Государственное бюджетное образовательное учреждение

среднего профессионального образования города Москвы

**Колледж связи №54 имени**

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

ПО ПРОВЕДЕНИЮ ЛАБОРАТОРНЫХ РАБОТ

для специальности 210414

**Техническое обслуживание и ремонт радиоэлектронной техники**

**по МДК.03.03** .Микроконтроллеры в управлении радиоэлектронной аппаратурой

**ПМ.03.** Проведение диагностики и ремонта различных видов радиоэлектронной техники

для специальности 210414**Техническое обслуживание и ремонт радиоэлектронной техники**( по отраслям)

 ( углубленная подготовка)

Москва, 2014

Рассмотрено: Утверждено:

на заседании цикловой комиссии

 Зам. директора по УР

Протокол № \_\_\_\_ от «\_\_\_\_» \_\_\_\_\_\_ 2014г. \_\_\_\_\_\_\_\_\_\_\_ И.Г. Бозрова

Председатель ПМЦК \_\_\_\_\_\_\_\_\_\_ В.Н. Кириленко«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2014г

Автор(ы) Ручко В.М. преподаватель ГБОУ СПО Колледж связи №54

 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Предисловие**
	1. **Назначение методических указаний**

 Методические указания составлены на основе рабочей программы МДК .03.03 «Микроконтроллеры в управлении радиоэлектронной аппаратурой**»**

ПМ.03 «Проведение диагностики и ремонта различных видов радиоэлектронной техники**»**и «Требований к разработке методических указаний для студентов по проведению лабораторных работ и практических занятий» и предназначены для обучающихся специальности 210414 «Техническое обслуживание и ремонт радиоэлектронной техники».

Выбор содержания и объем конкретной лабораторной работы обусловлен сложностью учебного материала для усвоения, междисциплинарными связями и учетом значения конкретной лабораторной работы для приобретения обучающимися соответствующих умений и компетенций, предусмотренных ФГОС.

Перечень лабораторных работ определен рабочей программой дисциплины.

* 1. **Требования к умениям и профессиональным компетенциям**

 В результате выполнения всех лабораторных работ обучающийся должен

*иметь*:

-основные сведения об элементах микропроцессорного управления: классификацию, характеристики, принцип действия;

- способы программирования микроконтроллеров;

-логические основы построения микроконтроллеров,

-типовые узлы и устройства микропроцессорной техники: регистры, дешифраторы, счетчики, сумматоры; принципы построения, и классификацию устройств памяти;

-способы организации интерфейсов в микроконтроллерных системах ; периферийные устройства радиотелевизионной техники;

-принципы взаимодействия аппаратного и программного обеспечения в работе микроконтроллеров; основы алгоритмизации и программирования на языке Ассемблер.

*Уметь:*

- проводить анализ исправности микроконтроллеров по состоянию на выходе и на его входах;

-выбирать тип микросхемы по справочнику, исходя из заданных параметров и условий использования;

-составлять программы для организации взаимодействия с памятью и с внешними устройствами; читать электрические схемы, построенные на микросхемах микроконтроллеров

Лабораторные работы проводятся по методическим указаниямразработанных в соответствии с программой ПМ 03 , МДК 03.03 в лаборатории оборудованной компьютерными рабочими местами (для каждого студента определенное рабочее место).

Лабораторные работы, и защита лабораторных работ, происходят на тренажерах ДЕГЕМ. Форма итогового контроля - зачет

2. Правила выполнения лабораторных работ

* 1. . Обучающийся должен подготовиться к выполнению лабораторной работы, для чего ему необходимо повторить соответствующий теоретический материал и подготовить ответы на теоретические вопросы к лабораторной работе;
	2. . Перед началом каждой работы проверяется готовность обучающегося к лабораторной работе путем устного опроса, проведения тестов и т. д.;
	3. . После выполнения лабораторной работе обучающийся должен оформить и представить отчет о проделанной работе по форме, представленной преподавателем,с представлением полученных результатов проведенных экспериментов и выполненных измерений, с выводами о проделанной работе;
	4. . Обучающийся, пропустивший выполнение лабораторной работы по уважительной или неуважительной причинам, обязан выполнить работу в дополнительно назначенное время.
	5. . Оценка за работу обучающемуся выставляется с учетом предварительной подготовки к работе, доли самостоятельности при ее выполнении, точности и грамотности оформления отчета по работе, аргументированной защите.
* Оценка «5» ставится, если лабораторная работа выполнена в полном объеме, в соответствии с заданием, с соблюдением последовательности выполнения, замеры и расчеты выполнены без ошибок, самостоятельно; работа оформлена аккуратно.
* Оценка «4» ставится, если лабораторная работа выполнена в полном объеме, в соответствии с заданием, с соблюдением последовательности выполнения, частично с помощью преподавателя, присутствуют незначительные ошибки при замерах и расчетах; работа оформлена аккуратно.
* Оценка «3» ставится, если лабораторная работа выполнена в полном объеме, в соответствии с заданием, частично с помощью преподавателя, присутствуют ошибки при замерах и расчетах; по оформлению работы имеются замечания.
* Оценка «2» ставится, если обучающийся не подготовился к лабораторной работе, при замерах и расчетах допустил грубые ошибки, по оформлению работы имеются множественные замечания.

**Лабораторная работа №1.**

**(**время выполнения 4 часа**)**

**«Исследование главных составляющих микрокомпьютера ЕВ-153 и их функции»**

Цель работы:

1.Изучить главные составляющие микрокомпьютера ЕВ-153 и описать их функции.

1. Научиться определять сигналы ввода-вывода и описывать их функции.
2. Использовать управляющие функции для анализа работы микрокомпьютера.

 **Подготовка к работе**:

Плата EB-153 является микрокомпьютером со всем набором необходимых функций (см. рисунок).



Это устройство включает в себя следующее элементы:

***Микропроцессор:*** 8-разрядный микропроцессор 8031, который служит в качестве центрального устройства обработки данных.

***Демультиплексор****:*Демультиплексор данных/адреса используется для разделения шины адреса и шины данных. Шина демультиплексора состоит из 8-битной шины данных и 16-битной шины адреса. Разделение шин реализуется с помощью стробирующего сигнала ALEAddressLatchEnable (включение регистра адреса), который поступает на 8-битный регистр-защёлку.

***ПЗУ:***Постоянное Запоминающее Устройство, которое содержит управляющую программу для проведения экспериментов.

***ОЗУ:***Оперативное запоминающее устройство используется микропроцессором для хранения временной информации и пользовательских программ.

***ППИ:***Программируемый периферийный интерфейс - 8255 - служит для передачи данных от клавиатуры к микропроцессору и обеспечивает дополнительную 8-битную шину ввода-вывода.

***ЖК (LCD)дисплей :***ЖК модуль содержит БИС-контроллер (LSIcontroller), который используется для буквенно-цифрового отображения. Благодаря генератору символов и ОЗУ контроллера, в котором хранится информация о визуализации данных, 96 различных ASCII и 32 специальных символа могут быть выведено на дисплей. Дисплей обеспечивает индикацию двух 16-ти символьных строк.

***Схема RS-232:*** Используется для подключения ЕВ-153 к другим устройствам ( IBMPC/XT/AT-совместимым персональным компьютерам, принтерам и т.д) через стандартный последовательный интерфейс.

***Интерфейсная шина:*** Дополнительный параллельный порт ЕВ-153. Данный порт содержит восемь линий данных, 4 адресные линии, одну линию для выбора микросхемы и заземление. Данные линии позволяют подключить к ЕВ-153 различные устройства: реле, датчики, системы управления чем-либо и т.д.

EB-153 имеет две кнопки:

***RESET:*** перезагружает микропроцессор;

***INTERRUPT:*** производит прерывание микропроцессора.

**Оборудование**

Для проведения этого эксперимента вам необходимо следующее оборудование:

* Лабораторный стенд PU-2000
* Печатная плата EB-153
* Комплект соединительных кабелей DL-20

**Порядок работы**

**Установка**

Включите электропитание лабораторного стенда.



Определите местоположение интегральной схемы микропроцессора 8031 и кварцевого кристалла в металлическом корпусе.

Центральный Процессор (ЦП) ЕВ-153 производит все операции с данными, арифметические операции, синхронизацию и управление внешними компонентами.

8031 - это ЦП для целей управления со встроенной программной памятью. Он может управлять 64 Кбайтами внешней програмной памяти в дополнение к 64 Кбайтам внешней памяти данных.

8051 - это 8031 с менее, чем 4 Кбайтамипрограмной памяти, которые реализована как кристалл ПЗУ с масочным программированием, тогда как 8751 имеет 4 Кбайтное ПЗУ с возможностью УФ или электрического стирания.

Общий термин "8051" используется как общее обозначения для серий 8031, 8051, и 8751.

Определите местоположение компонентов памяти (см. кнопку Zoom).

***Четыре ИС используются в качестве компонентов памяти:***

1. 8031 содержит 128 байт внутреннего ОЗУ данных и 128 байт регистров специальных функций.
2. 27256 EPROM (32 Кбайт) содержит управляющую программу. Данный объем памяти используется как программная память.
3. 6264 ОЗУ (8 Кбайт) - эта область памяти используется для програм пользователя или в качестве памяти внешних данных. Микропроцессор выполняет программы пользователя сохраненные в этой области памяти.
4. Посадочное гнездо ИС может быть использовано для подключения 28-ми контактной ИС памяти, что обеспечивает увеличение програмной памяти от 8 до 32 Кбайт. В качестве такой ИС может быть использована ИС ПЗУ предварительного программирования с возможностью УФ EPROM или электрического EEPROM стирания. Эта память используется для хранения программ специального назначения.

Определим местоположение 3-х шин, соединяющих ЦП с другими схемами.

***ШИНА ДАННЫХ*** - это двунаправленное 8-контактноесоединение используемое для передачи команд или данных между ЦП и другими схемами.Направление потока данных задается ЦП с помощью сигналов RD, WR, и PSEN которые передаются по шинам управления и адреса.

***ШИНА АДРЕСА*** - это 16-контактное соединение ЦП с другими схемами.

При использовании 40-контактного исполнения ИС 8051, существует проблема нехватки линий шины адреса для передачи 16-ти битного значения.

Эта проблема решается с помощью использования некоторых линий шины данных для передачи младшего разряда адреса. В первой половине каждого цикла данные линии содержат значения младшего разряда адреса. При этом сигнал ALE (включение регистра адреса) принимает уровень высокого потенциала, что является индикацией передачи адреса через шину данных.

В ЕВ-153, значение адреса хранится в регистре 74LS373. Во второй половине цикла, шина данных используется только для передачи данных.

***ШИНА УПРАВЛЕНИЯ*** служит для передачи всех ненаправленных и нефункциональных сигналов управления системой микрокомпьютера.

 Рассмотрим устройство и принцип работы ЖК-модуля.

***ЖК-модуль*** состоит из ЖК-панели, ЖК-драйвера и контроллера. Все три устройства смонтированы на одной печатной плате. ЖК-панель представляет из себя матрицу, которая может отображать две 16-значные строки. Каждый знак индицируется на участке 5\*7 точек.

Благодоря взаимодействию между генератором символов, построеном на ПЗУ, и ОЗУ, в котором хранится информация о визуализации данных, ЖК-дисплей может отображать данные полученные от микроконтроллера.

Определите местонахождение ***ППИ*** 8255 (программируемый переферийный интерфейс). Это устройство служит для передачи данных от клавиатуры к микропроцессору.

Нажмите кнопку RESET.

Нажмите клавишу "Н" для отображения меню помощи.

Используйте клавишу "стрелка вниз", чтобы просмотреть всю информацию, содержащуюся в меню помощи.

Используйте клавишу "стрелка вверх" для перехода в предыдущий дисплей.

Нажмите клавишу CLEARENTRY (очистить ввод), чтобы закрыть меню помощи.

Используйте ***КОМАНДЫ ДОСТУА К ПАМЯТИ*** для проверки и изменения памяти микропроцессора и внешней памяти.

Начните с ***памяти кода***. Память кода является внешней для микропроцессора 8031.

Обращение к этой области памяти выполняется в ходе работы программы по адресу, хранящемуся в програмном счетчике, и генерацией микропроцессором сигнала PSEN (сигнал разрешения выбора программы).

Это команда позволяет вам загружать, проверять и изменять программы записанные в шестнадцатеричном формате в памяти кода.

Напечатайте следующее:

 CBYTE 0000H

Помните, что адрес - это 4-значное шестнадцатеричное число, за которым следует Н.Адрес 0000Н - это первый адрес управляющей программы.

После ввода нажмите клавишу ENTER .

Используйте стрелки для изменения адреса.Данные не будут изменяться, пока вы не нажмете клавишу ENTER.Изменения адресов стрелками не меняет их содержимого.

Установите адрес в E000H и запишите данные хранящиеся по этому адресу.Это первый адрес кода памяти пользователя.Так как это ОЗУ, данные могут быть изменены.

Введите значение данных и нажмите клавишу ENTER.

Вернитесь к первому адресу и посмотрите, каким образом данные были записаны в память.

Попытайтесь изменить данные, принадлежащие местоположению ПЗУ (адрес с 000H до 7FFFH).Каков результат этой операции?

Нажмите клавишу CLEARENTRY (очистка ввода), чтобы отменить эту операцию.

***Использование команды внешней памяти.***

Для доступа к внешней памяти данных используется команда МОVХ.В 8031 эту операцию выполняют сигналы RD (чтения) и WR (записи).Команда XBYTE использует RD и WR для доступа к внешней памяти данных.

Напечатайте следующее:

XBYTEE000H

Нажмите клавишу ENTER и измените адрес данных.

Эта команда работает так же, как и команда СВYTE.Внешняя память и память для хранения программ перекрываются в 8 Кбайт ОЗУ.Следовательно, значения данных такие же, как если бы они считывались и RD-сигналом и PSEN-сигналом.

Используйте команды внутренней памяти. Внутренняя память - это ОЗУ данных сформированное внутри ИС процессора.Данная область памяти занимает адреса 00Н - 7FH.

Существуют 4 блока универсальных регистров (адреса с 00Н по 1FH)

***Стек*** - это область памяти внутреннего ОЗУ данных.

В области памяти 20Н - 2FH находятся байты, имеющие возможность побитовой адресации.

Обратите внимание на то, что значение байта должно быть адресом в ОЗУ данных. Причем это значение должно оставлять участок памяти для стека таким образом, чтобы указатель стека никогда не превысил значения 7FH.

