VВопрос №1. Две формы представления информации.

Вычислительная машина – это техническое устройство, в котором информация об исходных данных решаемой задачи, правилах ее решения (алгоритме) и результатах вычислений должна задаваться в виде изменения каких-либо величин: углов поворота или перемещений, намагниченности материала, освещенности экрана или фотодатчика.

В прошлые века, когда человечество не знало об электрических и магнитных явлениях или еще не умело их использовать, наиболее доступной, а следовательно, и удобной были механическая форма представления информации в вычислительных устройствах. Однако механические устройства громоздки, дороги и слишком инерционны. Поэтому сейчас во всех вычислительных машинах в качестве основной формы представления информации служат электрические сигналы (чаще всего – напряжения постоянного тока). Для передачи электрических сигналов нужны лишь провода, эти сигналы легко преобразовать с помощью различных полупроводниковых схем.

При использовании в качестве носителя информации напряжений постоянного тока возможны две формы представления численного значения какой-либо переменной.

Первая форма представления информации – аналоговая (или непрерывная) – с помощью сходной величины (аналога). Величины, представленные в такой форме, могут принимать принципиально любые значения в каком-то диапазоне. Они могут быть сколь угодно близки друг к другу, малоразличимы, но все-таки, хотя бы в принципе, различимы. Количество значений, которое может принимать такая величина, бесконечно велико. Основное свойство таких величин – отсутствие разрывов, промежутков между значениями, которые может принимать данная аналоговая величина.

Вторая форма представления информации – цифровая (или дискретная) – с помощью набора напряжений, каждое из которых соответствует одной из цифр представляемой величины. Такие величины, принимающие не все возможные, а лишь вполне определенные значения, называются дискретными (прерывистыми). В отличие от непрерывной величины количество значений дискретной величины всегда будет конечным.

При использовании непрерывной формы создателю вычислительной машины потребуется меньшее число устройств, но они будут сложнее. Устройства для обработки непрерывных сигналов обладают высокой «квалификацией» (они могут интегрировать сигнал, выполнять любое его функциональное преобразование и т.п.) и за счет этого, а также ряда других особенностей имеют высокое быстродействие.

Однако из-за сложности технической реализации устройств для логических операций с непрерывными сигналами, длительного хранения таких сигналов, их точного измерения подобная форма представления в основном используется в аналоговых вычислительных машинах (АВМ). Эти машины предназначены для решения задач, описываемых системами дифференциальных уравнений.

Но АВМ не могут решать задачи, связанные с хранением и обработкой больших объемов информации, которые легко решаются при использовании цифровой (дискретной) формы представления информации, реализуемой цифровыми электронными вычислительными машинами (ЭВМ).

Вопрос №2. Способы представления дискретной информации.

Каждое значение из набора исходных данных задачи, результатов ее решения может быть представлено в ЭВМ в виде нескольких электрических сигналов, один из которых соответствует числу единиц в значении, другой – числу десятков, третий – числу сотен и т.д. Однако такое представление не является наилучшим с технических позиций. Устройство, предназначенное для обработки подобных сигналов, должно различать в каждом из них десять состояний. Значительно проще построить устройство, которое различало бы всего два состояния (его наличие или отсутствие). Это тем более целесообразно, т.к. существующие сейчас дешевые устройства для ввода данных в ЭВМ также кодируют отдельные составляющие вводимой информации с помощью двух состояний.

Это обстоятельство натолкнуло создателей первых ЭВМ на применение другой системы счисления при внутреннем представлении чисел в машинах: вместо привычной десятичной системы счисления была взята двоичная. 2СС также является позиционной СС, т.е. в ней значение каждой цифры зависит от позиции этой цифры в записи числа.

Существуют специальные термины, широко используемые в вычислительной технике: бит, байт и слово.

Двоичный разряд называют битом.

Байт – восьмибитовая единица.

ЭВМ содержит большое количество ячеек памяти и регистров для хранения двоичной информации. Большинство этих ячеек имеет одинаковую длину n, т.е. они используются для хранения n бит двоичной информации. Информация, хранимая в такой ячейке, называется словом.

Вопрос №3. Системы счисления, используемые в вычислительной технике. Представление целых, символьных данных, данных с плавающей запятой. Прямой, обратный и дополнительный код.

Удобная для использования в ЭВМ двоичная система счисления совсем неудобна для записи и чтения чисел человеком. Для сокращения трудоемкости ручной обработки кодов чисел, команд широко применят 8- и 16СС. В 8СС используется 8 цифр (0-7), в 16СС – 10 цифр и 6 прописных букв (0-9, A-F). Т.к. основанием 8СС является $8=2^3$, то для перевода двоичных чисел в восьмеричные необходимо разделить двоичные числа на триады. Каждую из таких групп можно представить одной восьмеричной цифрой. Аналогичным образом осуществляется перевод двоичных чисел в шестнадцатеричные. Только в этом случае двоичное число разбивается на 4 тетрады, которые и представляются одной шестнадцатеричной цифрой.

Наконец, следует упомянуть и о двоично-десятичной СС, которая широко используется в цифровых устройствах, где основная часть операций связана не с обработкой и хранением вводимой информации, а с самим ее выводом на какие-либо индикаторы с десятичным представлением полученных результатов (калькуляторы, кассовые аппараты). В 2-10СС десятичные цифры от 0 до 9 представляют 4-разрядными двоичными комбинациями от 0000 до 1001. Две двоично-десятичные цифры составляют 1 байт (можно представлять значения от 0 до 99).

Целые числа являются простейшими числовыми данными, с которыми оперирует ЭВМ. Для целых чисел существует два представления: беззнаковое (только для неотрицательных целых чисел) и со знаком. Отрицательные числа можно представлять только в знаковом виде. Целые числа хранятся в компьютере в формате с фиксированной запятой.

Для беззнакового представления все разряды ячейки отводятся под представление самого числа. Для представления со знаком старший разряд отводится под знак числа, остальные – под само число. Такое представление называется прямым кодом двоичного числа. Положительные числа всегда представляются с помощью прямого кода. Для представления отрицательных чисел в ЭВМ используется дополнительный код. Он используется для упрощения выполнения арифметических операций.

Для представления вещественных чисел принят способ представления с плавающей запятой. Этот способ представления опирается на нормализованную запись действительных чисел. При представлении чисел с плавающей запятой часть разрядов ячейки отводится для записи порядка числа, остальные разряды – для записи мантиссы.

Вопрос №4. Базовые элементы вычислительной техники: ячейки, регистры, шины, вентили, тактовые генераторы.

Ячейка памяти – минимальный адресуемый элемент запоминающего устройства ЭВМ. Ячейки памяти имеют адрес (порядковый номер, число), по которому к ним могут обращаться команды процессора.

Регистр процессора – память внутри процессора, предназначенная для хранения адресов и промежуточных результатов вычислений или данных, необходимых для работы самого процессора.

Ячейки памяти состоят из элементов, которые могут находиться в одном из двух устойчивых состояний: конденсатор заряжен или разряжен, транзистор находится в проводящем или непроводящем состоянии. Одно из таких физических состояний создает высокий уровень выходного напряжения элемента памяти, а другое – низкий. Первое обычно принимается за двоичную 1, а второе – за двоичный 0. Возможно и обратное кодирование. Хотя переход от 0 к 1 и от 1 к 0 происходит не мгновенно, однако в определенные моменты времени этот сигнал достигает значений, которые воспринимаются элементами ЭВМ как 0 или 1.

Регистр характеризуется единственным числом: количеством битов, которые могут в нем храниться. Операция чтения информации, хранимой в регистре, сводится к созданию копии его содержимого, оригинал же сохраняется в регистре без изменений.

Электрическая цепь, соединяющая регистр с другим регистром или иным устройством ЭВМ, называется шиной. Шина состоит из параллельных проводов, каждый из которых предназначен для передачи соответствующего регистра. Также шина содержит несколько дополнительных проводов, используемых для передачи сигналов синхронизации и управления. Шины служат для передачи информации лишь в направлении, обозначенном стрелкой на шине. Специальные схемы позволяют в одни моменты времени передавать информацию по шине в одну сторону, а в другие – в обратном направлении, т.е. организовать двунаправленную шину.

Вентильные схемы – это электронные ключевые схемы, предназначенные для управления потоком информации из регистров в шины и обратно. Такая схема содержит два входа и один выход. На один вход подается информационный сигнал (данные с регистра), а на другой – управляющий. Если управляющий сигнал равен 1, то данные проходят схему без препятствий, если 0 – никакая информация не пройдет через схему. Для подачи информационного сигнала на вход вентильной схемы обычно используется многопроводная шина. Для передачи выходного сигнала требуется шина с таким же количеством проводов. Если управляющий сигнал равен 1, то информационные сигналы на входной и выходной шинах совпадают.

Тактовые импульсы (вырабатываемые генератором тактовых импульсов ЭВМ) используется для синхронизации процессов передачи информации между устройствами ЭВМ.

Вопрос №5. Базовые элементы вычислительной техники: логические схемы, триггеры, регистры, счетчики, сумматоры.

Функциональная логическая схема представляет собой совокупность логических элементов (простейшее устройство ЭВМ, выполняющее одну определённую логическую операцию над входными сигналами согласно правилам алгебры логики) и связей между ними.

Триггер — класс электронных устройств, обладающих способностью длительно находиться в одном из двух устойчивых состояний и чередовать их под воздействием внешних сигналов. Каждое состояние триггера легко распознаётся по значению выходного напряжения. Отличительной особенностью триггера как функционального устройства является свойство запоминания двоичной информации. Под памятью триггера подразумевают способность оставаться в одном из двух состояний и после прекращения действия переключающего сигнала.

Регистр процессора – память внутри процессора, предназначенная для хранения адресов и промежуточных результатов вычислений или данных, необходимых для работы самого процессора. Регистр характеризуется единственным числом: количеством битов, которые могут в нем храниться. Операция чтения информации, хранимой в регистре, сводится к созданию копии его содержимого, оригинал же сохраняется в регистре без изменений.

Счётчик числа импульсов — устройство, на выходах которого получается двоичный (двоично-десятичный) код, определяемый числом поступивших импульсов. Основной параметр счётчика — модуль счёта — максимальное число единичных сигналов, которое может быть сосчитано счётчиком.

Сумматор — устройство, преобразующее информационные сигналы (аналоговые или цифровые) в сигнал, эквивалентный сумме этих сигналов.

Вопрос №6. Структура и принцип функционирования ЭВМ.

Типичная ЭВМ состоит из процессора, памяти и устройств ввода-вывода.

«Сердцем» ЭВМ является процессор, в состав которого входят устройство управления выборкой команд из памяти и их выполнением, арифметико-логическое устройство, производящее операции над данными, регистры, осуществляющие временное хранение данных и состояний процессора, схемы для управления и связи с подсистемами памяти и ввода-вывода.

Устройство ввода обеспечивает считывание информации с определенных носителей информации и ее представление в форме электрических сигналов, воспринимаемых другими устройствами ЭВМ.

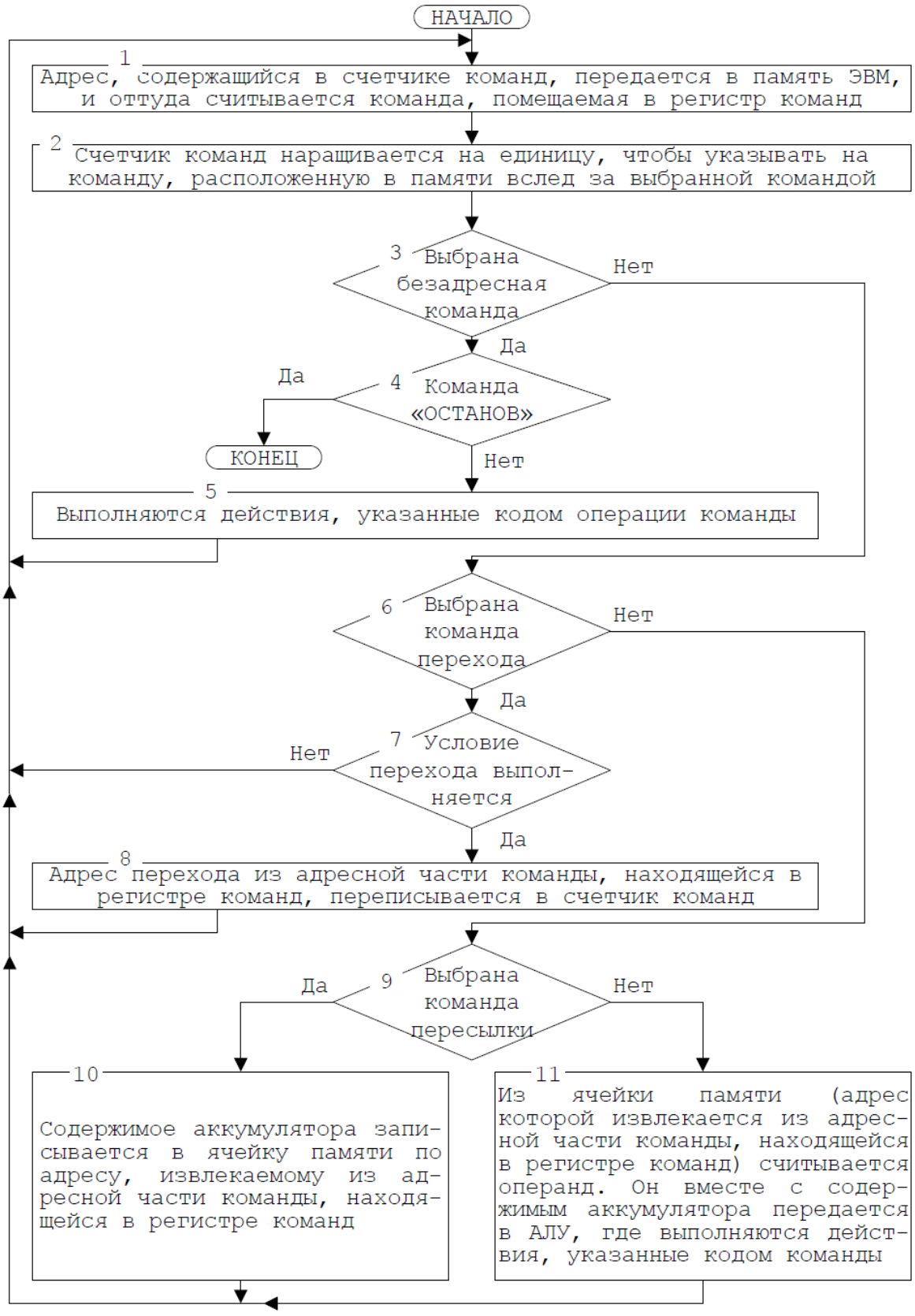
Устройства вывода представляют результаты обработки информации в форме, удобной для визуального восприятия.

