为 RTX51 Tiny 项目添加管程模块※

汪晓兢

（中国航天科工集团第三研究院 总体设计部，北京 100074）

摘要：RTX51 Tiny 是适配于 MCS51 系列单片机的小型多任务实时操作系统，该系统内核调度多任务并发运行时缺少对共享资源争用的管理工具。本文介绍一种在 RTX51 Tiny 项目中添加管程管理工具的方案。

关键词：RTX51-Tiny；管程；互斥锁；条件变量

中图分类号：TP316.2　　**文献标识码**：A

Adding Monitor Module to RTX51 Tiny Project※

Wang Xiaojing

(Overall Design Department, Third Research Institute of China Aerospace Science and Industry Corporation, Beijing 100074, China)

Abstract: RTX51 Tiny is a small multi-task real-time operating system(RTOS) adaptive to MCS51 MCU. The system kernel lacks the management tool to share the resources in times of synchronizing the execution of concurrent tasks. This paper introduces a proposal aiming to add a monitor and management tool to the RTX51 Tiny project.

Key words: RTX51-Tiny; monitor; mutual exclusion lock; condition variable

RTX51 Tiny 是应用于 8051 系列单片机的小型多任务实时操作系统，它完全集成在 Keil C51 编译器中，体积小，简洁高效，能够实现独立不相关的多任务并发执行，具体相关伙伴组任务之间的同步机制。被广泛应用在 8051 系列单片机的软件开发项目中。

当 Tiny 系统打开时间片（timeshare≠0）时，系统中多任务由内核按时间片轮转调度，并发运行。但这时只有独立不相关任务能够按时间片并发推进；而对于有共享资源关联的（非伙伴组之间的）相关任务，Tiny 却缺少对共享资源争用的管理工具。信号量是一种有效的、对共享资源进行保护的管理工具；管程是具有同等表达能力的另一种高级同步管理工具。

信号量实现两种管理机制：互斥和同步，两者完全由程序员负责。

管程相应实现两种管理机制：互斥锁和条件变量。理论上编译器负责互斥自动加锁部分，条件变量部分仍需程序员负责，这使程序员出错几率减少一半。管程机制的目标是为了更易于编写正确的程序，管程相对信号量更易于检错和控制。

信号量设计方法见参考文献[1]。信号量的优点是通用性，可以做成库函数，一次设计，长久应用。而管程，理论上是编译器依赖，与条件变量一起，随具体项目设计而不同。

Keil C51 没有管程支持，影响管程 C 语言应用的原因大多是一种习惯概念："管程是语言概念，C 语言并不支持它。……要使用管程，就需要一种带有管程的语言。"实际上，对 Tiny 这样的小系统，富有好奇心和探索欲的程序员可以自己代劳，替代编译器完成管程结构成分的功能。理论上进入管程时的互斥由编译器负责，而通常的做法是用一个互斥锁或二元信号量。当一个任务调用管程过程时。该过程中的前几条指令将检查在管程中是否有其他的活动任务。这是程序员替代编译器完成管程语言成分时所需做的少量工作。

管程结构如图 1 所示，管程把所有共享资源以及一组针对这些共享资源的操作过程集中在一个软件模块 Monitor 内部。并发争用管程内部共享资源的外部多任务，只能通过调用管程内部过程来间接使用管程内部共享资源。管程的互斥特性保证了外部无序并发任务只能以串行化方式顺序通过管程（临界区），以协作方式顺序使用共享资源。管程结构（见图 1）的右半部实现互斥机制；左半部实现条件变量同步机制。

下面举实际应用中的例子简单说明管程功能及应用。

例 1　无条件临界区（实现图 1 右半部互斥机制）

假定有三个 RTX51-TNY 任务，task1～task3，并发使

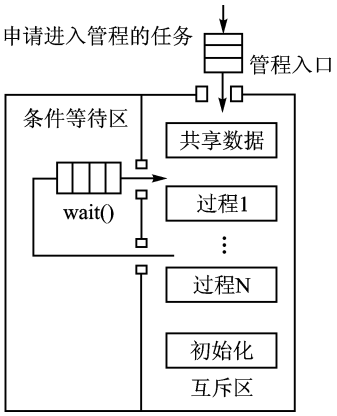

图 1　管程结构

用单一串口资源分别输出数据 s1～s3，这里串口是共享资源。

编者注：例 1 源程序略。

三任务借助回调函数方法进入管程 Monitor_f((*f)(void *),void*)，而管程函数 Monitor_f( )在入口处添加了一把互斥锁 mutex，使管程函数具有互斥特征，每次只允许一个申请使用管程内共享资源的外部活动任务进入管程。

申请进入管程的任务，必须先拿到互斥锁 mutex，获得使用权，使用完资源后，退出管程前又必须先归还锁，释放使用权。获取锁成功的任务，必须按照“以关锁操作开始，以开锁操作结束”的规则进行。

