МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

А.А.АВДЮХИН, Е.В.ДУШУТИНА, А.В.ЖУКОВ

ИНТЕРФЕЙСЫ ПЕРИФЕРИЙНЫХ УСТРОЙСТВ ЭВМ

Учебное пособие

Санкт-Петербург
Издательство СПбГПУ
2011

МИНИСТЕРСТВООБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

———

САНКТ-ПЕТЕРБУРГСКИЙ
ГОСУДАРСТВЕННЫЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

А.А.АВДЮХИН, Е.В.ДУШУТИНА, А.В.ЖУКОВ

ИНТЕРФЕЙСЫ ПЕРИФЕРИЙНЫХ УСТРОЙСТВ ЭВМ

Учебное пособие

Санкт-Петербург
Издательство СПбГПУ
2011

УДК 004.32 (075.8)

Авдюхин А.А., Душутина Е.В., Жуков А.В. Аппаратно-программные средства ввода/вывода: Учеб. пособие. СПб.: Изд-во СПбГПУ, 2011. 153 с.

Учебное пособие соответствует государственному образовательному стандарту дисциплины «ЭВМ и периферийные устройства» базовой части профессионального цикла (направление подготовки 230100 — информатика и вычислительная техника.

Рассмотрены основные принципы, способы и средства обмена информацией между центральной частью ЭВМ и множеством устройств вычислительной системы, которые принято рассматривать как периферийные устройства. Среди этого множества имеются периферийные устройства, знакомые всем по общению с персональными компьютерами, устройства сопряжения с объектами различного назначения, а также «не совсем периферийные» устройства и подсистемы, расположенные в непосредственной близости от центральной части ЭВМ, подключенные к системной шине и участвующие в процессах ввода/вывода информации. Несмотря на значительные различия между этими участниками процессов обмена данными имеется ряд установившихся принципов организации такого обмена, которые положены в основу унифицированных интерфейсов ввода/вывода, на которые направлено главное внимание авторов настоящего пособия.

Учитывая сложившуюся ситуацию на рынке вычислительной техники, большинство примеров в пособии иллюстрирует организацию интерфейсов персональных ЭВМ, построенных на процессорах архитектурного ряда IntelX86…PentiumX. В пособии дается краткое описание основных интерфейсов персональных компьютеров и некоторых функциональных узлов, поддерживающих работу с периферийными устройствами. Во втором и третьем разделах описаны лабораторные работы, которые предлагаются для закрепления лекционного курса.

Пособие предназначено для студентов, проходящих подготовку по направлению 230100, которые изучают на факультете технической кибернетики, кроме упомянутой дисциплины, также курс «Интерфейсы внешних устройств». Это же пособие рекомендуется студентам, обучающимся по направлению 220400 — управление в технических системах, которые изучают дисциплину «Средства сопряжения вычислительных систем с объектом управления»

Табл. , Ил. , Библиогр.: назв.

Печатается по решению редакционно-издательского совета Санкт-Петербургского государственного политехнического университета.

ВВЕДЕНИЕ

Периферийные устройства (ПУ) являются сложными и дорогостоящими изделиями вычислительной техники. Например, если рассмотреть только системный блок современной персональной ЭВМ, то увидим, что на ПУ приходится более 70 % стоимости аппаратных средств, а если учесть другие устройства, необходимые на рабочем месте (монитор, принтер и другие), то получится, что почти вся стоимость комплекса сосредоточена в периферийном оборудовании. Доля программных средств, обеспечивающая работу ПУ, во всем комплексе программного обеспечения ЭВМ также велика. Эффективность вычислительного комплекса в целом в значительной степени определяется организацией работы периферийного оборудования.

Физические принципы, конструктивные особенности периферийных устройств настолько разнообразны, что в рамках короткого курса невозможно подробно рассмотреть эти вопросы. Поэтому целесообразно сделать акцент на изучении интерфейсов. В нашем университете учебный процесс всегда был ориентирован на подготовку инженеров, создающих новую технику. Маловероятно, чтобы наш выпускник занимался разработкой собственно периферийного устройства, такого, как дисплей или принтер. Легче представить, что кому-то придется разрабатывать электронные схемы управления для таких изделий. А вот подключение их к системе, разработка нестандартных контроллеров связи ЭВМ с объектами различного назначения, разработка программных средств управления ими — все это повседневная практика целой армии инженеров. Для успешной работы в этой области необходимо уметь разбираться в технических описаниях как самого объекта, так и интерфейса, с помощью которого этот объект обменивается информацией с ЭВМ. При этом необходимо также иметь ясное представление о физических процессах, протекающих в этих объектах.

Несмотря на значительное разнообразие физических принципов работы ПУ, их подключение к системе и порядок взаимодействия с центральной частью ЭВМ имеет много общего. Это общее, по мнению авторов, должно быть отражено в курсе «ЭВМ и периферийные устройства» и более подробно рассмотрено в курсе «Интерфейсы внешних устройств». Знание основ организации ввода/вывода и, в частности, знакомство с основными интерфейсами позволяет лучше понять особенности вычислительных комплексов и систем.

Системы управления, включающие микропроцессор или компьютер в качестве устройства преобразования, хранения, переработки и представления информации (терминология ГСП — государственной системы промышленных приборов и средств автоматизации), доминируют в технике

систем управления. Поэтому если вычислительная машина (система) предназначена для управления станком, роботом и иными технологическими объектами или процессами, то и в этом случае организация подсистем обмена сигналами с этими объектами обладает первостепенной важностью. При этом подсистемы ввода/вывода имеют много общего с аналогичными подсистемами обычных ЭВМ. Различия обусловлены характером объектов управления (в одном случае это, например, принтер, а в другом — робот-штабелер на автоматизированном складе).

Наиболее распространенными и доступными ЭВМ в настоящее время являются персональные ЭВМ, построенные на процессорах ряда Intel 8X86…Pentium X и их аналогах, которые для краткости называют IBM/PC или просто PC (персональный компьютер — ПК). Поэтому в качестве примеров использованы интерфейсы и контроллеры, применяемые в этих системах. Это тем более оправдано, что по тем же стандартам выполнены ЭВМ других классов, например, управляющие (индустриальные) ЭВМ, промышленные программируемые контроллеры и др.

В предлагаемом пособии содержится изложение общих принципов обмена информацией в ЭВМ, дается краткое описание основных интерфейсов персональных компьютеров и некоторых функциональных узлов, поддерживающих работу с периферийными устройствами.

Пособие предназначено для студентов, проходящих подготовку по направлению 230100 (информатика и вычислительная техникаё) , которые изучают на факультете технической кибернетики курс «ЭВМ и периферийные устройства», а также курс «Интерфейсы внешних устройств». Это же пособие рекомендуется студентам, обучающимся по направлению 220400 (управление в технических системах), которые изучают дисциплину «Средства сопряжения вычислительных систем с объектом управления»

В последние годы по данной тематике для поддержки лабораторных работ были изданы учебные пособия [1, 2], тиражи которых повторялись. Настоящее издание представляет собой переработанное объединение указанных пособий, выполненное на основании опыта проведения занятий со студентами 3…4 курсов.

Для усвоения материала пособия студенты должны иметь достаточную подготовку по информатике и микросхемотехнике. Желательно иметь навыки программирования на языке ассемблера.

Пособие состоит из трех разделов. В первом — изложены общие вопросы организации обмена информацией в ЭВМ, описаны некоторые интерфейсы, характерные для персональных компьютеров. Эти материалы можно использовать для подготовки к экзамену по теоретической части курса.

Во втором разделе описаны лабораторные работы, которые предлагаются для закрепления лекционного курса. Первые две работы,

посвященные работе с регистрами ПУ вручную и в автоматическом режиме по опросу готовности и в режиме прерывания, являются вводными работами и выполняются всей группой одновременно. Для выполнения этих работ не требуется специального оборудования, и они могут выполняться на любых рабочих местах. Остальные работы для своего выполнения требуют, как правило, дополнительных устройств, не входящих в стандартный комплект ПК. Поэтому они выполняются по графику.

В третьем разделе предлагаются лабораторные работы, направленные на освоение традиционной системы прерываний, реализованной в архитектуре персональных компьютерах и в аналогичных по архитектуре управляющих ЭВМ.

Все работы рассчитаны на индивидуальное выполнение в течение четырех академических часов. Для успешного выполнения работ требуется домашняя подготовка по заданию, полученному на предыдущем посещении. Предполагается, что выполняющие работу студенты освоили приемы программирования и отладки программ на языке ассемблера (A86 или TASM).

Работы завершаются оформлением отчета, в котором должны быть представлены следующие материалы:

- цели работы,
- укрупненная функциональная схема исследуемого объекта (или графическое представление программной модели),
- формулировка задания,
- тексты отлаженных программ с комментариями,
- если требуется, временные диаграммы с оцифровкой осей,
- содержательные выводы по проделанной работе.

1. ОБЩАЯ ХАРАКТЕРИСТИКА ИНТЕРФЕЙСОВ ПЭВМ

1.1. Основные термины. Классификация интерфейсов

При рассмотрении проблем, связанных со взаимодействием элементов систем обработки информации, систем передачи данных и иных систем, построенных на основе вычислительных машин, используется ряд понятий и соответствующих терминов. Некоторые из них приведены в этом параграфе. Следует иметь в виду, что в различных литературных источниках можно встретить различающиеся толкования одинаковых терминов.

Интерфейс. Одно из центральных понятий. Существует целый ряд его определений, в том числе закрепленных ГОСТ'ами. Упрощенно интерфейс можно определить как совокупность средств и правил, обеспечивающих взаимодействие компонентов вычислительной системы или сети. Под средствами понимается как аппаратура, так и программно-алгоритмическое обеспечение.

Протокол. Понятие, обозначающее совокупность правил, алгоритм взаимодействия устройств, подсистем или специалистов при реализации определенных функций (в нашем случае — при обмене информацией).

Линия интерфейса. Техническая («в металле») реализация интерфейса представляется как совокупность линий интерфейса (обычно это электрические проводники) с элементами подключения к устройствам, между которыми устанавливается связь.

Шина. Это совокупность линий, сгруппированных по функциональному признаку (шина данных, шина адреса и др.). Иногда шиной называют совокупность всех линий интерфейса. В этом же смысле употребляется термин **магистраль**. Обычно, когда говорят о шине или магистрали, то предполагают, что они допускают параллельное подключение к ним нескольких устройств, обменивающихся информацией. Т.е. линии шины являются линиями коллективного пользования.

Канал. Этим термином обозначается среда распространения сигналов, что соответствует интуитивному пониманию. Однако термины **канал связи** или **канал передачи данных** охватывают кроме среды еще и аппаратуру, обеспечивающую связь или передачу данных. В вычислительной технике существуют **каналы ввода-вывода,** под которыми понимаются устройства (часто это специализированные ЭВМ), обеспечивающие передачу данных между основной памятью и периферийными устройствами.

Контроллер. Это общий термин для обозначения устройств управления. Когда речь идет об интерфейсах, то под контроллерами понимаются устройства управления внешними или периферийными устройствами. Часто их называют контроллерами ввода/вывода. В

повседневной практике для контроллеров разных периферийных устройств закрепились разные наименования, например, карта, адаптер и некоторые другие, но по сути дела все это контроллеры.

В соответствии с ГОСТ 26.016—81 установлено четыре **основных** классификационных признака интерфейсов [3]:

- способ соединения компонентов (магистральный, радиальный, цепочечный, смешанный),
- способ передачи информации (параллельный, последовательный, параллельно-последовательный),
- принцип обмена информацией (асинхронный, синхронный, изохронный (цикл обмена повторяется через равные промежутки времени)),
- режим передачи информации (двухсторонняя одновременная, то же поочередная, односторонняя).

Имеется ряд других классификационных признаков, более полно характеризующих интерфейсы (функциональное назначение, логическая организация, функциональная организация, физическая реализация и др.). Отметим классы по функциональному назначению:

- системные (машинные или ввода/вывода) интерфейсы ЭВМ,
- сосредоточенных магистральных мультипроцессорных систем,
- периферийного оборудования (общего назначения и специализированные),
- сетей передачи данных,
- модульных систем и приборов,
- локальных вычислительных сетей (ЛВС),
- распределенных систем (в том числе управления).

Эти классы можно бы укрупнить: машинные, периферийного оборудования и сетевые.

1.2. Принципы взаимодействия устройств на шине.

Устройства (компоненты) ЭВМ обмениваются между собой информацией при помощи системы шин. Обычно ЭВМ содержит шины процессора, памяти, ввода-вывода и, возможно, ряд других. Представление о составе ЭВМ и системе ее шин дает структурная схема (рис.1.1). Шины, показанные на этом рисунке, являются шинами межмодульного обмена. В них используется параллельный способ передачи основной информации. На физическом уровне элементарный акт взаимодействия устройств на шине сводится к установлению на линиях шины потенциалов стандартного уровня.

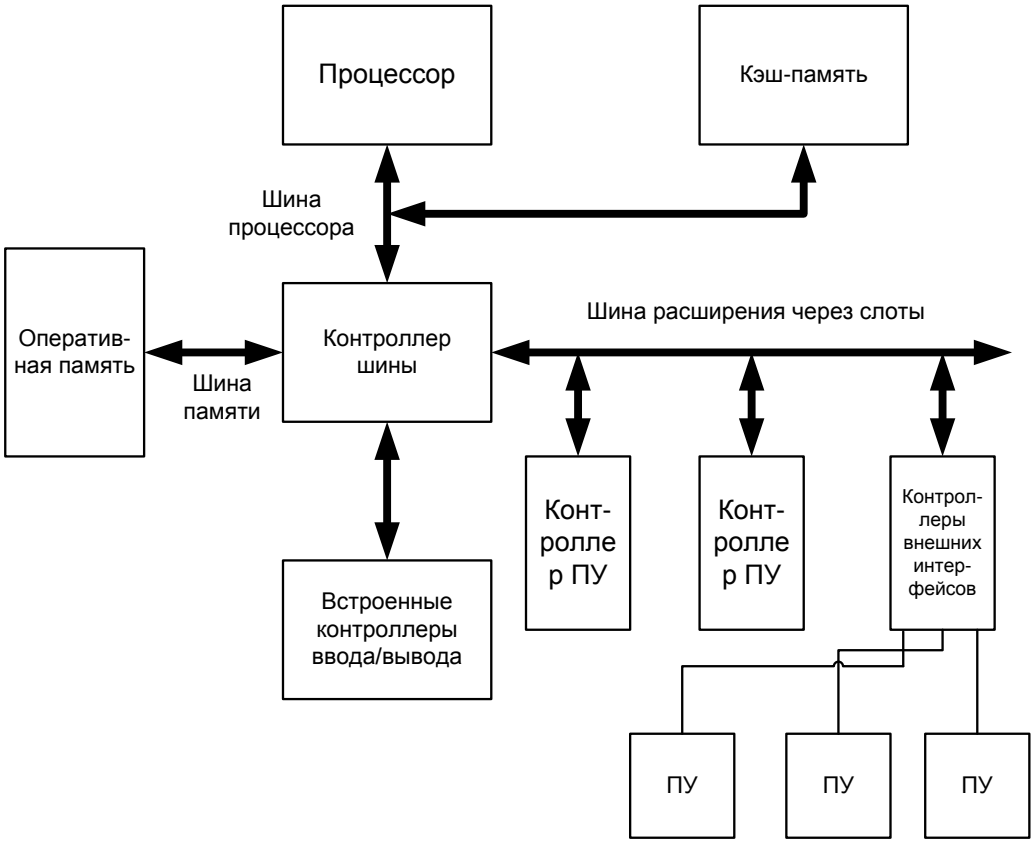

Рис.1.1. Система шин ЭВМ

Передача информации между устройствами ЭВМ является по существу межрегистровой передачей, знакомой студентам из курса схемотехники. Описание работы шины обычно ведется от имени «наблюдателя», находящегося в устройстве, осуществляющем управление процессом обмена. Часто в этой роли выступает центральный процессор ЭВМ.

Таким образом, на шине возможны такие виды взаимодействия устройств как чтение и запись. Например, чтением является передача информации из ячейки памяти или регистра контроллера ПУ в один из регистров процессора при выполнении команды чтения. Записью является передача информации в противоположном направлении.

Можно указать еще на один вид взаимодействия устройств на шине. Это установка управляющих сигналов, обеспечивающих вход в прерывание, в режим прямого доступа к памяти или в режим захвата шины и выход из этих режимов. Сама по себе работа в режимах прерывания, прямого доступа или захвата шины является по существу последовательностью передач информации типа чтение и запись.

Организация системного магистрального интерфейса и обмен информацией на шине основывается на ряде принципов, которые являются общими для всех шин.

Принцип подчиненности («ведущий-ведомый») заключается в том, что при взаимодействии устройств одно из них играет роль ведущего, управляя всем ходом обмена информации, а другое (или несколько других) являются подчиненными. В разных стандартах для понятий ведущий и ведомый используются разные термины. Так в шине Q-BUS это активное и пассивное устройства, в шине ISA это задатчик и исполнитель, в шине PCI это инициатор и цель. Термины «задатчик» и «исполнитель» являются стандартными и рекомендуются для использования в технической документации.

Принцип квитирования («запрос-ответ») заключается в том, что в определенных ситуациях продолжение цикла взаимодействия зависит от наличия ответного сигнала (quit) от ведомого устройства. Использование ответного сигнала придает обмену асинхронный характер, позволяет надежно и без излишних временных затрат осуществлять обмен информацией с устройствами различного быстродействия.

Принцип унификации состоит в обеспечении информационной, электрической и конструктивной совместимости компонентов вычислительной системы.

Информационная совместимость [3] — это согласованность взаимодействия функциональных элементов вычислительной системы в соответствии с логическими условиями. Логические условия определяют структуру и состав шины, порядок взаимодействия устройств в различных режимах работы интерфейса, кодирование и форматы команд, данных, адресов и информации состояния устройств, причинно-следственные связи и временные соотношения между сигналами управления. Т.е. логические условия информационной совместимости определяют в целом функциональную и структурную организацию интерфейса и сложность схемотехники и программного обеспечения интерфейса.

Электрическая совместимость — это согласованность параметров электрических сигналов в линиях интерфейса. Ограничения налагаются как на статические параметры (уровни сигналов), так и на динамические (длительности фронтов, задержки). При этом учитывается техническая реализация устройств, взаимодействующих на шине, т.е. потребление энергии во входных цепях и нагрузочная способность выходных каскадов элементов, задержки распространения сигнала. Учитывается также пространственное размещение устройств, так как от этого зависят электрические параметры сигналов.

Конструктивная совместимость — это согласованность конструктивных параметров изделий, входящих в состав интерфейса, предназначенных для механического их соединения. Унификация распространяется на типы соединителей (вилки и розетки, которые в повседневном общении чаще называют разъемами), конструкции плат, каркасов, стоек и т.п. Обычно

требования к конструктивным параметрам являются важной составной частью спецификации (описания) интерфейса.

Контроллеры ввода/вывода. К основным принципам организации магистральных интерфейсов (шин) следует отнести также применение контроллеров ввода/вывода — устройств, с помощью которых центральная часть ЭВМ через шину связана с периферийными устройствами. Контроллеры в своем составе имеют регистры, через которые и передается информация. Эти регистры часто называют регистрами ввода/вывода. Обращение к ним производится по адресам адресного пространства ввода/вывода. В некоторых архитектурах адресные пространства основной памяти и ввода/вывода объединены. Через эти регистры осуществляется чтение/запись данных, управление периферийным устройством, а также проверка его состояния. Соответственно указанные регистры называются регистрами данных, управления и состояния.

В практике последних лет за регистрами ввода/вывода закрепилось наименование «порты». Происхождение этого термина связано с традиционным его употреблением для обозначения аппаратуры сопряжения шины с устройством ввода-вывода. Для программиста операции записи в порт или чтения порта сводились к обращению по соответствующему адресу, поэтому регистры контроллера естественно отождествлялись с портами ввода/вывода.

Адресные пространства ввода/вывода и основной памяти могут быть объединенными или раздельными. Примером единого адресного пространства является межмодульный параллельный интерфейс (МПИ) [3]. Для регистров ввода/вывода там отведено 8К старших адресов адресного пространства. При этом команды чтения/записи этих регистров не отличаются от соответствующих команд обращения к памяти по записи их в тексте программы. Состав управляющих сигналов, которыми обмениваются устройства на шине при обмене с памятью и ПУ при выполнении этих команд, и временные диаграммы имеют лишь небольшие отличия.

Примерами разделенных адресных пространств памяти и ввода/вывода являются адресные пространства шин ISA и PCI, которые рассмотрены в последующих разделах настоящего пособия. В системе команд микропроцессоров, поддерживающих обмен информацией по этим шинам, для обращения к портам имеются специальные команды In (input, т.е. чтение или ввод) и Out (output, т.е. запись или вывод). Циклы ввода/вывода (временные диаграммы формирования сигналов на линиях шины при выполнении этих команд) отличаются от циклов обращения к памяти как по временным характеристикам, так и по составу управляющих сигналов на линиях шины, что и обеспечивает разделение адресных пространств.

Процессор и периферийные устройства работают относительно независимо, а главное, в разном темпе. Для успешной передачи информации между ними необходимо, чтобы шина была свободна, а участники обмена

были готовы. Избежать потерь информации при обмене помогает использование следующих режимов или видов обмена:

- режим программного опроса готовности,
- режим прерывания,
- режим прямого доступа к памяти (ПДП), а также похожий на него режим захвата шины.

В режиме программного опроса готовности по программе периодически опрашивается регистр состояния ПУ, в котором содержится информация о готовности его к обмену. При выявлении готовности устройства процессор по программе выполняет требуемое действие (чтение или запись).

Ниже приведен пример фрагмента программы на языке ассемблера, в котором в режиме опроса готовности производится обмен информацией с ПУ. В скобки заключены команды, относящиеся только к записи. Предполагается, что устройство, из которого читается (в которое записывается) информация, имеет регистр состояния (порт) с адресом **status** и регистр данных с адресом **buf.** В регистре состояния разряд n несет информацию о готовности к чтению (записи). Значение bit n = 1 свидетельствует о готовности устройства. Напомним, что команды **in** и **out** работают так, что обмен производится через регистр **al/ax** процессора, адрес порта задается либо в регистре **dx**, либо, если адрес не больше 0ff, непосредственно в самой инструкции.

```
    mov dx, status    ; адрес регистра состояния в dx
metka:
    in  al, dx        ; чтение регистра состояния
    test al, bit n    ; если устройство не готово, то
    jz metka          ; продолжить опрос
  (mov  al, bl        ; подготовка к выводу байта из bl)
    mov dx, buf       ; адрес регистра данных — в dx
    in  al, dx        ; чтение данных из ПУ
  (out dx, al         ; запись  данных в ПУ)
```

В результате выполнения этого фрагмента байт из порта **buf** окажется в регистре **al** (в случае вывода байт из регистра **bl** будет записан в порт **buf**). Таким образом, в режиме опроса готовности инициатором обмена является программа, а управление обменом осуществляется также программой.

Обмен в режиме прерывания производится по программе, которая выполняется по запросу от периферийного устройства. При этом процессор выполняет свою основную работу, не теряя времени на постоянные обращения к регистрам состояния своих ПУ. В целом схема взаимодействия

ПУ и центральной части ЭВМ выглядит следующим образом. Периферийное устройство выставляет сигнал запроса, который, в конечном счете, поступает на вход процессора. Последний, если прерывания не запрещены, заканчивает какие-то действия, которые нельзя прерывать (например, выполняет текущую команду до полного завершения, чтобы избежать потери данных), а затем, позаботившись о сохранении необходимых для возврата данных, переходит на выполнение той программы, которую требует ПУ. Эта программа называется программой обработки запроса на прерывание или просто обработчиком. Этот обработчик и выполняет необходимые действия по обмену информацией. Более подробно организация работы ЭВМ при входе в прерывание будут рассмотрена в п.1.4. Таким образом, инициатором обмена в режиме прерывания является ПУ, а управляет обменом — программа, но не основная, а специально заготовленная для обработки запроса на прерывание (обработчик).

Обмен информацией в режиме прямого доступа к памяти (ПДП) выполняется также по запросу от ПУ, но, в отличие от режима прерывания, управление обменом осуществляет не центральный процессор, а специальный контроллер ПДП, который формирует все необходимые для обмена сигналы. При этом центральный процессор находится в пассивном состоянии до окончания цикла ПДП. В буквальном смысле ПДП — это обмен данными между памятью и периферийным устройством, этим обменом и управляет контроллер ПДП. Во многих стандартах существует режим очень похожий на ПДП, который называется режимом прямого управления шиной или режимом захвата шины. В этом режиме устройство, выставившее запрос, после получения разрешения от задатчика (владельца шины) само становится задатчиком и управляет всеми процессами, управляя обменом (не обязательно с памятью).

Собственно передача информации на физическом уровне сводится к установлению на линиях шины в соответствии с принятым протоколом информационных и управляющих сигналов. Для всех шин характерна следующая последовательность действий (рис. 1.2). Ведущее устройство устанавливает адресные сигналы на шине адреса. Затем с достаточной для распознавания «своего» адреса ведомыми устройствами задержкой ведущее устанавливает командные сигналы на шине управления. Ведомое устройство за время t1 распознает факт обращения к себе, в случае надобности выдает ответные сигналы через время tq. Если выполняется чтение, то с задержкой t2, определяемой внутренними свойствами, ведомое устройство устанавливает сигналы на шине данных, которые фиксируются во внутренних регистрах ведущего устройства. Если выполняется запись, то ведущее устройство помещает сигналы на шине данных, которые под действием командных сигналов записываются в регистры ведомого устройства.

В простых интерфейсах прошлых поколений командные сигналы по шине управления передавались непосредственно в виде управляющих сигналов, таких как стробы чтения или записи. При этом фиксация информации на принимающей стороне производится по спаду строба. Для цикла чтения устанавливается ограничение сверху на время t4 удержания ведомым устройством данных на шине после снятия строба: шина данных должна быть освобождена для следующего цикла. Для цикла записи устанавливается ограничение снизу для времени t5 удержания ведущим устройством данных на шине для надежности записи.

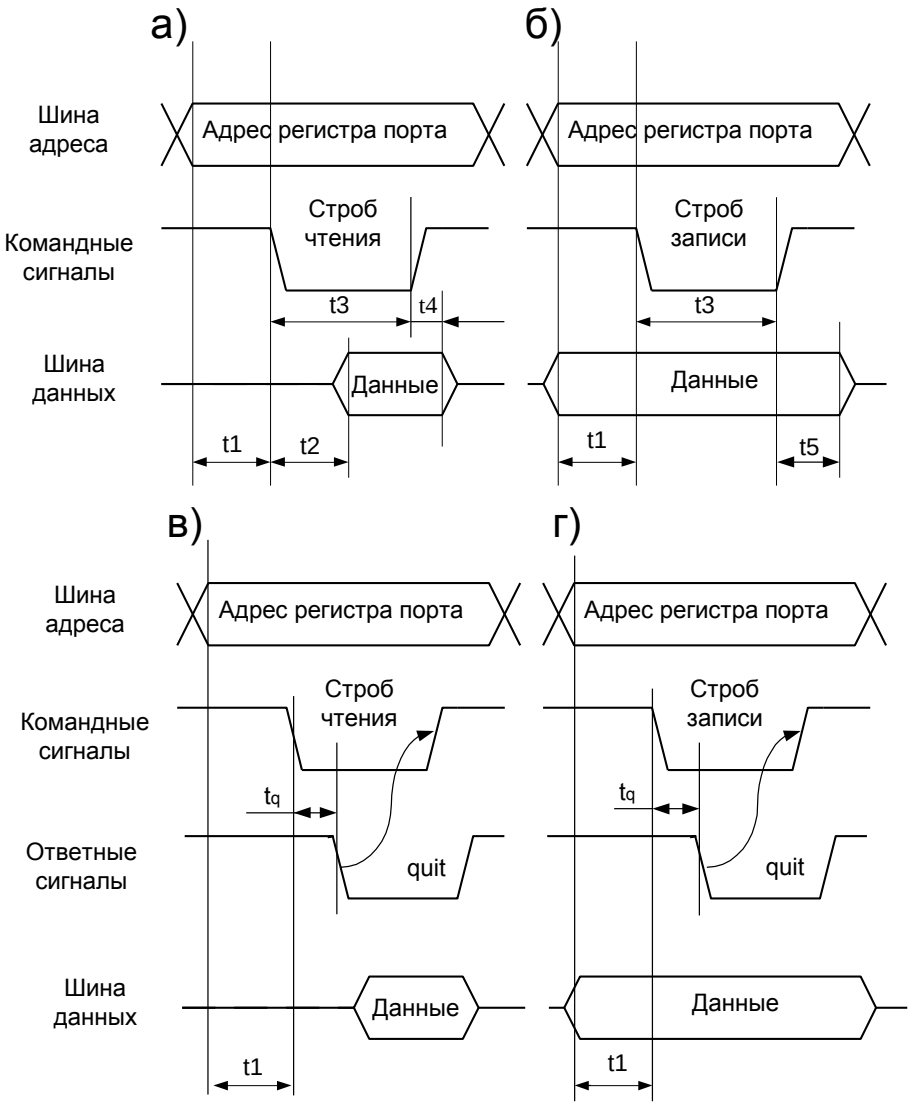

Рис.1.2. Временные диаграммы чтения/записи: а — синхронный цикл чтения; б — синхронный цикл записи; в — асинхронный цикл чтения; г — асинхронный цикл записи

При синхронном обмене длительности стробов (t3) являются фиксированными. При асинхронном обмене t3 зависит от момента прихода ответного сигнала: строб снимается только после получения ведущим устройством сигнала quit. Время tq определяется быстродействием ведомого устройства. Отсутствие ответа в течение заданного максимума расценивается как сбой в работе шины. В работе большинства шин, нашедших практическое применение, сочетаются синхронный и асинхронный принципы.

1.3. Шины расширения ЭВМ

1.3.1. Системные интерфейсы и шины расширения

Интерфейсы, предназначенные для обмена информацией между центральным процессором и основными устройствами ЭВМ, получили название системных интерфейсов или шин. По другому определению системной шиной считается та, через которую передается основной объем информации в ЭВМ. Системный интерфейс во многом определяет архитектуру ЭВМ в целом. В современных ЭВМ, в частности, в ПЭВМ, понятие системной шины оказывается размытым из-за того, что для повышения производительности целый ряд устройств подключается с помощью локальных шин. Например, видеоподсистема во многих современных ПЭВМ подключена к процессору с помощью параллельного интерфейса AGP. Несколько ранее для этой цели широко использовалась локальная шина VL-bus. Для обеспечения работы современных дисплеев в графическом режиме по этим шинам передается гигантский объем информации. Для подключения внешних накопителей и ряда других устройств используются интерфейсы SCSI или ATA, которые организованы по принципу шины. Таким образом, система шин ЭВМ является сложной многоярусной и шина, которая ранее считалась бы системной, оказывается лишь одной из нескольких. Вероятно поэтому в последнее время шину, через которую подключаются контроллеры периферийных устройств, не требующих особо высокой производительности обмена информацией, называют шиной ввода/вывода или шиной расширения. Примерами таких шин являются шины Q-bus, ISA, VME, EISA, MCA, PCI и целый ряд других.

1.3.2. Краткая характеристика шины ISA

Шина ISA (Industrial Standart Architecture) была разработана для персональных ЭВМ типа IBM PC AT и первоначально использовалась в

качестве системной шины и сохранилась в современных ПЭВМ для подключения контроллеров или плат расширения широкой номенклатуры. Важным классом ЭВМ, использующих эту шину, является класс промышленных или индустриальных ПЭВМ, имеющих в своем составе разнообразные и часто нестандартные платы сопряжения с объектом управления. Существует также шина PC-104, логически эквивалентная шине ISA, применяемая в управляющих микро-ЭВМ [5]. Фактически шина ISA стала стандартом для производителей персональной и индустриальной вычислительной техники, хотя в литературе отмечается, что этот факт не закреплен в официальных документах и это приводит к существованию различий в конкретных образцах продукции разных фирм. Здесь, однако, будем называть общедоступные сведения об этой шине [3…5] стандартом.

Шина ISA имеет раздельные линии (шины) адреса и данных. Разрядность шины данных равна 16, но обмен может осуществляться и 8-разрядными порциями (байтами). Адрес имеет разрядность до 24 при обращении к памяти. Для адресации устройств ввода/вывода может использоваться 16 адресных линий, но практически используется только десять. В стандарте ISA ведущее устройство называется задатчиком, а ведомое — исполнителем. На магистрали присутствуют следующие устройства, способные быть задатчиками:

- центральный процессор,
- контроллер прямого доступа к памяти (ПДП),
- контроллер регенерации памяти,
- периферийная плата.

Первые три устройства расположены на системной или материнской плате (для ПЭВМ обычного исполнения). Кроме них на материнской плате имеется ряд устройств, которые не могут быть задатчиками, но взаимодействуют с шиной (часы реального времени, таймер-счетчик, «кросс», основная память, контроллер прерываний, перестановщик байтов, ряд устройств ввода-вывода). В литературе по шине ISA все перечисленные устройства называются ресурсами.

Магистраль может работать в четырёх основных режимах, называемых циклами:

- доступ к ресурсу (обращение центрального процессора или периферийной платы на шине, к другому ресурсу на шине),
- прямой доступ к памяти — ПДП (обмен данными между памятью и устройством на шине, получившим это право, под управлением контроллера ПДП, который является задатчиком на шине),
- регенерация (регенерация динамической памяти ЭВМ под управлением контроллера регенерации),
- захват шины (выполняется периферийной платой для того, чтобы стать задатчиком на шине).

Основные режимы имеют по несколько реализаций в зависимости от типа устройств, обменивающихся информацией (см. п. 1.3.4).

1.3.3. Сигналы шины ISA

Рассмотрим состав шины по группам сигналов с самыми краткими пояснениями. Для получения более подробной информации, необходимой, например, при выполнении курсового проекта, следует обратиться к литературе [3…5].

В магистрали ISA используется положительная логика. Символ «-» (минус) перед обозначением сигнала указывает на то, что данный сигнал имеет активное (в терминах описания шины ISA «разрешенное») состояние при низком потенциале на соответствующей линии. Циклы на шине ISA выполняются по-разному в зависимости от типа ресурса, который распознается задатчиком по ответным сигналам от исполнителя.

Адресные сигналы

Это две группы линий. SA0...SA19 — фиксируемые адресные разряды (действительны в течение всего цикла обмена) и LA17...LA23 — нефиксируемые адресные разряды (действительны только в начале цикла обмена).

При обращении к памяти SA0...SA19 это 20 младших разряда адреса. При обращении к УВВ действительны только SA0...SA15, но большинство реальных плат работают только с SA0...SA9. Многие из адресов адресного пространства УВВ закреплены за конкретными устройствами ЭВМ. Резервными являются адреса 360h...36Fh. В адресном пространстве памяти также имеется ряд областей для специальных целей [6].

LA17...LA23 использовались для адресации добавочной памяти, подключенной через слоты ISA. В моделях последних лет платы памяти подключаются иным способом (см. рис.1.1), и данные адресные линии практически не используются.

К этой же группе относятся еще три сигнала.

-SBHE (System Bus High Enable) - разрешение байта, определяет тип цикла (8- или 16-разрядный).

BALE (Bus Address Latch Enable) — сигнал стробирования адресных разрядов. Использовался для фиксации LA17...LA23. Может использоваться в УВВ, но применяется редко.

AEN (Address Enable) — разрешение адреса. В режиме ПДП сообщает всем устройствам, что выполняется цикл ПДП, т.е. на адресной шине установлен адрес памяти, поэтому периферийные устройства не должны на этот адрес реагировать.

Шина данных

SD0...SD15 — разряды данных.

Командные сигналы

Сюда входят стробы чтения/записи обоих адресных пространств, а также ряд ответных сигналов, реализующих принцип квитирования.

-SMEMR, -MEMR (Memory Read) -SMEMW, -MEMW (Memory Write) представляют собой стробы чтения/записи данных при обращении к памяти.

-IOR (-I/O Read) ,-IOW (-I/O Write) — то же при обращении к УВВ.

-MEM CS16 (Memory Cycle Select) — выбор цикла для памяти. Сигнал выставляется памятью для извещения задатчика о том, что она имеет 16-разрядную организацию.

-I/O CS16 (-I/O Cycle Select) — выбор цикла для УВВ. Ответный сигнал УВВ, извещающий задатчика о том, что оно является 16-разрядным.

I/O CH RDY (I/O Channel Ready) — готовность канала ввода-вывода. Сигнал снимается исполнителем, если он не успевает выполнить требуемую операцию в темпе задатчика и требуется удлинение цикла.

-0WS (0 Wait States) — 0 тактов ожидания. Формируется исполнителем при необходимости проведения цикла обмена без тактов ожидания. Единственный сигнал шины ISA, синхронизированный с тактами SYCLK.

-REFRESH — регенерация. Сигнал выставляется контроллером регенерации для информирования всех устройств на шине о выполнении цикла регенерации динамической памяти, что повторяется с периодом около 15 мкс.

Центральные сигналы управления

-MASTER — формируется устройством, желающим стать задатчиком на шине.

RESET DRV (Reset of Driver) — сброс устройства. Сигнал начальной установки всех устройств на шине.

-I/O CHCK (I/O Channel Check) — проверка канала ввода-вывода. Сигнал ошибки.

CLK или SYSCLK (System Clock) — системный такт. Меандр со скважностью 2. Во многих ЭМВ его частота равна 8 МГц независимо от тактовой частоты процессора.

OSC — не синхронизированный с SYSCLK сигнал кварцевого генератора с частотой 14,31818 МГц со скважностью 2.

Запросы на прерывание

IRQ0… IRQ 15 (Interrupt Request) — запрос прерывания. Не все из них выведены на слоты.

Сигналы режима ПДП (DMA — Direct Memory Access)

DRQ0…DRQ7 (DMA Request) — запрос ПДП.

-DACK0…-DACK7 (DMA Acknolege) — подтверждение ПДП.

Не все DRQ/DACK выведены на слоты.

T/C (Terminal Count) — окончание счета циклов передачи данных в режиме ПДП.

Сигнгал AEN, формируемый контроллером ПДП в этом режиме, обычно относят к группе адресных сигналов.

На шине кроме того имеются линии, по которым подается питание в устройства ЭВМ: GND – линия нулевого потенциала («земля»), +5В, -5В, +12В, -12В.

1.3.4. Временные диаграммы циклов шины ISA

В рассматриваемой шине конкретный вид и параметры временных диаграмм зависят не только от типа цикла из указанных в п.1.3.2, но и от типа (память или УВВ), и разрядности ресурса доступа или передаваемой информации (8 или 16 бит), и от быстродействия ресурса, с которым ведется обмен. На рис. 1.3 приведены временные диаграммы выполнения циклов чтения и записи регистров ввода/вывода (портов). Об этом свидетельствуют командные сигналы –IOR и –IOW. По существу это диаграммы процессов, происходящих на шине при выполнении команд `in` и `out` процессором. Рис. 1.3 охватывает случаи обмена байтами (8 бит) и словами (16 бит), нормальный и удлиненный циклы обмена.

Если выполняется нормальный цикл доступа (ресурс не требует формирования тактов ожидания), то сигнал IO CH RDY в течение всего цикла будет сохранять высокий уровень (H). В этом случае говорят, что сигнал «разрешен».

Если выполняется 8-разрядный обмен, то сигнал -IO CS16 в течение всего цикла имеет уровень H («запрещен»). Если ресурс доступа (УВВ) формирует ответный сигнал -IO CS16, то выполняется 16-разрядный цикл обмена, который имеет другие временные параметры.

Таким образом, если сигналы IO CH RDY и -IO CS16 не участвуют в работе, то рис. 1.3 становится похожим на диаграммы синхронного обмена рис. 1.2, и временные параметры t1…t5 имеют тот же смысл. Отметим только некоторые моменты. На рис. 1.3 показана тактовая частота CLK. Этот сигнал фактически присутствует на шине, но сигналы шины с ним не синхронизированы (за исключением 0WS). На рисунке показан также сигнал BALE, который формируется задатчиком одновременно с установкой нового адреса на шине. Этот сигнал используется редко, но при наблюдении процесса обмена он удобен тем, что отмечает начало каждого цикла на шине.

В цикле записи на рис. 1.3 показано опережение на время t16 установки данных относительно строба записи. В [6] отмечается, что в некоторых компьютерах данные при записи устанавливаются после начала формирования строба. Важно то, что данные должны быть действительны на положительном спаде сигнала IOW (окончание интервала t3).

В табл. 1.1 приведены временные параметры диаграмм [7]. Следует обратить внимание, что в разных режимах работы шины один и тот же параметр принимает разные значения.

Если устройство, к которому выполняется обращение, является 16-разрядным, то оно должно не позже, чем через время t7 ответить установкой в низкий уровень (L) сигнала -IO CS 16. Тогда временные параметры цикла будут совсем другими (табл.1.1). При отсутствии ответного сигнала -IO CS 16 задатчик формирует временную диаграмму для 8-разрядного цикла обмена. Это пример реализации принципа квитирования.

