嵌入式操作系统混合任务调度技术与策略研究

陆 伟[1],张龙妹[2]
LU Wei[1], ZHANG Longmei[2]

1.西安财经学院 信息学院,西安 710100
2.西安科技大学 通信与信息工程学院,西安 710054
1.School of Information, Xi'an University of Finance and Economics, Xi'an 710100, China
2.Communication and Information Engineering College, Xi'an University of Science and Technology, Xi'an 710054, China

LU Wei, ZHANG Longmei. Mixed tasks scheduling technique and policy in embedded operating systems. Computer Engineering and Applications, 2015, 51(15):6-11.

Abstract: A mixed kernel architecture of embedded operating system which can support time triggered and event triggered tasks at the same time is designed. The mixed architecture is based on the architecture of μC/OS-II operating system and its interfaces are consistent with OSEK/VDX specification so it is portable. A static, periodic, preemptive scheduling policy for mixed tasks is proposed in the architecture. The scheduling policy supports task switching both at interrupt level and task level and the EDF(Earliest Deadline First) algorithm is used for tasks retrieving. Compared to OSEKtime OS which support task switching only at interrupt level and FIFO(First In First Out) algorithm for tasks retrieving, the proposed policy can improve resource utilization and ensure real-time tasks execution as far as possible. The results of experiment and analysis show that the mixed kernel architecture of embedded operating system is convenient for porting and the scheduling policy proposed for mixed tasks is feasible and effective.
Key words: embedded operating systems; time triggered/event triggered; tasks scheduling; Open System and the Corresponding Interfaces for Automotive Electronics/Vehicle Distributed Executive(OSEK/VDX) standard; μC/OS-II

摘 要:针对当前嵌入式系统中时间触发与事件触发混合任务的特点,以μC/OS-II操作系统架构为基础,设计了一种能够同时支持时间触发与事件触发的混合操作系统内核架构。该架构符合OSEK/VDX标准,具有良好的可移植性。针对混合任务调度问题,提出了一种静态周期性可抢占式混合任务调度策略,该策略同时支持中断级与任务级的任务切换,并采用EDF(最早截止时间优先)算法对被抢占的时间触发任务进行恢复,相比OSEKtime OS只能在中断级进行任务切换以及FIFO(先进先出)恢复算法,能够提高系统资源利用率,并最大限度保证任务实时性。实验分析结果表明,所设计的混合操作系统架构移植方便,所提出的混合任务调度策略可行有效,调度过程具有良好的可预测性。
关键词:嵌入式操作系统;时间/事件触发;任务调度;汽车电子类开发系统和对应接口标准/汽车分布式执行标准(OSEK/VDX);μC/OS-II
文献标志码:A **中图分类号:**TP31 doi:10.3778/j.issn.1002-8331.1504-0153

1 引言

传统嵌入式系统为追求响应性能,大多采用事件触发的任务调度策略,以特定的外部事件作为任务的触发源[1]。为满足特定应用需求,研究人员针对时间触发系统也展开了一定研究[2]。随着嵌入式系统应用越来越广泛,越来越多系统的任务表现出事件触发与时间约束的双重特性,尤其在航空、航天及核电等安全关键应用领域,有些任务具有严格的时间约束,一旦这些任务调度违反了时间约束,将造成严重的后果[3]。在此类既具有事件触发特征任务,又具有较强时间约束要求任务的系统中,单纯采用事件触发或时间触发的任务调度策略将增加系统设计的复杂性,导致系统设计灵活性不足,同时还会造成系统运行的可预测性下降,使得系统整体可靠性与确定性难以保证[4-5]。因此,时间触发与事件触发

基金项目:西安财经学院科研基金项目(No.14XCK01);西安科技大学培育基金(No.201356)。
作者简介:陆伟(1975—),男,博士,讲师,研究领域为系统故障诊断、软件工程;张龙妹(1977—),女,博士,讲师,研究领域为信号与信息处理。E-mail:luweinpu@nwpu.edu.cn
收稿日期:2015-04-16 **修回日期:**2015-05-31 **文章编号:**1002-8331(2015)15-0006-06

