Министерство образования Республики Беларусь
БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра «Робототехнические системы»

УПРАВЛЕНИЕ ВНЕШНИМИ УСТРОЙСТВАМИ МИКРОЭВМ

Методическое пособие
по дисциплинам
"Проектирование микропроцессорных систем управления"
и "Системы управления технологическим оборудованием"
для студентов специальностей
1-53 01 01 "Автоматизация технологических процессов
и производств (в приборостроении и электронике)",
1-53 01 02 "Автоматизированные системы обработки информации"
и 1-53 01 06 "Промышленные роботы и робототехнические комплексы"

В 2 частях

Ч а с т ь 1

М и н с к 2 0 0 4

УДК 681.327 (076.5)
ББК 32.973
М 54

Авторы:
А.А. Москаленко, З.И. Кононенко, А.Н. Дербан,
Ю.Н. Позник

Рецензенты:
В.Б. Ковалевский, В.П. Загорский

Москаленко А.А.

М 54 Управление внешними устройствами микроЭВМ: Метод. пособие по дисц. "Проектирование микропроцессорных систем управления" и "Системы управления технологическим оборудованием" для студ. спец. 1-53 01 01 "Автоматизация технологических процессов и производств (в приборостроении и электронике)", 1-53 01 02 "Автоматизированные системы обработки информации" и 1-53 01 06 "Промышленные роботы и робототехнические комплексы." В 2 ч. Ч. 1. / А.А. Москаленко, З.И. Кононенко, А.Н. Дербан, Ю.Н. Позник. – Мн.: БНТУ, 2004. – 108 с.

ISBN 985-479-089-4.

Методическое пособие предназначено для обучения студентов внутренней структуре микроЭВМ и управляющих ЭВМ, а также принципам обмена информацией и управления внешними устройствами.

В работе рассмотрены структурные схемы микроЭВМ двух наиболее распространенных в Республике Беларусь и странах СНГ микроЭВМ систем команд DEC и INTEL, их особенности при обращении к оперативному запоминающему устройству и внешним устройствам, принципы обмена информацией между блоками микроЭВМ и ее внешними устройствами, включая устройства пользователя.

Пособие представляет собой руководство к выполнению лабораторных работ по указанным курсам. В каждой работе приводятся краткое описание необходимого материала, метода, алгоритм управления и задание. Язык программирования для разработки управляющих программ выбирается студентом.

Пособие может быть полезно инженерам, аспирантам и преподавателям, занимающимся управлением на базе микропроцессорной техники.

УДК 681.327 (076.5)
ББК 32.973

ISBN 985-479-089-4

Введение

Появление микропроцессорной техники (микропроцессоров, микропроцессорных комплектов и секций, микроЭВМ и микроконтроллеров) было обусловлено, прежде всего, необходимостью автоматизации технологических процессов и производств и потребностью в высоконадежных средствах вычислительной техники для этой цели.

Проблема повышения надежности ЭВМ встала особенно остро после первых попыток внедрения больших ЭВМ второго поколения для комплексной автоматизации (контроля, сигнализации, отображения информации, реализации алгоритмов логики и дискретного управления, цифрового непосредственного управления) на сложных технологических обьектах – энергоблоках электростанций (в 1959 г. – в США, в начале 1960-х годов – в СССР). Однако, несмотря на огромные затраты, ни одна система не была внедрена на этих объектах в полном обьеме. Главной причиной неудачных внедрений ЭВМ была их низкая надежность. В дальнейшем опыт показал, что легче поддаются внедрению системы централизованного контроля, децентрализованные системы управления на базе специализированных цифровых вычислительных устройств, локальные информационно-управляющие системы. Другое направление автоматизации технологических процессов связано с использованием управляющих вычислительных комплексов (УВК) на базе более надежных транзисторных миниЭВМ, а затем – ЭВМ, выполненных на основе микросхем низкой степени интеграции.

Ни одно из направлений, сложившихся к концу 1970-х годов, не позволяло перейти к комплексной автоматизации технологических процессов. Для решения этой проблемы требовалось значительно повысить надежность элементной основы ЭВМ.

Революционным прорывом в этой области явился 1971 год, когда американская фирма Intel разработала первый микропроцессор, ознаменовавший новый период в развитии вычислительной техники и систем управления на их основе.

Уже в 1973 году фирма Control Logic разработала первую микроЭВМ для целей управления. В 1974 году была создана первая микроЭВМ с микропрограммным управлением, а в 1975 году – первая персональная микроЭВМ (ПЭВМ).

Получила широкое признание концепция полностью распределенных систем управления, локальных вычислительных сетей и интегрированных компьютерных производств.

Бурное развитие микропроцессорной техники и систем на ее основе стало возможным благодаря ее основным особенностям – высокой надежности, относительно невысокой стоимости и малым габаритам. Использование микропроцессорной техники в системах управления позволило получить целый ряд дополнительных преимуществ, реализовать не только цифровые микропроцессорные регуляторы, но и адаптивные регуляторы, реализующие оптимальные алгоритмы управления.

Внедрение микропроцессорных систем управления потребовало от специалистов дополнительных аппаратных и программных решений – разработки устройств сопряжения с объектом, выбора и реализации метода гальванической развязки, использования витой пары проводников, разделения сигнальных и силовых проводников, экранирования, аналоговой фильтрации, выбора и обоснования метода цифровой фильтрации, разработки алгоритмов управления и др.

Кроме того, использование микропроцессорной техники в системах управления требует от разработчиков дополнительных знаний как об аппаратных, так и о программных особенностях, связанных с управлением внешними устройствами и устройствами пользователя, что не обязательно знать обычному пользователю микроЭВМ при решении математических задач или работе в режиме использования специализированных и универсальных пакетов.

В данном методическом пособии описываются основные принципы управления внешними устройствами для двух широко распространенных в нашей стране микроЭВМ системы команд DEC и INTEL. Предложенные для самостоятельного выполнения задания направлены на закрепление теории и приобретения практических навыков разработки управляющих программ пользователя.

1. ПРИНЦИПЫ УПРАВЛЕНИЯ ВНЕШНИМИ УСТРОЙСТВАМИ МИКРОЭВМ

1.1. Понятия модульности, интерфейса и магистрали

Любая микроЭВМ и микропроцессорная система имеют ***модульный принцип построения***, при котором все функциональные блоки выполняются в виде конструктивно законченных устройств (***плат-модулей***). Принцип модульности является одной из современных особенностей архитектурного и структурного построения микро- и мини-ЭВМ. Он распространяется на аппаратную (техническую) и программную части системы. Модульность предусматривается на всех иерархических уровнях семейства узлов определенного назначения для вычислительных, контролирующих и управляющих систем.

Другим немаловажным принципом построения микроЭВМ и систем на их основе является ***магистральный способ обмена информацией***.

Связь функционально законченных модулей в микроЭВМ и микропроцессорной системе осуществляется с помощью электрических проводников, называемых ***линиями***. Линии, сгруппированные по некоторому признаку или назначению, обьединяются в ***шины*** (шина данных ШД, адресная шина АШ, шина управления ШУ). Совокупность шин, служащих для обмена информацией между компонентами системы, образует ***магистраль интерфейса*** (или просто ***магистраль***).

Термин Interface в переводе с английского означает "сопряжение". Он имеет два значения:

1) совокупность методов и средств сопряжения любых двух обьектов;

2) средство стандартного сопряжения периферийных устройств – ПУ (печатающего устройства, пультового терминала, накопителей на магнитных дисках и др.).

Взаимодействие между модулями, подключенными к магистрали, осуществляется путем изменения уровней напряжения на линиях. Магистральный способ организации связи является современным в отличие от принципа "каждый элемент или модуль с каждым".

Различают одно-, двух-, трех- и многомагистральные связи. В микроЭВМ и микропроцессорных системах обычно выделяются 3 основные магистрали – ШД, ША и ШУ. В микроЭВМ системы ко-

манд INTEL используются все эти магистрали, а в микроЭВМ системы DEC – только 2, так как для передачи данных и адресов используется одна и та же магистраль из 16 линий адресов/данных. Таким образом, магистраль используется ***в режиме мультиплексирования*** (разделения во времени функционального назначения магистрали). Магистральный принцип организации обмена информацией присутствует на всех уровнях иерархии микропроцессор-микроЭВМ-система. Периферийные устройства имеют существенно меньшее быстродействие по сравнению с основной памятью микроЭВМ (оперативной, постоянной или комбинацией обоих видов), что позволяет организовать параллельную работу подобных ПУ через один и тот же канал за счет разделения их работы во времени, то есть за счет мультиплексирования канала (называемого ***мультиплексным***).

1.2. Каналы и интерфейсы

В общем случае ***системы ввода-вывода*** (СВВ) характеризуются разделением управления вводом-выводом по следующим основным уровням иерархии: центральный процессор (ЦП) – канал – контроллер ввода-вывода (КВВ) – периферийное устройство (ПУ).

Процессор в (общем случае) формирует команды ввода-вывода. Канал получает необходимую информацию о команде ввода-вывода от ЦП, связывается с КВВ, выбирает из основной памяти (ОП) команды канала, дешифрирует и выполняет их. КВВ осуществляет логическое подключение к каналу запрашиваемого ПУ; передает в ПУ приказы, преобразовывает форматы данных ПУ в формат стандартного интерфейса ввода-вывода; формирует запросы в ПУ в соответствии с его приоритетом на передачу данных через интерфейс ввода-вывода; управляет последовательностью функционирования ПУ; определяет его готовность для связи с каналом. ПУ выполняет различные приказы КВВ и вспомогательные функции.

Итак, наиболее общие функции управления операциями ввода-вывода, которые не зависят от вида ПУ, выполняются унифицированными средствами СВВ, называемыми каналами. ***Канал ввода-вывода*** – это системное устройство, которое осуществляет функции передачи данных между ОП и периферийными устройствами или между ПУ.

Специфические функции для данного типа ПУ реализуются специализированными блоками (устройствами) управления ПУ, которые в зависимости от класса и типа микроЭВМ называются контроллерами ввода-вывода, устройствами сопряжения или интерфейсными модулями.

Современные микроЭВМ в общем случае имеют или должны иметь унифицированное (стандартное) сопряжение и схемы, обеспечивающие подключение ПУ к устройствам управления, которые подключаются к каналу через другие унифицированные схемы.

Совокупность унифицированных шин, элементов физического и логического соединения, а также алгоритмов управления обменом информацией между собой образует сопряжения, или ***интерфейс*** между компонентами СВВ. При этом необходимо различать ***интерфейс ввода-вывода*** (сопряжение между каналом и КВВ) и ***малые интерфейсы*** ПУ (сопряжения между КВВ и ПУ). Интерфейс ввода-вывода (ИВВ) является наиболее важным компонентом в СВВ, так как его параметры существенно влияют на основные характеристики СВВ, в частности, на пропускную способность всех каналов ввода-вывода.

Разделение СВВ по уровням иерархии для реализации процесса ввода-вывода обеспечивает гибкость программного управления вводом-выводом благодаря возможной параллельной работе ЦП, каналов, КВВ и нескольких ПУ.

Следует иметь в виду, что каналы могут совмещаться с КВВ. В настоящее время в микроЭВМ используются программируемые микроконтроллеры прямого доступа в память, что способствует значительному повышению быстродействия и эффективности работы микроЭВМ и микропроцессорной системы.

Конструктивно канал микроЭВМ выполняется в виде печатной платы, обеспечивающей необходимые электрические соединения между контактами розеток, к которым подключаются различные устройства. Кроме унифицированных шин в канал входят линии питания и шины аналоговых и дискретных сигналов первичной информации.

1.3. Микропроцессорные системы контроля и управления на базе микроЭВМ

МикроЭВМ широко используются ***в микропроцессорных системах контроля и управления*** во всех областях народного хозяй-

ства и, в частности, в системах управления роботов, робототехнических комплексов и систем.

Для подключения любой системы контроля и/или управления к микроЭВМ необходимо выполнение двух основных условий:

1) наличие свободного платоместа в микроЭВМ или умение подключиться к каналу микроЭВМ при его отсутствии;

2) наличие готового или разработанного пользователем устройства сопряжения микроЭВМ с объектом контроля и/или управления; при этом устройство сопряжения должно быть разработано с учетом специфики канала, требований и рекомендаций, которые обычно приводятся в техническом описании конкретной микроЭВМ, а также особенностей самой системы.

Системы контроля и/или управления принято называть ***устройствами пользователя***.

1.4. Внешние устройства микроЭВМ

Периферийные устройства микроЭВМ и устройства пользователя принято называть ***внешними устройствами*** (ВУ). Все внешние устройства микроЭВМ имеют аналогичное управление. Кроме того, связь между отдельными компонентами системы, включая ЦП (или собственно микроЭВМ), осуществляется через канал одинаково, и внешние устройства так же легко доступны для ЦП, как и оперативное запоминающее устройство (хотя скорость обращения может сильно отличаться).

Поэтому принципы управления внешними устройствами микроЭВМ могут быть рассмотрены на примере управления любым периферийным устройством микроЭВМ. Рассмотрим их на примере микроЭВМ "Электроника ДВК-3М2" (ДВК-2, ДВК-3).

1.5. Структурная схема микроЭВМ "Электроника ДВК-3М2"

Структурная схема микроЭВМ "Электроника ДВК-3М2" изображена на рис. 1.1. Она отражает все вышеизложенные особенности построения микроЭВМ и системы контроля и/или управления.

МикроЭВМ "Электроника ДВК-3М2" относится к семейству микроЭВМ системы команд DEC ("Электроника 60" (60М, 60Т),

семейство микроЭВМ ДВК, "Электроника 85" и др.). МикроЭВМ выполнена на базе микропроцессорных комплексов К1801, имеющих довольно высокую степень интеграции, что позволило реализовать в конструкции микроЭВМ "Электроника 60М" более мощную микроЭВМ.

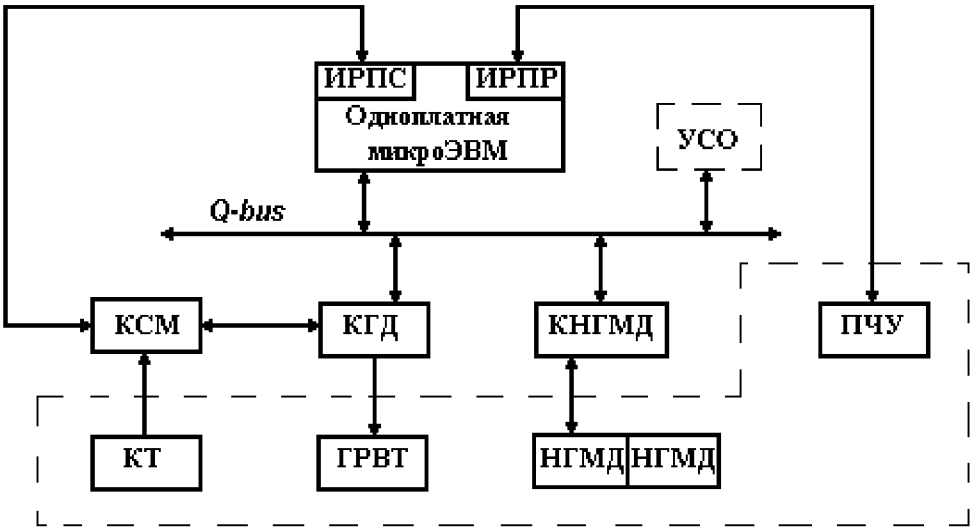


Рис. 1.1. Структурная схема микроЭВМ "Электроника ДВК-3М2"

Здесь:

УСО – устройство сопряжения с объектом контроля или управления;

ИРПС – интерфейс радиально-последовательный;

ИРПР – интерфейс радиально-параллельный;

КСМ – контроллер символьного монитора;

КГД – контроллер графического дисплея;

КНГМД – контроллер накопителя на гибком магнитном диске;

КТ – клавиатурный терминал;

ГРВТ – графический видеотерминал;

НГМД – накопитель на гибком магнитном диске;

ПЧУ – печатающее устройство.

Как показано на рис. 1.1, собственно микроЭВМ (центральный процессор, оперативное запоминающее устройство (ОЗУ), графическое ОЗУ, таймер, генератор тактовых импульсов и интерфейсы) выполнена в виде единой платы-модуля. Все ее блоки объединены

общей шиной Q-bus, которая входит во внутренний канал микроЭВМ, к которому через интерфейсы ИРПС и ИРПР подключены соответственно пультовой терминал (через КСМ) и ПЧУ (непосредственно).

Кроме внутреннего канала, внешнего по отношению к ЦП, данная микроЭВМ имеет внешний канал по отношению к микроЭВМ, что является более перспективной тенденцией по сравнению с микроЭВМ "Электроника 60М". Наличие такого канала предпочтительно с точки зрения расширения возможностей микроЭВМ и создания на их основе микропроцессорных систем.

Быстродействующие ПУ (ГРВТ и НГМД) подключены к внешнему каналу микроЭВМ через соответствующие контроллеры прямого доступа в память КГД и КНГМД, устройство пользователя – через УСО (показано на рис. 1.1 в виде штрихового прямоугольника).

Канал ЭВМ содержит 38 линий связи, из которых 31 являются двунаправленными (что характерно для интерфейса типа Q-bus). Двунаправленность означает, что по одним и тем же линиям информация может как приниматься, так и передаваться относительно одного и того же устройства.

1.6. Принципы организации обмена информацией с внешними устройствами

Для понимания принципов организации обмена информацией с внешними устройствами следует уяснить следующие вопросы: распределение адресов канала, связь типа "управляющий – управляемый", замкнутую асинхронную связь, режимы обмена данными, принципы организации циклов обращения к каналу и сигналов канала.

1.6.1. Распределение адресов канала

Адреса канала определяются емкостью ОЗУ. Так как количество линий в шине ША-ШД равно 16, с их помощью можно адресовать 64 кб или 32 К 16-разрядных слов (К = 2^{10} = 1024). Полное слово содержит 2 б. Адреса байтов могут быть как четными, так и нечетными. Ячейки ОЗУ, содержащие полные слова, всегда имеют четные адреса.

Ячейки ОЗУ с 000000_8 по 000376_8 зарезервированы под векторы прерывания, и использовать их для других целей не рекомендуется.

Для каждого вектора необходимы две 16-разрядные ячейки (двойное слово), поэтому адреса векторов прерываний являются четными и заканчиваются на 0 или на 4.

Последние 4 К 16-разрядных адресов $(160000\text{-}177776)_8$ обычно отводятся для регистров внешних устройств. Таким образом, максимальный объем реальной памяти равен 28 К 16-разрядных слов.

Часть адресов из последних 4 К зарезервирована под периферийные устройства микроЭВМ.

Таким образом, при обращении к ОЗУ адреса изменяются в сторону возрастания, а при обращении к внешним устройствам убывают (сверху вниз).

1.6.2. Связь типа "управляющий – управляемый"

Связь между двумя устройствами, подключенными к каналу, осуществляется ***по принципу "управляющий – управляемый" (активный – пассивный***).

В любой момент времени только одно устройство является активным. Оно управляет циклами обращения к каналу, удовлетворяет (при необходимости) требования прерывания от внешних устройств, контролирует предоставление прямого доступа к памяти. Пассивное устройство (управляемое) является исполнительным и может принимать или передавать информацию только под управлением активного устройства. Типичным примером таких связей является центральный процессор (как активное устройство), выбирающий команду из памяти, которая всегда является пассивным устройством. Другим примером может служить устройство, работающее в режиме прямого доступа к памяти (например, контроллер КНГМД), при этом память является пассивным устройством.

1.6.3. Замкнутая (асинхропная) связь

Связь через канал является ***замкнутой***, то есть на управляющий сигнал, передаваемый активным устройством, должен поступить ответный сигнал от пассивного устройства. Время ответа не лимитировано, но не должно превышать максимально допустимого.

Асинхронное выполнение операций передачи данных устраняет необходимость в тактовых импульсах, в результате чего обмен с ка-

ждым устройством может происходить с максимально возможным для данного устройства быстродействием.

1.6.4. Режим обмена данными через канал

Канал обеспечивает 3 типа обмена данными:

1) программный обмен;

2) обмен в режиме ПДП;

3) обмен в режиме прерывания программ.

Программный обмен данными – это передача данных по инициативе и под управлением программы. Обычно перед началом обмена проверяется содержимое *регистра состояния* устройства, чтобы определить, готово ли оно к обмену данными.

Обмен данными в режиме прямого доступа к памяти (ПДП) – самый быстрый способ передачи данных между памятью и внешним устройством. Он осуществляется по инициативе внешнего устройства, не изменяя состояние центрального процессора, и поэтому может выполняться в промежутках между циклами обращения к каналу, проводимых центральным процессором.

Адресация и управление размерами передаваемого массива данных находятся под управлением устройства, получившего прямой доступ к памяти. Массивы данных ПДП могут передаваться со скоростью, определяемой быстродействием памяти.

Обмен данными в режиме прерывания программы осуществляется по требованию внешних устройств. В этом случае ЦП приостанавливает выполнение текущей программы, чтобы обслужить запрашивающее устройство. После завершения выполнения программы обслуживания ЦП возобновляет выполнение прерванной программы с того места, где она была прервана.

1.6.5. Принципы организации обмена данными с внешними устройствами

Как показано выше, все модули микроЭВМ объединены каналом. Для выполнения любой команды процессору необходимо выполнить хотя бы одну операцию обращения к каналу. Для некоторых команд требуется выполнить несколько операций. Первой операцией для всех команд является *ввод данных из ячейки памяти*, адрес которой определяется счетчиком команд (СК).

Все операции обращения к каналу для ввода и вывода информации называются ***циклами обращения к каналу***. Если для выполнения команды не требуется обращения за операндами к памяти или внешним устройствам, дополнительных циклов канала не требуется.

Первый принцип обмена данными в микроЭВМ: последовательность операций при выполнении обмена данными между центральным процессором и памятью аналогична последовательности операций при выполнении обмена между центральным процессором и любым внешним устройством.

Дальнейшее рассмотрение организации обмена данными с внешними устройствами проводится по отношению к микроЭВМ типа DEC.

В случае, если выполняется команда с обращением к памяти или внешним устройствам, могут выполняться любые из следующих циклов:

1) ввод;
2) ввод – пауза – вывод;
3) вывод.

В промежутках между циклами центральный процессор может предоставлять канал устройствам ПДП. Требование прерывания может быть удовлетворено только перед выборкой команды, то есть в промежутках между выполнением команд.

Чтобы управлять, надо знать два важных момента: адрес устройства (т.е. к какому устройству производится обращение) и информацию (данные), которые следует передать ему или получить от него. Кроме того, надо учитывать состояние самого устройства, – например, может ли оно принять информацию.

Второй принцип обмена данными в микроЭВМ: все модули микроЭВМ характеризуются определенным состоянием в процессе работы, которое нельзя нарушать. Поэтому для всех модулей, кроме основной памяти, введено понятие "слово состояния", которое хранится в *регистре состояния (РС)* процессора или внешних устройств. Регистры состояния – это фактически адреса устройств (в частности, устройств ввода/вывода РС УВВ), которые содержат информацию об операциях, выполняемых УВВ, характеризуют их состояние и участвуют в операциях по предоставлению прерывания. Каждый регистр состояния подвергается анализу при обмене информацией. Данные, которые принимаются или передаются, хранятся в специальном регистре, который называется *регистром данных* (РД).

Третий принцип обмена данными зависит от специфики устройств микроЭВМ и требует введения дополнительных регистров: регистра адреса памяти (РАП), регистра счета слова (РСС), регистра адреса устройства (РАУ) большой информационной емкости и др. (например, обмена данными с графическим ОЗУ).

Регистр адреса памяти используется для хранения памяти при передаче массивов слов или байтов. Содержимое этого регистра наращивается после передачи каждого слова.

Регистр счета слов предназначен для управления длиной передаваемого массива данных. Информация в РСС заносится из центрального процессора.

Регистр адреса устройств служит для хранения номера дорожки или блока в ЗУ большой емкости.

Четвертым принципом обмена данными является обеспечение информационной совместимости между внешними устройствами и модулем центрального процессора. Эту функцию выполняют интерфейсы.

