Разработка RISC-V

Оглавление

Введение

Эта глава посвящена техническим деталям архитектуры набора команд (Instruction Set Architecture, ISA) RISC-V и ее отличиям от других ISA. В ней рассмотрены некоторые правила и нормы, регулирующие работу RISC-V International, а также то, как разрабатываются расширения и стандарты ISA.

В этой главе вы узнаете:

- о процессе, который используется для разработки RISC-V ISA и его расширений;
- о различиях базовой, расширенной и стандартной ISA RISC-V;
- об основах непривилегированной (Unprivileged) и привилегированной (Privileged) спецификаций.

Руководство по архитектуре набора команд RISC-V

Определение архитектуры набора команд (ISA)

Архитектура набора команд (ISA, Instruction Set Architecture) – это абстрактная модель процессора. Ее также называют архитектурой или архитектурой процессора. Реализация ISA, например, центрального процессора (CPU, Central Processing Unit), называется имплементацией. Существуют такие распространенные ISA как x86, ARM, MIPS, PowerPC или SPARC. Все эти ISA требуют лицензии для их имплементации. В свою очередь, RISC-V ISA предоставляется под лицензией с открытым исходным кодом, которая не требует платы за использование.

Чем отличается ISA RISC-V

Наиболее заметным отличием RISC-V от других ISA является то, что RISC-V разрабатывается организацией с открытым членством, в которую можно вступить совершенно бесплатно, и лицензировать свои ISA на основе разрешительных лицензий с открытым исходным кодом. Это означает, что любой разработчик может внести свой вклад в архитектурные спецификации, и ни одна компания или группа компаний не может определять направление развития стандартов ISA.

RISC-V International управляется Советом директоров. Совет состоит из членов, избранных для представления всех классов членства, чтобы обеспечить возможность вносить вклад на всех уровнях. Кроме того, Технический комитет (Technical Steering Committee, TSC) обеспечивает руководство техническими инициативами по разработке долгосрочной стратегии, формированию тактических комитетов и рабочих групп и утверждение технических результатов для ратификации или выпуска спецификаций.

Модель совместной разработки

Начало разработки спецификации RISC-V начинается с момента утверждения Целевой группы Техническим комитетом (TSC). После того, как Целевая группа получила утвержденный регламент, она начинает публичную работу на GitHub, документируя свою деятельность в формате AsciiDoc. GitHub-репозиторий Целевой группы на может получать запросы на исправление (pull-request) только от членов RISC-V International, но при этом работа над спецификацией ISA ведется публично и прозрачно. Для групп, решивших отслеживать разработку спецификации ISA, в открытом доступе публикуются протоколы заседаний Целевой группы. Любой пользователь GitHub может свободно

задавать вопросы и давать раннюю обратную связь по любым аспектам спецификации ISA. Спецификации и стандарты, не относящиеся к ISA (например, трассировка процессора, архитектурные тесты, оверлейное программное обеспечение), разрабатываются аналогичным образом.

