

Тема 11. Устройства и принципы управления ЭВМ

Основные вопросы:

- 11.1. Устройства управления с жесткой логикой работы
- 11.2. Микропрограммное управление
 - 11.2.1. Горизонтальное микропрограммирование
 - 11.2.2. Вертикальное микропрограммирование
- 11.3. Принципы управления
- 11.4. Прямой доступ к памяти
- 11.5. Способы организации совместной работы периферийных и центральных устройств

Любая система обработки данных всегда рассматривается как совокупность трех устройств:

- памяти;
- устройства обработки;
- **устройства управления.**

Устройство управления (УУ) предназначено для выработки управляющих сигналов, необходимых для выполнения любого действия, происходящего в системе обработки данных.

Классификация УУ:

1. По структурной организации:
 - Централизованные
 - Смешанные – централизованные + местные
 - Иерархические (в ВС)
2. По технической организации:
 - С жесткой логикой работы
 - Устройства микропрограммного управления

Выполнение операций в машине сводится к элементарным преобразованиям информации (передача информации между узлами в блоках, сдвиг информации в узлах, логические поразрядные операции, проверка условий и т.д.) в логических элементах, узлах и блоках под воздействием функциональных управляющих сигналов блоков (устройств) управления. Элементарные преобразования, неразложимые на более простые, выполняются в течение *одного такта сигналов синхронизации* и называются **микрооперациями (МИО)**.

Таким образом, микрооперация – элементарное действие, выполняемое тем или иным функциональным узлом ЭВМ. Набор микроопераций образует **микропрограмму**.

Известны два подхода к построению логики формирования функциональных импульсов. Один из них – аппаратный.

11.1. Устройства управления с жесткой логикой работы

В аппаратных (схемных) устройствах управления каждой операции соответствует свой набор логических схем, выполненных на диодах, транзисторах и т.д. и определяющих, какой функциональный импульс (ФИ) и в каком такте должен быть возбужден. Т.е. **логические схемы вырабатывают определенные**

функциональные сигналы для выполнения микроопераций в определенные моменты времени. Реализация микроопераций достигается за счет однажды соединенных между собой логических схем, поэтому компьютеры с аппаратным УУ называют ЭВМ с ***жесткой логикой управления***. Это понятие относится к фиксации системы команд в структуре связей ЭВМ и означает практическую *невозможность* каких-либо изменений в системе команд компьютера после ее изготовления.

Такой принцип управления операциями получил название "**жесткой**" или "**запаянной**" логики и широко применяется во многих компьютерах.

Рассмотрим пример формирования функционального импульса (ФИ) в такте *j*-ом операции *m* при условии наличия переполнения сумматора или в такте *i*-ом операции *n*. Требуемое действие будет выполнено, если подать сигналы, соответствующие указанным кодам операции, тактам и условиям на входы схем И, а выходы последних через схему ИЛИ соединить с формирователем ФИ (рис. 11.1).