互斥锁是信号量锁的简化应用，差别在于：当取锁失败时，管程互斥锁调用 os_switch_task( ) 调度程序，主动放弃 CPU 给下一任务，并不阻塞当前任务，仍为就绪态，因此无需等待其他任务唤醒，也不像自旋锁那样忙等待；在该任务下次被调度时，它再一次对锁进行测试。

当取锁成功时，互斥锁能够自动关锁以防止其他任务进入；这避免了程序员“以关锁操作开始”的疏忽。

例 2　条件临界区

例 1 中，管程提供了一种实现互斥的简便途径；如果仅仅替代编译器的互斥加锁功能，这好像已经够用了。但管程还需要有同步机制，还需要一种方法使进入管程的任务能够像使用信号量一样实现同步机制。实现前驱或资源条件临界区，即进入管程的任务在发现继续运行条件不满足时，能够阻塞自己。这是应该由程序员主要负责的部分，编译器无能为力。

解决方法就是引人条件变量以及利用 RTX51 Tiny 已有的相关的两个操作：os_wait( )和 os_send_signal( )。当一个管程过程发现它无法继续运行时（例如前驱条件或者资源不满足时），它会在某个条件变量上执行 os_wait( )操作。该操作导致调用任务自身阻塞，并且还将另一个以前等在管程之外的任务调入管程。一个进入管程无法继续运行，被迫进入自身阻塞（等待条件满足）的任务，必须按下列规则进行：

- ◆ 释放互斥锁给管程入口；
- ◆ 进入条件等待队列；
- ◆ 睡眠。等待伙伴唤醒。

假定例 1 中三个任务要求按图 2 所示逻辑规则顺序轮流使用串口资源输出，管程内部就需要有条件变量支持。条件变量同步机制理论上是由程序员负责的部分，编译器（替代）仍只负责互斥锁部分。

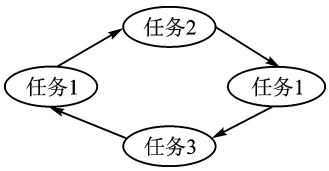

图 2　三任务轮流执行逻辑规则

编者注：例 2 源程序略。

这个例子当然有更简单的解法。测试流程如图 3 所示，图中写成完整 if 嵌套形式，是为了突出条件变量的逻辑表达式概念。退出管程前的任务测试程序，代码已经裁剪掉一半（在等待队列初始化全 0 条件下）。任务 1 和任务 2 按照逻辑表达式规则判定（predicate），从不会进入条件变量等待队列，只有 3 以上的任务才会进入条件变量队列等待。

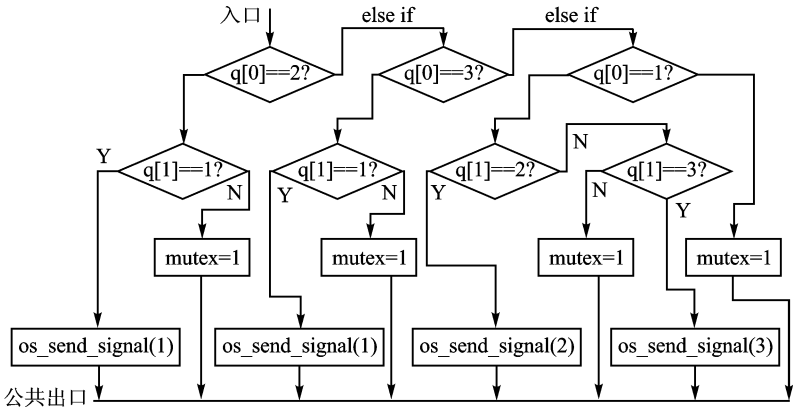

图 3　管程内活动进程退出管程前对伙伴组条件变量队列的测试

从例 1 看到，C51 带有 Test and Set Lock 语句（由 8051 硬件功能确定），这是 C51 能够简单实现管程入口互斥锁的首要条件，也即管程（模块）实现的互斥机制部分（理论上编译器负责的加锁部分）。

从例 2 看到，RTX51 Tiny 具有 wait/signal 函数，自身已能完成伙伴组间同步机制，因而能够方便实现条件变量机制。这是管程（模块）实现的同步机制部分。

