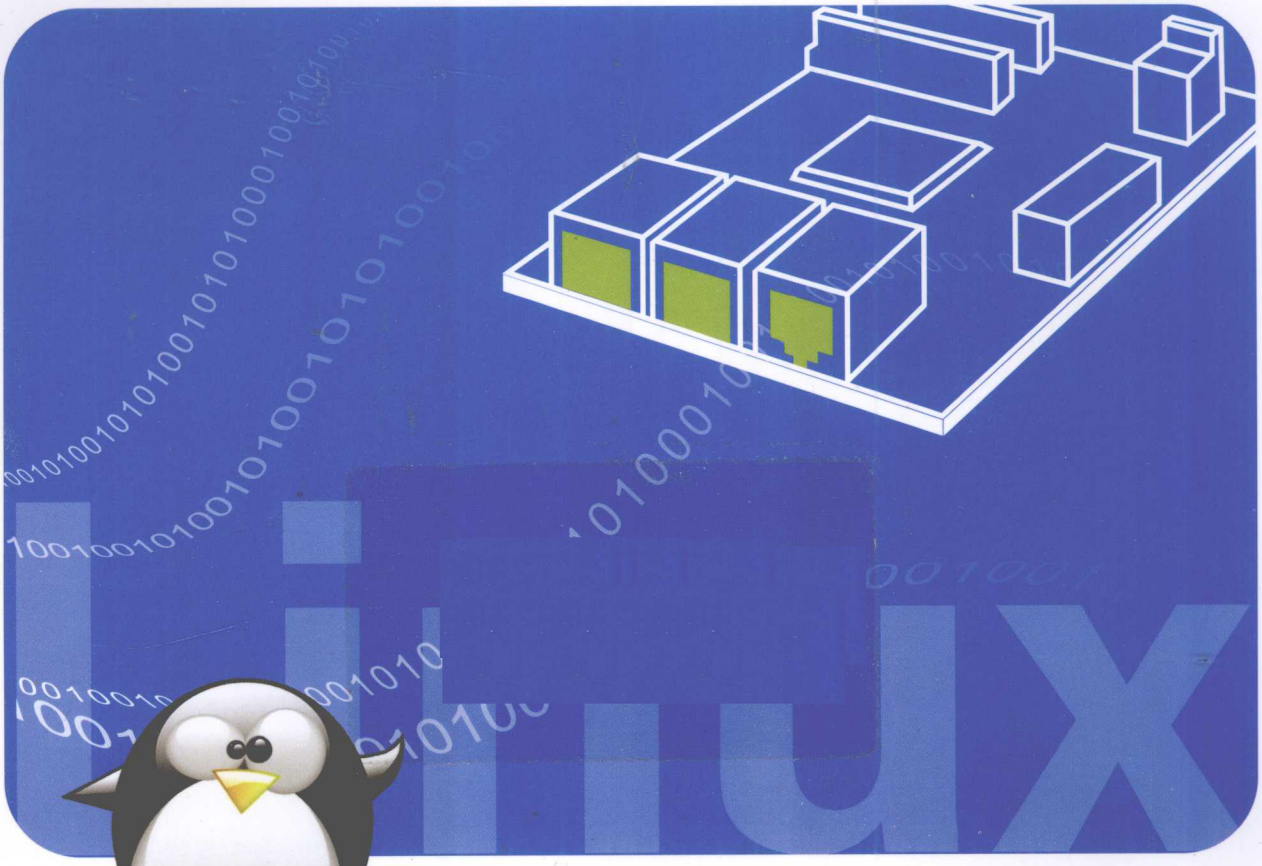
“十一五”高等院校规划教材



郑灵翔 编著

嵌入式接口技术与Linux驱动开发

北京航空航天大学出版社

“十一五”高等院校规划教材

嵌入式接口技术与 Linux 驱动开发

郑灵翔　编著

北京航空航天大学出版社

内 容 简 介

本书以软硬件相结合、底层驱动与上层应用相结合的方法，介绍了嵌入式接口技术的基本原理及应用设计技术。全书可分为4个部分：第1、2章是全书的基础，主要介绍了嵌入式接口技术的软硬件基础知识，并深入介绍了嵌入式Linux驱动的基本原理；第3章在介绍嵌入式存储接口设计的基础上说明了最小系统设计、Bootloader移植以及嵌入式Linux的芯片级和板级移植；第4～6章以小键盘、实时时钟和触摸屏接口为例，在硬件方面介绍嵌入式系统中简单I/O接口的扩展技术以及GPIO的输入/输出和中断功能的使用，同时在软件上介绍这些设备接口的驱动与控制方法，如基于定时器的程序查询式I/O控制、阻塞型和非阻塞型I/O、异步I/O信号的支持方法和中断处理方法、Linux内核的实时时钟子系统和输入设备子系统的使用等；第7～9章介绍了一些软硬件都较为复杂的嵌入式接口，它们包括以太网接口与网络设备驱动原理、PCMCIA接口与PCMCIA驱动原理、AC97音频接口与基于ALSA架构的音频驱动设计。

