基于 MCS51 单片机的多任务操作系统内核设计

行 江[1]，沙林秀[2]
（1. 中国电子科技集团公司 第二十研究所，西安 710068；2. 西安石油大学，西安 710065）

[摘 要] 针对 MCS51 单片机资源较少、不具有实时多任务支持功能，文章研究了 MCS51 单片机的硬件特性和多任务机制的实现策略，分析了 MCS51 单片机实现多任务机制的基本条件，详细介绍了基于 MCS51 单片机的多任务操作系统内核的设计策略和实现方法，降低了实现多任务机制的开销，提高了系统的效率，该操作系统内核可大大提高产品控制系统的研发效率，缩短开发时间，进一步降低了硬件成本，拓宽了 MCS51 的应用范围。

[关键词] MCS51 单片机；多任务系统；操作系统；内核
[中图分类号] TP368.1 [文献标识码] B [文章编号] 1002－1183（2013）01－0016－03

MCS51 系列单片机以其体积小、功能全、价廉、面向控制、开发应用方便等优点，广泛应用于工业控制、智能接口、仪器仪表、通用测控等领域[1,2]。随着人们对控制系统的要求不断提高，现有的基于 MCS51 单片机的控制系统的局限性越发明显，特别是它不具有实时多任务支持功能，大大限制了其在控制系统中的进一步发展。本文针对 MCS51 单片机在多任务应用中的局限性，首先研究了 MCS51 单片机的硬件特性和多任务机制的实现策略，分析了 MCS51 单片机实现多任务机制的基本条件，然后将多任务机制引入 MCS51 单片机系统，设计了一个基于 MCS51 单片机的多任务操作系统内核。该内核不仅实现了基于 MCS51 单片机的多任务设计，同时降低了任务切换之间的时间消耗，提高了多任务机制的工作效率，而且降低了硬件成本，缩短了研发周期，使得现有的单片机应用更加广泛。

1 多任务操作系统内核设计思想

从宏观上看，几个任务“同时运行”。即多个任务先后开始了各自的运行，且均未结束。

从微观上看，几个任务“交替执行”。对于单处理机系统而言，各个任务只能轮流地占用 CPU。其实质是指多个任务在单片机中“交替执行”。

多任务要求系统在同一时间内执行多个任务[3]，如果只有一个中央处理器（Central Processing Unit，CPU），那么 CPU 并非在给定时间段内运行多个程序，而是按时间片在各个程序间快速地切换，从宏观上看，CPU 似乎在同一时间内运行了多个程序。采用 MCS51 单片机实现多任务操作系统内核的设计就是基于这种多任务的快速调度的思想。

MCS51 单片机内部只有一个执行单元，即程序只能按照一定流程一条一条地执行，一个任务完成之后（中断除外），才能执行下一个任务。因此，只有通过中断的方式才能够实现任务之间的切换。通过对 MCS51 单片机的硬件资源和多任务操作系统的分析，要实现基于 MCS51 单片机的多任务操作系统必须解决以下几个问题。

首先，考虑到通用控制系统组成，并行任务的数量并不多，主要的任务只有三个：一是处理与用户的交互，二是执行特定的控制操作，三是完成与其它外围设备的通信与控制。考虑到 MCS51 单片机的八个工作寄存器 R0～R7 能够映射到 0 区～3 区这四个区域中，只需通过设置状态字寄存器 PSW 中的 RS0、RS1 位的值即可进行简单的区域切换。因此，在基于 MCS51 单片机多任务系统内核中设定并行任务数量最大值为 4 个，它们各自占用一个工作寄存器映射区域，从而减少了任务切换时用于保存和恢复任务运行环境的开销。

其次，任务的切换采用时间片轮转（调度）方式[4]。即将时间量化为一个个固定长度的时间片，各个任务包含有不同数量的时间片段。每当一个任务运行了一个时间片段，则该任务的时间片数量减 1，直至所有任务的时间片数量都为 0 后再重新赋值。在时

[收稿日期] 2012－03－05
[作者简介] 行 江（1977－），男，陕西西安人，工程师，硕士，毕业于西安石油大学，从事仪器仪表和计量标准的研发工作。

© 1994-2013 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

间片段内，任务不会切换；在时间片段之间，按照任务的优先级对所有就绪任务进行排序，切换至优先级最高的就绪任务执行。

然后，考虑到任务的性质不同，处理的方式、优先级也不同。例如：一个用于处理与用户交互的任务，需要尽快执行以响应用户的输入和显示；而一个用于科学计算的任务就没那么紧迫，可在单片机空闲时执行。因此本内核设置了与任务数量相同的优先级(即共设有 4 个优先级)，每个优先级上最多只能有一个任务。任务切换时总是先执行优先级高的任务。