Для отображения содержимого адреса 70Н во встроенной памяти данных, введите следующее:

 CBYTE 70H

Используйте стрелки в соответствии с указаниями данными ранее, при описании команды CBYTE.

***Использование команды памяти регистра.***

В 8031 регистры специального назначения занимают диапазон памяти от 80Н до FFH. Однако регистры не используют эту область памяти целиком.

Для отображения содержания регистра 90Н, введите следующее:

 RBYTE 90H

Заметьте, что значение байта должно быть адресом в регистре ОЗУ

Нажмите клавишу ENTER.На дисплее высветится состояние Порта 1, соответствующее адресу 90Н.

Измените данные порта и посмотрите как изменяется состояние светодиодов, подключенных к этому порту.

Обратите внимание, что только 4 контакта (бита) порта подключены к светодиодам. Остальные 4 бита подключены к переключателям.

Используйте стрелки, чтобы изменить адрес и вернуться к нему, меняя позиции переключателей.

 ***Использование команды битовой памяти.***

Часть встроенной памяти данных и регистры специального назначения имеют возможность побитовой адресации.

Битовые адреса с 00Н по 7FH - это внутренняя память.

Битовые адреса с 80Н по FFH относятся к отдельным битам регистров.

Для отображения содержания Бита 0, Порта 1, введите следующее:

 RBIT 90H

Нажмите клавишу ENTER.На дисплее будет отображено состояние Бита 0, Порта 1.

Измените установку переключателя подключенного к этому порту и используйте стрелки для отображения данных, каждого бита.

Измените данные Бита 0, Порта 1. Как изменилось состояние светодиода, управляемого данным битом?

Выключите питание лабораторного стенда. Отключите печатную плату ЕВ-153 от PU-2000.

**Вопросы**

1. В чем различие между встроенным (внутренним) и внешним ОЗУ?
* Внутреннее ОЗУ находится в микропроцессоре, а внешнее в ИС ОЗУ на плате.
* И внутреннее, и внешнее ОЗУ сформированы внутри ИС установленных на плате.
* Внутреннее ОЗУ установлено на плате, а внешнее может быть вставлено в контактное гнездо, если необходимо.
1. Сколько контактов имеет шина данных?
* Восемь.
* Шестнадцать.
* Двадцать четыре.
1. Сколько адресов может быть выбрано шиной адреса?
* 16 Кбайт
* 32 Kбайта
* 64 Kбайта
1. Как и когда микропроцессор обращается к памяти кода?
* Во время выполнения программы после перезагрузки системы по адресу в программном счетчике.
* Перед выполнением программы по адресу в программном счетчике и по PSEN сигналу от микропроцессора.
* Во время выполнения программы по адресу в программном счетчике и по PSEN сигналу от микропроцессора.
1. Какой сигнал записывает данные в память кода?
* RD сигнал
* WR сигнал
* PSEN сигнал
1. Как работает стек?
* Данные, записанные последними, могут быть прочитаны первыми.
* Данные, записанные первыми, могут быть прочитаны первыми.
* Данные могут читаться в произвольном порядке.

**Лабораторная работа №2.**

**(**время выполнения 4 часа**)**

**«Исследование управляющих подпрограмм»**

Цель работы:

1.Научиться запускать простые программы на ЕВ-153.

2.Использовать управляющие программы.

3.Выполнять программы в пошаговом режиме и в режиме реального времени.

4.Менять формат записи программы с формата машинного языка на формат Ассемблера и наоборот.

 Подготовка к работе:

Управляющая программа содержит указатель адреса команды, которая должна быть выполнена в ходе следующего рабочего цикла.

Команда ASM начинает свою работу с адреса, указанного после ключевого слова АSM. Если адрес не указан, используется значение указателя по умолчанию.Это значение равно первому доступному коду ОЗУ (Е000Н).

Управляющая программа отображает адрес указателя.

Для редактирования мнемонической команды 8051, пользователь должен нажать клавишу ENTER.

Стрелки вправо и влево могут быть использованы для правки текста.

Если пользователь введет неправильную команду, на дисплее высветится сообщение о синтаксической ошибке.

Синтаксическая ошибка не изменяет значение указателя.

**Оборудование**

* Лабораторный стенд PU-2000
* Печатная плата EB-153
* Комплект соединительных кабелей DL-20

Поместите печатную плату ЕВ-153 на лабораторный стенд.Включите питание.

**Порядок работы**

Чтобы выполнить команду ASM, наберите следующее: ASME000H и нажмите клавишу ENTER.

Наберите на клавиатуре команды указанные ниже и нажмите клавишу "CLEAR" после последней команды.

 CLR A

 INC A

 LJMP E001H

Это простая программа увеличивает содержимое аккумулятора.

**Теория**

Дезассемблирующая программа позволяет пользователю перевести значения памяти в мнемонические команды 8051.

Управляющая программа рассматривает начальный адрес команды как первый байт команды. При этом второй и третий байты, если необходимо, интерпритируются как операнды.

Кнопка "стрелка вниз" может быть использована для увеличения указателя адреса на 1 и вывода других дезассемблированных команд на дисплей.

**Эксперимент - 1**

Наберите DASME000H и нажмите клавишу ЕNTER.

С помощью клавиши "стрелка вниз", выведите на экран программу сохраненную в памяти. Сравните ее с командами Ассемблера, которые Вы ввели.

Нажмите клавишу "CLEAR", чтобы выйти из данного режима.

С помощью команды CBYTE, убедитесь, что команды были записаны в память правильно, в шестнадцатеричном формате.

**Теория - 2**

КОМАНДЫ режима ИСПОЛНЕНИЯ ПРОГРАММЫ используются для управления выполнением программ, которые уже записаны.

Команда GO запускает выполнение программы в режиме реального времени.Чтобы запустить программу, наберите следующую команду и нажмите клавишу ENTER.

 GOE000H

Во время выполнения программы на дисплее высвечивается сообщение: RUNNINGUSER'SPROGRAM (идет выполнение программы пользователя).

Программа будет выполняться до тех пор пока вы не нажмете клавишу INTR.Тогда выполнение программы остановится и значение счетчика команд и соответствующая ему команда отобразиться на дисплее.

Нажимая клавишу "стрелка вниз", Вы можете вывести на дисплей следующие регистры и их значения:

SP (указатель стека) A (аккумулятор)

DPTR (указатель данных) РSW (слово состояния программы)

Нажмите клавишу "стрелка вниз" снова, чтобы вернуться в управляющую программу.

**Эксперимент - 2**

При прерывании программы, используя специальные команды отображения данных (команды дампинга), можно вывести на дисплей содержимое регистров и внутренней памяти.

Наберите RDUMPEOH и нажмите клавишу ENTER, чтобы вывести на экран содержимое аккумулятора.

Нажмите на клавишу "CLEAR", чтобы отменить эту команду.

**Теория - 3**

В пошаговом режиме, выполнение программы останавливается после выполнения каждой команды.

После остановки, управляющая программа отображает следующую команду и ее адрес (значение счетчика команд), содержимое аккумулятора, указатель данных, указатель стека и слово состояния программы.

Вы можете приостановить автоматическое пошаговое выполнение программы, нажав любую клавишу. В таком режиме отображаемые на экране данные не меняются до тех пор пока Вы не отпустите нажатую клавишу.После того как вы отпустите клавишу, выполнение программы продолжиться.

**Эксперимент - 3**

Наберите на клавиатуре STEPE000H и нажмите клавишу ENTER, чтобы запустить пошаговое выполнение программы.

Наблюдайте за работой программы и приращением аккумулятора.

Попробуйте нажимать клавиши для того, чтобы приостанавливать выполнение программы на различных участках.

Нажмите клавишу INTR, чтобы прекратить выполнение программы.

Нажмите клавишу "стрелка вниз", чтобы вывести на дисплей содержание регистров.

Снова нажмите клавишу "стрелка вниз", чтобы выйти из режима пошагового выполнения программы.

С помощью команд "дампинга" сравните текущие значения регистров с предыдущими.

**Теория - 4**

Команда контрольной точки устанавливает или отменяет размещение контрольной точки по определенному адресу.

Размещение контрольной точки проверяется при выполнении программы командами "Go" и "Step".

Когда программа обращается к адресу, содержащему контрольную точку, система прерывает работу микрокомпьютера и останавливает обработку данных.

Затем отображаются адрес контрольной точки и соответствующая ему команда.

Нажимая клавишу "стрелка вниз", Вы можете вывести на дисплей следующие регистры и их значения:

SP (указатель стека) A (аккумулятор)

DPTR (указатель данных) PSW (слово состояния программы)

Обратите внимание, что все эти значения соответствуют состоянию микрокомпьютера ДО выполнения команды, размещенной по адресу контрольной точки.

Адрес контрольной точки должен быть первым адресом команды, а не адресом операнда.

**Эксперимент - 4**

Наберите на клавиатуре BRE002H, чтобы установить контрольную точку по адресу E002H и снова запустите пошаговый режим.

Обратите внимание на остановку выполнения программы по адресу контрольной точки.

Вернитесь к управляющей программе и сбросьте контрольную точку, набрав на клавиатуре BR, а затем нажав ENTER.

Снова запустите пошаговый режим.

**Эксперимент - 5**

Сейчас Вы попробуете использовать несколько встроенных функций управляющей программы.

Следующий набор программ используется для взаимосвязи микропроцессора и дисплея.

Операция "CLEARDISPLEY" (очистить дисплей) (адрес FF80H) удаляет всю информацию с дисплея.

Введите следующую программу:

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000HLCALLFF80H вызов подпрограммы очистки дисплея

E003HLJMPE003H ОСТАНОВКА

Наберите на клавиатуре GOE000H и нажмите клавишу ENTER.

Посмотрите, как выполняется очистка дисплей.

Нажмите кнопку INTR для выхода из программы.

ASCII-данные могут быть переданы на дисплей с помощью заполнения буфера памяти, адреса FF90H - FFAFH, и выполнения процедуры DISPLAY-BUFFER (адрес FF83H).

Используйте команду XBYTE, чтобы загрузить в буфер данные в соответствии с приведенными ниже значениями.

|  |
| --- |
| АДРЕС ДАННЫЕ ASCII символ |
| FF90H 30H 0 |
| FF91H 31H 1 |
| FF92H 32H 2 |
| FF93H 33H 3 |
| FF94H 34H 4 |
| FF95H 35H 5  |
| FF96H 36H 6 |
| FF97H 37H 7 |
| FF98H 38H 8 |
| FF99H 39H 9 |
| FF9AH 41H A |
| FF9BH 42H B |
| FF9CH 43H C |
| FF9DH 44H D |
| FF9EH 45H E |
| FF9FH 46H F |

|  |
| --- |
| АДРЕС ДАННЫЕ ASCII символ |
| FFA0H 61H a |
| FFA1H 62H b |
| FFA2H 63H c |
| FFA3H 64H d |
| FFA4H 65H e |
| FFA5H 66H f |
| FFA6H 67H g |
| FFA7H 68H h |
| FFA8H 69H i |
| FFA9H 6AH j |
| FFAAH 6BH k |
| FFABH 6CH l |
| FFACH 6DH m |
| FFADH 6EH n |
| FFAEH 6FH o |
| FFAFH 70H p |

Введите следующую программу:

 АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H LCALLFF83H вызывает операцию отображения

 данных из буфера

E003H LJMPE003H ПАУЗА

Наберите на клавиатуре GOE000H и нажмите клавишу ENTER.Посмотрите, как отображается информация на дисплее.

Нажмите кнопку INTR, чтобы выйти из программы.

Выполните эту программу, используя различные значения из таблицы ASCII (от 00Н до 7FH).

**Теория - 5**

Для передачи однобайтного ASCII символа в буфер дисплея необходимо указать позицию на дисплее (по адресу FFB0H), данные для отображения (по адресу FFB1H) и выполнить операцию DISPLAY-CHARACTER (адрес FF86H).

Значение позиции на дисплее может лежать в диапазоне от 00Н до 1Fh.

С помощью команды XBYTE запишите данные во внешнюю память.

АДРЕС ДАННЫЕ КОММЕНТАРИЙ

FFB0H 05H адрес символа: 5

FFB1H 58HASCII символ: "X"

**Эксперимент - 6**

Введите следующую программу:

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000HLCALLFF80H очистка дисплея

E003HLCALLFF86H вывод символа на дисплей

E006HLJMPE006H ПАУЗА

Наберите на клавиатуре GOE000H и нажмите клавишу ENTER. Посмотрите как символ Х появляется на шестой позиции дисплея.

Нажмите кнопку INTR, чтобы выйти из программы.

Выполните эту программу для разных ASCII-символов и позиций дисплея.

Выключите питание.Отключите EB-153 от PU-2000.

**Вопросы**

1. Каким образом можно установить аккумулятор в 0?
* Нажатием клавиши "CLEAR" или клавиши "стрелка вниз".
* Нажатием клавиши INTR.
* С помощью команды RDUMPE0H.
1. Сколько ASCII-символов может быть отображено?
* 32
* 64
* 128
1. Какова разница между пошаговым выполнением программы и выполнением в реальном времени?
* В пошаговом режиме программа прерывается после выполнения каждой команды. В режиме реального времени программа прерывается только по адресу контрольной точки или при нажатии клавиши INTR.
* Никакой разницы нет, разница только в применении программы.
* Выполнение в реальном времени не может быть прервано.

**Лабораторная работа №3.**

**(**время выполнения 4 часа**)**

**«Исследование команды передачи данных»**

Цель работы:

1.Научиться понимать разницу между режимами адресации микрокомпьютера.

2.Писать программы, в которых используются команды передачи данных.

3.Передавать и извлекать данные и команды из различных областей памяти.

8051 имеет четыре блока, каждый из которых содержит 8 регистров. Эти блоки находятся по адресам 0 - 31 (00H - 1FH) во внутреннем ОЗУ данных.

Адресация регистра обеспечивает доступ к восьми регистрам (R0-R7) выбранного блока регистров (RB).

Один из четырех блоков выбирается с помощью двухбитного поля в регистре PSW (слово состояния программы), показанного на схеме ниже.

Это два бита RS0 и RS1.



**Оборудование**

* Лабораторный стенд PU-2000
* Печатная плата EB-153
* Комплект соединительных кабелей DL-20

Подключите EB-153 к PU-2000.Включите питание PU-2000.

