1. Понятие вычислительной машины (компьютера).

ЭВМ - искуственная (инженерная) машина, предназначенная для вычислений на основе алгоритмов.

Принципы построения ЭВМ, с одной стороны, определены назначением ЭВМ и, с другой стороны, элементной базой (набором элементов, которые используются для создания ЭВМ).

Основным назначением ЭВМ является выполнение вычислений на основе алгоритмов, и поэтому свойства алгоритмов предопределяют принципы построения ЭВМ или, точнее, ее архитектуру (организацию).

2. Понятие архитектуры и организации ЭВМ. Виды архитектур.

Под архитектурой ЭВМ обычно понимается ее представление и описание возможностей с точки зрения пользователя, разрабатывающего программу на машинно-ориентированном языке.

Структурная организация ЭВМ определяет как устроена ЭВМ, т.е. задает ее структуру на уровне устройств, входящих в состав ЭВМ, и организацию связей между этими устройствами на уровне аппаратных интерфейсов.

Функциональная организация определяет, в свою очередь, принципы функционирования ЭВМ, т.е. как в ней протекают вычислительные процессы при решении различных задач.

Архитектуру ЭВМ подразделяют на 2 класса:

- программная (прикладная и системная) архитектура, которая включает в себя аспекты, видимые программистом и, собственно, программам.
- аппаратная архитектура, включающая аспекты, невидимые для программиста

3. Основные элементы архитектуры компьютера.

Программная архитектура:

- прикладная: типы и форматы данных, программная модель процессора, адресная структура памяти, режимы адресации, система команд, структура и формат машинных команд.
- системная: организация прерываний, организация ввода/вывода, организация виртуальной памяти, организация защиты памяти и т.д.

Аппаратная (структурная) организация:

- организация устройств: процессор (регистры, АЛУ, FPU, конвейер команд, устройство управления), память (основная, cash, внешняя), устройства ввода/вывода (организация обмена с памятью, контроллеры, каналы ввода/вывода)

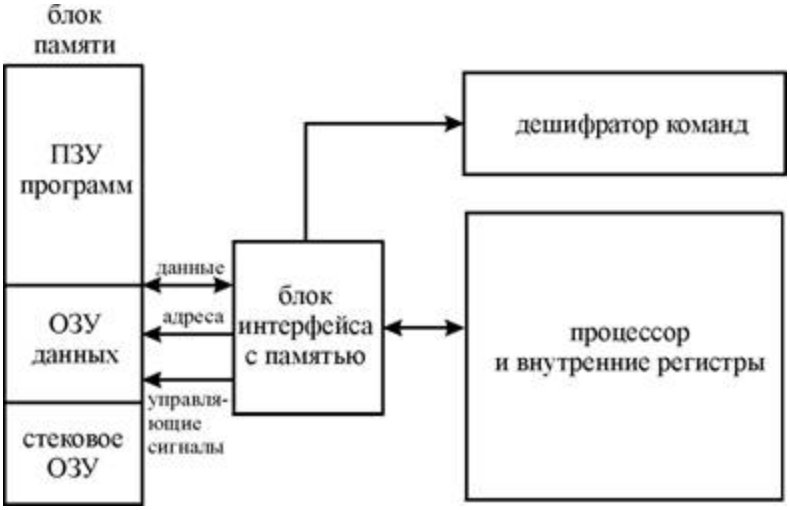
4. Принцип программного управления.

Все вычисления, предусмотренные алгоритмом решения задачи, должны быть представлены в виде программы, состоящей из последовательности управляющих слов — команд. Каждая команда предписывает некоторую операцию из набора операций, реализуемых вычислительной машиной. Команды программы хранятся в последовательных ячейках памяти вычислительной машины и выполняются в естественной последовательности, то есть в порядке их положения в программе. При

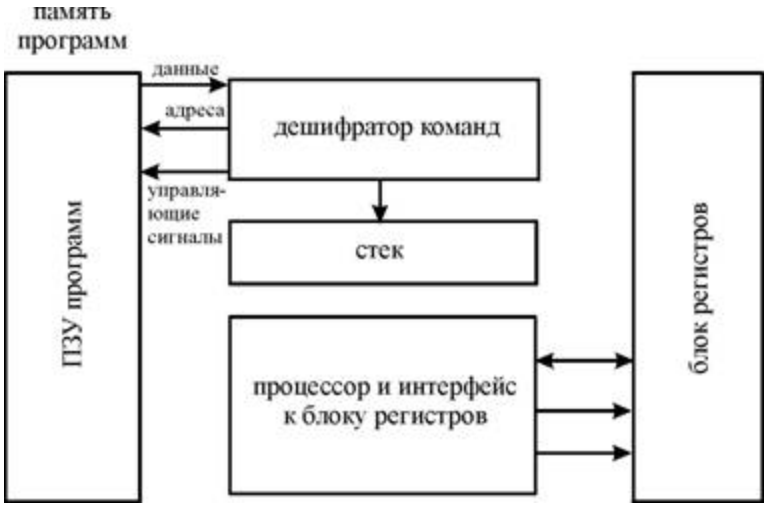
необходимости, с помощью специальных команд, эта последовательность может быть изменена. Решение об изменении порядка выполнения команд программы принимается либо на основании анализа результатов предшествующих вычислений, либо безусловно.

5. Каноническая структура компьютера. Принстонская и гарвардская архитектура ЭВМ.

Принстонский университет разработал компьютер, который имел общую память для хранения программ и данных. Такая архитектура компьютеров больше известна как архитектура Фон-Неймана по имени научного руководителя этой разработки



Гарвардский университет представил разработку компьютера, в котором для хранения программ, данных и стека использовались отдельные банки памяти



Принстонская архитектура выиграла соревнование, так как она больше соответствовала уровню технологии того времени. Использование общей памяти оказалось более предпочтительным из-за ненадежности ламповой электроники (это было до широкого распространения транзисторов) - при этом возникало меньше отказов.

Гарвардская архитектура почти не использовалась до конца 70-х годов, когда производители микроконтроллеров поняли, что эта архитектура дает преимущества устройствам, которые они разрабатывали.

6. Достоинства и недостатки неймановской архитектуры ЭВМ.

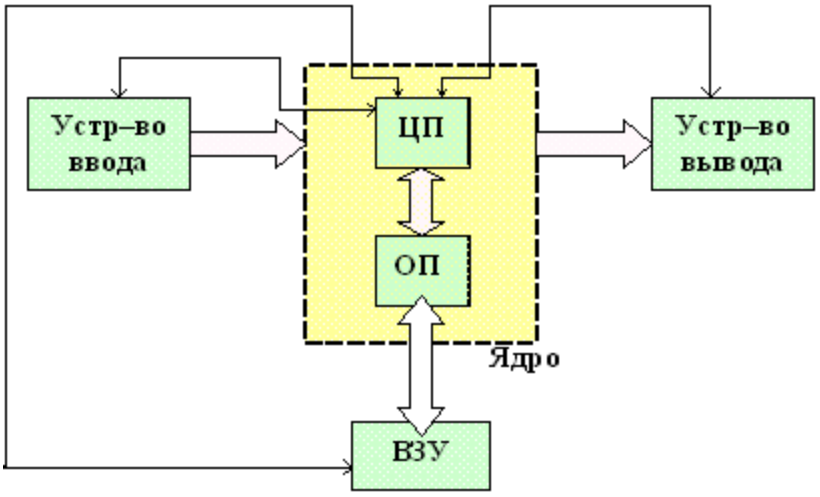
Достоинства:

- Универсальность (решение любой алгоритмически поставленной задачи, при условии отсутствия ограничений на время)
  а) функциональная полнота системы команд
  б) возможность смены программы
- Предельная экономичность (минимум затрат на оборудование и аппаратные средства для вычисления)

Недостатки:

- Несоответствие принципам Неймановской архитектуры концепции языков программирования высокого уровня (семантический разрыв)
- Наличие узкого места в виде памяти компьютера
- Невозможность реализации внутреннего параллелизма, присущего многим алгоритмам (векторно-матричная обработка).

7. Понятие ядра компьютера и периферии. Организация их взаимодействия.



Независимо от принадлежности компьютера некоторому классу или типу, его в первом приближении можно разделить на 2 части:

1. центральную;
2. периферийную.

Центральную часть принято называть ядром. В ядро входят два основных устройства компьютера: ЦП и ОП.

