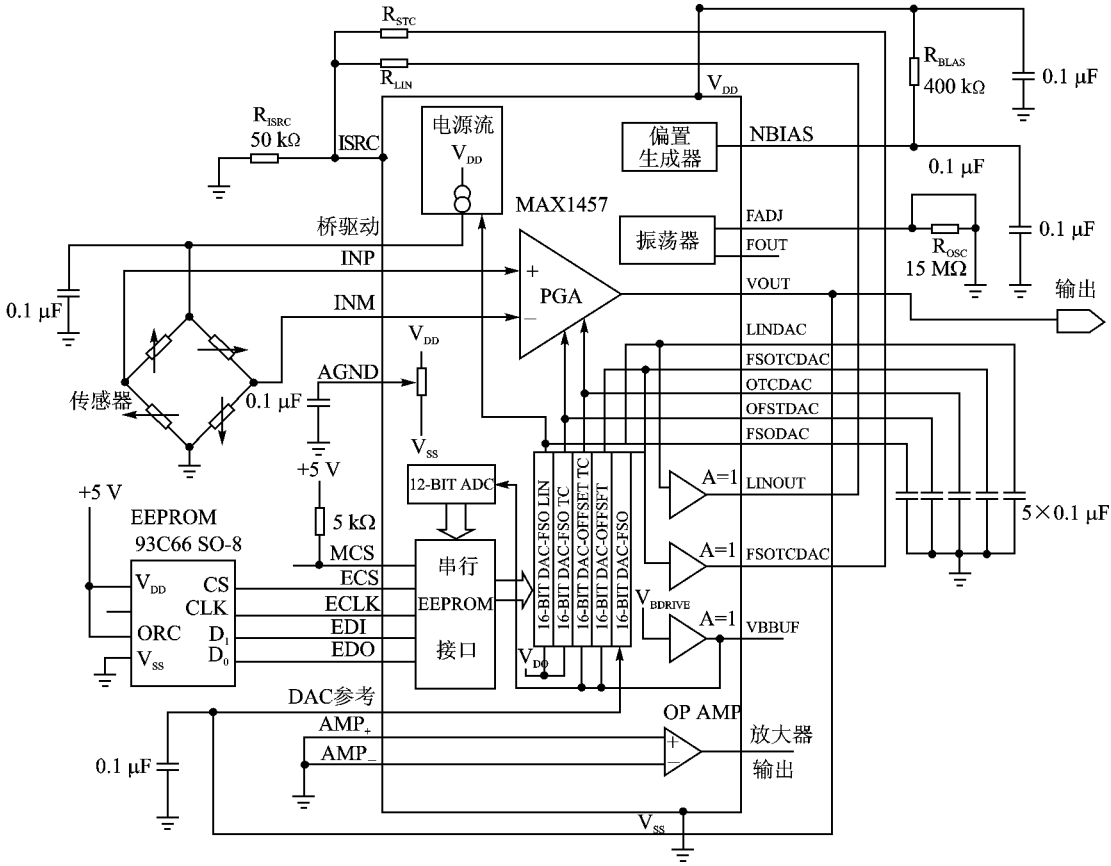


图 5.7－1 MAX1457 传感器接口 IC 及其测量配置

路包括 1 个独立的运算放大器,5 个 16 位增益和失调控制 DAC 和 1 个 12 位 ADC。MAX1457 采用的模拟域校准结构比大多数模拟输出的传感器更为简单。模拟输出可适当偏置以提供 4～20 mA 信号,或者也可将其直接送到系统 ADC,与 PRT 压力传感器配合,MAX1457 能够提供的典型校准精度为 0.1%。

三、补偿原理

MAX1457 的补偿原理如图 5.7－2 所示。它采用二种补偿方法。第 1 种是模拟方式,采用 2 个 DAC 补偿一阶温度误差:用 1 个失调温度补偿 DAC 调节输出失调;另一个满度输出温度补偿 DAC 调节激励电流来改变激励电压。第 2 种方式为数字补偿,一个由桥路激励电压(表示温度的信号)驱动的 ADC 产生 EEPROM 地址,EEPROM 输出采用多段逼近(高达 120 段)的方法校正残余的高价误差。

初始失调的校准是通过将电源电压的一部分与 1 个 16 位字相乘(由失调 DAC 完成)所得电压送到 PGA 的求和端来完成的。满度输出(FSO 或增益)的校准包括二个调节:增益粗调由送到 PGA 的三位数据设定;细调是通过另一个 16 位字调节桥路电流来实现的。

连接到桥路电压端的 2 个 DAC(失调温补 DAC 和满度温补 DAC)对零点和满度系数的线性分量进行补偿。正比于温度的桥路电压和 1 个适当选定的数据(乘数)使 DAC 输出跟随桥

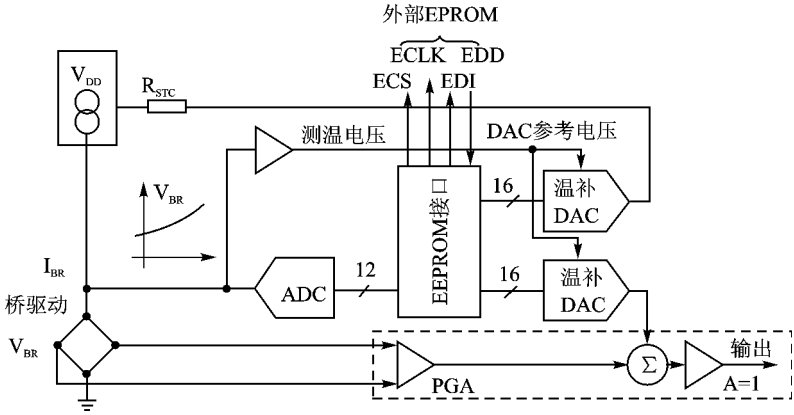


图 5.7－2 MAX1457 的补偿原理

路电压的准线性变化，实现对温度斜率的补偿。

四、补偿效果

图 5.7－3 描述了 MAX1457 对温度和线性误差的补偿效果。图 5.7－3 (a) 表示 1 个未经补偿的压阻式压力传感器的低电平输出；图 5.7－3 (b) 为未补偿的失调和增益的温度漂移；图 5.7－3 (c) 和图 5.7－3 (d) 表示经过补偿的信号。MAX1457 将传感器输出范围调整在 0.5～4.5 V (见图 5.7－3 (c))，并且在很宽的温度范围内将增益和失调误差限制在 0.1% 以内 (图 5.7－3 (d))。

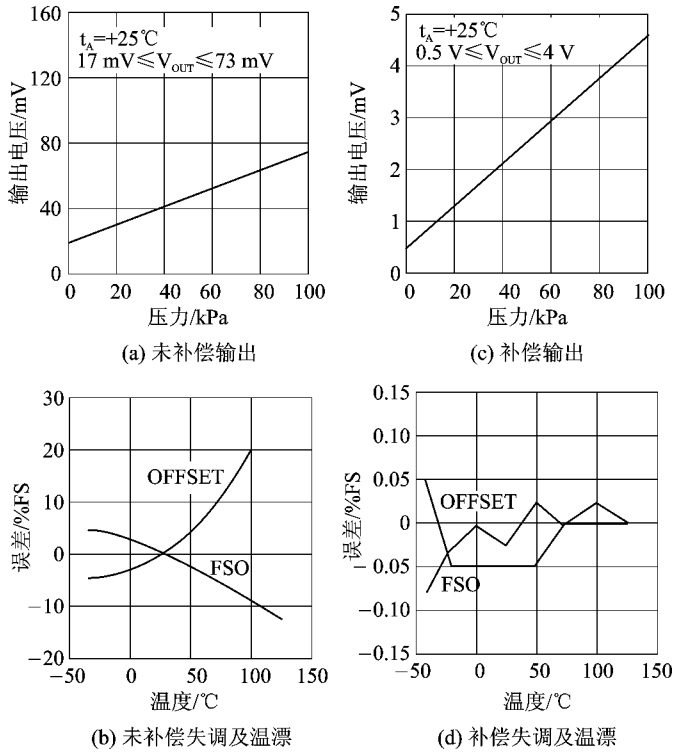


图 5.7－3 MAX1457 补偿效果

本文介绍了传感器接口 IC 的基本情况，随着传感器技术的不断发展，更高级的传感器接口 IC 必然会不断涌现。

参 考 文 献

1 刘广玉，等. 新型传感器技术及应用. 北京：北京航空航天大学出版社，1995
2 何圣静，陈彪. 新型传感器. 北京：兵器工业出版社，1993
3 赵负图. 国内外最新常用传感器和敏感元器件性能数据手册. 沈阳：辽宁科学技术出版社，1994

选自《自动化仪表》月刊，2002 年第 6 期

5.8 新型 CMOS 图像传感器及其应用

南京大学电子科学与工程系 (210093)
李 冬 徐家恺 孔令红

目前,固体图像传感器分为两大类,即 CCD 器件和 CMOS 器件。CCD 是传统的摄像器件,近十几年已经发展得非常成熟,具有灵敏度高、噪声小、图像质量好等优点,已经逐步发展成为图像传感器的主导产品。但生产 CCD 器件需要采用专门工艺,与一般集成电路工艺不兼容,比较复杂,成本也高,并且 CCD 传感元件和信号处理电路不能集成在同一芯片上,造成了由 CCD 器件制成的图像采集系统体积大、功耗大的局限。CMOS 图像传感器并不是新产品,20 世纪 70 年代初,国外就开发出早期的 CMOS 图像传感器,但在分辨率、动态范围、噪声、功耗和成像质量等方面都不如当时的 CCD 图像传感器,因而未获得充分发展。但是随着 CMOS 工艺技术的发展,采用标准 CMOS 工艺生产的高质量、低成本的 CMOS 图像传感器已显示出强劲的发展势头。CMOS 图像传感器的高度集成化减小了系统的复杂性,可以根据需要将多种功能集成在一块芯片上,单芯片就可以完成 CCD 摄像机全部电学功能,降低了制造成本,对获得的图像信息读出及处理变得简单而快捷,能设计出更灵巧的小型成像系统。它还具有单一工作电压、功耗低 (仅为普通 CCD 图像传感器的十分之一)、可与其他的 CMOS 电路兼容、对局部像素图像的编程随机访问等优点。表 5.8-1 列举了 CMOS 图像传感器优于 CCD 器件的一些主要特点。

表 5.8-1 CMOS 器件和 CCD 器件的比较

项 目	CMOS 图像传感器	CCD 图像传感器
模拟/数字转换	片内含有	片外另配
时序及控制电路	片内含有	片外另配
自动增益控制	片内含有	片外另配
信息处理	片内含有	片外另配
彩色编码	片内含有	片外另配
系统功耗	<150 mW	>1.5 W
模块体积	14 mm×14 mm×7 mm	30 mm×30 mm×15 mm
集成情况	单片高度集成	多片、多元件

一、CMOS 图像传感器的结构与工作方式

ARAMIS EVS100K 是 Elec Vision 公司研究生产的一款彩色 CMOS 图像传感器,内部结构和工作方式既有自己的特色,又具有一定的代表性。下面以 ARAMIS EVS100K 为例,剖析一下 CMOS 图像传感器的大体结构和工作方式。

1. 结 构

ARAMIS EVS100K 属于有源像素传感器 (Active Pixel Sensor),像素点阵为 352 × 290 (10 万像素),每个像素大小仅为 12.1 μm × 12.1 μm,并且每个像素都有一个模拟帧缓冲结构,用于存储感光得到的一帧模拟信号。该传感器采用高度集成的 CMOS 工艺,将光敏单元阵列 (像素点阵)、控制与驱动电路、模拟信号处理电路、8 位 A/D (模-数转换) 电路高度集成在一块芯片上,封装形式为 LCC—48,大小仅约为 11.2 mm × 11.2 mm。ARAMIS EVS100K 的功耗很小,一般小于 200 mW,外围电路也非常简单,不需要外接晶振,仅需外接阻容电路和一些门电路。这极大地方便了它与各种控制芯片,如 DSP、PC 机和单片机的连接。ARAMIS EVS100K 的这些结构特色集中体现了 CMOS 图像传感器单芯片、体积小、功耗低等优点。图 5.8-1 给出了它的内部功能模块。

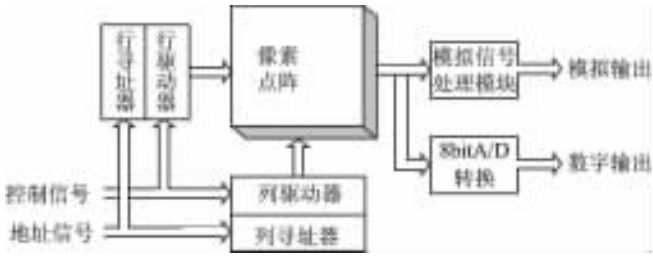


图 5.8-1 ARAMIS EVS100K 功能模块

2. 工作方式

ARAMIS EVS100K 基本操作如下。

(1) 行寻址和列寻址

像素点阵的行寻址和列寻址都是通过 9 位地址线实现的,由外接信号 XYSEL 控制寻址对象。当 XYSEL 为高电平时,地址线上的数据为行地址,被送入行地址锁存器;当 XYSEL 为低电平时,地址线上的数据为列地址,被送入列地址锁存器。

(2) 光敏单元复位

在每次图像采集前,光敏单元阵列都需要进行复位操作。复位操作有两种模式:帧模式和行模式。当运用帧模式复位时,整个光敏单元阵列都被复位到外接的复位电压 V_{rst};当运用行模式复位时,只有当 XYSEL 为高电平时被选中的行复位到 V_{rst}。

(3) 曝 光

像素点阵感光获得图像信息的过程称为曝光期,持续时间由外界的亮度决定。曝光操作只有一种模式——帧模式。

(4) 8 bit A/D 转换

ARAMIS EVS100K 芯片内部集成的 8 位 ADC (模-数转换器) 将感光的模拟图像信号转换为数字信号,便于进一步处理。模拟信号在时钟信号的下降沿被采样,在时钟信号的低电平期间进行 A/D 转换,数字信号在时钟信号的上升沿后延迟半个时钟低电平期间输出。图 5.8-2 是片内 ADC 的工作时序。

(5) 像素采样

采样操作是将光敏单元感光得到的模拟电压传送到每个像素对应的帧缓冲结构中。和复位操作一样,采样操作也有两种模式:帧模式和行模式。采用帧模式将整个像素点阵感光得到

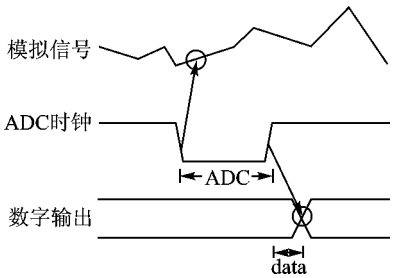


图 5.8－2 ADC 工作时序

的一帧模拟信号存入帧缓冲，行模式对 XYSEL 为高电平时被选中的行执行相应的操作。

⑹ 像素点阵读出

像素点阵只能采用行模式读出，XYSEL 先为高电平来选定行地址，再变为低电平来选定这一行中的列地址，逐点读出这一行的各个像素。这种读出方式保证了可以对任意局部像素的图像进行随机访问，体现了 CMOS 图像传感器的一大优点——对局部像素图像的编程随机访问能力。

二、应用实例

CMOS 图像传感器的正常工作，正是基于以上基本操作的正确组合。下面结合具体应用——一个基于 PC 的图像采集系统来介绍 CMOS 图像传感器的工作流程。

本系统以 CMOS 传感器 ARAMIS EVS100K 为核心图像采集器件，Microchip 公司的单片机 PIC16C57 为主控芯片，通过 EPP 口与 PC 相连进行图像采集和显示，硬件框图如图5.8－3所示。

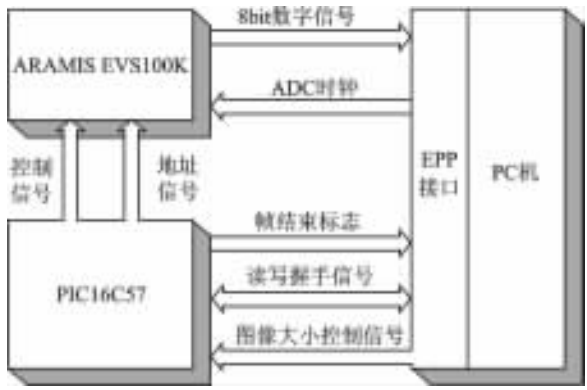


图 5.8－3 图像采集系统硬件框图

主控芯片 PIC16C57 主要有两个作用：一是利用部分输入输出端口作为 ARAMIS EVS100K 的控制信号，用来使 ARAMIS EVS100K 执行各个基本操作；二是通过部分输入输出端口与 PC 的 EPP 口通信，实现数字图像信号的读取。与此相对应，PIC16C57 内的固件，就是将前面所述的 ARAMIS EVS100K 的一系列基本操作按照正确的时序组合，使 ARAMIS EVS100K 正常工作，然后与 PC 配合，使 PC 正确读入图像信号。

本系统中 ARAMIS EVS100K 的一系列基本操作都是通过帧模式来实现的，按照像素复位→曝光→像素采样→按行逐点寻址，进行 A/D 转换，输出数字信号的时序正常工作。这样的工作时序具有一定的通用性，在帧模式下，大多 CMOS 图像传感器都是按照这样的时序进行图像采集的。主控芯片 PIC16C57 内固件关于 ARAMIS EVS100K 控制部分的流程如图 5.8－4 所示。

另外，PIC16C57 还要和 PC 的 EPP 口通信，使 PC 通过 EPP 口读入 CMOS 图像传感器输出

的数字图像信号。

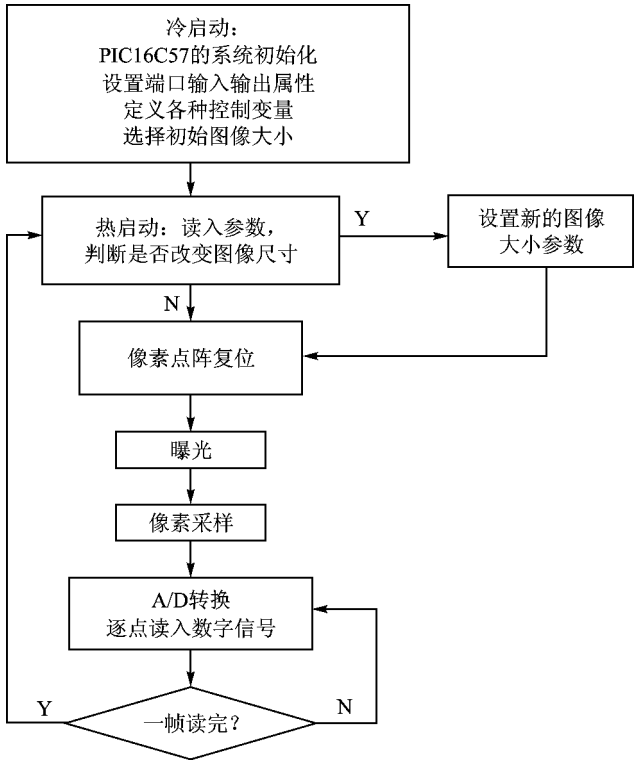


图 5.8－4 ARAMIS EVS100K 操作流程图

EPP 口读操作的步骤如下：

⑴ 外设的 nWait 为低电平；

⑵ 主机将 nWrite 置高，禁止输出 D0～D7，并将 nDstrb 置低；

⑶ 外设启用 D0～D7 输出，写入数据，将 nWait 置高，表示已经准备好了待读数据；

⑷ 主机在 EPP 数据寄存器中读取 D0～D7，并将 nDstrb 置高；

⑸ 外设禁止输出 D0～D7，并将 nWait 置低，整个读周期结束。

EPP 口读操作时序、PIC16C57 固件 EPP 通信部分以及 PC 读取图像信号部分的流程如图 5.8－5、图 5.8－6、图 5.8－7 所示。

本系统 PC 读入图像信号后采用 Direct Draw 显示，在此不再赘述。

目前，CMOS 图像传感器已经步入稳步发展的阶段，随着技术的不断更新，CMOS 图像传感器正在逐渐克服以往光照灵敏度、信噪比和解像度等方面的局限，开始同 CCD 分庭抗争，争夺市场。现在的 CMOS 器件在解析度和亮度方面已同 CCD 旗鼓相当，目前面临的最大问题是如何把信噪比再提高。由于 CMOS 的高度集成和节能特性，迎合了目前业界的 SOC (System on chip) 潮流，可以制造出微型化成像产品，开拓了许多新的应用领域，比如嵌入移动电话、手持电脑和 PDA 的数字摄像机，以及保安监视场合的隐形摄像机，其中 PC 摄像机是 CMOS 器件的应用主流 (上述图像采集系统也是一个简单的 PC 摄像头)。预计在未来的 5 年中，CMOS 图像传感器将取代 CCD 器件成为视频通信、保安监控等领域的主流产品。

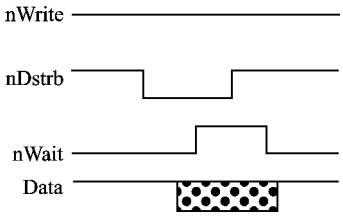


图 5.8－5 EPP 口读操作时序

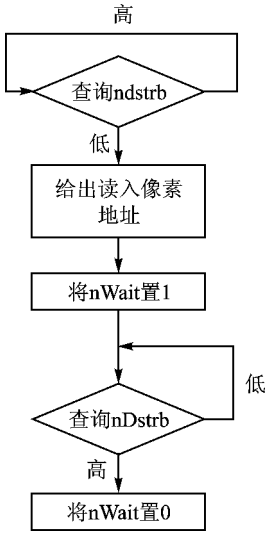


图 5.8－6 EPP 口读操作流程

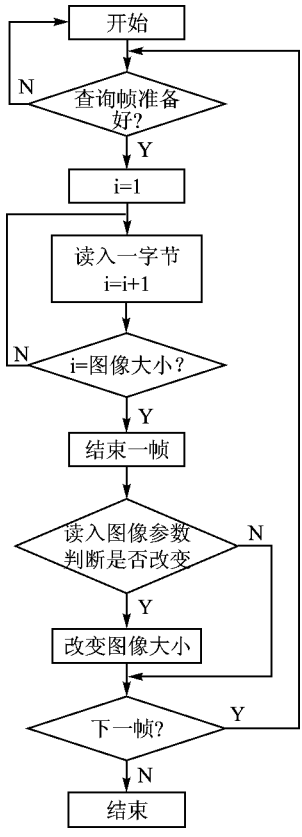


图 5.8－7 PC 读取图像信号流程

参 考 文 献

1 Elec Vision Inc. Data Sheet of EVS100K [EB/OL]. http ∶//www. elecvision. com,2000 - 05 - 01

2 Microchip Technology Inc. PIC16C5X EPROM/ROM-Based 8-bit CMOS Microcontroller Series [EB/OL]. http ∶//www. microchip. com,2000

3 韩文彬,袁潮译. 并行端口编程 [M]. 北京 ∶中国电力出版社,2000

选自《电子元件与材料》月刊,2002 年第 10 期

5.9 GMS97C2051 与 ISD2560 组成的小型语音系统

四川师范大学 梁文海

目前电脑语音服务行业越来越广,如电脑语音钟、语音型数字万用表、手机话费查询系统、排队机以及公共汽车报站器等等。笔者用单片机 GMS97C2051 和 ISD2560 设计了一款微电脑语音板,实现了语音的分段录取、组合回放,通过软件的修改还可以实现整段录取,循环播放。该系统完成语音录放功能,可作为语音服务系统的子系统,而且不必使用专门的 ISD 语音开发设备。

一、系统简介与接口电路

GMS97C2051 是 LG 半导体公司生产的一种功能强大的微控制器,为很多嵌入式控制应用系统提供了一个高度灵活有效的解决方案。GMS97C2051 带有 2K 字节可编程的 EEPROM、128 字节 RAM、15 根 I/O 线、2 个 16 位定时/计数器、1 个全双向的串口、1 个精密比较器等等。其与工业标准 MCS - 51 的指令集和引脚兼容。引脚排列如图 5.9 - 1 (a) 所示。

P1 口是一个双向 I/O 口,其中 P1.2 ~ P1.7 口内部提供了上拉电阻,P1.0、P1.1 需外部上拉。P1.0、P1.1 同时也是片内精密比较器的正输入端 (AIN0) 和负输入端 (AIN1)。P3 口是 7 个带有内部上拉电阻的双向口 (P3.6 除外,其为片内比较器的输出脚,而不能作为普通的 I/O 口使用)。GMS97C2051 具体性能请见参考文献 [1]。

ISD2560 是 ISD 系列单片语音录放集成电路的一种,是一种永久记忆型录放语音电路,录音时间为 60 s,能重复录放达 10 万次。它采用直接电平存储技术,省去了 A/D、D/A 转换器。ISD2560 集成度较高,内部包括前置放大器、内部时钟、定时器、采样时钟、滤波器、自动增益控制、逻辑控制、模拟收发器、解码器和 480 K 字节的 EEPROM 等等。内部 EEPROM 存储单元,均匀分为 600 行,具有 600 个地址单元,每个地址单元指向其中一行,每一个地址单元的地址分辨率为 100 ms。ISD2560 控制电平与 TTL 电平兼容,接口简单,使用方便。引脚排列如图 5.9 - 1 (b) 所示。

· A0 ~ A9 为地址线,共有 1 024 种组合状态。最前面的 600 个状态作内部存储器的寻址用,最后 256 个状态作为操作模式,具体使用见参考文献 [2]。本系统采用对地址直接进行操作的方式。

· 微处理器接口端:P/R 录放音控制端,此端为高电平时为放音状态,为低电平时为录音状态;$\overline{CE}$端用于录放音时的启停控制,通常与 P/R 端配合使用;$\overline{EOM}$端为每段信息结束信号输出端,$\overline{EOM}$为负向信号,时间为 12.5 ms,$\overline{EOM}$上升沿标志信息结束。

· MIC IN 是话筒前置放大器输入端;MIC REF 为话筒补偿端,与麦克风连接电路如图 5.9 - 2 所示;AGC 自动增益控制端;ANA IN 与 ANA OUT 是模拟信号的输入端和输出端,它们之间连接耦合电容,通常取值为 0.22 ~ 1 μF。

ISD2560 与单片机 GMS97C2051 的接口电路以及外围电路如图 5.9 - 2 所示。单片机的

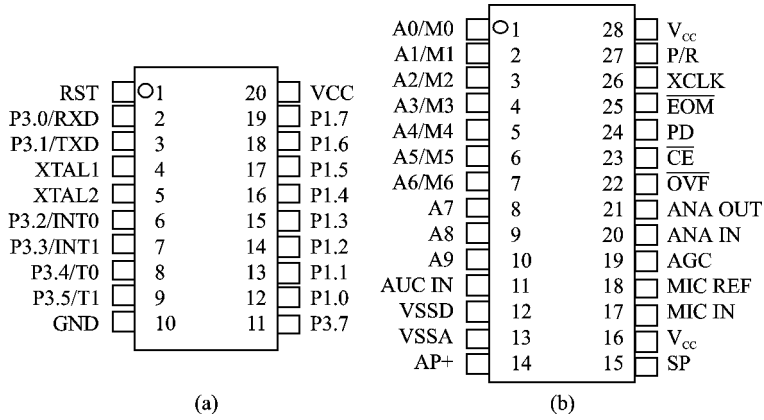


图 5.9－1 GMS97C2051 和 ISD2560 引脚

P1 口、P3.4 和 P3.5 与 ISD2560 的地址线相连，用以设置语音段的起始地址。P3.0 ~ P3.3 用以控制录放音状态。P3.7 扩展一录音键，供录音时使用。ISD2560 具体性能和使用方法见参考文献 [2]。

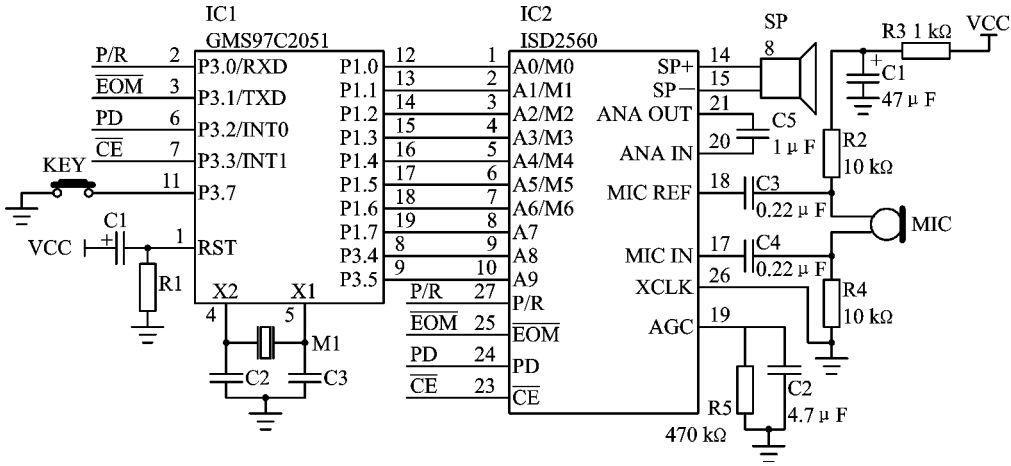


图 5.9－2 ISD2560 与单片机 GMS97C2051 的接口电路

二、系统工作原理及程序设计

1. ISD2560 内部地址单元寻址

ISD2560 虽然提供了地址输入线，但它的内部信息段的地址却无法读出。通常使用不需要知道地址的操作模式，但这不能满足实际的不同需要。一般使用对地址进行直接操作，而要读出 ISD2560 内部信息地址需专用的 ISD 开发设备，其价格较昂贵。本系统采用单片机来控制，不需读出信息地址，而直接设置信息段起始地址。其实现方式有多种，一种方式为：由于 ISD2560 的地址分辨率为 100 ms，所以可用单片机内部定时器定时 100 ms，然后再利用一计数器对单片机定时次数进行计数，则计数器的计数值为语音段所占用的地址单元。该方式能充分利用 ISD2560 内部的 EEPROM，在字段较多时可利用该方法。该方法的具体使用请见参考文献 [4]。语音字段如果较少，则可用下面的方式：根据每一字段的内容多少，直接分配地址单元。一般按每 1 s 说 3 个字计算，60 s 可说 180 个字，再根据 ISD2560 的地址分辨率为 100

ms，即可计算出语音段所需的地址单元数。本系统即采用该方式。

2. 录放音时 GMS97C2051 单片机对 ISD2560 的控制

录音时，按下录音键，单片机通过口线设置语音段的起始地址，再使 PD 端、P/R 端和$\overline{CE}$端为低电平启动录音 结束时，松开按键，单片机又让$\overline{CE}$端回到高电平，即完成一段语音的录制。同样的方法可录取第二段、第三段等等。值得注意的是，录音时间不能超过预先设定的每段语音的时间。

放音时，根据需播放的语音内容，找到相应的语音段起始地址，并通过口线送出。再将 P/R 端设为高电平，PD 端设为低电平，并让$\overline{CE}$端产生一负脉冲启动放音，这时单片机只需等待 ISD2560 的信息结束信号，即$\overline{EOM}$的产生。$\overline{EOM}$信号为一负脉冲，在负脉冲的上升沿，该段语音才播放结束，所以单片机必须要检测到$\overline{EOM}$的上升沿才能播放第二段，否则播放的语音就不连续，而且会产生啪啪声，这一点在编制软件时一定要注意。录放音程序框图如图 5.9－3、图 5.9－4 所示。

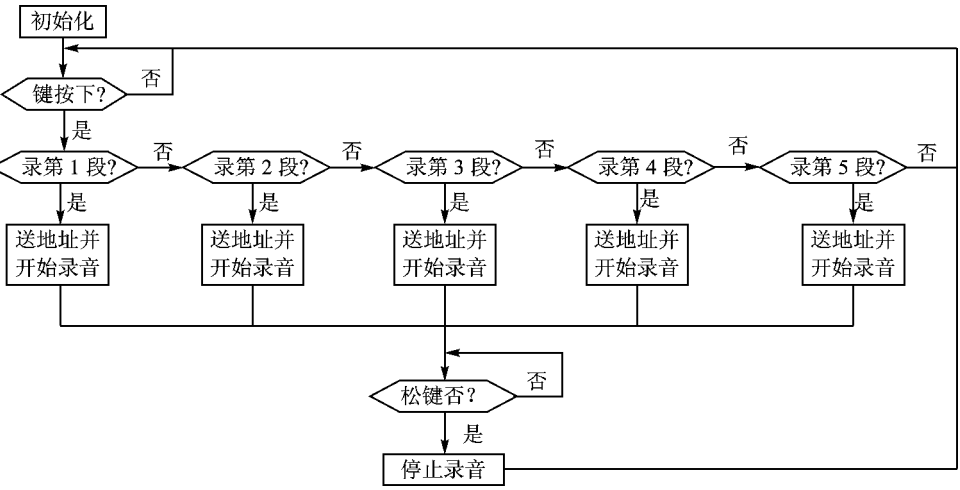


图 5.9－3　录音软件程序框图

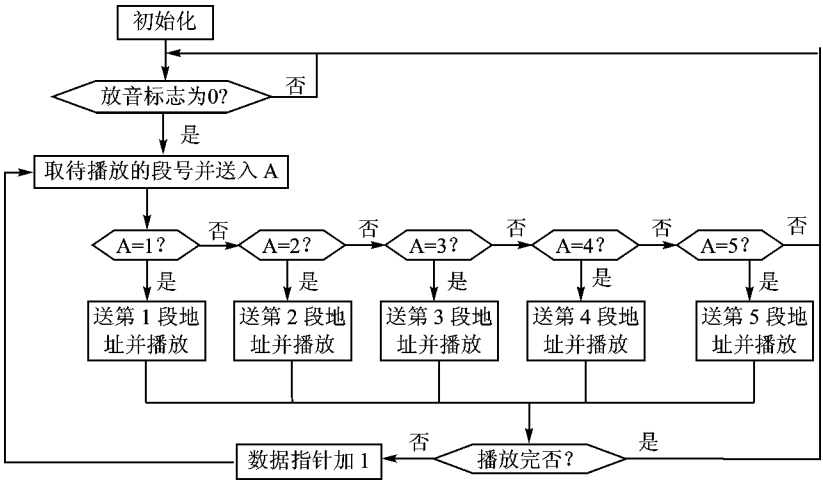


图 5.9－4　放音软件程序框图

3. 系统接口源程序

下面编制了录取 5 段语音信息的录音程序和对这 5 段语音进行组合播放的源程序。播放时,可根据实际情况组合回放。实际需要时,可对该程序进行扩充。录放音源程序见本刊网站:www. dpj. com. cn

参 考 文 献

1 武汉力源电子股份有限公司. GMS97C2051 数据手册. 1997
2 任致远. 语音录放和识别集成电路应用与制作实例. 北京:人民邮电出版社,1999
3 张友德,等. 单片微型机原理、应用与实验. 修订版. 上海:复旦大学出版社
4 刘欣,等. IDS 语音器件分段地址的获取. 电子技术应用,1999 (10)

选自《单片机与嵌入式系统应用》月刊,2002 年第 3 期

5.10　73M2901 芯片在嵌入式 Modem 中的应用

四川大学电气信息学院 (610065)　袁笑鹏　田远富

随着科学技术的发展,人们对通信设备的技术要求越来越高,尤其是在一些特殊场合,往往需要设计成体积小、重量轻、可以直接插入终端的嵌入式结构。以调制解调器 (Modem) 为例,目前市场上的商用 Modem 多为通用类型,体积大、价格高、灵活性差,难以适应特殊需要,因此需要提出一种专用 Modem 的嵌入式设计方案。

一、芯片简介

73M2901 是 TDK 公司最新的低速 Modem 芯片。它是一个真正的单芯片 Modem IC,内置标准的 8032 微处理器来处理复杂的 Modem 信号,同时能够完成多种控制功能。它还包含了 RAM、ROM 和一些 A/D、D/A 转换器,数据端采用异步串行传输方式。它支持 AT 指令集以及 CCITT V. 22bis、V. 22、V. 23 等多种数据通信协议标准,具有功耗低、稳定性好等特点。该芯片集成度高,体积小巧,非常适合用于嵌入式 Modem 中。芯片封装形式如图 5.10－1 所示,主要引脚功能如表 5.10－1 所列。

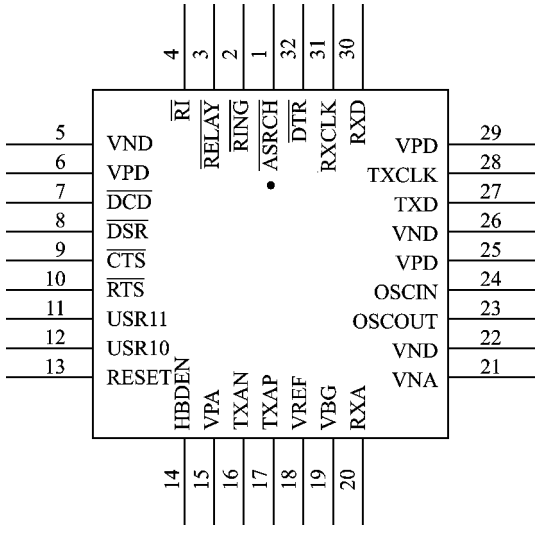


图 5.10－1　73M2901 引脚图

表 5.10－1 主要引脚功能

引脚	名称	功 能
20	RXA	模拟信号接收
16	TXAN	模拟信号发送(负)
17	TXAP	模拟信号发送(正)
2	$\overline{\text{RING}}$	振铃检测输入
1	$\overline{\text{ASRCH}}$	波特率自动检测
32	$\overline{\text{DTR}}$	数据终端准备好
31	RXCLK	接收数据同步时钟
30	RXD	串行数据输出至 DTE
28	TXCLK	输出数据同步时钟
27	TXD	串行数据输入自 DTE
10	$\overline{\text{RTS}}$	请求发送
9	$\overline{\text{CTS}}$	清除发送
8	$\overline{\text{DSR}}$	数据设备准备好
7	$\overline{\text{DCD}}$	载波检测
4	$\overline{\text{RI}}$	振铃指示
3	$\overline{\text{RELAY}}$	继电器驱动输出

二、Modem 硬件设计的总体结构

Modem 的硬件设计总体结构由三大部分构成:数据终端接口、调制解调部件和模拟终端接口。如图 5.10－2 所示。

1. 调制解调部分

调制解调部分的核心是调制解调芯片。Modem 的绝大部分功能都是由这片大规模集成电路完成的,如调制解调过程、扰码解扰码过程、信道分割、线路均衡、指示工作状态等。

2. 数据终端接口电路

数据终端接口电路的主要功能,是完成数据终端(DTE)与调制解调器之间的连接。

73M2901 芯片提供的串行数据终端接口包括 TXD、RXD、$\overline{\text{RTS}}$、$\overline{\text{CTS}}$、$\overline{\text{DSR}}$、$\overline{\text{DCD}}$、TXCLK、RXCLK 等。

Modem 芯片的控制器采用 Atmel 公司的 AT89C51 芯片,它的 P1 口与 73M2901 串行数据接口中各个控制引脚相连,向它发出各种控制信号。单片机的 P2.0 端和 P2.1 端作为模拟串行口与 73M2901 的 TXD、RXD 引脚相连,用于传输 AT 指令以及 DTE 发来的控制信号和数据。DTE 与 AT89C51 的串行口 RXD、TXD 相连,采用 RS－232 接口,并用 MAX232 芯片完成电平转换。

3. 模拟接口电路

模拟接口电路是将 Modem 与通信信道连接起来的接口电路,主要功能如下。

⑴ 调制解调器内部不平衡电路与平衡型通信信道之间的转换;

⑵ 调制解调器内部四线电路与二线通信信道之间的转换;

⑶ 识别沿通信信道传来的交流振铃信号,将它转换成 TTL 直流电平;

⑷ 拨号时,能发出符号规定的脉冲串或双音多频信号。

模拟接口电路包括三部分:拨号脉冲发号电路、振铃检测电路和音频信号通道。

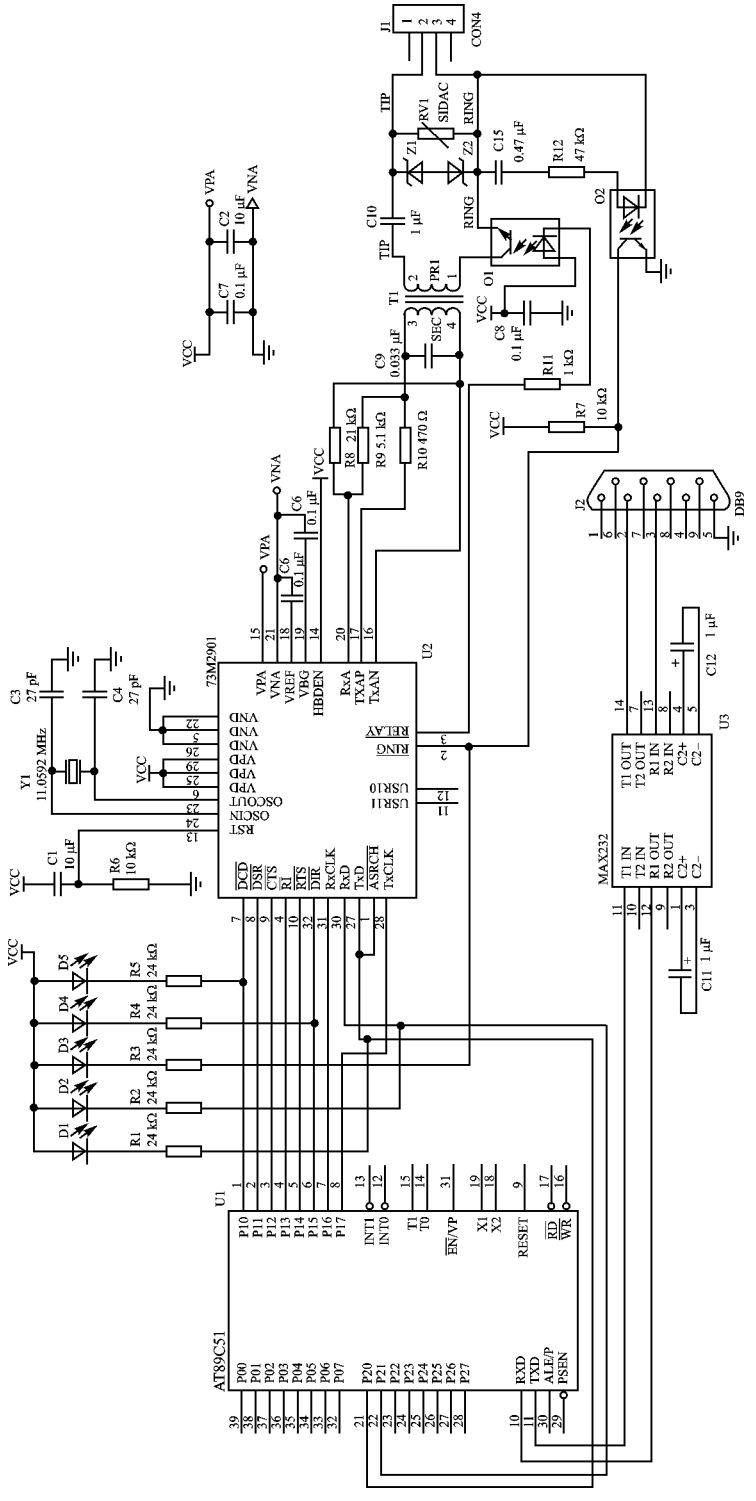

J1
CON4
TIP
RING
RV1
SIDAC
Z1
Z2
C15
0.47 μF
R12
47 kΩ
O2
C10
1 μF
O1
T1
PR1
SEC
C9
0.033 μF
C8
0.1 μF
R11
1 kΩ
R7
10 kΩ
VPA
VNA
C2
10 μF
C7
0.1 μF
VCC
R8 21 kΩ
R9 5.1 kΩ
R10 470 Ω
C6
0.1 μF
J2
DB9
U2
73M2901
VPA
VNA
VREF
VBG
HBDEN
RxA
TXAP
TXAN
RING
REL.AV
USR11
USR10
VND
VPD
OSCOUT
OSCIN
RST
DCD
DSR
CTS
RI
RTS
DIR
RxCLK
RxD
TxD
ASRCH
TxCLK
C3
27 pF
C4
27 pF
Y1
11.0592 MHz
C1
10 μF
R6
10 kΩ
U3
MAX232
T1 OUT
T2 OUT
R1 IN
R2 IN
C2+
C2−
T1 IN
T2 IN
R1 OUT
R2 OUT
C12
1 μF
C11
1 μF
D1
D2
D3
D4
D5
R1
R2
R3
R4
R5
24 kΩ
U1
AT89C51
P10
P11
P12
P13
P14
P15
P16
P17
INT1
INT0
T1
T0
EN/VP
X1
X2
RESET
RD
WR
P00
P01
P02
P03
P04
P05
P06
P07
P20
P21
P22
P23
P24
P25
P26
P27
RXD
TXD
ALE/P
PSEN

图 5.10－2　总体结构

⑴ 拨号脉冲发号电路

拨号脉冲发号电路由光电耦合器 O1,电阻 R11 以及电容 C8 等组成。73M2901 的引脚 3 (RELAY) 控制光电耦合器 O1 完成摘、挂机功能。RELAY端输出高电平时,光电耦合器中的发光二极管没有电流通过,因而不发光,光接收三极管没有基极电流而截止,节点打开,Modem 呈挂机状态;当RELAY端输出低电平时,光电耦合器中的发光二极管有电流通过而发光,光接收三极管导通,节点闭合,Modem 呈摘机状态。73M2901 的RELAY端在 AT89C51 的控制下,在高低电平之间有规律的变化,电路随之打开、闭合,相当于发出了拨号脉冲。

⑵ 振铃检测电路

振铃检测电路由光电耦合器 O2、稳压二极管 Z1、Z2,隔直电容 C15 以及电阻 R7、R12 构成。振铃检测电路将通信信道内的振铃信号检测出来,并转换成 TTL 电平。

当信道内没有振铃信号时,光电耦合器的发光二极管不发光,光接收三极管截止,73M2901 的RING端被 R7 上拉至 +5 V,为无效的高电平;当振铃信号来到时,稳压二极管 Z1、Z2 被击穿,电流流过光电耦合器的发光二极管,发光二极管发光,光接收三极管有电流流过,73M2901 的RING变为有效的低电平,完成振铃检测。

⑶ 音频信号通道

73M2901 的模拟输出端口 (TXAP 和 TXAN 端) 采用差分输出方式,可直接驱动音频耦合变压器;73M2901 的 RXA 端是非平衡的模拟输入端口,接收的音频信号为单端对地的模拟信号。

音频信号通道由音频耦合变压器 T1,平衡网络 R8、R9、R10 组成。音频耦合变压器的匝比为 1:1,介入衰耗不超过 0.4 dB。平衡网络将 TXAP、TXAN 差分输出端的输出信号最大限度的分配到音频耦合变压器的初级上,使得输出功率最大。RV1 的主要作用是消除脉冲干扰,并且保护电路。

三、软件设计

1. 数据传输过程

现假设主站 A 向主站 B 发出呼叫,请求得到主站 B 的有关数据。整个通信过程可以分为以下几步,如图 5.10-3 所示。

⑴ 整个系统首先进行初始化 (使 A、B 两端数据终端就绪 DTR 信号有效)。然后主站 A 向本端 89C51 发出呼叫命令,于是单片机向 73M2901 发出拨号指令,73M2901 收到拨号指令后向被叫端发出拨号音,使被叫端 73M2901 振铃。振铃次数达到软件设置的次数以后,被叫端 73M2901 自动应答,进入摘机状态。

⑵ 被叫端摘机以后,一边向主叫端发送应答载波,一边向本端 89C51 发出 DSR 信号。然后被叫端 89C51 便开始监视 DCD 信号,等待对方载波信号的到来。主叫端 73M2901 检测到应答载波以后,向 89C51 发出 DCD 信号,标志着呼叫成功。

⑶ 呼叫成功后,主叫端 73M2901 向 89C51 发出 DSR 信号,89C51 收到该信号后,得知线路连接已完全建立,即向 73M2901 发出 RTS。73M2901 将向被叫端发出载波并回送 CTS 信号。当主叫端 89C51 收到 CTS 信号以后,即向主站 A 发出信号,表示握手成功,等待主站发出下一命令。

⑷ 被叫端检测到主叫端发来的载波信号以后,就发出 DCD 信号,通知被叫端 89C51 数

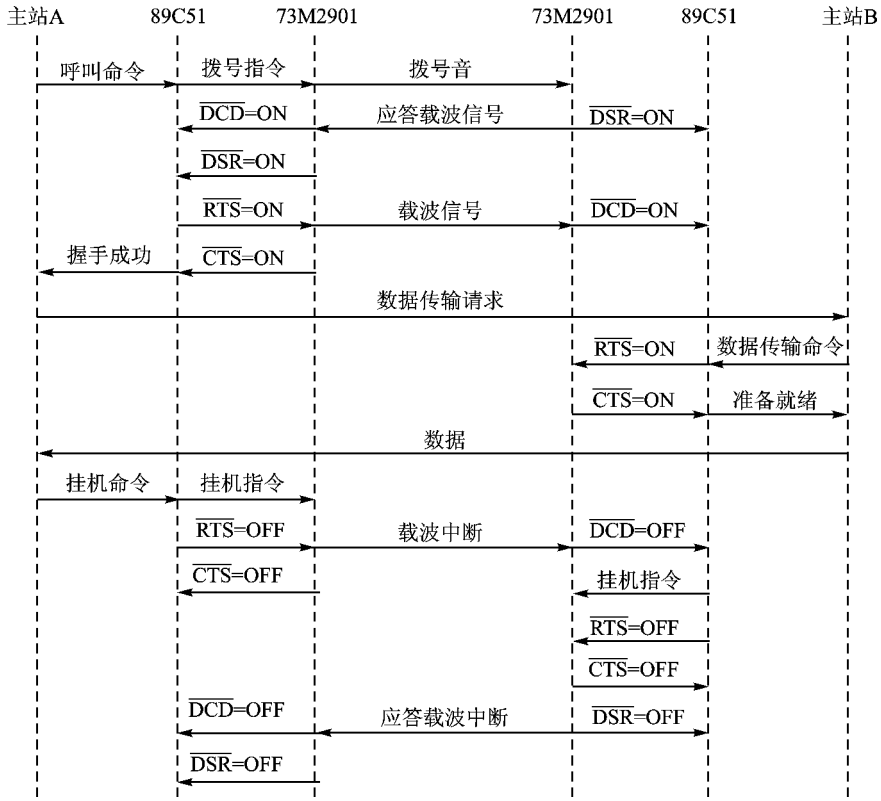


图 5.10－3　数据传输过程

据链路已经建立。

⑸ 在数据链路建立以后，主站 A 便可向主站 B 发出数据传输请求，要求得到有关数据。

⑹ 主站 B 收到主站 A 的数据传输请求后，向本端 89C51 发出数据传输命令。89C51 收到命令以后，向 73M2901 发出 RTS 信号，73M2901 会回送一个 CTS 信号，表明 73M2901 数据传输准备就绪。89C51 收到 CTS 信号以后，向主站 B 发出准备就绪的信号，等待主站 B 向主站 A 传送数据。

⑺ 主站 A 收到主站 B 传来的数据后，重复第 5 步，直到所需数据接收完毕。

⑻ 主站 A 在数据传送完毕后发出挂机命令，主叫端 89C51 向 73M2901 发出挂机指令，并发出无效的 RTS 信号，73M2901 立即停发载波，并回送无效的 CTS 信号。被叫端 73M2901 因收不到主叫端发来的载波信号，便使 DCD 信号无效，被叫端 89C51 即向 73M2901 发出挂机指令，并使 RTS 信号无效。73M2901 挂机并回送无效的 CTS 信号，挂机后 DSR 信号无效，应答载波停发。主叫端 73M2901 因不发载波又收不到载波而使 DCD 信号、DSR 信号无效。至此，数据链路拆除，系统处于待机状态，等待主站再次要求建立连接。

2. Modem 的软件设计

这里首先引入“智能 Modem”的概念。所谓“智能 Modem”，就是使用微处理器和存储器等所组成的控制部件，给 Modem 加上一定的智能。在专用 Modem 中，控制部分是由单片机 89C51 担任的，程序固化在 89C51 中。Modem 带上智能后，可以实现一系列的功能和操作，如自动呼叫应答、传输速率的选择、差错的自动检测和纠正等。

从上文所述的数据传输过程当中可以看出，主叫端在整个通信过程当中依次经历空闲、握手、在线、发送、接收和拆线 6 个状态。被叫端在整个通信过程当中依次经历空闲、接收、在线、发送和拆线 5 个状态。

由此，可以画出程序的整体流程图，如图 5.10－4 所示。Modem 上电之后，先对 73M2901 进行初始化，设定 89C51 中的数据缓冲区，如果此时主站发出拨号命令，则串口产生中断，程序时入主叫状态 如果 73M2901 的$\overline{\text{DSR}}$、$\overline{\text{DCD}}$端为低电平，程序进入被叫状态。

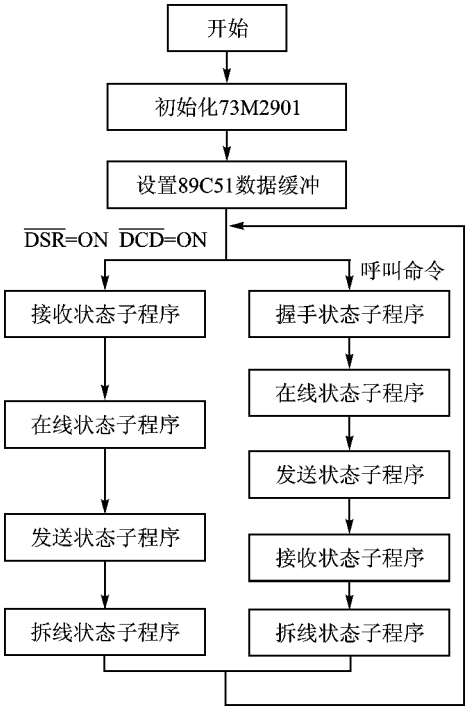


图 5.10－4 程序流程

综上所述，采用 73M2901 芯片的嵌入式 Modem 体积小、可靠性高、灵活方便，非常适合用于通信量不太大的终端设备之间的通信。在配电自动化、远程抄表等方面都有广阔的应用前景。

参 考 文 献

1 增志民. 调制解调器原理及其应用. 人民邮电出版社

2 73M2901 User Guide. TDK Semiconductor Corp

3 73M2901 Advanced Information. TDK Semiconductor Corp

选自《电子技术》月刊，2002 年第 10 期

5.11 电能计量芯片组 AT73C500 和 AT73C501 及其应用

北京工商大学信息工程学院 (100037)
吴叶兰 廉小亲 石芹侠

AT73C500 和 AT73C501 是 ATMEL 公司推出的电能计量芯片。AT73C501 是 6 路 16 位模数转换芯片,AT73C500 是含有高效数字信号处理 (DSP) 内核的计量芯片。这组套片可用来测量几乎所有的电量参数,包括有功功率、无功功率、视在功率、电压、电流、频率、功率因数等,同时还能输出电能表常数脉冲和显示脉冲,既可以单相测量,也可以多相测量。精度等级可达 IEC1036 的 1 级,如果提供外部温度补偿参考电压,则精度可达 IEC687 的 0.5 级或 0.2 级。该组套片为多功能全电子电能表的实现提供了设计方案,下面对该组套片的性能及其在实际中应用情况作一介绍。

一、AT73C501 芯片简介

AT73C501 由六路独立的 16 位模数转换器组成。转换器包含了一个 SIGMA - DELTA 调节器和数字滤波器、参考电压发生器、电压监控模块和串行输出接口。其引脚功能如如下。

(1) 模拟信号

AIN0 ~ AIN5:6 路模拟信号输入,分别送给 6 路转换器转换。每个转换器的满量程输入值为 $2V_{pp}$,对应于所测电压的满量程值为 $270V_{rms}$,所测电流的满量程值为 80 A_{rms}。这里必须说明的是:AIN0 ~ AIN5 的输入电压一定不能超过 $2V_{pp}$,否则有烧坏 AT73C501 的危险。

VCIN:电压监控模块的输入端。实际应用中,可将该模块的工作电源 +5 V 接到此端,电压监控模块可监视 501 模块工作电源情况,如电源下降到 4.2 V,则该监控模块即可向 500 产生报警。

(2) 数字控制信号

BGD:参考电压的旁路输入。当要设计 0.5 级表或 0.2 级表时,由于对温漂要求严格,就不能用 AT73C501 内部的参考电压,而需要使用具有温度补偿的外部参考电压。此时 BGD 端应置为高电平。

CS:片选输入端。

MODE:模式选择控制端。接地。

RESET:复位引脚,高有效。

(3) 输出总线信号

CLK:主时钟输出信号。频率范围为 1 ~ 6 MHz。给 AT73C500 提供主时钟信号。

CLKR:串行总线时钟输出信号。是 CLK 的 2 分频,为 AT73C500 提供串口时钟信号。

ACK:数据准备好应答输出。当 3 路电压和 3 路电流转换完毕,AT73C501 将 ACK 置高,通知 AT73C500 接收数据。

DATA:串行数据输出信号。AT73C501 将转换完的数字信号经 DATA 引脚输出,该数字

信号是 6 个 16 位的采样数据,其顺序为 I1,U1,I2,U2,I3,U3,且高位在前,低位在后。

FSR :输出采样帧信号。悬空。

⑷ 状态信号

PFAIL :电压监控模块的输出。当 VCIN 的输入低于 4.2 V 时,PFAIL 被置高;一旦 VCIN 高于 4.3 V,则 PFAIL 被清 0。PFAIL 用来作为 AT73C500 的中断输入,以保证在掉电时 500 能及时地对数据作出正确的处理。

⑸ 晶振信号

XI :晶振的输入端;XO :晶振的输出端。晶振范围在 1 ~ 6 MHz 间,典型值为 3.276 8 MHz。

⑹ 电源及地信号

VDDA、VDA :模拟电源信号,+5 V 有效。

VCC :数字电源信号,为 AT73C501 的数字电路提供 +5 V 电压。

VSSA,VSA :模拟地。

VGAND :数字地。

AGND :模拟地参考输出,其典型值为 2.5 V。

VREF :参考电压输出。由一精密电源提供,典型值为 3.75 V。

二、AT73C500 简介

AT73C500 接收从 AT73C501 来的采样数据,由内部的 DSP 模块计算出三相的有功、无功和视在功率,以及三相的电压、电流、功率因数等参数,通过 8 位并行总线输出。

1. 引脚功能

⑴ 微处理器总线

B7 ~ B0 :8 位并行总线,是复用线,既可以和外部微处理器的 I/O 线直接相连,输出各种电量参数;也可以作为电能表常数脉冲输出和显示脉冲输出。

⑵ 状态/模式总线

B15 ~ B8 :8 位 I/O 口,复用口,输出 AT73C500 的工作状态或作为 AT73C500 的模式输入。AT73C500 的工作状态各位的含义如表 5.11 - 1 所列。

表 5.11 - 1 T73C500 的工作状态各位的含义

位	含 义
B15	当一相或多相电流为 0 或为负,B15 位被置高
B14	当三相中只要有一相电流低于启动电流时,B14 被置高
B13	第三相的电压高于满量程电压($2V_{pp}$)的 10% 时,该位置高
B12	第二相的电压高于满量程电压($2V_{pp}$)的 10% 时,该位被置高
B11	第一相的电压高于满量程电压($2V_{pp}$)的 10% 时,该位被置高
B10	出现操作错误时被置高,如 AT73C501 和 AT73C500 的串口通信有错,或是 AT73C500 通信有错,与微处理器的通信有错等
B9	为高精度电能表 AT73C500 已将计算完的数据送上总线等待微处理器取走
B8	在 500 初始化时被置低,使 500 和外部 EEPROM 接口,AT73C500 正常工作时为高

注 :B11、B12 和 B13 作为输入时,是表示 AT73C500 的工作模式。

(3) 控制信号

STROBE:选通脉冲输出信号。一个 STROBE 脉冲对应总线上的一个字节数据。

BRDY:微处理器准备好信号,输入低有效。在 BRDY 的下降沿,数据在 STROBE 的上升沿被选通到总线上。

ADDR1:地址输出 1 信号,当它为高电平时,B7 ~ B0 用来输出数据信号。

ADDR0:地址输出 0 信号,高有效时,B15 ~ B8 输出状态信号,B7 ~ B0 上输出脉冲信号。

RD/WR:500 的读写信号。

这 5 个控制信号都用来作为 AT73C500 和微处理器间的握手信号。

(4) 和 AT73C501/EEPROM 的接口信号

SOUT0:串行输出信号,作为 EEPROM 的时钟信号。

SOUT1:串行输出信号,作为 AT73C501 的片选信号和 EEPROM 的数据输入信号。

SIN:串行数据输入信号,数据来自 AT73C501 或 EEPROM。

SCLK:串行输入时钟信号,来自 AT73C501 的 CLKR。

(5) 数字输入信号

CLK:主时钟信号,由 AT73C501 的 CLK 提供。

XRES:复位信号,低有效。在复位期间,AT73C500 要从 EEPROM 中读取校正系数,并从 B11 ~ B13 中读取 AT73C500 的工作模式。

IRQ0:中断输入信号,和 AT73C501 的 PFAIL 相连。

IRQ1:中断输入信号,和 AT73C501 的 ACK 相连。

2. AT73C500 的工作方式

AT73C500 有六种工作模式,如表 5.11 - 2 所列。

表 5.11 - 2 AT73C500 有六种工作模式

模式编号	操作模式	校正系数存放的位置
0	Notinuse	
1	Nomal operation	EEPROM
2	Multi - channel operation	EEPROM
3	Nomal operation	Micro - processor
4	Multi - channel operation	Micro - processor
5	Testmode	None
6	Notinuse	
7	Nomal operation	EEPROM

AT73C500 支持两种配置:单独配置和微处理器配置。这两者的惟一区别是读取校正系数的不同。校正系数既可以放在 EEPROM 中,也可以放在微处理器中,AT73C500 允许两种配置以各种方式联合使用。作者在实际使用中是把校正系数放在 EEP - ROM 中,由微处理器读取 AT73C500 中的数据加以处理。

从表 5.11 - 2 中可以看出,除了校正系数存放的位置不同,AT73C500 的操作模式实际上只有 3 种,即正常测量模式、多通道模式和测试模式。

在正常测量模式中,AT73C500 要计算出各个电参数。首先进行数字的高通滤波,其目的

是从电压和电流采样信号中去掉直流分量,并通过乘法和加法运算计算出有功功率 P 和无功功率 Q。由 P、Q 值很容易得到视在功率 S 和功率因数 PF,如 PF 的计算公式

$$PF = \sin(Q) \times \frac{abs(P)}{abs(S)}$$

各相的电压是一均方根(有效)值,而电流则由视在功率和各相电压相除得到。频率计算是在线频率和 AT73C500 的采样频率相比较的基础上得到的,测量范围在 20 ~ 350 Hz 之间。这些计算出的电参数被放在 25 个寄存器中,寄存器的位长为 16 位或 32 位,AT73C500 以 6 个数据包的形式从并口 B7 ~ B0 发送出去,每 200 ms 更新一次。同时,正常测量模式还提供 8 位脉冲输出,4 位电能表常数脉冲,4 位显示脉冲。这些脉冲信号应该被外部锁存以保证总线 B7 ~ B0 上的数据信号不被丢失。

多通道模式下,AT73C500 是以三个独立的单相表的方式工作,除脉冲输出和正常模式不同外,其基本工作时序和其他操作基本相同。

测试模式下,AT73C501 传给 AT73C500 的串行数据除 6 个电压/电流值外,还加上了 4 字节的同步字节和状态模式字节。该方式用在初始化校准中或用在对采样数据作额外处理的特殊电表中。在实际应用中,一般采用正常测量模式。

三、AT73C500 和微处理器的接口

由于 AT73C500 的主时钟频率是 3.2768 MHz,它对单片机的速度要求很高,在我们设计的电参数测试仪中采用了 ATMEL 公司的产品 AVR8515。这款芯片采用了增强的 RISC 结构,内载 Flash,8 MHz 晶振,在一个时钟周期内(125 ns)即能完成一条指令的执行,完全能够满足 AT73C500 对速度的要求。AT73C500 与 AVR8515 的接口电路如图 5.11 - 1 所示。

AT73C500 和 AT8515 的数据传输是通过 8 位并口 B7 ~ B0 实现的。为了保证数据的正确传输,用 74HC374 进行了锁存。AT73C500 和 CPU 之间的控制信号是 DATARDY 和 STROBE,其时序如图 5.11 - 2 所示。

DATRDY 信号由 B9 和 RD/WR 产生,指明一包数据的开始;STROBE 信号由 AT73C500 内部产生,表示一个有效字节的开始,所以用它作为 AT8515 的中断信号。

AT73C500 传给 CPU 的数据共有 6 个包(package0 ~ 5),其时序图如图 5.11 - 3 所示。

每个包由 16 字节组成,各包的头 4 个字节分别为:Sync LS、Sync MS、Mode、Status。Sync LS 指明包的序号,其值为 00H ~ 05H;Sync MS 是个常量 80H;Mode 指明了 AT73C500 的工作模式,其值为 00H ~ 07H;Status 反映了 AT73C500 的工作状态,即 B15 ~ B8 的内容。各个包的其余 12 个字节分别是数据信息,包含了 AT73C500 内部计算得到的所有电参数。

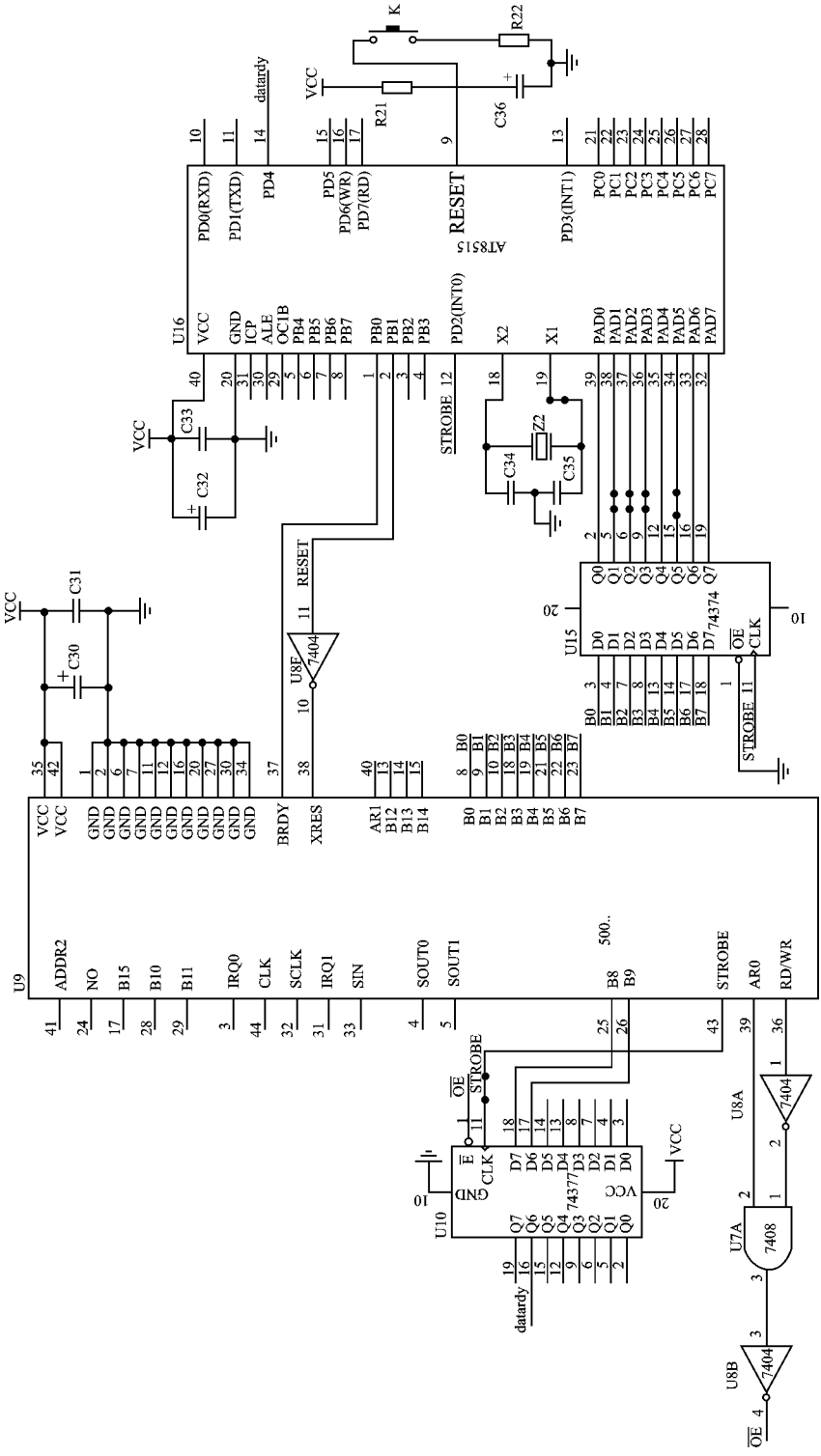


图 5.11－1　AT73C500 和 AT8515 的接口电路

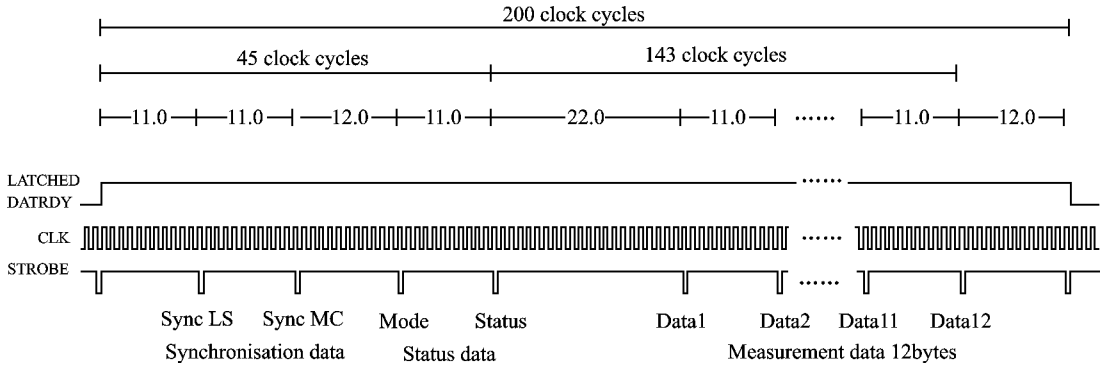


图 5. 11 − 2 一个数据包的字节时序

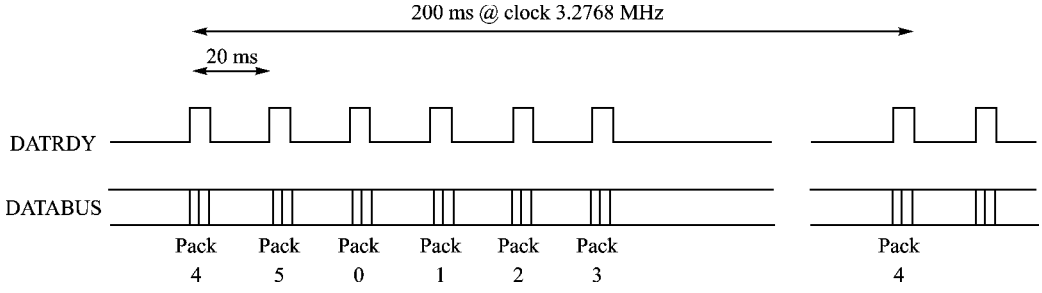


图 5. 11 − 3 时序图

四、误差的来源

AT73C500 和 AT73C501 套片构成了电子式电能表的内核。由于电表间的离散性，且给 AT73C501 提供信号的前端电路中所使用的元件各不相同，如 501 电压前端输入电路，实质上是一个电阻分压器，而电路中所使用的电阻值存在离散性，501 电流前端输入电路中所用的电阻、滤波电容的离散性以及电流互感器（CT_s）所引起的相移，所有这些都导致了电能计量芯片组计量误差产生的原因。

五、校正系数及其含义、公式

AT73C500 和 AT73C501 套片构成了电子式电能表的内核。由于电表间的离散性，且给 AT73C501 提供信号的前端电路各不相同，所以 AT73C500 提供了校正系数来补偿可能出现的误差。

1. 校正系数及其含义

AT73C500 提供了校正系数来补偿可能出现的误差。表 5. 11 − 3 说明了校正系数的含义及其校正范围。

表 5.11－3 校正系数的含义及其校正范围

校正系数名称	含 义	校正范围
PCN	第 N 相的相位校正因子	±5.625°
AGCN	第 N 相有功功率和有功能量的增益校正因子	±20%
RGCN	第 N 相无功功率和无功能量的增益校正因子	±20%
UGCN	第 N 相相电压的增益校正因子	±8%
AOFN	第 N 相有功功率和有功能量的偏置校正因子	满量程的±0.4%
ROFN	第 N 相无功功率和无功能量的偏置校正因子	满量程的±0.4%
MCC	有功能量和无功能量的显示调整因子	1...6
STUPC	启动电流的调节因子	0.2...10
OFFMOD	偏置因子的控制模式	0/非 0

说明 表 5.11－3 中 N 的范围为 1～3，校正系数包括 9 个增益校正、6 个偏置校正、3 个相位校正、显示脉冲速率校正和启动电流校正。其中增益校正是对各相的有功功率、无功功率和电压的校正，用来补偿电压分压器、电流变压器和 A/D 转换器积累的量值误差，当这些参数被校正后，AT73C500 自动校正各相的视在功率和电流。偏置校正是针对各相的有功功率和无功功率，用来补偿测量中的非线性误差。相位校正用来补偿由于电流互感器的相移而导致的电压和电流相位间移位误差。启动电流校正用来补偿因启动电流低于基本电流的 0.4% 时产生的能量误差。

2. 校正系数公式

⑴ 相位校正

$$PO_N = \frac{PC_N}{128} \times 5.625°$$

式中 PO_N——第 N 相 U 与 I 的实际相位差与理论相位差的差值；

PC_N——N 相的相位校正因子。根据 PO_N 的值可计算得到 PC_N 的值。PC_N 的范围为 -128 ～ +127，相位补偿范围为 -5.625° ～ +5.581°。

⑵ 增益校正

$$P_N = P_N \times \left(1 + 0.2 \times \frac{AGC_N}{128}\right)$$

$$Q_N = Q_N \times \left(1 + 0.2 \times \frac{RGC_N}{128}\right)$$

$$U_N = U_N \times \left(1 + 0.08 \times \frac{UGC_N}{128}\right)$$

式中 等式右边的 P_N、Q_N、U_N——第 N 相的实际测量的有功功率、无功功率和电压；

等式左边的 P_N、Q_N、U_N——第 N 相的校正后的有功功率、无功功率和电压值；

AGC_N、RGC_N、UGC_N——第 N 相有功、无功、电压的增益调节因子，范围是 -128 ～ +127，P_N、Q_N、U_N 的调整范围分别是原值的 -20% ～19.84%、-20% ～19.84% 和 -8% ～7.94%。

⑶ 偏置校正

$$P_N = P_N + \frac{AOF_N}{128} \times 0.004157 \times \sin(P_N) \times P_{FS}$$

$$Q_N = Q_N + \frac{POF_N}{128} \times 0.00457 \times \sin(Q_N) \times Q_{FS}$$

式中 等式右边的 P_N、Q_N——第 N 相的实际测量的有功功率、无功功率；

等式左边的 P_N、Q_N——第 N 相的校正后的有功功率、无功功率值；

AOF_N、ROF_N——第 N 相有功、无功功率的偏置调节因子，范围是 -128 ~ +127；

P_{FS}、Q_{FS}——满量程有功功率、无功功率值。

⑷ 启动电流调节

$$I_{SU} = \frac{1}{4000} \times I_{FS} \times (1 + 0.2 \times STUPC)$$

式中 I_{SU}——启动电流；

I_{FS}——满量程电流值；

STUPC——启动电流因子。STUPC 范围是 -4 ~ +45，I_{SU}的调整范围为 0.2 ~ 10 倍的原值。

⑸ 显示脉冲速率调整

$$PR = \frac{1000}{(25 + MCC) \times E_{LSB}}$$

式中 PR——显示脉冲速率；

MCC——显示脉冲调整因子。MCC 范围是 0 ~ +127，PR 调整范围为 1 ~ 0.16 倍原值。

其中

$$E_{LSB} = \frac{U_{FS} \times I_{FS}}{270\ V \times 80\ A} \times 0.4\ Wh$$

式中 U_{FS}——满量程电压值；

I_{FS}——满量程电流值。

⑹ 校正系数求取过程

首先设定电压、电流和相位为特定（如 +60°）的值，校正系数的求取按照一定的顺序进行，其过程如下。

⑴ 校正系数设为全零，装入 EEPROM 中，在给定条件下，将 AT73C500 测量的每相电压、电流的实际相位差和理论的电压、电流相位差作比较，得结果 PO_N，再根据相位校正公式求得 PC_N。

⑵ 将 PC_N 的值装入到 EEPROM 中，在相同条件下测量有功功率、无功功率和电压，并和理论值一并代入名增益校正公式，得到 9 个增益校正因子。

⑶ 将 AGC_N、RGC_N、UGC_N 的值同时装入到 EEPROM 中，并将给定条件下的电压、电流相位差调整为 0°，得到每相的有功功率，和理论值一并代入有功功率的偏置校正公式，即可求得 AOF_N。

⑷ 将 AOF_N 的值装入到 EEPROM 中，并将给定条件下的电压、电流相位差调整为 90°，得到每相的无功功率，和理论值一并代入无功功率的偏置校正公式，即可求得 ROF_N。

⑸ 将 ROF_N 的值装入到 EEPROM 中，根据启动电流的大小和显示脉冲的速率按照启动电流系数和显示脉冲速率校正系数公式，即可求得 STUPC 和 MCC。

六、结 论

AT73C500 和 AT73C501 套片是 ATEML 公司推出的专门用于多功能电表的专用芯片组，能将采集到的模拟量转换成数字量，整个过程不受外界控制，只要满足它们之间的时序关系，就能得到正确的数据。在我们设计的三相电参数测试仪中采用了该款套片，取得了良好的效果。由于该组套片具有速度快、精度高、输出的数据信息量大等诸多优点，相信它在三相多功能电表的设计中具有较好的应用前景。

但是在实际应用中，我们发现校正系数对 AT73C500 套片的测量精度有很大的影响。从实际测得的数据分析来看，在装载校正系数之前，电压电流的实测值和理论值误差平均在 ±1.34% 左右，装载校正系数后的误差在 ±0.3% 左右，测量精度大大提高，表明了套片计量误差的校正方法切实可行。

参 考 文 献

1 ATMEL 公司 CD－ROM 资料盘. 2001 [Z]

2 耿德根，宋建国，马潮，叶勇建. AVR 高速嵌入式单片机原理与应用 [M]. 北京航空航天大学出版社，2001

选自《电测与仪表》月刊，2002 年第 7 期

第六章

总线技术

6.1 PCI 总线及其接口芯片的应用

西南交通大学 董晓明 张翠芳

PCI (Peripheral Component Interconnect) 总线是一种高性能局部总线,是为了满足外设间以及外设与主机间高速数据传输而提出来的。在数字图形、图像和语音处理,以及高速实时数据采集与处理等对数据传输率要求较高的应用中,采用 PCI 总线来进行数据传输,可以解决原有的标准总线数据传输率低带来的瓶颈问题。

一、PCI 局部总线特点

(1) 传输率高:在 33 MHz 的时钟频率下,对于 32 位的 PCI 总线,峰值数据传输率可以达到 132 MB/s;64 位的 PCI 总线可达 264 MB/s。对于 64 位的 66 MHz 时钟的 PCI 总线,可以达到 528 MB/s,远远大于标准 ISA 的 5 MB/s 和 EISA 的 33 MB/s 传输率。

(2) 线性突发传输:减少了地址操作,更有效地利用总线的带宽来传输数据,可以确保总线满载数据。

(3) 采用独立于处理器的结构:因为 PCI 总线开发的设备是针对 PCI 的,不受处理器的限制,所以 PCI 设备的设计独立于处理器的升级。

(4) 自动配置功能:每个 PCI 设备有 256 字节的配置寄存器,可以实现设备的即插即用。

(5) 软件透明:在与 PCI 设备通信时,软件驱动程序使用相同的命令集和状态定义。

二、PCI 接口的两种设计方法

PCI 接口的设计必须符合 PCI 总线规范定义的电气特性和时序要求。有两种 PCI 接口的实现方案:使用可编程逻辑器件 (FPGA 或 CPLD) 和专用 PCI 接口芯片。使用可编程逻辑器件,可以选择实现部分 PCI 规范的一个子集,这种方法比较灵活,但开发难度大,开发周期长。采用专用 PCI 接口芯片,可以缩短开发周期,降低开发难度。

三、CY7C09449PV 介绍

CY7C09449PV 是 Cypress 公司推出的 PCI 主/从接口芯片,符合 PCI2.2 规范,可以直接与很多微处理器进行无缝连接。CY7C09449PV 提供 16 KB 的双端口共享存储器 (SRAM),用来在 PCI 总线和局部处理器间传输数据;I_2O 消息传输单元具有 4 个 32 位 FIFO,用来实现消息队列和中断功能;局部总线时钟最大 50 MHz,单一 3.3 V 电源供电;对 3.3 V 和 5 V 的 PCI 信号兼容,使用 160 引脚的 TQFP 封装。

1. CY7C09449PV 结构

CY7C09449PV 的结构如图 6.1-1 所示。

2. 功能模块介绍

CY7C09449PV 提供 64 字节的 PCI 头标区配置寄存器空间。其中 Vender ID、Device ID、

Revision ID、Header Type、Class Code 用于设备的识别。命令寄存器（Command）包含设备控制位，包括允许存储器读写响应、I/O 读写响应等。状态寄存器（Status）用于记录 PCI 总线相关事件的状态信息。基址寄存器 0（BAR0）提供设备在 PCI 存储空间起始地址，31～15 可读写，通知系统 BIOS 此设备所要求的 PCI 存储空间为 32 KB，任何对 31～15 位与 BAR0 相符合 PCI 存储空间访问的，CY7C09449PV 都会响应并接受传输。基址寄存器 1（BAR1）是 CY7C09449PV 的 I/O 指针空间的地址。

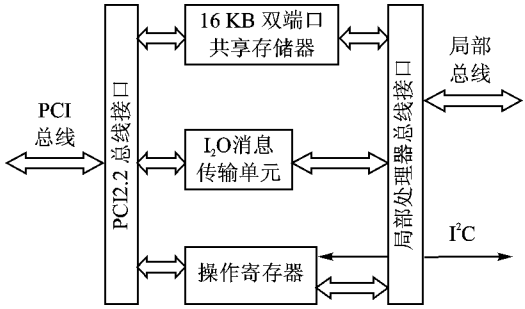


图 6.1－1 CY7C09449PV 结构框图

CY7C09449PV 提供 PCI 总线接口、局部处理器总线接口和 I^2C 串行 EEPROM 接口，以及内部 16 KB 双端口 SRAM、I_2O 消息传输单元和控制寄存器。数据的传输可以在 PCI 总线和局部处理器总线间进行。PCI 总线和局部总线都能通过操作寄存器对 I^2C 接口进行操作。

PCI 总线和局部处理器总线间可以通过共享的双端口存储器进行数据交换。由于共享存储器是“临界资源”，CY7C09449PV 在仲裁寄存器（ARB_FLAGS）中，为 PCI 总线和局部处理器总线各提供了 4 个仲裁标志位（L0，P0，L1，P1，L2，P2，L3，P3），来对共享存储器进行并发互斥访问。为了实现线性突发传输，提供了 DMA 控制寄存器，PCI 总线和局部处理器总线都能通过 DMA 控制寄存器，来启动突发传输。为了保证对 DMA 控制器的互斥访问，DMA 控制寄存器也提供了仲裁标志位。CY7C09449PV 为局部处理器提供了一个映射到整个 PCI 地址空间的 8 KB 存储器窗口。通过这个窗口，局部处理器可以不经过共享存储器，直接对 PCI 地址空间进行访问，直接访问寄存器（DAHBASE）负责对这个窗口的操作。

CY7C09449PV 的 I_2O 消息传输单元具有符合智能 I/O（Intelligent I/O）1.5 规范的消息队列和中断功能，由 4 个深度为 32 的 32 位 FIFO（Inbound Free/Post，Outbound Free/Post）和共享存储器来实现。这 4 个 FIFO 也能用作通用 FIFO。

PCI 总线和局部处理器间的消息也能通过邮箱来传输。主机通过写 Host to Local Data Mailbox（HLDATA）邮箱寄存器，向局部处理器传送消息数据，并产生中断标志位，等局部处理器读取消息数据后，中断标志位自动复位。局部处理器能通过写（Local To Host Data Maibox）邮箱寄存器，向主机发送消息数据，并产生中断标志位。主机读取消息数据后，标志位自动复位。

通过 I^2C 接口连接 EEPROM，可以保存 CY7C09449PV 的初始化数据。这些数据包括 PCI 总线和局部处理器总线的配置信息。在复位完成后，CY7C09449PV 在响应 PCI 总线和局部总线交易以前自动下载这些数据，进行初始化操作。通过 I^2C 控制寄存器组可以对 I^2C 端口进行读写。I^2C 控制寄存器组包括 3 个寄存器：I^2C 命令寄存器（NVCMD）、I^2C 读数据寄存器（NVREAD）、I^2C 状态寄存器（NVSTAT）。

CY7C09449PV 的中断控制器为 PCI 总线和局部处理器总线分别提供了独立的中断掩码和命令/状态寄存器：HINT（主机中断控制和状态寄存器）、LINT（局部处理器中断控制和状态处理器）。中断源有：DMA 完成、邮箱、FIFO 非空、FIFO 溢出等，以及 1 个外部中断引脚。内

部资源如表 6.1－1 所列。

表 6.1－1 CY7C09449PV 内部资源存储映射

存储模块	偏移地址[14: 0]	大小/KB
I_2O 寄存器	0x0000-0x03FF	1
操作寄存器	0x4000-0x07FF	1
保留空间	0x0800-0x1FFF	6
PCI 直接访问窗口	0x2000-0x3FFF	8
共享双端口存储器	0x4000-0x7FFF	16

四、基于 PCI 与 C32 数据采集系统的设计方案

C32 是一种 32 位的浮点 DSP,具有较高的性价比,在实时数据采集与高速信号处理中得到了广泛的应用。使用 CY7C09449PV 作为接口芯片,可以实现 PCI 总线与基于 C32 的 DSP 系统间的连接。

将 CY7C09449PV 的 RSTOUTD 引脚与 C32 的复位信号相连,可以实现主机对 C32 进行复位。I^2C 接口引脚要接 2.2 kΩ 到 10 kΩ 的上拉电阻。接口输入引脚在不用时,要上拉到高电平或接地。图 6.1－2 中的 PLD 实现 C32 对 CY7C09449PV 和 A/D 转换器的片选,以及 C32 的复位 BOOT LOADER 所需的信号逻辑。EPROM 存储 C32 程序,SRAM 存放数据和运行时的程序,EEPROM 存放 CY7C09449PV 的初始化信息。

系统结构如图 6.1－2 所示。

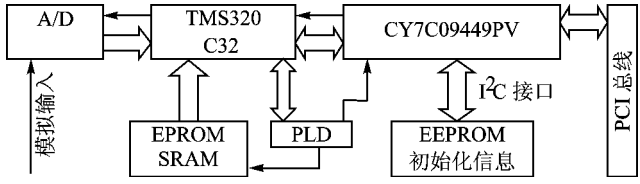


图 6.1－2 基于 PCI 的数据采集系统

采集到的数据有两种方案进行传输,使用 FIFO 或 DMA。FIFO 传输可以保证顺序的传输,但传输效率低,适用于速率要求较低的应用。使用 DMA 进行传输,能充分利用 PCI 总线的突发传输能力,适用于对传输率要求较高的情况。共享存储器分为模块 A 和模块 B。DSP 先把采集到的数据写入模块 A,等数据写满后,启动模块 A 的 DMA 传输;同时将接下来采集到的数据写入模块 B,等模块 B 数据装满后,启动模块 B 的 DMA 传输。这样循环下去,可以使数据的采集和 DMA 传输并行进行。

参 考 文 献

1 李贵山,戚德虎. PCI 局部总线开发者指南. 西安:西安电子科技大学出版社,1997

2 CY7C09449PV Data Manual. Cypress,2000

3 Tom Shanley, Don Anderson. PCI System Architecture. Mindshare, 2000

6.2 实现 RS485/RS422 和 CAN 转换——总线网桥的构建

西安科技工程学院信控系 (710048) 丁学文 张 琦 杨 玲

一、引 言

现场总线控制系统 (FCS) 将是 21 世纪自动控制系统发展的主流,是继集中式数字控制系统、集散控制系统 DCS 后的新一代控制系统。现场总线是综合自动化发展的需要,而智能仪表为现场总线的出现奠定了基础。CAN (Controller Area Network) 是现场总线的一种,最早用于汽车内部检测部件与执行部件间的数据通信。由于本身的特点,其应用范围已不再局限于汽车工业,而向过程控制、机械工业、机器人、数控机床、医疗器械、智能建筑等领域发展。CAN 已形成国际标准,并已被公认为几种最有前途的现场总线之一。

CAN 属于总线型串行通信网络,与一般的通信总线相比,有下列特点:

⑴ 多主工作方式,网络上任何节点可在任意时刻主动向网络上其他节点发送信息,不分主从。当多个节点同时向总线发送信息时,优先级较低的节点会主动退出发送,等待总线空闲时再发。采用非破坏性总线仲裁技术,大大节省了总线冲突仲裁时间。

⑵ 只需通过报文滤波即可实现点对点、一点对多点及全局广播几种方式收发数据,无须专门的调度。直接通信距离最远可达 10 km (5 Kbps),通信速率最高可达 1 Mbps (40 m)。

⑶ 采用短帧结构,传输时间短。具有极好的检错效果和出错重发功能,出错率极低。在严重出错的情况下,节点具有自动关闭功能,以使总线上其他节点的工作不受影响。

CAN 总线通信控制器中集成了 CAN 协议的物理层和数据链路层功能,可完成对通信数据的成帧处理,包括零位的插入/删除、数据块编码、循环冗余检验、优先级判别等项工作。CAN 协议的一个特点是废除了传统的站地址编码,而代之以对通信数据块进行编码。采用这种方法的优点是可使网络内的节点个数理论上不受限制,数据块的标识码可由 11 位 (按 CAN 技术规范 2.0A) 或 29 位 (按 CAN 技术规范 2.0B) 二进制数组成,因此可以定义 2^{11} 或 2^{29} 个不同的数据块。这种按数据块编码的方式,还可使不同的节点同时接收到相同的数据,这一点在分布式控制系统中非常有用。数据段长度最多为 8 个字节,可满足通常工业领域中控制命令、工作状态及测试数据传送的一般要求。同时,8 个字节不会占用总线时间过长,从而保证了通信的实时性。CAN 协议采用 CRC 校验并可提供相应的错误处理和重发功能,保证了数据通信的可靠性。

RS422/RS485 是一种较早流行的串行通信协议,使用广泛,很多变频器和可编程控制器带有 RS485/RS422 接口。由于采用了平衡传输技术,RS485/RS422 比 RS232 (使用单端传输技术) 传输的距离远。但和 CAN 相比,RS485/RS422 只有物理层,构成的通信系统只能采用主从结构,使用不便。另外它在传输速度、传输距离和传输可靠性等性能上也不如 CAN。如果想把具有 RS485/RS422 接口的自动控制装置、智能仪表等接入 CAN 总线,以构成性能更好的

测控系统，一个能把 RS485/RS422 协议和 CAN 协议相互转换的桥接器是必不可少的。

二、RS422/CAN 桥接器的构成

从硬件方面考虑，这个桥接器应能把符合 RS485/RS422 标准的逻辑电平和符合 CAN 标准的逻辑电平相互转换；从软件方面考虑，这个桥接器应能把按 RS485/RS422 协议传输的字节和按 CAN 协议传输的帧相互转换。图 6.2－1 示出了一个 RS422/CAN 桥接器的硬件原理图（注：RS422 是全双工，RS485 是半双工）。图中芯片 MAX1490B 完成 RS422 总线逻辑电平对 TTL 逻辑电平的转换，芯片 SJA1000 完成 TTL 逻辑电平对 CAN 总线逻辑电平的转换和其他的 CAN 通信协议，CPU89C52 完成软件方面的功能，把按 RS485/RS422 协议传输的字节和按 CAN 协议传输的帧相互转换。MAX1490B 是一种具有完全隔离功能的 RS422 接口芯片，它的 TTL 侧和 RS422 侧实现了完全的电隔离。其中 A、B 为 422 侧驱动输入端，Y、Z 为 422 侧驱动输出端（注：全双工），DI 为隔离的 TTL 侧驱动输入端，RO 为隔离的 TTL 侧驱动输出端。MAX1490B 内部具有 DC/DC 隔离变换器，无须外部提供隔离电源，只需一个单一的 5 V 电源。芯片 SJA1000 是新一代的 CAN 通信控制器，兼容老的 82C200，由它实现 CAN 物理层和数据链路层协议。82C250 是 CAN 总线收发接口电路，借此可以扩大带负载能力，可支持多达 110 个节点相连接。

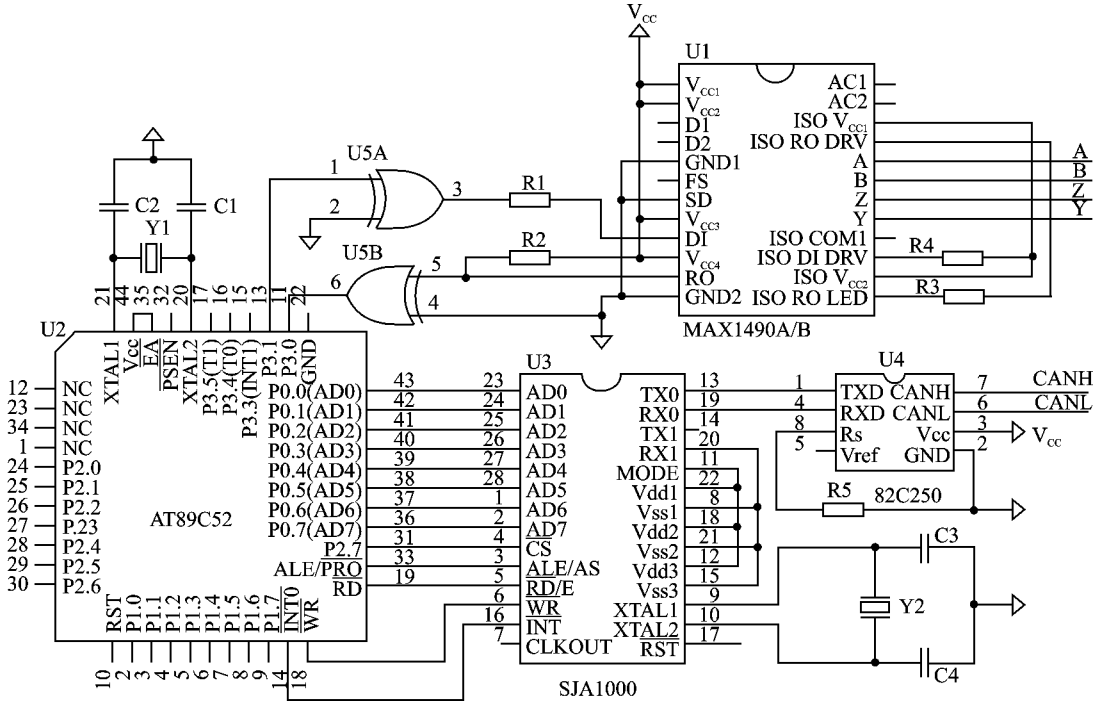


图 6.2－1 RS422/CAN 桥接器原理图

三、RS422/CAN 桥接器应用实例

图 6.2－2 示出了一个 RS422/CAN 桥接器应用实例简图。图中用一台工控机通过 CAN 总线控制 11 台变频器实现同步调速。这里使用的变频器是三菱的 FRA500，它带有RS－422 通信接口，波特率有 4.8 Kbps、9.6 Kbps 和 19.2 Kbps 三种可选。通过串行通信口可以对变频器进行设定，包括频率给定；可以对变频器进行监控，包括电压、电流、转速、转矩和故障等；可以对变频器进行控制，包括启动和停止操作等。由于固有的主从通信方式所带来的麻烦以及通信速率不高，使得对单台变频器通信时，实时性尚且可以，但用轮询方法对 11 台变频器通信时，实时性太差，难以满足系统同步的要求。为了解决这个问题，给每台变频器配了一个 RS422/CAN 桥接器。工控机通过 CAN 总线实现与变频器的通信，如图 6.2－2 所示。CAN 总线为多主结构，各节点用载波监听多路争用的方法取得总线使用权，ID 号较小的帧有较高的优先权，免除了建立通信的麻烦。CAN 的通信速率比较高，高达 1 Mbps，大大高于变频器的 RS－422 口所允许的通信速率。实际使用证明，这个方案比较好地满足了系统的要求。

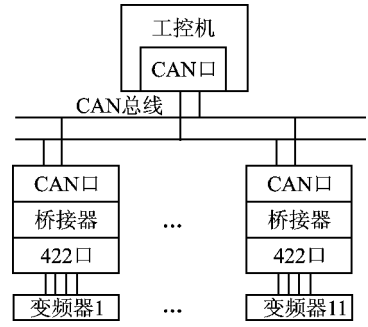


图 6.2－2 RS422/CAN 桥接器应用实例

四、相关程序流程图

桥接器初始化流程图如图 6.2－3 所示。

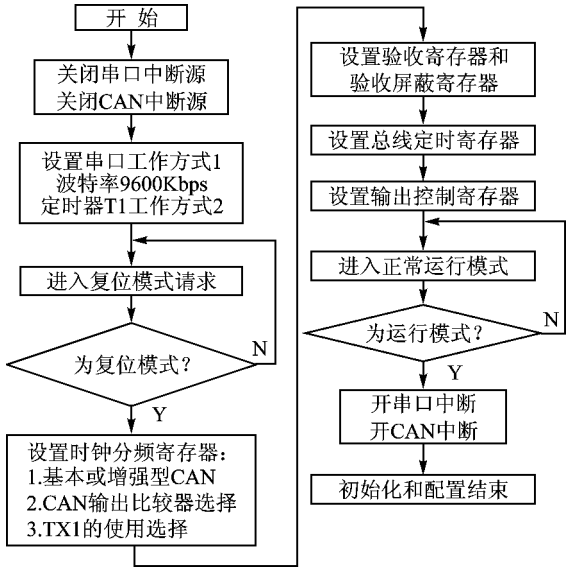


图 6.2－3 桥接器初始化流程图

参 考 文 献

1 SJA1000 Stand-alone CAN controller，Philips Semiconductors APPLICATION NOTE，1997，12

2 mAX1490B，BS－485/RS－422 Data Interface，MAXIM Products Manual，1999，11

3 8-Bit Microcontroller with 4K Bytes Flash At89C51，ATMEL Application Note，1997，12

4 郭宽明. CAN 总线原理和应用系统设计. 北京 北京航空航天大学出版社，1996

5 郭宽明. 单片机外围器件实用手册——数据传输接口器件分册. 北京 北京航空航天大学出版社，1998

6 何立民. 单片机应用技术选编. 北京 北京航空航天大学出版社，1999

选自《电气自动化》月刊，2002 年第 3 期

6.3 工控系统应用 CAN 总线的几种改进方法

清华大学自动化系 (100084) 孙 敏 梁泰文 阳宪惠

CAN (Controller Area Network) 即控制器局域网,最初是为汽车内部网络系统而设计,其特点是:定义明确、建网容易且廉价,不需要交换机或集线器;不基于令牌,响应时间短。由于这些极具吸引力的特点,使得 CAN 在工业领域得到了广泛的应用。

然而,传统 CAN 协议的媒体访问技术使它存在着内在的缺陷。为了让总线仲裁机制能够正确地执行,必须满足以下条件:信息在媒体上的传输延迟时间相对于位时间 (bit time) 来说可以忽略。这个限制条件对网络扩展产生了严重制约,因而 CAN 的标准把最大传输速率限制在 1 Mbps。在 1 Mbps 的速率即位时间为 1 μs 时,CAN 总线最大的传输距离大约为 40 m,这个距离对于许多工业控制现场设备来说是太短了。

另外,随着工业控制领域分布式自动化系统复杂性的增加,控制网络要求传输的信息量也越来越大,对网络传输速率的要求也随之提高,因而传统的 CAN 已经难以满足这些要求。

一、CAN 的基本技术

CAN 是多主站的总线型网络,遵循 ISO/OSI 标准。CAN 被划分为物理层、数据链路层和应用层。其中数据链路层又被分为 LLC (逻辑链路控制) 子层和 MAC (媒体访问控制) 子层。

CAN 采用总线仲裁技术,不使用与系统结构相关的任何信息 (如节点地址等),因此系统中的节点性能基本与它们在网络中地具体位置无关。CAN 采用短帧结构,数据在 CAN 上传输时,通过报文滤波实现点对点、一点对多点以及全局广播等几种传播方式,并用差错检测和差错处理保证正确的数据交换。

CAN 总线上的传输是基带传输,任一时刻总线处于以下三种状态之一:“空闲”、“显位”或“隐位”。在总线具有“线与”能力时,“显位”用逻辑“0”表示,“隐位”用逻辑“1”表示。当“显位”与“隐位”同时发送时,总线上表现为“显位”。

由于基本的 CAN 协议无法满足工业控制领域越来越高的网络范围和数据传输的要求,近年来提出了多种对 CAN 协议的改进方法。

二、采用动态优先级提高系统的实时性

传统的 CAN 采用静态优先级分配机制,缺少非周期数据传输的定时信息,因此静态优先级分配无法总能满足周期限制。而且在网络负载繁重的情况下,大量的数据传输要求可能阻塞,造成严重的传输延迟。可采用动态优先级策略,来提高系统的实时性及公平性。

1. 分布式优先级队列 DPQ 机制

分布式优先级队列 DPQ (Distributed Priority Queue) 机制采用了基本 CAN 协议中的扩展格式数据帧的结构实现,因此能保持对传统 CAN 系统的兼容。图 6.3-1 (b) 所示为 DPQ 帧的标识符域的结构。

优先级类 PC (Priority Class) 字段占用 2 bit,用于服务优先级的编码,表示对象的紧急度,服务优先级分为四类。QP (Queue Position) 字段占用 8 bit,用于表示节点的队列位置。

DPQ 机制建立在全局分布式 FIFO 队列上,所有使用该机制的字节都被虚拟的插入该队列中。每个节点在本地存储并动态更新自己的全局虚拟队列中的位置,记为 L. QP,也即自己的节点优先级。系统中总共有 4 个队列,每个队列对应于一个服务优先级类,因此每个节点上在本地保存了分别对应不同服务优先级类的 4 个节点优先级值。

当节点发送一帧时,记该帧为 F,首先将任务的服务优先级类 PC 复制到帧 F 的 PC 域中,并将对应于该服务优先级类的节点优先级 L. QP 复制到该帧的 F. QP 中,而基本标识符字段的其余位用于指示该帧在同类中的优先级。如果该帧发送成功,则节点的相应服务优先级类的节点优先级降低 如果仲裁失败或发送出错,则重新向总线上发送直至成功。网络上其他节点从总线上读取该帧,分析它的服务优先级类、类中的优先级和节点优先级,若自身的节点优先级低于发送节点,则在相应的队列中向队头前移一位。

在 DPQ 机制下,总线先进行送紧急性更高即服务优先级更高的通信,然后才进行紧急性低的通信。DPQ 的行为类似于一个插入了不同节点的 FIFO 队列,但是,节点不一定要在队头才能进行信息的发送,只是在总线上发送的数据帧的服务优先级和类中优先级相同情况下,才应用节点优先级进行仲裁。

在 DPQ 中,当一个新节点加入网络并开始发送信息时,可能出现节点优先级冲突 ;当一个节点离开网络时,可能造成队列中出现间隙。设计适当的程序避免这些例外发生,可以证明在分布式队列中出现这些例外时,DPQ 机制不做任何修改仍能正确运作。

2. 优先级晋级机制

优先级晋级 PP (Priority Promotion) 机制也采用基本 CAN 协议中的扩展格式数据帧的结构实现,从而保持对传统 CAN 系统的兼容。它的基本思想是 :从节点首次尝试发送当前帧开始计时,随着计时长度的增加,相应增加该节点的优先级。

图 6.3 - 1 (c) 所示为优先级晋级帧的标识符结构。它也定义了四个服务优先级类。与 DPQ 中类似,各节点都在本地存储优先级级别,记为 L. PL,分别对应于不同的服务优先级类。节点在每次尝试发送失败时都对 L. PL 进行动态更新。当某一节点在同一服务优先级类上与其他的节点冲撞时,则通过减小该服务优先级类的 L. PL 的值来增加它的优先级 ;成功发送信息后,发送节点将优先级水平设备成最低。

在 PP 机制中,不要求不同节点具有不同的优先级级别,这与 DPQ 不同,因此在 PP 中不需要为节点加入或离开网络专门设计特殊的程序以防止发生例外。当节点启动时,仅需将自己的各个 L. PL 设为最低优先级,当多个具有相同优先级级别的节点同时发送信息,则依据有效标识符位 (Effective Identifier) 进行仲裁。在这种机制优先考虑的是服务等待的时间,以减少信息堵塞,保证系统通信的实时性。

3. 使用保留位 r1 和 r2 实现动态优先级的策略

仍采用基本 CAN 协议中的扩展格式数据帧的结构实现,将保留位 r1 和 r2 用于仲裁域,而不再用于控制域,从而支持动态优先级。这种策略也有两种方案

(1) r1 和 r2 在帧中的位置不变

这种方案中,数据帧的结构维持不变,利用 r1、r2 定义四类服务优先级,相当于将数据帧的标识符位增加到 31 位。r1、r2 不影响帧的基本优先级,只是在网络负载繁重时,解决具有相

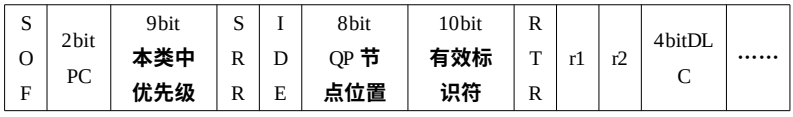
同基本优先级的数据帧的碰撞,能让实时性要求更高的信息优先发送,以提高实时响应。

(2) 将 r1 和 r2 前置

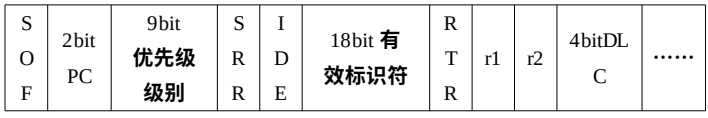
在该方案中,将保留位 r1 和 r2 移到仲裁域的最前面,r1、r2 起的作用比上一方案中更重要。同样在 r1、r2 定义四类紧急程度不同的服务优先级。新的 31 bit 的仲裁域定义了一个虚拟对象地址,由原有的对象地址和优先级域组成。并在每个节点上定义服务类与虚拟对象地址的映射,每个节点都能动态的改变自己的虚拟对象地址,但接收滤波器使用惟一的基地址。

这样提高了网络中紧急事件的实时性,并减少了网络的堵塞情况。

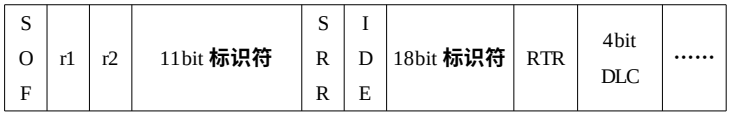
(a) CAN 扩展格式帧结构



(b) DPQ 的帧格式



(c) PP 的帧格式



(d) r1、r2 前置的帧格式

图 6.3-1 CAN、DPQ、PP、r1 和 r2 前置的帧格式

三、采用双单向通道的 FastCAN 延长网段长度

FastCAN 借鉴了 Interbus 系统与分布式队列双总线 (DQDB) 协议的某些思想,其网络拓扑如图 6.3-2 所示,包括两个单向物理通道——前向通道 FC 和逆向通道 RC。FC 用于传输信号并进行仲裁,RC 用于节点接收获胜的数据帧,并收集接收信息 (如差错信息等)。网络中的每个节点具有两个输入端口——前向接受 FRX 和逆向接收 RRX,以及两个输出端口——前向发送 FTX 和逆向发送 RTX。相邻的节点以两条单向连接方式相连。系统的两端分别是智能终端 SH 和哑终端 DH。它们的功能可分别由系统中每一个节点和最后一个节点完成:SH 用于进行确认并发送确认帧;DH 只是一个转发器,只需将最后一个节点的 FTX 输出联到 RRX 输入即可。

FastCAN 中的数据帧和确认帧的格式如图 6.3-3 所示。数据帧与传统 CAN 帧相似,只是增加了一个“忙域”。忙域有两位:“忙槽”和“忙定界符”,用于通知各节点是否仲裁获胜。确认帧仅由 SH 发送,有否定和肯定两种格式。

每次 FastCAN 信息交换都包括两种帧的传输。各节点首先在 FC 上发送数据帧,发生碰

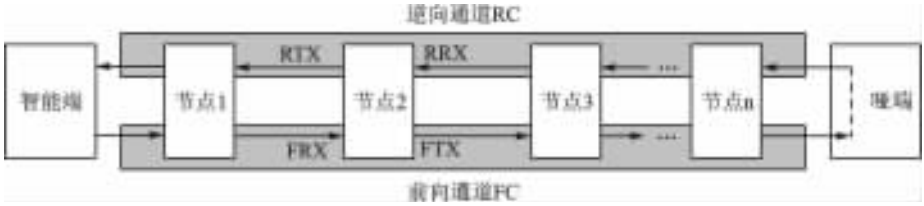


图 6.3－2 FastCAN 的网络拓扑

撞时被仲裁，最后到达 DH 并转发到 RC 上的数据帧为获胜帧，此时该帧未被任何节点读取；在 VC 上从 DH 到 SH 的途中，该数据帧被相应节点读取，并进行差错检验。若发送成功则在该数据帧返回 SH 后，由 SH 向 FC 上发送一个肯定的确认帧；否则发送一个否定的确认帧。

肯定帧起始位为一显性 SSF 位，随后是 8 位隐性确认定界符。当肯定帧在 FC 上传输时，网络中等待发送的节点在 FC 上检测到 SSF 位时，则立即发送新数据帧的仲裁域，这样认可帧转变成一个实际帧，此时 SSF 位与传统 SOF 位作用相同，如图 6.3－3 (b) 所示；若网络中没有任何数据帧需要发送，则转换到空闲状态，下一新数据帧发送时以 SOF 位打头，如图 6.3－3 (c) 所示。因此，FC 上的肯定帧和数据帧都是对前一次数据帧交换的肯定应答。在 RC 上接收到肯定帧时，所有节点都不必理睬它。

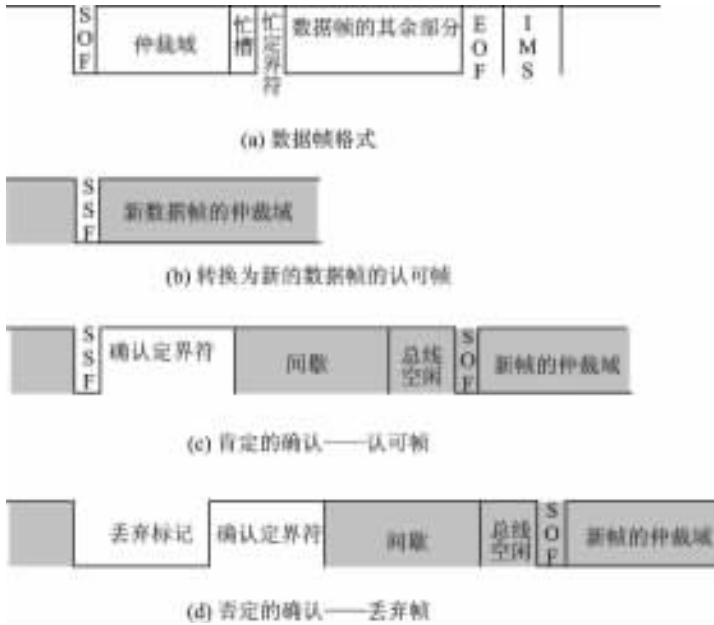


图 6.3－3 FastCAN 中数据帧和确认帧的格式

否定帧用于在差错发生后通知各节点丢弃出错的数据帧，并清除节点状态。如图 6.3－3 (c) 所示，否定帧以丢弃标记开头，由 18 个显位组成，接着是 8 个隐位组成的确认定界符。

FastCAN 由于采用了允许分布式仲裁的双通道结构，显著缓和了传统 CAN 协议的严厉定时限制。例如，在比特率相同情况下，FastCAN 网络的范围可达传统 CAN 网络的 10 倍以上；同理在网络范围相同时，FastCAN 允许的最大比特率至少是 CAN 的 10 倍。此外，FastCAN 继承了传统 CAN 的优势，诸如异步数据交换的快速响应，利用优先级动态控制信息调度，网络初

始组态及错误后重新组态不需复杂过程等,并具备与 CAN 的下向兼容性。

四、结束语

CAN 由于其较低的造价和适用于短帧传输的特性,在工业领域得到了广泛的应用,但随着系统对通信要求和网络范围的要求越来越高,CAN 无法完全满足。因此,有必要在传统 CAN 网络上进行改进,以提高它的性能,使之能更好地应用在工业领域中。

参 考 文 献

1 Gianluca Cena, Adriano Valenzano. FastCAN: A High-Performance Enhanced CAN-Like Network. IEEE Transactions on Industrial Electronics, 2000 (4)

2 S. Hasnaoui, A. Bouallegue. A Proposal Modification of CAN Protocol to Support a Dynamic Priority Policy Being Able to be Implemented on CAN Fieldbus Controller Components. Industry Application Conference 2000. Conference Record of the 2000 IEEE, 2000 (2): 1129 ~ 1136

3 Gianluca Cena, Adriano Valenzano. An Improved CAN Fieldbus for Industrial Applications. IEEE Transactions on Industrial Electronics, 1997 (4)

选自《工业控制计算机》月刊,2002 年第 11 期

6.4 快速和高可靠性的 CAN 网络模块 ADAM-500/CAN

武汉大学 黄天成 王海燕 任清珍 孙夫雄

CAN 最早是由欧洲一家汽车制造厂 BOSCH 开发的,用于汽车工业。CAN 用低成本的网络电缆代替了车身内昂贵的导线,具有快速响应时间和高可靠性,并适合对实时性要求较高的应用,如刹车装置和气囊。CAN 芯片现在可以抗高温和高噪声,并且具有较低的价格。CAN 作为广播式串行总线系统,先后成功地应用在汽车工业和楼宇自动化中。这些成功为它在自动化工业中取得同样的成功奠定了良好的基础。

CAN 总线采用多主工作方式,节点之间不分主从,有优先级之分。通信方式灵活,可实现点对点、一点对多点及广播方式传输数据,无须调度。采用非破坏性总线仲裁技术,优先级发送,可大大节省总线冲突仲裁时间,在重负荷下表现出良好的性能。CAN 采用短帧结构传输,每帧有效字节为 8 个,传输时间短,受干扰的概率低。每帧信息都有 CRC 校验和其他检错措施,保证数据出错率极低。当节点严重错误时,具有自动关闭功能,使总线上其他节点不受影响。可见,CAN 是所有总线中最为可靠的。

一、ADAM-5000/CAN 系统

1. ADAM-5000/CAN 系统简介

ADAM-5000/CAN 设备使用控制器局域网络 (CAN) 协议,是一开放、广播式的通信系统,主要特点如下:

⑴ 单个 CAN 网络可以支持 63 个 ADAM－5000 系统;

⑵ 可选数据传输速率,对于 DeviceNet 协议可选 125 Kbps、250 Kbps 和 500 Kbps;

⑶ 涌流和电压反相保护;

⑷ 具有宽电源范围,未经调理的 DC 电压 +10 ~ +30 V;

⑸ 启动时硬件自检;

⑹ 容易安装在 DIN 导轨或面板上;

⑺ 通过插入式螺钉接线端子块容易进行接线。

ADAM-5000/CAN 系统使用两种通信协议:Allen－Bradley 开发的 DeviceNet 或 CiA (CAN in Automation) 的 CANopen。目前应用较广泛的 DeviceNet 是一种低成本的通信链接协议,可以连接工业设备,如极限开关、电磁阀、光/电传感器、条形码阅读器、电机启动器、过程传感器、变频器、平板显示和人机界面;可以降低网络和复杂硬件连线的成本;直接连接可以改进设备之间的通信,并能完成不易进行或须连接硬件 I/O 接口的设备级诊断;当多个复杂设备间的互换可能时,还允许简单设备之间进行互换。DeviceNet 是一种开放的网络标准。其规格和协议是开放的,任何人都可以通过 ODVA (Open DeviceNet Vendor Association) 获得 DeviceNet 的规格。

2. CAN 网络系统实现

ADAM-5000 是一种基于模块的设备，整个系统作为一个从设备，将工业现场数据采集到 PC 机，并利用 PC 机进行控制。CAN 网络设计如图 6.4－1 所示。

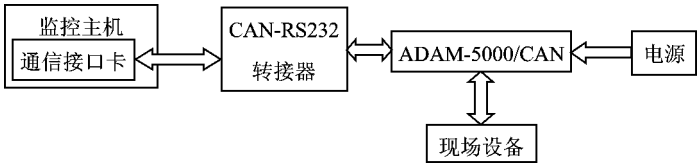


图 6.4－1 ADAM-5000 系统配置

通过通信接口卡接入 CAN 总线网络来监控主机，使操作人员可以在上位机上对位于现场的各个测量与控制模块进行设置、管理、监视与维护。在 ADAM-5000/CAN 中可以挂接 4 个 I/O 插槽，挂接在 CAN 总线网络上的各种 I/O 模块起到数据采集、监视与控制的作用。ADAM-5000/CAN 支持所有的工业 I/O 模块，包括数字量 I/O、模拟量 I/O、计数器和特殊用途的 I/O 模块，如热电偶和热电阻模块。此外，所有这些模块都可提供 DC 3000 V 隔离。

系统中位于现场的检测与控制部分，可以根据实际对象选用研华公司的 ADAM-5000/CAN 系列功能模块。这样，可以大大缩短系统的设计周期，使设计人员从反复设计常用输入/输出功能电路的繁重工作中解脱出来，避免重复劳动。只是对于某些比较特殊的应用场合，需要有特别的数据采集、信号处理或者控制输出电路，这就要求设计人员自行开发针对特殊对象的专用 I/O 模块了。一些专用的功能模块，只要配置有 CAN 总线的 I/O 接口，也可以挂接到 CAN 总线网络上来，与 ADAM-5000/CAN 系列功能模块一起实现分布式测控系统的现场测控任务。ADAM-5000/CAN 的功能模块如图 6.4－2 所示。

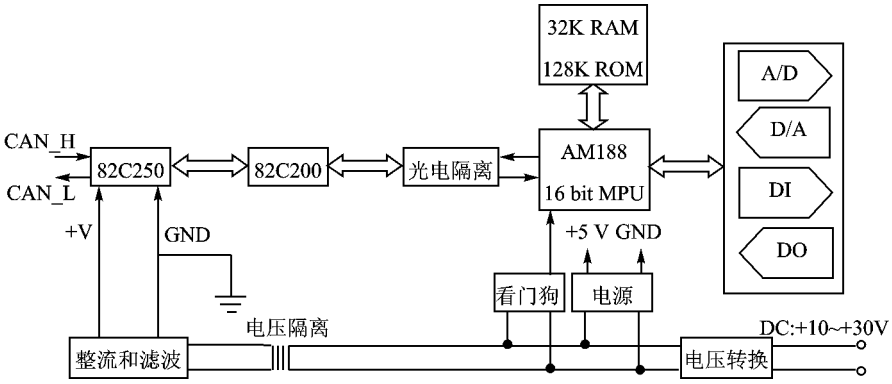


图 6.4－2 ADAM-5000/CAN 功能模块图

图 6.4－2 中 CAN 协议控制器采用 PHILIPS 公司生产的 PCA82C200，CAN 接口控制器采用 PHILIPS 公司生产的 PCA82C250 芯片。PCA82C250 可以提供对总线的差动发送能力和对 CAN 控制器的差动接收能力。

通信接口卡采用双端口隔离 CAN 接口卡 PCL-841，将其插入主机 ISA 插槽，其传输速度可达到 1 Mbps，有 4 个 LED 指示灯显示发送和接收的状态。用专用的总线接口装置 ADAM-4525 将 CAN 与主机相连，将计算机接入 CAN 总线网络，实现在主计算机上对位于现场的测

量、控制装置进行管理与维护。以此方式能够将计算机强大的功能应用于 CAN 总线系统及其现场设备,实现控制室与现场、软件与硬件的资源共享及功能互补。

3. 组态软件

在 ADAM-5000/CAN 的系统结构与功能特点的基础上,结合 CAN 总线的通信协议 DeviceNet,采用 ANSI C 语言编程。下面仅介绍在本系统中的一个输入/输出程序。

```
void demo() {
……                                  //变量的定义与初始化
send_buf = set_ch0_alarm;           //设置槽 1 的 0 通道越限值
Dn_SndMsg($ send_buf);
Dn_GetResp();
send_buf = get_alarm_stus;          //获取槽 1 的警戒值
canSndMsg(&send_buf);
GetResp();
if (receive_buf[6] &0x01) {         //如果槽 1 的 0 通道发生越限输入,槽 2 所有通道值置为 1
   send_buf = set_do;
   canSndMsg(&send_buf);
}
……                                  //变量的处理
}
```

程序功能如下 如果在槽 1 上,模拟量输入模块(AI)的通道 0 (ch0)中发生了越限输入,就使槽 2 上的模块中所有数字量的输出设定为 1。该程序实现了关于 CAN 总线的 DeviceNet 通信协议在 ADAM-5000/CAN 系统上的基本应用,可以由 ADAM-5000/CAN 系统所提供的一个下载工具 DNU,从主计算机上的 RS-232 串口下载到 ADAM-5000/CAN 的主单元中,并在其中运行。每个 ADAM-5000/CAN 主单元都设有一个 RS-232 接口用于接受主计算机对它的编程。

DNU 主要是针对研华公司的硬件产品开发的,因此,可与研华 ADAM 模块很好地契合。功能主要有:① 系统和模块设置;② 模块校准;③ 数据输入和输出;④ 查找已连接系统;⑤ 事件监控;⑥ 协议固件下载;⑦ 终端仿真。可以根据实际需要编写或更改程序,满足所需的功能。

二、ADAM-5000/CAN 系统优势

我们在应用中发现该系统具有以下优点。

1. 实时能力

CAN 系统通过使用非破坏性的位总线仲裁技术来处理多于一个的节点,同时访问网络的冲突,从而满足实时性的需要。使用这种方法,不会丢失任何数据或损失带宽。CAN 的仲裁机制处理冲突并决出一个“优胜者”,给它最高优先级并立即发送数据。系统反应时间不仅依赖于站的多少,更依赖于优先级的级别。

2. 减少系统维护和故障处理

(1) 硬件自检和软件诊断

ADAM-5000 系统有两种诊断方式:硬件自检和软件诊断。它们可以帮助用户检查和鉴别各种系统或 I/O 模块的故障。

(2) 看门狗定时器管理

看门狗定时器会对微处理器进行监控并能自动复位系统。这种设计减少了现场的维护工作。

3. 适合工业环境

(1) 3 端隔离

ADAM-5000 系统提供了 I/O 模块隔离、电源隔离和通信隔离。3 端隔离设计避免了接地环路,并减少了电磁干扰对系统的影响,提供电涌保护以避免电压尖峰对系统造成破坏。

(2) 调理电源反相保护

ADAM-5000 系统可以接受未调理的 DC 电压 +10 ~ +30 V,可以在电源意外反相后对系统进行保护。

(3) 宽温工作范围

ADAM-5000 模块的工作温度范围可以从 -10 ~ +70 。

4. 通信的可靠性

ADAM-5000/CAN 的一个突出特点是它的高传输可靠性。CAN 总线最早用于汽车发动机的控制。汽车的强电磁干扰(如火花塞放电)需要通信系统具有高度的可靠性。CAN 具有 5 种通信校验机制和快速的故障恢复时间。故障发生后,消息在 29 位时间内重复,汉明距离为 HD=6(能辨认任何位置到 5 位的误差)。没有其他现场总线具有如此高的可靠性,它足以满足对工业现场数据准确性的要求。

本文以台湾研华公司的 ADAM-5000/CAN 系统作为平台,给出了一个包括主计算机在内的 CAN 总线测控系统集成方案。它能实现工业现场数据的快速高可靠性采集和传输,现场应用效果较好,是一套较实用的工业现场总线系统。

参 考 文 献

1 邬宽明. CAN 总线原理和应用系统设计. 北京:北京航空航天大学出版社,1996

2 Advantech. ADAM5000/CAN User's Manual. 1997

3 阳宪惠主编. 现场总线技术及其应用. 北京:清华大学出版社,1999

选自《单片机与嵌入式系统应用》月刊,2002 年第 6 期

6.5 SJA1000 在 CAN 总线系统节点的应用

北京理工大学 岑雪松 朱 丹

CAN 总线是德国 Bosch 公司 20 世纪 80 年代初,为解决现代汽车中众多的控制与测试仪器之间的数据交换而开发的一种串行数据通信协议。1993 年 11 月,ISO 正式颁布了道路交通运载工具,进行数据信息交换用的高速通信控制器局部网(CAN)的国际标准(ISO11898)。PHILIPS、Intel、MOTOROLA 等公司出品了很多支持 CAN 协议的集成芯片,如 82526、SJA1000、68HC05X4/X16/X32 和具有片内 CAN 的电磁兼容微控制器 P8XCE598、16 位微控制器 87C196CA/CB 等。下面介绍 PHILIPS 半导体公司推出的 CAN 总线控制器 SJA1000,并给出其应用实例。

一、CAN 总线控制器 SJA1000 芯片介绍

SJA1000 是一种独立的 CAN 总线控制器。PHILIPS 半导体公司将它作为 PCA82C200 CAN 控制器(Basic CAN)的替代产品。SJA1000 增加了一种新的工作模式(Peli CAN),这种模式支持具有很多新特性的 CAN 2.0B 协议。

1. SJA1000 引脚介绍

图 6.5-1 是 SJA1000 引脚图。SJA1000 具有 28 个引脚,下面对部分引脚进行介绍。

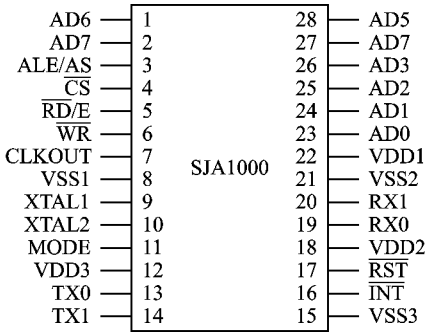


图 6.5-1 SJA1000 引脚图

MODE:模式选择输入,1(高电平)= Intel 模式;0(低电平)= Motorola 模式。

TX0、TX1:从 CAN 输出驱动器 0,1 输出到物理总线上。

RX0、RX1:从物理 CAN 总线输入到 SJA1000 的输入比较器。

$\overline{INT}$:中断输出,用于中断微控制器。$\overline{INT}$在内部中断寄存器各位都置位时,低电平有效。$\overline{INT}$是开漏输出。

CLKOUT:SJA1000 产生的提供给微控制器的时钟输出信号,时钟信号来源于内部振荡器且通过编程驱动,时钟控制寄存器的时钟关闭位可禁止该引脚。

SJA1000 的其他引脚分别为:AD0 ~ AD7,数据/地址复用总线;ALE/AS,Intel 模式/Motorola 模式的地址锁存信号;($\overline{RD}$)/E、$\overline{WR}$,读写控制信号;$\overline{CS}$,片选信号输入,低电平有效;XTAL1,输入到振荡器放大电路,外部振荡信号由此输入;XTAL2,振荡器放大电路的输出,使用外部振荡信号时左开路输出;VDD1、VDD2、VDD3,5V 电压端;VSS1、VSS2、VSS3,与上述电压端相对的接地端。SJA1000 有两种封装形式,分别是塑质双列直插封装和塑质小型线外封装。

2. SJA1000 芯片功能介绍

SJA1000 与它的前一款——PCA82C200 独立控制器是兼容的。SJA1000 具有很多新的功能,修改了两种模式:Basic CAN 模式、PCA82C200 兼容模式;增加了 Peli CAN 模式,此模式支持 CAN 2.0B 协议规定的所有功能(29 字节的识别码)。

SJA1000 的主要新功能:

- 标准结构和扩展结构信息的接收和发送;
- 具有 64 字节长度的接收队列;
- 在标准和扩展格式中,都有单/双接收过滤器(含屏蔽和代码寄存器);
- 读/写访问的错误计数器,可编程的错误限制报警,最近一次的误码寄存器;
- 每一个 CAN 总线错误的错误中断;
- 由功能位定义的仲裁丢失中断;
- 一次性发送(当错误或仲裁丢失时不重发);
- 只听模式(CAN 总线监听,无应答,无错误标志);
- 支持热插拔(无干扰软件驱动位速检测);
- 硬件禁止 CLKOUT 输出。

下面只介绍 Basic CAN 模式,对于 Peli CAN 模式请查看参考文献[1]。

SJA1000 复位,默认为 Basic CAN 模式,或者通过时钟分频寄存器的 CAN 模式位来选择模式。当此位清零时,为 Basic CAN 模式;置位时,为 Peli CAN 模式。在 Basic CAN 模式下,对 SJA1000 进行控制以及收发数据,都是通过对 SJA1000 内部寄存器的读/写访问来实现的。对于单片机而言,操作 SJA1000 就像访问外部 RAM 一样简单。有两种模式可以对 SJA1000 的内部寄存器访问,而在这两种模式下对其寄存器的访问是有区别的。这两种模式分别是复位模式和工作模式。当硬件复位,或控制器掉线,或置位复位请求位时,SJA1000 进入复位模式;而当清除复位请求位时,SJA1000 进入工作模式。SJA1000 的寄存器分布于 0~31 连续的地址空间中。这 32 个字节可分为控制段(10 字节)、发送缓冲器段(10 字节)、接收缓冲器段(10 字节)、时钟分频器和 1 个无效字节。在复位模式下可写的寄存器为控制段的控制寄存器、命令寄存器、接收代码寄存器、屏蔽寄存器、总线时序 0、总线时序 1、输出控制寄存器,还包括接收缓冲器段和时钟分频器;而在工作模式下可写的寄存器为控制寄存器、命令寄存器、发送缓冲器段、接收缓冲器段和时钟分频器。

下面根据应用需要具体介绍 SJA1000 的控制寄存器、命令寄存器、状态寄存器、中断寄存器。

(1) 控制寄存器(CR)

控制寄存器位于 SJA1000 寄存器区的 0 地址,用来设置 CAN 总线的模式和各种中断。其各位的意义如表 6.5-1 所列。

在硬启动或总线状态位设置为 1(总线关闭)时,复位请求位被置为 1。在外部复位期间,微控制器不能把复位请求位置为 0。如果要把复位请求位置为 0,微控制器必须先检查这一位,以确定外部复位引脚不为低电平。复位请求位被设为 0 后,SJA1000 将会等待:① 1 个总线空闲信号(11 个弱势位),如果前一次复位请求是硬件复位或 CPU 初始复位;② 等待 128 个总线空闲,如果前一次复位请求是 CAN 控制器在重新进入总线开启模式前初始化总线造成的。

表 6.5－1 控制寄存器各位的功能说明(CR):CAN 地址 0

位	符号	名称	值	功 能
CR.7	—	—	—	保留
CR.6	—	—	—	保留
CR.5	—	—	—	保留
CR.4	OIE	溢出中断使能	1	使能。如果置位数据溢出位,微控制器接收溢出中断信号
			0	禁止。微控制器不从 SJA1000 接收溢出中断信号
CR.3	EIE	错误中断使能	1	使能。若出错或总线状态改变,此中断信号有效
			0	禁止。微控制器不从 SJA1000 接收错误中断信号
CR.2	TIE	发送中断使能	1	使能。当信息被成功发送或发送缓冲器又被访问时,产生中断信号
			0	禁止。微控制器不从 SJA1000 接收发送中断信号
CR.1	RIE	接收中断使能	1	使能。信息被无错误接收时,产生此中断信号
			0	禁止。此中断信号被禁止
CR.0	RR	复位请求	1	当前。SJA1000 检测到复位请求后,忽略当前发送/接收信息,进入复位模式
			0	空缺。复位请求位接收到一个下降沿后,SJA1000 回到工作模式

(2) 命令寄存器(CMR)

命令寄存器对微控制器来说是只写存储器。在复位模式和工作模式下都可对此寄存器进行访问,但是读这个地址返回值是“11111111”。表 6.5－2 是命令寄存器各位的功能说明。将睡眠模式位置为 1,SJA1000 进入睡眠模式,此时没有总线活动,没有中断等待。CMR.3 位是用来清除由数据溢出状态位指出的数据溢出。如果数据溢出位被置位,就不会产生数据溢出中断了。在释放接收缓冲器命令的同时,可以发出清除数据溢出命令。读接收缓冲器之后,微控制器可以通过设置释放接收缓冲器为 1,来释放接收队列当前信息的内存空间。

表 6.5－2 命令寄存器各位的功能说明(CMR):CAN 地址 1

位	符号	名称	值	功 能
CMR.7	—	—	—	保留
CMR.6	—	—	—	保留
CMR.5	—	—	—	保留
CMR.4	GTS	睡眠	1	睡眠。若没有 CAN 中断等待和总线活动,SJA1000 进入睡眠模式
			0	唤醒。SJA1000 正常工作模式
CMR.3	CDO	清除数据溢出	1	清除。清除数据溢出状态位
			0	无动作
CMR.2	RRB	释放接收缓冲器	1	释放。接收缓冲器存放信息的内存空间将被释放
			0	无动作
CMR.1	AT	忽略发送	1	当前。若不是在处理过程中,等待处理的发送请求将取消
			0	空缺。无动作
CMR.0	TR	发送请求	1	当前。信息被发送
			0	空缺。无动作

(3) 状态寄存器 (SR)

状态寄存器对微控制器来说是只读存储器,表 6.5-3 是状态寄存器各位功能的说明。当传输错误计数器超过限制 (255) (总线状态位置 1,即总线关闭),CAN 控制器就会将复位请求位置 1,在错误中断允许的情况下,会产生一个错误中断。这种状态会持续到 CPU 清除复位请求位。对于错误状态位,当至少有一个错误计数器满或超出 CPU 警告限制 (96) 时,错误状态位被置位。在中断使能的情况下,会产生错误中断。

表 6.5-3 状态寄存器各位的功能说明 (SR):CAN 地址 2

位	符号	名称	值	功 能
SR.7	BS	总线状态	1	总线关闭。SJA1000 退出总线活动
			0	总线开启。SJA1000 加入总线活动
SR.6	ES	出错状态	1	出错。至少出现一个错误计数器满或超过 CPU 报警限制
			0	正常。两个错误计数器都在报警限制以下
SR.5	TS	发送状态	1	发送。SJA1000 在传送信息
			0	空闲。没有要发送的信息
SR.4	RS	接收状态	1	接收。SJA1000 正在接收信息
			0	空闲。没有可接收的信息
SR.3	TCS	发送完毕状态	1	完毕。最近一次发送请求被成功处理
			0	未完毕。当前发送请求未处理完毕
SR.2	TBS	发送缓冲器状态	1	释放。CPU 可以向发送缓冲器写信息
			0	锁定。CPU 不能访问发送缓冲器,有信息正在等待发送或正在发送
SR.1	DOS	数据溢出状态	1	溢出。信息丢失,因为 RXFIFO 中没有足够的空间来存储
			0	空缺。自从最后一次清除数据溢出命令执行,无数据溢出发生
SR.0	RBS	接收缓冲器状态	1	满。RXFIFO 中有可用信息
			0	空。无可用信息

(4) 中断寄存器 (IR)

通过中断寄存器可识别中断源。当寄存器的 1 位或多位被置位时,$\overline{\text{INT}}$ (低电平有效) 引脚被激活。寄存器被微控制器读过之后,所有会导致$\overline{\text{INT}}$引脚上的电平变化的位被复位。中断寄存器对微控制器而言是只读存储器。中断寄存器各位的功能说明如表 6.5-4 所列。

表 6.5-4 中断寄存器各位的功能说明 (IR):CAN 地址 3

位	符号	名称	值	功 能
IR.7	—	—	—	保留
IR.6	—	—	—	保留
IR.5	—	—	—	保留
IR.4	WUI	唤醒中断	1	置位。退出睡眠模式时此位被置位
			0	复位。微控制器的任何读访问将清除此位

续表 6.5－4

位	符号	名称	值	功 能
IR.3	DOI	数据溢出中断	1	设置。当数据溢出中断使能位被置为 1 时,向数据溢出状态位传送“0－1”,此位被置位
			0	复位。微控制器的任何读访问清除此位
IR.2	EI	错误中断	1	置位。错误中断使能时,错误状态位或总线状态位的变化会置位此位
			0	复位。微控制器的任何读访问清除此位
IR.1	TI	发送中断	1	置位。发送缓冲器状态从 0 变为 1 (释放) 和发送中断使能时,置位此位
			0	复位。微控制器的任何读访问清除此位
IR.0	RI	接收中断	1	置位。当接收 FIFO 不空和接收中断使能时置位此位
			0	复位。微控制器的任何读访问清除此位

二、SJA1000 在节点中的应用实例

该节点的微控制器选用了 8 位单片机 AT89C51,SJA1000 作为 CAN 总线控制器,并且使用了 CAN 接口芯片 82C250。此节点可直接运用到 CAN 总线网络系统中,或者对此节点电路稍加变动来满足设计的要求。下面从硬件电路和软件设计两部分来介绍。

1. 节点硬件电路设计

图 6.5－2 是节点的电路原理图。注意 SJA1000 复位端的连接,AT89C51 是高电平复位,而 SJA1000 是低电平复位,因此复位信号要通过一个反相器与 SJA1000 的复位端相连。另外 SJA1000 的 11 脚 MODE 接高电平,选择 Intel 二分频模式。SJA1000 的 16 脚是中断信号输出端,在中断允许情况下,有中断发生时,16 脚出现由高电平到低电平的跳变,因此 16 脚可以直接与 AT89C51 的外部中断输入脚相连接。该设计中之所以选择 82C250 芯片,是因为其具有高速性 (最高可达 1 Mbps),具有抗瞬间干扰保护总线的能力,具有降低射频干扰的斜率控制。此外,它可以与 110 个节点相连,防止电池与地之间发生短路,当某一个节点掉电时,不会影响总线。在设计节点电路时,还要注意下面几点:

(1) SJA1000 通过光耦与 82C250 的连接是电流隔离的接法,这样可以防止线路间的串扰。在总线两端要接 2 个 120Ω 的总线阻抗匹配电阻。忽略掉它们会降低总线的抗干扰能力,甚至导致无法通信。

(2) 通过在地和 82C250 的 8 脚 (RS) 之间接不同阻值的电阻,可选择三种不同的工作方式:高速、斜率控制和待机,如表 6.5－5 所列。

表 6.5－5 RS 选择的三种工作方式

R8 提供条件	方式	R8 上的电压或电流
$V_{RS} > 0.75V_{CC}$	待机方式	$I_{RS} < \lvert 10\ \mu A \rvert$
$10\ \mu A < -I_{RS} < 200\ \mu A$	斜率方式	$0.4V_{CC} < V_{RS} < 0.6V_{CC}$
$V_{RS} < 0.3V_{CC}$	高速方式	$-I_{RS} < 500\ \mu A$

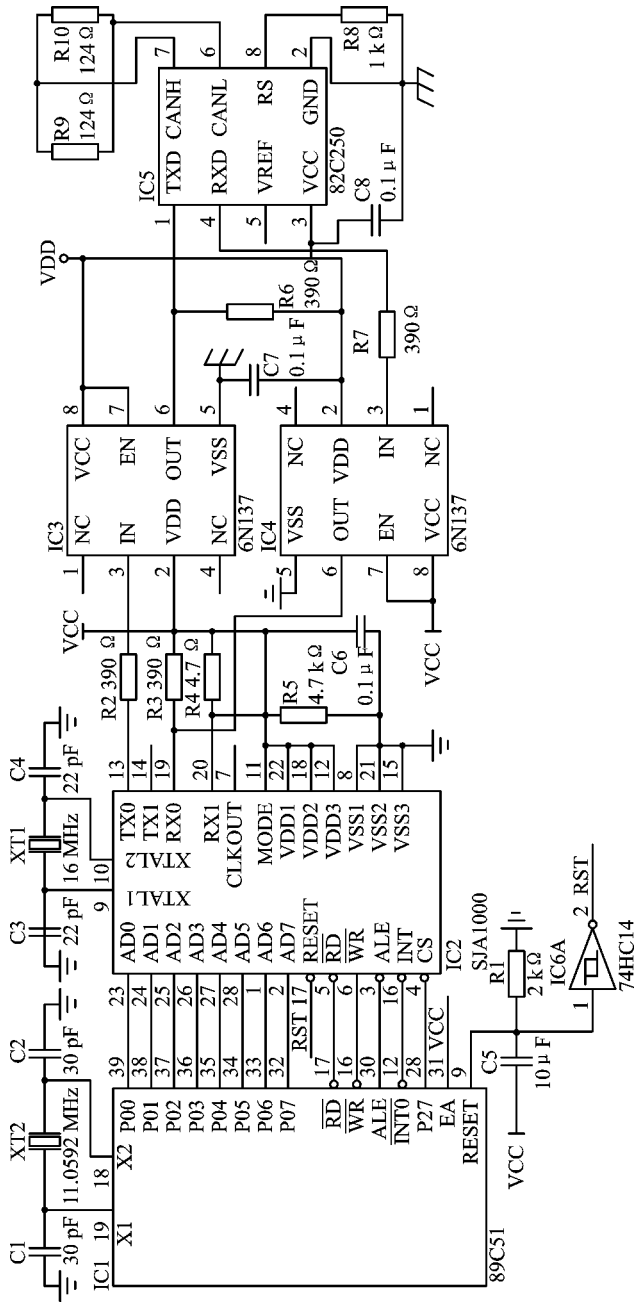


图 6.5－2　节点电路原理图

在高速工作方式下,发送器输出晶体管简单地以尽可能快的速度启闭。在这种方式下,不采取任何措施限制上升和下降斜率。建议使用屏蔽电缆以避免射频干扰问题。通过将引脚 8 接地,可选择高速方式。

对于较低速度或较短总线长度,可用非屏蔽双绞线或平行线作总线。为降低射频干扰,应限制上升和下降斜率。上升和下降斜率可通过由引脚 8 至地连接的电阻进行控制。斜率正比于引脚 8 上的电流输出。

若引脚 8 加有高电平,则电路进入低电流待机方式。在这种方式下,发送器被关掉,而接收器转至低电流。由于在待机方式下,接收器是慢速的,因此,第一个报文将被丢失。

(3) SJA1000 的 TX1 脚悬空,RX1 引脚的电位必须维持在约 0.5 V_{CC}上,否则,将不能形成 CAN 协议所要求的电平逻辑。

2. 节点软件设计

根据节点电路原理图,SJA1000 的首地址为 0000H。用 MCS-51 汇编语言编制的初始化程序如下:

```
CR      EQU     0000H           控制寄存器
CMR     EQU     0001H           命令寄存器
SR      EQU     0002H           状态寄存器
IR      EQU     0003H           中断寄存器
ACR     EQU     0004H           接收代码寄存器
AMR     EQU     0005H           接收屏蔽寄存器
BTR0    EQU     0006H           总线时序寄存器 0
BTR1    EQU     0007H           总线时序寄存器 1
OCR     EQU     0008H           输出控制寄存器

MOV     DPTR,#CR
MOV     A,#1BH                  开放接收、出错、溢出中断
MOVX    @DPTR,A                 置位复位请求,开始初始化
MOV     DPTR,#ACR
MOV     A,#03H                  接收代码寄存器为 03H
MOVX    @DPTR,A
INC     DPTR
MOV     A,#0FCH                 接收屏蔽寄存器为 0FCH
MOVX    @DPTR,A
INC     DPTR
MOV     A,#00H
MOVX    @DPTR,A
INC     DPTR
MOV     A,#1CH                  ;500 Kbps
MOVX    @DPTR,A
INC     DPTR
MOV     A,#0AAH                 正常输出模式
MOVX    @DPTR,A
```

```
MOV   DPTR,#CR
MOV   A,#1AH                    初始化结束,SJA1000 进入工作状态
MOVX  @DPTR,A
RET
```

注意:SJA1000 初始化程序中,只有当控制寄存器 CR 中的复位请求为 1 时(SJA1000 工作在复位模式),允许访问上述寄存器,否则既写不进去,也读不出正确的内容。在接收寄存器(RXB)为空,满足下述条件时,报文可被正确地接收。

报文接收条件:接收代码位(AC.7 ~ AC.0)和信息识别码的高 8 位(ID.10 ~ ID.3)相等,且与接收屏蔽位(AM7 ~ AM.0)的相应位相或为 1,则报文被接收。例如:在初始化程序中,ACR = 03H,AMR = 0FCH,由此只有信息识别码的高 8 位为 XXXXXX11 的数据帧被接收。BTR0 的值可决定波特率预分频器(BRP)和同步跳转宽度(SJW)的数值;BTR1 可决定位周期的宽度,采样点的位置及在每个采样点进行采样的次数。程序中 BTR0 = 00H,BTR1 = 1CH,则波特率为 500 Kbps。有一点必须要注意,系统中所有节点的 BTR0 和 BTR1 的内容都应该相同,否则将无法进行通信。对 CR 的第二次写访问是清除复位请求位,使 SJA1000 返回工作模式。

参 考 文 献

1 SJA1000 Stand - alone CAN controller. PHILIPS Data Sheet

2 邬宽明. CAN 总线原理和应用系统设计. 北京:北京航空航天大学出版社,1996

3 肖海荣,周风余. 基于 SJA1000 的 CAN 总线系统智能节点设计. 计算机自动测量与控制,2001,9(2)

选自《单片机与嵌入式系统应用》月刊,2002 年第 3 期

6.6 用 C167CR 实现 CAN 总线通信

河北廊坊华北航天工业学院 (065000) 李国洪 胡 辉 王喜斌
河北石油职业技术学院 (065000) 刘 静

现场总线是 20 世纪 90 年代兴起的一种先进的工业测控技术,在现场仪表智能化及全数字化测控系统中得到广泛应用。由于 CAN 总线具有通信速率高、可靠性高、连接方便及性能价格比高等特点,在众多现场总线中占有较大市场份额,推动了其应用开发的快速发展;反过来,又促进生产厂商不断推出新的 CAN 总线产品,并逐渐形成系列,如 Philips、Intel、Infineon 等 CAN 控制器及片内集成 CAN 控制器的系列单片机。

在 Infineon (SIEMENS) 16 位系列单片机 C167CR 中都集成了 CAN 总线通信控制器。它可以支持高速串行通信协议 CAN2.0B,即支持标准 (11 位 ID) 和扩展 (29 位 ID) 的通信协议。本文介绍用 C167CR-LM 实现 CAN 总线通信的具体方法。

一、CAN 控制器基本结构

在 C167CR 中有 256 Byte 的 RAM 空间用于 CAN 总线串行通信,地址范围为在 Seg0 中的 EF00 ~ EFF0,所有寄存器均为 16 位,但它们均可进行字节寻址以便选择特殊功能。内存地址分布如图 6.6 - 1 所示。CAN 控制器由通用功能寄存器和信息体两部分组成,其中通用功能寄存器由控制/状态寄存器、中断寄存器、位定时寄存器、标准和扩展中断屏蔽寄存器组成。

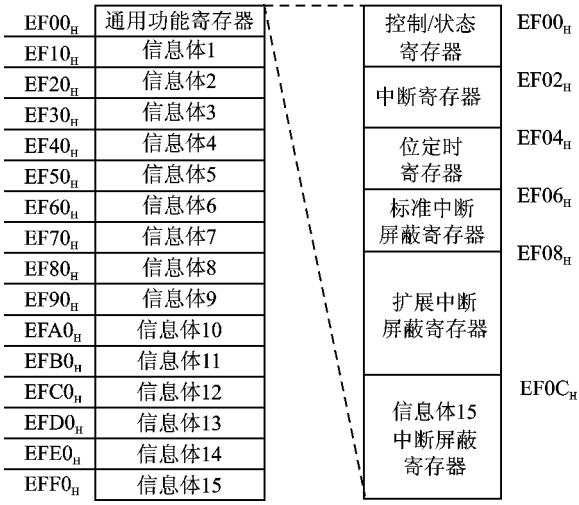


图 6.6 - 1 内存地址分布

1. 信息体 ($EF10_H \sim EFF0_H$)

信息体是 CPU 与 CAN 控制器之间通信的基本载体,主要用于发送和接收数据的存储器,

共有 15 个信息。每一个信息体由连续的 15 Byte 组成,其地址结构如表 6.6－1 所列。它由信息体控制寄存器、识别寄存器、配置寄存器和数据位场组成。信息体 0～14 既可用于发送,也可用于接收信息体,信息体 15 仅可用于接收信息体。

表 6.6－1 信息体结构

(地址) H	内容描述
EFx0～EFx1	信息控制寄存器 MCR (Message Control Register)
EFx2～EFx5	信息识别寄存器 UAR,LAR (Upper/Lower Arbitration Register)
EFx6	信息配置寄存器 MCFR (Message Configuration Register)
EFx7～EFxE	数据位 0～7

注:x 表示信息体数 (用十六进制表示)

(1) 控制寄存器 MCR:用于定义发送接收中断、中断悬挂、信息体有效、发送请求、远程请求悬挂、数据覆盖等功能。

(2) 识别寄存器 UAR,LAR:由 4B 组成,用于定义辨识代码组合,这样就可以辨识出不同的信息 (由 MCFR 定义) 进行接收与发送。

(3) 配置寄存器 MCFR:用于定义传送或接收数据的长度,有效值为 0～8。它指信息体本身所含有的数据,同时它还定义方向,即接收还是发送、定义标准模式还是扩展模式。

(4) 数据位:用于存放接收和发送的数据,最多只能存放 8 个数据,所发送的数据必须与在 MCFR 中 (DLC) 所定义的数据长度相同。

2. 定时器 ($EF04S_H$ BTR:复位值 $UUUUS_H$)

位定时器主要用于定义 CAN 总线通信的速率,对同一个网络总线上各个节点应定义同一种通信速率,否则无法进行通信联系。CAN 控制器的总线工作频率可由下式计算:

$$F_{CPU} = F_{CPU} / \{2 (BRP + 1) [1 + (TSEG1 + 1) + (TSEG2 + 1)]\}$$

式中:F_{CPU}为系统时钟频率;BRP 为 CAN 波特率预分频值 (BRP + 1),取值范围 0～63;TSEG1 为采样点之前时间段域中的值,取值范围 2～15;TSEG2 为采样点之后时间段域中的值,取值范围 1～7;例如当定义 1 个 125 kb/s 的通信频率时,则可以设置:BTR = 0X4944

此时,TSEG2 = 4、TSEG1 = 9、BRP = 4 及 80% 的采样点和 1 个系统时钟周期的同步跳转宽度 SJW。当系统时钟为 20 MHz 时,CAN 通信频率由上式计算得到 125 kb/s。

