Многоядерные вычислительные среды и безопасное ПО
Часть 2

Пол Паркинсон, Wind River
Перевод Николая Горбунова

Во второй части статьи рассматриваются причины перехода к использованию многоядерных процессоров, вызванные этим изменения в программном обеспечении и различные подходы к его разработке и тестированию.

Многоядерные процессоры

Долгое время в разработке процессоров доминировала тенденция, определявшаяся так называемым законом Мура, суть которого состоит в том, что число транзисторов, которое можно разместить в интегральной схеме, со временем растёт экспоненциально, удваиваясь каждые два года. Можно предположить, что этот закон будет справедлив ещё несколько лет, хотя единого мнения по вопросу, когда будет достигнут предел миниатюризации транзисторов, не существует. С 1970 по 2005 год тактовая частота процессоров увеличилась от 1 МГц до единиц ГГц, что дало предсказуемый прирост производительности. Однако жизнеспособность этого подхода постепенно подходит к концу, т.к. с повышением тактовой частоты пропорционально растёт энергопотребление. Кроме того, для повышения тактовой частоты необходимо увеличивать напряжение питания, что также существенно увеличивает потребляемую мощность, т.к. она пропорциональна квадрату напряжения. Суммарный эффект можно выразить следующим упрощённым уравнением:

$$Мощность = C \times V^2 \times f,$$

где C – динамическая ёмкость переключающего элемента, V – напряжение питания, f – тактовая частота.

В результате в эволюции вычислительной техники наметилась в корне иная тенденция, выражающаяся в появлении многоядерных процессоров. Многоядерные процессоры совмещают в одном корпусе интегральной схемы два или более независимых вычислительных ядер, каждое из которых может работать на более низких тактовых частотах, что ведёт к снижению потребляемой мощности при сохранении производительности. На рисунке 4 слева показаны данные для одноядерного процессора на максимальной тактовой частоте. Средняя пара колонок демонстрирует, что снижение тактовой частоты до 80% от максимума сокращает энергопотребление вдвое. Наконец, правая пара колонок показывает, что использование двухъядерного процессора вместо одноядерного позволяет при том же энергопотреблении получить выигрыш по производительности в 1,6 раза (т.е. как у двух одноядерных процессоров, работающих на 80% максимальной тактовой частоты).

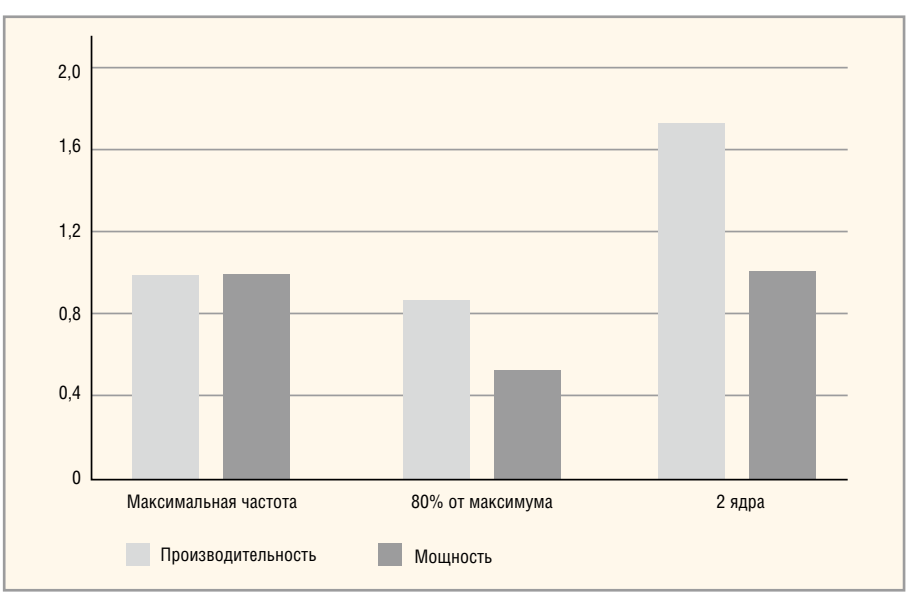

Рис. 4. Производительность и энергопотребление многоядерных процессоров

Однако это не означает, что все приложения будут выполняться в 1,6 раза быстрее. Чтобы в полной мере использовать преимущества двухъядерного процессора, приложение должно демонстрировать достаточный уровень параллелизма вычислений, то есть чтобы различные его части могли выполняться на двух ядрах параллельно. В общем случае прирост производительности приложения за счёт распараллеливания его выполнения по нескольким процессорам (или ядрам) определяется долей кода приложения, который должен выполняться строго последовательно. Эта закономерность известна как закон Амдала и может быть формально выражена следующим уравнением:

$$P_{пр} = \frac{1}{(Доля_{пар.кода} / N_я) + (1 - Доля_{пар.кода})},$$

где $P_{пр}$ – прирост производительности; $Доля_{пар.кода}$ — доля параллельного кода; $N_я$ – число ядер.