Память ЭВМ включает устройство, обеспечивающее хранение команд и данных. Это устройство состоит из блоков одинакового размера – ячеек памяти, предназначенных для хранения одного слова информации. Ячейка памяти состоит из элементов памяти, состояние каждого из которых соответствует одной двоичной цифре. Совокупность нулей и единиц, хранящихся в элементах одной ячейки, представляет собой содержимое этой ячейки памяти.

В микроЭВМ используются безадресные, одноадресные и реже двухадресные команды. В одноадресных командах один из операндов выбирается из специального регистра – аккумулятора. В него же заносится и результат операции. Безадресные команды или задают какое-либо действие с устройствами ЭВМ, или используются для работы с операндами, имеющими фиксированное расположение (чаще всего с аккумулятором).

В процессе работы ЭВМ последовательно выполняет набор достаточно простых операций: выборку команды, определение ее типа, исполнение команды и определение адреса следующей команды.

Вопрос №7. Порядок функционирования простого процессора.



Вопрос №8. Операционная система Unix — интерпретаторы и стандартные потоки ввода вывода.

Традиционный способ взаимодействия пользователя с системой UNIX основывается на использовании командных языков (хотя, в настоящее время все большее распространение получают графические интерфейсы). После входа пользователя в систему для него запускается один из командных интерпретаторов (в зависимости от параметров, сохраняемых в файле /etc/passwd). Обычно в системе поддерживается несколько командных интерпретаторов с похожими, но различающимися своими возможностями командными языками. Общее название для любого командного интерпретатора ОС UNIX - shell (оболочка), поскольку любой интерпретатор представляет внешнее окружение ядра системы.

Вызванный командный интерпретатор выдает приглашение на ввод пользователем командной строки. После выполнения очередной командной строки и выдачи на экран терминала или в файл соответствующих результатов, shell снова выдает приглашение на ввод командной строки, и так до тех пор, пока пользователь не завершит свой сеанс работы путем ввода команды logout.

Командные языки, используемые в ОС UNIX, достаточно просты, чтобы новые пользователи могли быстро начать работать, и достаточно мощны, чтобы можно было использовать их для написания сложных программ. Последняя возможность опирается на механизм командных файлов (shell scripts), которые могут содержать произвольные последовательности командных строк. При указании имени командного файла вместо очередной команды интерпретатор читает файл строка за строкой и последовательно интерпретирует команды.

По умолчанию каждому процессу при запуске ставится в соответствие три открытых файла: стандартного ввода, стандартного вывода и стандартного вывода ошибок. С помощью средств командной строки такие потоки для разных процессов могут быть объединены так, что, к примеру, вывод одного процесса будет подаваться на ввод другого. В более общем смысле такие потоки называют неименованными каналами.

Вопрос №9. Операционная система Unix — основные команды и права файлов.

cd - Изменяет текущий рабочий каталог. Синтаксис: cd <directory>. Аргумент <directory> – это каталог, в который необходимо перейти (’.’ ссылается на текущий каталог, ‘..’ – на родительский каталог).

ls - Выдает информацию о файлах в каталоге. Синтаксис: ls <file1>... <fileN>. Аргументы <file1> ... <fileN> – это имена файлов или каталогов, информацию про которые надо выдать. Наиболее часто используемые опции: -F (для представления информации о типах файлов), и –l (выдает в длинном (long) формате информацию о размерах файлов, владельцах, правах доступа и т.д.).

cp - Копирует файл(ы) в файл или каталог. Синтаксис: cp <file1>... <fileN><destination>. Аргументы <file1> ... <fileN> – это имена копируемых файлов, а <destination> файл или каталог, в который копируют.

mv - Перемещает файл(ы) в другой файл или каталог. Эта команда не эквивалентна копированию с последующим уничтожением оригинала. Она может быть использована для переименования файлов. Синтаксис: mv<file1> ...<fileN><destination>. Аргументы <file1> ... <fileN> – это имена перемещаемых файлов, а <destination> имя файла или каталога, в который перемещают.

rm - Удаляет файлы. Синтаксис: rm <file1>... <fileN>. Аргументы <file1>... <fileN> – это имена удаляемых файлов. Опции: -i потребует подтверждения перед удалением файла.

mkdir - Создает новые каталоги. Синтаксис: mkdir <dir1> .... <dirN>. Аргументы <dir1> ... <dirN> – это создаваемые каталоги.

rmdir - Удаляет пустые каталоги. Синтаксис: rmdir <dir1> ... <dirN>. Аргументы <dir1> ... <dirN> – это удаляемые каталоги.

more - Выдает содержимое названных файлов поэкранно. Синтаксис: more <file1> ... <fileN>. Аргументы <file1> ... <fileN> – это отображаемые файлы.

cat - Используется для конкатенации файлов, а также для выдачи полного содержания файла разом. Синтаксис: cat <file1> ... <fileN>. Аргументы <file1> ... <fileN> – это выдаваемые файлы.

echo - Выводит на экран аргументы. Синтаксис: echo <arg1> ... <argN>. Аргументы <arg1> ... <argN> – это выдаваемые на экран аргументы.

grep - Выдает все строки в названном файле(лах), которые содержат заданный образец. Синтаксис: grep <pattern> <file1> ... <fileN>. Аргумент <pattern> – это образец (представленный регулярным выражением) и <file1>... <fileN> - файлы, в которых производится поиск.

UNIX различает три типа доступа к файлам: чтение из файла, запись в файл, выполнение файла. Кроме того, UNIX различает три категории пользователей, каждая из которых имеет свои собственные ограничения по любому из типов доступа к файлу. Этими категориями пользователей являются: владелец , члены группы (член группы, к которой принадлежит владелец файла) и прочие пользователи.

Структура кода защиты следующая: -rwx rwx rwx (разрешение на доступ к файлу для прочих пользователей, разрешение на доступ к файлу для членов группы, разрешение на доступ к файлу для владельца).

Соответствующие символы имеют следующее назначение:

r - разрешение на доступ к файлу по чтению; если вместо символа r стоит символ -, то разрешения на доступ к файлу по чтению нет;

w - разрешение на доступ к файлу по записи; если вместо символа w стоит символ -, то разрешения на доступ к файлу по записи нет;

x - разрешение на доступ к файлу по выполнению; если вместо символа x стоит символ -, то разрешения на доступ к файлу по выполнению нет.

Для файлов, являющихся каталогами, код защиты имеет несколько иной смысл, чем для обычных файлов:

d - файл является каталогом; если вместо символа d стоит символ -, то это обычный файл;

r - разрешение осуществлять вывод на терминал содержимого каталога;

w - разрешение создавать файлы в каталоге и удалять из него;

x - разрешение на доступ к файлам, содержащимся в каталоге, и на установку каталога в качестве текущего каталога с помощью команды cd.

Изменить код защиты файла, владельцем которого Вы являетесь, можно с помощью команды chmode, изменить владельца файла - командой chown, изменить идентификатор группы файла - командой chgrp.

Вопрос №10. Базы данных — предназначение и состав.

База данных — организованная в соответствии с определёнными правилами и поддерживаемая в памяти компьютера совокупность данных, характеризующая актуальное состояние некоторой предметной области и используемая для удовлетворения информационных потребностей пользователей.

Система управления базами данных (СУБД) — совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

Данные в БД логически структурированы (систематизированы) с целью обеспечения возможности их эффективного поиска и обработки в вычислительной системе.

Структурированность подразумевает явное выделение составных частей (элементов), связей между ними, а также типизацию элементов и связей, при которой с типом элемента (связи) соотносится определённая семантика и допустимые операции.

БД включает схему, или метаданные, описывающие логическую структуру БД в формальном виде (в соответствии с некоторой метамоделью). Схема включает в себя описания содержания, структуры и ограничений целостности, используемые для создания и поддержки базы данных. База данных включает в себя набор постоянных данных, определенных с помощью схемы. Система управления данными использует определения данных в схеме для обеспечения доступа и управления доступом к данным в базе данных.

Вопрос №11. Базы данных — таблицы и язык запросов.

В 70 гг. XX в. была предложена реляционная (relation — отношение, связь) или иначе табличная модель данных, обладающая следующими основными свойствами:

1. Данные воспринимаются пользователями как таблицы (и никак иначе).

2. Каждая таблица состоит из однотипных строк и имеет уникальное имя.

3. Строки имеют фиксированное число полей (столбцов) и значений (множественные поля и повторяющиеся группы недопустимы). Иначе говоря, в каждой позиции таблицы на пересечении строки и столбца всегда имеется в точности одно значение или ничего.

4. Строки таблицы обязательно отличаются друг от друга хотя бы единственным значением, что позволяет однозначно идентифицировать любую строку такой таблицы.

5. Столбцам таблицы однозначно присваиваются имена, и в каждом из них размещаются однородные значения данных (даты, фамилии, целые числа или денежные суммы).

6. Полное информационное содержание базы данных представляется в виде явных значений данных и такой метод представления является единственным. В частности, не существует каких-либо специальных "связей" или указателей, соединяющих одну таблицу с другой.

7. При выполнении операций с таблицей ее строки и столбцы можно обрабатывать в любом порядке безотносительно к их информационному содержанию. Этому способствует наличие имен таблиц и их столбцов, а также возможность выделения любой их строки или любого набора строк с указанными признаками.

SQL (Structured Query Language — «язык структурированных запросов») — универсальный компьютерный язык, применяемый для создания, модификации и управления данными в реляционных базах данных. Основные команды:

- SELECT - используется для извлечения данных из таблиц.
- UPDATE – для внесения изменений в существующие данные в таблицах.
- INSERT – предназначена для добавления данных в таблицы.
- DELETE – для удаления данных из таблиц.
- CREATE – используется для создания объектов базы данных.
- ALTER – для внесения изменений в определения объектов и параметры базы данных.
- DROP – используется для удаления объектов из базы данных.

Вопрос №12. Состав и структура БЭВМ.

Базовая ЭВМ – это простая гипотетическая машина, обладающая типичными чертами многих конкретных ЭВМ.

Память состоит из 2048 ячеек (16-битовых) с адресами 0, 1, …, 2046, 2047. Восемь ячеек памяти с адресами 008-00F называются индексными и их лучше использовать в циклических программах.

Процессор состоит из ряда регистров (РК, А, РД, СК, РА, С), арифметико-логического устройства и устройства управления.

Счетчик команд служит для организации обращений к ячейкам памяти, в которых хранятся команды программы. После исполнения любой команды СК указывает адрес ячейки памяти, содержащий следующую команду. Имеет 11 разрядов.

Регистр адреса – 11-разрядный регистр, содержащий значение исполнительного адреса (адреса ячейки памяти, к которой обращается ЭВМ за командой или данными).

Регистр команд – 16-разрядный регистр, который используется для хранения кода команды, непосредственно выполняемой машиной.

Регистр данных используется для временного хранения 16-разрядных слов при обмене информацией между памятью и процессором.

Аккумулятор – 16-разрядный регистр, являющийся одним из главных элементов процессора. Машина может одновременно выполнять арифметические и логические операции только с одним или двумя операндами. Один из операндов находится в аккумуляторе, а второй – в регистре данных. Результат помещается в А.

Регистр переноса – это 1-разрядный регистр, выступающий в качестве продолжения аккумулятора и заполняющийся при переполнении А. Этот регистр используется при выполнении сдвигов.

АЛУ может выполнять такие арифметические операции, как сложение и вычитание с учетом переноса, полученного в результате выполнения предыдущей операции. Кроме того, оно способно выполнять операции логического умножения, инвертирования, циклического сдвига и наращивания аккумулятора на 1.

<u>Вопрос №13. Система команд БЭВМ, форматы команд.</u>

ЭВМ способна понимать и выполнять точно определенный набор команд. При составлении программы пользователь ограничен этими командами. В зависимости от того, к каким блокам базовой ЭВМ обращается команда или на какие блоки она ссылается, команды можно разделить на три группы: обращения к памяти (адресные команды), обращения к регистрам (регистровые или безадресные команды), команды ввода-вывода.

Команды обращения к памяти предписывают машине производить действия с содержимым ячейки памяти, адрес которой указан в адресной части команды.

Безадресные команды выполняют различные действия без ссылок на ячейку памяти.

Команды ввода-вывода осуществляют обмен данными между процессором и внешними устройствами ЭВМ.

Разработчики базовой ЭВМ выбрали три формата 16-битовых (однословных) команд с 4-битовым кодом операции.