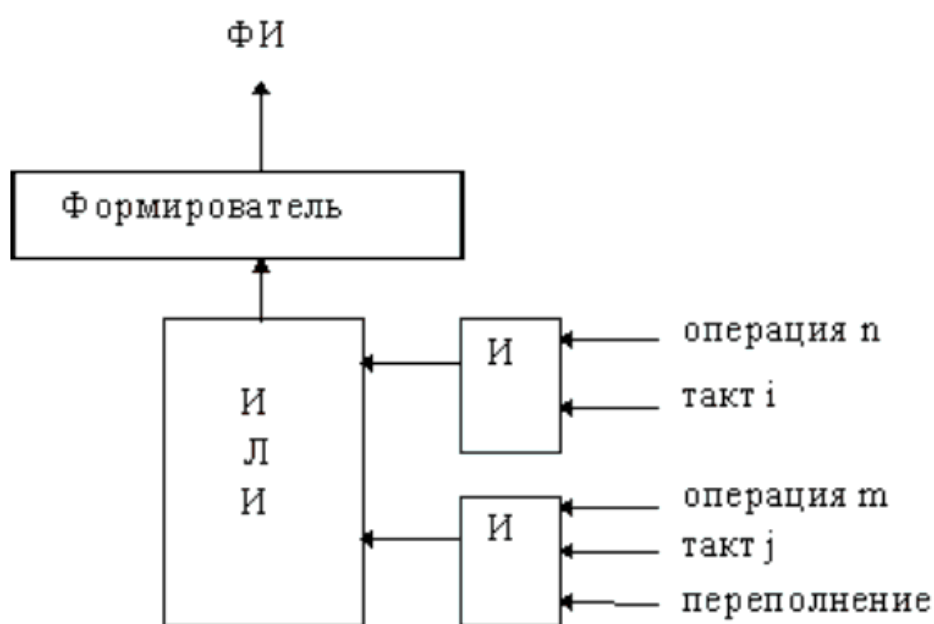


Рис. 11.1. Формирование функционального импульса

Существует и другой принцип организации управления компьютером, при котором каждая команда выполняемой программы является обращением к так называемой **микропрограмме**, содержащей **набор микрокоманд**¹, указывающих, какие ФИ и в какой последовательности необходимо возбуждать.

Например, **микропрограмму** образует **набор микрокоманд** для выполнения команды умножения.

Любое действие, выполняемое в операционном блоке, описывается некоторой микропрограммой и реализуется за один или несколько тактов. Элементарная функциональная операция, выполняемая за один тактовый интервал и приводимая в действие управляющим сигналом, называется **микрооперацией (МИО)**.

¹ Микрокоманда – совокупность совместимых микроприказов, инициирующих выполнение микрооперации

Т.е. каждой **микрооперации** в компьютере ставится в соответствие слово (или часть слова), называемое **микрокомандой (МИК)**, хранимой в постоянной памяти, составляющей неотъемлемую часть УУ.

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.

Микрокоманда может содержать три части:

- **оперативную**, в которой указываются управляющие входы всех исполнительных устройств машины;
- **адресную**, определяющую адрес следующей микрокоманды с учётом условий логических переходов (передач управления);
- **временную**, определяющую время выполнения микрокоманды.

При этом код конкретной операции программы совпадает с адресом первой микрокоманды соответствующей микропрограммы.

Такой подход получил название **микропрограммирования** или "**хранимой логики**". В микропрограммном компьютере логика управления реализуется не в виде электронной схемы, а в виде закодированной информации, находящейся в каком-либо регистре.

11.2. Микропрограммное управление

Идея микропрограммирования, высказанная в 1951 г. Уилксом, нашла широкое применение в машинах серии IBM 360, когда появились надежные и быстродействующие ЗУ для хранения микропрограммы. За счет микропрограммного управления появилась возможность эмуляции системы команд старых моделей ЭВМ.

Достаточно долго неправильно понимались задачи и выгоды микропрограммирования.

Ценность микропрограммирования считалась в том, что каждый потребитель может сконструировать себе из МИК нужный ему набор операций в конкретной задаче. Замена наборов команд достигалась бы заменой информации в ЗУ без каких-либо переделок в аппаратуре. Однако в этом случае программисту необходимо было бы знать все тонкости работы инженера-разработчика компьютера. А основная тенденция развития ЭВМ в связи с автоматизацией программирования состоит в том, чтобы освободить программиста от детального изучения устройств компьютера и в максимальной степени приблизить язык компьютера к языку человека. Поэтому микропрограммные компьютеры считали трудными для пользователя.

Микропрограммный принцип приобрел актуальность, когда были:

- созданы односторонние (читающие) быстродействующие ЗУ с малым циклом памяти;
- микропрограммирование стало рассматриваться не как средство повышения гибкости программирования, а как метод построения системы управления процессором, удобный для инженера-разработчика компьютера.

Программист в своей работе может и не подозревать о микропрограммной структуре компьютера и использовать все средства программного обеспечения и языки программирования самого высокого уровня. Использование микропрограммного принципа позволяет облегчить разработку и изменение логики процессора.

С появлением программного доступа к состоянию процессора после выполнения каждой МИК обеспечивается возможность создания экономичной системы автоматической диагностики неисправностей и появляется способность к эмуляции, т. е. к выполнению на данной ЭВМ программы, составленной в кодах команд другого компьютера. Это достигается введением дополнительного набора МИК, соответствующих командам эмулируемого компьютера.

Эти возможности способствуют распространению методов микропрограммирования при построении УУ в современных компьютерах.

