实时嵌入式双操作系统架构研究综述

张美玉[1],张倩颖[1,2],孟子琪[1],施智平[1,3],关 永[1,4]
(1. 首都师范大学信息工程学院,北京 100048;2. 成像技术北京市高精尖创新中心,北京 100048;
3. 轻型工业机器人与安全验证北京市重点实验室,北京 100048;4. 电子系统可靠性技术北京市重点实验室,北京 100048)

摘 要: 随着全球工业 4.0 战略的提出,工业控制及航空航天等领域对嵌入式系统的要求越来越高,对实时性具有严格要求同时还需要应用功能丰富的通用操作系统的支持.针对该需求现状,学术界和产业界提出双操作系统的解决思路,即同时运行实时操作系统和通用操作系统.本文归纳总结国内外嵌入式双操作系统的研究现状,对各种实现技术进行深入探讨和分析,列举嵌入式双操作系统典型应用和主要应用领域,最后对该类技术的研究趋势进行总结.

关键词: 双操作系统;嵌入式操作系统;实时性;虚拟化;双核
中图分类号: TP316 **文献标识码:** A **文章编号:** 0372-2112 (2018)11-2787-10
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2018.11.029

A Survey of Research on Real-Time Dual-OS Architecture for Embedded Platform

ZHANG Mei-yu[1], ZHANG Qian-ying[1,2], MENG Zi-qi[1], SHI Zhi-ping[1,3], GUAN Yong[1,4]
(1. *College of Information Engineering, Capital Normal University, Beijng* 100048, *China*;
2. *Beijing Advanced Innovation Center for Imaging Technology, Beijing* 100048, *China*;
3. *Beijing Key Laboratory of Light Industrial Robot and Safety Verification, Beijing* 100048, *China*;
4. *Beijing Key Laboratory of Electronic System Reliability Technology, Beijing* 100048, *China*)

Abstract: With the proposing of Global Industry 4.0, the requirements for the embedded system become higher in industrial control, aerospace and other fields which have strict requirements for real-time and also need the supports of the application-rich general-purpose operating system. In view of this situation, Academia and Industry fields proposed a solution called dual operating system that a real-time operating system and a general-purpose operating system run at the same time. This paper summarizes the current states of dual operating system for embedded platform both at home and abroad, discusses and analyzes various implementation technologies, lists the typical application and main application fields of dual operating system for embedded platform, and finally summarizes the research trend of dual operating system for embedded platform.

Key words: dual operating system; embedded operating system; real-time; virtualization; dual-core

1 引言

嵌入式系统出现于上世纪 70 年代,最初应用于工业控制和国防系统等领域,负责控制这些领域的核心组件,对实时性和安全性等具有较高要求,因此当时的嵌入式操作系统主要是能够提供精确实时控制功能的实时嵌入式操作系统.随后,嵌入式的发展使嵌入式系统迅速得到广泛的应用,其功能和软件数量也日趋增长.同时以智能制造为主的全球工业 4.0(Industry4.0)的发展趋势对传统嵌入式系统提出了信息化和自动化等功能需求,因此传统单一嵌入式实时操作系统已经难以满足当前相关领域提出的新要求.

目前操作系统主要分为两类:通用操作系统(General-Purpose Operating System,GPOS)和实时操作系统(Real-Time Operating System,RTOS).通用操作系统,譬如 Linux 和 Windows,主要目标是提供丰富的应用功能,在计算能力方面注重系统的吞吐率;实时操作系统则强调输出结果的逻辑正确性和时效性,保证任务的可预测性.为了保证实时性,RTOS 通常最小化软件数量,尽量减少一些不必要的功能.一方面,嵌入式操作系统负责系统运行的核心关键功能,对实时性具有严格要求;另一方面,信息化和实时性要求使得传统单核嵌入式操作系统难以

收稿日期:2017-11-14;修回日期:2018-02-08;责任编辑:蓝红杰
基金项目:国家自然科学基金(No. 61602325, No. 61572331, No. 61472468, No. 61702348);国家重点研发计划(No. 2017YFC0806700, No. 2017YFB1301100);国家科技支撑计划(No. 2015BAF13B01);北京市科委(No. LJ201607)

满足嵌入式领域的发展需求,需要应用功能丰富的通用操作系统的支持.要想在保留通用操作系统丰富应用程序功能的前提下同时拥有实时性,双操作系统架构成为新的解决思路.在双操作系统架构中,GPOS 内核和 RTOS 内核运行在同一硬件平台,共享底层硬件资源.其中,GPOS 提供满足信息化需求的非实时性功能,保证系统的功能需求;RTOS 提供精确的实时控制和任务管理功能,保证系统的实时性需求.

本文对国内外嵌入式双操作系统实现技术进行调研,并对现有架构进行分类及比较,对嵌入式双操作系统应用进行举例,总结嵌入式双操作系统的发展趋势,最后对全文进行总结与展望.

2 双操作系统要求

嵌入式系统是一种专用计算机系统,通常被用来控制精密机械.高实时性操作系统是嵌入式系统的基本要求.双操作系统由于其架构复杂,具有更加严格的要求.一般来说,双操作系统架构必须满足如下要求[1,2]:

(1)实时性.实时性是实时操作系统最基本的要求.在 RTOS 中,任务调度机制与其他操作系统不同,这确保实时中断发生时可以抢占当前任务占用的系统资源从而优先得到处理.在双操作系统架构中,GPOS 的运行可能会对 RTOS 造成干扰,因此必须在架构设计上保证 RTOS 实时性.

(2)安全性.在双操作系统架构中,GPOS 的软件错误可能会导致对 RTOS 中实时任务、内存和外设的攻击,使 RTOS 的实时性和安全性得不到保证.因此,需要对两个操作系统进行必要的隔离来保护 RTOS,阻止 GPOS 对实时任务和安全数据的篡改,以保证 RTOS 的独立性和安全性.

(3)开销.双操作系统架构往往需要虚拟化等技术的介入与支持,这必然会增加系统开销,所带来的性能损失会影响实时任务的实时性.嵌入式系统资源有限,系统开销最小化也是实现双操作系统的必要条件之一.

(4)硬件支持.在嵌入式操作系统发展过程中,CPU 硬件逐渐加入了对双操作系统架构的支持,例如基于 CPU 硬件隔离的 ARM TrustZone 技术,实现 RTOS 对 GPOS 的隔离,实现实时性的同时增加了系统安全性.

(5)可维护性.可维护性的高低直接关系到系统可靠性和安全性,也是衡量软件质量的标准之一.在双操作系统架构中,GPOS 和 RTOS 紧密配合,同时可能需要对操作系统的代码进行修改,所以对操作系统代码尽量少的修改量才能适应操作系统内核的快速发展.

