<<Linux嵌入式系统开发>>

图书基本信息

书名：<<Linux嵌入式系统开发>>

13位ISBN编号：9787121152177

10位ISBN编号：7121152177

出版时间：2012-1

出版时间：电子工业出版社

作者：朱小远 谢龙汉

页数：538

<<Linux嵌入式系统开发>>

内容概要

本书以Linux嵌入式系统的基本开发技术为主线，以基于ARM架构的嵌入式处理器为嵌入式硬件平台，全面介绍嵌入式系统开发过程、ARM体系结构、Linux基础、Linux进程、Linux开发环境的建立、Linux操作系统移植、Bootloader的使用、Linux根文件系统的构建、设备驱动程序的开发、嵌入式GUI开发等嵌入式知识，最后介绍近年来较为热门的GPS导航系统的设计。

本书可作为机电控制、信息家电、工业控制、手持仪器、医疗器械、机器人技术等方面嵌入式系统开发与应用的参考书，也可作为高等院校有关嵌入式系统教学的本科生或研究生教材。

<<Linux嵌入式系统开发>>

书籍目录

第1章 嵌入式系统概述
1.1 嵌入式系统的概念
1.1.1 嵌入式系统的定义
1.1.2 嵌入式系统的特点
1.2 嵌入式系统的组成
1.2.1 嵌入式处理器
1.2.2 外围设备
1.2.3 嵌入式操作系统
1.2.4 应用软件
1.3 嵌入式处理器
1.3.1 嵌入式处理器的分类
1.3.2 嵌入式微处理器
1.3.3 嵌入式微控制器
1.3.4 嵌入式DSP处理器
1.3.5 嵌入式片上系统
1.3.6 选择嵌入式处理器
1.4 嵌入式操作系统
1.4.1 操作系统的概念和分类
1.4.2 实时操作系统
1.4.3 常用的嵌入式操作系统
1.5 新型的嵌入式操作系统
1.5.1 Android
1.5.2 MontaVista
1.6 嵌入式系统的应用
1.7 嵌入式系统的发展趋势
1.7.1 嵌入式系统面临的挑战
1.7.2 嵌入式系统的发展前景
1.8 本章小结
第2章 嵌入式系统开发过程
2.1 嵌入式软件开发介绍
2.1.1 嵌入式软件开发的特殊性
2.1.2 嵌入式软件的分类
2.1.3 嵌入式软件的开发流程
2.1.4 嵌入式软件开发工具的发展趋势
2.2 嵌入式软件的调试技术
2.2.1 调试技术介绍
2.2.2 基于JTAG的ARM系统调试
2.3 嵌入式软件测试技术
2.3.1 宿主机-目标机开发模式
2.3.2 目标监控器
2.4 嵌入式系统集成开发环境
2.4.1 ADS的介绍
2.4.2 ADS建立工程的使用介绍
2.4.3 AXD调试器的使用介绍
实例2-1：ARM开发环境ADS的使用实例

2.5 本章小结
第3章 ARM体系结构
3.1 ARM体系结构概述
3.1.1 ARM体系结构简介
3.1.2 ARM体系结构的技术特征
3.1.3 CISC的体系结构
3.1.4 RISC的体系结构
3.1.5 RISC系统和CISC系统的比较
3.2 ARM微处理器的分类
3.2.1 ARM7微处理器
3.2.2 ARM9微处理器
3.2.3 ARM9E微处理器
3.2.4 ARM10E微处理器
3.2.5 ARM11微处理器
3.2.6 SecurCore微处理器
3.2.7 trongARM微处理器
3.2.8 XScale微处理器
3.3 ARM微处理器的应用
3.3.1 ARM微处理器的应用选型
3.3.2 S3C2410处理器
3.4 存储器
3.4.1 存储器简介
3.4.2 SDRAM操作
3.4.3 Flash
3.5 ARM编程模型
3.5.1 数据类型
3.5.2 存储器格式
3.5.3 处理器工作状态
3.5.4 处理器运行模式
3.5.5 寄存器组织
3.5.6 内部寄存器
3.6 ARM指令的寻址方式
3.6.1 立即寻址
3.6.2 寄存器寻址
3.6.3 寄存器间接寻址
3.6.4 相对寻址
3.6.5 堆栈寻址
3.6.6 块复制寻址
3.6.7 变址寻址
3.6.8 多寄存器寻址
3.7 ARM指令集
3.7.1 ARM指令的格式
3.7.2 ARM指令分类
3.7.3 Thumb指令介绍
3.7.4 Thumb指令分类
3.7.5 ARM指令集和Thumb指令集的区别
3.8 ARM微处理器的异常

