



00:00:00

60:01:00

12:05:60

00:30:10

周航慈 吴光文 著

基于嵌入式实时操作系统的 程序设计技术

 北京航空航天大学出版社

基于嵌入式实时操作系统的 程序设计技术

周航慈 吴光文 著

北京航空航天大学出版社

内 容 简 介

本书详细介绍了基于嵌入式实时操作系统的程序设计技术,内容有“绪论”、“开发环境”、“任务划分”、“任务设计”、“中断服务程序设计”、“行为同步”、“资源同步”、“数据通信”、“时间管理”、“采样任务设计”、“串行通信任务设计”、“键盘任务设计”、“显示任务设计”、“嵌入式操作系统的剪裁”和“设计实例”。书中介绍的内容以源码开放的嵌入式实时操作系统 $\mu\text{C}/\text{OS}-\text{II}$ 为软件运行环境,以 ARM7 为硬件环境。

本书内容深入浅出,为加深理解,列举了很多程序设计实例和实验。

本书可作为高等院校电子类相关专业本科和研究生的学习参考书,也可作为电子技术人员从事嵌入式应用系统研制开发的参考书。

图书在版编目(CIP)数据

基于嵌入式实时操作系统的程序设计技术/周航慈,
吴光文著. —北京:北京航空航天大学出版社,

2006. 11

ISBN 7 - 81077 - 941 - 9

I. 基… II. ①周…②吴… III. 实时操作系统—
程序设计 IV. TP316. 2

中国版本图书馆 CIP 数据核字(2006)第 131886 号

© 2006, 北京航空航天大学出版社, 版权所有。

未经本书出版者书面许可,任何单位和个人不得以任何形式或手段复制或传播本书内容。侵权必究。

基于嵌入式实时操作系统的程序设计技术

周航慈 吴光文 著

责任编辑 孔祥燮

*

北京航空航天大学出版社出版发行

北京市海淀区学院路 37 号(100083) 发行部电话:010-82317024 传真:010-82328026

<http://www.buaapress.com.cn> E-mail: bhpress@263.net

北京时代华都印刷有限公司印装 各地书店经销

*

开本:787 mm×960 mm 1/16 印张:13.75 字数:308 千字

2006 年 11 月第 1 版 2006 年 11 月第 1 次印刷 印数:5 000 册

ISBN 7 - 81077 - 941 - 9 定价:19.50 元

前 言

在上个世纪 90 年代,单片机在我国迅速普及,已很难找到一种没有使用单片机的电子新产品。作者在那时出版了《单片机应用程序设计技术》和《单片机程序设计基础》两部著作,也算为推动我国单片机应用尽了一份力。由于那时的单片机以 8 位机为主,本身的硬件资源有限,所处理的问题也不是很复杂,故程序设计方法绝大多数还是“基于裸机编程”的,即不使用操作系统。进入 21 世纪后,一大批所谓“信息电子产品”开始普及,其特点是处理对象包含多媒体信息流,CPU 的信息吞吐量极大提高,系统功能复杂,人机界面丰富,具有网络接口等。这类产品的硬件系统集成度非常高,大多采用 32 位 CPU,存储器资源充足,具备了操作系统所需要的运行条件。这类产品的软件系统规模非常大,如果不使用操作系统来管理,则几乎不可能开发成功,电子工程师熟练掌握“基于嵌入式实时操作系统的程序设计技术”,已经是开发这类信息电子产品的必备条件。

搞电子产品设计的技术人员几乎都是“电子类”专业出身的,其“硬件”功底不容置疑,但所学软件课程有限,大多只学过“C 程序设计”、“单片机原理与应用”和“汇编语言程序设计”这几门课,基本没有学过“数据结构”、“算法分析”和“操作系统”等软件课程,大多数工程技术人员还是靠一边工作一边自学来提升自己。各种软件书籍和教材大多采用“一本正经”的论述风格,对于软件功底不深的自学者来说,就像是难啃的天书。作者是电子类专业“科班出身”的大学教师,长期从事电子专业教学,并以程序设计类课程为主。为了教学需要,自己也自学了若干相关的软件课程,深深体会其中之艰辛。近年来,我开始为电子专业研究生上“嵌入式实时操作系统”的课程,对如何给电子专业讲授这门课有了一些体会,萌生了用“电子专业人员”看得懂的叙述风格写一本书的想法。经过几个月的构思,利用暑假的教学空闲时间撰写了本书,了却了这个心愿。

