Многоядерные вычислительные среды и безопасное ПО
Часть 1

Пол Паркинсон, Wind River
Перевод Николая Горбунова

В статье представлены современные направления развития технологий безопасности и процессорных архитектур. Затрагиваются вопросы применимости многоядерных процессоров в системах с повышенными требованиями к функциональной и информационной безопасности.

Введение

Изначально большинство систем, к которым предъявлялись повышенные требования с точки зрения функциональной и информационной безопасности (в англоязычной терминологии «safety» и «security», соответственно), а также защищённости, реализовывались на одноядерных процессорах. Типовым способом удовлетворения растущих требований к производительности для таких систем было увеличение тактовой частоты. Однако к настоящему времени резервы этого сравнительно простого метода практически исчерпаны, и сегодня для повышения производительности всё чаще используются многоядерные процессорные архитектуры.

Кроме перехода к многоядерным процессорам существует несколько других приёмов повышения безопасности, защищённости и производительности встраиваемых систем. Некоторые из них развиваются независимо, в то время как другие определённо взаимосвязаны — рост производительности процессоров обусловливает способность выполнять на одном и том же процессоре несколько приложений одновременно, что, в свою очередь, отражается на процедуре сертификации.

В данной статье описываются современные направления развития технологий безопасности и процессорных архитектур, затрагиваются вопросы применимости многоядерных процессоров в системах с повышенными требованиями к функциональной и информационной безопасности. Обсуждаются связанные с этим трудности, роль технологий виртуализации в вопросах безопасности и возможность сближения методов обеспечения безопасности, защищённости и повышения производительности за счёт использования многоядерных архитектур, способного привести к выработке унифицированного подхода.

Функциональная безопасность

Роль программного обеспечения (ПО) в системах с повышенными требованиями к безопасности в последнее время резко возросла. Скажем, в военной авиации постоянно растущий объём требований к возможностям летательных аппаратов (ЛА) привёл к взрывному росту номенклатуры и сложности систем авионики, призванных обеспечить превосходство в воздухе. В качестве примера таких систем можно привести коллиматорные дисплеи, электронные картографические системы, нашлемные системы целеуказания, системы радиоэлектронного противодействия...

Некоторые бортовые электронные системы в настоящий момент достигли такого уровня, что способны существенно снижать рабочую нагрузку на пилота, например, модуль распознавания речи истребителя Eurofighter Typhoon позволяет управлять «некритичными» функциями самолёта при помощи голосовых команд. По мере того как возможности таких систем продолжают развиваться, повышается вероятность использования их лётчиком для выполнения изначально непредусмотренных задач: скажем, использования электронной картографической системы для полётов на малых высотах при сложном рельефе или плохой видимости. Это автоматически переводит такую систему в разряд критических по функциональной безопасности (safety critical).

С ростом функциональных возможностей систем авионики соответственно вырос и объём применяемого в них программного кода. В 80-х годах XX века объём программного кода, применяемого в типовом серийном истребителе, составлял около 100 тысяч строк. К концу 90-х в истребителях следующего поколения он достиг 1 млн строк. Один же из истребителей текущего поколения, F-35 Lightning, содержит более 5 млн строк кода.

Авионика с «федеративной» архитектурой

В 1980-х годах в архитектуре систем военной авионики использовался т.н. «федеративный» подход, при котором каждая функциональная единица реализовывалась на отдельном процессоре (см. рис. 1). Неотъемлемым свойством такого подхода является надёжная изоляция приложений друг от друга и, как следствие, изоляция сбоев (т.е. сбой одного приложения не может вызвать сбой в другом), но ему свойственны и ряд недостатков. Например, для него