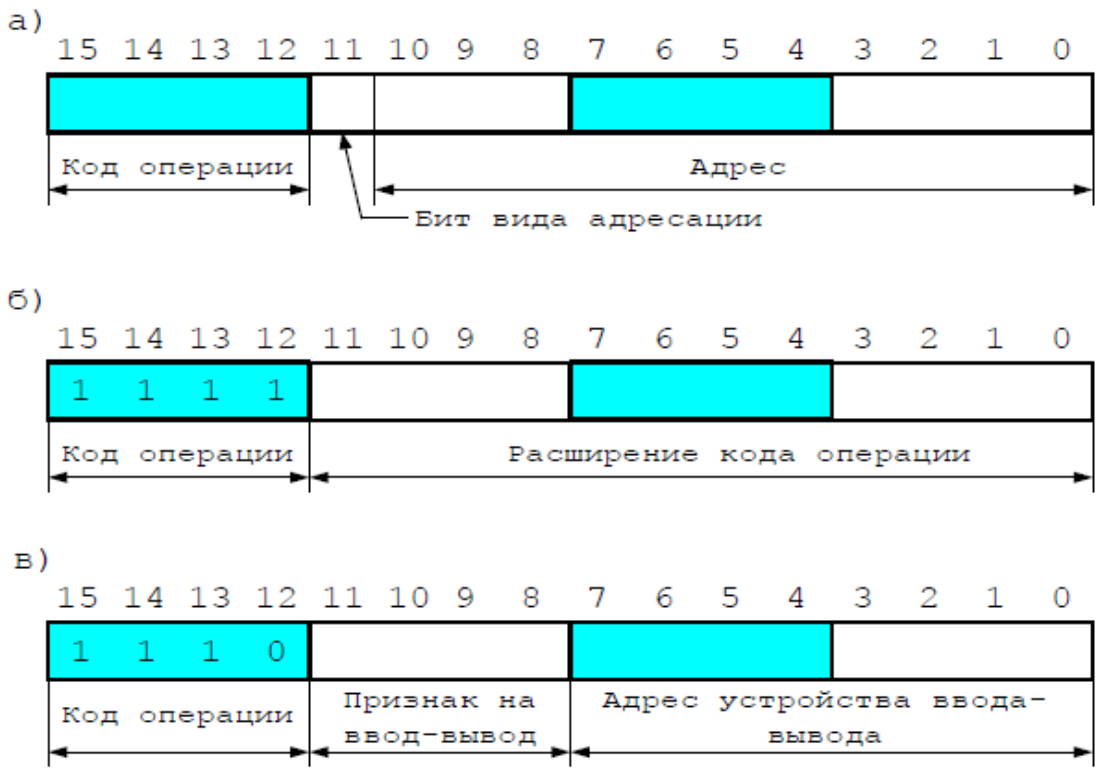


Рис. 2.3. Форматы команд базовой ЭВМ
а) адресных, б) безадресных, в) ввода-вывода

В командах обращения к памяти на адрес отведено 11 бит. Следовательно, можно прямо адресоваться к 2^{11}=2048 ячейкам памяти, т.е. ко всей памяти базовой ЭВМ (прямая адресация). В этом случае бит вида адресации должен содержать 0. Если же в этом бите установлена 1, то адрес, размещенный в адресной части команды, указывает на ячейку, в которой находится адрес операнда (косвенная адресация).

Система команд базовой ЭВМ

Наименование	Мнемоника	Код	Описание
Адресные команды			
Логическое умножение	ADD M	1XXX	(M) & (A) → A
Пересылка	MOV M	3XXX	(A) → M
Сложение	ADD M	4XXX	(M) + (A) → A
Сложение с переносом	ADC M	5XXX	(M) + (A) + (C) → A
Вычитание	SUB M	6XXX	(A) - (M) → A
Переход, если перенос	BCS M	8XXX	Если (C)=1, то M → СК
Переход, если плюс	BPL M	9XXX	Если (A)>=0, то M → СК
Переход, если минус	BMI M	AXXX	Если (A)<0, то M → СК
Переход, если ноль	BEQ M	BXXX	Если (A)=0, то M → СК
Безусловный переход	BR M	CXXX	M → СК
Приращение и пропуск	ISZ M	0XXX	(M)+1 → M; Если (M)>=0, то (СК)+1 → СК
Обращение к п/программе	JSR M	2XXX	(СК) → M; M+1 → СК
Безадресные команды			
Очистка аккумулятора	CLA	F200	0 → A
Очистка рег. переноса	CLC	F300	0 → C
Инверсия аккумулятора	CMA	F400	($\overline{A}$) → A
Инверсия рег. переноса	CMC	F500	($\overline{C}$) → C
Циклический сдвиг влево на 1 разряд	ROL	F600	Содержимое A и C сдвигается влево, A(15) → C и C → A(0)
Циклический сдвиг вправо на 1 разряд	ROR	F700	Содержимое A и C сдвигается вправо, A(0) → C и C → A(15)
Инкремент аккумулятора	INC	F800	(A) + 1 → A
Декремент аккумулятора	DEC	F900	(A) - 1 → A
Останов	HLT	F000	
Нет операции	NOP	F100	
Разрешение прерывания	EI	FA00	
Запрещение прерывания	DI	FB00	
Команды ввода-вывода			
Очистка флага	CLF B	E0XX	0 → флаг устройства B
Опрос флага	TSF B	E1XX	Если флаг устройства B равен 1, то (СК) + 1 → СК
Ввод	IN B	E2XX	(B) → A
Вывод	OUT B	E3XX	(A) → B

Условные обозначения:

(M),(A),(СК),(C) и (B) – содержимое ячейки с адресом M, аккумулятора, счетчика команд, регистра переноса и регистра данных устройства ввода-вывода с адресом B;

A(X) – разряд аккумулятора с номером X;

XXX – адрес ячейки памяти;

XX – адрес устройства ввода-вывода.

Вопрос №14. Организация вычислений в БЭВМ. Сдвиги, арифметические и логические операции.

Целые двоичные числа без знака можно использовать для представления нуля и целых положительных чисел. При размещении таких чисел в одном 16-разрядном слове они могу изменяться от 0 до 65535. Подобные числа относятся к числам с фиксированной запятой. Целые двоичные числа со знаком используются тогда, когда необходимо различать положительные и отрицательные числа. Отрицательные числа представляются в дополнительном коде. Это упрощает конструкцию ЭВМ.

Сложение целых двоичных чисел со знаком и без знака выполняется в базовой ЭВМ с помощью команды ADD. По команде INC к содержимому аккумулятора прибавляется единица, а по команде DEC – единица вычитается. Если при этом возникает перенос из старшего разряда А, то в регистр переноса заносится 1, в противном случае в него заносится 0.

Вычитание может выполняться путем сложения уменьшаемого и дополнительного кода вычитаемого.

В базовой ЭВМ нет команд для выполнения умножения и деления (АЛУ не выполняет таких операций), поэтому произведение и частное необходимо получать программным путем.

Для изменения знака числа необходимо его инвертировать, а затем прибавить единицу к младшему разряду.

Побитовая обработка данных обеспечивается командами логического умножения, циклических сдвигов, а также командами инвертирования и очистки регистра переноса.

Команда AND выполняет над каждым разрядом аккумулятора и содержимым ячейки булеву операцию «И». Результат выполнения команды для каждой пары битов операндов равен 1 только тогда, когда оба бита равны 1, а в остальных случаях бит результата равен 0, т.е. команда позволяет выделять или очищать определенные биты слова.

Команды ROL и ROR замыкают аккумулятор и регистр переноса в кольцо и сдвигают все биты кольца влево или вправо. Сдвигами числа можно реализовать операции умножения или деления на 2 (один сдвиг), 4 (два сдвига), 8 (три сдвига) и т.д.

Вопрос №15. Управление вычислительным процессом в БЭВМ. Подпрограммы в БЭВМ. Стек.

Задача управления вычислительным процессом, т.е. требуемой последовательностью выполнения команд, решается в базовой ЭВМ при помощи команд переходов, команд «Приращение и пропуск» и «Останов». Команды переходов не изменяют состояние аккумулятора и регистра переноса, а лишь изменяют содержимое СК, поместив в него адрес, определяемый адресной частью команды.

BCS (переход, если перенос), BPL (переход, если плюс), BMI (переход, если минус), BEQ (переход, если ноль), BR (безусловный переход).

Команды перехода широко используются для организации циклических программ, которые используются в тех случаях, когда требуется несколько раз выполнить набор одинаковых действий с различными наборами данных.

Если произвести косвенное адресацию какой-либо из индексных ячеек, то сначала ее содержимое будет использовано в качестве адреса операнда, а затем оно автоматически увеличится на единицу.

Достаточно часто встречаются ситуации, когда отдельные части программы должны выполнить одни и те же действия по обработке данных. В подобных случаях повторяющиеся части программы выделяют в подпрограмму. В базовой ЭВМ для этой цели используется команда JSR. При оформлении подпрограммы перед ее первой командой следует разместить ячейку, в которую будет пересылаться адрес возврата из подпрограммы. В команде обращения к подпрограмме указывается адрес именно этой ячейки (например, адрес M в команде JSR M). Последней командой подпрограммы должна быть команда выхода (команда BR (M)). По ней осуществляется переход к команде, адрес которой сохраняется в первой ячейке подпрограммы.

Вопрос №16. Порядок выполнения машинных команд. Пример на основе адресной команды с косвенной адресацией.

В процессе исполнения команд устройство управления ЭВМ производит анализ и пересылку команд, отдельных ее частей (кода операции, признака адресации и адреса) или операнда из одного регистра ЭВМ в другой ее регистр, АЛУ, память или устройство ввода-вывода. Эти действия (микрооперации) протекают в определенной временной последовательности и скоординированы между собой. Для обеспечения такой последовательности в ЭВМ используется ГТИ.

Для реализации одной команды требуется выполнить определенное количество микрокоманд, каждая из которых инициируется одним тактовым импульсом. Общее число тактовых импульсов, требуемых для выполнения команды, определяет время ее выполнения, называемое циклом команды. Цикл команды обычно включает один или несколько машинных циклов. Устройство управления базовой ЭВМ может находиться в 4 возможных состояниях: выборки команды, выборки адреса, исполнения, прерывания.

Выборка команды: **1)** СК -> РА, **2)** ОП(РА) -> РД, СК + 1 -> СК, **3)** РД -> РК, **4)** Определение типа команды, вида адресации, **5*)** Выполнение безадресных команд и команд ввода-вывода.

Выборка адреса следует за циклом выборки для адресных команд с косвенной адресацией: **1)** РД -> РА, **2)** ОП(РА) -> РД. Если косвенно адресуется одна из индексных ячеек, то данный цикл продолжается: **3)** РД + 1 -> РД, **4)** РД -> ОП(РА), **5)** РД – 1 -> РД.

Последовательность действий, выполняемых в цикле исполнения, определяется типом выполняемой адресной команды.

Вопрос №17. Организация ввода-вывода в вычислительных системах. Программно-управляемая передача информации.

К ЭВМ можно подключать большое число разнообразных устройств ввода-вывода или внешних устройств (ВУ). Эти устройства передают в ЭВМ и получают из нее большой объем информации, который не может быть размещен только в регистрах процессора. Поэтому информация передается из ВУ в память ЭВМ и поступает на ВУ из ее памяти. При этом обмен может идти под управлением программы ЭВМ через регистры процессора (программно-управляемая передача данных) или под управлением специального внешнего устройства (контроллера прямого доступа в память), минуя процессор (передача данных при прямом доступе к памяти). Программно-управляемый обмен осуществляется малыми порциями, при прямом доступе к памяти информация передается большими блоками.

При использовании программно-управляемого обмена должна быть составлена программа, обеспечивающая пересылку данных из памяти ЭВМ в аккумулятор и далее в регистр памяти контроллера ВУ (вывод данных) или из регистра данных контроллера ВУ в аккумулятор и затем в память ЭВМ (ввод данных). В этой программе можно реализовать один из трех типов обмена: синхронный, асинхронный и по прерыванию.

Вопрос №18. Организация ввода-вывода в БЭВМ. Устройства ввода-вывода, команды.

В базовой ЭВМ реализована только программно-управляемая передача данных, т.е. обмен данными происходит только под управлением программы ЭВМ через регистры процессора.

В БЭВМ используются простейшие внешние устройства: два устройства ввода (ВУ-1, ВУ-2) и одно устройство вывода (ВУ-3). Устройства представлены 8-разрядными регистрами данных. Через регистры данных ВУ-1 и ВУ-2 информация может быть введена в базовую ЭВМ, а в регистр данных ВУ-3 принята из базовой ЭВМ.

Между ВУ и процессором установлены простейшие контроллеры, каждый из которых содержит: регистр данных (для обмена данными между ВУ и процессором), дешифратор адреса (позволяющий выделить обращение к данному ВУ среди всех обращений у устройствам ввода-вывода, подключенным к процессору), дешифратор приказов (декодирующий приказ от процессора на выполнение тех или иных операций) и регистр состояния (в котором хранится информация о готовности ВУ к обмену данными с процессором). В контроллерах простейших ВУ обычно используется однобитовые регистры состояния, которые часто называют флагом или флажком. Контроллеры ВУ связаны с процессором шинами ввода и вывода информации, шиной адреса и шиной управления, по которым передаются приказы от процессора и сведения о состоянии ВУ.

Команды ввода-вывода БЭВМ имеют одинаковый формат: 4-разрядое поле операции, 4-разрядное поле кода приказа, 8-разрядное поле кода выборки устройства ввода-вывода. Код операции $(1110)_2$ служит для отличия этих команд от других команд ЭВМ. Между собой они различаются кодом приказа: пересылка данных (IN B – пересылка содержимого РД контроллера ВУ с адресом B в A, OUT B – пересылка содержимого A в РД контроллера ВУ с адресом B), проверка готовности ВУ (TSF B) и сброс состояния готовности (CLF B). Адрес позволяет связать процессор с одним из подключенных к нему ВУ (их может быть до 256).

