Игорь Афонин, Дмитрий Кабачник

Современные процессорные архитектуры

Процессорная архитектура

В ходе развития компьютерных технологий были разработаны различные вычислительные системы. Многие из них забыты, а влияние некоторых было весьма значимым. Наметились стратегические тенденции в развитии вычислительной техники и сформировались компьютерные архитектуры. На текущий момент существует несколько основных архитектур и значительное количество процессоров на их основе.

Процессорную архитектуру можно трактовать как комбинацию вычислительной архитектуры и её реализацию в процессоре (в кремнии), то есть рассматривать в аспекте программирования и аппаратно-технических (и технологических) решений. Нужно отметить, что кардинальное отличие архитектур и их несовместимость обнаруживаются именно на уровне машинного кодирования или низкоуровневого программирования (ассемблирования).

С программной точки зрения, процессорная архитектура определяет набор регистров, команд, их структуру и способ выполнения, в результате чего, с одной стороны, программы, собранные для процессоров одной архитектуры, могут выполняться практически на всех процессорах одинаковой (или подобной) архитектуры, а с другой – не смогут работать на процессорах иной архитектуры. Для работы на разных платформах производители программного обеспечения вынуждены выпускать специально скомпилированные (или портированные – перенесённые) для них версии. Примером может служить операционная система Ubuntu Server (ядро Linux), для которой производитель, компания Canonical, кроме основной версии для архитектуры Intel x86 (AMD64), выпустила версии для архитектур ARM, IBM Power и s390x [1]. Также в качестве примера можно привести компанию Microsoft, которая изначально распространяла операционную систему Windows исключительно для архитектуры x86, но с недавнего времени, следуя требованиям рынка и отрасли, объявила о сотрудничестве с компанией Qualcomm [2] и выпустила версию операционной системы Windows 10, работающую на устройствах с процессорами архитектуры ARM (Qualcomm Snapdragon 835) [3]. Из российских ОС следует отметить многоплатформенную операционную систему Astra Linux Special Edition компании АО «НПО РусБИТех», которая существует в версиях для архитектур x86-64 (релиз «Смоленск»), ARM (релиз «Новороссийск»), MIPS (релиз «Севастополь»), IBM System Z (релиз «Мурманск»), POWER (релиз «Керчь») и «Эльбрус» (релиз «Ленинград») [4].

С аппаратной точки зрения, архитектура процессора – это набор составных частей, компонентов и технологий, присущих линейке процессоров. Аппаратная часть постоянно совершенствуется, как по микроархитектуре, так и по технологическому процессу. Выпускаются новые поколения процессоров с целью увеличения производительности и функциональности. Так, на рынке существуют процессоры Intel нескольких поколений: Coffee Lake (восьмое поколение), Kaby Lake (седьмое поколение), Skylake (шестое поколение) и другие. Несмотря на смену микроархитектуры (аппаратной части), они остаются программной архитектурой x86, и на них работает всё ранее написанное для этой архитектуры программное обеспечение, за некоторым исключением, если разработчик ПО использовал недокументированные методы, вызовы и процедуры.

Поэтому, с точки зрения практического применения процессоров, основной является программная архитектура. На текущий момент актуальные и распространённые архитектуры – это CISC, RISC, VLIW.

Архитектура CISC

Первоначально почти все производители первых микропроцессоров использовали архитектуру с расширенным набором команд – CISC (Complex Instruction Set Computer). Причина этого в том, что разработчики пытались уменьшить так называемый семантический разрыв между тем, что компьютеры способны делать, и тем, что требуют языки программирования высокого уровня, пытаясь заменить одной инструкцией многочисленные машинные коды. Также в то время на рынке коммерческих вычислений доминировали мини-компьютеры PDP компании DEC и мейнфреймы компании IBM, которые были основаны на архитектуре CISC. Среди микропроцессоров типичными представителями данной архитектуры стали процессоры компании Intel. На начальном этапе развития микропроцессоров (семидесятые годы прошлого века) были и другие процессоры подобной архитектуры производства компаний Motorola, Zilog, MOS Technology и т.д. Но именно благодаря коммерческой привлекательности микропроцессоров Intel эта архитектура стала самой популярной на текущий момент и практически единственной для персональных компьютеров. Даже компания Apple в своих компьютерах Apple Macintosh в итоге перешла от процессоров PowerPC к процессорам Intel.