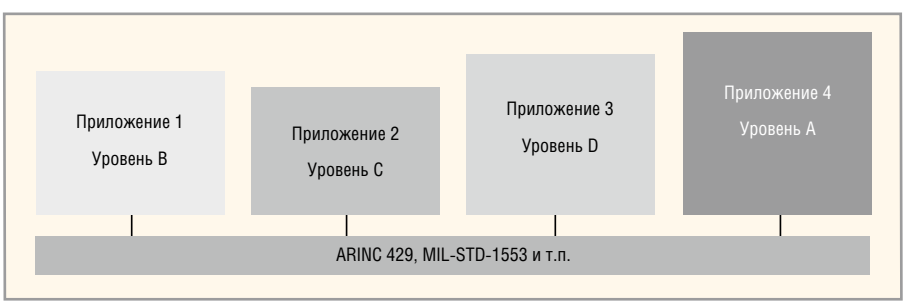

Рис. 1. Авионика с «федеративной» архитектурой

характерна высокая стоимость техобслуживания и ремонта (ТОиР), т.к. во всех точках обслуживания приходится одновременно хранить полный набор ЗИП для всех конструктивно-сменных блоков (КСБ), плюс для их диагностики, замены и ремонта требуется обучение персонала.

«Федеративная» архитектура также избыточна с точки зрения массогабаритных характеристик и потребляемой мощности, т.к. в рамках данного подхода каждому приложению полагается выделенный процессор, расположенный в выделенном КСБ, а он занимает пространство и потребляет энергию. Это, в свою очередь, выливается в дополнительные требования к межблочным кабельным соединениям, ещё более увеличивая общую массу системы.

Ещё одним примечательным недостатком «федеративной» архитектуры, который легко упустить из виду, является её неустойчивость к проблеме устаревания оборудования. Современный военный ЛА содержит множество электронных систем, с большой вероятностью разработанных различными производителями независимо друг от друга. В случае «федеративной» архитектуры это означает множество процессоров с различными архитектурами и множество различных одноплатных компьютеров на их основе. Это усложняет и удорожает процесс контроля ЗИП (за счёт низкой степени унификации КСБ) и усугубляет проблему устаревания (за счёт увеличения вероятности снятия компонентов с производства).

Проблема устаревания оборудования особенно обострилась в последнее десятилетие, после выпущенного в 1994 году министром обороны США меморандума об использовании коммерческих (COTS) компонентов в оборонных программах. Это позволило оборонному сектору получить доступ к новинкам полупроводниковых технологий, возникающим под влиянием промышленного и телекоммуникационного (а впоследствии и потребительского) рынков, а также существенно сократить издержки производства. Однако у этого явления была и оборотная сторона, заключавшаяся в снижении доступности процессоров военного класса с длинным жизненным циклом (ЖЦ), необходимым для оборонных программ. ЖЦ процессоров коммерческого и промышленного класса обычно значительно короче, т.к. это позволяет производственным программам быстро адаптироваться к появлению новых, более производительных и менее энергопотребляющих процессоров по мере их разработки и производства.

В свете вышеперечисленного выбор процессоров для оборонных программ стал требовать особой тщательности. К счастью, ряд производителей полупроводниковых приборов предусматривают для некоторых типов процессоров удлинённые ЖЦ, что позволяет применять их в оборонных программах при сохранении планов регулярного обновления основных продуктовых линеек. Но даже этот выход не признаётся идеальным, т.к. в оборонных программах предпочтение отдаётся стандартизации конкретной процессорной архитектуры в рамках всего проекта – иначе пришлось бы многократно повторять процедуру сертификации оборудования по стандарту DO-254 (а значит, и предусмотренную в её рамках верификацию процессора), а также сертификации ПО по функциональной безопасности согласно стандарту DO-178C.

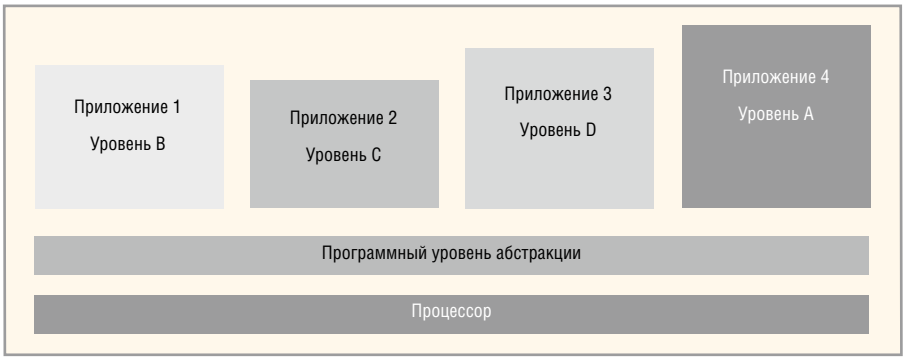

Рис. 2. Концепция интегрированной модульной авионики (ИМА)

Концепция интегрированной модульной авионики

Основной предпосылкой для развития концепции интегрированной модульной авионики (ИМА, см. рис. 2) за последние десять лет послужила проблема снижения массогабаритных характеристик и потребляемой мощности.

