1. Типовая структура микропроцессора и ее основные блоки.

Появление МП сыграло революционную роль в автоматике и вычислительной технике, послужив мощным ускорителем развития средств и систем управления и контроля. с созданием МП впервые появилась возможность применять вычислительные устройства в таких областях, где раньше это казалось неэффективным либо просто невозможным. Уже первые использования МП определили два основных направления их применения: использование МП в качестве центральных процессоров — ВМ и реализация на базе МП встраиваемых систем управления различными объектами.

В самом общем случае функциональную схему МП можно представить в виде композиции трех функциональных блоков: операционного блока (ОБ), блока управления и интерфейсного блока. Кроме них в состав микропроцессора могут входить и некоторые другие блоки, участвующие в организации вычислительного процесса, например, блок прерывания, блок защиты памяти, блоки контроля, диагностики и др.

Операционный блок. Предназначен для выполнения некоторого функционально полного набора логических и арифметических операций. Как правило, в его состав входят АЛУ, буферные регистры операндов, регистр результата (аккумулятор), регистр признаков и блок регистров общего назначения (РОН). Комбинационная схема, являющаяся основой АЛУ, содержит двоичный сумматор и набор логических схем. В АЛУ выполняются несколько простейших арифметических (сложение, вычитание) и поразрядных логических (И, ИЛИ, НЕ и др.) операций.

Важной составляющей ОБ современных МП является блок внутренней памяти, реализованный в виде набора программно доступных регистров, называемых регистрами общего назначения (РОН). Время обращения к Рон меньше, чем к любым другим устройствам Памяти, поэтому память на РОН называется сверхоперативной, а устройство, в виде которого она реализована, — сверхоперативным запоминающим устройством (СОЗУ).

Блок управления. В процессе выполнения программы блок управления координирует работу всех блоков МП и микропроцессорной системы в целом. С помощью блока управления формируются управляющие сигналы, необходимые для организации обмена информацией с внешними устройствами, и обеспечивается выборка команд программы из памяти. В целом блок управления выполняет следующие действия:

• считывает и запоминает текущую команду;

• формирует адрес следующей команды;

• реализует выполнение по тактам алгоритма поступившей команды;

• управляет обменом информацией с внешними устройствами по сис. шине.

Блок управления состоит из регистра команд (PrK), дешифратора команд (

ДК) и блока формирования управляющих сигналов (БФУС). Управляющие сигналы с выходов БФУС поступают на управляющие входы других блоков МП, настраивая их на Выполнение определенных микроопераций. В состав блока управления также включают программно доступные счетчик команд PC (Program Counter) и указатель стека SP (Stack Pointer).

Интерфейсный блок. Предназначен для организации взаимодействия МП с памятью и устройствами ввода-вывода, расположенными на системной шине процессора, а также для обмена данными между ОБ и внутренними устройствами МП. Непосредственное подсоединение устройств ввода-вывода к МП осуществляется с помощью специальных схем сопряжения, которые называются интерфейсом ввода-вывода. В общем случае интерфейсный блок процессора должен выполнять следующие функции

• формировать выходные сигналы на шинах адреса, данных и управления в режиме вывода;

• формировать выходные сигналы адреса и управления и считывать (воспринимать) сигналы с шины данных в режиме ввода;

• синхронизировать процессы внутри процессора и на системной шине;

• реализовывать стандартный для системной шины протокол обмена.

Системная шина объединяет сигналы шин данных, адреса и управления. Протокол обмена информацией по с определяет последовательности сигналов (временную диаграмму сигналов в шине), обеспечивающих правильную передачу информации между устройствами микропроцессорной системы.

2. Формат команды микропроцессора. Назначение основных полей команды. Особенности основных типов команд: безадресных, 1-, 2- и 3 -адресных.

Система команд является одной из важнейших архитектурных характеристик процессора ВМ в целом. В понятие системы команд входит:

- форматы команд
- список команд
- способы адресации данных

Команды у любого МП делятся на группы:

- **пересылки данных** обеспечивают обмен информацией между регистрами и внешние обмены данными при передачи в МП из памяти устройства ввода и наоборот, также порты ввода - вывода. (load, store, exchange, input, output)
- **арифметических и поразрядных логических операций** различные сдвиги и банальные арифметические операции(add, sub, or, xor, and) умножение здесь представлено невсегда - иногда в связи с особенностью архитектуры процессора.
- **передачи управления необходимы** необходимы для ветвления сложных программ. Несколько команд условных переходов, команда останова, вызов подпрограммы(call, ret, jz, jnz ...)

Дополнительно бывают различные расширяющие возможности МП, группы команд для работы с числами с плавающей запятой, для умножения.

Структура команд в общем случае содержит операционную и адресную части. В операционной части - n - k разрядов, содержащих код операции, обеспечивающий кодирование 2 ^ (n - k) операций в оставшейся части - адресная часть.
В зависимости от указываемого числа адресов команды подразделяются на безадресные (0-адресные), 1-, 2-, 3- и 4-адресные. Практически во всех МП в адресном поле команды исключен адрес А4.Из-за того, что команды относятся к линейным участкам алгоритмов и такие команды могут быть размещены в ячейках ОП с последовательно возрастающими адресами. Просто инкрементируем адрес - этот способ адресации команд называется естественным. Использование адреса результата А3 также часто оказывается избыточным, поскольку помещаем на место одного из операндов(так повышаем производительность). При переходе к 2-адресным командам в их адресное поле необходимо вводить дополнительные разряды, кодирующие назначение адресуемых операндов: кто из них является источником, а кто - приемником информации. По схеме 2-адресного вычислителя реализованы все процессоры семейства x86 компании Intel, большинство процессоров фирмы Motorola и др.
В МП аккумуляторной архитектуры число адресов в адресной части команды уменьшено до одного. В них один из операндов, размещенный в аккумуляторе, неявно задается кодом команды, и результат помещается в аккумулятор. По схеме 1-адресного вычислителя реализованы МП 8080, 8085, MCS-51 компании Intel и ряд других.

Наконец, существует относительно небольшая группа безадресных команд, в которых осуществляется безадресное (неявное) задание операнда. К безадресным командам относятся команды управления процессором, например, пуска, останова, реализующие операции со стеком(SP неявно

задается кодом команды). Безадресные команды за счет исключения адресного поля имеют предельно сокращенный формат, но не могут образовать функционально полную систему команд и применяются вместе с адресными.

Число адресов, указываемых в команде, влияет на время решения задач, затраты памяти, сложность процессора и зависит от класса решаемых задач. В частности, для научно-технических расчетов - более эффективными оказываются 1-адресные команды и безадресные команды. Для задач управления(основа - пересылки), где 2 - адресные команды. Обычно используют только безадресные, 1- и 2-адресные команды. 3-адресные команды используют очень редко, а 4-адресные не используют.

Рассмотренные способы указания адресов операндов иногда используют для классификации МП по числу адресуемых в команде операндов. В соответствии с этим классификационным признаком различают безадресные, 1-, 2- и 3-адресные архитектуры.

3. Способы адресации. Формирование исполнительного адреса операнда при относительной адресации с помощью базирования. Использование базирования при организации виртуальной памяти.

Основными способами адресации являются:

- прямая
- непосредственная
- неявная,
- косвенная и относительная адресации.

Встречаются и другие способы представляющие собой разновидности и комбинации перечисленных способов. Далее используются следующие обозначения. Адрес указываемый в команде-Ак, а адрес физической ячейки памяти к которой происходит обращение, назовем исполнительным адресом Аи.

Прямая адресация — адрес операнда содержится в коде команды и следует за кодом операции. Прямая адресация используется с простыми переменными и константами, положение которых в памяти не меняется в процессе выполнения задачи. При прямой адресации Аи = Ак.

Прямая адресация операндов, размещенных в регистрах МП, имеет специальное название прямая регистровая адресация, или регистровая

адресация. При использовании регистровой адресации в адресном поле команды указывается адрес (код) регистра. Команды, содержащие только регистровые операнды, являются наиболее компактными и быстрыми(не обращаемся к внешней памяти).

Непосредственная адресация позволяет задавать фиксированные значения операнда(Оп) прямо в команде(Оп = Ак). Удобно при работе с константами(адресации на самом деле нет). Но необходимо расширять формат команды для указания Оп.

Неявная адресация - способ адресации, при котором в команде не содержатся явные указания об адресе операнда. При неявной адресации сам код операции определяет адрес операнда. При этом в адресной части команды не указывается, но этот адрес подразумевается. Например, это могут быть аккумулятор, индексный и базовый регистры, указатель стека и некоторые др.

