Микропроцессорные системы

(конспект лекций)

СОДЕРЖАНИЕ

В любой сфере человеческой деятельности - в науке, технике, производстве - методы и средства вычислительной техники направлены на повышение производительности труда. В связи с этим уровень специалистов в существенной мере определяется их подготовкой в следующих направлениях, связанных с применением средств вычислительной техники:

- автоматизированное управление технологическими процессами, включая автоматизированные контроль и диагностику технических средств;
- использование ЭВМ для автоматизированного проектирования, научных исследований, административно-организационного управления.

Стремительное совершенствование технологии производства интегральных полупроводниковых компонентов, обеспечившее возможность создания высокоэкономичных цифровых устройств обработки и хранения информации, а также появление эффективных средств программирования оказывают все более существенное влияние не только на развитие техники измерений и управления, но и на подход к автоматизации вообще. Первые попытки применения цифровых устройств для автоматизации производственных процессов относятся к началу 60-х гг., когда были разработаны первые управляющие вычислительные машины. В 70-х гг. ЭВМ стала обычным элементом оборудования автоматизированных систем.

Дальнейшее развитие электронной вычислительной техники привело к ее широкому применению в военном деле, как составной части автоматизированных систем управления войсками и вооружением. Что предопределило повышение требований к квалификации современного командира-инженера, которому необходимо знать основы организации и функционирования универсальных и специализированных управляющих электронных вычислительных машин.

Автоматизация производственного процесса

Автомат (греческое automatos - самодействующий) - устройство (машина, аппарат, прибор), позволяющее осуществлять производственный процесс без непосредственного участия человека и лишь под его контролем.

Автоматизация - применение технических средств, математических методов и систем управления, освобождающих человека полностью или частично от непосредственного участия в процессах получения, преобразования, передачи и использования энергии, материалов и информации.

Внедрение автоматизированных систем дает большой положительный эффект: повышает экономическую эффективность использования оборудования и военной техники, обеспечивает более оперативное и обоснованное принятие решения командирами и начальниками, обеспечивает заметное сокращение сроков выполнения определенных работ, улучшает условия труда, позволяет на более высоком качественном уровне решать стоящие задачи.

Первые приборы, созданные человеком, естественно не были автоматизированными. Начало развития приборостроения связано с именем великого ученого М.В.Ломоносова (1711-1765). Он вместе с академиком Г.В.Рихманом впервые в мире построил электрический измерительный прибор со шкалой для доказательства, что "электричество взвешено быть может".

Автоматический регулятор, принцип которого лежит в основе всех современных регуляторов (принцип обратной связи), был разработан и испытан в 1765г. И.И.Ползуновым.

В 1918г. М.А.Бонч-Бруевич изобрел основной элемент вычислительной машины - триггер.

Разработка и активное использование автоматизированных систем, обусловлено, в первую очередь, прогрессом микропроцессорной техники. Первые попытки применения цифровых устройств для автоматизации производственных процессов относятся к началу 60-х гг. 20-го века, когда были разработаны первые управляющие вычислительные машины (ЭВМ второго поколения на транзисторах, первое поколение ЭВМ было построено на электронных лампах - громоздко и ненадежно). Но настоящий прорыв в автоматизации производственных процессов начался с момента создания микропроцессора в 70-х годах (в ЭВМ третьего поколения элементная база - интегральные схемы), основные преимущества которых миниатюрные размеры и высокая надежность.

Микропроцессор - самостоятельное, или входящее в состав ЭВМ устройство, выполненное на одной или нескольких больших интегральных схемах (БИС), осуществляющее обработку информации и управляющее этим процессом. Обычно микропроцессор содержит арифметико-логическое устройство (АЛУ), блок управления и синхронизации, локальную память (сверхоперативное запоминающее устройство), регистры и другие блоки, необходимые для выполнения вычислит, процесса.

Микропроцессор реализует такие функции, как выборку в предписанной программой последовательности, декодирование и управление выполнением команд, а также выполнение операций тестирования и преобразование данных. Таким образом, он организует и отчасти осуществляет заданную в виде программы последовательность действий - процесс, откуда и название - процессор, микропроцессор.

Конечно, микропроцессор работает не сам по себе, а в составе вычислительного устройства, куда также входят запоминающее устройство для хранения программ и данных и органы управления периферийным оборудованием. Все это может быть размещено и обычно размещается на одной плате с печатными электрическими соединениями, образуя так называемый одноплатный микрокомпьютер. Однако наряду с одноплатными, существуют микрокомпьютеры, выполненные на нескольких печатных платах, а также однокристальные микрокомпьютеры в виде большой интегральной схемы, на которой размещены процессор, память и некоторые средства управления периферией.

Многоплатная реализация характеризуется наибольшей гибкостью: можно варьировать и расширять набор контроллеров, наращивать память, подключать вспомогательные процессоры. Однокристальный микрокомпьютер наименее гибок, но обладает такими преимуществами как компактность, надежность, низкая стоимость.

Для автоматизированного контроля технического состояния специальных изделий и их составных частей на различных этапах жизненного цикла: этапах разработки, исследования, производства и эксплуатации в 1976 году была создана и до настоящего времени эксплуатируется агрегатированная испытательная система (АИС) ТАКТ51 (автоматизированный контрольный тестер). Управляющим ядром системы является блок вычислителя цифрового (БВЦ) ТАКТ51.51.000, построенный на базе малой цифровой вычислительной машины. Все остальные блоки по отношению к БВЦ являются периферийными. Они управляются блоком вычислителя цифрового, построены по одному принципу и содержат блоки местного управления и исполнительную часть.

Поколения электронных вычислительных машин

Основные принципы построения электронных вычислительных машин (ЭВМ), притом в весьма законченном виде, были высказаны еще в 1937 г. американским физиком болгарского происхождения Д.В. Атанасовым. Более того, он предпринял первую попытку спроектировать и построить электронный компьютер. Этот компьютер, названный позже "АВС", был практически закончен к 1942 г. Однако ввести его в эксплуатацию по разным причинам так и не удалось.

Первая ЭВМ заработала в 1945 г. Электронный интегратор и вычислитель (ЭНИАК) - так назвали первую ЭВМ ее создатели инженеры Маучли и Эккерт. Строилась машина при Пенсильванском университете (США) в обстановке глубокой секретности, и только после окончания войны в 1946 г. впервые состоялась публичная демонстрация ЭВМ.

В последующий период до 1955 г. происходило становление вычислительной техники. В это время определились основные принципы построения ЭВМ. Затем с периодичностью 5-7 лет происходил переход к ЭВМ принципиально новых типов, использующих более совершенную элементную базу, имеющих новую структуру, расширяющую их возможности и обеспечивающую большие удобства при работе с ними человека. В связи с этим появилось понятие *поколение* ЭВМ.

Для *ЭВМ первого поколения* (40-е - начало 50-х годов) характерны следующие признаки. Строились они на дискретных компонентах с использованием электровакуумных приборов, имели низкую надежность, в них применялись запоминающие устройства (ЗУ) на ультразвуковых линиях задержки и электронно-лучевых трубках. Ориентировались машины в основном на решение научно-технических задач, для которых характерны относительно небольшие объемы исходных данных и результатов решения.

В *ЭВМ второго поколения* (середина 50-х - 60-е годы) в качестве элементной базы применялись дискретные компоненты и полупроводниковые приборы (транзисторы и диоды), монтаж осуществлялся с использованием печатных плат, ЗУ выполнялись на тороидальных ферритовых сердечниках. Все это повысило быстродействие и надежность машин. В ЭВМ второго поколения обеспечивалась возможность обмена данными между ЭВМ и большим числом внешних устройств. ЭВМ стали успешно применяться и для решения экономических задач.

В *ЭВМ третьего поколения* (60-е годы) в качестве элементной базы используются интегральные микросхемы. Благодаря этому ЭВМ третьего поколения по сравнению с ЭВМ второго поколения имеют меньшие габаритные размеры и потребляемую мощность, большие быстродействие и надежность, широко применяются в самых разнообразных областях деятельности человека.

В *ЭВМ четвертого поколения* (70-е годы) в качестве элементной базы используются интегральные микросхемы высокой степени интеграции - *большие интегральные схемы* (БИС). С их помощью на одном кристалле можно создать устройства, содержащие тысячи и десятки тысяч транзисторов. Компактность узлов при использовании БИС позволяет строить ЭВМ с большим числом вычислительных устройств - процессоров (так называемые многопроцессорные вычислительные системы).

Трудно указать формальное отличие вычислительной системы (ВС) и ЭВМ. Вычислительную систему обычно отличает специализация, большая связь со средой, широкие вычислительные возможности. Так, очень часто в вычислительной системе используют не один процессор, а несколько, образуя тем самым многопроцессорную (мультипроцессорную) систему. Как правило, ВС используют для управления технологическим процессом в реальном масштабе времени, когда обработка информации должна производиться за время, не превышающее время течения самого процесса. От ВС в этом случае требуется много: большое быстродействие и высокий уровень надежности, чрезвычайная оперативность и "живучесть", т.е. способность выполнять возлагаемые на нее функции даже при выходе из строя каких-то элементов. Современные ЭВМ еще не обеспечивают выполнение этих требований, поэтому приходится создавать специализированные вычислительные системы.

К настоящему времени созданы и развиваются *ЭВМ и вычислительные системы пятого поколения.* Эти ЭВМ обладают высокой производительностью, компактностью и низкой стоимостью (эти характеристики улучшаются в каждом следующем поколении ЭВМ). Основная особенность ЭВМ пятого поколения состоит в их высокой интеллектуальности, обеспечивающей возможность общения человека с ЭВМ на естественном языке, способности ЭВМ к обучениюи т.д. Быстродействие ЭВМ пятого поколения достигает десятков и сотен миллиардов операций в секунду, они обладают памятью в сотни мегабайт и строятся на *сверхбольших* БИС, на кристалле которых размещаются миллионы транзисторов.

Общие сведения о процессорах

Как известно, процессор является основным вычислительным блоком компьютера, в наибольшей степени определяющим его мощь. Процессор является устройством, исполняющим программу - последовательность команд (инструкций), задуманную программистом и оформленную в виде модуля программного кода. Чтобы понять, что делает процессор, рассмотрим его в окружении системных компонентов IBM PC-совместимого компьютера. Этой компьютерной архитектурой, естественно, не ограничивается сфера применения процессоров.

Всем известный IBM PC-совместимый компьютер представляет собой реализацию так называемой фон-неймановской архитектуры вычислительных машин. Эта архитектура была

представлена Джоном фон-Нейманом еще в 1945 году и имеет следующие основные признаки. Машина состоит из *блока управления, арифметико-логического устройства* (АЛУ), *памяти и устройств ввода/вывода.* В ней реализуется *концепция хранимой программы:* программы и данные хранятся в одной и той же памяти. Выполняемые действия определяются блоком управления и АЛУ, которые вместе являются основой центрального процессора. Центральный процессор выбирает и исполняет команды из памяти последовательно, адрес очередной команды задается "счетчиком адреса" в блоке управления. Этот принцип исполнения называется последовательной передачей управления. Данные, с которыми работает программа, могут включать переменные - именованные области памяти, в которых сохраняются значения с целью дальнейшего использования в программе. Фон-неймановская архитектура - не единственный вариант построения ЭВМ, есть и другие, которые не соответствуют указанным принципам (например, потоковые машины). Однако подавляющее большинство современных компьютеров основано именно на этих принципах, включая и сложные многопроцессорные комплексы, которые можно рассматривать как объединение фон-неймановских машин. Конечно же, за более чем полувековую историю ЭВМ классическая архитектура прошла длинный путь развития.

В общем смысле под *архитектурой* процессора понимается его программная модель, то есть программно-видимые свойства. Под *микроархитектурой* понимается внутренняя реализация этой программной модели. Для одной и той же архитектуры разными фирмами и в разных поколениях применяются существенно различные микроархитектурные реализации, при этом, естественно, стремятся к максимальному повышению производительности (скорости исполнения программ).

Сейчас существует множество архитектур процессоров, которые делятся на две глобальные категории - RISC и CISC.

RISC - Reduced (Restricted) Instruction Set Computer - процессоры (компьютеры) с сокращенной системой команд. Эти процессоры обычно имеют набор однородных регистров универсального назначения, причем их число может быть большим. Система команд отличается относительной простотой, коды инструкций имеют четкую структуру, как правило, с фиксированной длиной. В результате аппаратная реализация такой архитектуры позволяет с небольшими затратами декодировать и выполнять эти инструкции за минимальное (в пределе 1) число тактов синхронизации. Определенные преимущества дает и унификация регистров.

CISC - Complete Instruction Set Computer - процессоры (компьютеры) с полным набором инструкций, к которым относится и семейство x86. Состав и назначение их регистров существенно неоднородны, широкий набор команд усложняет декодирование инструкций, на что расходуются аппаратные ресурсы. Возрастает число тактов, необходимое для выполнения инструкций.

Процессоры x86 имеют самую сложную в мире систему команд. Хорошо ли это, вопрос спорный, но груз совместимости с программным обеспечением для IBM PC, имеющим уже 20-летнюю историю, не позволяет расставаться с этим "наследием тяжелого прошлого". В процессорах семейства x86, начиная с 486, применяется комбинированная архитектура - CISC-процессор имеет RISC-ядро.

Различают следующие способы организации вычислительного процесса:

- один поток команд - один поток данных (Simple Instruction - Simple Data, SISD) - характерно для традиционной фон-неймановской архитектуры (иногда вместо Simple пишут Single);

- один поток команд - множественный поток данных (Simple Instruction - Multiple Data, SIMD) - технология MMX;

- множественный поток команд - один поток данных (Multiple Instruction - Simple Data, MISD);

- множественный поток команд - множественный поток данных (Multiple Instruction - Multiple Data, MIMD).

Рассмотрим порядок исполнения инструкций обработки данных - выполнения арифметических или логических функций. Во многих случаях инструкция работает с парой операндов - операндом назначения dest (destination) и операндом-источником src (source). Традиционная схема действия инструкции: dest = F (dest, src), где F - некоторая функция от двух переменных. Это означает, что при выполнении инструкции процессор извлекает из указанных в инструкции мест (регистр, память, константа в самой инструкции) пару двоичных чисел, и результат действия над ними записывает на место одного из них (dest). Для выполнения той же функции над следующей парой чисел требуется повторное исполнение инструкции, но уже с другой парой операндов.

Такой принцип исполнения естественен для базовой архитектуры процессоров x86. В процессоры Pentium, "под занавес" их развития, было введено расширение MMX, направленное на ускорение обработки потоков и массивов данных. Ключевым в этом расширении стал принцип SIMD. Здесь вводятся новые упакованные форматы данных: в один регистр MMX можно помещать не только один операнд (64-битное число), но и пару 32-битных, четверку 16-битных или восьмерку 8-битных чисел. Одна инструкция MMX выполняет однотипные действия сразу над всеми числами, упакованными в регистры MMX, заданные операндной частью данной инструкции. Поначалу набор инструкций MMX ограничивался целочисленной арифметикой и логикой, и он стал стандартом для всех современных процессоров x86. Позже появились расширения 3DNow! (от AMD) и SSE (от Intel) для чисел в формате с плавающей точкой, сильно различающиеся по набору инструкций.

Несколько слов о числах с плавающей точкой. Архитектура процессора 8086 позволяет выполнять арифметические функции (сложение, вычитание, умножение и деление) над целочисленными данными (знаковыми и беззнаковыми, двоичными и двоично-десятичными) разрядностью 8 или 16 бит. В процессорах 386+ можно обрабатывать и 32-разрядные числа. Для работы с числами в формате с плавающей точкой (представленными в виде мантиссы и порядка) предусмотрен математический сопроцессор.

Сопроцессор представляет собой набор 80-битных регистров и специализированное арифметическое устройство, которое кроме четырех арифметических действий способно вычислять значение квадратного корня, тригонометрических функций, логарифмов и степеней чисел. Сопроцессор может только перехватывать адресованные ему инструкции из потока команд, выполняемых центральным процессором. Все манипуляции с памятью выполняет центральный процессор. Сложные функции сопроцессора требуют довольно больших затрат времени, но во время их выполнения центральный процессор может продолжать выполнение инструкций, вплоть до момента появления следующей инструкции, адресованной сопроцессору. Однако эта эпизодическая параллельность вычислений не противоречит принципу последовательной передачи управления (самостоятельно сопроцессор передать управление не способен). При отсутствии сопроцессора его функции можно выполнять программно целочисленными средствами центрального процессора, но сопроцессор их выполняет в сотни и тысячи раз быстрее. Программная эмуляция сопроцессора может включаться прозрачно для прикладных программ, обращающихся к сопроцессору. Для этого используется механизм исключений.

