ВикипедиЯ

MIPS (архитектура)

Материал из Википедии — свободной энциклопедии

MIPS (сокращение от названия соответствующего проекта Стэнфордского университета англ. *Microprocessor without Interlocked Pipeline Stages, без блокировок в конвейере*[1]) — система команд и микропроцессорных архитектур, разработанных компанией MIPS Computer Systems (в настоящее время Wave Computing[2]) в соответствии с концепцией проектирования процессоров RISC (то есть для процессоров с упрощенным набором команд). Ранние модели процессора имели 32-битное машинное слово, позднее появились его 64-битные версии. Существует множество модификаций процессора, включая MIPS I, MIPS II, MIPS III, MIPS IV, MIPS V, MIPS32 и MIPS64, из них действующими являются MIPS32 (для 32-битной реализации) и MIPS64 (для 64-битной реализации). MIPS32 и MIPS64 определяют как набор регистров управления, так и набор команд.

Существует несколько наборов команд: MIPS32, MIPS64, microMIPS и nanoMIPS. Помимо этого, доступны дополненные наборы инструкций и модели процессоров, например, MIPS-3D, включающий в себя набор SIMD-команд для обработки чисел с плавающей запятой, предназначенный для решения простых 3D-задач, MDMX (MaDMaX) — с ещё более широкими возможностями — набором SIMD-команд и использующий 64-битные регистры с плавающей запятой для работы с целыми числами, MIPS16e, который сжимает поток команд, чтобы уменьшить объём памяти, занимаемый программами, а также MIPS MT, обеспечивающий многопотоковый режим обработки.

Архитектуру MIPS часто изучают в программе курса «Компьютерная архитектура» в университетах и технических лицеях. Эти процессоры значительно повлияли на более поздние RISC-архитектуры, в частности на Alpha.

В настоящее время различные реализации MIPS используются в основном во встроенных системах, например, в смартфонах, маршрутизаторах, шлюзах, а также до 2010-х в игровых консолях, таких, как Sony PlayStation 2 и Sony PlayStation Portable. До конца 2006 года они применялись и в компьютерах SGI. К концу 1980-х и 1990-х эта архитектура широко

MIPS	
Разработчик	MIPS Technologies, Inc.
Разрядность	64 (32→64)
Представлена	1985
Архитектура	RISC
Тип	Регистр-регистр
Кодирование СК	фиксированное
Реализация переходов	по сравнению двух регистров
Порядок байтов	Bi-endian (big→bi)
Расширения	MDMX, MIPS-3D, MIPS16e, MIPS MT
	Регистры
Общего назначения	31 (нулевой регистр всегда хранит 0)
Вещественные	32 (для чисел двойной точности используются пары регистров в 32-битных версиях процессора)

Медиафайлы на Викискладе

использовалась многими компаниями, среди них Digital Equipment Corporation, NEC, Pyramid Technology, Siemens Nixdorf и Tandem Computers. С середины до конца 1990-х годов каждым третьим микропроцессором на рынке был процессор под управлением MIPS.

Содержание

История

Основоположник RISC

В 1981 году коллектив под руководством Джона Хеннесси (John L. Hennessy) из Университета Стэнфорда начал работу над проектом, который получил название MIPS. Главной идеей было увеличить производительность процессора, используя удлиненный конвейер. Концепция применения конвейера в качестве основной технологии была известна ещё задолго до этого (например, в IBM 801), но она не использовала весь свой потенциал. Центральный процессор включает в себя несколько специальных субблоков, таких, как декодеры команд, целочисленное АЛУ (арифметико-логическое устройство), блоки загрузки/хранения (работа с памятью) и т. д. В традиционной не оптимизированной реализации отдельная команда в программе должна быть (почти всегда) завершена, прежде, чем запустится другая; в то время как в конвейерной архитектуре последовательные команды могут выполняться параллельно. Например, когда математическая инструкция вносится в блок с плавающей запятой, блок загрузки/хранения памяти может в этот же момент вызвать следующую команду.

Одним из главных препятствий в использовании конвейера был тот факт, что некоторые команды, такие, как деление, выполняются намного дольше, и, вследствие этого, центральному процессору приходится ждать, прежде, чем передать на конвейер следующую команду. Единственное решение этой проблемы — использовать серию блокировок, позволяющих определенным стадиям конвейера показать, что они заняты и, в этом случае, приостанавливать вышестоящие в потоке команды. Группа Хеннеси рассматривала эти блокировки как огромный барьер в увеличении производительности, поскольку было необходимо обращаться ко всем модулям Центрального процессора, что занимает лишнее время и ограничивает тактовую частоту. Главным аспектом устройства MIPS было согласовать каждую подфазу каждой команды, в том числе кэширование, в один цикл, таким образом избегая необходимости в блокировках и пропуская на конвейер только один цикл.

Хотя такая реализация и исключала бы некоторые очень полезные операции, такие, как умножение и деление, очевидно, что предельная производительность системы значительно увеличилась бы, так как микросхемы смогли бы работать с более высокой тактовой частотой. Достижение высокой скорости с использованием блокировок было бы затруднительным, так как время, требуемое для установки блокировок, пропорционально тактовой частоте, в свою очередь, зависящей от размера кристалла. Вот почему исключение вышеупомянутых операций стало спорным вопросом.

Другое отличие архитектуры MIPS от конкурирующих с ним Berkeley-архитектур — внедренная в Berkeley-RISC возможность обработки вызова подпрограмм. Чтобы увеличить производительность столь общей задачи, в Berkeley-RISC была использована технология, называемая регистровым окном, которая, тем не менее, ограничивала максимальную глубину многоуровневых вызовов. Каждый вызов подпрограммы требовал свой набор регистров, что делало необходимым увеличение их количества. Тогда как аппаратная реализация данного механизма занимала дополнительное пространство в кристалле ЦП. Но Хеннесси полагал, что более «тщательный» компилятор

мог бы найти свободные регистры для передачи параметров функции, и что всего лишь увеличение числа регистров могло бы не только упростить эту задачу, но и увеличить производительность всех операций. Поэтому было принято решение отказаться от данной технологии в MIPS.

Архитектура MIPS была, в некотором отношении, наиболее типичной для RISC. Чтобы сэкономить биты в коде команды, в RISC было уменьшено количество инструкций для кодирования. В MIPS из 32 битов слова всего 6 используются для основного кода, а остальные могут содержать либо единственный 26-битный адрес перехода, либо до 5 полей, устанавливающих от 1 до 3 регистров + длина сдвига регистра. Существует и ряд других форматов, например, когда 2 регистра задаются непосредственно выделенным 16-битным полем и т. д. Такое распределение позволило процессору загружать команду и необходимые ей данные за один машинный такт, в то время как в более старых архитектурах (не являвшихся RISC), например, таких, как MOS Technology 6502, требовались отдельные такты для загрузки основного кода и данных.

Это было одним из главных усовершенствований, наращивающих производительность, предлагаемых RISC. Однако другие архитектуры всё же достигли подобной скорости, но другими средствами (такими, как очереди в ЦП).

Первая аппаратная реализация

В 1984 году, убежденный в коммерческом спросе на свою разработку, Хеннесси покинул Стэнфорд и основал компанию MIPS Computer Systems. В 1985 году вышла первая коммерческая реализация микропроцессора MIPS — R2000, доработанная в 1988 году и получившая название R3000. Эти 32-битные процессоры легли в основу продуктовой линейки компании в 1980-х и использовались преимущественно в SG-сериях рабочих станций. Новые коммерческие проекты не соответствовали Стэнфордским научным исследованиям, так как практически все блокировки выполнялись на аппаратном уровне, к тому же операции умножения и деления были полностью реализованы.

