Для цитирования: Куцевол В.Н., Мешков А.Н., Черных С.В. Методы оптимизации производительности программного моделирования многоядерных микропроцессоров с архитектурой «Эльбрус» // Вопросы радиоэлектроники. 2017. № 3. С. 57–61.
УДК 004.414.23

В.Н. Куцевол[1], А.Н. Мешков[1,2], С.В. Черных[1]

[1] АО «МЦСТ», [2] ПАО «ИНЭУМ им. И.С. Брука»

МЕТОДЫ ОПТИМИЗАЦИИ ПРОИЗВОДИТЕЛЬНОСТИ ПРОГРАММНОГО МОДЕЛИРОВАНИЯ МНОГОЯДЕРНЫХ МИКРОПРОЦЕССОРОВ С АРХИТЕКТУРОЙ «ЭЛЬБРУС»

Рассматриваются подходы к повышению производительности симулятора архитектуры «Эльбрус». Приводится описание работы многопоточного режима работы симулятора, а также трудности его реализации. Предлагается оптимизация путем введения промежуточного представления широкой команды и добавления программного кэша широких команд. Приведены численные показатели эффективности этих решений.

Ключевые слова: *программное моделирование, симулятор, многопоточное выполнение, архитектура Эльбрус.*

Введение

В типичном случае одним из первых этапов разработки вычислительного комплекса (ВК) является создание программной модели, симулятора, запуск которого на инструментальной компьютерной платформе воспроизводит поведение целевой платформы, интерпретируя специфицированные в ней инструкции. Принцип интерпретации относительно прост в реализации, в то же время он не всегда обеспечивает быстродействие модели, достаточное для воспроизведения ряда сценариев функционирования ВК. Поэтому производительность является важной характеристикой, напрямую влияющей на эффективность применения программной модели [1].

Этот фактор приобрел особое значение в проектной деятельности АО «МЦСТ» в связи с переходом к разработке многоядерных микропроцессоров [2]. В данном случае скорость моделирования при работе в одном потоке обратно пропорциональна количеству процессорных ядер в системе, т.к. требуется в режиме интерпретации последовательно воспроизводить работу каждого вычислительного ядра.

Данная проблема усугубляется реализуемой в архитектуре «Эльбрус» парадигмой VLIW – параллельного исполнения операций, представленных в рамках широкого командного слова (в дальнейшем, для простоты, – широкой команды). Последовательное моделирование всех операций, выполняемых аппаратурой параллельно, требует значительного времени, что приводит к дополнительному снижению производительности.

В статье рассматриваются обусловленные этими факторами подходы к оптимизации работы симулятора в однопоточном режиме, а также поддержка многопоточного режима исполнения.

Принципы работы интерпретирующего симулятора

За единицу времени модели, тик (tick), интерпретируется исполнение каждым процессорным ядром всех операций выделенной ему широкой команды. Исполнение операции включает все стадии конвейера: выборку, декодирование, чтение операндов, исполнение, запись результата и продвижение счетчика инструкций. После моделирования исполнения команд процессорными ядрами симулятор обновляет состояния моделей периферийных устройств (рис. 1).

При параллельном исполнении нескольких операций каждая стадия их синхронных конвейеров моделируется последовательным ее выполнением для каждой из операций [3, 4, 5]. Важным следствием такого подхода является воспроизводимость работы симулятора – при заданных параметрах запуска и исполнения программы поведение системы должно быть строго детерминировано, т.е. обеспечивается повторяемость ее работы. Это обусловлено тем фактом, что выполнение атомарных действий происходит в определенной последовательности, а т.к. атомарные действия, в свою

