文章编号:1000－8829(2003)12－0045－03

一种嵌入式实时操作系统——OSEK/VDX OS

An Embedded Real Time Operating System —— OSEK/VDX OS

(同济大学 电子与信息工程学院,上海 200092) 袁铭蔚
(同济大学 汽车电子研究所,上海 200092) 孙泽昌,陈觉晓

摘要:OSEK/VDX 操作系统是一种嵌入式实时操作系统(RTOS),具有可裁剪性。本文着重描述了操作系统的内核,包括任务管理机制、一致类、事件机制、调度机制、中断管理、资源管理机制和报警器机制。

关键词:嵌入式实时操作系统(RTOS);OSEK/VDX 操作系统;任务管理;一致类;调度机制
中图分类号:TP3
文献标识码:B

Abstract:OSEK/VDX operating system is a kind of RTOS, which has the character of tailor. The kernel of OS, including task management, conformance class, event mechanism, scheduling policy, interrupt processing, resource management and alarms, is described.

Key words: embedded real time operating system (RTOS); OSEK/VDX OS; task management; conformance class; scheduling policy

当今嵌入式系统的发展成为一大潮流,嵌入式产品设计师积极地把嵌入式设计和 PC 机积累的技术普遍用于专用的、非 PC 设备中,如电话、汽车、家电、环境监控和保安系统等。这些技术包括:行业性编程接口 API 规范、无线网络操作系统、IP 构件库和嵌入式 JAVA 等。许多地区和国家的若干行业协会纷纷制定嵌入式产品标准,特别是软件编程接口 API 规范。汽车行业在这方面的发展引起人们广泛的注意,汽车工业中使用嵌入式处理器,提高了车辆的安全性和舒适性。

欧共体汽车产业联盟规定以 OSEK/VDX 标准作为开发汽车嵌入式系统的公用平台和应用编程接口(OSEK 的名称来自于德文其英文意义为"open systems and the corresponding interfaces for automotive electronics",而法语缩写 VDX 代表着"vehicle distributed executive")。OSEK/VDX 本身不是一个操作系统,而是一种规范,它规定了汽车控制器的通信规程(COM)、网络管理规则(NM)以及与 OSEK/VDX 规范一致的操作系统(OS)。

OSEK/VDX 为实时操作系统(RTOS)的开发提供了一个标准的事件触发环境。作为一种实时操作系统,应像一种高级语言,能使开发工作更容易进行,甚至提前完成,但要选择正好适用于嵌入式产品的 RTOS,也不是件容易的事情。在安装 RTOS 前,应先了解一下有关诸如调度算法、信号传送特性、存储器要求、等待时间、工具的支持以及价格模型等方面。为了推广 OSEK/VDX 规范的应用,有必要了解它。OSEK/VDX 的出现是为了解决商用的 RTOS 太大,功能过多不适用于容量大而存储器少的车辆控制系统的矛盾。

OSEK/VDX 是一个为车辆控制处理器设计的通过事件驱动的实时操作系统,定义了任务、事件、中断、消息和资源 5 个对象,内容包括任务管理、同步管理、中断管理、报警管理和错误处理管理。内核是操作系统的核心,提供了像任务调度和任务间的通信等机制。OSEK/VDX 的内核是十分小的,且是可根据应用裁剪的。值得注意的是其操作系统的内核不仅适用于汽车工业,还适用于其他一些需要实时操作系统的场合,但 OSEK/VDX 规范的其他部分如通信和网络管理在汽车应用外具有较少的应用。

1 任务管理——任务的分类及调度机制

1.1 任务的分类

任务可以分为基本任务和扩展任务,基本任务和扩展任务的根本区别是任务的可能具有的状态数。基本任务具有 3 种状态:运行状态、就绪状态、挂起状态;而扩展任务还有一个等待状态。

基本任务比起通常提的实时操作系统的任务更基本,换句话说基本任务的功能比通常所知的实时操作

收稿日期:2003－04－22
基金项目:国家 863 资助项目(2001AA501994)
作者简介:袁铭蔚(1979－),女,山东省荣成市人,博士研究生,主要研究方向为机器人控制与智能控制、网络控制;孙泽昌,男,教授,博士生导师;陈觉晓,女,博士研究生。

(C)1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

系统的任务的功能少。扩展任务更类似于平时所提的一些实时操作系统任务。扩展任务比基本任务多了个等待状态，其在管理上也比基本任务复杂，占用的资源更多。此外基本任务只在开始和结束时才有同步点，故基本任务对环境要求较低，而扩展任务运行时可能进入等待状态，因此不仅在开始和结束有同步点，而且运行过程中可能有多个同步点，故扩展任务对环境要求较高。值得一提的是即使 OSEK/VDX 具有了扩展任务，由于是静态配置所以它仍然具有小而简的特点，这也是它比其他实时操作系统显出优势的地方。

