嵌入式操作系统任务切换方法对比分析※

孙利锋

（贵州大学 电气工程学院，贵阳 550000）

摘要：嵌入式操作系统在很多领域得到应用。由于嵌入式实时操作系统支持多任务，使得程序开发更加容易，在便于维护的同时还能提高系统的稳定性和可靠性，所以逐步成为嵌入式系统的重要组成部分。本文介绍4种嵌入式实时操作系统VxWorks、μClinux、μC/OS-II和Windows CE，对多任务的调度切换进行了分析比较。

关键词：嵌入式操作系统；任务切换；VxWorks；Linux；μC/OS-II；Windows CE

中图分类号：TP316.2　　**文献标识码**：A

Comparison and Analysis of Embedded Operating System on Context Switching※

Sun Lifeng

(Department of Information Electronic Engineering, Guizhou University, Guiyang 550000, China)

Abstract: Embedded operating system is applied in many fields. Because embedded real-time operating system supports multiple tasks, it makes the program development easier, stabler and more reliable. Because of that, it becomes an important part of the embedded system. The paper introduces four kinds of embedded real-time operating systems, i.e. VxWorks, μClinux, μC/OS-II and Windows CE. Task scheduling in switching is analyzed and compared.

Key words: embedded OS; context switch; VxWorks; Linux; μC/OS-II; Windows CE

引　言

嵌入式系统在航天、军事、工控以及家电等方面得到了广泛应用。大量的嵌入式系统具有实时性的要求，但是由于体积、能耗、价格等方面的约束，其处理器速度往往比较慢，存储器容量也有限。而传统的实时操作系统难以简单地移植到嵌入式系统中，所以需要重新开发针对嵌入式系统特性的实时操作系统。任务调度策略是实时系统内核的关键部分，如何进行任务调度，使得各个任务能在其期限之内得以完成，是实时操作系统的重要研究领域。而不同的操作系统对任务调度的机制也有所不同，本文对目前比较流行的操作系统——VxWorks、μClinux、μC/OS-II、Windows CE的任务切换机制进行分析和比较。

1　操作系统介绍

1.1　VxWorks

VxWorks是美国WindRiver公司的产品，是目前嵌入式系统领域中应用很广泛、市场占有率比较高的嵌入式操作系统。VxWorks实时操作系统由400多个相对独立、短小精悍的目标模块组成，用户可根据需要选择适当的模块来裁剪和配置系统；提供基于优先级的任务调度、任务间同步与通信、中断处理、定时器和内存管理等功能，内建符合POSIX（可移植操作系统接口）规范的内存管理，以及多处理器控制程序；具有简明易懂的用户接口，在核心方面甚至可以微缩到8 KB。

1.2　μC/OS-II

μC/OS-II是在μC/OS的基础上发展起来的，是美国嵌入式系统专家Jean J. Labrosse用C语言编写的一个结构小巧、抢占式的多任务实时内核。μC/OS-II能管理64个任务，并提供任务调度与管理、内存管理、任务间同步与通信、时间管理和中断服务等功能，具有执行效率高、占用空间小、实时性能优良和可扩展性强等特点。

1.3　Linux

Linux是一种自由的Unix类多用户、多任务操作系统，可运行在Intel 80386及更高档次的PC、ARM、DEC Alpha等多种计算机平台上，已经成为应用广泛、可靠性高、功能强大的计算机操作系统。

1.4　Windows CE

微软Windows CE是一个开放且多样化的32位嵌入式操作系统。其设计目的是为符合广泛的智能设备的需求，例如从企业工具（如工业控制器、通信集线器和收款机

系统)到电子消费性产品(如摄影机、电话和家庭娱乐设备等),提供自动控制、视听娱乐、行动计算、终端机等各个应用领域一个稳定、实时及多任务的操作系统。

2 任 务

2.1 任务切换概述

上下文切换(context switch),其实际含义是任务切换,或者 CPU 寄存器切换。当多任务内核决定运行另外的任务时,它保存正在运行任务的当前状态,也就是 CPU 寄存器中的全部内容。这些内容被保存在任务自己的堆栈中,入栈工作完成后就把下一个将要运行的任务的当前状况从该任务的栈中重新装入 CPU 寄存器,并开始下一个任务的运行,这一过程就是 context switch。

每个任务都是整个应用的一部分,都被赋予一定的优先级,有自己的一套 CPU 寄存器和栈空间,如图 1 所示。

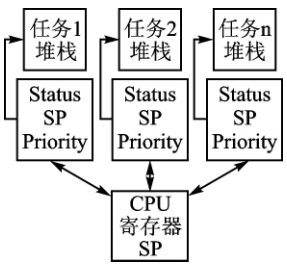

图 1 多任务

2.2 任务的切换与调度

μC/OS-II 是可抢占实时多任务内核,它总是运行优先级最高的就绪任务,不支持时间片轮转调度法,每个任务的优先级要求不一样,且是唯一的。它有 5 种状态,如图 2 所示。

