文章编号: 1671－9352(2002)04－0317－03

嵌入式实时系统中分布式 RTOS 的设计与实现

贾智平[1] 潘 辉[1] 孙 宇[2]
(1. 山东大学 计算机科学与技术学院; 2. 山东大学计财处, 济南 250061)

摘要: 在构建了嵌入式实时系统的分布式硬件平台基础上, 研究了嵌入式实时系统中分布式 RTOS 的设计与实现问题, 抽象出应用于此类系统的 RTOS 层次模型, 并给出了实时通信核心技术和分布任务调度策略.

关键词: 分布式系统; 嵌入式系统; RTOS; 任务调度
中图分类号: TP316 文献标识码: A

The Design and Implement of the Distributed RTOS of Embedded Real－Rime System

JIA Zhi－ping, PAN Hui &SUN Yu
(*School of Computer Sci. &Tech., Shandong Univ., Jinan* 250061, *China*)

Abstract: Based on constructed a hardware plane of the embedded distributed real－time system, the design and implement of the distributed RTOS for the embedded real－time system is discussed, the layered model of the RTOS is abstracted, and the technology of the real－time communication kernel and the policy of the distributed task schedule is presented.

Key words: distributed system; embedded system; RTOS; task schedule

计算技术、芯片技术和软件技术的迅猛发展, 现代控制技术、多媒体技术及 Internet 的应用和普及, 加快了消费电子、计算机、通信一体化的步伐, 使嵌入式技术成为后 PC 时代的研究和发展热点.

嵌入式系统的主要特征是实时性, 系统运行不仅要求逻辑上的正确性, 还必须满足相应的时间限制, 所以嵌入式系统一般也是嵌入式实时系统. 在源于网络共享思想的分布式系统不断发展的今天, 人们已经提出了网格计算的思想, 一方面, 嵌入式分布式系统的发展在提高单一系统的性价比、可靠性及可扩展性方面有很大的优势; 另一方面, 嵌入式系统作为计算资源也必将融入全球化的网格计算中, 成为大分布式系统中的一个节点, 因此, 研究适用于嵌入式实时系统的分布式 RTOS (Real－time Operating System)对于嵌入式技术的发展有着广泛的意义.

1 嵌入式实时系统的分布式结构

嵌入式实时系统根据实时性要求分为: 硬实时——系统运行如果无法满足时间限制, 将会造成系统的崩溃, 产生无法挽回的后果; 软实时——超过时限后, 系统是可恢复的. 因此在分布式实时系统的结构设计中首先要满足应用的实时性需要, 此外还应考虑处理器及 I/O 系统的合理拓扑结构、系统进程间通信的速度及实时可预测性、支持错误检测和处理的结构等设计原则. 主要的结构有以下几种[1, 4]:

(1)总线型分布式实时结构

该类结构包括: 共享存储器的多处理器总线结构和多计算机总线结构, 共享存储器的多处理器总线结构主要支持嵌入式硬实时的应用. 这类结构最

收稿日期: 2002－07－04
基金项目: 山东省科技厅资助项目(003090403)
作者简介: 贾智平(1964－), 男, 副教授, 工学硕士, 主要从事计算机网络与技术方面的研究.

?1994-2016 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

大的问题在于需要一种可预测的总线通信协议,即通信协议的实时化,从而达到处理机在访问总线时不会发生冲突.

(2) 环型分布式实时结构

采用令牌通信协议的环形结构,成熟的令牌协议如 IEEE802.4,可以通过对信带宽度、TTRT(Target Token Rotate Time)和数据缓冲区的选择,设置协议周期参数,保证数据的通信满足时限.令牌环通信实时协议的实现要比总线通信实时协议简单,但是需要增加系统的带宽和稳定性.

(3) 总线与环型结合的分布式实时结构

将系统中用于提供软实时服务的部分分离出来,构成实时服务子系统采用扩展性好的总线方式相连,而系统中要求硬实时的执行子系统则采用通信方式简单的令牌环结构,这种结构结合了两种系统的优点,充分考虑了系统的实时要求,又具有易扩展和灵活的特点.

(4) 主从总线式分布式实时结构

基于一种实时控制的分级思想提出的主从总线式结构,实时服务子系统通过总线相连,而实时执行子系统则直接与各节点机相连,便于系统的动态配置和重构,适于特定的分布式实时应用.

