

<<ARM体系结构与编程>>

图书基本信息

书名：<<ARM体系结构与编程>>

13位ISBN编号：9787302062240

10位ISBN编号：7302062242

出版时间：2003-2-1

出版时间：清华大学出版社

作者：杜春雷

页数：496

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>

<<ARM体系结构与编程>>

内容概要

ARM处理器是一种16/32位的高性能、低成本、低功耗的嵌入式RISC微处理器，由ARM公司设计，然后授权给各半导体厂商生产，它目前已经成为应用最为广泛的嵌入式处理器。

本书分14章对ARM处理器的体系结构、指令系统和开发工具作了比较全面的介绍。

其中包括ARM体系介绍、ARM程序设计模型、ARM汇编语言程序设计、ARM C / C++语言程序设计、ARM连接器的使用、ARM集成开发环境CodeWarrior IDE的介绍及高性能的调试工具ADW的使用。

并在此基础之上介绍一些典型的基于ARM体系的嵌入式应用系统设计时的基本技术。

通过阅读本书可以使读者掌握开发基于ARM的应用系统的各方面的知识。

本书既可作为学习ARM技术的培训材料，也可作为嵌入式系统开发人员的参考手册。

<<ARM体系结构与编程>>

书籍目录

第1章 ARM概述及其基本编程模型	1.1 ARM技术的应用领域及其特点	1.2 ARM体系结构的版本及命名方法	1.2.1 ARM体系结构的版本	1.2.2 ARM体系的变种	1.2.3 ARM / Thumb体系版本的命名格式	1.3 ARM处理器系列	1.3.1 ARM7系列	1.3.2 ARM9系列	1.3.3 ARM9E系列	1.3.4 ARM10E系列	1.3.5 SecurCore系列	1.4 ARM处理器模式	1.5 ARM寄存器介绍	1.5.1 通用寄存器	1.5.2 程序状态寄存器	1.5.3 ARM体系的异常中断	1.6.1 ARM中异常中断种类	1.6.2 ARM处理器对异常中断的响应过程	1.6.3 从异常中断处理程序中返回	1.7 ARM体系中存储系统	1.7.1 ARM体系中的存储空间	1.7.2 ARM存储器格式	1.7.3 非对齐的存储访问操作	1.7.4 指令预取和自修改代码																														
第2章 ARM指令分类及其一般编码格式	2.1 ARM指令集概要介绍	2.1.1 ARM指令的分类	2.1.2 ARM指令的寻址方式	2.1.3 ARM指令的条件码域	2.1.4 数据处理指令的操作数的寻址方式	2.1.5 杂类Load / Store指令的寻址方式	2.1.6 协处理器Load / Store指令的寻址方式	2.1.7 批量Load / Store指令的寻址方式	2.1.8 协处理器Load / Store指令的寻址方式	2.1.9 批量Load / Store指令的寻址方式	2.1.10 ARM协处理器指令	2.2 ARM指令寻址方式	2.2.1 数据处理指令的操作数的寻址方式	2.2.2 字及无符号字节的Load/Store指令的寻址方式	2.2.3 杂类Load / Store指令的寻址方式	2.2.4 批量Load / Store指令的寻址方式	2.2.5 协处理器Load / Store指令的寻址方式	2.2.6 批量Load / Store指令的寻址方式	第3章 ARM指令集介绍	3.1 ARM指令集	3.1.1 跳转指令	3.1.2 数据处理指令	3.1.3 乘法指令	3.1.4 杂类的算术指令	3.1.5 状态寄存器访问指令	3.1.6 Load / Store内存访问指令	3.1.7 批量Load / Store内存访问指令	3.1.8 信号量操作指令	3.1.9 异常中断产生指令	3.1.10 ARM协处理器指令	3.2 一些基本的ARM指令功能段	3.2.1 算术逻辑运算指令的应用	3.2.2 跳转指令的应用	3.2.3 Load / Store指令的应用	3.2.4 批量Load / Store指令的应用	3.2.5 信号量指令的应用	3.2.6 与系统相关的一些指令代码段																	
第4章 ARM汇编语言程序设计	4.1 伪操作	4.1.1 符号定义伪操作	4.1.2 数据定义伪操作	4.1.3 汇编控制伪操作	4.1.4 栈中数据帧描述伪操作	4.1.5 信息报告伪操作	4.1.6 其他的伪操作	4.2 ARM汇编语言伪指令	4.3 ARM汇编语言语句格式	4.3.1 ARM汇编语言中的符号	4.3.2 ARM汇编语言中的表达式	4.4 ARM汇编语言程序格式	4.4.1 汇编语言程序格式	4.4.2 汇编语言子程序调用	4.5 ARM汇编编译器的使用	4.6 汇编程序设计举例	4.6.1 ARM中伪操作使用实例	4.6.2 ARM中汇编程序实例	第5章 ARM存储系统	5.1 ARM存储系统概述	5.2 ARM中用于存储管理的系统控制协处理器CP15	5.2.1 访问CP15寄存器的指令	5.2.2 CP15中的寄存器	5.3 存储器管理单元MMU	5.3.1 存储器管理单元MMU概述	5.3.2 禁止 / 使能MMU	5.3.3 MMU中地址变换过程	5.3.4 MMU中存储访问权限控制	5.3.5 MMU中的域	5.3.6 关于快表的操作	5.3.7 ARM中的存储访问失效	5.4 高速缓冲存储器和写缓冲区	5.4.1 基本概念	5.4.2 cache的工作原理和地址映像方法	5.4.3 cache的分类	5.4.4 cache的替换算法	5.4.5 缓冲技术的使用注意事项	5.4.6 存储系统的一致性问题	5.4.7 cache内容锁定	5.4.8 与cache和写缓冲区相关的编程接口	5.5 快速上下文切换技术	5.5.1 快速上下文切换技术原理	5.5.2 快速上下文切换技术编程接口	5.6 与存储系统相关的程序设计指南	5.6.1 地址空间	5.6.2 存储器格式	5.6.3 非对齐的存储访问操作	5.6.4 指令预取和自修改代码	5.6.5 IMB	5.6.6 存储器映射的I / O空间	5.7 AIOA存储系统的实例	5.7.1 L7205的存储系统概述	5.7.2 L7205中的SDRAM	5.7.3 L7205中的MMU
第6章 ATPCS介绍	6.1 ATPCS概述	6.2 基本ATPCS	6.2.1 寄存器的使用规则	6.2.2 数据栈使用规则	6.2.3 参数传递规则	6.3 几种特定的ATPCS	6.3.1 支持数据栈限制检查的ATPCS	6.3.2 支持只读段位置无关 (ROPI) 的ATPCS	6.3.3 支持可读写段位置无关 (RWPI) 的ATPCS	6.3.4 支持ARM程序和Thumb程序混合使用的ATPCS	第7章 ARM程序和Thumb程序混合使用	7.1 概述	5 处理浮点运算的ATPCS																																									