3. 屏蔽寄存器

信息体 ID 可由 MCFR 中的 XTD 位定义为标准 (11 位 ID) 或扩展 (29 位 ID) CAN 格式,因此屏蔽寄存器分为标准中断屏蔽寄存器 (GMS:Global Mask Short 用于 11 位 ID)、扩展中断屏蔽寄存器 (UGML/LGML:Upper/Lower GLobal Mask Long 用于 29 位 ID),而信息体 15 有自己独立的中断屏蔽寄存器。三者功能相同,与信息体的辨识寄存器功能相类似。以扩展中断屏蔽寄存器 (UGML/LGML) 为例,其共 4Byte、0～28 位,某位置 1 则要求接收到的辨识码 ID 要与它本身的辨识码一致,如果某位置 0 则屏蔽了 ID 位,即它可以接收到这一位为 0 或 1 的信息体的辨识码,这就形成一种信息体的滤波方法,即接收信息体。通过扩展屏蔽寄存器的定义,即可以接收与它本身辨识码相同的发送信息体,也可以接收与它本身辨识码不同的发送信息体。如果信息体辨识码与发送信息体辨识码匹配,信息体也可以接收远程信息 (即要求数据传送),远程信息体辨识码存储于发送信息体中,覆盖任何屏蔽位。

4. CAN 中断寄存器 ($EF00_H$ IR：复位值——XX_H)

CAN 中断寄存器有 16 个中断源，用于记录悬挂 CAN 中断情况。当 CAN 发生中断时，由 CAN 中断寄存器低八位的 INTID 数值来判断是什么中断 (高 8 位保留)，INTID 数值越小其中断优先级越高。每个信息体 N 都有 1 个中断源 (INTID = 2 + N，N = 1，…，14)，CAN 状态寄存器变化 (SR：CSR 的高 8 位) 的中断优先级最高 (INTID = 01)，其次是信息体 15 (INTID = 02)。

5. CAN 控制/状态寄存器 ($EF00_H$ CSR：复位值 $XX01_H$)

CAN 控制/状态寄存器 CSR 分为两部分。低 8 位为控制寄存器 CR，用于控制位定时器的写允许、使能中断和对 CAN 总线的接入。高 8 位为状态寄存器 SR，用于标识 CAN 控制器的各种状态。其各位的意义如图 6.6－2 所示。

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BOFF	EWRN	—	RXOK	TXOK	LEC	0	CCE	0	0	EIE	SIE	OE	INIT		

INIT——软件初始化使能，置位时则隔离 CAN 总线系统、停止系统的接收发送；IE——中断使能，它包括所有的中断：错误状态变化、接收和发送中断；SIE——状态变化中断使能，用于接收成功中断、发送成功中断和错误变化中断的使能和非使能；EIE——错误中断使能，用于切断总线并发警告中断；CCE——配置变化使能，用于写位定时器；1——允许写位定时器；0——禁止写位定时器。

图 6.6－2 CSR 控制/状态寄存器

如果 CAN 控制器状态变化产生中断，软件可以读 CAN 的状态寄存器 (SR)，以决定哪个中断读请求源。如切断 CAN 总线状态 (BOFF)、警告状态 (EWRN)、接收成功 (RXOK)、发送成功 (TXOK)、错误码 (LEC)。

由此可见，CAN 中断系统是由多级中断构成的。它只有一个中断输出，并和其他片内中断源一样连接到标准的中断节点上。它由 CAN 中断控制寄存器 XP0IC 定义 CAN 控制器中断，这是由多个中断源产生的，通过 CAN 中断寄存器 IR 可以寻找出哪个中断源发出的中断；CAN 状态寄存器 SR 又是一个多中断，通过进一步判断状态寄存器 SR，又可以进一步判断哪一个中断源发出的中断。

当定义完信息体、通信的速率、中断控制、屏蔽控制时，即完成 CAN 总线通信初始化，将要传送到总线上的数据写入信息体的数据区就可以。CAN 的实际通信是由硬件来自动实施的，硬件是按信息帧格式 (主要有数据信息帧和远程信息帧) 来传递数据的。总线传送数据按信息帧来传送，帧与帧之间由 3 个分离位 (为 recessive 隐性 1) 来分离，没有信息帧时，总线则处于空闲时间。数据信息帧的格式如图 6.6－3 所示 (扩展模式)，由下列各域组成。

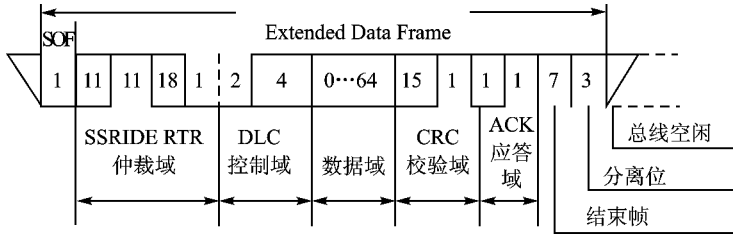


图 6.6－3 扩展数据帧格式

SOF：信息帧起始，低电平则标志信息帧开始。IDE：辨识器扩展位，始终为隐位 1，若为标准 CAN，此位始终为显位 0。

(1) 仲裁域：由 29 位信息辨识位场、SRR、IDE、RTR 组成。29 位辨识位场：由 11 位的标准辨识位场和 18 位扩展辨识位场组成，以保证与标准 CAN 协议的兼容。SRR 替代远程发送请求位，始终为隐位 1，是为兼容标准 CAN 协议而设置。RTR 远程发送请求位，用于区分数据帧 (始终为显性位 dominant = 0) 还是远程帧 (始终为隐位 1)。

(2) 控制域：由保留位 r0、r1 (始终为显位 0) 和 DLC 位组成。DLC 则是一个由 4 位表示的信息体数据长度码，传送数据长度为 0 ~ 8 字节。

(3) 数据域：由 1 ~ 8 个数据字节帧构成，0 字节则表示 1 个远程帧。

(4) 校验域：即 CRC 码，由循环冗余码求得的帧检查序列组成。它由一个 15 位 CRC 码加上一个界定符组成。

(5) 应答域：由一个节点接收帧发送的显位 0 和一个应答界定符隐位 1 构成。

(6) 结束帧：为 7 个连续的隐位 1 标志信息帧的结束。

信息帧传送是按信息体的仲裁域 ID 优先级大小来发送的，ID 数越小其优先级越高。

二、CAN 通信编程实现

CAN 通信编程一般由 CAN 节点初始化、定义信息体、装载数据、接收/发送、验证及检查等过程 (子程序) 组成。下面通过一个发送 8 Byte 数据的部分程序实例来说明 CAN 通信的软件实现。

```
/ * CAN control register definitions and Interrupt Set * /
#include < REG167. H >        //Register definitions C167
#define  CR   * (unsigned char * ) 0xef00
#define  SR   * (unsigned char * ) 0xef01
#define  IR   * (unsigned char * ) 0xef02
#define  BTR   * (unsigned int * ) 0xef04
#define  GMS   * (unsigned int * ) 0xef06
#define  UGML   * (unsigned int * ) 0xef08
#define  LGML   * (unsigned int * ) 0xef0a
#define  UMLM   * (unsigned int * ) 0xef0c
#define  LMLM   * (unsigned int * ) 0xef0e
IEN = 1;              //General Interrupt Enable = yes
XP0IC = 0x00;         //Interrupts from CAN mod. (CPU side) = no
/ * Initialization of CAN-Module: (baud rate, eie, sie, ie) * /
SR = 0x00;            //Clear Status Partition (EF01h) in CSR
CR = 0x41;            //Set CCE and INIT in Control Register
(EF00h)
BTR = 0x4944;         // = 125kbit/s (@F_CPU = 20MHz)
GMS = 0xe0ff;         //Global Mask Short, each bit of standard ID must match to store mess
UGML = 0xffff;        //Upper Global Mask Long
LGML = 0xf8ff;        //each bit of extended ID must match to store mess
```

```
UMLM = 0x0000;      //Upper Mask of Last Message (EF0Ch)
LMLM = 0x0000;      //every message into MO 15 (Basic CAN Feature)
MCR_M1 = 0x5555;    //set Message Object 1 (MO1) to invalid, reset MSGAL in all MCR
MCR_M2 = 0x5555;    //set Message Object 2 (MO2) to invalid
/* Define message object 1、2 for data frame transmission: */
/* xtd = no, dlc = 8, TXIE = no, RXIE = no */
/* When sent, MO 1 gen. data frames;it receives remote frames */
/* Remote frames will be automatically answered with a data frame */
MCFR_M1 = 0x88;     //定义信息体 1 为发送,发送 8B,标准 CAN 格式
UAR_M1 = 0XE068;    //信息体 1 标识寄存器初始化 ID-Bits:13 ~ 28, 11-bit
LAR_M1 = 0X0000;    //
MCR_M1 = 0x5995;    //Configure Transmit MO1:MSGAL 位置 1,标明信息体 2 有效
/* Other Message control register loading (transmitobject):
    MCR_MX = 0x59 ??
        01          01          10          01          10
    RMTPND      TXRQ        CPUUPD      NEWDAT      MSGVAL
     15 14       13 12       11 10        9 8         7 6
   user        user          01
  TXIE        RXIE        INTPND
   5 4         3 2          1 0
*/
MCFR_M2 = 0x80;     //定义信息体 2 为接收,接收数据长度 8B,标准 CAN 格式
UAR_M2 = 0XE068;    //信息体 2 标识寄存器初始化 ID-Bits:13 ~ 28,11-bit
LAR_M2 = 0X0000;    //
MCR_M2 = 0x5595;    //Configure Transmit MO2:MSGAL 位置 1,标明信息体 2 有效
/* Other Message control register loading (receive object):
  MCR_MX = 0x55 ??
        01          01          10          01          10
    RMTPND      TXRQ        MSGLST      NEWDAT      MSGVAL
     15 14       13 12       11 10        9 8         7 6
   user        user          01
  TXIE        RXIE        INTPND
   5 4         3 2          1 0
*/
CR = 0x0E;          //Reset CCE and INIT and Set IE、SIE、EIE,初始化完成
```

下面的程序为发送 8B 数据:

```
SR &= 0xF7;         //Reset TXOK
MCR_M1 = 0xFAFF;    //CPUUPD 置 1 使软件刷新数据,NEWDAT 置 1 使信息体有效
DB0_M1 = 0x00;      //给信息体 1 数据赋值
DB1_M1 = 0x11;
DB2_M1 = 0x22;
DB3_M1 = 0x33;
```

```
DB4_M1 = 0x44;
DB5_M1 = 0x55;
DB6_M1 = 0x66;
DB7_M1 = 0x77;
MCR_M1 = 0xE7FF;          //NEWDAT 复位,TXRQ 置位使发送请求
```

此程序用 Keil 公司的 C 编译器和连接器编译连接通过并运行,由 Infineon 公司的 Starter kits C167CR/CS 评估板组成 CAN 节点,实现了 CAN 通信。

参 考 文 献

1 Infineon. C167CR/SR, 16-bit microcontroller manual [M]. Germany: Infineon Technologies AG, 1999,224

2 Infineon. SK-167/167CS Starter Kit Version 3.0 [M]. Germany: Infineon Technologies AG,1999,8-57

3 邬明宽. CAN 总线原理和应用系统设计 [M]. 北京: 北京航空航天大学出版社,1996

4 黄攀,王俊杰. 单总线数字温度传感器 DS1820 及其应用. 仪表技术与传感器,2001 (12): 29-31

选自《仪表技术与传感器》月刊,2002 年第 7 期

6.7 1-WIRE 网络的特性与应用

河北保定华北电力大学动力工程系 (071003) 田 亮 刘鑫屏

一般构成小型分布式测控系统需要采用支持通信功能的单片机设计各个通信单元的硬件,采用 RS485 或 CAN、LONWORKS 总线构成通信网络,并设计复杂的通信协议。这样的网络设计、安装、调试都比较复杂,系统的高成本也限制了其在一些场合中的应用。

1-WIRE 网络是 MAXIM-DALLAS 公司倡导的总线网络。网络总线的构成非常简单,一般情况下包括电源线、地线和信号线 3 根线。在功率消耗不是特别大的场合,甚至只需要电源/信号线和地线 2 根线就可以完成器件的供电和通信。

所有的 1-WIRE 通信协议都集成在器件中,所以构成 1-WIRE 网络十分简单,只需要将器件直接挂在总线上即可。

MAXIM-DALLAS 公司推出的功能丰富的 1-WIRE 器件,使其可以完成一般小型分布式测控系统的功能。

一、1-WIRE 系统结构

系统需要一个总线控制器,一个或多个支持 1-WIRE 的器件,为了增加驱动能力和增加总线所挂器件的数量,可以在网络上增加分支器。总线控制器可以由单片机为核心构成,PC 不是必需的,但一般可以和单片机通信以更方便地监视总线上的数据。总线控制器也可以只是一个通信协议转换芯片,那么总线控制器的功能实际上是由 PC 实现的。

目前支持 1-WIRE 总线的器件,按照功能分类包括:

⑴ 温度传感器系列

DS1820、DS18B20:测量范围 - 55 ~ 125 ,测量分辨率 ±0.5 。

DS1822:测量范围 - 55 ~ 125 ,测量分辨率 ±0.5 。

⑵ 开关量输入输出

DS2405:1 路 PIO。

DS2406:2 路 PIO + 1k 位 EEPROM。

⑶ 模拟量输入

DS2450:4 通道 8 位 A/D 转换器。

DS2438:4 通道 10 位 A/D 转换器。

⑷ 定时/计数器

DS2415:定时/计数器。

DS2417:定时/计数器 + 可编程周期性中断输出。

⑸ 协议接口器件

DS2480B:UART (RS232) 到 1-WIRE 转换。

DS2490:USB 到 1-WIRE 转换。

DS2409：1-WIRE 网络分支（主/辅网络）。

⑹ 接口保护器件

DS9502：ESD（静电放电）保护器件。

DS9503：带电阻的 ESD（静电放电）保护器件。

⑺ 存储器系列

DS2422：带计算器的 1 KB RAM。

DS2423：带计算器的 4 KB RAM。

以上类型只是实际产品的一部分。由器件的类型可以看出，1-WIRE 总线的功能已经可以实现一般的小型分布式测控系统的功能，而其设计上的简便性和低廉的成本是其他系统所无法比拟的。

二、1-WIRE 的通信协议

1-WIRE 总线系统需要一对信号线上连接多个器件。每个器件的输出是采用类似集电极开路的形式，所以必须外加上拉电阻。

由于总线多路复用，因此要求每一个器件必须具有极高的输入阻抗，而总线最多可以挂多少器件，也取决于器件的输入阻抗。总线外加上拉电阻的阻值取决于器件输出可以承受灌电流的能力，电阻阻值一般为 4.7 kΩ，为了保证通信的正常进行，阻值最大为 16.3 kΩ。

1-WIRE 网络通信遵循严格的通信协议。协议有 5 种类型的基本信号在一条线上传输，包括：导前脉冲、复位脉冲、写 0 或写 1、读数据、编程脉冲，其他信号均由导前脉冲控制。

1. 初始化时序

初始化使总线上的设备处于已知状态。其顺序如下：总线控制器发送一导前脉冲，紧接着发低电平的复位脉冲，如图 6.7－1 所示。总线控制器向总线发送的低电平脉冲，包括导前脉冲（1～15 μs）和复位脉冲（至少 480 μs），然后释放总线并进入接收状态。这时总线被上拉电阻拉高（15～60 μs），总线控制器将检测总线上 1-WIRE 器件的低电平返回信号（60～240 μs），整个检测过程必须超过 480 μs。如果总线控制器发出初始化信号后无低电平信号返回，则说明总线上没有 1-WIRE 器件。只有经过初始化过程后，1-WIRE 器件才可以接收其他的信号。

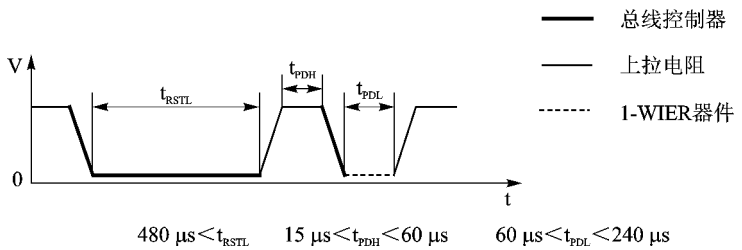
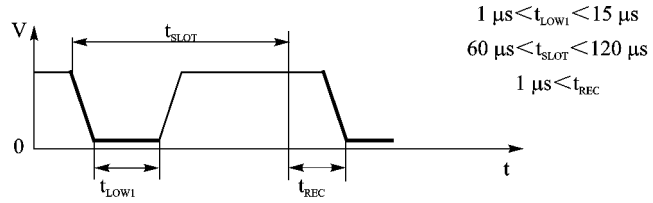


图 6.7－1　1-WIRE 总线复位时序

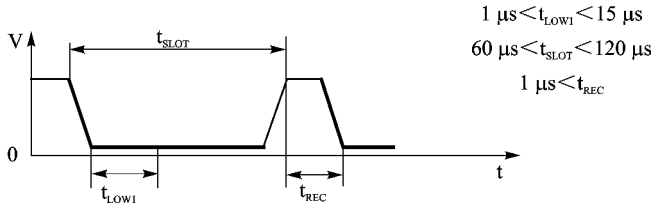
2. 写时序

如图 6.7－2 所示，写“1”过程如下：总线控制器发送导前脉冲拉低总线 1～15 μs，释放总线由上拉电阻拉高，整个过程持续 60～120 μs，再延时至少 1 μs 之后，开始一轮新的过程。写“0”过程如下：总线控制器拉低总线 60～120 μs 后释放总线，再延时至少 1 μs 之后，开始一轮

新的过程。



(a) 写“1”时序



(b) 写“0”时序

图 6.7－2 1-WIRE **总线写时序**

3. **读时序**

过程如图 6.7－3 **所示,控制方拉低总线后马上释放总线,发送方将在** 1～15 μs **内返回数据,数据最多保持** 45 μs**。整个过程持续** 60～120 μs**,再延时至少** 1 μs **之后,开始一轮新的过程。**

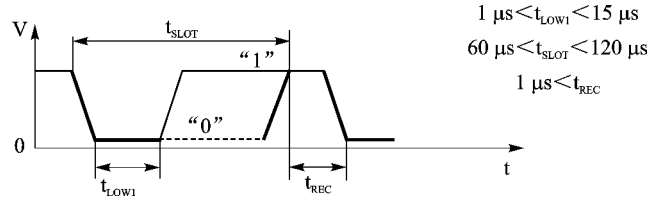


图 6.7－3 1-WIRE **总线读时序**

4. **编程脉冲**

总线退出正常状态至少 5 μs **后,加入** 12 V/10 mA **的功率信号,持续** 480 μs**,再经过至少** 5 μs **后,总线恢复正常状态。**

正常情况下,传送一个二进制位需要约 60～120 μs**,那么总线每秒可以传送** 16 600～8 300**个二进制位,数据手册给出的最大通信速率为** 16.3 Kbps**,实际应用中,通信的波特率一般在**2 400～9 600 bps **之间。**

三、1-WIRE 的指令系统

指令系统包括通用指令和专用指令。通用指令为对所有 1-WIRE **器件都适用的指令,专用指令为只对特定的器件有效的指令。**

通用指令包括复位指令和 ROM **操作指令。**

1. **复位指令**

此信号将使总线上所有的器件处于活动状态,可以接受控制器的指令。

2. ROM 操作指令

读 ROM：33H。此指令将读出器件中保存的惟一的 64 位序列号，包括 8 位公司序列号，48 位产品标志和 8 位 CRC 校验码。当总线上有多个器件时，所有器件将同时回送序列号，使用此指令会产生总线冲突。

配置 ROM：55H。这条指令后跟随一 64 位序列号，用以识别总线上一特定的器件，其他非此序列号的器件将等待一复位脉冲。这样以后的指令就只针对此器件进行了，直到下一个复位信号的到来。实际上 1-WIRE 总线正是依靠这条指令来实现多设备通信功能的。

搜寻 ROM：F0H。这条指令用以搜寻总线上连接器件的 ROM 序列号。总体的过程为：总线管理器读总线上器件 64 位 ROM 中的一位，再读这一位的补码，然后根据需要写一位到总线上，以在这一位上识别某个器件。

下面举例说明：例如总线上有 4 个器件，其 ROM 中的序列码为：

器件 1：XX ~ XX10101100

器件 2：XX ~ XX01010101

器件 3：XX ~ XX10101111

器件 4：XX ~ XX10001000

其中 XX ~ XX 代表器件的更高位，例子中只写了器件的低 8 位。搜索过程如下：

⑴ 总线控制器发出一复位脉冲，开始整个搜索过程。

⑵ 总线控制器发出搜索 ROM 指令（F0H）。

⑶ 总线控制器从总线上读一位。

由于每一个设备都把其 ROM 中 64 位序列码的最低位发送到总线上，在本例中，器件“1”和“4”发送 0，“2”和“3”发送 1，在总线上形成“与”的关系，故总线接收到的将是 0。这时总线控制器将从总线上再读一位，每一个设备都把其 ROM 中 64 位序列码的最低位的补码发送到总线上，器件“1”和“4”发送 1，“2”和“3”发送 0，在总线上形成“与”的关系，故总线控制器接收到的还是 0。

通过一个位的测试可以获得以下信息，如果总线控制器接收到的是 00，说明总线上所有器件至少有一个器件 ROM 序列号中这一位为 0，至少有一个器件 ROM 序列号中这一位为 1；如果总线控制器接收到的是 01，说明总线上所有器件中没有 ROM 序列号中这一位为 1 的器件；同理，如果总线控制器接收到的是 10，说明总线上所有器件中没有 ROM 序列号中这一位为 0 的器件；如果接收到的为 11，则说明总线上没有任何器件。

⑷ 根据第一个位的测试结果，总线控制器决定发送 0 还是发送 1。假如发送 0，对于本例来说，则淘汰了器件“2”、“3”，仅剩下器件“1”、“4”参与以后的搜寻。

⑸ 同第 3 步类似。由于器件“1”、“4”的这一位是 0，所以总线控制器接收到的是 01。

⑹ 总线控制器发送 0，以继续选中器件“1”、“4”。

⑺ 同第 3 步类似，由于器件“1”、“4”的这一位分别是 1、0，所以总线控制器接收到的是 00。

⑻ 根据这一个位的测试结果，总线控制器决定发送 0 还是发送 1。假如发送 0，对于本例来说，则淘汰了器件“1”，仅剩下器件“4”参与以后的搜寻。

⑼ 同第 3 步类似，由于器件“4”的这一位为 1，所以总线控制器接收到的是 10。

⑽ 这样总线控制器将根据接收到的信息发送相应的选择位，以保持器件“4”始终处于

选中状态,直到测试完所有的 64 位 ROM 序列号。这样就选择出了一个器件。

(11) 采用相同的方法测试其余的器件,直到所有器件被测试出。

每一个 1-WIRE 器件都有自己的专用指令。例如对于温度传感器 DS18B20,有读温度信号的指令;例如对于开关量输入输出器件 DS2405,有读器件输入的指令,有控制器件输出的指令。专用指令相对比较简单,使用时可以查阅相应的数据手册。

四、应用简例

由于 1-WIRE 网络以其构成简单、功能丰富的优点,一推出即在各个领域中获得了广泛应用。包括家居安全监测、粮库粮仓温度监控系统、农业大棚监测系统、桥梁建筑应力监测系统、工业现场电缆沟防火监测系统等等。下面以一个火灾监测系统的设计为例说明 1-WIRE 网络的构成与应用。

火灾监测的传感器采用烟雾传感器和温度传感器。烟雾传感器输出的是继电器开关量信号,使用 DS2405 进行采集;温度传感器采用 DS1822。总线控制器使用 PHILIPS 公司的单片机 P89C664,有 64 KB 的 IAP (在应用中编程) FLASH 程序存储器,2 KB 数据存储器,指令系统同 MCS-51 系列单片机兼容。可以利用 P89C664 内部的串行通信口实现与 PC 通信。

在采用信号线供电时,1-WIRE 总线可以支持几百个 1-WIRE 器件,通信距离在 300 m 以内。一条 1-WIRE 总线可挂 1-WIRE 器件的数量主要受到最大允许访问周期的限制,对 DS2405 进行一次访问通信的数据量约为 80 位,对 DS1822 进行一次访问通信的数据量约为 160 位。1-WIRE 总线一般可以达到 10 Kbps 的通信速率,假设系统中有相同数量的 DS2405 和 DS1822,则总线控制器 1 s 中可以访问约 50 个器件。对于火灾监测系统,一般允许的响应时间在几秒之内,系统设计中使用了 100 个器件,访问周期为 2 s,即系统可以对在 2 s 中内任何一个地点的火警发出响应。

五、结束语

本文介绍的 1-WIRE 总线系统有着广泛的应用价值,在此基础上,MAXIM-DALLAS 公司更推出了 i-Button 系列芯片,利用芯片中全球惟一的 64 位 ROM 序列号和要关的技术支持实现 Internet 范围内的芯片级互联。网络信息化是当今时代发展的潮流,这项技术不仅为网络家电、移动 Internet 等技术提供有力的支持,也为以 Internet 等公用通信网络为平台的工业控制和信息处理提供了有力的支持。

参 考 文 献

1 魏智. 1-wrie 网络设计. 国外电子元器件,2002. 2

2 陈兴梧,刘鸣,赵笠. 数字式温度计 DS18B20 的特性及应用. 国外电子元器件,2002. 3

选自《中国仪器仪表》双月刊,2002 年第 5 期

6.8 基于 TINI 的一线制网络互连技术

华东理工大学 姜 捷 王永红 凌志浩

许多器件(如照相机、自动售货机、实验设备等)都具有内置的与外界进行通信的能力。这些设备通常带有处理器,用来管理低层的可与其他电子设备进行通信的端口。而那些不具备与外界通信能力的器件要实现连网则必须借助于某些硬件的支持。TINI 正是为这些设备实现连网而提供的技术支持和实现手段。通过 TINI 可使本来不具备连网能力的器件有效地连入一线制网络,进而被赋予与 Internet 连接的能力,从而满足商业的和工业的嵌入式网络应用的需求。

一、TINI 的基本概念及原理

TINI (Tiny InterNet Interface) 是基于一线制通信协议的设备,能将各种不具备连网能力的硬件简易、灵活、低成本地实现连网。TINI 由软、硬件两部分构成,硬件部分提供有处理、控制以及设备级的通信和连网功能;而软件部分则实现了 Java 的编程运行环境,为系统设计者和用户提供了一系列 Java 应用编程接口。利用 TINI 可以方便地实现一线制网络与信息网络(如 Internet/ Intranet)的互连。

一线制网络作为一种新型的网络技术,采用的是主从结构,其中总线主控器具有总线控制、网络管理等功能,而支持一线制网络通信协议的一线制元器件可充当一线制网络的节点。TINI 作为一种特殊的一线制网络主控器,除了具有一般主控器所具备的功能外,还具有异型网络互连功能,能有效实现一线制网络与 Internet/Intranet 的连接。若对 TINI 设置 IP 地址后,可使之成为 Internet/Intranet 的一个网络节点,供远程用户访问。TINI 可通过其一线制网络接口,不断收集一线制网络上的实时信息,并提供给远程 Internet/Intranet 用户访问。

图 6.8-1 为 TINI 的一般硬件结构示意图,主要包括 CPU、Flash 存储器、SRAM、一线制网络接口、信息网络接口、RS-232C 接口以及辅助电路等若干功能模块。有些 TINI 甚至还带有 CAN 总线接口,直接支持与 CAN 总线网络的连接。

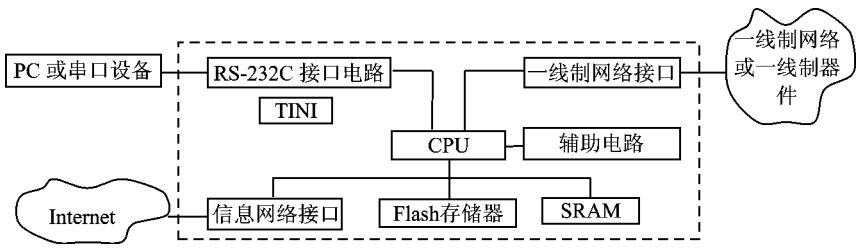


图 6.8-1 TINI 一般硬件结构

CPU 是 TINI 的核心,控制着其他的功能模块。在此,采用由 8051 发展而来的 A390 处理器,其时钟频率可达 40 MHz,指令系统为 32 位,数据指针为 24 位,并带有 2 个串口和 6 个外

部中断输入口。

Flash 存储器和 SRAM 的容量均为 512 KB (字节),TINI 将 Flash 存储器的 512 KB 空间分为 8 页,每页 64 KB。第 0 页放置 bootstrap loader 引导程序。第 1 ~ 6 页放置 TINI OS 操作系统和 Java API 包,其中 TINI OS 提供任务管理、文件系统管理、存储器管理、I/O 管理等功能,以有效协调多个应用程序并发运行;而 Java API 包为用户应用程序的开发提供了支持和便利。第 7 页留给用户应用程序使用。

RS - 232C 串口是为 PC 机或其他串口设备保留的连接接口。PC 机可通过该接口初始化 TINI 以及下载有关应用程序到 TINI 上。各种串口设备也可通过该接口与 TINI 相连,进而实现与信息网络的连接。

由于 TINI OS 支持 FTP 和 Telnet 等服务功能,使得远程用户可通过 Internet 向 TINI 发 FTP 或 Telnet 请求。TINI 监听并接收到来自远程用户的 FTP 或 Telnet 连接请求后,可在两者之间成功建立起 FTP 或 Telnet 连接,并允许远程用户通过执行类 UNIX 命令 (如增删文件、创建文件目录、查询当前访问者、设置 IP 地址及子网掩码等),实现对 TINI 进行操作。

二、TINI 的软件环境

TINI 所需要的软件环境主要包括如下几部分:Flash 存储器中运行的实时操作系统 RTOS、TCP/IP 栈、Java 虚拟机以及 Java API 包;FTP、Telnet、DHCP、DNS 等高层网络协议;JDK 软件开发工具;TINI SDK 等。

TINI 操作系统是一个非常小的嵌入式操作系统,提供有文件管理、内存管理、I/O 管理以及任务调度等基本服务。与大多数小型嵌入式操作系统有所不同的是,TINI 操作系统可在多任务间切换,能够非常好地在多个 Java 字节码解释器正在执行的事件间进行切换,因而可以并发运行多个 Java 应用程序。

TINI 上的 Java 虚拟机与 Sun 公司的嵌入式 Java 平台 1.1 版本的 Java API 包一致。TINI 的 Flash 存储器中装载有 java.lang、java.net、java.io、java.util 和 javax.comm 等函数包。Javax.comm 包是 JDK 所扩充的函数包,提供给 TINI 特殊的 I/O 能力。另外,在 Flash 存储器中还有 com.dalsemi 包,该函数包为进入 TINI 的命令内核 Slush 以及操作 1-Wire 总线、设置众多系统参数等提供支持。若需再装入其他的函数包,可以将它作为应用程序放在 RAM 空间内。

TINI SDK 是进行 TINI 应用开发的软件包,包括 tini.jar、tiniclasses.jar、tini.db、tini.tbin 和 slush.tbin 等。Tiniclasses.jar 中封装有所有的 TINI API 类,它们是实现 TINI 应用开发的基础。Tini.jar 中包括两个重要的程序:JavaKit 和 TINIConvertor。其中 JavaKit 用于引导固件、执行系统维护任务,而 TINIConvertor 则用于将应用程序中的类文件转成 TINI 可执行的二进制文件。Slush.tbin 是 TINI 可执行的 Slush 应用程序。在系统初始化即将结束时,该程序开始运行。Slush 的一个线程创建服务器套接字 (server socket),用来侦听和连接来自客户端 (client) 的 FTP 和 Telnet 连接请求。当没有与远程客户建立 FTP 和 Telnet 连接时,Slush 占用很少的 CPU 资源;当建立起连接并成功登录后,远程用户可以执行类 Unix 风格的命令来操作文件系统、设置或获取配置信息、启动或停止其他的 Java 应用程序等。

三、基于 TINI 的网络互连及应用

1. 温度型 iButton DS1920

DS1920 是一种温度型 iButton 器件,支持一线制通信协议。该器件带有接触式温度传感器,可以数字形式串行输出温度测量值,改变了以往温度传感器需加 A/D 转换器才能转换为数字量的模式。其测量的温度可从 - 50 ~ +125 ,分辨率为 0.5 。其内部电路包括三部分:64 位 ROM、温度传感器和温度报警触发器。工作电源采用"寄生电源"方式供电。

2. 基于一线制技术的小型气象仪

图 6.8-2 为支持一线制通信协议的小型气象仪,上面为风速涡轮,下面是风向标,中间装有温度传感器。温度、风速、风向三个基本气象参数可以方便地通过该气象仪测得。若再装上支持一线制协议的雨量传感器、湿度传感器,则可以获取雨量、湿度等气象参数信息。

图 6.8-2 一线制气象仪

3. 通过 TINI 实现网络互连的应用系统

通过 TINI,可以方便地实现一线制网络与 Internet 的连接,如图 6.8-3 所示。一方面,若干 DS1920 和 TINI 通过双绞线组建成一线制网络。TINI 在一线制网络中处于主控地位,具有网络控制、网络管理等功能;DS1920 处于从设备的地位,主要负责采集所在地的实时温度信息,并送到 TINI 中。另一方面,TINI 通过其所带的信息网络接口,与上层的 Internet 相连。DS1920 采集到的温度信息可以通过 TINI 送到 Internet 上。TINI 在此过程中起沟通两个网络的"桥梁"作用。若在 TINI 上运行 Web 服务程序,不断收集一线制网络上的实时温度信息供远程用户访问,该 TINI 则成为 Internet 上的一个 Web 服务器。若分布于各地的 TINI 将所检测到的实时温度信息送入 Internet 上的某个大型数据库系统中,则构成一基于 Internet 的分布式实时温度监测系统。

若采用一线制气象仪取代 DS1920,那么所构成的应用系统,除了能够检测温度信息外,还可以检测气象仪所在地的风速、风向等信息,从而成为一基于 Internet 的分布式实时气象信息

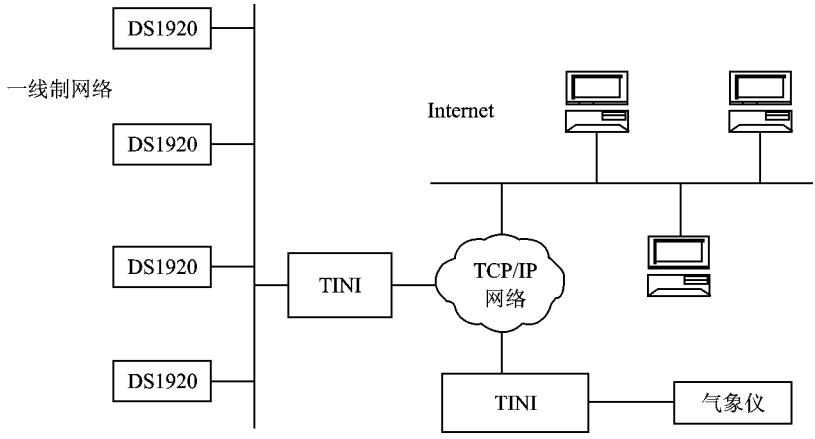


图 6.8-3 通过 TINI 实现网络互连

监测系统。一线制气象仪等一线制设备,除了作为一线制网络的节点与 TINI 连接外,还可以作为单独器件直接与 TINI 相连,并向 Internet 上发布有关信息。

根据 Web 服务器需要从 DS1920 或一线制气象站获取有关信息并通过其向网上发布的应用要求,软件设计应考虑两方面的功能需求:其一,以主控器的身份对 DS1920 或一线制气象站进行寻址并获取有关信息;其二,为远程客户提供 Web 服务。整个程序采用 JDK 结合 TINI SDK 来进行应用编程。

参 考 文 献

1 Dan Awtrey. The 1-wire weather station. http://www.ibutton.com, 2000

2 Dan Awtrey. Transmitting data and power over a one-wire bus. Sensors, 1997, 2

3 Dallas Semiconductor Corp. TINI Developer Guide. 2000

4 王永红,凌志浩. 智能信息载体 iButton 及其应用. 单片机与嵌入式系统应用,2001 (4)

选自《单片机与嵌入式系统应用》月刊,2002 年第 6 期

6.9 单总线数字温度传感器的自动识别技术

广西工学院电子信息与控制工程系 罗文广

一、引 言

在多点温度测量系统中,单总线数字温度传感器(例如 DS18X20)因其体积小、构成的系统结构简单等优点,应用越来越广泛。每一个数字温度传感器内均有惟一的 64 位序列号(最低 8 位是产品代码,其后 48 位是器件序列号,最后 8 位是前 56 位循环冗余校验码),只有获得该序列号后才可能对其进行操作,也才能在多传感器系统中将它们一一识别。实际应用时的一般做法是:将每一个传感器的序列号测出,以表格的形式和程序存放在一起,并且给每个测温点编上号,做成标签粘贴在对应的传感器上。当系统中有传感器故障时,必须由专业人员测出备用的传感器序列号,贴上相应的标签,并在程序中修改表格,再将程序固化到程序存储器中。显然,这样做非常不利于系统维护。

现有的单总线数字温度传感器的文献很少涉及自动识别序列号和排序(即与测量点对应)的问题,文献[1]给出了一种方法:通过特制的编码器,将一个传感器的序列号读出,并将其中 48 位器件序列号转换成 BCD 码,再通过手动拨盘将测温点编号拨入编码器,与器件序列号一起写入到传感器内的上下限温度报警寄存器 TH/TL 中(两个字节的 EEPROM)。使用该方法,系统可以由运行人员来维护,并减少维护工作量,但仍有缺点:需要专门的编码器,维护工作量减少得仍不够;必须是在 TH/TL 不使用的前提下。本文给出一种方法,只需在系统中增加一片 EEPROM 芯片,通过编程,可实现多个传感器的出错指示、自动识别。

二、硬件设计

图 6.9-1 为系统电路原理图,主要由 PIC 单片机 PIC16C63、四位 LED 显示器、锁存器 74LS373、二-四译码器 74LS139、2 KB 的 EEPROM 存储器 DCM0016C、拨位开关 K 以及数字温度传感器 DS18B20 等组成。PIC 系列单片机是一种采用精简指令集(RISC)、哈佛(Harvard)双总线和两级指令流水线结构的高性价比的 8 位嵌入式控制器,I/O 口线直接驱动 LED,片内有 4 KB 的程序存储器和 256 B 的数据存储器。可电改写的 DCM0016C 主要用于存储 DS18B20 的序列号。LED 显示器用于显示各测量点的编号、温度以及传感器故障时的指示。拨位开关 K 在系统正常运行时处于打开状态,需要更换传感器时将其拨至闭合位置,单片机调用相应的子程序进行传感器自动识别。图中数字温度传感器 DS18B20 的接线拓朴结构考虑了两种结构:总线结构和星型结构。总线结构是指在一根 I/O 口线上挂接若干只温度传感器,星型结构则在若干根 I/O 口线上分别挂接若干只温度传感器(图中虚线表示)。实际应用时考虑单总线的驱动能力、布线问题,更多地是采用星型结构,同时这种结构更容易对多个温度传感器进行出错指示、自动识别其序列号和排序。

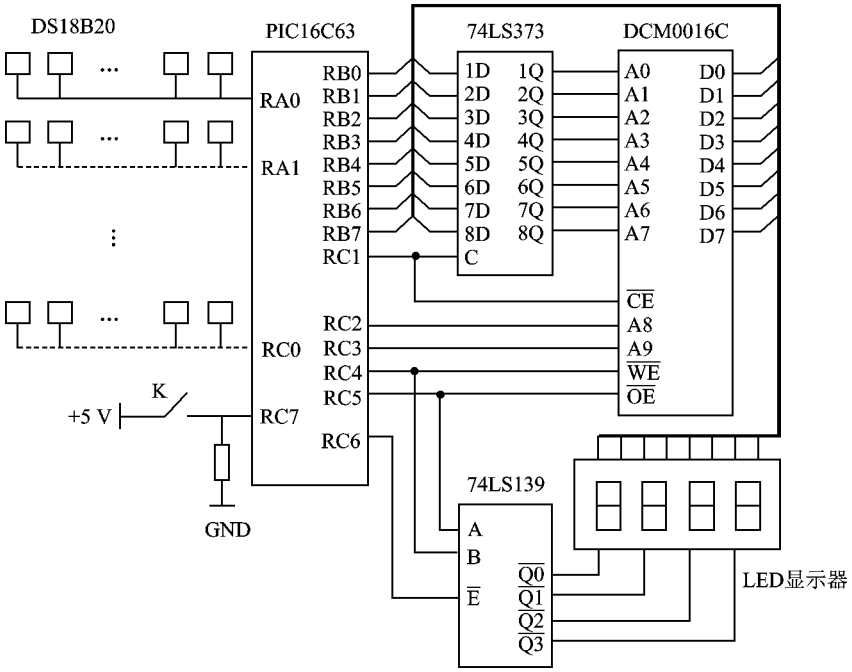


图 6.9－1 系统电路原理图

三、总线结构的传感器识别

1. 获取序列号的 ROM 操作命令

操作单总线数字温度传感器必须严格按规定的协议操作,即应按以下顺序操作:初始化、ROM 操作命令、暂存存储器操作命令。在 ROM 操作命令中,有两条命令专门用于获取传感器序列号:读 ROM 命令(33H)和搜索 ROM 命令(F0H)。读 ROM 命令只能在总线上仅有一个传感器的情况下使用。搜索 ROM 命令则允许总线主机使用一种"消去"处理方法来识别总线上所有的传感器序列号。搜索过程为三个步骤:读一位,读该位的补码,写所需位的值。总线主机在 ROM 的每一位上完成这三个步骤,在全部过程完成后,总线主机便获得一个传感器 ROM 的内容,其他传感器的序列号则由相应的另外一个过程来识别。具体的搜索过程为:(1) 总线主机发出复位脉冲进行初始化,总线上的传感器则发出存在脉冲做出响应;(2) 总线主机在单总线上发出搜索 ROM 命令;(3) 总线主机从单总线上读一位。每一个传感器首先把它们各自 ROM 中的第一位放到总线上,产生"线与",总线主机读得"线与"的结果。接着每一个传感器把它们各自 ROM 中的第一位的补码放到总线上,总线主机再次读得"线与"的结果。总线主机根据以上读得的结果,可进行如下判断:结果为 00 表明总线上有传感器连着,且在此数据位上它们的值发生冲突;为 01 表明此数据位上它们的值均为 0;为 10 表明此数据位上它们的值均为 1;11 表明总线上没有传感器连着;(4) 总线主机将一个数值位(0 或 1)写到总线上,则该位与之相符的传感器仍连到总线上;(5) 其他位重复以上步骤,直至获得其中一个传感器的 64 位序列号。

根据以上分析,搜索 ROM 命令可以将总线上所有传感器的序列号识别出来,但不能将各传感器与测温点对应起来,即不能实现真正意义上的自动识别。

2. 关系表的建立

以总线上连着 8 个传感器为例。系统中使用相同型号的传感器,因此产品代码都是一样的,比如 DS18B20 为 28H,可以只用一个单元存储该代码。用 DCM0016C 的第一个单元 0000H 存储。剩下的 56 位序列号用 7 个字节单元来存储。对每个测温点进行编号,与存储序列号的单元地址建立如表 6.9-1 所列的关系。

表 6.9-1 关系表

位置编号	存储单元地址
1	0001H~0007H
2	0008H~000EH
3	000FH~0015H
4	0016H~001CH
5	001DH~0023H
6	0024H~002AH
7	002BH~0031H
8	0032H~0038H

工程调试时,要设法将每个测温点的传感器序列号写入对应地址的单元中,即用编程方式自动建立表 6.9-1,则可实现传感器的自动寻址和排序。程序流程图如图 6.9-2 所示。

按图 6.9-2 编制的程序工作,需要按一定的要求操作:按位置编号由小到大顺序插入传感器。具体工作流程为:

(1) 首先检测总线上是否挂接了传感器。若没有,则四位 LED 显示器显示“0000”;有则识别传感器的序列号。

(2) 识别传感器序列号。从 0 位到 63 位共 64 位二进制数分别识别出来,若该序列号与 EEPROM 单元的内容不同,说明获得了一个新的序列号,将其存到相应的单元中,并显示位置编号和“PPPP”。例如,在 1 号位置插入一个传感器,单总线上只挂接一个传感器,很容易将其序列号识别出来,并存入到 0001H~0007H 单元中;接着在 2 号位置插入另一个传感器,单总线上挂接了两个传感器,系统首先识别出一个序列号来,若与存储在 0001H~0007H 单元的内容相同,说明这次识别出的仍是 1 号传感器的序列号,不存储并接着进行另一个过程,得到不同的序列号(即 2 号传感器的序列号),将其存储在 0008H~000EH 单元中。获得序列号的同时也就获得了总线上挂接传感器的个数。

由上述知,总线上挂接多个传感器时,若编制的程序不当,可能同一个传感器要被识别多次,识别时间增长。为了减少识别时间,编程时要注意“需要的位值”取值。传感器序列号的最低 8 位为产品代号,“需要的位值”可按对应的值给出,关键是其后的 48 位器件序列号的识别。这里采用了“完全二叉树”的排序思想,如图 6.9-3 所示。具体思路:设在 K 位首次发生数据位冲突,这时所有的传感器分成两类,即该位为 1 的传感器和为 0 的传感器。“需要的位值”给 1,K 位为 1 的传感器仍挂接在总线上。若接下来 K+M、K+N 位发生数据位冲突,“需要的位值”仍分别给 1,获得一个序列号。下一个过程在 K、K+M 位“需要的位值”仍给 1,但在K+N位则给 0,获得另一个传感器的序列号。第三个过程在 K 位仍给 1,而在 K+M 位给 0,在这条支路上继续识别。K 位为 1 的传感器的序列号识别完后,回到 K 位时,“需要的位值”

给0,按同样的方法识别该支路的传感器序列号。按此思路,多个传感器的序列号只需要分别识别一次。

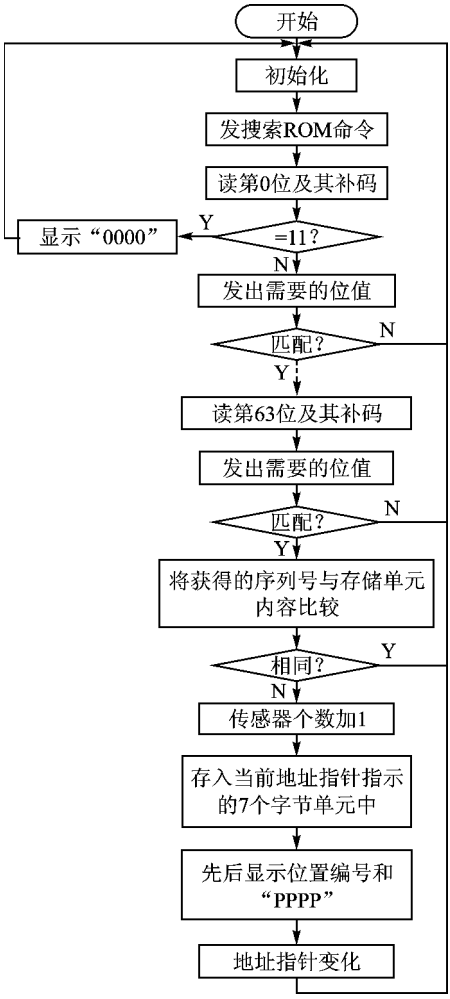


图6.9-2 建立关系表的程序流程图

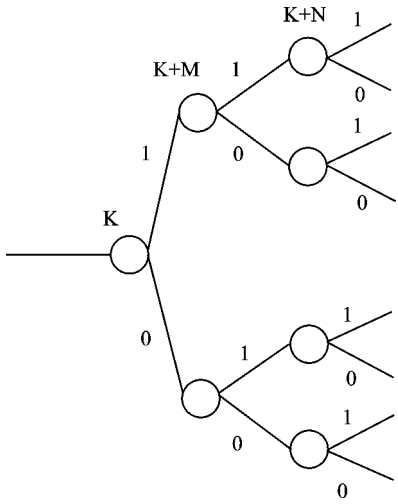


图6.9-3 完全二叉树

3. 出错指示

建立关系表后,编制好程序,系统可投入运行。读取每个测温点的温度时,需要用到“符合”ROM命令,该命令要求将关系表中的序列号取出送到总线上,只有序列号与之相符的传感器才挂接在总线上,可读取其温度。若相应的传感器出错,显然不会有序列号与之相符的传感器,这时显示器显示位置编号和“FFFF”,表明该测温点的传感器出错。

4. 更换传感器后的自动识别

给8个传感器建立临时关系表,占用EEPROM存储单元地址范围为0039H~0070H,将更换后传感器的序列号存入这些单元中。当只有一个传感器出错时,临时关系表中的序列号与关系表的序列号比较,只有一个号不同,用该号取代前文所述的关系表中序列号,即存入对应的存储单元中,便完成了更换传感器后的自动识别。有两个及以上传感器出错时,若同时将这些传感器更换掉,则它们的序列号同时进入临时关系表,将无法进行排序,因此更换传感器

时也要按一定的顺序进行:只能一个一个更换,且位置编号小的先更换;更换一个传感器,显示器显示位置编号和“PPPP”后,再更换下一个传感器。实现该功能的子程序主要流程:(1)建立临时关系表;(2)临时关系表的内容与关系表比较。建立临时关系表子程序与建立关系表的程序基本一样,主要是存储单元地址不同。建立该表的同时也就获得了总线上挂接的传感器数,该数据决定两个表的比较次数。

5. 星型结构的传感器识别

拓扑结构采用星型结构,如图 6.9-1 所示,RA0、RA1、RA2、RA3、RA4、RA5、RC0 为七根单总线,分别挂接若干个传感器,以挂接 8 个为例,建立表 6.9-2 所列的关系表。

表 6.9-2 星型结构的关系表

总线	位置编号范围	存储单元地址	临时存储单元地址
RA0	1~8	0001H~0038H	0039H~0070H
RA1	9~16	0071H~00A8H	00A9H~00E0H
RA2	17~24	00E1H~0118H	0119H~0150H
RA3	25~32	0151H~0188H	0189H~01C0H
RA4	33~40	01C1H~01F8H	01F9H~0230H
RA5	41~48	0231H~0268H	0269H~02A0H
RC0	49~56	02A1H~02D8H	02D9H~0310H

因为各单总线分别操作,它们的编程及操作和上述单总线差不多,此处不再详述。采用星型结构有一个主要优点:当七根总线上分别只有一个传感器出错,按上述编程思想,可以同时更换七个传感器,有利于系统维护,工作量相应减少。

综上所述,用简单的硬件以及编程方法自动建立关系表,在单总线多点温度测量系统中实现了数字温度传感器的出错指示、自动识别,大大有利于系统的调试、维护,减少维护工作量,并解决了过去维护工作必须由专业人员来完成,而不是由运行人员来完成的不便。

参 考 文 献

1 陈良光,刘剑亮. DS18X20 在多点测温中的编码优化技术[J]. 传感器技术,2001,20(9)

选自《电子产品世界》月刊,2002 年第 8 期

6.10 TM 卡信息纽扣在预付费水表中的应用

淮海工学院 鱼瑞文 龚成龙

智能水表是一种涉及到多方面技术整合的机电一体化系统,在计量控制精度、功耗、数据保密性、动作可靠性等方面都有严格的性能要求。目前,国内企业与研究机构主要致力于远传表有线水表网络和 IC 卡预付费智能卡无线水表网络方面的研究开发。远传表有线网络系统需要配套的远传通信网络支持,其初期投资大,因此只适用于在一些新建住宅小区组成相对独立的小网;而后者的网络服务可由银行的金融网络提供,应用储值卡作为网络传输介质,可容易地组建城域网,不但适用于新建住宅小区,还可在旧水网系统改造中发挥极大作用,特别适合我国现阶段的国情。目前,预付费智能水表主要采用 IC 卡,在信息保密性与防盗用性能方面存在明显不足。本文研究应用 Dallas 公司的 TM 卡半导体信息纽扣作为充值存储介质,构建预付费智能卡式冷水水表,在基表上附加计量控制器和管道开闭控制阀门,实现预付费智能卡式水表的设计。

一、TM 卡信息纽扣应用研究

1. TM 卡信息纽扣简介

作为一种便携式信息载体,Dallas 公司的 iButton 信息纽扣[1]可靠安全,特别适宜作为预付费充值载体。它采用直径 17 mm、厚 3 ~ 6 mm 的纽扣状不锈钢外壳封装,内部由 I/O 处理器和存储器两个基本部分组成,其内部结构如图 6.10 - 1 所示。作为一种新颖的智能化信息载体,iButton 信息纽扣采用接触式存取方式的存储器(Touch Memory,简称 TM 卡),以1-Wire规范作为通信协议,其外壳为信号地,用 1 根数据线按照特定的时序要求由数据线逐位与外界交换数据。与微处理器的典型接口电路如图 6.10 - 2 所示,当外部上拉引线接触到其信号线时,可自动发出下拉报到脉冲,存取操作极为方便。

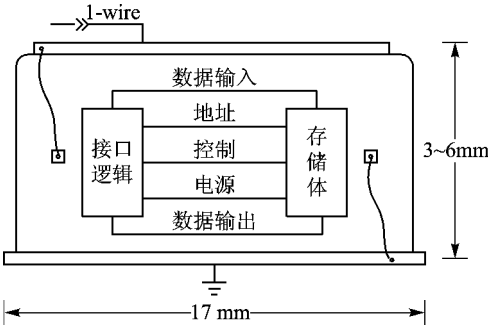


图 6.10 - 1 内部结构

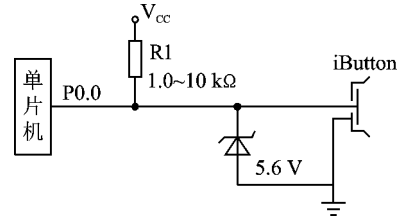


图 6.10 - 2 典型接口

笔者在 TM 卡智能预付费冷水水表项目研究中,通过对 IC 卡、TM 卡的存储数据格式、保

密性能以及耐用性等多方面性能进行比对研究,最终选定 TM 卡作为水表智能信息存储介质。TM 卡采用不锈钢外壳封装,无易损部件或易腐部件,具有携带方便、抗撞击、防水渍、耐腐蚀、抗磁扰、防折叠等显著特点,适用于恶劣的环境。1-Wire 通信协议使得 TM 卡存取数据极为方便,与触头轻轻一碰,瞬间即可完成数据信息的读写操作;其完善的保密存取方式,确保数据信息具有相当高的安全可靠性。TM 卡与普通 IC 卡的性能比较如表 6.10－1 所列。

表 6.10－1　TM 卡与 IC 卡的性能比较

性　能	TM卡	普通IC卡
耐用性	不锈钢封装,寿命在10年以上	塑料卡片封装,弯曲会损坏芯片,开放的引脚裸露,容易磨损导致接触不良,18个月左右寿命
识别成功率	100%	需要一定的刷卡技巧
使用方便性	瞬间接触,无需对准	4个不同插卡方向,但只有1个方向是正确的
安全性	具有全球惟一的、不可复制的光刻序列号,同时提供64位密码保护	单纯依靠软件密码保护数据内容,且密码位数少,容易被破解
携带方便性	不怕高温、磨损、碰摔,耐腐,不易丢失	卡片容易丢失、折断、易腐,须小心保护

2. 1-Wire 通信协议与读写控制程序设计

TM 卡内部包含有 3 个独立的 64 字节数据存储区和 1 个 64 字节读写缓冲区,每个数据存储区均由 8 字节 ID 身份码、8 字节 PASSWORD 和 48 字节 NV RAM 数据区构成。TM 卡的 1-Wire 通信协议以 15 μs 低电平脉冲表示数据“1”,以 60μs 低电平脉冲表示数据“0”。内部工作过程可描述为:首先,由微机主动向 TM 卡发测试脉冲,以识别 TM 卡是否已与其触头接触,若已正确连接,可接收到 TM 卡发来的应答脉冲,表示可以进入数据通信过程。这时,微机先发操作 TM 卡的 ROM 区的指令,如读 ROM 区数据指令、匹配操作指令、搜寻操作指令等,这些指令被 TM 卡接受并执行。然后,发操作 TM 卡的 NV RAM 区数据的指令,如读写 NV RAM 区数据指令、读写或复制读写缓冲区数据的指令等。TM 卡的读写时序可分为测试连接与应答、从 TM 卡读取数据和向 TM 卡写入数据 3 种类型。

笔者在项目研发中重点研究了 TM 卡的信息存储机理,并研究获得了一个性能优良的密码保护模式,确保了一表一卡对应。我们根据研究获得的信息存储模式编制了存取可靠安全的 TM 卡读写控制程序模块。水表控制器 1-Wire 读写模块采用 MCS-51 汇编语言,按 TM 卡通信协议编制。管理 PC 机的读写控制程序采用 Visual C++6.0 编写。应用软件开发过程中,对 TM 卡进行数据读写的过程需要遵循其工作机理和时序要求,具体包括:

⑴ 测试连接及应答。微机发测试负脉冲给 TM 卡,查询 TM 卡是否已与触头正确连接。若与触头连接良好,TM 卡则将数据线拉低,产生应答负脉冲。如果微机检测到这个应答脉冲,就可以进行数据读写操作了。

⑵ 从 TM 卡读取数据。微机先向 TM 卡发 1 个读负脉冲,TM 卡接收该脉冲后立即将被读取位的内容送至数据线上,微机从数据线上获得数据。若数据线在 TM 卡的采样时区内维持高电平,则读取值为“1”;否则,为“0”。最后,TM 卡释放数据线,数据线恢复为高电平,为微机继续从 TM 卡读取数据位做好准备。

⑶ 将数据写入 TM 卡。与读取数据类似,微机向 TM 卡发 1 个写负脉冲,然后开始写数

据。微机维持数据线低电平特定时间,再恢复为高电平,则表明写入“0”;微机发出写负脉冲后立即将数据线拉高并维持特定时间,则表明写入“1”。完成数据写入后,数据线恢复为高电平,为微机继续向 TM 卡写入数据位做好准备。

二、TM 卡预付费智能冷水水表设计

TM 卡预付费智能冷水水表由基表、SCP 微处理器系统、LCD 显示驱动电路、电动陶瓷阀门及其控制电路、刷卡电路等部分组成。采用符合 ISO4064B 标准的 CDTAR 系列单流旋翼式冷水水表作为基表。该表计数机构与测量机构经磁耦合传动,可将用水量转换成电信号输出;表内设有磁保护装置,具有较强的抗外磁干扰能力。设计过程中重点对水表整机功耗、成本、体积、重量、外观等方面进行优化研究,水表控制器设计中采用了 I^2C 总线、最小功耗设计、表面贴装技术专门定制 LCD 液晶驱动器模块以及产品造型设计等多项先进技术,完成嵌入式机电系统优化设计。

1. 微控制器系统设计

使用的微控制器 P87LPC76X 属于 MCS－51 兼容型[2],与标准 51 单片机相比,尽管只有 20 引脚,却提供了 I^2C 通信总线、灌电流达到 20 mA 的 18 条 I/O 口线、1 个 WDT 看门狗定时器。它具有许多独特的功能,特别适用于设计高集成度、低成本、低功耗的智能设备。本项目充分利用这些资源,扩展了 I^2C 接口 EEPROM、时钟芯片电路、LCD 显示驱动电路,完成高集成化系统扩展设计。控制器硬件框图如图 6.10－3 所示。

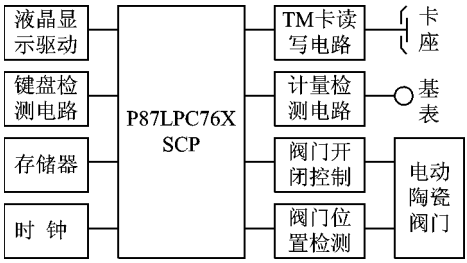


图 6.10－3 控制器硬件框图

2. 水表控制器程序设计

P87LPC76X 单片机检测到用户按键后,通过 TM 卡读写电路读入用户购买的水量,并保存到数据保存器 EEPROM 中,同时液晶显示模块显示用户购买的水量及表中剩余的水量。阀门驱动检测电路检测阀门开关状态,打开阀门,用户可以正常用水。水表每 10 升发 1 个计量脉冲,MCU 通过计量检测电路检测到水表发来的脉冲后,从数据保存器 EEPROM 中保存的剩余水量中减去 10 升。当剩余水量为提示性关阀水量时,MCU 通过阀门驱动检测电路关闭阀门。用户按键后控制电路打开阀门,恢复供水,当剩余水量为预设值(通常为零吨)时彻底关阀。整表工作状态检测电路主要实现整个控制电路的电池供电电压的检测、磁干扰检测、TM 卡读写电路异常检测、阀门开闭异常检测等,并将整表工作状态信息保存到数据保存电路中,同时还写到 TM 卡中,以便于收费管理软件在读写用户 TM 卡时能及时了解用户水表的工作状态。用户在任何时候都可以按键,通过液晶显示用水情况及水表工作状态。程序流程如图 6.10－4 所示。

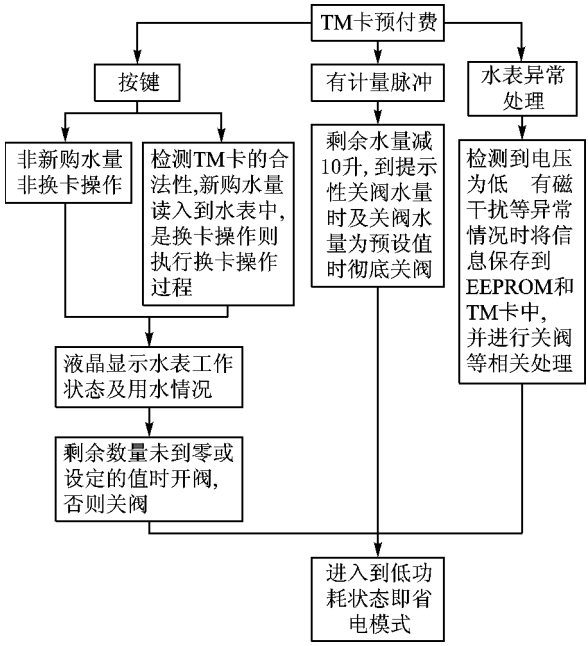


图 6.10－4 控制器程序流程图

3. **电池长寿供电与最小功耗设计**

智能水表作为一种新型换代产品,必须具备良好的免维护性能,其设计目标应为在规定整机寿命内,需要用户进行的相关操作越少越好。本设计采用以下技术措施,彻底解决了目前各类 IC 卡预付费水表均未能突破的电池供电最小功耗设计这一难题,整机待机电流只有18～40 μA。

⑴ 应用 P87LPC76X 单片机的节电模式,降低控制器功耗。CDTAR 基表每 10 升发 1 个计量脉冲,若设用户每月充值 1 次,若再设计 LCD 显示器只在用户按键后显示一段时间后自动关闭,这意味着控制器只在极小的时间片内进行流量检测、读写 TM 卡和显示操作,控制器在绝大部分时间片内可以处于待机状态。P87LPC76X 单片机支持掉电模式,掉电工作电流仅 1μA。控制程序设计中,P87LPC76X 单片机自检后即将 PCON 寄存器内 PD 位置"1",进入掉电模式,以 TM 卡插卡中断、按键中断、流量脉冲中断、WDT 定时中断(对应程序跑飞出错等故障)作为唤醒 P87LPC76X 的动作,P87LPC76X 处理完相应中断服务后再次进入掉电模式。经测试发现,采用这一方式可使 P87LPC76X 以极低的平均电流实现对系统的控制操作。

⑵ 合理完善的接口电路低功耗优化措施,降低整机功耗。P87LPC76X 进入掉电模式后,外围接口电路的功率消耗不可忽视。本设计中的外围接口电路主要有:EEPROM、LCD 显示驱动、日历时钟、流量脉冲检测电路、阀门开闭控制电路等。为减小电流消耗,需对电路进行反复精简优化,采用的措施包括:应用 I^2C 总线设计外围接口,降低功耗;尽量由 P87LPC76X 的 I/O 口线提供外围芯片供电或片选端控制;门电路使用 CMOS 电路;上拉电路采用小电流结构。

⑶ 灵巧合理的电动陶瓷阀门,降低控制执行机构耗能。自行开发了 DC2.7～3.6 V 电池供电陶瓷电动阀门,其机械结构灵巧合理,动作电流仅 120 mA。设计独特的阀门开闭状态

位置检测机构既保证了动作可靠性,也可避免无谓的电池能量损失。

(4) 严密完善的电池寿命测算,保证长寿供电。设计中,我们选取武汉力兴 14505M DC3.6V/3Ah 功率型锂电池。该电池具有自放电电流小、瞬时电流大等优点,符合系统长寿供电要求。

三、结束语

三表(电表、水表、煤气表)智能网络工程是我国重点支持的新兴高技术产业,《中国住宅产品发展纲要》等国家建设部和科委的若干文件中都明确对三表提出了智能化、网络化要求。开发可靠价廉、易于推广应用的预付费智能卡式水表及其网络系统是今后智能水表行业的主要产业发展方向。本文介绍的基于 TM 卡的预付费智能冷水水表已通过江苏省质量技术监督局的样机试验检测。该表计量范围为 0 ~ 99 999.99 m^3,单电池工作寿命 6 ~ 10 年,工作稳定可靠,数据保密性强,具有良好的抗外力敲击、外磁干扰等恶性盗用能力。各项技术指标符合 GB/T778.1.2.3-96《冷水水表》国家标准和 CJ/T133-2001《IC 卡冷水水表》行业标准的要求。

参 考 文 献

1 Dallas Semiconductor Corporation. Book of DS19xx iButton Standards. http ∶//www.ibutton.com

2 周立功,陈智红. P87LPC76X OTP 单片机使用指南. 广州周立功单片机发展有限公司,1999

选自《单片机与嵌入式系统应用》月刊,2002 年第 12 期

6.11 USB 2.0 性能特点及其应用

山东曲阜师范大学计算机系 (273165) 齐邦强

通用串行总线 (Universal Serial Bus,USB) 是在 PC 领域广为应用的外设连接规范。随着 PC 性能的提高,处理大容量数据的能力越来越强。与此同时,PC 外设的性能也随之加强。许多应用如数字影像处理等,要求在高速 PC 和高速外设之间更快地传输数据。而目前普遍采用的是 1998 年发布的 USB1.1 规范。该规范只提供了低速 (Low-Speed) 1.5 Mbps 和全速 (Full-Speed) 12 Mbps 两种传输模式,不能适应更快的数据传输要求。2000 年发布了新的 USB 2.0 规范,该规范提供了 480 Mbps 的高速 (High-Speed) 传输模式,以满足更快的数据传输要求。本文就 USB 2.0 的性能特点和应用作些探讨。

一、USB 2.0 性能分析

USB 2.0 是对原有规范 (USB1.1) 的自然发展,为大数据量传输提供足够的带宽。

USB 2.0 的主要性能特点如下。

(1) 完全兼容 USB1.1。USB 2.0 使用和 USB 1.1 版本相同的电缆、连接器和软件接口。从用户角度看,USB 2.0 很像原来的 USB 1.1,只不过具有更高的带宽。遵循原规范的 USB 外设可以在支持 USB 2.0 规范的 PC 上继续使用,用户使用 USB 设备的方式也没有变化。

(2) 用户对 PC 外设扩展简单,支持热插拔。USB 2.0 采用标准化的电缆和连接器 (插头和插座),用户可以很方便地把外设连接到 PC 上,PC 能自动识别外设,自动配置设备,并实时感知设备的动态接入和动态移除;协议内置错误捕捉/故障覆盖机制,支持对故障设备的识别;支持多达 127 个外设,可以为低功耗外设提供电源,并允许使用复合设备 (由多个外设组成的设备)。

(3) 支持 480 Mbps 的传输速度,可以选择的外设类型广泛。外设和主机的通信带宽可以从几 Kbps 到几百 Kbps;协议内置流量控制,支持多设备同时发生的事件;支持主机和设备间的多种数据及信息流;同一线路中,既支持等时传输模式也支持不等时传输模式;保证电话、音频、视频等实时信息的传输带宽和反应时间。

(4) 成本低。USB 2.0 继承了原 USB 低成本的特点,性能的提高而对系统的成本影响很小。用户已经购买的 USB 外设可以工作在 USB 2.0 系统中,而且有更高性能的外设可供选用。高带宽的接口、简单的结构使将来许多 PC 只需要 USB 连接器,在同一个系统中将不再需要各种适配器,因而降低了成本。系统制造商将用 USB 2.0 替代 USB 1.1。

USB 2.0 支持 3 种不同的传输速度,对不同的外设根据速度的要求采用不同的传输模式。3 种传输模式适用的外设和特性如表 6.11-1 所列。

表 6.11－1 USB 2.0 3 种传输模式适用的外设和特性

传输模式	适用外设	特 点
Low-Speed (10～100 Kbps)	键盘、鼠标、输入笔、游戏杆等	低费用、易用、动态连接/动态分离、可连接多个外设
Full-Speed (500 Kbps～10 Mbps)	电话、压缩视频设备、宽带设备、音频设备、麦克风等	低费用、易用、动态连接/动态分离、可连接多个外设、有保障的带宽、有保障的反应时间
High-Speed (25～480 Mbps)	视频设备、外部存储设备、图像设备、宽带设备	低费用、易用、动态连接/动态分离、可连接多个外设、有保障的带宽、有保障的反应时间、更高的带宽

二、USB 2.0 结构

一个完整的 USB 系统由 3 部分组成：USB 互连、USB 设备、USB 主机。

USB 互连是 USB 设备与 USB 主机连接和通信的方式。包括：

⑴ 总线拓扑：USB 设备和主机连接的模型。

⑵ 数据流模式 数据通过 USB 总线在数据产生者和消费者之间流动的方式。主要有控制、等时、中断和块传输 4 种模式。

⑶ 层间的关系 在逻辑上，USB 可以分成几个逻辑层，USB 的通信是在每一层按顺序完成的。每一层完成特定的任务，将结果交给上层或下层。

⑷ USB 任务计划表：USB 提供共享的连接。为了等时传输数据和消除竞争，必须将所有对 USB 连接的存取安排在任务计划表中。

USB 设备有 2 种：① Hub，用来提供额外的 USB 连接端口的设备。② 功能部件（Function），扩充系统性能的设备，如数字操纵杆、喇叭、键盘、鼠标等。

USB 设备提供标准的 USB 接口，包括如下信息：① 有关的 USB 协议。② 对 USB 基本操作的响应。③ 基本性能描述信息。

USB 主机负责 连接或移除 USB 设备，管理主机和 USB 设备之间的控制流及数据流，收集各部件的状态，为接入的设备提供电源。在任何 USB 系统中，只能有 1 个 USB 主机。主机中对 USB 的接口称为主机控制器（Host Controller），它是硬件、固件和软件的组合体。在主机内部集成了 1 个根 Hub（Root Hub），对外提供 1 个或多个端口。

USB 外设通过 USB 总线和 USB 主机相连。USB 的物理连接是分层的星型拓扑结构，如图 6.11－1 所示。USB 主机位于拓扑的中心。它每一级之间都是点对点连接，可以是主机和 Hub 或主机和功能部件的连接，也可以是 Hub 和其他 USB 设备连接。为防止产生闭路，USB 的星型拓扑的分层顺序是必须的。为了限制主机到 Hub 或功能部件的传播时间，物理连接最多不能超过 7 层（包括根层）。如果没有根 Hub，主机到 USB 设备的通信通路不能超过 5 层。因为复合设备要占用 2 层，因此主机不能处在第 7 层，只有功能部件才能处在第 7 层。

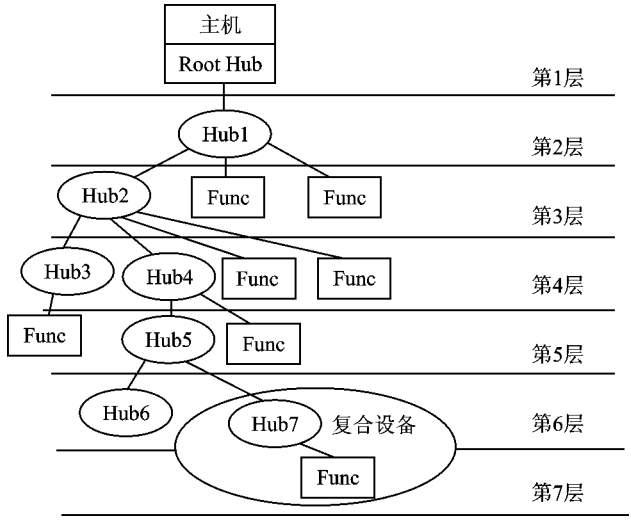


图 6.11－1　USB 的物理连接

三、USB 2.0 的应用

USB 2.0 是 USB 1.1 规范的优化，提供了更高性能的接口。USB1.1 连接器和全速电缆无需任何改变就可以支持 USB 2.0。传输速度是以设备–设备为基础协商的。如果某一外设不支持高速传输，连接操作将在该外设设定的 12 Mbps 或 1.5 Mbps 的较低速度下进行。USB 2.0 系统的应用连接示例如图 6.11－2 所示。

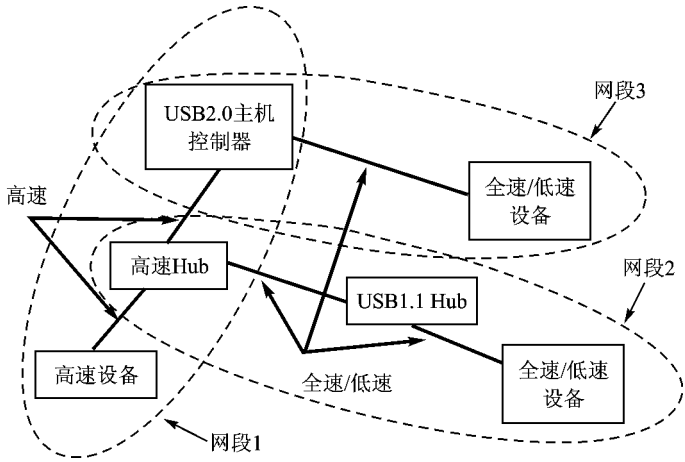


图 6.11－2　USB 2.0 系统的应用连接示例

高速设备连接在根 Hub、外部 USB 2.0 Hub 和高速外设之间。全速/低速设备通过USB 1.1 Hub 和高速 Hub 相连（也可直接连在根 Hub 上），这样就形成了 3 个网段。USB 2.0 Hub 在更快的速率下接收高速传输的事务，且必须分发给高速 USB 2.0 外设和 USB 1.1 外设。最简单的情况是与 USB 2.0 外设的通信。Hub 在上游和下游电缆上重发高速信号，就像 USB 1.1 Hub 对 USB 1.1 设备重发全速和低速信号一样，并允许 USB 2.0 外设利用大部分 USB 2.0 带宽。为与

USB 1.1 外设通信,USB 2.0 Hub 包含一个支持数据速率与下游设备的性能相匹配的机制,即 Hub 管理数据速率从主机控制器的高速到 USB 1.1 设备的低速的转换。USB 2.0 Hub 的这种特性意味着 USB 1.1 设备可以和 USB 2.0 设备一起操作,而且不需要消耗与其不成比例的大量 USB 2.0 的带宽。

各个部件都支持高速模式,网段 1 可以工作在 480 Mbps 传输速度的模式下。全速/低速设备和 USB 1.1 Hub 不支持高速模式,网段 2 只能在全速/低速模式下工作。网段 3 中根 Hub 适应全速/低速设备的传输速度,故网段 3 可工作在全速/低速模式下。

系统软件能理解 USB 2.0 外设性能的增强,并最优化外设配置。例如,当一个 USB 2.0 外设连接到 USB 1.1 Hub 时,它将警告用户并要求对连接的外设作更好的配置。新的应用将会更好地利用 USB 2.0 外设及设备的高速性能和易用性。

综上所述,USB 2.0 规范速度快,支持多个设备的同时连接,而且具有真正的"即插即用"特性,受到了用户和外设厂家的普遍青睐。目前的 PC 机大多配备了 USB 2.0 功能。相信在不久的将来,USB 2.0 必将在 PC 机应用的各个领域得到广泛的应用。

参 考 文 献

1 Universal Serial Bus Specification (Revision 2.0). www.usb.org,2000

2 胡小军,季军杰. 通用串口总线 (USB) 性能特点及其应用. 微机发展,2001;(3)

选自《微型机与应用》月刊,2002 年第 7 期

6.12 USB 总线协议信息包分析

江苏扬州市大自然电脑有限公司 (225001) 吕 扬
郑州解放军信息工程大学信息技术学院 (450002) 陈露晨 顾雪琳

通用串行总线 USB (Universal Serial Bus) 并不是一种新的总线标准,而是应用在 PC 领域的新型接口技术。USB 同传统 PC 机 I/O 接口相比,其功用有很大改变。它具有 ONE-SIZE-FITS-ALL、PNP 和节省系统资源等多种优点,并且能优化 PC 机资源配置,提高 I/O 接口的数据传输质量,充分满足了市场对 PC 机输入/输出接口的新需求。随着双层 ATX 主板的盛行和 USB 2.0 标准的推出,USB 获得了广泛支持,更趋向于 HP 模式 (480 Mbps)。USB 已逐渐成为计算机系统连接外围设备的 I/O 接口标准的主流。

开发 USB 产品的关键就在于根据协议实现并扩展其功能。USB 协议于 1994 年 11 月 11 日发表了 USB 0.7 版本,到现在已经发展为 USB 2.0 版本。虽然 USB 发展迅速,但目前关于 USB 协议包深层分析的文章比较少见。本文在描述 USB 协议的基础上,讨论了对 USB 2.0 协议中有关 USB 总线传输信息包的应用。

一、USB 2.0 协议概述

1. USB 系统描述

USB 系统包括 3 部分:USB 设备、USB 主机和 USB 连接。每个 USB 系统有一台 USB 主机,最多可拥有 127 台外部设备。USB 设备可分为 HUB 和功能外设二大类。这些 USB 设备通过 host-scheduled 分享带宽,并遵循 token-based 协议的总线接口标准。USB 连接是指 USB 设备和 USB 主机连接并进行通信的方式。可以将存在于 USB 主机和 USB 设备之间的 USB 数据传输模型描述为一个管道 (pipe)。USB 系统的这种管道有 2 种:stream 和 message。其中,stream 的结构在 USB 中未被定义,而 message 则不同。stream 管道的作用在于可由系统的传输进度动态控制,这样就保证了同步,并防止由于使用握手包 (handshake) 应答信号而造成的硬件缓冲区的欠载或溢出以及由此造成的交换率下降。管道包含数据带宽、传输服务类型和端口特征 (如方向和缓冲区大小) 等信息。多数管道在 USB 设备配置时就形成了。当 USB 设备加电时一个消息管道即默认的控制管道就已经存在,这样可以获得配置的设置、状态和控制信息。

2. USB 总线协议

USB 采用投票式总线结构,并由主机系统的 USB 接口 (即主控器) 按照预定原则发起所有的数据传输。多数总线传输包括 3 种信息包,即令牌包 (token packet)、数据包 (data packet) 和握手包 (handshake packet)。每次数据传输首先由主控器发送 1 个 USB 令牌包。该包包含 PID 类型标志、传输方向 (由主机到终端或由终端到主机)、USB 设备地址和端口地址,然后由数据源传输一个数据包或者指示当前没有数据传输。通常在目的地以“握手包 (handshake)”进行应答,指示传输是否成功。

3. USB 数据传输方式

USB 体系结构包括 4 种基本的数据传输方式。

(1) 同步传输:占用大量 USB 带宽,有严格的时间间隔,又被称为实时流传输。

(2) 控制传输:双向传输,该方式传输数据量较小,但要求交付无损且强调实时效果。USB 系统软件主要用它来查询配置和给 USB 终端发送通用命令。

(3) 中断传输:用于少量的、分散的、不可预知的数据传输,例如数据控制指令、设备状态查询和确认命令。

(4) 批量传输:用于大数据量传送和接收精确度较高的数据,且没有对带宽和时间间隔的要求。

二、USB 总线传输信息包

USB 总线传输包括 3 种信息包:令牌包、数据包和握手包。包格式如图 6.12-1 所示。

	(LSB)			(MSB)	(LSB)		(MSB)	(LSB)	(MSB)
type	令牌包				数据包			握手包	
field	PID	ADDR	ENDP	CRC	PID	DATA	CRC	PID	
bits	8	7	4	5	8	0~8192	16	8	

图 6.12-1 USB 总线传输的 3 种信息包格式

由图 6.12-1 可知,所有的信息包都以 PID 码开始,标志包和数据包都以 CRC 码结束。事实上所有的 USB 包都以同步码(SYNC)开始。全/低速设备使用 8 bit 同步码,高速设备使用 32 bit 同步码。任何同步码的最后 2 bit 是同步码结束标志,也是包标志(PID)开始标志。下面是信息包各字段的详细分析。

1. 包标志 PID(Packet IDentifier)

在每个 USB 包中,PID 紧跟在 SYNC 之后。每个 PID 都包含 4 bit 的包类型和 4 bit 的包检测,如图 6.12-2 所示。

(LSB)							(MSB)
PID_{T0}	PID_{T1}	PID_{T2}	PID_{T3}	PID_{C0}	PID_{C1}	PID_{C2}	PID_{C3}

图 6.12-2 PID 格式

PID 指示包的类型并由此推断出包的格式和应用的错误检测类型。PID 中 4 bit 检测码保证了 PID 解码的可靠性,并可保证将该包的剩余部分正确译码。PID 中的检测字段在包类型字段的基础上产生,所以检测码一定是各自包类型标志的补充。否则,该 PID 一定是出错了。

主机和所有的功能终端必须能够完全解读所有接收到的 PID。若接收到的 PID 有 1 个错误检测或者得到的是没有意义的解码就被认为是错误 PID,然后就和包的剩余部分一起被接收者忽略。如果功能终端接收到它并不支持的 PID 有效传输数据或描述字,则该功能终端将被禁止响应。例如,IN-only 功能终端就不能响应 OUT 信号。PID 的类型、编码和描述字如表 6.12-1 所列。

表 6.12-1 包标志 PID 类型

PID Type	PID Name	PID (0～3)	描述字
Token	OUT IN SOF SETUP	0001B 1001B 0101B 1101B	传输方向为由主机到终端的地址字段数 传输方向为由终端到主机的地址字段数 帧开始标记和帧序号 用于设置控制管道的传输方向为由主机到终端的地址字段数
Data	DATA0 DATA1 DATA2 MDATA	0011B 1011B 0111B 1111B	数据包 PID 偶数 数据包 PID 奇数 微帧的数据包 PID 高速宽频同步传输 数据包 PID 高速分解宽频同步传输
Handshake	ACK NAK STALL NYET	0010B 1010B 1110B 0110B	接收方接收到无误差数据包 接收设备不能接收数据或发送设备不能发送数据 连接点中断或控制管道请求没有响应 接收方仍无响应
Special	PRE ERR SPLIT PING Reserved	1100B 1100B 1000B 0100B 0000B	Host-issued 前同步码标志,允许下行总线连接到低速设备 分解传输错误信息握手包 高速分解传输标志 令牌部分的大量 (控制) 功能终端的高速信息流探试 保留

2. 地址字段 (address field)

功能终端的访问地址包括 2 个字段 功能地址字段 ADDR (function address field) 和功能终端字段 ENDP (endpoint field)。每个功能终端既要译地址码又要译终端码。地址码和终端码不能混淆,否则可能使 2 段码字不匹配,或是访问未初始化的终端,都将导致这一帧被忽略。

(1) 功能地址字段 ADDR

功能地址字段 ADDR 的作用是通过地址码的设定来指定功能终端的功能。而功能终端是源点还是目的取决于 PID 标志的值。完整的 128 个地址由 ADDR (0～6) 指示,如图 6.12－3 所示。ADDR 字段规定了 IN、SETUP、OUT 等包标志和 PING、SPLIT 等特殊标志,这样每个 ADDR 值标志一个功能终端。一旦重新设置或功能加强,终端的编码地址将被默认为零值,这时需要主机重新编码。零地址是保留的默认地址,不能分派其他用途。

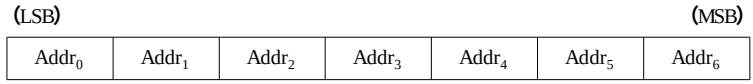


图 6.12－3 功能地址字段 ADDR 格式

(2) 功能终端字段 ENDP

附加 4 bit 的功能终端字段可以在安装多个功能终端时灵活编址,如图 6.12－4 所示。除

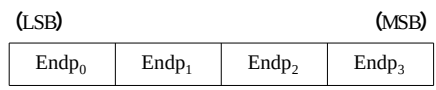


图 6.14－4 功能终端字段格式

零地址外,连接点号码与功能终端明确对应。连接点地址定义了 IN、SETUP、OUT 等标志和 PING 这个特殊标志。所有的功能终端码必须支持零值连接点 (默认值) 的数据控制通道。每

个低速功能终端最多支持 3 个数据通道：1 个零值连接点控制通道和 2 个附加通道（可以是 2 个控制通道，或者是 1 个控制通道与 1 个中断连接点，或是 2 个中断连接点）。全/高速功能终端最多支持 16 个 IN 和 OUT 连接点。

3. 数据字段 DATA

数据字段的范围为 0～1024 字节，而且必须是整型。每个字节的比特流从 LSB 开始发送，如图 6.12－5 所示。

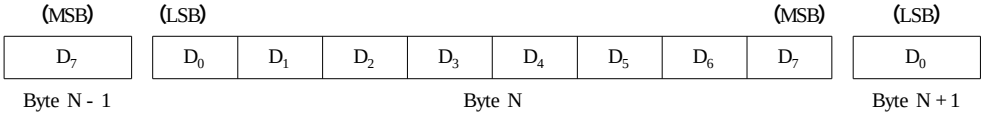


图 6.12－5 数据字段 DATA 格式

4. 循环冗余校验 CRC（Cyclic Redundancy Checks）

CRC 用于令牌包和数据包中除 PID 字段外其他部分的校验。发送端在各个字段进行位填充之前就产生了 CRC 码，相应地，接收端在填充位删除之前就要译出 CRC 码。令牌包和数据包的循环冗余校验码 CRC 能够校验出单、双位错误。若循环冗余校验失败，则接收端认为该包的部分字段已损坏，将忽略这些字段甚至整个包。

三、结束语

PC 技术日渐成熟，外设产品极大丰富，商用和家庭用户的需求加大，这些都使 USB 系统的研究成为今后研究工作的重点。新的 USB 标准规范已在酝酿之中，相信它将赋予 USB 系统更快的传输速度、更强大的功能。

参 考 文 献

1 Universal Bus Specification Revision2.0. Hewlett-packard, Intel, Lucent, Microsoft, NEC, Philips. 2000

2 Http://www.Usb.org

选自《微型机与应用》月刊，2002 年第 1 期

6.13 USB 设备的开发

北京理工大学自动控制系 (100081) 王 洪
中国航天科工集团第二研究院 706 所 (100854) 顾本斗

一、引 言

通用串行总线 USB (Universal Serial Bus),是一种快速、灵活的总线接口。与其他通信接口比较,USB 接口的最大特点是易于使用,这也是 USB 的主要设计目标。作为一种高速总线接口,USB 接口适用于多种设备,比如数码相机、MP3 播放器、高速数据采集设备等。易于使用还表现在 USB 接口支持热拔插,并且所有的配置过程都由系统自动完成,无需用户干预。

从表 6.13-1 可知,数据传输速率高是 USB 接口的另一特点。USB 接口支持 1.5 Mbps (低速)、12 Mbps (全速) 和高达 480 Mbps (USB 2.0 规范) 的数据传输速率,扣除用于总线状态、控制和错误监测等的数据传输,USB 的最大理论传输速率仍高达 1.2 Mbps 或 9.6 Mbps,远高于一般的串行总线接口。

表 6.13-1 常用串行接口比较[1]

接口	格式	负载能力	速率 (最大值) /bps
USB	异步串行	127	1.5M、12M、480M
RS-232	异步串行	2	115.2K
RS-485	异步串行	32	10M
IrDA	红外异步串行	2	115.2K
Microwire	同步串行	8	2M
SPI	同步串行	8	2.1M
I^2C	同步串行	40	400K
IEEE-1394	串行	64	400M
IEEE-488	串行	15	8M
以太网	串行	1024	10M/100M/1G
MIDI	电流环	2	31.5K

USB 接口芯片价格低廉,一个支持 USB 1.1 规范的 USB 接口芯片价格在 $ 1-2 之间,跟一个 232 或 485 接口芯片价格差不多,这也大大促进 USB 设备的开发与应用。

虽然 USB 接口有如此多的优点,并且 586 以上的计算机上都配置有 USB 端口,但由于其接口协议复杂,而目前性能完善、使用方便、价格低廉的接口芯片还没有,导致其开发困难,限制了 USB 的应用。

二、USB 设备的一般开发过程

1. 选定 USB 控制器

在进行一个 USB 设备开发之前,首先要根据具体使用要求选择合适的 USB 控制器。目前,市场上供应的 USB 控制器主要有两种:带 USB 接口的单片机(MCU)和纯粹的 USB 接口芯片。

带 USB 接口的单片机从应用上又可以分成两类:一类是从底层设计专用于 USB 控制的单片机,比如 Cypress 公司的 CY7C63513(低速)、CY7C64013(全速),但由于价格、开发工具以及单片机性能限制等问题,所以一般不推荐选用;另一类是增加了 USB 接口的普通单片机,例如 Intel 公司的 8X931(基于 8051)、8X930(基于高速、增强的 8051)、Cypress 公司的 EZ-USB(基于 8051),选择这类 USB 控制器的最大好处在于开发者对系统结构和指令集非常熟悉,开发工具简单,但对于简单或低成本系统,价格高将会是最大的障碍。

纯粹的 USB 接口芯片仅处理 USB 通信,必须有一个外部微处理器来进行协议处理和数据交换。典型产品有 Philips 公司的 PDIUSBD11(I2C 接口)、PDIUSBD12(并行接口),NS 公司的 USBN9603/9604(并行接口),NetChip 公司的 NET2888 等。USB 接口芯片的主要特点是价格便宜、接口方便、可靠性高,尤其适合于产品的改型设计(硬件上仅需对并行总线和中断进行改动,软件则需要增加微处理器的 USB 中断处理和数据交换程序、PC 机的 USB 接口通信程序,无需对原有产品系统结构作很大的改动)。

2. 硬件电路设计

在选定 USB 控制器以后,如果是带 USB 接口的单片机,则是一般单片机应用系统的开发;反之,就是如何把 USB 接口芯片与单片机应用系统融合的问题,一般 USB 接口芯片都支持多种并行总线结构(复用/非复用),可以方便地与多种单片机接口。硬件设计中要注意的就是 USB 接口芯片的时钟速度比较高,如果芯片内部没有 PLL 来倍频,则外部晶体振荡电路(多数在 48 MHz)的设计就应该特别注意,包括晶体的选择(负载电容大小)、匹配网络的设计以及 PCB 布线。

3. 软件设计

USB 设备的软件设计主要包括两部分:一是 USB 设备端的单片机软件,主要完成 USB 协议处理与数据交换(多数情况下是一个中断子程序)以及其他应用功能程序(比如 A/D 转换、MP3 解码等);二是 PC 端的程序由 USB 通信程序和用户服务程序两部分组成,用户服务程序通过 USB 通信程序与系统 USBDI(USB Device Interface)通信,由系统完成 USB 协议的处理与数据传输。PC 端程序的开发难度非常大,程序员不仅要熟悉 USB 协议,还要熟悉 WINDOWS 体系结构并能熟练运用 DDK 工具。

USB 接口软件主要完成 USB 协议的处理和数据的交换,一定要严格遵循 USB 2.0 规范第 9 章的规定(详见 Universal Serial Bus Specification Revision 2.0:Chapter 9. USB Device Framework www.usb.org)。

4. 调 试

要快捷、成功地开发一个 USB 设备,正确、合理的调试方法是必不可少的环节。调试基本分三步进行:首先对外部设备(单片机部分)借助 PC 调试软件(芯片生产商提供或从网上下载 WINRT-USB、KernelDriver 等调试软件)将设备端的 USB 协议(主要有描述符请求、端口配置、

地址设置以及基本数据交换) 调通 ;然后,用调试好的 USB 设备接口来开发、调试 PC 软件,这一步相对比较容易 ;最后,加上 USB 设备端的其他用户程序,对整个完整的系统进行系统调试。

三、应用实例

下面从硬件、软件两方面具体介绍作者设计的一个便携式 USB 数据采集设备,重点介绍其 USB 接口部分。该数据采集系统以 AD 公司的带 8 通道 12 位 A/D、2 路 12 位 D/A 的 52 内核单片机 ADuC812 作为系统控制器,采用 Philips 公司的 PDIUSBD12 作为 USB 接口芯片。USB 接口电路原理图于图 6.13 - 1 所示。由于系统中需要断电后保存采集数据,扩展了两片

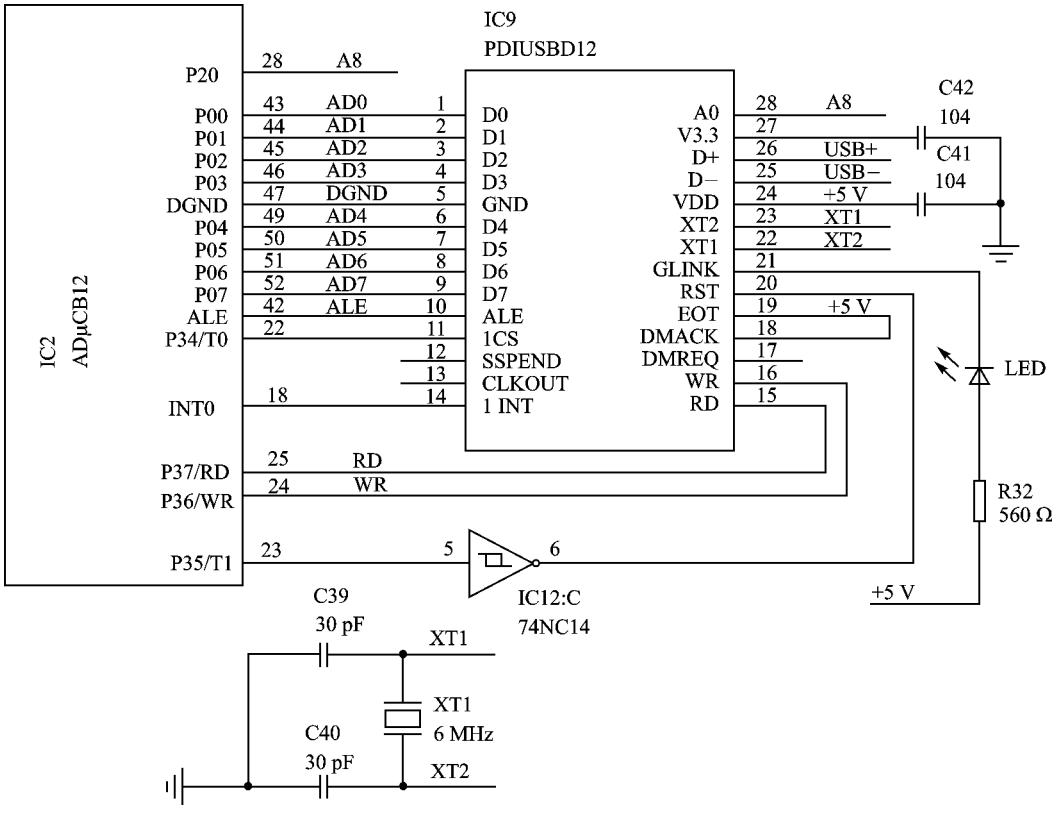


图 6.13 - 1 USB 数据采集系统 USB 接口电路原理图

28F040,因此在这里把 PDIUSBD12 与 CPU 的接口采用了总线复用方式,通过 ALE 信号把数据分离出来,并把低 64K RAM 空间全留给 PDIUSBD12 (ADuC812 的 RAM 空间有 1M,分页管理,每页 64K,共 256 页,对应 DPP 寄存器值 0 ~ 255,PDIUSBD12 占第 0 页,即 DPP = 0),地址线 A8 (P2.0) 作为 PDIUSBD12 的指令/数据选择线,则地址 000100H 写指令、000000H 读写数据 ;单片机的 P3.5 口线提供 PDIUSBD12 的复位信号,接非门是保证单片机复位时 PDIUSBD12 也复位。PDIUSBD12 与单片机的数据交换采用中断方式 (INT0),实际应用中如果系统中断资源不够 (特点是系统改型设计时),也可以接成查询方式,只是注意查询间隔不要超过 USB 接口的最大等待时间 (最大 50 ms[2])。PDIUSBD12 的 GOOD_LINK 指示灯 (LED) 在 USB 通信时会

闪烁,常亮或一直不亮说明 USB 接口有问题,调试时非常有用。PDIUSBD12 采用 PLL 倍频产生系统时钟,只需外接低频晶体,PCB 设计比较方便。单片机软件设计主要注意以下几点:

(1) PDIUSBD12 的中断输出引脚只要中断寄存器不为 0 就保持低电平,所以单片机的对应中断 (INT0) 应设置成电平触发;中断处理后要用读上次传输状态寄存器清除中断寄存器中对应位 (D0 ~ D5)。

(2) PDIUSBD12 靠软件控制 USB 端口的连接,程序在系统初始化处理完后软件设置连接到 USB 端口,然后开中断。

(3) PDIUSBD12 对内部寄存器的读写没有边界限制,程序设计中一定不要读写超过端点深度的数据。特别对于描述符请求,由于其长度大于 Control IN 深度 (16 Bytes),要分几个数据周期传输。

(4) 描述符一定要设置正确,并且注意 USB 协议中所有字数据均定义为低字节在前传输 (LSB),例如 Phlips 的 ID 为 471H,应在 iDVendor 中定义成 71H、04H。

(5) 在接收到 Setup 包后,一定要用 ACK Setup 指令来重新使能 Control IN 和 Control OUT 端点。向 IN 端点写数据后,要用 Validate Buffer 指令使数据可以在下一个 IN 数据周期发送。从 OUT 端点读数据后,要用 Clear Buffer 指令来清空缓冲区,否则后面 OUT 周期传输的数据将被丢弃 (返回 NAK)。

(6) 协议的处理一定要按 USB 规范要求进行,对无效请求,用 Set Endpoint Status 指令将 Control IN 和 Control OUT 端点 Stall 即可。

PC 机软件作者用 VC6.0 开发,分 USB 接口通信程序和应用程序两部分,其开发以及系统调试过程与前文所述相同,此处不再复述。

参 考 文 献

1 Axelson J. USB Complete [J]. Lakeview Research, 1999

2 Anderson D. Universal Serial Bus System Architecture [M]. Addison Wesley Longman, Inc, 2000

3 Analog devices [S]. ADuC812 User s Manual, 2000

选自《计算机工程与设计》月刊,2002 年第 3 期

6.14 嵌入式系统中 USB 总线驱动的开发及应用

北京工业大学电子工程系 (100022) 邵同震 孙光民

一、引 言

USB (Universal Serial Bus) 是一种新型的通用串行总线。它具有即插即用、可热插拔和传输速率快等特点,使得支持 USB 技术的产品和设备越来越多,在工业界已经获得了广泛的支持和应用。目前一般的 PC 机、笔记本电脑等都为用户提供了 USB 接口,并且 Windows 和 Linux 等流行操作系统都支持 USB 协议。随着国内外嵌入式产品开发和应用的深入,要求许多设备能作为 USB 主机支持 USB 协议,比如掌上电脑、PDA、机顶盒 (Set-top) 等要求本身能作为一个 USB 主机支持各种 USB 接口的外设,提供便捷和可靠的服务。这样,USB 的协议栈和总线驱动的开发就变得必不可少,特别是嵌入式设备的实时、小巧等特性使得它的设计显得尤为重要,因为它的好坏会直接对 USB 主机产生影响,从而会对嵌入式系统性能和稳定产生较大影响。下面笔者结合在嵌入式设备中开发基于 USB 协议 1.1 版软件系统的实践,对 USB 主机软件系统中 USB 总线驱动的开发进行介绍。

二、USB 系统组成

USB 系统组成如图 6.14-1 所示,由软、硬件两大部分组成。

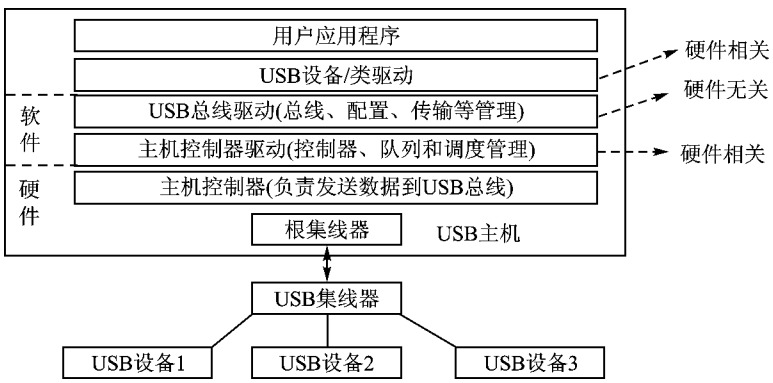


图 6.14-1 USB 系统组成

USB 软件系统由 USB 主机中的软件和 USB 设备中的固件构成。USB 主机中的软件则主要由 USB 设备驱动 USBDD、USB 总线驱动 USBD 和 USB 主机控制器驱动 HCD (Host Control Driver) 三部分构成。

处于最底层的是主机控制器驱动软件 (HCD),它负责对主机控制器进行抽象和对 USB 的低级支持。目前 USB 支持两种主机控制器接口:通用主机控制器接口 (UHCI) 和开放主机控制器接口 (OHCI),两者具有相同的处理能力。主机控制器驱动并为在上层的 USB 系统软件

提供统一的软件接口来控制不同的主机控制器，使得 USB 总线驱动可不用理会各种主机控制器的硬件差异和细节。它直接控制主机控制器的可操作寄存器来管理主机，并且为硬件中断提供中断服务处理例程。此外，主机控制器负责分配端点及传输带宽、数据打包等。

处于最顶层的是 USB 设备驱动软件，它指某支持特定设备类的驱动，负责与其对应的 USB 设备进行通信和读写控制，实现各个 USB 设备特殊的功能应用。

在主机控制器驱动和 USB 设备驱动之间的模块，我们在此称之为 USB 总线驱动。它对应于 USB 协议的 USBD，在 Windows 系统中由 USBD. SYS 模块提供，在 Linux 系统中由 Usbcore. o 模块提供。USB 总线驱动是指在某一操作系统上对 USB 总线和协议提供支持的软件，独立于 USB 设备和 USB 设备驱动软件，并对它们进行控制和提供统一编程接口。USB 总线驱动对协议规定的 USB 总线以及 USB 设备共有的操作和性质提供支持，如设备插入和拔出的动态处理、地址分配、配置、数据传输、供电管理、设备请求处理等。具体包括：检测 USB 设备的插入和拔出；管理主机与设备之间的数据流，对设备进行必要的控制；收集各种状态信息并进行处理；负责各种类型的数据传输；对插入的设备供电等。

三、USB 总线驱动设计

USB 总线驱动设计主要包括五部分，分别是对 USB 设备驱动和应用向上提供的函数接口 USBD API、对主机控制器驱动向下提供的函数接口 HCD、API、USB 系统资源、集线器驱动、系统配置及总线枚举器（见图 6. 14 - 2）。定义好这些接口之后，后三部分可进行并行设计和开发。

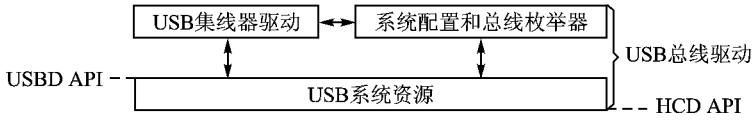


图 6. 14 - 2 USB 总线驱动逻辑结构图

目前嵌入式系统中软硬件产品种类很多，由于本文设计的 USB 总线驱动与 USB 设备和 USB 主机之间通过定义的标准软件接口，对 USB 设备和 USB 主机的操作则分别通过各自的驱动来完成，从而避免了与硬件直接打交道，所以这部分的设计与硬件和操作系统的相关性不大，适用于各种不同的系统。

1. HCD API 设计

在我们的设计中，USB 主机控制器驱动对上层模块只提供一个访问点，这样做的好处就是 USB 总线驱动和主机控制器驱动之间的接口非常简单，如果系统选择不同的主机控制器，对于 USB 总线驱动和上层的驱动与应用没有影响，只要修改主机控制器驱动就足够。因此会减少移植该系统到不同的硬件平台中的难度和复杂度。这个访问点在以下结构 USBD_HCD_operations 中体现，它在 Usbd. h 文件中定义。

```
struct USBD_HCD_operations
 {
(* HCD_allocate) (struct usb_device * dev);                    //分配 USB 设备资源
(* HCD_deallocate) (struct usb_device * dev);                  //收回 USB 设备资源
(* HCD_alloc_bandwidth) (stuct usb_device * dev, S32 endpoint);  //分配带宽
```

```
(* HCD_get_frame_number) (void);                    //得到帧号
(* HCD_submit_urb) (struct urb * purb);            // 提交 USB 请求块
(* HCD_abort_urb) (struct urb * purb);             //放弃 USB 请求块
(* HCD_suspend_bus) (void);                        //总线挂起
(* HCD_resume_bus) (void);                         //总线恢复
(* HCD_adjust_bus_time) (S32 bits);                // 调整总线时间
}
```

USB 总线驱动只要维护这样一个结构的全局变量,就可以得到主机控制器驱动全部的服务。USB 总线驱动在初始化时,将 HCD 对应的变量指针赋给它。需要某项服务时读取该结构变量相应的函数指针,总线驱动就让主机控制器驱动将控制权和服务交给该函数来完成,它可以完成所有底层的操作。

2. USB 系统资源

USB 总线驱动为系统和 USB 设备驱动提供统一的资源服务。其主要包括基本的数据结构和全局变量。涉及的全局变量有 USB 设备/类驱动队列指针、已注册的设备驱动数、指向由主机控制器提供的服务例程的入口点指针和一些信号量。

在驱动程序中都用一个结构体类型 (Struct) 来表示一种设备,具体的结构变量都代表具体设备,这些变量存放了与设备有关的所有信息,它们都在头文件 Usb. h 中定义。图 6.14-3 为 USB 总线驱动中定义的重要数据结构和彼此之间的关系。

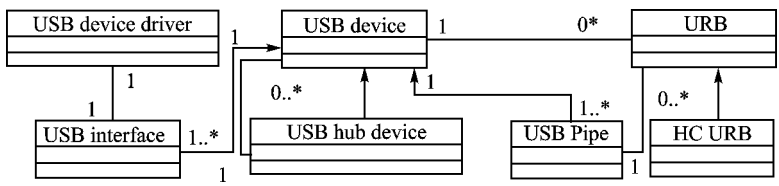


图 6.14-3 USB 总线驱动基本数据结构的关系图

(1) Usb_driver 结构。USB 总线驱动要维护一系列 USB 设备类驱动的数据结构 Usb_driver,如下所示。Driver_name 为该驱动的名称,Probe () 和 Disconnect () 是 USB 设备/类驱动的两个函数入口点。当设备连上或断开总线时,这两个函数分别被设备驱动调用。所有的 USB 设备驱动都应该实现这两个函数。

```
struct usb_driver                    //USB 设备驱动结构
{
driver_name;                         //驱动名称
probe ();                            //设备连接调用时函数入口点
disconnect ();                       //设备断开时函数入口点
}
```

(2) Usb_device 结构。该结构包含一个 USB 设备的基本信息,并且 USBD 把这些 Usb_device 结构连成一个树机构,它代表总线上 USB 设备的逻辑拓扑。它们实现各种功能,但对主机都提供一个同样的接口。USB 中每个功能设备必须有自己的配置信息,描述自身的功能和资源要求。在一个 USB 设备被启用之前,主机必须先对它进行配置,它只有在被主机承认并配置之后才可以进入系统工作。任何时候 USB 设备连上 USB 总线时,主机会给它分配一个

Usb_device 结构,而且该新设备的基本信息会存入该设备。

(3) Usb_interface 结构。USB 设备可以包含多个接口,它能基于一个设备操作,也能基于一个接口操作,这是 USB 的一个重要特性。所以,软件开发者能够开发出基于厂商的驱动或者基于某一设备或接口的类驱动。如音频接口、Modern 接口等都可以有自己的接口驱动。设计 Usb_interface 结构就是要维护这个信息,它包含一个确定的驱动信息和从接口描述符的基本信息。

(4) Usb_pipe 结构。管道 (Pipe) 是主机上的软件和设备端点之间的逻辑连接,它在设备插入、系统对设备供电后立刻建立。主机与它通信,对设备进行配置。在设备正常工作时,主机用它对设备进行一些基本的控制。每条管道和端点的特性有直接关系,它只能支持一种通信方式,缺省管道 (Default Pipe) 是主机上 USB 系统软件和设备端点 0 之间连接。管道的基本信息包含在 Usb_pipe 结构中。

(5) URB 结构。USB 结构系统包含四种基本的数据传输类型:等时传输方式 (Isochronous)、中断传输方式 (Interrupt)、控制传输方式 (Control) 和批 (Bulk) 传输方式。为了给设备/类驱动完成 USB 传输请求提供统一接口,使用一种称为 URB (USB Request Block) 的数据结构,如下所示。该结构含有所有建立 USB 传输类型的参数。

```
struct urb        //USB 请求块结构:
{
stuct urb * next;                    /* 指向下一个 urb 的指针 */
struct usb_device * dev;             /* 指向相关的 usb 设备 */
void phc;                            /* HCD 使用的指针 */
struct usb_pipe pipe;                /* 与该请求相关的管道 (pipe) 信息 */
int flags;                           /* 数据传输类型标志 */
void * buffer;                       /* 传输的数据缓冲区 */
unsigned long int buf_len;           /* 数据缓冲区的长度 */
unsigned long int actual_len;        /* 实际传输的字节数 */
unsigned short int start_frame;      /* 用于该传输的起始帧 */
}
```

除了以上介绍的基本结构之外,Usbd. h 文件还定义了设备、配置、接口、端点和字符串的描述符来管理它们的基本信息,可直接根据规范进行定义,在此不再赘述。

3. USBD 函数接口

USB 总线驱动必须为 USB 设备驱动和上层应用提供统一的编程接口 USBD API,包括以下几方面,本文只给出函数用途和实现方法,不涉及具体细节和过程。

(1) URB 设备驱动注册例程。系统分别为 USB 设备驱动和主机控制器驱动注册提供函数接口。

· USBD_driver_register ()。由于插上一个新设备,USB 总线驱动需要为它找到一个匹配的设备/类驱动。这样所有的 USB 设备/类驱动在生效之前,都需要进行注册,以便 USB 总线驱动能够找到。USB 总线驱动需要提供这样一个函数负责将一个新的 USB 设备/类驱动注册到 USB 总线驱动程序中。

· HCD_driver_register ()。该函数用于主机控制器驱动给 USB 总线驱动提供服务接

入点。

(2) USB 总线操作例程。USB 总线支持挂起 (Suspend) 和恢复 (Resume) 操作。挂起时主机进入节能状态,恢复时主机回到正常模式。这些函数可以直接调用 USBD_HCD_operations 结构中提供的相应函数来完成。

· USBD_suspend_bus ()。该函数用于挂起 USB 总线活动,主机进入节能状态。

· USBD_resume_bus ()。该函数用于恢复已经挂起的 USB 总线,使主机进入工作状态。

· USBD_suspent_device ()。该函数用于 USB 总线驱动,通过总线操作例程将 USB 设备挂起,并调用标准设备请求例程 SET_FEATURE () 函数实现。

(3) 标准 USB 请求例程。USB 协议定义了一系列的设备标准请求,所有设备都支持这些操作请求,作为 USB 总线驱动必须提供这些操作的函数接口。USB 设备都以缺省的控制管道来响应来自主机的请求,通过控制传输来完成,所以标准 USB 设备请求例程都通过调用 USBD_control_transfer () 函数来实现。在此不一一列出。

(4) USB 数据传输例程。所有的 USB 传输请求都通过对 URB 操作来完成。提供 USB 请求块操作的函数有 USBD_submit_urb (),USBD_abord_urb ()。URB 是被异步地递交给 USB 总线驱动,对于中断和异步传输都属于异步方式的传输方式,调用者不必等到这些传输完成,可以在递交数据传输请求之后就继续工作,所以可以直接通过 USBD_submit_urb () 函数来完成。对于异步的 URB,它包含几帧数据,在传输完成之后,URB 中指定的 Complete 函数被唤醒来通知调用者准备发送下一帧。对于中断传输,调用者只需要提交一次中断 USB 请求块操作,USB 主机会不断依据中断端点的轮询间隔来安排中断传输。

· USBD_submit_urb ()。用于高层软件模块异步地发送一个 USB 传输 (Transfer) 请求给 USB 总线驱动,参数 Purb 是事先分配并初始化的 URB 结构的指针。客户软件使用 USBD_submit_urb () 请求从 USB 总线驱动的服务,并且通过一个回调函数 Complete 来通知该请求的完成。

· USBD_abort_urb ()。该函数用于在 USB 传输完成之前取消发送。

(5) USB 同步数据传输例程。由于 URB 是异步地递交给 USB 总线驱动,所以包括控制传输和批量传输在内的同步数据传输例程在调用时有特殊考虑。

· USBD_synch_callback ()。该函数用于同步数据传输完成时的回调函数,主要工作是为通过信号量调用者解锁,可以继续进行下一次数据的传输。

· USBD_control_transfer ()。用于发出控制传输,它们都通过主机控制器驱动的 HCD_summit_urb () 来完成,在请求完成或超时之前一直阻塞发起者。发起者负责检查传输返回的结果。

```
USBD_control_transfer (,,,)
{
struct urb control_urb;
assign memory and initialize for the control_urb;        为结构分配地址并初始化
control_urb. complete = usb_synch_callback;              /* 填充同步回调例程的地址 */
result = USBD_submit_urb (&control_urb);
USB_WAIT_EV (pCltEvent)                                  //等待超时
USB_FREE_EV (pCltEvent);
```

```
return result;
}
```

· USBD_bulk_transfer () 。该函数发出块批量数据传输,和控制传输一样在调用 USBD_submit_urb () 函数之前或者过了预定的时间之前一直锁住,直到本次传输完成。

(6) USB 设备资源管理例程。在 USB 设备从系统中拔出或者没有使用这个设备时,由 USB 设备驱动调用 USBD_release_device () 来释放该 USB 设备的内存等资源。

4. USB 集线器驱动

集线器驱动在文件 Usbd_hub. h 和 Usbd_hub. c 中实现。它主要完成集线器类设备请求,监测管理集线器状态,并将任何状态的改变报告给配置和总线枚举器。它通过以下几方面来完成:

(1) Usbd_hub 结构。USB 集线器是一种特殊的 USB 设备,Usbd_hub 结构包含有一个集线器基本的信息。任何时候 USB 集线器接到 USB 总线,Hub_driver 就会给它分配一个 Usbd_hub 结构,如果 HUB 从总线断开,集线器驱动就会收回它的资源。Has_event 的状态表示在集线器中是否有中断发生,如果有,集线器驱动就会检查该 HUB 的状态来做进一步处理。

```
struct usbd_hub
{
struct usb_device * dev;                    /* 相关的设备 */
boolean has_events;                         /* TRUE:发生中断 FALSE:没有 */
U8 num_ports;                               /* 集线器端口 */
U8 flag;                                    /* USB_HUB_FREE:未用 USB_HUB_USED:使用 */
U8 power_mode;                              /* USB_HUB_SELF_POWER:自供电 */
/* USB_HUB_BUS_POWER:总线供电 */
struct urb irq_urb;                         /* 中断断点的 URB */
}
```

(2) 集线器类请求例程。这些函数提供集线器类请求和服务,它们和其他的 USB 设备标准请求一样通过调用 USBD_control_transfer () 函数来实现。

(3) 除了 Usbd_hub 结构和集线器类请求例程外,集线器驱动还包括一些其他例程。其中,Hub_probe () 是由配置任务调用来监测是否有新设备连接到集线器的;Hub_disconnect () 是当有集线器从系统断开时,该函数被调用释放与集线器相关的资源的;Hub_soft_irq_handler () 是负责集线器驱动的中断处理的,实际上,它只是一个软中断,在集线器的中断端点通知主机有事务发生时,由主机控制器驱动调用的一个回调函数;Usb_hub_power_on () 是通过调用 Usb_set_port_feature () 标准请求函数来给集线器的每个端口供电的;Hub_init () 负责初始化集线器驱动,首先调用 Usb_driver_register () 函数,将 Usbd_hub_driver 结构注册到 USB 系统,并将所有集线器设备状态标为未用状态,同时创建一个 Event 对象;Root_hub_init () 用于产生一个根集线器。

5. USB 系统配置和总线枚举实现

USB 设备可以在任何时候插入或者拔出系统,因此软件系统必须反映物理总线的动态变化并做出相应处理。总线枚举指的就是识别一个插到 USB 总线的设备,并为它分配一个惟一的地址的过程。USB 系统配置包括对设备插入、拔出和总线枚举活动,具体处理过程如图

6.14－4所示。系统配置和总线枚举在文件 Usbd_configuration. h 和 Usbd_configuration. c 中实现。

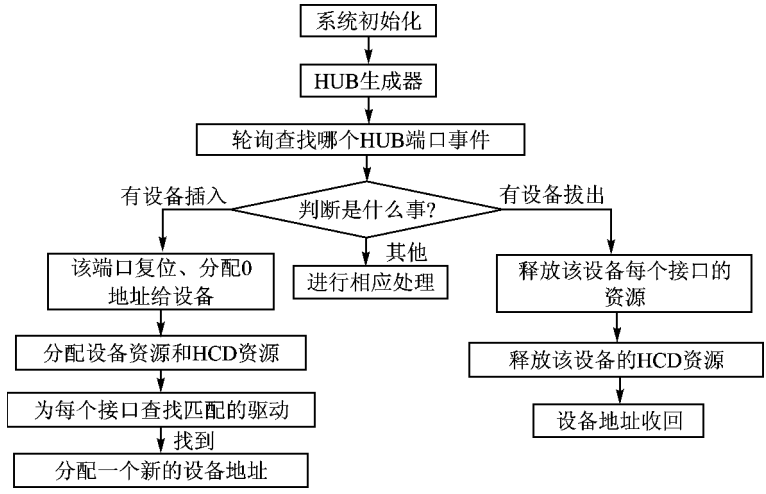


图 6.14－4 设备插入和拔出的处理流程

系统首先进行一系列初始化工作,其中主要包括对主机控制器、根集线器在内的硬件和时钟变量及其他全局变量等。接下来在系统运转之前,必须注册并启动集线器和根集线器的驱动,使之进入工作状态。而所有的 USB 设备/类驱动在起作用之前都应该通过 USB 总线驱动调用 USBD_driver_register () 函数来注册自己,加入到系统中 USB 设备驱动的链表中去,并且同时让系统的设备驱动计数加 1。这样当有设备连接或断开时,USB 系统、主机系统都能从驱动队列中找到相应的驱动。

在系统初始化完成后,启动 Configuration_task () 函数来响应随时可能发生的设备热插拔和总线枚举事件。Configuration_task () 函数是系统配置的主要部分,它是一个无穷循环,负责等待和集线器事务,并且为系统中每个集线器轮询进行处理。进入该函数循环主体之后,主机不停地监听集线器状态,Hub_driver 使用一个信号量来通知配置任务有集线器事务发生。如果监测到有设备插入或者拔出等事件,就会进入相应的函数 Usbd_hub_events () 进行处理。

```
While (true)
 {
     wait for configuration events ;          /* 有某事件发生 */
     clear_congfigEvent () ;                  /* 清除状态标志 */
     usbd_hub_events () ;                     /* 处理这些事件 */
}
```

函数 Usbd_hub_events () 轮询查找系统中的每一个集线器。它通过比较找到发生中断的集线器和相应端口,然后通过集线器驱动部分提供的标准请求得到端口和集线器的状态并进行处理。如果改变的状态表明发生的事情是设备的插入或拔出事件,该函数则通过调用 Usbd_hub_port_connect_change () 函数进行后续处理。

如果有设备插上,调用 Usbd_device_connect () 函数处理。该函数先给端口复位,使之有效,然后继续查询该端口状态,确认该端口可用。如果可用,就为该设备分配一个 Usbd_device

结构单元,并分配缺省的 0 地址。设备以 0 地址从系统的设备驱动库中查找驱动,首先通过一系列的标准设备请求得到设备描述符,并为之分配一个未用的 1 ~ 127 范围的地址和相应的主机控制器驱动资源。然后为每个接口到系统中已经注册过的驱动队列中调用每个设备驱动的 Probe () 函数进行比较,直到找到并返回成功标志 否则,返回失败。

如果是设备拔出则直接调用 Usbd_device_disconnect () 函数进行处理。该函数首先释放所有端口,随后释放接口资源和相关的主机控制器驱动资源,并收回自己的设备地址。

四、结束语

USB 总线标准 1.1 版传输速率为 12 Mbps,升级到 2.0 版后,增至 480 Mbps,且它的介质连接距离也能够支持近百米,能更好地满足包括远端控制、多媒体和视频等在内的实时传输要求。加上 USB 设备的控制、管理和信息交换完全是由系统软件按 USB 协议进行传输的,不占用主机的硬件资源 (如 I/O 地址、内存、中断、DMA 等),具有很高的传输可靠性和兼容性。USB 的这些特性使得它在数字图像、电话语音合成、交互式多媒体、嵌入式系统、消费电子产品等领域将会得到更广泛的应用。

参 考 文 献

1 Chris Chat. Windows WDM 设备驱动程序开发指南 [M]. 北京 机械工业出版社,2000

2 Alessandro Rubini. Linux Device Drivers, 2st Edition [M]. Publisher: O'Reilly, 2001

3 Programming Guide for Linux USB Device Drivers [EB/OL]. http://usb.cs.tum.edu,2002

4 R Dunstan. USB Mobile System Design Guidelines [EB/OL]. Revision 1.0, 1996, http://developer.intel.ru/design/usb/designex/usbgl10.pdf,2002

5 邵高平. 通用串行总线 (USB) 及其开发方法 [J]. 微计算机信息,1999,15 (3): 10 ~ 13

6 张宏伟. Linux 下 USB 设备驱动程序的编写 [J]. 计算机应用研究,2001,18 (9): 141 ~ 146

7 刘少锋,韦克平. USB 软件系统的开发 [J]. 计算机应用研究,2002,19 (3): 102 ~ 104

选自《计算机应用研究》半月刊,2002 年第 12 期

6.15 USB 接口单片机 SL11R 的特点及应用

衡阳南华大学机械学院 (421001) 赵立宏 程品晶

SL11R 是 Scanlogic 公司生产的带有 USB 接口的 16 位 RISC 单片机,内核处理速度达 48MIPS,有很强的控制功能和灵活的工作方式。SL11R 固化有类似于 80X86 的内部 BIOS,可以直接调用,使用非常简单,可以让开发者在很短的时间内完成设计任务。

一、USB 接口简介

USB 总线是通用串行总线 (Universal Serial Bus) 的简称,已经成为 PC 机的标准接口。目前 586 以上的 PC 机基本上都已经配置了 USB 接口。USB 接口具有数据传输速率高、使用方便等特点。USB 1.1 协议规定的全速传输速率为 12 Mbit/s,而 USB 2.0 协议所规定的高速传输速率为 480 Mbit/s,非常适合有大量数据传输的系统。USB 设备即插即用,无需要重新启动计算机。

二、SL11R 介绍

1. SL11R 概述

SL11R 是 Scanlogic 公司的 SL11 产品家庭的一员,是一种带 USB 接口的 16 位单片机,内部运行频率为 48 MHz,采用 RISC 结构,有 16 位数据总线,32 位通用 I/O 口 (GPIO),其中 22 位可作为地址总线进行寻址 (A0 ~ A21),可以直接扩展多种外设。

2. SL11R 主要特点

· USB 接口 SL11R 的 USB 接口符合 USB 1.1 协议,有四个端点 (endpoint),两种数据传输速率,全速模式为 12 Mbit/s,低速模式为 1.5 Mbit/s,并且具有 USB 协议所规定的四种数据传输方式,即控制传输方式 (Control mode)、同步传输方式 (Iochronous mode)、中断传输方式 (Intrerupt mode)、批量传输方式 (Bluk mode)。

· 硬件资源丰富 SL11R 有 3 KB 的内部 RAM、两个定时器、两个外部中断、一个看门狗电路、一个普通串行接口 (UART)、32 位通用可编程 I/O 口 (GPIO)、一个 16 位的可编程 DMA 接口、四个 PWM 输出引脚及扩展外围元件用的控制引脚。SL11R 的外围元件扩展非常方便,扩展 EPROM、串行 E^2PROM、SRAM 即 EDO DRAM 等常用元件时均无需另加控制电路。

· 多种工作方式 SL11R 有四种工作模式,即通用输入输出模式、快速增强并行端口模式、8 位/16 位 DMA 模式、DVC8 位 DMA 模式,可根据实际应用场合用软件进行设置。尤其 8 位 DVC 模式,可以直接与 CCD 接口,方便地开发 CCD 图像采集系统或数码相机。

· 无需专和开发装置 SL11R 内部有 3 K×16 位的程序存储器,类似于 80X86 的 BIOS,已经把单片机的启动配置、联机调试及常用功能等固化在内部,开发者直接调用即可。CPU 复位后,内部 BIOS 会把外部程序存储器中的代码读入内部 RAM 中执行。如果没有外部程序存储器,SL11R 会自动运行在监控状态,与 PC 机进行联机通信,并能够在线对线路板上的串

行 EEPROM 进行编程或直接调试程序。由此可见,SL11R 无需专用开发装置就可以进行开发,这一点对开发者非常有利。由于 SL11R 可以在线编程,这就意味着即使用户也可以进行软件升级。这一点对新产品开发很必要,因为有些 BUG 可能要用户使用后才能发现。

3. SL11R 工作方式简介

· 通用输入输出模式(GPIO 模式) 在这种模式下,SL11R 的外部有 32 个通用输入输出引脚,其中 4 个已经分配给 USB 和 UART 串行接口专用,其他 28 个引脚可以通过软件编程,分别设置成输入或输出状态。这种模式一般用于处理普通的外部并行接口类设备的数据,是用途较广的模式。

· 快速增强并行端口模式(Fast EPP 模式) 快速增强并行端口(Fast EPP)是计算机外设的一种标准并行接口。SLR11R 在这种模式下,可以直接读写快速 EPP 并行增强端口。一般用于 USB 接口和 Fast EPP 接口的转换。

· 8 位/16 位快速 DMA 模式 SL11R 的 DMA 模式包括邮箱协议(Mailbox Protocol)和 DMA 协议两种方式。邮箱协议工作方式允许外部处理器与 SL11R 进行异步通信,它们通过邮箱的输入、输出寄存器交换数据。DMA 协议工作方式一般用于 SL11R 与外部设备大量的数据高速传输。这种传输无需 CPU 的干预,而且允许外设直接与 DRAM 进行数据交换,适合数据量大的场合,如打印机、Modem、扫描仪等。

SL11R 在 DMA 模式下,还有四个可编程的 PWM 输出引脚,可以控制 DMA 模式下的外设与不同的外设通信,如连接 CCD、CIS、COMS 等图像传感器或其他外设。只要根据外设的控制要求对 PWM 编程,就可以控制外设在 DMA 模式下传输数据。

· DVC8 位 DMA 模式 这种模式专门用于与 CCD 相机接口,SL11R 通过串行方式控制 CCD 相机,图像数据以 DMA 的方式传给 SL11R。

三、SL11R 设计应用

1. SL11R 硬件设计

SL11R 的硬件设计比较简单,因为实际应用中一般的外围元件可以直接扩展。Scanlogic 公司在开发套件中提供了一个比较完善的电路图,但对一些简单应用场合显得稍繁琐。事实上 SL11R 的内部有 3K 字节的 RAM,在数据量不是特别大的场合,无需扩展外部数据存储器。图 6.15-1 是笔者设计的 SL11R 应用的一个基本电路,已经在实际项目中应用。

虽然 SL11R 经过编程可以使用 12 MHz 晶振,但调试模式不支持 12 MHz,而且笔者在实际使用过程中发现,如果晶振质量不太好,电路稳定性稍差。故建议在条件许可的情况下,尽量使用 48 MHz 的晶振。

SL11R 的工作电压为 3.3 V,电路中其他元件均应选用低电压型器件。

Scanlogic 公司提供的 SL11R 开发工具中附带有调试程序,在线调试时需要使用 RS232 口,所以电路中设计了 RS232 接口芯片。

2. SL11R 固件设计

SL11R 的固件直接控制 CPU 的运行,程序代码可以存储在外部 EPROM 或 I^2C 串行 EEPROM 中,甚至可以存在主机上,在适当的时候下载到 SL11R 的内部运行。最简单的方式是把代码写到串行 EEPROM 中,因为 SL11R 提供了专用的工具软件可以直接对 EEPROM 在线编程,无需另外的编程装置。

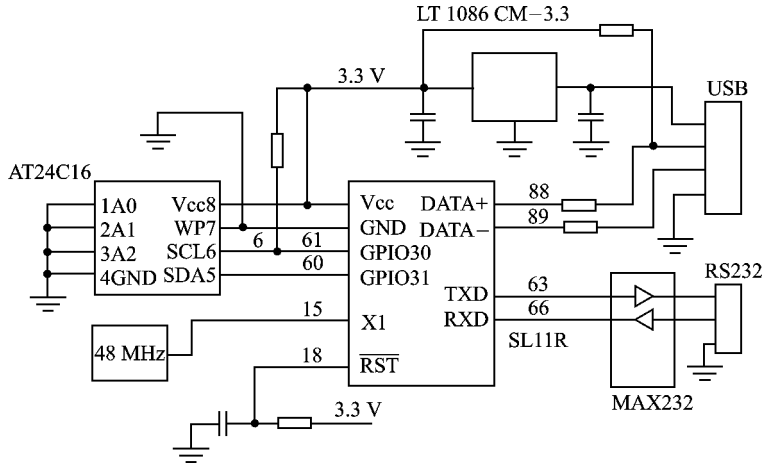


图 6.15－1　SL11R 基本应用电路图

(1) SL11R 程序结构

SL11R 汇编语言的语法结构与 80X86 相似,而且也有内部 BIOS。

MCS51 等没有 BIOS 的单片机,需要开发者控制 CPU 的每一步运行,程序必须在某一段反复循环,程序结构见图 6.15－2。SL11R 由于有 BIOS 支持,它的程序结构就与 MCS51 有所区别。SL11R 的主体循环是在 BIOS 内部,实际上用户程序一般只是 BIOS 的中断响应子程序。也就是说,开发者所编的 SL11R 的用户程序可以没有主循环体。SL11R 的用户程序结构见图 6.15－3。值得指出的是,开发者也可以摆脱 BIOS 的控制,程序不在 BIOS 内部循环。但该程序设计难度较大,因为这时开发者需要自己直接处理 USB 接口的底层软件,一般没有特殊要求不要使用这种方式。

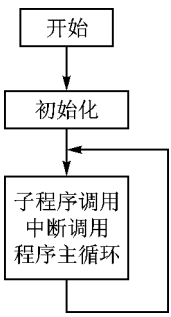


图 6.15－2　MCS51 程序结构

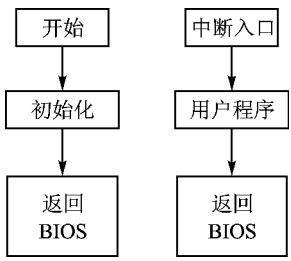


图 6.15－3　SL11R 用户程序结构

(2) SL11R 的 USB 接口编程

SL11R 的大部分功能都可以通过 BIOS 调用实现。USB 的编程也是调用 BIOS 实现。SL11R 复位后会自动配置 USB 接口,与主机建立 USB 通信,一般情况下开发者可以不进行干预。

开发者主要使用的与 USB 有关的 BIOS 软件中断:

· USB_STAND_INT、USB_CLASS_INT、USB_VENDOR_INT、USB_FINISH_INT 这些中断主要是为了实现 USB 通信。其中 USB_VENDOR_INT 是接受主机控制指令中断,主机对 SL11R

的控制基本上都是通过它实现的。

· USB_SEND_INT、USB_RECEIVE_INT 这两个中断用于 USB 发送、接收数据。可以通过配置,分别使用 USB 的四个端口。

3. 主机软件设计

要开发 SL11R 主机软件,一种简单、快捷的方式是借助于 SL11R 开发工具包 (SL11R DVK)。通过学习工具包中附带的例子程序可以在较短的时间内开发出一个应用软件,用户不需自己开发驱动程序。该工具包可以从 ScanLogic 公司的网站 (www. scanlogic. com) 下载得到。工具包中包括的内容有:通用驱动程序、例子程序和开发用的文档资料。对于一般的开发工作可以直接使用工具包中的驱动。用于 Windows 98 系统的三个文件分别是:slusbgen. sys、usb2epp. sys 和 usbdev. inf。

在 AppWizard (zip) 源代码包中,可以找到用 VC 写成的主机例子程序 (usbtest. exe) 的源代码。其中有几个文件对利用 ScanLogic 公司提供的驱动程序来开发应用程序很有帮助。在头文件 slusb. h 中含有如下函数原型:

```
BOOL FAR PASCAL CloseUsbDev (VOID) ;                    //关闭 USB 口
BOOL FAR PASCAL FindUsbDev (WORD wProd) ;               //打开 USB 口
BOOL UsbVendorCmdRead (BYTE bCmd, WORD wValue, WORD wLen, PVOID pData) ;   //发送读控制指令
BOOL UsbVendorCmdWrite (BYTE bCmd, WORD wValue, WORD wLen, PVOID pData) ;  //发送写控制指令
BOOL UsbDataRead (DWORD n,PVOID pData) ;                //读数据块
BOOL UsbDataWrite (DWORD n, PVOID pData) ;              //写数据块
```

只要在应用程序中灵活用好以上几个函数,做一些简单的数据采集工作是完全可行的,笔者就是借用这几个函数在实际工作中成功完成了应用系统的开发。

文件 devioctl. h 包含有 slusb. h 中需要的常数和宏的定义。在文件 usbtest. cpp 中定义了控制 SL11R 操作的指令代码。

SL11R 是一种功能强大的 USB 接口单片机,它的 16 位总线及 DMA 传输模式允许进行大容量数据的高速传输,可以进行高速数据采集。而且 SL11R 的编程简单,无需专用开发装置,开发成本低,初次接触 USB 的开发人员可以很容易实现应用系统的开发。

参 考 文 献

1 SL11R 技术资料. http://www. scanlogic. com
2 Don Anderson. USB 系统体系. 北京:中国电力出版社
3 Jan Axelson. USB 大全. 北京:中国电力出版社
4 张念淮. USB 总线接口开发指南. 北京:国防工业出版社

选自《电子技术应用》月刊,2002 年第 7 期

6.16 USB 接口器件 PDIUSBD12 的接口应用设计

天津大学 王朔 李刚

一、引 言

USB 是近年来应用在 PC 领域的新型接口技术,是一些 PC 大厂商,如 Microsoft、Intel 等为了解决日益增加的 PC 外设与有限的主板插槽和端口之间的矛盾而制定的一种串行通信的标准,自 1995 年在 Comdex 上亮相以来至今已广泛地为各 PC 厂家所支持。现在生产的 PC 几乎都配备了 USB 接口,Microsoft 的 Windows98、NT 以及 MacOS、Linux、FreeBSD 等流行操作系统都增加了对 USB 的支持。

USB 的主要优点如下。

⑴ 使用方便。连接外设不必再打开机箱,允许外设热插拔,而不必关闭主机电源。

⑵ 速度快。USB 接口的最高传输率可达 12 Mb/s,提供低速方式,速率为 1.5 Mb/s。扣除用于总线状态控制和错误检测等数据传输,最大理论速度也能达到 1.2 Mb/s 和9.6 Mb/s。

⑶ 连接灵活。一个 USB 口理论上可以连接 127 个 USB 设备。连接的方式也十分灵活,既可以使用串行连接,也可以使用集线器 Hub,把多个设备连接在一起,再同 PC 机的 USB 口相接。

⑷ 独立供电。USB 接口提供了内置电源。

现在的 USB 生产厂商很多,几乎所有的硬件厂商都有 USB 的产品。USB 控制器一般有两种类型:一种是 MCU 集成在芯片里面的,如 Intel 的 8X930AX、CYPRESS 的 EZ-USB、SIEMENS 的 C541U 以及 MOTOLORA、National Semiconductors 等公司的产品;另一种就是纯粹的 USB 接口芯片,仅处理 USB 通信,如 PHILIPS 的 PDIUSBD11 (I²C 接口)、PDIUSBP11A、PDIUSBD12 (并行接口),National Semiconductor 的 USBN9602、USBN9603、USBN9604 等。前一种由于开发时需要单独的开发系统,因此开发成本较高;而后一种只是一个芯片与 MCU 接口实现 USB 通信功能,因此成本较低,而且可靠性高。本文主要介绍 PHILIPS 公司的 PDIUSBD12 器件。

二、PDIUSBD12 芯片特点和内部结构

PDIUSBD12 是一个性能优化的 USB 器件,通常用于基于微控制器的系统并与微控制器通过高速通用并行接口进行通信,也支持本地 DMA 传输。该器件采用模块化的方法实现一个 USB 接口,允许在众多可用的微控制器中选择最合适的作为系统微控制器,允许使用现存的体系结构并使固件投资减到最小。这种灵活性减少了开发时间、风险和成本,是开发低成本且高效的 USB 外围设备解决方案的一种最快途径。PDIUSBD12 完全符合 USB1.1 规范,也能适应大多数设备类规范的设计,如成像类、大容量存储类、通信类、打印类和人工输入设备等。因此,PDIUSBD12 非常适合做很多外围设备,如打印机、扫描仪、外部大容量存储器 (Zip 驱动器) 和数码相机等。现在用 SCSI 实现的很多设备如果用 USB 来实现可以直接降低成本。

PDIUSBD12 挂起时的低功耗以及 LazyClock 输出符合 ACPI、OnNOW 和 USB 电源管理设备的要求。低功耗工作允许实现总线供电的外围设备。

PDIUSBD12 还集成了像 SoftConnect、GoodLink、可编程时钟输出、低频晶振和终端电阻等特性。所有这些特性都能在系统实现时节省成本,同时在外围设备上很容易实现更高级的 USB 功能。

1. 主要特性

主要特性叙述如下。

⑴ 符合 USB 1.1 协议规范;

⑵ 集成了 SIE、FIFO 存储器、收发器和电压调整器的高性能 USB 接口芯片;

⑶ 适应大多数设备类规范的设计;

⑷ 与任何微控制器/微处理器有高速(2MB/s)的并行接口;

⑸ 完全自动 DMA 操作;

⑹ 集成了 320 B 的多配置 FIFO 存储器;

⑺ 主端点有双缓存配置,增加吞吐量,容易实现实时数据传输;

⑻ 在块传输模式下有 1 MB/s 的数据传输率,在同步传输模式下有 1 Mb/s 的数据传输率;

⑼ 具有总线供电能力,有非常好的 EMI 性能;

⑽ 在挂起时有可控制的 LazyClock 输出;

⑾ 可通过软件控制 USB 总线连接 SoftConnect;

⑿ 在 USB 传输时有闪亮的 USB 连接指示灯 GoodLink;

⒀ 时钟频率输出可编程;

⒁ 符合 ACPI、OnNOW 和 USB 电源管理要求;

⒂ 具有内部上电复位和低电压复位电路;

⒃ 有 SO18 和 TSSOP28 封装;

⒄ 能在 - 40 ~ +85 工业级工作;

⒅ 片内 8 kV 静电保护;

⒆ 双电压工作:(3.3 ±0.3) V 或扩大的 5 V 电压范围(3.6 ~5.5 V);

⒇ 多中断模式,方便块传输和同步传输。

2. 内部结构

PDIUSBD12 的内部结构框图如图 6.16 -1 所示。其结构介绍如下。

⑴ 模拟收发器。集成的收发器直接通过终端电阻与 USB 电缆接口。

⑵ 电压调整器。片上集成的 1 个 3.3 V 电压调整器为模拟收发器供电,也提供连接到外部 1.5 kΩ 上拉电阻的输出电压。作为选择,PDIUSBD12 提供集成 1.5 kΩ 上拉电阻的 SoftConnect 技术。

⑶ PLL。片上集成 1 个 6 ~48 MHz 的倍频 PLL(锁相环),允许使用 6 MHz 的晶振,EMI 也由于使用低频晶振而减小。PLL 的工作不需要外部器件。

⑷ 位时钟恢复。位时钟恢复电路用 4 倍过采样原理从输入的 USB 数据流中恢复时钟,能跟踪 USB 规范中指出的信号抖动和频率漂移。

⑸ PHILIPS 串行接口引擎 PSIE。PHILIPS 的 SIE 完全实现 USB 协议层。考虑到速度,

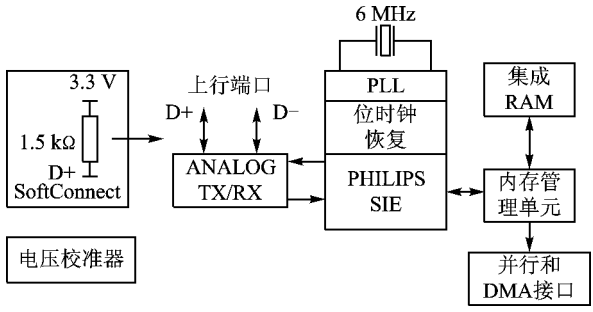


图 6.16－1　PDIUSBD12 内部结构框图

它是全硬件的,不需要固件 (微程序) 介入。这个模块的功能包括 :同步模式识别、并/串转换、位填充/不填充、CRC 校验、PID 确认、地址识别以及握手鉴定。

⑹ SoftConnect。高速设备与 USB 的连接是靠把 D + 通过 1 个 1.5 kΩ 的上拉电阻接到高电平来建立的。在 PDIUSBD12 中,这个上拉电阻是集成在芯片内的,缺省是没有连接到 V_{DD},这个连接是靠外部 MCU 发一个命令来建立的。这使得系统微处理器可以在决定建立 USB 连接之前完成初始化。重新初始化 USB 总线连接也可以不用拔掉电缆来完成。

⑺ GoodLink。GoodLink 是靠一个引脚接发光二极管实现的。在 USB 设备枚举时 LED 指示灯将立即闪亮 ;当 PDIUSBD12 被成功枚举并配置时,LED 指示灯将会始终亮 ;经过 PDIUSBD12 的 USB 数据传输过程中,LED 将一闪一闪,传输成功后 LED 熄灭 ;在挂起期间,LED 熄灭。这种特性可以使我们知道 PDIUSBD12 的状态,方便电路调试。

⑻ 存储器管理单元 MMU 和集成 RAM。MMU 和集成 RAM 能缓冲 USB (工作在 12 Mb/s) 数据传输和微控制器之间并行接口之间的速度差异,这允许微控制器以自己的速度读写 USB 包。

⑼ 并行和 DMA 接口。并行接口容易使用、速度快并且能直接与主微控制器接口。对于微控制器,PDIUSBD12 可以看成是一个有 8 位数据总线和 1 位地址线的存储设备。PDIUSBD12 支持多路复用和非多路复用的地址和数据总线。在主端点 (端点 2) 和局部共享存储器之间也可使用 DMA (直接存储器存取) 传输。它支持单周期模式和块传送模式两种 DMA 传输。

三、PDIUSBD12 的引脚说明及典型连接

1. PDIUSBD12 引脚说明

PDIUSBD12 引脚如图 6.16－2 所示,引脚说明如表 6.16－1 所列。

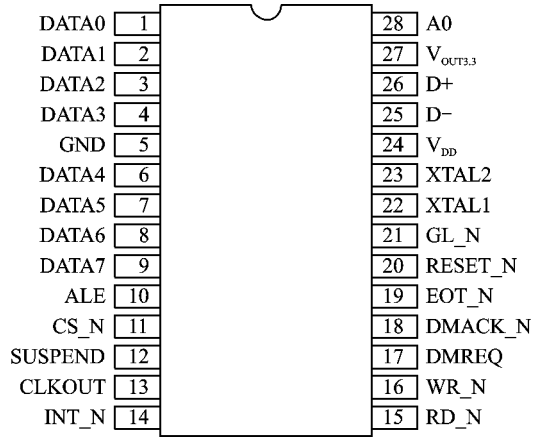


图 6.16－2 PDIUSBD12 **的引脚图**

表 6.16－1 PDIUSBD12 **引脚说明**

引脚号	符 号	说 明
1～4、6～9	DATA0～DATA7	8 位双向数据
5	GND	地
10	ALE	地址锁存允许。在多路复用地址/数据总线时，ALE 下降沿用于锁存地址信息 独立地址/数据总线时将 ALE 永久接地
11	CS_N	片选（低电平有效）
12	SUSPEND	芯片进入挂起状态
13	CLKOUT	可编程时钟输出
14	INT_N	中断输出（低电平有效）
15	RD_N	读选通（低电平有效）
16	WR_N	写选通（低电平有效）
17	DMREQ	DMA 请求
18	DMACK_N	DMA 响应（低电平有效）
19	EOT_N	DMA 传输结束（低电平有效）。另一个功能是 VBUS 感和器
20	RESET_N	复位（低电平有效、异步）。有片内上电复位电路，该引脚可以接高
21	GL_N	GoodLink 发光二极管指示器（低电平有效）
22	XTAL1	晶振连接 1（6 MHz）
23	XTAL2	晶振连接 2（6 MHz）
24	V_{DD}	正电源（4.0～5.5 V）。让芯片工作在 3.3 V，将 3.3 V 电压加到 V_{DD} 和 $V_{OUT3.3}$ 两个引脚上
25	D -	USB D - 数据线
26	D +	USB D + 数据线
27	$V_{OUT3.3}$	3.3 V 输出
28	A0	地址位。A0 = 1 选择命令，A0 = 0 选择数据。在多路复用地址和数据总线配置时，这一位将不考虑，应接高电平

2．PDIUSBD12 的典型连接

PDIUSBD1 与 80C51 的连接电路如图 6.16－3 所示。在这个例子中，ALE 始终接低电平，说明采用单独地址和数据总线配置。A0 脚接 80C51 的任何 I/O 引脚，控制是命令还是数据输入到 PDIUSBD12。80C51 的 P0 口直接与 PDIUSBD12 的数据总线相连接，CLKOUT 时钟输出为 80C51 提供时钟输入。

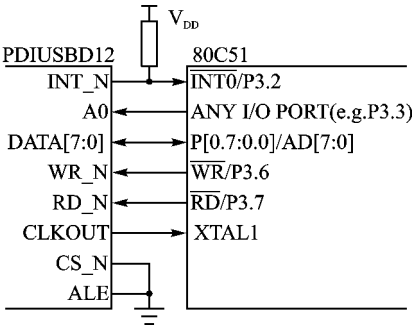


图 6.16－3　PDIUSBD12 与 80C51 的连接电路图

四、软件设计

1．单片机方面软件设计

对于单片机控制程序，目前没有任何厂商提供自动生成固件（firmware）的工具，因此所有程序都要由自己手工编制。USB 单片机控制程序通常由三部分组成：第一、初始化单片机和所有的外围电路（包括 PDIUSBD12）；第二、主循环部分，其任务是可以中断的；第三、中断服务程序，其任务是对时间敏感的，必须马上执行。根据 USB 协议，任何传输都是由主机（host）开始的，这样，单片机作它的前台工作，等待中断。主机首先要发令牌包给 USB 设备（这里是 PDIUSBD12），PDIUSBD12 接收到令牌包后就给单片机发中断，单片机进入中断服务程序，首先读 PDIUSBD12 的中断寄存器，判断 USB 令牌包的类型，然后执行相应的操作。因此，USB 单片机程序主要就是中断服务程序的编写。在 USB 单片机程序中要完成对各种令牌包的响应，其中比较难处理的是 SETUP 包，主要是端口 0 的编程。

单片机与 PDIUSBD12 的通信主要是靠单片机给 PDIUSBD12 发命令和数据来实现的。PDIUSBD12 的命令字分为三种：初始化命令字、数据流命令字和通用命令字。PDIUSBD12 给出了各种命令的代码和地址。单片机先给 PDIUSBD12 的命令地址发命令，根据不同命令的要求再发送或读出不同的数据。因此，可以将每种命令做成函数，用函数实现各个命令，以后直接调用函数即可。

在编写 USB 的单片机程序时，需要注意：

⑴ 单片机的中断应设置为电平触发；中断后一定要读上次传输状态寄存器（命令 40-45H），以清除中断寄存器中的中断标志。这样，PDIUSBD12 的中断输出才能变回高电平，这一点非常重要。

⑵ 在接收到 Setup 包后，一定要调用 ACK setup 命令重新使能端口 0。

⑶ 在向 IN 端点写完数据后，一定调用 Validate Buffer（命令 FAH），指明缓冲区中的数据有效，可以发送到主机。

(4) 当读完数据后,一定调用 Clear Buffer (命令 F2H),以保证可以接收新的包。

(5) 可以通过调用 Read Chip ID (命令 FDH) 检查 PDIUSBD12 是否工作。该命令要读两个字节数据。

USB 初始化过程为:

(1) Set Address Enable;

(2) Set Endpoint Enable (此时 LED 亮);

(3) Disconnect;

(4) delay (1 ~ 2 s);

(5) Connect (即用 43h 参数调用 Set Mode,此时 LED 灭);

(6) Read Interrupt Register。

完成初始化工作后,就可作其他的前台工作了,并在前台判断是否有 Setup 包 (通过一个变量,当中断服务程序检测到有 Setup 包时,设置该变量),然后执行响应的控制传输。

在调试 USB 单片机程序时,还要特别注意 Windows 对 USB 设备的枚举顺序:

(1) GetDeviceDescriptor。主机主要对 Length 域感兴趣,发送内容一定要正确,特别是第 2 字节 type 一定为 0x01,即 Device 否则,主机将不响应,或者再重复 2 次后放弃。可检查一下对 EP0 的 RX、TX 的设置次序。

(2) SetAddress。一般为 02 或 03。

(3) 连续 3 次 GetDeviceDescriptor,读取全部设备描述符,一般为 18 B,分为多次传输。如果不正确,主机将不响应或重复 2 次后放弃。

(4) GetConfigDescriptor。注意第 2 字节一定为 0x02,即 config。

(5) GetStringDescriptor (可能没有),根据在设备描述符中是否有 String 索引而定。一般先读取 LanguageID,再读取 product string。

(6) 读取全部 ConfigDescriptor,次数根据描述符的大小决定 (端点个数不同,描述符大小不同),如果不正确,主机将不响应或再重复 2 次后放弃。

(7) 如果以上步骤都正确,主机将找到新设备,提示安装驱动程序 否则找到未知设备,不可用。安装驱动程序后,以后的每次 PlugIn,枚举次序与以上步骤略有不同,之后会有 SetConfiguration、GetConfiguration 和 GetInterface 等调用。

2. 主机方面软件设计

Windows98 提供了多种 USB 设备的驱动程序,但好像还没有一种是专门针对数据采集系统的,所以必须针对特定的设备来编制驱动程序。尽管系统已经提供了很多标准接口函数,但编制驱动程序仍然是 USB 开发中最困难的一件事情,通常采用 Windows DDK 来实现。目前有许多第三方软件厂商提供了各种各样的生成工具,像 Compuware 的 driver works,Blue Waters 的 Driver Wizard 等,它们能够很容易地在几分钟之内生成高质量的 USB 的驱动程序。作为 WIN98 和 WIN2K 推荐的一项新技术来说,USB 的驱动程序和以往的直接跟硬件打交道的 WIN95 的 VXD 方式的驱动程序不同。它是 WDM 类型的。

