Лекция17. Классификация архитектур современных микропроцессоров

Одной из важных задач, возникающих при создании средств проектирования микропроцессорных систем, является разработка компиляторов, обеспечивающих эффективный перевод программы с языка высокого уровня на машинно-ориентированный язык, называемый *ассемблером* (assembler). Известно, что существующие компиляторы увеличивают количество команд машинно-ориентированного языка по сравнению с эталонной программой в 1,2 – 2 раза, что приводит к ряду нежелательных эффектов, таких как:

- неэффективное использование команд;
- чрезмерный объем памяти для хранения программы;
- сложность компилятора.

На архитектурном уровне перечисленные проблемы решают посредством создания процессоров, имеющих архитектуры как с полным (Complex Instruction Set Computer — CISC), так и с сокращенным (Reduced Instruction Set Computer — RISC) набором команд.

На практике указанные архитектуры могут дополнять друг друга. Например, современные процессоры общего назначения фирмы Intel имеют все внешние признаки CISC-архитектуры, однако их структура содержит преобразователь CISC-команд в последовательности операций для RISC-ядра, которое непосредственно выполняет все действия над операндами.

CISC-архитектура появилась в начале 1960-х годов как результат усилий разработчиков по созданию универсальной системы команд, обеспечивающей на аппаратном уровне выполнение команд, заданных естественным языком. Это направление было приоритетным по двум основным причинам:

• компиляторы языков высокого уровня только начали появляться, программирование осуществлялось преимущественно с использованием ассемблера;

• многообразие команд и способов адресации позволяло уменьшить объем памяти и количество обращений к ней, что, в свою очередь, давало выигрыш в быстродействии и стоимости.

В целом CISC-архитектура характеризуется большим набором разноформатных команд с использованием многочисленных способов адресации. Основное достоинство такой системы состоит в возможности создавать эффективные алгоритмы для решения широкого круга задач. Поэтому CISC-процессоры называют универсальными, или общего назначения.

Типичным представителем CISC-архитектуры является семейство Pentium фирмы Intel. Первый процессор семейства Pentium имеет архитектуру Intel P5 и появился в 1993 году. Его система команд вместе с расширениями включает около 1000 различных команд, длина командного слова от 1 до 15 байтов, при этом можно использовать более 10 способов адресации.

Одним из существенных недостатков практической реализации архитектуры с полным набором команд является сложная структура устройства управления, реализующего декодирование CISC-команд. Это приводит к увеличению площади и стоимости кристалла, а также снижению производительности. Анализ результатов выполнения различных программ процессором с CISC-архитектурой показал, что в отдельно взятой задаче большинство команд и способов адресации не используются. При этом устройство управления занимает 70 – 80% площади кристалла микропроцессора.

Недостатки CISC-систем способствовали развитию процессоров с сокращенным набором команд (RISC), для которых характерно использование команд фиксированного формата. Современные RISC-процессоры реализуют порядка 100 команд длиной 4 байта, с минимальным числом способов адресации памяти.

Термин «RISC» как название программы исследований в университете Беркли (США) появился в 1980 году, однако первым RISC-компьютером можно назвать суперкомпьютер CDC 6600, который был разработан в 1964 году Сеймуром Креем, реализовывал два способа адресации (регистровый и непосредственный) и мог выполнять 74 операции.

Вычислительная машина CDC 6600 обладала характерной особенностью, которая и сейчас присутствует в RISC-системах и заключается в том, что и операнды, и результаты выполнения любой операции размещаются в регистровой памяти процессора. Данная особенность следует из того, что оператор присвоения является одним из самых распространенных операторов, поэтому операнды целесообразно хранить в непосредственной близости к операционному устройству в памяти, обладающей максимальным быстродействием, т. е. в РОН. При этом возникает задача эффективного распределения операндов по регистрам.

Указанная задача в RISC-процессорах может решаться как программно, так и аппаратно. Программный подход основан на использовании компилятора, который обеспечивает загрузку регистров теми переменными, которые в течение определенного периода времени будут использоваться наиболее часто. Аппаратный подход заключается в наращивании количества РОН для того, чтобы поместить в них наибольшее количество операндов, а также в использовании конвейера. RISC-процессоры имеют увеличенный объем регистровой памяти — от 32 до нескольких сотен РОН.

Важной особенностью RISC-процессоров является использование принципа «load-store». Он обеспечивает сокращение количества обращений к памяти за счет эффективной загрузки операндов и последующей записи результатов выполнения операций в РОН. Данный принцип позволяет выполнять большинство инструкций за один машинный цикл, поскольку не требуется обращения к памяти, работающей на меньшей частоте, чем РОН процессора.