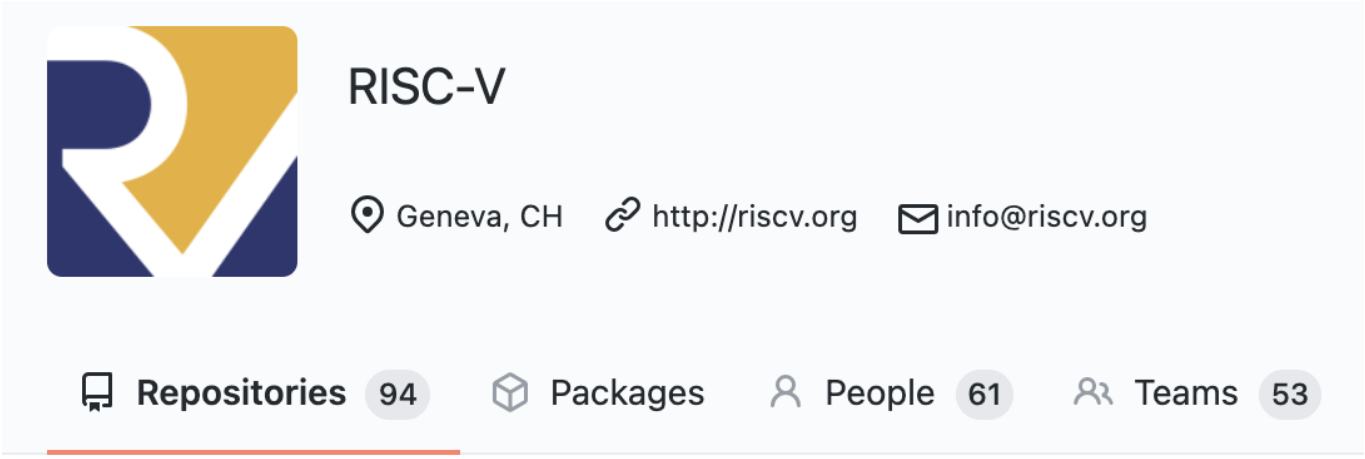

Рисунок 1. Репозиторий RISC-V на GitHub.

Спецификации RISC-V размещены на GitHub и находятся рядом с десятками программных проектов. Список ратифицированных спецификаций [1] и ссылки на их репозитории можно найти на GitHub.

Создание и сопровождение открытых спецификаций

Процесс разработки спецификаций обычно возглавляет архитектор аппаратного обеспечения в одной из организаций-членов RISC-V International. Архитектор может непосредственно не участвовать в составлении спецификации, но выступет в качестве председателя Целевой группы, контролирующей разработку спецификации. Для завершения разработки спецификации группе может потребоваться от нескольких месяцев до одного года и более. Жизненный цикл расширения будет описан далее.

Открытость процесса разработки основывается на трех ключевых факторах:

1. Список рассылки Целевой группы является общедоступным.
2. Документ спецификации находится в открытом доступе, и к нему можно оставлять комментарии.
3. Существует общедоступный адрес электронной почты, куда любой желающий может отправить письмо (isa-dev@groups.riscv.org).

Используя эту методологию, даже те, кто не являются членами организации, могут участвовать в разработке любой спецификации или стандарта, задавая вопросы, внося предложения или просто следя за ходом работы. Более того, в процессе ратификации существует 45-дневный период, когда все работы над спецификацией должны быть приостановлены, а сама спецификация опубликована для ознакомления сообществом разработчиков. В это время все желающие могут высказать свои замечания, и все вопросы будут решены до голосования за ратификацию.

Самый простой, хотя и не единственный, способ внести свой вклад в разработку открытых спецификаций это стать членом RISC-V International. Кроме этого, любой разработчик может внести свой вклад, взаимодействуя с целевыми группами, как на публичных форумах так и через GitHub.

Жизненный цикл расширений RISC-V

Каждое расширение RISC-V проходит несколько этапов на пути к ратификации (рис. 2). В этом разделе кратко рассмотрен каждый этап в отдельности :

1. **Планирование.**

   Целевая группа разрабатывает окончательную версию спецификации расширения и устанавливает сроки её реализации.

2. **Разработка.**

   Группа выпускает несколько версий спецификации расширения, которые считаются нестабильными.

3. **Приостановка (freeze).**

   Группа готовит полную окончательную версию спецификации, в которой не предусматриваются существенные изменения (версия, в которой возможны незначительные исправления, но не допускается добавления нового функционала).

4. **Подготовка к утверждению.**

   Проект спецификации рассылается на общественное рассмотрение, учитываются любые комментарии или вопросы общественности, и Технический Комитет (TSC) ставится в известность о необходимости голосования.

5. **Развитие экосистемы.**

   После ратификации расширения оно начинает поддерживается как часть RISC-V ISA. Однако, после ратификации, сообществу RISC-V предстоит решить еще множество задач. Например, несмотря на то, что спецификация Vector уже была ратифицирована, однако,

добавление функционала автовекторизации в компиляторы является частью списка задач по развитию экосистемы этой спецификации.