**Порядок работы**

PSW-регистр расположен по адресу D0H.

Следующая программа демонстрирует как управлять блоками регистров микроконтроллера 8051.

Знак решетки (#) расположенный сразу перед выражением показывает, что это непосредственный (определяемый программой) операнд данных, а не адрес.

Введите следующую программу в память микрокомпьютера.

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000HMOVD0H,#18H выбор блока регистров 3

RS1=1, RS0=1

E003HMOVR0,#55H устанавливает R0=55H

E005HMOVR1,#66H устанавливает R1=66H

E007HMOVR2,#77H устанавливает R2=77H

E009HMOVD0H,#00H восстановление блока регистров 0

 RS1=0, RS0=0

E00CHLJMPE00CH ПАУЗА

Запустите программу с помощью команды GO.

Нажмите кнопку INTR, чтобы остановить программу и покинуть режим выполнения программы.Не используйте команду "STEP", если вы работаете с регистром PSW.

Используйте команду DDUMP, чтобы вывести на экран содержимое регистров R0 и R2, блока 2. Обратите внимание, что числовые значения из нашей программы были сохранены по соответствующим адресам (адрес с 18Н по 1АН).

**Теория - 1**

Прямая адресация является средством доступа только к регистрам специального назначения, имеющим байтовую организацию, и к битам памяти внутри этих регистров.

Прямая адресация байтов может быть так же использована для доступа к младшим 128 байтам внутреннего ОЗУ данных.

**Эксперимент - 1**

Следующая программа показывает несколько примеров прямой адресации к внутреннему ОЗУ данных.

Введите в память микрокомпьютера программу, показанную ниже.

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000HMOV 20H,#11H запись 11H в адрес 20H

E003HMOV 21H,20H запись содержимогоадреса20Н

в адрес 21H

E006HMOV 22H,21H запись содержимогоадреса 21H

в адрес 22H

E009HLJMPE009H Пауза

Выполните программу и нажмите кнопку INTR для остановки программы.

Используйте команду DDUMP, чтобы вывести на экран содержимое внутреннего ОЗУ данных. Обратите внимание, что числовое значение 11H сохранилось в соответствующих ячейках памяти (адрес 20Н и 21H).

Определите различие между командами MOV.

Знак решетки (#) указывает на численное значение, которое передается по определенному адресу.

В нашем случае, число 11H передается по адресу 20H (команда MOV 21H,#11H).

Если знак "решетка" отсутствует, переданные данные - это содержимое ячейки по данному адресу.

В нашей программе, данные, находящиеся по адресу 20H передаются по адресу 21H (команда MOV 21H,20H) и адресу 22H (команда MOV 22H,21H).

**Теория - 2**

Прямая побитовая адресация обеспечивает доступ к 128-битовой области внутреннего ОЗУ данных и 128-битовой области регистров специального назначения.

128-битовая область внутреннего ОЗУ данных находится по адресам 20H - 2FH внутренней памяти данных, которая имеет байтовую организацию.

Например, битовый адрес 00H соответствует адресу байта 20H, бит 0, внутреннего ОЗУ данных.

**Эксперимент - 2**

Следующая программа показывает некоторые примеры прямой адресации к битам внутреннего ОЗУ данных.

Введите программу.

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H CLRC сброс флага переноса

E001H MOV 00H,C запись флага переноса по

адресу бита

E003H CPLC инвертация флага переноса

E004H MOV 01H,C запись флага переноса по

адресу бита

E006H CPLC инвертация флага переноса

E007H MOV 02H,C запись флага переноса по

адресу бита

E009H CPLC инвертация флага переноса

E00AH MOV 03H,C запись флага переноса по

адресу бита

E00CH CPLC инвертация флага переноса

E00DH MOV 04H,C запись флага переноса по

 адресу бита

E00FH CPLC инвертация флага переноса

E010H MOV 05H,C запись флага переноса по

 адресу бита

E012H CPLC инвертацият флага переноса

E013H MOV 06H,C запись флага переноса по

адресу бита

E015H CPLC инвертация флага переноса

E016H MOV 07H,C запись флага переноса по

 адресу бита

E018H LJMP E018H Пауза

Запустите данную программу. Нажмите кнопку INTR для остановки программы.

Используйте команду DDUMP, чтобы вывести на экран содержимое внутреннего ОЗУ данных.

Наберите на клавиатуре DDUMP 20h и нажмите ENTER.Этот адрес должен показать содержимое битовых адресов с 00H по 07H.

Обратите внимание, что программа, бит за битом, записала величину 10101010 (AAH) в битовые адреса с 00H по 07H.Эти битовые адреса соответствуют адресу байта 20H.

Вам необходимо использовать функции "дампинга", для того, чтобы вывести на дисплей содержимое внутреннего ОЗУ данных, так как управляющая программа использует эти ячейки и изменяет их содержимое, когда управление микрокомпьютером возвращается к ней.

**Теория - 3**

Обращение к внутреннему ОЗУ данных может выполняться с помощью косвенной адресации регистров. При такой адресации, при выборе блока регистров, используются значения регистров R0 и R1.

Знак @ ("эт"), стоящий перед символом регистра (R0 или R1), обозначает косвенную адресацию.

**Эксперимент - 3**

Следующая программа показывает некоторые примеры косвенной адресации к внутреннему ОЗУ данных.

Введите следующую программу в память микрокомпьютера:

 АДРЕС КОМАНДА КОММЕНТАРИЙ

E000HMOVR0,#20H установка указателя адреса на 20H

E002HMOV @R0,#55H перемещение 55H в адрес укзанный в R0

E004HINCR0 увеличение указателя адреса

E005HMOV @R0,#66Hперемещение 66H в адрес укзанный в R0

E007HINCR0 увеличение указателя адреса

E008HMOV @R0,#77H перемещение 77H в адрес укзанный в R0

E00AHMOVA,@R0 перемещение указателяданных

в регистр А

E00BHLJMPE00BH остановка

Запустите данную программу. Нажмите кнопку INTR, чтобы прекратить выполнение программы.

Используйте команду DDUMP, чтобы вывести на экран содержимое внутреннего ОЗУ данных.Наберите на клавиатуре DDUMP 20H и нажмите клавишу ENTER.Обратите внимание, что программа записывает значения 55Н, 66H и 77Н по адресам 20Н, 21Н и 22Н соответственно.

Используйте команду RDUMP, чтобы вывести на экран содержимое аккумулятора.Наберите на клавиатуре RDUMPE0H и нажмите кнопку ENTER.

Заметьте, что эта программа записывает последнее значение указателя адреса в аккумулятор.

**Теория - 4**

16-битовый указатель данных (DPTR) может быть использован для доступа к любому адресу полного 64К внешнего адресного пространства.

Команды MOVX используются для доступа к внешней памяти данных.

Непосредственная адресация позволяет обращаться к памяти для хранения программ, как к неизменяемым данным.

Непосредственная адресация выполняется с помощью команды MOVC.

**Эксперимент - 4**

Следующая программа показывает пример доступа к внешней памяти данных и к памяти для хранения программ.

Программа, показанная на следующем экране копирует первые байты управляющей программы в ячейки внешнего ОЗУ.

Введите программу, в память микрокомпьютера.

|  |  |  |
| --- | --- | --- |
| АДРЕС | КОМАНДА | КОММЕНТАРИЙ |
| E000H | MOV DPTR,#0000H | установка указателя адреса на 0000H |
| E003H | CLR A | очисткааккумулятора |
| E004H | MOVC A,@A+DPTR | читает первый байт кода |
| E005H | MOV DPTR,#E100H | установка указателя адреса на E100H |
| E008H | MOVX @DPTR,A | запись байта во внешнее ОЗУ |
| E009H | MOV DPTR,#0001H | установка указателя адреса на 0001H |
| E00CH | CLR A | очистка аккумулятора |
| E00DH | MOVC A,@A+DPTR | читает второй байт кода |
| E00EH | MOV DPTR,#E101H | установка указателя адреса на E101H |
| E011H | MOVX @DPTR,A | запись байта во внешнее ОЗУ |
| E012H | MOV DPTR,#0002H | установка указателя адреса на 0002H |
| E015H | CLR A | очистка аккумулятора |
| E016H | MOVC A,@A+DPTR | читает третий байт кода |
| E017H | MOV DPTR,#E102H  | установка указателя адреса на E102H |
| E01AH | MOVX @DPTR,A | запись байта во внешнее ОЗУ |
| E01BH |  LJMP E01BH | остановка |

Запустите программу. Нажмите кнопку INTR, чтобы остановить выполнение программы.

Используйте команду DASM, чтобы сравнить адреса ПЗУ с 0000H по 0002H и адреса ОЗУ с E100H по E102H соответственно.

Выключите стенд.Отключите ЕВ-153 от лабораторного стенда.

**Вопросы**

1. Как можно прочитать код программы из памяти для хранения программ?
* С помощью команды MOV.
* С помощью команды MOVХ.
* С помощью команды MOVC.
1. С помощью какой команды программа осуществляет доступ к данным во внутренней памяти?
* Используя команды MOV.
* Используя команды RDUMP.
* Используя команды MOVX.
1. С помощью какой команды программа осуществляет доступ к данным из внешней памяти?
* Используя команды MOV.
* Используя команды MOVХ
* Используя команды DASM.
1. Какие сигналы активируются во время передачи кода из памяти для хранения программ?
	* ANSWER
* PSEN
* CLEARENTRY
* INTR
1. Какие сигналы активизируются во время передачи внешних данных?
* ANSWER
* RD и WR
* PSEN
* INTR

**Лабораторная работа №4.**

**(**время выполнения 4 часа**)**

**«Исследование портов ввода-вывода микроконтроллера»**

Цель работы:

1.Научиться выполнять операции с булевыми переменными.

2.Использовать возможности ввода-вывода микрокомпьютера.

3.Понимать принципы работы программ, в ходе которых используются порты микрокомпьютера.

4.Писать программы, которые используют Булевы функции.

**Оборудование**

* Лабораторный стенд PU-2000
* Печатная плата EB-153
* Комплект соединительных кабелей DL-20

Подключите EB-153 к лабораторному стенду PU-2000.Включите лабораторный стенд.

**Теория**

В упражнении приведен пример Булевой обработки данных.

Булевыми переменными являются биты.

Одна из важных характеристик 8051 - это возможность непосредственной битовой адресации.Возможность непосредственной битовой адресации позволяет писать более эффективные программы, поддерживающие режимы двоичного ввода и вывода, что часто необходимо в программах управления.

К порту 8051 подключено четыре светодиода и четыре переключателя:

Контакты, подключенные к светодиодам: бит 7, бит 6, бит 5 и бит 4;

 к переключателям: бит 3, бит 2, бит 1, бит 0.

В следующих программах приведены примеры обработки данных битов порта Р1.

Программа, показанная на следующем экране используется для передачи светодиодам логических состояний переключателей.

**Порядок работы**

Напишите программу

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000HMOVC,93H читает бит 3 и передает биту 7

E002HMOV 97H,C

E004HMOVC,92H читает бит 2 и передает биту 6

E006HMOV 96H,C

E008HMOVC,91H читает бит 1 и передает биту 5

E00AHMOV 95H,C

E00CHMOVC,90H читает бит 0 и передает биту 4

E00EHMOV 94H,C

E010HLJMPE000H переходит к начальному адресу

Нажмите на кнопку RESET, чтобы установить Порт1 в исходное состояние.

Выполните программу пошагово, изменяя при этом положение переключателей.

Нажмите кнопку INTR, чтобы остановить пошаговое выполнение программы.

**Теория - 1**

Программа, показанная на следующем экране, считывает значения положения переключателей, преобразует их в ASCII-значения ("0" и "1") и отображает эти символы на алфавитно-цифровом дисплее.

Операция CLEAR-DISPLAY-BUFFER (очистить буфер дисплея) (адрес FF89H) очищает буфер дисплея (адреса FF90H - FFAFH), записывая ASCII-значение знака пробела (20H) во все ячейки буфера.

Перед выполнением команды JNC происходит увеличение значения программного счетчика, затем к нему добавляется значение относительного смещения (в нашем случае - 02H).

**Эксперимент - 1**

Введите программу в память микрокомпьютера:

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H LCALLFF89H вызывает операцию очистки

 буфера дисплея

E003H MOVDPTR,#FF90H устанавливает указатель данных

 на адрес буфера

E006H MOVC,93H читает бит 3 и передает его в бит

 переноса

E008H MOVA,#30H записывает в буфер памяти

 символ ASCII "0"

E00AH MOVX @DPTR,A

E00BH JNC 02H проверяет бит порта - и переходит

 +4

E00DH MOVA,#31H записывает в буфер памяти

 символ ASCII "1"

E00FH MOVX @DPTR,A

E010H INCDPTR увеличивает на 1 указатель

 буфера дисплея

E011H MOVC,92H читает бит 2 и передает данные в

 бит переноса

E013H MOVA,#30H записывает в буфер памяти

 символ ASCII "0"

E015H MOVX @DPTR,A

E016H JNC 02H проверяет бит порта – переходит

 +4

E018H MOVA,#31H записывает в буфер памяти

 символ ASCII "1"

E01AH MOVX @DPTR,A

E01BH INCDPTR увеличивает указатель буфера

 дисплея

E01CH MOVC,91H читает бит 1 и передает данные в

 бит переноса

E01EH MOVA,#30H записывает в буфер памяти

 символ ASCII "0"

E020H MOVX @DPTR,A

E021H JNC 02H проверяет бит порта – переходит

 +4

(Программа – продолжение.)

АДРЕС КОМАНДА КОММЕНТАРИЙ

E023H MOVA,#31H записывает в буфер памяти

 символ ASCII "1"

E025H MOVX @DPTR,A

E026H INCDPTR увеличивает на 1указатель буфера

 дисплея

E027H MOVC,90H читает бит 0 и передает данные

 в бит переноса

E029H MOVA,#30H записывает в буфер памяти

 символ ASCII "0"

E02BH MOVX @DPTR,A

E02CH JNC 02H проверяет бит порта – переходит

 +4

E02EH MOVA,#31H записывает в буфер памяти

 символ ASCII "1"

E030H MOVX @DPTR,A

E031H LCALLFF83H вызывает операцию обработки

 буфера дисплея

E034H LJMPE003H возвращается к началу цикла

С помощью команды GO запустите эту программу.

Измените положение переключателей.

Нажмите кнопку INTR, чтобы остановить программу.