根据不同的实时环境要求,选用不同的分布式实时结构,还可通过双总线及双环结构增加系统的容错能力.这些分布式实时系统中除了共享存储器的总线结构外,其余的都属于松散耦合,本文研究的分布式实时系统是指逻辑上紧密耦合的多机系统,这种逻辑上的紧耦合关系就需要通过分布式实时操作系统来实现.

2 嵌入式实时系统的分布式 RTOS

设计应用于嵌入式实时系统的分布式 RTOS,在保证系统的全局一致性和负载均衡的情况下,重点要满足分布式系统对实时性的要求,为此建立面向消息的同步通信机制,使进程调度、资源管理满足分布式处理的要求,从而实现实时通信的可预测性[2].

2.1 层次结构模型

分布式 RTOS 的结构模型如图 1 所示,该模型共分五层:硬件抽象层、分布式实时通信核心、嵌入式实时内核、分布式执行子层、服务模块层.

(1) 硬件抽象层——屏蔽系统硬件特征,以适应嵌入式硬件多样性的特点,便于系统移植.

(2) 嵌入式实时内核——由于嵌入式系统存储资源的限制,采用微内核结构,在实时内核部分实现一些系统必需的功能,如任务调度等.

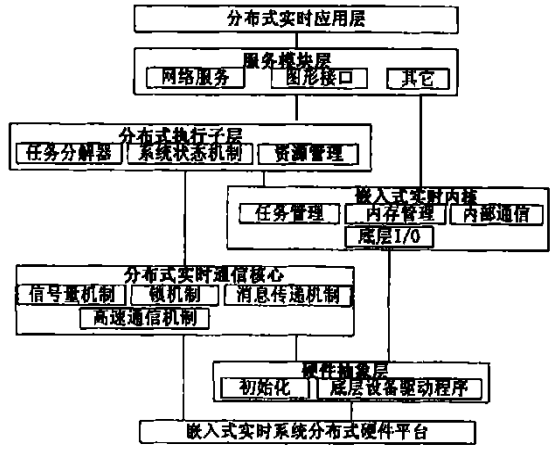

图 1 嵌入式实时系统中分布式 RTOS 层次结构模型

Fig. 1 The layered model of distributed RTOS for embedded real-time system

(3) 服务模块层——以 API 形式提供内核以外的功能,如网络、图形等功能,可以根据用户的需要量身定做,增加或删减某些模块.

(4) 分布式实时通信核心——实时内核运行于单一处理器,在分布式环境下,有多个处理器共同协作完成一个实时任务,分布式实时通信内核提供分布式通信机制和多机协作机制,与实时内核共同构成分布式实时操作系统.

(5) 分布式执行子层——将系统的实时任务分解为逻辑上相互独立的,可在不同的处理器上并行执行的多个子任务,这些子任务及完成时限及资源要求一起下传给实时内核去执行.

以上模型为抽象的分布式 RTOS 的结构模型,虽然由于运行的硬件平台的结构有差异,但对于实时应用有很好的适应性,并可根据需求进行灵活的调整.

2.2 实时通信核心层的功能设计

实时通信核心层是分布式 RTOS 的重要组成部分,主要实现多机通信功能,具体内容如下:

(1) 差错校正.对于在不同的节点间传送的消息,由于物理信道故障或人为破坏所造成的包重复或者包丢失,予以丢弃或重传,以保证消息传递的正确性和完整性.

(2) 流量控制.通过控制各节点的发送流量控制通信线路的负载,防止负载过重、发生阻塞甚至崩溃的现象.同时要注意实时内核与通信核心间信息交换的数量和与时间有关的调度.

(3) 时间管理.通过对每个消息加上时间死限(Deadline),以便接受方根据这一时间来判断消息是否过期,完成实时管理和控制.时间管理需采用全

?1994-2016 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

局时钟, 以使整个系统对发送和接收进行真正的时间死限管理, 实现真正的实时.

(4) 划分优先级. 传送的消息的优先级是由消息的紧急程度、传输时限、消息类型以及要求发送该消息的进程的优先级等因素来决定的, 并且所有发送的消息要根据优先级排队处理.

(5) 系统重构. 当系统中加入新的节点或有节点退出时, 通知系统中其他节点和实时内核, 实现系统的重构.

(6) 地址映射. 以 port 口形式为实时内核调度单位提供通信机制, 通信核心内维护一张 port 口和物理节点地址的映射表并能自动更新, 从而能自动为要发送的数据包填上目的和源节点地址, 保证系统是透明的.