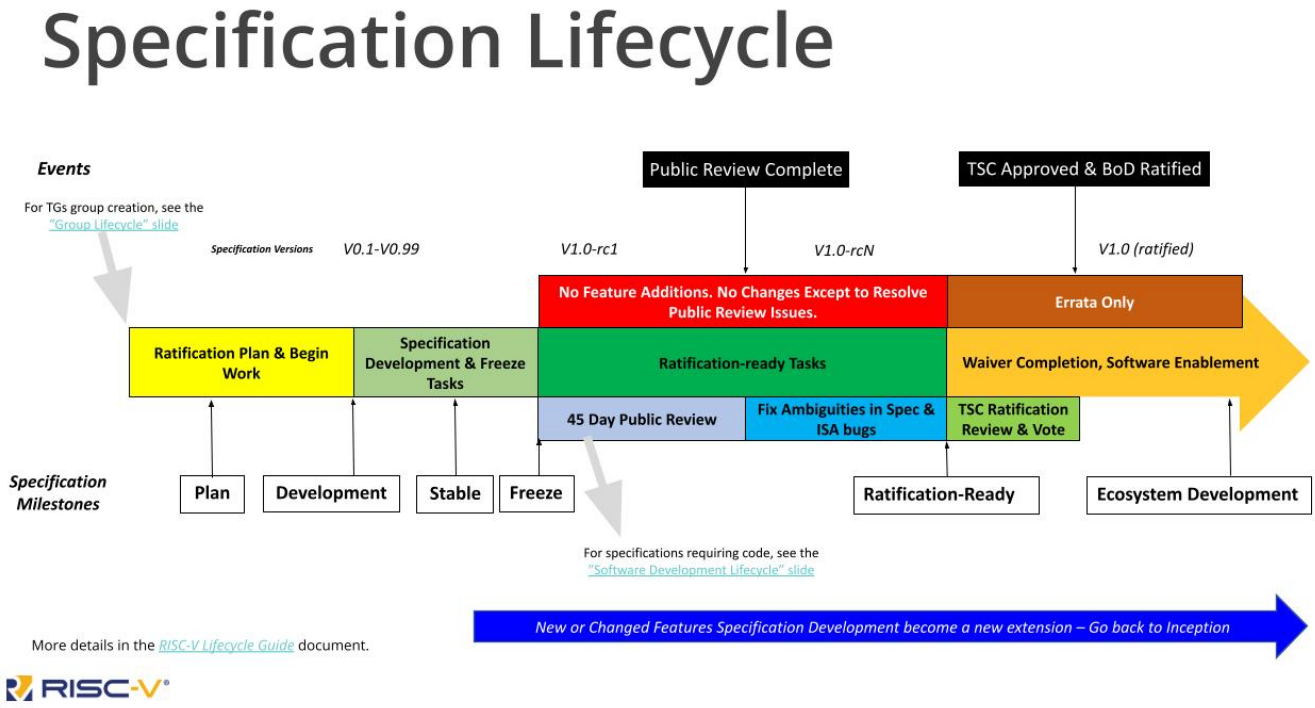

Рисунок 2. Жизненный цикл спецификации.

После ратификации расширения оно добавляется либо в непривилегированную, либо в привилегированную версию спецификации. Иногда спецификация создается как часть отдельного документа (наиболее распространенным примером является спецификация отладки). Но это редкий случай и обычно указывает на то, что расширение не является частью ISA, а представляет собой «стандарт» или «не-ISA спецификацию». Ниже представлены детали непривилегированной и привилегированной спецификаций.

Непривилегированная спецификация

Организация спецификаций

RISC-V ISA состоит из двух частей:

- часть 1, непривилегированная спецификация;
- часть 2, привилегированная спецификация.