图 1、图 2 所示的是扩展任务与基本任务的状态转化图。

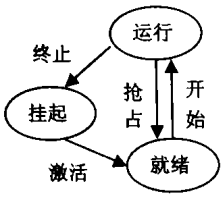

图 1　基本任务

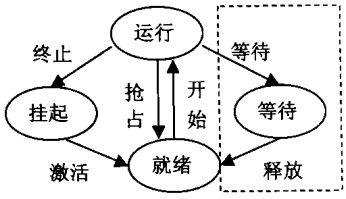

图 2　扩展任务

1.2　任务的调度机制

操作系统任务之间的同步通过调度程序来实现，调度程序决定执行什么任务，确定 OSEK/VDX 操作系统的内部活动。OSEK/VDX 支持 3 种调度方式：抢占调度、非抢占调度、混合调度。

非抢占调度机制（如图 3 所示），只有在重调度点才能从运行中的任务转去执行其他任务。因此，一个优先级较低的任务中不可抢占的部分就可能将延迟优先级较高任务的执行开始时间。非抢占调度机制中，优先级较低的任务将优先级较高的任务延迟到下一个重调度点才能执行。重调度在下列情况下发生：任务执行结束、任务执行结束并且激活了后续任务、调用调度程序和扩展任务由运行状态转化到了等待状态。

如果执行的是一个（充分）抢占调度的任务（如图 4 所示），那么在满足操作系统预定触发条件时就重新调度处于运行的任务。优先级较高的任务一就绪就会将优先级较低的任务抢占掉并且将处于运行状态的任务转到就绪状态，这时必须保护任务运行的现场。使用充分抢占调度方式时优先级较高任务的延迟与低优先级任务执行无关。

如果在同一系统中既存在充分抢占调度任务又存在非抢占调度任务，那么该系统的调度方式就属于混合调度方式。在该调度方式中调度方式取决于运行任务的抢占性，如果运行的任务采用的是非抢占调度方式，那么执行的是非抢占调度方式，否则，执行的是抢占调度方式。如果在应用场合中存在很少几个任务执行时间较长而较多的任务执行较短，那么执行时间较长的任务采用充分抢占调度方式而执行时间较短的任务采用非抢占方式效率较高。软件开发商通过配置任务的优先级和抢占性来决定任务的执行顺序。

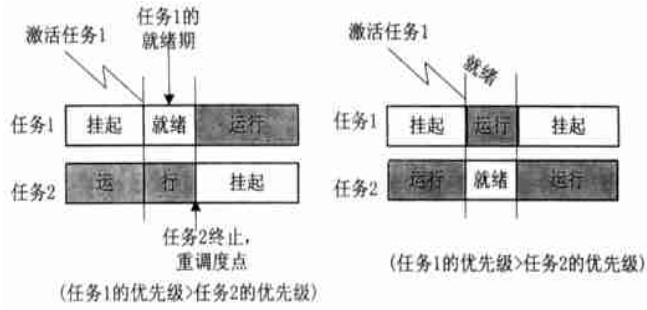

图 3　非抢占调度　　图 4　抢占调度

2　一致类

为了更容易地理解及配置操作系统，对操作系统特性进行简单分组，从而提出一致类的概念。一致类的概念使开发者可根据应用灵活地配置操作系统调度程序。操作系统定义了 4 种一致类：BCC1、BCC2、ECC1、ECC2。一致类是根据每个优先级可能具有的任务个数，需要的是基本任务还是扩展任务等来划分。一致类的确定：每个优先级上只有一个任务，且任务只能被激活一次，若该任务是基本任务则定义这种一致类为 BCC1，若该任务为扩展任务则定义为 ECC1；每个优先级上可以有多个任务且只有基本任务可以被激活多次，扩展任务只能被激活一次，若这些任务为基本任务则定义这种一致类为 BCC2，若这些任务为扩展任务则定义为 ECC2。系统的调度方式既不取决于任务的类型也不取决于一致类，因此对于每种一致类，任务的调度都可以采用抢占、非抢占、混合调度方式。例如对于 BCC1 可以采用抢占调度、非抢占调度、混合调度 3 种调度机制，BCC2、ECC1、ECC2 同理。由此可知操作系统提供了 12 种不同的任务调度和管理机制，用户可根据具体需要进行静态配置。

3　中断处理

在一些嵌入式系统，中断是外部事件的关键接口，中断源可能是实时时钟、能发送脉冲流的传感器或用户自己定义的中断源。OSEK/VDX 的中断服务程序（ISR，interrupt service routine）分为 ISR1 和 ISR2 两种。ISR1：此类中断服务程序不使用操作系统的资源，ISR 结束后，处理程序将从产生中断的地方继续执行。这类中断对任务的管理没有影响，它不要求调用操作系统的 API，因此占用的资源少，处理速度快，花费也较小。ISR2：这类中断服务程序是在系统生成时，操作系统通过用户子程序配置成的，它可以调用操

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

作系统的 API 函数。中断调度与硬件相关，不像任务是通过调度程序来管理的。中断的优先级高于任务，因此可以抢占任何任务。在执行中断子程序的时候如果激活了一项任务，那么要等到所有中断服务结束之后该任务才能开始执行。