3.8.1 ARM体系结构所支持的异常类型
3.8.2 异常向量表
3.8.3 异常优先级
3.8.4 应用程序中的异常处理
3.8.5 各类异常的具体描述
3.9 本章小结
第4章 Linux基本操作
4.1 Linux系统的介绍
4.1.1 Linux的概况
4.1.2 Linux操作系统的构成
4.1.3 Linux常见的发行版本
4.1.4 Linux内核的特点
4.2 Linux命令的使用
4.3 vi编辑器的使用
4.3.1 vi编辑器的进入
4.3.2 命令模式的命令
4.3.3 末行模式的命令
实例4-1：vi编辑器使用实例
4.4 Shell编程
4.4.1 Shell基础介绍
4.4.2 Shell程序的变量和参数
4.4.3 运行Shell程序
4.4.4 Shell程序设计的流程控制
4.4.5 Shell输入与输出
4.4.6 bash介绍
4.5 综合实例
实例4-2：编写清除/var/log下的log文件综合实例
实例4-3：编写寻找死链接文件综合实例
4.6 本章小结
第5章 Linux进程
5.1 进程概述
5.1.1 进程结构
5.1.2 进程的控制操作
5.1.3 进程的属性
5.1.4 进程的创建和调度
5.1.5 Linux进程命令
5.2 系统调用
5.2.1 系统调用简述
5.2.2 系统调用的进入
5.2.3 与进程管理相关的系统调用
5.3 管道
5.3.1 管道系统调用
5.3.2 管道的分类
实例5-1：管道通信实例
5.4 信号
5.4.1 常见的信号种类
5.4.2 系统调用函数

5.4.3 信号的处理
5.4.4 信号与系统调用的关系
实例5-2：信号实例
5.5 信号量
5.5.1 信号量概述
5.5.2 相关的数据结构
5.5.3 相关的函数
实例5-3：信号量实例
5.6 共享内存
5.6.1 共享内存原理
5.6.2 共享内存对象的结构
5.6.3 相关的函数
实例5-4：共享内存实例
5.7 消息队列
5.7.1 有关的数据结构
5.7.2 相关的函数
实例5-5：消息队列实例
5.8 综合实例
实例5-6：多线程编程实例
5.9 本章小结
第6章 建立Linux开发环境
6.1 建立Linux开发环境
6.1.1 Cygwin开发环境
6.1.2 VMwareWorkstation开发环境
6.2 交叉编译的使用
6.2.1 GNU交叉工具链的设置
6.2.2 ARMGNU常用汇编语言
6.2.3 GNU交叉工具链的常用工具
6.2.4 交叉编译环境
6.3 Linux下的C编程
6.3.1 Linux程序设计特点
6.3.2 Linux下C语言编码的风格
6.3.3 Linux程序基础
6.3.4 Linux下C编程的库依赖
6.4 gcc的使用与开发
6.4.1 gcc简介和使用
6.4.2 gcc选项
6.4.3 gcc的错误类型
实例6-1：gcc编译器环境的应用实例
6.5 gdb调试器的介绍和使用
6.5.1 gdb调试器的使用
6.5.2 在gdb中运行程序
6.5.3 暂停和恢复程序运行
6.5.4 远程调试
实例6-2：gdb调试器环境的应用实例
6.6 GNUmake和Makefile的使用
6.6.1 Makefile的基本结构

6.6.2 Makefile的变量
6.6.3 Makefile的隐含规则
6.6.4 Makefile的命令使用
6.6.5 Makefile的函数使用
6.6.6 Makefile文件的运行
6.6.7 Makefile规则书写命令
实例6-3：Makefile的命令使用实例
6.7 autoconf和automake的使用
6.7.1 autoconf的使用
6.7.2 Makefile.am的编写
6.7.3 automake的使用
6.7.4 使用automake和autoconf产生Makefile
6.7.5 自动生成Makefile的方法
6.8 综合实例
实例6-4：gcc编译器的综合实例
实例6-5：gdb调试器的综合实例
实例6-6：Makefile的综合实例
6.9 本章小结
第7章 Linux操作系统移植
7.1 移植的概念
7.1.1 Linux可移植性发展
7.1.2 Linux的移植性
7.2 Linux内核结构
7.2.1 Linux内核组成
7.2.2 子系统相互间的关系
7.2.3 系统数据结构
7.2.4 Linux内核源代码
7.3 Linux内核配置
实例7-1：Linux内核配置实例
7.4 Linux操作系统移植介绍
7.4.1 Linux系统移植的两大部分
7.4.2 内核文件的修改
7.4.3 系统移植所必需的环境
7.5 综合实例
实例7-2：编译Linux内核应用实例
实例7-3：Linux内核的烧写实例
实例7-4：使用Kgdb构建Linux内核调试环境
7.6 本章小结
第8章 Bootloader的使用
8.1 Bootloader概述
8.1.1 Bootloader的作用
8.1.2 Bootloader的功能
8.1.3 Bootloader的种类
8.1.4 Bootloader的工作模式
8.1.5 Bootloader的启动方式
8.1.6 Bootloader的启动流程
8.1.7 Bootloader与主机的通信