Чтобы понять, почему спецификация делится на две части, необходимо сначала разобраться в компьютерной архитектуре и безопасности. Исторически процессоры использовали иерархические уровни защиты, часто называемые кольцами защиты (Рисунок 3), для защиты данных и кода от злоумышленников.

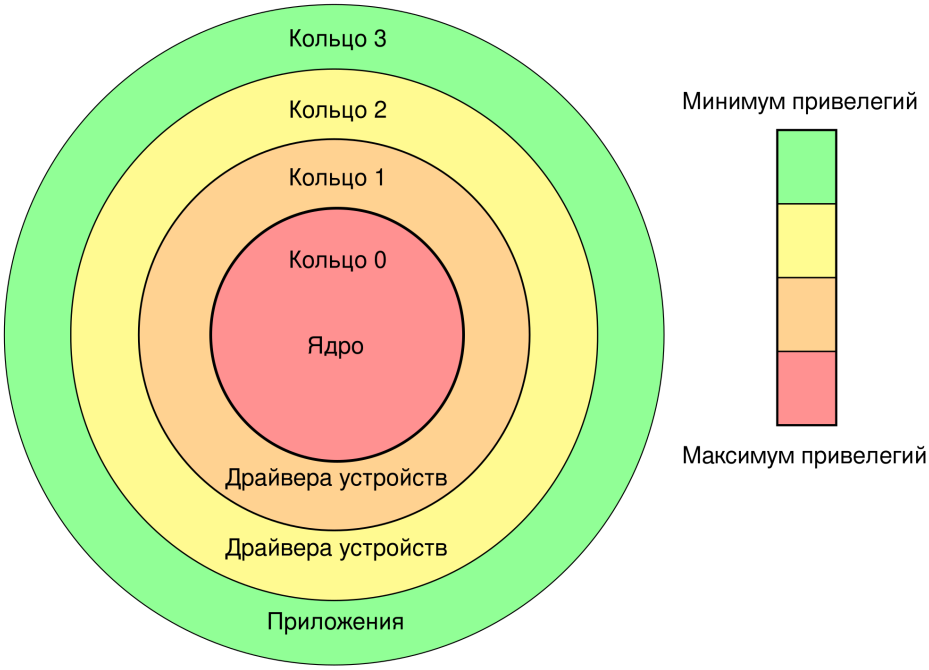

Рисунок 3. Кольца привилегий.

Наиболее привилегированный код работает в «кольце 0» и имеет доступ ко всей системе. Процессор решает, какие привилегии предоставить исполняемому коду, в зависимости от уровня привилегий. Например, доступ к памяти по физическому адресу может быть ограничен «кольцом 0», так что другие кольца должны ссылаться на виртуальное адресное пространство. В текущем моменте времени процессор может работать только в одном из режимов привилегий, а для перехода между режимами необходимо использовать специальные инструкции. Все эти детали

могут меняться от системы к системе, но они должны соответствовать правилам, описанным в документах спецификации конкретной архитектуры.

В настоящее время RISC-V имеет три уровня привилегий: режим пользователя (U-режим, User Mode), режим супервизора (S-режим, Supervisor Mode) и машинный режим (M-режим, Machine Mode). Их можно представить как «кольцо 2», «кольцо 1» и «кольцо 0» соответственно. Другие режимы, такие как режим гипервизора (H-режим, Hypervisor Mode), вероятно, будут добавлены в ближайшем будущем. Как и на рисунке выше, U-режим предназначен для пользовательских процессов, S-режим – для ядра и/или драйверов устройств, а M-режим – для загрузчика и/или микропрограммного обеспечения. Каждый уровень привилегий имеет доступ к определенным регистрам управления и состояния (Control and Status Registers, CSR), и более высокие уровни привилегий могут получить доступ к CSR менее привилегированных уровней.

Содержание непривилегированной спецификации

Непривилегированная спецификация описывает элементы, не относящиеся к машинному режиму (M-режим) или режиму супервизора (S- режим). В нее входят описание базового ISA, а также его расширения, такие наборы инструкций как: целочисленные (I, integer), с плавающей точкой (F, float), двойной точности (D, double), упакованные (C, compressed) и многие другие.