本书共分 15 章。

第 1 章是“绪论”,介绍了嵌入式系统和嵌入式实时操作系统的一些基本知识。

第 2 章是“开发环境”,介绍了本书内容的硬件环境和软件环境。

第 3 章是“任务划分”,介绍了任务划分的基本原则和方法。

第 4 章是“任务设计”,介绍了任务函数设计的基本方法和需要注意的问题。

第 5 章是“中断服务程序设计”,介绍了中断服务程序的设计步骤。

第 6 章是“行为同步”,介绍了几种主要的行为同步手段和实现方法。

第7章是“资源同步”，介绍了访问共享资源的方法。

第8章是“数据通信”，介绍了各种通信手段的使用方法。

第9章是“时间管理”，介绍了时间管理服务的使用场合和方法。

第10章是“采样任务设计”，介绍了各种不同情况下的采样任务设计方法。

第11章是“串行通信任务设计”，介绍了串行通信任务中的风险和正确设计方法。

第12章是“键盘任务设计”，介绍了键盘任务的监控角色和程序设计方法。

第13章是“显示任务设计”，介绍了显示任务与消息队列的配合方法。

第14章是“嵌入式操作系统的剪裁”，介绍了有关的系统常量含义和剪裁方法。

第15章是“设计实例”，介绍了一个完整的设计实例，使读者能够综合理解各章节的内容。

本书第1章、第2章和第15章由吴光文撰写，第3~14章由周航慈撰写。吴光文参与了书中大部分程序的编写，并完成了全部实验程序的测试工作。周航慈负责全书的策划、内容安排、程序修改、文稿修改和审定。

本书在编写过程中得到北京航空航天大学出版社和何立民教授与马广云博士的大力支持，得到周立功先生的鼓励和资助，在此表示衷心感谢！

由于本书涉及知识领域广泛且变化日新月异，加之作者并非“软件专业”科班出身且水平有限及时间仓促，难免有差错和不足之处，甚至包含错误观点，敬请专家和读者指正！力争过一两年再版一次，消除差错，扩充内容，弥补不足。

作者

2006年8月31日

目 录

第 1 章 绪 论	1
1.1 嵌入式系统的发展历史	1
1.2 嵌入式实时操作系统的特点	3
1.3 基于嵌入式实时操作系统的程序设计技术	4
第 2 章 开发环境	6
2.1 嵌入式实时操作系统的选择	6
2.2 CPU 芯片的选择	7
2.3 嵌入式实时操作系统的移植	8
2.4 开发调试环境的建立	9
2.5 工程模板的使用	9
2.5.1 scf 文件组	9
2.5.2 *.h 文件组	10
2.5.3 arm 文件组	12
2.5.4 target 文件组	12
2.5.5 μ C/OS-II 文件组	15
2.5.6 用户文件组	15
第 3 章 任务划分	16
3.1 任务的特性	16
3.1.1 任务的动态性	16
3.1.2 任务的独立性	16
3.1.3 任务的并发性	17
3.2 任务划分的目标	18
3.3 任务划分的方法	18
3.3.1 设备依赖性任务的划分	18

3.3.2	关键任务的划分	19
3.3.3	紧迫任务的划分	21
3.3.4	数据处理任务的划分	23
3.3.5	功能聚合任务的划分	23
3.3.6	触发条件相同任务的划分	23
3.3.7	运行周期相同任务的划分	24
3.3.8	顺序操作任务的划分	24
3.3.9	任务划分小结	25
3.4	任务的可调度性分析	25
第4章	任务设计	27
4.1	任务函数的结构	27
4.1.1	单次执行的任务	27
4.1.2	周期性执行的任务	31
4.1.3	事件触发执行的任务	32
4.2	任务优先级安排	35
4.2.1	任务的优先级资源	35
4.2.2	任务优先级安排原则	36
4.3	任务的数据结构设计	37
4.3.1	与操作系统有关的数据结构	37
4.3.2	与操作系统无关的数据结构	37
4.4	任务设计中的问题	38
4.4.1	公共函数的调用	38
4.4.2	与其他任务的协调	39
4.4.3	共享资源的访问	39
4.4.4	与其他任务进行数据通信	40
4.5	任务的代码设计过程	40
4.5.1	系统总体任务关联图	40
4.5.2	任务的关联分析	41
4.5.3	任务的程序流程图	42
4.5.4	编写任务的程序代码	43
第5章	中断服务程序设计	45
5.1	中断优先级安排	45
5.1.1	中断的优先级资源	45
5.1.2	中断优先级安排原则	45