**Теория - 2**

8255 - это программируемый периферийный интерфейс (ППИ), созданный для использования в микрокомпьютерных системах.

Его функция та же, что и у обычного устройства ввода-вывода - подключение периферийных устройств к системной шине микрокомпьютера.

8255 запрограммирован системной программой таким образом, что, обычно, не требуется никаких дополнительных элементов для подключения периферийных устройств.

8255 имеет 24 контакта ввода-вывода, которые могут быть индивидуально запрограммированы как две независимые группы, по 12 контактов каждая, и использоваться в 3-х основных режимах работы.

В первом режиме (MODE 0), каждая группа контактов ввода-вывода может быть разделена на подгруппы, по 4 контакта в каждой. Все контакты подгруппы работают либо в режиме ввода либо в режиме вывода.

Во втором режиме (MODE1) в группе выделяется подгруппа из 8 контактов для работы в режиме ввода или вывода. 3 контакта из 4 оставшихся используются для сигналов подтверждения установления связи и прерывания.

Третий режим работы (MODE 2) - это режим двунаправленной шины. Подгруппа из 8 контактов формирует двунаправленную шину и 5 контактов, один из которых принадлежит другой группе, используются для обмена сигналами для установления связи.

На этом рисунке дан формат определения режима ППИ.



Адреса ППИ указаны ниже:

АДРЕС ОПИСАНИЕ

0000H Порт A

0001H Порт B

0002H Порт C

0003H Регистр режима

Следующая программа использует Порт C ППИ (8255-программируемый периферийный интерфейс) для вывода значения двоичного счетчика.

**Эксперимент - 2**

Введите программу в память микрокомпьютера:

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H MOVDPTR,#0003H устанавливает указатель

 данных на адрес регистра режима

 ППИ

E003H MOVA,#80H определяет Порт С как

 выход

E005H MOVX @DPTR,A

E006H CLRA устанавливает значение

 счетчика (аккумулятора) в "0"

E007H MOVDPTR,#0002H устанавливает указатель

 данных на Порт C ППИ

E00AH INCA увеличивает счетчик на 1

E00BH MOVX @DPTR,A записывает значение

 счетчика в Порт С

E00CH LJMPE00AH возвращается к началу цикла

Используйте команду GO, чтобы запустить эту программу.

Подключите осциллограф к выходам Порта С. Убедитеь, что на выходе линии Бит 0 частота следования импульсов наибольшая, а на каждом последующем выходе значение частоты в 2 раза меньше чем на предыдущем выходе.

Нажмите кнопку INTR для остановки программы.

**Теория - 3**

Следующая программа считывает логические состояния переключателей Порта 1 8031 и передает эти значения Порту С ППИ.

**Эксперимент - 3**

Введите следующую программу в память компьютера.

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H MOVDPTR,#0003H устанавливает указатель данных на

 адрес регистра режима ППИ

E003H MOVA,#80H определяет порт С как выходной

E005H MOVX @DPTR,A

E006H MOVDPTR,#0002H устанавливает указатель данных на

 Порт C ППИ

E009H MOVA,90H считывает данные Порта 1

E00BH MOVX @DPTR,A передает данные Порту C

E00CH LJMPE009H переходит к началу цикла

Используйте команду GO, чтобы запустить программу.

Подключите выходы Порта С к осциллографу и проследите, что их значения соответствуют состояниям Порта 1.

Нажмите кнопку INTR, чтобы остановить программу.

На следующих экранах в систему будет внесена ошибка и она будет имитировать неисправность.

Теперь запустите предыдущую программу. Подключите осциллограф для наблюдения за состоянием выходов Порта С. Состояние выходов Порта С должно соответствовать логическим состояниям Порта 1.

Теперь в схеме произошли изменения. Повторите действия, описанные на предыдущем экране.

Соответствуют ли состояние выходов Порта С логическим состояниям Порта 1?

Какие биты не соответствуют?

Нажмите кнопку INTR, чтобы остановить программу.

Перейдите к следующему экрану.

**Теория - 4**

Следующая программа сдвигает "1" по разрядам Порта1, которые подключены к светодиодами.

**Эксперимент - 4**

Введите программу в память компьютера.

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H MOVA,#10H устанавливает начальное состояние

E002H CLRC очищает флаг переноса

E003H MOV 90H,A записывает данные в Порт1

E005H RLCA циклически сдвигает данные

 аккумулятора влево с переносом

 старшего бита в бит переноса

E006H JCF8H продолжает сдвиг - переход к E000Н

E008H JNCF9H возвращается к началу цикла - переход

 к E003

Запустите программу с помощью команды STEP.

Проследите за сдвигом разряда, наблюдая за индикацией светодиодов.

Нажмите кнопку INTR, чтобы остановить программу.

**Вопросы**

1. Что такое Булева переменная?
* Булева переменная - это "0" или "1".
* Булева переменная - это шестнадцатеричное число.
* Булева переменная - это десятичное число.
1. Каковы признаки Булева процессора?
* Булев процессор производит числовые и логические операции с битами.
* Булев процессор производит числовые операции с битами.
* Булев процессор производит логические операции с битами.
1. Как вычисляется положительный относительный сдвиг для команды JNC?
* JNC-команда добавляет значение относительного сдвига с помощью программного счетчика.
* Команда JNC увеличивает значение программного счетчика на два после добавления значения относительного сдвига.
* Команда JNC увеличивает значение программного счетчика на два перед добавлением значения относительного сдвига.
1. Как вычисляется отрицательный относительный сдвиг для команды JC?
* Команда JC добавляет значение относительного сдвига в дополнительном коде к програмному счетчику (вычитает относительный сдвиг).
* Команда JC увеличивает значение счетчика команд на 2, затем добавляет значение относительного сдвига в дополнительном коде (вычитает относительный сдвиг).
* Команда JC уменьшает значение счетчика команд на 2, затем добавляет значение относительного сдвига в дополнительном коде (вычитает относительный сдвиг).
1. Какое управляющее слово используется, чтобы установить Порт С ППИ-8255 в режим ввода?

***Примечание:*** Порт A и Порт B могут быть установлены в Режим 0. (X = любое состояние)

* 1XXX1XX1
* 0XXX1XX1
* 1XXX0XX1

Выключите лабораторный стенд.Отключите EB-153 от PU-2000.

**Лабораторная работа №5.**

**(**время выполнения 4 часа**)**

**«Выполнение арифметических и логических операций**

**микроконтроллером 8051»**

Цель работы:

1.Научиться производить вычисления на микрокомпьютере.

2.Заменять логические схемы, используя функции микрокомпьютера.

3.Писать программы, использующие арифметические и логические команды.

4.Писать свои собственные программы и подпрограммы.

**Оборудование**

* Лабораторный стенд PU-2000
* Печатная плата EB-153
* Комплект соединительных кабелей DL-20

Подключите EB-153 к PU-2000.Включите питание PU-2000.

**Теория**

С помощью 8051 можно выполнять логические операции И, ИЛИ и исключающее ИЛИ, используя регистр A в качестве второго операнда.

8051 также может осуществлять операции логического ИЛИ или логического И между булевым аккумулятором (т.е. флагом переноса) и любым битом или его обратным кодом, к которому возможна непосредственная адресация.

ANL и ORL - команды для выполнения логических операций между битом переноса и определенным битом.

Так как бит переноса играет роль "аккумулятора" в Булевом процессоре 8051, все результаты вышеупомянутых логических операций будут заноситься в бит переноса.

Изучите схему.



На рисунке изображена электрическая схема построенная на основе элементарных логических элементов и выполняющая некоторую логическую функцию. Функция содержит несколько Булевых переменных.

Математически, данная функция может быть записана следующим образом:

Q = U \* (V + W) + X

Следующая программа имитирует работу данной логической схемы.

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H MOVC,90H,#F0H устанавливает Порт 1 в исходное

 положение

E003H MOVC,91H читает бит 1 и передает его значение

 в бит переноса

E005H ORLC,92H логическое ИЛИ между битом 2 и

 битом переноса

E007H ANLC,90H логическое И между битом 0 и битом

 переноса

E009H ORLC,/93H логическое ИЛИ между обратным

 кодом бита 3 и битом переноса

E00BH MOV 97H,C вычисление конечного значения

E00DH LJMPE003H переходит к началу цикла

Порт 1 используется в качестве входов и выходов логической схемы.

Входы подключены к переключателям, а выходы - к светодиодам следующим образом:

ВХОД U = ПОРТ 1 - БИТ 0 (соединен с переключателем)

ВХОД V = ПОРТ 1 - БИТ 1 (соединен с переключателем)

ВХОД W = ПОРТ 1 - БИТ 2 (соединен с переключателем)

ВХОД X = ПОРТ 1 - БИТ 3 (соединен с переключателем)

ВЫХОД Q = ПОРТ 1 - БИТ 7 (соединен со светодиодом)

**Порядок работы**

Используйте команду GO для запуска программы.

Измените положение переключателей и заполните таблицы истинности, показанные на следующих экранах.



Снова запустите последнюю программу; сейчас в системе присутствует ошибка. Снова поменяйте положения переключателей в соответствии с таблицами истинности. Значение какого бита неправильное и как оно изменилось?

Нажмите кнопку INTR для остановки программы.

**Теория - 1**

8051 может осуществлять следующие арифметические операции: сложение, увеличение на единицу, уменьшение на единицу, сравнение с нулем, уменьшение на единицу и сравнение с нулем, коррекция десятичного сложения, вычитание с заемом, сравнение, умножение и деление.

 Команда DIVAB делит восьмибитное целое число без знака в аккумуляторе на восьмибитное целое без знака в регистре В.

Целая часть частного возвращается в аккумулятор; остаток - в регистр В.

Если регистр В первоначально содержал 00H, то будет выставлен флаг переполнения, чтобы отобразить ошибку деления, и конечные значения определены не будут.

В ином случае флаг переполнения будет сброшен.

Команда деления также полезна для таких целей, как преобразование систем счисления или разделение полей битов в аккумуляторе (от 0 до 255) на три цифры (три байта) в двоично-десятичном формате.

Программа показана на следующем экране.

Результат преобразования записывается в регистры R0, R1 и R2.

Рассмотрим программу:

АДРЕС КОМАНДА КОММЕНТАРИЙ

E100H MOVF0H,#64H Записывает в B (адр. F0H) число

 100 (десятичное)

E103H DIVAB Делит на 100, чтобы определить

 число сотен

E104H MOVR2,A Сохраняет результат в R2

E105H MOVA,#0AH Записывает в A число 10

 (десятичное)

E107H XCHA,F0H Обмен данными между A и B

 (адрес F0H)

E109H DIVAB Делит остаток на 10, чтобы

 определить число десятков

E10AH MOVR1,A Сохраняет результат в R1, остаток

 – одна цифра

E10BH MOVR0,F0H Сохраняет результат в R0

E10DH RET Возвращается к основной

 программе

Обратите внимание, что это программа является подпрограммой.

8051 имеет в своем распоряжении два вида команд, используемых для работы с подпрограммами.

Такими командами являются: СALL (ACALL или LCALL) и RETURN (RET).

Последней командой подпрограммы должна быть команда RET.

 **Эксперимент - 1**

Программа, показанная на следующем экране, использует подпрограмму преобразования системы счисления для преобразования содержимого аккумулятора и отображения результатов на дисплее.

Результаты преобразуются в ASCII-символы добавлением значения 30H (соответствующего символу "0").

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H LCALLFF89H Вызывает операцию очистки буфера

 дисплея

E003H MOVA,#FFH Устанавливает байт для преобразования

E005H LCALLE100H Вызывает программу

 преобразования системы счисления

E008H MOVA,#30H Записывает в аккумулятор ASCII "0"

E00AH MOVDPTR,#FF90H Устанавливает указатель данных на

 адрес буфера

E00DH ADDA,02H Преобразует первый байт в формат

 ASCII

E00FH MOVX @DPTR,A ЗаписываетвбуферпамятиASCII R2

E010H INCDPTR Увеличивает на 1 указатель буфера

 дисплея

E011H MOVA,#30H Записывает в аккумулятор ASCII "0"

E013H ADDA,01H Преобразует второй байт в формат

 ASCII

E015H MOVX @DPTR,A ЗаписываетвбуферпамятиASCIR1

E016H INCDPTR Увеличивает на 1 указатель буфера

 дисплея

E017H MOVA,#30H Записывает в аккумулятор ASCII "0"

E019H ADDA,00H Преобразует третий байт в формат

 ASCII

E01BH MOVX @DPTR,A Записывает в буфер памяти ASCIIR2

E01CH LCALLFF83H Вызывает операцию обработки

 буфера дисплея

E01FH LJMPE01FH Конец программы

Запустить программу с помощью команды GO.

Нажмите кнопку INTR, чтобы остановить программу.

Измените значение для преобразования с первоначального FFH (адрес E004H) на другое и запустите программу снова.

**Теория - 2**

Программа, показанная на следующем экране использует команду INC, чтобы увеличить на 1 значение двоичного счетчика, которое отображается с помощью светодиодов подключенных к Порту 1.

Рассмотрим программу.

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H MOVDPTR,#FFC0H Записывает в указатель данных

 значение задержки - самый старший

 бит

E003H MOVA,#01H Определяет постоянную задержки –

 самый старший бит

E005H MOVX @DPTR,A

E006H INCDPTR Записывает в указатель данных

 значение задержки - самый младший

 бит

E007H MOVA,#00H Определяет постоянную задержки –

 самый старший бит

E009H MOVX @DPTR,A

E00AH CLRA Сбрасывает значение регистра А на

 ноль

E00BH SWAPA Переставляет полубайты

E00CH MOV 90H,A Записывает в Порт1

E00EH SWAPA Восстанавливает значениеполубайтов

E00FH PUSHE0 Сохраняет аккумулятор (адрес E0H)

E011H LCALLFF8CH Вызывает программу задержки

E014H POPE0 Восстанавливает значение

 аккумулятора

E016H INCA Увеличивает значение счетчика на 1

E017H JBE4H,F0H Начинает сначала, если значение

 счетчика 15 (Аккумулятор- бит4=1)

 - переходит на E00AH

E01AH LJMPE00BH Переходит к началу цикла

Программа задержки (адрес FF8CH) обеспечивает задержку, определенную по адресам FFC0H (MSB) и FFC1H (LSB).

Базовое время - 1мс.

Таким образом, для данного в программе аргумента 0100H время задержки составляет 0.256 сек.