在调试 USB 设备时,可使用 UsbView 程序检测设备是否能被 Windows 枚举并配置,如果成功,还可在该程序中查看设备描述符、配置描述符和端点描述符是否正确。之后可以使用 Driver Wizard 生成一个通用驱动程序,在 Windows 提示安装驱动程序时,选择 Driver Wizard 生成的驱动程序。其实 Driver Wizard 生成的仅是一个 Windows 控制台的应用程序,它会调用安

装 Driver Wizard 时安装在系统中的通用 USB 驱动程序。使用该程序就可测试设备是否能够正确传输数据以及传输速度。该程序也可作为最终产品 USB 传输部分的框架 如果不能满足要求,也可用 WDM 重新编制驱动程序,用调试好的 USB 设备来开发、调试主机软件。

五、应用实例

本文介绍一个高速数据采集系统,以 AD 公司的 ADμC812 为系统控制器。该单片机本身就是高度集成的高精度 12 位数据采集系统,在其片内不仅组合了可重新编程非易失性闪速/电擦除程序存储器的高性能 8 位 (与 8051 兼容) MCU,还包含了高性能的自校准多通道 (8 个输入通道) 12 位 ADC 和两个 12 位 DAC,且内核与 8051 指令集兼容。PDIUSBD12 作为 ADμC812 的存储器外设,接口比较简单。需要注意的地方是$\overline{\text{DMACK}}$引脚必须接高电平,否则将不能接收任何命令和数据 ;EOT_N 必须通过电阻接到 USB 的 +5 V,以正确检测到 USB 连接 ;INT_N 引脚加 1 个上拉电阻, +5 V 接到 V_{DD} 引脚 ;在 $V_{OUT3.3}$ 引脚加 1 μF 电解电容和 0.1 μF 两个退耦电容。

综上所述,PDIUSBD12 是一个性能优化的 USB 器件,它的 SoftConnect 和 GoodLink 技术使开发和调试 USB 设备时非常方便,在性能、速度、方便性以及成本上都具有很大的优势。因此,使用 PHILIPS 公司的 PDIUSBD12 可以快速开发出高性能的 USB 设备。

参 考 文 献

1 Philips Corp. PDIUSBD12 Users Manual

2 Universal Serial Bus Specification,Compaq,Intel,Micrisoft,NEC,Revision 1.1 September 23, 1998

3 刘丁,毛德柱,王云飞. USB 在数据采集系统中的应用. 电子技术应用

4 晁建刚,陈善广,薛亮. 基于 USB 接口技术的外设应用设计. 嵌入式系统论文集,2000,11

选自《单片机与嵌入式系统应用》月刊,2002 年第 4 期

6.17 USB 2.0 控制器 CY7C68013 特点与应用

长沙国防科学技术大学 扈 啸 张 玘 张连超

本刊 2002 年第 2、3 期已有对 EZ-USB 单片机的介绍。本文在此只重点介绍 USB 2.0 的特殊之处以及芯片 CY7C68013 的主要特点。

一、USB 2.0 的主要特点

USB 协议的 2.0 版本于 2000 年 4 月推出。支持以下 3 种速度模式：

低速模式 (low speed) 1.5 Mb/s；

全速模式 (full speed) 12 Mb/s；

高速模式 (high speed) 480 Mb/s。

USB2.0 协议支持现存的所有 USB 设备，既可以把 USB1.1 设备插入 USB2.0 的 PC 机接口，也可以把新的 USB2.0 设备插入 USB1.1 的 PC 机接口，并且在电气上兼容 USB1.1 的连接线。

1. 数据包

USB 传输的数据包的类型用称之为 Packet IDs (PIDs) 的特定代码来定义。USB 包中共有 4 种 PID 类型，如表 6.17－1 所列。

表 6.17－1 USB 2.0 的数据包类型

PID 类型	PID 名称
令牌	IN,OUT,SOF,SETUP
数据	DATA0,DATA1,DATA2,MDATA
握手	ACK,NAK,STALL,NYET
特殊类型	PRE,ERR,SPLIT,PIN

注：黑体字表示 USB2.0 增加的 PID 类型。

在全速模式时，每个 OUT 传输发送 OUT 数据包，不考虑外设是否处于“忙”状态而不能接收数据。针对这种浪费带宽的情况，在高速模式时推荐使用新的 PID 类型“PING”。主机先对 OUT 端点发出一个较短的“PING”令牌，询问当前外设是否有数据空间来存放 OUT 数据包。仅仅当外部设备回答“ACK”时，主机才发送较长 OUT 数据包。

SETUP 令牌只用于控制传输。它是数据包中的前 8 个字节。通过这 8 个字节，外设对主机的设备请求进行译码。

SOF 令牌代表一个 USB 帧的开始。

ACK (Acknowlegde) 表示成功，数据接收无误。

NAK (Negavite Acknowlegde) 表示忙，重发。这并不是出错，USB 外设没有应答表示出错。

STALL 表示未知错误，外设未能理解主机发出的设备请求，有可能是外设端出错，或是主

机访问并不存在的资源。USB 协议提供了从 stall 状态恢复的方法。

NYET (Not Yet) 类似于 ACK,表示数据接收无误,并且指出外设还没准备好接收下一个 OUT 数据包。NYET PID 只用于在高速模式。

其他 PID 详见参考文献 [1]。

2. 帧结构

USB 主机每毫秒向所有的 USB 设备发送 1 个 SOF 包 (Start of Frame),以此来提供时间基准。SOF 包括 1 个自增的 11 位帧序号。FX2 随时可以从寄存器中读出这个范围在 [0 ~ 2047] 的帧序号。

在高速模式下 (480 Mb/s),每个 1 ms 长的帧被分成了 8 个 125 μs 长的微帧。每个微帧也都由 1 个 SOF 包开始。帧序号还是每个毫秒自增 1 次,所以这 8 个微帧都含有相同的帧序号。为了区别每个微帧,FX2 提供了 1 个只读的微帧计数器,并且 FX2 能在收到 SOF 包时产生 1 个中断请求,即在全速模式下 1 ms/次,高速模式下 125 μs/次。

3. 传输类型

为了适应 480 Mb/s 的高速数据传输,USB 2.0 协议扩大了各种传输类型数据包的长度,与 USB1.1 的对照如表 6.17 - 2 所列。

表 6.17 - 2　USB 2.0 与 USB1.1 数据包长度的对照

传输类型	数据包长度/B	
	USB1.1	USB2.0
控制传输	8,16,32,64	64
块传输	8,16,32,64	512
中断传输	1 ~ 64	1024
同步传输	1023	1024

4. 高速模式和全速模式的检测

USB 2.0 规范要求高速设备必须能在全速模式下枚举。每个高速设备都在全速模式下开始枚举过程。当与主机达成“Chirp”协议后设备再切换到高速工作模式下。详细内容见参考文献 [1] 第 7 章。FX2 能自动检测高速主机,并切换到高速模式下。

5. 传输性能分析

以 USB 硬盘为例分析 USB 2.0 的高速传输性能。图 6.17 - 1 为 USB 2.0 与硬盘接口的带宽分析。

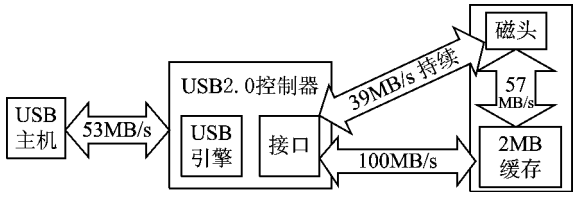


图 6.17 - 1　USB2.0 与硬盘接口的带宽分析

每分转速 7200 带有 2 MB 缓存的 ATA100 硬盘,接口数据传输速率可达 100 MB/s,可持

续的有效传输速率只有 39 MB/s。

USB2.0 在每个微帧中最大可传输 13 个块传输包,而每个微帧长固定为 125 μs,所以其最大传输速率为:512 ×13 ×8 ×1000 = 53 MB/s。

二、EZ-USB FX2 的主要特点

1. 芯片结构

EZ-USB FX2 芯片包括 1 个 8051 处理器、1 个串行接口引擎 (SIE)、1 个 USB 收发器、8.5 KB 片上 RAM、4 KB FIFO 存储器以及 1 个通用可编程接口 (GPIF),如图 6.17 -2 所示。FX2 是一个全面集成的解决方案,它占用更少的电路板空间,并缩短开发时间。

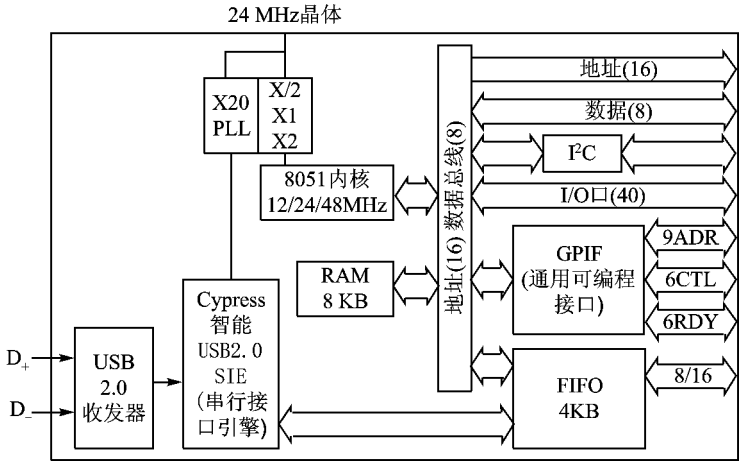


图 6.17 -2 FX2 结构框图

EZ-USB FX2 拥有 1 个独特的架构,其中包括 1 个智能串行接口引擎 (SIE)。它执行所有基本的 USB 功能,将嵌入式 MCU 解放出来以用于实现专用的功能,并保证其持续的高性能的传输速率。FX2 还包括 1 个通用可编程接口 (GPIF),允许它"无胶粘接",即可与任何 ASIC 或 DSP 进行连接,并且它还支持所有通用总线标准,包括 ATA、UTOPIA、EPP 和 PCMCIA。EZ-USB FX2 完全适用于 USB 2.0,并向下兼容 USB1.1。

FX2 有 3 种封装形式:56 脚的 SOPP、100 脚的 TQFP (薄形四方扁平封装)、128 脚的 TQFP。引脚数的区别在于输入、输出引脚数的不同,以针对不同的应用要求。

2. 结构特点

当前大部分 USB1.1 器件都需要微控制器参与数据从端点 FIFOs 到应用环境转移,如图 6.17 -3 所示。显然,微控制器本身的工作频率在相当程度上限制了带宽的进一步提高。虽然在 12 Mb/s 的全速模式下,这种限制并不明显,但当速度提升至 480 Mb/s 时,在成本严格控制下的微控制器就必然成为整个系统的带宽瓶颈。

EZ-USB FX2 提供了一种独特架构,使 USB 接口和应用环境直接共享 FIFO,而微控制器可不参与数据传输但允许以 FIFO 或 RAM 的方式访问这些共享 FIFO,如图 6.17 -4 所示。这种被称之为"量子 FIFO"(Quantum FIFO) 的处理架构,较好地解决了 USB 高速模式的带宽问题。

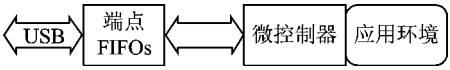


图 6.17－3　USB1.1 数据传输示意图

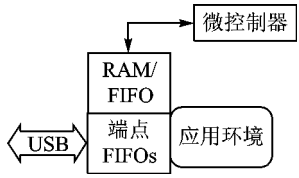


图 6.17－4　FX2 数据传输示意图

具体来说，如图 6.17－5 所示，USB 执行 OUT 传输，将 EP2 端点设成 512 字节四重 FIFO（如二.3 小节所述）。在 USB 端和外部接口端都并不知道有四重 FIFO。看来，USB 端只要有 1 个 FIFO 为“半满”，就可以继续发送数据。当前操作的 FIFO 写“满”时，FX2 自动将其转换到外部接口端，排队等候读取；并将 USB 接口队列中下一个为“空”的 FIFO 转移到 USB 接口上，供其继续写数据。外部接口端与此类似，只要有 1 个 FIFO 为“半满”，就可以继续读取数据。当前操作的 FIFO 读“空”时，FX2 自动将其转换到 USB 接口端，排队等候写入；并将外部接口队列中下一个为“满”的 FIFO 转移到外部接口上，供其继续读取数据。

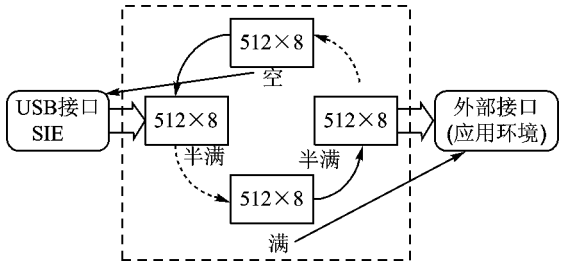


图 6.17－5　FX2 数据传输原理图

3. 端点缓存

USB 协议定义了端点作为数据的接收器和发送器。主机发送 4 个 bit 的地址和 1 个 bit 的方向来选择端点，因此 USB 最多可有 32 个端点定义：IN0～IN15 和 OUT0～OUT15。

FX2 定义了 7 个端点，在高速模式下的端点缓存结构如图 6.17－6 所示。EP0IN&OUT、EP1IN、EP1OUT 是 64byte 的端点缓存。EP0 是默认的控制传输端点，既是 IN 端点也是 OUT 端点。EP1IN、EP1OUT 支持块、中断和同步传输。EP0、EP1IN 和 EP1OUT 只能由 FX2 的固件访问；而 EP2、4、6 和 8 无需固件干涉即可同片外互传高速数据。

FX2 端点配置方式非常灵活。EP2、4、6 和 8 是大容量高带宽的数据传输端点，可设为 IN 或 OUT 端点的一种，能配置成多种形式以适应带宽需要。在图 6.17－6 中，每一列代表 1 种配置方式。带阴影的方框可包括 2、3、4 个 512 或 1024 字节的缓存，分别表示端点可配置成双重、三重和四重缓存。双缓存是指 USB 可以读或写 1 个数据包，而另一个数据包（同一个端点内另一个缓冲存储器中的）可供外部接口操作；三重缓存加了第 3 个数据包存储器可供 USB 和外部接口需要的一方使用；四重缓存增加了第 4 个数据包存储器。多缓存的结构可以在读写双方速度相似时有效地改善带宽，平滑带宽抖动，减少双方的互相等待时间。

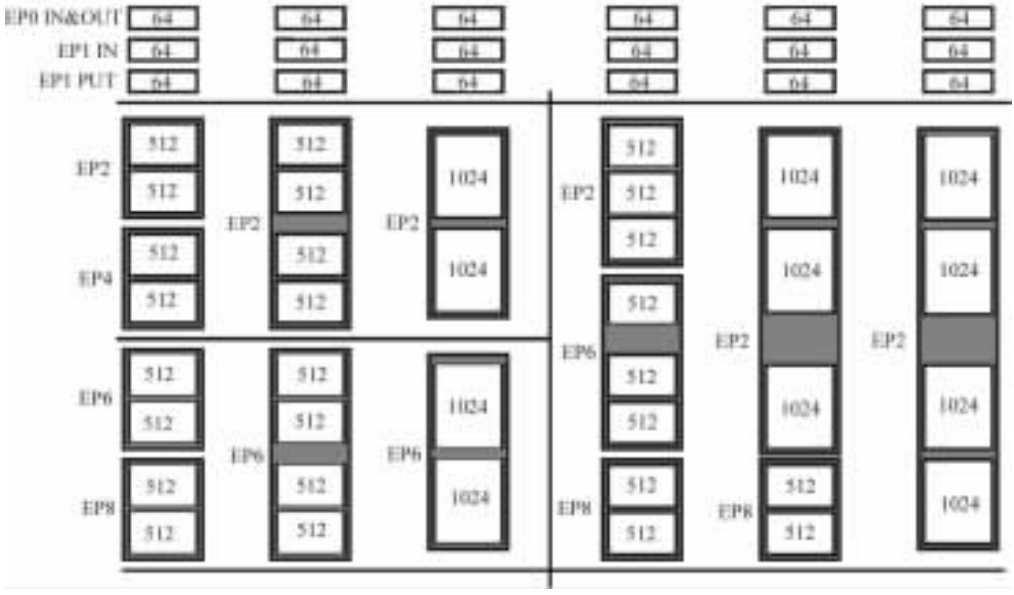


图 6.17－6 FX2 高速模式下的端点缓存

三、EZ-USB FX2 的接口方式

FX 有 2 种接口方式：Slave FIFOs 和可编程接口 GPIF。

Slave FIFOs 方式是从机方式，外部控制器可像普通 FIFO 一样对 FX2 的多层缓冲 FIFO 进行读写。FX2 的 Slave FIFOs 工作方式可设为同步或异步；工作时钟可选为内部产生或外部输入；其他控制信号也可灵活地设置为高有效或低有效。

可编程接口 GPIF 是主机方式，可以软件编程读写控制波形，几乎可以对任何 8/16 bit 接口的控制器、存储器和总线进行数据的主动读写，非常灵活。

四、EZ-USB FX2 的开发工具

如同 EZ-USB 系列的其他控制器一样，Cypress 公司对 FX2 也提供了较为完备的开发套件 CY3681。它包括带有 128 脚 CY7C68013 的硬件开发板和相应的控制面板（control panel）、GPIF 代码自动生成软件（GPIFTool）以及固件方面丰富的例子和大量的帮助文档。可以尽量节省学习时间，加快开发速度。

参 考 文 献

1 Universal Serial Bus Specification Revision 2.0
2 Cypress. CY7C68013 Data Manual. 2001
3 Cypress. CY7C68013 Technical Reference Manual. 2001
4 www.cypress.com
5 www.usb.org

选自《单片机与嵌入式系统应用》月刊，2002 年第 10 期

6.18 基于 EZ-USB 的数据采集与控制

武汉空军雷达学院 颜荣江
空军雷达兵第 14 团 阴大兴

一、概 述

在目前 PC 的 I/O 模式中,外围设备通常被映射为 CPU 的 I/O 地址空间,并且被分配一个指定的 IRQ (中断请求),在某些情况下也可以是一个 DMA 通道。这些系统资源被分配给指定的外围设备。这种地址分配的方法已经成为一种标准,软件开发者要根据这对指定的设备进行访问。这给编程者带来了不便,同时外设消耗了 PC 的许多系统资源,使许多系统资源不可使用,并且产生了很多冲突,由此造成了许多问题。

Cypress 推出的带智能 USB 接口的单片机 EZ-USB,极大地降低了 USB 外设的开发难度,为 PC 外设的制造商提供了一个性能优良、价格较低的设计方案。基于 EZ-USB 的强大功能,让我们看到了应用 USB 的美好前景。使用该芯片后,我们在很短的时间内便实现了基于 USB 传输的采集系统。为便于理解,有必要介绍一下相关的 USB 知识。

1. USB 信息包

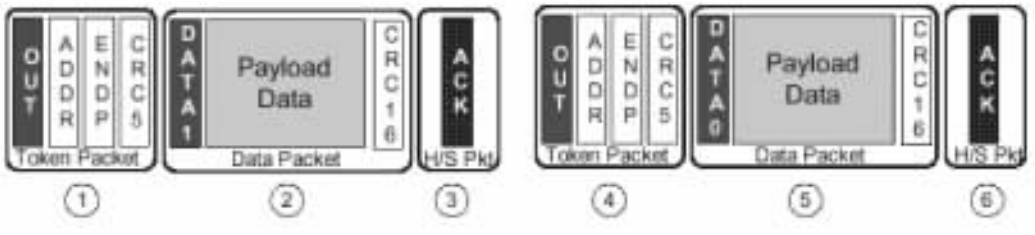
USB 传送的数据包由被称为 Packet IDs (PIDs) 的特定代码所定义,PID 表示了正在被传送的包的类型。USB 的包中一共有四种 PID 类型,如表 6.18-1 所列。

表 6.18-1 USB PID

PID 类型	PID 名称
数据信号	IN、OUT、SOF、SETUP、DATA0、DATA1
握手信号	ACK、NAK、STALL
特殊信号	PRE

图 6.18-1 举例说明了 USB 传输。包①是一个由 OUT 标志表示的输出信号,表示主机发出的数据将要通过总线进行传送。包②由数据组成,它由 DATA1 标志表示。包③是一个由外设发送的握手信息包。外设使用 ACK (确认) 标志向主机报告外设已正确地接收数据。接下来发出的 OUT 信号④,开始进行第二次传输。紧跟其后的是数据包,这一次使用的是 DATA0 标志。最后,设备通过传送握手信息包中的 ACK 标志表示接收成功。

为什么要用到 DATA0 与 DATA1 两种数据标志呢?这是因为 USB 的设计者对待错误的检测非常严格。如前面提到的,ACK 握手信号是一个向主机报告外设无差错接收数据的信号 (包中的 CRC 位用于错误检测)。但是如果在传输过程中,握手信号本身出了错,那又该怎么办呢?为了检测这种错误,在主机和外设双方都保留了一个数据轮换位,它在数据信息包的传送过程中轮流改变其状态。内部轮换位的状态与随数据到达的 PID 相比较,要么是 DATA0,



Token Packet :令牌包或命令包
ENDP :端点
DATA0 :数据 0 标志
CRC16 :16 位 CRC 校验
OUT :输出标志
CRC5. 5 :位 CRC 校验
DATAl :数据 1 标志
H/s Pkt :主/从包
ADDR :地址
ADDR :地址
Data Packet :数据包
Payload Data :有效的数据
ACK :确认位

图 6. 18 - 1 USB 包

要么是 DATA1。当传送数据时,主机或设备发出交替的 DATA0 - DATA1 标志。通过比较数据 PID 和内部轮换位,主机或设备能够检测到一个出错的握手信号包。

SETUP 令牌只针对控制传输。它是数据包中的前 8 个字节,通过这几个字节外设对主机的设备请求进行解码。

SOF 信号每 ms 发生 1 次,代表 1 个 USB 帧的开始。

3 个握手信号 :ACK、NAK 和 STALL。

· ACK 表示"成功",即数据无误接收。

· NAK 表示"忙,重发"。NAK 看上去好像表示"出错",但其实不是。USB 设备不响应,才表示传输有错误。

· STALL 表示有意外事件发生。设备发出 STALL 握手信号,表明它不能接受某个设备请求,或是外部终端出错,或是主机企图访问一个并不存在的资源。该状态有些像"停止",但要好一些,因为 USB 提供了从 STALL 状态中恢复的方法。

PRE (Preamble) PID 的传送优先于一个 USB 低速 (1. 5 Mbps) 传输。EZ-USB 系列仅支持 USB 高速 (12 Mbps) 传输,因此可以忽略 PRE 信息包及接下来的低速传输。

2. USB 主控制器

在 USB 系统中,主机是控制器。USB 设备回应主机的请求,但 USB 设备之间不能互送信息。当主机向 USB 外设发送数据时,主机发出 1 个 OUT 信号。外设如果有时间接收数据,并接收无误,就返回 1 个 ACK 信号给主机 如果忙,就返回 1 个 NAK 信号 如果发现错误,就不返回信号,主机过一段时间会重发。USB 设备从不主动给主机发送数据。然而,在 EZ-USB 芯片中,没有什么可以阻止 8051 把要发送给主机的数据放到端点缓冲器以用于传输,但数据会一直留在缓冲器中,直到主机发出 1 个 IN 信号到那个端点。如果主机不发出 IN 信号,数据就一直留在那里。

3. USB 的方向

如果知道了主机是控制器,就很容易记住 USB 的方向 :OUT 表示从主机到外设,IN 表示从外设到主机。例如,送数据给主机的端点是 IN 端点,即 EZ-USB 送数据到 IN 端点缓冲器。

也就是说,8051 的 OUT 就是对主机的 IN。

4. 帧

USB 主机通过每 1 ms 传送 1 个 SOF (起始帧) 包,向所有 USB 设备提供时间基准。SOF 包包括一个递增的、11 位的帧计数器。8051 可以从两个 EZ-USB 寄存器中读该帧计数器。SOF 时间对同步端点具有意义。EZ-USB 内核为同步端点数据服务提供给 8051 一个中断请求。

5. USB 设备的加载过程

当 USB 设备接入 HUB (集线器) 后,主机控制器和主机软件能自动侦测到设备的接入。然后,主机软件读取一系列的数据用于确认设备特征,如 Vendor ID、Product ID、接口工作方式、功率消耗等参数。之后,主机分配给外设一个单独的地址。地址是动态分配的,各次可能不同。在分配完地址之后,对设备进行初始化,初始化完成以后,就可以对设备进行 I/O 操作了。

二、EZ-USB 的端点和描述符

EZ-USB 芯片具有软配置的特性。通过 USB 接口从主机下载的 8051 代码和数据被存放在内部 RAM。使用 EZ-USB 的外设,可以在无 ROM、EPROM 或 FLASH 存储器的情况下操作。为支持软配置的特性,当 USB 设备没有固件时,EZ-USB 芯片自动进行枚举,所以 USB 接口本身可用来下载 8051 代码和描述符。

一个叫"ReNum"的 EZ-USB 控制位决定是 SIE 内核还是 8051 处理端点 0 上的设备请求。开机时,RENUM 位是 0,表明 EZ-USB 内核自动处理设备请求。一旦 8051 运行,它就置 RENUM = 1,表明使用下载的 8051 代码处理接下来的设备请求。

以下是枚举模式。

当 EZ-USB 芯片脱离复位状态时,EZ-USB 芯片根据 I^2C 总线上的外部 EEPROM 的内容,对如何枚举做出描述。如表 6.18 - 2 所列,其中 VID 表示厂家识别码,PID 表示产品识别码,DID 表示设备识别码。

表 6.18 - 2 EZ-USB 枚举方式

EEPROM 的第一个字节	EZ-USB 内核的处理
既不是 0xB2 也不是 0xB0	由 EZ-USB 内核提供描述符,PID/VID/DID。置 RENUM = 0
0xB0	由 EZ-USB 提供描述符,由 EEPROM 提供 PID/VID/DID。置 RENUM = 0
0xB2	将 EEPROM 的内容写入 EZ-USB 的 RAM,置 RENUM = 1。由 8051 提供描述符,PID/VID/DID

(1) 如果没有 EEPROM 或 EEPROM 的第一字节,既不是 0xB0 也不是 0xB2。EZ-USB 芯片使用其内部存储的描述符数据枚举,它包含 Cypress 半导体的 VID、PID、DID。这些 ID 字节使主机操作系统装载 Cypress 半导体设备驱动程序。

USB 主机在枚举过程中询问设备,读取设备描述符并使用表 2 字节决定下载哪一个固件驱动程序到操作系统。

⑵ 如果串行 EEPROM 与 I^2C 总线相连且第一个字节是 0xB0，EZ-USB 芯片使用内部存储的描述符数据枚举。但是，由外部的 EEPROM 提供 6 字节的 PID/VID/DID 数据，它使主机操作系统装载与之有关的设备驱动程序。

⑶ 如果 EEPROM 的第一字节是 0xB2，EZ-USB 内核将 EEPROM 的内容写到内部 RAM。EZ-USB 内核置 RENUM 位为 1，表明 8051（而不是 EZ-USB 内核）通过控制端点 0 对设备请求作出应答。当然，所有的描述符数据，包括 VID/DID/PID 的值，由 8051 固件提供。

三、EZ-USB 的传输类型

EZ-USB 支持四种传输类型，可满足不同的通信要求。

⑴ 块传输。块传输用于支持突发的大量的数据，以 8、16、32 或 64 字节的信息包传送。

⑵ 中断传输。中断传输用于量少且不经常传送的周期性数据。中断数据的包长限制在 64 字节内。

⑶ 同步传输。同步传输主要用于音频和视频数据流，不具有 USB 定义的格式。

⑷ 控制传输。控制传输应用于控制命令和状态命令的传送。

四、EZ-USB 内核在块（bulk）传输中的作用

在每一个 USB 设备中都有 1 个串行接口引擎（SIE）。SIE 与 USB 数据线 D + 和 D - 相连，传输发向或来自 USB 设备的字节。在块传输中，SIE 对 PID 信息包解码，并通过 CRC 位对数据进行错误检测，然后发送有效数据。如果 SIE 发现 1 个出错的数据，它会自动地不进行响应，而不是提供 1 个 PID 握手信号，并告诉主机延时重发。

块传输是异步传输，采取了使用 ACK 和 NAK 握手 PID 的数据流控制机制。SIE 通过发出 1 个 NAK，向主机报告其“busy”状态。当外设传送数据成功，它就会要求 SIE 发送一个 ACK 信号，表示发送成功。

SIE 接收来自 USB 设备的数据和控制信号，将它们变为 USB 传输格式，并通过 2 根 USB 数据线传送。因为 USB 传输采用的编码方式是 NRZI（反向归零制）编码方式，所以，SIE 还必须在位数据流的适当位置上插入位，以确保数据发送的完整性。这种做法称为“位填充”，它由 SIE 完成。

EZ-USB 芯片采用软配置，即包含内部程序/数据 RAM。EZ-USB 可作为 USB 设备进行连接，将程序下载到内部 RAM，这一切都是由改进后的 SIE 完成。SIE 能够用内部描述符表进行枚举操作。

五、EZ-USB 块传输

为了简化设计，我们采用了块传输方式进行数据的采集。因此，在制定设计方案前，必须对块传输的特点有一个深入的了解。

块传输用于支持突发的大量的数据，对连续性不做要求，如打印数据。它可以利用任何可获得的带宽：

① 以可获得的带宽访问总线；

② 总线错误导致传送失败时，可以重发；

③ 可以保证数据传送，但不能保证传送的带宽和延迟。

仅当有可获得的带宽时,块传输才会发生。当空闲带宽很多时,块传输进行得较快;空闲带宽少时,可能在很长时间内不发生块传输。

1. 块传输的包长限制及使用的端点

USB 规格说明允许块传输数据的最大信息包长度为 8、16、32 或 64 字节。EZ-USB 为 0~7IN和 0~7OUT 16 个端点中的每一个提供了最多能容纳 64 字节的缓冲区。

2. 与块传输相关的控制位及寄存器

块端点中有 6 个端点:2-IN、4-IN、6-IN、2-OUT、4-OUT 和 6-OUT 可与下一个连续编号的端点配对,以提供双缓冲器。这就允许 8051 为一个数据信息包服务,而另一个正通过 USB 进行传输。6 个端点的成对位(USBPAIR 寄存器)控制双缓冲器。

8051 在初始化时,设置了 14 个端点有效位(IN07VAL、OUT07VAL 寄存器),以及 1 个 2 位的控制和状态(CS)寄存器。8051 可读取 CS 寄存器中的位,以判定端点是否忙,并写入另一位强制进入端点 STALL 状态。当设置端点忙位时,8051 绝对不能读写端点缓冲器或字节计数器。

当为 8051 服务的端点变忙时,EZ-USB 内核设 1 个中断请求位。为了自动地把控制权移交到为端点请求服务的 ISR(中断服务程序),EZ-USB 中断向量系统要对端点发出的中断请求继续划分。

3. BULK IN 传输

USB 的 BULK IN 数据从设备流向主机。主机通过向 EZ-USB 内核发出 1 个 IN 标志,请求 1 次 IN 传输,当 EZ-USB 准备好时响应数据。8051 通过装载端点的字节计数器表明准备好。如果 EZ-USB 内核接收到 1 个未准备好的端点的 IN 信号,它将用 NAK 握手信号响应 IN 信号。

在 BULK IN 传输中,8051 在端点缓冲器中已装入了 1 个数据信息包,然后在端点字节计数寄存器中装入信息包的字节数,以协助下一个 IN 传输的进行。这时设置了 BUSY 位。主机发出 1 个 IN 信号,EZ-USB 内核通过传送 IN 端点缓冲器中的数据响应 IN 信号。当 EZ-USB 发出一个表示数据接受正确的 ACK 信号时,EZ-USB 内核清除端点的 BUSY 位,并设置它的中断请求位,告诉主机端点缓冲器为空。如果这是一个多信息包传输,那么主机将发出另一个 IN 信号,以得到下一个信息包。

如果第二个 IN 信号在 8051 有时间装满端点缓冲器之前到达,EZ-USB 将发出 1 个 NAK 握手信号,表示忙。主机继续发出 IN 信号,直到数据准备好。最终,8051 将端点缓冲器装满数据,并将信息包的字节数装入端点的字节计数寄存器(INnBC),为下一次的传输做准备。

4. BULK OUT 传输

USB BULK OUT 数据从主机到达外设。主机通过向 EZ-USB 发出一个紧跟在数据信息包后的 OUT 信号,请求一次 OUT 传输。如果 EZ-USB 内核正确地接收数据,那么 EZ-USB 内核响应 ACK 信号。如果端点没有准备好接收数据,EZ-USB 丢掉主机的 OUT 数据并返回一个 NAK 信号,表明没有准备好。相反地,主机将继续发出 OUT 信号和数据给端点,直到 EZ-USB 内核响应 ACK 信号。

每一个 EZ-USB 的 BULK OUT 端点都有一个字节计数寄存器,它有两个作用。8051 读取字节寄存器,以确定来自主机的最后一次 OUT 传输期间,接收了多少字节。8051 写字节计数器(用任何值),告诉 EZ-USB 内核已经完成了从缓冲器中读字节的动作,使缓冲器能够接收下一个 OUT 传输。

5. 错误恢复

EZ-USB 内核自动检查错误,如果它用建立在 USB 上的错误检测器件 (CRC 检测和数据替换) 时,发现错误,就请求主机再次传输数据。

六、程序设计

我们采用 EZ-USB 系列单片机的 AN2131QC 芯片实现块传输的应用。AN2131QC 芯片采用 80 引脚的封装,除了具有 24 个 I/O 引脚外,还包含 1 个 16 位地址总线和 1 个 8 位数据总线,以用作外部存储器的扩展。在该芯片内部有 1 个 8 KB 的 RAM,使得固件可以从主机下载。

BULK 传输源程序清单:

```
;-------------------------------------------------------
$ NOMOD51                                    ;取消默认的 8051 寄存器定义
$ nolist
$ include (hezusb\htarget\hinc\hezregs.inc)  ;EZ-USB 寄存器申明
$ list
NAME    ezbulk
    ISEG AT 60H                              ;堆栈段
STACK:  DS      20                           ;申明使用 20 个堆栈单元
        CSEG AT 00h
        LJMP    START
        ORG     200h
START:  MOV     SP,#STACK - 1                ;设置堆栈指针
LOOP:   MOV     DPTR,#OUT2CS                 ;EP2IN 控制和状态寄存器
        MOVX    A,@DPTR
        JB      ACC.1,LOOP                   ;检查"忙"位,为低时允许进行服务处理
        MOV     DPTR,#IN2CS                  ;EP2OUT 控制和状态寄存器
        MOVX    A,@DPTR
        JB      ACC.1,LOOP                   ;检查是否处于"忙"状态,为"1"时表示忙
service_IN2:
        MOV     DPTR,#OUT2BC
        MOVX    A,@DPTR
        MOV     R2,A
        MOV     DPTR,#OUT2BUF
        MOV     R3,DPL
        MOV     R4,DPH
        MOVX    A,@DPTR
        MOV     DPTR, #IN2BUF
        MOV     R5,DPL
        MOV     R6,DPH
        MOVX    @DPTR,A
        DEC     R2
```

```
LOOP1：  MOV     DPL,R3
         MOV     DPH,R4
         INC     DPTR
         MOVX    A,@DPTR
         MOV     R3,DPL
         MOV     R4,DPH
         MOV     DPL,R5
         MOV     DPH,R6
         INC     DPTR
         MOVX    @DPTR,A
         MOV     R5,DPL
         MOV     R6,DPH
         DJNZ    R2,LOOP1
         MOV     DPTR,#OUT2BC      ;装载任意值到字节计数器,以清除输入缓冲区
         MOVX    A,@DPTR
         MOVX    @DPTR,A
         MOV     DPTR,#IN2BC       ;将收到的数据通过 EP2IN 发送出去
         MOVX    @DPTR,A           ;写发送字节计数器
         SJMP    LOOP
         END
```

选自《单片机与嵌入式系统应用》月刊,2002 年第 3 期

6.19 基于 USB 接口的 IC 卡读写器的设计

岳阳师范学院物理系(414006) 张国云 彭仕玉
湖南大学电气与信息工程学院(410082) 禹柳飞 龚辂洲

过去,人们在研制计算机外设时,均采用扩展卡、行打印机接口或串行 RS232 接口完成外设与计算机之间的物理连接,市场中现该类产品仍占有很大的份额。自从 Intel、Microsoft、NEC、IBM 等七家公司共同推出 USB 通用串行接口标准以来,带 USB 接口的产品陆续出现,考虑到 USB 接口具有速度快、易扩展、支持热插拔和即插即用,并可提供总线供电,还可通过 USB 集线器以菊花链的形式连接多达 127 个 USB 外设等优点,而且目前计算机主板都带有 USB 接口,Windows98 也全面支持 USB 标准。因此,我们在设计 IC 卡收费管理系统中发行 IC 卡和充卡处的 IC 卡读写器时,决定采用 USB 接口,克服了以往的 IC 卡读写器不支持热插拔和不能灵活扩展外设等缺点。

一、读写器硬件设计

本控制器针对 Siemens 公司加密型 IC 卡 SLE4442 而设计。其中 USB 接口控制器采用 USBN9603,MCU 采用 Atmel 公司的 AT89C52,有关 AT89C52 与 SLE4442 加密型 IC 卡的硬件连接原理及软件设计方法参阅文献[1],带 USB 接口的 IC 卡读写器硬件原理框图如图 6.19-1 所示。

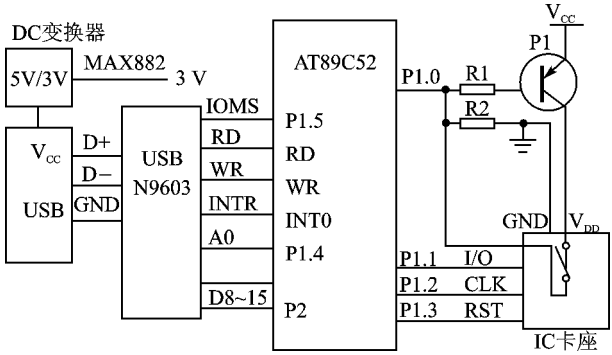


图 6.19-1 读写器电路原理框图

1. USB 接口控制器 USBN9603 简介

USBN9603 是由 National Semiconductor 生产的一种专用 USB 接口控制器。它集成了一个 USB 收发器,来满足传输时的电气性能要求;另外还集成了 SIE (Srial Interface Engine),主要负责时钟恢复,数据包结束 EOP (End of Packet) 检测,位填充,位解填充,CRC 编解码、组帧、拆帧、包类型识别以及结点状态识别等。它通过四根线 VBUS、GND、D+、D- 与主机或 USB HUB 实现物理连接。其中 VBUS 为总线电源,可对 USB 外设提供 +5 V 电源;GND 为地线;D

+和 D- 为数据线。USB 利用 D+和 D- 两线,采用差分信号的传输方式传输串行数据,支持高速或者低速传输模式。如果在 D+接一上拉电阻,则工作在高速传输模式;反之,如果在 D-接一上拉电阻,则工作在低速传输模式。另外,它还具有一个双向的控制端点 Endpoint0 (8 byte FIFO),三个发送端点,三个接收端点 (各有 64 byte FIFO),并且具有两种模式的 8 bit 并行接口,与 Intel MCU 兼容等特点。

2. 读写工作原理

在设计本系统时,按系统功能要求,IC 卡读写器受控于主机,具体工作过程是当用户通过主机端的应用软件向 IC 卡读写器发读写操作命令,AT89C52 通过 USB 接口接收到命令后,首先检查是否有 IC 卡插入,若有 IC 卡插入,则检查 IC 卡的有效性,否则均反馈给主机相应的信息;若是有效卡,则根据主机发来的读写命令完成相应的操作,读写操作可以是读写 IC 卡内的数据信息,也可以是 3 字节的密码。

AT89C52 对 USBN9603 操作时,由 P1.5 控制 IOMS 设为 I/O 端口访问方式,并根据 RD、WR 信号配合 A0 电平状态通过 D8~15 完成端口数据的读写。端口访问方式采用中断方式,即由 USBN9603 向 AT89C52 的 INT0 产生中断请求信号,AT89C52 通过读取 USBN9603 相关寄存器判断产生中断的原因,从而执行相应的中断处理程序。

二、IC 卡读写器软件设计

要实现主机与 USB 设备进行通信,系统软件包括在主机方面为 USB 设备驱动程序及用户层操作软件,在设备方面为固件程序。

由于 Windows 98 提供了 USBD (USB 系统驱动程序),且 USBD 又给用户提供了直接支持的 USBDI,用户在编写 USB 驱动程序时,只需配置满足 USB 要求的 URB (USB 请求块),并通过 USBDI 发送下去即可实现对 USB 设备的控制。在 USB 支持的四种传输模式中,我们选择中断传输模式,这根据 IC 卡读写时钟速度选择的结果。所以,在编写 USB 驱动程序时,首先读出 IC 卡读写器的描述符,根据描述符,选择配置和接口,再利用 USBDI 传输设备操作命令。

下面给出读取 IC 卡读写器描述符函数 UsbGetDevice Descriptor () 及选择配置和接口函数 UsbSelectConfigurationl ()。

```
NTSTATUS UsbGetDeviceDescriptor (OUT PUSB_DEVICE
DESCRIPTOR & DeviceDescriptor)
{
USHORT UrbSize = sizeof (struct URBCONTROL_DESCRIPTOR_REQUEST);
PURB urb = (PURB) ExAllocatePool (NonPagePool, Urb-Size);
Return STATUS_INSUFFICINT_RESOURCE;
IF (urb = = NULL)
    Return STATUS_INSUFFICIENT_RESOURCE;
ULONG SizeDescriptor = sizeof (USB_DEVICE_DESCRPTOR);
DeviceDescriptor = (PUSB_DEVICE_DESCRIPTOR) ExAllocatePool (NonPagedPool, SizeDescriptor);
IF (DeviceDescriptor = = NULL)
    Return STATUS_INSUFFICIENT_RESOURCE;
UsbBuildGetDescriptorRequest (urb, UsbSize, USB_DEVICE_DESCRIPTOR_TYPE, 0, 0, DeviceDescriptor,
NULL, SizeDescriptor, UNLL);
```

```
ExFreepool (urb);
Return status;
}
NTSTATUS UsbSelectConfiguration ()
{
PUB_CONFIGURATION_DESCRIPTOR  Descriptor = NULL;
NTSTATUS status = UsbGetConfigurationDescriptor (Descriptors);
PUSB_INTERFACE_DESCRIPTOR
id = USBD_ParseConfigurationDescriptorEx (Descripor, Descriptor);
USBD_INTERFACE_LIST_ENTRY ilist [2];
ilist [0]. interfaceDescriptor = id;
ilist [0]. interface = NULL;
ilist [0]. interfaceDescriptor = NULL;
PURB urb = USBD_CreateConfigurationRequesEx (Descriptor, ilist);
Status = Callusbdi (urb);
ExFreePool (urb);
ExFreePool (Descriptor);
Return status;
}
```

参 考 文 献

1 张国云. 加密型 IC 卡与 AT89C51 单片机接口程序设计 [J]. 电子与自动化,2000 (5)

2 Chris Cant 著. Windows WDM 设备驱动程序开发指南 [M]. 孙义等译. 北京 机械工业出版社,2000

选自《电子技术》月刊,2002 年第 1 期

6.20 IEEE 1394 总线技术与应用

西安第二炮兵工程学院 (710025) 郑青阳 许化龙

一、发展背景

IEEE 1394 又称火线 (firewire)、i. link 和 Lynx,是一种高速串行总线标准。早在 1985 年,苹果公司 (Apple) 开发了名为火线的串行总线,开始在其 Mac 机上应用。德州仪器 (TI) 与索尼公司对该技术给予了支持,为其研制了通信芯片,用于数码相机等影像设备的数字接口。1995 年,电气与电子工程师协会 (IEEE) 以火线为蓝本正式制定了 IEEE 1394 标准。然而,由于当时对数据传输速度的要求不高,加之该技术由苹果公司主导,因而在以"标准为王"的 IT 业竞争中受到了 Intel 等主流 PC 厂商的冷落,长期得不到重视和普及。随着计算机高速外围设备和信息家电 (IA) 的迅速发展,人们迫切需要一种在高速数字设备之间实现灵活通信的串行总线。Intel 的 USB 1.0 总线的最大传输速率仅有 12 Mb/s,远不能满足此类需求,而 USB 2.0 和串行 ATA1.0 等新规范又缺乏软硬件支持,此时,性能出色、传输速度高达 400 Mb/s 且技术成熟的 1394 总线开始显露锋芒,成为应用热点。进入 21 世纪,IT 业的标准之争更加激烈。出于与 Intel 竞争的需要,台湾的威盛公司 (VIA) 全力倡导 1394 架构,并为此开发了一体化的新型接口芯片,而 Intel 也改变了从前的冷淡态度,参与了 1394 贸易协会的工作并力求成为新标准的主导者。在这种背景下,1394 总线在最近的 1、2 年中得到了迅速发展和广泛应用,极有可能成为 IT 和 IA 领域的基本总线标准。

二、主要技术规范

IEEE 1394 定义了数据的传输协定及连接系统,其初始设计目标是在较低成本下实现 PC 机与外设和消费性电子产品 (CE) 的高速数据传输,主要有以下规范。

(1) IEEE 1394—1995 是 1394 体系的原始规范,定义了 1394 的总线结构、数据传输协议和传输媒介,是目前所有 1394 设备遵循的标准。

(2) IEEE P1394a 又称 IEEE 1394—2000,是 IEEE 1394—1995 的附加规范,对原规范中含糊的地方做出了详尽的解释,增强了产品的兼容性,同时规定了增强性能的措施,并对电源管理特性做了较大的改进,是目前业界最欢迎的标准。

(3) IEEE P1394b 是正在由 Intel 主持制定的最新附加规范,在保持向下兼容的同时极大地提高了传输速率,增加了传输距离,并支持包括光纤在内的多种传输媒介。该规范很可能成为未来的 1394 基本标准。

(4) 开放式主机控制器接口 (OHCI) 规范 是基于 1394 基础规范之上制定的,定义了 1394 总线接入主机的方式,是向所有支持 1394 厂商提供的开放式标准。该规范定义 1394 接口由物理层、链路层、交易层和总线管理层 4 部分构成,其结构参见图 6.20-1。物理层提供设备和线缆之间的物理连接,直接控制数据的收发 链路层提供同步和异步模式下的数据包接

收确认、定址、数据校验与分帧,物理层与链路层由硬件实现。交易层只处理异步数据包,而总线管理层提供全部总线的控制功能,包括电气供应、优化定时机制、分配同步通道 ID 和处理出错信息,这两层由软件实现。

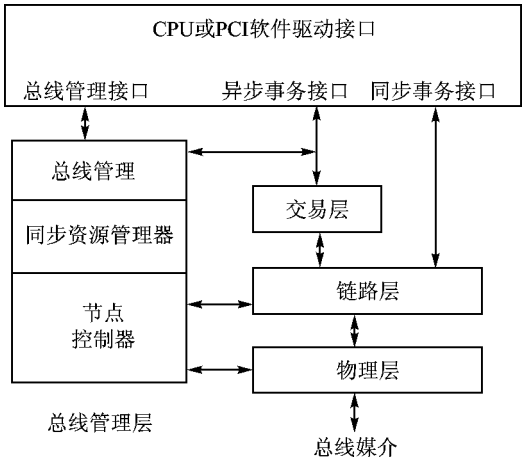


图 6.20－1 1394 协议结构

此外还有一些应用中的附加规范,如 IEC 61883 定义了控制 1394 上特定的音视频设备的详细情况,串口协议 2 (SBP—2) 定义了在 1394 上压缩设备命令的标准方式等。这些规范构成了完整的 1394 系统体系,也是应用 1394 技术的基本依据。

三、网络连接与工作机理

1394 支持底板环境、线缆环境与无线环境。底板环境支持与并行总线同时建立冗余串行通信信道,线缆环境允许外部设备的远程连接,无线环境包括采用红外或无线电方式的连接,目前还处在试验阶段。本文将主要介绍线缆环境。

1. 线缆连接设备

线缆连接设备包括电缆和连接器,是总线外部特征的重要体现,分为 6 芯和 4 芯两种类型。图 6.20－2 是 6 芯电缆剖面图和两端插头的连接图。

1394 采用两对双绞线传输数据,一对用来发送,另一对用来接收,因此实现了真正的全双工通信。插头 1、2 点分别是电源和地,3、4 点为数据发送端,5、6 点为接收端,两对数据线在端头之间是交叉的。线缆两端的连接器是相同的,因此网络可能被误接成环型拓扑从而造成配置失败,此时节点可检测到这各情况并向管理层报告。两根电源线提供直流 8～40 V,最大 1.5 A的电流,故功率不超过 60 W,可驱动小型设备,还可在本地供电设备掉电后维持其 1394 接口物理层正常工作,以保证由其下延的网络拓扑不被分割,从而提高了连接的可靠性。6 芯插槽的电源针比信号针长,以保证电源先于信号线接通并后于信号线断开。4 芯线缆是索尼公司的 i. link 版本,去掉了电源线并对连接器形状作了改动,其他特性与 6 芯相同。4 芯头可与 6 芯头互连。

2. 拓扑结构与工作过程

1394 网络由网段和节点构成,每个网段 (即一条总线端口) 可包含 64 个节点 (0～63),其中节点 63 被用作公共广播解址节点,因此一条总线上可连接 63 台设备。1394 网络允许最多

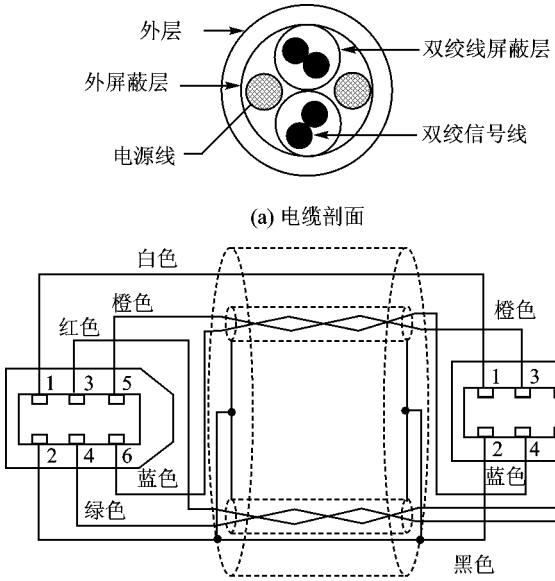


图 6.20－2 1394 线缆连接设备

1024 个网段，网段间可用网桥互连，所以共可连接 64512（1024 × 63）台设备，拓扑结构为树型或菊花链型。图 6.20－3 是 1394 网络的拓拟示意图，可以看出 1394 的网络连接形式非常灵活。图中的每个节点都可作为根节点（root）向下延展，当增添和移除设备时网络会自动重组。网桥将整个网络分为左右 2 个网段，每一时刻 1 个网段只可被 1 台设备占用，而网桥则可对网段间的数据传输进行智能控制，既能保证网段间的相互访问也可进行有效隔离，保证各网段的独立。中继器可将两节点的间距延长 1 倍，分离器可将慢速设备（打印机）隔离于高速设备连接之外，防止造成传输瓶颈。

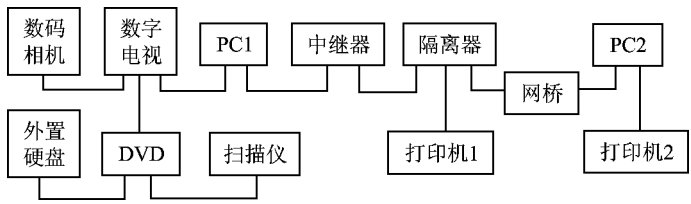


图 6.20－3 1394 网络拓扑示意图

各节点在数据传输之前需进行握手通信，建立连接后即可开始传输。具体过程如下：设备首先要求控制物理层。若进行异步传输，则数据发送方和接收方交换地址，双方确认后转入交易层进行数据传输。接收方收到数据包后会进行校验并向发送方传回确认信息，若出错则启动错误修复机制进行重发。如果进行同步传输，发送方首先要求获得一个特定带宽的数据通道，然后将通道 ID 号附加在数据中发送，接收方对数据流进行检测，只接受具有特定 ID 号的数据。

四、技术特点分析

1. 优点分析

(1) 高带宽 目前的 IEEE 1394—1995 与 P1394a 可以达到 100、200 和 400 Mb/s 的速率，而 P1394b 更可达到惊人的 800 Mb/s、1.6 Gb/s、3.2 Gb/s 和 4 Gb/s，不但高于其他任何一种串行总线，也不逊色于主流的并行总线。如此高的传输速率得益于先进的内存映射架构。在这种架构下，所有 1394 总线上的资源都可映射到相应设备的某段内存地址，设备资源被看作寄存器和内存单元，设备通过对该段内存地址的访问来完成对数据的存取。正是基于这种特殊的数据地址识别和传输方式，使 1394 具有了高速传输速率。

1394 网段上的传输速率是可变的，这增加了连接的灵活性，同时也应注意在两台高速设备之间不可有低速设备，否则将产生传输瓶颈。此种情况下可用分离器对低速设备进行隔离。

(2) 支持同步和异步混合传输 同步与异步传输的过程已在前文介绍，异步传输的意义在于随机数据的可靠传输。异步事务要经过复杂的握手方式建立传输关系，传输过程中要对数据进行 CRC 校验和回传，如出错还需重发。而同步方式则适用于要求传输速率恒定但对错误不敏感的场合，如在实时的音频播放传输中，数据的生命期很短，数据流的连续性远比准确性重要，因此同步事务强调恒定带宽而不要求传输确认，因此较为简单。图 6.20－4 示意了两种传输的带宽分配，1394 总线以 125 μs 的周期传送数据，同步事务一旦完成申请，即在每个周期中占据相同的带宽直至结束，每个周期中所有同步事务占据的带宽不超过 80%，其余不小于 20% 由异步事务分享。在每个通道中数据包的大小是有限制的，在 400 Mb/s 的传输速率下，每个同步数据包不得超过 4 KB，异步数据包不得超过 2 KB。这种混合传输方式有利于音视频流等实时数据的传输，也能有效利用总线带宽资源。

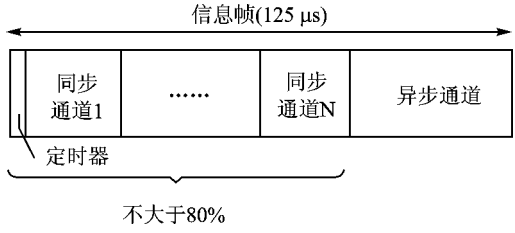


图 6.20－4 1394 传输带宽的分配

(3) 灵活的点对点式通信架构 (peer to peer) 1394 网络是一个全分布控制的对等网，任何两个设备都可直接连接而无需设置服务器。事实上，1394 体系中的 PC 仅仅作为一个普通节点而不是网络的控制中心，这是与 USB 的重大差别，也反映出双方对 PC 在未来信息网络中地位的不同理解。这种点对点的方式增强了网络的灵活性与可靠性，例如在进行热插拔时不必开启计算机即可自动完成配置，而 PC 的故障也不会导致网络的瘫痪。

2. 局限性

(1) 连接地域有限 目前 IEEE 1394—1995 标准的相邻节点最大间距只有 4.5 m，P1394a 规定为 10 m，P1394b 达到 100 m。用中继器可将距离加倍，但中继器占用资源，且 1394 最多支持 16 层树形网段，故目前两节点最大间距为 72 m (4.5×16)，因而 1394 应用的领域是延伸的桌面系统和局域信息网络，而不适合广域网。

⑵ 首字节延迟大 由于 1394 采用同步与异步混合仲裁,使得在传输循环包开始之前出现相当大的时滞,即为开始一次传输所需要的系统消耗较多。公布的测算数据是首字节延迟时间为 50 ~ 200 μs,不利于其在工业测控中的应用。

⑶ 由于 1394 采用点对点式的对等网结构,不需要服务器的控制,访问与延时的随机性大,因此不太适合集中管理的场合。

五、应用分析

1. 信息家电

1394 首先在数码摄像机等需要高带宽的数字影像领域得到了广泛的应用,并开始进入数字电视、机顶盒、DVD 和游戏机等领域,是目前数码影像设备接口的事实标准。凭借其巨大的数据吞吐量和灵活的点对点架构,1394 可将各种家庭数字电器和 PC 组成网络,实现家庭信息共享,并可接入宽带互联网,实现高质量视频点播、可视化通信等现代家庭信息服务。从发展趋势看,1394 很可能成为未来信息家电数字接口标准,Intel 也希望将其应用定位于数码消费性电子产品领域。

2. PC 体系的连接

1394 接口早已是苹果 Mac 电脑的标准配置,目前正被高端 PC 广泛采用,以连接外置硬盘、光驱和扫描仪等高速外设。尽管 Intel 提出了 USB 2.0 标准,但并未动摇 1394 在该领域的地位。最新的 Windows XP 操作系统便增强了对 1394 的支持而并未加入对 USB 2.0 的标准驱动环境。事实上,随着 IT 与 IA 技术的融合,上述 PC 外围设备正在逐步融入数字家电,而这正是 1394 的传统领地。在 PC 内部,由于并行 ATA 总线的体积大且控制复杂,未来必将被串行总线取代。Intel 为此提出了串行 ATA1.0 标准,而新一代的 1394 标准也完全可以满足此项需求,其底板环境甚至有能力取代 PCI 总线。所以 1394 未来能否在 PC 体系中获得广泛应用取决于和 Intel 标准竞争的结果。

3. 工业测控领域

由于控制与测量系统中的数字传输量越来越大,迫切需要一种轻便灵活、高带宽、高可靠的传输总线。1394 的优异性能引起了工业界的重视,美国军方就曾对其进行过深入研究,希望将其应用于未来航空电子设备的连接系统。从目前的实际情况来看,1394 比较适合连接高端测试系统,实时传输来自示波器、逻辑分析仪、图像采集装置和外围高速 AD 的海量数据,可以充分满足高端用户的需求。另外,1394 也可用于外挂式虚拟测量仪器,用来将外挂仪器功能卡与 PC 相连,其速度性能远高于目前应用的 IEEE 1284 并口总线,可与内置的 PC 总线虚拟仪器持平,能满足各种虚拟仪器的速度要求,其热插拔特性还使得系统的扩展和重组更加方便。

六、结 语

IEEE 1394 总线从苹果公司的火线算起已有了十几年的历史,但今天仍是应用与研究的热点,这在以摩尔定律发展的 IT 业堪称奇迹。从 1999 年起,1394 开始高速发展,据统计,现在每季度投放市场的带有 1394 接口的数字摄像机不低于 100 万台,到 2000 年,各种应用 1394 的设备超过了 3 000 万台,预计在 2004 年将超过 2 亿台。1394 在工业领域的应用研究也正在开展,相信不久便有产品问世。随着 1394 产品生产与开发力度的加大,其价格也不断下降,将

逐渐从高端走向普及,市场前景非常光明。未来的 1394 总线将向光纤与无线传输方向发展,传输速度更大,连接也会更加灵活。有理由相信,IEEE 1394 凭借其优异的性能将成为名副其实的未来总线标准。

参 考 文 献

1 刘卫明,喻金平. 计算机之间连接分析. 西安:现代电子技术,2001 (10):5~9

2 [美] Don Anderson 著. FireWire 系统体系. 姜汉龙译. 北京:中国电力出版社,2001

3 Bill Pearson. Protocol Enhancements In P1394a And Why They Are Important For 1394 Devices In A PC Environment (White Paper). Intel Corporation, 1998:3~9

4 黄迅. 火线总线及其在测试系统互联中的应用. 北京:电子产品世界,2001 (8):68~69

5 Sid Jones. Next Generation Instrumentation Bus Team (Report). Naval Air Warfare Center, Aircraft Division, 1998:1~15

6 王省书等. 通用串行总线及其应用. 成都:计算机应用研究,2000 (1):5~8

选自《现代电子技术》月刊,2002 年第 3 期

第七章

可靠性及安全性技术

7.1 单片机复位电路的可靠性分析

上海理工大学光电子学院(200093) 袁旭军 庄松林

单片机目前已被广泛地应用于家电、医疗、仪器仪表、工业自动化、航空航天等领域。市场上比较流行的单片机种类主要有 Intel 公司、Atmel 公司和 Philip 公司的 8051 系列单片机,Motorola 公司的 M6800 系列单片机,Intel 公司的 MCS96 系列单片机以及 Microchip 公司的 PIC 系列单片机。无论用户使用哪种类型的单片机,总要涉及到单片机复位电路的设计。而单片机复位电路设计的好坏,直接影响到整个系统工作的可靠性。许多用户在设计完单片机系统,并在实验室调试成功后,在现场却出现了"死机"、"程序走飞"等现象,这主要是单片机的复位电路设计不可靠引起的。图 7.1-1 是一个单片机与大功率 LED 八段显示器共享一个电源,并采用微分复位电路的实例。在这种情况下,系统有时会出现一些不可预料的现象,如无规律可循的"死机"、"程序走飞"等。而用仿真器调试时却无此现象发生或极少发生此现象。又如图 7.1-2 所示,在此图中单片机复位采用另外一种复位电路。在此电路的应用中,用户有时会发现在关闭电源后的短时间内再次开启电源,单片机可能会工作不正常。这些现象,都可认为是由于单片机复位电路的设计不当引起的。

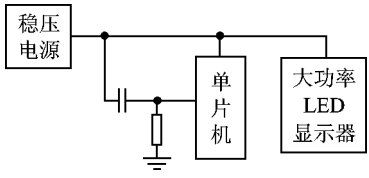


图 7.1-1 单片机系统与大功率 LED 显示数组共享电源原理示意图

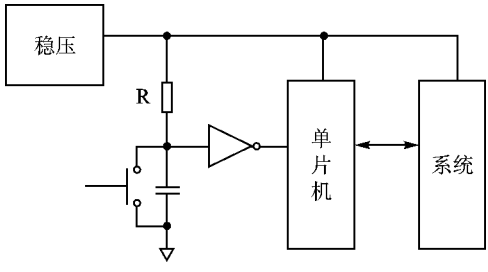


图 7.1-2 单片机积分型复位电路原理示意图

目前为止,单片机复位电路主要有四种类型:(1) 微分型复位电路;(2) 积分型复位电路;(3) 比较器型复位电路;(4) 看门狗型复位电路。另外,Maxim 等公司也推出了专用于复位的专用芯片[1]。

一、复位电路的数学模型及可靠性分析

1. 微分型复位电路

微分型复位电路的等效电路如图 7.1－3 所示。以高电平复位为例,建立如下方程:

$$\begin{cases} U_{RST}(0) = U_0 \\ U_{RST} = U_S \cdot \dfrac{R}{R + \dfrac{1}{SC}} = \dfrac{RCS}{RCS + 1} \cdot U_S \end{cases}$$

电源上电时,可以认为 U_S 为阶跃信号,即 $U_{RST}(t) = U_0 + U_S \cdot e^{-\frac{t}{\tau}}$。其中 U_0 是由于下拉电阻 R 在 CPU 复位端引起的电压值,一般为 0.3 V 以下。但在实际应用中,U_S 不可能为理想的阶跃信号。其主要原因有两点:(1) 稳压电源的输出开关特性;(2) 设计人员在设计电路时,为保证电源电压的稳定性,往往在电源的输入端并联一个大电容,从而导致了 U_S 不可能为阶跃信号特征。由于第一种情况与第二种情况在本质上是一样的,即对 U_S 的上升斜率产生影响,从而影响了 U_{RST}的复位特性。为此假设 U_S 的上升斜率为 k,从 0 V ~ U_S 需要 T 时间,即:

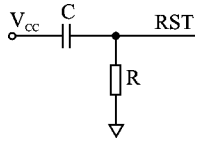


图 7.1－3 微分型复位电路原理示意图

$$U_S = \begin{cases} kt & (0 \leqslant t \leqslant T) \\ kT & (t > T) \end{cases}$$

因为

$$U_S(S) = \frac{k}{S^2}(1 - e^{-ST})$$

所以

$$U_{RST}(S) = \frac{k}{S\left(S + \dfrac{1}{\tau}\right)}(1 - e^{-ST})$$

所以

$$U_{RST}(t) = k\tau e^{-t/\tau}(e^{T/\tau} - 1)$$

当 $T \ll \tau$ 时,U_S 上电时可等效为阶跃信号。与前相同,当 $T \gg \tau$ 时,令 $A = T/\tau$,则:

$$U_{RST}(t) = \frac{k\tau}{T} \cdot Te^{-t/\tau}(e^{T/\tau} - 1),\text{即 } U_{RST}(t) = \frac{U_S}{A} \cdot e^{-t/\tau}(e^{T/\tau} - 1)$$

因为 $A \geqslant 1$,所以 $U_{RST}(t) \geqslant U_S \cdot e^{-t/\tau}$,即此时的复位可靠性较前面的好。

另一种情况就是设计人员将一些开关性质的功率器件,如大功率 LED 发光管与单片机系统共享一个稳压电源,而单片机系统的复位端采用微分复位电路,由此也将造成复位的不正常现象。具体分析如图 7.1－4 所示。

将器件等效为电阻 R_L,其为开关特性即 R_L 很小或 R_L 很大两种工作状态。而稳压电源的基本工作原理是:$\Delta R_L \rightarrow \Delta I \rightarrow \Delta U \rightarrow -\Delta I \rightarrow -\Delta U$。从中可以看出,负载的变化必然引起电流的变化。为了分析简单,假设 $R > R_L$,并且 $R >> R_0$。这样,可以近似地将以上电路网络看作两个网络的组合,并且网络之间的负载效应可以忽略不计。

第一个电路网络等效为一个分压电路。当 R_L 从 $R_{Lmin} \rightarrow R_{Lmax}$ 时,使其变化为阶跃性质,则 U_A 为一个典型的阶跃信号。

$$U_A(t) = \frac{R_{Lmax}}{R_{Lmax} + R_0} - U \qquad t \geqslant 0$$

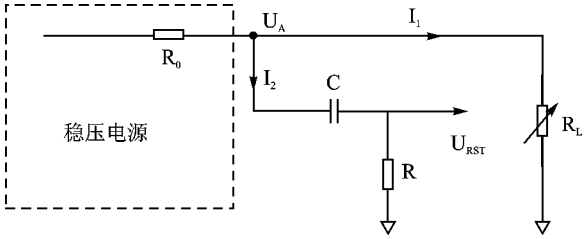


图 7.1－4 微分电路与稳压电源、功率器件一起应用的场合

$$U_A(t) = \frac{R_{Lmin}}{R_{Lmin} + R_0}U \qquad t < 0$$

用此阶跃信号作为第二个电路网络，一阶微分电路的输入，则可得下式：

$$\frac{d}{dt}U_A(t) = \frac{1}{RC}U_{RST}(t) + \frac{d}{dt}U_{RST}(t)$$

$$U_{RST}(0) = 0$$

解之得：

$$U_{RST}(t) = \left(\frac{R_{Lmax}}{R_{Lmax} + R_0} - \frac{R_{Lmin}}{R_{Lmin} + R_0}\right) \times e^{\frac{t}{RC}}$$

从上式可以看出，由于负载的突变和稳压电源的稳压作用，将在复位端引入一个尖脉冲，从而导致 CPU 工作不正常。

2. 积分型复位电路

此电路的等效电路如图 7.1－5 所示。仍以高电平复位为例，同样可以建立如下方程：

$$\begin{cases} C \cdot \dfrac{dU_C}{dt} = (U_S(t) - U_C(t))/R \\ U_C(t) = U_0 \end{cases}$$

当系统上电时，假设 $U_S(t) = AU(t)$ 为阶跃函数，$U_0 = 0$，则：

$$U_C(t) = A(1 - e^{-\frac{t}{RC}})$$

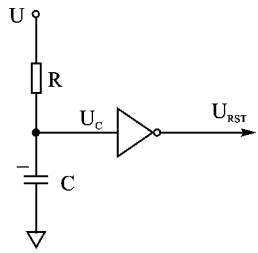


图 7.1－5 基本积分型复位电路

当反相器正常工作后，U_C 若仍能保持在 V_{IL} 以下，则其输出就可以为高电平；而且如果从反相器正常工作后开始，经过不小于复位脉冲宽度的时间 T_R 后，U_C 才能达到 V_{IL} 以上，那么上电复位就能保证可靠。所以在实际应用中，设计人员常常将 R、C 的值增大以提高时间常数，并且应用具有斯密特输入的 CMOS 反相器以提高抗干扰性。然而此复位电路常常在二次电源开关相对较短的时间间隔情况下出现异常。这主要是由于放电回路与充电回路相同，导致放电时间常数较大，从而导致 U_C 电压下降过度。为此有文献 [2] 介绍如图 7.1－6 所示的改进电路。

从图 7.1－6 可以看出，放电回路的时间常数一般远远小于充电时间常数。这时，上面所提到的重复开关电源而造成上电复位不可靠的现象就可以得到控制。然而，由于放电时间常

数过短，降低了此复位电路在工作中对电源电压波动的不敏感性。例如，当电源电压有波动时，此时由于放电过快，从而有可能造成 U_C 低于反相器的 V_{IL} 电压值，带来不必要的复位脉冲。此现象在单片机工作于 Sleep 方式与 Active 方式切换，而电源输出功率又相对较弱时可能出现。为此提出针对以上现象的改进积分型复位电路，如图 7.1－7 所示。图中，$R_1 \ll R_2$，适当调整 R_1 值的大小就可避免以上情况发生。

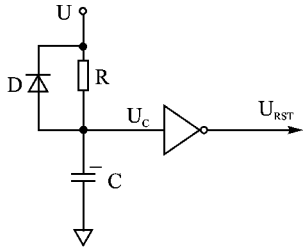


图 7.1－6　改进型 I 积分复位电路

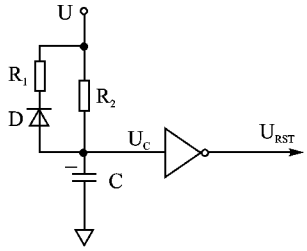


图 7.1－7　改进型 II 积分复位

3. 比较器型复位电路

比较器型复位电路的基本原理如图 7.1－8 所示。上电复位时，由于组成了一个 RC 低通

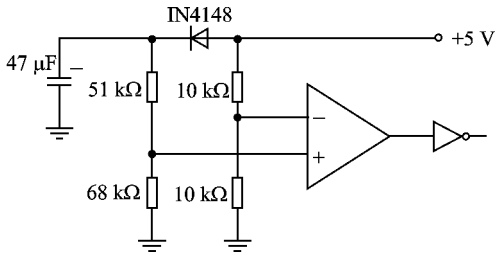


图 7.1－8　基本比较器型复位电路

网络，所以比较器的正相输入端的电压比负相端输入电压延迟一定时间。而比较器的负相端网络的时间常数远远小于正相端 RC 网络的时间常数，因此在正端电压还没有超过负端电压时，比较器输出低电平，经反相器后产生高电平。复位脉冲的宽度主要取决于正端电压上升的速度。由于负端电压放电回路时间常数较大，因此对电源电压的波动不敏感。但是容易产生以下二种不利现象：(1) 电源二次开关间隔太短时，复位不可靠；(2) 当电源电压中有浪涌现象时，可能在浪涌消失后不能产生复位脉冲。为此，将改进比较器重定电路，如图 7.1－9 所示。这个改进电路可以消除第一种现象，并减少第二种现象的产生。为了彻底消除这二种现象，可以利用数字逻辑的方法与比较器配合，设计如图 7.1－10 所示的比较器重定电路。此电路稍加改进即可作为上电复位与看门狗复位电路共同复位的电路，大大提高了复位的可靠性。

4. 看门狗型复位电路

看门狗型复位电路主要利用 CPU 正常工作时，定时复位计数器，使得计数器的值不超过某一值；当 CPU 不能正常工作时，由于计数器不能被复位，因此其计数会超过某一值，从而产生复位脉冲，使得 CPU 恢复正常工作状态。典型应用的 Watchdog 复位电路如图 7.1－11 所示。此复位电路的可靠性主要取决于软件设计，即将定时向复位电路发出脉冲的程序放在何处。一般设计，将此段程序放在定时器中断服务子程序中。然而，有时这种设计仍然会引起程

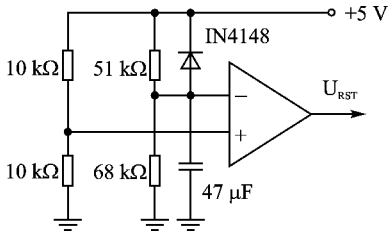


图 7.1－9 改进型比较器复位电路

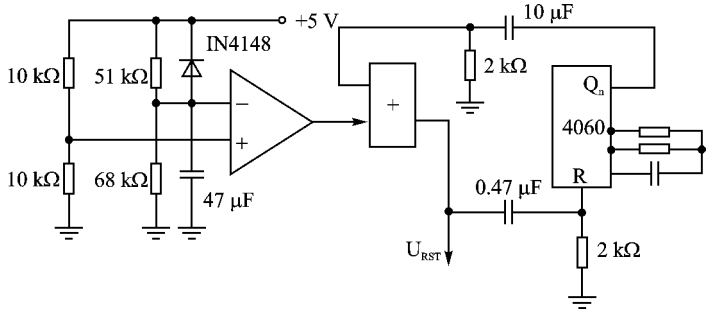


图 7.1－10 进一步改进的比较器复位电路

序走飞或工作不正常[3]。原因主要是:当程序“走飞”发生在定时器初始化以及开中断之后的话,这种“走飞”情况就有可能不能由 Watchdog 复位电路校正回来。因为定时器中断一直在产生,即使程序不正常。Watchdog 也能被正常复位。为此提出定时器加预设的设计方法。即在初始化时压入堆栈一个地址,在此地址内执行的是一条关中断和一条死循环语句。在所有不被程序代码占用的地方尽可能地用子程序返回指令 RET 代替。这样,当程序走飞后,其进入陷阱的可能性将大大增加。而一旦进入陷阱,定时器停止工作并且关闭中断,从而使 Watchdog 复位电路会产生一个复位脉冲将 CPU 复位。当然这种技术用于实时性较强的控制或处理软件中有一定的困难。

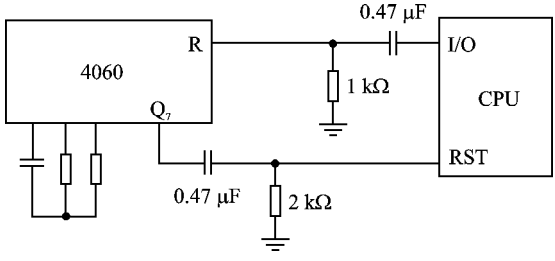


图 7.1－11 看门狗型复位电路

二、专用复位芯片简介(MAX813L)

目前,在市场上有许多流行的专用复位芯片,了解它们的工作原理对电路可靠性的分析及设计至关重要。以 Maxim 公司生产的 MAX813L 为例,解剖专用复位芯片的一般工作原理。对于其他芯片,可根据本文所提供的四种复位电路一一对其分析即可求得结论。

MAX813L 具有上电复位、Watchdog 输出、掉电电压监视、手动复位四大功能。具体原理框图如图 7.1－12 所示。本文局限于讨论复位电路部分及看门狗定时器部分。从图 7.1－12 中可以看出，WDI (Watchdog Input) 主要是作为 Watchdog 计时器重定用的。在 1.6 s 内若 CPU 不触发复位看门狗定时器，则 WDO (Watchdog Output) 将输出低电平。复位电路分为手工复位与上电复位。从原理图 7.1－12 中可以看出，上电复位与本文图 7.1－10 所提到的电路原理相同，即用比较器产生触发信号触发触发器，以此产生复位信号。同时，对时基产生的脉冲进行定时，当复位时间达 140 ms 时，Reset 发生器产生一脉冲使复位信号无效。上电复位时，只要电压低于 4.63 V，复位信号 Reset 就有效；当电源电压超过 4.63 V 时，Reset 信号仍将继续保持 140 ms 左右，以保证 CPU 复位可靠后无效。手动复位时，MR (Manual Reset) 接地时间不小于 150 ns，则可产生一个手动复位过程。即在复位端产生 140 ms 的有效复位信号 (高电平有效)。若将 WDO 端与 MR 连接，则可组成上电复位及看门狗复位电路。

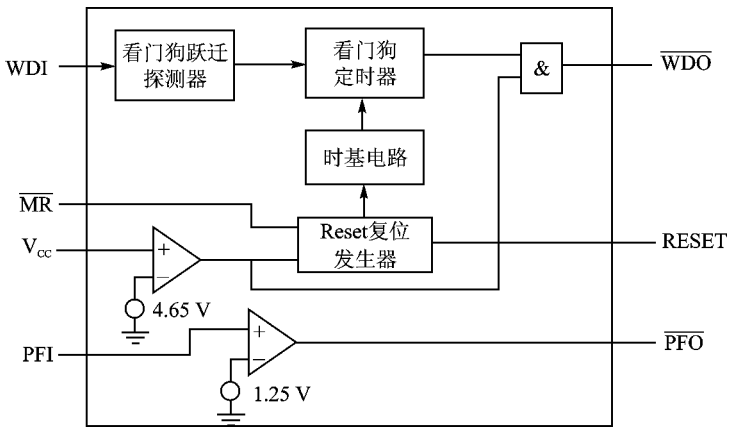


图 7.1－12　MAX813L 原理图

三、复位电路设计的注意点

本文所提到的各种复位电路中，微分复位电路简单，但易引入干扰且没有监控 CPU 运行的能力；积分复位电路简单可靠，但由于对电源电压波动不敏感，从而有可能出现 CPU 由于电源电压的瞬间过低而造成工作不正常的情况；比较器复位电路电路较复杂，工作可靠；Watchdog 复位电路电路较复杂，工作可靠并且具有监控 CPU 运行的能力。在使用中应根据电路板的空间、电源电压特性、系统运行现场等情况，综合考虑而定。一般有以下几条可供参考。

⑴ 在使用微分型复位电路并且使用稳压电源时，应考虑在电容输入端加入适当的电感以减少负载突变而引起的干扰复位脉冲的产生。在电路板空间有限的情况下可以选用此复位电路。

⑵ 在使用积分型复位电路时，一方面应着重考虑上电复位时电源电压的上升率，特别在电源电压上升率较小时，应考虑用较为复杂的比较型复位电路。另一方面应考虑电路是否有降压举措以降低功耗。若有则应考虑二极管的正向压降对复位电路的影响。

⑶ 在设计比较器型复位电路时，应着重考虑电源电压的波动性。当系统工作在恶劣环境下时，外界干扰的窜入可能引起毛刺电压，从而导致不正常的复位。为此有必要根据毛刺电压的峰峰值以及脉宽采取以下措施：① 当毛刺电压峰峰值没有达到电源电压的正常值与系统

正常工作所需最低电压值之差时,可适当降低比较器的复位电压下限;② 当毛刺电压峰峰值超过电源电压的正常值与系统正常工作所需电压之差时,一方面应采取措施降低毛刺电压,另一方面应采用较为复杂的比较器型上电复位电路(见图 7.1-10)。

(4) 在选用或自己设计 Watchdog 型复位电路时,应注意输入 Watchdog 的"喂狗"信号应该是时沿信号,而不是电平信号,同时应考虑撤销复位电压的电源电压值应大于系统最小正常电压值。

参 考 文 献

1 MAXIM Datasheet. Low-Cost, uP, Supervisory Circuits. 1995

2 何立民. MCS-51 系列单片机应用系统设计. 北京:北京航空航天大学出版社,1989

3 王柏林. 单片机系统设计的误区与对策. 电子技术应用,2002,28(2):22~25

选自《电子技术应用》月刊,2002 年第 11 期

7.2　提高移位寄存器接口电路可靠性的措施

山东科技大学 信电学院 (250031)
陶安利　蔺增金　王　霞　郭红琳

用移位寄存器扩展单片机开关量接口的显著优点是简化硬件设计,降低系统成本。但当移位寄存器增多时,数据传送的可靠性明显降低。主要原因:光电隔离器件造成的脉冲边沿抖动;移位寄存器对干扰的敏感和记忆;时钟信号的延迟。以下给出了三种提高移位寄存器接口电路可靠性的措施。

1. 消除光电隔离器件所造成的脉冲边沿抖动

图 7.2－1 是用移位寄存器 74HC165 和 74HC164 作开关量输入、输出接口,且加入 TLP521 进行光电隔离的电路,适用于传送波特率小于 10^4 bit/s 的应用场合。用存储示波器观察,经光电隔离后的信号上升、下降沿均有窄尖峰毛刺,毛刺宽度小于 1 μs,这种毛刺作用到移位寄存器的 CP 端会导致误移位。使用一阶低通 RC 电路和施密特触发器 74HC14 滤波、整形,即可消除此种干扰。一阶低通滤波器 RC 参数的选择应满足 $RC \leqslant 1.061 \times 10^{-5}$ (此式适用于消除 TLP521 所造成的脉冲边沿抖动)。

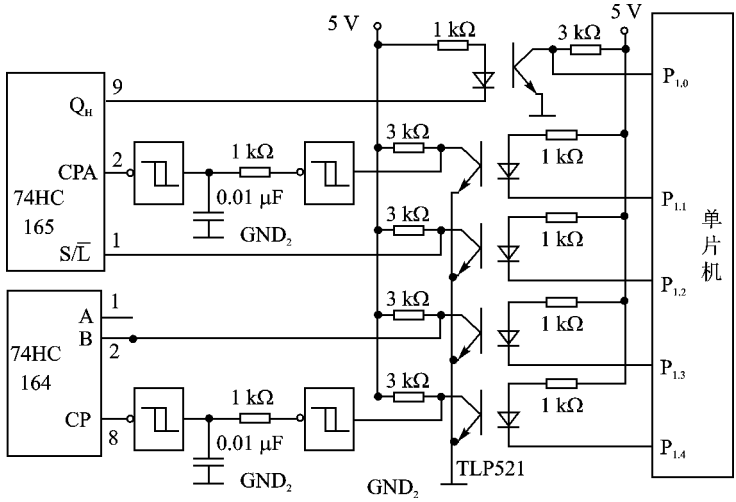


图 7.2－1　光电隔离移位寄存器输入输出接口

2. 减弱或消除基准线的不等位

建立基准线 (地线) 的分布参数模型,可得距基准线始端 x 远处的基准线电位:

$$\varphi(x,t) = \frac{\varphi_{1p}}{\cos \alpha l} \cdot \sin \omega t \cdot \cos \alpha x \tag{1}$$

式中,φ_{1p} 为由基准线始端引入的干扰幅值;$\alpha = \omega \sqrt{L_0 C_0}$,ω 为干扰的角频率;$L_0$ 及 C_0 分别为单位长度基准线的分布电感和其对机壳或屏蔽地的分布电容;l 为基准线长度。由式 (1) 知:

在干扰作用下,同一时刻 t 基准线上各处的电位并不一致,基准线不再为理想的等位线,在一定范围内基准线上某两点间的电位差随两点间距离的增加而变大,使得接于基准线上的各移位寄存器间无相同的电位参考点,相当于给移位寄存器的时钟端、数据端加入了等效信号,从而在干扰严重时产生误移位和错码。缩短基准线长度,使其满足式 (2) 则可消除由基准线电位波动造成的移位错误:

$$L < \frac{3}{\omega \times 10^{-8}} \sqrt{\frac{2u_{n\,min}}{\varphi_{1p}}} \tag{2}$$

式中:L 为应取基准线长度;$u_{n\,min}$为移位寄存器的最小噪声容限;其他变量的说明同式 (1)。如单片机系统处于有大电感负载切投的电网环境中,移位寄存器采用 CMOS 器件,工作于 5 V 电源下,则 $u_{n\,min}$ = 1.7 V;干扰信号取现场实测典型值;φ_{1p} = 100 V, $\omega = 2\pi \times 40 \times 10^6$, 算得L < 0.22 m。

缩短基准线长度的主要方法:以始端为中心将基准线排为放射状,使每段长度满足式 (2)。当工艺上难以实现时,可如图 7.2-2 所示,在过长的基准线中加入容量为 0.01 μF 的瓷片电容,使基准线上的电位尽可能相等,须注意电容的引线不能过长。

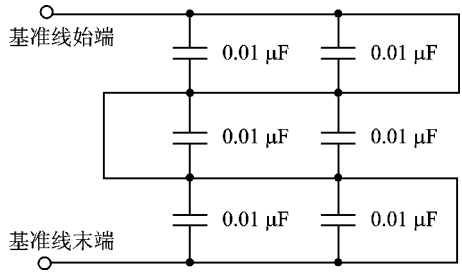


图 7.2-2 在过长的基准线上加入电容

3. 消除因时钟信号滞后造成的信息丢失

用移位寄存器扩展端口,其特点是时钟线公用,即各移位寄存器的时钟端并联接于时钟线上。而移位寄存器的数据端口却串接于数据线中,从而导致时钟线长度必大于每段数据线的长度,时钟线上的对地分布电容远大于每段数据线的对地分布电容 (因为挂于时钟线上的每一芯片端子的输入电容均要计入),带来的主要问题是时钟信号滞后于数据信号,可能造成数据丢失和错移位。图 7.2-3 以 74HC164 为例,描绘了此种错误。74HC164 (1)、74HC164 (2) 距时钟信号始端的连线较短,几乎无时钟滞后;而 74HC164 (3) 距时钟信号始端的连线过长且对地电容较大,时钟信号明显滞后。即在同一时钟脉冲 CP 作用下,74HC164 (2) 的 Q_H 状态已变化,CP 才到达 74HC164 (3)。设移位前各寄存器的状态如图 7.2-3 (a) 所示,连续发出 8 个移位脉冲后,各移位寄存器的状态如图 7.2-3 (b) 所示,结果丢失了信息 D_{23},重复了信息 D_{15};D_{16} ~ D_{22}未移入预期位置。

时钟信号的滞后时间 t_d 可由下式计算:

$$t_d = RC \ln \frac{u_{OH}}{u_{OH} - u_T} \tag{3}$$

式中:u_T 为移位寄存器的门槛电压;u_{OH}为时钟源输出的高电平;R,C 分别为时钟线的等效电阻和对地等效电容。使数据线做同样延迟并非解决此类问题的实用方法,原因是分布参数不

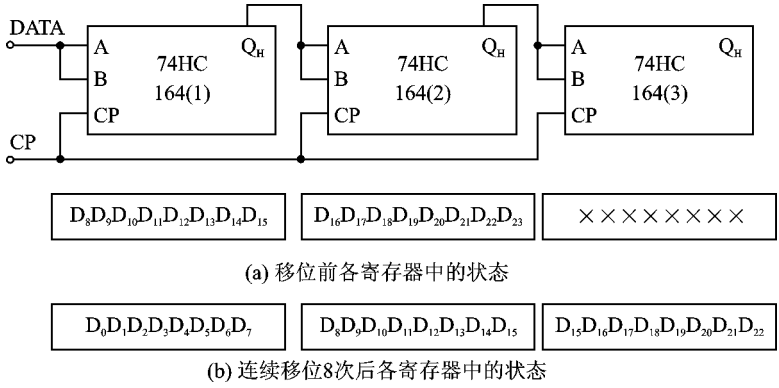


图 7.2－3　时钟线过长造成的移位错误

易计算和测准，且易受多种因素的影响，时钟信号与数据信号的延迟难以做到完全一致。有效地解决时钟信号滞后的措施是将移位寄存器分组，即空间上靠近的为一组，且将时钟信号经数据分配器引入，如图 7.2－4 所示。图中 74HC138 用作分配器，分配时钟脉冲。此法的主要优点是时钟线因移位寄存器分组而分段，长度缩短，时钟信号滞后明显减小，且干扰作用下的时钟线的不等位程度也得到了改善。

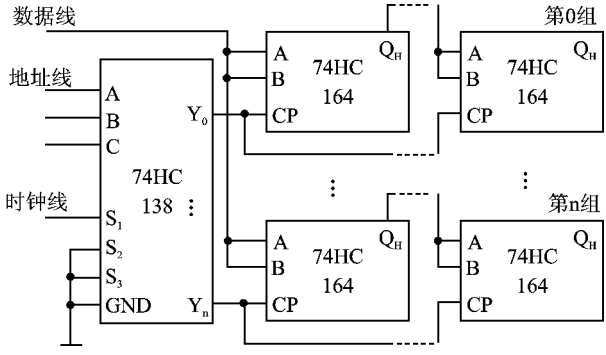


图 7.2－4　移位寄存器分组使用

以上三种方法已综合应用于由移位寄存器扩展的单片机开关量输入、输出控制系统中，开关量口线多达 160 个。该系统在电磁环境恶劣的工业场合中长期运行，未发生端口数据的输入、输出错误，而未采用上述措施的系统无法正常工作。实践验证了设计思路和方法的正确。

参 考 文 献

1　陶安利. 对地干扰与计算机基准线电位波动 [J]. 微电子学与计算机，1990 (3)

选自《自动化与仪表》月刊，2002 年第 5 期

7.3 单片机嵌入式系统软件容错设计

安徽淮南淮化集团设计研究院 (232038) 洪 滨
安徽淮南工业学院电气工程系 (232001) 周 莉

一、引 言

采用搏弈、冗余数字滤波等软件容错 (Software Fault Tolerance) 技术可以低成本地解决诸如干扰、元件老化或元器件选择不当造成的数据采集不可靠、控制失灵、程序运行失常等等故障。因此,软件容错技术 (SFT) 是嵌入式单片机系统电磁兼容性设计和可靠性设计的重要方法。

接地、滤波、屏蔽及抗扰芯片的选用,也是单片机系统电磁兼容性设计和可靠性设计的可选方法,但这类硬件抗扰措施是设备复杂度的函数,它消耗的是元器件与各种电工材料。而且,过度地引人硬件抗干扰措施会明显提高产口的常规成本。增加硬件设备的同时,还会使电路板复杂化,导致系统平均无故障工作时间缩短,甚至还会引出新的噪音和不恰当的电路耦合。

软件容错 (SFT) 技术是时间复杂度和空间复杂度的函数,SFT 设计消耗的是 CPU 的时间与存储空间。它在不提高嵌入式系统成本的同时,可大幅度地提高整机性能,故 SFT 对单片机系统电磁兼容性设计及可靠性设计具有特别重要的现实意义。

二、博弈技术

“博弈”概念取之于经济学的一种决策技术,即决策者试图预测竞争对手的可能行为和反应时的最优决策。软件博弈 (Software subgame) 技术借助经济学的博弈思想,在程序设计时,充分考虑来自不同渠道的各种干扰对监控程序和功能模块的影响及消除办法。

1. 软件屏蔽法

利用单片机的“空闲方式”,使 CPU 平时处在“磕睡状态”;当有任务时,用中断唤醒它,处理完工作后,再进入“打磕睡”状态。这样一来单片机系统既节省能源,又减少了沿传输线而来的随机干扰对系统的影响。

软件屏蔽法的具体做法是在主程序中加上一小段指令:

```
MOV   PCON,#01H ;IDL←1
NOP
NOP
```

上述指令段中的“P_{CON}”为电源控制寄存器,“IDL”为空闲方式控制位。“软件屏蔽法”特别适合于以 CMOS 型单片机 (80C31、80C51、87C51 等) 为核心,控制对象运行简单,而环境干扰又较大的实时控制系统。(关于“空闲方式”的工作原理,可参考有关产品样本。)

2. 软件陷阱法

软件陷阱法的指导思想是:把控制系统中未使用的单元,用某种“引导指令”填满,作为“陷阱”,来捕捉“弹飞”的程序,并强行将捕获的程序引向一个特定的地址,在那里有一段专门进行错误处理的程序,来恢复系统的正常运行。如果把该段错误处理程序的入口标号称为“ERR”,则软件陷阱的核心就是一条“LJMP ERR”的指令。为了加强其捕捉弹飞程序的效果,一般还在其前面加两条“NOP”指令,即“软件陷阱”一般由三条指令构成,它们是:

```
NOP
NOP
LJMP ERR
```

下面,针对 51 系列单片机的具体情况给出软件陷阱的实际用法。

(1) 对中断向量区设置软件陷阱

如果干扰信号,使未使用的中断开放,并激活这些中断时,就会导致监控程序运行混乱;但如果在这些地方设置软件陷阱,就能及时捕捉到错误中断,提高系统的可靠性。

设某嵌入系统共使用三个中断:INT_0、T_0、T_1,令其对应的中断子程序分别为 $PGINT_0$、PGT_0、PGT_1。考虑到对中断向量区使用软件陷阱,则可按如下方式进行设置。

```
START: LJMP    MAIN       ;引向主程序入口
       LJMP    PGINT0     ;INT0 中断正常入口
       NOP                ;冗余指令
       NOP                ;冗余指令
       LJMP    ERR1       ;软件陷阱
       LJMP    PGT0       ;T0 中断正常入口
       NOP                ;冗余指令
       NOP                ;冗余指令
       LJMP    ERR1       ;软件陷阱
       LJMP    ERR1       ;未使用的 INT1 中断入口
       NOP                ;冗余指令
       NOP                ;冗余指令
       LJMP    ERR1       ;软件陷阱
       LJMP    PGT1       ;T1 中断正常入口
       NOP                ;冗余指令
       NOP                ;冗余指令
       LJMP    ERR1       ;软件陷阱
       LJMP    ERR1       ;未使用的串行口中断入口
       NOP                ;冗余指令
       NOP                ;冗余指令
       LJMP    ERR1       ;软件陷阱
```

出错处理程序 ERR_1 仅需要一条中断返回指令 RETI。

(2) 对未使用的 ROM 空间设置软件陷阱

单片机外扩展程序存储器常常使用 2764、27128、27512 等 EPROM 芯片,但嵌入式应用系统很少有将其空间全部用完,剩余的大片未编程 ROM 空间,芯片都维持“OFFH”(原状态),而

“OFFH”对 MCS51 系列单片机指令来说,恰巧是一个单字节指令:“MOV R_7,A”。若程序弹飞到这一区域,将顺流而下,不再跳跃(除非又受到新的干扰)。为了系统的可靠,可在未使用的 ROM 空间,软件陷阱做以下设置:

```
    MOV   R7,A
    MOV   R7,A
    ……
    LJMP  ERR2
    MOV   R7,A
……
    MOV   R7,A
```

出错处理程序 ERR_2,应安排在此期间 030H 开始的地方。这样不管随机干扰怎样修改程序,编译后的 ERR_2 的地址总是固定的(因其前面的中断向量区为系统保留区,它总是固定的)。也就是说可以用“00 00 20 00 30”五个字节作为陷阱来填充 ROM 中的未使用区,或每隔一段设置一个陷阱:LJMP ERR_2(02 00 30)。其他单元保持 OFFH 不变。

(3) 对表格使用软件陷阱

单片机表格有两类:一类是数据表格供“MOVC A,@A + PC”或“MOVE A,@A + DPTR”两类指令使用。它的内容完全不是指令,仅仅是指令所使用的数据,可以不安排陷阱。另一类是散转表格供“JMP @A + DPTR”指令使用,其内容为一系列三字节指令 LJMP 或两字节指令 AJMP,单片机对此类表格可以在最后安排五字节的陷阱。即:NOP

```
NOP
LJMP  ERR2
```

其机器代码为:“00 00 20 00 30”。

(4) 对程序区设置软件陷阱

当程序运行到含有 LJMP、SJMP、AJMP、RET、RETI、ACALL、LCALL 等指令时,则会发生跳转,即:正常执行的程序运行到此类指令便不会继续往下执行(程序断裂点),PC 的值将出现正常转移。如果干扰使弹飞来的程序刚好落到断裂点的操作数上或断裂点下一条指令的操作数上时,则程序就会越过断裂点,继续往前冲,应用软件必然出错。

所以,在程序断裂点处设置软件陷阱可有效地捕获因干扰引起的跨断点弹飞,而不会影响正常执行的程序流程。

3. 软件看门狗

当程序受干扰,弹飞到一个临时构成的死循环中,软件陷阱就无能为力了。此时,系统将面临着完全瘫痪。而软件看门狗(Software watch dog)法可以成功地解决此类问题。软件看门狗实质上是一种监控定时器,它具有以下特征:

(1) 本身能独立工作,基本上不依赖于 CPU。

(2) CPU 在一个固定的时间间隔与该系统打一次交道(喂狗一次),以表明系统目前尚正常。

(3) 当 CPU 陷入死循环后,能及时发觉并使系统复位。

软件看门狗(SWD)技术具体实现方法如下:

首先,在初始化程序中设置 T_i 的工作方式,并启动其中断和计数功能。其次,计算各条指令执行时所耗时间,以适当的间隔设置 T_i 的初值。最后,设计 T_i 溢出所对应的中断子程序。此子程序只有一条指令,即 在 T_i 对应的中断向量地址(000BH)写入无条件转移指令,把 PC 拖回整个程序的第一行,对单片机重新进行初始化,并获得正确执行顺序。

设 T_0 为"监控定时器",定时 16 ms,则软件年看门狗程序如下:

```
        0RG     0000H
        AJMP    MAIN
        0RG     000BH
        LJMP    TOP
MAIN:   MOV     SP,#60H         ;主程序
        MOV     PSW,#00H
        MOV     SCON,#00H
        … …
        MOV     IE,#00H
        MOV     IP,#00H
        MOV     TMOD,#01H
        LGALL   DOG
        … …
DOG:    MOV     THO,#0B1H       ;喂狗程序
        MOV     TLO,#0E0H
        SETB    TRO
        RET
TOP:    POP     Acc             ;空弹断点地址
        POP     Acc
        CLR     A
        PUSH    Acc             ;软件复位
        PUSH    Acc
        RETI
```

可见:一旦单片机程序受干扰跑飞,便不能正常"喂狗",定时器 T_0 溢出,进入中断矢量地址 000BH,执行"LJMP TOP"指令,进入空弹程序 TOP。当执行完 TOP 程序后,就将其 0000H 送入 PC,从而实现了软件复位。

三、冗余技术

"冗余设计"原是系统可靠性设计的一种技术,常用于系统工程设计上。在单片机嵌入式系统的软件容错模块设计中,笔者充分利用冗余设计思想,有效的解决许多难点问题。软件冗余模块,可分为两类:其一是,"工作余软件模块",它利用冗余资源(CPU 时间,RAM 空间)把干扰的后果"屏蔽"掉,而不改变现行监控程序的进程;其二是"备用冗余软件模块",它在发现特定的随机干扰后,通过中断程序,调用对应的软件模块,恢复或重构数据结构。然后,再返回现行监控程序。

下面,简单介绍笔者在嵌入式系统应用软件开发中,常使用的几种软件冗余设计技巧。

1. 重复初始化

监控程序一般在上电、复位等操作时,对单片机及片外扩展接口芯片的各种功能、端口、方式、状态等采用永久性或临时性的设置,这就是通常用的初始化。考虑到可能的干扰入侵,为保证嵌入式系统的自恢复功能,可进行"重复初始化",即:在重要功能模块调用前,再一次对相应的控制寄存器设定动作模式,以加强该功能模块的健壮性。

2. 数据传输冗余

在工厂环境下,控制或采集端与上位机之间的串行传输可考虑给数据添加冗余位的方式,以延长数据代码之间的海明距离(Hamming);或采用循环冗余校验(CRC)手段,来增加数据的检测和纠正错误能力。

3. 重复执行

把重要的指令设计成定时扫描模块,使其在整个程序的循环运行过程中反复执行。这样,既使干扰信号改写了指令的内容,也能在受控设备的反应时间内自动恢复正常。(例如:对频率较低的传感器数据,可在有效时间内多次采集并比较;对控制外部设备的指令,则需多次重复执行以确保输出信号的可靠性。)

4. 标志冗余法

考虑到干扰信号有可能对 RAM 区的数据进行破坏,我们可以在 RAM 中每隔一定单元置入一些标志(标志冗余),而这些标志在初始化时便设置好。事故处理程序一开始,便检查这些标志是否正常;如不正常,说明数据已被破坏,转向事故处理程序。

具体做法是:在单片机内存(片内 RAM)中设立运行标志 FLAG(冗余标志)。若数据写在 2^k 的 RAM 空间,地址从 0000H 开始,每隔 256 个单元置一个标志 EEH。

"冗余标志"还可以完成单片机系统的冷、热启动的自动判别。"冷启动"就是清零启动,一切重新开始。它通常包括清除内外 RAM 中的所有中间运行数据、各种状态标志。"热启动"则是停电时保存的中间数据和各种有关标志为基础的条件下,继续运行。如果在 RAM 中将连续 5 个单元设为"AAH"作为正常标志。即正常开机时,初始化程序将 Flag 全部置为 AAH,然后,再执行运行程序。若软复位后,程序查到 Flag 全为 AAH,则是热启动,转而执行"事故处理程序"。

5. 指令冗余

为了抑制程序的弹飞,在多字节指令后增加两条没有意义的空指令 NOP,称之为"指令冗余"。单片机受到干扰后,往往会把操作数当作指令码来执行,从而,引起整个程序的混乱。由于程序弹飞到某一条单字节指令上,不会发生将操作数当成指令的错误,而弹飞到双字节或三字节指令的操作数量,程序就会发生将操作数当成指令码的错误,使程序继续弹飞,引起整个软件的崩溃。为了保证指令在干扰条件下不被拆散,在程序多字节指令后,通常,插入两个单字节的空操作指令 NOP,来抑制程序的弹飞。(冗余指令 NOP,在程序中不宜加得太多,否则,会使程序运行速度减慢。)

四、数字滤波

在嵌入式系统中,通过某种数学模型对采样信号进行平滑加工,消除各种干扰,提高有用信号在采样值中所占比例的软件方法叫数字滤波。数字滤波对消除信道干扰特别有效。

1. 程序判断滤波法

大多数物理量都是连续惯性变量,故相邻两次采样值之间的变化有一定的限度。程序判断滤波法的思路是:首先,根据实践经验确定出相邻两次采样信号之间可能出现的最大偏差 ΔY(它取决于采样周期 T 及参数 Y 的动态响应)。然后,对任意两次相邻的采样值进行相减得出随机增量。最后,让随机增量与最大偏差值进行比较,若超过最大偏差 ΔY,则表明该输入信号受到干扰,把它去掉;若小于最大偏差 ΔY,可将该信号作为本次采样值。

(1) 限幅滤波

设顺序采样时刻 T_i 所采集的参数为 Y(k),那么:

如果 $$|Y(k)-Y(k-1)|\leqslant \Delta Y,$$

则 $$Y(k)=Y(k)$$

即取本次采样值。

如果 $$|Y(k)-Y(k-1)|>\Delta Y,$$

则 $$Y(k)=Y(k-1)。$$

即取上一次采样值。

式中:Y(k)——第 k 次采样值;

Y(k-1)——第(k-1)次采样值;

ΔY——相邻两次采样值所允许的最大偏差,其大小取决于采样周期 T 及 Y 值的动态响应范围。

限幅滤波法就是把两次相邻的采样值相减求出增量,然后与两次采样允许的最大差值 ΔY 进行比较,若小于或等于 ΔY,则取本次采样值;若大于 ΔY,则取上一次采样值作为本次采样值。(该法主要用于变化比较缓慢的参数系统,如温度、位移等测量系统。)

(2) 限时滤波

设顺序采样时刻 T1、T2、T3 所采集的参数分别为 Y(1)、Y(2)、Y(3),那么,如果

$$|Y(2)-Y(1)|\leqslant \Delta Y$$

则 $$Y(k)=Y(2)$$

即取本次采样值。

|Y(2)-Y(1)|>ΔY,则 Y(2)不采用,但保留其值,并继续采样一次 Y(3)。

|Y(3)-Y(2)|≤ΔY,则取"Y(k)=(Y(2)+Y(3))/2"为本次采样值,以此类推。限时滤波是一种折衷的方法。它既照顾了采样的实时性,又顾及了采样值变化的连续性。

2. 平均值滤波法

平均值滤波法是对多次采样值进行某种数学处理,求得某种平均值,以减轻干扰对输入信号的影响。其数学模型不同,处理的算法不同,适合的领域也不同。

(1) 算术平均值滤波法

算术平均值数字滤波的数学模型是:

$$\overline{Y}=\frac{1}{N}\sum_{i=1}^{N}X(i)$$

其中:$\overline{Y}$——N 个采样值的算术平均值;

X(i)——第 I 次的采样值;

N——采样次数。

算术平均值滤波法的实质是把一个采样周期内 N 次采样值相加,然后再除以采样次数 N,就可得到该周期的平均采样值。该法主要用于对压力、流量等周期波动的采样值进行平滑加工;但对脉冲性干扰较大的采样值进行平滑加工尚不理想。

(2) 加权平均值滤波法

加权平均值数字滤波的数学模型是:

$$\overline{Y} = \sum_{i=0}^{N-1} C_i X_{n-i}$$

加权系数

$$\sum_{i=0}^{N-1} C_i = 1$$

其中:Y (k) ——第 k 次 N 个采样值的加权平均值;

$X_{(N-I)}$ ——第 I 次个采样值;

N——采样次数;

C_i ——加权系数。

加权系数 C_i 体现了各种采样值在平均值中所占的比例。一般来说采样次数愈靠后,取的比例愈大。这样可增加新的采样值在平均值中所占的比重。

加权平均值滤波法可突出信号的某一部分,抑制信号的另一部分。

(3) 滑动平均值滤波法

"滑动平均值滤波法"是改进型的算术平均值滤波法或加权平均值滤波法。具体的做法是:首先,在 RAM 中建立一个数据缓冲区;然后,按顺序存入 N 次采样的数据,并且每采样一个新的数据,就将最初采样的那个数据丢掉;最后,计算包括最新数据在内的 N 个数据的算术平均值或加权平均值。

(4) 中值滤波法

"中值滤波法"就是对某一参数连续采样 n 次 (n 一般取寄数),然后把 n 次采样值从大到小或从小到大进行排列,再取其中间值做为本次采样值。中值滤波法软件上比较容易实现,且对去掉偶然因素引起的波动或消除因采样器不稳定而造成的误差方面特别有效;但对快速变化过程的参数 (如流量) 则不宜采用此法。

(5) 去极值平均滤波法

"去极值平均滤波法"的思想是:连续采样 n 次后累加求和,同时找出其中的最大值与最小值,再从累加和中减去之。即:按 n - 2 个采样值求平均,可得"有效采样值"。具体作法有两种:对于快变参数,先连续采样 n 次,然后再处理 (需在 RAM 中开辟出 n 个数据的暂存区);对于慢变参数,可一边采样,一边处理 (不必在 RAM 中开辟出 n 个数据的暂存区)。

3. RC 低通数字滤波器

RC 低通滤波器的传递函数为:

$$G(s) = Y(s)/X(s) = 1/(\tau s + 1) \qquad (1)$$

式中 $\tau = RC$ 为滤波器的时间常数。

将式 (1) 离散后,可得 RC 低通数字滤波器的数字模型:

$$Y(k) = (1 - \alpha) Y(k - 1) + \alpha X(k) \qquad (2)$$

式中:X (k) ——第 k 次采样值;

Y (k - 1) ——第 k - 1 次滤波结果输出值;

Y(k)——第 k 次滤波结果输出值；

α——滤波平滑系数。

对滤波平滑系数 α 有：

$$\alpha = 1 - e^{-T/\tau} \tag{3}$$

式中：T 为采样周期，由采样系统确定；τ 为时间常数，由滤波器确定。由式(3)得：

$$\tau = T/\ln(1-\alpha) - 1$$

当 $\alpha \ll$ 时，则 $\tau \approx T/\alpha$。

由式(2)可用程序实现 RC 低通数字滤波器的设计。

4. 数字滤波应用的有关说明

把两种或两种以上的上述数字滤波器组合起来，就可组成复合数字滤波器或多数字滤波器。例如：算术平均值滤波加上中值滤波构成的复合数字滤波器，即可对周期性的采样值进行平滑加工，又可以消除随机的脉冲干扰。

一般来说，对变化比较慢的参数，如温度，可选用程序判断滤波法或中值滤波法；对变化比较快的脉冲参数，如压力、流量等，可选用算术平均值滤波法和加权平均值滤波法；对要求较高或较特殊的系统，可选用复合数字滤波法或特殊数字滤波器。

五、结　语

算术平均值滤波法和加权平均值滤波法，其滤波效果与选择的采样次数 N 有关，N 越大，则滤波效果越好，但消耗 CPU 的时间也越多。

数字滤波在热工和化工过程并非一定需要，应根据具体情况，分析、试验后加以选用。不适当地应用数字滤波，反而会降低控制效果。

参 考 文 献

1 王幸之. 单片机应用系统抗干扰技术. 北京：北京航空航天大学出版社，2000

2 潘新民，王燕芳. 微型计算机控制技术. 北京：人民邮电出版社，1999

3 李朝青. 单片机原理及接口技术. 北京：北京航空航天大学出版社，1998

选自《电气自动化》双月刊，2002 年第 2 期

7.4 键盘信息泄漏与防泄漏键盘设计

华北计算技术研究所 崔 屹

一、键盘工作原理概述

键盘是计算机中最通用的设备,也是除显示器外信息最容易被截获并被复现的设备。按照红黑分离式防信息泄漏原理,我们成功地开发了红黑分离式防信息泄漏键盘。

首先分析一下键盘的工作原理。现在的键盘主芯片只有 1 个。1 个键盘由专用芯片、按键和接口 3 部分组成。其中专用芯片提供主机接口、行线、列线及键盘分系统控制微程序;按键被安排在行列线的交叉点上;主机接口共 4 根线:电源、地、时钟、数据。工作原理如下。

⑴ 时钟和数据线在主机方和键盘方的引脚都是 OC 门,正常时电平为高。主机和键盘任何一方都可以把这两根线上的电平拉低。当两根线都为高时,键盘可以发数据;当时钟为低时,禁止键盘发送数据;当时钟为高、数据为低时,表示主机要发送命令,键盘要准备接收。

⑵ 加电后键盘开始自检,如自检正常,则向主机发出 AAH,并开始扫描按键。

⑶ 判断出有键按下后向主机发这一键的扫描码并开始计时,然后继续扫描。若 0.5 s 后,这个键仍未抬起,且没有新键按下的话,就要连续发这一键的扫描码:每秒 30 个。最多支持 3 个键同时按下。在 0.5 s 内若有新键按下的话,就为新键计时。

⑷ 待有键抬起时发这一键的结束码。

⑸ 收到主机发来的命令码后,键盘发 FAH 以应答,并开始执行这一命令。

键盘与主机通信的数据规则是:每组数据由 11 位组成:1 位启始位(逻辑 0)、8 位数据位(低位在前)、1 位校验位(奇校验)、1 位停止位(逻辑 1)。其数据位的数据格式为:

1	2	□1	9	10	11
启始位	数据位			校验位	停止位

时钟是键盘分系统发出的方波,周期约为 80 μs 下降沿有效,只在发码的时候才有时钟。每个键有 1 个扫描码。主机还会发一些命令。表 7.4-1 给出了每个键的扫描码。

表 7.4－1 键盘扫描码

键名	扫描码	键名	扫描码	键名	扫描码	键名	扫描码
F9	01	V	2A	P	4D	F7	83
F3	04	F	2B	－	4E	F5	03
F1	05	T	2C	" '	52	F2	06
F12	07	R	2D	{ [	54	F4	0C
F10	09	%5	2E	+ =	55	Caps Lock	58
F8	0A	N	31	Shift 右	59	Num Lock	77
F6	0B	B	32	Enter 主键盘上	5A	Scroll Lock	7E
Tab	0D	H	33	}]	5B	Alt 右	E0 11
~	0E	G	34	| h	5D	Ctrl 右	E0 14
Alt 左	11	Y	35	←BS 左清	66	- 左	E0 1F
Shift 左	12	^6	36	小键盘 1 End	69	+ 右	E0 27
Ctrl 左	14	M	3A	小键盘 4←	6B		E0 2F
Q	15	J	3B	小键盘 7 Home	6C	/小键盘	E0 4A
! 1	16	U	3C	小键盘 0 Ins	70	Enter 小键盘	E0 5A
Z	1A	&7	3D	小键盘. Del	71	End	(E0 12) E0 69
S	1B	＊8	3E	小键盘 2 ↓	72	←	(E0 12) E0 6B
A	1C	< ,	41	5 小键盘	73	Home	(E0 12) E0 6C
W	1D	K	42	小键盘 6→	74	Insert	(E0 12) E0 70
@2	1E	I	43	小键盘 8 ↑	75	Delete	(E0 12) E0 71
C	21	O	44	Esc	76	↓	(E0 12) E0 72
X	22	) 0	45	F11	78	→	(E0 12) E0 74
D	23	(9	46	+ 小键盘	79	↑	(E0 12) E0 75
E	24	> .	49	小键盘 3PgDn	7A	Page Down	(E0 12) E0 7A
$ 4	25	? /	4A	- 小键盘	7B	Page Up	(E0 12) E0 7D
#3	26	L	4B	＊小键盘	7C	Print Screen	E0 12 E0 7C
空格	29	: ;	4C	小键盘 9PgUp	7D	Pause	E1 14 77 E1 F0 14 F0 77

这是一个开放式的工业标准，PC 机的键盘都是这样的。其与主机的通信必须按上述标准执行。这为零配件的生产、维修、使用提供了极大的方便，但同时也使键盘按键造成信息泄漏成为了可能。

二、键盘信息泄漏的分析

为了验证键盘信息泄漏的电磁场的特性，进行如下试验：当键盘连续保持按下“H”键时，用频谱仪测量键盘与主机连接的信号线的传导发射特性，结果如图 7.4－1 所示。

当按不同的键时，频谱仪接收到的谱线发生频移。按信息的相关原理证明，所得的谱线与按键信息相关，说明其中含有键盘的扫描码信息。该信息为键盘编码，并将其定义为红信号。

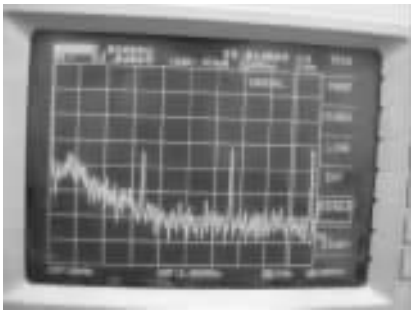
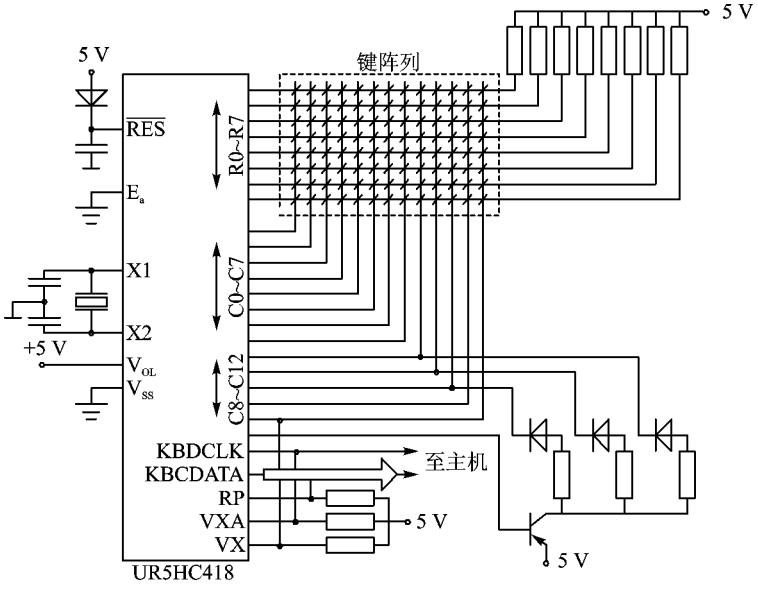


图 7.4－1 键盘的传导发射特性

下面具体分析一下键盘产生的红信号走过的路径。图 7.4－2 是普通键盘的电路图，是用 8051 单片机实现的。



注：空心箭头表示红信号通路

图 7.4－2 普通键盘电路

图 7.4－2 中键阵列部分的引脚（P0、P2 和 P1 的一部分）流过的是高低变换的电平，用以判断哪个键按下了，哪个键抬起了。这些信号即使被截获也是没有意义的，因此，将它们定义为黑信号。此外复位电平、晶振等也为黑信号。

键盘有 2 根信号线与主机相连，即时钟线（KBDCLK）和数据线（KBCDATA）。时钟线提供键盘与主机通信时的时钟信号，由键盘发出，下降沿有效。也就是说在每个时钟的下降沿，主机将键盘准备好的数据读入累加器“ACC”中，读到有效的“停止位”后送 CPU 处理。但对于同一种键盘来说，时钟的周期、频率、电平高低都是一样的；对于不同键盘会略有不同。在同一个键盘中，发出的所有数据的时钟都是相同的。所以这一信号与按键信息无关，也是黑信号。键盘有不同的键，它们被依此选通后，将通过数据线发出相应的键码数据传送给主机，所以，图 7.4－2 中只有数据线上走的是红信号。

下面再分析一下在芯片内部的红信号的通路情况。图 7.4－3 为 8051 的内部框图。

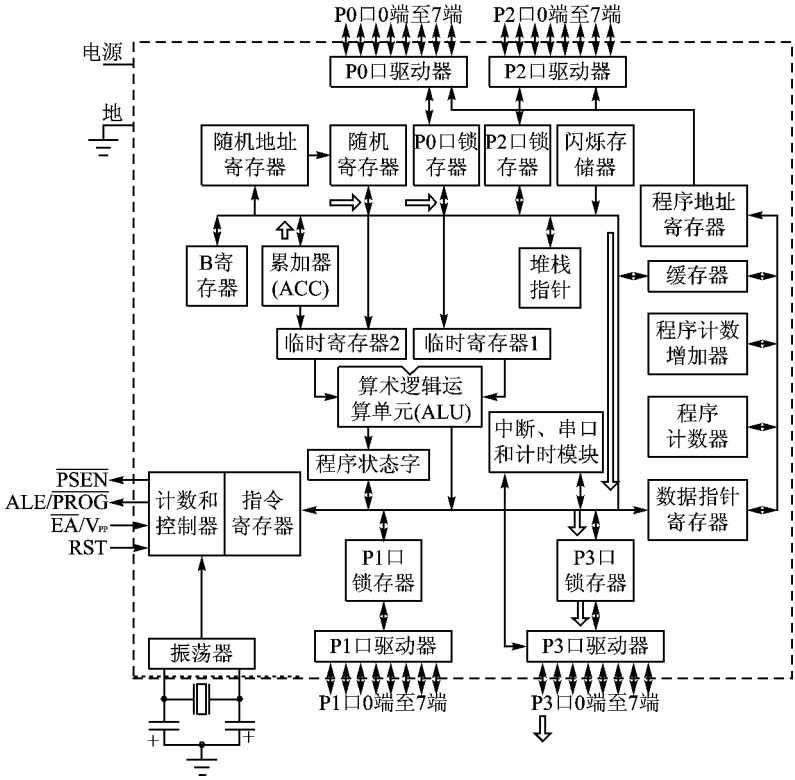


图 7.4－3　8051 框图

图 7.4－3 中以空心箭头表示红信号的路径。在 8051 内部，这一部分发出列扫描电平，读入行扫描电平，键按下后 ALU 通过计算将放在累中器 ACC 中。ACC 再一位一位地送到 P3 的某个引脚上。在芯片内部，这一红信号是串行二进制码数据，其波特率为 12.5 Kbps，脉冲宽度为 80 μs，转换时间为 1.4 μs，奇校验。具体波形如图 7.4－4 所示。

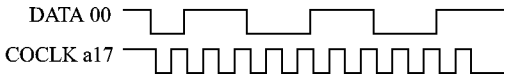


图 7.4－4　键盘发“H”的扫描码波形

通过上述试验可知，键盘中红信号的路径是从微处理器中的累加器开始，经一个数据引脚至主机数据口止的一段电路上。

键盘扫描周期谐波的 RF 辐射有两种主要的威胁：其一为攻击键盘电缆在其响应频率谐波的辐射；其二是攻击被非线性交叉效应调制的返回信号中被检波的扫描码。

三、红黑分离式防信息泄漏键盘

1. 设　计

为了预防键盘泄密，我们研制了红黑分离式防信息泄漏键盘。这种键盘使用光信号传输数据，键盘与主机间用塑料光缆连接，键盘以电池供电，使其最大限度地减小电磁辐射。

所设计的低电压电路,用 2 节 5 号电池供电。使用低电压的 8051 单片机作主芯片,实现键选扫并发送数据。为了省电,设计中采用一种技术,即在没有键按下时单片机处于休眠状态。

普通键盘的编码是固定的标准值,如表 7.4－1 所列。这种明码如果防御不当,一旦被截获将可被复现,造成严重后果。

这里的防御技术包括可编程的键盘微控制器。由于扫描周期是随机的,在它们传送给 PC 机前加密扫描码。当按键时,在周期内键扫描次数将是随机的,并且改变了值,而不是原来的常数。这样,当用户打印图案或所有情况下使用时,即使由攻击者截获到该值,但给他们的不是该按键值的信息。

为此需要修改系统,自己设置密码。我们改变了键盘微控制器的程序,即使对方能够探测到键盘在工作中的电磁泄漏信息,对于截获的信息也是没有意义的。因为键盘产生并发出的光信号不是通用的扫描码,如图 7.4－3 中的红信号通路中传输的不再是通用扫描码,而是密码,到主机方有一个相应的单片机接收密码,并转换成键盘标准码送往主机。主机方的这一机构密封在屏蔽机箱的内部,电磁波不会泄漏出来。加密编码还有一个好处,就是可以经常更换编码方式,这一点对于机要部门很有用。

这里只涉及 PC 机的设备驱动程序,并将其固化在键盘微控制器中。

下面主要讨论键盘的设备驱动程序或者称为键盘微控制器程序算法。

⑴ 实现的功能与第 1 节中描述的普通键盘的功能完全相同。

⑵ 键盘要连续扫描每一个键,遇到一个键按下了就要发出数据说明并开始记时。若在 0.5 s 之后这个键还没有松开,就连续发这一键,每秒 30 个。

⑶ 在记时过程和连续发码过程中,还不停顿地监视其他键。

⑷ 在连续发一个键时,若有其他键按下就停止发这个键,发新按下去的那个键并开始为其记时。

⑸ 若有键抬起,要发一个结束码给主机,告之这个键已抬起。

⑹ 在 RAM 中的位寻址区中固定 104 位,每一位为一个键提供标志。扫描到某个键时,无查看相应的位。这个键以前按下了,设这一位是 1 ;这个键以前没按下,设这一位是 0。然后再看现在按下没有,现在刚按下发扫描码,现在没按下就去处理下一个键,现在抬起来了就发结束码。

⑺ 这 104 个键要轮流扫描,同时要计时。如有键一直按着没抬起来,要在 0.5 s 之后连续发,每秒发 30 个。同时按下的键达到 4 个时,就不再记新的了。

根据上述算法键盘微控制程序编程流程如图 7.4－5 所示。

2. 实 现

红黑分离式防信息泄漏键盘中,单片机选择了低电压芯片,光缆选择了直径 1 mm 的塑料光缆。在键盘结构上按红黑分离规则进行了结构和电路两部分设计,做成了样机,并进行了小批量生产。图 7.4－6 是我们研制出的红黑分离式防信息泄漏键盘。

红黑分离式防信息泄漏键盘可靠性很高,信息安全测试技术指标可达 A 型机标准。用频谱仪测试该红黑分离式防信息泄漏键盘时,结果如图 7.4－7 所示。由测试结果可以看出已没有电磁辐射了。

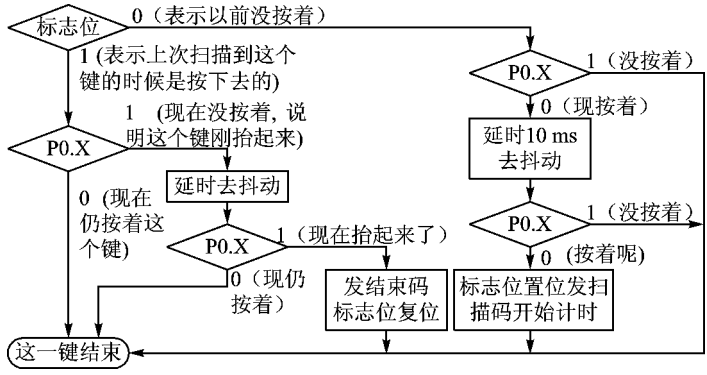


图 7.4－5　键盘程序流程

图 7.4－6　TEMPEST－III 型键盘

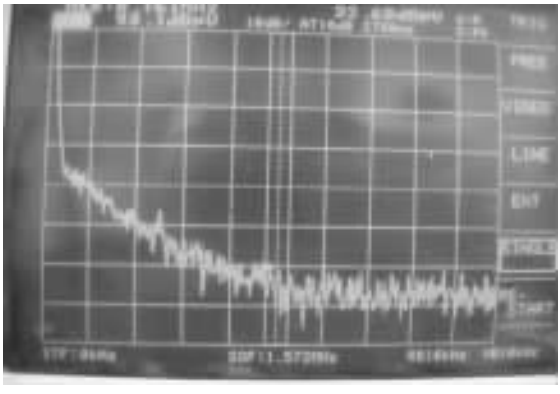


图 7.4－7　TEMPEST 键盘发“H”时的频谱

四、结束语

通过红黑分离式防信息泄漏键盘的设计使我们体会到，要设计一种防信息泄漏设备，必须先进行电路分析，找出红信号的泄漏路径。通过分析和论证，才能开始进行电路的红黑分离工作和软件编码的加密工作。在对电路红信号线的屏蔽、吸收、隔离和接地的设计中，一定要避免红信号被非线性交叉效应调制，这是本设计成功的经验之一。

选自《单片机与嵌入式系统应用》月刊，2002 年第 7 期

7.5 USB 安全钥功能扩展与优化设计

清华大学 Motorola 单片机与数字信号处理器应用开发研究中心 (100084)
马 伟

一、USB 安全钥的完整功能

USB 安全钥最早基于 USB 的热插拔、速度以及硬件等优势,结合加密算法,用于办公文件、软件等的存储和加密。但 USB 安全钥的用武之地远不止这些,与网络技术结合,用于时下最时尚的电子商务中,才使其大显神通。USB 安全钥结合传统的电子商务核心技术和新兴的 USB 技术,用于实现电子商务中的关键技术——身份识别,在未来电子商务领域具有广阔的应用前景。USB 安全钥集数据加密和数据存储两大功能于一体,推动了电子商务的发展。

传统的电子商务或是网络 email 等的身份认证基本上是通过两种方式来实现的。一种是密码机制,双方约定好规则。这是目前最为普遍的方式,但是这种方式的严重缺点显而易见。密码作为最重要的信息,在网络上传输,很容易被黑客攻击截获,经常发生密码被盗。第二种方式是通过第三方的认证,双方共同信任第三方公司提供的信息,从而进行交易。微软在 .NET计划中推出的认证服务器就提供这种服务。但是,信誉度建立在第三方上,便会受到第三方的制约,掏钱不说,还要担心第三方是否会倒闭。USB 安全钥解决了这两种方式无法解决的问题。

完整的 USB 安全钥系统由三部分组成:安全钥端,采用 Motorola 公司带 USB 接口的 8 位单片机 MC68HC908JB8 构成;PC 端,由任何一台可接入网络的 PC 构成,并安装 PC 端的用户身份认证软件;Server 端,任何一台网络服务器安装用于身份认证的 Server 端软件。

USB 安全钥系统结构体系及功能流程如图 7.5-1 所示,列出了九个步骤,描述了 USB 安全钥从插入 PC 到完成一次身份识别的完整流程。

需要强调的是,在上述步骤中,PC 仅仅起一个 Media (媒介) 的作用。任何重要的数据都没有经过 PC,在网络上传输的仅仅是 8 个字节的随机数 (它只在 Server 服务器和安全钥端有意义,只对特定的加密算法和密钥有意义),被黑客截取也不会有问题。这 8 个字节的随机数由网络 Server 产生,经由 PC 传递给 USB 安全钥加密;加密后的随机数再由 PC 不加任何改变地传递给 Server;Server 去调用解密算法解开加密的随机数,与原来未加密的随机数比较,如果相同则说明 USB 安全钥的持有者身份合理。整个身份认证也告结束。这里,USB 安全钥体现出两大优点:(1) 没有任何重要的个人信息在网上传递,保证了安全性;(2) Server 由网络商自己维护,安全钥由用户携有,双方的认证没有依靠第三方,快捷、安全、信誉度高。当然,USB 安全钥还有其他很多优点,例如可以在 PC 上热插拔,可以在任何一台支持 USB 的 PC 上工作 (现在几乎所有的 PC 都应该支持 USB) 等。

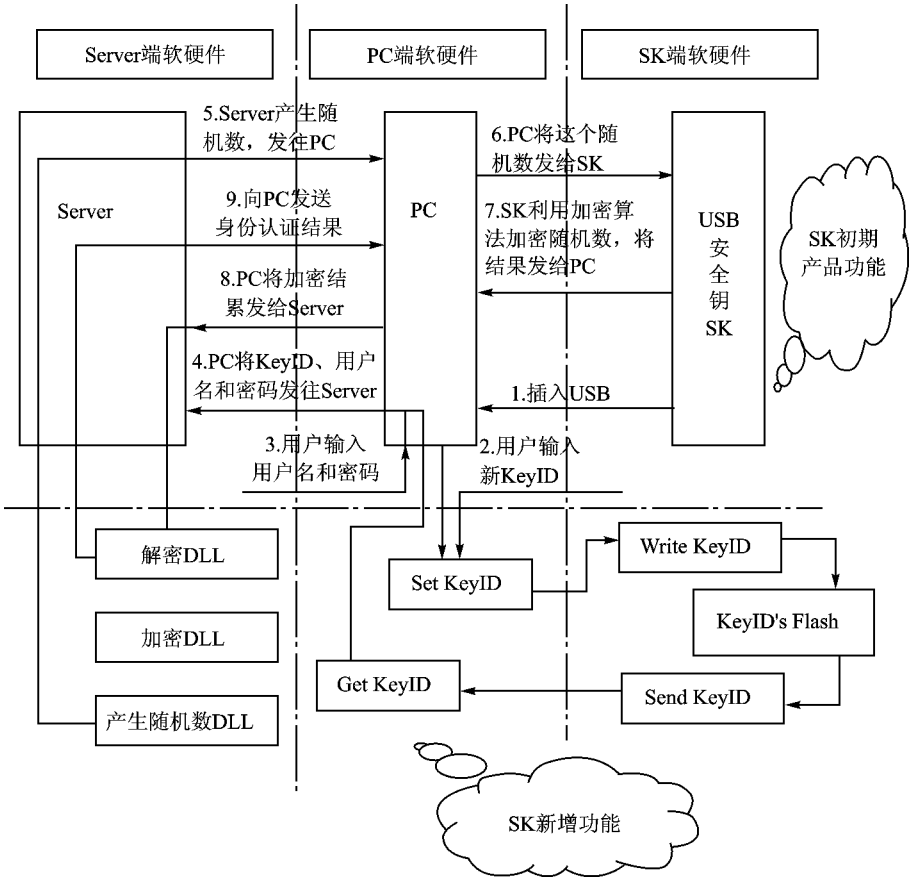


图 7.5－1 USB 安全钥完整功能图

二、USB 安全钥的技术细节

USB 安全钥技术，从设计上可以分为三个模块：Server 端的网络通信和加密算法设计、PC 端的 USB 驱动程序和网络通信设计、安全钥端的 USB 固件和加密算法设计。涉及到的计算机技术包括 Socket 网络编程技术、USB 驱动程序设计技术和加密算法技术。可以说整个设计内容庞杂，技术难度高。因此，设计时就需细化，一步步完成单个功能，再进行联调，将单个模块融合成完整的 USB 安全钥。

后期的功能扩展和优化设计也是针对三个模块，应用三大技术完成。主要是：服务器（Server）端 DES 加密算法的研究，设计加密算法的动态链接库 DLL，提供给客户最简单的 API；PC 和安全钥端驱动程序的研究，实现 PC 端友好的程序界面，动态在线修改存储在安全钥内的用户产品信息。本文将详细介绍扩展和优化的设计方法，从而揭示 USB 安全钥的技术细节。

1. 如何设计 Server 端加密算法及其 DLL

密码算法（Algorithm）就是指加密函数（Encryption）和解密函数（Decryption）。有加密函数，那么必然有一套与它对应的解密函数。现代密码学用密钥技术解决了保密性不够的问题。密钥用 K 表示。K 的取值范围叫做密钥空间。可以用如下式子来表示加密和解密函数之间

的关系：

$$D_K(E_K(M)) = M$$

其中，E 为加密函数，D 为解密函数，M 为被加密的原文。有一个重要的结论：所有算法的安全性都基于密钥的安全性，而不是算法细节的安全性。这就是说，算法可以公开，只要密钥是保密的，则这个算法就是安全的。简单地说，密钥就是与密文叠加在一起的一组数。

标准加密算法 DES 作为 ANSI 的数据加密算法和 ISO 的 DEA－1，成为世界范围内的标准已经 20 多年。就目前密码学的发展情况来说，DES 的安全性还是能够满足用户需求的。由于完整的 DES 算法相当复杂，这里仅简单介绍算法的结构。

DES 是分组加密算法，以 64 位为一组对明文进行分组，然后进行加密和解密。加密和解密的算法相同，只是密钥的编排不同。密钥长度为 56 位，通常是 64 位，但是每字节第 8 位都用来作为奇偶校验位，因此实际上只有 56 位。DES 共有 16 轮，即对同一组明文结合密钥进行 16 轮相同的加密过程，最终达到加密要求。

具体到每一轮的加密过程是这样的：每一轮中，密钥位移位，然后从密钥中选出 48 位。数据的 32 位右半部分数据扩展成 48 位，与密钥结合。然后再将这 48 位数据变换为 32 位，并与数据的 32 位左半部分相与后作为新的 32 位右半部分。而 32 位左半部分基本不变。最后，左右各 32 位数组合在一起便构成了一轮加密后的 64 位密文。重复同样运算 16 次，便完成了加密/解密功能[4]。

Server 端的加密算法采用 DES。加密和解密是整个 USB 安全钥身份认证的核心。在安全钥的初期产品中，已经实现了 DES 算法下的加密功能。但是，作为产品，其安全性是第一位的。而且，对于要将加密算法嵌入自己系统的用户来说，提供给他们大量的加密算法的源供码是不合适的。要对 DES 算法进行修改，将其从 Server 端的源程序中提出，改掉原来复杂的调用机制，改为提供给用户三个简单的接口函数：产生随机数、加密和解密函数、实现 DES 加密算法的 DLL。

动态链接库（DLL）是一个包含了若干个函数的可执行模块，Windows 应用程序可以调用这些函数来完成实际任务。对于调用 DLL 的用户来说，利用的资源仅仅是应用函数接口和一个后缀为. dll 的文件，实现加密算法的模块化。

在建立了一个 VC 工程之后，需要建立主程序头文件 KeyDll. h，加入如下代码。这些代码中定义了导出的四个函数。

```
class_declspec (dllexport) CKeyDllApp
{public:
    BOOL GetChallenge () ;
    int * Challenge () ;                    //导出函数
    int * DecryptData (BYTE []) ;           //导出函数，需要解密的随机数，可存储在数组 InputNum [8] 中。
                                              此函数输出值即为加密后的数据，输出格式为数组 DESDeData
                                              [8]
    int * EncryptData (BYTE []) ;           //导出函数，需要加密的随机数，可存储在数组 InputNum [8] 中。
                                              此函数输出值即为加密后的数据，输出格式为数组 DESEnData
                                              [8]
    BOOL cha_gen ;
    void DESDecrypt () ;                    //BYTE * Data, BYTE * Key) ;//解密函数定义
```

```
    void DESEncrypt ();                    //BYTE * Data, BYTE * Key);//加密函数定义
    BOOL Init();
protected:
    BYTE DESKey[8];                        //密钥
    BYTE IniDeData[8];                     //外部输入的需要解密的数据
    BYTE IniEnData[8];                     //外部输入的加密前的随机数
    BYTE DESDeData[8];                     //解密后的数据
    BYTE DESEnData[8];                     //加密后的数据
    WORD subkey[16][48];                   //子密钥
    BYTE challenge[8];
……}
```

然后,在主文件 KeyDll.cpp **中实现各功能函数的具体功能,主要是算法的实现。**

```
BOOL CKeyDllApp::GetChallenge ()           //这是产生随机数的函数,它调用 API 的函数 srand(),最
                                           终产生的 8 位随机数存在数组 challenge[8]中
{
    int i;
    srand((unsigned) time(NULL));
     if(!cha_gen){
        for(i=0;i<8;i++){
        do{challenge[i] = (rand()/256);}
        while((challenge[i] == ht)||(challenge[i] == 0)||(challenge[i] == 255||(challenge[i] ==
256 - ht)));}
        challenge[8] = 0;
        cha_gen = TRUE;
        return TRUE;}
    return FALSE;}

void CKeyDllApp::DESDecrypt ()             //解密函数,完成对已加密的 8 位随机数的解密功能
{
    WORD TempInput[64],TempOutput[64],TempKey[64];
    stringtobit(IniDeData,TempInput);
    stringtobit(DESKey,TempKey);
    decry(TempInput,TempKey,TempOutput);
    bittostring(TempOutput,DESDeData);}

void CKeyDllApp::DESEncrypt ()             //加密函数,可完成对 8 位随机数的加密功能,然后
                                           可与原随机数比较,看是否相等
{
    WORD TempInput[64],TempOutput[64],TempKey[64];
    stringtobit(IniEnData,TempInput);
    stringtobit(DESKey,TempKey);
    encry(TempInput,TempKey,TempOutput);
```

```
    bittostring (TempOutput,DESEnData) }

int * ,CKeyDllApp ::DecryptData (BYTE InputDeNum [8])   //导出的获取解密数据的函数。此函数需要
                                                          赋值——已加密了的 8 位随机数,并进行解
                                                          密,最终函数值为解密后的 8 位随机数
{
        int i;
        for (i = 0 ;i < 8 ;i + + )
        IniDeData [i] = InputDeNum [8] ;
        return (int * ) DESDeData }

int * CKeyDllApp ::EncryptData (BYTE InputEnNum [8])    // 导出的获取加密数据的函数。此函数需要
                                                          赋值——8 位随机数,直接调用并赋 8 位随
                                                          机数后,此函数将调用加密函数并进行加
                                                          密,最终函数值为加密后的 8 位随机数
{
    int i;
        for (i = 0 ;i < 8 ;i + + )
        IniEnData [i] = InputEnNum [i] ;
        return (int * ) DESEnData }
```

编译、连接后将产生一系列文件,在加上源工程文件,将会有数量比较宠大的文件系统。最终,只需提供给用户 3 个文件即可,它们是:

- hKeyDll hDebug hKeyDll. dll,这是 DLL 文件;
- hKeyDll hDebug hKeyDll. lib,这个文件将在应用 DLL 的程序编译和连接时,提供连接向导;
- hKeyDll hKeyDll. h,这个头文件告诉用户此 DLL 中导出了哪些量可以用。

DES 的 DLL 导出了一个类 :CkeyDllApp。在这个类中共有 4 个导出函数可以导入应用程序中,用户在导入了加密 DLL 后,可以在自己的程序中直接调用以下函数:

- BOOL GetChallenge () ,用于在应用程序支持循环结构;
- int * Challenge () ,产生随机数,并存储在 Challenge [8] 中;
- int * DecryptData (BYTE []),用于解密随机数;
- int * EncryptData (BYTE []),用于加密随机数。

2. USB 安全钥新增功能描述

USB 安全钥和 PC 传输的数据量不大,而且没有很高的速度要求。因此,在编写固件时就将其归类为 HID (USB 的人机接口设备类)。在编写 PC 端的驱动程序时可以直接调用 Windows 提供的 HID 的 API 函数,大大降低了编程的难度。更重要的是,Windows 对 HID 设备的支持非常完备,不需要用户再编写底层的驱动。

安全钥端的设计内容主要是 :实现在线修改存储在安全钥内的 KeyID 和读取 KeyID 两个功能,分别由函数 Set_KeyID 和 Get_KeyID 实现。KeyID 是安全钥的标识符,在安全钥插到 PC 上后,被读出并送往 Server 进行检查。在初期产品中,KeyID 只能是安全钥首次接到 PC 上读

取,且不能更改,这为厂家和开发者造成了不便。因此要更改初期产品中的 KeyID,就必须修改安全钥端的汇编程序,然后再“烧”写到安全钥中,非常麻烦。新增功能可实现 KeyID 的在线修改。

PC 端的设计包括两步:首先要实现在 PC 上读取安全钥内的 KeyID。通过安全钥的端点 1,8 个字节的 KeyID 被周期地送出。PC 要获取这些数据,调用 HID 类库 Get_Report (Feature)。从安全钥发来的包含 KeyID 的包的特性及技术指标如表 7.5-1 所列。

表 7.5-1 调用 HID 类库获取 KeyID

bmRequestType	bRequest	wValueL	wValueH	wIndex	wLength	Data
10100001B ($A1)	Get_Report ($01)	Zero	Feature ($03)	Zero	8	KeyID

第 2 步,在 PC 上实现修改 KeyID 功能。调用 HID 类库 Set_Report (Feature),将新的 KeyID 发送到安全钥中,具体指标如表 7.5-2 所列。

表 7.5-2 调用 HID 类库发送 KeyID 到 SK

bmRequestType	bRequest	wValueL	wValueH	wIndex	wLength	Data
00100001B ($21)	Set_Report ($09)	Zero	Feature ($03)	Zero	8	KeyID

3. 如何设计安全钥端新增功能的 USB 固件

USB 固件 (Firmware),就是 USB 安全钥硬件上采用的单片机和其他处理器中有关 USB 通信的程序。这里采用 Motorola 公司的 8 位单片机 MC68HC908JB8 作为 USB 安全钥的控制器芯片。MC68HC908JB8 带有 USB 接口,8 K 的 Flash,支持 USB1.1 版本中的低速 (Low Speed) 设备,资源有限,主要用于实现 USB 通信,价格比较低廉。因此,很适合于 USB 安全钥。MC68HC908JB8 中 USB 通信的程序模块,包含在实现 MC68HC908JB8 所有功能的汇编程序中。

图 7.5-2 是经典的 USB 固件的流程图。考虑到 USB 安全钥中 USB 数据通信量很小,不需要考虑通信时间,采用中断传输方式。整个程序就是在等待数据传输要求的中断到来,从而进入数据传输模块。读/写数据缓冲区,往 USB 端点 (Endpoint) 中读/写数据,交给 ISB 模块收发数据。当 USB 安全钥不需要传输数据时,就进入挂起状态 (Suspend)。在得到 PC 主机远程唤醒后启动,继续工作。

新增功能中,主要完成的两个功能就是 KeyID 的读取和修改,即实现 Get_KeyID 和 Set_KeyID 功能。程序构思大致是:对于 Get_KeyID,在接收到 PC 端发来的读取 KeyID 的中断后,立即从端点 1 发送 8 字节的 KeyID,这一段没有什么特别之处;对于 Set_KeyID,在接收到信号后,立即转入 Set_KeyID 子程序。首先将存储 KeyID 的 Flash 去保护,然后寄存器置位,即在硬件上给 Flash 一个高电平,接着进行擦除,再将保存于缓冲区的 PC 发来的新 KeyID 存储到 Flash 中。最后,置 Flash 状态寄存器位,给 Flash 加保护。

4. PC 端新增功能的 USB 驱动程序设计

Windows 98 的驱动程序从结构上来说分为两层:内核层和用户层。USB 的客户驱动程序

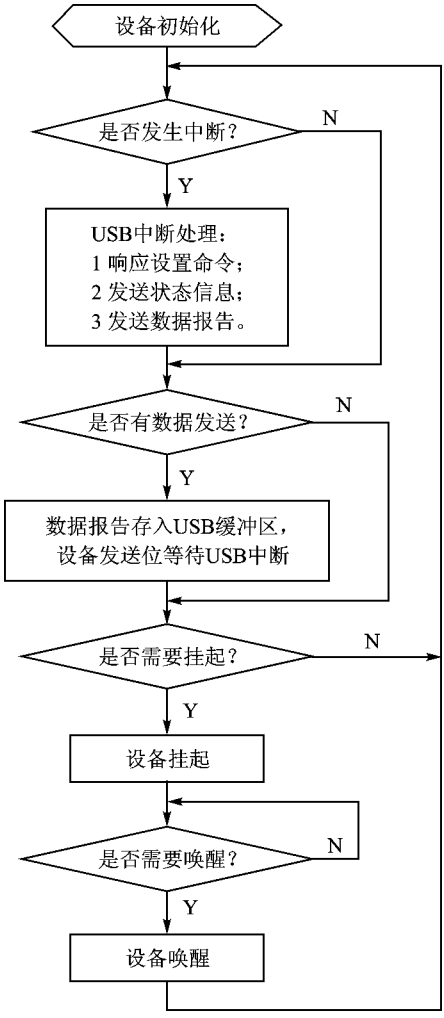


图 7.5-2 USB 设备端程序流程图

属于用户层,而 USB 类驱动程序和底层驱动程序则属于内核层。目前,USB 还属于起步阶段,Windows 对 USB 的支持还不够完善,仅支持内核层。USB 开发人员所要做的,就是开发客户驱动程序,直接与类驱动程序打交道。

HID 属于 USB 设备类中的一个子类,Windows 对它提供了非常强大的支持,尤其是在用户层提供了 Hid. dll,其中包含了用户层驱动程序与类驱动程序通信需要的各种功能模块,将它们以 API 的形式提供给用户函数接口。这样,在编写客户驱动程序的时候就可以直接调用这些 API 函数来完成诸如 IN、OUT 等功能,大大降低了编写驱动程序的难度。

HID 客户驱动程序访问 HID 类驱动程序,由 HID 类驱动程序完成大多数工作,而硬件交互由 HID 小驱动程序 HidUsb. sys 处理,HID 小驱动程序调用 USB 底层驱动程序 USBD. sys 访问设备。

这里介绍用户模式的 HID 客户驱动程序设计过程。它主要包括 3 个方面的工作:

· 查找所有 HID 设备;

· 对于查找到的每一个 HID 设备,检查其功能,判断是否为感兴趣的设备;

· 根据用户需要读取 HID 输入 Report (Feature) 或者写 HID 输出 Report (Feature)。

程序流程如下:

(1) 查找 USB 安全钥设备;

(2) 读取 HID 设备功能;

(3) 具体实现 Get_KeyID 和 Set_KeyID 子函数;

```
BOOL    CUsbDey::GetKeyID()                          //Get_KeyID 子程序
{…
  result = HidD_GetFeature (HidDevice,ReadBuffer,0x09);  //调用此函数,获取从端点 1 发来的 8 字节 KeyID
  for (tmpInt = 0;tmpInt < ReadBuffer [1];tmpInt + +)    //将数据存入缓冲区
    KeyID_Get [tmpInt] = ReadBuffer [tmpInt + 1];
  return TRUE;
}
BOOL  CUsbKey::SetDeyID()                          //Set_KeyID 子程序
{  int i;
```

```
    long result;
    int DataBuffer[16];
    WriteBuffer[0] = 0;                          //写缓冲区首字节清 0,作为 Set_Feature 函数的要求
    char *c;                                     //获得对话框内输入 8 字节新 KeyID 字符串的指针
    c = (char *)(LPCSTR) str_KeyIDSet;
    for(i = 0;i < str_KeyIDSet.GetLength();i++)  //将新 KeyID 存入数据缓冲区
      DataBuffer[i] = *c++;
    ……                                          //此处省略了对输入的 8 个字节的 KeyID 的 16 进制检查代码
    for(i = 0;i < 8;i++)
      WriteBuffer[i + 1] = DataBuffer[2 * i] + DataBuffer[2 * i + 1];
    result = HidD_SetFeature(HidDevice,WriteBuffer,0x09);
    return TRUE;
}
```

(4) 程序运行结果。

编译连接之后,最终会生成可执行文件 KEYDEMO.exe。执行它即可 SK 通信,实现各种功能。

参 考 文 献

1 王云飞. USB 系统研究. 研究生论文,北京 清华大学,2001

2 MOTOROLA. MC68HC908JB8 Technical Data. 2000

3 (美) Chris Cant. Windows WDM 设备驱动程序开发指南. 孙义译,北京 机械工业出版社,2000

4 (美) Bruce Schneier. 应用密码学——协议、算法与 C 源程序. 吴世忠译,北京 机械工业出版社,2000

5 USB Implementers' Forum. Universal Serial Bus Specification, Revision 1.0. January 15,1996

选自《电子技术应用》月刊,2002 年第 7 期

7.6 单片机多机冗余设计及控制模块的 VHDL 语言描述

南京大学物理学系(210093) 刘先昆 潘红兵 纪圣谋 徐健健

本文提出一种表决式单片机多机冗余设计方案。该方案不同于中央系统的多机冗余设计。大规模系统冗余大多采用完善而复杂的机间通信协议实现系统重构,不太注重系统的实时性。本方案结构简单,易于实现,具有极强的实时性,没有电子开关切换总线的咔嗒声输出。单片机价格低廉、功能灵活,也使得该设计在类似仪器仪表的小系统中的运用成为可能。

一、设计原理

设计结构如图 7.6-1 所示。完成整个冗余设计的电路被置于 1 个核心控制模块中,如果该模块以 FPGA 实现也就是一块芯片。图 7.6-1 中单片机 1、2、3 被假定为冗余的 3 个单片机,它们的输入总线并联,接收核心控制模块中输入缓冲的输出。输出总线分别接到模块的输出总线仲裁器。核心控制模块包括输入缓冲、输出总线仲裁、电源控制、时钟产生、复位电路和报警控制输出 6 个部分。

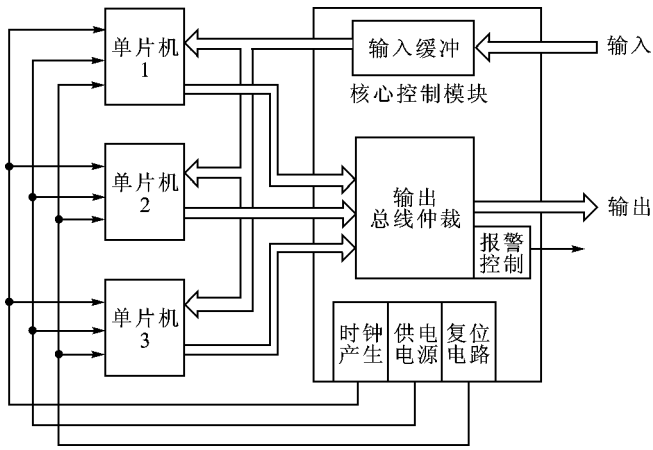


图 7.6-1 设计结构框图

1. 输入缓冲

为了消除输入端并联输入阻抗带来的影响,在输入端增加了一级缓冲器,减小外围电路的影响。采用输入缓冲,可以实现单片机和外围电路的输入隔离。

2. 输出总线仲裁

该总线仲裁是建立在所有单片机在时钟级上同步的基础上,通常采用总线表决法。即相同输出总线上的值作为仲裁的结果输出,不同输出总线被当作出错而封止,所有的输出皆不相同则是失败状态,无表决输出。表决的实现当然不能采用软件比较,以 3 个单片机系统的一位为例介绍表决方法。假设位输入变量 X1、X2、X3,输出 Q,状态指示:正常 N、X1 出错 E1、X2

出错 E2、X3 出错 E3。真值表如表 7.6－1 所列，位仲裁单元如图 7.6－2 所示。

表 7.6－1　真值表

X1	X2	X3	Q	E1	E2	E2
0	0	0	0	0	0	0
0	0	1	0	0	0	1
0	1	0	0	0	1	0
0	1	1	1	1	0	0
1	0	0	0	1	0	0
1	0	1	1	0	1	0
1	1	0	1	0	0	1
1	1	1	1	0	0	0