В состав интерфейсного модуля, как правило, входят следующие узлы:

1) компаратор адреса, вырабатывающий признак обращения к данному УВВ;

2) буферный регистр адреса, фиксирующий часть его разрядов, являющихся кодом выбора одного из регистров адресуемого УВВ;

3) дешифратор управляющих сигналов (ДУС), формирующий требуемый набор сигналов управления;

4) схема синхронизации, обеспечивающая под действием канальных сигналов управления разрешение работы ДУС, БУРА, магистральных вентилей и выработку канального сигнала управления пассивного устройства;

5) регистры УВВ, приведенные выше.

Практически регистры УВВ могут быть построены на триггерах и выполнять функции хранения или же состоять из простых вентилей, открываемых для прохождения через них информации.

Соответствующие регистры могут использоваться как для записи, так и для чтения, или только для записи, или только для чтения. Центральный процессор выявляет готовность УВВ к обмену, либо считывая содержимое его РС, либо принимая его вектор. Затем в

зависимости от направления обмена ЦП загружает или выбирает информацию через РД УВВ.

1.6.6. Форматы регистров внешних устройств

1. Рекомендуемый формат РС ВУ. Регистры данных (РД ВУ), как правило, являются обычными накопительными регистрами; их формат определяется только требованиями пользователя.

Рекомендуемый формат регистров состояния (РС ВУ) показан на рис. 1.2. Данный формат не является обязательным для пользователя, но его структура представляет интерес с двух сторон:

1) форматы регистров состояния стандартных периферийных устройств (печатающего устройства, видеотерминала или дисплея и др.) соответствуют рекомендуемому формату, хотя большинство РС ВУ имеют меньше 16 разрядов;

2) он облегчает разработку и организацию различных устройств в системе с единым каналом передачи информации и повышает эффективность использования системы команд.

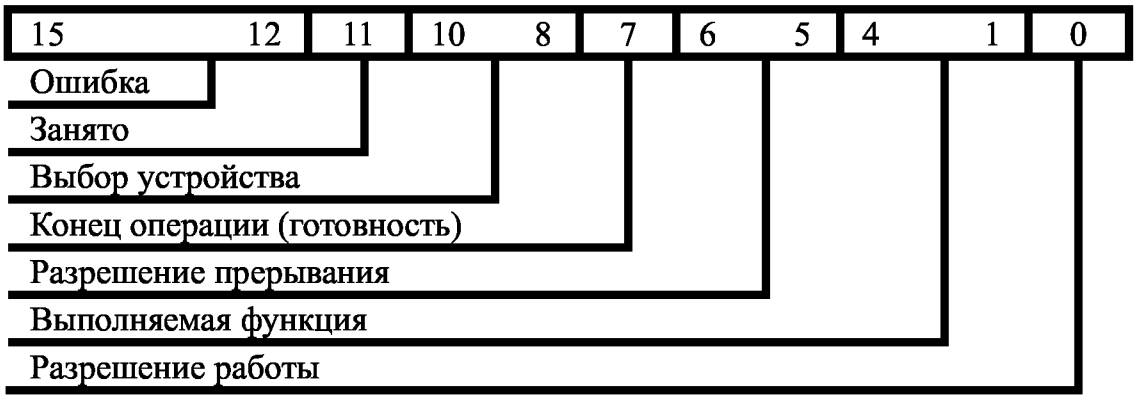


Рис. 1.2. Форматы регистров состояния ВУ

Разряды ***Ошибка (15-12)*** используются для сигнализации о наличии ошибки (разряд 15) и ее характере (разряды 14-12), устанавливаются внешним устройством и должны очищаться центральным процессором для того, чтобы разрешить выполнение операций. Большинство устройств используют разряд 15 для вызова прерывания программы по ошибке (если установлен разряд разрешения прерывания).

Разряд ***Занято (11)*** устанавливается для сигнализации о том, что ВУ занято выполнением внутренних операций и не может принять участие в обмене данными (для некоторых устройств не является необходимым; его функции выполняет разряд "Конец операции").

Разряды ***Выбор устройства (10-08)*** используются для выбора различных устройств, подключенных к общему устройству управления (например, НГМД, подключенных к общему контроллеру).

Разряд ***Конец операции (07)*** устанавливается внешним устройством, когда оно готово для обмена данными. Обычно используется для требования прерывания.

Разряды ***Разрешение прерывания (06-05)*** устанавливаются центральным процессором для разрешения прерывания и передачи адреса вектора прерывания внешним устройством. Разряд 05 используется в случае, если РС имеет два разряда "Конец операции".

Разряды ***Выполняемая функция (04-01)*** определяют операцию, выполняемую ВУ (считывание, запись, перфорацию, поиск зоны и т.д.).

Разряд ***Разрешение работы (00)*** используется для запуска внешнего устройства.

2. Адреса регистров периферийных устройств. Цикл "Вывод" аналогичен операции записи, цикл "Ввод" – считыванию (чтению). Кроме того, цикл "Ввод-пауза-вывод" включает ввод данных, выполнение арифметико-логических операций и вывод результата операции без повторений передачи адреса (результат записывается по адресу последнего выбранного операнда).

При выполнении цикла "Ввод" данные передаются от пассивного устройства к активному (направление передачи при выполнении операций обмена данными определено по отношению к активному устройству).

В табл. 1.1 приведены адреса регистров состояния (РС) и регистров данных (РД) некоторых периферийных устройств.

Таблица 1.1

Адреса регистров состояния и регистров данных
некоторых периферийных устройств

№ пп	Периферийное устройство	Регистр	Адрес в 8 с/с
1	Фотосчитыватель	РС РД	177550 177552
2	Перфоратор	РС РД	177554 177556
3	Клавиатура терминала (дисплея)	РС РД	177560 177562
4	Видеотерминал	РС РД	177564 177566
5	Передающее устройство	РС РД	177514 177516
6	НГМД	РС РД	177130 177132
7	Графическое ОЗУ	РУ РД РА РСС	176640 176642 176644 176646

2. ПРЕДСТАВЛЕНИЕ ЧИСЕЛ С ФИКСИРОВАННОЙ И ПЛАВАЮЩЕЙ ЗАПЯТОЙ В МИКРОЭВМ СИСТЕМЫ DEC

2.1. Числа с фиксированной занятой

Для управления технологическими процессами и объектами в реальном времени, как правило, используются 16-разрядные микроЭВМ системы DEC ("Электроника 60" и "60М", "Электроника ДВК-3", "ДВК-3М2", "ДВК-4" и др.) и системы INTEL (ЕС-1841, 1842, 1843, IBM PC AT и др.).

Общепринятый формат двоичного слова для данного типа любой ЭВМ, в частности, микроЭВМ, принято называть ***разрядной сеткой***.

В микроЭВМ применяется две формы представления чисел: ***с фиксированной запятой*** и ***с плавающей запятой***. Далее материал излагается применительно к микроЭВМ системы Dec.

В двоичной системе любое десятичное число может быть представлено соответствующей последовательностью двоичных цифр:

$$X_{10} = a_{k-1}\ a_{k-2}\ \ldots\ a_1\ a_0,\ a_{-1}\ a_{-2}\ \ldots\ ,$$

где $a_i = 0$ или 1;

k – количество цифр в целой части числа.

Эта запись соответствует коэффициентам многочлена, в виде которого представляется любое число десятичной системы при разложении в ряд с основанием 2.

При представлении чисел с фиксированной запятой положение запятой закрепляется в определенном месте относительно разрядов числа и сохраняется неизменным для всех чисел, которые изображаются в данной разрядной сетке.

Обычно запятая фиксируется перед старшим разрядом (a_1) или после младшего (a_0). В первом случае в разрядной сетке микроЭВМ могут быть представлены числа, которые по модулю меньше единицы ($|x| < 1$), во втором – только целые числа ($|x| > 1$).

Для кодирования знака числа используется *знаковый* разряд: плюс кодируется нулем, минус – единицей. На рис. 2.1 а, б показаны два возможных варианта представления 16-разрядных чисел (включая знак) с фиксированной запятой.

а)

Знак	2^{-1}	2^{-2}	2^{-3}		2^{-14}	2^{-15}
Зн	a_{-1}	a_{-2}	a_{-3}	...	a_{-14}	a_{-15}

б)

Знак	2^{14}	2^{13}	2^{12}		2^{1}	2^{0}
Зн	a_{14}	a_{13}	a_{12}	...	a_1	a_0

Рис. 2.1. Разрядные сетки с фиксированной запятой

Использование представления чисел с фиксированной запятой в микроЭВМ упрощает ее схемы, повышает быстродействие, но создает некоторые трудности при программировании, связанные с необходимостью масштабирования всех исходных данных и получающихся в процессе вычислений промежуточных и конечных величин, которые не должны выходить за пределы разрядной сетки. Такое представление чисел характерно для специализированных микроЭВМ, использующихся для управления технологическими процессами и обработки информации в реальном масштабе времени.

2.2. Числа с плавающей запятой

В универсальных микроЭВМ в качестве основной используется форма представления чисел с плавающей запятой, практически не требующая масштабирования. Наряду с этой формой используется также представление чисел с фиксированной запятой, т.к. операции с целыми числами сокращают время вычислений. В большинстве микроЭВМ используется формат чисел с фиксированной запятой (см. рис. 2.1 б), который (кроме операции с числами) используется в адресной арифметике.

В общем виде любое число X с плавающей запятой имеет вид

$$X = \pm qS^{\pm p},$$

где q – мантисса числа X;

S – основание характеристики, которое обычно совпадает с основанием системы счисления;

p – порядок.

Для двоичной системы счисления мантисса q – правильная двоичная дробь, поэтому

$$X = \pm q2^{\pm p},$$

где q и p – двоичные числа, причем $|q| < 1$.

В 16-разрядных микроЭВМ для размещения чисел с плавающей запятой (в форме одного слова, т.е. с обычной точностью) используются 2 разрядные сетки, как показано на рис. 2.2.

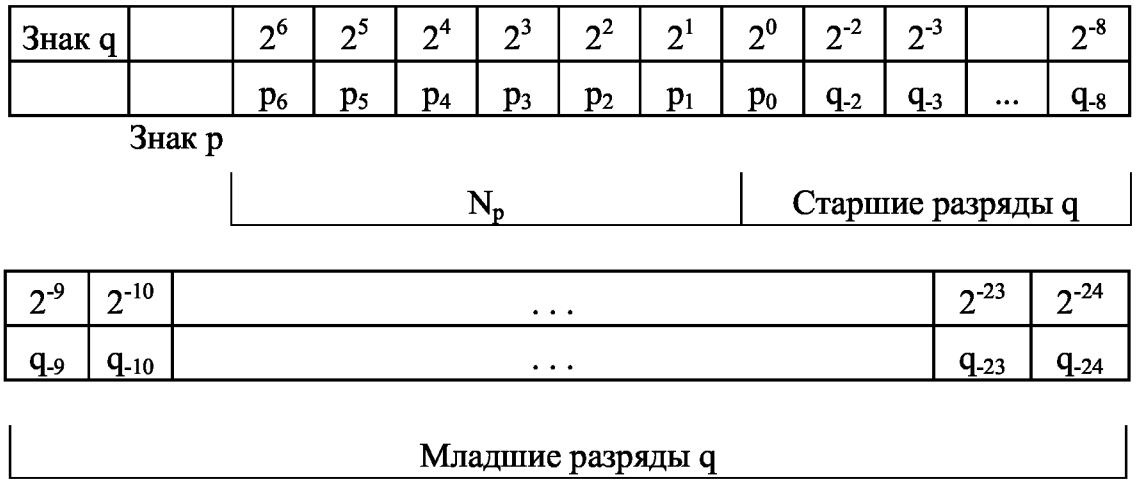


Рис. 2.2. Разрядная сетка микроЭВМ типа DEC
с плавающей запятой в формате слова

В 1-й разрядной сетке старший разряд отводится под знак мантиссы; затем следует разряд знака порядка; следующие 7 разрядов предназначены для размещения значения порядка p; оставшиеся 7 разрядов – для размещения старших разрядов мантиссы q.

Во 2-й разрядной сетке размещаются младшие разряды мантиссы. Веса разрядов, в которых размещаются порядок и мантисса, показаны на рис. 2.2. Из рисунка видно, что под мантиссу отведено 23 разряда, причем сразу за нулевым разрядом порядка p_0 следует разряд мантиссы q_{-2} с весом 2^{-2}, а старший разряд мантиссы q_{-1} с весом 2^{-1} отсутствует в реальной разрядной сетке. Принято считать, что этот разряд мантиссы является ***скрытым*** и всегда равен единице.

Это обстоятельство, пожалуй, самое сложное для понимания, так как, с одной стороны, разряд отсутствует как на рис. 2.2, так и при выводе числа из ОЗУ на экран видеотерминала (как в 2 с/с, так и в 8 с/с), но, с другой стороны, его необходимо учитывать при определении истинного значения мантиссы при вычислениях. Это обусловлено чисто техническими (аппаратными) трудностями. Известно, что для представления каждого разряда числа в 8 с/с требуется 3 двоичных разряда, а число 23 под мантиссу не кратно трем. Если уменьшить количество разрядов под порядок, сужается диапазон представления чисел с плавающей запятой, поэтому недостачу старшего разряда мантиссы решено компенсировать косвенным путем.

Нормализованным двоичным числом называется такое двоичное число, для которого выполняется следующее условие:

$$1/2 \le |q| < 1.$$

Это значит, что старший разряд мантиссы нормализованного двоичного числа всегда равен единице.

Пример. Двоичное число $0.00101...0 \cdot 2^{0...011}$ после нормализации имеет вид

$$0.101...0 \cdot 2^{0...01}.$$

При нормализации двоичного числа производится сдвиг мантиссы влево, а порядок уменьшается на количество сдвигов. Для рассмотренного выше примера мантисса в разрядной сетке имеет другой вид: 0.01...0, а старший разряд с весом 2^{-1} подразумевается как скрытый (скрытый разряд в примере подчеркнут).

Нормализация чисел выполняется в микроЭВМ автоматически, результат вычислений также подвергается нормализации.

Таким образом, очевидно, что при использовании чисел с плавающей запятой арифметическое устройство помимо аппаратуры, выполняющей операции над мантиссами, должно иметь аппаратуру для операций над порядками и для вспомогательных операций (выравнивания порядков, нормализации и др.).

Для упрощения операций над порядками их сводят к действиям над целыми положительными числами, используя представление чисел с плавающей запятой со "смещенным порядком".

2.3. Смещенный порядок

Смещенный порядок p* получается путем прибавления к порядку p целого положительного числа K, называемого ***смещением***:

$$p^* = p + K, \qquad (2.1)$$

где $K = |p_{max}| + 1 = 2^m$;

m – количество разрядов, отведенных под порядок ($n_p = 7$ согласно рис. 2.2);

$|p_{max}| = 1111111$.

Отсюда следует, что смещение K равно коэффициенту пересчета счетчика из $n_p = 7$, т.е.

$$K = 2^7 = 10000000_2 = 200_8,$$

где индексы 2 и 8 обозначают соответствующие системы счисления.

Смещенный порядок p* всегда положителен. Для его представления необходимо количество разрядов, равное сумме разрядов порядка (n_p) и знака порядка, т.е.

$$n_p^* = n_p + 1 = 8.$$

При $n_p = 7$ значение порядка p с учетом знака изменяется в диапазоне от +127 до -128. Смещенный порядок в этом диапазоне показан в табл. 2.1.

Таблица 2.1

Представление чисел в смещенном порядке
в диапазоне от +127 до -128

10 с/с	2 с/с	8 с/с
+127	11 111 111	377
...	.....	...
+1	10 000 001	201
0	10 000 000	200
-1	01 111 111	177
...	.....	...
-128	00 000 000	000

Таким образом, смещенный порядок изменяется от 0_8 до 377_8.

Для получения такого порядка необходимо помнить, что в микроЭВМ системы DEC операции сложения и вычитания заменяются алгебраическим сложением с использованием дополнительного кода.

Правило алгебраического сложения двух двоичных чисел с использованием дополнительного кода следующее: положительные слагаемые представляются в прямом коде, а отрицательные – в дополнительном. Затем производится арифметическое суммирование этих кодов, включая разряды знаков, которые при этом рассматриваются как разряды целых единиц. При возникновении переноса из

знакового разряда единица циклического переноса игнорируется. В результате получается алгебраическая сумма в прямом коде, если эта сумма положительна, и в дополнительном коде, если эта сумма отрицательна.

Так как $K > |p_{max}|$, то при получении p* сумма всегда будет положительна, если $p > 0$, причем для получения p* достаточно в знаковый разряд порядка записать единицу.

Рассмотрим пример, когда $p < 0$.

Пример. Пусть p = -127.

Решение. Прямой код числа

-127 : (ПК - 127) = 1 · 1111111.

Обратный код числа

-127 : (ОК - 127) = 1 · 0000000.

Дополнительный код числа

-127 : (ДК - 127) = 1 · 0000001.

$$p^* = \begin{array}{l} 10000001 \quad (\text{ДК } -127) \\ +\ 10000000 \qquad\quad (\text{К}) \\ \hline 100000001 \quad (\text{результат}) \\ \Big| \text{единица циклического переноса} \end{array}$$

Итак, $p^* = 00000001_2 = 001_8$.

2.4. Нахождение порядка числа

Формула (2.1) не дает однозначного перехода от смещенного порядка p* к действительному p .

Предлагается простой способ нахождения порядка p, при этом учитывается состояние старшего разряда смещенного порядка p*. Из табл. 2.1 видно, что для всех значений от 0 до +127 в старшем разряде двоичного кода смещенного порядка присутствует единица, а для всех значений отрицательного p – нуль. Поэтому для перехода к действительному порядку необходимо выполнить следующее:

1) восстановить знак порядка p, проинвертировав старший разряд смещенного порядка p*;

2) считать полученный код результатом вычисления действительного порядка и, используя сформулированное выше правило алгебраического сложения двух двоичных чисел с дополнительным кодом, записать код порядка р.

Пример.

Найти порядок р:

1) $p_1^* = 10001010$;

2) $p_2^* = 00000001$.

Решение. Восстанавливаем знаки порядков:

1) $P_1 = \mathbf{0}0001010$ (ПК);

2) $P_2 = \mathbf{1}0000001$ (ДК).

Знаковые разряды в P_1 и P_2 выделены. Согласно принятому правилу кодирования знаков порядка, число $P_1 > 0$, а $P_2 < 0$. Так как P_1 и P_2 являются результатами вычислений, то, согласно правилу алгебраического сложения с использованием дополнительного кода, P_1 получено в прямом коде (ПК), а P_2 – в дополнительном коде (ДК).

Значит,

$$p_1 = \text{ПК}\ (P_1); \quad p_2 = \text{ДК}\ (P_2).$$

Отсюда следует, что $p_1 = P_1 = +\ 10_{10}$. Для получения p_2 необходимо найти дополнительный код от P_2. Учитывая, что дополнительный код от дополнительного кода числа дает прямой код, получим p_2:

$$\text{ОК}\ (P2) = 11111110;$$
$$\text{ДК}\ (P_2) = 11111111_2 = -127_{10}.$$

В табл. 2.2 показано представление некоторых характерных чисел с плавающей запятой (в формате слова): *теоретическое* (со всеми разрядами) и *в памяти микроЭВМ* (с использованием скрытого разряда). В теоретическом представлении числа записаны в одну строку, в которой предполагается 32 разряда, при этом смещенный порядок выделен, а старший разряд мантиссы, который становится скрытым в ОЗУ микроЭВМ, подчеркнут.

Следует обратить внимание на следующее:

1. Положительный знак числа ноль при выводе на экран видеотерминала и печатающее устройство в 2 с/с не отображается.

2. Код числа в 2 с/с и 8 с/с выводится на экран и печатающее устройство без учета скрытого разряда.

3. Код двоичного числа, хранимого в памяти микроЭВМ, выводится в 10 с/с с учетом скрытого разряда.

4. Для перевода в 8 с/с двоичного кода, хранимого в ОЗУ микроЭВМ, используется известное правило: двоичное число разбивается на триады влево и вправо от запятой, которая зафиксирована перед старшим разрядом мантиссы (см. п. 2.1), а триады представляются восьмеричными эквивалентами.

5. Таблица справедлива как слева направо, так и справа налево.

6. Для простоты восприятия порядок нормализованного числа указан в десятичном коде.

Таблица 2.2

Представление некоторых характерных чисел
с плавающей запятой в формате слова

Исходное число			Представление чисел в формате двух слов		
		нормализованное	2 с/с		8 с/с
10с/с	8с/с	2 с/с	теоретическое	в ОЗУ микроЭВМ	
4	4	$0.100*2^{3}$	**01000001**1<u>0</u>...0	0100000110000000 0000000000000000	406000
2	2	$0.100*2^{2}$	**01000001**0<u>1</u>0...0	0100000100000000 0000000000000000	404000
1	1	$0.100*2^{1}$	**01000000**1<u>1</u>0...0	0100000010000000 0000000000000000	402000
0.5	0.4	$0.100*2^{0}$	**01000000**0<u>1</u>0...0	0100000000000000 0000000000000000	400000
0.25	0.2	$0.100*2^{-1}$	**00111111**1<u>0</u>...0	0011111110000000 0000000000000000	376000
0.125	0.1	$0.100*2^{-2}$	**00111111**0<u>1</u>0...0	0011111100000000 000000000000000	374000
0.0625	0.01	$0.100*2^{-3}$	**00111110**1<u>1</u>0...0	0011111010000000 000000000000000	372000
-4	-4	$1.100*2^{3}$	1**1000001**1<u>0</u>...0	1100000110000000 0000000000000000	1406000
-0.0625	-0.01	$1.100*2^{-3}$	1**0111110**1<u>0</u>...0	1011111010000000 000000000000000	1372000

Задание:

1. Составить на языке БЕЙСИК две программы:

1) программу последовательного ввода с пульта десятичных чисел с фиксированной точкой с записью их в ОЗУ микроЭВМ в виде двоичных чисел с плавающей запятой в формате 2 слов (предусмотреть вывод на экран видеотерминала значений чисел в 8 с/с);

2) программу последовательного ввода с пульта кода двоичных чисел с плавающей запятой в формате 2 слов (предусмотреть вывод на экран видеотерминала значений чисел в 10 с/с).

При составлении программы воспользоваться оператором POKE и функциями VARPTR, PEEK.

2. Используя программу (п.1 а) и данные табл. 2.3 (п.2), составить таблицу, аналогичную табл. 2.2 (слева направо).

3. Используя программу (п.1 б) и данные табл. 2.3 (п. 3), составить вторую таблицу, аналогичную табл. 2.2 (справа налево).

4. Найти порядок числа в 2 с/с и 10 с/с, используя смещенный порядок, приведенный в табл. 2.3 (п. 4) (п. 2, 3, 4 соответствуют нумерации столбцов).

Таблица 2.3

Представление некоторых чисел с плавающей запятой в смещенном порядке

Вариант	Число в 10 с/с	Двоичное число с плавающей запятой	Смещенный порядок
1	2	3	4
1	1275.75	1010101011010101	01100110
	430.25	1010101010101010	10011011
	-1275.75	0101010101011010	
	-430.25	0101010101010101	01110010
2	3040.625	0000111101110011	10011001
	860.5	1101100010001111	01110100
	-3040.625	1111000100011111	
	-860.5	0001111000010000	00001110
3	6020.375	1111000101111001	01010111
	1720.685	0001111000110010	10111101
	-6020.375	0111110000001110	
	-1720.685	1100111100110011	00001111

Окончание табл. 2.3

1	2	3	4
4	0.012 3875.25 -0.012 -3875.25	0001110010011100 1111000011110101 1111101101000111 0000110001110100	10011100 00011110 00001011
5	0.00275 4375.65 -0.00275 -4375.65	1000001111110000 0001110000110011 0000001111100011 1100000011011111	01111100 10011110 00101101
6	795.625 180.25 -795.625 -180.25	0110011000011111 0001111000111111 1000111000111000 1111000111000111	01110001 11111000 00011100
7	2335.7 0.0035 -2335.7 -0.0035	1001100111100000 1110011000000000 0111000111000111 0000111000111000	10001110 00000111 01100111
8	0.0233 4830.5 -0.0233 -4830.5	0111100011110000 1111000011110011 1000011000001111 0000111111100000	00011100 00101011 11111100

Содержание отчета

1. Титульный лист.
2. Задание.
3. Листинг программ на языке БЕЙСИК, содержащий фамилию, номер группы и вариант задания.
4. Две таблицы по форме табл. 2.2; третья таблица может быть произвольной формы.