本书可作为高等院校电类相关专业硕士研究生或高年级本科生的教材，也可以作为嵌入式系统工程师的实用参考书。

图书在版编目(CIP)数据

嵌入式接口技术与Linux驱动开发 / 郑灵翔编著. --
北京 :北京航空航天大学出版社，2010.4
ISBN 978-7-5124-0064-1

Ⅰ. ①嵌… Ⅱ. ①郑… Ⅲ. ①微型计算机－接口－系统设计②Linux操作系统－系统设计 Ⅳ. ①TP364.721②TP316.89

中国版本图书馆CIP数据核字(2010)第065450号

嵌入式接口技术与Linux驱动开发
郑灵翔 编著
责任编辑 刘 晨
*
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100191) http:www.buaapress.com.cn
发行部电话:(010)82317024 传真:(010)82328026
读者信箱:bhpress@263.net 邮购电话:(010)82316936
北京市媛明印刷厂印装 各地书店经销
*
开本:787 mm×960 mm 1/16 印张:18 字数:403千字
2010年4月第1版 2010年4月第1次印刷 印数:5 000册
ISBN 978-7-5124-0064-1 定价:32.00元

前言

随着嵌入式技术的发展，各种各样的嵌入式系统应用已经渗入到我们生活的各个角落。这些种类繁多的嵌入式系统，其本质上都是相同的，都是一种专用的计算机系统，只不过它们的形态各异，接口的种类繁多。在嵌入式系统中，这些复杂多变的接口是嵌入式系统与外部世界联系的桥梁与纽带，有着重要的作用。这也就意味着嵌入式接口技术在嵌入式系统中有着重要的作用。

嵌入式接口的设计是嵌入式系统设计中的一个重要部分。然而，嵌入式接口的设计并不仅仅是硬件设计，嵌入式系统软硬件紧密结合的特点在嵌入式接口的设计中体现得尤为突出。嵌入式接口的设计既要完成硬件接口的设计又要完成软件接口的设计，并使软硬件能相互配合、协同工作，这使得嵌入式接口的设计成为嵌入式系统设计的一大难点。为此，本书以软硬件相结合、底层驱动与上层应用相结合的方式介绍嵌入式接口的原理与设计，以帮助有志于从事嵌入式系统设计的读者，提高嵌入式系统软硬件综合设计的能力。

本书源于《嵌入式系统设计与应用开发》一书，原书中与嵌入式接口以及 Linux 底层驱动相关章节的内容经扩充和改写后形成本书（原书中对嵌入式系统基础知识阐述部分，经扩充和改写后形成了《嵌入式 Linux 系统设计》一书，已于 2008 年在北京航空航天大学出版社出版）。本书主要面向已掌握嵌入式系统设计开发基础识，希望进一步了解嵌入式系统的接口设计与 Linux 底层驱动开发的读者。书中 Linux 内核源码版本以 2.6.20 版为基础。

本书的主要内容有以下几个部分：

- 第 1 章简要介绍了嵌入式接口技术的基本原理，其主要内容包括嵌入式系统中 I/O 接口的连接方式、接口的基本原理、I/O 端口及其编址方式、I/O 接口的扩展方法以及 I/O设备的控制方法等。
- 第 2 章介绍 Linux 2.6 内核底层设备驱动的基本原理与基础知识，本章内容对于未接触过 Linux 设备驱动的读者较为重要，它是嵌入式 Linux 设备驱动设计的基础，应熟练掌握。

本章与第1章的内容是全书的基础，它们介绍了嵌入式接口技术的软硬件基础知识。

- 第3章介绍了嵌入式SOC处理器的存储接口与最小系统设计，本章还分析介绍了与最小系统硬件设计密切相关的Bootloader的移植原理和Linux芯片级移植和板级移植的基本原理。
- 第4章介绍了嵌入式系统中常见的简单键盘/按键接口电路设计，并基于简单字符设备介绍了键盘接口的Linux设备驱动设计。本章涉及了接口技术中常用的一些软硬件设计方法，如统一编址方式下嵌入式处理器的I/O扩展方法，基于定时器的I/O设备程序查询控制方法，阻塞型I/O、异步I/O操作以及异步I/O信号的底层驱动实现与应用软件的编程方法等。
- 第5章介绍了实时时钟与嵌入式处理器接口的设计和实时时钟的Linux设备驱动设计。此外，本章还介绍了如何通过软件控制，利用嵌入式SOC处理器的GPIO引脚实现串行数据的输入输出。
- 第6章介绍了触摸屏的工作原理、触摸屏触点测量原理、触摸屏与处理器的接口设计、GPIO的中断功能、采用Microwire同步串行通信协议进行数据传输的方法以及基于Linux输入设备子系统的触摸屏驱动设计等。在触摸屏的驱动设计中还介绍了中断的使用以及如何使用内核守护线程来完成中断工作的推后执行。