Структурная схема микропрограммного УУ представлена на рисунке 11.2. Преобразователь адреса микрокоманды преобразует код операции команды, присутствующей в данный момент в регистре команд, в начальный адрес микропрограммы, реализующей данную операцию, а также определяет адрес следующей микрокоманды выполняемой микропрограммы по значению адресной части текущей микрокоманды.

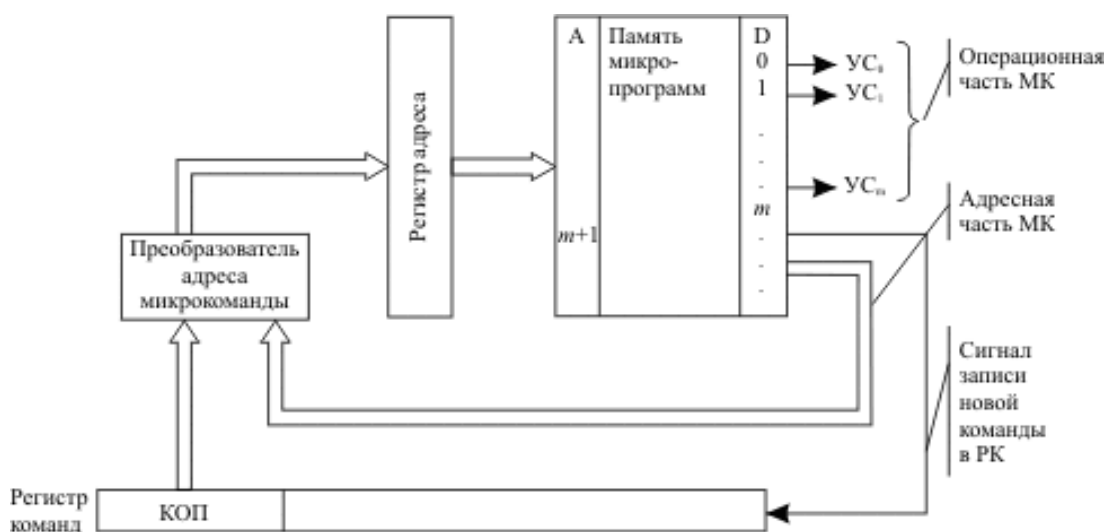


Рис. 11.2. Функциональная схема микропрограммного устройства управления (УС_i - управляющие сигналы, вырабатываемые устройством управления)

Как уже отмечалось, **микропрограмма** записывается в специализированную память УУ – **память микропрограмм или микрокоманд**.

Каждый разряд выходного кода такого ЗУ определяет появление определенного функционального сигнала управления.

Способ управления операциями путем последовательного считывания и интерпретации микрокоманд из ЗУ (наиболее часто в виде микропрограммного ЗУ используют быстродействующие программируемые логические матрицы), а также использования кодов микрокоманд для генерации функциональных

управляющих сигналов **называют микропрограммным**. Компьютеры с таким способом управления обычно называют микропрограммными или с хранимой (гибкой) логикой управления.

К микропрограммам предъявляют **требования функциональной полноты и минимальности**.

Первое требование необходимо для обеспечения возможности разработки микропрограмм любых машинных операций, **второе** связано с желанием уменьшить объем используемого оборудования.

В целом, принцип микропрограммного управления включает следующие позиции:

1. любая операция, реализуемая устройством, является последовательностью элементарных действий - микроопераций;
2. для управления порядком следования микроопераций используются логические условия;
3. процесс выполнения операций в устройстве описывается в форме алгоритма, представляемого в терминах микроопераций и логических условий, называемого микропрограммой;
4. микропрограмма используется как форма представления функции устройства, на основе которой определяются структура и порядок функционирования устройства во времени.

Принцип микропрограммного управления обеспечивает гибкость микропроцессорной системы и позволяет осуществлять проблемную ориентацию микро- и миниЭВМ.

Существуют два вида микропрограммного управления:

- **горизонтальное микропрограммирование** (схема с минимальным кодированием, когда требуется достичь максимальной скорости работы процессора);
- **вертикальное микропрограммирование** (используются сильно закодированные команды, но снижены аппаратные затраты на обработку микрокоманд).

