БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ГУМАНИТАРНЫЙ ФАКУЛЬТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ
Кафедра технологии программирования

М. А. Акинфина, С. П. Бондаренко

АРХИТЕКТУРНЫЕ РЕШЕНИЯ СОВРЕМЕННЫХ КОМПЬЮТЕРОВ

Учебно-методическое пособие
для студентов гуманитарного факультета

МИНСК
2010

УДК 004.2(075.8)
ББК 32.973-02я73
А 39

Утверждено на заседании
кафедры технологии программирования
12 мая 2010 г., протокол № 12

Акинфина, М. А.

А 39 Архитектурные решения современных компьютеров : учеб.-метод. пособие для студентов гуманит. фак. / М. А. Акинфина, С. П. Бондаренко. – Минск : БГУ, 2010. – 52 с.

В пособии рассматриваются история, основные этапы развития вычислительной техники, классификация имеющихся архитектурных решений современных компьютеров. Подробно излагаются такие вопросы как структура персонального компьютера, многоуровневая организация памяти, основной функционал подсистемы ввода-вывода информации, представление информации в памяти ЭВМ.

Пособие предназначено для студентов гуманитарного факультета БГУ, но может быть использовано и студентами других факультетов: экономического, филологического и международных отношений.

УДК 004.2(075.8)
ББК 32.973-02я73

ВВЕДЕНИЕ

В настоящее время в мире произведены, работают и продолжают выпускаться миллионы вычислительных машин, относящихся к различным поколениям, типам, отличающихся своими областями применения, техническими характеристиками и вычислительными возможностями.

Рынок современных компьютеров отличается разнообразием и динамизмом, каких еще не знала ни одна область человеческой деятельности. Практически каждое десятилетие меняется поколение машин, каждые два года – основные типы микропроцессоров, определяющих характеристики новых ЭВМ. То, что 10-15 лет назад считалось современной ЭВМ, в настоящее время является устаревшей техникой с очень скромными возможностями.

Научно-техническая революция во всех областях науки и техники постоянно выдвигает новые научные, инженерные, экономические задачи, которые требуют проведения крупномасштабных вычислений (задачи проектирования новых образцов техники, моделирования сложных процессов, атомная и космическая техника и др.).

В силу вышесказанного необходимо постоянно анализировать традиционные и новые области применения ЭВМ, классы и типы используемых вычислительных средств, сложившуюся конъюнктуру рынка информационных технологий и его динамику, количество и качество вычислительной техники, выпускаемой производителями средств ЭВМ.

Цель данного пособия – рассмотреть существующие в настоящее время архитектурные решения компьютеров, обеспечивающие их высокую производительность, современные подходы к организации памяти компьютера и ввода-вывода информации, возможности организации суперкомпьютерных систем, используемых в самых различных областях экономики, науки и техники.

1. АРХИТЕКТУРНЫЕ РЕШЕНИЯ КОМПЬЮТЕРА

Сегодня прогресс в любой стране невозможен без компьютеризации всех сфер деятельности. В настоящее время сфера применения вычислительных систем (ВС) непрерывно расширяется, охватывая все новые области в различных отраслях науки, бизнеса и производства. Если ранее ВС применялись в основном в научной сфере для решения вычислительных задач, требующих мощных вычислительных ресурсов, то сейчас из-за бурного развития бизнеса резко возросло количество компаний, отводящих использованию компьютерных технологий и электронного документооборота главную роль. В связи с этим непрерывно растет потребность в построении централизованных ВС для критически важных приложений, связанных с управлением базами данных и обслуживанием телекоммуникаций. Можно отметить такие существенные сферы применения ВС, как обработка транзакций в режиме реального времени (OLTP, On-Line Transaction Processing) и создание хранилищ данных (DW, Data Warehousing) для организации систем поддержки принятия решений (DSS, Decision Support System). Используемые для этих задач ВС должны обладать такими характеристиками как повышенная производительность, масштабируемость, минимально допустимое время простоя.

1.1. ТИПЫ КОМПЬЮТЕРОВ

Современный компьютер представляет собой электронное вычислительное устройство, которое принимает дискретную входную информацию, обрабатывает ее в соответствии со списком хранящихся внутри команд и генерирует результирующую выходную информацию. Упомянутый список команд называется *компьютерной программой*, а место, где он хранится, – *памятью* компьютера. Аппаратное обеспечение компьютера состоит из электронных схем, дисплеев, магнитных и оптических устройств для хранения информации, электромеханического оборудования и средств коммуникации.

Существующие в настоящее время типы компьютеров очень многочисленны и разнообразны; они различаются размерами, стоимостью, вычислительной мощью и назначением. Наиболее распространенным типом компьютеров являются *персональные компьютеры (ПК)*, широко используемые как дома, так и в учебных заведениях, офисах всевозможных компаний. *Настольные компьютеры* – наиболее популярная форма ПК. У настольного компьютера имеются устройства для обработки и хранения данных, дисплей, звуковые выходные устройства, клавиатура.

Устройствами для хранения данных являются жесткие и оптические диски, дискеты. *Портативным компьютером (ноутбуком)* называется компактная версия ПК, в которой все компоненты размещаются в одном блоке, имеющем размер небольшого тонкого портфеля. *Рабочие станции* с графическими входными и выходными устройствами, характеризующиеся высокой разрешающей способностью и имеющие размер настольных компьютеров, обладают значительно большей вычислительной мощью, нежели ПК. Они часто используются в инженерных расчетах, например, для решения задач автоматизированного проектирования.

Наряду с рабочими станциями существует еще целый спектр больших и очень мощных компьютерных систем – от *корпоративных серверов* до *суперкомпьютеров*, относящихся к его вершине. *Корпоративные серверы* используются для обработки деловых данных в крупных корпорациях, которым необходимы большая вычислительная мощь и емкость запоминающих устройств, чем могут обеспечить рабочие станции. Серверы содержат устройства для хранения баз данных и могут обрабатывать большое количество запросов. Они широко используются в сфере образования, в бизнесе и различных некоммерческих организациях. Запросы к серверам и их ответы часто транспортируются с помощью *коммуникационных средств* Интернета. *Суперкомпьютеры* предназначены для проведения крупномасштабных числовых вычислений, необходимых таким приложениям, как метеорологические системы или системы для конструирования самолетов и имитационного моделирования.

Существенные успехи в развитии технологий проектирования ВС и программного обеспечения, успешно применяемых в самых различных сферах экономики, науки, образования и искусства, привели к появлению новых архитектурных решений компьютеров. Концепция последовательного исполнения операций дополнилась идеями параллельной и распределенной обработки данных, а на смену однопроцессорным компьютерам пришли многопроцессорные и параллельные архитектуры.

В вычислительной науке используются три термина, связанные с устройством электронно-вычислительной машины: *архитектура* компьютера, *структура* компьютера и *схема* компьютера.

Архитектурой компьютера называется его описание на некотором общем уровне, включающее описание пользовательских возможностей программирования, системы команд, системы адресации, организации памяти и т.д. Архитектура определяет принципы действия, информационные связи и взаимное соединение основных логических узлов компьютера: процессора, оперативного запоминающего устройства, внешних запоминающих устройств и периферийных устройств. Общность архитек-

туры разных компьютеров обеспечивает их совместимость с точки зрения пользователя.

Структура компьютера – это совокупность его функциональных элементов и связей между ними. Элементами могут быть самые различные устройства – от основных логических узлов компьютера до простейших схем. Структура компьютера графически представляется в виде структурных схем, с помощью которых можно дать описание компьютера на любом уровне детализации. Компьютеры одного семейства, как правило, имеют сходную архитектуру. У них одинаковое число внутренних регистров, используемых для временного хранения данных, одинаковый набор машинных команд, одинаковый формат представления данных. Организация же компьютеров разных моделей может существенно различаться: у них может быть разное число процессоров, разный размер оперативной памяти, разное быстродействие и т.д.

Третий термин – это *схема компьютера*, детальное описание его электронных компонент, их соединений, устройств питания, охлаждения и т.д. Программисту довольно часто требуется знание архитектуры компьютера, реже его организации и никогда – схемы компьютера.

Архитектура компьютера – это многоуровневая иерархия аппаратно-программных средств компьютера. Каждый из уровней допускает многовариантное построение и применение. Конкретная реализация уровней определяет особенности структурного построения компьютера. Детализацией архитектурного и структурного построения компьютера занимаются различные категории специалистов вычислительной техники. Инженеры схемотехники проектируют отдельные технические устройства и разрабатывают методы их сопряжения друг с другом. Системные программисты создают программы управления технического средства информационного взаимодействия между уровнями или программой вычислительного процесса. Программисты прикладники разрабатывают пакеты программ более высокого уровня, которые обеспечивают взаимодействия пользователей с компьютером и необходимый сервис при решении ими своих задач.

Рассмотрим наиболее распространенные архитектурные решения.

Классическая архитектура (архитектура фон Неймана). Эта конструкция содержит одно арифметико-логическое устройство (АЛУ), выполняющее команды, через него проходит поток данных, одно устройство управления (УУ), выполняющее функции управления устройствами, через него проходит поток команд – программа, память (запоминающее устройство), состоящую из перенумерованных ячеек, устройство ввода, устройство вывода. Устройство управления и арифметико-логическое

устройство входят в состав процессора. Обычно эти два устройства выделяются чисто условно, конструктивно они не разделены. Все эти устройства соединены каналами связи, по которым передается информация.

Структура однопроцессорного компьютера представлена на рис. 1.1. Жирными стрелками показаны пути и направления движения информации, а простыми стрелками – пути и направления передачи управляющих сигналов.

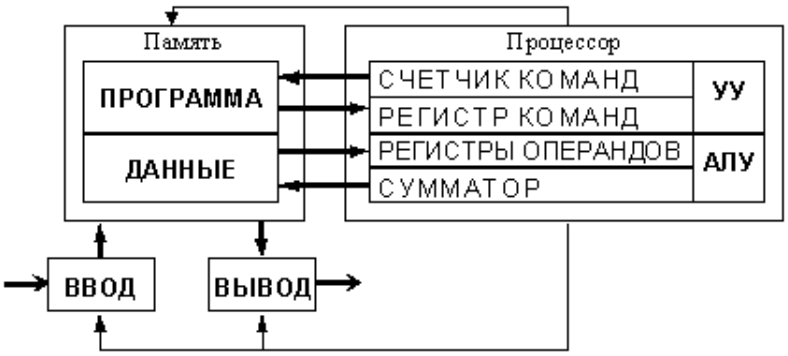

Рис. 1.1. Архитектура однопроцессорного компьютера

К этому типу архитектуры относится и архитектура ПК с общей шиной. Все функциональные блоки: центральный процессор, оперативная память, устройства ввода-вывода, связаны между собой общей шиной, называемой системной магистралью. Физически магистраль представляет собой многопроводную линию с гнездами для подключения электронных схем. Периферийные устройства (принтер и др.) подключаются к магистрали компьютера через специальные контроллеры – устройства управления периферийными устройствами.

До сих пор рассматривались компьютеры с одним процессором. Однако компьютерные системы могут содержать достаточно много процессорных устройств, работающих параллельно и разделяющих общую память системы. Такие системы называются *мультипроцессорными (многопроцессорными) системами с общей памятью*. Они могут параллельно выполнять либо несколько различных прикладных задач, либо несколько подзадач одной крупной задачи. Высокая производительность таких систем достигается за счет их большой сложности и очень высокой стоимости. Их стоимость определяется не только большим количеством процессоров и объемом памяти, но и более сложными схемами их внутренних соединений. Структура такой машины представлена на рис. 1.2.

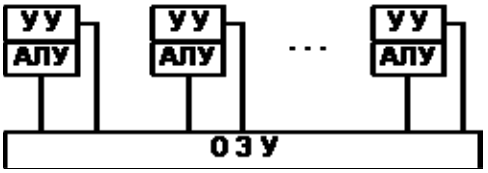

Рис. 1.2. Архитектура многопроцессорного компьютера

Архитектура с параллельными процессорами, является частным случаем мультипроцессорных систем. Здесь несколько АЛУ работают под управлением одного УУ. Это означает, что множество данных может обрабатываться по одной программе – то есть по одному потоку команд.

Высокое быстродействие такой архитектуры можно получить только на задачах, в которых одинаковые вычислительные операции выполняются одновременно на различных однотипных наборах данных. Структура таких компьютеров представлена на рис. 1.3.

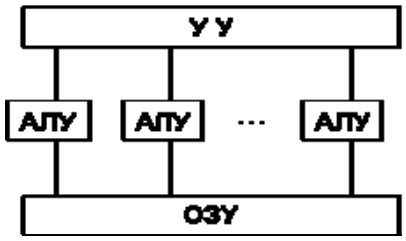

Рис. 1.3. Архитектура с параллельными процессорами

Системы с кластерной архитектурой. Это направление представляет собой дальнейшее развитие многопроцессорной архитектуры. Из нескольких процессоров (традиционных или векторно-конвейерных) и общей для них памяти формируется вычислительный узел. Если полученной вычислительной мощности не достаточно, то несколько узлов объединяются между собой высокоскоростными каналами для обмена данными. Примером такой системы является HP Exemplar, компьютер с кластерной архитектурой от Hewlett-Packard. До 16 процессоров можно объединить в рамках одного узла с общей оперативной памятью до 16 Гб. В свою очередь узлы в рамках вычислительной системы соединяются между собой через высокоскоростные каналы передачи данных.

