ВикипедиЯ

RISC

Материал из Википедии — свободной энциклопедии

RISC (англ. *reduced instruction set computer*[1] — вычислитель с набором упрощённых/редуцированных команд) — архитектурный подход к проектированию процессоров, в которой быстродействие увеличивается за счёт такого кодирования инструкций, чтобы их декодирование было более простым, а время выполнения — меньшим. В системах команд первых RISC-процессоров даже отсутствовали команды умножения и деления. Это также облегчает повышение тактовой частоты и делает более эффективной суперскалярность (распараллеливание инструкций между несколькими исполнительными блоками).

RISC-микропроцессор UltraSPARC компании Sun

Содержание

Отличия от CISC

Наборы инструкций в более ранних архитектурах для облегчения ручного написания программ на языках ассемблеров или прямо в машинных кодах, а также для упрощения реализации компиляторов, выполняли как можно больше работы. Нередко в наборы включались инструкции для прямой поддержки конструкций языков высокого уровня. Другая особенность этих наборов — большинство инструкций, как правило, допускали все возможные методы адресации (т. н. «ортогональность системы команд»). К примеру: и операнды, и результат в арифметических операциях доступны не только в регистрах, но и

через непосредственную адресацию, и прямо в оперативной памяти. Позднее такие архитектуры были названы CISC (англ. *Complex instruction set computer (Компьютер с комплексным набором команд)*).

Однако многие компиляторы не задействовали все возможности таких наборов инструкций, а на сложные методы адресации уходит много времени из-за дополнительных обращений к сравнительно медлительной оперативной памяти. Было выяснено, что такие функции быстрее выполняются последовательностью более простых машинных команд, если при этом процессор упрощается и в нём остаётся место для большего числа регистров, за счёт которых можно сократить количество обращений к оперативной памяти. В первых архитектурах, причисляемых к RISC, большинство инструкций для упрощения декодирования имеет одинаковую длину и похожую структуру, арифметические операции работают только с регистрами, а работа с памятью идёт через отдельные команды инструкции загрузки (load) и сохранения (store). Эти свойства и позволили лучше сбалансировать этапы конвейеризации, сделав конвейеры в RISC значительно более эффективными и позволив поднять тактовую частоту.

Философия RISC

В середине 1970-х разные исследователи (в частности в IBM) выяснили, что большинство комбинаций инструкций и ортогональных методов адресации не используются в большинстве программ, генерируемых компиляторами того времени. Также было выявлено, что в некоторых архитектурах с микрокодной реализацией выполняемые одной машинной командой сложные операции были медленнее, чем алгоритмически эквивалентные последовательности более простых операций. Это было вызвано, в частности, тем, что многие архитектуры разрабатывались в спешке и хорошо оптимизировался микрокод только тех машинных инструкций, которые использовались чаще.[2]

Поскольку многие реальные программы тратят большую часть своего времени на выполнение простых операций, многие исследователи решили сфокусироваться на том, чтобы сделать эти операции максимально быстрыми. Производительность процессора ограничена временем, которое процессор тратит на выполнение наиболее медленных операций в процессе обработки любой инструкции. Уменьшение длительности таких шагов даёт общее повышение производительности, а также зачастую ускоряет выполнение инструкций за счёт более эффективной конвейеризации.[3] Фокусирование на простых инструкциях и ведёт к архитектуре RISC, цель которой — сделать инструкции настолько простыми, чтобы они легко конвейеризировались и тратили не более одного машинного такта на каждом шаге конвейера на высоких частотах.

Позднее было отмечено, что наиболее значимая характеристика RISC в разделении инструкций для обработки данных и обращения к памяти — обращение к памяти идёт только через инструкции load и store, а все прочие инструкции ограничены работой с аппаратными регистрами процессора. Это упростило архитектуру процессоров:

- позволило придать машинным командам фиксированную длину,
- упростило конвейеры и изолировало логику, имеющую в качестве побочного эффекта задержки при доступе к оперативной памяти, только двумя инструкциями.

В итоге RISC-архитектуры стали неформально называть также *архитектурами load/store*.[4]

Количество инструкций

Нередко слова «сокращённый набор команд» понимаются как минимизация количества инструкций в системе команд. В действительности, машинных команд у многих RISC-процессоров больше, чем у CISC-процессоров.[5][6] В системах команд некоторых RISC-процессоров вроде транспьютеров фирмы INMOS машинных команд не меньше, чем, например, в системах команд CISC-процессоров IBM System/370, и, наоборот, — CISC-процессор DEC PDP-8 имеет только 8 основных и несколько расширенных инструкций.