第4～6章在硬件上主要介绍的是如何使用基本的逻辑器件和GPIO接口设计简单的嵌入式I/O接口，在软件上介绍的是这些简单设备接口的驱动与控制方法。

第7～9章则以软硬件都较复杂的嵌入式接口介绍为主。

- 第7章介绍了嵌入式处理器与以太网控制芯片的接口设计方法以及Linux网络设备驱动的原理与实现方法。此外，本章还具体介绍了如何使用Linux内核的平台总线设备模型对设备驱动进行封装，以提高设备驱动的可移植性。
- 第8章介绍了PCMCIA/CF接口的设计原理以及Linux内核的PCMCIA驱动子系统的工作原理和设备驱动的实现方法。
- 第9章介绍了嵌入式处理器AC97音频设备接口的设计原理，Linux内核的ALSA音频驱动架构，以及AC97音频控制器的ALSA驱动设计原理。

本书在编写过程中得到了各方面无私的帮助和支持，在此深表感谢！

厦门大学的陈辉煌老师一直都非常关心和支持本书的编写工作，本书的出版正是在他的大力支持与鼓励下完成的。

感谢厦门大学智能图像与信息处理实验室的洪景新、石江宏、周剑扬、汤碧玉、施海彬、吴晓芳、王飞舟、杨琦、何瑜红等老师以及海西工业技术研究院通信工程技术中心的卢敏、陈路、薛财锋、谢家隆、方志远、吴昕等，无论是平时的工作还是本书的编写，他们都给了我很多的帮助。

感谢厦门大学电子工程系和通信工程系的程恩、肖明波、黄联芬和施芝元，无论是课程建

设还是本书的出版，他们都给予了热情的帮助与支持。

感谢厦门大学智能图像与信息处理实验室的陈何杰、郑春芳、钟光清、林涛、谢思雄等同学，他们为本书的编写提供了许多帮助。

感谢中山大学的李晓宁老师，她为本书的编写提供了许多宝贵的意见。

感谢深圳亿道电子有限公司的张治宇、何章龙和喻继桑等，他们为本书编写提供了许多帮助。

感谢北京航空航天大学出版社的马广云老师、嵌入式系统编辑部主任胡晓柏先生以及其他相关工作人员，他们为本书的出版付出了辛勤的劳动。

本书受福建省科技项目“通信技术及产品”重大专项（项目编号：2007HZ0003）和福建省自然科学基金（项目编号：2009J01306）支持。

由于作者水平有限，许多问题还在摸索之中，书中难免有错误和不确切之处，敬请读者批评指证，作者的电子邮箱是 lxzheng@xmu.edu.cn。

作 者

2010 年 1 月于厦门大学

目 录

第 1 章 嵌入式系统接口技术概述 …… 1
1.1 嵌入式系统的架构与 I/O 接口的连接方式 …… 1
1.1.1 嵌入式系统的基本结构 …… 1
1.1.2 嵌入式 SOC 处理器与片上 I/O 接口 …… 2
1.1.3 嵌入式系统的 I/O 接口的扩展 …… 3
1.2 接口的基本功能 …… 4
1.3 I/O 接口及其编址方式 …… 5
1.4 I/O 接口扩展方法 …… 6
1.5 I/O 设备的控制方法 …… 12
第 2 章 Linux 内核设备驱动原理 …… 15
2.1 基本原理 …… 15
2.1.1 Linux 操作系统的架构 …… 15
2.1.2 嵌入式系统的开发模式与嵌入式 Linux 设备驱动 …… 16
2.1.3 Linux 设备的分类 …… 17
2.1.4 Linux 设备的标识 …… 18
2.2 内核模块 …… 18
2.2.1 什么是内核模块 …… 18
2.2.2 内核模块的框架 …… 19
2.2.3 内核模块的编译 …… 21
2.3 Linux 内核编程常见操作 …… 25
2.4 设备驱动程序的结构 …… 30
2.4.1 虚拟文件系统与硬件驱动的接口 …… 31
2.4.2 简单字符设备的驱动 …… 32
2.5 Linux 2.6 内核的中断处理 …… 37
2.5.1 Linux 中断处理流程 …… 37
2.5.2 外部中断的描述与处理 …… 41
2.5.3 中断处理程序的注册与释放 …… 45