В 1991 году впервые был представлен как 64-битный микропроцессор MIPS — модель R4000. R4000 имеет расширенный TLB, в котором запись содержит не только виртуальный адрес, но и виртуальный идентификатор адресного пространства. Такой буфер устраняет основные проблемы производительности микроядра, достаточно медлительного в конкурирующих архитектурах (Pentium, PowerPC, Alpha) из-за необходимости сбрасывать TLB во время частого переключения контекста.

Тем не менее, у MIPS возникали финансовые трудности в связи с поставкой процессоров на рынок. Проект был настолько важен для SGI (в то время являвшихся одними из немногих основных покупателей MIPS), что в 1992 году SGI выкупили права на компанию с условием гарантии, что конструкция микропроцессоров не изменится. Став дочерней компанией, MIPS Computer Systems получили название MIPS Technologies.

Лицензируемая архитектура

В начале 1990 года MIPS начали лицензирование своих разработок для сторонних производителей. Идее сопутствовала удача из-за простоты ядра, которое находило множество применений, где ранее использовались намного менее эффективные CISC-архитектуры, с тем же количеством и той же ценой схем (2 этих критерия тесно связаны: цена ЦП, как правило, зависит от количества схем и контактов). Компания Sun Microsystems сделала аналогичную попытку лицензировать SPARC-ядра, но Sun подобная удача не сопутствовала. К концу 1990-х MIPS стали наиболее важной компанией в производстве встроенных процессоров, и в 1997 году 48-миллионные поставки процессоров на MIPS-ядрах позволили RISC-архитектурам вытеснить популярное семейство процессоров 68k. MIPS были настолько популярны, что в 1998 году SGI передали часть активов MIPS Technologies. На сегодняшний день половину доходов MIPS дает лицензирование разработок, а большую часть другой половины — контракты на разработку ядер для производства сторонними производителями.

В 1999 году MIPS формализовали свои системы лицензирования вокруг двух основных конструкций — 32-разрядной MIPS32 (на базе MIPS II с некоторыми дополнительными функциями MIPS III, IV MIPS и MIPS V) и 64-разрядных MIPS64 (на базе MIPS V). Лицензия на MIPS64 была приобретена каждой из компаний NEC, Toshiba и SiByte (впоследствии приобретенная Broadcom) сразу же после объявления о её выпуске. Вскоре к ним присоединились Philips, LSI Logic и IDT. Успех следовал за успехом, и сегодня процессоры MIPS являются одним из наиболее востребованных товаров на рынке устройств компьютерного типа (карманных компьютеров, приставок и т. п.), наряду с другими разработчиками, тщетно пытающимися их вытеснить.

Через несколько лет после того, как MIPS-архитектура стала лицензируемой, она начала привлекать все больше и больше новых компаний по разработке процессоров. Первой такой компанией была Quantum Effect Devices (см. следующий раздел). Команда разработчиков, собравших MIPS R4300i, основала компанию SandCraft, предоставившую компании NEC новый процессор R5432, а немного позднее смоделировавшую R71000 — один из первых нестандартных процессоров для рынка встраиваемых систем. Команда основателей компании DEC StrongARM, в конце концов, разделилась на две новых компании по разработке процессоров, в основу которых лег MIPS: SiByte, производившая SB-1250 — одну из первых чиповых систем с высокой производительностью, основанных на MIPS (SOC) и Alchemy Semiconductor (позднее приобретенная AMD), производившая Au-1000 SOC для маломощных приложений. Компания Lexra использовала архитектуру, подобную MIPS, добавив к ней DSP для рынка аудиомикросхем, а также поддержку многопотокового режима для сетевого рынка. Так как Lexra не покупала лицензию на MIPS, вскоре между двумя компаниями разгорелись судебные процессы. Первый был довольно быстро погашен уже после того, как Lexra пообещала не продвигать свои процессоры, как сходные с MIPS. Второй процесс (о патенте MIPS 4814976 на обработку инструкции невыровненного (unaligned) доступа к памяти) был более затяжным и негативно отразился на бизнесе обеих компаний, а по его завершении MIPS Technologies выдали Lexra бесплатную лицензию и выплатили денежную компенсацию в крупном размере.

Следом за этими событиями на рынке появились две компании, специализирующиеся на создании многоядерных устройств, использующих архитектуру MIPS. Корпорация Raza Microelectronics выкупили производственную линию у менее успешных SandCraft, а затем начала выпускать восьмиядерные устройства для рынка телекоммуникаций и сетей. Cavium Networks, изначально являвшиеся поставщиком средств защиты процессоров, тоже начали производство восьми-, а позже и 32-ядерных архитектур для тех же рынков. Обе компании сами проектировали ядра, и лишь лицензировали разработки вместо того, чтобы покупать готовые проекты процессоров MIPS.

Потеря рынка ПК

Среди производителей, создавших рабочие станции с использованием микропроцессоров MIPS, — такие компании, как SGI, MIPS Computer Systems, Inc., Whitechapel Workstations, Olivetti, Siemens-Nixdorf, Acer, Digital Equipment Corporation, NEC, и DeskStation. В числе операционных систем, портированных на архитектуру MIPS: IRIX компании SGI, Windows NT (до версии 4.0) компании Microsoft, Windows CE, Linux, UNIX (System V и BSD), SINIX, QNX, и операционная система RISC OS, непосредственно принадлежащая компании MIPS Computer Systems.

В начале 1990-х существовало предположение, что MIPS вместе с другими мощными процессорами RISC вскоре обгонят архитектуру IA32 компании Intel. Этому способствовала поддержка двух первых версий Windows NT для Alpha, MIPS и PowerPC компании Microsoft, и, в несколько меньшей степени, — архитектуры Clipper и SPARC. Однако, как только Intel выпустил новейшие версии ЦП семейства Pentium, Microsoft Windows NT v4.0 перестала поддерживать все, кроме Alpha и Intel. После решения SGI перейти на архитектуры Itanium и IA32 процессоры MIPS практически полностью перестали использоваться в персональных компьютерах.

Рынок встраиваемых систем

В 1990-е годы MIPS-архитектура была широко распространена на рынке встраиваемых систем: для сетей, телекоммуникаций, видеоигр, игровых консолей, принтеров, цифровых приставок, цифровых телевизоров, xDSL и кабельных модемов, а также карманных компьютеров.

Ingenic JZ4725 — пример SOC, базированной на MIPS

Низкое энергопотребление и температурные характеристики встраиваемых MIPS-архитектур, широкие возможности внутренних функций делают этот микропроцессор универсальным для многих устройств.

Синтезируемые ядра для рынка встраиваемых систем

В последние годы большинство технологий, используемых в различных поколениях MIPS, предложены в виде IP-ядер (стандартных блоков) для встраиваемых реализаций процессора. Более того, предложены оба типа ядер — основанные на 32 и 64 битах, известные как 4K и 6K. Такие ядра могут совмещаться с другими структурными элементами, такими, как FPU, системами SIMD, различными устройствами ввода-вывода и т. д.

Некогда коммерчески успешные ядра MIPS и в настоящее время нашли потребительское и промышленное применение. Эти ядра можно найти в новых маршрутизаторах Cisco, Linksys, ZyXEL и MikroTik, кабельных и ADSL-модемах, смарт-картах, механизмах лазерных принтеров, цифровых приставках, роботах, карманных компьютерах, Sony PlayStation 2 и Sony PlayStation Portable. Тем не менее, в приложениях мобильных телефонов и PDA MIPS не удалось сместить прочно установившуюся там конкурирующую ARM-архитектуру.