最后，考虑到 MCS51 单片机上的资源较少，因此采用信号量加全局变量的方式来实现多任务间的通信。任务的状态字中有 1 位代表信号量，通过这个信号量和约定的全局变量来实现任务间的通信。

2 多任务操作系统内核的实现

2.1 多任务操作系统内核数据结构设计

由于 MCS51 单片机上的资源较少，内存只有 128 字节，即 0x00 至 0x7F，因此需要慎重使用每一内存单元。在本内核中按功能划分的内存分配图如图 1、图 2 所示。

0x2F	位地址区	位寻址区
…		
0x20		
0x1F	3 区	4 个工作寄存器区 R0 ~ R7
…		
0x18		
0x17	2 区	
…		
0x10		
0x0F	1 区	
…		
0x08		
0x07	0 区	
…		
0x00		
——地址——	————功能————	

图 1 多任务内核低端内存功能划分图

如图 1、图 2 所示，在本内核内存分配的设计中，4 个工作寄存器区域占用了低端 32 字节(0x00 ~ 0x1F)；4 个任务的堆栈占用了高端 64 字节(0x40 ~ 0x7F)，每个任务的堆栈空间为 16 字节；中部 32 字节分为两部分，其中高 16 字节(0x30 ~ 0x3F)为系统数据区，低 16 字节(0x20 ~ 0x2F)为用户自定义数据区。在多任务内核系统数据区内分别定义有：各个任务的堆栈指针、状态、时间片以及当前任务指针等数据结构。其内存具体分配图如图 3 所示。

0x7F	任务 4 堆栈区	4 个任务堆栈区
…		
0x70		
0x6F	任务 3 堆栈区	
…		
0x60		
0x5F	任务 2 堆栈区	
…		
0x50		
0x4F	任务 1 堆栈区	
…		
0x40		
0x3F	系统数据区	系统数据区
…		
0x30		
——地址——	————功能————	

图 2 多任务内核高端内存功能划分图

0x3F	任务 4 堆栈指针	系统数据区
0x3E	任务 3 堆栈指针	
0x3D	任务 2 堆栈指针	
0x3C	任务 1 堆栈指针	
0x3B	任务 4 状态	
0x3A	任务 3 状态	
0x39	任务 2 状态	
0x38	任务 1 状态	
0x37	任务 4 时间片	
0x36	任务 3 时间片	
0x35	任务 2 时间片	
0x34	任务 1 时间片	
…	…	
0x30	当前任务指针	
——地址——	————功能————	

图 3 多任务内核系统数据区内存分配图

在本设计中，为每个任务的堆栈分配了 16 字节，而任务切换时用于保存任务运行环境(包括 PCL、PCH、PSW、ACC、B、DPTRL、DPTRH)就需要 7 个字节，因此实际分配给每个任务的堆栈空间就只有 9 个字节，在任务子程序设计时一定要注意不能超过这个限制，否则会引起系统崩溃。

2.2 多任务操作系统内核时间片设计

多任务操作系统内核时间片设计主要是通过 MCS51 内部的定时器/计数器来实现的。MCS51 单片机内部提供了两个 16 位的定时器 T0 和 T1，若系统使用的晶振频率为 12MHz，则理论上定时器最长可以定时 65535μs，最短为 1μs。参考一般嵌入式系统设计，本系统设计定时器时间片为 10ms。由于本系统中采用的晶振频率为 11.0592MHz，因此 T0 的 TH0 初始化为 0xDC，TL0 为 0x00。内核中最多可有 4 个任务并行执行，任务优先级分为 4 级，每个优先级上最多只

© 1994-2013 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

有 1 个任务。多任务内核状态字功能定义见图 4。

功能	就绪	阻塞	同步	运行	保留	保留	优先级	
位地址	7 位	6 位	5 位	4 位	3 位	2 位	1 位	0 位

图 4 多任务内核状态字功能定义图

2.3 多任务操作系统内核任务切换设计

任务切换过程中最重要的环节包括当前任务运行环境的保存、挑选出符合条件的切换任务以及恢复切换任务的运行环境。图 5 描述了一个任务 1 向任务 2 切换的大致过程。

