

图书基本信息

书名：<<ARM Cortex-MO微控制器原理与实践>>

13位ISBN编号：9787512410374

10位ISBN编号：7512410379

出版时间：2013-1

出版时间：北京航空航天大学出版社

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>

内容概要

《ARM Cortex-M0微控制器原理与实践》以新唐公司ARM Cortex—M0内核的NuMicro M051系列微控制器为蓝本，由浅入深，软硬结合，全面系统地介绍基于该微控制器的原理与结构、开发环境与工具、各种接口与功能单元应用的软件编写方法。

《ARM Cortex-M0微控制器原理与实践》以夯实基础，面向应用，理论与实践、方法与实现紧密结合为主线展开，根据ARM Cortex—M0的运行速度快、资源丰富、功能强大等显著特点，采用C语言作为系统软件的开发平台，由浅入深，以螺旋式上升的方式进行编排。

在讲解原理和设计方法的同时，还穿插了作者相关的经验、技巧和注意事项，有很强的实用性和指导性。

书籍目录

绪论 0.1什么是微控制器 0.2微控制器历史 0.3微控制器应用领域 第1篇初步认知篇 第1章微控制器发展趋势 1.1概述 1.2 ARM Cortex—M微控制器优势 1.2.1指令集效率 1.2.2 8位应用程序的神话 1.2.3 1生能 1.2.4 8位和16位微控制器的局限 1.2.5低功耗 1.2.6内存访问效率 1.2.7通过降低操作频率来降低能耗 1.2.8通过缩短活跃周期来降低能耗 1.2.9低功耗的总体优势 1.2.10软件开发 1.2.11从8位或16位微控制器向ARM移植软件 1.2.12调试 1.2.13选择 1.2.14软件可移植性 1.2.15迁移成本 1.2.16结论 1.3 ARM Cortex—M微控制器程序迁移 第2章ARM概述 2.1 ARM 2.2 RISC 2.2.1简介 2.2.2概念分析 2.2.3特点 2.2.4 区别 2.2.5种类 2.2.6 CPU发展 2.2.7 CPU的制造过程 第3章ARM Cortex—M0 3.1总线架构 3.1.1什么是AMBA 3.1.2什么是AHB—Lite 3.1.3什么是CoreSight 3.2 Cortex—M0的结构特点 3.2.1编程模型 3.2.2存储模型 3.2.3异常处理 3.2.4功耗管理 3.2.5指令集 3.3开发工具 第4章ARM微控制器的指令集 4.1 ARM微控制器的指令的分类与格式 4.2 ARM指令的条件域 4.3 ARM指令的寻址方式 4.4 ARM指令集 4.4.1跳转指令 4.4.2数据处理指令 4.4.3乘法指令与乘加指令 4.4.4程序状态寄存器访问指令 4.4.5加载 / 存储指令 4.4.6批量数据加载 / 存储指令 4.4.7数据交换指令 4.4.8移位指令 (操作) 4.4.9协微控制器指令 4.4.10异常产生指令 4.4.11 Thumb指令及应用 第5章ARM C语言编程 5.1 C语言简史 5.2 C语言特点 5.2.1优点 5.2.2缺点 5.3数据类型 5.3.1基本数据类型 5.3.2数据类型修饰符signed和unsigned 5.4常量和变量 5.4.1常量 5.4.2变量 5.5操作符 5.5.1算术操作符 5.5.2关系操作符 5.5.3逻辑操作符 5.5.4位操作符 5.6控制结构 5.6.1选择 5.6.2循环 5.7结构体 5.8编译指令 5.8.1 #define和#under 5.8.2 #if和#endif 5.8.3 #error 5.9标准C库的应用 5.9.1标准C库的组成 5.9.2标准C库的使用流程 第2篇基础入门篇 第6章NuMicro M051系列微控制器 6.1概述 6.1.1低门数微控制器特征 6.1.2 NVIC特征 6.1.3调试 6.1.4总线接口 6.2系统管理器 6.2.1 系统复位 6.2.2系统电源架构 6.3系统存储映射 6.4系统管理器控制寄存器映射 6.5嵌套向量中断控制器 (NVIC) 6.5.1异常模式和系统中断映射 6.5.2操作描述 第7章平台搭建与下载工具 7.1平台搭建 7.1.1启动程序 7.1.2创建工程 7.1.3编译代码 7.1.4安装Nu—Link for Keil驱动 7.1.5设置Nu—Link 7.1.6下载代码 7.1.7硬件仿真 7.2 ISP下载 7.2.1 ISP下载工具概述 7.2.2 ISP下载步骤 7.3 ICP下载 7.3.1 ICP下载工具概述 7.3.2 ICP下载步骤 7.4 JTAG与串行调试 (SWD) 7.4.1 JTAG简介 7.4.2 SWD简介 第3篇深入篇 第4篇番外篇