Процессоры под управлением MIPS включают в себя: IDT RC32438; ATI Xilleon; Alchemy Au1000, 1100, 1200; Broadcom Sentry5; RMI XLR7xx, Cavium Octeon CN30xx, CN31xx, CN36xx, CN38xx and CN5xxx; Infineon Technologies EasyPort, Amazon, Danube, ADM5120, WildPass, INCA-IP, INCA-IP2; Microchip Technology PIC32; NEC EMMA and EMMA2, NEC VR4181A, VR4121, VR4122, VR4181A, VR5432, VR5500; Oak Technologies Generation; PMC-Sierra RM11200; QuickLogic QuickMIPS ESP; Toshiba Donau, Toshiba TMPR492x, TX4925, TX9956, TX7901; KOMDIV-32, KOMDIV-64; Мультикор.

Суперкомпьютеры MIPS

Одним из наиболее интересных применений архитектуры MIPS является их использование в многопроцессорных вычислительных суперкомпьютерах. В начале 1990-х компания Silicon Graphics (SGI) перенаправила свой бизнес с графических терминалов на рынок высокопроизводительного вычисления. Успех первых попыток компании в области серверных систем (а именно, серия Challenge, основанная на R4400, R8000 и R10000) мотивировал SGI создать гораздо более мощную систему. Использование R10000 позволило компании спроектировать систему Origin 2000, в конечном счете расширяемую до 1024 ЦП, используя собственную межсистемную связь cc-NUMA (NUMAlink). Позже Origin 2000 породила новую систему — Origin 3000, вышедшую с теми же максимальными 1024 ЦП, но использовавшую в разработке микросхемы R14000 и R16000 с частотой до 700 МГц. Однако, в 2005 году, когда SGI приняла стратегическое решение о переходе на архитектуру Intel IA-64, суперкомпьютеры, базированные на MIPS, были сняты с производства.

В 2007 году корпорация SiCortex представила новый многопроцессорный персональный суперкомпьютер, основанный на архитектуре MIPS. В его разработку легли MIPS64 и высокопроизводительная межсистемная связь с использованием топологии графов Кауца (англ. *Kautz graph*). Данная система является предельно эффективной и вычислительно мощной. Её уникальный аспект — многоядерный узел обработки, интегрирующий шесть ядер MIPS64, коммутатор контроллера памяти, межсистемную связь механизмов прямого доступа к памяти, локальную сеть с пропускной способностью 1 Гбит и PCI Express контроллеры. И все это на одном кристалле, который потребляет 10 Вт энергии, но выполняет максимум 6 миллиардов операций с плавающей запятой в секунду. Самая мощная конфигурация такого суперкомпьютера — версия SC5832, состоящая из 972 узлов (всего 5832 ядер MIPS64) и выполняющая 8,2 триллионов операций с плавающей запятой в секунду.

Loongson возвращает к персонализации

Компания Loongson, в надежде обойти патент MIPS, выпустила свою архитектуру, которая была полностью схожа с разработкой MIPS Technologies и поддерживалась ОС Linux. Так как производство процессоров Loongson было более дешёвым, MIPS получили возможность возродиться на рынке персональных компьютеров в лице Loongson. (В дальнейшем Loongson купили лицензию на MIPS — см. основную

статью)

Процессоры под управлением MIPS также используются в нетбуках компаний iUnika, Bestlink, Lemote и Golden Delicious Computers.

MIPS IV

MIPS IV — четвёртое поколение архитектуры, представляет собой расширенную версию MIPS III и совместим со всеми существующими моделями MIPS. Первая реализация MIPS IV была представлена в 1994 году под названием R8000. MIPS IV включил в себя:

- Простой регистр + регистр адресации для загрузки и хранения чисел с плавающей запятой
- Операции FMA и FMS одинарной и двойной точности для чисел с плавающей запятой
- Команды условного перехода для целых чисел и для чисел с плавающей запятой
- Дополнительные условные биты в регистре контроля и состояния числа с плавающей запятой: в общей сложности 8 битов.

MIPS V

MIPS V — пятая версия архитектуры, была представлена 21 октября 1996 года на Микропроцессорном форуме 1996 года. Эта модель была разработана для того, чтобы повысить производительность графических 3D-приложений. В середине 1990-х большая часть не встроенных микропроцессоров MIPS приходилась на графические терминалы от SGI. Разработка MIPS V была дополнена целочисленными мультимедийными расширениями MDMX (MIPS Digital Media Extensions), которые были представлены в тот же день, что и MIPS V.

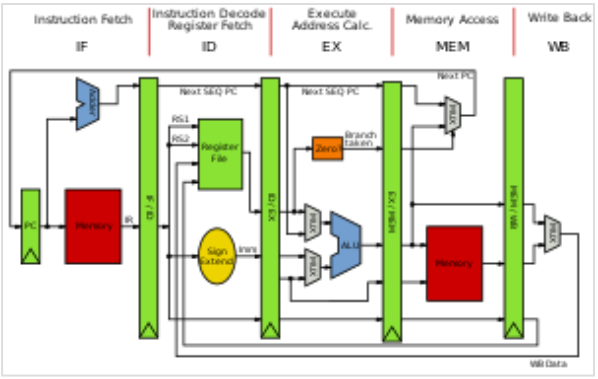
Реализации MIPS V так никогда и не были внедрены. В 1997 году SGI представила микропроцессоры под названиями «H1» («Beast») и «H2» («Capitan»), которые должны были быть произведены в 1999 году. Но вскоре их объединили, и в конечном итоге в 1998 году эти проекты были отменены.

В MIPS V был добавлен новый тип данных — PS (pair-single), который представляет собой два числа с плавающей запятой двойной точности (32-битные), хранящиеся в 64-битном регистре с плавающей запятой. Чтобы работать с этим типом данных в режиме SIMD, были добавлены различные варианты арифметических, сравнительных операций над числами с плавающей запятой, а также команда условного перехода. Появились новые инструкции для загрузки, реконфигурации и преобразования PS-данных. Это первая архитектура, сумевшая реализовать обработку чисел с плавающей запятой в SIMD-режиме с имеющимися ресурсами.

Семейство процессоров с архитектурой MIPS

Первым коммерческим микропроцессором с архитектурой MIPS был микропроцессор R2000, представленный в 1985 году. В нём были реализованы операции умножения и деления, которые выполнялись за несколько тактов. Устройство умножения и деления не было тесно интегрировано в ядро процессора, хотя и размещалось на том же кристалле; по этой причине система команд расширена инструкциями

для загрузки результатов умножения и деления в регистры общего назначения, эти инструкции блокировали конвейер.



Конвейер MIPS, проходящий пять стадий (получение инструкции, декодирование, исполнение, доступ к памяти и вывод)

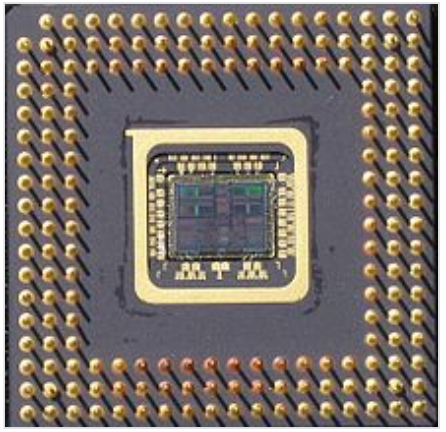
Микропроцессор R2000 мог быть загружен как в режиме big-endian, так и в режиме little-endian, содержал 32 32-разрядных регистра общего назначения. Подобно процессорам AMD 29000 и Alpha, микропроцессор R2000 не имел отдельного регистра флагов условий, так как разработчики посчитали его потенциальным «узким местом». Счётчик команд непосредственно недоступен.