Первый процессор Intel, обозначивший начало эпохи микропроцессоров, – микросхема Intel 4004 (рис. 1) – появился в 1971 году [5]. Это был первый коммерческий процессор, реа-

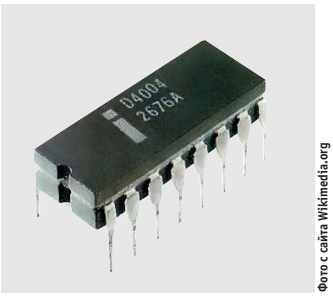

Фото с сайта Wikimedia.org

Рис. 1. Первый микропроцессор Intel

Фото с сайта Wikimedia.org

Рис. 2. Первый 32-битный микропроцессор Intel

лизованный в одной микросхеме. Следует отметить, что сотрудники Intel не догадывались, какое грандиозное открытие они совершили. Эта микросхема вызвала большой интерес и значительный спрос. Компания Intel стала наращивать функциональность, разрядность и повышать частоту микропроцессора. В 1978 году был представлен 16-битный процессор Intel 8086, положивший начало архитектуре x86, или Intel x86. Популярность микроархитектуры x86 была столь велика, что аналогичные процессоры стали выпускать другие производители.

В 1985 году компания Intel выпустила первый 32-битный процессор Intel 386 (рис. 2). Таким образом сформировалось понятие архитектуры Intel Architecture 32-bit (IA-32), она же Intel x86, или просто x86 [6].

В дальнейшем она стала 64-битной и получила название x86-64, или AMD64, так как впервые 64-битное расширение архитектуры x86 представила компания AMD. Нужно отметить, что Intel x86 не следует путать с Intel Architecture 64-bit (IA-64), которая является принципиально другой архитектурой VLIW, о чём будет сказано позже.

Формально все процессоры x86 являются процессорами CISC-архитектуры.

Итак, x86 – это типичный представитель CISC-архитектуры. Таким образом, в современной интерпретации, говоря CISC, подразумеваем x86, и наоборот.

Для архитектуры CISC характерно:

1) малое количество регистров общего назначения;
2) большое количество различных машинных команд, каждая из которых выполняется за несколько тактов процессора;
3) различные форматы команд с разной длиной;
4) преобладание двухадресной системы команд;
5) развитой механизм адресации операндов.

Основными плюсами данной архитектуры можно считать простоту и эффективность программирования (несколько команд могут быть заменены одной более сложной), а также большое историческое наследие в виде множества написанных для неё программ.

Данная архитектура на текущий момент является основной для настольных и серверных систем.

Производительность

Разработчики вычислительных систем постоянно стремятся к повышению их производительности, определяющим показателем которой является количество инструкций, выполненных за единицу времени.

Общую формулу производительности можно представить в виде:

$$P = \frac{N}{t}, \quad (1)$$

где P – производительность, N – количество инструкций, t – время выполнения.

Добавив количество тактов, необходимых для выполнения инструкций (n), формулу (1) можно представить следующим образом:

$$P = \frac{N}{n} \times \frac{n}{t}. \quad (2)$$

Первая часть произведения – это количество инструкций, выполняемых за один такт, а вторая – количество тактов процессора за единицу времени, то есть тактовая частота процессора. Таким образом, как следует из выражения (2), для увеличения производительности процессора нужно либо поднимать тактовую частоту, либо увеличивать число инструкций, выполняемых за один такт.

Самый простой способ увеличения скорости вычислений – повышение тактовой частоты процессора. Однако на этом пути существуют некоторые технологические ограничения, не позволяющие постоянно наращивать частоту. Поэтому большинство проектировщиков для повышения производительности при данной тактовой частоте процессора применяют параллелизм – исполнение двух и более инструкций одновременно.

