<<ARM Cortex微控制器教程>>

图书基本信息

书名：<<ARM Cortex微控制器教程>>

13位ISBN编号：9787811249453

10位ISBN编号：7811249456

出版时间：2010-1

出版时间：北京航空航天大学出版社

作者：马忠梅，徐琰，叶青林　编著

页数：401

前言

笔者从1985年初接触单片机到现在已25年，亲历了我国嵌入式系统的发展，同时也进行了嵌入式系统教学的探索。
笔者1997年出版的《单片机的C语言应用程序设计》是国内第1本单片机C语言图书。
该书推广单片机编程，以C语言为主，汇编为辅，采用模块化的编程方法。
用C语言编程，程序可读性强、可重用性高，提高了编程的效率。
为了程序的可移植性更强，上操作系统是进一步发展的必然趋势，但8／16位单片机上操作系统有其局限性。
正是由于想上操作系统，才关注到了ARM。
从计算机体系结构发展看，精简指令集计算机（RISC）在低功耗、流水线执行方面更具优势。
ARM公司由于手机的火爆，已成为32位RISC处理器的事实标准。
笔者自2002年出版国内第1本ARM图书《ARM嵌入式处理器结构与应用基础》后，一直致力于ARM处理器的应用推广工作，编写了《ARM&Linux嵌入式系统教程》，在本科教学中使用。
Linux的教学难度很大，一般要求学生学习过操作系统。
最早的嵌入式系统课在教学生怎么用Linux上耗费了很大精力，直到计算机操作系统课加了Linux上机实验，情况才好转。
但有限的理论课时和实验学时，软硬件很难兼顾。
研究生教学一直就没有放弃单片机。
从研究生教学调查看，还是微控制器（俗称单片机，简称MCU）用得多。
嵌入式系统应用以微控制器为主。
微控制器用量大，应用面广，已有很好的应用基础。
一个MCU设计可以改造一个旧产品，也可以创造一个新产品。
MCU对于本科生和研究生容易上手，更重要的是它集成度高，能够便于学生学做电路板、学习硬件接口技术和学习直接面向硬件的软件编程技术，这样才能对嵌入式软硬件有深入的理解。
ARM公司推出的Cortex-M核专门针对微控制器市场，并收购了德国的Keil公司，推出中国版的RealViewMDK开发工具。
这样原用8051单片机的用户可以将熟悉的开发环境过渡到ARMMCU的应用。
2008年，ARM公司“微控制器市场创新研讨会”的新理念是就支离破碎的微控制器市场，以ARM来统一微控制器市场开发环境，以后微控制器的选型不再以8位、16位和32位来划分。

<<ARM Cortex微控制器教程>>

内容概要

本书是32位微控制器教材，基于ARM Cortex讲述嵌入式系统概念、软硬件组成、开发过程以及Cortex体系结构和应用程序开发设计方法。
全书共8章，有嵌入式系统基础和ARM CortexM体系结构等理论内容，也有TI、ST、NXP和Atmel四家公司的Cortex微控制器时钟控制和应用程序设计等实践内容，另外还包括μC/OS移植和UML设计方法。
本书有两个主要目的，一是普及高端MCU，不要再面向寄存器编程，而要使用库函数；二是体现Cortex MCU很好的“生态环境”，有多家厂商支持。
本教材的特点是取材于最新资料，总结实际竞赛经验，实例较多，实用性较强。
本书适用于没有操作系统知识的单片机开发人员学习嵌入式系统，可作为研究生和本科生嵌入式系统课程的教材使用。

