doi：10.11830/ISSN.1000-5013.201605023

ARM TrustZone 的轻量级嵌入式虚拟化架构

王亮

（西藏民族大学 信息工程学院，陕西 咸阳 712082）

摘要： 针对现存的基于软件的虚拟化解决方案存在的不足，利用 ARM 标准硬件技术和赛灵思 ZC702 商业平台，实现通用操作系统（GPOS）与轻量级实时操作系统（FreeRTOS）同时运行. 测试结果表明：虚拟机从 RTOS 到 GPOS 的上下文切换系统开销是 3.10 μs，相反过程为 2.64 μs，内存占用也仅为 1 KB；由虚拟机监视系统（VMM）引入的性能开销低和内存占用较小；利用 ARM TrustZone 技术可实现一个具有低成本和高可靠性的轻量级虚拟化解决方案.

关键词： 嵌入式系统；虚拟化技术；实时操作系统；处理器；软件模块

中图分类号： TP 311.52 **文献标志码：** A **文章编号：** 1000-5013(2016)05-0641-04

Towards Lightweight Embedded Virtualization Architecture Exploiting ARM TrustZone

WANG Liang

(School of Information Engineering, Xizang Minzu University, Xianyang 712082, China)

Abstract: According to the existing solution scheme of software based virtualization, the arm standard hardware technology, combined with the mature technology of Xilinx ZC702 business platform, the general-purpose operating system (GPOS) and real-time operating system (FreeRTOS) system running at the same time are realized. Test data shows that the virtual machine from the RTOS to GPOS system context switching overhead is 3.10 μs, opposite is 2.64 μs, memory is only 1 KB; the performance overhead introduced by the virtual machine monitor (VMM) is low and with a smaller memory footprint; ARM TrustZone technology is exploited to implement a lightweight virtualization solution with low overhead and high determinism.

Keywords: embedded systems; virtualization; real-time operating system; processor; software module

虚拟化技术可以使得同一处理器并行执行多个虚拟机（virtual machine，VM），并允许多个操作系统（operating system，OS）并存于同一硬件平台[1]. 虚拟化技术多被用在企业和云计算空间，最大限度地提高资源的可用性[2]. 随着新型嵌入式设备的大量涌现，对系统技术要求也日益提高，系统不仅应具备通用计算能力和功能扩展能力，而且还要满足对实时性和可靠性的技术要求. 因此，嵌入式虚拟化技术应运而生. 典型的嵌入式虚拟化解决方案[3-8]本质上遵循两种不同的技术实现方式：全虚拟化和超虚拟化[9]. 对于全虚拟化，客户操作系统不需要作任何的修改，管理程序或者虚拟机监控程序（virtual machine monitor，VMM）需要作出捕捉指令，协调硬件执行操作. 因此，全虚拟化性能比使用裸机硬件慢. 对于超虚拟化，管理程序实现了操作系统对底层硬件的共享访问，并将虚拟化有关的代码集中在操作系

收稿日期： 2016-06-20

通信作者： 王亮（1968-），男，副教授，博士，主要从事计算机网络工程的研究. E-mail：wzjwlwl@163.com.

基金项目： 西藏科技厅科研项目（2015ZR-14-20）

统本身,因此,可直接从管理程序请求服务.这种静态方法,执行性能明显提高,但系统自定义修改难度较大,研发成本较高、研发周期较长和后期维护难度较大.目前,嵌入式领域关注重点为硬性限制,研究重点也集中在基于硬件的辅助虚拟化技术,开发具有高性能的嵌入式虚拟化解决方案[10-14].较成熟的嵌入式虚拟化案例是采用 Intel,ARM,AMD 等硬件厂商的虚拟化技术配合 ARM TrustZone 和 Intel TxT(trusted execution technology)技术.本文提出一种基于 Trust Zone 的虚拟化架构实现方案,它允许通用操作系统(GPOS,Linux)与实时操作系统(RTOS,FreeRTOS)并行运行,并基于标准硬件搭建系统平台,使得性能功耗和内存占用比较小.