Периферийную часть можно условно представить устройствами трех типов:

- внешние запоминающие устройства, которые образуют внешнюю память (к ним относятся накопители на магнитных дисках и на магнитных лентах);
- устройства ввода;
- устройства вывода.

Обмен информацией между ядром и периферией, а так же между устройствами ядра осуществляется на уровне аппаратных интерфейсов. Организация обмена между ядром и периферийной частью компьютера возлагается на систему ввода/вывода (I/OS – Input/Output System).

8. Система ввода-вывода компьютера. Аппаратные и программные средства ввода-вывода

Система ввода/вывода представляет собой аппаратно - программный комплекс. Аппаратная часть I/O S включает в себя:

1. собственно периферийные устройства (разделённые на УВ/В и ВЗУ);
2. контроллеры ПУ(устройства управления);
3. контроллеры для организации обмена, в частности, контроллер DMA – Direct Memory Accses (ПДП-прямой доступ к памяти) PIC (Program Interrupt Controller – Программируемый Контроллер Прерываний);
4. интерфейсы (шины);
5. система прерываний.

Программная часть I/O S включает в себя:

1. супервизор (Supervisor) в/в;
2. драйверы ВУ.

Для современных программных средств I/O S типичным свойством является многоуровневая (иерархическая) организация, в частности, многоуровневые драйверы.

Программное обеспечение I/O S разделяется на устройство-зависимую часть и устройство независимую часть. Устройство-независимое ПО выполняет следующие функции:

- буферизация;
- защита (сообщения об ошибках);
- блокирование (блочный характер передачи);
- обеспечивание единообразного программного интерфейса для драйверов устройств.

9. Способы адресации внешних устройств.

Существует два подхода:

- раздельное адресное пространство
- единое адресное пространство

Использование раздельного адресного пространства предполагает, что одни и те же адреса могут применяться к адресации ячеек памяти, так и для адресации портов ввода/вывода. При этом в системе команд процессора должны использоваться специальные команды для ввода/вывода, отличные от команд обмена с памятью. Достоинством подобного подхода принято считать возможность использования более короткого адреса порта ввода/вывода по сравнению с адресом ячейки памяти.

Использование единого адресного пространства существенно влияет как на систему команд процессора, так и на управление вводом/выводом на аппаратном уровне. Некоторая область адресного пространства в старших адресах используется не

для адресации памяти, а для адресации портов ввода/вывода. Подобный способ хорошо вписывается в рамки так называемого магистрального интерфейса.

В системе команд отсутствуют специальные команды ввода/вывода, и обмен между памятью и портами ввода/вывода реализуется по аналогии с обменом между процессором и памятью с использованием обычной команды типа MOV.

10. Понятие интерфейса. Виды интерфейсов, используемых в компьютерных системах.

Интерфейс - способ сопряжения и взаимодействия между несколькими объектами. Подразделяется на: аппаратный, программный, пользовательский.

Основными типами линий (шин), входящих в состав аппаратного интерфейса, являются шина адреса, шина данных, шина управления, линии синхронизации, линии запросов прерывания, линии питания, линии заземления.

Основные характеристики интерфейса:

- Пропускная способность определяемая максимальным количеством бит или байт данных, передаваемых по интерфейсу за одну секунду.
- Информационная ширина (количество бит или байт данных, передаваемых параллельно по шине данных, т.е. разрядность линии)
- Максимально возможное удаление устройств, подключенных к интерфейсу.

Уровни представления интерфейсов:

- логический
- физический
- конструктивный

11. Понятие системного интерфейса. Уровни его представления.

Внутримашинный системный интерфейс – система связи и сопряжения узлов и блоков ЭВМ между собой – представляет собой совокупность электрических линий связи (проводов), схем сопряжения с компонентами компьютера, протоколов (алгоритмов) передачи и преобразования сигналов. Существуют два варианта организации внутримашинного интерфейса.

1. Многосвязный интерфейс , каждый блок ПК связан с прочими блоками своими локальными проводами; многосвязный интерфейс применяется, как правило, только в простейших бытовых ПК.

2. Односвязный интерфейс: все блоки ПК связаны друг с другом через общую или системную шину.

В подавляющем большинстве современных ПК в качестве системного интерфейса используется системная шина. Важнейшими функциональными характеристиками системной шины являются: количество обслуживаемых ею устройств и ее пропускная способность, т.е. максимально возможная скорость передачи информации. Пропускная способность шины зависит от ее разрядности (есть шины 8 -, 16 -, 32-и 64-разрядные) и тактовой частоты, на которой шина работает. В качестве системной шины в разных ПК использовались и могут использоваться:

• шины расширений – шины общего назначения, позволяющие подключать большое число самых разнообразных устройств;

• локальные шины, специализирующиеся на обслуживании небольшого количества устройств определенного класса

12. Состав шин системного интерфейса.

В состав системной шины в зависимости от типа процессора входит одна или несколько шин адреса, одна или несколько шин данных и шина управления. Несколько шин данных и адреса применяется для увеличения производительности процессора и используется только в сигнальных процессорах. В универсальных процессорах и контроллерах обычно применяется одна шина адреса и одна шина данных.

В понятие шины вкладывают разное значение при рассмотрении различных вопросов. В простейшем случае под понятием шина подразумевают параллельно проложенные провода, по которым передается двоичная информация. При этом по каждому проводу передается отдельный двоичный разряд. Информация может передаваться в одном направлении, как, например, для шины адреса или шины управления, или в различных направлениях (для шины данных). По шине данных информация передается либо к процессору, либо от процессора в зависимости от операции записи или чтения, которую в данный момент осуществляет процессор.

В любом случае все сигналы, необходимые для работы системной шины формируются микросхемой процессора как это рассматривалось при изучении блока обработки данных. Иногда для увеличения скорости обработки информации функции управления системной шины берет на себя отдельная микросхема (например контроллер прямого доступа к памяти или сопроцессор). Арбитраж доступа к системной шине при этом осуществляет контроллер системной шины (в простейшем случае достаточно сигнала занятости шины).

В некоторых случаях в понятие шина дополнительно включают требования по уровням напряжения, которыми представляются нули и единицы, передаваемые по ее проводам. В состав требований могут быть включены длительности фронтов передаваемых сигналов, типы используемых разъемов и их распайка, последовательность передаваемых сигналов и скорость их передачи.

13. Контроллеры внешних устройств. Параллельная и последовательная передача данных.

Контроллер – специализированное устройство (или плата), управляющее работой некоторого периферийного устройства и обеспечивающее его связь с системной платой. Например, контроллер клавиатуры или жёсткого диска.

Параллельная и последовательная передачи данных хотя и служат одной цели - обмену данными и связи между периферией и модулем обработки данных, но используют различные методы и принципы обмена информацией.

Параллельная связь означает, что биты передаются все одновременно (параллельно).

В отличие от последовательной передачи данных параллельная передача, как правило, однонаправленная, т.е. данные передаются только в одном направлении.

В последовательной передаче данных отдельные биты пересылаются (или принимаются) последовательно друг за другом, при этом возможен обмен данными в двух направлениях.

Поскольку данные обычно представлены на шине микропроцессора в параллельной форме (байтами, словами), их последовательный ввод-вывод оказывается несколько сложным. Для последовательного ввода потребуются средства преобразования последовательных входных данных в параллельные данные, которые можно поместить на шину. С другой стороны, для последовательного вывода необходимы средства преобразования параллельных данных, представленных на шине, в последовательные выходные данные. Это осуществляется с помощью регистров сдвига.

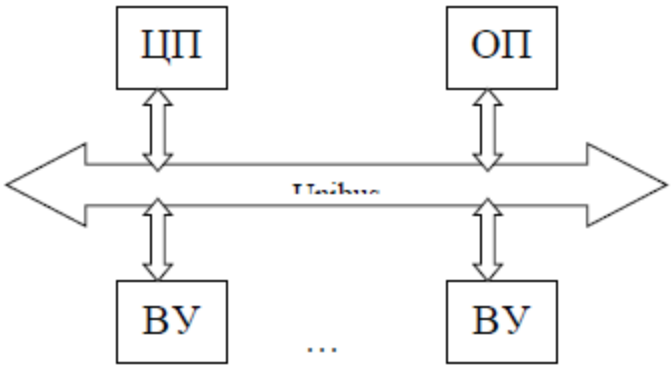
14. Структура компьютера с программно-управляемым интерфейсом.