Многомашинная вычислительная система. Здесь несколько процессоров, входящих в ВС, не имеют общей оперативной памяти. Каждый процессор имеет свою локальную оперативную память, а возможно и дисковую память, и соединяются посредством некоторой коммуникационной среды. Каждый компьютер в многомашинной системе имеет классическую архитектуру. Такие системы применяются достаточно широко.

Однако эффект от применения такой ВС может быть получен только при решении задач, имеющих очень специальную структуру: она должна разбиваться на столько слабо связанных подзадач, сколько компьютеров в системе. Недостатком ВС этого класса является то, что межпроцессорное взаимодействие идет медленнее, чем происходит обработка данных процессорами. Примером является IBM SP2, массово-параллельный компьютер фирмы IBM. В настоящее время строится на основе стандартных микропроцессоров POWERPC 604e или POWER2 SC, соединенных между собой через высокоскоростной коммутатор, причем каждый имеет свою локальную оперативную память и дисковую подсистему.

К этому же классу можно отнести и сети компьютеров. Обладая свойством масштабируемости, они представляют собой дешевую альтернативу дорогим суперкомпьютерам.

Преимущество в быстродействии многопроцессорных и многомашинных вычислительных систем перед однопроцессорными очевидно. В современных машинах часто присутствуют элементы различных типов архитектурных решений. Существуют и такие архитектурные решения, которые радикально отличаются от рассмотренных выше.

1.2. ИСТОРИЯ РАЗВИТИЯ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

Появлению 60 лет назад первых компьютеров предшествовала долгая и медленная эволюция механических вычислительных устройств. В течение последних трех столетий, вплоть до середины двадцатого века, для выполнения базовых операций сложения, вычитания, умножения и деления изобретались все более сложные механизмы, состоящие из колесиков, рычагов и блоков.

Это не было делом усилий одного единственного человека. Если все же назвать кого-то одного, то им может быть Лейбниц, которого во многих отношениях можно считать основателем информатики.

Лейбниц, занимаясь двоичной системой счисления, описал, как выполнять вычисления в двоичной системе. Позже был разработан в общих чертах проект вычислительной машины, работающей в двоичной системе счисления. Наконец, в 1671 – 1674 гг. Лейбницу удается построить машину, выполняющую все четыре арифметических действия.

В XIX веке английским математиком и инженером Чарльзом Бэббиджем был разработан проект вычислительной машины, которая предназначалась для автоматического проведения длинных цепочек вычислений. Конструкция его аналитической машины включала 50 тысяч дета-

лей: зубчатых колес, рычагов и пружин, взаимодействовавших определенным образом. Совершенствуя и уточняя конструкцию машины, Бэббидж первым смог выделить необходимые для ее работы части:

- устройство для хранения чисел, как исходных, так и получающихся в результате вычисления;
- специальный вычислительный блок – процессор;
- устройство для ввода и вывода информации.

В качестве средства хранения информации в аналитической машине использовалась перфокарта – картонная прямоугольная пластина с рядами пробитых в ней дырочек. Каждый ряд состоял из двух частей, разделенных столбцом, содержащим отверстия во всех рядах. Первая часть представляла собой запись числа, вторая – код команды, указывающей, что делать с числом. В созданной Бэббиджем аналитической машине присутствовала хранимая в памяти машины программа ее работы. Меняя программу (перфокарту), можно было изменять порядок вычислений, то есть переходить от одной задачи к другой. Главной особенностью конструкции этой машины является программный принцип работы.

Во время второй мировой войны были сконструированы компьютеры, основанные на электромеханических реле, подобных тем, которые использовались в ранних телефонных коммутаторах. Тогда же в Университете штата Пенсильвания был разработан и первый компьютер, основанный на технологии вакуумных ламп, использовавшихся в то время в радиоприемниках и военных радарах. Вакуумные лампы применялись для выполнения логических операций и хранения данных. Эта технология положила начало эре электронных цифровых компьютеров. Первая выполненная на электронных лампах вычислительная машина ЭНИАК Дж. Эккерта и Дж. Маучли, проектирование которой началось в 1943 г., начала функционировать в 1946 г.

Основополагающая идея вычислительной машины, управляемой размещенной в ее памяти программой, была впервые описана Джоном фон Нейманом и его сотрудниками: Дж. Маучли и Дж. Эккертом 30 июня 1945 г. Она была развита Морисом Уилксом, который в мае 1949 г. сдал в эксплуатацию первую вычислительную машину ЭДСАК.

Все созданные за это время компьютеры в зависимости от технологий, используемых при проектировании и изготовлении процессоров, устройств памяти и систем ввода-вывода, можно отнести к одному из четырех поколений: первое – с 1945 по 1955 год; второе – с 1955 по 1965, третье – с 1965 по 1975, а четвертое с 1975 года по сегодняшний день.

Первое поколение. Столь важная для компьютеров концепция хранимых программ была введена Джоном фон Нейманом. Согласно этой

концепции, программы и их данные, располагались в одной и той же области памяти. Для написания программ использовался язык ассемблера, который затем транслировался в машинный язык.

Для реализации логических функций применялась технология вакуумных ламп, обеспечивавшая выполнение базовых арифметических операций за несколько миллисекунд. По сравнению с механическими и электромеханическими машинами на основе реле скорость вычислений увеличилась в сотни и даже тысячи раз. В компьютерах первого поколения поначалу использовалась память на основе линий задержки, а функции ввода-вывода выполнялись устройствами, похожими на печатные машинки. Затем появилась память на магнитных сердечниках и устройства хранения на магнитных лентах. Широко стала применяться разрядно-параллельная память, разрядно-параллельная арифметика.

Второе поколение. Первые транзисторы были разработаны сотрудниками AT&T Bell Laboratories в начале 1940-х годов. Применение транзисторов, которые очень быстро заменили вакуумные лампы, ознаменовало появление компьютеров второго поколения.

В этих компьютерах уже использовались память на магнитных сердечниках и накопители на магнитных барабанах. Появились языки высокого уровня, и в частности FORTRAN, значительно облегчившие разработку прикладного программного обеспечения. Со временем были изобретены компиляторы для трансляции программ с языков высокого уровня на язык ассемблера, который, в свою очередь, транслировался в машинные коды. В это же время были созданы независимые процессоры ввода-вывода, функционирующие параллельно с выполнявшим программы центральным процессором, за счет чего увеличивалась общая производительность компьютера.

Третье поколение. С появлением технологии объединения множества транзисторов на одном кремниевом чипе, названной технологией интегральных схем, стало возможным создание недорогих, но быстрых процессоров и элементов памяти. Интегральные схемы памяти заменили память на магнитных сердечниках. Начался отсчет эры компьютеров третьего поколения. В этот период было создано множество программных технологий, широко используемых до настоящего времени: микропрограммирование, параллелизм, конвейерная обработка. Программное обеспечение операционных систем позволило совместно использовать ресурсы компьютера несколькими пользовательскими программами. Были разработаны концепции кэширования и виртуальной памяти. Кэш-память представляет основную память для процессора более быстрой, а виртуальная память – намного большей, чем она есть на самом деле. До-

минирующими коммерческими продуктами третьего поколения стали мэйнфреймы System 360 от IBM и линия миникомпьютеров PDP от Digital Equipment Corporation.

Четвертое поколение. В начале 1970-х годов развитие технологии производства интегральных схем достигло того этапа, когда стало возможным интегрировать в одном чипе все компоненты процессора и большие фрагменты основной памяти малых компьютеров. Речь идет о технологии производства чипов, содержащих десятки тысяч транзисторов, которая получила название VLSI (Very Large Scale Integration – очень крупномасштабная интеграция). VLSI позволяет создавать процессоры, состоящие из единственного чипа. Такие процессоры получили название микропроцессоров. Лидерами этой технологии стали компании Intel, National Semiconductor, Motorola.

В производстве компьютерных систем широко стали использоваться такие архитектурные решения, как параллельная и конвейерная обработка, кэширование и виртуальная память. Появляются матричные и векторно-конвейерные процессоры. Введение векторных команд, работающих с целыми массивами независимых данных, позволяло эффективно использовать конвейерные функциональные устройства.

Портативные компьютеры, настольные ПК и рабочие станции, соединенные локальными и глобальными сетями, а также Интернет стали основными средствами решения различных вычислительных задач.

После четвертого поколения. Иногда самые современные, управляемые с помощью приложений компьютеры, называют компьютерами следующего поколения. В последние годы появилась тенденция при именовании каждой новой компьютерной технологии использовать уже не номер поколения, а название, определяющее ее функции. Например: системы с элементами искусственного интеллекта, машины с высокой степенью параллелизма, сильно распределенные системы. Пожалуй, наиболее важной особенностью развития современной компьютерной индустрии является увеличение мощи и доступности настольных компьютеров и широчайшее использование информационных ресурсов Интернета.

1.3. АРХИТЕКТУРНЫЕ ПРИНЦИПЫ ДЖОНА ФОН НЕЙМАНА

В основу построения подавляющего большинства компьютеров положены следующие общие принципы, сформулированные в 1945 г. американским ученым Джоном фон Нейманом.

1. **Принцип программного управления.** Из него следует, что программа состоит из набора команд, которые выполняются процессором автоматически друг за другом в определенной последовательности. Выборка программы из памяти осуществляется с помощью счетчика команд. Этот регистр процессора последовательно увеличивает хранимый в нем адрес очередной команды на длину команды. А так как команды программы расположены в памяти друг за другом, то тем самым организуется выборка цепочки команд из последовательно расположенных ячеек памяти. Если же нужно после выполнения команды перейти не к следующей, а к какой-то другой, используются команды условного или безусловного переходов, которые заносят в счетчик команд номер ячейки памяти, содержащей следующую команду. Выборка команд из памяти прекращается после достижения и выполнения команды "стоп".

2. **Принцип однородности памяти**. Программы и данные хранятся в одной и той же памяти. Поэтому компьютер не различает, что хранится в данной ячейке памяти – число, текст или команда. Над командами можно выполнять такие же действия, как и над данными. Последнее открывает целый ряд возможностей. Например, программа в процессе своего выполнения может подвергаться переработке, что позволяет задавать в самой программе правила получения некоторых ее частей (так в программе организуется выполнение циклов и подпрограмм).

Более того, команды одной программы могут быть получены как результаты исполнения другой программы. На этом принципе основаны методы трансляции – перевода текста программы с языка программирования высокого уровня на язык конкретной машины.

3. **Принцип адресности**. Структурно основная память состоит из перенумерованных ячеек; процессору в произвольный момент времени доступна любая ячейка. Отсюда следует возможность давать имена областям памяти, так, чтобы к запомненным в них значениям можно было впоследствии обращаться или менять их в процессе выполнения программ с использованием присвоенных имен.

Компьютеры, построенные на этих принципах, относятся к типу фон Неймановских. Но существуют компьютеры, принципиально отличающиеся от данного типа. Для них, например, может не выполняться принцип программного управления, т.е. они могут работать без "счетчика команд", указывающего текущую выполняемую команду программы. Для обращения к какой-либо переменной, хранящейся в памяти, этим компьютерам не обязательно давать ей имя.

Архитектура традиционных последовательных компьютеров, основанная на идеях Джона фон Неймана, включает в себя (рис 1.1.) цен-

тральнее УУ, АЛУ, оперативную память – адресное пространство с линейной адресацией, блок управления и устройства ввода-вывода. Последовательность команд применяется к последовательности данных. Быстродействие такого традиционного компьютера определяется быстродействием его центрального процессора и временем доступа к оперативной памяти. Быстродействие центрального процессора может быть увеличено за счет увеличения тактовой частоты, величина которой зависит от плотности элементов в интегральной схеме, способа их "упаковки" и быстродействия микросхем оперативной памяти.

Другие методы повышения быстродействия последовательного компьютера основаны на расширении традиционной неймановской архитектуры. К ним относится применение RISC-процессоров, конвейеров, векторных и суперскалярных процессоров.

2. АРХИТЕКТУРА ПЕРСОНАЛЬНОГО КОМПЬЮТЕРА

Архитектурой ПК называется его логическая организация, структура и ресурсы, которые может использовать программист.

2.1. ФУНКЦИОНАЛЬНАЯ СТРУКТУРА ПК

Как следует из рис. 2.1, ПК состоит из пяти главных, функционально независимых частей: устройство ввода (УВв), устройства памяти (ОЗУ, ВЗУ), арифметико-логическое устройство (АЛУ), устройство вывода (УВыв) и устройство управления (УУ). Устройство ввода принимает через цифровые линии связи закодированную информацию от операторов, электромеханических устройств типа клавиатуры или от других компьютеров сети. Полученная информация либо сохраняется в памяти компьютера для последующего применения, либо немедленно используется арифметическими и логическими схемами для выполнения необходимых операций. Последовательность шагов обработки определяется хранящейся в памяти программой. Полученные результаты отправляются обратно, во внешний мир, посредством устройства вывода. Все эти действия координируются устройством управления. Арифметические и логические схемы в комплексе с главными управляющими схемами называют *процессором*, а все вместе взятое оборудование для ввода и вывода часто называют *устройством ввода-вывода* (input-output unit) .