Микропроцессор R2000 поддерживал подключение до четырёх сопроцессоров, один из которых является встроенным и обеспечивает работу с исключениями, а также управление памятью (MMU). В случае необходимости в качестве ещё одного сопроцессора можно было подключить микросхему R2010, арифметический сопроцессор, который содержал тридцать два 32-разрядных регистра, которые можно было использовать как шестнадцать 64-разрядных регистров для работы с числами двойной точности. Следующим в семействе стал R3000, который появился в 1988 году. Он содержал кэш-память данных объёмом 64 КБ (R2000 — 32 КБ). Кроме того, R3000 обеспечивал когерентность кэш-памяти при работе в мультипроцессорных конфигурациях. Несмотря на то, что в поддержке мультипроцессорности R3000 имеется ряд недостатков, на базе R3000 было создано несколько работоспособных многопроцессорных систем. Как и для R2000, для R3000 был создан арифметический сопроцессор в виде отдельной СБИС: R3010. Микропроцессор R3000 стал первым коммерчески успешным процессором с архитектурой MIPS, было изготовлено более миллиона процессоров. Ускоренная версия R3000, работающая на тактовой частоте 40 МГц, названная R3000A, достигла производительности в 32 VUPs (VAX Unit of Performance). Дальнейшее развитие R3000A, микропроцессор R3051, работающий на частоте 33,8688 МГц, был использован в игровой приставке Sony PlayStation. Другие производители также представили процессоры, совместимые с R3000A: в Performance Semiconductor был разработан R3400, в то время как компания IDT создала R3500, оба упомянутых процессора имели в интегрированный математический сопроцессор R3010. Первой системой на кристалле, использующей процессор с архитектурой MIPS, стала разработка R3900 фирмы Toshiba; данная микросхема использовалась в портативном компьютере, работавшем под управлением Windows CE. Был разработан радиационно-устойчивый вариант R3000 с интегрированным R3010, предназначенный для применения в космических аппаратах, который получил название Mongoose-V.

Серия R4000, выпущенная в 1991 году, расширила процессоры MIPS до 64 битов. (MIPS Technology была первой компанией, выпустившей процессоры с 64-битовой архитектурой) R4000 состоит из 1,3 млн транзисторов, имеет встроенный кэш данных и кэш инструкций (оба по 8 Кб). В этом процессоре внешняя тактовая частота 50 МГц удваивается, а внутренняя тактовая частота составляет 100 МГц. Процессор R4400 выполнен на основе R4000, состоит из 2,2 млн транзисторов, имеет встроенный кэш данных и кэш инструкций (оба по 16 Кб), а внутренняя тактовая частота составляет 150 МГц. Набор команд этих процессоров (спецификация MIPS II) был расширен командами загрузки и записи 64-разрядных чисел с плавающей запятой, командами вычисления квадратного корня с одинарной и двойной точностью, командами условных прерываний, а также атомарными операциями, необходимыми для поддержки мультипроцессорных конфигураций. В процессорах R4000 и R4400 реализованы 64-битовые шины данных и 64-битовые регистры.

MIPS, теперь являющийся отделом SGI под названием MTI, разработал недорогие процессоры R4200, послужившие основой для будущих (ещё более дешёвых) R4300i. Производная этого процессора, NEC VR4300, использовалась в игровых консолях Nintendo 64.

Quantum Effect Devices (QED), самостоятельная компания, основанная разработчиками MIPS, разработала серию процессоров R4600 Orion, R4700 Orion, R4650 и R5000. Если в R4000 увеличили тактовую частоту, но пожертвовали количеством кэш-памяти, то QED уделили большое внимание и ёмкости кэш-памяти (доступ к которой можно получить всего за 2 цикла), и эффективному использованию поверхности кристалла. Процессоры R4600 и R4700 использовались в недорогих версиях рабочей станции SGI Indy, а также в первых маршрутизаторах Cisco (основанных на MIPS), например, серии 36x0 и 7x00. Микропроцессор R4650 применялся в телевизионных приставках WebTV (в настоящее время — Microsoft TV). В процессоре R5000 FPU диспетчеризация операций с плавающей запятой (одинарной точности) была более гибкой, чем в R4000, и, вследствие этого, рабочие станции SGI Indys, базированные на R5000 отличались лучшей графической производительностью, чем R4400 с такой же тактовой скоростью и графическим аппаратным устройством. Чтобы подчеркнуть улучшение после объединения R5000 и старой графической платы, SGI дала ей новое название. Немного позднее QED разработали семейство процессоров RM7000 и RM9000 для рынка сетей и лазерных принтеров. В Августе 2000 года компания QED была приобретена производителем полупроводников PMC-Sierra, и последняя продолжила инвестирование MIPS-архитектур. Процессор RM7000 включал в себя 256 Кб встроенной кэш-памяти 2го уровня и контроллер для дополнительной кэш-памяти 3го уровня. Были созданы процессоры RM9xx0 — семейство SOC-устройств, в которые включены такие периферийные составляющие (на северном мосту), как: контроллер памяти, PCI-контроллер, контроллер Ethernet, а также быстрые устройства ввода-вывода (например, высокопроизводительная шина типа HyperTransport).



нижняя сторона R4700 Orion с удаленной защитной крышкой, на которой видно кремниевый чип, изготовленный IDT и спроектированный Quantum Effect Devices

R8000 (представлен в 1994 году) был первой суперскалярной архитектурой MIPS, способной осуществлять 2 целочисленные инструкции (или с плавающей запятой) и 2 инструкции обращения к памяти за один цикл. Данная разработка использовала 6 схем: устройство для целочисленных команд (16 Кб — команды и 16 Кб — кэш данных), для команд с плавающей запятой, три вторичных дескриптора кэш-памяти ОЗУ (два для вторичного доступа к кэш-памяти + один для отслеживания шины), а также кэш-контроллер ASIC. Архитектура имеет два полностью конвейеризованных устройства умножения-сложения (с двойной точностью), которые могут передавать поток данных в 4 Мб внекристального вторичного кэша. В середине 1990-х процессоры R8000 запустили SGI серверы POWER Challenge, а позже стали доступны на рабочих станциях POWER Indigo2. Хотя производительность этого FPU и была наиболее подходящей для научных сотрудников, ограниченность его целочисленной производительности и высокая цена не смогли привлечь большинство пользователей, поэтому R8000 был на рынке всего год, и даже сейчас его едва ли можно найти.

лицевая сторона R4700 Orion

В 1995 году был выпущен R10000[3]. Этот процессор предлагался в однокристальном исполнении, работал с более высокой тактовой частотой, чем R8000, а также включал в себя объемную (32 КБ) первичную кэш-память данных и команд. Кроме того, он был суперскалярным, но это главное новшество было неисправно. Но даже с более простым FPU, значительно увеличенная производительность целочисленный вычислений, более низкая цена и высокая плотность записи сделали R10000 предпочтительным для большинства пользователей.

Все более поздние проекты были основаны на ядре R10000. В R12000 был использован 0.25 микронный технологический процесс с целью уменьшить чип и достигнуть большей тактовой скорости. Исправленный R14000 имел более высокую тактовую частоту в дополнение с поддержкой DDR SRAM для внекристальной кэш-памяти. Следом были выпущены R16000 и R16000A, тактовая частота которых была также увеличена; в них была встроена дополнительная кэш-память первого уровня, а их производство требовало более мелких кристаллов, чем прежде.

Среди других представителей семейства MIPS — R6000, ЭСЛ-реализация, выполненная компанией Bipolar Integrated Technology. R6000 относится к поколению процессоров MIPS II. Его TLB и устройство кэш-памяти значительно отличаются от остальных представителей данного семейства. R6000 не принес обещанной выгоды, и, хотя был признан в некоторой степени полезным для компьютеров Control Data, он мгновенно исчез с основного рынка.