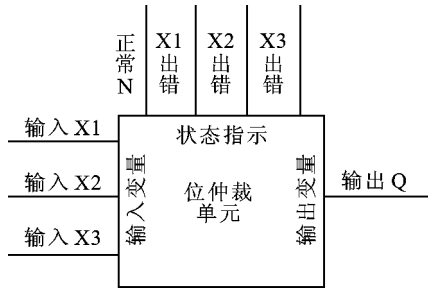


图 7.6－2　位仲裁单元图

显然以上位单元用数字电路不难实现，后面给出整体的 VHDL 语言描述。总线仲裁由多个这样的位单元组成，个数由单片机输出总线的最大数 n 决定。仲裁器除了 n 根输出线，同时还对每个位单元的状态位进行逻辑组合输出正常、出错、失败三种状态指示。失败信号也用作报警保护控制输出，或重新复位输出。失败输出有效时输出失效。

以 3 个单片机的系统为例，如果将仲裁器的 3 个总线某一时刻输入看作为 n 位二进制变量 X，Y，Z。如果 X，Y，Z 在任何时候都逐位相同，则系统处于正常工作状态。如果三者中有两个变量逐位相同，而另一个不同，则系统处于出错状态。如果三者皆不相同则系统失败。正常和出错状态可以运行，而失败状态必须保护和处理。

FPGA 技术的发展，使得设计中的比较、决策等数字电路的设计实现变得非常容易，而且系统简明可靠。如果采用中规模集成电路来实现的话，将相当烦琐和复杂。

3. 单片机时钟级同步的实现

系统的所有单片机必须达到时钟级的同步。单片机选用相同的型号（可以是不同的厂家），完全相同的程序和同一机器时钟。

同一时钟是实现时钟同步的第一步。时钟发生电路在控制模块内产生并送到各单片机的时钟输入端，要求单片机可外接时钟输入。时钟同步并不容易，以 89C51 为例，51 系列单片机上电后振荡器起振输出，ALE 脉冲由时钟经分频电路得到，一旦形成，机器周期脉冲和时钟脉冲相位关系固定，不受复位电路影响，直到电源掉电为止。

第二步是实现机器周期脉冲同步。MCS51 一个机器周期包括 6 个状态周期,每个状态周期包括 2 个节拍,对应 2 个时钟节拍有效期。也就是说一个机器周期包括 12 个振荡周期,指令工作在时钟节拍上,同时更是同步工作在机器周期上。不论是单字节指令还是双字节指令,指令周期均是机器周期的 1、2、4 倍。要同步单片机节拍,必须同步机器周期。考虑到上电时间上可能产生的差异,采用先上电后加时钟脉冲的方法。上电时确保时钟输入端没有干扰脉冲引入,所有单片机上电后的内部分频电路起始点一致,然后加入时钟脉冲,各单片机获得同步的机器周期。

第三步是同步指令周期。指令的同步需要依靠复位电路来实现。在时钟脉冲正常输入和分频电路正常工作的情况下,复位操作是在复位端加上至少 2 个机器周期的复位电平而实现的。复位信号由核心控制器发出送至每片单片机。复位后,统一了片内主要寄存器内容,所有单片机程序从起始位置开始执行。

单片机时钟级同步的实现主要依靠电源控制、时钟产生、复位电路三部分硬件。

(1) 电源控制

3 个单片机的供电电源由控制模块控制。主控元件需保证足够电流容量,可采用功率三极管或场效应管实现。不能采用继电器,以避免触点电流跳变。

(2) 时钟产生

晶体振荡器输出脉冲作为单片机时钟,中间增加可控的缓冲级。缓冲级可以增加时钟信号的输出负载能力,并可被控制模块控制。

(3) 复位电路

3 个单片机的复位端并联接至同一个复位端。复位信号在信号极性和脉冲宽度上满足单片机复位要求,驱动能力满足多单片机需要。复位电路同样是受控于控制模块,用以实现单片机同步。

4. 报警与控制

不同状态下核心控制模块有不同的信号输出,异常状态同时也是报警信号。正常状态输出绿灯,出错状态输出黄灯,失败状态输出红灯。黄灯输出时系统可以暂时继续工作,等到系统空闲或许可时进行纠错。红灯输出时系统立即进入保护状态,输出端呈现高阻状态,需要时可以马上纠错,恢复系统。

系统恢复需要对控制模块进行复位,复位脉冲可以是自身的失败状态输出,也可以是出错脉冲输出和其他信号的组合逻辑。控制模块的复位,实际是对各单片机重新进行时序对齐和复位单片机程序。此处设计需结合具体使用场合考虑。

二、控制模块的 VHDL 语言描述

本控制模块主要采用 VHDL 语言进行描述。

```
library ieee;
use ieee. std_logic_1164. all;
use ieee. std_logic_unsigned. all;

Entity redu_control is
Port ( a_bus,b_bus,c_bus :in std_logic_vector (7 downto 0) );        --三输入总线,本设计定为 8 位
```

```
    o_bus :out std_logic_vector (7 downto 0) ;          --8 位输出总线
    error_out,fail_out :out std_logic ;                 --出错、失败输出
    reset_in, clock_in :in std_logic ;                  --复位、时钟输入
    power,clock,reset :out std_logic ;                  --电源、时钟、复位输出
    )
end ;

architecture control_pro of redu_control is
signal int :std_logic ;
begin
    bus_pro :process (a_bus,b_bus,c_bus)                --总线控制过程
    begin
    if a_bus = b_bus then
        o_bus < = a_bus ;
        if a_bus = c_bus then                           --正常输出
            error_out < = '0' ;
            fail_out < = '0' ;
        else
            error_out < = '1'                           --给出出错信号
        fail_out < = '0' ;
        end if
    elsif a_bus = c_bus then
        o_bus < = a_bus ;
        error_out < = '1'                               --给出出错信号
        fail_out < = '0'
    elsif b_bus = c_bus then                            --不同的出错情况
        o_bus < = b_bus ;
        error_out < = '1' ;
        fail_out < = '0' ;
    else                                                --失败输出
        o_bus < = (others = > 'z') ;
        fail_out < = '1' ;
    end if
end process bus_pro ;                                   --总线过程结束
start_pro process                                       --启动过程
begin
    wait until reset_in = '1'                           --等待外部复位启动
    power < = '0' ;
    clock < = '0' ;
    reset < = '0' ;                                     --停止电源、时钟、复位输出
    power < = '1' after 3 s ;                           --3 s 后输出电源信号
    clock < = clock_in after 6 s ;                      --6 s 后输出时钟信号
    reset < = '1' after 9 s ;                           --9 s 后输出复位信号
```

```
        reset < = '0' after 10 s;                    --复位信号回到高电平
        end process start_pro;                       --启动过程结束
    end;
```

本文所述的时钟对齐方法实现比较简单但并不惟一。复杂一点的方法可以采用不同时钟输出到不同单片机,比较反馈后,调整时钟输出个数达到调节目标。

参 考 文 献

1 何立民. 单片机应用系统设计. 北京 北京航空航天大学出版社,1990
2 边计年. VHDL 设计电子线路. 北京 清华大学出版社,2000
3 赵志敏. 实时双机容错系统的双机切换及同步控制. 计算机工程,1998

选自《电子技术应用》月刊,2002 年第 1 期

7.7 一种快速可靠的串行 flash 容错系统的设计与实现

广东工业大学 (510090) 李之彦 张立臣

一、引 言

工业控制节点一般由嵌入式处理器,如单片机/DSP 等来负责运算和控制。为减少连线,节省单片机/DSP 机的 I/O 引脚资源,一般用串行非易失性固态数据存储器/E^2PROM/flash (如 93CXX,24WCXX 等),来保存重要数据。这种典型的存储系统模型如图 7.7-1 所示。

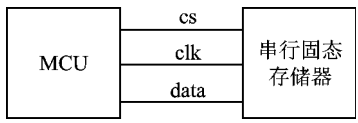


图 7.7-1 典型的串行固态存储器与 MCU 的连接图

图中的 CLK 为 MCU 收发串行数据的时钟,DATA 为串行数据线,CS 为片选信号。

这种串行存储系统的错误图样与固态并行存储系统[1],及 CPU—CPU,CPU—智能芯片的串行通信有较大的差异,这是因为串行 flash 的命令信息、地址信息、数据信息都是通过同一个信息包/帧,在同一根数据线上传送,而 flash 又不能发现或纠正错误的缘故。

一次串行 flash 的读写过程通常以信息帧为单位进行。信息帧的格式一般如图 7.7-2 所示。

命令字	地址信息	数据信息

图 7.7-2 信息帧的格式

在控制现场中,由于环境恶劣,串行存储系统中出现错误是难免的。由于串行信道上传送的有控制、地址、数据信息,而信道上的错误是随机的。故它们出错的概率正比于该类信息的流量。而控制信息在 CPU—串行存储器中的交互又以小批量 (即 1 个至数个字节) 居多,所以地址信息、命令信息出错的机率占很大的比例,从而形成了串行 flash 错误图样的特殊性。

二、串行 flash 的常见错误图样/特征

1. 错误出现在控制信息部分

串行 flash 的命令可分为随机/顺序读写,移动当前地址指针等基本类别[2]。当命令中的某些二进位出现错误时,这个错误可能导致原命令转移为命令集中的其他命令,如"读"命令变为"写"命令,或反之等。这样,1 个 bit 的命令错误,可能导致 n 个 bits 的错误,即错误扩张了。这种错误就很难用传统的纠错码,如无线寻呼的 BCH 码[3]、卷积码等,因为虽然香农定理证明理论上可纠正无限个错误,但在合理的运算里,特别是实时控制系统上,它们常用于纠正

一两位错误。因此,纠正这类错误要用其他可行的方法。

2. 地址信息部分出现的错误

与命令信息的错误一样,地址信息出现的错误也会将错误扩张。如果是读操作,若能发现错误,只需重读就可纠正错误,而不会对其他信息造成破坏 若为写操作,就会错误地改写了其他单元的数据,即本单元的错误转移到其他单元了。这种错误有更大的隐蔽性、欺骗性,因而更难检测及纠正。

3. 数据信息的错误

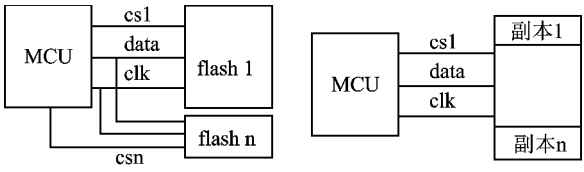
这类错误就是数据信息在串行传送过程中出现的错误,是传统的错误形式,常用的纠错方法均可适用。

4. Flash 内的物理错误

flash 的写操作寿命是有限的。因而会出现某些频繁写的单元由于操作次数接近了写生命周期而出现写错误,或由于电磁场冲击而造成永久损坏。这种情况与磁盘的某些扇区损坏的情况相似,也可用相似的方法标识这些字节是否可用。

三、实现串行 flash 容错的一种方法

由以上分析可知,串行 flash 的容错系统应能有效地纠正以上 4 类错误的任一类。现有绝大多数串行 flash 是没有智能的,它本身不能检测出 CPU 送来的数据是否有错,错误只能在下一次操作后才有可能检测出来。因此,纠错系统就显得极其重要。随着电子器件技术的发展,使得这种器件的容量越来越大,价格不断降低,从而使器件冗余和同一器件的存储空间冗余成为有效的容错措施,其系统构成如图 7.7－3 所示。



(a) 器件冗余结构 (b) 存储空间冗余结构

图 7.7－3 容错系统的结构

在器件冗余方式中,既可使用自定义的串行总线,也可用标准串行总线 (如 I^2C 等),这样不用大幅度增加处理器的 I/O 线。对每一单元的数据,在 n 个存储体上都有一个副本。为增加抗干扰能力,每个存储体使用不对称编码,如 1 号存储体用原码,2 号存储体用反码,3 号存储体用余 3 码等等。

存储空间冗余方式为在同一个存储体中划分出 n 个存储区域,每个区域可以用器件冗余的同样方法存放数据。

不论是器件冗余还是存储空间冗余,均为副本冗余。因此,只要有可行的查错误方法,就可保证有足够强的纠错能力。本系统使用二维奇偶监督码来实现查错。

四、快速纠错方法及其算法实现

二维奇偶监督码是将要检测的信息排成二进制矩阵,分别在水平和垂直方向的每一行列设置一个检验位。对 m 个信息单元 X1,X2,…,Xm,每个单元有 n 位二进制数据,即 Xi = dim

- 1. . di1 di0。其检验矩阵如图 7.7 - 4 所示。

d10	d11	…	d1i	…	d1n - 1	r00
d20	d21	…	d2i	…	d2n - 1	r01
.						
dj0	dj1	…	dji	…	djn - 1	r0j
.						
dm0	dm1	…	dmi	…	dmn - 1	r0
mr10	r11	…	r1i	…	r1n - 1	

图 7.7 - 4 纠错矩阵

其中：$r0i = \sum_{j=0}^{n} di + 1j \ MOD \ 2$；$r1i = \sum_{j=1}^{m} dji \ MOD \ 2$。

由于这种异或/半加运算是一个可交换、可分配群[4]，易于算法实现，且除了错误全在矩阵长方形顶点上外，其他所有错误均能查出，所以这是一种检验错误能力很强的查错方法。下面给出一个信息位为 8 ×8 以内的、用 51 系列单片机实现的二维奇偶监督码算法，可推广至信息位大于 8 ×8 的检验。

```
;产生检验码的程序
;入口：r0 为检验信息缓冲区首址，r1 为字节计数
;出口 检验结果存放在 r2，r3 中
LRC_CHECK：mov   r2，#0
           mov   r3，#0        ;清结果缓冲区
           mov   a，#0
Check1：   mov   a，r2         ;计算垂直奇偶监督位
           xrl   a，@r0        ;异或
           mov   r2，a         ;保存中间结果
           ;计算水平奇偶监督位
           mov   a，@r0
           mov   c，p
           mov   a，r3
           rlc   a
           mov   r3，a
           ;计算下一个字节
           inc   r0
           djnz  r1，check1
           ;结果由 r2，r3 返回调用程序
           ret
;检验二维奇偶监督码是否正确的程序
;入口：r0 为存储器中存放二维奇偶监督码的首址，r2，r3 为读出信息的二维
;奇偶监督码
;出口：进位 cy 为 0，检验无错；cy = 1，检验有错
verify：   mov   a，r2
```

```
        xrl     a,@r0
        jnz     error
        inc     r0
        mov     a,r3
        xrl     a,@r0
        jz      ok
error:  setb    c
        ret
ok:     clr     c
        ret
```

若检验出有错,可取另一无错的副本改正。执行这两个算法的时间在 200 个 CPU 时钟以内。

五、容错性能分析

本串行固态存储器的错误类型主要是白噪声,错误由命令、地址、数据及物理损坏等部分构成。由于现代处理器的性能一般较好,故单个错误出现的概率 p 很小(如 10E - 6),数据部分出现 3 个以上的错误的概率为 p^3,可以忽略。一般情况,物理损坏也易于发现。所以,本系统的容错性能主要体现在对出现 1 位错的命令、地址信息纠错上。由于使用副本容错,故只要能发现错误,就可用另一副本纠正错误。

命令、地址上的错误传递到固态存储器后,错误变为概率问题。因为命令、地址的 1 bit 错误,可能导致图 7.7 -4 存储矩阵的任意个可能错误。在图 7.7 -4 中,检测不到的错误只能出现在 4,8,12 等错误个数为 4 的倍数上。对于图 7.7 -4 的信息矩阵,在错误个数为 4 个时,检测不到的概率为:

$$p/(m\ n)\binom{2}{n}\binom{2}{n}\Big/\binom{4}{m\ \ n}$$

其中,p 为地址、命令中出现 1 个错的概率;1/(m n)为图 7.7 -4 的信息矩阵中,可能的错误个数为 m n 个,出现 4 个错的概率为其 1/(m n)。在出现 4 个错的图样中,只有错误处于矩形顶点上的图样未能检测到。这种矩形的个数为 $\binom{2}{n}\binom{2}{n}$ 个。这个样本的错误图样总个数为 $\binom{4}{m\ \ n}$。

对于 8 个错误的情况,只有错误恰好处于两个矩形的顶点时不能检测,其概率一般小于 4 个错误的一个数量级以上。对于 n = 8,m = 8 的情况,发现/纠正 4 位错的容错性能可在原始 1 位错的概率 p 上提高 3 个数量级以上。

六、结束语

本容错系统算法简明,运算速度快,容错能力可提高几个数量级,基本满足大多数场合的要求。加入本容错系统后,无论器件价格的开销,还是运算时间的开销均不大。经我们科研基金支持的仿真测试,及在汽车 GPS/GSM 防盗跟踪系统、住宅保安系统、模具控制系统的批量

产品应用中,都收到良好效果。

参 考 文 献

1 詹辛农等. 航天数据固态记录器设计问题. 遥测遥控,1999,1

2 CAT24WCXX User s Menu, Catalyst Semiconductor,inc,1998

3 许伟平. 无线寻呼系统的原理、设计与维护. 北京:电子工业出版社,1996

4 林舒等. 差错控制编码基础和应用. 北京:人民邮电出版社,1986

选自《微电子学与计算机》月刊,2002 年第 5 期

7.8 射频电路印刷电路板的电磁兼容性设计

东南大学 吴建辉

随着通信技术的发展,手持无线射频电路技术运用越来越广。如:无线寻呼机、手机、无线PDA等,其中的射频电路的性能指标直接影响整个产品的质量。这些掌上产品的一个最大特点就是小型化,而小型化意味着元器件的密度很大,这使得元器件(包括SMD、SMC、裸片等)的相互干扰十分突出。电磁干扰信号如果处理不当,可能造成整个电路系统的无法正常工作。因此,如何防止和抑制电磁干扰,提高电磁兼容性,就成为设计射频电路PCB时的一个非常重要的课题。同一电路,不同的PCB设计结构,其性能指标会相差很大。本文讨论采用Protel99 SE软件进行掌上产品的射频电路PCB设计时,如何最大限度地实现电路的性能指标,以达到电磁兼容要求。

一、板材的选择

印刷电路板的基材包括有机类与无机类两大类。基材中最重要的性能是介电常数 ε_r、耗散因子(或称介质损耗)tanδ、热膨胀系数CET和吸湿率。其中 ε_r 影响电路阻抗及信号传输速率。对于高频电路,介电常数公差是首要考虑的更关键因素,应选择介电常数公差小的基材。

二、PCB设计流程

由于Protel99 SE软件的使用与Protel 98等软件不同,因此,首先简要讨论采用Protel99 SE软件进行PCB设计的流程。

(1) 由于Protel99 SE采用的是工程(PROJECT)数据库模式管理,在Windows 98下是隐含的,所以应先建立1个数据库文件用于管理所设计的电路原理图与PCB版图。

(2) 原理图的设计。为了可以实现网络连接,在进行原理设计之前,所用到的元器件都必须在元器件库中存在,否则,就在SCHLIB中做出所需的元器件并存入库文件中。然后,只需从元器件库中调用所需的元器件,并根据所设计的电路图进行连接即可。

(3) 原理图设计完成后,可形成一个网络表以备进行PCB设计时使用。

(4) PCB的设计。① PCB外形及尺寸的确定。根据所设计的PCB在产品的位置、空间的大小、形状以及与其他部件的配合来确定PCB的外形与尺寸。在MECHANICAL LAYER层用PLACE TRACK命令画出PCB的外形。② 根据SMT的要求,在PCB上制作定位孔、视眼、参考点等。③ 元器件的制作。假如需要使用一些元器件库中不存在的特殊元器件,则在布局之前需先进行元器件的制作。在Protel99 SE中制作元器件的过程比较简单,选择"DESIGN"菜单中的"MAKE LIBRARY"命令后就进入了元器件制作窗口,再选择"TOOL"菜单中的"NEW COMPONENT"命令就可以进行元器件的设计。这时只需根据实际元器件的形状、大小等在TOP LAYER层以PLACE PAD等命令在一定的位置画出相应的焊盘并编辑成所需的焊盘(包括焊盘形状、大小、内径尺寸及角度等,另外还应标出焊盘相应的引脚名),然后以PLACE

TRACK 命令在 TOP OVERLAYER 层中画出元器件的最大外形,取一个元器件名存入元器件库中即可。④ 元器件制作完成后,进行布局及布线,这两部分在下面具体进行讨论。⑤ 以上过程完成后必须进行检查。这一方面包括电路原理的检查,另一方面还必须检查相互间的匹配及装配问题。电路原理的检查可以人工检查,也可以采用网络自动检查(原理图形成的网络与 PCB 形成的网络进行比较即可)。⑥ 检查无误后,对文件进行存档、输出。在 Protel99 SE 中必须使用"FILE"选项中的"EXPORT"命令,把文件存放到指定的路径与文件中("IMPORT"命令则是把某一文件调入到 Protel99 SE 中)。

注 在 Protel99 SE 中,"FILE"选项中的"SAVE COPY AS…"命令执行后,所选取的文件名在 Windows 98 中是不可见的,所以在资源管理器中是看不到该文件的。这与 Protel 98 中的"SAVE AS…"功能不完全一样。

三、元器件的布局

由于 SMT 一般采用红外炉热流焊来实现元器件的焊接,因而元器件的布局影响到焊点的质量,进而影响到产品的成品率。而对于射频电路 PCB 设计而言,电磁兼容性要求每个电路模块尽量不产生电磁辐射,并且具有一定的抗电磁干扰能力。因此,元器件的布局还直接影响到电路本身的干扰及抗干扰能力,这也直接关系到所设计电路的性能。因此,在进行射频电路 PCB 设计时除了要考虑普通 PCB 设计时的布局外,主要还须考虑如何减小射频电路中各部分之间的相互干扰、如何减小电路本身对其他电路的干扰以及电路本身的抗干扰能力。根据经验,对于射频电路效果的好坏不仅取决于射频电路板本身的性能指标,很大部分还取决于与 CPU 处理板间的相互影响。因此,在进行 PCB 设计时,合理布局显得尤为重要。

布局总原则:元器件应尽可能同一方向排列,通过选择 PCB 进入熔锡系统的方向来减少甚至避免焊接不良的现象;根据经验元器件间最少要有 0.5 mm 的间距才能满足元器件的熔锡要求,若 PCB 板的空间允许,元器件的间距应尽可能宽。对于双面板一般应设计一面为 SMD 及 SMC 元件,另一面则为分立元件。

布局中应注意:

⑴ 首先确定与其他 PCB 板或系统的接口元器件在 PCB 板上的位置,必须注意接口元器件间的配合问题(如元器件的方向等)。

⑵ 因为掌上用品的体积都很小,元器件间排列很紧凑,因此对于体积较大的元器件,必须优先考虑,确定出相应位置,并考虑相互间的配合问题。

⑶ 认真分析电路结构,对电路进行分块处理(如高频放大电路、混频电路及解调电路等),尽可能将强电信号和弱电信号分开,将数字信号电路和模拟信号电路分开,完成同一功能的电路应尽量安排在一定的范围之内,从而减小信号环路面积;各部分电路的滤波网络必须就近连接,这样不仅可以减小辐射,而且可以减少被干扰的几率,提高电路的抗干扰能力。

⑷ 根据单元电路在使用中对电磁兼容性敏感程度不同进行分组。对于电路中易受干扰部分的元器件在布局时还应尽量避开干扰源(比如来自数据处理板上 CPU 的干扰等)。

四、布 线

在基本完成元器件的布局后,就可开始布线了。布线的基本原则为:在组装密度许可情况下,尽量选用低密度布线设计,并且信号走线尽量粗细一致,有利于阻抗匹配。

对于射频电路,信号线的走向、宽度、线间距的不合理设计,可能造成信号传输线之间的交叉干扰;另外,系统电源自身还存在噪声干扰。所以,在设计射频电路 PCB 时一定要综合考虑,合理布线。

布线时,所有走线应远离 PCB 板的边框 (2 mm 左右),以免 PCB 板制作时造成断线或有断线的隐患。电源线要尽可能宽,以减少环路电阻,同时,使电源线、地线的走向和数据传递的方向一致,以提高抗干扰能力;所布信号线应尽可能短,并尽量减少过孔数目;各元器件间的连线越短越好,以减少分布参数和相互间的电磁干扰;对于不相容的信号线应尽量相互远离,而且尽量避免平行走线,而在正反两面的信号线应相互垂直;布线时在需要拐角的地方应以 135°角为宜,避免拐直角。

布线时与焊盘直接相连的线条不宜太宽,走线应尽量离开不相连的元器件,以免短路;过孔不宜画在元器件上,且应尽量远离不相连的元器件,以免在生产中出现虚焊、连焊、短路等现象。

在射频电路 PCB 设计中,电源线和地线的正确布线显得尤其重要,合理的设计是克服电磁干扰的最重要的手段。PCB 上相当多的干扰源是通过电源和地线产生的,其中地线引起的噪声干扰最大。

地线容易形成电磁干扰的主要原因在于地线存在阻抗。当有电流流过地线时,就会在地线上产生电压,从而产生地线环路电流,形成地线的环路干扰。当多个电路共用一段地线时,就会形成公共阻抗耦合,从而产生所谓的地线噪声。因此,在对射频电路 PCB 的地线进行布线时应该做到以下几点。

⑴ 首先,对电路进行分块处理,射频电路基本上可分成高频放大、混频、解调、本振等部分,要为各个电路模块提供一个公共电位参考点即各模块电路各自的地线,这样信号就可以在不同的电路模块之间传输。然后,汇总于射频电路 PCB 接入地线的地方,即汇总于总地线。由于只存在一个参考点,因此没有公共阻抗耦合存在,从而也就没有相互干扰问题。

⑵ 数字区与模拟区尽可能以地线进行隔离,并且数字地与模拟地要分离,最后接于电源地。

⑶ 在各部分电路内部的地线也要注意单点接地原则,尽量减小信号环路面积,并与相应的滤波电路的地线就近相接。

⑷ 在空间允许的情况下,各模块之间最好能以地线进行隔离,防止相互之间的信号耦合效应。

综上所述,本射频电路 PCB 设计的关键在于如何减少辐射能力以及如何提高抗干扰能力,合理的布局与布线是设计射频电路 PCB 的保证。文中所述方法有利于提高射频电路 PCB 设计的可靠性,解决好电磁干扰问题,进而达到电磁兼容的目的。

参 考 文 献

1 陈穷. 电磁兼容性工作设计手册. 北京:国防工业出版社,1993

2 马伟明. 电力电子系统中的电磁兼容. 武汉:武汉水利电力大学出版社,2000

选自《单片机与嵌入式系统应用》月刊,2002 年第 8 期

7.9 去耦电容在 PCB 板设计中的应用

武汉大学电气工程学院 (430072)
徐 亮 阮江军 甘 艳 莫付江 文 武

随着电子技术的发展,电子设备在各个方面的应用越来越普及,对于电子设备的电磁兼容性要求也越来越高,对电子设备的抗干扰能力也有了严格要求,而 PCB 板设计在电子设备的电磁兼容性及抗能力方面有着重要的作用。本文主要讨论的是在 PCB 板设计中如何合理地使用去耦电容的问题。

一、去耦电容的作用

PCB 上供电线路的寄生阻抗有可能产生下述三种电磁干扰,所以应采用去耦电容 (见图 7.9-1) 来防止和减轻这些干扰的影响。

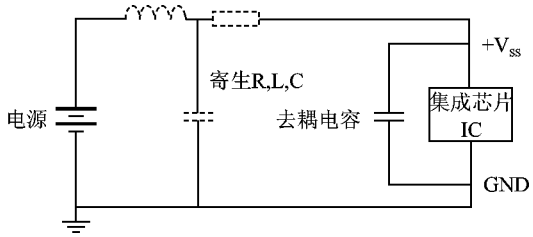


图 7.9-1 加在电源端和接地端之间的去耦电容

(1) 门电路开关瞬间电流是跳跃式变化的,由于集成片通过电源线与电源相连接,电源线的电感将会阻止电流的瞬态变化,从而影响集成片的响应速度。

(2) 集成片的瞬态变化电流流过环路面积较大的电源线路时,将会产生较为强烈的对外辐射噪声。且由于各集成片很可能会流经相同的线路,相互之间存在较大的公共阻抗,从而产生了较严重的共阻抗耦合干扰。

(3) PCB 板的电源线存在寄生电阻、电容、电感,线路电感的反电动势又使集成片得到的电源电压高于额定值。所以当集成片电源端子上电压振荡的幅值超过数字逻辑元件的噪声容限时就会产生干扰。

去耦电容的充放电作用使集成片得到的供电电压比较平稳,减小了电压振荡现象;集成片可以就近在各自的去耦电容上吸收或释放电流,不必通过电源线从较远的电源中取得电流,因此不会影响集成片的速度;同时去耦电容为集成片的瞬态变化电流提供了各自就近的高强通道,从而大大减小了向外的辐射噪声且相互之间没有公共阻抗,因此抑制了共阻抗耦合。

由于去耦电容在高频时的阻抗将会减小到其自谐振频率 (具体计算方法如下面内容所述),因而可以有效地去除信号线中的高频噪声,同时相对低频来说对能量没有影响,所以可在每一个集成片的电源/地脚之间加一个大小合适的去耦电容。在选择去耦电容类型时,应考

虑那些低电感的高频电容,例如高频性能较好的多层陶瓷电容或独石电容。

二、去耦电容的容量

从上述三种作用出发可得到去耦电容容量必须满足下列条件。

⑴ 集成片与去耦电容两端电压差 ΔV_0 必须小于噪声容限 V_{NI}

$$\Delta V_0 = \frac{L\Delta I}{\Delta t} \leqslant V_{NI} \tag{1}$$

式中 ΔI——门电路开启时所需暂态电流幅值;

Δt——门电路开启所需时间,一般为脉冲上升时间;

L——去耦电容的电感(包括引线电感以及去耦环路电感)。

⑵ 从去耦电容为集成片提供所需的电流的角度考虑,其容量应满足

$$C \geqslant \Delta I\Delta t/\Delta V \tag{2}$$

理想的去耦电容应能提供逻辑设备所需的暂态电流。当集成片门电路由关闭状态翻转成开启状态时,将在 Δt 内从去耦电容中吸收大量的暂态电流 ΔI,该电流不仅包括门电路开启所需的电流,还包括驱动下一级门电路负载的扇出电流。由于去耦电容在 Δt 内仅提供能量,还来不及从电源中获得补充能量,所以电容上的电压将下降 ΔV,该电压跌落应被控制在规定的范围内;由于电容器上的电压就是集成片上的电压,所以电压的跌落应该不引起集成片门电路的错误逻辑动作,一般取 ΔV 为 $20\% V_{NI}$。

⑶ 集成片开关电流 i_c 的放电速度必须小于去耦电容电流的最大放电速度[6]

$$\frac{di_c}{dt} \leqslant \frac{\Delta V}{L} \tag{3}$$

⑷ 去耦电容的自谐振频率 f_0 必须大于集成片的最高应考虑的谐波频率 f_{max}

$$f_0 \geqslant f_{max} \tag{4}$$

从直观上看似乎电容值越大提供电流的能力越强,因此许多人喜欢使用容量大的去耦电容,实际上这是一个错误的观念。因为实际使用的电容器总存在一定的引线电感,这些电感与电容将产生串联谐振。

在谐振频率点 f_0 处阻抗最小,其为高频电流所提供的通道阻抗最小,所以去耦效果最佳。所以谐振频率将是使用去耦电容时应首要考虑的问题。计算引线电感的公式为[1]

$$L = \frac{\mu_0}{2\pi} l\left\{\ln\left[\frac{l}{r} + \sqrt{\left(\frac{l}{r}\right)^2 + 1}\right] + \frac{r}{l} - \sqrt{\left(\frac{r}{l}\right)^2 + 1}\right\} \tag{5}$$

式中 l——导线的长度;

r——导线的半径。

一般实际使用电容的引线电感 L 约为每英寸 15 nH,同时每个过孔约有 1 到 3 nH 的电感,由

$$f_0 = \frac{1}{2\pi\sqrt{LC}} \tag{6}$$

即得理论上的谐振频率。具体使用的去耦电容的自谐振频率随封装形式以及引脚长度的不同而有所变化,应进行具体的测量。引线为 0.25 英寸的去耦电容的自谐振频率可参见文献[7]。

应注意到由于表面安装式电容器有较小封装形式,所以其引线电感较低且没有径向引线或轴向引线电感,故其自谐振频率比普通形式的电容器高 10 倍。

在进行去耦电容器容量选择时,一旦工作频率超过其谐振频率时,串联电路呈感性,阻抗随频率升高而增加。由于去耦支路作为高频去耦通道的条件是

去耦支路阻抗≪电源线阻抗

所以去耦支路的去耦效果将随频率升高而下降,最后甚至不起作用。此时应根据集成片的最高谐波频率 f_{max} 来选择去耦电容的自谐振频率 f_0。频率取值的最佳条件为 $f_0 = f_{max}$ (即在高频时其通道阻抗最小),其中

$$f_{max} = \frac{1}{\pi t_c} \tag{7}$$

式中 t_c ——门电路脉冲上升时间[6]。

由式 (6) 及式 (7) 可得去耦电容的最大值为

$$C_{max} = \frac{t_c^2}{4L} \tag{8}$$

上述公式仅是定性的分析,实践中常常采用试验的方法来确定最佳电容值。根据试验,对 14 和 16 脚的芯片,470 ~ 1 000 pF 的电容具有最好的效果。

实际电路中当集成片同时有多个逻辑门状态发生变化时,电容值要按逻辑门的数量扩大。例如为确保在刷新过程中正确的运作,内存需要大容量的去耦电容以提供更大的电流,例如 256 K 的 RAM 需要 0.1 μF 的电容。对插脚多的超大规模集成电路元件也是如此。同时对高密插脚方格布置模型需更多的去耦电容,特别是比较大的容性负荷下信号、地址和控制端同时变化时更是如此。

三、多层印制电路板的去耦电容

在设计 PCB 板时采用多层 PCB 板,其中的一个好处是可以将电源层和地层相邻布置。多层 PCB 板的物理关系产生了一个较大的去耦电容,通常这个电容为低频提供了足够的去耦能力。应注意该去耦电容只有在层间相距很近 (0.01 英寸,高频时最好是 0.005 英寸) 时才能达到最佳效果。如果门电路脉冲上升时间 t_r 小于 10 ns (例如标准的 TTL 系列逻辑芯片),就不需要再加去耦电容。但还需要储能电容来保证正常运行时所需电平,例如在电源进线两端接 0.1 μF 的储能电容。

使用电源层和地层作为主要去耦电容仍要考虑层间电容的自谐振频率。如果电源层与地层的自谐振频率与 PCB 板上总的去耦电容的自谐振频率相等,则遇到该频率时就会发生剧烈的谐振,不再有宽频的去耦能力。如果时钟频率与该谐振频率相等,高频时将没有去耦能力。发生这种情况时,PCB 板会变成辐射源且有可能达不到控制电磁干扰的要求。这时可以采用外加具有不同的自谐振频率的去耦电容以改变 PCB 板层间的谐振频率。

改变电源层和地层自谐振频率的一个方法是改变其层间距离。增加或减小隔离高度或重新布置层面将改变电容值,但缺点是随之也改变了信号轨线层的阻抗。通常情况下多层电路板电源层和地层的自谐振频率在 200 ~ 400 MHz 之间。可以考虑应用 20 - H 规则使自谐振频率增到 2 ~ 3 倍。下面将给出计算多层板自谐振频率的公式。

多层印制电路板中芯板厚度、介电常数以及印制电路板中电源层的位置都会影响其去耦

电容的大小。用于计算 PCB 板去耦电容谐振频率 f_0 (MHz) 的两个公式如下

$$f_0 = \frac{1}{2\pi\sqrt{LC}} \tag{9}$$

$$f_0 = \frac{c}{2l\sqrt{\varepsilon_{eff}}} \tag{10}$$

式中 L——引线电感 (nH)；

C——平板电容器的电容 (pF)；

l——内部连接长度；

c——真空中的光速；

ε_{eff}——媒质中的介电常数。

通过谐振频率可求出去耦电容的容量 C[2]

$$C = \varepsilon_0\varepsilon_r\frac{(N-1)A}{t} \tag{11}$$

式中 ε_0——真空的介电参数；

ε_r——极板间介质的相对介电常数，典型值为 4.5；

N、A、t——平板的数量、面积及两极之间的厚度。

上面的公式仅适用于所有安装有去耦电容的印刷线路板具有完全相同的区域。但在某些实际设计中由于在电路板上存在着用于内部连接的过孔，使其不能成为一个整体，设每个分离区域的面积分别为 A_1、A_2、…，相应的厚度为 t_1、t_2、…。这时应使用下面的公式[3]求其容量

$$C_{total} = \varepsilon_0\varepsilon_r\left[\frac{A_1}{t_1} + \frac{A_2}{t_2} + \frac{A_3}{t_3} + A\right] \tag{12}$$

四、增强去耦效果的方法

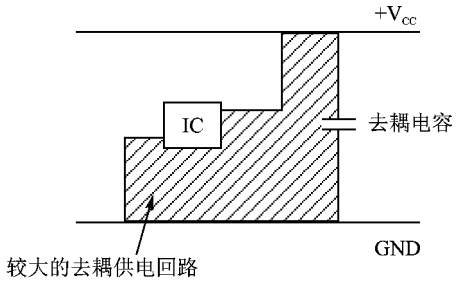
增强去耦电容的想法就是要尽量阻止一个电路对另一个电路的影响。所以合理使用去耦电容就成为增强去耦电容效果的重要手段。

去耦电容的作用是减小耦合干扰和为芯片提供瞬态高能量。应尽量降低去耦电容器自身的引线电感，所以单个去耦电容或并联去耦电容的引线越短，去耦电容的工作状况越好。由于安装电容器时其引线长度 (应注意引线长度还包括将电容器连接至层面的通道长度) 是固定的，因而实际中减小引线电感是不可能的。由于去耦电容安装到实际印制电路板上时还存在去耦环路电感，去耦环路电感同样会影响去耦效果，所以一般认为在布线时要尽量使去耦电容靠近芯片。但这种提法有时不够确切，更确切的要求是使去耦电容的供电回路面积尽量小。也就是说使去耦电容与芯片电源端和地端间的联线尽量短。

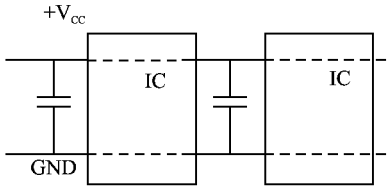
所以应将图 7.9－2 (a) 中去耦电容的安装方法改为如图 7.9－2 (b) 所示。

由于电源插头位于元件中间可以提供最优的去耦电容布置，所以应尽量选择电源和地端位于设备中间而不是在对角线上。这种布局不但减小了设备和去耦电容连接线的长度，还减小了封装体内电源和地之间的引线电感。

为了抑制元件在开关电路同时翻转时产生的频带较宽的暂态电流，可将两个电容器并联安装在电源插脚上以优化工作状况。但是并联电容器的数值必须相差两个数量级 (如 0.1 μF



(a) 改进前的安装



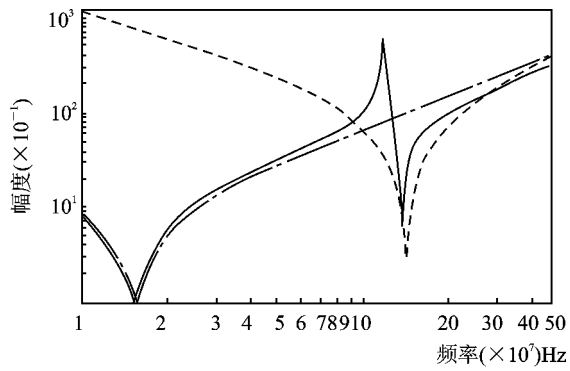
(b) 改进后的安装

图 7.9－2 去耦电容安装方法的改进

和 0.001 μF)。并联电容器的总电容并不重要,并联电容的并联电感值才是重要的问题(由于自谐振频率)。这是由于增加一个较小容量的电容后,其总容量并没有较大改变,但是两组引线将比一组引线拥有更宽的信号线,所以引线电感将会有所减小。该方法还可用于改变电源层和地层的自谐振频率。

根据对多重去耦电容的研究表明[4],并联去耦电容并不十分有效。在高频区域并联去耦电容仅比单个大电容有 6 dB 的改善。尽管 6 dB 对于暂态电流的抑制来说只是一个小数目,但这 6 dB 也许可以使一个不符合内部电磁干扰规定的产品变为符合规定的产品。

图 7.9－3 显示了电容值为 0.01 μF 和 100 pF 的电容分别安装以及并联安装时的响应[4]。



—— 0.01 μF与100 pF并联; – – – 100 pF陶瓷电容; —·— 0.01 μF陶瓷电容

图 7.9－3 并联去耦电容的响应

根据去耦电容的工作原理,如果增加芯片从电源吸收能量的难度,就能使芯片尽量从去耦电容吸收能量,从而充分发挥去耦电容的作用,减小电源线上的噪声(di/dt)。根据这个思路,

可以人为地增加去耦电容电源一侧电源线的阻抗。所以在布线时,使电源侧的布线尽量细(但要满足供电的要求),增加布线的电感,相当于增加了阻抗,可以起到一定的效果。

也可如图 7.9-4,在去耦电容的电源侧安装一只铁氧体磁珠,由于磁珠对高频电流呈现较大的阻抗,因此增强了电源去耦电容的效果。该方法不仅能在芯片级的去耦电容上应用,在二级去耦电容和线路板电源入口处也可使用,减小了较长电源线上的电流波动,从而降低辐射程度。

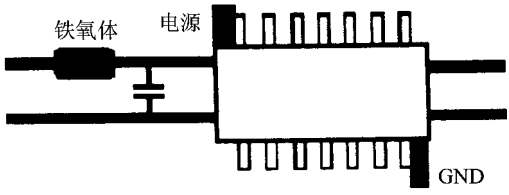


图 7.9-4 在去耦电容电源侧加装铁氧体磁珠

但是必须将铁氧体安装在靠近电源处而不能安装在靠近芯片的一端。靠近电源处相当于增加了芯片从电源线吸取电流的难度,使其尽量使用去耦电容中的电流。如果将铁氧体安装在芯片侧,则相当于增加了电容放电回路的电感,会起到相反的效果。应注意铁氧体在直流电流作用下磁导率会下降,甚至由于磁饱和而完全消失。另外铁氧体的实际电感量很小,只对高频电流其阻抗较大,所以铁氧体主要在高频中发挥作用。

另一种减小由于共阻抗耦合带来的地电位波动的传统方法是加一个与电源层/地层并联的有源滤波器。但是文献[5]中提出,使用有源滤波器将消耗大量的工作电流,整个电路的驱动能力受到影响,文献中同时提出了替代方法。图 7.9-5 显示出其用于降低电源以及地点位

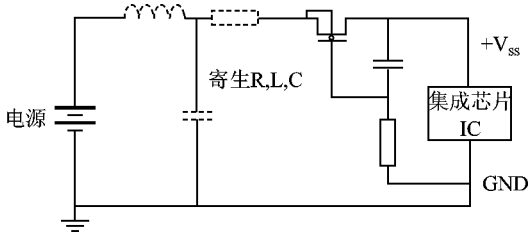


图 7.9-5 加装 PMOS 管的去耦电路

波动的电路。当电源线电压急剧增加或地线电压急剧下降时,电容将会提供电流给电阻,这将会使 PMOS 门级电压升高。同时 PMOS 的栅级以及源级电阻增加,使通过寄生电感的电流减少并且将会有更多的电流流过电容。这样,电容将会比传统去耦电容更有效。

参 考 文 献

1 C. R. Paul. Introduction to electromagnetic compatibility New York [Z], Wiley, 1992

2 E. Bogatin. Design rules for microstrip capacitance [Z]. IEEE Trans. Comp., Hybrids, Manufact. Tecnol., vol. 11, no. 3, p. 253, Sept. 1990

3 E. Bogatin. A closed from analytical model for the electrical properties of microstrip interconnects [Z]. IEEE Trans. Comp., Hybrids, Manufact. Tecnol., vol. 13, no. 2, p. 259, June 1990

4 M. Goetz. Time and frequency domain analysis of integral decoupling capacitors [Z]. IEEE Trans. Comp., Hybrids, Manufact. Tecnol., vol. 19, no. 3, p. 518, Augu. 1996

5 C. R. Paul. Effectiveness of multiple decoupling capacitors, IEEE Trans. Comp [Z], EMC-34, p130, 1992

6 Y. Yun, H. Yoo, S. Ham, Y. Kim, Y. Lee. A filter for low EMI and low noise [Z]. IEEE, AP-ASIC'99

7 V. Golumbeanu, P. Svasta, D. Leonescu. The decoupling efficiency of power for low voltage logic circuits [Z]. IEEE, Elec. Tech-Med'98, vol. 1, p. 120, 1998

8 M. Montrose. Printed circuit board design techniques for EMC compliance [Z]. 1996

9 沙斐. 机电一体化系统的电磁兼容技术 [M]. 北京 中国电力出版社, 1999

选自《电测与仪表》月刊, 2002 年第 4 期

7.10 密码访问器件 X76F100 在单片机系统中的应用

株洲硬质合金厂 何丽平

X76F100 是一种密码访问安全监控器件,内部含有 1 个 112 ×8 位的保密数据阵列,对该阵列的访问由 2 个 64 位的读写密码来控制,密码与数据通过 I^2C 总线接口完成输入输出。正常情况下,X76F100 提供最少为 10 万次的擦写期限和最少 100 年的数据保存期。

一、器件的特点

器件的特点如下:

⑴ 可编程 64 位读写密码保护;
⑵ 重试计数寄存器允许 8 次密码试验,然后阵列清零;
⑶ 32 位对复位的响应 (RST 输入);
⑷ 8 字节页写方式;
⑸ 最大 1 MHz 时钟速率;
⑹ I^2C 总线接口;
⑺ 宽电压 (3.0 ~5.5 V) 低功耗 CMOS;
⑻ 10 万次擦写和 100 年数据保存;
⑼ 多种 (8 脚 PDIP、SOIC、MSOP、智能卡) 外形封装。

二、封装与引脚说明

图 7.10 -1 是 X76F100 的各种封装形式。

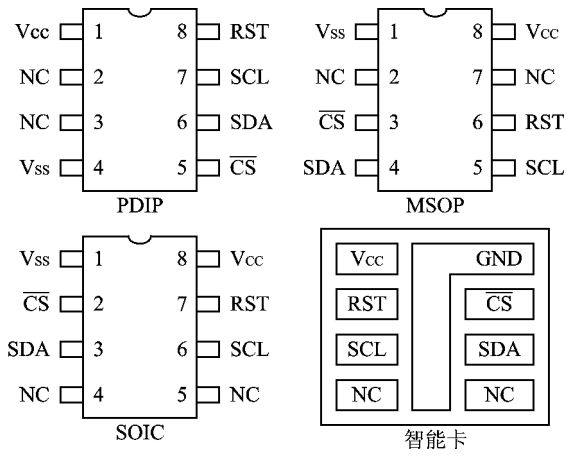


图 7.10 -1 外型封装图

引脚功能如下：

① 串行时钟 SCL。串行时钟输入端用来控制所有的数据输入和输出器件。

② 串行数据 SDA。SDA 是一个漏极开路的串行数据输入/输出引脚。

③ 片选$\overline{CS}$。$\overline{CS}$为低电平时，X76F100 处于工作方式；否则，处于等待方式。

④ 复位 RST。RST 是器件的复位脚。当 RST 被触发为高电平而$\overline{CS}$为低电平时，X76F100 将输出标准的 32 位“复位同步响应”数据。

⑤ 电源 V_{CC}，地 V_{SS}。在器件上应施加 3.0～5.5 V 的工作电压。

三、器件的读写时序

1. I^2C 总线协议

X76F100 支持 I^2C 二线制总线协议。数据发送以字节为单位，高位在前，低位在后，且在所有的应用中 X76F100 都被作为从机。总线协议约定如下：

开始条件——当 SCL 为高电平时，SDA 由高电平到低电平的跳变。

停止条件——当 SCL 为高电平时，SDA 由低电平到高电平的跳变。

数据改变——SDA 线上数据的状态只有在 SCL 为低电平时才能改变。

应答 ACK——用来表示数据传送成功的软件约定。发送器件在发送 1 个字节的 8 位以后，将释放总线，在额外的第九个时钟周期，接收器将 SDA 线拉至低电平，以应答它接收到了 8 位数据。此时 SDA 的高电平被认为是一个 NO－ACK（数据无效）。I^2C 的通信时序等详细技术规范可参看有关手册。

2. 命令代码

如表 7.10－1 所列，X76F100 一共有 5 种操作，分别由不同的命令代码来选择。器件操作时，命令代码必须跟随在起始信号后发出。

表 7.10－1

命令代码	功　能
100S3S2S1S00	页写
100S3S2S1S01	页读
1111110	改变写密码
11111110	改变读密码
010101	密码问询

对其他的非法命令代码器件将用 NO－ACK 作为响应，然后回到等待方式。页面读与写命令中的 S3S2S1S0 为页面地址，代表 14 个 8 字节的页中的一个。对阵列的读或写总是从页的第一个地址开始。读操作可以不限定地继续，而写操作一次必须是全部 8 个字节。

3. 页写与页读

页写或页读时，应让$\overline{CS}$和 RST 均保持低电平，其工作流程如图 7.10－2 所示。写命令或读命令控制字节中包括所要写入的页地址，随后紧跟 8 字节的写操作或读操作密码。密码是否正确应通过一个密码问询（起始信号＋01010101）来判断，ACK 回答表示密码正确，NO-ACK 回答则表示刚输入的密码有误。X76F100 内部包含一个重试计数器，对任何不正确的密码重

试计数器都将计数加 1。当计数到 8 次溢出时,器件的存储器区和两个密码会自动清除为 0(就像刚出厂一样)。如果在重试计数器溢出以前收到一个正确的密码,则重试计数器复位并允许访问器件。

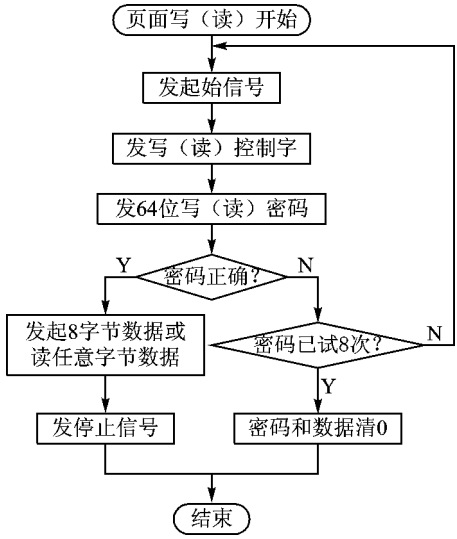


图 7.10-2 页写(读)流程图

密码正确时,写操作要求传送 8 个字节的数据,在最后字节传送以后发出一个停止条件。这个条件启动非易失性写周期,数据从页的第一个地址开始写入。如果传送少于或多于 8 个字节,则写周期不会被启动,页中的数据将保持不变。在数据写入时(约 10 ms),器件对新的命令将不予应答(回答 NO-ACK)。

如果是读操作则必须从页的第一个地址开始,但一次读出的字节数可任意。读到最后一页时,器件将自动转回到第一页继续。在读完最后一个字节后,主机可不作 ACK 确认而直接发出结束信号,至此,此次读操作完成。

四、密码的修改

器件从工厂中运出时所有的密码都等于"0",使用前应对其作必要的修改,另外,用户也有可能需要更改已知的密码。它们可通过在正常的页面写操作过程中送一个"改变读密码"或"改变写密码"命令来实现。

图 7.10-3 是修改密码流程图。送出"改变读密码"或"改变写密码"命令后,应将现在已知的密码(如出厂时的全 0)发出,然后进行密码问询。得到一个有效的密码应答响应 ACK 后,再传送一个全部 8 字节的新的密码即可。用户可在 2 ms 内,用一个数据应答轮询命令来判断新密码的写操作是否已经启动,一个 ACK 表明写操作未能成功启动,而 NO-ACK 应答则表明密码正在写入;另外,用户也可用一个重复的密码应答轮询命令来检查新的密码是否已被正确地写入,一个 ACK 应答(通常 10 ms 后)表示新的密码已有效。

X76F100 中的密码可以修改,但无论如何都无法将其读出。

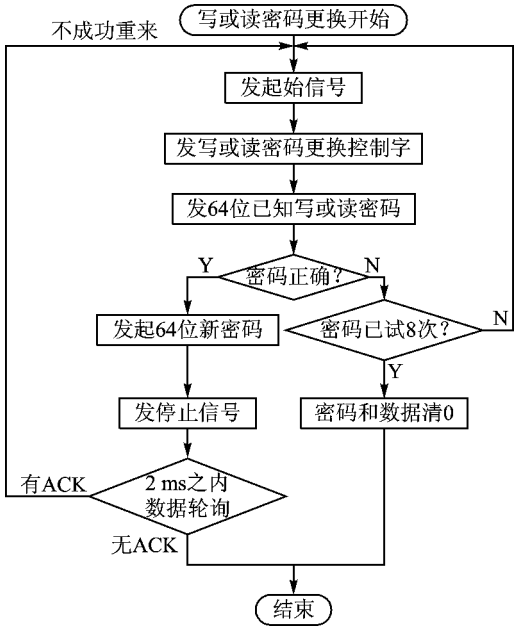


图 7.10－3　更换写或读密码流程图

五、应用编程举例

只要符合 X76F100 的数据通信规范，微控制器与其接口就不会有问题。下面给出的是 AT89C51 对 X76F100 进行写操作的例程。读操作或修改密码的程序可照此编写。稍加修改，该程序也可移植到其他微控制器与 X76F100 的接口通信中。

```
;将 40H 单元开始的 8 个数据写入 X76F100 的第二页。写
;密码在 70H 开始的 8 个单元中
;
;***AT89C51 与 X76F100 接口***
SCL     EQU     P3.2
SDA     EQU     P3.3
CS      EQU     P3.4
;***起始条件与结束条件***
START:  SETB    SDA             起始条件
        SETB    SCL
        CLR     SDA
        CLR     SCL
        RET
STOP:   CLR     SCL             结束条件
        CLR     SDA
        SETB    SCL
        SETB    SDA
        RET
```

```
;***发送一个字节到 X76F100 中,发送数据在 A 中,返回的应答在进位中***
WBYTE: MOV     R7,#8
WBYTE1:RLC     A
       MOV     SDA,C          ;发送一位
       SETB    SCL
       CLR     SCL
       DJNZ    R7,WBYTE1
       SETB    SCL
       MOV     C,SDA          ;读应答 ACK 信号
       CLR     SCL
       RET
;***主程序***
MAIN:  CLR     CS
       MOV     R5,#8          ;置密码重试次数
MAIN0: LCALL   START          ;发起始条件
       MOV     A,#100001000B  ;写控制字
       LCALL   WBYTE          ;发送写控制字
       JNC     MAIN0          ;NO-ACK,重来
       MOV     R6,#8
       MOV     R0,#70H        ;8 字节密码首址
MAIN1: MOV     A,@R0
       LCALL   WBYTE          ;发送 8 位密码
       JNC     MAIN0
       INC     R0
       DJNZ    R6,MAIN1
       MOV     A,#55H
       LCALL   WTYBE          ;密码问询
       JNC     ERROR          ;密码有误,转错误处理
       MOV     R6,#8          ;密码正确
       MOV     R0,#40H        ;发送 8 字节数据
MAIN2: MOV     A,@R0
       LCALL   WBYTE
       JNC     MAIN0
       INC     R0
       DJNZ    R6,MAIN2
       LCALL   STOP           ;发停止条件,启动写操作
       SETB    CS
MAIN3: SJMP    MAIN3          ;其他程序省略
ERROR: DJNZ    R5,MAIN0
ERROR1:SJMP    ERROR1         ;出错处理程序省略
```

六、结束语

X76F100 主要应用在电子货币、身份识别、考勤管理等方面。在 IC 卡的应用过程中，电路及编程设计都要考虑因插卡而造成的电路短路、接触抖动等干扰因素。上述编程实例没有考虑 X76F100 因故障造成 NO - ACK，程序进入死循环的问题。实际应用编程时，应注意解决。

参 考 文 献

1 窦振中. PIC 系列单片机原理和程序设计. 北京 北京航空航天大学出版社，1998
2 陈伟人. 单片微机计算机原理及其应用. 北京 清华大学出版社，1989
3 WWW. XICOR. COM

选自《单片机与嵌入式系统应用》月刊，2002 年第 3 期

7.11 计算机的电磁干扰研究

西安石油学院计算机系 (710065) 李海泉

随着科学技术的发展,计算机和各种电子设备、广播、电视、仪器及军用雷达等无线电设备的广泛应用,各种电磁波在地球空间中长期共存。电子计算机要在这样浩瀚的电磁干扰环境中工作,不仅其可靠性、稳定性、安全性要受到严重影响,而且还会干扰其他设备,因此迫切需要解决计算机的电磁兼容性问题。

所谓"电磁兼容性"是指电子计算机在电磁环境中的适应性,即能保持其固有性能,完成规定功能的能力。它要求计算机与同一时空环境的其他电子设备相容兼顾,既不受电磁干扰的影响,也不会对其他电子设备产生影响,因此需要进行计算机的电磁兼容性的研究。

一、来自计算机中的电磁干扰

计算机的电磁干扰,包括计算机本身产生影响的电磁干扰和来自计算机系统外部的电磁干扰。

计算机内部的电磁干扰,主要来自以下几个方面:① 元器件噪声干扰;② 寄生耦合干扰;③ 信号反射干扰;④ 高频电路辐射干扰;⑤ 地线干扰。

1. 元器件噪声干扰

计算机中的元器件不都是理想的,即使是合格的元器件经过长期使用后,性能也会衰变。它们的特征参数往往会偏离理论值,这种误差会影响元器件的噪声特性。在实际应用中,无论是电阻、电容、电感等无源元件,还是 TTL、MOS 器件等有源器件均不同程序地存在噪声干扰。

2. 寄生耦合干扰

在计算机内,电路板上的每个元器件、每根导线中均流过一定大小的电流,都具有一定的电位,因此其周围形成一定大小的电磁场。当计算机电路中的元器件和线路布局不合理,地线、屏蔽线接法不当,电路间耦合不良时,就会在导线间产生分布电容或电感,寄生信号便通过它们耦合进计算机,使信号畸变出错。

3. 信号干扰反射

计算机电路中,如果信号线阻抗与负载阻抗不完全匹配,脉冲信号就会在传输线中产生反射现象,使信号波形产生瞬间冲击,造成电路逻辑故障。插件印制板金属化孔通导不良,印制线粗细不均匀,都会产生信号反射干扰。

4. 地线干扰

在计算机系统中,如果接地不妥,交直流地线混合连接或者接地线自成闭合回路,或者一根接地导线两端在不同的地点接地,当负载不平衡或因其他因素影响,使导线两端电位不等于 0,就会在大地、导体和电源之间及设备电路之间形成"环路",通称"地环路",如图 7.11-1 所示。这个地环路会引起很大的电流,不仅会产生地线干扰,使计算机信号出错,还会损坏计算机元器件。

5. 高频电路辐射干扰

计算机中的高频电路，不仅会产生有用时序信号，还会产生辐射干扰，其能量集中在它的谐波和杂乱辐射上。对计算机本身所产生的电磁干扰，需要在电路设计、系统设计、元器件选择、安装布线时考虑，并采取必要措施加以解决。

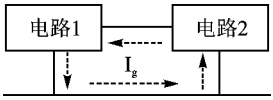


图 7.11－1　地环路干扰

二、来自外部的电磁干扰

计算机外部的电磁干扰，主要是指电气设备的干扰、自然干扰和静电干扰。如下所示：

- 来自外部的干扰
  - 电器设备干扰
    - 工频干扰-电源干扰
    - 开关干扰
    - 放电干扰
      - 火花放电
      - 辉光放电
      - 弧光放电
    - 射频干扰
  - 自然干扰
    - 雷电干扰
    - 射电干扰
    - 辐射干扰
    - 电磁脉冲
  - 静电干扰

1. 电气设备干扰

工业电气设备产生的干扰，是计算机电磁干扰的主要来源。这种干扰源按其干扰性质可分为工频干扰、开关干扰、放电干扰和射频干扰。

⑴ 工频干扰：工业频率的电流互感器、整流器、高压输电线、交流稳压器中的交流电，不仅会产生交流噪声，还会因所含的高频谐波分量产生噪音干扰。电源中的高频瞬间电压、浪涌电压、谐波畸变和大电流冲击，可通称为“电源噪声”。当市电电压忽高忽低或者频繁开关机器或将一个大负载接入到一个系统都会在电源线及其供电设备上产生一个电源噪声。当市电电压经常在 170～250 V 之间波动，特别是上、下班时间电网负荷变动很大，而且是突发性过度过程，会产生很强的干扰脉冲。实践中观察这种脉冲间隔为 10～100 ms、10～100 kHz、600～700 V。一次开、关电传机竟在电源线上产生 1 000 V 的尖峰脉冲，使一些元器件损坏。另外，来自别的噪声源的噪音，也可以沿着电源线进入计算机。据统计，尖峰脉冲干扰平均每月发生 50.7 次，大电流冲击每月发生 10 多次，严重影响了计算机设备的安全性、系统的稳定性及可靠性。

⑵ 开关冲击：工业电气中的大功率开关、继电器、点焊机、交流整流器开关的通断，都会使电流发生急剧变化，产生脉冲式干扰。另外，机房内的其他用途的设备，如日光灯、吸尘器、手电钻等也都会产生瞬间的干扰。这些冲击干扰的电流，不仅含有丰富的高次谐波，容易产生感应电磁场，而且干扰电平很大，会损坏计算机器件和造成计算机信息出错。

⑶ 放电现象：电气设备中各种放电现象，如火花放电、辉光放电、弧光放电、电晕放电都会产生高频幅射，会使计算机发生电压、电流冲击，尤其是弧光和火花放电，对计算机设备非常有害。

⑷ 射频干扰 :空间中的各种无线发射、广播、电视、雷达、高频加热、焊接、淬火、短波理疗等电子设备的电磁场会通过电磁辐射对计算机造成干扰。测试分析表明,工作场附近的电场强度,一般在 10 ~90 V/m,有时可以达到 1 000 V/m,磁场强度在几十安/米以上,电磁场干扰有时达到 135 dB,严重影响了计算机的可靠性。实验和实践都表明,超过 5 V/m 的电场,超过 15 A/m 的磁场和 50 Qc 的电磁干扰,就会使计算机及其磁记录设备中的信息出错或丢失,使计算机无法正常的工作。

2. 自然干扰

属于自然干扰的有雷电干扰、宇宙干扰、大气放电干扰、地球热辐射干扰和地爆电磁脉冲干扰。

⑴ 雷电干扰 :这是自然界产生的弧光放电现象,其感应产生的尖峰电压可以达到 10 kV 以上,而且电流强度很大,严重威胁计算机设备安全。据报道,1983 年美国很多计算机和通信设备遭到雷击损害。1985 年一次雷电竟使美国一栋 15 层楼内的计算机全部损坏。我国某气象中心引进了日本的 M－170、160 大型机两次受到雷击。1986 年 7 月 19 日,太原某厂的 PDP-11/73 被雷击损坏。铁道部成都铁路局引进的 VAX－11 机 1985 年 7 月 1 日雷击损坏了主机 1 台、终端 5 台。沈阳铁路局 1986 年 4 月 9 日因雷击损坏微机 3 台。在国内外,类似事件已经发生多起。

⑵ 宇宙干扰 :宇宙干扰就是指地球以外的能源干扰,包括太阳能产生的无线辐射,其干扰电压取为 1/10 μV 左右。

⑶ 电磁脉冲 :这是指地球内部核爆炸产生的电磁脉冲,它以电磁辐射的形式,穿透计算机的防护层或者通过计算机的各种孔口进入计算机,感应出电压和电流,干扰计算机正常运行。

⑷ 大气放电 :其干扰电压在 0.1 ~1.0 μV 左右。

这些自然干扰,会产生随机电流。轻则增加电噪声干扰,使计算机信息出错 ;重则使器件击穿,会使计算机设备损坏。各种电磁干扰的幅频特性如图 7.11－2 所示。

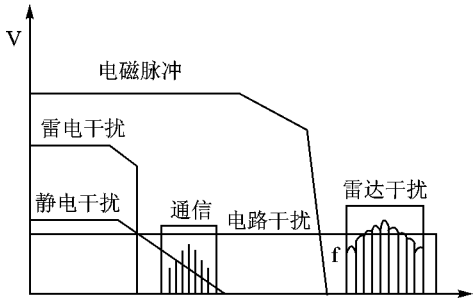


图 7.11－2 各种电磁干扰的幅频特性

3. 静电干扰

静电危害是计算机、半导体器件的“大敌”,是造成微机半导体器件损坏的重要原因。据美国电子工业界的统计,由于静电损害每年造成了计算机及其元器件损坏就达到 5 亿美元以上。电磁干扰不仅使磁记录设备破坏,还会使计算机设备外壳产生静电感应。MOS 器件的氧化层通常为热生长的 SiO_2,其临界电压强度为 5 - 8x10↑(- 6) V/cm,MOS 器件的栅极厚 1

500 (A),理论上击穿电压可以达到 75 ~ 120 V,但实际上由于元器件往往存在缺陷,因此,实际击穿电压很低。MOS 器件的电容性结构和 SiO_2 高绝缘性能,使得即使有很小的电量也会感应出很高的静电电压,使 MOS 器件栅介质击穿。一些半导体器件损坏时的静电放电电压如表 7.11 -1 所列。

表 7.11 -1 半导体器件损坏时的静电电压范围

器件类型	ESD 损坏电压/V	器件类型	ESD 损坏电压/V
VMOS	30 ~ 1 800	OP - AMP	190 ~ 2 500
MOSFET	100 ~ 200	CMOS	250 ~ 3 000
CaAsFET	100 ~ 130	ECL	500 ~ 1 500
EPROM	100	SCR	600 ~ 1 500
SEFT	140 ~ 7 000	SCHOTTKYTTL	1 000 ~ 2 500
SAW	150 ~ 500	TRANSISTORS	380 ~ 7 000
DIODES	300 ~ 3 500	FILMRESISTORS	300 ~ 3 000

当机房湿度过低 (如相对湿度低于 20% 时),因为摩擦也会产生静电,当人在绝缘性能良好的地板或者木质地板上走动时,穿着的绝缘鞋和尼龙袜、丝绸工作服会被静电充电。行走前后两步之间同时又会放电,交替冲充放电电压达到平衡时就会出现 0.1 ~ 5 μQ 的感应电荷。人体质电容一般在 100 ~ 250 pF 之间,因此会产生出几十千伏的静电电压。据美国惠普公司的报告,当相对湿度在 10% ~ 20% 时,可以产生出表 7.11 -2 所列的静电电压,引起静电放电,使计算机损坏。

表 7.11 -2 在干燥空气中气体产生的静电电压

产生静电感应的场合	产生的静电电压
橡皮清洁电路	12 kV
泡沫塑料垫	18 kV
人在乙烯地板上走动	4 ~ 13 kV
人在地毯上走动	12 ~ 39 kV
穿尼龙或丝绸工作服在普通工作台上	500 V ~ 3 kV

三、计算机中电磁干扰耦合形式

电磁干扰源产生的电压或者电流干扰波,通过耦合进入计算机,使计算机电路损坏和使计算机系统信息丢失。要有效地抑制干扰,必须弄清干扰的耦合形式,然后才能采取相关的措施。理论分析及实践表明,这些电磁干扰信号是通过公共耦合介质 (如导线电阻、电容、电感、互感等) 进入被干扰的电路的。按耦合介质,可以将干扰耦合分为以下四种形式。

1. 直接耦合

直接耦合是干扰信号经过导线直接传导到被干扰的电路中,而造成对电路的干扰。在计算机工程中,干扰噪声通过电源线耦合进计算机电路,是最常见的直接耦合现象。对这种耦合的方式,采用滤波去耦方法能有效地抑制或者防止电磁干扰传入。

2. 共阻抗耦合

当两个电路电流经过一个公共阻抗，一个电路上的电流在该阻抗上产生压降，这样就会影响到另外的电路。这种通过公共阻抗把耦合电压加到另一个电路的耦合形式就是共阻抗耦合。产生共阻抗耦合干扰的阻抗可以是接地线的地线公共阻抗，如图 7.11－3 所示；也可以是电路间的公共线电阻或电源输出电阻，如图 7.11－4 所示。其干扰电压可以表示为 $V_S = I_g \cdot Z_S$。式中 I_g 为耦合电流，Z_S 为公共阻抗。由此可知，抑制干扰 V_S 的根本方法：一是切断或阻隔地环路；二是限制公共阻抗。

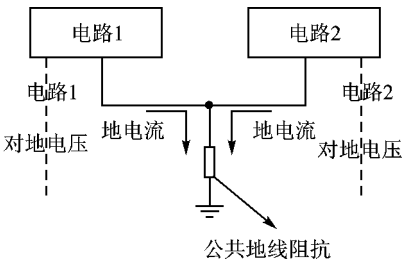


图 7.11－3 通过公共阻抗耦合的地电流

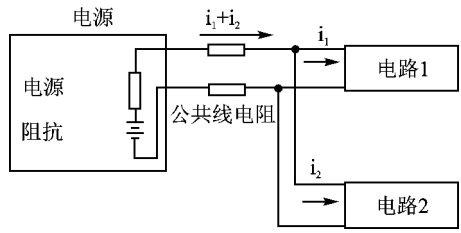


图 7.11－4 通过电源内阻和公共线阻抗耦合的干扰

3. 电场耦合

当干扰源产生的干扰波是以电压的形式出现时，干扰源与被干扰的电路之间会产生分布电容，如图 7.11－5 所示。干扰电压 V_G 会通过分布电容耦合到信号电路。经过分析简化，感应电压可以表示为：

$$V_S = \frac{j \cdot \omega \cdot C_j \cdot Z_S \cdot V_G}{1 + j \cdot \omega \cdot C_j \cdot Z_S}$$

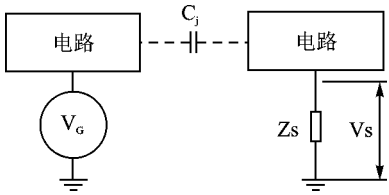


图 7.11－5

式中：V_G 为感应电源产生的干扰电压；V_S 为被干扰电路产生的电压；C_j 为分布电容；Z_S 为被干扰电路的等效电阻。

由此可知，抑制 V_S 的最根本方法是削弱分布电容。在计算机工程中，一是合理设计线路和布局元器件及电路；二是采用电场屏蔽，阻隔 C_j。

4. 磁场耦合

当干扰源产生的干扰波是以电流的形式出现时，干扰源对被干扰的电路会产生感应耦合。通过这种互感到耦合，称为磁场耦合。

互感器耦合产生的干扰电压 V_S 可以表示为：

$$V_S = j \cdot \omega \cdot M \cdot I_G = j \cdot \omega \cdot I_G \cdot B \cdot A \cdot \cos\theta$$

式中：B 是磁感应强度；A 是环路面积；θ 是夹角；j、ω 同前。

由此可知，抑制磁场耦合，一是要尽量削减 I_G，二是要尽量削减互感器值 M。在计算机工程中，一是采用磁屏蔽，对 I_G 进行分流和接地短路，二是尽量减少环路面积 A。对 A，一是要减少干扰源的电流环路面积，削弱干扰场；二是减少受损电路面积，抑制对干扰的接收。

5. 电磁感应

计算机在辐射电磁场的附近时，就会在电路中产生感应电势。抑制电磁干扰的基本方法

是采用电磁屏蔽并进行接地。

四、计算机中的干扰抑制

由以上分析可知,抑制电磁干扰的基本方法是滤波去耦、隔离、屏蔽和接地。

1. 滤波去耦

在一定的通频带内,滤波器削减很小,电流很容易通过。而在此通频带外,则衰减很大,能有效地抑制传输。因此,对于不需要的干扰信号,可采用不同形式的滤波器进行抑制。

滤波器有阻容滤波器、T 型、Π 型、L－C 型滤波器等多种结构形式,其插入衰减是不同的。在计算机工程中,可根据不同需要灵活运用 对于供电系统的噪声耦合,可采用感容或阻容滤波器把计算机电路与电源阻隔,以消除干扰源与计算机之间的耦合,或抑制噪声进入计算机电路 对于脉冲干扰,可采用组合抑制方法 对于放电干扰,可在开关电路端线与计算机设备之间加 0.25 ~1 μF 的电容滤波器来抑制 对于地环路还可以采用隔离变压器、光电耦合器等措施来阻隔地环路,切断地环路电流,抑制地环路干扰。

2. 电磁屏蔽

在计算机工程中,凡是受电磁场干扰的地方,都可以用屏蔽的办法来削弱干扰,以确保计算机正常运行。对于不同的干扰场要采取不同的屏蔽方法,如电屏蔽、磁屏蔽、电磁屏蔽,并将屏蔽体良好地接地。

对于电场耦合,要采用良导电材料屏蔽罩把电路包围起来,并将金属外壳接地。一可将杂散电容 (分布电容) 切断 ;二可将干扰信号引入大地。在计算机内部,常采用将地线布于插件上电路外围,并用地线或接地平面将各电路隔开,可以有效地消除电路之间、各插件之间的分布电容耦合进来的干扰。

在磁场耦合时,对低频磁场采用铁等高导磁率材料作屏蔽罩,利用它对 I_G 进行分流和短路,使通过空气的磁道大大减小。一是削弱干扰源 ;二是减少计算机电路对干扰的接收。对于高频磁场,可采用铝等低电阻率的半导体材料做屏蔽罩,利用屏蔽壳表面产生的涡流反磁场来排斥、抵消干扰场,以削弱和抑制磁场干扰。

对电磁感应耦合,可同时采用电屏蔽和磁屏蔽的方法抑制电磁辐射干扰。

3. 电磁隔离

电磁隔离是从电路上把干扰源和容易被干扰的部分隔离开来,使计算机与现场仅保持信号联系,但不发生电的联系。其实质是把引进的干扰通道切断,从而达到隔离电磁场干扰的目的。为了达到该目的,可实行弱电与强电隔离,以保证系统的工作稳定、可靠和人员的安全。常采用的隔离方法有光电隔离、变压器隔离、继电器隔离和布线隔离 4 种。

光电隔离是由光电耦合器来完成的。其输入端配置发光源,输出端配置受光器,因而输入和输出在电气上是完全隔离的。这种方法具有较高的电气隔离和抗干扰能力。

继电器的线圈和触点之间没有电气上的联系,也可以利用继电器的线圈接受电气信号,利用触点发送和输出信号,从而避免强电和弱电信号的直接接触,实现了抗干扰隔离。

脉冲变压器可以实现数字信号隔离,所以脉冲变压器隔离法在计算机系统中得到了广泛应用。

五、接地和电源

1. 接地系统

用接地系统,一是可以消除各电路之间经过公共阻抗时所产生的共阻抗干扰,避免计算机电路受磁场和电位差的影响;二是可以保证设备及人身安全。对于计算机系统的交流地、直流地、防雷地和安全地接地线要分开,不要互连。进入计算机的电源线、信号线均要采用金属屏蔽线或穿在铁套管内,并在屏蔽层两端接地,以防干扰及雷电入侵。计算机的直流地采用辐射地或栅格地较好。辐射式直流地应根据实际需要灵活选用串联一点接地、并联一点接地或者多点接地、混联接地,并保证其接地电阻符合国家计算机场地的技术要求。

2. 电源系统

电源电压波动或者负载幅度变化引起的瞬态电压、电流冲击,会通过电源线进入计算机,不但会使计算机信息出错,还会威胁计算机及其器件的寿命和安全。为了保证计算机系统的稳定性和安全性,供电电源最好不要与大用户合用一个电源变压器。在条件允许的情况下,可以考虑采用独立的变压器。对于大、中型计算机系统,还可以采用中频发电机组供电,然后,再经过交流稳压器和低通滤波器送直流稳压器。一般认为较理想的供电系统配置如图 7.11-6 所示。图中噪声滤波器和隔离变压器之间、隔离变压器与计算机交流线入口处均应采用屏蔽线,并且 AC 线前后不得交叉,以免发生耦合。

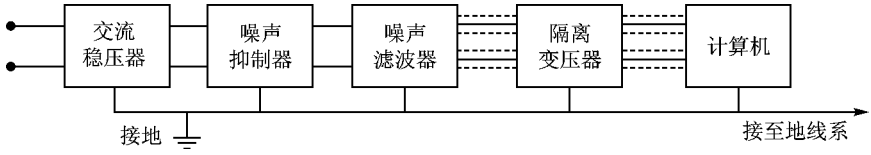


图 7.11-6 计算机供电系统

目前出现和应用的 UPS 电源是一种比较理想的供电设备。它可以对付各种突发脉冲、瞬变、噪声和电源电压降低及停电。它可以在没有外电输入的情况下独立支持计算机系统半小时。有的 UPS 电源甚至可以支持系统断电工作长达 45 min。在电网电压波动较大而系统电源要求又较高时应考虑选用一定容量的 UPS 电源。

综上所述,对于一个计算机系统来说,要完全避免和防止电磁干扰是不现实的。找到电磁干扰的原因,采取以上措施往往可以将电磁干扰控制在一定范围内,以致不影响和破坏系统的正常工作。随着科学技术的发展及计算机的广泛应用,电磁干扰问题将越来越突出,需要给予足够的重视。

参 考 文 献

1 李海泉. 微机系统的 RAS 技术 [M]. 北京:清华大学出版社,1996
2 陈世经. 试谈计算机系统安全运行诸问题 [M]. 广州:地球物理研究所,1986
3 李海泉. 微机系统的故障诊断与维护 [M]. 北京:科学出版社,2001
4 李海泉. 计算机系统安全技术 [M]. 北京:人民邮电出版社,2001

选自《计算机工程与设计》月刊,2002 年第 12 期

7.12 EMI 和屏蔽 (一)

用电路屏蔽来防止 EMI 问题由来已久,好的屏蔽可使发射机噪声信号保持在内部,而不外泄,同样也可让其他电路免受噪声的影响。发射机一般都产生谐波或其他噪声信号,如果辐射出去,将干扰其他应用。发射机天线终端发出的信号通过调谐和滤波被接收,调谐和滤波的目的之一是使信号更加洁净,如果发射机电路未被屏蔽,来自底座的直接辐射使滤波彻底失效。

理论上屏蔽可以起到相当好的作用,但实际上有些屏蔽毫无用途,有时带来的问题比解决的问题还多。发射机如此,通常的 RF 电路及所有的电路都是如此。科学和医用电子仪器一般很少用到高于 1 000 Hz 的频率,但同样遭受严重的电磁干扰,问题来自于几十 Hz 的电源线。

下面讨论一下屏蔽材料和方法,如图 7.12 - 1 所示。图 7.12 - 1 (a) 是一通用的三端“黑盒子”电路,A 是输入端,B 是输出端,C 是共用端。这里使用“黑盒子”电路的概念是为了讨论通用、一般化,其内部电路可以是一个电路,一个系统,如发射机、接收机、音频放大器或医用心电图仪。

屏蔽需要围绕着电路设置金属屏障,如图 7.12 - 1 (b) 点虚线所示,电路放入被屏蔽的盒子内,V_{AC}是输入电压,V_{BC}是输出电压。众所周知,两个导体在靠近时会产生电容,无论有意无意都会导致这种“偶然”电容。在底座布设绝缘线时产生的电容如图 7.12 - 1 (b) 中所示的杂散电容 C_{AD}、C_{BD}和 C_{CD}。这样的屏蔽并不完全有效,导致工作不稳定,在某些情况下甚至会形成振荡。从图 7.12 - 1 (c) 所示等效电路可更明显看出,C_{BD}和 C_{CD}形成电容分压器,输出经过 C_{AD}加到“黑盒子”电路的输入端,在特定条件下,可导致非常严重的 EMI/EMC 结果。

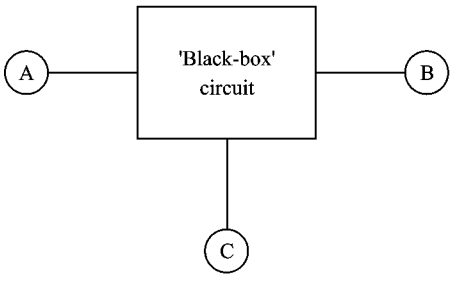
一、屏蔽准则 1

为了解决上述问题,需要采用屏蔽准则 1。

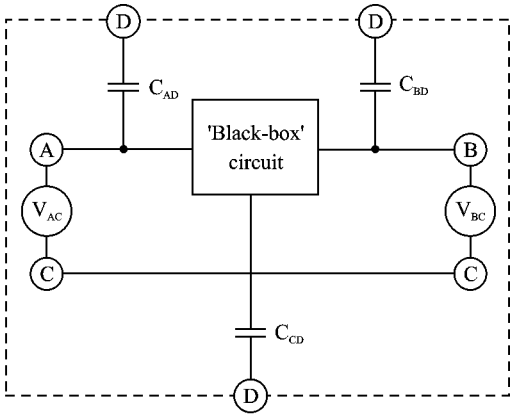
屏蔽准则 1 :屏蔽必须要连接到被保护电路的零信号基准点,即输入和输出的共用端。

有些情况下,共用端没有接地,电压非零,但相对于输入和输出信号仍是一零信号基准点。图 7.12 - 1 (d) 是该准则的应用例证,共用端 C 连接到屏蔽层 D,有效消除 C_{CD}和图 7.12 - 1 (c) 中的反馈路径,因此必须将屏蔽和共用信号线连接在一起。在上述几个例子中,“黑盒子”电路是单端型,内部电路的共用信号线可直接和屏蔽连接。

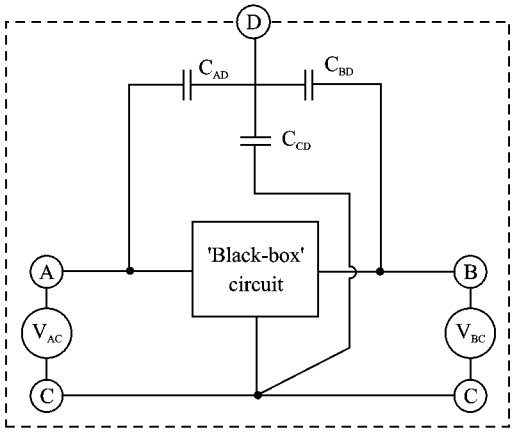
图 7.12 - 2 是稍复杂一点的情况,“黑盒子”电路已经处于屏蔽的隔板内,并向电阻性负载输出信号,负载通过屏蔽缆线连接到屏蔽隔板,同样屏蔽信号源 V_{IN}由另一屏蔽缆线连接到“黑盒子”电路输入端。假定主屏蔽盒内的信号共用端接到屏蔽板,然后到点 A 接地,信号源也接地,但到不了点 B,如图 7.12 - 2 (a) 所示。由于外部电路的影响或屏蔽电路内部两接地点之间电势差的作用,电流 I_G 经过地电阻 R_G,形成压降 V_G,电路所呈现出的输入信号即是 $V = V_{IN} + V_G$,这就是所谓的“地环路”问题。如图 7.12 - 2 (b) 所示。



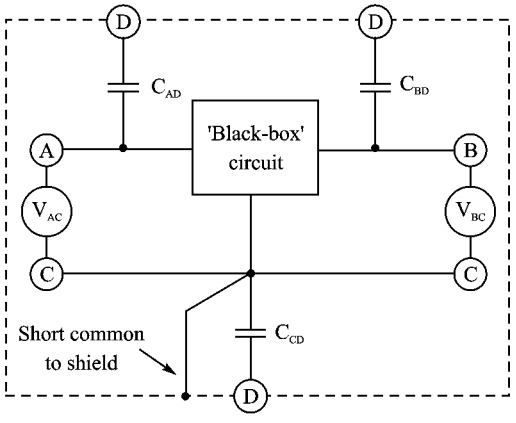
(a) 通用三端“黑盒子”电路



(b) 简单的电路屏蔽以及所产生的“偶然”电容



(c) 电路屏蔽等效电路



(d) 正确屏蔽及接地

图 7.12－1 **屏蔽方法**

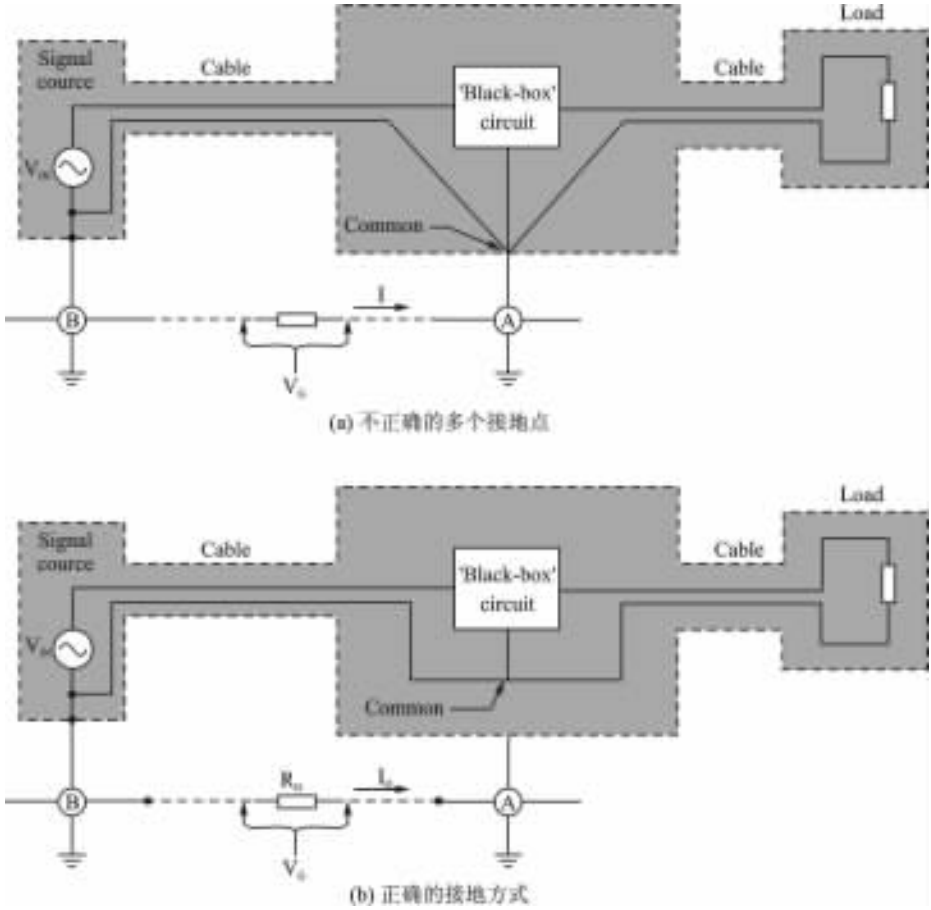


图 7.12－2 稍复杂的屏蔽方法

解决上述问题的关键是在信号端点 B 将屏蔽板与接地层连接，而不能在其他任何点。因此屏蔽准则 1 又可表述为：屏蔽、内部电路的共用端应和信号线接地端连在一起。在上述例子中，应断开 A 点连接，而只连接到同一 B 点。这种情况可以表述为共用阻抗（一个阻抗同时耦合同一电路中两部分）问题，如果共用阻抗上有电压，肯定会有问题出现。

二、屏蔽的两种途径

实现屏蔽有两种基本途径：吸收和反射，多数情形两者都采用。假定有一个强外场，在吸收起主要作用时，外场可透过屏蔽层，但大幅度衰减；在反射起主要作用时，外场被屏蔽层反射回去。吸收的方法通常用在低于 1 000 Hz 的磁场下，采用铁磁材料如钢以及特定磁屏蔽的 μ 金属合金（镍铁高导磁合金）。在高频下，尤其是电场比磁场更重要时，比较好的屏蔽材料是铜、黄铜和铝。

三、趋肤效应和趋肤深度

交流电流在流经导体截面时并不像直流电一样均匀分布，由于趋肤效应，交流电流仅仅集中在导体表面附近。因此导体的交流电阻大于直流电阻，从柱形导体的中心向表面，电流密度

呈抛物线分布。柱形导体的临界深度定义为电流密度下降到表面的 0.368 倍时的深度,该值被用来确定交流电阻。

屏蔽用的方形或盘形金属在有电流经过时也表现出趋肤效应,如图 7.12－3 所示,柱形导体的趋肤深度类似于临界深度。两种情况下,63.2% 的电流经过表面和该深度 (δ) 下的区域,其大小可由下式计算:

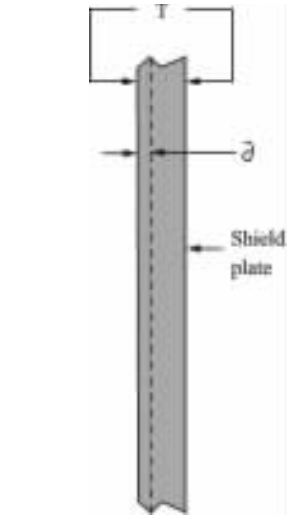


图 7.12－3 柱形导体的趋肤效应在条形导体同样出现,需同样注意

$$\delta = 2.602\ k/(F_{HZ})^{1/2} \quad (1)$$

δ 单位是英寸;F_{HZ}的单位是 Hz;常数 k 对于铜是 1,对于铝是 1.234,对于钢是 0.1。

在吸收损耗下,衰减是 8.7 dB/δ,如果用钢做屏蔽,在 60 Hz 下趋肤深度为 0.86 mm (0.034 in),用 1.6 mm 的钢材,即 1.84 倍的趋肤深度,则对磁场的衰减是 8.7 dB×1.84 =16 dB。

为了在 RF 条件下得到最大程度的反射损耗,屏蔽材料的厚度最少应是趋肤深度的 3～10 倍,在一定限度内,屏蔽层越厚,效果越好。例如,在 10 Hz 下,铝的趋肤深度是 0.025 4 mm (0.001 in),铜的趋肤深度是 0.02 mm (0.000 8 in),因此最小屏蔽厚度对于铝应是 0.254 mm,而对于铜应是 0.2 mm。有些文献中将屏蔽深度定为 3 倍趋肤深度,那只是最小应屏蔽的厚度,但效果有限。

四、接地板和接线大小

接地层可以是实际大地,但对多数电路而言接地板是印刷电路板或底座。如果印刷电路板作为接地板,在 RF 电路中最好使用双面板,上表面铜作为接地板,不宜使用细线或印刷电路板的通道作为接地线。

柱形导体线的交流电阻与线体直径和频率有关,具体可表示为:

$$RAC = kR_{DC}(F_{MHz})\,1/2 \quad (2)$$

每英尺电线的感抗约是 0.6 μH,在音频电路甚至许多低频 RF 电路中这或许无足轻重,但在频率升高时,感抗已很明显,在较高 HF 和低 VHF 频段则更高。如果该线接地,形成公用阻抗,在此感抗上形成的 RF 电压构成有效信号,导致问题出现。解决问题的办法是采用“星形”接地,即将所有的电路单元接地到同一点。如果信号源接地,则其接地点应作为总接地点。

五、屏蔽盒

许多厂商有预先做好的屏蔽盒,有些很不错,但并非全都如此。图 7.12－4 所示为一个屏蔽效果较差的铝屏蔽盒例子,它包括两个半壳,底壳向上弯成“U”形,顶壳稍大一点,刚好套进底壳,顶壳的一对导卡突耳嵌入底壳相应的凹槽。这种盒子适合于直流和低频 (最大几千 Hz) 等对外界 EMI 不太敏感的应用。

图 7.12－5 是改进的屏蔽盒,底壳基本保持原样,但顶壳采用重叠边缘,不再使用突耳—凹槽结构,这种屏蔽盒可适合于几 MHz,但对于 VHF 及以上频段不适合。

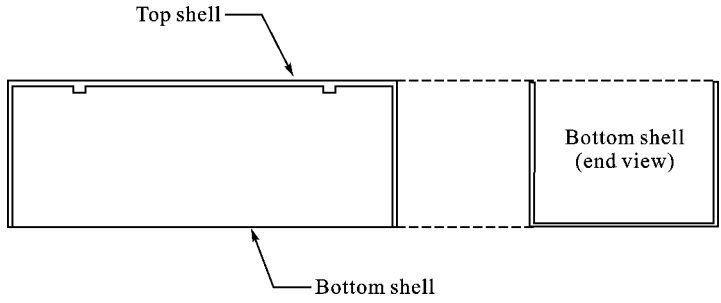


图 7.12－4 许多原型压制的铝和钢屏蔽盒只是简单结合,没有有效屏蔽,尤其是在较高频段

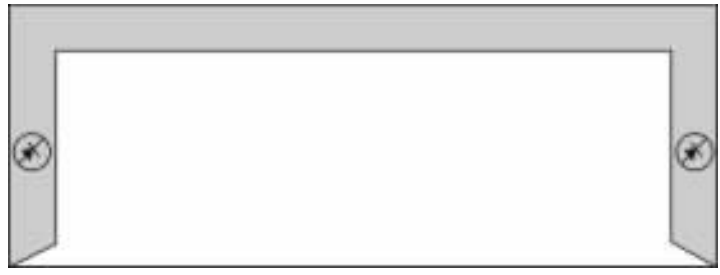


图 7.12－5 改进的屏蔽盒采用重叠边缘

六、屏蔽盒中的孔问题

理想的屏蔽应没有孔,但实际不可能。输入、输出以及电源等都要通过屏蔽盒。而且,电路产生大量的热量,需要通风散热,孔的大小应远小于所要屏蔽电磁信号的最短波长。对于螺孔和其他安装孔,一个通用的原则是间距不能超过 1/20 最短波长。图 7.12－6 为一种比较好的解决方案,螺孔间距小于 1/20 个最短波长。

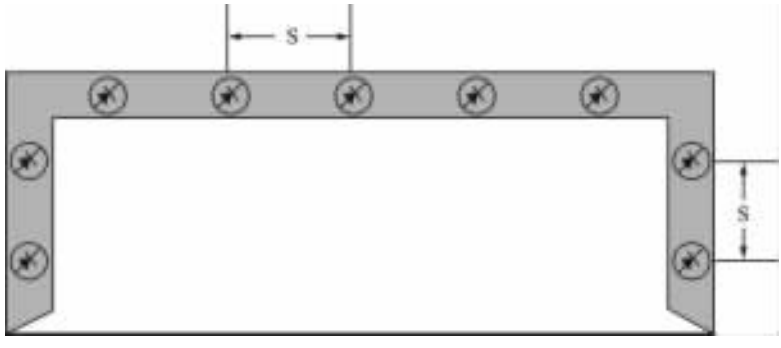


图 7.12－6 比较好的屏蔽解决方案,螺孔间距小于 1/20 个最短波长

宽的螺孔或安装孔间距对屏蔽效果影响很大,对于一个没有按上述原则确定的发射机屏蔽盒,屏蔽基本无效。

另外一种屏蔽盒如图 7.12－7 所示,底壳除上顶外全部封闭,顶壳带有 RF“指状结构”。该指状结构深入底壳金属,产生一个紧紧的 RF 接合,并在两部分连接处形成较低的阻抗。目前 SESCOM 公司已经生产这种屏蔽盒,由镀锡钢板制成。

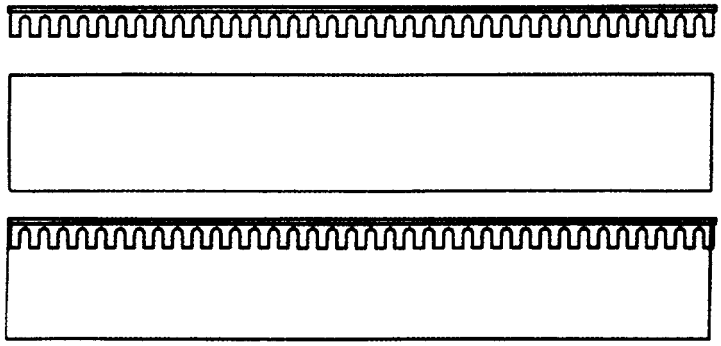


图 7.12－7 RF 频段特定的屏蔽结构

七、屏蔽盒中的狭缝问题

对屏蔽盒中的狭缝(见图 7.12－8)要特别注意,它们是产生信号泄漏的有效渠道,在狭缝达到 1/8 波长或更大时,会有相当多的辐射。如用做 RS－232C 等数字接口的“DB－x”型连接器,在装配到屏蔽盒时,即可形成辐射。当然,连接器并非仅有的“狭缝”,如果铝结构屏蔽盒各个部分只是简单平接在一起,如图 7.12－4 所示,非严紧结合将形成一辐射狭缝,最好的解决方法是使用有重叠部分的两个半壳,并紧密结合在一起。

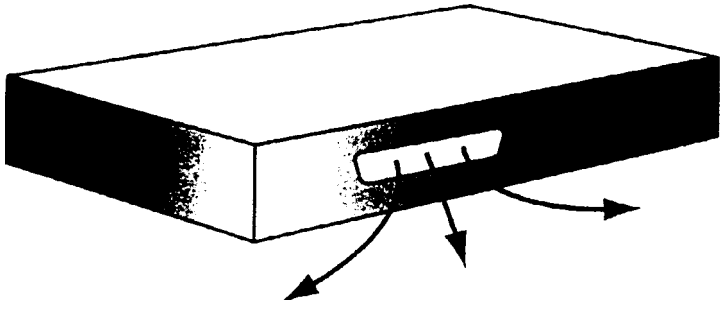


图 7.12－8 屏蔽盒中的狭缝容易成为信号辐射源

在使用内部屏蔽嵌板形成多个屏蔽仓或机械吻合不好时,也会产生其他偶然的狭缝,铜或黄铜做的屏蔽盒可使用焊点将这些嵌板牢牢接地,不产生“狭缝”效应。

选自《电子产品世界》月刊,2002 年第 5 期

7.13 EMI 和屏蔽 (二)

仔细观察接收机或其他科学仪器,会发现关键电路都是双层屏蔽,原因是每一层屏蔽都能有 60 ~100 dB 的信号衰减,当然,要达到 100 dB 并非易事,需要极好的屏蔽。假定一般的屏蔽层有 60 dB 的信号衰减,两层的衰减程度即可达到 120 dB 的量级。敏感仪器和高增益设备之所以采用双层屏蔽,特别是在前端电路部分,道理即是如此。

一、多仓屏蔽

在一些敏感仪器中要有多个内屏蔽仓,防止各个电路互相干扰,同时也防止来自外界的干扰。图 7.13 - 1 是一个总屏蔽盒内放有三个彼此屏蔽的电路。对于这种情况,通用准则 2 是:所需的屏蔽仓总数等于需要保护的电路数和输入电源数的总和。

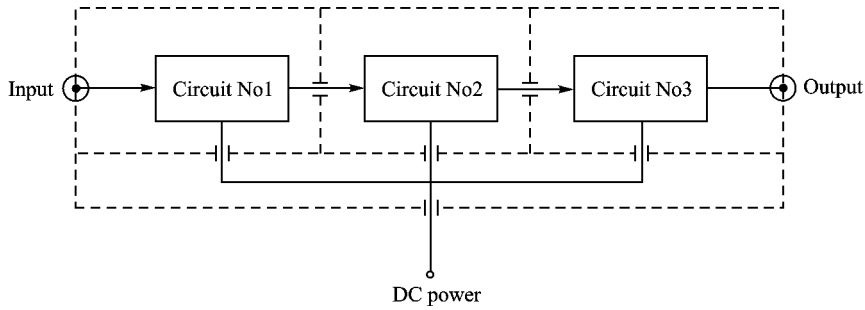


图 7.13 - 1 敏感仪器或高增益电路需要将各个电路单独屏蔽,再总体屏蔽

二、喷涂层屏蔽

许多仪器外壳由塑料或其他非导电合成材料制成,这些材料对于防止 EMI 毫无用途。有时,厂商会在外壳的内表面做一层导电薄膜作为屏蔽,如铜、铝和银等涂层,但这些对于屏蔽有时并不奏效,需要格外注意。所选择的涂层应是专门用来做屏蔽的,许多材料虽含有导电性金属,但金属的密度和厚度不足以构成屏蔽。

三、连接器、仪表和度盘

穿过仪器面板、后盖及所有屏蔽层的物体都会对屏蔽造成损害。图 7.13 - 2 表示了几个不同的问题和解决方法。数字和模拟仪表的情况如图 7.13 - 2 (a) 所示,屏蔽层所挖开的大洞可使 EMI 进入或离开屏蔽仓。此时可以做一个屏蔽盖在仪表的后面,除了一个带垫圈的小孔用作 DC 电源线的接入外,其余部分完全被封闭,也可以再在电源线上加一个馈通电容 (feed-through)。

有时需要用到多芯电缆连接器,所选择的电缆应是屏蔽型。连接器的后端应和仪表的处理方式相同,许多连接器厂商提供可选择的 EMI 屏蔽,但如果没有,必须要再做屏蔽。图

7.13－2 (c) 是多芯电缆和连接器的接线方式，假定通用电路或信号源在一个屏蔽的盒子内，需要用电缆连接到一个屏蔽的仪器。信号线、共用线和电源线由分立的导线连接，而信号源的屏蔽与电缆的屏蔽接到一起。

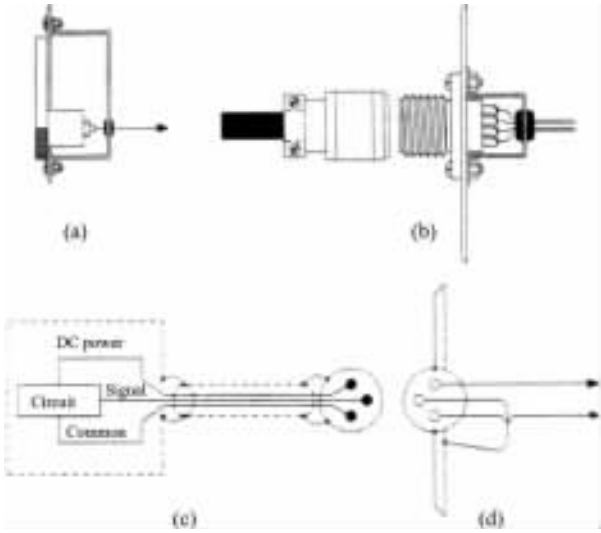
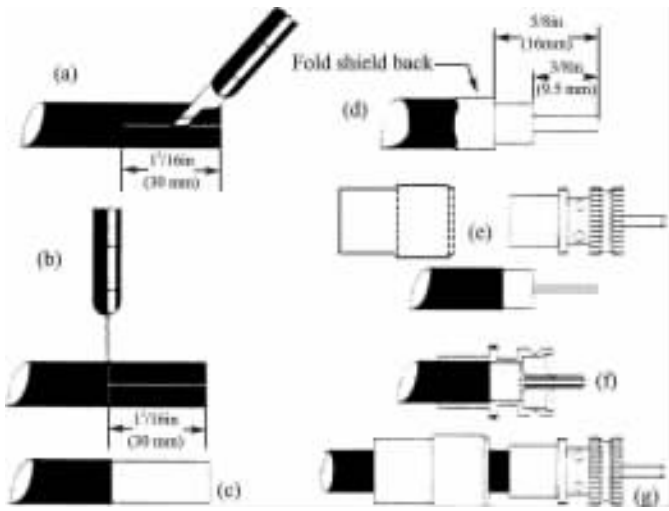


图 7.13－2 所有经过屏蔽层的物体都会对屏蔽造成损害，这里是一些解决方法

在电缆另一端，连接器的处理方法也同样：电缆的屏蔽层和连接器的屏蔽层接在一起，在仪器内部，各个部分的屏蔽及外壳的屏蔽应和地接在一起。

四、同轴连接器的安装

从表面看起来，在长长的同轴电缆安装“UHF”同轴连接器很困难，但是实际非常简单，图 7.13－3 示出了具体步骤。首先，将电缆切断到合适的长度，剥去最外边的绝缘层，长度约 30



(a) 沿电缆纵向切开一 30 mm 开口，避免损坏内部屏蔽层；(b)、(c) 横向切断绝缘层；
(d) 反折屏蔽层；(e) 剥去内部绝缘层；(f)、(g) 连接器安装

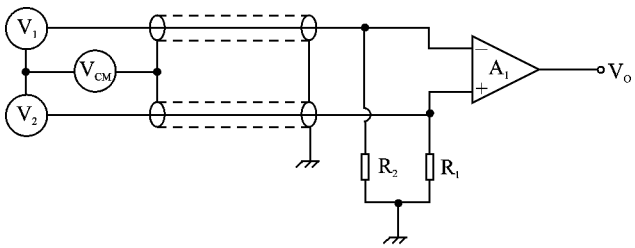
图 7.13－3 共轴连接器与电缆连接步骤

mm 左右,注意不要损坏内部的屏蔽网(见图 7.13－3(a)、(b)、(c))。然后,用电烙铁将屏蔽网敷一薄层锡使其硬化,但不要太热以免破坏内部绝缘层。锡不能太厚,否则插不进连接器。接下来的步骤如图 7.13－3(d)、(e)、(f)、(g)所示,按部就班,非常简单。

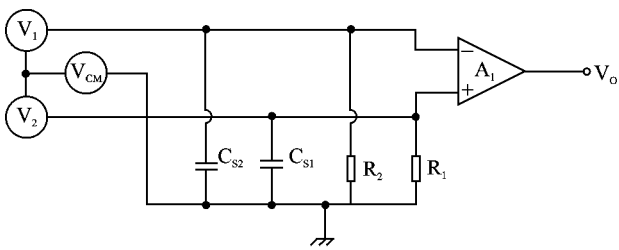
五、防护屏蔽

包括仪用放大器在内,差分放大器的特性之一是能够抑制来自外部环境的干扰信号,共模抑制技术是这种应用的基础。如果放大器通过电线连接到外部信号源,信号线容易受到附近 50 Hz 交流电源线的干扰,但由于输入线受到的干扰相同,因此相应的干扰被放大器抑制抵消。但多数情况下,由于内部或外部原因造成电路不平衡,干扰信号的抵消并不完全,也损害电路的共模抑制比。

图 7.13－4(a)表示了一种比较常见的情况,差分放大器连接到信号源的屏蔽端,对于局部的电磁场,屏蔽虽可起到一些作用,但仍有可能在屏蔽电缆由共模信号产生有效的差分信号电压,导致问题出现。图 7.13－4(b)示出的是一等效电路,由此可说明一对电缆如何由共模信号产生有效的差分信号电压。众所周知,在电缆的中心导体和周围的屏蔽线之间存在电容,除此之外,连接器以及仪器内部电线也有电容,这些电容集总在一起如图 7.13－4(b)中的 C_{S1} 和 C_{S2} 所示。只要源阻抗和并联阻抗相等,该两电容也相等,电路的平衡即不存在问题。但通常情况下,它们之间是不等值的,从而导致电路不平衡,共模信号电压对两个电容的充电不同,二者之间的电压差值构成有效的差分信号。



(a) 屏蔽电缆导致差分放大器产生差分信号电压



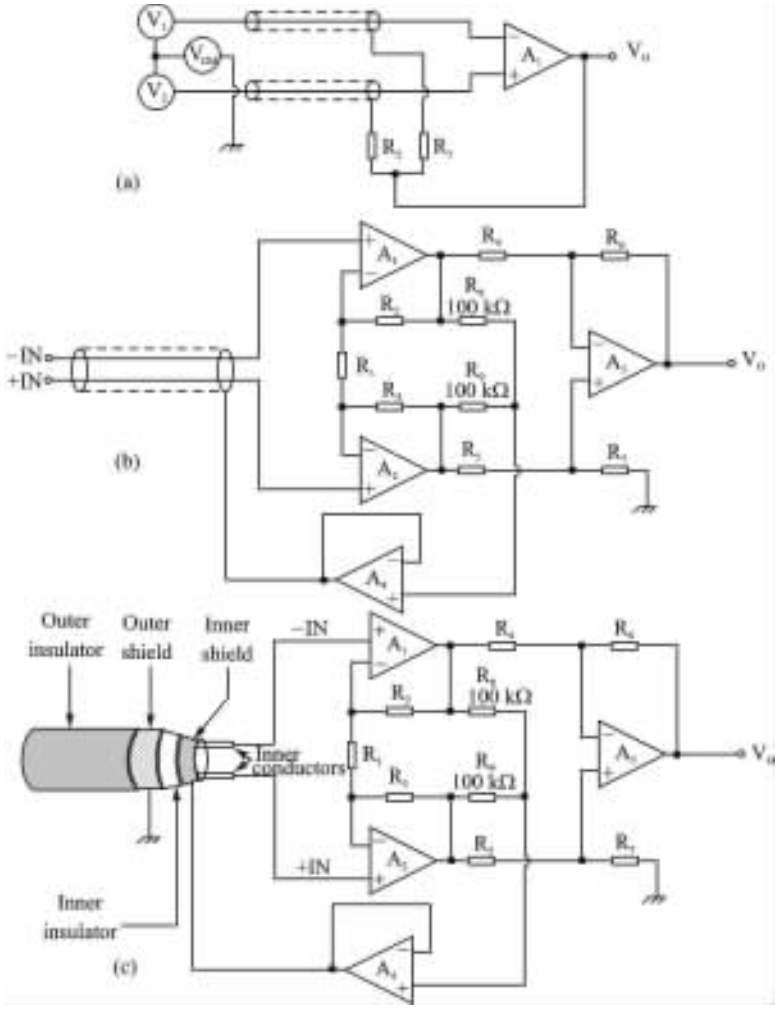
(b) 等效电路

图 7.13－4 干扰信号

图 7.13－5(a)所示为一个解决上述问题的低成本电路,两路输入信号的采样反馈给屏蔽,而该电路中屏蔽并没有接地。也可以用放大器的输出信号驱动屏蔽层,这样的屏蔽称为防护屏蔽,采用双屏蔽方式,每一个接到一输入线 或两个输入线共用一个屏蔽。图 7.13－5(b)是一个改进型仪用放大器的防护屏蔽例证,一个屏蔽覆盖两个输入线,当然也可以使用两个独立的屏蔽。在该电路中,两路输入信号在 R_8 和 R_9 连接处被采样,送到单位增益缓冲/驱动

“防护放大器”A_4 的输入端，A_4 的输出用来驱动防护屏蔽。

图 7.13－5 (c) 所示为最常见的防护屏蔽设计方法，使用两个屏蔽，输入电缆采用双屏蔽的绝缘线。防护放大器和内屏蔽相连，此时内屏蔽作为系统的防护屏蔽；而外屏蔽和以往一样，在输入端接地，起抑制电磁干扰的屏蔽作用。



(a) 低成本解决上述问题的电路；(b) 改进电路；(c) 常见的双屏蔽电路

图 7.13－5 防护屏蔽

六、PCB 防护屏蔽

防护屏蔽的概念可以扩大到印刷电路板上高增益放大器的输入引脚，图 7.13－6 所示为在印刷电路焊点周围放置防护环的方法。防护环多数情况下要接地，并要连接到输入电缆的防护屏蔽电缆的中心导体线连接到的焊点。上述电路中，为了使防护环连接到电路板焊点，其一端是断开的。如果电路板是双边的或多层，防护环可以封闭，焊点连接则处于最上层或中间某一层。

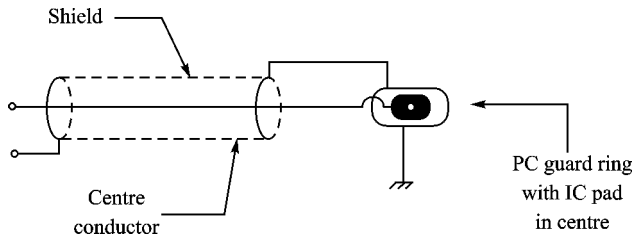


图 7.13－6　集成电路管脚采用屏蔽防护电路图

七、防护屏蔽专用放大器

市场上有许多专用集成电路，大多是为基于 PC 的仪器中模拟子系统而设计，然而在格式上千篇一律。作为一个通用 IC 放大器的例子，图 7.13－7 是 Burr-Brown 公司的 INA-101AG 器件，它是一个集成电路仪用放大器 A_1，增益由外电阻 R_G 决定。防护屏蔽直接和 IC 管脚 4 和 8 连接，两管脚再接到求和放大器的独立输入端，而输出端则驱动防护屏蔽。

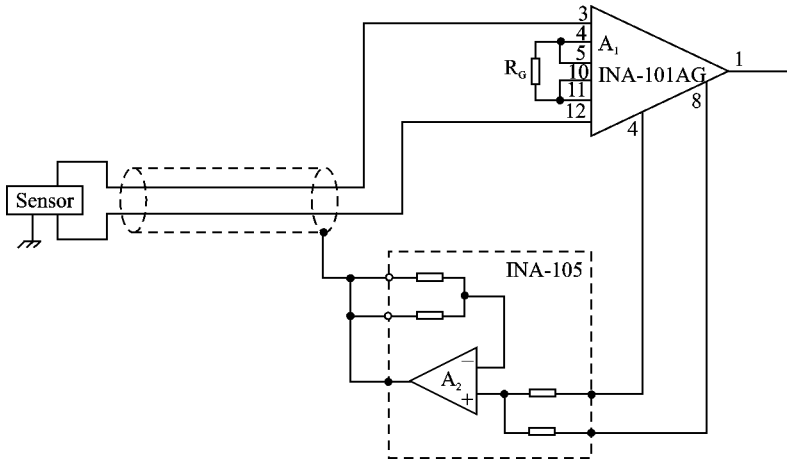


图 7.13－7　专用仪器放大器的屏蔽方式

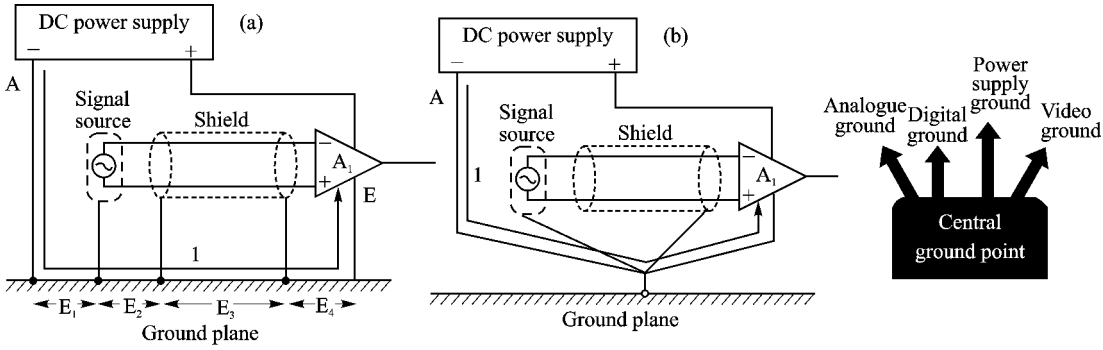
八、接地和地环路

来自电弧、雷电、电机和其他设备的脉冲噪声会干扰传感器及其电路，图 7.13－7 所示的屏蔽线虽有一些作用，但远非尽善尽美，这种条件下，滤波是有益处的。但滤波的方法也是“双刃剑”，须以适当的方式谨慎采用。信号线滤波容易加宽上升时间短的脉冲，衰减高频信号，在有些电路中，解决问题的同时，也带来同样多的麻烦。

就仪器系统而言，附近的其他电气设备也可形成感应信号，以 50 Hz 的交流电源为甚。最好只用差分放大器输入，因为它有高共模抑制比，信号可连接到两个输入端，成为差分信号，即使 50 Hz 的交流电源有一些影响，也是对两路输入同时有，最终相互抵消。

由于屏蔽中的地环路问题，共模信号也可以产生差分信号，起因是有太多的接地点，如图 7.13－8 (a) 所示。信号源屏蔽、输入线屏蔽、放大器以及直流电源的屏蔽接地到不同点，直流电流从 A 点出发直到放大器接地端 E。由于地阻抗存在，形成 $E_1 \sim E_4$ 等压降，这些电压被放

大器视为有效信号，如果电流 I 是变化的，问题更严重。解决办法是采用同一点接地，如图 7.13－8 (b) 所示。一些用在敏感图像或 CRT 示波器的放大器将信号接地和电源接地分开，有些甚至有几个信号接地点，尤其模拟和数字信号同时存在时比较常见。但这些都是特例。



(a) 在有太多接地点时，形成地环路　　(b) 解决地环路问题的方法是同一点接地

图 7.13－8　接地和地环路

总而言之，电路屏蔽并非易事，本文只是抛砖引玉，具体细致的解决方法有待设计者在实际工作中总结、完善。　　(麦凯)

选自《电子产品世界》月刊，2002 年第 6 期

7.14 微机接口设计中的静电冲击(ESD)防护措施

青岛大学自动化工程学院(266071) 潘松峰

一、ESD 的来源与危害

若两种材料的物体摩擦并分离,则两个物体可能会带静电,并产生电势差。具有不同电势的两个物体接近或接触时,静态电荷可从一个物体移到另一个物体。静态电荷的移动类似于一次很小的闪电过程,即静电放电(ESD)或称静电冲击。当静电冲击的能量足够高时,将造成半导体器件的损坏。静电放电 ESD,可能随时发生。例如,插拔电缆或人体接触微机接口器件的 I/O 端口、或者是一个带电的物体接触半导体器件,以及由于静电场或电磁干扰产生足够高的电压引起静电放电 ESD。

ESD 基本上可以分为 3 种类型:一是各种设备引起的 ESD;二是设备移动引起的 ESD;三是人体接触。在半导体器件的生产、电子产品的生产、电子产品的使用过程中,这 3 种 ESD 的防护是非常重要的。

在接口电路 IC 经受 ESD 时,放电回路的电阻通常都很小,无法限制放电电流。例如将带静电的电缆插到电路接口上时,放电回路的电阻几乎为零,造成可高达几十安培的瞬间放电尖峰电流,流入相应的 IC 管脚。瞬间大电流会严重损失 IC,局部发热的热量甚至会熔化硅片管芯。ESD 对 IC 的损伤一般还包括内部金属连接被烧断,钝化层被破坏,晶体管单元被烧坏。

ESD 还会引起接口电路 IC 的死锁(Latchup)。这种效应和 CMOS 器件内部的类似可控硅的结构单元被激活有关。高电压可激活这些结构形成大电流通道,一般是从 V_{CC} 到地。串行接口器件的锁死电流可高达 1 A,锁死电流会一直保持直到器件被断电,不过到那时 IC 通常早已因过热而烧毁了。

ESD 将损害接口器件,降低设备性能或产生故障,导致维修费用的增加。尤其是便携式电子产品,最容易受到人体接触 ESD 的损坏。因此,接口电路设计时,要注重 ESD 的防护问题。

二、ESD 保护的测试标准

欧洲共同体所规定的 ESD 保护有其严格的测试标准,从 1996 年 1 月已经开始实行这些标准,包括:

(1) 对于正常工作方式下 ESD 结构必须完全透明。

(2) ESD 过程中不能发生闭锁现象。

(3) 必须通过所有相关 ESD 测试标准:15 kV ESD 人体模式测试标准;8 kV ESD IEC 1000-4-2 接触放电模式测试标准;15 kV ESD IEC 1000-4-2 空气间隙放电模式测试标准;4 kV ESD IEC 1000-4-4 电气快速瞬变/猝发模式测试标准。

其中 IEC 1000-4-2 与 15 kV 人体模式测试标准之间的主要差别在于峰值电流,相同电

压下,IEC 1000－4－2 冲击的吸收电流要比人体模式高出 5 倍以上。±4 kV ESD IEC 1000－4－4 电气快速瞬变(FTB)/猝发模式测试标准是模拟产生开关和继电器的电弧放电结果。MAXIM 器件可提供±4 kV 的保护——两倍于 IEC 1000－4－4 标准的±2 kV 指标。

三、接口电路设计中 ESD 防护措施

在接口电路设计中,可采取以下 ESD 防护措施。

⑴ 气体放电管:一种充满氖气的蝶形电容器。电压超过一定值,例如 100 V 左右时,产生一个等离子区限制最高电压,它可以承受较大的电流,具有较小的漏电流。气体放电管可吸收高电平瞬态脉冲,但是,由于生成等离子区需要一定的时间,它不能吸收快速瞬变脉冲,不能用作主要保护手段。

⑵ 串联电阻:是重要的且廉价的保护器件之一,适当地选定电阻值和功率耗散值,可以替代许多昂贵的保护器件。

⑶ 电容器:是重要的保护元件之一,主要参数包括等效串联电阻(ESR)、电感应系数、电流容量和电压容量。

⑷ 压敏电阻器:一种由金属(主要为锌)氧化物制成的保护器件。它的功能近似于齐纳二极管,响应速度比气体放电管快,但漏电流比较高,尤其是在信号接近于钳位电压时。

⑸ Transzorb 二极管:用于限制低压信号的快速瞬变,其功率耗散能力受其尺寸的制约。同压敏电阻类似,在接近击穿电压时,有较大的漏电流。节点电容较大,在快速瞬变系统中用二极管桥退耦。

⑹ ESD 结构:MAXIM 公司的专利技术,一种新颖的设计方案,它的左右类似于双向二极管,被集成在器件内部,既不会增加任何输入输出管脚的等效电容,又节省了电路板面积,适合于 ESD 和 FTB 保护。采用内部集成 ESD 防护功能的接口器件比使用普通无防护功能的器件价格要贵,但增加的费用比起外加防护二极管的费用要低。下面,以 MAXIM 公司开发的一系列接口器件为例,介绍几种典型接口电路设计中 ESD 的防护措施。

四、应用实例

1. 串行通信接口应用

在 RS－485 系统中,差动接收器能抑制较大的共摸电压,因此用差动信号进行数据传输,提高了噪声环境中数据传输的可靠性。但是为了防止静电这类较高压对器件的损伤必须采用不同类型的保护措施。由于人体相当于一个充满电荷的电容,因此即使手碰一下 IC 也可能将其损坏。而这种情况很容易发生在安装 RS－485 电缆的过程中。为了避免这种损坏出现,我们选用 MAXIM 在其接口 IC 增加了 ESD 保护结构的芯片 MAX485E。RS485 接口电路示意图如图 7.14－1 所示。这样可以保护发送器的输出端或接收器的输入端在高达±15 kV 静电电压时不会被损坏。

同样,在 RS232 接口电路中,可以选择 MAX202E/232E 等芯片设计具有 ESD 防护措施的 RS232 接口。

2. 模拟量输入通道

多路模拟量输入通道设计时,除了选择合适的多路开关外,还要考虑过压保护以及 ESD 防护问题,一般须额外增加器件,电路繁杂且效果并不理想。若采用 MAXIM 公司的 MAX4558

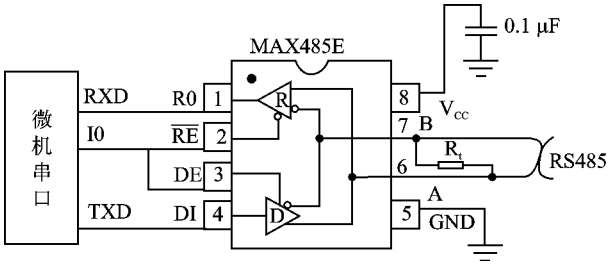


图 7.14－1　RS－485 接口电路示意图

等具有 ESD 防护功能的器件,则问题迎刃而解。同时,MAX4558 是与 CD4051 引脚兼容的 8 到 1 的多路模拟切换开关,单 +5 V 供电时,可切换单极性电压,导通电阻典型值为 220 Ω;±5 V 供电时,可切换双性电压,导通电阻典型值为 110 Ω。多路模拟通道接口设计示意图如图 7.14－2 所示。

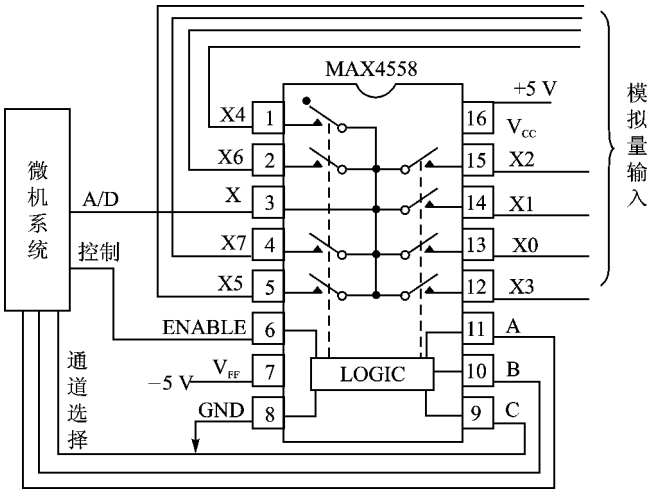


图 7.14－2　多路模拟通道接口设计示意图

当微机系统控制端有效时,通道选择信号(C、B、A)可选择某一路模拟输入信号(X0 ~ X7)切换到输出端(X 端),从而进入微机 A/D 通道。由图 7.14－2 可以看出,线路简单实用。

3. 开关量输入通道

在设计开关量输入通道时,同样要考虑静电冲击的防护、过压保护等问题,还要考虑消除开关抖动问题。按传统的设计方法,将增加软件或硬件成本,如消除开关抖动的常用方法是软件消抖动或硬件消抖动。而选用 MAXIM 开发生产的 MAX6816 系列产品(MAX6816/6817/6818 分别为 1/2/8 路开关输入电路),由于该系列产品具有防抖动、ESD 防护和过压保护功能,则大大简化了开关量输入通道的设计,如图 7.14－3 所示。

MAX6818 输入端(IN1 ~ IN8)可接入 8 个干触点开关,有三态输出控制端($\overline{EN}$)。在图 7.14－3中,当$\overline{EN}$ = 0 时,开关状态(SW1 ~ SW8)输出(OUT0 ~ OUT7)经微机系统 I/O 端口读入微机。当开关输入状态改变时,$\overline{CH}$端由高电平变为低电平,可作为中断请求信号。每次三态输出控制端$\overline{EN}$有效时,即对 MAX6818 进行读操作后,$\overline{CH}$端被重置。

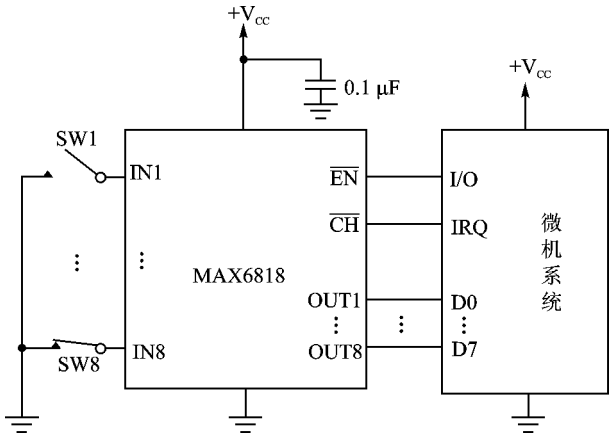


图 7.14－3 开关量输入电路

五、结束语

静电冲击（ESD）对微机接口电路的危害是非常严重的，必须引起足够的重视。在微机系统设计时，采取有效措施，提高静电冲击的防护能力，以提高微机系统的可靠性和性能。虽然增加了部分成本，但最终会大大降低维护成本和因故障维护造成的停产损失。

选自《仪表技术与传感器》月刊，2002 年第 7 期

7.15 单片机应用系统中去除工频干扰的快速实现

天津大学 谌雅琴 李 刚 叶文宇

针对工频干扰的特点,本文使用参考文献[1]所提出的自适应相干模板法。这是一种极其简单、有效的滤除工频干扰的算法,十分有利于单片机快速实现,在采样率不太高的情况下,能达到实时滤波。该算法之所以利于单片机快速实现,是因为算法本身多数为加法和减法运算,不涉及乘法运算,且通过合理的选择 M 值,可将除法运算巧妙地简化为移位运算或更简单地直接甩掉低位字节[2]。

一、自适应相干模板法

1. 滤除工频干扰的原理

参考文献[1]所提出的自适应相干模板法,是根据工频干扰的特点,从原始信号中得到工频干扰的模板,再从原始信号中减去该模板,达到滤除工频干扰的目的。

假设 X(n)为原始信号,S(n)为其中的有用信号,N(n)为工频干扰信号,则

$$X(n) = S(n) + N(n)$$

定义模板信号为

$$M(n) = \frac{1}{M}\sum_{i=1}^{M} X\left(n - \frac{f_s}{f_g}\cdot i\right) = \frac{1}{M}\sum_{i=1}^{M} S(n - k\cdot i) + \frac{1}{M}\sum_{i=1}^{M} N(n - k\cdot i)$$

式中,f_s 是信号的采样频率,f_g 是工频干扰的频率(50 Hz)。在自适应模板法中,要求 f_s 为 f_g 的整数倍,即 $f_s = k\cdot 50\text{Hz}$(k 为正整数)。

由于 N(n)为周期信号,若 S(n)为零均值信号,当 M 足够大时,有

$$\begin{cases}\dfrac{1}{M}\displaystyle\sum_{i=1}^{M} N(n - k\cdot i) = N(n)\\[2ex] \dfrac{1}{M}\displaystyle\sum_{i=1}^{M} S(n - k\cdot i) = 0\end{cases}$$

所以,只要从原始输入信号中减去模板信号就能达到滤除工频干扰的目的,即

$$S(n) = X(n) - \frac{1}{M}\sum_{i=1}^{M} X(n - k\cdot i) \qquad (1)$$

对式(1)两端取 Z 变换,可得该系统的传递函数为

$$H(z) = 1 - \frac{1}{M}\sum_{i=1}^{M} Z^{-k\cdot i} \qquad (2)$$

2. 幅频响应特性

根据系统传递函数式(2),利用 MATLAB 语言,对不同采样频率、不同 M 值的幅频响应特性进行比较,如图 7.15-1 所示。

从图 7.15-1 所示 A 组可看出,该滤波器不仅对 50 Hz 有滤波效果,对所有频率为 50 Hz 整数倍的信号都有滤波作用。因此,若采用自适应相干模板法滤除工频干扰,则当有用信号频

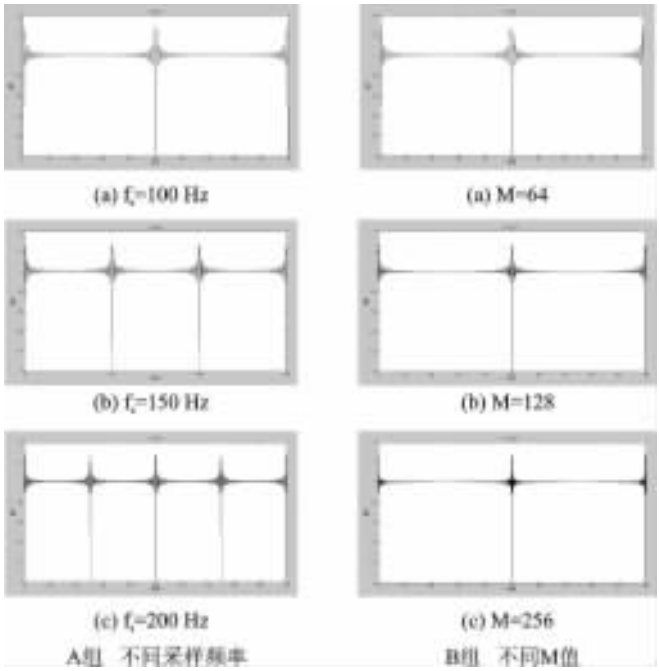


图 7.15－1 系统幅频响应特性

带范围较宽,信号采样率较高时,将对工频干扰 50 Hz 及其各谐波信号都有很好的抑制作用。

因此不仅要求采样率为 50 Hz 的整数倍,而且要求有用信号的最高频率分量不超过 100 Hz,否则,频率为 100 Hz 的有用信号也和工频干扰一样被滤除。这样,滤波后的信号将产生失真。

从图 7.15－1 所示 B 组可看出,滤波器的幅频特性受 M 值影响较大。当 M 值较大时,通频带的纹波系数较小,阻带宽度也较窄。也就是说,M 值越大对滤除理想的 50 Hz 来说效果越好 然而,实际工频干扰具有一定的频率变化范围,当 M 值大到超过某一值后将导致工频干扰滤除效果下降,因此,在实际设计滤波器时,M 值的选取要综合考虑。一般 M 值可选 256。

二、单片机实现

用自适应相干模板法去除工频干扰,可以达到实时滤波,这由信号的采样频率、单片机的速度决定。若信号的采样频率不高,单片机速度较快,则在信号的采样间隔时间内就能实现工频干扰的滤除。因此,在使用该方法前,应大概估计信号滤波所需的时间(与信号的通道数成正比),再适当选择采样率和晶振。

为方便说明,下面以 A/D 采样精度为 16 位、单片机为 89C51、f_s = 200 Hz、M = 256 为例,来讨论单通道信号中工频干扰去除的快速实现问题。由于 A/D 精度为 16 位,因此,单片机中所涉及的运算一般为双字节或三字节加法或减法运算,且由于 M 值取为 256,使得除法运算也变得极其简单,直接简化为甩掉低字节即可。

1. 建立初始模板

如前所述,利用自适应相干模板法去除工频干扰的关键在于建立工频干扰的模板,而为实现连续滤波,首先需建立一个初始模板。

由于信号的采样率为 200 Hz,是工频干扰 50 Hz 的 4 倍,即在一个工频干扰周期内有 4 个采样值,所以,建立的模板包括 4 个值,对应 4 个不同相位的采样值。另外,由于 M 值为 256,因此,最初采集的 1 024 个(=256×4)数据(16 位)是用来建立初始模板的:将 1 024 个数据根据 mod(n/4)(n=0,1,2,…,1 023)取值的不同,分别将数据进行累加存入不同地址的内存单元中(共需 3 字节×4=12 字节)。当 1 024 个数据分别累加完毕,此时对应内存单元中存放的数据即为 $\sum_{i=0}^{255} X(m+4i)$(m = 0,1,2,3)。这 4 个数据分别取高 16 位(中、高字节,即求 $\frac{1}{256}\sum_{i=0}^{255} X(m+4i)$(m = 0,1,2,3))就是工频干扰的初始模板。

2. 滤除工频干扰

在建立初始模板之后,就可对信号进行工频干扰的滤除:对于第 1 024 个(n = 1 024)采样数据,由于 mod(n/4) = 0,因此,只要将该采样值减去初始模板值 0,即进行减法运算 $[XC1024) - (1/256)\sum_{i=0}^{255} X(4i)]$,就完成了该时刻信号的工频干扰的滤除。但为快速、连续、实时地实现 50 Hz 滤波,还需将初始模板值 0 进行修改,即将存放 $\sum_{i=0}^{255} X(4i)$ 的内存单元的内容修改为

$$[\sum_{i=0}^{255} X(4i) + X(1024) - X(0)] = \sum_{i=1}^{256} X(4i)$$

同理,对于第 1 025 个(n = 1 025)采样数据,此时 mod(n/4) = 1,要滤除该时刻信号中包含的工频干扰,只需减去初始模板值 1,即完成减法运算

$$[X(1025) - (1/256)\sum_{i=0}^{255} X(4i+1)]$$

当然,也需修改初始模板值 1,即将存放 $\sum_{i=0}^{255} X(4i+1)$ 的内存单元内容修改为

$$[\sum_{i=0}^{255} X(4i+1) + X(1025) - X(1)] = \sum_{i=1}^{256} X(4i+1)$$

依此类推,以后采入的每一个数据都做相应处理:滤波和修改模板,最终就可实现整个信号的快速、连续、实时去除工频干扰。由于滤波过程中涉及减法运算,而单片机对于有符号数的运算处理较复杂,因此,在进行减法运算之前,应先将被减数加上一个常数,以保证运算结果都为正值。图 7.15-2 为滤波程序流程框图。详细程序清单此处从略,读者若有需要可参看本刊网站:www.dpj.com.cn。

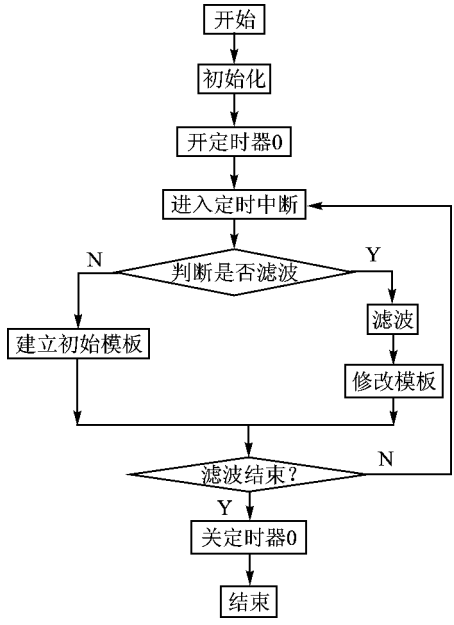


图 7.15－2 程序流程框图

三、结 论

本文主要讨论了自适应相干模板法去除工频干扰在单片机应用系统中的快速实现问题。与模拟滤波方法相比较,该方法具有成本低、滤波效果好等特点;而与其他数字滤波方法比较,该方法易于实现、速度快、频响特性极佳。

参 考 文 献

1 李刚,林凌,虞启琏. 滤除工频干扰的自适应相干模板法. 中国生物医学工程学报,1997,16(3):280～283
2 孙涵芳,徐爱卿. 单片机原理及应用. 北京:北京航空航天大学出版社,1988

选自《单片机与嵌入式系统应用》月刊,2002 年第 2 期

7.16　传输线路引起的数字信号畸变与抑制

信阳师范学院应用电子技术研究所 (464000)　周胜海　马建中

数字信号传输过程中的畸变是数字系统中的重要问题之一。引起畸变的原因主要有两个方面：一是由干扰信号引起；二是由传输线路的特性引起。对于前者，很多文献上都有较详细的讨论，人们一般较重视。本文将从应用的角度出发，讨论数字信号（矩形脉冲）传输过程中，由传输线路的特性引起的主要畸变及相应的抑制措施。这对数字系统的设计与实践都具有指导意义。

一、传输线上电容引起上升沿和下降沿变坏

各种传输线上一般都有电容，主要包括线间分布电容和信号线对地分布电容，一般后者比前者小得多。不同传输线上的电容可能相差很大。

以常用的平行圆导线和同轴电缆为例，其分布电容 C 分别由下式估算：

$$C = \frac{\pi\varepsilon_0\varepsilon_r}{\cosh^{-1}(H/d)} \quad (F/m) \tag{1}$$

$$C = \frac{\pi\varepsilon_0\varepsilon_r}{\ln(D/d)} \quad (F/m) \tag{2}$$

式中 $\varepsilon_0 = 1.854 \times 10^{-12}$ F/m 为真空介电常数，ε_r 为导体间介质的相对介电常数；式 (1) 中 H 为两根导线的轴心距离，d 为导线的直径；式 (2) 中 D 为外层导体的直径，d 为芯线的直径。

直径 0.5 mm、间距 2 mm 的两根远离机壳的平行圆导线，每 10 cm 的分布电容约为 1.4 pF。间距 2 mm、长 5 cm 的两条平行印制导线间的电容为 2～3 pF。数控装置和计算机用的双绞线每米有 50～60 个绞结时，分布电容 50～57 pF/m。

数字信号通过传输线时，其上的电容会出现充放电过程，引起矩形脉冲的上升沿和下降沿变坏，造成波形畸变，图 7.16－1 为畸变波形的示意图。电容越大或脉冲信号的频率越高，畸变越严重。

图 7.16－1　传输线上电容引起的畸变

所以，在实践中应注意以下几点：(1) 合理估算畸变的允许程度和电容的允许值，注意将信号接收端的输入电容一并考虑；(2) 不同传输线的电容相差很大，如同轴电缆抗干扰能力最强，但电容较大；(3) 传输线越短，电容越小；(4) 合理布线，可减小电容。

二、长线传输引起的振荡

设传输线的特性阻抗为 Z_O、驱动门(信号源)的输出阻抗为 R_S、负载阻抗为 R_L,则始端电压反射系数 ρ_S 和终端电压反射系统 ρ_L 分别为

$$\rho_S = (R_O - Z_O) / (R_O + Z_O) \tag{3}$$

$$\rho_L = (R_L - Z_O) / (R_L + Z_O) \tag{4}$$

可见,阻抗不匹配时,反射系数不等于零,即存在反射现象。

研究表明[1] 存在反射时,原始波在阻抗不连续点发生繁殖,原来的一个入射波到后来变成许多杂乱的纹波。由于出现多次反射与透射,若每次反射都滞后一个时间,反射波与原始波叠加起来,就使信号发生严重畸变。

对于脉冲信号,最常见的畸变是在脉冲的上升沿和下降沿出现过冲或振荡(振铃)现象,图 7.16-2 为振荡现象示意图。振荡幅度足够大时,就会在负载电路(接收端)的输入端产生非法的电平过渡,使传送的信息出错,并可能出现影响逻辑设计的寄生逻辑状态。在有些情况下,振荡幅度可能超过电压的极限值,造成器件损坏。

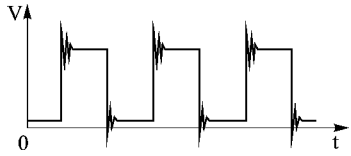


图 7.16-2 长线传输引起的振荡

畸变的程度主要取决于传输线的传输延迟时间 t_d 和脉冲信号上升时间 t_r。当传输延迟达到 $t_d = t_r/2$ 的程度,即信号的跳变过程在信号传到终端又被反射回始端之前就已结束,则必须在始端及终端进行阻抗匹配。对选定的传输线,可估算出相应的最大匹配线长度 l_{max}。长度超过 l_{max},即作长线处理,应进行阻抗匹配。

对具有连续地线层且信号走线在其邻近层的 PCB,延迟时间 t_d 由下式估算[2]:

$$t_d = 3.336\ 6\ (0.475\ \varepsilon_r + 0.67)^{1/2} \quad \text{ns/m} \tag{5}$$

式中 ε_r 为电路板材料的相对介电常数。对 FR4 材料板,$\varepsilon_r = 4.7 \sim 4.9$。高速逻辑器件的 $t_r \approx 2$ ns,据式(5)可得 $l_{max} \approx 17$ cm。传输线的 l_{max} 也可由下式估算:

$$l_{max} = t_r v/k \tag{6}$$

式中:t_r 为脉冲信号的上升时间(ns);v 为电磁波速度,$v = (1.4 \sim 2) \times 10^8$ m/s;k 为经验常数,$k = 4 \sim 5$。应当指出,当负载变重、延迟时间变长时,l_{max} 需相应缩短。

常用的阻抗匹配方法有 3 种[1-3]:发送端阻抗匹配、终端特性阻抗匹配和终端二极管匹配。每种方法各有特点,可根据具体情况选用。传输数字信号时,通常按传输电缆的长度来选择终端匹配方法。

ECL 电路是目前各种数字集成电路中工作速度最快的一种,目前 ECL 门电路的传输延迟时间已能缩短至 0.1 ns 以内,因而需要特别注意其传输线种类选择及阻抗匹配。具体方法可参考文献[4]。

三、传输线造成的幅度衰减

一般情况下,人们往往只注意到导线的电阻,事实上导线的电感却往往更大。即使在低频时,传输线上的电抗也可能比电阻要大。

一根直径为 d 的圆直导线在高于地平面 h 的位置时,其外电感 L 由下式估算:

$$L = 0.2 \ln(4h/d) \quad \mu H/m \tag{7}$$

进一步研究还表明:L 虽随 h 而变化(假设地面为回路),但当 h 达到几个 cm 时,L 渐近于其自由空间值,即不再随 h 的增高而变大;L 虽随 d 而变化,但变化数量不大,故通过选择粗导线的办法来降低 L 值是非常有限的。

两根载有均匀电流但方向相反的平行圆导线(直径为 d、中心距离为 H),自感 L 由下式估算:

$$L = 0.4 \ln(2h/d) \quad \mu H/m \tag{8}$$

宽度为 b、厚度为 d、长度为 l、间距为 H(单位均取 cm)的两条平行印制导线,每条线的自感 L 和两条导线间的互感 M 分别由下式估算[5]:

$$L = 2l[\ln(2l/b) + 1/2] \quad nH \tag{9}$$

$$M = 2l\left[\ln\left(\frac{2l}{b+H}\right) - 1\right] \quad nH \tag{10}$$

平行导线的电流方向相反时,其电感 L' 为

$$L' = 2(L - M) = 4l[\ln(1 + H/b) + 3/2] \quad (nH) \tag{11}$$

由于集肤效应,导线的有效电阻随频率的升高而变大。

对直径为 d 的实心圆铜导线,其交流电阻 R_a 与直流电阻 R_d 有如下近似关系[6]:

$$R_a = (0.038d\sqrt{f} + 0.26)R_d \tag{12}$$

式中 f 为电流频率。

配线用铜导线的阻抗可视为电阻与电感的串联,低频时二者都很小,但二者都随频率的升高而增大,对信号幅度的衰减也随之增大。例如,截面积为 5.5 mm^2 的铜圆导线直流电阻是 3.3 mΩ/m,1 MHz 时增加到 30 mΩ/m,10 MHz 时增加到 90 mΩ/m;直径 2 mm、长度 1 m 的铜圆直导线的电感约为 1 μH,1 MHz 时的感抗($2\pi fL$)为 6.3 Ω,10 MHz 时的感抗为 63 Ω;宽 2.5 mm、长 10 cm、间距 25 mm 两条平行印制导线的电感约为 156 nH;数控装置和计算机用的双绞线每米有 50 ~ 60 个绞结时,分布电感为 0.73 ~ 0.80 μH/m。

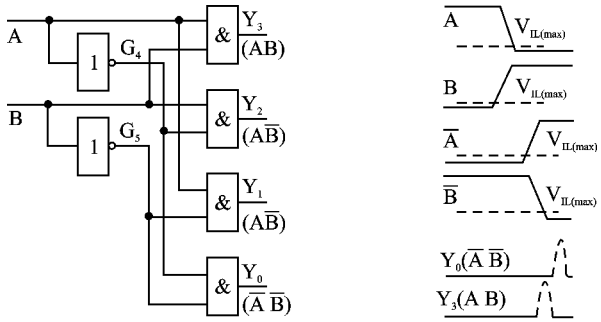
可见,在高速数字系统中,传输线造成的信号幅度的衰减不可忽视。

四、过渡引起的毛刺

在数字系统中,逻辑器件和传输线都有传输延迟时间。不同的逻辑器件和不同的传输线种类或长短的传输延迟时间可能相差很大(如 74LS、74AS、74ALS 3 个系列门电路的平均传输延迟时间分别为 10 ns、1.5 ns、4ns),甚至同类逻辑器件因内部元件参数的离散性等,也有明显差异。数字信号沿不同的路径进入同一逻辑门,因传输延迟时间不同而到达有先有后。这样,在输入信号的逻辑电平发生跳变时,可能在逻辑门的输出端出现不应有的持续时间很短的尖脉冲(毛刺)[7,8]。这种现象在数字电路中普遍存在,且电路的工作速度越高,出现的可能性

越大。

图 7.16－3 所示为 2/4 线译码器及其电压波形图。据电路图，在 A、B 的稳定状态下，输出 Y_0 和 Y_3 都应为 0 状态。然而，由于门 G_4 和 G_5 的传输延迟时间不同，在 AB 从 10 跳变 01 的过程中，Y_0 端有毛刺产生。此外，由于 A、B 在变化过程中到达 $V_{IL(max)}$ 的时刻不同，Y_3 也有毛刺出现。



(a) 电路图　　(b) 电压波形图

图 7.16－3　2/4 线译码器的过渡干扰

逻辑门的关门电平不一致也可能出现上述现象。

消除或抑制毛刺的常用方法有[7,8]：修改逻辑电路设计、逻辑门的输出端加装滤波电容、逻辑门的输入端或输出端加装 RC 积分电路等。

五、信号串扰引起的附加噪声

信号串扰是指传输线在传输数字（或模拟）信号的过程中，在其相邻信号线上产生噪声，此噪声与相邻信号线上传送的信号叠加，引起畸变。这是一种附加噪声。这种情况大多发生在多芯电缆、束捆导线和 PCB 上平行的印制导线之间。这种畸变的程度与相邻两信号线间的互阻抗、信号线自身的阻抗、信号的频率、信号的上升沿和下降沿时间等都有关系。

串扰引起数字信号畸变的情形多种多样，实践中常根据具体情况采取以下措施予以抑制[9～11]：⑴ 改变 PCB 结构和设计；⑵ 选择合适的传输线种类；⑶ 选择合适的布线设计；⑷ 把上升沿和下降沿时间相近的同级电平信号划分为一组，并保持一定距离；⑸ 用一根接地导线把两个相邻信号组的导线分开等。

六、结束语

数字信号的畸变会影响电路的工作可靠性和正常运行，损坏元器件，甚至造成大事故。且随着数字部件和系统工作速度的提高，这一问题更加突出[12]。所以，必须将畸变抑制到允许的程度。

在实践中，引起畸变的原因很多，往往又是多种原因同时起作用，因而畸变的表现形式通常较复杂。例如，上升沿和下降沿的变坏，往往伴随幅度衰减，还可能有附加噪声。一般要根据具体情况，借助理论分析的指导，在电路的设计、安装、调试等各个环节都考虑周全，有时需要多次调试，甚至需要修改设计方案，往往需要同时采取数种措施，才能将畸变抑制到允许的程度。大量实践证明，若将理论分析与实际经验有机结合，则往往收到事半功倍之效。所以，

本文的讨论对数字系统的设计与实践都具有指导意义。

参 考 文 献

1 马明建,周长城. 数据采集与处理技术 [M]. 西安:西安交通大学出版社,1999

2 赵元平,汤林森. 高速数字逻辑电路设计技巧 [J]. 电子技术应用,1997,(5):52~54

3 张捍东. 微机控制系统传输线的考虑 [J]. 电子技术应用,1996,(3):37

4 席德勋. 现代电子技术 [M]. 北京:高等教育出版社,1999

5 王幸之,王雷,翟成,等. 单片机应用系统抗干扰技术 [M]. 北京:北京航空航天大学出版社,2000

6 庄梓新,张纶,梁恺,等. 测量与控制核心系统手册 [M]. 北京:宇航工业出版社,1989

7 NELSON V P, NAGLE H T, CARROLL B D, et al. Digital logic circuit analysis & design [M]. Prentice Hall, Inc., 1995:206~210

8 韩刚,徐万玉. 工业电子控制装置的抗干扰技术 [M]. 北京:中国铁道出版社,1984

9 张松春,竺子芳,赵秀芬,等. 电子控制设备抗干扰技术及其应用 (第 2 版) [M]. 北京:机械工业出版社,1995

10 李海泉. 计算机的电磁干扰研究 [J]. 计算机自动测量与控制,2001,9 (6):1~3

11 邵贝贝. 微控制器 (单片机) 抗干扰能力与电磁兼容性 [J]. 电子技术,1996,(11):39~42

12 RABAEY J M. Digital integrated circuits:a design perspective [M]. Prentice-Hall, Inc., 1996, 438~502

选自《计算机测量与控制》月刊,2002 年第 10 期

第八章

DSP 及其应用技术

8.1 TMS320VC5402 电路设计中应注意的几个问题

合肥工业大学 (230009) 齐美彬

TMS320VC5402 (以下简称 C5402) 的时钟频率高达 100 MHz,机器周期为 10 ns,接口电源为 3.3 V,内核电源为 1.8 V,输入/输出引脚的逻辑电平复杂。它的应用系统的电路设计较普通单片机的电路设计复杂得多。电路设计的时候一般都会遇到输入/输出引脚的逻辑电平兼容、外围扩展电路时序、DSP 多余引脚的处理等问题,这些最基本问题的妥善解决是设计一个性能优良的 DSP 应用系统的前提条件。