В аккумуляторе хранится значение счетчика.

Таким образом, при вызвове программы задержки, значение аккумулятора необходимо сохранить.

Простейший способ сохранить значение аккумулятора - это использовать команду PUSH, которая запишет значение аккумулятора в стек.

Команда POP восстановит значение аккумулятора.

SWAPA переставляет значения младшего полубайта аккумулятора (0-3) и старшего (4-7).

Операция передает значение счетчика на светодиоды, подключенные к выходам старших разрядов

 **Эксперимент - 2**

Запустить программу с помощью команды GO.

Проследите как выполняется индикация двоичного счета с помощью светодиодов.

Нажмите кнопку INTR, чтобы остановить программу.

Снова запустите программу для разных значений временной задержки.

Выключите лабораторный стенд.Отключите ЕВ-153 от PU-2000.

**Вопросы**

1. Что такое подпрограмма?
	* Короткая программа, вызываемая командами LCALL и ACALL. Ее последней командой должна быть команда RET.
	* Короткая программа для работы с элементами ввода-вывода.
	* Программа, используемая только управляющей системной программой. Ее последней командой должна быть команда RET.
2. Что такое Булевый аккумулятор?
	* Старший разряд аккумулятора.
	* Младший разряд аккумулятора.
	* Бит переполнения.
	* Бит переноса.
3. Где хранится програмный счетчик (командный счетчик), когда исполняется команда CALL?
* В стеке.
	+ В регистре R0.
	+ В ПЗУ.
1. Какие операции являются логическими?

INC, DEC, MOV, ADD, SUB

* + LCALL, ACALL, RET, DIV
	+ AND, OR, XOR, ROT, CLR, CPL and SWAP
	1. Какие операции являются арифметическими?
	+ INC, DEC, DIV, ADD, SUB, MUL, DA
	+ LCALL, ACALL, RET, RETI, CLR
	+ MOV, MOVX, PUSH, POP

**Лабораторная работа №6.**

**(**время выполнения 4 часа**)**

**« Выполнение операций передачи управления»**

Цель работы:

1.Изучить принципы управления стеком.

2.Изучить программы, использующие команды передачи управления.

3.Изучить программы для управления устройством звуковой сигнализации.

**Теория**

Существуют 3 класса операций управления:

* + безусловные вызовы, возвраты и переходы
	+ условные переходы
	+ прерывания

Все операции передачи управления изменяют (некоторые при определенных условиях) ход выполнения программы. То есть выполнение программы продолжается с адреса, определенного командой передачи управления.

 Безусловные вызовы, возвраты и переходы определяют новый адрес, с которого должно продолжиться выполнение программы.

ACALL и LCALL заносят адрес следующей команды в стек (PCL для младших разрядов адреса и PCH для старших разрядов адреса) и затем определяют новый адрес, с которого программа должна продолжить свое выполнение.

RET передает управление адресу, записанному в стеке предыдущей операцией вызова и уменьшает значение регистра SP на два для того, чтобы SP соответствовал правильному адресу стека.

 Стек можно организовать в любой области внутреннего ОЗУ данных.

8-битовый указатель стека (SP) содержит адрес стека, по которому был записан последний байт.

Это также адрес стека из которого будет извлечены данные.

Программа может записать и считать данные из SP.

При использовании стека необходимо придерживаться следующих принципов:

* Определяйте указатель стека в процессе инициализации.
* Некоторые команды требуют обязательного использования других команд: для каждой команды ВЫЗОВА подпрограммы (Call), должна быть использована команда ВОЗВРАТА (Ret), а для каждой записи в стек (PUSH) - команда считывания из стека (POP).
* Вы должны восстанавливать данные регистров (команда РОР) в обратном порядке их сохранения (команда РUSH).

Изучите следующую программу.

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H MOV 81H,#30H Записывает в указатель стека (81H) 30H

E003H MOVA,#55H Записывает в аккумулятор 55H

E005H PUSHE0H Сохраняет значение аккумулятора (E0)

E007H LCALLE100H Вызов подпрограммы

E00AH POPE0H Восстановление аккумулятора (E0)

E00CH SJMPF2H Переход на первый адрес программы

 (E000)

Подпрограмма

E100H CLRA Изменяет аккумулятор

E101H RET Возврат к главной программе

Команда SJMP передает управление команде, записанной по определенному адресу.

В приведенном примере счетчик команд увеличивается на 1, затем его значение подвергается операции относительного смещения (F2H) и программа продолжает работу с адреса E000H.

Адрес кода при относительном переходе должен быть закрыт для самой команды относительного перехода.

Диапазон значений от -128 до +127 байт, начиная с первого байта команды, которая следует за относительным переходом.

Адреса кодов представлены в виде восьмибитных дополнительных кодов.

Поэтому для отрицательных адресов дополнительный код вычисляется как обратное значение относительного смещения плюс 1.

**Оборудование**

* Лабораторный стенд PU-2000
* Печатаная плата EB-153
* Комплект соединительных кабелей DL-20
* Осциллограф

Подключите EB-153 к PU-2000.Включите лабораторный стенд.

**Порядок работы**

Следующая программа показывает пример использования стека и примеры безусловных вызовов, возвратов и переходов.

Введите в память микрокомпьютера программу, показанную ниже.

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H MOV 81H,#30H Записывает в указатель (81H) 30H

E003H MOVA,#55H Записывает в аккумулятор 55H

E005H PUSHE0H Сохраняет значение аккумулятора (E0)

E007H LCALLE100H Вызов подпрограммы

E00AH POPE0H Восстанавливает аккумулятор (E0)

E00CH SJMPF2H Переход к начальному адресу (E000)

Подпрограмма

E100 HCLRA Изменяет аккумулятор

E101 HRET Возврат к главной программе

Запустить программу с помощью команды STEP.

Убедитесь, что подпрограмма (адрес E100H) изменяет только содержимое аккумулятора.

Обратите внимание на изменения указателя стека во время операций управления стеком (PUSH, POP, CALL и RET).

Нажмите кнопку INTR, чтобы остановить программу.

**Теория - 1**

Следующая программа используется для генерации звукового сигнала, который воспроизводиться пьезоэлектрическим элементом, который подключен к битам 0 и 1 Порта 1.

Все регистры, используемые главной программой, должны быть сохранены в стеке, во время вызова подпрограммы задержки времени.

На пъезоэлемент подаются звуковые частоты, которые генерируются посредством изменения состояния 0-го и 1-го контактов Порта 1.

Изучите программу.

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H MOVR0,#80H Определяет постоянную задержки

E002H MOV 96H,C Определяет значение одной из линий

 пъезоэлемента

E004H CPLC Инвертирует бит переноса

E005H MOV 97H,C Определяет значение другой линии

 пъезоэлемента

E007H PUSHD0H Сохраняет бит переноса

E009H LCALLE100H Вызывает программу задержки

E00CH POPD0H Восстанавливает бит переноса

E00EH LJMPE000H Возврат в начало цикла

Подпрограмма

E100H DJNZR0,01H Задержка

E102H RET Возврат если конец счета

E103H LJMPE100H Продолжает

Программа использует команду условного перехода DJNZ.

В приведенном примере, данная команда уменьшает значение регистра R0 на 1 и помещает результат в определенный регистр.

Если результат уменьшения регистра равен 0, то управление переходит к следующей команде; в ином случае, управление переходит к команде, расположенной по определенному адресу.

Программный счетчик увеличивается на 1 и указывает на адрес следующей команды.

Если результат уменьшения не 0, тогда значение относительного смещения прибавляется к значению счетчика. После этого выполняется команда по соответствующему адресу.

Подпрограмма E100H обеспечивает простую задержку в соответствии со значением, хранящимся в R0.

**Эксперимент - 1**

Введите программу в память микрокомпьютера.

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H MOVR0,#80H Определяет постоянную задержки

E002H MOV 96H,C Устанавливает состояние одной из линий пьезоэлемента

E004H CPLC Инвертирует бит переноса

E005H MOV 97H,C Устанавливает состояние другой линии пьезоэлемента

E007H PUSHD0H Сохраняет бит переноса

E009H LCALLE100H Вызов программы задержки

E00CH POPD0H Восстанавливает бит переноса

E00EH LJMPE000H Возврат в начало цикла

Подпрограмма

E100H DJNZR0,01H Задержка

E102H RET Возврат, если конец счета

E103H LJMPE100H Продолжает

Запустить программу с помощью команды GO.

Нажмите кнопку INTR, чтобы остановить программу.

Измените постоянную задержки (адрес E001H), чтобы задать другие частоты и снова запустите программу.

**Теория - 2**

Следующая программа включает все светодиоды, подключенные к Порту 1, если переключатель, подключенный к биту 0 Порта 1, установлен в положение "ON".

В ином случае светодиоды выключены.

Изучите программу.

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H CLRC Очищает бит переноса

E001H JNB 90H,01H Проверяет переключатель

E004H SETBC Определяет бита переноса

E005H MOV 94H,C Включает или выключает LED

E007H MOV 95H,C Включает или выключает LED

E009H MOV 96H,C Включает или выключает LED

E00BH MOV 97H,C Включает или выключает LED

E00DH LJMPE000H Возврат в начало цикла

Команда JNB проверяет значение по определенному адресу (в данном примере 90H).

Если проверяемое значение равно 0, управление переходит к определенному адресу программы.

В ином случае управление переходит к следующей команде.

Программный счетчик увеличивается на единицу и указывает на следующую команду.

Если проверка успешна, то значение счетчика увеличивается на значение относительного смещения. Таким образом выполняется команда, адрес которой соответствует значению программного счетчика.

**Эксперимент - 2**

Введите программу в память микрокомпьютера.

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H CLRC Очищает бит переноса

E001H JNB 90H,01H Проверяет переключатель

E004H SETBC Устанавливает бита переноса

E005H MOV 94H,C Включает или выключает LED

E007H MOV 95H,C Включает или выключает LED

E009H MOV 96H,C Включает или выключает LED

E00BH MOV 97H,C Включает или выключает LED

E00DH LJMPE000H Возврат в начало цикла

Запустить программу с помощью команды STEP.

Поменяйте положение переключателя, подключенного к биту 0 Порта1 и проследите как изменилась индикация светодиодов.

Нажмите кнопку INTR, чтобы остановить программу.

Выключите лабораторный стенд. Отключите ЕВ-153 от PU-2000.

**Вопросы**

1. В чем заключается разница между командами PUSH и POP?
	* PUSH используется для сохранения значений регистра, POP для восстановления значений регистров.
	* PUSH восстанавливает значения регистров из стека, POP сохраняет значения регистров.
	* Команда PUSH увеличивает значение программного счетчика на 1, а POP уменьшает на 1 значение программного счетчика.
2. Где находится стек?
* Он находится во внешнем ОЗУ.
* Он находится внутри ПЗУ.
* Он находится во внутреннем ОЗУ.
1. Что такое указатель стека?
* Это 8-битовый регистр, который содержит адрес, в который был записан первый байт стека.
* Это 8-битовый регистр, который содержит начальный адрес стека.
* Это 8-битовый регистр, который содержит адрес, в который был записан последний байт стека.
1. В чем заключается разница между командами JB и JNB?
* Команда JB производит переход, если адрес тестируемого бита равен 0, а команда JNB производит переход, если адрес проверяемого бита равен 1.
* Команда JB производит переход, если проверяемый бит имеет значение 1, а команда JNB производит переход, если проверяемый бит имеет значение 0.
* Команда JB производит переход, если есть отрицательный перенос, а команда JNB производит переход, если его нет.

**Лабораторная работа №7.**

**(**время выполнения 4 часа**)**

**«Исследование внешнего прерывания- операции передачи управления»**

Цель работы:

1.Изучить внешние прерывания.

2. Научиться писать программы обработки прерываний.

3.Управлять работой программы в режиме реального времени.

**Теория**

Прерывание - это операция передачи управления.

Внутренние и внешние прерывания могут влиять на ход программы.

При выполнении любого прерывания, значение программного счетчика сохраняется в стеке, а затем происходит переход к программе, сохраненной в памяти.

Программист должен записать в стек значения всех регистров, которые будут изменены программой обработки прерывания, чтобы избежать потери данных.

 Последней командой программы обработки прерываний должна быть команда RETI.

Эта команда возвращает управление основной программе аналогично команде RET.

К тому же, RETI разрешает повторное прерывание для текущего уровня приоритета прерывания.

8051 может получить запрос прерывания от пяти источников: два от внешних источников через выводы INT0 и INT1, два от двух внутренних счетчиков и один через последовательный порт ввода-вывода.

Каждый запрос прерывания инициирует выполнение соответствующей служебной программы, хранящейся в определенной ячейке памяти.

|  |  |  |  |
| --- | --- | --- | --- |
| Источник прерывания | Описание | Начальный арес | Адрес перехода управляющей программы |

INT0 внешний запрос 00003H FF00H

RxDorTxD внутренний 0023H FF10H

 последовательныйпорт

T1 внутренний/счетчик 1 001BH FF20H

T0 внутренний/счетчик 0 0011H резервируется

INT1 внешний запрос 1 0019H резервируется

Ниже показаны области для хранения программ обработки прерываний.

**Оборудование**

* \*Лабораторный стенд EB-2000
* \*Печатная плата EB-153
* \*Комплект соединительных кабелей DL-20
* \*Осциллограф
* \*Генератор сигналов

Подключите EB-153 к PU-2000.Включите питание лабораторного стенда.

**Порядок работы**

С помощью команды DASM выведите на экран адреса перехода системной управляющей программы.

Наберите на клавиатуре DASM 0003H и нажмите клавишу ENTER.

Обратите внимание на команду LJMP.

Используйте кнопку "стрелка вниз", чтобы вывести на экран другие адреса перехода.

Нажмите кнопку RESET, чтобы остановить выполнение команды DASM.

**Теория - 1**

Каждому из пяти источников прерываний, указанных выше, может быть присвоен один из двух уровней приоритета. Также каждый из источников может быть разрешен или запрещен независимо от других источников прерывания.

К тому же, все разрешенные источники могут быть одновременно разрешены или запрещены.

Каждое внешнее прерывание может быть запрограммировано на активизацию либо низким уровнем сигнала либо переходом с высокого уровня на низкий.

Источник запрашивает прерывание, выставляя соответствующий флаг запроса прерывания в регистре TCON или SCON.

На рисунке показан формат регистра TCON (Timer/CounterControl/Status) , таймер/счетчик управление/состояние.