Допустимая степень распараллеливания для различных типов приложений может существенно отличаться. Например, ряд вычислительных задач имеет по определению последовательный характер, в то время как алгоритмы, используемые в радио- и гидролокации или обработке изображений, параллельны по своей природе. Этот разброс привёл к появлению различ-

ных конфигураций ПО для многоядерных процессоров (см. рис. 5).

Симметричная многопроцессорность (SMP)

Симметричная многопроцессорность подразумевает выполнение единственного экземпляра ОС со всеми прикладными программами несколькими одинаковыми процессорами. Изначально это реализовывалось за счёт использования группы одноядерных процессоров, сейчас же тенденцией является консолидация этих процессоров в одном корпусе многоядерного процессора, что позволяет сократить массогабаритные характеристики и снизить потребляемую мощность.

Степень масштабируемости многоядерной SMP-архитектуры будет определяться балансом между производительностью ядер, памяти и ввода/вывода. Например, если процессор обладает производительными ядрами с высокой скоростью исполнения команд, но ограничен в объёме кэша команд и данных, а также имеет низкую пропускную способность каналов ввода/вывода, то он квалифицируется как *ограниченный вводом/выводом* (I/O bound), и реально доступный объём вычислительных ресурсов у него будет занижен.

SMP предоставляет приложениям однородную вычислительную среду, и за распределение процессов/потоков приложений по доступным вычислительным ядрам в этом случае отвечает ядро ОС. Для приложений эти действия обычно прозрачны, хотя ряд реализаций дополнительно предоставляют возможность закреплять конкретные процессы/потоки за конкретными ядрами (т.н. *привязка* – affinity) с целью увеличения скорости обработки данных.

Перенос приложений, изначально разработанных для одноядерных архитектур, на многоядерные SMP-дизайны может потребовать значительных усилий, т.к. в таких приложениях изначально не закладывался достаточный уровень параллелизма. Даже при сбалансированной архитектуре процессора у SMP-дизайнов существует практический лимит количества ядер, до которого их можно масштабировать, т.к. SMP-приложениям в любом случае необходимо защищать разделяемые данные от ситуаций критических состязаний.

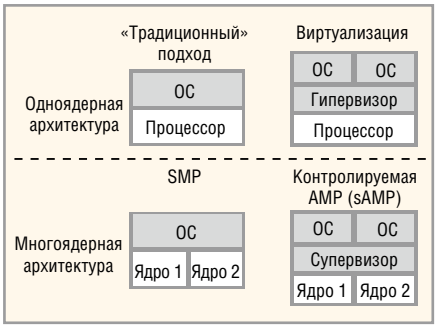

Рис. 5. Варианты конфигурации ПО для многоядерных процессоров

В результате уровень накладных расходов, вносимых SMP-дизайнами, вполне приемлемый для систем с небольшим количеством ядер, с увеличением количества ядер в системе неизбежно выходит за допустимые пределы. В этом случае, чтобы использовать все доступные вычислительные ресурсы оптимальным образом, SMP целесообразно сочетать с другими техниками многопроцессорности.

Асимметричная многопроцессорность (AMP)

Асимметричная многопроцессорность (AMP, иногда ASMP) использу-

ет противоположный подход, когда каждый процессор является независимым вычислительным устройством. Это позволяет выполнять на различных ядрах множество копий одного и того же приложения, работающих над различными наборами данных, а также (при необходимости) выполнять на различных ядрах различные ОС. Типовым примером последнего может быть выполнение ОСРВ, отвечающей за сбор данных или управление объектом, на одном ядре, и ОС общего назначения (например, Linux), отвечающей за интерфейс с пользователем, на другом.