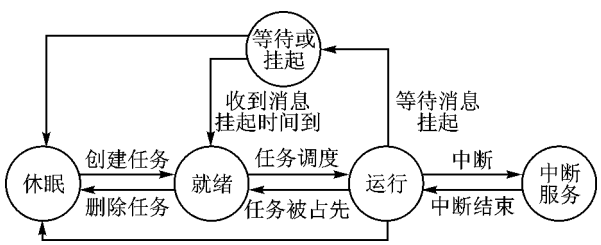

图 2 任务的状态

当一个任务在运行状态中时,如果没有关闭中断,就有可能被中断打断,去执行中断服务子程序 ISR。执行完后内核要判断此时是否有更高优先级,新的任务就绪,如果有则原有的任务被抢占,实现了任务的切换。

当一个任务在运行状态中时,调用 OSTimeDly()或 OSTimeDlyHMSM()函数,该任务进入等待状态,一直到延时时间到,这 2 个函数立即强制执行任务切换,让下一个优先级最高的就绪任务运行。当然,如果运行的任务需要等待某一事件的发生,可以调用一些函数(如 OSFlagPend()、OSSemPend()、OSMutexPend()、OSMboxPend()、OSQPrnd()等)挂起该任务,来实现任务的切换。

实际的任务切换是调用 OS_TASK_SW()函数。OS_TASK_SW()是一个宏,是在 μC/OS-II 从低优先级切换到高优先级任务时须用到的。OS_TASK_SW()总是在任务级代码中被调用。另一个函数 OSIntExit()用在中断服务子程序 ISR 中。当中断任务子程序使更高优先级任务进入就绪态时,OSIntExit()完成任务切换功能,任务切换只是简单地将处理器的寄存器保存到将被挂起的任务的堆栈中,并且从堆栈中恢复要运行的更高优先级的任务。

任务切换 OS_TASK_SW():

```
void OSCtxSw(void){
  将 R1,R2,R3,R4 推入当前堆栈;
  OSTCBCur->OSTCBStkPtr=SP;
  OSTCBCur=OSTCBHighRdy;
  SP=OSTCBHighRdy->OSTCBStkPtr;
  将 R4,R3,R2,R1 从新堆栈中弹出;
  执行中断返回指令;
}
```

μC/OS-II 总是运行进入就绪态任务中优先级最高的任务,确定哪个任务优先级最高,以及下面该哪个任务运行。这一工作是由调度器完成的,所以任务调度的工作就是:查找准备就绪的最高优先级的任务并进行上下文切换。该工作由函数 OSSched()完成。中断级的调度由 OSIntExt()完成。代码如下:

```
void OS_Sched(void){
#if OS_CRITICAL_METHOD==3
                                    //保护和恢复处理器状态字 PSW
  OS_CPU_SR cpu_sr;
#endif
  INT8U y;
  OS_ENTER_CRITICAL();   //进入临界中断
  if((OSIntNesting==0)&&(OSLockNesting==0)){
  /*判断调用是否来自中断服务子程序或者至少调用了一次
    上锁函数*/
    y=OSUnMapTbl[OSRdyGrp];//找出就绪态任务最高的任务
    OSPrioHighRdy=(INT8U)((y<<3)+OSUnMapTbl
[OSRdyTbl[y]]);
    if(OSPrioHighRdy !=OSPrioCur){
      OSTCBHighRdy=OSTCBPrioTbl[OSPrioHighRdy];
      OSXtxSwCtr++;
      OS_TASK_SW();
    }
  }
  OS_EXIT_CRITICAL();      //退出临界中断
}
```

在 Linux 系统中,任务的上下文切换和调度比较复杂。Linux 的上下文切换功能是由 context_switch()函数完成的。代码如下:

```
Static inline
```

```
Task_t *context_switch(runqueue_t *rq, task_t *prev, task_t
*next){
  struct mm_struct *mm=next->mm;
  struct mm_struct *oldmm=pre->active_mm;
  switch_mm(oldmm,mm,next);
  switch_to(pre,next,prev);
  return prev;
}
```

context_switch()完成了 2 个工作：

① 切换虚拟内存映射，即负责把虚拟内存从被切换下来的进程映射到新进程中，该功能由函数 switch_mm()实现。

② 切换进程的寄存器状态，即负责从一个进程的处理器状态切换到新进程的处理器状态，该功能由函数 switch_to()实现。

在多任务系统中，都会提供一个系统函数来进行进程（任务）间切换，综合来说，它们有两种进程（任务）切换方式：

① 由进程（任务）本身直接调用任务切换函数进行进程（任务）切换。在当前进程（任务）因为不能获得必需的资源而立即被堵塞时，就由进程（任务）本身直接调用进程（任务）切换函数进行进程（任务）间调度。在 Linux 中可以直接调用 schedule()函数来实现。