例 3　经典同步/互斥问题（生产者与消费者问题）

操作系统文化中有一些经典的进程同步问题被广为讨论和分析，它们是从各种类型的同步问题中归纳、抽 27

然后将周期分成 64 等分，分别测量和存储该周期内 64 个相位的不同位移值。软件流程图如图 2～图 3 所示。

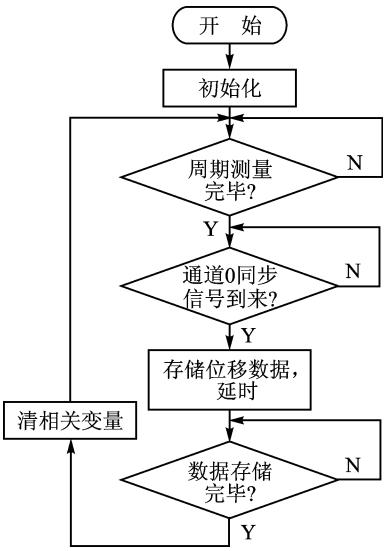

图 2 主程序流程图

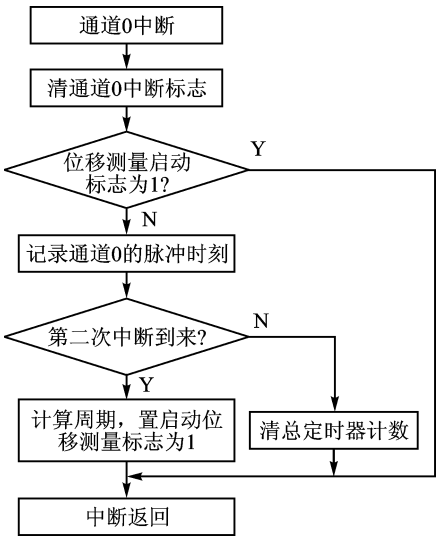

图 3 通道 0 中断流程图

编者注：源程序见本刊网站 www.mesnet.com.cn。

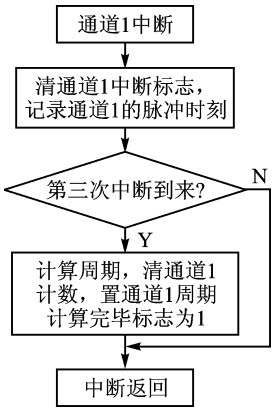

图 4 通道 1 中断流程图

结 语

其捕捉功能精度高，可靠性好，对高频信号周期测量十分方便，但对低频信号就会存在溢出问题，导致单一的捕捉功能无法对低频信号进行测量。本文通过捕捉功能和总定时器溢出中断处设置计数变量的方法，实现了对高、低频信号的同时测量。

参考文献

[1] 刘连鑫，胡琦. MCS-52 单片机的捕捉功能及应用[J]. 山东工程学院学报，2000，14(2)：76-78.

[2] 张阳，吴晔，滕勤，等. MC9S12XS 单片机原理及嵌入式系统开发[M]. 北京：电子工业出版社，2011：181-226.

[3] Freescale Semiconductor. MC9S12XS256 Reference Manual，2008.

[4] 飞思卡尔 xs128 的输入捕捉程序[EB/OL]. [2013-05]. http://wenku.baidu.com/view/ba1c17c62cc58bd63186bd7d.html.

[5] 手把手教你写 S12XS128 程序(17)—Timer 模块介绍 1[EB/OL]. [2013-05]. http://wenku.baidu.com/view/3295471a10a6f524ccbf8584.html

[6] MC9S12XS128 定时器模块及其应用实例[EB/OL]. [2013-05]. http://wenku.baidu.com/view/e410f0e85ef7ba0d4a733b8f.html.

王永龙(研究生)，研究方向为故障诊断。

(责任编辑：杨迪娜 收稿日期：2013-05-14)

24 象出来的一般性模型；每个新的同步/互斥机制的发明都希望通过这些经典问题来展示自己的精妙之处。生产者与消费者问题被普遍用来检验同步/互斥机制的正确性。

两个任务：生产者/消费者任务通过共享一个公共的固定大小的缓冲区(例 3 中为 8 字节。)而关联。生产者将信息(产品)存入缓冲区供消费；消费者从缓冲区取出信息(产品)以消费。如果生产速度大于消费，供大于求，缓冲区将很快填满，生产者必须睡眠以停止生产，等待消费者从缓冲区中取走一些数据项消费掉后再唤醒生产者。同样，如果消费速度大于生产，供不应求，缓冲区很快为空，消费者必须睡眠以等待生产者生产一些数据项后再唤醒消费者。例 3 源程序略——编者注。

编者注：本文为期刊缩略版，全文见本刊网站 www.mesnet.com.cn。

参考文献

[1] 程远，勾勇华，龚志勇，等. RTX51 Tiny 中信号量操作的实现[J]. 单片机与嵌入式系统应用，2004(10).

[2] Qingli. 嵌入式系统的实时概念[M]. 王安生，译. 北京：北京航空航天大学出版社，2004.

[3] Andrew S. Tanenbaum. 现代操作系统[M]. 3 版. 陈向群，马红兵，等译. 北京：机械工业出版社，2009.

(责任编辑：高珍 收修改稿日期：2013-06-28)