Косвенная адресация – эффективный и важный способ адресации, при котором адрес, указываемый в команде, является указателем ячейки, содержащей исполнительный адрес операнда в памяти. Фактически при косвенной адресации в команде указывается адрес адреса обозначается(Аи=(Ак)). Частным случаем косвенной адресации является регистровая косвенная адресация, при которой исходный адрес в команде адресует общий регистр процессора, содержащий адрес операнда в памяти. С точки зрения затрачиваемых разрядов для представления адреса регистровая косвенная адресация оказывается много эффективнее прямой адресации, так как в этом случае указывается только адрес общего регистра, а он много короче полного адреса операнда в памяти.

Развитием и модификацией метода косвенной адресации является **относительная адресация, или базирование**. Это обобщенное название ряда методов адресации, обеспечивающих вычисление исполнительного адреса Аи операнда в памяти в виде суммы базового значения адреса и «смещения» disp, указываемого в команде.

Поскольку вычисление исполнительного адреса Аи, связано с потерей времени, при определении адреса Аи, операцию суммирования часто заменяют операцией конкатенации (приписывания разрядов).

Базирование (относительная адресация) широко применяется для адресации памяти, представленной в виде блоков фиксированного или произвольного размера. Блоки фиксированного размера называют страницами, а произвольного - сегментами. Соответственно различают память со страничной организацией и сегментированную память. Полная информация, необходимая для определения физического адреса произвольной ячейки памяти с подобной организацией, содержится в указателе адреса, который включает в себя идентификатор базового адреса блока и смещении внутри блока. Для

определения базового адреса блока (сегмента или страницы) используют различные способы идентификации. Чаще всего базовые адреса блоков хранятся в специальных таблицах (сегментных или страничных) и идентификатор в указателе адреса служит индексом (номером) строки такой таблицы. Разрядность базового адреса в общем случае определяет максимальное число адресуемых блоков памяти, а число бит в смещении задает максимальный размер блока. Исполнительный (физический) адрес операнда образуется в результате суммирования базового адреса блока и смещении внутри блока.

Важной особенностью базирования (относительной адресации) является то, что при изменении базовых адресов блоков их содержимое не меняется и блоки можно свободно перемещать в пределах всего адресного пространства памяти. Благодаря этому свойству базирование обеспечивает очень важную функцию ОС — так называемую перемещаемость программ. Перемещаемость программы предполагает неизменяемость адресных ссылок в программе при ее перемещении внутри доступного процессору пространства памяти. Базовые адреса исполняемых программ определяются ОС непосредственно при загрузке программы в ОП.

В системах виртуальной памяти базирование используется для адресации памяти существенно большего размера, чем реальная оперативная память ВМ. Наличие виртуальной памяти обеспечивает возможность решения задач, объем которых превышает объем ОП. Управление обменом информации в системах виртуальной памяти осуществляет ОС.

4. Факторы , влияющие на производительность процессора. Измерение производительности. MIPS и MFLOPS. Смеси команд их использование при оценке производительности.

(Про смеси команд нет ничего)

Частота

Во время своей работы процессор синхронизируется с встроенным тактовым генератором. Это явление формирует такую важную характеристику, как тактовая частота. Обычно тактовая частота показывается (измеряется) в мегагерцах или гигагерцах, и она прямо определяет, насколько быстро процессор обрабатывает информацию. Чем выше будет тактовая частота выбранного вами процессора, тем производительнее он будет. Стоит отметить, что это не единственный параметр, прямо влияющий на скорость процессора, но является одним из самых важных.

Шина данных

Можете себе представить, что внешняя шина данных (front side bus) является неким транспортом, который переносит данные между процессором и другими компонентами системы. Контроллер шины данных регулирует потоки данных, передаваемые между процессором и оперативной памятью, а также различными накопителями данных и графическими устройствами. Процессор передает данные на компоненты, а они затем их возвращают назад. На эти действия уходит значительное количество времени. Вот почему тактовая частота процессора не является единственным параметром, влияющим на производительность. Его высокая скорость может быть нивелирована бутылочным горлышком в виде медленной шины данных. Например, сегодня медленная шина данных передает около 400 мегабайт в секунду, в то время как лучшие образцы легко преодолевают скорость в 3600 мегабайт в секунду.

Кэш-память

Кэш-память является маленьким модулем памяти, встроенным в процессорное ядро. Она состоит из таких же чипов памяти, которые используются при производстве внешних модулей оперативной памяти. Вы можете определить объем и вид кэш-памяти, встроенной в процессор, по модели процессора – эта информация публикуется в спецификации к каждой модели от любого производителя.

Больший размер кэш-памяти увеличивает производительность процессора. Если вы в одном сеансе длительное время используете одну и ту же программу, то ее самые повторяющиеся блоки помещаются процессорам во встроенный кэш. При следующем обращении они будут выгружаться не с жесткого диска, а из кэш-памяти. При этом передача данных произойдет по внутренней шине данных ядра, что произойдет на порядок быстрее, чем через системную шину FSB.

Казалось бы, основной мерой оценки производительности микропроцессора должна выступать его тактовая частота. В то же время с развитием архитектуры, появлением суперскалярных микропроцессоров, возможностей обработки данных по схеме SIMD, оптимизации структуры кэш-памяти тактовая частота становится скорее показателем уровня совершенства технологического процесса, но отнюдь не производительности

MIPS и MFLOPS

С подходом, основанным на сравнении производительности микропроцессоров по их рабочим частотам, тесно связан подход по оценке производительности системы по тому, насколько быстро система может выполнять команды процессора.

Однако это весьма расплывчатый показатель. Скорость работы процессора, обычно выражаемая в миллионах операций в секунду (millions of INsTRuctions per second - MIPS), сильно привязана к его тактовой частоте.
Кроме того, оценка производительности в MIPS существенно зависит от системы команд микропроцессора: одна команда в микропроцессоре одного типа может быть эквивалентна по вычислительной мощности нескольким командам другого МП. К тому же различные операции, особенно в CISC-микропроцессорах, требуют разного времени для их выполнения. Следовательно, MIPS -оценка существенно зависит от того, какие команды принимаются в расчет. Таким образом, MIPS является полезным показателем лишь при сравнении процессоров одного производителя. Такие процессоры должны поддерживать одинаковую систему команд. Кроме того, следует применять одинаковые компиляторы. Существенная слабость MIPS как показателя производительности часто являлась поводом для шутки: MIPS - это аббревиатура выражения "MeanINgless INdicator of Processor Speed" ("бессмысленный показатель скорости процессора").
Аналогичным подходом является измерение производительности работы процессора в миллионах операций с плавающей точкой в секунду (millions of floatINg-poINT operations per second - MFLOPS). Обычно скорость в MFLOPS вычисляют для смеси операций сложений и умножений с плавающей точкой. Но поскольку микропроцессоры становятся все быстрее и быстрее, значение максимума MFLOPS перестает быть полезным в качестве разумной меры производительности операций с плавающей точкой: ограничивающим фактором становится пропускная способность каналов памяти (насколько быстро данные можно перемещать из процессора и в процессор).

5. CISC- и RISC -архитектуры процессоров, их характерные признаки. Особенности RISC -процессоров и их влияние на производительность.

CISC (англ. complex instruction set computing или complex instruction set computer) — тип процессорной архитектуры, которая характеризуется следующим набором свойств:

- · нефиксированное значение длины команды;
- · арифметические действия кодируются в одной команде;
- · небольшое число регистров, каждый из которых выполняет строго определённую функцию.