3. ПРЕДСТАВЛЕНИЕ ЧИСЕЛ В РАЗРЯДНОЙ СЕТКЕ МИКРОЭВМ СИСТЕМЫ INTEL

3.1. Микропроцессоры фирмы INTEL

Американская фирма Intel, разработавшая в 1971 году первый в мире 4-разрядный микропроцессор I4004, сохраняет свое лидерство в области создания новых микропроцессоров.

Широко известно семейство микропроцессоров I80X86, относящихся к CISC-архитектуре (CISC – Complex Instruction Set Computer – полный (обычный) набор команд для компьютера), которая на уровне машинного языка реализует комплексные наборы команд различной сложности (от простых, характерных для микропроцессоров первого поколения 8086, до сложных, реализованных в современных 32-разрядных микропроцессорах Pentium-3, Pentium-4).

Значительным успехом этой фирмы явилось создание RISC-процессора в одном кристалле, в то время как другие фирмы создавали процессоры в нескольких кристаллах (RISC – Reduced Instruction Set Computer – сокращенный набор команд для компьютера). RISC-архитектура обеспечивает высокоэффективную реализацию минимальных наборов простых команд, определенных на основе статистического анализа большого количества программ для CISC-процессоров исходной архитектуры. О создании RISC-процессора I80860 (I860) фирма Intel заявила в 1989 году. По мнению разработчиков, он является микропроцессорной версией суперкомпьютера Cray, размещенного в одном кристалле.

Постепенное усложнение CISC-процессоров происходит в направлении более совершенного управления машинными ресурсами, совместимости, сближения машинных языков с языками высокого уровня. Однако сложная система команд и переменный формат команды процессоров с CISC-архитектурой привели к быстрому росту сложности схем: например, версия I80386 содержит 270 тыс. транзисторов, I80486 – 1 млн., Pentium-3 – до 28 млн., а Pentium-4 – 55 млн.

Более простой набор команд RISC-микропроцессоров способствует повышению тактовых частот и обработке большего числа команд на такт по сравнению с CISC-процессорами. Исторически RISC-процессоры отличались большей производительностью при операциях с плавающей точкой, чем их CISC-аналоги. Однако ус-

ложнение RISC-процессоров для расширения их области применения фактически приближает их архитектуру к CISC-архитектуре и далее – переход в HLL-архитектуру (HLL – High-Level Language – язык высокого уровня), которая имеет машинный язык, соответствующий высокому уровню.

В настоящее время число микропроцессоров с RISC-архитектурой существенно возросло, однако фирма Intel, занимающая доминирующее положение на рынке процессоров, делает основной упор на сочетание элементов RISC-технологии в рамках проектируемых CISC-микропроцессоров. Такая позиция позволяет сохранить высокую совместимость выпускаемых процессоров, повышая при этом их производительность в таких областях, как Multimedia и САПР (системы автоматизированного проектирования). Например, реализация набора команд для ускорения обработки графики и видео, получившего название Multi Media Extension (MMX), является ярким примером RISC-вставок в архитектуру CISC. Расширения MMX впервые были встроены в процессоры Pentium 166 MMX (тактовая частота ядра – 166 МГц, частота шины – 66 МГц) и поддерживаются всеми последующими поколениями чипов от Intel. Современные процессоры класса Pentium-3 и Pentium-4 наряду с MMX содержат в себе дополнительные наборы команд SSE и SSE2, призванных повысить производительность в операциях с плавающей запятой.

Одновременно с усложнением архитектуры компания Intel продолжает существенно увеличивать быстродействие своих чипов за счет повышения тактовых частот. Внутренний коэффициент k умножения частоты системной шины (шина адреса, команд, данных) у процессоров является фиксированным, т.е. назначается в ходе технологического процесса изготовления микросхемы. Частоты системных шин современных Pentium-4 достигают 533 МГц при тактовой частоте ядра более 3 ГГц (1 ГГц = 1000 МГц); таким образом, коэффициент k принимает значения 5 ~ 6, характеризуя тем самым, насколько процессор опережает по скорости подсистемы управления оперативной памятью. Разные тактовые частоты обусловлены удаленностью друг от друга составных частей компьютера (процессор, память, интерфейсы), что, в свою очередь, не позволяет синхронизировать частоты из-за существенного влияния помех.

3.2. МикроЭВМ системы команд INTEL

Из зарубежных микроЭВМ системы INTEL в Республике Беларусь широко распространены микроЭВМ фирмы IBM PC XT и AT различных модификаций.

В этом обозначении: XT – шина XT-bus, применяемая в IBM PC совместимых ПЭВМ на базе микропроцессоров I8086/8088, использующих 8-разрядную двунаправленную внешнюю шину данных; AT – шина AT-bus, которая расширяет XT-bus в 16-разрядную двунаправленную шину данных для IBM совместимых компьютеров на базе микропроцессоров 80286, 80386, 80486.

Интерфейсы XT-bus и AT-bus положены в основу международного промышленного стандарта ISA (ISA-8 и ISA-16), который модифицирован рядом фирм путем функционального расширения в EISA (Extended-ISA). Имеется также интерфейс ISA-32 с 32-разрядной шиной данных для семейства ПЭВМ на базе I80386, 80486.

В обозначении компьютеров после типа шины (XT или AT) через наклонные черточки ставятся тип микропроцессора и сопроцессора (при этом первые две цифры опускаются): AT/286/287 или AT/286.

Последним в линейке AT является чип 486, на смену которому пришло поколение Pentium. Постоянное развитие ядра микропроцессоров данного направления выразилось в появлении нескольких поколений Pentium, – например, Pentium-2, Pentium-3, Pentium-4. Кроме того, начиная с Pentium-2, компания Intel параллельно выпускает более дешевые версии своих чипов, получившие название Celeron. Отличие Celeron-версии процессора от полноценного Pentium-2 или Pentium-3 в том, что он содержит меньший обьем быстродействующей промежуточной памяти на кристалле (КЭШ-память) и, как следствие, показывает меньшую производительность.

Далее через наклонные черточки могут указываться тактовая частота микропроцессора в мегагерцах (для XT и AT типичные величины – 12, 16, 20, 25, 33, 40, 50 и 66; для Pentium – от 75 до более 3000 МГц), размер КЭШ-памяти в килобайтах (для XT и AT типичные величины – 32, 64, 128; для Pentium – от 64 кб до нескольких мегабайт). В качестве дополнительных характеристик в маркировке микропроцессора может быть указана частота системной шины (FSB), – например, 66 МГц, 100 МГц, 133 МГц, 400 МГц, 533 МГц и более. Повышение частоты шины существенно сказывается на бы-

стродействии системы, т.к. в этом случае вырастает пропускная способность подсистемы процессор-память.

Размер оперативной памяти (RAM) указывают в мегабайтах, – например, для AT компьютера типичные величины – 1, 2, 4, 8, 16 и до 128 Мб, для Pentium они могут достигать нескольких гигабайт. Кроме объема могут быть указаны тип и частота памяти, – например, SDRAM – синхронная динамическая память (в процессе хранения битов информации микросхема постоянно обновляется), которая может работать на частотах 66 МГц, 100 МГц, 133 МГц. Дальнейшее развитие SDRAM выразилось в появлении на рынке памяти DDR – динамической памяти с удвоенной по отношению к SDRAM частотой, – например, DDR 266 МГц или 333 МГц. Другой тип динамической памяти – RAMBUS или RDRAM – может работать на частоте 400 МГц по обоим фронтам тактового импульса (так же, как DDR), т.е. на результирующей частоте 800 МГц. В отличие от SDRAM и DDR, которые имеют 64-разрядную шину, RDRAM имеет 16 разрядов (в перспективе – 64), что позволило существенно поднять частоту.

Размер современных накопителей (Hard Disk Drive, HDD, Винчестер) может достигать десятков гигабайт, – например, 2,1, 8,4, 10, 20, 60, 80, 120 Гб. Каждая из указанных выше архитектур (XT, AT, Pentium разных поколений) имеет собственные ограничения на максимальный размер подключаемого накопителя из-за особенностей программного обеспечения базовых систем ввода-вывода (BIOS).

Наряду с объемами оперативной и дисковой памяти может указываться тип видеоадаптера и объем локальной видеопамяти, расположенной на нем. Кроме функции преобразования сигнала для монитора на современный видеоадаптер могут быть переложены ряд операций по ускорению обработки плоских (2D) и объемных (3D) моделей, а также обработки видео. Таким образом, маркировка видеоадаптера начинается с типа интерфейса подключения ISA (см. выше), PCI (частота – 33 МГц, универсальный 32-разрядный интерфейс) или AGP (специализированный 32-разрядный графический интерфейс с частотой 66 МГц, выпущенный в нескольких версиях: AGP, AGPx2, AGPx4). Далее указывается маркировка чипа, – например ATI Radeon, GForce, S3 Savage, – по кодовому названию которого можно установить его основные параметры производительности. Последними указываются объем и тип видеопамяти, необходимой для хранения текущего изображения, а также графических примитивов.

Пример.

IBM PC AT/386/387/33/64/2/80

Pentium 3 900/128/60/GForce2-32

В обозначениях могут использоваться:

DX – вариант микропроцессоров I80386 и I80486 с 32-битовой шиной данных (в отличие от более дешевых и менее быстродействующих микропроцессоров типа SX, в которых используется 16-битовая шина);

SX – вариант микропроцессоров I80386 и I80486 с 16-битовой шиной данных, которые дешевле, но приблизительно в 1,5 раза медленнее, чем аналогичные микропроцессоры типа DX (кроме того, I80486SX в отличие от I80486 не содержит сопроцессора);

DX2(DX4) – вариант микропроцессора I80486 с удвоенной (x4) внутренней тактовой частотой; работает вдвое быстрее, а быстродействие компьютера увеличивается в 1,5 (2,5) раза;

EGA – адаптер дисплея, обеспечивающий разрешающую способность 640x350 точек с 16 цветами (этот тип уже устарел, с ним работает EGA-монитор);

VGA – адаптер дисплея, обеспечивающий разрешающую способность 640x480 точек с 16 цветами или 320x200 с 256 цветами, – самый распространенный в настоящее время (с ним работает VGA-монитор);

SVGA (super VGA) – адаптер дисплея, обеспечивающий возможности VGA и работу в графическом режиме с разрешением 800x600, 1024x768 и более точек; выбор конкретного разрешения и количества цветов для SVGA-адаптера зависит от объема локальной видеопамяти и возможностей подключаемого к нему монитора (видеоадаптер SVGA работает с VGA или SVGA-монитором);

TFT – жидкокристаллическая матрица, используемая в качестве устройства вывода; отличается повышенной четкостью изображения и может поддерживать набор режимов SVGA-дисплеев, построенных на электроннолучевой трубке.

3.3. Особенности микроЭВМ системы INTEL

МикроЭВМ системы INTEL имеют следующие особенности:

1. Кодирование команд и адресов осуществляется в шестнадцатеричной системе счисления.

2. Обычно используется трехшинная организация магистрали интерфейса (иногда называемая системной шиной): шина данных (ШД), адресная шина (ША), шина управления (ШУ), за исключением микроЭВМ на базе I8086, в которой, как и в микроЭВМ системы DEC, используется двухшинная организация магистрали. Разрядность адресной шины колеблется от 20 до 24 в зависимости от типа ПЭВМ, что дает возможность обращаться к ОЗУ емкостью от 1 до 16 Мб.

3. Алгебраическое сложение в микроЭВМ осуществляется с использованием обратного кода.

4. Использование виртуальной памяти начинается с компьютеров на базе I80286.

5. Используется мультипрограммный режим работы (с I80286).

6. Осуществляется конвейерная обработка информации (с I80286).

7. Использование КЭШ-памяти начинается с 32-разрядных компьютеров (I80386).

8. Используется побайтовый ввод/вывод информации.

Восьмиразрядные микроЭВМ системы INTEL на базе микропроцессоров I8080 и отечественных КР580 здесь не рассматриваются.

3.4. Размещение чисел в разрядной сетке

Рассмотренная выше теория о размещении чисел в разрядной сетке микроЭВМ системы DEC полностью справедлива и для микроЭВМ системы INTEL.

При этом следует обратить внимание на следующее:

1. Диапазон представления целых чисел для микроЭВМ на базе микропроцессоров I8086, I8088, I80186 и I80286 – такой же, как для микроЭВМ системы DEC. В микроЭВМ на базе I80386-I80586 диапазон расширяется, т.к. микроЭВМ являются 32-разрядными.

2. Для целей управления технологическими процессами ПЭВМ на базе I8086 и I8088 не приспособлены.

3. Для представления вещественных чисел с одинарной точностью используется четырехбайтовый стандарт американского Института инженеров по электротехнике и радиоэлектронике (формат IEEE), соответствующий формату микроЭВМ системы DEC и поддерживаемый сопроцессорами.

4. Все вещественные числа в микроЭВМ приводятся к виду $\pm q2^{\pm p}$ (где q – мантисса; p – порядок). В 16-разрядных регистрах микроЭВМ они хранятся в виде, показанном на рис. 3.1. Старшая единица мантиссы нормализованного числа хранится в скрытом разряде.

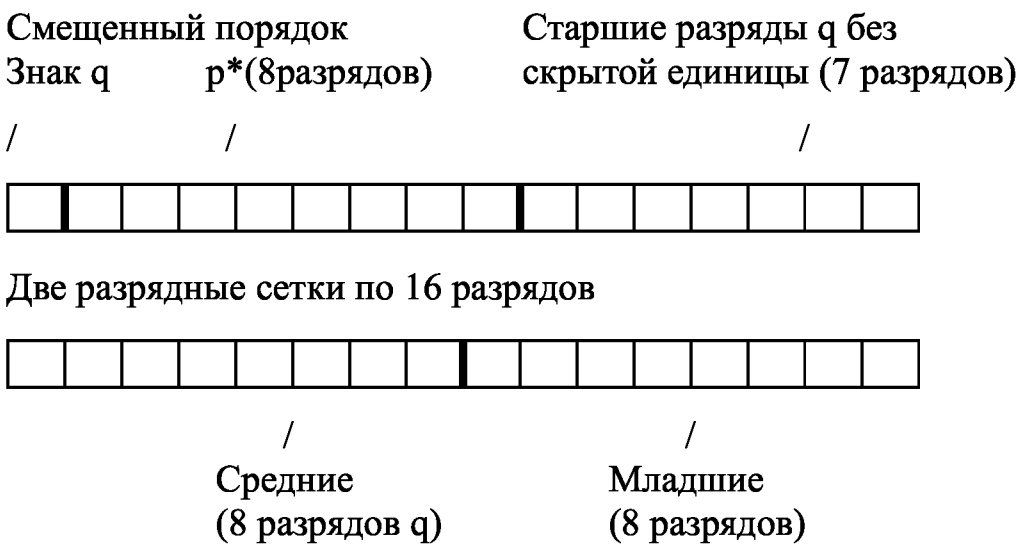


Рис. 3.1. Размещение вещественного числа в разрядной сетке

5. В обеих системах микроЭВМ ОЗУ имеют байтовую структуру памяти. В микроЭВМ системы DEC достаточно указать 2 четных адреса при записи-считывании вещественного числа, в микроЭВМ системы INTEL – 4 адреса.

Пример. Пусть X – вещественное число. Адрес при работе с БЕЙСИКом определим с помощью функции VARPTR. Тогда:

для микроЭВМ DEC:

A_1 = VARPTR (X) – два старшие байта числа;

A_2 = A1 + 2 – два младшие байта числа;

для микроЭВМ INTEL:

A_1 = VARPTR (X) – младший байт q;

A_2 = A1 + 1 – средний байт q;

A_3 = A2 + 1 – старший байт q;

A_4 = A3 + 1 – старший байт числа, где хранится p*.

6. В микроЭВМ DEC число вводится в ОЗУ и выводится на экран в виде двух частей по 16 разрядов: 2 двоичных или 2 восьмеричных числа, причем знак (0 или 1) стоит в старшем разряде старших 16 разрядов числа (ноль не отображается). В микроЭВМ INTEL

ввод-вывод осуществляется по байтам. В старшем байте мантиссы (см. рис. 3.1) – всего 7 разрядов, поэтому знак q переносится в этот байт. Тогда структура 4 байтов в памяти выглядит, как показано на рис. 3.2.

Используя оператор PRINT со знаком ";" в конце списка, можно вывести в строку по 2 байта.

Рис. 3.2. Структура 4 байтов при вводе/выводе

7. Старший байт числа (смещенный порядок p*) соответствует табл. 3.1, в которой указаны характерные значения диапазона изменения порядка p*, и табл. 2.1 из трех столбцов (без 16 с/с)).

Таблица 3.1

Представление чисел в смещенном порядке в диапазоне p*

в 10 с/с (p)	в 2 с/с (p*)	в 8 с/с (p*)	в 16 с/с (p*)
+127	11111111	377	FF
....	.....	...	...
+1	10000001	201	81
0	10000000	200	80
-1	01111111	177	7F
....	.....	...	...
-128	00000000	0	0

8. При восстановлении порядка p на основании p* можно использовать правило восстановления порядка, изложенное в разделе 2, несмотря на то, что операции в INTEL-системах осуществляются в обратных кодах.

В табл. 3.2 показано представление некоторых характерных чисел с плавающей запятой: теоретическое (со всеми разрядами) и в памяти микроЭВМ (с использованием скрытого разряда).

В теоретическом представлении числа записаны в одну строку, в которой размещается 32 разряда, при этом смещенный порядок выделен, а старший разряд мантиссы, который становится скрытым, подчеркнут. Четыре байта в ОЗУ показаны по 2 в строке, начиная со старшего, причем байты разделены пробелами.

9. Положительный знак числа ноль при выводе на экран видеотерминала и печатающее устройство в 2 с/с не отображается.

10. В языке GW-BASIC отсутствует функция BIN$ и двоичная константа, начинающаяся с символов &B, поэтому рекомендуем использовать HEX$, OCT$, &0..., &H... .

11. Коды чисел, хранимые в памяти микроЭВМ, выводятся на экран и печатающее устройство в 2 с/с, 8 с/с и 16 с/с без учета скрытого разряда.

12. Код двоичного числа из памяти микроЭВМ выводится в 10 с/с с учетом скрытого разряда.

13. При переводе двоичного кода числа в 8 с/с и 16 с/с следует без учета скрытого разряда каждый байт перевести по известным правилам в соответствующий код.

14. При вводе нулевого значения байта достаточно указывать один ноль; при выводе также отображается один ноль.

15. Табл. 3.2 справедлива как слева направо, так и справа налево.

Задание

1. Составить на языке БЕЙСИК две программы:

1) программу последовательного ввода с пульта десятичных чисел с фиксированной точкой с записью их в ОЗУ микроЭВМ в виде шестнадцатеричных чисел с плавающей запятой в формате 4 байтов (предусмотреть вывод на экран видеотерминала значений чисел в 8 с/с и в 16 с/с);

2) программу последовательного ввода с пульта кода шестнадцатеричных чисел с плавающей запятой в формате 4 байтов (предусмотреть вывод на экран видеотерминала значений чисел в 10 с/с).

При составлении программ воспользоваться оператором POKE и функциями VARPTR и PEEK.

2. Используя программу п.1 а и данные табл. 3.3 (п. 2), составить таблицу, аналогичную табл. 3.2 (слева направо).

3. Используя программу п.1 б и данные табл. 3.3 (п. 3), составить вторую таблицу, аналогичную табл. 3.2 (справа налево), предварительно разбив каждые два 16-разрядных слова на 4 байта и выразив каждый байт в 8 с/с или в 16 с/с.

4. Найти порядок числа в 2 с/с и 10 с/с, используя смещенный порядок, приведенный в табл. 3.3 (п. 4).

(Номера пунктов берутся по столбцам.)

Таблица 3.2

Представление некоторых характерных чисел
с плавающей запятой в формате 4 слов

Исходные данные			Нормализованное число	Представление чисел в формате 4 слов			
10 с/с	8 с/с	16 с/с	2 с/с	2 с/с		на экране	
				Теоретическое	в ОЗУ микроЭВМ	8 с / с	16 с / с
1	2	3	4	5	6	7	8
4	4	4	$0.100*2^{3}$	**0**1000001**1**100...0	10000011__00...0 00.....0__00...0	203__0 0 __0	83__0 0 __0
2	2	2	$0.100*2^{2}$	**0**1000001**0**100...0	10000010__00...0 00.....0__00...0	202__0 0 __0	82__0 0 __0
1	1	1	$0.100*2^{1}$	**0**1000000**1**100...0	10000001__00...0 00.....0__00...0	201__0 0 __0	81__0 0 __0
0.5	0.4	0.8	$0.100*2^{0}$	**0**1000000**0**100...0	10000000__00...0 00.....0__00...0	200__0 0 __0	80__0 0 __0
0.25	0.2	0.4	$0.100*2^{-1}$	**0**0111111**1**100...0	01111111__00...0 00.....0__00...0	177__0 0 __0	7F__0 0 __0
0.125	0.1	0.2	$0.100*2^{-2}$	**0**0111111**0**100...0	01111110__00...0 00.....0__00...0	176__0 0 __0	7E__0 0 __0
0.0625	0.01	0.1	$0.100*2^{-3}$	**0**0111110**1**100...0	01111101__00...0 00.....0__00...0	175__0 0 __0	7D__0 0 __0
-4	-4	-4	$0.100*2^{3}$	**1**1000001**1**100...0	10000011__10...0 00.....0__00...0	203__200 0 __0	83__80 0 __0
-0.0625	-0.01	-0.1	$0.100*2^{-3}$	**1**0111110**1**100...0	01111101__10...0 00.....0__00...0	175__200 0 __0	7D__80 0 __0

Таблица 3.3

Представление некоторых характерных чисел с плавающей запятой в смещенном порядке в 32-разрядном формате

Вариант	Числа в 10 с/с	Двоичные числа с плавающей запятой, размещенные в 32-разрядном формате	Смещенный порядок
1	1275.75	10101010110101 01	01100110
	430.25	10101010101010 10	10011011
	-1275.75	01010101011010 10	
	-430.25	01010101010101 01	01110010
2	3040.625	00001111011100 11	10011001
	860.5	11011000100011 11	01110100
	-3040.625	11110001000111 11	
	860.5	00011110000100 00	00001110
3	6020.375	11110001011110 01	01010111
	1720.685	00011110001100 10	10111101
	-6020.375	01111100000011 10	
	-1720.685	11001111001100 11	00101100
4	0.012	00011100100111 00	10011100
	3875.25	11110000111101 01	00011110
	-0.012	11111011010001 11	
	-3875.25	00001100011101 00	00101111
5	0.00275	10000011111100 00	01111100
	4375.65	00011100001100 11	10011110
	-0.00275	00000011111000 11	
	-4375.65	11000000110111 11	00011110
6	795.625	01100110000111 11	01110001
	1807.25	00011110001111 11	11111000
	-795.625	10001110001110 00	
	-1807.25	11110001110001 11	01101101
7	2335.7	10011001111000 00	10001110
	0.0035	11100110000000 00	00000111
	-2335.7	01110001110001 11	
	-0.0035	00001110001110 00	01100111
8	0.0233	01111000111100 00	00011100
	4830.5	11110000111100 11	00101011
	-0.0233	10000110000011 11	
	-4830.5	00001111111000 00	11111100

Содержание отчета

1. Титульный лист.
2. Задание.
3. Листинг программы на языке БЕЙСИК, содержащий фамилию, номер группы и вариант задания.
4. Две таблицы по форме 3.2; третья таблица может быть произвольной формы.

4. УПРАВЛЕНИЕ ТЕХНОЛОГИЧЕСКИМ ПРОЦЕССОМ (ОБЪЕКТОМ) С ПОМОЩЬЮ МИКРОЭВМ СИСТЕМЫ DEC

4.1. Задачи, стоящие перед разработчиком микропроцессорной системы управления

Микропроцессорная система управления (МСУ), как правило, содержит 2 основные части: аппаратную и программную.

Перед разработчиком МСУ стоят следующие довольно сложные задачи:

1. Изучить обьект (процесс) управления, его особенности и степень подготовленности к автоматизации.

2. Разработать алгоритм контроля и управления технологическим процессом (объектом) с учетом выбранного или заданного критерия оптимального управления. В общем случае могут подлежать разработке как простые законы дискретного управления (типа "да – нет"), так и более сложные алгоритмы непосредственного цифрового управления, включая сложные законы оптимального управления, отличные от классических, вплоть до адаптивных.

3. Исходя из разработанного алгоритма управления, разделить МСУ на аппаратную и программную части с учетом дополнительных критериев (надежности, стоимости, быстродействия и др.).