<<ARM Cortex微控制器教程>>

书籍目录

第1章 嵌入式系统基础 1.1 嵌入式系统概念 1.1.1 嵌入式系统定义 1.1.2 嵌入式系统组成 1.1.3 嵌入式系统特点 1.1.4 嵌入式系统应用 1.1.5 实时系统 1.2 嵌入式处理器 1.2.1 嵌入式处理器分类 1.2.2 微控制器 1.2.3 嵌入式微处理器 1.2.4 DSP处理器 1.2.5 片上系统 1.2.6 典型的嵌入式处理器 1.3 嵌入式操作系统 1.3.1 操作系统概念和分类 1.3.2 实时操作系统 1.3.3 常见的嵌入式操作系统 1.4 实时操作系统的内核 1.4.1 任务管理 1.4.2 任务间的通信和同步 1.4.3 存储器管理 1.4.4 定时器和中断管理 习题第2章 嵌入式系统开发过程 2.1 嵌入式软件开发的特点 2.2 嵌入式软件的开发流程 2.3 嵌入式系统调试 2.4 板级支持包 习题第3章 CortexM体系结构 3.1 ARM体系结构概述 3.1.1 ARM体系结构的特点 3.1.2 流水线 3.1.3 ARM处理器核 3.1.4 结构框图 3.1.5 典型的连接方式 3.1.6 ARM JTA调试接口 3.2 编程模型 3.2.1 Thumb2指令集体系结构（ISA） 3.2.2 寄存器 3.2.3 工作模式和特权级别 3.2.4 CortexM3堆栈 3.2.5 数据类型 3.2.6 存储器和存储器映射I/O 3.3 ARM基本寻址方式 3.4 Thumb2指令集说明 3.4.1 条件执行 3.4.2 指令分类说明 3.4.3 Thumb2指令集的特点 3.4.4 ARM汇编语言程序设计 3.5 存储器映射 3.5.1 存储系统简介 3.5.2 存储器映射空间 3.5.3 位绑定操作 3.6 中断和异常 3.6.1 异常类型 3.6.2 优先级的定义 3.6.3 向量表 3.6.4 中断输入及挂起行为 3.6.5 NVIC与中断控制 3.6.6 中断/异常的响应序列 3.6.7 尾链中断 3.6.8 迟到异常处理 3.6.9 异常返回值 3.6.10 中断延迟 3.7 ARM CortexM的优势 习题第4章 ARM CortexM微控制器第5章 片上资源的编程技术第6章 嵌入式系统接口及编程第7章 μC/OSII移植第8章 UML设计方法参考文献

章节摘录

1）消息邮箱　　消息邮箱通常是内存空间的一个数据结构。
除了包括一个代表消息的指针型变量外，每个邮箱都有相应的正在等待的任务队列。
要得到消息的任务时，如果发现邮箱是空的，就挂起自己，并放人到该邮箱的任务等待队列中等待消息。
通常，内核允许用户为任务等待消息设定超时。
如果等待时间已到仍没有收到消息，就进入就绪态，返回等待超时信息。
如果消息放入邮箱中，内核将把该消息分配给等待队列的其中一个任务。
2）消息队列　　消息队列实际上是一个邮箱阵列，在消息队列中允许存放多个消息。
对消息队列的操作和对消息邮箱的操作基本相同。
2.任务间的同步　　任务同步中也常常使用信号量。
与任务通信不同的是，信号量的使用不再作为一种互斥机制，而是代表某个特定的事件是否发生。
任务的同步有单向同步和多向同步两种。
（1）单向同步　　标志事件是否发生的信号量初始化为0。
一个任务在等待某个事件时，查看该事件的信号量是否为非0。
另一个任务或中断处理程序在进行操作时，当该事件发生后，将该信号量置为1。
等待该事件的任务查询到信号量的变换，代表事件已经发生，任务继续自身的运行。
（2）双向同步　　两个任务之间可以通过两个信号量进行双向同步。
双向同步有两个初始化为0的信号量，每个信号量进行一个方向的任务同步，两信号量的同步方向是相反的。
在每个方向上，信号量的操作与单向同步是完全相同的。
1.4.3 存储器管理　　存储器管理提供对内存资源的合理分配和存储保护功能。
由于其应用环境的特殊性，实时内核的存储器管理与一般操作系统的存储器管理存在着很大的差异。
通常操作系统的内核，由于可供使用的系统资源相对比较充足，实时性能只需满足用户能忍耐的限度，一般在秒级，系统考虑的是提供更好的性能和安全机制，所以操作系统通常都引入虚拟存储器管理。
嵌入式实时操作系统的存储管理相对较为简单。
由于虚拟存储器中经常要对页进行换人换出操作，所以内存中页命中率和换入换出所耗费的时间严重破坏了整个系统的确定性。
这种存储机制不能提供实时系统所要求的时间确定性，对于大多数嵌入式实时应用来说，响应和运行时间的确定是至关重要的。
对于实时应用，一个失去时效的正确结果与错误结果没有什么本质的不同，这就是实时内核不采用虚拟内存管理的原因。

<<ARM Cortex微控制器教程>>

版权说明