

原创经典，嵌入式系统一线工程师倾力打造

深入Linux系统底层和内核，透析ARM嵌入式Linux系统开发的本质
通过嵌入式系统开发实例，归纳ARM嵌入式Linux系统开发的整个过程

ARM嵌入式 Linux系统开发技术详解

Embedded Linux System Development for ARM

杨水清 张剑 施云飞 等编著



- ◎ 内容循序渐进，从硬件平台选择、开发环境构建、Linux移植到软硬件协同调试等一气呵成
- ◎ 介绍了多种工具软件，如ADS、VIM、GCC、GDB、MAKE、Eclipse等
- ◎ 紧跟ARM嵌入式系统开发技术步伐，深入讲解嵌入式Linux的多任务编程、网络编程等
- ◎ 通过实例讲解CAN总线、SD卡、网络设备等驱动程序的具体开发过程
- ◎ 详细讲解开发嵌入式B超综合案例，总结实际项目开发过程



CD-ROM



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

ARM嵌入式 Linux系统开发技术详解

Embedded Linux System Development for ARM

杨水清 张剑 施云飞 等编著

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书由浅入深、通俗易懂地讲解了嵌入式 Linux 的系统设计与开发。全书共 25 章，从嵌入式处理器 ARM 开始，讲解了 ARM 处理器的资源、ARM 的指令集、ADS 开发工具、嵌入式系统硬件环境的构建、Boot Loader、Linux 内核移植、嵌入式文件系统、嵌入式 Linux 多任务程序开发、嵌入式 Linux 设备驱动开发、嵌入式 Linux 网络程序开发、MiniGUI 图形界面开发、设备驱动开发案例、综合案例等内容。书中通过大量的例程来讲解知识要点，并提供了大量极有参考价值的开发案例，读者可以通过这些例程和开发案例对嵌入式 Linux 开发有一个系统的学习和提高。

本书共有四个方面的内容。在嵌入式系统的硬件结构中讲述了嵌入式处理器 ARM 的特点、嵌入式系统硬件环境的构建和 ADS 开发工具的使用。在嵌入式 Linux 系统移植中讲述了目标板软件环境的构建，主要包括 Boot Loader、Linux 内核、文件系统以及交叉开发环境的构建。在嵌入式 Linux 软件开发中讲述了嵌入式 Linux C 语言开发工具的使用、标准库的使用、多任务开发基础和设备驱动开发基础。在嵌入式应用系统实例分析中讲述了嵌入式 Linux 的网络程序开发、MiniGUI 图形界面开发、CAN 总线设备驱动设计、DM9000 网络驱动设计、SD 卡驱动设计和嵌入式 B 超系统设计。

本书语言通俗易懂，内容丰富，注重理解与实例，知识涵盖面广。非常适合从事嵌入式 Linux 系统开发的初级工程师、高校学生、Linux 程序开发人员阅读和学习。

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

图书在版编目 (CIP) 数据

ARM 嵌入式 Linux 系统开发技术详解 / 杨水清等编著. —北京: 电子工业出版社, 2008.11
ISBN 978-7-121-07469-1

I. A… II. 杨… III. ①微处理器, ARM—系统设计 ②Linux 操作系统—系统设计 IV. TP332 TP316.89

中国版本图书馆 CIP 数据核字 (2008) 第 151614 号

责任编辑: 高洪霞

印 刷: 北京东光印刷厂

装 订: 三河市皇庄路通装订厂

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本: 787×1092 1/16 印张: 40 字数: 1095 千字

印 次: 2008 年 11 月第 1 次印刷

印 数: 4000 册 定价: 75.00 元 (含光盘 1 张)

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

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

服务热线: (010) 88258888。

前 言

你想知道手中的 MP4 由什么组成吗？

你想亲手制作一个电子相框吗？

你想深入研究操作系统内核，学习编写驱动程序吗？

你想为嵌入式系统设计功能强大的图形界面吗？

你想知道一个真实的嵌入式系统是如何从无到有建立起来的吗？

本书将从作者亲身开发的经验带给你详细实用的解答！

嵌入式 Linux 开发的前景

嵌入式产业发展迅猛，已成为计算机体系的重要组成部分。嵌入式系统产品正不断渗透各个行业，其产业增幅也不断加大。在计算机后 PC 技术时代，嵌入式系统将拥有最大的市场。任何一个普通人都有可以拥有从小到大的各种使用嵌入式技术的电子产品，小到手机、MP3 等，大到智能家电、车载电子设备等。在工业和服务领域中，使用了嵌入式技术的数字机床、工业机器人、安全系统等也在逐渐改变传统工业和服务方式。正是因为有如此大的需求，所以促使笔者结合多年的开发经验，编写了本书。

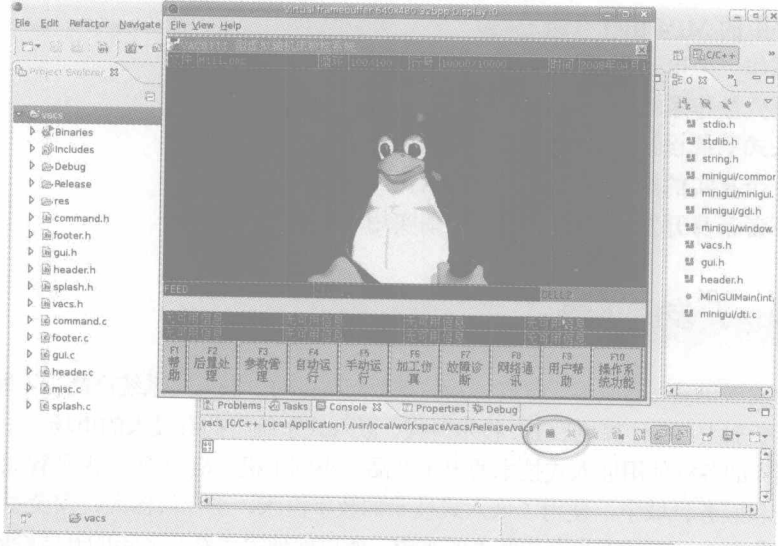
本书特点

1. **注重读者的学习与接受情况：**在嵌入式 Linux 系统下进行开发，一个很大的困难就是各种工具纷繁复杂，不知从何入手。本书在每个部分都安排了一章，引导读者一步一步熟悉各种开发工具和环境，使读者首先建立起感性认识，为进一步深入学习打下基础。