一、接口电平的兼容性问题

C5402 的接口电源为 3.3 V,其输入/输出信号的电压特性较为复杂[1],如表 8.1-1 所列。

表 8.1-1 C5402 的输入/输出引脚电压特性

参数	引脚名称	电压范围
输入范围	所有输入引脚	-0.3~4.5 V
输出范围	所有输出引脚	-0.3~4.5 V
V_{IH}	$\overline{RS}$ $\overline{INT_n}$ $\overline{NMI}$ $\overline{BIO}$ $\overline{HCS}$ $\overline{HDS1}$ $\overline{HDS2}$	
	BCLKR0 BCLKR1 BCLKX0 BCLKX1 TDI TMS CLKMDn	2.2~3.6 V
	TCK $\overline{TRST}$	2.5~3.6 V
	X2/CLKIN	1.35~3.6 V
V_{IL}	$\overline{RS}$ $\overline{INT_n}$ $\overline{NMI}$ $\overline{BIO}$ $\overline{HCS}$ $\overline{HDS1}$ HDS2 BCLKP0 BCLKR1 BCLKX0 BCLKX1 TDI TMS CLKMDn X2/CLKIN	-0.3~0.6V
	所有其余输入引脚	-0.3~0.8V
V_{OH}	所有输出引脚	≥2.4V
V_{OL}	所用输出引脚	≤0.4V

注:DVDD = 3.3 V,CVDD = 1.8 V,时钟频率 100 MHz。

由表 8.1-1 可见,C5402 的输入电压的绝对范围是 -0.3~+4.5 V,除少数引脚外,其输入电平是与 TTL 逻辑电平兼容的,因此 C5402 的输入引脚仅能与 3.3 V 的 CMOS 电路连接,不能与 5 V TTL 电路、5 V CMOS 电路连接。5 V TTL/CMOS 电路的输出信号要经过电平转换后才能送给 C5402,否则可能损坏 C5402。

由于 C5402 的输出信号与 TTL 逻辑电平兼容,因此可以直接送给 5 V TTL 电路或者输入电平与 TTL 逻辑电平兼容的 3.3 V CMOS 和 5 V CMOS 电路。

在设计电路、选取器件时要综合考虑以上原因,尽量选择 3.3 V CMOS 器件,以简化电路设计。但是,很难保证整个系统都使用 3.3 V 的 CMOS 器件,对所有与 C5402 电平不兼容的信

号要进行电平转换。图 8.1－1 是一个基于 C5402 的视频图像采集电路，视频 A/D 转换器 TLC 5510 (U5) 为 5 V 电源，输入/输出引脚我们用 TI 公司的高速总线收发器对 C5402 的外部数据总线 (D_0 ~ D_7) 和其他一些输入信号 (如行、场同步信号、异步收发器中断信号) 进行了电平转换，很好解决了 C5402 的输入电平兼容问题。

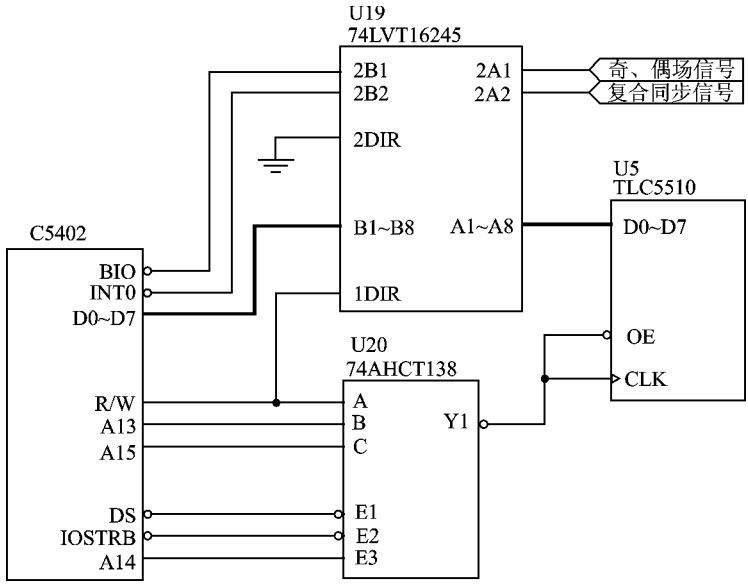


图 8.1－1 视频图像采集电路与 C5402 的连接

二、扩展电路的时序问题

时序问题是任何数字电路设计所必须重视的问题。在低速数字电路或普通单片机应用系统设计中要着重解决的问题是时序的逻辑性是否正确，而在高速数字电路设计中除了要解决时序逻辑性问题外，还要着重解决时序的时延性问题。C5402 的时钟频率高达 100 MHz，机器周期仅 10 ns。在不插等待周期的情况下，C5402 读外部存储器的典型时间为 1 个周期 (10 ns)，写外部存储器的典型时间为 2 个周期 (20 ns)，读/写扩展输入口的典型时间均为 2 个周期 (20 ns)。

为保证 C5402 在规定的时间内正确读/写外部扩展器件，首先要选用高速器件，要求扩展器件的读/写周期小于 C5402 的机器周期的 60% [2]，否则要插等待周期，这样 C5402 的高速特性就不能得到充分发挥。

其次，要求扩展器件的总线接口电路的时延尽量小，否则要另插等待周期。解决此问题的方法是尽量采用高速接口器件和单级接口电路。下面以地址译码电路为例来说明这个问题。图 8.1－2 所示电路为某 DSP 应用系统的 I/O 口地址译码电路，译码器 U_2 输出 Y_2 作为 U_3 的片选信号，由于 U_3 仅有一个控制端，所以将 R/W 与 Y_2 相或后作为 U_3 的输出控制信号。此电路的逻辑是正确的，但译码使用了 U_1、U_2 两级电路，若 74AHC138 和 74AHC32 的时延均为 8.5 ns，那么总的时延为 17 ns。C5402 不得不另插 2 个等待周期。如果将 R/W 信号直接与 U_2 的输入端 A 相连，就可以将 Y_2 直接作为 U_3 的输出控制信号，这样总的时延为8.5 ns，只需要另插一个等待周期。

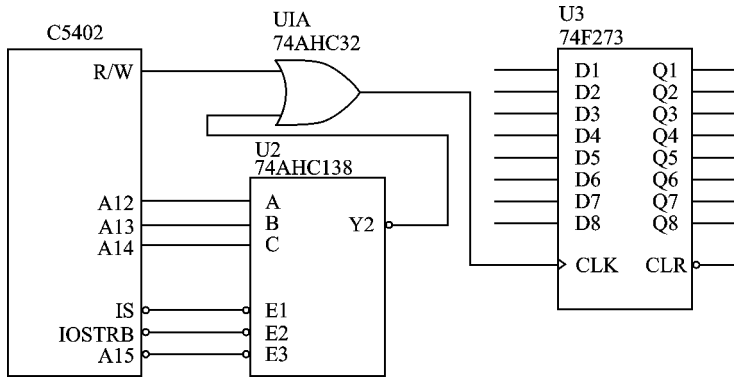


图 8.1－2 某 DSP 应用系统地址译码电路

电路如图 8.1－3 所示。

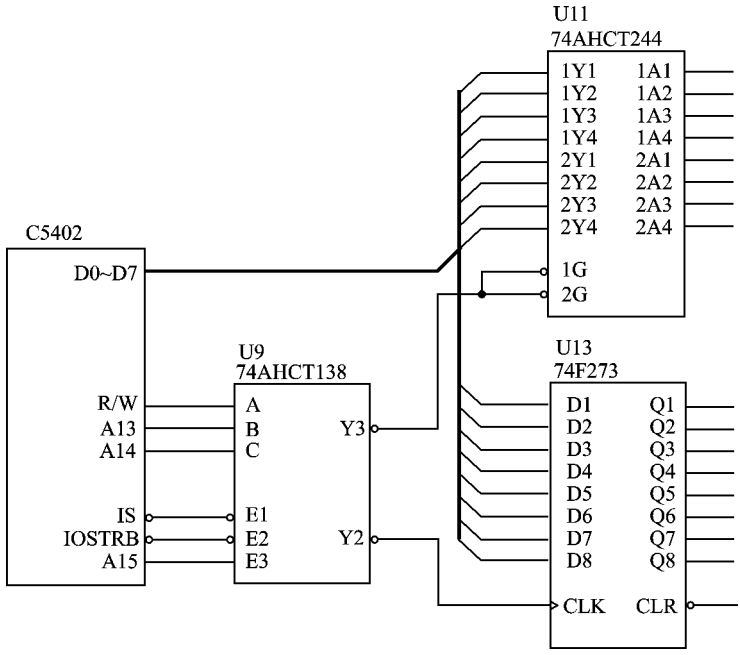


图 8.1－3 输入/输出口扩展

基于上面的考虑,我们采取了以下 3 个方面的措施：

⑴ 选用高速扩展器件,扩展 I/O 口的时延为 8.5 ns,扩展数据存储器的时延是 15 ns；

⑵ 选用高速总线接口器件,总线收发器的时延是 3 ns,地址译码器的时延是 8.5 ns；

⑶ 采用单级地址译码电路和总线收发电路。

三、DSP 多余引脚的处理

对集成电路多余引脚的处理原则为 :多余输出引脚可以悬空 ;多余输入引脚一般不能悬空,要进行相应处理 ;多余的双向引脚也应处理。在设计 DSP 应用系统时,同样要遵循这个原则。我们特别要强调的是以下三点：

(1) 没有使用的串行口或 HPI 接口的所有引脚可以不处理,不会引起 DSP 的误操作。

(2) DSP 数据总线的最高位 (D15) 最好与扩展器件数据总线的最高位连接,这样做的目的是避免符号位错误扩展。而多余的数据总线引脚可悬空,也可接上拉或下拉电阻。

(3) 特别要处理好输入引脚 HOLD 和 READY 的状态。要保证在没有外部设备请求占用 DSP 的外部存储器时 HOLD 为高电平;外部扩展器件不插硬件等待周期时 READY 应为高电平。

四、结束语

妥善处理好上述问题,可以简化系统的电路设计,提高系统的可靠性以及使用寿命。在合肥工业大学—TI DSP 联合实验室研制的便携式图像压缩卡[3]中,作者较好地解决了这些问题,取得了较好的效果。

参 考 文 献

1 TMS320C54x DSP CPU and Peripherals. 1999

2 戴明桢,周建江. TMS320C54x 数字信号处理器结构、原理及应用. 北京:北京航空航天大学出版社,2001

3 齐美彬,蒋建国,杨艳芳. 便携式图像压缩卡的设计. 微电子学与计算机,2001,18(2):56~58

选自《电子技术》月刊,2002 年第 6 期

8.2 DSP 系统中的外部存储器设计

南京大学电子科学与工程系 1012#信箱 (210093) 韩康榕

DSP 系统的存储空间分为内部和外部 2 种。尽管内部存储器具有存储效率高、使用方便的优点,但价格昂贵,其应用受到限制。因而外部存储器是 DSP 系统中扩展存储空间的重要手段。外部存储器亦分为 2 种 存储程序和固定数据的只读存储器 (ROM) 以及存储可变数据的可读写存储器 (RAM) 。对于常用的 DSP 系统,内置高速 RAM 区的大小足以满足系统需求,而内部程序存储器容量通常不足,需要外接外部程序存储器。

一、外部存储器的选择

从接口方式考虑,外部存储器分为串口存储器和并口存储器 2 种。在 DSP 系统中,由于要求高速交换数据,一般都采用并口存储器。下面介绍选择外部存储器时应注意的几个问题。

1. 电 压

目前在市面上多数外部存储器工作电压设定在 +5 V。而随着芯片集成度的增加,为降低芯片功耗、减少发热量,很多新型芯片将工作电压设定为 +3.3 V,内部电压降到 +1.8 V。如果仍然使用工作电压为 +5 V 的外部存储器,就必须在处理器芯片与外部存储器之间加入总线收发器或者专门的电平转换芯片,才能正常工作。因此,建议在 DSP 系统中使用同一工作电压的 DSP 与外部存储器,以方便系统的硬件设计,提高存取效率。目前常用的 +3.3 V 的外部存储器主要是闪速存储器 (FLASH MEMORY) 。它无需加高压来擦除、写入数据,使用非常方便。

2. 存取时间

DSP 系统中,对存储器而言,存取时间是一项十分重要的指标。外部存储器与高速的内部存储器不同,它通常无法实现与 DSP 的同步,而需要以软件或硬件的方法使 DSP 插入等待周期,这就人为地降低了 DSP 的工作效率。

如图 8.2 - 1 所示,假设 DSP 的机器周期为 25 ns,对外部的存储器的读/写操作是在单周期内进行的 (外部零等待状态) 。单个机器周期内完成的读操作分为 3 段 地址建立时间、数据有效时间和存储器存取时间。在这种情况下,DSP 要求外部存储器的存取时间小于 60% 的机器周期,即小于 15 ns。

图 8.2 - 1 DSP 读外部存储器的时序图

3. 容 量

外部存储器的容量大小应由系统需求来决定。在选取外部存储器时,除应注意总容量的大小外,还要注意数据总线的位数。在系统设计时,建议选用具有相同数据总线位数的 DSP 芯片与外部存储器,这样将有助于简化软件设计。如 DSP 芯片采用 16 位数据总线,则应尽量采用 16 位并口外部存储器。

二、DSP 与外部存储器的接口

1. 直接接口

根据以上的分析,如果外部存储器的存取速度可与 DSP 芯片同步,则可以采用直接接口法,接口电路如图 8.2-2 所示。图中,ADDRESS 为地址总线,DATA 为数据总线,R/$\overline{W}$ 为读/写信号(输出),$\overline{MSTRB}$为外部存储器选通信号(输出),$\overline{DS}$为数据空间选择信号(输出),$\overline{PS}$为程序空间选择信号(输出),READY 为数据准备好信号(输入),$\overline{MSC}$为微状态完成信号(输出)。READY 信号用来检测外部存储器是否已经准备好传送数据。若 READY = 1,表示外部器件已经准备好;READY = 0,表示没有准备好,此时 CPU 会自动插入 1 个等待状态(所有外部地址线、数据线以及控制信号均延长 1 个机器周期),并再次检测 READY 信号。若采用直接接口,可将 DSP 芯片的 READY 引脚直接接到高电平上。

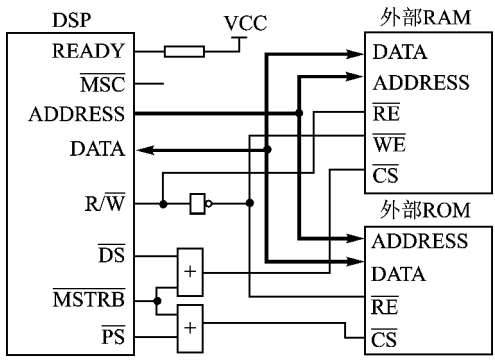


图 8.2-2 DSP 与外部存储器的直接接口

但由于外部存储器速度的限制,通常系统的设计不能采用直接接口的方法,必须以软件或硬件的方法实现延时的要求。

2. 带软件延时的接口

利用 DSP 片内的状态寄存器 SWWSR 可以设置软件等待状态(SWWSR 映射为数据存储器的 0028H 单元)。SWWSR 寄存器的定义如表 8.2-1 所列。

当设置完 SWWSR 寄存器(有的还可以设置 XSWWR)后,只要将图 8.2-2 中的$\overline{MSC}$引脚与 READY 引脚相连(即 READY 引脚不再是接 VCC)即可实现以软件延时 0~7 个机器周期(若设置了 SWSM = 1,则可以延迟 0~14 个机器周期)。此时若 SWWSR 设置为延时 0 或 1 个机器周期,CPU 将不检测 READY 信号;若 SWWSR 设置为延时 2~7 个机器周期,CPU 执行完最后一个软件等待周期结束后,$\overline{MSC}$引脚(微状态完成信号)将变成高电平。此时,READY 引脚随$\overline{MSC}$变成高电平,CPU 采样到的 READY 信号也是高电平,并开始接收数据。

表 8.2－1 SWWSR 寄存器

位	默认值	含 义
15	0	系统保留 (注 1)
14～12	1	I/O 存储空间插入的等待状态数 (注 2)
11～9	1	高位数据存储空间插入的等待状态数 (注 2)
8～6	1	低位数据存储空间插入的等待状态数 (注 2)
5～3	1	高位程序存储空间插入的等待状态数 (注 2)
2～0	1	低位程序存储空间插入的等待状态数 (注 2)

注 1 在'548,'549,'5402 等 DSP 中该位作为扩展程序地址的控制位。

注 2 在'549,'5402,'5410 等 DSP 中还有一个用作软件延时的寄存器 XSWWR,映像为数据存储器的 002BH 单元。XSWWR 的最低位 SWSM 设为 1 时,表示实际 DSP 软件延时的时间是 SWWSR 寄存器中所设的 2 倍。

3. 带硬件延时的接口

当需要插入 7 个 (或 14 个) 以上的等待周期时,仅仅用软件延迟是不够的,这时必须在软件延时的基础上插入硬件等待周期。硬件等待状态由$\overline{\text{MSC}}$、READY 信号以及外部逻辑电路形成,其硬件电路如图 8.2－3 所示。

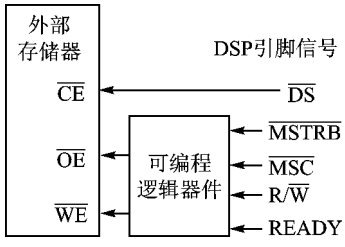


图 8.2－3 DSP 与外部存储器的延时接口

三、实 例

本实例是 TMS320VC5409 与 SST39VF400A 接口。TMS320VC5409 是 TI 公司出品的一种 16 位定点高速数字信号处理芯片,单指令周期 10 ns,最大运算能力为 100 MIPS (兆指令每秒)。SST39VF400A 是工作电压为 3.3 V、容量为 512 KB、存取时间为 120 ns 的闪速存储器。由前面分析可知,TMS320VC5409 与 SST39VF400A 必须采用硬件延时接口,故在此选用可编程逻辑器件 iM4A5－32/32 来实现。其接口电路如图 8.2－4 所示。

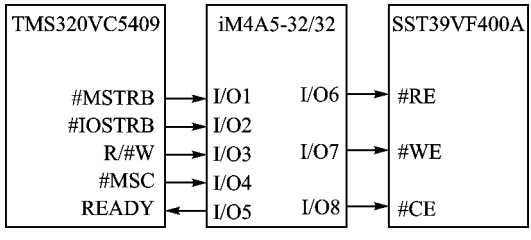


图 8.2－4 接口电路

iM4A5－32/32 的逻辑用 ABEL 语言描述如下：

```
MODULE DSPABEL
  DSP_MSTRB,DSP_RW,DSP_MSC,DSP_IOSTRB PIN;
  MEM_CE,MEM_WE,MEM_RE,DSP_READY PIN;
EQUATIONS
  MEM_CE = DSP_MSTRB&DSP_IOSTRB;
  MEM_RE = DSP_MSTRB&DSP_IOSTRB# ! DSP_RW;
  MEM_WE = DSP_MSTRB&DSP_IOSTRB#DSP_RW;
  DSP_READY = DSP_MSC;
END
```

TMS320VC5409 对 SST39VF400A 操作时，必须先擦除才可以逐字写入程序。擦除、写入操作通过不同的命令字来控制。具体程序实现如下：

```
erasesubprogram:
      STM     # 00AAH,AR3          第 1 个字命令
      PORTW   *(13H),5555H
      RPT     #1 000
      NOP
      STM     # 0055H,AR5          第 2 个字命令
      PORTW   *(15h),2AAAh
      RPT     #10 000
      NOP
      STM     #0080H,AR5          第 3 个字命令
      PORTW   *(15H),5555H
      RPT     #10 000
      NOP
      STM     #00AAH,AR5          第 4 个字命令
      PORTW   *(15H),5555H
      RPT     #10 000
      NOP
      STM     #0055H,AR5          第 5 个字命令
      PORTW   *(15h),2AAAh
      RPT     #10 000
      NOP
      STM     #0010H,AR5          第 6 个字命令
      PORTW   *(15H),5555H
      STM     #10 000,BRC         擦除等待
      RPTB    erasew ait－1
      NOP
      NOP
      RPT     #10 000
      NOP
```

```
erasewait：NOP
        ret

writesubprogram：
        STM     #Prognumber. ar7        程序数据代码的长度
        STM     #Pcode，AR2             程序数据代码首地址
                                        ；(指向 SST39VF400A)
        STM     #Code_start，AR6        编译后的程序代码
again：                                 开始写入程序
        STM     #00AAH，AR5             第 1 个字命令
        RPT     #1000 - 1
        PORTW   * (15H)，5555H
        RPT     #10 000
        NOP
        STM     #0055H，AR5             第 2 个字命令
        PORTW   * (15H)，2AAAH
        RPT     #10 000
        NOP
        STM     #00A0H，AR5，           第 3 个字命令
        PORTW   * (15H)，5555H
        RPT     #10 000
        NOP
        LD      * AR6 + ，A             写入
        STL  A，* AR2 +
        RPT     #1 000h
        NOP
loop：NOP
        STM     #10，BRC                擦除等待 100 μs
        RPTB    progwait - 1
        RPT     #1 000
        NOP
progwait：
        BANZ    again，* AR7 -
        NOP
        STM     #00AAH，AR5
        PORTW   * (15H)，5555H
        STM     #0055H，AR5
        PORTW   * (15H)，2AAAH
        RPT     #10 000
        NOP
        STM     #00A0H，AR5
        PORTW   * (15H)，5555H
        RPT     #10 000
```

```
NOP
STM     #8000H,AR5
RPT     #10-1
PORTW   *(15H),0ffffH
RPT     #10 000
NOP
ret
```

参 考 文 献

1 张雄伟,曹铁勇. DSP 芯片的原理与开发应用. 北京:电子工业出版社,2000

2 Texas Instruments Inc. TMS320C54x User's Guide. 1999

3 Texas Instruments Inc. TMS320C54xDSP Enhanced and Peripherals. 1999

选自《微型机与应用》月刊,2002 年第 3 期

8.3 TMS320C24x 的 C 语言与汇编语言的接口技术

上海交通大学信息与控制工程系 (200030) 张文荣 潘俊民 邹勇波

一、引 言

TMS320C24x (以下简称 C24x) 是美国 TI 公司推出的高性能 16 位数字信号处理器 (DSP),是专门为电机的数字化控制而设计的。这种 DSP 包括一个定点 DSP 内核及一系列微控制器外围电路,将数字信号处理的运算能力与面向电机的高效控制能力集于一体,可以实现对各类电机进行复杂而有效的控制。

对 C24x 的软件开发,可以用汇编语言,也可以用 C 语言实现。作为一种低级语言,汇编语言的代码执行效率高、运行速度快,可以充分发挥 DSP 处理器的硬件性能,但其开发的工作量大,程序可读性差。相比之下,C 语言具有可读性强、编程简单、调试方便等优点,适合编写结构和算法比较复杂的程序。然而,对于电机数字化控制来说,用 C 语言开发程序也有其明显的缺点:首先,C 语言代码有冗余,降低了执行效率,对于实时性要求很高的电机控制来说是不符合要求的;其次,C 语言无法实现某些底层的操作。

在实际应用中,可以将 C 语言和汇编语言结合起来编程,扬长避短,发挥各自的长处,这样就能够做到既满足实时性要求又能实现所需的功能,同时兼顾程序的可读性和编程效率。为此,必须了解 C24x 的 C 语言与汇编语言的接口技术。

二、C 语言与汇编语言接口的基本方式

对于 C24x 系统,C 语言与汇编语言的接口有两种基本方式。

1. 在 C 语言中嵌入汇编语句

C24x 的 C 语言也支持 asm 指令,可以用这条指令将一行汇编代码嵌入 C 程序中。可是这样做要担很大的风险,因为编译器完全不对嵌入的汇编代码进行分析检查,也就是说这些汇编代码很有可能破坏原来的 C 语言环境,从而导致不可预料的结果。在绝大多数情况下,笔者并不推荐用这种方式。但是,在汇编语言中开中断用 setc INTM,而在 C 语言中却找不到简单而有效的方法,这时嵌入 asm ("setc INTM") 就是最佳的选择。

2. 连接独立的 C 语言模块和汇编语言模块

单独编写 C 语言模块和汇编语言模块,将它们分别编译后连接成一个可执行目标文件。这种方式通常用于以下情况:某段代码要求实时性很高或者涉及底层的硬件,C 语言无能为力,这时用汇编语言将其编写成函数的形式,并由 C 语言模块调用,就能满足要求。使用这种方法时,用户必须遵循相关协议自行维护模块的入口、出口代码,管理堆栈。

三、C 语言与汇编语言接口约定

C24x 的 C 语言支持 ANSI C 标准。由于汇编语言很灵活,而 C 语言编程则需要一套完备

的工作环境,因此两者混合编程时必须保证汇编语言模块不破坏 C 环境,也就是说必须严格遵守以下一系列约定。

1. 标识符命名约定

C24x 的 C 语言程序编译后,在所有的标识符(包括变量名和函数名)前自动加一个下划线"_"。因此,如果需要 C 语言模块和汇编语言模块共享同一个标识符,就必须符合以下规则:在汇编语言模块中,标识符要以"_"开头,并声明为全局型,如

```
·bss   _var,1      ;定义变量
·global  _var      ;声明为全局变量
```

同时在 C 语言模块中,以 extern 修饰该标识符,如

```
extern int var;
```

2. 寄存器使用约定

C24x 系统有 8 个辅助寄存器(AR0 ~ AR7)可供使用,在 C 环境中这些寄存器都有明确的分工。

(1) AR1:Stack pointer(SP)软件栈的栈顶指针

C24x 只提供了大小为 8 个字的硬件栈,不能满足需要,因此,C 环境定义了一段特殊的存储器空间,作为所谓的软件栈。软件栈的作用是分配局部变量、传递函数的参数、保存处理器的状态、保存临时结果等。AR1 正是被用作指向该软件栈栈顶的专用指针。

(2) AR0:Rrame pointer(FP)帧指针

AR0 指向软件栈中函数局部数据空间的起始处。

(3) AR2:Local variable pointer(LVP)局部变量指针

AR2 存放局部变量的偏移量,与 AR0(FP)一起对局部变量进行寻址定位。

(4) AR6、AR7:寄存器型变量

在 C 语言程序中用 register 修饰的变量存放在 AR6、AR7 中。

(5) AR3 ~ AR5:自由寄存器

AR3 ~ AR5 没有特殊的约定,可以由用户自由决定其用途。

3. 函数调用约定

C 语言函数调用有一套严格的规则,要实现汇编语言模块和 C 语言模块之间函数的互相调用,必须遵从这些规则。

按照 C 语言的规定,函数的参数是按从右到左的次序压栈的。在 C24x 中函数的返回值存放在累加器 ACC 中。

对软件栈的维护是函数互调的关键。函数的参数是通过软件栈传递的,函数的出口地址存放在软件栈里,函数用到的局部变量也是在软件栈中分配的。图 8.3-1 是一个典型的函数调用时软件栈的分配情况。调用步骤如下:① 在主调函数的局部帧后将被调函数的参数依次压栈;② 程序指针(PC)跳转到被调函数的代码段;③ 保存出口地址;④ 保存主调函数局部帧指针;⑤ 分配局部变量;⑥ 调用结束前将以上所有内容弹出软件栈。

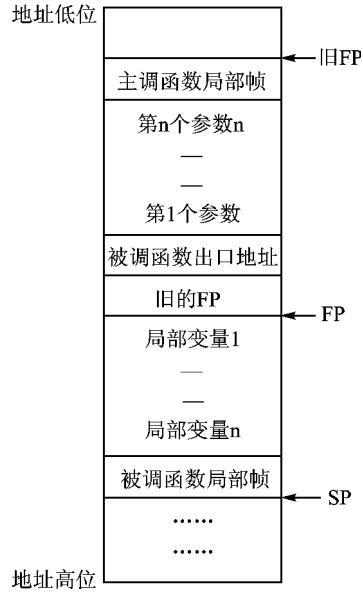


图 8.3－1 函数调用时软件栈分配情况

四、C 语言与汇编语言的应用

1. C 语言调用汇编语言函数

有时用户可能会将一些常数或数据表存放在程序存储器中，而 C24x 的 C 语言无法直接访问程序存储器。这时可以用汇编语言编写访问程序存储器的代码，由 C 语言来调用。下面给出具体实现的方法。

在 C 模块中调用函数的格式如下：

```
……
unsigned a;
a = readrom (0x100) ;/ * 函数的参数是要访问的程序存储器地址，返回该地址的数据 * /
……
```

确定了调用的格式，就能用汇编语言编写访问程序存储器的函数的实现代码了。

```
·global   _readrom          ;将函数名声明为全局型的标识符
readrom:
popd  * +                   ;将出口地址从硬件栈中弹出，放入软件栈中
sar AR0, * +                ;保存主调函数的 FP
sar AR1, *
lar AR0,#2
lar AR0, * 0 + ,AR2         ;分配局部变量的存储单元，并保存新的 FP，确定新的栈顶
lar AR2,#0fffdh             ;设置 LVP 相对与 FP 的偏移量，这里是－3
mar  * 0 +                  ;取得惟一一个参数的绝对地址，保存在 AR2 (LVP) 中
lacc  *                     ;取出参数
lar AR2,#1;
```

```
mar *0+                    ;LVP 定位到局部变量存储单元
tblr *                     ;从程序存储器读取数据,保存到局部变更中
lacl *,AR1                 ;把返回值存入 ACC
sbrk 3;
lar AR0, *-                ;把 FP 恢复为主调函数的 FP,并将局部变量空间释放
pshd *                     ;出口地址重新压入硬件栈
ret                        ;函数返回
```

2. 汇编语言调用 C 语言函数

C24x 的汇编语言没有除法指令,可以调用 C 语言函数来实现。做法如下:

在 C 语言模块中编写除法函数。

```
int division (int a,int b)
/*函数的参数是被除数和除数,返回值是商*/
 {
return a / b;
}
```

在汇编模块中可以这样调用上面的 C 函数:

```
·global_division           ;将函数名声明为全局型的标识符
stack. set #200h           ;定义软件栈的起始地址
.bss dividend, 1           ;分配被除数存储单元
.bss divisor, 1            ;分配除数存储单元
.bss quotient, 1           ;分配商的存储单元
……
lar AR1,#stack             ;初始化 SP
mar *, AR1                 ;当前辅助寄存器设为 AR1
lacl divisor               ;读取除数
sacl *+                    ;除数压栈
lacl dividend              ;读取被除数
sacl *+                    ;被除数压栈
                           ;注意上面的压栈次序
call_division              ;调用 C 语言的除法函数
sacl quotient              ;从 ACC 中取出返回值,即商的值
subk 2                     ;释放软件栈
……
```

3. 用 C 语言处理中断

C24x 提供了多个中断,如 INT1 ~ INT6、NMINT 等。程序存储器空间 0000h ~ 003Fh 是保留给中断向量的,每个中断向量占 2 字。一旦发生中断,程序指针 (PC) 就指向相应的中断向量,并通过中断向量映射到相应的中断服务子程序。比如,INT1 的中断向量在 0002h ~ 0003h,在此地址写入一条跳转指令,跳转到 INT1 的服务子程序,这样 INT1 中断产生时,PC 指到 0002h,执行跳转指令后就进入了服务子程序了。

C24x 的 C 语言有 interrupt 修饰符可以用来定义中断服务子程序,如

```
void interrupt INT1_Sevice ()
/* 假设这是 INT1 的中断服务子程序函数 */
/* 该函数必须用 interrupt 修饰,返回值类型是 void,且不能有参数 */
{
……
}
```

仅仅这样还不够,因为还没有定义 INT1 的中断向量,所以要另写一个汇编语言模块来定义中断向量,如下

```
·global_INT1_Sevice
……
0002h  b_INT1_Sevice
……
```

将以上的 C 语言模块和汇编语言模块分别编译后连接,就可以正确响应 INT1 中断了。

五、结束语

以上提到的 C24x 的 C 语言和汇编语言的接口技术已经在笔者参与开发的无刷直流电动机伺服系统的控制软件中得到应用。实践证明,利用这两种语言的混合编程可以获得高质量的代码,并具有较高的开发速度,是编制复杂的 C24x 控制软件的有效方法。

参 考 文 献

1 TMS320C2x/C2xx/C5x Optimizing C Compiler User s Guide. Texas Instrument Inc., 1997

2 TMS320F/C24x DSP Controllers Reference Guide. Texas Instrument Inc., 1999

选自《微处理机》季刊,2002 年第 3 期

8.4 DSP 环境下 C 语言编程的优化实现

南京邮电学院通信工程系 (210003) 胡金波 陈慧剑

一、引 言

DSP (Digital Signal Processor,数字信号处理器) 是一种具有特殊结构的微处理器。自 20 世纪 80 年代初诞生以来,DSP 在短短的十多年间里得到了飞速的发展。随着 DSP 性能价格比和开发手段的不断提高,DSP 已经在通信和信息系统、信号与信号处理、自动控制、雷达、军事、航空航天、医疗、家用电器等许多领域得到了广泛的应用。

与单片机相比,DSP 多用于算法比较复杂、乘加运算量比较大的应用,如通信、雷达、音视频处理等。为了追求代码的高效,过去一般用汇编语言来编制 DSP 程序。随着 DSP 应用范围不断延伸,应用的日趋复杂,汇编语言程序在可读性、可修改性、可移植性和可重用性的缺点日益突出,软件需求与软件生产力之间的矛盾日益严重。引入高级语言 (如 C 语言、C + +、Java),可以解决该矛盾。在高级语言中,C 语言无疑是最高效、最灵活的。各个 DSP 芯片公司都相继推出了相应的 C 语言编译器。

鉴于 DSP 应用的复杂度,在用 C 语言进行 DSP 软件开发时,一般先在基于通用微处理器的 PC 机或工作站上对算法进行仿真,仿真通过后再将 C 程序移植到 DSP 平台中。

按照软件开发的顺序,相应的优化工作包括两个部分:一是仿真环境中的优化,二是 DSP 目标环境中的进一步优化。本文主要探讨的是前者,给出了在 DSP 开发环境下有效 C 语言编程的策略,以获得最高效的编译代码;并针对策略,设计了具体实例,并比较了不同策略在 TMS320C54x CCS (v1.2) 和 TMS320C6000 CCS (v1.2) 环境下编译的结果。

二、DSP 开发环境下有效 C 编程的策略

基于通用微处理器的 PC 机环境中的优化工作是针对 C 程序的通用特性来考虑的。这方面的优化工作主要包括数据类型选择、数值操作优化、快速算法[6]、变量定义和使用优化、函数调用优化、程序流程优化以及计算表格化[6]等。

1. 数据类型

标准 C 语言提供了丰富的数据类型——整型、浮点、枚举、指针、结构、联合等。编程面对的问题是使用怎样的数据类型使编译生成的代码小、效率高。

整型有 signed 和 unsigned 之分,分别称为 char、shortint、int、long int、enum。ANSI C 没有规定每个类型的大小,它只是声明 short int 不大于 int,long 不小于 int,enum 和 int 具有相同尺度。这种模糊的定义影响了程序由一个处理器向另一个处理器的移植操作。为了避免这种影响,比较良好的编程风格是将数据类型按类型定义 (typedef) 在一个头文件中,当移植时只需要更改头文件即可:

	TMS320C54x [4]	TMS320C6000 [5]
U8		unsigned char
S8		char/signed char
U16	char/unsigned short/unsigned int	unsigned short
S16	signed char/short/int	short
U32	unsigned long	unsigned int
S32	long/signed long	int/signed int

使用浮点数是非常危险的,除非系统有浮点协处理器或专门针对浮点设计的处理器,使用浮点变量会使编码尺度膨胀。即使使用协处理器,消耗时间也很多。同时,浮点数的存储空间是可变的。IEEE 单精度浮点需 4 B,双精度需 8 B,扩展双精度需 10 B。一些小的 CPU 的交叉编译器仅支持单精度浮点型。

struct temptag {	struct temptag {
long a;	long a;
sohrt b;	long c;
long c;	long e;
char d;	long g;
long e;	short b;
char f;	char d;
long g;	char f;
} temp1;	} temp 2;

避免浮点操作的方法一般是采用浮点运算定点化,用定点函数运算替代浮点操作。对于定点 DSP 的操作,应仔细考虑硬件的限制。同时,在编程时要大致估计数据的范围,做到所采用的数据类型刚好满足要求,并尽量做到在一个 CPU 指令周期内完成数据载入。

由于 DSP 常采用可变的定位方式,结构的不适当声明会浪费很多 RAM 和 ROM 空间。如左图示例中两种不同声明方式,用 sizeof () 分析,在 C54x 中分别为 14 和 12 (单位为字 [4]),在 C6201 中分别为 56 个字节和 40 个字节,可见不适当地声明会导致内存空间的浪费。

当然,一些高性能的编译器,可根据内存空间优化各个变量的位置,但此时变量存储的次序可能和它们定义时的次序不同。

2. 数值操作优化

对数值操作优化,主要特别注意以下几点:

(1) 用比特的移位操作来代替 2 次幂整数的乘除法运算更为有效;

(2) 用查表法代替三角函数运算。特别是在 FFT 等程序中,同时将一些运行时计算的参数做成查找表或常数数值,这样可以将运行时的计算转化为编译时的计算,从而提高运算效率;

(3) 当使用浮点设备时,尽量使用浮点数据类型,这能够减小定点处理单元的负担;

(4) 尽量避免数值的上下溢出,除非是算法本身的需要。

3. 变量定义及使用优化

C 语言把局部变量放在堆栈中,这种访问是间接的,因此较慢。更为有效的方法是将变量放在堆 (heap) 中,有两种方法实现:一种是声明为全局变量;另一种是声明变量为 static。同时,要注意提高全局变量的重复利用率。

对于需要多次重复访问的变量,如 for 循环中的变量值,一般可以设置为 register 变量。声明变量为 register 能够提高效率,但必须小心使用。在某些编译器中,优化器会自动分配一些变量为 register。

在 C 语言程序中指针和数组是可以相互替代的。对数组的寻址是非常耗时的,特别是多维数组。因此,首先应降低数组的维数,再指针化。同时,配合 DSP 中寻址机构所支持的增量寻址,效率会大大提高,代价是降低了可读性。

4. 函数调用

函数调用往往产生大量代码。当 C 调用一个函数时,它首先把参数传递给寄存器或堆栈。如果函数参数很多,则调用开销将很大。此外,还需大量堆栈空间。最坏的情况是函数参数传递的是结构,编译器在调用函数时必须首先复制整个结构到堆栈。此外,若函数返回的是结构,调用程序保留堆栈空间,传递结构地址给函数,调用函数,然后函数返回。最后,调用程序还要清除堆栈,并将返回的结构复制到另一个结构。代码和堆栈的开销将是惊人的,特别是资源有限的 DSP 或其他片上嵌入式开发系统。为了避免这种开销,应禁止传递结构,一般用结构指针替代。如果结构是不可修改的,可用常量结构指针替代。

函数调用的另一方面开销是局部变量。这些变量定位于堆栈,因此增加了对堆栈的要求。如果这些变量需被初始化,则在程序每次被调用时均需做一次初始化。可以这样说,限制局部变量的数目也就是对堆栈空间的限制。

第三方面的开销是调用函数的返回值。如果要返回值,一般需要在函数返回前复制返回值到返回位置,然后把结果复制到调用程序中。如果函数的返回值赋值给一个变量或很少使用,可以考虑传递指向返回值的指针。被调用的函数可以直接改变返回值。

对于用 C + + 开发的用户,采用 inline 技术可以完全消除函数调用的开销,然而这增加了目标代码的大小。在这种情况下,应根据实际采用的编译器判断优化后程序生成的代码是否增加不大。

5. 程序流程设计

在 C 语言中,程序流程控制有 if…else, switch…case,do…while,for,while 等,它们的使用不当也会影响程序生成代码的大小和效率。下面,本文将分别分析使用判断选取控制语句和循环控制语句时应该注意的事项。

在使用判断选取控制语句时应减少判断转移。DSP 多采用流水线结构。如 TMS320C54x 中就采用了 6 级流水线结构,频繁的转换指令将使流水线难以发挥作用。另外 DSP 的大多数指令为单周期指令,但转移类指令却通常要耗费较多的机器周期。因此,应尽可能减少程序中的转移分支。一般通过对程序流的分析,许多判断转移可以用简单的条件组合来实现。

采用判断转移的 C 程序	进行逻辑优化的 C 程序

if (m_iSum < 0)
m_iSum = - m_iSum;
if (m_iSum > 0)
m_iSum = - m_iSum;
	m_iSum = abs (m_iSum);
	m_iSum = - abs (m_iSum);

当有多种选择时,switch…case 语句可读性强,然而它会带来很大开销,if…else 语句更灵活,但它需要更多的 C 代码,if…else 语句在实际的编译中可能会更为有效一些。另外一个需要考虑的是 switch…case 语句中参数可以是任意的整数类型,然而,若这些整数在 case 语句中

生成一系列整数,如 enum 类型,许多编译器将产生跳转表,这可以减少编译代码,而且平均下来,执行的效率也比较快。

C 语言提供 3 种类型的循环结构:for 循环、do - while 循环和 while 循环。编译器的任务是将指令映射为处理器的指令集。许多 DSP 设计有零开销循环处理能力。零开销循环不需要循环计数更新、测试和回跳指令,因此能够加速处理能力。

为了尽量实现循环的零开销,编译器必须知道循环的初始化、更新和结束条件。当循环表达式过于复杂或者含有的循环变更随循环体本身中的条件变化而改变量值时,许多编译器不生成零开销的循环。基于这种准则,循环表达式应写的尽可能清楚。并且尽量地对表达式做预处理,如将常数表达式移出循环,预先计算结果等。如下给出典型情况下的分析结果:

```
for (i = 0 ; i < NumRows * NumColumns ; i + + )
u8_array [i] = 0 ;
for (i = NumRows * NumColumns - 1 ; i > = 0 ; i - - )
u8_array [i] = 0 ;
```

上述实例,在 C54x 下,后者较前者节省了 1592CPU 周期,在 C6000 下节省了 480CPU 周期。同样,分析如下代码:

```
void strlower (char * s)
{size_t i ;
for (i = 0 ; i < strlen (s) ; i + + )
s [i] = tolower (s [i]) }
```

虽然在循环中字符串的长度没有改变,函数 strlen () 却被调用了 strlen (s) + 1 次。编译器不能优化这种多余调用函数的情况。一般情况下,一次函数调用返回一个不同的值,循环体对函数的结果产生影响。一种较好的编程思路是不对中间变量进行存储,循环改为 for (i = strlen (s) - 1 ; i > = 0 ; i - -)。

以上从多方面探讨了用 C 语言开发 DSP 软件时的一些优化考虑。为了最有效的用 C 编程,应该按"程序是怎样在汇编语言中执行"的思想来编程。随着实时操作系统、嵌入式操作系统、可视开发环境的引入,以及以 DSP 为平台的 C 编译器的功能不断完善,用 C 开发 DSP 应用将更便捷。

参 考 文 献

1 Robert Jan Ridder programming digital signal processors with highlevel languages DSP Engineering,2000

2 Numerix' Techniques For Optimizing C Code' http ://www. numerixdsp. com/

3 Joseph Lemieux J. Moving Efficiently from Assembly Language to C Class # 401 Embedded Systems Conference Papers,2000

4 TMS320C54x Optimizing C/C++ Compiler User's Guide. Texas Instruments Inc. ,2001 (6)

5 TMS320C6000 Optimizing Compiler User's Guide. Texas Instruments Incorporated,2001 (4)

6 刘朝晖,郑玉墙. 用 C 语言进行 DSP 软件设计的优化考虑. 空军雷达学院学报,2001 (6)

选自《现代电子技术》月刊,2002 年第 12 期

8.5 基于 TMS320C6000 高速算法的实现

合肥解放军电子工程学院 318 教研室 (230037)
董 晖 姜秋喜 毕大平

对于 DSP 的使用者而言,如何充分发挥 DSP 的性能,在尽可能短的时间完成给定数据的处理,是他们关心的主要问题。要解决这一问题,就需要结合 DSP 器件的特殊结构及工作方式,开发出经过反复优化能充分发挥其性能的专用程序。

一、TMS320C6000 的特殊结构

DSP 器件可以以较高的速度实现特定的算法,这是和它的特殊结构和工作方式密不可分的,以 TI 公司的 C6000 为例,其结构和工作方式有如下特点。

1. 独特的算术单元

硬件乘法器 :在 C6000 器件内有硬件乘法器,其中 C6701 的乘法器可以完成浮点乘法运算,用 C6000 内的乘法器代替通用处理器中的乘法程序可以显著缩短乘法的执行时间。

多功能单元 :C6000 内有 8 个并行的功能单元,这 8 个功能单元最多可以在一个周期内同时执行 8 条指令。同时大多数 DSP 器件支持在同一周期内实现乘加操作,和位翻转寻址,这可以显著提高 FFT 运算速度。但是 C6000 不支持这两种功能,而采用精简指令集 (RISC),指令的条数较少,实现 FFT 算法稍复杂一些。

2. 灵活的总线结构

C6000 采用修正的哈佛总线结构,有一套 256 位程序总线、两套 32 位数据总线和一套 32 位的 DMA 专用总线。由于程序总线和数据总线相互独立,能在取指的同时取操作数,大大缓解了数据瓶颈对系统性能的限制,而其 256 位程序总线一次就可以取出 8 条 32 位的指令。

3. 专用的寻址单元

随着数据量的增大, 数据访问时的地址计算时间也线性增长,如不采取特殊方法,地址计算的时间可能会大于算述操作的时间。因此,C6000 有专用地址计算单元——地址产生器。地址产生器与 ALU 并行工作,不占用 CPU 资源,可以在后台传输数据。

4. 较大的片内存储器

为提高对存储器的访问速度,C6000 系列芯片集成有 1M ~ 7M 位的程序 RAM 和数据 RAM。当程序存放在片内 RAM 时,可以减少指令传输时间,缓解总线压力 ;而内部的数据 RAM 没有访问速度匹配问题和总线竞争问题,可以缓解 DSP 的数据瓶颈。

5. 高效的流水线工作方式

任何指令的处理均能分成几个子操作,每个子操作由不同的单元来完成。对每个单元来说,每一个时钟周期都可以接受一条新指令,这样,在不同单元中可以同时处理多条指令,这种工作方式称为“流水线”工作方式。理想情况下,一条 k 段流水线能在 k + (n - 1) 个周期内处

理 n 条指令，当 n 很大时，可以认为每个周期内执行的最大指令个数为 k。

二、软件流水的概念

从以上的分析可以看出 DSP 器件的特殊结构和流水线工作方式为实现 C6000 高效算法提供了可能，而这种可能转化为现实的关键就是能否实现软件流水提高代码的并行性。DSP 器件一般都是用来对大量数据进行高速处理，这些算法的核心部分大多是一些循环迭代运算，而耗时最多的也正是这些部分。因此要实现高效算法就要尽量减少循环体的执行时间，而为减少循环体的执行时间，就要尽可能实现软件流水。所谓软件流水是用来安排循环指令，使这个循环的多次迭代并行执地的一种技术。这里特别要说明的是，软件流水和 C6000 本身的流水处理是两个不同的概念。前者是对循环代码进行优化提高运算速度的一种技术，后者则是 C6000 的一种工作方式。但这两者又是紧密相关的，实现软件流水的程序能充分发挥流水线工作方式的优点，保持流水线的充满，这时的流水线最有效，运算速度也最快。

图 8.5－1 是一个循环代码的软件流水示意图。图中 A、B、C、D 和 E 表示各次迭代，每一迭代中有 4 条指令，字母后面的数字表示各次迭代的第几条指令，同一行的指令在同一周期内并行执行。假设，由于采用了软件流水技术，使得本例中同一周期内可并行执行 4 条指令（阴影部分）。如果所有指令均为单周期指令，则在软件流水条件下，执行这个循环只需要 8 个时钟周期。图中阴影部分称为循环核，核前面的部分称为循环填充，核后面的部分称为循环排空。对 C6000 而言，一次可以取 8 条的指令构成一个取指包，其内部的 8 个处理单元在理想情况下可允许 8 条指令并行执行，从而形成软件流水。

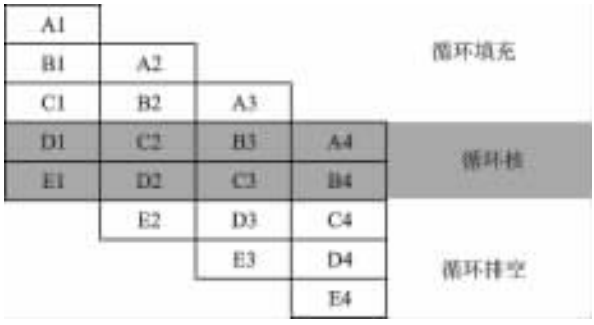


图 8.5－1 软件流水循环

如果不能实现软件流水，也就是说同一迭代内的 4 条指令不能并行执行而只能顺序执行，如图 8.5－2 所示，则需要 20 个时钟周期。同样，对 C6000 而言，最差的情况下，每个取值包内的 8 条指令只能顺序执行，流水线严重阻塞，执行效率很低。

图 8.5－2 非软件流水循环

下面讨论一下实现软件流水的循环执行时间，设 n 为迭代次数，m 为每次迭代的指令条数，k 为可并行执行的最大指令条数，为简单起见，假定每条指令均为单周期指令，且 m 可以被 k 整除，设 c 为执行软件流水的循环所需时钟周期数，则有：

$$c = m \times (n - k + 1) / k + 2 \times (k - 1)$$

设 c' 为执行非软件流水的循环所需时钟周期数,则有： $c' = m \times n$

显然,当 $m \gg k$ 时,有 $c/c' \approx k$,也就是说实现软件流水的循环执行时间是未实现软件流水的循环的 1/k。以上讨论的是理想情况,实际动用中,由于部分指令是多周期的,以及存在存储器冲突等因素很难达到这种理想情况。通过实验分析(实验结果见表 8.5-1 ~ 表 8.5-3),对于 C6000 而言,经充分优化实现软件流水的程序执行时间约为未优化程序执行时间的 1/30,未优化的程序的执行速度仅略快于 Pentium MMX 200,根本不能体现 DSP 的高速特性,这更加说明了实现软件流水的重要性。

表 8.5-1 不同点数定点 FFT 在 C6201 仿真器上的执行时间

FFT 点数		256	1 024	4 096	16 384
手工优化汇编代码执行时间/μs	基 2	23.44	112.15	528.24	2 438.07
	基 4	15.56	73.32	338.7	1 556.04
_O 自动优化 C 代码执行时间/μs	基 2	126.32	572.71	2 572.37	11 430.28
	基 4	248.54	1 212.28	5 730.79	26 443.09
未优化 C 代码执行时间/μs	基 2	764.81	3 644.28	16 941.68	77 256.06
	基 4	449.38	2 150.77	10 061.36	45 935.34

表 8.5-2 不同点数定点 FFT 在 PC 机上的执行时间

FFT 点数		256	1 024	4 096	16 384
Pentium Ⅲ 866 执行时间/ms	基 2	0.088	0.42	2.00	8.40
	基 4	0.055	0.27	1.28	6.10
Celeron 466 执行时间/ms	基 2	0.104	0.79	3.75	19.60
	基 4	0.148	0.50	2.40	12.60
Pentium MMX 200 执行时间/ms	基 2	1.070	5.20	25.40	140.00
	基 4	0.590	2.90	14.60	76.00

表 8.5-3 不同点数基 2 定点 FFT 在 C6701 仿真器及 C6701EVM 上的执行时间

FFT 点数		1 024	2 048	4 096	8 192
C6701 仿真器执行时间/ms	片内	0.169	0.367	0.794	1.711
	片外	0.169	0.367	0.794	1.711
C6701EVM 执行时间/ms	片内	0.169	0.367	0.794	1.711
	片外	2.563	5.549	11.348	25.627

三、优化程序实现软件流水

对 C6000 而言,用户编写的源程序需要经过编译、汇编和联接来生成最终的可执行文件,这一过程是由 C6000 集成开发环境 Code Composer Studio (CCS) 的编译工具中内嵌的壳程序 cl6x 来完成的。为实现软件流水需要对用户程序进行优化,这一优化过程也是依靠它完成的。

C6000 代码开发的流程可以分为三个阶段:产生 C 代码阶段、优化 C 代码阶段和使用线性汇编优化代码阶段。本文重点讨论与优化程序有关的后两个阶段的内容。

1. 优化 C 代码源程序

用户使用 C 语言编写的代码经过 C 编译器编译后生成 C6000 汇编代码,对 C 语言源代码的优化包括两方面的内容:一方面是根据 C6000 的结构特点编写高效 C 代码,另一方面是合理选择编译器选项来实现软件流水。

编写高效 C 代码要注意以下问题。首先,要合理选择数据类型,对定点乘法输入,尽可能用 short 型数据,本文的定点 FFT 程序使用的就是 short 类型的。其次,可以通过调用内联函数(intrinsics)来代替复杂的 C 代码,内联函数可以实现很多 C 语言较难实现的特殊功能,能显著提高运算速度。第三,在可能的情况下,对 C6000 可以使用字访问 2 个 16 位数据,用双字访问 2 个 32 位数据,这可以把数据访问速度提高一倍,本文的定点 FFT 程序中整序运算就采用了这种方法。最后,还可以用向优化器传递循环次数人工干预软件流水,消除冗余循环及展开循环的方法优化 C 代码。

在选择编译器选项时,关键是要选择 - o 选项打开壳程序 cl6x 中的优化器,实现不同级别的程序优化。编译器还有很多选项,有些是控制编译器壳程序执行的,例如, - z 用来控制联接器;有些是设置硬件的,例如, - mv 用来选择芯片类型;有些是控制优化器和解析器的,例如前面提到的 - o 选项。在使用时,必须根据不同情况进行选择。当寻址范围超过 2^{15} 后,要选择 - ml 选项,否则联接时会出错。

CCS 提供的 C 语言编译器的编译效率可达到汇编语言的 70% ~80%,对大多数程序而言,已可满足需要,通过实验也验证了这一点,优化后的程序执行时间约为优化前的 1/6,但使用优化器对程序进行优化对程序本身有诸多限制,尤其是对 FFT 而言,因为基 4 算法过于复杂,无法有效优化,致使优化后的基 4 算法比基 2 算法慢,这更说明了 C 语言优化器的局限性。

2. 使用线性汇编语言优化程序

当对 C 语言的优化不能满足需要时,就必须从 C 代码中抽出影响速度的关键部分,用线性汇编语言来改写这部分代码,并用汇编优化器来优化这些代码。线性汇编语言与 C6000 的汇编语言类似,两者有相同的指令集,但线性汇编语言可以不包含有关指令延迟时隙和寄存器使用的信息,这样做的目的是可以让汇编优化器来决定这些内容,以降低编程难度,缩短软件开发时间。即使如此,使用线性汇编语言优化代码,仍要比优化 C 代码花费更多的时间,所以要尽可能的充分优化 C 代码,减少使用线性汇编优化代码的工作量。汇编优化器对线性汇编代码进行优化,该优化器主要完成以下工作:一是用 C6000 指令集的并行性调度指令提高效率,二是在确保指令符合 C6000 延迟时隙要求的前提下给代码分配寄存器。

在写线性汇编代码的时候要注意以下问题,首先,线性汇编代码必须包含必要的伪指令,如". cproc"和". endproc"来限定线性汇编代码,另外还有一些伪指令,可以向优化器提供优化所需的信息,以便更好的优化程序。其次,线性汇编代码必须遵守一定的语法规则。第三,在进一步优化时可以指定处理单元,避免寄存器冲突。一般情况下,线性汇编代码的效率可达到汇编代码的 90% ~100%,在绝大多数情况下,已可满足用户需要,如还不能满足需要,就只能对汇编代码进行手动优化,这将花费更长的时间,而且有相当大的难度。本文的 FFT 程序就对汇编代码做了手动优化,把循环变量放在了内层循环中,使用了字访问,最终代码高度并行,充分实现了软件流水,其执行时间约为优化后的 C 代码的执行时间的 1/5,约为 Pentium Ⅲ

866 执行时间的 1/4。

3. 合理安排存储空间提高运算速度

使用片内存储空间也是 DSP 具有高速性能的一个重要原因,虽然最后的系统可以包括片外存储空间,但使用这些存储空间会显著降低运算速度。如表 8.5-3 所列,采用相同算法在 C6701 评估板(EVM)进行测试,使用片外存储空间的执行时间约为使用片内存储空间的 16 倍,因此,为了提高运算速度,应该尽可能的把数据安排在片内存储空间,对 C62/701 而言,其内部数据存储空间的长度为 64k(Bytes),以本文的 FFT 算法为例,其数据类型为 16bit 的定点数,采用查表法要预先存储旋转因子表,设 FFT 点数为 N,占用的存储空间长度为 L,则有:

$$L = 6 \times N \text{ (Bytes)}$$

显然,利用片内存储空间最多只能完成 8 k 点的 FFT。从试验结果还可看出,仿真器的执行时间可以认为是真实器件在片内存储空间的执行时间,但不能反映出片外和片内存储空间执行时间的差别。

本文讨论了基于 C6000 的高效程序的开发问题,并以数字信号处理中最常用的 FFT 算法为例,进行了深入的研究,通过大量的对比实验,证明以上方法是有效的,基本可以满足实际需要。本文给出的实验结果,对其他算法的开发也有一定的参考价值。

参 考 文 献

1 任丽香,马淑芬,李方慧. TMS320C6000 系列 DSPs 的原理与应用. 北京:电子工业出版社,2000

2 TMS320C6x Assembly Language Tools User's Guide. Texas Instruments Incorporated,1999

3 TMS320C62x/C67x Programmer's Guide. Texas Instruments Incorporated,1999

4 TMS320C6000 Optimizing C Complier User's Guide. Texas Instruments incorporated,1999

选自《微处理机》季刊,2002 年第 2 期

8.6 TMS320F240 串行外设接口及其应用

武汉大学电气工程学院 (430072) 郭必胜 陈昌旺

TMS320F240 是美国德州仪器 (TI) 公司推出的一种具有高速运算性能的数字信号处理芯片。它采用哈佛结构,将程序和数据存于不同的存储空间,即程序存储器和数据存储器是 2 个相互独立的存储器,每个存储器独立编址,独立访问。与 2 个存储器相对应的是系统设置了程序总线和数据总线,从而使数据的吞吐率提高了 1 倍。同时采用深度流水线操作,使取指、译码和操作可同时进行,减少了指令执行时间,从而增强了处理器的处理能力。为便于和各种外围设备进行通信,TMS320F240 提供了 2 种串行 I/O 端口,即串行通信接口 (SCI) 和串行外设接口 (SPI)。其中 SPI 接口是标准的同步串行接口,它允许处理器与各种外围设备以串行方式 (8 位数据同时、同步地被发送和接收) 通信,数据的传输只需 3 条线 (时钟、发送和接收)。本文重点介绍串行外设接口在多机通信中的应用。

一、SPI 结构与工作原理

1. SPI 引脚说明

SPI 模块通过 4 个引脚与外围设备进行接口,其功能如下:

(1) SPISIMO:串行外设从输入/主输出数据引脚。

(2) SPISOMI:串行外设从输出/主输入数据引脚。

(3) SPICLK:串行外设接口时钟引脚,为串行外设接口发送和接收数据提供同步时钟信号。若为主模式,SPI 时钟由 SPI 产生,并通过 SPICLK 引脚输出;若为从模式,SPI 时钟通过 SPICLK 引脚从外部网络主设备输入,其频率最大不超过系统时钟频率的 1/8。

(4) SPISTE:串行外设接口选通引脚。在主模式下,该引脚通常用作 I/O 引脚,而与 SPIPC1.5 位的设置无关。在从模式下,当 SPIPC1.5 设置为 0 时,该引脚则作为通用 I/O 引脚;当 SPIPC1.5 设置为 1 时,则作为从模块的选通引脚。若为高电平则使从模块的移位寄存器停止工作且输出引脚呈高阻状态而不进行发送;若为低电平则使能从模块的发送功能。

2. SPI 控制寄存器

SPI 模块内有 10 个寄存器,各寄存器的地址如图 8.6-1 所示。

寄存器	SPICCR	SPISTS	SPIBBR	SPIEMU	SPIBUF	SPIDAT	SPIPC1	SPIPC2	SPIPRI	SPICTL
地址	7040H	7042H	7044H	7046H	7047H	7049H	704DH	704EH	704FH	7041H

图 8.6-1 SPI 控制寄存器地址

图 8.6-1 中各寄存器用于控制 SPI 模块的操作。

(1) SPI 配置控制寄存器 (SPICCR)。主要用于控制 SPI 的软件复位、SPI 时钟极性选择以及 SPI 字符长度。

⑵ SPI 操作控制寄存器（SPICTL）。主要用于控制数据发送、SPI 产生中断、SPI 时钟相位选择、主/从操作模式选择以及数据发送使能。

⑶ SPI 状态寄存器（SPISTS）。包含接收缓冲器的 2 个状态位：接收器超限标志位（RECEIVER OVERRUN）和串行外设中断标志位（SPI INT FLAG）。

⑷ SPI 波特率寄存器（SPIBBR）。主要用于选择串行外设位传送速率（共有 128 种数据传送速率可供串行外设选择）。

⑸ SPI 仿真缓冲寄存器（SPIEMU）。包含接收的数据，并支持仿真。

⑹ SPI 串行缓冲寄存器（SPIBUF）。包含接收的数据和从 CPU 读出的数据，读 SPIBUF 会清除串行外设中断标志位（SPISTS.6）。

⑺ SPI 串行数据寄存器（SPIDAT）。包含 SPI 待发送的数据，作为发送/接收移位寄存器使用，写入 SPIDAT 的数据在后续 SPICLK 周期被移出。

⑻ SPI 端口控制寄存器 1（SPIPC1）。主要用于选择 SPISTE 和 SPICLK 引脚功能。

⑼ SPI 端口控制寄存器 2（SPIPC2）。主要用于选择 SPISIMO 和 SPISOMI 引脚功能。

⑽ SPI 优先级控制寄存器（SPIPRI）。主要用于设定中断优先级并在控制程序暂停期间决定 XDS 仿真器上的 SPI 操作。

3. SPI 模块的主/从模式

SPI 模块工作于主模式或从模式，由串行外设控制寄存器决定。若 SPICTL.2 设置为 1 则为主模式，否则为从模式。

⑴ 主模式。在主模式下，数据从 SPISIMO 引脚移出，从 SPISOMI 引脚移入。发送数据时，在同步时钟 SPICLK 的节拍下，从 CPU 写到 SPIDAT 的数据启动 SPISIMO 引脚上的发送；接收数据时，接收到的数据通过 SPISOMI 引脚移入 SPIDAT，接收完 1 个数据块后，再复制到 SPIBUF。

⑵ 从模式。在从模式下，数据从 SPISOMI 引脚移出，从 SPISIMO 引脚移入。发送数据时，SPI 在从网络主设备接收到的 SPICLK 节拍下，从 CPU 写到 SPIDAT 的数据发送到 SPISOMI 引脚上；接收数据时，SPI 要等待网络主设备发出 SPICLK 信号，然后将数据从 SPISIMO 引脚移入 SPIDAT，接收完 1 个数据块后，再复制到 SPIBUF。

无论主模式还是从模式，当选定的位数发送完后，数据的最高位先被传送到 SPIBUF，供 CPU 读出。数据在 SPIBUF 中是右对齐，而写入 SPIDAT 则是左对。

4. 时钟模式

SPI 模式通过控制时钟极性（CLOCK POLARITY）和时钟相位（CLOCK PHASE）可获得 4 种不同的时钟模式。SPI 模块的时钟模式和时序分别如表 8.6－1 所列和图 8.6－2 所示。

表 8.6－1 SPI 的 4 种时钟模式

时 序	时钟极性	时钟相位	说 明
无延时上升沿	0	0	上升沿发送，下降沿接收
有延时上升沿	0	1	上升沿的前半个周期发送，下降沿接收
无延时下降沿	1	0	下降沿发送，上升沿接收
有延时下降沿	1	1	下降沿的前半个周期发送，下降沿接收

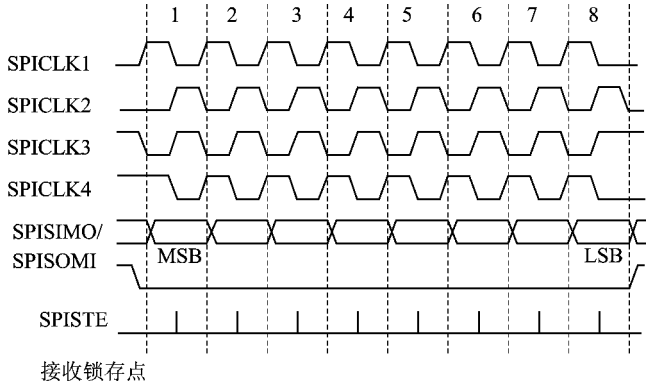


图 8.6－2 SPICLK 时序图

二、应 用

与异步串行通信一样，SPI 总线上也可以挂接多台设备，如图 8.6－3 所示。SPI 多机通信与 SCI 多机通信不同，SPI 通过 SPICLK 引脚以实现主机和从机同步，因此其数据交换不像 SCI 那样需要起始、停止、数据/地址等用于同步的格式位，而是直接将传送的 8 位数据信息送到 SPIDAT 中。因此采用何种方式以区分地址信息和数据信息就成为进行数据传输的关键。为此在图 8.6－3 中增加了一地址片选线，在主机与某个从机进行通信时，先在 I/O 端口发从机的地址信息，经地址译码将对应从机的选通引脚 SPISTE 激活，而其余的从机则被译码电路封锁，以保证主机每次只与总线上的一个从机进行数据通信。

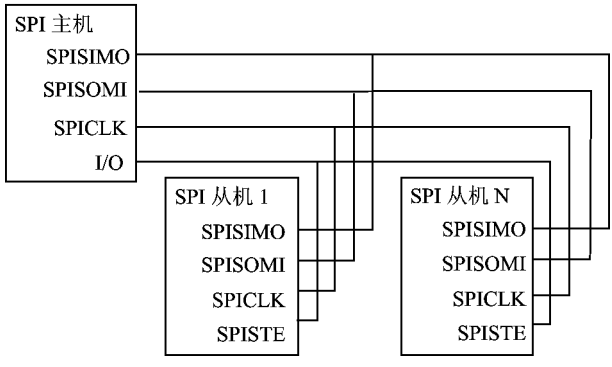


图 8.6－3 SPI 的多机通信接口电路

1. 软件协议

主机先发送数据个数，再发送数据，最后发送累加校验和。从机根据接收到的“校验和”来判断已收到的数据是否正确，如正确则向主机回发 0FH；若不正确则向主机回发 F0H。主机根据回发信号决定是否重发数据。

主机和从机的程序流程图如图 8.6－4 和图 8.6－5 所示。

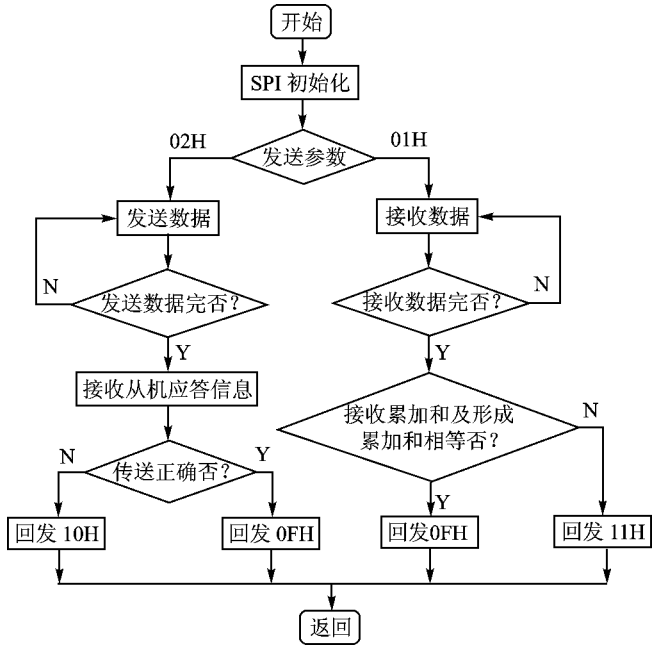


图 8.6－4 主机流程图

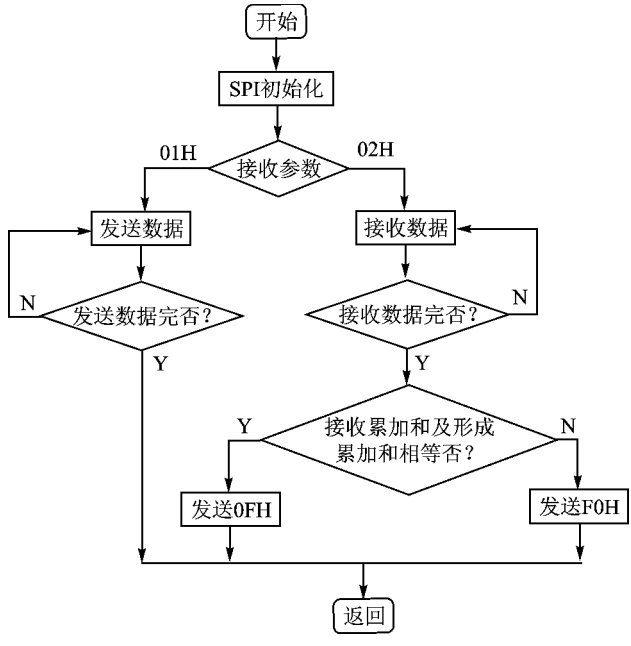


图 8.6－5 从机流程图

2. 参数说明

01H 表示主机要求从机发送数据,02H 表示主机要求从机接收数据。待发送的数据首地址为 DATAOUT,接收的数据存入的首地址为 DATAIN,发送/接收的数据个数存放地址为 LENGTH。

下面给出从机的部分程序。

```
……
LAR     AR1,#LENGTH
LAR     AR2,#DATAOUT
LAR     AR3,#DATAIN
……
MAR     *,AR3
LACL    SPIBUF                  接收参数
B       LOOP3                   参数接收完否
LACL    AR5
XOR     #01H
BCND    RXDATA,NEQ              ;ACC! =0,转接收子程序
B       TXDATA                  否则,转发送子程序
TXDATA:
SPLK    #000BH,SPITL            使能发送和中断,工作于从模式
SPLK    #07H,SPICCR             准备发送数据
MAR     *,AR1
LACL    *
SACL    SPIDAT                  发送数据个数
B       LOOP1                   等待发送完
LACL    AR5
MAR     *,AR4
SACL    *                       形成累加和
LOOP1:
MAR     *,AR2
SPLK    #07H,SPICCR             准备发送下一数据
LACL    * +
SACL    SPIDAT                  发送数据
B       LOOP3
LACL    AR5
MAR     *,AR4
ADD     *                       形成累加和
SACL    *,AR1
SBRK    #01H
BANZ    LOOP1                   数据未发送完,继续发送
SACL    SPIDAT                  发送累加和
B       LOOP3
SPLK    #0009H,SPICTL           禁止发送,工作于从模式
```

```
   B      RETURN                    恢复
RXDATA：
   MAR    *,AR3
   LACL   SPIBUF                    接收数据个数
   SACL   *,AR4
   B      LOOP3                     等待接收完
   LACL   AR5
   MAR    *,AR4
   SACL   *                         形成累加和
   LAR    AR1,*
LOOP2：
   LACL   SPIBUF                    接收数据
   B      LOOP3
   LACL   AR5
   MAR    *,AR3
   SACL   *+,AR4
   ADD    *,AR1                     形成累加和
   SBRK   #01H
   BANZ   LOOP2                     数据未接收完,继续接收
   LACL   SPIBUF
   B      LOOP3
   LACL   AR5                       接收累加和
   MAR    *,AR4
   SUB    *
   MAR    *,AR3
   SPLK   #000BH,SPICTL             使能发送,工作于从模式
   BCND   TX2,NEQ
TX1：
   SACL   SPIDAT,#0FH               向主机回发 0FH
   B      TT
TX2：
   SACL   SPIDAT,#F0H               向主机回发 F0H
TT：
   B      LOOP3                     等待发送完
   SPLK   #0009H,SPICTL             禁止发送,工作于从模式
   B      RETURN
LOOP3：
   SACL   AR5
   LACL   SPISTS
   AND    #040H
   XOR    #040H
   BCND   LOOP3,NEQ
RETURN：
   ……
```

综上所述,同步串行外设接口可以在短距离内进行主机和从机之间的数据传输,具有连接电路简单、使用方便等特点,从而为实现主机与从机以及各种外围设备之间的通信提供了一种简单易行的方案。

参 考 文 献

1 何立民. MCS－51 单片机应用系统设计. 北京 北京航空航天大学出版社,1990
2 章云,谢莉萍,熊红艳. DSP 控制器及其应用. 北京 机械工业出版社,2001
3 武汉力源电子股份有限公司. TMS320C240/F240 数据信号处理手册. 1998

选自《微型机与应用》月刊,2002 年第 11 期

8.7 基于 DSP 的 Modem 及其驱动程序的设计与实现

西安理工大学 (710048) 李长河 张 熙

一、引 言

近年来,随着 PC 市场的不断增长、软件应用范围的扩展以及互联网的逐步普及,PDA 正成为人们交流和信息沟通的简单又便利的工具之一,其发展相当迅速。为了能更加方便地进行移动办公和简单的商务交流活动,PDA 应该可以通过有线或无线的方式接入 Internet,帮助人们进行网上操作和随时随地获取有用信息。

PDA 接入 Internet 的方式从总的方式来说分为有线和无线两种。有线接入的方式主要是使用调制解调器通过电话线来上网,调制解调器有内置式和外置式,外置式接入设备一般通过串口和手持终端相连接,内置式一般采用 Modem 卡的方式来上网。

modem 的设计普遍是采用专用的 modem 芯片,数据流的编解码以及协议转化算法固化在 modem 芯片内部,采用这种方法后,modem 不能灵活地支持新推出的协议,升级困难。而且对 PDA 来讲,modem 的功耗和体积是一个很重要的方面,因此对 modem 的设计不能采用传统的方案。

本文要介绍的方案是采用 Analog Device 公司的 ADSP2187L 作为处理核心,编码和解码由专用芯片完成,协议转化则由 DSP 用软件完成,这样不仅使得系统的灵活性大大加强,而且该 DSP 采用一种称作 IDMA 的方式来运行程序,这样还可以省去 DSP 的程序存储器,降低系统设计成本。

二、系统原理

1. 硬件原理

modem 的功能框图如图 8.7－1 所示。

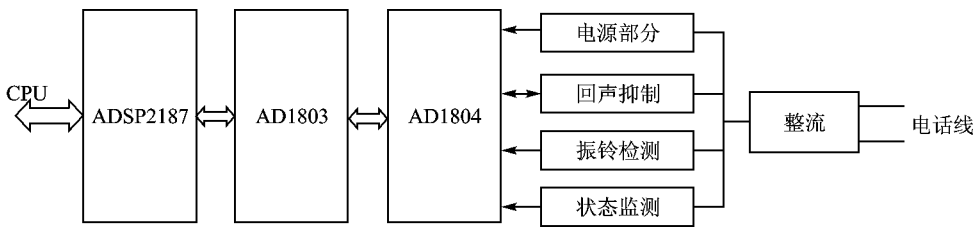


图 8.7－1 modem 功能框图

本文的编码和解码采用 Analog Device 公司的 AD1803 和 AD1804 芯片组。其中 AD1804 完成的是信号的转换以及对电话线上的状态进行采集,并将数据传给 AD1803。AD1804 部分电路包括电源部分、回声抑制部分、振铃检测部分以及状态检测部分。电源部分从电话线上获

得稳定的电压,供 AD1804 内部的电路正常工作;回声抑制部分使得通信可以以全双工方式进行,是通信中极为重要的部分;振铃检测部分完成对电话线上的振铃信号进行检测,并送给 AD1803;状态检测部分完成对电话线上的电压进行监测,使得 AD1804 能够正常工作。AD1803 从 AD1804 接收到数据后会对数据进行一些简单处理并将数据以串行方式传给 DSP,DSP 对从 AD1803 传来的数据进行处理并通知 CPU 来读取。值得注意的是,由于 AD1804 部分电路的电源是从电话线上获得的,因此,AD1803 和 AD1804 之间的连接要通过隔离电容隔离开。

2. DSP 和 CPU 介绍

本文所采用的 DSP 是 Analog Device 公司的 ADSP2187,该 DSP 的主要性能指标如下。

- 每条指令执行周期为 19 ns。
- 三总线结构。在每个指令周期内可完成两次存取操作。
- 支持节电模式。从节电模式转化到正常模式需 400 个机器周期。
- 160 kB 的内部 RAM。分为 32 k Words 的程序存储区 (PM - Program Memory RAM,24 位) 和 32k Words 的数据存储区 (DM - Data Memory RAM,16 位),PM 区既可存储数据又可存储程序。
- 两个独立的数据地址发生器,在执行循环、条件指令时实现零开销。
- 16 位的内部 DMA 口。可快速访问内部存储器区 (模式可选择)。
- 4 MB 的存储区接口,可存储数据表和程序的 OVERLAY 区。
- 可编程的 Wait State。
- 可通过外部 8 位的存储区 (如 EPROM) 或内部的 DMA 口自动引导片内 PM。
- 6 个外部中断。
- 13 个可编程的标志管脚。提供了方便的系统信号。

ADSP2187 的一个显著特点是外部的 CPU 可以通过 DSP 的 IDMA 端口 (Internal Memory DMA Port) 来访问 DSP 内部的 DM 和 PM,而且 DSP 的程序也可由 CPU 通过 IDMA 口下载到 DSP,这样程序就可以存放在 PDA 的 PLASH 中,需要对 modem 进行升级时,只要通过 PDA 将程序代码更新即可。

本文中的 PDA 使用的 CPU 是 Motorola 的 MC68EZ328,目前国内大部分 PDA 所采用的都是该 CPU。该 CPU 具有强大而灵活的总线接口,能够适合 ADSP2187L 的要求。该 CPU 的性能指标如下。

- 时钟周期 60 ns,工作电压 3.3 V。
- 8 个片选信号,每个片选信号的地址空间可由程序设置。
- 支持三种工作模式:正常模式、突发模式、睡眠模式。
- 18 个边沿和电平中断,分为 7 个不同等级的中断。
- 54 个 I/O 口,大部分都是复用的,可以对相应的寄存器进行设置来改变 I/O 口的用途。
- 脉宽调制器,可以用来产生声音。
- 通用定时器具有 60 ns 的分辨率,最长定时时间为 524 s。
- 通用异步收发器支持的最高的波特率为 115 200 bps,可以无缝支持红外通信。
- 支持单色 LCD 显示屏,支持 4 位、2 位和 1 位的数据线接口,最大尺寸 640 × 250 像素,支持 4 级或 16 级灰度,有硬光标。

· 实时时钟模块:提供采样定时中断,每秒一次的中断和每天一次的中断,2 s 的看门狗定时器。

· DRMA 控制模块:支持 8 位和 16 位宽度 DRAM,可程控的刷新周期,对 LCD、DMA 存取支持快速页面模式和 EDO 模式,最高支持到 4 M×16 的容量。

3. DSP 与 CPU 之间的接口设计

ADSP2187 的 IDMA 口由以下几部分组成:16 位的地址/数据复用总线(IAD[15..0])、片选信号(IS)、地址锁存信号(IAL)、读信号(IRD)、写信号(IWR)、响应信号(IACK)。MC68EZ328 负责所有数据传输的初始化工作。ADSP2187 的存储器地址装载在 IDMA 地址寄存器中。IDMA 地址寄存器包括 14 位的内存地址和 1 位传输类型标志位。该标志位用于区分所传输的是程序代码还是数据。CPU 不论向 DSP 下载数据还是程序之前,都要先执行一个地址锁存步骤,过程如下:先由微处理器发出 IAL 和 IS 信号,然后在 IAD 上传输控制字(该控制字指出了操作起始地址和的存储类型信号),锁存完成后,由 CPU 发出读信号时,DSP 会从起始地址中送出数据到 IAD 数据线上,当 CPU 发出写信号时,DSP 会将数据写入起始地址所指向的地址空间。

由于 DSP 的 IDMA 口的数据线要用来传送地址和数据,而 MC68EZ328 的地址线和数据线是分开的,因此要设计其接口电路。接口电路如图 8.7-2 所示。其中 OE 是 MC68EZ328 的读信号,UWE 是写信号,A1 是地址线,CSA1 是片选(这里我们给 CSA1 分配的地址范围是 0x300000 - 0x3ffff)。在地址锁存时,我们使用的地址是 0x300002,此时 A1 为高电平,因此输入到 DSP 的 IAL 为高电平,IS 为低,而 IWR 则为高,满足地址锁存时序;当向 DSP 内写入数据时,我们使用的地址是 0x300000,此时 A1 为低电平,输入到 DSP 的 IAL 为低,IWR 为低,IS 为低,满足写操作时序;从 DSP 读数据时,我们使用的地址是 0x300000,此时 A1 为低,输入到 DSP 的 IAL 为低,IRD 为低,IS 为低,满足读数据时序。

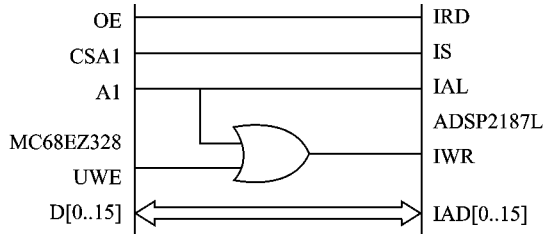


图 8.7-2 CPU 和 DSP 接口电路

当向 DSP 写入数据时,程序如下:

```
/* 锁存 */
*address_of_control_register = value;
/* 写数据 */
*address_for_data = value;
```

当从 DSP 读数据时,程序如下:

```
/* 锁存 */
*address_of_control_register = value;
/* 读数据 */
```

value = * address_for_data；

4. 软件设计

(1) DSP 芯片的启动

由于 DSP 是通过 IDMA 口来传输程序和数据的，因此其运行过程是在 CPU 的控制下完成的。ADSP2187 的程序代码是以 24 位为一个字来存储的，而 DSP 和 CPU 之间的数据线是 16 位的，因此传送一个字的程序代码需要两个写周期，先传送 16 位，再传送 8 位。

另一方面由于程序代码的长度超过了 DSP 内部 RAM 的大小，因此将 DSP 需要执行的程序代码又分为几个部分，我们称之为页。页是一段程序代码，主要分为 Startup 页、Dial 页、V.32 页、V.34 页、V.8 页、info 页等。

启动时 CPU 会将 Startup 页下载到 DSP 内部，下载完成后，DSP 会完成内部的初始化，并通知 CPU 下载 Dial 页，CPU 接到通知后，会将 Dial 页下载，下载完成后，CPU 就可以通过 DSP 拨号和接收数据了，在接收和发送数据时还可能用到其他页，如图 8.7－3 所示。

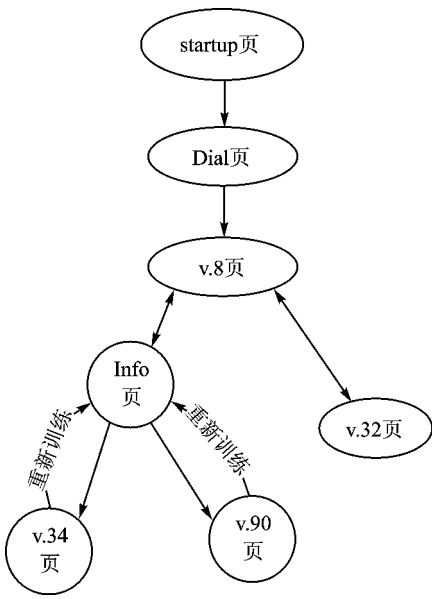


图 8.7－3 页面下载顺序

(2) 事件处理

Modem 和 CPU 之间的通信是通过事件机制来完成的，如下载页面事件，接收缓冲区满事件，发送缓冲区空事件等。各种不同的事件都有各自的代码。DSP 内部有一个区域用来存储 DSP 产生的事件代码以及总共产生的事件个数（我们称之为事件计数器），当 DSP 检测到产生了一个事件时，就会在事件区域中写入事件代码并将事件个数加 1，同时 DSP 会在它的一个 I/O 口上产生一个脉冲信号，这个信号可以作为 CPU 的中断输入信号。在 CPU 端也有自己的事件计数器，当有中断来到，CPU 先读入 DSP 的事件计数器的值，然后和自己的事件计数器相比，如果大于自己的计数器，则 CPU 会从 DSP 的事件区域中读入事件代码，并根据代码执行相应的操作，CPU 每处理完一个事件后，就将自己的事件计数器加 1，当两边的计数器的值相等时，说明 CPU 已经处理完 DSP 的事件。

CPU 侧的软件是在 PPSM (Personal Portable System Manager) 下编制的，在 PPSM 中，每个不同的程序又称为任务，不同任务之间的切换是通过消息机制来实现的，当需要进行任务切换时，当前任务就会发出一条消息给要切换到的任务，该消息都放在消息队列当中，发送完消息后，就可以切换到另一个任务中去了。在此我们将 DSP 产生的中断信号接至 CPU 的 IRQ2 中断引脚，定义为边沿中断。由于 CPU 侧的事件处理程序比较长，如果直接放在 CPU 的中断处理程序中，就会丢失部分中断申请信号。故采用了 PPSM 的消息机制，因此，在 IRQ2 中断处理程序中，只是发送一条消息给 PPSM，在主程序中对 PPSM 内部消息队列进行查询，查询到有发出 IRQ2 中断消息后，再执行事件处理程序。采用这样的机制后，CPU 可以捕捉到 DSP 的所有中断申请信号，不会丢失中断。其程序处理的整体流程如图 8.7－4 所示。事件处理的流程如

图 8.7－5 所示。

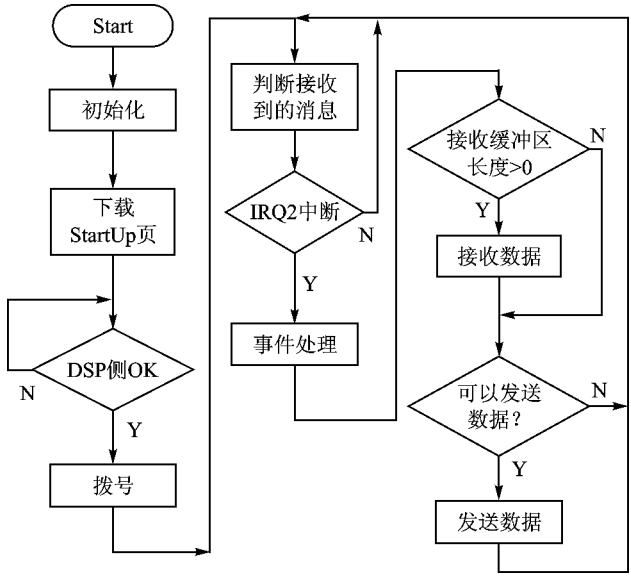


图 8.7－4 CPU 侧程序总体流程

⑶ 发送接收数据

在 DSP 中设有 8 个发送缓冲区和 8 个接收缓冲区，每当一个发送缓冲区空或每一个接收缓冲区满时，就会产生一个事件，并对应不同的事件代码，CPU 接收到 IRQ2 中断消息后，先读入事件代码，如果事件代码对应的是发送缓冲区空事件，此时 CPU 可以发送数据到 DSP；如果事件代码对应的是接收缓冲区满事件，则 CPU 可以从接收缓冲区中读入数据。

对 CPU 来讲，数据和 AT 命令都是相同的方式发往 DSP 的，但 AT 命令和数据是以一个标志来区分的，当该标志为 1 时，是 AT 命令，为 0 时是数据；DSP 接收到数据后会对接收到的数据进行判别，如果是 AT 命令，DSP 就会执行相应的动作；如果是数据，DSP 就会将数据发送出去。

当两个 modem 通过 AT 命令完成拨号并成功建立连接后，他们之间就可以进行发送和接收数据了。

综上所述，可得如下结论。

⑴ 按照以上方法设计的硬件和软件系统可以正常工作。

⑵ 可以支持 V.90、V.34 等通信协议，通信速率最高可达到 56 Kbps。

⑶ 支持 MNP5 数据压缩协议或基于 V.42 的 MNP2_4 数据压缩标准。

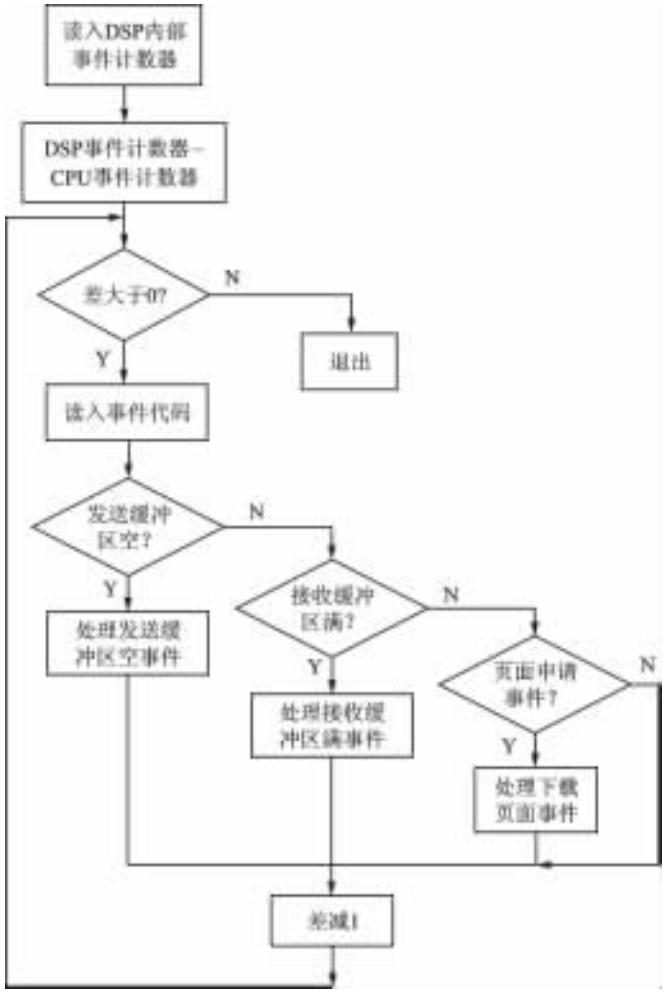


图 8.7－5 **事件处理流程**

参 考 文 献

1 ADSP2187 datasheet REV. 0. Analog Device, Inc., 1998

2 ADSP—2100 FAMILY USER S MANUAL. Analog Device, Inc., 1999

3 V. 90 modem software on ADSP 218x family User s Guide, Version 5. 3. Analog Device, Inc., Feb. 6th, 1999

4 **陈坚,孙志月. 通信编程技术. 西安:西安电子科技大学出版社,**1999

选自《微处理机》季刊,2002 **年第** 2 **期**

8.8 W3100 在 DSP 系统以太网接口中的应用

北京理工大学电子工程系信息系统实验室 (100081) 徐元军

一、W3100 功能简介

W3100 是韩国 Wiznet 公司生产的 Internet 接入芯片,它包含了 TCP、IP Ver. 4、UDP、ICMP、ARP、DHCP 等 Internet 协议和 DLC、MAC 以太网协议。其功能框图如图 8.8-1 所示。W3100 芯片由 4 部分组成:微控器接口单元、网络协议引擎、双口 RAM 及网络物理层介质开关接口 MII (Media Intependent Interface) 单元。W3100 支持全双工 4~5 Mbps 的数据通信,并可同时支持 4 个独立的网络连接;提供 24 KB 的数据缓冲双口 RAM;采用 0.35 μm 的 CMOS 工艺,64 引脚 LQFP 封装;采用 3.3 V 电源电压,其 I/O 接口兼容 5 V 的数字逻辑电平,可非常方便地与单片机和 DSP 接口。

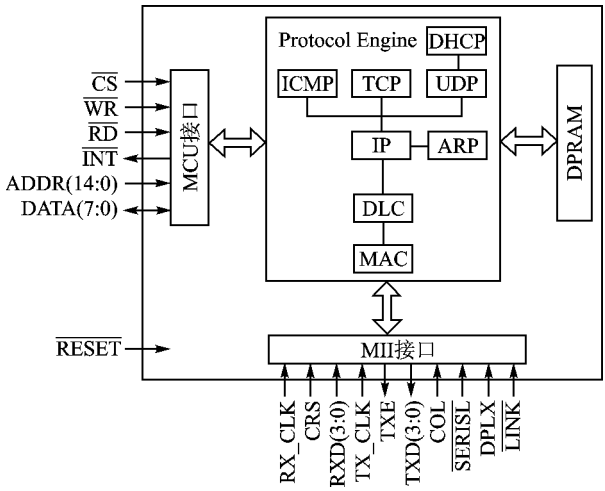


图 8.8-1 W3100 功能框图

W3100 的引脚分布如图 8.8-2 所示。它通过数据线 D[0]~D[7]、地址线 A[0]~A[14] 和写信号 $\overline{WR}$、读信号 $\overline{RD}$、中断信号 $\overline{INT}$、片选线 $\overline{CS}$ 与微控器接口,电源 V_{CC} 为 3.3 V,/REST 是芯片的外部复位信号。其余的接口线是与标准 MII 物理层芯片 (如 RTL8210) 的接口线,在此不再赘述,可参考文献 [2]。

W3100 集成了功能强大的网络接入协议[1],设计者完全可以像在局域网中配置 IP 地址一样简单地配置设计的系统,通过灵活创建和选择 TCP 或 UDP 套接字 (socket) 来完成网上的数据交换。以上工作都是通过设置 W3100 内部的控制寄存器来完成的。微控器通过执行外部 RAM 的操作时序来写或读这些寄存器,发送和接收的数据均保存在 W3100 内部的双口 RAM 中。W3100 的存储空间分配如图 8.8-3 所示。控制寄存器位于地址的低 512 字节,其中 0~

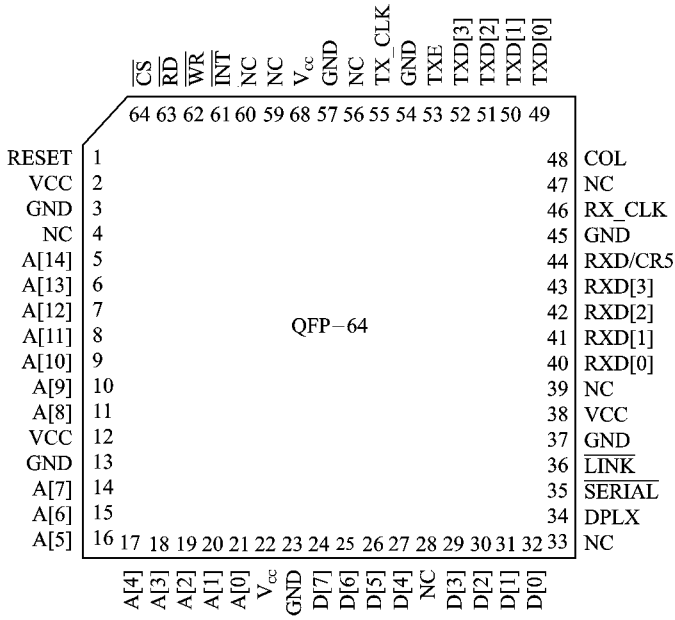


图 8.8-2 W3100 的引脚配置

255 字节是控制寄存器空间,256 ~ 512 字节是控制寄存器的影子存储器,其内容与控制寄存器相同。地址空间 8192 ~ 16383 是 8 KB 的发送数据缓冲器区,地址空间 16384 ~ 32762 是 16 KB 的接收数据缓冲区。其中,低 512 字节内的控制寄存器分为 3 类:① 与网络命令、状态和中断相关的寄存器。② 与 IP 地址、网关地址、子网掩码、物理地址、超时值相关的寄存器。③ 与网络连接的通道相关的寄存器。本文重点介绍常用的几个重要的控制寄存器:通道控制寄存器、通道中断状态寄存器、中断寄存器、中断屏蔽寄存器、系统寄存器。

图 8.8-3 W3100 的存储空间分配

1. 通道控制寄存器

因 W3100 同时支持 4 个连接,所以分别有 4 个通道控制寄存器。除 D0、D7 位在通道 0 控制寄存器中有定义外,其余几位的定义完全一样。D0、D7 位在通道控制寄存器 1、2、3 中均作为保留位。其各位定义如图 8.8-4 所示。

7	6	5	4	3	2	1	0
S/W Reset	Recv	Send	Close	Listen	Connect	Sock_Init	Sys_Init

图 8.8-4 通道控制寄存器各位定义

Sys_Init (仅通道 0 有定义):系统初始化位。用于设置系统的 IP 地址、网关、子网掩码、物理地址。

Sock_Init:通道初始化位。当选择了相应的 Internet 协议后,用此命令进行初始化。

Connect:连接命令。用于以客户 (client) 模式连接到服务器。

Listen:监听模式。用于以服务器模块监听下面客户的连接。

Close:关闭通道及其连接。

Send:启动数据发送位。

Recv:启动数据接收位。

S/W Reset (仅通道 0 有定义):用于初始化芯片内部的复位设定值。

2. 通道中断状态寄存器

4 个通道各对应一个中断状态寄存器。这些寄存器分别对应各通道的控制寄存器。当微控器利用通道控制寄存器发出相应的命令后,中断状态寄存器将返回其执行的结果。D0 位除在通道 0 中有定义外,在其余的 3 个通道中是保留位。其各位定义如图 8.8-5 所示。

7	6	5	4	3	2	1	0
	Recv_OK	Send_OK	Timeout	Closed	Established	SInit_OK	Init_OK

图 8.8-5 通道中断状态寄存器各位定义

Init_OK:系统初始化成功标志位 (仅通道 0 有定义)。

SInit_OK:通道初始化成功标志位。

Established:通道连接已建立标志位。

Closed:通道已关闭标志位。

Timeout:通道连接建立超时标志位。

Send_OK:数据成功发送标志位。

Recv_OK:数据成功接收标志位。

3. 中断寄存器

中断寄存器用于指示当前发生中断的具体事件。当某事件发生时,如果它未被中断屏蔽寄存器屏蔽掉,W3100 将通过/INT 引脚通知微控器,微控器再查询中断寄存器就可以知道具体发生了什么事件。其各位的定义如图 8.8-6 所示。

7	6	5	4	3	2	1	0
C3R	C2R	C1R	C0R	C3	C2	C1	C0

图 8.8-6 中断寄存器各位的定义

其中:C0、C1、C2、C3 分别指示通道 0、1、2、3 引发了中断,C0R、C1R、C2R、C3R 分别标明通道 0、1、2、3 接收到了数据。

4. 中断屏蔽寄存器

与中断寄存器的各位一一对应,用于屏蔽 (置 0) 或允许 (置 1) 相应的中断。其各位的定义如图 8.8-7 所示。

7	6	5	4	3	2	1	0
IM_C3R	IM_C2R	IM_C1R	IM_C0R	IM_C3	IM_C2	IM_C1	IM_C0

图 8.8－7 中断屏蔽寄存器各位定义

系统寄存器包含网关寄存器、子网掩码寄存器、MAC 地址（物理地址）寄存器、IP 地址寄存器、超时值寄存器。分别用于设置系统相应的连接参数。

二、W3100 与 TMS320C6201 的接口设计

TMS320C6201 是 TI 公司 C6000 并行处理 DSP 系列中的定点型产品。它在片内集成了 EMIF（扩展存储器接口），通过设置 EMIF 的控制寄存器，可以方便地与 SDRAM（同步动态随机存储器）、ASRAM（异步静态随机存储器）和 ROM 实现无缝连接。EMIF 采用独立的地址总线和数据总线，且针对不同速度的存储器，EMIF 可以单独地设定读写周期的总线延迟时间，以适应大多数的外接存储器。EMIF 提供 CE0、CE1、CE2、CE3 4 个独立的存储空间。尽管其数据总线的宽度是 32 位，但是通过对应的字节使能控制位 BE0、BE1、BE2 和 BE3，可以方便地实现单周期按字节、半字（16 bit）、字（32 bit）进行访问。作者利用 EMIF 的异步接口，成功地完成了对 W3100 的控制。其原理图如图 8.8－8 所示。

EMIF 提供了 W3100 所需的全部信号线，将 W3100 安排在 TMS320C6201 的 CE3 地址空间。W3100 的中断输出信号接到 DSP 的外中断输入引脚上（即图 8.8－8 中的网络标号 DSP_INT）。DSP 可以时钟周期为单位，为每一个存储空间设置读写操作的建立时间、触发时间、保持时间。DSP 的异步写时序如图 8.8－9 所示。考虑到 DSP 读写的速度相对较快，系统在初始化中设置 EMIF 寄存器组时，将建立时间和触发时间设为 2 个周期，保持时间设为 3 个周期。这样 DSP 在 120 MHz 的主频下，读写操作的时间余量是充足的。

三、DSP 以太网控制程序设计

TMS320C6201 的控制程序是在 TI 公司的 CCS（Code Composer Studio）集成 DSP 开发环境下，采用 C 语言编写。对 DSP 而言，由于采用了 W3100 来完成以太网协议，其控制程序简单了许多。DSP 基本的操作与读写 ASRAM 一样，程序包括 6 个子模块：初始化模块、创建 socket 模块、网络连接模块、数据发送模块、数据接收模块、关闭 socket 模块。初始化模块主要完成对 DSP 自身的初始化和对 W3100 的初始化。DSP 的初始化任务主要是设置中断、设定 EMIF 接口参数。W3100 的初始化包括对网关、子网掩码、IP 地址、MAC 地址的设置。

下面是 W3100 的初始化程序。在头文件 W3100.h 中分别声明了各变量名并分配了与其对应的控制寄存器的地址。对 W3100 的所有设置均采用写－查询的模式进行，即先往 W3100 的控制寄存器中写控制字，再查询其状态寄存器，这样就可知道设置是否成功。其他模块的操作与初始化程序的基本原理相同。值得注意的是：由于 W3100 的发送缓冲双口 RAM 仅为 8 KB，在成批的数据发送时，一定要先查询 1 次发送数据指针，从而计算出可以利用的发送缓冲区的大小。

```
#include < W3100. h >
server. sin_addr. S_un. S_un_b. s_b1 = 192 ;
```

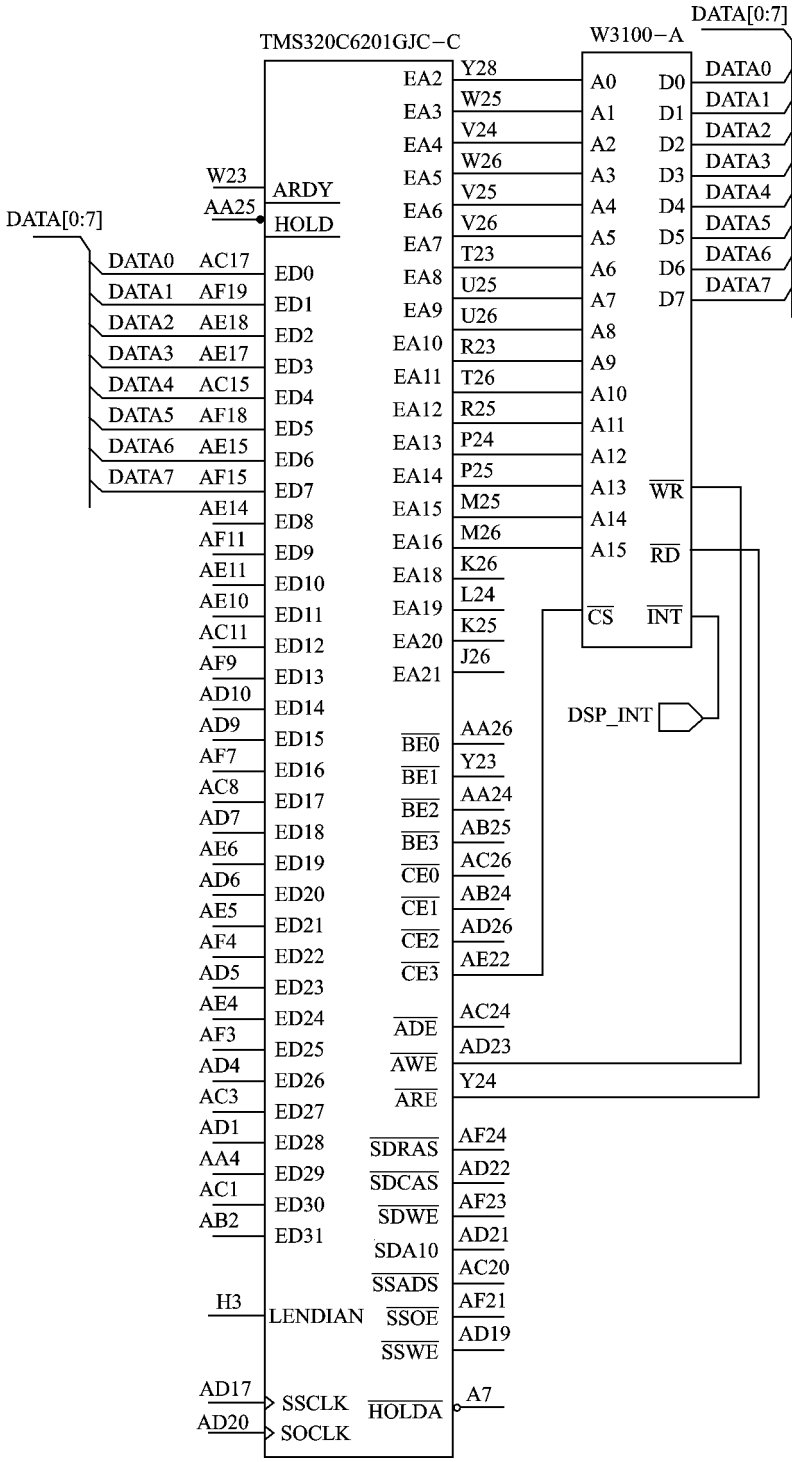


图 8.8－8　TMS320C6201 与 W3100 的接口电路

server. sin_addr. S_un. S_un_b. s_b2 = 168；

server. sin_addr. S_un. S_un_b. s_b5 = 55；

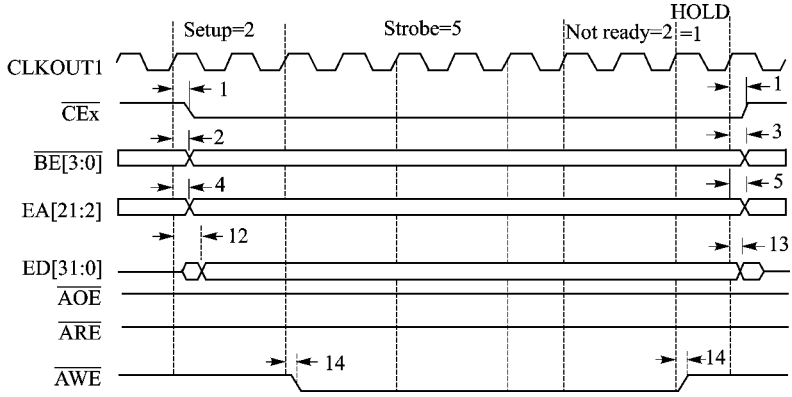


图 8.8－9 DSP 的异步写时序

```
server. sin_addr. S_un. S_un_b. s_b4 = 1 ;        / * gateway address * /
server. sin_addr. S_un. S_un_b. s_b1 = 255 ;
server. sin_addr. S_un. S_un_b. s_b2 = 255 ;
server. sin_addr. S_un. S_un_b. s_b3 = 255 ;
server. sin_addr. S_un. S_un_b. s_b4 = 0 ;        / * subnet mask * /
server. sin_addr. S_un. S_un_b. s_b1 = 192 ;
server. sin_addr. S_un. S_un_b. s_b2 = 168 ;
server. sin_addr. S_un. S_un_b. s_b3 = 55 ;
server. sin_addr. S_un. S_un_b. s_b4 = 101 ;      / * ip address * /
MACAddr [0] = 0x11 ;
MACAddr [1] = 0x42 ;
MACAddr [2] = 0x56 ;
MACAddr [3] = 0x43 ;
MACAddr [4] = 0x32 ;
MACAddr [5] = 0x68 ;                              / * MAX address * /
```

用 W3100 完成系统以太网接口具有硬件简单、软件编写容易等特点，几乎无需了解 TCP/IP 和以太网协议就可以方便地开发出具有 Internet 接入能力的产品。它一方面可以充分利用丰富的网上资源，另一方面系统功能也可以在网上实现共享。随着 Intetnet 应用的进一步普及与开发，原先的 PC 至 PC 连接模式将被彻底打破，必然出现 PC 到各种设备的连接和设备间的连接等各种模式。W3100 芯片的应用会越来越广泛。

参 考 文 献

1 W3100 Technical Datasheet. http ：//www. i2chip. com

2 (美) Arnett M F. TCP/IP 实用技术指南. 北京 清华大学出版社，1997

选自《微型机与应用》月刊，2002 年第 9 期

8.9 CAN 总线控制器与 DSP 的接口

武汉理工大学 (430070) 廖传书 李 崇

现场总线是一种开放式、数字化、多点通信的控制系统局域网络,是当今自动化领域中最具有应用前景的技术之一。CAN 总线是现场总线中的应用热点,CAN 总线支持分布式控制和适时控制的串行通信网络。由于 CAN 总线具有通信速率高、开放性好、报文短、纠错能力强以及控制简单、扩展能力强、系统成本低等特点,越来越受到人们的关注。基于 CAN 总线的 CAN 控制器具有完成 CAN 总线通信协议所要求的全部必要功能,因此 CAN 控制器与其他微处理器的接口成为设计 CAN 总线系统的首要工作。当前已有一些微处理器将 CAN 控制器嵌入到系统之中,成为在片的微处理器,例如 P8XC592 (其内核即为 80C51 的 CPU),MCS96 系列中的 87C196CA、87C196CB,TMS320 系列中的在片 CAN 微控制器 TMS320LF2407、TMS320F2810/F2812,但是仍有大量人们比较熟悉的微处理器并不带有 CAN 控制器。本文讨论这些微处理器与 CAN 控制器的接口问题,重点介绍 CAN 控制器与 TMS320 系列 DSP 的接口方法和接口电路。

一、CAN 控制器的接口信号和时序

CAN 控制器 (以 PCX 82C200 或 SJA1000 为例) 提供的微处理器的接口信号主要有 AD0 ~ AD7共 8 根地址数据线和 ALE、$\overline{CS}$、$\overline{RD}$、$\overline{WR}$、$\overline{RST}$、MODE、$\overline{RESET}$和$\overline{INT}$,控制器的数据和地址是分时复用线。其中 MODE 为接口方式选择信号,可选用 INTEL 方式或 MOTOROLA 方式。不同方式下引脚定义如表 8.9 - 1 所列,接口时序如图 8.9 - 1 和图 8.9 - 2 所示。

表 8.9 - 1 SJA1000 引脚定义

引脚符号	INTEL (MODE = Vdd)	MOTOROLA (MODE = Vss)
ALE	ALE	AS
$\overline{RD}$	$\overline{RD}$	E
$\overline{WR}$	$\overline{WR}$	RD/$\overline{WR}$

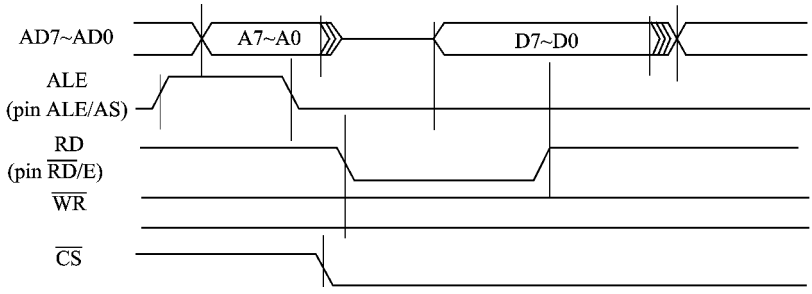


图 8.9 - 1 读周期时序图 (INTEL 模式)

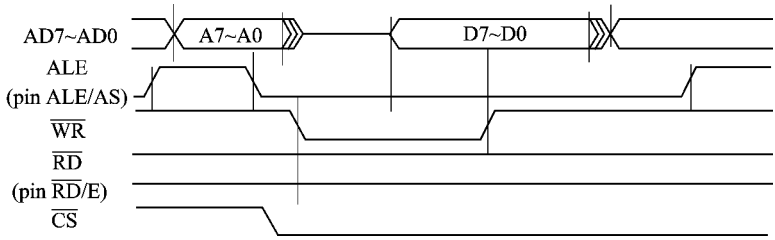


图 8.9－2 写周期时序图 (INTEL 模式)

从引脚定义和时序关系可知 CAN 控制器提供了与 INTEL 方式和 MOTOROLA 方式的直接接口信号,其中 INTEL 方式对于目前流行的 51/96 系列单片机来说提供了方便快捷的直接接口设计。

二、DSP 的接口信号和时序

DSP 芯片以 TI 公司生产的 TSM320 系列产品为国内的主流产品。TSM320 系列产品至今已经历了若干代,有 'C1X、'C2X、'C2XX、'C5X、'C54X、'C62X 等定点 DSP,有 'C3X、'C4X、'C67X 等浮点 DSP 和 'C8X 多处理器 DSP。DSP 采用了先进的哈佛结构,内部采用多总线结构和流水线的工作方式,从而大大地提高了系统的运行速度和数字信号的处理能力,DSP 的指令执行时间在 ns 数量级,内部程序和数据存储器目前已达几十 K 字,并带有内部的硬件乘法器,这些都为 DSP 提供了广阔的应用空间。

DSP 芯片的片外引脚一般采用地址线和数据线分离的设计方法,不再使用地址数据分时复用线,也没有 ALE 地址有效信号,这样就给 CAN 控制器与 DSP 的接口带来一定困难,且不同的 DSP 芯片外部引脚和时序也略有区别。要设计 CAN 控制器与 DSP 的接口,首先必须讨论一下 DSP 的时序,下面以 DSP 中较流行的 TMS320LF2407 和 TMS320VC5402 为例进行讨论。

1. TMS320LF2407 DSP 的 I/O 时序

DSP 的存储器分为三个空间 程序存储器空间、数据存储器空间和 I/O 空间。I/O 空间有专用的输入指令 PORTR 和输出指令 PORTW 以及专用的 I/O 空间选择信号$\overline{IS}$,TMS320LF2407 的 I/O 信号与存储器操作信号复用,它们是存储器和 I/O 设备选通信号$\overline{STBR}$、写选通信号$\overline{WR}$、读选通信号$\overline{RD}$和读写信号 R/$\overline{W}$,TMS320LF2407 的 I/O 时序如图 8.9－3 和图 8.9－4 所示。

2. TMS320VC5402 DSP 的 I/O 时序

TMS320VC5402 与 TMS320LF2407 一样,用$\overline{IS}$作为 I/O 空间选择信号,不同的地方是 I/O 空间有专用的 I/O 设备选通信号$\overline{IOSTRB}$和通用的读写信号 R/$\overline{W}$,而不设读选通信号$\overline{RD}$,和写选通信号$\overline{WR}$,其时序如图 8.9－5 和图 8.9－6 所示。

3. DSP 的 I/O 时序分析

I/O 的输入或输出工作周期一般在两个机器周期内完成,在此期间,$\overline{IS}$信号和地址总线一直保持有效。对于 TMS320LF2407,I/O 选通信号$\overline{STRB}$发生在第一个机器周期有效之后并持续一个机器周期以上,$\overline{RD}$和$\overline{WE}$有效时数据有效。对于 TMS320VC5402,I/O 设备选通信号$\overline{IOSTRB}$的低电平有效发生在延迟了半个机器周期的上升沿到下一个机器周期的上升沿,持续

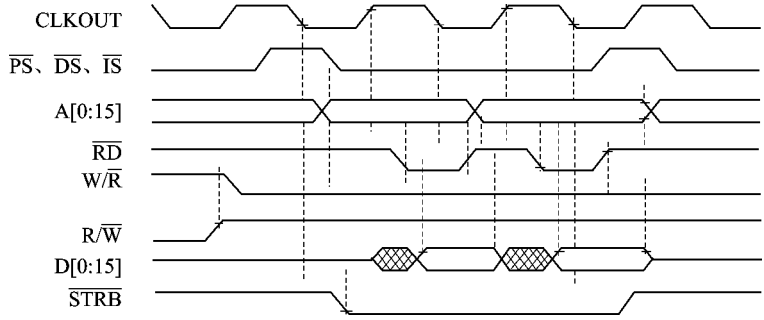


图 8.9-3 存储器接口读/读时序

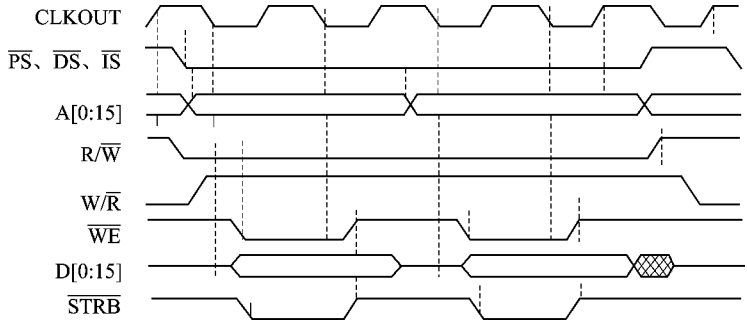


图 8.9-4 存储器接口写/写时序

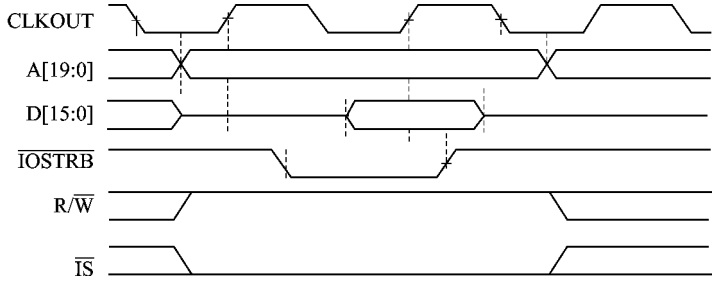


图 8.9-5 并行 I/O 口读时序 ($\overline{IOSTRB}=0$)

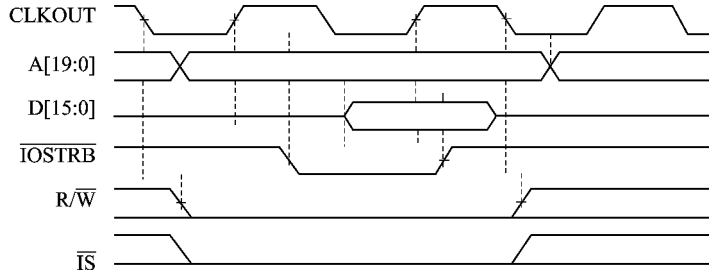


图 8.9-6 并行 I/O 口写时序 ($\overline{IOSTRB}=0$)

一个机器周期,数据有效发生在第二个机器周期内。R/$\overline{W}$ 读写信号在输入周期内一直保持为“1”,在输出周期一直保持为“0”,仅起到控制数据流的方向作用。以上分析都没有考虑插入

等待周期的情况,若插入一个等待周期,则每次 I/O 操作均延长一个机器周期,即需要三个机器周期完成 I/O 操作 (等待周期时序从略)。

三、CAN 控制器与 DSP 的接口设计方法

从以上分析可以看到,TMS320 系列 DSP 没有提供与 SJA1000 CAN 控制器的直接接口信号,以 SJA1000 的 INTEL 方式为例,为了使 TMS320 系列 DSP 满足 SJA1000 的接口信号要求,可以从以下几点进行设计。

1. 地址数据复用线的设计

将 DSP 的数据线 D0 ~ D7 作为 CAN 的地址/数据复用线,用 DSP 的数据线去选择 CAN 的内部端口和传送数据。

2. 地址有效信号 ALE 的产生

对于 TMS320LF2407,用地址线 A0、写选通信号$\overline{WR}$和端口选通信号$\overline{STRB}$的逻辑组合产生 DSP 的 ALE 信号。对于 TMS320VC5402,则用地址线 A0、I/O 端口选通信号$\overline{IOSTRB}$的逻辑组合产生 ALE 信号。

3. 读写信号的产生

对于 TMS320LF2407,用读信号和 A0 的逻辑组合产生 SJA 1000 的读选通信号,用写信号和 A0 的逻辑组合产生 SJA 1000 的写选通信号。对于 TMS320VC5402,则用 A0、$\overline{IOSTRB}$和 R/$\overline{W}$ 的逻辑组合产生 SJA1000 的读和写选通信号。逻辑关系如表 8.9 - 2 所列。

表 8.9 - 2 TMS320LF2407 和 TMS320VC5402 与 SJA 1000 接口逻辑

TMS320LF2407				TMS320VC5402			SJA1000		
A0	$\overline{STRB}$	R/$\overline{W}$	$\overline{WE}$	A0	$\overline{IOSTRB}$	R/$\overline{W}$	ALE	$\overline{WE}$	$\overline{RD}$
1	0	0	X	1	0	0	1	1	1
0	0	0	0	0	0	0	0	0	1
0	0	1	1	0	0	1	0	1	0

4. 片选信号的产生

用 DSP 的 I/O 空间选通信号$\overline{IS}$和高位地址的译码信号的逻辑组合产生 CAN 的片选$\overline{CS}$。

从以上设计思想可以看到,这种方法是将 DSP 的数据线改为适应 CAN 控制器的数据地址线。为此将 DSP 的 A0 作为地址数据选择线。A0 = 1 时,地址有效;A0 = 0 时,数据有效。即用奇数地址选择端口,用偶数地址传送数据。同时,通过信号的逻辑组合,在地址有效期间不产生读写信号,而是产生满足 CAN 的地址有效信号 ALE 在数据有效期间产生满足 CAN 的读和写逻辑信号时序。

四、CAN 与 DSP 的接口电路

以 TMS320VC5402 与 SJA1000 芯片为例设计的接口电路如图 8.9 - 7 所示。图中,用一片 GAL16V8B 作为接口逻辑转换电路。为突出接口电路,其他部分从略。用 FM 书写的设计文件如下:

GAL16V8B

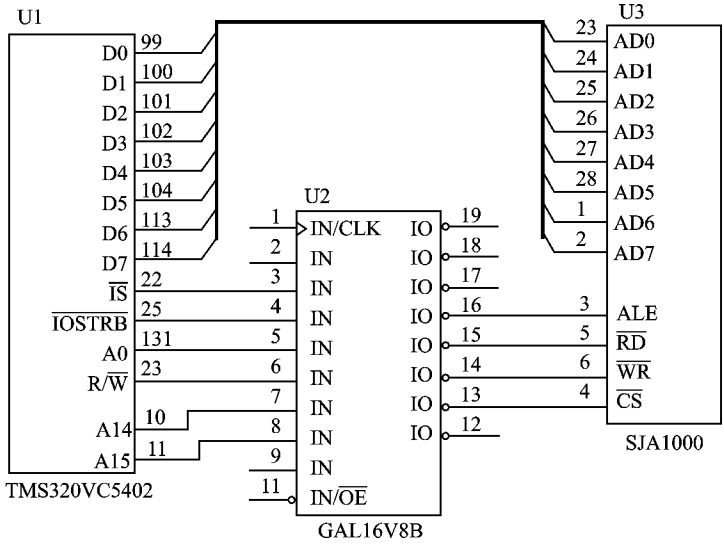


图 8.9－7 TMS320VC5402 与 SJA1000 的接口电路

INTERFACE

CH SH APR 19. 20002

DECODER

NC NC IS IOSTRB A0 RW A14 A15 NC GND

NC NC CS WR RD ALE NC NC NC VCC

$\overline{CS} = A15 * A14 * \overline{IS}$

$ALE = \overline{A0} * \overline{IOSTRB} * R/\overline{W}$

$\overline{RD} = A0 * \overline{IOSTRB} * R/\overline{W}$

$\overline{WR} = A0 * \overline{IOSTRB} * R/\overline{W}$

DESCRIPTION

参 考 文 献

1 邬宽明. CAN 总线原理和应用系统设计. 北京 北京航空航天大学出版社,1996

2 阳宪惠. 现场总线技术及其应用. 北京 清华大学出版社,1999

3 TMS320C2XX Uesr s Guide. Texas Instruments, 1997

4 戴明桢,周建江. TMS320C54XDSP 结构、原理及应用. 北京 北京航空航天大学出版社,2001

选自《电子技术应用》月刊,2002 年第 11 期

8.10 基于 DSP 的 USB 传输系统的实现

上海交通大学电子工程系信号处理研究所 (200030)
陶 峻 潘亚涛 陈 健

一、概 述

通用串行总线 (USB) 是一种通用串行总线系统[1]。它可以解决传统的计算机外设接口不通用,以及有限的接口数目无法满足多外设连接的问题。USB 协议最早出现于 1996 年 1 月,目前已得到了广泛的推广,包括 Windows 98,Windows 2000,Windows CE 和 MAC OS 在内的多种流行操作系统均对 USB 技术提供了全面的支持。

USB 的主要特点包括:

(1) 在外设要求电压为 5 V,且所需电流小于 500 mV 时,可由 USB 总线直接供电。

(2) USB 系统中的外设接口之间采用菊花链形式连接,最多可连 127 个外设。

(3) USB 即插即用,几乎无需用户的干预。

(4) USB 能提供三种速率——低速 (1.5 Mbps)、中速 (12.5 Mbps) 和高速 (360 Mbps 以上)。

(5) 提供了四种数据传输形式——控制、等时、中断和块传输。

现在,国内对 USB 的研究也已经成为一个热点,通常采用微控制器 (MCU) 来控制 USB 接口芯片,它的开发和实现简单,价格较为便宜,但是随着多媒体的发展,需要处理和传输的信息也不断地增加,而传统的 MCU 无法适应这种高速的数据处理。相比之下,DSP 则拥有高速的数字信号处理能力。以 MP3 解码为例,在 TMS320C54 定点 DSP 平台上,以汇编语言编写的解码程序在 128 Kbps stereo 条件下,可实现约 40MIPS 的实时解码[2],而现在正趋流行的 MPEG-2 AAC 同样在 TMS320C54 定点 DSP 平台上,码流取样率为 48 kHz,64 Kbps/Channel 条件下,运算量则高达约 72 MIPS[3]。如果通过 DSP 对 USB 进行控制,则可将 USB 接口芯片直接加入用户系统,令系统集简便,高速的数据运算处理和传输于一体。现在市场上较为流行的 MP3 播放器、数码照相机等就是采用这种方法实现与计算机的互联。但对基于 DSP 的 USB 传输系统的具体实现过程,国内尚少有报道。本文针对一个通用 USB 传输模块,研究了基于 DSP 的 USB 系统的完整结构和工作过程。并准备将它用于 MP3 编解码器与主机进行 MP3 文件的传输。

二、USB 系统结构

图 8.10-1 为 USB 的系统结构图。主要由以下四个部分组成。

(1) USB 物理设备 连于 USB 电缆上作为终端的用户设备,如调制解调器、打印机等。

(2) 客户软件 较高层的 USB 设备操作软件,在 USB 主机上运行。通常,如果主机是 PC,

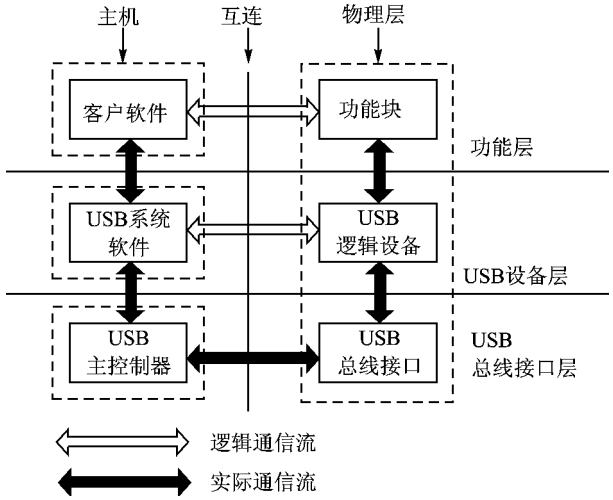


图 8.10－1 USB 系统结构

而 USB 物理设备符合定义的 USB 类设备,则 USB 客户软件将成为主机的操作系统的一部分。

(3) USB 系统软件:它是在操作系统中支持 USB 操作的软件。它独立于特定的 USB 设备和用户软件,经常是被作为核心操作系统的一部分提供的。

(4) USB 控制器:是使 USB 设备连到 USB 主机上的软、硬件,例如 IC 控制器。控制芯片是实现 USB 潜力并使其得到广泛、便宜的应用的关键。

整个 USB 系统的工作原理如下:

USB 主机在整个 USB 体系中占有独一无二的地位,除了拥有特殊的物理位置,主机还对总线及其上连接的外设负责。主机能够随时自动搜索,识别新添加的 USB 外设,并自行启动该 USB 外设的驱动程序和应用软件。操作系统可通过与 USB 外设的对话自动完成带宽、地址的分配等一系列操作。设备拆除后,操作系统也能够自动地卸载相应的驱动程序,收回已分配的资源。一个 USB 外设只有得到主机的授权才能接入总线。

USB 物理设备为主机提供附加的功能,所有的 USB 逻辑设备向主机展示相同的基本接口,这使得主机可以以相同的方式管理不同的 USB 外设中的 USB 部分。为使主机能识别 USB 外设,完成装置,每个外设都带有并能向主机汇报与配置有关的信息,一些信息在所有逻辑设备中是一样的,另一些则与外设担负的功能有关,即与外设所对应的 USB 类密切相关。

客户软件决定要对外设作什么样的传输,它利用指定的操作系统接口要求 I/O 请求包(IRPs)。客户软件只与它要操纵的那个外设的所拥有的端点有关,它提出的请求通过通用串行总线驱动器(universal serial bus driver,简称 USBD)接口提交。所有被提交的 IRPs 必须遵守在管道建立时所协定的一切,如带宽。

USB 系统软件(支持 USB 的操作系统一般都提供)一般包括 USBD 和主机控制驱动器(host controller driver,简称 HCD)。USBD 在外设接入总线及配置时被调用,以保证总线能够提供所需的资源。USBD 收到的配置请求包括:端点号、传输类型、传送周期、数据长度等。USBD 根据可用带宽和总线能力决定是否接受该请求。如果它接受了该请求,就为该传输创建一个管道。当一个外设配置完毕后,客户软件就可提出 IRPs,在它和外设之间进行数据传输。HCD 负责跟踪 IRPs,并保证 USB 带宽和最大帧时间不会被超出。当针对某一管道的 IR-

Ps 出现后，HCD 就将它加到事务队列中。当 IRP 完成后，HCD 将通知对应的客户程序。

三、DSP 控制的 USB 传输系统的硬件结构

本系统要用的 DSP 芯片是美国 Taxes Instruments 公司生产的通用芯片（TMS320C54）。USB 接口芯片是美国 National Semiconductor 公司的 USBN9602 芯片，它的控制较简便，目前在国内使用较为广泛。它们之间的连接如图 8.10－2 所示。在 TMS320C54 上对 USBN9602 这

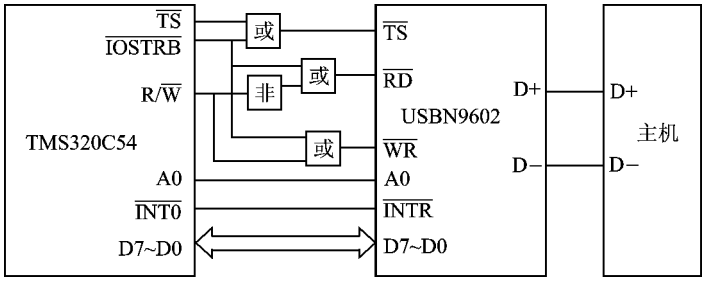


图 8.10－2 DSP 控制 USB 的硬件连接图

一 USB 接口芯片的控制。主要通过读写 USBN9602 内的寄存器完成各种信息的交流和相应的操作来实现的。USBN9602 向控制器提供三种接口模式——微传线（microwire）接口、复用并行接口、非复用并行接口。Microwire 是一种串行模式。复用并行接口是在一个周期内在并行线上地址与数据复用，这种方式一般用在 MCU 控制的 USBN9602 系统中，通过 MCU 发出的锁存信号来控制时序。非复用并行模式的地址和数据是在不同的周期分别出现于数据线上。由于 C54x 系列地址线与数据线分开，数据与地址在整个周期内有效，所以在此 USBN9602 采用的是非复用并行接口读写模式。USBN9602 靠中断（INTR）通过 TMS320C54 的外部中断口（INT0）来要求服务。TMS320C54 的 I/O 设备选通信号（IOSTRB）和 I/O 空间选择信号（IS）决定了 USBN9602 的片选信号（CS）。TMS320C54 的I/O地址线 A0 连接 USBN9602 的 A0 线决定了当前数据线上（D0 ~ D7）走的是寄存器地址还是设置（或读取）寄存器内的值[4,5]。

四、DSP 控制 USBN9602 的软件实现

TMS320C54 对 USBN9602 的操作是基于它的端点结构的。USBN9602 有 7 个端点，其中 0 号端点是双向的（既可作为接受管道，也可作为发送管道），一般用来完成管理功能。在 USBN9602 连上主机后，外设首先要完成的是与主机的参数交换，以确定设备类型、端点地址设定、最大包长度等各种配置值。0 号端点作为缺省端点用以在 USB 设备和主机之间传递各种配置信息，如发 Descriptor 数据包等。其余的有三个作为发送端点，三个作为接受端点。

对 TMS320C54 而言，USBN9602 是外设，扩展为 TMS320C54 的 I/O 地址空间。根据硬件时序要求，当要对 USBN9602 进行读写时，首先往一个 I/O 口输出 USBN9602 的相应的寄存器地址，待 USBN9602 做好交换数据的准备后，TMS320C54 就从另一个 I/O 口送出处理好的数据（状态字）或读入所需要处理的数据（状态字）。

读过程由以下程序实现：

```
read_usb：    port (PA1) = @addr
              @ddata = PORT (PA0)
```

```
        return
```

写过程由以下程序实现：

```
write_usb：  port (PA1) = @addr
             port (PA0) = @ddata
             return
```

图 8.10－3 是整个主程序的大致流程。TMS320C54 首先完成的是对 USBN9602 的初始化，包括使能缺省端点 0，设置终端屏蔽寄存器（包括主中断屏蔽寄存器、接受、发送、ALT 和 NAK 等事件屏蔽寄存器），设置端点 0 为接受状态，将外设连入总线等。TMS320C54 完成对 USBN9602 的初始化之后，可以承担 USB 接口以外的用户系统的其他服务，如 MP3 的解码程序等。一旦 USB 总线上有变化后，USBN9602 除了能在状态寄存器上有所反映之外，还会立刻通过 INTR 管脚向 TMS320C54 提出中断请求。于是，TMS320C54 立刻转到中断服务程序[6]。

图 8.10－3 DSP 控制 USBN9602 的主程序

图 8.10－4 是整个中断程序的结构。中断服务程序首先读 USBN9602 的主事件寄存器，判断中断原因，包括：主机要求向 USBN9602 的某个接受端点发送数据；对某个发送端点运行反馈（通知传送是否成功，是否能继续传送数据）；发生有关响应的 NAK 事件；发生状态转变的 ALT 事件。根据这些原因，TMS320C54 将进一步深入处理，如探寻是哪一个端点需要服务，并据此调用相应的子程序。

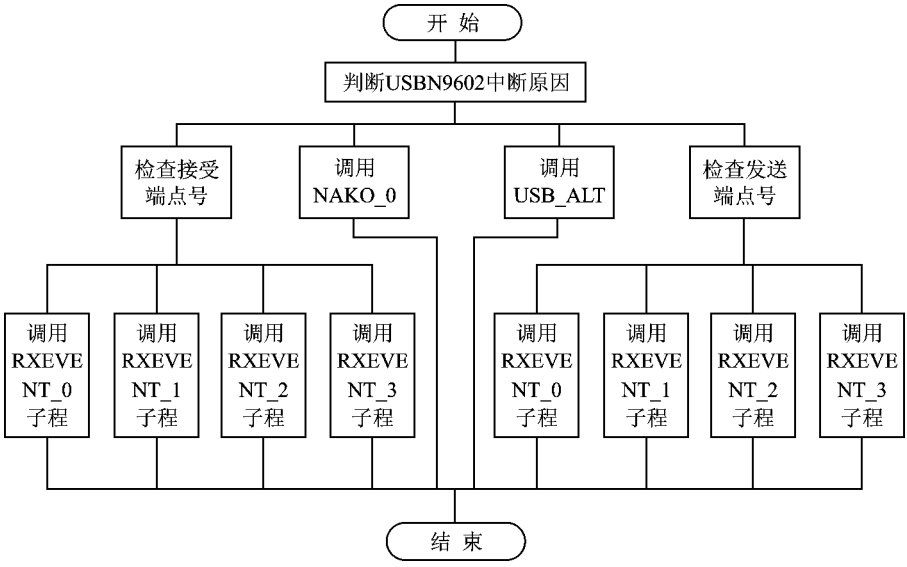


图 8.10－4 中断服务程序流程图

一般接收端点的子程序，处理过程如下：

(1) 检查所收到的包的状态，如果正确，就将数据从端点中读出。

⑵ 解读数据并做出相应处理,如对数据进行压缩、编码等。

⑶ 如果是端点 0,就向管道写入相应的数据包,并使能发送端点,否则,再次使能该接收管道。

一般发送端点的子程序,处理过程如下:

⑴ 检查上次传送是否成功,如果出错就进行相应的处理。

⑵ 如果还需传送下一个数据包,就将其写入该发送端点,并发送使能,否则对于发送端点 0,就使能接受端点,对于其他端点,就什么都不做。

以 0 号接受端点的子程序为例。它将负责所有的标准 USB 外设请求,TMS320C54 首先读取 USBN9602 的接收状态寄存器,根据其内的值,判断此次接收到的数据包是 SETUP 包还是 OUT 包。如果是标准的 SETUP 包,则进一步解读这次要求的类型,并由发送端点 0 向主机发送需要的信息。

五、结 论

本文对以 DSP 芯片为控制器的 USB 传输系统进行了研究,详述了完整的 USB 系统工作原理,研究了具体的硬件和软件实现。我们研究表明对于现有的各种以 DSP 为核心电子产品,只需加入 USB 接口芯片就可实现用 USB 接口传输数据。DSP 完全可以代替 MCU 实现对 USB 接口芯片的控制。这种控制方法可以大大降低对系统进行改进的复杂程序,减少成本,缩短研发周期,在数字图像、电话语音合成、交互式多媒体、消费电子产品等领域具有广阔的应用前景。

参 考 文 献

1 Universal Serial Bus Specification, Revision 1. 1. 1998 (9)

2 高智衡,韦岗. MP3 宽带音频压缩中的核心技术. 电声技术,2000 (9) :9 ~ 13

3 王华明,徐盛,陈健. 用定点 DSP 实现 MPEG-2 AAC 实时解码器. 电声技术,2001 (3) :3 ~ 7

4 Taxes Instruments. TMS320C54x Data Sheet. 1999

5 Jim Lyle. USBN9602 Interface Examples. 1998

6 Jim Lyle. USBN9602 Firmware Description. 1998

选自《电子技术》月刊,2002 年第 3 期

第九章

HDL 与可编程器件技术

9.1 谈谈 EDA 的硬件描述语言

韩 鹏 彭斯福

在电子信息时代的今天,电子设计自动化 EDA 技术已经占领了整个电子领域的重要地位。掌握 EDA 技术的关键之一是掌握一门甚至几门硬件描述语言 HDL (Hard Description Language),正如软件工程师要进行软件工程设计就得掌握如 C、C + +、java 等高级编程语言一样,电子工程师也有属于自己领域的设计语言。HDL 是用于硬件造型结构的高级编程语言,它能更有效地表示硬件电路的特性,更便于阅读交流和相互调用。硬件描述语言至今已有几十年的发展历史,在这个过程中产生了上百种硬件描述语言,它们是由厂商、企业根据各自的需要创造的,具有一定的针对性,没有标准化,所以对于一般用户来说,要选择其中一门语言进行学习则会显得无从下手。这里我们介绍三种现今电子电路及系统设计中应用较广泛的三种语言,方便读者根据自身的需要选择学习。

一、三种语言的简单介绍

1. ABEL - HDL

这是一种语言结构简单,限定较少的 HDL 语言。于 1983 年推出,不是标准语言,但问世早,使用方便。在可编程逻辑器件的设计中,可方便准确的描述所设计的电路逻辑功能。

2. Verilog—HDL

是一种专门为复杂数字逻辑电路和系统的设计仿真而开发的 HDL 语言,也可以说 Verilog—HDL 是在应用最广泛的 C 语言的基础上发展起来的一种硬件描述语言。Verilog—HDL 是由美国电气和电子工程师协会 IEEE 制定的目前惟一两种标准化语言之一。由 Cadence 公司 1990 年公开,并成立了 OVI (Open Verilog International) 组织来负责其发展。

3. VHDL

VHDL (VHSIC Hardware Description Language, VHSIC 即为 Very High Speed Integrated Circuit 的缩写词) 的中文译义为超高速集成电路硬件描述语言,由美国军方研制专用集成电路所指定的硬件描述语言。Verilog—HDL 是 IEEE 制定的另一种标准化语言。

二、三种语言的比较

1. 表格表示

表格表示如附表所列。

附 表

语言	硬件描述能力	应用范围	掌握难度
ABEL—HDL	支持逻辑电路的多种表达形式； 属描述级别； 语言的特性受支持的程度不如 Verilog—HDL	适用于各种不同规模的可编程器的设计； 更适合于对几千门的小系统使用	容易
Verilog—HDL	门级开关电路描述方面比 VHDL 强，在系统及抽象方面比 VHDL 略差一些	适合复杂数字逻辑电路和系统的仿真和综合	较容易
VHDL	支持硬件的设计、验证、综合和测试，还支持硬件设计数据的交换、维护、修改和硬件的实现； 行为描述能力强； 能进行系统级的硬件描述	适合特大型（几百万门级以上）的系统级设计	较困难，通常在 Verilog—HDL 的基础上学习

2. 几点说明

⑴ 硬件描述能力

ABEL—HDL 支持逻辑电路的多种表达形式，其中包括逻辑方程、真值表和状态图。与后两种相比描述能力要弱一些，ABEL 语言和 Verilog 语言同属一种描述级别，但 ABEL 语言的特性受支持的程度远远不如 Verilog—HDL，但是在应用于小系统中其能力足够了。

Verilog—HDL 语言是从集成电路的设计中发展而来，同时它早在 1983 年就已推出，因此它拥有广泛的设计群体，语言较成熟，资源更丰富。在门级开关电路描述方面比 VHDL 强。

VHDL 语言具有多层次描述系统硬件功能的能力，可以从系统的数学模型直到门级电路。尤其具有更强的行为描述能力，强大的行为描述能力是避开具体的器件结构、从逻辑行为上描述和设计大规模电子系统的重要保证，这就使它能够成为系统设计领域中最佳的硬件描述语言。例如，在计算机与外部的连接上要设计一块接口电路，这时，对计算机部分的功能可以用行为方式描述，而接口电路可以采用 RTL 方式描述，在系统仿真时可验证接口电路是否正确，这样在接口电路设计出来以前就可知接口电路是否能正常工作。VHDL 语言最突出的一个优点是能进行系统级的硬件描述，可以自定义数据类型，给编程人员带来较大的自由和方便。

⑵ 应用范围

ABEL—HDL 语言易于对几千门的小系统使用，因为它不含延迟描述，故上万门的系统不宜采用。ABEL—HDL 被广泛用于各种可编程逻辑器件的逻辑功能设计，由于其语言描述的独立性，因而适用于各种不同小规模的可编程器的设计。

Verilog—HDL 语言适合复杂数字逻辑电路和系统的仿真和综合，较适合系统级、算法级、寄存器传输级、逻辑级、门级、电路开关级等的设计。

VHDL 语言则是面向多层次、多领域的，它广泛用于电路设计文档、设计描述的逻辑综合和电路仿真，更适合特大型（几百万门级以上）的系统级设计。虽然它源于军方集成电路使用，但由于自身的优点，越来越被大规模集成电路厂家及教育界所重视和使用。

从长远来看，VHDL 和 Ver—ilog—HDL 的运用会比 ABEL—HDL 多得多，ABEL—HDL 只会在较小的范围内继续存在。

⑶ 掌握的难易程度

ABEL—HDL 易学易用。

Verilog—HDL 语法自由，由于风格类似于 C 语言，因此只要有 C 语言基础的初学者要掌握 Verilog—HDL 是非常容易的。

VHDL 掌握起来较困难，它语法严格，比较抽象，一般需要有 Ada 的编程基础。通常先入手学习 Verilog—HDL 再学习 VHDL。

三、今后的发展预测

从 EDA 技术的发展趋势上看，直接采用 C 语言设计 CPLD/FPGA 将是一个发展方向，现在已出现用于 CPLD/FPGA 设计的 C 语言编译软件。当今，VHDL 和 Verilog—HDL 已作为 IEEE 的工业标准硬件描述语言，且得到众多 EDA 公司的支持，在电子工程领域，已成为事实上的通用硬件描述语言。VHDL 与 Verilog—HDL 将承担起大部分的数字系统设计任务。另外，目前的硬件描述语言大多用在数字电路的设计，对模拟电路的硬件描述语言正在逐步发展，并开始应用到实际当中。

选自《电气时代》月刊，2002 年第 10 期

9.2 基于 VHDL 语言的 FPGA 设计

南京邮电学院计算机科学与技术系 (210003) 王华 王汝传 吴 凡

现场可编程门阵列 FPGA (Field Programmable Gate Array) 技术是近几年发展起来的新型电路实现技术。它具有保密性强、体积小、重量轻、可靠性高等 ASIC (Application Specific Integrated Circuit) 电路的共同优点,是一种新型的 ASIC 产品。同时它还具有门阵列的高速、高密度和 PLD 的现场可编程特性。因此与普通 IC 和 PLD 等相比,FPGA 具有设计开发周期短、设计制造成本低、开发工具先进等优点。因而在短短几年里被广泛应用在数字通信、图形图像、仪表、兵器等系统中。

但是随着集成电路设计技术的日趋进步和日臻完善,集成电路的设计规模日益增大,复杂程序日益增高,使逻辑图和布尔方程的硬件描述方法变得相当复杂。这就迫切需要一种更高抽象层次的现代设计方法和现代测试手段来描述硬件电路。

在上百种硬件描述语言 HDL 中,最有代表性的是美国国防部开发的超高速集成电路硬件描述语言 VHDL (Very High Speed IC Hardware Description Language)。它是一种数字硬件系统设计和描述的标准语言,具有与工艺无关、支持大规模系统设计等特点。它的产生使得"高层设计"逐渐进入实用阶段。

"高层设计"就是"概念驱动式"设计。设计人员无需通过门级原理图描述电路,而是针对设计目标"自顶而下"地进行功能描述。这些高层描述输入计算机后,电子设计自动化系统就能以规则驱动的方式完成整个设计。由于高层设计抽象程度高,因此设计工作省时省力,缩短了设计周期。此外,高层设计只定义系统的行为特性,可以不涉及实现工艺。采用高层设计方法,人们利用综合优化工具 VHDL 或其他方法描述的定义将设计电路转换成针对某种工艺优化的网表。由于不同修改原设计,工艺转换变得轻松容易。在现今大规模复杂的数字电路与系统的设计中,"高层设计"将逐步取代门级描述、逻辑电路图和布尔方程等级别较低的、繁琐的硬件描述方法。

本文将介绍使用 VHDL 语言实现 FPGA 设计的方法 ,并以此方法在 Xilinx 公司的 FPGA 器件 XC4010EPC84 上实现 8255 芯片。

一、设计分析

1. 8255 芯片简介

Intel 8086/8088 系列可编程外设接口 (Programmable Peripheral Interface,PPI) 电路,型号为 8255 的芯片 (改进型为 8255A 及 8255A-5),具有 24 条输入/输出引脚和可编程的通用并行输入/输出接口电路,其框图如图 9.2-1 所示。8255 的通用性强、使用灵活,通过它 CPU 可直接与外设相连接。8255 具有 3 个相互独立的输入/输出通道 :通道 A、通道 B、通道 C。A、B、C 3 个通道可以联合使用,构成单线、双线或 3 线联络信号的并行接口,此时 C 口完全服务于 A 口和 B 口。A 口有 3 种工作方式 :方式 0、方式 1、方式 2。B 口有 2 种工作

方式 :方式 0、方式 1。

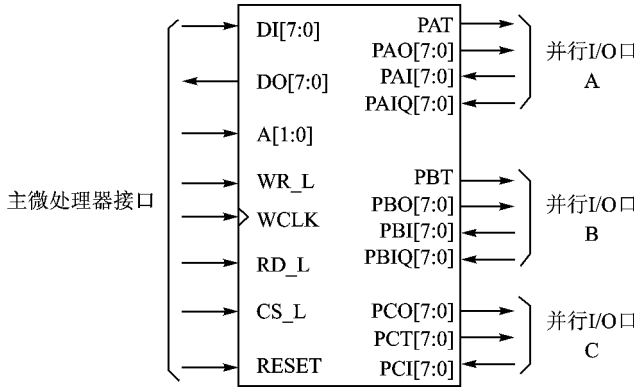


图 9.2 -1 8255 芯片框图

2. 系统设计分析

系统采用模块化设计,以提高设计的可重用性。将 8255 内核和外围逻辑分离设计,分离结果如图 9.2 -2 所示。内核模块包括 :A 口、B 口和 C 口控制器,读写控制逻辑,以及 A、B、C 3 个 8 位端口。外围逻辑包括 8255 内核与外部设备的接口以及数据总线缓冲器。本设计使用的软件为 Xilinx 公司的 PLD 开发软件 Foundation 3.1i,它支持多种设计输入方式,如原理图、状态机、VHDL 语言。这 3 种方式各有优点,可以灵活使用。在本设计中,8255 芯片本身采用 VHDL 语言设计。

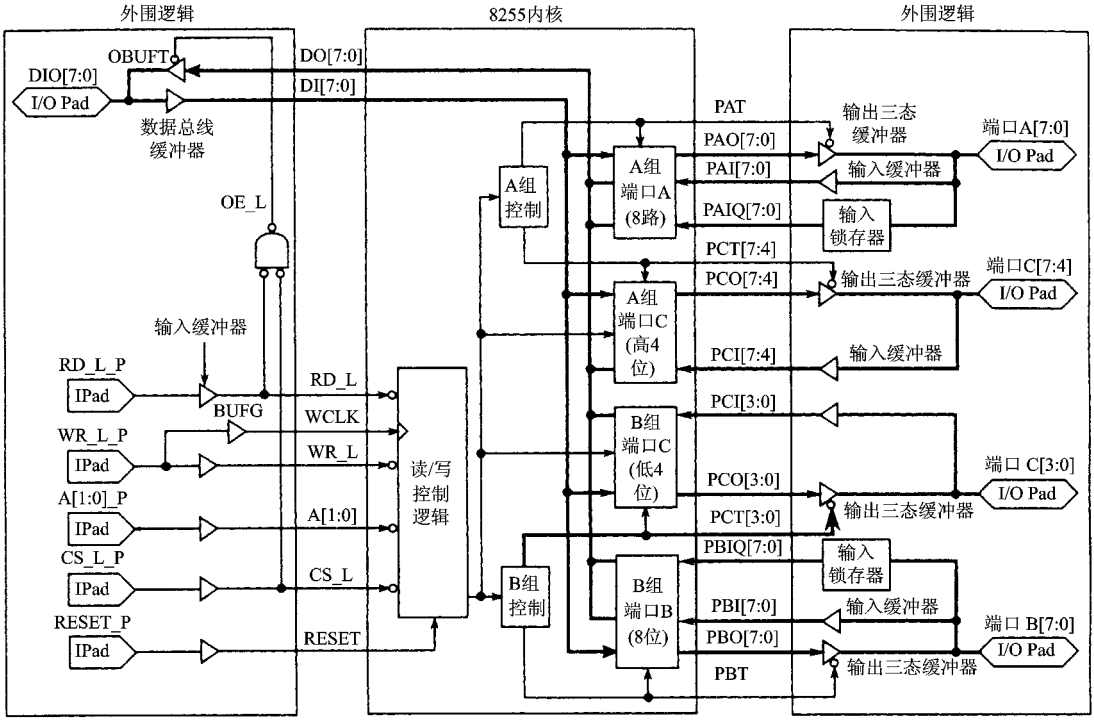


图 9.2 -2 8255 模块分解图

二、8255 实现

1. 程序设计

8255 由多个模块组成,可以分别设计再加以组合。VHDL 把一个数字系统当成一组模块来描述,每个模块具有与其他模块的接口以及关于模块内容的描述。VHDL 把它们放在不同的设计单元——实体 (Entity) 和结构体 (Architecture) 中。实体与传统设计方法中的符号相对应,它定义了模块被调用时所需的信息,包括输入、输出信号接口及有关参数。结构体描述模块内部的结构和行为状态,它与传统设计方法中的电路原理图相对应。一个实体可以有多个结构体,各结构体分别代表该实体所定义模块的不同实现方案。下面以设计外围逻辑部分所用到的 IO 缓冲器的设计为例,给出实体及结构体的定义。