的混合系统越来越受到研究人员的关注[6-7]。

为适应当前嵌入式系统中事件触发与时间触发混合任务的特点，现代嵌入式操作系统必须为上层应用提供可靠与确定的运行环境。目前，大多数嵌入式操作系统并不支持事件/时间触发混合任务的调度，如常见的嵌入式操作系统PowerPAC，μC/OS-II等，仅支持事件触发任务调度[8]。本文研究分析OSEK/VDX标准与μC/OS-II操作系统架构，给出一种能够支持时间/事件触发任务的混合操作系统架构，并进一步提出一种静态周期性可抢占式混合任务调度策略，最后，通过实验证实所提出的混合任务调度策略具有可行性与有效性。

2　时间/事件触发操作系统架构

2.1　OSEK/VDX标准

为增强汽车工业中应用软件的可移植性和不同厂商控制模块间的可兼容性，德国推出了OSEK（Open System and the Corresponding Interfaces for Automotive Electronics，汽车电子类开发系统和对应接口标准），后来法国推出的VDX标准（Vehicle Distributed Executive，汽车分布式执行标准）并入其中，合称为OSEK/VDX标准（http://www.osek-vdx.org/）。

OSEK/VDX标准包含四个独立模块，实时操作系统标准（OSEK Operating System，OSEK OS）是其中之一，而OSEKtime OS是OSEK/VDX三个附加标准之一，主要描述时间触发操作系统的标准。

OSEK OS标准为应用程序提供标准的系统服务接口，这些接口适用于操作系统在多种类型处理器上的实现，具有高度模块化和可灵活配置的特点。OSEKtime OS是基于静态周期性调度的时间触发操作系统标准，可以与OSEK OS共存[9]。在两者共存的混合操作系统中，OSEKtime OS中的时间触发任务和中断具有更高的优先级。研究人员基于OSEK OS与OSEKtime OS的共存支持，针对混合任务调度操作系统设计展开了相关研究，并提出了一些任务调度的改进方案[10-12]。但这些研究没能给出改进后操作系统的完整架构，也没能针对混合任务调度给出具体实现方案。

2.2　混合任务操作系统架构

μC/OS-II（http://micrium.com/rtos/ucosii/overview/）是一款被广泛应用的开源、易移植、可裁剪的实时嵌入式操作系统，提供了实时系统应用所需的基本功能，具有良好的稳定性与可靠性。基于μC/OS-II操作系统架构，按照OSEK/VDX标准要求，本文给出一种支持时间/事件触发的混合操作系统架构，如图1所示。混合操作系统架构主要包含三个层次结构，分别为应用层、操作系统服务层和硬件接口层。

（1）应用层位于整个系统架构的最上层，包括操作系统配置模块，系统服务API和用户应用程序。

通过系统配置模块，用户可以对混合操作系统所支持任务的最大数量、任务类型、任务同步与通信机制，系统支持的应用模式数等进行静态配置，使得系统能够根据用户应用环境和需求的不同，灵活配置系统资源。系统服务API是用户应用程序获取操作系统服务的接口，