Микропроцессоры MIPS

Модель	Частота (МГц)	Год	Технология разработки (μm)	Транзисторы (млн.)	Размер кристалла (мм²)	Число выводов	Мощность (Вт)	Напряжение (В)	Кэш данных (КБ)	Кэш инструкций (КБ)	Кэш 2го уровня	Кэш 3го уровня
R2000	8—16.67	1985	2.0	0.11	?	?	?	?	32	64	НЕТ	НЕТ
R3000	12—40	1988	1.2	0.11	66.12	145	4	?	64	64	0-256 Кб Внешняя	НЕТ
R4000	100	1991	0.8	1.35	213	179	15	5	8	8	1 Мб Внешняя	НЕТ
R4400	100—250	1992	0.6	2.3	186	179	15	5	16	16	1-4 Мб Внешняя	НЕТ
R4600	100—133	1994	0.64	2.2	77	179	4.6	5	16	16	512 Кб Внешняя	НЕТ
R4700	133	1996	?	?	?	179	?	?	16	16	Внешняя	НЕТ
R5000	150—200	1996	0.35	3.7	84	223	10	3.3	32	32	1 Мб Внешняя	НЕТ
R8000	75—90	1994	0.7	2.6	299	591+591	30	3.3	16	16	4 Мб Внешняя	НЕТ
R10000	150—250	1996	0.35, 0.25	6.7	299	599	30	3.3	32	32	512 Кб—16 Мб Внешняя	НЕТ
R12000	270—400	1998	0.25, 0.18	6.9	204	600	20	4	32	32	512 Кб—16 Мб Внешняя	НЕТ
RM7000	250—600	1998	0.25, 0.18, 0.13	18	91	304	10, 6, 3	3.3, 2.5, 1.5	16	16	256 Кб Внутренняя	1 Мб Внешняя
R14000	500—600	2001	0.13	7.2	204	527	17	?	32	32	512 Кб—16 Мб Внешняя	НЕТ
R16000	700—1000	2002	0.11	?	?	?	20	?	64	64	512 Кб—16 Мб Внешняя	НЕТ
R24K	750+	2003	65 nm	?	0.83	?	?	?	64	64	4-16 Мб Внешняя	НЕТ

Формат инструкций MIPS I

Инструкции делятся на три типа: R, I и J. Каждая инструкция начинается с 6-битного кода. В дополнение к коду, инструкции R-типа определяют три регистра, область размера сдвига регистра, и область функции; инструкции I-типа определяют два регистра и непосредственное значение; инструкции J-типа состоят из кода операции и 26-битного адреса перехода.

Далее приведена таблица применения трех форматов инструкции в архитектуре ядра:

Тип	Формат инструкций					
	31..26	**25..21**	**20..16**	**15..11**	**10..6**	**5..0**
R	Код (*6 бит*)	rs (*5 бит*)	rt (*5 бит*)	rd (*5 бит*)	степень сдвига (shamt, от англ. *shift amount*) (*5 бит*)	функция (*6 бит*)
I	Код (*6 бит*)	rs (*5 бит*)	rt (*5 бит*)	Постоянное или непосредственное значение (англ. *immediate value*) (*16 бит*)		
J	Код (*6 бит*)	Адрес (*26 бит*)				

Язык ассемблера MIPS

Данные инструкции языка ассемблера имеют прямую аппаратную реализацию, в отличие от псевдоинструкций, которые перед сборкой транслируются в настоящие составные инструкции.

- Далее регистровые буквы d, t, и s будут обозначать указатели на номера и имена регистров.
- Буква C обозначает константу.
- Все последующие команды являются собственными.
- Все коды операций и функций представлены в шестнадцатеричной системе счисления.
- Руководство «Набор инструкций MIPS32» предупреждает, что слово «беззнаковый (unsigned)», используемое в описании инструкций сложения и вычитания является вводящим в заблуждение. Разница между знаковыми и беззнаковыми такими инструкциями заключается лишь в генерации исключения при обнаружении переполнения (в случае команд «со знаком»), или же игнорирование переполнения для «беззнаковых» (переполнением, как понятно, здесь считается именно переполнение для чисел со знаком). Операнд константа, в соответствии с этими инструкциями, всегда должен иметь знак.

Целочисленные операции

MIPS имеет 32 регистра для целочисленных операций. Для выполнения арифметических вычислений данные должны находиться в регистрах. Регистр $0 всегда хранит 0, а регистр $1 резервируется для сборки (для хранения псевдоинструкций и больших констант). Нижеприведенная таблица показывает некоторые из базовых инструкций MIPS:

Категория	Название	Синтаксис инструкции	Значение	Формат/код/функция			Примечания
Арифметическая	Add	add $d,$s,$t	$d = $s + $t	R	0	20_{16}	Складывает два регистра, выполняет прерывание при переполнении.
	Add unsigned	addu $d,$s,$t	$d = $s + $t	R	0	21_{16}	Эквивалентна предыдущей, но игнорирует переполнение.
	Subtract	sub $d,$s,$t	$d = $s − $t	R	0	22_{16}	Вычитает два регистра, выполняет прерывание при переполнении.
	Subtract unsigned	subu $d,$s,$t	$d = $s − $t	R	0	23_{16}	Эквивалентна предыдущей, но игнорирует переполнение.
	Add immediate	addi $t,$s,C	$t = $s + C (знаковое)	I	8_{16}	-	Выполняет сложение регистра $s и константы C, результат записывается в регистр $t, выполняет прерывание при переполнении.
	Add immediate unsigned	addiu $t,$s,C	$t = $s + C (знаковое)	I	9_{16}	-	Эквивалентна предыдущей, но игнорирует переполнение, C остается знаковым.
	Multiply	mult $s,$t	LO = (($s * $t) << 32) >> 32; HI = ($s * $t) >> 32;	R	0	18_{16}	Умножает два регистра и записывает 64-битный результат в два специальных поля для памяти — LO and HI. Аналогично можно записать результат операции в виде: (int HI,int LO) = (64-bit) $s * $t. См. mfhi и mflo для доступа к LO и HI регистрам.
	Divide	div $s, $t	LO = $s / $t; HI = $s % $t	R	0	$1A_{16}$	Делит один регистр на другой и записывает 32-битный результат в LO, а остаток в HI[4].
	Divide unsigned	divu $s, $t	LO = $s / $t; HI = $s % $t	R	0	$1B_{16}$	Делит один регистр на другой и записывает 32-битный результат в LO, а остаток — в HI.
Передача данных	Load double word	ld $t,C($s)	$t = Memory[$s + C]	I	23_{16}	-	Загружает двойное слово из: MEM[$s+C] и следующих 7 байтов в $t и следующий регистр.