8.2 vivi
8.2.1 vivi的常用命令和文件结构
8.2.2 vivi第一阶段的分析
8.2.3 vivi第二阶段的分析
8.2.4 vivi的配置与编译
8.3 U-boot
8.3.1 U-boot常用命令和源代码目录结构
8.3.2 U-boot支持的主要功能
8.3.3 U-boot的编译和添加命令
8.3.4 U-boot的启动介绍
8.3.5 U-boot的移植和使用
8.3.6 U-boot的启动过程
8.3.7 U-boot的调试
8.4 其他常见的Bootloader
8.5 综合实例
实例8-1：vivi编译实例
实例8-2：U-boot在S3C2410上的移植实例
实例8-3：Bootloader设计实例
8.6 本章小结
第9章 构建Linux根文件系统
9.1 Linux文件系统概述
9.1.1 Linux文件系统的特点
9.1.2 其他常见的嵌入式文件系统
9.1.3 Linux根文件目录结构
9.1.4 Linux文件属性介绍
9.2 使用BusyBox生成工具集
9.2.1 BusyBox概述
9.2.2 BusyBox进程和用户程序启动过程
9.2.3 编译/安装BusyBox
实例9-1：用BusyBox建立简单的根文件系统
9.3 构建根文件系统
实例9-2：构建根文件系统
9.4 配置yaffs文件
9.4.1 yaffs文件系统设置
9.4.2 yaffs文件系统测试
9.5 综合实例
实例9-3：制作/使用yaffs文件系统映像文件
实例9-4：制作/使用jffs2文件系统映像文件
9.6 本章小结
第10章 设备驱动程序开发
10.1 设备驱动程序概述
10.1.1 驱动程序的简介
10.1.2 设备分类
10.1.3 设备号
10.1.4 设备节点
10.1.5 驱动层次结构
10.1.6 设备驱动程序的特点

10.2 设备驱动程序与文件系统
10.2.1 设备驱动程序与文件系统的关系
10.2.2 设备驱动程序与操作系统的关系
10.2.3 Linux设备驱动程序的接口
10.2.4 设备驱动程序开发的基本函数
10.2.5 Linux驱动程序的加载
10.3 设备驱动程序的使用
10.3.1 驱动程序模块的加载
10.3.2 创建设备文件
10.3.3 使用设备
10.4 网络设备基础知识
10.4.1 网络协议
10.4.2 网络设备接口基础
10.5 网络设备驱动程序的架构
10.5.1 网络设备驱动程序体系结构
10.5.2 网络设备驱动程序模块分析
10.5.3 网络设备驱动程序的实现模式
10.5.4 网络设备驱动程序的数据结构
10.6 综合实例
实例10-1：键盘驱动开发实例
实例10-2：I2C总线驱动的编写实例
实例10-3：TFT-LCD显示驱动实例
10.7 本章小结
第11章 嵌入式GUI开发
11.1 嵌入式系统中的GUI简介
11.1.1 嵌入式GUI系统的介绍
11.1.2 基于嵌入式Linux的GUI系统底层实现基础
11.1.3 嵌入式GUI系统的分析与比较
11.2 嵌入式系统下MiniGUI的实现
11.2.1 图形用户界面MiniGUI简介
11.2.2 MiniGUI的发布版本
11.2.3 MiniGUI在S3C2410处理器上的移植过程
11.3 Qt/Embedded嵌入式图形开发基础
11.3.1 Qt/Embedded开发环境的安装
11.3.2 Qt/Embedded底层支持及实现代码分析
11.3.3 Qt/Embedded信号和插槽机制
11.3.4 Qt/Embedded窗口部件
11.3.5 Qt/Embedded图形界面编程
11.3.6 Qt/Embedded对话框设计
11.3.7 数据库
实例11-1：Qt/Embedded图形开发应用实例
11.4 Qtopia移植
11.4.1 Qtopia简介
11.4.2 交叉编译、安装Qtopia
实例11-2：Qtopia移植应用实例
11.5 Qt/Embedded应用开发
11.5.1 嵌入式硬件开发平台的选择