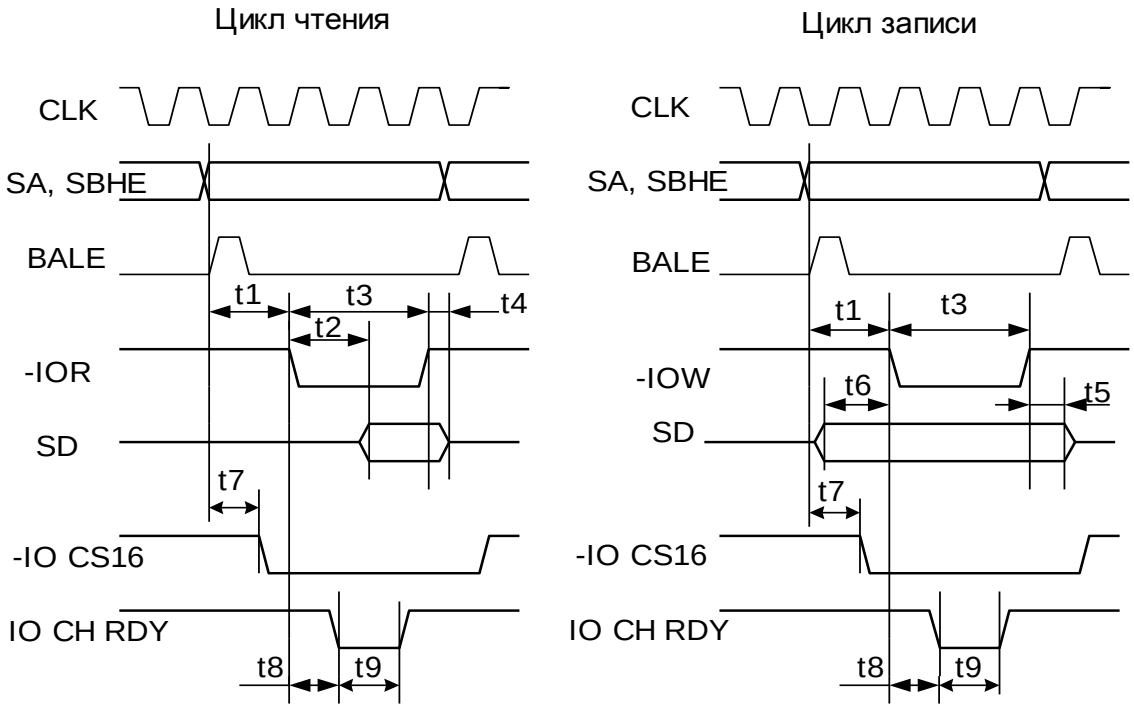

Рис. 1.3. Временные диаграммы циклов доступа к ресурсу шины ISA

Если устройство-исполнитель не в состоянии поддерживать нормальный темп обмена, то для успешного завершения цикла оно должно не позже, чем через интервал t8 после установки командного сигнала «запретить», т.е. установить уровень L на соответствующей линии, ответный сигнал IO CH RDY. При этом задатчик формирует такты ожидания, и состояние линий шины сохраняется в течение всего промежутка времени, пока IO CH RDY запрещен. Длительность этого промежутка (t9) кратна

периоду тактовой частоты CLK и не должна превышать 15600 нс. В противном случае формируется запрос на немаскируемое прерывание процессора NMI. Такой цикл называется удлиненным.

Величина t9, указанная в табл. 1.1, является максимальной и определяется, по-видимому, допустимым значением периода регенерации. Более значительная приостановка работы шины влечет за собой потерю данных в микросхемах динамической памяти. В некоторых моделях ПЭВМ в качестве допустимых для длительности запрещенного IO CH RDY указываются более короткие промежутки времени. Удлиненный цикл под управлением ответного сигнала IO CH RDY является другим примером реализации принципа квитирования в шине расширения ISA.

Таблица 1.1

Некоторые временные параметры циклов обмена для шины ISA с тактовой частотой 8 МГц

Обозначение на рис. 1.3	Содержательный смысл параметра	Для 8-разрядного обмена время, нс		Для 16-разрядного обмена время, нс	
		задатчик	ресурс	задатчик	ресурс
t1	Опережение установки адреса относительно команды	≥102	≥91	≥102	≥91
t2	Время установления данных после сигнала чтения	≤ 489	≤ 467	≤ 132	≤ 110
t3	Длительность команды чтения/записи	≥ 530	≥ 519	≥ 187	≥ 176
t4	Перевод линий сигналов SD в третье состояние	≤ 32	≤ 32	≤ 32	≤ 32
t5	Удержание данных при записи	≥ 30	≥ 30	≥ 30	≥ 30
t6	Опережение установки данных относительно команды записи	≥ 33	≥ 22	≥ 33	≥ 22
t7	Задержка -IO CS 16 относительно установки адреса SA	—	—	≤ 126	≤ 90
t8	Задержка IO CH RDY относительно команды	≤ 378	≤ 350	≤ 66	≤ 44
t9	Длительность запрещенного состояния сигнала IO CH RDY	до 15600 нс			

Временные диаграммы циклов обращения к памяти на шине выглядят качественно так же, как диаграммы, показанные на рис. 1.3, только в качестве командных сигналов используются стробы чтения/записи памяти (-MEMR,

-SMEMR, -MEMW, -SMEMW) и ответный сигнал -MEM CS 16. Значения временных параметров при обращении к памяти также будут другими. Следует напомнить, что обращение к памяти со стороны процессора производится по внутренней шине процессор-память. Платы расширения памяти через слоты ISA уже давно не подключаются. Обращение к памяти по шине ISA возможно со стороны ПУ в режимах прямого доступа к памяти (ПДП) или захвата шины. В этих случаях также возможен асинхронный обмен с управлением длительностью цикла ответным сигналом IO CH RDY, который может запрещаться платой памяти.

Режимы ПДП и захвата шины представляют собой способы высокоскоростного обмена информацией между периферийным устройством и основной памятью под управлением контроллера ПДП (собственно ПДП), или под управлением аналогичного контроллера в ПУ (в режиме захвата шины). При этом обмен происходит автономно от центрального процессора.

1.4. Внешние аппаратные прерывания

1.4.1. Особенности организации прерываний

Под прерыванием понимается временное прекращение выполнения текущей программы и переключение процессора на выполнение другой программы под действием запроса на это прерывание и с последующим возвратом к исполнению прерванной программы. Во многих литературных источниках принята следующая классификация причин возникновения запросов на прерывания:

- предопределенные события в процессах, например, по команде в программе вызываются программные прерывания (в списках команд многих процессоров имеются соответствующие команды). Такие прерывания используются для обращения к часто применяемым процедурам, которые отлажены и поставляются производителями ЭВМ,
- непредопределенные события в процессах, возникающие в результате вычислений, например, переполнение или другие недопустимые ситуации,
- события, возникающие в периферийных устройствах, внешние прерывания.

Иногда в этот перечень причин включают прерывания в результате действий оператора и прерывания от таймера, но мы такие прерывания будем относить к внешним. Здесь нас, в первую очередь, будет интересовать этот третий тип прерываний.

В IBM PC-совместимых ПЭВМ внешние прерывания, которые называются также аппаратными, делятся на маскируемые и немаскируемые.

Запросы на немаскируемые прерывания вырабатываются системной платой при возникновении ошибки четности при обращении к памяти (сигнал –I/O CHCK шины ISA), при зависании в канале (длительное отсутствие разрешения сигнала I/O CH RDY) и в ряде других случаев [5]. В ПЭВМ, использующих другие шины, например PCI, также предусмотрены возможности вызова прерываний этого типа.

Все виды прерываний объединяет общий порядок перехода к обработке запроса на прерывание, который сводится к следующим действиям:

- завершение работы, которую нельзя прервать без потерь данных,
- сохранение состояния процесса, т.е. информации, необходимой для возврата к выполнению прерванной программы,
- определение начального адреса программы-обработчика для поступившего запроса,
- выполнение программы обработки прерывания, в конце которой обычно записана команда возврата из прерывания (для процессоров ряда i80x86 … Pentium это команда **`iret`**),
- возврат в прерванную программу, т.е. выполнение команды `iret`.

Рис. 1.4 иллюстрирует эту последовательность действий процессора. Напомним, что регистр флагов (flags) содержит информацию о состоянии процессора и программы, в сегментном регистре кода (**`CS`**) хранится адрес сегмента с машинными командами, к которому имеет доступ процессор, а указатель команд (**`IP`**) хранит смещение относительно начала командного сегмента **следующей** подлежащей выполнению команды. Адрес команды определяется как **`CS:IP`.**

Состояние процесса сохраняется в стеке. Необходимая для продолжения работы исполняемой программы информация — это, прежде всего, адрес (точка) возврата и слово состояния программы. Пусть запрос IRQ поступил в момент выполнения i-й команды, а следующей должна была выполняться j-я команда (см. рис. 1.4). В **`CS:IP`** уже находится адрес j-й команды. Процессор выполнит до конца i-ю команду, при этом, возможно, изменится состояние регистра флагов. И если внешние прерывания не замаскированы, перед входом в прерывание содержимое регистров процессора, показанных на рис. 1.4 (для процессоров i80x86), будет переписано в стек. Если для продолжения правильной работы прерванной программы требуется сохранение содержимого других регистров, то это нужно предусмотреть в обработчике.

Определение начального адреса обработчика выполняется по-разному в ЭВМ различных архитектур. Известно два способа решения этой задачи: организация прерываний по радиальному и векторному принципам.

При **радиальной** организации адрес обработчика для каждого источника прерывания фиксирован аппаратно. Способ применяется в

системах с небольшим числом источников. Например, единственный внешний источник для микроконтроллера i8048 при прерывании вызывает переход по адресу 007. А в микропроцессоре К1801ВМ1 имеется три входа для запросов на радиальные прерывания.

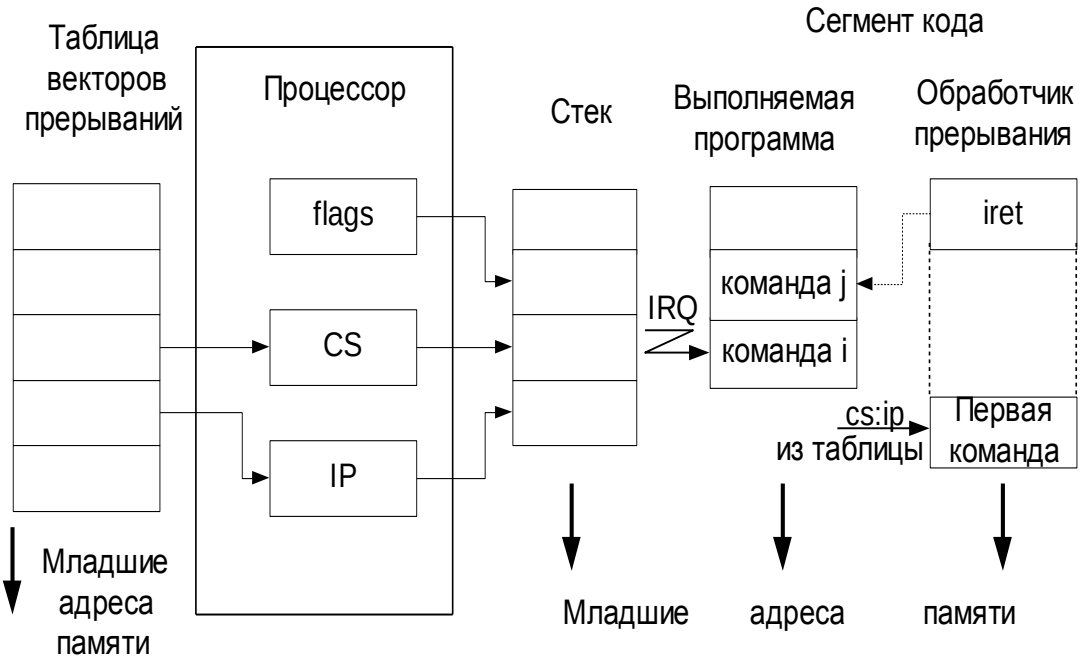

Рис. 1.4. Последовательность действий процессора при обработке запроса на прерывание: flags — регистр флагов; CS — сегментный регистр кода; IP — указатель команды

При **векторной** организации адрес обработчика прямо или в закодированном виде поступает в процессор из специального контроллера. Это происходит после прихода на вход векторного прерывания процессора сигнала запроса в результате выполнения процессором цикла чтения **адреса вектора прерывания** (АВП) или **кода прерывания** из контроллера прерывания. В ЭВМ на микропроцессорах i80x86…Pentium адреса обработчиков хранятся в таблице, которая находится в фиксированном месте в памяти. Эта таблица называется таблицей векторов или дескрипторов прерывания (IDT — Interrupt Descriptor Table). Каждому источнику назначен свой номер в таблице. Векторная организация применяется в системах с большим числом источников прерываний. Этот способ рассматривается в дальнейшем применительно к системам на базе микропроцессора i80x86.

На рис. 1.5 приведена упрощенная структурная схема подключения периферийных устройств, способных вызывать прерывания, к центральной части ЭВМ. С помощью этой схемы можно проследить взаимодействие центрального процессора ПРЦ (CPU — central processing unit) с другими функциональными узлами при входе в прерывание. Подобная схема

применяется в ЭВМ различных архитектур. На рис. 1.5 использованы наименования входов/выходов, принятые для соответствующих блоков IBM PC-совместимых ПЭВМ.

На схеме показаны периферийные устройства, подключенные к линиям запроса на прерывания IRQ (interrupt request) системной шины (ISA). Эти линии подключены ко входам IR программируемого контроллера прерываний ПКП (PIC — programmable interrupt controller). Контролер назван программируемым, так как обычно возможна какая-то программная его настройка.

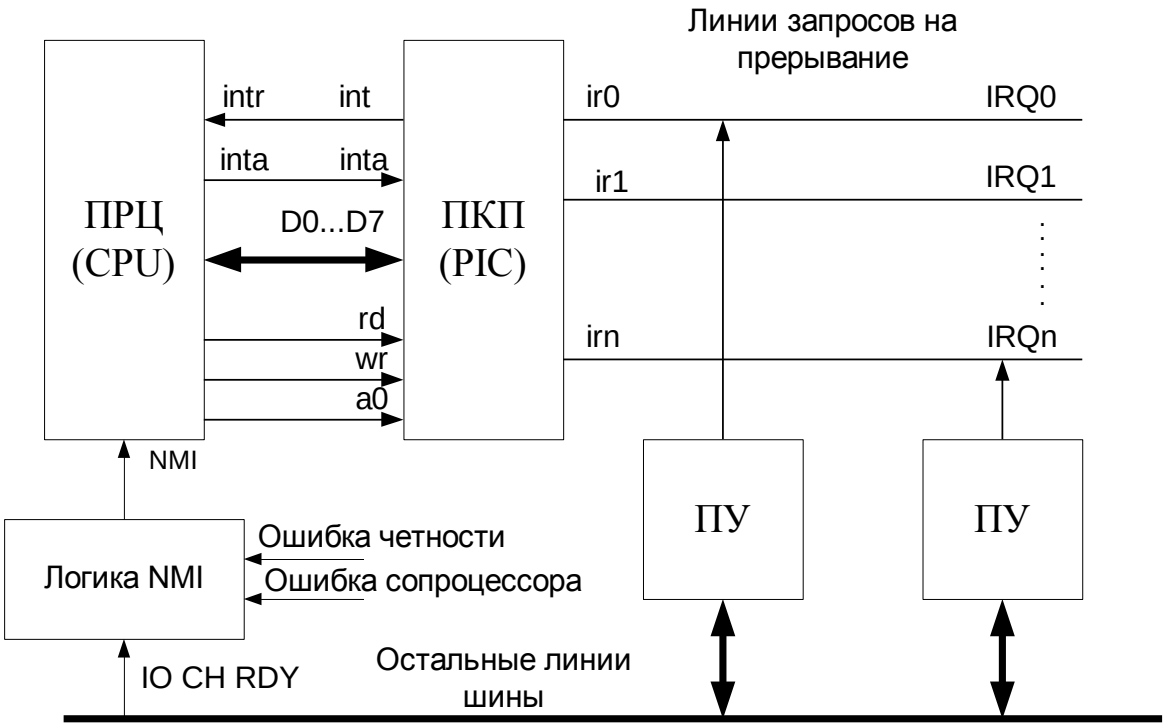

Рис. 1.5. Подключение периферийных устройств к системе обработки прерываний

Периферийное устройство выдает запрос на прерывание, т.е. устанавливает сигнал на шине IRQ и на соответствующем входе IR ПКП. Это приводит к активизации выхода INT контроллера прерываний и поступлению сигнала запроса на вход INTR процессора, который является входом сигнала запроса на векторное прерывание. Если флаг i разрешения внешних прерываний установлен (в регистре flags), процессор опрашивает вход INTR после завершения выполнения очередной инструкции.

Обнаружив сигнал на входе INTR, ПРЦ посылает подтверждение через выход INTA (Interrupt acknowledge). Приняв сигнал со входа INTA, ПКП

выставляет на шину данных D0…D7 номер или код прерывания, который считывается процессором.

Теперь процессор, получив необходимую информацию, завершает вход в прерывание:

- записывает в стек полный адрес следующей инструкции, содержащийся в регистрах **CS** и **IP**, а также текущее значение регистра флагов (см. рис. 1.4),
- обнуляет флаг i разрешения прерываний от внешних источников,
- считывает запись из таблицы векторов по номеру, полученному от источника прерывания, т.е. из ПКП (см. рис. 1.4, 1.5),
- выполняет переход по адресу, считанному из таблицы векторов прерываний (см. рис 1.4).

На этом вход в прерывание заканчивается. Результат — переход по адресу из таблицы векторов. По этому адресу должна находиться программа, которая выполняет обслуживание прерывания, т.е. производит необходимые манипуляции с регистрами периферийного устройства в соответствии с причиной прерывания.

Затем должен произойти **возврат в прерванную программу** — обратный переход по адресу, который был сохранен в стеке, а также должны быть восстановлены флаги процессора. Эти действия выполняются инструкцией **iret**, которая считывает из стека три слова и записывает их значения в регистры **CS, IP** и регистр флагов (см. рис 1.4).

З а м е ч а н и я :

- Для разрешения и запрета внешних прерываний в процессорах i80x86…Pentium предусмотрены команды установки и сброса флага i — команды **sti** и **cli.** Процессоры других типов имеют аналогичные команды.
- При переходе процессора к обработке прерываний флаг i автоматически сбрасывается, чем запрещаются прерывания от любых внешних источников. Если для решения задачи требуется разрешить обработку более приоритетных запросов, то это должен сделать программист.
- В обычных ПЭВМ (архитектурная линия IBM PC) микросхема ПКП расположена внутри системной (материнской) платы. Шина, по которой осуществляется обмен данными между ПРЦ и ПКП (D0…D7 на рис. 1.5) находится на системной плате. А к слотам, через которые подключены ПУ, подведены только линии запросов IRQ. Так что можно сказать, что для самих ПУ прерывания носят радиальный характер в том смысле, что каждое ПУ подключено к своей индивидуальной линии IRQ, и для входа в прерывание ПУ только выставляет запрос и ничего более. А все существенные действия выполняет ПКП, расположенный в непосредственной близости от ПРЦ. Такая организация внешних прерываний налагает существенные ограничения при

попытке подключить дополнительные ПУ, которые должны бы вызывать прерывания, так как свободных линий IRQ может оказаться недостаточно, а подключение нескольких ПУ к одной линии вызывает трудности.

В других архитектурах, например, использующих шины Q-bus или VME-bus, к одной линии запроса на прерывание могут быть подключены несколько источников (по схеме монтажного ИЛИ). В этом случае контроллер каждого такого ПУ должен выполнять некоторые функции ПКП, в частности, при получении сигнала подтверждения ПУ должно выдать на шину данных (через слоты системной шины) код, идентифицирующий источник запроса, чтобы центральная часть ЭВМ могла правильно выбрать программу-обработчик. Например, по шине Q-bus передается адрес вектора прерываний, который используется непосредственно для обращения к таблице векторов. При этом вектор прерывания содержит не только адрес обработчика, но и слово состояния процессора, которое помещается в соответствующий регистр (аналог регистра flags).

- На рис. 1.5 показана цепь сигнала NMI, который формируется одним из чипсетов при поступлении на его входы сигнала, требующего немедленной реакции. Для многих ПЭВМ ранних моделей событиями, вызывавшими появление этих сигналов, являлись следующие ситуации:

— ошибка канала ввода/вывода (I/O CH RDY),

— ошибка сопроцессора,

— ошибка четности при обращении к памяти или к другому устройству, которое может сформировать сигнал -I/O CHCK.

В моделях последних лет единственным источником немаскируемого прерывания является ошибка четности (сопроцессоры в современных ПЭВМ не используются, а превышение длительности сигнала I/O CH RDY значения 15600 нс, указанного в табл. 1.1, немаскируемого прерывания не вызывает).

1.4.2. Контроллер прерываний ПЭВМ

Как показано на рис 1.5, контроллер прерываний (ПКП) обеспечивает развязку между внешними источниками запросов на прерывание и процессором. Роль и место ПКП в системе прерываний в целом описана в п. 1.4.1. Однако практическое программирование процедур обработки прерываний требует более детального знакомства с этим функциональным узлом. Контроллеры прерываний в ЭВМ разных архитектурных линий имеют определенное сходство. В настоящем пособии ознакомимся с ПКП персональных ЭВМ. Такие контроллеры уже более двадцати лет выпускаются в виде больших интегральных схем. В современных ПЭВМ ПКП размещены в чипсетах обрамления центрального процессора.

Прототипом современных ПКП является микросхема i8259A, которая имеет восемь входов запросов ir0…ir7 и использовалась в ПЭВМ класса IBM PC/XT с шиной XT-Bus, называемой также восьмиразрядной шиной ISA. ПКП i8259A имеет сложную внутреннюю структуру, может программно настраиваться на различные режимы работы, допускает каскадное включение нескольких ПКП с целью увеличения количества линий запросов IRQ. Эти свойства обеспечивают гибкость и эффективность системы прерываний. Среди микросхем отечественных серий К580 и К1810 имеются аналогичные ПКП.

В первых 16-разрядных ПЭВМ использовались две микросхемы ПКП, включенные каскадно, как показано на рис. 1.6. Это позволило довести до 15 число линий IRQ на шине. Можно заметить, что к ведущему контроллеру можно подключить до восьми ведомых, тогда число линий запросов увеличится до 64. Каждый ПКП кроме входов/выходов ir0…ir7, D0…D7, Int, Inta, назначение которых объяснено в п. 1.4.1, имеет следующие управляющие входы:

- RD, WR — входы сигналов чтения и записи. В контроллер записывается управляющая информация, из контроллера читается код, по которому будет определен вектор прерывания, а также информация о состоянии ПКП,
- CS — выбор микросхемы,
- A0 — вход нулевого разряда адреса, который используется при обращении к внутренним регистрам контроллера,
- SP/EN — признак подчиненности (низкий уровень напряжения на этом входе переводит контроллер в состояние ведомого).

Кроме этого контроллеры связаны линиями CAS [2…0], по которым должен передаваться номер подчиненного ПКП. В более поздних моделях ПЭВМ структура, показанная на рис. 1.6, реализована в СБИС с полным сохранением логики работы пары микросхем ПКП i8259A.

Функции контроллера:

- фиксация запросов от внешних источников (на рис. 1.6. указано обычное для ПЭВМ распределение линий IRQ),
- формирование запроса Int на вход Intr процессора (рис. 1.5, 1.6),
- формирование кода прерывания и выдачу его по шине D0…D7 после прихода сигнала Inta и под действием сигнала RD (рис. 1.6),
- маскирование запросов, приходящих на входы ir0…ir7,

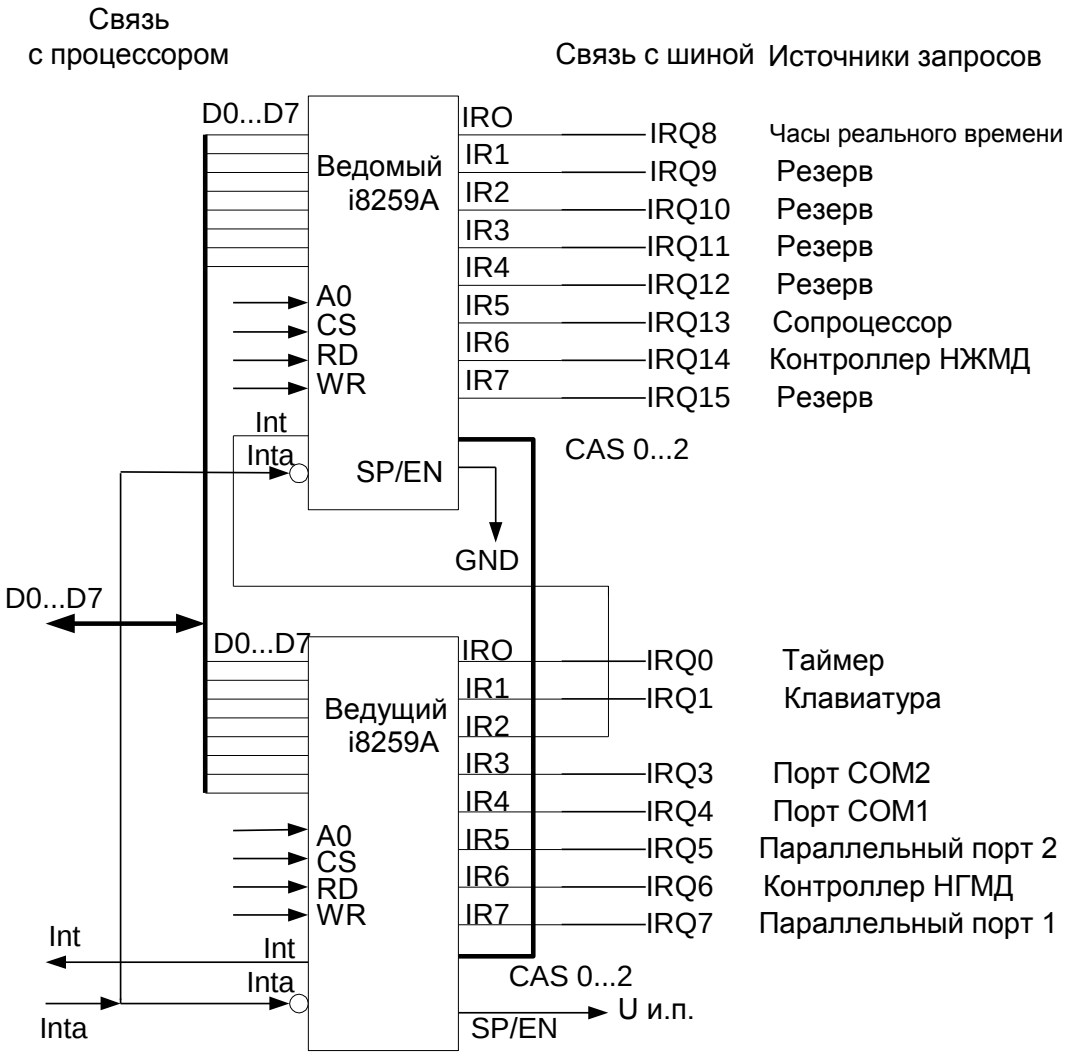

Рис. 1.6. Каскадное включение двух микросхем i8259A

- управление очередностью обработки запросов на прерывание в соответствии с назначенными приоритетами, т.е. выполнение функций арбитража.

Для реализации этих функций контроллер содержит функциональные узлы, часть из которых показана на рис. 1.7:

- регистр запросов на прерывания (IRR – Interrupt Request Register) для фиксации поступающих запросов,
- регистр маскирования прерываний (IMR — Interrupt Mask Register) позволяет запретить действие одного или нескольких запросов,
- арбитр приоритетов (PR — Priority Resolver) — схема, выявляющая приоритеты запросов и выбирающая запрос с наивысшим приоритетом,

- регистр обслуживаемых прерываний (ISR — Interrupt Service Register) для хранения уровней запросов на прерывание, которые обрабатываются в данный момент.

Перечисленные регистры (IRR, IMR, ISR) являются восьмиразрядными по числу входов запросов микросхемы.

Для обращения к регистрам ПКП в адресном пространстве ввода/вывода ПЭВМ выделено две пары соседних адресов (портов). Для ведущего ПКП это адреса 020h и 021h, а для ведомого — адреса 0A0h и 0A1h.

ПКП может находиться в двух состояниях или режимах работы — инициализация и обслуживание запросов.

Инициализация. В этом состоянии ПКП находится после сброса. При этом он воспринимает команды инициализации, которые обозначаются ICW1…ICW4 (Initialization Command Words). Каждое из ICW имеет разрядность 8 бит и записывается в ПКП байтовыми командами вывода (out) по указанным адресам. ICW1 записывается по адресу 020h (0A0h), а ICW2…ICW4 — по адресу 021h (0A1h). Последовательность ICW устанавливает конфигурацию системы прерываний и некоторые особенности работы ПКП.

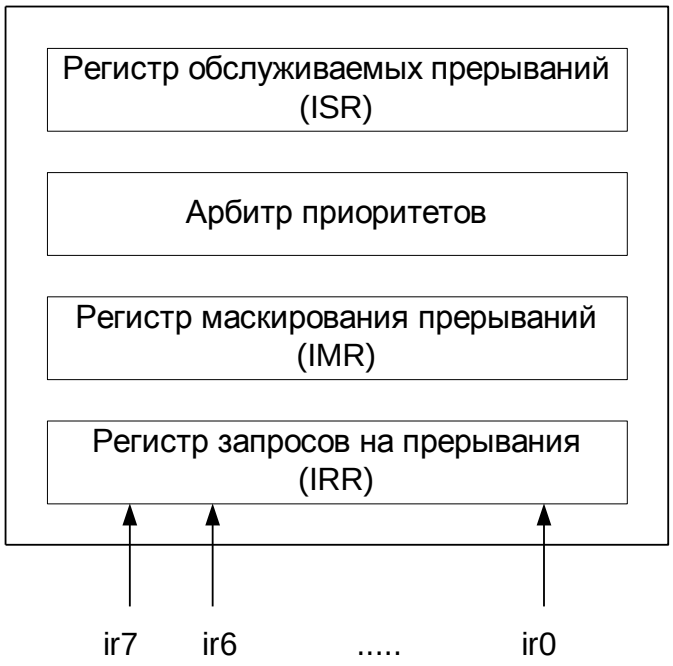

Рис.1.7. Основные функциональные узлы ПКП i8259A

В частности, выполняются следующие настройки:

- чувствительность ПКП к характеру сигналов по входам ir0…ir7 — возможна настройка на работу по фронту или по уровню сигнала запроса,
- работа с единственным (для PC/XT) или с двумя ПКП (согласно рис.1.6, для PC/AT и более новых моделей),

- положение в каскаде (ведущий или ведомый),
- начальный номер (код) прерывания из набора последовательных кодов, генерируемых данным контроллером (команда ICW2). В стандартной конфигурации ПЭВМ в реальном режиме работы для ведущего контроллера устанавливается начальный код 08h, а для ведомого — 70h.

При инициализации выполняется также ряд других настроек [3,6].

После выполнения инициализации ПКП переходит в **режим обслуживания прерываний или операционный режим.** В этом режиме ПКП воспринимает команды управления OCW1…OCW3 (Operation Control Words). Эти команды также являются 8-разрядными и посылаются в ПКП байтовыми командами вывода.

OCW1 — это байт маски запросов, посылается в IMR командой записи по адресу 021h (для ведомого 0a1h). Единица в разряде OCW1 означает запрет реакции ПКП на сигнал по соответствующему входу ir.

OCW2 — это команда управления приоритетами и способом завершения обслуживания прерывания, посылается по адресу 020h (0a0h).

OCW3 — это команда управления контроллером, посылается также по адресу 020h (0a0h).

Командами OCW2, OCW3 можно перевести ПКП в различные режимы обслуживания запросов и завершения обслуживания.

Предусмотрены следующие режимы обслуживания запросов:

- режим вложенных прерываний. Входам ir0…ir7 присвоены фиксированные приоритеты, которые убывают с ростом номера входа. Ir0 имеет высший приоритет,
- режим циклической обработки запросов. После обработки прерывания соответствующему запросу присваивается низший приоритет, остальные приоритеты циклически сдвигаются,
- режим адресуемых приоритетов. Командой можно присваивать запросу высший приоритет, приоритеты остальных запросов циклически сдвигаются,
- режим опроса. В этом режиме ПКП не формирует сигнала Int. Процессор «узнает» о наличии запросов путем чтения IRR,
- режим специальной маски. Позволяет отменить приоритетное упорядочение обработки запросов и обрабатывать их по мере поступления (в ПЭВМ не применяется).

Командами OCW2, OCW3 можно также установить автоматическое или неавтоматическое завершение прерывания. При автоматическом завершении соответствующая заявка в ISR сбрасывается при поступлении в ПКП сигнала Inta. Использование этого режима завершения влечет ряд трудностей при программировании [8].

Неавтоматическое завершение прерывания требует, чтобы в программе обработки были выполнены команды сброса соответствующего бита в регистре ISR. Возможно специфицированное и обычное завершение

прерывания. В первом случае выполняется команда сброса конкретного бита в регистре ISR. При обычном завершении выполняется сброс бита с максимальным приоритетом в ISR. В обоих случаях требуется посылка OCW2. Так как обрабатываемое прерывание является наиболее приоритетным, то достаточно обычного завершения, которое выполняется путем записи кода 020h по адресу 020h (0a0h). Этот простой способ сброса заявки в ISR и рекомендуется для использования при программировании процедур обработки прерывания.

Командой OCW2 можно выбрать один из режимов управления приоритетами. В ПЭВМ ряда IBM PC/XT/AT в результате работы программ начальной инициализации после включения питания ПКП устанавливается в режим вложенных прерываний с фиксированными приоритетами. В отдельном ПКП приоритеты заявок возрастают от ir7 к ir0. Из рис. 1.6 видно, что высшим приоритетом обладает запрос по шине IRQ0, а низшим — запрос IRQ7. Приоритеты запросов, поступающих на ведомый ПКП, находятся между приоритетами IRQ1 и IRQ3.

З а м е ч а н и е. Еще раз отметим, что переход к обработке внешнего прерывания производится процессором при сброшенном флаге i, поэтому разрешение обработки заявок с более высокими приоритетами возлагается на программиста.

Относительно исходного (после начальной инициализации) режима завершения прерываний имеются противоречивые сведения. В большинстве источников, например [8], указывается, что стандартным является режим неавтоматического завершения, т.е. сброс ISR возлагается на программиста. В практике составителей настоящего пособия также не встречался иной вариант.

При наличии нескольких заявок от разных источников выполняется та заявка, у которой приоритет выше. Заявки от отдельных входов IR можно замаскировать. Функции маскирования и выбора наиболее приоритетной заявки обеспечиваются тремя байтовыми регистрами ПКП и арбитром приоритетов (см. рис. 1.7).

Каждый из входов IRx в контроллере подключен к разряду *x* регистра IRR через входной вентиль. Вентиль открыт, если в регистре маски IMR сброшен разряд *x*. Регистр маски подключен к порту по адресу 021h (0a1h). Таким образом, для разрешения запросов по входу IR*x* нужно, чтобы разряд *x* порта 021h (0a1h) был сброшен.

Рассмотрим возможное прохождение сигнала прерывания от некоторого внешнего источника, подключенного к контроллеру прерываний с помощью линий IRQ.

Пусть поступление запросов по линии IRQ*x* и по соответствующему входу IR*x* ПКП разрешено. Если в регистре запросов IRR разряд *x* сброшен, то по фронту сигнала IRQ*x* в разряд *x* регистра IRR запишется единица.

Запрос запоминается, и арбитр приоритетов по значению регистра ISR принимает решение о возможности обслуживания запроса.

Заявка *x* из регистра IRR может быть принята к обслуживанию, если в регистре ISR не зафиксированы заявки от входов с номером $y \le x$. Контроллер выставляет сигнал INT. Заявка принимается к обслуживанию, когда будет получен ответный сигнал INTA. Контроллер обнуляет бит *x* в IRR и устанавливает бит *x* в ISR, затем выставляет на шину данных номер прерывания, соответствующий этой заявке.

После того, как заявка *x* принята к обслуживанию, установленный в ISR бит *x* блокирует обслуживание заявок с номерами $y \ge x$. Это означает, в частности, что блокированы следующие заявки от входа *x*. Для снятия маскирования необходимо, чтобы программа обслуживания прерывания в конце работы сбросила бит *x* в ISR. Сброс выполняется записью в порт 020h байта со значением 03*x*h (команда специфицированного завершения прерывания, OCW2), где *x* = 0…7 — номер входа IR контроллера прерываний. Альтернативный вариант сброса — запись в порт 020h (0a0h) байта со значением 020h (команда обычного завершения прерывания, OCW2). В результате в ISR сбрасывается заявка с наименьшим номером: это именно та заявка, которая обслуживается в текущий момент как наиболее приоритетная.

З а м е ч а н и е. Все пояснения, сделанные на сс. 29…31 относительно управления контроллером прерываний в режиме обслуживания, в равной мере относятся как к ведущему ПКП, так и к ведомому. Т.е. для маскирования прерываний от запросов, поступающих на входы ведомого ПКП (IRQ8…IRQ15), нужно устанавливать соответствующие разряды в IMR ведомого ПКП по адресу 0a1h.

Для обычного сброса заявки в ISR, если запрос поступил в ведомый ПКП, нужно записать код 020h как по адресу 0a0h (для сброса ISR ведомого ПКП), так и по адресу 020h (для сброса ISR ведущего ПКП). Это необходимо потому, что запросы IRQ8…IRQ15 после обработки в ведомом ПКП транслируются на вход IR2 ведущего контроллера. Таким образом для сброса заявки в ведомом ПКП нужно в обработчике прерывания записать следующую последовательность команд:

```
mov al, 020
out 020,al
out 0a0, al ,
```

а для сброса заявки в ведущем ПКП — только две первые их них.

1.5. Проектирование плат расширения

В современной вычислительной технике преобладают устройства с так называемой магистрально-модульной организацией, когда основные

функциональные узлы ЭВМ стандартным образом подключены к магистрали или шине, называемой системной. Здесь речь идет об устройствах и системах, конструктивно исполненных в виде единого блока. Совокупность ограничений, обеспечивающих логическую, электрическую (или вообще физическую), конструктивную совместимость составляет существо понятия межмодульного интерфейса. В сложных современных вычислительных системах, в том числе в ПЭВМ, наблюдается иерархическая организация шин, используются различные интерфейсы, и понятие системной шины оказывается размытым, но на одном «этаже» такой системы шин сохраняются все свойства классической магистрально-модульной структуры.

В практике создания управляющих вычислительных комплексов (УВК) с широким использованием стандартных вычислительных средств часто возникает необходимость разработки нестандартных периферийных плат, позволяющих добиться желательных свойств комплекса. В частности, это могут быть платы сопряжения УВК с объектом управления. Такую специализированную плату можно представить состоящей из двух частей, как это показано на рис. 1.8 :

- интерфейсная часть, обеспечивающая сопряжение с шиной данного стандарта,
- специальная или функциональная часть, обеспечивающая информационный обмен с собственно периферийным устройством (ПУ).

Эти устройства называются контроллерами ПУ. В качестве примера можно привести контроллеры стандартных ПУ (накопителей на магнитных дисках, лентах) или в управляющих ЭВМ контроллеры электроприводов, электроавтоматики и т.п. На рис. 1.8 показано, что со стороны центральной части ЭВС контроллер подключен к шинам адреса и данных (в ряде интерфейсов они являются совмещенными), к линиям управляющих сигналов шины. На рис. 1.8 показаны УС — командные и управляющие сигналы шины, а также ОС — ответные сигналы контроллера, корректирующие темп обмена в соответствии с протоколом. Так как обмен информацией со многими ПУ может происходить в режиме прерывания или ПДП, то на схеме показано подключение контроллера к линиям запросов (IRQ, DRQ), а также к линии подтверждения прямого доступа (DACK). В некоторых стандартах требуется подтверждение и прерывания (на схеме не показано). Для устранения конфликтов с другими устройствами, присутствующими на шине, подключение платы к линиям IRQ и DRQ часто делается через перемычки или микровыключатели. Использование режимов прерывания и ПДП для обмена с одним контроллером встречается редко, но примеры имеются (контроллеры накопителей на магнитных дисках).

Интерфейсная часть контроллера ПУ должна формировать сигналы управления элементами специальной части. Это, как правило, сигналы чтения и записи регистров (стробы). Ответные сигналы от специальной части, отражающие состояние объекта управления (ПУ), принимаются

интерфейсной частью по командам чтения по внутренней шине данных, поэтому специально на схеме не показаны.