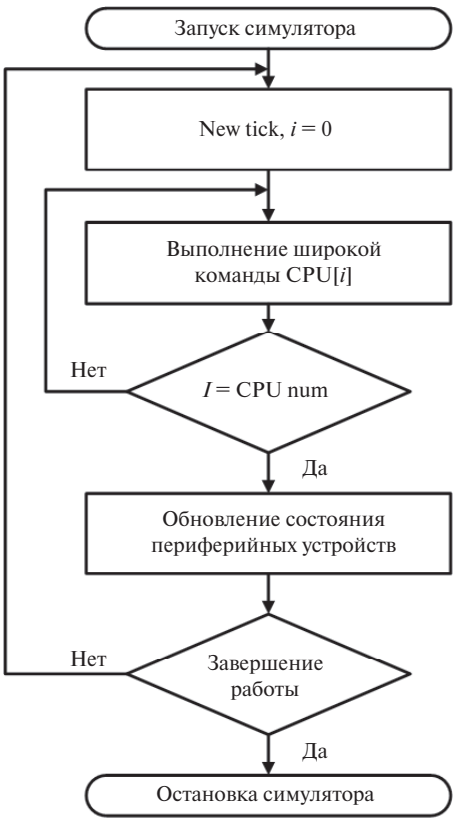

Рисунок 1. Цикл работы однопоточного симулятора

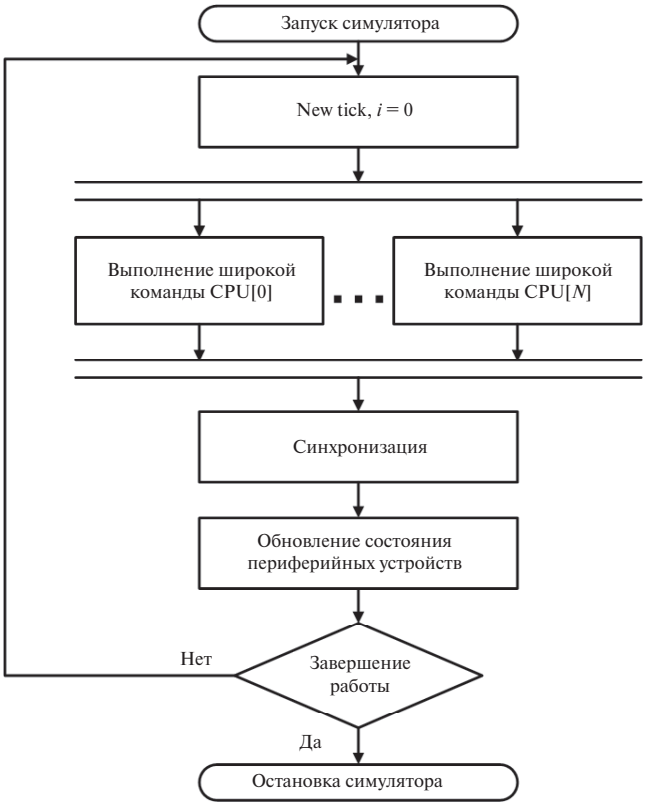

Рисунок 2. Параллельное исполнение широких команд на разных ядрах

очередь, детерминированы, работа всей системы воспроизводима.

Многопоточное моделирование многоядерных задач

Увеличение числа процессорных ядер в ВК приводит к кратному уменьшению скорости моделирования потока исполнения команд для каждого ядра при сохранении количества широких команд, моделируемых всеми ядрами за единицу времени. Это может привести к снижению эффективности моделирования работы операционной системы, выполняющей на одном ядре интересующую пользователя симулятора задачу, а на остальных – фоновые процессы. Подобный эффект может быть компенсирован применением многопоточного моделирования на многоядерной инструментальной машине, которое наиболее эффективно, если количество процессорных ядер моделируемой машины не превышает количества ядер инструментальной машины. В данном случае в каждом потоке будет моделироваться работа только одного процессорного ядра.

Необходимость обеспечения детерминизма работы симулятора при многопоточном моделировании требует сохранить порядок обращений к разделяемым ресурсам, таким как память, регистры северного и южного мостов. При нарушении порядка обращений необходимо повторно выполнить вызвавший нарушение участок программы в однопоточном режиме. Ввиду сложной реализации такого подхода было принято решение синхронизировать работу потоков в конце каждого тика, чтобы не допустить потенциального переупорядочивания инструкций относительно однопоточного исполнения (рис. 2).