4. Выбрать микропроцессорные средства вычислительной техники для проектирования аппаратной части системы (серийный микроконтроллер, микроЭВМ, персональный компьютер, специализированный контроллер и т.д.). Преимущество серийного контроллера (если он подходит по техническим характеристикам) очевидно.

5. При выборе микроЭВМ или персонального компьютера разработать устройство сопряжения с объектом (УСО).

6. Разработать программное обеспечение (ПО) системы, включающее, кроме операционной системы (ОС) или отдельных файлов ОС, основную управляющую программу (ОУП) пользователя.

При разработке ОУП неизбежно встает вопрос об организации обмена информацией между микропроцессорным блоком управления и обьектом через УСО. На этой стадии необходимо выбрать режим обмена – программный, прямого доступа в память, по прерыванию или комбинированный. Этот выбор во многом определяет эффективность обмена информацией и накладывает отпечаток на степень сложности, а значит, и стоимости ОУП.

Как следует из вышеизложенного материала, разработчик должен решить вопрос о регистрах состояний (РС) и регистрах данных (РД) и при необходимости ввести дополнительные специальные регистры. Адреса регистров выбираются из области оперативного запоминающего устройства (ОЗУ), определенной адресами $(160000\text{-}177776)_8$ в восьмеричном коде, исключая адреса, которые зарезервированы для периферийных устройств.

Количество регистров РС и РД для МСУ зависит от многих факторов: сложности алгоритма контроля и управления, количества контролируемых и управляемых цепей, степени их взаимосвязанности, последовательности включения-выключения и т.п.

Регистры данных внешних устройств (РД ВУ) являются, как правило, обычными накопительными регистрами; их формат определяется только требованиями разработчика (пользователя) (рекомендуемый формат регистров состояния (РС ВУ) изложен в разделе 1).

4.2. Управление моделью объекта

Алгоритм контроля и управления может быть опробован при натурных испытаниях, то есть непосредственно на обьекте, либо на испытательном стенде с имитационной моделью обьекта, либо на микроЭВМ системы DEC с помощью моделирующей программы. В последнем случае моделирующая программа должна поддерживать обьект. Степень поддержки зависит от задачи, которая стоит перед разработчиком (проверка правильности программы, реализующей алгоритм; проверка правильности программы и контроля поведения

обьекта и т.д.). Задачи, которые ставит разработчик перед моделирующей программой, определяют ее сложность и стоимость, что дополнительно увеличивает затраты на разработку и испытания МСУ.

При моделировании на микроЭВМ пользователь может использовать свою программу так, как будто существует сам объект.

При этом возникает определенная проблема, состоящая в том, что при отсутствии УСО нарушается принцип асинхронной замкнутой связи, и поэтому адреса регистров (РС и РД), заложенные в ОУП, не существуют.

При моделировании эту проблему можно решить двумя путями, заменив реальные адреса регистров моделирующими адресами:

1) выделить область адресов ОЗУ, которая не используется при программировании на языке моделирования;

2) зарезервировать адреса ячеек ОЗУ с помощью переменных, – например: RS_1 – RS_i (для РС) и RD_1 – RD_j (для РД); в этом случае для определения адреса следует использовать функцию VARPTR или оператор DEF SEG.

В данной работе будет использован 1-й вариант, так как при программировании на БЕЙСИКе микроЭВМ системы DEC не использует ячейки с адресами $(11000 - 11.....)_8$ в восьмеричном коде.

4.3. Описание объекта управления

Реальный обьект управления представляет собой автоматизированное транспортное средство робототехнического комплекса, управляемое встроенным микроконтроллером. Основная управляющая программа для контроллера после разработки, отладки и апробации заносится в перепрограммируемое постоянное запоминающее устройство (ППЗУ).

Программа обеспечивает возможность движения с различной скоростью (с помощью набора переключаемых скоростей V_1 - V_n) к различным пунктам назначения, – например, к стеллажам (с помощью набора переключаемых путей S_1 - S_n), – и цепь включения напряжения питания (U).

Наборы V_n, S_n и напряжение U представляют собой узлы управления, внешние по отношению к контроллеру, то есть три внешних устройства ВУ, каждое из которых имеет по 2 регистра – РС и РД. Каждый из трех РС характеризуется своим разрешенным для управления

кодом состояния. Выбор скорости и пути определяется кодом регистра данных. Регистр данных напряжения питания имеет один код.

В данном обьекте управления установлена определенная последовательность включения цепей, нарушение которой не приведет к движению обьекта:

1) скорость – из набора V_n;

2) путь – из набора S_n;

3) напряжение питания.

4.4. Модель объекта и моделирующая программа

Модель обьекта управления в данной работе является упрощенной, так как предназначена не для отработки алгоритма управления реальным объектом, а для обучения принципам управления с помощью микропроцессорного блока или микроЭВМ любым объектом, имеющим ряд узлов (цепей) контроля и управления.

Поэтому моделирующая программа поддерживает набор только из 2 скоростей (V_1 и V_2), 2 путей (S_1 и S_2) и напряжения питания (U). В табл. 4.1 приведены реальные и моделирующие адреса РС и РД трех узлов управления, разрешенные коды состояния каждого РС и коды каждого РД с возможностью выбора параметра из набора. Все коды в таблице записываются в восьмеричном коде.

Таблица 4.1

Реальные и моделирующие адреса и коды трех узлов управления

№ пп	Наименование узлов	Регистр	Реальный адрес	Моделирующий адрес в 8 с/с	Разрешен код РС в 8 с/с	Наименование параметра	Код РД в 8 с/с
1	Узел скорости	РС РД	171000 171002	11000 11002	200	 V1 V2	 340 350
2	Узел пути (пункта назначения)	РС РД	171004 171006	11010 11012	400	 S1 S2	 360 370
3	Узел напряжения питания	РС РД	171010 171012	11020 11022	600	 U	 400

Моделирующая программа состоит из двух подпрограмм: первая начинается с номера строки 1000, вторая – с номера 2000. На диске они хранятся в файле под именем "M1".

Первая подпрограмма отображает на экране упрощенно сам объект и три узла управления им. В правой части экрана приведена таблица, отражающая состояние цепей; в нижней его части (в области объекта управления) отражаются коды состояния всех трех регистров PC в момент запуска основной управляющей программы пользователя.

Вторая подпрограмма поддерживает динамику переключения узлов управления. При этом происходят соответствующие изменения в цепях управления каждого узла по мере выполнения программы, дополнительно фиксирующиеся в правой таблице; в нижней части фиксируются новые коды PC.

При правильно составленной основной управляющей программе с обращением к подпрограммам 1000 и 2000 обьект движется со скоростью V_1 или V_2 к пункту 1 или к пункту 2 в зависимости от S_1 и S_2.

Рисунок модели обьекта и узлов управления здесь не приводится, так как будет выведен на экран видеотерминала.

4.5. Рекомендации к разработке алгоритма управления объектом и пользования моделирующей подпрограммой

При разработке алгоритма управления обьектом и пользовании моделирующей подпрограммой следует обратить внимание на следующее:

1. Основная управляющая программа пользователя начинается с операторов комментариев, содержащих информацию о названии программы, фамилии исполнителя и номере группы.

2. После операторов комментариев ставится оператор обращения к первой подпрограмме 1000.

3. С помощью команды MERGE к написанным операторам подсоединяется подпрограмма из файла с именем "M1". После подсоединения моделирующей подпрограммы основная программа готова к частичной работе, так как организовано взаимодействие с моделирующей подпрограммой 1000, осуществляемое в диалоговом режиме. После каждого обращения к подпрограмме на экране высвечивается

мерцающая подсказка: "Нажмите ВК", позволяющая пользователю проанализировать происходящие изменения в узлах управления.

4. Запускается программа. На экране появятся модель объекта, узлы управления и информация об объекте. Для того, чтобы уяснить исходное состояние узлов управления объекта, следует еще раз прочитать подраздел 4.4, а затем записать коды регистров состояния. Клавишу <ВК> нажимать не следует, так как основная программа не разработана.

5. Возвращаемся к написанию основной программы. Для удобства следует очистить экран, используя команды Color2 и CLS.

6. При дальнейшем написании основной программы следует руководствоваться алгоритмом управления узлами управления обьекта, который аналогичен для каждого из 3 узлов обьекта управления:

1). Сначала анализируется состояние соответствующего регистра:

$$< PC_i >= PK, \qquad (4.1)$$

где < > – содержимое;

i – произвольный индекс, указывающий на наличие нескольких РС;

РК – разрешенный для управления код.

Если условие (4.1) не выполняется, программа должна ждать, непрерывно опрашивая состояние <PC_i>. Если это условие выполняется, в содержимое регистра данных необходимо занести соответствующий код:

$$< РД_i > = KD, \qquad (4.2)$$

где РД – регистр данных;

KD – код данных.

При анализе следует обратить внимание на следующее:

1) при анализе (4.1) используются функции OCT$ и PEEK; при записи кода данных в РД в соответствии с (4.2) используется оператор POKE;

2) все коды в 8 с/с приведены в табл. 4.1.

2). Следующим в основной программе должен стоять оператор обращения к подпрограмме 2000.

7. Алгоритм по п. 6 должен быть реализован в основной программе 3 раза – применительно к V, S и U.

8. Основная программа заканчивается оператором Stop, отделяющим основную программу от моделирующих подпрограмм.

После разработки программы ее необходимо отладить.

Задание

1. Разработать основную управляющую программу для управления моделью автоматизированного транспортного средства с использованием моделирующей подпрограммы "M1", используя три параметра обьекта: V_1, S_1 и U.

2. Записать коды регистров состояния на каждом этапе выполнения программы.

3. С помощью редактирования основной программы выполнить п. 1 – 2 при других параметрах:

1) V_2, S_1 и U; 2) V_1, S_2 и U; 3)V_1, S_2 и U.

Содержание отчета

1. Листинги основной программы для всех случаев.

2. Объяснение причины задержек выполнения программы на разных стадиях (путем анализа разрядов PC_i и рекомендуемого формата РС ВУ).

3. Вывод о наибольшей скорости и наибольшем пути.

4. Рисунок модели объекта и узлов управления с описанием изменения узлов управления на каждом этапе выполнения программы.

5. УПРАВЛЕНИЕ ТЕХНОЛОГИЧЕСКИМ ПРОЦЕССОМ (ОБЪЕКТОМ) С ПОМОЩЬЮ МИКРОЭВМ СИСТЕМЫ INTEL

5.1. Структура MS DOS

Управление технологическим процессом (обьектом) с помощью микроЭВМ системы INTEL имеет свои особенности и требует знания дополнительного материала: структуры MS DOS, метода адресации ячеек памяти, распределения оперативной памяти, принципа управления периферийными устройствами и устройствами пользователя (два последних типа устройств с точки зрения управления представляют собой внешние устройства) через порты.

Операционная система (ОС) MS DOS является одним из главных компонентов общего программного обеспечения. Она обеспечивает выполнение двух главных задач:

1) поддержки работы всех программ и обеспечения их взаимодействия с аппаратурой;

2) представления пользователю возможностей общего управления машиной.

В рамках 1-й задачи обеспечиваются взаимодействие программ, работа с внешними устройствами (ВУ), динамическое распределение оперативной памяти (ОП) и т. д.

Одна из важнейших функций ОС – поддержка работы широкого набора периферийных и дополнительных устройств.

В состав MS DOS входят два важнейших компонента: встроенный в машину компонент и программные модули, записанные на магнитном диске. Встроенным в машину компонентом является BIOS (Basic Input – Output System – базовая система ввода-вывода). К программным модулям относятся: блок начальной загрузки, модуль расширения базовой системы ввода-вывода, модуль обработки прерываний, командный процессор и утилиты ОС.

В области данных BIOS находятся сведения о портах ввода-вывода; к этой области необходимо обращаться при управлении внешними устройствами.

5.2. Адресное пространство. Линейная и сегментная адресации

Когда разрабатывались 8-разрядные процессоры, считалось, что для их практического применения достаточно 64 кб адресного пространства. Поэтому в широко используемых компьютерах до конца 1980-х годов длина адресного слова составляет 16 б. При такой длине слова с адресами и данными можно обращаться одинаково, что значительно упрощает структуру ЦП.

Успехи в технологии БИС привели к появлению кристаллов ЗУ с информационной емкостью 1 Мбит. В результате существенно увеличился объем памяти микроЭВМ, а программное обеспечение стало сложным и разнообразным. Увеличение обьема памяти микроЭВМ вынудило проектировать 16-разрядные процессоры с длинным

адресным словом, определяющим адресное пространство микроЭВМ: 20 бит – в I8086, 23 бит – в Z8001 и 24 бит – в МС 68000.

20-разрядное адресное слово обеспечивает возможность обращения к адресному пространству в 1 Мб, а 24-разрядное адресное слово увеличивает эту возможность в 16 раз, т.е. до 16 Мб.

В 16-разрядных процессорах 2-го поколения I80286 используется 24-разрядное адресное слово (2^{24} = 16 Мб).

По экономическим соображениям микроЭВМ IBM PC AT на базе I80286 обеспечиваются ОЗУ в 1 Мб. Из-за увеличения адресного слова при 16-разрядной сетке (для информационного слова) возникают вопросы: в какой форме поместить такой адрес в 16-разрядный регистр и каким образом осуществить вычисление адреса?

Для решения этих задач используются два вида адресации: линейная и сегментная.

При ***линейной адресации*** адрес представляет собой отдельное целочисленное значение. В 16-разрядных процессорах адреса хранятся в 32-разрядных регистрах (слово двойной длины), а вычисление адреса производится с помощью 32-разрядного сумматора. Архитектура микропроцессора довольно сложная, но зато удобная, так как все адресное пространство используется как единое целое.

При ***сегментной адресации*** все пространство адресов делится на множество сегментов, как показано на рис. 5.1 при 24-разрядном адресном слове. Адресное пространство, разбитое на такие сегменты, называется *сегментированным пространством адресов*. Начальный адрес сегмента называется *базовым*. За каждым сегментом закреплен соответствующий номер. Порядок разбиения пространства может быть произвольным, а после того как он установлен, адрес можно представить в виде двух целочисленных величин – номера сегмента и смещения, т.е. можно использовать два 16-разрядных регистра, которые хранят номер сегмента и значения смещений.

Следовательно, максимальный размер одного сегмента составляет 64 кб. Таким образом, вычисление адреса в принципе сводится только к вычислению смещения, и для этого можно использовать те же 16-разрядные АЛУ, не прибегая к помощи 32-разрядных регистров и сумматоров, что значительно упрощает структуру процессора. В процессорах I8086, I80286 фирмы Intel используется сегментная адресация, но не в виде номеров сегментов и смещения, а с использованием их базового адреса.

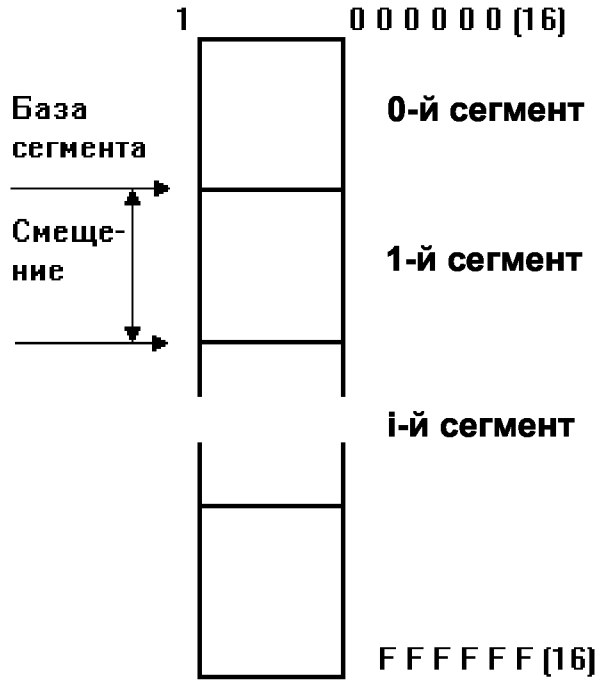


Рис. 5.1. Сегментированное адресное пространство

Оба эти числа являются 16-разрядными и записываются в следующем виде:

$$B : D, \qquad (5.1)$$

где B – базовый адрес сегмента;

D – смещение.

База сегмента в 20-разрядном коде адреса представлена старшими 16 разрядами. Оставшиеся 4 младших разряда равны нулю. Таким образом, 20-битный базовый код имеет вид

$$XXXX0_{16}. \qquad (5.2)$$

Ограничения на базу сегмента практически отсутствуют (базовым адресом может быть любое число, кратное 16), т.е. сегменты могут начинаться на границах блоков по 16 б.

Компонент B в зависимости от адресуемого типа данных хранится в одном из 4 регистров: CS – сегмент команд, DS – сегмент данных, SS – сегмент стека и ES – экстракодовый сегмент, называемый также дополнительным. Это значит, что в процессоре I80286 одновременно могут использоваться 4 сегмента одного пространства.

Сдвиг 16-разрядного базового адреса D на 4 разряда влево позволяет получить 20-разрядный начальный адрес сегмента согласно (5.2). Компонент D задает смещение адресного байта относительно начала сегмента. Поэтому полный физический адрес определяется по формуле

$$A = 16 \cdot B + D. \qquad (5.3)$$

Следует учитывать, что смещение не всегда указывают относительно базового адреса сегмента; его могут указывать и относительно сегмента с адресом 0. Зная формулу (5.3), легко установить, что адреса 0040:0002 и 0:0402 указывают на один и тот же байт памяти. Действительно, если адрес 0040:0002 подвергнуть преобразованию по (5.3), то получится 0:0402; при этом умножение на 16 следует заменить сдвигом влево на один шестнадцатеричный разряд (тетраду).

5.3. Распределение оперативной памяти

В табл. 5.1 приведена карта распределения оперативной памяти (ОП).

Таблица 5.1

Карта распределения оперативной памяти

Границы ОП, кб	Сегментный адрес: смещение	Содержание ОП
1	2	3
0	0000:0000	Векторы прерываний: 256 четырехбайтных адресов
1	0040:0000	Область данных BIOS
1.25	0050:0000	Область данных ОС
	xxxx:0000	Модуль ОС IO.SYS
	xxxx:0000	Модуль ОС MS DOS.SYS
	xxxx:0000	Буферная часть в области данных ОС
	xxxx:0000	Резидентная часть командного процессора COMMAND.COM (приблизительно 4 кб)
	xxxx:0000	Резидентные утилиты
	xxxx:0000	Область памяти для прикладных программ типа COM и EXE

Окончание табл. 5.1

1	2	3
		Стек для программ
		Нерезидентная часть командного процессора COMMAND.COM
640	A000:0000	Экранная память для адаптера EGA
704	B000:0000	Экранная память для монохромного дисплея
736	B800:0000	Экранная память для адаптера с CGA
768	C000:0000	Начало области ПЗУ
984	F600:0000	Интерпретатор языка БЕЙСИК в ПЗУ
1016	FE00:0000	Модуль BIOS в ПЗУ

Первые 1024 б ОП отведены для векторов прерываний. Каждый из них задает адрес сегмента и смещение для программы обработки прерывания. Один вектор занимает 4 б: 2 б со старшими адресами определяют сегмент, а 2 б с младшими адресами – смещение.

По адресу F000:FFF5 записывается адрес версии BIOS. Здесь используются 8 б в ASCII-коде. По адресу F000:FFFE записывается код типа ПЭВМ. Интерпретатор БЕЙСИКа занимает 32 кб, модуль BIOS – последние 8 К.

Области памяти для IBM AT помечаются символами AT.

5.4. Порты ввода-вывода

Порты ввода-вывода (ПВВ) – это блоки (модули), задачей которых является осуществление взаимодействия между микропроцессорной системой (МПС) и внешней средой (внешними коммуникациями). Они представляют собой унифицированное средство подключения внешних устройств к МПС – элементы интерфейса.

Порт ввода (***Пвв***) – это любой источник данных, – например (но не обязательно), адресуемый регистр, подключенный к шинам МПС, – выдающий слово в МП, когда к нему осуществляется обращение.

Порт вывода (***Пвыв***) – это любой приемник данных, – например, адресуемый регистр, подключенный к шинам МПС, получающий слово от МП, когда тот обращается к нему.

Итак, Пвв – это адресуемые одно- или двунаправленные буферные регистры, предназначенные для построения программируемого

интерфейса, имеющие свои адреса, поэтому к МПС может подключаться несколько внешних устройств ВУ.

Каждый порт является составной частью интерфейса между МП и ВУ, – например, датчиками, ЦАП, АЦП, терминалом, внешней памятью и т.д.

В большинстве МП для адресации портов (т.е. выборки нужного порта) используется адресная шина или ее часть. Очень часто адреса Пвв отличаются от адресов Пвыв и от адресов памяти не значениями, а сигналами на соответствующих управляющих линиях.

5.5. Основные принципы ввода-вывода

Существует 3 основных способа инициирования передачи данных и управления этим процессом (аналогично микроЭВМ системы DEC):

1) ввод-вывод, управляемый программой;

2) ввод-вывод, управляемый подпрограммой обработки прерываний;

3) ввод-вывод, управляемый аппаратными средствами (прямой доступ в память (ПДП)).

В данной работе внимание сосредоточено на программно-управляемом вводе-выводе.

Порты могут быть соединены с микропроцессором по такому же принципу, как в микроЭВМ системы DEC, чтобы обращение к ним могло осуществляться аналогично ячейкам памяти по заданным адресам. Такое техническое решение получило название ***ввода-вывода, адресуемого по принципу памяти***.

Однако в микроЭВМ системы INTEL большее распространение нашел метод соединения портов для обращения к ним по специальным командам микропроцессора – IN и OUT, – обеспечивающим передачу данных для ввода-вывода с распределением канала между внешними устройствами.

Не следует путать команды процессора IN и OUT с операторами языка БЕЙСИК INP и OUT, которые при интерпретации (построчной трансляции) превращаются в соответствующие команды процессора.

Совместимость между ВУ. и микроЭВМ так же, как и в микроЭВМ системы DEC, обеспечивают интерфейсные схемы, рассредоточенные по модулям микроЭВМ (см. раздел 2).

Логические схемы декодирования генерируют импульс выбора интегральных схем, используя для этого адрес с адресной шины и управляющие сигналы ввода-вывода. Импульс выбора ИС для входного порта стробирует занесение данных в порт из устройства ввода и вводит данные, полученные процессором, в буферный регистр порта. Входной порт содержит 3 стабильных буфера с тремя логическими состояниями – высокое напряжение, низкое напряжение и большое сопротивление, – предназначенных для развязки линий ввода данных от шины (причем эта развязка действует все время, за исключением периодов чтения с шины). В микроЭВМ системы DEC используется 2 уровня: высокое и низкое напряжение.

Структура выходного порта включает буферные регистры, сохраняющие данные в течение того времени, пока они поступают бит за битом на относительно медленные устройства вывода.

5.6. Программно-управляемый ввод-вывод

Имеется 2 типа программно-управляемого ввода-вывода: с условной и с безусловной передачей.

При ***безусловной передаче*** обмен данными с портом ввода-вывода осуществляется без предварительного определения готовности порта принимать или передавать данные.

Как правило, безусловная передача используется для обмена командной информацией, а также данными о состоянии устройств. Иногда она применяется при обмене информацией с внешними устройствами, при котором предполагается, что устройство ввода-вывода готово принимать или передавать данные. Ошибки при передаче данных могут возникать в случаях, если при программировании не принимается необходимых мер предосторожности, – например, когда микропроцессор направляет данные в более быстром темпе, чем тот, в котором выходной порт в состоянии их принимать (тогда данные могут быть потеряны). При обращении микропроцессора к входному порту со слишком большой частотой последний будет предоставлять ему избыточные данные. Очевидно, что при таком способе обмена следует очень четко знать аппаратурные особенности микропроцессора, портов, устройств сопряжения с объектом, особенности технологического процесса и т.п.

Чтобы устранить трудности, возникающие при безусловной передаче информации, применяют ***условную передачу данных*** с использованием процедуры квинтирования установления связи, при которой до начала передачи собственно данных осуществляется безусловная передача информации о состоянии компьютера из устройства ввода/вывода в процессор. Информация о состоянии выдается в виде комбинации битов, называемой *флагом* и указывающей на состояние готовности порта ввода-вывода. Необходимость использования подпрограммы или дополнительных операторов для проверки флага состояния вызывает увеличение времени выполнения операции ввода-вывода. Если в системе с программно-управляемым вводом-выводом имеется несколько внешних устройств (узлов), необходимо провести поочередную проверку флагов всех устройств (узлов), называемую *опросом флагов*.