История развития микропроцессоров

Термин "микропроцессор" был впервые употреблен в 1972 г., хотя годом рождения этого прибора следует считать 1971 г., когда фирма Intel выпустила микропроцессор серии 4004 - "интегральное микропрограммируемое вычислительное устройство", представляющее собой однокристальный центральный процессор, имеющий в своем составе 4-разрядный параллельный сумматор, 16 4-х - разрядных регистров, накапливающий сумматор и стек. Микропроцессор 4004 был реализован на 2300 транзисторах и мог выполнять 45 различных команд. Последующие поколения микропроцессоров, представляющие собой 8-, 16- и 32-разрядные устройства, появились соответственно в 1972, 1974 и 1981гг.

При оценке параметров микропроцессора и выборе микропроцессорной серии наибольшую роль играет разрядность прибора, которая задает элементарный объем обрабатываемых данных. Чем больше разрядность, тем выше производительность и шире возможности адресации. В ранних приборах разрядность регистров, шин управления, а также информационных шин почти всегда была одинаковой. Сейчас такая структура встречается редко. Например, микропроцессор Motorola 6800 имеет 32-разрядную внутреннюю архитектуру, 16-разрядную шину данных и 24-разрядную адресную шину (адресует до 16 Мбайт оперативной памяти). Для удобства такую архитектуру называют 32/16/24.

В настоящее время стремятся к большей разрядности, например делают полную 32-разрядную архитектуру (32/32/32).

Если считать, что выпуск предыдущих микропроцессоров должен прекращаться при появлении кристаллов с более высокой разрядностью, то 4-разрядные производились бы всего 1 год, 8-разрядные - 5 лет, 16-разрядные - 5 лет (табл.1.1).

Таблица 1.1.

Разрядность	Модель	Количество транзисторов, шт	Год выпуска	Торговая марка
4	4004	2200	1971	Intel
8	8008	2300	1972	Intel
8	8080	4800	1973	Intel
8	280	8400	1976	Zilog
8	8048	12400	1977	Intel
16	8086	29000	1978	Intel
16	68000	75000	1980	Motorola
16	80286	130000	1982	Intel
32	NR-9000	450000	1982	Hewlett-Packard

Однако следует оговориться: 4-разрядные микропроцессоры производятся и применяются до настоящего времени.

В начале развития микропроцессоры изготовлялись по p-МОП технологии, затем специалисты стали отдавать предпочтение комплементарной МОП-технологии (КМОП) (технологии, используемые при изготовлении процессоров, будут рассмотрены ниже в подразделе 3.1). Теперь применяется множество разнообразных видов технологии и технологических приемов при изготовлении микропроцессоров: n-МОП технология с обогащением и с обеднением, биполярная технология, технология $И^2Л$ и др. Например, за

первые 20 лет развития микропроцессорной техники в США зарегистрированы 237 технологических нововведений, из них 67 революционных.

Проследим историю развития микропроцессоров на примере семейства процессоров x86.

Первый *16-разрядный* процессор i8086 фирма Intel выпустила в 1978 году. Частота - 5 Мгц, производительность - 0,33 MIPS для инструкций с 16-битными операндами (позже появились процессоры 8 и 10 МГц). Технология 3 мкм, 29 000 транзисторов. Адресуемая память 1 Мбайт. Через год появился i8088 - тот же процессор, но с 8-разрядной шиной данных. С него началась история IBM PC, неразрывно связанная со всем дальнейшим развитием процессоров Intel, Массовое распространение и открытость архитектуры IBM PC привели к лавинообразным темпам появления нового программного обеспечения, разрабатываемого крупными, средними и мелкими фирмами, а также энтузиастами-одиночками. Технический прогресс тогда и сейчас был бы немыслим без развития процессоров, но, с учетом огромного объема уже существующего программного обеспечения для PC, уже тогда возник принцип обратной программной совместимости - старые программы должны работать на новых процессорах. Таким образом, все нововведения в архитектуре последующих процессоров должны были пристраиваться к существующему ядру.

Процессор i80286, знаменующий следующий этап архитектуры, появился только в 1982 году. Он уже имел 134000 транзисторов (технология 1,5 мкм) и адресовал до 16 Мбайт физической памяти. Его принципиальные новшества - защищенный режим и виртуальная память размером до 1 Гбайт - не нашли массового применения; процессор большей частью использовался как очень быстрый 8088.

Рождение *32-разрядных* процессоров (архитектура IA-32 - Intel Architecture 32 bit) ознаменовалось в 1985 году моделью i80386 (275000 транзисторов, 1,5 мкм). Разрядность шины данных (как и внутренних регистров) достигла 32 бит, адресуемая физическая память - 4 Гбайт. Появились новые регистры, новые 32-битные операции, существенно доработан защищенный режим, были введены режим V86 и страничное управление памятью. Процессор нашел широкое применение в PC; на его благодатной почве стал разрастаться "самый большой вирус" - Microsoft Windows с приложениями. С этого времени стала заметна тенденция "положительной обратной связи": на появление нового процессора производители ПО реагируют выпуском новых привлекательных продуктов, последующим версиям которых становится тесно на новом процессоре. Появляется более производительный процессор, но после непродолжительного восторга и его ресурсы быстро признаются недостаточными, затем история повторяется. Этот "замкнутый круг", конечно, естественен, но есть обоснованное подозрение, что большие ресурсы развращают (или, по крайней мере, расслабляют) разработчика ПО, не принуждая его напрягаться в поисках более эффективных способов решения задачи. Примером эффективного программирования можно считать игрушки на Sinclair ZX-Spectrum, которые работают на минимальных ресурсах - 8-разрядном процессоре и 64 (128) Кбайт ОЗУ. С противоположными примерами большинство пользователей PC сталкиваются регулярно, но, имея процессор Celeron 533 и 64 Мбайт ОЗУ, на них не всегда обращают внимание.

История процессора 80386 повторила судьбу 8086/8088: первую модель с 32-разрядной шиной данных (впоследствии названной 386DX) сменил 386SX с 16-разрядной шиной. Он довольно легко вписывался в архитектуру PC AT, ранее базировавшуюся на процессоре 80286.

Процессор Intel486DX появился в 1989 году. Транзисторов - 1,2 млн, технология 1 мкм. От процессора 80386 существенно отличается размещением на кристалле первичного кэша и встроенного математического сопроцессора - FPU (предыдущие процессоры использовали внешние сопроцессоры x87). Кроме того, для повышения производительности в этом CISC-

процессоре (как и в последующих) применено RISC-ядро. Далее появились его разновидности, отличающиеся наличием или отсутствием сопроцессора, применением внутреннего умножения частоты, политикой кэширования и другим. Тогда же Intel занялась энергосбережением, что отразилось и в линии 386 - появился процессор Intel386SL.

В 1993 году появились первые процессоры Pentium с частотой 60 и 66 МГц - 32-разрядные процессоры с 64-разрядной шиной данных. Транзисторов 3,1 млн, технология 0,8 мкм, питание 5 В. От 486 процессор Pentium принципиально отличается суперскалярной архитектурой - способностью за один такт выпускать с конвейеров до двух инструкций (что, конечно, не означает возможности прохождения инструкции через процессор за полтакта). Интерес к процессору со стороны производителей и покупателей PC сдерживался его очень высокой ценой. Кроме того, возник скандал с ошибкой в сопроцессоре. Хотя фирма Intel математически обосновала невысокую вероятность ее проявления (раз в несколько лет), она (фирма, а не ошибка) все-таки пошла на бесплатную замену уже проданных процессоров на новые, исправленные.

Процессоры Pentium с частотой 75, 90 и 100 МГц, появившиеся в 1994 году, представляли второе поколение процессоров Pentium. При почти том же числе транзисторов они выполнялись по технологии 0,6 мкм, что позволило снизить потребляемую мощность. От первого поколения отличались внутренним умножением частоты, поддержкой мультипроцессорных конфигураций и другим типом корпуса. Появились версии (75 МГц в миниатюрном корпусе) для мобильных применений (блокнотных PC). Процессоры Pentium второго поколения стали весьма популярными в PC. В 1995 году были выпущены процессоры на 120 и 133 МГц, выполненные уже по технологии 0,35 мкм (первые процессоры на 120 МГц делались по технологии 0,6 мкм). 1996-й называют годом Pentium - появились процессоры на 150, 166 и 200 МГц, и Pentium стал рядовым процессором в массовых PC.

Параллельно с Pentium развивался и процессор Pentium Pro, который отличался "динамическим исполнением", направленным на увеличение числа параллельно исполняемых инструкций. Кроме того, в его корпусе разместили вторичный кэш, работающий на частоте ядра, - для начала объемом 256 Кбайт. Однако на 16-разрядных приложениях, а также в среде Windows 95 он был ничуть не быстрее Pentium. Процессор содержит 5,5 млн транзисторов ядра и 15,5 млн транзисторов для вторичного кэша объемом 256 Кбайт. Первый процессор с частотой 150 МГц появился в начале 1995 года (технология 0,6 мкм), а уже в конце года были достигнуты частоты 166, 180 и 200 МГц (технология 0,35 мкм), а кэш увеличен до 512 Кбайт.

В начале 1997 года фирма Intel выпустила процессоры Pentium MMX. Технология MMX (MultiMedia Extensions, мультимедийные расширения) предполагает параллельную обработку группы операндов одной инструкцией. Технология MMX призвана ускорить выполнение мультимедийных приложений, в частности операций с изображениями и обработки сигналов. Ее эффективность вызывает споры в среде разработчиков, поскольку выигрыш в самих операциях обработки компенсируется проигрышем на дополнительных операциях упаковки-распаковки. Кроме того, ограниченная разрядность ставит под сомнение применение MMX в декодерах MPEG-2, в которых требуется обработка 80-битных операндов. Кроме MMX, эти процессоры, по сравнению с обычным Pentium, имеют удвоенный объем первичного кэша и некоторые элементы архитектуры, позаимствованные у Pentium Pro, что повышает производительность Pentium MMX на обычных приложениях. Процессоры Pentium MMX имеют 4,5 млн. транзисторов и выполнены по технологии 0,35 мкм. Развитие линейки моделей Pentium MMX сейчас остановилось. Последние достигнутые тактовые частоты - 166, 200 и 233 МГц. Для мобильных применений (блокнотных ПК) процессоры под кодовым названием Tillamook выпускались по технологии 0,25 мкм, тактовая частота достигла 266 МГц при уменьшенной потребляемой мощности.

В мае 1997 года появился процессор Pentium II. Он представляет собой слегка урезанный вариант ядра Pentium Pro с более высокой внутренней тактовой частотой, в которое ввели поддержку MMX. Трудности размещения вторичного кэша и процессорного ядра в корпусе одной микросхемы преодолели нехитрым способом - кристалл с ядром (processor core) и набор кристаллов статической памяти и дополнительных схем, реализующих вторичный кэш, разместили на небольшой печатной плате - картридже. Первые процессоры имели частоту ядра 233, 266 и 300 МГц (технология 0,35 мкм), летом 1998 года была достигнута частота 450 МГц (технология 0,25 мкм), причем внешняя тактовая частота с 66 МГц повысилась до 100 МГц. Вторичный кэш этих процессоров работает на половине частоты ядра.

В 1999 году появились процессоры Pentium III - в них ввели новый блок 128-битных регистров XMM и новые инструкции, названные SSE. Частота ядра подбирается к 1 ГГц, частота системной шины - 100 и 133 МГц. С конструктивами начались "колебания генеральной линии", и теперь снова предпочтение отдается процессорам со штырьковыми выводами (необходимость картриджей отпадает). На базе Pentium II появилось семейство "облегченных" процессоров Celeron, сначала без вторичного кэша, а потом и с интегрированным вторичным кэшем размером 128 Кбайт. Позже процессоры Celeron приобрели и расширение SSE. Для мощных компьютеров имеется семейство процессоров Xeon, которое охватывает и Pentium II, и Pentium III. Для этих процессоров характерен больший объем вторичного кэша, поддержка более чем двух процессорных конфигураций и более крупный картридж. Есть процессоры Pentium II/III и для мобильных применений.

Конечно же, перечисленными моделями не исчерпывается весь мировой ассортимент микропроцессоров. Это только представители семейства процессоров, имеющих обобщенное название x86. Ряд фирм (например, AMD, Cyrix, IBM) выпускает процессоры, совместимые с перечисленными процессорами Intel и имеющие свои характерные особенности. Обычно они слегка отставали от изделий Intel, выпускаемых в то же время. Однако процессор K7 от AMD изменил ситуацию. Ряд фирм (DEC, Motorola, Texas Instruments, IBM) имеет разработки процессоров, существенно отличающиеся от семейства x86; есть другие классы процессоров и у Intel. Среди них присутствуют и гораздо более мощные процессоры, относящиеся как к RISC-, так и к CISC-архитектуре.

Поколения процессоров

В настоящее время семейство x86 насчитывает 6 поколений процессоров у Intel и 7 - у AMD.

Первое поколение (процессоры 8086 и 8088 и математический сопроцессор 8087) задало архитектурную основу - набор неравноправных 16-разрядных регистров, сегментную систему адресации памяти в пределах 1 Мбайт с большим разнообразием режимов, систему команд, систему прерываний и некоторые другие черты. В процессорах применялась "малая" конвейеризация - пока одни узлы выполняли текущую инструкцию, блок предварительной выборки выбирал из памяти следующую. На выполнение каждой инструкции уходило в среднем по 12 тактов процессорного ядра.

Второе поколение (80286 с сопроцессором 80287) привнесло в семейство защищенный режим, позволяющий использовать виртуальную память размером до 1 Гбайт для каждой задачи, пользуясь адресуемой физической памятью в пределах 16 Мбайт. Защищенный режим является основой для построения многозадачных операционных систем (ОС), в которых система привилегий жестко регламентирует взаимоотношения задач с памятью, ОС и друг с другом. Защищенный режим 80286 не нашел массового применения - эти процессоры, в основном, использовались как "очень" быстрые 8086. Их производительность повысилась не только за счет роста тактовой частоты, но и за счет значительного усовершенствования

конвейера. Здесь на выполнение инструкции уходило в среднем по 4,5 такта. Во втором поколении появились новые инструкции: системные (для обслуживания механизмов защищенного режима) и несколько прикладных (в том числе для блочного ввода/вывода). Наличие защищенного режима не отменяет возможности работы в реальном режиме 8086, и эта возможность сохраняется во всех последующих поколениях (дань совместимости с программным обеспечением, включая и MS DOS).

Третье поколение (386/387 с суффиксами DX и SX, определяющими разрядность внешней шины) ознаменовалось переходом к 32-разрядной архитектуре IA-32. Кроме расширения диапазона непосредственно представляемых величин (16 бит отображают целые числа в диапазоне 0-65535 или от -32767 до +32767, 32 бита - более чем 4 миллиарда) увеличился и объем адресуемой памяти (до 4 Гбайт реальной, 64 Тбайт виртуальной). Для этого почти все программно-доступные регистры были расширены и получили в названии приставку "E" (EAX, EBX...). В систему команд ввели возможность переключения разрядности адресации и данных. Защищенный режим был несколько усовершенствован, но оставлена и обратная совместимость с 286. На таком процессоре стала "расцветать" система MS Windows - сначала оболочка, а потом и операционная система. В плане организации исполнения инструкций существенных изменений, повлекших за собой сокращение числа тактов на инструкцию, не произошло - те же средние 4,5 такта, но частота уже достигла 40 МГц.

Четвертое поколение (486, опять-таки DX и SX) в видимую архитектурную модель больших изменений не внесло, но зато принят ряд мер для повышения производительности. В этих процессорах значительно усложнен исполнительный конвейер - основные операции выполняет RISC-ядро, "задания" для которого готовят из входных CISC-инструкций x86. Этот конвейер стал способным выполнять инструкцию в среднем за два такта. Конечно, каждая инструкция проходит через весь конвейер процессора за гораздо большее количество тактов, но темп выполнения в потоке именно таков. Производительность конвейера процессора оторвалась от возможностей доставки инструкций и данных из оперативной памяти, и прямо в процессор ввели быстродействующий первичный кэш объемом 8-16 Кбайт. В этом же поколении отказались от внешнего сопроцессора: теперь он размещается либо на одном кристалле с центральным (называется FPU), либо его нет вообще. По сравнению с предыдущим поколением и сопроцессор стал работать значительно эффективнее. А тактовая частота в этом поколении достигла 133 МГц (у AMD, а у Intel - только 100).

Пятое поколение - процессор Pentium у Intel и K5 у AMD - привнесли *суперскалярную архитектуру*. Суперскалярность означает наличие более одного конвейера. У процессоров пятого поколения после блоков предварительной выборки и первой стадии декодирования инструкций имеется два конвейера, U-конвейер и V-конвейер. Каждый из этих конвейеров имеет ступени окончательного декодирования, исполнения инструкций и буфер записи результатов. U-конвейер "умеет" все, у V-конвейера возможности немного скромнее. Конвейеризирован и блок FPU. Процессор с такой архитектурой может одновременно "выпускать" до двух выполненных инструкций, но в среднем получается 1 такт на инструкцию. Не все инструкции могут выполняться парно, эффективность использования конвейеров (коэффициент их загрузки или простоя) зависит от программного кода - есть широкие возможности оптимизации. В процессорах применяется блок *предсказания ветвлений* (инструкций программы, выполняемых после очередного условного перехода или вызова), в обязанности которого входит не оставлять конвейеры без работы "на поворотах" алгоритмов. Для быстрого снабжения конвейеров инструкциями и данными из памяти шина данных процессоров имеет разрядность 64 бит, из-за чего поначалу их даже ошибочно называли 64-разрядными процессорами. На закате этого поколения появилось расширение MMX, новизна которого заключается в принципе SIMD: одна инструкция выполняет действия сразу над несколькими (2, 4 или 8) комплектами операндов. В MMX появился и новый тип арифметики - с насыщением (saturated): если результат операции не умещается в разрядной сетке, то вместо

переполнения (антипереполнения) устанавливается максимально (минимально) возможное значение числа.