Load word	lw $t,C($s)	$t = Memory[$s + C]	I	23_{16}	-	Загружает слово из: MEM[$s+C] и следующих 3 байтов.
Load halfword	lh $t,C($s)	$t = Memory[$s + C] (знаковое)	I	21_{16}	-	Загружает половину слова из: MEM[$s+C] и следующего байта. Знак расширен до ширины регистра.
Load halfword unsigned	lhu $t,C($s)	$t = Memory[$s + C] (беззнаковое)	I	25_{16}	-	Эквивалентна предыдущей, но без расширения знака.
Load byte	lb $t,C($s)	$t = Memory[$s + C] (знаковое)	I	20_{16}	-	Загружает один байт из: MEM[$s+C] и расширяет знак.
Load byte unsigned	lbu $t,C($s)	$t = Memory[$s + C] (беззнаковое)	I	24_{16}	-	Эквивалентна предыдущей, но без расширения знака.
Store double word	sd $t,C($s)	Memory[$s + C] = $t	I		-	Сохраняет два слова: из $t и следующего регистра в MEM[$s+C] и следующие 7 байтов соответственно. Порядок операндов может создать путаницу.
Store word	sw $t,C($s)	Memory[$s + C] = $t	I	$2B_{16}$	-	Сохраняет слово в: MEM[$s+C] и следующие 3 байта. Порядок операндов может создать путаницу.
Store half	sh $t,C($s)	Memory[$s + C] = $t	I	29_{16}	-	Сохраняет первую половину регистра (halfword) в: MEM[$s+C] и следующий байт.
Store byte	sb $t,C($s)	Memory[$s + C] = $t	I	28_{16}	-	Сохраняет первую четверть регистра (byte) в: MEM[$s+C].
Load upper immediate	lui $t,C	$t = C << 16	I	F_{16}	-	Загружает 16-битный операнд в вышестоящие 16 битов определенного регистра. Максимальная величина константы $2^{16}-1$
Move from high	mfhi $d	$d = HI	R	0	10_{16}	Помещает значение из HI в регистр. Не используйте инструкции умножения и деления внутри инструкции mfhi (это действие не определено из-за конвейера MIPS).
Move from low	mflo $d	$d = LO	R	0	12_{16}	Помещает значение из LO в регистр. Не используйте инструкции умножения и деления внутри инструкции mflo (это действие не определено из-за конвейера MIPS).
Move from Control Register	mfcZ $t, $s	$t = Coprocessor[Z].ControlRegister[$s]	R	0		Перемещает 4-байтовое значение из сопроцессора регистра Z-контроля в регистр общего назначения. Расширение знака.

	Move to Control Register	mtcZ $t, $s	Coprocessor[Z].ControlRegister[$s] = $t	R	0		Перемещает 4-байтовое значение из регистра общего назначения в сопроцессор регистра Z-контроля. Расширение знака.
Логическая	And	and $d,$s,$t	$d = $s & $t	R	0	24_{16}	Побитовая конъюнкция регистров $s и $d, результат записывается в регистр $d.
	And immediate	andi $t,$s,C	$t = $s & C	I	C_{16}	-	Побитовая конъюнкция регистра $s с константой C, результат записывается в регистр $t.
	Or	or $d,$s,$t	$d = $s | $t	R	0	25_{16}	Побитовая дизъюнкция регистров $s и $d, результат записывается в регистр $d.
	Or immediate	ori $t,$s,C	$t = $s | C	I	D_{16}	-	Побитовая дизъюнкция регистра $s с константой C, результат записывается в регистр $t.
	Exclusive or	xor $d,$s,$t	$d = $s ^ $t	R	0	26_{16}	
	Nor	nor $d,$s,$t	$d = ~ ($s | $t)	R	0	27_{16}	Побитовая логическая операция NOR (Стрелка Пирса)
	Set on less than	slt $d,$s,$t	$d = ($s < $t)	R	0	$2A_{16}$	Проверяет, является ли один регистр меньше другого.
	Set on less than immediate	slti $t,$s,C	$t = ($s < C)	I	A_{16}	-	Проверяет, является ли один регистр меньше константы.
Битовый сдвиг	Shift left logical	sll $t,$s,C	$t = $s << C	R	0	0	Логический сдвиг влево. Сдвигает на C битов влево (эквивалентно умножению числа на 2^C)
	Shift right logical	srl $t,$s,C	$t = $s >> C	R	0	2_{16}	Логический сдвиг вправо. Сдвигает на C битов вправо (эквивалентно делению числа на 2^C). Заметьте, что эта инструкция работает как деление в дополнительным двоичном коде, только если значение положительно.
	Shift right arithmetic	sra $t,$s,C	Если принять числа за беззнаковые, то операцию можно записать следующим образом: $t = \lfloor \frac{s}{2^C} \rfloor + \left(\left(\sum_{n=1}^{C} 2^{32-n} \right) \lfloor \frac{s}{2^{31}} \rfloor \right)$	R	0	3_{16}	Арифметический сдвиг вправо. Сдвигает на C битов — (эквивалентно делению числа на 2^C даже для отрицательных чисел в дополнительном коде)

Условное ветвление	Branch on equal	beq $s,$t,C	if ($s == $t) then goto PC+4+(4*C)	I	4_{16}	-	Переходит к инструкции по указанному адресу, если два регистра равны.
	Branch on not equal	bne $s,$t,C	if ($s != $t) then goto PC+4+(4*C)	I	5_{16}	-	Переходит к инструкции по указанному адресу, если два регистра не равны.
Безусловный переход	Jump	j C	PC = (PC+4[31:28] | C*4)	J	2_{16}	-	Выполняет безусловный переход к инструкции по указанному адресу.
	Jump register	jr $s	goto address $s	R	0	8_{16}	Переходит по адресу, содержащемуся в указанном регистре.
	Jump and link	jal C	$31 = PC + 8; PC = (PC+4[31:28] | C*4)	J	3_{16}	-	Используется для вызовов подпрограмм Инструкция записывает значение PC + 8 в регистр $ra ($31), выполняет следующую после нее инструкцию, и совершает переход по нужному адресу. Такое поведение является следствием использования слота задержки перехода (англ. *delay slot*). До появления MIPS32/MIPS64, выполнение переходов предварялось выполнением следующей за ней команды, а уже потом происходил переход. Это позволяло более равномерно нагрузить конвейер — пока команда перехода извлекала инструкцию по адресу перехода процессор не простаивал, а исполнял следующую за ней команду, что, однако, влечет за собой сложность чтения и правильного понимания ассемблерного кода и требует учета компиляторами.

Стоит отметить несколько особенностей:

1. В коде языка ассемблера MIPS, смещение для ветвящихся инструкций может быть представлено маркировками в другом месте кода.

2. В MIPS не существует команды «*копирование в регистр, начиная с первых битов*» (англ. *load lower immediate*). Это можно сделать с помощью инструкций addi (*add immediate*) или ori (*or immediate*) с регистром $0. Например, обе команды addi $1, $0, 0xFF и ori $1, $0, 0xFF загружают в регистр $1 значение FF_{16}. При этом, стоит отметить, что инструкция addi расширит знак константы, т. е. например addi $1, $0, 0xFFFF приведет к тому, что $1 = $FFFFFFFF_{16} = (-1)_{10}$.

Операции над числами с плавающей запятой

MIPS имеет 32 регистра с плавающей запятой. Регистры соединены по 2 для двойной точности вычислений. Регистры с нечетными номерами не могут быть использованы для арифметических операций или ветвления, они могут лишь частично указывать двойную точность в паре регистров.