Итак, анализ флага состояния компьютеров системы INTEL аналогичен анализу регистра состояния компьютера системы DEC или регистра статуса при управлении печатающим устройством компьютера системы INTEL (см. раздел 7).

Флаг состояния обычно соответствует состоянию одного из разрядов выбранного для этих целей порта, называемого флаговым. Если флаг состояния равен единице, это свидетельствует о наличии данных. При приеме компьютером данных с входного порта сигнал выбора для данного порта разрешает считывание данных и одновременно сбрасывает флаг их наличия. Аналогично осуществляется выдача управляющей информации во внешние устройства.

5.7. Карта распределения адресов портов ввода-вывода

Внешние порты напоминают ячейки оперативной памяти. Как и байты оперативной памяти, порты могут иметь адреса от 0 до 65535. Задействуются далеко не все порты. Порт с адресом A не имеет никакого отношения к байту оперативной памяти с тем же адресом, это – два совершенно разных объекта; запись в один из них никак не влияет на содержимое другого. Доступ к портам на языке БЕЙСИК осуществляется с помощью операторов OUT, WAIT и функции INP. Доступ к ячейкам памяти осуществляется с помощью оператора POKE и функции PEEK.

Перечень адресов некоторых портов приведен в табл. 5.2.

Таблица 5.2

Адреса некоторых портов

№ группы	Адреса портов в 16 с/с	Обслуживающие устройства
1	00-1F	Контроллер прямого доступа к памяти DMA, 8237 (для AT – первый контроллер DMA)
	20-2F	Контроллер прерываний 8259 (XT)
	20-3F	Контроллер прерываний 1 (AT)
	40-5F	Программируемый таймер 8253/8254
	60-6F	Адаптер интерфейса с периферией 8255
	70-7F	Автономные часы реального времени (AT)
	80-9F	Регистры страниц DMA
	AO-BF	Контроллер прерываний 2 (AT)
	CO-DF	Второй контроллер DMA (AT)
	1FO-1F8	Контроллер винчестера (AT)
2	200-20F	Адаптер игровых пультов
	278-27E	Адаптер параллельного принтера (дополнительный)
	2F8-2FF	Адаптер канала связи COM2 (асинхронная передача (21))
3	300-31F	Макетная плата
4	320-32F	Контроллер винчестера (XT)
	783-37F	Адаптер параллельного принтера (основной)
	3BO-3BF	Адаптер параллельного принтера и черно-белого дисплея
	3DO-3DF	Адаптер цветного графического дисплея
	3FO-3F7	Контроллер флоппи-дисков (ГМД)
	3F8-3FF	Адаптер канала связи COM1

С каждым внешним устройством обычно связано несколько портов, выполняющих определенные функции на различных стадиях обмена, в одни из которых заносятся управляющие признаки и адреса (номера) внутренних регистров обслуживаемой микросхемы, другие используются для обмена данными между центральным процессором и выбранными внутренними регистрами, по содержимому третьих операционная система получает информацию о состоянии внешнего устройства, анализирует признаки завершения очередной процедуры обмена, контролирует правильность функционирования оборудования (см. разделы 7, 8).

Из табл. 5.2 видно, что адреса портов можно разделить на несколько больших групп: 1 – адреса, закрепленные за системной объ-

единительной платой (512 адресов с &H0000 по &H01FF); 2, 4 – адреса, закрепленные за портами функциональных плат (512 адресов с &H0200 по &H03FF); 3 – адреса, отведенные под макетную плату, то есть используемые для устройства сопряжения с объектом управления (адреса с &H0300 по &H31F).

Следует учитывать следующие обстоятельства:

1) по мере развития компьютеров IBM может использоваться значительно большее количество адресов;

2) адреса портов данных стандартных периферийных устройств могут быть определены при обращении к области данных BIOS (см. раздел 7).

5.8. Макетные платы

Для управления объектом необходимо обеспечить сопряжение компьютера IBM PC с датчиками и исполнительными механизмами.

У фирмы IBM можно приобрести макетную плату, которая дает пользователю возможность разрабатывать свои собственные средства сопряжения с шиной IBM PC, обеспечивает подключение к системной шине, содержит большую зону для выполнения соединений накруткой и небольшой участок для печатного соединительного монтажа, на котором реализован шинный интерфейс, включающий базовые средства декодирования для группы портов (использование буферов гарантирует, что шина не будет чрезмерно нагружена шинными схемами). Плата обеспечивает 32 порта.

Более широкие возможности (до 64 портов) обеспечивает макетная плата для компьютера IBM PC, разработанная компанией Vector 4613-1, в которой имеется 32 входных и 32 выходных порта.

Зная системную шину компьютера IBM PC и используя микросхему 8255 фирмы Intel или отечественный эквивалент – однокристальный программируемый интерфейс периферийных устройств, в котором реализовано 3 программируемых порта ввода-вывода, – пользователь сам может разработать и изготовить макетную плату.

5.9. Управление моделью объекта

К моменту выполнения данной работы студенты должны ознакомиться с материалом по управлению технологическим процессом (объектом) с помощью микроЭВМ системы DEC (раздел 4).

На основании вышеизложенного в сравнении с материалом раздела 4 нетрудно установить следующее:

1. Принципы управления внешними устройствами в обеих системах микроЭВМ схожи (если в системе DEC надо иметь дело с регистром состояния и регистром данных, то в системе INTEL – с флагом состояния и регистром данных, причем вместо флага состояния (когда установлен один разряд в единице) пользователь может использовать любой код как состояние, назвав его регистром состояния, регистром статуса (см. раздел 6) или флагом состояния.

2. Задачи, стоящие перед разработчиком микропроцессорной системы управления, полностью совпадают.

3. Если к портам не подключено устройство сопряжения с объектом или макетная плата, то адреса портов, как и адреса регистров в системе DEC, не существуют. В этом можно убедиться, прочитав содержимое любого порта макетной платы согласно табл. 5.2 с помощью функции INP языка БЕЙСИК. Содержимое неподключенного порта соответствует коду 377_8 или FF_{16}.

4. Обьект управления выбран тот же, что и в разделе 4.

5. Модель объекта управления аналогична.

6. Моделирующая подпрограмма аналогична (M1). Единственная разница состоит в способе резервирования адресов ячеек ОЗУ для регистров данных и флагов состояний. Здесь рекомендуется использовать резервирование адресов ячеек ОЗУ с помощью переменных RDV, RDS, RDU, FV, FS и FU, где RD – регистр данных; F – флаги; V, S, U – скорость, путь и напряжение.

Основные разрешающие коды приведены в табл.5.3.

Таблица 5.3

Основные разрешающие коды

№ п/п	Наименование узлов	Регистр	Моделирующий адрес	Разрешающий код	Параметр	Код 8 с/с	RD 16 с/с
1	Узел скорости	F RD	FV RDV	1	V1 V2	340 350	E0 E8
2	Узел пути (пункт назначения)	F RD	FS RDS	1	S1 S2	360 370	F0 F8
3	Узел напряжения	F RD	FU RDU	1	U	300	C0

5.10. Рекомендации к разработке алгоритма управления

При разработке алгоритма управления необходимо учитывать следующее:

1. Основная управляющая программа пользователя начинается с операторов комментариев, содержащих информацию о названии программы, фамилии исполнителя и номере группы.

2. После операторов комментариев ставятся операторы присваивания: RDV = 0, RDS = 0, RDU = 0, FV = 0, FS = 0, FU = 0 (что равносильно определению моделирующих адресов).

3. Производится подсоединение с помощью команды MERGE к написанным операторам основной программы подпрограммы из файла с именем "M1".

4. Запускается программа. На экране появятся модель объекта, узлы управления и информация об объекте.

5. Осуществляется возвращение к написанию основной программы. Для удобства следует очистить экран.

6. Алгоритм управления аналогичен алгоритму управления в разделе 4.

Задание

1. Разработать основную управляющую программу для управления моделью автоматизированного транспортного средства с использованием моделирующей подпрограммы "M1" и 3 параметров обьекта: V_1, S_1 и U.

2. Записать коды флаговых состояний на каждом этапе выполнения программы.

Выполнить п. 1, 2 при других параметрах:

V_2, S_1 и U;

V_1, S_2 и U;

V_2, S_2 и U.

Содержание отчета

1. Листинги основной программы для всех случаев.

2. Вывод о наибольшей скорости и наибольшем пути.

3. Адреса флагов и регистров данных.

4. Рисунок модели объекта и узлов управления с описанием изменения узлов управления на каждом этапе выполнения программы.

6. КОДИРОВАНИЕ АЛФАВИТНО-ЦИФРОВОЙ ИНФОРМАЦИИ И УПРАВЛЕНИЕ ПЕЧАТАЮЩИМ УСТРОЙСТВОМ С ПОМОЩЬЮ МИКРОЭВМ СИСТЕМЫ DEC

6.1. Основные технические характеристики ПЧУ

Работа печатающего устройства (ПЧУ) типа "Электроника" УВВ ПЧ-30-004 основана на принципе матрично-точечной печати ударного действия. Печатаемый знак (буква, цифра) формируется из точек, расположенных в одной вертикальной колонке. Точки наносятся на носитель информации (бумагу) печатающими иглами, приводимыми в действие электромагнитами.

Основные характеристики ПЧУ:

1. Скорость печати – не менее 30 знаков в секунду.

2. Печать осуществляется в обе стороны, т. е. при движении печатающей головки слева направо и справа налево.

3. Характеристики знака:

1) для стандартного знака – растр 7 х 5 точек;

2) для знака пользователя – растр от 1 х 1 до 8 х 8 точек.

4. Количество знаков в строке определяется типом используемого шрифта (для стандартного шрифта – 80 знаков).

5. Наборы знаков:

1) стандартный (158 знаков – прописные и строчные буквы русского и латинского алфавита, цифры, знаки препинания, знаки арифметических операций, специальные символы);

2) задаваемый пользователем (64 произвольных символа в растре матрицы до 7 х 7 точек).

6. Кодирование информации осуществляется по ГОСТ 13052-74 (КОИ-7) или ГОСТ 19768-74 (КОИ-8).

6.2. Порядок тестирования ПЧУ

Тестирование ПЧУ производится в следующем порядке:

1. Включается тумблер "Сеть" (загорается светодиод "Сеть"), и ПЧУ готово отрабатывать входные коды, поступающие по интерфейсу ИРПР ОСТ 25778-77.

2. Для запуска ПЧУ в режиме автотестирования следует нажать одну из кнопок ("↓" или "↑") на верхней крышке устройства и удерживать ее до включения тумблера "Сеть":

1) при нажатии кнопки "↓" обрабатывается тест проверки печати;

2) при нажатии кнопки "↑" выполняется тест проверки команд.

3. Для отмены теста проверки печати необходимо нажать кнопку "↑", а теста проверки команд – кнопку "↓" (длительность нажатия – не менее 3 с).

6.3. Кодирование входной информации

При кодировании входной информации необходимо обратить внимание на следующее:

1. Прием информации с интерфейса осуществляется двоичными 7-битными кодами в соответствии с ГОСТ 13052-74 или 8-битными кодами в соответствии с ГОСТ 19768-74.

2. В соответствии с ГОСТ 19767-74 установлено 2 класса символов:

1) графические – предназначенные для представления данных;

2) управляющие, которые инициируют или изменяют действие, влияющее на регистрацию, обработку или интерпретацию данных.

3. В качестве стандартных символов в печатающем устройстве используются алфавитно-цифровые символы наборов КОИ-7НО, КОИ-7Н1, КОИ-7Н2 и КОИ-8. Кодирование графических символов наборов 0, 1, 2, 8 определяется данными рис. 6.2; 6.3; 6.4; 6.5 соответственно.

Код символа (К) в таблице определяется числом, включающим порядковый номер столбца (Х) и порядковый номер строки (У), на пересечении которых находится символ, по формуле

$$К = 16 \cdot Х+У.$$

Например, кодом символа "#" является число 35.

4. Помимо стандартных наборов символов пользователь может самостоятельно определить 31 графический символ, которые вместе с символом "пробел" образуют стандартный набор пользователя.

После включения ПЧУ в качестве стандартного набора пользователя устанавливается набор графических символов ШП-8, который в данной работе не используется.

После включения питания устанавливается состояние печатающего устройства, соответствующее набору КОИ-7Н2.

6.4. Управление печатающим устройством

6.4.1. Формат регистра состояния ПЧУ

В регистре состояния ПЧУ используются не все разряды. Формат РС ПЧУ показан на рис. 6.1, где заштрихованы используемые разряды.

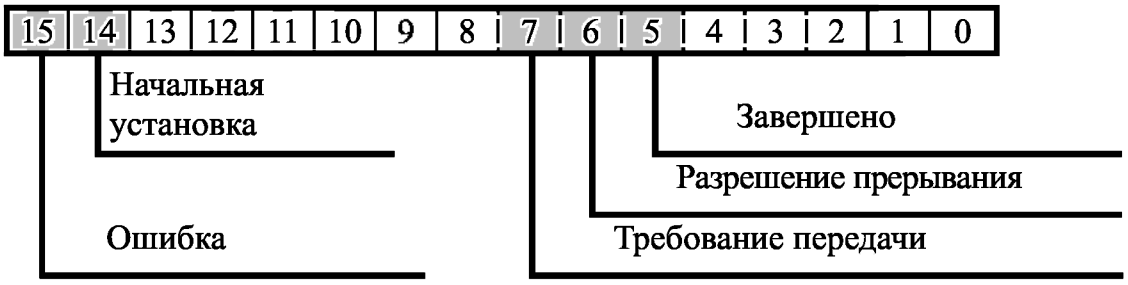


Рис. 6.1. Формат РС ПУ

Разряд ***Ошибка (15)*** доступен только по чтению (вводу). Устанавливается в "1" по приходу сигнала "Ошибка".

Разряд ***Начальная установка (14)*** доступен только по записи (выводу). При записи "1" в этот разряд возникает сигнал "Сброс ВУ".

Разряд ***Требование передачи (7)*** доступен только по чтению (вводу). Устанавливается в "1" при наличии сигнала "Требование передачи".

Разряд ***Разрешение прерывания (6)*** доступен по чтению и записи (вводу и выводу). Сбрасывается начальным сигналом "Сброс". При установке в 7-м и 6-м разрядах "1" устройство выставляет сигнал "Требование прерывания".

Разряд ***Завершено (5)*** доступен только по чтению (вводу). Устанавливается в "1" по приходу сигнала "Завершено".

Из рассмотрения РС ПЧУ следует, что код, разрешающий чтение (вывод) информации, следующий:

$$(10100000) = 240_8.$$

В 16-разрядном формате регистра данных ПЧУ используется только младший байт, т.е. разряды 0 - 7. Этот регистр доступен только по записи. При записи в младший байт РД ПЧУ на шинах данных появляется записанная информация, которая может быть выведена на печатающее устройство, если состояние РС ПЧУ соответствует разрешенному коду, т.е.

$$< \text{РС ПЧУ} > = 240_8.$$

Адреса регистров:

РС 177514.

РД 177516.

6.4.2. Управляющие символы

Управляющие символы приведены в табл. 6.1.

Таблица 6.1

Управляющие символы

Код управляющего символа (в 10 с/с)	Обозначение	Наименование
10	ПС	Перевод строки
13	ВК	Возврат каретки
14	ВЫХ	Выход
15	ВХ	Вход
19	СУЗ	Символ устройства
20	СТП (СУ4)	Символ устройства
24	АН	Аннулирование
27	АР2	Авторегистр 2

Переходы между наборами осуществляются кодами управляющих символов СУЗ, СТП, ВХ и ВЫХ.

В табл. 6.2 определяется управляющий символ, необходимый для перехода от исходного набора к требуемому.

Таблица 6.2

Управляющий символ для перехода от исходного набора к требуемому

Исходный набор	Устанавливаемый набор				
	КОИ-8	КОИ-7НО	КОИ-7Н1	КОИ-7Н2	ШП-8
КОИ-8	СУЗ, ВХ	-	-	СТП	ВЫХ
КОИ-7НО	СУЗ	ВХ	ВЫХ	СТП	-
КОИ-7Н1	СУЗ	ВХ	ВЫХ	СТП	-
КОИ-7Н2	СУЗ	ВХ	ВЫХ	СТП	-
ШП-8	ВХ	-	-	СТП	СУЗ, ВЫХ

Требуемый управляющий символ читается на пересечении строки, определяющей исходный набор, и столбца, определяющего требуемый набор.

6.4.3. Разработка алгоритма программного управления ПЧУ

При управлении внешним, в частности, печатающим устройством необходимо знать следующее:

1) код управляющего символа, который инициирует или изменяет действие, влияющее на регистрацию, обработку или интерпретацию данных (см. табл. 6.1);

2) код выводимого графического символа, предназначенного для представления данных;

3) код регистра состояния печатающего устройства.

Если <РС ПЧУ> = $(240)_8$, возможен вывод символа по первому пункту в соответствии со вторым пунктом.

Вывод информации на ПЧУ начинается после заполнения буферного регистра. Емкость буферного устройства равна 128 б. Поэтому вывод каждого символа следует завершать командой "ВК" (код 15_8).

КОИ-7Н0

Номера							
разрядов	C7	0	0	1	1	1	1
	C6	1	1	0	0	1	1
	C5	0	1	0	1	0	1

C7	C6	C5	C4	C3	C2	C1	N	02	03	04	05	06	07
			0	0	0	0	00	Пр	0	@	P	'	p
			0	0	0	1	01	!	1	A	Q	a	q
			0	0	1	0	02	"	2	B	R	b	r
			0	0	1	1	03	#	3	C	S	c	s
			0	1	0	0	04	¤	4	D	T	d	t
			0	1	0	1	05	%	5	E	U	e	u
			0	1	1	0	06	&	6	F	V	f	v
			0	1	1	1	07	'	7	G	W	g	w
			1	0	0	0	08	(	8	H	X	h	x
			1	0	0	1	09	)	9	I	Y	i	y
			1	0	1	0	10	*	:	J	Z	j	z
			1	0	1	1	11	+	;	K	[	k	{
			1	1	0	0	12	,	<	L	\	l	/
			1	1	0	1	13	-	=	M	]	m	}
			1	1	1	0	14	.	>	N	¬	n	Ï
			1	1	1	1	15	/	?	O	_	o	

Рис. 6.2. Таблица для кодирования графических символов набора 0

КОИ-7Н1

Номера							
разрядов	C7	0	0	1	1	1	1
	C6	1	1	0	0	1	1
	C5	0	1	0	1	0	1

C7	C6	C5	C4	C3	C2	C1	N	02	03	04	05	06	07
			0	0	0	0	00	Пр	0	ю	п	Ю	П
			0	0	0	1	01	!	1	а	я	А	Я
			0	0	1	0	02	"	2	б	р	Б	Р
			0	0	1	1	03	#	3	ц	с	Ц	С
			0	1	0	0	04	¤	4	д	т	Д	Т
			0	1	0	1	05	%	5	е	у	Е	У
			0	1	1	0	06	&	6	ф	ж	Ф	Ж
			0	1	1	1	07	'	7	г	в	Г	В
			1	0	0	0	08	(	8	х	б	Х	Ь
			1	0	0	1	09	)	9	и	ы	И	Ы
			1	0	1	0	10	*	:	й	з	Й	З
			1	0	1	1	11	+	;	к	ш	К	Ш
			1	1	0	0	12	,	<	л	э	Л	Э
			1	1	0	1	13	-	=	м	щ	М	Щ
			1	1	1	0	14	.	>	н	ч	Н	Ч
			1	1	1	1	15	/	?	о	ъ	О	

Рис. 6.3. Таблица для кодирования графических символов набора 1

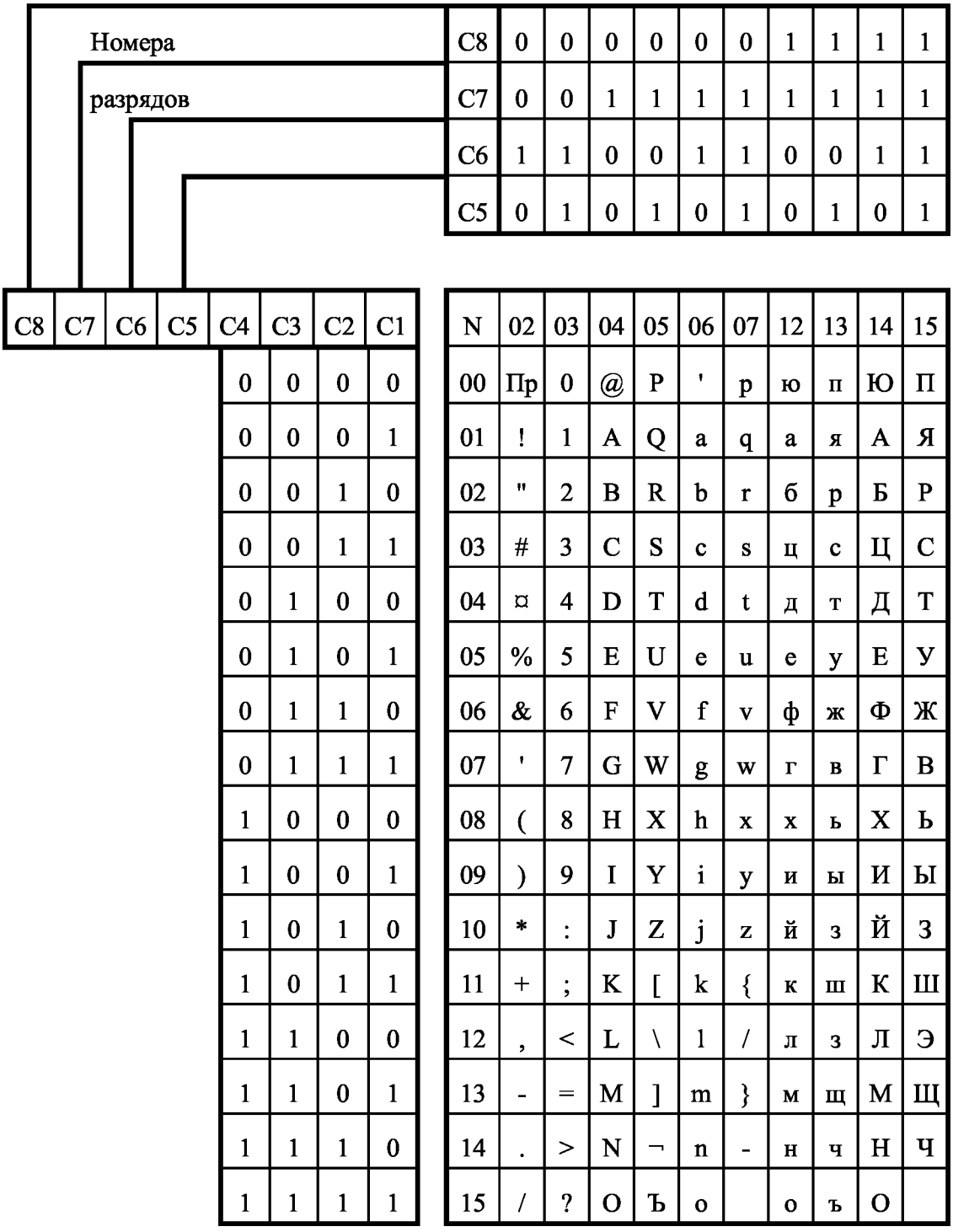
КОИ-7Н2

Номера							
разрядов	C7	0	0	1	1	1	1
	C6	1	1	0	0	1	1
	C5	0	1	0	1	0	1

C7	C6	C5	C4	C3	C2	C1	N	02	03	04	05	06	07
			0	0	0	0	00	Пр	0	@	P	Ю	П
			0	0	0	1	01	!	1	A	Q	А	Я
			0	0	1	0	02	"	2	B	R	Б	Р
			0	0	1	1	03	#	3	C	S	Ц	С
			0	1	0	0	04	¤	4	D	T	Д	Т
			0	1	0	1	05	%	5	E	U	Е	У
			0	1	1	0	06	&	6	F	V	Ф	Ж
			0	1	1	1	07	'	7	G	W	Г	В
			1	0	0	0	08	(	8	H	X	Х	Ь
			1	0	0	1	09	)	9	I	Y	И	Ы
			1	0	1	0	10	*	:	J	Z	Й	З
			1	0	1	1	11	+	;	K	[	К	Ш
			1	1	0	0	12	,	<	L	\	Л	Э
			1	1	0	1	13	-	=	M	]	М	Щ
			1	1	1	0	14	.	>	N	¬	Н	Ч
			1	1	1	1	15	/	?	O	Ъ	О	

Рис. 6.4. Таблица для кодирования графических символов набора 2

КОИ-8



	C8	0	0	0	0	0	0	1	1	1	1
	C7	0	0	1	1	1	1	1	1	1	1
	C6	1	1	0	0	1	1	0	0	1	1
	C5	0	1	0	1	0	1	0	1	0	1
C8 C7 C6 C5 C4 C3 C2 C1	N	02	03	04	05	06	07	12	13	14	15
0 0 0 0	00	Пр	0	@	P	'	p	ю	п	Ю	П
0 0 0 1	01	!	1	A	Q	a	q	а	я	А	Я
0 0 1 0	02	"	2	B	R	b	r	б	р	Б	Р
0 0 1 1	03	#	3	C	S	c	s	ц	с	Ц	С
0 1 0 0	04	¤	4	D	T	d	t	д	т	Д	Т
0 1 0 1	05	%	5	E	U	e	u	е	у	Е	У
0 1 1 0	06	&	6	F	V	f	v	ф	ж	Ф	Ж
0 1 1 1	07	'	7	G	W	g	w	г	в	Г	В
1 0 0 0	08	(	8	H	X	h	x	х	ь	Х	Ь
1 0 0 1	09	)	9	I	Y	i	y	и	ы	И	Ы
1 0 1 0	10	*	:	J	Z	j	z	й	з	Й	З
1 0 1 1	11	+	;	K	[	k	{	к	ш	К	Ш
1 1 0 0	12	,	<	L	\	l	/	л	з	Л	Э
1 1 0 1	13	-	=	M	]	m	}	м	щ	М	Щ
1 1 1 0	14	.	>	N	¬	n	-	н	ч	Н	Ч
1 1 1 1	15	/	?	O	Ъ	o		о	ъ	О	

Рис. 6.5. Таблица для кодирования графических символов набора 8

Задание

1. Выполнить тестирование печатающего устройства (по п. 6.2).
2. Написать программы на языке БЕЙСИК для распечатки таблиц символов и соответствующих кодов наборов КОИ-7НО, КОИ-7Н1, КОИ-7Н2, КОИ-8, используя оператор LPRINT с функцией CHR¤.
3. Распечатать листинг программы.
4. Составить программу для вывода на печать фамилии, имени, отчества, не используя оператор LPRINT.
5. Распечатать листинг программ.