2.5.4 中断处理程序的编写 …… 46
2.6 Linux 2.6 内核的工作推后执行的机制 …… 46
2.6.1 软中断 …… 47
2.6.2 Tasklet …… 47
2.6.3 工作队列 …… 49
2.7 Linux 2.6 内核设备模型 …… 52
2.7.1 Linux 2.6 设备模型概述 …… 52
2.7.2 内核设备驱动模型的组件 …… 54
2.7.3 sysfs 文件系统、udev 和 Linux 内核设备模型 …… 58
2.7.4 平台总线设备 …… 61
本章小结 …… 66
习题与思考题 …… 67
第 3 章 最小硬件系统设计与底层软件移植 …… 68
3.1 最小硬件系统设计 …… 68
3.1.1 系统存储器接口 …… 69
3.1.2 串行通信接口电路原理 …… 73
3.2 最小硬件系统的配置 …… 74
3.2.1 处理器的配置 …… 74
3.2.2 FLASH & SDRAM 的配置 …… 76
3.2.3 GPIO 和串口的配置 …… 79
3.3 最小硬件系统与 Bootloader …… 81
3.3.1 U-boot 启动阶段 1 的处理过程 …… 82
3.3.2 U-boot 启动阶段 2 的处理过程 …… 87
3.3.3 U-boot 移植原理 …… 88
3.3.4 基于 U-boot 的硬件调试 …… 94
3.4 Linux 2.6 内核移植原理 …… 96
3.4.1 外部中断初始化 …… 97
3.4.2 DMA 接口 …… 101
3.4.3 系统时钟接口 …… 103
3.4.4 片上设备 I/O 地址空间的静态映射 …… 106
3.4.5 片上 I/O 设备的定义 …… 108
3.5 最小硬件系统与 Linux 2.6 内核移植 …… 109
3.5.1 建立开发板平台描述文件 …… 109
3.5.2 编写硬件 include 文件 …… 111

3.5.3 修改内核配置文件 …… 111
本章小结 …… 112
习题与思考题 …… 112
第 4 章 小键盘接口设计与 Linux 驱动开发 …… 113
4.1 硬件原理 …… 113
4.1.1 接口设计 …… 114
4.1.2 电路原理 …… 114
4.2 软件驱动原理 …… 116
4.2.1 内核模块的加载和卸载函数 …… 117
4.2.2 虚拟文件系统与硬件驱动的接口 …… 118
4.2.3 设备打开操作接口函数 …… 118
4.2.4 设备读取操作接口函数 …… 119
4.2.5 设备关闭操作接口函数 …… 119
4.2.6 攫取键值子函数 …… 120
4.2.7 读缓冲区子函数 …… 121
4.2.8 定时器在程序查询式 I/O 控制方式中的应用 …… 122
4.2.9 利用等待队列实现阻塞型 I/O …… 123
4.2.10 poll()系统调用接口函数 …… 124
4.2.11 信号驱动的异步 I/O 操作的支持 …… 126
4.3 键盘信息读取应用程序 …… 127
4.3.1 打开键盘设备 …… 127
4.3.2 读取键值 …… 128
4.3.3 关闭键盘设备 …… 129
本章小结 …… 129
习题与思考题 …… 129
第 5 章 实时时钟接口与 Linux 驱动开发 …… 130
5.1 实时时钟接口电路设计 …… 130
5.1.1 处理器与 RTC－4513 接口设计 …… 130
5.1.2 RTC－4513 电路原理 …… 131
5.1.3 RTC－4513 操作参数 …… 134
5.1.4 RTC－4513 的串行操作流程 …… 136
5.2 RTC 软件驱动原理 …… 140
5.2.1 内核模块的加载和卸载 …… 140
5.2.2 虚拟文件系统与硬件驱动的接口 …… 140

5.2.3 设备打开操作接口函数 …… 141
5.2.4 ioctl 方法 …… 141
5.2.5 设备关闭操作接口函数 …… 145
5.2.6 读时钟寄存器子函数 …… 146
5.2.7 写时钟寄存器子函数 …… 146
5.3 RTC 操作应用程序 …… 147
5.4 基于 Linux 内核实时时钟子系统的 RTC 驱动 …… 149
5.4.1 RTC 设备驱动接口 …… 150
5.4.2 实时时钟子系统的 rtc - dev 模块与上层 API …… 152
本章小结 …… 154
习题与思考题 …… 154
第 6 章 触摸屏接口设计与 Linux 驱动开发 …… 156
6.1 触摸屏的工作原理 …… 156
6.2 ADS7843 触摸屏控制器简介 …… 157
6.3 处理器与 ADS7843 的接口设计 …… 160
6.3.1 接口电路设计 …… 160
6.3.2 Microwire 数据帧结构 …… 161
6.4 软件驱动原理 …… 162
6.4.1 Linux 输入设备子系统 …… 162
6.4.2 触摸屏硬件操控原理 …… 167
6.4.3 触摸屏驱动与输入设备子系统的接口 …… 169
6.4.4 中断处理 …… 173
6.5 基于触摸屏驱动的应用示例 …… 175
本章小结 …… 177
习题与思考题 …… 178
第 7 章 以太网电路设计与 Linux 驱动开发 …… 179
7.1 CS8900A 以太网芯片简介 …… 179
7.1.1 功能介绍 …… 179
7.1.2 引脚定义 …… 180
7.2 处理器与以太网接口电路设计 …… 180
7.3 CS8900A 片内寄存器介绍 …… 182
7.3.1 总线接口寄存器组 …… 182
7.3.2 状态与控制寄存器组 …… 183
7.3.3 发送初始化寄存器组 …… 191