RISC (англ. reduced instruction set computer — «компьютер с сокращённым набором команд») — архитектура процессора, в котором быстродействие увеличивается за счёт упрощения инструкций, чтобы их декодирование было более простым, а время выполнения — меньшим. Первые RISC-процессоры

даже не имели инструкций умножения и деления. Это также облегчает повышение тактовой частоты и делает более эффективной суперскалярность (распараллеливание инструкций между несколькими исполнительными блоками). В таких микропроцессорах содержится меньшее количество транзисторов, что снижает их стоимость и энергопотребление. При этом, как правило, повышается их производительность. Архитектура RISC является основой современных высокопроизводительных ЭВМ.
Унификация набора команд, ориентация на конвейерную обработку, унификация размера команд и длительности их выполнения, устранение периодов ожидания в конвейере - все эти факторы положительно сказываются на общем быстродействии.
Недостатки RISC прямо связаны с некоторыми преимуществами этой архитектуры. Принципиальный недостаток - сокращенное число команд: на выполнение ряда функций приходится тратить несколько команд вместо одной в CISC. Это удлиняет код программы, увеличивает загрузку памяти и трафик команд между памятью и ЦП. Исследования показали, что RISC-программа в среднем на 30% длиннее CISC-программы, реализующей те же функции.
Характерные особенности RISC-процессоров

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

6. Конвейеризация исполнения команд. Основные этапы конвейерной обработки команд. Промежуточные буферы в конвейере. Синхронный и асинхронный конвейеры

Конвейер — способ организации вычислений, используемый в современных процессорах и контроллерах с целью повышения их производительности (увеличения числа инструкций, выполняемых в единицу времени — эксплуатация параллелизма на уровне инструкций), технология, используемая при разработке компьютеров и других цифровых электронных устройств.
Идея заключается в параллельном выполнении нескольких инструкций процессора. Сложные инструкции процессора представляются в виде последовательности более простых стадий. Вместо выполнения инструкций последовательно (ожидания завершения конца одной инструкции и перехода к следующей), следующая инструкция может выполняться через несколько стадий выполнения после первой инструкции. Это позволяет управляющим цепям процессора получать инструкции со скоростью самой медленной стадии обработки, однако при этом намного быстрее, чем при выполнении эксклюзивной полной обработки каждой инструкции от начала до конца.
На иллюстрации справа показан простой пятиуровневый конвейер в RISC-процессорах. Здесь:

- · IF (англ. Instruction Fetch) — получение инструкции,
- · ID (англ. Instruction Decode) — раскодирование инструкции,
- · EX (англ. Execute) — выполнение,
- · MEM (англ. Memory access) — доступ к памяти,
- · WB (англ. Register write back) — запись в регистр.

Конвейеризация увеличивает пропускную способность процессора (количество команд, завершающихся в единицу времени), но она не сокращает время выполнения отдельной команды. Имеются некоторые накладные расходы на конвейеризацию, возникающие в результате несбалансированности задержки на каждой его ступени. Частота синхронизации (такт синхронизации) не может быть выше, чем время, необходимое для работы наиболее медленной ступени конвейера. Конвейер не всегда представляет собой линейную цепочку этапов. В ряде ситуаций оказывается выгодным, когда функциональные блоки соединены между собой не последовательно, а в соответствии с логикой обработки. Отдельные блоки в цепочке могут пропускаться, а другие — образовывать циклические процедуры. Это позволяет с помощью одного конвейера вычислять более одной функции.

В ходе выполнения команда продвигается по конвейеру, освобождая его очередную ступень для следующей команды. Для хранения информации, предаваемой с одной ступени на другую, используют внутренние промежуточные буферы. Содержимое буферов обновляется в каждом такте по завершению этапа исполнения очередной команды. Промежуточные буферы обеспечивают параллельную независимую работу блоков конвейерной цепочки. В то время, когда последующий блок начинает этап очередной команды, предыдущий блок может приступать к обработки следующей команды, благодаря этому в конвейере одновременно могут обрабатываться несколько следующих друг за другом команд.
В современных процессорах применяют синхронные и асинхронные конвейеры выполнения команд. Если конвейер работает в принудительном темпе и для выполнения любого этапа выделено одно и то же время, каковым считается время самого длительного этапа, то конвейер называется синхронным. Синхронные конвейеры реализованы в ранних версиях процессора Pentium. В асинхронном конвейере отсутствует единый квант времени работы его блоков. В таком конвейере информация с предыдущего блока передается на следующий, сразу же после окончания ее обработки этим блоком, при условии, что следующий блок полностью завершил обработку предыдущей команды.

7. Конфликты при конвейерном исполнении команд. Причины и виды конфликтов.

Ситуации, называемые конфликтами конвейера (англ. hazards), препятствуют выполнению очередной команды из потока команд в предназначенном для неё такте. Конфликты уменьшают реальное ускорение в производительности конвейерной обработки и могут вызвать необходимость остановки конвейера. Для разрешения конфликта нужно, чтобы некоторые команды в конвейере могли продолжать выполняться, в то время как другие были задержаны. Существует три класса конфликтов.
Структурные конфликты
Структурные конфликты возникают из-за конфликтов ресурсов, когда аппаратура не может поддерживать все возможные комбинации одновременно выполняемых команд. Если какая-то комбинация команд не может быть поддержана, то говорят, что процессор имеет структурный конфликт. Наиболее часто структурные конфликты происходят, когда некоторый функциональный блок не полностью конвейеризован. Например, некоторые процессоры совместно используют единый конвейер памяти для данных и команд. В результате, когда команда содержит обращение к памяти данных, она вступает в конфликт с обращением более поздней командой. Чтобы этот конфликт

разрешался при обращении к памяти за данными, конвейер приостанавливается на один такт.
В качестве альтернативы такому структурному конфликту разработчик мог бы обеспечить отдельное обращение к памяти команд либо путём разбиения кэша на отдельные кэш команд и кэш данных, либо используя множество буферов, называемыми буферами команд для хранения команд, однако, этого не делается во избежание увеличения стоимости блока.
Конфликты по данным
Конфликты по данным возникают, когда зависимость команды от результатов предыдущей проявляется при совмещении команд в конвейере. Данные конфликты происходят, когда конвейер изменяет порядок обращений считывания/записи к операндам так, что он отличается от порядка, который существует для последовательно выполняемых команд в процессоре без конвейера. Существует метод устранения конфликта по данным: форвардинг (англ. register forwarding) (иногда называется bypass). К сожалению, не все потенциальные конфликты по данным можно обработать с помощью байпаса, в этом случае конвейер приостанавливается до разрешения конфликта.
Конфликты по управлению
Конфликты по управлению возникают при конвейерном выполнении условных передач управления и других команд, которые изменяют значение программного счетчика. Существует много способов обработки остановов конвейера, вызванных задержкой передачи управления, но для глубоких конвейеров в основном используются агрессивные средства, такие как предсказания передач управления.

8. Способы повышения производительности современных процессоров. Суперскалярная обработка.

Способы повышения производительности:

- · Суперскалярная архитектура
- · Конвейерная обработка
- · Суперконвейер
- · Спекулятивное выполнение и предсказание переходов
- · Динамическое выполнение команд
- · Переименование регистров и буфера записи
- · Многопроцессорность

Суперскалярный процессор (англ. superscalar processor) — процессор, поддерживающий так называемый параллелизм на уровне инструкций (то есть, процессор, способный выполнять несколько инструкций одновременно) за счёт включения в состав его вычислительного ядра нескольких одинаковых функциональных узлов (таких как АЛУ, FPU, умножитель, сдвигающее

устройство и другие устройства). Планирование исполнения потока инструкций осуществляется динамически вычислительным ядром (не статически компилятором).
Использование конвейера, увеличение количества функциональных узлов процессора (суперскалярность), увеличение количества ядер (многоядерность) и увеличение количества процессоров (многопроцессорность) — разные способы увеличения производительности, которые могут использоваться совместно. При использовании конвейера количество узлов остаётся прежним; увеличение производительности достигается за счёт одновременной работы узлов, ответственных за разные стадии обработки инструкций одного потока. При использовании суперскалярности увеличение производительности достигается за счёт одновременной работы большего количества одинаковых узлов, независимо обрабатывающих инструкции одного потока (в том числе, и большего количества конвейеров). При использовании нескольких ядер каждое ядро выполняет инструкции отдельного потока, может быть суперскалярным и/или конвейерным. При использовании нескольких процессоров каждый процессор может быть многоядерным.
В суперскалярном процессоре инструкция извлекается из потока инструкций (находящегося в памяти), определяется наличие или отсутствие зависимости инструкции по данным от других инструкций, затем инструкция выполняется. Одновременно, в течение одного такта, может выполняться несколько независимых инструкций.
Если в процессе работы процессора несколько инструкций, обрабатываемых конвейером, независимы, то ядро может выполнить их одновременно. В суперскалярных системах решение о запуске инструкции на исполнение принимает само ядро процессора, что требует много ресурсов. В более поздних системах, таких, как Эльбрус-3 и Itanium, используется статпланирование, то есть решение о том, какие инструкции выполнять одновременно, принимает компилятор; компилятор находит независимые инструкции и объединяет их в одну длинную инструкцию (архитектура VLIW).
Суперскалярный процессор, обычно, способен выполнять больше одной инструкции за такт. Но способность обрабатывать несколько инструкций одновременно не делает архитектуру суперскалярной, так как одновременности можно добиться и другими методами: применением конвейера, применением нескольких ядер и/или применением нескольких процессоров.