Концепция ИМА подразумевает использование унифицированных вычислительных платформ, выполняющих несколько приложений параллельно, тем самым снижая общее количество вычислительных устройств на борту ЛА по сравнению с «федеративным» подходом (хотя в ряде ЛА из соображений эффективности или обратной совместимости может использоваться комбинация «федеративных» и ИМА-систем).

ИМА базируется на двух базовых принципах – пространственной и временной изоляции приложений. Это гарантирует отсутствие влияния отдельных приложений друг на друга как случайно, так и преднамеренно. Изоляция приложений в ИМА реализуется за счёт особой программной архитектуры (ARINC 653 предписывает разделение времени между диспетчеризуемыми программными разделами) и использования аппаратного диспетчера памяти (для изоляции разделов друг от друга).

В дополнение к изоляции приложений в процессе проектирования системы необходимо убедиться, что приложения не будут влиять друг на друга посредством доступа к разделяемым ресурсам (т.н. «сцепление» (coupling) – проверки на предмет его отсутствия предписываются стандартом DO-178C). В случае наличия разделяемых устройств ввода/вывода в программной архитектуре системы должен быть предусмотрен либо отдельный раздел, отвечающий за ввод/вывод, либо администратор виртуальных устройств, предоставляющий приложениям дополнительный уровень абстракции и преобразующий параллельный доступ разделов к устройствам в последовательный. Для прочих разделяемых ресурсов, например кэша 2-го уровня, необходимо построить реализацию так, чтобы изменение содержимого кэша, вызванное работой отдельных приложений, не влияло на стабильность временны́х характеристик системы, например, времени переключения контекста между разделами. Типовым решением здесь является сброс кэша в конце каждого временно́го интервала раздела (partition timeslot, он же «младший кадр» (minor frame) в терминах ARINC 653), в результате время старта следующего раздела становится предсказуемым.

ИМА-подход на настоящий момент достиг необходимого уровня технологической готовности (technology readiness level, TRL) для широкого применения в программах гражданской и военной авионики. Это подтверждается более чем 200 глобальными программами, использующими ARINC 653. ИМА-архитектура используется даже для одновременной реализации нескольких приложений с различными требованиями к функциональной безопасности на одном и том же процессоре, например, в заправщике Boeing KC-767 и авиалайнере Boeing 787 Dreamliner, использующих ОС реального времени (ОСРВ) VxWorks 653.

Обычно если система содержит в себе несколько приложений (или функциональных единиц) с разными уровнями безопасности, вся система разрабатывается и сертифицируется с учётом максимального необходимого уровня. Этот подход может оказаться очень дорогостоящим, особенно если приложения, не предъявляющие высоких требований к безопасности (например, приложение 3 на рисунке 2, требующее уровня D по DO-178C), содержат большой объём кода. В рамках же концепции ИМА приложению достаточно пройти тестирование и сертификацию на уровень *не более необходимого*, что существенно снижает трудоёмкость и стоимость сертификационных работ.

Здесь, правда, есть пара нюансов. Во-первых, используемый программный модуль, обеспечивающий нужный уровень абстракции (обычно это ОС, реализованная согласно ARINC 653 или стандартам ASAAC), должен пройти сертификацию по самому высокому из требуемых приложениями уровней безопасности. Во-вторых, необходимо провести тщательный анализ изоляции разделов, чтобы удостовериться, что как пространственная, так и временна́я изоляция реализованы корректно. Это даст уверенность, что, например, сбой вышеупомянутого приложения уровня D не повлечёт за собой нарушение работы остальных приложений, выполняющихся на этом же процессоре.