3 实时嵌入式双操作系统架构实现技术

目前实时嵌入式双操作系统的实现有虚拟化、轻量级虚拟化等多种方式.在本节中,将分别对国内外实时嵌入式双操作系统架构的各种实现方式进行归纳和总结,并列举目前实时嵌入式双操作系统的一些典型应用.

3.1 虚拟化技术

在单处理器情况下,为了同时运行多个操作系统,虚拟化技术[3,4]是一种较好的解决方法.在广义上,虚拟化技术是通过将计算机体系结构中某一层或某一类的资源进行模拟,然后映射到上层构建的虚拟机(Virtual Machine,VM)中,以实现资源复用[5],提高系统灵活性[6].通过在操作系统和硬件资源之间构建特殊的软件管理层,称为虚拟机监视器(Virtual Machine Monitor,VMM)或虚拟机监控程序(hypervisor),来虚拟化硬件资源并进行管理.在嵌入式双操作系统架构中,通过 hypervisor 构建两个虚拟机,分别运行 RTOS 和 GPOS,架构如图 1 所示,hypervisor 负责两个操作系统的切换并保存或恢复上下文状态.

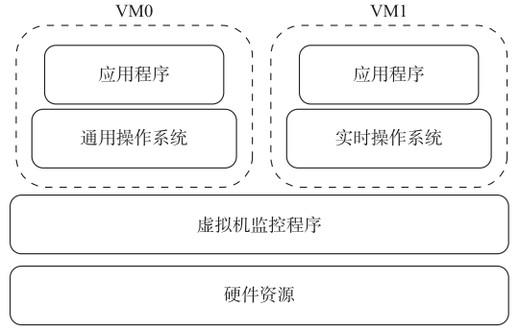

图1 虚拟化的实现

3.1.1 具体实现

国内外基于虚拟化方式的实时嵌入式双操作系统典型方案包括 VLX[7]、OKL4 Microvisor[8]、Xen on ARM[9,10]、KVM/ARM[11,12]、Xtratum[13]和 Xvisor[14]等.下文以 Xen on ARM 为例进行具体分析.

Xen on ARM 是高性能开源虚拟机监控器 Xen[15]在基于 ARM 嵌入式平台上的扩展.在双操作系统架构中,通过 Xen 构建两个虚拟机,分别运行 GPOS 和 RTOS,Xen 运行在操作系统底层以提供统一的硬件资源访问接口.Xen on ARM 架构中,ARM 虚拟化扩展提供了 3 个级别的运行模式:EL0(用户模式)、EL1(内核模式)和 EL2(管理程序模式),并引入了新的指令 HVC,内核使用 HVC 指令来向 Xen 发出调用,以便在内核模式和管理程序模式之间切换.系统运行时,Xen 根据需要负责 RTOS 和 GPOS 的切换并保存上下文,恢复被切换的任务.这种方法虽引入一定开销,但实现了双操作系统并行运行,使得 Xen on ARM 在业界得到广泛应用.

学术界在应用分配与实时调度算法等方面对 Xen on ARM 架构进行了优化. 摩德纳大学计算机工程系将开源实时操作系统 ERIKA Enterprise 与 Linux 集成[16], ERIKA Enterprise 用于汽车应用的硬实时任务支持, Linux 用于满足用户浏览新闻及音视频娱乐等功能,为用户提供灵活友好的交互界面;通过增加管理程序 Xen 作为软件抽象层来对共享内存进行管理及处理中断. 针对 Xen 默认调度算法 Credit 不具备抢占式特性并且不能保证实时性的缺点,上海交通大学通过将实时中断感知、事件驱动机制及系统负载平衡方法映射到 Xen 的 Credit 调度程序[17]上以此提高 Xen 性能,优化 RTOS 调度延迟和响应时间,并通过实验数据表明,对实时性能的改善约 20%.

3.2　轻量级虚拟化技术

在 3.1 节的虚拟化技术中,管理程序对系统资源的调度依赖于复杂调度算法[18,19]. 与虚拟化调度策略不同的是,轻量级虚拟化技术采用混合操作系统架构,将上层操作系统上运行的进程封装为下层操作系统的一个(或一组)进程运行,其架构如图 2 所示,通常下层为 RTOS,上层为 GPOS,以同时满足嵌入式系统实时性和功能性的需求.

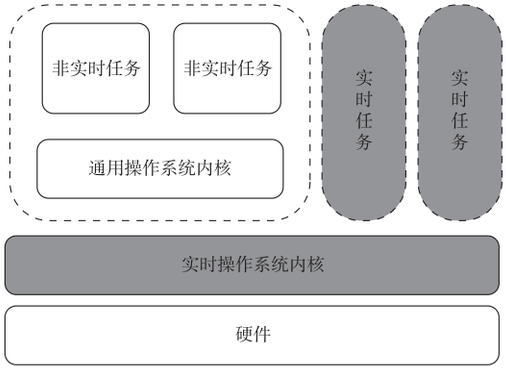

图2　混合操作系统架构图

在混合操作系统架构中,RTOS 内核具有最高优先级,当外部中断到来时,实时内核最先拦截中断并进行处理,无需经过 GPOS 的调度以免增加延迟. 类似这样的混合操作系统架构有: Linux on ITRon[20]、RTLinux[21]、RTAI[22]、Xenomai[23]等. 下文将对实现轻量级虚拟化技术的混合操作系统架构的两种方法,微内核(micro-kernel)方法和纳内核(nano-kernel)方法,进行详细介绍.

3.2.1　微内核方法

微内核[24]架构不同于 Linux 等宏内核架构,它只在内核态保留操作系统最核心的功能,将大多数系统服务放在用户态执行. 微内核最典型的是 L4 微内核系列[25],最初由德国计算机科学家 Jochen Liedtke 设计,现已成为第二代微内核系列设计标准. 基于 L4 衍生出多种微内核操作系统版本,其中在实时操作系统领域典型的是 Fiasco[26].

Fiasco 是一个可抢占的具有实时性的微内核,可与 GPOS 如 Linux 同时运行. 基于 Fiasco 微内核的双操作系统架构[27]如图 3 所示,Fiasco 直接运行在硬件之上,负责硬件资源的分配调度及处理实时中断,修改后的 L4/Linux 系统作为 Fiasco 的一个任务执行. 微内核的设计使更多的系统服务被放在用户模式下执行. 为了保证实时内核可以最先截获到硬件中断并能及时处理,Fiasco 内核比 Linux 具有更高的优先级,同时 L4 微内核的安全机制会保障实时任务的正确执行而不受 Linux 的干扰.