Шестое поколение процессоров Intel началось с Pentium Pro и продолжается по сей день в процессорах Pentium II, Pentium III, Celeron и Xeon. Его лейтмотивом является *динамическое исполнение,* под которым понимается исполнение инструкций не в том порядке (out of order), как это предполагается программным кодом, а в том, как "удобно" процессору. Инструкции, поступающие на конвейер, разбиваются на простейшие микрооперации, которые далее выполняются суперскалярным процессорным ядром в порядке, удобном процессору. Ядро процессора содержит несколько конвейеров, к которым подключаются исполнительные устройства целочисленных вычислений, обращений к памяти, предсказания переходов и вычислений с плавающей точкой. Несколько различных исполнительных устройств могут объединяться на одном конвейере.

Результаты "беспорядочно" выполняемых микроопераций собираются в переупорядочивающем буфере и в корректном порядке записываются в память (и порты ввода/вывода). Чтобы можно было одновременно выполнять разные инструкции с одними и теми же программно-адресуемыми регистрами, внутри процессора выполняется аппаратное переименование регистров (их у процессора больше, чем доступных по программной модели). Конечно, при этом учитывается и связь по данным, которая сковывает "беспорядочные" параллельные исполнения, даже пользуясь дополнительными регистрами. В процессорах 6-го поколения реализовано *исполнение по предположению*: процессор пытается исполнить инструкцию, последующую (по его мнению) за переходом еще до самого перехода. В итоге всех этих ухищрений среднее число тактов на инструкцию у Pentium Pro сократилось до 0,5 такта. В систему команд были введены новые инструкции, позволяющие писать более эффективные коды (с точки зрения минимизации ветвлений).

Полтакта на инструкцию - звучит, конечно, странно. Но если вспомнить о 8-байтной шине данных, позволяющей за один такт загрузить "кусок" кода, содержащего несколько команд, и о нескольких исполнительных устройствах, одновременно приступающих к их выполнению, то вопросы рассеиваются. Правда, вопрос доставки инструкций и данных из памяти к ядру процессора становится острым, и один первичный кэш здесь не спасает. В то время как частоты ядра процессора (и первичного кэша) неуклонно растут по мере усовершенствования технологий изготовления микросхем (чем тоньше, тем быстрее), частота системной шины, по которой процессор обменивается данными с памятью, так быстро расти не может. Здесь уже сильно сказываются паразитные параметры проводников и разъемов, которые остаются относительно большими по размерам. Кроме того, и сама оперативная память не такая уж и быстрая.

Проблему доставки "сырья" для работы процессоров 6-го поколения фирма Intel стала решать, используя так называемую *двойную независимую шину* (DIB). Одна из шин процессора, "фасадная" (FSB - Front Side Bus), связывает его с системной платой, на которой находится и оперативная память. Другая шина связывает процессор со вторичным кэшем, который находится в одной упаковке с процессором (для пользователя вторичный кэш неотделим от процессора). Частота FSB долгое время оставалась в пределах 66 МГц, что обеспечивало пиковую пропускную способность 528 Мбайт/с. Лишь совсем недавно эта частота поднялась до 100 и даже 133 МГц. А вот тактовая частота второй шины пропорциональна частоте ядра - либо полная частота, либо ее половина. Пиковую пропускную способность этой шины можно оценить, умножив ее тактовую частоту на 8 - число байт данных на шине (у новых процессоров Pentium III разрядность этой шины уже 32 байта). Наличие двойной независимой шины у Intel является одним из атрибутов шестого поколения. Системная шина при этом имеет протокол, принципиально отличающийся от протокола шины процессоров Pentium.

Фирма AMD в своих процессорах шестого поколения (K6) реализовала "беспорядочное исполнение", но двойную независимую шину применять не стала. Вместо этого была увеличена тактовая частота той же шины, которой пользовался Pentium - весьма эффективной в однопроцессорных конфигурациях. Двойная шина появилась лишь в процессорах K6-III. Благодаря такому решению сокет-7 (Super7) пережил целых два поколения процессоров. По микроархитектуре (способу реализации "беспорядочного исполнения") процессоры K6 заметно отличаются от своих Intel'овских собратьев.

Как пятое поколение по ходу развития было "сдобрено" расширением MMX (целочисленное), так шестое поколение получило расширение 3DNow! (AMD) и SSE (Intel). Однако в отличие от единого MMX, эти два расширения не эквивалентны. У них общая идея "потоковой" направленности и реализации SIMD для чисел с плавающей точкой. Поток в данном контексте подразумевает, что с его данными должны выполняться однотипные операции. Кроме того, данные, уже прошедшие обработку, в дальнейшем этим вычислительным процессом использоваться не будут и ими не следует засорять кэш. Теперь появились инструкции загрузки данных в кэш, а также записи в память, минуя кэш. Прежде такого явного управления кэшированием не было.

Седьмое поколение (по AMD) началось с процессора Athlon. Причисление его к новому поколению мотивировано развитием суперскалярности и суперконвейерности, которая теперь охватила и блок FPU (в прежних поколениях FPU если и конвейеризировали, то не распараллеливали).

Завершает линию процессоров IA-32 от фирмы Intel процессор Willamette (в начале 2000 года демонстрировался опытный образец с частотой ядра 1,5 ГГц). Его микроархитектура существенно отличается от привычной архитектуры P6. Конвейер этого процессора имеет 20 ступеней, в то время как у Pentium III 12-ступенчатый целочисленный конвейер и 17-ступенчатый FPU. Длинный конвейер упрощает микрооперации каждой стадии, что позволяет повышать тактовую частоту. Однако при этом растет задержка прохождения инструкции, и, что особенно критично, растут потери времени при ошибках в предсказании ветвлений. Чтобы минимизировать вероятность этих ошибок, в процессоре существенно улучшены узлы, отвечающие за загрузку конвейеров, - блок предсказания переходов, буферы микроинструкций. Первичный кэш имеет объем 256 Кбайт, и в кэше применяется упорядочивание инструкций (чтобы инструкция, следующая за ветвлением, всегда оказывалась в кэше). Существенно повышена производительность исполнительных блоков целочисленных инструкций, но у стандартного FPU (не SIMD) производительность практически та же, что и у Pentium III (в пересчете на эквивалентную тактовую частоту). Для чисел с плавающей точкой основной упор сделан на инструкции SIMD. В процессоре появился набор инструкций SSE2: 76 новых инструкций обработки данных и управления кэшированием. Новые инструкции обработки работают с числами разных форматов, включая учетверенные слова (64 бит) и числа двойной точности с плавающей точкой (64 бит). Процессор имеет совершенно новую шину с тактовой частотой 100 МГц, но передающую до четырех 64-битных пакетов за такт (Quad Pumped) - производительность до 3,2 Гбайт/с. Эта шина является переходной к шине процессоров IA-64. Процессор устанавливается в Socket-462, естественно, не совместимый ни с каким из существующих сегодня сокетов или слотов. В 2001 году ожидается мобильный вариант Willamette - Northwood, а также серверный вариант - Foster.

Фирма Intel сейчас занимается 64-разрядной архитектурой - такая разрядность позволит считать целые числа с числом разрядов почти до $2 \square 10^{19}$. Первый представитель *64-разрядных процессоров* - Itanium, разрабатываемый под кодовым названием Merced. Его архитектура - IA-64 - обеспечивает совместимость с существующим программным обеспечением для используемой ныне архитектуры IA-32.

Микропроцессор Itanium использует 10-уровневый конвейер и может выполнить до шести инструкций за один такт. В новой архитектуре предусмотрено 128 регистров для вычислений с плавающей запятой и столько же для целых чисел, 64 регистра для предсказания переходов и 8 регистров ветвления. На кристалле расположены два блока вычислений с плавающей запятой, обеспечивающие производительность до 6 Гфлоп при операциях с одинарной точностью и до 3 Гфлоп - с повышенной точностью на частоте 1ГГц. Они существенно ускоряют и обработку графической 3D-информации. Вся сверхоперативная память разделена на три уровня, два из которых интегрированы на самом кристалле. Кэш-память третьего уровня, выполненная на дискретных микросхемах SRAM общим объемом до 4 Мб, располагается в картридже микропроцессора.

В начале 2000 года фирма Transmeta заявила процессор Crusoe, который является аппаратно-программным комплексом. Этот комплекс работает нетрадиционным способом: инструкции x86 транслируются в длинные слова VLIW (Very Long Instruction Word) регулярной структуры длиной 64 или 128 бит, которые исполняются процессорным ядром. При этом оттранслированные инструкции хранятся в кэш-памяти и при многократном исполнении транслируются лишь единожды. Ядро процессора исполняет элементы кода в строгом порядке. С этим процессором уже могут работать ОС Windows 9x/NT/2000, Linux. Плавающее энергопотребление составляет от 10-20 мВт до 1-3 Вт, в зависимости от выполняемой работы. Процессор имеет наилучшее отношение производительности к потреблению энергии и предназначается для мобильных систем.

Семейство x86 фирмы Intel началось с 16-разрядного процессора 8086. Все следующие модели процессоров, в том числе 32-разрядные (386, 486, Pentium, Pentium Pro, Pentium II, Celeron) и с 64-разрядным расширением MMX, включают в себя систему команд и программную модель предыдущих, обеспечивая совместимость с ранее написанным программным обеспечением.

ОСНОВЫ ПОСТРОЕНИЯ ЭВМ

Структура электронных вычислительных машин

Все современные вычислительные машины построены по принципам и имеют структуру, предложенную еще в 40–х годах академиком Джоном Фон Нейманом.

Принципы Фон Неймана:

- вычислительная машина конструктивно делится на ряд устройств: процессор, запоминающее устройство (для хранения программ и данных), устройство ввода–вывода и т.д.;
- наличие хранимой в памяти программы;
- одинаковое представление чисел и команд в виде двоичных кодов;
- принцип микропрограммного управления процессом вычислений;
- естественный порядок выборки команд (команды выполняются последовательно, так как они хранятся в памяти; изменение порядка выполнения команд, при необходимости, осуществляется специальными командами перехода).

Согласно первому принципу ЭВМ состоит из ряда устройств, взаимодействующих друг с другом в процессе решения задачи. Рассмотрим кратко основные устройства и их функции (рис. 2.1).

Рис.2.1. Структурная схема ЭВМ

Арифметико–логическое устройство (АЛУ) предназначено для выполнения предусмотренных в ЭВМ арифметических и логических операций. Участвующие в операциях данные выбираются из ОЗУ, результаты операций отсылаются в ОЗУ. Для ускорения выборки операндов (данных, участвующих в операциях) АЛУ может снабжаться собственной местной памятью (сверхоперативным запоминающим устройством – СОЗУ) на небольшое число данных (в сравнении с ОЗУ), но обладающей быстродействием, превышающим быстродействие ОЗУ. При этом результаты операций, если они участвуют в последующих операциях, могут не отсылаться в ОЗУ, а храниться в СОЗУ. Оперативная память вместе с СОЗУ представляет собой единый массив памяти, непосредственно доступный процессору для записи и чтения данных, а также считывания программного кода. К настоящему времени для оптимизации работы созданы процессоры с несколькими уровнями (от одного до трех) кэширования ОЗУ (несколькими СОЗУ).

Устройство управления (УУ) – координирует работу процессора, посылая в определенной временной последовательности управляющие сигналы в устройства ЭВМ, обеспечивая их соответствующее функционирование и взаимодействие друг с другом.

Оперативная память (ОЗУ) – реализуется, как правило, на модулях (микросхемах) динамической памяти. ОЗУ служит для хранения программы, исходных данных задачи, промежуточных и конечных результатов решения задачи.

Память ЭВМ к настоящему времени приобрела довольно сложную структуру и "расползлась" по многим компонентам. Кроме оперативной, память включает также и постоянную (ПЗУ), из которой можно только считывать команды и данные, и некоторые виды специальной памяти (например видеопамять графического адаптера). Вся эта память вместе с оперативной располагается в едином пространстве с линейной адресацией. В любом компьютере обязательно есть *постоянная память,* в которой хранится программа начального запуска компьютера и минимальный необходимый набор сервисов (например: ROM BIOS).

Все узлы ЭВМ не входящие в ядро называются периферийными. *Они* обеспечивают расширение возможностей ЭВМ, облегчают пользование ими. В состав периферийных (внешних) устройств могут входить следующие узлы.

Внешняя память (устройства хранения данных, например, дисковые) – память, имеющая относительно невысокое быстродействие, но по сравнению с ОЗУ существенно более высокую

емкость. Внешняя память предназначена для записи данных с целью последующего считывания (возможно, и на другом компьютере). От рассмотренной выше памяти, называемой также внутренней, устройства хранения отличаются тем, что процессор не имеет непосредственного доступа к данным по линейному адресу. Доступ к данным на устройствах хранения выполняется с помощью специальных программ, обращающихся к контроллерам этих устройств. В силу того что быстродействие внешней памяти значительно ниже быстродействия АЛУ, последнее в процессе работы взаимодействует лишь с ОЗУ, получая из него команды и данные, отсылая в эту память результаты операций. Часто при решении сложных задач емкость ОЗУ оказывается недостаточной. В этих случаях в процессе решения задач данные определенными порциями могут пересылаться из внешней памяти в ОЗУ, откуда они затем выбираются для обработки в АЛУ.

Устройства ввода/вывода (УВВ) служат для преобразования информации из внутреннего представления в компьютере (биты и байты) в форму, доступную окружающим, и обратно. Под окружающими понимаем как людей, так и другие машины (например технологическое оборудование, которым управляет компьютер). К устройствам ввода относятся клавиатура, мышь, джойстик, микрофон, сканер, видеокамера, различные датчики; к устройствам вывода – дисплей, принтер, плоттер, акустические системы (наушники), исполнительные механизмы. Список устройств ввода/вывода безграничен – благодаря фантазии и техническому прогрессу в него входят все новые и новые устройства; так, например, шлем виртуальной реальности из области фантастики вышел в производственно–коммерческую. Устройства хранения к УВВ относить некорректно, поскольку здесь преобразования информации ради доступности внешнему миру не происходит. Устройства хранения вместе с УВВ можно объединить общим понятием *периферийные устройства.* Существует еще большой класс *коммуникационных устройств,* предназначенных для передачи информации между компьютерами и (или) их частями. Эти устройства обеспечивают, например, соединение компьютеров в локальные сети или подключение терминала (это УВВ) к компьютеру через пару модемов. Периферийные и коммуникационные устройства снабжаются контроллерами или адаптерами, которые доступны процессору.

Общие сведения о специализированном вычислителе БВЦ ТАКТ51

Ядром автоматизированного средства контроля является специализированный вычислитель – блок вычислителя цифрового (БВЦ ТАКТ51.51.000). Блок вычислителя цифрового предназначен для управления системой ТАКТ51, а также для обработки информации при проверке изделия и самопроверке работоспособности ТАКТ51. БВЦ относится к классу малых одноадресных управляющих специализированных цифровых вычислительных машин.

БВЦ реализует следующие функции:

- ввод программы с 8–дорожечной перфоленты в оперативное запоминающее устройство и хранение программы в ОЗУ;
- обмен данными с периферийными блоками;
- математическую обработку результатов измерений, полученных с периферийных блоков;
- выявление и обработку неисправностей, возникающих в системе;
- взаимодействие оператора с БВЦ путем операций ручного управления и наблюдения посредством ПУ и ПО;
- отсчет текущего времени работы БВЦ.

По существу блок вычислителя цифрового – это электронная вычислительная машина, автоматически выполняющая интерпретацию программы (алгоритма) в виде физических

процессов, назначением которых является реализация арифметических и логических операций над информацией, представленной в цифровой форме.

Для того чтобы любая ЭВМ, в том числе и БВЦ, могла автоматически решать задачи, она должна обеспечивать выполнение следующих функций:

- восприятие вводимой в машину информации – исходных данных и программы решения задач;
- хранение введенной информации и выдачу ее в требуемые моменты времени, обусловленные программой;
- выполнение арифметических и логических операций;
- выдачу по программе результатов вычислений в удобной для восприятия форме;
- автоматическое управление вычислительным процессом в соответствии с введенной программой.

Для выполнения перечисленных функций в состав БВЦ входят: устройство ввода, запоминающее устройство (память), процессор, устройства вывода (являются периферийными по отношению к БВЦ и, по существу, не входят непосредственно в состав БВЦ).