Концепция ИМА также предоставляет мощный потенциал для обновления (и добавления) приложений на протяжении жизненного цикла системы с минимальными затратами. Это важный момент, т.к. внесение изменений в уже сертифицированную систему означает проведение повторной сертификации. В частности, исследования показали, что «стоимость внесения изменения в систему пропорциональна размерам системы, а не размерам самого изменения». Для больших систем это может иметь катастрофические последствия с точки зрения величины дополнительных затрат при внесении изменений.

Несмотря на то, что концепция ИМА подразумевает модульную структуру системы, само по себе это ещё не гарантирует модульную и инкрементную сертификацию. Например, если в реализации ИМА бинарные модули разделов и ОСРВ скомпонованы в единый монолитный образ, инкрементная сертификация такого решения будет невозможна. Однако было показано, что ИМА-платформа на базе ARINC 653, пригодная для инкрементной сертификации, может быть реализована при наличии надёжной изоляции разделов и механизма ролевого конфигурирования (role-based configuration) в рамках стандарта DO-297/ED-124. В этом случае кон-

фигурация системы и разделов задаётся в формате XML, который затем компилируется в бинарный файл конфигурации (т.н. макет системы – system blueprint) и затем компонуется с бинарными модулями ОСРВ и приложений (разделов).

Информационная безопасность

Традиционно повышенная информационная безопасность была привилегией специализированных систем, используемых в секретных правительственных и оборонных системах, разработанных в рамках «Оранжевой книги» (стандарт 5200.28-STD, одна из книг «Радужной серии» министерства обороны США) или «Общих критериев» (стандарт МЭК 15408). Однако в последнее время наблюдается растущая тенденция к повышению безопасности систем, используемых в критической инфраструктуре, включая разработку и сертификацию систем повышенной безопасности, соответствующих в рамках «Общих критериев» уровням EAL 4 и выше. Эта тенденция вызвана растущими оценками угроз кибертерроризма и кибервойн, предсказанных Объединённым комитетом по экономическим вопросам Конгресса США (U.S. Congressional Joint Economic Committee, JEC) ещё в 2002 году. Тогда же было отмечено, что степень угрозы кибертерроризма для национальной инфраструктуры США ещё не осознаётся в полной мере, а также что «уязвимость авиации и прочих элементов ключевой инфраструктуры существенно повышается за счёт их неотделимости от глобальных каналов коммуникаций». Степень реализуемости кибератак и их потенциального влияния на ключевую инфраструктуру резко возросла в последние годы за счёт расширения сферы персональных коммуникаций, электронной коммерции и прочих деловых и промышленных операций в интернет-пространстве.

Основными областями применения приложений повышенной информационной безопасности до сих пор являются правительственные и оборонные программы. Это объясняется политикой правительства США и соответствующей инициативой, выраженной в Исполнительном приказе Президента США № 12333 от 4 декабря 1981 года, в котором утверждается, что «системы должны быть безопасными». В результате Директоратом информационной безопасности Агентства национальной безопасности (National Security Agency, NSA) был выпущен Меморандум о безопасности национальных телекоммуникационных и информационных систем (National Security Telecommunications and Information Systems Security Policy, NSTISSP) № 11. Этот меморандум предписывает необходимость обеспечения информационной безопасности для всех систем ввода, обработки, хранения, отображения и передачи информации, связанной с национальной безопасностью, а с 1 июля 2002 года он также требует от коммерческих компонентов, применяемых в таких системах, демонстрации необходимого уровня информационной безопасности путём валидации их согласно утверждённым методикам.