Параллелизм может быть на уровне команд и уровне процессоров. В случае параллелизма на уровне команд происходит запуск большего количества команд в секунду. В случае параллелизма на уровне процессоров над одним заданием работают одновременно несколько процессоров. Каждый подход имеет свои преимущества, и в современных системах используются оба подхода.

Конвейер

Первоначально с целью повышения быстродействия при той же тактовой частоте в центральный процессор была введена конвейерная архитектура (Pipelining).

Обычно для выполнения каждой команды требуется осуществить некоторое количество однотипных операций, таких как выборка команды, дешифрация команды, выборка операнда, выполнение команды и запись результата.

Выполнение каждой из этих операций сопоставляют с одной ступенью конвейера (рис. 3). На рисунке видно, как работает конвейер во времени. В момент времени 5 происходит выполнение уже пяти команд одновременно. Если принять время цикла равным 2 нс, тогда для выполнения инструкции (про-

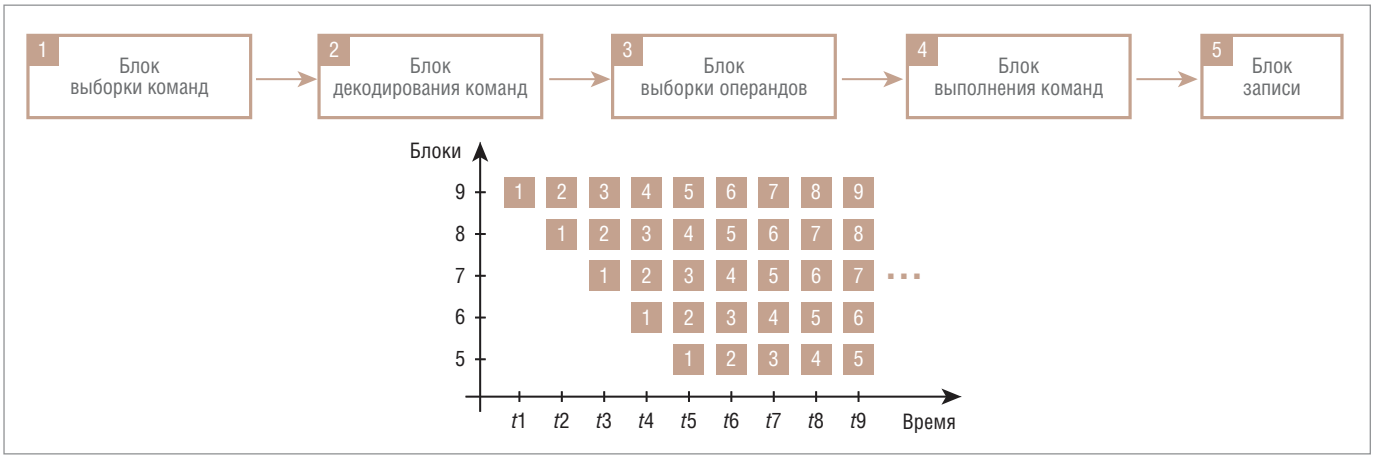

Рис. 3. Пятиступенчатый конвейер и его состояние во времени

хождения через конвейер) потребуется 10 нс. В результате конвейерной обработки, когда каждый такт конвейер выдаёт результат, время выполнения инструкции будет 2 нс.

Архитектура RISC

Повышение производительности CISC-микропроцессоров из-за особенностей архитектуры приводило к росту количества транзисторов, в результате чего кристаллы становились всё более сложными и дорогостоящими в производстве. Вопросы закрывались конструктивно-технологическими решениями, но в конечном итоге по экономическим соображениям уже не давали адекватного роста производительности.

CISC-архитектура в первозданном своём виде достигла потолка производительности. Применение конвейера для повышения производительности требовало использования простых и быстрых команд. Необходимость дальнейшего роста производительности привела к использованию архитектуры RISC (Reduced Instruction Set Computer), что означает «компьютер с сокращённым набором команд» (табл. 1).