Структура БВЦ изображена на рис.2.2.

Рис.2.2. Структура БВЦ ТАКТ51.

Устройство ввода (ПБВД–5) обеспечивает фотоэлектрическое считывание информации с восьмидорожечной перфоленты и представление считанной информации в двоичной форме в виде электрических сигналов, воспринимаемых оперативной памятью.

Запоминающее устройство (платы ОЗУ и ПЗУ) служит для хранения информации, необходимой для производства вычислений. В памяти размещаются программы, задающие порядок вычислений, и данные, представляющие исходные значения, промежуточные и конечные результаты вычислений.

Процессор – это центральное устройство БВЦ. Процессор "воспринимает" программу и на ее основе управляет работой всех устройств БВЦ, инициируя выполнение действий в памяти и устройствах ввода–вывода. Функцией процессора является выборка команд из памяти и их выполнение.

Периферийными устройствами вывода (цифропечатающее устройство ПЦПУ–6, устройство вывода перфоленточное ПУВЛ–1) обеспечивается вывод информации из памяти для ее последующего использования (печать цифр и символов, пробивка отверстий на ленте).

Работа БВЦ протекает следующим образом. Программа и исходные данные, представленные на носителе информации (перфоленте), считываются устройством ввода ПБВД–5 и загружаются в память, в адреса, указанные на перфоленте. Выполнение программы сводится к последовательной выборке команд из памяти и их выполнению средствами процессора и устройств ввода–вывода.

Программа проверки изделия или самопроверки системы ТАКТ51 определяет объем и последовательность операций, выполняемых ТАКТ51. БВЦ вводит программу проверки с перфоленты в оперативную память и в соответствии с программой выдает команды на управляющие и измерительные блоки системы, принимает и обрабатывает информацию с измерительных блоков, а результаты проверки выводит на печать на ЦПУ и/или перфорацию на УВЛ.

Команда представляет собой цифровой код, преобразующийся в БВЦ в управляющие сигналы. Для БВЦ определен 13–разрядный двоичный формат команды, показанный на рис.2.3:

0	1	2	3	4	5	6	7	8	9	10	11	12
Код команды			Адресная часть									КР

Рис.2.3. Структура команды БВЦ ТАКТ51.

Команда состоит:

- из кода команды (с 0–го по 2–й разряд), определяющего тип команды;
- из кодированной информации (с 3–го по 11–й разряды), определяющей информацию: действие, предписанное командой, или адрес (номер) ячейки памяти;
- из контрольного разряда (12–й разряд), обеспечивающего контроль правильности команды и ее неискажения при передаче в устройствах БВЦ. Это достигается записью, при необходимости, единицы в контрольном разряде (дополнение до нечетного количества единиц в команде). Все команды программы располагаются в ячейках памяти запоминающего устройства. Для удобства обращения к памяти ячейки пронумерованы последовательно от 0 до 4095_{10}. Адреса (номера) представляются или в двоичном виде (в БВЦ) от $000\ 000\ 000\ 000_2$ до $111\ 111\ 111\ 111_2$, или в восьмеричном виде (при программировании) от 0000_8 до 7777_8.

Формы представления чисел в электронных вычислительных машинах

ЭВМ оперирует с числами, содержащими конечное число разрядов. Количество разрядов ограничено длиной разрядной сетки машины. Под разрядной сеткой понимается совокупность двоичных разрядов, предназначенных для хранения и обработки машинных слов (двоичных кодов).

Для представления чисел в ЭВМ используются две формы: естественная (с фиксированной запятой) и нормальная с (плавающей запятой).

Естественная (с фиксированной запятой) (FIXED – POINT REPRESENTATION) форма представления чисел предполагает, что положение запятой, отделяющей целую часть от дробной, фиксировано в разрядной сетке машины. Для представления знака выделяется

специальный разряд – знаковый. Обычно это крайний левый разряд. Для положительных чисел в знаковом разряде записывается 0, а для отрицательных 1.

Количество двоичных разрядов и положение запятой в разрядной сетке машины определяют такие важные характеристики ЭВМ, как точность и диапазон представляемых чисел. Так, например, для n–разрядной сетки точность (дискретность) равна 2^{-n}, а диапазон $0 \le |N| \le 2^{-1}-1$. Рассмотрим это более подробно.

Обычно в ЭВМ используются два способа расположения фиксированной запятой: перед старшим разрядом или после младшего разряда. В первом случае ЭВМ работает только с числами, меньшими единицы (рис.2.4а), во втором – с целыми (рис.2.4б).

0	1	2	. . .	n–1	n
	2^{-1}	2^{-2}	. . .	$2^{-(n-1)}$	2^{-n}
Знак	Мантисса				

а) запятая перед старшим разрядом

0	1	2	. . .	n–1	n
	2^{n}	2^{n-1}	. . .	2^{1}	2^{0}
Знак	Мантисса				

б) запятая после младшего разряда

Рис.2.4. Разрядная сетка с фиксированной запятой.

Длину разрядной сетки с фиксированной запятой в современных универсальных ЭВМ принято выбирать кратной байту (8 бит или 8 двоичных разрядов). В персональных ЭВМ используется разрядная сетка длиной 8, 16, 32 или 64 разряда. Для специализированных вычислителей возможны другая кратность и длина разрядной сетки.

Одним из важных преимуществ данной формы представления числа является возможность построения сравнительно несложных операционных устройств ЭВМ с высоким быстродействием, а недостатком – малый диапазон представления числа.

Нормальная (с плавающей запятой) (FLOATION – POINT REPRESENTATION) форма представления чисел позволяет значительно увеличить диапазон представления чисел (рис.2.5).

0	1	2	. . .					n–1	n
	2^{m}	...	2^{0}		2^{m}	2^{m-1}	. . .	2^{1}	2^{0}
Знак	Порядок			Знак	Мантисса				

Рис.2.5. Разрядная сетка с плавающей запятой.

Представление числа в форме с плавающей запятой в общем виде определяется выражением $N=\pm M*2^{\pm p}$, где M – мантисса числа, p – порядок, 2^p характеристика числа. Знак числа совпадает со знаком мантиссы. Говорят, что число представлено в нормальной форме. Однако такое представление приводит к неоднозначности, поэтому мантисса M обычно представляется правильной дробью в нормализованном виде (первая цифра справа от запятой в числе должна быть отличной от нуля). Таким образом, значение нормализованной мантиссы должно удовлетворять неравенству $2^{-1} \square |M|<1$. Для кодирования отрицательных чисел в ЭВМ применяют прямой, обратный и дополнительный коды. Обратный код числа получают инвертированием всех разрядов, а дополнительный код получают инвертированием всех разрядов и дальнейшим суммированием единицы с младшим разрядом.

Иногда для кодирования отрицательных чисел в ЭВМ применяют модифицированные прямой, обратный и дополнительный коды. В модифицированных кодах для кодирования числа отводят два разряда, причем знак плюс кодируется сочетанием 00, а знак минус – 11. Два разряда под знак позволяют контролировать переполнение разрядной сетки отведенной под мантиссу числа при выполнении арифметических операций. В остальном модифицированные коды аналогичны обычным.

Числа в специализированном вычислителе БВЦ ТАКТ51 представлены в немодифицированном дополнительном коде с фиксированной запятой рис.2.6.

0	1	2	3	4	5	6	7	8	9	10	11
Знак	Мантисса										

Рис.2.6. Форма представления числа в БВЦ ТАКТ51.

Нулевой разряд отведен для записи знака числа, причем плюс соответствует 0, а минус 1. Запятая считается фиксированной после знакового разряда.

Дробная двоичная мантисса представляется 11–ю разрядами, причем старший разряд первый. Отрицательные числа хранятся в памяти и участвуют в операциях в дополнительном коде. Применение этого кода позволяет свести операцию вычитания к операции сложения, а операции умножения и деления к сериям сложений и сдвигов. В результате для выполнения всех арифметических операций оказывается возможным использовать одно суммирующее устройство, в котором знаковые разряды складываются так же, как и разряды значащих частей чисел, в результате чего знак суммы получается автоматически.

Выполнение арифметических операций над двоичными числами

Основной операцией, которая используется в цифровых устройствах при различных вычислениях, является операция алгебраического сложения чисел (сложения, в котором могут участвовать как положительные, так и отрицательные числа). Вычитание легко сводится к сложению путем изменения на обратный знака вычитаемого. Операции умножения и деления также выполняются с помощью операции сложения и некоторых логических действий. Поэтому именно с операции сложения начнем рассмотрение способов выполнения арифметических операций.

Принципы выполнения арифметических операций в различных процессорах сходны, но отличаются лишь конкретным исполнением. Порядок выполнения ряда арифметических операций рассмотрим на примере работы специализированного вычислителя БВЦ ТАКТ51.

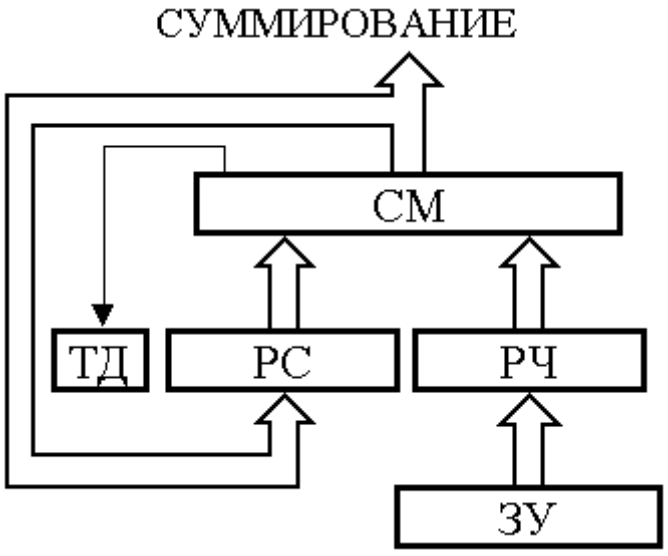
Сложение.

При выполнении операции сложения по команде СЛ число из запоминающего устройства считывается в регистр числа РЧ и посылается на один вход сумматора. На другой вход сумматора передается число из регистра сумматора РС. Сложение двух чисел в двоичной системе, как и в десятичной, начинается с младшего разряда по правилам, которые даются таблицей двоичного сложения:

0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 0 + единица переноса в старший разряд, т.е. $1_2+1_2=10_2$.

Суммирование во всех остальных разрядах выполняется по тем же правилам. Отличие состоит в том, что для получения цифры очередного разряда надо складывать не только цифры данного разряда слагаемых чисел, но и единицу переноса, если она образовалась при сложении в соседнем младшем разряде. Знаки чисел складываются по общему правилу.

Так как в цифровом вычислителе ТАКТ51 используется немодифицированный дополнительный код, то возможное переполнение, с одной стороны, исключается путем тщательного предварительного анализа решаемых задач, с другой стороны, используется триггер дополнительный ТД для получения суммы в модифицированном дополнительном коде. Перенос из нулевого разряда сумматора инвертирует содержимое триггера дополнительного ТД, что позволяет программным способом определить переполнение.



Сдвиги.

Из последующего описания будет ясно, как широко используются операции сдвига вправо (влево) при умножении и делении.

Ввиду особой важности операции сдвига рассмотрим ее подробно.

В цифровых машинах используются три разновидности сдвига, отличающиеся друг от друга следующими правилами:

- если при сдвиге вправо (влево) освобождающиеся старшие (младшие) разряды заполняются нулями, то сдвиг называется логическим;
- если при сдвиге вправо в освобождающиеся старшие разряды записываются значения, совпадающие с теми, которые до сдвига были записаны в старшем знаковом разряде, то сдвиг называется арифметическим или модифицированным. При сдвиге влево младшие разряды заполняются нулями (в случае применения дополнительного кода).
- если при сдвиге вправо освобождающиеся старшие разряды принимают значения выдвигаемых младших разрядов, или если при сдвиге влево освобождающиеся младшие разряды принимают значения выдвигаемых старших разрядов, сдвиг называется циклическим.

В БВЦ используются все три разновидности сдвигов.

При циклическом сдвиге участвуют два регистра:

- триггер дополнительный ТД
- регистр сумматора РС.

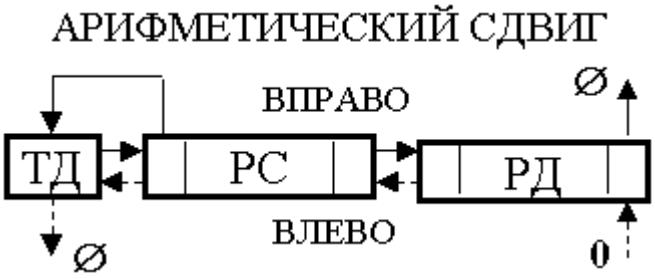
Причем при выполнении циклического сдвига вправо содержимое регистра ТД передается в нулевой разряд регистра РС, содержимое нулевого разряда РС передается в первый разряд, содержимое первого разряда – во второй и т.д. Содержимое 11 разряда РС передается в ТД.

При циклическом сдвиге влево содержимое ТД передается во 11–й разряд РС, 11–й разряд РС – в 10–й и т.д. Содержимое нулевого разряда РС передается в ТД.

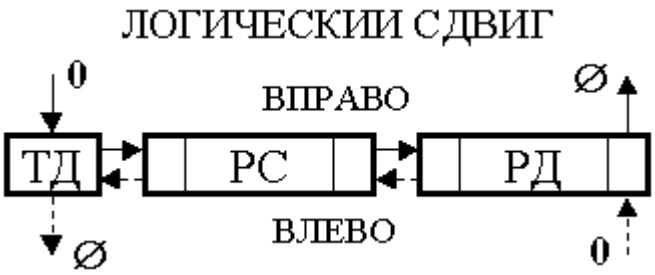
При логическом или арифметическом сдвиге вправо (влево) участвуют три регистра:

- триггер дополнительный ТД;
- регистр сумматора РС;
- регистр дополнительный РД.

Причем при арифметическом сдвиге вправо (перед началом операции) триггеру дополнительному ТД присваивается значение нулевого разряда регистра РС и далее содержимое ТД не изменяется и передается в нулевой разряд РС, нулевой разряд РС – в первый и т.д. Содержимое 11 разряда РС передается в 0 – разряд РД, 0 – разряд РД – в первый и т.д. Содержимое 11–го разряда РД выходит за пределы регистра и теряется.



При логическом сдвиге вправо (перед началом операции) триггеру дополнительному ТД присваивается нулевое значение и далее сдвиг имеет место как описано выше. При сдвиге влево (арифметическом или логическом) в 11–й разряд регистра РД записывается нуль, а его содержимое передается в 10–й разряд РД и т.п. Содержимое 0–го разряда РД передается в 11–й разряд РС, 11–й разряд РС – в 10–й и т.д. Содержимое нулевого разряда РС передается в ТД, а содержимое ТД выходит за пределы регистра и теряется.



В БВЦ предусмотрены цепи сдвига вправо (влево) на один разряд. Сдвиг на несколько разрядов осуществляется путем последовательного повторения сдвигов на один разряд.

Если количество сдвигов задано, то учет выполненного количества сдвигов ведется счетчиком сдвига.

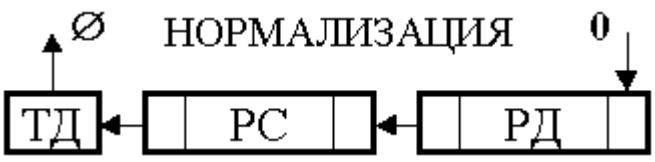
Нормализация.

Операция нормализации – это сдвиг влево до тех пор, пока не выполнится условие окончания нормализации.

Число называется нормализованным, если выполняются следующие условия:

- сочетание "01" в нулевом и первом разрядах для положительного числа;
- сочетание "10" в нулевом и первом разрядах для отрицательного числа;
- если появляется код 6000_8 в регистре РС и нули во всех разрядах РД;
- если присутствуют нули во всех разрядах регистров РС и РД.

Количество сдвигов, которое требуется для того чтобы нормализовать 24–х–разрядное число, фиксируется в счетчике сдвигов.



Умножение.

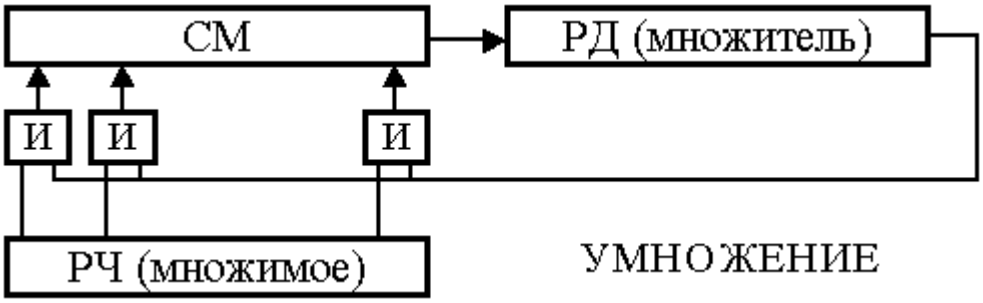
Умножение в вычислительной машине может быть реализовано путем выполнения некоторого циклического процесса, который сводится к последовательности операций

прибавления множимого к сумме уже ранее вычисленных частных произведений и сдвига получаемых сумм.