В специализированных УВК получили распространение несколько межмодульных параллельных системных интерфейсов, например, Q-bus, ISA,

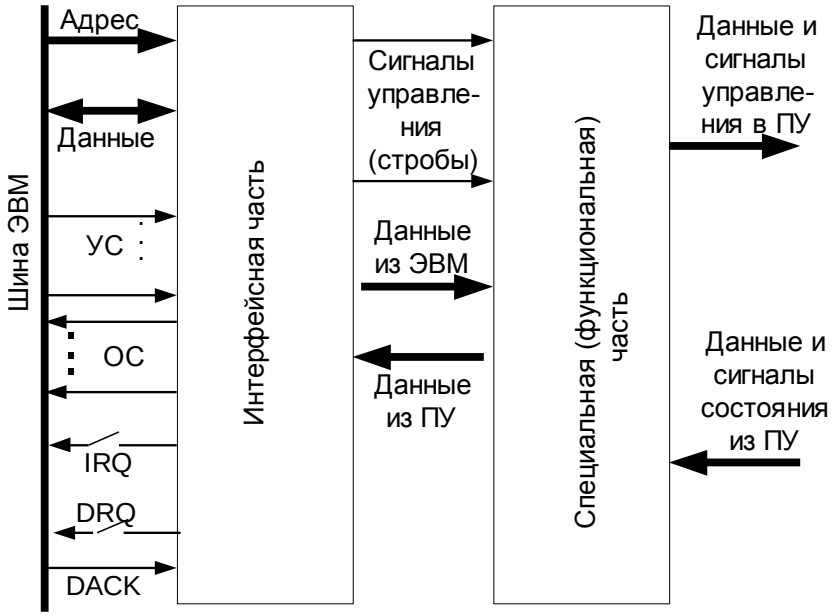

Рис. 1.8. Обобщенная структурная схема платы сопряжения (контроллера ПУ)

PCI и ряд других. Несмотря на значительные различия между этими шинами, организация обмена информацией по ним имеет общие принципы, сформулированные в п.1.2

Функциональная схема интерфейсной части контроллера ПУ определяется используемым интерфейсом и для данной шины является унифицированной. Если контроллер является ведомым устройством, то его интерфейсная часть должна распознать факт обращения именно к данному устройству, под управлением сигналов от ведущего устройства принять или выдать информацию на шину данных, сопроводив эти действия ответными сигналами в случае необходимости. Если контроллер является устройством ведущим, то он формирует сигналы на захват магистрали (режимы прямого доступа к памяти или захвата шины) и после получения доступа к ней формирует сигналы управления обменом.

Таким образом, для разработки интерфейсной части нестандартной платы ЭВМ необходимо изучить ограничения, налагаемые используемым интерфейсом. В документации на интерфейс описаны все электрические линии, управляющие сигналы, их функции в различных режимах работы шины, причинно-следственные отношения между сигналами, их временные и

электрические характеристики, конкретные типы соединителей (разъемов), конструктивные требования к платам и т.п.

В качестве примера рассмотрим обобщенную функциональную схему платы расширения для шины ISA. Типовая функциональная схема интерфейсной части такой платы приведена на рис. 1.9. Ограничимся случаем, когда такая плата не может быть задатчиком на шине. Временные диаграммы взаимодействия центрального процессора с контроллером ПУ приведены на рис. 1.3, временные параметры указаны в табл. 1.1.

В соответствии с принципом унификации (п. 1.2) при проектировании платы расширения необходимо обеспечить все виды совместимости. Конструктивная совместимость обеспечивается на этапе конструкторского проектирования путем соблюдения размеров и других требований, указанных в спецификации шины. Здесь уделим внимание только информационной и электрической совместимости.

Информационная совместимость обеспечивается выполнением требований протокола взаимодействия устройств на шине, правильным использованием линий интерфейса. Информационная и электрическая совместимость тесно связаны в части выполнения элементами схемы устройства требований к временным задержкам и последовательности срабатывания. Рассмотрим это подробнее на примере требований к интерфейсной части контроллера ПУ шины ISA (см. рис. 1.3 и 1.9).

Распознавание факта обращения именно к нашему контроллеру выявляет селектор адреса СА за время t_{CA}. В цикле чтения под действием выходного сигнала от СА и строба –IOR узел формирователей управляющих сигналов (ФУС) вырабатывает строб чтения выбранного регистра контроллера (которым, например, открываются ключи, подающие сигналы на линии «данные из ПУ»). Пусть ФУС вносит задержку t_α, а схема чтения выбранного регистра вносит задержку t_β. ФУС также под действием строба чтения –IOR шины за то же время t_α формирует внутренний сигнал, настраивающий шинный приемопередатчик на чтение ПУ (передача справа налево по рис. 1.9). Сам приемопередатчик вносит задержку t_γ. Тогда для правильной работы схемы необходимо $t_1 > t_{CA}$, а также $t_2 > t_\alpha + t_\beta + t_\gamma$. В этих рассуждениях не учитывалась возможная задержка, вносимая шинным приемником, который показан пунктиром на рис. 1.9.

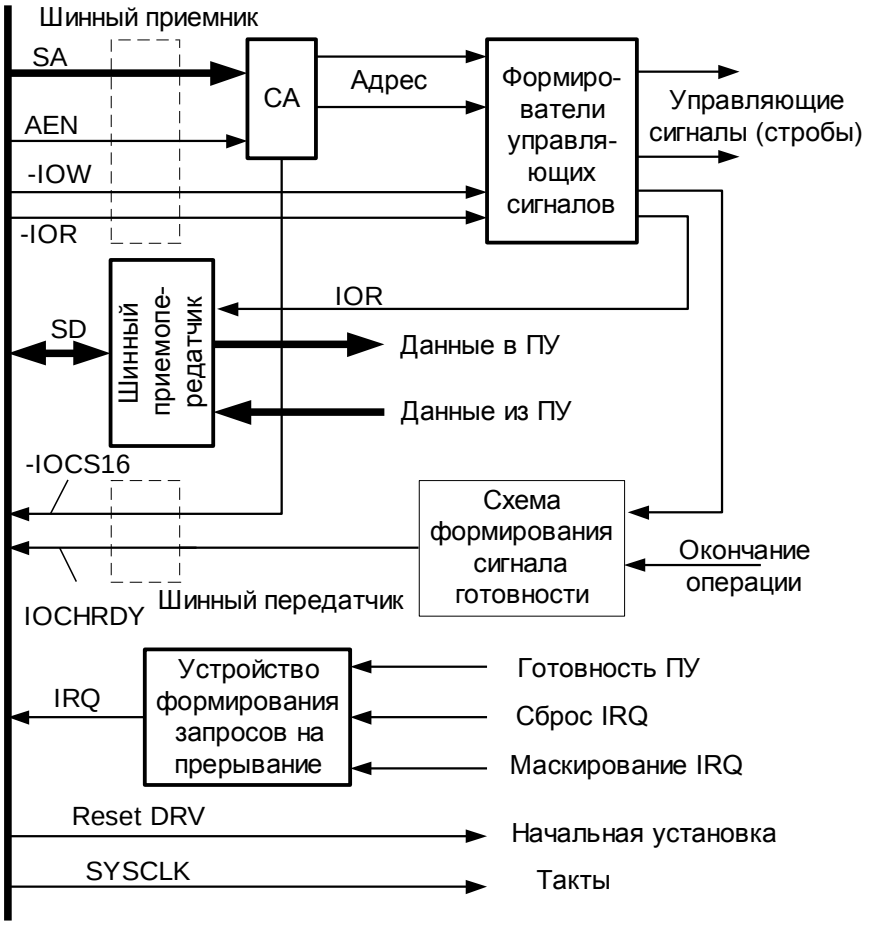

Рис. 1.9. Функциональная схема интерфейсной части контроллера ПУ

В цикле записи на прохождение сигналов данных из шины через приемопередатчик (в этом цикле он настроен на передачу слева направо) времени вполне достаточно (рис.1.3). Но для правильного завершения цикла и сохранения данных в выбранном регистре требуется, чтобы данные не исчезли раньше, чем закончится строб, выработанный узлом ФУС. Т.е. необходимо, чтобы выполнялось $t_\alpha < t_\gamma + t_5$.

Здесь t_1, t_2 и t_5 — требования стандарта шины, а t_{CA}, t_α, t_β и t_γ — динамические параметры использованных элементов схемы.

Аналогичные соотношения легко получить и для элементов, формирующих сигналы -IOCS16 и IOCHRDY.

На рис. 1.9 показано устройство формирования запросов на прерывание. В него приходят три сигнала. «Готовность ПУ» — это сигнал, означающий, что наступило событие, требующее переключения программы на обработку прерывания. Т.е. этот сигнал инициирует установку IRQ. Желательно проектировать схему так, чтобы запрос IRQ сбрасывался по команде во время обработки прерывания. «Сброс IRQ» может быть инициирован командой записи так, чтобы один из разрядов записываемых

данных обеспечивал это действие. Многие контроллеры ПУ допускают маскирование прерываний по программе. Это также может быть сделано командой записи в соответствующий регистр управления данного ПУ. Эту возможность отражает на рис 1.9 сигнал «маскирование IRQ».

Для начальной установки контроллера ПУ, когда это необходимо, используется сигнал Reset DRV. Тактовые импульсы SYSCLK также могут быть использованы для работы контроллера, хотя нужно отметить, что в простейших случаях (чтение и запись) в них нет необходимости, так как стробов шины ISA достаточно для управления этими процессами.

Понятие электрической совместимости включает в себя также соблюдение ограничений по нагрузочной способности элементов. На шине ISA могут быть использованы устройства (элементы), имеющие выходы типа ТТЛ (транзисторно-транзисторная логика), выходы типа ОК (открытый коллектор) и элементы с тремя состояниями выхода. Элементы с выходами ТТЛ или ОК являются либо только входными, либо только выходными для устройства. Элементы с тремя состояниями выхода могут быть входными, выходными и, кроме этого, находиться в третьем состоянии.

Например, для сигналов, перечисленных в п. 1.3, установлено, что формирователи сигналов адресных (SA, LA, SBHE), данных (SD), стробов чтения/записи памяти/УВВ, а также тактовых импульсов SYSCLK должны быть с тремя состояниями выхода. Командные сигналы BALE, RESET DRV, запросы на прерывание (IRQ), сигналы режима ПДП (DRQ, DACK, T/C, AEN), а также OSC должны формироваться элементами с выходами ТТЛ. Ответные сигналы -MEM CS 16, -I/O CS 16, -I/O CH RDY, -I/O CH CK, -0WS, а также сигналы центрального управления — REFRESH и -MASTER должны формироваться элементами с ОК.

При проектировании платы расширения требуется обеспечить, чтобы элементы, формирующие выходные сигналы платы, были способны обеспечить выходные токи низкого уровня 24 мА (втекающий ток при низком уровне выходного сигнала). Нагрузочная способность при высоком уровне выходного сигнала должна быть не хуже, чем 3 мА.

Потребление по входам элементов, подключенных к шине ISA, не должно превышать 0,8 мА (по линии с ОК не более 0,4 мА для каждого слота) при низком уровне сигнала передающего элемента. Потребление по входам при высоком активном уровне сигнала не должно превышать 0,04 мА (для линии с ОК 0,02 мА).

Приведенные электроэнергетические ограничения установлены достаточно давно и ориентированы на применение устаревшей элементной базы. В настоящее время на шине вряд ли можно встретить устройства со столь значительным потреблением тока. Современные микросхемы чаще всего выполнены по КМОП технологии. Это значит, что потребление по их входам не превышает нескольких микроампер, что в тысячи раз меньше допустимых значений. В то же время нагрузочная способность выходов

современных КМОП элементов превышает 10 мА. В этих пределах гарантируется их совместимость с ТТЛ элементами по уровням сигнала и достаточно хорошие динамические параметры. Следовательно, в реальном проектировании, используя современные элементы, практически можно, по крайней мере, вдвое сократить требования к нагрузочной способности используемых элементов по выходам. Необходимость включать в состав платы показанные на рис. 1.9 пунктиром приемопередатчики и приемники при этом также отсутствует.

При проектировании плат расширения традиционно используются интерфейсные микросхемы, среди которых имеются БИС, ориентированные на применение в конкретных шинах. Однако появление микросхем гибкой логики со средствами автоматизации проектирования делает доступным и целесообразным использование именно их при реализации интерфейсной части контроллеров. При этом повышается надёжность и гибкость разрабатываемых устройств, снижается время, необходимое на их разработку и отладку.

Кроме ограничений по нагрузке в статике необходимо выполнить ряд требований к размещению элементов на плате (длина проводника от контакта внешнего разъема до вывода микросхемы не должна превышать 65 мм), к емкости нагрузки (не более 20 пФ на контакт разъема) и к фронтам выходных сигналов передатчиков (не хуже 3 нс), соблюдение которых гарантирует допустимый уровень помех [7].

1.6. Параллельный порт и интерфейс Centronics

1.6.1. Основные положения. Разновидности параллельного порта

Параллельный порт был включен в состав ПЭВМ с 1981 г. для подключения принтеров, поэтому он называется LPT-портом (LPT – Line Printer, т.е. линейный принтер или принтер построчной печати, хотя к такому порту подключаются принтеры и других типов). Однако, благодаря удобству программирования и простоте подключения к нему, он широко применяется для сопряжения персональных ЭВМ с разнообразными периферийными устройствами (ПУ) и приборами. Отмечается, в частности, безопасность для ПЭВМ подключения к параллельному порту внешних устройств [6]. Удобство подключения обусловлено тем, что такой порт имеется у всех ПЭВМ, а соответствующий разъем установлен снаружи, и для подключения не нужно вскрывать корпус системного блока. Отсутствие шин питания на разъеме не вызывает существенных затруднений. В качестве недостатков интерфейса принято указывать невысокое быстродействие и ограничения на протяженность линий связи (до 2 м). Заметим, что скорость передачи данных через параллельный порт все же выше, чем через последовательный, а в

последние годы появилось несколько усовершенствований параллельного порта, которые позволяют поднять скорость обмена информацией. Причем обеспечивается двунаправленная передача данных (режимы ECP и EPP), а также увеличена максимально допустимая длина кабеля [5].

Подключение к параллельному порту осуществляется через 25-контактный разъем на корпусе ПЭВМ. Перечень сигналов, обеспечивающих интерфейс, приведен в табл. 1.2. Так как порт первоначально предназначался только для подключения принтеров, сигналы в таблице имеют соответствующие обозначения.

Знак «-» (минус) в обозначением сигнала означает, что для данного сигнала активным является низкий уровень напряжения на соответствующей линии.

Таблица 1.2

Сигналы на выходном разъёме параллельного порта

Обозначение сигнала	I/O	Назначение сигнала/контакта
D7-D0	O	Шина данных для передачи информации в принтер
-ACK	I	Сигнал подтверждения принятия данных принтером и готовность принять следующие
BUSY	I	Сигнал занятости принтера. Компьютер может передавать данные только после снятия сигналов –ACK и BUSY
PE	I	Сигнал означающий, что в принтере кончилась бумага.
SLCT	I	Принтер выдает этот сигнал, если он выбран и готов к работе. У многих принтеров он всегда равен 1
-ERROR	I	Принтер выдает этот сигнал при внутренней ошибке, состоянии off-line или при отсутствии бумаги
-STROBE	O	Сигнал стробирования шины D7-D0. Данные действительны как по фронту, так и по спаду сигнала
AUTO FD	O	Получив этот сигнал, принтер переводит каретку на следующую строку
-SLCT IN	O	Сигнал принтеру о том, что он выбран и последует передача данных
-INIT	O	При подаче этого сигнала на принтер происходит его инициализация и очистка буфера печати (длительность его должна быть не менее 2,5 мкс)

Тип выходных каскадов для всех сигналов — ТТЛ, каждая линия может быть нагружена на один стандартный вход ТТЛ. Все сигналы программно доступны, что позволяет реализовать произвольную временную диаграмму на выходах параллельного порта (с ограничениями на скорость смены сигналов,

которые определяются временем выполнения команд ввода/вывода, т.е. команд **in** и **out**).

Из таблицы следует, что для управления объектом, подключенным к порту, располагаем следующими ресурсами:

- восьмиразрядная шина данных. Обычное назначение — передача данных в принтер,
- четырехразрядная шина управляющих сигналов из ЭВМ в ПУ,
- пятиразрядная шина чтения осведомительных сигналов от ПУ.

Таким образом, параллельный порт обеспечивает выдачу двенадцати управляющих сигналов на объект и прием пяти сигналов от объекта.

Описанный порт не обеспечивал приема байтов в параллельном коде. Кроме того, для управления периферийным устройством (принтером) требовалось программное формирование управляющих сигналов. Для устранения этих недостатков было предложено несколько модификаций параллельного порта, в которых предоставлялась возможность как выдачи, так и приема информации, повышалась скорость обмена за счет наделения контроллера порта функциями управления обменом (порты типа 2, типа 3, а также EPP и ECP и несколько менее распространенных типов [5]).

В 1994 г. принят стандарт IEEE 1284, в котором определяются режимы работы параллельных портов и сопряженной с ними аппаратуры, процедура согласования режимов порта и периферийного устройства (своего рода автоконфигурация), также характеристики интерфейса на физическом уровне.

Стандартом IEEE 1284 предусмотрены пять режимов работы порта: режим совместимости (с традиционным портом), полубайтовый обмен, двунаправленный обмен, EPP и ECP.

Режим совместимости (Compatibility Mode) — однонаправленный вывод по протоколу Centronics. Этот режим соответствует стандартному параллельному порту (SPP). Настройка порта на этот режим имеется в меню программы Setup большинства современных ПЭВМ. Стандартный порт представлен в адресном пространстве ввода/вывода тремя регистрами (регистр данных, регистр состояния и регистр управления). Сигналы, показанные в табл. 1.2 соответствуют разрядам этих регистров. Более подробно этот режим рассмотрен в п. 1.6.2.

Полубайтовый обмен (Nibble Mode) — ввод байта за два цикла (по 4 бита), используя для ввода линии состояния (входы по табл. 1.2). Этот режим обмена может использоваться на любых адаптерах.

Двунаправленный байтовый обмен (Byte Mode) — ввод/вывод байта целиком, используя для приема линии данных (D0…D7 по табл.2.1). Для приема информации программно отключаются выходные буферы данных. Этот режим работает только на портах, допускающих чтение выходных данных (Bi-Directional или PS/2 Type 1).

EPP (Enhanced Parallel Port) Mode — двунаправленный обмен данными, при котором управляющие сигналы интерфейса генерируются самим

контроллером порта во время цикла обращения к порту (чтения или записи в порт). Периферийное устройство, поддерживающее обмен через EPP, может работать со скоростью УВВ, подключенного к шине ISA. Протокол обеспечивает автоматическую подстройку темпа обмена в зависимости от длины кабеля. Кроме трех регистров, аналогичных SPP, контроллер EPP имеет в адресном пространстве ввода/вывода еще два регистра, обращение к которым и инициирует обмен в режиме EPP. Режим эффективен при работе с устройствами внешней памяти, адаптерами локальных сетей.

ECP (Extended Capability Port) Mode — двунаправленный обмен с дополнительными возможностями (аппаратное сжатие данных, буферизация данных для прямого и обратного каналов, использование режимов прямого доступа к памяти и программного ввода/вывода). Управляющие сигналы интерфейса генерируются аппаратно. Порт ECP имеет несколько режимов работы и расширенный по сравнению с EPP набор регистров. Эффективен для принтеров и сканеров.

В современных ПЭВМ с LPT-портом на системной плате режим порта (SPP, EPP, ECP или их комбинация задается) в BIOS Setup. Режим Compatibility Mode полностью соответствует SPP и часто установлен по умолчанию. Все остальные режимы расширяют функциональные возможности интерфейса и повышают его производительность.

З а м е ч а н и е. Обычный или стандартный параллельный порт (SPP) предназначен для передачи данных из ПЭВМ к периферийному устройству и является, как отмечалось, однонаправленным. Однако схемотехника выходов его контроллера такова, что при попытке чтения регистра данных порта принимаются либо ранее записанные данные, либо то, что подано на линии D0…D7. Если предварительно записать в регистр данных код 0ffh, то через LPT-порт можно прочитать код, поданный извне. На этом свойстве основана работа большого количества различных приборов, подключаемых к этому порту. Например, широко применяются программаторы для микросхем памяти или однокристальных микроконтроллеров, подключаемые к LPT-порту, где прием данных нужен для верификации. Однако, как отмечается в [5], для многих современных ПЭВМ такое использование SPP невозможно.

1.6.2. Стандартный параллельный порт и интерфейс Centronics

Для программиста стандартный параллельный порт представлен тремя адресами или портами ввода/вывода. В ПЭВМ может быть до четырех параллельных портов, которые имеют обозначение LPT1…LPT4. Чаще всего имеется один LPT1 (базовый адрес 0378h или 03bch), реже он соседствует с портом LPT2 (базовый адрес 0278h или 0378h). Через регистр с базовым адресом (base) осуществляется передача данных — 8-разрядного байта. Через

регистр с адресом (base + 1) принимаются осведомительные сигналы от объекта. Назовем этот регистр регистром состояния. Через регистр с адресом (base + 2) выдаются сигналы управления. Назначение разрядов регистров приводится ниже в табл. 1.3. Значком «X» отмечены неиспользуемые разряды.

Для установки в активное состояние сигнала, имеющего знак «-» перед своим обозначением необходимо записать единицу в соответствующий разряд регистра управления. Исключением является разряд 2 регистра управления (сигнал -INIT): запись единицы в этот разряд приводит к установлению потенциала +5В на соответствующем выходе.

Таблица 1.3

Назначение разрядов регистров состояния и управления стандартного параллельного порта LPT

Номера разрядов	7	6	5	4	3	2	1	0
Регистр состояния (base+1)	BUSY	-ACK	PE	SLCT	-ERROR	X	X	X
Регистр управления (base+2)	X	X	X	Разрешение прерывания	-SLCTIN	-INIT	-AutoFD	-Strobe

Временная диаграмма обмена данными между ПЭВМ и периферийным устройством формируется программно в соответствии с требованиями, которые предъявляются со стороны периферийного устройства. На рис. 1.10 приводится типовая временная диаграмма цикла передачи данных для этого случая [3,4].

Для успешной передачи данных должны быть обеспечены следующие временные соотношения: $t1 \geq 500$ нс, $t2 \geq 500$ нс, $t3 \geq 500$ нс, $t4 \geq 2500$ нс . Первые три временных интервала формируются программно и оказываются в норме без специальных мер, так как установка сигналов на линиях интерфейса выполняется командой **`out dx, al`**, которая содержат в себе цикл 8-разрядной записи в УВВ, который, в свою очередь, длится не менее 500 нс (см. рис. 1.3 и табл. 1.1). Интервал t4 формируется внутренними схемами принтера.

Согласно рис. 1.10 в этом режиме работы порта необходимо программно реализовать следующие фазы передачи байта данных:

Записать данные в регистр данных (базовый адрес порта).

Прочитать регистр состояния и убедиться, что принтер не занят (проанализировать сигналы BUSY и/или -ACK).

Если принтер не занят, установить сигнал -STROBE путем записи соответствующего кода в регистр управления порта.

Записать в регистр управления код, снимающий сигнал -STROBE

.

Порядок выполнения действий по пп. 1 и 2 можно поменять.

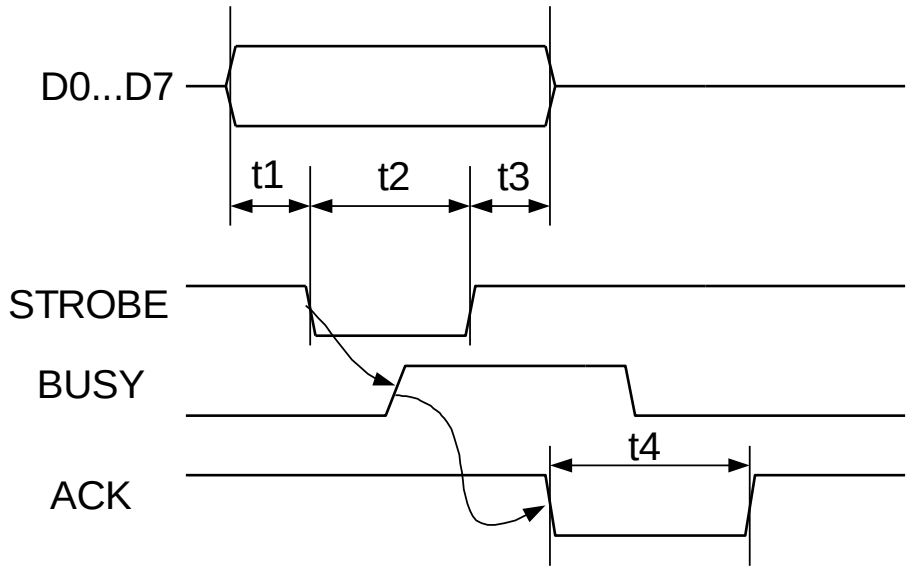

Рис. 1.10. Временная диаграмма передачи данных интерфейса Centronics

Международный стандарт «де-факто» Centronics определяет состав сигналов, их электрические параметры, тип соединителя и распределение сигналов по его контактам, а также протокол взаимодействия источника и приемника информации, описанные в этом параграфе. Отечественным аналогом этого интерфейса является интерфейс радиальный параллельный модифицированный ИРПР-М, который допускает применение соединителей других типов и использует наряду с описанными другие обозначения для сигналов [5].

1.7. Последовательные интерфейсы

1.7.1. Общие положения

Последовательные интерфейсы в вычислительной технике применяются для подключения к ЭВМ периферийных устройств различного назначения, а также для организации передачи информации между ЭВМ, в том числе для подключения к вычислительным сетям. Имеется также ряд служебных интерфейсов последовательного типа, например, JTAG, служащий для тестирования компонентов вычислительных устройств, и др.

При рассмотрении техники передачи информации по последовательным каналам связи, в том числе последовательных интерфейсов и портов, используются представления и терминология техники вычислительных сетей. Напомним некоторые из них.

Под вычислительной сетью, в частности, под локальной вычислительной сетью (ЛВС или LAN — Local Area Network) понимается совокупность технических средств и программного обеспечения, которая позволяет организовать совместное использование ресурсов (аппаратных и программных) вычислительных машин, объединенных в эту сеть. В качестве примера общего ресурса можно привести базу данных, к которой имеют доступ множество сотрудников учреждения со своих рабочих мест, или общий принтер, на котором они же могут распечатывать свои документы.

Передача информации в вычислительных сетях представляет собой сложную многоэтапную процедуру со сложными взаимодействиями между программными и аппаратными средствами, участвующими в этой работе. Международной организацией по стандартам (ISO) рекомендована так называемая эталонная модель взаимодействия открытых систем OSI (Open System Interconnection), в которой определено содержание этапов передачи информации между ЭВМ. Модель OSI предусматривает следующие семь уровней коммуникационных протоколов: физический, канальный, сетевой, транспортный, сеансовый, представительный и пользовательский. Вопросы, касающиеся работы интерфейсов периферийных устройств ЭВМ, относятся, главным образом, к двум нижним уровням этой модели: к физическому и канальному.

Физический уровень является чисто аппаратным. Он определяет физические и электрические параметры взаимодействующих устройств (параметры электрических сигналов, характеристики физических линий связи, буферных усилителей, конструкцию соединителей). Канальный уровень (он называется еще уровнем звена данных [3] и уровнем линии передачи [12]) определяет способы представления передаваемой информации электрическими сигналами (кодирование) и формирует структуру посылки. На этом же уровне осуществляется контроль и коррекция ошибок, а также

выполняется ряд других функций. Остальные уровни в данном пособии не рассматриваются.

С техникой сетей и применением последовательных каналов связи связаны некоторые специфические понятия, из которых упомянем топологию и среду передачи информации.

Топология — это метод соединения абонентов вычислительной сети в пространственно-логическая структуру. Характерными для ЛВС примерами являются следующие разновидности топологии: радиальная (звезда), шина, кольцо, полный граф, дерево и др.

Среда передачи информации — это физическая реализация канала связи. В технике последовательных каналов и ЛВС имеют применение следующие разновидности среды: электрический кабель (витая пара, коаксиальный кабель), волоконно-оптический кабель, радиоканал, инфракрасный канал, электрическая сеть.

1.7.2. Интерфейс RS-232C и последовательный асинхронный приемопередатчик

Последовательный интерфейс RS-232C является наиболее широко распространенным интерфейсом между ЭВМ и периферийным оборудованием различного назначения. Этот интерфейс используется также для связи между ЭВМ. В персональных ЭВМ стандартной конфигурации интерфейс RS-232C используется совместно с последовательными портами. Пользуясь представлениями эталонной модели OSI, можно сказать, что интерфейс RS-232C является реализацией физического уровня соответствующего комплекса.

Назначение интерфейса RS-232C — связь при двухточечном соединении устройств в полудуплексном и дуплексном режиме. В [3] отмечается возможность многоточечного подключения устройств к этому интерфейсу, однако, такая топология (шина) при использовании именно RS-232C практически не встречается. Интерфейс предназначен для передачи информации на небольшие расстояния (до 15 м) и с небольшими скоростями (от 50 до 19200 бит/с). Данные могут передаваться в синхронном или асинхронном режимах. Средой передачи информации является витая пара.

Стандарт RS-232C определяет параметры электрических сигналов в линии связи, допустимые нагрузки, временные соотношения, состав и функциональное назначение сигналов интерфейса, типы соединителей (разъемов), распределение сигналов по контактам этих соединителей, а также ряд других характеристик, которые попадают под понятие физического уровня.

В стандарте RS-232C рассматриваются два класса устройств, которые называются оконечным оборудованием данных (ООД) и аппаратурой

передачи данных (АПД). В наиболее полном виде связь между двумя пунктами ООД выглядит следующим образом.

ООД→(RS-232C)→АПД→(линия связи)→АПД→(RS-232C)→ООД

Типичным примером ООД является ПЭВМ, а примером АПД является модем, т.е. основное назначение интерфейса — передача данных между ООД и АПД.

Стандарт рекомендует, но не обязывает использовать для подключения кабеля RS-232C 25-контактный разъем (DB25P). На IBM/PC-совместимых ПЭВМ установлен либо такой разъем, либо 9-контактный (DB9P). Сигналы, выведенные на эти разъемы приведены в табл. 1.4.

Из таблицы видно, что кроме обязательных для любых интерфейсов линий данных TD и RD (их часто обозначают TxD и RxD), а также линий

Таблица 1.4

Сигналы на выходном разъёме последовательного порта персональной ЭВМ (интерфейс RS232C)

Обозначение сигнала	I/O	Назначение сигнала/контакта
CG	-	Защитная земля (Chassis Ground)
TD	O	Передаваемые данные (Transmitt Data)
RD	I	Принимаемые данные (Receive Data)
RTS	O	Запрос передачи (Request To Send)
CTS	I	Готов к передаче, т.е. готовность АПД к передаче данных в линию связи (Clear To Send)
DSR	I	Готовность АПД к работе (Data Set Ready)
SG	-	Сигнальная земля (Signal Ground)
DCD	I	Обнаружение несущей частоты в линии связи (Data Carrier Detect)
DTR	O	Готовность ООД (Data Terminal Ready)
RI	I	Индикатор вызова (Ring Indicator)

заземления и нулевого потенциала CG и SG имеется группа сигналов, с помощью которых осуществляется управление темпом приема/передачи данных (“hand-shaking”).

В табл. 1.4 указано 10 сигнальных линий. В 9-контактном разъеме отсутствует линия CG. Остальные контакты 25-контактного разъема используются для подключения сигналов, которые обеспечивают сложные

режимы работы редко применяемых модемов. Различные источники указывают разное назначение не включенных в таблицу контактов.

Все десять сигналов из таблицы используются только при соединении ЭВМ с модемом. Для подключения многих устройств достаточно так называемого нуль-модемного кабеля. Пример четырехпроводного кабеля приведен на рис. 1.11. Кабель соединяет непосредственно два ООД, например, ЭВМ и периферийное устройство или две ЭВМ.

При отсутствии вывода CG на разъеме кабель становится трехпроводным. Простейшим вариантом является двухпроводный кабель для односторонней передачи данных.

Каждый из сигналов интерфейса имеет два уровня, которые измеряются относительно линии SG. Со стороны источника сигнала высокий уровень должен находиться в диапазоне от +5 В до +15 В, низкий уровень должен быть в пределах от –5 В до –15 В. Со стороны приемника сигнала высокий уровень может находиться в диапазоне от +3 В до +25 В, а низкий уровень —

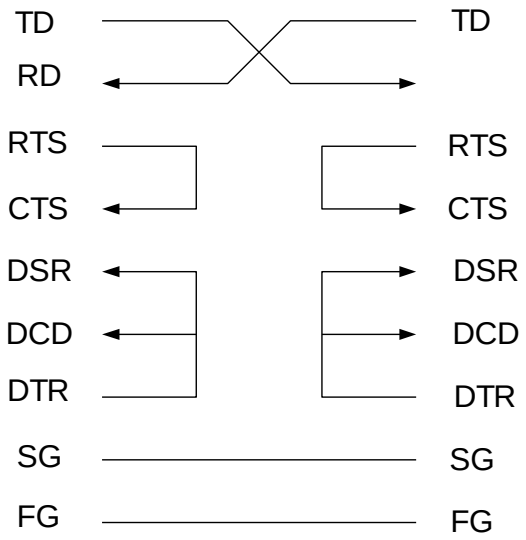

Рис. 1.11. Схема четырехпроводного кабеля для RS-232C

в диапазоне от –3 В до –25 В. При этом диапазон от –3 В до +3 В является зоной нечувствительности. Таким образом, стандарт обеспечивает хорошее соотношение сигнал/помеха.

Следует отметить «слабые» стороны данного интерфейса:

- не предполагается использование дифференциальных входов (выходов) у приемников (передатчиков), т.е. линия связи электрически несимметрична, что делает ее уязвимой для внешних помех,
- не обеспечивается гальваническая развязка устройств, связанных этим каналом, что также снижает уровень помехозащищенности.

Организация работы интерфейса на канальном уровне сводится к следующему.

Для управляющих сигналов активному их состоянию («включено» или ON) соответствует низкий уровень потенциала на линии. Пассивному состоянию («выключено» или OFF) соответствует высокий уровень. Для линий передачи/приема данных логической единице соответствует низкий уровень («отмеченное» состояние или MARK), логическому нулю соответствует высокий уровень (состояние SPACE). Данные передаются и принимаются последовательно по битам, т.е. в каждый момент времени по передачи/приема передается не более одного бита. При передаче к битам собственно данных добавляются синхронизирующий стартовый бит, контрольный бит, позволяющий выявить ошибку нечетной кратности в принятых данных, и стоповые биты. Контрольный бит не обязателен.

Передача одной посылки (назовем ее байтом) по последовательному каналу иллюстрируется рис. 1.12. В начале передачи выдается синхронизирующий стартовый бит — линия переводится в состояние положительного потенциала на время t_S. В результате, за счет переключения линии из исходного состояния запускается приемник на другом конце линии.

Затем передаются информационные биты, начиная с младшего бита байта данных. Количество информационных бит может быть меньше 8 (5,6 и 7), что определяется настройкой передатчика. В этом случае старшие разряды байта из регистра данных передатчика не передаются. Затем может следовать контрольный бит, если его формирование было задано при настройке передатчика. Значение контрольного бита определяется предшествующей настройкой (контроль на четность или нечетность) и количеством единичных информационных бит в текущей передаче.

Затем линия возвращается в состояние низкого потенциала, и после паузы в один или два периода t_S передача считается законченной. Конечную паузу рассматривают обычно как передачу стоповых бит — соответственно одного или двух. Пауза нужна, в основном, для компенсации расхождения в скоростях приема и передачи.

Успешная связь предполагает, что приемник и передатчик настроены одинаково — по скорости, по числу информационных бит, по типу контроля (четность, нечетность, отсутствие контроля). Согласование по количеству стоповых бит не является столь же существенным. Встречаются ПЭВМ, оснащенные приемопередатчиками, которые посылают в линию «лишний» стоповый бит. Этот факт иллюстрирует важное свойство применяемого способа кодирования, который называется кодированием «без возврата к нулю» (Non Return to Zero —NRZ).

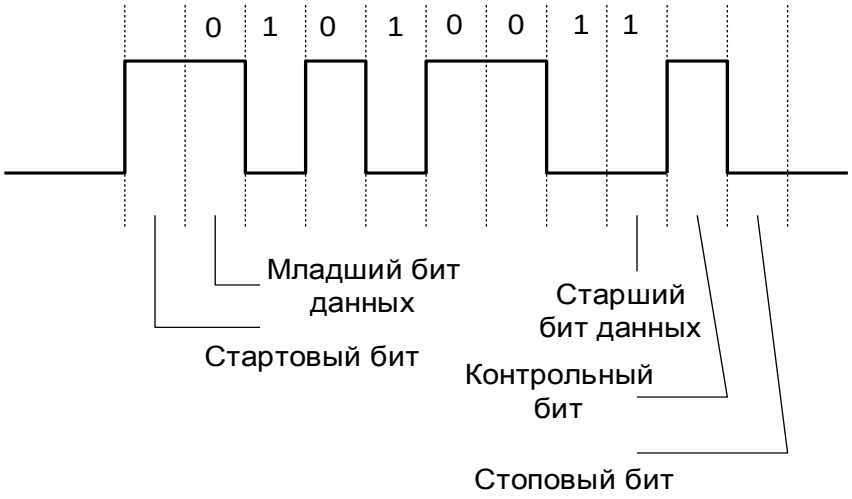

Рис. 1.12. Временная диаграмма посылки байта 11001010 (8 информационных бит с контролем дополнением до четности)

Такой код хорош своей простотой, однако, обладает существенным недостатком: отсутствует самосинхронизация. Если представить себе передачу длинной последовательности бит (см. рис. 1.12) и предположить, что частоты тактовых импульсов в передатчике и приемнике «разъехались», то становится понятно, что со временем приемник пропустит или ошибочно примет повторно очередной бит информации.

Для решения проблемы синхронизации в данном интерфейсе применены следующие меры:

- ограничена длительности посылки. Информационные биты обрамляются стартовым и стоповыми битами. По фронту стартового бита происходит сброс схем синхронизации. Стоповый бит отделяет посылку от стартового бита следующей посылки,
- стандартизованы частоты работы приемо-передающих схем.

Для реализации связи по этому интерфейсу в состав ЭВМ входит асинхронный приемопередатчик (АСПП или UART — Universal Asynchronous Receiver/Transmitter). В ПЭВМ прошлых лет выпусков это были специализированные микросхемы, например, i8250 в первых ПЭВМ класса IBM PC/XT, микросхема 16550A в моделях начала 90-х гг. Подробное описание работы и программирования этих микросхем имеется в литературе. В современных ПЭВМ приемопередатчик входит в состав многофункциональных СБИС (частности, в состав так называемого

«южного моста»), при этом обеспечивается программная совместимость с микросхемой 16550A.

Заметим, что в настоящее время различными производителями предлагается множество микросхем АСПП, предназначенных, главным образом, для использования в специализированных вычислительных устройствах. Современные микросхемы также поддерживают описанный протокол и при этом обладают дополнительными возможностями. Например, микросхемы 16550D (ряд модификаций) фирмы National Semiconductor способны вести передачу/прием со скоростью до 1,5 Мбод.

Логическую организацию АСПП поясняет рис. 1.13. Как передатчик, так и приемник имеют в своем составе буферные регистры данных, сдвиговые регистры для преобразования из параллельного кода в последовательный и обратно. Имеются также регистры состояния, информация в которых отражает текущее состояние данного функционального узла. Показана связь единственного передатчика одного ООД с единственным приемником другого ООД, в то время как в реальных микросхемах АСПП присутствует и то и другое. Регистры данных и состояния доступны в адресном пространстве ввода/вывода. Буферный регистр данных передатчика доступен для записи. Буферный регистр данных приемника и регистр состояния доступны для чтения. На рис. 1.13 не показаны схемы управления и синхронизации, а также интерфейс с центральной частью ООД. Но нужно иметь в виду, что устройство имеет еще ряд программно доступных регистров, через которые устанавливается режим работы.

До начала работы производится инициализация АСПП путем настройки его на выбранный режим работы. Устанавливается частота работы АСПП, формат посылки (число информационных бит, число стоповых бит, вид контроля), а также, при необходимости, производятся настройки, связанные с обеспечением работы по прерыванию. Примеры таких настроек приведены в п. 2.2 настоящего пособия. В ПЭВМ инициализация АСПП выполняются при загрузке операционной системы после включения напряжения питания.

Элементарный акт передачи/приема сводится к тому, что при готовности передатчика, которая отражается в регистре состояния, или по запросу IRQ от передатчика в буферный регистр передатчика записывается очередной байт передаваемого массива. Далее процессом передачи/приема управляют микросхемы АСПП. В темпе частоты работы АСПП (в очередном такте) байт данных переписывается в сдвиговый регистр передатчика. При этом буферный регистр оказывается свободным и готовым к приему следующего байта, что отражается в регистре состояния. Из сдвигового регистра информация по одному биту поступает в линию связи с той же самой установленной частотой.