11.2.1. Горизонтальное микропрограммирование

При горизонтальном микропрограммировании – каждому разряду МИК соответствует определенная микрооперация, выполняемая независимо от содержания других разрядов. Микропрограмма может быть представлена в виде матрицы $n \times m$, где n – число ФИ, m – количество МИК, т. е. строка соответствует одной МИК, а столбец – одной МИО (рис. 11.3).

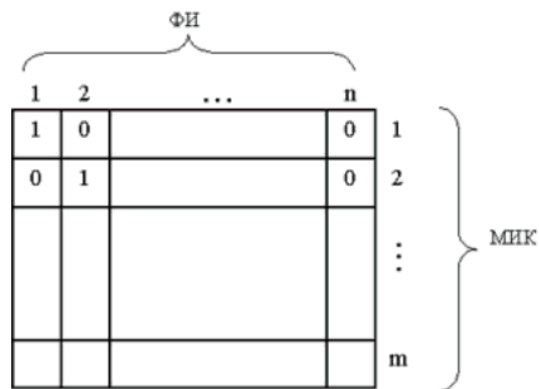


Рис. 11.3. Микропрограмма при горизонтальном микропрограммировании

Примерные значения разрядов МИК приведены на рис. 11.4.

1	2	3	4	5	6	7	8	9	..
---	---	---	---	---	---	---	---	---	----

Рис. 11.4. Значение разрядов МИК (МИО):

1 – гашение сумматора; **2** – гашение указателя переполнения; **3** – обратный код сумматора; **4** – гашение регистра множителя частного; **5** – инвертирование знака; **6** – сдвиг содержимого сумматора влево; **7** – сдвиг содержимого сумматора вправо; **8** – увеличение содержимого сумматора на 1; **9** – чтение из ЗУ в сумматор.

Наличие "1" в пересечении какой-либо строки и столбца означает посылку ФИ в данную МИК, а наличие "0" – его отсутствие.

Размещение "1" в нескольких разрядах МИК означает выполнение нескольких МИО одновременно. Конечно, возбуждаемые МИО должны быть совместимы.

Пусть, например, разряды 9-разрядной МИК принимают следующие значения: 001001101. Тогда, если заданные разряды соответствуют семантике, указанной на рис. 11.3, то МИО, определяемые разрядами 9, 7 и 6, несовместимы.

Для расширения возможностей МИК иногда используют многотактный принцип исполнения МИК. При этом каждому разряду присваивается номер такта, в котором выполняется соответствующая ему МИО, т. е. здесь все совместимые МИО имеют один номер такта. Все остальные такты нумеруются в порядке их естественного выполнения. Однако универсальную нумерацию МИО в МИК указать затруднительно.

Достоинства горизонтального микропрограммирования:

- возможность одновременного выполнения нескольких МИО;
- простота формирования ФИ (без схем дешифрации).

Недостатки:

- большая длина МИК, так как число ФИ в современных компьютерах достигает нескольких сот, и соответственно большой объем ЗУ для хранения МИК;
- из-за ограничений совместимости операций, а также из-за последовательного характера выполнения алгоритмов операций лишь небольшая часть разрядов МИК будет содержать "1". В основном

матрица будет состоять из нулей. Неэффективное использование ЗУ привело к малому распространению горизонтального микропрограммирования.

11.2.2. Вертикальное микропрограммирование

При вертикальном микропрограммировании каждая МИО определяется не состоянием одного разряда, а двоичным кодом, содержащимся в определенном поле МИК. Микрокоманда несколько напоминает формат обычных команд.

Отличие состоит в том, что:

- выполняется более элементарное действие – МИО вместо операции;
- адресная часть (в большинстве случаев) определяет не ячейку памяти, а операционный регистр процессора.

Формат МИК при вертикальном микропрограммировании приведен на рис. 11.5.