② 延迟调用任务切换函数进行进程（任务）切换。此方式是把当前进程（任务）设置一调度标志而以延迟方式调用任务切换函数进行进程（任务）切换。在 Linux 系统中，总是在恢复用户态进程执行之前，检查这一调度标志，在这里标志是 need_resched，如果有这一标志，就调用调度函数进行进程切换。

此种情况主要包括以下几种：

① 当前进程用完了它的 CPU 时间片，由 scheduler_tick()函数完成 schedule()的延迟调用。

② 当一个被唤醒进程的优先级比当前进程优先级高时，由 try_to_wake_up()函数完成 schedule()的延迟调用。

③ 当发出系统调用 sched_setscheduler()时。在这些情况中，主要由于系统调用或中断而进入内核态，或者当前进程本来在内核态时，返回用户态时发生的。

调度策略与优先级列表见本刊网站 www.mesnet.com.cn——编者注。

在 VxWorks 系统中，任务的优先级为 0～255。任务有 4 种状态：就绪态、悬置态、休眠态和延迟态，如图 3 所示。

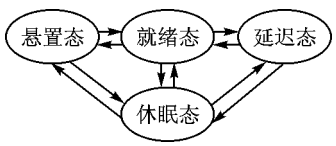

图 3 任务状态迁移

内核缺省调度机制为基于优先级的抢占式调度。采用这种机制，系统把处理机分配给优先级最高的进程，使之执行。一旦出现优先级更高的进程时，该任务被剥夺 CPU 使用权，而去执行优先级更高的任务。而在相同优先级的多个任务之间，采用时间片轮转调度机制。采用这种机制，当一个任务到达时，它被安排在轮转队列的后面，等待分配给自己的时间片的到来，如果在时间片内没有结束，则在等待属于自己的时间片的到来，直到任务完成。

在 VxWorks 系统中，对于优先级相同的任务，如果状态为 Ready，则可以通过时间片轮转方式公平享有 CPU 资源。轮转调度法给处于就绪态的每个同优先级的任务分配一个相同的时间片，该时间片的大小由系统调用 KernelTimeSlice 决定。

在 Windows CE 系统中，Windows CE 3.0 之后，系统支持的优先级增长到 256 个，0 优先级级别最高，255 优先级级别最低。0～247 的优先级属于实时性优先级，248～255 的优先级一般分配给普通应用程序。Windows CE.NET 采用基于动态优先级的抢占式多任务机制，越重要的任务，优先级越高。Windows CE.NET 在任务调度中采用任务优先级制、优先级动态调整机制和抢占式调度，都是为了最大限度地满足系统的实时性要求。对于一个优先级只有一个任务的简单系统内核，上述的 3 种调度足以满足要求，但对于 Windows CE.NET 这样复杂、高性能的多任务实时内核，由于多个任务允许公用一个优先级，则相同优先级的任务要采用 Windows CE.NET 提供的时间片轮转法实现。具体实现如图 4 所示。

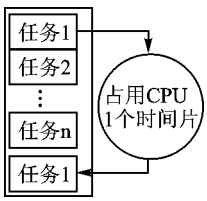

图 4 同一优先级时间片轮转调度图

在没有更高优先级任务就绪时，相同优先级的任务依照就绪的先后次序执行。执行一定的时间片后，无论任务完成与否，均转入下一任务运行。未运行完的任务释放处理器的控制权后转入就绪队列的末尾，依次往复。这样的轮转策略保证了具有相同优先级的任务平等地享有控制权的处理权。在 Windows CE 系统中，一般设置的时间片大小为 10 ms。

3 总 结

本文对几种操作系统的内核的主要部分（任务切换与调度）进行了分析比较，便于理解其实时性、可靠性等方面的优缺点，为以后进行系统的移植和开发打下基础。由于笔者时间和精力有限，而且目前的操作系统很多，本文只分析了 4 种系统，还不够完善。未来可以对其他更多的实时操作系统进行分析比较。

下转 16 页

以太网通信验证设备主要有：一台 PC 机、一根 9 针的串口线、一个带总线供电的 EPA 集线器、一个 JTAG 调试器、EPA 主控卡。EPA 主控卡与 PC 机通过 EPA 集线器连接在同一局域网内，EPA 主控卡的 IP 地址为 192.168.1.2，PC 机的 IP 地址为 192.168.1.161。