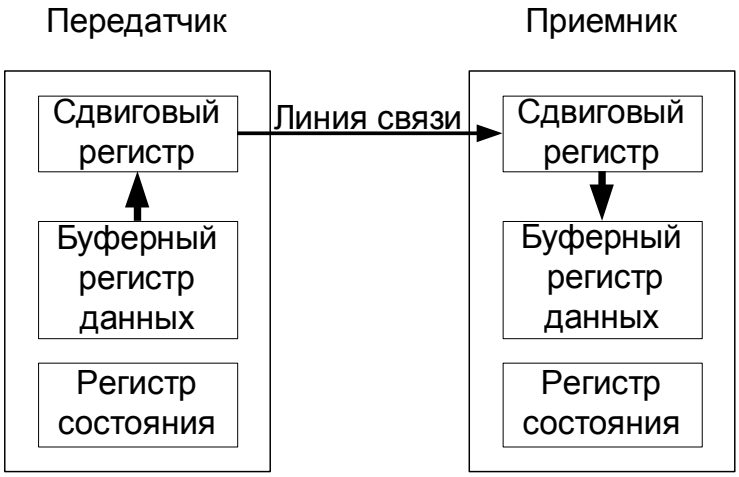

Рис. 1.13. Логическая организация последовательного асинхронного приемопередатчика

На приемном конце линии в момент прихода положительного фронта стартового бита (см. рис. 1.12) происходит установка времязадающих элементов, а затем — прием информации в сдвиговый регистр приемника. Когда байт принят, он в очередном такте будет переписан в буферный регистр приемника, что отразится в регистре состояния на приемной стороне путем установки разряда готовности приемника. Завершение приема байта может сопровождаться формированием запроса на прерывание.

В этом описании отсутствуют указания на меры, которые принимаются для обеспечения целостности данных. Так, например, передатчик оказывается в состоянии готовности, как только его буферный регистр освободится, и передача может быть продолжена. При этом нет уверенности, что на приемной стороне данные приняты и обработаны. Для исключения потерь данных стандарт RS-232C предполагает использование одного из двух протоколов управления потоком передачи:

- аппаратный протокол (RTS/CTS) использует сигналы RTS и CTS. На передающей стороне снятие сигнала CTS (перевод соответствующей линии в состояние OFF — высокий, т.е. положительный уровень) трактуется как приказ прекратить передачу,
- программный протокол (XON/XOFF) требует двунаправленной линии передачи данных. Приемная сторона в случае опасности потери данных посылает по обратной линии команду XOFF, что также прекращает передачу.

Более подробное описание АСПП и его программирование приведено в п. 2.8 второй части пособия.

В заключение данного параграфа отметим, что аналогом описанного интерфейса является отечественный интерфейс, получивший название «Стык С-2» [3].

1.7. Организация прямого доступа к памяти

1.7.1. Прямой доступ к памяти как механизм высокоскоростного обмена информацией.

Прямой доступ к памяти (ПДП или DMA — Direct Memory Access) представляет собой высокоскоростной способ обмена информацией между периферийным устройством и основной памятью, например, при загрузке данных в оперативную память с внешнего носителя. В режиме ПДП обмен данными между периферийными устройствами и основной памятью или между основной и внешней памятью осуществляется автономно от центрального процессора. В этом режиме скорость передачи данных определяется только внешними устройствами и благодаря этому компьютер может выполнять ввод/вывод с максимальной скоростью самих внешних устройств. Кроме автономности от процессора и высокой производительности обмена важным свойством ПДП является то, что переход в этот режим чаще всего выполняется асинхронно по отношению к основной программе ЭВМ, по запросу от УВВ, подобно прерыванию. Но в отличие от прерывания, где процессор, переходя к обработке запроса, продолжает управлять шиной, в режиме ПДП управление передается другому устройству, которое можно назвать процессором или контроллером ввода/вывода. Заметим, что переход в режим ПДП может быть выполнен и программно. Здесь опять имеется аналогия с системой прерываний.

Для эффективного управления вводом/выводом и осуществления высокоскоростных ПДП-пересылок в современных компьютерах используют либо специализированные сопроцессоры ввода/вывода, либо специальные контроллеры DMA .

Сопроцессор ввода/вывода – это вспомогательный процессор, работающий в паре с центральным процессором и имеющий собственную систему команд, которая ориентирована на операции ввода/вывода. С помощью процессора ввода/вывода все действия по организации передач ввода/вывода, включая настройку внешнего устройства, программный ввод/вывод и операции ПДП, реализуются без участия ЦП. Сопроцессор, кроме передач данных, при выполнении команд ввода/вывода может реализовывать арифметические и логические операции, переходы, поиск и преобразования данных. Он программируется с помощью ЦП, выполняет его задания и возвращает ему результаты выполнения. Когда команда ввода/вывода встречается в программе, ЦП передает управлении

сопроцессору ввода/вывода и далее сопроцессор работает независимо от ЦП. Сопроцессор ввода/вывода является достаточно сложным устройством. При его использовании для организации ПДП часто применяют раздельные шины доступа к ОЗУ со стороны ЦП и периферийных устройств.

Контроллер DMA по сравнению с процессором ввода/вывода является более простым устройством. Предварительно запрограммированный контроллер DMA в режиме ПДП непосредственно, также без участия ЦП, управляет обменом данных между основной памятью и периферийным устройством. При программировании контроллера DMA обеспечивается его настройка на определенный тип передачи, формирование адресов памяти (память может быть источником или приемником данных), а также размер передаваемого массива данных. Запрограммированная передача в режиме ПДП инициируется по запросу контроллера DMA и реализуется параллельно с выполнением центральным процессором своих программ. При использовании контроллера DMA раздельные шины доступа к ОЗУ обычно не применяют. Обычно, в случае отсутствия запроса прямого доступа от контроллера DMA, системной шиной управляет процессор, осуществляющий обмен данными обычным образом. При поступлении запроса прямого доступа управление системной шиной передается контроллеру DMA, который формирует все необходимые для передачи данных сигналы управления и обеспечивает требуемый обмен. Процессор в данном случае отключается от системной шины, переключая свои тристабильные буферы шины в состояние высокого сопротивления. Контроллер управляет системной шиной и производит передачу данных, до полного завершения этой операции.

В большинстве архитектур предусмотрен очень похожий на ПДП режим, который называется захватом шины или прямым управлением шиной (в англоязычной литературе для подобных режимов часто используется название Bus Mastering). Переход в такой режим начинается по запросу от УВВ, но в отличие от ПДП управление шиной получает не контроллер DMA, а контроллер УВВ, который берет на себя полную ответственность за обмен. При этом обмен может осуществляться не только между УВВ и памятью, но и между двумя УВВ.

1.7.2. Контроллер ПДП IBM PC-совместимых персональных ЭВМ.

Контроллер DMA, как правило, реализуется в виде БИС, управляющей работой нескольких независимых каналов прямого доступа к памяти. В персональных ЭВМ используется контроллер i8237A (в ранних моделях) и его логические аналоги. В качестве примера рассмотрим реализацию режима ПДП под управлением такого контроллера. Эта реализация обладает всеми характерными свойствами рассматриваемого режима обмена информацией с устройствами ввода-вывода (УВВ).

Сама по себе микросхема i8237A поддерживает работу четырех независимых каналов обмена. Каждый канал обслуживает одно УВВ. Ранние модели ПЭВМ имели единственную микросхему i8237A. Более поздние модели оснащались парой каскадно включенных контроллеров, подобно контроллерам прерывания, так что общее число каналов выросло до семи (один из входов контроллера израсходован на каскадирование). Логический эквивалент такой пары просуществовал до настоящего времени в ЭВМ, имеющих шину расширения ISA или ей подобную. При одновременном поступлении запросов ПДП каналы конкурируют на системной шине согласно заданным приоритетам.

Рис. 1.14 и 1.15 иллюстрируют характер взаимодействия контроллера DMA, центрального процессора и УВВ, подключенного к шине расширения. Для примера рассматривается взаимодействие на шине ISA, поэтому на рисунках использованы соответствующие обозначения сигналов. Инициатором обмена является УВВ. Цикл обмена начинается с того, что УВВ посылает сигнал по своей индивидуальной шине запроса DRQ на вход контроллера DMA. Тот, в свою очередь направляет запрос по линии HRQ на вход HOLD процессора. Когда это оказывается возможным, процессор разрешает обмен сигналом HLDA в контроллер. Последний подтверждает переход в режим ПДП сигналом DACK по индивидуальной линии, адресованным устройству, которое делало запрос. Одновременно контроллер устанавливает сигнал AEN (Address Enable), который оповещает все остальные устройства, о том, что на шине выполняется цикл ПДП. С этого момента именно контроллер DMA является на шине ведущим устройством (задатчиком). После этого контроллер формирует цикл обмена, аналогичный циклу, представленному на рис. 1.2 и 1.3. Отличие от простого адресного обмена состоит в том, что формируются одновременно два строба: строб чтения источника информации и строб записи в приемник информации. Для шины ISA это либо пара (IOR#, MEMW#), либо пара (MEMR#, IOW#), т.е. чтение УВВ—запись в память, либо чтение памяти—запись в УВВ. Обозначения сигналов шины см. в п.1.3.3. Отметим, что строб записи должен «охватываться» стробом чтения, как это показано на рис. 1.15. Если УВВ не может поддержать темп выполнения цикла, формируемого контроллером, то оно может замедлить цикл путем «запрета» сигнала IO CH RDY, как показано на диаграмме. Тогда продолжительность цикла увеличится на время пребывания этого сигнала в состоянии низкого уровня.

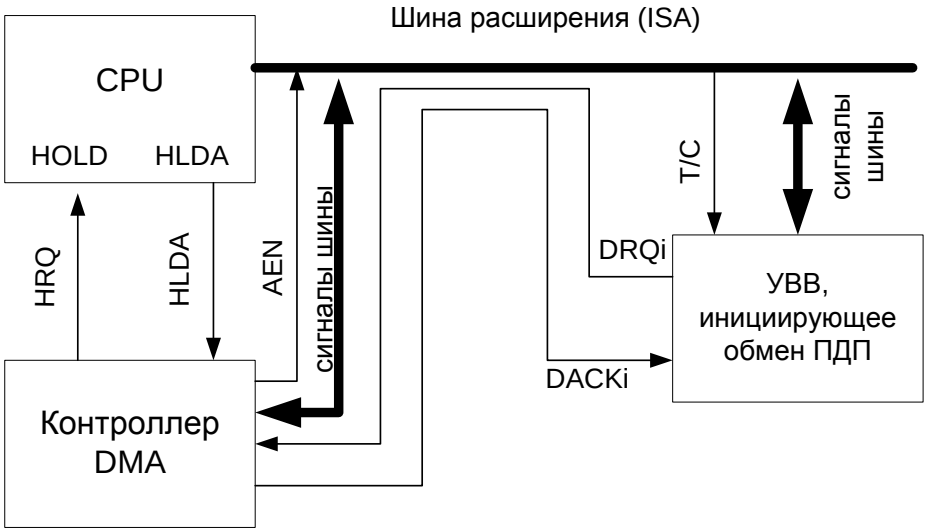

Рис. 1.14. Взаимодействие контроллера DMA с центральным процессором и УВВ.

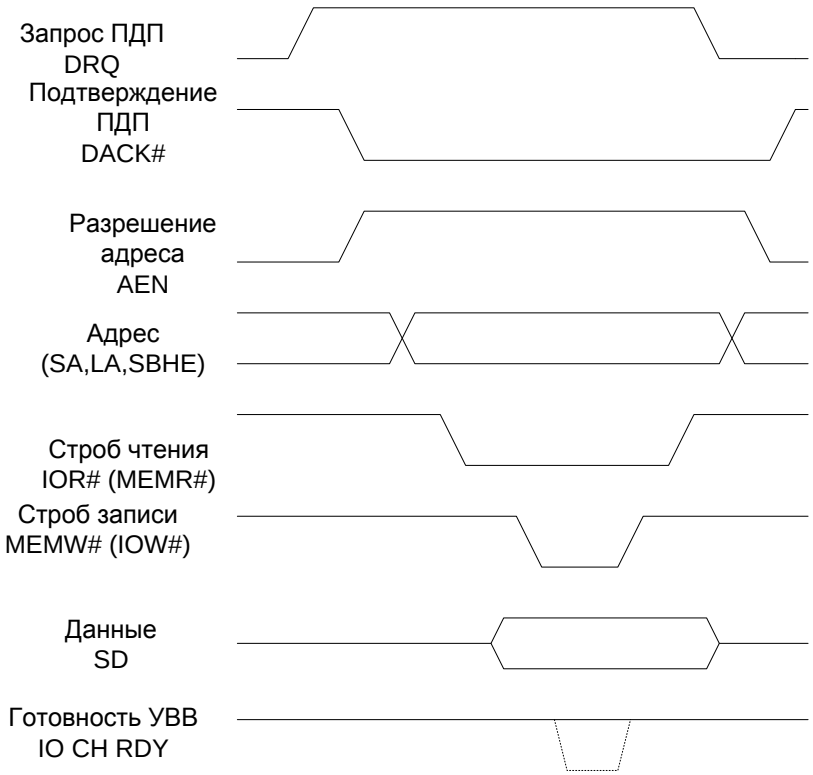

Рис. 1.15. Временная диаграмма цикла ПДП.

Обычно в режиме ПДП выполняется обмен массивами единиц информации. Тогда цикл, показанный на рис.1.15, повторяется с измененным адресом. Повторение циклов обмена прекращается либо, когда будет обнулен счетчик переданных единиц информации, либо, когда УВВ снимет свой запрос DRQ.

Все управляющие сигналы во время цикла ПДП формируются контроллером DMA (кроме считанных данных и ответных сигналов от ведомого устройства). Заметим, что адреса, устанавливаемые контроллером на шине адреса, относятся памяти, а не к УВВ. Эти адреса могут совпадать с адресами регистров УВВ (портов), а это, в сочетании со стробами чтения/записи УВВ, которые также формируются контроллером DMA, могло бы привести к ложной реакции УВВ на эти сигналы. Для предотвращения ошибочного срабатывания формируется сигнал AEN, который запрещает реакцию схем селекции адреса всех УВВ на адресные сигналы. В то же время УВВ, участвующее в цикле ПДП, работает нормально, так как оно выбрано сигналом DACK (на шине ISA по индивидуальной линии).

Из всего сказанного выше можно сформулировать следующий перечень функций контроллера:

- прием запросов DRQ на ПДП от контроллеров УВВ,
- формирование запроса HRQ в процессор на переход в режим ПДП,
- получение от процессора сигнала HLDA, разрешающего режим ПДП,
- формирование индивидуального для УВВ, имеющего в данный момент высший приоритет, сигнала DACK, а также общего для всех устройств на шине сигнала AEN, оповещающих о начале работы в режиме ПДП,
- формирование адреса ячейки памяти и стробов чтения/записи памяти и УВВ в соответствии с направлением передачи данных,
- модификация адреса памяти для повторения цикла обмена при передаче массива единиц информации,
- повторение цикла обмена до появления признака завершения передачи массива,
- снятие запросов на ПДП и сигнала AEN по завершении передачи массива.

Особенностью рассматриваемого контроллера DMA является то, что для каналов с номерами 0…3 единицей передаваемой информации является 8-разрядный байт, а для каналов 5…7 — 16-разрядное слово. При этом сигналы выбора 16-разрядного цикла (MEM CS 16# и IO CS 16# — см. п.1.3) контроллером DMA игнорируются. Канал 4 отсутствует, так как в первых моделях IBM PC/AT, содержавших пару контроллеров i8237, этот канал был использован для каскадирования этих микросхем.

Контроллер DMA является программируемым устройством и допускает множество вариантов настройки. Каналы ПДП программируются индивидуально. Каждый из каналов может работать в одном из трех режимов:

- режим одиночной передачи (соответствует диаграмме рис.1.15),
- режим блочной передачи (цикл чтения/записи повторяется заранее установленное число раз),
- режим передачи по требованию (циклы повторяются, пока присутствует сигнал запроса от УВВ)

Имеется также четвертый режим — режим каскадирования, в котором находится используемый для каскадирования канал.

Кроме режима программируется направление передачи (в память или из памяти), направление модификации адреса (увеличение или уменьшение), расположение в памяти массива ячеек, участвующих в операциях обмена данного канала, дисциплина приоритетного обслуживания запросов и ряд других параметров. Для настройки параметров обмена контроллер содержит несколько десятков регистров (портов), расположенных в начале адресного пространства ввода/вывода.

1.7.3. Прямое управление шиной или режим захвата шины.

Работу в этом режиме также поясним на примере шины ISA. На рис.7.9 показано, что УВВ, способное захватить шину, начинает работу с того, что устанавливает запрос DRQ, который, в соответствии с рис.7.7, поступает в контроллер DMA. Последний, в свою очередь, начинает выполнять те же действия, что и при входе в режим ПДП, но по получении подтверждения DACK УВВ устанавливает сигнал MASTER#, который возвещает о том, что управление шиной получило УВВ. Основной контроллер DMA, получив,

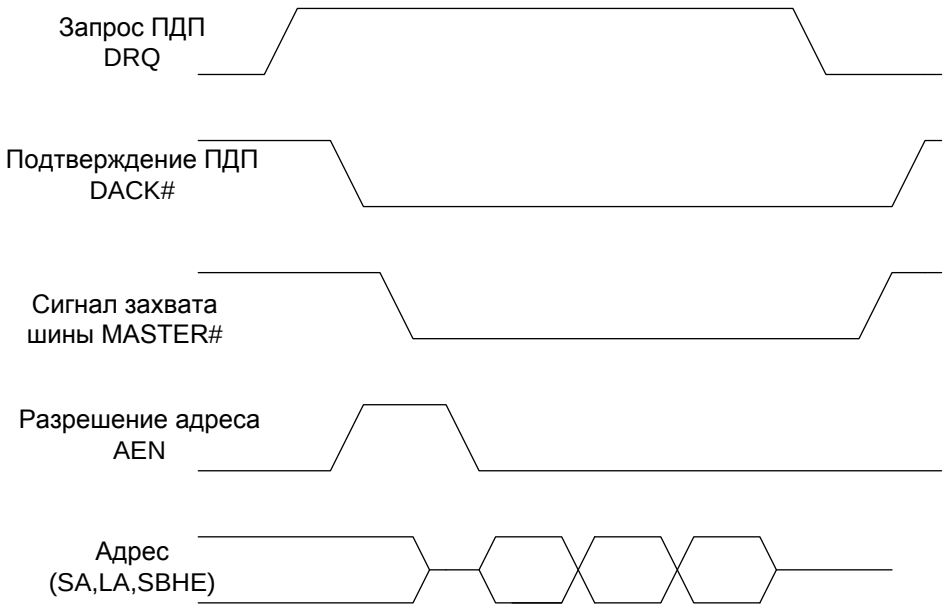

Рис.1.16. Временная диаграмма цикла захвата шины

сигнал MASTER#, снимает свой AEN. Далее всеми действиями управляет захватившее шину УВВ. На рис. 1.16 показаны три цикла передачи данных. Для упрощения показаны только адресные сигналы. «На фоне» адресных сигналов выполняются обычные циклы адресного обмена, а стробы записи/чтения, данные и ответные сигналы во времени размещены так же, как показано на рисунках 1.2 и 1.3.

Все адресные и управляющие сигналы во время обмена в этом режиме формируются устройством, захватившим шину, которое кроме всего прочего должно обеспечить выполнение цикла регенерации памяти, если захват шины продолжается более периода регенерации.

1.7.4. Особенности режима захвата шины на шине PCI.

Режим ПДП «в буквальном смысле» на шине PCI отсутствует, т.е. на шине нет центрального контроллера DMA, подобного контроллеру i8237A, описанному в п.1.7.2. Отсутствуют, таким образом, каналы DMA, заранее запрограммированные на определенный тип обмена. Однако режим, подобный захвату шины, имеется. Для его реализации PCI-устройства, которые могут стать ведущими (инициаторами) на шине, имеют подключенные к шине внешние выводы REQ# (Request, т.е. запрос на захват

шины) и GNT# (Grant, т.е. предоставление управления шиной устройству, посылавшему запрос). Настройка контроллеров обмена в этом режиме может выполняться по программе центральным процессором. Начало обмена может также инициироваться центральным процессором, или же самим контроллером PCI-устройства. Во всех вариантах собственно обмен происходит под управлением контроллера устройства, внешнего по отношению к процессору, чем достигается разгрузка центрального процессора.

Для обеспечения совместимости PCI-устройств с ранее разработанным программным обеспечением для ПЭВМ архитектурной лини IBM PC имеется протокол PC/PCI DMA и соответствующая схемотехническая поддержка [14]. PCI-устройства, выполненные в соответствии с требованиями этого протокола, имеют внешние по отношению к шине PCI сигналы DRQ и DACK, аналогичные сигналам шины ISA. По линиям REQ# и GNT# шины PCI в последовательном коде передаются номера каналов, по которым поступили запросы на захват шины и номер канала, которому предоставляется управление шиной.

2. ЛАБОРАТОРНЫЙ ПРАКТИКУМ

2.1. Работа с регистрами периферийных устройств

2.1.1. Цели работы

Целью работы является ознакомление с организацией доступа к регистрам ввода/вывода (к портам), а также с возможностями исследования периферийных устройств ПЭВМ архитектурной линии IBM PC при помощи редактора портов.

2.1.2. Общие положения

Периферийные устройства (ПУ), взаимодействующие с центральной частью ЭВМ через контроллеры (см. п.1.2), в программной модели вычислительной системы представлены регистрами ввода-вывода, которые подключены к системной шине и доступны процессору для чтения/записи. С точки зрения процессора регистры ввода-вывода различаются по адресам.

В архитектуре IBM PC регистры ввода-вывода (порты) имеют адреса в диапазоне от 0 до 0ffff. Порт можно рассматривать как байтовую ячейку в пространстве адресов ввода-вывода.

Поскольку, как правило, для обмена информацией с ПУ и управления его работой требуется несколько однобайтовых регистров, то для полноценного взаимодействия используется группа портов по смежным адресам. Начальный адрес такой группы называется базовым адресом устройства. Например, последовательный канал связи COM1 подключен через порты 03f8h…03fdh, базовый адрес равен 03f8h.

Доступ к портам обеспечивают две группы команд процессора: **`in`** — для чтения, **`out`** — для записи. Возможен доступ либо к одному порту, либо (для процессоров, начиная с i80286) сразу к двум смежным портам, которые рассматриваются как 16-разрядное слово, либо (для процессоров, начиная с i80386) сразу к четырем смежным портам (в дальнейшем не рассматривается).

На практике наиболее часто требуется доступ к отдельным портам, реже — к двум смежным портам. Обмен производится через регистр процессора **`al/ax`**, адрес порта задается либо в регистре **`dx`**, либо (если адрес не больше 0ff) непосредственно в самой инструкции. В ассемблерных программах операторы чтения/записи портов записываются следующим образом:

```
in  al, dx   ;считываем порт по адресу, заданному
             ;регистром dx, в регистр al

in   al, 040 ;адрес, не превышающий 0ffh, задается
             ;непосредственно

out  070, al ;записываем содержимое регистра al  в
             ;порт по адресу 070

mov  dx, 03f8
out  dx, ax  ;записываем содержимое ax в два
             ;смежных порта по адресам (dx),(dx)+1
```

В систему команд, начиная с процессора i80286, включены также строковые варианты команд **in/out: ins/outs.** С их помощью можно считывать/записывать последовательность байт (**insb/outsb**) или слов (**insw/outsw**). Обмен осуществляется между портом (парой портов) и памятью, минуя регистр **al/ax**. Адрес приемника/источника в памяти задается регистрами **es:di/ds:si** и автоматически изменяется после выполнения чтения/записи в соответствии с установкой флага направления d и размерностью операндов (содержимое регистра **di/si** увеличивается или уменьшается на 1 или на 2). Адрес порта задается в **dx** и не изменяется командой **ins/outs**. С командами **ins/outs** можно использовать префикс повторения **rep;** предварительно в **cx** должно быть записано число повторений.

Пример:

```
mov  dx, 040      ; адрес порта
lea  di, dest     ; адрес приемника в памяти
cld               ; направление в сторону увеличения
mov  cx, 1000     ; счетчик повторений
rep  insb
int  020          ; выход в DOS
dest db
```

Команда **rep insb** в этом примере заменяет цикл:

```
l1:      in   al, dx
         stosb
         loop l1
```

2.1.3. Инструментальные средства

Для выполнения работы используется ПЭВМ стандартной конфигурации, резидентный редактор портов rport.exe, резидентная утилита digit.exe для представления чисел в разных системах счисления.

2.1.4. Программа выполнения работы. Исследование периферийных устройств вручную

Во многих случаях изучение периферийного устройства имеет смысл начинать с простейших операций на уровне команд **in/out**. Для таких операций подходит практически любой отладчик. Мы будем использовать более удобный в обращении резидентный редактор портов rport.exe.

Запуск редактора портов `rport`

Запустите **rport** (его запуск рекомендуется включить в **autoexec.bat**). Активизируйте редактор портов нажатием Alt-Z. Для доступа к порту наберите адрес порта в шестнадцатиричном формате и нажмите клавишу « / » (рекомендуется использовать дополнительную клавиатуру), например:

```
3f8/
```

Результат чтения порта 03f8h выводится справа от « / ». Для повторного чтения порта еще раз нажмите "/". Результат вместе с адресом будет выведен на следующей строке. Для чтения порта по следующему адресу (03f9h) нажмите «+». Для чтения порта по предыдущему адресу используйте клавишу «-». Если необходимо заново задать адрес, предварительно нужно перейти в начало следующей строки по клавише «Enter».

Определение наличия устройств

Проведите эксперимент по определению количества адаптеров последовательной связи, установленных на вашем компьютере. В IBM PC их может быть до двух (в DOS им присвоены названия COM1, COM2); каждый занимает от 6 до 8 смежных портов (количество зависит от модификации), базовые адреса - 3f8h и 2f8h.

Для определения наличия внешнего устройства нужно прочесть порты, предназначенные для подключения этого устройства. Результат чтения 0ffh обычно свидетельствует о том, что доступа к соответствующему регистру не было, либо к порту ничего не подключено, либо регистр устройства доступен

только для записи. Таким образом, если результат чтения всех портов устройства - 0ffh, то устройство, скорее всего, отсутствует.

Прочитайте шесть портов, начиная с адреса 03f8h, затем начиная с адреса 02f8h. Сколько адаптеров последовательной связи установлено на вашем компьютере? Какие у них базовые адреса?

Определите количество адаптеров параллельной связи с принтерами, установленных на вашем компьютере. Каждый из таких адаптеров занимает три смежных порта. Базовые адреса - {0378h, 0278h} или {03bch, 0378h}.

Рассмотренный способ дает практически достоверную информацию об отсутствии устройства. Вместе с тем, наличие устройства в некоторой зоне адресов не всегда свидетельствует о том, что это за устройство. Например, при включении адаптера EtherNet, базовый адрес которого настраивается переключателями на плате, возможно «попадание» в одну из рассмотренных зон адресов. Такое пересечение допустимо, если в результате эту зону использует только одно устройство.

Изучение методов доступа к регистрам устройства

Наличие в составе стандартной ПЭВМ разнообразных и сложных ПУ со своими контроллерами привело к тому, что адресного пространства ввода/вывода оказывается недостаточно для прямой адресации всех регистров контроллеров ПУ, которые могут входить в комплекс. Для обмена информацией со всеми регистрами применяются различные способы доступа, которые позволяют при небольшом числе портов (адресов в пространстве ввода/вывода) обращаться к более широкому множеству регистров. Можно сказать, что способ доступа к регистрам устройства определяется схемой подключения регистров к портам. Примером является множество регистров ПКП (см. п.1.4 настоящего пособия), доступ к которым осуществляется по двум адресам ввода/вывода.

Доступ при непосредственном подключении регистров к портам

Наиболее просто доступ организуется при непосредственном подключении регистров к портам. В этом случае чтение/запись порта означает чтение/запись соответствующего регистра устройства, т.е. реализуется прямая адресация.

Обычно разработчики периферийного оборудования выбирают такой способ подключения, если регистров немного. Характерный пример - адаптер параллельной связи с принтером; каждый из его трех байтовых регистров подключен к отдельному порту.

Если объем подключаемых регистров устройства превышает 6-7 байт, то с целью экономии адресного пространства ввода/вывода к одному порту

подсоединяют несколько регистров через ***схему коммутации***, которая обеспечивает доступ к нескольким регистрам по одному адресу.

Коммутация по чтению/записи

В этом варианте коммутации к одному порту подключается два байтовых регистра. Коммутация производится по уровню сигнала чтения/записи порта на системной шине: при чтении подключается один регистр, при записи — другой. Для шины ISA это сигналы -IOR и -IOW (см. рис.1.3).

Такой способ коммутации используется, например, в адаптере последовательного канала связи. Последовательный канал связи в составе IBM PC позволяет передавать и принимать данные. Регистр данных передатчика и регистр данных приемника доступны через один порт, находящийся по смещению 0 относительно базового адреса устройства. При выполнении команды чтения порта к шине данных подключается регистр данных приемника, при записи — регистр данных передатчика.

Прочитайте порт 03f8h (или 02f8h, если по адаптер по адресу 03f8h отсутствует). Результат чтения — значение регистра данных приемника.

Запишите в порт 03f8h произвольное значение, отличающееся от прочитанного. Для записи в порт при использовании редактора портов Port Editor нужно продолжить ввод на той строке, где выведен результат чтения; ввод завершается нажатием либо «Enter», либо «+» (для перехода к адресу на единицу больше), либо «-» (для перехода к предыдущему адресу).

При записи в порт 03f8h к нему был автоматически подключен регистр данных передатчика. Значение, записанное в регистр данных передатчика прочитать невозможно, так как при чтении порта 03f8h коммутируется регистр данных приемника. Прочитайте еще раз порт 03f8h, сравните с тем, что было записано.

З а м е ч а н и е. Несовпадение результатов записи и чтения какого-либо порта — довольно обычная ситуация. Это может быть связано с тем, то некоторые разряды регистра устройства доступны только для чтения.

Прочитайте порт по смещению 2. Этот порт подключен к регистру идентификации прерываний последовательного адаптера. Единица в нулевом бите означает, что заявки на прерывание от COM1 отсутствуют, единичные значения в битах 1...3 говорят о том, какие из трех возможных заявок установлены. Этот регистр доступен только для чтения, так как отражает состояние устройства.

Запишите в этот порт число 0eh (биты 1...3 установлены в единицу). Это действие означает попытку навязать устройству состояние, не соответствующее действительности (если горит лампочка "пожар", разбить лампочку, и пожар прекратится). После записи прочитайте порт. Объясните результат чтения.

Коммутация, управляемая через отдельный порт

При таком способе линии управления коммутатором подключены к отдельному порту. Схема управления зависит от количества подключаемых регистров.

Простейший вариант коммутации — для двух регистров — использован в адаптере последовательной связи (приемопередатчик COM-порта). В нем имеется 16-разрядный регистр делителя частоты, который определяет скорость приема/передачи. Этот регистр подключен к портам по смещениям 0 и 1 — через коммутатор, ко входам которого также подключены байтовый регистр данных (к порту по смещению 0) и байтовый регистр разрешения прерываний (к порту по смещению 1).

Прочитайте порты адаптера по смещениям 0 и 1. Сейчас эти порты подключены к регистру данных и к регистру разрешения прерываний. Запомните значения.

Прочитайте порт по смещению 3. К нему подключен управляющий регистр адаптера, который определяет параметры настройки канала связи, кроме скорости передачи. Старший бит этого регистра управляет коммутатором, который подключает порты по смещениям 0 и 1 либо к регистру делителя частоты, либо к регистрам данных и идентификации прерываний. Для коммутации регистра делителя частоты, установить старший бит управляющего регистра в 1. Перед тем, как записать требуемое значение 080, запомните прочитанное число.

Прочитайте порты по смещениям 0 и 1. Через порт по смещению 0 сейчас доступен младший байт 16-разрядного делителя частоты, через порт по смещению 1 — старший байт. Полученное 16-разрядное значение делителя переведите в десятичную систему счисления, и вычислите настройку скорости передачи по формуле r = 115200 / dv, где r — скорость (бит/с), dv — коэффициент из регистра делителя частоты.

Запишите в управляющий регистр исходное значение, затем прочитайте порты по смещениям 0 и 1. Что изменилось? Какие регистры устройства вы прочитали?

При увеличении числа регистров, доступ к которым требуется в равной степени, применяется более развитая схема коммутации регистров — адресная. Такая схема принята при организации доступа к CMOS-памяти в составе микросхемы Motorola MC146818, которая обеспечивает хранение данных о конфигурации вычислительной системы, а также данных часов реального времени. Все эти данные хранятся в 64 однобайтовых регистрах CMOS-памяти. Для доступа к ним используются всего два порта: порт по адресу 070h, подсоединенный к адресному коммутатору, и порт по адресу 071h, к которому подключается один из 64 регистров CMOS. Запись числа в порт 070h определяет номер регистра CMOS, связанного с портом 071h.

Запишите в порт 070h число 4. В результате к порту 071h подключен четвертый регистр CMOS, в котором хранится текущий час. Прочитайте порт 071h.

З а м е ч а н и е. Данные часов реального времени обычно представлены в двоично-десятичном формате (в каждой позиции шестнадцатеричного числа — десятичная цифра).

Прочитайте текущие минуты и секунды — в регистрах номер 2 и 0. В первых реализациях микросхемы Motorola MC146818 коммутатор адреса сбрасывается каждый раз после чтения или записи регистра данных; рекомендуется каждый раз перед доступом к порту 071h записывать номер регистра в порт 070h, даже если этот номер прежний.

Последовательная коммутация

При последовательной коммутации регистры подключаются к порту по очереди после каждого обращения к порту. При такой схеме отпадает необходимость в управлении коммутатором через отдельный порт.

Последовательную коммутацию двух байтовых регистров иллюстрирует микросхема трехканального таймера 8253/8254. Это устройство содержит три независимо работающих таймера (каналы с номерами 0, 1, 2).

Централизованный регистр команд подключен к порту 043h. Регистры данных у каждого канала свои: данные канала 0 подключены к порту 040h, данные канала 1 — к порту 041h, данные канала 2 — к порту 042h. У каждого канала имеется собственный набор 16-разрядных регистров данных. Для нас интерес представляют регистр начального значения и регистр счета — для доступа к ним используется последовательная коммутация.

Когда счет разрешен, импульсы с частотой 1193 кГц поступают на вход вычитания регистра счета, и с каждым импульсом значение счетчика уменьшается на 1. При обнулении счетчика на выходе канала генерируется сигнал, который используется другими устройствами (например, выход канала 2 подключен через промежуточный вентиль к динамику). В режиме циклического счета после обнуления счетчика в нем устанавливается исходная величина из регистра начального значения, и счет возобновляется.

В этом устройстве последовательная коммутация используется для доступа через один порт к 16-разрядному регистру. При очередном обращении к порту коммутируется младший или старший байт 16-разрядного регистра. Таким образом для доступа к двум байтам регистра требуется двукратный доступ к порту.

Для двукратного чтения в редакторе Port Editor рекомендуется вместо « / » (косая черта) использовать клавишу «*». Наберите адрес 040h и нажмите «*». При текущей настройке канала 0 чтение порта 040h последовательно коммутирует старший и младший байты счетчика этого канала. Еще

несколько раз нажмите «*». Из того, что значения меняются, следует, что счет для этого канала разрешен.

Аналогично проверим состояние канала 2. Несколько раз прочитайте двойным чтением порт 042h. Данные не меняются, счет запрещен. Для разрешения счета этому каналу нужно установить в единицу нулевой разряд порта 061h (подключен к микросхеме интерфейса с периферией).

Прочитайте порт 061h и запишите в него прочитанное значение с единицей в младшем разряде.

З а м е ч а н и я:

- Для преобразования шестнадцатеричных чисел, с которыми работает **`rport`**, в двоичные можно использовать резидентную утилиту **`digit.exe`** (после запуска активизируется по Ctrl-Z). В строке «Hex» вводится полученное число в шестнадцатеричном формате, и после нажатия «Enter» или перехода на другую строку клавишей-стрелкой это число будет показано во всех форматах.
- Старшая тетрада регистра, подключенного к порту 061h, недоступна для записи, и можно не заботиться о ее сохранении. Например, если значение, прочитанное из порта 061h, равно 020h или 030h, то для запуска счета второго канала таймера можно записать в порт 061h единицу.

Проверьте, разрешен ли теперь счет для канала 2.

Продолжим работу с этим каналом до получения звука заданной частоты. Для того, чтобы импульсы с выхода этого канала поступали на динамик, нужно открыть промежуточный вентиль, за счет установки в единицу первого разряда порта 061h. Не путайте первый разряд (bit 1) с младшим нулевым разрядом (bit 0). Выполните эту операцию. Для отключения звука запишите в порт 061h исходное значение.

Чтобы изменить частоту, нужно записать начальное значение счета в соответствующий регистр канала 2. Запишите в порт 043h число 0b6h. Это код команды «запись двух байтов регистра начального значения счетчика для канала 2 и установка режима циклического счета». По этой команде включается последовательная коммутация двух байтов регистра начального значения; первая запись «направляется» в младший байт регистра, вторая — в старший.

З а м е ч а н и е. Регистр таймера, куда записывается начальное значение счета, называется регистром констант пересчета (CR). Его содержимое автоматически переписывается в счетчик, когда последний обнуляется, при условии, если канал таймера переведен в циклический режим работы. Именно такой ражим и предлагается установить. **Будьте внимательны:** в регистр CR требуется записать именно коэффициент пересчета, а не значение заданной частоты, как ошибочно полагают иногда студенты.

Для примера настроим канал 2 таймера на частоту 1000 Гц. Для этого после записи команды 0b6h в порт 043h запишем в регистр начального значения счетчика число 1193 (шестнадцатеричное 04a9h). Сейчас через порт 042h доступен младший байт счетчика, поэтому записываем в этот порт число 0a9h. После записи порт 042h подключен к старшему байту счетчика. Запишите 4.

Включите звук. Прочитайте несколько раз нажатиями «*» порт 042.

Выключите звук. Можно ли определить по результатам чтения, какой байт счетчика читаетсяется первым — младший или старший?

Получите звук с частотой, заданной преподавателем. Для самоконтроля используйте программу **sound.exe**, значение частоты указывайте в командной строке. Например,

```
sound.exe 1000
```

генерирует звук с частотой 1000 Гц (длительность звучания всегда одна и та же — около 0.6 с).

Чтобы проверить правильность выполнения задания по настройке частоты звука, целесообразно поступить следующим образом:

- настройте канал 2 таймера на заданную частоту,
- включите звук,
- закройте экран **rport** (клавишей Esc),

запустите программу **sound.exe** с указанием заданной частоты в командной строке.

Если все действия выполнены правильно, то перед прекращением звучания Вы не услышите изменения высоты тона.

Управление устройствами

Для управления устройствами в большинстве случаев достаточно организовать доступ к его регистрам по чтению/записи. Схемы доступа рассмотрены в предшествующем изложении. Регистры устройства можно разбить на следующие группы: регистры состояния, регистры управления, регистры данных.

В регистрах состояния отражается текущее состояние устройства, или события, связанные с его работой: завершение операции ввода/вывода (приема/передачи), ошибки, временные события и т.д. Регистры управления предназначены для ввода команд, определяющих режим или работы или параметры функционирования. Регистры данных обеспечивают запись данных для операций вывода (передачи) или чтение данных, полученных в результате операции ввода (приема).

Работа с регистром управления опробована на примере системного таймера. Работу с регистрами данных и состояния проиллюстрируем на примере адаптера последовательной связи.

Регистр состояния последовательного адаптера имеет смещение 5. Его разряды имеют следующее назначение:

Бит	Состояние
0	Готовность приемника (данные приняты).
1	Ошибка переполнения при приеме (до начала приема не был прочитан регистр данных приемника, где сохранялся результат предыдущего приема).
2	Ошибка четности при приеме.
3	Ошибка синхронизации (не принята стоповая посылка).
5	Буферный регистр передатчика пуст, можно записывать байт для передачи.
6	Сдвиговый регистр передатчика пуст, передача закончена.

Биты 0...3 отражают состояние приемника, биты 5 и 6 — состояние передатчика.

Прочитайте порт 03fdh. Охарактеризуйте состояния приемника и передатчика.

Регистр данных приемника и регистр данных передатчика подключены к одному порту по смещению 0 (для COM1 - порт по адресу 03f8) и коммутируются по чтению/записи (см. соответствующий пункт программы выполнения работы).

При выполнении этого опыта будем предавать данные из передатчика в приемник одного и того же устройства. В этом случае для наблюдения за результатами приема и передачи выход передатчика должен быть подключен ко входу приемника. В лаборатории кафедры на выходных разъемах адаптера последовательной связи поставлена такая перемычка, т.е. организована линия связи. Имеется возможность эту линию подключить к осциллографу, чтобы непосредственно наблюдать сигналы в ней. При отсутствии физической линии связи можно воспользоваться диагностическим режимом адаптера. В этом режиме выход передатчика замкнут внутри адаптера на вход приемника. Для перехода в диагностический режим установите в единицу бит 4 регистра управления модемом, подключенный к порту по смещению 4. Наблюдать сигналы с помощью осциллографа в этом режиме нельзя.