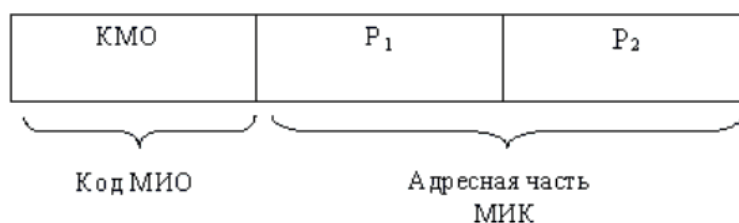


Рис. 11.5. Формат вертикальной МИК

Поля P_1 и P_2 в адресной части МИК указывают двоичные номера операционных регистров, содержимое которых участвует в одной операции. Одно из полей является одновременно и адресом результата. Таким образом, реализация арифметической или логической МИО, указанной в данной МИК, может быть выражена формулой $(P_1) \otimes (P_2) \rightarrow P_1$, или $(P_2) \rightarrow P_1$, где \otimes – символ МИО.

Для МИК обращения к памяти поле P_1 указывает регистр, куда принимается информация, а P_2 – регистр, содержимое которого является адресом обращения к ЗУ. Указанный формат МИК не единственный.

Каждая МИК выполняет следующие функции:

- указывает выполняемую МИО;
- указывает следующую МИО через задание "следующего адреса";
- задает продолжительность МИК;
- указывает дополнительные действия – контроль и т. д.

Обычно в слове МИК имеются четыре зоны, соответствующие указанным функциям. Вообще говоря, некоторые из зон могут указываться неявно, например, выбор очередной МИК может осуществляться из следующей ячейки, продолжительность МИК может быть определена одинаковой для всех МИК и т. д.

Первыми компьютерами с микропрограммным управлением среди отечественных ЭВМ были: МИР, НАИРИ, среди зарубежных – IBM/360, Spectra 70 (70-ые годы).

Вывод:

- Существуют два подхода к реализации управляющего блока процессора
- Аппаратную реализацию УУ эффективнее использовать, если на первом плане быстродействие компьютера

УУ на жесткой логике имеют сложную нерегулярную структуру, которая требует специальной разработки для каждой системы команд и должна практически полностью перерабатываться при любых модификациях системы команд. В то же время оно имеет достаточно высокое быстродействие, определяемое быстродействием используемого элементного базиса.

- Микропрограммное УУ используется, если на первый план выступает требование гибкости реализации команд.

Устройство управления, реализованное по микропрограммному принципу, может легко настраиваться на возможные изменения в операционной части компьютера. При этом настройка во многом сводится лишь к замене микропрограммной памяти. Однако УУ этого типа обладают худшими временными показателями по сравнению с УУ на жесткой логике.

Для реализации УУ этого типа необходимо применение быстродействующих ЗУ небольшого объема (несколько тыс. слов) с временем обращения, соизмеримым с временем выполнения элементарных операций в исполнительных устройствах.

11.3. Принципы управления периферийными устройствами

Вычислительные машины, помимо процессоров и основной памяти, образующих ее ядро, содержат многочисленные периферийные устройства (ПУ): внешние запоминающие устройства (ВЗУ) и устройства ввода/вывода (УВВ).

Передача информации с периферийного устройства в ЭВМ называется операцией ввода, а передача из ЭВМ в ПУ - операцией вывода.

Производительность и эффективность ЭВМ определяются не только возможностями ее процессора и характеристиками ОП, но и составом ПУ, их техническими данными и способами организации их совместной работы с ЭВМ.

В общем случае для организации и проведения обмена данными между двумя устройствами требуются **специальные средства**:

- специальные управляющие сигналы и их последовательности;
- устройства сопряжения;
- линии связи;
- программы, реализующие обмен.

Весь этот комплекс линий и шин, сигналов, электронных схем, алгоритмов и программ, предназначенный для осуществления обмена информацией, называется **интерфейсом**.

В зависимости от типа соединяемых устройств различаются:

- **внутренний интерфейс ЭВМ** (например, интерфейс системной шины, НМД), предназначенный для сопряжения элементов внутри системного блока ПЭВМ;
- **интерфейс ввода-вывода** - для сопряжения различных устройств с системным блоком (клавиатурой, принтером, сканером, мышью, дисплеем и др.);
- **интерфейсы межмашинного обмена** (для обмена между разными машинами) предназначены для сопряжения различных ЭВМ (например, при образовании вычислительных сетей, многопроцессорных и многомашинных комплексов);
- **интерфейсы "человек – машина"** – для обмена информацией между человеком и ЭВМ.

Если интерфейс обеспечивает обмен одновременно всеми разрядами передаваемой информационной единицы (чаще всего байта или машинного слова), он называется параллельным интерфейсом.