В многоядерных AMP-системах необходимо обеспечивать вычислительным ядрам корректный совместный доступ к разделяемым ресурсам – например, кэшу 2 уровня (L2), шинам данных и ряду периферийных устройств. Это увеличивает сложность системы и требует более тщательного подхода к проектированию и разработке ПО, особенно в случае, когда на различных ядрах выполняются различные ОС, и результат некорректной операции на одном ядре может распространиться на все остальные ядра. По этой причине всё более набирает популярность реализация AMP на многоядерных архитектурах с поддержкой контроля (supervision) и виртуализации – более подробнее об этом см. далее.

На уровне приложения ядра работают независимо, что позволяет минимизировать усилия по портированию ПО на многоядерную архитектуру. Если приложению требуется «общаться» с другим приложением, выполняющимся на другом ядре, это реализуется непосредственно на уровне приложения с использованием соответствующего механизма межпроцессорного взаимодействия, использующего, в свою очередь, предоставляемый аппаратурой физический транспорт. Это отличается от реализации, используемой в SMP, где межзадачное взаимодействие между приложениями, выполняющимися на разных ядрах, является программно прозрачным.

Контролируемая и виртуализированная асимметричная многопроцессорность

Многоядерные AMP-системы могут быть очень сложными, особенно это касается консолидированных систем, использующих несколько ОС одновременно. На практике это означает, что на обеспечение корректности их реализации и конфигурирования приходится затрачивать больше усилий (в частности, это касается распределения аппаратных ресурсов – его приходится делать вручную).

Поскольку рост сложности (а значит, и трудозатрат) нежелателен, в последнее время набирает популярность подход с использованием программного *супервизора*, который выполняется ниже уровня ОС и управляет доступом к разделяемым ресурсам, таким образом, реализация этой функции со стороны применяемых ОС значительно упрощается. Супервизор также может предоставлять возможность контролируемой перезагрузки отдельных ядер, никак не затрагивая при этом работу остальных.

Ещё одна популярная в последнее время тенденция – использование *полной виртуализации* – является дальнейшим развитием подхода с супервизором. В этом случае ниже уровня ОС выполняется *гипервизор*, ответственный за создание для каждого ядра отдельной виртуализированной среды, в т.ч. настройку защиты памяти для каждой используемой ОС и их последующую загрузку. Это обеспечивает надёжную изоляцию операционных сред каждого ядра и может служить хорошей технологической основой для консолидации нескольких приложений, ранее выполнявшихся на отдельных процессорах, в рамках интегрированной многоядерной архитектуры. Обычно подобная задача включает в себя приложения, выполняющиеся под управлением ОС общего назначения (например, Windows или Linux), и приложения, выполняющиеся в среде ОСРВ (например, VxWorks).

В системах с повышенными требованиями к безопасности залогом максимальной эффективности использования многоядерных технологий является применение надёжного гипервизора. В процессе регулярного обновления технологической базы самой сложной задачей обычно является переход на новый процессор. Реализация безопасной системы на базе валидированного или сертифицированного гипервизора позволяет вынести аппаратно-зависимую функциональность за пределы ОС и приложений, а значит, обеспечивает переход на новые процессоры с минимумом повторного тестирования и верификации.

Конвергенция

В предыдущих главах современные тенденции в области безопасности ПО и многоядерных технологий рассматривались отдельно друг от друга. Очевидно, что, несмотря на то, что некоторые из этих тенденций не связаны между собой, у них также есть ряд важных точек соприкосновения, открывающих богатый потенциал их совместного использования.

Функциональная безопасность и многоядерные процессоры

В системах с повышенными требованиями к безопасности всё чаще используется консолидация приложений, что в рамках программной архитектуры IMA означает совместное выполнение на одном процессоре нескольких независимых приложений, а также переход от использования для этих целей однопроцессорных систем к многоядерным. Обязательным этапом при этом становится выяснение, пригодны ли многопроцессорные системы для построения надёжных и безопасных систем и какой тип многоядерных архитектур обеспечивает наибольшую степень безопасности.

SMP, на первый взгляд, кажется привлекательным решением для функционально-безопасных систем, т.к. сулит существующим «федеративным» приложениям увеличение производительности за счёт перехода на многоядерные процессоры. Выигрыш в производительности, правда, будет частично скомпенсирован проигрышем в детерминизме: во-первых, станет непредсказуемым время доступа к разделяемым ресурсам, т.к. теперь оно будет зависеть от активности всех остальных ядер, которые тоже могут запрашивать к ним доступ. Во-вторых, скажется побочный эффект кеширования, и перепланирование задачи будет занимать разное время в зависимости от того, назначается ли задача на то же самое ядро или на другое. К тому же для SMP не существует очевидного решения для реализации нескольких уровней безопасности для параллельно исполняемых задач. Из этого можно сделать вывод, что ожидать беспроблемного использования SMP-технологий в системах с высокими требованиями к функциональной безопасности не следует.