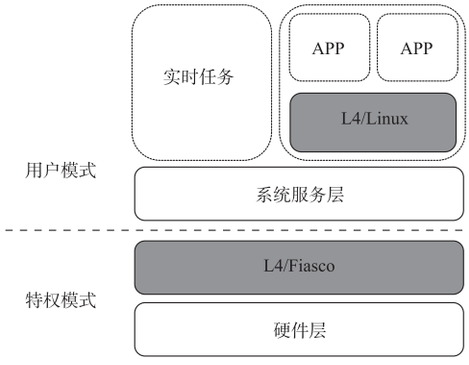

图3　微内核双操作系统架构图

3.2.2　纳内核方法

在纳内核方法中,操作系统可适应性域环境 ADEOS[28,29](Adaptive Domain Environment for Operating System)提供了一套介于 Linux 和硬件之间的纳内核,为在其上运行的操作系统提供操作系统原语和机制以实现硬件资源共享,ADEOS 为上层操作系统内核提供两个可运行的域(domain)环境,分别运行实时操作系统内核和通用操作系统内核,形成双内核混合操作系统结构如图 4 所示,并使用中断管道(I-pipe)在域之间传递中断信息. I-pipe 保证了当外部中断到来时,最高优先级的实时内核可以优先处理中断. 但 ADEOS 只是提供了一个执行环境,需要和其他项目协作完成系统运行,如 Xenomai 和 RTAI3.2 以上版本都是基于 ADEOS 实现. 下文以 Xenomai 为例说明如何基于 ADEOS 实现双操作系统架构.

Xenomai[30]是一个为标准 Linux 内核添加实时功能并使各种实时操作系统 API 可用于基于 Linux 平台的自由软件框架. Xenomai3 的双内核架构如图 5 所示,基于 ADEOS 隔离出一个域运行实时内核,将 Linux 内核作为该实时内核的空闲任务,仅当实时内核空闲时才

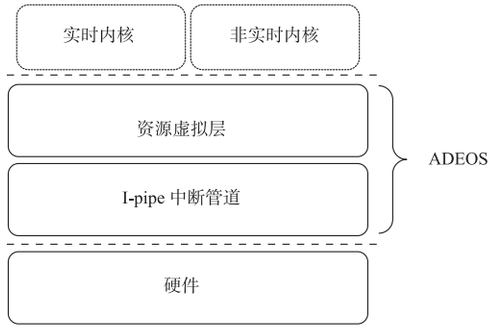

图4 ADEOS 结构图

调度 Linux 内核,且可随时抢占 Linux 内核占用的系统资源进行实时任务的处理. 在实时操作系统接口兼容方面,Xenomai 通过提供一层应用中间件 skin 来兼容 VxWorks、pSOS 等传统 RTOS API,保证其他常规 RTOS 应用移植到 Linux 上运行而无需重新编写程序,在应用方面具有一定的优势.

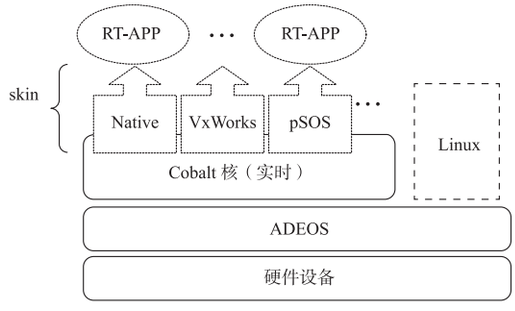

图5 Xenomai/Linux 架构图

3.3 硬件辅助虚拟化技术

虚拟化技术的最大优点在于实现了硬件资源复用,但如果不进行严格的内存隔离,会导致实时任务运行故障或敏感信息被窃取. 硬件辅助虚拟化技术是通过在微处理器体系结构中增加特殊的指令来利用硬件辅助实现虚拟化,该技术在硬件之上构建一个安全内核,为安全敏感任务提供一个可信执行环境[32](Trusted Execution Environment,TEE). 在移动嵌入式安全领域,TEE 在移动支付、数字版权管理(DRM)、近场通信(NFC)等安全相关功能有着重要应用. 与 TEE 对应的 REE(Rich Execution Environment,普通执行环境)为系统提供大部分功能应用的执行环境.

在嵌入式领域中,最早引入硬件辅助虚拟化的是 ARM 公司的 TrustZone 技术[33]. ARM TrustZone 技术通过对处理器进行重构,利用 TZASC 组件(TrustZone Address Space Controller)和 TrustZone 存储适配器将内存划分为两个独立的物理地址空间,形成隔离的操作系统执行环境. TrustZone 把整个计算机资源划分为安全区(secure world)和普通区(normal world),为了进行两个区域的切换,ARM 处理器增加了一个新的模式,安全监视模式(Secure Monitor Mode),运行于该模式的软件被称为监控器(monitor),负责保存上下文环境,恢复被切换的状态. 类似这样的监控器有 TLR[34]、ViMoExpress[35]、SafeG[36,37]、ARMithril[38]、TMM[39] 和 RTZVisor[40,41]等. 以 SafeG(Safety Gate)为例,双操作系统架构[42]如图 6 所示. TrustZone 技术将 RTOS 和 GPOS 分别分配在两个独立的虚拟机中,SafeG 运行在监视器模式,负责加载操作系统,接受系统调用并控制上下文切换. RTOS 的设备和内存数据以及 SafeG 被划分为安全区,GPOS 禁止访问安全区,从而保证了 RTOS 的安全性. 为了保证实时性,SafeG 的调度遵循两个重要原则:空闲调度和处理器控制权快速恢复. 空闲调度原则指的是在系统运行时当且仅当 RTOS 空闲时才会调度 GPOS 任务运行,充分保证了 RTOS 的实时性. 然而空闲调度对于 GPOS 的软实时任务如视频播放会产生消极影响,对此日本名古屋大学 Sangorrin D 等人又提出集成调度方法[43],将 GPOS 高优先级任务与 RTOS 实时任务混合分级,达到综合调度两个操作系统,兼顾所有高优先级任务的目的;处理器进入监视器模式的途径之一是 FIQ(Fast Interrupt Request,快速中断请求),处理器控制权快速恢复原则指的是 GPOS 运行时 FIQ 不被禁用,保证 RTOS 随时可以通过 FIQ 取得处理器的控制权,保证实时任务的执行.