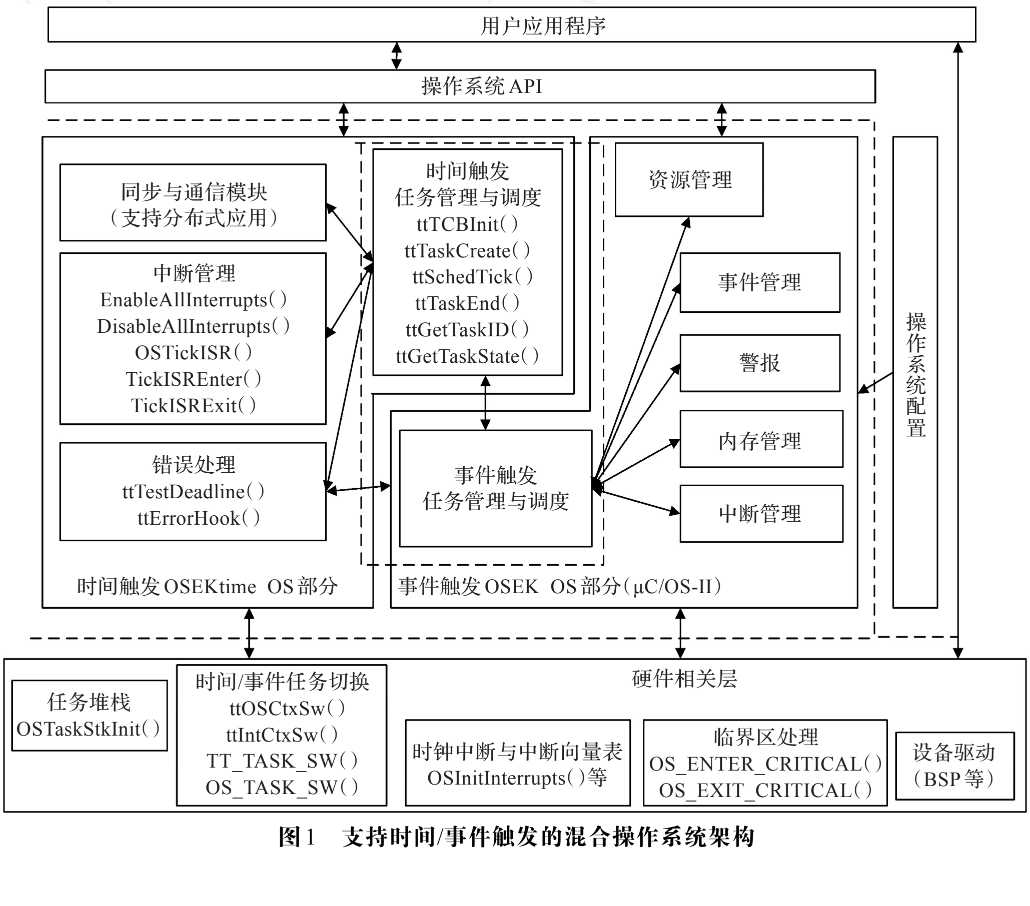

图1　支持时间/事件触发的混合操作系统架构

满足OSEK/VDX标准，以支持应用程序的移植。

（2）操作系统服务层位于系统架构的中间层，也是操作系统架构的核心。为支持不同类型任务，操作系统服务层可分为事件触发OSEK OS与时间触发OSEKtime OS两个部分。

OSEK OS部分是基于μC/OS-II操作系统，并按照OSEK OS标准对内核进行部分重新设计，以支持OSEK标准描述的事件机制，提供事件的创建、设置、清除等系统服务接口。其中，中断管理模块用于管理OSEK OS类型的中断，该模块的中断具有用户定义的静态优先级，其处理优先级低于OSEKtime OS部分的时间触发任务和中断的执行优先级。

OSEKtime OS部分在μC/OS-II操作系统内核基础上，增加时间触发任务管理与调度、错误处理、中断管理等模块。其中，时间触发任务管理与调度模块是该部分的核心，用于管理时间触发任务，提供时间触发任务的创建、初始化及任务状态查询等服务，并进行时间触发任务调度。

（3）硬件接口层实现与具体的硬件环境相关，主要包括任务堆栈，任务切换，时钟中断，临界区处理，以及设备驱动等。

3　混合任务调度策略设计与实现

3.1　混合任务调度策略设计

OSEKtime OS标准中的系统任务采用静态可抢占式调度策略。时间触发任务创建时，系统会根据其开始时间和所属应用模式生成静态调度表。系统启动时应用模式作为参数，根据相应的调度表，进行多任务调度。时间触发任务没有静态优先级，但其执行优先级比事件触发任务高；时间触发任务能够相互抢占或由一些中断打断。这种调度策略的局限性在于：（1）该调度策略不支持事件触发任务调度。（2）任务切换只在时钟中断中进行。若任务在下一时钟中断来临前已经执行完毕，CPU就会处于空闲状态，从而造成一定的系统资源浪费。（3）被抢占的任务采用FIFO策略恢复。对于时间触发任务，若后入栈任务时限早于先入栈任务，可能导致后入栈任务恢复超时。

