УДК 004.318

Средства аппаратной виртуализации в системе команд микропроцессоров «Эльбрус»

Д.В. Знаменский, А.В. Блинников

АО «МЦСТ»

Требования к производительности и масштабируемости виртуализованных систем стремительно возрастают. Мировой опыт показывает, что для развития вычислительной платформы в этом направлении необходима аппаратная виртуализация. Поддержка аппаратной виртуализации была добавлена в систему команд (СК) микропроцессоров «Эльбрус», начиная с версии 6.

Анализ существующих подходов к виртуализации различных архитектур, включающих программную виртуализацию [1], паравиртуализацию [1] и аппаратно поддержанную полную виртуализацию [2][3][4][5], позволил сформулировать ключевые требования к аппаратной поддержке виртуализации в СК «Эльбрус»: наличие изолированных режимов исполнения виртуальной машины (ВМ) и гипервизора с атомарным переключением между ними, добавление средств управления исполнением ВМ, средств виртуализации памяти ВМ и внешних устройств. Требуемая функциональность была реализована в виде набора расширений СК, учитывающих специфику архитектуры «Эльбрус».

Введено разделение на режимы гостя и гипервизора, ортогональное существующим пользовательскому и привилегированному режимам. Режим гипервизора включает набор средств управления гостевым режимом: команды входа в гостевой режим, управляющие и статусные регистры виртуализации, идентификатор гостя GID, набор регистров теневого контекста с копиями дескрипторов аппаратных стеков, состояния MMU и др. Эти средства позволяют гипервизору прозрачным для гостя способом осуществлять вмешательство в исполнение ВМ посредством перехвата «опасных» команд и событий, управлять физической памятью и прерываниями ВМ.

Предусмотрено два механизма переключения между режимами гостя и гипервизора:

- Переходы glaunch и перехваты - для поддержки полной виртуализации.
- Пара команд hcall-hret, реализующая функциональность гипервызовов – для поддержки паравиртуализации.

Атомарность переключения архитектурного состояния машины в обоих механизмах обеспечивается аппаратно. В отличие от реализаций Intel VMX [2] и AMD SVM [3], переключение выполняется без использования специально введённой области системной памяти. Для переключения аппаратных стеков используется штатный spill-fill механизм; остальная часть контекста снабжена так называемой теневой копией, активируемой при переходе в противоположный режим. Теневая копия контекста гостя доступна гипервизору напрямую.

Гипервизор снабжён широким набором настроек перехвата гостевого исполнения, в числе которых доступ к управляющему контексту, особые ситуации, операции очистки TLB и кэш-памятей и др. Также режим гипервизора имеет исчерпывающую информацию о перехваченном событии. В зависимости от ситуации, гипервизор имеет возможность виртуализовать перехваченное гостевое событие (например, подменив результат гостевой операции или выполнив её эмуляцию), инжектировать или изъять особую ситуацию или аварийно остановить исполнение гостя.

Для виртуализации памяти ВМ в архитектуру «Эльбрус» внедрены два подхода:

1. Поддержка теневых таблиц страниц (ТС).
2. Введение двух уровней (этапов) трансляции адреса.

Первый подход (использование теневых ТС, или virtual TLB [2]) позволяет использовать для управления гостевой виртуальной памятью единственную видимую аппаратуре, но невидимую гостю таблицу (называемую теневой ТС). Гостевая ТС при этом не управляет трансляцией виртуальных адресов напрямую, но её содержимое переносится гипервизором в теневую ТС посредством перехвата доступа к регистрам MMU, ошибок трансляции адреса (page miss) и команд очистки TLB. Помимо поддержки необходимой для теневой ТС функциональности, в архитектуру «Эльбрус» введён механизм трансляции гостевых физических адресов посредством таблицы Guest Physical Page Table, которую формирует гипервизор. Это позволяет, например, виртуализовать физическую память ВМ на время загрузки гостевой ОС, то есть до тех пор, пока последняя не перешла в режим виртуальной адресации и не создала собственной ТС.

Второй подход, также известный как EPT [2], Nested Paging [3] или Two-Stage Address Translation [4], предполагает наличие двух независимых этапов трансляции адреса: из гостевого виртуального адреса (VA) в гостевой физический адрес (GPA), и из GPA – в физический адрес хоста

(HPA). Каждому этапу трансляции соответствует собственная ТС, причём гость управляет трансляцией VA → GPA без вмешательства гипервизора; гипервизор же управляет только уровнем трансляции GPA → HPA. Этот подход требует поддержки на уровне СК, которая была реализована в версии 6.

Использование теневой ТС является более простым с точки зрения аппаратуры и имеет меньшую среднюю задержку поиска по единственной видимой аппаратуре ТС. Однако этот подход усложняет гипервизор и требует частой работы через перехват, с соответствующими накладными расходами. Двухэтапная трансляция, при большей аппаратной сложности, увеличивает производительность за счёт невмешательства гипервизора в управление гостевой виртуальной памятью. Стоит отметить, что, для линейной многоуровневой ТС физический адрес, формируемый на каждом уровне гостевой ТС, требует трансляции через ТС гипервизора, что сильно увеличивает итоговую длительность поиска по ТС при промахе в TLB. Для компенсации этого эффекта в процессорах с СК версии 6 добавлены дополнительные кэши устройства TLU для хранения промежуточных уровней трансляции.

Представленный набор расширений СК позволил разработать системы виртуализации для архитектуры «Эльбрус» на основе нативного ядра linux, как в паравиртуализованном варианте, так и в схеме с полной аппаратной виртуализацией на базе KVM+QEMU, с возможностью запуска гостевых ОС в кодах «Эльбрус» или в кодах x86 с использованием технологии двоичной трансляции.

Литература

1. VMWare® Understanding Full Virtualization, Paravirtualization, and Hardware Assist, p. 4. March 11, 2008. Availabe at: https://www.vmware.com/techpapers/2007/understanding-full-virtualization-paravirtualizat-1008.html
2. Intel® 64 and IA-32 Architectures Software Developer’s Manual Volume 3C: System Programming Guide, Part 3, Order Number: 326019-060US, September 2016. Available at: https://www.intel.ru/content/dam/www/public/us/en/documents/manuals/64-ia-32-architectures-software-developer-vol-3c-part-3-manual.pdf
3. AMD64® Architecture Programmer`s Manual Volume 2: System Progamming, rev.3.33, AMD Corp., 2020. Available at: https://www.amd.com/system/files/TechDocs/24593.pdf
4. Arm® Instruction Set Version 1.0 Reference Guide, Issue 0100-00, 25 October 2018. Available at: https://static.docs.arm.com/100076/0100/arm_instruction_set_reference_guide_100076_0100_00_en.pdf
5. MIPS64® Architecture for Programmers Volume IV-i: Virtualization Module of the MIPS64® Architecture, Document Number: MD00847 Revision 1.06, December 10, 2013. Available at: httpsвав://www.mips.com/downloads/virtualization-module-of-the-mips64-architecture/