维普资讯 http://www.cqvip.com

嵌入式操作系统 μC/OS 的运行机制

石世光[1],陈云洽[2],叶奇明[3]
(1.电子科技大学 中山学院 计算机工程系,广东 中山 528403;
2.中山大学 电子与通信系,广东 广州 510275;
3.茂名学院 师范学院 计算机系,广东 茂名 525000)

摘 要:目前对嵌入系统的引入和应用,已越来越成为广大科研人员关心的问题。而嵌入式操作系统是嵌入系统应用的核心,文中主要探讨了源代码开放的嵌入式操作系统 μC/OS 的内核运行机制。首先分析任务块的基本结构,然后阐述了任务管理的实现过程,最后着重讨论了任务启动过程以及在两种不同情况下的任务切换工作原理,为嵌入式系统开发应用提供指导作用。

关键词:嵌入式操作系统;μC/OS;运行机制;任务
中图分类号:TP316.2 **文献标识码**:A **文章编号**:1673-629X(2006)08-0085-03

Operation Mechanism of Embedded Operating System μC/OS

SHI Shi-guang[1], CHEN Yun-qia[2], YE Qi-ming[3]
(1.Dept. of Computer Eng., Zhongshan Coll. of Univ. of Electronic Sci. and Tech. of China, Zhongshan 528403, China;
2.Dept. of Electronics and Communication Eng., Zhongshan Univ., Guangzhou 510275, China;
3.Dept. of Computer Eng., Teachers School of Maoming Coll., Maoming 525000, China)

Abstract: Nowadays it has already become a noticeable problem for many scientific researchers to introduce and apply the embedded operating system. And the embedded operating system is the core of the embedded application. Operation mechanism of the emmedded operating system μC/OS is mainly described in this paper. Firstly, analyses the fundamental structure of task control block. Secondly, expatiates on the realization of task management. Finally, dicusses the process of task start and both of the task-switch's operation principle in detail. Therefore it will provide good help for the researcher of the embedded operating system.

Key words: embedded operating system; μC/OS; operation mechanism; task

0 引 言

当前的嵌入式操作系统层出不穷,而且它们以其各自不同的特色广泛分布于通信、航空航天、汽车、医疗、电子消费等不同的领域。它们性能卓越、功能完备、技术成熟、服务周全,通过使用这样的操作系统,可以大大缩短产品的开发周期,降低开发成本,同时产品的品质也能够大大提升。对于一名从事嵌入式系统开发的研究人员或工程师来说,了解并熟悉嵌入式操作系统是最基本的。

1 使用 μC/OS 操作系统的目的

μC/OS 是专门为中低端嵌入式应用设计的可剥夺型实时操作系统内核,主体用标准的 ANSI C 语言写成,可移植性极好。目前已被成功地移植到 MCU, DSP, CPU 上,包括 8 位、16 位、32 位及 64 位。它包括了一个操作系统最基本的一些特性,如任务调度、任务通信、内存管理、中断管理等,而且代码完全开放、结构简单明了、代码风格严谨,非常适合初涉嵌入式操作系统的人士学习,它可以让人们以最快的速度来了解操作系统的概念、结构和模块工作原理,并由浅入深逐步推广到商用操作系统上。同时对于那些对操作系统感兴趣的爱好者来说,μC/OS 的浅显易懂,给人们提供了一个很好的研究标本,这就是要学习 μC/OS 的一个很重要的原因。

μC/OS 在高校教学使用是不需要申请许可证的,只有将 μC/OS 的目标代码嵌入到商业产品中,才需要购买销售许可证。自 1992 年以来,μC/OS 已经被应用到数以百计的产品中,性能完全可以与商业产品相媲美,而且通过了美国联邦航空局(FAA)商用航行器认证,其性能优良稳定并且免费,这是要学习 μC/OS 的另一个很重要的原因[1]。

2 μC /OS 内核运行的机制

简单地讲,μC /OS 内核运行机制的基本思路是:近似地让最高优先级的就绪任务处于运行状态。下面就详细谈谈其工作的实现机制。

收稿日期:2006-03-14
作者简介:石世光(1974-),男,湖北大冶人,助教,研究方向为嵌入式系统开发。

维普资讯 http://www.cqvip.com

2.1 任务控制块基本结构

```
Struct OS_TCB {
OS_STK * OSTCBStkPtr;// 一个指向当前任务栈顶的指针;
struct os_tcb * OSTCBNext;
struct os_tcb * OSTCBprev;// OSTCBNext 和 OSTCBPrev 用于任务控制块的双重链接;
INT16U OSTCBDly;// 用于任务延时或超时限制,每中断一次减一,为 0 表示任务准备就绪;
INT8U OSTCBStat;// 任务的状态字,0 表示就绪态;
INT8U OSTCBPrio;// 任务的优先级,值越小,优先级越高。
INT8U OSTCBX,OSTCBY,OSTCBBitX,OSTCBBitY;// 用于加速任务进入就绪态的过程或进入等待事件发生状态的过程。
}[2]
```