5.2	不受操作系统管理的中断服务程序	46
5.3	受操作系统管理的中断服务程序	52
5.3.1	中断服务程序的结构	52
5.3.2	中断句柄	53
5.3.3	配置和初始化中断源	53
5.3.4	设计与关联任务的通信手段	54
5.3.5	编写中断服务程序的功能函数	55
第 6 章	行为同步	63
6.1	用于行为同步的通信手段	63
6.1.1	二值信号量	63
6.1.2	计数信号量	64
6.1.3	事件标志组	65
6.1.4	消息邮箱	65
6.1.5	消息队列	66
6.1.6	小 结	66
6.2	行为同步	66
6.2.1	ISR 与任务之间的同步	66
6.2.2	两个任务之间的单向同步	68
6.2.3	两个任务之间的双向同步	74
6.2.4	两个以上任务同步一个任务	77
6.2.5	多个任务相互同步	83
第 7 章	资源同步	90
7.1	关中断	90
7.1.1	保障数据可靠性	91
7.1.2	保障数据完整性	99
7.1.3	关中断小结	106
7.2	关调度	107
7.3	使用互斥信号量	111
7.4	使用计数信号量	115
第 8 章	数据通信	117
8.1	全局变量	117
8.2	内存数据块	121
8.3	消息邮箱	121
8.4	消息队列	128

目 录

第 9 章 时间管理	134
9.1 控制任务的执行周期	134
9.2 控制任务的运行节奏	135
9.3 状态查询	135
9.4 终止周期性任务	136
第 10 章 采样任务设计	141
10.1 使用延时函数控制采样周期	141
10.2 使用定时中断控制采样周期	144
10.3 使用节拍钩子函数进行采样	144
10.4 使用快速定时中断进行采样	148
10.5 被动采样	148
第 11 章 串行通信任务设计	152
11.1 通信协议设计	152
11.2 风险评估	153
11.3 帧缓冲区	154
11.4 数据发送	155
11.5 数据接收	158
第 12 章 键盘任务设计	162
12.1 可靠地获取键盘操作信息	162
12.2 基于菜单操作的监控流程	164
12.2.1 系统功能分析和菜单结构设计	164
12.2.2 监控程序设计	164
12.3 与显示任务的通信	166
第 13 章 显示任务设计	167
13.1 GUI 简介	167
13.2 画面设计	167
13.3 与其他任务的接口	169
13.4 短消息设计	170
13.5 显示任务设计	170
第 14 章 嵌入式操作系统的剪裁	176
14.1 服务功能的剪裁	176
14.1.1 任务管理功能的剪裁	176
14.1.2 通信服务功能的剪裁	177
14.1.3 其他功能的剪裁	179

14.2	数据结构的剪裁	179
14.2.1	与任务有关的数据结构	179
14.2.2	与通信功能有关的数据结构	180
14.2.3	其他参数	180
第 15 章	设计实例	181
15.1	实例简介	181
15.2	硬件系统	182
15.3	任务分析	183
15.3.1	实例任务划分	183
15.3.2	实例任务设计	184
15.4	程序设计详解	186
15.4.1	键盘任务	186
15.4.2	显示任务	193
15.4.3	使用延时函数的采样任务	198
15.4.4	使用快速中断的采样任务	200
15.4.5	使用时钟节拍钩子函数的采样任务	201
15.4.6	使用定时中断的采样任务	203
15.4.7	串行口发送任务	205
参考文献	207

第 1 章

绪 论

在信息技术高度发达的今天,计算机进入了一个充满机遇和挑战的时代——后 PC 时代。随着工业界对智能控制要求的不断提高,形式多样的数字化智能产品应运而生,并且成为替代通用 PC 机进行信息处理的主要部件。在这些部件中都嵌入了微处理器,也就成为所谓嵌入式系统。嵌入式系统已经融入到人们的生活之中,而且无处不在,远远超过了通用 PC 机的应用领域。嵌入式系统带来的工业年产值已超过一万亿美元,在 IT 产业中占有很大的比重。