针对时间/事件触发混合任务情况下的调度问题，本文对OSEKtime OS标准中的任务调度策略进行改进，提出一种静态周期性可抢占式混合任务调度策略，描述如下：

（1）在时间触发任务空闲时可进行事件触发任务的调度；

（2）系统在中断级和任务级均可以进行任务切换；

（3）被抢占的时间触发任务采用EDF（最早截止时间优先）算法恢复，以最大限度保证时间触发任务执行，并提高系统资源利用率。

假设系统中有三个时间触发任务TT1、TT2、TT3与若干事件触发任务，在OSEKtime OS标准任务调度策略下，可能的调度过程如图2所示。

在图2中，任务TT1在0时刻激活后开始执行，1时刻TT2任务激活后，立即抢占TT1执行；任务TT2执行结束后系统进入空闲状态，直到2时刻时钟中断来临时切换任务，恢复任务TT1继续执行；任务TT1执行结束后系统进入空闲状态，直到3时刻时钟中断来临切换到空闲任务；当4时刻时钟中断来临时切换至任务TT3执行。由于OSEKtime OS标准并不支持事件触发任务调度，因此，所有事件触发任务无法直接调度，必须转换为时间触发任务，在时钟中断时刻进行切换调度。

若采用静态周期性可抢占式混合任务调度策略，则可能的调度过程如图3所示。图3的调度过程与图2相比，区别在于：（1）由于静态周期性可抢占式混合任务调度策略同时支持任务级和中断级任务切换，任务TT2执行结束后可直接切换回任务TT1，而不必等到2时刻的时钟中断到来。（2）任务TT1执行结束后，由于时间触发任务空闲，系统可以直接切换到事件触发任务调度，同样不必等到3时刻时钟中断的到来，直至时间触发任务TT3到来，系统再次切换到时间触发任务执行。

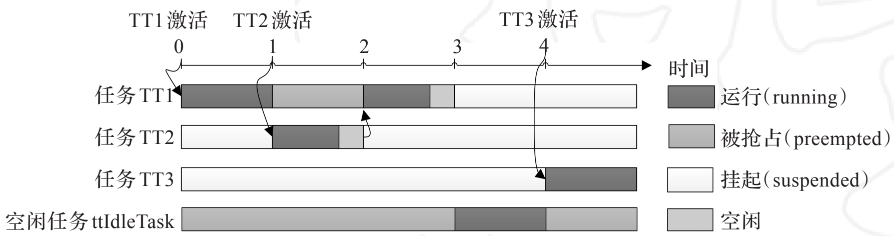

图2　OSEKtime OS标准任务调度示意图

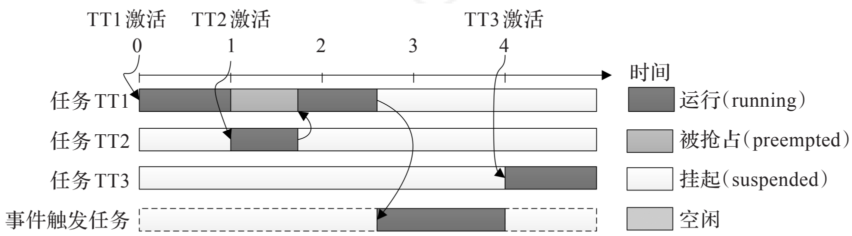

图3　静态周期性可抢占式混合任务调度策略下的任务调度示意图

通过图2与图3调度过程的比较可以发现，静态周期性可抢占式混合任务调度策略能够直接支持时间/事件混合任务调度，从而为用户提供灵活丰富的任务设计接口；静态周期性可抢占式混合任务调度策略能够同时支持任务级和中断级的任务切换，从而能够更加充分利用系统资源，提高系统运行效率。