7.3.4 地址过滤寄存器组 …… 192
7.4 CS8900A 的操作方法 …… 193
7.4.1 CS8900 的初始化 …… 193
7.4.2 CS8900A 的 I/O 模式寄存器 …… 194
7.4.3 读写 CS8900A 的片内寄存器 …… 194
7.5 软件驱动原理 …… 195
7.5.1 Linux 网络设备驱动框架 …… 195
7.5.2 Linux 网络协议栈与驱动间的接口 …… 196
7.5.3 Linux 网络设备接口 …… 201
7.5.4 CS8900A 驱动中的网络设备操作接口实现 …… 203
7.5.5 数据接收与中断处理 …… 207
7.5.6 利用平台总线设备封装网络驱动 …… 212
本章小结 …… 217
习题与思考题 …… 217
第 8 章 PCMCIA 外围电路设计和 Linux 驱动开发 …… 218
8.1 基于 PXA2XX 处理器的 PCMCIA 接口 …… 218
8.1.1 PCMCIA 和 CF 接口简介 …… 218
8.1.2 PCMCIA 存储器映射 …… 219
8.1.3 PCMCIA 外部接口设计 …… 220
8.2 外围电路驱动原理 …… 220
8.2.1 双向收发器 74LCX245 的使用 …… 220
8.2.2 CF 卡插入检测 …… 224
8.2.3 “卡读(写)使能”信号 …… 224
8.2.4 PCMCIA 接口的 Socket Select …… 224
8.2.5 PCMCIA 接口的电源管理 …… 226
8.3 GPIO 连接原理 …… 227
8.3.1 存储控制单元的 GPIO 连接原理 …… 227
8.3.2 与中断相关的 GPIO 连接原理 …… 228
8.4 PCMCIA 软件驱动原理 …… 229
8.4.1 PCMCIA 驱动程序体系 …… 229
8.4.2 PCMCIA 工作流程 …… 230
8.4.3 插口驱动设计 …… 231
8.5 PCMCIA Card Services 软件包 …… 233
8.5.1 软件包的安装 …… 233

8.5.2 特定卡驱动 …… 234
8.5.3 CF 存储卡的使用 …… 235
本章小结 …… 237
习题与思考题 …… 237
第 9 章 AC97 音频处理电路设计与 Linux 驱动开发 …… 238
9.1 PXA2XX AC97 硬件工作原理 …… 238
9.1.1 PXA2XX AC97 音频处理电路设计 …… 238
9.1.2 PXA2XX AC97 控制单元 …… 239
9.1.3 CS4299 音频编解码器 …… 245
9.2 Linux ALSA 音频设备驱动原理 …… 249
9.2.1 ALSA 简介 …… 249
9.2.2 ALSA 声卡驱动架构 …… 250
9.2.3 ALSA 驱动中的声卡描述对象与音频设备组件管理 …… 251
9.2.4 ALSA 驱动的 PCM 中间层 …… 254
9.2.5 ALSA 驱动的 AC97 编解码器中间层 …… 256
9.3 PXA2XX AC97 的 ALSA 驱动 …… 259
9.3.1 硬件初始化与声卡描述对象的创建和注册 …… 260
9.3.2 PCM 接口及其底层硬件操作接口的实现 …… 262
9.3.3 AC97 的底层硬件操作接口实现 …… 266
本章小结 …… 270
习题与思考题 …… 270
参考文献 …… 271

第1章

嵌入式系统接口技术概述

嵌入式系统是一种应用于特定领域的专用计算机系统，它的应用场合多种多样，所连接的外设种类繁多。这些设备不仅结构和工作原理不同，它们的连接方式也可能完全不同。为了便于实现嵌入式系统的处理器与各种外设的连接，避免嵌入式处理器陷入与各种外设打交道的沉重负担之中，通常需要使用 I/O 接口作为嵌入式处理器与外设之间信息交换的中间环节。这也意味着，I/O 接口是嵌入式系统中必不可少的重要组成部分，是嵌入式处理器与外部世界进行信息交换和传输的纽带与桥梁。