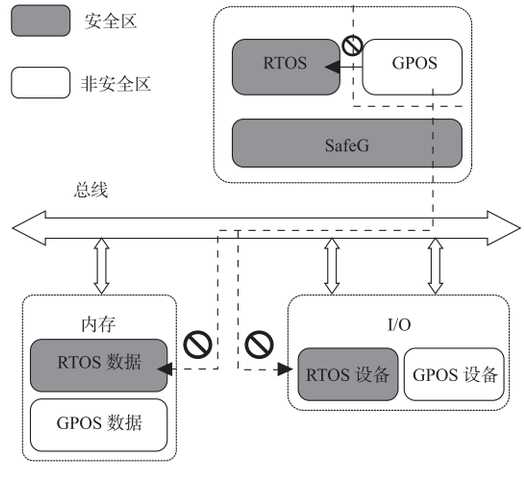

图6 双内核架构图

3.4 基于多核 CPU 的双操作系统

针对高频率时钟会导致单处理器发热过高的问题,CPU 厂商提出了多核 CPU,采用多线程并发执行的技术实现更强的并行计算能力,并带来更少的功耗和更高的效率. 在 MP(Multi-Processing)技术中,对称多处

理器(Symmetric Multi-Processor,SMP)越来越多的被应用.SMP 允许在一个芯片上集成两个或多个相同的 CPU,各 CPU 之间通过系统总线来实现内存等资源的共享和相互通信,从而大大提高了系统数据处理能力.SMP 结构图如图 7 所示.

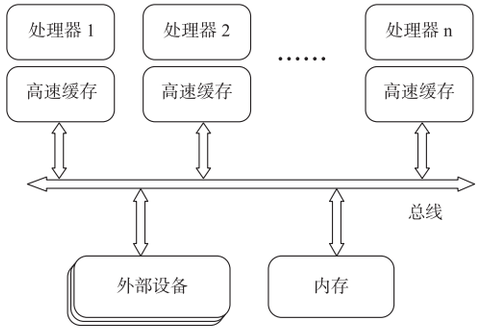

图7　SMP的典型结构

构建实时嵌入式双操作系统时,多核架构抛弃了传统的在硬件和操作系统之间插入一个软件层管理资源的方法,而是在适当增加硬件开销的基础上实现并行处理进程,提高系统运行效率.

美国亚特兰大大学提出的 DynOS SPUMONE[44] 基于嵌入式多核处理器内核的动态分配操作系统来实现多核 CPU 上运行多个操作系统,实现了在多核处理器上并存双操作系统.德国 Heinz Nixdorf 研究院提出基于 PowerPC 的嵌入式系统多核管理程序 Proteus[45],无需特殊硬件支持即可实现虚拟化.日本早稻田大学计算机科学与工程系 Kinebuchi Y 等人开发实现精简虚拟机管理程序 SPUMON[46],共同执行 TOPPERS RTOS 和 Linux,并提出两种方法降低中断延迟:在配备 SH-4A 体系结构内核的多核处理器上利用 SH-4A 处理器支持的中断优先级(interrupt priority level,IPL)机制保证 RTOS 可以即时抢占 GPOS 的资源;利用虚拟内核迁移技术为 RTOS 提供安全独立的执行环境,防止 GPOS 内核活动阻塞 RTOS 中断.

国内北京航空航天大学开发出基于多核处理器架构的双操作系统中间件 RGMP(RTOS and GPOS Multi-Processor),其架构[47]如图 8 所示.

RGMP 使用 Linux 作为 GPOS,并允许开发者根据需要移植不同的 RTOS.RGMP 将 GPOS 和 RTOS 分别运行在独立的 CPU 核上,两个操作系统之间通过共享内存进行通信.在中断处理方面,RGMP 引入了 SMP 技术的名为 I/O 高级可编程中断控制器(I/O APIC)的组件.此外,每个 CPU 都有自己的本地 APIC(Local APIC,LAPIC).在 RGMP 架构中,外部设备被分成实时设备和非实时设备,分别由 RTOS 和 GPOS 管理.当外部设备产生中断信号时,I/O APIC 把信号传递给相应 CPU 的

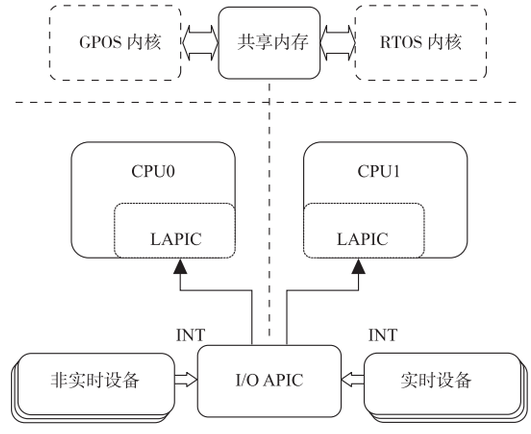

图8　RGMP架构图

LAPIC,由 LAPIC 中断当前 CPU 的进程来处理中断.

对于多核处理器,其性能和效率等各方面相对于单核处理器有很大的提升,更好的实现了资源复用,在相同条件下,多核处理器的使用可以获得更高的性能和更好的计算能力.

3.5　基于多处理器的双操作系统

随着嵌入式的发展和更多复杂任务的出现,嵌入式领域的软件复杂性也迅速增加.多处理器系统由不同芯片上的多个处理器组成,开发者可以根据不同的开发需要对处理器进行搭配,以提升系统处理复杂性和多样性任务能力.

Meier L 等人采用嵌入式平台搭配 Linux 设计联合体系结构,实时节点运行在实时嵌入式平台上,控制器运行在 Linux 配套计算机上,通过两个平台的整合协调,实现了使用发布/订阅设计模式的深度嵌入式平台完全多线程模块化机器人框架[48],系统在硬件和软件方面都具有高度可扩展性.

基于现场可编程门阵列(Field Programmable Gate Array,FPGA)可重构计算[49,50]可实现系统动态配置,具有可重编程、高性能和高灵活性等特点,在嵌入式领域具有广泛应用.WEI H X 等人使用 ARM 处理器和 FPGA 两种硬件为模块化可重构机器人(Modular Reconfigurable Robot,MRR)开发了可重构控制器,架构[51]如图 9 所示.该控制器在 ARM 处理器上运行实时操作系统 μC/OS,负责运动规划、数据处理和文件系统管理等任务;FPGA 上运行的 μC/OS 可以完成运动控制和系统输入输出等任务.当 MRR 的配置发生变化时,控制器将会动态地重新配置 FPGA 的内部逻辑结构.