3.2　混合任务调度策略实现

3.2.1　相关数据结构

（1）时间触发任务控制块

时间触发任务由时间触发任务控制块OS_TT_TCB结构体管理，其定义如下：

```
typedef struct os_tt_tcb{
  OS_STK*   OSTT_TCBStkPtr;//任务堆栈指针
  struct os_tt_tcb*   OSTT_TCBNext;
  ttTaskType   OSTT_TCBTaskID;//任务标识符
  ttAppModeType   OSTT_TCBAppMode;//任务模式
  INT8U   OSTT_TCBStartTime;//任务开始时间
  INT8U   OSTT_TCBWcet;//任务最坏执行时间
  INT8U   OSTT_TCBDeadline;//任务截止时间
  ttTaskStateType   OSTT_TCBState;//任务状态
}OS_TT_TCB;
```

时间触发任务状态参数OSTT_TCBState的可取值为TT_RUNNING（运行态），TT_SUSPENDED（挂起态）和TT_PREEMPTED（被抢占态）。

（2）任务恢复链表

任务恢复链表用于保存被抢占时间触发任务的任务控制块指针，当任务恢复完成后，将从链表中删除，其定义如下：

```
typedef struct os_preempted_tt_tcb{
  OS_TT_TCB*   OSTT_PTcb;//被抢占任务
  struct os_preempted_tt_tcb*   OSTT_PNext;
}OS_TT_PREEMPTED_TCB;
```

3.2.2　任务调度

（1）时间触发任务调度

时间触发任务调度涉及中断级和任务级两种情形，调度过程中任务状态转换如图4所示。

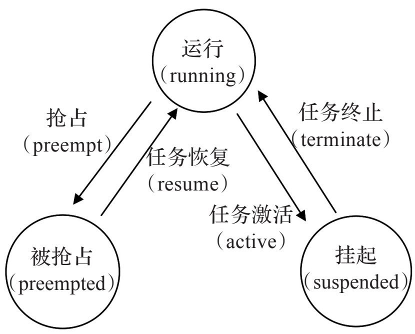

图4　时间触发任务状态转换图

中断级时间触发任务调度在时钟中断服务函数OSTickISR()中进行，调度主函数为ttSchedTick()。ttSchedTick()函数首先更新时钟节拍，然后判断当前是否需要启动新一轮时间触发任务，若需要则进行时间/事件触发任务切换的逻辑处理；任务切换在时钟中断退出函数TickISRExit()中进行。

启动新一轮任务调度的情形有两种：一是检测到有时间同步帧时，需要进行重新调度；二是当前调度周期结束，需要进行下一轮调度。任务调度逻辑如图5所示。时钟中断退出函数TickISRExit()用于完成具体的时间/事件触发任务切换，其函数流程如图6所示。