Вопрос №19. Организация асинхронного обмена в БЭВМ. Пример программы.

Программа такого обмена строится так: сначала проверяется готовность ВУ к обмену и если оно готово, то дается команда на обмен. ВУ сообщает о готовности установкой флага. При асинхронном обмене ЭВМ должна тратить время на ожидание момента готовности, а так как готовность проверяется командным путем (команда TSF), то в это время ЭВМ не может выполнять никакой другой работы по преобразованию данных.

Программа ввода кода двух символов

Адрес	Содержимое		Комментарии
	Код	Мнемоника	
5	FFF8		Константа -8, используется для сдвига
6	0000		Ячейка для записи «Да»
	…		
20	E101	TSF 1	Опрос флага контроллера ВУ-1 и повторение этой
21	C020	BR 20	операции, если ВУ не готово к обмену (флаг=0)
22	E201	IN 1	Это действие осуществляется лишь после нажатия кнопки ГОТОВ у ВУ-1, т.е. когда при выполнении TSF 1 выясняется, что флаг=1 и пропускается BR 20. По коман-де IN 1 содержимое регистра данных контроллера ВУ-1 пересылается в восемь младших
23	E001	CLF 1	разрядов аккумулятора
24	F600	ROL	Сброс готовности ВУ-1 для предотвращения считывания кода с ВУ-1 до тех пор, пока на нем не будет установлен код символа А и об этом не будет сообщено нажатием кнопки ГОТОВ
25	0005		Код первого введенного символа (Д) сдвигается на
26	C024		восемь разрядов влево, освобождая место для
27	E101		ввода следующего символа
28	C027		Опрос флага контроллера ВУ-1 … (см.
29	E201		комментарий к командам 20 и 21
2A	E001		Ввод кода символа, установленного на тумблерах ВУ-1 (если подан сигнал готовности ВУ-1)
2B	3006		Сброс готовности ВУ-1
2C	F000	HLT	Пересылка «ДА» в ячейку 006
			Останов ЭВМ

Вопрос №20. Организация обмена по прерыванию программы в БЭВМ. Пример программы.

Этот вид обмена отличается от асинхронного тем, что сигнал готовности ВУ к обмену анализируется не программным, а аппаратным путем. ЭВМ может выполнять любую не связанную с обменом программу (основной), а когда из ВУ по линии «Запрос прерывания» поступит сигнал готовности ВУ к приему или выдаче информации, прервать (приостановить) выполнение этой программы на время выполнения программы обмена данными. Все эти действия осуществляются с помощью контроллера прерываний, входящего в состав устройства управления БЭВМ. Команды EI и DI переводят контроллер прерываний в одно из двух состояний, в которых он соответственно реагирует или не реагирует на сигналы готовности ВУ, передаваемые по линии «Запрос прерывания».

Основная программа решения задачи примера 3.3

Адрес	Содержимое		Комментарии
	Код	Мнемоника	
20 21	FA00 F200	E1 CLA	Установка состояния разрешения прерывания Очистка аккумулятора
22 23 24	F200 F100 C022	INC NOP BR 22…	Цикл для наращивания содержимогоаккумулятора
25	0000		Ячейка для хранения кодов, поступающих из ВУ-1

Первый вариант подпрограммы обработки прерываний для примера 3.3

Адрес	Содержимое		Комментарии	
	Код	Мнемоника		
0	0000		Ячейка для хранения адреса возврата (этот адрес будет занесен сюда на 2-м шаге)	
1	C030	BR 30	Первая команда подпрограммы – переход к основному её тексту, расположенному в ячейках 30–4D	
…	…	…	…	
30	304C	MOV 4C	Сохранение в буферных ячейках 4C и 4D содер-	**Шаг 3**
31	F600	ROL	жимого аккумулятора и регистра переноса	
32	304D	MOV 4D		
33	E101	TSF 1	Опрос флага ВУ-1. Если он сброшен, то переход	**Шаг 4**
34	C036	BR 36	к опросу флага ВУ-2. В противном случае	
35	C039	BR 39	переход на ввод данных из ВУ-1	
36	E103	TSF 3	Опрос флага ВУ-3. Если он сброшен, то переход	
37	C043	DR 43	к опросу флага ВУ-2. В противном случае	
38	C03E	BR 3E	переход на ввод данных из ВУ-3	
39	F200	CLA	Ввод данных из ВУ-1, пересылка их в ячейку 25,	**Шаг 5**
3A	E201	IN 1	сброс флага ВУ-1, переход к восстановлению	
3B	E001	CLF 1	содержания основных регистров и выходу из	
3C	3025	MOV 25	подпрограммы	
3D	C044	BR 44		
3E	F200	CLA	Пересылка в аккумулятор содержимого буфер-	
3F	404C	ADD 4C	ной ячейки 4C, вывод на ВУ-3 восьми младших	
40	E303	OUT 3	разрядов аккумулятора, сброс флага ВУ-3,	
41	E003	CLF 3	переход к восстановлению регистров и выходу	
42	C044	BR 44		
43	E002	CLF 2	Очистка флага ВУ-2	
44	F200	CLA		**Шаг 6**
45	404D	ADD 4D	Восстановление содержимого регистра переноса	
46	E700	ROR	и аккумулятора	
47	F200	CLA		
48	F400	CMA		
49	104C	AND 4C		
4A	FA00	E1	Возобновление состояния разрешения прерыва-	**Шаг 7**
4B	C800	BR (0)	ния и выход из подпрограммы	
4C	0000		Ячейки для сохранения содержимого аккумулятора и	
4D	0000		регистра переноса	

Вопрос №21. Понятие многоуровневой ЭВМ.

Многоуровневая ЭВМ – это вычислительная машина, имеющая средства для работы с n различными уровнями языков программирования. Нижний язык, или уровень, является наиболее простым, верхний – наиболее сложным. Такую машину можно рассматривать как n различных виртуальных машин, каждая из которых имеет свой машинный язык. Сложность аппаратурной реализации этих виртуальных машин возрастает по мере усложнения языка (увеличения номера уровня).

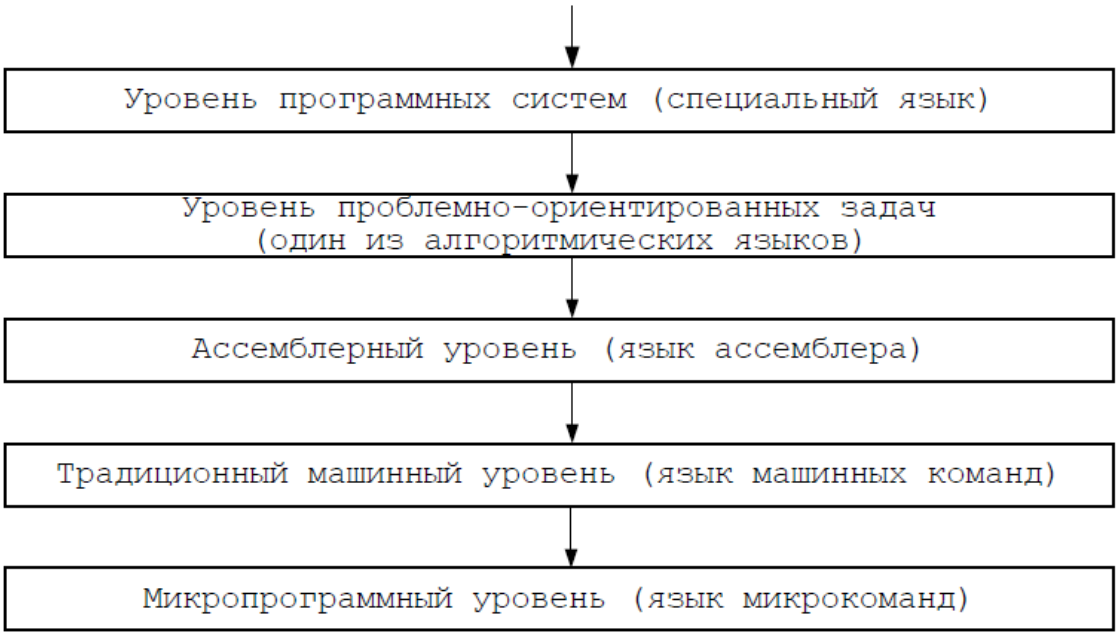


Рис 4.1. Многоуровневая ЭВМ

Вопрос №22. Микропрограммный уровень БЭВМ.

Если рассматривать базовую ЭВМ с позиции многоуровневых ЭВМ, то окажется, что язык команд БЭВМ не является языком самого нижнего уровня. На ее нижнем уровне выполняются элементарные действия (микрооперации) над словами информации.

Управление порядком следования микроопераций осуществляется с помощью устройства управления базовой ЭВМ, которое, в свою очередь, является очень простой ЭВМ. Для этой ЭВМ регистры и вентильные схемы базовой ЭВМ служат как бы устройствами ввода и вывода. Программа работы такой ЭВМ – микропрограммного устройства управления – называется микропрограммой, а ее команды, содержащие информацию об элементарных действиях, подлежащих выполнению в течение одного рабочего такта ЭВМ, - микрокомандами.

Микрокоманда хранится в постоянном запоминающем устройстве – памяти микрокоманд (МП). В каждом такте работы ЭВМ из МП в регистр микрокоманд пересылается очередная микрокоманда, т.е. микрокоманда, на которую указывает счетчик микрокоманд. Затем СчМК наращивается на 1.

Если из памяти выбрана операционная микрокоманда, биты которой определяют нужный набор микроопераций, то состояние этих битов передается на вентильные схемы процессора. Производится соответствующая настройка АЛУ и пересылка через него содержимого одних регистров в другие.

При выборке управляющей микрокоманды в устройство управления МПУ пересылается содержимое указанного в микрокоманде регистра ЭВМ, из него выделяется указанный в микрокоманде бит и сравнивается с определенным битом микрокоманды. Если результат сравнения положителен, то в счетчик микрокоманд пересылается из микрокоманды адрес, по которому должна выбираться следующая микрокоманда. В противном случае никаких пересылок не производится и в следующем такте будет выполняться микрокоманда, расположенная вслед за исполняемой.

Вопрос №23. Работа арифметико-логического устройства.

АЛУ использует содержимое аккумулятора и РД в качестве операндов для получения результата, который помещается в аккумулятор. Все арифметические и логические команды базовой ЭВМ и вспомогательные арифметические операции можно выполнить с помощью АЛУ.

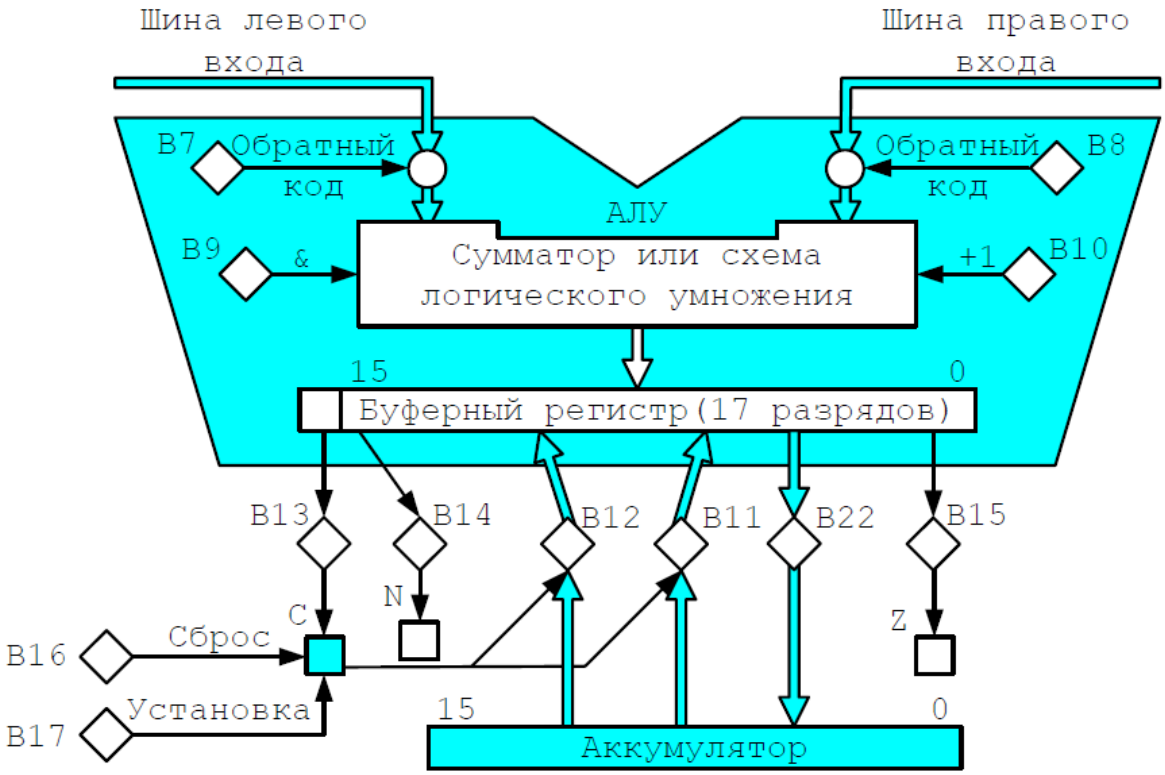


Рис. 4.5. АЛУ, аккумулятор и регистр переноса

Вопрос №24. Регистр состояния, вентильные схемы.

РС – объединение однобитовых регистров признаков и состояний, сделанное в целях формального уменьшения числа регистров, с которыми работает микропрограммное устройство управления, что позволяет сократить разрядность управляющих микрокоманд.