实体定义为:

```
entity IOB is port(
    P   :inout std_logic;      G   :in   std_logic;
    T   :in std_logic;         I   :in   std_logic;
    O   :out std_logic;        Q   :out  std_logic);
end IOB;
```

结构体定义为:

```
    architecture rt1 of IOB is
    component PULLUP port(
        O   :out std_logic);
    end component;
    component ILD_1 port(
        Q   :out std_logic;D   :in std_logic;
        G   :instd_logic);
    end component;
    component IBUF port(
        O   :out std_logic;I   :in std_logic);
    end component;
    component OBUFT port(
        O   :out std_logic;I   :in std_logic;
        T   :in std_logic);
    end component;
begin
    pup :PULLUP      port map(Q=>P);
    ild :ILD_1       port map(Q=>Q,D=>P,G=>G);
    ibf :IBUF        port map(O=>O,I=>P);
    obf :OBUFT       port map(O=>P,I=>I,T=>T);
end rtl;
```

将各个功能独立的模块单独编写,再将几个相关的模块以 COMPONENT 的形式组合成功

能更复杂的模块,最后在顶层的模块中调用。例如首先生成 IO 缓冲器模块 IOB,然后在 8 路 IO 缓冲器模块中以 COMPONENT 形式调用 IO 缓冲器模块 IOB,最后在顶层 8255 实体中再以 COMPONENT 的形式调用 8 路 IO 缓冲器模块 IOB8。

8255 的各种行为如写控制字、读/写端口数据等可以用过程来实现,然后在实现结构体的时候调用。下面给出写控制字时所使用的过程:

```
procedure cmd_write (
    signal cmd_addr :in std_logic_vector (1 downto 0) ;
    signal cmd_data :in std_logic_vector (7 downto 0) ;
    signal D    :in std_logic_vector (7 downto 0) ;
    signal  A   :out std_logic_vector (1 downto 0) ;
    signal CS_L    :out std_logic ;
    signal WR_L    :out std_logic ;
    signal d_oen   :out std_logic ;
    signal d_in    :out std_logic_vector (7 downto 0) ) is
        variable outvects : line ;
begin
    A      < = cmd_addr ;CS_L    < = '0' ;
    WR_L   < = '0' ;        d_in    < = cmd_data ;
    d_oen    < = '1' ;        wait for 100 ns ;-- 100ns
    write (outvects,string' ("--------------------")) ;
    writeline (simresul,outvects) ;
    write (outvects,string' ("Write :Time = ")) ;
    write (outvects,now) ;write (outvects,string' (",Address = ")) ;
    write (outvects, (to_bitvector (cmd_addr))) ;
    write (outvects,string' (",Data = ")) ;
    write (outvects, (to_bitvector (D))) ;
    writeline (simresul,outvects) ;
    WR_L   < = '1' ;        wait for 20 ns ;--120ns
    CS_L    < = '1' ;        wait for 10 ns ;--130ns
    A < = "00" ;d_one    < = '0' ;wait for 70 ns ;--200ns
end cmd_write ;
```

这是一个写控制字的通用过程,而且实现不同的功能只需要修改参数。例如在目标为写控制字时参数为:

```
cmd_write (CNTRL,cmd_data,D,A,CS_L,WR_L,d_one,d_in)    CNTRL 代表写控制字
```

而如果写数据的目标地址为 C 口,则参数为:

```
cmd_write (PORTC,cmd_data,D,A,CS_L,WR_L,d_one,d_in)    PORTC 代表写 C 口
```

2. 系统仿真

在将设计下载到实际系统之前可以对设计的逻辑进行校验,以验证设计结果是否达到设计指标要求。这可以通过软件的功能仿真和时序仿真来完成。图 9.2 - 3 是本系统设计的

8255 芯片工作在方式 0,且 A 口、B 口、C 口都为输出的情况下功能仿真的结果。

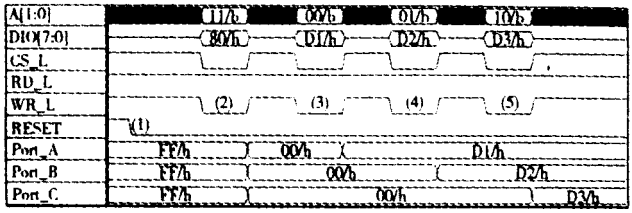


图 9.2-3 8255 设计结果功能仿真图

3. 系统优化

设计实现的最后要进行逻辑综合,即将较高抽象层次的描述自动地转换到较低抽象层次描述的一种方法。目前情况下,就是将 RTL 级的描述转换成门级网表的过程。由于芯片的资源有限,为了优化输出和工艺映射,一定要有相应的约束条件加以控制。以 FPGA 芯片作为目标器件时,设计实体中一部分电路需要尽量压缩面积而忽略性能要求;而另一部分为了满足关键信号的要求及提高性能,则要忽略面积占用。

(1) 面积约束条件

多数逻辑综合工具允许设计者按工艺库中门级宏单元所用的单位来制定面积的约束条件。如果用等效门作为测量单位,则面积约束条件即可以所用门的个数来描述。确定了面积约束条件,在逻辑综合时,综合工具就可利用各种可能的规则和算法,尽可能地减少设计占用的面积。

(2) 时间延时约束条件

时间延时约束条件常用的描述方法是指定输入、输出的最大延时时间。为了对每个所设计的节点进行延时计算,要进行静态分析,即根据网表中每个连接元件的延时模型,对节点进行定时分析并给出最好和最坏的延时情况,然后再检查电路所有的延时约束条件是否满足要求。

速度优化和面积优化的选择在大多数情况下是矛盾的。对速度优化将占用较多资源,而对面积优化又会导致系统速度的降低。这就需要设计者采用一定的优化设计方案,如资源共享、流水线设计和预进位处理。资源共享是通过数据缓冲或多路选择的方法来共享数据通路的工作单元。流水线方法是把一个周期内执行的逻辑操作分成几步较小的操作,并在多个高速时钟内完成。而预进位方式则可以用来减少加法器中进位信号的传输延迟。

综上所述,FPGA 芯片的使用可以有效地缩短系统的开发周期,降低系统成本。VHDL 语言描述和抽象能力强,覆盖面广,用它来开发 FPGA 可以大大减轻设计人员的工作强度,提高设计质量。但要注意必须采用 VHDL 设计进行优化,这对系统实现有很大影响。

参 考 文 献

1 侯伯亨,顾新. VHDL 硬件描述语言与数字逻辑电路设计. 西安:西安电子科技大学出版社,1999
2 王索萍. 电子设计自动化 (EDA) 教程. 成都:电子科技大学出版社,1999
3 赵雅兴. FPGA 原理、设计与应用. 天津:天津大学出版社,1999
4 王毅平. VHDL 编程与仿真. 北京:人民邮电出版社,2000

选自《微型机与应用》月刊,2002 年第 11 期

9.3 VHDL 的设计特点与应用研究

上海交通大学自动化系 (200030) 李 志 田永清 朱仲英

随着科学技术的飞速发展,电子工业界经历巨大的飞跃。集成电路的设计正朝着速度快、性能高、容量大、体积小和微功耗的方向发展,这种变化必将导致集成电路的设计规模日益增大、复杂程度日益增高。而在传统的硬件电路设计方法中,电路的模块和调试通常是在硬件电路设计后期才能进行,一旦考虑不周,硬件电路设计存在较大的缺陷,就有可能重新进行硬件电路的设计,从而使得设计周期增加和浪费大量的人力、物力;设计完成后形成的硬件电路文件主要是电路原理图,如果设计硬件电路的规模较小,那么就需要大量的电路原理图,这将给设计人员阅读和修改硬件设计带来很大的不便。在这种情况下,沿用了十几年的传统硬件设计方法已不能满足需要。一种新的硬件电路设计方法 EDA (电子设计自动化) 应运而生。在 EDA 的设计过程中,除了硬件的行为和功能描述外,其他设计过程都可以用计算机来自动完成。它可以大大节省人力和物力,缩短研制周期,从而增强了设计的实时性,因而 EDA 设计方法得到广泛的应用。

电子设计自动化要求采用硬件描述语言 (HDL) 来进行硬件的行为和功能描述。1981 年,美国国防部提出一种新的硬件描述语言,称为"超高速集成电路硬件描述语言" (VHSIC Hardware Description Language),简称 VHDL。后来经过多次修改和扩充,VHDL 语言成为 IEEE 标准,在世界各地得到广泛的应用。一种硬件描述语言能够成为标准并获得广泛的应用,必然具有许多独有的特点。本文分析了 VHDL 与其他硬件描述语言相比的特点,总结出用 VHDL 设计的流程,并结合实际设计中的一个实例详细叙述流程的各个相关步骤。

一、VHDL 的特点

与其他硬件描述语言相比,VHDL 具有以下五大特点。

1. 功能强大,设计灵活

VHDL 支持各种设计方法,既支持自底向上的设计,也支持自顶向下的设计;既支持模块化设计,也支持层次性设计。VHDL 还支持同步电路,异步电路和随机电路的设计,这是其他硬件描述语言所不能比拟的。

2. 支持广泛,易于修改

由于 VHDL 已经成为 IEEE 标准所规范的硬件描述语言,目前几乎所有的 EDA 工具都支持 VHDL,包括一些像 Synopsys, Cadence 等大公司。这使得用 VHDL 进行的设计具有很强的移植功能;在硬件电路设计过程中,主要的设计文件是用 VHDL 编写的源代码,因为 VHDL 易读和结构模块化,所以易于修改设计。

3. 强大的系统硬件描述能力

VHDL 具有多层次的设计描述功能,既可以描述系统级,又可以描述门级电路。描述形式既可采用行为描述,寄存器传输描述或结构描述,也可以采用三者混合的混合级描述。同时

VHDL 具有丰富的数据类型,既可以支持预定义的数据类型,也可以自己定义数据类型。这样给了硬件描述更大的自由度,使得设计人员能够更方便的设计。

4. 独立于器件的设计

当一个设计描述用 VHDL 模拟器和 VHDL 综合器进行编译,模拟和综合后,可以采用不同的映射工具映射到不同的工艺上去。映射成不同的工艺,只需要改变相应的映射工具,而无需改变 VHDL 设计描述。因此设计人员用 VHDL 进行硬件电路设计时,不需要首先考虑编程器件的具体工艺和结构,而可以将主要精力集中在设计的优化上,这使得设计人员可以面对极其复杂的电路设计。

5. 易于共享与复用

VHDL 采用基于库 (Library) 的设计方法。在设计过程中,可以建立各种可再次利用的模块,一个大规模的硬件电路设计往往不可能从门级电路开始一步步地进行设计,而是一些模块的累加。这些模块可以是一些标准库,也可以是预先设计或以前设计的模块,将这些模块存于库中,就可以在以后的设计中反复使用。这种设计方法可以大大的减少设计工作量,降低设计周期。

二、VHDL 设计流程

一般来说,用 VHDL 进行设计的流程图如图 9.3-1 所示。

由图 9.3-1 可知,一个完整的设计流程可以分为七个步骤。其中编译和仿真在器件编程前就已经完成,并且可以在不同阶段根据结果调整设计,以满足设计要求。这样就解决了前面所述传统电路设计中模拟和调试只有在硬件电路设计后期才能进行的缺陷。图一仅仅给出了 VHDL 设计的总流程,至于这个流程中每一具体步骤的作用和如何实施,将在本文第三部分结合一个具体硬件电路设计具体阐述。

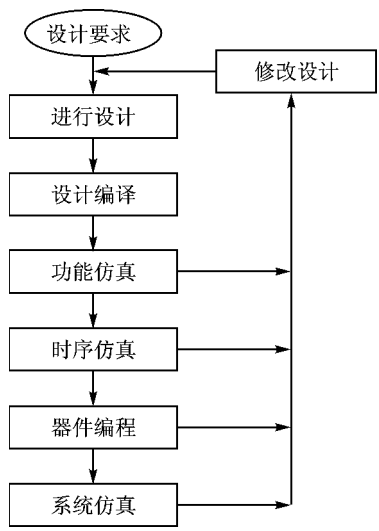


图 9.3-1

三、VHDL 的设计实例

现有一工业打印机控制系统设计,上位机将已经处理好的图形数据传向嵌入式系统 5820,经 5820 的 ISA 总线接口将数据传给控制卡,再由控制卡将数据送往打印头完成打印。在整个过程中,非常重要的一步是 ISA 总线与控制卡的接口电路设计,因为这一接口位于整个数据传输过程的中间位置,对于数据的正确传送起着至关重要的作用。因此下面就以 ISA 总线与控制卡的接口电路为例,详细叙述 VHDL 设计步骤。

1. 第一步 明确设计要求

这是整个流程的第一步,也是完成设计的基础和前提。只有明确了具体的功能要求才能在接下来的设计中有的放矢,有所遵循。经过仔细分析打印速率和分辨率,可知数据传输速率约为 1 M 字节/s。打印头一次打印 32 字节,每次约经过 0.4 mm。对于打印长度为 2 m 的打印机,打印一排数据打印头需运动约 5 000 次。为了保证打印的连续进行,控制卡上带有

SRAM,即嵌入式系统 5820 一次性将打印一排所需数据经 ISA 接口电路送至控制卡的 SRAM;然后再由控制卡将打印头每次打印所需数据送往打印头打印。

由上可知,ISA 与控制卡的接口电路应至少满足以下三点要求:

(1) ISA 与控制卡的接口电路的数据通过率必须≥1 M 字节/s。

(2) 当 5820 已将打印一排所需数据全部送至控制卡的 SRAM 时,5820 应能通过该接口通知控制卡可以开始读取 SRAM 中的数据。

(3) 控制卡经过约 5 000 次将数据全部传给打印头时,控制卡能够通过该接口通知 5820 发送下一排的数据。

2. 第二步 进行设计

在确定了设计方式以后,就可以根据设计要求划分功能模块,然后进行 VHDL 源代码的编写。对于整个设计流程来说,这是最重要的一步,也是最繁琐的一步。为了更好的说明,下面分划分模块和编写代码两部分分别说明。

(1) 划分模块

① ISA 总线最高数据传输速率可以达到 8 M 字节/s,而设计要求只有 1 M 字节/s。为了简化电路设计,ISA 总线采用 8 位 I/O 传送。

② 为了传输数据,特设计一个 8 位的 DATA - REGISTER (数据寄存器),端口地址为 0x341 为了实现 5820 与控制卡之间的相互通信,特设计一个 CS - REGISTER 控制/状态寄存器,端口地址为 0x340。CS - REGISTER 的结构见图 9.3 - 2。

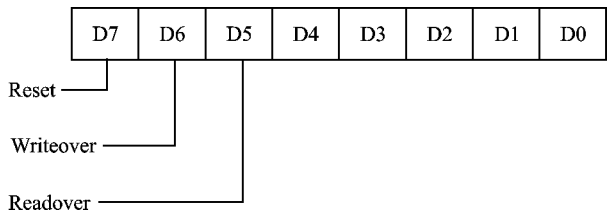


图 9.3 - 2

Reset 为 1 将所有有关的寄存器清空,复位控制卡。

Writeover 为 1 5820 指示控制卡,已将打印一排的数据全部送入 SRAM 中。

Readover 为 1 控制卡已将 SRAM 中的数据全部读出,5820 可以传送下一排数据。

③ ISA 总线地址为 0x340 时选中 CS - REGISTER,总线地址为 0x341 时选中 DATA - REGISTER。为实现寄存器切换,特设计一地址译码器 (PORTSWITCH)。这三者的关系如图 9.3 - 3 所示。图 9.3 - 3 中,SA [9..0] 为 ISA 的低十位地址线;DIN [7..0] 为 ISA 的低八位数据线;ALE 是 ISA 的地址锁存信号线;IOW、IOR 为 ISA 的写,读信号。

(2) 编写 VHDL 代码

由上面确定方案,用 VHDL 分别实现 PORTSWITCH、DATA - REGISTE 和 CS - REGISTER 各个模块的描述。

① PORTSWITCH 的 VHDL 描述如下,共分为应用库说明,实体 Entity 和结构 Architecture 三部分。其中前面四条语句为引用库说明,这部分在 DATA - REGISTER 和 CS - REGISTER 模式代码中同样包含;实体 Entity 中给出了本模块的端口输入输出信号:addr_isa 将与 ISA 总线的低十位地址线相连;ale 与 ISA 总线的地址锁存信号相连,利用它的下降沿锁存 ISA 总线地

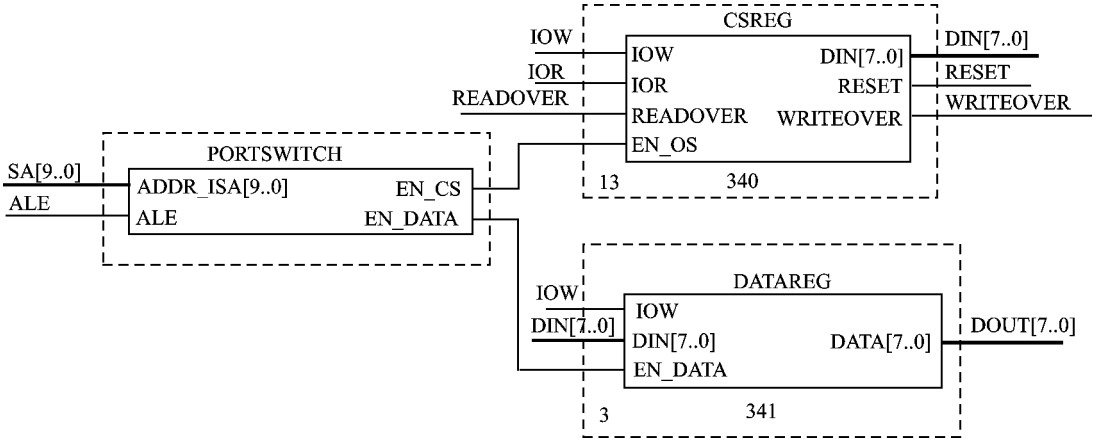


图 9.3-3

址；en_cs 与 DATA-REGISTER 相连，en_cs 为‘1’时，选通 DATA-REGISTER；en_data 与 CS-REGISTER 相连，en_data 为‘1’时，选通 CS-REGISTER。

```
library ieee;
USE ieee. std_logic_1164. ALL;
USE ieee. std_logic_arith. ALL;
USE ieee. std_logic_unsigned. ALL;
ENTITY portswitch IS
    PORT(
      addr_isa      :in std_logic_vector(9 downto 0);
      ale           :in std_logic;
      en_cs         :out std_logic; en_data :out std_logic;
       );
END portswitch;
ARCHITECTURE a OF portswitch IS
signal addr :std_logic_vector(9 downto 0);
  begin
u0 :process(ale)
  begin
    if ale event and ale = '0 then
      addr <= addr_isa;
      end if;
end process u0;
u1 :PROCESS(addr)
  begin
    ifaddr = 340 then
      en_cs <= '1;
      en_data <= '0;
    elsif addr = 341 then
```

```
        en_cs < = '0' ;
        en_data < = '1' ;
      else
        en_cs < = '0' ;
        en_data < = '0' ;
      END IF;
  END process U1;
END a;
```

② DATA－REGISTER 的 VHDL 描述如下,实体 Entity 中给出本模块的端口信号,分别为 din、iow、en_data 和 data,其中 din 与 ISA 总线的低八位数据线相连;iow 与 ISA 总线的写信号相连;当 en_data 为'1',并且 ISA 总线写有效时,数据端口将 ISA 总线上的数据输出到 data 具体源程序略。

③ CS－REGISTER 的 VHDL 描述如下,实体 Entity 中共有七个端口信号,分别为 din、iow、ior、en_cs、reset、readover 和 writeover。其中,din 与 ISA 数据总线相连;iow、ior 分别与 ISA 的写、读信号相连;当 en_cs 为'1',iow 信号有效时,reset 随着 din 的最高位变化;Writeover 随着 din 的次高位变化;ior 信号有效时,din 随着 readover 的高低而显示不同的值,readover 为'1'时,显示"20";readover 为'0'时,显示"00",具体源程序略。

以上源程序均在 MAX + PLUS Ⅱ 软件编译环境中通过运行。

3. 第三步:设计编译

这一阶段,主要实现以下五点功能。

⑴ 处理所有与设计相关的文件;

⑵ 检查代码中的语法错误和常见的设计陷阱;

⑶ 完成逻辑综合和布线;

⑷ 生成用于仿真和时序分析的文件;

⑸ 生成用于器件编程的文件。

从列出的功能可以看出,第二阶段完成的设计可以得到很好的检查,一旦发生错误,应对设计进行相应的调整,修改 VHDL 源代码。

4. 第四步:设计仿真

设计仿真包括流程图中的功能仿真和时序仿真两部分。两者各有特点,功能仿真编译速度快,没有延时,却无法实现完全逻辑综合。一般在实际操作时先进行功能仿真,验证逻辑关系是否正确,然后进行时序仿真,验证延时是否在允许范围内。如果仿真结果与要求不符,应该回到第二步,修改 VHDL 代码。本实例的时序仿真波形图如图 9.3－4 所示。

由图 9.3－4 可知,ISA 总线地址为 0x340,写信号有效时,RESET 和 WRITEOVER 随着 DIN 输入"C0"变高,从而允许 5820 通知控制卡写结束;读信号有效时,ISA 数据总线随着 READOVER 的高低而出现不同的值,高电平时为"20",低电平时为"00",从而使得控制卡可以向 5820 指示读结束。ISA 总线地址为 0x341,写信号有效时,DATA－REGISTER 的输出 DOUT 随着 ISA 总线上的输入数据变化而变化,仿真结果显示,本接口电路设计已满足给定的要求。

5. 第五步:器件编程

器件编程就是将 VHDL 设计描述经过模拟、综合、布局布线的结果,经过一定的映射化成

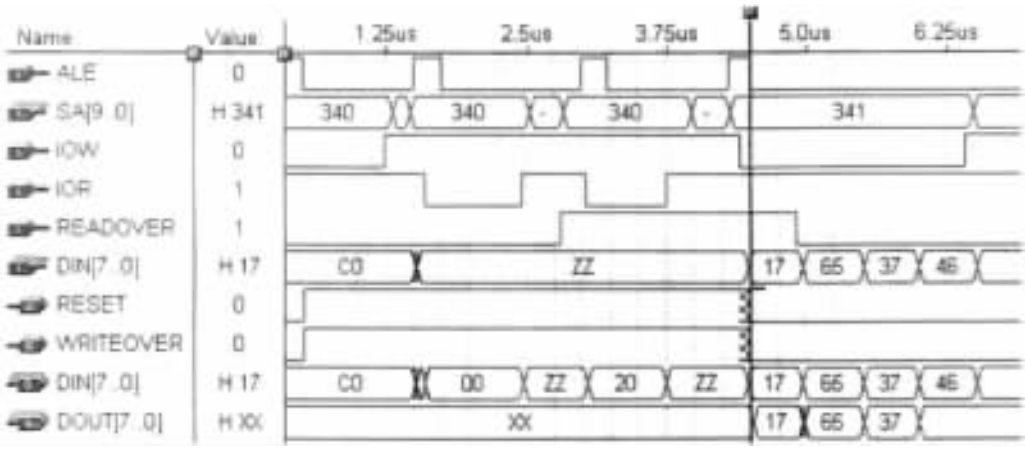


图 9.3－4 ISA 接口电路时序仿真结果

一个器件编程所用的数据文件格式。在使用 MAX + PLUS Ⅱ 软件编译环境时，器件编程即是选择 Altera 的型号合适的芯片下载后缀为. pof 或. sof 的数据文件。

至于器件编程后的系统功能测试，对于用于不同用途的电路有着许多不同的方法，主要检测设计电路与系统中其余部分是否能协同工作。这需要丰富的实践经验，此处就不赘述。

四、结束语

本文在总结出 VHDL 自身的一些特点的基础上，结合一个实际硬件电路设计，比较详细的阐述了利用 VHDL 进行电路设计的流程。由于篇幅的限制，主要着眼于总体框架的把握上，对于 VHDL 语言本身的语法，结构和规则并没有展开介绍。

参 考 文 献

1 姜立冬编著，VHDL 语言程序设计与应用. 北京 北京邮电大学出版社，2001
2 卢毅，VHDL 与数字电路设计. 北京 科学出版社，2001
3 王小军. VHDL 简明教程. 北京 清华大学出版社，1997
4 丁文娟，EDA 技术 ISP 数字系统中的应用. 微型电脑应用，2000
5 刘红. 用 VHDL 设计十六路彩灯控制器. 微型电脑应用，2000

选自《微型电脑应用》月刊，2002 年第 10 期

9.4 单片机应用系统的 CPLD 应用设计

深圳市赋安安全系统有限公司 郑春华 张 杰 李 东

一、引 言

单片机产品的普及促使单片机的应用设计日益广泛。单片机的应用一方面向多单片机系统方向发展以满足大型系统的联网需要,另一方面向嵌入式应用方面发展以满足电子产品短、小、轻、薄的要求。但我们在设计单片机应用系统时,基本上是在最小系统基础上根据实际项目要求进行系统功能扩展。典型的单片机应用系统,一般有程序存储器、数据存储器、键盘扫描电路、显示/指示电路、A/D 电路、D/A 电路、通信电路、打印控制电路等部分。在嵌入式系统和智能化仪器中,功能扩展除了采用通用外围扩展芯片实现外,还可采用可编程外围接口芯片 PSD 与 MCU 组成两片系统,以节省空间,提高系统可靠性。然而,在功能要求较高的单片机应用系统中,由于 PSD 内 SRAM 较小(一般为 2 KB),不能满足使用要求,一般均须外扩 RAM,同时还须扩展 I/O 接口电路以实现控制量的输出、状态量/开关量的输入和其他数字量的输入/输出。此时,PSD 芯片的一些口线资源因被占用而使其利用率降低,并且 PSD 价格也较高,因此,目前使用并不普及。

众所周知,在单片机应用系统中,除了地址译码外,还有一些控制逻辑电路来保证系统正常有序的工作,因此,目前在系统设计时,除了采用标准 TTL/CMOS 逻辑电路外,还普遍采用 GAL 等可编程逻辑器件实现译码和控制逻辑,但 GAL 的资源有限,其应用范围受到一定限制。在 GAL 基础上发展起来的复杂可编程逻辑器件 CPLD (Complex Programmable Logic Device) 具有高集成度和高速度特性,因此,在嵌入式系统设计中使用 CPLD,可使设计的产品达到小型化、集成化和高可靠性。近年来,随着 FPGA 和 CPLD 的迅速发展,其价格已明显下降,开发软件也较易获得,具备了在产品开发中使用的条件。我们在开发产品时,用 CPLD 和 MCU 组成单片机应用系统,根据实际系统需要进行系统重构、在线编程 (ISP)、引脚重定义,符合高集成度、高可靠性、短开发周期、低成本的要求。经多个产品的生产使用,效果很好。

二、MCS-51 系统中的 CPLD 应用设计

1. MCS-51 系统实例

由于 Intel 公司将 8051 MCU 内核向一些大公司进行授权许可,目前可以从 Philips、Atmel、Dallas、SST、Winbond 等公司获得 100 多种 8051 内核的功能不同的单片机来满足设计的需要,8051 成了事实上的工业标准,在国内得到普遍应用。我们设计的系统也是一个 MCS-51 单片机应用系统,包含 64 KB 程序存储器、32KB 数据存储器、1 Mb 的 Flash 存储器、4KB 的 EEPROM、WDT、2 个 RS-485 通信口、1 个 RS-232 通信口、实时时钟电路、串行热敏微型打印机、24 键键盘、三色背光的字符式 LCD 显示器、4 个 LED 指示灯、9 个输出继电器等。我们常用可编程接口器件构成此系统,如用 8279 作键盘和显示控制,2 片 8255 作输入输出控制,2 片 GAL

(16V8,20V8 等) 实现地址译码和逻辑控制、通信切换等,还可使用多片标准逻辑电路如 LS244、LS245、LS374、LS377 等来设计接口电路,实现 I/O 扩展,但器件多必然带来 PCB 面积大、布线较困难、可靠性降低等缺点,同时可编程接口芯片提供的端口数未必与实际需要的端口数相等,容易造成芯片部分资源的浪费。我们采用 Xilinx 公司的 CPLD 芯片 XC9572PC84 构成的系统如图 9.4-1 所示。它资源利用率高、设计灵活、体积小、可靠性高,更重要的是,由于 XC9500 系列的引脚重定义特性,在 PCB 设计时可以根据需要调整引脚位置使走线距离缩短,不仅可减少过孔 (Via),提高线路的抗电磁干扰能力,还能有效地减少 PCB 设计的工作难度和工作量;同时,由于 XC9500 系列的在线可编程功能,可以在不改动 PCB 板的情况下根据特殊工程需要配合软件实现系统现场修改或升级。图 9.4-1 为了节省 PCB 空间和提高可靠性,采用了带 64 KB Flash 的 MCS-51 内核的单片机。这类单片机目前有不少,例如 Philips 公司生产的 P89C51R+和 P89C51RD2、Winbond 公司的 W78E516B 和 W78LE516、台湾 Mosel 公司的 MSU2964 等。Atmel、Dallas、SST 等公司也生产多种不同容量的 Flash MCU,如 89CXX 系列,为我们提供了更多的选择机会。MCU 内的 Flash 主要用于存放该系统的监控软件。

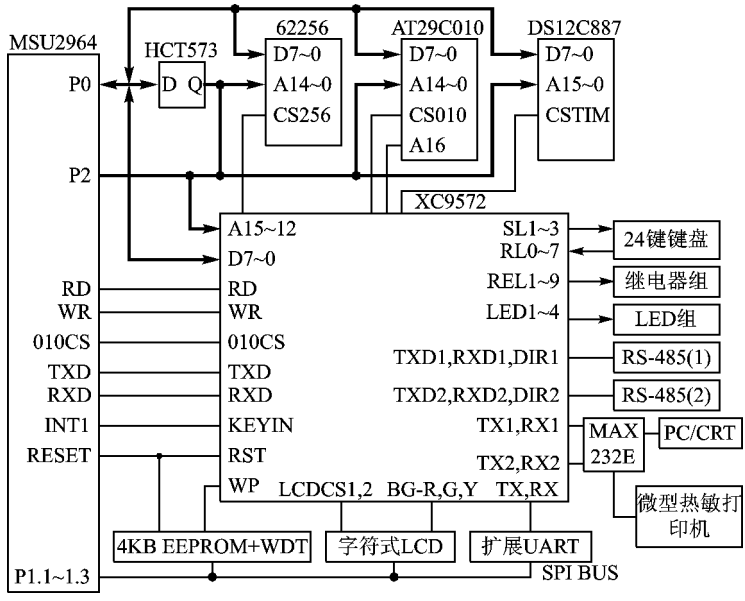


图 9.4-1 采用 CPLD 设计的单片机应用系统实例

2. CPLD 的应用设计

Xilinx 公司的 CPLD 芯片 XC9572 由可编程逻辑功能块 FB (Function Block)、可编程 I/O 单元 I/OB (Input/Output Block)、可编程开关矩阵 FastCONNECT 组成。每个 FB 由 18 个独立的宏单元组成,提供具有 36 个输入和 18 个输出的可编程逻辑容量。I/OB 提供器件输入和输出的缓冲。XC9572 有 72 个宏单元、1600 个门,其详细结构请参阅参考文献 [2]。

单片机有较丰富的指令系统支持,通过程序设计可以灵活实现各种功能。在实际的单片机应用系统中,除了单片机实现系统监控和算法处理程序之外,因单片机 I/O 引脚数量的限制,使得扩展输入/输出口成为系统设计的一个重要组成部分。一般常用可编程并行接口芯片 8255 扩展外围 I/O,使用其基本输入/输出工作模式,将 PA、PB、PC 口定义为输入口,用于接收

外部设备的数据，或定义成输出口控制外部设备。对于 XC9572，其内部功能块（FB）的宏单元（Macrocell）具有可配置成 D 触发器的触发器，可实现寄存器功能，因此，通过宏单元可以对单片机的控制输出信号实现输出锁存，实现 8255 的输出口功能。要实现数据输入，要求数据总线具有三态特性。XC9572 的 I/O 块包括输出使能多路选择器，可由来自宏单元的乘积项（PTOE）或来自全局三态控制（GTS）产生的全局输出使能（Global OE）来产生，因此，可以在 XC9572 中实现 8255 的输入口功能。至于控制逻辑，通过 XC9572 中的开关矩阵 FastCONNECT 和 FB 则可容易实现。在简单分析了用 XC9572 等 CPLD 实现单片机外围器件的扩展的可行性之后，通过分析 MCS－51 单片机的外部数据存储器读写时序，我们设计的 XC9572 与单片机的接口电路如图 9.4－2 所示。XC9572 包含地址译码与地址控制器、键盘电路、通信切换电路、扩展输出口及输入口等部分。在图 9.4－1 中，由于系统有外扩的数据存储器和 Flash 存储器，为节约 CPLD 的口线，地址锁存器 HCT573 没有放在 XC9572 中。

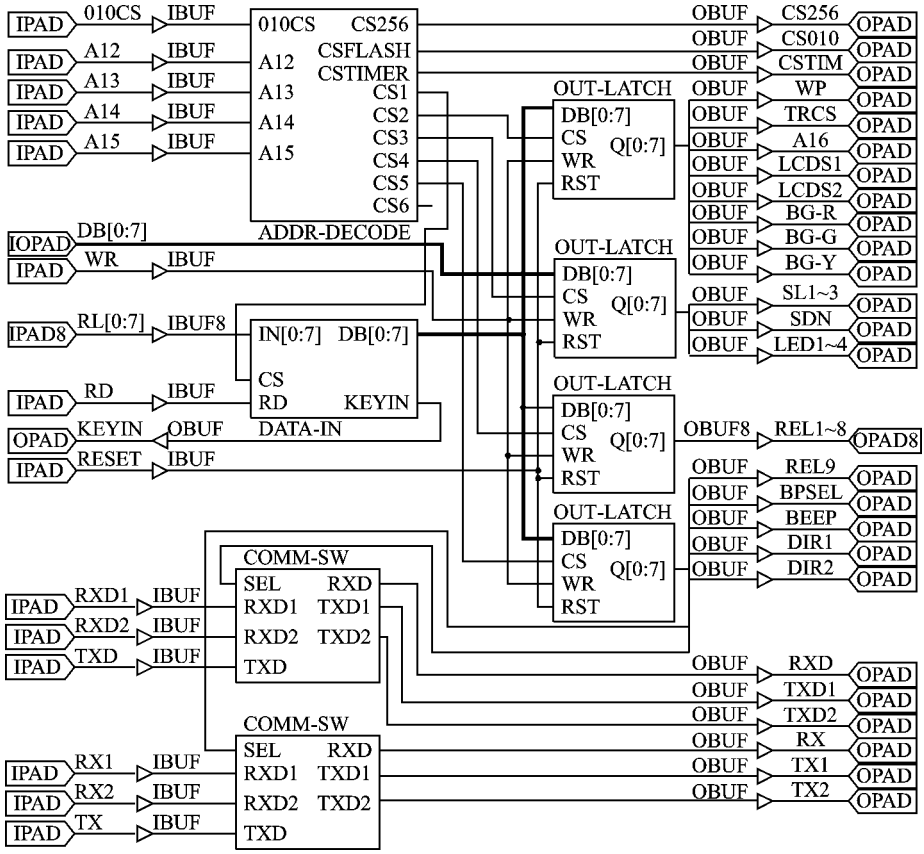


图 9.4－2 单片机系统中的 CPLD 电路实现

在图 9.4－1 中，由于 Flash 容量为 128 KB，其地址空间与 62256、扩展 I/O 相重叠，故在图 9.4－2 中通过 A12 ～A15 和 010CS 实现地址译码与地址控制。地址译码实现外部 RAM 和外扩 I/O 设备的片选，地址控制防止 Flash 与 RAM 及 I/O 口的地址冲突，并且通过 CPLD 扩充一个控制信号 A16，实现对 128 KB Flash 存储器的访问。CPLD 内扩展了 4 个输出口和 1 个输入口。本系统使用 3×8 键盘，键盘电路涉及到键盘扫描输出和键值读入，扫描输出 SL1～SL3 由

XC9572 内扩展的输出口提供,列信号由 XC9572 内扩展的输入口 RL0 ~ RL7 提供,输入口具有"三态"功能。通信切换由多路开关和门电路实现。RS - 485 方向选择信号、通信口的选通信号、LCD 的片选信号、三色背光控制信号、LED 灯控制信号和继电器控制信号也都由 XC9572 内扩展的不同输出口提供。由于 XC9500 系列的输出引脚高达 24 mA 驱动电流,所以,用其扩展输出口时可直接驱动继电器和 LED 灯,但要注意总电流不要超过器件的要求。

Xilinx 公司的 CPLD 开发软件为 Foundation Series 2.1i,支持原理图输入、文本输入、状态图输入和混合输入等输入方法。原理图输入就是使用该软件中已有的符号库文件的元件,如:门电路、触发器、计数器、寄存器等,在原理图编辑器(schematic editor)中以图形的方式连接成所需的电路图,在对其功能仿真(simulation)测试通过后,再通过"执行"(implementation)将所设计的逻辑关系等适配于所选定的 CPLD 中,形成可用于"编程"(programming)下载的标准 JEDEC 文件。原理图设计输入的优点是显示直观、分析方便、易于理解。如图9.4-2中输出扩展口就可利用元件库中的 X74-377 实现。用 CPLD 扩展外设,也可以使用硬件描述语言 VHDL 进行设计。用 VHDL 实现的硬件描述与具体的工艺技术和器件无关,可移植性好、模块可再利用性好、有利于简化设计过程、降低设计风险,应用日渐广泛。

3. 软件设计

在 MCS-51 系列单片机中,扩展的外部 I/O 设备作为数据存储器映像,占用存储器地址,因此,在单片机程序设计时,对于 CPLD 中扩展的输入和输出外设,只要 CPU 给出正确的 I/O 口地址,就能像访问数据存储器一样访问所带的设备,进行读/写操作。如对于图 9.4-2 电路,要将外设的数据(如键盘值)读入单片机中,先选中设备地址(图中为 A000H),然后用 MOVX A,@DPTR 即可,即:

```
MOV   DPTR,#0A000H
MOVX  A,@DPTR
```

而输出数据到外设,可给设备分配一个寄存器存储外设状态,在须对外设内容修改时先修改该寄存器的值,然后选中地址,用 MOVX @DPTR,A 命令将寄存器内容输出到外设。如:将地址 C000H 的 bit4 位置 0,其余位状态不变(原值已读入 R6 中),则:

```
MOVA,R6
ANDA,#0EFH
MOVDPTR,#0C000H
MOVX@DPTR,A
```

三、CPLD 选用和使用

目前,生产 PLD 器件的公司很多,如 Xilinx、Altera、Atmel、Vantis、Lattice 等,各公司的产品在性能、价格、逻辑规模和封装以及对应的开发软件等方面各有所长。因此,在开发产品选用 CPLD 时,应对器件的容量、速度、功耗、接口要求、引脚数目等进行综合考虑,根据项目的要求作出最合理的选择。

(1) 容量和结构。每种系列的 CPLD 都有不同容量的产品可供选择,如 XC9500 系列 CPLD,就有 XC9536、XC9572、XC95108、XC95144、XC95216、XC95288 等规格可供选用。而每一种器件又都有多种封装:对于 XC9572,有 44 脚 PLCC(PC44)、84 脚 PLCC(PC84)、100 脚 PQFP(PQ100)、100 脚 TQFP(TQ100)等封装形式;同样,每一种封装又都有多种容量的器件:

如对于 PC44,有 XC9536、XC9572 可选择,对于 PC84,有 XC9572、XC95108 可选择,因此,在选用器件时应考虑到这些因素。目前,CPLD 的系列化使其有从 1000 门到数万门逻辑资源的器件可供选用。为便于扩展和修改设计,在选择器件时,应先估测一下所需的功能资源,对所选器件的逻辑单元和引脚数最好留有一定数量的余量,一般用到其资源的 80 % 较好。

⑵ 速度问题。对于某种器件的某一种封装形式,一般有不同速度等级的芯片,以满足不同速度要求的需要。目前,CPLD 的引脚到引脚的时延已达 ns 级(如 XC9500 系列,其最高工作频率可达 125 MHz,所有引脚到引脚的传输延迟为 7.5 ns)。芯片速度的选择应与所设计系统的最高工作速度相匹配,以避免高速器件对外界微小毛刺的灵敏反应而使系统处于不稳定的工作状态。并不是器件速度越高越好。对于 XC9500 系列 CPLD,每个 I/O 引脚有独立可编程的输出电压摆率控制位,可配置成高摆率输出(fast)或低摆率输出。在速度要求不高的情况下,通过编程使输出沿的转换率变慢可减少 PCB 上线间干扰和地线上的毛刺,减少系统噪声。

⑶ 功耗问题。目前,CPLD 的工作电压越来越低,如:Xilinx 公司的 CPLD 就有 5 V 工作的 XC9500 系列,3.3 V 工作的 XC9500XL 系列,2.5 V 工作的 XC9500XV 系列。因此,在要求低功耗,高集成度的便携式或手持式设备中,可选用低电压系列产品,也可选用 Xilinx 新推出的极低功耗 CPLD——CoolRunner。另外,XC9500 系列也对宏单元提供低功耗方式,可由用户对除关键部件外的其他部分编程为低功耗运行模式,以减少系统功耗。

⑷ 电平问题。使用 XC9500 系列 CPLD 可以方便地实现系统的电平配合。如在 5 V 系统中,将 V_{CCINT} 和 V_{CCIO} 连接到 5 V 即可;在 5 V/3.3 V 的混合系统中,将 V_{CCINT} 连接到 5 V,将 V_{CCIO} 连接到 3.3 V 即可使用 3.3 V 的外部设备。在多电源混合应用系统中这一特性特别有用,可以节省大量的电平转换器。

⑸ 应尽量选用 ISP (In-System Programmable) 器件,并具有系统内电擦除功能,以保证所设计的系统可随时进行修改和硬件升级。ISP 技术可以使在 PCB 中不用取下 CPLD 器件,而在所设计的目标系统中通过 JTAG 接口对 CPLD 器件进行编程或者反复改写,在现场对系统进行逻辑重构和升级,使硬件随时能够改变组态,实现硬件设计的软件化。

⑹ 输入引脚不要悬空,不用的引脚应接地或电源。XC9500 系列的每个 IOB 提供用户编程接地引脚能力,因此,通过编程设定器件内部不用的引脚接地,可以减少因大量瞬时转换输出产生的系统噪声。

⑺ 芯片内的时序电路应有“上电”复位电路,保证开机加电后,时序电路处于初始状态。

⑻ 在 CPLD 的电源和地线间必须并联 1 个 1.0 μF 和 1 个 0.1 μF 的电容进行电源滤波和去耦,以提供稳定的无噪声电源。

四、结束语

由于 CPLD 是纯硬件结构,不会存在单片机的因外部干扰而使程序跑飞和死机等问题,具有极强的抗干扰能力,所以,在单片机应用系统设计中使用 CPLD,可以将单片机较强的逻辑控制和数据处理能力与 CPLD 的高集成度、高可靠性和高速度结合起来,使所设计的嵌入式系统或智能仪表具有较高的性价比。在单片机应用系统中使用 CPLD 进行系统设计,具有很大的灵活性,可减少系统芯片数量,缩小系统体积,提高系统可靠性。同时,先进的 EDA 工具可以减少设计周期和开发费用,加快产品上市时间,通过功能和时序仿真可以降低设计风险,通

过对器件的加密可以保护开发者的知识成果。在单片机系统中采用 ISP 技术的 CPLD,通过好的设计方案还可能实现远程硬件升级或远程逻辑重构。随着 SoC 技术的发展和 IP 核(core)的日益推广,相信 CPLD 会在很多方面得到普遍应用。

参 考 文 献

1 Xilinx 公司. Foundation Series 2.1i CDROM
2 孟宪元. 可编程 ASIC 集成数字系统. 北京:电子工业出版社,1998
3 何立民. 单片机应用系统设计. 北京:北京航空航天大学出版社,1990
4 侯伯亨,等. VHDL 硬件描述语言与数字逻辑电路设计. 西安:西安电子科技大学出版社,1997
5 李华. MCS-51 系列单片机实用接口技术. 北京:北京航空航天大学出版社,1993

选自《单片机与嵌入式系统应用》月刊,2002 年第 7 期

9.5 用 CPLD 实现单片机与 ISA 总线接口的并行通信

武汉空军雷达学院二系新装备维修教研室 (430010)
肖小峰 盛 文 李演仁

CPLD (Complex Programmable Logic Device) 是一种复杂的用户可编程逻辑器件,由于采用连续连接结构,易于预测延时,从而使电路仿真更加准确。CPLD 是标准的大规模集成电路产品,可用于各种数字逻辑系统的设计。近年来,由于采用先进的集成工艺和大批量生产,CPLD 器件成本不断下降,集成密度、速度和性能大幅度提高,一个芯片就可以实现一个复杂的数字电路系统;再加上使用方便的开发工具,使用 CPLD 器件可以极大地缩短产品开发周期,给设计修改带来很大方便[1]。本文以 ALTERA 公司的 MAX7000 系列为例,实现 MCS51 单片机与 PC104ISA 总线接口的并行通信。采用这种通信方式,数据传输准确高速,在 12 MHz 晶振的 MCS51 单片机控制的数据采集系统中,可以满足与 PC104 ISA 总线接口实时通信的要求,通信速率达 200 Kbps。

一、系统总体设计方案

用 CPLD 实现单片机与 PC104ISA 总线接口的并行通信。由于 PC104 主要完成其他方面的数据采集工作,只是在空闲时才能接收单片机送来的数据,所以要求双方通信的实时性很强,但数据量不是很大。因此在系统设计中,单片机用中断方式接收数据,PC104 采用查询方式接收数据。系统设计方案如图 9.5-1 所示。

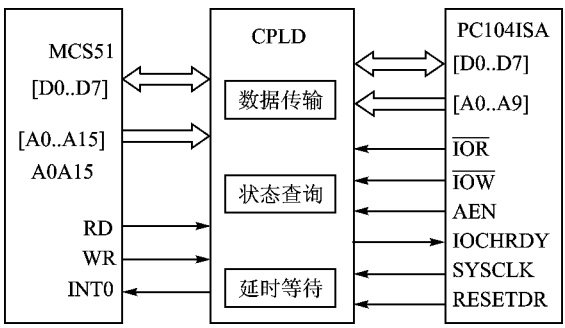


图 9.5-1 系统设计方案

在单片机部分,D [0..7] 是数据总线,A [0..15] 是地址总线,RD 和 WR 分别是读写信号线,INT0 是单片机的外部中断,当单片机的外部中断信号有效时,单片机接收数据。

在 CPLD 部分,用一片 MAX7000 系列中的 PM7128 ESLC84 来实现,用来完成 MCS51 与 PC104ISA 总线接口之间的数据传输、状态查询及延时等待。

在 PC104ISA 部分,只用到 PC104 的 8 位数据总线 D [0..7],A [0..9] 是 PC104 的地址总线;$\overline{\text{IOW}}$和$\overline{\text{IOR}}$是对指定设备的读写信号;AEN 是允许 DMA 控制地址总线、数据总线和读写命

令线进行 DMA 传输以及对存储器和 I/O 设备的读写；IOCHRDY 是 I/O 就绪信号，I/O 通道就绪为高，此时处理机产生的存储器读写周期为 4 个时钟周期，产生的 I/O 读写周期和 DMA 字节传输均需 5 个时钟周期，MCS51 通过置此信号为低电平使 CPU 插入等待周期，从而延长 I/O 周期；SYSCLK 是系统时钟信号，使系统与外部设备保持同步；RESETDR 是上电复位或系统初始化逻辑信号，是系统总清信号。

二、基于 MAX + plusll 的硬件实现

ALTERA 公司的 CPLD 开发工具 MAX + plusII，支持多种输入方式，给设计开发提供了极大的方便，因此本系统采有 MAX + plusII 进行设计。系统的主体部分用原理图输入方式，由于库中提供了现成的芯片，所以使用很方便。原理图输入部分如图 9.5－2 和图 9.5－3 所示。图 9.5－2 主要完成单片机与 ISA 接口通信中的数据传输和握手判断。在图 9.5－2 中，各信号说明如下：

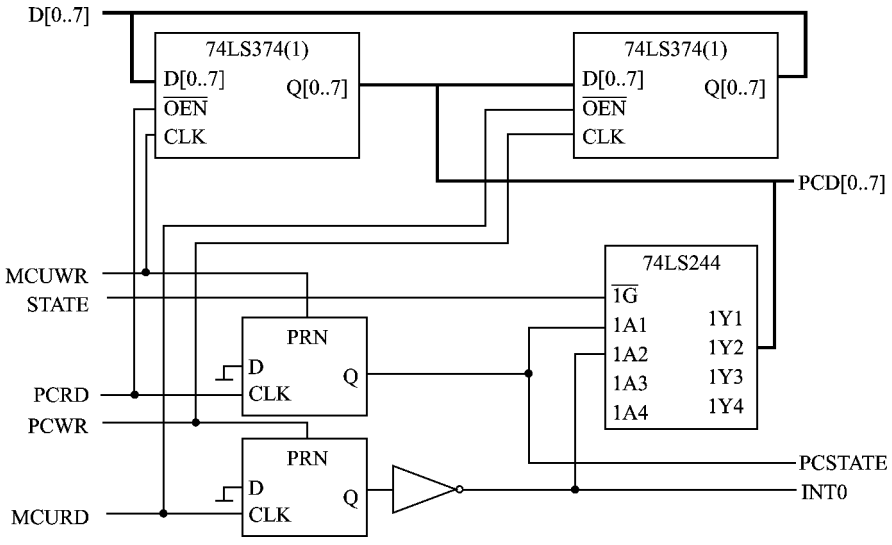


图 9.5－2　MCS51 与 PC104ISA 握手逻辑图

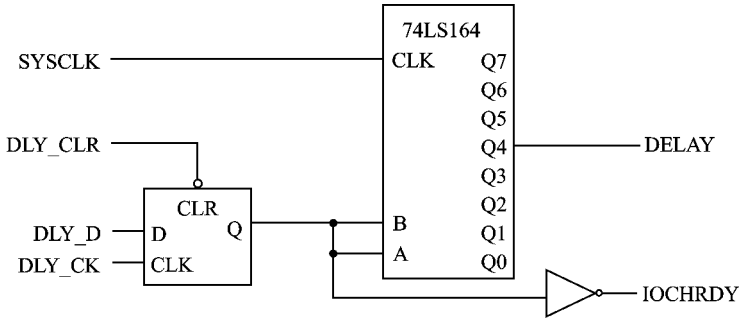


图 9.5－3　延时电路

D[0..7] 单片机的 8 位双向数据总线；
PCD[0..7] ISA 接口的 8 位双向数据总线；
PCRD ISA 接口的读有效信号；
PCWR ISA 接口的写有效信号；
STATE ISA 接口的查询选通信号，用来判断单片机已写数据或读走数据；
PCSTATE 单片机用此查询 ISA 接口已取走数据；
MCURD 单片机的读有效信号；
MCUWD 单片机的写有效信号；
INT0 单片机的外部中断信号。

当 MCUWR 信号有效时，单片机把数据锁存于 74LS374 (1) 中，此时 PCSTATE 变为高电平；PC104 用 STATE 信号选通 74LS244 来判断数据位 PCD0 是否为高电平，如果为高，说明单片机送来了数据，则使 PCRD 有效，从数据锁存器 74LS374 (1) 中取走数据，此时 PCSTATE 变为低电平，单片机通过判断此信号为低电平来判定 PC104 已取走了数据，此时可以发下一个数据。

当 PCWR 信号有效时，PC104 把数据锁存于 74LS374 (2) 中，此时 INT0 变为低电平；单片机产生外部中断，使 MCURD 信号有效，从数据锁存器 74LS3734 (2) 中取走数据。此时 INT0 变为高电平，PC104 用 STATE 信号选通 74LS244 判断数据位 PCD1 是否为高电平，如果为高电平，则说明单片机取走了数据，可以发送下一个数据。

PC104 与单片机进行通信，最关键的就是速度匹配问题。由于 PC104 的速度快，而单片机的速度较慢，所以要在 PC104 的 IOCHRDY 处插入等待周期。如图 9.5－3 所示，各信号说明如下：

IOCHRDY 用来使 ISA 接口等待 5 个时钟周期；
DLY_D 延时输入信号；
DLY_CK 延时等待时钟信号；
DLY_CLR 等待清除信号，为开始下一次送数周期做准备；
DELAY 延时 5 个时钟周期后的输出信号，作为 DLY_CLR 信号的输入；
SYSCLK ISA 接口的系统时钟信号。

在 MCS51 与 PC104 进行通信的过程中，DLY_D 信号一直有效（高电平），在信号 SYSCLK 的作用下，每 5 个时钟周期 DELAY 信号有效一次，即为高电平。此时 DLY_CLR 信号有效（低电平），IOCHRDY 信号变为高电平，PC104 可以读写数据。

地址译码部分采用文本输入方式，用 ALTERA 公司的硬件设计开发语言 AHDL（Altera Hardware Description Language）实现。AHDL 是一种模块化的高级语言，完全集成于 MAX + plus II 系统中，特别适合于描述复杂的组合逻辑、状态机和真值表，地址译码部分采用文本输入方式充分体现了文本输入方式的优点。文本输入内容如下：

```
SUBDESIGN          Address
(
PCA[9..0]          :INPUT;
AEN,IOR,IOW        :INPUT;
RESETDR,DELAY      :INPUT;
```

```
A [15..14]          :INPUT;
RD,WR               :INPUT;
DLY_D               :OUTPUT;
DLY_CK              :OUTPUT;
DLY_CLR             :OUTPUT;
STATE               :OUTPUT;
PCRD                :OUTPUT;
PCWR                :OUTPUT;
MCURD               :OUTPUT;
MCUWR               :OUTPUT;
)
BEGIN
!DLY_CLR = RESETDR#DELAY;
DLY_D = !AEN& (PCA [9..1] = = H"110");
DLY_CK = !AEN& (PCA [9..1] = = H"110") & (!IOR#!IOW);
!PCWR = !AEN& (PCA [9..0] = = H"220") & !IOW;
!PCRD = !AEN& (PCA [9..0] = = H"220") &IOR;
!STATE = !AEN& (PCA [9..0] = = H"221") & !IOR;
!MCSWR = (A [15..14] = = H"2") & !WR;
END;
```

说明:PCA [9..0] 是 PC104 的地址信号,A [15..14] 是单片机的地址信号,PC104 用到端口地址 220H 和 221H。

三、通信软件设计

PC104 是基于 ISA 总线的,在系统软件设计中要防止地址冲突。PC104 中使用 A0 ~ A9 地址位来表示 I/O 端口地址,即可有 1024 个口地址,前 512 个供系统板使用,后 512 个供扩充插槽使用。当 A9 = 0 时表示为系统板上的口地址;当 A9 = 1 时表示扩充插槽接口卡上的口地址[2]。因为本设计中采用保留的口地址 220H 和 221H,保证不会发生地址冲突。

在本程序中,PC104 采用查询方式接收数据,单片机采用中断方式接收数据。

```
#define pcreadwrite 0x220          ;PC104 读写数据口地址
#define pcrdstate 0x221            ;PC104 查询状态口地址
```

PC104 写数据函数:

```
void pcwrite (int port,unsigned char ch)
{    outportb (pcreadwrite,ch);
while ((inportb (pcrdstate) &0x02) != 0x02) 等待单片机读走数据
{ }
}
```

单片机读子程序:

```
MCLLR :MOV DPTR,#400H
```

```
        MOVX A,@DPTR
        RETI
```

PC104 读数据函数：

```
unsigned char pcread (int port)
{     while ((inportb (pcrdstate) &0x0) != 0x01)        等待单片机写数据
   { }
  return inportb (pcreadwrite) ;
}
```

单片机写子程序：

```
MCUWR :MOV DPTR,#8000H
       MOVX  @DPTR,A                          等待 PC104 读写数据
RET
```

参 考 文 献

1 李冬梅. PLD 器件与 EDA 技术. 北京 北京广播学院出版社,2000

2 王士元. C 高级实用程序设计. 北京 清华大学出版社,1996

选自《电子技术应用》月刊,2002 年第 8 期

9.6 FPGA 实现 PCI 总线接口技术

国防科技大学电子科学与工程学院 ATR 实验室
郭天天 卢焕章 常 青

PCI (Peripheral Component Interconnect) 是一种高性能的局部总线,采用高度综合优化的总线结构,保证系统各部件之间的运行可靠,目前广泛应用于各种计算机系统中。PCI 总线可同时支持多组外围设备,具有很高的数据传输速率,峰值传输速率可达 132 MB/s (32 位、33 MHz)。

目前开发 PCI 接口大体有两种方式:一是使用专用的 PCI 接口芯片,二是使用可编程器件。如果使用厂家提供的专用接口芯片,用户可能只使用到它的部分功能,会造成一定的资源浪费,而且专用芯片价格高,不经济。而使用可编程器件比前者具有以下两个优点:一方面用户可以根据需要设计 PCI 接口,不会浪费资源;另外一方面用户逻辑和接口部分可以做在同一个器件内,PCI 接口和用户逻辑会结合得更紧密。现在已经有越来越多的用户使用可编程器件,如 FPGA、CPLD 等进行 PCI 设备的开发。对可编程器件进行 PCI 接口开发的研究具有很强的实际意义。

PCI 接口的实现包括配置空间和数据接口两方面。

一、PCI 总线配置空间的实现

1. 配置空间

PCI 总线定义了 3 种物理地址空间:存储器地址空间、I/O 地址空间和配置地址空间。前两者是普通的计算机系统地址空间,而配置空间是 PCI 所特有的一种空间。根据 PCI 总线规范[1],所有的 PCI 设备都必须提供配置空间。定义 PCI 配置空间的目的在于:提供一套适当的配置措施,使之满足现行的和可预见的系统配置机构。

配置空间是一长度为 256 字节并且有特定记录结构或模型的地址空间,可以在系统自举时访问,也可在其他时间访问。该空间分为首部区和设备有关区两部分。设备在每个区中只须实现必要的和与之相关的寄存器。首部区的长度为 64 字节,每个设备都必须支持该区的寄存器分配,这个区包括进行设备识别的惟一标示码,和允许对设备进行控制的若干区域。表 9.6-1所列为 64 字节首部区的布局情况。

表 9.6-1 配置空间首部区

字节 4	字节 3	字节 2	字节 1	偏移地址
设备代码		供应商代码		00h
状态		命令		04h
类别代码			修改版本	08h
内含自测试	首部类型	延时计数	Cache 大小	0ch

续表 9.6－1

字节 4	字节 3	字节 2	字节 1	偏移地址
基地址寄存器 0～5				10h 14h 18h 1ch 20h 24h
保留				28h 2ch
保留				
扩展 ROM 基地址寄存器				30h
保留				34h 38h
保留				
Max－Lat	Min－Gnt	中断引脚	中断线	3ch

表中颜色较深部分为强制性的首部寄存器,包括供应商代码寄存器、设备代码寄存器、命令寄存器、状态寄存器、修改版本寄存器、类别寄存器和首部类型寄存器。供应商代码由 PCI SIG 分配,设备代码由设备制造商分配;设备代码和供应商代码一起用于定位设备指定的驱动程序。命令寄存器用于存放 PCI 命令;状态寄存器包含 PCI 的状态信息。修改版本寄存器包含版本标识号;类别代码寄存器是只读寄存器,用来说明设备的通用功能。首部类型寄存器包含两个字段,D6～D0 位定义配置空间首部区字节 10 H～13 H 的格式,D7 位表示该 PCI 设备是多功能还是单功能设备,如果为 1 表示是多功能设备,反之则是单功能设备。

除了上面所说的强制性首部类型寄存器以外,还有一些寄存器是经常用到的,比如基地址、中断线和中断引脚寄存器等。基地址寄存器提供了一种为设备指定存储空间或 I/O 空间的机制。操作系统在启动的时候要判断系统中有多少存储器、系统中的 I/O 设备需要多少地址空间,然后根据得到的结果,自动配置系统的存储空间和 I/O 空间,实现设备无关管理。基地寄存器中的值有两个含义:一是存储空间或 I/O 空间的基地址;二是存放定义空间的长度。它的 D31 位用来区分是基地址还是长度,如果为 1 则是基地址,反之是长度;另外 D0 位用来区分存储器空间和 I/O 空间,如果为 1 则是 I/O 空间,反之是存储器空间。

另外如果要实现中断,必须对中断线寄存器和中断引脚寄存器进行设置。中断线寄存器由系统中的所有中断源共同使用,它的值说明设备的中断引脚连接到中断控制器的哪个输入上。中断引脚寄存器说明设备使用 PCI 总线上的哪一个中断引脚。

2. 实现方法

我们以一实例说明如何在 FPGA 中实现 PCI 总线的配置空间。系统要求为:实现一个 PCI 从设备,把设备上的存储器资源映射到系统的 I/O 空间,大小为 16 字节;实现中断功能(单中断)。

配置空间的实现,首先是实现强制性寄存器,然后根据需要实现一个或多个可选寄存器。在本例系统中,要求把设备上的资源映射到 I/O 空间,所以要实现一个基地址寄存器;另外要实现中断功能就还需要实现中断线寄存器和中断引脚寄存器。在本例中要实现的配置寄存器包括所有强制性寄存器以及一个基地址寄存器、中断线寄存器和中断引脚寄存器。

在 FPGA 中资源有限，如果配置空间全部用 FPGA 内部的寄存器实现要占用很多资源，其他功能势必要受到削弱，并且会带来布局、布线上的麻烦。所以对一些只读配置寄存器或者不需要修改的配置寄存器，我们可以采用硬连线的方式实现，比如强制性寄存器当中的设备代码、供应商代码、命令、状态、类别码、修改版本、首部类型寄存器，以及可选寄存器当中的中断引脚寄存器都可以用硬连线的方式实现。至于基地址寄存器和中断线寄存器，因为系统在启动过程中要写入分配的起始地址和中断号，所以必须用寄存器实现。其他不用实现的配置寄存器给一个初始值 0 就可以了。这样 64 字节的配置空间首部，我们只需要用 40 个寄存器以及若干硬连线就可以实现，大大节省了资源。具体方案如图 9.6－1 所示。

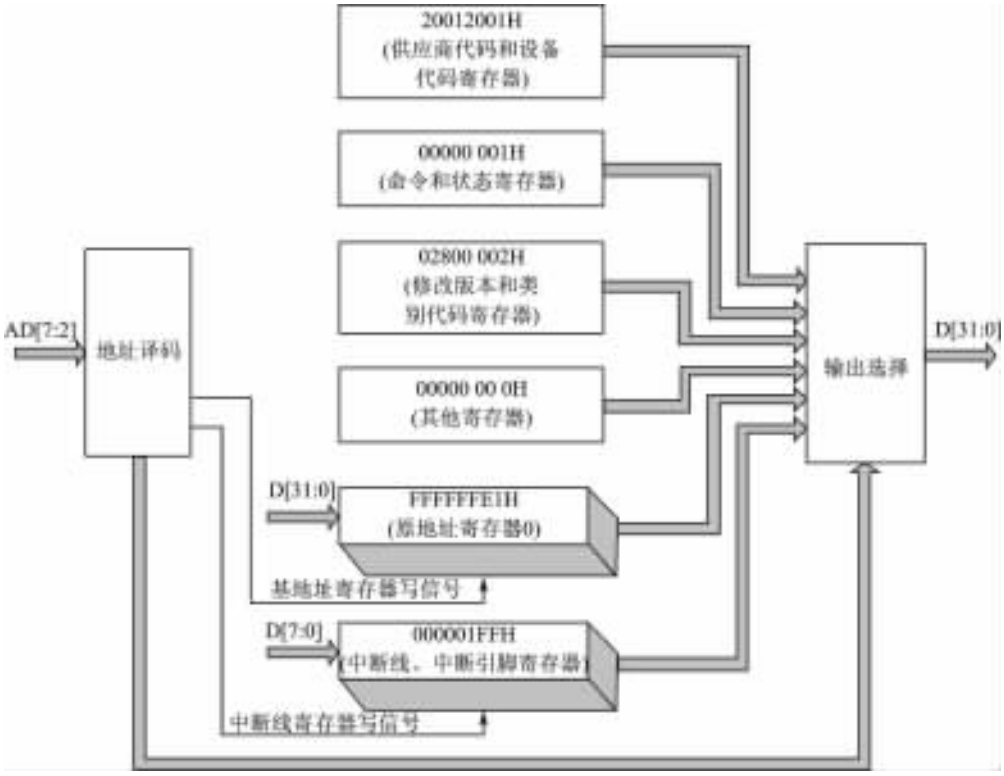


图 9.6－1 配置空间实现框图

图中基地址寄存器由 32 个 D 触发器实现，中断线寄存器由 8 个 D 触发器实现，其他寄存器都为硬连线实现。输出选择由三态门完成。AD [7 :2] 在配置交易时为配置寄存器号。各个寄存器值的确定请参看参考文献 [1]。

采用这种方式我们在设备上实现了要求的 PCI 配置空间，占用资源很少，并且可以利用 FPGA 的可编程性很容易实现不同的配置空间。

二、PCI 总线数据传输接口设计

1. PCI 总线读写时序

PCI 总线上的基本传输机制是突发分组传输，一个突发分组由一个地址周期和一个（或多个）数据周期组成。PCI 支持存储空间和 I/O 空间的突发传输，它所有的数据传输基本上都是

由以下三条信号线控制的：FRAME#、IRDY#和 TRDY#。

当数据有效时，数据资源需要无条件设置 xRDY#信号（写操作为 IRDY#，读操作为 TRDY#）。接受方可在适当时间发出它的 xRDY#信号。FRAME#信号有效后的第一个时钟上升沿是地址周期的开始，此时传送地址信息和总线命令。下一个时钟上升沿开始一个（或多个）数据周期，每当 IRDY#和 TRDY#同时有效时，所对应的时钟上升沿使数据在主、从设备之间传送。在此期间，可由主设备或从设备分别利用 IRDY#和 TRDY#的无效而插入等待周期。

PCI 总线的读时序如图 9.6－2 所示。

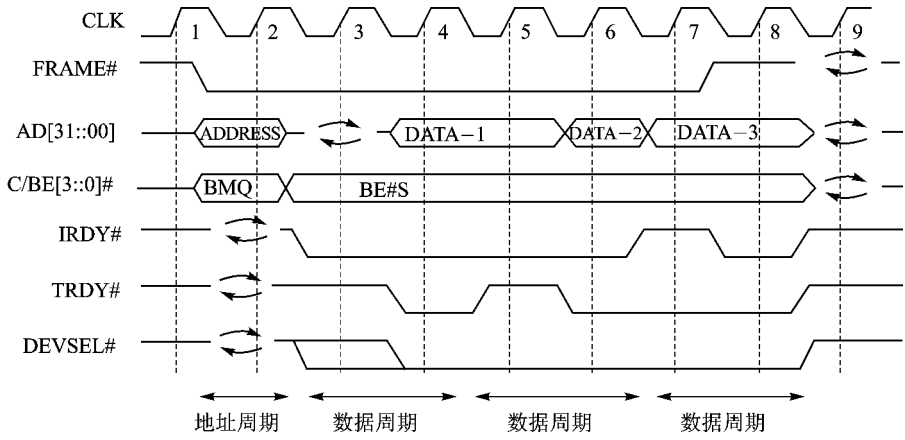


图 9.6－2 PCI 读操作时序

当 FRAME#信号有效时，读传输开始，在 AD [31]总线上保持有效的地址信号，同时 C/BE# [3]上包含有效的总线命令。如果总线命令为 0110（存储器读命令）或者 0010（I/O 读命令），同时 AD [31]总线上的地址又在目标设备的地址范围内，该设备将设置 DEVSEL#信号有效。然后主设备停止驱动 AD [31]总线，置 IRDY#为低，表明主设备准备好接收数据。第一个数据节拍产生于第三个时钟周期。第二个时钟周期是总线转换周期，目的是为了避免主设备和从设备因竞争总线发生冲突。在数据节拍里，主设备在每个时钟周期的上升沿检查 TRDY#信号，如果它无效表示从设备没有准备好，主设备自动插入等待周期；反之将传送数据，完成一个数据节拍。当主设备使 FRAME#信号无效，表示当前是最后一个数据节拍。在数据节拍里，C/BE [3]总线上为字节允许信号。

DEVSEL#信号和 TRDY#信号是由从设备提供的，但必须保证 TRDY#信号在 DEVSEL#信号之后出现。数据的真正传输是在 IRDY#和 TRDY#信号同时有效的时钟上升沿进行的，这两个信号中的一个无效表示插入等待周期，此时不进行数据传输。例如在时钟 7 处，尽管是最后一个数据周期，但由于 IRDY#无效，FRAME#不能变为无效，只有在时钟 8 处 IRDY#有效后，FRAME#信号才能撤消。

PCI 总线的写操作时序如图 9.6－3 所示。

PCI 总线上的写操作与读操作相类似，FRAME#信号有效表示地址周期的开始，且在时钟 2 的上升沿处达到稳定有效。从图中可看出，主设备在时钟 5 处因撤消了 IRDY#而插入等待周期，表明要写的数据将延时发送，但此时字节使能信号不受等待周期的影响，不得延迟发送。从设备在时钟 5、时钟 6 和时钟 7 插入 3 个等待周期。AD [31]总线在地址周期后没有像读

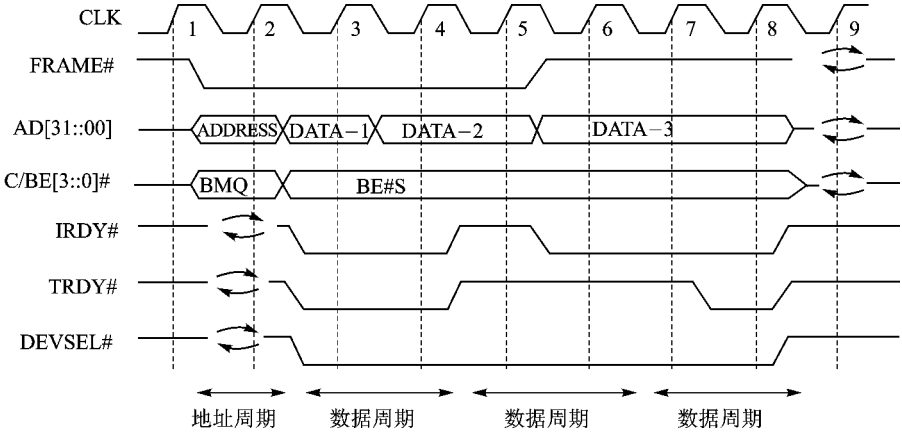


图 9.6－3 PCI 写操作时序

操作那样插入转换周期,这是因为地址和数据都是由主设备发出的。

2. 接口设计

接口的设计还是以上述系统为例。设计思路如下:在时钟的上升沿采样 FRAME#、地址和命令,如果 FRAME#有效则译码地址和命令,如果总线命令为 001x,并且总线上的地址在目标地址范围内,表明这是对本设备的 I/O 操作;或者总线命令为 101x,且 IDSEL 信号有效,表明这是对本设备配置空间的操作。在这两种情况下,根据总线命令的最后一位确定是读操作还是写操作,有效 DEVSEL#和 TRDY#信号,开始数据传输;并在传输过程中采样 FRAME#和 IRDY#信号,确认最后一个数据周期,无效 DEVSEL#和 TRDY#信号,结束数据传输。图 9.6－4 为接口电路的设计框图。

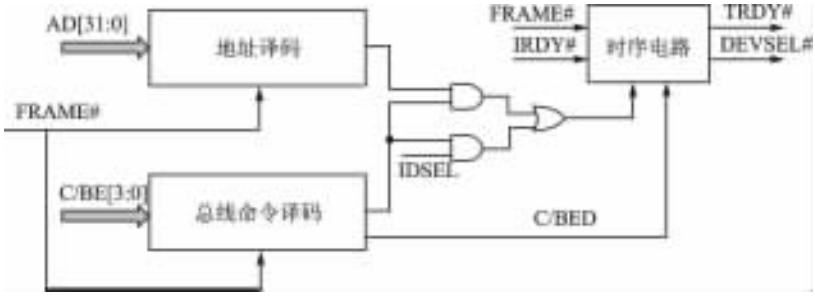


图 9.6－4 读写接口电路框图

从图 9.6－5 的仿真波形看,完全符合 PCI 规范要求的时序。读操作与此类似,这里我们就不再给出波形了。在实际应用中,这个接口电路也得到了检验,不论是单周期还是突发数据传输都能很好地完成。

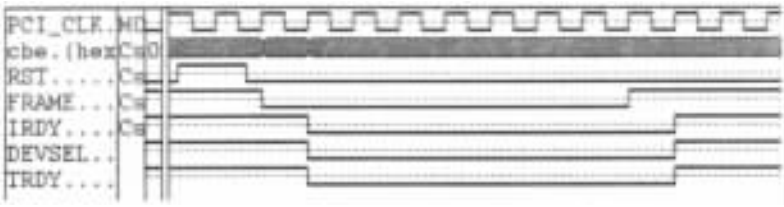


图 9.6－5 接口电路写操作仿真波形

三、结束语

本文介绍了在 FPGA 上实现 PCI 接口技术，并且在自制的多媒体通信卡上得到了应用。我们使用的 FPGA 器件是 Xilinx 公司的 XCS20，PCI 接口部分占用的资源为 55 个 CLB，256 个 TBUF，相当于 1543 个逻辑门。经测试通信卡的速度可以达到 32 MB/s，完全满足应用的要求。这种 PCI 接口实现技术较简单、占用资源少、具有很强的灵活性，并且可以很方便地移植到其他可编程器件上，具有较强的实用价值。

参 考 文 献

1 Tom Shanley，Don Anderson. PCI 系统结构第四版. 电子工业出版社，2000

2 陈利学，孙彪，赵玉连. 微机总线与接口设计. 电子科技大学出版社，1998

3 Xilinx. The Programmable Logic Data Book. 1999

选自《电子产品世界》月刊，2002 年第 4 期

9.7 用 FPGS 实现 DES 算法的密钥简化算法

深圳艾克斯耐特网络设备公司 李世强

DES 密码算法被广泛应用,但它的算法用软件实现需要的时间较长。所以在对运算时间有要求的场合,一般都用硬件实现。很多情况下需要用 FPGA 来实现 DES ;而我的体会是 DES 的密钥计算是用硬件实现 DES 的关键。

一、DES 的密钥算法

第一步 :舍弃原始密钥 K 中的所有奇偶校验位,经换位选择 1 处理,如表 9.7 - 1 所列,变成 $C^{(0)}$、$D^{(0)}$ 两个各 28 位的数据区组,其中:

表 9.7 - 1 换位选择 1

$C^{(0)}$							
	57	49	41	33	25	17	9
	1	58	50	42	34	26	18
	10	2	59	51	43	35	27
	19	11	3	60	52	44	36

$D^{(0)}$							
	63	55	47	39	31	23	15
	7	62	54	46	38	30	22
	14	6	61	53	45	37	29
	21	13	5	28	20	12	4

① $C^{(0)} = c_1^{(0)} c_2^{(0)} \cdots c_{28}^{(0)} = k_{57} k_{49} \cdots k_{36}$

② $D^{(0)} = d_1^{(0)} d_2^{(0)} \cdots d_{28}^{(0)} = k_{63} k_{55} \cdots k_4$

第二步 :$C^{(0)}$、$D^{(0)}$ 各作 $\sigma^{(1)}$ 位左循环移位,如表 9.7 - 2 所列,得

$$C^{(1)} = \lambda\sigma^{(1)} C^{(0)}$$

$$D^{(1)} = \lambda\sigma^{(1)} D^{(0)}$$

表 9.7 - 2 对应不同 i 的左循环移位位数

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$\sigma^{(i)}$	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

第三步 :拼接 $C^{(1)}$、$D^{(1)}$,得

$$E^{(1)} = e_1^{(1)} e_2^{(1)} \cdots e_{56}^{(1)} = c_1^{(1)} c_2^{(1)} \cdots c_{28}^{(1)} d_1^{(1)} d_2^{(1)} \cdots d_{28}^{(1)}$$

用换位选择 2,如表 9.7 - 3 所列,从中选出 48 位,形成第一次迭代使用的加密密钥。

$$K^{(1)} = k_1^{(1)} k_2^{(1)} \cdots k_{48}^{(1)} = e_{14}^{(1)} e_{17}^{(1)} \cdots e_{32}^{(1)}$$

表 9.7－3 换位选择 2

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

第四步:用同样方法递推地产生第 i 次 (i = 2,…,16) 迭代的密钥。

二、密钥算法的化简分析

可以看出如果用 HDL (硬件描述语言) 按照上面的描述一步一步地推导出 16 次 DES 迭代的密钥,不仅语言描述麻烦,占用很多的硬件资源,而且每一个密钥需要的运算时间也不同,会给 DES 的密码迭代运算带来很多麻烦。

如果仔细观察就会发现,密钥的获得只是将输入的原始密钥经过换位和不同次数的循环移动,而且每一次 DES 迭代需要的密钥 Ki (0 < i < 17) 进行循环移位的次数对于原始密钥来说是固定的。这就是说,我们需要的密钥 Ki 的每一个 bit,相对原始密钥 K 的每一个 bit 一定有固定的关系,找到了这种关系就能用简单的方法实现密钥的计算。

这里用 C 语言编写了一个程序,找到了这种关系。程序的设计思路是用数组模拟密钥的位组。数组中的每一个存储单元表示所需密钥的一个 bit,数组中存储的数据是所需密钥的 bit 对应的原始密钥的 bit 序号。将数组经过相应的置换和循环移位后,就可以得到所需密钥和原始密钥在位上的对应关系。

三、DES 密钥预计算的 C 语言源程序

```
#include <stdio.h>

void main(void)
{
void GetKeyVal(char KeySel);
int KeySel = 1;
//--输入需要的密钥号----------
while(KeySel > 0&&KeySel <= 16) {
    printf("please enter key code");
    scanf("%d",&KeySel);
    GetKeyVal(KeySel);
}
}
//-------------------------------------------------------------------------
/* 函数名称:GetKeyVal                    */
```

```
//-------------------------------------------------------------------------
/* 入口:KeySel   如 KeySel = 1,表示第一次迭代需要的密钥,依此类推 */
/* 出口:无                                                    */
/* 功能:计算所需密钥和原始密钥的关系并在显示器上输出 */
//-------------------------------------------------------------------------
void GetKeyVal (char KeySel)
{
int C [28],D [28]; //原始密钥的初始置换结果
int E [56]; //循环左移后的结果
int K [48]; //所需密钥的对应结果
int Loop Time; //得到所需密钥的左移次数
int tempL,tempR,i,j; //临时变量
//----------------------------原始密钥的初始置换----------------------------------
  C [0] = 57;C [1] = 49;C [2] = 41;C [3] = 33;C [4] = 25;C [5] = 17;
  C [6] = 9;C [7] = 1;C [8] = 58;C [9] = 50;C [10] = 42;C [11] = 34;
  C [12] = 26;C [13] = 18;C [14] = 10;C [15] = 2;C [16] = 59;
  C [17] = 51;C [18] = 43;C [19] = 35;C [20] = 27;C [21] = 19;
  C [22] = 11;C [23] = 3;C [24] = 60;C [25] = 52;C [26] = 44;
  C [27] = 36;
  D [0] = 63;D [1] = 55;D [2] = 47;D [3] = 39;D [4] = 31;D [5] = 23;
  D [6] = 15;D [7] = 7;D [8] = 62;D [9] = 54;D [10] = 46;D [11] = 38;
  D [12] = 30;D [13] = 22;D [14] = 14;D [15] = 6;D [16] = 61;
  D [17] = 53;D [18] = 45;D [19] = 37;D [20] = 29;D [21] = 21;
  D [22] = 13;D [23] = 5;D [24] = 28;D [25] = 20;D [26] = 12;
  D [27] = 4;
  //----------------------------计算相应的左循环移位次数----------------------------
  switch (KeySel) {
      case 1:LoopTime = 1;break;
      case 2:LoopTime = 2;break;
      case 3:LoopTime = 4;break;
      case 4:LoopTime = 6;break;
      case 5:LoopTime = 8;break;
      case 6:LoopTime = 10;break;
      case 7:LoopTime = 12;break;
      case 8:LoopTime = 14;break;
      case 9:LoopTime = 15;break;
      case 10:LoopTime = 17;break;
      case 11:LoopTime = 19;break;
      case 12:LoopTime = 21;break;
      case 13:LoopTime = 23;break;
      case 14:LoopTime = 25;break;
      case 15:LoopTime = 27;break;
      case 16:LoopTime = 28;break;
```

```
default：break；
  }
  //----------左循环移位----------
  for (j = 0 ;j < LoopTime ;j + +) {
    tempL = C [0] ;
    tempR = D [0] ;
    for (i = 0 ;i < 27 ;i + +) {
    C [i] = C [i + 1] ;
    D [i] = D [i + 1] ;
    }
     C [27] = tempL;
     D [27] = tempR;
}
for (i = 0 ;i < = 27 ;i + +) {
  E [i] = C [i] ;
  E [28 + i] = D [i] ;
}
  //----------压缩置换----------
K [0] = E [14 - 1] ;K [1] = E [17 - 1] ;K [2] = E [11 - 1] ;K [3] = E [24 - 1] ;
K [4] = E [1 - 1] ;K [5] = E [5 - 1] ;
K [6] = E [3 - 1] ;K [7] = E [28 - 1] ;K [8] = E [15 - 1] ;K [9] = E [6 - 1] ;
K [10] = E [21 - 1] ;K [11] = E [10 - 1] ;
K [12] = E [23 - 1] ;K [13] = E [19 - 1] ;K [14] = E [12 - 1] ;K [15] = E [4 - 1] ;
K [16] = E [26 - 1] ;K [17] = E [8 - 1] ;
K [18] = E [16 - 1] ;K [19] = E [7 - 1] ;K [20] = E [27 - 1] ;K [21] = E [20 - 1] ;
K [22] = E [13 - 1] ;K [23] = E [2 - 1] ;
K [24] = E [41 - 1] ;K [25] = E [52 - 1] ;K [26] = E [31 - 1] ;K [27] = E [37 - 1] ;
K [28] = E [47 - 1] ;K [29] = E [55 - 1] ;
K [30] = E [30 - 1] ;K [31] = E [40 - 1] ;K [32] = E [51 - 1] ;K [33] = E [45 - 1] ;
K [34] = E [33 - 1] ;K [35] = E [48 - 1] ;
K [36] = E [44 - 1] ;K [37] = E [49 - 1] ;K [38] = E [39 - 1] ;K [39] = E [56 - 1] ;
K [40] = E [34 - 1] ;K [41] = E [53 - 1] ;
K [42] = E [46 - 1] ;K [43] = E [42 - 1] ;K [44] = E [50 - 1] ;K [45] = E [36 - 1] ;
K [46] = E [29 - 1] ;K [47] = E [32 - 1] ;
//----------输出结果----------
printf ("key %d is \n",KeySel) ;
 for (i = 0 ;i < = 47 ;i + +) {
     if (K [i] > = 10)          printf ("%3d", K [i] ) ;
     else                       printf ("%3d", K [i]) ;
     if ( (i + 1) %8 = = 0)     printf ("  :%d\n",i + 1 ) ;     //换行显示
 }
}
```

四、密钥的 HDL 简化计算的实现

将上面的程序编译后执行,屏幕上会分行(每 8 个数字 1 行)显示出 48 个数字。这 48 个数字就是所需密钥所对应的原始密钥的位关系。简单地用 HDL 语言将计算所得的位关系表示出来,即可得到所需的密钥。

如输入 key code 为 1 时屏幕会显示:

10 51. 38

…

. . . . . 31 48

这就表示所需 DES 运算的第一次迭代的密钥 K1 的第 1 位对应原始密钥 K 的第 10 位,第 2 位对应原始密钥的第 51 位……第 48 位对应原始密钥的第 31 位。

用 HDL 简单地表示出这种对应关系就可以得到相应的密钥。比如,可以用 Verilog 表示为

```
    //K1 表示第一次迭代的密钥,K 表示原始密钥
module subkey1 (K,K1) ;
input [1 :54] K ;              //原始密钥
output [1 :48] K1 ;            //第一次迭代的密钥
    assign K1 [1] = K [10] ;
    assign K1 [2] = K [51] ;
    ……
    assign K1 [48] = K [31] ;
endmodule
```

综上所述,可以看出这种方法非常简单地实现了密钥的计算,极大地节约了硬件实现的资源开销。能够简单实现密钥计算的关键在于能够发现迭代所需密钥相对于原始密钥有固定的对应关系,然后用预计算进行化简。

选自《单片机与嵌入式系统应用》月刊,2002 年第 2 期

9.8 可编程模拟器件原理与开发

西安电子科技大学电子工程学院 (710071)
赵曙光 陈丽萍 殷廷瑞 赵明英

可编程模拟器件 (Programmable Analog Device) 是近年来崭露头角的一类新型集成电路。它既属于模拟集成电路,又同可编程逻辑器件一样,可由用户通过现场编程和配置来改变其内部连接和元件参数从而获得所需要的电路功能。配合相应的开发工具,其设计和使用均可与可编程逻辑器件同样方便、灵活和快捷。与数字器件相比,它具有简洁、经济、高速度、低功耗等优势;而与普通模拟电路相比,它又具有全集成化、适应性强,便于开发和维护 (升级) 等显著优点,并可作为模拟 ASIC 开发的中间媒介和低风险过渡途径。因此,它特别适用于小型化、低成本、中低精度电子系统的设计和实现,未来其应用将会日益广泛。

一、内部结构与基本原理

通用型可编程模拟器件主要包括现场可编程模拟阵列 (FPAA) 和在系统可编程模拟电路 (ispPAC) 两大类。二者的基本结构均与可编程逻辑器件相似,主要包括可编程模拟单元 (Configurable Analog Block,CAB)、可编程互连网络 (Programmable Interconnection Network)、配置逻辑 (接口)、配置数据存储器 (Configuration Data Memory)、模拟 I/O 单元 (或输入单元、输出单元) 等几大部分,如图 9.8-1 所示。模拟 I/O 单元等与器件引脚相连,负责对输入、输出信号进行驱动和偏置;配置逻辑通过串行、并行总线或以在系统编程 (ISP) 方式,接收外部输入的配置数据并存入配置数据存储器;配置数据存储器可以是移位寄存器、SRAM 或者非易失的 E^2PROM、FLASH 等,其容量可从数十位至数千位不等;可编程互连网络是多输入、多输出的信号交换网络,受配置数据控制,完成各 CAB 之间及其与模拟 I/O 单元之间的电路连接和信号传递;CAB 是可编程模拟器件的基本单元,一般由运算放大器或跨导放大器配合外围的可编程电容阵列、电阻阵列、开关阵列等共同构成。各元件取值及相互间连接关系等均受配置数据控制,从而呈现不同的 CAB 功能组态和元件参数组合,以实现用户所需的电路功能。CAB 的性能及其功能组态和参数组合的数目,是决定可编程模拟器件功能强弱和应用范围的主要因素。

数模混合可编程器件可看作是可编程模拟器件的推广形式。以 SIDSA 公司 (www.sidsa.com/fipsoc) 的 FIPSOC 系列 (数模混合现场可编程片上系统) 为例,它既包含有模拟的可编程单元和互连网络,又包含有由逻辑宏单元和开关矩阵组成的 FPGA,还包含有 A/D、D/A 转换器和用于配置与控制的嵌入式微处理器等,可用于片上系统 (SOC) 的开发与实现。但其模拟部分的规模较小,主要面向数据采集、实时监控等特定应用。

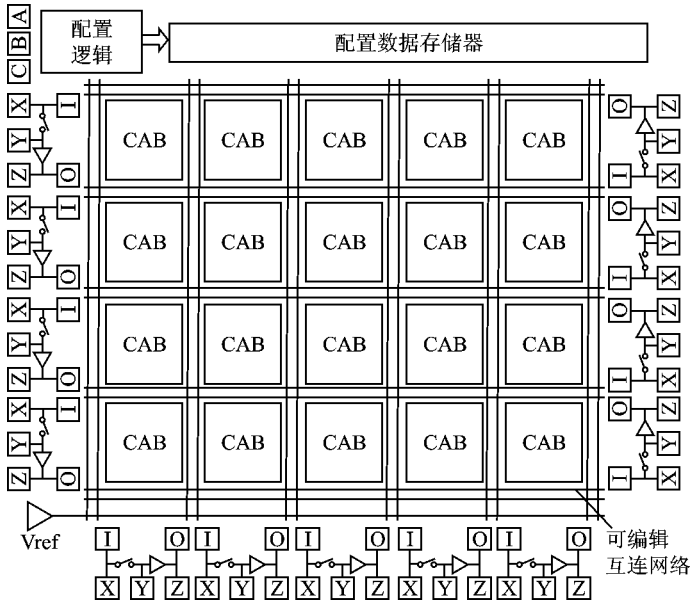


图 9.8－1 可编程模拟器件结构简图

二、基本开发流程

可编程模拟器件开发的主要步骤依次为：

⑴ 电路表达，即根据设计任务，结合所选用的可编程模拟器件的资源、结构特点，初步确定设计方案；

⑵ 分解与综合，即对各功能模块进行细化，并利用开发工具输入或调用宏函数自动生成电原理图；

⑶ 布局布线，即确定各电路要素与器件资源之间的对应关系以及器件内部的信号连接等，可自动或手动完成；

⑷ 设计验证，即对设计进行仿真（根据器件模型和输入信号等，计算并显示电路响应），以初步确定当前设计是否满足功能和指标要求，如果不满足，应返回上一步骤进行修改；

⑸ 由开发工具自动生成当前设计的编程数据和文件；

⑹ 器件编程，即将编程数据写入器件内部的配置数据存储器，一般通过在线配置方式完成，也可利用通用编程器脱机编程；

⑺ 电路实测，即利用仪器对配置后的器件及电路进行实际测试，详细验证其各项功能和指标，如果发现问题，还需返回前面有关步骤加以修改和完善。

可编辑模拟器件设计的基本流程图如图 9.8－2 所示。

该流程主要在微机上利用开发工具完成，基本可做到“所见即所得”。以往由于元件超差、接触不良等实际因素造成的延误和返工可基本消除，对设计者的要求也大大降低。

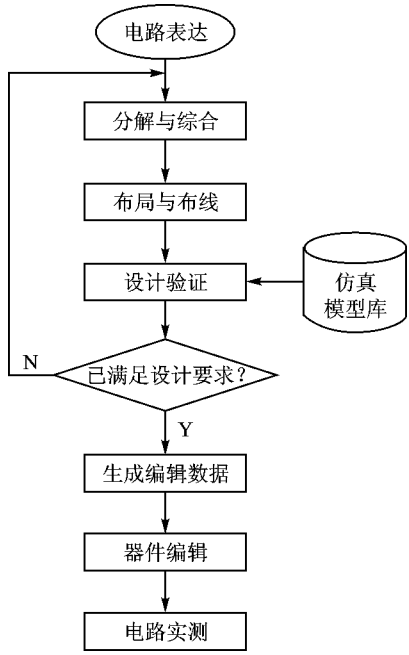


图 9.8－2 可编程模拟器件设计的基本流程图

三、主流器件与核心技术

FAS 公司（http ：//www. zetex. com）的 TRAC 系列现有 TRAC020、TRAC020LH（微功耗版本）、ZXF36Lxx（模拟门阵列）等器件，采用电压运算技术——以随时间连续变化的模拟电压为信号参量。其 CAB 由运放配合电阻、电容、多路模拟开关等组成，可编程互连网络也主要利用模拟开关实现。利用配置数据控制多路模拟开关即可改变 CAB 的内部连接（即功能组态）；改变一组按特定规律取值的同类元件（电阻或电容）之间的连接关系，获得所需的等效元件取值 ;改变各 CAB 间的信号传递关系等。

该系列具有接近常规器件的优良特性（如闭环带宽可达 12 MHz），面向模拟计算的器件结构和便于向 ASIC 移植的产品线。其 CAB 具有加（ADD）、取负（NEG）、对数（LOG）、反对数（ANT）、积分（AUX－def）、微分（AUX－int）等运算型功能组态，设计者可根据设计目标的数学描述或信号流图，利用开发工具以绘制框图方式完成电路设计而无须考虑其内部细节。缺点是可编程能力较弱，器件内部连接基本固定（参见图 9.8－3），仅能利用 NIP（直通）和 OFF（断开）功能组态或外部连接线（Link）等加以改变 器件内电阻等元件均取值固定，须外接 RC 元件来改变有关的电路参数。设计过程的自动化程度和电路的整体集成度也因而降低。

Lattice 公司的 ispPAC 系列等采用跨导运算技术，以模拟电流作为主要信号参量，以跨导运算放大器（OTA）取代电压运算放大器，以基于 OTA 的有源元件取代部分无源元件。该类器件利用 D/A 转换器按照配置数据改变 OTA 的偏置电流，从而改变其互导增益 g_m 和电压放大器增益 A_u，实现对 CAB 的配置和参数调整。由于在 IC 中易于改变且调整范围较大，控制精确较高，因此该类器件的参数变化范围和分辨率均可显著提高。此外，该类器件还具有电流模电路共有的高速、低电压、低功耗、宽动态范围、高稳定性等优点。

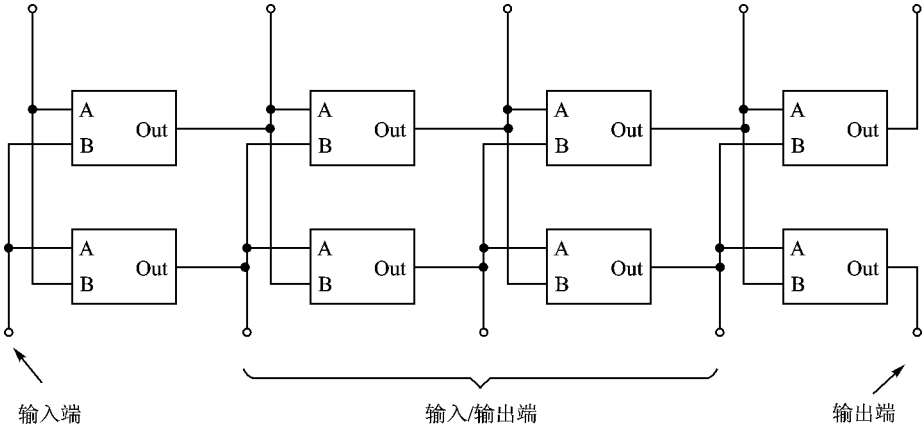


图 9.8－3 TRAC 系列内部结构示意图

ispPAC 系列包括 PAC10、PAC20、PAC30 等通用型器件和 PAC80、PAC82 等 ISP 滤波器。以 PAC10 为例（参见图 9.8－4），其可编程模拟单元（PAC Block）以两个增益可配置（±1 ~ ±

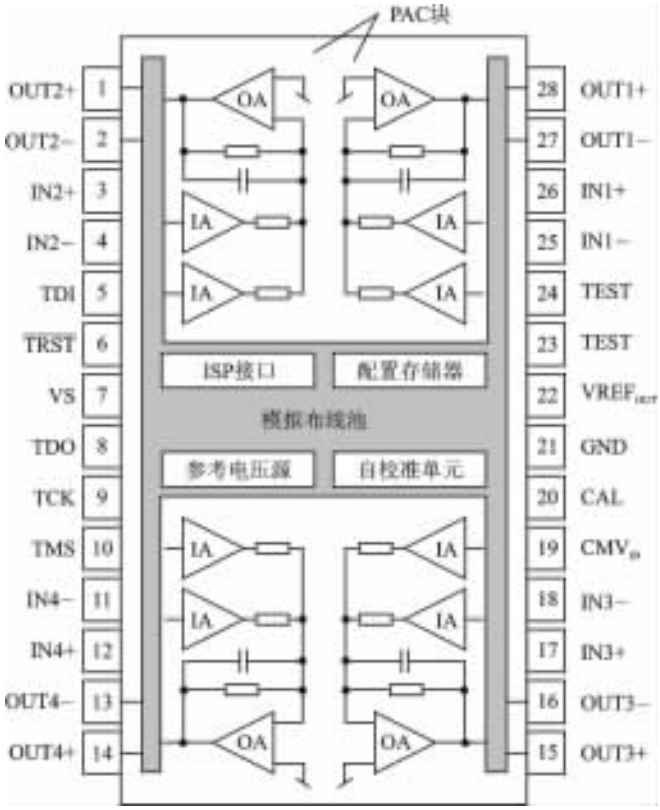


图 9.8－4 ispPAC10 内部结构简图

10）的跨导型仪表放大器作为输入级，以运放、有源反馈元件（跨导放大器）和电容阵列（7 个电容可组合出 128 种等效电容）等构成输出级，可实现放大、迭加、积分和滤波等功能且精度较高；其模拟布线池可灵活地配置器件内部及其与引脚之间的连接关系；自校准单元可自动测

量输出失调并利用专用 DAC 加以补偿；ISP 接口支持在系统编程和数据保密。因此，ispPAC 的电路性能与可编程能力俱佳。PAC20 等还配有 DAC 和迟滞比较器，仅需单片便可构成简单的监控系统。

Anadigm 公司（www. anadigm. com）的 AN10E40 器件则采用开关电容技术（同 MOTOROLA 原产的 MPAA020），通过改变电容比或开关电容的时钟频率来配置电路参数。其内部为典型的阵列式结构（参见图 9. 8－4），由 CAB、模拟 I/O 单元和分布其间的布线资源及可编程时钟资源等组成，信号带宽约为 250 kHz。其 CAB 由运放、电子开关和开关电容等组成（参见图 9. 8－5），对信号来源、去向和各电容容量（均有 256 种选择）等均可灵活配置。可编程时钟资源则为各开关电容提供所需的时钟频率（共 32 种分频比）和相位（每种频率 4 种）。这样，单个 CAB 即可实现整流器、放大器、可编程比较器和一阶滤波器等信号调理功能；将多个 CAB 加以组合、连接，便可实现高阶滤波器、脉宽调制器等更为复杂的电路。由于现有 IC 工艺可制造的电阻和电容范围有限且误差较大，而电容比的制造精度较高（＜0. 1%），因此该类器件的电路精度较高，可编程能力较强而制造成本较低，但信号带宽较小，内部噪声较大。

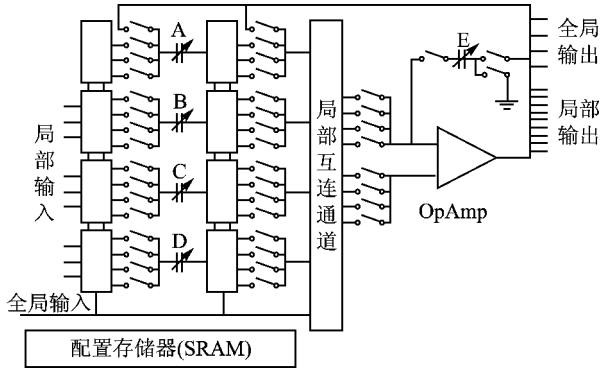


图 9. 8－5 AN10E40 的 CAB 结构

此外，一旦低成本的可编程电流镜或模拟乘法器研制成功，具备兼容数字 IC 工艺等多种优势的开关电流技术便可应用于可编程模拟器件，极大地降低其成本并提升其性能。

目前，可编程模拟器件已在数据采集、信号处理、仪器仪表、控制与监测、人工神经网络、电路实验等重要领域得到应用，其典型应用包括信号调理、模拟计算、中高频应用、人工神经网络、电路进化设计（EHW）等。尽管可编程模拟器件问世不久，有关的技术与产品仍显稚嫩，但其内在的便利性和经济性以及作为其数字域对应物的可编程逻辑器件的成功经历，都使我们有理由相信：在不远的将来，可编程模拟器件的技术必将日益成熟，器件品种必将日益丰富，最终将成为模拟电路设计和应用中的首选器件。

参 考 文 献

1 赵曙光，殷庭瑞，赵旺英. 可编程模拟器件原理、开发及应用. 西安：西安电子科技大学出版社，2002

选自《电子技术应用》月刊，2002 年第 12 期

9.9 数字/模拟 ISP 技术及其 EDA 工具

南京东南大学仪器科学与工程系 (210096) 李宏生

一、在系统可编程技术

在系统可编程是指在用户自己设计的目标系统中或线路板上为重构器件逻辑进行编程或反复编程的能力。这一技术在 20 世纪 90 年代由 Lattice 公司首先提出,并应用于其可编程逻辑器件 ispLSI (In-System Programmable Large Scale Integrated circuit) 中。目前许多公司的 CPLD/FPGA 产品都支持 ISP 技术。

ISP 技术的应用,给仪器仪表系统的设计带来了革命性的变化。应用 ISP 器件,可以很早就完成 PCB 板的制作,并把器件焊在电路板上。需要器件完成什么样的功能只需在计算机上进行设计,然后通过编程电缆把设计方案下载到器件中,瞬间即可完成器件的重配置和重编程。它使得仪器仪表的硬件系统不再是固定结构,而具有软件的灵活性。在调试过程中,通过不断更改"软件"就可以达到硬件功能的改进,这种"软"硬件的全新设计概念,使系统具有了极强的灵活性和适应性。

二、可编程逻辑器件

20 世纪 70 年代,人们发明了可编程逻辑器件 PLD。最早的可编程逻辑器件是可编程只读存储器 PROM,随后又相继出现了可编程阵列逻辑器件 PAL (Programmable Array Logic) 和通用阵列逻辑器件 GAL (Generic Array Logic)。PAL 器件由一个可编程的"与"平面和一个固定的"或"平面构成,或门的输出可以通过触发器有选择地被置为寄存状态。PAL 器件是现场可编程的,它的实现工艺有反熔丝技术、EPROM 技术和 EEPROM 技术。还有一类结构更为灵活的逻辑器件是可编程逻辑阵列 (PLA),它也由一个"与"平面和一个"或"平面构成,但是这两个平面的连接关系是可编程的。在 PAL 的基础上,发展了通用阵列逻辑 GAL,它采用了 EEPROM 工艺,实现了电可擦除、电可改写,其输出结构是可编程的逻辑宏单元,因而它的设计具有很强的灵活性,至今仍有许多人使用。

这些早期的 PLD 器件的一个共同特点是可以实现速度特性较好的逻辑功能,但其过于简单的结构也使它们只能实现规模较小的电路。为了弥补这一缺陷,20 世纪 80 年代中期,Altera 和 Xilinx 分别推出了类似于 PAL 结构的扩展型的复杂可编程逻辑器件 CPLD (Complex Programmable Logic Device) 和与标准门阵列类似的现场可编程门阵列 FPGA (Field Programmable Gate Array)。它们都具有体系结构和逻辑单元灵活、集成度高以及适用范围宽等特点。这两种器件兼容了 PLD 和通用门阵列的优点,可以实现较大规模的电路,编程也很灵活。目前,这两种器件已成为可编程逻辑器件的主流产品。

CPLD 将 PLD 的概念扩展到更高层次的集成度范畴,和 PLD 相比,CPLD 允许具有更多的输入信号、乘积项和宏单元,它内含多个逻辑快,每一个逻辑块相当于一片简单的 PLD,这些逻

辑块使用可编程内连线的布线来实现相互间的联系。由于在 CPLD 器件内可以通过逻辑阵列将大型函数在一级逻辑中实现,因此它能够提供最高的系统运行速度,同时它易于确定的时序参数有助于逻辑分析仪工作,但是它的寄存器资源相对 FPGA 较少。CPLD 设计的中心工作主要是将逻辑分配到各个逻辑块中并通过内部互连进行连接。

FPGA 实际上是由一系列逻辑单元的阵列构成的,这些阵列单元通过可编程连线阵列实现逻辑单元之间的互连,及和可编程 I/O 单元的互连。FPGA 的逻辑单元从功能上而言,比 CPLD 的组合乘积项及宏单元要简单得多,但是它却可以通过各逻辑单元的级联组合来创建很大的函数功能。目前,FPGA 主要用于数据通路、多 I/O 口、多寄存器的应用,工作在 33 或 66 MHz 频率下的 PCI 总线接口,可达到 3 ns 信号建立时间的 DRAM 控制器,时钟/输出延时为 6 ns 的 DMA 控制器,以及包括以太网和 ATM 的网络应用。面向 FPGA 的逻辑综合、布局布线工作主要围绕着根据器件结构对设计逻辑和信号传输路径进行优化,以便在密度和速度之间取得一个合理的平衡。

三、VHDL 语言和 EDA 工具

在 $500 \sim 10 \times 10^4$ 门大容量 CPLD 和 FPGA 的应用设计中,若采用传统的布尔方程、卡诺图或门级描述方式,则难以快速和有效地完成设计,于是硬件描述语言 HDL (Hardware Description Language) 应运而生。

VHDL 的全称是 VHSIC Hardware Description Language,而 VHSIC 则是 Very High Speed Integrated Circuit 的缩写。VHDL 是美国国防部资助的 VHSIC 项目开发的产品。今天,VHDL 成为数字电路和系统描述、建模、综合的工业标准。它的优异性能和一些独特的功能有:(1) VHDL 具有功能强大的语言结构,可以用简洁明确的代码描述来进行复杂控制逻辑的设计,它具有多层次的设计描述功能,支持设计库和可重复使用的元件生成;(2) VHDL 允许设计者生成一个设计而不需要首先选择一个用来实现设计的器件,对于同一个设计描述,可以采用多种不同的器件结构来实现其功能,这样就可以集中精力进行设计的构思;(3) VHDL 是一个标准语言,所以 VHDL 的设计描述可以被不同的 EDA 工具所支持,这意味着同一个 VHDL 设计描述可以在不同的设计项目中采用;(4) 如果设计被综合到一个 CPLD 或 FPGA 的话,就可使设计的产品以最快速度上市,当产品的产量达到相当的数量时,采用 VHDL 能够很容易地帮助你实现 ASIC 的转化设计。

采用 VHDL 的数字系统设计的过程可以划分为 6 个步骤:(1) 对设计要求的定义;(2) 用 VHDL 进行设计描述 (系统描述与代码设计);(3) 源代码模拟;(4) 设计综合、设计优化和设计的布局布线;(5) 布局、布线后的设计模块模拟;(6) 器件编程。其中 3、4、5 步需要借助 EDA 工具进行。目前比较常用的 VHDL 综合工具有 Synopsys 公司的 FPGA Express 和 Exemplar Logic 公司的 Leonardo Spectrum,VHDL 仿真软件有 Mentor 公司的 Modelsim,另外还有 PLD 厂商的 EDA 工具,如 Altera 公司的 MAX + plus Ⅱ、QUARTUS Ⅱ,Xilinx 公司的 Foudation Series 等。

四、可编程模拟器件

自从 ISP 技术被 Lattice 公司提出之后,它一直被用于数字系统之中。直至 1999 年 11 月,Lattic 公司推出 ispPAC (In-System Programmability Programmable Analog Circuit) 在系统可编程

模拟电路,才将 ISP 技术引入到模拟系统中,从而为模拟电路的设计自动化翻开了新的一页。在 ispPAC 出现前,模拟系统的设计往往用大量标准器件来搭建。ispPAC 的出现,使得高集成度的精确模拟设计能够通过一小片单片 ispPAC 来实现,从根本上简化和加速了模拟电路的设计、集成和装配。

目前已经推出的在系统可编程模拟器件有三种:ispPAC10、ispPAC20 和 ispPAC80。这些器件虽然在细节上有些差别,但有许多共同的地方。它们都有数个基本单元电路、参考电压、自动校正单元和 ISP 接口。基本单元电路称为 PAC 块(PAC block),它由两个仪用放大器和一个输出放大器组成,配以电阻、电容构成一个真正的差分输入、差分输出的基本单元电路。所谓真正的差分输入、差分输出是指每个仪用放大器有两个输入端,输出放大器也有两个输出端。电路的输入阻抗为 $10^9\ \Omega$,共模抑制比为 69 dB,增益调整范围为 - 10 dB ~ + 10 dB。PAC 块中电路的增益和特性都可以用可编程的方法来改变,如器件可配置成 1 ~ 10 000 倍的各种增益。每个 PAC 块都可以独立地构成电路,采用级联的方式也可以构成更大的电路以实现复杂的模拟电路功能。

ispPAC 可实现三种基本功能:(1)信号调理;(2)信号处理;(3)信号转换。信号调理主要是对信号进行放大、衰减、滤波。信号处理是指对信号进行求和、求差、积分运算。信号转换是指把数字信号转换成模拟信号。同时,还可以将这些基本的模拟功能进行灵活的组合配置,设计出更加复杂的模拟系统。ispPAC 器件的开发软件为 PACDesigner。开发完成后,通过 Lattic 公司的 ispDOWNLOAD CABLE 编程电缆将设计方案下载到电路板上的芯片中。器件的编程次数可达 10 000 次。因此,仪器仪表中的模拟部分,例如信号的放大、滤波电路等,可以根据信号的性质随时修改,甚至可以根据在现场观测到的信号的特性加以修改,从而大大方便了仪表电路的设计,并增强了其灵活性和适应性。

参 考 文 献

1 Kevin Skahill. 可编程逻辑系统的 VHDL 设计技术 [M]. 南京:东南大学出版社,1998

2 ispPAC10/20/80 In-System Programmable Analog Circuit [Z]. America: Lattice Semiconductor Corporation, 1999

3 薛宏熙,边计年,苏明. 数字系统设计自动化 [M]. 北京:清华大学出版社,1996

选自《仪表技术》月刊,2002 年第 2 期

9.10 可编程模拟器件 ispPAC20 在电路设计中的应用

山东东营石油大学 (257061) 任旭虎 刘润华 王心刚

一、概 述

美国 Lattice 公司的系统可编程技术 (in-system programmability) 改变了传统数字电子系统的设计和实现方法。Lattice 公司推出的系统可编程模拟电路 (ispPAC),为电子设计自动化 (EDA) 技术的应用开拓了更广阔的前景。与系统数字可编程逻辑器件 (ispPLD) 一样,ispPAC 中允许设计者使用开发软件在计算机上设计、修改模拟电路,进行电路特性模拟,最后通过编程电缆将设计方案下载至芯片中。ispPAC 把高集成度、精确的设计集成到一片模拟器件中,取代了用许多独立标准器件去实现电路的功能,大大提高了电路设计的效率,提高了系统的可靠性,降低了成本。从另一角度来看,随着计算机技术和集成电路技术的发展,现代电子产品设计已经进入了 EDA 时代,采用可编程器件进行设计,已经成为一种发展的必然趋势。

ispPAC 的应用非常灵活和广泛,可实现对模拟信号进行放大、转换、滤波等功能;除此之外,还可实现信号的比较、变换、产生等。

二、压力测量报警电路

压力测量报警电路如图 9.10－1 所示。R_1、R_2 和 R_t (压敏电阻,阻值为 $R+\Delta R$) 为压力传

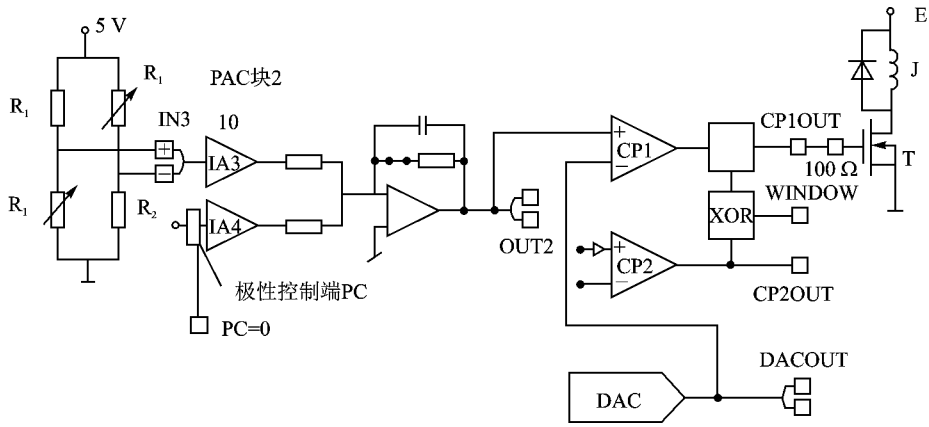


图 9.10－1 压力测量电路

感器的四个桥臂电阻,$R_1=R_2=R$,其供电电压为 5 V。压力为 0 时,四个桥臂电阻阻值均为 R,桥路输出电压为 0 V,压力升高时,R_t 阻值增加,桥路输出电压增大,该电压加在 ispPAC20 的 IN3 输入端,经 IA3 的放大后在 OUT2 端输出。显然 OUT2 端的差分输出电压正比于桥路输出电压,它反映了压力变化。根据信号大小可设置 IA3 的增益。CP1 是一电压比较器,同相端

接 OUT2 端的输出电压，反相端接 DAC 的输出电压，DAC 输出电压的大小和极性可通过编码设置，该电压即为报警门限电压。当压力超过门限值时，CP1OUT 端输出高电平，场效应管 T 导通，驱动继电器线圈 J，继电器常开触点接通电源即可报警。

若 IA3 的增益不能满足要求时，还可同时利用放大器 IA4，即将 IA4 的输入端连接到 IN3 输入端。这充分说明 ispPAC 的使用是非常灵活的。

三、温度监控电路

温度监控电路如图 9.10－2 所示。温度传感器的输出信号可以是电桥输出，也可以是半导体温度传感器的输出，接在 ispPAC20 的 IN3 输入端，经 IA4 的放大后在 OUT2 端输出。CP1 的同相端接 OUT2 端，反相端接 CPIN 输入端，该电压即为上行门限电压；CP2 为一滞环电压比较器，将 Hyst 置为 off，则 CP2 变为一单门限电压比较器。CP2 的反相端接 OUT2 端，同相端接 DAC 输出端 DACOUT，其输出电压值可以通过编码设置为正值或负值，该电压即为下行门限电压。CP1 输出设置为 Direct，XOR 设置为触发器模式 FF，这样，CP1、CP2、FF 共同组成一施密特触发器，WINDOW 端为输出端。当温度低于设定下门限值时，CP2OUT 端输出高电平，触发器 FF 复位，WINDOW 端输出低电平，光电器件导通，驱动继电器线圈 J，继电器常开触点闭合接通加热设备电源，开始加热。之后，温度逐步升高，当温度高于设定上门限值时，CP1OUT 端输出高电平，触发器 FF 置位，WINDOW 端输出高电平，光电器件截止，继电器线圈 J 无电流通过，继电器常开触点断开，切断加热设备电源，温度逐步降低。这样，利用该电路即可进行温度的自动监控。

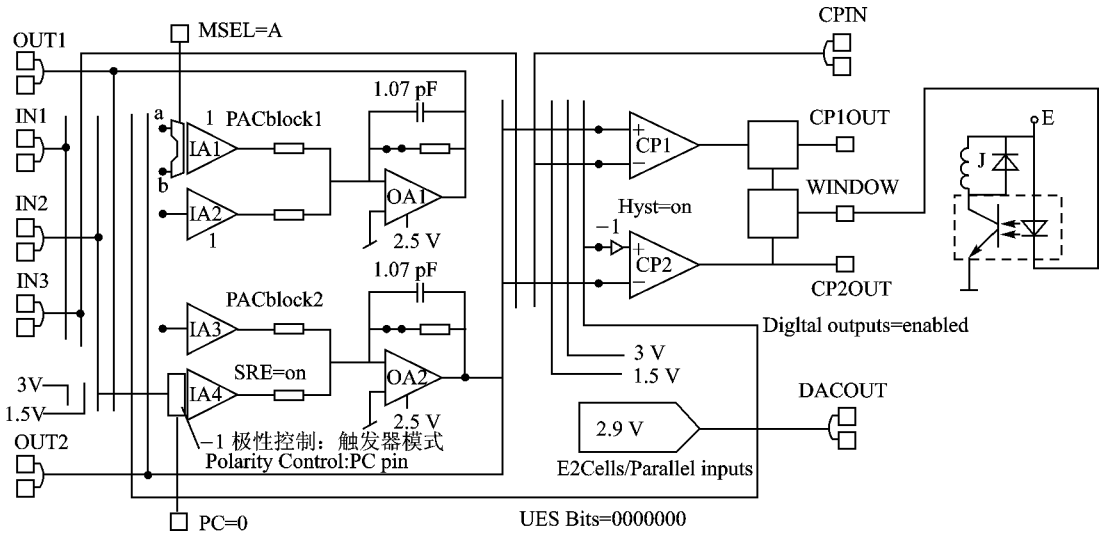


图 9.10－2　温度监控电路

四、压控振荡器

压控振荡器如图 9.10－3 所示。电压控制信号接在 ispPAC20 的 IN2 输入端，经 OA2 的积分后在 OUT2 端输出。CP1 的同相端和 CP2 的反相端接 OUT2 端，CP1 的反相端和 CP2 的同相端接 DAC 的输出端 DACOUT；CP2 为一滞环电压比较器，将 Hyst 置为 off，同时将 CP1 和

CP2 的输出信号配置为 RS 触发器模式,则 WINDOW 输出为一迟滞比较器,该迟滞比较器上限电压和下限电压分别为正、负 DAC 输出电压。同时将 ispPAC20 的 PC 控制配置为触发器模式,这样当积分器输出电压大于迟滞比较器上限电压和小于迟滞比较器下限电压时改变 IA4 的增益极性,往复进行积分器电容充电和放电,在 OUT2 端输出三角波信号,同时在 WINDOW 端输出方波信号。IN2 的输入电压控制积分速度的快慢,因此在 OUT2 和 WINDOW 端出现压控频率的三角波和方波信号。

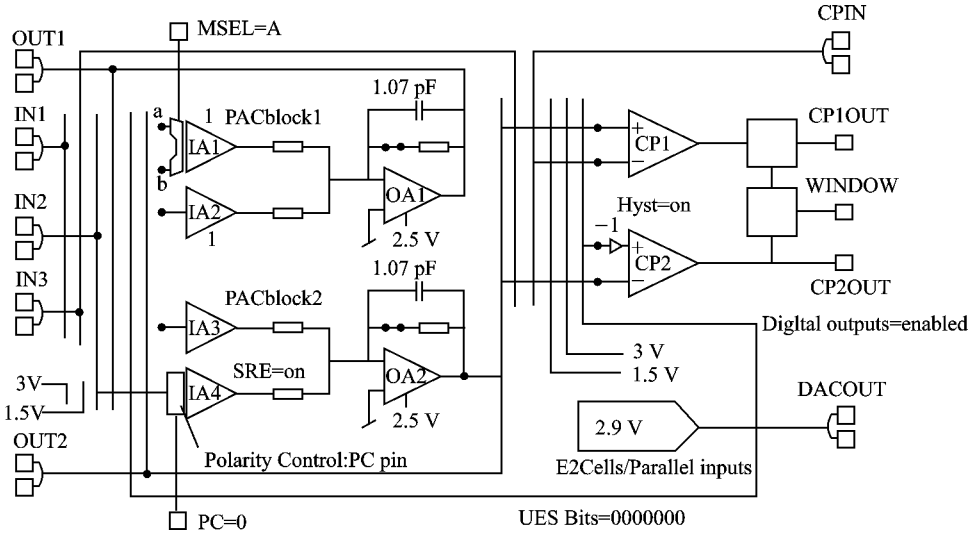


图 9.10－3 压控振荡器

压控振荡器的波形如图 9.10－4 所示。

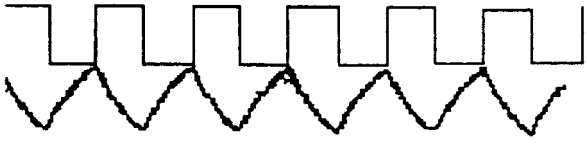


图 9.10－4 压控振荡器的波形

从上面的例子中可以看出,ispPAC 对于电子线路实验和实际电子产品的设计是很有帮助的。对电子工程、信息工程和自动控制领域的学习和设计人员来说,它都具有很高的价值。国内外可编程模拟器件的研究和应用都才刚刚起步,品种也较少,这方面的资料也很缺乏。但是随着电子技术的发展,可编程模拟器件肯定会像数字可编程器件一样得到充分的发展和广泛的应用。

选自《电子技术》月刊,2002 年第 10 期

9.11 基于 FPGA 的 I²C 总线接口实现方法

广州华南理工大学电信学院 (510641)
王 前 吴淑泉 刘喜英

一、引 言

由于 I²C 总线的连线少,结构简单,可不用专门的母板和插座直接用导线互连各个设备,因而可大大简化系统的硬件设计。许多半导体厂商都引进了此项总线技术,并推出了不少带 I²C 总线接口的芯片。已有不少文献讨论了 I²C 总线接口的单片机编程技术,本文从另一个角度论述基于 FPGA 的 I²C 总线接口的实现方法。

二、I²C 接口及通信协议

I²C 总线是以双向的数据线 SDA 和时钟线 SCL 两根连线实现了完善的全双工同步数据传送。总线备用时 SDA 和 SCL 都必须保持高电平状态。只有关闭 I²C 总线时才使 SCL 钳位在低电平。所有具有 I²C 接口的芯片,其 SDA 线都接到总线的 SDA 上,其时钟线 SCL 接到总线的 SCL。由于 I²C 总线接口均为开漏 (OD) 或开集电极 (OC) 输出,从而总线上所有电路的输出能实现“线与”逻辑功能,故要求 I²C 系统总线的输出端必须接上拉电阻。

总线的数据传输由主器件控制。主器件在时钟线 (SCL) 上产生时钟脉冲。在数据线 (SDA) 上产生寻址信号、启始信号、停止信号。任何被寻址选中的器件都被看成是从器件。每个具有 I²C 接口的设备都有一个惟一的 7 位地址,以便于主控器寻访。I²C 总线的数据传输格式如图 9.11-1 所示。

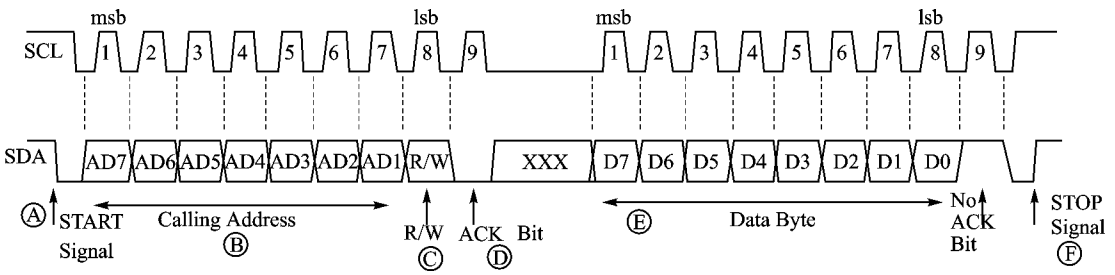


图 9.11-1 I²C 标准通信协议

主器件发出启始信号后 (SCL 为高时,SDA 出现下降沿),接着发送寻址信号 (由 7 位的从器件地址和数据方向位 R/W 组成,其中 R/W = ‘0’表明数据发送到从器件;R/W = ‘1’表明主器件读取从器件的数据),注意先发送最高位。ACK 为应答信号,此时主器件在 SCL 线上产生一个应答脉冲 (第 9 个脉冲),当被选中的从器件接收到数据后,从器件将 SDA 线拉低,这时主器件可以继续发送数据。当从器件由于某种原因不产生应答信号或全部数据传送结束后,

主控器可产生一个停止信号来终止总线数据传输。此时 SCL 线必须保持高电平,SDA 线出现由低到高的电平变化。I^2C 总线的其他具体要求参见文献 [1]。

三、硬件构成及实现方案

I^2C 总线通过器件地址的硬件设置并通过软件寻址方法,完全避免了对器件的片选线寻址操作。由于它只由两根线构成,从而可以极方便地构成多机系统和实现系统的外围器件扩展。一般而言,I^2C 总线系统常用于控制而无需高速传送数据的应用场合。本文通过计算机 ISA 插槽与 Motorola 公司的 SCM20014 芯片之间的控制接口为例,说明基于 FPGA 的 I^2C 接口的设计方法。

1. 系统总体结构设计

SCM20014 芯片是一种具有标准 I^2C 接口的 Digital Image Sensor 芯片 (以下简称为 SENSOR)。计算机通过对 SENSOR 内部的寄存器送初始化数据来控制 SENSOR 的工作。由于 SENSOR 内部寄存器的地址不是连续的,因而一次只对 SENSOR 内的某一个寄存器通过 I^2C 进行读/写操作。整个系统的结构框图如图 9.11-2 所示。

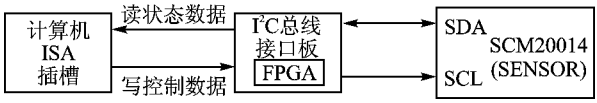


图 9.11-2 系统结构框图

基于 FPGA 的 I^2C 总线接口板主要完成两个功能:① 接收计算机送来的相关数据并存储在 FPGA 内;② 根据 I^2C 通信协议的时序要求完成与 SENSOR 之间的数据交换。本设计中 I^2C 总线接口分为两个模块:I^2C 控制模块和 I^2C 数据传输模块。为实现计算机对 SENSOR 的控制或采集 SENSOR 的数据,一方面用 C 语言编写对 SENSOR 写控制数据和读 SENSOR 状态信息的界面程充,另一方面用 VHDL 编写实现 I^2C 总线协议的程序。根据 I^2C 通信协议和 SENSOR 的技术资料[2],计算机通过 ISA 插槽往 SENSOR 写控制数据的顺序是:先送 SENSOR 的器件地址,再送 SENSOR 内部某个寄存器的地址,最后送控制数据。因为本系统是主/从的 I^2C 总线系统,分配 3 个计算机的 I/O 口 (0300H、0301H、0302H) 给 FPGA 上的 I^2C 控制模块,0300H 写器件地址 (7 位) 和 R/W 位;0301H 写 SENSOR 内部某个寄存器地址;0302H 写这个寄存器的控制字。以下主要介绍 I^2C 控制模块和 I^2C 数据传输模块的实现方案。

本系统是基于 MAXPLUSII 平台开发的,其框图如图 9.11-3 所示。图中:

isadata 是 ISA 插槽送给 SENSOR 的控制数据或读到的 SENSOR 的状态信息;

bclk 是 ISA 插槽上的时钟信号;

isaaddr 是 ISA 插槽送来的 16 位端口地址;

ior,iow 是 ISA 插槽上的读/写信号;

deviceaddr 是 SENSOR 的器件地址 (7 位) 和 R/W 位 (1 位);

regaddr 是被控制的 SENSOR 内部的某个寄存器地址;

regdata 是 ISA 送来的寄存器初始化数据;

reg_read_data 是从 SENSOR 中读到的某个寄存器的内容;

reg_read_data_en = '1' 表示正在进行 I^2C 的读过程;

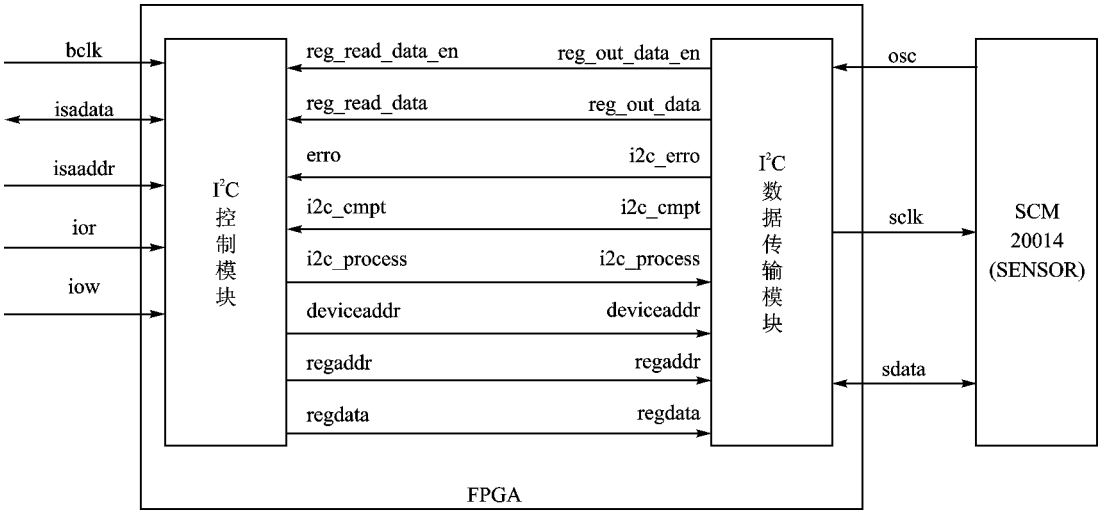


图 9.11－3 I²C 的结构模块

sclk 是 FPGA 产生的 I²C 总线时钟；

sdata 是 I²C 总线的双向数据线。

2. I²C 控制模块

该模块主要完成 FPGA 与 ISA 的数据交换。它主要由 3 个数据寄存器和 1 个控制/状态寄存器构成。其写操作过程是：当 isaaddr 为 0300H 时，此模块将 isadata 上的数据存放到 deviceaddrbuf 中；0301H 时，isadata 上的数据存放到 regaddrbuf 中；0302H 时，isadata 上的数据存放到 regdatabuf 中；0303H 时，ISA 往模块控制/状态寄存器的 controllerbuf (0) 写‘1’，使 I²C_process = ‘1’开始 I²C 传送；当 erro = ‘1’时，controllerbuf (1) = ‘1’，告诉计算机 I²C 传输出错，进行出错处理。读操作过程是：当 reg_read_data_en = ‘1’时，I²C 控制模块将 reg_read_data 送来的 SENSOR 数据存放到 regdatabuf 中，再送到 isadata 上。

实现 I²C 控制模块的技术关键是：① isadata 是双向引脚，我们可利用 ior 和 iow 做方向控制，当 ior = 0 时，isadata 做输出；当 iow = 0 时，isadata 做输入。② 因为 ISA 要在大约 4 个 bclk 后才动作，所以从 isadata 输出的数据一定要在 4 个 bclk 后才送到 isadata 上。

3. I²C 数据传输模块

通过 I²C 数据传输模块可实现 I²C 严格的通信协议，其关键是严格遵守 I²C 的时序要求来发启始信号、停止信号和等待响应信号。在本设计中当 I²C_process = ‘1’时，完成一个完整的 I²C 的读操作过程或写操作过程。注意一定要在 SCL 为高时，SDA 出现下降沿发出启始信号；并且在传完一个字节 (8 位) 后，主器件一定要将 SDA 线拉高，等待从器件发出响应信号；在 SCL 为高时，将 SDA 从低拉到高，发停止信号。另外，在实现过程中，还要注意 SCL 的时钟频率不能太高，否则会影响 I²C 传送的稳定性。

4. FPGA 的实现

本设计选用 ALTERA 公司 FPGA 产品 FLEX10KE。整个设计采用 VHDL 语言描述，在 MAXPLUSII 平台上完成了系统的仿真、综合、映射、布局。在仿真结果正确后，通过器件编程 (即通过编程器将设计下载到实际芯片中) 进行系统调试，直到最后实现。

四、结束语

本文介绍了一种简易的基于 FPGA 的 I^2C 总线接口板的设计方法。VHDL 语言的可移植性和不依赖器件的特性，较之传统的总线接口设计方法，设计者能在更抽象的层次上把握和描述系统电路的设计结构和功能特性，使设计更具灵活性。

参 考 文 献

1 何立民. I^2C 总线应用系统设计. 北京：北京航空航天大学出版社. 1995

2 赵雅兴. FPGA 原理、设计与应用. 天津：天津大学出版社. 1999.