На самом деле, термин «сокращённый» в названии описывает тот факт, что сокращён объём (и время) работы, выполняемый каждой отдельной машинной командой — как максимум один цикл доступа к памяти, — тогда как сложные инструкции CISC-процессоров могут требовать сотен циклов доступа к памяти для своего выполнения.[7]

Некоторые архитектуры, специально разработанные для минимизации количества инструкций, сильно отличаются от классических RISC-архитектур и получили иное названия: Minimal instruction set computer (MISC), Zero instruction set computer (ZISC), Ultimate RISC (также называемый OISC), Transport triggered architecture (TTA) и т. п.

Характерные особенности RISC-процессоров

- Фиксированная длина машинных инструкций (например, 32 бита) и простой формат команды.
- Специализированные команды для операций с памятью — чтения или записи. Операции вида Read-Modify-Write («прочитать-изменить-записать») отсутствуют. Любые операции «изменить» выполняются только над содержимым регистров (т. н. архитектура load-and-store).
- Большое количество регистров общего назначения (32 и более).
- Отсутствие поддержки операций вида «изменить» над укороченными типами данных — байт, 16-разрядное слово. Так, например, система команд DEC Alpha содержала только операции над 64-разрядными словами, и требовала разработки и последующего вызова процедур для выполнения операций над байтами, 16- и 32-разрядными словами.
- Отсутствие микропрограмм внутри самого процессора. То, что в CISC-процессоре исполняется микропрограммами, в RISC-процессоре исполняется как обыкновенный (хотя и помещённый в специальное хранилище) машинный код, не отличающийся принципиально от кода ядра ОС и приложений. Так, например, обработка отказов страниц в DEC Alpha и интерпретация таблиц страниц содержалась в так называемом PALcode (Privileged Architecture Library), помещённом в ПЗУ. Заменой PALCode можно было превратить процессор Alpha из 64-разрядного в 32-разрядный, а также изменить порядок байтов в слове и формат входов таблиц страниц виртуальной памяти.

Другие архитектуры

За годы после появления архитектуры RISC были реализованы и другие альтернативы — например, VLIW, MISC, OISC, массово-параллельная обработка, систолическая матрица (англ. *Systolic array*), переконфигурируемые вычисления (англ. *Reconfigurable computing*), потоковая архитектура.

- Суперскалярные архитектуры (первоначально — большие ЭВМ конца 1960-х годов, в микропроцессорах — Sun SPARC, начиная с Pentium использованы в семействе x86).

Распараллеливание исполнения команд между несколькими устройствами исполнения, причём решение о параллельном исполнении двух или более команд принимается аппаратурой процессора на этапе исполнения. Эффективное использование такой архитектуры требует специальной оптимизации машинного кода в компиляторе для генерации пар независимых команд (когда результат одной команды не является аргументом другой).

- Архитектуры VLIW (very long instruction word — очень длинное слово команды). Отличаются от суперскалярной архитектуры тем, что решение о распараллеливании принимается не аппаратурой на этапе исполнения, а компилятором на этапе генерации кода. Команды очень длинны и содержат явные инструкции по распараллеливанию нескольких субкоманд на несколько устройств исполнения. Элементы архитектуры содержались в серии PA-RISC. Процессорами с архитектурой VLIW в её классическом виде являются процессоры с архитектурой Эльбрус 2000. Процессор Itanium использует архитектуру EPIC, основанную на VLIW. Разработка эффективного компилятора для VLIW является сложнейшей задачей. Преимущество VLIW перед суперскалярной архитектурой заключается в том, что компилятор может быть более развитым, нежели устройства управления процессора, и он способен хранить больше контекстной информации для принятия более верных решений по оптимизации.

Иные архитектурные решения, типичные для RISC

- Спекулятивное исполнение. При встрече с командой условного перехода процессор исполняет (или, по крайней мере, читает в кэш инструкций) сразу обе ветви до тех пор, пока не окончится вычисление управляющего выражения перехода. Позволяет отказаться от простоев конвейера при условных переходах.
- Переименование регистров. Каждый регистр процессора на самом деле представляет собой несколько параллельных регистров, хранящих несколько версий значения. Используется для реализации спекулятивного исполнения.

Начало развития архитектуры «RISC»

Первая система, которая может быть названа системой «RISC», — суперкомпьютер «CDC 6600», который был создан в 1964 году, за десять лет до появления соответствующего термина. CDC 6600 имел архитектуру «RISC» всего с двумя режимами адресации («регистр+регистр» и «регистр+непосредственное значение») и 74 кодами команд (тогда как 8086 имел 400 кодов команд). В «CDC 6600» было 11 конвейерных устройств арифметической и логической обработки, а также пять устройств загрузки и два устройства хранения. Память была многоблочной, поэтому все устройства загрузки-хранения могли работать одновременно. Базовая тактовая частота/частота выдачи команд была в 10 раз выше, чем время доступа к памяти. Джим Торнтон и Сеймур Крэй, разработчики «CDC 6600», создали для него мощный процессор, позволявший быстро обрабатывать большие объёмы цифровых данных. Главный процессор поддерживался десятью простыми периферийными процессорами, выполнявшими операции ввода-вывода и другие функции ОС.[8] Позднее появилась шутка, что термин «RISC» на самом деле расшифровывается как «Really invented by Seymour Cray» («на самом деле придуман Сеймуром Крэем»).