Регистр состояния

Разряд	Содержимое
0	Перенос (C)
1	Нуль (Z)
2	Знак (N)
3	0 - используется для организации безусловных переходов в МПУ
4	Разрешение прерывания
5	Прерывание
6	Состояния ВУ (Ф)
7	Состояние тумблера РАБОТА/ОСТАНОВ (1 - РАБОТА)
8	Программа
9	Выборка команды
10	Выборка адреса
11	Исполнение
12	Ввод-вывод

В1. РД в АЛУ. Используется при арифметических и логической операциях, а также при передачах через АЛУ в РК, СК, РА и МПУ.

В2. РК в АЛУ. Используется для передачи содержимого РК в МПУ.

В3. СК в АЛУ. Используется для увеличения на 1 содержимого СК и передачи через АЛУ в РА, РД и МПУ.

В4. А в АЛУ. Используется при арифметических и логической операциях, а также при передачах через АЛУ в РД и МПУ.

В5. РС в АЛУ. Используется для передачи содержимого РС в МПУ.

В6. КР в АЛУ. Используется во время работы с пультом управления ЭВМ для передачи содержимого клавишного регистра (КР) в СК и РД.

Арифметические операции и сдвиги.

В7. Когда эта вентильная схема открыта, сигнал на левый вход сумматора поступает в обратном коде.

B8. Когда эта вентильная схема открыта, сигнал на правый вход сумматора поступает в обратном коде.

B9. Когда эта вентильная схема открыта, в БР поступает результат логического умножения операндов. При закрытой схеме в БР поступает сумма операндов.

B10. Когда эта вентильная схема открыта, к сумме операндов добавляется 1.

B11. Когда эта вентильная схема открыта, содержимое A и C асимметрично пересылается в БР: содержимое младшего разряда A попадает в
16-й бит БР, содержимое C - в 15-й бит БР, содержимое старшего разряда A - в 14-й бит БР и т. д.

B12. Когда эта вентильная схема открыта, содержимое A и C асимметрич-но пересылается в БР: содержимое младшего разряда A попадает в 1-й бит БР, содержимое C - в 0-й бит БР, содержимое старшего разряда A – в 16-й бит БР и т. д.

Установка признаков.

B13. Когда эта вентильная схема открыта, перенос (0 или 1) из старшего разряда сумматора, зафиксированный в старшем разряде БР, поступает в C.

B14. Когда эта вентильная схема открыта, знак числа, хранящегося в БР, поступает в N.

B15. Когда эта вентильная схема открыта, в Z поступает 1, если содержи-мое БР равно 0, и 0, если содержимое БР не равно 0.

B16. Когда эта вентильная схема открыта, в C записывается 0.

B17. Когда эта вентильная схема открыта, в C записывается 1.

Выходной сигнал АЛУ (БР).

B18. БР в РА. Используется при передаче в РА содержимого РД или СК.

B19. БР в РД. Используется при передаче в РД содержимого СК, А или КР.

B20. БР в РК. Используется при передаче в РК содержимого РД.

B21. БР в СК. Используется при передаче в СК содержимого РД или КР и для наращивания СК на 1.

B22. БР в А. Используется для пересылки результатов арифметических операций и сдвигов в А.

Чтение из памяти интерпретируемой машины и запись в эту память.

B23. Из памяти в РД. Используется для загрузки в РД слова памяти, адрес которого указан в РА.

B24. Из РД в память. Используется для записи содержимого РД в слово памяти, адрес которого указан в РА.

Организация ввода-вывода информации.

B25. Когда эта вентильная схема открыта, в контроллеры ВУ из РД пересылается приказ на ввод-вывод и адрес требуемого ВУ.

B26. Когда эта вентильная схема открыта, на все контроллеры ВУ одно-временно поступает сигнал, сбрасывающий их флаги.

B27. Когда эта вентильная схема открыта, производится сброс бита разрешения прерывания в РС (бит 4).

B28. Когда эта вентильная схема открыта, производится установка бита разрешения прерывания в РС (бит 4).

Останов ЭВМ.

B0. Эта вентильная схема используется для передачи сигнала прекращения выполнения программы (команда HLT).

Нетрудно заметить, что в предложенной структуре передача данных между регистрами осуществляется только через буферный регистр АЛУ. Это сделано для уменьшения числа шин и вентилей, т. е. для упрощения и удешев-ления процессора. Однако такое решение замедляет процедуру обмена.

Вопрос №25. Микропрограммное управление вентильными схемами.

При появлении тактового импульса из памяти микрокоманд извлекается и загружается в РМК слово, на которое указывает СчМК, и к содержимому этого счетчика прибавляется единица. Если из памяти извлечена УМК, то в 31-бите РМК содержится 1 (код операции УМК), которая открывает вентильную схему ВР1 и тем самым создает условия для исполнения УМК. Если же извлечена ОМК, то в 31-бите РМК – ноль. Этот сигнал с помощью инвертора НЕ открывает вентильную схему ВР0, и через нее на В0-В28 передаются состояния соответствующих битов РМК. Разряды РМК, содержащие единицы, создают открывающий управляющий сигнал, а содержащие нули – закрывающий (У0-У28).

При исполнении УМК по сигналу, создаваемому каким-либо битом поля выбора проверяемого регистра, открывается одна из вентильных схем В1, В2, В4 или В5 и на вентили ВВ0-ВВ15 поступает через АЛУ содержимое соответствующего регистра. Одновременно на эти же вентили поступает с РМК содержимое поля выбора проверяемого бита. Так как в этом поле должна быть записана только одна единица, то открывается лишь один из вентилей ВВ0-ВВ15, через который на схему сравнения поступает содержимое проверяемого бита из проверяемого регистра.

На второй вход схемы поступает содержимое однобитового поля сравнения (24-й бит УМК), в которое при кодировании УМК записали цифру 0 или 1. Если проверяемый бит и бит из поля сравнения идентичны, то схема сравнения формирует единичный сигнал, который открывает вентильную схему ВА, и на СчМК пересылается адрес перехода (биты с 16 по 23). В противном случае на СчМК сохраняется адрес микрокоманды, расположенной вслед за исполняемой.

Вопрос №26. Интерпретатор БЭВМ.

Таблица 4.5

Интерпретатор учебной ЭВМ (микропрограмма

Адрес	Микрокоманды		Комментарии	
	Горизонт.	Верт.	Метка	Действие
Цикл выборки команды				
01	0000 0008	0300	нач	СК ==> БР
02	0004 0000	4001		БР ==> РА
03	0080 0408	0311		ОП(РА) ==> РД, СК + 1 ==> БР
04	0020 0000	4004		БР ==> СК
05	0000 0002	0100		РД ==> БР
06	0010 0000	4003		БР ==> РК
				Определение типа команды
07	880C 8000	AF0C		IF BIT(15,РК) = 0 THEN АДЦ(0С)
08	880C 4000	AE0C		IF BIT(14,РК) = 0 THEN АДЦ(0С)
09	880C 2000	AD0C		IF BIT(13,РК) = 0 THEN АДЦ(0С)
0A	895E 1000	EC5E		IF BIT(12,РК) = 1 THEN БАД(5Е)
0B	828E 0008	838E		GOTO B/B(8E)
				Определение вида адресации

Адрес	Микрокоманды		Комментарии	
	Горизонт.	Верт.	Метка	Действие
0C	881D 0800	AB1D	АДЦ	IF BIT(11,PK) = 0 THEN АДР(1D)
Цикл выборки адреса операнда				
0D	0000 0002	0100		РД ==> БР
0E	0004 0000	4001		БР ==> РА
0F	0080 0000	0001		ОП(РА) ==> РД
10	881D 0008	A31D		IF BIT(3,PK) = 0 THEN АДР(1D)
11	891D 0010	E41D		IF BIT(4,PK) = 1 THEN АДР(1D)
12	891D 0020	E51D		IF BIT(5,PK) = 1 THEN АДР(1D)
13	891D 0040	E61D		IF BIT(6,PK) = 1 THEN АДР(1D)
14	891D 0080	E71D		IF BIT(7,PK) = 1 THEN АДР(1D)
15	891D 0100	E81D		IF BIT(8,PK) = 1 THEN АДР(1D)
16	891D 0200	E91D		IF BIT(9,PK) = 1 THEN АДР(1D)
17	891D 0400	EA1D		IF BIT(10,PK) = 1 THEN АДР(1D)
18	0000 0402	0110		РД ==> БР
19	0008 0000	4002		БР ==> РД
1A	0100 0000	0002		РД ==> ОП(РА)
1B	0000 0082	0140		РД + COM(0) = РД - 1 ==> БР
1C	0008 0000	4002		БР ==> РД
Цикл исполнения адресных команд				
				Декодирование адресных команд
1D	892D 8000	EF2D	АДР	IF BIT(15,PK) = 1 THEN ПРХ(2D)
1E	0000 0002	0100		РД ==> БР
1F	0004 0000	4001		БР ==> РА
20	8927 4000	EE27		IF BIT(14,PK) = 1 THEN АРФ(27)
21	8824 2000	AD24		IF BIT(13,PK) = 0 THEN A1(24)
22	8857 1000	AC57		IF BIT(12,PK) = 0 THEN JSR(57)
23	8238 0008	8338		GOTO MOV(38)
24	0080 0000	0001	A1	ОП(РА) ==> РД
25	8850 1000	AC50		IF BIT(12,PK) = 0 THEN ISZ(50)
26	8235 0008	8335		GOTO AND(35)
27	0080 0000	0001	АРФ	ОП(РА) ==> РД
28	882B 2000	AD2B		IF BIT(13,PK) = 0 THEN СУМ(2B)
29	8843 1000	AC43		IF BIT(12,PK) = 0 THEN SUB(43)
2A	82B0 0008	83B0		GOTO P - A(B0)
2B	883C 1000	AC3C	СУМ	IF BIT(12,PK) = 0 THEN ADD(3C)
2C	823F 0000	833F		GOTO ADC(3F)
2D	8830 4000	AE30	ПРХ	IF BIT(14,PK) = 0 THEN УПХ(30)
2E	8847 1000	AC47		IF BIT(12,PK) = 0 THEN BR(47)
2F	82D0 0008	83D0		GOTO P - П(D0)
30	8833 2000	AD33	УПХ	IF BIT(13,PK) = 0 THEN П1(33)
31	884C 1000	AC4C		IF BIT(12,PK) = 0 THEN BMI(4C)
32	824E 0008	834E		GOTO BEQ(4E)
33	8846 1000	AC46	П1	IF BIT(12,PK) = 0 THEN BCS(46)
34	824A 0008	834A		GOTO BPL(4A)
				Исполнение адресных команд
35	0000 0212	1120	AND	A & РД ==> БР
36	0040 C000	4035		БР ==> A, N, Z
37	828F 0008	838F		GOTO ПРЕ(8F)
38	0000 0010	1000	MOV	A ==> БР
39	0008 0000	4002		БР ==> РД
3A	0100 0000	0002		РД ==> ОП(РА)
3B	828F 0008	838F		GOTO ПРЕ(8F)
3C	0000 0012	1100	ADD	A + РД ==> БР
3D	0040 E000	4075		БР ==> A, C, N, Z
3E	828F 0008	838F		GOTO ПРЕ(8F)
3F	823C 0001	803C	ADC	IF BIT(0,PC) = 0 THEN ADD(3C)
40	0000 0412	1110		A + РД + 1 ==> БР
41	0040 E000	4075		БР ==> A, C, N, Z

Адрес	Микрокоманды		Комментарии	
	Горизонт.	Верт.	Метка	Действие
42	828F 0008	838F		GOTO ПРЕ(8F)
43	0000 0512	1190	SUB	A + COM(РД) + 1 = A - РД ==> БР
44	0040 E000	4075		БР ==> A, C, N, Z
45	828F 0008	838F		GOTO ПРЕ(8F)
46	828F 0001	808F	BCS	IF BIT(0,PC) = 0 THEN ПРЕ(8D)
47	0000 0002	0100	BR	РД ==> БР
48	0020 0000	4004		БР ==> СК
49	828F 0008	838F		GOTO ПРЕ(8F)
4A	838F 0004	C28F	BPL	IF BIT(2,PC) = 1 THEN ПРЕ(8F)
4B	8247 0008	8347		GOTO BR(47)
4C	828F 0004	828F	BMI	IF BIT(2,PC) = 0 THEN ПРЕ(8F)
4D	8247 0008	8347		GOTO BR(47)
4E	828F 0002	818F	BEQ	IF BIT(1,PC) = 0 THEN ПРЕ(8F)
4F	8247 0008	8347		GOTO BR(47)
50	0000 0402	0110	ISZ	РД + 1 ==> БР
51	0008 0000	4002		БР ==> РД
52	0100 0000	0002		РД ==> ОП(РА)
53	858A 8000	DF8F		IF BIT(15,РД) = 1 THEN ПРЕ(8F)
54	0000 0408	0310		СК + 1 ==> БР
55	0020 0000	4004		БР ==> СК
56	828F 0008	838F		GOTO ПРЕ(8F)
57	0000 0402	0110	JSR	РД + 1 ==> БР
58	0010 0000	4003		БР ==> РК
59	0000 0008	0300		СК ==> БР
5A	0008 0000	4002		БР ==> РД
5B	0100 0004	0202		РД ==> ОП(РА), РК ==> БР
5C	0020 0000	4004		БР ==> СК
5B	828F 0008	838F		GOTO ПРЕ(8F)
Продолжение цикла выборки команды декодирование и исполнение безадресных команд				
5E	8861 0800	AB61	БАД	IF BIT(11,РК) = 0 THEN Б0(61)
5F	886C 0400	AA6C		IF BIT(10,РК) = 0 THEN Б1(6C)
60	82E0 0008	83E0		GOTO Р - Б(E0)
61	8867 0400	AA67	Б0	IF BIT(10,РК) = 0 THEN Б2(67)
62	8865 0200	A965		IF BIT(9,РК) = 0 THEN Б3(65)
63	8882 0100	A882		IF BIT(8,РК) = 0 THEN ROL(82)
64	8285 0008	8385		GOTO ROR(85)
65	887B 0100	A87B	Б3	IF BIT(8,РК) = 0 THEN CMA(7B)
66	827E 0008	837E		GOTO CMC(7E)
67	886A 0200	A96A	Б2	IF BIT(9,РК) = 0 THEN Б4(6A)
68	8876 0100	A876		IF BIT(8,РК) = 0 THEN CLA(76)
69	8279 0008	8379		GOTO CLC(79)
6A	8888 0100	A888	Б4	IF BIT(8,РК) = 0 THEN HLT(88)
6B	8287 0008	8387		GOTO NOP(87)
6C	886F 0200	A96F	Б1	IF BIT(9,РК) = 0 THEN Б5(6F)
6D	888A 0100	A88A		IF BIT(8,РК) = 0 THEN EI(8A)
6E	828C 0008	838C		GOTO DI(8C)
6F	8873 0100	A873	Б5	IF BIT(8,РК) = 0 THEN INC(73)
70	0000 0110	1080	DEC	A + COM(0) = A - 1 ==> БР
71	0040 E000	4075		БР ==> A, C, N, Z
72	828F 0008	838F		GOTO ПРЕ(8F)
73	0000 0410	1010	INC	A + 1 ==> БР
74	0040 E000	4075		БР ==> A, C, N, Z
75	828F 0008	838F		GOTO ПРЕ(8F)
76	0000 0200	0020	CLA	0 ==> БР
77	0040 C000	4035		БР ==> A, N, Z
78	828F 0008	838F		GOTO ПРЕ(8F)
79	0001 0000	4080	CLC	0 ==> C