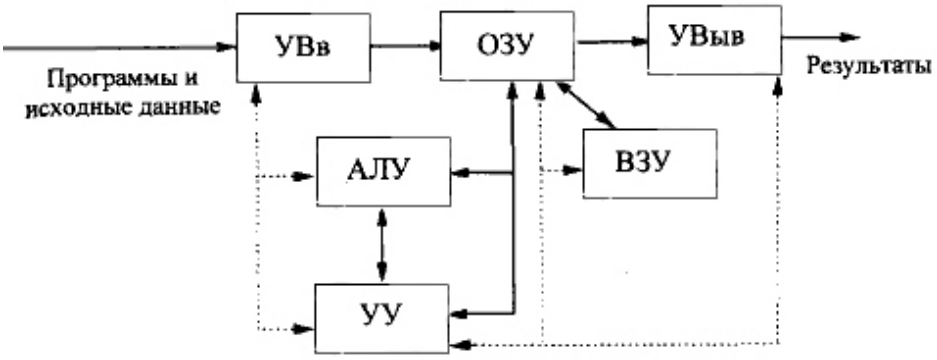

Рис. 2.1. Базовые функциональные устройства ПК

Рассмотрим обрабатываемую компьютером информацию. Ее удобно разделять на две основные категории: команды и данные. *Команды*, или *машинные команды*, – это явно заданные инструкции, которые:

- управляют пересылкой информации внутри компьютера, а также между компьютером и его устройствами ввода-вывода;
- определяют подлежащие выполнению арифметические и логические операции.

Список команд, выполняющих некоторую задачу, называется *программой*. Процессор по очереди извлекает команды программы, хранящейся в памяти, и реализует определяемые ими операции. Компьютер полностью управляется *хранимой программой*, если не считать возможность внешнего вмешательства оператора и подсоединенных к машине устройств ввода-вывода.

Данные – это числа и символы, используемые в качестве операндов команд. Информация, предназначенная для обработки компьютером, должна быть закодирована, чтобы иметь подходящий для компьютера формат. Современное аппаратное обеспечение в большинстве своем основано на цифровых схемах, у которых имеется только два устойчивых состояния, ON (включено) и OFF (выключено). В результате кодирования любое число, символ или команда преобразуется в строку двоичных цифр, называемых *битами*, каждый из которых имеет одно из двух возможных значений: 0 или 1. Для представления чисел обычно используется *позиционная двоичная нотация*. Иногда применяется *двоично-десятичный формат* (Binary-Coded Decimal, BCD), в соответствии с которым каждая десятичная цифра кодируется с помощью четырех бит.

Буквы и цифры также представляются посредством двоичных кодов. Для них разработано несколько разных схем кодирования. Наиболее распространенной считается схема ASCII (American Standard Code for In-

formation Interchange – американский стандартный код для обмена информацией), где каждый символ представлен 7-битовым кодом.

Архитектура современных ПК основана на *магистрально-модульном принципе* (рис. 2.2). Модульный принцип позволяет потребителю подобрать нужную ему конфигурацию компьютера. *Магистраль* или *системная шина* – это набор электронных линий, связывающих воедино программное обеспечение адресации памяти, передачи данных и служебных сигналов, процессор, память и периферийные устройства. Совокупность проводов магистрали разделяется на отдельные группы: шину данных, шину адреса и шину управления.

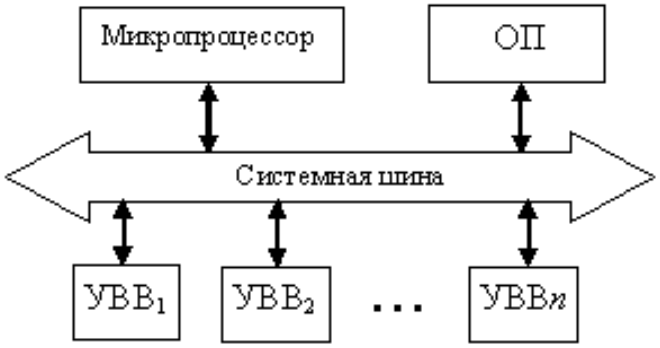

Рис. 2.2. Архитектура системы с общей шиной

Обмен информацией между отдельными устройствами ЭВМ производится по этим трем многоразрядным шинам, соединяющим все модули. Подключение отдельных модулей компьютера к магистрали на физическом уровне осуществляется с помощью *контроллеров*, а на программном обеспечивается *драйверами*. Поэтому внешние устройства ЭВМ заменяемы, и набор таких модулей произволен. Каждое из устройств имеет свой адрес. Когда нужно обратиться к какому-нибудь из них, в магистраль посылается его адрес. Контроллер принимает сигнал от процессора и дешифрует его, чтобы соответствующее устройство смогло принять этот сигнал и отреагировать на него. Приняв сигнал "свободно", процессор посылает ему информацию. За реакцию устройства процессор не отвечает – это функция контроллера.

2.2. АРИФМЕТИКО-ЛОГИЧЕСКОЕ УСТРОЙСТВО

Основой вычислительной машины является процессор. В нем расположены *арифметико-логическое устройство*, *устройство управления* и *регистры* для временного хранения информации. Большинство компьютерных операций выполняется в АЛУ. Рассмотрим типичный при-

мер. Предположим, нам нужно сложить два находящихся в памяти числа. Эти числа пересылаются в процессор, где АЛУ выполняет их сложение. Полученная сумма может быть записана в память или оставлена в процессоре для немедленного использования.

Любые другие арифметические или логические операции, в том числе умножение, деление и сравнение чисел, начинаются также с пересылки этих чисел в процессор, где АЛУ должно выполнить соответствующую операцию. Когда операнды переносятся в процессор, они сохраняются в высокоскоростных элементах памяти, называемых *регистрами*. Каждый регистр может хранить одно слово данных.

Кроме АЛУ, управляющих схем и регистров хранения данных, процессор содержит множество регистров, предназначенных для других целей. В *регистре команды* (Instruction Register, IR) содержится код выполняемой в данный момент команды. Ее результат доступен управляющим схемам, которые генерируют сигналы для управления различными элементами, участвующими в выполнении команды. Еще один специализированный регистр, называемый *счетчиком команд* (Program Counter, PC), служит для контроля за ходом выполнения программы. В нем содержится адрес следующей команды, подлежащей выборке и выполнению. Пока выполняется команда, содержимое регистра PC обновляется – в него записывается адрес следующей команды. Говорят, что регистр PC *указывает* на команду, которая должна быть выбрана из памяти.

Управляющее и арифметико-логическое устройства работают во много раз быстрее, чем все остальные устройства, подключенные к компьютерной системе. Это позволяет одному процессору контролировать множество внешних устройств, таких как клавиатуры, дисплеи, магнитные и оптические диски, сенсоры и механические устройства.

2.3. БЛОК УПРАВЛЕНИЯ

Устройства памяти, арифметики и логики, ввода и вывода хранят и обрабатывают информацию, а также выполняют операции ввода и вывода. Работу таких устройств нужно как-то координировать. Именно этим и занимается *блок управления*. Это, если можно так выразиться, нервный центр компьютера, передающий управляющие сигналы другим устройствам и отслеживающий их состояние.

Управление операциями ввода-вывода осуществляется командами программ, в которых идентифицируются соответствующие устройства ввода-вывода и пересылаемые данные. Однако реальные *синхронизирующие сигналы* (timing signals), управляющие пересылкой, генерируют-

ся *управляющими схемами*. Синхронизирующие сигналы – это сигналы, определяющие, когда должно быть выполнено данное действие. Кроме того, посредством синхронизирующих сигналов, генерируемых блоком управления, осуществляется передача данных между процессором и памятью. Блок управления можно представить себе как отдельное устройство, взаимодействующее с другими частями машины. Но на практике так бывает редко. Большая часть управляющих схем физически распределена по разным местам компьютера. Сигналы, используемые для синхронизации событий и действий всех устройств, передаются по множеству управляющих линий (проводов).

В целом, функционирование компьютера можно описать следующим образом.

- Компьютер с помощью блока ввода принимает информацию в виде программ и данных и записывает ее в память.
- Хранящаяся в памяти информация при необходимости пересылается в арифметико-логическое устройство для дальнейшей обработки.
- Данные, полученные в результате обработки информации, направляются на устройства вывода.
- За все действия, производимые внутри машины, отвечает блок управления.

Важнейшими характеристиками процессора являются:

- разрядность;
- тактовая частота;
- размер адресного пространства.

Разрядность процессора. Обычно команда выполняется не по одному биту, а одновременно группами по 8, 16, 32, 64 бита. Число одновременно обрабатываемых битов и называется *разрядностью процессора*. Чем больше разрядность процессора, тем больше информации он может обработать в единицу времени, тем выше его эффективность.

Тактовая частота процессора характеризует быстродействие компьютера. Режим работы процессора задается микросхемой, которая называется *генератором тактовой частоты*. Это своеобразный метроном внутри компьютера. На выполнение процессором каждой операции отводится определенное количество тактов. Ясно, что если метроном "стучит быстрее", то и процессор работает быстрее.

Каждый процессор может работать не более чем с определенным количеством оперативной памяти. Максимальное количество памяти, которое процессор может обслужить, называется *адресным пространством процессора* и является важной характеристикой компьютера. Объем адресного пространства определяется *разрядностью адресной шины*.

2.4. БЛОК ПАМЯТИ

Задачей блока памяти является хранение программ и данных. Существует два класса запоминающих устройств, а именно первичные и вторичные. *Первичное запоминающее устройство* (primary storage) – это память, быстродействие которой определяется скоростью работы электронных схем. Пока программа выполняется, она должна храниться в первичной памяти. Эта память состоит из большого количества полупроводниковых ячеек, каждая из которых может хранить один бит информации. Ячейки редко считываются по отдельности – обычно они обрабатываются группами фиксированного размера, называемыми *словами*. Количество битов в каждом слове называют *длиной машинного слова*. Обычно слово имеет длину от 16 до 64 бит. Одним из факторов, характеризующих класс компьютера, является *емкость его памяти*. Малые машины обычно могут хранить лишь несколько десятков миллионов слов, тогда как средние и большие машины обычно способны хранить сотни миллионов слов. Типичными единицами измерения количества обрабатываемых машиной данных являются слово, несколько слов или часть слова. Как правило, память организована так, что за время одного обращения к памяти (за одну базовую операцию) считывается или записывается только одно слово. Команды и данные записываются в память и считываются из памяти под управлением процессора.

Для облегчения доступа к словам в памяти с каждым словом связывается отдельный *адрес*. Адреса – это числа, идентифицирующие конкретные местоположения слов в памяти. Для того чтобы прочитать слово из памяти или записать в память, необходимо указать его адрес и задать управляющую команду, которая начнет соответствующую операцию.

Память, к любой точке которой можно получить доступ за короткое и фиксированное время, называется памятью с произвольным доступом (Random-Access Memory, RAM). Время, необходимое для доступа к одному слову, называется *временем доступа к памяти*. Первичная память компьютера обычно представляет собой *иерархическую структуру*, состоящую из трех или четырех уровней полупроводниковых RAM-элементов с различной скоростью и разным размером. Наиболее быстродействующим типом RAM-памяти является *кэш-память* (или просто *кэш*). Она напрямую связана с процессором и часто находится на одном с ним интегрированном чипе. Память большей емкости, но менее быстрая, называется *основной памятью* (main memory). Основная память состоит из *постоянного* (ПЗУ) и *оперативного* (ОЗУ) запоминающих устройств.

Постоянное запоминающее устройство (ROM – Read Only Memory) предназначено для хранения информации, специальным образом "зашиваемой" в устройство при его изготовлении, и может только считываться. В ПЗУ хранятся программы для проверки оборудования компьютера, инициирования загрузки операционной системы и выполнения базовых функций по обслуживанию устройств компьютера, а также настройки конфигурации компьютера. Так как основное содержимое ПЗУ связано с обслуживанием ввода-вывода, часто программы ROM-памяти называют BIOS (Basic Input-Output System).

Оперативное запоминающее устройство предназначено для временного хранения программ (системных и прикладных), исходных данных, промежуточных и окончательных результатов. При выключении компьютера, информация в ОЗУ стирается. Часть ОЗУ, называемая "видеопамять", содержит данные, соответствующие текущему изображению на экране дисплея.

Процесс взаимодействия процессора и памяти сводится в основном к двум операциям: запись информации в память и чтение информации из памяти. При записи процессор по шине адреса передает биты, кодирующие адрес, по шине управления передает управляющий сигнал – "запись" и по шине данных передает записываемую информацию. При чтении по шине адреса передается соответствующий адрес оперативной памяти, по шине управления передается управляющий сигнал – "чтение", и с шины данных считывается требуемая информация.

Первичные запоминающие устройства являются исключительно важными компонентами для компьютера, но они довольно дороги. Поэтому компьютеры оборудуются дополнительными, более дешевыми *вторичными (внешними) запоминающими устройствами*, используемыми для хранения больших объемов данных и большого количества программ. В настоящее время таких устройств имеется достаточно много. Но наиболее широкое распространение получили *магнитные диски* и *оптические диски* (CD-ROM, CD-RW, DVD).