Данный режим характеризуется тем, что все действия по вводу/выводу реализуются командами прикладной программы.

Для большинства ВУ до выполнения операций ВВ надо убедиться в их готовности к обмену, т.е. ВВ является асинхронным. Общее состояние устройства характеризуется флагом готовности READY, называемым также флагом готовности/занятости (READY/BUSY).

Процессор проверяет флаг готовности с помощью одной или нескольких команд. Если флаг установлен, то инициируются собственно ввод или вывод одного или нескольких слов данных. Когда же флаг сброшен, процессор выполняет цикл из 2-3 команд с повторной проверкой флага READY до тех пор, пока устройство не будет готово к операциям ВВ.

15. Структура компьютера с общей шиной.



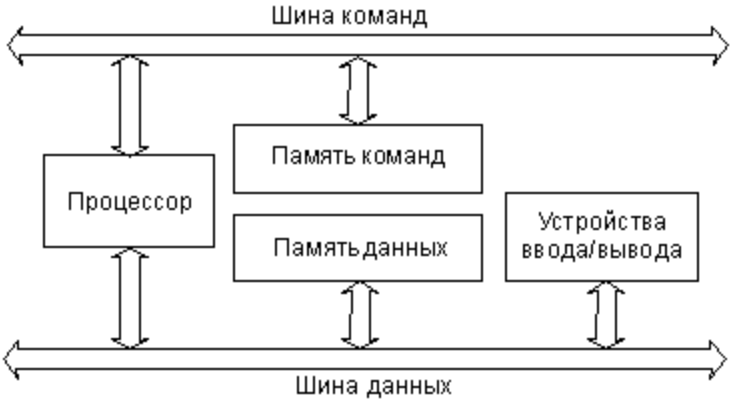
Основные особенности:

1. Для связи между любыми компонентами используются одни и те же линии интерфейса (ША, ШД, ШУ и т.п.), а также сигналы, управляющие передачей. Этот факт в значительной степени упрощает организации связи между устройством и обеспечивает простоту наращивания структуры путем подключения дополнительных устройств.
2. Использование единого интерфейса предполагает, что в любой момент времени по нему может быть организован обмен только между двумя устройствами, одно из которых является ведущим, а другое исполнительным, ведомым. Ведущим устройством

не может быть ОП. Подобная структура является источником конфликтов между различными активными устройствами, требующими практически одновременного взаимодействия с шиной для передачи данных. Компьютеры с подобной структурой не предполагают значительного наращивания числа устройств. Подобная структура для увеличения производительности требует введения дополнительных шин для разгрузки основной.

3. В компьютерах с общей шиной могут быть реализованы различные способы организации В/В, такие как PIO, В/В по прерыванию, а также В/В в режиме DMA.

16. Структура компьютера с раздельными шинами.

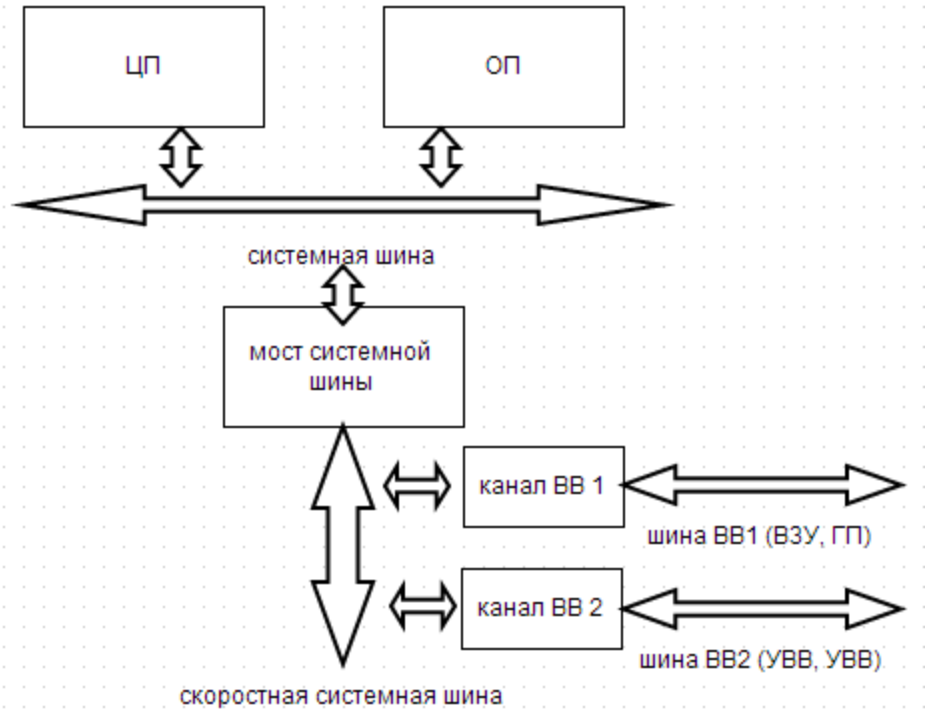


Архитектура с раздельными шинами данных и команд сложнее, она заставляет процессор работать одновременно с двумя потоками кодов, обслуживать обмен по двум шинам одновременно. Программа может размещаться только в памяти команд, данные — только в памяти данных. Такая узкая специализация ограничивает круг задач, решаемых системой, так как не дает возможности гибкого перераспределения памяти. Память данных и память команд в этом случае имеют не слишком большой объем, поэтому применение систем с данной архитектурой ограничивается обычно не слишком сложными задачами.

Преимущество данной архитектуры в быстродействии. В случае двухшинной архитектуры обмен по обеим шинам может быть независимым, параллельным во времени. Соответственно, структуры шин (количество разрядов кода адреса и кода данных, порядок и скорость обмена информацией и т.д.) могут быть выбраны оптимально для той задачи, которая решается каждой шиной. Поэтому при прочих равных условиях переход на двухшинную архитектуру ускоряет работу микропроцессорной системы, хотя и требует дополнительных затрат на аппаратуру, усложнения структуры процессора. Память данных в этом случае имеет свое распределение адресов, а память команд — свое.

Проще всего преимущества двухшинной архитектуры реализуются внутри одной микросхемы. В этом случае можно также существенно уменьшить влияние недостатков этой архитектуры. Поэтому основное ее применение — в микроконтроллерах, от которых не требуется решения слишком сложных задач, но зато необходимо максимальное быстродействие при заданной тактовой частоте.

17. Структура компьютера с каналами ввода-вывода.



18. Функции канального процессора.

1) Функции по установлению логической связи между внешним устройством и ОП (проверка готовности ВУ к обмену и инициализации канальной программы)
2) Функции по управлению обменом между ВУ и ОП
- последовательная выборка команд канальной программы из ОП, их декодирование и исполнение
- обеспечение приема, передачи и контроля, промежуточного хранения данных
- формирование текущего адреса ячеек основной памяти по которым считываются или записываются данные
- согласование форматов данных интерфейса ВУ с форматом интерфейса ОП
- подсчет числа переданных байт для определения окончания блока передаваемых данных
- выработка синхронизирующих и управляющих сигналов в соответствии с интерфейсом ВВ
- анализ особых ситуаций во внешнем устройстве во время обмена и передачи сигналов прерывания ЦП

3) Функции по завершению обмена
- определение момента завершения в канальной программе
- передача в ЦП сигнала прерывания по завершению обмена

19. Система команд компьютера. Основные виды команд.

Система команд компьютера включает в себя перечень машинных команд, реализуемых непосредственно аппаратными средствами. Именно система команд определяет возможности компьютера в плане решения разнообразных задач обработки данных.

Основной характеристикой системы команд является её мощность. Обычно под этим термином понимается количество разнообразных мнемокодов, используемых для символического обозначения на ассемблере.

Существует и другой подход к определению мощности системы команд, когда в разнообразие машинных команд включают не только разнообразие мнемокодов, но и разнообразие используемых режимов адресации и форматов команд.

Основные виды: команды передачи данных, арифметические операции, логические операции, сдвиги, команды ввода/вывода, команды управления.

20. Структура и формат машинных команд.

Структура машинной команды в общем случае состоит из двух основных частей: операционной и адресной.

Операционная часть задаёт тип выполняемой операции и обычно называется кодом операции, Operation Code (OpC).

Адресная часть задаёт адреса операндов, участвующих в операции, а так же адрес результата.