从上述结构可以看到,每一个任务都建立相应的任务控制块 OS_TCB(Task Control Block),任务控制块的开头,即 OS_TCB 的 0 偏址内存单元中存放任务堆栈指针 OSTCBStkPtr,指向任务堆栈;通过双向链表 OSTCBNext 和 OSTCBPrev 将不同的任务控制块 OS_TCB 连起来,任务控制块还包括任务延时、状态、优先级等信息。

系统初始化时,将根据任务的多少来建立任务控制块链表。任务创建函数分配的一块内存,建立任务堆栈,用来保存寄存器的值和该任务的代码地址;再把堆栈的地址存入任务控制块,从而把任务程序代码、任务堆栈和任务控制块联系在一起。操作系统内核可以通过任务控制块找到任务堆栈,从堆栈中取得任务代码[3]。

2.2 任务管理的实现

任务管理主要是通过对任务控制块(TCB)的管理来实现的。当一个任务建立时,μC/OS 系统为其所对应的 OS_TCB 赋值;当任务的 CPU 使用权被剥夺时,系统用 OS_TCB 来保存该任务的状态;当任务重新得到 CPU 使用权时,系统就可通过任务控制块来使任务从被中断处继续执行下去。

每个任务都被分配一个任务控制块 TCB,按照任务优先级的次序将各个任务 TCB 的指针放入任务优先级表数组中,这样通过该数组就可访问各个任务。

同时有一个 TCB 指针 OSTCBCur 指向当前任务的 TCB,另有一个 TCB 指针 OSTCBHighRdy 指向具有更高优先级的并且将要执行任务的 TCB,如果两者任务优先级不等说明要进行任务切换,完成了任务切换后,CPU 就开始运行更高优先级任务。

2.3 任务启动过程

使就绪状态的任务开始运行的函数叫做 OSStart()。在调用 OSStart()之前,必须至少已经建立了自己的一个任务。前面谈到过,由于操作系统是按照"比较任务优先级,找到高优先级任务,然后再使之运行"的思路来启动任务的,所以就有了任务就绪表和优先级判定表。就绪表中有两个变量(OSRdyGrp 和 OSRdyTbl[])用来存放每个任务的就绪标志,通过分组与优先级表中的项实现一一对应,进而确定进入就绪态的优先级最高的任务。任务启动过程如图 1 所示。

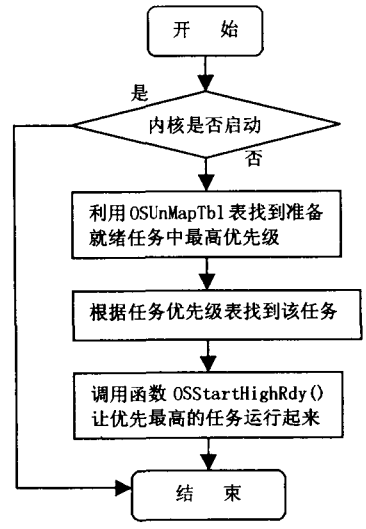

图 1 OSStart( )的流程

2.4 任务切换原理

任务切换指保存当前任务的上下文,并恢复需要执行任务的上下文的过程。当发生任务切换时,当前正在运行的任务的上下文就需要通过该任务的任务控制块保存起来,并把需要投入运行的任务的上下文从对应的任务控制块中恢复出来[4]。

任务的切换有两种情况:一种是任务级切换,另一种是中断级切换。

所谓任务级切换,即当前任务运行完毕,下一个准备就绪的最高级优先任务进行任务切换。

1) 任务级切换的大致流程如下:

◇当前任务完成;

◇OSSched()任务调度,选出高优先级就绪任务;

◇OS_Task_Sw()任务切换;

◇运行高优先级任务。

其中 OS_Task_Sw()的工作过程比较简单:

(1)保存寄存器值入当前任务堆栈;

(2)修改当前任务控制块指针 OSTCBCur 和当前任务优先级 OSProCur,指向就绪任务;

(3)恢复就绪任务堆栈中的值到寄存器中,完成切换。

而 OSSched()用于任务调度,其流程如图 2 所示[5]。

2)中断级切换,即是在中断程序处理事务过程中,可能有一些任务运行就绪,重新整理和搜索任务就绪表,找到最高优先级任务进行切换。

中断级切换的大致流程:

◇调用函数 OSTickISR(),完成中断入口工作;

◇调用 OSIntExit()选出高优先级就绪任务;

◇利用 OSIntCtxSw()完成任务切换;