1.1 嵌入式系统的架构与 I/O 接口的连接方式

1.1.1 嵌入式系统的基本结构

嵌入式系统作为一种特殊形态的计算机系统，尽管其形态各异，应用场合也不尽相同，但各种嵌入式系统与其他的计算机系统有着相同的内涵。从硬件系统结构上看，嵌入式系统同普通的计算机系统一样，由处理器、存储器(RAM 和 ROM)和输入/输出设备组成。从逻辑上看，嵌入式系统和其他计算机系统一样，都是采用总线连接方式将系统的各个部件连接在一起，也就是通过地址总线、数据总线和控制总线将计算机系统的处理器、存储器和外部 I/O 设备等各个部件连接在一起，并以总线作为处理器与其他部件之间信息传输的公共通道。这种以总线连接方式构成的计算机系统结构如图 1-1 所示。

在计算机系统的这种结构中，处理器是整个系统的核心，数据的存取和计算以及外部 I/O 设备的操作和控制工作都由处理器承担。采用总线结构能使计算机系统的构造较为方便和灵活，但计算机系统中的各种外设千差万别，它们的控制方式、处理速度和信息输入/输出的格式各不相同。这些种类繁多的外设显然不可能直接通过系统总线与处理器连接，通常它们都需

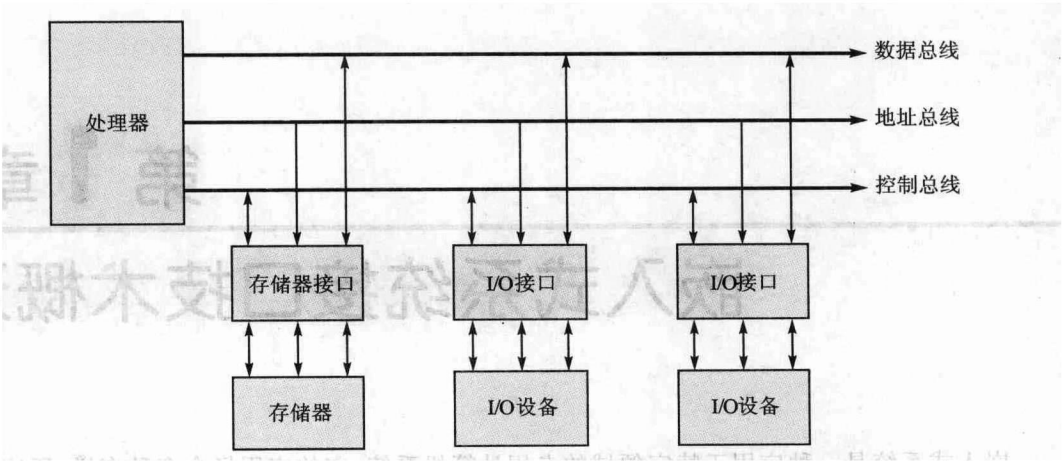


图 1-1　计算机系统的总线结构

通过适当的接口与处理器进行联系。I/O 接口包括将外设连接到处理器系统总线上的一组硬件逻辑电路(硬件接口,在讨论硬件设计时接口多指的是硬件接口)及其相应的软件控制程序。

1.1.2　嵌入式 SOC 处理器与片上 I/O 接口

在通用的计算机系统中,一个外设的 I/O 接口的功能通常由一个独立模块完成。嵌入式系统与通用计算机系统有所不同,嵌入式系统通常采用 SOC(System on Chip)芯片来构成系统的核心。这类芯片的内部采用片上总线将微处理器与外部接口模块连接起来,在一块芯片上集成了微处理器和多种外部接口,构成一个片上计算机系统。一个典型的嵌入式 SOC 处理器的架构如图 1-2 所示,该 SOC 处理器芯片包括一个处理器核心、片上总线以及各种片上 I/O 接口(LCD 接口、USB 接口、存储器接口、硬件接口、串口等)。该 SOC 处理器用片上总线将处理器核心与各种外设连接起来。

嵌入式处理器是嵌入式系统的核心部件,相应地,处理器核则是整个 SOC 芯片的核心。目前 32 位嵌入式 SOC 处理器大多采用 RISC 指令系统和哈佛体系结构。它们的架构主要有 ARM、MIPS、PowerPC、SPARC、ColdFire 等,这其中,又以 ARM 架构的处理器所占市场份额最大。ARM 处理器是种典型的 RISC 处理器,它采用分离的指令高速缓存 I-Cache 和数据高速缓存 D-Cache 以实现哈佛体系结构。ARM 指令集体系结构定义了 V1～V7 共七个版本。目前广泛使用的 ARM 处理器主要采用 ARM V4 以上版本的指令集,其主要的产品系列有 ARM7、ARM9、ARM9E、ARM10、ARM11、Cortex-A/R/M 以及 Intel/Marvell 公司的 XScale 系列处理器。