嵌入式系统是一个包括硬件和软件的完整的计算机系统。根据实际应用的需要,设计者可以选择和剪裁所用的计算机系统的硬件和软件。在一些简单的应用中,可以选择相对简单的微处理器,不使用操作系统,所有软件由设计者自己完成;但是在一些复杂的系统中,所选的微处理器比较复杂,对时间的要求也较为苛刻,在这种情况下,为了更好地管理微处理器,设计者可以选择实时操作系统。操作系统可以屏蔽微处理器的底层硬件,使系统设计者可以在不太了解硬件的情况下,用操作系统提供的接口函数来编写自己的应用程序;这样可以缩短开发的周期,更方便地使用微处理器,以提高微处理器的通用性。例如,我们的个人计算机在进行文件复制时,我们并不知道磁盘具体是怎么工作的,但我们能完成所想要的的数据移动。当然,这种通用性是以牺牲微处理器硬件的特异性为代价的。使用操作系统,我们就不能直接对硬件进行控制,这样势必使硬件的性能受到限制。

本章主要介绍嵌入式系统的发展历史、嵌入式实时操作系统的特点和基于嵌入式实时操作系统的程序设计技术等内容。

1.1 嵌入式系统的发展历史

在介绍嵌入式系统的发展历史以前,先介绍一下什么是嵌入式系统。尽管嵌入式系统的定义在计算机领域不尽相同,但大家普遍能够认可的定义是:“嵌入式系统是以应用为中心,以计算机技术为基础,并且软硬件可剪裁,适用于应用系统对功能、可靠性、成本、体积和功耗有严格要求的专用计算机系统。”嵌入式系统所用的计算机是嵌入到被控对象中的专用微处理器,这与通用计算机不同,它必须适合被嵌入对象的工作环境,如以上提到的:体积、功耗、成本和稳定性等参数;嵌入式系统可以根据实际需要剪裁所用的软硬件。接下来分别从硬件和

第1章 绪论

软件两个方面说明嵌入式系统的发展过程。

嵌入式系统在硬件方面的发展过程指的是：嵌入式系统中内部计算机的发展过程。内部计算机也就是指专用计算机，这种专用计算机在嵌入系统中应用的历史几乎与计算机自身的历史一样长。在通信领域，20世纪60年代晚期，计算机被用于电子电话交换机，称为“存储程序控制”系统，“计算机”这词那时并不普遍。存储程序指内存装有程序和例程信息，存储控制逻辑，而不是将其固化在硬件中，这在当时确实是突破性的。有了存储程序和执行程序的能力，机器就有了灵魂，就从普通的模拟、数字电路分离开来，有了“智力”。

1971年，Intel公司推出了微处理器4004，这是人类历史上的第一颗集成在芯片上的计算机，对于计算机的发展具有划时代的意义。至此，以微处理器为核心的微型计算机以其体积小，价格低，可靠性高等特点，迅速走出机房。

基于高速数值计算能力的微型计算机表现出的智能化水平，引起了控制专业人士的兴趣，他们要求将微型机嵌入到一个对象体系中，实现对象体系的智能化控制，从而导致了嵌入式计算机系统的诞生。从此，计算机进入了通用计算机系统与嵌入式计算机系统两大分支并行发展时代。通用计算机不必兼顾嵌入式系统的要求，集中精力发展通用的硬件和软件，主要用于提升计算机对海量数据的处理能力。通用微处理器从286、386、486到奔腾机，一些商业操作系统已经把通用计算机的性能发展到了尽善尽美的程度。在通用计算机上进行开发的工程师不需要非常了解计算机的硬件结构和工作原理，他们通过操作系统提供的接口函数即可完成开发设计。而嵌入式计算机系统走的是一条完全不同的道路，是单芯片计算机发展道路。最早从事嵌入式开发的人是一些电子工程师，他们成功地把专用计算机芯片作为通用部件嵌入到传统电子产品中，从而使得这些产品具有了智能。所用的专用微处理器也经历了从4位机、8位机、16位机、32位机的发展过程。

嵌入式系统在软件方面的发展过程主要有如下几个阶段：在早期应用阶段，嵌入式系统都是由设计者直接设计外围电路和编写应用程序，这要求设计者有高超的电路设计技术和编程技巧；随着嵌入式微处理器的不断发展和性能的完善，结构复杂的微处理器已经不能不使用操作系统来管理了。因此，一些适用于嵌入式系统的实时操作系统就应运而生。