Базовые наборы инструкций описывают формат инструкций, основные целочисленные инструкции, инструкции загрузки и хранения, а также особенности ISA. Базовые наборы инструкций бывают следующих типов:

- RV32I – целочисленный, 32 бита;
- RV32E – сокращенный RV32I для встраиваемых систем, 32 бита;
- RV64I – целочисленный, 64 бита;
- RV128I – целочисленный, 128 бит.

Все перечисленные типы базовых наборов являются либо уменьшенными, либо расширенными версиями базового набора инструкций RV32I. Например, RV64I расширяет целочисленные регистры и поддерживаемое адресное пространство пользователя до 64 бит. Это означает, что инструкции LOAD и STORE работают немного иначе, чем в RV32I. Непривилегированная спецификация содержит раздел, объясняющий эти различия.

Расширения базового ISA

Непривилегированная спецификация также содержит описания расширений базовых ISA. Повторим, что любое расширение, которое не предусматривает M-режима, может быть описано в непривилегированной спецификации.

Каждое расширение базового ISA разрабатывается и поддерживается Целевой группой:

- Crypto Task Group работает над криптографическими расширениями, которые могут быть реализованы в аппаратуре, что повысит их надежность и производительность;
- Extension Task Group работает над расширениями для работы с битовыми операциями, которые могут ускорить выполнение многих распространенных математических задач;
- Vector (V) Extension Task Group работает над векторными инструкциями, которые лежат в основе многих вычислений по обработке графики.

После ратификации эти расширения добавляются в непривилегированную спецификацию. Ниже перечислены некоторые из ратифицированных расширений, которые можно встретить в имплементациях процессоров RISC-V.

Расширение M

Глава 7 непривилегированной спецификации [1] описывает, как должно выполняться умножение и деление целых чисел. В ней описывается поведение каждой из инструкций умножения (MUL, MULH, MULHU, MULHU, MULW), какие регистры используются для множителя и множимого, и где хранится результат. То же самое делается и для деления, поскольку функционально можно рассматривать деление как операцию, обратную умножению. Может показаться странным, что это расширение не вошло в стандартную спецификацию. Но для многих встроенных процессоров умножение можно выполнять программно, если оно не используется часто или используется не часто. Исключение этой логики из процессора позволяет сэкономить средства при разработке и снизить стоимость конечного продукта.

Расширение F

В главе 11 непривилегированной спецификации описано, как добавляются вычислительные инструкции с плавающей точкой одинарной точности, соответствующие стандарту арифметики с плавающей точкой IEEE 754-2008. Существует множество Idокументов, описывающих детали арифметики с плавающей точкой в вычислениях. В этой главе описывается, как этот процесс

реализован в RISC-V, и дополняется главой 12 (расширение D), которая описывает арифметические инструкции двойной точности с плавающей точкой. А в главе 13 рассматривается расширение стандарта Q для 128-битных двоичных инструкций с плавающей точкой четверной точности. Все эти три расширения соответствуют стандартам IEEE. Многие встроенные приложения не требуют логики с плавающей точкой, и поэтому это расширение не является частью базовых ISA.

Расширение C

В главе 16 описано расширение компактного набора инструкций, которое уменьшает статический и динамический размер кода путем добавления сокращенных 16-битных кодов инструкций для общих операций. Как правило, 50%-60% инструкций RISC-V в программе могут быть заменены инструкциями RVC, что приводит к уменьшению размера кода на 25%-30%. Расширение C совместимо со всеми другими стандартными расширениями инструкций. Оно позволяет свободно смешивать 16-битные инструкции с 32-битными, причем последние могут начинаться с любой 16-битной границы. Таким образом, при добавлении расширения C в любую систему ни одна инструкция не вызывает исключения, связанные с выравниванием адресов инструкций.

