

图书基本信息

书名：<<嵌入式Linux C语言应用程序设计与实践>>

13位ISBN编号：9787302225515

10位ISBN编号：7302225516

出版时间：2010-6

出版时间：清华大学出版社

作者：熊茂华 等著

页数：353

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

## 前言

在世界范围内，社会经济的发展产生了一些新的需求，这也促进了嵌入式技术的广泛应用。而中国正在成为世界制造大国，在消费电子、工业应用、军事国防、网络设备等领域都有嵌入式系统的应用，同时嵌入式技术反过来刺激了许多新的应用需求，如信息家电、医疗电子病历、微小型智能武器等领域，嵌入式技术的应用日益广泛，相应地，企业针对嵌入式技术人才的需求也越来越多。因此近几年来，各高职高专院校也开始增设了嵌入式技术应用专业或方向。

但是，各院校在嵌入式技术应用专业教学过程中面临教材难觅的困境。

针对教材缺乏的情况，我们调研了几十所已开设了“嵌入式技术应用”专业的高职高专院校的嵌入式技术人才培养方案、课程设置、教材建设与开发、学生的学习情况及嵌入式技术应用人才就业现状，通过认真地整理、分析和研讨，编写了这套适于高等职业院校嵌入式技术人才培养规划的系列教材。

具体包括以下7本：《ARM体系结构与程序设计》 《嵌入式Linux C语言应用程序设计与实践》 《嵌入式操作系统与编程》 《嵌入式C/OS-II实时操作系统及应用编程》 《嵌入式Linux实时操作系统及应用编程》 《嵌入式windows CE应用开发技术》 《ARM 9嵌入式系统设计与开发应用》 本书主要内容如下。

第1章对ARM微处理器、ARM技术的基本概念做一些简单的介绍，从ARM体系结构的发展及技术特征、ARM微处理器的应用领域及特点、ARM微处理器的体系结构、ARM微处理器的应用选型等方面分别进行阐述。

第2章介绍嵌入式Linux的开发环境、Linux开发工具GNU gcc·的使用、GNU make命令和Makefile文件、gdb调试器、嵌入式Linux编辑器vi的使用、Emacs综合编辑器等。

第3章介绍嵌入式C语言程序设计的一些基本概念。主要内容包括嵌入式C语言预处理伪指令、嵌入式C语言的基本数据类型、嵌入式C语言程序结构、嵌入式C语言函数、嵌入式C语言数组、指针、结构体和联合等。

第4章介绍嵌入式C语言和汇编语言的混合编程、嵌入式Linux静态链接库与动态链接库及嵌入式Linux下程序调试等知识。

## 内容概要

《嵌入式Linux C语言应用程序设计与实践》详细介绍嵌入式Linux的开发环境与工具软件、嵌入式C语言程序设计基础、嵌入式Linux C高级编程与调试、Linux下的文件的操作和I/O应用、ARM Linux进程与进程调度、ARM Linux多线程开发、ARM Linux网络编程和嵌入式Linux设备驱动程序开发。

《嵌入式Linux C语言应用程序设计与实践》是嵌入式Linux C语言应用程序设计的一本实用指导书，通过案例详细介绍嵌入式Linux C语言应用程序设计，案例中的程序都取自实际的项目，并且对程序有详细的注解。