Ещё одна ранняя машина с архитектурой «RISC» — мини-компьютер «Data General Nova», разработанный в 1968 году.

Первая попытка создать процессор с архитектурой «RISC» на микросхеме была предпринята «IBM» в 1975 году. Эта работа привела к созданию семейства процессоров «IBM 801», которые широко использовались в различных устройствах «IBM». 801-й, в конце концов, был выпущен в форме микросхемы под именем «ROMP» в 1981 году. «ROMP» расшифровывается как «Research OPD (Office Product Division) Micro Processor», то

есть «исследовательский микропроцессор», разработанный в подразделении офисных разработок. Как следует из названия, процессор был разработан для «мини»-задач, и когда в 1986 году «IBM» выпустила на его основе компьютер «IBM RT-PC», он работал не слишком хорошо. Однако за выпуском 801-го процессора последовало несколько исследовательских проектов, в результате одного из которых появилась система «POWER».

RISC в Беркли

Проект «RISC» в Университете Беркли был начат в 1980 году под руководством Дэвида Паттерсона и Карло Секвина. Исследования основывались на использовании конвейерной обработки и агрессивном использовании техники регистрового окна. В обычном процессоре имеется небольшое количество регистров, и программа может использовать любой регистр в любое время. В процессоре, использующем технологии регистрового окна, очень большое количество регистров (например, 128), но программы могут использовать ограниченное количество (например, только 8 в каждый момент времени).

Программа, ограниченная лишь восемью регистрами для каждой процедуры, может выполнять очень быстрые вызовы процедур: «окно» просто сдвигается к 8-регистровому блоку нужной процедуры, а при возврате из процедуры сдвигается обратно, к регистрам вызвавшей процедуры (в обычном процессоре большинство процедур при вызове вынуждено сохранять значения некоторых регистров в стеке для того, чтобы пользоваться этими регистрами при исполнении процедуры. При возврате из процедуры значения регистров восстанавливаются из стека).

Проект «RISC» произвёл на свет процессор «RISC-I» в 1982 году. В нём было 44 420 транзисторов (для сравнения: в процессорах «CISC» того времени их было около 100 тыс.). «RISC-I» имел всего 32 инструкции, но превосходил по скорости работы любой одночиповый процессор того времени. Через год, в 1983 году, был выпущен «RISC-II», который состоял из 40 760 транзисторов, использовал 39 инструкций и работал в три раза быстрее «RISC-I». Проект RISC-Беркли оказал влияние на RISC-процессоры семейства SPARC и DEC Alpha.

RISC в Стэнфорде

Практически в то же время, в 1981 году, Джон Хеннесси начал аналогичный проект, названный «архитектура „MIPS“» в Стэнфордском университете. Создатель «MIPS» практически полностью сосредоточился на конвейерной обработке — попытался «выжать всё» из этой технологии. Конвейерную обработку применяли и в других процессорах, некоторые идеи, которые появились в MIPS, позволили разработанному процессору работать значительно быстрее подобных. Самое важное требование было таким: любая инструкция процессора занимает один такт. Так конвейер смог гораздо быстрее передавать данные, и процессор стал значительно быстрее работать. К сожалению, ради этого требования удалили из набора инструкций такие полезные операции, как умножение или деление.

В первые годы попытки развития архитектуры «RISC» были хорошо известны, однако оставались в рамках породивших их университетских исследовательских лабораторий. Многие в компьютерной отрасли считали, что преимущества процессоров «RISC» не проявятся при использовании в реальных продуктах из-за низкой эффективности

использования памяти в составных инструкциях. Однако с 1986 года исследовательские проекты «RISC» начали выпускать первые работающие изделия. RISC-процессор из Стэнфорда был реализован в процессорах семейства Rxxxx компании MIPS Technologies.

Последние годы

Как оказалось в начале 1990-х годов, RISC-архитектуры позволяют получить большую производительность, чем CISC, за счёт использования суперскалярного и VLIW-подхода, а также за счёт возможности серьёзного повышения тактовой частоты и упрощения кристалла с высвобождением площади под кэш, достигающий огромных ёмкостей. Также RISC-архитектуры позволили сильно снизить энергопотребление процессора за счёт уменьшения числа транзисторов.