基于 ARM/FPGA 的多处理器系统各自分工、互相配合,能够高效的处理系统任务,实现并行计算.目前嵌入式处理器有超过 1000 多种,大多数都具有较强的实

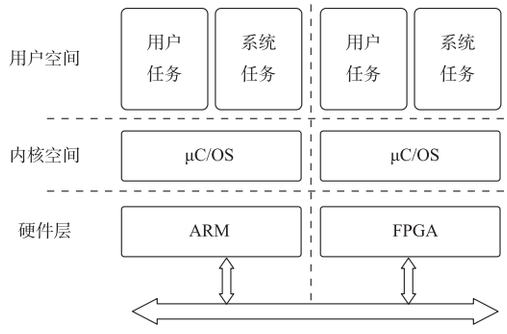

图9 可重构控制器架构图

时任务处理能力以及中断处理能力,除 FPGA 和 ARM 系列之外,Power PC、MIPS 和 DSP 等处理器也具有非常出色的性能.

3.6 比较与分析

上文对各种实时嵌入式双操作系统架构进行了深入的探讨,并分析其技术实现,下文将对各种技术进行比较和分析,并对其进行总结.

虚拟化技术使系统服务请求与资源交付功能分离. 传统的虚拟化解决方案基本遵循两种实现方法:全虚拟化和半虚拟化[52,53]. 嵌入式系统的特征[54]之一是由于计算资源有限要求操作系统足够小巧,半虚拟化技术通过修改客户操作系统来配合 hypervisor 的协调工作,操作系统可直接管理底层硬件资源,性能比全虚拟化更高. 然而半虚拟化技术的适用性是有限的,不是所有的操作系统都支持其编程接口,同时对客户操作系统的代码修改使其可维护性降低[55].

在混合操作系统轻量级虚拟化中,RTOS 内核的优先级高于 GPOS,保证了实时性,但如果 RTOS 的任务负载过重,会导致 GPOS 的 CPU 分配时间不足,且 hypervisor 同样会导致开销问题[42]. 韩国高丽大学创新信息与计算研究所 Kim B 等人在 ARM Cortex-A15 处理器上对单独执行 RTOS 和同时运行 FreeRTOS 或 uC/OS 和 Linux 操作系统的性能[56]进行了量化测试,实验结果表明由于管理程序的调度,RTOS 实际运行时间与系统预估时间不匹配,系统性能轻微下降.

硬件辅助虚拟化需要硬件平台的支持,该特点与 hypervisor 支持的虚拟化相比有一定的局限性. 在处理器层面,硬件辅助虚拟化技术最重要的变化是划分硬件资源,而不是共享虚拟化之后的硬件资源,减少了 RTOS 的隔离负载[57],但设备共享也增加了系统硬件负担,Sangorrín D 等人提出基于设备的新动态共享资源划分机制[58],提高了系统可靠性和实时性. 在性能方面,利用 ARM TrustZone 硬件技术,精简的虚拟化架构也具有较低的开销、较高的性能实现[59]及高效通信[60].

单处理器上可以利用虚拟化技术实现双操作系统,但是高频率时钟导致的发热问题会使系统性能下降. 多核(multi-core)技术真正实现了并行处理技术,提高了系统的计算能力,大大提高性能并降低了成本[61]. 但多核虚拟化技术增加了系统的复杂性,如多个操作系统由于共享内存出现的资源竞争需要用同步机制来解决问题,同时会带来一定的系统开销[62]. 从性能角度来看,系统架构的复杂性似乎会导致系统性能下降,然而有研究表明,多核体系结构的实现可能会导致性能几近线性增加[63].

双核处理器是在一个芯片上集成了两个 CPU,而多处理器系统是由不同芯片上的多个处理器组成,且操作系统必须支持多处理器工作[64]. 多处理器系统具有专用性,系统芯片往往根据不同功能需求专业定制,具有集成专业硬件模块的优势,优化特定应用程序任务的执行,但同时适用范围小.

表 1 对各种方法的优缺点进行了总结,L 代表 Low,表示基本不符合;M 代表 Medium,表示基本符合;H 代表 High,表示非常符合;Yes/No 表示是否需要硬件支持. 由该表可以看出,硬件辅助虚拟化技术及多核架构满足嵌入式双操作系统的更多方面要求. 嵌入式系统的趋势是利用多核处理器芯片整合双操作系统,实现更高的 CPU 利用率. 因此,随着 ARM 和 FPGA 等技术的不断进步,他们将会成为今后嵌入式系统的主要发展方向.

表 1 不同方法的比较结果

方法 要求	虚拟化	轻量级 虚拟化	硬件辅助 虚拟化	双核 技术	多处 理器
低开销	M	L	H	M	M
硬件支持	No	No	Yes	Yes	Yes
安全性	M	L	H	H	H
实时性	H	H	H	H	H
可维护性	L	L	H	H	M

4 双操作系统架构在嵌入式的应用

嵌入式系统在各个领域都有广泛应用,应用领域包括工业控制、国防航空及空间系统、消费电子和医疗器械等. 下文将对双操作系统架构在嵌入式的应用进行简要举例说明.

4.1 机器人操作系统

在工业控制领域,机器人的发展始终都与嵌入式的发展紧密结合. 在众多机器人开发框架中,Willow Garage 与斯坦福大学合作开发的用于编写机器人软件的机器人操作系统 ROS[65](Robot Operating System)以其免费开源等特性得到广泛应用,但如果没有实时性的支持,严重者甚至会导致生命财产受到损失. 国内外

越来越多的研究组专注于提高 ROS 实时性研究. 由于 ROS 主要支持 Linux,不满足硬实时需求,实时性需要单独实时操作系统的支持.

Wei H 等人利用双核 RGMP 技术实现 RT-ROS[66],其架构同 RGMP 架构,把 ROS 节点分为实时节点和非实时节点,分别运行在 RTOS 和 Linux 上,以实现 ROS 的实时性. Dong Q 等人实现了在双核处理器上集成 RTOS 和 GPOS 的方法,通过在 RTOS 上运行高精度 RGMP-ROS[67]来实时控制机器人应用.

4.2　实时护理管理系统

医疗卫生永远是最受关注的民生问题之一. 随着病人数量的增加,患者的护理问题也成为不小的难题. 在一定程度上,科技的发展可以缓解医疗资源紧张的问题.

Palaniappan R 等人利用 Xenomai 实现了医疗保健行业的实时护理管理系统[68]. 该系统作为一个中央管理系统,配合以基于射频的遥控器和可穿戴设备如智能手表等设备的支持,患者可以使用遥控器在任何时间点击需求按钮,向护理人员传达信息以得到及时的护理;可穿戴设备负责监测患者的基本生命体征如体温、血压和心率等,如遇紧急情况可立即上报给中央管理系统,这就要求系统有实时中断处理的能力. 同时,这些数据信息被存储在后台数据库中以便进行大数据分析,得到有效的信息.