7. КОДИРОВАНИЕ АЛФАВИТНО-ЦИФРОВОЙ ИНФОРМАЦИИ И УПРАВЛЕНИЕ ПЕЧАТАЮЩИМ УСТРОЙСТВОМ С ПОМОЩЬЮ МИКРОЭВМ СИСТЕМЫ INTEL

7.1. Основные технические характеристики печатающих устройств

В настоящее время разработано много моделей принтеров, предназначенных для использования в составе ПЭВМ системы INTEL (IBM PC AT).

Большой популярностью при относительно небольшой стоимости пользуются принтеры семейства EPSON (серия FX). Имея в своем составе программно-загружаемый знакогенератор, они обеспечивают печать алфавитно-цифровой и графической информации и могут формировать любые наборы символов.

В принтерах используется режим точечной графики, что позволяет печатать символы различной конфигурации, чертежи, рисунки и т.п. Электрические сигналы приводят в действие 9 вертикальных штырьков, расположенных на печатающей головке. Наличие сигнала вызывает удар штырька, и получается точка на бумаге. При возбуждении всех штырьков печатается вертикальная линия.

Перемещая головку в горизонтальном направлении, можно напечатать различные изображения в полосе, состоящей из 9 точек по вертикали и n точек по горизонтали, причем значение n определяется моделью принтера.

Ширина бумаги для принтера FX-800 – 254 мм, для FX-1000 – 406 мм. Плотность печати в вертикальном направлении составляет 72 точки на дюйм, а в горизонтальном направлении может быть 60, 120 и 240 точек на дюйм (1 дюйм = 25,4 мм).

Скорость печати зависит от модели принтера: для FX-85 она составляет 160 знаков в секунду, для FX-800/1000 – до 240. Принтер обеспечивает одно- и двунаправленную печать. Во втором случае символы печатаются при перемещении головки слева направо и справа налево.

Для модели FX-800/1000 знаки из стандартной таблицы поддерживаются знакогенератором принтера и имеют высоту 3,1 мм. Ширина одного знака – от 1,05 до 2,1 мм. Стандартный интервал между строками составляет 1/6 дюйма. Максимальное число знаков в строке – до 160 у FX-800 и 272 у FX-1000. Допускается изготовление нескольких копий через копировальную бумагу.

7.2. Кодирование алфавитно-цифровой информации

Зарубежные компьютеры, совместимые с ЭВМ IBM PC, имеют единую кодировку символов, т.е. таблицу кодов, в которой каждому изображаемому на экране символу соответствует код от 0 до 255. Однако в этой таблице отсутствуют символы кириллицы (русские буквы), поэтому в бывшем СССР и Болгарии созданы различные модификации таблицы кодов IBM, содержащие символы кириллицы.

Следует учитывать, что эти кодировки не согласуются между собой и, как следствие, программы, выдающие сообщения на русском языке, при переносе на компьютер с другой кодировкой работают неправильно. Кодировки символов с кодами 0 - 127 (управляющие коды, латинские буквы, цифры, знаки пунктуации и т.д.) совпадают с кодировкой IBM на основании ASCII-кода (American Standart Code for Information Interchange – Американский стандартный код для обмена информацией, внедрен в 1963 г.). В связи с этим программы, которые выводят на экран сообщения на английском языке, работают одинаково независимо от того, какая кодировка символов используется в компьютере.

Коды от 0 до 31 и 127 являются управляющими (табл. 7.1).

Таблица 7.1

Управляющие коды для символов от 0 до 31 и 127

Коды		Символ	Клавиши	Назначение
10 с/с	16 с/с			
0	00	NUL	@	Машинный нуль
1	01	SON	A	Начало заголовка (Start of Heading)
2	02	STX	B	Начало текста (Start of Text)
3	03	ETX	C	Конец текста (End of Text)
4	04	EOT	D	Конец передачи (End of Transmission)
5	05	EMQ	E	Справка (Enquiry)
6	06	ASK	F	Подтверждение (Acknowledge)
7	07	BEL	G	Звонок (Bell)
8	08	BS	H	Шаг назад (Back space)
9	09	HT	I	Горизонтальная табуляция (Tab Horizontally)
10	0A	LF	J	Перевод строки (Line Feed)
11	0B	VT	K	Вертикальная табуляция (Tab Vertically)
12	0C	FF	L	Подача формата (Form Feed)
13	0D	CR	M	Возврат каретки (Carridge Return)
14	0E	SQ	N	Внешнее перемещение (Shift Out)
15	0F	SI	O	Внутреннее перемещение (Shift In)
16	10	DLE	P	ESC-последовательность (Data Line Escape)
17	11	DC1	Q	Управление 1 (Device Control 1)
18	12	DC2	R	Управление 2 (Device Control 2)
19	13	DC3	S	Управление 3 (Device Control 3)
20	14	DC4	T	Управление 4 (Device Control 4)
21	15	NAK	U	Отрицательное подтверждение (Negative Acknowlege)
22	16	SIN	V	Синхронизация (Synchronous Indl)
23	17	ETB	W	Конец передаваемого текста (End of Transmited Block)
24	18	CAN	X	Отмена линии (Cancel Line)
25	19	EM	Y	Переход через середину (End of Medium)
26	1A	SUB	Z	Конец текстового файла (Substitute)
27	1B	ESC	I	Символ ESC (Escape)
28	1C	FS	\	Разделитель файла (File Separator)
29	1D	GS	I	Разделитель групп (Group Separator)
30	1E	RS	^	Разделитель записей (Recorol Separator)
31	1F	US	-	Разделитель единиц (Unit Separator)
127	7F	DEL	8	Забой (удаление символа) (Delete)

Для некоторых команд используются управляющие ASCII-коды, например: 7 – звонок, 10 – перевод строки, 13 – возврат каретки, 24 – отмена строки, 127 – забой знака. Для большинства из них формируются так называемые ASC-последовательности, т.е. наборы байтов, первый из которых имеет код ESC (десятичный номер 27); затем следует байт с кодом команды, которым может быть любой символ ASC-кода; затем – байты с параметрами команды (если они необходимы).

С помощью ESC-последовательностей можно устанавливать большое число различных шрифтов, задавать графический режим, определять любые собственные знаки, выравнивать текст на странице, устанавливать интервал между строками и символами, производить табуляцию, выбирать различные наборы знаков для расширенной таблицы ASCII-кода и т.д.

Символы с кодами от 32 до 126 приведены в табл. 7.2. Они являются также общими для всех кодировок.

Таблица 7.2

Управляющие коды для символов от 32 до 126

Код	Символ	Код	Символ	Код	Символ	Код	Символ	Код	Символ	Код	Символ
32	Про	48	0	64	@	80	P	96	'	112	p
33	!	49	1	65	A	81	Q	97	a	113	q
34	"	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	'	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	|
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	Ï
47	/	63	?	79	O	95	_	111	o		

Из пяти наиболее распространенных кодировок с использованием кириллицы наиболее удачными считаются альтернативная кодировка ГОСТа и модифицированная альтернативная кодировка ГОСТа, в которых символы кириллицы расположены на тех позициях, где в кодировке IBM располагаются относительно редко используемые символы национальных алфавитов и греческие буквы.

Альтернативная кодировка ГОСТа и модифицированная альтернативная кодировка ГОСТа отличаются, в основном, кодами с 242 по 255.

В данной работе используется альтернативная кодировка (символы с кодами 128 - 255), в которой символы кириллицы имеют коды: А - Я (128 - 159); а - п (160 -175); р - я (224 - 239). Псевдографические символы находятся на тех же местах, что и в кодировке IBM. Символы с кодами 240 - 255 отличаются от соответствующих символов в кодировке IBM. (Алфавитная кодировка ГОСТа приведена в книге: Фигурнов В.Э. IBM PC для пользователя. – 6-е изд., перераб. и доп. – М.: ИНФРА-М, 1995. – С. 421, табл. П.5.3).

7.3. Управление печатающим устройством

7.3.1. Порты и регистры

Управление печатающим устройством, которое относится к периферийным устройствам (ПУ) компьютера, осуществляется так же, как и любым внешним устройством, подключенным к системной шине.

Принтеры, как правило, являются параллельными устройствами. Печатающие устройства серии FX не составляют исключения и для подключения к системному блоку используют параллельный интерфейс, в соответствии с которым информация передается побайтно.

Операционная система MS DOS может работать с несколькими параллельными устройствами, каждое из которых имеет имя LPTn (n = 1,2...) и свой адаптер, управляемый тремя регистрами ввода/вывода: ***регистром данных***, ***регистром статуса*** и ***регистром управления***. Адреса портов этих регистров различны для каждого адаптера. Базовые адреса для каждого адаптера находятся в области данных BIOS (базовая система ввода-вывода MS DOS, находящаяся в постоянной памяти компьютера). В табл. 7.3 даны адреса портов LPTn.

Таблица 7.3

Адреса портов LPTn

Адрес в 16 с/с	Количество байтов	Содержимое памяти
0:0408	2	Адрес порта LPT 1
0:040A	2	Адрес порта LPT 2
0:040C	2	Адрес порта LPT 3
0:040E	2	Адрес порта LPT 4

Напомним, что адрес занимает 4 байта: 2 байта со старшими адресами определяют сегмент, 2 байта с младшими адресами – смещение. В табл. 7.3 смещение взято относительно сегмента с адресом 0.

Следует отметить, что при инициализации базовому адресу присваивается значение 0, когда соответствующий адаптер не установлен. Программа, которая прямо адресуется в параллельный порт, должна выискивать используемые им адреса.

7.3.2. Регистр данных

Каждый посылаемый в принтер байт передается через регистр данных, имеющий 8 разрядов. Программа пользователя должна вычислить номер порта выходных данных в зависимости от выбранного LPTn.

Если выбрать канал LPT1, то тогда согласно табл. 7.3, необходимо обратиться в область BIOS памяти по адресу 0:0408. Прочитанное определенным образом значение после дополнительного преобразования даст номер порта выходных данных, т.е.

$$NPORT\ (D) = [<0:0408>], \qquad (7.1)$$

где NPORT (D) – номер порта данных;

< > – угловые скобки, указывают содержимое ячейки по адресу;

[ ] – квадратные скобки, указывают на дополнительное преобразование.

На рис. 7.1 изображены 4 байта, где показаны сегмент и смещение и их адреса соответственно 0 и 0408 для LPT1.

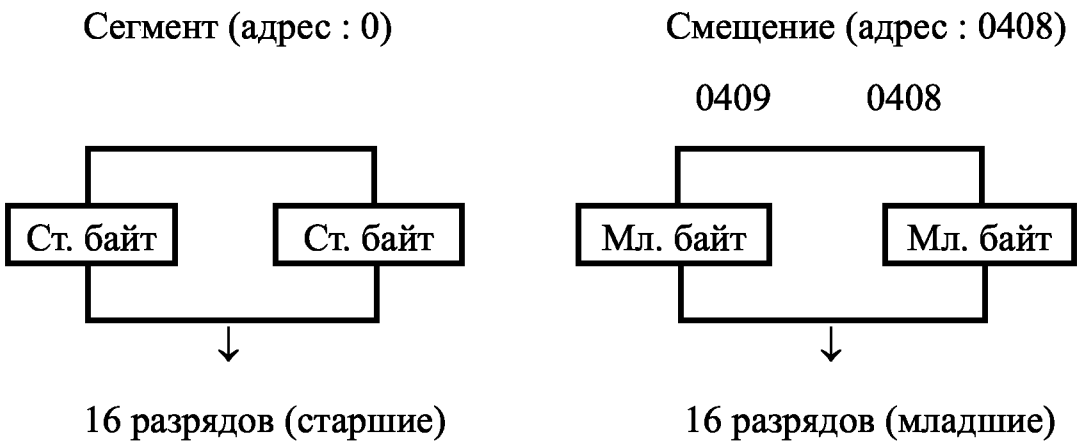


Рис. 7.1. Адреса сегмента и смещения

Из рис. 7.1 видно, что в адресе, указанном в табл. 7.3 для LPT1, в качестве смещения приводится адрес самого младшего байта, а именно: 0408. Адрес второго младшего байта автоматически считается равным 0409. Очевидно, что младший разряд байта с адресом 0409 больше единицы младшего разряда байта с адресом 0408 в $2^8 = 256$ раз.

Таким образом, если содержимое младшего байта с адресом 0408 обозначить через PORT (D1), а содержимое байта с адресом 0409 – через PORT (D2), то

$$\text{NPORT (D)} = \text{PORT (D1)} + \text{PORT (D2)} \cdot 256, \qquad (7.2)$$

причем

$$\text{PORT (D1)} = < 0408 >;$$

$$\text{PORT (D2)} = < 0409 >.$$

Формула (7.2) показывает алгоритм дополнительного преобразования, который содержится в формуле (7.1).

Таким образом, для определения номера порта, в частности, LPT1, необходимо:

1) задать текущий адрес сегмента (в БЕЙСИКе это делается с помощью оператора DEF SEG = адрес, например: нс DEF SEG = &H0);

2) определить содержимое PORT (D1);

3) определить содержимое PORT (D2);

4) определить номер порта данных PORT (D) по формуле (7.2).

Для определения содержимого PORT (D1) и PORT (D2) надо использовать функцию PEEK.

7.3.3. Регистр статуса

Регистр статуса (РС) сообщает различную информацию. Он имеет 8 разрядов. На рис. 7.2 показан формат РС.

Разряды с 0 - 2 не используются.

3-й разряд. Ошибка принтера: 0 – ошибка ; 1 – отсутствие ошибки.

4-й разряд. Связь ПЭВМ: 0 – принтер не связан с машиной (off-line); 1 – принтер связан с машиной (on-line).

Рис. 7.2. Формат РС

5-й разряд. Бумага: 0 – бумага вставлена; 1 – нет бумаги.

6-й разряд. Подтверждение приема символа: 0 – принтер подтверждает прием символа; 1 – нормальная установка.

7-й разряд. Занято: 0 – принтер занят; 1 – принтер свободен.

Неиспользуемые разряды 0-2 имеют единичное состояние. Если принтер выключен, то все разряды, кроме 3-го, установлены в единице, что соответствует коду $F7_{16}$. Если принтер не готов к работе (7-й разряд в "0"), то 3-й разряд имеет "0", что соответствует коду 57_{16}. Если отсутствует бумага, то код будет 77_{16}.

Двоичный код готовности принтера к работе: $11011111_2 = DF_{16}$.

Перед обращением к принтеру необходимо проводить анализ на его готовность. Это можно делать в БЕЙСИКе с помощью оператора WAIT ("Ждать"). Формат оператора:

WAIT A, n1.

Этот оператор приостанавливает процесс выполнения программы до тех пор, пока в порту с адресом А не появится кодовая комбинация, удовлетворяющая n1, которая задает условие прекращения задержки.

Номер порта регистра статуса вычисляется по формуле:

$$NPORT\ (S) = NPORT\ (D) + 1\ , \qquad (7.3)$$

т.е. номер порта регистра статуса R'S (NPORT (S)) отличается от номера порта регистра данных на единицу в сторону увеличения.

Тогда оператор WAIT можно записать следующим образом:

$$WAIT\ NPORTS,\ \&HDF. \qquad (7.4)$$

При выполнении этого оператора может быть начата передача данных в регистр данных. Однако до появления этого оператора в программе пользователя необходимо выполнить определенные операции с принтером, то есть инициализировать принтер через регистр управления.

7.3.4. Регистр управления

Регистр управления (РУ) устанавливает адаптер в исходное состояние и координирует вывод данных. Он имеет 8 разрядов. Формат РУ показан на рис. 7.3.

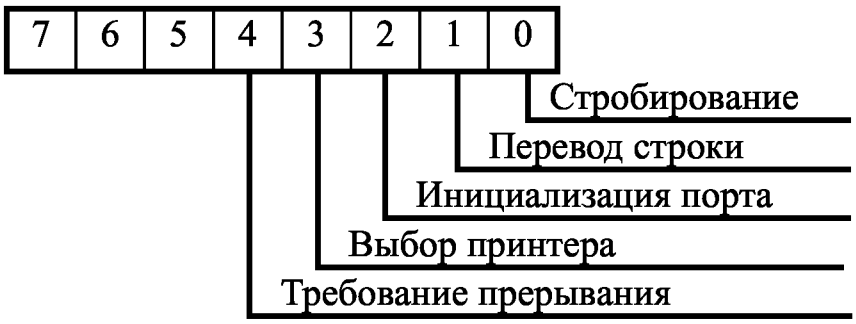


Рис. 7.3. Формат регистра управления

Q-разряд. Стробирование: 0 – нормальная установка; 1 – кратковременное единичное значение, воспринимаемое как стробирующий сигнал для вывода байта.

1-й разряд. Перевод строки: 0 – нормальная установка; 1 – автоматический перевод строки после возврата каретки.

2-й разряд. Инициализация порта принтера: 0 – инициализация; 1 – нормальная установка.

3-й разряд. Выбор принтера: 0 – отмена выбора принтера; 1 – нормальная установка.

4-й разряд. Требование передачи: 0 – прерывание запрещено; 1 – прерывание разрешено.

5-й - 7-й разряды не используются.

Программа пользователя должна инициализировать порт принтера перед первым его использованием; в случае появления ошибки необходимо провести повторную инициализацию.

Не следует путать инициализацию порта принтера с инициализацией самого принтера, которая происходит автоматически при его включении. Программа может повторно инициализировать включенный принтер, при этом устанавливаются начальные параметры работы принтера (принятые по умолчанию), отменяя все специальные шрифты, специальные режимы и т.д., то есть все, что оставила предшествующая программа. Считается правилом хорошего тона производить такой сброс принтера после того, как программа завершает работу. Однако восстановление начальных параметров печати требуется во всех программах.

Эпсоновские принтеры имеют главный код сброса, который приводит к полному сбросу принтера. Для инициализации необходимо в регистре управления сбросить в "0" 2-й разряд (инициализация порта) на 0,05 с. В этот момент в единице должен быть только 3-й разряд (выбор принтера), что равносильно записи в РУ следующего кода:

$$KI1 = 1000_2 = 8_{16}, \qquad (7.5)$$

где KI1 – условно обозначенное первое значение кода для инициализации.

После задержки, которая должна быть не менее 1 - 2 с, 2-й разряд (инициализация) устанавливается в единице (нормальная установка), при этом 3-й разряд остается в единице, что соответствует записи в РУ следующего кода:

$$KI2 = 1100_2 = C_{16}, \qquad (7.6)$$

где KI2 – условно обозначенное второе значение кода при инициализации.

Номер порта регистра управления вычисляется по формуле

$$NPORT\ (U) = NPORT\ (D) + 2, \qquad (7.7)$$

где NPORT (U) – номер порта регистра управления;

NPORT (D) вычислен в (7.2).

Запись в порт РУ кода 8_{16} согласно (7.5), а затем – кода C_{16} согласно (7.6) – осуществляется с помощью специального оператора OUT. Формат этого оператора:

$$OUT\ порт,\ код, \qquad (7.8)$$

где "порт" – номер порта; "код" – значение байта в 16 с/с.

После инициализации в программе следует использовать оператор присваивания номера символа, который хочет напечатать пользователь.

Например:

$$SYMBOL = <код>, \qquad (7.9)$$

где код выбирается по табл. 7.1 - 7.3; его удобно задавать в 10 с/с.

Например:

$$SYMBOL = 33_{10} – восклицательный\ знак.$$

Прежде чем записать значение SYMBOL в порт, необходимо проверить состояние регистра статуса с помощью оператора WAIT (см. формулу (7.4)).

Если оператор WAIT дает разрешение на выполнение последующих операторов программы, следующим должен быть оператор записи в порт данных заданного символа.

Например:

$$OUT\ NPORTD,\ SYMBOL, \qquad (7.10)$$

где NPORTD – имя порта регистра данных в отличие от условного обозначения в (7.2).

Вслед за посылкой байта символа в порт выходных данных необходимо организовать стробирование РУ (см. рис. 7.3). Кратковременная установка нулевого разряда РУ в единице осуществляется

аналогично операции инициализации: в порт РУ посылаются друг за другом два кода:

$$\text{КСТР1} = 1100\ + 1 = 1101\ = D_{16};$$
$$\text{КСТР2} = 1100\ = C_{16}. \qquad (7.11)$$

После посылки байта и включения стробирующего сигнала программа должна ждать сигнала о готовности принтера (ожидание через (7.4)). Подтверждающий сигнал выражается не только в изменении 7-го разряда (занято); дополнительно на короткое время будет сброшен в ноль 6-й разряд регистра статуса, что свидетельствует о подтверждении приема сигнала принтером.

После этого может быть послан следующий байт символа. Если символы выводятся подряд, следующим оператором может быть "подготовка":

$$\text{SYMBOL} = \text{SYMBOL} + 1. \qquad (7.12)$$

Для наглядности программы можно использовать оператор вывода следующего символа на экран через PRINT.

7.3.5. Алгоритм управления печатающим устройством для вывода символов в цикле

Вывод символов в цикле производится в следующей последовательности:

1. Определение адреса сегмента.
2. Вычисление номеров портов регистра данных, регистра статуса и регистра управления (формулы: (7.2), (7.3), (7.7)).
3. Инициализация порта управления с помощью оператора (7.8) для записи двух кодов ((7.5) и (7.6)).
4. Выбор символа для печати (7.9).
5. Ожидание готовности (7.4).
6. Запись символа в регистр выходных данных с помощью (7.8).
7. Организация стробирующего сигнала (7.11).
8. Ожидание готовности (7.4).
9. Подготовка следующего символа (7.12).
10. Вывод на экран (PRINT).

11. Формирование условия распечатки количества символов в строке листа.

12. Формирование условия окончания вывода символов.