9. Иерархическая организация системы памяти ВМ . Средства для построения устройств памяти на различных уровнях организации.

Память — одна из наиболее важных подсистем ВМ. В первую очередь от памяти зависят функциональные возможности ВМ как средства обработки данных (зависит возможность установки и исполнения того или иного ПО). Организация и характеристики памяти существенно влияют на все общетехнические показатели ВМ: производительность, стоимость и надежность.

Назначение памяти — запись, хранение и чтение информации, используемой в процессе работы ВМ.

Совокупность устройств, обеспечивающих запись, хранение и чтение информации в ВМ, образует *систему памяти.* Она включает: регистры процессора, кэш-память (одного или двух уровней), ОП, внешнюю память (на ЖД и МД, на компакт-дисках с оптическим считыванием, на магнитной ленте), архивную память в виде сменных дисков и кассет.

Кэш-память является буфером между ОП и процессором. Поэтому адреса кэша повторяют соответствующие адреса ОП. Необходимость введения кэш-памяти и основные принципы ее организации были рассмотрены в гл. 4.

Организация информации во внешней памяти построена на принципах, отличных от линейно-адресной организации ОП. Информация структурируется с учетом содержания и соответствующих логических связей. Основной единицей информации здесь является файл. *Файл* — совокупность связанных записей, рассматриваемая как единое целое. *Записью* называют совокупность данных, характеризующую тот или иной объект. По виду информации различают программные файлы и файлы данных. Пользователь определяет имя файла и его логическую организацию. Поиск нужного файла осуществляется не по адресу, а по имени файла.

ОП строится на СБИС полупроводниковых запоминающих устройств. Как указано в табл. 6.1, в основном используются СБИС динамической и статической памяти, выполненные по КМОП- технологии. Особенности схемотехники и организации работы СБИС существенно влияют на структурную организацию устройства ОП. Иерархическая организация ОП имеет следующие уровни: СБИС, модуль памяти (SIMM, DIMM), накопитель, разделенный на банки. На организацию ОП и ее взаимодействие с центральным процессором существенно влияет использование кэшей.

Характеристика	Уровень иерархии			
	Регистр	Кэш-память	ОП	НЖМД
1. Информационная емкость $V_п$, байт	$<10^3$	$<4 \cdot 10^6$	$<4 \cdot 10^9$	$>10^{11}$
2. Время доступа $t_{дост}$, нс	0,5—3	1—3	30—100	$2 \cdot 10^6$
3. Применяемая технолоия изготовления	CMOS*, BiCMOS**	CMOS SRAM***	CMOS DRAM****	Магнитный диск
4. Устройство (программа) управления обменом	МПА МП	Контролер кэша, ОП	ОС	ОС и пользователь

Иерархия компьютерной памяти — концепция построения взаимосвязи классов разных уровней компьютерной памяти на основе иерархической структуры.
Сущность необходимости построения иерархической памяти — необходимость обеспечения вычислительной системы достаточным объёмом памяти, как оперативной так и постоянной.
Учитывая неоднородность периодичности обращения к конкретным записям (внутренним регистрам процессора, кэш-памяти, страницам и файлам) применяются различные технические решения, имеющие отличные характеристики, как технические так ценовые и массо-габаритные.
Долговременное хранение в дорогой сверхоперативной и даже оперативной памяти, как правило, не выгодно, поэтому данные такого рода хранятся на накопителях — дисковых, ленточных, флеш и т.д.

- В большинстве современных ПК используется следующая иерархия памяти:
- · Регистры процессора, организованные в регистровый файл — наиболее быстрый доступ (порядка 1 такта), но размером лишь в несколько сотен или, редко, тысяч байт.
- · Кэш процессора 1го уровня (L1) — время доступа порядка нескольких тактов, размером в десятки килобайт
- · Кэш процессора 2го уровня (L2) — большее время доступа (от 2 до 10 раз медленнее L1), около полумегабайта или более
- · Кэш процессора 3го уровня (L3) — время доступа около сотни тактов, размером в несколько мегабайт (в массовых процессорах используется недавно)
- · ОЗУ системы — время доступа от сотен до, возможно, тысячи тактов, но огромные размеры в несколько гигабайт, вплоть до сотен. Время доступа к ОЗУ может варьироваться для разных его частей в случае комплексов класса NUMA (с неоднородным доступом в память)

- · Дисковое хранилище — многие миллионы тактов, если данные не были закэшированны или забуферизованны заранее, размеры до нескольких терабайт
- · Третичная память — задержки до нескольких секунд или минут, но практически неограниченные объёмы (ленточные библиотеки).

10. Принцип хранения информации в СБИС динамической памяти. Структурная схема СБИС DRAM . Временные диаграммы работы СБИС в режимах чтения, записи, регенерации.

Динамическая память способна запоминать нули и единицы благодаря емкости p/n перехода. Ячейки состоят из конденсаторов и транзисторов, расположенных внутри полупроводниковых микросхем памяти. Конденсаторы заряжают при записи в ячейку единичного бита и разряжают при записи в ячейку нулевого бита.
При прекращении подачи электроэнергии конденсаторы разряжаются, и память обнуляется (опустошается). Для поддержания необходимого напряжения на обкладках конденсаторов (для сохранения данных) конденсаторы необходимо периодически подзаряжать. Подзарядку выполняют путём подачи на конденсаторы напряжения через коммутирующие транзисторные ключи. Необходимость постоянной зарядки конденсаторов (динамическое поддержание заряда конденсаторов) является основополагающим принципом работы памяти типа DRAM.
Важным элементом памяти типа DRAM является чувствительный усилитель-компаратор (англ. sense amp), подключённый к каждому из столбцов «прямоугольника». При чтении данных из памяти усилитель-компаратор реагирует на слабый поток электронов, устремившихся через открытые транзисторы с обкладок конденсаторов, и считывает одну строку целиком. Чтение и запись выполняются построчно; обмен данными с отдельно взятой ячейкой невозможен. В DRAM каждую ячейку можно отыскать по ее адресным координатам, оформленным в строки и столбцы. Все ячейки выводятся на общую числовую шину. Выбор соответствующего адреса строки и столбца позволяет определить место ячейки. Содержимое нескольких ячеек, объединенных на выходе, образует информационную группу — байт, или слово, и следует на шину данных памяти.
Запоминающие ячейки микросхем DRAM организованы в виде двухмерной матрицы. Адреса строки и столбца передаются по мультиплексированной шине адреса (MA - Multiplexed Address) и стробируются по спаду импульсов RAS (

Row Access Strobe) и CAS (Column Access Strobe). Временная диаграмма классических циклов записи и чтения приведена на рис. 1.

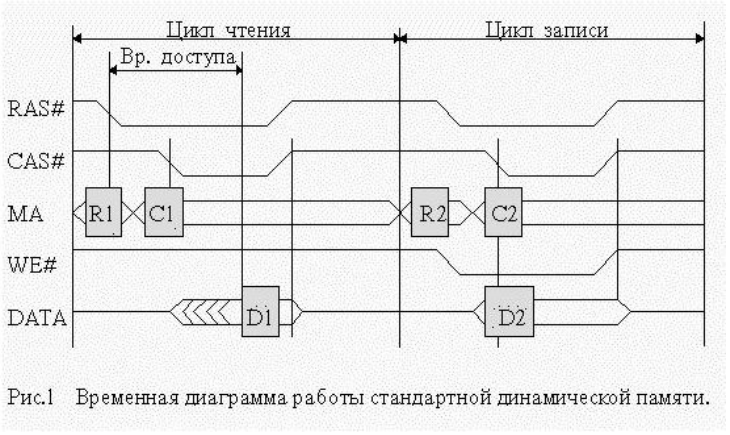


Рис.1 Временная диаграмма работы стандартной динамической памяти.

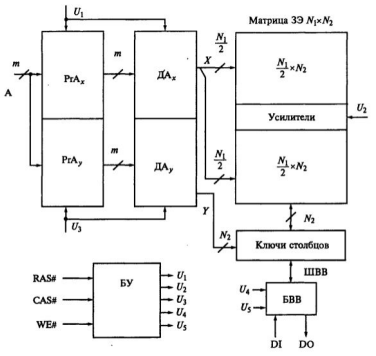


Рис. 6.2. Структурная схема СБИС DRAM:

11. Организации взаимодействия процессора с основной и внешней памятью . Линейно-адресная организация ОП. Физическая структура данных во внешней памяти (в ВЗУ).