Приведенные выше расширения охватывают большинство ратифицированных в настоящее время расширений по непривилегированной спецификации. Важно отметить, что многие расширения включены в спецификацию на стадии "разработка" или "приостановка (freeze)". Как уже было сказано в разделе «Жизненный цикл расширений RISC-V», эти спецификации еще не ратифицированы, и следует избегать их использования в производстве.

Привилегированная спецификация

Обзор

Как следует из названия, привилегированная спецификация содержит описания RISC-V ISA, которые работают в машинном режиме (M-режим) или режиме супервизора (S-режим). Эти режимы имеют повышенные привилегии и поэтому описаны в отдельном документе от базового ISA и стандартных расширений. Эта спецификация также содержит описание дополнительных функциональных возможностей, необходимых для запуска полнофункциональных операционных систем, таких как Linux.

В первой части каждой главы привилегированной спецификации подробно описаны регистры управления и состояния (CSR), доступ к которым возможен только из M- и S-режима. Мы не будем рассматривать их особенности здесь, а сосредоточимся на других деталях, специфичных для указанных режимов.

ISA машинного уровня (M-режим), версия 1.11

В этой главе описаны функции, доступные в машинном режиме. M-режим используется для низкоуровневого доступа к аппаратной платформе и является первым режимом, который включается после сброса, когда процессор завершает инициализацию и готов к выполнению кода. M-режим также может использоваться для реализации функций, которые слишком сложно или трудоемко реализовывать в аппаратном обеспечении непосредственно. Хорошим примером может служить сторожевой таймер, реализованный в низкоуровневом программном обеспечении (микропрограмме), который автоматически выполняет сброс системы после сбоя. Далее рассмотрены три важные особенности M-режима, описанные в спецификации: немаскируемые прерывания, атрибуты физической памяти и защита физической памяти.

Немаскируемые прерывания (Non-Maskable Interrupts, NMI)

Немаскируемые прерывания (NMI) используются только при возникновении аппаратных ошибок. При наступлении прерывания они вызывают немедленный переход к обработчику NMI, работающему в M-режиме, независимо от того, как установлен бит разрешения прерывания для этого аппаратного потока. Другими словами, это прерывание будет обрабатываться без возможности его заблокировать. Каждое NMI будет иметь связанный с ним регистр «mcause». Это

позволяет разработчикам самостоятельно решать, как будут обрабатывать эти прерывания, и позволяет им определять множество условий их возникновения. NMI не сбрасывают состояние процессора, что позволяет сообщать о возникновении ошибки с возможностью ее локализации.

Атрибуты физической памяти (Physical Memory Attributes, PMA)

Физическая карта памяти системы включает в себя такие диапазоны адресов, как: доступные области памяти, отображаемые на память управляющие регистры и пустые области в адресном пространстве. Некоторые области памяти могут не поддерживать чтение, запись или хранение программы; некоторые могут не поддерживать доступ к подслову или подблоку; некоторые могут не поддерживать атомарные операции; а некоторые могут не поддерживать когерентность кэша или иметь различные модели памяти. В системах RISC-V эти свойства и возможности каждого региона физического адресного пространства называются атрибутами физической памяти (PMA).

PMA некоторых областей памяти фиксируются во время проектирования процессора – например, для ПЗУ на кристалле. Другие фиксируются во время проектирования системной (материнской) платы, в зависимости, например, от того, какие другие микросхемы подключены к внешним шинам процессора. Некоторые устройства могут быть конфигурируемыми во время работы для поддержки различных применений, которые подразумевают различные PMA. Например, оперативная память на кристалле может использоваться в качестве кэш-памяти одного процессорного ядра для одного приложения, или использоваться как общий некэшируемый блок памяти для другого приложения. Большинство систем требуют, чтобы по крайней мере некоторые атрибуты физической памяти динамически проверялись аппаратно на более поздних этапах конвейера, после того, как будет известен физический адрес (поскольку некоторые операции не будут поддерживаться по всем физическим адресам памяти, а некоторые операции требуют знания текущей настройки конфигурируемого атрибута).