<<ARM体系结构与编程>>

- 7.2 在汇编语言程序中通过用户代码支持interwork 7.2.1 可以实现程序状态切换的指令
- 7.2.2 与程序状态切换相关的伪操作 7.2.3 进行状态切换的汇编程序实例 7.3
- 在C/C++程序中实现interwork 7.4 在汇编语言程序中通过连接器支持interwork 7.4.1
- 利用veneers实现汇编程序间的程序状态切换 7.4.2 利用veneers实现汇编程序与C/C++程序间的
- 程序状态切换 第8章 C/C++以及汇编语言的混合编程 8.1 内嵌汇编器的使用
- 8.1.1 内嵌的汇编指令用法 8.1.2 内嵌的汇编器和armasm的区别 8.1.3
- 在C/C++程序中使用内嵌的汇编指令 8.1.4 内嵌汇编指令的应用举例 8.2 从汇编程序
- 中访问C程序变量 8.3 汇编程序.C程序以及C++程序的相互调用 8.3.1 在C++程序中
- 使用C程序头文件 8.3.2 汇编程序.C程序以及C++程序的相互调用举例 第9章
- 异常中断处理 9.1 ARM中异常中断处理概述 9.1.1 ARM体系中异常中断种类
- 9.1.2 异常中断向量表及异常中断优先级 9.1.3 异常中断使用的寄存器 9.2 进入
- 和退出异常中断的过程 9.2.1 ARM处理器对异常中断的响应过程 9.2.2 从异常中断
- 处理程序中返回 9.3 在应用程序中安装异常中断处理程序 9.3.1 在系统复位时安装异
- 常中断处理程序 9.3.2 在C程序中安装异常中断处理程序 9.4 SWI异常中断处理程序
- 9.4.1 SWI异常中断处理程序的实现 9.4.2 SWI异常中断调用 9.5 FIQ和IRQ异
- 常中断处理程序 9.5.1 IRQ/FIQ异常中断处理程序 9.5.2 IRQ异常中断处理程序举例
- 9.6 复位异常中断处理程序 9.7 未定义指令异常中断 9.8 指令预取中止异常中
- 断处理程序 9.9 数据访问中止异常中断处理程序 第10章 ARM C/C++编译器
- 器 10.1 ARM C/C++编译器概述 10.1.1 ARM C/C++编译器及语言库介绍
- 10.1.2 ARM编译器中与搜索路径相关的一些基本概念 10.2 ARM编译器命令行格式
- 10.2.1 过程调用标准 10.2.2 设置源程序语言类型 10.2.3 指定搜索路径 10.2.
- 4 设置预处理选项 10.2.5 设置输出文件类型 10.2.6 指定目标处理器和ARM体系版本
- 10.2.7 生成调试信息 10.2.8 代码生成的控制 10.2.9 控制警告信息的产生
- 10.2.10 编译时进行的一些额外的检查 10.2.11 控制错误信息 10.3 ARM编译器
- 中的pragmas 10.4 ARM编译器特定的关键词 10.4.1 用于声明函数的关键词
- 10.4.2 用于声明变量的关键词 10.4.3 用于限定数据类型的关键词 10.5 ARM编译器
- 支持的基本数据类型 10.6 ARM编译器中预定义宏 10.7 ARM中C/C++库 10.
- 7.1 ARM中C/C++运行时库概述 10.7.2 建立一个包含C/C++运行时库的C/C++应用程序
- 10.7.3 建立不包含C运行时库的应用程序 10.7.4 裁减C/C++运行时库以适应特定的目标
- 运行环境 第11章 ARM连接器 11.1 ARM映像文件 11.1.1 ARM映像
- 文件的组成 11.1.2 ARM映像文件的入口点 11.1.3 输入段的排序规则 11.2