商品化操作系统在上世纪70年代后期才出现，许多是用汇编语言写成的，并且只能用于特定的微处理器，当微处理器被淘汰时，它的操作系统除非被新微处理器重写；否则就要被淘汰。此时，迫切需要一种在多种微处理器上通用的语言，这种语言既要有高级语言的能力，又能够处理底层数据，甚至能够嵌入汇编程序。C语言正是这种语言。它的诞生伴随着UNIX系统的发展过程，是一个为了写操作系统而产生的语言。其代码的效率和可移植性都无可挑剔，功能强大，深不可测。用C语言编写的操作系统的效率、稳定性、可移植性都提高了很多，尤其是它的可移植性，为微处理器被淘汰时保护软件投资带来了一线生机。

用C语言写成的操作系统今天越来越普遍，软件也就变得可重复使用。一般说来，可重复使用的软件已经占主导地位，而且越做越好。很多商品化的操作系统是在20世纪80年代

出现的,发展到今天,在设计嵌入式系统时设计者有很多商业化的实时操作系统可以选择,如 VxWorks、pSOS、Nucleus、OSE、eCos 和 Windows CE 等;还有一些免费的实时操作系统,如嵌入式 Linux 和 $\mu\text{C}/\text{OS}-\text{II}$ 等。

嵌入式系统的发展日新月异,据专家预计,随着嵌入式产品竞争的进一步加剧,产品更新周期将越来越短。残酷的竞争对产品的更新时间要求十分苛刻,且要求技术十分前沿。为了满足要求,商家就将一块块复杂的 IP 核堆积起来,实现复杂的功能。眼下最好的例子就是手机,越是高档的手机,其内部 SoC(System on Chip)所具有的技术含量越高。在如此复杂的系统中,操作系统是必不可少的;否则,软件将变得不可思议的复杂。

嵌入式系统结合了微处理器或微控制器的系统电路与其专属的软件,可使产品达到最高的性价比。

1.2 嵌入式实时操作系统的特点

操作系统是计算机中最重要的软件,它有很多基本的功能,如:为应用软件提供舞台;为计算机硬件和程序开发提供软件接口;最贴近硬件的软件实现。从程序员的角度看,操作系统可以简化编程环境,并且帮助程序员有效地使用硬件。操作系统直接作用在硬件之上,它为那些需要使用系统资源的其他软件提供接口。这就意味着操作系统可以应用于很多领域,允许不同的应用程序通过使用它提供的资源管理策略来共享硬件资源。资源抽象和资源共享是操作系统的两个关键方面。资源抽象对设计者隐藏了硬件操作的细节,通过提供一些抽象模型,使应用程序在不十分了解硬件的情况下使用计算机硬件资源。资源抽象不仅简化了设计者对硬件资源的控制使用,也限制了对硬件使用的灵活性,通用性的实现是靠牺牲特异性作为代价的。资源共享有两种方法:空分复用和时分复用。操作系统的设计是建立在几种不同的基本策略之上,以提供不同种类的服务,如批处理、分时系统、个人计算、专用计算、网络计算、过程控制和实时计算等。本章描述的内容就是过程控制和实时计算的策略。过程控制专用于单个应用程序,设计者可以决定将资源管理交给应用程序来实现,这就是所谓的基于裸机编程。实时计算基于一种思想,即用户要求系统在规定的时间内对多个请求做出应答,并完成预定工作。

实时操作系统的特点如下:

(1) **执行时间的可确定性是实时操作系统的基本特性。**使用实时操作系统的目的,就是要提高计算机的执行效率。根据实时性的不同,还可以将其分为软实时性系统和硬实时性系统。前者要求计算机在尽可能短的时间内实现用户的请求;后者则要求用户的任务必须在限定的时间内完成,一旦时序和逻辑出现偏差将产生灾难性的后果。所有的实时系统都有自己的实时参数,这是实时操作系统的一个重要的性能指标。组成一个系统的各个任务对于实时性的要求是不相同的,每个任务之间可能还会有一些复杂的关联和同步关系,这为保证系统的