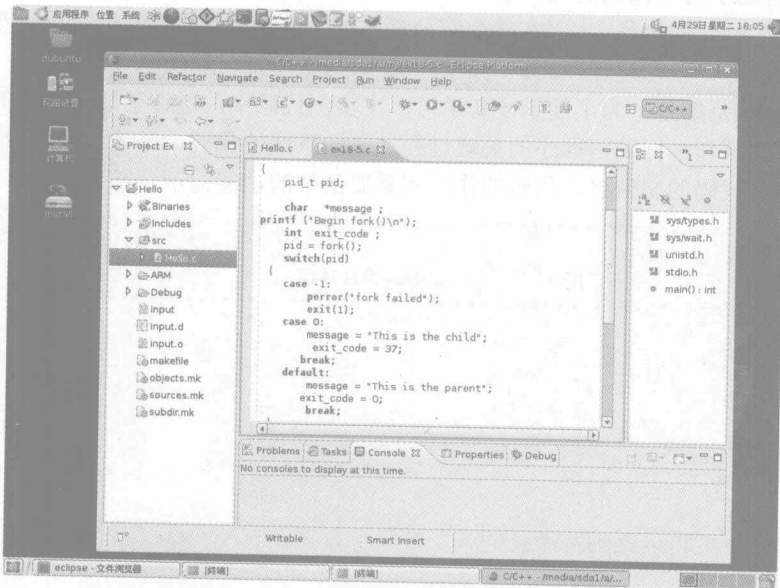
2. **代码注释规范细致：**为了便于读者学习，在实例代码中，进行了非常详细的注释。这些注释可以引导读者理解和掌握编写程序的关键过程。例如下面的代码，是 C 语言标准库章节中的实例。在实例中，对代码的文件名、代码的作用和重要函数的功能提示，都体现在了代码注释中。

```
/******  
/*文件: program_16_3.c  
/*简介: 将字符串 a,b,c 分别采用 10, 2, 16 进制转换成数字  
/******  
#include <stdio.h>  
#include <stdlib.h>  
  
int main()  
{  
    char *a="1000000000";  
    char *b="1000000000";  
    char *c="ffff";  
    /*将字符串 a 采用 10 进制转换成数字*/  
    printf("a=%d\n",strtol(a,NULL,0));  
    /*将字符串 b 采用 2 进制转换成数字*/  
    printf("b=%d\n",strtol(b,NULL,2));  
    /*将字符串 c 采用 16 进制转换成数字*/  
    printf("c=%d\n",strtol(c,NULL,16));  
    return 0;  
}
```

3. **Eclipse 集成开发环境**: 长期以来 Linux 开发没有统一的开发平台, 大多数的开发者都直接使用 VI、GCC 等命令行工具, 这样其实不利于大型项目的开发, 这也是很多开发者感到遗憾的地方。笔者经过多年的比较, 引入了 Eclipse 作为进行嵌入式 Linux 开发的集成开发环境, 取得了很好的效果 (如下图所示), 因此把它介绍给读者。

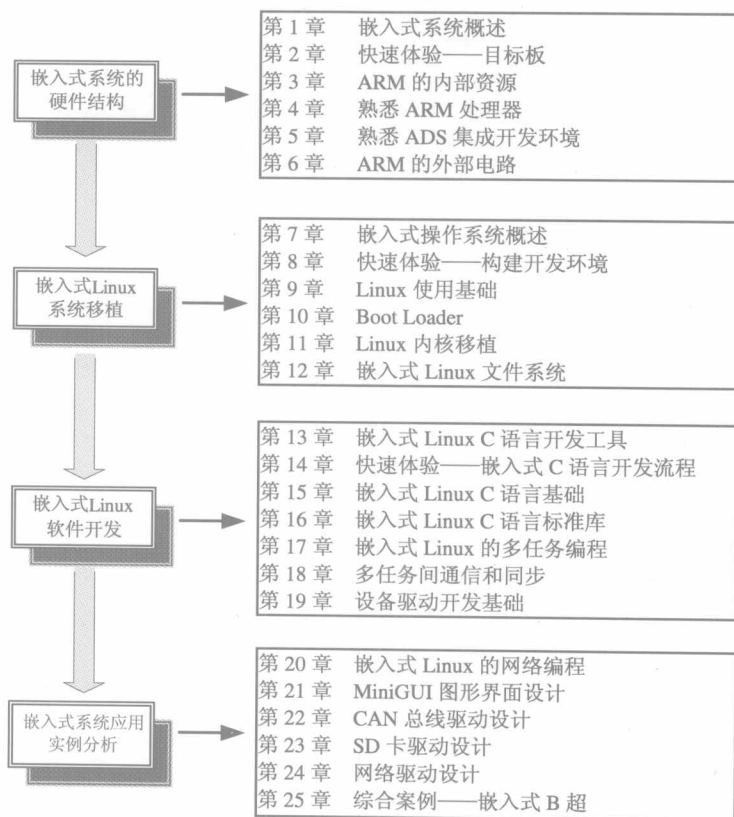


4. **起点高、内容新**: 市场上大多介绍嵌入式 Linux 开发的同类书籍都是基于 Linux 2.4 内核的, 现在 2.6 版本的内核已经十分成熟了, 取代 2.4 版本的内核指日可待。而且, 与之配套的还需要基于 2.6 内核的 Linux 开发平台。因此笔者选择了基于 2.6 内核的 Ubuntu 系统、arm-linux-gcc 3.4 作为开发平台 (如下图所示), 介绍给读者, 帮助读者抢占嵌入式 Linux 开发领域的制高点。



5. **内容深入**: 本书在注重基础和实际的同时, 注重知识的扩充。设备驱动开发、网络程序设计和图形界面编程是在实际中应用极为广泛的内容, 本书中对这些知识的讲述可以使读者对嵌入式 Linux 系统的开发有一个从书本到实践的本质上的提高。