При этом необходимо учитывать следующее:

1. При выполнении условий 11 и 12 должна формироваться управляющая последовательность для принтера с помощью LPRINT и функции CHR$: "Символ ESC "(27); "Возврат каретки" (13); "Перевод строки" (10) (см. также табл. 7.1). Функции пишутся подряд без разделительных знаков или через знак "+".

2. После выполнения условия 11 и сформированной управляющей последовательности управление передается к п. 5.

3. При управлении ПУ FX-800 пункт 3 необходимо выполнять, а при FX-1000 – использовать как комментарии.

Задание

1. Составить программу для вывода на принтер символов согласно табл. 7.2 (коды 32-126, 128-255) без использования LPRINT. Сравнить выведенные коды (128-255) с табл. П.5.3 (из источника: Фигурнов В.Э. IBM PC для пользователя. – 6-е изд., перераб. и доп. – М.: ИНФРА-М, 1995).

2. Составить программу для вывода на печать фамилии, имени, отчества, не используя оператор LPRINT.

Содержание отчета

1. Листинги программы.

2. Распечатка результатов.

8. УПРАВЛЕНИЕ ГРАФИЧЕСКИМ ОЗУ

8.1. Графическое ОЗУ

Емкость графического ОЗУ составляет 16 К восьмибитовых слов. Конструктивно графическое ОЗУ выполнено отдельно от основной памяти микроЭВМ и размещается на плате-модуле контроллера графики (КГР) (см. структурную схему микроЭВМ "Электроника ДВК-3").

МикроЭВМ, имеющая КГД, может работать при наличии графического видеотерминала (видеомонитора) в режиме как символьных, так и графических изображений. Экран видеотерминала имеет 400 столбцов (0-399) и 300 строк (0-299), но гарантированными являются 286 строк (0-285).

При работе с графическим пакетом любого языка высокого уровня используются специальные операторы, каждый из которых реализуется большей или меньшей совокупностью машинных команд в кодах.

Например, оператор БЕЙСИКа

PSET (a, в) <вк>

выводит на экран точку с координатами (a, b), где "a" – номер столбца; "b" – номер строки.

Зная принципы управления графическим ОЗУ, пользователь может строить любые графические изображения, не используя специальные операторы графических пакетов.

Для облегчения изучения принципов управления графическим ОЗУ можно воспользоваться функцией PEEK и оператором POKE языка БЕЙСИК, обеспечивающими доступ к ячейкам памяти. Хотя при этом скорость управления значительно уменьшается по сравнению с машинным языком или АССЕМБЛЕРом, общность рассуждений при этом не нарушается.

8.2. Форматы регистров

Управление графическим ОЗУ осуществляется 16-разрядными регистрами, доступными по записи и чтению. Регистры (регистр управления РУ, регистр адреса РА, регистр данных РД и регистр счета слов РСС) и их адреса приведены в табл. 8.1 (РСС программистом не используется).

Таблица 8.1

Форматы регистров РУ, РД, РА

Регистр	Адрес в 8 с/с
РУ	176640
РД	176642
РА	176644

8.2.1. Формат регистра РУ

Формат регистра РУ показан на рис. 8.1 а. В нем используются 15-й и 14-й разряды.

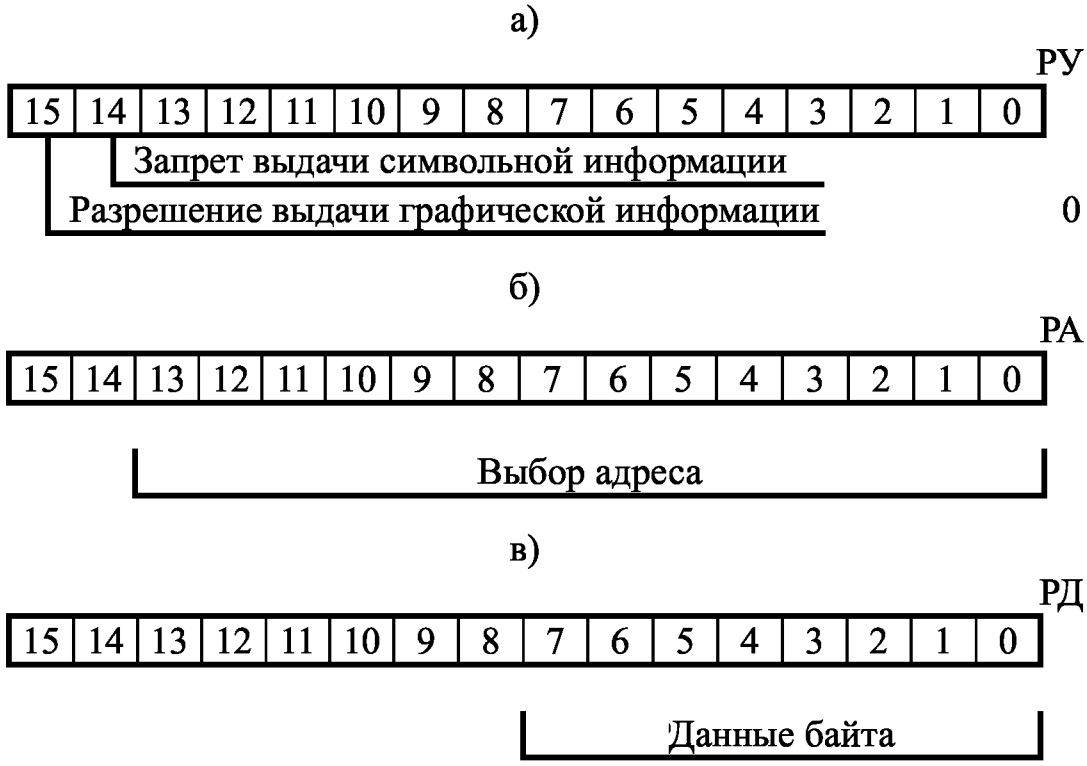


Рис. 8.1. Форматы регистров

15-й разряд. При записи единицы в этот разряд на экран видеотерминала выдается содержимое графического ОЗУ. При записи нуля графическая информация не отображается на экране.

14-й разряд. При записи единицы в этот разряд запрещается вывод символьной информации на экран видеотерминала. При записи нуля символьная информация отображается на экране. Таким образом, для вывода графической и символьной информации в РУ надо занести код 100000_8.

8.2.2. Формат регистра РА

В формате регистра РА (см. рис. 8.1 б) используется 14 разрядов (0?13). Его содержимое задается программистом; в него записывается абсолютный номер байта. Абсолютные адреса байтов (или их номера) показаны в табл. 8.2.

Таблица 8.2

Абсолютные адреса (номера) байтов

Номера строк (N стр)	Номера байтов в строке (NB ст)				
0	0	1	...	48	49
1	50	51	...	98	99
...	...	...	...	...	...
285	14250	14251	...	14298	14299

Как видно из табл. 8.2, каждая строка определена 50 номерами байтов ($50 \cdot 8 = 400$ ст.).

Номер байта, то есть его абсолютный адрес, может быть вычислен по формуле

$$NB = 50\ Nстр + NBст, \qquad (8.1)$$

где NB – номер байта в строке;

Nстр – номер строки;

NBст – номер байта в столбце, причем NBст = 0?49.

При работе с языком БЕЙСИК целесообразно указывать NB в 10 с/с.

8.2.3. Формат регистра РД

Байт графической информации определяется разрядами 0?7 (см. рис. 8.1 в). Разряды 8?15 не используются. Вывод точки кодируется с помощью единицы.

Крайней левой точке на экране в каждом байте (см. табл. 8.2) соответствует нулевой (0-й) разряд регистра РД графической информации. Это значит, что по отношению к экрану разряды 7-0 (при обычной записи) следует развернуть на 180 (0?7).

8.3. Определение номера байта но координатам точки

Номер байта в строке может быть определен по координатам заданной точки.

Пусть заданы координаты (a, b). На основании номера столбца "a" может быть найден номер байта столбца по формуле

$$NBст = INT(a/8), \qquad (8.2)$$

где INT – нахождение ближайшего целого.

Получающийся при делении остаток (OST) указывает на номер разряда РД, в который должна быть занесена единица:

$$OST = a - NBст \cdot 8. \qquad (8.3)$$

Пример. Пусть A (a, b) = A (386,2).

1. NBст = INT(386/8) = 48 ; OST=386 – 48 · 8=2.

2. Согласно п.1, NBст = 48; во 2-й разряд регистра РД должна быть занесена единица.

3. NB = 50 · 2 + 48 = 148.

8.4. Алгоритм управления

Управление графическим ОЗУ осуществляется в следующей последовательности:

1. По адресу РУ подается код, разрешающий вывод графической и символьной информации в 8 с/с: 100000_8.

2. Определяется NBст по (8.2), а OST – по (8.3).

3. Вычисляется NB по (8.1).

4. По адресу РА подается код NB в 10 с/с.

5. По адресу РД подается код данных выбранного байта в соответствии с OST, указывающим разряд, в котором должна записываться единица.

Задание

Составить схему алгоритма и программу для вывода на экран дисплея геометрической фигуры.

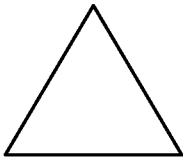
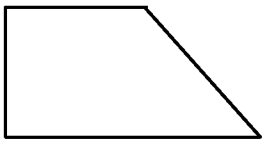
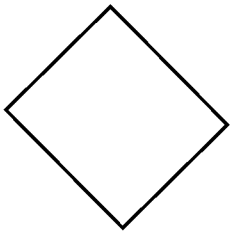
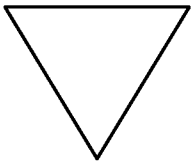
№ варианта	Фигура
1	
2	
3	
4	
5	
6	
7	

Рис. 8.2. Геометрические фигуры для вывода на экран

9. УПРАВЛЕНИЕ ГРАФИЧЕСКИМ ОЗУ МИКРОЭВМ СИСТЕМЫ INTEL

9.1. Краткие сведения о развитии видеоадаптеров системы INTEL

На первой стадии развития персональные компьютеры серии IBM PC комплектовались видеоадаптером MDA (Monochrome Display Adapter) с монохромным дисплеем MD, который имел небольшую разрешающую способность, не мог отображать графическую информацию и был монохромным. Через некоторое время фирма Hercules Computer Technology, Inc. выпустила монохромный видеоадаптер Hercules, который уже имел возможность вывода графики и обеспечивал довольно высокую разрешающую способность.

CGA (Color Graphics Array) стал первым цветным видеоадаптером фирмы IBM. Он был разработан для дешевых и широко распространенных бытовых телевизоров и композитных мониторов: вертикальное разрешение – 200 строк; горизонтальное – 320 строк, хотя на композитных и RGB-мониторах можно использовать 640 строк, т.е. более высокую разрешающую способность. Из-за высокой стоимости памяти разработчики были вынуждены ограничить ее размер. CGA снабжен графической памятью (16 К) для поддержки 4 цветов при разрешении 320x200, но при этом объема памяти не хватает для поддержки более 2 цветов в режиме разрешения 640x200, причем один из них должен быть черным. Даже 4-цветный режим ограничивал пользователя двумя наборами по 4 цвета в каждом, хотя для фона может использоваться любой из цветов.

После своего появления в конце 1984 года усовершенствованный графический адаптер EGA (Enhanced Graphics Array) стал стандартом графики для персональных компьютеров IBM и являлся наиболее популярной дополнительной платой для компьютеров PC, XT, AT.

В апреле 1987 года IBM объявила о начале выпуска компьютеров серии PS/2, оснащенных новым графическим устройством VGA (Video Graphics Array). EGA использует дисплеи цифрового типа (монохромный монитор с разрешением 720x350 и усовершенствованный цветной дисплей ECD с разрешением 640x350) и позволяет использовать 16 цветов из 64. В зависимости от емкости адресуемой памяти для EGA характерны 3 следующих варианта:

1. Емкость 64 К - 16 цветов 4 атрибутам и 2 бита памяти на пиксель.

2. Емкость 128 К – 64 цвета 16 атрибутам (т.е. возможность использовать 16 цветов из 64) и 4 бита памяти на пиксель.

3. Емкость 256 К – аналогично варианту 2, но имеется возможность организации двухэкранных страниц.

VGA очень похож на EGA, но не поддерживает работу с прочими оригинальными дисплеями (за исключением некоторых мультичастотных мониторов). Для работы VGA требуется аналоговый дисплей. Режим высокого разрешения VGA составляет 640x480 и поддерживает использование 16 цветов из 262144 возможных (или 64 градации серого цвета).

В настоящее время VGA-карта является стандартом в области IBM-совместимых компьютеров. Стандарт VGA является базовым для таких стандартов, как Super VGA и HiRes, на его основе разработаны карты-ускорители, например, карты VLB.

VGA-карты совместимы снизу вверх, т.е. способны эмулировать все созданные ранее от MDA до EGA.

Для большинства применений разрешения стандарта VGA вполне достаточно. Однако программы, ориентированные на графику, работают значительно лучше и быстрее, если информационная плотность экрана выше (для этого необходимо повышать разрешение).

Таким образом, стандарт VGA развился в так называемый стандарт Super VGA (SVGA). Стандартное разрешение этого режима составляет 800x600 пикселей.

Отметим закономерность: при объеме видеопамяти 256 кб и SVGA-разрешении можно обеспечить только 16 цветов; 512 кб видеопамяти дают возможность отобразить уже 256 цветовых оттенков при том же разрешении. Карты, имеющие 1 Мб памяти, позволяют при этом же разрешении достичь отображения 32768, 65536 (Hi Color) или даже 16,7 млн. (True Color) цветовых оттенков.

Стандарт HiRes VGA (High Resolution – высокое разрешение) был также разработан фирмой IBM. В режиме 8514/A можно повысить разрешение до 1024x768 пикселей. Обычно при разрешении 1024x768 пикселей ограничена цветовая гамма. Способность монитора или видеокарты поддерживать высокое разрешение существенно влияет на их стоимость, особенно если речь идет о Hi Color и True Color. Обычно для стандарта HiRes характерна поддержка 16 или 256 цветов.

Графические карты XGA (Extended Graphics Array) также оснащены процессором, который берет на себя выполнение графических операций и при наличии большого обьема видеопамяти (1…4 Мб) обладает большой скоростью обработки видеоданных.

Графическое ОЗУ размещается в адаптере видеотерминала. Для VGA его емкость составляет 256 К.

9.2. Организация видеопамяти

Наиболее серьезным архитектурным изменениям в EGA/VGA была подвергнута организация памяти. В CGA для описания каждой точки (пикселя) использовалась последовательность битов данных. Этот метод достаточно прост с точки зрения программирования, но адресное пространство при этом расходуется неоптимально (удвоенное число цветов, удвоенный размер карты памяти).

Ряд нововведений с организацией памяти в виде битовых матриц позволил создать эффективную память со значительно большими возможностями и повышенным быстродействием.

Рассмотрим подробнее организацию памяти EGA/VGA с учетом двух режимов: алфавитно-цифрового и графического.

В процессе работы в алфавитно-цифровом режиме имеющаяся память используется для сохранения ASCII-кодов выводимых символов и их атрибутов (цвет, яркость и/или мерцание). Первые два байта памяти адаптера рассматриваются центральным процессором (ЦП) как место расположения символа в левом верхнем углу экрана, а последующими словами (двухбайтовыми) определяются позиции символов в направлении слева направо и сверху вниз по экрану соответственно. Первый байт каждого слова представляет собой ASCII-код символа, а второй байт разделен на группы, состоящие из 1 и 3 битов, как показано на рис. 9.1.

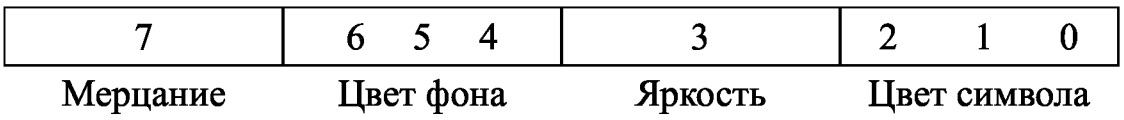


Рис. 9.1. Изображение двух байтов памяти
при работе в алфавитно-цифровом режиме

На цветных мониторах 3 бита, определяющие цвет фона и символа, позволяют получить по 8 цветов. Для монохромного режима

или монохромных мониторов в эти разряды записываются либо 000 (черный цвет), либо 111 (альтернативный цвет, – например, белый).

В графическом режиме память используется для хранения цвета каждой растровой точки (пикселя). Распределение памяти по растровым точкам зависит от вида графического режима и размера памяти. Для случая работы с VGA используется такое же распределение памяти, как для EGA, с объемом памяти 256 К. Во всех режимах пиксели размещаются слева направо и сверху вниз в соответствии с увеличением адресов памяти.

В новом режиме работы по отношению к CGA соотношение дисплей/память представляет собой один байт на пиксель, что позволяет использовать до 256 цветов.

Организация памяти, при которой каждому байту поставлен в соответствие один пиксель, а память не разделена на области четных и нечетных строк развертки, как в CGA, значительно упрощает расчет адресов каждого элемента изображения.

Прочие новые режимы, поддерживаемые EGA/VGA, имеют более простую организацию, чем CGA, однако при этом усложняется запись различных цветов.

Начальным адресом всех новых графических режимов является адрес

$$\text{HA000}. \qquad (9.1)$$

Каждому биту поставлен в соответствие 1 пиксель. Таким образом, каждый байт описывает 8 пикселей. Байт пересылается в видеобуфер монитора, причем 7-й бит соответствует самой левой, а 0-й – самой правой точкам из 8 пикселей.

Вывод точки кодируется единицей, а отсутствие – нулем в соответствующих разрядах байта. Получается, что с помощью одного бита как бы описывается 16 цветов, что объясняется своеобразной организацией EGA/VGA.

Память организована в виде 1-битовых матриц-плоскостей (Bit Planes), которые могут размещаться по одному и тому же адресу: для 4-цветных режимов требуется 2 матрицы ($2^2 = 4$), для 16-цветных – 4 ($2^4 = 16$). Матрицы можно наглядно представить в виде блоков памяти, помещенных один над другим, что позволяет каждой адресуемой ячейке памяти ставить в соответствие 4 бита (по одному

из каждой матрицы). Тогда каждая точка экрана описывается 4 битами. Каждая комбинация матриц определяет цвет (выбранный из 64 возможных цветов), при этом любые комбинации матриц могут быть изменены одновременно с помощью одного из 3 режимов записи на основании 1 байта данных, формируемого процессором.

Организация памяти в виде битовых матриц (плоскостей) удобна по 3 причинам:

1) положение пикселя на экране непосредственно связано с его расположением в памяти (в простейшем случае каждая битовая матрица связана с основным цветом и яркостью);

2) количество цветов может быть удвоено просто за счет добавления одной битовой матрицы (программы, осуществляющие операции прямой записи в память, не нуждаются в пересчете адресов для новых совместимых адаптеров, если добавление цветов выполнено через новые матрицы);

3) модифицирование памяти в 1024-цветном режиме (если бы такой режим поддерживался) выполняется так же быстро, как и в 2-цветном режиме, поскольку модифицирование всех битовых матриц может быть выполнено за одно обращение к памяти.

Однако для управления битовыми матрицами требуются некоторые дополнительные операции. Дополнительные матрицы не принесут много пользы даже в том случае, если все они записываются одновременно.

Адаптер предусматривает два метода для установки цветной точки по команде ЦП (третий метод предназначен для перемещения данных из одной ячейки памяти адаптера в другую с полным сохранением информации о цвете).

Регистр Маскирование Растра предназначен не для определения матриц, в которые записывается единица, а для определения изменяемых матриц. Каждый раз при изменении цвета в регистр Маскирование Растра должно записываться новое значение. Для записи чистых цветов необходимо записать ноль во все битовые матрицы для очистки немодифицированных матриц перед записью новых значений.

Для выполнения этой функции имеется альтернативный вариант, который заключается в использовании режимов Установка/Сброс и Разрешение Установки/Сброса для очистки этих же матриц (см. подраздел 9.3.2).

В связи с тем, что ЦП осуществляет запись полного байта, после каждого обращения к памяти осуществляется изменение состояния 8 пикселей, заключающееся в подключении пикселей, записанных в измененные битовые матрицы, где значение битов равно единице, и отключении пикселей для нулевых значений битов. Такой метод удобен для записи символьных данных в графическом режиме (где каждый символ представлен трафаретом шириной в 8 битов), однако непригоден для построения отдельных пикселей. Поэтому вводится дополнительный регистр.

Регистр Битовой маски может быть использован для выборки отдельных пикселей из байта. Если бит Маски имеет значение "1", возможно изменение значения соответствующего бита каждого байта из 4, и, следовательно, модифицирование пикселя может быть выполнено установкой только 1 бита в Маске. Следует отметить, что пиксель может быть включен или выключен путем установки значения соответствующего бита в единицу и ноль самим ЦП (т.к. установка пикселя в единице не выполняется принудительно регистром Битовой маски).

Использование регистра Маскирования Растра для определения цветов является принятой по умолчанию технологией BIOS и может рассматриваться как ***запись в режиме 0***, так как большинство функций BIOS работает с символьными данными.

В EGA используется второй метод, который называется ***записью в режиме 2*** (здесь не рассматривается).

Однако осложнения, связанные с архитектурой аппаратных средств, – например, отсутствие возможности прямого доступа к памяти адаптера со стороны ЦП, – затрудняют задачу управления графическим ОЗУ. Нужны дополнительные регистры.

Регистры-защелки служат для загрузки текущего содержимого памяти адаптера с целью сохранения немодифицируемых битов. Их количество соответствует количеству битовых матриц. Загрузка в регистры-защелки выполняется путем перемещения данных из памяти адаптера в ЦП. При записи из ЦП в память данные ЦП комбинируются с данными в регистре-защелке и в АЛУ, после чего сохраняются в памяти адаптера. Эта операция, большей частью, скрыта от пользователя, но несколько регистров дают возможность управлять описанным процессом. Дополнительные аппаратные сложности отражены в подразделе 9.3.2.

9.3. Методы управления графическим ОЗУ

Для управления графическим ОЗУ у всех IBM микроЭВМ независимо от типа используемого видеоадаптера предусмотрено 2 режима:

1) управление с использованием BIOS;

2) регистровое управление с использованием прямого доступа в память.

9.3.1. Управление с использованием BIOS

При разработке сложных прикладных программ необходимо принимать во внимание непрерывное наращивание вычислительных мощностей микроЭВМ, поддержку новых аппаратных средств и операционных систем. Программирование с использованием программ BIOS часто представляет собой простой и правильный способ предусмотреть такие модификации. Например, обращения к BIOS, написанные для CGA, будут работать на EGA и VGA, чего нельзя сказать о регистровых функциях. Для изображения пикселя с помощью BIOS используется тот же метод, для которого применяется комбинация адаптер/разрешение, хотя запись в память у CGA и EGA/VGA сильно отличается.

Программирование с использованием BIOS позволяет также расширить степень совместимости между компьютерами и адаптерами, изготовленными различными фирмами. Некоторые регистры EGA и VGA могут быть изменены только в определенных временных интервалах или должны восстанавливаться в интервалах между записями. Эта задача решается с помощью BIOS автоматически.

BIOS EGA предусматривает 20 основных программ для работы с экраном. BIOS EGA/VGA предусматривает новые дополнительные функции и подфункции, поддерживаемые EGA и VGA, с помощью которых может быть решен широкий спектр проблем – от модификации палитры и символа до получения информации о конфигурации адаптера.

Однако управление через BIOS-программы имеет недостатки:

1) BIOS не гарантирует полной совместимости между компьютерами даже с помощью модификации кодов, выполненных в различных операционных системах;

2) несмотря на простоту обращений к BIOS, очень большое число программ BIOS характеризуется невысоким быстродействием.

Преимуществом второго метода управления является устранение вышеуказанных недостатков. Поэтому с целью повышения эффективности работы программ применяются методы прямого доступа к регистрам и памяти.

Для эффективного использования регистрового программирования и использования методов прямого доступа к памяти необходимо хорошо разбираться в вопросах организации видеопамяти адаптера.

9.3.2. Регистровое управление

EGA/VGA имеют регистры, предназначенные для реализации управляющих функций адаптеров.

Регистры адаптера можно разделить на 5 основных групп:

1) внешние регистры;

2) регистры Указателя последовательности;

3) регистры Контроллера электронно-лучевой трубки (КЭЛТ);

4) регистры Графического контроллера;

5) регистры Атрибута.

Внешние регистры предназначены для работы с функциями BIOS.