Адрес	Микрокоманды		Комментарии	
	Горизонт.	Верт.	Метка	Действие
7A	828F 0008	838F		GOTO ПРЕ(8F)
7B	0000 0090	1040	СМА	COM(A) ==> БР, инверсия А
7C	0040 C000	4035		БР ==> A, N, Z
7D	828F 0008	838F		GOTO ПРЕ(8F)
7E	8280 0001	8080	СМС	IF BIT(0,PC) = 0 THEN Б6(80)
7F	8279 0008	8379		GOTO CLC(79)
80	0002 0000	40C0	Б6	1 ==> C
81	828F 0008	838F		GOTO ПРЕ(8F)
82	0000 1000	0008	ROL	RAL(A) ==> БР, сдвиг влево
83	0040 E000	4075		БР ==> A, C, N, Z
84	828F 0008	838F		GOTO ПРЕ(8F)
85	0000 0800	0004	ROR	RAR(A) ==> БР, сдвиг вправо
86	0040 E000	4075		БР ==> A, C, N, Z
87	828F 0008	838F	NOP	GOTO ПРЕ(8F)
88	0000 0001	4008	HLT	Останов машины
89	8201 0008	8301		GOTO НАЧ(01)
8A	1000 0000	4800	EI	Разрешение прерывания
8B	8201 0008	8301		GOTO НАЧ(01)
8C	0800 0000	4400	DI	Запрещение прерывания
8D	8201 0008	8301		GOTO НАЧ(01)
Продолжение цикла выборки команды декодирование и исполнение команд ввода-вывода				
8E	0200 0000	4100	В/В	Организация связей с ВУ
Цикл прерывания				
8F	8288 0080	8788	ПРЕ	IF BIT(7,PC) = 0 THEN HTL(88)
90	8201 0020	8501		IF BIT(5,PC) = 0 THEN НАЧ(01)
91	0000 0200	0020		0 ==> БР
92	0004 0000	4001		БР ==> РА
93	0000 0008	0300		СК ==> БР
94	0008 0000	4002		БР ==> РД
95	0100 0400	0012		РД ==> ОП(РА), 1 ==> БР
96	0020 0000	4004		БР ==> СК
97	0800 0000	4400		Запрещение прерывания
98	8201 0008	8301		GOTO НАЧ(01)
Пультовые операции				
				Ввод адреса
99	0000 0040	3000	В/А	КР ==> БР
9A	0020 0000	4004		БР ==> СК
9B	828F 0008	8388		GOTO ПРЕ(88)
				Чтение
9C	0000 0008	0300	ЧТ	СК ==> БР
9D	0004 0000	4001		БР ==> РА
9E	0080 0408	0311		ОП(РА) ==> РД, СК + 1 ==> БР
9F	0020 0000	4004		БР ==> СК
A0	828F 0008	8388		GOTO ПРЕ(88)
				Запись
A1	0000 0008	0300	ЗАП	СК ==> БР
A2	0004 0000	4001		БР ==> РА
A3	0000 0040	3000		КР ==> БР
A4	0008 0000	4002		БР ==> РД
A5	0100 0408	0312		РД ==> ОП(РА), СК + 1 ==> БР
A6	0020 0000	4004		БР ==> СК
A7	828F 0008	8388		GOTO ПРЕ(88)
				Пуск
A8	0000 0200	0020	ПУСК	0 ==> БР
A9	005C E000	4077		БР ==> A, C, N, Z, РА, РД, РК
AA	0400 0000	4200		Сброс флагов ВУ
AB	0800 0000	4400		Запрещение прерывания

Адрес	Микрокоманды		Комментарии	
	Горизонт.	Верт.	Метка	Действие
AC	828F 0008	838F		GOTO ПРЕ(8F)
. . .				
B0			Р - А	Арифметическая команда 7###
. . .				
D0			Р - П	Команда перехода D###
. . .				
E0			Р - Б	Безадресная команда FC##
. . .				
FF				

В системе команд базовой ЭВМ ряд кодов операций зарезервирован для включения новых команд. Это арифметическая команда 7xxx, команда перехода Dxxx и безадресные команды FC00, FD00, FE00 или FF00. Когда при декодировании команды выясняется, что выбрана команда 7xxx, производится передача управления к строке B0. Следовательно, часть микропрограммы, описывающая последовательность микроопераций по реализации этой команды, должна начинаться со строки B0. Сюда можно записать, например, действия по умножению или делению операндов. Часть микропрограммы, реализующая дополнительные безадресные команды, начинается от строки E0. Первые микрокоманды этой части должны осуществить декодирование выбранной команды (определить расширение кода операции: C, D, E или F) и передать управление соответствующим микрокомандам нового куска микропрограммы. Так как память микрокоманд имеет 256 ячеек, то при составлении микропрограмм новых команд можно использовать лишь строки с номерами от AD до FF.

Вопрос №27. Структура типичных ЭВМ. Современные многопроцессорные архитектуры UMA и NUMA.

Процессоры современных микроЭВМ (микропроцессоры) обычно изготавливают в виде одной большой интегральной схемы (БИС), имеющей до 300 тыс. элементов в кристалле. Такой микропроцессор должен соединяться системой шин с памятью и контроллерами ВУ, и эти соединения должны осуществляться через выводы БИС.

Сокращения выводов БИС добиваются за счет использования двунаправленных шин. Так, вместо шин «Чтение» и «Запись» можно иметь лишь одну шину, по которой в зависимости от значения дополнительного управляющего сигнала данные передаются либо из памяти в процессор, либо наоборот. Полученная таким образом структура называется структура с раздельными шинами.

Сходство процессов обмена информацией процессора – памяти и процессора – регистров контроллеров ВУ приводит к мысли об использовании одних и тех же выводов для связи с памятью и контроллерами ВУ. При такой структуре (с изолированными шинами) число выводов БИС уменьшается, но исчезает возможность одновременного обмена информацией между процессором, памятью и ВУ.

Дальнейшего сокращения выводов БИС микропроцессора с трехшинной структурой (ША, ШД, ШУ) можно добиться за счет объединения адресной шины и шины данных. При этом возникает так называемая мультиплексируемая шина, по которой в одни моменты времени передаются адреса, а в другие – данные.

Структуры с вводом-выводом, организованным по аналогии с обращением к памяти – структуры с общими шинами. Процессор как бы обманывают, заставляя его выполнять операцию с регистром данных контроллера ВУ, тогда как процессор работает так, как если бы он имел дело с ячейкой памяти ЭВМ.

Uniform Memory Access (сокращённо UMA — «однородный доступ к памяти») — архитектура многопроцессорных компьютеров с общей памятью. Все микропроцессоры в UMA-архитектуре используют физическую память одновременно. При этом время запроса к данным из памяти не зависит ни от того, какой именно процессор обращается к памяти, ни от того, какой именно чип памяти содержит нужные данные.

NUMA (Non-Uniform Memory Access — «неравномерный доступ к памяти» или Non-Uniform Memory Architecture — «Архитектура с неравномерной памятью») — схема реализации компьютерной памяти, используемая в мультипроцессорных системах, когда время доступа к памяти определяется её расположением по отношению к процессору.

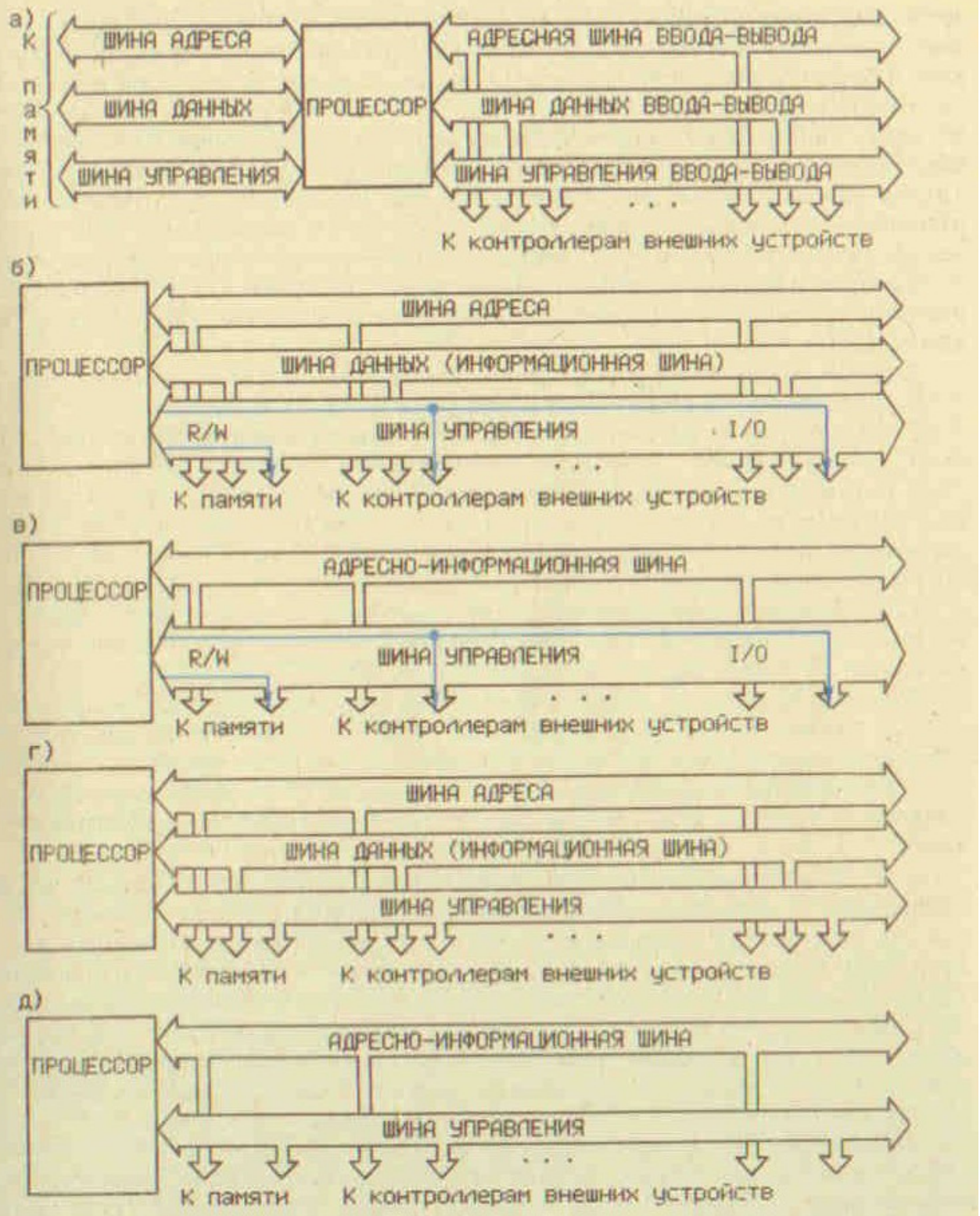


Рис. 5.2. Структуры микроЭВМ:

а — с раздельными шинами; *б* — с изолированными шинами; *в* — с изолированными шинами и мультиплексированием шин адресов и данных; *г* — с общими шинами; *д* — с общими шинами и мультиплексированием шин адресов и данных

Вопрос №28. Понятие адресного пространства вычислительных систем. Раздельные и смежные адресные пространства. Организация памяти.

Адресное пространство — это набор адресов, который может быть использован процессом для обращения к памяти.

Вопрос №29. Адресное пространство БЭВМ. Команды работы с памятью.

Вопрос №30. Характеристики запоминающих устройств. Современные типы памяти и их характеристики.

Запоминающие устройства имеют ряд показателей качества, характеризующих их информационные и временные свойства.

Информационная емкость памяти выражается в количестве битов, байтов или слов, состоящих из определенного числа байтов.

Время доступа – временной интервал, определяемый от момента, когда процессор выставил на адресной шине адрес требуемой ячейки и послал по шине управления приказ на чтение или запись данных, до момента осуществления связи адресуемой ячейки с шиной данных.

Время записи – интервал времени, необходимый для переписи содержимого шины данных в связанную с ней ячейку памяти.