本书包括的内容



本书适合的读者

- 嵌入式Linux开发入门者；
- 高校计算机、电子专业学生；
- 高校毕业设计的学生；
- 嵌入式Linux爱好者；
- 嵌入式Linux相关学习和研究的研究生；
- 嵌入式Linux程序员。

致谢

感谢吴伟的通力合作，精心撰写了第11章、第15章和第19章。
感谢我们的家人（唐颖、俞秋华），是你们的支持使我们坚持写完本书。
感谢所有开放源码软件和开发文档的作者们。

目 录

Contents

第 1 章 嵌入式系统概述

1

本章介绍了嵌入式系统的一些基础知识,解释了什么是嵌入式系统、嵌入式系统中的处理器以及嵌入式系统中的软件系统。在学习了本章的内容以后,读者会对嵌入式系统有个整体的印象。本章将读者引入嵌入式系统开发的大门,从下一章开始,将逐步学习嵌入式系统的开发过程。

1.1 嵌入式系统简介	1	1.2.3 数字信号处理器	5
1.1.1 嵌入式系统	1	1.2.4 嵌入式片上系统	5
1.1.2 嵌入式系统的特点	2	1.3 嵌入式系统中的软件系统	5
1.1.3 嵌入式系统的发展趋势	2	1.3.1 嵌入式系统中的软件系统	5
1.2 嵌入式系统中的处理器	4	1.3.2 嵌入式系统软件开发的一般过程	6
1.2.1 微处理器	4	1.3.3 嵌入式应用程序的开发	6
1.2.2 微控制器	4	1.4 本章小结	6

第 2 章 快速体验——目标板

7

本章首先介绍了目标板的结构,然后带着读者对该目标板做了一番初次体验。包括软件的安装和简单使用,以及在 Linux 下体验 Telnet 和 ftp 命令。希望通过本章的讲解,使读者对嵌入式系统开发有一个初步的、感性的认识。

2.1 目标板结构	7	2.2.3 安装 Multi-ICE 和配置 AXD	10
2.2 ARM 初体验	8	2.2.4 ADS 的简单使用	12
2.2.1 测试 ARM 处理器	8	2.2.5 使用 Telnet 和 ftp	15
2.2.2 安装 ADS 1.2	9	2.3 本章小结	16

第 3 章 ARM 的内部资源

17

本章对 S3C2440 嵌入式处理器片内资源的基本特性及功能进行了简单的描述,希望读者通过对本章的阅读,能对 S3C2440 嵌入式处理器的片内资源有一个比较全面的了解和掌握,但在此处不过于深入讨论每一个功能模块的具体细节。如果读者有兴趣对 S3C2440 的各个功能模块进行详细的了解,可以参考相关用户手册。

3.1 S3C2440 微处理器	17	3.3.4 脉冲带宽调制定时器 (PWM)	23
3.1.1 主要结构	17	3.3.5 实时时钟 (RTC)	23
3.1.2 片内资源	18	3.3.6 通用 I/O 端口	24
3.1.3 体系结构	18	3.3.7 LCD 控制器	24
3.2 S3C2440 存储器映射	18	3.3.8 UART 控制器	25
3.2.1 bank0 总线宽度	18	3.3.9 A/D 转换和触摸屏接口	25
3.2.2 nWAIT 引脚的作用	19	3.3.10 看门狗定时器	26
3.2.3 nXBREQ/nXBACK 引脚操作	19	3.3.11 IIC 总线接口	26
3.3 S3C2440 内部资源详解	20	3.3.12 AC97 音频解码器接口	27
3.3.1 Cache 高速缓存	20	3.3.13 USB 设备控制器	27
3.3.2 时钟和电源管理	20	3.3.14 SD 接口	27
3.3.3 中断控制器	22	3.3.15 SPI 接口	28

3.3.16 相机接口	28	3.4 本章小结	29
3.3.17 工作电压	29		

第4章 熟悉 ARM 处理器

30

本章首先介绍了 ARM 公司的历史,并列出了目前使用比较流行的 ARM 处理器系列,读者可以根据不同的需要做出自己的选择。接下来详细介绍了 ARM 的指令集。希望读者通过本章的学习,能对 ARM 处理器有一个大体的认识,熟悉 ARM 的指令集并在实际开发中学习和应用。

4.1 为什么用 ARM	30	4.7.1 立即寻址	36
4.2 ARM 公司简介	30	4.7.2 寄存器寻址	36
4.3 ARM 微处理器系列	30	4.7.3 寄存器间接寻址	36
4.3.1 ARM7 微处理器	30	4.7.4 基址变址寻址	36
4.3.2 ARM9 微处理器	31	4.7.5 多寄存器寻址	37
4.3.3 ARM10 微处理器	31	4.7.6 相对寻址	37
4.3.4 ARM11 微处理器	32	4.7.7 堆栈寻址	37
4.4 ARM 微处理器的结构	32	4.8 ARM 指令集详解	37
4.4.1 体系结构	32	4.8.1 跳转指令	38
4.4.2 寄存器结构	33	4.8.2 数据处理指令	38
4.4.3 指令结构	33	4.8.3 乘法指令与乘加指令	42
4.5 ARM 微处理器的选择	33	4.8.4 程序状态寄存器访问指令	44
4.5.1 内核的选择	33	4.8.5 加载/存储指令	45
4.5.2 工作频率的选择	33	4.8.6 批量数据加载/存储指令	47
4.5.3 芯片内存存储器的选择	33	4.8.7 数据交换指令	47
4.5.4 片内外围电路的选择	34	4.8.8 移位指令	48
4.6 ARM 的指令集概述	34	4.8.9 协处理器指令	49
4.6.1 ARM 微处理器的指令分类和格式	34	4.8.10 异常产生指令	51
4.6.2 指令的条件域	35	4.9 本章小结	51
4.7 ARM 指令的寻址方式	36		

第5章 熟悉 ADS 集成开发环境

52

本章介绍了 ADS 集成开发环境。通过具体的例子,展示了 ADS 的使用过程和主要功能。希望读者通过本章的学习,能够初步学会使用 ADS,帮助自己学习和开发 ARM 系统。