Перечисленные особенности RISC-архитектуры позволили упростить структуру процессоров, что привело к уменьшению площади и стоимости кристалла. С применением конвейерного принципа появилась возможность выполнять несколько команд одновременно. Однако различие между машинно-ориентированным и языком высокого уровня стало еще больше, чем у CISC-процессоров. Как следствие, усложнился компилятор, возникли конфликты, связанные с конвейером команд.

В целом RISC-процессоры эффективны в тех областях применения, в которых можно использовать структурные способы уменьшения времени доступа к оперативной памяти. Однако если программа генерирует произвольные последовательности адресов обращения

к памяти и каждая единица данных применяется только для выполнения одной команды, то производительность процессора определяется временем обращения к основной памяти. В этом случае использование сокращенного набора команд даже уменьшит производительность МП, поскольку требует пересылки «память—РОН» вместо пересылки «память—память».

Дальнейшее повышение производительности процессоров RISC-архитектуры основано:

- на увеличении тактовой частоты и совершенствовании конвейерной обработки данных;
- уменьшении времени доступа к памяти;
- параллельной обработке данных за счет использования ресурсов кристалла.

Современные микропроцессоры могут содержать десять и более операционных устройств, каждое из которых представляет собой конвейер. Увеличение тактовой частоты возможно в результате совершенствования конвейерной обработки и технологии производства. Совершенствование конвейерной обработки заключается в увеличении количества ступеней и эффективном разделении поступающей инструкции на микрокоманды. Это позволяет сократить время выполнения каждой ступени конвейера.

Уменьшение времени доступа к памяти возможно за счет увеличения объема памяти, размещенной на кристалле процессора. С совершенствованием технологии производства объем памяти, размещенный на кристалле, может вырасти, однако память представляет собой ресурс вычислительной системы, который непосредственно не производит вычислений. В связи с этим привлекательным является использование кристалла процессора для построения совокупности функциональных устройств — микропроцессорных ядер, размещенных на кристалле процессора. Основное препятствие на пути повышения производительности в результате увеличения количества функциональных устройств — организация загрузки всех функциональных устройств полезной работой, для этого необходимо обеспечить распараллеливание команд.

Можно выделить два основных подхода к распараллеливанию команд на архитектурном уровне: аппаратный и программный.

Аппаратный подход заключается в том, что параллельная обработка достигается с помощью специальной аппаратуры, которая вхо-

дит в состав микропроцессора и разделяет поступающую команду на микрокоманды для каждого исполнительного устройства. При этом система команд процессора явным образом не ориентирована на параллельную обработку. Такая архитектура называется *суперскалярной*. Достоинства такой системы заключаются в простоте компилятора и возможности миграции программ, написанных под другие архитектуры. Однако при этом не всегда достигается оптимальная загрузка исполнительных устройств. В качестве примера можно привести МП общего назначения с архитектурой Intel P6, ядро которого является суперскалярным: входящая в состав МП аппаратура выполняет трансляцию поступающих сложных CISC-команд на микрокоманды для RISC-исполнительных устройств. Такая архитектура позволяет обеспечить миграцию программ, написанных под CISC-архитектуру, но приводит к дополнительным аппаратным затратам.

Программный подход состоит в использовании специализированного компилятора, разделяющего микрокоманды для каждого исполнительного устройства уже в программе. Этот подход предоставляет программисту все возможности параллельной обработки. В специально отведенных полях макрокоманды каждому из исполнительных устройств микропроцессора указывается своя микрокоманда. Такая архитектура носит название *архитектуры с длинным* (*очень*) *командным словом* (Very Long Instruction Word — VLIW). Длина командного слова составляет 128 и более бит.

Отметим преимущества этой архитектуры перед суперскалярной:

- открытость архитектуры для программиста;
- простота аппаратной части и, как следствие, уменьшение площади кристалла и энергопотребления.

В то же время архитектура с длинным командным словом имеет ряд недостатков:

- большой объем программного кода;
- невозможность миграции программ, написанных для других архитектур;
- сложность отладки;
- сложность компилятора.

Из-за указанных недостатков архитектура VLIW не получила широкого распространения среди процессоров общего назначения. Тем не менее, она активно применяется в процессорах цифровой обработки сигналов и в шейдерных ядрах видеопроцессоров AMD.

В настоящее время применяются комбинированные подходы с использованием векторных сопроцессоров, размещенных на одном кристалле с МП, например, технологии MMX и SSE в процессорах Intel.