2.5. СТРУКТУРА ШИНЫ

До сих пор речь шла о функциях отдельных частей компьютера. Однако для того чтобы составить действующую систему, эти части должны быть соединены между собой определенным образом. Способов их соединения существует очень много, но мы рассмотрим лишь простейшее и самые распространенное из них.

Для соединения нескольких функциональных устройств компьютера проще всего использовать *общую шину* (single bus) (рис. 2.2). К этой шине подсоединяются все устройства компьютера. Поскольку за один раз по шине может пересылаться только одно слово данных, в каждый конкретный момент шину могут использовать только два устройства. Главным достоинством архитектуры с общей шиной является ее низкая стоимость и гибкость в отношении подключения периферийных устройств. При наличии в системе нескольких шин возможно одновременное выполнение нескольких операций пересылки данных, благодаря чему такая система работает быстрее, но и стоимость ее выше.

Процессор и основная память находятся на большой плате, которая называется материнской. Для подключения к ней периферийных устройств (дисководов, манипуляторов типа мыши, принтеров и т.д.) служат специальные платы – контроллеры. Они вставляются в разъемы (слоты) на материнской плате, а к их концу (порту), выходящему наружу компьютера, подключается дополнительное устройство. Таким образом, периферийные устройства подключаются к системной магистрали не непосредственно, а через специальные устройства – контроллеры.

Подсоединенные к шине устройства могут заметно отличаться друг от друга по скорости функционирования. Некоторые электромеханические устройства, в том числе клавиатура и принтеры, работают относительно медленно. Значительно выше скорость работы, скажем, магнитных и оптических дисков. А память и процессор функционируют со скоростью электронных схем, благодаря чему являются самыми быстрыми частями компьютера. Поскольку все эти три типа устройств могут взаимодействовать между собой через шину, необходим такой механизм пересылки данных, который не ограничивал бы скорость обмена информацией между любыми двумя устройствами скоростью более медленного из них и сглаживал бы разницу в скорости работы процессора, памяти и внешних устройств.

Самый распространенный подход к решению этой задачи основан на использовании *буферных регистров*, которые встраиваются во внешние устройства для хранения получаемой ими информации. Рассмотрим процесс передачи кода символа от процессора принтеру. Процессор пересылает данные по шине в буфер принтера. Поскольку буфер представляет собой электронный регистр, пересылка выполняется очень быстро. Когда буфер будет заполнен, принтер начнет печатать, и вмешательство процессора больше не потребуется. Шина и процессор освобождаются для другой работы, которая может выполняться одновременно с печатью символа, хранящегося в буфере принтера. Таким образом, использование

буферов сглаживает различия в скорости функционирования процессора, памяти и устройств ввода-вывода и предотвращает блокирование высокоскоростного процессора медленными устройствами на все время выполнения операций ввода-вывода. Процессор может быстро переключаться от одного устройства к другому, обслуживая их параллельно.

Как отмечалось выше, основной обязанностью системной шины является передача информации между процессором и остальными компонентами компьютера. По этой шине осуществляется не только передача информации, но и адресация устройств, а также обмен специальными служебными сигналами. Таким образом, упрощенно системную шину можно представить как совокупность сигнальных линий, объединенных по их назначению (данные, адреса, управление).

Компьютер сможет работать с достаточной скоростью лишь при условии, что будет организован таким образом, чтобы полное слово данных обрабатывалось им за указанное время. Когда слово данных пересылается между устройствами, параллельно перемещаются и все его биты. Каждый бит пересылается по своей линии, так что для пересылки слова требуется несколько параллельных линий.

Число одновременно передаваемых по шине адреса и шине данных разрядов (битов) называется *разрядностью* соответствующей шины и является важной характеристикой ПК. Разрядность шины адреса определяет максимальное общее количество доступной памяти (адресное пространство процессора); разрядность шины данных – максимальную порцию информации, которую можно получить из памяти за один раз. Общая схема взаимосвязи этих элементов: процессора, памяти и шины представлена на рис. 2.3.

Шина адреса предназначена для передачи по ней адреса того устройства (или той ячейки памяти), к которому обращается процессор. Адрес на нее выдает всегда только процессор.

По шине данных передается вся информация. При операции записи информацию на нее выставляет процессор, а считывает то устройство (например, память или принтер), адрес которого выставлен на шине адреса. При операции чтения информацию выставляет устройство, адрес которого выставлен на шине адреса, а считывает процессор.

На шине управления устанавливаются управляющие сигналы, такие, например, как сигналы чтения, записи, готовности. Кроме того, каждое внешнее устройство, которому нужно обратиться к процессору, имеет на этой шине собственную линию. Когда периферийное устройство «хочет обратиться» к процессору, оно устанавливает на этой линии специальный сигнал (сигнал прерывания), заметив который, процессор прерывает

выполняемые в этот момент действия и обращается (командой чтения или записи) к устройству.

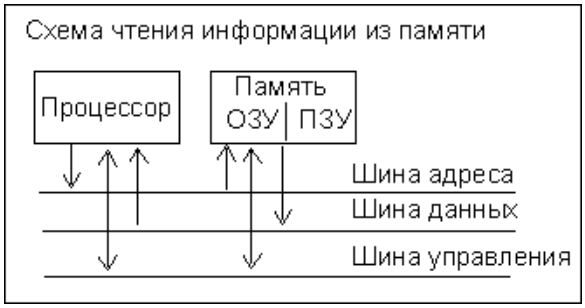

Рис. 2.3. Общая схема взаимосвязи процессора, памяти и шины

Компьютер не только пересылает данные между памятью и процессором, но и принимает их от входных устройств, а также отсылает выходным устройствам. Поэтому среди машинных команд имеются и команды для выполнения операций ввода-вывода.

2.6. УСТРОЙСТВА ВВОДА-ВЫВОДА

Компьютер принимает кодированную информацию через устройство ввода, задачей которого является чтение данных. Наиболее распространенным устройством ввода является клавиатура. Когда пользователь нажимает клавишу, соответствующая буква или цифра автоматически преобразуется в определенный двоичный код и по кабелю пересылается либо в память, либо процессору. Существует и ряд других устройств ввода, среди которых джойстики, трекболы и мыши. Они используются совместно с дисплеем в качестве графических входных устройств. Для ввода звука могут использоваться микрофоны, для ввода видеоизображений видеокамеры. Воспринимаемая ими информация конвертируются в цифровые коды для хранения и обработки.

Функция блока вывода противоположна функции блока ввода: он направляет результаты обработки в так называемый внешний мир. Типичным примером устройства вывода является *принтер*.

Некоторые устройства, и в частности графические дисплеи, выполняют одновременно и функцию вывода, и функцию ввода. Поэтому они называются устройствами ввода-вывода.

В заключении отметим, что быстродействие персонального компьютера определяется быстродействием его центрального процессора и временем доступа к оперативной памяти. Существенное влияние на его

производительность оказывают также технологии изготовления процессора, тактовая частота системной шины, размер кэш-памяти.

3. ВВОД-ВЫВОД ИНФОРМАЦИИ

Операции ввода-вывода являются одной из важнейших составляющих работы компьютера. От того, как они выполнятся, в значительной мере зависит его производительность.

3.1. ОРГАНИЗАЦИЯ ВВОДА-ВЫВОДА

Существует много разнообразных устройств, которые могут взаимодействовать с процессором и памятью: таймер, жесткие диски, клавиатура, дисплеи, мышь, модемы и т. д., вплоть до устройств отображения и ввода информации в авиационно-космических тренажерах. Часть этих устройств может быть встроена внутрь корпуса компьютера, часть – вынесена за его пределы и общаться с компьютером через различные линии связи: кабельные, оптоволоконные, радиорелейные, спутниковые и т. д. Конкретный набор устройств и способы их подключения определяются целями функционирования вычислительной системы.

Простейшая схема подключения всех внешних устройств, дисплея, клавиатуры, внешних запоминающих устройств и других устройств к компьютеру заключается в использовании общей шины (рис. 3.1).

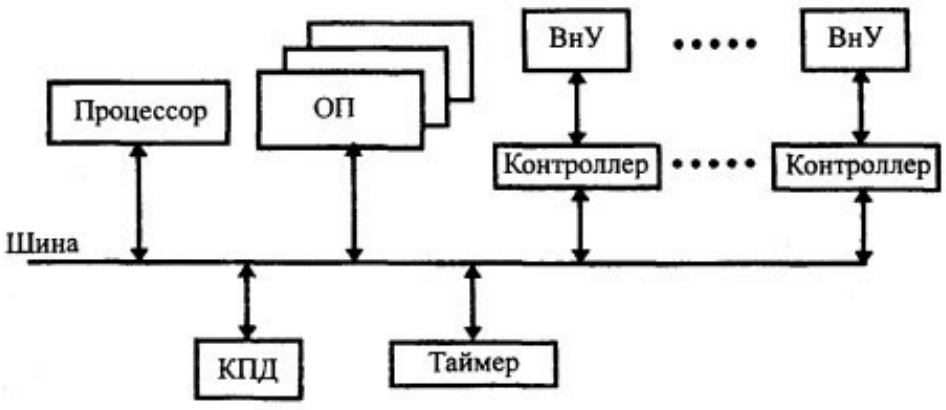

Рис. 3.1. Структурная схема компьютера с общей шиной

Все устройства, подключенные к шине, могут обмениваться между собой информацией. Подключение обеспечивается через соответствующие адаптеры – согласователи скоростей работы сопрягаемых устройств, или контроллеры – специальные устройства управления периферийной

аппаратурой. Контроллеры в ПЭВМ играют роль каналов ввода-вывода. В качестве особых устройств следует выделить таймер – устройство измерения времени и контроллер прямого доступа к памяти – устройство, обеспечивающее доступ к оперативной памяти, минуя процессор.

Контроллер связывает периферийное оборудование или каналы связи с центральным процессором, освобождая процессор от непосредственного управления функционированием данного оборудования. Контроллеры устройств ввода-вывода весьма различны как по своему внутреннему строению, так и по исполнению (от одной микросхемы до специализированной вычислительной системы со своим процессором, памятью и т. д.), поскольку им приходится управлять совершенно разными приборами. Обычно каждый контроллер имеет, по крайней мере четыре внутренних регистра, называемых *регистрами состояния***,** *управления, входных данных* и *выходных данных*. Для доступа к содержимому этих регистров вычислительная система может использовать один или несколько портов. Для простоты изложения будем считать, что каждому регистру соответствует свой порт.

Регистр состояния содержит биты, значение которых определяется состоянием устройства ввода-вывода и которые доступны только для чтения ВС. Эти биты индицируют завершение выполнения текущей команды на устройстве (*бит занятости*), наличие очередного данного в *регистре выходных данных* (*бит готовности данных*), возникновение ошибки при выполнении команды (*бит ошибки*) и т. д.

Регистр управления получает данные, которые записываются вычислительной системой для инициализации устройства ввода-вывода или выполнения очередной команды, а также изменения режима работы устройства. Часть битов в этом регистре может быть отведена под код выполняемой команды, часть битов будет кодировать режим работы устройства, бит *готовности команды* свидетельствует о том, что можно приступить к ее выполнению.

Регистр выходных данных служит для помещения в него данных для чтения вычислительной системой, а *регистр входных данных* предназначен для помещения в него информации, которая должна быть выведена на устройство. Разумеется, набор регистров и составляющих их битов приблизителен, но в том или ином виде он обычно присутствует во всех контроллерах устройств.

Обычно шина состоит из трех наборов линий, предназначенных для передачи адресов, данных и управляющих сигналов. Каждому устройству ввода-вывода присваивается уникальный набор адресов. Когда процессор помещает на адресные линии конкретный адрес, устройство, рас-

познавшее с помощью дешифратора (декодера) этот адрес, отвечает на команду, помещенную на управляющие линии. Процессор запрашивает либо операцию чтения, либо операцию записи, и запрошенные данные пересылаются по линиям данных.

Как отмечалось выше данные, которыми устройство обменивается с процессором, хранятся в *регистрах данных*. Специальный регистр – *регистр состояния*, содержит информацию, относящуюся к функционированию устройства ввода-вывода. Регистры данных и состояния соединяются шиной данных, и им присваиваются уникальные адреса. Дешифратор адреса, регистры данных и состояния, управляющие схемы, необходимые для координирования операций ввода-вывода, составляют *схему сопряжения*, или *интерфейс*, устройства.

Предположим, что необходимо считать вводимые с клавиатуры символы и вывести их на экран. Простейший способ выполнения этих задач заключается в использовании метода, называемого *программно-управляемым вводом-выводом*. Оба устройства связываются с процессором с помощью шины. В ответ на нажатие клавиши соответствующий код символа сохраняется в буферном регистре, связанном с клавиатурой. Дополнительно в 1 устанавливается специальный флаг, оповещающий процессор о происшедшем событии, после чего и осуществляется пересылка символа в память. Реализация вывода символа на экран осуществляется аналогично. Опрос флага состояния осуществляется в цикле, периодически через короткие интервалы времени, что позволяет синхронизировать работу процессора с работой внешнего устройства.