5.1 命令行开发工具	52	5.2.1 建立一个工程	58
5.1.1 armcc 介绍	52	5.2.2 编译和链接工程	60
5.1.2 armcc 用法详解	53	5.2.3 target 设置选项	60
5.1.3 armlink 介绍	54	5.2.4 Language Settings	61
5.1.4 armlink 用法详解	54	5.2.5 Linker 设置	62
5.1.5 ARM 运行时库	54	5.2.6 ARM fromELF 工具	63
5.1.6 CodeWarrior 集成开发环境	55	5.2.7 命令行下编译工程	64
5.1.7 ADS 调试器	56	5.3 使用 AXD 调试代码	65
5.1.8 实用程序	57	5.3.1 打开调试文件	65
5.1.9 支持的软件	57	5.3.2 查看存储器内容	65
5.2 使用 ADS 创建工程	57	5.3.3 设置断点	66

5.3.4 查看变量值	67	5.4 本章小结	67
-------------------	----	----------------	----

第6章 ARM 的外部电路

68

本章详细介绍了基于 S3C2440 处理器的基本应用电路图。这些电路在 S3C2440 微处理器的系统设计中会经常使用到,读者也可以根据自身的实际需求,做必要的修改。

6.1 核心板电路	68	6.2.2 串口电路	74
6.1.1 晶振电路	68	6.2.3 USB 接口	74
6.1.2 复位电路	69	6.2.4 以太网接口	76
6.1.3 启动配置电路	69	6.2.5 JTAG 调试接口	76
6.1.4 FLASH 接口	69	6.2.6 音频接口	76
6.1.5 SDRAM 接口	70	6.2.7 LCD 接口	77
6.2 底板电路	73	6.2.8 SD 卡接口	78
6.2.1 电源电路	73	6.3 本章小结	79

第7章 嵌入式操作系统概述

80

本章从操作系统的一般特性讲起,介绍了操作系统的进程管理、存储管理、文件管理和设备管理等基本功能。然后对嵌入式操作系统与一般操作系统的特性进行了区别,介绍了嵌入式操作系统的特点、发展和分类。最后介绍了目前常见的一些嵌入式操作系统及它们的特点和主要的应用领域。

7.1 操作系统的结构和功能	80	7.5.3 中断处理	93
7.2 进程管理	81	7.5.4 设备驱动程序	94
7.2.1 进程的描述	82	7.6 嵌入式操作系统的特点	95
7.2.2 进程的调度	82	7.6.1 嵌入式操作系统的发展	95
7.3 存储管理	83	7.6.2 嵌入式操作系统的优势	96
7.3.1 存储器的体系结构	83	7.6.3 嵌入式操作系统的分类	96
7.3.2 内存管理的基本概念	84	7.7 常见的嵌入式操作系统	97
7.3.3 连续分配存储管理方式	84	7.7.1 VxWorks	97
7.3.4 页式存储管理方式	86	7.7.2 pSOS	97
7.4 文件管理	87	7.7.3 Palm OS	98
7.4.1 文件	88	7.7.4 QNX	98
7.4.2 目录	89	7.7.5 Windows CE	99
7.4.3 EXT2 文件系统	89	7.7.6 μ C/OS-II	100
7.5 设备管理	92	7.7.7 嵌入式 Linux	100
7.5.1 设备的分类	92	7.8 本章小结	101
7.5.2 数据传输控制方式	93		

第8章 快速体验——构建开发环境

102

本章主要介绍了嵌入式开发环境的构建过程,首先对交叉开发的概念进行了介绍,说明了在嵌入式系统的开发中采用交叉开发的原因。然后分别介绍了主机环境和目标板系统环境的构建过程,同时对这两个环境的连接方式也做了简要的介绍。

最后介绍了目标板系统的三个主要方面:Bootloader、Kernel 和根文件系统,说明了它们的作用和构建方法,同时举例说明了一些常用配置的配置方法。

8.1 交叉开发环境介绍	102	8.2 主机与目标板的连接方式	103
--------------------	-----	-----------------------	-----

8.2.1 串口通信接口	103	8.3.3 Tftp 服务的安装配置	109
8.2.2 以太网接口	104	8.3.4 NFS 的安装配置	110
8.2.3 USB 接口	104	8.3.5 建立交叉工具链	112
8.2.4 JTAG 接口	105	8.4 启动目标板系统	116
8.3 建立主机开发环境	105	8.4.1 Bootloader 和 Kernel	117
8.3.1 Ubuntu 6.06 的安装	105	8.4.2 根文件系统	118
8.3.2 Minicom 的安装配置	107	8.5 本章小结	119

第 9 章 Linux 使用基础

120

Linux 的命令行博大精深,有数不清的内容等待读者去探索。本章仅就一些最基本的知识做了最简要的介绍。另外由于 Linux 中的命令非常多,要全部介绍也是几乎不可能的。因此,本章按照命令的用途进行了分类讲解,力求举一反三。通过本章的介绍,相信读者可以很快地掌握 Linux 命令行的基本操作,并熟练掌握常用的命令。

9.1 Linux 的基本概念	120	9.2.3 重定向符号	125
9.1.1 文件	120	9.2.4 获取帮助	125
9.1.2 目录	120	9.3 Linux 的常用命令	127
9.1.3 分区	121	9.3.1 文件管理	127
9.1.4 挂载	122	9.3.2 内容管理	130
9.1.5 用户系统	122	9.3.3 权限管理	131
9.1.6 用户权限	122	9.3.4 备份压缩	132
9.1.7 shell	123	9.3.5 系统设置	133
9.1.8 环境变量	124	9.3.6 进程控制	136
9.2 Linux 的命令行	124	9.3.7 网络设置	138
9.2.1 执行命令	124	9.4 本章小结	139
9.2.2 参数	125		

第 10 章 Boot Loader

140

本章介绍了 Boot Loader 的概念、基本结构,并对 Boot Loader 的一般工作原理进行了介绍,重点介绍了 vivi 和 U-Boot 的开发调试及使用。通过学习这两种常见的 Boot Loader,可以使读者实际接触到 Boot Loader 的工作原理和代码实现。