1 ARM TrustZone

支持 TrustZone 技术的 ARM 处理器型号主要包括 ARM1176,Cortex-A5/A7/A8/A9/A15 及最新的 64 位 Cortex-A53/A57.该技术基于硬件支持的安全性扩展,利用虚拟化技术,将一个物理核心虚拟成两个虚拟内核,并提供两个完全独立的执行环境:安全可信任的安全环境和非安全的普通执行环境.TrustZone 通过启用或禁用协处理器 CP15 的安全配置寄存器(secure configuration register,SCR)的 NS 位转换处理器的执行状态.当 NS=0 时,处于安全状态;当 NS=1 时,处于非安全状态.为实现处理器在安全状态和非安全状态的转换,引入监控模式.监控模式被用来控制系统的安全状态和数据的访问权限,负责保存当前上下文状态.进入监控模式,需要一种新的权限指令,即安全调用(secure monitor call,SMC).硬件层面,TrustZone 技术从 CPU 内核开始设置系统安全,以确保普通环境与安全环境之间的完全隔离.

2 TrustZone 虚拟化架构

Frenzel 等[15]将 TrustZone 技术应用于嵌入式系统虚拟化,认为 TrustZone 技术能够提供一个基于可信硬件专用的系统虚拟化架构,尤其针对具有两个虚拟机的情况.由于虚拟机数量与处理器支持的独立状态完全对应,使得多媒体操作系统(如 Linux,Android)可在非安全环境上运行,而安全关键软件则在安全环境运行.同时,监视模式可对处理器进行全面观测,执行 VMM 不再需要修改驻留在非安全环境的操作系统.但是,尽管基于特权模式运行,GPOS 仍不能直接访问安全环境资源,其权限仍然要比 VMM 组件低.TrustZone 技术使得系统开销降低,并加快虚拟机之间的上下文切换.

2.1 体系结构描述

基于 TrustZone 技术的嵌入式虚拟化架构,如图 1 所示.图 1 中:RTOS (FreeRTOS),GPOS (Linux)和 VMM 为 3 个主要的软件.首先,GPOS 运行于非安全的普通执行环境,保证良好的人机交互界面及基于互联网的应用和服务;其次,RTOS 运行于安全的执行环境,为系统软件应用提供安全实时保证;最后,VMM 组件以监控模式运行于安全的执行环境,负责管理每个虚拟机的虚拟机控制块(virtual machine control block,VMCB).当虚拟机由物理处理器来执行时,VMM 在对应 VMCB 中保存虚拟处理器的当前状态,从 VMCB 中恢复与当前 VM 相应的处理器状态.

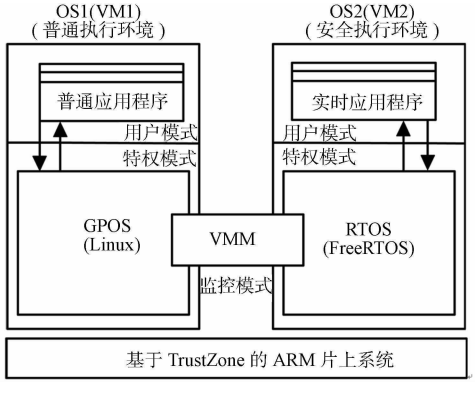

图 1 嵌入式虚拟化架构

Fig. 1 Embedded virtualization architecture

2.2 执行流程

系统在安全环境下,由引导程序启动,该程序主要负责初始化硬件设备,建立内存空间的映射图、为最终操作系统内核的运行准备好系统的软硬件环境.中断控制器(generic interrupt controller,GIC)在安全环境下,调用快速中断指令(fast interrupt requests,FIQ),在非安全环境下,调用普通中断指令(interrupt requests,IRQ).执行流程图,如图 2 所示.通过禁用 SCR 寄存器的 FIQ 和 IRQ 位,保证异常 FIQ/IRQ 不会造成监视模式的切换,并使得 VM 切

换仅调用 SMC 指令. 引导程序执行后，RTOS 启动，并开始调度系统程序. 当实时任务被中断或挂起，空闲任务执行一个系统调用，用来唤醒 VMM 和执行 SMC 指令. 同时，处理器进入到监视模式，并开始执行 VMM，跳转至监控模式向量表的特定处理程序. 因此，为切换至非安全环境，处理器执行 SMC 处理程序.