Архитектура была разработана в рамках проекта Berkeley RISC. В 1980 году группа разработчиков приступила к созданию процессора, не ориентированного на интерпретацию, в котором инструкции должны выполняться процессорным ядром без использования микрокода. Исследования работы процессора Motorola 68000 показали, что программы попросту не использовали подавляющее большинство инструкций, заложенных в процессор. Работал принцип 80/20, то есть большее время выполнения типовых программ (80–90%) приходится на относительную малую часть команд процессора (10–20%). Планировалось создать процессор, который бы содержал лишь самые необходимые инструкции. При этом не только уменьшилось общее количество процессорных инструкций, принципиальное отличие заключается в том, что любая инструкция платформы RISC является простой и выполняется за один такт (по крайней мере, должна выполняться), тогда как на выполнение CISC-инструкции могло уходить несколько десятков тактов. При этом длина команды является фиксированной.

Основные особенности RISC-процессоров [7]:

1. Сокращённый набор команд.
   Первый «настоящий» RISC-процессор имел всего 31 команду. В дальнейшем их количество постепенно росло и достигло 100–200 инструкций в зависимости от реализации процессора. Но это всё равно в несколько раз, а то и на порядок меньше инструкций CISC-процессоров.
2. Большинство команд выполняется за один такт.
   Все команды выполняются непосредственно аппаратным обеспечением, то есть напрямую без интерпретации микрокомандами. Устранение уровня интерпретации повышает скорость выполнения команд. В компьютерах типа CISC более сложные команды разбиваются на несколько шагов, которые потом выполняются как последовательность микрокоманд.
3. Большое количество регистров общего назначения.
   Доступ к памяти происходит относительно медленно. Если слово было загружено из памяти, оно может храниться в регистрах до тех пор, пока не потребуется. Возвращение слова из регистра в память весьма нежелательно, и лучший способ избежать лишних изменений – это наличие достаточного количества регистров.
4. Наличие жёстких многоступенчатых конвейеров.
   Компьютер должен запускать как можно большее количество команд в секунду. Важным фактором повышения производительности является параллелизм, поскольку запустить на выполнение большое количество команд за короткий промежуток времени можно только в случае, если есть возможность одновременного выполнения нескольких команд. Параллелизм на уровне команд (одновременный запуск) обеспечивается многоступенчатыми конвейерами.
5. Все команды имеют простой формат, и используются немногие способы адресации.
   Команды легко декодируются, и к памяти обращаются только команды загрузки и сохранения.
6. Наличие вместительной раздельной кэш-памяти.
   Это необходимо для уменьшения обращений к памяти и тем самым обеспечения необходимого быстродействия для заполнения регистров и конвейеров.
7. Использование оптимизирующих компиляторов, которые анализируют исходный код и частично меняют порядок следования команд.

Упрощение набора команд призвано сократить конвейер, что позволяет избежать задержек на операциях условных и безусловных переходов. Однородный набор регистров упрощает работу компилятора при оптимизации исполняемого программного кода. Кроме того, RISC-процессоры отличаются меньшим энергопотреблением и тепловыделением.

Уже первые микропроцессоры RISC значительно опережали процессоры CISC по производительности. Учитывая это, можно было предположить, что они должны были занять доминирующее положение на рынке. Но этого не произошло, по крайней мере, по двум причинам. Во-первых, компьютеры RISC несовместимы с архитектурой Intel x86, а многие компании уже вложили значительные средства в программное обеспечение для продукции Intel. И во-вторых, компания Intel сумела воплотить те же идеи в своей архитектуре. Здесь следует отметить, что упомянутый ранее процессор Intel 386 был последним процессором Intel с так называемой классической CISC-архитектурой. Столкнувшись с ограничениями по повышению производительности, компания Intel в процессоре следующего поколения Intel 486 применила RISC-ядро и добавила другие элементы RISC-архитектуры, такие как кэш-память и конвейеры. Теперь процессорное ядро стало выполнять самые простые (и обычно самые распространённые) команды за один цикл, а по обычной технологии CISC интерпретируются более сложные команды. В результате обычные команды выполняются быстро, а более сложные и редкие – медленно. Хотя при таком смешанном подходе производительность ниже, чем в архитектуре RISC, новая архитектура CISC имеет ряд преимуществ, поскольку, с одной стороны, появилась возможность повышения производительности, а с другой, можно использовать старое программное обеспечение без изменений.