Вычислительный процесс в ВМ организует процессор. При обработке данных процессор обменивается данными с памятью или периферийными устройствами по адресу (соответственно используются два АП). При этом используется принцип линейно-адресной организации этих пространств. Адрес является именем информации, записываемым в программе.

Во ВЗУ, предназначенных для долговременного хранения информации, таких адресов у запоминающих ячеек нет. Здесь используется иной способ логической организации информации. Информация структурируется с учетом содержания и соответствующих логических связей. Основной единицей долговременного хранения информации является файл (см. подразд. 6.1). Информация на внешних носителях имеет определенную физическую структуру, определяемую организацией ВЗУ. Между физическими файлами и логическими нет однозначного соответствия. Это соответствие организуется с использованием ОС. В целях упрощения программирования в ОС вводят два уровня программного управления вводом-выводом данных: логический и физический. Логический уровень содержит ряд логических модулей, каждый из которых относится к определенному периферийному устройству. Программы пользователей имеют дело с логическими модулями. Логический модуль формирует входные данные для периферийного устройства в стандартном для него виде. Передача информационных и служебных сигналов осуществляется с использованием соответствующих интерфейсных устройств — контроллеров.

Для того чтобы информация, хранимая в ВЗУ, стала доступной процессору, ее необходимо переписывать в ОП. Управляют процессами обмена ОС, чипсеты и контроллеры внешних устройств

Для знакомства с физической структурой данных в ВЗУ рассмотрим ее на примере гибкого магнитного диска (ГМД или ГД). Данные ГД (рис. 6.31) хранятся на концентрических дорожках в последовательной форме. Дорожки нумеруются от периферии диска к центру. Диск разделен на секторы. Таким

образом, информация на диске разделена на блоки. Блок идентифицируется номером дорожки и номером сектора. Отверстие в центре диска используется для соединения с приводом. Индексное отверстие служит для синхронизации вращения диска и работы электронных схем, участвующих в чтении и записи. Внешняя дорожка с номером «00» содержит идентифицирующую информацию для всего диска. Следующие две дорожки обычно являются запасными, а остальные используются для записи рабочей информации. Разделение поля записи на секторы осуществляется магнитными маркерами. Положение магнитных маркеров определяется программно в процессе разметки дискеты. Маркер представляет собой байт с особым набором сигналов D и C. Маркеры выделяются схемой контроллера ГД и используются для управления процессом обмена.
Разделение поля записи на секторы осуществляется магнитными маркерами. Положение магнитных маркеров определяется программно в процессе разметки дискеты. Маркер представляет собой байт с особым набором сигналов D и C. Маркеры выделяются схемой контроллера ГД и используются для управления процессом обмена.
Каждый сектор имеет поле признаков и поле данных. Для повышения надежности хранения информации используют циклические коды Основным типом ВЗУ в компьютере являются *накопители на жестких магнитных дисках* (HDD — Hard Disk Drive) — ЖД. В ЖД магнитная пленка наносится с двух сторон на алюминиевые диски. Диски собираются в пакет, жестко связанный с двигателем. Магнитные свойства поверхности диска, конструкция и высота полета головок, скорость движения определяют предельную плотность расположения ЗЭ.

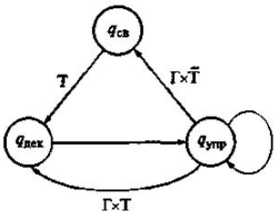
12. Организация кэш -памяти. Принципы организации кэш -памяти прямого отображения и частично-ассоциативной кэш -памяти. Основные преимущества и недостатки.

Кэш-память (КП), или кэш, представляет собой организованную в виде ассоциативного запоминающего устройства (АЗУ) быстродействующую буферную память ограниченного объема, которая располагается между регистрами процессора и относительно медленной основной памятью и хранит наиболее часто используемую информацию совместно с ее признаками (тегами), в качестве которых выступает часть адресного кода.
В процессе работы отдельные блоки информации копируются из основной памяти в кэш-память. При обращении процессора за командой или данными сначала проверяется их наличие в КП. Если необходимая информация находится в кэше, она быстро извлекается. Это кэш-попадание. Если необходимая информация в КП отсутствует (кэш-промах), то она выбирается из основной памяти, передается в микропроцессор и одновременно заносится в кэш-память. Повышение быстродействия вычислительной системы

достигается в том случае, когда кэш-попадания реализуются намного чаще, чем кэш-промахи.
Самой простой организацией обладает КЭШ память с прямым отображением. В этом случае адрес памяти полностью определяет используемую строку КЭШ. Для КЭШ памяти с прямым отображением этот адрес разделяется на 3 части: ТЕГ, Номер строки, Номер байта (смещение)
Младшая часть байта определяет порядковый номер байта в строке КЭШа и является смещением.
Среднее поле позволяет однозначно выбрать одну строку КЭШа, это поле «номер строки». Оставшиеся старшие разряды несут информацию о признаке, теге.
Если значение тега строки КЭШа совпадает со значением тега из адреса, то происходит чтение байта информации из КЭШа. Такая ситуация квалифицируется как КЭШ-попадание.
Если эти теги не совпадают, то это означает, что искомый байт отсутствует в КЭШе и для его чтения (байта) следует обращаться к ОЗУ. Эта ситуация квалифицируется как КЭШ-промах.
Одновременно с чтением байта из ОЗУ происходит запись в КЭШ строки из ОЗУ, которая содержит нужный байт. Эта строка, прочитанная из ОЗУ, помещается на место той строки, к которой только что было неудачное обращение.
При этом в КЭШ для данной строки записывается и новое значение тега.
Достоинством такой организации КЭШ-памяти является ее простая реализация, поскольку требуется проводить всего одну операцию сравнения тегов. При этом сама КЭШ-память является обычной адресной памятью.
Однако такая КЭШ память имеет и существенный недостаток, который заключается в том, что происходят частые обращения к ОЗУ, если две строки данных, претендующих на одну и ту же строку КЭШа, используются одинаково часто и поэтому происходит частая их запись в КЭШ.
Частично ассоциативная организация кэш память.
В этом случае несколько строк кэша объединяются в наборы, к которым происходит обращение с помощью средних бит адреса (поле набора). Сравнение тегов КЭШа с тегом адреса происходит только для строк, входящих в выбранный набор. При совпадении тега адреса с одним из тегов набора происходит чтение байта из КЭШа (КЭШ попадание). Если такого совпадения нет (КЭШ промах), то происходит обращение к ОЗУ.

13. Организация обмена данными между основной памятью и процессором по системной шине. Принцип управления циклом шины. Влияние показателей быстродействия СБИС DRAM.

Быстродействие СБИС DRAM увеличивается существенно медленнее, чем быстродействие процессоров. Это противоречие потребовало новых архитектурных решений. Важнейшим архитектурным усовершенствованием для сглаживания этого противоречия явилось введение в структуру ВМ кэш-памяти . При обмене данными процессора с ОП по СШ используется принцип квитирования. Взаимодействие по шине (регистровая пересылка между одним из регистров ОБ процессора и ячейкой ОП либо портом (регистром) периферийного устройства) осуществляется за цикл шины. Длительность цикла шины может изменяться в зависимости от быстродействия внешних устройств (памяти и устройств ввода-вывода). Для пояснения принципов организации циклов шины рассмотрим граф переходов автомата, управляющего обменом по СШ. Приведенный граф представлен в упрощенном виде и характеризует только функцию переходов автомата. Автомат имеет три внутренних состояния q - свободное, декодирования типа обмена, управления. Автомат находится в свободном состоянии, когда процессор не требует обмена по шине. В состоянии декодирования расшифровывается тип обмена (тип цикла шины). Пример различных типов: ввод из памяти, вывод в память, ввод от устройства ввода, вывод на устройство вывода, подтверждение прерывания, останов и т.д. В соответствии с определенным типом далее в состоянии управления формируются управляющие сигналы в шину управления. Условия переходов содержат два сигнала: Г — готовность, Т — требование цикла шины. Сигнал Г поступает в процессор (и в автомат управления обменом по шине) через шину управления от устройства на шине. Сигнал Т поступает от блока управления процессора, когда для выполнения команды требуется обмен по шине.