10.1 Boot Loader 的概念	140	10.3.3 vivi 启动的第二阶段	154
10.1.1 Boot Loader 所支持的嵌入式体系	141	10.4 vivi 的基本命令	158
10.1.2 Boot Loader 的安装位置	141	10.4.1 mem 命令	158
10.1.3 Boot Loader 的启动过程	142	10.4.2 load 命令	159
10.1.4 Boot Loader 与主机的通信	142	10.4.3 part 命令	159
10.1.5 Boot Loader 的操作模式	142	10.4.4 param 命令	160
10.2 Boot Loader 的基本结构	143	10.4.5 boot 命令	160
10.2.1 Boot Loader 的 stage1	143	10.4.6 go 命令	161
10.2.2 Boot Loader 的 stage2	144	10.4.7 bon 命令	161
10.3 vivi 简介	149	10.4.8 reset 命令	161
10.3.1 vivi 的体系架构	149	10.4.9 help 命令	162
10.3.2 vivi 启动的第一阶段	150	10.5 U-Boot 简介	162

10.5.1 U-Boot 的特点	162	10.6.2 数据通信	167
10.5.2 U-Boot 的目录结构	163	10.6.3 存储器操作	168
10.5.3 U-Boot 的启动过程	163	10.6.4 系统引导	168
10.5.4 U-Boot 的移植	165	10.6.5 其他	169
10.6 U-Boot 的基本命令	167	10.7 本章小结	169
10.6.1 设置环境变量	167		

第 11 章 Linux 内核移植 170

本章给读者讲解了 Linux 内核移植中需要的基本知识,包括:Linux 内核的 5 个子系统及其相互关系;Linux 内核源码各目录包含模块及其功能;Linux 内核编译的方法和步骤;Linux 内核编译时的详细配置指南;Linux 内核下载和调试的相关知识。其中需要初学者熟练掌握 Linux 编译的方法和步骤,并对其他相关知识有所了解。

11.1 Linux 内核结构	170	11.4.2 模块和块设备层	182
11.2 Linux 源码结构	171	11.4.3 CPU 类型	183
11.2.1 arch 目录	172	11.4.4 电源管理	185
11.2.2 drivers 目录	172	11.4.5 总线和网络	187
11.2.3 fs 目录	173	11.4.6 驱动	188
11.2.4 其他目录	174	11.4.7 文件系统	195
11.3 内核编译	175	11.4.8 其他	198
11.3.1 编译准备	175	11.5 下载内核	200
11.3.2 设置 flash 分区	176	11.6 内核调试	201
11.3.3 配置内核	178	11.6.1 内核调试步骤	201
11.4 内核配置选项	180	11.6.2 常见内核问题	202
11.4.1 常规设置	180	11.7 本章小结	203

第 12 章 嵌入式 Linux 文件系统 204

本章首先介绍了嵌入系统中两种主要的存储介质 NAND Flash 和 NOR Flash,对它们的特点和区别进行了阐述。然后介绍了常见的适用于嵌入式系统的 5 种文件系统:CramFS 文件系统、YAFFS 1/2 文件系统、JFFS 1/2 文件系统、Ramdisk 文件系统和 TmpFS/RamFS 文件系统,分析了这些文件系统的基本结构和实现原理,介绍了各种文件系统及相应的映像文件的制作方法。最后针对文件系统的选择和设计提出了参考意见和方法。

12.1 嵌入式文件系统基础	204	12.2.6 CramFS 文件系统的挂载流程	216
12.1.1 NOR 型 Flash 存储器	204	12.3 YAFFS 文件系统	216
12.1.2 NAND 型 Flash 存储器	205	12.3.1 YAFFS 文件系统的数据存储方式	217
12.1.3 MTD 简介	206	12.3.2 YAFFS 文件系统的工作原理	218
12.1.4 日志型文件系统	207	12.3.3 YAFFS 文件系统对 MTD 的依赖性	219
12.1.5 BusyBox	208	12.3.4 YAFFS 文件系统驱动的安装流程	221
12.2 CramFS 文件系统	210	12.3.5 YAFFS 文件系统的制作	222
12.2.1 CramFS 文件的特性	210	12.4 JFFS 文件系统	223
12.2.2 CramFS 文件系统映像文件的结构	211	12.4.1 JFFS1 文件系统简介	224
12.2.3 CramFS 文件系统的工作原理	211	12.4.2 JFFS2 文件系统简介	225
12.2.4 CramFS 文件系统的初始化过程	212	12.4.3 JFFS3 文件系统简介	227
12.2.5 CramFS 文件系统的制作	213	12.4.4 JFFS2 文件系统的工作原理	227

12.4.5 JFFS2 文件系统的制作	230	12.6 嵌入式文件系统的设计	233
12.5 基于 RAM 的文件系统	231	12.6.1 文件系统格式选择的基本策略	233
12.5.1 Ramdisk 文件系统	231	12.6.2 混合型文件系统格式设计方法	234
12.5.2 RamFS/TmpFS 文件系统	232	12.7 本章小结	235

第 13 章 嵌入式 Linux C 语言开发工具 236

“工欲善其事，必先利其器”，熟悉手头的开发工具是每一个开发人员都必须完成的功课。本章介绍了在开发嵌入式 C 语言时需要用到的主要开发工具，包括命令行模式下的开发工具：编辑器 VIM、编译器 GCC、调试器 GDB、项目管理 MAKE、版本管理 CVS 和图形化模式下的集成开发环境 Eclipse 等。读者可根据个人的喜好选择一种适合自己的开发环境。

13.1 编辑器 VIM	236	13.3.3 GdbServer 远程调试	253
13.1.1 VIM 的编辑模式	236	13.4 工程管理 Make	255
13.1.2 VIM 的进入与退出	238	13.4.1 Makefile 文件介绍	255
13.1.3 光标的移动	239	13.4.2 Makefile 的规则	256
13.1.4 删除和恢复	239	13.4.3 Makefile 的变量	258
13.1.5 复制和粘贴	240	13.4.4 Make 命令的使用	260
13.1.6 查找和替换	240	13.4.5 使用自动工具生成 Makefile	261
13.1.7 网络资源	241	13.5 集成开发环境 Eclipse	263
13.2 编译器 GCC	242	13.5.1 Eclipse 的安装	263
13.2.1 GCC 的编译流程	242	13.5.2 Eclipse 的界面简介	264
13.2.2 GCC 的常用编译选项	245	13.5.3 创建 Hello 项目	265
13.2.3 实例分析	248	13.5.4 调试 Hello 项目	267
13.3 调试器 GDB	250	13.5.5 使用 CVS 进行版本管理	267
13.3.1 GDB 使用概述	250	13.6 本章小结	271
13.3.2 GDB 的使用流程	251		