Эта гибридная структура привела современные процессоры x86 к тому, что бóльшую часть площади кристалла занимают элементы, предназначенные для переделки CISC-инструкций в RISC-инструкции, разрешения конфликтов, прогнозирова-

Таблица 1

Сравнение архитектур CISC и RISC

	CISC	RISC
Набор команд	Расширенный	Сокращённый
Время выполнения команд	Различное	Команда за такт
Программный код	Короткий	Длинный
Использование конвейера	Плохое	Отличное
Реализация процессора	Очень сложная	Простая
Модели процессоров	IBM 360/370, DEC PDP/VAX, Intel, AMD, Motorola	SPARC, PowerPC, ARM, MIPS, МЦСТ R

ния переходов, исправления последствий неправильных прогнозов и для решения других подобных задач. Для реальной вычислительной работы остаётся только небольшое количество элементов.

Реализацией RISC-архитектуры являются процессоры ARM, MIPS, PowerPC, SPARC и R1000 – российский процессор с 64-битной архитектурой SPARC v.9 производства АО «МЦСТ».

Суперскалярность

Вычислительные операции могут быть скалярными и векторными.

При выполнении инструкции скалярным процессором обрабатывается один или два операнда (скаляра). В векторных операциях в качестве операндов выступают упорядоченные массивы данных – векторы.

В векторном процессоре, в отличие от обычного, который также может выполнять однотипные операции с множеством данных (инструкция SIMD – Single Instruction Multiple Date), все операции выполняются в одном блоке суммирования, который имеет конвейерную структуру. Примером являются процессоры компании Cray Research, начиная с модели Cray 1, выпущенной в 1974 году.

Скалярный процессор выполняет операции последовательно одна за другой. Последним скалярным процессором компании Intel была модель 80486 – 32-битный микропроцессор x86 четвёртого поколения, выпущенный в 1989 году. Он имел пятиступенчатый конвейер [8]. Следующий процессор Intel Pentium имел уже два таких конвейера.

Схема с двумя конвейерами представлена на рис. 4. После выборки из памяти двух инструкций каждая помещается в один из конвейеров. Чтобы выполняться параллельно, инструкции не должны конфликтовать из-за ресурсов и ни одна из них не должна зависеть от результата выполнения другой. Как и в случае с одним конвейером, либо компилятор должен гарантировать отсутствие конфликтов при выполнении инструкций, либо это выявляется и устраняется дополнительным оборудованием непосредственно в ходе выполнения инструкции.

Специальные компиляторы для процессора Intel Pentium объединяли совместимые команды в пары и генерировали программы, которые могли выполняться быстрее, чем в предыдущих версиях процессоров. В некоторых случаях, например при операциях с целыми числами, при той же тактовой частоте программы на процессоре Intel Pentium исполнялись вдвое быстрее, чем на процессоре Intel 486. Выигрыш достигался благодаря второму конвейеру.

Дальнейшее увеличение конвейеров было весьма сложно и громоздко в технической реализации, поэтому был использован другой подход – один конвейер с большим количеством функциональных блоков – суперскалярная архитектура.

Суперскалярный процессор реализует параллелизм на уровне инструкций, это обеспечивается за счёт включения в конвейер нескольких функциональных узлов. В 1987 году для обозначения данного подхода был введён термин «суперскалярная архитектура». На рис. 5 показан конвейер с пятью функциональными блоками.

VLIW-процессоры

Архитектура VLIW (Very Long Instruction Word – сверхдлинное командное слово) основывается на явно выраженном параллелизме вычислений, заложенном в систему команд процессора, – EPIC (Explicitly Parallel Instruction Computing – вычисления с явным параллелизмом команд).

Основной принцип организации этой архитектуры сводится к тому, чтобы перенести нагрузку с периода исполнения в период компиляции. Суперскалярный процессор в ходе исполнения переупорядочивает команды, подменяет регистры, распределяет функциональные блоки и выполняет множество других функций, что ведёт к максимальной загрузке аппаратных ресурсов. В архитектуре VLIW эти задачи заранее решает компилятор, который располагает полной и достоверной информацией о регистрах процессора и генерирует оптимальный код, в котором нет конфликтов между регистрами. Кроме того, компилятор следит за загрузкой функциональных блоков и не запускает команды, в которых предполагается обращение к занятым функциональным блокам.