Категория	Название	Синтаксис инструкции	Значение	Формат/код/функция			Примечания/Кодирование
Арифметические	FP add single	add.s $x,$y,$z	$x = $y + $z				Сложение чисел с плавающей запятой (одинарная точность)
	FP subtract single	sub.s $x,$y,$z	$x = $y - $z				Вычитание чисел с плавающей запятой (одинарная точность)
	FP multiply single	mul.s $x,$y,$z	$x = $y * $z				Умножение чисел с плавающей запятой (одинарная точность)
	FP divide single	div.s $x,$y,$z	$x = $y / $z				Деление чисел с плавающей запятой (одинарная точность)
	FP add double	add.d $x,$y,$z	$x = $y + $z				Сложение чисел с плавающей запятой (двойная точность)
	FP subtract double	sub.d $x,$y,$z	$x = $y - $z				Вычитание чисел с плавающей запятой (двойная точность)
	FP multiply double	mul.d $x,$y,$z	$x = $y * $z				Умножение чисел с плавающей запятой (двойная точность)
	FP divide double	div.d $x,$y,$z	$x = $y / $z				Деление чисел с плавающей запятой(двойная точность)
Передача данных	Load word coprocessor	lwcZ $x,CONST ($y)	Coprocessor[Z].DataRegister[$x] = Memory[$y + CONST]	I			Загружает 4 байта типа word из: MEM[$2+CONST] в регистр данных сопроцессора. Расширение знака.
	Store word coprocessor	swcZ $x,CONST ($y)	Memory[$y + CONST] = Coprocessor[Z].DataRegister[$x]	I			Записывает 4 байта из регистра данных сопроцессора в MEM[$2+CONST]. Расширение знака.
Логические	FP compare single (eq, ne, lt, le, gt, ge)	c.lt.s $f2,$f4	if ($f2 < $f4) cond=1; else cond=0				Сравнение на меньшее команд с плавающей запятой. Одинарная точность.
	FP compare double (eq, ne, lt, le, gt, ge)	c.lt.d $f2,$f4	if ($f2 < $f4) cond=1; else cond=0				Сравнение на меньшее команд с плавающей запятой. Двойная точность.
Ветвление	branch on FP true	bc1t 100	if (cond == 1) go to PC+4+100				если формат FP, выполняется ветвление.
	branch on FP false	bc1f 100	if (cond == 0) go to PC+4+100				если формат не FP, выполняется ветвление.

Псевдоинструкции

Эти инструкции принимаются языком ассемблера MIPS, однако они не являются реальными. Ассемблер переводит их в последовательности настоящих инструкций.

Название	Синтаксис инструкции	Трансляция в обычные инструкции	значение
Load Address (с англ. *Загрузить адрес*)	la $1, LabelAddr	lui $1, LabelAddr[31:16] ori $1, $1, LabelAddr[15:0]	$1 = Маркировка адреса
Load Immediate	li $1, IMMED[31:0]	lui $1, IMMED[31:16] ori $1, $1, IMMED[15:0]	$1 = 32-битное прямое значение
Branch if greater than (с англ. *Совершить ветвление если больше чем*)	bgt $rs, $rt, Label	slt $at, $rt, $rs bne $at, $zero, Label	if(R[rs]>R[rt]) then PC=Label
Branch if less than (с англ. *Совершить ветвление если меньше чем*)	blt $rs, $rt, Label	slt $at, $rs, $rt bne $at, $zero, Label	if(R[rs]<R[rt]) then PC=Label
Branch if greater than or equal (с англ. *Совершить ветвление если больше или равно*)	bge $rs, $rt, Label	slt $at, $rs, $rt beq $at, $zero, Label	if(R[rs]>=R[rt]) then PC=Label
Branch if less than or equal (с англ. *Совершить ветвление если меньше или равно*)	ble $rs, $rt, Label	slt $at, $rt, $rs beq $at, $zero, Label	if(R[rs]<=R[rt]) then PC=Label
Branch if greater than unsigned (с англ. *Совершить ветвление если больше чем, не учитывая знак*)	bgtu $rs, $rt, Label	sltu $at, $rt, $rs bne $at, $zero, Label	if(R[rs]=>R[rt]) then PC=Label
Branch if greater than zero (с англ. *Совершить ветвление если больше нуля)*	bgtz $rs, Label		if(R[rs]>0) then PC=Label
Multiply and return only first 32 bits	mul $1, $2, $3	mult $2, $3	$1 = $2 * $3

(с англ. *Умножить и вернуть только первые 32 бита*)		mflo $1	

Несколько других важных инструкций

- NOP (без операции) (машинный код 0x00000000, интерпретируется процессором как sll $0, $0, 0)
- BREAK (разрывы программы, используется отладчиками)
- Системный вызов (используется для системных вызовов операционной системы)

Использование регистра транслирования

Аппаратная архитектура определяет следующие критерии:

- Регистр общего назначения $0 всегда возвращает значение 0.
- Регистр общего назначения $31 используется в качестве регистра-ссылки для команд перехода и связи.
- HI и LO используются для доступа к результатам умножения/деления, доступ к которым осуществляется командами mfhi (move from high) и mflo (move from low).

Это единственные ограничения, которые аппаратная архитектура накладывает на использование регистров общего назначения.

Различные устройства MIPS реализовывают специальные соглашения о вызовах, которые ограничивают использование регистров. Соглашения о вызовах полностью поддерживаются комплексом ПО, но не требуются аппаратным обеспечением.

Регистры

Название	Номер	Применение	нужно ли резервировать?
$zero	$0	всегда хранит 0	N/A
$at	$1	временный регистр для языка ассемблера	НЕТ
$v0—$v1	$2—$3	значения функций и выражений	НЕТ
$a0—$a3	$4—$7	аргументы функций	НЕТ
$t0—$t7	$8—$15	временные	НЕТ
$s0—$s7	$16—$23	сохраненные временные значения	ДА
$t8—$t9	$24—$25	временные	НЕТ
$k0—$k1	$26—$27	зарезервирована для ядра операционной системы	НЕТ
$gp	$28	глобальный указатель	ДА
$sp	$29	указатель стека	ДА
$fp	$30	указатель фрейма	ДА
$ra	$31	адрес возврата	N/A

Защищенные регистры (по соглашению) не могут быть изменены вызовом системы или процедуры (функции). Например, $s-регистры должны быть сохранены в стеке процедурой, которая собирается ими воспользоваться; к $sp и $fp-регистрам приращиваются константы, а по окончании процедуры регистры вновь уменьшаются. Противоположным примером служит регистр $ra, который автоматически меняется при его вызове любой функцией. $t-регистры должны сохраняться программой перед вызовом любой процедуры (если программе нужны данные, полученные после вызова).

Эмуляторы

Среди Open Virtual Platforms существует бесплатный эмулятор OVP-sim, доступный для некоммерческого использования, который представляет собой библиотеку моделей процессоров и платформ, а также интерфейсов API, позволяющих пользователю проектировать свои собственные модели. Библиотека моделей является открытым ресурсом, написанным на языке C, и включает в себя ядра MIPS 4K, 24K и 34K. Данные модели созданы и поддерживаются компанией Imperas, которая в сотрудничестве с MIPS Technologies протестировала эмулятор и отметила его знаком MIPS-Verified. Образцы платформ, основанных на MIPS, включают в себя как само металлическое оборудование, так и платформы для загрузки немодифицированных двоичных отображений Linux. Такие платформы-эмуляторы эффективны для обучения, а также доступны, бесплатны и просты в использовании. OVPsim, разработанный и поддерживаемый Imperas, работает с высокой скоростью (сотни миллионов инструкций с секунду), и применим для описания многоядерных архитектур.

Существует свободно доступный эмулятор MIPS32 (ранние версии могли имитировать только R2000/R3000), выпущенный под названием SPIM и предназначенный для использования в обучении. EduMIPS64 — это межплатформенный графический эмулятор процессора MIPS64, написанный на языке Java/Swing. Он поддерживает множество MIPS64 ISA и позволяет пользователю наглядно увидеть, что происходит в конвейере, когда ЦП выполняет программу на языке ассемблера. Проект имеет строго образовательные цели и широко используется на некоторых курсах компьютерной архитектуры во всем мире.

Ещё один GUI-эмулятор процессоров MIPS — это MARS, тоже разработанный в образовательных целях, особенно эффективен вкупе с книгой Хеннесси *Computer Organization and Design*.

Более продвинутые версии бесплатных эмуляторов — Gxemul (ранее известные как проекты mips64emul), а также проекты QEMU. Они имитируют различные модели микропроцессоров MIPS III и MIPS IV (в качестве дополнения к компьютерным системам, их использующим).