Автомат, управляющий обменом по СШ, для завершения цикла обмена ожидает от пассивного устройства (памяти) сигнала готовности (квитанции). Этот сигнал формируется устройством памяти в соответствии с его быстродействием. Работа УА синхронизирована тактовыми импульсами с частотой, принятой в ВМ для СШ. Ожидая сигнал готовности, автомат включает

в цикл шины такты ожидания. В соответствии с такой организацией быстродействие памяти по отношению к быстродействию процессора характеризуют числом тактов ожидания в цикле обращения к памяти. Внешний кэш, вынесенный за пределы кристалла процессора, обычно реализуют на основе СБИС статической памяти и располагают в непосредственной близости от процессора. Чем выше быстродействие СБИС памяти, тем выше стоимость. Рассмотрим функциональную схему, поясняющую организацию информационного взаимодействия центрального процессора с ОП в современном компьютере, когда используется 2-уровневая кэш-память (рис. 6.13). Из рисунка видно, что кэш-память включена параллельно ОП и служит промежуточным буфером. Потребителями данных, извлекаемых из ОП, являются ОУ и блок управления в составе процессора. При этом информационный обмен этих блоков может осуществляться как с ОП (аналогично тому, как показано на рис. 6.12), так и с кэш-памятью 1-го либо 2-го уровня.

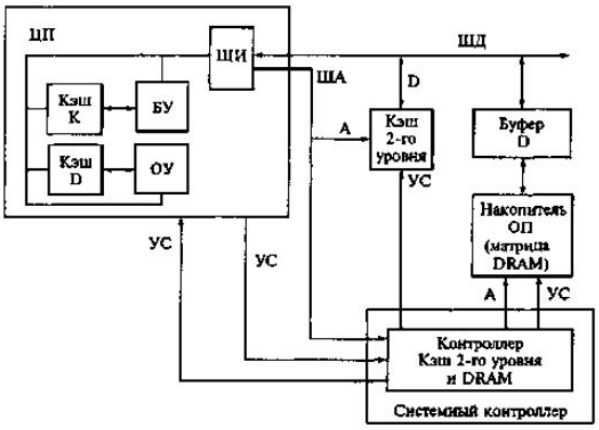


Рис. 6.13. Схема взаимодействия ЦП и ОП (с кэшированием): ЦП — центральный процессор; БУ — блок усиления; ОУ — операционное устройство; ША — шина адресов; ШД — шина данных; УС — управляющие сигналы

14. Динамическое распределение памяти. Виртуальная память. Основные модели виртуальной памяти: модель сегментированной памяти и модель памяти со страничной организацией.

Динамическое распределение памяти ОП, адресуемая процессором, имеет сравнительно небольшой объём и не может вместить все команды и данные исполняемых программ. Все они хранятся в ВЗУ, но процессор не имеет к ним прямого доступа. Поэтому они переписываются в ОП, однако они все не могут быть помещены физически в ограниченном объёме ОП, поэтому распределение ОП

между вызываемыми программами идёт динамически. Процессом динамического распределения занимается ОС, которая активным частям программ выделяет определённые области ОП и делает привязку адресов загружаемых программ к конкретным адресам физической ОП. Это называется динамическим распределением памяти. В основе этого метода лежит два положения: 1) Каждому заданию необходимо выделять непрерывную и перемещаемую область памяти 2) Должна быть возможность попеременной загрузки заданий в ОП. В целом ОС выполняет следующее: 1) Размещение программ и их частей в уст-вах разного типа – в ОП и ВЗУ 2) Выделяет каждой задаче определённые области ОП и осуществляет настройку адресов программы на конкретные области ОП 3) перераспределяет ОП, когда та не может вместить все 4) освобождает ОП по завершению задачи Виртуальная память – совокупность программно-аппаратных средств представления одноуровневой памяти, которую использует пользователь при подготовке своих программ. Используя логические адреса, пользователь отходит от реального распределения памяти и не учитывает ни объём реальной физической памяти, ни объёмы памяти других программ, выполняемых параллельно с его программой. Систему вирт. памяти можно представить в виде одноуровневой логической или 2-уровневой (ОП + ВЗУ) физической памяти. Адреса, к которым программа может обратиться, образуют виртуальное АП (адресное пространство) системы, а реальные адреса ОП – физическое АП. Размер виртуального АП много большего пространства физических адресов ОП. Все копии исполняемых программ лежат в ВЗУ, если надо – программа переписывается из ВЗУ в ОП и исполняется. Если в ОП не хватает места, то сначала в ВЗУ идёт неисполняемая на данный момент программа, а на её место идёт запрашиваемая. Наиболее известные модели вирт. памяти – память со страничной организацией и сегментированная модель. Обе модели подразумевают разбиение физ. и лог. АП памяти на блоки с линейным способом адресации операндов внутри блока.

15. Организация и способы обмена данными между вычислительным ядром системы и периферийными устройствами. Синхронизация процессов в центральном процессоре и периферийных устройствах.

Проблемы организации обмена данными между периферийными устройствами и вычислительным ядром (процессором и ОП) связаны с асинхронным характером процессов, обусловленных случайным характером событий,

инициирующих начало и конец обмена. В ВС используют три способа организации обмена, каждый из которых по-разному решает отмеченные проблемы: программно-управляемая передача, инициируемая процессором, передача информации прерыванием программы, активизируемая по запросу прерывания от периферийного устройства, и передача информации в режиме прямого доступа к памяти. При программно-управляемой передаче и передачах данных с прерыванием программы обмен осуществляется под управлением процессора. Операции ввода-вывода при таких обменах инициируются либо текущей командой программы, либо запросом прерывания от периферийного устройства. В режиме прямого доступа к памяти передача информации выполняется без участия процессора под управлением специализированного управляющего блока — контроллера прямого доступа к памяти. Программно-управляемый обмен бывает синхронным и асинхронным. Инициатором обмена всегда выступает процессор, реализующий требуемые операции ввода-вывода с помощью соответствующих команд. Синхронная передача применяется при взаимодействии с быстродействующими периферийными устройствами, для обмена с которыми не требуется дополнительной синхронизации (такие устройства ввода-вывода всегда готовы к обмену информацией). Этот способ передачи реализуется при минимальных затратах аппаратных и программных средств. Асинхронный обмен является более универсальным и более сложным способом программно-управляемого обмена. Он используется при работе с периферийными устройствами, быстродействие которых ниже быстродействия процессора. В некоторые моменты времени такие периферийные устройства могут оказаться не готовыми к обмену. Поэтому для выполнения программно-управляемого обмена в общем случае необходимо использовать специальные средства, синхронизирующие процесс приема-передачи. Эти средства содержатся в адаптере (контроллере) периферийных устройств. Адаптер (контроллер) подключается к шине ВМ и служит посредником между периферийными устройствами и вычислительным ядром. Программно-управляемая передача является идеальным (самым быстрым) способом обмена данными между периферийным устройством и процессором. Недостатком такого способа обмена являются вынужденные непроизводительные затраты времени процессора на ожидание готовности периферийного устройства к обмену. Более серьезные последствия при таком способе обмена возникают в ситуациях, когда по каким-либо причинам (например, из-за возникшей неисправности) сигнал готовности вообще не может быть сформирован периферийным устройством. В такой ситуации процессор не сможет выйти из режима ожидания, и его работа будет заблокирована. Для исключения подобных ситуаций в современных вычислительных системах используют специальный программный прием, называемый тайм-аутом. Тип обмена данными, при котором для выполнения операций ввода-вывода производят

прерывание программы, называется передача данных с прерыванием программы. Важным отличием обмена данными с прерыванием программы от синхронного и асинхронного обменов является то, что в нем инициатором обмена является не процессор, а внешнее устройство, запросившее обмен. Возможность прерывания программ — важное архитектурное свойство ВМ, позволяющее повышать производительность процессора при наличии нескольких протекающих параллельно во времени процессов, требующих в произвольные моменты времени управления и обслуживания со стороны процессора. Прямым доступом к памяти (ПДП) называют способ обмена данными, обеспечивающий установление связи и передачу данных между оперативной памятью и периферийным устройством автономно от центрального процессора. При наличии кэш-памяти (внутренней или внешней) режим ПДП позволяет осуществлять обмен данными между ОП и периферийным устройством параллельно с выполнением процессором основной программы. Использование режима ПДП разгружает процессор от обслуживания операций ввода-вывода и способствует повышению общей производительности ВМ. Прямой доступ к памяти требует более сложного интерфейса, по сравнению с программно-управляемой передачей. Для управления обменом данными в режиме ПДП применяют специальные управляющие контроллеры, которые в зависимости от вариантов аппаратной реализации подразделяются на специализированные процессоры ввода-вывода и контроллеры ПДП. В отличие от других способов обмена данными в режиме ПДП управляющий контроллер после получения запроса на организацию передачи от периферийного устройства принимает на себя функции управления системной шиной, при этом сам процессор путем перевода своих тристабильных выходных буферов шины в состояние высокого сопротивления отключается от системной шины.