Время восстановления необходимо для приведения памяти в исходное состояние после того, как процессор снял с шин адрес, сигнал ЧТЕНИЕ/ЗАПИСЬ и данные.

Цикл считывания и цикл записи определяются как время с момента выдачи процессором адреса требуемой ячейки памяти и сигнала на считывание или запись до того момента, когда заканчиваются все действия, связанные с выполняемой операцией, и память будет готова реализовать следующую операцию.

Стоимость 1 бита определяется отношением стоимости памяти к ее информационной емкости.

Возможность изменения информации характеризует тип памяти. Существует элементы памяти с легко изменяемыми состояниями (ОЗУ). Есть более дешевые элементы памяти, в которые единожды записывают 0 или 1 (ПЗУ, ППЗУ, СППЗУ (содержимое можно стереть, а затем вновь заполнить информацией)).

Различают 2 основных типа ЗУ: с произвольной выборкой и последовательной выборкой. В первых время доступа к заданному слову не зависит от месторасположения этого слова в памяти, а во вторых – зависит.

В некоторых видах ЗУ происходит потеря записанной информации при отключении питающего напряжения. Такие устройства называют энергозависимыми ЗУ. В энергонезависимых ЗУ информация при отключении питания сохраняется.

Вопрос №31. Предназначение и организация виртуальной памяти.

Виртуальная память – технология управления памятью ЭВМ, благодаря которому операционная система может обращаться к памяти, большей, чем память, фактически установленная в компьютере. Это достигается за счет помещения данных в свободное дисковое пространство внешнего ЗУ, которое задействовано в роли оперативной памяти. Необходимо понимать, что часть программ, которые мы не смогли разместить в оперативной памяти из-за её нехватки, теперь будут размещены на ВЗУ и это будет эквивалентно размещению в оперативной памяти. Использование ВЗУ очень удобно, так как в это время пользователь оперирует с общим адресным пространством и ему безразлично, какая физическая память при этом используется: внешняя или внутренняя.

В большинстве современных ОС виртуальная память организуется с помощью страничной адресации. ОП делится на страницы: области памяти фиксированной длины, которые являются минимальной единицей выделяемой памяти. Процесс обращается к памяти с помощью адреса виртуальной памяти, который содержит в себе номер страницы и смещение внутри страницы. Если страница выгружена из ОП, то ОС подкачивает страницу с жёсткого диска. При запросе на выделение памяти операционная система может «сбросить» на жёсткий диск страницы, к которым давно не было обращений. Критические данные (например, код запущенных и работающих программ, код и память ядра системы) обычно находятся в оперативной памяти.

Сегментная организация – механизм организации виртуальной памяти, при котором виртуальное пространство делится на части произвольного размера — сегменты. Для каждого сегмента, как и для страницы, могут быть назначены права доступа к нему пользователя и его процессов. При загрузке процесса часть сегментов помещается в оперативную память, а часть сегментов размещается в дисковой памяти. Сегменты одной программы могут занимать в оперативной памяти несмежные участки. Недостатком данного метода распределения памяти является фрагментация на уровне сегментов и более медленное по сравнению со страничной организацией преобразование адреса.

<u>Вопрос №32. Интерфейсы ввода-вывода. Уровни стандартизации, сопряжение с системной шиной, циклы обмена.</u>

Интерфейс – совокупность технических и программных средств и правил (описаний, соглашений, протоколов), обеспечивающих взаимодействие устройств и/или программ вычислительной системы или сопряжение между системами.

Уровни стандартизации интерфейсов: логический (все алгоритмы неким образом понимаются системой), физический (напряжение, ток), конструктивный (особенности конструкции).

Можно выделить 2 уровня сопряжения ВУ с процессором и памятью. На первом уровне контроллеры ВУ сопрягаются с процессором и памятью через системный интерфейс микроЭВМ, который обеспечивает комплексирование отдельных устройств микроЭВМ в единую систему. На втором уровне сопряжения контроллеры посредством шин связи с ВУ соединяются с соответствующими внешними устройствами микроЭВМ.

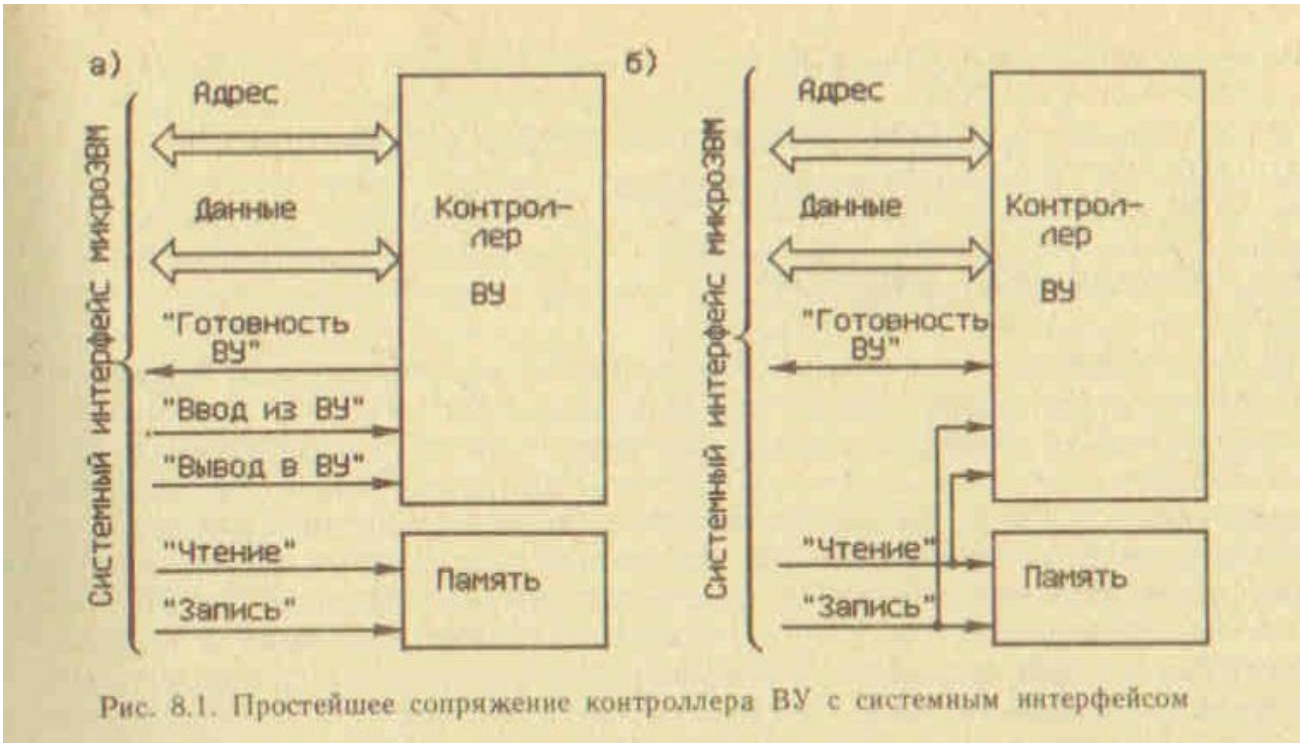


Рис. 8.1. Простейшее сопряжение контроллера ВУ с системным интерфейсом

Обмен информацией в микропроцессорных системах происходит в циклах обмена информацией. Под циклом обмена информацией понимается временной интервал, в течение которого происходит выполнение одной элементарной операции обмена по шине. Например, пересылка кода данных из процессора в память или же пересылка кода данных из устройства ввода/вывода в процессор. Циклы обмена информацией делятся на два основных типа:

- Цикл записи (вывода), в котором процессор записывает (выводит) информацию.
- Цикл чтения (ввода), в котором процессор читает (вводит) информацию.

Вопрос №33. Контроллеры внешних устройств. Уровни сопряжения ВУ с ЭВМ. Регистры контроллера.

Подключение любого внешнего устройства к микроЭВМ осуществляется через контроллер ВУ. Способы структурной и функциональной организации контроллеров ВУ определяются двумя основными факторами: форматами данных и режимами работы конкретных ВУ; типом системного интерфейса микроЭВМ.

Можно выделить 2 уровня сопряжения ВУ с процессором и памятью. На первом уровне контроллеры ВУ сопрягаются с процессором и памятью через системный интерфейс микроЭВМ, который обеспечивает комплексирование отдельных устройств микроЭВМ в единую систему. На втором уровне сопряжения контроллеры посредством шин связи с ВУ соединяются с соответствующими внешними устройствами микроЭВМ.

Основу контроллера ВУ составляют несколько регистров, которые служат для временного хранения передаваемой информации. Каждый регистр имеет свой адрес, и зачастую такие регистры называют портами ввода-вывода. Регистры входных и выходных данных работают соответственно только в режиме чтения или только в режиме записи. Регистр состояния работает только в режиме чтения и содержит информацию о текущем состоянии ВУ (включено/выключено, готово/не готово к обмену данными и т.п.). Регистр управления работает только в режиме записи и служит для приема из микроЭВМ приказов для ВУ. В контроллерах, используемых для подключения достаточно простых ВУ, удается совместить в одни регистры состояния и управления, что позволяет сократить количество используемых в контроллере портов ввода-вывода, а следовательно, и адресов, выделенных для данного ВУ.

Вопрос №34. Параллельная передача данных. Контроллеры параллельной передачи и приема.

Параллельная передача данных между контроллером и ВУ является по своей организации наиболее простым способом обмена. Для организации ППД помимо ШД, количество линий в которой равно числу одновременно передаваемых битов данных, используется минимальное количество управляющих сигналов.

В простом контроллере ВУ, обеспечивающем побайтную передачу данных во ВУ, в шине связи с ВУ используются два управляющих сигнала – «Выходные данные готовы» и «Данные приняты».

Для формирования управляющего сигнала «Выходные данные готовы» и «Данные приняты» в контроллере используется одноразрядный адресуемый регистр состояния и управления А2. Одновременно с записью очередного байта данных из ШД системного интерфейса в адресуемый регистр данных контроллера (порт вывода А1) в регистр состояния и управления записывается логическая единица. Тем самым формируется управляющий сигнал «Выходные данные готовы» в шине связи с ВУ.

ВУ, приняв байт данных, управляющим сигналом «Данные приняты» обнуляет РС контроллера. При этом формируется управляющий сигнал системного интерфейса «Готовность ВУ», признак готовности ВУ к обмену, передаваемый в процессор по одной из линий ШД системного интерфейса посредством стандартной операции ввода при реализации программы асинхронного обмена.

Логика управления контроллера обеспечивает селекцию адресов регистров контроллера, прием управляющих сигналов системного интерфейса и формирование на их основе внутренних управляющих сигналов контроллера, формирование управляющего сигнала системного интерфейса «Готовность ВУ». Для сопряжения регистров контроллера с ША и ШД системного интерфейса в контроллере используются приемники ША и приемопередатчики ШД.

Алгоритм асинхронной передачи (вывода): 1) Процессор микроЭВМ проверяет готовность ВУ к приему данных. 2) Если ВУ готово к приему данных (логический 0 в регистре А2), то данные передаются из шины данных системного интерфейса в регистр данных А1 контроллера и далее в ВУ. Иначе повторяется п. 1.

В контроллере ВУ, обеспечивающего побайтный прием данных, при взаимодействии с внешним устройством также используются два управляющих сигнала – «Данные от ВУ готовы» и «Данные приняты».

Для формирования управляющего сигнала «Данные приняты» и приема из ВУ управляющего сигнала «Данные от ВУ готовы» используется одноразрядный адресуемый регистр состояния и управления А2.

ВУ записывает в регистр данных контроллера А1 очередной байт данных и управляющим сигналом «Данные от ВУ готовы» устанавливает в единицу регистр состояния и управления А2.

При этом формируется: управляющий сигнал системного интерфейса «Готовность ВУ»; признак готовности ВУ к обмену, передаваемый в процессор по одной из линий ШД системного интерфейса посредством операции ввода при реализации программы асинхронного обмена.

Тем самым контроллер извещает процессор о готовности данных в регистре А1. Процессор выполняя программу асинхронного обмена, читает байт данных из регистра данных контроллера и обнуляет

регистр состояния и управления А2. При этом формируется управляющий сигнал «Данные приняты» в шине связи с ВУ.

Логика управления контроллера и приемопередатчики шин системного интерфейса выполняют те же функции, что и в контроллере вывода.

Алгоритм асинхронного ввода так же прост, как и асинхронного вывода: 1) Процессор проверяет наличие данных в регистре данных контроллера А1. 2) Если данные готовы (логическая 1 в регистре А2), то они передаются из регистра данных А1 в ШД системного интерфейса и делее в регистр процессора или ячейку памяти микроЭВМ. Иначе повторяется п. 1.