В БВЦ реализован метод умножения младшими разрядами множителя вперед со сдвигом частных произведений и множителя вправо на один разряд.

В каждом цикле множимое либо передается в сумматор, если младший разряд регистра множителя равен 1, либо не передается, если он равен 0, после чего содержимое сумматора и регистра множителя сдвигается вправо на один разряд. После окончания n–го цикла (где n – количество разрядов множителя) в сумматоре образуется произведение. Реализация умножения указанным методом требует сдвига вправо в регистре множителя и в сумматоре и дополнения сумматора n разрядами для заполнения их младшими разрядами произведения.

Вычисление произведения с полным количеством разрядов может быть выполнено без добавления к сумматору n разрядов. Для этого можно, связав младший разряд сумматора со старшим разрядом регистра множителя, объединить их в общий сдвиговый регистр (рис.2.9). Освобождающиеся при сдвиге старшие разряды регистра множителя заполняются младшими разрядами произведения. После окончания умножения сумматор заполняется старшими разрядами произведения, а весь регистр множителя – младшими его разрядами.



Так как отрицательные числа с фиксированной запятой хранятся в запоминающем устройстве в дополнительном коде, то в вычислителе реализуется способ умножения без преобразования отрицательных сомножителей в прямой код. Знак произведения вырабатывается автоматически в ходе перемножения сомножителей, и произведение получается в коде, соответствующего его знаку.

Рассмотрим примеры умножения с поразрядным анализом младшего разряда множителя, представленного в дополнительном коде.

Если множимое A>0 и множитель B>0, представленные n–разрядными прямыми кодами $[A]_{пр}$ и $_{[Bпр]}$, то положительное произведение [C] получается в прямом коде просто как $[C]_{пр.}=[A]_{пр}\square[B]_{пр}$

Прямой код множимого $[A]_{пр}$ умножается на каждую цифру прямого кода $[B]_{пр}$. Если младший разряд множителя равен 1, то частное произведение, равное множимому, прибавляется к накопленной сумме частных произведений. Полученная новая сумма и множитель сдвигаются на разряд вправо и освобождающие при сдвиге старшие разряды регистра множителя заполняются младшими разрядами произведения. Если младший разряд множителя равен 0, то и очередное частное произведение равно 0. При этом этап суммирования пропускается, а происходит только сдвиг на разряд вправо множителя и суммы частных произведений, накопленной к данному шагу умножения.

Пример 1. Множимое A=7/8 и множитель B=5/8 заданы 4х–разрядными двоичными кодами соответственно $[A]_{пр}$=0.111 и $[B]_{пр}$=0.101. Требуется найти их произведение A□ B = 7/8□5/8 = 35/64 (табл.2.1).

Таблица 2.1.

Выполняемое действие	Множимое	Множитель
Исходное состояние	0.111	0.101
Исходная сумма частных произведений	0.000 + 0.111	0.101
1–я сумма частных произведений	0.111	0.101
Сдвинутая 1–я сумма	0.011	1.010
Сдвинутая 2–я сумма	0.001 + 0.111	1.101
3–я сумма частных произведений	1.000	1.101
Сдвинутая 3–я сумма	0.100	0.110
Окончательное произведение $[AB]пр. = 0.100011_2 = 35/64_{10}$.		

Если множимое A>0 и множитель B<0, то в результате перемножения должно получиться отрицательное произведение в дополнительном коде, однако непосредственное умножение $[A]_{пр}$.□ $[B]_{доп.}$ без специальной коррекции приводит к псевдопроизведению. Для того чтобы получить точное произведение, необходимо алгоритм умножения, если множитель отрицательный, строить в соответствии с формулой:

$$[C]_{доп.}=[A]_{пр}□[B]_{доп}+[–A]_{доп}.$$

Пример 2. Множимое A = 7/8 и множитель B = –5/8 заданы 4х–разрядными двоичными кодами: $[A]_{пр}$=0.111 и $[B]_{доп}$=1.011. Требуется найти их произведение A □ B = 7/8 □ (–5/8) = –35/64. (табл.2.2).

Таблица 2.2.

Выполняемое действие	Множимое	Множитель
Исходная сумма частных произведений	0.111 0.000 + 0.111	0.011
1–я сумма частных произведений	0.111	1.011
Сдвинутая 1–я сумма	0.011 + 0.111	1.101
2–я сумма частных произведений	1.010	1.101
Сдвинутая 2–я сумма	0.101	0.110
Сдвинутая 3–я сумма Коррекция	0.010 + 1.001	1.011
	1.011	1.011
Окончательное произведение $[AB]_{доп.} = 1.011101_2 = 35/64_{10}$. $[AB]_{пр.} = –0.100011_2 = –35/64_{10}$.		

Программное управление вычислительным процессом

Один из принципов построения ЭВМ, предложенных Дж. фон Нейманом – наличие хранимой в памяти программы. Процессор исполняет программный код, находящийся к моменту исполнения в пространстве памяти. Программный код – это последовательность команд, или *инструкций,* каждая из которых определенным образом закодирована и расположена в целом числе смежных байт памяти. Каждая инструкция обязательно имеет *операционную часть,* несущую процессору информацию о требуемых действиях. *Операндная часть,* указывающая процессору, где находится его "предмет труда" – операнды, может присутствовать в явном или неявном виде и даже отсутствовать. Операндная часть может описывать от нуля до двух операндов, участвующих в данной инструкции (есть инструкции, в которые кроме двух операндов задается еще и параметр инструкции). Здесь могут быть сами значения операндов (непосредственные операнды); явные или неявные указания на регистры процессора, в которых находятся операнды; адрес ячейки памяти (или его составная часть); регистры процессора, участвующие в формировании адреса, и разные комбинации этих

компонент. Длина инструкции в семействе x86 может быть от одного до 12 байт и определяется типом инструкции. Исторически сложившийся формат инструкций x86 довольно сложен, и "понять", сколько байт занимает конкретная инструкция, процессор может, лишь декодировав ее первые 1–3 байт. Инструкции могут предшествовать префиксы (к счастью, всегда однобайтные), указывающие на изменение способа адресации, размера операнда или (и) необходимость многократного (по счетчику и условию) повторения данной инструкции. Адрес (логический) текущей исполняемой инструкций хранится в специальном регистре – *указателе инструкций* (Instruction Pointer, IP), который соответствует счетчику команд фон–неймановской машины. После исполнения так называемой *линейной инструкции* этот указатель увеличивает свое значение на ее длину, то есть указывает на начало следующей инструкции. Линейная инструкция не нарушает порядок выполнения, определяемый последовательностью расположения инструкций в памяти (по нарастанию адреса). Кроме линейных инструкций, существуют *инструкции передачи управления,* среди которых различают инструкции *переходов* и *вызовов* процедур. Эти инструкции в явном или неявном виде содержат информацию об адресе следующей выполняемой инструкции, который может указывать на относительно произвольную ячейку памяти. Инструкции переходов и вызовов могут быть *безусловными* (ни от чего не зависящими) и *условными.* Произойдет условный переход (вызов) или нет, зависит от состояния флагов (признаков) на момент исполнения данной инструкции. Если переход (вызов) не состоится, то исполняется инструкция, расположенная в памяти следом за текущей. *Вызов* процедуры характерен тем, что перед ним процессор сохраняет в стеке (стек – это область ОЗУ) адрес следующей инструкции, и на этот адрес передается управление после завершения исполнения процедуры (этот адрес извлекается из стека при выполнении инструкции возврата). При *переходе* в стеке ничего не сохраняется, то есть переход выполняется безвозвратно.

Последовательность исполнения инструкций, предписанная программным кодом, *может* быть нарушена под воздействием внутренних или внешних (относительно процессора) причин. К внутренним причинам относятся *исключения* (exceptions) – особые ситуации, возникающие при выполнении инструкций. Наглядным примером исключений является попытка деления на ноль. При возникновении условия исключения процессор автоматически выполняет вызов процедуры обработки исключения, после которой он может вернуться к повторному исполнению инструкции, породившей исключение или следующей за ней. Вариант поведения зависит от типа произошедшего исключения. Исключения широко используются современными операционными системами, на основе обработки исключений строится система виртуальной памяти и реализуются многие функции многозадачных операционных систем. Внешними причинами изменения нормальной последовательности инструкций являются *аппаратные прерывания* – вызовы процедур под воздействием электрических сигналов, поступающих на специальные выводы процессора. Эти сигналы могут подаваться совершенно неожиданно для исполняемой программы, правда, у программиста есть возможность заставить процессор (компьютер) игнорировать все прерывания или их часть. Злоупотреблять этой возможностью нельзя (да и не всегда она есть), поскольку на аппаратных прерываниях строится, например, отсчет времени и другие системные и прикладные функции компьютера. Источниками аппаратных прерываний являются контроллеры и адаптеры периферийных устройств, генераторы меток времени, системы управления питанием и другие подсистемы.

Есть еще так называемые *программные прерывания,* но они отнюдь не нарушают последовательность инструкций, предписанную программистом. Поэтому прерываниями они по сути не являются – это всего лишь особый способ вызова процедур, широко используемый для вызова системных сервисов BIOS и операционной системы. И наконец, последовательность инструкций может изменяться по сигналу аппаратного сброса или инициализации процессора. С этого, собственно, и начинается функционирование компьютера: процессор переводится в исходное состояние и запускается. При этом указатель инструкций совместно с другими регистрами, участвующими в формировании адреса инструкции, генерирует адрес "начального

пуска". По этому адресу должна располагаться инструкция, с которой начинается инициализация компьютера.

В процессоре предусматривается возможность выполнения большого числа различных операций. Несмотря на то что число таких операций может быть более 100, каждая из них представляет собой простейшие арифметические либо логические действия, такие, например, как сложение, вычитание, умножение и деление чисел, пересылка кодов и т.п. При этом в каждой операции участвует не более двух операндов. В связи с этим решаемая задача должна быть предварительно представлена последовательностью таких операций, которые способна выполнять ЭВМ. Затем на каждую из этих операций должна быть составлена так называемая *команда.* Совокупность команд, образующая *программу* решения задачи, должна быть помещена в ОЗУ.

Команда должна содержать все необходимые для выполнения операции указания: вид операции, место, где хранятся операнды данной операции и куда должен быть помещен результат операции. Такая команда имеет следующий формат:

Коп	A_1	A_2	A_3

Здесь КОп — код операции. В каждой ЭВМ предусматривается определенная система кодирования операций. Например, может быть принят следующий способ записи вида операции: 01 — сложение, 02 — вычитание, 03 — умножение и т.д.; A_1 — первый адрес — адрес ячейки оперативной памяти, в которой хранится первый операнд; A_2 — второй адрес — адрес второго операнда; A_3 — третий адрес — адрес ячейки оперативной памяти, в которую должен помещаться результат операции. Команда с таким содержанием называется трехадресной.

ЭВМ могут использовать двухадресные команды, имеющие следующий формат:

Коп	A_1	A_2

Результат операции в этом случае помещается в ячейку одного из операндов либо остается в АЛУ.

Одноадресные команды имеют формат:

Коп	A_1

При таком формате для выполнения одного арифметического действия над двумя числами от машины может потребоваться исполнение нескольких команд. Например, для сложения двух чисел необходимо выполнить три команды:

- ввести в АЛУ число, хранящееся в оперативной памяти (ОЗУ) по приведенному в команде адресу;
- прибавить к принятому числу число, хранящееся в памяти по указанному в команде адресу;
- поместить полученный в АЛУ результат в память по адресу, указанному в данной команде.

Широкое распространение получили машины с переменной адресностью. В них при выполнении операций операнды (один либо оба) могут выбираться не из оперативной памяти, а из местной (СОЗУ). Команды этих машин по существу являются двухадресными, но оба адреса либо один из них могут быть адресами не оперативной памяти, а регистров местной памяти АЛУ.

Рассмотрим взаимодействие устройств ЭВМ в процессе решения задачи. Для определенности примем, что рассматриваемая ЭВМ является трехадресной. Перед решением задачи набор команд, образующий программу решения, помещается в последовательные ячейки оперативной памяти так, что адрес ячейки, содержащей следующую команду, на единицу больше адреса ячейки, в которую помещена предыдущая команда.

Процесс реализации программы состоит в последовательной выборке из ОЗУ команд и их исполнении. Вызванная из ОЗУ в устройство управления (УУ) очередная команда хранится в нем все время исполнения операции. УУ выбирает из команды первый адрес A_1, пересылает его в ОЗУ и подает сигнал считывания. Из ОЗУ выдается первый операнд. УУ подает в АЛУ сигнал отпирания входов регистра, в который должен быть принят этот операнд. Аналогично по второму адресу A_2 производится передача из ОЗУ в АЛУ второго операнда. Затем УУ подает в АЛУ управляющие сигналы, под действием которых выполняется предусмотренная командой операция. После получения результата операции УУ передает в ОЗУ третий адрес A_3, подает сигнал записи и открывает выход регистра АЛУ, хранящего результат операции.

Далее в ОЗУ передается адрес очередной команды, сформированной в УУ (например, путем увеличения на единицу адреса предыдущей команды), в УУ поступает следующая команда и т.д.

Система команд специализированного вычислителя БВЦ ТАКТ51

Состав системы команд и структура команд неразрывно связаны с архитектурой вычислительной системы.

В блоке цифрового вычислителя ТАКТ.51.51.000, как и во всех цифровых машинах с программным управлением, процесс переработки информации в итоге слагается из выполнения последовательности арифметических, логических и других операций. Тип каждой исполняемой операции, местоположение операндов, указание, куда поместить результат, и другая необходимая управляющая информация определяются командой, представленной в виде условного двоичного кода.

Система команд блока цифрового вычислителя – одноадресная. Для выполнения каждой операции требуется отдельная команда. При работе вычислителя команды исполняются одна за другой, так как они расположены в последовательных ячейках памяти, за исключением команд передачи управления, пропуска и обработки прерывания.

Текущий адрес команды находится в счетчике команд (СК). При естественном порядке выполнения команд из последовательных ячеек памяти СК увеличивается на единицу после выборки однословной команды и на два после выборки двухсловной команды.

Виды адресации в БВЦ:

- абсолютная и относительная (страничная);
- прямая и косвенная.

Адресация ячеек памяти в ОЗУ и ПЗУ БВЦ сквозная, начиная с ОЗУ. Диапазон адресов ОЗУ 0000_8–0777_8, диапазон адресов ПЗУ 1000_8–7777_8. Для обеспечения возможности обращения в любую ячейку памяти с помощью 12-разрядной команды в БВЦ используется страничная организация памяти. С этой целью вся память делится на страницы. Так как общий объем памяти 4096 ЯП (5I2 – ОЗУ, 3584 – ПЗУ), то память имеет всего 32 страницы. Нумерация страниц проводится в двоичном коде, начиная с нулевой страницы ОЗУ. Длина страницы памяти составляет 128 (200_8) ЯП, поэтому в ОЗУ располагаются только четыре первые страницы, начиная с нулевой (табл.2.3). Таким образом, максимальный относительный адрес равен 0177_8, т.е. в пределах текущей страницы, а максимальный абсолютный – 7777_8.

Таблица 2.3.

Диапазон адресов	Порядковый номер страницы
0000_8–0177_8	нулевая страница
0200_8–0377_8	первая страница
0400_8–0577_8	вторая страница
0600_8–0777_8	третья страница

Адресные команды обращения к памяти могут непосредственно адресовать одно из 256 слов: 128 слов в нулевой странице и 128 слов в текущей странице. Текущей страницей называется страница памяти, в которой хранится выполняемая команда.

В адресной команде (рис.2.3) выделен один разряд (4-й разряд – признак страницы ПСТ), указывающий, к какой странице производится обращение – к нулевой (если 4-й разряд содержит 0) или текущей (если 4-й разряд содержит 1). Для обращения к любой другой странице введено понятие косвенной адресации.

При косвенной адресации в адресной части команды указывается адрес ЯП, в который записан 12-разрядный адрес слова. При непосредственной (прямой) адресации, в отличие от косвенной, в адресной части команды указывается адрес ЯП, в который записано само слово. Для записи признака прямой или косвенной адресации в команде также выделен один разряд (3-й разряд – признак адресации ПКА). При наличии 1 в 3–м разряде выполняется косвенная адресация, а, при 0 – прямая.

На нулевой странице находятся модифицируемые ячейки памяти (в диапазоне 0010_8–0017_8). При обращении к ячейкам памяти из этого диапазона их содержимое сначала увеличивается на единицу, а уже затем выполняется предписанное действие.