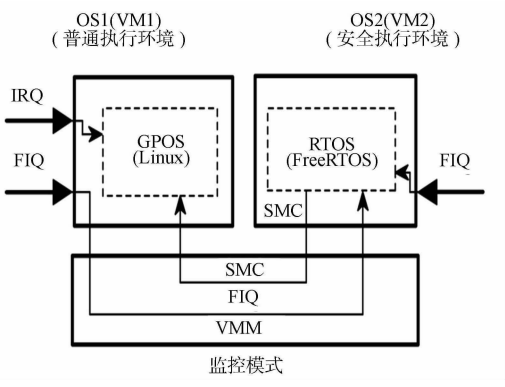

图 2　执行流程图

Fig. 2　Execution flow chart

完成上述执行流程后，系统进行上下文切换操作. 实际上，安全环境的处理器状态保存于相对应的 VMCB 中，从非安全环境的 VMCB(Linux)中还原. 但是，程序的首次执行是个特例，出于最优化的目的，只对安全环境的处理器状态进行保存. 设置特权模式(supervisor mode，SVC)，并更新链接寄存器的 GPOS 内核的启动地址. 然后，通过 VMM 调用 FIQ 指令，对 SCR 寄存器的 NS 位使能，并跳转至非安全环境状态的初始化或还原地址. 值得注意的是，缓存中一直都没有进行操作. TrustZone 允许安全环境和一般环境的缓存条目共存，并支持在缓存控制器指定一个 NS 标记位关联缓存中所有数据. 这种方法在虚拟机之间切换时不需对高速缓存进行刷新，并有助于加快上下文切换.

由图 2 可知：在非安全环境，当 FIQ 得到触发时，GPOS 开始启动运行. 由于 SCR 寄存器的 FIQ 位已经启用，FIQ 触发使处理器进入监控模式，跳转至监控模式向量表的 FIQ 处理程序. 同时，VMM 开始执行，并准备上下文切换. 首先，通过禁用 SCR 寄存器的 FIQ 和 NS 位，保存非安全环境的处理器状态于相对应的 VMCB 中，确认 FIQ 请求，并从 VMCB 还原安全环境上下文. 其次，处理器切换回 RTOS 内核，调度任务再次启动. 处理器将一直运行于安全环境，直到空闲任务执行新的系统调用，处理器重新执行所有先前描述的执行流程.

3　测试分析

基于赛灵思(Xilinx)ZC702 对虚拟化架构运行测试，搭载功能强大的双核 ARM Cortex-A9 处理器，运行频率为 800 MHz，支持多核硬件架构，使用单个芯片即可构建高性能片上系统. 系统测试内容主要包括系统性能和内存占用，系统性能通过性能监视单元(performance monitoring unit，PMU)获得，内存占用则是利用赛灵思工具链获得.

为了评估由 VMM 执行产生的系统开销，进行两次上下文切换操作.

1) 切换到非安全环境. 从安全环境切换到非安全环境，VMM 执行上下文切换操作，将控制权交给操作系统 Linux 内核. 时钟周期的计数是从安全环境下的 SMC 指令调用时刻至处理器访问非安全环境的地址瞬间.

2) 切换到安全环境. 从非安全环境切换到安全环境，VMM 执行上下文切换操作，将控制权交给 FreeRTOS 内核. 时钟周期的计数是从非安全环境的 FIQ 异常的确切时刻至处理器在安全环境下运行的时刻.

每个试验重复 20 次，测试结果包括最小值、最大值、平均值及测量值的标准偏差. 就 Cortex-A9 处理器而言，尽管 VMM 在每次执行的指令相同，但由于体系结构的动态特征，执行指令的时钟周期的次数是变化的.

VMM 运行统计，如表 1 所示. 表 1 中：$t_{\min}$ 为最小值；$t_{\max}$ 为最大值；μ 为平均值；σ 为测量值的标准偏差. 由于所用处理器的时钟频率为 800 MHz，结合测量的平均值进行计算，得到安全环境到非安全环境切换和非完全环境到安全环境切换的平均执行时间分别为 3.10，2.64 μs. 考虑到 FreeRTOS 内进行任务切换所需的时间为 2.02 μs，虚拟机进行上下文切换平均时间仅超出 53.4%和 30.6%. 当 VMM 运行时禁用所有中断源，在安全环境到非安全环境上下文切换时，如果触发 FIQ 请求，意味最差情况下中断执行时间为 6.02 μs.