选自《半导体技术》月刊，2002 年第 1 期

9.12 基于 CPLD 的串并转换和高速 USB 通信设计

天津大学 王 朔 李 刚 于学敏

可编程逻辑器件(PLD)是20世纪70年代在ASIC设计的基础上发展起来的一种划时代的新型逻辑器件。自PLD器件问世以来,制造工艺上采用TTL、CMOS、ECL及静态RAM技术,器件类型有PROM、EPROM、E^2PROM、FPLA、PAL、GAL、PML及LCA等。PLD在性能和规模上的发展,主要依赖于制造工艺的不断改进,高密度PLD是VLSI集成工艺高度发展的产物。20世纪80年代末,美国ALTERA和XILINX公司采用EECMOS工艺,分别推出大规模和超大规模的复杂可编程逻辑器件(CPLD)和现场可编程逻辑门阵列器件(FPGA)。这种芯片在达到高集成度的同时,所具有的应用灵活性和多组态功能是以往的LSI/VLSI电路无法比拟的。自从跨入20世纪90年代以来,可编程逻辑器件CPLD/FPGA得到了飞速发展,向高集成度、高速度和低价位方向不断迈进,不仅具有电擦除特性,而且出现了边缘扫描及在线编程等高级特性,其应用领域不断扩大,可用于状态机、同步、译码、解码、计数、总线接口、串并转换等很多方面,而且在信号处理领域的应用也活跃起来。使用CPLD可以提高系统集成度、降低噪声、增强系统可靠性并降低成本。

本文主要介绍ATMEL公司的CPLD芯片ATF1508AS的特点及应用。ATF1508AS是利用ATMEL成熟的电擦除技术实现的高性能、高密度的复杂可编程逻辑器件(CPLD),与ALTERA公司的EPM7000系列引脚完全兼容,可以将EPM7000的POF文件转换为适合ATF1508AS的工业标准JEDEC编程文件,下载到ATF1508AS芯片中。

一、ATF1508AS 的特点

ATF1508AS是利用ATMEL成熟的电擦除技术实现的高性能、高密度的复杂可编程逻辑器件(CPLD)。它有128个逻辑宏单元和最大100个输入,能很容易地集成一系列TTL、SSI、MSI、LSI和传统PLD的逻辑功能。ATF1508AS的增强型路由开关矩阵增加了可用的门数和设计改变时引脚锁定的成功率,这是非常重要的。ATF1508AS有96个双向I/O引脚和4个输入引脚。这4个输入引脚也可以用于全局控制信号、全局寄存器时钟、全局复位和全局输出允许。

128个宏单元中的每一个都产生一个隐藏的反馈回路到全局总线,每一个输入引脚、I/O引脚也都汇入全局总线。每个逻辑块的开关矩阵从全局总线中选择40个独立的信号,每一个宏单元也产生一个返送逻辑项到局部总线。宏单元之间的级联逻辑可以快速有效地实现复杂的逻辑功能。ATF1508AS包括八个这样的逻辑链,每一个都能通过扇入最多40个乘积项实现逻辑项求和。

ATF1508AS是在系统编程(ISP)器件。它用工业标准的4脚JTAG接口(IEEE标准1149.1),完全与JTAG的边界扫描描述语言(BSDL)兼容。ISP允许器件不用从印刷电路板上拿走就可编程,除简化生产流程外,ISP也允许通过软件进行设计修改。

ATF1508AS 的引脚保持电路提供对所有输入和 I/O 引脚的设置。当任何引脚驱动到高电平或低电平,紧接着引脚被悬空时,引脚将保持先前的高电平或低电平状态。这种电路防止没有用到的输入和 I/O 线悬空而成为中间的电压值,这会导致不必要的功耗和系统噪声。引脚保持电路去除了对外部上拉电阻的需要和直流功耗。

ATF1508AS 的加密特性可以保护 ATF1508AS 的设计内容。两个字节 (16 位) 的用户信号可被用户存取,能存放工程名、部件号、版本或日期等,而且用户信号的存取不受加密熔丝的状态影响。

ATF1508AS 具有上电复位特性。在上电期间,所有的 I/O 引脚将为三态,直到 V_{CC} 到达上电电压,这样可防止在上电期间发生总线竞争。ATF1508AS 的寄存器设计成在上电时复位,从 V_{CC} 上升到 V_{RST} 后的微小的延时,所有的寄存器将复位到低电平,输出状态要根据缓冲器的极性设置。这种特性对于状态机的初始化是比较关键的。

二、ATF1508AS 的宏单元

ATF1508AS 的宏单元如图 9.12－1 所示。它的宏单元非常灵活,足以支持高复杂逻辑功能并且高速工作。宏单元包括五个部分:乘积项和乘积项选择多路复用器、或/异或/级联逻辑、触发器、输出选择和使能、逻辑阵列输入。没有用到的宏单元可由编译器禁止以降低功耗。

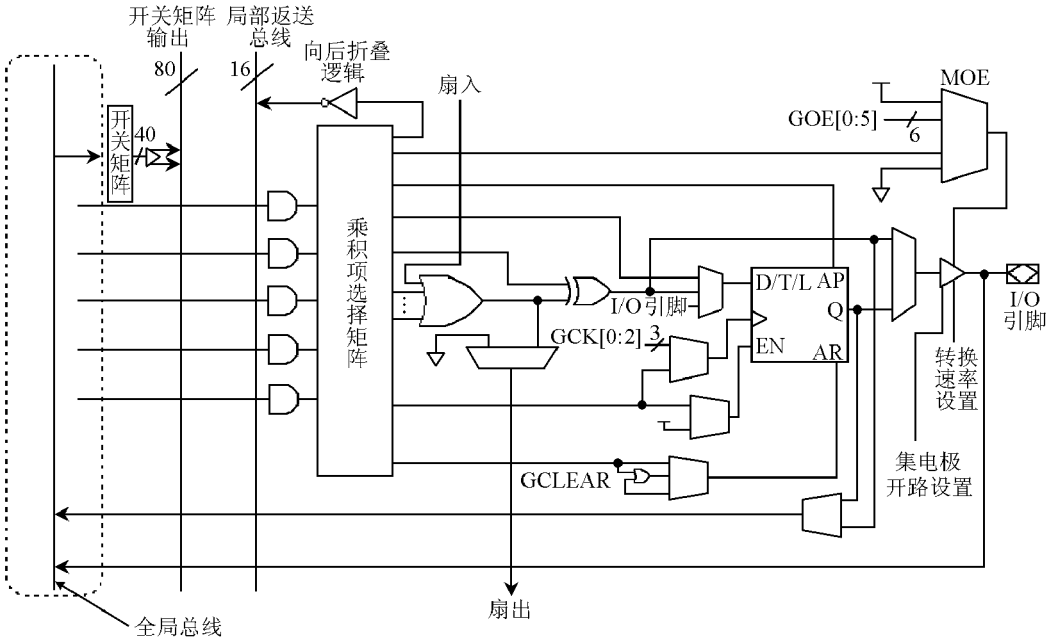


图 9.12－1 ATF1508AS 的宏单元

1. 乘积项和乘积项选择矩阵

每一个宏单元有 5 个乘积项,每个乘积项作为它的输入从全局总线和局部总线接收所有信号。乘积项选择矩阵 (PTMUX) 按需分配这 5 个乘积项到宏单元的逻辑门,也负责分配控制信号。乘积项选择矩阵的编程是由设计编译器决定的,编译器将选择优化的宏单元配置。

2. 或/异或/级联逻辑

ATF1508AS 的逻辑结构是为有效地支持所有的逻辑而设计的。在一个宏单元内,所有的乘积项可以被布进或门,产生一个 5 输入的与/或求和项。通过邻近的宏单元扇入额外的乘积项,可以扩展到 40 个乘积项而只有很小的延时。宏单元的异或门允许有效地实现比较和算术功能,其中异或门的一个输入来自或运算的求和项,另一个输入可以是一个乘积项或一个固定的高电平或低电平。对于组合逻辑输出,固定电平允许极性选择;对于时序逻辑,固定电平允许利用反演规则(摩根定律的推论)化简乘积项。异或门也可以用于仿真 T 型和 JK 型触发器。

3. 触发器

ATF1508AS 的触发器有非常灵活的数据和控制功能。触发器的输入可以来自于异或门、一个单独的乘积项或直接由 I/O 口输入。选择单独的乘积项允许在一个组合逻辑输出宏单元内生成一个隐藏的寄存器反馈(这个特性是由 fitter 软件自动实现的)。除 D、T、JK 和 SR 类型外,ATF1508AS 的触发器还可配置为锁存器。在这种模式中,当时钟为高时,数据通过;当时钟为低时,数据锁存。

时钟信号可以是全局 CLK 信号(GCK)和一个单独的乘积项。触发器在时钟的上升沿改变状态。当 GCK 信号作为时钟信号时,宏单元的一个乘积项可以选择作为时钟允许信号。当使用时钟使能功能时,使能信号(乘积项)为低时,所有的时钟边沿将被忽略。触发器的异步复位信号(AR)可以是全局复位信号(GCLEAR)、一个乘积项或不使用。AR 也可以是 GCLEAR 和一个乘积项的逻辑或输出。异步置位信号(AP)可以是一个乘积项或不使用。

4. 输出选择和使能

ATF1508AS 宏单元的输出可以选择为寄存器型和组合型。隐藏的反馈信号可以是组合或寄存器信号而不管输出是组合型还是寄存器型。输出使能多路复用器(MOE)控制输出使能信号。如果是简单的输出操作,任何缓冲器都可以永久使能。如果引脚用作输入,缓冲器也可以永久禁止。在这种配置下,所有的宏单元资源仍然可用,包括隐藏的反馈信号、扩展器和级联逻辑。每一个宏单元的输出使能信号都可以选择一个全局输出使能信号。该器件有 6 个全局输出使能信号(OE)。

5. 逻辑阵列输入

逻辑阵列输入包括全局总线/开关矩阵和返送总线。

(1) 全局总线/开关矩阵

全局总线包括所有的输入和 I/O 引脚信号以及所有 128 个宏单元的隐藏反馈信号。每个逻辑块的开关矩阵将全局总线的所有信号作为其输入。在软件的控制下,这些信号中最多可以有 40 个被选择作为逻辑块的输入。

(2) 返送总线

每一个宏单元可以产生一个返送乘积项。这个信号连接到局部总线上,并且对 16 个宏单元有效,它是宏单元一个乘积项的反极性。每个局部总线的 16 个返送项允许产生高扇入求和项(最多 21 个乘积项),而只有很小的延时。

三、设计软件支持

ATMEL 公司提供了 CPLD 的设计软件,而且很多第三方的工具软件也支持 ATF1508AS 的设

计,可以用多种高级描述语言和格式进行逻辑描述,如 CUPL、ABEL、VHDL 等。由于 ATF1508AS 与 ALTERA 公司的 EPM7000 系列是完全引脚兼容的,因此可以使用 ALTERA 公司的 MAXPLUSII 软件。它能进行 VHDL 语言的编译和综合,使用方便,功能强大。MAX-PLUSII 综合后产生适合 ALTERA 的 CPLD 编程的 POF 文件,使用 POF2JED 软件(ATMEL 公司提供),就可将 POF 文件转换为适合 ATF1508AS 的工业标准 JEDEC 编程文件,下载到 ATF1508AS 芯片中。

四、器件编程

ATF1508AS 器件是利用 4 脚 JTAG 协议在系统编程(ISP)的。ATMEL 提供了 ISP 硬件(下载电缆)和软件,以允许从 PC 对 ATF1508AS 进行编程。若要允许 ISP 编程支持"自动测试装置(ATE)"向量,必须通过 ATMEL 的 ISP 软件生成串行向量格式(SVF)文件,也可转换为除 SVF 外的其他 ATE 测试格式。ATF1508AS 器件也可以用标准的第三方编程器来编程,这时 JTAG ISP 口可以被禁止从而允许这四个额外的 I/O 引脚用于逻辑功能。

ATF1508AS 还有一个特性就是如果由于任何原因编程过程被中断,则器件将被锁定以防止输入和 I/O 引脚被驱动。在这种状态下,输入和 I/O 引脚缺省下为高阻状态。在编程器件时,输入和 I/O 引脚也将为高阻状态。此外,引脚保持电路设置在器件编程期间将保持以前的状态。ATF1508AS 器件出厂时被初始化为已擦除状态,可以直接用来 ISP 编程。

五、应用实例

1. 应用 ATF1508AS 进行串并转换

本系统应用 ATMEL 公司的 ATF1508AS 进行串行数据到并行数据的转换,在进行数据采集中,用到 Crystal 半导体公司生产的 24 位高精度 Σ-Δ 模/数转换器 CS5321/CS5322 组件。该组件最终输出字长为 24 位的 2 的补码格式的串行数字信号,将其转换为并行数据可以方便与单片机的接口。串并转换可采用移位寄存器来实现。对实现 6 通道 24 位采样,若采用移位寄存器,则需要 8 位移位寄存器,共 3 × 6 = 18 片,另外还要用几片译码器。这样,会使芯片数量大增,占用大片电路板面积,使系统的体积增大。本系统使用 ATF1508AS 来实现 6 通道 24 位数据的串并转换,可将大部分数字逻辑设计(包括组合逻辑和时序逻辑)集成在一个芯片内,大幅减少芯片数量,减小系统体积。

由于 ATF1508AS 内部有 128 个宏单元,而且 24 位串并转换需要 24 个移位寄存器,因此不能同时进行 6 通道的串并转换,只能分时复用。本系统分 3 次进行串并转换,每次转换 2 个通道,等待单片机读取 2 个通道的并行数据后再进行剩下的转换。部分串并转换 VHDL 程序如下(硬件描述语言是 VHDL,软件是 ALTERA 公司的 MAXPLUS Ⅱ 软件和 ATMEL 公司的 POF2JED 软件,下载软件是 ATMEL 公司的 ATMISP,下载电缆是 ATMEL 公司的专用电缆):

```
s2p :process (SCLK1M,DRDYIN,WORKING,RESET)
begin
    if WORKING = '1' or RESET = '1' then
        shift_enable < = '0';
        state < = s0;
    elsif SCLK1Mevent and SCLK1M = '0' then
```

```
count1 <= count1 + 1;
case state is
  when s0 => if DRDYIN = '0' then
                shift_enable <= '1';
                count1 <= (others => '0');
                int_reg <= '1';
                state <= s1;
             elsif READOK = '1' then
                int_reg <= '1';
             end if;
when s1 => shift_reg0 <= shift_reg0 (22 downto 0) & SOD (0);
           shift_reg1 <= shift_reg1 (22 downto 0) & SOD (1);
              if count1 = 23 then
                shift_enable <= '0';
                int_reg <= '0';
                state <= s2;
             else
                int_reg <= '1';
             end if;
when s2 => if shift_enable = '1' then
           shift_reg0 <= shift_reg0 (22 downto 0) & SOD (2);
           shift_reg1 <= shift_reg1 (22 downto 0) & SOD (3)
                if count1 = 23 then
                   shift_enable <= '0';
                   int_reg <= '0';
                   state <= s3;
                else
                   int_reg <= '1';
                end if;
             elsif READOK = '1' then
                shift_enable <= '1';
                count1 <= (others => '0');
             end if;
when s3 => if shift_enable = '1' then
           shift_reg0 <= shift_reg0 (22 downto 0) & SOD (4);
           shift_reg1 <= shift_reg1 (22 downto 0) & SOD (5);
                if count1 = 23 then
                   shift_enable <= '0';
                   int_reg <= '0';
                   state <= s0;
                else
                   int_reg <= '1';
                end if;
```

```
                elsif READOK = '1' then
                    shift_enable <= '1';
                    count1 <= (others => '0');
                end if;
        end case;
    end if;
end process;
```

2. 应用 ATF1508AS 进行高速 USB 通信

USB 是近年来应用在 PC 领域的新型接口技术,具有使用方便、速度快、连接灵活、支持热插拔等特点。USB1.1 协议定义在高速下 12 Mb/s、低速下 1.5 Mb/s 的传输速度。若要达到高速 12 Mb/s (相当于近 1 MB/s) 的速度,就要大约 1μs 传输 1 个字节。但由于 USB 的控制传输、错误检测以及单片机本身速度的限制,很难达到这么高的速度,因此,必须采用 DMA 方式才能达到真正的高速传输,使用 CPLD 就可以实现类似 DMA 方式。单片机负责解释 USB 的控制传输,当要进行从外存取数送到 USB 接口芯片时,单片机让出总线,由 CPLD 完成该工作。CPLD 产生外存的读取时序和地址、片选信号,同时产生 USB 接口芯片的写时序和地址、片选信号,这样就可以自动实现外存数据到 USB 接口芯片的工作,而且速度很快,不需要单片机干预。

以下给出 RAM 的读取时序、地址信号和 USB 接口芯片写时序的 VHDL 程序片断。

```
rram1 :process (SCLK2M) -- RAM_OE (RAM 读时序)
begin
    if SCLK2Mevent and SCLK2M = '1' then
        if read = 0 then
            ram_oe_reg <= '1';
            cpld2_counter <= (others => '0');
        elsif read = '1' then
            cpld2_counter <= cpld2_counter + 1;
            if cpld2_counter > 0 then
                ram_oe_reg <= not ram_oe_reg;
            end if;
        end if;
    end if;
end process;

rram2 :process (SCLK2M,WORKING,RESET) -- ADDRESS (RAM 地址信号)
begin
    if WORKING = '1' or RESET = '1' then
        adr_reg <= (others => '0');
    elsif SCLK2Mevent and SCLK2M = '0' then
        if read = '1' and ram_oe_reg = '1' and cpld2_counter > 2 then
            adr_reg <= adr_reg + 1;
        end if;
```

```
    end if;
end process;
wd12 :process (SCLK2M) --USB 芯片读时序
begin
    if SCLK2M'event and SCLK2M = '0' then
      if read = '0' the
        d12_wr_reg < = '1';
      elsif read = '1' and cpld2_counter / = 129 then
        d12_wr_reg < = not d12_wr_reg;
      end if;
      end if,
end process;
```

六、结束语

CPLD 器件的优势在于缩短开发生产周期,现场灵活性好,而且随着电子技术的发展,其集成度越来越高,速度越来越快,价格也逐渐降低,因此市场发展很快。ATMEL 公司的 ATF1508AS 是高性能、高密度的复杂可编程逻辑器件,使用方便,具有很高的性价比,因此具有广阔的应用前景。

参 考 文 献

1 ATMEL Corp. ATF1508AS Users Manual

2 ATMEL Corp. Designing for In-System Programmability with Atmel CPLDS

3 ATMEL Corp. Atmel PLD Frequently Asked Questions

4 Stefan Sjoholm, Lennart Lindh, 用 VHDL 设计电子线路. 北京:清华大学出版社,2000

5 PHILIPS Corp. PDIUSBD12 Users Manual

选自《单片机与嵌入式系统应用》月刊,2002 年第 5 期

9.13 用 HDL 语言实现循环冗余校验

西安微电子技术研究所(710054) 王 耿 姜智忠

数据在计算机系统中形成、存取和传送的过程中,由于随机噪声的影响可能会产生错误。为了检测和减少这类错误,一方面是精确设计电子电路,提高硬件自身的可靠性;另一方面是在数据编码上想办法,即采用特定的编码方法,增加少量的附加电路,使之能检测/纠正某些错误。冗余编码就是通过在原始数据中加进一些冗余位,使合法数据在出现错误时变为非法数据,从而就可通过检测数据的合法性来达到发现错误的目的。合理地安排冗余位的数量和位置可提高检错/纠错的能力。常用的冗余校验有奇偶校验、海明校验和循环冗余校验。

一、循环冗余码

1. 概 述

循环冗余码(CRCCs)是在严密的代数学理论基础上建立起来,这种码的编码/解码电路不太复杂,且检错/纠错的能力较强,目前在理论上和实践上都有了较大发展。CRC 校验具有如下突出的特点:

⑴ 它能提供一般性错误的准确校验,不但适用于随机信道中独立性错误的检测,例如一位错、两位错、奇数位错等,也适用于在突发信道中出现的成串相关性突发错误。

⑵ 在编码规则上,原始数据即为循环冗余码的前一部分,紧接着就是冗余校验码部分,这就方便系统的理解和实现。

2. 编码原理

设 D(x)为信息码多项式,G(x)为系统选定的生成多项式。信息码长为 k 位,校验码长为 n - k 位,则循环冗余码长为 n 位,如图 9.13 - 1 所示。

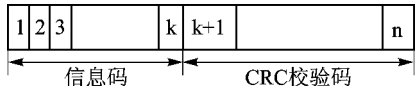


图 9.13 - 1 编码原理图

⑴ 循环冗余码的发送

首先,用生成多项式 G(x)的最高位 x^{n-k} 去乘 D(x),即 $x^{n-k} \cdot D(x)$。

其次,用 G(x)去除以上乘积,得

$$x^{n-k} \cdot D(x) / G(x) = Q(x)\text{,余数为 } R(x)$$

由于不带进(借)位模 2 的加减法是等同的,故

$$x^{n-k} \cdot D(x) + R(x) = G(x) \cdot Q(x)$$

最后,将 $x^{n-k} \cdot D(x) + R(x)$ 作为信息码 M(x)发送出去。

⑵ 循环冗余码的接收

接收方收到信息码后,作如下处理:

计算 M(x)/G(x),如果传输正确,则 $M(x) = D(x) \cdot x^{n-k} + R(x)$,即 M(x)/G(x) = R(x) 表示除尽。如果传输有错,则 M(x)/G(x),余数不为 0。

3. 关于生成多项式

通过上述,可看出生成多项式在 CRC 校验中的重要作用。应指出并非任何一个 n+1 位多项式都可作为生成多项式的,从检错及纠错的要求出发,生成多项式应满足下列要求。

(1) 任何一位发生错误都应使余数不为 0;

(2) 不同位发生错误应使余数不为 0;

(3) 对余数作模 2 除,且使余数循环。

目前,为了满足在不同应用场合及传输介质下对各种错误模式进行校验的要求,国际上已经公布了几种 CRC 生成多项式的标准,如表 9.13-1 所列。

表 9.13-1 CRC 生成多项式的标准表

生成 CRC 的名称	多 项 式
SDLC (CCITT)	$X^{16} + X^{12} + X^5 + X^0$
SDLC Reverse	$X^{16} + X^{11} + X^4 + X^0$
CRC-16	$X^{16} + X^{15} + X^2 + X^0$
CRC-16 Reverse	$X^{16} + X^{14} + X^1 + X^0$
CRC-12	$X^{12} + X^{11} + X^3 + X^2 + X^1 + X^0$
CRC-32	$X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X^1 + X^0$

二、CRC 的设计与实现

1. 串行 CRC 的设计与实现

通常 CRC 计算是通过线性反馈移位寄存器 (LFSRs) 实现的。移位寄存器由 n-k 位组成,加上几个异或门和一条反馈回路。寄存器初始化为全 0 (或全 1),数据字按由低位到高位 (或反之) 依次输入 LFSRs。当一位从最右边移出寄存器时就通过反馈回路和后续输入数据进行异或运算。当所有 k 位数据移入寄存器后,移位寄存器中所形成的结果即为 CRC 校验码。

用 Altera 硬件描述语言 AHDL 设计实现的串行 CRC 程序节录如下:

```
VARIABLE
    bita[15…11]:DFFE;
    bitb[10…4]:DFFE;
    bitc[3…0]:DFFE;
    feedback:NODE;
BEGIN
    …
    feedback = data_in XOR bitc0.q;
    bita15.d = feedback XOR GND;
    bita[14..11].d = bita[15..12].q;
    bitb10.d = feedback XOR bita11.q;
```

```
    bitb [9..4].d = bitb [10.5].q;
    bitc3.d = feedback XOR bitb4.q;
    bitc [2..0].d = bitc [3..1].q;
    crc_out = bitc0.q;
END;
```

串行 CRC 仿真结果如图 9.13－2 所示。

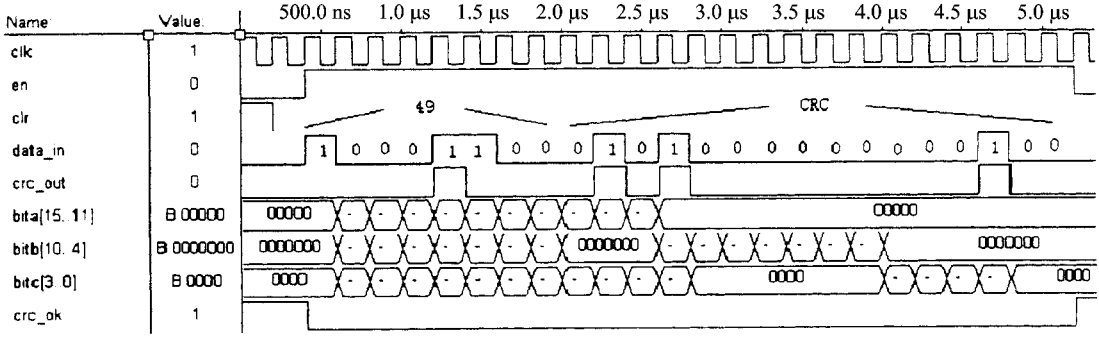


图 9.13－2 用 AHDL 设计实现的串行 CRC 程序仿真结果

2. 并行 CRC 的设计与实现

当数据是以位串形式表示时，串行方法能够很好地完成校验功能。目前的高速信号处理系统是以字节 (byte)、字 (word)、双字 (32 位) 或更宽的字段为处理单位的，即使在通信系统中，数据仍为串行发送，而在接收端数据被转换为字节表示形式进行处理的。因此就需设计同时进行 8 位数据校验的并行 CRC 算法，如图 9.13－3 所示。图中 8 位数据 D [7…0] 经过 8 次移位后寄存器中的内容 C [15…0] 即为 CRC 校验码，Mn = Cn⊕Dn。

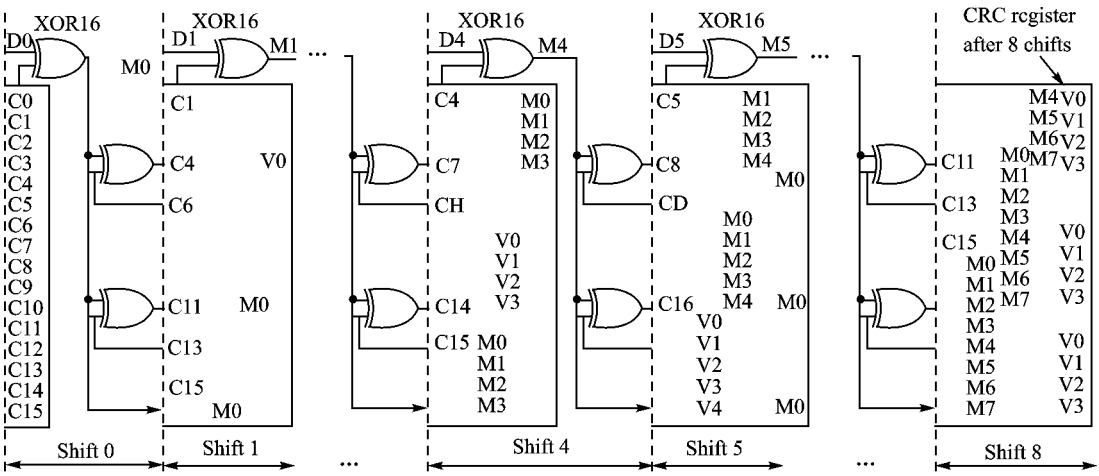


图 9.13－3 并行 CCITT CRC-16 校验码的生成器

用 VHDL 设计实现的并行 CRC 程序节录如下：

```
BEGIN
--compute interim terms for the first byte.
```

```
m(i) <= crc_reg(i) XOR data_byte(i);
--compute interm. byte product i bit of a byte (i=0~7)
Gen_Rx_CRC16 :PROCESS
  BEGIN
    WAIT UNTIL (clk EVENT AND clk = '1');
    IF rst = '1' THEN
      crc_reg <= hex_0;-stat byte,clear reg
    ELSE
      crc_reg(0) <= NOT en OR (crc_reg(8) XOR m(0) XOR m(0));
      crc_reg(1) <= NOT en OR (crc_reg(9) XOR m(5) XOR m(1));
      crc_reg(2) <= NOT en OR (crc_reg(10) XOR m(6) XOR m(2));
      crc_reg(3) <= NOT en OR (crc_reg(11) XOR m(7) XOR m(3) XOR m(0));
      crc_reg(4) <= NOT en OR (crc_reg(12) XOR m(1));
      crc_reg(5) <= NOT en OR (crc_reg(13) XOR m(2));
      crc_reg(6) <= NOT en OR (crc_reg(14) XOR m(3));
      crc_reg(7) <= NOT en OR (crc_reg(15) XOR m(4) XOR m(0));
      crc_reg(8) <= NOT en OR (m(5) XOR m(1) XOR m(0));
      crc_reg(9) <= NOT en OR (m(6) XOR m(2) XOR m(1));
      crc_reg(10) <= NOT en OR (m(7) XOR m(3) XOR m(2));
      crc_reg(11) <= NOT en OR (m(3));
      crc_reg(12) <= NOT en OR (m(0) XOR m(4));
      crc_reg(13) <= NOT en OR (m(1) XOR m(5));
      crc_reg(14) <= NOT en OR (m(2) XOR m(6));
      crc_reg(15) <= NOT en OR (m(3) XOR m(7));
    END IF;
  END PROCESS;
crc <= crc_reg;
END rtl_crc16;
```

并行 CRC 仿真结果如图 9.13－4 所示。

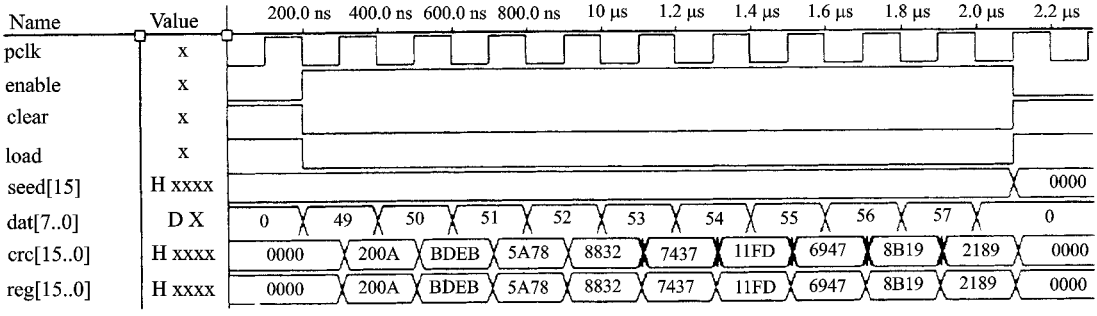


图 9.13－4 用 VHDL 设计实现的并行 CRC 程序仿真结果

三、结束语

由于循环冗余编码有良好的代数结构,编码简单,易于实现,检错能力强,在计算机通信系统中有广泛应用。

在当今数字电路设计开发中,HDL 已成为描述、验证和设计电子线路的重要方法之一,并有一套全新的系统设计开发流程和相关的国际标准支持。它是电子设计自动化 (EDA) 发展的产物,今后将被广泛应用并将继续推动 EDA 设计的进步与发展。

本设计是在 Altera 公司的 MAX + PLUSII (学生版) 设计平台下完成的。该开发系统是一个完全集成化、易学易用的可编程设计环境。在此,对于文中所引用的 Altera 公司提供的相关文档表示感谢。

参 考 文 献

1 AN49 "Implementing CRCCs in Altera Devices. Altera Corporation, 1995

2 王爱英. 计算机组成与结构. 北京:清华大学出版社,1995

3 雷震甲. 计算机网络. 西安:西安电子科技大学出版社,2000

选自《微电子学与计算机》月刊,2002 年第 5 期

9.14 利用单片机和 CPLD 实现直接数字频率合成 (DDS)

西安电子科技大学机电工程学院 (710071) 毕红军 张永瑞

直接数字频率合成 (DDS) 技术是美国学者 J. Tierncy, C. M. Rader 和 B. Gold 在 1971 年首次提出的。这是一种全数字技术,该技术从相位概念出发直接合成所需要的波形。同传统的频率合成技术相比,DDS 技术具有很多优点:频率切换时间短、频率分辨率高、相位变化连续、容易实现对输出信号的多种调制等[5]。但是由于当时的技术以及器件水平的限制,它的性能指标还无法与已有的技术相比,因此该技术当时并没有引起足够的重视。最近几年来,随着技术和器件水平的提高,国外一些公司先后推出各种各样的 DDS 专用芯片,如 Qualcomm 公司的 Q2230、Q2334,AD 公司的 AD9955、AD9850 等[3]。这些产品的问世,为电路设计者提供了良机,满足了工程实际的需要。然而,商用 DDS 专用电路芯片也有它的局限性,并不能满足所有要求。例如,在实现线性调频 (LFM) 等复杂的调制功能时,利用现有的商用芯片就会遇到一些困难[8]。由于近几年来可编程器件 CPLD、现场可编程门阵列 FPGA 技术的迅速发展和广泛应用,使用可编程器件实现 DDS 技术也越来越受到人们的关注。

一、DDS 工作原理

DDS 工作原理框图如图 9.14-1 所示,其实质是以参考频率源 (系统时钟) 对相位进行等可控间隔的采样。由图 9.14-1 可见,DDS 包括由相位累加器和 ROM 查询表构成的数控振荡源 (NCO)、DAC 以及低通滤波器 (LPF) 3 部分。在每一个时钟周期,N 位相位累加器与其反馈值进行累加,其结果的高 M 位作为 ROM 查询表的地址,然后从 ROM 中读出相应的幅度值送到 DAC。低通滤波器 LPF 用于滤除 DAC 输出中的高次谐波。因此通过改变频率控制字 K 就可以改变输出频率 f_{out}。容易得到输出频率 f_{out} 与频率控制字 K 的关系为:$f_{out} = Kf_c/2^N$,其中 f_c 为相位累加器的时钟频率,N 为相位累加器的位数。定义当 K = 1 为系统频率分辨率,即 $f_{out}^T = f_c/2^N$。

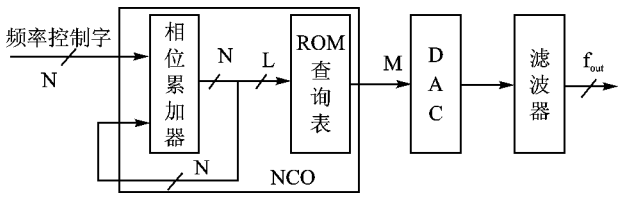


图 9.14-1 DDS 基本原理图

二、系统的总体设计

系统的原理框图如图 9.14-2 所示,本系统主要由单片机部分、DDS 主通道部分、键盘及显示部分以及输出信号调理等部分组成。

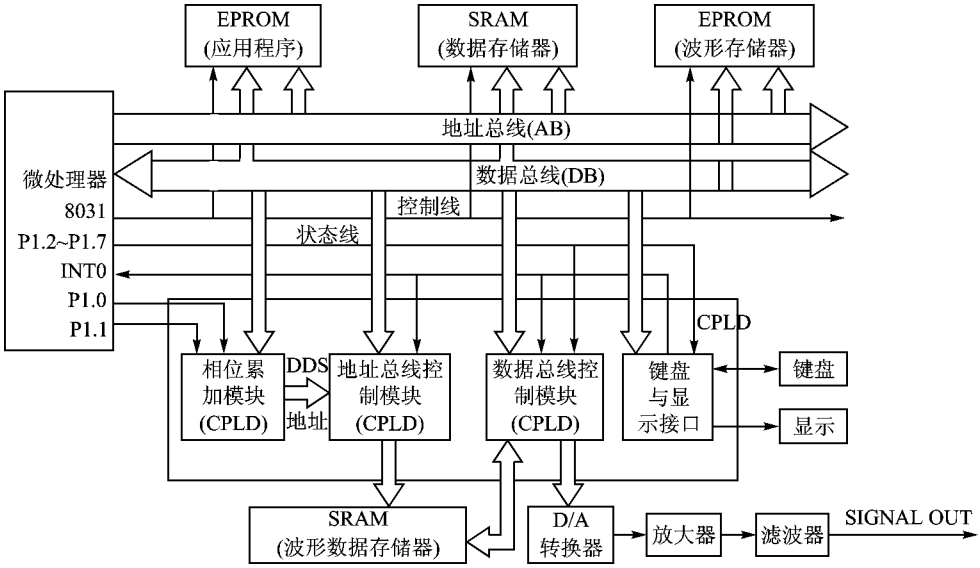


图 9.14－2 DDS 系统原理框图

单片机芯片采用的是比较常见的 AT80C31 芯片。同时片外还各扩展了 1 片程序存储器 2764 与数据存储器 6264，分别用来存放运行中所需的程序与随机数据。

DDS 主通道部分是我们设计的关键所在，该部分主要由相位累加模块、地址总线控制模块、数据总线控制模块与波形数据存储器 EPROM、SRAM 等组成。其中相位累加模块、地址总线控制模块和数据总线控制模块都是在 CPLD 上实现，采用的芯片是 ALTERA 公司的 FLEX10K 系列器件。我们将所需要合成的波形采样数据固化在 EPROM 2764 中，但是我们知道 EPROM 的读周期比较长，很难满足系统的访问时间要求。因此设计中又使用了 1 片 HSRAM，在 DDS 系统合成波形的过程中，代替 ROM 进行波形数据的快速查询。

键盘和显示部分是系统和用户进行交互的重要手段。这一部分的逻辑功能，也是在 CPLD 上实现的。

输出信号调理部分是把从 HSRAM 中读出的波形的数字幅度值首先转换成模拟信号，然后再进行放大、滤波处理后输出。这一部分包括 D/A 转换器、幅度放大器和滤波器。DAC 器件采用 AD 公司的 12 位 AD9713B，该器件特点是具有较高的更新速率（100 MSPS）和较低的功耗（725 mW）[1]，因此特别适合于 DDS 信号合成。幅度调节电路使用的是双极性放大器 AD708、AD9617 和 AD9713 所组成的电路。

三、系统总体工作状态说明

前面已经提到过，由于 EPROM 的读取时间比较长，很难满足系统对时间的要求，因此在系统中又增加了 1 片高速 SRAM，作为波形数据缓存器。这样，系统就有两个工作状态：首先，系统开始工作时，需要将波形数据从 EPROM 调到 HSRAM 中，即波形数据的加载状态；数据加载完毕后，按照 DDS 合成原理进行信号合成，即信号的合成状态。系统设计中使用单片机的 P1 口控制这两种工作状态之间的切换。

1. 波形数据的加载

单片机系统上电自检完毕后，开始进行波形数据加载过程。此时，地址总线控制模块和数

据总线控制模块，将总线的控制权交给单片机系统。在该过程中，EPROM 处于读状态，而 SRAM 为写状态。8031 按照 EPROM、SRAM 的时序要求，将 8 K 的波形数据从 EPROM 加载到 HSRAM 中。该过程大概需要几毫秒时间。

由此我们知道，用这种方法不仅能够合成标准波形（如：正弦波、方波、三角波等），而且还可以合成各种非标准波形。对此我们只要通过数据采集器或 PC 机获得 8 K 的波形数据，然后存入到 EPROM 中，就可以按所需要的频率输出相应波形。

2. 波形合成电路的设计

当波形数据加载完毕后，系统就可以进行信号合成。单片机将接收到的频率值转换成频率控制字，送到相位累加器。相位累加器在每一个时钟周期进行相位累加，然后将每次的累加和作为地址去寻址 SRAM，读出与该地址所对应的波形幅度值，然后送到 D/A 转换器转换成模拟信号，最后经幅度放大、滤波输出。

(1) 频率值的接收与显示

键盘、显示部分用来实现用户与单片机的交互。系统采用中断查询的方式接收通过键盘输入的频率值。该频率值一方面送到数码显示接口进行显示，另一方面转化成频率控制字送往相位累加模块。键盘显示接口部分如图 9.14－3 所示，图中虚线框内部分均由 CPLD 实现。

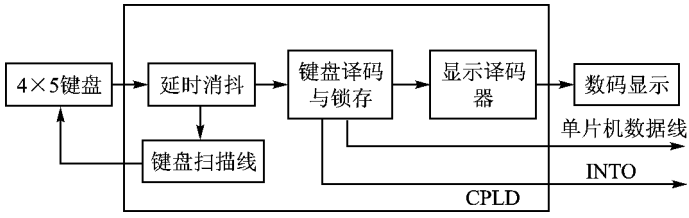


图 9.14－3　键盘显示电路

(2) 数控振荡源 (NCO) 设计实现

这一部分是 DDS 信号合成中的关键部分，由 DDS 系统原理框图 (9.14－2) 可知，这一部分主要是由相位累加器、地址总线控制器、数据总线控制器与 SRAM 组成。其中，除了 SRAM 外，其余 3 个模块都是在 CPLD 上实现。

相位累加器是整个 DDS 系统运转的关键，它设计的好坏直接影响到整个系统的功能和如图 9.14－4 所示。它实质上是 1 个带反馈的 32 位加法器，性能。把输出数据作为另一路输入数据和从微处理器送来的频率控制字进行连续相加，产生有规律的 32 位相位地址码。设计中采用流水线技术实现 32 位加法器，通过在组合逻辑之间插入触发器，降低了寄存器之间的传输延时，从而保证系统能够在较高的时钟速度下运行。

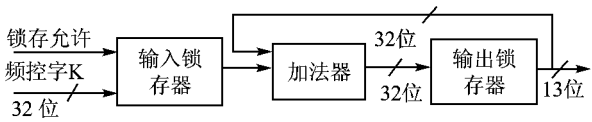


图 9.14－4　相位累加器设计

地址总线控制模块和数据总线控制模块是根据系统工作状态的不同,对系统的地址总线、数据总线以及控制线进行切换,这一部分的设计比较容易实现,这里就不再赘述。

(3) 输出信号调理部分

这一部分是由 D/A 转换器、幅度放大器和滤波器构成,其电路如图 9.14-5 所示。

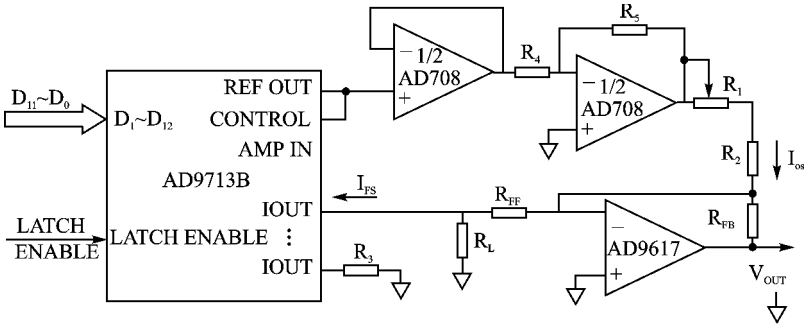


图 9.14-5 D/A 转换与幅度调节电路

DA 器件选用的是 AD 公司的高速芯片 9713B,该芯片的输入是 12 位的。幅度调节电路是由放大器组成。这是 1 个电流反馈的高速放大电路。它把 DA 输出的电流转换成电压,通过反馈电阻 R_{FB}的电流决定 9617 输出的幅度。R_L 和 R_{FF}起分流作用,限制用于 I/V 转换的电流,同时在 9617 内部提供一个输出电压幅度。流过 R_2 的电流给 9617 输出端提供一个直流偏置,调节 R_1 的阻值可以调整偏置电流的大小。整个放大电路最大的幅度是 ±4.096 V。模拟输出的最后部分是滤波电路,滤波器的选择主要取决与系统所要输出的波形。譬如我们在用 DDS 技术合成正弦信号时,可以选用椭圆滤波器滤波。

四、结 语

与传统的频率合成方法相比,DDS 合成方法具有频率切换快、频率分辨率高、相位变化连续等一系列突出优点。使用单片机灵活的控制能力以及良好的人机对话功能与 CPLD 的高性能、高集成度相结合,能够突破传统设计中的许多设计瓶颈,使系统性能大幅度提高;同时,用这种方法实现的 DDS 电路具有很大灵活性,它可以根据用户的需要设计,满足用户的特殊要求。因此,该系统具有很好的开发、应用前景。

同时,我们也应该注意到由于 DDS 数字化实现的固有特点,像相位累加器的相位舍位、波形幅度量化和 DAC 器件非理想特性,使得输出信号频谱杂散较大。当合成信号的输出频率比较高时,表现得尤为突出,从而限制了输出信号的频率范围。对此,我们一方面在设计过程中应尽量减小能够引起杂散的各种因素,另外更重要的是采取一些便于 CPLD 实现而同时能够有效降低输出杂散的技术,如对 DDS 相位累加器的改进[2]、ROM 数据压缩[3]、使用抖动注入技术[4]等。从而使开发出的 DDS 系统性能更加优良。

参 考 文 献

1 ANALOG DEVICES,K 12_bit,100MSPS D/A CONVERTERS

2 Nicholas H T,III H. Samulei. An analysis of the output spectrum of Direct Digital Frequency Sythesizers in the presence of phase—accumulator truncation,IEEE Proc. 41st AFCS,1987:495~502

3 Nicholas H T,III H. Samulei,Kim B. The optimization of direct digita frequency synthesizer performance in the presence of finite word length effects,IEEE Proc. 42th AFCS,1988:357~363

4 Vankka J. Spur reduction techniques in sine output direct digital synthesis,IEEE Proc. 50th AFCS,1996:951~959

5 张厥盛,曹丽娜. 锁相与频率合成技术. 西安:电子科技大学出版社,1995

6 周国富. 利用 FPGA 实现 DDS 专用集成电路. 电子技术应用,1998(2):49~51

选自《现代电子技术》月刊,2002 年第 11 期

9.15 基于 Verilog-HDL 的轴承振动噪声电压峰值检测

太原理工大学 常晓明 谢 刚 李媛媛
大连科汇轴承仪器有限公司 孙连贵

在轴承生产行业中,轴承振动噪声的峰值检测是一项重要的指标。以往,该检测都是采用传统的模拟电路方法,很难做到 1: 1 地捕捉和保持较窄的随机波形的最大正峰值。本文叙述了基于 Verilog－HDL 与高速 A/D 转换器相结合所实现的快速轴承噪声检测方法。

一、振动噪声电压峰值检测方案的确定

1. 轴承振动噪声的产生及检测

图 9.15－1 是轴承振动噪声电压峰值检测系统的示意图。由于加工设备、技术、环境等因素的影响,生产的轴承都程度不同地带有伤疤。图 9.15－1 中,假设某待测轴承有一处伤疤。由于伤疤的存在,轴承在转动过程中,伤疤将与滚珠产生摩擦,从而表现在轴承整体产生微小的振动。这一振动通过加速度传感器输出电压信号,经电荷放大器、峰值检测后,即可得到振动噪声的峰值电压。图 9.15－2 给出了在有伤疤情况下的传感器输出电压波形。

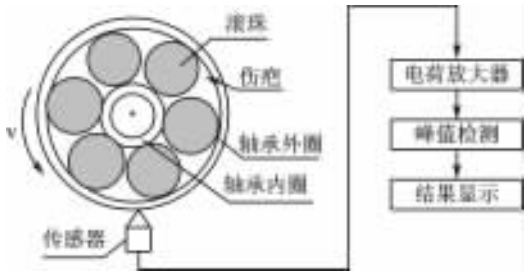


图 9.15－1 轴承振动噪声检测系统结构图

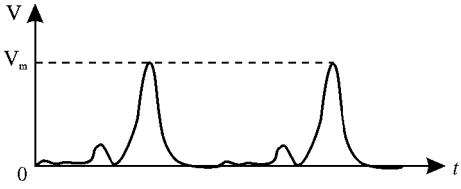


图 9.15－2 由轴承的伤疤所产生的噪声电压波形

2. 模拟式的峰值电压保持电路

以往的轴承振动噪声峰值电压检测,均采用了模拟式的峰值电压检测法。图 9.15－3 示出了由采样保持电路 LF398H 构成的该类检测电路。当噪声电压到来后,采样信号跟随模拟信号电压到峰值处,之后采样脉冲消失,电路处于保持状态。保持电容 C 上即存储了模拟信

号的峰值电压 V_m。要想较快地跟随输入电压 V_{in} 的变化，保持电容 C 的容量就应相对减小；而 C 的相对减小，又会导致在保持电压期间，输出电压 V_{out} 的下降速率加快。这两者相互矛盾，从而使这种电路难以达到较高的性能。

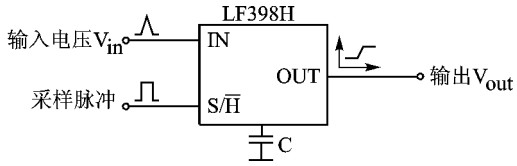


图 9.15－3　由 LF398H 构成的峰值检测电路

3. 数字式的峰值电压检测

模拟式的峰值检测电路不易做到高速采样。采样保持电路经长期使用后，多方面的性能会发生明显变化，且不易批量化生产；而由数字电路组成的系统可以做到结构简单、调试方便，长期使用不会导致系统性能指标的下降。图 9.15－4 是一种数字式的峰值检测系统的组成方案。它由 A/D 转换部分和数字电压的峰值检测部分组成，接口电路内含微处理器，负责与微机进行数据通信和接收来自微机的控制信号，并控制检测系统的工作。根据应用对象的不同，A/D 转换器的采样速率可高达上百 Msps[1]，并可自带采样保持电路。与 A/D 转换器相接的数字电压峰值检测电路可采用 FPGA，其工作速度也可达上百 Msps。因此，在信号的处理速度方面两者都是优于传统的模拟电路方式的。

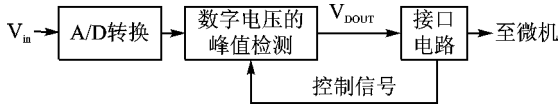


图 9.15－4　数字化处理方式的峰值检测系统组成

二、基于 Verilog-HDL 的峰值电压检测方案

1. 逻辑功能的设计

图 9.15－5 给出了数字电压峰值检测框图。图中除了 A/D 转换器外，虚线部分所示均为 FPGA 组成的功能模块。其功能由 Verilog－HDL（HDL：硬件描述语言）来实现[2]。工作原理如下：由 A/D 转换器取得的数字电压送入数据缓冲模块 GET_DATA，GET_DATA 中的数据与来自数据存储模块 DATA_MEM 中的数据都送入数据比较模块 DATA_COMP 进行比较。如果 X 端的数据大于 Y 端的数据，比较标志模块产生标志信号，同时该信号将 X 端的数据打入数据存储模块 DATA_MEM 中（系统复位后，DATA_MEM 中的数据为最小值 0），进而实现了保持 2 个数据中较大的一个之功能。当振动噪声电压经 A/D 转换器转换成数字电压后，数据存储模块便依 A/D 转换的次数做相应次的比较，最终将噪声电压的峰值检测并保持下来。V_{DOUT} 为数字式的峰值输出电压。

仅有图 9.15－5 的逻辑功能框图还不能方便地用 Verilog－HDL 来描述。为此将其进一步细化为图 9.15－6 所示的形式。图 9.15－6 中虚线框内的功能由 XC9572（Xilinx 公司的产品）实现。图 9.15－6 中，V_{in} 为模拟电压的输入，V_{DOUT} 为数字峰值电压的输出，V_{DOUT}、$\overline{RB1}$、$\overline{RB2}$ 均与接口电路相接，$\overline{RB1}$、$\overline{RB2}$ 受微机的控制。

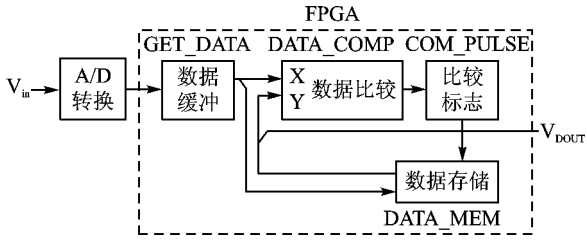


图 9.15－5 数字电压峰值检测逻辑框图

2. 时序图

图 9.15－7 为图 9.15－6 所示逻辑电路的时序图。按照轴承检测的工艺，当系统复位 $\overline{RB2}$、启动脉冲 $\overline{RB1}$ 到来后，经 0.7 s 的延时，便产生 1 个宽度为 1 s 的门脉冲 G_P。在此期间，A/D 转换器连续转换的数据送入数据缓冲器 GET_DATA，之后进行数字信号的峰值检测和保持。A/D 转换器在此采用 MAX120。该转换器的分辨率为 12 bit，转换时间为 1.6 μs。

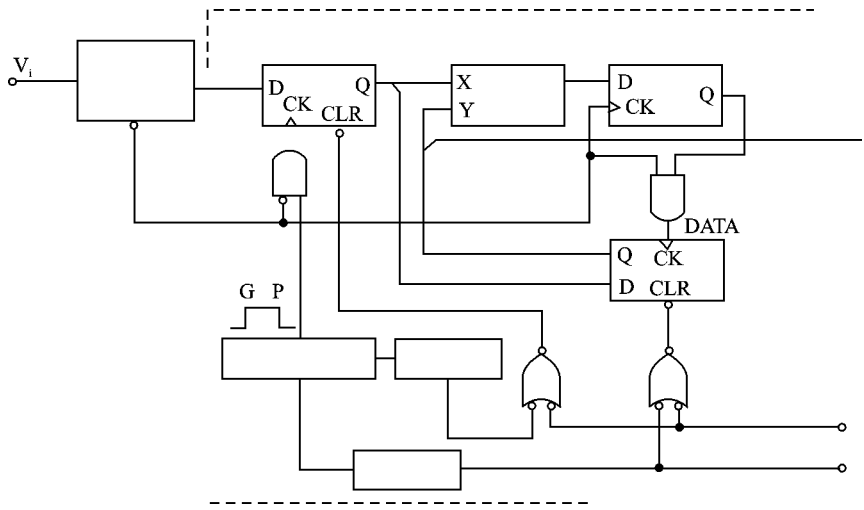


图 9.15－6 数字电压峰值检测逻辑电路图

3. 逻辑仿真

在硬件电路实现之前，用 Verilog-HDL 对图 9.15－6 所示的逻辑电路进行了仿真，图 9.15－8 即为仿真结果。从仿真结果中可以看出，系统复位后，D_OUT (V_{DOUT}) 输出为 0，在 1 s 门脉冲 G_P 有效期间，GET_DATA 接收时钟 GET_DATA_CLK。此间来自 A/D 转换器的数字电压 (分别为 FROM_ADC = 10、15、18、17、4、6、2) 相继输入至 GET_DATA。由于这期间的最大值为 FROM_ADC = 18，故有 D_OUT = 18。在门脉冲 G_P 无效期间，即使有数据 FROM_ADC = 11 输入，仍有 D_OUT = 0。

4. Verilog-HDL 主模块

限于篇幅，这里只将本系统所涉及到的 Verilog-HDL 的主模块部分列出：

```
Module  PK_SEL  (BUSY, RB1, RB2, FROM_ADC, D_OUT, P_OUT);
    input    BUSY, RB1, RB2;
    output   P_OUT;
```

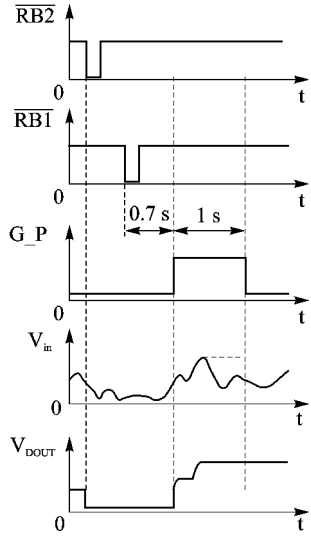


图 9.15－7 峰值电压检测时序图

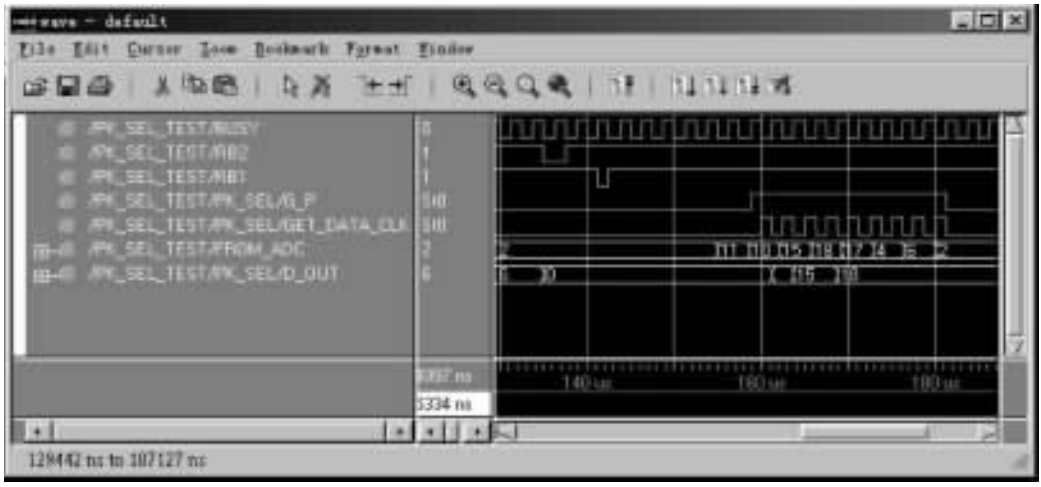


图 9.15－8 峰值电压检测逻辑仿真结果

```
    input      [11: 0] FROM_ADC;
    output     [11: 0] D_OUT;
    wire       [11: 0] TO_COM;
    wire       GET_DATA_CLK;
//产生秒脉冲
    CNT100     F_4kHz   (RB1, BUSY, F_4k);          //分频
    CNT100     F_37Hz   (RB1, F_4k, F_37);          //分频
    CNT5       F_1Hz    (RB1, F_37, F_7);           //分频
    DELAY_P1   START_DLY (RB2, RB1, F_7, DLY_05S);  //延时0.7 s
    DELAY_P2   GENE_SPB (RB2, DLY_05S, F_7, SPB);   //延时1 s
    GETE_GENE  GENE_GP (G_P, DLY_05S & RB2, SPB);
```

```
                                                    // 1s 的门脉冲
    assign      P_OUT = G_P;
                                    // ADC 数据最大值的比较和检测
    assign      GET_DATA_CLK = ~BUSY & G_P;
    DFF12       GET_DATA  (GET_DATA_CLK, FROM_ADC, TO_COM, ~SPB & RB2);
                                    // 获取 ADC 数据
    COMP_D      DATA_COMP (TO_COM, D_OUT, D_S);
                                    //数据比较
    DFF12       DATA_MEM  (BUSY & D_S, TO_COM, D_OUT, RB1 & RB2); //数据存储
endmodule
```

三、结束语

与模拟式的峰值电压检测方式相比,数字式的检测方式有着结构简单、系统开发周期短等优点,而采用 Verilog - HDL 可以方便地实现欲有的功能。笔者设计开发的该系统用在了大连科汇轴承仪器有限公司生产的 S0910 - 3 型轴承振动测量仪中,并于 2001 年 6 月在上海的国际轴承及装备博览会上引起了同行的关注。

参 考 文 献

1 高光天. 模数转换器应用技术. 北京 科学出版社,2001

2 王毅平,张振荣. VHDL 编程与仿真. 北京 人民出版社,2000

选自《单片机与嵌入式系统应用》月刊,2002 年第 12 期

第十章

综合应用

10.1 AVR 高速单片机 LED 显示系统

绵阳西南科技大学信息与控制工程学院 (621002)
李驹光 张 华 李众立

LED 点阵显示器亦称 LED 矩阵板,具有亮度高、发光均匀、可靠性好、接线简单、拼装方便等优点,能构成各种尺寸的大屏幕显示器。因此,它被广泛应用于大型 LED 智能显示屏、智能仪器仪表和机电一体化设备的显示单元中,取得了较好的效果。目前。LED 显示屏尽管已被广泛使用,但存在系统复杂、成本较高等缺点,不利于推广使用。对国内广大用户来讲,迫切需要一种比较经济、小型的显示系统,同时要求使用方便灵活。为此我们利用 AVR 高速单片机组成前级驱动电路,PC 用于后级管理和控制,方便地组成了由多块大屏幕 LED 显示器构成的显示系统。该系统可广泛用于商场、车站、码头及其他公共场合。

一、系统功能

该系统能方便地显示各种点阵、各种字体的汉字信息和简单的图形信息。它的显示内容可以滚动或翻屏的方式实现上下、左右移动,显示内容可由上位 PC 按用户要求随时修改,并由串行通信口传送至前级驱动电路保存,断电后存储器里的内容不丢失 (在实际应用中可使用带掉电保护电路的 RAM 或 Flash EPROM),一台 PC 可控制多面显示屏,使用灵活方便。显示屏工作电压为直流 5 V,耗电量低,显示稳定,受外界影响小。

二、硬件组成

系统的硬件电路由以下三部分组成:LED 显示屏、以 AVR 高速单片机为核心的驱动电路和 PC 及与单片机的通信电路。系统硬件组成如图 10.1-1 所示。

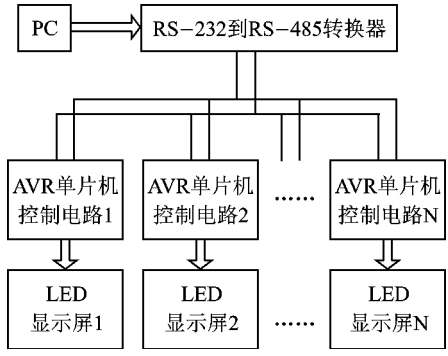


图 10.1-1 系统硬件组成

1. LED 显示屏

LED 显示屏由 LED 点阵显示器 (常见型号为 P2158A) 构成。它是以发光二极管为像素,

按照行与列的顺序排列而成的显示器件，采用逐行（或逐列）扫描方式工作，由峰值较大的窄脉冲驱动，从上到下逐次不断地对显示屏的各行进行选通，同时又向各列送出表示图形或文字信息的脉冲信号。反复循环以上操作，就可显示各种图形或文字信息。

2．以单片机为核心的驱动电路

以 AVR 高速单片机为核心的驱动电路如图 10.1－2 所示。

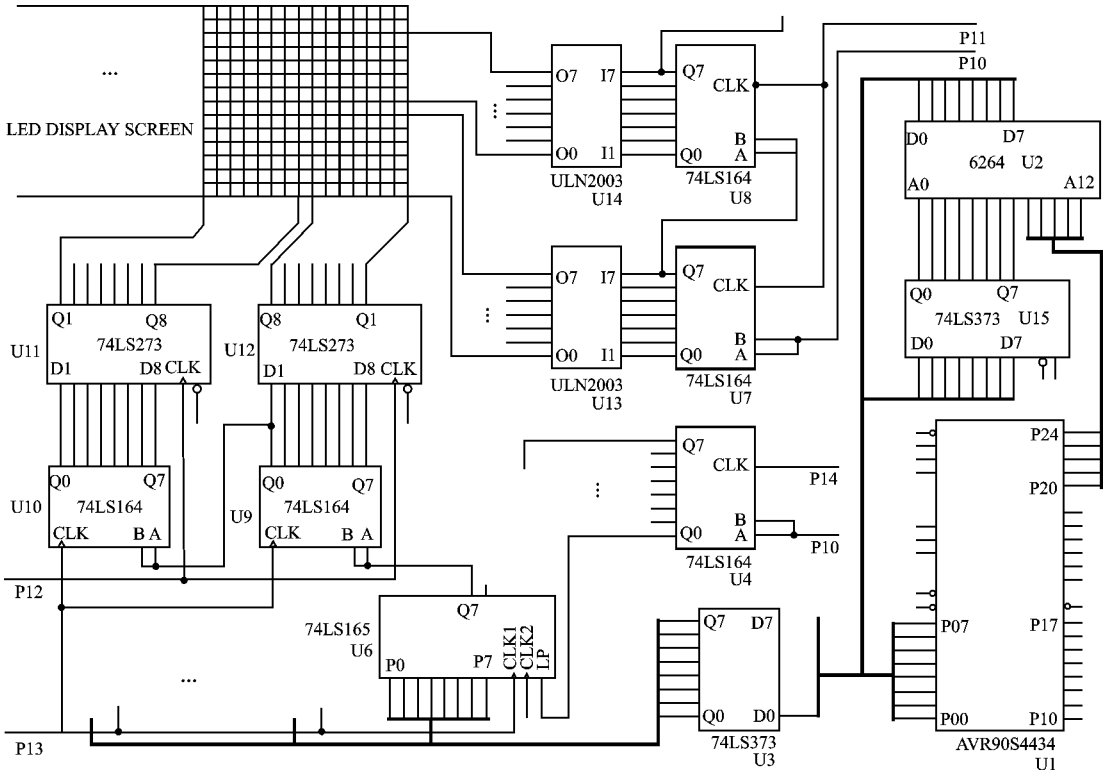


图 10.1－2　以 AVR 高速单片机为核心的驱动电路

工作原理如下：首先从 RAM 中读出待显示的信息，一次读取一个字节，传送到数据总线上，由 U3 锁存，然后将该字节数据锁入 U6 中，再由 P13 口送出八个移位脉冲，将数据传送至下一级的 U9，经 U12 驱动后作为列信息输出。

在一列信息输出完成后，由行驱动电路中的 U7 输出一个行脉冲，经 ULN2003 反相后作为行选通信号，即完成一行信息的显示。

列信息不断送出，行信号轮流选通，这样反复循环，即可在显示屏上得到稳定的信息。

在该电路图中，行、列的驱动电路都只画出了其中的一部分，其余部分与之相似。注意到 U9 和 U10 级连（也可以多片级连），此时，一片 U6 即可负责 U9 和 U10 的数据传送。采用这种方式，一是可以简化硬件电路，二是可以使用较少的指令完成较多数据信息的传送，有利于系统的设计。

对于存储器，该系统选用 6264，也可以根据系统的需要选用更大容量的存储器，以便可在存储器中同时存放多屏信息，定时轮流显示。

在单片机的选型上，考虑到系统功能与电路的简洁，采用 Atmel 公司推出的全新配置的精

简指令集(RISC)单片机 AT90S4434,该单片机内含 EEPROM 及 Flash 存储器,并可通过 SPI 接口,支持系统编程,使用非常方便。

3. PC 及与单片机的通信电路

上位 PC 用于在线修改显示信息。在实际使用中,考虑到系统成本,可与现场的原有计算机合用,而不单独设置(若显示信息长时间固定不变。在系统中亦可不设置上位 PC,直接将待显示信息写入 EPROM 中)。当需要修改显示信息时,由 PC 向 AVR 单片机系统传送新的显示信息,以刷新 RAM 中的原有信息,考虑到在现场中 PC 与由 AVR 单片机控制的显示屏一般相隔较远,若采用传统的 RS-232C 标准方式通信,则无法实现信息的可靠传输。在实际使用中采用 RS-485 接口标准,该标准采用双绞线传输,抗干扰能力强,传输速率高,传输距离远。在一点对多点的通信系统中,效果较好。

三、系统的软件组成

系统软件由两部分构成:用 Visual Basic 6.0 编写的上位 PC 控制及通信软件和用 AT90S4434 汇编语言编写的显示屏控制软件。

1. 上位 PC 控制及通信软件

在此系统中,上位 PC 控制软件用于对显示信息进行编辑和对汉字字模进行处理。用 Visual Basic 6.0 编写。首先输入待显的汉字信息,然后根据汉字编码从汉字字库中取出相应点阵和字体的字模信息,再把点阵根据显示屏所需的格式进行重排,最后通过串行口发送给 AVR 单片机控制电路的 RAM 中。这样,单片机在控制显示时,只需顺序读取 RAM 中的信息即可,有利于 AVR 单片机控制软件的编写,同时也有利于显示系统的稳定工作。

例如要显示"西南科技大学",二十四点阵,仿宋体,从字库中取出它们的点阵信息,由于二十四点阵的排列是从上到下,根据目前的显示方式,必须把它们重排为从左到右的顺序,然后再发送。若需显示图形信息,可先用相应的工具(如用 Windows98 的画图)编辑,读取点阵信息重排后发送到 RAM。

通信功能利用 Visual Basic 6.0 提供的 MSComm 通信控件,以多机通信方式,通过 PC 串行口发送信息,由不同的地址区分不同的显示屏。注意与下位机保持通信协议一致。

2. 显示屏控制软件

显示屏控制软件用 AT90S4434 汇编语言编写,由主程序、定时中断服务程序和串行口中断服务程序三部分构成。主程序完成必要的初始化工作,串行口中断服务程序接收从上位 PC 发来的待显信息并存放到 RAM 中,在定时中断服务程序中完成对显示屏的扫描和控制。

AVR 单片机从 RAM 中读出待显示的信息,一次读取一个字节,传送到数据总线上,由 74LS373(U3)锁存,然后由 P10 口经 74LS164(U4)送出一锁存脉冲至 U6 的$\overline{PL}$端,将该字节数据锁入 74LS165(U6)中,再由 P13 口送出八个移位脉冲,将数据由 U6 串行传送至下一级的 74LS164(U9)。以相同的方式传送第二字节至 U9,同时,原 U9 中的数据则移至 U10 中,最后由 P12 口送出脉冲将两片 74LS164(U9,U10)的数据锁入 74LS273 中(U11,U12),作为列信息输出。以同样的方式将一行信息全部锁入对应的 74LS273 中。

一列信息输出完成后,由行驱动电路中的 74LS164(U7)输出一个行脉冲,经 ULN2003 反相后作为行选通信号,即完成一个行信息显示。列信息不断送出,行信号轮流选通,这样反复循环,即可在显示屏上得到稳定的信息。程序流程如图 10.1-3 所示。

该软件中，中断时间常数的确定较为重要，根据人眼的视觉暂留时间，若每秒显示二十四帧以上，便可得到稳定的显示，取每秒二十五帧，即把中断时间设置为 40 ms，那么，只要每次执行中断服务程序都完成一次对全屏的扫描，将得到较为稳定的显示。

从理论上讲，显示屏的大小是任意的，但从上面的分析可知，显示屏越大，往显示屏上所送的数据就越多，即中断服务程序也将越长，然而，40 ms 的中断时间却是固定的，多于 40 ms 会有闪烁感。因此，在设计显示屏的大小时，该因素必须考虑，正因如此，AVR 单片机的高速性在此非常重要。当然，也可以通过适当提高晶振频率或选用更高性能的机型（如高性能 DSP），以尽可能地增大显示有效面积。

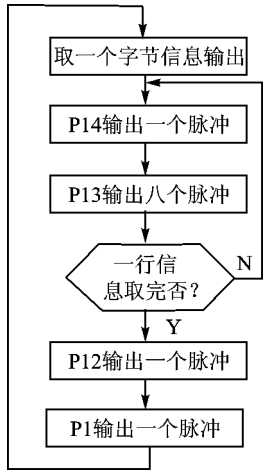


图 10.1－3　控制程序流程图

上述系统线路简单，所使用的器件均为常用集成电路，可方便制作。使用以上方式，我们曾制作了一块约 2 m^2 的显示屏，经过较长时间的运行，效果良好。如果显示屏更大，则应考虑电路的负载能力，可以增加驱动电路（如 74LS244，74LS245 等）或变拉电流负载为灌电流负载。

若使用彩色 LED 点阵显示器，只需在硬件和软件上作适当的变化，便可构成彩色显示屏，则显示效果更好。

参 考 文 献

1　耿德根. AVR 高速嵌入式单片机原理与应用. 北京 北京航空航天大学出版社，2001

选自《电子技术》月刊，2002 年第 5 期

10.2 基于 ADμC812 与 SJA1000 数据采集系统的设计

中国科学院合肥智能机械研究所 (230031)
卞亦文 吴忠城 戈 瑜

现场总线是应用在生产现场,实现各种现场智能化测控设备之间双向、串行和多节点数字通信的技术。现场总线技术发展的必然结果是改变了传统的控制系统的结构,形成了新型的网络集成式全分布控制系统,采用全数字通信,具有开放式、全分布、可互操作性等特点,形成了从测控仪表到操作控制计算机的全数字通信网络,具备了与数字网络互连的条件,顺应控制网络的发展。

本文在讲述 CAN 总线特性的基础上,结合 ADμC812 单片机的特性,提出了一种具有 CAN 总线接口、RS232 串口,并具有可在线调试参数特性的数据采集系统的设计,详述了系统的硬件结构、软件设计以及系统的工作原理。

一、CAN 总线技术特点

CAN (Control Area Network,控制局域网) 总线属于现场总线的范畴,最初是由德国 BOSCH 公司为汽车监测、控制系统而设计的。它是一种多主总线,通信介质可以是双绞线、同轴电缆或光纤,通信速率可达 1 Mbps ;这样既可保证信息处理的实时性,又可以构成多主系统,而保证了系统的可靠性。同时,CAN 总线采用短帧结构,且每帧信息都有 CRC 检验并提供相应的错误处理功能,保证了数据通信的可靠性。另外,CAN 总线中的数据段长为 8 B,可以满足一般工业控制中的应用 ;并且 8 B 不会占用总线时间过长,可以满足数据通信的实时性的要求。

二、SJA1000 CAN 总线控制器性能特点

SJA1000 是一种独立的 CAN 总线控制器,是 Philips 公司生产的作为 PCA82C200 CAN 总线控制器的替代产品,与 CAN2. B 完全兼容,同时支持 Basic CAN 与 Peli CAN 两种工作模式。该控制器内部集成了发送与接收缓冲区 ;并且具有位流处理器,主要用于处理发送缓冲器与 CAN 总线之间的控制数据流,以及错误的检测、仲载、总线填充和错误的处理 ;位时序逻辑监处理与总线有关的位时序 ;错误管理逻辑负责传送层中调制器的错误管制,接收 BSP 的出错报告,促使位流处理与接口管理逻辑进行错误统计。另外,SJA1000 还增加了如支持热插拔、只听模式 (总线监听,无应答) 等功能。

三、ADμC812 单片机的性能特点

ADμC812 是 AD 公司最新推出的 12 位数据采集芯片。片内集成了 8 路 12 位高性能的自校准 ADC,2 路 12 位 DAC 以及可编程 (与 8051 兼容) MCU。该芯片包括 8 KB 闪烁/电擦除 (Flash/EE) 程序存储器 (可通过串口下载程序) 、640 B 闪烁/电擦除 (Flash/EE) 数据存储器、256 B 内部 RAM、外部数据空间 16 MB,具有看门狗定时器 (WDG) 、电源监视器 (PSM) 、I^2C/

SPI 和标准的 UART 串性端口。ADμC812 的模拟接口功能,具有 8 路 12 位高精度自校准 ADC、高速率 200 KHz、高速 ADC－RAM 的 DMA 控制器等特性,使其特别适于瞬时捕捉系统、智能传感器、数据采集系统等。

四、系统结构

1. 系统总体结构

传统的工业现场控制系统中通信方式存在硬件结构复杂、现场布线困难、扩展能力低等不足之处。这里采用 CAN 总线作为通信网络连接各个节点,主控制器和各个现场智能仪表都连接在 CAN 总线上。数据采集系统以 ADμC812 单片机为核心,包括各种信号调理电路、传感器、执行器、CAN 控制器(如 SJA1000)和 CAN 驱动器(如 82C250)等,提供 CAN 总线接口,并保留传统的 RS232 串口。系统结构如图 10.2－1 所示。

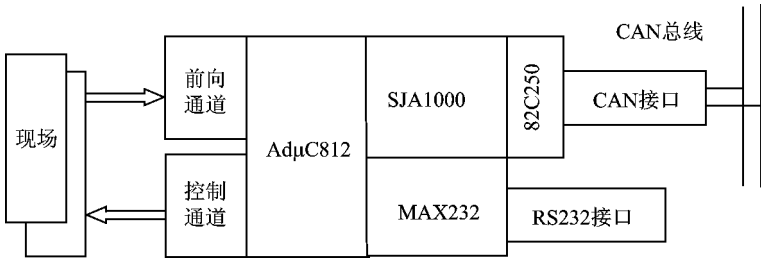


图 10.2－1 数据采集系统总体结构

系统中前向通道和控制通道中包括相应的信号调理电路和传感器、执行器。该系统在 ADμC812 单片机的控制下,通过控制通道对现场进行控制;并且通过前向通道采集的数据可以在现场显示,也可以保存在大容量的 Flash 存储器中以便以后使用;系统采集的数据可以通过 CAN 总线发送到上位机或网络上。由于 ADμC812 具有在线可编程的特性,通过 RS232 接口可以实现在线修改系统的工作方式、采样频率等有关数据采集的参数。

以 ADμC812 单片机为核心的数据采集系统具有高性能的数据采集能力,包括现场信号的调理、高速数据采集、信号处理等;具有灵活强大的通信能力,可以通过 CAN 总线接口方便与现场总线连接,从而实现远程数据采集和控制;具有灵活的在线编程的功能,可以在线修改数据采集的系统参数;并且由于 ADμC812 具有看门狗电路、电源监视等功能,该系统可以在强电磁干扰、波动以及恶劣的环境下正常工作。

2. SJA1000 的接口电路

SJA1000 在 ADμC812 的控制下进行信息的接收与发送,其通信主要是通过中断控制的。SJA1000 的接口电路包括其与 ADμC812 的连接和 CAN 总线的连接,如图 10.2－2 所示。

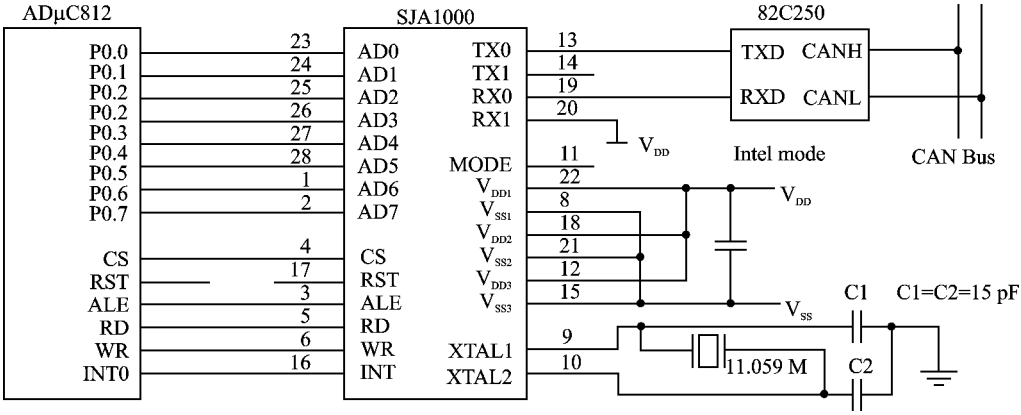


图 10.2－2 SJA1000 接口电路

五、软件设计

数据采集系统的软件设计采用模块化的设计方法，系统软件主要包括数据采集与处理软件、CAN 总线通信软件两大部分。

1. 数据采集与处理软件

数据采集处理部分主要是各种执行器的控制算法以及各种传感器信号的采样与控制算法，并将采集的数据进行处理，涉及到数据滤波，信号的补偿等；然后在微控制器的控制下将数据传送到 CAN 控制器发送缓冲区中。由于 ADμC812 是与 8051 兼容的内核，原来 8051 的用户使用非常方便。但这里特别注意 ADC 的采样周期，在一个采样周期中完成信号的采样、处理与发送，然后再采样下一个通道的信号，如此循环来进行多通道信号的采样。

2. CAN 总线通信软件

CAN 总线通信是根据 CAN 2.0B 协议进行的，CAN 通信协议的实现，包括各种通信帧的组织发送，是由集成在 CAN 控制器中的电路来实现的。利用 SJA1000 可以方便地设计出智能化的通信程序模块，如对本模块自测、通信波特率侦测、通信模块错误处理等部分。这里主要讲述 SJA1000 的初始化以及各种信息帧的发送与接收的实现。

⑴ CAN 总线控制器的初始化

SJA1000 的初始化部分要特别注意模式寄存器 MOD、接收代码寄存器 ACR、接收屏蔽寄存器 AMR、时钟分频器 CDR、总线时序寄存器 BTR0、BTR1 以及输出控制寄存器 OCR 的初始化设置。主要确定各种通信参数，如波特率、位周期宽度、采样点位置、采样次数、输出方式等。

⑵ CAN 总线信息的发送与接收

信息的发送与接收是由 SJA1000 自动完成的。SJA1000 提供了两种数据操作模式，即中断模式和状态查询模式。由于状态查询模式需要大量的 CPU 的开销，在实时性要求较高的系统常采用中断模式，这里采用中断模式。无论是信息的发送还是信息的接收都是采用中断程序控制的。通信模块中断程序流程如图 10.2－3 所示。

从 CAN 控制器发送信息到 CAN 总线是由发送中断程序控制，发送程序把信息发送到 CAN 控制器发送缓冲区，启动发送命令；从 CAN 总线发送信息到 CAN 控制器是由接收中断程序控制，接收程序从 CAN 控制器接收缓冲区读取数据。

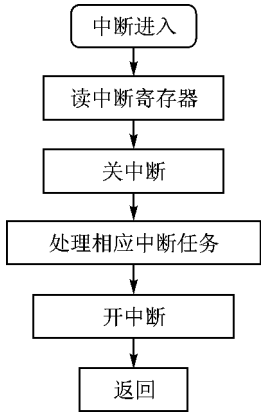


图 10.2－3 通信模块中断程序流程图

六、结 论

本文提出的基于 ADμC812 与 SJA1000 的数据采集系统具有 CAN 总线接口,且保留了 RS232 串口;不仅可以实现信息的远程传输,而且可以实现系统的在线调试;有利于实现现场智能设备的即插即用,以及数据采集系统的模块化、分布式的系统结构的转变。本文提出的数据采集系统的设计思想也可以应用于其他工业自动化场合,实现现场仪表的远程监控和操纵。其应用前景十分可观。

参 考 文 献

1 邬宽明. CAN 总线原理与应用系统设计 [M]. 北京:北京航空航天大学出版社,1999
2 刘书明. 数据采集芯片 ADμC812 原理与应用 [M]. 西安:西安电子科技大学出版社,2000
3 卞亦文等. 一种基于 CAN 总线的机器人感知系统接口模块的设计 [J]. 自动化博览
4 王军等. CAN 总线应用于铁路微机联锁系统的探讨 [J]. 自动化博览,2001 (8)
5 谢彦辉等. SJA1000 CAN 控制器在数据采集模块中的应用 [J]. 仪表技术,2002 (2)
6 王燕等. CAN－bus 仪表的通信模块设计 [J]. 电测与仪表,2001 (8)
7 SAJ1000 Stand－alone CAN Controller Application Note. Philips Semiconductor, 1997

选自《电测与仪表》月刊,2002 年第 9 期

10.3 用 AT89C2051 设计的 PC/AT 键盘

海军航空工程学院分院 杨日杰 张宗玉
空军 87424 部队 郑立新

在工业控制、测量仪器等领域,已大量使用嵌入式 PC,如 ADVANTECH 公司的 PC/104、AMD 公司的 E86 嵌入式 PC 等。它们除具有 PC 的功能外,还提供了功能强大的各种标准接口,如平板/VGA 显示器控制接口、光驱接口、以太网接口、RS-232/422/485 接口、PC/AT 键盘接口等。这就为新产品开发的标准化、模块化提供了方便,可大大缩小研发周期,降低研制成本,快速进入市场。由于嵌入式 PC 具有标准 PC/AT 键盘接口,也就是说,可以用标准的 PC/AT 键盘来对嵌入式 PC 进行操作与控制。但是,在很多实际应用中,由于一般只用到某几个固定的键,并希望键盘具有体积小巧、便于布放等特点,为此,希望能够设计一种小巧、灵活的 PC/AT 键盘,来满足各种需求。本文介绍一种由 AT89C2051 设计实现的 PC/AT 键盘。

一、PC/AT 键盘的特点

PC/AT 键盘由单片微控制器、键盘矩阵和支持逻辑三部分组成。键盘微控制器的主要功能是扫描键盘,以得到有效的闭合键,一旦键被按下或放开,就为系统板产生键代码,将键代码以串行格式传递到系统板,同时产生将键代码转换为供系统板使用的并行数据所需的时钟信号。AT 键盘使用接通键码,其值在 00~7F 之间,以串行数据格式传递到系统板;每发送一个键码包含 11 个数据位,即 1 个起始位、8 个数据位(低位在前,高位在后)、1 个奇偶校验位、1 个停止位。在键码传送的同时,微控制器还传送 1 个键码时钟同步信号,用于同步键码数据的接收。键码中每个数据位的传送发生在键盘时钟的下降沿,时钟的波特率为 16 Kb/s。图 10.3-1为接通键码是 2C,即按下 t 键时,键码的传送格式。

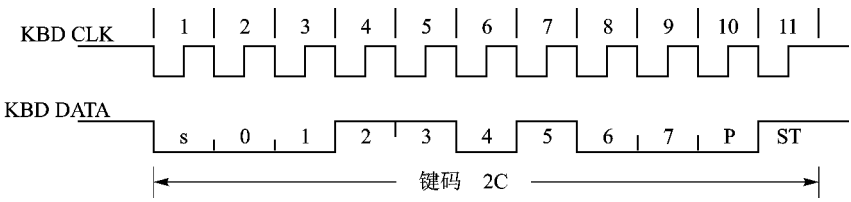


图 10.3-1 AT 键码 2C 传送格式

对于 PC/AT 键盘,如果按下键 0.5 s 之前放开该键,则键盘电路产生一个断开键码,将这个键码也以串行数据的格式传送出去。AT 键盘的断开键码为 F0,在断开键码之后再跟接通键码。其中断开键码通知 BIOS 键盘例程,按下的键序列功能已结束,键已被放开。如果在键按下 0.5 s 之后仍未放开该键,则键盘电路产生一个接通键代码(与接通键码相同),并以每秒 6 个键码的速率(每 166.7 ms 一个键码)进行传送,此过程直到键盘电路检测到断开代码为止。常用键的键码如表 10.3-1 所列。

表 10.3－1　常用键的键码

键	键码	键	键码	键	键码	键	键码
0	45	b	32	m	3A	x	22
1	16	c	21	n	31	y	35
2	1E	d	23	o	44	z	1A
3	26	e	24	p	4D	F1	05
4	25	f	2B	q	15	F2	06
5	2E	g	34	r	2D	F3	04
6	36	h	33	s	1B	F4	0C
7	3D	i	43	t	2C	F5	03
8	3E	j	3B	u	3C	F6	0B
9	46	k	42	v	2A	F7	83
a	1C	l	4B	w	1D	F8	0A

二、硬件设计

键盘电路如图 10.3－2 所示，由 ATMEL 公司的微控制器 AT89C2051、MAXIM 公司的看门狗自动复位电路 MAX813L 及键盘矩阵组成。由于 AT89C2051 的可用端口为 16 个，除复位端 RES、看门狗信号输出端 WDI、键码数据输出端 TXD 和时钟输出端 CLK 外，还剩 12 个可用端口，这样，其最大可独立响应 6 × 6 = 36 个键的输入，可满足工控机常用控制键的要求。MAX813L 为看门狗电路，它实时接收来自 AT89C2051 的 WDI 信号，并自动判断两次 WDI 信号的间隔时间。当时间间隔小于 1.6 s 时，其 RST 输出端保持低电平；当时间间隔大于 1.6 s 时，其 RST 输出端输出高电平，AT89C2051 被复位。AT89C2051 具有如下特点。

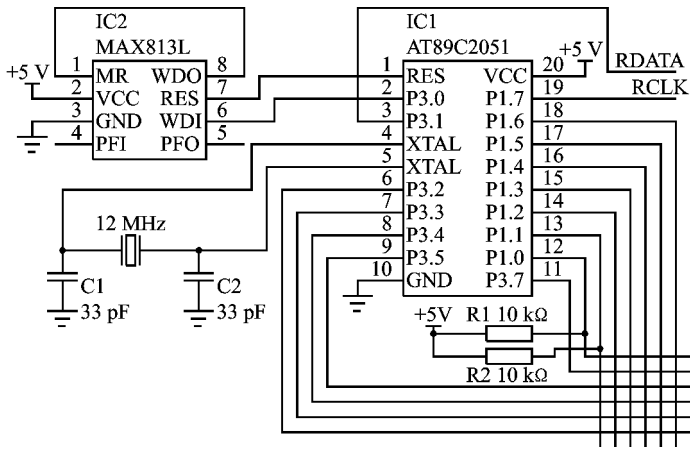


图 10.3－2　键盘硬件原理图

⑴ 具有 2K 字节可编程闪存；

⑵ 128 ×8bit 内部 RAM；

⑶ 15 根可编程 I/O 线；

⑷ 2 个 16 位定时/计数器；

⑸ 6 个中断源；

⑹ 可编程串行 UART。

三、软件设计

软件包括定时器 0 定时中断子程序、定时器 1 定时中断子程序、主程序等。其中，定时器 0 定时中断子程序用于定时检测有无键被按下、判断哪个键被按下并确定对应的键码；定时器 1 定时中断子程序用于确定输出键码和时钟信号的波特率，并定时输出看门狗信号，用于防止软件出现死机现象。主程序根据有无键被按下标志，确定是否输出键码和同步时钟信号。如有键被按下，则调入由定时中断子程序所确定的键码，输出相应的键码并同时输出同步时钟信号。主程序流程如图 10.3－3 所示。

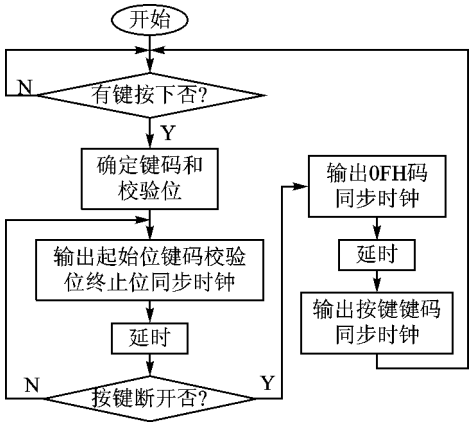


图 10.3－3 主程序流程图

四、设计实例

下面为一设计实例，要求所设计的小键盘输出 F1、F2、F3、F4、Page UP、Page Down、Esc、Enter 8 个 PC/AT PS/2 键盘信号。8 个按键的一端分别接 P3.7、P1.0 ~ P1.6 端口，8 个按键的另一端为公共接地端。全部程序见本刊网站：www.dpj.com.cn

本文介绍的 PC/AT 键盘具有结构简单、设计灵活性强、易于编程、体积小、成本低的特点，并可根据用户需要随意设计和布放，对使用工控机的各种测试与控制仪器具有通用性。

参 考 文 献

1 ATMEL 公司. AT89C2051 Data Sheet

2 MAXIM 公司. MAX813L Data Sheet

3 马忠梅. 单片机的 C 语言应用程序设计. 北京：北京航空航天大学出版社，1999

选自《单片机与嵌入式系统应用》月刊，2002 年第 4 期

10.4 利用 89C2051 实现 POCSAG 编码的方法

海军航空工程学院分院 杨日杰 张宗玉
空军 87424 部队 郑立新

POCSAG (Post Office Code Standardization Advisory Group) 码已广泛用于无线寻呼系统,目前,该系统已遍及全国城乡。其中,寻呼机作为一种接收机,具有灵敏度高、抗干扰能力强、接收误码率低、技术成熟、成本低等优点,因此对其进行二次开发,将其用作无线遥控接收机,可以有效地降低研制成本,加快研制进度,提高产品的性价比。为此,需对 POCSAG 码的编码格式、校验方法、编程实现等进行研究。

一、POCSAG 码编码格式

POCSAG 码是一种同步串行通信编码,属于纠错编码中的 BCH (31: 21) 码加一位偶数检验,可以纠正 2 位随机错误。POCSAG 码由前置同步码和码组构成,其中,一个完整的前置同步码包括 576bit 101010…10 的交替码,用来帮助寻呼机实现位同步,进而达到字同步和码组同步。码组由几批完整的码字构成,每批码字包括由 32 位构成的帧同步码和 8 个帧结构 (0 ~ 7),每帧又有 2 个码字。码字有 4 种:字同步码、地址码、空闲码、信息码。各种码字均为 32 位,其序号为 1 ~ 32。第 1 位最高,先发送;第 32 位最低,最后发送。字同步码是为了便于寻呼机实现帧与字的同步。地址码用于传送地址信息,其第 1 位为 0,用于和信息码区分;第 2 ~ 19 位是 21 位用户地址识别信号中的高 18 位,低 3 位为隐含位,不发送,用于确定 8 帧中的某一帧;第 20 ~ 21 位是功能位,用于表示寻呼机的类型等;第 22 ~ 31 位是纠错编码产生的校验位;第 32 位为偶校验位。信息码用于传送用户信息,其第 1 位为 1,用于和地址信息区分;第 2 ~ 21位是信息位,第 22 ~ 31 位是纠错编码产生的校验位,第 32 位为偶校验位。空闲码用来填充空闲码字,它也由 32 位构成。

二、BCH 编码原理

地址码与信息码的第 22 ~ 31 位是由第 1 ~ 21 位信息生成的 BCH 校验位,对 BCH 码而言,其循环码的生成多项式为

$$d(D) = LCM[m_1(D), m_2(D), \cdots, m_{2t-1}(D)]$$

式中,t 为纠错个数,$m_i(D)$ 为最小多项式,LCM 表示取最小公倍数。BCH 的码长为 $2^m - 1$ 的因子,码长为 $n = 2^m - 1$ 的 BCH 码称为本原 BCH 码;对于纠正 t 个错误的本原 BCH 码,其生成多项式为

$$g(D) = m_1(D) \cdot m_3(D) \Lambda m_{2t-1}(D)$$

其最小码距为 $d = 2t - 1$。

对 POCSAG 码而言,要求码长 31 位。它能纠正 2 个差错,差错数目少于 6 个,则生成多项式为

$$g(D) = m_1(D) \cdot m_3(D) \cdot m_5(D) =$$
$$(D^5 + D^2 + 1) \cdot (D^5 + D^4 + D^3 + D^2 + 1) \cdot$$
$$(D^5 + D^4 + D^2 + D + 1)$$

三、编码器硬件简介

POCSAG 码编码器硬件结构如图 10.4－1 所示。它由单片机 89C2051、电可擦除存储器 24C01、拨码开关 SW1、编程开关 SW2 等构成。输入每个编码时，都是先用拨码开关设置好要输入的码值，然后通过按编程开关确认。每次确认后，发光二极管亮 2 次；全部编程完毕，亮 3 次。

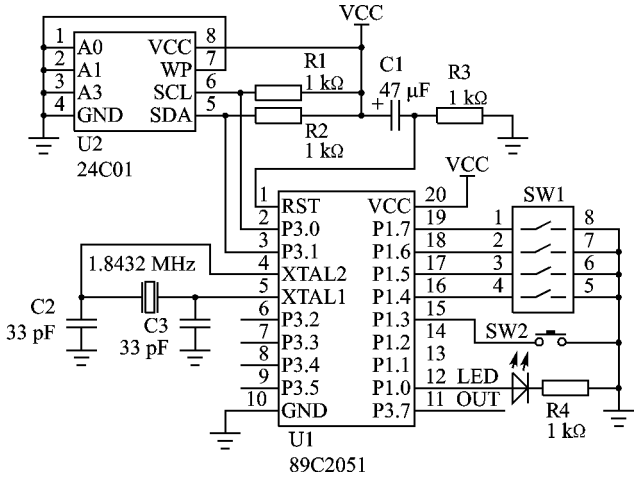


图 10.4－1 编码器硬件结构图

四、程序构成

程序由宏定义、24C01 读/写子程序、BP 机码输入子程序、CRC 校验子程序、输出 POCSAG 码子程序、定时中断子程序、主程序等构成。程序用 C 语言编写，其中几个重要的子程序有：宏定义、24C01 读/写子程序、输入编码子程序、CRC 校验子程序、输出 POCSAG 码子程序的源程序（程序见本刊网站）。

本文介绍的 POCSAG 码编码方法和程序，在实际应用中无错码与误码产生，应用效果理想。

选自《单片机与嵌入式系统应用》月刊，2002 年第 5 期

10.5 加载-感应 DAC 的应用

Maxim 集成产品公司北京办事处 (100083) 迈克·米勒 吴 忠

一、电压缓冲输出 DAC 的类型

图 10.5-1 给出了 3 种常用的带电压缓冲输出的 DAC。第一类是固定输出增益,增益由内部电阻网络确定,典型值为:+1.0,+1.638 或 +2.0V/V;第二类与前者类似,增益也由内部电阻网络确定,只是同相放大器的反相端通过一个电阻后有一个引出端,允许工程师进行失调校正;最后一类是加载-感应输出型,其反相端直接引出 IC 封装,具有更大的灵活性。

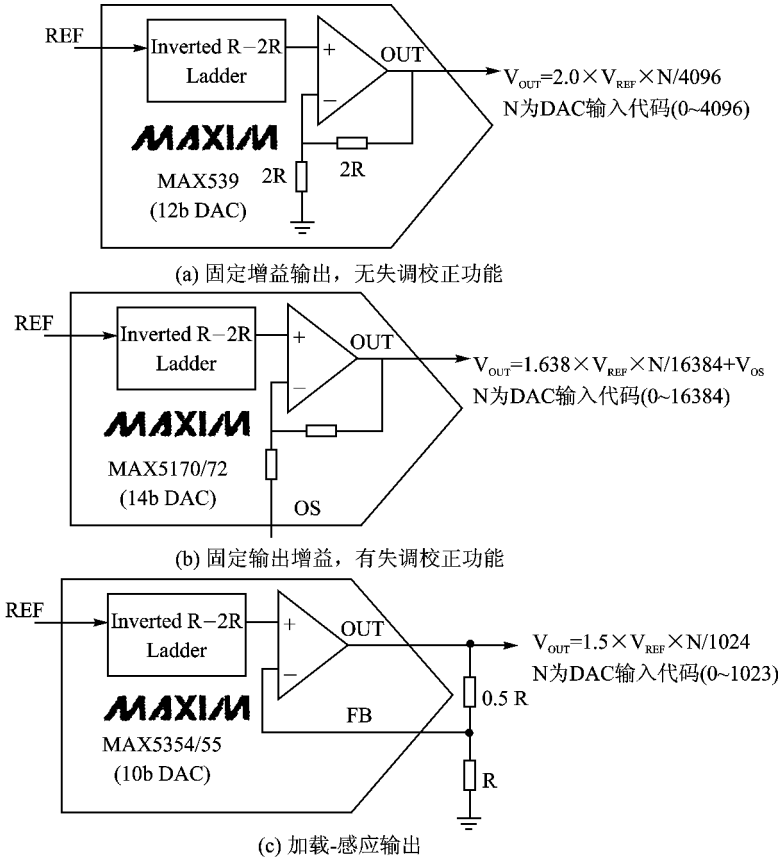


图 10.5-1 DAC 输出类型

第一、第二类 DAC 的主要优点是:内部电阻一致性好,增益误差的典型值小于 1%,增益温漂小于 10×10^{-6}。缺点是:只能提供单一的固定增益,如需不同增益,只能在 DAC 后增加放大器加以改变。第二、第三类 DAC 的失调误差可以通过外部电路修正。第三类 DAC (加载-

感应型) 与第一、二类 DAC 相比还能提供更多的优点,比如,可根据实际应用设定所要求的输出增益,其增益误差指标定义在单位输出增益条件下 (放大器的反相端与输出端相连接),它包括了由外部电阻偏差所产生的误差。但加载-感应型 DAC 需要昂贵的外部电阻、电阻阵列或数字电位器才能达到与固定增益 DAC 相同的增益误差及温度漂移指标。

在实际应用中,加载-感应型 DAC 只需要增加少许几个元件,就可以构造许多有用电路,有很大的应用灵活性。下面将介绍几个应用电路。

二、加载-感应 DAC 应用电路

1. 增益可调节的 DAC

图 10.5-2 是采用加载-感应 DAC MAX5123 构造的一个自定义、固定增益实例。如图 10.5-2 所示,利用内部 1.25 V 参考电压源和外部电阻,使输出增益被设定为 +2.2 V/V,DAC 的输出范围大约为 0~2.75 V。

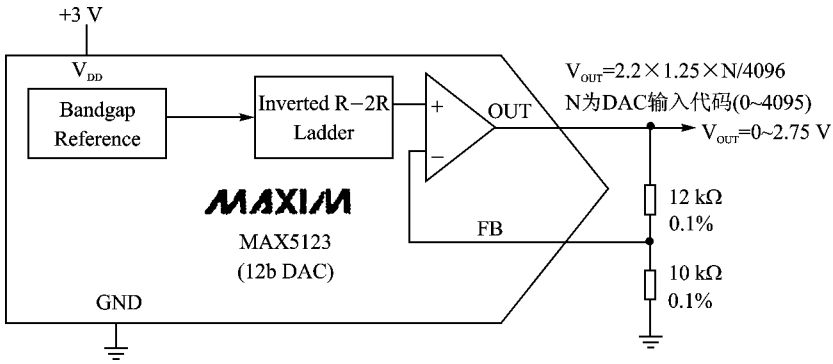


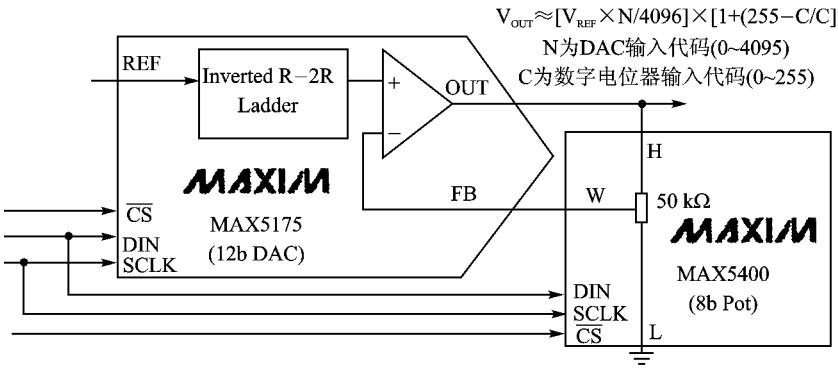
图 10.5-2 具有可选择固定 +2.2V/V 增益功能的 DAC

2. 增益可数字编程的 DAC

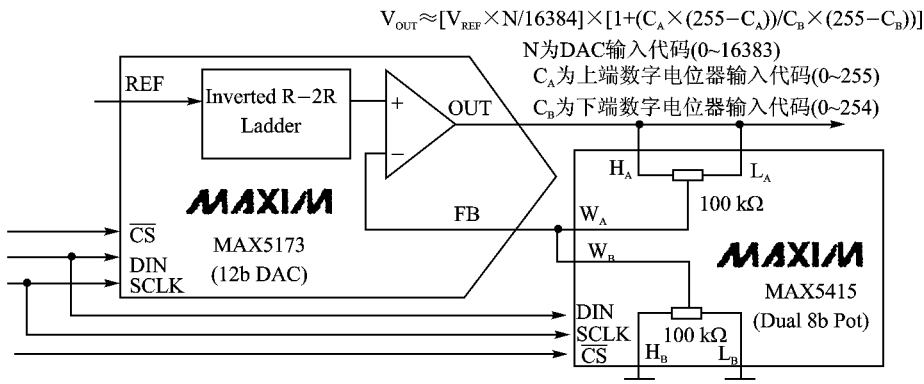
利用加载-感应 DAC 和数字电位器还可以构造增益可数字编程的 DAC。图 10.5-3 是采用 DAC MAX5175/MAX5177 和数字电位器 MAX5400/MAX5415 构成的程控增益 DAC 的两个示例。DAC 和数字电位器均采用 SPI 总线,因此只用 4 根 I/O 口线 (一根时钟线,一根数据输入口线和两根片选线) 就能实现"只写"功能操作。

图 10.5-3 (a) 电路中,MAX5175 的增益由 MAX5400 设置,上电后增益的初始值为 1.992V/V。在此附近,增益调整精度为 0.8%。尽管其上限增益受基准源和供电电压影响,非线性增益调节范围近似为 +1V/V ~ +255V/V,增益设定精度在较高增益时变差。该电路一个很大的优点是:由于单个数字电位器内部电阻匹配较好,增益的温漂系数的典型值可达5 ×10^{-6}/ 。

第 2 个电路 (见图 10.5-3 (b)) 则采用一个双 8 位数字电位器——MAX5415,设置一个 2 位 DAC-MAX5173 的增益。上电后,增益的默认值为 +2V/V,该值附近增益的调整精度为 0.02%。由于采用 2 个 8 位电位器,因此该电路在整个非线性增益 +1V/V ~ +65V/V 范围内,具有很高的精度。虽然温度漂移特性不如采用单个电位器好,但由于它们在同一硅片上,温漂同样接近 5 ×10^{-6}/ 。所以,图 10.5-3 所示电路把加载-感应 DAC 的增益设定灵活性和固定增益 DAC 的高精度和低温漂特性进行了有机结合。



(a) 较大增益变换范围，低增益设定精度



(b) 较窄增益范围，较高的增益设定精度

图 10.5-3 利用数字电位器，DAC 的增益可数字程控

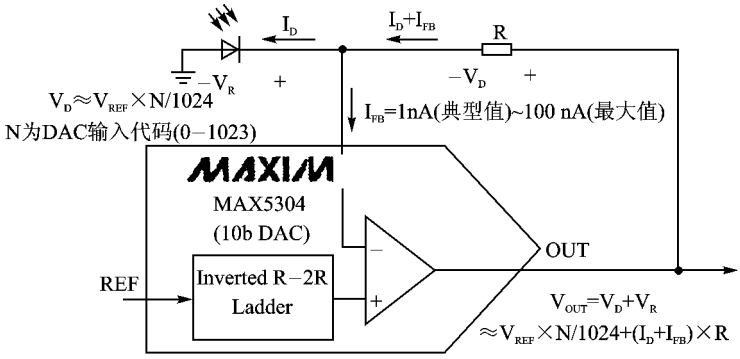
3. **光电二极管和互阻放大器的偏置电压控制电路**

图 10.5-4 两个电路说明如何利用加载-感应 DAC 的输出运放构造成一个互阻放大器。两个例子中，光电二极管的输出电流都被互阻放大器转换为电压信号。图 10.5-4 (a) 中，DAC-MAX5304 用来消除失调电压，确保接地光电二极管的反向偏置电压很小。在图 10.5-4 (b) 中，双 DAC-MAX5156/MAX5157 能够为光电二极管提供任何偏置电压；反向偏置甚至零电压偏置。需要提醒的是，目前 MAXIM 提供的加载-感应 DAC 的 FB 端最大输入偏置电流达 100 nA，该电流值可能对某些光电二极管来说过大，因此需要外加低偏置电流运放，比如 MAX4162，其 I_{bias} 的典型值仅为 1 pA。还需要注意的是，构造互阻放大器时要确定运放是否能够稳定工作，特别是当运放的反向输入端有容性负载时。

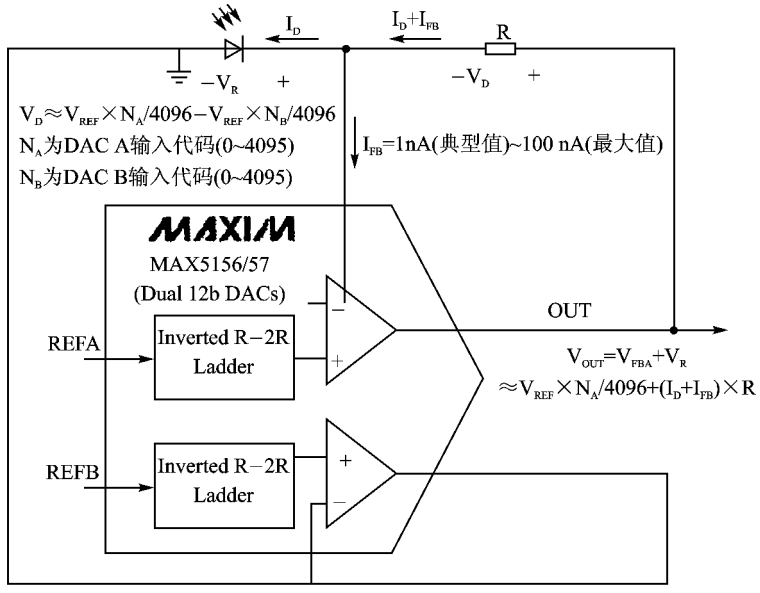
4. **互阻放大器**

图 10.5-5 电路是加载-感应 DAC 用于互阻放大器的另一个实例，图中 DAC 为电流表传感器提供一个直流偏置电压，而传感器的输出电流被互阻放大器转换为电压信号。电流表（或更通常的电压表）传感器通常被用于医疗设备中，而加载-感应 DAC 非常适合这种应用场合。

除以上列举的典型电路外，利用加载-感应 DAC 还可以实现数控电流源、温度检测等，其电路结构简单、使用灵活。



(a) 利用单个DAC对接地光电二极管偏置



(b) 利用双DAC进行零偏置或反向偏置

图 10.5－4 控制光电二极管和互阻放大器的偏置电压

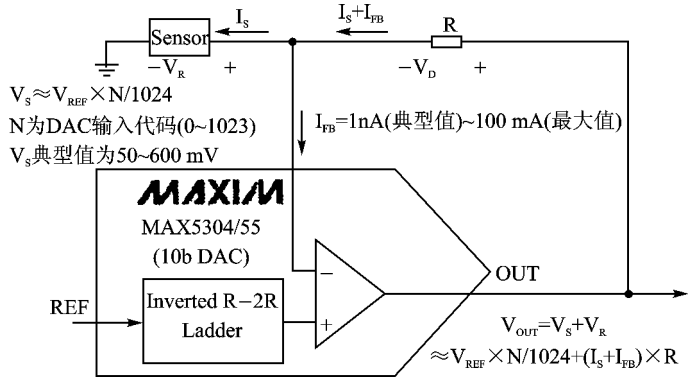


图 10.5－5 为电流表传感器产生可控偏置电压

选自《电测与仪表》月刊,2002 年第 2 期

10.6 利用 MAX7219 设计 LED 大屏幕基本显示模块

焦作大学 (454003) 李长有 魏丕勇

一、8×8 点阵单元结构及显示过程

制作 LED 大屏幕最基本的元器件是 LED 发光二极管,采用单个的发光二极管,1 平方米的大屏幕就需要 1 万多只,组装起来特麻烦。目前市场上有封装好的 8×8 点阵单元,其结构简图如图 10.6-1 所示。这是一个 8×8 单色的点阵单元,有 16 只管脚,其中字母脚 DP、A、B、C、D、E、F、G 分别接 8 位段数据,数字脚 1、2、3、4、5、6、7、8 分别接位数据,这是一种共阴接法。按照共阴接法,从结构上可知,8 行的段数据复用,某时刻只能点亮 1 行,8 行轮流点亮。

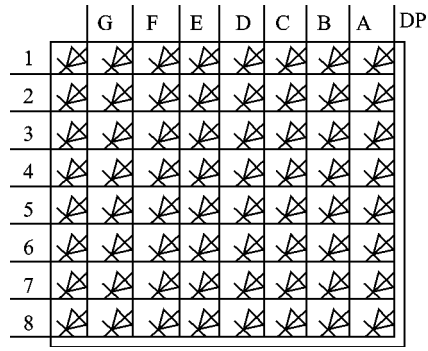


图 10.6-1 8×8 点阵单元结构

显示一个字符的过程 先在 8 个段数据脚上送第一行点阵数据 (即段数据),在 8 个位数据脚上送位数据,点亮第一行,延时一定时间,熄灭,再送第二行点阵数据,点亮第二行,延时一定时间,熄灭……,如此 8 行轮流点亮,显示一个完整字符。由此可知,要显示一个字符,就要不断更换点阵数据,同时更换点亮的行,这叫刷新。仅刷新过程由 CPU 完成,要花费大量时间。从硬件看, 不论段数据还是位数据,都需要锁存器和驱动器,至少 4 个芯片,以及不少电阻。为节约并行口,通常采用串行方式传递数据,还要再加上 2 片串转并芯片如 74LS164,这样至少要 6 个芯片,结构复杂。下面采用一种集串转并、锁存器、驱动器、自动刷新于一体的芯片,将 6 个芯片及许多电阻完成的功能由 1 个芯片完成,使硬件结构大为简化,且由于自动刷新,还省去不少 CPU 时间。

二、MAX7219 的总体性能及典型应用简介

MAX7219 是一种高集成化的串行输入/输出共阴极显示驱动器,可实现微处理器与 8 位 7 段数字 LED,或 64 位单一 LED 的接口。芯片上包括 BCD 码译码器、多位扫描电路、段驱动器、位驱动器、内含 8×8 位静态 RAM,用于存放 8 个数字的显示数据。只需外接一个电阻就可为所有 LED 提供段电流。MAX7219 的三线串行接口可方便连接所有通用微处理器,各个

数据可被单一寻址和更新，无需重写整个显示器。MAX7219 具有软件译码和硬件译码两种功能，软件译码是根据各段笔划与数据位的对应关系进行编码（称为不译码方式），硬件译码采用 BCD 码（简称 B 码）译码。MAX7219 工作模式包括 150 μA 低压电源关闭模式、模拟/数字亮度控制、允许用户从 1 位到 8 位数显示的扫描界限寄存器及测试模式（点亮所有 LED）。其结构简图见图 10.6－2。SEG A～G、DP 是段数据输出脚，可供出电流，典型值为 37 mA，极限值为 100 mA，DIG0～7 吸入电流，典型值为 330 mA，极限值为 550 mA。MAX7219 的典型应用是驱动 8 位数码 LED 显示器，如图 10.6－3 所示。

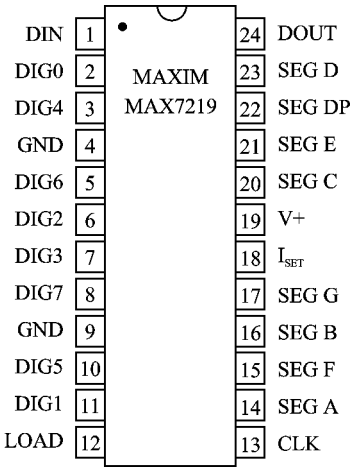


图 10.6－2　MAX7219 的引脚排列

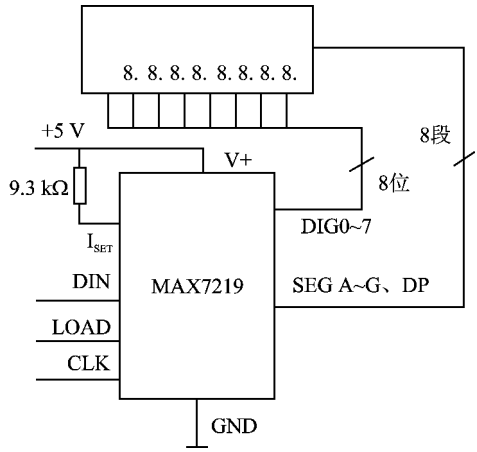


图 10.6－3　MAX7219 的典型应用

采用不译码方式，用一片 MAX7219 驱动一个 8×8 点阵单元也非常合适。MAX7219 驱动 8×8 点阵单元的连接图如图 10.6－4 所示。

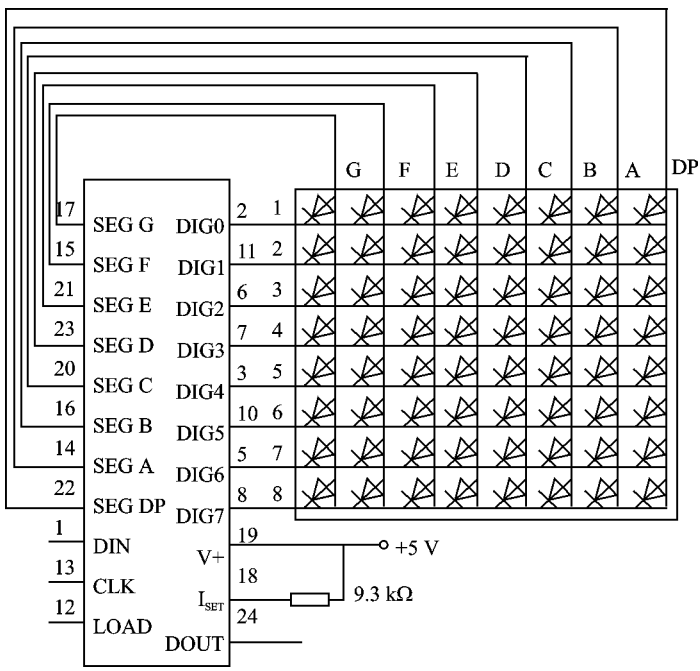


图 10.6－4　MAX7219 驱动 8×8 点阵单元的连接图

三、针对 8×8 点阵单元驱动的 MAX7219 具体设置

关于 MAX7219 的详细内容已有许多文章介绍,不再详述。这里仅叙述针对 8×8 点阵单元驱动 MAX7219 的具体设置。

1. 串行数据格式

MAX7219 要求微处理器以 16 位数据包方式串行发送数据到 DIN 端,串行数据在每个 CLK 的上升沿被移入内部 16 位移位寄存器。然后在 LOAD 的上升沿数据被锁存到数据或控制寄存器中。LOAD 必须在第 16 个时钟上升沿同时或之后,但在下一个上升沿之前变高,否则数据将会丢失。DIN 端的数据通过移位寄存器传送,并在 16.5 个时钟周期后出现在 DOUT 端,数据在 CLK 的下降沿输出。数据位标记为 D0 ~ D15,其数据格式如表 10.6-1 所列。

表 10.6-1 串行数据格式

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
X	X	X	X	地址				最高位			数据			最低位	

X = "无关"位

表 10.6-1 中 D8 ~ D11 为寄存器地址,D0 ~ D7 为 8 位寄存器数据,而 D12 ~ D15 为"无关"位。接收到的第一位为 D15,是最高位(MSB)。而微处理器串行口输出数据是 D0 在前,正好与此相反,因而,微处理器在发送数据之前,要先将数据的高低位颠倒过来之后,才能发送。

不译码方式数字位和对应段线如表 10.6-2 所列。从图 10.6-4 与表 10.6-2 的对照可知,MAX7219 的段数据输出线在接到 8×8 点阵单元时,已从硬件上将高低位作了颠倒,对 8×8 点阵单元 DP、A ~ G 是按顺序从字节的低位标起,而 MAX7219 从字节的高位标起。这样就避免了点阵数据输出前软件的高低位颠倒,从而节省了 CPU 时间。

表 10.6-2 不译码方式数字位和对应段线

寄存器数据	D7	D6	D5	D4	D3	D2	D1	D0
相应的段线	DP	A	B	C	D	E	F	G

2. 译码方式寄存器

译码方式寄存器对每个数字设置 BCD 代码 B 译码或不译码操作。寄存器中的每一位与一个数字相对应。逻辑高电平选择代码 B 译码而逻辑低电平旁路译码器。当驱动 8×8 点阵单元时,选择不译码,译码寄存器中的数据应设置为 00(HEX 代码)。当选择不译码时,数据位 D7 ~ D0 直接对应于 MAX7219 的段线。

3. 扫描界限寄存器

扫描界限寄存器设置有多少个字要显示,可从 1 到 8。它们一般以扫描率 1 300 Hz、8 位数字、多路复用方式显示。由于这里每个 MAX7219 是驱动 8×8 点阵单元,要设置成显示 8 位,所以扫描界限寄存器 XB 的数据设置为 X7。

4. 亮度控制和数字间空白

MAX7219 使显示亮度可由 V+ 和 I_{SET} 之间所接外部电阻(R_{SET})控制,并用计数法使用亮

度寄存器。来自段驱动器的峰值电流通常为进入 I_{SET} 电流的 100 倍。这个电阻既可为固定的,又可为可变的,以便由面板来进行亮度调节。其最小值为 9.53 kΩ,此值典型地设置段电流为 37 mA。

段电流数字控制由内部的脉宽调制 DAC 来控制,该 DAC 由亮度寄存器的低四位加载。该 DAC 将平均电流分成 16 级,从 R_{SET} 设置的峰值电流的最大值 31/32 到最小值 1/32。亮度寄存器格式如表 10.6－3 所列。

表 10.6－3　亮度寄存器格式

占空比	寄存器数据								HEX 代码
	D7	D6	D5	D4	D3	D2	D1	D0	
1/32 (最小亮度)	X	X	X	X	0	0	0	0	X0
3/32	X	X	X	X	0	0	0	1	X1
5/32	X	X	X	X	0	0	1	0	X2
7/32	X	X	X	X	0	0	1	1	X3
9/32	X	X	X	X	0	1	0	0	X4
11/32	X	X	X	X	0	1	0	1	X5
13/32	X	X	X	X	0	1	1	0	X6
15/32	X	X	X	X	0	1	1	1	X7
17/32	X	X	X	X	1	0	0	0	X8
19/32	X	X	X	X	1	0	0	1	X9
21/32	X	X	X	X	1	0	1	0	XA
23/32	X	X	X	X	1	0	1	1	XB
25/32	X	X	X	X	1	1	0	0	XC
27/32	X	X	X	X	1	1	0	1	XD
29/32	X	X	X	X	1	1	1	0	XE
31/32 (最大亮度)	X	X	X	X	1	1	1	1	XF

本设计中,采用可变电阻,允许手工调整亮度,并且 R_{SET} 尽可能设置为较小值,以便在最大范围内通过程序采用数字控制的方式调整显示亮度。当系统安装采光传感器时,就可以根据采光传感器测得的环境光亮度,程序自动调整亮度寄存器的数据值,从而获得最佳的显示效果。

5. 非工作寄存器

当 MAX7219 级联时,使用非工作寄存器。把所有器件的 LOAD 输入连接在一起,而把 DOUT 连接到相邻 MAX7219 的 DIN 上。级联时传送数据的方式是,例如,如果 4 片 MAX7219 级联,那么要对第 4 片芯片写入时,发送所需的 16 位字,其后跟有 3 个“非工作”代码 (十六进制数 X0XX)。当 LOAD 变高时,数据被锁存在所用器件中。前 3 个芯片接收非工作指令,而第 4 个芯片接收数据。

作为大屏幕要用成百上千片 MAX7219,由上所述可知,如果采用级联的方式,要传送一个

有用数据就要传送许多无用的“非工作”代码,因而作为大屏幕设计不能采用级联的方式。要提高数据传送速度,每片 MAX7219 都必须直接传送数据而与其他片无关。因而采用的接法是,将所有 MAX7219 的 DIN 脚与 CLK 脚都直接接在串行总线上,而将 LOAD 脚作为各片的片选分别接各自的片选信号,DOUT 脚空着不用。

四、LED 大屏幕基本显示模块的设计

1. 基本显示模块的构成

作为大屏幕必须能显示汉字。通常一个汉字的点阵是 8×8 点阵单元的倍数,比如 16×16、24×24、32×32 等,因而一个汉字至少需要 4 块 8×8 点阵单元。为了组装方便,以 4 个汉字为一个显示单位作成一个线路板。4 个汉字需要 16 块 8×8 点阵单元,对应只需要 16 块 MAX7219 驱动。根据上面的分析,多片 MAX7219 不能采用级联的方法,须采用片选的方法,因而也就需要 16 条片选线,采用 1 片 74LS154 译码,正好译出 16 条线作为 16 片 MAX7219 的片选信号。至此,一个基本的显示模块需 16 块 8×8 点阵单元、16 块 MAX7219 驱动器和 1 片 74LS154 译码器,组成一个 4 汉字的基本显示模块。当需要更大的显示面积时,只需要增加基本显示模块即可。

2. 基本模块的级联

由于所用的 MAX7219 都只并接在串行总线上,所以在设计基本显示模块线路板时,对串行总线设计一个入口和一个出口,级联时,两块基本显示模块用两根电线连接起来即可。

再来考虑 74LS154 的级联。由于每块基本显示模块上都有一块 74LS154,多块基本显示模块上的 74LS154 必须能区别开来,可采用两种方法。第一种是,每片 74LS154 各自有自己的 4 根输入线,这样也需要许多口线;第二种是,每片 74LS154 公用 4 根输入线,即所有的 74LS154 都并接在 4 根口线上,通过控制各块 74LS154 的使能端使具体的某一块译码。本设计选择第二种方法,这种方法特别利于级联。为了 74LS154 的级联,在基本显示模块线路板上,针对 4 根译码数据输入线,也设计一个入口和一个出口,级联时,两块基本显示模块用 4 根导线首尾连起来即可。当然,也可以将 6 根级联线合成一个线排连接。

当级联块数多时,必须考虑总线的驱动。因此,在每块基本显示模块线路板的总线输入口,再增加一片总线驱动器 74LS244。至此一个基本的显示模块就完全做好了,其对外的连线只有 2 根串行总线、4 根译码数据输入线及 1 根译码使能控制线,再就是电源线了。而 6 根总线是相互级联的,而只有一根译码使能控制线是需要单独连接的,原理图如图 10.6-5 所示。因而这种设计是模块化设计,根据用户的需要扩展起来非常方便,而且,每个基本显示模块是自刷新的,微处理器一旦送完显示数据,就不需要再管刷新的事了。因而,一个大屏幕只需要 1~2 片微处理器,而其他设计,需要多片微处理器。

利用 MAX7219 构造的 LED 大屏幕基本显示模块,硬件结构简单,可硬件或软件改变显示亮度,具有自刷新功能,省却许多 CPU 的刷新时间,使系统设计也简单。当用此基本显示模块构造大屏幕时,各基本模块级联方便,基本显示模块与系统连线少而容易。

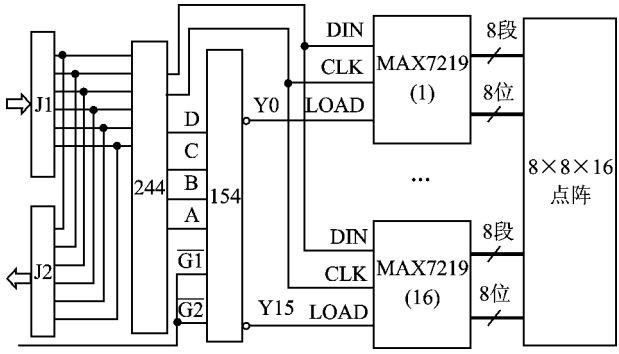


图 10.6－5　**基本显示模块构成原理**

选自《电子技术》月刊，2002 **年第** 9 **期**

10.7 单片机用作通用红外遥控接收器的设计

清华-华录信息技术研究所 朱纯益 路建华

红外线遥控是目前使用最广泛的一种通信和遥控手段。由于红外线遥控装置具有体积小、功耗低、功能强、成本低等特点，因而，继彩电、录像机之后，在录音机、音响设备、空调机以及玩具等其他小型电器装置上也纷纷采用红外线遥控。工业设备中，在高压、辐射、有毒气体、粉尘等环境下，采用红外线遥控不仅安全可靠而且能有效地隔离电气干扰。

一、红外遥控系统

通用红外遥控系统由发射和接收两大部分组成，应用编/解码专用集成电路芯片来进行控制操作，如图 10.7-1 所示。发射部分包括键盘矩阵、编码调制、LED 红外发送器；接收部分包括光、电转换放大器、解调、解码电路。

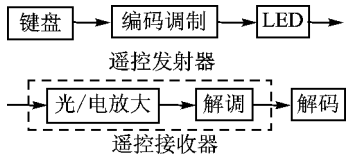


图 10.7-1 红外遥控系统框图

二、遥控发射器及其编码

遥控发射器专用芯片很多，现以日本 NEC 的 μPD6121G 组成发射电路为例说明编码原理。当发射器按键按下后，即有遥控码发出，所按的键不同遥控编码也不同。这种遥控码具有以下特征：

采用脉宽调制的串行码，以脉宽为 0.565 ms、间隔 0.56 ms、周期为 1.125 ms 的组合表示二进制的“0”；以脉宽为 0.565 ms、间隔 1.685 ms、周期为 2.25 ms 的组合表示二进制的“1”。其波形如图 10.7-2 所示。

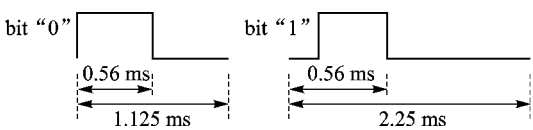


图 10.7-2 遥控码的“0”和“1”

上述“0”和“1”组成的 32 位二进制码经 38 kHz 的载频进行调制，提高发射效率，达到降低电源功耗的目的。然后，再通过红外发射二极管进行二次调制，产生红外线向空间发射，如图 10.7-3 所示。μPD6121G 产生的遥控编码是连续的 32 位二进制码组，其中前 16 位为 8 位用户识别码及其反码，能区别不同的电器设备，防止不同机种遥控码互相干扰。该芯片的用户识别码固定为十六进制 01H；后 16 位为 8 位操作码（功能码）及其反码。μPD6121G 最多有 128 种不同组合的编码。

引导码		用户识别码								用户识别码（反码）								操作码								操作码（反码）							
9 ms	4.5ms	C0	C1	C2	C3	C4	C5	C6	C7	C0	C1	C2	C3	C4	C5	C6	C7	D0	D1	D2	D3	D4	D5	D6	D7	D0	D1	D2	D3	D4	D5	D6	D7

图 10.7－3　遥控信号编码波形图

遥控器在按键按下后，周期性地发出同一种 32 位二进制码，周期约为 108 ms。一组码本身的持续时间随它所包含的二进制“0”和“1”的个数不同而不同，大约在 45～63 ms 之间，图 10.7－4 为发射波形图。

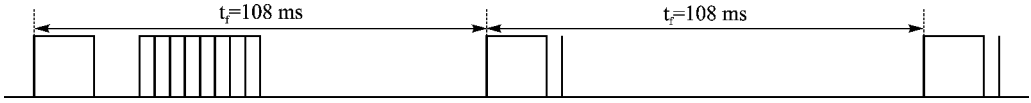


图 10.7－4　遥控信号的周期性波形

三、接收器及解码

TSOP1738 是 VISHAY 公司推出的一体化红外线接收器，集红外线接收和放大于一体，不需要任何外接元件，就能完成从红外线接收到输出与 TTL 电平信号兼容的所有工作，而体积和普通的塑封三极管大小一样，其功能如图 10.7－1 所示的虚线部分。它适合于各种红外线遥控和红外线数据传输。

解码就是识别二进制码“0”和“1”以及遥控信号起始位。由 8051 单片机对脉冲间隔计数，由计数值的大小区别脉冲间隔的时间，从而识别出二进制码“0”、“1”和遥控信号起始位。如前所述，红外遥控的 32 位二进制串行码是脉宽调制的，脉冲宽度固定（0.56 ms），而脉冲的间隔不同。因此，只要设法测出脉冲间隔时间，即可判断是二进制的“0”还是“1”。考虑到适当的容差，可把脉冲间隔为 0.256～0.768 ms 的判为“0”，脉冲间隔为 1.28～1.792 ms 的判为“1”。

1. 解码系统配置及接口

解码单片机系统由 8051、TSOP1738 和 74LS00 等组成，接口电路如图 10.7－5 所示。TSOP1738 的输出端通过 74LS00 的两个反相电路接至解码单片机 8051 的 INT0 和 INT1，作为输入接口。8051 解码单片机通过 P0 口作为输出接口，传送解码所得的指令控制码去控制电器设备。8051 单片机的 TCON 中有一个控制位，该位由软件设置为“1”，设置 INT0 和 INT1 为下降沿触发中断，其相应的定时/计数器 0 就可以测量对应的 INT 引脚上正脉冲的宽度。利用这一特点，用定时器 T0 来测量 INT1 引脚上正脉冲的宽度，即前后两脉冲的间隔时间，据此可判断它对应于二进制的是“1”还是“0”。

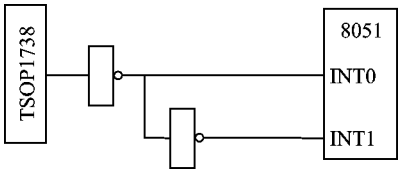


图 10.7－5　接口电路原理图

由图 10.7－5 可知，TSOP1738 送往 8051 解码单片机 INT0 和 INT1 两引脚上的波形相反，由 INT0 引脚上脉冲的下降沿所触发的中断服务程序完成启动计数器 T0，以测量 INT1 引脚上正脉冲的宽度。由 INT1 引脚上脉冲的下降沿所触发的中断服务程序完成关计数器 T0，并根据计数值来判断是对应于二进制的“0”还是“1”。

2. 软件设计

解码单片机 8051 的软件包括主程序、INT0 中断服务程序和 INT1 中断服务程序等 3 部分。在 8051 单片机内部 RAM 区建立的工作单元和标志位。

(1) BUF0 ~ BUF3——接收缓冲移位寄存单元 (32 bit),每次由 INT1 中断服务程序解出的存于 CY 的二进制位,通过累加器连同 CY 的右移操作,传送到 BUF0 ~ BUF3 内。

(2) LENG——码长计数器,用于计数解出的二进制位数。

(3) (29H). 0——码间隔标志位,当收到码组间隔时该标志被置位。

(4) (29H). 1——用户识别码标志位,当收到一组码的前 8 位为 01H 时该标志被置位。

以下是主程序的设计要点。

(1) 正确地解码必须从一组码的起始进行。为此程序在初始化后,首先检测码间隔标志 (29H). 0,如果为 1,表明是一组码的开始,程序就将码长计数器清零,以便从头开始计数。

(2) 为防止其他遥控码的干扰,当接收到前 8 位码后,要检查它的值是否为 01H。如果是,则置位用户码标志 (29H). 1。只有用户码标志为 1 时,收到的后 8 位码才作为有效操作码处理。

(3) 为了得到正确的解码结果,要检查 32 位遥控编码中用户码和操作码各自的反相一致性。正极性的用户码留在 BUF3 中,负极性的用户码留在 BUF2 中;正极性的操作码留在 BUF1 中,负极性的操作码留在 BUF0 中。然后比较 BUF3 和 BUF2 中的内容,比较 BUF1 和 BUF0 中的内容,如果都满足反相一致才进入下一步,否则,作无效码处理。

具体程序见本刊网站补充版。http ∶/ /www. dpj. com. cn

以上所述方法非常简单地实现了红外遥控信号的接收解码,极大地节约了硬件实现的资源开销。只要修改汇编代码的部分参数,就可以适用于多种红外遥控器信号的接收和解码。

参 考 文 献

1 VISHAY. Photo Modules for PCM Remote Control Systems

2 NEC. MOS Integrated Circuit uPD6121G

3 李朝青. PC 机及单片机数据通信技术. 北京 北京航空航天大学出版社,2000

4 李勋,刘源,李新民. 单片机实用教程. 北京 北京航空航天大学出版社,2000

选自《单片机与嵌入式系统应用》月刊,2002 年第 8 期

10.8 红外遥控器软件解码及其应用

皖西学院 李经达

在单片机控制产品的开发应用中,为了向控制系统输入控制命令,键盘往往是不可缺少的。传统的方法是利用并行输入/输出接口芯片扩展一个键盘接口,或者直接利用单片机的并行端口进行扩展。在某些应用环境下,这种方式有2个弊端:① 键盘和控制系统连在一起,不灵活,环境适应性差;② 浪费单片机的端口,且硬件成本较高。

使用红外遥控器作为控制系统的输入设备,具有成本低、灵活方便的特点。本文目的就在于介绍软件解码研究的一般方法和对红外遥控器进行二次开发的应用技术。该方法已在多个应用系统设计中成功地实现,效果良好。

红外遥控器是一种非常容易买到,且价格便宜的产品,种类很多,但它们都是配合某种特定电子产品的(如各种电视机、VCD、空调器等),由专用CPU解码,作为一般的单片机控制系统不能直接使用。使用现成遥控器作为控制系统的输入,需要解决如下几个问题:如何接收红外遥控信号;如何识别红外遥控信号;解码软件的设计。其他的问题都是非本质的,例如遥控器面板功能键标注问题,可自行设计、重印即可。

一、红外遥控信号的接收

接收电路可以使用集成红外接收器成品。接收器包括红外接收管和信号处理IC。接收器对外只有3个引脚:V_{CC}、GND和1个脉冲信号输出PO。与单片机接口非常方便,如图10.8-1所示。

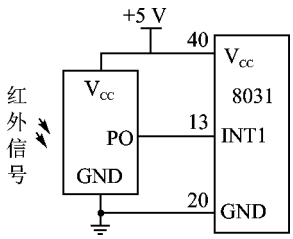


图10.8-1

(1) V_{CC}接系统的电源正极(+5 V);

(2) GND接系统的地线(0 V);

(3) 脉冲信号输出接CPU的中断输入引脚(例如8031的13脚INT1)。采取这种连接方法,软件解码既可工作于查询方式,也可工作于中断方式。

二、脉冲流分析

要了解一个未知的遥控器,首先要分析其脉冲流,从而了解其脉冲波形特征(以何种方式携带"0"、"1"信息),进而了解其编码规律。脉冲流的分析应从分析脉冲的高、低电平宽度入手。笔者用软件的方法实现了对脉冲流的分析。以图10.8-1所示的接口为例,如果没有红外遥控信号到来,接收器的输出端口PO保持高电平;当接收到红外遥控信号时,接收器将信号转换成脉冲序列加到CPU的中断输入引脚。用软件测试引脚的逻辑电平,同时启动TC计时器,测量该引脚分别为逻辑"0"和逻辑"1"情况下的时间值,存储起来,然后打印、分析。下面用8051汇编语言给出对脉冲流进行采样、存储的程序段:

```
MOV     R0,#00H
```

```
        MOV   R1,#28H
        MOV   TMOD,#01H
TK：    JB    P3.3,TK          等待低电平到来
 测低电平宽度
TK1：   MOV   TH0,#00H
        MOV   TL0,#00H
        SETB  TR0
TK2：   JB    TF0,TKE          超时无效返回
        JNB   P3.3,TK2
        CLR   TR0
        MOV   A,TH0
        MOVX  @R0,A
        INC   R0
        MOV   A,TL0
        MOVX  @R0,A
        INC   R0
 测高电平宽度
        MOV   TH0,#00H
        MOV   TL0,#00H
        SETB  TR0
TK3：   JB    TF0,TKE          超时无效返回
        JB    P3.3,TK3
        CLR   TR0
        MOV   A,TH0
        MOVX  @R0,A
        INC   R0
        MOV   A,TL0
        MOVX  @R0,A
        INC   R0
        DJNZ  R1,TK1           循环
TKE：   RET
```

这段程序首先将 TC0 设置成 16 位定时器方式,初始化 RAM 地址指针 R0 和循环计数指针 R1,每当引脚的逻辑电平发生跳变时,停止计时,将计时值保存到连续的 RAM 中。这段程序可以连续测量 40 个脉冲的时间值(包括 40 个低电平脉宽)。笔者以 TC9012 芯片的遥控器为对象,采集了所有按键的编码脉冲波形,并且对同一按键进行了重复实验。限于篇幅,采样数据不能给出,仅给出脉冲流的规律(仿真机 CPU 晶振为 6 MHz)。

(1) 引导脉冲是一个时间值为 0937H ~ 0957H 的低电平和时间值为 084FH ~ 086FH 的高电平;

(2) 数据脉冲的低电平时间值约为 0127H ~ 0177H;

(3) 高电平时间值有 2 种情况:00BBH ~ 00FFH(窄)、02EFH ~ 0333H(宽)。

由大量数据总结分析,按键编码有如下规律:

⑴ 除引导脉冲外的脉冲是数据编码脉冲,数据“位”信息由高电平脉宽决定:窄脉宽表示“0”、宽脉宽表示“1”;

⑵ 每个按键的脉冲流译码后,包含 4 个字节的信息:

· 所有按键的前 2 个字节编码都一样,都是 2 个字节的“0EH”;

· 第 3 字节是键码;

· 第 4 字节是键码的反码。

经过对相同按键脉冲流进行多次采样发现,相同按键脉冲序列的对应位置脉宽时间值是在一个小范围内波动的(不是一个确定值),因此,对模式的识别不能采取精确比较法。对此,本人采取模糊的办法进行了抽象处理。根据上述实验规律,将软件译码时对脉冲的分析判断依据及算法设计思想总结如下。

⑴ 引导脉冲的低电平和高电平宽度的判断依据是时间值的“高字节大于 08H”,低字节忽略。

⑵ 数据脉冲流的低电平脉宽相同,忽略不判断。

⑶ 高电平脉宽是判断数据流每位是“0”还是“1”的依据。本人抽取的判据是脉宽的高字节若小于 2 表示“0”,否则表示“1”,脉宽的低字节忽略。

实践证明,上述判据是有效可行的。这样处理不仅使解码软件的设计简单化,而且大大提高了解码的速度。使用上述判据编写软件解码程序时,要注意脉冲流采样数据存储地址与脉冲的对应关系。软件主要有如下几部分:

⑴ 判断遥控信号的到来(在解码前调用 1 个独立的子程序);

⑵ 采样并存储脉冲流;

⑶ 判断引导脉冲是否有效;

⑷ 解码前 2 个字节并判断是否为“0EH”;

⑸ 解码第 3 个字节,该字节即为有效键码;

⑹ 键码的查表映射(如果使用原键码,可省略这一步)。

三、解码软件的设计

基于上述思路设计的软件解码系统成功地应用于多个控制系统。下面给出一个实例(用 MCS-51 系列 MCU 对 TC9012 红外遥控器进行软件解码)的汇编语言程序。程序中使用的参数是针对 MCU 使用 6 MHz 晶振的情况,使用其他频率的晶振,只需修改脉宽判据即可。为便于理解,尽量保持与原理叙述中的一致性,程序中给出了较详细的注释,详见本刊网络补充版(http://www.dpj.com.cn)。

本文虽然是用 MCS-51 系列 MCU 对 TC9012 红外遥控器软件解码的研究,但其方法具有一般性。具体的应用,可自行变通。

参 考 文 献

1 孙景琪. 遥控彩色电视机集成电路及应用. 北京:人民邮电出版社,1996

2 何立民. 单片机应用系统设计. 北京:北京航空航天大学出版社,1990

选自《单片机与嵌入式系统应用》月刊,2002 年第 12 期

第十一章

文章摘要

一、专题论述

1.1 与 8051 兼容的单片机的新发展

摘自《电子产品世界 A》月刊,2002 年第 7 期

近年来,出现了以 AdμC8xx 系列单片机为代表的一批新型单片机,这些单片机是与 8051 兼容,但功能都远远胜过标准的 8051 单片机,这些单片机的出现,为提升测控系统和仪器仪表、家用系列提供了强有力的手段,这些新型的单片机代表了单片机的发展方向。本文介绍了这些新型单片机的性能和特点,可为研发人员选型提供参考。

1.2 正在崛起的低功耗微处理器技术

摘自《工业控制计算机》月刊,2002 年第 11 期

本文详细讨论了 x86 及其兼容架构微处理器的低功耗设计技术,探讨了该技术和产品的现状和发展趋势,预见了低功耗微处理器的广泛应用前景。

1.3 低功耗电子系统设计的综合考虑

摘自《仪表技术》月刊,2002 年第 4 期

本文探讨了低功率电路和系统的发展趋势,分析了功耗产生的主要原因,提出了几种实现低功率的方案。

1.4 数字电路设计方案的比较与选择

摘自《电子技术应用》月刊,2002 年第 1 期

DSP 和 FPGA 是目前数字电路设计采用的两种主要手段,其各有自身的优缺点,故在设计数字系统前必需进行方案选择。DSP + FPGA 结构的新思想的出现以及嵌入 DSP 模块的 FPGA 设计方案使得数字电路设计有了更大的选择空间。本文分析了 DSP 和 FPGA 结构特点,并讨论了 DSP 与 FPGA 的方案选择。

1.5 单片机应用系统中数学协处理器的开发

摘自《电子技术》月刊,2002 年第 8 期

为避免单片机应用系统开发中因为数学计算所引起的工作效率低下,本文介绍了一种自制单片机数学协处理器的方法。该方法以计算器集成电路为中心,辅以接口电路和单片机,组成具有通信功能的数学处理模块。通过对计算器输出信号的测量与分析,说明了将信号转换为二进制码的方法。

1.6 实现基于 IP 核技术的 SoC 设计

摘自《电子产品世界 A》月刊,2002 年第 9 期

知识产权 (IP) 核是系统芯片 (SoC) 的基础技术。本文介绍了 IP 核的硬化、IP 核的速模,典型 SoC 设计中的功能模型、时序模型、测试模型、物理模型与功率模型。

1.7 基于知识产权的 SoC 关键技术与设计

摘自《现代电子技术》月刊,2002 年第 3 期

本文介绍了基于 IP 的 SoC 设计特点、关键技术以及 IP 设计分类和 IP 标准。设计特点包括:设计抽象层次的提高、IP 核的重复使用、先进的体系结构评估技术;关键技术包括:设计描述技术、硬件协同设计、IP 集成复用技术及设计环境、深亚微米技术等。

1.8 基于 IP 核复用技术的 SoC 设计

摘自《半导体技术》月刊,2002 年第 4 期

本文概述了国内外 IP 产业的发展情况,论述了我国发展 IP 核复用技术 SoC 设计的可能性和必要性,指出我国急需发展的关键芯片及 IP 核种类。

1.9 将 IP 集成进 SoC

摘自《电子产品世界 B》月刊,2002 年第 9 期

本文介绍了典型 SoC 设计流程以及流程中的一些关键技术,如前端的设计与物理规划,后端的物理设计和验证工作。文中对 IP 整合到 SoC 中的过程和技术内容都作了详尽的叙述。

1.10 模拟/混合电路 SoC 的设计难题

摘自《电子产品世界 B》月刊,2002 年第 4 期

本文讨论了模拟/混合电路 SoC 设计中的一些关键设计与实现问题及一些可行的折衷方案。具体地介绍了模拟/混合 SoC 的电路设计、布图、SoC 验证、测试设计、混合信号 SoC。

1.11 系统级可编程芯片 (SOPC) 设计思想与开发策略

摘自《现代电子技术》月刊,2002 年第 11 期

针对 SOPC 全新的设计流程,提出了基于 IP 的 SOPC 设计集成平台概念、SOPC 发展关键要素及设计思想与开发策略,并介绍了基于 FPGA/CPLD 的 SOPC 实现方案。

1.12 基于 SoC 的 PAGER 控制芯片设计

摘自《微电子学与计算机》月刊,2002 年第 6 期

本文讨论了 PAGER 控制器芯片 (ZQD021) 的系统设计,该控制器内部集成了 FLASH,SRAM、POCSAG 协议解码器和嵌入式 MCU CORE。重点分析了芯片的可测性设计 (DFT)、内嵌 FLASH 设计、低功耗设计。其设计方法和思路对消费类和嵌入式控制芯片的设计有一定的借鉴意义。

1.13 一种高性能 CMOS 带隙电路的设计

摘自《微电子学与计算机》月刊,2002 年第 12 期

本文提出一种 CMOS 带隙参考源 (BGR) 电路设计,它可以在很宽的电压范围内有效的工作。能够在 [2.8V,10V] 范围内实现稳定工作。抗干扰能力强,结构相对简单,由 CMOS 运放、二极管以及电阻组成,用常规的 0.6 umCMOS 工艺制作。在模拟环境下仿真结构表明其最小

工作电压为 2.75 V,完全能够满足锁相环设计的要求。

1.14 基于结构的指纹分类技术

摘自《电子技术应用》月刊,2002 年第 2 期

指纹分类技术是指纹数据库的一个重要的索引机制。本文提出了一种基于指纹方向图的结构分类算法。通过图像分割,抽取图像的有用部分,而后基于指纹方向图,寻找指纹奇异点,利用脊线跟踪技术和规则确定指纹的类别。

1.15 指纹识别的预处理组合算法

摘自《计算机应用》月刊,2002 年第 10 期

本文提出了基于方向分割的一套完整的指纹预处理组合算法,主要包括方向图的计算、平滑、方向滤波、局域自适应、二值化、去噪和细化等算法,能很好地增强指纹图像,减少因指纹旋转及平移因素而造成的误识别,为可靠、准确地实现指纹自识别提供一种可行的方法。

1.16 一种指纹识别的细节特征匹配的方法

摘自《测控技术》月刊,2002 年第 5 期

本文介绍了指纹识别的处理步骤,提出了一种基于细节特征匹配的自动指纹识别算法,分别给出了图像预处理、二值化、细化、纹路提取、确定指纹中心点以及指纹匹配的特征和算法步骤,并用实验进行了证明。

1.17 指纹 IC 卡及其应用

摘自《遥测遥控》双月刊,2002 年第 11 期

本文介绍指纹识别技术和 IC 卡技术,结合实践经验提出指纹 IC 卡的解决方案,最后讨论指纹 IC 卡作为身份认证的应用。

1.18 人脸照片的特征提取与查询

摘自《计算机应用研究》月刊,2002 年第 7 期

根据人脸图像的灰度特性及几何特征,本文作者提出了一种简单、实用、有效的人脸特征提取方法,进而在大样本图像库中进行较准确的识别和查询。该方法可广泛应用于身份识别和公安档案管理等许多领域。

1.19 一种快速、鲁棒的人脸检测方法

摘自《计算机应用研究》半月刊,2002 年第 9 期

本文根据特征脸思想,提出了一种新的基于特征肤色的人脸检测方法。首先从 RGB 彩色空间转换到 HSI 彩色空间中,然后训练肤色点样式本集得到特征肤色,最后在特征肤色构成的肤色空间中检测人脸。实验表明,该方法是快速的、鲁棒的。

1.20 128 条码的编码分析和识别算法

摘自《计算机工程与科学》双月刊,2002 年第 1 期

本文对 128 条码的编码进行了理论分析，证明了由相似边距离确定其编码的惟一性，并据此给出了一种适于批量表格录入的识别条码的有效算法。我们还提议了一种利用条码进行汉字编码的方案。

1.21 身份证号码快速识别系统

摘自《实用测试技术》双月刊，2002 年第 1 期

本文阐述了身份证号码识别系统，整个识别步骤中的关键算法。系统可以自动解析包含二值化身份证号码的 BMP 文件，在经过平滑、粗分割、细分割等一系列处理之后，可以最大限度地减少图像受到的干扰，从而快速准确地识别出每个数字。基于所描述的算法而实现的程序，有着很好的识别性能，应用前景十分广泛。

1.22 汉字识别技术的新方法及发展趋势

摘自《实用测试技术》双月刊，2002 年第 2 期

本文对目前我国在汉字识别领域研究和发展较为活跃的几种汉字识别新技术作一简单介绍，并根据其研究现状对未来汉字识别的发展提出了几点展望。

1.23 蓝牙技术及其应用展望

摘自《现代通信》月刊，2002 年第 9 期

本文介绍了蓝牙技术特点、蓝牙技术的网络结构以及蓝牙应用中的点对点、点对多点的连接、应用于个人局域网等技术。

1.24 蓝牙技术浅析

摘自《现代电子工程》季刊，2002 年第 4 期

本文对一种新兴的无线通信技术——蓝牙技术进行了较为全面的介绍，重点阐述其系统结构、安全体系、应用前景，以及存在问题。

1.25 蓝牙 HCI USB 传输层规范

摘自《计算机工程》月刊，2002 年第 2 期

蓝牙的 SIG 规定了 4 种与硬件连接的物理总线方式：USB，RS232，UART 和 PC 卡。本文详细讨论了蓝牙 USB 接口的硬件设计规范。

1.26 蓝牙服务发现协议（SDP）的实现

摘自《无线电工程》月刊，2002 年第 2 期

本文在介绍蓝牙服务发现协议的服务记录和协议数据单元的基础上，着重描述了蓝牙服务发现协议的实现方法、实现的流程及数据包的封装，并分析了蓝牙 SDP 的发展。

1.27 蓝牙技术安全性解析

摘自《计算机应用》月刊，2002 年第 10 期

本文简要介绍蓝牙技术（IEEE 802.15）的基本概念和原理，在此基础上重点分析蓝牙技

术的两种安全模式,从抗网络攻击角度出发,提出完善蓝牙安全的一些策略和方法。

1.28 蓝牙技术及其应用

摘自《电测与仪表》月刊,2002 年第 2 期

蓝牙技术是一种近距离、低功耗的无线通信标准,它为仪器设备提供了一种低成本的无线联网方式。本文介绍蓝牙技术的基本原理和技术特点,讨论它现在或将来在测控领域的应用,还介绍了其目前所面临的问题。

1.29 Bluetooth ASIC 接口技术

摘自《电测与仪表》月刊,2002 年第 7 期

本文介绍蓝牙 ASIC 芯片开发情况以及由两芯片组(或单芯片组)构成蓝牙模块的方法,并通过蓝牙模块与主机设备之间的 HCI 接口设计,来形成蓝牙设备。

1.30 RF CMOS 蓝牙收发器的设计(一)

摘自《无线电工程》月刊,2002 年第 5 期

该文介绍了一种设计方法,它是通过使用晶片厂模型、论证测试模板以及与测试仪器的链接,实现 RFIC 设计并很快获得原型芯片。描述了蓝牙收发器的设计,其中包括,在统一 EDA 环境下的仿真结果和硬件验证。

1.31 RF CMOS 蓝牙收发器的设计(二)

摘自《无线电工程》月刊,2002 年第 6 期

该文介绍了一种设计方法,它是通过使用晶片厂模型、论证测试模拟以及与测试仪器的链接,实现 RFIC 设计并很快获得原型芯片。描述了蓝牙收发器的设计,其中包括,在统一 EDA 环境下的仿真结果和硬件验证。

1.32 单片蓝牙控制器 AT76C551

摘自《半导体技术》月刊,2002 年第 4 期

本文介绍了 Atmel 公司生产的单芯片蓝牙控制器 AT76C551 的功能结构,并给出了由该芯片构成的蓝牙通道简图。

1.33 设计 RF CMOS 蓝牙收发器

摘自《电子产品世界 B》月刊,2002 年第 4 期

本文探讨了使用 CMOS RFIC 设计 2.4 GHz 蓝牙收发器的诸多技术问题,讨论了两芯片 SoC 解决方案、RF 设计方案、实现方法、设计验证及设计结果。

1.34 ROK 101 007/1 蓝牙模块的特性与应用

摘自《电子技术》月刊,2002 年第 9 期

蓝牙技术是一项新兴的短距离无线通信技术。而蓝牙芯片则是蓝牙技术的基础和关键。本文简单介绍了 Ericsson 公司推出的蓝牙模块 ROK 101 007/1 的特性、原理及其应用。

1.35 基于 nRF401 的 PC 机无线收发模块的设计

摘自《电子技术应用》月刊,2002 年第 4 期

本文介绍了基于 nRF401 无线收发芯片的 PC 机串口通信模块和 PC 机 ISA 插槽通信模块的设计思路和实现方法。通过该两种模块可以方便地实现 PC 机数字信号的载频传播,使计算机之间的无线数据传输成为可能。

1.36 无线收发芯片 nRF401 在监测系统中的应用

摘自《电子技术》月刊,2002 年第 11 期

本文简要介绍了输电线路污秽监测系统,着重介绍了无线收发芯片 nRF401 的特点、结构、工作模式以及在该监测系统应用中的电路设计和软件实现方法,并讨论了通信中存在杂波的解决办法。

1.37 基于射频收发芯片 nRF401 的计算机接口电路设计

摘自《微电子学与计算机》月刊,2002 年第 5 期

本文介绍了采用单片射频收发芯片 nRF401A 接口芯片构成的计算机无线接口电路,工作频率 433 MHz,发射功率 +10 dBm,接收灵敏度 - 105 dBm,数据速率 20 kbit/s。

1.38 采用 nRF401 实现单片机与 PC 机无线数据通信

摘自《微计算机信息》月刊,2002 年第 9 期

本文介绍了利用无线收发芯片 nRF401 实现 PC 机与多台单片机系统的无线数据通信,并给出单片机控制 nRF401 硬件和通信软件的具体设计。该系统数据传输速度较快,可靠性高,功能易扩展,适于多种应用领域。

1.39 基于射频收发芯片 nRF403 的无线接口电路设计

摘自《电子技术》月刊,2002 年第 4 期

本文介绍了采用单片射频收发芯片 nRF403 和 MAX232A 接口芯片构成的无线接口电路,工作频率 315/433 MHz,发射功率 +10 dBm,接收灵敏度 - 105 dBm,数据速率 20 kbit/s。

1.40 蓝牙局域网无线接入网关的研制

摘自《计算机工程》月刊,2002 年第 8 期

本文首先介绍了蓝牙局域网无线接入的网络协议层次。在此基础上,比较详细地介绍了蓝牙局域网无线接入网关的研制,包括 :依据的应用模型、无线接入网关设计、软硬件开发平台选择等。

1.41 基于蓝牙的无线数据采集系统

摘自《电测与仪表》月刊,2002 年第 9 期

本文给出了一种基于蓝牙技术的无线数据采集系统,该系统可以取代有线数据采集系统复杂的现场连线,并且提高了数据采集的抗干扰性能。

1.42 安立蓝牙无线测试解决方案

摘自《国外电子测量技术》双月刊,2002 年第 2 期

蓝牙相关产品在研发和生产过程中必须对该产品的射频性能进行测试,以保证其无线指标符合规范要求。本文介绍了安立公司的蓝牙无线综合测试仪 MT8850A 和蓝牙无线预认证系统 ME7865A。

1.43 嵌入式系统中的蓝牙电话应用规范的实现

摘自《无线电工程》月刊,2002 年第 1 期

本文从蓝牙电话提出的背景入手,全面清晰地阐述了两个对应的应用规范 CTP 和 ICP 的工作原理和流程,并且根据流行的并发软件设计思想,提出了系统解决方案。

1.44 蓝牙“三合一电话”的解决方案

摘自《无线电工程》月刊,2002 年第 9 期

本文介绍了蓝牙“三合一电话”的概念和功能,详细分析了该产品的软硬件解决方案,同时,给出了它的工作原理和工作时的信号流程。

1.45 用 Bluetooth 技术构建分布式污水处理控制系统

摘自《电子产品世界 B》月刊,2002 年第 5 期

本文详细介绍了采用“Bluetooth (蓝牙) ”技术组建的一新型分布式污水处理—控制—管理系统的设计方法。采用蓝牙技术,能在近距离范围内使数据信息有互用、相互操作的性能,同时有效克服了有线方式铺设、安全等方面问题,该系统集控制、管理、信息于一体,可达到“少人值守”的效果,运行可靠、稳定。

1.46 MPEG 的发展动态及其未来预测

摘自《电子技术应用》月刊,2002 年第 9 期

本文回顾了 MPEG 的发展历史,介绍了 MPEG－1、MPEG－4 和 MPEG－7 标准的特点,并对 MPEG 的未来作了展望。

1.47 软件无线电的关键技术与未来展望

摘自《通信世界》月刊,2002 年第 1 期

软件无线电的概念源于军事通信。本文介绍了软件无线电的基本概念及硬件平台结构,介绍了软件无线电的关键技术如低功耗小型化、宽带多频段无线及射频前端。宽带 A/D 变换、高速并行 DSP 等,并展望了软件无线电的未来。

1.48 软件无线电与虚拟无线电

摘自《空间电子技术》季刊,2002 年第 2 期

本文介绍了软件无线电和虚拟无线电的系统结构、特点和应用,以及实现软件无线电和虚拟无线电的关键技术,并为解决这些技术进行探讨。

1.49 射频无线测控系统及其应用

摘自《制造业自动化》月刊,2002 年第 8 期

本文介绍一种测控系统数据无线传输方法,该系统采用专用射频芯片 nRF0433,工作于 ISM 频段,采用 FSK 调制方式,以串行通信方式实现了数据的半双工无线传输,传输距离可达 200 米,具有抗干扰能力强的特点。并就其在铁路货场中的应用作了介绍。

1.50 一种新的感知工具——电子标记笔

摘自《微型机与应用》月刊,2002 年第 2 期

本文在讨论远程指针技术基础上,提出了一种新的感知工具——电子标记笔的实现方案,对其所采用的技术进行了阐述。

1.51 智能住宅用户控制器设计

摘自《微计算机信息》月刊,2002 年第 12 期

本文讨论了智能住宅用户控制器的设计方法,采用 AT89C52 单片机实现的节点控制器适用于低成本的智能住宅。特别是单片机系统中的容错功能设计方案,结构简单,效果明显。系统网络采用总线结构。

1.52 利用 GPS 对计算机实现精确授时

摘自《计算机测量与控制》月刊,2002 年第 7 期

本文阐述了如何利用 GPS 对单个计算机进行精确授时。系统利用软件,从 GPS 接收器获取 UTC 时间码和秒定时信息,并对计算机时钟进行校准。该系统具有投资省、精度高、用途广等特点,取得了满意的效果。

1.53 IP 代理远程测控系统

摘自《电子测量技术》双月刊,2002 年第 1 期

TCP/IP 是因特网上广泛采用的组网协议。通过将 TCP/IP 协议嵌入到电子装置中,可以通过基础网络设施实现远程测控功能。本文介绍了一种通过 IP 代理来实现功能的软硬件方案。文中探讨了系统组成、代理实现的软硬件结构以及代理与测控前端间的数据传递方式。

1.54 曼彻斯特码编码与解码硬件实现

摘自《电子测量技术》双月刊,2002 年第 6 期

针对曼彻斯特码传统编码和解码过程中出现的相位模糊问题,通过对曼彻斯特和信息帧的特点进行分析,本文给出了一种曼彻斯特编码与解码的硬件实现新思路,解决了相位模糊的问题,并在工程实践中得到了应用与验证。

1.55 便携式设备中电源软开关设计的一种方法

摘自《电子技术》月刊,2002 年第 9 期

本文简要介绍了低压差电源管理芯片 MAX603/604 的基本结构和功能,以 MCS－51 单片

机系统为例,详细介绍了电源“软开关”设计的一种方法,给出其具体硬件设计电路。

1.56 便携式设备的电源方案设计

摘自《电子技术》月刊,2002 年第 10 期

本文主要讨论便携式设备中与电源方案设计相关的几个方面:电池特性、充电方案、电源转换和电源管理,并且结合实际应用,给出了一种典型的设计实例。

1.57 StrongARM 及其嵌入式应用平台

摘自《单片机与嵌入式系统应用》月刊,2002 年第 7 期

在简要介绍嵌入式 CPU SA1110/SA1111 的基础上,本文着重从硬件方面介绍其接口的扩展方法及扩展信号定义,可满足平台中对所需接口的要求。

1.58 嵌入式系统在光传输设备中的应用

摘自《无线电工程》月刊,2002 年第 7 期

本文介绍了运用嵌入式 Linux 系统和单片机管理光传输接入设备及在 Windows 软件界面上进行远程监控的实现过程。

1.59 光纤无源器件技术的发展方向

摘自《现代通信》月刊,2002 年第 1 期

光纤无源器件是光纤通信系统中的重要组成部分,本文在介绍光纤无源器件的基础上,深入讨论了光纤连接器小型化、光纤耦合器的宽带化、波分复用器的复杂化、光开关的矩阵化以及无源器件的集成化诸问题。

二、综合应用

2.1 数据存储技术的应用

摘自《计算机系统应用》月刊,2002 年第 11 期

本文详细介绍了数据存储技术在当今计算机世界中的应用情况,并分别说明了目前主要流行的几种数据存储技术的特点、发展趋势、适用范围。

这些存储技术包括:直接连接存储、RAID 技术、存储网络技术、SAN 技术、NAS 技术、IP 存储技术、磁带技术与光存储技术。

2.2 SL11R 单片机外部存储器扩展

摘自《电子技术应用》月刊,2002 年第 11 期

本文介绍了 USB 接口单片机 SL11R 进行外部存储器扩展的方法和实例,并测试了外部 SRAM 及 EDO DRAM 的工作速度。

2.3 构成大容量非易失性 SRAM 方法分析

摘自《电测与仪表》月刊,2002 年第 2 期

本文分析了静态存储器的待机节电模式,介绍了构成非易失性 SRAM 两种方法:一是利用非易失性 RAM 控制器和静态存储器构成 NVSRAM;二是利用继电器和静态存储器构成 NVSRAM。

2.4 一种专用高速硬盘存储设备的设计与实现

摘自《电子技术应用》月刊,2002 年第 8 期

本文介绍一种专用高速硬盘存储设备,可以脱离微机平台实时将高速数据送入 SCSI 硬盘。给出了该设备的系统结构和硬件设计方法。

2.5 基于 CDROM 的嵌入式系统设计

摘自《电子技术》月刊,2002 年第 6 期

本文介绍了以 CDROM 作为大容量只读数据存储设备的嵌入式系统的设计方法,并通过汽车导航系统中的具体应用介绍了其软硬件的实现。

2.6 串行 E^2PROM 的应用设计与编程

摘自《微型机与应用》月刊,2002 年第 1 期

本文介绍了目前比较常用的 3 种串行 E^2PROM 的原理、应用设计及编程,并举例说明。同时给出了采用 PLM/96 编写的程序并对编程过程进行简要分析。

2.7 利用 UART 扩展大容量具有 SPI 接口的快速串行 E^2PROM 的方法

摘自《计算机应用》月刊,2002 年第 9 期

文中以具有 SPI 接口的快速串行 E^2PROMX25F128 为例,阐述了利用 UART 扩展 SPI 的必

要性和可行性，给出了硬件连接图。经过时序分析得出，进行适当的数据排序变换，可用 UART 方式 0 有效地模拟 SPI 对快速 E^2PROM 进行读写操作。文中着重介绍了扩展 SPI 的硬软件设计方法。

2.8 用单片机实现异步串行数据再生

摘自《自动化技术与应用》双月刊，2002 年第 4 期

本文介绍了一种运用 AT89C2051 单片机实现异步串行数据再生，消除由于通信码速调整造成的异步串行数据波形畸变的方案，可应用于远程监控。

2.9 非易失性数字性电位器与单片机的接口设计

摘自《微计算机信息》月刊，2002 年第 4 期

本文介绍了近来出现的非易失性数字电位器的主要特点，就一个具体电路详细叙述了单片机控制数字电位器 X9313 的硬件联结与软件编程方法。

2.10 数控电位器在频率可调信号源中的应用

摘自《工业控制计算机》月刊，2002 年第 1 期

本文研究了应用单片机数控电位器，制作了一个智能信号源，对 X9312 数控电位器作了简介，给出一个实用的算法。

2.11 单片机上一种新颖实用的 e^x 函数计算方法

摘自《计算机测量与控制》月刊，2002 年第 4 期

本文详细介绍了一种在 8096/98 系列单片机上实现的新颖的 e^x 函数的计算方法。该算法是在数的二进制分解的基础上，充分利用了指数函数的特性和浮点数在单片机中的存储特点，计算速度快、精度高，有很强的实用性。

2.12 单片机系统设计的误区与对策

摘自《电子技术应用》月刊，2002 年第 2 期

本文用电磁兼容性理论剖析了单片机系统设计中的某些传统观念，指出其过时和失误之处，给出了根据电磁兼容性理论设计单片机系统的新理念，文中还给出了双时限看门狗、定时复位看门狗、抗快速脉冲群滤波器、电磁兼容 PCB 等新的设计方法。

2.13 基于 SystemC 的嵌入式系统软硬件协同设计

摘自《计算机应用研究》月刊，2002 年第 6 期

本文提出了一种基于 SystemC 的嵌入式系统软硬件协同设计方法。SystemC 是 OSCI (Open SystemC Initiative) 组成制定和维护的一种开放源代码的 C++ 建模平台，提供支持硬件建模和仿真的 C++ 类库及相应的仿真内核。通过 SystemC 的支持，该方法在整个嵌入式系统设计流程内使用 C++ 语言来统一描述硬件和软件，实现软硬件的协同设计和仿真。该方法同传统的设计方法相比更加灵活和有效。

2.14 一种基于 JTAG TAP 的嵌入式调试接口设计

摘自《微计算机应用》双月刊,2002 年第 6 期

基于 IEEE 1149.1 JTAG 架构,本文设计了一个嵌入式调试接口。从该接口的整体结构划分到内部各组成单元的设计,进行了详细的阐述。经过仿真验证,证明其设计可靠、方案可行,具有很好的实用价值。

2.15 工作频率可动态调整的单片机系统设计

摘自《电子技术应用》月刊,2002 年第 12 期

本文介绍一种采用可编程 CMOS 频率合成器 DS1077 设计的单片机系统,使单片机根据环境需要动态调整系统的工作频率,既能满足系统要求的实时处理能力,又尽可能地降低系统的耗电量及引起的电磁干扰。同时还讨论了系统改变工作频率后对 RS232 串行通信的影响以及解决办法,确保该系统在不同的工作频率下仍能正常进行 RS232 串行通信。

2.16 嵌入式系统高效多串口中断源的实现

摘自《电子产品世界》月刊,2002 年第 9 期

本文针对嵌入式系统精简特性,提出一种用 CPLD 实现的一个中断源高效管理多个串行口,不但节省了系统资源,并且实现多个串行口中断的无漏检测与服务的有效方法。

2.17 AVR 单片机计时器的优化使用

摘自《制造业自动化》月刊,2002 年第 9 期

本文对利用 AVR 单片机的 MEG103 芯片作为主控制单片机的组合称系统设计过程中出现的计时器数量不足的问题,提出了一个计时器优化使用的解决方案。

2.18 可编程定时/计数器提高输出频率准确度方法

摘自《单片机嵌入式系统应用》月刊,2002 年第 6 期

用可编程定时/计数器作脉冲发生器时,输出脉冲频率等于输入时钟频率除以计数值,但其数值是离散的,期望输出频率只能用这些离散频率点来近似,频率准确度随输出频率升高而下降。本文提出了采用提高输入时钟频率,增加输入时钟源数来提高脉冲发生器输出频率准确度。

2.19 用插值调整法设计单片机串行口波特率

摘自《单片机与嵌入式系统应用》月刊,2002 年第 3 期

传统方法设计单片机串行口波特率时,往往要使用特殊频率的晶振。本文在分析 MCS-51 单片机串行口工作原理的基础上,提出基于 12 MHz 晶振的单片机系统,通过编程实现所需波特率的插值调整设计方法。

2.20 “频率准确度”自动校准

摘自《电子测量技术》双月刊,2002 年第 1 期

本文主要介绍了 VXI 总线校准系统的基本构成与功用,以及计数器/频率计类仪器中的“频率准确度”参数自动校准的实现。重点提示了在该设计过程中遇到的一些关键问题及其解决方法。

2.21 双时基频率校准电路

摘自《仪表技术与传感器》月刊,2002 年第 8 期

提供精确的时间基准是精密仪器和高精度设备计时和工作的关键技术之一。本文中采用两套晶振电路,通过分频产生高频和低频两个频率信号,采用编码的方法对两个频率信号在每一个时钟周期内进行动态校核,只有两个晶振的频率在允许的精度范围内,或同时增大(缩小)相同的倍数,校核后频率信号才是正确的,且能给系统提供精确的时间基准。该方法能有效防止系统的时钟周期因意外的原因伸长或缩短。

2.22 电压-频率转换电路的动态特性分析及求解

摘自《仪表技术与传感器》月刊,2002 年第 6 期

在测量与仪表技术领域,V/F 转换器作为一种信号变换和处理技术,应用日益广泛。本文从 V/F 转换器内部工作原理出发,分析了 V/F 转换电路的动态特性,提出了能说明 V/F 转换过程快慢程度的一个动态参数概念,即转换时间,并求解出普遍有效的一般表达式。