Система команд вычислителя имеет обширный набор команд, обеспечивающий выполнение более 260 видов операций. По назначению команды вычислителя можно сгруппировать в следующие классы (рис.2.7):

- адресуемые команды или команды обращения к памяти;
- безадресные команды;
- специальные команды;
- арифметические команды;
- команды управления дополнительной памятью;
- команды обмена с внешними устройствами.

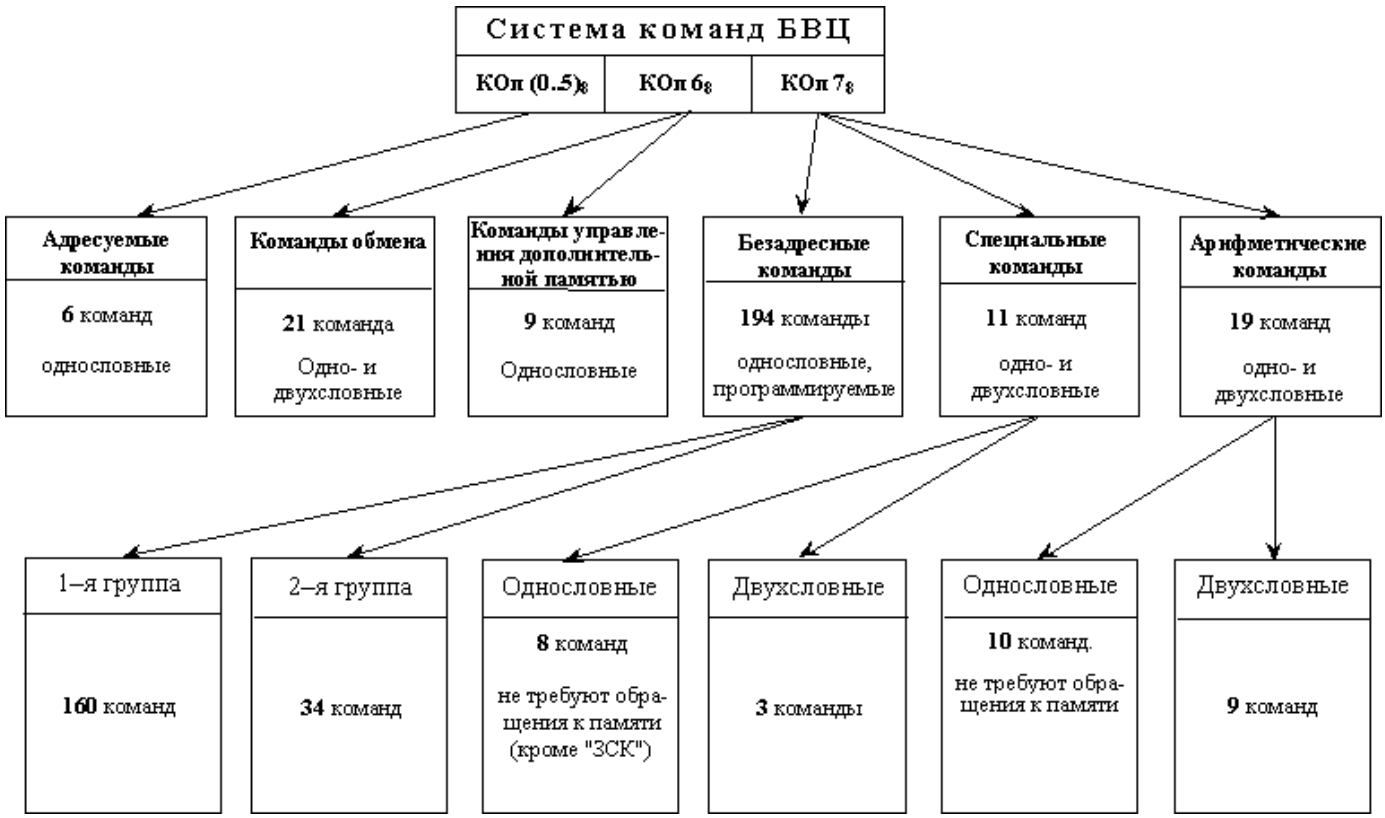


Рис.2.7. Система команд БВЦ.

Рассмотрим каждый класс команд отдельно.

1. Адресные (адресуемые) команды.

В адресных (адресуемых) командах кроме КОДа указывается адрес ЯП ЗУ, к которой необходимо обратиться за очередным адресом, операндом или для записи результата выполнения действия. Адресных команд шесть.

Это команды:

- Логическое умножение (КОД 000_2) – И;
- Сложение (КОД 001_2) – СЛ;
- Конец цикла (КОД 010_2) – КЦ;
- Запись (КОД 011_2) – ЗРС;
- Безусловная передача управления с возвратом (КОД 100_2) – БВП;
- Безусловная передача управления (КОД 101_2) – БП.

Логическое умножение.

Структура команды И.

0	1	2	3	4	5	6	7	8	9	10	11
0	0	0									
код			адресное слово								

Код команды 000 показывает, что это команда "Логическое умножение", которая определяет операцию конъюнкций РС:=РС П[А]. Регистру сумматора присваивается значение регистра сумматора, поразрядно логически умноженное на содержимое ЯП по адресу "А" согласно правилу логического умножения. Разряд знака умножается на общих основаниях. После выполнения операции содержимое ЯП с адресом "А" не изменяется.

Сложение.

Структура команды СЛ.

0	1	2	3	4	5	6	7	8	9	10	11
0	0	1									
код			адресное слово								

Код команды 001_2 показывает, что это адресная команда "Сложение", которая определяет операцию: РС:=РС+П[А]. В РС после выполнения данной команды будет занесено значение бывшего в РС операнда, суммированное со значением операнда, взятым из памяти по адресу "А" (адресу слова).

Конец цикла.

Структура команды КЦ.

0	1	2	3	4	5	6	7	8	9	10	11
0	1	0									
код			адресное слово								

Код команды 010_2 показывает, что это команда "конец цикла", которая определяет операцию П[А]:=П[А]+1. По этой команде производится алгебраическое сложение содержимого ЯП с адресом "А" с единицей. Результат снова заносится в ЯП с адресом "А" и анализируется. Если содержимое ЯП не равно нулю, то выполняется очередная команда, обеспечивающая возврат к началу цикла. Адрес этой команды формируется в счетчике команд в результате операции СК:=СК+1. Как только в ЯП появится ноль, содержимое СК увеличится на два СК:=СК+2, и следующая команда (команда зацикливания "БП") пропускается. Таким образом, прекращается выполнение цикла и идет продолжение программы.

Запись.

Структура команды ЗРС.

0	1	2	3	4	5	6	7	8	9	10	11
0	1	1									
код			адресное слово								

Код команды 011_2 показывает, что это команда "Запись", которая определяет операции П[А]:=РС; РС:=0. В ЯП по адресу "А" записывается содержимое РС, после чего РС обнуляется. Предыдущее значение ЯП стирается.

Безусловная передача управления с возвратом.

Структура команды БПВ.

0	1	2	3	4	5	6	7	8	9	10	11
1	0	0									
код			адресное слово								

Код команды 100_2 показывает, что эта команда "Безусловная передача управления возвратом", которая определяет операции: П[А]:=СК+1; СК:=П[А+1]. В ЯП по адресу "А" записывается содержимое СК, увеличенное на единицу (сформированный адрес возврата на продолжение программы). Управление передается команде, адрес которой находится в ЯП "А+1". В конце подпрограммы выполняется команда БП на ЯП с адресом "А".

Безусловная передача управления.

Структура команды БП.

0	1	2	3	4	5	6	7	8	9	10	11
1	0	1									
код			адресное слово								

Код команды 101_2 показывает, что эта команда "Безусловная передача управления", которая определяет операцию СК:=П[А]. Управление передается команде, адрес которой находится в ЯП с адресом "А", для чего содержимое ЯП записывается в СК. Таким образом, обеспечивается безусловный переход на другие участки программы.

2. Команды обмена с внешними устройствами.

Команды обмена с внешними устройствами имеют КОД 6_8. С помощью команд обмена осуществляется обмен информацией между ВЧУ и периферийными блоками (устройствами) по интерфейсному каналу и прием информации от блока ввода данных БВД. К этой же группе команд относятся команды управления режимом обмена и обработки прерывания интерфейса, а также команды управления дополнительной памятью.

3. Безадресные команды.

Безадресные команды имеют КОД 7_8 (111_2). Команды этого типа не требуют обращения к памяти. В них 3...11–й разряды используются под код операции. Все безадресные команды

делятся на две основные группы. Первая группа характеризуется наличием 0 в 3–м разряде. Вторая группа безадресных команд характеризуется единицей в 3–м разряде и нулем в 11–м разряде.

3.1. Безадресные команды первой группы.

0	1	2	3	4	5	6	7	8	9	10	11
1	1	1	0								
код			адресное слово								

Команды этой группы служат для кодирования операций обнуления и инвертирования регистра сумматора и дополнительного триггера, циклического сдвига РС в ТД вправо (влево) на один (два) разряда и увеличения на "1" содержимого РС.

3.2. Безадресные команды второй группы.

0	1	2	3	4	5	6	7	8	9	10	11
1	1	1	1								0
код			адресное слово								

Команды этой группы служат для кодирования операций обнуления РС, останова программы и операций, осуществляющих различного рода пропуски следующей команды по условию.

4. Специальные команды.

0	1	2	3	4	5	6	7	8	9	10	11
1	1	1						1	1		
код			адресное слово								

Специальные команды имеют код 7_8 и характеризуются наличием 1 в 8–м и 9–м разрядах. Специальные команды служат для кодирования операций: записи в РС содержимого регистров маски, ошибок, состояния программы и счетчика времени, установки и сброса регистров маски и маски контроля, записи содержимого счетчика команд в память, выдачи требуемой последовательности импульсов, проверки схем контроля. Для кодирования этих операций служат 4, 5, 10 и 11–й разряды. Специальные команды бывают одно– и двухсловные.

4.1. Однословные специальные команды.

Количество однословных специальных команд – восемь. Однословные команды не требуют обращения к памяти (за исключением команды "ЗСК").

4.2. Двухсловные специальные команды.

К двухсловным специальным командам относятся: команды обращения к устройству выдачи последовательностей и команда проверки контроля (идентификатор – ПК). Команда ПК используется для проверки исправности следующих схем контроля: тактового генератора, блока местного управления ОЗУ, на четность СК, РА, РЧ, РК и др. Проверка осуществляется путем имитации неисправности отдельно в каждом из контролируемых устройств. Если проверяемая схема контроля исправна, то возникает прерывание по аппаратному контролю. Если схема контроля неисправна, то управление передается следующей команде. Команда ПК занимает две расположенные друг за другом ячейки памяти. В первой ячейке записывается сама команда, во второй – слово, где в соответствующих разрядах указывается проверяемая схема контроля.

5. Арифметические команды.

0	1	2	3	4	5	6	7	8	9	10	11
1	1	1	1								1
код			адресное слово								

Арифметические команды имеют код 7_8 и характеризуются наличием "1" в 3–м и 11–м разрядах. Команды этого типа имеют идентификатор "АК" и делятся на две группы. К первой группе относятся однословные команды, не требующие обращения к памяти. Арифметические команды второй группы являются двухсловными и требуют обращения к памяти.

5.1. Однословные арифметические команды.

Разряды 4, 5, 6, 7 и 10 в командном слове используются для кодирования операций обмена между регистрами РС, РД (регистр дополнительный) и СС (счетчик сдвигов). Однословные арифметические команды микропрограммируются для выполнения непротиворечивых логических операций.

5.2. Двухсловные арифметические команды.

Команды инвертирования, умножения и деления, сдвиги влево и вправо. Команды этого типа занимают две расположенные друг за другом ячейки памяти. В первой ячейке записывается собственно команда, во второй – операнд или адрес ячейки, в которой находится операнд. В арифметическом регистре команд (РКА) устройства управления записываются 6, 8, 9 и 10–й разряды командного слова, которые определяют код арифметической операции. Тип информации (операнд или адрес операнда), расположенной во втором слове арифметической команды, определяется специальным триггером модификации ТМ.

МИКРОПРОЦЕССОРНЫЕ СИСТЕМЫ

Структура микропроцессорной системы

Процессор, построенный на одной или нескольких БИС, называется микропроцессором.

Набор БИС, обеспечивающих построение цифровых устройств, образует микропроцессорный комплект (МПК). Он позволяет совместно со сравнительно небольшим числом микросхем средней и малой степени интеграции создавать миниатюрные вычислительные устройства для разнообразных применений.

С помощью МПК реализуются микропроцессорные системы (МПС). Если в устройстве, построенном на принципе схемной логики, любое изменение или расширение выполняемых функций влечет демонтаж устройства и монтаж другого устройства по новой схеме, то в МПС благодаря использованию принципа программируемой логики изменение функций может быть достигнуто заменой хранящейся в памяти программы новой программой, соответствующей новым функциям устройства. Подобная гибкость вместе с другими связанными с использованием БИС достоинствами (низкой стоимостью, малыми размерами), а также высокая точность и помехозащищенность, характерные для цифровых методов, обусловили бурное внедрение МПС в различные сферы производства, научные исследования и бытовую технику.

Микропроцессорные системы в свою очередь обеспечили широкое использование цифровых методов в различных технических применениях, и размах внедрения этих новых методов рассматривается как революция в технике.

Рассмотрим структурную схему микропроцессорной системы (МПС), приведенную на рис. 4.1. Функционирование МПС сводится к следующей последовательности действий: получение данных от различных периферийных устройств (с клавиатуры терминала, от дисплеев, из каналов связи, от различного типа внешних запоминающих устройств), обработка данных и выдача результатов обработки на периферийные устройства (ПУ). При этом данные от ПУ, подлежащие обработке, могут поступать и в процессе их обработки.

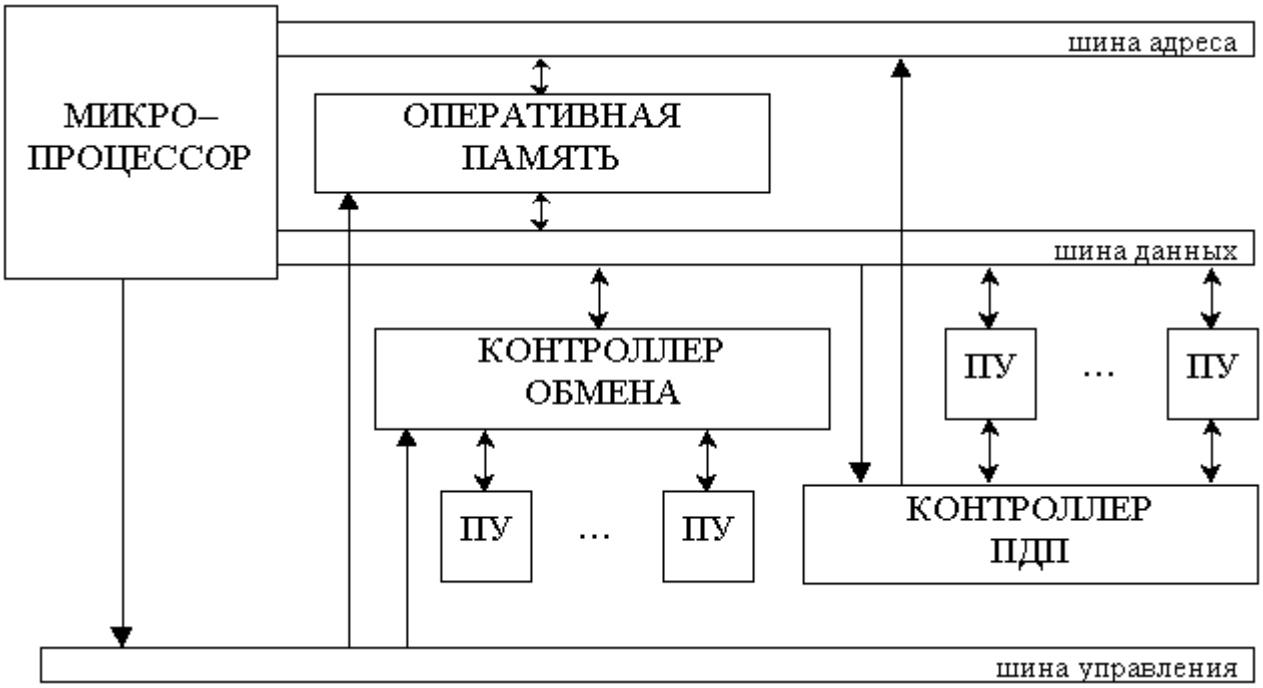


Рис.4.1. Структурная схема микропроцессорной системы.

Для выполнения этих действий в МПС кроме микропроцессора предусматриваются следующие устройства: - оперативная память (ОП), предназначенная для хранения и выдачи по запросам команд программ, определяющих работу микропроцессора, различных данных (исходных данных, промежуточных и конечных результатов обработки данных в микропроцессоре); - контроллеры - устройства, обеспечивающие обмен данными различных ПУ с микропроцессором и ОП.