Поток команд в суперскалярном процессоре планируется динамически аппаратным планировщиком с ограничениями по времени и ресурсам. В архитектуре VLIW компилятор планирует поток команд статически и в принципе не ограничен временны́ми и аппаратными ресурсами, что позволяет гене-

Рис. 4. Двойной конвейер

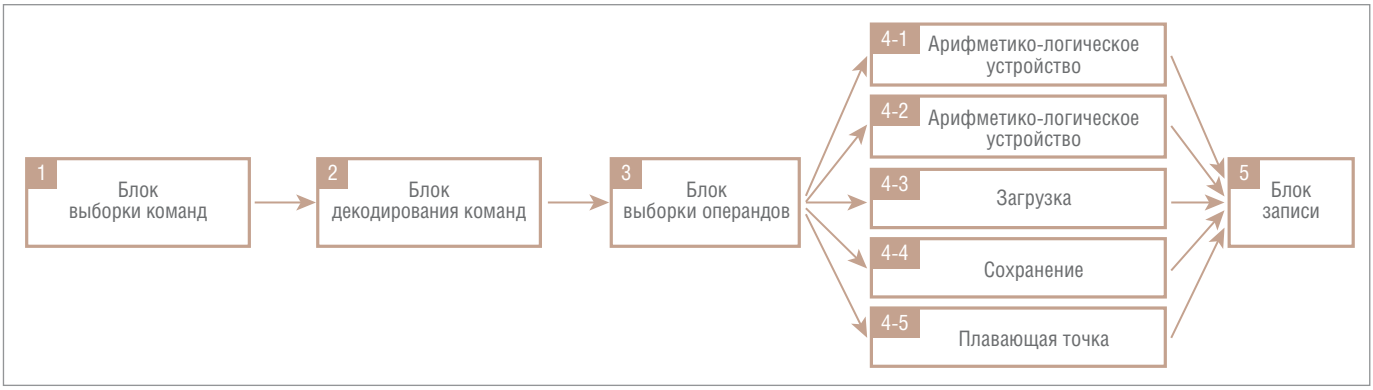

Рис. 5. Суперскалярный процессор с пятью функциональными блоками

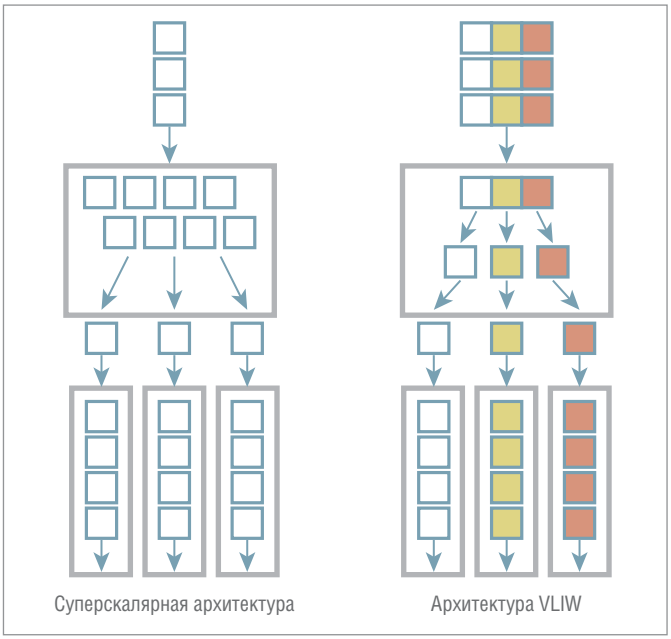

Рис. 6. Сравнение суперскалярной архитектуры и VLIW

рировать оптимальный исполнительный код. В результате на вход VLIW-процессора поступает последовательность больших команд, состоящих из нескольких простых операций, которые могут выполняться параллельно разными функциональными блоками (рис. 6).

В отличие от суперскалярной архитектуры VLIW-процессор имеет простой, а значит, и более быстрый исполнительный конвейер с минимальным количеством ступеней (рис. 7).