4.3　移动设备操作系统

近年来,市场上基于 ARM 的各种智能设备备受欢迎,用户越来越期望这些设备能够执行大量任务,包括网页浏览和游戏,并可以下载并运行更多其他应用程序. 与此同时,智能设备也存在着安全威胁,如网上交易支付、第三方 APP 授权管理和恶意软件攻击等,这些潜在的安全隐患引起了广泛关注. ARM 芯片的 TrustZone 技术可被用来二次开发来提高安全性,例如指纹识别技术,通过指纹传感器将用户指纹数据存放在安全区域来保护数据等.

Santos N 等人为移动设备上开发安全敏感的应用程序设计了 TLR(Trusted Language Runtime)系统[69],利用 ARM TrustZone 技术的支持将安全应用放在可信环境中执行,保护安全数据的机密性和完整性. Hwang J Y 等人使用 Xen 管理程序实现了 ARM 平台的虚拟机管理软件[70],将可信计算能力引入移动嵌入式设备,并通过实验计算表明,Xen on ARM 带来的系统开销问题也在可接受的范围. Yoo S 等人通过设计实现轻量级管理程序 MobiVMM[71],将不同实时安全需求的应用程序分别放在多个虚拟机中隔离,来解决智能设备实时性、安全性和能耗问题.

5　嵌入式双操作系统研究趋势

目前,嵌入式操作系统的发展迅速,已经应用到智能家居、消费电子和工业控制等各个领域,这些领域对嵌入式产品的要求也越来越高. 从硬件技术、开发和应用需求等各方面的发展变化,下文对嵌入式双操作系统架构未来的研究发展趋势进行总结.

定制化. 嵌入式双操作系统将面向特定应用和任务提供操作系统功能. 开发者将利用某些操作系统譬如 Linux 可定制的特点对内核进行裁剪修改以适应特殊环境应用,通过装卸某些模块来互相配合以实现特定功能.

标准化. 技术的快速发展需要一种标准来规范. 譬如虚拟化技术,VMWare、Microsoft 和 Dell 等公司提出开放虚拟化格式(Open Virtualization Format,OVF),为虚拟机在不同平台之间提供一种标准的封装分发方法. IBM、OpenStack 以及 Citrix 等更多嵌入式厂商都加入了 OVF,使虚拟机具有可移植性,实现虚拟机的跨平台复用,减少系统开发周期. 随着嵌入式操作系统的广泛应用和发展,兼容性等各种问题需要建立相应的标准来规范其应用.

轻量级. 虚拟机监控程序对资源的动态分配和调度是实现虚拟化技术的关键. 但虚拟机监控程序的调度不可避免的会引起系统开销增加,导致 RTOS 的性能下降. 因此,轻量级虚拟机监控程序的设计和优化可以减少系统开销,提高系统运行效率,对提升系统性能有着至关重要的作用.

安全性. ARM 平台硬件辅助虚拟化技术的出现使嵌入式双操作系统的安全性增加. 随着嵌入式操作系统的发展,源码的可靠性也越来越高. 智能移动设备敏感数据的保护也至关重要. 因此嵌入式操作系统对安全设计的要求也愈加严格.

6　总结

双操作系统架构的提出同时满足了实时嵌入式系统的实时性要求和功能需求. 本文研究了国内外嵌入式双操作系统架构发展状况,分类归纳各种实现技术的原理,并基于每一类技术的典型实现对该类技术的双操作系统架构构建方法进行总结,按照不同的实现技术分别总结其技术原理. 单处理器主要运用虚拟化技术将硬件资源抽象,利用虚拟机监控程序提供多操作系统的技术支撑,以达到宏观上同时运行双操作系统的目的. 但是安全方面如信息篡改和恶意软件等问题欠缺考虑. 硬件辅助虚拟化解决了单内核安全方面的问题. 尽管单处理器可以实现双操作系统,但其只是宏观上的并行运行,微观上同一时刻仍然只能运行一

个进程.为了实现真正意义的多线程以及满足不同复杂任务的需求,多核和多处理器应用也很广泛.

参考文献

[1] Armand F, Gien M. A practical look at micro-kernels and virtual machine monitors [A]. Consumer Communications and Networking Conference [C]. Las Vegas: IEEE Press, 2009. 1 – 7.

[2] 孔德岐,李亚晖,郭鹏. 高可靠嵌入式计算机系统的发展[J]. 通信学报,2013,34(Z1):170 – 175.
Kong Deqi, Li Yahui, Guo Peng. Development of dependable embedded computer systems[J]. Journal on Communications, 2013, 34(Z1): 170 – 175. (in Chinese)

[3] Chiueh S N T, Brook S. A survey on virtualization technologies [J]. Rpe Report, 2005, 1 – 42.

[4] Sheridan-Barbian K K. A survey of real-timeoperating systems and virtualization solutions for space systems [D]. Monterey, California: Naval Postgraduate School, 2015.

[5] Sandström K, Vulgarakis A, Lindgren M, et al. Virtualization technologies in embedded real-time systems [A]. Emerging Technologies & Factory Automation (ETFA) [C]. Cagliari, Italy: IEEE Press, 2013. 1 – 8.

[6] Weltzin C, Delgado S. Using virtualization toreduce the cost of test [A]. Autotestcon [C]. Anaheim: IEEE Press, 2009. 439 – 442.

[7] VirtualLogix, Inc. VirtualLogix homepage [EB/OL]. http://www. virtuallogix. org/, 2017-08-08.

[8] Heiser G, Leslie B. The OKL4 microvisor: convergence point of microkernels and hypervisors [A]. ACM Sigcomm Asia-pacific Workshop on Systems [C]. New Delhi, India: DBLP, 2010. 19 – 24.

[9] Wikipedia. Xen ARM with Virtualization Extensions whitepaper [EB/OL]. https://wiki. xen. org/wiki/Xen_ARM_with_Virtualization_Extensions_whitepaper, 2017-07-13.

[10] Yoo S, Yoo C. Real-time scheduling for xenarm virtual machines [J]. IEEE TransactionsonMobile Computing, 2014, 13(8): 1857 – 1867.

[11] KVM/ARM. KVM/ARM Open Source Project [EB/OL]. http://systems. cs. columbia. edu/projects/kvm-arm/, 2017-08-01.

[12] Dall C, Nieh J. KVM/ARM: the design and implementation of the linux ARM hypervisor [J]. ACM SIGARCH Computer Architecture News, 2014, 42(1): 333 – 348.

[13] Masmano M, Ripoll I, Crespo A, et al. Xtratum: a hypervisor for safety critical embedded systems [A]. 11th Real-Time Linux Workshop [C]. Dresden, Germany: Real Time Linux Workshop, 2009. 263 – 272.

[14] Patel A, Daftedar M, Shalan M, et al. Embedded hypervisor xvisor: A comparative analysis [A]. Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP) [C]. Turku, Finland: IEEE, 2015. 682 – 691.