- ARM连接器介绍 11.3 ARM连接器生成的符号 11.3.1 连接器生成的与域相关的符号
- 11.3.2 连接器生成的与输出段相关的符号 11.3.3 连接器生成的与输入段相关的符号
- 11.4 连接器的优化功能 11.5 运行时库的使用 11.5.1 C/C++运行时库与目标文
- 件 11.5.2 查找需要的C/C++运行时库 11.5.3 选择合适种类的C/C++运行时库
- 11.5.4 扫描C/C++运行时库 11.6 从一个映像文件中使用另一个映像文件中的符号
- 11.6.1 symdefs文件 11.6.2 建立symdefs文件 11.6.3 symdefs文件的使用 11.7
- 隐藏或者重命名全局符号 11.7.1 steering文件的格式 11.7.2 steering文件中的命令
- 11.8 ARM连接器命令行选项 11.9 使用scatter文件定义映像文件的地址映射 11.9.1
- scatter文件概述 11.9.2 satter文件中各部分介绍 11.9.3 scatter文件使用举例
- 第12章 嵌入式应用程序示例 12.1 嵌入式应用程序设计的基本知识 12.1.1 嵌入
- 式应用系统中的存储映射 12.1.2 系统初始化 12.2 使用semihosting的 C语言程序示例
- 12.2.1 源程序分析 12.2.2 生成映像文件 12.3 一个嵌入式应用系统示例
- 12.3.1 源程序分析 12.3.2 生成映像文件 12.3.3 本例中地址映射模式 12.4
- 进行ROM/RAM地址重映射的嵌入式应用系统 12.4.1 地址映射模式 12.4.2 源程序分
- 析 12.4.3 生成映像文件 12.5 一个嵌入式操作系统示例 第13章 使
- 用CodeWarrior 13.1 CodeWarrior for ADS概述 13.2 简单工程项目的使用 13.

<<ARM体系结构与编程>>

2.1 工程项目窗口	13.2.2 简单工程项目的使用	13.3 配置生成目标	13.3.1
Debug Settings对话框介绍	13.3.2 设置生成目标的基本选项	13.3.3 汇编器选项设置	
13.3.4 编译器的选项设置	13.3.5 连接器的选项设置	13.3.6 fromELF工具的选项设置	
13.4 复杂工程项目的使用	13.4.1 建立一个新的生成目标	13.4.2 将一个生成目标更名	
13.4.3 建立生成目标之间的依赖关系	13.4.4 子工程项目的使用		
13.5 工程项目模板	13.5.1 ADS中工程项目模板的使用	13.5.2 建立用户工程项目模板	
13.6 编译和连接工程项目	13.6.1 编译文件	13.6.2 生成工程项目	
第14章 ARM体系中的调试方法			
14.1 ARM体系中调试系统概述	14.2 基于Angel的调试系统	14.2.1 基于Angel的调试系统的概述	14.2.2 使用Angel开发应用程序
14.2.3 Angel执行的操作	14.2.4 将Angel移植到特定的目标系统	14.3 基于JTAG的调试系统	14.3.1 基于JTAG的调试系统的特点
14.3.2 基于JTAG的调试系统结构	14.3.3 目标系统中的调试功能扩展部件	14.3.4 基于JTAG的调试过程	
14.4 ADW使用介绍	14.4.1 ADW概述	14.4.2 ADW中的窗口	14.4.3 ADW使用介绍

<<ARM体系结构与编程>>

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>