Существует еще два распространенных механизма реализации ввода-вывода: *прерывания* и *прямой доступ к памяти*. При использовании прерываний синхронизация достигается за счет того, что устройство ввода-вывода само сообщает о своей готовности, отправляя через шину специальный сигнал. Прямой доступ к памяти является характерным для высокоскоростных устройств ввода-вывода. Эта технология позволяет интерфейсным схемам устройства самостоятельно обмениваться данными с памятью, без постоянного участия процессора.

3.2. ПРЕРЫВАНИЯ. ОБРАБОТКА ПРЕРЫВАНИЙ

Обычно программа выполняет цикл ожидания, постоянно проверяя состояние устройства. В течение этого времени процессор не делает никакой полезной работы. Для устранения этого недостатка нужно так организовать совместную работу процессора и внешнего устройства, чтобы это устройство само оповещало процессор о своей готовности к пере-

сылке данных. Такое оповещение выполняется с помощью специального сигнала, называемого *прерыванием*. Для прерываний обычно выделяется как минимум одна управляющая линия шины, называемая *линией запроса прерывания*.

Программа, выполняемая в ответ на запрос прерывания, называется *программой обработки прерываний*. Устройство, запросившее прерывание, может идентифицировать себя с помощью специального кода, пересылаемого процессору по шине, что позволяет процессору идентифицировать отдельные устройства даже в том случае, если они используют одну линию запроса прерывания. Этот код способен определять начальный адрес программы обработки прерывания, предназначенной для данного устройства. При этом предполагается, что программа обработки прерываний от конкретного устройства всегда располагается по одному и тому же адресу. То место в памяти, на которое указывает вызвавшее прерывание устройство, используется для хранения начального адреса программы обработки прерывания. Процессор считывает этот адрес, называемый *вектором прерывания*, и загружает его в регистр адреса команд (счетчика команд, PC). Кроме адреса вектор прерывания может содержать новое значение регистра состояния процессора (PS).

Предположим, что запрос прерывания поступает во время выполнения некоторой команды. Процессор завершает выполнение этой команды, а затем загружает в регистр PC адрес первой команды программы обработки прерываний. После выполнения программы обработки прерываний процессор должен вернуться к следующей команде. Для этого перед вызовом программы обработки прерывания должно быть временно сохранено содержимое регистра PC. Команда возврата из прерывания, расположенная в конце программы обработки прерывания, загрузит этот сохраненный адрес в регистр PC, в результате чего выполнение прерванной программы будет продолжено со следующей команды. К числу сохраняемой и восстанавливаемой информации обычно относятся значения флагов условий и содержимое всех тех регистров, которые используются и прерванной программой, и программой обработки прерывания.

Обрабатывая прерывание, процессор должен проинформировать устройство о том, что его запрос распознан, после чего данное устройство сможет снять сигнал запроса прерывания. Для этого по шине может быть передан специальный управляющий сигнал.

Поскольку запросы на прерывание могут поступать в любой момент, выполнение программ обработки прерываний должно тщательно контролироваться. Одной из основных возможностей, которой должен обладать любой компьютер, является возможность запрещать и разре-

шать прерывания по мере необходимости. Существует несколько механизмов решения этой проблемы.

Первый механизм заключается в игнорировании схемой процессора сигнала на линии запроса прерывания до окончания выполнения первой команды в программе обработки прерывания. При этом первая команда программы обработки должна запретить прерывания до окончания действия данной программы. Как правило, команда, разрешающая прерывания, является последней командой программы его обработки (она предшествует команде возврата из прерывания). При этом процессор должен гарантировать, что выполнение команды возврата из прерывания будет завершено до того, как станут возможными следующие прерывания.

Второй механизм в большей мере подходит для простого процессора с единственной линией запроса прерывания. Он заключается в том, что процессор сам запрещает прерывания перед началом выполнения программы обработки прерываний и разрешает таковые по ее завершении. После сохранения в стеке регистра PC и регистра PS процессор выполняет действия, эквивалентные команде запрета прерываний. Очень часто для запрета и разрешения прерываний используется один разряд в регистре PS, называемый *флагом разрешения прерываний* (interrupt-enable). Если прерывания разрешены, этот разряд содержит 1, а если запрещены – 0. После сохранения в стеке регистра PS, в котором разряд разрешения прерываний установлен в 1, процессор очищает этот разряд в своем регистре PS, запрещая тем самым дальнейшие прерывания. При выполнении команды возврата из прерывания содержимое регистра PS восстанавливается из стека и флаг разрешения прерываний снова становится равным 1. Это значит, что прерывания разрешены.

Третий механизм предполагает, что у процессора имеется специальная линия запроса прерываний и что схема управления прерываниями отвечает только на передний фронт сигнала. Эта линия называется управляемой фронтом сигнала. При такой схеме работы процессор получает только один запрос прерывания, независимо от того, как долго линия остается активной. Это значит, что повторяющихся прерываний быть не может и нет необходимости явно отключать запросы прерывания на данной линии.

Определим еще раз коротко последовательность событий, происходящих в ходе обработки запроса прерывания от одного устройства. Если предположить, что изначально прерывания разрешены, эта последовательность будет следующей.

1. Устройство генерирует запрос прерывания.
2. Процессор прерывает текущую выполняемую программу.

3. Последующие прерывания запрещаются, для чего изменяются управляющие биты в регистре PS (за исключением схем, в которых линия запроса прерывания управляется фронтом сигнала).

4. Устройство информируется о том, что его запрос распознан, и в ответ сбрасывает сигнал запроса на прерывание.

5. Запрошенное прерыванием действие выполняется программой обработки прерывания.

6. Прерывания разрешаются, выполнение программы продолжается.

Если на время выполнения программы обработки прерывания все прерывания запрещены, то запрос от одного устройства не сможет вызвать более одного прерывания. Этот принцип часто используется и в тех случаях, когда в системе имеется несколько устройств. В результате их прерывания обрабатываются по очереди, а начатая программа обработки прерывания выполняется до конца, до того как процессор получает запрос прерывания от другого устройства. Однако в некоторых случаях большая задержка с ответом на запрос прерывания может привести к неверному функционированию устройств. Поэтому, для правильной организации ввода-вывода используется система приоритетов устройств. Во время обслуживания процессором прерывания от одного устройства им должны приниматься запросы прерываний только от устройств с более высоким приоритетом.

3.3. ПРЯМОЙ ДОСТУП К ПАМЯТИ

Как отмечалось выше, пересылка данных между процессором и устройствами ввода-вывода будет осуществляться лишь после того, как процессор определит, что устройство ввода-вывода готово к очередной операции. Для этого процессор опрашивает флаг состояния в интерфейсе устройства или ждет, пока устройство само направит ему запрос на прерывание. В любом случае производится много лишней работы. Поэтому для быстрой пересылки больших блоков данных применяется другой подход. Компьютер может содержать специальное управляющее устройство, позволяющее пересылать блоки данных между внешним устройством и основной памятью без постоянного участия процессора. Эта технология называется *прямым доступом к памяти* (ПДП), по-английски – Direct Memory Access (DMA).

Операции ПДП выполняются управляющей схемой, входящей в состав интерфейса устройства ввода-вывода. Эта схема называется *контроллером* ПДП. Контроллер ПДП выполняет ту же задачу, что и процессор, обращающийся к основной памяти. Для каждого пересылаемого

слова он генерирует адрес памяти и сигналы шины, управляющие пересылкой данных. Поскольку контроллер ПДП производит пересылку блоков данных, он сам увеличивает адрес, по которому будет записываться каждое следующее слово, и отслеживает количество таких операций.

Хотя контроллер ПДП работает без участия процессора, он управляется выполняемой процессором программой. В частности, чтобы инициировать пересылку блока слов, процессор пересылает контроллеру начальный адрес этого блока, сведения о количестве составляющих его слов и направлении пересылки. Получив такую информацию, контроллер приступает к выполнению операции. По окончании пересылки он информирует об этом процессор с помощью сигнала прерывания.

3.4. ШИННАЯ АРХИТЕКТУРА. АРБИТРАЖ

Как указывалось выше, процессор, основная память и устройства ввода-вывода могут соединяться между собой посредством общей шины, основным назначением которой является предоставление канала связи для пересылки данных. Шина содержит линии для поддержки прерываний, арбитража и пересылки данных, которые бывают трех типов: линии данных, линии адреса и управляющие линии. Для пересылки данных по шине используются шинные протоколы. *Шинный протокол* – это набор правил, управляющих поведением соединенных с шиной устройств, а также последовательностью помещения информации на шину, выдачи управляющих сигналов и т. п.

Управляющие сигналы определяют, какую операцию, чтения или записи, следует выполнить. Сигналы управления шиной также используются для тактирования операций. Они определяют, в какой момент процессор и устройства ввода-вывода могут поместить данные на шину или прочитать их с таковой. Для тактирования пересылки данных по шине разработано множество схем, которые можно разделить на два основных типа: *синхронные* и *асинхронные*. В операциях пересылки данных по шине одно из устройств играет роль хозяина шины (bus master). Это устройство инициирует пересылку данных с помощью команд чтения или записи. Обычно хозяином шины является процессор, но эту роль могут выполнять и другие устройства, поддерживающие функцию прямого доступа к памяти. Устройство, к которому обращается хозяин шины, называется *подчиненным* или *целевым*.

В случае *синхронной* шины все устройства получают синхронизирующую информацию по общей тактовой линии. На эту линию подаются тактовые импульсы со строго фиксированной частотой. Промежуток

времени между последовательными тактовыми импульсами в простейшей синхронной шине составляет *цикл шины*, в течение которого выполняется одна операция пересылки данных.

В *асинхронных* шинах пересылка данных по шине основывается на механизме *квитирования*, то есть подтверждения связи, между хозяином шины и подчиненным устройством. В схеме с квитированием тактовая линия заменяется двумя управляющими линиями синхронизации: Master-ready и Slave-ready. Первая принадлежит хозяину шины, который передает по ней сигнал готовности к транзакции, а по второй отвечает подчиненное устройство. Пересылка данных, управление которой осуществляется посредством протокола с квитированием, выполняется следующим образом. Хозяин шины помещает на нее адрес и информацию о команде. Затем по линии Master-ready он сообщает об этом всем устройствам. В ответ подключенные к шине устройства декодируют адрес. То устройство, для которого предназначена команда, выполняет таковую и информирует об этом хозяина шины по линии Slave-ready. Хозяин дожидается этого сигнала и только после этого удаляет с шины свои сигналы. В случае операции чтения он считывает данные в свой входной буфер.

Выбор очередного устройства, которое станет хозяином шины, осуществляется посредством специальной процедуры, получившей название *арбитраж*. При этом учитываются потребности различных устройств, для чего опять-таки используется система приоритетов. Существует два подхода к разрешению конфликтов на шине: *централизованный* арбитраж и *распределенный* арбитраж. При централизованном подходе разрешение конфликтов выполняется *арбитром шины*, в качестве которого может служить как процессор, так и какое-либо отдельное устройство, подключенное к шине. При *распределенном арбитраже* все устройства, ожидающие своей очереди на использование шины, на равных правах участвуют в арбитражном процессе.

3.5. ИНТЕРФЕЙСНЫЕ СХЕМЫ

Интерфейс устройства ввода-вывода представляет собой схему, соединяющую устройство с шиной компьютера. По одну сторону этой схемы расположены сигналы шины для адреса и данных, а также управляющие сигналы, по другую сторону, называемую *портом*, – линии для передачи данных и управляющих сигналов между интерфейсом и устройством ввода-вывода. Порт может быть параллельным или последовательным. Параллельный порт одновременно пересылает от устройства или к устройству группу битов данных, обычно 8 или 16. Последова-

тельный порт пересылает данные по одному биту за один раз. Взаимодействие с шиной в обоих случаях осуществляется по одному принципу, а взаимопреобразование последовательного и параллельного форматов выполняется внутри интерфейсной схемы.

Перечислим функции интерфейса ввода-вывода, который

- предоставляет буфер для хранения как минимум одного слова данных (или одного байта, как в случае байт-ориентированных устройств);
- содержит доступные процессору флаги состояния, по которым тот может определить, заполнен буфер (в случае ввода данных) или он пуст (в случае вывода);
- содержит схему декодирования адреса, позволяющую устройству определить, когда оно адресуется процессором;
- генерирует тактовые сигналы, необходимые для функционирования схемы управления шиной;
- выполняет преобразование формата, если таковое требуется для пересылки данных между шиной и устройством ввода-вывода (например, преобразование из параллельного формата в последовательный, необходимое для отправки данных через последовательный порт).

Наличие самых разнообразных моделей устройств ввода-вывода и конструкций шины компьютера потребовало разработки стандартных интерфейсов, сигналов и протоколов, позволяющие подключать одни и те же внешние устройства к самым разным компьютерам.

Типичный ПК состоит из большой печатной платы, называемой материнской. На ней располагается микросхема процессора, основная память, несколько интерфейсов ввода-вывода, а также имеется несколько разъемов для подключения дополнительных интерфейсов.