《嵌入式Linux C语言应用程序设计与实践》深入浅出，既可作为高等职业院校相关课程的教材，也可作为嵌入式系统编程人员的技术参考书。

## 书籍目录

第1章 嵌入式系统基础1.1 嵌入式系统简介1.1.1 嵌入式系统的概述1.1.2 嵌入式系统的组成1.1.3 嵌入式系统的应用领域1.1.4 嵌入式系统的发展趋势1.2 嵌入式处理器1.2.1 嵌入式处理器的分类1.2.2 嵌入式处理器的现状1.2.3 ARM处理器1.3 嵌入式系统开发过程1.3.1 嵌入式系统的总体结构1.3.2 嵌入式系统的开发特点1.3.3 嵌入式系统的开发流程1.3.4 调试嵌入式系统练习题第2章 嵌入式Linux的开发环境及工具软件2.1 嵌入式Linux的开发环境2.1.1 嵌入式Linux开发环境建立2.1.2 嵌入式Linux开发的一般过程2.2 Linux开发工具的使用2.2.1 Linux开发工具GNU gcc的使用2.2.2 gdb调试器简介2.3 GNU make命令和Makefile文件2.3.1 Makefile文件的规则2.3.2 Makefile文件中隐含规则2.3.3 Makefile文件的命令2.3.4 Makefile文件的变量2.3.5 Makefile文件的条件判断2.3.6 Makefile文件中常用函数2.3.7 子目录Makefile2.4 嵌入式Linux编辑器vi的使用2.4.1 vi的基本模式2.4.2 vi的基本操作2.5 Emacs综合编辑器2.5.1 Emacs的启动与退出2.5.2 Emacs的基本编辑2.5.3 Emacs的C模式2.5.4 Emacs的Shell模式练习题第3章 嵌入式C语言程序设计基础3.1 嵌入式C语言预处理伪指令3.2 嵌入式C语言的基本数据类型3.2.1 数据类型与表达式3.2.2 常量3.2.3 变量3.2.4 运算符3.3 嵌入式Linux C语言程序结构和控制语句3.3.1 嵌入式Linux C语言3种程序结构3.3.2 嵌入式Linux C语言基本语句3.4 选择语句3.4.1 if语句3.4.2 switch语句3.5 循环语句3.5.1 while和do-while语句3.5.2 for循环语句3.5.3 break语句和continue语句3.5.4 ARM Linux循环语句应用实例3.6 goto语句3.6.1 goto语句语法3.6.2 ARM Linux中goto语句应用实例3.7 函数3.7.1 C语言函数概述3.7.2 函数定义与声明3.7.3 函数的参数、值和基本调用3.7.4 函数的嵌套、递归调用3.8 数组3.9 指针3.10 结构体与联合练习题第4章 嵌入式Linux C高级编程及调试举例4.1 内嵌汇编指令4.2 汇编语言与C/C++的混合编程4.3 从汇编程序中访问C程序变量4.4 汇编程序与C/C++程序的相互调用规则——ATPCS4.5 汇编程序与C/C++程序的相互调用4.6 嵌入式Linux静态链接库与动态链接库4.6.1 Linux静态链接库及创建4.6.2 Linux动态链接库及创建4.7 嵌入式Linux下程序调试应用举例4.7.1 Linux宿主机下的应用程序调试4.7.2 目标机下的应用程序调试练习题第5章 嵌入式Linux下文件的操作5.1 ARM Linux文件I/O系统概述5.1.1 虚拟文件系统5.1.2 通用文件模型5.2 不带缓存的文件I/O操作5.2.1 文件的创建和读写5.2.2 移动文件的读写位置5.2.3 文件的各个属性5.2.4 目录文件的操作5.3 Linux标I/O流5.4 嵌入式Linux的文件操作应用举例5.4.1 文件锁的应用5.4.2 嵌入式Linux串口通信设计练习题第6章 ARM Linux进程与进程调度6.1 ARM Linux进程管理6.1.1 进程描述符及任务结构6.1.2 进程的调度6.2 ARM Linux进程控制相关API6.3 ARM Linux进程间通信API6.3.1 Linux中进程间通信6.3.2 管道6.3.3 命名管道6.3.4 信号通信6.3.5 共享内存6.3.6 消息队列6.4 Linux守护进程6.4.1 守护进程概述6.4.2 编写规则6.4.3 守护进程实例练习题第7章 ARM Linux多线程开发实例7.1 Linux多线程相关API7.1.1 线程的创建7.1.2 线程的终止7.1.3 线程的标识7.1.4 线程的一次性初始化7.1.5 线程的私有数据7.2 信号灯7.2.1 POSIX有名信号灯的API函数7.2.2 POSIX基于内存的信号灯的API函数7.3 互斥量7.4 条件变量练习题第8章 ARM Linux网络编程实例8.1 TCP/IP协议简介8.1.1 TCP/IP的分层模型8.1.2 TCP/IP分层模型特点8.1.3 Internet层中的协议8.1.4 传输层的协议8.2 网络基础编程8.2.1 Socket概述8.2.2 Socket基础8.2.3 Socket网络编程练习题第9章 嵌入式Linux设备驱动程序开发9.1 嵌入式Linux的设备管理9.1.1 设备驱动和文件系统的关系9.1.2 设备类型9.1.3 设备号9.1.4 设备驱动中关键数据结构9.1.5 设备驱动开发中基本函数9.2 设备驱动模块化编程9.2.1 设备驱动程序的开发流程9.2.2 内核空间和用户空间9.2.3 设备注册和初始化9.2.4 中断管理9.2.5 加载和卸载驱动程序9.3 GPIO字符设备驱动程序设计9.3.1 GPIO接口设计9.3.2 LED驱动程序代码分析9.3.3 LED驱动程序加载及测试9.4 A/D转换器驱动程序设计9.4.1 S3C2410X中断控制器9.4.2 S3C2410X中断接口函数及宏定义9.4.3 S3C2410XA/D转换器9.4.4 A/D转换驱动程序设计9.4.5 A/D转换驱动程序的测试练习题参考文献