При упрощенном способе многопоточного моделирования счетчик модельного времени остается общим для всех потоков, а сохранение порядка обращений к разделяемым ресурсам сводится к запрету на одновременный доступ. Недостатком выбранного способа является высокий расход ресурсов на синхронизацию между потоками, выполняемую в каждом такте. Следствием является то, что повышение производительности задачи на четырех ядрах при моделировании в четыре потока не превышает 45%.

Оптимизация выполнения широкой команды

Модуль, отвечающий за моделирование инструкций, представляет собой интерпретатор двоичного представления широких команд архитектуры «Эльбрус». Ввиду того, что прямая интерпретация снижает скорость работы модели, целесообразно использовать метод предекодовой интерпретации, суть которого заключается в трансляции широкой

команды, представленной в двоичном коде, в промежуточное представление, которое можно быстрее интерпретировать.

Промежуточное представление создается при первой интерпретации широкой команды, расположенной по определенному адресу. Оно сохраняется в специальном программном кэше (ICache) и используется при повторном исполнении широкой команды. Это позволяет увеличивать скорость интерпретации при повторных проходах исходного кода [3]. Алгоритм оптимизированного выполнения широкой команды представлен на рис. 3.

Промежуточное представление широкой команды

Чтобы исключить необходимость в написании отдельного интерпретатора промежуточных представлений, их можно задавать в виде сущностей языка, на котором реализован симулятор. Результатом создания промежуточного представления является выполнение выборки и декодирования только при первом исполнении текущей широкой команды.

Промежуточное представление широкой команды в симуляторе формируется как вектор из операций, закодированных в широкой команде (рис. 4). Каждая операция задается в виде структуры, которая включает в себя функции, реализующие обобщенные стадии R (чтение) и E (выполнение), и контекст выполнения операции – данные, передаваемые между стадиями.

Обобщенная стадия R состоит из действий, не изменяющих состояние моделируемой системы, – вносимые этой стадией изменения локализованы в контексте выполнения операции. Обобщенная стадия E изменяет состояние системы, используя данные, сформированные стадией R. Разделение алгоритмических действий на две стадии вызвано необходимостью моделировать исключительные ситуации. Широкая команда, вызвавшая исключительную ситуацию, не должна вносить изменения в состояние системы, поэтому при фиксации исключительной ситуации во время выполнения стадии R стадия E блокируется.

Программный кэш широких команд

Программный кэш широких команд (ICache) – это ассоциативный контейнер, предназначенный для хранения интерпретированных команд и доступа к ним по виртуальному адресу команды.

Реализация ICache должна обеспечивать высокий процент попаданий, высокую скорость доступа к элементу и возможность вытеснения элементов. С учетом этих требований были рассмотрены следующие альтернативы: упорядоченные ассоциативные массивы на основе бинарного дерева и неупорядоченные ассоциативные массивы на основе хэш-таблиц.

Сложность доступа к элементу сбалансированного бинарного дерева составляет $O(\log(n))$, где n – количество элементов в контейнере. Для оценки производительности данного типа контейнеров в симуляторе был использован упорядоченный ассоциативный контейнер из стандартной библиотеки.

Сложность доступа к элементу хэш-таблицы, определяемая константным временем вычисления хэш-функции и временем разрешения хэш-конфликта, может быть оценена как $O(n)$, где n – ассоциативность хэш-таблицы. В симуляторе для решения конфликтов был применен метод цепочек с переупорядочиванием (рис. 5), а хэш-функция реализована следующим образом: $INDEX = ADDR\%(SIZE\text{-}1)$.