Шина процессора – это шина, управляемая теми же сигналами, что и микросхема процессора. К ней могут быть подключены устройства, которым требуется очень высокая скорость взаимодействия с процессором, и в частности основная память. На материнской плате обычно имеется еще одна шина – *шина расширения*, способная поддерживать большее количество устройств. Эти две шины соединены между собой с помощью специальной схемы, называемой *мостом* и предназначенной для преобразования сигналов в соответствии с протоколами, регулирующими применение этих двух шин. Устройства, подключенные к шине расширения, представляются процессору непосредственно соединенными с его собственной шиной.

Универсальный стандарт для шины процессора определить невозможно, поскольку ее структура очень тесно связана с архитектурой про-

цессора. Однако на шину расширения эти ограничения не распространяются, поэтому для нее можно использовать стандартную схему сигналов. Для шин расширения разработан целый ряд стандартов ведущими производителями. Некоторые из них появлялись «естественным» путем, когда конкретная архитектура завоевывала популярность на рынке. Например, IBM разработала для своего персонального компьютера шину ISA (Industry Standard Architecture), которая стала настолько популярной, что производители устройств ввода-вывода стали снабжать свои устройства ISA-совместимыми интерфейсами, и ISA стал стандартом де-факто.

Некоторые стандарты разрабатывались объединенными усилиями крупных компаний, которые, хотя и конкурировали на рынке вычислительной техники, были заинтересованы в создании совместимых устройств. В некоторых случаях эти стандарты, одобренные такими организациями, как IEEE (Institute of Electrical and Electronics Engineers – Институт инженеров по электротехнике и электронике), ANSI (American National Standards Institute – Национальный институт стандартизации США), и даже международными организациями, в частности ISO (International Organization for Standardization – Международная организация по стандартизации), получали официальный статус.

Отметим три широко применяемых стандартов шин: PCI (Peripheral Component Interconnect), SCSI (Small Computer Systems Interface) и USB (Universal Serial Bus). Стандарт PCI определяет шину расширения на материнской плате. Шины стандарта SCSI и USB предназначены для подключения дополнительных устройств как внутри, так и вне корпуса компьютера. SCSI представляет собой высокоскоростную параллельную шину, предназначенную для подключения таких устройств, как диски и дисплеи. Шина USB поддерживает последовательную передачу данных. Она используется для подключения самого разнообразного оборудования, от клавиатур, микрофонов, цифровых видеокамер до игровых устройств, а также для Интернет-соединений.

Шина PCI – это разновидность системной шины, появившейся в ответ на потребность в стандартизации используемых устройств. Она поддерживает функции, типичные для шины процессора, но в стандартизированном формате, независимо от типа процессора. Подключенные к этой шине устройства представляются процессору непосредственно соединенными с его собственной шиной.

Создавалась шина PCI как недорогое устройство, по-настоящему независимое от процессора. Ее конструкция была обусловлена назревшей потребностью в поддержке высокоскоростных дисковых и графических устройств, а также специфическими нуждами мультипроцессорных сис-

тем. Благодаря этому PCI до сих пор, после своего появления в 1992 году, популярна как промышленный стандарт.

Важной особенностью шины PCI было то, что она предоставила новый механизм подключения устройств ввода-вывода, получившего название plug-and-play. Для подключения к системе нового устройства пользователю теперь достаточно вставить интерфейсную плату в разъем на шине. Все остальное сделает за него программное обеспечение.

В одном компьютере может использоваться несколько разных шин. Так, типичный компьютер Pentium содержит шины PCI и ISA, что позволяет пользователю выбирать устройство для подключения из достаточно широкого диапазона устройств.

4. ОРГАНИЗАЦИЯ ПАМЯТИ

Одна из базовых компонент архитектуры – организация памяти и способы взаимодействия информации на ее различных уровнях.

4.1. ИЕРАРХИЯ ПАМЯТИ ЭВМ

Одним из определяющих факторов производительности компьютера является время взаимодействия процессора с памятью, которое определяется ее строением, объемом и архитектурой подсистем доступа в память. Максимальный размер памяти, который может использоваться компьютером, определяется его системой адресации. К примеру, 32-разрядный компьютер, генерирующий 32-разрядные адреса, может иметь память объемом до 2^{32} = 4 Гбайт адресуемых единиц. Большинство компьютеров адресуют память по-байтово. Но обычно память разрабатывается с учетом того, что данные извлекаются и считываются не байтами, а словами. Понятие длины слова чаще всего определяется как количество битов, сохраняемых или считываемых за одно обращение к памяти.

В большинстве современных компьютеров в качестве организации наиболее эффективного доступа к памяти используется так называемая многоуровневая иерархическая память**.** В качестве уровней используются *регистровая память*, *кэш-память*, основная *оперативная память*, *виртуальные* и *жесткие* диски. При этом выдерживается следующий принцип формирования иерархии: при повышении уровня памяти скорость обработки данных должна увеличиваться, а объем уровня памяти – уменьшаться. Эффективность использования такого рода иерархии достигается за счет хранения часто используемых данных в памяти верхнего

уровня, время доступа к которой минимально. Как правило, в компьютере используются все типы памяти. Быстрее всего осуществляется доступ к данным, хранящимся в регистрах процессора, но по объему они составляют ничтожно малую часть всей памяти компьютера.

На следующем уровне иерархии располагается сравнительно небольшой объем быстрой памяти, называемый *кэшем*, содержащий копии команд и данных, хранящихся во внешней по отношению к процессору основной памяти. Обычно в компьютере имеется два уровня кэш-памяти. Первичный кэш располагается на микросхеме процессора и называется *кэшем первого уровня* (L1). Вторичный кэш имеет больший объем и располагается между первичным кэшем и остальной памятью и называется *кэшем второго уровня* (L2). Однако бывает, что микросхема процессора вообще не содержит кэша или же, напротив, содержит оба кэша: L1 и L2.

Ниже по иерархии располагается *основная память*. Она значительно больше и намного медленнее кэша. На последнем уровне иерархии находятся внешние запоминающие устройства, которые предоставляют огромный объем недорогой памяти, но по сравнению с полупроводниковыми устройствами являются очень медленными.

4.2. ПОЛУПРОВОДНИКОВАЯ RAM-ПАМЯТЬ

Для реализации основной памяти компьютера используются полупроводниковые интегральные схемы, обеспечивающие произвольный доступ (Radom Access Memory, RAM) к любому адресу памяти. Полупроводниковая память реализуется в виде микросхем с очень разным быстродействием. Длительность их цикла варьируется от 100 до менее чем 10 нс. Память на основе микросхем, которые могут сохранять свое состояние лишь до тех пор, пока к ним подключено питание, называется *статической* (Static RAM, SRAM). Статическая RAM работает быстро, но стоит очень дорого, поскольку каждая ее ячейка содержит несколько транзисторов. Вот почему выпускается еще и более дешевая память с более простой конструкцией ячеек. Однако эти ячейки не способны бесконечно долго сохранять свое состояние, поэтому такая память называется *динамической* (Dynamic RAM, DRAM).

В ячейке динамической памяти информация хранится в форме заряда на конденсаторе, и этот заряд может сохраняться всего несколько десятков миллисекунд. Поскольку ячейка памяти должна хранить информацию гораздо дольше, ее содержимое должно периодически обновляться путем восстановления заряда на конденсаторе. Обычно эта работа ав-

томатически выполняется с помощью специальной схемы, называемой *схемой регенерации*.

Статическая RAM обычно используется только в тех случаях, когда на первом месте стоит скорость работы системы. Схемы реализации ее базовых ячеек достаточно сложны, из-за чего стоимость и размер микросхем получаются очень большими. Как правило, статическая RAM применяется для реализации кэш-памяти второго уровня. Для реализации основной памяти в большинстве компьютеров используется динамическая RAM. Такие микросхемы характеризуются очень высокой плотностью, благодаря чему память достаточно большого объема имеет приемлемую стоимость.

Данные между памятью и процессором, а точнее, между памятью и кэшем процессора, пересылаются в виде слов или небольших блоков слов. Большие блоки, составляющие страницы данных, пересылаются между памятью и дисками.

Быстродействие памяти характеризуется интервалом времени между инициированием операции и ее завершением. Это время определяют как *время доступа к памяти*. Еще одной важной характеристикой быстродействия памяти является *цикл памяти* – минимальный промежуток времени между моментами начала двух последовательных операций с памятью, например, между двумя последовательными операциями чтения.

Термином *время ожидания памяти* или *латентностъ* (latency) определяется время, уходящее на пересылку в память или из памяти одного слова данных. Если данные считываются и записываются по одному слову, латентность полностью характеризует производительность памяти.

Поскольку блоки имеют разную длину, производительность можно определять количеством битов или байтов, пересылаемых за одну секунду. Эту характеристику называют *пропускной способностью* памяти.

Микросхемы SRAM и DRAM являются энергозависимыми, и как только питание выключается, хранящаяся в них информация попросту исчезает. Однако существует множество устройств и компонентов, которым требуются запоминающие устройства, сохраняющие информацию и после выключения питания. Примерами таких устройств могут служить жесткие диски обыкновенных компьютеров, предназначенные для хранения огромных объемов информации. Кроме этого компьютер обычно содержит небольшую энергонезависимую память, в которой хранятся команды, выполняемые при включении компьютера первыми и обеспечивающие копирование программ загрузки операционной системы с диска в основную память.

Существуют разные типы энергонезависимой памяти. Как правило, содержимое памяти считывается так же, как из SRAM и DRAM, а вот для его записи применяется специальная процедура. Так как в рабочем режиме содержимое такой памяти только считывается, поэтому она называется *памятью, доступной только для чтения* (Read Only Memory, ROM), или постоянной памятью.

Некоторые микросхемы ROM разрабатываются таким образом, что данные в них может записывать пользователь. В этом случае память называется *программируемой ROM* (Programmable ROM, PROM).

Еще один тип микросхем ROM позволяет не только записывать, но и перезаписывать данные. Такая память обычно называется *стираемой перепрограммируемой ROM* (Erasable Programmable ROM, EPROM).

Кроме основной оперативной памяти и постоянной памяти, в компьютере имеется также небольшой участок памяти для хранения параметров конфигурации компьютера. Его часто называют CMOS-памятью, поскольку эта память обычно выполняется по технологии CMOS (complementary metaloxide semiconductor), обладающей низким энергопотреблением. Содержимое CMOS-памяти не изменяется при выключении электропитания компьютера, поскольку для ее электропитания используется специальный аккумулятор. Для изменения параметров конфигурации компьютера в BIOS содержится программа настройки конфигурации компьютера – SETUP. Аккумулятор, питающий CMOS-память, снабжает электроэнергией и встроенные в компьютер часы (так называемые часы реального времени).

Одна из сравнительно недавних разработок памяти получила название *флэш-памяти*. Флэш-память имеет большую плотность ячеек, а следовательно, большую емкость и меньшую стоимость в пересчете на бит. Для нее достаточно напряжения питания одного уровня, и к тому же она более экономична.

Благодаря своей экономичности флэш-память удобна для использования в портативных системах, работающих на батареях: портативных компьютерах, сотовых телефонах, цифровых видеокамерах и MP3-плеерах. Однако чаще всего в них используются большие модули, состоящие из множества микросхем. Существует две популярные разновидности таких модулей: флэш-карты и флэш-диски.

Для создания флэш-карты флэш-микросхемы можно вмонтировать в небольшую карту. Такие карты имеют стандартный интерфейс, благодаря чему их можно использовать в самых разных устройствах. Существуют и более крупные модули флэш-памяти, предназначенные для замены

жестких дисков. Каждый такой модуль полностью эмулирует жесткий диск и может вставляться в предназначенный для него отсек.

4.3. ОРГАНИЗАЦИЯ И ФУНКЦИОНИРОВАНИЕ КЭШ-ПАМЯТИ

Поскольку по сравнению с быстродействием современных процессоров скорость функционирования основной памяти очень мала, необходимы архитектурные решения, сокращающие время доступа к требуемой информации. Таким решением является использование кэш-памяти, благодаря которой основная память представляется процессору более быстрой, чем есть на самом деле.

Эффективность механизма кэширования основывается на свойстве компьютерных программ, называемом *локализацией ссылок*. Анализ процесса реализации различных программ показывает, что большую часть времени в них выполняется код, в котором определенные группы команд повторяются по многу раз. Локализация ссылок происходит и во времени, и в пространстве. Локализация во времени означает, что недавно выполнявшиеся команды, скорее всего, очень скоро будут выполнены снова. А локализация в пространстве означает большую вероятность того, что очень скоро будут выполнены команды, расположенные в непосредственной близости от только что реализованных команд (имеется в виду близость адресов команд).

Если поместить активные сегменты программы в быструю кэш-память, общее время их выполнения сократится. Идея кэширования команд очень проста. Управляющие схемы памяти разрабатываются таким образом, чтобы можно было использовать свойство локализации ссылок. Исходя из принципа локализации во времени, каждый элемент, к которому обращается процессор, будь то команда программы или элемент данных, копируется в кэш, где он остается до тех пор, пока не потребуется снова. Исходя из принципа локализации в пространстве, в кэш копируется не только текущий элемент программы или данных, но еще и несколько близлежащих элементов. Набор элементов с последовательными адресами определенного размера называется блоком или строкой кэша.