2.3 实时通信核心层的实现

通过实时通信核心层实现通信机制后, 本文设计实现了向实时内核层提供消息传递的一组通信原语: CREATE-PORT, LOCATE-PORT, IPC-SEND-MESSAGE, IPC-JAM-MESSAGE, IPC-RECEIVE-MESSAGE, FREE-PORT 和 DELETE-PORT.

所有消息的传送均是通过上述 port 口来实现的, 通过 CREATE-PORT 可以指定端口缓冲区的大小. 通过 IPC-SEND-MESSAGE, IPC-JAM-MESSAGE, IPC-RECEIVE—MESSAGE 可以进行独立处理器内部进程间的消息交换, IPC-JAM 与 IPC-SEND 功能相同, 只是 IPC-JAM 是把消息放到信道的末尾, 从而为高优先级的消息提供了更多的发送时间.

LOCATE-PORT 原语, 返回连接标识符, 通过查找端口映射表, 得到地址数据等, 在调用 IPC-SEND-MESSAGE 时, 插入这些字段, 形成发送消息的数据结构, 由实时通信核心发送. 而 IPC-RECEIVE-MESSAGE 由 port 口的数据缓冲区中取出消息, 并释放缓冲区.

由此可见, 通过实时通信核心实现的分布式实时通信机制, 是对分布式实时内核透明的, 从而在保证了分布式多机通信及协同工作的同时, 实现了对用户的透明.

2.4 任务调度算法

由分布式执行子层将实时任务分解为逻辑上相互独立的、可在不同的处理机上并行执行的多个子任务集, 而在哪个处理器上处理哪些子任务就需要进行任务的分配和调度, 这就需要任务调度策略, 在实际应用中一般需要采用动态任务调度策略.

动态任务调度策略是在系统运行中, 将任务分配给各处理机, 并对其上的任务数进行动态调整, 以达到系统中各处理器负载均衡的目的, 并尽量减少平均响应时间[3]. 该策略需要有全局的数据结构支持, 我们设计了每个处理器维护一张独立的任务队列, 同时又有一张全局负载状态表的数据结构. 当新任务到达时, 处理器根据动态任务调度算法及全局负载状态表, 确定该任务应加入的处理器的任务队列.

动态调度算法根据调度的着眼点和标准不同, 在设计时可以根据系统的实际要求加以选择. 针对分布式实时操作系统, 基于任务的最短响应时间的动态调度算法如下:

设 Pt $(0 \leq t \leq T)$ 是某一计算机对进入任务响应时间的标量集, 如果能用 M 个以前值得出线性组合来近似这些值, 那么对下一个任务的响应时间的预算值为:

$$\overline{\overline{p(t)}} = \sum_{k=1}^{M} \alpha(k) p(t-k), \qquad 0 \leq t \leq T$$

其中, $a(k)$ 为预算系数. $p(t)$ 和 $\overline{\overline{p(t)}}$ 的误差为:

$$e(t) = \overline{\overline{p(t)}} = p(t) = \sum_{k=1}^{M} a(k) p(t-k);$$

所有带负下标的变量都置为 0.

用最小二乘法于 $e(t)$, 得

$$E(M, T) = \sum_{t=0}^{T} e(t)^2 = \sum_{t=0}^{T} (p(t) + \sum_{k=1}^{M} (a(k) p(t-k))^2;$$

相对预测系数 $a(k)$, 求其导数并置为零, 得到方程:

$$\sum_{k=1}^{M} a(k) \sum_{T=0}^{T} p(t) p(t-i) + \sum_{t=0}^{T} p(t) p(t-i) = 0;$$

$$p(t-i) = 0;$$

其中 $l \leq i \leq M$.

由该线性方程组, 可以得到相应的预测系数 $a(k)$, 然后就可以用该方法来预算计算机对下一个任务的响应时间, 由此当新任务到达时, 根据响应时间的大小, 可以将该任务分配到对它响应时间最短的计算机中, 从而完成对任务的动态调度.

参考文献:

[1] Andrew S. Tanenbaum : Distributed Operating System [M]. Prentice Hall, 1999.

[2] K G Shin HARTS. A distributed real— time architecture[J]. IEEE computer, 1991; 25~35.

[3] Stewart D B, Kbosla P K. Real— time scheduling of sensor— based control system[J]. In: Proc Eighth IEEE Workshop of Real— time Operating System. Software Atlanta GA, 1991; 144~150.

[4] Kopetz H. Real— time system[A]. In: Mc Demid J ed. Software Engineer′s Reference Book. Boca Raton FL: CRC Press, 1991.

?1994-2016 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net