В связи с возрастающим объемом информации, обрабатываемой современными микропроцессорами, отдельно стоит задача организации памяти в вычислительных системах.

В 1946 году группа ученых под руководством фон Неймана опубликовала статью, в которой рассматривались принципы, по которым должны строиться компьютеры, а именно:

- двоичное кодирование;
- последовательное программное управление;
- однородность памяти;
- адресуемость памяти.

Принцип двоичного кодирования заключается в использовании двоичной системы счисления для построения цифровых вычислительных машин.

Принцип последовательного программного управления состоит в том, что арифметические устройства выполняют последовательно поступающие из памяти команды.

Принцип однородности памяти заключается в том, что программы и данные хранятся в одной и той же памяти, и неважно, что хранится в данной ячейке памяти – число или команда.

Принцип адресуемости памяти заключается в том, что память должна состоять из пронумерованных ячеек, причем к каждой из них должен быть обеспечен мгновенный доступ.

Архитектуру компьютеров, отвечающих этим принципам, называют *принстонской*, или *фон Неймана*. Однако обычно, когда говорят о принстонской архитектуре, речь идет о способе организации памяти, потому что большинство современных компьютеров отвечают принципам фон Неймана.

Согласно принципу адресуемости, память микросистемы представляет собой упорядоченный набор K-разрядных ячеек с произвольным доступом. Такая память называется *линейной*. Совокупность адресов памяти от 0 до $2^n - 1$, где n — количество двоичных разрядов адреса, называется *адресным пространством*. Обычно адресное пространство разделяется на два подмножества: пространство ввода/вывода и адресное пространство памяти. В этом случае ввод/вывод называют *изолированным*, в противном случае — *совместным*.

Как правило, система ввода/вывода представляет собой набор адресуемых буферных схем или регистров (портов), через которые осуществляется связь с внешними и внутренними аппаратными средствами микросистемы. Система ввода/вывода использует механизм распределения портов, размещая их в специальном адресном пространстве ввода/вывода (Input/Output Segment — IOSEG).

Область РОН (Register Segment — RSEG) может быть полностью изолирована от пространства данных или частично пересекаться с ним. Адресные пространства команд (Code Segment — CSEG) и данных (Data Segment — DSEG) могут быть как отдельными, так и совместными.

Характерной особенностью *принстонской архитектуры* является то, что в ее состав входит отделенная от исполнительных устройств общая память команд и данных. В системе с такой архитектурой исполнительные устройства и основная память соединены одним коммутационным трактом, передачи команд и данных разделены во времени. Достоинствами принстонской архитектуры стали гибкость вычислительной системы, простота реализации и отладки.

Гибкость заключается в том, что неважно, к какому типу относится содержимое ячейки: к командам или к данным.

Простота реализации обусловлена структурной простотой: исполнительные устройства, устройство управления исполнительными устройствами, память — структура, ставшая классической (рис. 4.1.1).

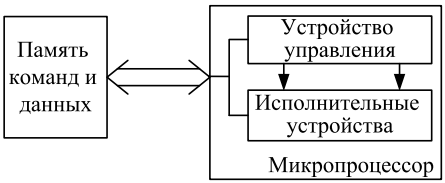

Рис. 4.1.1. Структура вычислительной системы с принстонской архитектурой

Простота отладки определяется наличием только одной системной шины. Однако в связи с увеличением производительности исполнительных устройств и ростом объема передаваемых данных единая системная шина стала ограничивать производительность таких систем.

В *гарвардской архитектуре* память команд и память данных физически разделены. Шины, соединяющие их с процессором, могут иметь как разную разрядность, так и разный объем памяти под команды и данные. Такое архитектурное решение (рис. 4.1.2) дает выигрыш в быстродействии по сравнению с принстонской архитектурой, поскольку за один машинный цикл процессор может получить команды и данные.

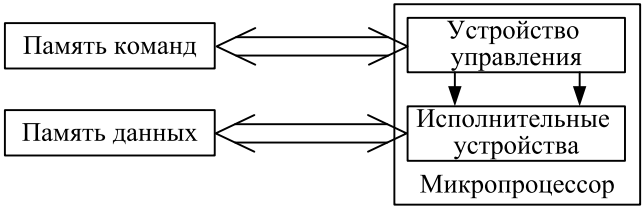

Рис. 4.1.2. Структура вычислительной системы с гарвардской архитектурой

Основным недостатком гарвардской архитектуры является сложность технической реализации двух независимых шин, требующая дополнительных аппаратных затрат. Другим недостатком является то, что объемы памяти команд и памяти данных фиксированы, нет возможности динамически перераспределять информацию между ними.