Для RISC-V выделяется спецификация и проверка PMA в отдельную аппаратную структуру – устройство контроля PMA (PMA checker). Во многих случаях атрибуты известны на этапе проектирования системы для каждой области физических адресов и могут быть непосредственно подключены к устройству контроля PMA. Если атрибуты конфигурируются во время выполнения, то могут быть предусмотрены специфические для платформы отображенные на память регистры управления для указания этих атрибутов гранулярности, соответствующей каждой области памяти (например, для статической памяти с произвольным доступом (SRAM) на кристалле, доступ к которой разделен на кэшируемое и некэшируемое использование).

Подробное описание PMA может занять целую главу. Здесь не рассматриваются PMA с упорядочиванием памяти, PMA с идемпотентностью, PMA с когерентностью или PMA с возможностью кэширования. Подробности реализации PMA описаны в разделе 3.5 Привилегированной спецификации [1].

Защита физической памяти (Physical Memory Protection, PMP)

Общей особенностью большинства современных процессоров является возможность выполнения безопасных удаленных вычислений или «доверенная среда исполнения». Примеры этой технологии включают Intel Software Guard Extensions (SGX), AMD Secure Encrypted Virtualization (SEV) и Arm TrustZone. Хотя RISC-V ISA не предоставляет комплексного решения для реализации надежной среды исполнения, возможности защиты физической памяти (PMP) являются прочной основой, на которой можно построить такую систему.

RISC-V PMP ограничивает доступ программному обеспечению по физическим адресам, доступным программному обеспечению в текущем контексте. Дополнительный модуль PMP предоставляет регистры управления машинным режимом для каждого потока, позволяющие задавать привилегии доступа к физической памяти (чтение, запись, выполнение) для каждой области физической памяти. Значения PMP проверяются параллельно с проверками PMA, которые были рассмотрены в предыдущем разделе. Степень детализации настроек управления доступом PMP зависит от платформы и внутри платформы может варьироваться в зависимости от области физической памяти, но стандартное кодирование PMP поддерживает разметку областей памяти размером до четырех байтов. Привилегии определенных регионов могут быть жестко закреплены (например, некоторые регионы могут быть видны только в машинном режиме, но не на уровнях с более низкими привилегиями).

Записи PMP описываются 8-битным регистром конфигурации и одним 32-битным (или 64-битным) адресным регистром. Поддерживается до 16 записей PMP. Если реализованы какие-либо записи PMP, то должны быть реализованы все CSR PMP, но любые поля CSR PMP могут быть установлены в ноль. CSR PMP доступны только в M-режиме.