Наибольший потенциал для использования в системах повышенной функциональной безопасности имеют AMP-конфигурации. Кроме того, для

изоляции приложений разной степени «критичности» друг от друга и для предотвращения критических состязаний за разделяемые ресурсы имеет смысл сочетать AMP с технологиями виртуализации и гипервизорами. Такой подход может быть привлекателен, скажем, в задачах автоматизации технологических процессов или медицинского приборостроения, поскольку позволяет объединить, например, на одном двухъядерном процессоре «критическое» управляющее приложение, реализованное в среде ОСРВ, и «некритическое» интерфейсное приложение, использующее ОС общего назначения. Аналогично, в задачах авиаприборостроения AMP можно использовать для запуска нескольких приложений с «федеративной» архитектурой на отдельных ядрах или, наоборот, для запуска ИМА-приложений на выделенных ядрах под управлением ARINC 653-совместимой ОСРВ.

Несмотря на заметный интерес к применению многоядерных архитектур в AMP-конфигурации в «критических» системах авионики, на пути к широкомасштабному внедрению имеется ряд нерешённых вопросов. Один из них – возможность сцепления (coupling) между приложениями, выполняющимися на отдельных ядрах, из-за необходимости доступа к разделяемым ресурсам (включая кэши, контроллеры памяти и прочие устройства), что может вызвать, например, блокировку одного приложения другим, выполняющимся на другом ядре. Это в потенциале создаёт проблему для группы приложений как одного, так и разных уровней «критичности». В дополнение к этому, по результатам ряда исследований, сложность многоядерных процессорных архитектур усложняет расчёт наихудшего времени выполнения (worst-case execution time, WCET), необходимого для доказательства предсказуемости поведения системы при любом сочетании условий. Однако при соответствующем конфигурировании AMP-систем влияние всех этих проблем может быть сведено к минимуму.

В вопросах сертификации многоядерных систем для гражданского авиаприборостроения сертификационные органы придерживаются консервативного подхода, описанного в стандарте DO-254. Вероятно, в ближайшем будущем в результате исследований в области многоядерных технологий в авионике будут сформированы рекомендации по применению и конфигурированию многоядерных архитектур. Это, в свою очередь, даст новый толчок разработчикам многоядерных процессоров для улучшения создаваемых ими систем с точки зрения их применимости в системах повышенной функциональной безопасности, например, назначения ядрам раздельных кэшей для стабилизации временны́х характеристик и, соответственно, упрощения WCET-анализа.

Информационная безопасность и многоядерные процессоры

В системах с повышенными требованиями к информационной безопасности консолидация выражается в переходе от исполнения приложений с разными требованиями к информационной защищённости на физически разделённых процессорах к MILS-архитектуре, где приложения разного уровня «критичности»

исполняются на одном или нескольких процессорных ядрах одной и той же системы. В то же время применение многоядерных процессоров в подобных решениях также оставляет ряд нерешённых вопросов. Многие из них значимы и для обеспечения функциональной безопасности, но одна из проблем связана исключительно с информационной безопасностью – это проблема скрытых каналов.

В параграфе «Множественные независимые уровни безопасности» рассматривалась проблема скрытых каналов в однопроцессорных системах. В многоядерных архитектурах опасность образования скрытых каналов повышается по двум причинам. Во-первых, ряд устройств обычно используется вычислительными ядрами совместно, например, кэши, контроллеры памяти и прерываний, периферийные устройства. Во-вторых, за счёт одновременной работы нескольких ядер (а не поочерёдном выполнении различных приложений на одном ядре), пропускная способность скрытых каналов повышается. Например, двухъядерный процессор с разделяемым кэшем 2-го уровня предоставляет потенциал для образования высокоскоростного скрытого канала, исключить который можно, предоставив каждому ядру выделенный кэш.