В зависимости от числа операндов, задаваемых в адресной части команды, машинные команды разделяются на трёхадресные, двухадресные, одноадресные и безадресные (нольадресные). В процессорах семейства Intel 80X86 используются безадресные, одно – и двухадресные команды. Использование безадресных команд (однобайтные команды, содержащие только код операции) может быть связано с двумя аспектами:

- использование неявной адресации операндов (их адреса не задаются в команде, но подразумеваются по умолчанию)
- для выполнения команды операндов не требуется (NOP, HLT)

В терминологии фирмы Intel операнды двухадресной команды называются источником Source (SRC) и приемником.

21. Понятие CISC и RISC-архитектуры.

Расширение набора команд, увеличение числа способов адресации, введение сложных команд сопровождаются увеличением длины кода команды, в первую очередь кода операции, что может приводить к использованию “расширяющегося кода операции”, увеличению числа форматов команд. Это вызывает усложнение и замедления процесса дешифрации кода операции и других процедур обработки команд. Возрастающая сложность процедур обработки команд заставляет прибегать к микропрограммному управлению устройствами с управляющей памятью вместо более быстродействующего устройства управления с “жесткой” логикой.

Сложный процессор с микропрограммным управлении трудно реализовать на одном кристалле, а это ведет к увеличению длинны межмодульных связей, таким образом устроены большинство процессоров с полным набором команд.

Напротив, при сокращении количества команд до некоторого оптимального значения, можно сократить длину команд и упростить управляющее устройство МП.

Поэтому при проектировании структуры МП выделилось два направления в отношении набора системы команд:

· CISC (Complicated Instruction Set Computer — использующий полный набор команд). Традиционная архитектура с широкой системой команд МП.

· RISC (Reduced Instruction Set Computer). Архитектура с сокращенным набором команд.

При использовании RISC архитектуры выбор набора команд и структуры процессора направлены на то, чтобы команды набора выполнялись за один машинный цикл МП. Выполнение более сложных, но редко встречающихся операций обеспечивают подпрограммы, состоящие из набора простых команд.

22. Особенности компьютеров RISC-архитектуры.

1) Не много машинных команд
2) Выполнение большинства машинных команд за 1 машинный такт
3) Упрощение устройства управления, как следствие - минимальная его площадь на кристалле.
4) Каждой программе выделяется некоторое подмножество регистров, образующих окно.
5) Ограниченное число форматов машинных команд и режимов адресации
6) Все команды обработки данных реализуются только над регистровыми операндами.
7) Широкое использование принципов суперскалярной и суперконвейерной обработки.

23. Способы адресации, используемые в ЭВМ.

1. Прямая
    a. памяти
    b. регистровая
2. Косвенная
    a. памяти
    b. регистровая
3. Относительная
    a. базовая
    b. индексаная
    c. базово-индексная без смещения
    d. базово-индексная со смещением
    e. относительно текущего счетчика команд

24. Способы адресации с модификацией адреса.

Все выше приведенные типы адресации касались одного операнда. А как же быть в случае с несколькими операндами или говоря другим языком – массивами? Тогда обычно указывается адрес массива и номер (индекс) элемента. Базовый (начальный) адрес указывается в команде. Кроме этого, там же, в команде, есть поле, где указан номер регистра, в котором лежит значение индекса или номер ячейки в

массиве относительно начального адреса. Тогда адрес каждой ячейки массива будет получаться из суммы начального адреса и того, что содержит указанный регистр. Это называется модификацией адресов. Кроме того, существует тип адресации, когда в регистре лежит начальный адрес. В команде указан адрес этого регистра, а так же записано смещение относительно начального адреса. Все остальные адреса операндов будут получены из суммы адреса и смещения. Такой вот тип адресации называется относительным.

При относительной адресации можно еще и модифицировать адреса. В этом случае адрес будет равен сумме начального адреса плюс смещение плюс содержимое индексного регистра.

Если система использует несколько типов адресации, то в команде обязательно записывается, какой способ будет применен в данный момент. Говоря языком ученым, в команде указывается признак адресации в поле признака операции.

25. Стековая адресация. Выполнение вычислений в стековых ЭВМ (на примере).

Инфиксное выражение $(1 + 2) \times 4 + 3$ в обратной польской нотации может быть записано так: 1 2 + 4 × 3 +

Вычисление производится слева направо, ввод интерпретируется как указано в приведённой ниже таблице (указано состояние стека после выполнения операции, вершина стека выделена красным цветом):

Ввод	Операция	Стек
1	поместить в стек	1
2	поместить в стек	1, 2
+	сложение	3
4	поместить в стек	3, 4
*	умножение	12
3	поместить в стек	12, 3
+	сложение	15

Результат, 15, в конце вычислений находится на вершине стека

26. Однопрограммный режим работы компьютера.

В однопрограммном режиме работы в памяти ЭВМ находится и выполняется только одна программа. Загрузка программы - счет - в/в - окончание обработки.

Достоинство: простота управления ходом вычислений.

Недостатки: простой отдельных устройств компьютера и следовательно невысокая эффективность.

27. Мультипрограммный режим работы компьютера.

В компьютер загружено несколько программ, которые выполняются во времени параллельно, совмещая для одной программы работу в ЦПУ, а остальных - с устройствами В/В. Характеризуется усложнением управления ходом задач в компьютере в целом, но позволяет обеспечить повышение загрузки ЦП, при этом может увеличиться общее время на выполнение той или иной программы в сравнении с однопрограммным режимом.

Ввод задания - обеспечения задания ресурсами - загрузка программы - (ожидание счета - счет - ожидание в/в - в/в) - окончание выполнения.

28. Средства мультипрограммирования.

Для обеспечения мультипрограммирования компьютер должен иметь:

1) Память, достаточную для размещения данных, относящихся к нескольким задачам, которые выполняются мультипрограммно.
2) Обеспечена параллельная работа процессора и ВУ.
3) Оснащен средствами обеспечивающих управление порядком задач
    a) ввод заданий относящихся к новым задачам
    b) обеспечение задач основной и внешней памятью, а также УВВ
    c) загрузке программ в основную память
    d) распределение времени работы CPU и внешних устройств
    e) обработка особых ситуаций в процессе выполнения программ и в процессе сбоев функций устройств
4) Компьютер должен иметь систему прерывания программ.
5) Компьютер должен иметь систему защиты памяти.

29. Функции управляющих программ операционной системы.

Управляющая программа – обязательный компонент любой ОС. Ее функции – планирование прохождения непрерывного потока заданий, управление распределением ресурсов, реализация принятых методов организации данных, управление операциями ввода-вывода (В-В), организация мультипрограммной работы, управление работоспособностью системы после сбоев и др.

Управляющая программа состоит из ряда компонентов, среди которых следует выделить четыре основных.

Управление статическими ресурсами (управление заданиями) осуществляет предварительное планирование потока заданий для выполнения и статич. распределение ресурсов между одновременно выполняемыми заданиями в процессе подготовки к выполнению. К статическим ресурсам относят разделы памяти (основной, виртуальной, внешней), доступные для использования устройства, допускающие только монопольное использование, наборы данных и др. такие ресурсы закрепляются за заданием или его частью с момента инициализации до момента завершения и используются обычно в монопольном порядке.

Управление динамическими ресурсами (управление задачами) осуществляет динамическое распределение ресурсов системы между несколькими задачами, решаемых одновременно в мультипрограммном режиме для выполняемого потока заданий. Входящие в ядро ОС и постоянно находящиеся в ОП.

Управление данными обеспечивает все операции В-В (т.е. обмен между ОП и ПУ) на физическом и логическом уровнях. Оно включает в себя ряд служб, обеспечивающих выполнение таких функций, как управление каталогом, управление распределением памяти прямого доступа, обработку ошибок В-В и др., реализует различные структуры данных и возможность доступа к ним.

Управление восстановлением регистрирует машинные сбои и отказы и восстанавливает работоспособность системы после сбоев, если это возможно.

Системные обрабатывающие программы выполняются под управлением управляющей системы, так же как и любая обрабатывающая программа, в т.ч. пользовательская. Это значит, что она в полном объеме может пользоваться услугами управляющей программы и не может самостоятельно выполнять системные функции. Так, обрабатывающая программа не может самостоятельно осуществлять собственный В-В. операции В-В обрабатывающая программа реализует с помощью запросов к управляющей программе, которая и выполняет непосредственно ввод и вывод данных. Централизованное выполнение системных функций управляющей программой позволяет выполнять их более эффективно и обеспечивает высокий уровень услуг для пользователя.