第 14 章 快速体验——嵌入式 C 语言开发流程 272

本章分别介绍了命令行模式下和集成开发环境下开发嵌入式程序的基本流程。首先介绍了在命令行下如何通过 VI 编辑一个代码文件，如何在 PC 上和开发板上运行编译好的程序以及如何通过编写 Makefile 文件提高编译和文件管理的效率。然后介绍了在 Eclipse 下创建工程、编写代码、编译工程和运行最终程序的过程。

14.1 命令行下的开发流程	272	14.2.1 下载和安装 Eclipse	277
14.1.1 编写代码	272	14.2.2 新建工程	278
14.1.2 编译程序	274	14.2.3 编写代码	279
14.1.3 运行程序	275	14.2.4 编译工程	280
14.1.4 交叉编译	275	14.2.5 运行程序	281
14.1.5 编写 Makefile	276	14.3 本章小结	281
14.2 基于 Eclipse 的开发流程	277		

第 15 章 嵌入式 Linux C 语言基础 282

本章一起学习和回顾了 C 语言的基础知识，主体内容包括：C 语言概述、数据类型、运算符、表达式、流程、函数、数组和指针、复杂数据结构。本章的目的在于将 C 语言的语法精华浓缩成一章的内容，便于读

者随时翻阅和查找。通过本章的学习，读者可以立即进入编程的实战阶段。

15.1 C语言概述	282	15.5.3 选择结构设计	293
15.1.1 C语言的特点	282	15.5.4 循环结构设计	294
15.1.2 C语言程序的总体结构	282	15.6 函数	296
15.1.3 C语言的语句	283	15.6.1 概述	296
15.1.4 C语言的关键字	284	15.6.2 函数定义的一般形式	297
15.1.5 C语言程序设计步骤	284	15.6.3 函数的参数和函数的值	297
15.2 数据类型	284	15.6.4 函数的调用	298
15.2.1 基本数据类型	284	15.6.5 局部变量和全局变量	299
15.2.2 常量与变量	285	15.7 数组、指针	300
15.2.3 整型数据、实型数据	285	15.7.1 数组	300
15.2.4 字符型数据	286	15.7.2 指针的基本概念	302
15.3 运算符	287	15.7.3 指针与数组	304
15.3.1 算术运算符	287	15.7.4 指针与字符串	306
15.3.2 关系和逻辑运算符	287	15.7.5 指针与函数	307
15.3.3 位操作符	288	15.7.6 指针其他用法	308
15.3.4 ? 操作符	288	15.7.7 动态内存管理	308
15.3.5 表达式的优先级	288	15.8 复杂数据结构	309
15.4 表达式	289	15.8.1 结构体定义	309
15.4.1 类型转换	289	15.8.2 结构体使用	310
15.4.2 构成符 cast 和可读性	289	15.8.3 链表	312
15.5 流程控制	289	15.8.4 枚举类型	313
15.5.1 格式输入输出	289	15.8.5 共用体类型	313
15.5.2 顺序程序设计	292	15.9 本章小结	313

第 16 章 嵌入式 Linux C 语言标准库

314

本章介绍了在 Linux 下编程时常用的标准库函数，包括字符测试、数据转换、基本 I/O、标准 I/O、内存配置、字符串处理、日期时间、错误处理、系统日志和环境管理等函数。介绍了这些函数的基本格式和用法，并对主要的函数进行了举例说明。灵活使用这些函数，将给程序设计带来很大的便利。

16.1 Glibc 简介	314	16.4.3 fread()函数和 fwrite()函数	324
16.2 字符测试和数据转换函数	316	16.4.4 printf()函数和 scanf()函数	325
16.2.1 字符测试函数	316	16.4.5 其他函数	326
16.2.2 数据转换函数	317	16.5 内存配置及字符串处理函数	326
16.3 基本 I/O 函数	318	16.5.1 内存分配函数	327
16.3.1 open()函数	319	16.5.2 memXXX 函数	327
16.3.2 close()函数	320	16.5.3 strXXX 函数	330
16.3.3 read()函数和 write()函数	320	16.5.4 释放内存的函数	331
16.3.4 其他函数	321	16.5.5 动态内存分配的实例	332
16.4 标准 I/O 函数	322	16.6 日期时间函数	334
16.4.1 fopen()函数	322	16.6.1 时间的定义	334
16.4.2 fclose()函数	323	16.6.2 日历时间	334

16.6.3 时钟计时单元	336	16.7.1 错误处理函数	339
16.6.4 格式化日期和时间	337	16.7.2 系统日志函数	342
16.6.5 自定义时间格式	337	16.7.3 环境管理函数	344
16.7 其他函数	339	16.8 本章小结	345

第 17 章 嵌入式 Linux 的多任务编程

346

本章介绍了多任务处理的基本知识，主要介绍了基于进程和线程实现多任务处理的不同特点及区别。分别介绍了进程和线程的概念、数据结构，介绍了创建和终止一个进程或线程的基本方法，同时对于一些较为深入的内容，如僵尸（Zombie）进程、线程属性等，本章也做了基本的介绍。最后，介绍了一种多任务处理的解决方案：线程池，并给出了全部的实现代码，具有很好的参考价值。

17.1 什么是多任务	346	17.2.10 Zombie 进程	365
17.1.1 对话级多任务	346	17.3 线程	366
17.1.2 进程级多任务	347	17.3.1 线程的概念	366
17.1.3 线程级多任务	347	17.3.2 线程的创建	367
17.1.4 多任务处理的特点	348	17.3.3 线程的终止	368
17.2 进程	349	17.3.4 线程的基本属性	370
17.2.1 进程的概念	350	17.3.5 线程属性的修改	371
17.2.2 进程的数据结构	351	17.3.6 线程的扩展属性	374
17.2.3 进程的创建	353	17.4 线程池	377
17.2.4 文件描述符共享	355	17.4.1 线程池的工作原理	377
17.2.5 vfork()函数	357	17.4.2 线程池的实现	378
17.2.6 exec()函数族	357	17.4.3 工作状态的记录	383
17.2.7 执行新程序	359	17.4.4 线程池的测试	386
17.2.8 进程的终止	361	17.5 本章小结	387
17.2.9 进程的退出状态	363		