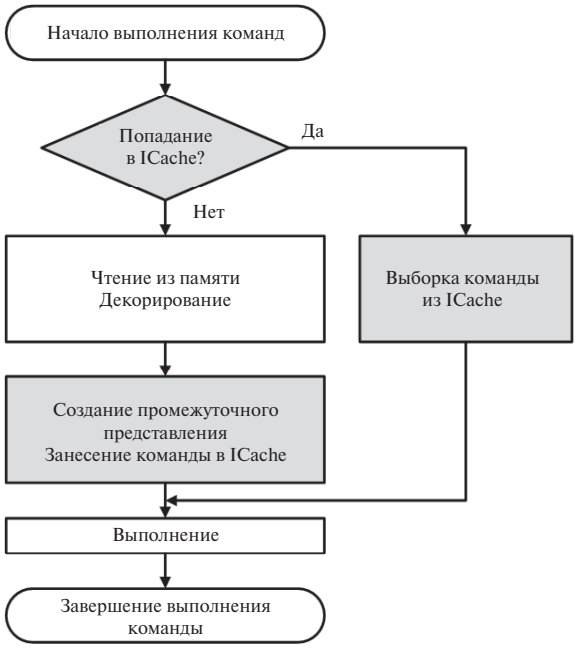

Рисунок 3. Алгоритм оптимизированного выполнения широкой команды

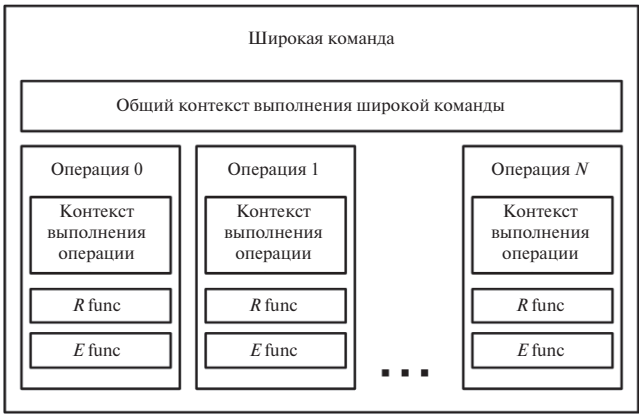

Рисунок 4. Промежуточное представление широкой команды «Эльбрус»

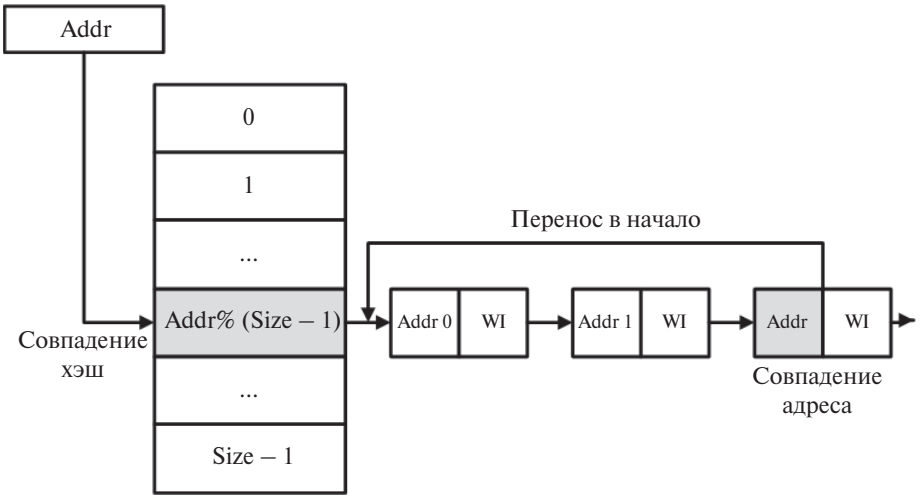

Рисунок 5. Организация ICache

Принцип переупорядочивания заключается в помещении новых элементов и элементов, в которые попадает запрос, в начало цепочки. Если длина цепочки превышает лимит, последние элементы отбрасываются. Хэш-функция выбирает младшие разряды адреса команды, обеспечивая высокий процент попаданий в первый элемент отсортированных цепочек в соответствии с принципом локальности кода.

