基于MSP430单片机的实时多任务操作系统※

■ 思达高科配网技术公司 刘兵 张庆强

摘 要 实时多任务操作系统(RTOS)能有效提高嵌入式平台的资源利用效率，是嵌入式应用的必然趋势。本文阐述基于MSP430F149的RTOS——M430/OS。它由汇编写成、短小精干、占用系统资源少、运行稳定可靠，目前已在思达高科配网技术公司产品上得到应用。

关键词 RTOS 任务调度 MSP430

1 在MSP430上使用RTOS的意义

一般的观点认为，MSP430上使用RTOS是没有意义的这是可以理解的。因为MSP430的硬件资源有限(以MSP430F149为例，只有2KB RAM),任何商业操作系统都不可能移植到MSP430上。目前在MSP430上得到应用的RTOS，只有μC/OS-II，但使用μC/OS-II 必须有昂贵的C编译器，这严重地限制了其在MSP430上的使用。

正是基于以上情况，笔者在应用MSP430过程中，编写了一个基于MSP430F149的RTOS，暂定名为M430/OS。它占用RAM量少、代码短小，稍加改动就可适用于大多数其它MSP430单片机。

在MSP430单片机系统上使用M430/OS，对系统有以下意义：

① 实现软件设计的模块化。可将不同的功能模块编制成相应的任务，由操作系统按级别调用，不必为先执行哪个功能、后执行哪个功能而费神。

② 能更合理、有效地利用CPU有限的资源。按任务的重要程度安排任务的级别，能够保证最重要的任务得以最及时执行。

③ 大大降低系统故障率。低优先级的任务发生阻塞时，高优先级任务的执行不受影响。

2 M430/OS在MSP430F149上的实现

2.1 M430/OS功能特点

M430/OS有以下特点：

① 采用占先式内核，即高优先级的任务可以从低优先级任务“抢”回CPU控制权；

② 每个任务都单独开辟一个任务栈；

③ 每个任务占十几到几百字节的任务堆栈，任务栈的大小可以根据任务中现场数据、局部变量和嵌套调用的情况估算；

④ 每个任务各分配一个优先级,不支持两个任务有相同的优先级；

⑤ 不支持信号量、邮箱功能；

⑥ 任务状态只有三种：运行(executing)、就绪(ready)、挂起(suspended)；

⑦ 系统占用RAM量=((任务个数+1)×4)+6字节，不包括任务堆栈；

⑧ 代码量少，目前版本的代码共有86行汇编代码，256字节目标代码；

⑨ 理论上最多支持126个任务；

⑩ 任务锁定功能：在一段低优先级的代码中，不想让操作系统把CPU权切换到别的任务，这时可

结 语

本文讨论了基于DSP的管理系统设计，重点是两级分布式系统之间的协调。在1553总线中断上，如果采用精确中断，程序会很复杂，但对于实时控制系统会有很大益处。由于这套系统高层命令的实时性并不是很强，采用非精确中断就足够了。

参考文献

1 David A.Patterson John L.Hennessy. Computer Architecture: A Quantitative Approach. Ed. San Francisco: Morgan Kaufmann Publishers, 1995

2 王鼎兴，陈国良. 互联网络结构分析. 北京：科学出版社，1990

(收稿日期：2003-02-23)

以把这代码锁定，在运行这段代码时，就不会引起任务切换；

⑪ 任务唤醒功能：在一个任务中产生一个的事件来触发其它任务运行（如果被触发的任务优先级高的话，就会马上运行）。

2.2 系统函数介绍

① OS_Init：多任务初始化，进行任务栈（任务栈的结构见图1）、任务延时计数、任务状态的初始化。初始化完成后，系统直接切换到最高优先级的任务，多任务系统启动。

② OS_Time_Dly：把当前任务挂起一段指定时间让其它任务运行。

③ OS_Sched：任务调度，它先把每个任务的延时数减1，然后再找出最高优先级的就绪任务，并切换到这个就绪任务。如果无就绪任务，就切换到空闲任务。

④ OS_Free_Task：空闲任务，是一个很重要的系统任务，当所有任务都挂起时，运行此任务。它主要是对一个计数器Free_Count一直进行累加，用户可以根据这个计数器计算出CPU的利用率。