Множественные независимые уровни безопасности

От ряда правительственных и оборонных систем требуется способность

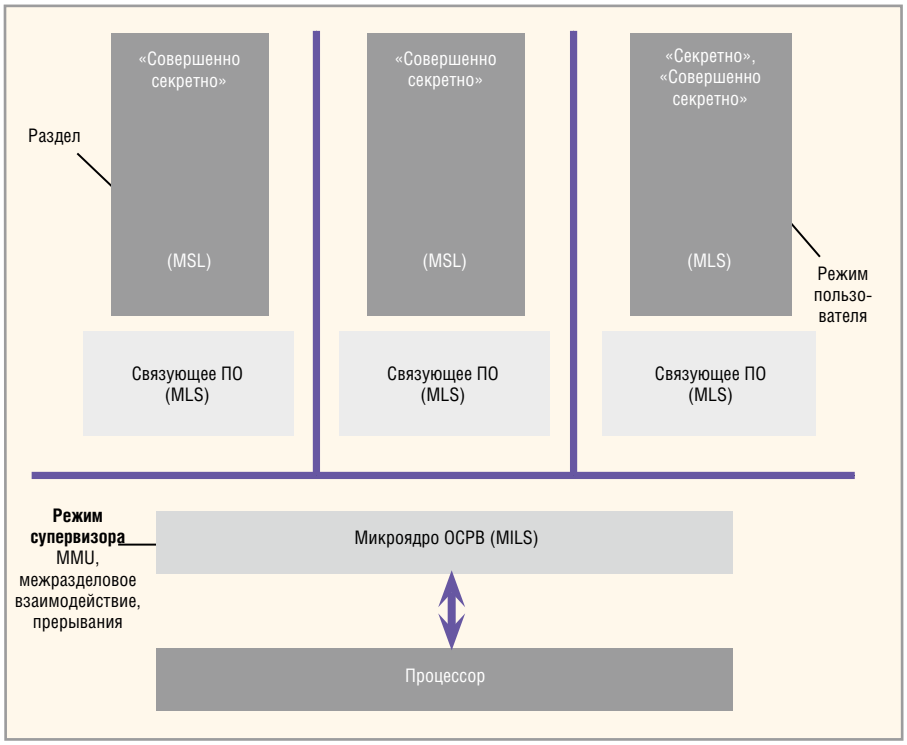

Рис. 3. Архитектура с множественными независимыми уровнями безопасности

параллельно обрабатывать информацию разной степени секретности – такой подход называется многоуровневой безопасностью (multilevel security, MLS). В последние годы необходимость в этой способности резко возросла в связи с необходимостью информационного обеспечения союзнических операций НАТО, когда от систем требуется, с одной стороны, обрабатывать секретные данные национального и союзнического уровня одновременно, а с другой – поддерживать необходимую степень изоляции.

В прошлом такие многоуровневые системы реализовывались при помощи множества физически разобщённых компьютеров, сетей и дисплеев. Всё это оборудование, кроме дороговизны, обладало ещё и существенной избыточностью с точки зрения массогабаритных характеристик и потребляемой мощности. Было предпринято множество попыток реализации подобной системы на базе унифицированной вычислительной платформы, работающей под управлением нескольких монолитных безопасных ОС, но сроки разработки и сертификации такой системы были оценены более чем в 10 лет.

Призванная решить эту проблему архитектурная концепция множественных независимых уровней безопасности (multiple independent levels of security, MILS, см. рис. 3) была изначально предложена Джоном Рашби (John Rushby) ещё в 1984 году. Однако на то время ни один из доступных на рынке процессоров не имел достаточной производительности — ни для одновременного выполнения множества приложений, ни для надёжной изоляции их друг от друга. Ограничения, связанные с производительностью процессоров, привели к появлению архитектур с множественными одиночными уровнями безопасности (multiple single-level, MSL), в которых различные уровни безопасности разделялись путём разнесения их по отдельным компьютерам (или виртуальным машинам). MSL-концепция позволяет реализовывать подмножества многоуровневой безопасности без необходимости в модификации приложений и ОС, но требует применения дополнительного оборудования.

В последние годы сближение между требованиями к производительности для обеспечения нужного уровня безопасности и возможностями полупроводниковых технологий позволило сделать MILS-системы реальностью. Это произошло благодаря неуклонному развитию процессорных архитектур. Современные процессоры значительно превосходят по производительности процессоры предыдущего десятилетия и обладают достаточной вычислительной мощностью, чтобы выполнять одновременно несколько приложений на одной и той же платформе в отдельных доменах, и при этом снабжены эффективными аппаратными средствами для обеспечения их изоляции. Это делает построение MILS-систем не только технически возможным, но и экономически оправданным.

Программное ядро безопасности (separation kernel) строится исходя из четырёх фундаментальных политик.

1. *Информационные потоки*. Задаёт допустимые информационные потоки между разделами.

2. *Изоляция данных*. Гарантирует, что раздел не может получить доступ к ресурсам других разделов.