К системным обрабатывающим программам относятся программы, входящие в состав ОС: ассемблеры, трансляторы, редакторы связей, загрузчик, программы обслуживания и ряд других. Трансляторы, редактор связей и загрузчик образуют основу систем программирования, построенных на базе ОС.

30. Привилегированные операции и состояния процессора.

Чтобы контролировать порядок использования привилегированных операций, принято выделять два альтернативных состояния процессора: состояние супервизор, в котором процессор выполняет программы супервизора, и состояние задача, в котором процессор выполняет прикладные программы. В состоянии супервизор допускается выполнение любых операций, а в состоянии задача - только непривилегированных операций. Если процессор находится в состоянии задача, появление в программе привилегированной операциисчитается ошибкой и выполнение программы прекращается. Состояние супервизор-задача устанавливается командой, инициирующей программу. Эта команда относится к классу привилегированных и используется только супервизором. Если инициируется прикладная программа, то устанавливается состояние задача; если инициируется программа супервизора, процессор переключается в состояние супервизор.

Команды, которые недопустимы в состоянии задача, называются привилегированными. К их числу относятся команды, предназначенные для модификации и проверки ключей памяти, полей управления системой, расположенных в PSW и управляющих регистрах, а также команды, предназначенные для измерения времени, обработки префиксов, обеспечения связи между несколькими CPU и для

управления вводом-выводом. Привилегированная команда, встретившаяся в состоянии задача, вызывает программное прерывание по привилегированной операции.

31. Организация прерывания программ. Источники прерываний.

Назначение системы прерываний - формирование реакции на определенные события путем прерывания работы процессора выполняющего одну программу и переключения на другую.

Причины прерываний:

1) Внешние прерывания (события вне компьютера)
2) Прерывания от ввода/вывода
3) Программные (некорректность кода операции, нарушение защиты памяти, переполнение)
4) Прерывание при обращении к супервизору
5) Прерывание от схем контроля

32. Основные сведения об организации ввода/вывода информации. Программно-управляемая передача данных.

Передача информации с периферийного устройства в ядро ЭВМ называется операцией ввода, а передача из ядра ЭВМ в периферийное устройство - операцией вывода.
Связь устройств ЭВМ друг с другом осуществляется с помощью средств сопряжения - интерфейсов.

Интерфейс представляет собой совокупность линий и шин, сигналов, электронных схем и алгоритмов, предназначенную для осуществления обмена информацией между устройствами. От характеристик интерфейсов во многом зависят производительность и надежность вычислительной машины. При разработке систем ввода-вывода должны быть решены следующие проблемы:

1) Должна быть обеспечена возможность реализации машин с переменным составом оборудования.
2) Для эффективного использования оборудования ЭВМ должны реализовываться параллельная во времени работа процессора над программой и выполнение периферийными устройствами процедур ввода-вывода.
3) Необходимо стандартизировать программирование операций ввода-вывода для обеспечения их независимости от особенностей периферийного устройства.
4) Необходимо обеспечить автоматическое распознавание и реакцию ядра ЭВМ на многообразие ситуаций, возникающих в ПУ (готовность устройства, различные неисправности и т.п.).

В системах ввода-вывода ЭВМ используются два основных способа организации передачи данных между памятью и периферийными устройствами:
программно-управляемая передача и прямой доступ к памяти.
Программно-управляемая передача данных осуществляется при непосредственном участии и под управлением процессора, который при этом выполняет специальную подпрограмму процедуры ввода-вывода. Данные между памятью и периферийным устройством пересылаются через процессор.

Операция ввода - вывода инициируется текущей командой программы или запросом прерывания от периферийного устройства. При этом процессор на все время выполнения операции ввода-вывода отвлекается от выполнения основной программы.

Кроме того при пересылке блока данных процессору приходится для каждой единицы передаваемых данных выполнять несколько команд, чтобы обеспечить буферизацию, преобразование форматов данных, подсчет количества переданных данных, формирование адресов в памяти и т.п. Это сильно снижает скорость передачи данных (не выше 100 Кб/сек), что недопустимо при работе с высокоскоростными ПУ. Между тем потенциально возможная скорость обмена данными при вводе-выводе определяется пропускной способностью памяти

33. Режим прямого доступа к памяти.

Одним из способов обмена данными с ВУ является обмен в режиме прямого доступа к памяти (ПДП). В этом режиме обмен данными между ВУ и основной памятью микроЭВМ происходит без участия процессора. Обменом в режиме ПДП управляет не программа, выполняемая процессором, а электронные схемы, внешние по отношению к процессору. Обычно схемы, управляющие обменом в режиме ПДП, размещаются или в специальном контроллере, который называется контроллером прямого доступа к памяти, или в контроллере самого ВУ.

Обмен данными в режиме ПДП позволяет использовать в микроЭВМ быстродействующие внешние запоминающие устройства, такие, например, как накопители на жестких магнитных дисках, поскольку ПДП может обеспечить время обмена одним байтом данных между памятью и ВЗУ, равное циклу обращения к памяти.

Для реализации режима прямого доступа к памяти необходимо обеспечить непосредственную связь контроллера ПДП и памяти микроЭВМ. Для этой цели можно было бы использовать специально выделенные шины адреса и данных, связывающие контроллер ПДП с основной памятью. Но такое решение нельзя признать оптимальным, так как это приведет к значительному усложнению микроЭВМ в целом, особенно при подключении нескольких ВЗУ. В целях сокращения количества линий в шинах микроЭВМ контроллер ПДП подключается к памяти посредством шин адреса и данных системного интерфейса. При этом возникает проблема совместного использования шин системного интерфейса процессором и контроллером ПДП. Можно выделить два основных способа ее решения: реализация обмена в режиме ПДП с "захватом цикла" и в режиме ПДП с блокировкой процессора.

34. Организация обмена в режиме прямого доступа к памяти.

В этом режиме функции CPU минимальны:

- инициализация обмена в режиме ПДП
- осуществление реакции на завершение обмена в режиме ПДП.

Поскольку обмен производится между ОП и ВУ, то такой обмен производится блоками данных. Для управления в режиме DMA имеется контроллер прямого доступа. Он осуществляет управление обменом чаще всего на аппаратном уровне, т.е. является

микропгрограммным автоматом. КПДП содержит некоторое число программно-доступных регистров, которые для ЦП являются портами ВВ.

Инициализация обмена в режиме ПДП сводится к заданию режима работы и необходимого порта ВВ. Для этого из ЦП в регистры КПДП вносятся следующие данные:

- начальный адрес блока/области данных
- код операции обмена (ввод или вывод)
- адрес прямого доступа (адрес порта ВВ)

35. Функции контроллера ПДП (прямого доступа к памяти).

Работа контроллера прямого доступа памяти происходит в такой последовательности:

1.Интерфейс устройства ввода–вывода при наличии у него готовых данных для оперативной памяти или готовности принять данные от оперативной памяти, выдаёт сигнал запроса ПДП.

2.Контроллер ПДП формирует сигнал запроса шины и получает управление шиной от центрального процессора.

3.На шину адреса помещается адрес ячейки памяти, который контроллер ПДП выдаёт из своего регистра адреса.

4.Контроллер ПДП выдаёт подтверждение интерфейсу устройства ввода–вывода на ввод или вывод данных.

5.Данные помещаются на шину данных, при этом при выводе данных, данные поступают из оперативной памяти, а при вводе—данные поступают от интерфейса устройства ввода–вывода.

6.Данные, в случае вывода фиксируются интерфейсом устройства ввода–вывода, а в случае ввода, данные поступают в ячейку памяти, адрес которой помещён на шине адреса.

7.Контроллер освобождает шину и управление шиной возвращается центральному процессору.

8.Если происходил ввод данных в оперативную память, то счётчик байт вычитает из своего значения «1», а регистр адреса прибавляет «1», тем самым указывая на

следующую свободную ячейку памяти. Если счётчик байт не содержит «0», то алгоритм повторяется, в противном случае передача данных останавливается.

36. Сходство и различие канального обмена и режима ПДП (прямого доступа к памяти).

Общее между ними то, что они осуществляют свою работу практически без участия ЦП.

Отличие: в случае канального ввода-вывода осуществляется программное управление, а при DMA функции управления возлагаются на контроллер DMA (и чаще всего реализуются аппаратно.