Микропроцессор выдает на шину адреса номер (адрес) ячейки ОП, в которой хранится очередная команда, и из шины управления в ОП поступают сигналы, обеспечивающие считывание содержимого указываемой шиной адреса ячейки памяти. Оперативная память выдает запрошенную команду на шину данных, откуда она принимается в микропроцессор.

Здесь команда расшифровывается. Если данные, действия над которыми предусматривает команда, находятся в регистрах микропроцессора, то микропроцессор приступает к выполнению указанной в команде операции. Если при расшифровке команды выяснится, что участвующие в операции данные находятся в ОП, то микропроцессор выставляет на шину адреса адрес ячейки, хранящей эти данные; после выдачи данных из ОП микропроцессор принимает их через шину данных, затем выполняется операция над данными. После завершения текущей команды на шину адреса выдается адрес следующей команды, и описанный процесс повторяется.

Обмен данными с ПУ может осуществляться следующим образом. Группа ПУ подключается к шине данных МПС через контроллер обмена (устройства сопряжения), управляющий процессом обмена данными. До начала непосредственного обмена данными с ПУ микропроцессор через шину данных должен выдать в контроллер информацию о режимах, используемых при передаче, направлениях передачи данных (от микропроцессора к ПУ либо, наоборот, от ПУ к микропроцессору), используемых в дальнейшем при обмене данными с каждым из подключенных к контроллеру ПУ. Затем в момент, когда потребуется, например, передать в ОП выдаваемые из ПУ данные, микропроцессор, выполняя команду ввода, подает на контроллер соответствующие управляющие сигналы; данные из ПУ принимаются в регистр контроллера, откуда они затем контроллером выдаются на шину данных. Далее эти данные с шины данных принимаются в микропроцессор, после чего в процессе выполнения соответствующей команды они передаются в ОП.

Аналогично происходит обмен данными в обратном направлении - от ОП к ПУ. По соответствующей команде программы осуществляется прием из ОП в микропроцессор данных, подлежащих передаче, после чего по одной из следующих команд эти данные выдаются на шину данных и через контроллер обмена передаются на УП.

Описанный обмен предполагает, что моменты обмена данными известны заранее уже на этапе программирования, и в программе предусматриваются в определенных местах соответствующие команды, обеспечивающие обмен. Моменты обмена могут определяться и самим ПУ. Тогда эти моменты программисту оказываются неизвестными, он не может предусмотреть в программе соответствующие команды обмена. В этих случаях ПУ, подавая в микропроцессор определенные сигналы, переводит его в состояние так называемого прерывания. В этом состоянии микропроцессор прекращает выполнение основной программы и переходит к исполнению команд другой хранящейся в ОП программы (прерывающей программы), обеспечивающей обмен данными, требуемый периферийным устройством. После окончания такой прерывающей программы микропроцессор возвращается к выполнению основной программы.

Описанные способы обеспечивают низкую скорость обмена, и применять их целесообразно при обмене данными с низкоскоростными ПУ. При работе с высокоскоростными ПУ (такими, как запоминающие устройства на дисках и др.) используется так называемый режим прямого доступа к памяти (ПДП). В этом режиме микропроцессор отключается от шин адреса и данных, предоставляя их в распоряжение ПУ для непосредственного обмена данными с ОП (без участия микропроцессора). Обмен при этом организуется специальным контроллером ПДП.

В режиме ПДП ПУ обменивается с ОП не одиночными данными, а большими блоками данных. В контроллер ПДП микропроцессор предварительно помещает информацию, необходимую для управления обменом (адрес ячейки ОП, куда помещается или откуда считывается первое подлежащее обмену слово, количество слов в блоке и др.). В процессе обмена контроллер ПДП выдает на шину адреса адрес ячейки ОП, после окончания передачи слова между ОП и ПУ через шину данных контроллер ПДП увеличивает на единицу значение

адреса, выдаваемого на шину адреса. После завершения передачи заданного количества слов контроллер ПДП прекращает обмен, информируя об этом микропроцессор. Последний восстанавливает связь с шинами адреса и данных и продолжает выполнение программы.

Построение микропроцессоров с использованием различных микропроцессорных комплектов

Все элементы микропроцессоров с программируемой логикой - операционное устройство (ОУ), управляющая память (УП) и блок микропрограммного управления (БМУ) - могут размещаться на одном кристалле, т.е. весь микропроцессор может быть выполнен в виде одной микросхемы. Так реализованы микропроцессоры в отечественных сериях микропроцессорных комплектов КР580 и КР1810. Управляющая память микропроцессоров такого типа хранит набор микропрограмм, записанный в нее уже на этапе изготовления микросхемы на заводе. Каждая микропрограмма представляет собой последовательность микрокоманд, обеспечивающую выполнение некоторой несложной операции. При поступлении в микропроцессор команды из ОП в УП находится соответствующая команде микропрограмма и путем последовательного считывания ее микрокоманд осуществляется прием из ОП операндов, выполнение над ними некоторых простейших действий и вызов из ОП очередной команды. В микропроцессоре серии КР580 такие микропрограммы содержат от 4 до 17 микрокоманд. Применение микропроцессора, выполненного на одной микросхеме, естественно, упрощает построение микропроцессорной системы, сокращая число используемых в ней элементов. Кроме того, упрощается процесс программирования, так как от программиста не требуется записывать выполняемые в каждом такте микрокоманды. Составляя программу, он оперирует командами, т.е. хранящимися в микропроцессоре группами микрокоманд, которые соответствуют командам.

Однако такое облегчение программирования сопровождается существенным снижением скорости решения задачи. Это связано со следующим. Система команд, которой снабжается микропроцессор при его заводском изготовлении, универсальна в том смысле, что она позволяет программировать решение любой задачи. Но при решении конкретной задачи такая фиксированная система команд может оказаться неэффективной: пользование ею потребует большого числа команд, на выполнение которых микропроцессор будет затрачивать много времени. Программа оказывается более эффективной (требующей меньшей емкости памяти для ее хранения и меньшего времени для исполнения), если для ее построения используется специально подобранная для данной конкретной задачи система команд. Такой прием с введением новых составленных программистом команд (т.е. модификация системы команд) оказывается невозможным в микропроцессорах, реализованных в виде одной микросхемы.

В тех случаях, когда требуется обеспечивать высокую скорость решения задачи, у разработчика микропроцессорного устройства возникает желание самому разработать систему команд, наилучшим образом приспособленную к решению конкретной задачи. При этом он должен знать, что ему придется преодолеть ряд трудностей, связанных с необходимостью определения состава команд и построения для каждой команды соответствующей микропрограммы, если программирование ведется на языке микрокоманд. Составленные таким образом микропрограммы затем записываются в постоянное запоминающее устройство управляющей памяти.

Рассмотрим, к каким изменениям в структуре микропроцессора приводит обеспечение указанной выше возможности программирования на языке микрокоманд.

При создании микросхемы приходится решать трудную проблему сокращения числа выводов. В представленном на рис.4.2,а варианте с совмещением в общей микросхеме всех элементов микропроцессора (ОУ, БМУ, УП) эта задача решается обычно путем мультиплексирования шин. Например, в микропроцессоре серии КР580 для 8-разрядных

выходов и входов используются общие выводы, которые переключаются в зависимости от направления передачи данных либо на ввод, либо на вывод данных; в микропроцессоре серии КР1810, оперирующем 16-разрядными данными и 20-разрядными адресами ОП, кроме объединения входов и выходов данных предусматривается использование этих выводов и для части разрядов адресной информации (при этом, очевидно, необходимо предусмотреть выдачу адреса и выдачу или прием данных в различные временные интервалы).

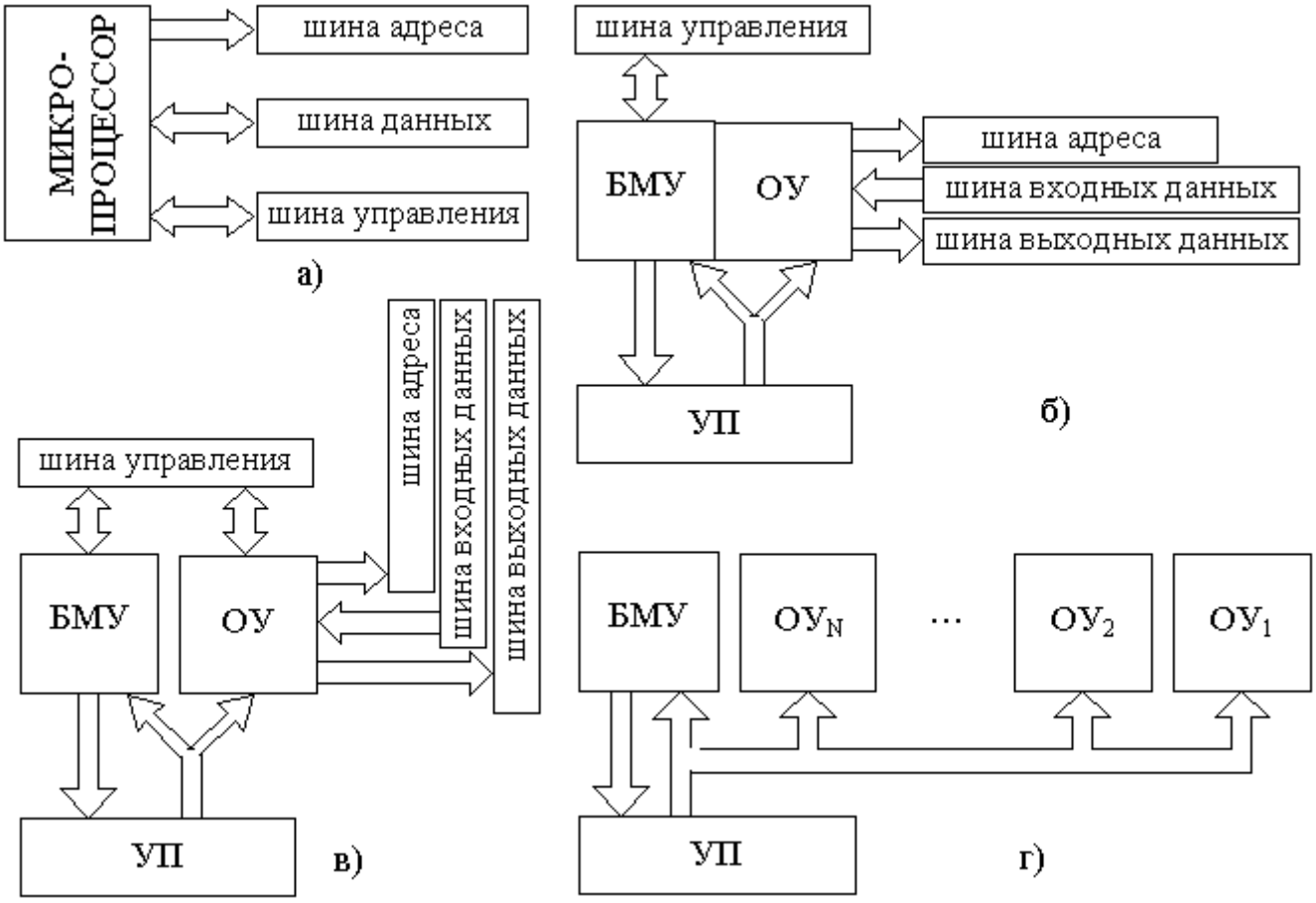


Рис.4.2. Варианты структур микропроцессора.

Для того чтобы разработчик микропроцессорного устройства имел возможность программировать на языке микрокоманд, он должен иметь доступ к УП для записи в нее составленных микропрограмм. Такой доступ можно обеспечить, если УП вынести из микросхемы процессора, как показано на рис.4.2,б. Из сравнения схем на рис.4.2,а и 4.2,б видно, что вариант на рис.4.2,б потребует в микросхеме, содержащей ОУ и БМУ, предусмотреть существенно большее число выводов, чем в микросхеме на рис.4.2,а. Это связано с необходимостью иметь в варианте на рис.4.2,б выходы для передачи в УП адреса и входы для приема микрокоманды из УП. В результате такое построение практически окажется нереализуемым из-за чрезмерно большого числа выводов, которые пришлось бы предусмотреть в микросхеме. В этом случае для сокращения числа выводов следует ОУ и БМУ выполнять не в общей микросхеме, а разнести в разные микросхемы, как показано на рис.4.2,в. Так как обеспечение высокого быстродействия требует отказа от мультиплексирования шин, то и в данном варианте число выводов в микросхеме ОУ окажется недопустимо большим. Число выводов можно сократить, если построить микросхему ОУ на небольшое число разрядов обрабатываемых данных (2-, 4-, 8-разрядных данных) и обеспечить возможность наращивать разрядность ОУ путем объединения соответствующего числа микросхем, как показано на рис.4.2,г.

Полупроводниковые запоминающие устройства

5.1.1. Классификация и параметры запоминающих устройств

Для хранения небольших массивов кодовых слов могут использоваться регистры. Но уже при необходимости хранить десятки слов применение регистров приводит к неоправданно большим аппаратурным затратам. Для хранения больших массивов слов строят запоминающие устройства (ЗУ) с использованием специальных микросхем, в каждой из которых может храниться информация объемом в тысячи битов.

Основными операциями в памяти в общем случае являются занесение информации в память ("запись") и выборка информации из памяти ("чтение"). Обе эти операции осуществляют ("обращение к памяти").

При обращении к памяти производится чтение или запись некоторой единицы данных - различной для устройств разного типа. Такой единицей может быть, например, байт, машинное слово или совокупность машинных слов (зона, сектор, блок) для внешних ЗУ.

Производительность и вычислительные возможности ЭВМ в значительной степени определяются составом и характеристиками ее ЗУ. В составе ЭВМ используется одновременно несколько типов ЗУ, отличающихся принципом действия, характеристиками и назначением. Важнейшими характеристиками ЗУ являются емкость памяти, удельная емкость, быстродействие и разрядность.

ЗУ классифицируется по характеру хранения информации, по физическим принципам работы и по назначению.

По характеру хранения информации ЗУ могут быть статическими и динамическими. В статических ЗУ физическое состояние носителя информации в процессе хранения не изменяется. В динамических ЗУ состояние носителя постоянно изменяется. По физическим принципам работы ЗУ делятся на магнитные, электронные, оптические, механические, криогенные.

По назначению различают следующие типы запоминающих устройств: оперативное запоминающее устройство (ОЗУ), постоянное запоминающее устройство (ПЗУ), перепрограммируемое постоянное запоминающее устройство (ППЗУ).

Оперативное ЗУ используется в условиях, когда необходимо выбирать и обновлять хранимую информацию в высоком темпе работы процессора цифрового устройства. Вследствие этого в ОЗУ предусматриваются три режима работы: режим хранения при отсутствии обращения к ЗУ, режим чтения хранимых слов и режим записи новых слов. При этом в режимах чтения и записи ОЗУ должно функционировать с высоким быстродействием (обычно время чтения или записи слова в ОЗУ составляет доли микросекунды). В цифровых устройствах ОЗУ используются для хранения данных (исходных данных, промежуточных и конечных результатов обработки данных) и программ.

Постоянное ЗУ предназначено для хранения некоторой однажды записанной в него информации, не нарушаемой и при отключении источников питания. В ПЗУ предусматриваются два режима работы: режим хранения и режим чтения с высоким быстродействием. Режим записи не предусматривается. Используются ПЗУ для хранения программ в таких специализированных цифровых устройствах, которые, функционируя

длительное время, многократно выполняют действия по одному и тому же алгоритму при различных исходных данных.

Перепрограммируемое ПЗУ в процессе функционирования цифрового устройства используется как ПЗУ. Оно отличается от ПЗУ тем, что допускает обновление однажды занесенной информации, т.е. в нем предусматривается режим записи. Однако в отличие от ОЗУ запись информации требует отключения ППЗУ от цифрового устройства, производится с использованием специальных предназначенных для записи устройств (программаторов) и занимает длительное время, достигающее десятков минут. Перепрограммируемые ПЗУ дороже ПЗУ, и их применяют в процессе отладки программы, после чего их можно заменить более дешевым ПЗУ.

Запоминающее устройство содержит некоторое число N ячеек, в каждой из которых может храниться слово с определенным числом разрядов n. Ячейки последовательно нумеруются двоичными числами. Номер ячейки называется адресом. Если для представления адресов используются комбинации m-разрядного двоичного кода, то число ячеек в ЗУ может составить N = 2m.

Количество информации, которое может храниться в ЗУ, определяет его емкость. Емкость можно выражать числом ячеек N с указанием разрядности n хранимых в них слов в форме N * n, либо ее можно определять произведением N и n: M = N * n бит. Часто разрядность ячеек выбирают кратной байту (1 байт равен 8 битам). Тогда и емкость удобно представить в байтах. Большие значения емкости часто выражаются в единицах К = 210 = 1024. Например, М = 64 Кбайт означает емкость, равную М = 64 * 1024 * 8 бит.