С помощью ***регистров Указателя последовательности*** осуществляется управление доступом к памяти, синхронизацией и потоком данных между другими регистрами. ***КЭЛТ*** регулирует длительность временного интервала для вывода информации. Реализация функций графического режима производится ***Графическим контроллером***. ***Регистры Атрибута*** осуществляют управление цветовыми палитрами. Для VGA к вышеуказанным регистрам добавляется группа ***регистров цифро-аналогового преобразователя*** (ЦАП), которые служат для преобразования номера цвета в напряжение для аналогового монитора.

При реализации управляющих функций адаптеров возникают дополнительные аппаратные сложности, обусловленные ограниченностью адресного пространства портов семейства IBM PC. Поэтому ко многим регистрам можно обратиться косвенно через адресные или другие регистры. По аналогии с битовыми матрицами, образующими "слои" памяти, можно говорить о "слоях" регистров (хотя за один раз возможно изменение только одного индексного регистра).

Таким образом, EGA и VGA используют косвенный режим адресации для обращения к регистрам и памяти. В некоторых случаях ссылка выполняется на глубину до 3 уровней: адрес указателя дает второй адрес, указывающий на таблицу значений.

Каждая группа регистров (за исключением внешних) имеет адресный регистр (Ai), который используется для выбора модифицируемого регистра. Все безадресные регистры (кроме внешних и регистров Графической позиции) имеют индекс, который записывается в адресный регистр, после чего разрешается доступ к нужному регистру.

Например, Графический контроллер (которому уделено наибольшее внимание в данной работе) имеет 10 регистров: регистр Адреса графики 1 и 2; регистр Установки/Сброса (индекс "0"); регистр Разрешения Установки/Сброса (индекс "1"); регистр Сравнения цветов (индекс "2"); Циклический сдвиг данных (индекс "3"); регистр Выбор схемы чтения (индекс "4"); регистр Режим (индекс "5"); регистр Смешанный (индекс "6"); регистр Цвет безразличен (индекс "7"); регистр Битовой маски (индекс "8").

При регистровом управлении необходимо восстанавливать регистры-защелки перед каждой операцией записи памяти, что осуществляется путем чтения содержимого памяти перед записью нового значения. Нет необходимости использовать считанную информацию (это просто способ, обеспечивающий обновление содержимого регистров-защелок).

Фактически регистры-защелки и АЛУ обеспечивают интерфейс между памятью и ЦП, а группа из 4 Сдвиговых регистров (или параллельно-последовательных преобразователей) образует интерфейс между адаптером и дисплеем. При выполнении обычных графических операций каждый из Сдвиговых регистров получает байт из памяти дисплея, после чего выполняется его побитовая пересылка в Атрибут-контролер. Для VGA Атрибут-контролер передает 8-битовое значение в ЦАП, устанавливающий соответствующее напряжение для дисплея.

Регистры VGA доступны для чтения/записи, за исключением регистра Атрибута адреса, регистра Состояния входа и регистров-защелок. Во все биты, отмеченные как "Не используются", должны записываться нули. Фирма IBM рекомендует при использовании адаптеров VGA выполнять чтение из порта только тех разрядов, ко-

торые будут модифицированы, после чего осуществлять запись результата обратно в порт (эта операция гарантирует совместимость с будущим развитием адаптеров).

9.3.3. Технология точечной графики

Байт, записываемый процессором в видеопамять, поступает в графический контроллер, подвергаясь преобразованию с помощью регистра Циклического сдвига данных (индекс "3", порт $3CF_{16}$). Этот регистр выполняет либо сдвиг данных вправо на n позиций ($0 \leq n \leq 7$), либо логические операции ("И","ИЛИ", Исключающее ИЛИ"), либо обе функции, начиная с циклического сдвига. В режиме рисования точки на экране регистр сдвигает байт вправо на 1 разряд, а затем результат складывает по логике "ИЛИ" с содержимым регистров-защелок. Дальнейшее преобразование данных осуществляется в соответствии с содержимым регистра Разрешение Установки/Сброса и регистра Установки/Сброса (первый выбирает функцию для второго):

1) если бит регистра Разрешения Установки/Сброса, управляющий данным цветовым слоем, равен нулю, то байт записывается в видеопамять без изменения;

2) если бит регистра Разрешения Установки/Сброса равен единице, в данный цветовой слой записывается байт, все биты которого устанавливаются в соответствии с содержимым регистра Установки/Сброса для данного цветового слоя.

Дальнейшая запись в видеопамять происходит в соответствии с содержимым регистра Битовой маски:

1) если данный бит регистра Битовой маски содержит единицу, соответствующие биты для каждого из цветовых слоев поступают из процессора;

2) если данный бит регистра Битовой маски содержит ноль, соответствующие биты для каждого из цветовых слоев поступают от регистров-защелок.

При разработке программы для точечной графики пользователь должен знать режимы 3 регистров: регистра Адреса графики 1 и 2 (индексный порт $3CE_{16}$), регистра Режима (индекс"5", порт $3CF_{16}$) и регистра Битовой маски (индекс "8", порт $3CF_{16}$).

9.3.4. Индексный порт $3CE_{16}$. Регистр Адрес графики 1 и 2 (Graphics 1 and 2 Address Register)

Регистр Адрес графики 1 и 2 определяет регистр порта $3CF_{16}$. Номер индекса выбранного регистра выдается в порт $3CE_{16}$ командой OUT. Значения индексов и соответствующих им регистров приведены в табл. 9.1

Таблица 9.1

Индекс	Регистр
0	Установка/Сброс
1	Разрешение Установки/Сброса
2	Сравнение цветов
3	Циклический сдвиг данных
4	Выбор схемы чтения
5	Режим
6	Смешанный
7	Цвет безразличен
8	Битовая маска

В данном методическом пособии используются регистры с индексами 5 (Режим) и 8 (Битовая маска).

9.3.5. Порт $3CF_{16}$. Регистр Режим (индекс "5", Mode Register)

Регистр Режим сначала должен быть выбран путем записи значения 5 в регистр Адрес графики 1 и 2 (порт $3CE_{16}$).

Регистр управляет несколькими различными функциями графического контроллера, – в частности, режимом записи в видеопамять, а также разрешением режима сравнения цветов.

На рис. 9.2 показана структура регистра. Ниже подробно рассмотрены его отдельные биты. Не рекомендуется изменять состояние битов D4 - D7, так как это может привести к потере изображения на экране дисплея.

В адаптере применяются 3 метода записи данных и 2 метода чтения. Оптимальным выбором режима для конкретной задачи можно значительно повысить скорость вывода и чтения пикселей.

Рис. 9.2. Структура регистра "Режим"

Здесь D1 и D0 – биты, определяющие режимы записи в видеопамять. Возможно использование 3 различных режимов (см. табл. 9.2).

Таблица 9.2

Использование 3 режимов записи

D1	D0	Номер режима	Режим записи
0	0	0	Режим непосредственной записи
0	1	1	Использование регистров-защелок
1	0	2	Заполнение N-го цветового слоя битом с номером N из данных процессора
1	1	-	Не используется

Режим 0 служит для непосредственной записи. Процессор имеет доступ к видеопамяти, при этом возможно использование следующих операций: Установки/Сброса, Циклического сдвига и всех логических функций. В этом режиме также возможно использование регистра Битовой маски.

Режим 1 служит для записи с использованием регистров-защелок. При чтении данных из видеопамяти (функция PEEK) происходит запись 8 битов из каждого цветового слоя в регистры-защелки. Затем при выполнении операции записи содержимое регистров-защелок может быть записано обратно в видеопамять, но уже по другому адресу. Режим полезен для быстрого копирования данных из одной области видеопамяти в другую.

Режим 2 служит для заполнения N-го цветового слоя (битовой плоскости) битом с порядковым номером N из байта данных, переданного процессором видеоадаптеру для записи. Отсюда следует, что содержимое четырех старших битов записываемого байта (т.е. битов D4-D7) не имеет значения.

По умолчанию в BIOS используются режимы Чтения/Записи "0". При каждой записи данных непосредственно в память адаптера важно, чтобы сначала были загружены регистры-защелки с текущим содержимым видеопамяти путем чтения.

9.3.6. Порт $3CF_{16}$. Регистр Битовой маски (индекс "8", Bit Mask Register)

Регистр Битовой маски управляет записью данных в видеопамять. Если какой-то бит регистра BMR содержит ноль, соответствующий бит будет записываться в видеопамять из регистра-защелки, в противном случае данный бит поступает от процессора (см. рис. 9.3). Этот регистр используется только в нулевом режиме записи.

Занести данные в регистр-защелку можно, если выполнить операцию чтения из видеопамяти. При этом в каждый регистр-защелку передается один байт из соответствующей битовой плоскости.

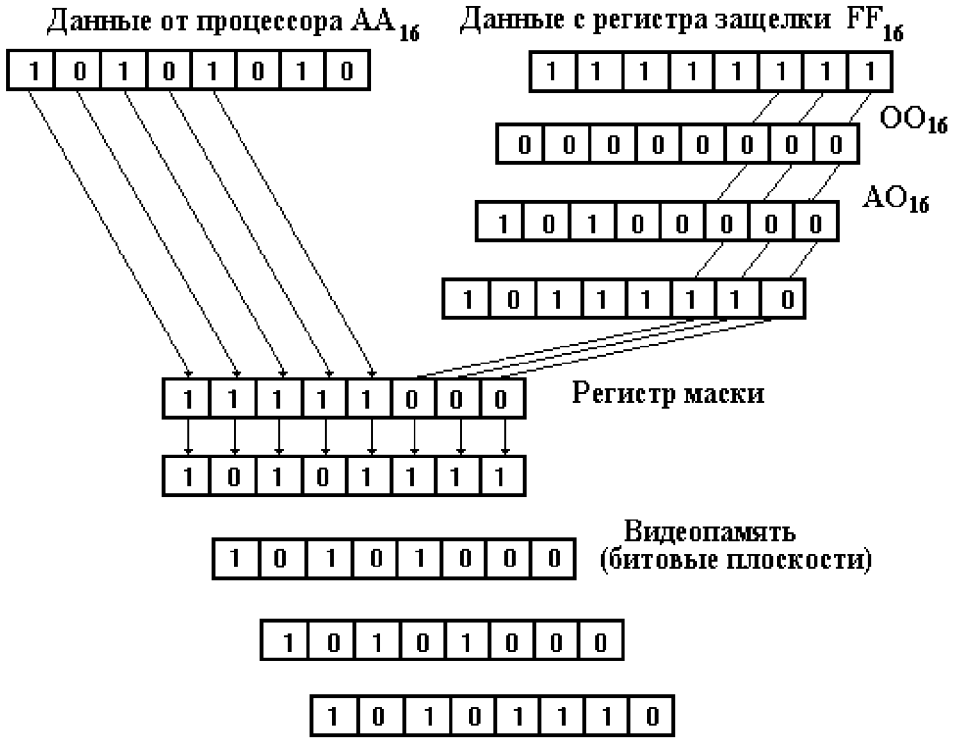


Рис. 9.3. Запись данных в видеопамять с помощью регистра-защелки

Таким образом, установка в любом из битов единицы позволяет центральному процессору изменять его значение. При установке нуля изменения запрещены. Например, установка нуля (0_{16}) защищает любой бит памяти от изменений, установка FF_{16} позволяет изменить состояние всех 8 битов (это же значение FF_{16} хранится в регистре по умолчанию во всех режимах).

9.4. Структура видеопамяти

Структура видеопамяти определяется режимом работы видеоадаптера. Режимы работы видеоадаптеров характеризуются типом информации, которую они отображают (текстовая или графическая), количеством используемых цветов, разрешающей способностью и размерами символов.

При выполнении заданий можно использовать графические режимы с улучшенными характеристиками адаптеров VGA:

1) режим $0D_{16}$ (SCREEN 10) с разрешающей способностью 640x350, который в качестве дисплея использует монохромный дисплей (MD) или дисплей VGA (каждый пиксель может быть черного цвета, белого цвета, интенсивно-белого цвета или отображаться постоянно мигающим);

2) режим $1D_{16}$ (SCREEN 9) с разрешающей способностью 640x350, который отображает 16 цветов (кроме EGA с 64 кб, дающего только 4 цвета) и в качестве дисплеев использует улучшенный цветной (ECD) дисплей VGA, а также некоторые мультичастотные дисплеи;

3) режимы 11_{16} и 12_{16} (SCREEN 11 и SCREEN 12) с разрешением 640x480: в первом реализуется только 2 цвета, во втором – 16 цветов; VGA-дисплеи и мультичастотные;

4) режим 13_{16} (SCREEN 13) с разрешением 320x200, который отображает 256 цветов; использует мультичастотные и VGA-дисплеи.

Напомним, что начальный адрес страницы видеопамяти во всех указанных режимах – $A000_{16}$.

Относительно этого адреса задаются абсолютные адреса (номера) байтов, которые указаны в табл. 9.3 (режим SCREEN 9).

Таблица 9.3

Абсолютные адреса байтов

Номера строк (Nстр)	Номера байтов столбца (NBст)						
0	0	1	2	...	77	78	79
0	80	81	82	...	157	158	159
...	...	...	...	...	...	...	...
349	27920	27921	27922	...	27997	27998	27999

По аналогии с разделом 8 абсолютный номер байта определяется по формуле

$$NB = 80Nстр + NBст. \quad (9.2)$$

Если необходимо работать с пикселем, координаты которого – (a, b), – следует определить номер байта по следующей формуле:

$$NB = ADDR = 80\ b + a\backslash 8, \quad (9.3)$$

где ADDR – номер байта видеопамяти, т.е. его адрес относительно начального адреса страницы видеопамяти;

a – координата пикселя по оси абсцисс OX – номер столбца;

b – координата пикселя по оси ординат OY – номер строки;

a\8 – целочисленное деление a на 8 (число битов в байте).

$$NP = OST = a \bmod 8. \quad (9.4)$$

Однако, как уже отмечалось выше, байты пикселей на экране повернуты на 180°, а в 7-м разряде находится самый младший номер пикселя. Чтобы не вращать номера пикселей вручную на бумаге, можно воспользоваться формулой

$$BITMASK = 2^{(7 - NP)}. \quad (9.5)$$

Из формулы видно, что при номере пикселя NP = 0 BITMASK = 2^7, что соответствует единице в 7-м разряде.

Таким образом, зная адрес байта видеопамяти и задав битовую маску по формуле (9.5), можно приступать к реализации точечной графики.

9.5. Алгоритм реализации точечной графики

Реализация точечной графики осуществляется в следующей последовательности:

1. Устанавливается желаемый режим работы видеоадаптера (в БЕЙСИКе это реализуется оператором SCREEN NR, где NR – номер режима).

2. Определяется начальный адрес страницы видеопамяти при использовании функции

DEF SEG = &HA000.

3. Записывается Индекс &H5 в индексный порт &H3CE с помощью OUT.

4. Через регистр Режим устанавливается режим записи 2, то есть в порт &H3CF помещается число &H2.

5. Выбираются требуемые координаты пикселя экрана (a, b); вычисляется абсолютный номер байта по формуле (9.3).

6. Определяется битовая маска по формуле (9.5).

7. Следует вновь обратиться к индексному порту &H3CE. Устанавливается (записывается) режим Битовой маски (индекс &H8).

8. Битовая маска передается по п.6 через порт &H3CF.

9. Перед тем как записать данные по адресу байта видеопамяти, необходимо заполнить регистры-защелки, т.е. произвести операцию чтения (PEEK) по адресу байта видеопамяти (см. п.5).

10. Чтобы высветить заданный пиксель с координатами (a, b), достаточно лишь поместить (записать) желаемый номер цвета этого пикселя (0?15 для SCREEN9) по адресу, вычисленному в п.5 (в начале программы он может быть задан в виде переменной COL%).

11. При необходимости продолжить окрашивание пикселей можно по какому-либо закону получить координаты следующего пикселя и выполнить операции, начиная с п.5.

12. По окончании работы необходимо провести восстановление значений до состояния, ожидаемого BIOS по умолчанию (проводится обязательно). Для этого необходимо:

1) установить режим Битовой маски, т. е. индекс &H8 в порт &H3CE;

2) разрешить к изменению любой бит байта, т. е. поместить число &HFF в порт &H3CF;

3) установить регистр-режим, т. е. индекс &H5, в порт &H3CE;

4) установить режим записи ноль, т. е. поместить число &H0 в порт &H3CF.

9.6. Установка цвета

Регистры Атрибут-контроллера управляют распределением цветов по соответствующим номерам цветов (N = 0,1...15), сканированием рамки и цветом фона. Для этого используются 7 регистров: регистр Атрибута (порт $3C0_{16}$); регистр Палитры (порт $3C0_{16}$, индекс $0F_{16}$); регистр Управления режимом (порт $3C0_{16}$, индекс 10_{16}); регистр Цвета рамки (порт $3C0_{16}$, индекс 11_{16}); регистр Разрешения матрицы цветов (порт $3C0_{16}$, индекс 12_{16}); регистр Горизонтального поэлементного панорамирования (индекс 13_{16}); регистр Выбора цвета (индекс 14_{16}).

В данной работе предполагается использование 3 регистров.

9.6.1. Порт $3C0_{16}$. Регистр Адреса атрибута

Регистр Адреса атрибута определяет, какой из регистров должен появляться в порте $3C0_{16}$. Номер индекса желаемого регистра записывается в порт $3C0_{16}$. В связи с тем, что этот порт разделяется регистром адреса и индексированными регистрами, регистр Адреса всегда должен быть инициализирован. Чтение порта (с помощью функции INP) с адресом $3DA_{16}$ всегда устанавливает регистр Адреса атрибута в функцию Адрес (например, RGA = INP(&H3DA)).

По адресу $3DA_{16}$ для VGA находится регистр Управления признаком, биты которого обеспечивают передачу сигналов соединителю. Таким образом, устройство, подключенное к соединителю, может программно контролироваться.

9.6.2. Порт $3C0_{16}$. Регистры Палитры (индекс $0F_{16}$)

В EGA 16 регистров управляют фактическим цветом, выводимым на экран по соответствующему ему номеру (комбинации битовых матриц). Индексы с 0 по 15 управляют соответственно цветами, пронумерованными с 0 по 15. Схема определения цвета по умолчанию заключается в том, что сумма номеров двух цветов дает в результате цвет, представляющий из себя смесь этих двух цветов. Например, цвет 1 (синий) + цвет 2 (зеленый) дают цвет 3 (голубой). Основные цвета – синий, зеленый и красный – образуют нормализованную двоичную последовательность 1,2,4. Цвета с 8 по 15 являются более яркими оттенками цветов 0 – 7.

Понятие основных (RGB) и вторичных (rgb) цветов относится только к мониторам цифрового типа. В аналоговых мониторах VGA применяются цифро-аналоговые преобразователи (ЦАП) для преобразования установок палитры в соответствующие цвета. В действительности ЦАП действует так же, как палитра, из которой выбираются фактические цвета. Регистры палитры служат только для определения индекса для встроенной в ЦАП таблицы цветов. Окончательный номер может быть определен через регистр Выбора цвета (индекс 14_{16}).

В VGA в режиме 13_{16} установка регистров не оказывает никакого действия, а регистры Атрибута могут быть считаны из порта $3C1_{16}$.

Для установки номера цвета используются только 6 младших битов регистра (0 – 5). Выбор цвета осуществляется в одном из регистров ЦАП (в ЦАП используются 256 18-битовых регистров цвета, обеспечивающих выбор из 262144 цветов). Значение может изменяться с помощью регистра Управления режимом (индекс 10_{16}) и регистра Выбора цвета (индекс 14_{16}). С помощью 6 битов можно осуществить выбор 64 цветов в палитре (от 0 по 63).

9.6.3. Порт $3C0_{16}$. Регистр Выбора цвета (индекс 14_{16})

Регистр Выбора цвета используется только в VGA для увеличения гибкости выбора цветов. Он добавляет 2 бита (2 и 3) для получения полного 8-битового значения количества цветов в палитре (256 цветов).

Регистр Выбора цвета воздействует на все регистры палитры. Он работает так же, как схема сегмент/смещение ЦП, то есть имеет выбор из 4 наборов (2 бита) по 64 цвета (6 битов). Другие биты регистра здесь не рассматриваются.

9.7. Алгоритм установки цвета

Следует сразу отметить, что в данной работе будет рассматриваться простой алгоритм с использованием режима, общего для EGA и VGA, без обращения к регистру Выбора цвета. Этот алгоритм позволит организовать просмотр цветов.

Установка цвета производится в следующей последовательности:

1. Задается цикл (FOR-NEXT) от 0 до 63 по управляющей переменной N (устанавливаемому значению в формате RGBrgb).
2. Инициализируется порт $3C0_{16}$ путем чтения порта $3DA_{16}$ (с помощью PEEK).
3. Исходный номер цвета (например, COL% = 15 (ярко-белый)) записывается в регистры Палитры по адресу &H3C0.
4. Повторяется п.2 с записью значения управляющей переменной N (см. п.1).
5. Повторяется п.2.
6. В порт по адресу $3C0_{16}$ записывается код 20_{16}.
7. Организуется задержка, например:

```
FOR T = 0 TO 10000 : NEXT T.
```

Задания

1. Разработать алгоритм и программу для точечной графики. Задавшись самостоятельно исходной координатой (a, b), нарисовать с помощью точек следующие геометрические фигуры.

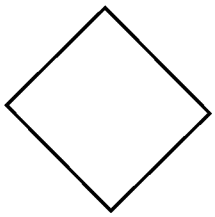
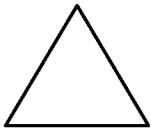
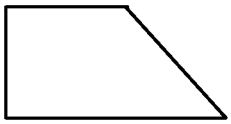
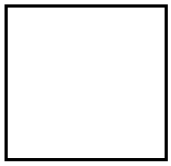
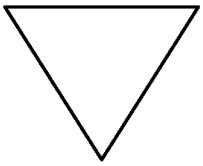
N варианта	Фигура
1	
2	
3	
4	
5	
6	
7	

Рис. 9.4. Геометрические фигуры для рисования

2. Разработать алгоритм и подпрограмму для закрашивания геометрических фигур по п.1 перебором цветов в диапазоне: N = 0?63.

Литература

1. Джордейн Р. Справочник программиста персональных компьютеров типа IBM PC, XT и AT; Пер. с англ. – М.: Финансы и статистика, 1992. – 544 с.

2. Фролов А.В., Фролов Т.В. Программирование видеоадаптеров CGA, EGA, VGA. – М.: ДИАЛОГ-МИФИ, 1992. – 288 с.

3. Графические адаптеры EGA и VGA: Руководство по программированию. – М: НТК "Программпродукт", 1992. – 273 с.

4. Язык GWBASIC. Программирование на персональной ЭВМ: Справ. пособие / Ю.Ф.Вашкевич, Д.А.Безмен, В.И.Костеневич и др.; Под общ. ред. В.И.Костеневича. – Мн.: Выш. школа, 1992. – 267 с.

5. Кетков Ю.Л. GW-, Turbo и Quick – BASIC для IBM PC. – М.: Финансы и статистика, 1992. – 240 с.

Содержание

Учебное издание

МОСКАЛЕНКО Алексей Анисимович
КОНОНЕНКО Зоя Ивановна
ДЕРБАН Андрей Николаевич
ПОЗНИК Юлия Николаевна

УПРАВЛЕНИЕ ВНЕШНИМИ УСТРОЙСТВАМИ МИКРОЭВМ

Методическое пособие
по дисциплинам
"Проектирование микропроцессорных систем управления"
и "Системы управления технологическим оборудованием"
для студентов специальностей
1-53 01 01 "Автоматизация технологических процессов
и производств (в приборостроении и электронике)",
1-53 01 02 "Автоматизированные системы обработки информации"
и 1-53 01 06 "Промышленные роботы и робототехнические комплексы"

В 2 частях

Ч а с т ь 1

Редактор Т.А.Палилова. Корректор М.П.Антонова
Компьютерная верстка Н.А.Школьниковой
Подписано в печать 24.09.2004.
Формат 60x84 1/16. Бумага типографская № 2.
Печать офсетная. Гарнитура Таймс.
Усл. печ. л. 6,3. Уч.-изд. л. 4,9. Тираж 100. Заказ 8.
Издатель и полиграфическое исполнение:
Белорусский национальный технический университет.
Лицензия № 02330/0056957 от 01.04.2004.
220013, Минск, проспект Ф.Скорины, 65.