第1章 绪论

实时性带来了很大的困难。因此,实时操作系统所必须遵循的、最重要的设计原则是:采用各种算法和策略,始终保证系统行为的可预测性。可预测性是指在系统运行的任何时刻,在任何情况下,实时操作系统的资源调配策略都能为争夺资源的多个任务合理地分配资源,以满足每个任务的实时性要求。与通用操作系统不同,实时操作系统注重的不是系统的平均表现,而是要求每个任务在最坏情况下都能满足其实时性要求;也就是说,实时操作系统注重的是个体表现,更准确地讲是个体最坏情况表现。

(2) **可剪裁性和可固化是嵌入式实时操作系统的重要特性。**用户可以根据自己设计的需要对操作系统进行剪裁,是实时操作系统在嵌入式领域广泛应用的重要前提之一。从嵌入式系统的概念可以得知:“嵌入式系统是软件可剪裁的系统”,剪裁的目的就是适用于应用系统对功能、可靠性、成本、体积、功耗的严格要求。嵌入式实时操作系统不同于通用计算机的操作系统,嵌入式设备无处不在,所用的微处理器也千差万别,导致了这些微处理器的可用资源差别很大,一个微处理器很难像通用计算机那样兼容别的微处理器。一个广泛应用的嵌入式实时操作系统应该具有很好的可剪裁性,这样才能够将它移植到很多形态各异的微处理器上。这就涉及到可固化的问题,一个好的实时操作系统应该是为底层硬件和用户提供一个桥梁,用户能够方便地使用,并且像调用库函数那么方便地使用操作系统提供的一些系统服务函数。

(3) **多任务占先式是实时操作系统的基本特性。**多个任务共同分享硬件系统资源,每个任务之间彼此独立,且根据任务的重要性不同给不同的任务分配不同的优先级。优先级越高的任务,越容易得到CPU的使用权。这主要是保证任务的实时性和充分地使用计算机资源。

(4) **对稳定性和可靠性要求特别高。**很多实时系统是用在十分重要的工业控制过程中,甚至是航空、航天项目中,关系到生命财产的安全,系统崩溃的后果是不堪设想的。所有实时操作系统的各个部分都必须经过严格测试,以保证系统的可靠性和稳定性。

1.3 基于嵌入式实时操作系统的程序设计技术

早期的嵌入式实时系统,都是由电子工程师设计的,他们用接近计算机的汇编语言编写了大量的优秀应用软件,用于管理自己设计的嵌入计算机的硬件电路系统。这种应用软件只能应用于特定的一种微处理器,换另一种微处理器时,这种软件就不能应用了,需要重新设计。而操作系统的研究和应用本该是计算机工程师研究的课题。前者被认为是硬件工程师,后者被认为软件工程师,二者没有什么联系。随着计算机技术的不断发展,越来越多的功能复杂的微处理器出现了,应用传统的基于裸机编写应用程序的做法已经不能很好地管理微处理器。而且,为了节省人力、物力,需要出现一些类似通用计算机的操作系统之类的东西来管理微处理器。这就需要硬件设计工程师了解操作系统,软件工程师了解硬件系统;否则,实时操作系统就不能进入嵌入式系统程序设计领域。

嵌入式实时操作系统给设计者提供了一个操作的平台,用户只须根据实际的需要定制一

些实际的任务,就可以完成系统的设计。尤其是使用已经移植了操作系统的微处理器进行开发时,设计过程就变得更加简单。设计者甚至不需要详细地了解微处理器的硬件结构和具体的汇编指令,就可以方便地应用C语言调用操作系统提供的服务函数进行编程;调用系统服务函数就像调用C语言的库函数那么方便,而且实时性、稳定性和可靠性都有很高的保障;需要设计者考虑的问题就会大大地减少,开发周期也会相应地缩短。

但是,使用操作系统进行程序设计也带来了一些副作用。比如,会增加系统代码的长度而占用系统更多的时间和空间资源,降低了微处理器的特异性取而代之为通用性。随着微处理器性能的不断改进,系统的资源短缺问题现在已经得到缓解,而且开发一些大的应用程序也需要用实时操作系统来实现。应用操作系统编写程序已经是大势所趋,这对传统基于裸机编程的方式提出了挑战。