11.5.2 Qt/Embedded常用工具的介绍
11.5.3 交叉编译Qt/Embedded的库
11.5.4 Qt/E程序的编译与执行
实例11-3：Qt/Embedded实战演练
11.6 综合实例
实例11-4：Hello，Qt/Embedded应用程序
实例11-5：基本绘图应用程序的编写
11.7 本章小结
第12章 综合工程实例
12.1 文件系统的生成与烧写
12.1.1 yaffs文件系统的制作与生成
12.1.2 jffs2文件系统的制作与生成
12.2 基于Linux的数码相框
12.2.1 系统需求分析
12.2.2 系统总体设计
12.2.3 软件设计实现
12.2.4 软硬件集成
12.3 基于Linux的MPlayer解码播放器
12.3.1 可行性分析报告
12.3.2 系统总体设计
12.3.3 软件总体设计
12.3.4 软件详细设计
12.3.5 软硬件集成
12.4 基于Linux的GPS导航系统的开发
12.4.1 嵌入式开发流程图
12.4.2 GPS导航定位系统的系统定义
12.4.3 GPS导航系统的可行性分析报告
12.4.4 GPS导航系统需求分析
12.4.5 GPS导航系统总体设计实现
12.4.6 GPS导航系统硬件设计实现
12.4.7 GPS导航系统软件概括设计
12.4.8 GPS导航系统软件详细设计
12.4.9 GPS导航系统数据库的配置设计
12.4.10 GPS导航系统软件实现
12.5 本章小结

章节摘录

版权页：插图：（2）远程调试远程调试是一种允许调试器以某种方式控制目标机上被调试进程的运行方式，并具有查看和修改目标机上内存单元、寄存器及被调试进程中变量值等各种调试功能的调试方式。
在嵌入式系统中，调试器运行在宿主机的通用操作系统之上，被调试的进程运行在目标机的嵌入式操作系统中，调试器和被调试进程通过串口或者网络进行通信，调试器可以控制、访问被调试进程，读取被调试进程的当前状态，并能够改变被调试进程的运行状态。
嵌入式系统的交叉调试可分为硬件调试和软件调试两种。
硬件调试需要使用仿真调试器协助调试过程，硬件调试器是通过仿真硬件的执行过程，让开发者在调试时可以随时了解到系统的当前执行情况。
而软件调试则使用软件调试器完成调试过程。
在目标机上，嵌入式操作系统、应用程序代码构成可执行映像。
可以在宿主机上生成完整映像，再移植到目标机上；也可以把应用程序做成可加载模块，在目标机操作系统启动后，从宿主机向目标机加载应用程序模块。
2.3.2 目标监控器嵌入式系统开发环境中，目标监控器对嵌入式软件的开发和调试有至关重要的意义。
嵌入式系统的调试，与一般台式机上编程调试显著不同。
嵌入式调试工具是用于嵌入式系统开发中代码定制和调试的工具，分为驻留主机部分和驻留目标机部分。
驻留主机部分称为调试器，驻留目标机部分称为目标监控器。
目标监控器是解决嵌入式软件开发工具与这些支撑硬件的连接和通信的一个重要支持部件，是嵌入式应用开发、调试环境的核心部件，是许多功能模块实现的基础。
按照具体的实现方式的不同，可以将目标监控器分为软件监控器、硬件监控器、软件仿真器和软件模拟监控器。
（1）软件监控器软件监控器是驻留在目标机上通过软件手段实现的调试代理。
实际上，主机端的调试命令不是直接交由目标机硬件执行的，而是首先发送给软件监控器，再由软件监控器转交给目标机执行，然后将所监控的程序运行到断点处的相关信息反馈给主机端的调试器。
按照对目标机硬件和软件的控制能力，软件监控器分为引导型监控器和应用型监控器。
1）引导型监控器引导型监控器是一种具有启动系统、加载和调试包括内核在内的程序等功能的软件监控器，它实际上是一个具有监控功能的微型操作系统。

编辑推荐

《Linux嵌入式系统开发》：Linux-嵌入式系统开发的首选软件、信息家电、工业控制、医疗器械、机器人领域的必备技术、基础知识－实训实例－工程实例、实例操作视频教学，轻松学习。

<<Linux嵌入式系统开发>>

版权说明