Внутренний интерфейс ЭВМ всегда делается параллельным или последовательно-параллельным (если одновременно передается не вся информационная единица, а ее часть, содержащая несколько двоичных разрядов).

Интерфейсы межмашинного обмена обычно **последовательные**, в которых обмен информацией производится по одному биту последовательно.

Для **параллельного и последовательно-параллельного интерфейса** необходимо, чтобы участники общения были связаны многожильным интерфейсным кабелем (**количество жил не меньше числа одновременно передаваемых разрядов – битов**). В последовательных интерфейсах участники

общения связываются друг с другом одно-двухпроводной линией связи, световодом, коаксиальным кабелем, радиоканалом.

Для каждого интерфейса характерно наличие специального аппаратного комплекса (рис. 11.6).

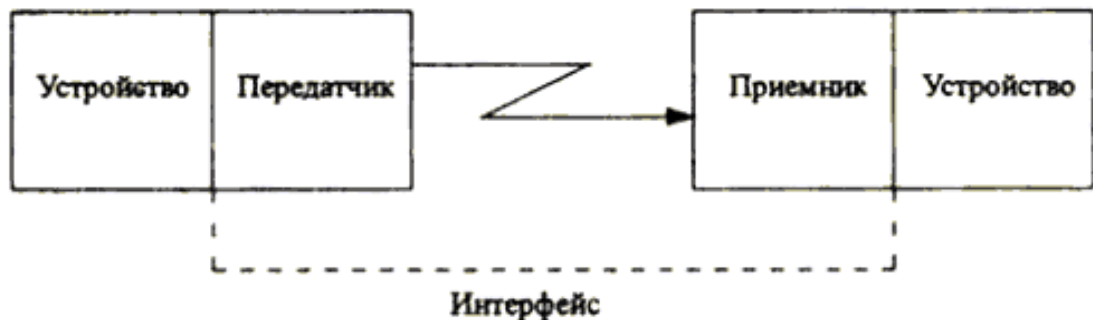


Рис. 11.6. Место интерфейса в аппаратном комплексе

В зависимости от используемых при обмене программно-технических средств интерфейсы ввода-вывода делятся на два уровня: физический и логический (рис. 11.7).



Рис. 11.7. Логический и физический уровни интерфейсов ввода-вывода

В зависимости от степени участия центрального процессора в обмене данными в интерфейсах могут использоваться три способа управления обменом:

- режим сканирования (так называемый "асинхронный" обмен);
- синхронный обмен;
- прямой доступ к памяти.

Режим сканирования ("асинхронный" обмен)

Для внутреннего интерфейса ЭВМ режим сканирования предусматривает опрос центральным процессором ПУ: готово ли оно к обмену, если нет – продолжение опроса ПУ (рис.11.8).

Операция пересылки данных логически слишком проста, чтобы эффективно загружать сложную быстродействующую аппаратуру процессора, в результате чего в режиме сканирования снижается производительность вычислительной машины.

Вместе с тем при пересылке блока данных процессору приходится для каждой единицы передаваемых данных (байт, слово) выполнять довольно много команд, чтобы обеспечить буферизацию данных, преобразование форматов, подсчет количества переданных данных, формирование адресов в памяти и т.п. В результате скорость передачи данных при пересылке блока данных даже через высокопроизводительный процессор может оказаться неприемлемой для систем управления, работающих в реальном масштабе времени.

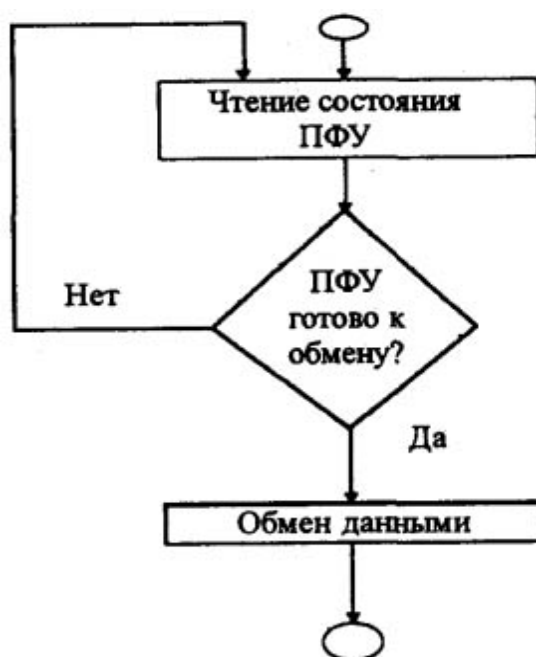


Рис. 11.8. Алгоритм сканирования

Режим сканирования упрощает подготовку к обмену, но имеет ряд недостатков:

- процессор постоянно задействован и не может выполнять другую работу;
- при высоком быстродействии ПУ процессор не успевает организовать обмен данными.

Синхронный режим

В синхронном режиме ЦП выполняет основную роль по организации обмена, однако, в отличие от режима сканирования не ждет готовности ПУ, а осуществляет другую работу. Когда возникает необходимость во взаимодействии с ПУ, внешнее устройство с помощью соответствующего прерывания обращает на себя внимание ЦП.

Для быстрого ввода-вывода блоков данных и разгрузки процессора от управления операциями ввода-вывода используют **прямой доступ к памяти (DMA – Direct Memory Access)**.

11.4. Прямой доступ к памяти

В режиме прямого доступа к памяти используется специализированное устройство – **контроллер прямого доступа к памяти (КПДП)**, который перед началом обмена программируется с помощью ЦП: в него передаются адреса основной памяти и количество передаваемых данных. Затем ЦП отключается от КПДП, разрешив ему автономное функционирование, и до окончания обмена может выполнять другую работу. Об окончании обмена КПДП сообщает процессору. В этом случае участие ЦП косвенное. Обмен ведет КПДП.

На рис. 11.9 приведена схема взаимодействия устройств микропроцессорной системы в режиме ПДП.



Рис. 11.9. Взаимодействие устройств в режиме прямого доступа к памяти

Прямой доступ к памяти (ПДП):

- освобождает процессор от управления операциями ввода-вывода;
- позволяет осуществлять параллельно во времени выполнение процессором программы с обменом данными между ВнУ и основной памятью;
- производит обмен данными со скоростью, ограничиваемой только пропускной способностью основной памяти и ВнУ.

При работе в **режиме прямого доступа к памяти КПДП выполняет следующие функции:**

- принимает запрос на прямой доступ к памяти от внешнего устройства;
- формирует запрос микропроцессору на захват шин СМ;
- принимает сигнал, подтверждающий вход микропроцессора в состояние захвата (перехода в z-состояние, при котором процессор отключается от системной магистрали);
- формирует сигнал, сообщающий ВнУ о начале выполнения циклов ПДП;

- выдает на шину адреса системной магистрали адрес ячейки ОП, предназначенной для обмена;
- вырабатывает сигналы, обеспечивающие управление обменом данными;
- по окончании ПДП контроллер либо организует повторение цикла ПДП, либо прекращает режим ПДП, снимая запросы на него.

11.2. Способы организации совместной работы периферийных и центральных устройств

Связь двух ЭВМ и внешнего устройства или двух ЭВМ друг с другом может быть организована в трех режимах: *симплексном*, *полудуплексном* и *дуплексном*.

В *симплексном режиме* передача данных может вестись только в одном направлении: один передает, другой принимает.

Полудуплексный режим позволяет выполнять поочередный обмен данными в обоих направлениях. В каждый момент времени передача может вестись только в одном направлении: один передает, другой принимает. И пока передача не закончилась, принимающий ничего не может сообщить передающему. Заканчивая передачу, передающая ЭВМ пересылает приемной специальный сигнал "перехожу на прием" (или просто "прием" – как выглядит этот сигнал, должны "договориться" между собой коммуникационные программы). Этот сигнал должен быть известен обеим ЭВМ, т.е. сигнал окончания связи должен выглядеть одинаково у ЭВМ, находящихся на связи. Затем они могут поменяться ролями. Этот режим является самым простым. Если во время передачи в приемной ЭВМ возникла нештатная ситуация (появилась ошибка в передаваемых данных, коммуникационная программа не успела обработать принятый байт до поступления следующего, при распечатке принимаемых данных одновременно с приемом замяло бумагу в принтере и др.), то принимающая ЭВМ неспособна сообщить об этом передающей до появления сигнала окончания передачи. Вся информация, передаваемая после появления нештатной ситуации, теряется. После устранения неполадок передачу приходится повторять. Поэтому при обмене большими объемами информации приходится все передаваемые данные делить на блоки и контролировать прохождение каждого блока. Общее время обмена информацией при этом возрастает.