В результате эксперимента были выбраны следующие параметры ICache: размер таблицы – 2048, максимальная длина цепочки – 15. При таких параметрах средний процент попаданий в ICache составил 99%, а в первый элемент цепочки – 85%. Среднее время доступа к элементу близко к константному.

Результаты практического исследования, подробно описанного в [6], совпали с теоретическими выкладками, и для реализации ICache была выбрана хэш-таблица.

В таблице представлено относительное время, затрачиваемое на выборку и декодирование команд на прогонах реальных задач: загрузка операционной системы Linux, загрузка операционной системы Linux с использованием бинарной трансляции, задача 403.gcc из SPEC CPU2006. В результате среднее время выборки и декодирования команд уменьшилось с 50 до 7% от общего времени исполнения.

Таблица. Время, затрачиваемое на выборку и декодирование команд

Задача	Время выборки (%)	
	ICache off	ICache on
Linux	52	9
Linux+Lintel	50	8
403.gcc	49	3,5

Заключение

В ходе данной работы проводился анализ проблем, наиболее влияющих на скорость работы симулятора ВК с архитектурой «Эльбрус», и была предпринята попытка применить ряд подходов и оптимизаций, позволивших повысить его производительность. Был введен режим многопоточного выполнения, позволивший снизить падение производительности при моделировании одновременной работы нескольких ядер. Кроме того, было разработано промежуточное представление широкой команды, позволившее реализовать программный кэш широких инструкций. Эта оптимизация позволила сократить время, затрачиваемое на повторное чтение и декодирование уже единожды декодированных широких команд, и тем самым повысила скорость работы одного потока.

Практическим результатом работы стало увеличение производительности примерно в полтора раза при использовании кэша инструкций (включен по умолчанию при работе симулятора) и примерно такое же – при использовании многопоточного режима (включение производится по опции, т.к. требует многоядерного сервера).

Дальнейшие направления исследования предполагают доработку оптимизации многопоточного режима, дающую возможность более редкой синхронизации между потоками путем введения механизма отслеживания конфликтов и отката на предыдущее состояние в случае возникновения конфликта. Для однопоточного режима предполагается оптимизировать работу операций load/store при помощи введения программного кэша трансляций виртуального адреса моделируемой инструкции в адрес памяти симулятора.

СПИСОК ЛИТЕРАТУРЫ

1. Herrod S. A. Using Complete Machine Simulation to Understand Computer System Behavior, PhD thesis, Stanford University, 1998.
2. Ким А. К., Волконский В. Ю., Груздов Ф. А. и др. Архитектура, программное обеспечение и применение компьютеров серии «Эльбрус» // IV Международная научно-практическая конференция «Современные информационные технологии и ИТ-образование»: Тез. докл. М., 2009.
3. Щербаков Е. С. Создание модели вычислительной системы по принципу соответствия структур модели аппаратуры // Высокопроизводительные вычислительные системы и микропроцессоры. Сб. науч. трудов ИВВС РАН, вып. 1. М., 1999. С. 82–90.
4. Щербаков Е. С., Тихорский В. В. Командная модель как базис для потактовой модели микропроцессора VLIW-архитектуры // Высокопроизводительные вычислительные системы и микропроцессоры. Сб. научн. трудов ИВВС РАН, № 2. М., 2001. С. 76–78.
5. Гурин К. Л., Мешков А. Н., Сергин А. В., Якушева М. А. Развитие модели подсистемы памяти вычислительных комплексов серии «Эльбрус» // Вопросы радиоэлектроники. 2010. Т. 3. № 3. С. 62–70.
6. Игнатенко А. А. Разработка программных средств для повышения быстродействия комплексов, моделирующих микропроцессоры с архитектурами SPARC–V9 и E3S: Дипломный проект, МИФИ (ГУ). М., 2009.