Запишите в регистр данных передатчика значение 055h. Запись в него автоматически запускает передачу. При работе вручную передача практически мгновенная (при начальной загрузке BIOS настраивает последовательные адаптеры на скорость приема/передачи 2400 бит/с, и передача занимает не более 5 мс).

Прочитайте регистр состояния. Передача завершена, о чем свидетельствует бит готовности приемника. Готовность приемника сохраняется, пока не прочитаны принятые данные. Прочитайте регистр

данных приемника, затем регистр состояния. Состояние изменилось: признак готовности приемника сброшен в результате чтения регистра данных приемника.

Для наблюдения за процессом приема/передачи уменьшим его скорость до минимума. Скорость определяется значением делителя частоты (доступ к соответствующему регистру рассмотрен в пункте программы выполнения работы «Коммутация, управляемая через отдельный порт»). Запишите в регистр делителя частоты максимальное значение 0ffff. В результате скорость уменьшится до 115200 / 65535 = 1.76 бит/с, а продолжительность приема/передачи возрастет по крайней мере до 4 с. Восстановите исходное значение разряда, управляющего коммутацией портов по смещениям 0 и 1. При этом через порт по смещению 0 вновь будет доступен регистр данных приемопередатчика (рекомендуется полностью восстановить содержимое регистра управления, чтобы не изменились остальные настройки).

Запишите в регистр данных передатчика произвольное значение. Быстро прочитайте регистр состояния и продолжайте читать, повторяя нажатия клавиши «/», пока состояние не станет 061, как в предыдущем опыте. Опишите изменения состояний передатчика и приемника в процессе передачи/приема.

Проведем еще один опыт с приемом/передачей, с целью получения ошибки переполнения. Эта ошибка возникает, если прием завершается в тот момент, когда регистр данных приемника занят (не прочитаны результаты предшествующего приема).

Запишите в регистр данных передатчика произвольное число. Сразу же запишите в него еще одно число, любое. Незамедлительно начинайте читать регистр состояния. Продолжайте чтение, пока состояние не станет равным 061.

Для правильной трактовки результатов этого опыта надо учесть буферизацию при приеме и передаче (см. рис. 1.13). У передатчика имеется буферный регистр данных (доступный через порт по смещению 0), где данные хранятся до того момента, когда передача станет возможной, и сдвиговый регистр, обеспечивающий текущую передачу. Когда сдвиговый регистр (при завершении текущей передачи) освобождается, в него немедленно передаются данные из буферного регистра (если буферный регистр не пуст). В результате, если оба регистра свободны, можно записать в буферный регистр передатчика вручную два байта сразу, поскольку первый байт немедленно передается в пустой сдвиговый регистр.

Приемник также содержит сдвиговый и буферный регистры; в сдвиговом регистре формируется результат текущего приема, по окончании которого данные передаются в буферный регистр, что отражается в изменении состояния (готовность приемника). Пока данные не прочитаны, буферный регистр занят. Сдвиговый при этом свободен, и может принимать следующие данные. В конце приема, если буферный регистр приемника все

еще занят (данные из него не прочитаны), то данные будут потеряны, о чем сигнализирует признак ошибки переполнения.

Объясните изменения состояний в процессе приема/передачи; обратите внимание, что признаки ошибок в регистре состояния сбрасываются после чтения этого регистра.

Выключите диагностический режим (если он использовался), сбросив bit 4 регистра управления модемом (порт по смещению 4).

Мы рассмотрели доступ к устройству "вручную" на уровне операций чтения/записи регистров. Применение подобных средств ограничивается проверкой наличия устройства по возможным базовым адресам, а также проверкой работоспособности в целом. В ряде случаев, как, например, в опытах с адаптером канала последовательной связи, оказывается возможным провести более подробное исследование.

2.1.5. Содержание отчета

Для каждого опыта программы выполнения работы (п. 2.1.4) привести формулировку задания, описать последовательность действий по чтению/записи портов — так, как она выглядит в окне **`rport`**, с объяснением результатов операций чтения и целей операций записи. Привести ответы на вопросы, содержащиеся в п. 2.1.4 и, где требуется, пояснить результаты.

Приведите ответы на вопросы, выбранные по указанию преподавателя, из следующего списка.

1. Запишите команду (одну) для чтения порта 043.

2. Запишите команду (одну) для чтения содержимого делителя частоты последовательного канала (считаем, что в управляющем регистре приемопередатчика последовательного канала бит 7 уже установлен в единицу).

3. Какой регистр последовательного канала доступен при чтении порта по смещению 0? Какой при записи? (Считаем, что устройство находится в исходном состоянии).

4. Результат чтения порта 02f8 — число 0ff. Что это значит?

5. Запишите команды (две) для чтения регистра минут часов реального времени.

6. Составьте программу для передачи по последовательного каналу одного байта на скорости 300 бит в секунду.

7. Запишите последовательность команд для чтения текущего значения счетчика канала 0 таймера.

8. Составьте подпрограмму для включения звука с частотой, заданной в регистре bx (в Гц). Вызовите эту подпрограмму для следующих значений параметра: 200, 450, 860, 1240, 3109.

9. Какие три регистра последовательного канала доступны через порт по смещению 0? Запишите три последовательности команд для доступа к этим регистрам.

10. Какие команды записаны неправильно и почему:

```
in    al, 070     ,
out   al, dx      ,
out   dx, ax      ,
in    ax, 03fa    ,
out   041, ax     ?
```

11. Составьте подпрограмму для записи числа из регистра dl в регистрданных параллельного порта (по смещению 0378h).

12. Составьте подпрограмму для изменения значения бита 7 (на противоположное) в управляющем регистре последовательного канала так, чтобы значения остальных битов этого регистра не изменялись.

2.2. Обмен информацией с периферийными устройствами в режимах опроса готовности и прерывания

2.2.1. Цели работы

Целями работы являются знание основных принципов автоматического управления устройством по программе, приобретение навыков программирования обмена информацией и измерения временных характеристик работы устройства.

2.2.2. Общие положения

Управление устройством включает в себя операции настройки, запуск операций вывода, реакцию на завершение операций ввода, восстановление после ошибок. Во многих таких задачах необходимо знать состояние устройства, отраженное в соответствующем регистре устройства. Например, запуск операции вывода через последовательный канал связи за счет записи данных в буферный регистр передатчика возможен лишь тогда, когда этот регистр свободен.

При автоматическом управлении процессор получает информацию о состоянии устройства, а реакция на особенности состояния определяется программой.

В соответствии с общими принципами организации взаимодействия устройств (п. 1.2) особые состояния устройства могут быть обнаружены с

помощью опроса готовности или по запросам на прерывание или прямой доступ (захват шины). В этой работе изучаются два первых режима.

В режиме опроса готовности, процессор периодически, по программе, опрашивает регистр состояния устройства. В режиме прерывания устройство в определенных состояниях прерывает выполнение текущей программы процессора, переключая процессор на другую программу.

2.2.3. Описание лабораторного стенда

Для выполнения работы используется IBM/PC-совместимая ПЭВМ стандартной конфигурации, ассемблер и отладчик. Для измерения времени работы устройства используется программа **`pztimer`**. Опыты проводятся с контроллером последовательного порта ПЭВМ. Для наблюдения процессов, происходящих в линии связи, желательно иметь перемычку между выходом передатчика и входом приемника и кабель, соединяющий эту перемычку со входом осциллографа. При отсутствии этих соединений можно проводить опыты в диагностическом режиме последовательного порта (см. п. 2.1).

2.2.4. Программа работы

При подготовке к работе повторите основные положения организации обмена информацией в вычислительных системах (пп. 1.2 и 1.4). Изучите приведенные ниже программы, используемые при выполнении работы.

Исследуйте режим опроса готовности на примере управления передачей массива информации через последовательный порт. Для этого проделайте следующие опыты:

- передача массива с опросом готовности передатчика,
- попытка передачи того же массива без опроса готовности,
- получение и фиксация ошибки переполнения при приеме.

Исследуйте обмен информацией в режиме прерывания на примере управления передачей и приемом через последовательный порт. Для этого проделайте следующие опыты:

- передача массива данных по прерыванию от передатчика,
- обработка другого запроса на прерывание (по указанию преподавателя).

Определите настройку частоты приемопередатчика путем измерения времени передачи одной посылки.

Исследование режима опроса готовности

Изучите следующую программу.

```
; Программа передачи 1000 байт через
; последовательный порт в режиме опроса
; готовности передатчика

base    equ  03f8          ; базовый адрес COM1
buf     equ  base          ; буферный регистр
                           ; приемника/передатчика
state   equ  base+5        ; регистр состояния


        mov     cx, 1000           ; счетчик циклов передачи
        mov     bl, 'A'            ; данные для передачи

m2:     mov     dx, state
m1:     in      al, dx             ; опрос регистра состояния
        test    al, bit 5          ; буферный регистр
                                   ; передатчика пуст?
        jz      m1                 ; если нет, продолжать
                                   ; опрос

        mov     dx, buf            ; запись
        mov     al, bl             ;  в буферный регистр
        out     dx, al             ;  передатчика

        loop    m2
        ret
```

В этом примере запрограммирована передача 1000 байтов со значением ASCII-кода 'A'. Перед передачей каждого байта выполняется цикл опроса регистра состояния — до тех пор, пока не будет зафиксирован признак «буферный регистр передатчика пуст». Очередной байт записывается в буферный регистр данных передатчика.

Текст программы приведен в файле **send.8**. Транслируйте программу и выполните ее. При наличии физической линии связи наблюдайте на экране осциллографа передачу данных. Если это невозможно, факт передачи косвенно подтверждается временем выполнения программы. Если скорость передачи 2400 бит/с (по умолчанию), то тысяча передач занимает

приблизительно 4 с. Все это время устройство работает непрерывно; процессор при этом работает «вхолостую», тратя время на циклы ожидания.

Метод периодического опроса регистра состояния прост для программирования, но полностью занимает процессор на продолжительные, по сравнению с темпом работы процессора, циклы ожидания. Масштаб этих затрат в рассмотренном примере можно оценить следующим образом.

Закомментируйте команду **`jz`**. Транслируйте программу и запустите ее. Что изменилось и почему? Сколько байт было передано — ни одного, один, два или тысяча? Точный ответ на этот вопрос можно получить, наблюдая сигнал в линии связи. При этом удобно передавать в линию код 0f0h (тогда один положительный импульс соответствует одному байту).

Самостоятельно запрограммируйте опыт по получению ошибки переполнения. Для получения такой ошибки нужно два раза подряд провести запись в регистр данных передатчика. Затем можно переходить к опросу регистра состояния до тех пор, пока в нем не будут обнаружена единица в битах, отмечающих ошибки приема. Если физической линии связи нет, то предварительно перевести адаптер в диагностический режим. Все эти действия, начиная с включения диагностического режима (если это необходимо), должны быть заданы программой. Перед завершением программы не забудьте выключить диагностический режим. Правильно написанная программа должна переслать до переполнения ровно два байта. Многократный запуск программы в процессе отладки может привести к тому, что разряд ошибки в регистре состояния линии (порт с адресом state) окажется установленным во время предыдущих попыток. Поэтому в начале программы рекомендуется выполнить чтение регистров данных и состояния для сброса разрядов готовности приемника и ошибки.

Исследование обмена в режиме прерывания

Для освобождения процессора от постоянного опроса разряда готовности нужно обеспечить автоматическое, по инициативе устройства, переключение на другую программу, которая, выполнив необходимые действия (в примере — запись в регистр данных), вернет управление прерванной программе.

Прерывание — это незапрограммированный переход по адресу, который определяется источником прерывания, с автоматическим сохранением в стеке адреса возврата. Программа, начало которой находится по адресу перехода, должна завершаться инструкцией возврата, для возобновления прерванной программы. Система прерываний IBM PC-совместимых ЭВМ описана в п. 1.4, а также в [6, 7, 12] и др.

В качестве примера приведем программу, которая настраивает адаптер последовательного канала связи на прерывания при возникновении готовности передатчика, а также обрабатывает эти прерывания. Обработка состоит в отображении в левом верхнем углу экрана меняющихся символов.

```
   com = 1              ; настройка программы на работу с
                        ; последовательным портом COM1
   #if com EQ 2
   base    equ  02f8    ; базовый адрес COM2
   irq     equ  3       ; номер IRQ-входа для COM2
   #endif

   #if com EQ 1
   base    equ  03f8    ; базовый адрес COM1
   irq     equ  4       ; номер IRQ-входа для COM1
   #endif

   buf     equ  base    ; адрес порта регистра данных
   ier     equ  base+1  ; адрес регистра разрешения
                        ; прерываний
   modem   equ  base+4  ; адрес регистра управления
                        ; модемом
   ni      equ  8+irq   ; номер прерывания — сумма
                        ; базового значения (8) и
                        ; номера IRQ-входа

           jmp     start

   count   dw  1000     ; счетчик передач

start:
   mov  ds, 0           ; сегментный адрес таблицы — 0
   mov w [ni*4],offset prog ; запись полного адреса
   mov  [ni*4+2], cs    ; в таблицу
   mov  ds, cs          ; восстановление ds
   mov     al, 0b       ; запись 0b в регистр уп-
                        ; равления модемом (иначе
   mov     dx, modem    ; прерывания по готовно-
                        ; сти передатчика в неко-
   out     dx, al       ; торых моделях не гене-
                        ; рируются); этим также
                        ; выдается питание на мышь
   mov     al, bit 1    ; разрешение прерываний
   mov     dx, ier      ; по готовности
   out     dx, al       ; передатчика

   in      al, 021      ; доступ к iRr для раз-
   and     al, not (bit irq); маскирования заявок от
```

```
   out     021, al          ; IRQ-входа с номером irq

   l1:                      ; основная программа
   mov     ah, 1            ; проверка нажатия клавиши
   int     016              ; выход из цикла, если
   jnz     >l2              ; клавиша нажата
   test    count            ; проверка счетчика
                            ; передач
   jnz     l1               ; выход, если обнулен
   l2:
   mov     dx, ier          ; в завершение — сброс
                            ; разрешения прерываний
   mov     al, 0            ; в регистре ier
   out     dx, al           ; приемопередатчика

   ret
prog:          ; п/п обслуживания прерывания

   push    ds, es, ax, dx     ; сохранение используемых
                              ; регистров
   mov     ds, cs             ; принудительная установка
                              ; ds - так надежнее
   mov     ax, count
   mov     es, 0b800          ; вывод младшего байта
   es mov  [0], al            ; счетчика в верхнем левом
                              ; углу экрана, чтобы
                              ; убедиться, что
                              ; прерывания происходят
   dec     count
   jnz     >l1                ; последняя передача?

   mov     dx, ier            ; да — запрет дальнейших
   mov     al, 0              ; прерывания по готовности
   out     dx, al             ; передатчика

   l1:
   mov     dx, buf            ; запись в регистр данных
   out     dx, al             ; передатчика - запуск
                              ; передачи

   mov     al, 020            ; сброс заявки в регистре
                              ; iSr ПКП
   out     020, al            ; контроллера прерываний
```

```
pop     dx, ax, es, ds      ; восстановление регистров
iret                        ; возврат к прерванной
                            ; программе
```

Текст программы приведен в файле **i_send.8.** Изучите программу. Обратите внимание на начальную часть ее текста. Там с помощью операторов условной трансляции задаются адреса регистров приемопередатчика. Для настройки программы на другой COM-порт достаточно просто изменить его номер в первой строке программы. Обратите также внимание на формирование номера прерывания и определение адреса вектора прерывания. Номер прерывания определяется как сумма номера *x* запроса (IRQ*x*) и начального номера, установленного для данного ПКП при инициализации командой ICW2 (см. п. 1.4.2). Для ведущего ПКП, на который поступают запросы IRQ4 и IRQ3 от портов COM1 и COM2, установлен начальный номер 08h (см. рис. 1.6). Таблица векторов прерываний (IDT) располагается в памяти с абсолютного адреса 0. Каждый вектор занимает 4 байта, поэтому запросу IRQ*x* в ведущий ПКП соответствует адрес (*x* + 8)∗4 вектора прерывания.

Например, запросу IRQ0 системного таймера назначен номер 8. Переход при прерывании от этого источника происходит по адресу, который задан в восьмой записи таблицы. Так как записи нумеруются от нуля, и каждая содержит по 4 байта, то восьмая запись находится по адресу 0:8∗4. Младшее слово записи содержит смещение, старшее слово — начальный адрес сегмента. Таким образом, для прерывания от IRQ0 системного таймера сегментный адрес перехода находится в слове по адресу 0:34, а смещение — в слове по адресу 0:32. Аналогично адреса переходов для прерывания IRQ4 от COM1 (IRQ3 от COM2) находятся в IDT по адресам соответственно 0:50 и 0:48 (0:46 и 0:44). Заметьте, что сегментный адрес хранится в старшем из пары адресов IDT, соответствующих одному вектору.

З а м е ч а н и е. В программе встречаются операторы-инструкции, которые не имеют прямых соответствий среди команд процессоров IBM PC-совместимых ПЭВМ. Это команды **mov ds, cs** и **push/pop** для группы регистров. Это встроенные макрокоманды ассемблера A86, которые транслируются в группу команд.

Выполните трансляцию программы **i_send.8** и запустите ее на выполнение. По мельканию символа в левом верхнем углу экрана убедитесь, что вход в прерывание и выход из него происходит. Подтверждением факта входа в прерывание является также появление сигналов в линии связи, которые можно наблюдать на экране осциллографа.

Измените программу так, чтобы прерывание вызывалось другой причиной (по заданию преподавателя). Возможны следующие причины прерываний: ошибка переполнения приемника, готовность приемника или «обрыв линии».

Для получения запроса по ошибке переполнения приемника нужно повторить в режиме прерываний опыт по получению такой ошибки (как при исследовании режима опроса готовности). Для разрешения прерываний по ошибке в соответствующем регистре адаптера нужно установить бит 2, остальные биты - обнулить. Требуется получить всего лишь одно прерывание по ошибке. В подпрограмме обработки прерывания необходимо обнулить счетчик count и запретить дальнейшие прерывания.

Другой модификацией этого опыта может быть получение ошибки переполнения при условии, что источником данных является внешнее устройство, например, мышь, подключенная к COM-порту.

Для получения запроса по готовности приемника нужно на вход приемника подавать сигналы в соответствии с принятым для последовательного порта протоколом. В условиях лаборатории это можно сделать, например, с помощью мыши, если она подключена к COM-порту. Для разрешения прерываний по готовности приемника в регистре разрешения прерываний адаптера нужно установить бит 0, остальные биты — обнулить. Чтобы программа «не висела», в обработчике прерывания нужно выполнять чтение регистра данных. Рекомендуется в начале программы выполнить профилактическое чтение регистров данных и состояния линии.

З а м е ч а н и е. Для правильной работы мыши на нее нужно подать питание. Это делается посылкой кода 0bh в регистр управления модемом, что и делается в приведенной программе **i_send.**

Измерение интервалов времени

При определении временных характеристик работы относительно медленных устройств рекомендуется пользоваться подпрограммами из модуля **pztimer.8.** Эти подпрограммы предназначены для измерения интервалов времени в пределах 55 мс с погрешностью -1...+1 мкс. Для этого используется микросхема 8253/8254 трехканального таймера.

В начале измерения программа переводит канал 0 системного таймера в режим однократного счета, загружает максимальное значение в счетчик этого канала и запускает счет. Эта операция выполняется обращением к подпрограмме **zTimerOn.**

После этого программа должна перейти к отработке фрагмента, время выполнения которого измеряется. Пока идет отработка этого фрагмента, значение в счетчике канала 0 уменьшается с периодом приблизительно 820 нс (частота 1193 кГц). Исследуемый фрагмент должен быть выполнен до того, как счетчик обнулится, т.е. не более чем за 55 мс. После завершения исследуемого фрагмента должно быть считано текущее значение счетчика канала 0. Эта операция выполняется вызовом подпрограммы **zTimerOff.**

Вывод результата выполняется подпрограммой **zTimerReport**. Если между вызовами **zTimerOn** и **zTimerOff** прошло более 55 мс, **zTimerReport** сообщит о переполнении таймера.

Набор подпрограмм для измерения временных интервалов содержится в файле **pztimer.8**. Этот файл должен быть транслирован вместе с исследуемой программой. Скопируйте pztimer.8 в свой каталог. В конце текста исследуемой программы запишите:

```
include pztimer.8
```

В качестве примера использования модуля **pztimer.8** для определения текущей настройки скорости последовательного канала связи выполним измерение интервала времени между запуском и завершением передачи, т.е. длительности одной посылки.

```
   base    equ  03f8          ; базовый адрес COM1
   buf     equ  base          ; буферный регистр
                              ; приемника/передатчика
   state   equ  base+5        ; регистр состояния
   call    zTimerOn
   mov     dx, buf            ; запуск передачи
   out     dx, al             ; что передаем - не имеет
                              ; значения
   mov     dx, state

m1:
   in      al, dx
   test    al, bit 6          ; передача закончена?
   jz      m1

   call    zTimerOff
   call    zTimerReport
   ret

   include pztimer.8
```

Запись в регистр данных передатчика (порт по смещению 0) адаптера канала последовательной связи автоматически запускает передачу записанного байта. О завершении передачи сигнализирует единица в бите 6 регистра состояния линии, который доступен через порт по смещению 5.

Исходный текст находится в файле **rate.8.** Транслируйте эту программу и выполните ее. Полученный результат — это время, затраченное на передачу от 7 до 11 бит (в зависимости от настройки адаптера).

Вычислите, с учетом этой неопределенности, в каких границах находится настройка скорости передачи COM1 на вашем компьютере.

2.2.5. Содержание отчета

В отчете приведите тексты программ, которые подвергались модификации в процессе выполнения работы. Сопроводите их комментариями, приведите ответы на вопросы из текста п. 2.2.4.

Укажите фрагмент программы **`i_send`**, во время выполнения которого поступает сигнал запроса на прерывание.

В выводах дайте сравнительную характеристику методов программного управления периферийными устройствами (режимы опроса готовности и прерывания).

2.3. Часы реального времени

2.3.1. Цель работы

Целью работы является изучение функциональных возможностей часов реального времени IBM PC AT и приобретение навыков их программирования.

2.3.2. Общие положения

Часы реального времени обеспечивают непрерывный отсчет времени суток и даты. При включении системного питания программа инициализации ПЭВМ считывает показания часов и устанавливает по ним счетчик системного времени в основной, энергозависимой памяти. До выключения ПЭВМ в этом счетчике регистрируется ход времени — по прерываниям от микросхемы системного таймера. При выключении ПЭВМ часы реального времени переходят на альтернативный источник питания: батарею или аккумулятор.

В IBM PC подсистема часов реального времени RTC (Real Time Clock) реализована на базе микросхемы Motorola MC146818. Собственно подсистема RTC входит в состав этой микросхемы. Она обеспечивает непрерывный счет времени суток и даты, а также предоставляет возможность включения аппаратных прерываний трех типов: в заданное время, по изменению показаний часов, и с заданной частотой.

Микросхема MC146818 содержит 64 байтовых регистра (CMOS). Первые 14 регистров используются RTC: в них хранятся текущие значения времени суток и даты, а также данные состояния и настройки (табл. 2.1).

Таблица 2.1

Основные регистры CMOS/RTC

Адрес	Назначение
00	Текущие секунды
01	Секунды будильника
02	Текущие минуты
03	Минуты будильника
04	Текущие часы
05	Часы будильника
06	День недели (1 = воскресенье)
07	День месяца
08	Месяц
09	Год (последние две цифры)
0a	Регистр A
0b	Регистр B
0c	Регистр C

Первые десять регистров (адреса 00…09) образуют группу регистров часов/календаря. Последние три регистра образуют группу регистров состояния/управления. В остальных 54 байтах CMOS хранятся сведения о конфигурации аппаратных средств компьютера. При начальной загрузке компьютера эти данные определяют настройку BIOS. В конфигурацию входят сведения о типах и количестве накопителей на магнитных дисках, о типе видеоадаптера, о количестве последовательных и параллельных портов и т.д. Область данных конфигурации не влияет на работу подсистемы RTC и не зависит от ее работы.

Регистры часов/календаря

Все регистры часов/календаря доступны для чтения и записи. Доступ к этим данным ограничен: каждую секунду происходит цикл корректировки, в ходе которого данные не определены.

Регистры состояния

Все регистры доступны для чтения. Разряды и поля (группы смежных разрядов), доступные для записи, выделены курсивом.

Регистр A

7	6	5	4	3	2	1	0
UIP	***DV***			***RS***			

Разряд 7 — флаг UIP (Update In Progress). Единица означает, что корректировка происходит или должна начаться, т.е. часы/календарь сейчас недоступны. Нулевое значение флага UIP говорит о том, что цикл корректировки начнется не раньше, чем через 244 мкс. При установке бита SET регистра B в единицу (запрет корректировки), UIP автоматически сбрасывается в ноль.

Разряды 6…4 — биты ***DV*** (DiVisor) делителя, определяющего темп работы RTC. При инициализации ПЭВМ биты делителя установлены 010. Запись других значений приведет к изменению темпа часов. Запись значения 000 делителя при сброшенном разряде SET регистра B выключает часы.

Разряды 3…0 — биты ***RS*** (Rate Selection) задают темп периодических прерываний, которые описаны ниже. Значение 0110, устанавливаемое при начальной загрузке ПЭВМ, соответствует частоте 1024 Гц.

Регистр B

7	6	5	4	3	2	1	0
SET	***PIE***	***AIE***	***UIE***	***SQWE***	***DM***	***24/12***	***DSE***

Разряд 7 (***SET***) — при установке в единицу корректировка запрещена, и программа может записывать данные часов/календаря/будильника. При установке SET в единицу автоматически обнуляется бит UIP регистра A. Обнуление бита SET разрешает циклы корректировки.

Разряд 6 (***PIE***) — (Periodic Interrupt Enable). Установка в единицу разрешает периодическое прерывание с частотой, заданной битами RS регистра состояния A.

Разряд 5 (***AIE***) — (Alarm Interrupt Enable). Установка в единицу разрешает прерывание будильника.

Разряд 4 (***UIE***) — (Update-ended Interrupt Enable). Установка в единицу разрешает прерывание по завершению цикла корректировки. При установке бита SET в единицу, бит UIE автоматически сбрасывается, и не восстанавливается после сброса SET.

Разряд 3 (***SQWE***) — (Square Wave Enable). Установка в единицу разрешает вывод меандра через контакт SQW микросхемы часов реального времени с частотой, определяемой битами RS регистра состояния A. В современных ПЭВМ IBM PC контакт SQW не используется.

Разряд 2 (***DM***) — (Data Mode) формат представления данных часов/календаря: 1 — двоичный формат, 0 — формат BCD (Binari Coded Decimal — двоично-десятичный формат [13]). DM показывает, в каком формате производится *корректировка* данных. При начальной загрузке BIOS устанавливает DM равным нулю.

Разряд 1 (***24/12***) — задает диапазон часов: 0 — 12 часов, 1 — 24 часа. При начальной загрузке устанавливается в единицу.

Разряд 0 (***DSE***) — (Daylight Savings Enable) флаг летнего времени. Установка DSE в единицу задает две дополнительные корректировки в год: в последнее воскресенье апреля (01:59:59 → 03:00:00) и в последнее воскресенье октября (01:59:59 → 01:00:00). При начальной загрузке BIOS сбрасывает DSE.

Регистр C

7	6	5	4	3	2	1	0
IRQF	**PF**	**AF**	**UF**				

Разряд 7 (IRQF) — (Interrupt Request Flag) флаг запроса прерывания. Единица означает, что произошло одно из прерываний, разрешенных в регистре состояния B, т.е. когда выполняется хотя бы одно из условий: PF = PIE = 1, AF = AIE = 1, UF = UIE = 1

Разряд 6 (PF) — (Periodic interrupt Flag), единица означает, что произошло периодическое прерывание.

Разряд 5 (AF) — (Alarm interrupt Flag), единица означает, что произошло прерывание от будильника.

Разряд 4 (UF) — (Update-ended interrupt Flag), единица означает, что произошло прерывание по завершению цикла корректировки.

Разряды 0...3 — читаются как нулевые, запись в них невозможна.

З а м е ч а н и е. Флаги PF, AF, UF регистра C устанавливаются независимо от значений соответствующих разрядов (PIE, AIE, UIE) регистра состояния B. Напротив, флаг IRQF устанавливается только тогда, когда установлен бит разрешения, соответствующий запросу на прерывание. Все четыре флага очищаются при чтении регистра состояния C. Пока флаги не сброшены, все последующие события не вызывают изменений в состоянии регистра C, т.е. повторные запросы на прерывание не могут быть сформированы (см. также замечания на с. 81).

2.3.3. Программирование RTC

Работа с RTC возможна как за счет использования функций BIOS, так и непосредственно через порты ввода/вывода. Использование функций BIOS предпочтительно, если требуется доступ к данным часов/календаря.

Функции BIOS для обслуживания RTC

Для обслуживания RTC имеется ряд функций прерывания 1Ah BIOS. В табл. 2.2 приведены основные из них; опущены функции чтения/записи даты.

Таблица 2.2

Основные функции BIOS для обслуживания RTC

Функция	Назначение	Входные параметры	Результат
02h	Чтение часов	AH = 02h	CF = 1 - ошибка CH = часы (BCD) CL = минуты (BCD) DH = секунды (BCD) DL = флаг летнего времени
03h	Установка часов	AH = 03h CH = часы (BCD), CL = минуты (BCD) DH = секунды (BCD) DL = флаг летнего времени	
06h	Установка будиль-ника	AH = 06h CH = часы (BCD), CL = минуты (BCD) DH = секунды (BCD) Замечание: предварительно установите адрес процедуры обработки прерывания 4Ah	CF = 1 - ошибка
07h	Сброс будиль-ника	AH = 07h Замечание: запрещает прерывание тревоги, установленное функцией 06h. Не забудьте восстановить вектор прерывания 4Ah.	

Непосредственный доступ к RTC

Регистры RTC доступны через порты 70h и 71h. Чтобы прочесть байт из RTC по адресу addr, нужно записать значение addr в порт 70h (**out 70h,addr**), после чего прочитать данные из порта 71h (**in al,71h**). Для записи байта value по адресу addr вновь нужно записать значение addr в порт 70h (**out 70h,addr**), после чего записать значение value в порт 71h (**OUT 71h,value**).

Пример — чтение регистра состояния B:

```
mov    al, 0bh
out    70h, al
nop                ; задержка
in    al, 71h
```

Рекомендуется программировать непосредственное обращение к данным RTC только с использованием макрокоманд, контролирующих превышение допустимого значения адреса (с адреса 10h начинается область данных конфигурации, запись в нее недопустима), например:

```
rdRtc    macro
##if #v1 GT 0c
     ?ERROR: addr > 0Ch
##endif
     mov al, #1
     out 070, al
     nop
     in  al, 071
#em
```

```
     wrRtc    macro
##if #v1 GT 0c
     ?ERROR: addr > 0Ch
##endif
     push ax
     mov al, #1
     out 070, al
     pop ax
     out 071, al
#em
```

Примеры вызовов:

```
rdRtc 0c        ; чтение регистра C в al
wrRtc 0b        ; запись из al в регистр B
wrRtc 010       ; ? ошибка, выявляемая на этапе
                ; трансляции (адрес больше 0c)
```

Данные часов/календаря/будильника не всегда доступны программе. Раз в секунду эти десять регистров переключаются на корректирующее устройство для увеличения показаний и для сравнения данных будильника с текущим временем. Чтение регистров во время корректировки дает неопределенный результат. Для извещения программы о завершении корректировки может быть использовано соответствующее прерывание.

Определить доступность часов можно также по значению флага UIP регистра состояния A.

Перед записью в регистры часов/календаря разряд SET регистра состояния B должен быть установлен в единицу, чтобы предотвратить изменение данных часов/календаря в процессе записи. Во всех ячейках RTC должны быть использованы данные одного формата (двоичного или BCD), который должен быть отражен в разряде DM регистра B.

Использование аппаратных прерываний от RTC

RTC содержит три независимых источника прерываний:

- будильник (прерывание тревоги),
- генератор периодических прерываний,
- собственно RTC — прерывания по завершению корректировки текущего времени.

Запрос на прерывание от будильника случается при совпадении содержимого всех трех регистров будильника с содержанием соответствующих регистров текущего времени (табл. 2.1). Периодичность этих прерываний является программируемой. Она может быть выбрана в диапазоне от одного раза в секунду до одного раза в сутки.

Частота периодических прерываний может быть задана в диапазоне от 2 до 8192 Гц.

Для извещения программы о завершении цикла корректировки может быть использовано прерывание конца корректировки.

Нужный режим прерывания выбирает программа процессора. Три разряда в регистре B разрешают три прерывания. При записи единицы в разряд разрешения прерывания запрос (IRQ8) будет выработан при наступлении соответствующего события. Напомним, что прерывание состоится, если кроме разрешения прерывания в регистре B часов реального времени, оно будет разрешено (не замаскировано) также в ПКП (см. пп. 1.4 и 2.2).

RTC подключен к линии прерываний IRQ8 AT (вектор прерывания 70h). Чтобы использовать прерывания от RTC, установите вектор 70h на процедуру обработки прерывания, запрограммируйте регистры состояния RTC и обнулите бит 0 в регистре IMR второго контроллера прерываний (порт A1h). Можно разрешить более одного типа прерываний. В этом случае обработчик должен проверять, какое прерывание произошло, считывая регистр состояния C. В любом случае ваш обработчик должен читать регистр состояния C, чтобы очистить флаги причин прерывания и, тем самым, разрешить прерывания снова.

Прерывание по завершению цикла корректировки генерируется после каждого обновления часов, через 1 с. Для разрешения установите разряд 4 (UIE) в регистре состояния B.

Прерывание от будильника генерируется в заданное время. Чтобы использовать это прерывание, установите секунды (адрес 01h), минуты (адрес 03h) и часы (адрес 05h) будильника, затем установите разряд 5 (AIE) в регистре состояния B. «Безразличное» значение 11XXXXXX задает любое время, например, FF:FF:00 задает прерывание каждую минуту, FF:00:FF - каждую секунду в течение первой минуты каждого часа.

Периодическое прерывание генерируется с частотой, заданной в пределах от 2 до 8192 Гц. Для настройки частоты установите разряды поля RS (Rate Select) в регистре состояния A в соответствии с табл. 2.3 (выделенная строка — настройка BIOS при начальной загрузке). Затем установите разряд 6 (PIE) в регистре состояния B.

З а м е ч а н и я:

- Флаги PF, AF и UF регистра C устанавливаются *независимо* от значений соответствующих разрядов (PIE, AIE, UIE) регистра состояния B. Фактически PF, AF, UF являются разрядами состояния (готовности), в которых фиксируется наступление временного события, т.е. программа может работать с ними в режиме *опроса готовности*, без прерываний. События установки этих битов готовности могут (но не обязаны) использоваться как источники аппаратных прерываний. Единицы в этих битах означает, что событие произошло **ХОТЯ БЫ ОДИН РАЗ** после последнего считывания регистра C (при считывании C эти биты сбрасываются), но возможно и большее число раз! Как правило, события PF, UF при запуске программы уже произошли и не раз, поэтому в начале работы с этими битами нужно их сбросить чтением регистра C.
- Все четыре флага очищаются при чтении регистра состояния C. Пока флаги не сброшены, все последующие прерывания фактически запрещены, т.е. пользовательская процедура обработки прерывания **ДОЛЖНА** читать регистр состояния C.

Т а б л и ц а 2.3

Частота периодических прерываний в зависимости от значения RS

RS	Частота (Гц)	Период
0000	0	-
0001	256	3.90625 мс
0010	128	7.8125 мс
0011	8192	122.070 мкс

Продолжение табл. 2.3

0100	4096	244.141 мкс
0101	2048	488.281 мкс
0110	***1024***	***976.562 мкс***
0111	512	1.93125 мс
1000	256	3.90625 мс
1001	128	7.8125 мс
1010	64	15.625 мс
1011	32	31.25 мс
1100	16	62.50 мс
1101	8	125.0 мс
1110	4	250.0 мс
1111	2	500.0 мс

2.3.4. Программа работы

Доступ к данным часов осуществляется при помощи редактора портов **`rport`**.

- Прочитайте напрямую регистры текущего времени. Выясните, изменяется ли значение в порте 070 после чтения данных из порта 071. Если да, то запись адреса в порт 070 требуется при каждом обращении. Если нет, проследите за изменениями секунд, читая из порта 071 без повторных записей 0 в порт 070.
- Прочитайте регистры состояния, охарактеризуйте состояние. Проследите за изменениями битов PF, UF при повторных обращениях к регистру C. Почему бит UF сбрасывается после чтения, а бит PF - нет? Почитайте регистр C клавишей «*». Объясните изменение результатов чтения. Уменьшите частоту периодических событий до 2 Гц (см. описание регистра A) и повторите опыт.
- Установите бит SET в регистре B и немедленно засеките время при помощи команды **`time`** операционной системы. Проверьте, изменяются ли данные текущего времени RTC. После паузы в несколько секунд восстановите исходное значение бита SET. Как изменились данные RTC? Повторите опыт со следующим отличием: после установки SET запишите что-нибудь в регистр секунд.
- Сформулируйте алгоритм работы RTC при установке/сбросе бита SET.

- По указанию преподавателя выполните несколько заданий из следующего перечня.

1. Напишите и проверьте программу для чтения текущего времени из RTC двумя способами — средствами BIOS и прямым обращением к регистрам RTC.

2. Напишите и проверьте программу, устанавливающую прерывания будильника раз в секунду средствами BIOS.

3. Напишите и проверьте программу для обработки событий (тип события и параметры задает преподаватель), в режиме опроса готовности и по прерываниям.

4. Разработайте программу, подсчитывающую события UF. При каждом событии выводите на экран символ «*», после 10 событий программа завершается. Первый символ выводится сразу после запуска, так как UF давно уже установлен (секунда прошла много раз с момента последнего обращения к регистру C). Добавьте действия по сбросу события в начале программы. Несколько раз запустите программу. Почему первый символ появляется через случайный промежуток времени?

5. Разработайте аналогичную программу, подсчитывающую события PF на частоте, заданной преподавателем. Работа программы должна сводиться к задержке в 5 с, за счет слежения за событиями PF. Не рекомендуется выводить на экран символы по этому событию, т.к. задержка на прокрутку экрана может привести к пропуску высокочастотных событий PF.

- Выполните индивидуальное задание из следующего списка по указанию преподавателя.

Варианты индивидуальных заданий:

1. Разработайте программу, которая насчитывает 3 с в начале ближайшей минуты текущего часа. Требуется установить данные будильника и трижды дождаться события AF.

2. Разработайте вариант программы подсчета событий UF или PF, использующий аппаратные прерывания.

3. Оцените равномерность наступления событий PF. Для этого дополните программу подсчета событий PF: в тело цикла ожидания PF включите подсчет числа итераций, после завершения полученный результат сохраняйте в массиве. После завершения всех циклов, отбросив первый результат, найдите max и min и оцените относительную погрешность (среднее оцените как (max + min)/2).

4. Определите, сколько раз возникает событие PF между двумя событиями UF. В некоторых реализациях RTC это число (на частоте 1024 Гц) заметно отличается от 1024.

2.3.5. Содержание отчета

В отчете приведите формулировки заданий и тексты программ с комментариями и объяснением результатов.

В выводах охарактеризуйте исследованные особенности RTC.

2.4. Межмодульный обмен информацией по шине ISA

2.4.1. Цели работы

Целью работы является формирование представления о физических процессах, происходящих при информационном взаимодействии устройств на системной шине. В качестве примера выбрана шина ISA, которая является стандартной для подключения устройств ввода/вывода (УВВ) широкой номенклатуры, в том числе плат сопряжения с объектом в управляющих и измерительных системах.

2.4.2 Общие положения

Краткие сведения о шине ISA и о структуре контроллера периферийного устройства приведены в пп. 1.3 и 1.5. В данной работе предлагается с помощью осциллографа исследовать временные диаграммы при выполнении различных циклов. В рассматриваемой шине конкретный вид и параметры временных диаграмм зависят от ряда условий:

- тип цикла (п.1.3.2),
- типа ресурса доступа (память или УВВ),
- разрядность ресурса доступа или передаваемой информации (8 или 16 бит),
- быстродействие ресурса, с которым ведется обмен.

Поэтому для основных циклов даются временные диаграммы для нормального и удлиненного циклов (для доступа к ресурсу ещё есть цикл с 0 тактов ожидания), обращения к памяти или УВВ.

На рис. 1.3 приведены временные диаграммы некоторых циклов шины ISA, которые можно исследовать на лабораторном стенде. Этими иллюстрациями охвачены варианты нормального и удлиненного циклов чтения и записи УВВ (8-разрядного или 16-разрядного). В табл. 1.1 приведены временные параметры диаграмм. Следует обратить внимание, что в разных режимах работы шины один и тот же параметр принимает разные значения. В процессе выполнения работы обратите внимание на соответствие измеренных временных задержек параметрам из табл. 1.1.