4 事件机制

事件机制保证不同扩展任务之间的同步并且使任务的状态从等待状态转化到就绪状态或者从运行状态转化到等待状态。只有扩展任务才具有事件。一个扩展任务只有等到一个事件触发才有可能进入等待状态，而且一个扩展任务只有它等待的事件发生后才能从等待状态转化到就绪状态。

由于事件是扩展任务进入或者跳出等待状态的标志，所以在操作系统中提供设置、清除、查询事件的服务程序，可以通过任何任务、中断或者钩子(hook)程序调用设置事件函数，在任何情况下事件的接收者是扩展任务并且只有该扩展任务才可以清除该事件。

5 资源管理

具有不同优先级的多任务访问共享资源需要使用资源管理机制进行协调，这些资源包括如调度程序(在 OSEK OS 中调度程序也被看作一种资源)、存储器或者硬件。任务通过资源采用最高限度优先级(priority ceiling)协议避免了优先级倒置。

在系统生成时给每个资源分配一个优先级，资源的优先级必须高于所有要访问该资源的任务和中断的优先级，但是低于不访问该资源的任务的优先级。如果一个任务要求访问资源并且其优先级比资源优先级低，那么就将该任务的优先级提升到资源的优先级，当该任务释放了资源后，该任务的优先级回到要求访问资源前的优先级。

资源优先级可能会引起低于或者等于资源优先级的任务的执行产生延迟。但是该延迟仅仅是任务占用资源的时间。资源优先级可以避免由于多个任务访问同一资源而导致优先级倒置和死锁。

给每个资源分配一个优先级能保证：一资源已被某个任务使用，访问同一资源的任务即使其优先级比正在运行的任务的优先级高也不能进入运行状态。只有占有资源的任务释放资源后想要访问该资源的任务中优先级较高的任务才能够进入运行状态。对于抢占任务这就是一个重调度点。图 5 所示是一个采用资源最高优先级从而避免了优先级倒置的例子。任务 T0 优先级最高，任务 T4 优先级较低。在该例子中资源的优先级高于任务 T1 低于任务 T0。任务 T1 和 T4 都想访问同一资源。

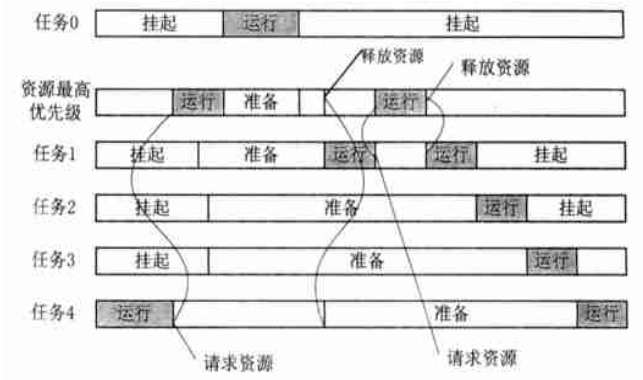

图 5 抢占性任务间使用资源最高优先级分配资源

6 报警器管理

报警器是 OSEK/VDX 为处理循环事件提供的服务机制，某种程度上可以把它看成是其他操作系统中的计数器。

当报警器的计数值等于设置的数值时，OSEK/VDX 就调用服务程序来设置事件或者激活任务。计数器可以设置单个报警器或者周期性报警器。报警器计数值可以是时间中断的个数也可以是角度位置等。OSEK/VDX 可以通过调用 API 来取消报警器并得到报警器目前的状态，可以将几个报警器以同一个计数器为基准，报警器都是静态配置的。

7 结束语

汽车工业 API 标准的出现是在开发时间和降低成本、提高软件质量和模块的可移植性上进了一大步。OSEK/VDX OS 只需最小的硬件资源(ROM、RAM、CPU 的运行时间)，甚至可在 8 位处理器上运行。不同的一致类、不同的调度模式和静态配置的特征保证了 OSEK/VDX OS 能广泛地运用在不同的场合，适用于不同的硬件。而 OSEK/VDX 操作系统内核又不局限于汽车行业，充分说明了其强大的生命力。现在有很多提供符合 OSEK/VDX 的操作系统的供应商，如 Wind River、Motorola、ETAS、Sysgo、Accelerated Technologies 和 Realogy 都在积极地推动 OSEK/VDX 的发展。

参考文献：

[1] OSEK OS/ISO WG. OSEK/VDX operating system, version 2.2[Z]. 2001-09.

[2] Webb W. 实时操作软件主宰 PC 后产品[J]. EDN China 电子设计技术, 2000, (6).

[3] Stepner D, Rajan N, Hui D. Embedded application design using a real-time OS[A]. Proceedings of the 36th ACM/IEEE conference on Design automation conference[C]. 1999: 151-156.

[4] Kolinsky D. A modern standard for super-small kernels[EB/OL]. Real-Time Magazine 99q4 on RTOS UPDATE II, page22, http://www.realtime-info.be.

[5] Lemieux J. Programming in the OSEK/VDX environment[M]. CMP Books, 1999.

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net