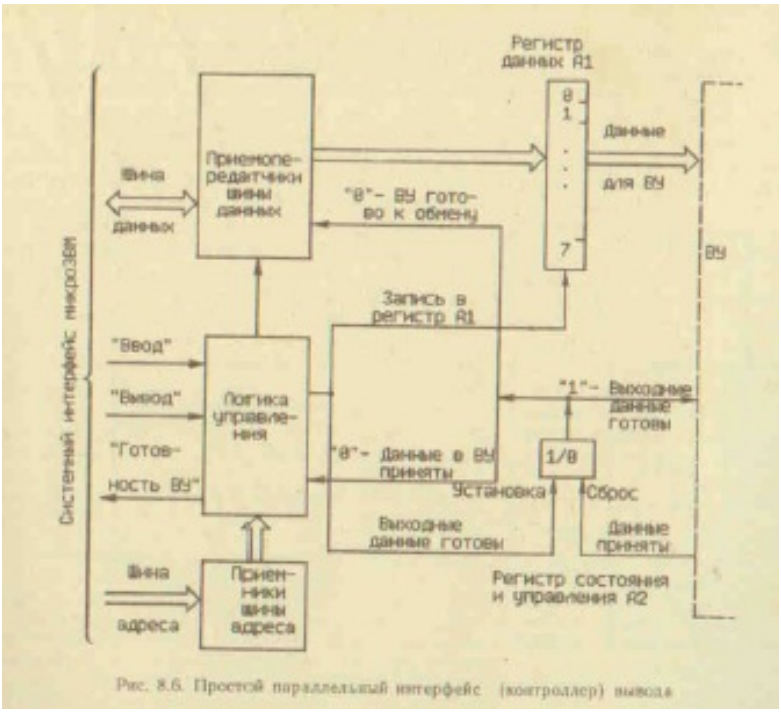


Рис. 8.6. Простой параллельный интерфейс (контроллер) вывода

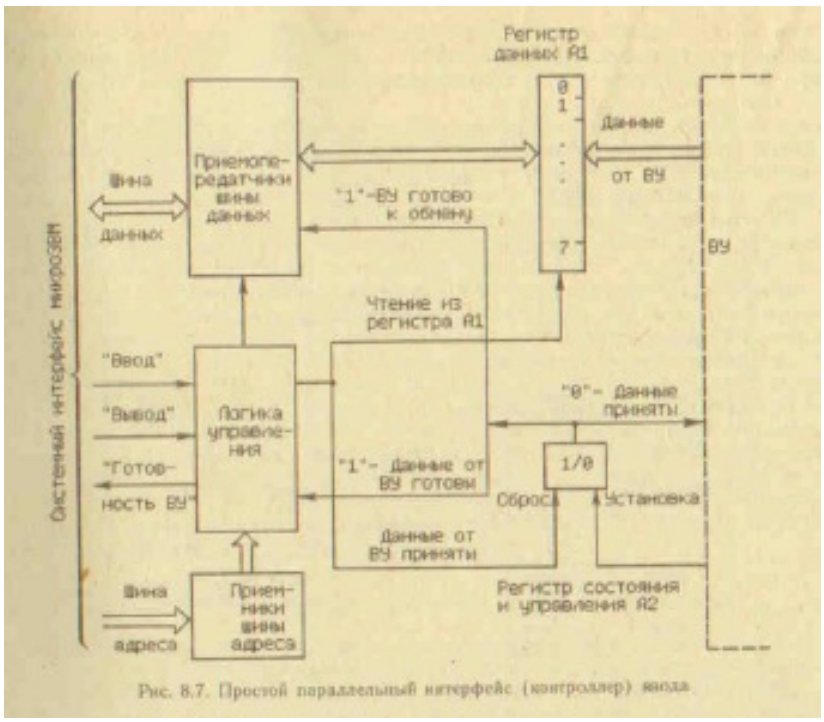


Рис. 8.7. Простой параллельный интерфейс (контроллер) ввода

Вопрос №35. Синхронные последовательные интерфейсы. Контроллеры последовательной передачи и приема.

Простой контроллер для синхронной передачи данных в ВУ.

Восьмиразрядный адресуемый буферный регистр контроллера А1 служит для временного хранения байта данных до его загрузки в сдвиговый регистр. Запись байта данных в буферный регистр из ШД системного интерфейса производится так же, как и в параллельном интерфейсе, только при наличии единицы в одноразрядном адресуемом регистре состояния контроллера А2. Единица в регистре состояния указывает на готовность контроллера принять очередной байт в буферный регистр. Содержимое регистра А2 передается в процессор по одной из линий ШД системного интерфейса и используется для формирования управляющего сигнала системного интерфейса «Готовность ВУ». При записи очередного байта в буферный регистр А1 обнуляется регистр состояния А2.

Преобразование данных из параллельного формата, в котором они поступили в буферный регистр контроллера из системного интерфейса, в последовательный и передача их в линию связи производится в сдвиговом регистре с помощью генератора тактовый импульсов и двоичного трехразрядного счетчика.

Последовательная линия связи контроллера с ВУ подключается к выходу младшего разряда сдвигового регистра. По очередному тактовому импульсу содержимое сдвигового регистра сдвигается на один разряд вправо и в линию связи «Данные» выдается значение очередного разряда. Одновременно со сдвигом в ВУ передается по отдельной линии «Синхронизация» тактовый импульс. Таким образом, каждый передаваемый по линии «Данные» бит информации сопровождается синхронизирующем сигналом по линии «Синхронизация», что обеспечивает его однозначное восприятие на приемном конце последовательной линии связи.

Количество переданных в линию тактовых импульсов, а следовательно, и переданных бит информации подсчитывается счетчиком тактовых импульсов. Как только содержимое счетчика становится равным 7, т.е. в линию переданы 8 бит (1 байт) информации, формируется управляющий сигнал «Загрузка», обеспечивающий запись в сдвиговый регистр очередного байта из буферного регистра. Этим же управляющим сигналом устанавливается в «1» регистр состояния. Очередным тактовым импульсом счетчик будет сброшен в «0», и начнется очередной цикл выдачи восьми битов информации из сдвигового регистра в линию связи.

Синхронная последовательная передача отдельных битов данных в линию связи должна производиться без какого-либо перерыва, и следующий байт данных должен быть загружен в буферный регистр из системного интерфейса за время, не превышающее времени передачи 8 битов в последовательную линию связи.

При записи байта данных в буферный регистр обнуляется регистр состояния контроллера. Нуль в этом регистре указывает, что в линию связи передается байт данных из сдвигового регистра, а следующий передаваемый байт данных загружен в сдвиговый регистр.

Простой контроллер для последовательного синхронного приема данных из ВУ состоит из тех же компонентов, что и контроллер для синхронной последовательной передачи, за исключением генератора тактовых импульсов.

Буферный регистр контроллера А1 служит для временного хранения байта данных, поступившего из сдвигового регистра. Чтение байта данных в системный интерфейс из буферного регистра производится так же, как и в параллельном интерфейсе. Единица в регистре состояния контроллера А2 указывает на готовность контроллера передать очередной байт данных в системный интерфейс.

Данные, поступающие из линии связи в последовательном коде, преобразуются в контроллере в параллельный код с помощью сдвигового регистра и трехразрядного двоичного счетчика тактовых импульсов.

Входная последовательная линия связи «Данные» подключается в котроллере к последовательному входу сдвигового регистра, а входная линия «Синхронизация» - на управляющий вход «Сдвиг» сдвигового регистра и на вход счетчика тактовых импульсов. По очередному тактовому сигналу, поступившему от синхрогенератора ВУ по линии «Синхронизация», производится сдвиг содержимого сдвигового регистра влево и запись очередного бита данных из линии связи «Данные» в младший разряд этого регистра. Одновременно увеличивается на единицу содержимое счетчика тактовых импульсов. Как только содержимое счетчика становится равным 7, т.е. в сдвиговый регистр приняты последовательно 8 бит информации, формируется управляющий сигнал «Запись», который обеспечивает запись в буферный регистр очередного принятого байта из сдвигового регистра. Этим же управляющим сигналом устанавливается в «1» регистр состояния.

За время приема в сдвиговый регистр следующих 8 бит информации байт данных из буферного регистра должен быть передан в шину данных системного интерфейса микроЭВМ. При этой передаче обнуляется регистр состояния контроллера и ноль в этом регистре означает, что в сдвиговый регистр принимается из линии связи очередной байт информации.

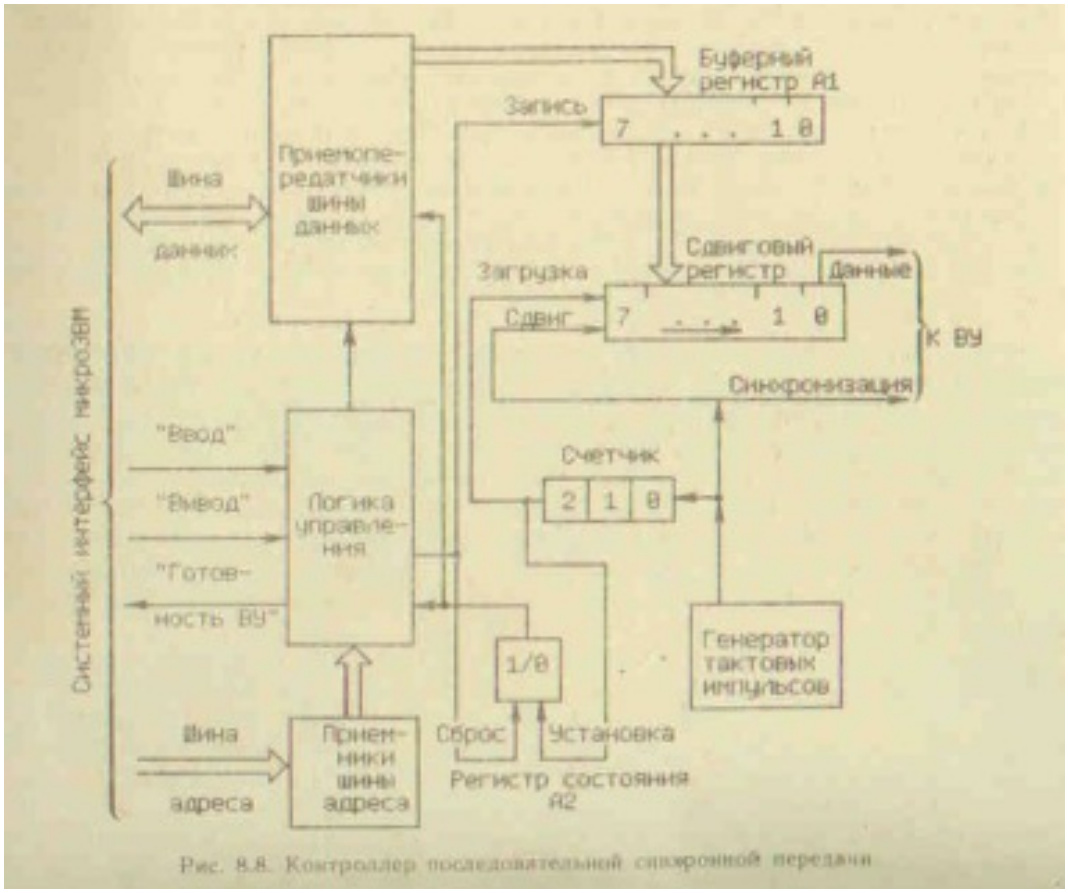


Рис. 8.8. Контроллер последовательной синхронной передачи

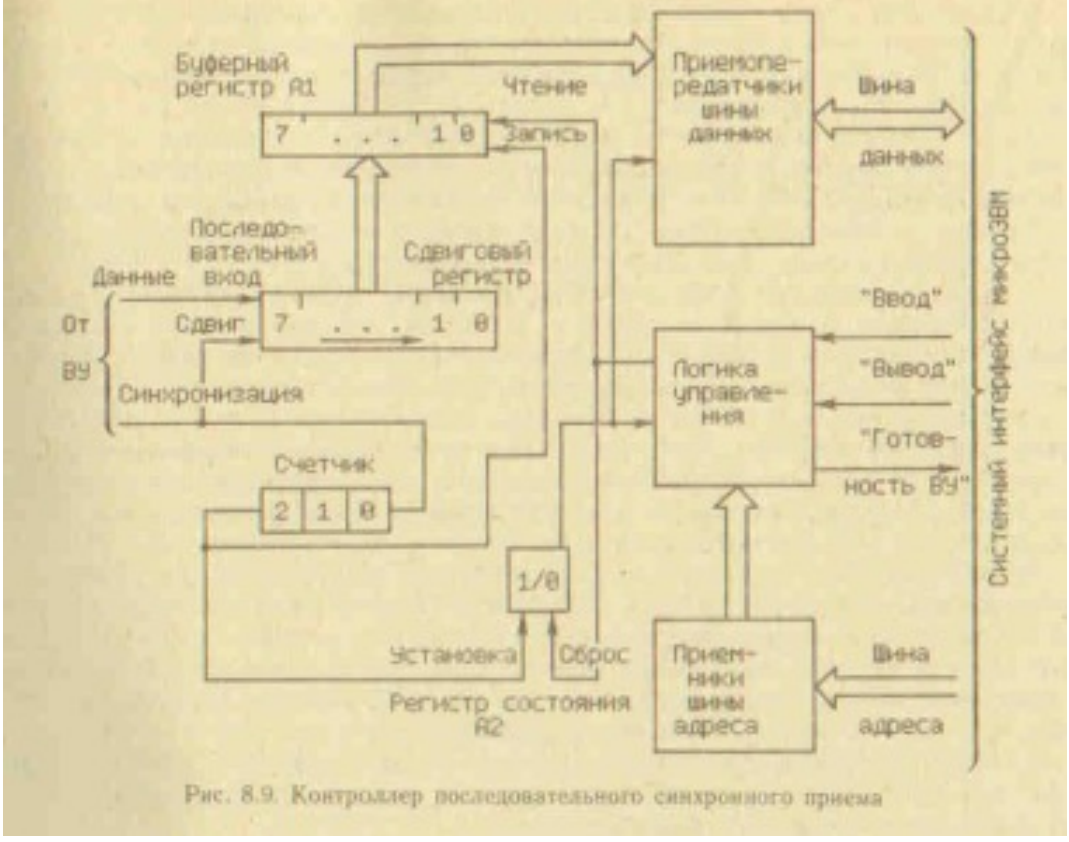


Рис. 8.9. Контроллер последовательного синхронного приема

Вопрос №36. Асинхронный обмен. Принципы деления частоты, формат кадра.

Организация асинхронного последовательного обмена данными с ВУ осложняется тем, что на передающей и приемной стороне последовательной линии связи используется настроенные на одну частоту, но физически разные генераторы тактовых импульсов и, следовательно, общая синхронизация отсутствует.