Рисунок показывает формат регистра IE (InterruptEnable), разрешение прерываний.



IP - приоритет прерывания - формат регистра показан ниже.



**Эксперимент - 1**

При использовании следующей программы микрокомпьютер работает в режиме счетчика событий.

Введите в память компьютера программу, показанную на следующих экранах.

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H MOVDPTR,#FF00H устанавливает указатель на адрес

 перехода INT0

E003H MOV A,#02H LJMP код

E005H MOVX @DPTR,A записываетLJMPE100Hпоадресу

 FF00H

E006H INCDPTR указывает на первый операнд

E007H MOVA,#E1H старший разряд адреса

E009H MOVX @DPTR,A записывает первый операнд

E00AH INCDPTR указывает на второй операнд

E00BH MOVA,#00H младший разряд адреса

E00DH MOVX @DPTR,A записывает второй операнд

E00EHL CALLFF80H очищает дисплей

E011H MOVB8H,#01H устанавливает высший приоритет

 для INT0 в регистре IP, адрес B8H

E014H MOV 88H,#01H устанавливает "задний фронт" для

 INT0 в регистр TCON, адрес 88H

E017H MOVA8H,#81H устанавливает "разрешен" для INT0

 регистр IE, адрес A8H

E01AH LJMPE01AH ждет прерывания

(Продолжение программы)

АДРЕС КОМАНДА КОММЕНТАРИЙ

**программа обработки прерываний**

E100H MOVDPTR,#E200H устанавливает указатель на адрес

 счетчика

E103H MOVXA,@DPTR считывает значение счета

E104H INCA увеличивает счет на единицу

E105H CJNEA,#0AH,01H проверяет, равен ли счет 10

E108H CLRA старший разряд адреса

E109H MOVX @DPTR,A сохраняет счет

E10AH ADDA,#30H ASCII преобразование

E10CH MOVDPTR,#FFB1H устанавливает адрес символа

E10FH MOVX @DPTR,A записывает символ

E110H CLRA определяет адрес символа = 0

E111H MOVDPTR,#FFB0H устанавливает адрес дисплея

E114H MOVX @DPTR,A сохраняет адрес дисплея

E115H LCALLFF86 отображает символ

E118H RETI прекращает режим прерывания

Установите генератор сигналов в следующий режим: генерирование прямоугольных импульсов, уровень TTL, частота 1Гц.

Подключите генератор сигналов к контакту INT0.

Используйте команду GO, чтобы запустить программу.

Убедитесь, что на дисплей выводится количество принятых входных импульсов.

Увеличьте частоту генератора и посмотрите как производится подсчет импульсов при новой частоте.

Нажмите кнопку RESET, чтобы остановить программу.

**Теория - 2**

Следующая программа подает сигнал на пьезоэлектрический зуммер, подключенный к контактам битов 6 и 7 Порта 1.

Программа, показанная на следующем экране, ожидает прерывания, чтобы изменить значения в разрядах порта, подключенных к звуковому устройству.

**Эксперимент - 2**

Введите программу

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H MOVDPTR,#FF00H устанавливает указатель на адрес

 перехода INT0

E003H MOV A,#02H LJMP код

E005H MOVX @DPTR,A записывает LJMP E100H в FF00H

E006H INCDPTR указывает на первый операнд

E007H MOVA,#E1H старший разряд адреса

E009H MOVX @DPTR,A записывает первый операнд

E00AH INCDPTR указывает на второй операнд

E00BH MOVA,#00H младший разряд адреса

E00DH MOVX @DPTR,A записывает второй операнд

E00EH SETB 96H устанавливает звуковое устройство –

 Порт 1 - бит 6

E010H CLR 97H устанавливает звуковое устройство –

 Порт 1 - бит 7

E011H MOVB8H,#01H определяет IP регистр

E014H MOV 88H,#01H определяет TCON регистр

E017H MOVA8H,#81H определяет IЕ регистр

E01AH LJMPE01bH ждет прерывания

**программа обработки прерываний**

E100H CPL 96H издает звук

E102H CPL 97H издает звук

E104H RETI возврат из режима прерывания

Установите следующий режим на генераторе сигналов: генерирование прямоугольных импульсов, уровень TTL, частота 10Гц.

Подключите генератор сигналов к контакту INT0.

Используйте команду GO, чтобы запустить программу.

Обратите внимание на то, что звуковое устройство издает сигнал с частотой генератора.

Увеличьте частоту генератора сигналов и обратите внимание как изменилась частота звукового сигнала.

Нажмите кнопку RESET, чтобы остановить программу.

Выключите PU-2000.Отключите ЕВ-153 от PU-2000.

**Вопросы**

1. Что такое прерывание?
	* Прерывание - это операция передачи управления.
	* Прерывание вызывает остановку программы.
	* С помощью прерываний выполняется управление переферийными устройствами.
2. Сколько источников прерывания у 8051?
	* Два, внешний и внутренний.
	* Пять, два внешних и три внутренних.
	* Одно внешнее прерывание.
3. Как активизируются внешние прерывания 8051?
	* Внешние прерывания активизируются внешним устройством, присоединенным к INT0 или INT1.
	* Внешние прерывания активизируются только RxD или TxD последовательных портов.
	* Внешние прерывания активизируются только T0 или T1 устройствами таймер/счетчик.

**Лабораторная работа №8.**

**(**время выполнения 4 часа**)**

**«Исследование таймеров и счетчиков событий»**

Цель работы:

1.Изучить принципы работы счетчиков событий и таймеров 8051.

2.Научится изменять конфигурацию входа таймера/счетчика.

3.Научиться генерировать различные типы временных сигналов.

**Теория**

8051 содержит два 16-битных счетчика для измерения временных интервалов, ширины импульсов, счета событий и генерации точных периодических запросов прерывания.

Каждый из них может быть независимо от других запрограммирован, как 8-битный таймер или как счетчик с предварительным делителем частоты на 32 (Режим 0), как 16-битный счетчик временных интервалов или событий (Режим 1) или как 8-битный счетчик временных интервалов или событий с автоматической перезагрузкой при переполнении (Режим 2).

 Если тактовая частота подается от внешнего источника через входы T0 или T1, счетчик событий/таймер называется счетчиком.

Если вход счетчика подключен к встроенному генератору частоты, он называется таймером.

Режим работы таймера/счетчика определяется двумя 8-битовыми регистрами, TMOD (режим таймера) и TCON (управление таймером).

В зависимости от того, равен бит C/T регистра ТМОD единице или нулю ввод в счетчик может быть подключен к внешнему устройству (для использования его в качестве счетчика) или ко встроенному генератору (для использования его в качестве таймера) соответственно.

Для измерения времени с помощью таймера, на его вход подаются импульсы с частотой, равной частоте встроенного генератора уменьшенной в двенадцать (12) раз.

 Когда бит стробирующего сигнала (GATE) регистра ТMOD равен 1, сигнал от внешнего устройства (T1, T0) или от встроенного генератора воспринимается счетчиком при условии, что уровень сигнала на втором внешнем входе (INT0, INT1) имеет высокий уровень.

Когда бит стробирующего сигнала (GATE) равен нулю, сигнал от внешнего устройства или внутреннего генератора разрешен безусловно.

В любом случае, обычная функция прерывания INT0 и INT1 не зависит от режима счетчика.

 Инкрементирование значений счетчиков разрешено если биты TR1 и TR0 регистра TCON равны единице.

Когда счетчики переполнены, биты TF1 и TF0 регистра TCON принимают значение единицы, что приводит к генерации запросов прерывания.

 Функции битов регистра TMOD показаны на рисунке.



**Управление стробирующим сигналом. Если данный бит равен 1, счетчик/таймер "Х" активен только когда уровень сигнала на выводе "INTx" высокий и бит управления "TRx" равен 1.**

**Если данный бит равен 0 счетчик/таймер "Х" активен всегда, когда бит управления "TRx" равен 1.**

**Выбор режима таймера или счетчика. Если данный бит равен 0, то счетчик/таймер работает в режиме таймера (сигнал поступает от встроенного генератора импульсов). Если данный бит равен 1, то счетчик/таймер работает в режиме счетчика (сигнал поступает с вывода "Тх").**

Следующая программа устанавливает Т1 (счетчик номер 1) в режим 8-битного автоперезагружаемого счетчика (C/T=1).

T1 установлен в режим 2.

Старший разряд T1, TH1, содержит значение перезагрузки F6H.

Младший разряд TL1 увеличивается на единицу после каждого тактового импульса.

Значение в TH1 перезагружается в TL1, когда TL1 переполняется.

Это происходит после 10 импульсов.

Каждые 10 импульсов таймер переполняется и на экран выводится значение счета.

**Порядок работы**

Введите следующую программу в память микрокомпьютера.

АДРЕС КОМАНДА КОММЕНТАРИЙ

**главная программа**

E000 MOVDPTR,#FF20H устанавливает указатель на адрес

 перехода к T1

E003 MOVA,#02H LJMP код

E005 MOVX @DPTR,A записывает LJMP E100H в FF20H

E006 INCDPTR указывает на первый операнд

E007 MOVA,#E1H старший разряд адреса

E00 9MOVX @DPTR,A записывает первый операнд

E00A INCDPTR указывает на второй операнд

E00B MOVA,#00H младший разряд адреса

E00D MOVX @DPTR,A записывает второй операнд

E00E LCALLFF80H очищает дисплей

E011 MOVB8H,#08H определяет IP регистр - адрес B8H

 T1 высший приоритет

E014 MOV 89H,#60H определяет TMOD регистр - адрес 89H,

 Режим 2 - 8-битовая автоперезагрузка

E017 MOVA8H,#88H определяет IE регистр - адрес A8H,

 Прерывание Таймера 1 разрешено

E01A MOV 8DH,#F6H определяет TH1 регистр - адрес 8DH,

 перезагружает F6H=FFH-10D+1

E01D MOV 88H,#40H определяет TCON регистр - адрес 88H,

 запускает Таймер 1

E020 LJMPE020H ждет прерывания

**программа обработки прерываний**

E100 MOVDPTR,#E200H устанавливает указатель на адрес

 счетчика

E103 MOVXA,@DPTR считывает счетчик

E104 INCA увеличивает значение счетчика на 1

E105 CJNEA,#0AH,01H проверяет, не равно ли значение

 счетчика 10

E108 CLRA старший разряд адреса

E109 MOVX @DPTR,A сохраняет значение счетчика

E10A ADD A,#30HASCII преобразование ASCII

E10C MOVDPTR,#FFB1H устанавливает адрес символа

E10F MOVX @DPTR,A сохраняет символ

E110 CLRA определяет адрес символа как 0

E111 MOVDPTR,#FFB0H определяет адрес дисплея

E114 MOVX @DPTR,A записывает адрес дисплея

E115 LCALLFF86H символ дисплея

E118 RETI возвращается из режима прерывания

Установите следующий режим для генератора сигналов: генерирование прямоугольных импульсов, уровень TTL, частота 10Гц.

Подключите генератор сигналов к Т1.

Запустить программу с помощью команды GO.

Проследите за тем, как программа выводит на дисплей значение счета для каждых 10 входных импульсов.

Увеличьте частоту генератора сигналов и посмотрите как изменилась индикация на дисплее.

Нажмите кнопку RESET, чтобы остановить программу.

**Теория - 1**

Следующая программа устанавливает режим 16-битового таймера (режим 1) для T1 .

Для T1 устанавливается значение счета равное 10 000 и 100 для программы обработки прерывания.

Поэтому общее значение счета равно 1 000 000.

Единица времени - 1 микросекунда, то есть частота встроенного генератора, деленная на 12.

Каждую секунду таймер переполняется и на дисплее появляется значение второго счетчика.

Загруженное значение вычисляется как

FFFFH-2710H+1=D8E1H,где 2710H=10000D.

**Эксперимент - 1**

Введите следующую программу в память микрокомпьютера.

АДРЕС КОМАНДА КОММЕНТАРИЙ

**главная программа**

E000 MOVDPTR,#FF20H устанавливает указатель на адрес

 перехода T1

E003 MOVA,#02H LJMP код

E005 MOVX @DPTR,A записывает LJMP E100H в FF20H

E006 INCDPTR указывает на первый операнд

E007 MOVA,#E1H старший разряд адреса

E009 MOVX @DPTR,A записывает первый операнд

E00A INCDPTR указывает на второй операнд

E00B MOVA,#00H младший разряд адреса

E00D MOVX @DPTR,A записывает второй операнд

E00E LCALLFF80H очищает дисплей

E011 MOVB8H,#08H определяет IP регистр - адрес B8HT1,

 высший приоритет.

E014 MOV 89H,#10H определяет TMOD регистр, адрес 89H,

 Режим 1 - 16-битный таймер

E017 MOVA8H,#88H определяет IE регистр, адрес A8H

разрешает прерывание Таймера 1

E01A MOV 8DH,#D8H определяет TH1 регистр, адрес 8DH

 значение перезагрузки D8H

E01D MOV 8BH,#E1H определяет TL1 регистр, адрес 8BH

 значение перезагрузки E1H

E020 MOV 88H,#40H определяет TCON регистр - адрес 88H

 Запускает Таймер 1

E023 CLRA очищает программный счетчик (100)

E024 MOVDPTR,#E300H устанавливает указатель на адрес

 счетчика

E027 MOVXA,@DPTR устанавливает начальное значение

E028 LJMPE028H ждет прерывания

**программа обработки прерываний**

E100 MOVDPTR,#E300H устанавливает указатель на адрес

 программного счетчика

E103 MOVXA,@DPTR считывает значение счета

E104 INCA увеличивает значение счета на 1

E105 CJNEA,#64H проверяет, если счет=100, то переход

Программа – продолжение.