[15] Xen Project. The Xen Project, the powerful open source industry standard for virtualization [EB/OL]. https://www. xenproject. org/, 2017-08-01.

[16] Avanzini A, Valente P, Faggioli D, et al. Integrating Linux and the real-time ERIKA OS through the Xen hypervisor [A]. IEEE International Symposium on Industrial Embedded Systems [C]. Siegen: IEEE Press, 2015. 1 – 7.

[17] Yu P, Xia M, Lin Q, et al. Real-time enhancement for Xen hypervisor [A]. International Conference on Embedded and Ubiquitous Computing (EUC) [C]. Hong Kong: IEEE, 2010. 23 – 30.

[18] Chisnall D. The Definitive Guide to the Xen Hypervisor [M]. Pearson Education, 2008. 217 – 233.

[19] 王吉,包卫东,朱晓敏. 虚拟化云平台中实时任务容错调度算法研究[J]. 通信学报,2014,35(10):171 – 180.
Wang Ji, Bao Weidong, Zhu Xiaomin. Fault-tolerant scheduling algorithm for real-time tasks in virtualized cloud [J]. Journal on Communications, 2014, 35(10): 171 – 180. (in Chinese)

[20] Takada H, Iiyama S, Kindaichi T, et al. " Li-nux on ITRON" : a hybrid operating system architecture for embedded systems [A]. Symposium on Applications and the Internet Workshops [C]. Nara: IEEE Press, 2002. 4 – 7.

[21] 吴一民. RT-Linux 的实时机制分析[J]. 计算机应用,2002,22(12):110 – 112.
Wu Yimin. Real-time mechanism analysis of RT-Linux [J]. Computer Application, 2002, 22(12): 110 – 112. (in Chinese)

[22] Mantegazza P, Dozio L, Papacharalambous S. RTAI: Real time application interface [J]. Linux Journal, 2000 (72es): 10.

[23] Gerum P. Xenomai: Real-time framework forLinux [EB/OL]. http://xenomai. org/, 2017-08-01.

[24] Heiser G. The role of virtualization in embedded systems [A]. Proceedings of the 1st Workshop on Isolation and Integration in Embedded Systems [C]. Glasgow, Scotland: ACM, 2008. 11 – 16.

[25] Hätig H, Hohmuth H, Liedtke J. The performance of μ-kernel-based systems [A]. ACM SIGOPS Operating System Review [C]. Malo, France: ACM, 1997. 31(5): 66 – 77.

[26] Tu Dresden. The Fiasco microkernel [EB/OL]. http://os. inf. tu-dresden. de/fiasco/, 2016-06-09.

[27] Bruns F, Traboulsi S, Szczesny D, et al. An evaluation of microkernel-based virtualization for embedded real-time systems [A]. Euromicro Conference on Real-Time Sys-

tems (ECRTS) [C]. Brussels: IEEE Press, 2010. 57 – 65.

[28] Wikipedia. Adaptive Domain Environment forOperating Systems [EB/OL]. https://en. wikipedia. org/wiki/Adaptive_Domain_Environment_for_Operating_Systems, 2017-03-15.

[29] Surhone L M, Tennoe M T, Henssonow S F. Adaptive Domain Environment for Operating Systems [M]. Betascript Publishing, 2010. 2 – 10.

[30] Roy K S, Gowthami K. Implementation of Xenomai Framework in GNU/Linux Environment to run applications in a Real-Time Environment [J]. Indian Journal of Science and Technology, 2016, 9(17): 1 – 8.

[31] Wikipedia. RT PREEMPT HOWTO [EB/OL]. https://rt. wiki. kernel. org/index. php/RT _ PREEMPT _ HOWTO, 2017-06-17.

[32] Sabt M, Achemlal M, Bouabdallah A. Trusted execution environment: What it is, and what it is not [A]. Trustcom/bigdatase/ispa [C]. Helsinki, Finland: IEEE, 2015. 57 – 64.

[33] Varanasi P, Heiser G. Hardware-supported vi-rtualization on ARM [A]. Asia-Pacific Workshop on Systems [C]. Shanghai: ACM, 2011. 1 – 5.

[34] Santos N, Raj H, Saroiu S, et al. Using ARM TrustZone to build a trusted language runtime for mobile applications [A]. ACM SIGARCH Computer Architecture News [C]. Salt Lake City: ACM, 2014. 67 – 80.

[35] Oh S C, Koh K W, Kim C Y, et al. Acceleration of dual OS virtualization in embededsystems [A]. International Conference on Computing and Convergence Technology [C]. Seoul, Korea: IEEE, 2012. 1098 – 1101.

[36] Sangorrin D, Honda S, Takada H. Dual ope-rating system architecture for real-time embedded systems [A]. Proceedings of the 6th International Workshop on Operating Systems Platforms for Embedded Real-Time Applicati-ons [C]. Brussels, Belgium, 2010. 6 – 15.

[37] Sangorrín D, Honda S, Takada H. Reliable and efficient Dual-OS communications for real-time embedded virtualization [J]. Information and Media Technologies, 2013, 8(1): 1 – 17.

[38] Shah J H. ARMithril: A secure OS leveraging ARM´s TrustZone technology [J]. Macromolecules, 2007, 40(20): 7075 – 7078.

[39] Jing L, Chunhua J, Xia Y. Design and implementation of security OS based on TrustZone [A]. International Conference on Electronic Measurement & Instruments [C]. Harbin, China: IEEE, 2013. 1027 – 1032.

[40] Pinto S, Tavares A, Montenegro S. Hypervis-or for teal time space applications [A]. 4SSymposium [C]. Malta: 4S Symposium, 2016. 1 – 14.

[41] Pinto S, Pereira J, Gomes T, et al. Towards a TrustZone-assisted hypervisor for real-time embedded systems [J]. IEEE Computer Architecture Letters, 2016, 16(2): 158 – 161.

[42] Sangorrin Lopez D. Advanced integration techniques for highly reliable dual-OS embedded systems [D]. Japan: Nagoya University, 2012. 1 – 114.

[43] Sangorrin D, Honda S, Takada H. Integrated scheduling for a reliable dualos monitor [J]. Information and Media Technologies, 2012, 7(2): 627 – 638.

[44] Aalto A. Dynamic management of multiple operating systems in an embedded multi-core environment [D]. Espoo: Aalto University, 2010. 1 – 80.

[45] Gilles K, Groesbrink S, Baldin D, et al. Prot-eus hypervisor: Full virtualization and paravirtualization for multi-core embedded systems [A]. International Embedded Systems Symposium [C]. Berlin, Heidelberg: Springer, 2013. 293 – 305.