Если процессор выдает запрос чтения, содержимое блока памяти считывается по заданному адресу по одному слову в кэш. Когда впоследствии программа обратится к любому элементу этого блока, он будет прочитан не с диска, а прямо из кэша. Соответствие между блоками в основной памяти и блоками в кэше определяется *функцией отображения*. Когда кэш полон и производится обращение к отсутствующему в нем

слову памяти (команде или данным), управляющее кэшем аппаратное обеспечение должно решить, какой из блоков удалить из кэша, чтобы добавить в него новый блок, содержащий требуемое слово. Набор правил для принятия такого решения составляет *алгоритм замещения*

Процессор выдает запросы чтения и записи, используя адреса, которые указывают на память. В ответ схема управления кэшем выясняет, имеется ли в таковом запрошенное слово. Если да, то в операции чтения или записи задействуется слово из кэша. При этом говорят, что имеет место *попадание в кэш*. Если же выполняется операция записи, система может действовать одним из двух способов. При использовании первого из них, называемого *протоколом сквозной записи*, предполагается, что кэш и основная память обновляются одновременно. Второй способ подразумевает, что данные обновляются только в кэше, после чего они помечаются с помощью соответствующего битового флага. Упомянутый бит называют *флагом изменения* или *модификации*. Соответствующее слово в основной памяти обновляется позже, при удалении из кэша того блока, который содержит помеченное слово. Описанная технология называется *протоколом обратной записи* или *обратного копирования*. Протокол сквозной записи проще, но при его использовании производятся лишние операции записи в основную память – в том случае, если некоторое слово обновляется в кэше несколько раз подряд. Конечно, подобное происходит и при обратной записи, поскольку при удалении блока из кэша в память записываются все его слова, даже если изменилось только одно из них.

Ситуация, при которой слово, адресуемое операцией считывания, отсутствует в кэше, называется *промахом чтения*. В этом случае из основной памяти в кэш копируется блок, содержащий такое слово. Запрошенное слово передается в процессор после загрузки в кэш всего блока. В качестве альтернативы оно может быть передано в процессор сразу после его прочтения из основной памяти. Последний подход, называемый *сквозной загрузкой*, сокращает время ожидания слова процессором, но для его реализации требуется более сложная схема.

Если в кэше не оказывается слова, адресуемого операцией записи, возникает ситуация, называемая *промахом записи*. В таком случае, при использовании протокола сквозной записи, информация записывается прямо в основную память. Но если применяется протокол обратной записи, в кэш сначала копируется блок, содержащий заданное слово, а затем это слово перезаписывается в кэше и помечается как измененное.

Рассмотрим, как происходит выборка блоков данных из памяти в кэш. Существует несколько стратегий.

1. Простейшим способом сопоставления адресов блоков в кэше и в памяти является *прямое отображение*. При использовании этой технологии блок *j* основной памяти отображается на блок *j* по модулю 128 кэша. Таким образом, когда загружается один из блоков основной памяти, начинающихся по адресам 0,128, 256 и т. д., он записывается в блок кэша 0. Блоки 1, 129, 257 и т. д. записываются в блок кэша 1 и т. д. Поскольку на каждый блок кэша отображается более одного блока основной памяти, то даже при не до конца заполненном кэше может возникнуть состязание за некоторую позицию. Например, команды программы, начавшиеся в блоке 1, после перехода могут продолжиться в блоке 129. В результате выполнения программы оба эти блока должны быть скопированы в блок 1 кэша. При этом старый блок заменяется новым.

2. При более гибком методе отображения – *ассоциативном отображении*, блок основной памяти можно помещать в любой блок кэша. При выполнении программы теговые биты сгенерированного процессором адреса по очереди сравниваются с теговыми битами каждого блока кэша. Если совпадение найдено, значит, содержащий данное слово блок уже присутствует в кэше. Новые блоки заменяют уже хранящиеся в кэше только в том случае, если кэш заполнен, причем для этой цели необходим алгоритм выбора удаляемого блока.

3. Технологии прямого и ассоциативного отображения могут использоваться совместно. В этом случае блоки кэша объединяются в множества, и каждый блок основной памяти может располагаться в любом из блоков определенного множества. Такой подход называется *множественно-ассоциативным отображением*. Причем в данном случае вероятность конфликтов, являющихся одним из недостатков прямого отображения, значительно снижается.

Для каждого блока в кэше должен храниться еще один управляющий бит, называемый *битом достоверности*. Он указывает, содержит ли блок достоверные данные. Его не следует путать с упоминавшимся ранее битом изменения, указывающим, был ли блок модифицирован за то время, пока он находится в кэше.

Бит достоверности используется при ПДП-пересылке данных из основной памяти на диск, если используется кэш с обратной записью. Когда блок кэша в первый раз загружается из основной памяти, его бит достоверности устанавливается в 1. Если блок основной памяти обновляется из другого источника, минуя кэш, система проверяет, находится ли загружаемый блок в кэше. Если да, его бит достоверности устанавливается в 0, чтобы в кэше не оказалось устаревших данных. Иногда данные, находящиеся в памяти, могут не отражать изменений, внесенных в кэ-

шируемую копию, вследствие чего перед их копированием на диск необходимо записать измененные данные из кэша в основную память. Обязательное использование двумя разными элементами (в данном случае процессором и подсистемой ПДП) одинаковых копий данных называется *согласованностью кэша*.

Рассмотрим *алгоритмы замещения*. В кэше с прямым отображением позиция каждого блока определена раз и навсегда, поэтому никакая особая стратегия замены блоков ему не требуется. А вот в ассоциативном или множественно-ассоциативном кэше замена блоков может выполняться по-разному. Когда в кэш нужно поместить новый блок, но свободной позиции для него там не окажется, контроллер кэша должен выбрать один из старых блоков для перезаписи. От того, как он будет решать эту задачу, зависит производительность системы. Главная идея, которой следует руководствоваться, принимая такое решение, состоит в следующем: в памяти должны оставаться блоки, для которых вероятность того, что они понадобятся в ближайшем будущем, максимальна. Но как их определить? Здесь можно опереться на принцип локализации ссылок. Так как повторяющиеся команды, лежащие в пределах некоторой области, выполняются в течение определенного времени, существует большая вероятность того, что блоки, обращение к которым производилось недавно, очень скоро потребуются снова. Поэтому, когда требуется освободить место для нового блока, имеет смысл удалить из кэша тот блок, к которому дольше всего не было обращений. Алгоритм работы этого блока называется алгоритмом удаления наиболее давно использовавшихся элементов (Least Recently Used, LRU).

Для использования алгоритма LRU контроллер кэша должен отслеживать обращения ко всем блокам кэша. Алгоритм LRU очень популярен. В большинстве случаев он работает прекрасно, но иногда его применение может привести к снижению производительности, например, при обращении к последовательным элементам массива, который слишком велик и не помещается в кэше целиком. Для того чтобы повысить производительность алгоритма, можно внести в него некоторую долю случайного выбора.

На практике используются и некоторые другие алгоритмы замещения. Правило замены самого «старого» блока кажется наиболее логичным, но оно не принимает в расчет частоту обращений к хранящимся в кэше блокам. Поэтому оно не так эффективно, как алгоритм LRU. Самым простым решением является случайный выбор перезаписываемого блока, и, что интересно, практика показывает его эффективность.

Отличным показателем эффективности конкретной реализации иерархии памяти является частота попаданий при доступе к информации на разных уровнях иерархии. Напомним, что попаданием называется успешное обращение к данным в кэше. Отношение количества попаданий к общему количеству попыток доступа называется *частотой попаданий*, а отношение количества промахов к общему количеству попыток доступа – *частотой промахов*.

Производительность системы очень зависит от того, какие действия выполняются в случае промаха. Дополнительное время, уходящее на копирование необходимой информации в кэш, *называется накладными расходами при промахах*. Эти накладные расходы выливаются в простой процессора, поскольку он не может функционировать без команды или обрабатываемых ею данных.

При использовании протокола сквозной записи каждой операции записи соответствует операция сохранения нового значения в основной памяти. Если процессор будет ждать окончания этой операции, его работа очень сильно замедлится. Однако дальнейшая работа процессора в ближайшее время, как правило, не зависит от результата операции записи, так что ему незачем дожидаться ее завершения. А раз так, в систему памяти можно добавить буфер записи, предназначенный для временного хранения результатов операций записи. Процессор должен поместить в него запрос на запись и продолжать свою работу. А тем временем, уже без его участия, запрос будет отослан в основную память, когда та не будет занята выполнением запросов на чтение. Запросы на чтение должны обслуживаться немедленно, поскольку процессор обычно не может продолжать работу без тех данных, которые он запросил. Поэтому у запросов на чтение более высокий приоритет, чем у запросов на запись.

4.4. КОНЦЕПЦИЯ ВИРТУАЛЬНОЙ ПАМЯТИ И ЕЕ ОРГАНИЗАЦИЯ

Еще одной важной концепцией, связанной с организацией памяти, является концепция *виртуальной памяти*. До сих пор предполагалось, что генерируемые процессором адреса полностью соответствуют физическим ячейкам памяти. Однако на практике это не всегда так. По определенным причинам данные могут храниться не по тем адресам, которые заданы в программе. Схемы управления памятью преобразуют указанный в программе адрес в другой адрес, используемый для доступа к физической памяти. В таком случае сгенерированный процессором адрес называют *виртуальным* или *логическим*. Виртуальное адресное про-

странство определенным образом отображается на физическую память, в которой хранятся данные. Отображение выполняется с помощью специальных управляющих схем памяти, часто называемых *блоком управления памятью*. В ходе выполнения программы функция отображения может меняться в соответствии с системными требованиями.

Виртуальная память предназначена для увеличения видимого компьютером объема физической памяти. Виртуальное адресное пространство и объем расположенных в нем данных могут быть настолько большими, насколько позволяют возможности адресации используемого процессора. Причем в каждый конкретный момент только часть этого адресного пространства отображается на физическую память. Остальные виртуальные адреса соответствуют устройствам массовой памяти – как правило, магнитным дискам. Когда выполняющейся программе требуется доступ к данным, адреса которых не отображаются на реальную память, блок управления памятью изменяет функцию отображения и пересылает данные с диска в основную память. Поэтому в каждом цикле обращения к памяти система обработки адресов определяет, находится ли адресуемая информация в основной памяти компьютера. Если да, происходит обращение к соответствующему слову памяти и выполнение программы продолжается. Если нет, с диска в память пересылается страница, содержащая нужное слово. Эта страница заменяет в памяти какую-нибудь другую страницу, пока не нужную программе. Поскольку на пересылку страниц между памятью и диском уходит некоторое время, при частом перемещении страниц скорость работы компьютера снижается. Но если выбор заменяемых страниц выполняется продуманно, вероятность того, что необходимые страницы окажутся в основной памяти, возрастет.

Простейший метод преобразования виртуальных адресов в физические основывается на предположении, что все программы и данные состоят из сегментов фиксированной длины, называемых *страницами*, которые, в свою очередь, состоят из блоков слов, последовательно расположенных в памяти. Размер страницы обычно варьируется от 2 до 16 Кбайт. Страница является базовой единицей информации, перемещаемой между основной памятью и диском по требованию механизма преобразования адресов. Страницы не должны быть слишком маленькими, поскольку время доступа к магнитному диску (составляет несколько миллисекунд) намного больше времени доступа к основной памяти. Значительная часть этого времени уходит на поиск данных на диске. Найденные данные пересылаются со скоростью несколько мегабайт в секунду. С другой стороны, если страница слишком велика, большая ее часть, ско-

рее всего, не будет использована, но место в основной памяти она, конечно же, будет занимать

Каждый сгенерированный процессором виртуальный адрес, будь то адрес для операции выборки команды или для чтения и записи операнда, интерпретируется как *номер виртуальной страницы* (старшие разряды) и *смещение* (младшие разряды) байта или слова от начала страницы. Информация о местонахождении каждой страницы в основной памяти содержится в *таблице страниц*. Она включает адрес основной памяти, по которому хранится страница, и данные о ее текущем состоянии. Область основной памяти, где может находиться одна страница, называется *страничным блоком*. Начальный адрес таблицы страниц хранится в *базовом регистре таблицы страниц*. Добавив номер виртуальной страницы к содержимому этого регистра, можно получить адрес нужного элемента таблицы страниц. А в самом этом элементе хранится начальный адрес страницы, если, конечно, она имеется в основной памяти.

Кроме адреса страницы каждый элемент таблицы страниц содержит несколько управляющих битов, которые определяют состояние страницы, находящейся в основной памяти, и еще один бит, указывающий, хранится ли страница в памяти. Последний бит позволяет операционной системе пометить страницу как отсутствующую в памяти, не удаляя ее на самом деле. Еще один бит указывает, была ли страница модифицирована за то время, пока она находилась в основной памяти. Как и в случае кэш-памяти, на основании этой информации принимается решение о том, записывать ли страницу снова на диск перед ее удалением из основной памяти (когда нужно освободить место для другой страницы). Остальные управляющие биты действуют в соответствии с различными ограничениями, налагаемыми на доступ к странице. Например, программе могут быть предоставлены полные права на чтение и запись страницы или же только на ее чтение.