В *модифицированной гарвардской архитектуре* эти недостатки устранены за счет использования общей памяти и общих шин данных и адреса для всей внешней информации. Внутри процессора применяются два модуля памяти с независимыми шинами адреса, данных и команд. Таким образом, сокращается количество физических линий, а также используется внешняя память для хранения как данных, так и команд (рис. 4.1.3).

Дальнейшим развитием гарвардской архитектуры является применение раздельной внутрикристальной кэш-памяти для команд и данных. Такая архитектура получила название *расширенной гарвардской архитектуры* (Super HArvard ARchitecture Computer — SHARC).

Сегментацию адресного пространства памяти в принстонской и гарвардской архитектурах поясним с помощью рис. 4.1.4. В принстонской архитектуре сегменты команд и данных (CSEG и DSEG) нахо-

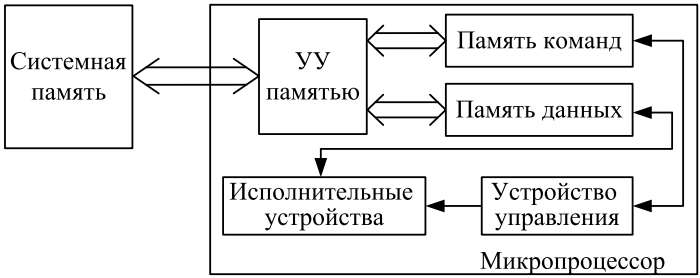

Рис. 4.1.3. Структура вычислительной системы с модифицированной гарвардской архитектурой

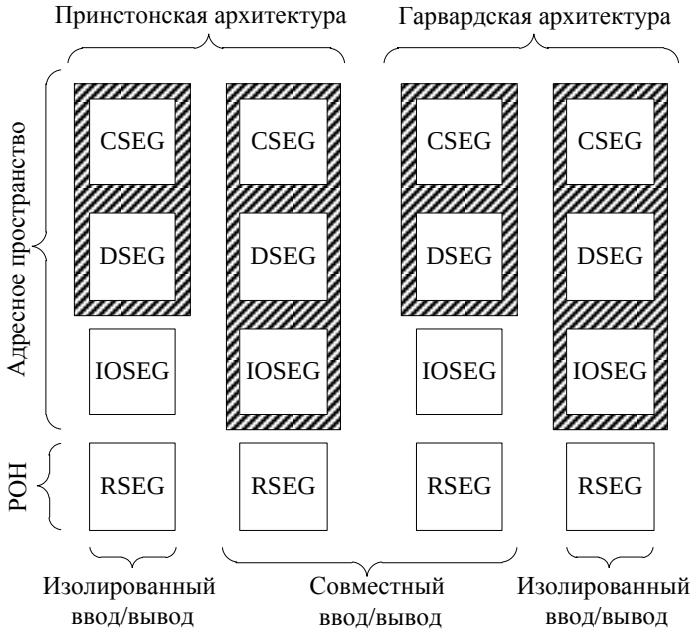

Рис. 4.1.4. Распределение адресного пространства в системах с принстонской и гарвардской архитектурами памяти

дятся в одном адресном пространстве и к ним добавляется сегмент ввода/вывода IOSEG, если ввод/вывод совместный. В гарвардской архитектуре CSEG и DSEG находятся в разных адресных пространствах. Если ввод/вывод изолированный, то DSEG и IOSEG также находятся в разных адресных пространствах.

Контрольные вопросы

1. По каким признакам принято классифицировать архитектуры вычислительных систем?
2. Что означает термин «архитектура с сокращенным набором команд»?
3. В чем заключаются отличия между суперскалярной и VLIW-архитектурами?
4. Сформулируйте принципы фон Неймана.

Литература

1. Микропроцессорные системы: учеб. пособие для вузов / ***Е.К. Александров, Р.И. Грушвицкий, М.С. Куприянов и др.***; ***под ред. Д.В. Пузанкова***. — СПб.: Политехника, 2002. — 935 с.

2. ***Антошина И.В., Котов Ю.Т.*** Микропроцессоры и микропроцессорные системы (аналитический обзор): учеб. пособие. — М.: МГУЛ, 2005. — 432 с.

3. ***Burks A.W., Goldstine H.H., Neumann J.*** Prelimininary discussion of the logical design of an electronic computing instrument. — Institute for Advanced Study, Princeton, N.J., 1946.