实时操作系统也不是一把万能的钥匙,有了操作系统,一切都迎刃而解的想法是错误的。设想一下,实现同样的功能用操作系统和不用操作系统,对CPU来说意味着什么?无疑是增大了开销,对同一个CPU来说,干的事情越多就越容易出错,这是永远不变的道理。因此,在使用实时操作系统的时候要慎重,如果一些低档的微处理器用了实时操作系统,就会使本来简单的问题复杂化。之所以说基于实时操作系统编程比基于裸机编程简单,是因为现在一些高档的微处理器已经复杂到必须用操作系统才能管理的程度了;换句话说,如果不用操作系统,就很难管理微处理器或者说很难充分实现它的功能。操作系统为用户与CPU之间建立了桥梁,用户现在考虑的问题是建立在操作系统的基础上,而不是建立在底层硬件的基础之上。这正如基于Windows系统编程,只要知道一些接口函数,就可以让操作系统实现许多功能。如果不是基于Windows操作系统而是在一个没有操作系统的计算机上编写程序,那么将是很困难的事情。应用操作系统编写实时程序,就要求掌握实时操作系统的原理。

在嵌入式实时操作系统的设计中,各“模块”是以任务的形式存在的,各任务的重要性不同,它们的优先级也肯定不同。怎样划分任务和分配任务的优先级就是一个非常关键的问题,如果划分不好,就不能实现实时性,甚至造成灾难性的后果。各任务之间的通信和同步,也是很需要技巧的一件事,弄不好就会发生讨厌的任务死锁情况而导致系统崩溃。如果对实时操作系统的时钟机制不清楚,就很难进行准确的延时,使微处理器总是疲于奔命,而导致实时性能降低。还有一些关键问题:如代码的重入、优先级反转等问题都需要引起使用实时操作系统的开发者注意。对于以上提到的一些问题,在本书里都有详细的讲解和应对方案。

本书就是介绍如何应用实时操作系统进行嵌入式系统设计,书中选用开源代码的嵌入式实时操作系统 $\mu\text{C}/\text{OS}-\text{II}$ 为例,说明基于实时操作系统编写应用程序的方法。为了使读者更好地理解书中所讲的理论,作者采用广州周立功单片机发展有限公司(以下简称周立功公司)的MagicARM2200实验箱在ARM Executable Image for UCOSII(MagicARM2200)模板上做了一个数字示波器,所有代码都经过调试并且有详细的注释,简单易懂。详细内容将在第15章中介绍。

第 2 章

开发环境

在基于实时操作系统的程序设计中,开发环境主要包括操作系统和微处理器两部分内容,这两部分都是使用者根据实际设计的需要来选择的。在本书中,为了方便说明,实时操作系统选用 $\mu\text{C}/\text{OS}-\text{II}$,微处理器选择 ARM7。

2.1 嵌入式实时操作系统的选择

在嵌入式系统领域,有很多实时操作系统可以选择,如嵌入式 Linux、Windows CE 和 VxWorks 等大批优秀的实时操作系统。它们当中有些是商业性的,代码不公开,有些代码比较长,不利于学习掌握。在众多的实时操作系统中,我们选择 $\mu\text{C}/\text{OS}-\text{II}$ 作为例子。

$\mu\text{C}/\text{OS}-\text{II}$ 是 Jean J. Labrosse 设计的完整、可移植、可固化、可剪裁的占先式实时多任务内核,代码绝大部分是用标准 C 语言编写的。到现在为止, $\mu\text{C}/\text{OS}-\text{II}$ 已经被移植到了 40 多种微处理器上,这些微处理器从 8 位到 64 位都有,在各个领域中已经有数千个应用使用该内核。它有很好的稳定性和可靠性。2000 年 7 月, $\mu\text{C}/\text{OS}-\text{II}$ 在一个航空项目中取得了美国联邦航空管理局用于商用飞机、符合 RTCA DO-178B 标准的认证。

$\mu\text{C}/\text{OS}-\text{II}$ 是一个公开源代码的操作系统,代码很短(约 5500 行)。与一些商用的公开源代码的实时内核相比,它的源代码清晰易读,结构协调,注释详尽。现在很多高等学校将 $\mu\text{C}/\text{OS}-\text{II}$ 应用于实时操作系统教学。