Дуплексный режим позволяет вести передачу и прием одновременно в двух встречных направлениях.

В симплексном режиме может быть осуществлена связь, например, между ЭВМ и принтером, клавиатурой и ЭВМ, или ЭВМ и дисплеем, а также между двумя ЭВМ, находящимися всегда в односторонней связи.

Для организации симплексного режима необходимо, чтобы передатчик одной ЭВМ был связан с приемником другой ЭВМ двухпроводной линией связи.

Для организации полудуплексного режима можно применить:

- либо специальное коммутационное устройство у каждой ЭВМ, переключающее линию связи с выхода передатчика на вход приемника и обратно,
- либо линию связи с большим количеством проводов (например, трехпроводную, в которой один провод связывает передатчик первой ЭВМ с приемником второй. Другой провод связывает приемник первой ЭВМ с передатчиком второй, а третий является общим проводом и называется *информационная земля*).

Для организации дуплексного режима необходимо, чтобы аппаратные средства (в состав которых входит и канал связи) обеспечивали возможность одновременной передачи информации во встречных направлениях. Например,

дуплексный режим может быть реализован при связи ЭВМ с принтером, если дополнительно к информационной связи канал обеспечивает передачу управляющего сигнала готовности принтера (сигнал DSR).

Сопряжение ЭВМ с каналом связи осуществляется с помощью последовательного (RS-232) или параллельного (Centronics) интерфейса, каждый из которых может обеспечить работу сопрягаемых устройств в любом из рассмотренных режимов - все зависит от типа используемого канала связи и технологии его использования.

Способ, с помощью которого интерфейс обеспечивает связь в заданном режиме, называется *протоколом*:

- **Программный протокол XON/XOFF** основан на использовании программно или аппаратно-реализуемых сигналов XON (код ASCII 17d или 11h) и XOFF (код ASCII 19d или 13h), вырабатываемых принимающим устройством.
- **Программно-аппаратурный протокол RTS/CTS** используется для синхронного обмена информацией (все ранее рассмотренные протоколы реализовали асинхронный обмен) между ЭВМ и ее внешним устройством. В соответствии с этим протоколом производится взаимное оповещение взаимодействующих устройств о выполненных ими действиях с помощью сигналов: **DTR** (Data Terminal Ready) – "ЭВМ готова к выходу на связь", **DSR** (Data Set Ready) – "Внешнее устройство готово", **RTS** (Request To Send) – "ЭВМ готова к обмену информацией" и **CTS** (Clear To Send) – "Готов к обмену".

Четыре управляющих сигнала: DTR, DSR, RTS, CTS вырабатываются ЭВМ и внешним устройством. Анализ поступивших сигналов производится коммуникационной программой. Передаваемые данные в синхронном режиме могут сопровождаться управляющим сигналом от передающего или приемного устройства (TXD - Transmitted Data и RXD – Received Data соответственно).

В синхронном дуплексном режиме взаимодействующие устройства работают наиболее эффективно, так как выработка большого количества управляющих сигналов позволяет им оперативно информировать друг друга об успешности выполнения каждого шага.

Для взаимодействия со сложными внешними устройствами могут предусматриваться и дополнительные сигналы, например, для модема протокол DTS/CTS содержит сигналы: DCD (Data Carrier Detected) – "Есть несущая частота" и RI (Ring Indicator) – "Индикатор звонка", информирующий ЭВМ, что по телефонной линии, подключенной к модему, поступили сигналы вызова (звонка), т.е. электрические сигналы, параметры которых отличаются от несущей.

Для того чтобы обеспечить взаимодействие ЭВМ по наиболее сложному протоколу DTS/CTS, последовательный интерфейс RS-232 предусматривает обмен всеми перечисленными сигналами.

Но тот же интерфейс позволяет реализовать обмен и по любому другому протоколу, например, протоколу DTR, для которого в симплексном режиме требуется двух- или трехпроводная линия связи.