16. Назначение и организация прерываний. Стандартная последовательность действий при обработке запросов прерываний. Назначение и функционирование программируемого контроллера прерываний.

Прерывание - это прекращение выполнения текущей команды или текущей последовательности команд для обработки некоторого события специальной программой - обработчиком прерывания, с последующим возвратом к выполнению прерванной программы. Событие может быть вызвано особой ситуацией, сложившейся при выполнении программы, или сигналом от внешнего устройства. Прерывание используется для быстрой реакции процессора на особые ситуации, возникающие при выполнении программы и взаимодействии с внешними устройствами. Механизм прерывания

обеспечивается соответствующими аппаратно-программными средствами компьютера.
Назначение: Любая особая ситуация, вызывающая прерывание, сопровождается сигналом, называемым запросом прерывания (ЗП). Запросы прерываний от внешних устройств поступают в процессор по специальным линиям, а запросы, возникающие в процессе выполнения программы, поступают непосредственно изнутри микропроцессора.
Аппаратные прерывания используются для организации взаимодействия с внешними устройствами:

- · маскируемые, которые могут быть замаскированы программными средствами компьютера;
- · немаскируемые, запрос от которых таким образом замаскирован быть не может.
- Программные прерывания вызываются следующими ситуациями:
- · (переполнение, нарушение защиты памяти, отсутствие нужной страницы в оперативной памяти и т.п.);
- · специальной команды прерывания INT n, используемой обычно программистом при обращениях к специальным функциям операционной системы для ввода-вывода информации.

Программируемый контроллер прерываний (Programmable Interrupt Controller, PIC) отвечает за приём запросов прерываний от различных устройств, их хранение в ожидании обработки, выделение наиболее приоритетного из одновременно присутствующих запросов и выдачу его вектора в процессор, когда последний пожелает обработать прерывание.
Схему передачи управления при прерывании можно отобразить следующим образом.
1. при поступлении прерывания производится идентификация устройства которое его запросило;
2. запоминается информация о состоянии процессора;
3. выполняется инициализация процессора для выполнения программы обработки прерывания;
4. производится запуск и исполнение программы обработки прерывания;
5. восстанавливается состояние процессора и возобновляется работа прерванной программы.
Процесс сохранения текущего состояния на момент прерывания и его последующее восстановление называется контекстным переключением. Под состоянием процессора понимается содержимое счетчика (указателя) команд и всех его регистров на момент прерывания.
При поступлении прерывания оценивается его приоритет. Если приоритет выполняемой программы выше, чем у прерывания, то обработка прерывания будет отложена до окончания выполнения программы (выполняемой программой может быть и обработка другого прерывания). Также при

поступлении одновременно нескольких прерываний на обработку должно быть отправлено прерывание с наибольшим приоритетом. Таким образом, необходимо более подробно рассмотреть вопрос оценки приоритета прерываний.

17. Мультипрограммный режим работы компьютера и его основные особенности.

Многозадачностью (мультипрограммным режимом работы) называют такой способ организации работы системы, при которой в ее памяти одновременно содержатся программы и данные для выполнения нескольких процессов обработки информации (задач). В этом режиме должна обеспечиваться взаимная защита программ и данных, относящихся к различным задачам, а также возможность перехода от выполнения одной задачи к другой (переключение задач). Важными понятиями, характеризующими особенности ОС, являются мультипрограммирование и многозадачность. Рассмотрим эти понятия на примере организации мультипрограммного режима работы компьютера. Мультипрограммный режим предполагает одновременное выполнение процессором нескольких программ. Одновременность в данном контексте понимается для интервала времени, соответствующего совместному выполнению программ. Естественно, в каждый конкретный момент времени процессор может выполнять только команды определенной программы. Многозадачностью называется способ организации работы компьютера, при котором в его оперативной памяти содержатся программы и данные для одновременного выполнения нескольких процессов обработки информации (задач). Многозадачность предназначена для повышения «пропускной способности» компьютера путем более равномерной и плотной загрузкой всего его оборудования, в первую очередь процессора. В многозадачном режиме работы компьютера каждая программа может находиться в одном из трех состояний: активном (программа обрабатывается процессором), готовности к обработке и ожидания некоторого события, например завершения операции ввода-вывода или освобождения нужного ресурса. Один из способов реализации мультизадачности, называемый разделением времени, заключается в предоставлении каждой задаче некоторого интервала времени (кванта обслуживания), в течение которого процессор выполняет команды соответствующей программы. Задачи обслуживаются последовательно. Если в течение выделенного программе кванта времени ее обработка не заканчивается, программа прерывается и она становится в очередь программ, ожидающих обработку. ОС, которые используют разделение времени, называют системами с разделением времени. В многозадачных системах каждая задача представлена собственной локальной программой и может

использовать фрагменты глобальных программ и данных, общие для нескольких задач. Многозадачная ОС, управляющая работой процессора, составляет расписание задач, обеспечивает их выполнение на основе устанавливаемых уровней приоритетов и для каждой задачи моделирует собственный виртуальный процессор. Виртуализация — это имитация работы аппаратного ресурса программными средствами. Виртуальный процессор находится в монопольном использовании для задачи. Выполнение отдельных задач с помощью виртуального процессора создает иллюзию, что все задачи выполняются непрерывно параллельно с другими. Однако это не так, поскольк в однопроцессорной системе различные задачи могут выполняться одним процессором только поочередно. Для поддержания впечатления, что каждая задача имеет свой собственный процессор, ОС просто часто переключает реальный процессор на виртуальные, выполняющие собственные задачи. Быстрое переключение процессора на решение различных задач является одной из важнейших проблем организации многозадачной работы. Современные МП содержат эффективные аппаратные средства поддержки многозадачного режима. В частности, процессор Pentium ассоциирует с каждой задачей специальный сегмент данных, содержащий всю информацию, необходимую для запуска и останова задачи. Этот специальный сегмент называется сегментом состояния задачи TSS. Содержимое этого сегмента фактически определяет состояние виртуального процессора задачи. Аппаратная реализация механизма запоминания и восстановления состояния виртуального процессора в сегменте TSS обеспечивает быстрое переключение задач.

18. Типовая структурная схема микроконтроллера. Состав и назначение функциональных устройств. Основные особенности архитектур современных МК.

Микроконтроллером называется программируемое однокристальное вычислительное устройство с встроенным набором средств для ввода и вывода, применяемое для решения задач управления и первичной обработки данных в технических системах. Микроконтроллер - это разновидность микропроцессорной системы, ориентированной на реализацию алгоритмов управления техническими устройствами и технологическим оборудованием.

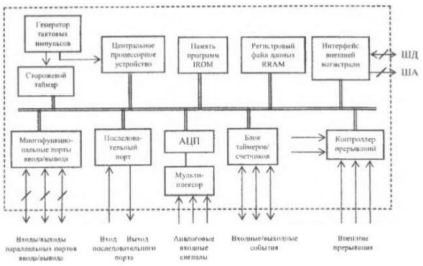


Рис. 1.3. Функциональная схема МК

Для связи МК с объектом управления (датчиками и исполнительными устройствами) в структуре МК обычно содержится широкий спектр специальных средств, обеспечивающих ввод-вывод как дискретных, так и аналоговых сигналов. В число таких средств входят устройства параллельного и последовательного ввода-вывода дискретных сигналов, а также устройства ввода-вывода аналоговых сигналов. Встроенные средства ввода и вывода аналоговых сигналов, размещенные на кристалле МК, обеспечивают связь МК с аналоговыми датчиками и исполнительными устройствами объекта управления, С их помощью входные аналоговые сигналы от датчиков преобразуются в цифровой код для последующей обработки процессором, а также производится формирование аналоговых управляющих воздействий. Для преобразования аналоговых сигналов в цифровой код используются аналого-цифровые преобразователи АЦП. Задачи управления различным оборудованием в реальном времени предопределили обязательное наличие и использование в МК таймеров/счетчиков. С их помощью осуществляется не только счет внешних событий, но и существенно упрощается формирование типовых управляющих сигналов в функции времени. В большинстве случаев таймеры обеспечивают реализацию следующих функций:
•деление внешней или внутренней частоты; •счет событий; •генерация импульсов заданной длительности с программным и/или аппаратным запуском; • регистрация и генерация событий; • функции сторожевого таймера. Практика применения МК диктовала необходимость использования таймеров и в качестве часов «реального времени». Блок быстрого ввода/вывода HSIO (High Speed Input-Output), реализованный фирмой Intel в МК 8xC51FX, стал первым устройством, предназначенным для регистрации входных и генерации выходных событий в реальном времени. Блоки быстрого ввода/вывода. Быстрый ввод заключается в обнаружении входного события- сигнала заданного (программируемого) вида и запоминании времени его наступления в заданной системе отсчета времени. В режиме высокоскоростного вывода блок HSIO осуществляет сравнение текущего времени, формируемого таймером, с заданным в регистре CCL/CCH. Сторожевой таймер WDT. Таймер WDT предназначен для борьбы с программными сбоями системы. Аппаратные средства контроля и защиты. WDT, схема обнаружения падения частоты