Если с диска нужно переместить новую страницу, а память уже заполнена, приходится удалять одну из страниц, уже имеющихся в памяти. Правильный выбор удаляемой страницы так же важен, как правильный выбор заменяемого блока в кэше, и здесь учитывается тот факт, что большую часть времени выполнение программы ограничивается несколькими локализированными областями. Поскольку основная память значительно больше кэш-памяти, в ней можно держать очень большие фрагменты программы. Поэтому частота обмена информацией с диском может быть сравнительно невысокой. К операции замены страниц применимы концепции, подобные концепциям, применяемым в алгоритме LRU, а индикаторами использования страниц могут служить управляю-

щие биты в таблице страниц. В одном из простейших алгоритмов замены задействован единственный управляющий бит, устанавливаемый при обращении к соответствующей странице в 1. Время от времени операционная система очищает этот бит во всех записях таблицы страниц, отмечая таким образом все эти страницы как давно не использовавшиеся.

4.5. ВНЕШНИЕ ЗАПОМИНАЮЩИЕ УСТРОЙСТВА

Обсуждавшаяся в предыдущих разделах полупроводниковая память удовлетворяет далеко не всем потребностям компьютера, связанным с хранением данных. Основным ее недостатком является высокая стоимость хранения единицы информации. Большинству компьютерных систем нужны запоминающие устройства очень большой емкости, роль которых выполняют магнитные диски или оптические диски, обычно называемые вторичными или внешними запоминающими устройствами.

Как следует из названия, магнитный диск состоит из одного или нескольких дисков, нанизанных на один шпиндель. Диск имеет тонкое магнитное покрытие, наносимое на обе его стороны. Вся эта конструкция с постоянной скоростью вращается с помощью мотора, так что намагниченные поверхности движутся над головками, выполняющими операции считывания и записи. Головка представляет собой электромагнит, состоящий из сердечника с обмоткой, по которой пропускается ток.

Информация записывается на магнитный слой с помощью электромагнитных импульсов нужной полярности, генерируемых током в обмотке. Головка намагничивает находящуюся непосредственно под ней поверхность диска, изменяя вектор намагниченности ее частиц. Она же используется и для чтения информации.

За функционирование дискового устройства отвечает *дисковый контроллер*, обеспечивающий интерфейс между этим дисковым устройством и шиной, соединяющей его с остальной частью компьютерной системы. Контроллер диска соединяется с процессорной системной шиной либо с шиной расширения, например с PCI. Он содержит множество регистров, содержимое которых может считываться и записываться операционной системой. Таким образом, операционная система взаимодействует с контролером диска точно так же, как с другими интерфейсами ввода-вывода. Для пересылки данных между диском и основной памятью контроллер использует механизм прямого доступа к памяти. Речь идет о пересылке данных в буфер и из буфера, входящего в состав модуля дискового контролера. Операционная система инициирует передачу с

помощью запросов считывания и записи с последующей загрузкой в регистры контроллера необходимой адресной и управляющей информации.

Запоминающие устройства можно создавать и на основе оптической технологии. Используемая в CD-системах оптическая технология основана на применении лазерного луча. Лазерный луч направляется на поверхность вращающегося диска, вдоль дорожек которого располагаются впадины, отражающие сфокусированный луч в направлении фотодетектора, фиксирующего записанные на диске двоичные данные. Лазер излучает когерентный свет, состоящий из синхронизированных волн одинаковой длины. Если объединить два одинаковых луча в одной фазе, получится более яркий луч, а если сдвинуть лучи на полфазы, они погасят друг друга. Но если два таких луча будут направлены на фотодетектор, то в первом случае он зафиксирует яркое пятно, а во втором – темное.

Поскольку информация на компакт-дисках хранится в двоичном формате, они могут использоваться в качестве носителей данных в компьютерных системах. Наибольшей проблемой в этом случае становится обеспечение целостности данных. Так как впадины очень малы, процесс их записи достаточно сложен. В аудио- и видеоприложениях незначительное количество ошибок вполне допустимо, поскольку оно не отражается на воспроизведении звука или изображения. А вот в компьютерных приложениях подобных ошибок следует избегать. Но поскольку этого добиться невозможно, необходимы дополнительные биты для выявления и исправления ошибок. Существуют два типа компакт-дисков: CD-ROM, содержимое которого после записи можно только считывать, как из полупроводниковой ROM, и CD-RW, которые могут многократно перезаписываться пользователем.

Успех технологии CD и возрастающие требования к вместимости дисков привели к разработке новой технологии, получившей название DVD (Digital Versatile Disk – универсальный цифровой диск). Физический размер DVD-диска такой же, как у CD: толщина – 1,2 мм, диаметр – 120 мм. А вот его вместимость благодаря изменениям в конструкции значительно выше, чем у CD. Первый стандарт DVD был разработан в 1996 году консорциумом компаний.

5. ПРЕДСТАВЛЕНИЕ ЧИСЕЛ В ПАМЯТИ ЭВМ

Как отмечалось выше, данные в компьютере представляются в закодированном виде. Числовая информация может храниться в двоичном или в двоично-десятичном формате.

5.1. СИСТЕМЫ СЧИСЛЕНИЯ

Система счисления – это совокупность правил записи чисел. Системы счисления (с/с) подразделяются на *позиционные* и *непозиционные системы счисления.* Как в позиционных, так и в непозиционных с/с используется определенный набор символов – *цифр,* последовательное сочетание которых образует число. В позиционной с/с количество символов в наборе равно *основанию* с/с. Место каждой цифры в числе называется *позицией.* Номер позиции цифры (за вычетом единицы) в числе называется *разрядом,* т. е. нумерация разрядов начинается с нуля. Имеются *сокращенная* и *развернутая* формы записи числа. Сокращенная форма записи представляет собой последовательность цифр, разделенных точкой на целую и дробную части.

Пример.

321 – сокращенная форма записи числа.
$321 = 3*10^2 + 2*10^1 + 1*10^0$ – развернутая форма записи числа.

Для получения количественного эквивалента числа в позиционной системе счисления необходимо сложить произведения количественных значений цифр на степени основания, показатели которых равны номерам разрядов.

Система счисления, применяемая в современной математике, является позиционной десятичной системой. Основание ее равно 10. Запись любых чисел производится с помощью десяти цифр: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Набор цифр для двоичной системы счисления: 0, 1. Основание ее равно 2. Шестнадцатеричная система счисления имеет набор цифр: 0, 1, 2,..., 9, A, B, C, D, E, F. Основание системы равно 16.

5.2. ПЕРЕВОД ЧИСЕЛ В ДЕСЯТИЧНУЮ СИСТЕМУ СЧИСЛЕНИЯ

Перевод чисел в десятичную систему счисления является самым простым: сначала необходимо записать число в развёрнутой форме в исходной системе счисления, затем в десятичную систему счисления пере-

водятся основание системы и цифры исходного числа. После чего выполняются все действия по вычислению полученного выражения в десятичной системе счисления.

Пример. Переведем два числа из двоичной и шестнадцатеричной систем счисления в десятичную систему счисления.

$10{,}101_2 = 1*2^1 + 0*2^0 + 1*2^{-1} + 0*2^{-2} + 1*2^{-3} = 0.5 + 0 + 0{,}125 = 2{,}625_{10}$

$3A{,}A_{16} = 3*16^1 + 10*16^0 + 10*16^{-1} = 48 + 10 + 0{,}125 = 58{,}125_{10}$

5.3. ПЕРЕВОД ЧИСЕЛ ИЗ ОДНОЙ СИСТЕМЫ СЧИСЛЕНИЯ В ДРУГУЮ СИСТЕМУ СЧИСЛЕНИЯ

Перевод *целых чисел* осуществляется следующим образом.

1. Основание новой с/с выразить в исходной с/с и все последующие действия производить в исходной с/с;

2. Последовательно выполнять деление данного числа и получаемых неполных частных на основание новой с/с до тех пор, пока получим неполное частное, меньшее делителя;

3. Полученные остатки, являющиеся цифрами числа в новой с/с, привести в соответствие с алфавитом новой с/с;

4. Составить число в новой с/с, записывая его, начиная с последнего частного.

Пример. Перевести число 542_{10} в шестнадцатеричную систему счисления. Следуя указанным действиям, получим решение задачи.

542	16	
48	33	16
62	32	2
48	1	
14		

Ответ: $542_{10} = 21E_{16}$

Перевод *дробных чисел* осуществляется следующим образом.

1. Основание новой с/с выразить в исходной с/с и все последующие действия производить в исходной с/с.

2. Последовательно умножать данное число и получаемые дробные части произведений на основание новой с/с до тех пор, пока дробная часть произведения не станет равной нулю или не будет достигнута требуемая точность представления числа в новой с/с;

3. Полученные целые части произведений, являющиеся цифрами числа в новой с/с, привести в соответствие с алфавитом новой с/с;

4. Составить дробную часть числа в новой с/с, записывая его, начиная с целой части первого произведения.

Пример: Перевести $0,3_{10}$ в шестнадцатеричную с/с. Следуя указанным действиям, получим решение задачи.

0	3
	16
4	8
	16
12	8
	16
12	8
	16
12	.....

Ответ: $0,3_{10} = 0,4(C)^{16}$

При переводе дробного числа из системы счисления с основанием P в систему счисления с основанием Q количество дробных разрядов x в новой с/с, приходящихся на один дробный разряд исходной с/с определяется соотношением: $x = log_Q P$.

$$\log_2 10 = 3,3; \qquad \log_{16} 10 = 1$$

Для перевода десятичной дроби в другую систему счисления необходимо перевести отдельно целую часть и дробную часть десятичной дроби в новую систему счисления и полученные результаты соединить через десятичную точку.

Пример:

$108,406_{10} = 1101100,011001111_2$.
$108,406_{10} = 7C,678_{16}$.

5.4. ПРЕДСТАВЛЕНИЕ ЦЕЛЫХ ЧИСЕЛ В ПАМЯТИ ЭВМ

Под целое число отводится 1, 2 или 4 байта. При представлении целого числа крайний слева бит старшего байта служит для хранения знака числа; 0 означает, что это положительное число, 1 – отрицательное. Оставшиеся биты представляют собой значение этого числа в двоичной системе счисления, если число положительное или дополнительный код числа, если число отрицательное. Дополнительный код некоторого отрицательного числа представляет собой результат инвертирования (замены

1 на 0 и наоборот) каждого бита двоичного числа, равного модулю исходного отрицательного числа плюс единица.

Пример. Рассмотрим десятичное число -185_{10}. Модуль данного числа в двоичном представлении равен 10111001_2. Сначала нужно дополнить это значение слева нулями, например до слова. Следующее действие – получить *двоичное дополнение.* Для этого все разряды двоичного числа нужно инвертировать и добавить 1:

0000000010111001_2-> 1111111101000110_2.
$1111111101000110_2 + 0000000000000001_2 = 1111111101000111_2$.

Результат преобразования равен 1111111101000111_2. Именно так и представляется число -185_{10} в памяти компьютера в слове.

ЛИТЕРАТУРА

Основная

1. *Таненбаум, Э.* Архитектура компьютера / Э.Таненбаум. – СПб.: ПИТЕР, 2002. – 704 с.
2. *Буза, М.К.* Архитектура компьютеров / М.К. Буза. – Минск: Новое знание, 2006. – 415 с.
3. *Корнеев, В.В.* Современные микропроцессоры / В.В. Корнеев, А.В. Киселев. – 2-е изд. – М.: НОЛИДЖ, 2000. – 320 с.
4. *Воеводин, В.В.*, Вл.В. Воеводин, Параллельные вычисления / В.В. Воеводин, Вл.В. Воеводин. – СПб.: ПИТЕР, 2002. – 608 с.
5. *Хамахер, К.* Организация ЭВМ / К. Хамахер, З. Вранешич, С. Заки. – 5-е изд. – СПб.: ПИТЕР, 2003. – 848 с.
6. *Столингс, У.* Структурная организация и архитектура компьютерных систем / У. Столлингс. – 5-е изд. – М.: Вильямс, 2002. – 896 с.

Дополнительная

1. *Коуги, П.М.* Архитектура конвейерных ЭВМ / П.М.Коуги; пер. с англ. – СПб.: ПИТЕР, 2002. – 360 с.
2. СуперЭВМ. Аппаратная и программная организация / под ред. С. Фернбаха; пер. с англ. – М.: НОЛИДЖ, 2001. – 320 с.

ОГЛАВЛЕНИЕ

Учебное издание

Акинфина Марина Александровна
Бондаренко Светлана Петровна

АРХИТЕКТУРНЫЕ РЕШЕНИЯ СОВРЕМЕННЫХ КОМПЬЮТЕРОВ

Учебно-методическое пособие
для студентов гуманитарного факультета

В авторской редакции

Ответственный за выпуск *С. П. Бондаренко*

Подписано в печать 18.06.2010. Формат 60×84/16. Бумага офсетная.
Гарнитура Таймс. Усл. печ. л. 3,02. Уч.-изд. л. 2,63. Тираж 50 экз. Зак.

Белорусский государственный университет.
ЛИ №02330/0494425 от 08.04.2009.
Пр. Независимости, 4, 220030, Минск.

Отпечатано с оригинала-макета заказчика
на копировально-множительной технике
гуманитарного факультета
Белорусского государственного университета.
Пр. Независимости, 4, 220030, Минск.