Итогом такого подхода, с одной стороны, являются следующие преимущества архитектуры [9].

1. Прежде всего, это более тщательное планирование выполнения программы и оптимизация кода, что даёт лучшее заполнение исполнительных устройств и выполнение большего количества операций за такт.
2. В связи с переносом функциональности на компилятор в процессоре меньше места тратится на функциональные блоки, отвечающие за управление, и больше площади остаётся непосредственно на вычислительные ресурсы, такие как регистры, исполнительные устройства и кэш-память.
3. Это, в свою очередь, приводит к упрощению конструкции процессора и технологического процесса его производства, уменьшению количества транзисторов и понижению тепловыделения.

С другой стороны, необходимо учитывать некоторые особенности.

1. Сложный компилятор, разработка, поддержка и оптимизация которого довольно ресурсоёмки, требует постоянных доработок и оптимизации и целиком лежит в зоне ответственности разработчика процессора.
2. Компилятор должен иметь точную информацию об архитектуре процессора – регистрах и исполнительных устройствах. Таким образом, возникает необходимость компиляции кода для каждого типа процессора при изменении в его архитектуре.
3. Возрастает время компиляции, что, в принципе, не является проблемой при наличии высокопроизводительных рабочих станций у программистов.
4. Становится сложно учесть динамику исполнения программы при наличии условных ветвлений на основе входящих динамических данных.

Рис. 7. Сравнение конвейеров

Несмотря на ограничения, связанные с несовместимостью процессоров VLIW с классической архитектурой, они незаменимы в случае, когда необходимо добиться высочайшей производительности. Примерами VLIW-процессора являются Intel Itanium архитектуры IA-64 (Intel Architecture 64-bit) и «Эльбрус» (архитектура «Эльбрус 2000») производства компании «МЦСТ».

Заключение

Развитие процессорных архитектур характеризуется постоянным стремлением к повышению производительности вычислительных систем. Каждая из указанных архитектур стремится компенсировать недостатки и ограничения других. И в то же время при улучшении одних характеристик могут ухудшиться другие. Поэтому все современные архитектуры находят применение в компьютерных системах, в зависимости от решаемых задач и условий применения. ●

Литература

1. Scale out with Ubuntu Server [Электронный ресурс] // Режим доступа : https://ubuntu.com/server.
2. Qualcomm Collaborates with Microsoft to Support Windows 10 Computing Devices on Next Generation Qualcomm Snapdragon Processors [Электронный ресурс] // Режим доступа : https://www.qualcomm.com/news/releases/2016/12/08/qualcomm-collaborates-microsoft-support-windows-10-computing-devices-next.
3. Windows 10 на архитектуре ARM [Электронный ресурс] // Режим доступа : https://docs.microsoft.com/ru-ru/windows/uwp/porting/apps-on-arm.
4. Операционная система специального назначения Astra Linux® Special Edition [Электронный ресурс] // Режим доступа : https://astralinux.ru/products/astra-linux-special-edition/.
5. The Story of the Intel® 4004. Intel's First Microprocessor. Its invention, introduction, and kasting influence [Электронный ресурс] // Режим доступа : https://www.intel.ru/content/www/ru/ru/history/museum-story-of-intel-4004.html.
6. Intel® 64 and IA-32 Architectures Software Developer's Manual [Электронный ресурс] // Режим доступа : https://software.intel.com/sites/default/files/managed/a4/60/253665-sdm-vol-1.pdf.
7. Sivarama P. Dandamudi. Guide to RISC Processors: For Programmers and Engineers. – USA : Springer Science+Business Media, Inc., 2005.
8. Embedded Intel486™ Processor Family Developer's Manual. – USA : Intel Corporation, 1997.
9. An Introduction To Very-Long Instruction Word (VLIW) Computer Architecture [Электронный ресурс] // Режим доступа : http://twins.ee.nctu.edu.tw/courses/ca_08/literature/11_vliw.pdf.

Авторы – сотрудники фирмы «Адвантикс»
Телефон: (495) 232-1693
E-mail: info@advatix-pc.ru