АДРЕС КОМАНДА КОММЕНТАРИЙ

**обновление дисплея**

E108 MOVDPTR,#E200H устанавливает указатель на адрес

 счетчика

E10B MOVXA,@DPTR читает значение счета

E10C INCA увеличивает значение счета на 1

E10D CJNEA,#0AH,01H проверяет, равно ли значение счета 10

E110 CLRA старший разряд адреса

E111 MOVX @DPTR,A сохраняет счет

E112 ADDA,#30H ASCII преобразование

E114 MOVDPTR,#FFB1H устанавливает адрес символа

E117 MOVX @DPTR,Aз записывает символ

E118 CLRA определяет адрес символа=0

E119 MOVDPTR,#FFB0H устанавливает адрес дисплея

E11C MOVX @DPTR,A записывает адрес дисплея

E11D LCALLFF86H выводит символ на дисплей

E120 MOVDPTR,#E300H устанавливает указатель на адрес

 счетчика

E123 CLRA очищает значение счета

**обновление программного счетчика**

E124 MOVX @DPTR,A обновляет программный счетчик

E125 MOV 8DH,#D8H определяет регистр TH1, адрес 8DH,

 значение перезагрузки D8H

E128 MOV 8BH,#E1H определяет регистр TL1, адрес 8BH,

 значение перезагрузки E1H

E12B MOV 88H,#40H определяет регистр TCON, адрес 88H,

 перезапускает Таймер 1

E12E RETI возврат из режима прерывания

Запустить программу с помощью команды GO.

Пронаблюдайте за тем, как программа выводит на экран значение второго счетчика.

Нажмите кнопку RESET, чтобы остановить программу.

**Теория - 2**

Следующая программа устанавливает T1 (таймер номер 1) в режим 8-битового автоперезагружаемого таймера (C/T=0).

T1 в режиме 2.

Когда таймер переполнен, на бите 7 Порта 1 генерируется прямоугольная волна.

**Эксперимент - 2**

Введите следующую программу.

АДРЕС КОМАНДА КОММЕНТАРИЙ

**главная программа**

E000 MOVDPTR,#FF20H устанавливает указатель на адрес

 перехода T1

E003 MOVA,#02H LJMP код

E005 MOVX @DPTR, записывает LJMP E100H в FF20H

E006 INCDPTR указывает на первый операнд

E007 MOVA,#E1H старший байт адреса

E009 MOVX @DPTR,A записывает первый операнд

E00A INCDPTR указывает на второй операнд

E00B MOVA,#00H младший разряд адреса

E00D MOVX @DPTR,A записывает второй операнд

E00E LCALLFF80H очищает дисплей

E011 MOVB8H,#08H определяет регистр IP, адрес B8H,

 T1 высший приоритет

E014 MOV 89H,#20H определяет регистр TMOD, адрес 89H,

 Режим 2 - 8-битный перезагружаемый

E017 MOVA8H,#88H определяет регистр IE, адрес A8H,

 разрешает прерывание Таймера 1

E01A MOV 8DH,#01H определяет регистр TH1, адрес

 8DH, значение перезагрузки 01H

E01D MOV 88H,#40H определяет регистр TCON, адрес

 88H, запускает Таймер1

E020 LJMPE020H ждет прерывания

**программа обработки прерываний**

E100 CPL 97H инвертирует бит 7 Порта 1

E102 RETI возвращается из режима прерывания

Запустить программу с помощью команды GO.

Обратите внимание на прямоугольную форму сигнала на осциллографе.

Измените программу, чтобы получить различные значения частоты (измените значения адреса F01CH).

Нажмите кнопку RESET, чтобы остановить программу.

**Вопросы**

1. В чем различие между таймером и счетчиком в 8051?
	* Счетчик требует подачи внешнего тактового сигнала, а таймер использует сигнал от встроенного тактового генератора.
	* Различий нет.
	* Оба и счетчик и таймер работают от встроенного генератора, но счетчик также управляется внешним сигналом на входе Т0.
2. Сколько режимов работы существует у таймеров и счетчиков в 8051?
	* Один режим: таймер/счетчик.
	* Два режима: режим таймера, Режим 1, и режим счетчика, Режим 2.
	* Четыре режима: Режим0, Режим1, Режим2 и Режим3.

**Лабораторная работа №9.**

**(**время выполнения 4 часа**)**

**«Изучение принципа работы порта последовательной передачи данных»**

Цель работы:

1.Изучить принципы работы порта последовательной передачи данных 8051.

2.Научиться передавать ASCII символы в асинхронном формате.

3.Устанавливать связь между EB-153 и персональным компьютером.

**Теория**

8051 имеет последовательный порт ввода-вывода, который нужен для связи с периферийными устройствами через стандартный асинхронный интерфейс в полнодуплексном режиме.

 Последовательный порт также может работать в синхронном режиме.

Полнодуплексный последовательный порт ввода-вывода может работать в асинхронных режимах, для поддержки связи со стандартными UART устройствами, такими как персональные компьютер, принтеры и т.д.

 Во время связи со стандартными UART устройствами последовательный канал может быть запрограммирован для приема/передачи 10-битового блока данных (фрейма) (Режим 1) или для приема/передачи 11-битового блока данных (фрейма) (Режим 2 или 3).

Фрейм содержит стартовый бит, восемь или девять битов данных и стоп бит.

В Режиме 1 и 3 модуль синхронизации передачи получает импульс от счетчика 1 каждый раз, когда счетчик переполняется.

Вход счетчика 1 может быть подключен к внешнему источнику или получать внутренний сигнал, частота которого равна одной двенадцатой частоты встроенного генератора.

 Режим автоперезагрузки счетчика T1 обеспечивает связь на скорости 122 - 31250 бит в секунду (включая стартовый и стоп биты), при частоте встроенного кварцевого генератора 12 МГц.

Управление последовательным портом и получение информации о его состоянии обеспечивается регистром управления последовательного порта (SCON).

 Содержимое 8-битового SCON - регистра показано ниже.



**Режим последовательного порта. Бит 0. Установка/сброс выполняется програмно см. примечание.**

**Режим последовательного порта. Бит 1. Установка/сброс выполняется програмно см. примечание.**

**Режим последовательного порта. Бит 2. Установка выполняется программой для отмены приема фреймов, в которых бит 8=0.**

**Бит разрешения приема. Установка/сброс выполняется программой для разрешения/запрета приема последовательных данных.**

**Примечание:**

**Состояние SM0, SM1 определяет:**

**(0,0) расширение I/O (входа/выхода) регистра сдвига**

**(0,1) фрейм 8 бит, переменная скорость передачи**

**(1,0) фрейм 9 бит, фиксированная скорость передачи**

**(1,1) фрейм 9 бит, переменная скорость передачи**



**Передача бита 8. Установка/сброс выполняется аппартным оборудованием для определения состояния 9-ого бита передаваемого в режиме 9-ти битного UART.**

**Прием бита 8. Установка/сброс выполняется аппратным оборудованием для индикации состояния 9-ого бита принятых данных.**

**Флаг прерывания передачи. Устаналиваетсяаппратным оборудованием после передачи байта. Сбрасывается программой после обработки данных.**

**Флаг прерывания приема. Устанавливается аппаратным оборудованием когда байт принят. Сбрасывается программой после обработки данных.**

Биты управления режимом SM0 и SM1 программируют последовательный порт для работы в одном из четырех режимов.

Режим 1 устанавливает UART интерфейс в режим приема/передачи 10-битового фрейма (8 бит данных, 1 стоп бит и 1 старт бит) при переменной скорости передачи.

Данные для передачи и приема находятся в регистре буфера последовательного порта (SBUF).

 Запись в SBUF обновляет регистр передатчика, тогда как при чтении из SBUF считывается информация из буфера, который обновляется регистром приемника если/когда флаг RI сброшен.

Приемник имеет два буфера, чтобы избежать переполнения, которое может произойти, если ЦП не ответит на прерывание приемника перед началом следующего фрейма.

Бит разрешения приема (REN) служит для перезагрузки алгоритма начала/конца приема.

Когда программа записывает в REN единицу (1), генератор скорости приема данных устанавливается в исходное состояние и обеспечивается возможность приема.

 Значение REN должно быть установлено при инициализации последовательного канала.Если значение REN равно нулю, прием запрещен.

ЦП получает информацию, что область передатчика в SBUF пустая или область приемника заполнена, считывая значения T1 и R1 соответственно.

Значения T1 и R1 должны быть очищены программой обработки прерывания, чтобы не создавать непрерывного цикла прерывания ЦП.

Так как запрос прерывания последовательного порта является результатьм логического сложения (ИЛИ) значений T1 и R1, эти биты должны быть последовательно опрошены, чтобы определить источник прерывания.

**Оборудование**

* Лабораторный стенд PU-2000
* Печатная плата EB-153
* Комплект соединительных кабелей DL-20
* Осциллограф
* Персональный компьютер

Подключите EB-153 к PU-2000.Включите PU-2000.

**Порядок работы**

Следующая программа устанавливает Таймер №1 в режим генератора скорости передачи данных со значением 2400 бод и определяет конфигурацию порта последовательной передачи данных.

Данная программа передает значение символа ASCII "A" (41H) по TxD линии.

Введите программу в память микрокомпьютера.

АДРЕС КОМАНДА КОММЕНТАРИЙ

**главная программа**

E000 MOVDPTR,#FF20H устанавливает указатель на адрес

 перехода T1

E003 MOVA,#02H LJMP код

E005 MOVX @DPTR,A записывает JMP E100H в FF20H

E006 INCDPTR указывает на первый операнд

E007 MOVA,#E1H старший разряд адреса

E009 MOVX @DPTR,A записывает первый операнд

E00A INCDPTR указывает на второй операнд

E00B MOVA,#00H младший разряд адреса

E00D MOVX @DPTR,A записывает второй операнд

E00E MOVB8H,#10H определяет регистр IP, адрес B8H, T1

 высший приоритет

E011 MOV 89H,#20H определяет регистр TMOD, адрес 89H,

 режим 2- 8-битная перезагрузка

(Продолжение программы.)

АДРЕС КОМАНДА КОММЕНТАРИЙ

E014 MOV 8DH,#F3H определяет регистр TH1, адрес 8DH,

 значение перезагрузки F3Hдля

 скорости двоичной передачи=2400

E017 MOV 98H,#40H определяет регистр SCON, адрес 98H,

 режим 1 - запрещен приемник TI=1

E01A MOV 88H,#40H определяет регистр ТCON, адрес 88H,

 запускает Таймер 1

E01D MOVA8H,#90H определяет регистр IE, адрес A8H,

 прерывание последовательного порта

 разрешено

E020 MOV 99H,#41H устанавливает SBUF (99H) в "A"

E023 LJMPE023H ждет прерывания

**программа обработки прерываний**

E100 MOV 99H,#41H устанавливает SBUF (99H) в "A"

E103 CLR 99H

E102 RETI возвращается из режима прерывания

Подключите осциллограф к клемме TxD.

Запустить программу с помощью команды GO.

На экране осциллографа должен появиться сигнал асинхронного формата (8 битов данных, 1 стартовый импульс и 1 стоп импульс).

Обратите внимание, что сигналы имеют логические уровни присущие RS-232, а не TTL уровни, и они инвертированы.

Нажмите кнопку RESET, чтобы остановить программу.

Запустите данную программу еще раз для другого ASCII символа.

В этой части эксперимента система изменит передаваемые данные.

Что произошло с линией связи?

**Теория - 1**

Команды связи позволяет пользователю послать и получить данные через последовательный интерфейс RS-232 EB-153.

Эти команды управляют работой UART в 8031.

Последовательные данные ввода-вывода передаются или принимаются от микрокомпьютера EB-153 по кабелю связи.

Другой конец этого кабеля должен быть подключен к последовательному порту COM1/COM2 персонального компьютера.

 Используются только три сигнала RS-232:

* TxD (передаваемые данные)
* RxD (принимаемые данные)
* GROUND (общий провод)

 Сигнал TxD ПК (контакт 2) должен быть подключен к сигналу RxD микрокомпьютера EB-153.

Сигнал RxD ПК (контакт 3) должен быть подключен к сигналу TxD микрокомпьютера EB-153.

Общий провод (контакт 5) должн быть связан с общим проводом микрокомпьютера EB-153.

8031 обеспечивает подачу тактовых импульсов для своего внутреннего UART.

Системная программа задает скорость передачи, равную 2400 бод.

**Эксперимент - 1**

Присоедините коммуникационный кабель.

Наберите команду UPLOAD на клавиатуре микрокомпьютера EB-153.

Вставьте коммуникационную дискету в дисковод, если программа передачи данных не была скопирована на жесткий диск компьютера.

Нажмите на иконку Приложение в правом нижнем углу.

На экране компьютера появиться программа, которая предложит пользователю выполнить все необходимые последующие действия (задать имя файла, адрес и т.д.).Адреса указываются с помощью программы передачи данных, установленной на ПК.

 Скопируйте блок управляющей программы EB-153 в файл на жестком диске ПК.

Наберите имя файла и нажмите OK.

Выберите стартовый адрес 0000H и конечный адрес 0200H. Нажмите Apply (применить).

Данные, находящиеся по адресу 0000H - 0200H будут переданы в файл на жестком диске ПК.

 Файл CN показывает файлы в шестнадцатеричном формате на экране ПК. Для вызова программы просто нажмите на иконку Application (Приложение) в правом нижнем углу экрана.

Команда загрузки копирует программу находящуюся в файле на жестком диске ПК в память микрокомпьютера EB-153. Формат команды: DOWN.

Адреса определяются с помощью программы передачи данных ПК.

 Чтобы загрузить программу, придерживайтесь следующего порядка действий:

**а)**Подключите кабель для последовательной связи.

**б)**Наберите команду DOWN на клавиатуре микрокомпьютера EB-153.

**в)**Вставьте коммуникационную дискету в дисковод, если программа передачи данных не была скопирована на жесткий диск ПК.

 Команда "DO" выполняет загрузку вашего двоичного файла в память EB-153, используя последовательную связь с ПК.

**г)**Нажмите иконку Application (Приложение) в правом нижнем углу.

**д)**На экране компьютера появиться программы, которая сообщит пользователю о всех необходимых последующих действиях (задание имени файла, адреса и т.д.).

Адреса определяются программой передачи данных, установленной на ПК.

Скопируйте файл расположенный на жестком диске в ОЗУ EB-153.Укажите имя файла, который Вы хотите загрузить в ОЗУ и нажмите ОК.

Выберите стартовый адрес E000 и конечный адрес E200. Нажмите Apply (Применить).

Данные из указанного файла будут переданы в ОЗУ.Просмотрите переданные данные с помощью команду CBYTE.

Выключите PU-2000.Отключите ЕВ-153 от PU-2000.