Быстродействие ЗУ характеризуется двумя параметрами: временем выборки tв представляющим собой интервал времени между моментом подачи сигнала выборки и появлением считанных данных на выходе, и циклом записи tцз, определяемым минимально допустимым временем между моментом подачи сигнала выборки при записи и моментом, когда допустимо последующее обращение к памяти.

Запоминающие устройства строятся из набора однотипных микросхем ЗУ с определенным их соединением. Каждая микросхема ЗУ кроме времени обращения и емкости характеризуется потребляемой мощностью, набором питающих напряжений, типом корпуса (числом выводов). Микросхемы ППЗУ дополнительно характеризуются временем хранения записанной в них информации (по истечении которого хранящаяся в ячейках информация может самопроизвольно измениться), допустимым количеством циклов перезаписи (после чего микросхема считается негодной для использования).

5.2.1. Оперативное запоминающее устройство

На рис. 5.1 приведена типичная структура микросхемы ОЗУ. Информация хранится в накопителе. Он представляет собой матрицу, составленную из элементов памяти (ЭП), расположенных вдоль строк и столбцов. Элемент памяти может хранить 1 бит информации (лог.0 либо лог.1). Кроме того, он снабжен управляющими цепями для установки элемента в любом из трех режимов: режиме хранения, в котором он отключается от входа и выхода микросхемы, режиме чтения, в котором содержащаяся в ЭП информация выдается на выход микросхемы, режиме записи, в котором в ЭП записывается новая поступающая со входа микросхемы информация.

Каждому ЭП приписан номер, называемый адресом элемента. Для поиска требуемого ЭП указывается строка и столбец, соответствующие положению ЭП в накопителе. Адрес ЭП в виде двоичного числа принимается по шине адреса в регистр адреса. Число разрядов адреса связано

с емкостью накопителя. Число строк и число столбцов накопителя выбираются равными целой степени двух. И если число строк Nстр = 2n1 и число столбцов Nст = 2n2, то общее число ЭП (емкость накопителя):

$$N = Nстр * Nст = 2n1 * 2n2 = 2*(n1+n2) = 2*n,$$

где n = n1 + n2 - число разрядов адреса, принимаемого в регистр адреса.

Например, при емкости N = 210 = 1024 число разрядов адреса n = 10; при этом выбирается n1 = n2 = n/2 = 5, в этом случае число строк и число столбцов накопителя 2*n1 = 2*n2 = 32.

Интерфейсы ввода-вывода

Вспомогательные элементы микропроцессорных устройств

СОВРЕМЕННЫЕ МИКРОПРОЦЕССОРЫ И ПЕРСПЕКТИВЫ ИХ РАЗВИТИЯ

Тенденции развития микропроцессоров

Современный уровень развития микропроцессорной техники достиг такого уровня что, в течение 5 лет происходит смена двух-трех поколений микропроцессоров. По прогнозам аналитиков к 2012 году число транзисторов в микропроцессоре достигнет 1 млрд., тактовая частота возрастет до 10 ГГц, а производительность достигнет 100 млрд. оп/с.

Поэтому при выборе аппаратно-программной платформы необходимо учитывать тенденции развития, позволяющие минимизировать затраты на модернизацию и поддержку актуального программного обеспечения.

В архитектуре современных микропроцессоров разных компаний-производителей имеется много общего. В предыдущих поколениях микропроцессоров при ограниченном объеме аппаратных ресурсов каждый разработчик микропроцессора выбирал ряд архитектурно-структурных приемов повышения производительности, за счет преимущественного развития которых этот микропроцессор должен был превосходить другие. В современных условиях большое число транзисторов на кристалле делает возможным применить в одном микропроцессоре все известные приемы повышения производительности, сообразуясь только с их совместимостью.

Рассмотрение конкретных семейств микропроцессоров разных производителей подтверждает общие тенденции их развития: повышение тактовой частоты, увеличение объема и пропускной способности подсистемы памяти, увеличение количества параллельно функционирующих исполнительных устройств.

Повышение тактовой частоты

Главным препятствием на пути повышения тактовой частоты служат внутрисхемные соединения. Для преодоления этого препятствия необходима разработка новых материалов с меньшими сопротивлением и емкостью, а также разработка новых методов многослойной металлизации.

Для повышения тактовой частоты при выбранных материалах используются более совершенный технологический процесс с меньшими проектными нормами, увеличение числа слоев металлизации, более совершенная схемотехника меньшей каскадности и с более

совершенными транзисторами, а также более плотная компоновка функциональных блоков кристалла.

Так все производители перешли на технологию КМОП, хотя Intel, например, использовала БиКМОП для первых представителей семейства Pentium. Еще более экзотическая попытка была предпринята компанией Exponential Technology, которая пыталась развить биполярную схемотехнику для производства процессоров с архитектурой Power PC и x86. Однако на частоте 466 МГц с кристалла площадью 150 кв. мм выделялось около 80 Ватт, что создало серьезную проблему теплоотвода (без этого кристалл превращается в святящуюся раскаленную пластину металла). Компания пыталась добиться повышения тактовой частоты до обещанного уровня, но не справилась с этим, по ее утверждению, из-за экономических проблем.

Вообще говоря, известно, что биполярные схемы и КМОП на высоких частотах имеют примерно одинаковые показатели тепловыделения, но КМОП схемы более технологичны, что и определило их преобладание в микропроцессорах.

Уменьшение размеров транзисторов, сопровождаемое снижением напряжения питания с 5 В до 2,5 - 3 В и ниже, увеличивает быстродействие и уменьшает выделяемую тепловую энергию. Все производители микропроцессоров перешли с проектных норм 0,7-0,5 мкм на 0,35 мкм и 0,25 мкм, а сейчас уверенно стремятся к 0,18 мкм и 0,12 мкм. Благодаря использованию уникальной 0,07 мкм технологии изготовления КМОП микросхем фирма Texas Instr. обещает к 2001 году разместить на пластине 100 мм2 более 400 млн. транзисторов.

Если диапазон линейных размеров вплоть до 0,1 мкм представлял собой прежде всего технический барьер, который к настоящему времени успешно преодолен за счет использования современных технологических решений, то дальнейшая миниатюризация наталкивается на фундаментальный физический барьер. Практически все свойства твердого тела, включая его электропроводность, резко изменяются и "сопротивляются" дальнейшей миниатюризации. Если до ширины проводника 0,1 мкм еще как-то можно было опираться на принцип пропорциональной миниатюризации, то за этим барьером возрастание сопротивления связей происходит экспоненциально. Потери даже на кратчайших линиях внутренних соединений такого размера "съедают" до 90% сигнала по уровню и мощности.

При минимальном размере деталей внутренней структуры интегральных схем 0,1-0,2 мкм достигается оптимум, ниже которого все характеристики транзистора быстро ухудшаются. При этом начинают проявляться эффекты квантовой связи, в результате чего твердотельное устройство становится системой, действие которой основано на коллективных электронных процессах. Проектная норма 0,05-0,1 мкм (50-100 нм) - это нижний предел твердотельной микроэлектроники, основанной на классических принципах синтеза схем. Сейчас работы в области нанотехнологий ведутся в четырех основных направлениях:
- молекулярная электроника;
- биохимические и органические решения;
- квазимеханические решения на основе нанотрубок;
- квантовые компьютеры.

Три последних пребывают пока еще в значительной степени на уровне теоретических работ. Наибольшие практические результаты достигнуты в области молекулярной электроники. Она логически близка к традиционной полупроводниковой электронике. Методами молекулярной электроники из углеводородных соединений удается получить аналоги диодов и транзисторов, а следовательно, и основные булевы модули И, ИЛИ и НЕ, из которых затем можно строить схемы любой сложности. Подобный подход позволяет сохранить преемственность архитектурных решений.

В 1999 году сотрудники компании Hewlett-Packard и Калифорнийского университета в Лос-Анджелесе (UCLA) смогли получить действующий молекулярный вентиль. Его толщина составляет всего одну молекулу. Первоначально он умел либо только открываться, либо только закрываться. Исследователи из Йельского университета смогли продвинуться дальше: их вентиль может принимать любое из двух положений, что позволяет произвольно записывать в него 0 или 1. Обе группы сейчас работают над объединением вентилей в регистры.

В обстановке строгой секретности разрабатываются молекулярные устройства памяти с произвольным доступом, но одним из первых их воплощений скорее всего станут постоянные ЗУ. Есть основания считать, что на одном современном кристалле кремния можно будет записать целиком фильм с качеством, как на диске DVD.

Проблема уменьшения длины межсоединений на кристалле при использовании традиционных технологий решается путем увеличения числа слоев металлизации. Так Cyrix при сохранении 0,6 мкм КМОП технологии за счет увеличения с 3 до 5 слоев металлизации сократила размер кристалла на 40% и уменьшила выделяемую мощность, исключив существовавший ранее перегрев кристаллов.

Уменьшение длины межсоединений актуально для повышения тактовой частоты работы, так как существенную долю длительности такта занимает время прохождения сигналов по проводникам внутри кристалла. Так в Alpha 21264 предприняты специальные меры по кластеризации обработки, призванные локализовать взаимодействующие элементы микропроцессора.

Одним из шагов в направлении уменьшения числа слоев металлизации и уменьшения длины межсоединений стала технология, использующая медные проводники для межсоединений внутри кристалла, разработанная фирмой IBM и используемая в настоящее время и другими фирмами-изготовителями СБИС.

Впервые рубеж тактовой частоты в 500 Мгц перешагнули микропроцессоры фирмы DEC, которая уже в конце 1996г. поставляла Alpha 21164 с тактовой частотой 500 МГц, в 1997г. - Alpha 21264 с тактовой частотой 600 МГц, а в 1998г. - Alpha 21264 с тактовой частотой 750 МГц и выше. К настоящему времени ряд фирм выпускает процессоры для персональных компьютеров с тактовой частотой свыше 1 ГГц.

Увеличение пропускной способности подсистемы памяти

Спектр возможных решений по увеличению пропускной способности подсистемы памяти, снабжающей функциональные устройства процессора работой, включает создание кэш-памятей одного или нескольких уровней, а также увеличение пропускной способности интерфейсов между процессором и кэш-памятью и конфликтующим с этим увеличением пропускной способности между процессором и основной памятью. Совершенствование интерфейсов реализуется как увеличением пропускной способности шин (путем увеличения частоты работы шины и/или ее ширины), так и введением дополнительных шин, расшивающих конфликты между процессором, кэш-памятью и основной памятью. В последнем случае одна шина работает на частоте процессора с кэш-памятью, а вторая - на частоте работы основной памяти. При этом частоты работы второй шины, например, равны 66, 66, 166 Мгц для микропроцессоров Pentium Pro-200, Power PC 604E-225, Al-pha 21164-500, работающих на тактовых частотах 300, 225, 500 Мгц соответственно. При ширине шин 64, 64, 128 разрядов это обеспечивает пропускную способность интерфейса с основной памятью 512, 512, 2560 Мбайт/с, соответственно. К моменту начала поставок Alpha 21264 на рынок поступили SRAM с частотой 200-250 МГц и пропускной способностью 3,2-4 Гбайт/с.

Общая тенденция увеличения размеров кэш-памяти реализуется по-разному:
- внешние кэш-памяти данных и команд с двухтактовым временем доступа объемом от 256 Кбайт до 2 Мбайт со временем доступа 2 такта в HP PA-8000;
- отдельный кристалл кэш-памяти второго уровня, размещенный в одном корпусе в Pentium Pro
- размещение отдельных кэш-памятей команд и данных первого уровня объемом по 8 Кбайт и общей для команд и данных кэш-памяти второго уровня объемом 96 Кбайт в Alpha 21164.

Наиболее используемое решение состоит в размещении на кристалле отдельных кэш-памятей первого уровня для данных и команд с возможным созданием внекристальной кэш-памяти второго уровня. Например, в Pentium II использованы внутрикристальные кэш-памяти первого уровня 16 Кбайт команд и 16 Кбайт данных, работающие на тактовой частоте процессора и внекристальный кэш второго уровня, работающий на половинной тактовой частоте.

Линии развития всех микропроцессоров укладываются в эту общую схему, используя кэш-памяти первого уровня:
- HP PA 8500 - 0,5 Мбайт кэш-команд и 1 Мбайт кэш-данных со временем доступа два такта;
- Alpha 21264 - 64 Кбайт кэш-команд и 64 КБайт кэш-данных со временем доступа два такта.

Повышение степени внутреннего параллелизма

Каждое семейство микропроцессоров демонстрирует в следующем поколении увеличение числа функциональных исполнительных устройств и улучшение их характеристик, как временных (сокращение числа ступеней конвейера и уменьшение длительности каждой ступени), так и функциональных (введение MMX расширений системы команд и т.д.).

В настоящее время процессоры могут выполнять до 6 операций за такт. Однако число операций с плавающей точкой в такте ограничено 2 для R10000 и Alpha 21164, а 4 операции за такт делает HP PA-8r00.

Для того чтобы загрузить функциональные исполнительные устройства, используется переименование регистров и предсказание переходов, устраняющие зависимости между командами по данным и управлению, буферы динамической переадресации. Устоявшихся решений в этой области практически нет, так как каждый микропроцессор демонстрирует изобретательность его создателей по симбиозу аппаратных средств и компилятора для статического и динамического устранения зависимостей между командами.

Обращается внимание на использование архитектур с VLIW с длинным командным словом. Так архитектура IA-64, развиваемая Intel и HP, использует объединение нескольких инструкций в одной команде (так называемая архитектура Ecplicity Parallel Instruction Computing - EPIC). Это позволяет упростить процессор и ускорить выполнение команд. Процессоры с архитектурой IA-64 могут адресоваться к 4 Гбайтам памяти и работать с 64-разрядными данными. Архитектура IA-64 используется в микропроцессоре Merced, обеспечивая производительность до 6 Гфлоп при операциях с одинарной точностью и до 3 Гфлоп - с повышенной точностью на частоте 1ГГц.

Системы на одном кристалле и новые технологии

В настоящее время получили широкое развитие системы, выполненные на одном кристалле - SOC (System On Chip). По сути дела в одной микросхеме заключается целый компьютер. Сфера применения SOC - от игровых приставок до телекоммуникаций. Такие кристаллы требуют применения новейших технологий.

Основной технологический прорыв в области SOC удалось сделать корпорации IBM, которая в 1999 году смогла реализовать сравнительно недорогой процесс объединения на одном кристалле логической части микропроцессора и оперативной памяти. В новой технологии, в частности, используется так называемая конструкция памяти с врезанными ячейками (trench cell). В этом случае конденсатор, хранящий заряд, помещается в некое углубление в кремниевом кристалле. Это позволяет разместить на нем свыше 24 тыс. элементов, что почти в 8 раз больше, чем на обычном микропроцессоре, и в 2-4 раза больше, чем в микросхемах памяти для ПК. Следует отметить, что хотя кристаллы, объединяющие логические схемы и память на одном кристалле, выпускались и ранее, например, такими фирмами, как Toshiba, Siemens AG и Mitsubishi, подход, предложенный IBM, выгодно отличается по стоимости. Причем ее снижение никоим образом не сказывается на производительности.

Использование новой технологии открывает широкую перспективу для создания более мощных и миниатюрных микропроцессоров и помогает создавать компактные, быстродействующие и недорогие электронные устройства: маршрутизаторы, компьютеры, контроллеры жестких дисков, сотовые телефоны, игровые и Интернет-приставки.

Для создания SOC IBM использует самые современные технологические решения, одним из которых являются медные межсоединения (copper interconnect). Первым микропроцессором IBM с медными межсоединениями в 1998 г. стал PowerPC 750.

Вообще говоря, по сравнению с технологией, где межсоединения выполнены на основе алюминия, медь позволяет сделать кристалл меньшим по размеру и более быстродействующим. Медная металлизация уменьшает общее сопротивление, что позволяет увеличить скорость работы кристалла на 15-20%. Обычно эта технология дополняется еще одной новинкой - технологией "кремний на изоляторе" - КНИ (SOI, Silicon On Insulator). Она уменьшает паразитные емкости, возникающие между элементами микросхемы и подложкой. Благодаря этому тактовую частоту работы транзисторов также можно увеличить. Возрастание скорости от использования КНИ приближается к 20-30%. Таким образом, общий рост производительности в идеальном случае может достигнуть 50%. В настоящее время увеличение стоимости процесса производства при использовании КНИ не превышает 10%, хотя буквально пару лет назад цены на КНИ-пластину и обычную восьмидюймовую кремниевую пластину различались чуть ли не на порядок.

Микропроцессорная технология потенциально имеет много назначений: создание персональных электронных партнеров, интеллектуализация (в известном смысле "оживление") всей техносферы, усиление и защита функций организма с помощью персональных медико-кибернетических устройств, в том числе вживляемых в организм...

В результате эволюции электронной технологии от "микро" к "нано" и ее слияния с "генной", вероятно, будет достигнуто состояние, при котором станет возможным синтез в массовых количествах любых технических устройств. Однако вряд ли в этом состоит основная цель будущей нано-технологии. Она, по всей вероятности, сможет синтезировать структуры, способные к эволюции и саморазвитию.