ИНФОРМАЦИЯ ОБ АВТОРАХ

Куцевол Виталий Николаевич, старший инженер-программист, АО «МЦСТ», 119334, Москва, ул. Вавилова, д. 24, тел.: 8 (499) 135-70-79, e-mail: kutsevol_v@mcst.ru.
Мешков Алексей Николаевич, к.т.н., начальник отдела, АО «МЦСТ», ПАО «ИНЭУМ им. И. С. Брука», 119334, Москва, ул. Вавилова, д. 24, тел.: 8 (499) 135-70-79, e-mail: alex@mcst.ru.
Черных Сергей Витальевич, начальник сектора, АО «МЦСТ», 119334, Москва, ул. Вавилова, д. 24, тел.: 8 (499) 135-70-79, e-mail: chernyh_s@mcst.ru.

For citation: Kutsevol V. N., Meshkov A. N., Chernykh S. V. The approaches to the performance optimization of multi-core «Elbrus» processors program models. Voprosy radioelektroniki, 2017, no. 3, pp. 57–61.

V. N. Kutsevol, A. N. Meshkov, S. V. Chernykh

THE APPROACHES TO THE PERFORMANCE OPTIMIZATION OF MULTI–CORE «ELBRUS» PROCESSORS PROGRAM MODELS

The approaches to the improving of the performance of «Elbrus» architecture simulators are described in the paper. The multithread execution mode and difficulties of its implementation are described. The optimization through the addition of the internal representation and the software cache of the wide instructions is proposed. The numeric values of the effectiveness of these optimizations are given.

Keywords: software emulation, simulator, multithread execution, Elbrus architecture.

REFERENCES

1. Herrod S. A. Using Complete Machine Simulation to Understand Computer System Behavior, PhD thesis, Stanford University, 1998.
2. Kim A. K., Volkonskiy V. Yu., Gruzdov F. A. et al. Architecture, software and application of «Elbrus» series computers. *IV Mezhdunarodnaya nauchno-prakticheskaya konferentsiya «Sovremennye informatsionnye tekhnologii i IT-obrazovanie»: Tez. dokl.* Moscow, 2009 (In Russian).
3. Scherbakov E. S. Computer system development based on principle of correspondence between model and hardware. *Vysokoproizvoditelnye vychislitelnye sistemy i mikroprocessory: Sb. nauchn. trudov IVVS RAN,* Issue 1, Moscow, 1999, pp. 82–90 (In Russian).
4. Scherbakov E. S., Tikhorskiy V. V. Functional model as a basis for cycle precise model of VLIW architecture microprocessor. *Vysokoproizvoditelnye vychislitelnye sistemy i mikroprocessory: Sb. nauchn. trudov IVVS RAN,* no. 1, Moscow, 2001, pp. 76–78 (In Russian).
5. Gurin K. L, Meshkov A. N., Sergin A. V., Yakusheva M. A. The evolution of memory subsystem of Elbrus series simulators. *Voprosy radioelektroniki,* 2010, vol. 3, no. 3, pp. 62–70 (In Russian).
6. Ignatenko A. A. Specialist degree thesis: the development of software components for performance improvement of simulators with SPARC V9 and E3S architectures: Diplomnyi proekt. Moscow, MIFI (GU), 2009 (In Russian).

AUTHORS

Kutsevol Vitaliy, senior software engineer, JSC «MCST», 24, Vavilova st., Moscow, 119334, Russian Federation, tel.: +7 (499) 135-70-79, e-mail: kutsevol_v@mcst.ru.
Meshkov Alexey, PhD, head of department, JSC «MCST», PJSC «Brook INEUM», 24, Vavilova st., Moscow, 119334, Russian Federation, tel.: +7 (499) 135-70-79, e-mail: alex@mcst.ru.
Chernykh Sergey, head of sector, JSC «MCST», 24, Vavilova st., Moscow, 119334, Russian Federation, tel.: +7 (499) 135-70-79, e-mail: chernyh_s@mcst.ru.