**Вопросы**

1. В чем разница между синхронной и асинхронной передачей?
	* Асинхронная передача данных добавляет старт бит и стоп бит к байту, и в ней нет синхронизующих сигналов.
	* Синхронная передача данных добавляет стартовый бит и стоп бит к байту, и ей необходимы синхронизующие сигналы.
	* Отличаются по скорости передачи.
2. Каковы логические уровни сигналов RS-232?
	* TTL уровни.
	* "1" = +24V, "0" = 0V или наоборот.
	* "1" = -12V, "0" = +12V.
3. Что такое скорость передачи информации?
	* Величина, которая показывает скорость последовательной передачи (биты в секунду).
	* Количество передаваемых байт (бит/секунда)
	* Число стоп битов, добавленных к байтам данных.

**Лабораторная работа №10.**

**(**время выполнения 4 часа**)**

**« Изучение использования клавиатуры и дисплея»**

Цель работы:

 1. Научиться писать программы для опроса клавиатуры.

2. Научиться программировать встроенный модуль ЖК-дисплея.

**Оборудование**

* Лабораторный стенд EB-2000
* Печатная плата EB-153
* Комплект соединительных кабелей DL-20

Подключите EB-153 к PU-2000.Включите PU-2000.

**Порядок работы**

***ИСПОЛЬЗОВАНИЕ КЛАВИАТУРЫ EB-153***

EB-153 имеет встроенную клавиатуру, которая позволяет пользователю вводить команды и данные.

Клавиши клавиатуры составляют матрицу. Ее ряды и колонки подключены к программируемому периферийному интерфейсу ППИ, как показано на следующем рисунке.

******

***Рис. 1. Подключение клавиатуры***

Системная программа выполняет следующие задачи:

* + 1. Генерирует импульсы сканирования рядов, которые подаются на ряды клавиатуры.
		2. Считывает состояние колонок клавиатуры и определяет нажатые клавиши.
		3. Устраняет эффект дребезга контактов нажатой кнопки.
		4. Преобразует скан-коды в АSCII символы.

Нажатая клавиша определяется следующим образом:

1. На одну из сигнальных линий колонок РВ подается "0", все остальные установлены в "1".
2. Сигнальные линии рядов PA - это линии возврата сигнала. Если нажата клавиша, на соответствующей сигнальной линии ряда, считывается "0".
3. Сканируя все колонки программа может определить нажатую клавишу.

Следующая программа определяет нажатую клавишу и выдает на экран номера ряда и колонки нажатой клавиши. Пользователь может присвоить любое ASCII значение полученным скан-кодам.

**Программа клавиатуры**

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H LCALLFF80H очищает дисплей

E003H MOVDPTR, #0003H устанавливает адрес Режима ППИ

E006H MOVA, #99H определяет выходы PA и PB

E008H MOVX @DPTR, A устанавливает Режим ППИ

E009H MOVR0, #00H устанавливает в исходное положение

 указатели ряда и колонки

E00BH MOVR1, #FEH устанавливает в исходное состояние

 активную колонку

E00DH MOVDPTR, #0001H устанавливает адрес PB

E010H MOVA, R1 считывает данные колонки PB

E011H MOVX @DPTR, A устанавливает активную колонку

E012H MOVDPTR, #0000H устанавливает адрес PA

E015H MOVXA, @DPTR читает данные ряда

E016H CPLA инвертирует данные ряда

E017H JNZ 0CH определяет нажатую клавишу

E019H INCR0 увеличивает значение указателя

 колонки

E01AH MOVA, R1 получает новый указатель

E01BH RLA циклически сдвигает указатель

 колонки

E01CH MOVR1, A устанавливает новую активную

 колонку

E01DH MOVA, R0 читает значение указателя

E01EH ADDA, #F8H проверяет, последняя ли колонка

E020H JZE7H Если да, возвращается в начальное

 состояние

E022H LJMPE00DH если нет, проверяет следующую

 колонку

E025H MOVR2, #00H программа нажатой клавиши

E027H RRCA ряд определен

E028H JC 04H проверяет бит переноса

E02AH INCR2 следующий ряд

E02BH LJMPE027H продолжает

(Продолжение программы)

АДРЕС КОМАНДА КОММЕНТАРИЙ

E02EH MOVA, R0 получает номер ряда

E02FH ADDA, #30H преобразует в ASCII

E031H MOVDPTR, #FF93H указатель дисплея

E034H MOVX @DPTR, A отображает на дисплее

E035H MOVA, R0 получает номер колонки

E036H ADDA, #30H преобразует в ASCII

E038H MOVDPTR, #FFA3H указатель дисплея

E03BH MOVX @DPTR, A отображает на дисплее

E03CH MOVDPTR, #FF90H отображает ряд

E03FH MOVA, #52H 'R'

E041H MOVX @DPTR, A определяет изображение

E042H INCDPTR следующее изображение

E043H MOVA, #3DH '='

E045H MOVX @DPTR, A определяетизображение

E046H MOVDPTR, #FFA0H отображаетколонку

E049H MOVA, #43H 'C'

E04BH MOVX @DPTR, A определяетизображение

E04CH INCDPTR следующееизображение

E04DH MOVA, #3DH '='

E04FH MOVX @DPTR, A определяет изображение

E050H LCALLFF83H отображает ряд и колонку

E053H LJMPE009H возвращается к началу цикла

1. Выполните программу.

Нажимайте на разные клавиши и наблюдайте за номерами ряда и колонки отображемыми на экране.

Сравните их с фактическим подключением клавиатуры, показанным на следующем рисунке.



***Подключение клавиатуры***

***ИСПОЛЬЗОВАНИЕ ДИСПЛЕЯ EB-153***

1. ЖК матричный модуль, входящий в состав EB-153 позволяет отобразить 32 символа. Этот модуль имеет свой собственный контроллер и принимает от внешнего микропроцессора информацию о том, что необходимо отобразить и на какой позиции. Пользователь может прочитать или записать Управляющую Функцию (адрес 1001H и 1002H) или прочитать или записать Данные (адреса 1001Н - 1003Н).

Набор команд используемый для работы ЖК дисплея приводится в следующей таблице. Следующая программа показывает, как использовать некоторые функции ЖК дисплея. Эта программа очищает дисплей и посылает ЖК модулю команды, чтобы отобразить весь набор символов.

***ЖК КОМАНДЫ***



записать данные

считать данные

**Очистка дисплея**

**Исходная позиция курсора**

**Установка режима ввода**

**Управление вкл/выкл дисплея**

**Сдвиг курсора**

**Установка функции**

**Установка адреса CG RAM**

**Установка адреса DD RAM**

**Чтение адреса/флага готовности**

**Запись в CG RAM/DD RAM**

**Чтение из CG RAM/DD RAM**

I/D=1 увеличение на 1 I/D=0 уменьшение на 1

S=1 сдвиг индикации S=0 фиксирование индикации

D=1 включить дисплей D=0 выключить дисплей

C=1 индикация курсора C=0 выключить индикацию курсора

B=1 символ на позиции курсора мигает B=0 символ на позиции курсора не мигает

S/C=1 сдвиг индикации S/C=0 движение курсора

R/L=1 сдвиг вправо R/L=0 сдвиг влево

DL=1 8 бит DL=0 4 бита

BF=1 в течении внутренней операции BF=0 окончание внутренней операции

6.Введите следующую программу.

АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H MOVDPTR, #1000H определяет адрес команды

E003H MOVA, #38H устанавливает длину данных 8 бит

E005H MOVX, @DPTR, A записывает команду

E006H LCALLE200H задержка

E009H MOVDPTR, #1000H определяет адрес команды

E00CH MOVA, #01H очищает дисплей

E00EH MOVX @DPTR, A записывает команду

E00FH LCALLE200H задержка

E012H MOVDPTR, #1000H определяет адрес команды

E015H MOVA, #06H устанавливает режим – увеличивает

 адрес на 1

E017H MOVX @DPTR, A записывает команду

E018H LCALLE200H задержка

E01BH MOVDPTR, #1000H определяет адрес команды

(Программа – продолжение)

АДРЕС КОМАНДА КОММЕНТАРИЙ

E01EH MOVA, #0CH дисплей включен - курсор выключен

E020H MOVX @DPTR, A записывает команду

E021H LCALLE200H задержка

E024H MOVDPTR, #E100H устанавливает данные дисплея

E027H MOVXA, @DPTR считывает значение счета

E028H MOVR7,A записывает A в R7

E029H ANLA,#E0H проверяет данные в A

E02BH JNZ 02H переходит, если А больше чем 20H

E02DH MOVR7,#20H записывает 20H в R7

E02FH MOVA,R7 перемещает значение R7 в A

E030H INCA увеличивает счет на 1

E031H ANLA,#7FH ограничивает счет

E033H MOVX @DPTR, A Сохраняет счет

E034H DPTR, #1001H Записывает указатель данных

E037H MOVX @DPTR, A Отображает счет

E038H LCALLE200H Задержка

E03BH LJMPE024H Возвращается в начало цикла

E200H MOVDPTR, #FFC0H Устанавливает указатель на

 задержку программы

E203H MOVA, #01H Определяет постоянную задержки

E205H MOVX @DPTR, A Записывает постоянную

E206H LCALLFF8CH Подпрограмма задержки

E209H RETI Возвращается в программу

7.Программа задержки (адрес E200H) упрощает программу и позволяет обойтись без проверки флага готовности дисплея.

Выполните программу и проследите за символами, появляющимися на ЖК дисплее. Вы можете сравнить их с таблицей символов ЖК дисплея.

Заметьте, что задержка между отображением строк обусловлена загрузкой информации во внутреннее ОЗУ ЖК.



***Таблица знакогенератора ЖК дисплея***

**Вопросы**

1. Сколько контактов порта необходимо, чтобы сканировать матрицу из 64 клавиш?
	* 8
	* 12
	* 16
	* 64
2. Почему на схеме подключения клавиатуры, выходы Порта А соединены с шиной питания через резисторы?
	* Они устанавливают все невыбранные входы в "0".
	* Они устанавливают все невыбранные входы в "1".

Они устанавливают все выходные линии в «1»

**Лабораторная работа №11.**

**(**время выполнения 4 часа**)**

**«Определение поврежденного участка цепи с помощью измерительных приборов»**

Цель работы:

1.Научиться отыскивать неисправности при обслуживании оборудования

2.Исследовать неисправные системы

 3.Проверка предполагаемого поврежденного участка цепи для нахождения неисправных элементов.

**Теория**

В режиме теста, PU-2000 автоматически имитирует одну из четырех неисправностей в схеме.

У вас есть три попытки и 20 минут, чтобы определить местоположение каждой неисправности.

Максимальное количество баллов 24 (за каждый ответ), Вы получите, если ответите правильно с первой попытки.

Если Вы отвечаете правильно со второй попытки, Вы получаете 16 баллов и с третьей попытки - 8 баллов.

Вы получите еще 4 балла, если определите все ошибки менее чем за 16 минут.

Если вы не определите ошибки после трех попыток или не уложитесь в 20 минут, на экране на несколько секунд высветится правильный ответ.

После этого начнется новый отсчет времени для поиска следующей ошибки.

Для определения поврежденного участка проверяйте выходы участков схемы.

Определив поврежденный участок схемы, найдите поврежденный элемент.

**Оборудование**

* Лабораторный стенд PU-2000
* Печатная плата EB-153
* Осциллограф

Подключите EB-153 к PU-2000.Включите стенд PU-2000.

Перед включением режима ТЕSТ, установите переключатели 0 - 3 в положение"ON".

**Порядок работы**

Следующая программа считывает логические состояния переключателей, подключенных к Порту 1 8031 и передает эти значения Порту С ППИ и светодиодам: бит7, бит 6, бит 5 и бит 4.

Введите программу, представленную далее.

 АДРЕС КОМАНДА КОММЕНТАРИЙ

E000H MOVDPTR,#0003H устанавливает указатель данных на

 адрес режима ППИ

E003H MOVA,#80H определяет Порт С как выходной

E005H MOVX @DPTR,A

E006H MOVDPTR,#0002H устанавливает указатель данных

 ППИ-Порт C

E009H MOVA,90H считывает значение Порта 1

E00BH MOVX @DPTR,A передает Порту С

E00CH MOVC,93H читает бит 3 и передает биту 7

E00EH MOV 97H,C

E010H MOVC,92H читает бит 2 и передает биту 6

E012H MOV 96H,C

E014H MOVC,91H читает бит 1 и передает биту 5

E016H MOV 95H,C

E018H MOVC,90H читает бит 0 и передает биту 4

E01AH MOV 94H,C

E01CH LJMPE000H переходит к начальному адресу

Выполните эту программу.

Эта программа активирует Порт С и Порт 1 и позволяет вам найти ошибки в Порте С и Порте 1 (другие ошибки могут проявиться в разных частях печатной платы).

Измените положение переключателей подключенных к Порту 1 и пронаблюдайте за сигналом с помощью осциллографа.

На следующем экране вы можете выбрать тип ошибки.

|  |  |
| --- | --- |
| **ЭЛЕМЕНТ** | **ОПИСАНИЕ НЕИСПРАВНОСТИ** |
| Порт 1 -Бит 0 | Переключатель отключен |
| Порт 1 -Бит 1 | Переключатель отключен |
| Порт 1 -Бит 2 | Переключатель отключен |
| Порт 1 -Бит 3 | Переключатель отключен |
| Порт 1 -Бит 4 | Светодиод отключен или подключен к "0" |
| Порт 1 -Бит 5 | Светодиод отключен или подключен к "0" |
| Порт 1 -Бит 6 | Светодиод отключен или подключен к "0" |
| Порт 1 -Бит 7 | Светодиод отключен или подключен к "0" |
| кнопка INTR | Активируется самопроизвольно |
| кнопка INTR | Отключен |
| Порт C -Бит 0 | Отключен |
| Порт C -Бит 1 | Отключен |
| Порт C -Бит 2 | Отключен |
| Порт C -Бит 3 | Отключен |
| Порт C -Бит 4 | Отключен |
| Порт C -Бит 5 | Отключен |
| Порт C -Бит 6 | Отключен |
| Порт C -Бит 7 | Отключен |
| TxD | Подключен к "0" вместо "-8V" |
| RxD | Подключен к "0" |
| Порт C -Бит 0 | Закорочен на "землю" |
| Порт C -Бит 1 | Закорочен на "землю" |
| Порт C -Бит 2 | Закорочен на "землю" |
| Порт C -Бит 3 | Закорочен на "землю" |
| Порт C -Бит 6 | Закорочен на "землю" |
| Порт C -Бит 7 | Закорочен на "землю" |

Нажмите кнопку RESET, чтобы остановить программу.

Выключите лабораторный стенд.