Коммерческие разработки эмуляторов доступны в основном для встроенного использования процессоров MIPS, например, Virtutech Simics (MIPS 4Kc и 5Kc, PMC RM9000, QED RM7000), VaST Systems (R3000, R4000), и CoWare (MIPS4KE, MIPS24K, MIPS25Kf и MIPS34K).

Список процессоров на базе архитектуры MIPS по компаниям

- Alchemy Au1000, 1100, 1200
- Atheros AR23xx, AR52xx, AR71xx, AR72xx, AR93xx
- ATI Xilleon
- Broadcom Sentry5
- Байкал Электроникс Baikal-T1
- IDT RC32438
- Infineon Technologies EasyPort, Amazon, Danube, ADM5120, WildPass, INCA-IP, INCA-IP2
- Ingenic JZ47, M
- Godson-3B1500
- Lemote Loongson I, Loongson II, Loongson III[5]
- Microchip Technology PIC32
- NEC EMMA and EMMA2, NEC VR4181A, VR4121, VR4122, VR4181A, VR5432, VR5500
- НИИСИ РАН KOMDIV-32, KOMDIV-64
- Oak Technologies Generation
- PMC-Sierra RM11200
- Realtek RTD1055, RTD1185, RTD1186[6]
- Sigma Designs SMP8640, SMP8650, SMP8910[7]

- QuickLogic QuickMIPS ESP
- RMI XLR7xx, Cavium Octeon CN30xx, CN31xx, CN36xx, CN38xx и CN5xxx
- Toshiba Donau, Toshiba TMPR492x, TX4925, TX9956, TX7901.

Интересные факты

- В декабре 2018 года корпорация Wave запустила инициативу MIPS Open, в рамках которой бесплатно предоставляла лицензию на разработку ядер MIPS. Инициатива существовала чуть менее года и была закрыта в ноябре 2019 года[8].
- Популярный PSP от Sony основан на MIPS R4000 с тактовой частотой от 1 до 333 мГц.

См. также

- ARM – семейство лицензируемых 32-битных и 64-битных микропроцессорных ядер разработки компании ARM Limited
- OpenRISC — свободная архитектура 2000 года с GPL реализацией or1k
- LEON — свободные реализации (GPL, LGPL) архитектуры SPARC V8, появившиеся в 1997 году
- OpenSPARC — свободная (GPL) реализация архитектуры SPARC V9 от 2005 года
- OpenPOWER — коллаборация вокруг архитектуры IBM Power, основанная в 2013 году IBM, Google, Mellanox, NVIDIA
- RISC-V — свободная и открытая архитектура и система команд для микропроцессоров и микроконтроллеров, созданная в 2010 году

Примечания

1. Эволюция и современное состояние архитектуры MIPS - Время электроники (http://www.russianelectronics.ru/leader-r/review/2192/doc/40351/). Дата обращения: 26 марта 2019. Архивировано (https://web.archive.org/web/20190327155121/http://www.russianelectronics.ru/leader-r/review/2192/doc/40351/) 27 марта 2019 года.
2. AI Pioneer Wave Computing Acquires MIPS Technologies (https://www.forbes.com/sites/moorinsights/2018/06/13/ai-pioneer-wave-computing-acquires-mips-technologies/#5673109f1990). Дата обращения: 26 марта 2019. Архивировано (https://web.archive.org/web/20190327090513/https://www.forbes.com/sites/moorinsights/2018/06/13/ai-pioneer-wave-computing-acquires-mips-technologies/#5673109f1990) 27 марта 2019 года.
3. В. Аваков, Микропроцессор MIPS R10000 (https://www.osp.ru/os/1995/06/178765/) Архивная копия (https://web.archive.org/web/20180925200639/https://www.osp.ru/os/1995/06/178765/) от 25 сентября 2018 на Wayback Machine — Открытые системы. СУБД 1995 № 06
4. MIPS R3000 Instruction Set Summary (https://web.archive.org/web/20180628185213/http://www.mrc.uidaho.edu/mrc/people/jff/digital/MIPSir.html). Дата обращения: 1 июня 2010. Архивировано из оригинала (http://www.mrc.uidaho.edu/mrc/people/jff/digital/MIPSir.html) 28 июня 2018 года.
5. Loongson cpu_CPU & Motherboard_江苏龙芯梦兰科技股份有限公司 (http://www.lemote.com/en/products/cpu/2010/0310/113.html) Архивировано (https://web.archive.org/web/20100516212151/http://www.lemote.com/en/products/cpu/2010/0310/113.html) 16 мая 2010 года.

6. Digital Media Processor (Realtek) (http://www.realtek.com/products/productsView.aspx?Langid=1&PNid=9&PFid=26&Level=3&Conn=2). Дата обращения: 21 декабря 2011. Архивировано (https://web.archive.org/web/20120101142746/http://www.realtek.com/products/productsView.aspx?Langid=1&PNid=9&PFid=26&Level=3&Conn=2) 1 января 2012 года.
7. Secure Media Processor™ Overview (http://www.sigmadesigns.com/media_processor_overview.php) Архивировано (https://web.archive.org/web/20111219024752/http://www.sigmadesigns.com/media_processor_overview.php) 19 декабря 2011 года.
8. Wave Computing Closes Its MIPS Open Initiative with Immediate Effect, Zero Warning (https://www.hackster.io/news/wave-computing-closes-its-mips-open-initiative-with-immediate-effect-zero-warning-e88b0df9acd0) (англ.). *Hackster.io* (7 ноября 2019). Дата обращения: 7 декабря 2019. Архивировано (https://web.archive.org/web/20210307041333/https://www.hackster.io/news/wave-computing-closes-its-mips-open-initiative-with-immediate-effect-zero-warning-e88b0df9acd0) 7 марта 2021 года.

Литература

- *David A. Patterson; John L. Hennessy.* Computer Organization and Design: The Hardware/Software Interface (англ.). — Morgan Kaufmann Publishers. — ISBN 1-55860-604-1.
- *Dominic Sweetman.* See MIPS Run, 2nd edition (неопр.). — Morgan Kaufmann Publishers. — ISBN 0-12088-421-6.
- *Dominic Sweetman.* See MIPS Run (неопр.). — Morgan Kaufmann Publishers. — ISBN 1-55860-410-3.
- *Erin Farquhar; Philip Bunce.* MIPS Programmer's Handbook (неопр.). — Morgan Kaufmann Publishers. — ISBN 1-55860-297-6.

Ссылки

- Patterson & Hennessy — Appendix A (http://www.cs.wisc.edu/~larus/HP_AppA.pdf)
- Summary of MIPS assembly language (https://web.archive.org/web/20070526082617/http://logos.cs.uic.edu/366/notes/MIPS%20Quick%20Tutorial.htm)
- MIPS Instruction reference (https://web.archive.org/web/20180628185213/http://www.mrc.uidaho.edu/mrc/people/jff/digital/MIPSir.html)
- MARS (MIPS Assembler and Runtime Simulator) (http://courses.missouristate.edu/KenVollmar/MARS/)
- MIPS processor images and descriptions at cpu-collection.de (http://www.cpu-collection.de/?tn=1&l0=cl&l1=MIPS%20Rx000)
- A programmed introduction to MIPS assembly (https://web.archive.org/web/20090305154304/http://chortle.ccsu.edu/AssemblyTutorial/TutorialContents.html)
- Mips bitshift operators (http://www.cs.umd.edu/class/spring2003/cmsc311/Notes/Mips/bitshift.html)
- MIPS software user’s manual (http://www.it.uu.se/edu/course/homepage/datsystDV/ht04/Project/tools/machinedata/4KcProgMan.pdf)

Источник — https://ru.wikipedia.org/w/index.php?title=MIPS_(архитектура)&oldid=134195268