2.4.3. Описание лабораторного стенда

Основу стенда составляет IBM PC-совместимая персональная ЭВМ, имеющая в своем составе плату (ISA-компонент), предназначенную для наблюдения процессов на шине ISA в различных режимах её работы. Назовем эту плату платой стенда и обозначим её ПС. Ее функциональная схема приведена на рис. 2.1. Для создания возможности осциллографирования сигналов шины ПС соединена с выносным пультом, на который выведены основные сигналы. Информацию в ПС можно вводить вручную с помощью микропереключателей, установленных на лицевой панели ПЭВМ. На той же панели расположены светодиоды, на которые выведено состояние регистра данных.

Функциональная схема ПС в целом совпадает со схемой, описанной в п.1.3. Выходы «Данные в ПУ» по схеме интерфейсной части подключены к приемному регистру, выходы которого подключены к светодиодному индикатору. Входы «Данные из ПУ» через буферные усилители подключены к линиям данных шины.

В адресном пространстве устройств ввода-вывода для ПС выделены четыре адреса, представленные в табл. 2.4.

При записи по адресу 360h данные записываются в регистр данных и индицируются с помощью светодиодов. При чтении по тому же адресу в шину и далее по адресу приемника информации направляется код, набранный на микропереключателях.

Таблица 2.4

Регистры платы стенда

Адрес	Назначение регистра
360h	Регистр данных для записи
361h	Сброс
362h	Свободен
363h	Регистр режимов ПС

Обращение по адресу 361h вызывает сброс сигналов запроса на прерывание (IRQ) или на прямой доступ к памяти (DRQ). При этом также происходит начальная установка схемы формирования сигнала IO CH RDY. Поэтому при исследовании работы шины в этих режимах, когда необходимо позаботиться о своевременном сбросе соответствующих схем, нужно в программный цикл включить обращение по этому адресу.

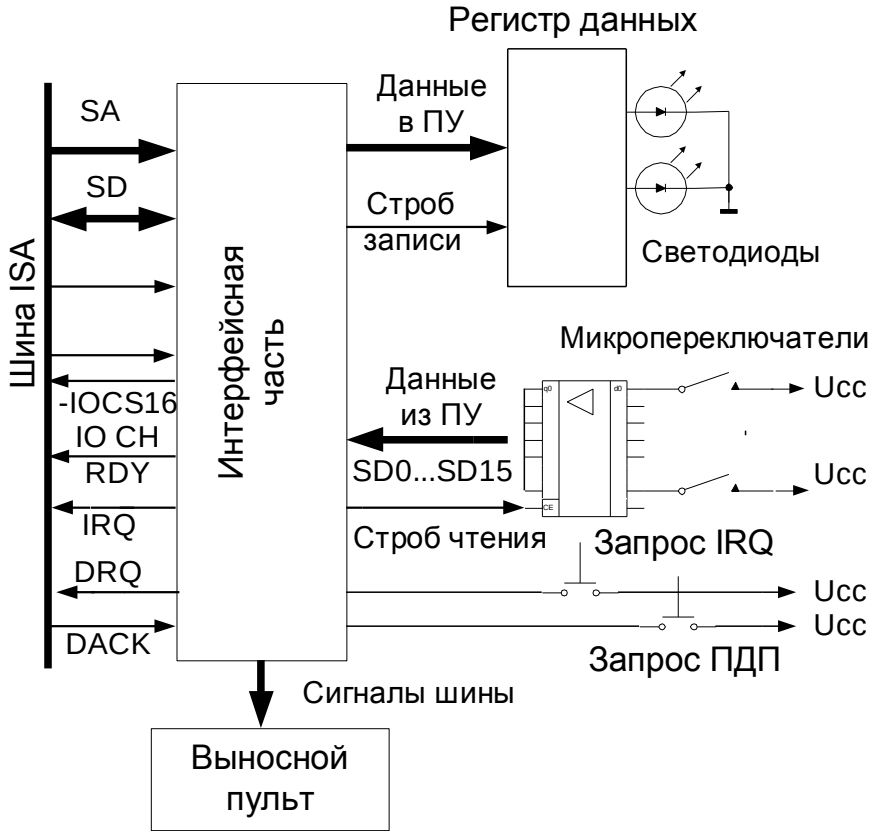

Рис. 2.1. Функциональная схема платы стенда

По адресу 363h производится обращение к регистру режимов ПС, который служит для настройки на избранный цикл обмена. Настройка производится путем записи в регистр режимов управляющего слова в соответствии с табл. 2.5. Это можно сделать в программе, а также с помощью редактора портов report (см. п.2.1)

Таблица 2.5

Назначение разрядов регистров режимов

Номер разряда	Назначение разряда
0	Цикл ввода/вывода
1	Цикл обработки прерываний
2	Удлиненный цикл
3	Цикл ПДП
4	Цикл программного доступа к памяти
5, 6	Свободны
7	Обмен 8/16 бит (единица — 16 бит, нуль — 8 бит)

При формировании управляющего слова иметь в виду следующее:

* должен быть выбран один, но только один из возможных режимов, т.е. в одном из разрядов выбора режима (0, 1, 3, 4) должна быть единица,
* не следует устанавливать признаки (разряды 2 и 7), не выбрав режима,
* для установки режима следует написать отдельную программу (особенность макета).

Выбранный режим индицируется на светодиодном табло.

2.4.4. Программа работы

* Для заданного режима написать и отладить программу циклического ввода-вывода информации (чтение, а затем запись слова или байта, с удлинением цикла или без такового).
* Запустить программу на выполнение и снять осциллограммы сигналов, участвующих в обмене. Используя снятые осциллограммы, построить временную диаграмму работы шины в данном режиме. При этом не следует изображать наблюдаемую диаграмму «как в учебнике» и пренебрегать искажениями и помехами, которые неизбежны в любой схеме. Снять осциллограмму — это, кроме зарисовки сигналов, означает ещё и выполнение измерений (амплитуды, длительности, задержки и т.п.).

При выполнении этого пункта особое внимание следует обратить на синхронизацию осциллографа, т.е. выбор сигнала для запуска его развертки. Иногда для наблюдения цикла обмена удобно запустить такой цикл дважды, чтобы получить на экране полноценное изображение. Для запуска развертки нужно выбрать такой сигнал, который, во-первых, предшествует временному интервалу, который собираемся наблюдать, а во-вторых, сигнал этот должен давать уверенность, что это именно тот самый интервал, т.е. желательно, чтобы в программном цикле он встречался один-два раза и не более.

* Оформить отчет по проделанной работе.

2.4.5. Содержание отчета

Примерный план отчета может выглядеть следующим образом:

- формулировка задания,
- структурная схема стенда или платы сопряжения в соответствии с заданием,
- тексты программ, реализующих задание,
- осциллограммы наблюдавшихся процессов, выполненные в соответствии с указаниями п. 2.4.4,

- таблицу, в которой привести наиболее важные временные параметры исследованного цикла. Для сравнения включить в таблицу нормативные значения из табл.1.1,
- выводы по работе, в которых отразить характерные особенности изученных процессов и архитектуры шины, дать оценку шины с точки зрения возможности применения её для различных целей, попытаться дать объяснение амплитудным и временным искажениям (выбросы, «звон», завал фронтов, дрожание изображения). Обратить внимание на амплитуду сигналов, попытаться объяснить несовпадение измеренных амплитуд с «ожидаемыми».

2.5. Проектирование платы расширения

2.5.1. Цель работы

Целью работы является получение навыков проектирования основных узлов интерфейсной части периферийных плат ЭВМ с использованием машинных методов и современной элементной базы.

2.5.2. Основные положения

Подход к проектированию плат расширения изложен в п. 1.5. В соответствии с ним такая плата может быть представлена, как показано на рис. 1.8. Более подробная функциональная схема интерфейсной части платы определяется протоколом обмена информацией на шине, к которой проектируемая плата должна быть подключена. Пример более детальной схемы интерфейсной части платы расширения приведен на рис. 1.9. Дальнейшая детализация схемы выполняется в соответствии с конкретным типом цикла на шине, который должна поддерживать проектируемая плата.

Шина Q-bus (отечественный аналог — МПИ — межмодульный параллельный интерфейс) широко использовалась в управляющей технике прошлых лет, а в настоящее время в значительной степени исчезла из употребления, за исключением специальных разработок на соответствующей элементной базе (микропроцессорный комплект К1806). Есть примеры использования периферийных плат для шины Q-bus в комплексах на базе IBM/PC-совместимых ЭВМ. Для такого комплекса разрабатываются специальные контроллеры шины, обеспечивающие правильное взаимодействие компонентов.

Шина ISA использовалась в первых персональных ЭВМ IBM PC, начиная с 1982 г. и сохранилась в современных устройствах для подключения контроллеров или плат расширения широкой номенклатуры, в том числе индустриальных (имеется промышленный аналог шины ISA — PC 104).

Шина PCI является примером современной высокопроизводительной шины, получившей широкое распространение в современных микроЭВМ как общего, так и специального назначения, в том числе в системах управления промышленным оборудованием (имеется промышленный аналог шины PCI, который называется Compact PCI).

Таким образом, для разработки интерфейсной части нестандартной платы ЭВМ необходимо изучить ограничения, налагаемые используемым интерфейсом. В документации на интерфейс описаны все электрические линии, управляющие сигналы, их функции в различных режимах работы шины, причинно-следственные отношения между сигналами, их временные и электрические характеристики, конкретные типы соединителей (разъемы), конструктивные требования к платам и т.п. При выполнении настоящей работы студентам предлагается разработать принципиальную схему только интерфейсной части периферийной платы. В качестве исходных данных к работе задается временная диаграмма работа шины в конкретном режиме.

Варианты заданий включают различные режимы работы шин ISA и МПИ. Разработка платы расширения для шины PCI или даже отдельной функции такой платы вследствие сложности протокола шины требует более значительного времени и поэтому в рамках лабораторной работы не предлагается.

Временные диаграммы и основные временные соотношения для шины ISA приведены в п. 1.3.

Ниже приводятся эти же данные для шины Q-Bus.

Межмодульный параллельный интерфейс (Q-bus)

На рис. 2.2 приведена временная диаграмма адресного обмена (операции ЧТЕНИЕ и ЗАПИСЬ) по шине МПИ ГОСТ 26785.57-88 (аналог шины Q-bus). Стрелками показаны причинно-следственные отношения между сигналами. В табл. 2.6 приведено краткое описание сигналов, используемых в этом режиме. В табл. 2.7 указаны временные параметры диаграммы.

Отметим некоторые особенности интерфейса.

Ведущее устройство называется активным, а ведомое — пассивным. В табл. 2.6 указано направление передачи сигнала (А > П — от активного к пассивному, П > А — в противоположном направлении)

Шина адреса/данных является совмещенной (мультиплексированной), т.е. на линиях АД последовательно во времени устанавливается сначала адрес, а затем данные. Адрес всегда и данные при записи направляются от активного устройства к пассивному. В цикле чтения данные направляются от пассивного устройства к активному.

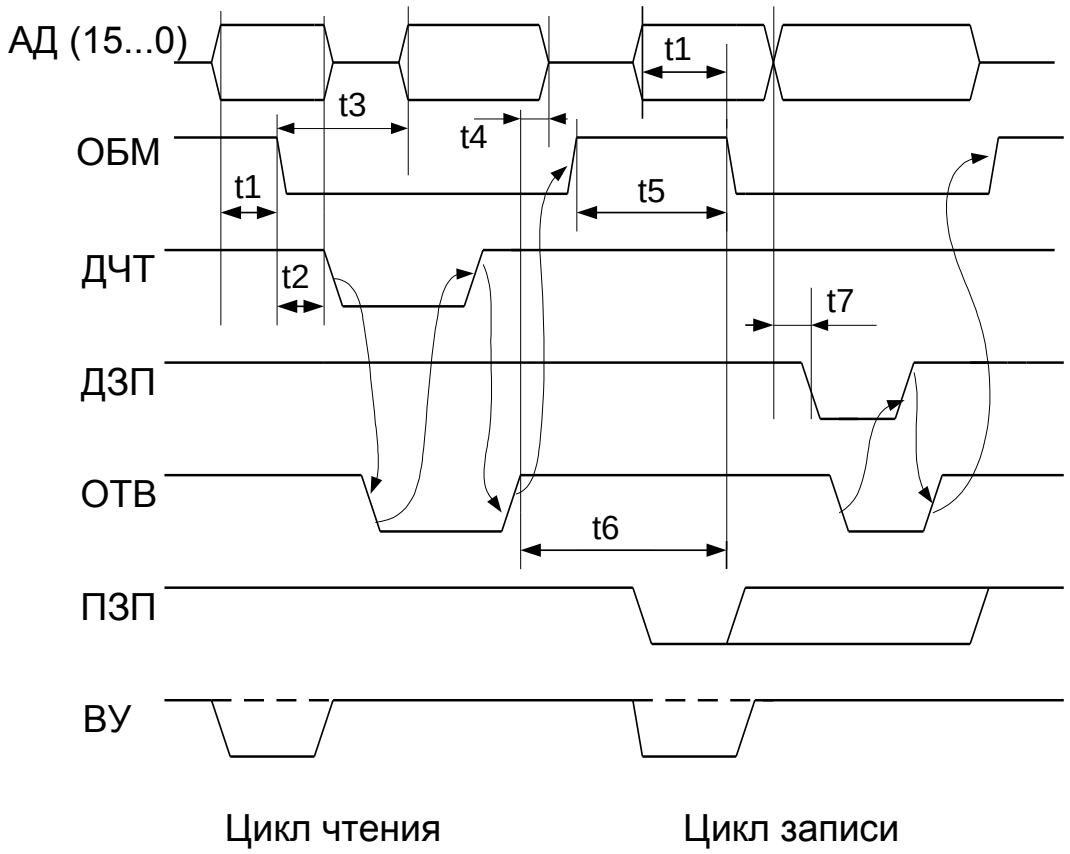

Рис. 2.2. Временные диаграммы циклов адресного обмена (чтение и запись) МПИ

Отрицательным фронтом сигнала ОБМ контроллер ПУ должен зафиксировать необходимые для дальнейшей работы разряды адреса. Сигналы адреса, ОБМ, ДЧТ (ДЗП) формируются синхронно, а дальнейшая работа зависит от поступления сигнала ОТВ. Отсутствие сигнала ОТВ в течение более 12…15 мкс расценивается активным устройством как сбой (зависание в канале). Таким образом, формирование сигнала ОТВ контроллером ПУ является обязательным

Адресное пространство ввода/вывода располагается в 8К старших адресах общего адресного пространства. При выполнении данной лабораторной работы можно считать, что адреса пространства ввода/вывода начинаются с восьмеричного адреса 160000. Сигнал ВУ формируется активным устройством всегда, когда идет обращение к УВВ, поэтому для упрощения схемы можно анализ трех старших разрядов адреса заменить анализом сигнала ВУ.

Таблица 2.6

Сигналы шины МПИ (Q-bus)

Обозначение сигнала		Назначение линии интерфейса	Направление передачи	Примечание
МПИ	Q-bus			
АД(15-0)	AD(15-0)	Двунаправленная шина адреса-данных		
ОБМ	SYNC	Синхронизация обмена	А > П	Отрицательный фронт сигнализирует о наличии адреса на шине АД. Удерживается до конца цикла
ДЧТ	DIN	Чтение данных	А > П	Строб чтения
ДЗП	DOUT	Запись данных	А > П	Строб записи
ВУ	BS	Выбор устройства	А > П	Обращение к ПУ (свидетельствует об обращении к адресному пространству ввода/вывода)
ПЗП	WTBT	Запись байта	А > П	Формируется в цикле записи. Если записывается слово, то сигнал совпадает по времени с адресом. Если записывается байт, то сигнал удерживается до окончания сигнала ДЗП (DOUT)
ОТВ	RPLY	Ответ устройства	П > А	Свидетельствует о наличии ПУ, к которому обращается программа и о распознавании им адреса обращения

Таблица 2.7

Временные соотношения цикла на шине Q-bus

Обозначение параметра	Описание параметра	Значение параметра (нс)	
		min	max
t1	Опережение установки адреса по отношению ко фронту сигнала ОБМ	150	
t2	Удержание адреса на шине АД после установки сигнала ОБМ	100	
t3	Задержка установки данных на шине АД пассивным устройством относительно фронта сигнала ОТВ (при чтении информации)		200
t4	Задержка сброса данных на шине АД после спада сигнала ОТВ		100
t5	Пауза между спадом и новым фронтом сигнала ОБМ	200	
t6	Пауза между спадом сигнала ОТВ и фронтом нового сигнала ОБМ	300	
t7	Задержка фронта сигнала ДЗП относительно момента замены адреса данными на шине АД в цикле записи	100	

2.5.3. Инструментальные средства проектирования

При проектировании подобных устройств обычно используются интерфейсные микросхемы, среди которых имеются БИС, ориентированные на применение в конкретных шинах. Однако появление микросхем гибкой логики со средствами автоматизации проектирования делает доступным и целесообразным использование именно их при реализации интерфейсной части контроллеров. При этом повышается надёжность и гибкость разрабатываемых устройств, снижается время, необходимое на их разработку и отладку.

Данную работу предлагается выполнить с помощью системы автоматизированного проектирования QUARTUS (или MAX+PLUS II). Эта

САПР разработана фирмой Altera непосредственно для разработки сложных электронных устройств на основе микросхем программируемой логики. Она позволяет проектировать и отлаживать такие устройства с использованием стандартных ПЭВМ, используя дружественный пользовательский интерфейс, а затем производить программирование микросхем гибкой логики, используя большинство из известных на данный момент протоколов.

2.5.4. Описание лабораторной установки

Лабораторная установка представляет собой персональный компьютер, оснащенный соответствующей САПР. Предполагается, что основные навыки работы с данными программами были получены студентами при предыдущем обучении.

2.5.5. Программа работы

- Перед началом работы получить задание, в котором указываются следующие характеристики проектируемой платы:

— тип шины — ISA или МПИ (Q-Bus),
— разрядность операции обмена,
— число регистров, доступных по чтению и/или записи,
— время замедления цикла обмена (путем формирования соответствующих ответных сигналов),
— возможность работы платы в режиме прерывания.

- По этим данным спроектировать функциональную схему интерфейсной части периферийной платы. При этом считать источником (приемником) информации в режимах чтения (записи) регистр соответствующей разрядности. При реальном проектировании это мог бы быть выходной регистр АЦП (входной регистр ЦАП).
- Перед вводом проекта в САПР желательно изобразить эскиз Вашего проекта на бумаге в графической форме и согласовать его с преподавателем.
- Ввод разработанной схемы в память компьютера может выполняться как в текстовой, так и в графической форме. Желательно использование примитивов из стандартных библиотек, мегафункций Altera, а также использование языков программирования высокого уровня AHDL или VHDL.
- Выполнить компиляцию введённой схемы.
- Выполнить моделирование работы схемы с помощью утилит Simulator и Waveform Editor.
- Оценить работоспособность схемы. Критерием правильности выполненной работы является соответствие между нормативной временной

диаграммой и результатами моделирования на некотором наборе входных сигналов, задаваемых преподавателем.

- Составить отчёт по выполненной работе.

З а м е ч а н и я:

- Спроектированная схема должна представлять собой конечный автомат с памятью. Для студентов привычной является ситуация, когда переходы между состояниями автомата синхронизируются тактовыми импульсами. Выше отмечалось, что ни для шины ISA, ни для Q-bus стандартом не установлена привязка сигналов к тактовым импульсам шины. В проектируемой схеме достаточно для синхронизации переходов состояния использовать стробы чтения/записи. Использование тактовых импульсов шины ISA, таким образом, для этого является излишним, а на шине Q-bus тактовые импульсы отсутствуют. Иное дело, когда требуется сформировать интервал времени (задержку). В этом случае использование тактовых сигналов вполне оправдано.
- При выполнении работы требуется использовать принятые для данной шины ограничения. Например, допускается выбирать только допустимые адреса пространства ввода/вывода. Допустимый диапазон адресов для шины Q-Bus указан в п. 2.5.2. Для адресов шины Q-Bus следует использовать восьмеричную систему счисления.

2.5.6. Содержание отчета

Примерный план отчета:

- постановка задачи (цель, методика выполнения, технические средства),
- исходная временная диаграмма, спроектированная схема, результат её моделирования,
- оценка точности выполнения задания,
- оценка преимуществ использования микросхем гибкой логики при проектировании интерфейсных блоков периферийных устройств.
- выводы по работе в целом, где кратко охарактеризовать методику и результаты проектирования.

2.6. Параллельный порт, интерфейс Centronics и управление принтером

2.6.1. Цели работы

Целью работы является овладение приемами управления периферийными устройствами через параллельный порт и интерфейс Centronics на примере вывода текста в принтер на печать.

2.6.2. Общие положения

В данной работе изучается стандартное подключение принтера через параллельный порт и интерфейс Centronics, кратко описанные в п. 1.6.

Принтер является программно управляемым устройством. В его состав входит достаточно сложная, как правило, микропроцессорная система управления, обеспечивающая инициализацию принтера, его настройку на тот или иной режим работы, выбор шрифтов, прочих параметров (спецификаций) текста, работу по приему данных от ЭВМ, формирование сигналов о своем состоянии и другие функции. Отметим наличие буферной памяти в составе системы управления принтером. Эта память позволяет с большой скоростью принимать данные от ЭВМ и распечатывать их в темпе, диктуемом электромеханикой принтера, что экономит процессорное время.

Настройка принтера осуществляется путем засылки в него кодовых последовательностей в соответствии с описанием устройства. Запись данных в принтер, подключаемый через параллельный интерфейс Centronics, выполняется за счет формирования на его входах временной диаграммы в соответствии с рис. 1.10. Для инициализации принтера требуется, кроме того, сбросить бит инициализации в 0 на заданное время, после чего установить его в обычное состояние 1. В течение всей инициализации бит -SLCT IN должен быть установлен в состояние 1.

Для облегчения работы с принтером при программировании на языке ассемблера в BIOS и MS DOS имеются стандартные функции, вызываемые внутренними (программными) прерываниями.

Прерыванием **INT 017h** вызываются функции BIOS, поддерживающие работу с принтером. Перед вызовом этой функции в dx засылается номер порта LPT, а в регистр ah — номер функции:

0 — посылка данных в принтер,

1 — инициализация порта (при этом в ah возвращается содержимое регистра состояния принтера),

2 — чтение регистра состояния, содержимое которого помещается в ah.

INT 05h вызывает функцию BIOS, распечатывающую экран.

INT 021h вызывает функции DOS. Функция с номером 05h, который предварительно записывается в ah, посылает код из регистра dl в принтер.

Имеются также другие функции, связанные с принтером.

Рассмотрим несколько примеров передачи информации на принтер.

Использование функции BIOS

```
mov al, 'A'      ; Символ, выводимый на печать
mov dx, 0        ; Использовать LPT1
mov ah, 0        ; Функция - «Послать байт в принтер»
int 017h         ; Вызов функции
```

Использование функций MS-DOS

```
mov dl, 'B'          ; Символ, выводимый на печать
mov ah, 5            ; Функция печати
int 021h             ; Вызов функции
```

Другой вариант — объявить вывод на принтер стандартным выходным потоком и использовать функцию 040h прерывания **int 021h**, подобно тому, как это делается при выводе результатов вычислений на экран или в файл. Для этого нужно записать в cx количество символов, выводимых на печать, а в bx записать дескриптор выходного файла (file handle). Для принтера определен дескриптор с номером 4.

Пример:

```
jmp start
txt   db  'Для вывода на печать использовался'
db  'дескриптор файлов'
      db   10, 13        ; Возврат каретки и перевод
; строки
txt_len equ $-txt        ; Объем текста

start:
mov   bx, 4              ; Дескриптор
          ; выходного потока (принтер)
lea   dx, txt
mov   cx, txt_len        ; Объем выводимого текста в
; байтах
mov   ah, 40h            ; Настройка функции DOS
int 21h                  ; Вызов функции DOS
```

Непосредственное низкоуровневое программирование вывода на принтер (в режиме опроса готовности)

```
   date  equ 378h          ; Адреса регистров
   status    equ 378h +1
   contr equ 378h +2

   jmp start

   txt   db 'Этот текст выводится на принтер', 10, 13
         db 'без использования функций MS DOS и BOIS'
         db 10, 13
   txt_len   equ $-txt

start:   lea bx, txt       ; Загрузка параметров текста
         mov cx, txt_len

print:   mov al, [bx]      ; Посылаем символ
         mov dx, date
         out dx, al

         mov al, 01101xb   ; Синхроимпульс (здесь и далее
         mov dx, contr     ; суффикс x перед буквой b,
         out dx, al        ; обозначающей двоичный
         mov al, 01100xb   ; операнд, устраняет неодно-
         out dx, al        ; значность в обозначениях —
                           ; буква b может быть транс-
                           ; лирована как шестнадцати-
                           ; ричная цифра «одиннадцать»)

    ; Проверка на ошибку и ожидание готовности
         mov dx, status
not_yet: in al, dx         ; Проверка  наличия бумаги в
         test al, bit 5    ; принтере
         jz error          ; Бумага закончилась

         test al, bit 7    ; Проверка состояния линии
         jz     not_yet    ; Принтер еще занят, повторяем
                           ; опрос готовности
         inc bx            ;
         loop print        ; Переход к печати следующего
                           ; символа
         ret
```

```
; Выдача сообщения об ошибке
error:    lea dx, err_st
          mov ah, 09h
          int 021h
          ret
err_st    db «В принтере закончилась бумага !$»
```

Использование аппаратного прерывания

Для передачи информации на принтер в фоновом режиме можно использовать аппаратное прерывание, генерируемое контроллером параллельного порта. Для такой работы надо выполнить следующие действия:

- установить вектор прерывания на новый обработчик,
- разрешить аппаратное прерывание, размаскировав его путем соответствующей записи в регистр IMR контроллера прерываний (см. п.1.4),
- послать на печать первый символ,
- остальные символы должны передаваться при помощи обработчика прерывания в фоновом режиме,
- после завершения печати запретить прерывания.

В целом эти действия аналогичны программе передаче массива данных через последовательный порт, приведенной в п. 2.2 настоящего пособия. Более подробно эти вопросы рассмотрены, например, в [7, 12].

Рассмотрим фрагменты такой программы. Напомним, что запрос на прерывание от параллельного порта LPT1 приходит по линии IRQ7 на вход ведущего контроллера (см. рис. 1.6).

Установка нового обработчика IRQ7, (INT 0Fh)

```
mov al, 0Fh     ; Номер вектора
lea dx, irq7    ; DS:DX — адрес нового обработчика
mov ah, 025h    ; Установка вектора прерывания
int 021h
```

Обработчик прерывания

```
irq7:   pusha
        cli
        call send          ; Посылаем символ
        mov al, 20h        ; Завершение прерывания (сброс
                           ; заявки в ISR — см.п.1.4)
        out 20h, al
        sti
        popa
        iret
```

Посылать символ можно описанным выше фрагментом программы непосредственного низкоуровневого управления портом принтера, но его нужно доработать таким образом, что бы устанавливался бит разрешения прерывания.

Разрешение прерывания

```
ena_int: cli
         in al, 021h       ; Разрешение IRQ7 через ПКП
         and al, not bit 7
         out 021h, al
         sti
         ret
```

Запретить формирование запроса на прерывание IRQ7 в контроллере параллельного порта можно следующим образом

```
         mov al, 01100xb
         mov dx, 0378h +2
         out dx, al
```

Запрещение запретить реакцию ПКП на запрос IRQ7 можно как показано ниже (см. также п. 1.4).

```
         in al, 21h
         or al, bit 7
         out 21h, al
```

Кроме кодов привычных текстовых символов, которые после передачи их в принтер изображаются на бумаге, имеются коды, управляющие положением печатающей головки, перемещением бумаги и т.п. Полный перечень управляющих кодов достаточно велик [11], кроме того, у принтеров разных типов могут быть отличия. В табл. 2.8 приведены некоторые из стандартных (ASCII) управляющих кодов.

Таблица 2.8

Примеры кодов управления принтером

Десятичные коды символов	Функция
8	Возврат назад на один символ
10	Переход на начало строки
13	Переход на новую строку
24	Очистка буфера

На многих принтерах символы не печатаются до тех пор, пока не получен код возврата каретки (код 10 из табл. 2.8) или до тех пор, пока не введена целая строка данных. Символы могут остаться в буфере принтера после завершения работы программы. Когда начнется новая передача данных на принтер, эти символы будут напечатаны. Чтобы избежать этой ситуации рекомендуется очистить буфер перед началом печати, а также после завершения программы [11]. Для этого достаточно послать в принтер код 24.

Настройка режимов (спецификаций) печати

Большинство принтеров могут воспринимать специальные управляющие последовательности символов и в соответствии с ними менять свой режим. Такие последовательности не выводятся на печать, а только обрабатываются внутренними схемами принтера, т.е. являются командой. Посылкой на принтер таких команд осуществляется настройка формата страницы, стиль шрифта, переход между алфавитно-цифровым и различного вида графическими режимами и т. п. Как правило, эти последовательности начинаются с управляющего символа ESC (десятичный код 27), отсюда и возникло название ESC-последовательности. Перечень режимов, устанавливаемых таким способом для каждого типа принтера свой. Некоторые ESC-последовательности (команды) приведены в табл. 2.9.

Большинство функций (режимов), приведенных в табл. 2.9, не требуют комментариев. Заметим только, что почти все они требуют отмены для перехода на другую настройку. Отмена выполняется посылкой в принтер соответствующих ESC-кодов. Особенности имеются у режима 9-игольчатой графики. В таблице отсутствует ESC-последовательность отмены этого режима. Команда, устанавливающая графический режим, должна сообщать, какое число байтов графических данных будет передано (но не больше одной строки). После того, как эти данные будут приняты принтером как графические, он автоматически вернется в текстовый режим.

Установка режима 9-игольчатой графики для принтера Epson, имеющегося в лаборатории, осуществляется посылкой в принтер следующей последовательности: 27, 94, **a, n**1, **n**2, «массив графической информации». Здесь первые два байта — ESC последовательность из табл. 2.9, **a** — параметр, задающий обычную (**a**=0) или двойную (**a**=1) плотность, а **n**1+ 255**n**2 — количество колонок графического изображения. Для печати одной колонки требуется посылка двух байтов, так как высота одной колонки составляет здесь девять точек, для передачи которых требуется 9 бит информации.

Таблица 2.9

Управляющие последовательности для принтера Epson

ESC-последователь-ность	Содержание
ESC 14 ESC 20	Выбор двойной ширины для одной строки Отмена ——— ’’———’’———’’———
ESC 15 ESC 18	Выбор уплотненного режима Отмена ——— ’’———
ESC '*'	Выбор графического режима
ESC '0' ESC '1' ESC '2' ESC '3'	Интервал между строками 1/8 дюйма Интервал между строками 7/72 дюйма Интервал между строками 1/6 дюйма Интервал между строками n/8 дюйма
ESC '4' ESC '5'	Режим курсива Отмена курсива
ESC 'G' ESC 'H'	Выбор двух ударного режима Отмена ——— ’’———
ESC '^'	Выбор 9-игольчатой графики

2.6.3. Описание лабораторной установки

Лабораторная установка состоит из IBM PC-совместимой ПЭВМ, соединенной с принтером Epson FX-80. В кабель, соединяющий LPT порт с принтером, «врезан» выносной пульт с гнездами, подсоединенными к линиям интерфейса и обозначенными согласно табл.1.2 п. 1.6. На этот пульт заведены также три кабеля, подключенных к осциллографу и к проводам с наконечниками-штырями, так что на любой из входов осциллографа можно подать любой из сигналов интерфейса.

2.6.4. Программа работы

- Повторить основные свойства параллельного порта и протокол обмена информацией через интерфейс Centronics (п.1.6)
- Изучить приемы работы с печатающим устройством, выяснить назначения кнопок и индикаторов на лицевой панели, проверить их действие.
- «Вручную», т.е. при помощи редактора портов rport или в отладчике изучить характер логики входов/выходов параллельного порта ПЭВМ (определить какому уровню сигнала, H или L, соответствуют единицы в регистрах управления и состояния).

- Изучить реакцию цепей интерфейса на состояние принтера. Для этого можно вставлять и вытаскивать бумагу из принтера, включать и выключать режим on-line на панели принтера.
- Вывести на печать ряд символов, воспользовавшись одной из функций BIOS или DOS, описанных выше.
- Вывести на печать последовательность символов, не пользуясь функциями BIOS и MS DOS, т.е. необходимо напрямую работать с портами принтера.
- Изучить протоколы обмена параллельного интерфейса Centronics, наблюдая сигналы с помощью осциллографа. Для этого на основе предыдущего пункта составить программу циклического вывода в принтер пачки символов. Следует выбрать коды, вывод которых не приводит в движение механизмов принтера, например, коды 1, 2, 4. Следует также ввести задержку после вывода пачки, что позволит более четко установить взаимодействие устройств, показанное на рис. 1.10. Снять осциллограммы сигналов. По осциллограммам определить интервалы времени, отмеченные на рис. 1.10, а также задержку фронта сигнала BUSY относительно фронта строба и задержку спада сигнала BUSY относительно фронта сигнала -ACK. Для этого построить три временных диаграммы:

— общая диаграмма, подобная рис. 1.10, но для пачки символов (сигналы D0, D1, D2, -STROBE, BUSY, -ACK),
— диаграмма взаимодействия сигналов STROBE и BUSY,
— диаграмма взаимодействия сигналов –ACK BUSY.

Для построения двух первых диаграмм использовать для запуска развертки один из сигналов D0, D1, D2. Для построения третьей диаграммы следует взять синхронизацию от фронта сигнала –ACK. На этой диаграмме показать строб следующего байта. Объяснить причину «дрожания» этого сигнала.

- Изучить функциональные возможности принтера при работе с ESC-кодами. Для этого составить программу, выводящую на печать управляющие последовательности и текстовые строки, по которым можно определить режим печати принтера.
- Используя режим девятиигольчатой графики, запрограммировать и напечатать символ, заданный преподавателем.

2.6.5. Содержание отчета

В отчете привести тексты программ, реализующих изученные режимы работы принтера и образцы напечатанного текста.

Привести временные диаграммы работы использованного интерфейса, полученные экспериментальным путем с помощью осциллографа, с указанием реальных временных параметров и амплитуд, обратив особое

внимание на относительные задержки фронтов и спадов сигналов. Показать на эпюре причинно-следственные связи между сигналами интерфейса.

В выводах отразить основные свойства изученного интерфейса, его возможности и удобства для программиста при использовании для управления периферийным устройством. Охарактеризовать использованный принтер с точки зрения потребителя.

2.7. Управление жидкокристаллическим индикатором

2.7.1. Цели работы

Целью работы является знание свойств и характеристик современных жидкокристаллических индикаторов (ЖКИ), приобретение навыков программного управления такими приборами. Дополнительная цель работы — приобретение навыков программного управления периферийным устройством через параллельный порт ПЭВМ.

2.7.2. Общие положения

Жидкокристаллические приборы используются для построения индикаторов различного назначения с начала 70-х годов, однако использование их до последнего времени было не всегда удобно с точки зрения схемотехники, в основном, по двум причинам. Во-первых, схемы управления такими индикаторами были достаточно громоздкими и поэтому неудобными для применения в недорогих цифровых приборах и устройствах (прежде всего, это относится к матричным индикаторам). Во-вторых, в ЖКИ прошлых лет для достижения необходимой контрастности изображения требовалось дополнительное напряжение отрицательной полярности. В последние годы на рынке появились сравнительно дешевые индикаторы, свободные от указанных недостатков. Удобство использования этих приборов обеспечено их внутренней структурой: такой индикатор содержит кроме собственно ЖКИ-дисплея специализированные большие интегральные схемы, управляющие индикацией и обменом информацией с внешней средой. Для разработчика такой индикатор представляет собой устройство, управляемое программно. В лаборатории имеется двухстрочный 16-разрядный алфавитно-цифровой матричный индикатор, структурная схема которого приведена на рис. 2.3.

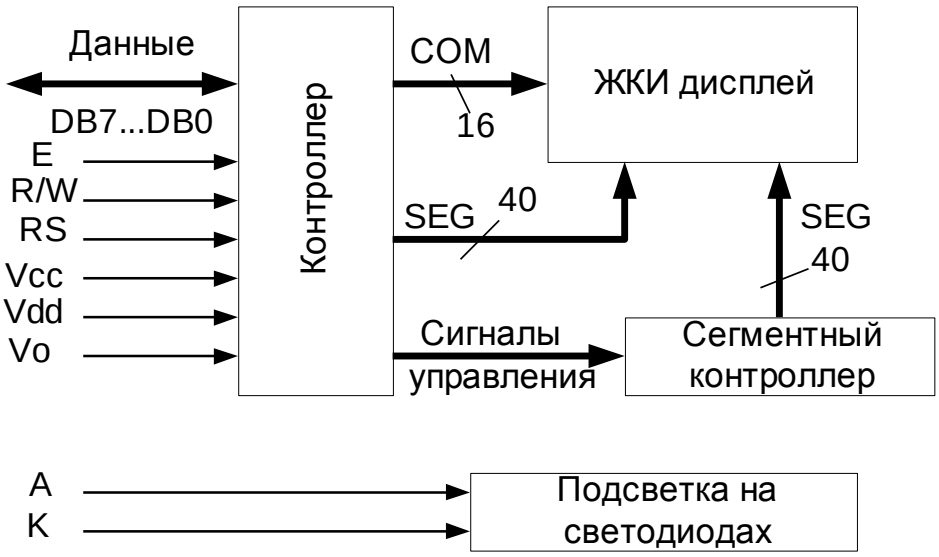

Рис. 2.3. Структурная схема жидкокристаллического индикатора

Внешние выводы прибора имеют следующее назначение.

DB7...DB0 — данные,

E — строб записи или чтения,

R/W — команда чтения/записи,

RS — выбор регистра данных или команды,

Vss, Vdd, Vo — входы питания, соответственно, общий (GND), +5В и напряжение, регулирующее контрастность изображения,

A, K — питание светодиодов подсветки дисплея (анод, катод).

Изучение индикаторов разных фирм показывает совпадение логики работы однотипных приборов. Контроллеры индикаторов содержат постоянное и оперативное запоминающие устройства. В постоянной памяти записана таблица кодов символов. Оперативная память состоит их памяти данных дисплея (Display Data RAM — DDRAM) — промежуточная память, в которой хранятся отображаемые символы, и памяти генератора символов (Character Generator RAM — CGRAM), в которой могут храниться поэлементно запрограммированные пользователем символы (не более восьми). Управление индикатором осуществляется путем подачи на его входы сигналов ТТЛ-уровня. При этом индикатор как программно управляемый прибор способен выполнять ряд команд, список которых приведен в табл. 2.10.

В таблице через H и L обозначены соответственно высокий и низкий уровень потенциала на соответствующем внешнем выводе прибора. X — безразличное состояние. Другие обозначения пояснены в левом столбце. Для каждой команды указано время ее выполнения. Это вносит определенные

Таблица 2.10

Команды индикатора

Команда	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Время испол-нения
Очистка дисплея	L	L	L	L	L	L	L	L	L	H	1.64 мс
Сброс курсора. Перемещает курсор в начало строки	L	L	L	L	L	L	L	L	H	X	1.64 мс
Режим ввода символов I/D устанавливает направле-ние движения курсора: H – вправо, L - влево. SH – сдвиг дисплея: H - есть, L – нет	L	L	L	L	L	L	L	H	I/D	SH	40 мкс
Включение/ выключение Дисплей D, Курсор C, Мигание B: H – включено, L – выключено	L	L	L	L	L	L	H	D	C	B	40 мкс
Сдвиг. S/C: H - сдвиг дисплея, L – сдвиг курсора. R/L: H -вправо, L – влево	L	L	L	L	L	H	S/C	R/L	X	X	40 мкс
Функция DL: H - 8-бит. интерфейс, L - 4-бит. интерфейс; N - 2-строчный дисплей, F: H – 5∗10 точек, L - 5∗7 точек	L	L	L	L	H	DL	N	F	X	X	40 мкс
Ввод адреса CGRAM	L	L	L	H	A5	A4	A3	A2	A1	A0	40 мкс
Ввод адреса DDRAM	L	L	H	A6	A5	A4	A3	A2	A1	A0	40 мкс
Чтение BF и счетчика адресов памяти BF: H - занято, L – готово	L	H	BF	A6	A5	A4	A3	A2	A1	A0	0
Запись данных в DDRAM и CG	H	L	D7	D6	D5	D4	D3	D2	D1	D0	46 мкс
Чтение данных из DDRAM	H	H	D7	D6	D5	D4	D3	D2	D1	D0	46 мкс

ограничения на частоту обращения к индикатору. Теоретически можно сократить время между командами за счет опроса флага занятости (BF - busy flag), но на практике большинство разработчиков не используют эту возможность. Если попытаться до истечения указанного в табл. 2.10 времени выполнения команды выполнить следующую команду, то последняя выполнена не будет.

На рис. 2.4 приведена типичная временная диаграмма сигналов на входах индикатора в режиме записи в него данных или команды. При обращении к регистру данных на вход RS должен подаваться высокий потенциал. Для записи на вход R/W подается низкий потенциал. Операция записи или чтения стробируется сигналом на входе E (положительный импульс).