VMM 内存(n)统计，如表 2 所示. 由表 2 可知：与 RTOS 相较而言，VMM 的内存占用相对较小；

VMM所需的内存分别约为FreeRTOS的1.30%和Linux的0.04%.

表1 VMM运行统计

Tab.1 VMM operating statistic

转换状态	t_{min}/s	t_{max}/s	μ	σ
Switch to NS world	2 431	2 568	2 478	52.5
Switch to S world	2 081	2 245	2 109	48.9

表2 VMM内存统计

Tab.2 VMM memory statistic

系统	n(text)	n(data)	n(bss)	n(Total)
VMM	848	0	244	1 092
FreeRTOS	17 646	16	66 000	83 690
Linux	2 874 978	52	4 120	2 879 150

4 结束语

嵌入式系统的最新发展方向之一就是虚拟化技术.基于TrustZone技术,利用ARM标准硬件技术,结合技术成熟的赛灵思ZC702商业平台,实现了GPOS与FreeRTOS的系统同时运行.虚拟机从RTOS到GPOS的上下文切换系统开销是3.10 μs,相反过程的为2.64 μs,内存占用也仅为1 KB,证明由VMM引入的性能开销低和内存占用较小.当前系统架构具有优越性,但就最优而言,还有亟需解决的问题.下一步研究思路是基于现有结构下针对共享设备集成新的访问机制,摒除每个外设专用于同一执行环境的限制.研究虚拟机数目和系统架构支持的虚拟处理器数量之间的相关性,扩展虚拟化架构以满足多用户支持和多核心支持.基于服务客户端技术,利用TrustZone API建立两个操作系统之间的标准通信机制,使得可信任执行环境与扩展VMM相关联,最终实现针对目标安全的完整框架.

参考文献:

[1] 张国亮,王展妮,王田.应用计算机视觉的动态手势识别综述[J].华侨大学学报(自然科学版),2014,35(6):653-657.

[2] 钟必能,陈雁,沈映菊,等.在线机器学习跟踪算法的研究进展[J].华侨大学学报(自然科学版),2014,35(1):41-45.

[3] HEISER G. The role of virtualization in embedded systems[C]// Proceedings of the 1st Workshop on Isolation and Integration in Embedded Systems. New York: Association for Computing Machinery, 2008: 11-16.

[4] 韦照川,李德明.嵌入式系统发展概述[J].科技信息,2010(1):839.

[5] MASMANO M, RIPOLL I, CRESPO A, et al. Xtratum: A hypervisor for safety critical embedded systems[C]// Proceedings of the 11th Real Time Linux Workshop. Nanjing: Real Time Linux Workshop, 2009: 153-159.

[6] STEINBERG U, KAUER B. NOVA: A microhypervisor based secure virtualization architecture[C]// Proceedings of the 5th European Conference on Computer Systems. Paris: EuroSys, 2010: 209-222.

[7] 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.

[8] 叶存奎.USB设备协议栈的设计与实现[D].武汉:华中科技大学,2011:1-60.

[9] 周亦敏,隋伟鑫.ARM架构中TrustZone安全处理技术的研究[J].微计算机信息,2009,24(36):69-71.

[10] VARANASI P, HEISER G. Hardware supported virtualization on ARM[C]// Proceedings of the Second Asia Pacic Workshop on Systems. Tokyo: User Evaluation, 2011: 231-236.

[11] HERLIHY M, SHAVIT N. 多处理器编程的艺术[M].金海,译.北京:机械工业出版社,2009:218-220.

[12] 林小茶,李光.基于嵌入式技术的信任根研究[J].计算机工程与应用,2007,43(16):165-168.

[13] 任爱芝.基于ARM的可信嵌入式系统设计[J].电脑编程技巧与维护,2011(20):24-25.

[14] ALVES T, FELTON D. TrustZone: Integrated hardware and software security[J]. ARM White Paper, 2004, 3(4): 18-24.

[15] FRENZEL T, LACLPRZUMSLI A, WARG A, et al. ARM TrustZone as a virtualization technique in embedded systems[J]. Twelfth Real Time Linux Workshop, 2010(6): 76-79.

(**责任编辑**:陈志贤 **英文审校**:吴逢铁)