第 18 章 多任务间通信和同步

388

本章介绍了在嵌入式 Linux 下进行多任务处理时，任务间通信和同步的方法。主要介绍了信号、信号集、管道、FIFO、共享内存（mmap()方式和 semget()方式）等数据通信方式，以及信号量（SystemV 方式和 POSIX 方式）、互斥锁、条件变量等同步方式。这些内容都是编制多任务处理程序时所必备的，而且不同的方式有它不同的特点和适用的场合，因此在实际应用中应该在对任务有充分理解和测试后进行优化选择。

18.1 信号	388	18.1.11 sigpending()函数	396
18.1.1 信号的概念	388	18.1.12 sigsuspend()函数	396
18.1.2 信号的产生	390	18.2 管道	398
18.1.3 kill()函数和 raise()函数	390	18.2.1 管道的相关概念	398
18.1.4 alarm()函数和 pause()函数	391	18.2.2 管道的创建	398
18.1.5 about()函数	392	18.2.3 多进程中的管道通信	400
18.1.6 信号的处理	392	18.2.4 管道的应用实例	401
18.1.7 signal()函数	392	18.2.5 FIFO 的相关概念	402
18.1.8 sigaction()函数	393	18.2.6 FIFO 的创建	403
18.1.9 信号集	395	18.2.7 FIFO 的读写规则	404
18.1.10 sigprocmask()函数	396	18.2.8 FIFO 的应用实例	405

18.3 共享内存	407	18.7.1 系统调用 sem_init()	422
18.3.1 系统调用 mmap()	408	18.7.2 系统调用 sem_wait()	422
18.3.2 系统调用 munmap()	408	18.7.3 系统调用 sem_post()	422
18.3.3 系统调用 msync()	409	18.7.4 系统调用 sem_destory()函数	422
18.3.4 mmap()的应用实例	409	18.7.5 POSIX 信号量的应用实例	422
18.4 System V 共享内存	411	18.8 互斥锁	424
18.4.1 系统调用 shmget()	411	18.8.1 系统调用 pthread_mutex_init()	424
18.4.2 系统调用 shmat()	412	18.8.2 系统调用 pthread_mutex_lock()	424
18.4.3 系统调用 shmdt()	412	18.8.3 系统调用 pthread_mutex_trylock()	424
18.4.4 System V 共享内存的应用实例	412	18.8.4 系统调用 pthread_mutex_unlock()	425
18.5 消息队列	413	18.8.5 系统调用 pthread_mutex_destory()	425
18.5.1 系统调用 msgget()	413	18.8.6 互斥锁的应用实例	425
18.5.2 系统调用 msgsnd()	414	18.9 条件变量	427
18.5.3 系统调用 msgrcv()	414	18.9.1 系统调用 pthread_cond_init()	428
18.5.4 系统调用 msgctl()	415	18.9.2 系统调用 pthread_cond_wait()	428
18.5.5 消息队列的应用实例	415	18.9.3 系统调用 pthread_cond_timedwait()	428
18.6 System V 信号量	417	18.9.4 系统调用 pthread_cond_signal()	428
18.6.1 系统调用 semget()	417	18.9.5 系统调用 pthread_cond_	
18.6.2 系统调用 semop()	418	broadsignal()	428
18.6.3 系统调用 semctl()	419	18.9.6 系统调用 pthread_cond_destory()	428
18.6.4 System V 信号量的应用实例	419	18.9.7 条件变量的应用实例	429
18.7 POSIX 信号量	421	18.10 本章小结	430

第 19 章 设备驱动开发基础

431

本章主要内容介绍了 Linux 驱动开发的基础知识,包括字符设备和块设备驱动。其中字符设备知识主要包括:内核与模块、并发控制、阻塞、中断、内存操作。

作为初学者,大多数读者并不需要在实际工作中接触到块设备的驱动编写,因此建议读者将主要精力放在学习字符设备驱动之上,而这也足可以满足大多数情况下的实际需要。

19.1 Linux 设备管理和驱动概述	431	19.3 块设备驱动编写	457
19.1.1 Linux 设备的分类	431	19.3.1 块设备的 I/O 操作特点	457
19.1.2 设备驱动程序的作用	431	19.3.2 block_device_operations 结构体	457
19.1.3 访问设备的实现	432	19.3.3 gendisk 结构体	458
19.1.4 Linux 设备控制方式	432	19.3.4 request 结构体	459
19.2 Linux 设备驱动开发流程	434	19.3.5 request 操作函数	462
19.2.1 构造和运行模块	434	19.3.6 bio 结构体	464
19.2.2 字符设备驱动编写	435	19.3.7 注册与注销	466
19.2.3 字符设备驱动示例	437	19.3.8 加载与卸载	466
19.2.4 并发控制	440	19.3.9 打开与释放	468
19.2.5 阻塞与非阻塞	445	19.3.10 ioctl 函数	469
19.2.6 select 和 poll	449	19.3.11 I/O 请求处理	469
19.2.7 中断处理	451	19.4 本章小结	472
19.2.8 内存与 I/O 操作	452		

本章介绍了开发嵌入式 Linux 网络程序的基本流程,先后分析了 TCP Sever 程序、TCP Client 程序、UDP 通信程序、多线程文件服务器程序和 PROXY 程序。针对网络程序涉及的函数多、内容复杂等特点,本章采取了在实际的网络程序代码中学习编程的方法,在阅读代码的同时进行讲解,将在进行网络编程中常用的函数和知识做了详细的阐述,希望对读者理解网络程序的编写能有所帮助。