ICMP 是 Internet 控制报文协议，它是 TCP/IP 协议族的一个子协议，用于在 IP 主机、路由器之间传递控制消息。控制消息是指网络通不通、主机是否可达、路由是否可用等网络本身的消息。ICMP 是一个“错误侦测与回报机制”，其目的就是检测网路的连线状况，也能确保连线的准确性。通过 ICMP 的回显请求和回显应答报文组合可以确定 PC 机和 EPA 主控卡能否彼此通信。以太网通信过程及结果见本刊网站 www.mesnet.com.cn——编者注。

4.2 USB Host 驱动验证

USB Host 驱动验证设备主要有：1 台 PC 机、1 根 9 针的串口线、1 个 JTAG 调试器、EPA 主控卡、USB 设备（U 盘）。

为验证硬件 USB 接口和软件 USB Host 的正确性，需要选定 USB 设备，并为这个设备编写 USB 主机驱动程序。本文中选用了常用移动存储设备——U 盘。在编写好 U 盘驱动后，通过 JTAG 调试器将 USB Host 下载到主控卡中，并将主控卡串口 0 与 PC 机串口相连，开启超级终端，然后将 U 盘插入主控卡主机端口。在 U 盘插入主控卡以后，开始对 U 盘枚举。枚举成功以后，往 U 盘内写入数据，然后从 U 盘读出数据并对读写数据进行比较，所有的枚举信息、读写信息都通过超级终端打印出来。在 U 盘读写完成以后，将 U 盘插入电脑，在电脑上读出 U 盘数据，再次验证主控卡 USB 主机的功能正确性。通过超级终端打印的设备枚举信息和读写信息见本刊网站——编者注。

结 语

EPA 标准成功进入国际标准，为我国工业自动化发展带来新的际遇，产生了巨大的社会效益。本文提出针对 EPA 主控卡的系统结构，并对主控卡进行功能需求分析，然后对主控卡硬件和软件进行了总体设计以及软件功能模块划分，为 EPA 的推广与应用打下坚实基础。

参考文献

[1] 夏德海. 现场总线的现状及其应用（一）[J]. 中国仪器仪表，1998(1)：1-4.

[2] 沈德耀，金敏. 现场总线纵横谈[J]. 基础自动化，2000，7(4)：1-5.

[3] 张军. 基于 EPA 工业以太网的智能变送器研究[D]. 重庆：重庆大学，2006.

[4] 浅析：现场总线与 DCS 系统的网络集成的共存[OL]. [2010-03-25]. http://www.chuandong.com/publish/news/2010-3/132978.html.

[5] GB/T 20171-2006 用于工业测量与控制系统的 EPA 系统结构与通信规范，2006.

[6] 王平，等. 工业以太网技术[M]. 北京：科学出版社，2007.

[7] Catsoulis John. 嵌入式硬件设计[M]. 徐君明，许铁军，黄年松，等译. 北京：中国电力出版社，2006.

[8] NXP Semiconductors. LPC2478 Preliminary data sheet，2008-02.

[9] NXP Semiconductors. LPC24XX User manual，2007-10.

[10] 周立功. USB2.0 与 OTG 规范及开发指南[M]. 北京：北京航空航天大学出版社，2004.

付蔚（讲师），主要研究领域为嵌入式智能系统；何培（硕士生），主要研究领域为嵌入式监控系统。

（收稿日期：2011-03-16）

上接 12 页

参考文献

[1] 罗修波. 实时操作系统 VxWorks 的内核任务调度[J]. 电脑应用技术，2006(1).

[2] 万柳. 嵌入式实时操作系统 VxWorks 内核调度机制分析[J]. 计算机应用与软件，2004(6).

[3] 俞发新. 常用嵌入式实时操作系统比较分析[J]. 计算机应用，2006(4).

[4] 何海涛. μC/OS-II 中优先级抢占的时间片调度算法的实现[J]. 计算机系统应用，2009，18(11).

[5] Claudia Salzberg Rodriguez，Gordon Fischer，Steven Smolki. The Linux Kernel Primer[M]. 北京：机械工业出版社，2006.

[6] 吴国伟，李张，任广臣. Linux 内核分析及高级编程[M]. 北京：电子工业出版社，2007.

[7] Labrosse Jean J. 嵌入式实时操作系统 μC/OS-II[M]. 北京航空航天大学出版社，2003.

[8] 张胜. 嵌入式 Linux 多任务实时调度算法及应用研究[J]. 医疗保健器具，2008(3).

孙利锋（硕士生），研究方向为嵌入式系统；王民慧（副教授），研究方向为嵌入式系统、网络通信。

（收稿日期：2011-03-16）