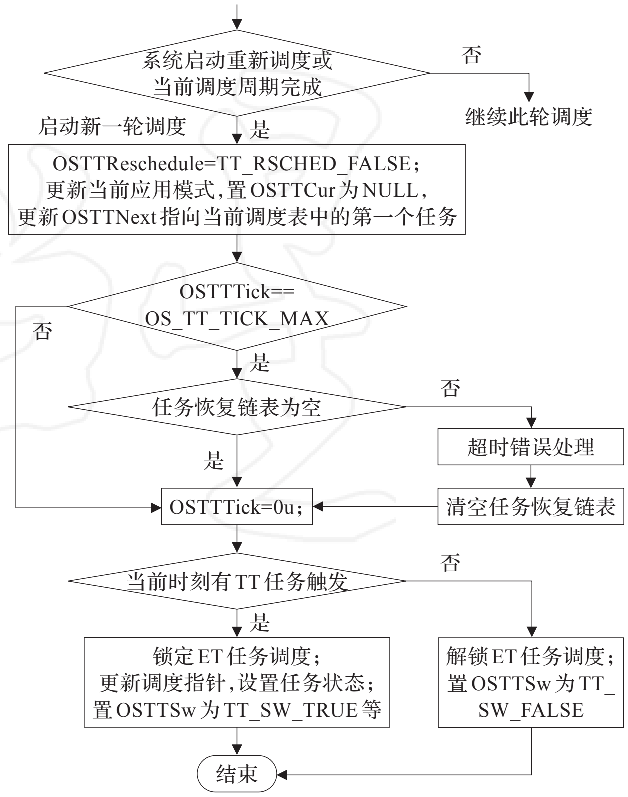

图5　ttSchedTick()函数启动新一轮任务调度逻辑图

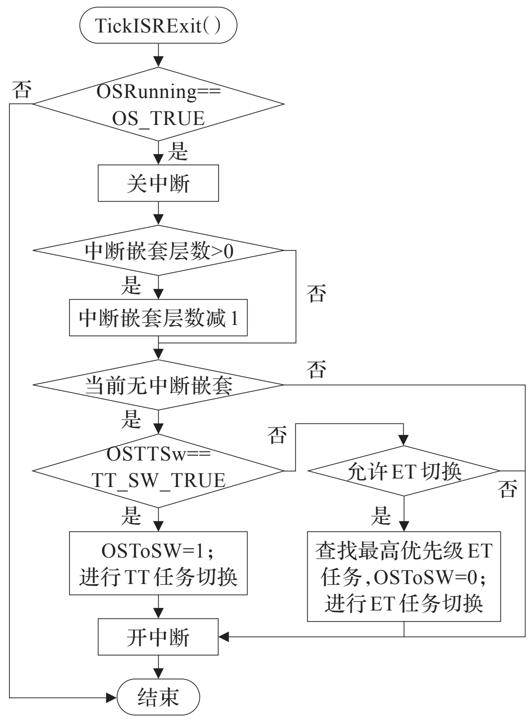

图6　时钟中断退出函数流程图

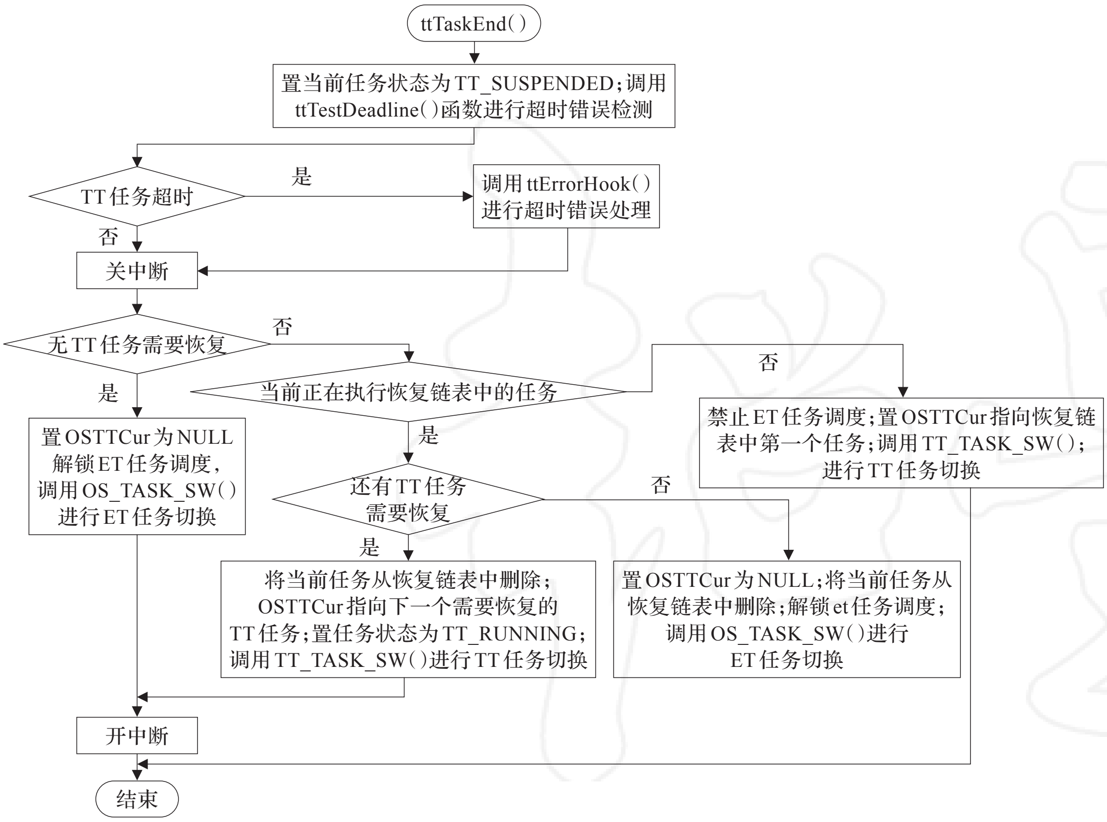

图7　ttTaskEnd()函数流程图

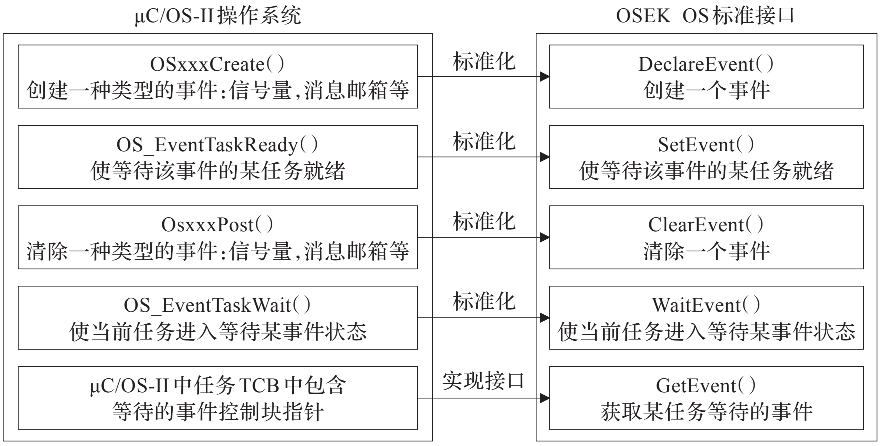

图8　μC/OS-II事件服务与OSEK标准接口对应关系

任务级时间触发任务调度的主函数为ttTaskEnd()。ttTaskEnd()函数在每个时间触发任务执行结束时调用，主要进行任务超时错误检测并进行错误处理，判断当前是否有需要恢复的时间触发任务，完成时间/事件任务的切换。函数流程图如图7所示。

（2）事件触发任务调度

事件触发任务调度主要基于μC/OS-II操作系统实现，μC/OS-II本身为事件触发操作系统，其事件管理模块包含信号量，消息邮箱，消息队列等多种事件类型，并提供了创建、设置、清除事件等系统服务接口，只需经过部分修改并实现接口的标准化，就能实现满足OSEK OS标准的事件机制。图8给出了μC/OS-II中事件模块的服务接口与OSEK标准接口的对应关系。

4　实验结果及分析

为测试所设计与实现的混合任务调度策略能否正常地进行任务相互切换与抢占恢复，在Virtex-5 FXT FPGA ML507（http://china.xilinx.com/support/index.html/co ntent/xilinx/zh/supportNav/silicon_devices/fpga/virtex5.html）评估平台上对所设计的操作系统内核架构进行了移植，并对任务调度进行了实验分析。ML507评估平台是Xilinx公司推出的一款通用的FPGA、RocketIO™ GTX和面向嵌入式应用的多功能系统开发板，搭载PowerPC440嵌入式处理器。