## 章节摘录

2.3 GNU make命令和Makefile文件      Makefile文件描述了目标文件之间的依赖关系，以及指定编译过程中使用的工具。

Makefile主要包含了5个方面：显式规则、隐晦规则、变量定义、文件指示和注释。

(1) 显式规则。

显式规则说明了如何生成一个或多个目标文件。

这由Makefile的书写者明显指出，要生成的文件、文件的依赖文件、生成的命令。

(2) 隐晦规则。

由于make有自动推导的功能，所以隐晦的规则可以比较简略地书写Makefile，这是由make所支持的。

(3) 变量定义。

在Makefile中可定义一系列的变量，变量一般是字符串，类似c语言中的宏，当Makefile被执行时，其中的变量都会被扩展到相应的引用位置上。

(4) 文件指示。

其包括3个部分，一个是在一个Makefile中引用另一个Makefile，就像C语言中的include一样；另一个是指根据某些情况指定Makefile中的有效部分，就像C语言中的预编译#if一样；还可定义一个多行的命令。

(5) 注释。

Makefile中只有行注释，和UNIX的Shell脚本一样，其注释是用“#”字符，如果在Makefile中使用“#”字符，可以用反斜杠进行转义，如：“\#”。

最后，还值得一提的是，在Makefile中的命令，必须要以Tab键开始。

Makefile定义了一系列的规则来指定哪些文件需要先编译，哪些文件需要后编译，哪些文件需要重新编译，甚至于进行更复杂的功能操作。

Makefile就像一个Shell脚本一样，其中也可以执行操作系统的命令。

Makefile带来的好处就是“自动化编译”，一旦写好，只需要一个make命令，整个工程完全自动编译，极大地提高了软件开发的效率。

GNU的make工作时的执行步骤如下。

(1) 读入所有的Makefile。

(2) 读入被include的其他Makefile。

(3) 初始化文件中的变量。

(4) 推导隐晦规则，并分析所有规则。

(5) 为所有的目标文件创建依赖关系链。

(6) 根据依赖关系，决定哪些目标要重新生成。

(7) 执行生成命令。

(1) ~ (5) 为第一个阶段，(6) ~ (7) 为第二个阶段。

在第一个阶段中，如果定义的变量被使用了，那么，make会把其展开在使用的位罝。

但make并不会完全马上展开。

make程序利用Makefile中的数据 and 每个文件的最后修改时间来确定哪个文件需要更新，对于需要更新的文件，make程序执行Makefile数据中定义的命令来更新。

### 2.3.1 Makefile文件的规则

#### 版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>