⑤ OS_Task_Lock：锁定任务调度，禁止任务调度。主要用来锁定在低优先级中的一些可重入的代码或一些重要代码。

⑥ OS_Task_Unlock：解锁任务调度，和上面的子程序功能相反。

⑦ OS_Task_Wakeup：唤醒指定优先级的任务，并产生一次任务调度，如果被唤醒任务的优先级比当前运行的任务的优先级高，任务就会切换到被唤醒的任务中，否则等待下一个调度时机。

2.3 主要功能的实现

（1）任务初始化

系统加电运行后，首先对硬件资源进行初始化，接着就要对多任务进行初始化了。主要是初始化每个任务的任务栈、每个任务的时钟滴答数和堆栈指针位置。我们把每个任务栈都初始化成图1形式。

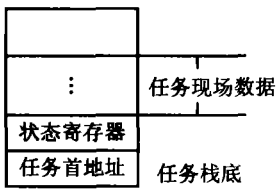

图1 初始化好的任务栈结构

任务栈的初始化如下程序（r11是用来初始任务堆栈的一个指针，r10是一个循环计数器）：

```
mov.w   #(栈底+2),r11
clr.w   Task_Tick(r10)          ;清0时钟滴答数
mov.w   #任务首地址,0(r11)      ;把任务地址压入堆栈
mov.w   SR,-2(r11)              ;把标志寄存器放入任务栈
mov.w   r11,Task_SP(r10)
sub.w   #现场所占的字节数,Task_SP(r10) ;SP位置放
                                ;入堆栈
```

初始化完任务栈之后，就把堆栈指针指向最高任务优先级任务栈的任务首地址处，再执行ret返回。这样，多任务就启动开了，程序如下：

```
mov.w   #09feh,sp   ;最高优先级的任务栈任务首地
                    ;址位置
ret                 ;返回到最高优先级的任务
```

任务初始化的流程如图2所示。

（2）时钟节拍

时钟节拍由MSP430F149的TimerA产生。TimerA工作于上升模式，CCR0中是TimerA计数最大值。TimerA初始化代码如下：

```
bis.w   #(TASSEL1+TACLR+MC_1),&TACTL
mov.w   2(sp),&CCR0 ;计数最大值，此值决定时钟节拍
bis.w   #CCIE,&CCTL0
```

（3）任务调度

应用程序调用OS_init进行初始化后，直接切换到最高优先级的任务。

每个任务在运行一个循环后执行OS_ Time_Dly挂起。这是通过把该任务的延时数填到该任务的Task_ Tick中，然后再执行任务调度程序实现。

任务调度就是在定时中断时对所有任务的Task_ Tick减1，然后再按优先级高低的顺序查找Task_Tick减到0的任务，并直接跳到任务切换程序。

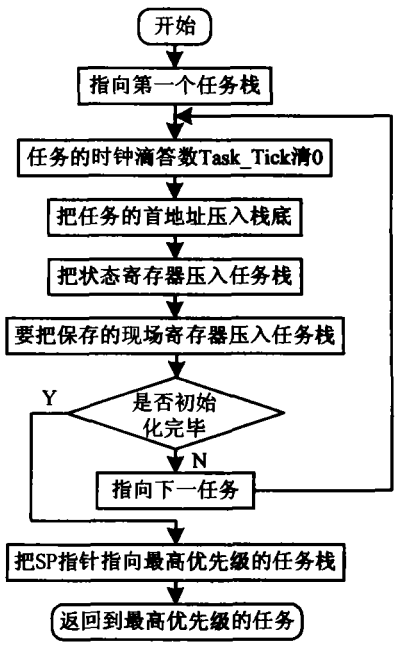

图2 任务初始化流程

下面是任务切换程序（r10的内容是就绪任务的标志，由调度程序找出）：

```
pushALL
        ;把当前任务现场入栈
```

```
mov.b  Now_Task,r11     ;当前任务标志放r11
mov.w sp,Task_SP(r11)   ;保存当前任务堆栈指针
mov.b  r10,Now_Task     ;就绪任务标志变为当前任务标志
mov.w Task_SP(r10),sp   ;就绪任务的任务栈指针放入SP
;此时再进行堆栈操作就是对就绪任务的任务栈操作了。
popALL                  ;把就绪任务的现场出栈
reti                    ;中断返回，返回到就绪任务
```