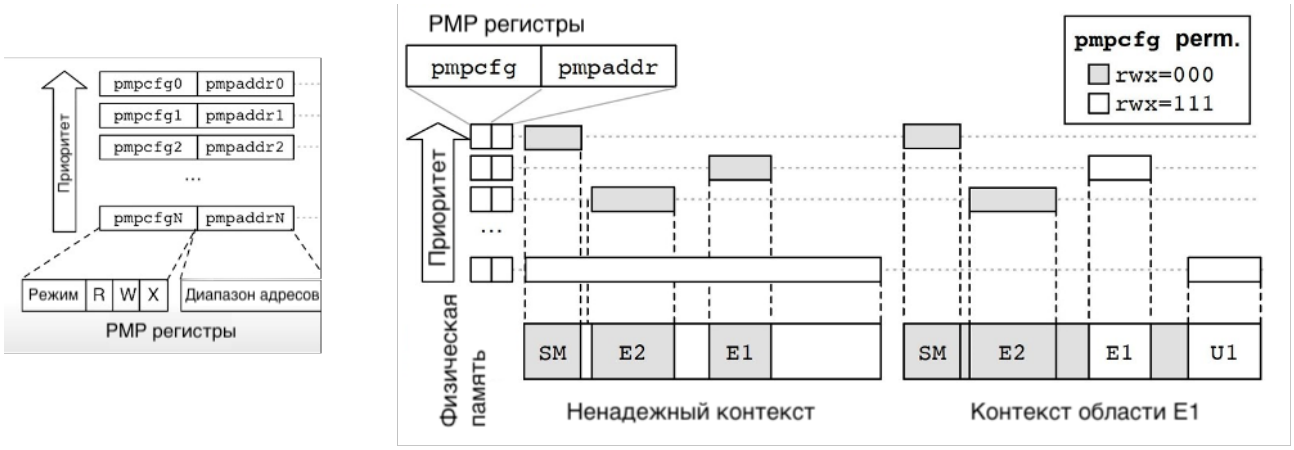

Рисунок 4. Пример настройки разных контекстов [2].

На рисунке 4 показан пример того, как можно настроить два разных контекста: один ненадежный и один с доступом к «Области E1». В этом примере приложение запускается в пользовательском контексте U1. Это приложение имеет доступ только к своей собственной памяти и памяти внутри Области E1. Память внутри Области E2 и в мониторе безопасности (Security Monitor, SM) недоступна пользовательскому приложению. Таким образом, конфиденциальность данных обеспечивается за счет того, что монитор безопасности (работающий в M-режиме) может изменять настройки PMP, разрешая или запрещая доступ к областям памяти в соответствии с конфигурацией PMP.

ISA уровня супервизора (S-Mode) ISA, Version 1.11

В этой главе описывается архитектура уровня супервизора RISC-V, которая содержит общее ядро, используемое с различными схемами трансляции и защиты адресов уровня супервизора. Режим супервизора для поддержки чистой виртуализации намеренно ограничен с точки зрения взаимодействия с базовым физическим оборудованием, таким как физическая память и прерывания устройств. В соответствии с этим, некоторые возможности супервизора, включая запросы на прерывания таймера и межпроцессорные прерывания, предоставляются механизмами, специфичными для конкретной реализации. В некоторых системах исполнительная среда супервизора (Supervisor Execution Environment, SEE) предоставляет эти возможности в виде бинарного интерфейса супервизора (Supervisor Binary Interface, SBI). В других системах эти возможности предоставляются напрямую, через какой-либо другой механизм, определяемый реализацией.

RISC-V поддерживает 32-битную, 39-битную и 48-битную страничную адресацию виртуальной памяти. Для синхронизации обновлений структур данных управления памятью с текущими

выполняемыми командами используется супервизорная инструкция доступа к памяти (SFENCE.VMA). Выполнение этой инструкции гарантирует, что все предыдущие сохранения, уже видимые текущему RISC-V аппаратному потоку (hardware thread), будут упорядочены перед всеми последующими неявными обращениями этого аппаратного потока к структурам данных управления памятью.

Виртуальная память — это довольно сложная концепция, и она выходит за рамки этого курса. Достаточно понимать, что RISC-V поддерживает виртуальную память страничной организации различных размеров, и что существует специальная инструкция S-режима, используемая для синхронизации обновлений между потоками.

Спецификации, не относящиеся к ISA

Целевые группы также могут работать над программным обеспечением или стандартами, которые не являются частью ISA. Например, следующие группы работают над проектами, которые не приводят к созданию новых спецификаций, а скорее к созданию стандартов, стимулирующих сообщество разрабатывать свои продукты на общей основе:

- рабочая группа по отладке, работает над поддержкой и стандартизаций внешней отладки;
- целевая группа по соблюдению требований архитектуры набора команд RISC-V ISA, работает над тестами и фреймворками соответствия;
- целевая группа по структурной конфигурации, работает над тем, как представить структуру конфигурации разработанной аппаратной реализации как в удобном для чтения человеку формате, так и в двоичном формате.

Список источников

1. Specifications: [Электронный ресурс] // RISC-V. URL: https://riscv.org/technical/specifications/.
2. Ли Д., Колбреннер Д., Шинде С., Сонг Д. и Асанович К. – 2019. Keystone: основа для проектирования TEEs. CoRR, том. abs/1907.10119.