章节摘录

版权页：插图：许多半导体公司持有ARM授权：Atmel、Broadcom、Cirrus Logic、Freescale（于2004从摩托罗拉公司独立出来）、Qualcomm、富士通、英特尔（借由和Digital的控诉调停）、IBM、英飞凌科技、任天堂、恩智浦半导体（于2006年从飞利浦独立出来）、OKI电气工业、三星电子、Sharp、STMicroelectronics，德州仪器和VLSI等许多这些公司均拥有各个不同形式的ARM授权。

虽然ARM的授权项目由保密合约所涵盖，在智慧财产权工业，ARM是广为人知最昂贵的CPU内核之一。单一的客户端产品包含一伞基本的ARM内核可能就需索取一次高达20万美元的授权费用。

而若是牵涉到大量架构上修改，则费用就可能超过千万美元。

2.2 RISC ARM公司设计的微控制器基于RISC架构，而RISC（reduced instruction Set computer，精简指令集计算机）是一种执行较少类型计算机指令的微控制器，起源于20世纪80年代的MIPS主机（即RISC机），RISC机中采用的微控制器统称RISC微控制器。

这样一来，它能够以更快的速度执行操作（每秒执行更多百万条指令，即MIPS）。

因为计算机执行每个指令类型都需要额外的晶体管和电路元件，计算机指令集越大就会使微控制器更复杂，执行操作也会更慢。

2.2.1 简介 纽约约克镇IBM研究中心的John Cocke证明，计算机中约20%的指令承担了80%的工作，他于1974年提出了RISC的概念。

第一台得益于这个发现的计算机是1980年IBM的PC / XT。

再后来，IBM的RISC System / 6000也使用了这个思想。

RISC这个词本身属于伯克利加利福尼亚大学的一个教师David Patterson。

RISC这个概念还被用在SUN公司的SPARC微控制器中，并促成了现在所谓的MIPS技术的建立，它是Silicon Graphics的一部分。

许多当前的微芯片都使用RISC概念。

2.2.2 概念分析 RISC概念已经引领了微控制器设计的一个更深层次的思索。

设计中必须考虑到：指令应该如何较好的映射到微控制器的时钟速度上（理想情况下，一条指令应在一个时钟周期内执行完）；体系结构需要多“简单”；以及在不诉诸于软件的帮助下，微芯片本身能做多少工作等等。

2.2.3 特点 改进特点 RISC和CISC相比，除了性能的改进，RISC的一些优点以及相关的设计改进还有：

如果一个新的微控制器其目标之一是不那么复杂，那么其开发与测试将会更快。

使用微控制器指令的操作系统及应用程序的程序员将会发现，使用更小的指令集使得代码开发变得更加容易。

RISC的简单使得在选择如何使用微控制器上的空间时拥有更多的自由。

比起从前，高级语言编译器能产生更有效的代码，因为编译器使用RISC机器上的更小的指令集。

除了RISC，任何全指令集计算机都使用的是复杂指令集计算（CISC）。

RISC典型范例如MIPS R3000、HP—PA8000系列，Motorola M88000等均属于RISC微控制器。

1. 主要特点 RISC微控制器不仅精简了指令系统，采用超标量和超流水线结构；它们的指令数目只有几十条，却大大增强了并行处理能力。

如：1987年SUN Microsystem公司推出的SPARC芯片就是一种超标量结构的RISC微控制器。

而SGI公司推出的MIPS微控制器则采用超流水线结构，这些RISC微控制器在构建并行精简指令系统多处理机中起着核心的作用。

RISC微控制器是当今UNIX领域64位多处理机的主流芯片。

编辑推荐

《ARM Cortex-M0微控制器原理与实践》可以作为高等院校电子、自动化、仪器仪表和计算机等相关专业的辅助教材，也可作为ARM Cortex—M0微控制器的培训教材，可供相关技术人员学习和参考。

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>