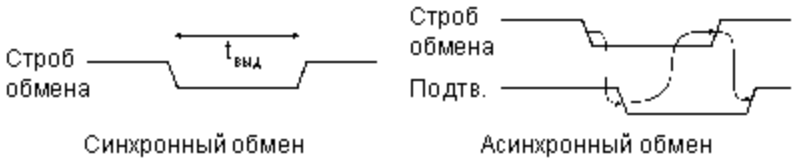
37. Сходство и различие программно-управляемого обмена и канального обмена.

Поскольку канальный ввод-вывод осуществляет организацию обмена по собственным канальным программам, он является программно-управляемым. Однако в отличие от PIO программу обмена выполняет не ЦПУ, а канальный процессор.

38. Организация синхронного обмена. (объединен со следующим)

39. Организация асинхронного обмена.

При синхронном обмене процессор заканчивает обмен данными самостоятельно, через раз и навсегда установленный временной интервал выдержки ($t_{выд}$), то есть без учета интересов устройства-исполнителя;



При асинхронном обмене процессор заканчивает обмен только тогда, когда устройство-исполнитель подтверждает выполнение операции специальным сигналом (так называемый режим handshake — рукопожатие).

Достоинства синхронного обмена — более простой протокол обмена, меньшее количество управляющих сигналов. Недостатки — отсутствие гарантии, что исполнитель выполнил требуемую операцию, а также высокие требования к быстродействию исполнителя.

Достоинства асинхронного обмена — более надежная пересылка данных, возможность работы с самыми разными побыстродействию исполнителями. Недостаток — необходимость формирования сигнала подтверждения всеми исполнителями, то есть дополнительные аппаратурные затраты.

Какой тип обмена быстрее, синхронный или асинхронный? Ответ на этот вопрос неоднозначен. С одной стороны, приасинхронном обмене требуется какое-то время на выработку, передачу дополнительного сигнала и на его обработку процессором. С другой стороны, при синхронном обмене приходится искусственно увеличивать длительность строба обмена для соответствия требованиям большего числа исполнителей, чтобы они успевали обмениваться информацией в темпе процессора. Поэтому иногда в магистрали предусматривают возможность как синхронного, так и

асинхронного обмена, причем синхронный обмен является основным и довольно быстрым, а асинхронный применяется только для медленных исполнителей.

По используемому типу обмена магистрали микропроцессорных систем также делятся на синхронные и асинхронные.

40. Организация обмена по прерыванию.

Одной из разновидностей программно-управляемого обмена данными с ВУ в микроЭВМ является обмен с прерыванием программы, отличающийся от асинхронного программно-управляемого обмена тем, что переход к выполнению команд, физически реализующих обмен данными, осуществляется с помощью специальных аппаратных средств. Команды обмена данными в этом случае выделяют в отдельный программный модуль - подпрограмму обработки прерывания. Задачей аппаратных средств обработки прерывания в процессоре микроЭВМ как раз и является приостановка выполнения одной программы (ее еще называют основной программой) и передача управления подпрограмме обработки прерывания. Действия, выполняемые при этом процессором, как правило, те же, что и при обращении к подпрограмме. Только при обращении к подпрограмме они инициируются командой, а при обработке прерывания - управляющим сигналом от ВУ, который называют "Запрос на прерывание" или "Требование прерывания".

Эта важная особенность обмена с прерыванием программы позволяет организовать обмен данными с ВУ в произвольные моменты времени, не зависящие от программы, выполняемой в микроЭВМ. Таким образом, появляется возможность обмена данными с ВУ в реальном масштабе времени, определяемом внешней по отношению к микроЭВМ средой. Обмен с прерыванием программы существенным образом экономит время процессора, затрачиваемое на обмен. Это происходит за счет того, что исчезает необходимость в организации программных циклов ожидания готовности ВУ (см. примеры 2.1 и 2.2, Параллельная передача данных), на выполнение которых тратится значительное время, особенно при обмене с медленными ВУ.

Обычно задача сохранения содержимого счетчика команд и регистра состояния процессора возлагается на аппаратные средства обработки прерывания. Сохранение содержимого других регистров процессора, используемых в подпрограмме обработки прерывания, производится непосредственно в подпрограмме. Отсюда следует достаточно очевидный факт: чем больший объем информации о прерванной программе сохраняется программным путем, тем больше время реакции микроЭВМ на сигнал прерывания, и наоборот. Предпочтительными с точки зрения повышения производительности микроЭВМ (сокращения времени выполнения подпрограмм обработки, а, следовательно, и основной программы) являются уменьшение числа команд, обеспечивающих сохранение информации о прерванной программе, и реализация этих функций аппаратными средствами.

Формирование сигналов прерываний - запросов ВУ на обслуживание происходит в контроллерах соответствующих ВУ. В простейших случаях в качестве сигнала

прерывания может использоваться сигнал "Готовность ВУ", поступающий из контроллера ВУ в системный интерфейс микроЭВМ. Однако такое простое решение обладает существенным недостатком - процессор не имеет возможности управлять прерываниями, т. е. разрешать или запрещать их для отдельных ВУ. В результате организация обмена данными в режиме прерывания с несколькими ВУ существенно усложняется.

Для решения этой проблемы регистр состояния и управления контроллера ВУ (рис. 3.11) дополняют еще одним разрядом - "Разрешение прерывания". Запись 1 или 0 в разряд "Разрешение прерывания" производится программным путем по одной из линий шины данных системного интерфейса. Управляющий сигнал системного интерфейса "Запрос на прерывание" формируется с помощью схемы совпадения только при наличии единиц в разрядах "Готовность ВУ" и "Разрешение прерывания" регистра состояния и управления контроллера.

41. Система прерываний компьютера. Основные виды источников прерывания. (смотрим предыдущий вопрос)

В микроЭВМ обычно используется одноуровневая система прерываний, т. е. сигналы "Запрос на прерывание" от всех ВУ поступают на один вход процессора. Поэтому возникает проблема идентификации ВУ, запросившего обслуживание, и реализации заданной очередности (приоритета) обслуживания ВУ при одновременном поступлении нескольких сигналов прерывания. Существуют два основных способа идентификации ВУ, запросивших обслуживания:

- программный опрос регистров состояния (разряд "Готовность ВУ") контроллеров всех ВУ;
- использование векторов прерывания.

Организация системы прерываний в микроЭВМ с использованием векторов прерываний позволяет устранить указанный недостаток. При такой организации системы прерываний ВУ, запросившее обслуживания, само идентифицирует себя с помощью вектора прерывания - адреса ячейки основной памяти микроЭВМ, в которой хранится либо первая команда подпрограммы обслуживания прерывания данного ВУ, либо адрес начала такой подпрограммы. Таким образом, процессор, получив вектор прерывания, сразу переключается на выполнение требуемой подпрограммы обработки прерывания. В микроЭВМ с векторной системой прерывания каждое ВУ должно иметь собственную подпрограмму обработки прерывания.

Основные источники прерываний:

- Внешние прерывания (события вне компьютера)
- Прерывания от ввода/вывода
- Программные (некорректность кода операции, нарушение защиты памяти, переполнение)
- Прерывания при обращении к супервизору
- Прерывания от схем контроля

42. Организация системы прерываний. Вектор прерывания. Понятие глубины прерывания. Уровни прерывания.

Вектор прерывания — закреплённый за устройством номер, который идентифицирует соответствующий обработчик прерываний. Векторы прерываний объединяются в таблицу векторов прерываний, содержащую адреса обработчиков прерываний. Местоположение таблицы зависит от типа и режима работы процессора.

Глубина прерывания - максимальное число обрабатываемых прерываний (простым языком - сколько раз мы можем прерываться в том числе и в прерванном состоянии). При глубине больше одного возникает задача управления приоритетами прерывания.

Уровень прерывания. Установка уровня прерывания на определенное значение отсекает прерывания этого и более низких уровней, разрешая обработку только прерываний с более высоким приоритетом.

43. Понятие приоритета прерываний. Абсолютный и относительный приоритет. Организация обработки запросов на прерывание.

Абсолютный приоритет означает, что сигнал прерывания с бОльшим приоритетом прерывает обработку текущего прерывания, а при относительном сначала обработается текущее прерывание, а потом выберется прерывание с наибольшим приоритетом.