任务调度的调度时机有两种：一种是在任务挂起时，一种是定时中断。任务挂起时的任务调度一定会引起任务切换，定时中断就不一定引起任务切换了。因为，如果就绪任务是当前正在运行的任务时不会引起切换。正是如此，任务调度是RTOS中执行得最频繁的一个功能，也是最重要的一个功能，所以必须尽量缩减其代码量，尽量用可靠的调度算法来减少任务调度所占的时间。这个子程序的流程如图3所示。

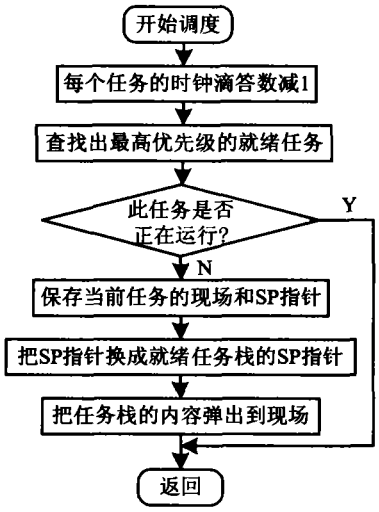

图3　任务调度流程

（4）任务锁和其它功能的实现

任务的加锁与解锁，是为了使一些在低优先级任务的不可重入代码，或对实时性要求较高的I/O操作在执行中不产生任务切换。这项功能是通过设置一个标志位实现的。当调度程序检查到任务被锁定时，就算有就绪任务也必须等开锁之后才能切换。

如果系统突然产生一个事件要某个挂起的任务来处理，可以在事件产生的程序中调用任务唤醒。它的原理是把Task_Tick清0，然后执行一次任务调度。如果这个任务优先级较高，就直接切换到这个任务里执行。

3　总　结

M430/OS已在笔者开发的基于MSP430F149的系统上应用，运行稳定可靠。该操作系统稍加改动，就可应用于其它MSP430单片机。当然，它的功能还是很有限的，也可能还存在一些尚未暴露的问题；但无论如何，它向我们证明，在MSP430单片机系统中使用RTOS是完全可能的。M430/OS源程序见本刊网站（http://www.dpj.com.cn）。

希望有兴趣的同志与我们交流。刘兵：public_rtos@163.com　张庆强：starzqq@163.com

参考文献

1　Labrosse Jean J. μC/OS-II 开放源代码的实时多任务操作系统. 邵贝贝译. 北京：中国电力出版社，2001

2　胡大可主编. MSP430系列超低功耗16位单片机原理与应用. 北京：北京航空航天大学出版社，2001

（收修改稿日期：2003-02-06）

ADI公司力推新的Blackfin® eMedia Platform

近日，美国模拟器件公司发布一款基于Blackfin® eMedia Platform新的可编程处理器——ADSP-BF532eM10。该公司称它能以低于同类处理器解决方案50%的成本优势为全新类型的消费类电子设备提供Windows Media 9 Series音频和视频质量。ADI公司运用其信号处理和多媒体信号压缩技术方面的内核设计能力推出新的Blackfin® eMedia Platform，以推动各种音频和视频设备设计从专用集成电路（ASIC）转向基于数字信号处理（DSP）的可编程芯片。这种Blackfin eMedia Platform支持Windows Media 9 Series，能够提供用户期望的高质量视频，而其文件大小仅有MPEG2的1/3和MPEG4的1/2。这种组合技术能够满足新一代数字音频和视频娱乐产品的需求，包括机顶盒、便携式娱乐设备、智能显示设备、便携式个人摄像机和家用数字多媒体设备。

ADI公司新的Blackfin eMedia Platform是基于ADI公司Blackfin处理器构建的，它能为高级多媒体设备所要求的音频和视频处理提供高性能信号处理、低功耗和最优化特性。ADI公司的Blackfin处理器兼具业界一流的DSP性能和微控制器（MCU）功能，能够满足各种占有大量市场的消费类电子设备对计算量大和功耗低的要求。

Blackfin eMedia Platform完全可编程，它将ADI公司低功耗高性能Blackfin处理器与各种软件算法和接口结合在一起，以便符合当今各种主流音频和视频编码解码标准，包括Windows Media Audio/Video 9 Series、多路媒体流（包括分组和抖动缓冲）以及数字版权管理协议。Blackfin eMedia Platform完全通过软件处理所有这些功能，其使用灵活性保证了OEM能够随应用发展而快速升级其产品或采用新的多媒体格式。