тактовых импульсов, защита памяти. Встроенная система прерываний. Реакция процессора на запросы прерывания от любого источника идентична: в конце выполнения каждой команды процессор опрашивает рег истр запросов и выбирает наиболее приоритетный из поступивших. Управление потребляемой мощностью В МК имеется два режима пониженного энергопотребления - режим холостого хода Idle и режим микропотребления.

19. Многоуровневая организация вычислительных процессов. Методы и средства взаимодействия между уровнями. Компиляция и интерпретация. Понятие архитектуры ВМ .

На концептуальном уровне пользователь анализирует задачу, выбирает метод её решения, разрабатывает алгоритм, определяет структуры данных. Затем пишется программа на одном из языков высокого уровня, которая не зависит от архитектуры вычислительной системы и особенностей аппаратного обеспечения. На уровне машинных команд обеспечивается связь программных и аппаратных средств: разрабатывается список команд, определяются способы кодирования операций и адресов, а также другие параметры, заложенные в структуру вычислительной машины. Связь между языками высокого уровня и машинными командами может осуществляться как методом компиляции, так и методом интерпретации. На уровне регистровых передач осуществляется микрооперации, выполняемые аппаратурой вычислительной машины. Это операции передач, запоминания и преобразования кодов, выполняемые пересылкой сигналов между регистрами через логические схемы. Для построения схем на выполнение требуемой микрооперации формируется набор управляющих сигналов – микрокоманда. Последовательность микрокоманд, соответствующая исполнению машинной команды, называется микропрограммой. На уровне логических вентилей, рассматриваются логические схемы, которые выполняют операции над двоичными переменными.

В управлении вычислительным процессом с иерархической организацией участвуют как программные, так и аппаратные средства. По мере развития архитектуры вычислительных машин и технологий их изготовления грань разделения функций, реализуемых аппаратно и программно, смещается в сторону аппаратной реализации.

Самые сложные преобразования информации, выполняемые вычислительной машиной (системой), в конечном счёте, сводятся к простейшим операциям над логическими переменными «0» и «1».

Связь между языками высокого уровня и машинными командами может осуществляться как методом компиляции, так и методом интерпретации. При компиляции программа на языке высокого уровня сначала преобразуется полностью в программу в машинных кодах, а затем может поступать для исполнения с использованием более низких уровней вычислительного процесса. При работе компилятора программа пользователя на языке высокого уровня – это входные данные, а соответствующая программа на языке машинных команд – выходные данные, результат работы компилятора. При исполнении полученная программа в машинных кодах уже используется не как данные, а как программа, управляющая вычислительным процессом.
При интерпретации программа пользователя на языке высокого уровня непосредственно используется для управления вычислительным процессом с использованием системной программы – интерпретатора. Интерпретатор выбирает очередную команду программы на языке высокого уровня, заменяет ее последовательностью машинных команд, выполняющих требуемые функции, и сразу передает эту последовательность машинных команд на исполнение. Затем с учетом признаков, характеризующих результат выполненных преобразований, выбирается следующая команда программы на языке высокого уровня и т.д.

При компиляции требуется более сложная системная программа – компилятор. В процессе компиляции требуется больший объем памяти, но зато исполнение пользовательской программы происходит существенно быстрее. В связи с этим в современных ВМ в основном используется метод компиляции.

На уровне регистровых передач осуществляются микрооперации, выполняемые аппаратурой ВМ. Это операции передачи, запоминания и преобразования кодов, выполняемые пересылкой сигналов между регистрами через комбинационные (логические) схемы.
На уровне логических вентилей рассматриваются логические схемы при логическом проектировании аппаратуры ВМ. Если на уровне регистровых передач рассматриваются операции с n-разрядными кодами, то на уровне вентилей – с отдельными двоичными переменными.
Архитектура вычислительной машины (Computer architecture) - концептуальная структура вычислительной машины, определяющая проведение обработки информации и включающая методы преобразования информации в данные и принципы взаимодействия технических средств и программного обеспечения (ГОСТ 15971-90).
Архитектура вычислительной машины - логическая структура и функциональные характеристики ВМ, включая взаимосвязи между ее аппаратными и программными компонентами

20. Способы подключения периферийных устройств к системной шине. Внешние интерфейсы вычислительных машин.

Система шин компьютера позволяет любому компоненту системы и, что особенно важно, новому компоненту взаимодействовать с любым другим компонентом. Контроллеры (адаптеры) основных периферийных устройств компьютера размещаются на системной плате, и проблемы их подключения к шинам практически не существует. Сложнее с дополнительными внешними устройствами, большинство из которых размещаются вне системного блока. Их подключение к шине, даже если они снабжены стандартными адаптерами (контроллерами), связано с определенными трудностями и не всегда обходится без ошибок. Для подключения каждого нового стандартного устройства необходимо открыть системный блок компьютера, вставить плату контроллера в свободный слот ISA или PCI (в зависимости от типа контроллера), закрыть системный блок и включить компьютер.

Процесс подключения дополнительных периферийных устройств к СШ существенно упрощается при использовании коммуникационных портов. Такими портами в ПК являются параллельные LPT-порты и последовательные COM-порты, снабженные выходными разъемами, установленными непосредственно на корпусе системного блока. Коммуникационные порты обеспечивают сопряжение компьютерами с различными дополнительными периферийными устройствами без вскрытия системного блока и установки плат расширения. Сами коммуникационные порты либо являются элементами системной платы, либо размещаются на платах расширения.

В общем случае подключение периферийных устройств к СШ осуществляется с помощью внешних интерфейсов общего пользования, таких как параллельный интерфейс Centronics, последовательные интерфейсы типа RS-232 и др. При расширении функциональных возможностей ПК для различных пользовательских применений число периферийных устройств, как правило, увеличивается. При этом не только усложняется их подключение к системному блоку, но и возрастает стоимость таких подключений из-за дороговизны расширения стандартных средств подключения. Решением проблемы расширения пространства периферийных устройств явилось использование универсальной последовательной шины — Universal Serial Bus (USB).

Параллельный LPT-порт был включен в состав первых ПК IBM PC для подключения принтеров. Отсюда название LPT-порта (от Line Printer). Подключение к параллельному порту осуществляется через 25-контактный разъем на корпусе ПК. Благодаря удобству программирования и простоте подключения к нему (не требуется вскрывать системный блок), LPT-порт широко применяется для сопряжения с компьютером разнообразных периферийных устройств. В качестве недостатков интерфейса считается невысокое быстродействие и ограничения на протяженность линий связи (до 2 м). Скорость передачи данных через параллельный порт выше, чем через последовательный.

Последовательная передача данных представляет собой реализацию трех последовательных процессов: преобразования параллельных данных источника информации в последовательный формат, передачи последовательной посылки по каналу связи и последующего преобразования приемником принятой посылки в параллельный формат. Под каналом передачи данных здесь и далее понимается физическая аппаратура связи (физический интерфейс) и протокол обмена данными. Последовательная передача характеризуется более низкой скоростью передачи, чем параллельная. Однако она обеспечивает связь на большие расстояния, и для ее реализации требуется меньшее число линий в физическом канале связи, что снижает стоимость. Вместо 3 —5 м при параллельных обменах при последовательной передаче обеспечивается надежная связь на расстояния более 15 м.

Список интерфейсов:

- RS-232
- RS-485
- RS-422
- Интерфейс последовательных мультиплексных каналов MIL-1553
- Универсальная последовательная шина USB