УДК 004.4

Оптимизация сценариев программного моделирования, использующих команду чтения времени RDTSC

В.В. Конычев[1,2], Е.А. Юлюгин[1,2], Г.С. Речистов[1,2]

[1]Московский физико-технический институт (государственный университет),

[2]АО «Интел А/О»

Существуют алгоритмы и задачи, требующие точного определения времени, такие как планировщики событий в операционных системах [1], что необходимо учитывать при разработке виртуальных машин (далее ВМ), так как они должны обеспечивать монотонность и согласованность устройств для измерения времени.

Данная задача не является тривиальной из-за многих причин: многообразия существующих устройств определения времени, особенностей современных процессорных архитектур и специфики виртуальной машины, для которой данная задача решается.

В работе рассматриваются проблемы и существующие методы решения задачи виртуализации устройства Time Stamp Counter (далее TSC), а также предлагается реализация нового метода и доказательство его практических преимуществ.

Исследование проводилось на полноплатформенном симуляторе Wind River® Simics 4.8 [2] (далее Simics). Для проведения тестов использовался стандартный функционал Simics, для обработки и визуализации использовались Python [3] и Gnuplot [4].

Simics может осуществлять как программную, так и аппаратную симуляцию. В первом случае он работает как пользовательское приложение. Во втором, работает как монитор виртуальных машин, используя технологию Intel VT-x [5]. ВМ исполняются в режиме VMX non-root, где поведение команд ограничено и изменено, чтобы удовлетворить достаточным условиям построения монитора ВМ [6]. Например, исполнение команды RDTSC (Read Time Stamp Counter) зависит от контрольных бит “RDTSC exiting” и “use TSC offsetting” [5]:

- если оба бита нулевые, то происходит обычное исполнение RDTSC,
- если “RDTSC exiting” = 0 и “use TSC offsetting” = 1, то после исполнения инструкции RDTSC в пару регистров EAX:EDX будет записана сумма регистра IA32_TIME_STAMP_COUNTER и поля TSC offset структуры VMCS [5],
- если “RDTSC exiting” = 1, то при исполнении команды RDTSC произойдёт выход в монитор ВМ.

На основе этого в работе предложен следующий метод виртуализации RDTSC. Перед переходом из программной модели в монитор ВМ (рис. 1) вычисляется время до

следующего события ΔT. В мониторе ВМ устанавливается Preemtion_Timer = ΔT. В режиме VMX non-root его значение будет уменьшаться, пока не достигнет нуля, после чего происходит переход в монитор ВМ. Вычисляется разница между симуляционным и реальным временем и используется для коррекции времени внутри ВМ. После чего происходит переход из монитора ВМ в VMX non-root − запуск/продолжение симуляции. После возврата в монитор ВМ вычисляется и продвигается симуляционное время. В зависимости от причины выхода происходит переход либо в VMX non-root, либо в программную модель.

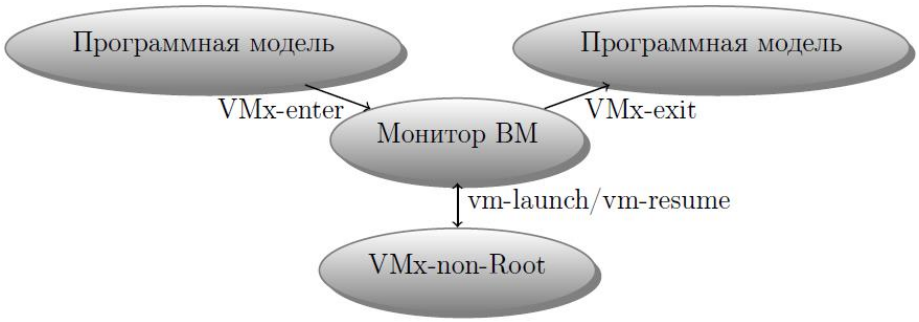

Рис. 1. Принцип работы симулятора

Измерения проводились на разработанном микротесте, в котором активно использовалась RDTSC, и на реальных сценариях, таких как запуск операционных систем Windows, Linux и гипервизора VMware ESX [7].

В результате, на первом этапе мы получили прирост производительности более чем на два порядка по сравнению с программной обработкой вызова RDTSC. Результаты второго этапа представлены на (рис. 2).

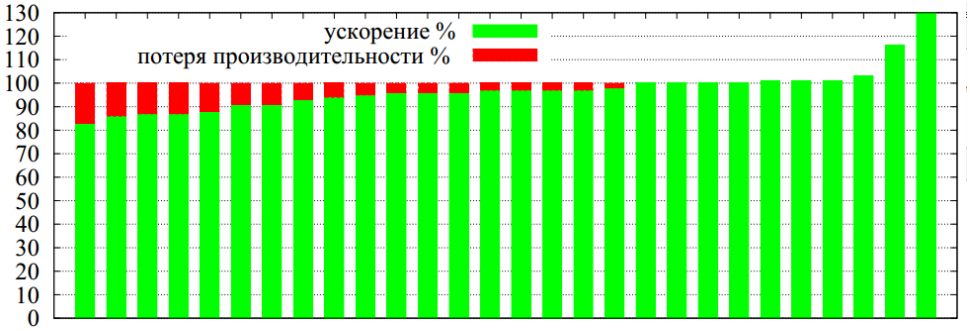

Рис. 2. Результаты второго этапа исследований

В дальнейшем планируется увеличить точность метода и использовать его в сценариях с частым обращением к TSC для улучшения производительности. Также целью будущих исследований станет определение условий, при которых целесообразно применение данного метода.

Литература

1. *Tanenbaum A.S., Bos H.* Modern Operating Systems 4th Edition, March 2014.

2. *Magnusson P.S.* [et. al.]. Simics: A Full System Simulation Platform — Computer Vol. 35, IEEE Computer Society Press, February 2002.

3. *Rossum G., Drake Fred L.* [et. al.]. Python Tutorial. Release 2.7.3 - Python Software Foundation, September 11, 2012.

4. *Williams T, Kelley C.* gnuplot 5.0 An Interactive Plotting Program — http://www.gnuplot.info/docs_5.0/gnuplot.pdf.

5. Intel Corporation. Intel® 64 and IA-32 Architectures Software Developer's Manual Volume 3.

6. *Popek G.J., Goldberg R.P.* Formal requirements for virtualizable third generation architectures, Communications of the ACM. Vol. 17. 1974, P 412-421.

7. VMware. VMware ESX and VMware ESXi. The Market Leading Production-Proven Hypervisors — 2009. — URL: http://www.vmware.com/files/pdf/VMware-ESX-and-VMware-ESXi-DS-EN.pdf.