$\mu\text{C}/\text{OS}-\text{II}$ 具有可移植性,绝大部分源代码是用 C 语言编写的;而与微处理器密切相关的部分用汇编语言编写。但这一部分内容很少,方便使用者将内核移植到自己的微处理器上,移植 $\mu\text{C}/\text{OS}-\text{II}$ 很简单,耗时为几小时到几天不等,完全取决于操作者的水平和所使用微处理器的复杂程度。实践证明, $\mu\text{C}/\text{OS}-\text{II}$ 可以在很大一部分微处理器上运行。用户一旦移植了 $\mu\text{C}/\text{OS}-\text{II}$ 到自己的微处理器,就可以在此基础上编写应用程序,并把 $\mu\text{C}/\text{OS}-\text{II}$ 作为自己产品的一部分固化在里面。

$\mu\text{C}/\text{OS}-\text{II}$ 是可剪裁的,用户可以根据自己的需要决定使用内核中的多少成分。剪裁可以通过定义宏来实现,简单易行。可剥夺性和多任务也是 $\mu\text{C}/\text{OS}-\text{II}$ 的重要特性,它总是运行处于就绪态的优先级最高的任务。 $\mu\text{C}/\text{OS}-\text{II}$ 可以管理 64 个任务,用户可以使用其中的

56 个。

绝大多数 $\mu\text{C}/\text{OS}-\text{II}$ 的函数调用和服务的执行时间具有可确定性。 $\mu\text{C}/\text{OS}-\text{II}$ 提供了很多系统服务,例如信号量、互斥型信号量、事件标志、消息邮箱、消息队列、信号量、块大小固定的内存申请与释放及时间管理函数等。中断可以挂起 $\mu\text{C}/\text{OS}-\text{II}$ 正在执行的任务,中断退出后执行的任务是当前就绪任务中优先级最高的任务,而不一定是被中断的任务。这就要求每个任务有自己的堆栈,保存任务切换和被中断时环境的上下文变量。

基于 $\mu\text{C}/\text{OS}-\text{II}$ 具有以上优点,我们以它为例来介绍基于实时操作系统的编程方法,就更容易被人们理解和接受。

2.2 CPU 芯片的选择

在我们国家,嵌入式系统的入门教育一般是从学习 8 位的 51 单片机开始的。因为该单片机发展的技术比较成熟,简单易学,所以通常用作应用电子专业的重要课程来学。但是,该单片机的资源相对缺乏,如果在上面应用操作系统,就显得资源比较紧张。这就需要高超的技巧和经验;否则很容易出错。与 51 单片机比较,32 位的 ARM 系列微处理器就显得功能非常强大,处理速度也快,内存资源相当丰富,具有使用实时操作系统的有利条件。

ARM 是 Advanced RISC Machines 的缩写。ARM 公司是微处理器行业的一家知名企业。该企业设计了大量的高性价比的、低功耗的精简指令集微处理器。在设计上,ARM 微处理器具有外形小,性能高,成本低和功耗低的特点。ARM 微处理器适用于很多领域,比如嵌入式控制、消费/教育类多媒体、DSP 和移动式应用等。此外,ARM 将其技术授权给世界上许多著名的半导体、软件和 OEM 厂商,每个厂商得到的都是一套独一无二的 ARM 相关技术及服务。利用这种合伙关系,ARM 很快成为许多全球性 RISC 标准的缔造者。目前,总共有 30 家半导体公司与 ARM 签订了硬件技术使用许可协议,其中包括 Intel、IBM、LG 半导体、NEC、SONY、飞利浦和美国国家半导体这样的大公司。至于软件系统的合伙人,则包括微软、升阳和 MRI 等一系列知名公司。ARM 微处理器具有 32 位和 16 位两种指令集,这两种指令集之间进行切换是没有开销的。

ARM 微处理器目前有 6 个产品系列: ARM7、ARM9、ARM9E、ARM10E、SecurCore 和 ARM11。本书选择周立功公司的 MagicARM2200 实验箱作为硬件电路系统。在该实验箱中,使用的微控制器为 ARM7 系列的 LPC2290,系统时钟可达 60 MHz,有 2 MB 的 NOR FLASH、8 MB 的 RAM、16 MB 的 NAND FLASH、256 字节的 E²PROM,16 位总线输出。

总之,在 ARM 系列微控制器上有足够的系统资源运行 $\mu\text{C}/\text{OS}-\text{II}$,且在国内比较普及,故本书选择 ARM 系列微控制器作为硬件平台。