在嵌入式 SOC 处理器中,片上总线将片上的各个部件互连在一起,它对于整个 SOC 处理

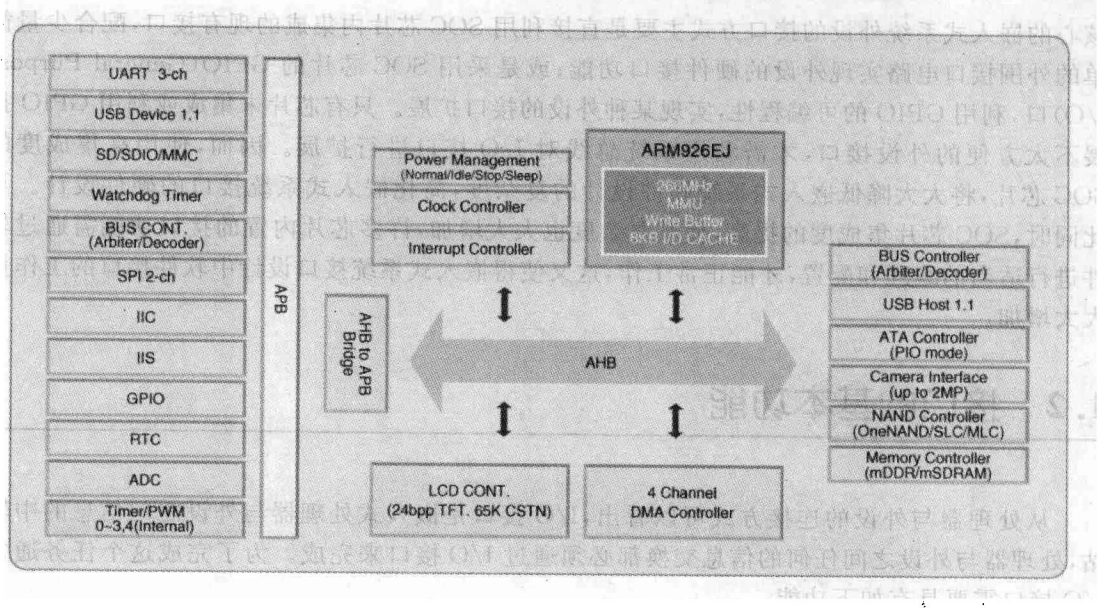


图 1-2　SOC 处理器的芯片架构

器能否正常工作起着至关重要的作用。目前在 SOC 芯片中使用的片上总线主要有 ARM 公司的 AMBA 总线、Silicore 公司的 WishBone 总线、Altera 公司的 Avalon 总线以及 IBM 公司的 CoreConnect 总线等。AMBA 总线因 ARM 处理器的巨大使用量成为一种广泛使用的标准片上总线。AMBA 总线规范主要包括 AHB(Advanced High performance Bus)系统总线、ASB(Advanced System Bus)系统总线和 APB(Advanced Peripheral Bus)外围总线。AHB/ASB 总线主要用于高性能模块之间的连接，如处理器、DMA 和存储控制器等；APB 总线主要用于低速的外设之间的连接，如串口、IIC、SPI 和 GPIO 等。AHB/ASB 总线与 APB 总线之间使用 APB 桥连接。

嵌入式 SOC 芯片在片上集成了多种 I/O 设备或 I/O 设备接口，根据芯片所面向的市场，在芯片上集成所需的外设及其接口。SOC 芯片所集成的外设接口决定了芯片的功能，芯片的集成度越高，芯片的功能就越丰富。最常见的片上外设与接口有存储控制器、片上 SRAM、片上 FLASH、串口、USB 接口、GPIO、RTC、IIC 总线接口和 SPI 总线接口等，对于一些面向移动通信市场的嵌入式 SOC 处理器常常还会集成 LCD 控制器、CMOS 摄像头、AC97 音频控制器、红外通信接口和蓝牙接口等。

1.1.3　嵌入式系统的 I/O 接口的扩展

由于嵌入式 SOC 处理器在一颗芯片上集成了多种 I/O 设备接口，这使得以 SOC 芯片为

核心的嵌入式系统外设的接口方式主要是直接利用SOC芯片内集成的现有接口，配合少量简单的外围接口电路实现外设的硬件接口功能；或是采用SOC芯片的GPIO(General Purpose I/O)口，利用GPIO的可编程性，实现某种外设的接口扩展。只有芯片未集成或利用GPIO扩展不太方便的外设接口，才需利用系统总线对I/O接口进行扩展。因而，使用高集成度的SOC芯片，将大大降低嵌入式系统硬件设计的复杂度，简化嵌入式系统接口的硬件设计。与此同时，SOC芯片集成度的提高，使其复杂度也大大增加，许多芯片内置的接口功能需通过软件进行适当的编程和配置，才能正常工作，这又使得嵌入式系统接口设计中软件接口的工作量大大增加。