Первое время RISC-архитектуры с трудом принимались рынком из-за отсутствия программного обеспечения для них. Эта проблема была решена переносом UNIX-подобных операционных систем (SunOS) на RISC-архитектуры.

В настоящее время многие архитектуры процессоров являются RISC-подобными, к примеру, ARM, DEC Alpha, SPARC, AVR, MIPS, POWER и PowerPC. Наиболее широко используемые в настольных компьютерах процессоры архитектуры x86 ранее являлись CISC-процессорами, однако новые процессоры, начиная с Intel Pentium Pro (1995 г.), являются CISC-процессорами с RISC-ядром[9]. Они непосредственно перед исполнением преобразуют CISC-инструкции x86-процессоров в более простой набор внутренних инструкций RISC.

После того, как процессоры архитектуры x86 были переведены на суперскалярную RISC-архитектуру, можно сказать, что большинство существующих ныне процессоров основано на архитектуре RISC.

См. также

- Усовершенствованные RISC-вычисления
- Система команд RISC-V
- Архитектура POWER
- Архитектура ARM

Примечания

1. Толковый словарь по вычислительным системам = Dictionary of Computing / Под ред. В. Иллингуорта и др.: Пер. с англ. А. К. Белоцкого и др.; Под ред. Е. К. Масловского. — М.: Машиностроение, 1990. — 560 с. — 70 000 (доп.) экз. — ISBN 5-217-00617-X (СССР), ISBN 0-19-853913-4 (Великобритания).
2. Примером является инструкция `INDEX` в архитектуре VAX, которая медленнее эквивалентной реализации, использующей более простые операции. См.: *D. A. Patterson, D. R. Ditzel.* The case for the reduced instruction set computing // SIGARCH Comput. Archit. News. — октябрь 1980. — Вып. 8, 6. — P. 25—33. — doi:10.1145/641914.641917 (https://dx.doi.org/10.1145%2F641914.641917).
3. *Andrew Schulman.* Microprocessors From the Programmer's Perspective (http://www.ddj.com/architect/184408418) // Dr. Dobb's Journal. — 1 сентября 1990. Архивировано (https://web.archive.org/web/20090427041820/http://www.ddj.com/architect/184408418) 27 апреля 2009 года.
4. *Kevin Dowd.* High Performance Computing (https://archive.org/details/highperformancec00dowd). — O'Reilly & Associates, 1993.

5. *Jon «Hannibal» Stokes.* RISC and CISC, Side by Side? (https://arstechnica.com/cpu/4q99/risc-cisc/rvc-5.html#Branch) *RISC vs. CISC: the Post-RISC Era*. Ars Technica (август 1999). Дата обращения: 11 июля 2010. Архивировано (https://web.archive.org/web/20100729204956/http://arstechnica.com/cpu/4q99/risc-cisc/rvc-5.html#Branch) 29 июля 2010 года.
6. *Lloyd Borrett.* RISC versus CISC (https://www.webcitation.org/61AAxuiyc?url=http://www.borrett.id.au/computing/art-1991-06-02.htm). Australian Personal Computer (июнь 1991). Дата обращения: 11 июля 2010. Архивировано из оригинала (http://www.borrett.id.au/computing/art-1991-06-02.htm) 23 августа 2011 года.
7. *Sivarama P. Dandamudi.* Chapter 3: RISC Principles // Guide to RISC Processors for Programmers and Engineers (http://www.springerlink.com/content/u5t457g61q637v66/). — Springer New York, 2005. — P. 39—44. — ISBN 978-0-387-21017-9 (Print) ISBN 978-0-387-27446-1 (Online). — doi:10.1007/0-387-27446-4_3 (https://dx.doi.org/10.1007%2F0-387-27446-4_3). (недоступная ссылка) doi:10.1007/0-387-27446-4_3 (https://dx.doi.org/10.1007%2F0-387-27446-4_3) — «the main goal was not to reduce the number of instructions, but the complexity»
8. Grishman, Ralph. Assembly Language Programming for the Control Data 6000 Series. Algorithmics Press. 1974. P. 12
9. Устройство процессора — "Все о Hi-Tech" (http://all-ht.ru/inf/pc/cp_struct.html). Дата обращения: 11 августа 2015. Архивировано (https://web.archive.org/web/20150812015213/http://all-ht.ru/inf/pc/cp_struct.html) 12 августа 2015 года.

Ссылки

-
- RISC (http://curlie.org/World/Russian/Компьютеры/Аппаратное_обеспечение/Процессоры/RISC/) в каталоге ссылок Curlie (dmoz)

Источник — https://ru.wikipedia.org/w/index.php?title=RISC&oldid=133829308