[46] Kinebuchi Y, Mitake H, Yasukawa Y, et al. A study on real-time responsiveness on virtualization based multi-OS embedded systems [A]. Proceedings of the 1st International Conference on Pervasive and Embedded Computing and Communication Systems [C]. Vilamoura, Portugal: PECCS, 2011. 369 – 378.

[47] Yu Q, Wei H, Liu M, et al. A novel multi-OS architecture for robot application [A]. Robotics and Biomimetics (ROBIO) [C]. Karon Beach, Phuket, Thailand: IEEE, 2011. 2301 – 2306.

[48] Meier L, Honegger D, Pollefeys M. PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms [A]. 2015 IEEE International Conference on Robotics and Automation (ICRA) [C]. Seattle, USA: IEEE, 2015. 6235 – 6240.

[49] 杨海钢, 孙嘉斌, 王慰. FPGA 器件设计技术发展综述 [J]. 电子与信息学报, 2010, 32(3): 714 – 727.
Yang Haigang, Sun Jiabin, Wang Wei. An overview to FPGA device design technologi-es [J]. Journal of Electronics & Information Technology, 2010, 32(3): 714 – 727. (in Chinese)

[50] Hauck S, Dehon A. Reconfigurable Computi-ng: The Theory and Practice of FPGA-Based Computation [M]. Morgan Kaufmann Publishers Inc, 2007.

[51] Wei H, Li C, Chen D, et al. Research on reconfigurable robot controller based on ARM and FPGA [A]. IEEE International Conference on Industrial Informatics [C]. Daejeon, Korea: IEEE, 2008. 123 – 128.

[52] Morabito R, Kjällman J, Komu M. Hypervisors vs. lightweight virtualization: a performance comparison [A].

IEEE International Conference on Cloud Engineering [C]. Tempe, USA: IEEE, 2015. 386 – 393.
[53] Li Y, Li W, Jiang C. A survey of virtual machine system: Current technology and future trends [A]. Third International Symposium on Electronic Commerce and Security [C]. Guangzhou: IEEE, 2010. 332 – 336.
[54] 涂刚,阳富民,胡贯荣,嵌入式操作系统综述[J]. 计算机应用研究,2000,17(11):4 – 5.
Tu Gang, Yang Fumin, Hu Guanrong. An overview of embedded operating system [J]. Application Research of Computers, 2000, 17(11): 4 – 5. (in Chinese)
[55] 叶常春. 嵌入式虚拟化技术[J]. 计算机工程与科学,2012,34(3):41 – 45.
Ye Changchun. The embedded virtualization technology [J]. Computer Engineering & Science, 2012, 34(3): 41 – 45. (in Chinese)
[56] Kim B, Choi M. Design and analysis of m-ultiple OS implementation on a single arm-based embedded platform [J]. Sustainability, 2017, 9(5): 684.
[57] 郑显义,李文,孟丹. TrustZone 技术的分析与研究[J]. 计算机学报,2016(9):1912 – 1928.
Zheng Xianyi, Li Wen, Meng Dan. Analysis and research on TrustZone technology [J]. Chinese Journal of Computers, 2016(9): 1912 – 1928. (in Chinese)
[58] Sangorrín D, Honda S, Takada H. Reliable device sharing mechanisms for dual-os embedded trusted computing [J]. Trust and Trustworthy Computing, 2012: 74 – 91.
[59] Pinto S, Oliveira D, Pereira J, et al. Towardsa lightweight embedded virtualizetion architecture exploiting ARM TrustZone [A]. Emerging Technology and Factory Automation [C]. Barcelona, Spain: IEEE, 2014. 1 – 4.
[60] Jeon M, Kim S, Yoo H. Inter-GuestOS communications in multicore-based ARM TrustZone [J]. Journal of KIISE, 2015, 42(5): 551 – 557.
[61] 杨际祥,谭国真,王荣生. 多核软件的几个关键问题及其研究进展[J]. 电子学报,2010,38(9):2140 – 2146.
Yang Jixiang, Tan Guozhen, Wang Rongsheng. Some key issues and their research progress in multicore software [J]. Acta Electronica Sinica, 2010, 38(9): 2140 – 2146. (in Chinese)
[62] Zaki Y. An embedded multicore platform formixed-criticality systems: Study andanalysis of virtualization techniques [D]. Sweden: School of Information and Communication Technology, 2016.
[63] Borkar S. Thousand core chips: a technologyperspective [A]. Proceedings of the 44th annual Design Automation Conference [C]. San Diego, USA: ACM, 2007. 746 – 749.
[64] Wikipedia. Multiprocessing [EB/OL]. https://en.wikipedia.org/wiki/Multiprocessing. 2017-05-10.
[65] Quigley M, Conley K, Gerkey B, et al. ROS: an open-source robot operating system [A]. ICRA Workshop on Open Source Software [C]. ICRA, 2009, 5 – 11.
[66] Wei H, Shao Z, Huang Z, et al. RT-ROS: A real-time ROS architecture on multi-core processors [J]. Future Generation Computer Systems, 2016, 56: 171 – 178.
[67] Dong Q, Huang Z, Wei H. A hybrid operating system for modular dualarm manipulator [A]. IEEE International Conference on Robotics and Biomimetics (ROBIO) [C]. Bali, Indonesia: IEEE, 2014. 1481 – 1486.
[68] Lal S V, Palaniappan R, Prakash V. Real-time nursing management system for health care industry by using Xenomai kernel [J]. Indian Journal of Science and Technology, 2015, 8(20): 1 – 10.
[69] Santos N, Raj H, Saroiu S, et al. Using ARM TrustZone to build a trusted language runtime for mobile applications [J]. ACM Sigarch Computer Architecture News, 2014, 42(1): 67 – 80.
[70] Hwang J Y, Suh S B, Heo S K, et al. Xenon ARM: system virtualization using xen hypervisor for ARM-based secure mobile phones [A]. IEEE Consumer Communications and Networking Conference [C]. Las Vegas, USA: IEEE Press, 2008. 257 – 261.
[71] Yoo S, Liu Y, Hong C H, et al. MobiVMM: a virtual machine monitor for mobile phones [A]. The Workshop on Virtualization in Mobile Computing [C]. Breckenridge, Colorado: ACM, 2008. 1 – 5.

作者简介

张美玉 女,1994 年生于山东威海. 硕士研究生,研究方向为实时操作系统.

张倩颖(通信作者) 女,1986 年生于河北廊坊. 首都师范大学信息工程学院讲师、硕士生导师. 研究方向为实时操作系统、形式化验证.
E-mail: qyzhang@cnu.edu.cn