3. *Временна́я диаграмма*. Обеспечивает выполнение приложений в разделах в запланированные временные интервалы.

4. *Изоляция сбоев*. Обеспечивает отсутствие влияния сбоев одного раздела на работу других.

Из этих четырёх постулатов следует архитектура, позволяющая создавать дополнительные компоненты, каждый из которых является неотключаемым (non-bypassable), верифицируемым (evaluatable), постоянно действующим (always invoked) и неповреждаемым (tamperproof). Эти свойства кратко обозначаются аббревиатурой NEAT[1]. В дополнение к этому для корректной реализации вышеперечисленного разработчики обязаны предпринять особые меры по минимизации неявных каналов коммуникации между приложениями (т.н. скрытые каналы, covert channels).

Например, скрытый канал по времени (covert timing channel) может образоваться, если время выполнения раздела подвержено флуктуациям (jitter). Если разброс времени выполнения раздела достаточно велик, чтобы приложение следующего раздела смогло его распознать, у приложений появляется возможность неявно передавать друг другу логический 0 или 1. Аналогично, скрытый канал по памяти (covert storage channel) может образоваться, если приложение пытается получить доступ к устройству ввода/

[1] *Детали реализации ядра безопасности VxWorks MILS описаны в статье Пола Паркинсона (Paul Parkinson) и Арлена Бейкера (Arlen Baker) «Разработка систем повышенной безопасности на базе архитектуры MILS» (High Assurance Systems Development Using the MILS Architecture), см. http://www.windriver.com/whitepapers.*

вывода, кэшу или какому-либо другому разделяемому устройству, а оно оказывается недоступно, потому что в настоящий момент используется другим приложением. Это также предоставляет приложениям потенциальный канал обмена двоичными значениями.

Гипервизоры и виртуализация

Современные реализации ядер безопасности на основе архитектуры MILS могут использовать аппаратные функции виртуализации, предоставляемые последними поколениями процессоров. Это позволяет, к примеру, реализовать гипервизор, способный выполнять гостевые ОС поверх ядра безопасности MILS в виртуализированной среде. Такой подход автоматически обеспечивает MILS-ядру изоляцию данных и контроль над информационными потоками, позволяя исключить появление скрытых каналов.

Здесь часто используются два подхода: полная виртуализация (full virtualization), когда гостевая ОС выполняется без изменений за счёт использования аппаратной поддержки виртуализации, и паравиртуализация (paravirtualization), когда аппаратной поддержки виртуализации нет, и гостевая ОС модифицируется с целью повышения общей производительности.

В случае полной виртуализации процессор аппаратно позволяет MILS-ядру выполнять гостевые ОС в виртуальных машинах (разделах) с пониженным уровнем привилегий, тем самым предотвращая доступ гостевых ОС к физической памяти за пределами соответствующих виртуальных машин. Изоляция данных реализуется и контролируется полностью MILS-ядром и не требует от разделов никаких действий.

В случае паравиртуализации (на процессорах, у которых аппаратная поддержка виртуализации отсутствует) MILS-ядро обеспечивает изоляцию данных посредством аппаратного диспетчера памяти процессора (memory management unit, MMU).

Помимо всего прочего, чтобы виртуализация была безопасной, гипервизор должен быть реализован корректно. В противном случае появляется возможность нарушения требований безопасности или внешней атаки, наподобие продемонстрированной в 2008 году Йоанной Рутковской (Joanna Rutkowska) и Рафалом Войчуком (Rafal Wojtczuk) атаки по схеме Blue Pill на гипервизор Xen, используемый в корпоративных сетях. По этой причине ряд MILS-ядер разрабатывается с использованием формальных методов. Это необходимо для прохождения сертификации согласно «Общим критериям» по высшим (EAL 6+) уровням информационной безопасности.

MILS-архитектура на основе гипервизора также может быть использована в сочетании с ролевой моделью разработки, предусмотренной стандартом DO-297/ED-124. Эта модель подразумевает разделение ролей между поставщиками компонентов решения («поставщик платформы», «разработчик приложения», «интегратор») и предоставляет необходимый потенциал для инкрементной модификации.

В следующем номере журнала «Современная электроника» речь пойдёт о многоядерных процессорах.