实验选取4个事件触发任务（ET任务）和3个时间触发任务（TT任务），任务参数如表1和表2所示。其中，ET任务的优先级数字越小表示其优先级越高；实验

环境下,系统的时钟节拍为1 ms,TT任务调度周期为50 ms。

表1 事件触发任务参数列表

任务名称	优先级	任务延时时间/ms
etTask1	10	5
etTask2	6	10
etTask3	3	20
etIdle	63	0

表2 时间触发任务参数列表 ms

任务名称	开始时间	最坏执行时间	时限
ttTask1	10	10	4
ttTask2	12	5	0
ttTask3	30	5	5

完成多次实验后,选取任务在第一个周期内的执行结果,如图9所示。

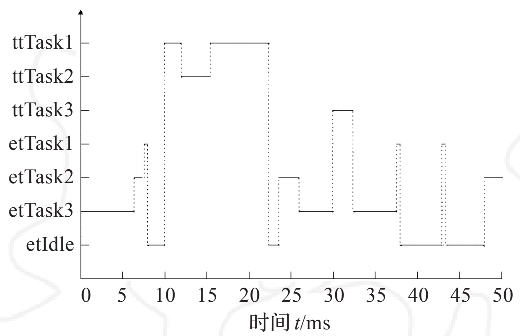

图9 混合任务调度过程

图9所示的混合任务调度过程显示,*tick*=0时刻,ET任务etTask1,etTask2,etTask3,etIdle在任务创建后就绪,由于优先级etTask3>etTask2>etTask1>etIdle,所以任务etTask3优先执行,此后etTask2,etTask1和etIdle依次执行。当*tick*=10时,ttTask1开始执行,直到*tick*=12时,ttTask1还未执行结束,这时ttTask2触发并抢占ttTask1任务执行。当*tick*=15时,ttTask2执行结束,ttTask1恢复执行;当*tick*=22时,ttTask1执行结束。之后,ET任务etIdle,etTask2,etTask3依次就绪并执行。当*tick*=30时,ttTask3抢占etTask3开始执行;在*tick*=32时ttTask3执行结束,之后etTask3恢复执行,并在*tick*=37时执行结束。此后,ET任务etIdle,etTask1,etTask2就绪并执行。

实验结果表明,时间/事件触发OSEK操作系统能够进行正常的时间/事件触发混合任务切换,任务间抢占与恢复正常。

5 总结

本文针对嵌入式操作系统时间/事件触发混合任务调度问题展开研究,参照μC/OS-II操作系统架构,设计了一种支持时间/事件触发混合任务调度且满足OSEK标准的操作系统架构,在此基础上,提出了一种静态周期性可抢占式混合任务调度策略,该策略同时支持任务级和中断级的任务切换,从而能够更加充分利用系统资源,提高系统运行效率。此外,调度策略采用EDF算法恢复被抢占的时间触发任务,能够最大限度保证任务实时性。实验结果表明,所提出的混合任务调度策略可行有效。本文仅仅针对所提出混合任务调度策略的有效性进行了分析,调度策略的任务切换开销以及大规模任务下的调度效率分析有待于进一步研究。

参考文献:

[1] Kopet H.Should responsive systems be event-triggered or time-triggered?[J].IEICE Transactions on Information & System,1993,E76-D(11):1325-1332.

[2] 苏罗辉,牛萌,刘坤.时间触发系统体系结构研究[J].计算机工程与设计,2014,35(6):1956-1961.

[3] 淡图南,朱立平,颜纪迅.一种基于时间触发的安全关键操作系统混合调度策略[C]//2013首届中国航空科学技术大会论文集,2013.

[4] Prashanth K V,Akram P S,Reddy T A.Real-time issues in embedded system design[C]//2015 International Conference on Signal Processing and Communication Engineering Systems(SPACES),2015:167-171.

[5] Borgers D P,Heemels W P M H.Event-separation properties of event-triggered control systems[J].IEEE Transactions on Automatic Control,2014,59(10):2644-2656.

[6] Schorr S,Fohler G.Integrated time-and event-triggered scheduling—an overhead analysis on the ARM architecture[C]//2013 IEEE 19th International Conference on Embedded and Real-Time Computing Systems and Applications(RTCSA),2013:165-174.

[7] Yang W J,Guo X S.Time triggered and event triggered codesign in CAN bus network[C]//3rd International Conference on Machinery Electronics and Control Engineering,2013:1036-1039.

[8] 刘森,王田苗,魏洪兴,等.基于uCOS-II的嵌入式数控系统实时性分析[J].计算机工程,2006,32(22):222-224.

[9] 谢昊飞,陈家佳,舒强.基于OSEKTime的共存模型的改进与实现[J].重庆邮电大学学报:自然科学版,2009,21(5):658-661.

[10] 章亮飞,李银国.嵌入式实时操作系统AutoOSEK的设计[J].计算机工程,2007,33(16):53-55.

[11] 毛成勇,高慧敏.基于OSEK的任务调度算法改进及实现[J].计算机工程,2010,36(4):233-235.

[12] 陈曦,吕伟杰,刘鲁源.事件/时间触发嵌入式操作系统内核的设计[J].计算机工程与应用,2009,45(16):87-89.