2.23 单片机测控系统的低功耗设计

摘自《电气时代》月刊,2002 年第 2 期

本文介绍了单片机低功耗应用系统特点,低功耗应用系统设计方法,重点介绍了单片机的低功耗运行、存储器的低功耗运行以及软件设计的低功耗措施。

2.24 MCS-96/196 三字节浮点库

摘自《单片机与嵌入式系统应用》月刊,2002 年第 5 期

本浮点子程序库只包含浮点加(FADD)、减(FSUB)、乘(FMUL)、除(FDIV)等子程序。程序清单见《单片机与嵌入式系统应用》网站:www.dpj.com.cn。

2.25 循环冗余校验方法研究

摘自《微处理机》季刊,2002 年第 2 期

本文对循环冗余校验码的生成步骤及校验原理做了研究和探讨,并给出了应用实例。

2.26 32 位微处理器下伪 SPI 技术的研究与实现

摘自《微型机与应用》月刊,2002 年第 2 期

SPI 是一种高效的串行双向同步通信接口,适合于主机与外围设备进行通信,但有些 MCU 不带 SPI。为实现这一类 MCU 与带 SPI 的外围设备进行通信,本文介绍了一种软硬件结合的

技术,并以伪 SPI 命名,模拟 SPI 的工作。

2.27 智能仪表 LED 点阵显示模块的设计

摘自《电测与仪表》月刊,2002 年第 7 期

本文中提出了一种智能仪表 LED 点阵显示模块的设计方案。与通用的 LED 数码显示方式相比,该方案更为灵活,不仅可以以多种方式显示测量值的幅值,而且可以显示汉字、特殊图符等,实现了仪表汉字和图形混合显示。

2.28 点阵式图形 VFD 与单片机的硬件接口及编程技术

摘自《电测与仪表》月刊,2002 年第 9 期

点阵式图形 VFD 亮度高、视角大、光线柔和,已得到了广泛应用。本文主要介绍 128S64AA1 VFD 模块的工作原理、与 8031 的硬件接口及软件编程时的一些技巧和注意事项。

2.29 内置汉字字模的 EPROM 制作技术

摘自《工业控制计算机》月刊,2002 年第 5 期

本文介绍了一种将汉字点阵字模固化于 EPROM 的技术。作为一种标准接口模块,可以应用在例如点阵式液晶显示器 LCD 等场合。

2.30 利用 VC + + 实现汉字字模的提取与小汉字库的生成

摘自《单片机与嵌入式系统应用》月刊,2002 年第 1 期

汉字显示是在只有西文操作系统的情况下,以及一些无操作系统的小应用系统中,需要经常用到的技术。如何得到汉字的字模是汉字显示技术中首先必须解决的问题。本文利用 VC + + 实现一种汉字字模的提取和小汉字库的生成。界面友好,且具有良好的通用性。

2.31 高分辨率电压与电流快速数据采集方法

摘自《电子测量技术》双月刊,2002 年第 1 期

本文中提出了一种高分辨率电压与电流的快速数据采集方法。该方法使用专用集成电路与微控制器组成基本测试电路,不仅提高了测量精度而且使其校准和调整更加方便。与双积分式数据采集方法相比,速度提高了三个数量级,与逐次逼近式数据采集方法相比,分辨率与提高了三个数量级且具有部分数值计算功能。

2.32 单片机与数字温度传感器 DS18B20 的接口设计

摘自《计算机测量与控制》月刊,2002 年第 4 期

本文简要介绍了 DALLSA 公司生产的一线式数字温度传感器 DS18B20 的基本原理、功能特点及工作时序,给出了 DS18B20 与单片机接口的软件编程实例。

2.33 新型温度传感器 DS18B20 高精度测温的实现

摘自《微处理机》季刊,2002 年第 2 期

文中介绍了一种数字式的温度传感器 DS18B20,提出了一种基于 89C2051 的测温电路及软件框图在内的实现方法。

2.34 MAX6576/6577 集成温度传感器

摘自《电工技术》月刊,2002 年第 5 期

文中介绍了 2 种新型 MAX6576 和 MAX6577 的工作原理和典型应用,详细阐述了智能化温度检测系统的电路设计。

2.35 AD22105 型低功耗可编程集成温度控制器

摘自《计量技术》月刊,2002 年第 11 期

本文介绍了 AD22105 型低功耗、电阻可编程集成温度控制器的工作原理、典型应用及电路设计要点。

2.36 基于 IEEE 1451.1 的网络化智能传感器设计

摘自《单片机与嵌入式系统应用》月刊,2002 年第 8 期

IEEE 1451 是一种从传感器或执行器到微处理器及网络之间的硬件和软件接口标准。本文根据 1451.1 标准,研制面向 Internet 的网络化智能机器人手爪传感器系统,并给出硬件设计框图和软件流程。

2.37 数字式温度传感器与仪表的智能化设计

摘自《仪表技术》双月刊,2002 年第 2 期

本文介绍利用 DS1620 等数字式温度传感器进行仪表的温度补偿算法以及邦联诊断、报警等,实现仪表的智能化设计方案。

2.38 用单片机软件实现传感器温度误差补偿

摘自《现代电子技术》月刊,2002 年第 10 期

本文介绍了用单片机软件实现传感器温度误差补偿一种方法,在建立了传感器误差的数学模型基础上,给出了如何用软件实现温度补偿的流程和具体方法,对自动检测仪表中提高传感器的测量精度,提供一条行之有效的途径。

2.39 Σ-Δ A/D 转换器的原理及分析

摘自《电测与仪表》月刊,2002 年第 11 期

本文简介了 Σ-Δ A/D 转换器的原理,初步推导了 Σ-Δ A/D 转换器的数学变换过程及其量化误差成型过程,定性分析了影响 Σ-Δ A/D 转换器测量精度的因素。

2.40 一种提高 A/D 分辨率的信号调理电路设计

摘自《电测与仪表》月刊,2002 年第 3 期

A/D 转换器的分辨率决定着数据采集电路的量化精度。通过对信号调理电路的合理设计,可以在原有的基础上使数据采集电路的分辨率提高一位。这可以使整个数据采集系统的性能得到进一步的提高。文中介绍了提高分辨率的意义,相应的信号调理电路以及量化值的处理方法。

2.41 高精度数据转换器接口技术

摘自《电子测量技术》双月刊,2002 年第 3 期

为保证高精度数模转换器的性能,在设计和使用高精度 DAC 的外围接口电路时应考虑到数据输入、基准电压、输出驱动、电源管理、数字时序逻辑等因素,只有严格设计才能保证 DAC 的正常使用,并减小外围接口电路带来的失调误差、增益误差等。文中介绍了参考电压设计、输出驱动电路设计、时序逻辑设计以及电源及地线设计等。

2.42 高精度双积分 A/D 转换器与单片机接口的新方法

摘自《自动化仪表》双月刊,2002 年第 1 期

本文提出一种高精度双积分 A/D 转换器 7135 与 89C52 单片机的接口电路。它具有接口简单,能最大限度地缩短采样程序的时间等优点。分析了该方法的接线原理,给出了中断服务程序。

2.43 一种高速 A/D 与 MCS－51 单片机的接口方法

摘自《电测与仪表》月刊,2002 年第 1 期

本文中介绍了一种高速 A/D 转换器 (CA3318) 与 MCS－51 单片机的接口方法,其采样速率可达 10 MHz 以上,并给出了详细的电路设计,分析了其工作过程。该电路可应用于高速数据采集系统、数字图像处理等领域。

2.44 基于串行 FIFO 双口 RAM 的高速 A/D 转换采集系统的设计

摘自《电子技术》月刊,2002 年第 5 期

文中介绍了串行 FIFO 双口 RAM IDT7206 和高速 A/D 转换器 AD7677 的功能,详细说明了由这两种芯片为主要元件构成的高精度 (16 位)、高速 (1 MHz) A/D 转换采集系统的设计方法,给出了设计的具体电路图。设计的优点是采样速度快,控制简单,与各种单片机系统接口方便。

2.45 超高速数据采集系统的设计与实现

摘自《电测与仪表》月刊,2002 年第 12 期

文中介绍了一种基于单片 ADC、采用多路存储技术、采样速度达 600 MHz 的超高速数据采集系统的设计思想、实现方案和性能分析。系统采用 8 路存储技术,大大减轻了数据存储和传输的压力,提高了系统工作的可靠性和稳定性。

2.46 廉价隔离型高精度 D/A 转换器

摘自《单片机与嵌入式系统应用》月刊,2002 年第 5 期

文中介绍隔离型高精度 D/A 转换器的设计方法:由单片机 89C52 产生 PWM,经过光电隔离和一个双 PC 电路,将数字信号转换为直流电压信号,再经过电压/电流转换电路(V/I),输出 0~20 mA 电流信号;通过软件校正,达到较高的精度。

2.47 智能卡及其应用技术研究

摘自《微型机与应用》月刊,2002 年第 12 期

本文在介绍智能卡历史沿革、复合卡的基础上,阐述了智能卡应用和发展趋势,着重分析了 RFID、软件技术、安全架构等智能卡的关键技术。

2.48 Jupiter GPS 接收机数据的提取

摘自《电子技术》月刊,2002 年第 6 期

在各种利用 GPS OEM 板进行二次开发应用的工作中,对 GSP 数据的提取是必不可少的过程。本文给出了基于 SBS-PC-104 嵌入式计算机提取 Jupiter GPS 接收机数据的过程和方法,并且给出了相应的程序流程图。

2.49 基于单片机的脉冲频率的宽范围高精度测量

摘自《仪表技术》双月刊,2002 年第 6 期

本文介绍一种基于 AT89C52 单片机高精度测量宽范围脉冲频率的方法,提出了硬件方案和"限时定数"软件算法。

2.50 电源模块输入软启动电路的设计

摘自《电子产品世界 B》月刊,2002 年第 8 期

本文就艾默生 DC/DC 变换器应用中的输入软启动电路作了详细的研究和对比分析,其中的应用电路对艾默生 DC/DC 变换器用户或其他爱好者有较好的参考价值。

2.51 不停车电子收费系统关键技术

摘自《电子测量技术》双月刊,2002 年第 2 期

高速公路不停车收费系统区别于其他的人工收费系统,主要在于自动车辆识别系统和自动车型分类系统。本文介绍了自动车辆识别系统和自动车型分类系统两项关键技术及其实现技术。

2.52 一种直接采用计算机串行口控制步进电机的新方法

摘自《电子技术应用》月刊,2002 年第 8 期

本文介绍了一种计算机串行口经二次开发,用作步进电机控制器的新方法。计算机通过向串行口发送数据产生控制脉冲,实现对步进电机的控制。

2.53 8051 系列单片机通用鼠标接口程序设计

摘自《测控技术》月刊,2002 年第 11 期

本文在分析鼠标通信协议的基础上,探讨了单片机与鼠标接口软件设计的原理及方法,给出了 8051 单片机与 Microsoft 兼容鼠标的具体接口程序。并分析了其他应用时设计接口软件的有关问题。

2.54 可编程 ASIC 与 MCS－51 单片机接口设计及实现

摘自《微计算机信息》月刊,2002 年第 4 期

针对可编程 ASIC 和 MCS－51 单片机的特点,本文对两者之间的接口方式进行了分析。用 Verilog HDL 给出了几个实用的接口参考程序。

三、软件技术

3.1 无线信息设备的理想操作系统 Symbian OS

摘自《单片机与嵌入式系统应用》月刊,2002 年第 8 期

本文从内存管理、进程调度、消息传递以及与内存管理有关的编程等方面详细介绍 Symbian OS,说明它是最适合无线信息设备的操作系统。

3.2 TMS320C55x 嵌入式实时多任务系统 DSP/BIOS II

摘自《电子产品世界 B》月刊,2002 年第 5 期

DSP/BIOS 是运行在数字信号处理器 (DSP) 中的一个小型软件,它为开发者提供对程序的控制执行和对变量的实时监测。而且可以合理地实时多线程系统进行时间规划。本文对 DSP/BIOS II 的基本特征与应用作了介绍。

3.3 两种嵌入式操作系统的比较

摘自《电子产品世界 B》月刊,2002 年第 10 期

嵌入式操作系统是嵌入式系统应用的核心。本文通过对两种典型的开源嵌入式系统的对比,分析和总结了嵌入式操作系统应用中的若干问题,归纳了嵌入式操作系统的选型依据。

3.4 用自由软件开发嵌入式应用

摘自《单片机与嵌入式系统应用》月刊,2002 年第 10 期

本文介绍了 Linux 的嵌入式应用的有关方面,包括 Linux 开发环境的建立、运行于 Linux 操作系统下的自由软件 GNU gcc 编译器、嵌入式操作系统 uCLinux 及 μC/osII、ColdFire5307 开发过程以及搭建 Linux 下开发 ColdFire 软件平台过程。

3.5 开放源代码软件的应用研究

摘自《计算机系统应用》月刊,2002 年第 6 期

目前,开放源代码的软件开发模式已经越来越多的被采用,其基本思想是通过公开软件的源代码使得不同的开发人员可以互相交流,发现错误从而提升软件质量。在某些环境下运行的软件由于对可靠性的高度要求,采用开放源代码的模式进行软件开发将能最大化的做到这一点。本文试从商业模型、法律规范、实现技术这三个角度论述了开放源代码软件的应用。

3.6 清华嵌入式软件系统的解决方案

摘自《计算机测量与控制》月刊,2002 年第 8 期

本文简要论述了嵌入式软件系统的总体结构,分析了系统中各个模块的组成及其作用,并对清华嵌入式操作系统 (THOS) 内核的构架也作了详细的介绍。

3.7 单片机应用程序的高级语言设计

摘自《仪表技术》月刊,2002 年第 1 期

本文介绍利用 Franklin C 高级语言编写单片机用户应用程序的特点,对汇编语言难以处理的数据分析、运算和显示等方面,它具有较好的处理效果。

3.8 基于 RTX51 的单片机软件设计

摘自《单片机与嵌入式系统应用》月刊,2002 年第 12 期

RTX51 是 Keil 公司开发用于 51 系列单片机的实时多任务操作系统。RTX51 有 2 个模式,即完全模式与最小模式。本文详细介绍了 RTX51 运行的任务状态、RTX51 事件、中断处理,并给出了 RTX51 在单片机控制的 GPS 接收板上的应用。

3.9 多网口通信在 VXWORKS 中的实现

摘自《微计算机信息》月刊,2002 年第 8 期

随着嵌入式实时系统的广泛应用,通信也越来越重要了。本文阐述了在 Motorola MPC860 系列芯片上采用 vxworks 实时嵌入式操作系统时实现双网口通信的方法。文中给出了实现的具体原代码。

3.10 嵌入式实时操作系统中实现 MBUF

摘自《工业控制计算机》月刊,2002 年第 3 期

MBUF 是 TCP/IP 中的概念,本文从它在嵌入式系统中的应用,讨论了它的基本原理和在嵌入式实时操作系统中的实现方法以及相关的内存管理支持。

3.11 硬实时操作系统——RT－Linux

摘自《电子技术应用》月刊,2002 年第 4 期

本文介绍了 RT－Linux 的两个重要特点:硬实时性和完备性,及其在嵌入式系统应用中的一些重要功能,并结合实时处理的具体实例对其编程方法加以说明。

3.12 Linux 嵌入式系统的上层应用开发研究

摘自《计算机应用》月刊,2002 年第 3 期

本文介绍 Linux 系统下使用快速开发工具包 (Fast Light Tool Kit) 进行 X windows 下的应用程序开发方法,同时对生成的源码进行解析,阐述了 Fast Light Tool Kit 中实现 C++面向对象机制的方法。

3.13 嵌入式 Linux 内核下串行驱动程序的实现

摘自《电脑与信息技术》双月刊,2002 年第 5 期

文中阐述了 Linux 下驱动程序的基本概念和中断处理方法,并以 uClinux 下的异步串行口为例,详细分析了串行驱动的实现过程,该方法和思路为在嵌入式 Linux 下访问其他串行通信外设提供了很好的借鉴。

3.14 嵌入式 Linux 的中断处理与实时调度的实现机制

摘自《计算机工程》月刊,2002 年第 10 期

本文论述了实时嵌入式 Linux 的实时机制。详细讨论了实现实时的两种方法:一种是硬件和软件相结合的中断机制,一种是实时调度机制和各种实时算法。

3.15 基于 Linux 平台的应用研究

摘自《微计算机应用》月刊,2002 年第 4 期

基于 Linux 的三种不同性能(内核模块的动态加载、支持不同的文件系统和高度的安全性),本文研究了三种不同的应用开发方案。重点阐述应用开发的实现思想,并提出了 Linux 广阔的应用前景。

3.16 基于 Linux 的嵌入式系统开发

摘自《计算机应用》月刊,2002 年第 7 期

在后 PC 时代,对嵌入式系统的研究与开发成为当前的一个热点。Linux 由于其种种优点成为很多厂家开发嵌入式应用的 OS,本文介绍了嵌入式 LinuxGUI 的特点,并且结合与天大天财公司合作开发的嵌入式浏览器的项目对嵌入式 GUI 和嵌入式浏览器作了比较详细的介绍。

3.17 基于 Linux 的嵌入式系统设计与实现

摘自《计算机工程》月刊,2002 年第 6 期

本文讨论了 Linux 在嵌入式系统与桌面应用方面的区别,包括在缩小 Linux 的系统需求方面的一些技术。并在其上设计与实现了一个基于 Linux 的嵌入式操作系统及其上层 GUI 环境。

3.18 基于 RT-Linux 的实时控制系统

摘自《电子技术应用》月刊,2002 年第 8 期

本文从技术背景、系统结构、硬件和软件设计等方面论述了基于 RT-Linux 的闸门实时控制系统的组成、原理以及实现方法,并着重分析了软件实现的关键问题。

3.19 基于 RT-Linux 的实时机器人控制器研究

摘自《计算机工程与科学》双月刊,2002 年第 6 期

本文结合机器人控制器的特点,提出了采用 RT-Linux 操作系统改进机器人控制实时性的方法,并对机器人控制器任务采用多线程机制进行实时域和非实时域的划分。最后给出一个在 RT-Linux 操作系统下实现硬件设备实时驱动程序的实例。

3.20 嵌入式 Linux 系统在温室计算机控制中的应用

摘自《计算机系统应用》月刊,2002 年第 2 期

本文介绍了 Linux 系统在现代温室计算机控制中的应用,主要就硬件实现平台,Linux 针对本系统的特点,嵌入式 Linux 内核实现,基于 Linux 的嵌入式软件实现方法作了较为详细的

说明。

3.21 基于 Linux 的 USB 驱动程序实现

摘自《计算机应用》月刊,2002 年第 8 期

由于 Linux 良好的开放性和 USB 总线极佳的通用性,USB 设备在 Linux 操作系统中得到了广泛的应用。文中首先介绍 Linux 驱动程序的架构,然后介绍 USB 设备,重点说明 USB 驱动程序的实现。

3.22 Linux 环境下实现串口通信

摘自《微型电脑应用》月刊,2002 年第 6 期

本文给出在 Linux 环境下实现串口通信设计的方法,简述了通信的基本原理,介绍了文件打开方法和参数设置方法。

3.23 Linux 系统下 RS－485 串行通信程序设计

摘自《计算机应用研究》月刊,2002 年第 2 期

本文介绍了在 Linux 操作系统下 RS－485 多用户串行接口卡的安装、配置及串口的程序设计技术。详细叙述了 Linux 系统下串行通信资源的程序设计方法、相关的系统调用、程序编制技巧。

3.24 Linux 系统下蓝牙设备驱动程序研究和实现

摘自《计算机应用研究》月刊,2002 年第 7 期

本文在系统分析蓝牙协议和驱动层框架的基础上,定义了 Linux 下蓝牙核心数据结构,给出了 Linux 下蓝牙设备驱动程序架构,编写了 Linux 系统下蓝牙设备 UART 驱动程序的开发。

3.25 基于 μCLinux 和 GPRS 的无线数据通信系统

摘自《电子技术》月刊,2002 年第 11 期

本文在介绍嵌入式操作系统 μCLinux 和 GPRS 网络的基础上,设计并实验了一套基于嵌入式 μCLinux 和 GPRS 的无线数据通信系统,并展望了该系统的实际前景。

3.26 嵌入式 Linux 开发平台的 USB 主机接口设计

摘自《电子产品世界 A》月刊,2002 年第 12 期

本文讨论了怎样在嵌入式 Linux 开发平台下设计 USB 主机接口设备,以及在遵循 GPL 协议下开发该设备的驱动程序。

3.27 CAN 通信卡的 Linux 设备驱动程序设计实现

摘自《电子技术应用》月刊,2002 年第 1 期

本文介绍了 Linux 下设备驱动程序的结构,描述了 CAN 通信卡设备驱动程序的软件框架以及如何将 CAN 设备驱动程序加入到 Linux 系统内核中。讨论了具体实现中为了提高通信效率和通信能力,改进设备驱动程序的缓冲区管理以及利用 Linux 的特点合理设计中断处理程序。

3.28 μC/OS－II 实时操作系统内存管理的改进

摘自《电子技术应用》月刊,2002 年第 5 期

本文分析了 μC/OS－II 实时操作系统在内存管理上存在的不足,提出了改进方法,通过一个具体实例描述了该方法的实现。

3.29 μC/OS－II 在总线式数据采集系统中的应用

摘自《电子技术应用》月刊,2002 年第 7 期

本文介绍源代码开放的 μC/OS－II 实时操作系统,以及基于此技术的总线式数据采集系统,讨论了 μC/OS－II 在实际开发应用中应注意的几个问题,并通过实例论述实时操作系统在单片机系统应用开发的广阔前景。

3.30 实时操作系统 μC/OS－II 在 MCF5272 上的移植

摘自《电子技术应用》月刊,2002 年第 9 期

本文介绍了实时操作系统 μC/OS－II 的特点和内核结构,并首次实现了 μC/OS－II 在 Motorola 嵌入式处理器 MCF5272 上的移植。

3.31 μC/OS－II 在 51XA 上的移植应用

摘自《工业控制计算机》月刊,2002 年第 10 期

本文介绍了嵌入式实时操作系统 μC/OS－II 的工作原理,分析了移植到 Philips 的 51XA 芯片上所要完成的工作,并给出了移植过程中出现的问题及解决的方法。

3.32 实时嵌入式内核在 DSP 上的移植实现

摘自《工业控制计算机》月刊,2002 年第 6 期

本文介绍如何将实时嵌入式内核 μC/OS－II 移植到 TI 公司的 DSP 处理器 TMS320C5409 上。主要说明有关该内核中跟处理器相关部分的编程,以及它们在系统中的作用,这种实时操作系统调度任务可达 63 个。

3.33 利用全局及外部变量实现 C51 无参数化调用 A51 函数

摘自《电子技术应用》月刊,2002 年第 1 期

利用 C51 全局及外部变量,可实现无参数化调用 A51 函数,不但避开了传统 C51 调用 A51 时繁琐的接口约定,而且把在 A51 中所用到的变量全部放至 C51 程序中而不必考虑变量在内存中的位置,使编程更加简洁。本文用实例验证了该方法的优越性和有效性。

3.34 基于状态分析的键盘管理软件设计

摘自《单片机与嵌入式系统应用》月刊,2002 年第 6 期

本文介绍一种基于状态分析的人机交互接口设计方法,提出运用状态分析法设计人机接口的几个关键步骤。运用此方法,可以很方便、快速地设计出各类人机交互接口。

3.35 PS/2 接口 C 语言通信函数库设计

摘自《单片机与嵌入式系统应用》月刊,2002 年第 9 期

本文深入分析 PS/2 接口通信协议,实现了 C 语言通信函数库,可以方便地应用于 PS/2 设备的使用和开发。程序在 AVR 单片机上实现并通过检验,最后给出一个使用该函数库的程序。

3.36 DS18B20 接口的 C 语言程序设计

摘自《单片机与嵌入式系统应用》月刊,2002 年第 7 期

DS18B20 是 DALLAS 公司生产的一款数字温度传感器。具有精度高、全数字化、连线少等优点,但其 I/O 时序要求严格,使大多数编程人员不得不用汇编语言编写接口程序。本文介绍 DS18B20 数字温度传感器的 C51 接口程序及其编程方法和编程思路。

3.37 基于 Keil C51 的 SLE4428 IC 卡驱动程序设计

摘自《现代电子技术》月刊,2002 年第 9 期

SLE4428 逻辑加密卡由于其具有容量大、安全保密、价格低廉等优点,已被广泛应用于交通、税务等数据安全场合。在以往的 IC 卡应用系统中,其底层单片机程序往往用汇编程序开发,但是汇编语言可读性差,难以维护,由于 C 语言的众多优点,现在越来越多的程序用 C 语言来编写。本文介绍了采用 Keil C51 来开发 SLE4428 底层驱动程序,取得了较好的效果。

3.38 智能型并口用软件加密狗的设计

摘自《电子技术》月刊,2002 年第 6 期

本文分析了软件加密和信息加密的异同,引出了一种有别于常见的通过硬件对软件加密的新思路。提出了利用插在计算机并行口上的硬件设备并结合相应的软件实现对软件加密的方案。主机向"加密狗"的数据传输采取了 4 位并行传输的方法,主机则串行地利用状态线接收数据。这种设计大大提高了解密分析的复杂性。"加密狗"利用并行口却不独占并行口。

3.39 啤酒发酵控制器中的多任务分析与实现

摘自《单片机与嵌入式系统应用》月刊,2002 年第 7 期

文中分析啤酒发酵控制器中存在着的多种任务,用传统的前后台编程方法实现难度较大,使用实时操作系统可简化程序设计,给出通信任务设计的流程。

3.40 CAN 网络应用软件的设计与研究

摘自《微计算机信息》月刊,2002 年第 5 期

CAN 可以满足许多工业监控系统的要求。本文介绍了 CAN 网络的硬件构成,CAN 总线通信协议,并讨论应用 Francling 的 C51 对 CAN 网络应用软件设计的方法。

3.41 USB 软件系统的开发

摘自《计算机应用研究》月刊,2002 年第 3 期

本文介绍了在 Windows98 和 Windows2000 下开发 USB 的 WDM (Windows Driver Module) 设备驱动程序过程及 USB 软件系统的组成,并提供了相应的例程。

四、网络、通信与数据传输

4.1 网际协议过渡——从 IPv4 到 IPv6

摘自《微型机与应用》月刊,2002 年第 4 期

随着 IP 地址空间日趋枯竭,新一代寻址方案 IPv6 将成为未来因特网的主流协议。为使新网络兼容 IPv4,并逐步向 IPv6 过渡,本文重点阐述从 IPv4 向 IPv6 过渡的几种可行方案。

4.2 IPv6 简介

摘自《现代通信》月刊,2002 年第 3 期

IPv6 是下一个版本的因特网协议,本文介绍了 IPv6 与 IPv4 的不同点、移动 IPv6 的工作原理、移动节点位置的确定、移动 IPv6 的布告、数据包的选路。

4.3 传输控制协议 (TCP) 介绍

摘自《工业控制计算机》月刊,2002 年 15 卷第 3 期

本文分别介绍了位于运输层的 TCP (传输控制协议) 和 UDP (用户数据报协议) 的概念,解释了连接的过程及流量控制方法。

4.4 TCP/IP 协议的 ASIC 设计与实现

摘自《微电子学》双月刊,2002 年第 4 期

本文介绍了一种 TCP/IP 协议族传输、处理 TCP 数据段和 IP 数据报过程的 ASIC 设计——TCP/IP 协议处理器的硬件实现。简单介绍了 TCP/IP 协议,着重介绍了 TCP/IP 协议处理器系统结构以及各模块设计。硬件实现的 TCP/IP 协议处理器提高了 IP 数据的处理速度,更重要的是,将 Internet 网络数据传输从传统的依赖电子计算机系统的模式中解放出来,实现了脱离计算机系统环境建立 Internet 网络连接。

4.5 IP 电话的 TCP/IP 协议的实现方法

摘自《单片机与嵌入式系统应用》月刊,2002 年第 9 期

本文在介绍实现 IP 电话基本原理的基础上,详细阐述 IP 电话的 TCP/IP 协议的实现过程,并简要描述 1 次通话的全过程。

4.6 基于嵌入式 TCP/IP 协议栈的信息家电连接 Internet 单芯片解决方案

摘自《电子技术应用》月刊,2002 年第 6 期

以分析和实验为基础,研究了嵌入式 TCP/IP 协议栈 SX－Stack 的结构及运行原理,提出了使用 SX－Stack 构造单芯片嵌入式网络服务器,将信息家电接入 Internet 的新方案。该服务器的组成、构造方法和工作原理,并比较现有的使用 PC/网关设备的接入方案,分析了该方案的优点。单芯片嵌入式网络服务器中 SX－Stack 与用户应用程序的接口方法,用户登录软件及信息家电监测软件的设计方法。

4.7 基于以太网的家庭网络平台

摘自《工业控制计算机》月刊,2002 年第 11 期

本文对家庭网络平台及其主网关作了分析和研究,介绍了以太网作为主干网的家庭网络平台,并在此基础上,着重阐述在 Linux 系统下主网控制终端的设计与实现。

4.8 单芯片家庭网关平台 CX821xx

摘自《现代通信》月刊,2002 年第 6 期

CX821xx 家庭网络处理器是基于 ARM940T 处理器并集成多重网络接口硬件功能的单片化产品,本文介绍了 CX821xx 的性能特点,并介绍了在 SOHO 多电脑防火墙、电话家用网络、以太网至电力线家庭网桥的应用。

4.9 用于单片机的以太网网关——网络通

摘自《单片机与嵌入式系统》月刊,2002 年第 3 期

本文介绍的"网络通"是基于普通单片机的廉价以太网测控网关。它可以将具有 RS-232、RS-485 等接口的测控设备简单而且直接地连接在以太网(因特网)上,利用丰富的现成的以太网资源,组成一系列以太网的分布式测控系统。

4.10 基于"网络通"的单片机以太网-CAN 网关的应用

摘自《单片机与嵌入式系统应用》月刊,2002 年第 4 期

本文介绍利用"网络通"做成基于单片机的以太-CAN 网关,将 CAN 总线系统直接连接在以太网上。它既可以将普通 CAN 接口测控设备变成以太网络测控设备,通过互联网进行数据的传送;也可以在一些大型分布式测控系统中,采用以太网-CAN 接口模块,很方便地将原来现场总线控制系统改造为以太网络分布式控制系统。

4.11 第三代快速以太网控制器及其应用

摘自《微计算机应用》双月刊,2002 年第 3 期

本文介绍了 SMSC 公司最新推出的第三代快速以太网控制器 LAN91C111 的特性、结构及其在一个基于 IPSec 的加密系统中的应用。

4.12 工业以太网在控制系统中的应用前景

摘自《仪表技术》双月刊,2002 年第 3 期

本文介绍以太网技术在工业控制应用中的优势与面临的问题,讨论以太网的确定性、实时性,展示它应用于控制系统的前景。

4.13 工业以太网控制模块的研究与研制

摘自《工业控制计算机》月刊,2002 年第 11 期

本文介绍了以 RCM2250 嵌入式模块为核心的控制模块,以及嵌入式开发语言 Dynamic C,该系统支持 Web Server、FTP Server、SMTP 及 POP3 等,能在浏览器上通过程序接口 CGI 对嵌入式设备进行操作。文中介绍了通过浏览器设置、读取处理器内部时钟以及模拟量输入的原理及方法,并给出了控制模块开关量输入/输出的程序,阐述了控制模块与浏览器之间进行信息交换的原理和软件的实现方法。

4.14 以太网、控制网与设备网的性能比较与分析

摘自《仪表技术》双月刊,2002 年第 3 期

本文阐述了以太网、控制网和设备网介质访问控制的特点以及报文帧的结构。对它们在控制中的优缺点进行评价与分析。

4.15 嵌入式系统以太网控制器驱动程序的设计与实现

摘自《电子技术》月刊,2002 年第 4 期

在嵌入式系统中,通过以太网传输数据已被广泛地应用。实现过程中,在实时操作系统软件平台下,编写 TCP/IP 协议栈的以太网控制器驱动程序是一项重要的工作。本文总结了以太网控制器驱动程序的编写方法,同时介绍了相关知识。

4.16 WIN9X 下微机与单片机的串行通信

摘自《微计算机信息》月刊,2002 年第 2 期

本文总结了 WIN9X 下实现微机与单片机通信的方法,以飞行器箭地通信的等效为例,详细介绍了用 VC++6.0 实现串行通信的具体步骤,给出了部分程序清单。实践证明,这种方法简单、可靠,具有通用性。

4.17 利用 VB6.0 实现 PC 机与单片机的串口通信

摘自《微计算机信息》月刊,2002 年第 10 期

本文介绍了如何利用 Visual Basic6.0 实现串口通信,并以实际项目为例,介绍了 PC 机与单片机间串口通信的硬件实现和软件设计。

4.18 基于 VB6 的 PC 机与多台单片机通信的应用

摘自《微计算机信息》月刊,2002 年第 10 期

用基于 VB6 的 PC 机与多台 MCS-51 单片机通信,来实现对制药厂库房管理系统的数据采集与保存。本文阐述了通信协议,给出了系统结构框图和硬件线路图。

4.19 用 C++Builder6.0 实现 80C51 与 PC 串行通信

摘自《微计算机信息》月刊,2002 年第 10 期

本文介绍了在 C++Builder6.0 开发环境下,运用基于 Windows98 的串行通信接口 API 函

数和安装微软 VB 中提供的串行通信 ActiveX 控件 MSCOMM32 两种方法来实现 8051 与 PC 之间的串行通信。

4.20 VC + + 中实现基于多线程的串行通信

摘自《计算机测量与控制》月刊,2002 年第 10 期

本文介绍了基于 Windows 多线程特性及重叠 I/O 操作,采用 Visual C + + 6.0 实现一种高效的串行通信程序,可供同类设计参考。

4.21 RS - 232 串行通信线路的连接方法设计分析

摘自《现代电子技术》月刊,2002 年第 4 期

RS - 232 通信线路连接可以使用多种类型连接器、引脚配置和信号传输方式的任何一种。由于有多种选择,因此正确设计线路连接或者解决连接中的问题尤为重要。本文主要探讨 RS - 232 的线路连接,连接器和导线配置问题。

4.22 高效率串行通信协议的设计

摘自《仪表技术》双月刊,2002 年第 3 期

本文介绍在谐波分析仪串行通信协议中如何使用滑动窗口技术以大幅度提高数据传输效率的方法。

4.23 利用增强并口协议传输数据

摘自《微计算机应用》双月刊,2002 年第 1 期

本文介绍了用 EPP 协议传输数据的方法,给出了详细的技术解决方案。设计的传输数据电路最高可达 1.4MB/s 的传送速度。

4.24 应用于 RS485 网络的多信道串行通信接口的设计

摘自《电气自动化》双月刊,2002 年第 4 期

本文给出了一种基于 RS485 网络的微机与多信道串行接口的设计,阐明了设计原理,并从软硬件上给出了具体设计方法。

4.25 以 Visual C + + 实现 PC 与 89C51 之间的串行通信

摘自《仪表技术》双月刊,2002 年第 4 期

本文在 Visual C + + 开发环境下,运用 C + + 提供的通信控件,实现 PC 机与 89C51 单片机之间的串行通信。

4.26 智能多路 RS422 串行通信卡的设计

摘自《微型机与应用》月刊,2002 年第 6 期

在对通信控制方法进行分析的基础上,本文介绍了一种智能控制的多路 RS422 串行通信卡的硬件设计及监控程序的设计和调试。

4.27 RS232 接口转换为通用串行接口的设计原理

摘自《微型机与应用》月刊,2002 年第 8 期

本文以通用串行总线微控制器 Cypress CY7C64013 为例,介绍了 RS232 接口转换为通用串行总线接口的接口转换器的硬件和软件设计原理。

4.28 基于智能模块的 RS485 通信协议转换路由器

摘自《自动化仪表》月刊,2002 年第 3 期

不同的仪表可能有不同的 RS485 通信方式,当需要将这些不同协议仪表进行统一的通信时就需要进行协议的转换。本文应用一种内置 DOS 最小系统的智能模块,实现了不同 RS485 协议转换的路由器,并可以用于 RS485 与其他通信协议的网络互联。

4.29 RS232 接口转 USB 接口的通信方法

摘自《单片机与嵌入式系统应用》月刊,2002 年第 12 期

本文以 IC 卡门禁考勤系统为例,提出一种方案,使传统的 RS232 接口转化为 USB 接口后直接通过 USB 总线接入 PC,同时使 IC 卡门禁考勤设备增加了 USB 总线具有的热插拔、自动配置和智能电源管理等功能 着重剖析 USB 通信内核,探讨系统软硬件设计方案。

4.30 用 VB 实现 PC 与 PDA 的串行通信

摘自《计算机工程》月刊,2002 年第 3 期

本文分析了 VB 串行通信控件遥特点及 PC 和 PDA 在传输数据中的差异。提出利用 Byte 型变量处理通信数据,消除了使用 variant 型变量处理通信数据的缺陷。还简要讨论了传输协议的制定,解决了多种命令和内容传输的问题,真正实现了 PC 机和 PDA 的串行通信。

4.31 利用 Windows API 实现与 GPS 的串口通信

摘自《计算机与数字工程》双月刊,2002 年第 4 期

本文介绍了 GPS 和计算机串口的连接方式,和在 Windows95 以上环境下,利用 Windows API 函数开发基于 Windows 消息机制的多线程串口通信程序,实现与 GPS 通信的方法。

4.32 VB6.0 在无线通信中的应用

摘自《自动化仪表》月刊,2002 年第 4 期

本文结合集散智能温度无线巡检系统的课题讨论了 VB6.0 在无线通信中的应用并给出实例。介绍了无线通信系统硬件连接、软件流程及用 MSComm 控件实现无线通信的方法,对通信中遇到的主要问题,特别是无线通信的特殊问题和定时确定性问题作了较详细的讨论并给出解决方案,同时给出循环校验码 (CRC) 的一种算法。

4.33 用 PTR2000 实现单片机与 PC 机之间的无线数据通信

摘自《微计算机应用》双月刊,2002 年第 2 期

许多应用领域要用到单片机与 PC 机之间的无线数据传输,本文论述了利用 PTR2000 实

现单片机和 PC 机之间无线数据传输的原理和方法,并给出了应用实例。

4.34 基于光纤 RS232/RS485 传输系统

摘自《电子测量技术》双月刊,2002 年第 6 期

文中提出使用光纤作为传输手段,设计了一种传输系统,具有数据传输距离远、传输稳定可靠、接口灵活等特点,并且组网方便,非常适合构成数据采集、监控网络。

4.35 利用串口实现 PC 与 PDA 的同步通信

摘自《计算机应用研究》月刊,2002 年第 8 期

本文介绍了在某嵌入式系统中利用串口实现 PC 机和 PDA 的同步通信方法。首先讲述了本系统的组成部分及其在实际生活中的应用前景,然后阐述了使用 Win32 API 开发串口通信程序以及在特定 PDA 上用 UART 实现串口通信的步骤。

4.36 实现 32 位单片机 MC68332 与 PC 机串行通信的底层程序设计

摘自《微计算机信息》月刊,2002 年第 10 期

本文介绍了 32 位单片机 MC68332 SCI 子系统在车用控制器标定系统开发中的实际应用,提出并实现适合单片机与 PC 机数据通信的通信协议,较为系统地阐述了通过单片机异步串行接口(SCI)与 PC 机 RS232 串口通信的软硬件的设计原理与方法。

4.37 基于 VB 的 USB 设备检测通信研究

摘自《计算机应用研究》月刊,2002 年第 11 期

本文通过分析 USB(Universal Serial Bus)的特点及内部结构,研究在 Windows 环境下用 Visual Basic 实现对 USB 设备检测通信方法。在 Windows 接口通信和 USB 设备驱动的基础上,使用 Windows API 函数实现了对 USB 设备的检测和通信。

4.38 USB 设备与 PC 机之间的通信机制的实现技术研究

摘自《微电子学与计算机》月刊,2002 年第 8 期

本文阐述了 USB 设备和 PC 之间的通信机制,并基于 Cypress 公司的 EZ－USB 芯片介绍了为多软驱工作平台系开发的 USB 设备的设计方案,以及 USB 设备和 PC 之间的通信机制在该 USB 设备中的实现技术。

4.39 利用 MODEM 实现单片机与 PC 机远程通信

摘自《仪表技术与传感器》月刊,2002 年第 1 期

本文介绍了由 MODEM(调制解调器)与单片机及 PC 机构成的一种远程通信系统,该系统借助现有的电话网就可以实现单片机与 PC 机之间的远程通信。

4.40 谈谈电力线通信

摘自《现代通信》月刊,2002 年第 4 期

本文介绍了电力线通信的难题所在、电力线通信的优缺点以及当前电力线通信研究的现状。

4.41 低压电力线载波高速数据通信设计

摘自《电测与仪表》月刊,2002 年第 10 期

本文分析了低压电力线载波通信的特点,讨论了低压电力线载波通信的标准。在分析电力线载波高速数据通信技术的基础上,介绍了多载波通信技术 OFDM(正交频分复用);讨论了 Intellon 公司提出的基于 OFDM 原理的 PowerPacket 技术;提出了利用电力线载波通信芯片 INT5130、INT1000 实现高速数据通信的设计方案。

4.42 PL2000 在低压电力线载波通信中的应用

摘自《仪表技术与传感器》月刊,2002 年第 2 期

本文介绍了国产低压电力线载波收/发芯片 PL2000 的基本通信原理,并以 89C2051 单片机为平台设计了一种实用的低压电力线载波通信电路,还结合工程实际给出了一些参考参数。详细介绍了该电路及其控制软件的设计。

4.43 一种电力线扩频载波通信节点的具体实现

摘自《电子技术应用》月刊,2002 年第 1 期

本文介绍了一种以 AT89C52 单片机为控制核心、以 P300/P11 电力线扩频载波通信芯片为基础的电力线载波通信节点的实现,阐述了系统结构及其工作原理,并指出了系统应用中应注意的问题。

4.44 一种基于电力线的家庭以太网络实现方法

摘自《电子技术应用》月刊,2002 年第 10 期

本文提出了一种将 Ethernet 技术嫁接到电力线上,实现基于电力线的家庭局域网络的构思。介绍了实现这种构思的以太网——电力线信号中继方案,为家庭网络的实现提供了新的思路和实现手段。

4.45 基于电力线载波的家庭智能化局域网研究

摘自《工业控制计算机》月刊,2002 年第 9 期

本文分析了电力线这一特殊介质中由调制解调实现信息传送的可能性和意义。在这一基础上,基上这一电力线载波通信技术,利用美国国家半导体公司的 LM1893 电力线调制解调芯片设计出了一个基于这种技术的局域网。

4.46 低压电力线扩频家庭自动化系统

摘自《现代通信》月刊,2002 年第 5 期

本文分析了低压电力线信道特性分析以及系统设计中的载波频率选择、同步的建立、过零抖动处理、高可靠性电力线扩频通信收发器设计。介绍了家庭保安及室内电器远程控制实况。

4.47 智能家庭网络研究与开发

摘自《计算机应用研究》月刊,2002 年第 6 期

本文介绍了智能家庭网络的功能和体系结构,分析了系统中的关键技术,包括嵌入式网络操作系统和通信协议,并且指出了智能家庭网络今后的研究方向。

4.48 蓝牙在家庭网络中的实现

摘自《现代通信》月刊,2002 年第 11 期

本文介绍了蓝牙技术优势,蓝牙在家庭网络中的技术实现原理与方法、家庭网络结构、蓝牙硬件模块、家居服务器模块、信息家电模块、特定通信协议与遥控器模块。

4.49 参照 CEBus 标准的家庭网络系统研究与实现

摘自《计算机工程与设计》月刊,2002 年第 11 期

本文介绍一种参照 CEBus (消费电子总线) 标准中电力线载波规范设计实现的家庭网络通信系统。它主要由集中配置的中心主控设备 (MCU) 和分散配置的数字控制单元 (DCU) 构成,以家庭中的 220V (或 380V) 电力线为传输媒介。MCU 可以实现对 DCU 中连接家电产品的开关调控及状态信息的收集处理,同时 MCU 还作为网关设备提供在 INTERNET、电话网络及无线遥控终端上对家电产品实现远程测控的接口。

4.50 采用蓝牙技术构建智能家庭网络

摘自《电子产品世界 B》月刊,2002 年第 6 期

本文介绍了智能家庭网络特点、家庭无线网络标准、组建蓝牙智能家庭无线网络的系统设计与软硬件设计。

4.51 家庭网络中的设备集成研究

摘自《计算机应用研究》月刊,2002 年第 6 期

随着技术的进步,现在和未来的家庭正充斥着越来越多的设备,家庭网络蓬勃发展。如何高效地利用这些设备成为当今研究的热点。针对此问题讨论了家庭网络中两种设备集成模型:用户-设备控制模型和设备-设备控制模型;分析了两种模型的结构、工作过程,并经创见具体的实例。该模型允许动态地将兼容的设备添加到网络中并自动运行,而且无需安装或更新任何软件。

4.52 一种嵌入式通信协议系统及在智能住宅网络中的应用

摘自《计算机系统应用》月刊,2002 年第 8 期

本文介绍的协议系统通过总线和耦合器,将家用电器组成家庭网络,运用组关联,来定义网络上末端节点复杂多变的逻辑关系,以组报文的形式,将数据在网络末端节点间传输。这种协议体现了智能家庭网络系统结构简单、实时性高、可扩充性好、家电设备间逻辑关系灵活多变等特点,并且在一些真正的家庭系统中得到应用,效果理想。

4.53 基于手机短消息(SMS)的远程无线监控系统的研制

摘自《计算机测量与控制》月刊,2002 年第 8 期

手机短消息是一种崭新的通信方式,它具有开发方便、费用低、免维护、可靠性高等特点。本文介绍了我们开发的一套基于手机短消息的远程无线监控系统,对通信方式、系统结构、工作原理及软件构成进行了详细的说明。该套系统在自动抄表,远程监控等领域具有很好的应用前景。

4.54 基于 GSM 短信息方式的远程自来水厂地下水位自动监控系统

摘自《仪表技术与传感器》月刊,2002 年第 8 期

文中介绍了一种 GSM 短信息方式的远程自来水厂地下水位自动监控系统的基本原理,给出了以 89C52 单片机为核心的液位测试系统的硬件结构组成,并详细介绍了上下位机的软件编程方法。单片机主要用于对自来水厂地下水位的数据采集,PC 计算机主要用于数据显示和数据管理。PC 计算机和单片机通过对 GSM Modem 发送 AT 命令来实现相互间的串行通信。

4.55 TC35 及其在短消息自动抄表系统中的应用

摘自《仪表技术》双月刊,2002 年第 3 期

介绍由 P87LPC767 单片机及 TC35 构成的基于短消息的自动抄表系统。该系统由多路复用的 RS232,RS485 串行接口电路及多路数据采集系统组成,具有结构简单、工作可靠、经济实用等特点。

4.56 计算机不同通信接口下的数据采集技术问题研究

摘自《仪表技术》双月刊,2002 年第 1 期

本文分析 GPIB 并行接口与 RS-232C 串行接口两种重要的计算机通信接口技术,对在这两种不同通信接口下有关的数据采集技术进行了详细讨论。

4.57 80C152 单片机在 HDLC 通信规程中的应用

摘自《电子技术应用》月刊,2002 年第 6 期

本文介绍了 80C152 单片机的工作原理和特点及其在 HDLC 通信规程中的应用,给出了它的编程方法,并对其系统误码率进行了分析。

4.58 内置 MODEM 通信模块在远程监测系统中的应用

摘自《电测与仪表》月刊,2002 年第 9 期

本文介绍了由单片调制解调芯片 MSM7512B、DTMF 芯片 MT8888C 和单片机组成的 MODEM 通信模块在远程监测系统中的应用。文章首先介绍了通信模块的应用背景及与 PSTN 的连接方式,然后对通信模块的硬件、软件实现进行了较为详细的阐述。

4.59 用单片机普通 I/O 口实现多机通信的一种新方法

摘自《现代电子技术》月刊,2002 年第 10 期

本文介绍了一种利用单片机的普通 I/O 实现多机通信的方法,解决了传统的多机通信应用系统中波特率不能任意改变的弊端,解放了主机、各分机的真正串行口,为分机连接不同波特率的串行设备进行通信,以及主单片机用于连接远程 PC 或数字电台提供了有利的条件。

4.60 利用串行通信实现实时状态监控

摘自《微型机与应用》月刊,2002 年第 9 期

本文介绍了工业控制中 PC 机与单片机之间实现实时串行通信的一种有效方法。具体介绍了利用 VB6.0 实现 PC 机对单片机系统的实时状态监控。

4.61 基于 FIFO 芯片的单片机并行通信

摘自《工业控制计算机》月刊,2002 年第 10 期

本文介绍以 FIFO 芯片作为数据暂存器的单片机并行通信方法。文中给出双单片机系统和多单片机系统通信接口设计方法。

五、新器件与新技术

5.1 CYGNAL 的 C8051F02x 系列高速 SoC 单片机

摘自《电测与仪表》月刊,2002 年第 11 期

文中介绍了 C8051F02x 系列单片机的性能、特点、CIP51 内核、存储器、可编程数字 I/O 和交叉开关、串行端口、模/数转换器、数/模转换器、比较器等内部电路单元。

5.2 AduC812 单片机控制系统的开发

摘自《现代电子技术》月刊,2002 年第 2 期

本文对 AD 公司新推出的一种 8 位带有高精度 A/D、D/A 转换器的单片机 AduC812 进行了分析,介绍了 AduC812 优越的性能以及如何用它构造单片机数字控制系统,最后,从软硬件方面讨论了开发 AduC812 单片机系统与 51 单片机系统的相异点。

5.3 可编程外围芯片 PSD5xx 与单片机 68CHC11 的接口

摘自《电子技术》月刊,2002 年第 2 期

文章介绍了可编程外围芯片 PSD5xx 的硬件特性,以及该芯片与单片机 68CHC11 的接口和编程方法。

5.4 模糊单片机 NLX230 及其接口软硬件设计

摘自《测控技术》月刊,2002 年第 1 期

本文介绍了模糊单片机 NLX230 的结构及其特点,描述了 NLX230 的典型接口及其开发软件 ADS230。

5.5 低功耗 MSP430 单片机在 3V 与 5V 混合系统中的逻辑接口技术

摘自《电子技术应用》月刊,2002 年第 10 期

本文介绍了低功耗 MSP430 单片机与传统的 LSTTL、HCMOS 和 CMOS 接口技术,特别阐述了 3V 器件具有 5V 容限的特点,介绍两种电平移位器,双电平移位器 74LVC4245 和简单的电平移位器 74LVC07。

5.6 MSP430F149 单片机在便携式智能仪器中的应用

摘自《微计算机信息》月刊,2002 年第 12 期

本文介绍了 MSP430F149 单片机的一些特性和它的数据采集子系统,并给出了其为基础设计的便携式智能测温仪的例子。

5.7 用 MSP430F149 单片机实现步进电机通用控制器

摘自《电子产品世界 B》月刊,2002 年第 11 期

本文介绍了基于 MSP430F149 单片机实现的步进电机通用控制器。该控制器可同时控制多台步进电机按曲线方式运行,包括加减速、定位及换向功能等。重点探讨步进电机升降速曲

线的设计方案及其实现方法。

5.8 PIC 和 DS18B20 温度传感器的接口设计

摘自《电子产品世界 B》月刊,2002 年第 9 期

本文简要介绍了 PIC 微处理器端口位寻址的方式,并由此讨论直接用 PIC 微处理器 I/O 口远距离驱动 DS18B20 数字温度传感器的驱动能力和编程方法。

5.9 用 P87LPC764 单片机的 I^2C 总线扩展“米”字形 LED 显示器

摘自《半导体技术》月刊,2002 年第 1 期

本文介绍一种利用 philips 公司生产的 p87LPC764 单片机作为 I^2C 总线控制器与 I^2C 总线显示器件 SAA1064 构成的“米”字形 LED 显示器电路,并给出相应的程序清单。

5.10 铁电存储器 FM24C04 原理及应用

摘自《电测与仪表》月刊,2002 年第 9 期

本文简述了铁电存储器的工作机理以及与其他存储器技术相比所独具的优势,重点介绍了 Ramtron 公司串行 FRAM 产品 FM24C04 的硬件结构、主要特性和具体应用。

5.11 CAT24C021 在天文望远镜控制器中的应用

摘自《电子技术》月刊,2002 年第 8 期

CAT24C021 芯片集 EEPROM 存储器,精确复位控制器和看门狗定时器三种功能于一体。本文介绍了 CAT24C021 的特点、功能和工作原理,并详细分析了它在天文望远镜控制器中的应用及具体的编程方法。

5.12 串行时钟芯片在智能传感器中的应用

摘自《现代电子技术》月刊,2002 年第 10 期

本文介绍 DS1337 串行时钟芯片的引脚描述,寄存器组的功能、串行时钟与 MPU 的连接方式、二总线串行接口的读、写工作模式及总线时序,最后给出了大部分实用的子程序。

5.13 RTC 器件 X1228 及其在不间断供电系统中的应用

摘自《电测与仪表》月刊,2002 年第 8 期

X1228 是 Xicor 公司开发的新型 I^2C 总线 RTC 和 CPU 监视器件。本文介绍了其功能、原理、特点及应用,并说明了其内部寄存器的结构与使用方法。最后给出了一个实现单片机系统电源不间断供电的应用实例。

5.14 新型 A/D 转换技术——流水线 ADC

摘自《电子测量技术》双月刊,2002 年第 4 期

流水线结构 ADC 是一种新型的 A/D 转换技术,文中主要介绍流水线结构 ADC 的工作原理,并以 MAX1205 为典型芯片为例介绍其性能,同时还介绍流水线型 ADC 在 CCD 成像技术中的应用。

5.15 集成芯片 AD558 及其应用

摘自《国外电子测量技术》双月刊,2002 年第 3 期

AD558 内部带有能隙参考电压源、高速度输出放大器、可实现模拟电压的单极性或双极性输出。由于内部带有精密的能隙参考基准电压、与微控制器兼容。文中介绍了 AD558 的功能、使用方法和实际应用电路。

5.16 14 位 3MHz 单片模数转换器 AD9243 的应用

摘自《微计算机信息》月刊,2002 年第 3 期

本文对 14 位 3MHz 单片模数转换器 AD9243 的结构、功能特性和工作过程进行了比较详细的介绍,讨论了器件的工作时序及应用问题,并给出了应用实例。

5.17 16 位模数转换器 MAX195 在单片机系统中的应用

摘自《测控技术》月刊,2002 年第 7 期

本文简要介绍了 16 位模数转换器 MAX195 的结构和性能,详细讨论了 MAX195 在单片机系统中的异步转换传输与同步转换传输两种应用,并分别给出了硬件电路和软件程序设计。

5.18 24 位模/数转换器 CS5532 及其应用

摘自《仪表技术与传感器》月刊,2002 年第 7 期

CS5532 是一种低噪声 24 位 $\Delta-\Sigma$ 型 A/D 转换器。文中详细阐述了 CS5532 的结构、组成、功能特点及工作方式,并以高精度称重仪——渗碳液体流量监测仪为例,论述了其在高精度测量方面的具体应用,给出了其与单片机接口的电路原理图和驱动软件程序。

5.19 ADS7825 模数转换芯片及其在高速数据采集系统中的应用

摘自《计算机测量与控制》月刊,2002 年第 6 期

ADS7825 是 BB 公司生产的高性能模数转换器件,它具有 4 路模拟输入通道,5V 单电源供电,16 位并行输出等独特性能,文章详细介绍了 ADS7825 的内部结构、工作原理、性能指标以及它与 TMS320C6201 组成的多路数据采集前端的具体应用。

5.20 新型 D/A 变换器 AD9755 及其应用

摘自《现代电子技术》月刊,2002 年第 11 期

AD9755 是 Analog Device 公司生产的一种 14 位高速数模转换芯片,其特有的双端数据复用结构,使之获得了优越的 DAC 性能。本文在介绍了 AD9755 的工作原理基础上,阐述了其在任意波形产生器中的设计应用。

5.21 单片机与串口 D/A 转换器 MAX525 的接口设计

摘自《现代电子技术》月刊,2002 年第 12 期

本文简要介绍了 MAXIM 公司生产的串口 4 通道 D/A 转换器 MAX525 的基本原理及功能特点,给出了 MAX525 与单片机 AT89C52 接口的硬件电路及软件编程实例。

5.22 几种 PWN 控制器

摘自《电子测量技术》双月刊,2002 年第 1 期

本文介绍了采用 MAX668/MAX669 构成的升压型、SEPIC、反激型和隔离型等几种结构的 PWM 控制器。

5.23 一种新型的可编程的 4~20mA 二线制变送器 XTR108 及其应用

摘自《仪表技术与传感器》月刊,2002 年第 10 期

XTR108 是一个完整的可编程的 4~20mA 两线制变送器芯片,允许用户数字化调节增益、偏移及线性化校正系数等。用户通过主机上的软件界面,经基于单片机组成的标准串行数字接口部件,即可对模拟信号路径进行数字校零校满,对传感器的非线性误差可方便地进行数字化补偿。文中简要介绍了 XTR108 芯片的特性、内部结构及其工作原理,对 XTR108 在铂电阻温度并送器的应用及数字校正方法进行了介绍,并给出了软件设计流程图及用软件模拟 SPI 总线时序对 XTR108 执行写操作的程序。

5.24 可编程温度监控器 ADT14 及其应用

摘自《电子技术》月刊,2002 年第 10 期

ADT14 是一种可编程温度监控器电路,可广泛应用于各种嵌入控制系统中进行温度检测和控制,文中介绍了 ADT14 的特点和用 ADT14 设计的新型双温电子温控器。

5.25 一种适用于 51 系列单片机的 R/F 转换电路

摘自《自动化仪表》月刊,2002 年第 7 期

本文介绍利用 AT89C2051/51 两个片内可编程 16 位定时器/计数器,实现 12 位以上分辨率的 A/D 转换效果,并且转换精度不受 R/F 转换时的振荡电容变化的影响的 R/F 转换电路,文中给出了 RF 转换电脑及程序清单。

5.26 通用集成滤波器的特点及应用

摘自《仪表技术》双月刊,2002 年第 2 期

根据集成有源滤波器的特点,本文介绍几款由 UAF42 通用有源滤波器构成不同滤波形式的集成滤波器的应用电路,文中还介绍了 MAX274 实时连续有源滤波器的特点,结构与典型应用。

5.27 串行显示驱动器 PS7219 及单片机的 SPI 接口设计

摘自《自动化与仪器仪表》双月刊,2002 年第 2 期

PS7219 是 SPI 串行接口 LED 显示驱动器。本文介绍该芯片的引脚、内容结构、功能、以及工作过程,并分析了它与单片机的 SPI 接口设计方法,给出了用 89C51 与 PS7219 的应用实例。

5.28 新型的键盘显示芯片——SK5279A 的应用

摘自《仪表技术与传感器》月刊,2002 年第 7 期

本文介绍一种新型的键盘显示控制芯片——SK5279A。SK5279A 具有串行接口,可同时驱动 8 位共阴式数码管或 64 只独立的 LED),内部带有译码器,具备两种译码方式,可直接接收 16 进制码,通过软件控制可选择两种译码方式的一种或不译码。文中介绍了它的结构、特点、电路原理、控制命令及软硬件设计。

5.29 高效语音压缩芯片 AMBE—2000™及其在语音压缩中的应用

摘自《现代电子技术》月刊,2002 年第 3 期

本文介绍了 Digital Voice Systems, Inc 开发的高效语音压缩芯片 AMBE—2000™的特点及其应用,详细阐述了 AMBE—2000™的通信格式及其应用。

5.30 适于语音处理的 SDA80D51 芯片及其数字录放音系统

摘自《电子技术应用》月刊,2002 年第 9 期

SDA80D51 UNISPEECH 是 Infineon 公司新推出的具有 DSP 和单片机双核的芯片。本文介绍了该芯片的组成框图及各功能模块,并且用该芯片完成一个硬件系统,实现了语音的 G.723.1 编码存储和解码放音。

5.31 基于 ISD2560 语音芯片的小型实用语音系统

摘自《国外电子测量技术》双月刊,2002 年第 6 期

本文介绍了一款基于 ISD2560 语音芯片的小型实用语音系统,给出了最简录放模式电路、显示电路及放音总电路图,并提供了 ISD 系列芯片级联的另一种方式。

5.32 发射信号处理器 AD6622 在软件无线电中的应用

摘自《电子产品世界 B》月刊,2002 年第 4 期

数字中频技术在传统模拟器件的基础上提供了更多优势,AD6622 与 AD9754 (DAC)、AD6645 或 AD6644 (ADC)、AD6624 (四通道接收信号处理器) 或 AD6620 (单通道接收信号处理器) 构成的一个完整平台,本文介绍了 AD6622 的主要功能、特点、结构和它在软件无线电中的应用设计,并给出了部分具体的设计过程。

5.33 基于 UM3758－108A 芯片远距多路参数监测系统

摘自《电子测量技术》双月刊,2002 年第 2 期

本文中简要介绍了 UM3758－108A 芯片的功能和引脚含义,给出并分析了一个由 UM3758－108A 和 A/D 转换器构成的十路模拟参数远距离监测系统。

5.34 单片频率计 ICM7216D 及应用

摘自《国外电子测量技术》双月刊,2002 年第 3 期

ICM7216D 是美国 Intersil 公司首先研制的专用测频大规模集成芯片。本文介绍了该芯片的结构、特点、工作原理及其在电机转速测量中的应用。

5.35 X25045 芯片在微机测控系统中的应用

摘自《电气时代》月刊,2002 年第 8 期

在工业环境下,微机测控系统常常受到干扰,其中大型设备的启停、强继电器的通断、电源波形畸变等因素会造成电源电压的波动,瞬间的压降往往造成系统死机、数据丢失和误操作,使系统无法正常运行,甚至出现事故,所以对系统电源电压的监测、控制和重要数据的有效保存十分重要。

5.36 MC14562B 在多 CPU 系统串行通信中的应用

摘自《微型机与应用》月刊,2002 年第 6 期

本文介绍如何利用 MC14562B 移位寄存器实现 CPU 间的双向串行通信方法,介绍 MC14562B 的结构原理、硬件应用接口及其软件编程的实现。

5.37 高级串行通信控制器 SAB82525 及其应用

摘自《电子技术》月刊,2002 年第 3 期

本文介绍了串行通信控制器 SAB82525 的结构原理、工作模式及进程,并给出了基于 SAB82525 的应用系统设计。在该系统中,采用 4 片 SAB82525 构成了两种总线型通信链路,实现了灵活、可靠的串行数据通信。

5.38 MAX121 芯片在高速串行接口电路中的应用

摘自《现代电子技术》月刊,2002 年第 4 期

MAX121 芯片是一个带串行接口的 14 位模数转换集成电路(ADC),它包含有跟踪/保持电路的一个底飘溢、底噪声、掩理式齐纳电压基准电源。本文介绍它的工作性能与特点,具体给出了 MAX121 芯片在数字信号处理集成电路(TMS320)高速串行接口电路的具体应用设计。

5.39 应用 DS2480 实现 RS-232 与单总线的串行接口

摘自《微计算机信息》月刊,2002 年第 12 期

在基于单总线技术的智能巡检系统中,应用串行接口芯片 DS2480 可较好地解决具有 RS-232 串口的主机与现场 1-Wire 器件的通信问题。本文较为详细地介绍了 DS2480 芯片及其在智能巡检系统现场检测模块中的应用。

5.40 介绍一种真正的单芯片 MODEM-73M2901C/5V

摘自《微计算机信息》月刊,2002 年第 5 期

73M2901C/5V 是一种全双工,FSK 的单芯片调制解调芯片,它具有 DTE 控制所需的所有信号和实现智能 V.22bis 数据调制解调的数据泵功能。它是用于嵌入式系统真正的单芯片 MODEM 解决方案。本文介绍了 73M2901/5V 芯片的基本功能、特点及应用。

5.41 HART 调制解调器 SYM20C15 应用设计

摘自《单片机与嵌入式系统应用》月刊,2002 年第 11 期

本文介绍 HART 调制解调器芯片 SYM20C15 的工作原理及其电路模块,给出其典型应用电路。该芯片专为实现 HART 协议而设计,与微处理器的接口简单、功耗低,被大量应用于 HART 智能仪表。

5.42 TM1300 同步串行接口与 Modem 模拟前端之间的通信

摘自《计算机系统应用》月刊,2002 年第 9 期

本文介绍了 Philip 公司的 Trimedia 系列多媒体 DSP 芯片 TM1300 的同步串行接口与 Modem 模拟前端 STLC7545 的结构、功能及特性,分析了两者之间的通信过程,并且介绍了通过 API 函数来实现两者之间通信。

5.43 TEMIC 系列射频卡及其应用

摘自《电测与仪表》月刊,2002 年第 1 期

本文介绍了 TEMIC 公司射频卡系列的特点、工作原理和使用方法,并就实际的使用经验,对软硬件的一些问题做了简要的叙述。

5.44 用 Philips PCD600x 实现多线电话并机

摘自《单片机与嵌入式系统应用》月刊,2002 年第 12 期

飞利浦公司开发的 PCD600x 数字电话应答机芯片内嵌 80C51 CPU 核、DSP 及 CODEC 等通信功能芯片,具有功能强、成本低等特点。本文拓展了 PCD600x 的应用,特别是重点解决了多线电话互连的问题 通过 IOM 数字通道提供了低成本、高可靠性的解决方案。

5.45 SDH 专用集成电路套片 DTT1C08A 和 DTT1C20A 及其应用

摘自《电子技术应用》月刊,2002 年第 12 期

DTT1C08A 和 DTT1C20A 是大唐电信微电子公司自主开发的用于光同步数字传送网的专用集成电路套片。介绍了该套片的特点、功能框图及其在台式 SDH155Mbit/s 光传输设备中的应用。DTT1C08A 和 DTT1C20A 专用集成电路具有集成度高、使用方便等特点,并且成本低廉,具有良好的应用前景。

5.46 GAL16V8 用于步进电动机驱动器

摘自《电气时代》月刊,2002 年第 3 期

本文介绍了 GAL16V8 的简单结构,介绍了利用开发工具软件 Palasm4 的开发方法,并给出了 GAL16V8 作为步进电机驱动器的环形分配器的程序。

5.47 UC3717 步进电机驱动电路与 89C2051 单片机的接口技术

摘自《电气自动化》双月刊,2002 年第 5 期

本文介绍用 UC3717 驱动二相步进电机的应用电路,它的整步,半步和 1/4 三种步距的工作模式及控制方法,与 89C2051 单片机的接口电路,控制程序。

5.48 TinySwitch 单片开关电源的设计方法

摘自《微计算机信息》月刊,2002 年第 3 期

本文介绍了 TinySwitch 单片开关电源的一种设计方法与具体步骤,该方法具有简单易行、经济高效的优点,对于电源设计者具有指导作用。

5.49 基于 MAX883 的动态供电设计

摘自《电测与仪表》月刊,2002 年第 9 期

介绍一种仪器动态供电的设计方法、供电控制应用及动态供电系统。该系统采用电源管理 IC MAX883,结合单片机控制,使得整机设计具有功能强、体积小、省电、可靠性高的特点。

5.50 高压 PWM 电源控制器 MAX5003 及其应用

摘自《仪表技术》双月刊,2002 年第 2 期

MAX5003 广泛应用于通信设备、42V 汽车系统、综合服务数字网络接口及工业电源设备中。本文介绍 MAX5003 的工作特征和引脚功能,给出了典型应用电路和参数设计方法,并给出具体设计步骤。

5.51 单片机与大功率负载的开关接口

摘自《仪表技术》双月刊,2002 年第 2 期

MOTOROLA 公司制造的 MOC3061 系列过零触发光电隔离双向晶闸管驱动器,不仅可消除或大大减小大功率双向晶闸管导通时对电网的影响。本文介绍了 MOC3061 的结构、工作原理和它与单片机的接口电路。

5.52 迟滞开关功率转换器 LM3485 在电源系统中的应用

摘自《电子元件与材料》月刊,2002 年第 10 期

电流控制型开关变换器由于需要双环控制,增加了电路设计和分析的难度,控制环变得不稳定以及抗干扰能力差。本文介绍的迟滞开关功率转换器 LM3485,旨在实现一种能够满足大量不同降压要求而无须进行补偿的开关调整器解决方案。迟滞 PFET 降压控制器的引入,提高了稳定性和抗干扰能力。

5.53 功率逻辑器件在嵌入式系统中的应用

摘自《电子产品世界 B》月刊,2002 年第 5 期

本文简要介绍了嵌入式可编程控制器及其 I/O 模块,结合功率逻辑器 TPIC6B273 的特点,对传统的输出模块进行了改进,实验证明其简捷有效。

5.54 TPS60101 用于低功耗系统的电源解决方案

摘自《单片机与嵌入式系统应用》月刊,2002 年第 10 期

本文介绍一种新颖的电荷泵直流稳压芯片 TPS60101 的性能特点和使用方式,结合实例分析其在低功耗单片机系统中的应用。

5.55 新型电能表芯片 AT73C550 及其应用

摘自《电测与仪表》月刊,2002 年第 2 期

本文介绍了美国 Atmel 公司推出的芯片 AT73C550 的原理、特点、主要功能及典型应用。该芯片功能强大,只需少量分立元件就能构成电子式电能表,具有结构简单、准确和性价比高

等诸多优点,在频率为 50Hz/60Hz 时,符合 IEC60521/60687/601036 工业设计标准。

5.56 运动控制芯片 MCX314 及其应用

摘自《电子技术》月刊,2002 年第 10 期

MCX314 是台湾 Plenty Island 公司生产的能够同时控制四个轴的可编程运动控制芯片,主要用于步进电机或伺服电机驱动控制。文中介绍了运动控制芯片 MCX314 的内部结构、主要特征及控制使用方法。并简要介绍了使用 MCX314 设计的一种基于 ISA 总线的运动控制卡。

六、总线技术

6.1 PCI－to－PCI 桥及其应用设计

摘自《电子技术》月刊,2002 年第 3 期

从电气负载和系统性能考虑,一条 PCI 总线不能驱动过多的 PCI 设备。PCI 系统扩展必须使用 PCI 桥设备。本文介绍了 PCI－to－PCI 桥的概念、原理、结构及 PCI 总线扩展方法,简要介绍了 PCI－to－PCI 透明桥 Intel21154 的特点及应用设计注意事项。

6.2 基于 PCI 总线的数据采集系统

摘自《计算机系统应用》月刊,2002 年第 12 期

本文提出了一种基于 PCI 总线的可变速率数据采集系统的结构以及该系统的实现。并给出了即插即用可变速率数据采集卡的 WDM 驱动程序设计方法,以及构建在采集卡和 WDM 驱动程序基础之上的 Win32 应用程序开发实例。

6.3 VXI 和 PXI 总线技术的应用及其发展前景

摘自《工业控制计算机》月刊,2002 年第 11 期

本文主要介绍了 VXI 和 PXI 总线技术在各个领域中的应用情况。讨论了两种总线各自的特点,并对 VXI 和 PXI 总线技术的发展前景进行了分析。

6.4 基于 PC104 总线的嵌入式以太网卡设计

摘自《单片机与嵌入式系统应用》月刊,2002 年第 8 期

针对近年来工业现场设备愈来愈多的上网需求,文中提出一种利用 Intel 嵌入式微处理器 386EX 设计智能以太网的技术方法。通过对系统硬件及软件的设计描述,说明利用此方案设计的智能以太扩展模块能够实现基于 PC104 总线的工业设备的快速上网需求。

6.5 基于 RS－485 总线的传感器网络化技术研究

摘自《传感器技术》月刊,2002 年第 8 期

传感器网络化是传感器领域发展的一项新兴技术。它在统一网络协议的基础上实现传感器信号的数字化、网络化的变送。本文介绍了一种基于 RS－485 总线的网络传感器的设计与组网方法,并给出了软硬件设计实例。

6.6 RS－232 总线转 CAN 总线装置的设计与实现

摘自《工业控制计算机》月刊,2002 年第 1 期

针对 RS－232 总线传输距离在 1km 以内,同时其最高速率不超过 20kb/s,本文提出用 CAN bus 来解决这个问题,同时给出了其硬件和软件设计方案。

6.7 现场总线技术的发展与工业以太网综述

摘自《工业控制计算机》月刊,2002 年第 1 期

本文分析了现场总线技术的现状以及发展方向,并就工业以太网作为今后现场总线技术标准的可行性进行了详细的论述。同时对 TCP/IP 与以太网相结合的问题进行了阐述,并提出了工业实时 Ethernet 即 Ethernet/IP 的概念以及应用前景。

6.8 广义现场总线标准与工业以太网

摘自《电气自动化》双月刊,2002 年第 4 期

本文较全面地论述了现场总线及不同工业领域的低层串行数字通信网络的国际标准,包括 8 类总线的 IEC 61158 标准,以及用于公路车辆信息交换的 ISO 11898 标准,列车通信网标准 IEC 61375,用于低压配电与控制装置的控制器与设备接口标准 IEC 62026,用于工业机械(机床)运动控制的实时通信标准 IEC 61491(SERCOS)。介绍了我国制订现场总线标准的动向。论述了高速以太网用作现场总线的发展趋势。

6.9 用单片机设计现场总线转换网桥

摘自《单片机与嵌入式系统应用》月刊,2002 年第 3 期

文中讨论工业控制系统设备连接转换网桥的基本概念和工业控制系统对网桥的基本要求。在此基础上,提出转换网桥和网络转换概念,并给出工业控制系统和现场总线技术对网桥和网络转换的基本要求。通过讨论,提出用 MC68HC05C8 设计的三种总线系统连接类型,并针对这三种连接类型,提出网桥模块和网络转换的基本结构。

6.10 基于 LonWorks 的在系统编程技术

摘自《单片机与嵌入式系统应用》月刊,2002 年第 10 期

LonWorks 技术的应用使得在系统编程的内涵得以更充分的体现。本文在概要介绍 ISP 以及 LonWorks 技术的基础上,详细说明采用基于 Neuron 芯片的控制节点实现对 CPLD 进行在系统编程的具体方法。

6.11 Neuron 芯片与 MCS-51 系列单片机串行通信的实现

摘自《仪表技术》双月刊,2002 年第 6 期

本文介绍了一种基于 MCS-51 系列单片机的 LON 节点,它利用 MCS-51 系列单片机的串行接口实现与 Neuron 芯片的数据交换。文中给出了具体的软硬件设计方案,并指出了其应用前景。

6.12 Neuron 芯片多总线 I/O 对象的应用

摘自《仪表技术》双月刊,2002 年第 5 期

本文介绍 Neuron 芯片的特性、TMPN3150 基本系统和应用电路,给出了多总线 I/O 对象在分体空调集散控制中的应用实例。

6.13 CAN 总线及其应用技术

摘自《微计算机信息》月刊,2002 年第 9 期

本文在总结 CAN 总线特点的基础上,对其通信介质访问方式进行了详细的描述,介绍了它在应用中需要解决的技术问题以及目前应用状况。

6.14 CAN 总线协议分析

摘自《中国仪器仪表》双月刊,2002 年第 4 期

CAN 是具有很高的安全水平、能有效支持分布式实时控制的一种串行通信协议。本文介绍了现场总线的发展现状,对 CAN 场总线协议进行了详细的描述,重点介绍信息格式和数据链路层。此外,还介绍了 CAN 控制器验收滤波器的原理。

6.15 CAN 总线智能节点的设计和实现

摘自《工业仪表与自动化装置》双月刊,2002 年第 5 期

本文介绍了 CAN 总线的基本特点和下位机智能节点的设计实现,智能节点由 8031、SJA1000 和 82C250 组成。文中给出了软件编制的方法。

6.16 CAN 总线控制器 SJA1000 的原理及应用

摘自《电测与仪表》月刊,2002 年第 4 期

本文介绍了一种新型的 CAN 总线协议控制器芯片 SJA1000 及应用芯片设计仪器仪表的智能化通信技术。

6.17 CAN 总线与 PC 机通信卡接口电路设计

摘自《微型机与应用》月刊,2002 年第 9 期

文中基于对现代轿车车身控制器局域网(CAN)及主要控制装置的研究开发,介绍了 CAN 总线和 PC 机接口卡的 3 种电路设计方案,即双口 RAM 通信方案、8255 通信方案和利用 ISA 总线直接进行 I/O 口读写的方案,并提出了在应用设计中应注意的问题。

6.18 CAN 总线及其在测控系统中的实现

摘自《中国仪器仪表》双月刊,2002 年第 1 期

本文主要介绍了 CAN 总线的特点、应用范围、通信协议和 CAN 总线系统的一种实现方法及 CAN 总线系统在测控网络中的典型结构。

6.19 基于 CAN 总线的温度、压力控制系统

摘自《仪表技术与传感器》月刊,2002 年第 10 期

本文作者设计了一种基于 CAN 总线的温度、压力测控系统,详细说明了系统的总体设计和各部分的结构原理,并对应用程序的设计进行了介绍。本系统功能灵活,可靠性高,可用于需要对现场多个部位的温度压力进行综合控制的场合。

6.20 基于 CAN 总线的新型网络数控系统

摘自《电气时代》月刊,2002 年第 10 期

本文介绍了基于 CAN 总线的网络数控系统,介绍了系统结构、组成、特点及新型网络数控系统的实现方法。

6.21 CAN 总线在混和动力汽车电机控制系统中的应用

摘自《电子技术应用》月刊,2002 年第 3 期

本文介绍了 CAN 总线的主要功能与特点、CAN 控制器以及 CAN 总线在混和动力汽车电机控制系统中的应用;CAN 总线与电机控制芯片 TMS320F241 的接口设计、帧结构以及通信中断服务程序流程图等。

6.22 CAN 总线技术在石油钻井监控系统中的应用

摘自《电子技术》月刊,2002 年第 6 期

本文介绍了基于 CAN 总线的分布式通信网络在石油钻井监控系统中的应用,对设计方案、硬件原理图和用 C 语言实现的通信程序模块作了详尽讨论。并根据系统的实际运行情况,对需要注意的技术细节作了针对性的介绍。

6.23 一种电动阀的 DeviceNet 总线接口设计

摘自《仪表技术》双月刊,2002 年第 1 期

本文介绍了一种智能阀的 DeviceNet 现场总线接口设计方案,给出了系统框图、硬件电路及部分程序设计方法。

6.24 单总线技术及其应用

摘自《仪表技术与传感器》月刊,2002 年第 1 期

在测量系统中,传感器通过导线与计算机接口连接,在传感器数量较多的情况下,系统中的接线会非常复杂,尤其是长距离的信号传输容易相互产生干扰。本文介绍了只用 1 根信号线使计算机与多个被测对象进行信息传输的技术称为单总线技术 (1 – Wire Bus),目前单总线技术在测控领域应用很广,结合单总线器件具体介绍单总线技术及其应用。

6.25 美国 DALLAS 公司单线可编程数字温度传感器技术

摘自《电工技术》月刊,2002 年第 4 期

DS1821 是单线可编程数字温度传感器,本文介绍了 DS1821 的特点、工作原理、单线模式与自动控温模式、模式转换、编程命令及典型应用。

6.26 基于单总线技术的农业温室控制系统设计

摘自《微型机与应用》月刊,2002 年第 3 期

本文介绍了一种利用 CAN 总线网络进行数据通信并基于单总线技术实现单片机农业温室控制系统,详细分析了其软硬件的设计。

6.27 单总线协议转换器在分布式测控系统中的应用

摘自《单片机与嵌入式系统应用》月刊,2002 年第 11 期

本文介绍了 DS2480B UART/RS232 至单总线协议转换器的主要特性、工作原理、接口技术,并具体阐述 DS2480B 在农业温室分布式测控系统设计中的应用。

6.28 单总线技术在电子信息识别系统中的应用

摘自《电子技术》月刊,2002 年第 9 期

单总线是 Dallas 半导体公司推出的总线标准。本文介绍了利用单总线技术特点及单总线系统与单总线器件之间的独特连接方式,构成信息识别系统,实现了移动性电子信息识别,为仓库管理、安全巡检、保密信息验证识别等工作提供解决方案。

6.29 信息纽扣及其在安全巡检管理系统中的应用

摘自《微型机与应用》月刊,2002 年第 7 期

本文介绍了信息钮扣的概念和信息钮扣的单总线串行传输协议以及基于信息钮扣技术的安全巡检管理系统的总体规划和软硬件设计的思路。

6.30 SPI 串行总线接口及其实现

摘自《自动化与仪器仪表》双月刊,2002 年第 6 期

本文以 ADμC812 单片机为例,详细介绍了 SPI 总线接口的逻辑时序,并运用软件模拟 SPI 总线时序的方法,实现 MCS51 系列单片机同 ADμC812 单片机的 SPI 总线接口通信。

6.31 通用串行总线 USB 及其产品开发

摘自《现代电子工程》季刊,2002 年第 2 期

本文论述了通用串行总线 USB 的特点及其体系结构,并对 USB 产品的开发方法做了介绍。

6.32 通用串行总线 (USB) 数据传输模型

摘自《计算机工程与设计》月刊,2002 年第 10 期

USB 作为一种数据传输接口,了解它的通信传输机制是正确利用 USB 进行数据传输的基础。本文介绍了分层次结构对 USB 数据传输模型加以分析,可以使读者建立起 USB 体系结构、通信模型和总线事务的框架和概念,并最终对 USB 传输工作流程有一个清晰的认识。

6.33 基于 USB 总线的测试系统开发

摘自《电测与仪表》月刊,2002 年第 5 期

本文在分析 USB 通用串行总线基本结构、协议和工作原理的基础上,讨论了 USB 总线测试系统中 USB 设备接口及 USB 主机有关的硬件和软件开发技术。

6.34 一种 USB 外设的实现方法

摘自《计算机工程》月刊,2002 年第 7 期

PDIUSBD12 芯片能够实现微控制器的并行总线到 USB 总线的接口功能,使用 PDIUSBD12 芯片可以实现我种类型的 USB 外设。本文介绍了使用该芯片实现 USB 外设的方法。

6.35 基于 USB 接口的 PTP 协议在 Win32 上编程实现

摘自《工业控制计算机》月刊,2002 年第 12 期

本文简单介绍了 PTP (Picture Transfer Protocol) 协议的背景和内容,并从协议的工作过程出发,在 PC 上实现其在 USB (USB1.1) 接口上的编程,进而在此基础上成功的完成了对数码相机的控制。

6.36 USB 在便携式外设间的应用及其协议

摘自《微型机与应用》月刊,2002 年第 9 期

本文介绍了 USB2.0 的补充规定 USB OTG 规范的技术特点,并对它便携式外设间通过 USB 接口进行通信的协议作了初步的研究。

6.37 多 USB 接口的局域网接入技术的实现

摘自《电子技术应用》月刊,2002 年第 10 期

本文提出了一种全新的计算机接入局域网的方案,使多台计算机可以方便地使用各自的 USB 接口接入局域网,并提供了该方案的实现方法。

6.38 USB 接口设计及其在工业控制中的应用

摘自《制造业自动化》月刊,2002 年第 11 期

文中首先介绍了 USB 以及设计 USB 外设的两种方案,即单片机加 USB 接口器件和带 USB 接口单片机。然后通过数控机床手持控制器作为实例说明 USB 外设的软硬件组成,最后对 USB 在工业控制中的应用进行了展望。

6.39 USB 技术在第四代数控测井系统中应用

摘自《电子测量技术》双月刊,2002 年第 5 期

介绍 USB 技术在第四代数控测井系统中的应用,给出 USB 应用系统的使用框图、单片机 W77E58 和 USB 控制器 USBN9603/4 硬件电路的设计。

6.40 用 AN2131Q 开发 USB 接口设备

摘自《电子产品世界 B》月刊,2002 年第 10 期

本文详细介绍了用 AN2131Q 开发计算机外围设备的全过程,它与计算机之间采用 USB 接口进行数据通信,真正实现了即插即用功能,设备开发和维护非常方便。

6.41 USB/IrDA 桥控制芯片 STIr4200S

摘自《单片机与嵌入式系统应用》月刊,2002 年第 3 期

本文从功能方块图、引脚说明、帧格式、寄存器及其控制指令等几个方面详细介绍 SigmaTel 公司开发的一款专用于 USB 与 IrDA 间桥接控制的新型 ASIC "STIr4200S。

6.42 一种基于 USB 接口的家庭网络适配器的设计

摘自《电子技术》月刊,2002 年第 10 期

本文首先介绍了 HomePNA 技术的发展,指出其技术特点,然后介绍了 HomePNA 家庭网络的系统构成,最后详细阐明了一种采用 USB 接口的符合 HomePNA1.0 规范的 USB1M HomePNA 10M Ethernet 适配器设计方案。

6.43 基于 USB 总线的实时数据采集系统设计

摘自《电子技术应用》月刊,2002 年第 2 期

本文介绍的基于通用串行总线 (USB) 的实时数据采集系统的设计严格遵循 USB1.1 协议,充分体现 USB 便捷、易扩展、低成本、低干扰的特点。文中详细介绍系统的 USB 设备驱动程序、设备固件、应用程序的具体设计。

6.44 基于 SL11R 的 USB 接口数据采集系统

摘自《电测与仪表》月刊,2002 年第 11 期

SL11R 是一种带有 USB 接口的 16 位微处理器,本文介绍了基于 SL11R 扩展动态存储器及 14 位 A/D 转换器 AD9243 组成 USB 接口数据采集系统的一个具体实例。

6.45 基于 USB 的数据采集系统设计与实现

摘自《工业控制计算机》月刊,2002 年第 10 期

本文介绍了一种采样频率为 125 kHz、采样精度 16 位以及带 32K 缓存的四通道 PC 机用数据采集卡,该卡是 USB 接口 DMA 技术和高速电路技术结合应用的新成果,适用于对中高速瞬态信号进行数字化处理。

6.46 USB2.0 在高速数采系统中应用

摘自《电子测量技术》双月刊,2002 年第 2 期

本文分析传统的基于 PC 的数采系统的优缺点,提出基于 USB2.0 的高速外置式数采系统的解决方案,以业界第一款 USB2.0 微控制器– EZUSB FX2 为核心,介绍系统硬件设计、数据传输方式的选择以及固件的开发。

6.47 基于 USB 的航空检测数据采集系统的设计

摘自《计算机应用研究》月刊,2002 年第 11 期

通用串行总线 (USB) 作为一种崭新的微机总线接口规范,可以实现较传统方式更有效、更经济、更多点的数据采集。本文详细介绍了基于 USB 总线的航空检测数据采集系统的开发方法。

6.48 基于 USB 总线的小型图像采集系统的设计

摘自《微型机与应用》月刊,2002 年第 12 期

本文介绍了一种基于 DSP 控制,用 USB 接口来实现数据传输的高速图像数据采集系统的设计。重点介绍了 DSP 芯片在进行常规图像处理的同时,对 USB 接口芯片的控制。

6.49 USB 技术及其在图像数据传输中的应用

摘自《计算机应用研究》月刊,2002 年第 2 期

本文介绍了通用串行总线 USB 的结构、特性及原理,并对 Cypress 公司的 EZ－USB 开发系统作了简要的介绍。利用 EZ－USB 开发板 AN2131Q 和配套软件,在硬件以及软件方面进行设计开发,实现了通过 USB 接口进行图像数据的传输。此外,还对数据传输系统的工作原理作了简要说明。

6.50 USB2.0 在遥感图像采集中的应用

摘自《电子技术》月刊,2002 年第 1 期

USB2.0 协议中,数据传输速率高达 480Mbit/s,可以满足遥感图像数据采集极高的带宽要求,与传统的数据采集方式相比更高效、更可靠、更灵活、更经济。文章介绍了如何利用 USB2.0 接口器件 Cy7C68013 来实现遥感图像数据的采集。

6.51 CCD 摄像机的 USB 接口设计

摘自《传感器技术》月刊,2002 年第 12 期

本文对通用串行总线 USB 的技术特点和数据传输机制进行了较详细的分析,并讨论了具有 USB 接口的数字摄像系统的实现方法。整个系统以总线控制器的单片机 8X930USB 为核心,通过 I^2C 总线对摄像机进行控制。

6.52 带 USB 接口的发动机点火波形测量系统

摘自《单片机与嵌入式系统应用》月刊,2002 年第 7 期

本文介绍了 USB 接口技术的特点,给出利用单片机 Intel 8x930Ax 和 PC 机通过 USB 接口实现对汽车发动机点火波形的测量系统。

6.53 USB 接口智能传感器标定数据采集系统的设计

摘自《计算机应用》月刊,2002 年第 12 期

传统智能传感器标定系统数据采集通常采用专用的板卡,成本高,采集点数十分有限。通用串行总线 USB 为传感器标定系统的多点数据采集和传感器的网络化标定提供了很大的便利。利用 USB 可以实现较传统方式更有效、更方便、更经济的多传感器标定系统数据采集。本文介绍了利用 USB 接口来实现智能传感器网络化标定系统的设计方案。

6.54 USB 接口在粮仓自动测温系统中的应用

摘自《电子技术》月刊,2002 年第 11 期

本文结合粮仓自动测温系统的应用,介绍了 USB 接口的原理和特点。在分析测温系统电路的基础上,着重介绍了 USB 接口的设计方法。USB 接口测温部分由 USB 接口器件 Cy7Cb3101 和 ADC0804 组成。

6.55 基于 GPIF 的 USB－ATA 解决方案

摘自《电子技术应用》月刊,2002 年第 7 期

本文提出利用 EZ－USB FX 及 FX2 系列中 GPIF 的功能,提出了一套解决 USB－ATA 的可行性方案,并给出了单片机控制硬盘时 PIO 模式和 UDMA 模式的实现方法。此方案具有速度快、性能高、占用 CPU 资源少等特点。

6.56 基于 USB 总线新型视频监视和会议系统

摘自《电子测量技术》月刊,2002 年第 4 期

本文介绍了一种把 USB 总线应用于视频监视和会议系统中的方案。而且由于采用了 MPEG－2 编解码芯片进行实时的压缩,使得图像的质量可以达到 DVD 级,而占用的空间却不大。本系统中还可以根据实际需要选择不同的压缩格式和参数,有广阔的应用前景。

6.57 基于 USB 接口的高性能虚拟示波器

摘自《电子产品世界 B》月刊,2002 年第 12 期

本文介绍了一种基于 USB 总线的高性能虚拟示波器的系统结构、软硬件工作原理及其主要特点。

6.58 IEEE 1394 与现场总线

摘自《工业仪表与自动化装置》双月刊,2002 年第 5 期

为解决 PC 领域多媒体通信问题而推出的 IEEE 1394 (又称为火线) 具有即插即用、速度高、成本低廉、易于使用等良好性能,有可能成为工业现场的新型总线标准。该文从协议框架的角度探讨了火线用作现场总线的可行性,并阐述了火线作为现场总线的不足之处。

6.59 IEEE 1394 高速串行总线及其应用

摘自《计算机工程》月刊,2002 年第 11 期

本文介绍了 IEEE 1394 高速串行总线的历史、现状及发展趋势,并通过对其物理层、链路层及传输层的结构的剖析,结合 TI 公司的低功耗 IEEE 1394 解决方案对其应用作了进一步的探讨。

6.60 EF4442 及其应用

摘自《仪表技术》双月刊,2002 年第 3 期

EF4442 是 THOMSON 公司开发的 ARINC 429 总线接口芯片,外接驱动芯片 HS－3182 便可产生 ARINC429 电平,本文介绍了 EF4442 芯片的结构、工作方式,以及实现 ARINC429 总线数据传输的硬件电路。

七、可靠性及安全性技术

7.1 单片机系统可靠掉电保护的实现

摘自《测控技术》月刊,2002 年第 1 期

本文以 MCS－51 系列单片机一般扩展系统为例,从软硬件两个方面分析了影响单片机系统掉电保护的关键因素,提出了可靠掉电保护对硬件的基本要求以及软件编程的中心思想,推论出针对不同硬件与系统结构情况下掉电保护全局处理方案,介绍了在正常供电情况下可靠实现掉电保护的新方法。

7.2 提高单片机应用系统可靠性的软件技术

摘自《计算机应用研究》月刊,2002 年第 6 期

软件抗干扰技术是抑制电磁干扰的重要手段。本文首先分析了电磁干扰对单片机应用系统告成干扰的主要后果,然后分别介绍了提高单片机应用系统可靠性的各种软件抗干扰技术。

7.3 单片机应用系统中元器件的可靠性设计

摘自《电子元件与材料》月刊,2002 年第 3 期

单片机应用系统的可靠性很大程度上依赖于所用元器件的可靠性。本文介绍了可靠性概念及影响单片机可靠性因素。着重阐述为保证单片机可靠性,对元器件的要求和可靠性设计。

7.4 DSP 复位问题研究

摘自《自动化与仪表》双月刊,2002 年第 6 期

在数字信号处理器的应用系统设计中,复位处理是一个最基本又极为关键的问题。本文较全面地阐述了 TMS320F206DSP 的复位和抗干扰问题,并就如何保证 DSP 系统运行的实时性进行了重点讨论,详细介绍了几种相关的复位方法。

7.5 计算机 RAM 检错纠错电路的设计与实现

摘自《控制工程》双月刊,2002 年第 1 期

本文论证了在空间环境下影响 RAM 可靠性的主要因素及主要故障模式的同时,介绍了利用 FPGA 实现 RAM 检错纠错电路的方法,给出了仿真结果。并将此方法同用中小规模集成电路实现 RAM EDAC 的方法进行了比较。

7.6 利用 USB 接口进行软件加密的设计思想和实现方法

摘自《工业控制计算机》月刊,2002 年第 12 期

本文首先概括了软件加密的两种方法软加密和硬加密,然后在详细剖析 USB 的物理接口、拓扑结构以及逻辑结构的基础上,详细阐述了利用 USB 接口对软件进行加密的设计思想和实现方法。

7.7 计算机电磁信息泄露与防护研究

摘自《电子技术应用》月刊,2002 年第 4 期

随着计算机安全技术的迅速发展,计算机电磁信息泄露问题已得到了广泛关注。本文对计算机电磁信息泄露进行了系统的分析,介绍了视频信息接收还原的原理,并详细说明了计算机电磁信息泄露的防护方法。

7.8 USB 软件狗的设计及反破解技术

摘自《电子技术应用》月刊,2002 年第 7 期

本文介绍了软件狗技术的发展,提出了一种改进的低成本 USB 软件狗的设计方案,分析了常见的加解密技术,并据此提出了一系列反破解措施。

7.9 全隔离微机与单片机的 RS－485 通信技术

摘自《单片机与嵌入式系统应用》月刊,2002 年第 7 期

本文介绍了利用微机串口与单片机进行全隔离 RS－485 通信的技术。包括工作原理、硬件设计和软件编制。

7.10 印制板的可靠性设计

摘自《计算机测量与控制》月刊,2002 年第 12 期

本文较全面地阐述了印制板可靠性设计涉及的几个问题,讨论了干扰的耦合方式,分析了常用的干扰抑制和隔离方法,提出了可靠性设计的具体方法,可供设计者参考。

7.11 多层布线的发展及其在电源电路电磁兼容设计中的应用

摘自《电子技术应用》月刊,2002 年第 1 期

在电源电路设计中,电源的电磁干扰水平受布线的影响很大。本文通过实践,分析了多层布线的发展及其在电源电路电磁兼容设计中的应用。结合 PROTEL 公司的 PROTEL99SE 软件,给出了一种在电源电路印制板设计中减少电磁干扰的设计方法。

7.12 印制电路板的电磁兼容性预测

摘自《现代电子技术》月刊,2002 年第 8 期

本文以有限元数值计算方法和 SPICE 仿真软件为工具,通过对印制电路板(PCB)耦合微带线的电磁兼容性预测,来对具有不同厚度 PCB 的电磁干扰水平进行估计分析。并对利用屏蔽地线把低频信号迹线和高速数字信号迹线分开来降低 EMI 水平的方法给以预测评估。最后归纳总线出影响 PCB 电磁兼容性能的因素,找出 PCB 设计中电磁兼容性(EMC)适应性方法的指导性原则,为 PCB 的电磁兼容性设计提供参考依据。

7.13 PCB 的热设计

摘自《现代电子技术》月刊,2002 年第 8 期

热分析、热设计是提高印制板热可靠性的重要措施。本文基于热设计的基本知识,讨论了

PCB 设计中散热方式的选择、热设计和热分析的技术措施。

7.14 密码术研究综述

摘自《微计算机信息》月刊,2002 年第 11 期

本文评述了密码术的研究现状,分析了对称密码系统的 EDS、IEDA、AES 标准,公钥密码体制的 RSA、Diffie－Hellman 算法、椭圆曲线密码体系。总结了近年来国内外密码理论的研究热点,并展望了其发展趋势。

7.15 利用汇编语言实现 DES 加密算法

摘自《单片机与嵌入式系统应用》月刊,2002 年第 6 期

DES 算法是一种数据加密算法。自从 1977 年公布以来,一直是国际上的商用保密通信和计算机通信的最常用的加密标准。DES 算法的实现一般用高级语言。本文介绍了 DES 算法原理以及用 51 汇编语言的实现方法,给出了一些子程序。

7.16 USB 保护电路的选择

摘自《电子产品世界 B》月刊,2002 年第 6 期

USB 接口接入受损电缆或瞬间短路时,必须对 USB 集线器及主机装置提供有效保护。本文介绍了几种 USB 保护电路,重点介绍了高分子聚合物型自恢复保险开关和 AAT4610 型限流负载开关的保护办法。

7.17 基于 CAN 总线的多机冗余系统的设计

摘自《计算机测量与控制》月刊,2002 年第 12 期

本文分析了传统多机冗余系统不足之处以及 CAN 总线特点,提出了一种基于 CAN 总线的多机冗余系统设计方法。该方法可以大幅度提高系统的可靠性,并实现从机到主机的无扰切换,且可使系统易于维护、成本低廉、开放性好。

7.18 蓝牙链路层安全性

摘自《计算机应用》月刊,2002 年第 11 期

本文主要讨论了蓝牙链路层的安全性问题,包括链路字的生成和传递、鉴权过程和加密过程,最后简要讨论了蓝牙安全体系中存在的一些漏洞。

7.19 开关电源谐波含量测试分析及抑制

摘自《电子测量技术》双月刊,2002 年第 1 期

文中通过对开关电源谐波含量的测试,分析了谐波产生的原因,提出了降低谐波含量的措施。如接入 EMI 滤波器、无源功率因数校正及有源功率因数校正等。

7.20 系统可靠性冗余的优化研究

摘自《自动化仪表》月刊,2002 年第 1 期

本文讨论了系统可靠性冗余决策方法的过程,冗余的优化模型及其算法,研究了 Aggarwal

启发式算法框图及程序。

7.21 电子工程系统中电磁干扰的诊断和控制方法初探

摘自《兵工自动化》双月刊,2002 年第 4 期

本文分析了电子工程系统中电磁干扰分信息传导干扰和电磁噪声传导干扰。传导干扰的电磁传输通道有电容传输耦合、电阻传输耦合、电感传输耦合,和以电磁波形式传播的辐射干扰。电磁干扰源可通过频谱分析仪,根据干扰信号的频率和带宽来测试。辐射源可通过近场测试方法,用电流卡钳检测共模电流和用近场探头检测机箱泄露来诊断。其测试诊断步骤依序为 :传导发射测试、辐射发射测试、共模电流测量和机箱泄露检查。解决电磁干扰的途径包括机箱屏蔽、电路滤波和电缆保护 3 项措施。

7.22 微机化仪器电磁兼容性设计

摘自《电测与仪表》月刊,2002 年第 12 期

本文从微机化仪器电磁兼容性问题的特殊性出发,依据增强总体抗干扰性能的原则,提出对这类仪器各单元部件、子系统以及由其形成的测量系统等,应采用的不同抗电磁干扰技术及具体的实现方法。

7.23 电磁兼容设计中的屏蔽技术

摘自《现代电子工程》季刊,2002 年第 1 期

本文结合成功完成某设备电磁兼容设计的经验,对电子产品电磁兼容设计中的屏蔽技术进行了较详细的阐述,介绍了一些屏蔽设计中常用的方法。

7.24 几种电磁干扰的分析与解决

摘自《电子产品世界 A》月刊,2002 年第 1 期

本文简要叙述在雷达信号处理机的研制过程中遇到的几个比较典型的电磁干扰现象,并对它们产生的原因作了系统的分析,且说明了相应的解决途径。

7.25 计算机的电磁干扰研究

摘自《微电子学与计算机》月刊,2002 年第 4 期

文中分析了来自外部和内部的电磁干扰的形式及其影响,讨论了干扰的耦合方式,分析了计算机工程中常用的干扰抑制的隔离方法,提出了一种较理想的供电、接地和避雷的电磁兼容系统方案。

7.26 电子电路中抗 EMI 设计

摘自《电子元件与材料》月刊,2002 年第 10 期

电磁干扰 (EMI) 可分为传导干扰和辐射干扰,消除干扰的方法可以采用硬件方法和软件方法,硬件方法主要有屏蔽、接地、隔离和滤波等,而软件方法主要是滤波。文中对各种方法进行总结,并指出各种方法所存在的问题以及适用场合,并对一些常用方法提出改进措施。

7.27 测试系统中干扰及其形成机理

摘自《电子技术应用》月刊,2002 年第 10 期

本文阐述了测试系统中的各类干扰,并对其产生的原因作了较详细的分析。针对干扰的特性,指出了它们的危害范围及程度,以便于对其进行抑制,增强抗干扰的效果。

7.28 一种基于 ST62 单片机的强抗干扰控制器的设计

摘自《测控技术》月刊,2002 年第 3 期

本文从实际应用角度出发,针对控制器工作环境中所存在的噪声干扰,以 ST62 单片机为核心,在硬件、印制电路板、软件等几方面采取一系列的抗干扰措施,设计了一种强抗干扰的控制器。

7.29 微控制器硬件抗干扰技术

摘自《单片机与嵌入式系统应用》月刊,2002 年第 9 期

文中介绍了干扰的形成过程,并从实用角度介绍微控制器硬件抗干扰的基本原理,给出设计微控制系统所应采取的基本方法。

7.30 一种具有高抗干扰能力单片机通信电路的设计

摘自《工业控制计算机》月刊,2002 年第 5 期

针对单片机间通信系统设计中存在的一些具体问题,设计了一种简便、经济、适用的通信电路,在简化设计和节约成本的情况下提高了通信系统的抗干扰能力。

7.31 测控系统抗干扰设计

摘自《电工技术》月刊,2002 年第 9 期

文中提出了一种由 MAX813 和 93C46 新颖集成芯片构成的低成本、高可靠性检测系统抗干扰设计方法。给出了看门狗、掉电后供电延时、电源故障实时监控、自动复位等功能的实现电路及相关程序,该法可有效克服来自电源等方面干扰,提高系统在恶劣环境下的工作可靠性。

7.32 单片机应用系统的抗干扰软件设计

摘自《计算机测量与控制》月刊,2002 年第 11 期

单片机应用系统中,除采用硬件干扰措施外,亦常采用软件抗干扰技术。本文从拦截失控程序、即刻状态恢复、软件数字滤波、RAM 数据保护、防止控制状态和系统死锁等方面论述了软件抗干扰技术。

7.33 变频系统测控软件抗干扰研究

摘自《计算机测量与控制》月刊,2002 年第 11 期

本文介绍了变频系统中强电条件下设计运行测控系统软件抗干扰技术,论述了抗干扰、防系统“死机”程序的设计思路,讨论了软件抗干扰的数据的置入方式、容错设计以及软件设计应注意的问题及方法。

7.34 快速瞬变脉冲群干扰的原理及硬件防护

摘自《电测与仪表》月刊,2002 年第 2 期

文中介绍了脉冲群干扰的危害以及其产生和作用原理,并给出外部接口电路的抗干扰设计方法。

7.35 巧用单片机软件抗系统瞬时干扰

摘自《自动化与仪器仪表》双月刊,2002 年第 1 期

本文介绍了 8031 单片机的待机方式及利用单片机待机方式实现软件抗瞬时干扰的工作原理,给出了软件抗瞬时干扰的设计和应用实例 ;最后,讨论了软件抗瞬时干扰的效果及应用前景。

7.36 微机式保护装置中浪涌干扰的硬件防护

摘自《电测与仪表》月刊,2002 年第 8 期

本文介绍了电力系统中浪涌干扰的产生以及对电路的作用原理和危害,并结合实际应用电路,给出了针对浪涌干扰,电路所应采取的防护措施。

7.37 具有抗干扰性能的单片机智能仪表的设计

摘自《电测与仪表》月刊,2002 年第 3 期

本文从阐明抗干扰智能仪表的工作原理着手,介绍了智能仪表的强抗干扰能力的模拟输入电路、CPU 主控模板、数码显示电路等几个硬件线路,并对于智能仪表的程序设计思想和智能仪表的其他特性出给出简要地说明。

7.38 RS232 串行通信消除干扰噪声的设计方法分析

摘自《现代电子技术》月刊,2002 年第 5 期

RS232 通信线路在传输信号的过程中,数据由于受到某些电磁干扰而使原来的数据信号发生错误。本文对 RS232 通信线路的防干扰问题进行了分析,介绍了通信线路的绝缘设计方法、大地地线、电源地线应用。电源保护及光电隔离技术。

7.39 热插拔冗余电源的设计

摘自《微型机与应用》月刊,2002 年第 4 期

本文介绍了热插拔冗余电源的基本概念、设计思想和电路结构,分析了热插拔和电流共享的原理,给出了以 LTC4350 为核心构成热插拔冗余电源阵列的设计示例。

7.40 IC 卡读写器的密码识别

摘自《电工技术》月刊,2002 年第 10 期

文中介绍了 IC 卡读写系统,它采用了 X76F100 作串行 E^2PROM,以 MCS51 单片机为核心,可实现查询、安全存储、修改密码等功能。文中对其密码识别过程做了详尽说明。

7.41 16 位高抗干扰 D/A 转换

摘自《电子技术》月刊,2002 年第 3 期

本文介绍了一种由可编程计数芯片 8254/8253 利用脉宽调制原理实现高精度、高抗干扰能力的 16 位 D/A 转换,文中给出了具体电路和软件方法。

八、DSP 及其应用技术

8.1 TMS320F206 定点 DSP 芯片开发实践

摘自《半导体技术》月刊,2002 年第 3 期

本文以 TMS320F206 为例,着重探讨了 DSP 系统开发过程中的硬件设计与调试、软件设计中的流水线冲突、等待状态设置以及如何利用闪速存储器等相关问题。

8.2 ADSP2181 精简开发板的研制

摘自《电子技术》月刊,2002 年第 6 期

本文介绍了 ADSP2181 的功能及其特性,详细地给出了基于 ADSP2181 的精简开发板的硬件和软件说明,最后介绍了该开发板的特点。

8.3 DSP 系统中的外部存储器设计

摘自《电气自动化》双月刊,2002 年第 3 期

本文介绍了高速 DSP 芯片与低速外部存储器(片外 RAM 和片外 ROM)之间的数据通信,包括外部存储器的选择和 DSP 与外部存储器的接口。并以 TMS320C54X 与 FLASH MEMORY 的接口为例详细说明。

8.4 Flash 存储器在 DSP 系统中的应用

摘自《电子技术》月刊,2002 年第 6 期

本文主要介绍了 ADSP2189 在 BDMA 应用模式下的设置以及对 Flash 存储器的基本操作,并就常用操作提供了硬件接口和程序范例。

8.5 DSP 系统的硬盘接口研究

摘自《仪表技术与传感器》月刊,2002 年第 3 期

在公路监测系统中,数据量大,DSP 和硬盘的接口可以较好地满足需要。本文介绍了 DSP 和硬盘的接口方法,详细叙述了 TMS320VC5402 通过 IDE 接口控制硬盘读写的方法,并参照 Microsoft 的 DOS 给出了 TMS320VC5402 的磁盘操作系统。

8.6 TMS320C6201 与 Flash RAM 的接口设计与编程技术

摘自《电子技术》月刊,2002 年第 3 期

本文以 SST39VF040 为例,详细介绍了 TI 公司的高性能数字信号处理芯片 TMS320C6201 与 Flash RAM 的接口设计,以及具体的编程方法。

8.7 基于 DSP 的实时 MPEG-4 编码的软件优化设计

摘自《电子技术应用》月刊,2002 年第 6 期

结合开发工具 TMS320C6201EVM 板的结构和特点,本文阐述了在实现 MPEG-4 实时视频编码中,对算法的软件优化所做的工作。

8.8 TMS320C62X DSP 的软件开发与优化编程

摘自《计算机工程》月刊,2002 年第 1 期

定点 DSP 芯片 TMS320C62X 具有高速的数据处理能力,该芯片为开发出具有实时分析处理能力的高速 DSP 系统提供了硬件基础,而开发优化的系统处理软件,是提高 DSP 系统实时处理数据性能的关键。本文介绍了其软件优化的流程过程,并探讨和介绍了源程序优化的常用方法。

8.9 IP 安全内核及其 DSP 实现的研究

摘自《计算机工程》月刊,2002 年第 1 期

本文在研究、综合分析国内外一些安全产品及其相关先进、成熟技术的基础上,提出并建立了一个具有自己特色的安全网关解决方案:将 IPSec 协议与 Linux 操作系统网络协议栈 IP 层相集成,同时,将重要加密算法和认证算法固化到专用 DSP 芯片中。系统实践结果证明,在系统的通用性、安全性以及处理效率等各方面都得到大大提高。

8.10 基于 TMS320C54X DSK 平台的 Zoom-FFT 的快速实现

摘自《电测与仪表》月刊,2002 年第 3 期

本文重点介绍了基于复调制的细化 FFT (ZFFT) 的原理,并详细讨论了在 TMS320C54X DSK 平台上快速实现的方法,包括频率移位、抽取器、N 点复数 FFT 的实现方法和实验结果。

8.11 高速 DSP 与串行 A/D 转换器 TLC2558 接口的设计

摘自《电测与仪表》月刊,2002 年第 9 期

根据高速定点 DSPTMS320F206 的特点,本文提出了使用串行 A/D 转换器 TLC2558 作为 DSP 系统的模拟量输入部分,解决了以往基于并行数据传输的 A/D 转换器不能与高速 DSP 进行很好配合的问题。在此基础上设计了 DSP 与串行 A/D 转换器连接的硬件电路,并就 A/D 转换的软件设计时应注意的问题进行了探讨。

8.12 TMS320C2X DSP 的一种实用人机接口的设计与实现

摘自《无线电工程》月刊,2002 年第 4 期

本文介绍了用 Intel8279 作为 TMS320C2X DSP 的一种键盘显示器接口的设计方法,用简易手段实现了一种实有的 TMS320C2X DSP 人机接口,文中给出了电路图及读键盘部分程序。

8.13 DSP 系统中常用串口通信的设计

摘自《微型机与应用》月刊,2002 年第 12 期

本文从硬件和软件设计介绍了 DSP 芯片与 AD/DA 芯片、具有标准异步串口的芯片及有线 MODEM 3 种常用外部设备之间的串口通信的设计及初始化程序。

8.14 DSP 与单片机之间串行通信的实现

摘自《电子技术》月刊,2002 年第 2 期

本文给出了 AT89C51 单片机系统和 TMS320F240 DSP 系统之间的进行实时交互通信的硬件接口电路、串行通信口的初始化设置及软件实现方法。通信双方均以查询方式进行收发。考虑到通信双方既是发送方又是接收方,为保证通信时序紧密配合,通信协议采用了"发送一个数据,等待接收到确认信号,再发送下一个数据"的方式。

8.15 基于 DMA 方式的 8 位单片机与 16 位 DSP 双机通信接口

摘自《微计算机信息》月刊,2002 年第 6 期

本文介绍通用 8 位单片机 (MCU) 8051 与 16 位信号处理单片机 (DSP) TMS320C25 基于后者提供的 DMA 功能的双机通信方法,给出了总线隔离硬件电路及总线连接以及地址译电路,简要介绍了这样一个双机系统的应用。

8.16 DSP 与 PC 机间的 DMA 通信接口设计

摘自《计算机测量与控制》月刊,2002 年第 5 期

本文介绍了应用于电器试验高速数据采集的以 DSP (数字信号处理器) 为核心的高速数据采集系统。研究了 DSP 与 PC 机之间的通信方式,对采用 DMA (直接存储器存取) 通信方式的数据传输接口电路进行了重点讨论,对所涉及到的主要硬件及软件技术进行了详细说明。

8.17 TMS320VC5402 与 I^2C 总线接口的实现

摘自《微处理机》季刊,2002 年第 1 期

本文在介绍 I^2C 总线的基础上,利用 TMS320VC5402 多信道缓冲串口 (McBSP) 和主机接口 (HPI),通过软件实现 TMS320VC5402 与 I^2C 总线接口,并给出了两种方式下具体实现方法。

8.18 ZLG7289A 与 DSP - SPI 的接口技术

摘自《电子技术》月刊,2002 年第 12 期

本文介绍串行 LED 数码管显示驱动与键盘管理芯片 ZLG7289A 与 TMS320F240 - SPI 的接口技术,给出软硬件设计思想,并讨论 ZLG7289A 存在的问题和解决方案。

8.19 DSP 与 PCI 总线接口设计及实现

摘自《微电子学与计算机》月刊,2002 年第 3 期

本文以 AD 公司的 SHARC 处理器芯片为例,介绍一种采用 FPGA 进行 ADSP - 2106x 与 PCI 总线的接口电路设计方法。并详细介绍了一种利用 FIFO 去耦合的设计思想,有效地消除了 DSP 外部总线与 PCI 总线数据传输中的瓶颈问题。

8.20 TMS320C6X 与 PC 高速通信的实现

摘自《中国仪器仪表》月刊,2002 年第 1 期

TL16C750 是 TI 公司生产的通用异步通信芯片,可应用在高速串行通信中。本文介绍了 TL16C750 的性能及有关寄存器,给出了其在 TMS320C6X 与 PC 通信应用中的硬件电路及 TMS320C6X 初始化 TL16C750 的汇编程序。

8.21 DSP 与 PC 之间的以太通信

摘自《电子技术》月刊,2002 年第 7 期

本文介绍了 Analog Device 公司的数字信号处理 (DSP) 通过网络接口设备——串行网络接口控制器 (SNIC) 与 PC 进行通信的硬件和软件设计,并给出了以太通信用于实时数据传输的实例。

8.22 TM1300 DSP 系统以太网接口的设计

摘自《单片机与嵌入式系统应用》月刊,2002 年第 11 期

基于 IP 网络的多媒体应用越来越广泛,本文首先解决多媒体 DSP 芯片 TM1300 与以太网控制器 CS8900A 的硬件接口的设计,分析嵌入式操作系统 pSOS + 内核中实现 TCP/IP 协议栈的网络模块 pNA + ,最后实现在 pSOS + 操作系统环境下 CS8900A 的网络驱动程序的设计。

8.23 基于 DSP 的 CAN 总线通信系统

摘自《工业控制计算机》月刊,2002 年第 3 期

本文给出了一种以 DSP 为微控制器的 CAN 总线通信系统,介绍了 DSP 与 CAN 控制器的接口电路、工作过程和软件功能。该系统在某电源监控系统中得到了成功的应用,取得了满意的效果。该方法提供的基本思路也可用于其他 CAN 分布式控制系统中。

8.24 TMS320VC5410 DSP 中 USB 客户驱动程序开发与实现

摘自《计算机应用》月刊,2002 年第 6 期

本文以 USB54X EVM 这种带 USB 接口的数字信号处理器系统评估板的二次开发为例,介绍了如何运用 WinDriver5.0 API 结合 USB 控制芯片 AN2131Q 开发基于 windows9x 平台 USB 客户驱动程序。

8.25 基于 TMS320C55x DSP 的 USB 通信研究与固件设计

摘自《电测与仪表》月刊,2002 年第 8 期

本文首先介绍了 USB 设备开发相关的 USB1.1 规范,并介绍了 TMS320C5509 中的 USB 模块,随后重点阐述了 SIE 和固件的分工,最后给出了固件设计思路及关键点。

8.26 基于 DSP 的 USB 口数据采集分析系统

摘自《电子技术应用》月刊,2002 年第 11 期

本文介绍了一种基于 DSP 的 USB 口振动、噪声信号采集分析系统构造方案,并对其各模块进行了分析,该方案完全实现了在系统编程和配置。针对 USB 模块详细介绍了 CYPRESS 公司的 EZ－USB 芯片,说明了其固件 (Firmware) 和驱动程序框架。

8.27 DSP 数字信号处理器的浮点数正弦的实现

摘自《仪表技术》双月刊,2002 年第 5 期

本文研究了浮点数求正弦的理论方法。该方法通过采用幂级数偏置展开法完成浮点数正弦运算,并在 DSP TMS320F240 上编制了一套算法。该方法思路新颖、精度高、速度快,可移植到其他浮点数运算的单片机上。

8.28 应用 TMS320F240 芯片设计高精度可控信号发生器

摘自《国外电子测量技术》月刊,2002 年第 3 期

本文介绍了 TI 公司 C2000 系列 DSP 中的 TMS320F240 芯片的结构、性能及其特点,并给出了用 TMS320F240 作为可控信号发生器的应用实例。

8.29 基于 MSP430C325 单片机的便携式体温计的设计

摘自《微处理机》季刊,2002 年第 4 期

本文介绍采用低功耗的 MSP430C325 单片机为主控制器,Pt100 为温度传感器,设计的一种便携式人体温度计。具有通信功能,时间功能,可作为病房监护系统的体温采集器。采用交、直两种电源供电方式,单纯使用 3.6V (2A/h) 锂电池供电可连续工作 6 年时间。

8.30 基于 TMS320VC5409 的语音识别模块

摘自《电子产品世界 B》月刊,2002 年第 3 期

本文详细介绍了一种非特写人的数码语音识别的算法 连续距离密度分段概率模型,并且给出了基于 TMS320VC5409 DSP 的语音识别模块的实现方案。

8.31 基于 DSP 的 ADμC812 应用系统设计

摘自《电子技术》月刊,2002 年第 6 期

本文提出了一种基于 DSP 的高性能数据采集系统芯片 ADμC812 应用系统的设计方案,系统采用了主从处理器结构,DSP 和 ADμC812 之间通过双端口 RAM 实现数据交换。

九、HDL 与可编程器件技术

9.1 一种基于 CPLD 器件的现代数字系统设计方法

摘自《自动化与仪表》月刊,2002 年第 1 期

本文介绍了一种利用 CPLD 芯片设计的数字钟电路,该系统采用自顶向下的层次模块化设计手段构建电路,代表了 BDA 的发展趋势。文中结合实例详尽介绍了原理图设计输入方式以及设计过程。

9.2 基于可编程逻辑器件 CPLD 及硬件描述语言 VHDL 的 EDA 方法

摘自《自动化与仪表》月刊,2002 年第 1 期

本文介绍了用可编程逻辑器件 CPLD 及硬件描述语言 VHDL 设计数字系统的方法,给出了一种字符发生器的硬件及软件的设计实例。

9.3 利用硬件描述语言 Verilog HDL 实现对数字电路的设计和仿真

摘自《自动驾驶仪与红外技术》季刊,2002 年第 4 期

本文作者利用新型的硬件描述语言 Verilog HDL 对数字式静电陀螺仪的信号脉冲宽度检测电路进行了设计和仿真研究,并在程序设计的计数过程中,研究了一种新的方法,可使计数和脉宽的检测精度提高一倍,为以后电路的改进提高和集成提供了一种良好的解决方案。

9.4 硬件描述语言 VHDL 指称语义的研究

摘自《微电子学与计算机》月刊,2002 年第 11 期

VHDL 是一种广泛使用的硬件描述语言。但长期以来缺乏严格的形式语义。本文介绍并分析了若干具有代表性的 VHDL 指称语义的研究工作。在此基础上,简要介绍了作者提出的基于时段逻辑的 VHDL 语义的框架时对 VHDL 指称语义的看法。

9.5 VHDL 语言逻辑综合的研究

摘自《电测与仪表》月刊,2002 年第 8 期

VHDL 语言的逻辑综合就是将较高抽象层次的描述自动转换到较低抽象层次描述的一种方法。本文对 VHDL 语言综合进程作了详细的讨论,认为综合过程就是将 RTL 级描述、对设计的电路约束和属性及工艺库这些输入产生一个优化的门级网表。

9.6 CPLD/FPGA 的优化设计

摘自《电测与仪表》月刊,2002 年第 2 期

CPLD/FPGA 面向速度或面向面积进行优化设计,在大多数情况下这两种选择是矛盾的。对速度进行优化设计,需要较多资源 对面积进行优化往往会导致系统速度降低。本文提出采用流水线设计、资源共享和预进位处理的方法解决了这个问题。

9.7 用单片机实现可编程逻辑器件的配置

摘自《单片机与嵌入式系统应用》月刊,2002 年第 9 期

本文介绍基于 SRAM 的可重配置 PLD 的原理,通过对多种串行配置的比较,提出单片机与存储器串行配置方式,从系统复杂度、可靠性和经济性等方面进行比较和分析。

9.8 UART 的 Verilog HDL 实现及计算机辅助调试

摘自《电子产品世界 B》月刊,2002 年第 1 期

通常,RS—232 接口均采用硬件(UART 专用芯片)方式来实现,本文介绍了如何用可编程器件(CPLD 或 FPGA)通过设计 Verilog HDL 语言来实现 UART,以及如何通过计算机来进行调试,为 RS—232 接口提供了一种新的解决方案。

9.9 基于 CPLD 的 UART 设计

摘自《微计算机信息》月刊,2002 年第 2 期

本文介绍一种采用可编程逻辑器件 CPLD 实现 UART 的方法,将 UART 的核心功能集成到 CPLD 上,使整体设计紧凑,小巧,实现的 UART 功能稳定、可靠。所有功能的实现全部采用 VHDL 进行描述。

9.10 用在系统可编程逻辑器件开发并行接口控制器

摘自《微计算机信息》月刊,2002 年第 7 期

本文介绍了微型机并行接口控制器 PIA(MC6821 兼容芯片)的 VHDL 设计与实现。方案采用 Lattice 公司的在系统可编程逻辑 ispLSI1032E 和 ispVHDL Viewlogic System 软件进行设计、仿真与实现。

9.11 用 CPLD 设计 EPP 数据采集控制器

摘自《电子技术》月刊,2002 年第 1 期

基于增强并行口(EPP)的数据采集系统与基于 SPP 和 RS232 接口的数据采集系统相比具有更高的数据传输速率。用 CPLD 开发的基于 EPP 接口的数据采集控制器(以下简称为 EPP 数据采集控制器)具备多项优点:高速度、灵活、功能齐全(得益于 CPLD 丰富的内部资源),引脚自定义为设计 PCB 板提供极大方便。本文介绍使用 CPLD 设计的 EPP 数据采集控制器的原理,并附有应用实例。

9.12 带 FPGA 的 PCI 接口应用

摘自《微电子学与计算机》月刊,2002 年第 7 期

本文介绍了一种带 FPGA 的 PCI 接口的应用及设计方法。该方法将 PCI 接口和 PCI 用户逻辑集成在一片 FPGA 里,可以对整个逻辑进行仿真测试,大大缩短了开发周期,提高了系统集成度和性能。文章重点叙述了 Quicklogic 公司提供的 32 位 TARGET 接口芯片 QL5030 的原理和结构,分析了时序设计要点,给出了典型应用的逻辑框图和注意事项。

9.13 基于 CPLD 的 PCI 总线存储卡的设计

摘自《现代电子技术》月刊,2002 年第 10 期

本文以自行开发的 PCI 总线存储卡为背景,详细论述了 PCI 总线的信号定义和命令操作,以及总线上的数据传输过程。介绍了所有芯片的基本情况,给出了采用 CPLD 实现数据存储功能的主要编程设计。

9.14 基于 CPLD 的中断控制器 IP 设计

摘自《现代电子技术》月刊,2002 年第 6 期

由于使用了 EDA 工具,在设计过程中直接进行时序和逻辑验证,避免了系统设计过程中可能发生的错误。本文论述了基于 CPLD 的中断控制器 IP 设计。用电路图编辑了整个设计,并在 Altera 公司 EPM7128SLC84—6 上实现了该中断控制器。

9.15 基于 FPGA 设计的精度管理策略

摘自《微计算机应用》双月刊,2002 年第 4 期

本文结合 FPGA 的硬件资源具有可定制的特点,对基于 FPGA 的信号处理中精度管理策略进行了探讨。

9.16 VHDL 语言在描述 DES 加密机中的应用

摘自《微电子学与计算机》月刊,2002 年第 12 期

现阶段银行信息安全的加密算法均建立在 DES 加密算法和 RSA 加密算法的基础上。文中提出了以 VHDL 语言为手段,介绍了在描述 DES 加密机的应用,并在 Active—HDL 模拟验证描述的正确性。

9.17 基于 P89C51RD2 IAP 功能的数据存取与软件升级

摘自《单片机与嵌入式系统应用》月刊,2002 年第 11 期

本文分析了 Boot ROM 中的部分源代码,重点是 IAP 功能以及 ISP 和 IAP 的相互关系;应用 IAP 功能将剩余程序空间转化为数据空间,以及自编 ISP 程序来实现仪器的软件升级方法。

9.18 在系统可编程模拟器件 ispPAC30 及其应用

摘自《电测与仪表》月刊,2002 年第 6 期

本文介绍了在系统可编程模拟器件 ispPAC30 的结构与特点、封装与管脚、及其开发环境,并给出一个 ispPAC30 应用的例子,最后还介绍了应用 ispPAC30 器件时的注意事项。

9.19 可编程模拟器设计及 ispPAC30 应用

摘自《电子技术》月刊,2002 年第 6 期

本文对 ispPAC30 的结构与特性、封装与管脚、工作原理、使用事项、开发环境及典型应用做了详细的说明。文中还介绍了 ispPAC30 的一些特殊功能如用户签名、断电模式、SPI 接口和独特的校准方法。

9.20 ispPAD 在模拟电路设计中的应用

摘自《现代电子技术》月刊,2002 年第 3 期

本文介绍了在系统可编程模拟器件 (ispPAC) 的功能和结构,并结合实例介绍了 ispPAC 器件介绍了实现滤波器的使用方法。

9.21 在系统可编程模拟器件 (ispPAC) 及其应用

摘自《现代电子技术》月刊,2002 年第 4 期

本文简要介绍了在系统可编程模拟器件 (ispPAC) 的基本组成和主要特点,并介绍了在 PAC Designer 环境中的编程方法。

9.22 在系统可编程模拟器件 ispPAC20 及其应用

摘自《电子元件与材料》月刊,2002 年第 8 期

本文介绍了在系统可编程模拟器件 ispPAC20 的主要结构和性能特点,以及用一片 ispPAC20 实现电压控制振荡器电路的方法及其工作原理。该压控振荡器电路的实现无需使用传统的运算放大器,也不需外接电阻、电容等元器件,且电路性能优良、工作可靠。只需在计算机上通过专用开发软件 PAC—Designer 进行电路设计,再下载至 ispPAC20 器件中即可实现电路,电路修改快捷、方便。

9.23 ispLSI1032E 器件及其应用

摘自《电子元件与材料》月刊,2002 年第 11 期

本文介绍了 Lattice 公司的大规模集成电路 ispLSI1032E,它是一种在系统复杂可编程逻辑器件 (CPLD),借助于 EDA 设计软件可以随时更改芯片的功能,而且不需要改动印制电路板 (PCB) 的结构,大大提高了系统设计的效率,并保证了设计的正确性。最后设计了基于该芯片的 EDA 教学实验系统。

9.24 用 ispPAC20 实现的最简温度测控系统

摘自《电测与仪表》月刊,2002 年第 1 期

本文介绍应用在系统可编程模拟电路 ispPAC20 和 89C2051 单片机开发的结构最简的温度控制系统。该系统具有温度实时测量、控制温度设定与显示以及控温信号输出等多种功能,硬件组成简单,软件开发方便,控制精度较高。如果改变测温传感器及其接口电路,本系统可适用于不同温度范围的温度控制场合。

9.25 在系统可编程器件设计应用实例

摘自《测控技术》月刊,2002 年第 1 期

本文介绍了 Lattice/Vantice 公司在系统可编程 (ISP,In－System Programmable) 器件下载电路的设计及一些显示电路、计数器、单片机 I/O 接口的应用实例。

9.26 在 FPGA 开发板上设计 8051 的开发平台

摘自《电子产品世界 B》月刊,2002 年第 12 期

本文论述了如何在 Altera 的 FPGA 开发板上设计一个 8051 的开发平台,从硬件设计和软件结构两个方面详细介绍了我们的思路。并且简要介绍了 Inter HEX 文件的格式。

9.27 由可编程逻辑器件与单片机构成的双控制器

摘自《电子技术应用》月刊,2002 年第 1 期

本文介绍一种利用可编程逻辑器件 CPLD 与单片机 AT89C51 串行双向通信而构成的双控制器。

9.28 用 VHDL 设计专用串行通信芯片

摘自《电子技术应用》月刊,2002 年第 1 期

本文介绍了一种专用串行同步通信芯片 (该芯片内部结构和操作方式以 INS8250 为参考) 的 VHDL 设计及 CPLD 实现,着重介绍了用 VHDL 及 CPLD 设计专用通信芯片的开发流程、实现难点及应注意的问题。

9.29 基于 FPGA 的 ARINC429 总线接口芯片的设计与实现

摘自《测控技术》月刊,2002 年第 1 期

本文介绍了 FPGA 芯片自行设计 ARINC429 专用接口芯片组的基本原理、数字信息传输规范、数字逻辑功能,并给出了核心部分的 VHDL 语言描述。

9.30 I^2C 总线通信接口的 CPLD 实现

摘自《单片机与嵌入式系统应用》月刊,2002 年第 5 期

本文介绍采用 ALTERA 公司的可编程器件,实现 I^2C 总线的通信接口的基本原理 给出部分 VHDL 语言描述。该通信接口与专用的接口芯片相比,具有使用灵活,系统配置方便的特点。

9.31 FPGA 模拟 MBUS 总线的实现

摘自《微处理机》季刊,2002 年第 4 期

本文讨论了利用 FPGA 工具实现 MBUS 总线的原理、方法,以实际操作介绍了 FPGA 设计流程,并给出 FPGA 常用设计技巧。

9.32 基于 FPGA 的 USB2.0 控制器设计

摘自《电子技术应用》月刊,2002 年第 12 期

本文介绍了一种用 VHDL 设计 USB2.0 功能控制器的方法,详述了其原理和设计思想,并在 FPGA 上予以实现。

9.33 USB 外设接口的 FPGA 实现

摘自《微电子学与计算机》月刊,2002 年第 12 期

本文讨论了 USB 外设接口的软硬件实现,其中重点研究了用硬件描述语言在 FPGA 中实现 USB 通信协议,同时提供与 USB 线收发器和 MCU 的接口,简述了 MCU 中的 Firmware 和 PC 端的客户程序。既可作为单片专用接口电路实现,也可以作为 IP Core 嵌入到其他 SOC 应用中。

9.34 循环冗余校验码的单片机及 CPLD 实现

摘自《单片机与嵌入式系统应用》月刊,2002 年第 9 期

循环冗余码校验(CRC)是一种可靠性很高的串行数据校验方法。本文介绍循环冗余码校验的基本原理,并分别用单片机和 CPLD 作了循环冗余码校验的软件实现和硬件实现。包括汇编语言和 VHDL 语言源程序。

9.35 可编程芯片在测控系统中的应用

摘自《航空计测技术》双月刊,2002 年第 6 期

本文分析了可编程芯片在设计测控系统中的应用,简述了测控技术发展现状及相关电子技术,介绍了测控系统中 PLD 的设计步骤,以及典型测控系统的实现方法。

9.36 可编程逻辑器件在浮点放大器中的应用

摘自《测控技术》月刊,2002 年第 6 期

本文以捷联惯导系统中陀螺仪的数据采集为背景,设计开发了浮点放大器,以满足陀螺仪较宽的动态输出范围。该设计采用可编程逻辑器件,ABEL－HDL 硬件描述语言为输入工具,接口简单,可靠性高,具有一定的实用价值。

9.37 FPGA 在高速多通道数据采集中的应用

摘自《电测与仪表》月刊,2002 年第 10 期

本文介绍了一个以 FPGA 为核心处理器的高速多路数据采集系统,整个数据采集系统可实现 48 路最大工作频率为 20 kHz 的信号采集任务。在该系统中,还采用了 TI 公司最新推出的高速低功耗 A/D 芯片 THS1206 从而大大降低了系统功耗。另外,采用了 ATMEL 公司的 AT89C2051 实现采集器与 PC 机的串行通信控制。

9.38 在 DSP 采样系统中采用 DAC 实现量程自动转换

摘自《单片机与嵌入式系统应用》月刊,2002 年第 4 期

本文给出了 MAXIM 公司的 12 位串行 D/A MAX543 在 TMS320F206 DSP 的采样系统中实现量程自动转换的具体实现方法,分析 MAX543 中 R－2R 电阻网络结构,讨论 DSP 应用程序的设计,并给出程序流程框图。该实现方法在精密测量中具有普遍的应用意义。

9.39 基于 VHDL 语言的数字频率计设计

摘自《现代电子技术》月刊,2002 年第 7 期

本文介绍了 VHDL 语言在数字频率计设计中的具体应用,说明了实现电子电路设计的自动化 (EDA) 过程和 EDA 技术在现代数字系统设计中的重要地位和作用。

9.40 基于 VHDL 语言的数字频率计的设计

摘自《微电子学》双月刊,2002 年第 3 期

本文介绍了用 EDA 技术作为开发手段,实现数字频率计的设计。系统基于 VHDL 语言,以 CPLD 为核心,具有体积小、可靠性高、灵活性强等特点。文中详细介绍了频率计数模块与扫插显示模块的具体设计方法。

9.41 CPLD 在 SPWM 变频调速系统控制中的应用

摘自《电子技术》月刊,2002 年第 4 期

本文章详细介绍了如何利用 Altera 公司的 Maxplus Ⅱ 软件及 FLEX6000 芯片,来实现 VVVF 控制 SPWM 变频调整方法,设计中提出一种三相分时运算的思路。试验证明,系统可以稳定的产生载波超音频的 SPWM 波,CPLD 应用于变频调整系统控制是非常有效的。

9.42 ISP 技术在交通控制器中的应用

摘自《测控技术》月刊,2002 年第 1 期

在系统可编程大规模集成逻辑器件 ispLSI 编程时无需专用编程器,用户可直接对目标系统反复编程。本文介绍了使用 Lattice 公司的 ISP 器件 1016 芯片,实现交通信号可编程定时控制系统的设计。

9.43 基于 ISP 技术的有限状态机控制系统设计

摘自《电气自动化》双月刊,2002 年第 6 期

本文以自动售邮票机控制系统为例,介绍一种采用 ISP 技术实现有限状态机控制系统设计的方法。由于采用了时钟同步有限状态机输出信号的方式,在系统的输出端没有“毛刺”产生。该方法适用于具有多输入信号、多输出信号和复杂状态转移关系的系统。

9.44 如何使用 ISP 技术产生任意波形

摘自《电子产品世界 B》月刊,2002 年第 9 期

本文简述了以 PC 为数据源产生任意波形的设计思想,对应用在系统可编程技术实现电路的逻辑控制和同步做了重点介绍,最后还对如何提高电路系统的 EMC 性能进行了说明。

9.45 打印控制卡的 FPGA 外围电路设计

摘自《微型电脑应用》月刊,2002 年第 11 期

随着 FPGA 在控制系统中广泛使用,FPGA 的外围电路设计成为影响 FPGA 性能发挥的因素之一。本文以一个打印机控制卡开发为例,详细介绍了如何设计控制卡上 FPGA 的外围电路。

9.46 加密可编程逻辑阵列芯片引脚的判别

摘自《电子技术》月刊,2002 年第 1 期

加密可编程逻辑器件的广泛使用给进口电子产品的维修与维护带来较大困难,对加密芯片引脚类型的判别有助于解决系统中硬件故障的定位问题。本文通过分析可编程逻辑阵列器件的内部结构,提出了利用伪随机循环测试码进行芯片引脚类型判别的具体实施过程。

9.47 蓝牙系统中的加密技术及其算法的 FPGA 实现

摘自《微电子学》双月刊,2002 年 2 月

蓝牙 (Bluetooth) 技术是一种最新的近距离无线连接技术,具有广阔的应用前景和巨大的潜在市场,而安全性是其中一个很重要的技术问题。本文重点分析研究了蓝牙系统中的安全加密技术,用硬件描述语言 VHDL 对加密的核心算法进行了描述,并在 FPGA 上进行了实现和验证。

9.48 运用 VHDL 语言设计电视墙数字图像处理电路

摘自《微计算机应用》双月刊,2002 年第 1 期

本文介绍了一种电视墙 (Video Wall) 数字图像处理电路的设计思路,即采用大规模可编程逻辑芯片,并运用 VHDL 语言设计出一种新型的电视墙显示系统,具有集成度高、模块化、可运用于多种显示方式、体积小、系统的可靠性高等特点。

9.49 CPLD 在电路板故障诊断中的应用

摘自《微计算机信息》月刊,2002 年第 3 期

本文简要介绍了带微处理器的数字电路板智能故障诊断系统的系统组成及逻辑控制与接口部分的工作原理,着重描述了 CPLD 器件的设计与实现,最后就仿真实验及设计过程中的一些注意事项做了简要的说明。

9.50 用硬件描述语言设计一个简单的超标量流水线微处理器

摘自《工业控制计算机》月刊,2002 年第 4 期

提高微处理器的指令级并行是微处理器体系结构发展的方向,硬件描述语言和抽象能力强,本文论述了用硬件描述语言设计一个具有超标量流水性能的简单微处理器的设计思想及实现。

9.51 用 CPLD 技术实现高速数据识别码检测器

摘自《现代电子技术》月刊,2002 年第 1 期

数字系统中信号的识别提取首先要检测识别码来确定信号是否到达,本文介绍了用 CPLD 技术和查表的方法实现一个高速数据识别码检测器,提高了数据的检测速度和可靠性。

9.52 用 CPLD 控制 ISD2590 语音芯片的技术应用

摘自《电子元件与材料》月刊,2002 年第 1 期

本文介绍如何利用可编程逻辑器件 (CPLD) 来控制 ISD2590 语音芯片,以及在公用电话网家电控制系统中的应用。

十、综合应用

10.1 嵌入式处理器 StrongARM 的开发研究

摘自《计算机工程》月刊,2002 年第 8 期

本文通过对目标机的调试程序 Angel、主机集成开发和调试环境 ARM Developer Suite (ADS)、主机和目标机的通信协议 Angel Debug Protocol (ADP) 的分析,对嵌入式处理器 StrongARM 系统的具体开发和调度进行了探讨。

10.2 基于 StrongARM 的视频采集与处理系统

摘自《电子技术应用》月刊,2002 年第 11 期

本文介绍一个基于 StrongARM 的视频数据采集处理系统。该系统将采集到连续视频图像数据以 MJPEG 的方式进行压缩处理,然后由 StrongARM 进行打包处理,生成 UDP 包,向网络发送。视频服务器可以通过网络 (局域网或广域网) 获取视频采集器发送的图像数据,并对图像数据进行显示、存储、回放等管理,同时视频服务器也可以通过网络控制视频采集器上的摄像机及云台。系统中使用 Intel® 公司的 StrongARM SA－1110 高性能微处理器芯片作为处理平台。

10.3 基于 StrongARM 的远程网络监控系统设计

摘自《计算机应用》月刊,2002 年第 11 期

介绍一个基于 IP 网络的数字远程实时监控系统,其前端视频数据采集设备中使用 Intel 的 StrongARM 嵌入式微处理器,并采用 Motion JPEG 压缩、分组传输的方式,实现了视频信号的实时采集与远程处理。文中首先简单介绍整个系统的构成,然后详细介绍视频采集设备的结构以及视频服务器网络通信、多路显示等功能模拟的软件实现。

10.4 基于 80C196KC 的 CAM 锁定功能实现可控硅的触发控制

摘自《微计算机信息》月刊,2002 年第 8 期

利用 80C196KC 单片机新增的 CAM 锁定功能和定时器 T2 的内部时钟功能,可以实现可控硅触发脉冲的精确控制,该方法简单实用,文中还给出了相应的软件实现方法。

10.5 基于 MSP430F149 的低成本智能型电力监测仪

摘自《电子技术》月刊,2002 年第 4 期

本文以 MSP430F149 单片机为核心,研制了一种智能电力监测仪,对其硬件和软件结构作了详细介绍。该监测仪具有低成本、低功耗、多功能等特点。

10.6 一种基于 ADμC812 单片机的数据采集器

摘自《电测与仪表》月刊,2002 年第 1 期

本文介绍了一种基于 ADμC812 单片机的数据采集器的硬件组成结构和软件设计方法,该方法简单实用,具有较广的使用价值。

10.7 基于 PIC16C72 单片机的线性 V/F 转换器设计

摘自《单片机与嵌入式系统应用》月刊,2002 年第 4 期

本文介绍了一种利用 PIC16C72 单片机实现脉冲输出的线性电压/频率转换器的方法;阐述转换器的电路设计、工作原理、软件设计及线性化处理技术。

10.8 基于 PIC16C923 单片机的非接触式光纤温度测量仪

摘自《中国仪器仪表》双月刊,2002 年第 3 期

本文介绍了一种新型基于 PIC16C923 单片机的非接触式光纤传感测温仪。整个系统测量精度高,响应速度快,体积小,并且具有非线性校正、可选择测量方式、LCD 显示和 RS—232 串口通信功能。

10.9 用 89C2051 构成智能仪表的键显接口

摘自《电测与仪表》月刊,2002 年第 4 期

本文给出了一种采用 89C2051 单片机的智能仪表键显接口电路的设计,其功能为:驱动 16 位 LED 数码管及 8 个 LED 发光二极管;采集 32 个按键。其特点为:实现了键显接口的智能化;电路硬件结构简单且价廉。

10.10 基于 89C2051 的解码器设计

摘自《微电子学与计算机》月刊,2002 年第 5 期

本文简述了解码器的工作原理,详细描述了一种基于单片机的软标签解码器的设计思路及其实现方法,给出了系统的硬件原理图和软件流程图。

10.11 基于 AT89C2051 的准方波逆变电源

摘自《电工技术》月刊,2002 年第 1 期

本文以移相控制方式为机理,介绍一种基于 AT89C2051 单片机为核心的户用太阳能准方波逆变电源。分析了电源装置运行的软件实现思想,并给出了主要程序清单。

10.12 单片机 AT89C2051 构成的智能型频率计

摘自《现代电子技术》月刊,2002 年第 2 期

本文介绍利用单片机 AT89C2051 实现频率测量的原理及其硬件结构和软件设计,讨论了提高系统测量准确度的方法,并对用单片机实现频率零量化误差测量作出了探讨。

10.13 基于 AT89C2051 单片机的旋转变压器位置测量系统设计

摘自《计算机测量与控制》月刊,2002 年第 3 期

本文介绍了旋转变压器鉴相型位置测量系统的结构和工作原理,深入阐述了基于 AT89C2051 单片机的旋转变压器鉴相型位置测量系统的硬件及软件的设计方法。

10.14 AT89C2051 单片机对显示驱动芯片 MC14499 的 IC 级代换

摘自《电测与仪表》月刊,2002 年第 12 期

本文介绍了采用 AT89C2051 单片机实现对显示驱动芯片 MC14499 的 IC 级代换的思路和具体实现方法。文中详细介绍了 MC14499 的工作原理、应用电路及 AT89C2051 代用的具体电路及驱动程序。

10.15 实用变量程模拟信号单片机检测电路

摘自《电测与仪表》月刊,2002 年第 3 期

本文介绍了一种单片机自动换量程的 V/F 变换信号采集电路;分析了 V/F 的原理,用以提高对变化缓慢的信号采集的抗干扰性。从两个方面提高信号采集的精度。

10.16 GPS 高精度时钟的设计和实现

摘自《单片机与嵌入式系统应用》月刊,2002 年第 11 期

本文介绍采用 GPS OEM 接收板来实现精密时钟系统的设计思路和方法,给出基本的硬件电路和软件流程。文中详细介绍了 GSU－16 的硬件接口、软件接口、应用电路及初始化程序。

10.17 一种基于 GPS 的高速数据采集卡的实现

摘自《工业控制计算机》月刊,2002 年第 4 期

本文阐述了一种基于 GPS 同步时钟的高速数据采集系统的基本原理及硬件电路的具体构成。此卡采样频率为 20MSPS,采样字长 8 位。其性能指标符合高速数据采集的要求,适用于多种场合。

10.18 V/F 转换电压测量系统

摘自《微型机与应用》月刊,2002 年第 12 期

本文提出了一种用 V/F 转换器实现电压自动测量的硬件和软件方案,阐述了相关测量原理。文中有详细的系统电路图、光耦隔离驱动电路及参考电压测量电路。

10.19 用 20 位 DAC 实现 0～10 V 可程控精密直流参考源的设计

摘自《电测与仪表》月刊,2002 年第 4 期

本文介绍了单片低功耗、具有片内自校准功能的 20bit 串行 DAC1220 在可程控精密 DC 参考源中的设计应用,给出了与 51 系列单片机的接口电路。

10.20 单片 MAX752 实现的 CCD 供电电源的设计

摘自《仪表技术与传感器》月刊,2002 年第 4 期

本文介绍了一种升压式 DC－DC 变换器芯片 MAX752,利用该芯片设计出了适宜电池供电的 CCD 供电电源。给出了测试结果和用途,并讨论了改进性能的措施。

10.21 基于双口 RAM 的智能型开关量控制卡的设计

摘自《微型机与应用》月刊,2002 年第 1 期

介绍采用新型的双端口 RAM 芯片 IDT7132 设计的智能型开关量 I/O 卡的软硬件设计,并对其中关键技术进行了详细说明。

10.22 矩阵键盘产生 PC 机键盘信号的应用设计

摘自《单片机与嵌入式系统应用》月刊,2002 年第 11 期

本文介绍了一个用矩阵式键盘为 PC104 总线嵌入式系统配置标准 PC 机键盘的方法。文中详细介绍了 PC 机键盘原理、键盘接口设计,并给出了 C51 的键盘源程序。

10.23 基于 C51 的汉字/数字混合液晶显示及更新的方法

摘自《微处理机》季刊,2002 年第 4 期

本文介绍了 8051 单片机与 T6963C 控制的液晶显示器之间的接口电路,并基于 C51 开发语言,系统地讨论了液晶汉字/数字混合菜单的显示方法以及数据动态更新的实现方法。该方案已获应用,实际使用效果良好。

10.24 实现串行 E^2PROM 芯片的 PC 界面操作

摘自《仪表技术与传感器》月刊,2002 年第 8 期

本文以 X25045 为例,给出了一种实现串行 E^2PROM 芯片功能的新方案,即通过 PC 机和单片机之间的串行通信,可以很方便的实现 PC 机对 E^2PROM 的读写操作,并且可以把需要存入 E^2PROM 的数据以数据库文件的形式保存起来。PC 机界面采用 VB6.0 编程,形象直观。

10.25 一种软硬件结合的 POCSAG 码解码装置研制

摘自《微型机与应用》月刊,2002 年第 4 期

本文针对 POCSAG 码的编码格式,研制了一种基于单片机的接收解码装置。该装置用 MOTOROLA BRAVO 型 BP 寻呼机的射频接收电路接收无线寻呼信号,用硬件实现位同步,用软件实现接收和解码。文中介绍了该装置的硬件组成和工作原理及软件的设计思路。

10.26 蓝牙技术在医疗监护中的应用

摘自《电子技术应用》月刊,2002 年第 5 期

本文简要介绍了蓝牙技术的特点,着重介绍了蓝牙技术在医疗监护中的应用,讨论了蓝牙技术的安全性问题。

10.27 一种红外感应泵液器的单片机应用设计

摘自《单片机与嵌入式系统应用》月刊,2002 年第 11 期

本文介绍的红外微电脑自动泵液器采用 Microchip PIC16C54 单片机,GR40101 红外发射二极管和 GD1611 硅 PIN 型光敏二极管作为红外发射和接收器件,微型电机 QDB-30-3.0 作为泵液器驱动。它采用红外技术感应人手,由单片机控制出液量,具有抗干扰能力强、无误操

作、省电节能等特点。文中介绍了具体的软硬件设计。

10.28 电话报警系统的设计

摘自《单片机与嵌入式系统应用》月刊,2002 年第 3 期

采用单片机技术开发的电话报警系统,当有外人侵入或发生火灾时,能及时地报警,避免事态进一步扩大。本文主要介绍设计的主要思路,包括系统硬件框图结构、部分主要器件和单元电路、软件编程思想及设计中应考虑和解决的几个主要问题。

10.29 无轨电车整流站自动化监控系统

摘自《电子技术应用》月刊,2002 年第 3 期

本文介绍了一种基于上、下位机的无轨电车整流站自动化监控系统。叙述了该系统的组成、工作过程和功能,并着重叙述了短路检测系统的基本原理、组成和工作过程。

10.30 PWM 恒流充电系统的设计

摘自《电子技术应用》月刊 2002 年第 3 期

一种好的蓄电池充电方法是分级恒流充电。本文分析了 PWM 控制 DC/DC 变换的原理,给出了一种基于 IGBT 功率器件、单片机控制的新型蓄电池恒流充电系统的设计方法,并对该系统进行了试验,给出了试验结果。

10.31 微功耗智能 IC 卡燃气表的研制

摘自《微型机与应用》月刊,2002 年第 4 期

本文介绍了一种以 AT89C2051 单片机为核心的智能 IC 卡燃气表。文中详细介绍了其硬件组成和软件设计及为了实现微功耗所采取的措施。

10.32 软件接口技术在串行通信中的应用

摘自《微计算机信息》月刊,2002 年第 1 期

本文介绍了一种在上位机主程序无法与下位机进行通信的情况下,利用软件接口程序实现上位机与下位机通信的新方法。该方法采用了多线程技术和虚拟键盘输入技术将整个系统有机的分成三个各自独立的模块,降低了系统的复杂性,减小了开发成本。

10.33 数字化直流接地系统绝缘检测仪的设计与开发

摘自《计算机测量与控制》月刊,2002 年第 2 期

本文重点阐述了基于 89C51 单片机设计和开发具有数字显示、多路巡检功能的直流接地系统绝缘电阻检测仪的硬件系统。

10.34 4Mbps 红外无线计算机通信卡研制

摘自《计算机应用》月刊,2002 年第 12 期

本文重点研究了 IRDA－1.1 串行红外物理层标准,提出基于 ISA 总线的 4Mb/s 红外无线计算机通信卡的硬件组成原理和驱动程序的设计方法,并且通过两台计算机点对点通信验证

系统可行性。实验结果表明在协议规定的通信距离和视角范围内,其通信速率和误比特率均满足协议标准。

10.35 MCB－1 电力测量控制仪中 CAN 总线通信模板的设计及编程

摘自《电测与仪表》月刊,2002 年第 1 期

本文介绍了现场总线 CAN－BUS 的主要特性和主要技术指标,并结合开发设计的 MCB－1电力测量控制仪表的 CAN 通信接口模板,给出了 CAN 通信接口的一种设计方案和编程方法。以及在现场安装、调试、运行的实际体会。

10.36 单片机在晶闸管触发电路中的应用

摘自《自动化与仪器仪表》双月刊,2002 年第 1 期

本文详细介绍了 8031 单片机在晶闸管触发电路中的应用:高分辨率的数字触发器。文中介绍了硬件组成原理、同步信号输入与触发脉冲输出电路及流程图。

10.37 基于 DS1302 的子母钟系统

摘自《电子技术》2002 年第 4 期

本文在对系统的需求进行分析的基础上,概要介绍了串行实时时钟芯片 DS1302 的基本特性和工作原理,给出了 DS1302 与 AT89CX 单片机在防空子母钟系统中的接口电路实例,并对其中的实现关键技术进行了分析,经测试,系统运行情况良好。