Многоядерные процессоры могут сильно отличаться по своему внутреннему устройству – от однородных двух- и четырёхъядерных до массово-параллельных и даже гетерогенных архитектур, содержащих процессор общего назначения и целевые вычислительные модули, поэтому делать общие выводы о применимости многоядерных процессоров в системах повышенной информационной безопасности весьма затруднительно. Однако можно ожидать, что результатом идущих в настоящий момент исследований будет единый набор приемлемых практик для применения многоядерных процессоров в системах повышенной информационной безопасности, и это даст необходимую основу для широкомасштабного их внедрения.

Три в одном?

Теперь, когда потенциал для конвергенции между многоядерными технологиями и технологиями функциональной и информационной безопасности освещён, в качестве следующего шага следует рассмотреть ещё более сложный вопрос, включающий в себя конвергенцию всех трёх составляющих. Дело в том, что между требованиями к функциональной и информационной безопасности всегда существуют неявные противоречия, что в конечном итоге увеличивает общую сложность. Например, если к многоядерной системе предъявляются требования по функциональной и информационной безопасности одновременно, открытым остаётся вопрос, как обрабатывать сбой в работе одного из ядер. Должна ли система пытаться перезагрузить сбойное ядро? Или отключить его, выполнив деградацию? Или следует предположить, что сбой является следствием атаки, и перевести и другие ядра в безопасное состояние?

Несмотря на то что тенденция к объединению многоядерных технологий с технологиями безопасности ещё не оформилась, ожидается, что по мере разрешения текущих вопросов конвергенция между ними станет неизбежной реальностью.

Если говорить о программных архитектурах, имеющих максимальный потенциал для одновременной поддержки многоядерных архитектур и технологий функциональной и информационной безопасности, то очевидно, что, например, ARINC 653-совместимые функционально-безопасные ОСРВ вряд ли удастся привести в соответствие с требованиями информационной безопасности, в первую очередь за счёт объёма кода ядра, а также из-за возможности реализации драйверов в пространстве ядра. При этом MILS-архитектуры на основе гипервизора, применимые в информационно-безопасных системах, напротив, способны удовлетворить требованиям функциональной безопасности за счёт своей способности к изоляции приложений. Поэтому именно их следует рассматривать как наиболее перспективный вариант решения проблемы безопасного ПО в многоядерных вычислительных средах.

Заключение

Конвергенция технологий в области безопасности ПО и многоядерных вычислений уже началась и постепенно набирает силу. Уровень технологической готовности технологий ARINC 653 и ИМА за последнее десятилетие существенно возрос, переводя их из разряда лабораторных исследований в реальные успешные проекты гражданской и военной авионики.

Внедрение MILS-технологий в настоящее время пока находится на зачаточной стадии и начнёт более активно развиваться по мере появления успешных реализаций. По мере того как производители кремния развивают многоядерные технологии, они всё более переходят в разряд повседневной реальности. Индустрия требует комплексного решения проблемы безопасного ПО в многоядерных вычислительных средах; при этом потребность в нём растет быстрее, чем развиваются технологии, и несмотря на то, что ключевые технологии уже существуют, единого решения всё ещё не найдено. Ожидается, что наибольший потенциал для реализации такого решения содержат в себе MILS-архитектуры на основе гипервизора.

Процесс развития технологий нельзя ускорить, но подходить к нему необходимо с особой тщательностью, поскольку в функционально-безопасных системах на кону стоят человеческие жизни, а в информационно-безопасных – национальные интересы. Текущие изменения в нормативной базе и рост требований к связности вычислительных систем в областях функциональной и информационной безопасности говорят о том, что вскоре мы станем свидетелями уверенного движения индустрии в сторону унифицированных архитектур, позволяющих создавать сложные виртуализированные системы на многоядерных платформах и обеспечивать надёжную вычислительную среду для выполнения безопасных приложений.

Литература

1. First Flight of Carrier-Based Version of F-35 Joint Strike Fighter Scheduled for This Week. Military Aerospace Electronics. 2010.
2. Software Considerations in Airborne Systems and Equipment Certification. DO-178C. RTCA. 2011.
3. Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations. RTCA DO-297/EUROCAE ED-124. 2005.
4. National Security Telecommunications and Information Systems Security Policy (NSTISSP) No. 11, Information Assurance Directorate. 2000. www.cnss.gov/Assets/pdf/nstissp_11_fs.pdf.
5. *Kinnan, Larry*. Use of Multi-core Processors in Avionics Systems and Their Potential Impact on Implementation and Certification. 28th Digital Avionics Systems Conference. 2009.