Обработка запросов прерывания может быть осуществлена:

- по уровню
- по маске
- по порогу

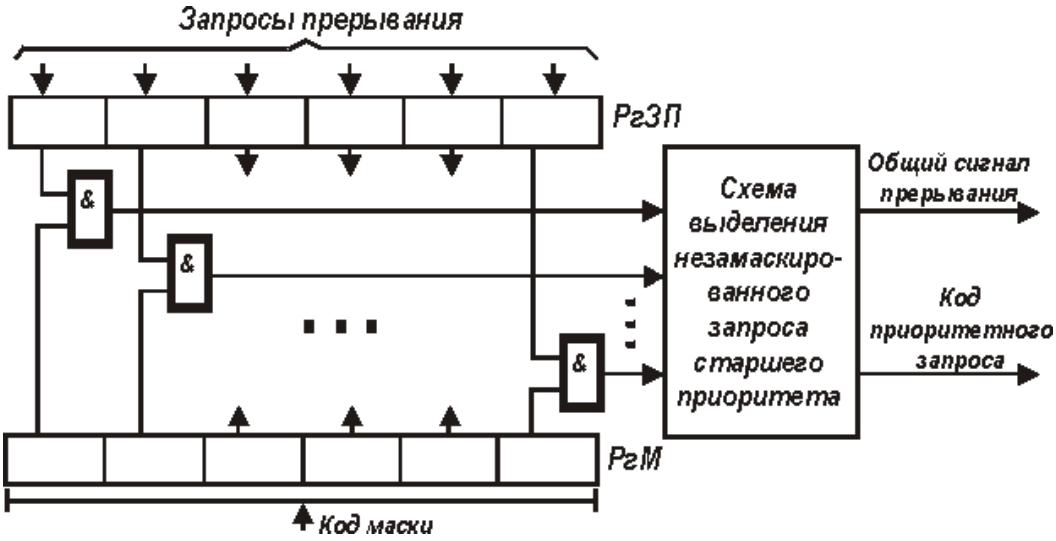
44. Программирование приоритетов по маске и по порогу.

В ЭВМ широко применяются два способа реализации пронграммно-управляемого приоритета прерывающих программ, в которых используются соответственно порог прерывания (в малых и микро-ЭВМ) и маски прерывания (в ЭВМ общего назначения).

Порог прерывания. Этот способ позволяет в ходе вычислинтельного процесса программным путем изменять уровень принаритета процессора (а следовательно, и обрабатываемой в данный момент на процессоре программы) относительно приоритетов запросов источников прерывания (в основном пенриферийных устройств), другими словами, задавать порог пренрывания, т. е. минимальный уровень приоритета запросов, конторым разрешается прерывать программу, идущую на процеснсоре.

Порог прерывания задается командой программы, устананвливающей в регистре порога прерывания код порога прерываннния. Специальная схема выделяет наиболее приоритетный занпрос прерывания, сравнивает его приоритет с порогом прерывания и, если он оказывается выше порога, вырабатывает общий сигнал прерывания, и начинается процедура прерывания (рис. 1.4, в).

В современных ЭВМ общего назначения наибольшее раснпространение получило программное управление приоритетом на основе маски прерывания



Маска прерывания представляет собой двоичный код, разнряды которого поставлены в соответствие запросам или класнсам прерывания. Маска загружается командой программы в регистр маски. Состояние 1 в данном разряде регистра маски разрешает, а состояние 0 запрещает(маскирует) прерывание тенкущей программы от соответствующего запроса. Таким обранзом, программа, изменяя маску в регистре маски, может устаннавливать произвольные приоритетные соотношения между программами без перекоммутации линий, по которым постунпают запросы прерывания. Каждая прерывающая программа может установить свою маску. При формировании маски 1 устанавливаются в разряды, соответствующие запросам (пренрывающим программам) с более высоким, чем у данной пронграммы, приоритетом.

45. Принцип микропрограммного управления. Операционный и управляющий автоматы, их взаимодействие.

Дня выполнения операций над информацией используютсяоперационные устройства— арифметико-логические, управления, контроллеры ВУ и т. п. Фу нкцией операционного устройства является выполнение заданного множества операций F = {/i./j.—./*} над входными словами из множества /), с целью вычисления выходных слов из множестваD_0, представляющих результаты операций$D_a = f_k(D,), fa^{-}K•$

Функциональная и структурная организация операционных устройств, определяющая порядок функционирования и структуру устройств, базируется на принципе микропрограммного управления,который состоит в следующем :

1. Любая операцияf_k, реализуемая устройством, рассматривается как сложное действие, которое разделяется на последовательность элементарных действий над словами информации, называемыхмикрооперациями.
2. Для управления порядком следования микроопераций используютсялогические условия,которые, в зависимости от значений слов, преобразуемых микрооперациями, принимают значения "истина" или "ложь" (I или 0).
3. Процесс выполнения операций в устройстве описывается в форме алгоритма, представляемого в терминах микроопераций и логических условий и называемогомикропрограммой.Микропрограмма определяет порядок проверки

значений логических условий и следования микроопераций, необходимый для получения требуемых результатов.

4. Микропрограмма используется как форма представления функции устройства, на основе которой определяются структура и порядок функционирования устройства во времени.

Сказанное можно рассматривать как содержательное описание принципа микропрограммного управления, из которого следует, что структура и порядок функционирования операционного устройства предопределяютсяалгоритмами выполнения операции из F.

46. Микрооперация. Микрокоманда. Виды микрокоманд. Микропрограмма.

Процесс функционирования ВМ состоит из последовательности пересылок инфы между ее узлами и элементарных действий, выполняемых в узлах. Любое элементарное действие производится при поступлении соответствующего сигнала управления из микропрограммного аппарата устройства управления. Элементарные пересылки или преобразование инфы, выполняемые в течении одного такта сигналов синхронизации называется микрооперацией. В течении одного такта могут выполнятся несколько микроопераций. Совокупность сигналов управления, вызывающих микрооперацию, выполняемых в одном такте – называют микрокомандой. Последовательность микрокоманд – микропрограмма.

Записанные в памяти микрокоманды определяют работу всех устройств машины, выбирая в каждом такте нужные совокупности элементарных машинных операций, а последовательность микрокоманд обеспечивает выполнение заданной команды.

47. Горизонтальное кодирование микрокоманд. (объединен)

48. Вертикальное кодирование микрокоманд. (объединен)

49. Смешанное кодирование микрокоманд.

Кодирование микроопераций (МО) в микрокоманде может выполняться по одному из 3-х вариантов:

1) горизонтальное кодирование;
2) вертикальное (максимальное) кодирование;
3) смешанное кодирование.

При горизонтальном кодировании за каждой микрооперацией в составе операционной части микрокоманды (ОЧМК) закрепляется свой разряд. Единичное значение разряда характеризует выполнение микрооперации в текущей микрокоманде, а нулевое - ее отсутствие.

Вертикальное кодирование предусматривает выделение для каждой МО отдельного многоразрядного кода, разрядность n которого определяется общим числом N различных МО реализуемых в операционном автомате, причем

$$n = 1 + \text{int}\ \log_2 N$$

Поскольку для первого способа кодирования длина ОЧМК может быть велика (при большом N), а второй способ при минимальной длине ОЧМК не позволяет включать в

микро-команду более одной микрооперации и требует применения достаточно сложных дешифраторов, чаще всего используется третий компромиссный вариант - смешанное кодирование.

Суть его в следующем: операционная часть разбивается на ряд операционных полей.

Их число определяется максимальным количеством одновременно выполняемых МО в одной микрокоманде. С каждым полем связана группа несовместных микроопераций (которые никогда не выполняются одновременно). Внутри каждого поля кодирование осуществляется вертикальным способом. Это позволяет с одной стороны иметь практически любую комбинацию МО в составе микрокоманды, с другой стороны, вертикальное кодирование внутри полей уменьшает общую длину ОЧМК, а более короткие коды приводят к упрощению дешифраторов ОЧ микрокоманды.

50. Управляющий автомат с хранимой микропрограммой.

Управляющий автомат с хранимой в памяти логикой – это последовательностное устройство, вырабатывающее распределенные во времени управляющие функциональные сигналы, задаваемые содержимым микропрограммной памяти.

Во всех случаях, перед тем как выполнить некую команду, надо считать ее из запоминающего устройства. А значит, что надо иметь некоторый регистр, где эта команда будет храниться во время исполнения.