1.2 接口的基本功能

从处理器与外设的连接方式可以看出，I/O接口是嵌入式处理器与外设之间信息的中转站，处理器与外设之间任何的信息交换都必须通过I/O接口来完成。为了完成这个任务通常I/O接口需要具有如下功能。

1. 数据传输与缓冲功能

I/O接口作为处理器与外设之间信息交换的桥梁，需要具有数据传输通道，同时为了完成数据的传输，接口还需具备数据缓冲功能。通常处理器的速度较高，一些常见的ARM处理器主频高达几百兆赫兹，而许多外设的处理速度较低，例如，一些常见嵌入式SOC芯片集成的RS-232串口通信速率最高也只有115.2 kb/s。为了解决高速的处理器与低速的外设之间存在速度不匹配引起的通信矛盾，避免丢失数据，就需要I/O接口具有数据缓冲的功能，用于保证高速处理器与低速I/O设备之间数据传送能够保持同步。I/O接口中低速外设单字节数据的缓冲通常可以用数据缓冲寄存器或锁存器实现，对于间断性瞬时高速数据的缓冲则可以使用FIFO缓冲芯片实现，对于数据量较大的数据缓冲则可以使用双口SRAM实现。对于大数据量的数据传输，在现代的嵌入式系统和通用计算机系统一样，都可以采用DMA(Direct Memory Access)技术，在存储器与外设之间直接进行数据传送。

2. 设备寻址与操控功能

由于系统中的所有外设都挂接在系统总线上，为此，接口电路中需包含地址译码电路，以使处理器输出该设备地址时，接口能识别出自身的地址已被选中。通常硬件接口电路中会有一些寄存器，用于存储设备的控制信息和状态信息，这些寄存器每个都对应有一个唯一的地址。处理器可以通过寄存器的地址访问相应的寄存器。当处理器向控制寄存器写入操纵设备的控制信息后，I/O设备接口电路对寄存器中的控制命令进行译码，产生控制信号，使I/O设备完成相应的操作；I/O设备的状态信息则保存在接口电路的状态寄存器中，处理器可以通

过读取状态寄存器的内容，了解I/O设备的当前状态。此外，现代的嵌入式系统与通用计算机系统一样，大多采用了中断技术。在接口电路中可以使用中断触发器，当设备处于处理器感兴趣的某一状态时，主动向处理器发出中断请求，通知处理器设备已处于处理器感兴趣的状态。

3. 信号格式与数据格式转换功能

外设的输入输出信号和所需的控制信号常与处理器的工作信号不匹配，因此I/O接口必须具有信号格式转换的功能，这些信号转换功能包括处理器与设备之间的逻辑关系匹配、时序匹配、电平匹配以及A/D和D/A信号转换等。此外，处理器处理的数据宽度由数据总线的宽度决定，通常32位处理器处理的数据为32位宽。这就意味着，其处理的数据宽度最大不能超过32位，若I/O设备的数据宽度超过32位则需将其转换为32位宽才能传给处理器；反之若I/O设备并行传送的数据宽度不足32位，处理器以32位数据宽度接收数据时则需将其无效位截断。由于外设的数据格式多种多样，有串行的，也有并行的，且其数据的位宽也不一致，因而接口电路需要具有串行与并行数据间的转换能力以及数据宽度的变换功能等。

1.3 I/O接口及其编址方式

通常I/O接口中包含一些处理器能够进行读写操作的寄存器，这些寄存器称为I/O接口。每个I/O接口都对应着一个唯一的端口地址（地址有可能重用，也就是同一个的地址可能对应着多个不同功能的端口），即相应端口寄存器的地址。处理器能通过这些端口地址访问I/O接口，读取设备的当前状态，发出设备的控制命令，从设备读取数据信息或向设备写入数据信息。根据I/O接口所承担功能的不同，可将它们分为控制端口（保存控制命令的寄存器）、状态端口（存放设备状态信息的寄存器）和数据端口（存放输入/输出数据的寄存器）。

处理器访问I/O接口时需对I/O接口进行寻址操作。处理器对I/O接口的寻址方式与I/O接口地址的编址方式有关。通常编址方式有统一编址方式（存储器映射方式）和独立编址方式两种。

1. 统一编址方式

在统一编址方式下，I/O接口被当成存储器单元对待，与存储器单元在同一线性地址空间统一编址，这种方式有点类似于地址映射，故又称为存储器映射编址方式。目前大多数嵌入式处理器采用这种编址方式。在这类系统中，处理器对I/O接口的访问较简单，可以采用同访问存储器一样的方式进行I/O接口的访问，不需要使用专门的I/O指令。但这种编址方式需占用存储器地址空间，减少了可用的存储器地址范围，对存储器的扩充有潜在的影响。此外，在这种编址方式中对存储器和I/O接口的访问使用相同的指令，因而很难仅从指令上对二者