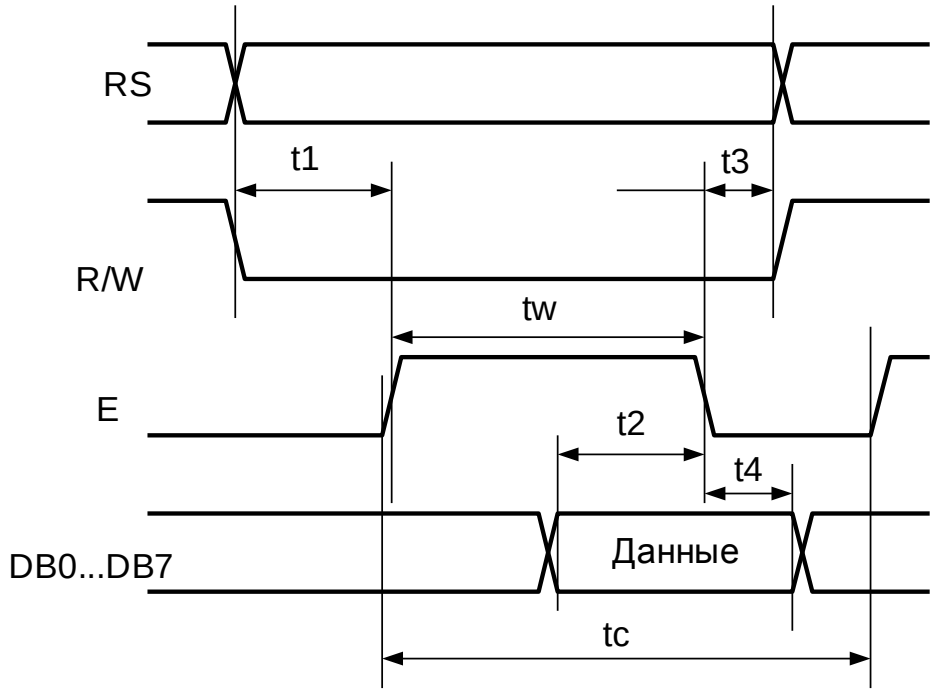

Рис. 2.4. Временная диаграмма записи данных и команд в ЖКИ

Данные фиксируются в соответствующем регистре по спаду строба. Временные параметры диаграммы приведены в табл. 2.11. Следует обратить внимание на то, что большинство параметров ограничено снизу, т.е. увеличение задержки не приводит к нарушению работы прибора. Сверху ограничены лишь длительности фронтов строба, но эти параметры определяются быстродействием и мощностью применяемых формирователей управляющих сигналов, а также электрическими параметрами линий передачи этих сигналов. В данной работе управление дисплеем осуществляется сигналами с выходного разъема параллельного порта стандартного персонального компьютера, которые формируются микросхемами, удовлетворяющими требованиям табл. 2.11.

Таблица 2.11

Временные характеристики для цикла записи

Параметр	Обозначение	Значение, нс		Контакт
		min	max	
Период повтора цикла	tc	500		E
Длительность строба	tw	220		E
Длительность фронта и спада строба	на рис. 2.4 не показана		25	E
Длительность пребывания RS и R/W в активном состоянии до строба	t1	40		RS, R/W
Установка данных до строба	t2	20		DB0...DB7
Время удержания RS и R/W после строба	t3	10		RS, R/W
Время удержания данных	t4	60		DB0...DB7

2.7.3. Управление ЖКИ-дисплеем через параллельный порт ПЭВМ

При выполнении работы требуется программно сформировать временную диаграмму в соответствии с рис. 2.4. Входы индикатора соединены с выходами параллельного порта, как описано в п. 2.7.4. Для формирования необходимых управляющих сигналов нужно последовательно записывать в регистры параллельного порта соответствующие коды.

Логическое описание параллельного порта приведено в п. 1.6.

Для ввода символов в DDRAM и вывода его на индикацию используются коды, представленные в табл. 2.12. Заметим, что коды в диапазоне 020h…07Ah совпадают со стандартными кодами ASCII. Не все коды, содержащиеся в CGRAM, приведены в табл. 2.12. Незаполненные ячейки таблицы либо не используются, либо содержат редко применяемые символы.

Таблица 2.12

Сокращеная таблица кодирования символов для ЖКИ

Млад-шие 4 бита кода символа	Старшие 4 бита кода символа (hex)															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	CGR			0	@	P	`	p			Б	Ю	ч		Д	¼
1	CGR		!	1	A	Q	a	q			Г	Я	ш		Ц	⅓
2	CGR		“	2	B	R	b	r			Ё	б	ъ		Щ	½
3	CGR		#	3	C	S	c	s			Ж	в	ы		д	
4	CGR		$	4	D	T	d	t			З	г	ь		ф	
5	CGR		%	5	E	U	e	u			И	ё	э		ц	
6	CGR		&	6	F	V	f	v			й	ж	ю		щ	
7	CGR		‘	7	G	W	g	w			Л	з	я			
8	CGR		(	8	H	X	h	x			П	и				
9	CGR		)	9	I	Y	i	y			У	й				
A	CGR		*	:	J	Z	j	z			Ф	к				
B	CGR		+	;	K	[	k				Ч	л				
C	CGR		,	<	L		l				Ш	м				
D	CGR		-	=	M	]	m				Ъ	н				
E	CGR		.	>	N	^	n				Ы	п				
F	CGR		/	?	O	_	o				Э	т				

Адреса 00h…0fh предназначены для символов, спроектированных пользователем. В табл. 2.12 это обозначено как «CGR» — Character Generator RAM. На экране дисплея символы формируются по мозаичному принципу в ячейке 5∗7 пикселей, поэтому понятно, что реальные очертания символов отличаются от приведенных в таблице.

Инициализация ЖКИ

После включения питания индикатор автоматически приходит в исходное состояние, но для этого нарастание напряжения питания должно происходить по установленному разработчиком закону (линейный рост со скоростью, лежащей в установленных рамках). Так как не всегда удается гарантировать эти условия, то разработчиком рекомендован алгоритм

программной инициализации дисплея, который описывается ниже. Инициализация состоит в подаче на информационные и управляющие входы ЖКИ кодовой последовательности в определенном темпе. Коды инициализации подаются на входы RS, R/W, DB7,...,DB0. Такую комбинацию на входах прибора можно рассматривать как 10-разрядное слово. Назовем его управляющим словом УС. Подача каждого УС должна стробироваться сигналом E в соответствии с диаграммой на рис. 2.4. Тогда инициализация сводится к следующей последовательности действий:

Включение питания
Пауза 15 мс
УС = 000011xxxx
Пауза 4.1 мс
УС = 000011xxxx
Пауза 100 мкс
УС = 000011xxxx
Пауза 40 мкс
УС = 00001110xx
Пауза 40 мкс
УС = 0000001000
Пауза 40 мкс
УС = 0000000001
Пауза 1.64 мс
УС = 0000001100
Пауза 40 мкс
УС = 0000001100

Рекомендации по программированию.

Управление дисплеем сводится, главным образом, к записи в его регистры либо команд, либо данных. Стробы записи при этом формируются одинаково, но сигналы на входе RS выбора регистра различны. Поэтому рекомендуется заранее разработать процедуры записи команды и записи данных и оформить их в виде подпрограмм или пользовательских макрокоманд. Такую же процедуру следует заготовить для чтения данных из ЖКИ, если чтение необходимо.

Выдержка времени (пауза) после выполнения любого действия с ЖКИ в соответствии с табл. 2.10 является обязательной. Эти паузы также рекомендуется оформить в виде процедур.

Необходимо помнить, что управляющие сигналы E, R/W и RS при выполнении данной работы поступают с выходов регистра управления параллельного порта, а характер логики (положительная или отрицательная) в разных разрядах этого регистра разный (см. п. 1.6).

2.7.4. Описание лабораторной установки

Лабораторная установка представляет собой IBM/PC-совместимую ПЭВМ, к параллельному порту которой подключен исследуемый ЖКИ дисплей (в лаборатории использован дисплей типа PC 1602 LRS-FEN-B — две строки по 16 разрядов, изготовлен фирмой Powertip Technology Corporation). Подключение индикаторов к внешним разъемам портов выполнено в соответствии со следующей табл. 2.13.

Таблица 2.13

Подключение индикаторов к разъемам портов

Входы ЖКИ	Выходы параллельного порта
Выбор регистра RS	AUTO FD
Команда R/W	-SLCT IN
Строб E	- INIT
Данные B0…DB7	D0…D7

2.7.5. Программа выполнения работы

- Изучить логическое описание индикатора и порядок работы с параллельным портом.
- Написать и отладить программу инициализации дисплея и вывода на него текста.
- Исследовать структуру DDRAM и взаимодействие DDRAM с «окнами» дисплея (т.е. определить объем памяти данных и выяснить, из каких ячеек выводится информация на индикацию).
- Изменить программу вывода текста на дисплей, включив в неё, по согласованию с преподавателем, дополнительные «эффекты» (увеличенный размер символа, мерцание курсора, сдвиг дисплея, «бегущая строка», запись символа по адресу в произвольную ячейку DDRAM, чтение DDRAM).

2.7.6. Содержание отчета

- Функциональная схема исследуемого дисплея.
- Текст программы, реализующей заданный режим работы ЖКИ.
- Выводы, содержащие качественную оценку эргономических качеств ЖКИ данного класса и удобства их применения с точки зрения схемотехники.

2.8. Исследование последовательного порта и асинхронного приемопередатчика

2.8.1. Цели работы

Целью работы является знание принципов организации последовательного порта (COM-порта) и получение навыков программирования последовательного асинхронного приемопередатчика, реализующих обмен информацией по стандарту RS-232C в составе IBM PC.

2.8.2. Общие положения

В работе исследуется логическая организация и работа наиболее распространенного средства связи ПЭВМ с внешним миром — последовательного асинхронного приемопередатчика (АСПП), работающего на интерфейс RS-232C. Описание интерфейса приведено в п. 2.7. Здесь приводится описание работы и порядок программной настройки АСПП.

Универсальный асинхронный приемопередатчик в составе IBM PC позволяет организовать две разнонаправленные линии последовательной связи. Во многих приложениях используется только одна линия, например, приемная линия при работе с мышью. Аппаратная реализация АСПП в ПЭВМ архитектурной линии IBM/PC нашла свое воплощение в ряде специализированных микросхем, из которых наиболее распространенной в настоящее время является АСПП 16550A и ее модификации.

В состав IBM PC обычно входят два АСПП, которые называются последовательными портами COM1 и COM2. Для программиста каждый из них представлен восемью адресами в пространстве ввода-вывода (которые также принято называть портами). Каждому из АСПП также предоставлен вход контроллера прерываний. COM1 занимает порты с адреса 03F8, и линию IRQ4 (номер вектора — 0Ch). Для COM2 базовый адрес — 02F8, используется линия IRQ3 (вектор 0Bh).

Ниже приведены выборочные сведения о портах АСПП, необходимые для выполнения лабораторной работы. Конкретные адреса портов относятся к COM1. В скобках указаны смещения относительно базового адреса.

Порт 03F8 (+0). Регистр данных

Основное назначение — запись данных для передачи, чтение принятых данных. Фактически по этому адресу идет обращение к двум разным регистрам с коммутацией по чтению/записи, т.е. командой записи в этот порт данные записываются в буферный регистр *передатчика*, а командой чтения данные читаются из буферного регистра *приемника*.

При настройке скорости, когда установлен бит 7 управляющего слова (см. порт 03FB), по этому же адресу доступен регистр младшего байта делителя частоты тактового генератора (старший байт делителя записывается в порт по адресу 03F9). Делитель вычисляется по формуле 115200 / r, где r — скорость в бит/с. В табл. 2.14 приведены значения делителя для стандартного ряда скоростей. В регистр делителя можно записать любое 16-разрядное число, кроме нуля, и АСПП будет работать в соответствующем темпе. Запись нуля в регистр делителя в разных ЭВМ вызывает различную реакцию.

Таблица 2.14

Значения делителя для стандартного ряда скоростей работы АСПП

Скорость, бит/с	Делитель		Скорость, бит/с	Делитель
110	0410h		4800	0018h
150	0300h		9600	000ch
300	0180h		19200	0006h
600	00c0h		38400	0003h
1200	0060h		57600	0002h
2400	0030h		115200	0001h

Замечание. При записи в порт 03F8 данные помещаются в буферный регистр передатчика. Сдвиговый регистр передачи, получив данные из буферного регистра, преобразует их в последовательный код. При этом буферный регистр снова готов к записи следующего байта для передачи, что сигнализируется битом 5 регистра состояния линии, и порождает заявку на прерывание (см. рис. 1.13).

Порт 03F9 (+1). Регистр разрешения прерываний

Основное назначение — разрешение прерываний от АСПП. При установке в единицу бита 0, 1, 2 или 3 прерывания по соответствующей причине разрешены, при установке в ноль — запрещены.

Бит 0 — прерывание по готовности приемника (байт пришел и находится в буферном регистре данных приемника).

Бит 1 — прерывание по готовности передатчика (передача предыдущего байта завершена, передатчик готов к посылке следующего байта).

Бит 2 — прерывание по обрыву в линии или по ошибке переполнения приемника (пришел очередной байт, а предыдущий еще не был прочитан).

Бит 3 — разрешение прерывания по изменению состояния модема (в лаборатории не используется).

Регистр доступен для чтения и записи. При настройке скорости по этому же адресу доступен регистр, куда записывается старший байт делителя частоты тактового генератора.

Порт 03FA (+2). Регистр идентификации прерываний

Позволяет определить причину прерывания (использование этого регистра имеет смысл, если были разрешены прерывания по нескольким причинам). Регистр доступен только для чтения, в данной работе не используется.

Порт 03FB (+3). Регистр управления линией

Основное назначение — настройка параметров приемопередатчика. При настройке устанавливается частота работы приемопередатчика и формат посылки (см. п.1.7.2). Регистр доступен как для записи, так и для чтения. Функциональное назначение разрядов регистра приведено в табл. 2.15.

Порт 03FC (+4). Регистр управления модемом

Регистр управления модемом позволяет, в частности, установить диагностический режим работы АСПП. Установка бита 4 этого регистра в единицу приводит к замыканию выхода передатчика на вход приемника внутри АСПП. При попытке выполнить передачу в этом режиме сигналы в линию не уходят. В лаборатории не следует пользоваться диагностическим режимом, так как на всех рабочих местах организована «физическая» линия связи. Регистр доступен для чтения и записи.

Таблица 2.15

Назначение разрядов регистра управления линией

Биты	Назначение
[1:0]	Количество информационных бит: 00 — 5 бит 01 — 6 бит 10 — 7 бит 11 — 8 бит
2	Количество стоповых бит: 0 — 1 бит 1 — 2 бита (для пятибитовой посылки «полтора бита»)
[4:3]	Контроль: x0 — без контроля 01 — дополнение до нечетности 11 — дополнение до четности
5	Постоянная четность 0 — отмена постоянной четности 1— постоянное значение контрольного бита при единичном значении бита 3 в зависимости от бита 4
6	Имитация обрыва линии 1 — посылка нулей в линию
7	При установке в 1 — режим настройки скорости (порты 3F8, 3F9 используются для загрузки значения делителя частоты)

Порт 03FD (+5). Регистр состояния линии

Доступен только для чтения. Отражает состояние АСПП. Назначение разрядов приведено в табл. 2.16. Установка битов 1…4 вызывает запрос на прерывание, если оно разрешено (см. порт 03F9).

Таблица 2.16

Назначение разрядов регистра состояния линии

Биты	Состояние
0	Данные получены и готовы для чтения. Сбрасывается чтением приемника
1	Ошибка переполнения при приеме (принят новый байт раньше, чем прочитан предыдущий; предыдущий теряется).
2	Ошибка четности при приеме
3	Ошибка синхронизации (не принята стоповая посылка)
4	Обнаружен обрыв линии (длительность нахождения входного сигнала в состоянии «0» превышает длительность посылки)
5	Буферный регистр передатчика пуст, можно записывать следующий передаваемый байт.
6	Сдвиговый регистр передатчика пуст, передача закончена

2.8.3. Программирование АСПП

Инициализация АСПП

Для инициализации нужно записать байт с параметрами настройки в управляющий регистр. Если необходимо изменить скорость, то установить при этом бит 7 управляющего регистра в единицу. В порт 03F8 записать младший байт делителя, в порт 03F9 — старший байт делителя, затем сбросить 7 бит порта 03FB в 0. Данные, записанные в управляющий регистр, сохраняются и могут быть впоследствии прочитаны. После этого проинициализировать регистр разрешения прерываний (порт 03F9). Если прерывания не используются, записать ноль.

Передача данных

Убедиться в том, что буферный регистр передатчика пуст. Признаком этого является установленный бит 5 регистра состояния линии (порт 03FD). Записать байт для передачи в регистр данных (порт 03F8), при этом бит 5 регистра состояния линии будет автоматически сброшен в 0.

Прием данных

Убедиться в том, что бит 0 регистра состояния линии (порт 03FD) установлен в 1, т.е. данные приняты из линии и находятся в буферном регистре приемника. Прочитать байт из регистра данных (порт 03F8), при этом бит 0 регистра состояния линии будет автоматически сброшен в 0.

Использование прерываний

Вызовы прерываний изучались во второй вводной работе (п. 2.2). Там же приведен пример настройки асинхронного приемопередатчика на работу в режиме прерываний.

В программе обработки прерывания необходимо предусмотреть действия, снимающие причину прерывания (см. выше описание регистра идентификации прерывания).

2.8.4. Описание лабораторной установки

Работа выполняется на ПЭВМ, оснащенной последовательным портом. Линия связи организована путем соединения входов приемника с выходами передатчика того же самого АСПП. Для наблюдения сигналов в линии она подключена к входу осциллографа.

Для выполнения случайной настройки обоих портов на рабочих местах установлена программа **rand_rs.exe.**

Обратите внимание на то, что на разных рабочих местах доступными для наблюдения являются линии связи, подключенные к разным портам (COM1 или COM2). Проверить факт подключения к одному или другому порту можно с помощью записи кода в регистр данных передатчика с последующим чтением регистра данных приемника. При этом следует убедиться, что проверяемый порт не находится в диагностическом режиме.

2.8.5. Программа работы

- При подготовке к выполнению работы изучить основные сведения об АСПП и интерфейсе RS-232C (пп.1.7 и 2.8).
- По согласованию с преподавателем выполнить несколько заданий из приведенного перечня:

1. Написать и отладить программу циклического приема данных в режиме опроса готовности с индикацией на экране дисплея факта приема байта (выводить на экран произвольный символ или количество принятых байт). В качестве источника данных использовать мышку, подключенную к последовательному порту. Для подачи электрического питания в мышку записать **02bh** в регистр управления модемом. Попытайтесь определить, сколько байт передает мышь при перемещении и при нажатии/отпускании кнопок. Драйвер мыши при этом должен быть выгружен.
2. Написать и отладить программу циклической передачи данных в режиме опроса готовности. Снимите осциллограмму посылки символа (байта) и укажите на ней структуру посылки. «Оцифровка» осей осциллограммы, т.е. указание масштабов времени и напряжения с учетом полярности, обязательна. Для облегчения синхронизации осциллографа рекомендуется после посылки байта вставить регулируемую временную задержку.
3. Запустить на исполнение программу **rand_rs.exe**, после чего с помощью программы из предыдущего пункта определить настройку порта. Привести осциллограммы, подтверждающие правильность решения задачи.
4. Настроить АСПП на заданный режим и убедиться в правильности выполнения задания с помощью программы циклической передачи данных. Привести осциллограммы, подтверждающие правильность настройки.
5. Написать и отладить программу циклической передачи в режиме опроса готовности символа, введенного с клавиатуры.
6. Снять осциллограммы в режимах сигналов в начале линии передачи и на конце при длине 50 м и 100 м.
7. Выполнить одно из приведенных заданий в режиме прерывания.

- Составить отчет о проделанной работе.

2.8.6. Содержание отчета

В отчете привести цель работы, формулировки заданий, тексты программ с комментариями, а также осциллограммы наблюдавшихся процессов.

В выводах кратко охарактеризовать свойства АСТПП и интерфейса. Отметить особенности (если они обнаружены), не отраженные в настоящем описании.

3. ПРАКТИКУМ ПО РАБОТЕ С СИСТЕМОЙ ПРЕРЫВАНИЙ

3.1. Общие положения

3.1.1. Цели практических работ

В данном разделе предлагается цикл практических работ, посвященных различным аспектам работы с прерываниями в ЭВМ и вычислительных системах на основе микропроцессоров архитектурной линии Intel 8086…Pentium IV. Целью этих работ является изучение организации системы прерываний современных ЭВМ и приобретение навыков программирования с использованием возможностей, предоставляемых базовой системой ввода/вывода (BIOS), операционной системой MS DOS и средствами языка ассемблера. Рассмотрению подлежат следующие вопросы:

- типы прерываний и источники возникновения запросов,
- принципы обработки прерываний и флаговая логика микропроцессоров,
- особенности обработки внутренних и аппаратных прерываний,
- логика взаимодействия микропроцессоров i80x86 и контроллера прерываний любой модификации на основе БИС i8259, включая расширения функциональных возможностей системы прерываний,
- правила перехвата прерываний пользовательской программой и правила дополнения стандартных обработчиков,
- организация собственных прерываний,

Перечисленный круг вопросов способствует углублению знаний и приобретению практических навыков по курсам "ЭВМ и периферийные устройства", "Интерфейсы внешних устройств" и "Системное программное обеспечение ЭВМ". Интегрирование полученных теоретических знаний по этим дисциплинам на практических занятиях позволит студентам сформировать целостное представление о взаимодействии аппаратных и программных средств при организации обмена информацией между устройствами вычислительной системы.

Лабораторной базой для выполнения предлагаемого курса работ может быть класс компьютеров типа IBM PC/AT и более поздних моделей персональных ЭВМ этой архитектурной линии.

Для успешного выполнения работ требуется владение навыками разработки программ на языке ассемблера для процессоров Intel 80x86/88 в рамках стандартного курса ассемблеров (в учебном плане кафедры автоматики и вычислительной техники ФТК СПбГПУ это соответствующий раздел курса "Интерфейсы внешних устройств ") и первичное представление об операционной системе DOS.

Курс работ изложен в трех разделах и рассчитан на выполнение в течение 24 академических часов при условии предварительной домашней подготовки. Задания в каждом разделе составлены таким образом, что каждая следующая программа является небольшим усложнением предыдущей. Это позволяет экономить время на наборе текстов программ (рекомендуется использовать копирование файлов) и существенно облегчает процесс отладки. Поэтому целесообразно выполнять задания в порядке, указанном в программе работ.

Все работы могут быть дополнены индивидуальными заданиями, полученными у преподавателя.

3.1.2. Общие правила и рекомендации

В отличие от лабораторных работ, описанных в разделе 2 настоящего пособия, предлагаемые здесь работы выполняются на персональных ЭВМ обычной комплектации и не требуют специальной аппаратуры. Работы относятся к категории НИР и, в соответствии с правилами проведения этого вида занятий, к этим работам предъявляются следующие требования:

Все работы выполняются индивидуально. Предварительная подготовка дома к каждому занятию обязательна.

Полезно иметь заготовки подпрограмм или макрокоманд утилитарного назначения типа преобразования данных из 16-ричной флрмы в символьный вид, вывода текста или символа на экран, ввода параметра с клавиатуры и т.д., если это может потребоваться для выполнения вашего задания.

В отчетах требуется письменно формулировать назначение каждой из разработанных программ и в явном виде указывать их особенности. Желательно представить алгоритм, соответствующий вашему индивидуальному заданию, полученному у преподавателя на предыдущем занятии.

Для проведения работ за каждым пользователем на весь лабораторный цикл закрепляется индивидуальный каталог на пользовательском диске винчестера либо файл-сервера, если занятия проводятся на компьютерах связанных в сеть.

Нецелесообразно делать следующее:

копировать в свой каталог общедоступные программные приложения (например, транслятор и отладчик, а также тексты методических указаний),

хранить модули, легко формируемые на основе сохраняемых исходных текстов программ (листинги трансляции, карты памяти, объектные и загрузочные модули).

ЗАПРЕЩАЕТСЯ самостоятельно производить любые переключения в аппаратуре вычислительных комплексов а также вносить изменения в конфигурацию системных программных средств.

3.2. Характеристика системы прерываний IBM/PC совместимых персональных ЭВМ

3.2.1. Типы прерываний

В развитие общих замечаний о системе прерываний, приведенных выше в п. 1.4.1 и применительно к конкретному классу вычислительных машин рассмотрим одну из возможных классификаций прерываний для IBM/PC-совместимых ЭВМ. В соответствии с источником возникновения прерываний принято выделять три основных типа: внутренние, внешние и программные (рис. 3.1).

Внутренние прерывания (exceptions или исключения) вырабатываются процессором и делятся, в свою очередь, на логические и отладочные. Логические прерывания являются реакцией на арифметические ошибки типа деления на ноль или переполнения. Отладочные прерывания применяются для организации отладочных режимов. Имеются также прерывания типа «фатальная ошибка» или abort-исключения, которые возникают при невосстановимых системных сбоях, они генерируются только в защищенном режиме процессора и далее не рассматриваются

Запросы на внешние прерывания вырабатываются внешними по отношению к микропроцессору устройствами и делятся на маскируемые и немаскируемые.

Запросы на маскируемые прерывания формируются контроллером прерываний по запросам от периферийными устройств. Такие прерывания принято называть аппаратными. В случае их возникновения происходит временное переключение процессора с выполняемой задачи на обработку прерывания. Возврат происходит в ту же точку прерванной программы, из которой процессор "ушел" на обработку.

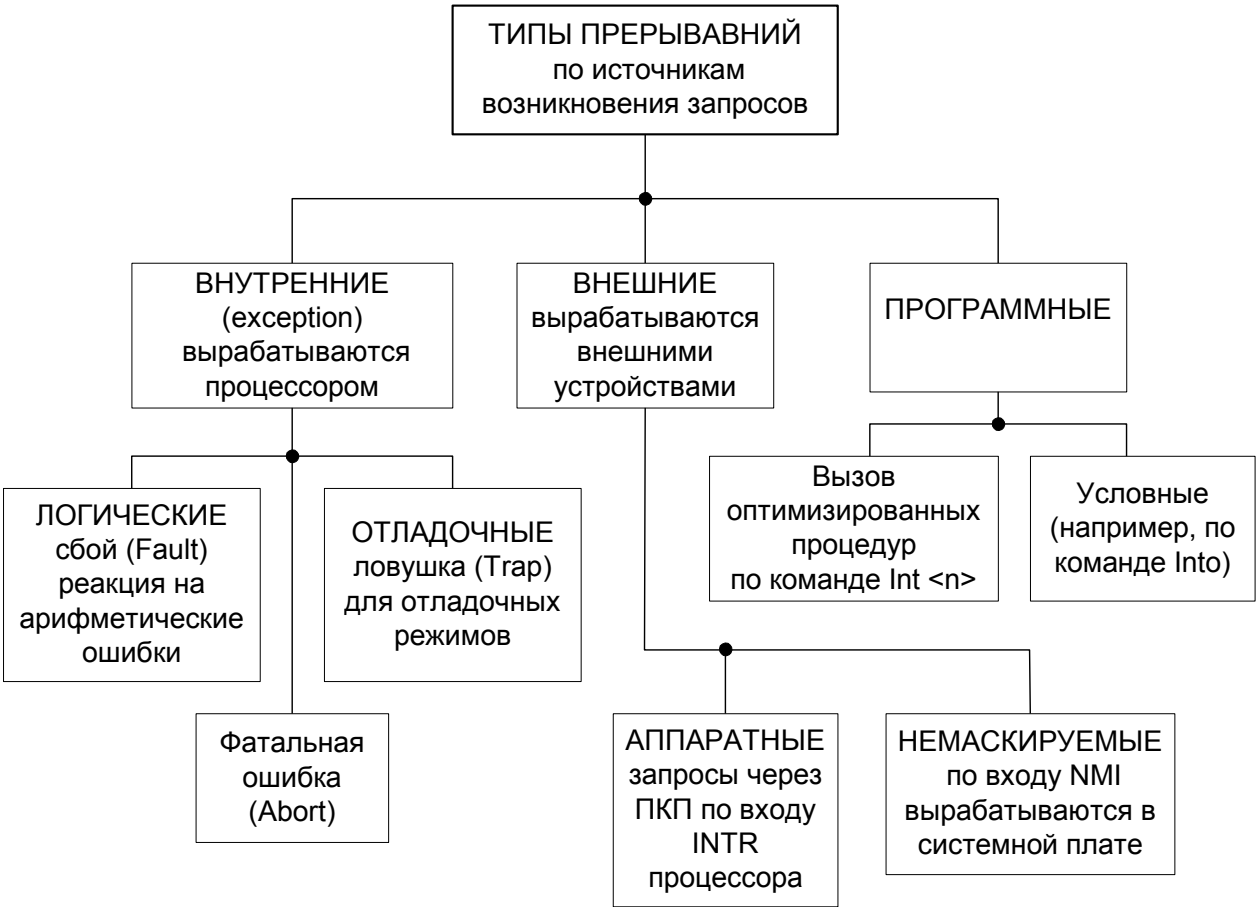

Рис. 3.1. Классификация прерываний в IBM/PC-совместимых ЭВМ

З а м е ч а н и е. Переход к обработке внешних прерываний выполняется процессором по завершении текущей команды. В это время в регистрах CS и IP (см. рис. 1.4) уже находится полный адрес следующей команды прерываемой программы. Таким образом, возврат в ту же точку означает переход к выполнению очередной команды (для внешних прерываний).

Запросы на немаскируемые формируются системной платой при возникновении ошибки четности памяти (см. также п. 1.4).

Программные прерывания представляют собой оптимизированные процедуры операционной системы, вызываемые пользовательскими или системными программами для выполнения наиболее часто применяемых операций. Для обращения к этим процедурам используется механизм прерываний. Каждая программа обработки прерывания сохраняет значения всех используемых ею регистров и затем восстанавливает их при возврате в вызывающую программу.

Программные прерывания можно инициировать специальной командой: **int <n>**, где n — тип прерывания, т.е. номер, идентифицирующий один из векторов в таблице векторов прерываний.

При исполнении команды **int <n>** микропроцессор выполняет следующие действия:

- помещает в стек регистр флагов,
- обнуляет флаг трассировки TF и флаг включения-выключения прерываний IF для исключения пошагового режима исполнения команд и блокировки других маскируемых прерываний,
- помещает в стек значение регистра CS,
- вычисляет адрес вектора прерывания, умножая n (тип прерывания) на четыре,
- загружает второе слово вектора прерывания в регистр CS,
- помещает в стек значение указателя команд IP,
- загружает в указатель команд IP первое слово вектора прерывания.

Возврат из прерывания происходит по команде IRET. При этом из стека извлекается три 16-битовых значения, загружаемых затем в указатель команд IP, регистр сегмента CS и регистр флагов RF соответственно.

Перечислим отличия при выполнении процедуры и программы обработки программного прерывания.

Вызов процедуры может быть прямым или косвенным и иметь атрибуты NEAR или FAR в зависимости от принадлежности вызывающему сегменту. Прерывание всегда вызывает косвенный переход к своей программе обработки за счет получения ее адреса из вектора прерывания.

При вызове процедуры в стеке запоминается только адрес возврата, тогда как прерывания сохраняют еще и флаги.

Содержимое регистра флагов процессора (RF) приведено на рис. 3.2.

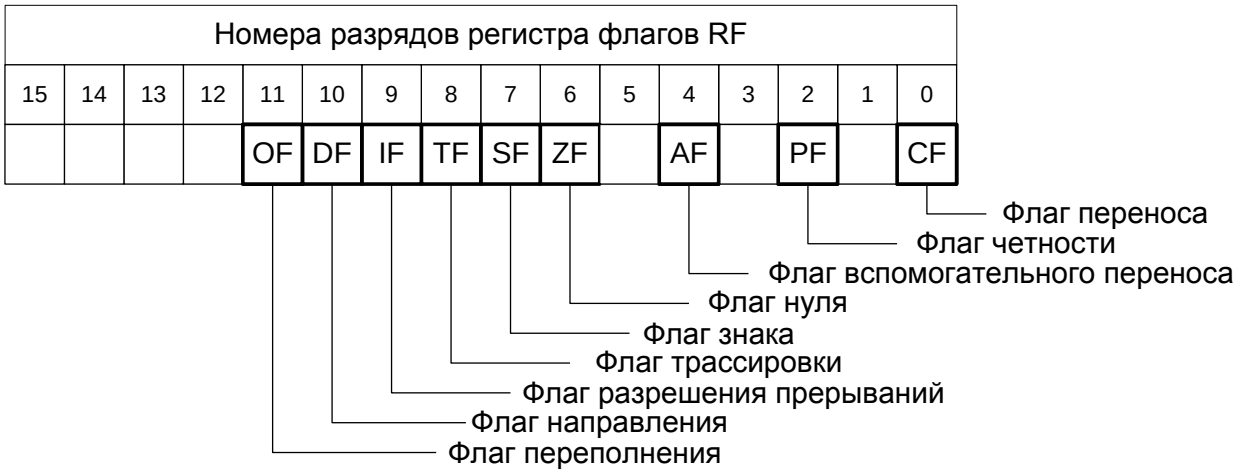

Рис. 3.2. Регистр флагов процессора

Назначение битов регистра флагов МП i80x86:

1. Бит 0, флаг переноса CF (carry flag), равен единице, если произошел перенос единицы при сложении или заем единицы при вычитании. В противном случае он равен нулю. Кроме того, CF содержит значение бита, который при сдвиге или циклическом сдвиге регистра или ячейки памяти вышел за их границы и отражает результат операции сравнения. Наконец, CF служит индикатором результата умножения. Детали см. в описании бита 11 (OF).

2. Бит 2, флаг четности PF (parity flag), равен 1, если в результате операции получено число с четным числом единиц в его битах. В противном случае он равен нулю. Флаг PF, в основном, используется в операциях обмена данными.

3. Бит 4, вспомогательный флаг преноса AF (auxiliary carry flag), аналогичен флагу CF, только контролирует перенос или заем для третьего бита данных. Полезен при выполнении операций над упакованными десятичными числами.

4. Бит 6, флаг нуля ZF (zero flag), равен единице, если результат выполнения процессором операции равен нулю. Ненулевой результат операции сбрасывает ZF в нуль.

5. Бит 7, флаг знака SF (sign flag), имеет значение только при операциях над числами со знаком. Флаг SF равен единице, если в результате арифметической или логической операции, сдвига или циклического сдвига получено отрицательное число. В противном случае он равен нулю. Иными словами, SF дублирует старший (знаковый) бит результата независимо от того, имеет результат длину 8 или 16 битов.

6. Бит 8, флаг трассировки TF (trap flag), разрешает микропроцессору исполнять программу "по шагам" и используется при отладке программ.

7. Бит 9, флаг прерывания IF (interrupt enable flag), разрешает микропроцессору реагировать на прерывания от внешних устройств. Сбрасывание IF в нуль заставляет микропроцессор игнорировать прерывания до тех пор, пока IF не станет равным единице.

8. Бит 10, флаг направления DF (direction flag), заставляет микропроцессор уменьшать на единицу (DF=1) или увеличивать на единицу (DF=0) регистр(ы) индекса после выполнения команды для работы со строками. Если DF=0, то МП будет обрабатывать строку "слева направо" (от младших адресов к старшим). Если DF=1, то обработка пойдет в обратном направлении (от старших адресов к младшим или справа налево).

9. Бит 11, флаг переполнения OF (overflow flag), в первую очередь, служит индикатором ошибки при выполнении операций над числами со знаком. Флаг OF равен единице, если результат сложения двух чисел с одинаковым знаком или результат вычитания двух чисел с

противоположными знаками выйдет за пределы допустимого диапазона значений операндов. В противном случае он равен нулю. Кроме того, OF=1, если старший (знаковый) бит операнда изменился в результате операции арифметического сдвига. В противном случае он равен 0. В сочетании с флагом CF флаг OF указывает длину результата умножения. Если старшая половина произведения отлична от нуля, то OF и CF равны 1. В противном случае оба эти флага равны 0.

Наконец, OF=0, если частное от деления двух чисел переполняет результирующий регистр.

Регистры флагов 32- и 64-разрядных процессоров имеют более сложную структуру и в данном пособии не рассматриваются.

3.2.2. Сведения о таблице векторов прерываний

Адреса программ обработки запросов на прерывания называют векторами. Каждый вектор имеет длину 4 байта, его структура показана на рис. 3.3 (см. также п.1.4 настоящего пособия).

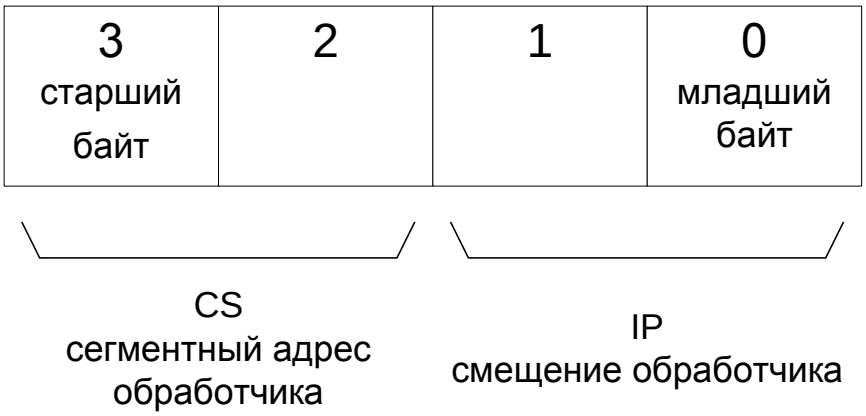

Рис. 3.3. Вектор прерывания

Младшие 1024 байта ОЗУ занимает таблица векторов или дескрипторов прерываний, которая содержит адреса 256 прерываний (табл. 3.1).

BIOS (Basic Input/Output System) — базовая система ввода/вывода — служит для организации обмена информацией процессора с внешней аппаратурой и является связующим звеном между операционной системой (ОС) и периферийными устройствами (ПУ). BIOS содержит набор программ обработки аппаратных прерываний и базирующиеся на их основе обработчики программных прерываний. Территориально BIOS размещается в ПЗУ или во флэш-памяти на материнской плате в виде отдельной БИС.

Прерывания операционной системы (в данном случае DOS), поддерживающие обмен с ПУ на более высоком уровне, опираются на прерывания BIOS.

Таблица 3.1

Распределение векторов в таблице векторов прерываний

Диапазон номеров (кодов) прерываний	Виды прерываний
0…7	Внутренние прерывания, немаскируемое прерывание NMI, Int 5 — печать текущего экрана (BIOS)
8…0f	Внешние аппаратные прерывания с запросами через ведущий контроллер прерываний (IRQ0…IRQ7)
010…01f	Вызовы функций базовой системы ввода/вывода (BIOS), указатели расположения в памяти системной информации
040…05f	Прерывания BIOS
060…067	Пользовательские прерывания
069…06f	Обслуживание системных функций
070…077	Внешние аппаратные прерывания с запросами через ведомый контроллер прерываний (IRQ8…IRQ15)
078…0ff	Прерывания, обеспечивающие ряд функций операционной системы, BIOS, интерпретатора языка BASIC, а также зарезервированные для пользовательских функций

Использование программных прерываний BIOS и DOS существенно облегчает программирование и наиболее предпочтительно в пользовательских программах. При проектировании программно-аппаратных комплексов на базе PC и при решении задач системного программирования необходимо уметь управлять процессами обмена, используя систему прерываний и непосредственное обращение к портам периферийных устройств. Поэтому в предлагаемых работах основное внимание уделяется изучению внутренних и внешних аппаратных прерываний, а функции BIOS и DOS используются по мере необходимости в качестве инструментальных средств.

3.3. Изучение принципов обработки внутренних прерываний

3.3.1. Цели работы

Целями работы являются приобретение знаний о типах внутренних прерываний и флаговой логике процессора, об источниках возникновения внутренних прерываний. Целью работы также фвляется формирование навыков использования механизма внутренних прерываний при решении практических задач (приемы перехвата прерываний и передачи управления соответствующим системным обработчикам).

3.3.2. Программа работы

1. Проанализируйте содержимое стандартного обработчика прерывания по ошибке операции деления. Т.е. проверьте, как реагирует операционная система, на переполнение при выполнении команды деления. Для этого составьте программу, содержащую деление, которое вызывает переполнение. Запустите программу несколько раз. Результат работы программы покажите преподавателю.

2. Замените стандартный обработчик **`int 0`** на собственный, используя функции DOS для замены вектора прерывания и функцию 9h прерывания **`int 021h`** (функция **`dos 9`**) для иллюстрации работы вашего обработчика. Рекомендуемый алгоритм приведен на рис. 3.4.

3. Преобразуйте вашу программу так, чтобы не применять функции DOS для получения и установки вектора прерываний **`int 0`**.