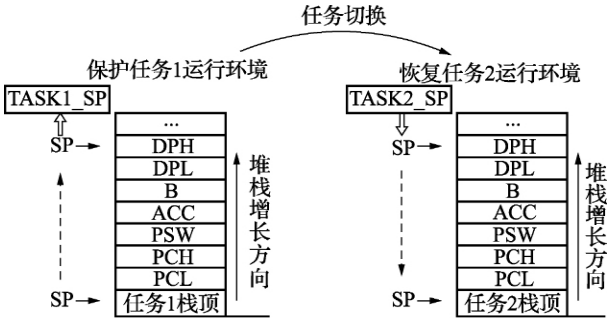

图 5 多任务内核任务切换示意图

其过程为：首先，正在运行的任务 1 被定时器中断，当符合任务切换条件时开始将运行环境，即 PSW、ACC、B、DPL、DPH 寄存器的值压入任务 1 的堆栈中，随后将 SP 寄存器的值保存在任务 1 的堆栈指针单元中；其次，根据任务状态以及优先等级挑选出将要切换的任务，这里假设挑选出来的任务为任务 2；最后，先将任务 2 的堆栈指针单元中的数值恢复到 SP 寄存器中，然后从任务 2 的堆栈中弹出 DPH、DPL、B、ACC、PSW 寄存器的值以恢复任务 2 的运行环境，接着用一个中断返回指令“reti”来继续任务 2 的运行。部分源代码如下所示：

```
；任务切换子程序
；保护当前任务运行环境
        PUSH   PSW
        PUSH   ACC
        PUSH   B
        PUSH   DPL
        PUSH   DPH
；将当前堆栈寄存器中的值保存起来
        MOV    A，#TASK1_ SP
        ADD    A，TASK_ ID_ CURRENT
        MOV    R0，A
        MOV    @ R0，SP
；寻找下一个准备就绪的最高优先级的任务
        ……
；恢复新任务的堆栈地址
        MOV    A，#TASK1_ SP
        ADD    A，TASK_ ID_ CURRENT
        MOV    R0，A
        MOV    SP，@ R0
；恢复新任务的运行环境
        POP    DPH
        POP    DPL
        POP    B
        POP    ACC
        POP    PSW
；转向新的任务
        RETI
```

由于每一个任务的 PSW 寄存器中的 RS0、RS1 位已经被预先初始化，因此通过保存与恢复 PSW 寄存器即可完成工作寄存器区的切换，从而减少了任务切换程序本身所占用的时间，提高了多任务操作系统内核的效率。

2.4 多任务操作系统内核任务间通信设计

在一个控制系统中，仅仅依靠一个任务往往不能完成一个完整的控制过程，通常都是两三个任务来共同协作实现一个功能，因此需要一种机制来协调共同参与控制的多个任务。鉴于 MCS51 单片机资源的限制，本内核只采用一种信号量的控制机制，即通过任务状态字中的“同步”控制位以及若干用户自定义全局变量来实现任务间的通信。

3 结束语

通过 Proteus 软件的模拟以及 MCS51 单片机硬件电路实验验证，该多任务操作系统内核不仅提高了控制系统的响应速度和运行效率，降低了控制系统对硬件的需求，而且扩展了控制系统的应用范围，延长了控制系统的设计寿命，具有较高的实用价值设计，顺应了当前单片机控制系统软件的发展方向。但是限于 MCS51 单片机资源的局限性，一些复杂的多任务操作系统内核的功能都未能实现，亟待在以后的学习研发中对此做进一步的研究。

[参考文献]

[1] 吴金戌，沈庆阳，郭庭吉. 8051 单片机实践与应用 [M]. 北京：清华大学出版社，2002.
[2] 刘湘涛. 单片机原理与应用 [M]. 北京：电子工业出版社，2006.
[3] 王辉堂，颜自勇，陈文芗. 一种基于 C51 的多任务机制及应用 [J]. 电子设计应用，2006 (6)：45 - 48.
[4] 陈劲松，程新民，魏忠. 时间片轮转算法在单片机程序设计中的应用 [J]. 电子技术应用，2003，29 (3)：79 - 82.

[编辑：曹微言]

© 1994-2013 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net