20.1 TCP/IP 协议	473	20.6.1 DNS 操作	486
20.2 TCP 协议	474	20.6.2 连接服务器	487
20.2.1 TCP 连接建立的过程	474	20.6.3 测试实例	488
20.2.2 TCP 连接的标识	474	20.7 UDP 通信的程序设计	489
20.2.3 关闭 TCP 连接	475	20.7.1 UDP 的通信过程	489
20.3 UDP 协议	475	20.7.2 UDP 通信服务器端	490
20.4 socket 简介	475	20.7.3 UDP 通信客户端	491
20.4.1 socket 的定义	476	20.8 多线程文件服务器	493
20.4.2 socket 的类型	476	20.8.1 文件服务器主程序	493
20.5 TCP Server 程序设计	476	20.8.2 动态分配监听端口	495
20.5.1 TCP 的通信过程	476	20.8.3 多线程服务器的实现	495
20.5.2 TCP Server 程序	477	20.8.4 大数据量的读写函数	496
20.5.3 网络地址的表示	478	20.8.5 客户端测试例程	499
20.5.4 建立 socket	479	20.8.6 编译和测试	501
20.5.5 绑定本地地址	479	20.9 PROXY 源代码分析	501
20.5.6 字节顺序转换	480	20.9.1 主函数 main()	502
20.5.7 IP 地址转换	481	20.9.2 参数处理函数 parse_args()	504
20.5.8 Listen()函数	481	20.9.3 守护进程函数 daemonize()	505
20.5.9 等待连接	482	20.9.4 代理服务函数 do_proxy()	507
20.5.10 数据通信	483	20.9.5 错误信息函数 errorout()	510
20.5.11 关闭套接字	484	20.10 本章小结	510
20.6 TCP Client 程序设计	485		

第 21 章 MiniGUI 图形界面设计

本章主要介绍了嵌入式图形界面库 MiniGUI 的程序设计。首先介绍了 MiniGUI 的安装、配置以及使用 Eclipse 进行 MiniGUI 应用开发的配置方式。然后对开发 MiniGUI 程序的基本函数进行解释,包括消息和消息传递、窗口过程、对话框、控件、图形设备和配置文件等方面。最后列举了一个利用 MiniGUI 编成的图形界面程序 MiniQQ,对这个例程的三个主要窗口的设计与编写做了详细的介绍。

21.1 MiniGUI 概述	511	21.2.4 编译应用程序例子	516
21.1.1 MiniGUI 的特点	511	21.2.5 交叉编译 MiniGUI 库	516
21.1.2 MiniGUI v1.3.3 软件包	512	21.2.6 交叉编译例程	517
21.1.3 MiniGUI 运行模式	512	21.2.7 QVFB 图形引擎	518
21.2 MiniGUI 的安装和使用	514	21.2.8 FrameBuffer 图形引擎	519
21.2.1 安装 MiniGUI 库	514	21.3 利用 Eclipse 编写 MiniGUI 程序	520
21.2.2 安装 MiniGUI 的资源	515	21.3.1 建立 vacs 工程	520
21.2.3 配置 MiniGUI	515	21.3.2 配置编译选项	521

21.3.3	配置外部工具 QVFB	522	21.6.4	控件子类化	535
21.3.4	运行 vacs	523	21.6.5	对话框和对话框模板	536
21.3.5	调试 vacs	523	21.6.6	模态和非模态对话框	538
21.4	MiniGUI 的编程基础	524	21.7	图形设备接口	539
21.4.1	头文件	525	21.7.1	图形设备上下文	539
21.4.2	程序入口	525	21.7.2	矩形操作和区域操作	540
21.4.3	创建和显示主窗口	526	21.7.3	像素值和调色板	542
21.4.4	进入消息循环	526	21.7.4	位图操作函数	543
21.4.5	窗口过程函数	528	21.7.5	字体和文本输出	544
21.4.6	屏幕输出	528	21.7.6	绘制图形	546
21.4.7	程序的退出	528	21.8	其他功能	547
21.5	消息循环和窗口过程	529	21.8.1	读写配置文件	548
21.5.1	消息处理函数	529	21.8.2	定点数运算	548
21.5.2	重要的消息	530	21.9	MiniQQ 界面设计	549
21.5.3	窗口及窗口过程	530	21.9.1	登录窗口	551
21.6	对话框和控件编程	531	21.9.2	好友列表窗口	554
21.6.1	控件的概念	531	21.9.3	聊天窗口	558
21.6.2	预定义控件	532	21.9.4	其他	559
21.6.3	自定义控件	534	21.10	本章小结	560

第 22 章 CAN 总线驱动设计

561

随着需求的发展,许多工业控制场合需要嵌入式设备,需要提供 CAN 总线接口,来连接 CAN 设备,以满足应用的需要。本章详细介绍了 CAN 总线的基础知识,对 Linux 系统下的 CAN 驱动程序进行了详细的分析。最后结合本书使用的 ARM9 平台,介绍了移植 CAN 驱动程序的要点。

22.1	CAN 总线介绍	561	22.3.4	中断寄存器 IR	567
22.2	SJA1000 介绍	561	22.3.5	发送缓冲器	568
22.2.1	SJA1000 的特性	561	22.3.6	接收缓冲器	568
22.2.2	SJA1000 内部原理	562	22.3.7	验收滤波器	569
22.2.3	SJA1000 管脚说明	563	22.4	SJA1000 与 S3C2440 连接	569
22.3	SJA1000 寄存器介绍	564	22.5	SJA1000 的驱动程序	571
22.3.1	控制寄存器 CR	564	22.5.1	驱动程序源代码	571
22.3.2	命令寄存器 CMR	565	22.5.2	测试的操作方法	579
22.3.3	状态寄存器 SR	566	22.6	本章小结	580

第 23 章 SD 卡驱动设计

581

SD 卡在嵌入式设备中使用越来越多,本章首先介绍了 SD 卡的基本知识和协议内容,然后结合 S3C2440 的 SD 卡控制器,详细分析了 Linux 下 SD 卡的驱动程序,并给出了实现过程。驱动程序的编写对程序员要求很高,不仅代码要求完全正确,效率低下的驱动程序也是不合格的。因此如果要编写出好的驱动程序,经验非常重要。

23.1	SD 卡概述及协议	581	23.2	SD 卡驱动设计	583
23.1.1	SD 卡概述	581	23.2.1	块设备驱动设计	583
23.1.2	SD 卡协议	582	23.2.2	SD 卡驱动程序分析	584