4. Организуйте передачу управления обработчику BIOS после выполнения всех команд вашего прерывания, т.е. для выхода из Вашего обработчика используйте команду безусловного перехода по адресу

системного обработчика. Перед завершением программы не забудьте восстановить вектор прерывания для обеспечения возможности повторного запуска программы.

5. Составьте программу инкрементации содержимого регистра BX. Дополните ее так, чтобы стала возможной трассировка этого фрагмента инкрементации (используйте функции DOS). Обработчик прерывания по T-биту должен содержать вывод любого символа (например *) для контроля шагов трассировки. Установку и сброс T-бита предусмотрите в основной программе. Рекомендуемый алгоритм приведен на рис. 3.5.

6. Преобразуйте программу так, чтобы сброс T-бита производился в прерывании **`int 1`**, используя алгоритм на рис. 3.6. Сброс T-бита выполнить по заданию преподавателя одним из трех способов:

- обращением в стек с помощью команды `mov`, используя содержимое указателя стека (SP), сохраненное перед вызовом прерывания,
- обращением в стек с помощью той же команды, используя содержимое указателя стека, сохраненное после входа в обработчик прерывания,
- обращением в стек с помощью команд `push/pop`.

7. Функционально завершите программу, используя вызовы прерываний по точке останова для ограничения трассируемого фрагмента программы. Для этого в программу обработки **`int 3`** введите необходимые функции управления T-битом. В качестве алгоритма используйте последовательность действий, показанную на рис. 3.7.

Для наглядности работы программы введите входной параметр, в зависимости от значения которого происходит ветвление в трассируемом фрагменте на программные ветви различной длины. Убедитесь в правильности результата.

В прерывании **`int 1`** предусмотрите вывод адреса (смещения) исполняемой команды в трассируемом фрагменте. Сравните полученные Вашей программой значения смещений со смещениями, которые можно наблюдать с помощью отладчика.

8. Результаты работы последней программы выведите на печать, используя прерывание **`int 5`** (печать копии экрана).

9. Исследуйте стандартные обработчики прерываний int4, int6, int7 (если это позволяет аппаратура). Перехватите указанные прерывания, позаботившись о восстановлении старых векторов (дополнительное задание).

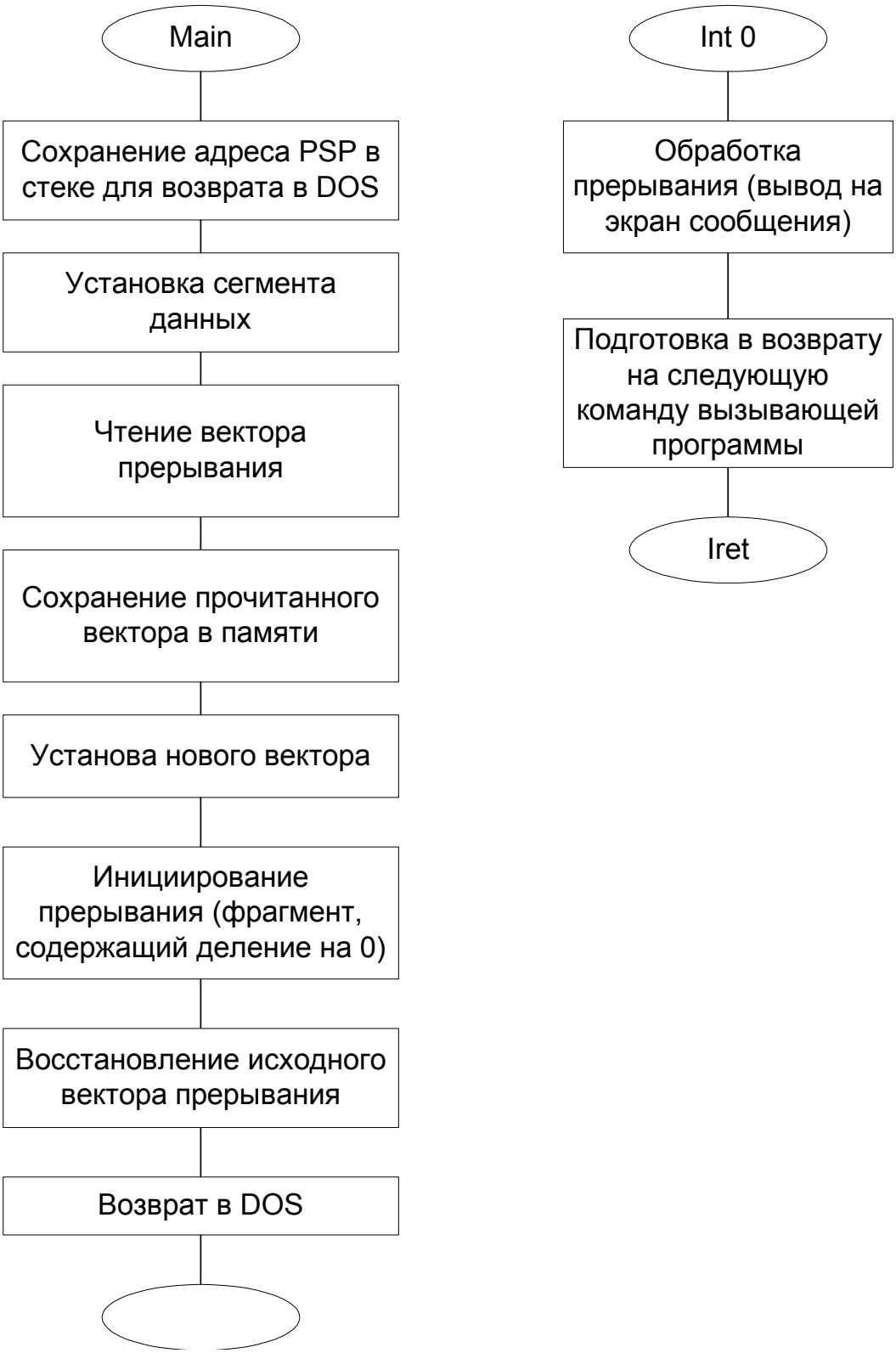

Рис. 3.4. Возможная схема алгоритма перехвата прерывания

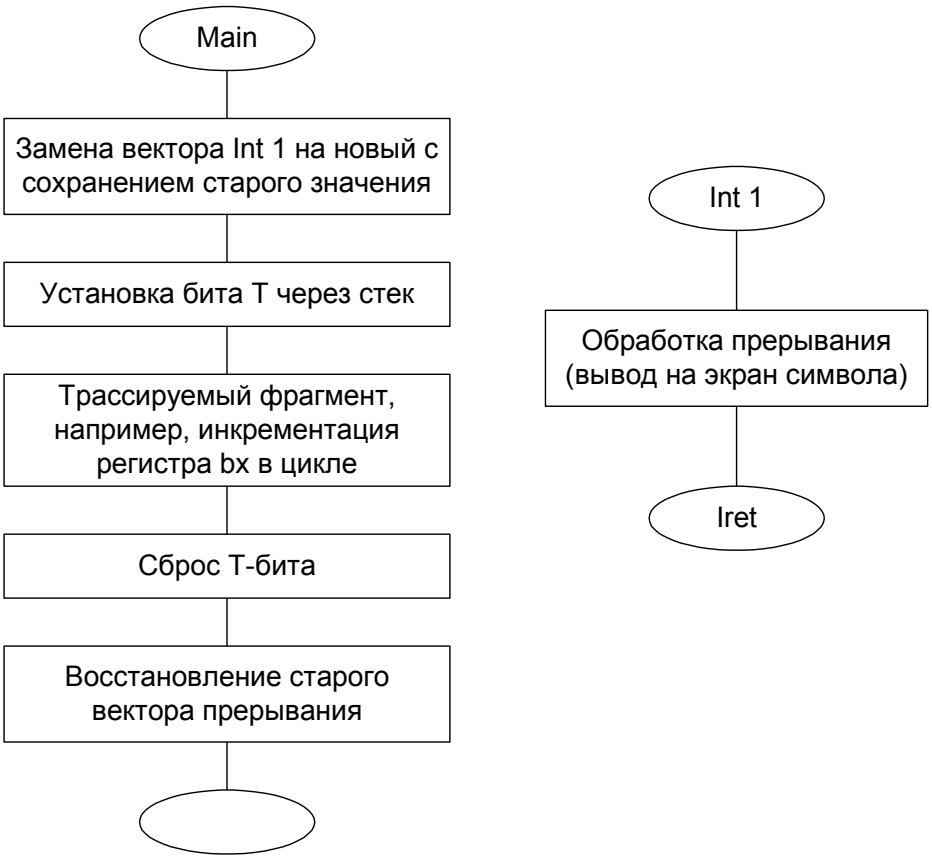

Рис. 3.5. Рекомендуемый алгоритм перехвата прерывания по биту трассировки

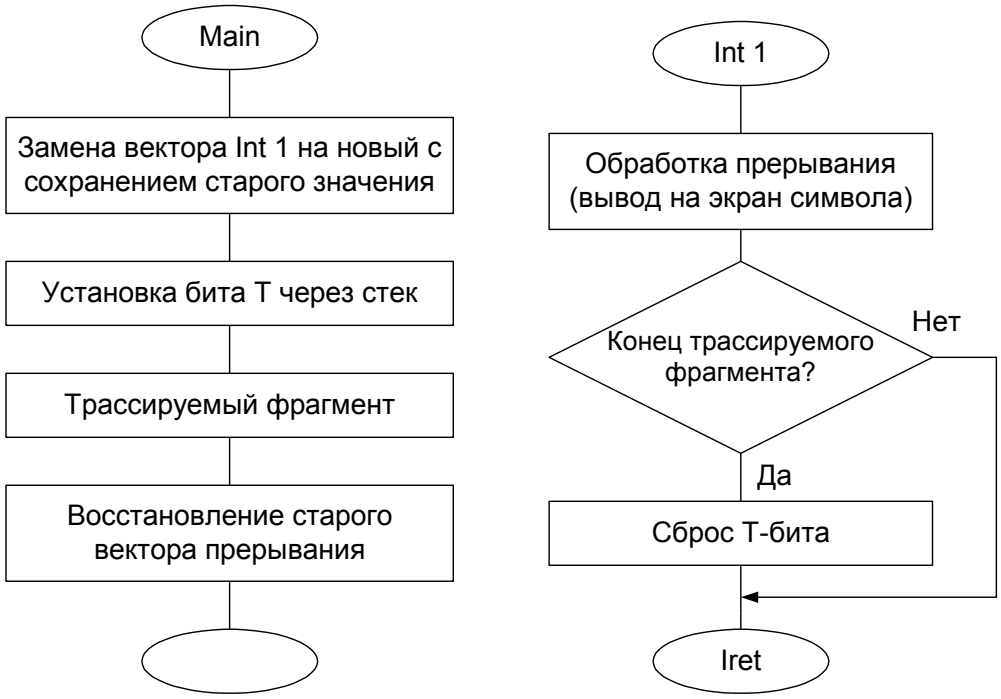

Рис. 3.6. Алгоритм трассировки со сбросом T-бита в обработчике

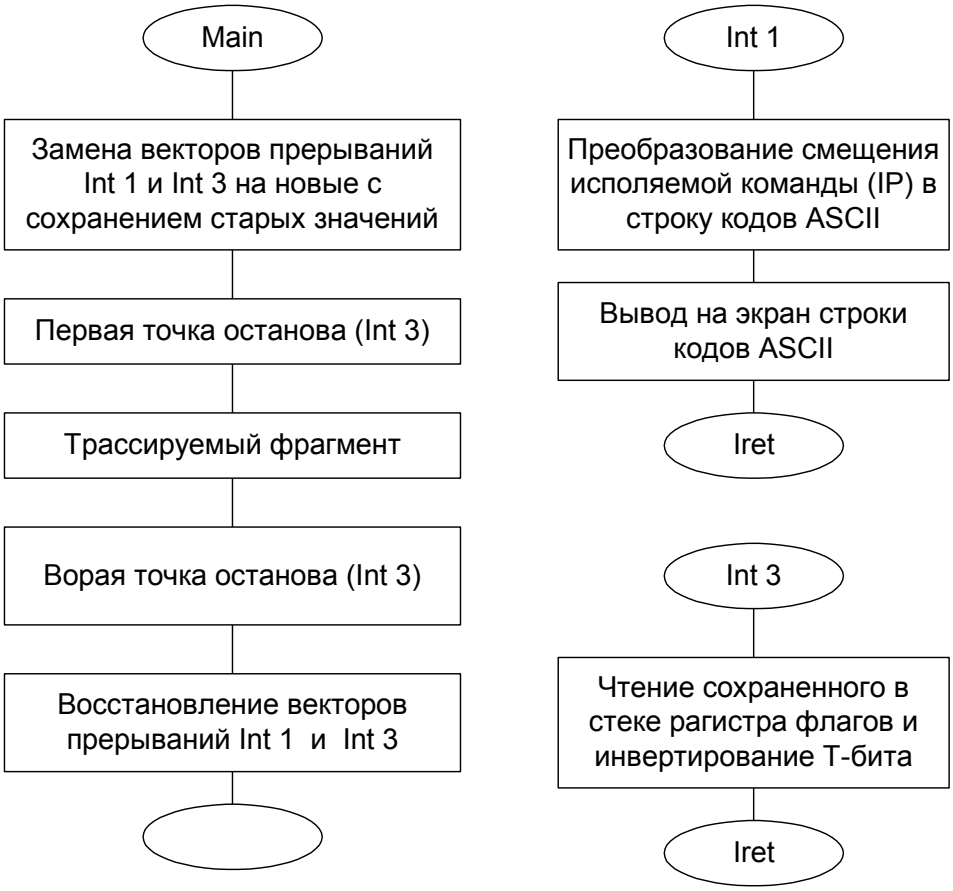

Рис. 3.7. Алгоритм трассировки с использованием точек останова для ограничения трассируемого фрагмента программы

3.3.3. Методические указания к работе

Под внутренние прерывания в таблице векторов прерываний отведены первые восемь позиций (с нулевой по седьмую) для реального режима работы процессора (табл. 3.2). Строго говоря, приведенные в этой таблице прерывания поддерживаются не любыми процессорами, которые использовались в персональных ЭВМ. Однако, к настоящему времени ранние модели ПЭВМ с процессорами i8086…i80286, в которых некоторые функции могли отсутствовать, практически вышли из употребления.

Вызов прерывания осуществляется путем использования операнда в качестве индекса в таблице векторов прерываний (или в таблице дескрипторов прерываний IDT). В реальном режиме IDT содержит 4-байтные указатели на процедуры обработки. При возникновении внутреннего прерывания процессор оставляет свою работу, сохраняет в стеке содержимое регистра флагов RF, значения сегмента CS и смещения IP, после чего процессор переходит к выполнению команд программы обработки прерывания по адресу, равному номеру прерывания, умноженному на 4 (см. пример настройки прерывания в программе п. 2.2.4). Вызов прерываний 3 и 5 может быть инициирован пользователем командой ассемблера **`int <n>`**, где n — номер (код) прерывания. После входа в прерывание структура стека имеет вид, показанный на рис. 3.8 (см. также рис. 1.4).

Таблица 3.2

Внутренние прерывания

Номер (код) прерывания	Функция прерывания
0	Деление на ноль
1	Пошаговая трассировка
2	Немаскируемое прерывание
3	Печать копии экрана
4	Арифметическое переполнение

5	Печать копии экрана
6	Недействительный код операции
7	Сопроцессор не доступен

На рис. 3.8 IP, CS и Flags — это содержимое соответствующих регистров процессора на момент ухода в программу обработки прерывания («точка возврата»), т.е. эти данные «принадлежат» прерванной программе. После выполнения обработки прерывания содержимое стека извлекается в обратном порядке (IP, CS, Flags). Возврат происходит к следующей команде прерванной программы. Отсюда понятно, в частности, как из программы обработки прерывания получить данные «точки возврата»: можно либо воспользоваться командами работы со стеком, либо командами пересылки данных, предварительно скопировав указатель стека (SP), чтобы потом использовать его как базу для непосредственного обращения к ячейкам.

Логические прерывания

Особенностью прерывания при делении на ноль (**`int 0`**) является передача управления в точку вызова в основной программе. Чтобы избежать "зависания", необходимо в программе обработки позаботиться о переходе на следующую команду, предварительно известив о неприятности пользователя.

Аналогично запрос на прерывание **`int 4`** — арифметическое переполнение — генерируется при установленных флагах OF и IF (переполнение и разрешение прерывания). Этот запрос инициируется командой условного прерывания при переполнении INTO. Обычно вектор этого прерывания является указателем на команду IRET, тем самым пользователю предоставляется возможность самостоятельно программировать реакцию системы в случае предполагаемой возможности возникновения переполнения.

Чтобы проанализировать работу стандартного обработчика подобных прерываний (например, деления на ноль) достаточно написать программу, создающую ситуацию, при которой это прерывание будет инициировано.

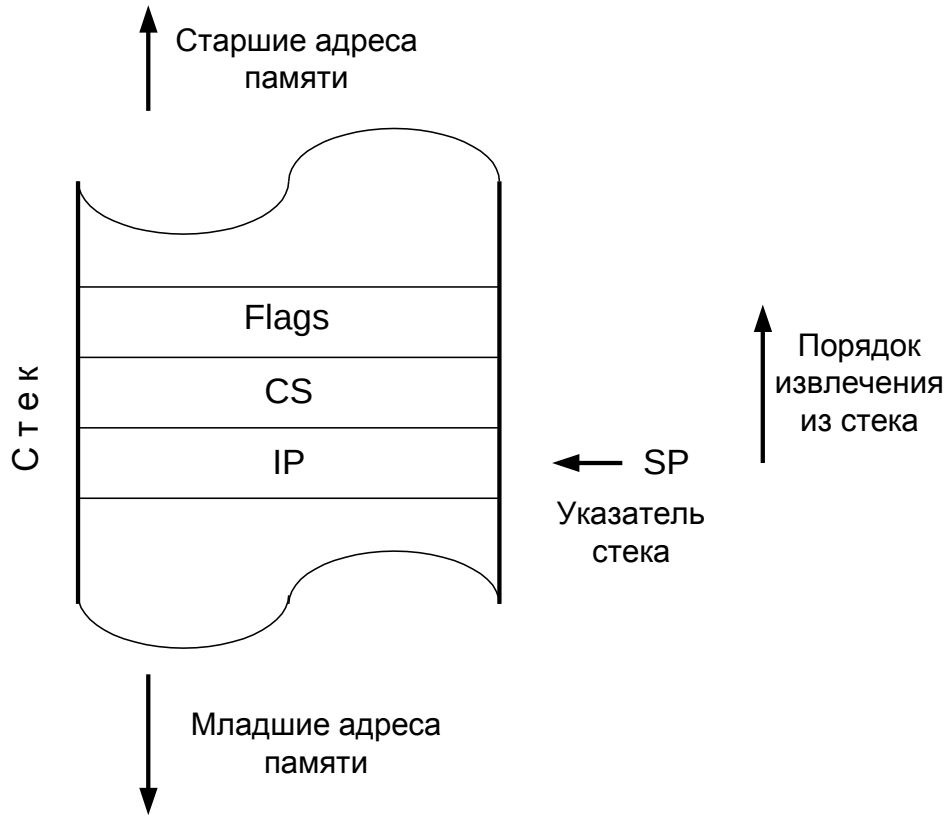

Рис. 3.8. Содержимое вершины стека после входа в прерывание

При выполнении заданий, связанных с перехватом логических прерываний, для устранения возврата программы в точку вызова прерывания следует «исправить» адрес возврата из прерывания, воспользовавшись одним из указанных выше приемом.

Отладочные прерывания

Прерывания **`int 1`** и **`int 3`** позволяют организовать отладочные режимы пошаговой трассировки и обработки точек останова, что необходимо для создания различного рода отладчиков. Прерывание **`int 1`** происходит при установленном бите T в регистре флагов. Процессор, выполнив одну команду, переходит на программу обработки прерывания int 1, выполняет команды обработчика, после чего приступает к выполнению следующей команды. Установить и сбросить T-бит можно непосредственно в теле трассируемой программы. При этом необходимо установить начало и проанализировать условие окончания трассируемого фрагмента. Трассировка другого фрагмента программы потребует редактирования исходного программного кода. Поэтому целесообразно применять точки останова, используя прерывание **`int 3`**, и возложить на него функции управления T-

битом. Один из возможных вариантов построения программы для пошагового исполнения приведен на рис. 3.7.

Трассируемый фрагмент помещается между двумя точками останова (**int 3**). Программа обработки третьего прерывания анализирует состояние Т-бита: если он не установлен, то устанавливает его посредством записи соответствующего слова в стек по адресу сохраняемого регистра флагов вызывающей программы. Сброс происходит аналогичным образом в случае установленного Т-бита при повторном вызове **int 3**.

В операционной системе DOS прерывания **int 1** и **int 3** не поддерживаются, программы их обработки содержат только команду возврата **iret**.

Немаскируемое прерывание вырабатывается на системной плате контроллером системной шины в случае возникновения ошибки четности при обращении к памяти или к устройству на шине. При этом сигнал запроса подается на соответствующий вход микропроцессора (NMI). О немаскируемых прерываниях см. также п. 1.4.1 настоящего пособия.

Прерывания **int 6** и **int 7** не используются в МП 8086/88. Прерывание **int 7** генерируется при обращении к сопроцессору, присутствие которого предусмотрено не во всех конфигурациях ВС. Невозможность идентифицировать команду процессора приводит к возникновению прерывания **int 6**.

Правила замены стандартного обработчика прерывания

Если действия программы обработки прерывания не устраивают, то ее можно заменить собственной, перенастроив вектор прерывания. Для этого необходимо выполнить следующие действия:

- получить текущее значение вектора прерывания,
- сохранить старое значение вектора в заранее зарезервированных ячейках сегмента данных,
- установить новое значение вектора,
- написать основную программу, инициирующую замененное прерывание
- восстановить старое значение вектора для дальнейшей бесконфликтной работы. В противном случае последующая программа может вызвать данное прерывание и передать управление на то место в памяти, где вашей программы уже нет,
- написать собственную программу обработки прерывания,
- в случае необходимости передать управление стандартному обработчику. Это целесообразно сделать, если собственный обработчик функционально дополняет имеющийся в системе.

З а м е ч а н и е:

Выполнить чтение или запись вектора прерывания можно двумя способами: либо по номеру прерывания вычислить адрес ячейки памяти в области IDT и выполнить прямое обращение для чтение или записи (см. пример установки вектора прерывания в п. 2.2), либо воспользоваться функциями DOS прерывания `int 021h`. Для чтения вектора из IDT применяется функция 035h, а для записи используется функция 025h прерывания `int 021h`. Параметры функций 35h и 25h приведены в табл. П.2. приложения. Функция 25h автоматически запрещает аппаратные прерывания в процессе изменения вектора.

В [13] приведен ряд примеров чтения/записи векторов прерывания. Ниже приведен эквивалент функции 035h прерывания `int 021h` операционной системы DOS (чтение вектора прерывания, номер которого должен быть предварительно помещен в регистр `al`):

```
mov di, al; номер прерывания – в DI
xor ax, ax   ; устанавливаем ES в начало ОЗУ
mov es, ax
shl di, 2    ; умножаем номер прерывания на 4
mov bx, es:[di] ; помещаем в BX смещение
                ; обработчика прерывания
mov ax, es:[di]+2 ; помещаем в ES сегментный адрес
                ; обработчика прерывания
mov es, ax
```

В результате полный адрес обработчика прерывания с номером из регистра `al` окажется в регистрах `es:bx`, как это делает функция 035h DOS.

Аналогичным образом осуществляется установка нового вектора прерывания. Эквивалентная функции 025h прерывания `int 021h` DOS.

Контрольные вопросы

1. Как вычислить адрес вектора прерывания, если известен номер данного прерывания?
2. Объясните назначение бита T в регистре флагов процессора.
3. Можно ли замаскировать внутренние прерывания?
4. Каковы действия процессора при входе и возврате из прерывания? Сбрасываются ли флаги T и I при переходе к программе обработки внутреннего прерывания?

5. Что означает "перехватить прерывание"?

6. Зачем и как осуществляется передача управления системному обработчику?

3.4. Изучение принципов обработки аппаратных прерываний

3.4.1. Цели работы

Целями работы являются знание логики взаимодействия процессора и контроллера прерываний, способов маскирования аппаратных прерываний, системы приоритетов, приобретение навыков использования стандартных средств для обслуживания аппаратных прерываний, умение создавать собственные и дополнять уже существующие программы обработки прерываний.

3.4.2. Программа работы

В процессе подготовки к выполнению работы написать и отладить следующие процедуры:

- вывода на экран строки символов прямым отображением в память, так как при использовании функций DOS или BIOS аппаратные прерывания, возникающие в случайные моменты времени, могут привести к разрушению операционной системы [13],
- преобразования байта информации, хранящегося в ОЗУ, в десятичный эквивалент в символьном виде.

Выполните следующие эксперименты по перехвату прерываний:

- Перехватите прерывание `int 8h`, предусмотрев в программе обработки передачу управления исходному обработчику. Для иллюстрации результата выведите в левом верхнем углу дисплея звездочку прямым отображением в память.
- Повторите эксперимент без передачи управления. Не забудьте снять заявку в регистре ISR контроллера прерываний.
- Замените в обработчике прямое отображение в видеопамять на процедуру инкрементации счетчика прерываний от таймера так, чтобы по истечении одной секунды происходил вызов пользовательского прерывания `int 060h`. Перед возвратом из обработчика `int 8`

передайте управление исходному обработчику, чтобы не сбивать системные часы. Для иллюстрации результата в обработчик `int 60h` включите вывод на экран символа.

- Убедившись, что перехватывание аппаратного прерывания и вновь созданное собственное прерывание происходят успешно, замените процедуру вывода в `int 60h` на подсчет относительного времени с момента начала работы программы.

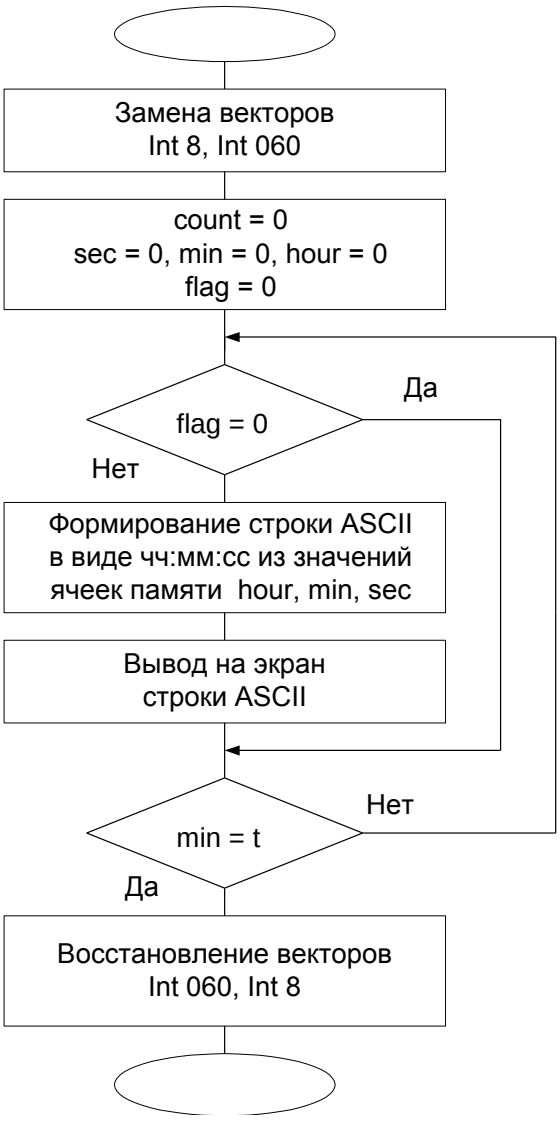

Рис. 3.9. Рекомендуемый алгоритм основной программы вывода времени работы программы

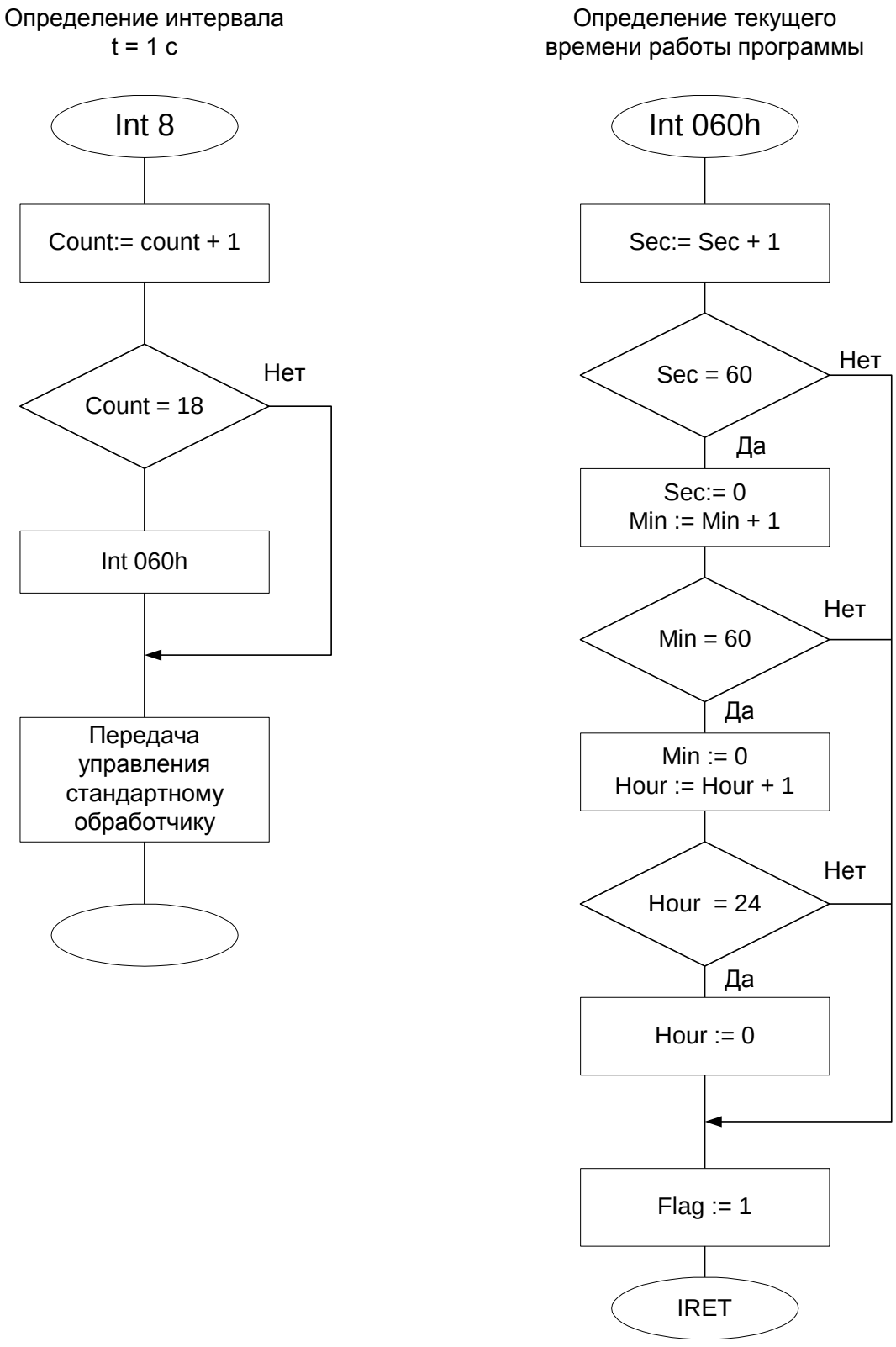

Рис. 3.10. Рекомендуемый алгоритм счета времени

- Напишите программу вывода на экран времени работы вашей программы. Для этого в основную программу включите цикл, ограничивающий время работы. Функции коррекции времени удобнее выполнять в пользовательском прерывании `int 60h`, а функции преобразования в символьный вид и вывод на экран — в теле основной программы. Окончание цикла можно установить по значению относительного времени работы программы. Возможный алгоритм решения этой задачи приведен на рисунках 3.9, 3.10
- Используя свою программу, замаскируйте доступ к клавиатуре на 1 мин. Попробуйте "перезапустить" компьютер с "теплого старта".
- В основной программе замените функцию 025h прерывания DOS 021h установки вектора прерывания. Позаботьтесь о том, чтобы во время смены вектора аппаратного прерывания не произошло другое аппаратное прерывание.
- Исключите цикл, ограничивающий время работы вашей программы. Реализуйте возможность выхода из программы по "горячим клавишам". Для этого перехватите аппаратное прерывание от клавиатуры.
- Преобразуйте программу так, чтобы на экран выводились часы суточного времени. Для этого в начале основной программы примените вызов функции 02CH прерывания DOS 021h.
- Выполните индивидуальное задание, полученное у преподавателя.

3.4.3. Методические указания к работе

Режим прерываний в общей системе обмена информацией с периферийными устройствами

Обмен с ПУ может осуществляться двумя способами: по прерываниям и по прямому доступу к памяти (ПДП). Как правило, ПДП применяется для поблочной передачи информации между ОЗУ и ВЗУ. Устройство, принимающее на себя функции управления процессом обмена — контроллер ПДП. Прямой доступ позволяет на время обмена разгрузить процессор, освободив его от управления передачей, и существенно повысить как скорость обмена, так и быстродействие процесса в целом.

ПУ, отличные от ВЗУ побайтно обмениваются либо с процессором, либо с памятью (ОЗУ). Очевидно, что целесообразно применять единый механизм обмена и в том, и в другом случае во избежание излишних аппаратных затрат и для унификации программного обеспечения. Побайтный обмен позволяет обеспечить гибкость при распределении ресурсов вычислительной системы с целью повышения ее быстродействия за счет

распараллеливания работы периферийных устройств с различной скоростью. Процесс управления обменом в режиме прерываний осуществляет программируемый контроллер прерываний (ПКП) при участии процессора. ПКП обрабатывает запросы, поступающие от ПУ в соответствии с предусмотренной в нем схемой приоритетов и маскирования, формирует сигнал INTR, поступающий на соответствующий вход процессора, и выставляет на шину адрес вектора прерывания, рекомендуемого им к обработке.

Сведения о контроллере прерываний

В дополнение к описанию, приведенному в п.1.4 укажем, что логика работы программируемого контроллера прерываний, построенного на БИС Intel 8259, принята за основу при создании современных устройств подобного типа. Это обеспечивает совместимость выпускаемых сегодня компьютеров с более ранними модификациями.

Программируемый контроллер прерываний Intel 8259 поддерживает восемь уровней приоритетов. Это достаточно для конфигурации персонального компьютера типа XT. Для AT с шестнадцатью уровнями приоритетов используются две такие БИС, соединенные каскадно. Таким образом, ведущий контроллер обрабатывает первые восемь аппаратных прерываний от IRQ0 до IRQ7 включительно, а ведомый следующие прерывания от IRQ8 до IRQ15. Приоритеты между ними распределены так, что уровни приоритетов ведомого контроллера находятся между вторым и третьим уровнями ведущего (см. п.1.4).

КП на основе БИС 8259 имеет три однобайтовых регистра, которые управляют восемью линиями аппаратных прерываний:

IRR — регистр запроса на прерывание, устанавливает соответствую щий бит, когда линия прерывания сигнализирует о запросе;

ISR — регистр обслуживания прерывания, содержит информацию о том, обрабатывается ли в данный момент другое прерывание и контролирует цепь приоритетов;

IMR — регистр маски прерывания служит для хранения, разрешения и запрета (маскирования) прерываний.

Как правило, программисты обращаются только к регистру IMR через порт 21h и к командному регистру или регистру обслуживания прерывания ISR через порт 20h.

Маскирование аппаратных прерываний

Регистр IMR позволяет программно запретить или разрешить прерывания от периферийных устройств (поэтому их иначе называют маскируемыми). Маскирование может быть полным, когда блокируются все прерывания, и частичным, когда маскируются только определенные аппаратные прерывания.

Полное маскирование необходимо в случаях, когда критический участок программы должен быть выполнен целиком прежде, чем компьютер выполнит какое-либо другое действие. Например, при изменении вектора аппаратного прерывания, чтобы избежать выполнения прерывания по не полностью измененному вектору.

Частичное маскирование может потребоваться, когда некоторые прерывания могут взаимодействовать с операциями критичными к временным интервалам. Например, точно рассчитанная процедура ввода/вывода не должна быть прервана длительным дисковым прерыванием.

Следовательно, дисковые прерывания необходимо временно запретить. Такое маскирование часто используют в задачах реального времени.

Регистр IMR обычно используют для маскирования отдельных аппаратных прерываний. Для этого используется вывод в порт с адресом 21h ведущего контроллера прерываний и в порт с адресом A1h ведомого контроллера прерываний, позволяющего маскировать прерывания IRQ8 - IRQ15. По указанному порту устанавливаются в "1" те биты регистра, которые соответствуют номерам запрещаемых вами прерываний.

IMR доступен только по записи. В конце программы его необходимо очистить, иначе обращение к замаскированным вами устройствам будет невозможно и после завершения вашей программы.

```
; пример маскирования запросов от КНМГД
mov   al,    01000000b  ; маскирование бита 6
out   21h,   al         ; запись в IMR
    * * *
mov   al,    0          ; очистка IMR в конце
out   21h,   al         ; программы
```

Чтобы провести полное блокирование аппаратных прерываний обычно используют специальные команды управления битом I в регистре флагов процессора RF: cli (clear interrupt) и sti (set interrupt). Когда флаг I равен 1 - аппаратные прерывания запрещены, когда он равен 0 - разрешены все прерывания, разрешенные в IMR.

Таким образом, управление от процессора более приоритетно по сравнению с управлением от контроллера прерываний .

Для очиски (обнуления) I-флага применяют команду STI, а команду CLI для записи в RF 1. При использовании этих команд необходимо соблюдать следующие правила:

- нельзя отключать прерывания на длительный период, так как это влечет за собой нарушение системного времени;
- за командой CLI всегда должна следовать команда STI, иначе не избежно "зависание" компьютера из-за блокировки клавиатуры;
- при создании своих программных прерываний начинайте программу обработки с команды STI, если аппаратные прерывания допустимы.

Библиографический список

1. **Авдюхин А.А., Жуков А.В.** Интерфейсы париферийных устройств ЭВМ: Учеб. пособие. СПб.: Изд-во СПбГПУ, 2003.— 115 с.
2. **Душутина Е.В.** Организация обмена информацией в режиме прерываний в ЭВМ и ВС на основе микропроцессоров Intel 80x86: Учеб. пособие/ СПб. гос. техн. ун-т. СПб.: 1996. 68 с.
3. **Мячев А.А., Степанов В.Н., Щербо В.К.** Интерфейсы систем обработки данных: Справочник/Под ред. А.А.Мячева. М.: Радио и связь, 1989. 416 с.
4. **Скэнлон Л.** Персональные ЭВМ IBM PC и XT. Программирование на языке ассемблера: Пер. с англ. М.: Радио и связь, 1991. 336 с.
5. **Гук М.** Аппаратные средства IBM PC. Энциклопедия. СПб.: Питер, 2000. 816 с.
6. **Новиков Ю.В., Калашников О.А., Гуляев С.Э.** Разработка устройств сопряжения для персонального компьютера типа IBM PC: Практ. пособие/Под общ. ред. Ю.В.Новикова. М.: ЭКОМ, 1997. 224 с.
7. **Блохнин С.М.** Шина ISA персональных компьютеров IBM PC/AT. М.: Сплайн, 1992 г. 78 с.
8. **Юров В., Хорошенко С.** Assembler: Учебный курс. СПб.: Питер Ком, 1999. 672 с.
9. **Фролов А.В., Фролов Г.В.** Аппаратное обеспечение персонального компьютера. М.: ДИАЛОГ-МИФИ, 1998. 304 с. (Б-ка системного программиста Т. 33).
10. Руководство по архитектуре IBM PC AT / Ж.К. Голенкова, А.В. Заболоцкий, М.Л. Мархасин и др.; Под общ. ред. М.Л.Мархасина. Минск: ООО Консул, 1992. 949 с.
11. **Джордейн Р.** Справочник программиста персональных компьютеров типа IBM PC, XT и AT: Пер. с англ. М.: Финансы и статистика, 1992. 544 с.
12. **Скотт Мюллер.** Модернизация и ремонт персональных компьютеров/ Пер. с англ. М.: Восточная Книжная Компания, 1996 . 896 с.
13. **Жуков А.В., Авдюхин А.А.** Ассемблер. СПб.: БХВ-Петербург, 2003. 448 с.
14. **Гук М.** Аппаратные интерфейсы ПК. Энциклопедия.— СПб.: Питер, 2002.— 528 с

ОГЛАВЛЕНИЕ

АВДЮХИН Андрей Андреевич
Душутина Елена Владимировна
ЖУКОВ Андрей Владимирович

Аппаратно-программные средства ввода/вывода

Учебное пособие

Подписано в печать Формат 60×84/16

Усл. печ. л. Уч.-изд.л. Тираж ___ экз.

Заказ

Санкт-Петербургский государственный политехнический университет.