◇运行高优先级任务。

维普资讯 http://www.cqvip.com

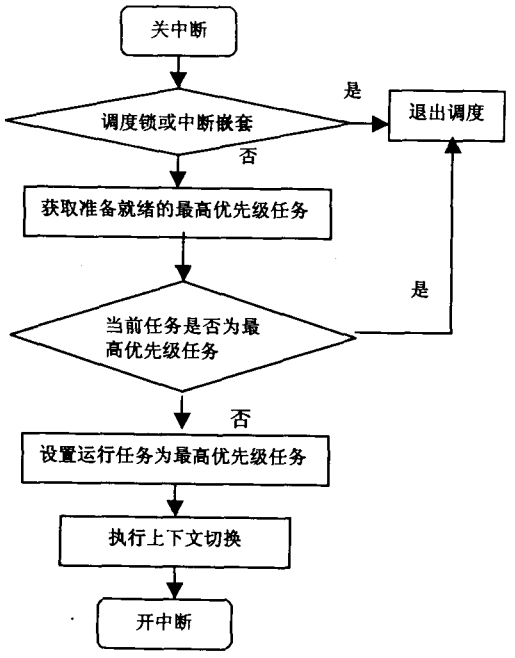

图 2　OSSched()函数流程

对于定时中断函数 OSTickISR(),它主要负责中断进入时保存处理器寄存器内容,完成任务切换退出时恢复处理器寄存器内容并返回,相当于中断服务程序的入口。

中断级的上下文切换是 OSIntExit()通过调用 OSIntCtxSw()来执行切换功能。下面重点谈谈 OSIntExit()函数过程。与任务级切换不同的是中断级切换时,中断返回函数将决定是返回到被中断的任务,还是让优先级最高任务运行。其流程如图 3 所示。

3　结束语

以上是一些经验与体会,笔者曾在研制电力谐波检测仪的项目中,根据嵌入式操作系统 μC/OS 的内核运行机制原理,在 TI DSP F2407 芯片上移植取得很好的效果。现在网站上有很多关于各种类型芯片的移植程序,但不仅要知其然,更要知其所以然,才能做到举一反三、灵活运用,所以有必要掌握 μC/OS 内核的运行基本原理,从而能深入理解移植程序,达到了解嵌入式操作系统原理的目的,为从事各种嵌入式操作系统的研究打下坚实的基础。

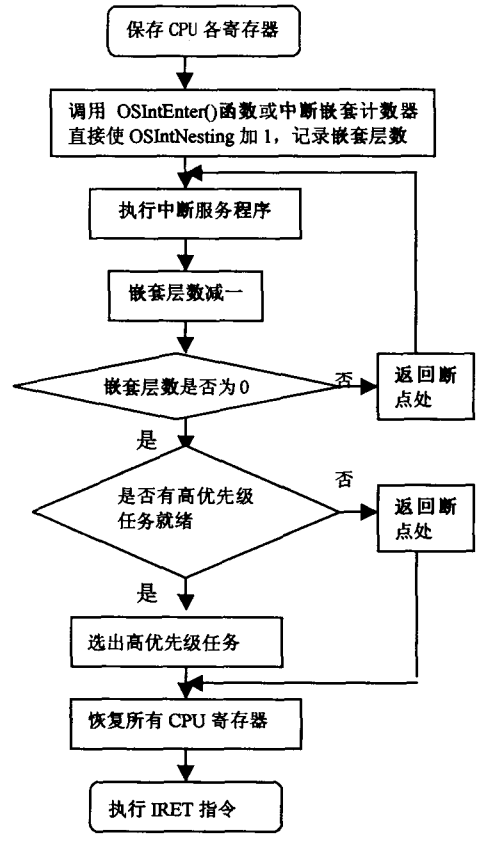

图 3　中断返回函数流程

参考文献:

[1] 邵贝贝.μC /OS 源码公开的实时嵌入式操作系统[M].北京:中国电力出版社,2001.

[2] 崔树林.嵌入式系统通用的应用软件结构研究[J].单片机与嵌入式系统应用,2003(8):9-10.

[3] 王劲松.嵌入式操作系统 uc/os 的内核实现[J].现代电子技术,2003(8):48-49.

[4] 罗　蕾.嵌入式实时操作系统及应用开发[M].北京:北京航空航天出版社,2005.

[5] 王克星.实时多任务操作系统的开发与应用[J].计算机工程与应用,2003(5):132-134.

(上接第 84 页)

3　小　结

决策树的建立可以使数据规则可视化,结构清晰,所以在对知识的分类中常常用决策树表示。利用典型 ID3 算法构造决策树,按照信息增益最大的原则,相对抽象,计算繁琐;利用粗集近似精度选择根结点时计算简单,而且分类可以使决策树和粗集更容易理解。

参考文献:

[1] 朱　明.数据挖掘[M].合肥:中国科学技术大学出版社,2002.

[2] Quinlan J R. Induction of decision trees[J]. Machine Learning,1986(1):81-106.

[3] 王静红,王熙照,邵艳华,等.决策树算法的研究及优化[J].微机发展,2004,14(9):30-32.

[4] 尹阿东,郭秀颖,宫　雨,等.增量决策树算法研究[J].微机发展,2005,15(2):63-66.

[5] 曾黄麟.粗糙集理论及其应用[M].重庆:重庆大学出版社,1998.

[6] 董祥军,宋瀚涛,姜　合,等.时态关联规则的研究[J].计算机工程,2005,15:24-26.