Регистр адреса микрокоманд (РАМК) является синхронным, и мы будем тактировать его некоторым сигналом $d\overline{H}$. Необходимо так же учитывать состояние операционного блока, то есть его флаги. По адресу микрокоманды из памяти микропрограмм извлекается микрокоманда.

Автоматы с программной логикой являются микропроцессорными системами, в которых алгоритм функционирования реализован программным путем. Алгоритм программы для такого автомата строится следующим образом ? каждая вершина графа автомата заменяется группой блоков, в которых формируются выходные сигналы и анализируются входящие, стрелки, соединяющие вершины графа автомата заменяются командами условного или безусловного перехода.

51. Управляющий автомат с жесткой логикой.

В управляющих автоматах на элементах с жесткой логикой - для хранения информации о состоянии используется набор триггеров, на их тактовый вход подается тактирующий сигнал, входные сигналы X подаются на комбинационные устройства, вырабатывающие сигналы управления для (функции возбуждения) триггеров. Выходные сигналы Yформируются при помощи других комбинационных схем из выходных сигналов триггеров A (для автомата Мили) или из выходных сигналов триггеров Аи входных сигналов Y (для автомата Мура, именно его блок схема представлена на рисунке).

Схемы с жесткой логикой, как правило, позволяют обеспечить наибольшее быстродействие из всех возможных методов построения цифрового автомата, однако при возрастании сложности реализуемых алгоритмов схемы автоматов с жесткой логикой очень быстро становятся более сложными , чем схемы автоматов с микропрограммным или программным управлением. Поэтому схемы автоматов с жесткой логикой в настоящее время используют только в том случае, когда требуется максимальное быстродействие.

52. Каноническая структура процессора.

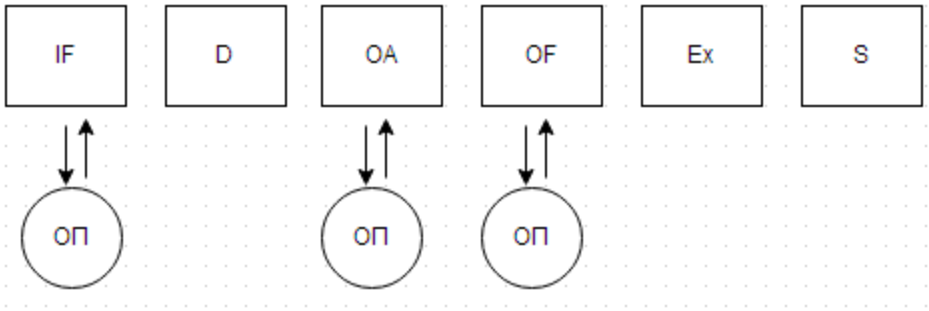
Процессор внутри себя содержит

- операционный автомат (внешний регистровый файл, ALU, MMX(работа с мультимедиа), FPU(плавающая точка), SSE (выполняет распараллеривание вычислительного процесса между данными, когда нужно произвести одну и ту же последовательность действий над разными данными)
- управляющий автомат (Control Unit, Bus Interface Unit, Prefetch Unit (блок предварительной выборки), Decode Unit, BTB (Branch Target Unit(Bufer) блок предсказания переходов), Memory Managment Unit (формирование адресов при обращении к основной памяти. MMU содержит в себе блок страничного механизма Page Unit + блок сегментного механизма Segment Unit

53. Цикл выполнения машинных команд и его фазы.

+ 1. Этап выборки команды (Instraction Fetch)
+ 2. Этап декодирования (Decode)
+/- 3. Этап формирования адреса операнда (OA)
+/- 4. Этап выборки операнда (OF - Operand Fetch)
+ 5. Этап выполнения предписанного действия (Execute)
+/- 6. Этап сохранения результата

54. Синхронный конвейер команд. Оценка его производительности.



Тц=Сумма(от 1 до M) tk. tк - длительность отдельного этапа выполнения машинной команды. M - число ступеней конвейера.
Быстродействие процессора P=1/Тц
Виртуальное быстродействие: Рв=1/d(дельта)t

Синхронный конвейер в принципе построен для идеальной ситуации

- длительность всех фаз работы конвейера одинакова
- выполнение вех машинных команд требует выполнения всех фаз

- при работе конвейера отсутствуют ситуации, связанные с перезагрузкой конвейера.

55. Причины снижения производительности при конвейерном режиме обработки команд.

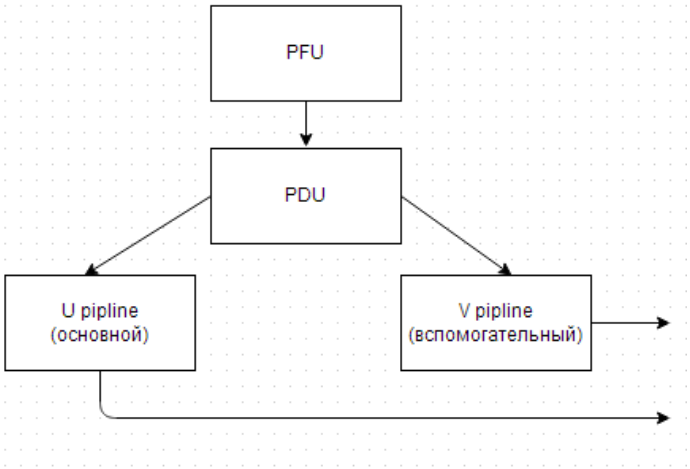
1) Наличие в программе конфликтов по управлению (нарушение естественного порядка следования команд)
2) Зависимости (конфликты) по данным (результат выполнения предыдущей команды является операндом для следующей команды)
3) Использование различными блоками конвейера одного и того же ресурса - структурные конфликты (память, например).
4) Наличие при выполнении программ особых случаев приводящих к прерыванию
5) Различное время выполнения отдельных фаз машинных команд.
6) Для ряда машинных команд требуется выполнение далеко не всех циклов машинной команды
7) Большой разброс фазы EX (выполнение) для различных машинных команд

56. Способы повышения производительности при конвейерной обработке команд.

1) Использование межблочных буферов позволяет сгладить длительность отдельных фаз различных команд
2) Использование нескольких потоков команд при многоальтернативной выборке
3) Использование блока (буфера) предсказания выполнения перехода
4) Использование циклического буфера команд

57. Особенности организации конвейера команд в процессорах Pentium. Структура процессора. Понятие суперскалярной архитектуры.

Под суперскалярной организацией понимают выполнение более одной машинной команды на каждом такте процессора.



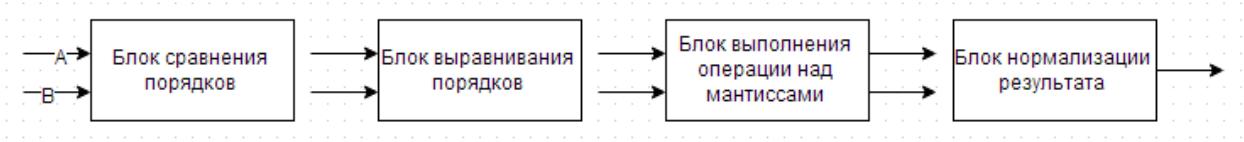
В основном блоке выполняются все основные команды, а во вспомогательном относительно простые операции при условии их спаривания с основными командами. Также команды в этих двух блоках должны быть независимы по данным.

В конвейерах выполняются:

- вторая фаза декодирования (вычисления адресов операндов в MEM)
- исполнение
- буферизация результата в буфере каждого из конвейеров

58. Понятие конвейера операций. Способы его организации.

Конвейер операций похож на конвейер команд, но здесь конвейеризации подвергаются отдельные фазы выполнения машинной команды. Чаще всего конвейер операций применяют для выполнения выражений с плавающей точкой.



Подобный конвейер операций оказывается эффективным при его полной загрузке. Это достижимо при обработке векторно-матричных операций. Поэтому использование конвейера операций характерно для векторных процессоров входящих в состав суперЭВМ.

59. Принцип многофункциональной обработки данных.

В основу многофункциональной обработки данных положен принцип независимости ряда выполняемых последовательных команд. Например, есть два выражения: A=B*C и Y=D+F. Если имеются отдельные блоки сложения/вычитания и блок умножения, то эти две операции можно совместить.

Многофункциональная обработка реализуется с помощью:

- разделением АЛУ на блоки по типу обрабатываемых данных
- разделением АЛУ на блоки по типу выполняемых операций для данных определенного типа
- использование нескольких АЛУ