Министерство образования Республики Беларусь

Учреждение образования
«Гомельский государственный технический
университет имени П. О. Сухого»

Кафедра «Промышленная электроника»

МИКРОПРОЦЕССОРНАЯ ТЕХНИКА

ПРАКТИКУМ
по выполнению лабораторных работ для студентов специальности 1-36 04 02 «Промышленная электроника» заочной формы обучения

Электронный аналог печатного издания

Гомель 2022

УДК 681(075.8)
ББК 32.81я73
М59

Рекомендовано научно-методическим советом заочного факультета ГГТУ им. П. О. Сухого
(*протокол № 2 от 02.12.2021 г.*)

Составитель *Э. М. Виноградов*

Рецензент: доц. каф. «Информационные технологии» ГГТУ им. П. О. Сухого
канд. техн. наук, доц. *В. С. Захаренко*

М59 **Микропроцессорная** техника : практикум по выполнению лаборатор. работ для студентов специальности 1-36 04 02 «Промышленная электроника» заоч. формы обучения / сост. Э. М. Виноградов. – Гомель : ГГТУ им. П. О. Сухого, 2022. – 65 с. – Систем. требования: PC не ниже Intel Celeron 300 МГц ; 32 Mb RAM ; свободное место на HDD 16 Mb ; Windows 98 и выше ; Adobe Acrobat Reader. – Режим доступа: https://elib.gstu.by. – Загл. с титул. экрана.

ISBN 978-985-535-490-2.

Содержит пять лабораторных работ с порядком их выполнения, основными теоретическими сведениями, заданиями для самостоятельной работы и контрольными вопросами.

Для студентов специальности 1-36 04 02 «Промышленная электроника» заочной формы обучения.

УДК 681(075.8)
ББК 32.81я73

ISBN 978-985-535-490-2

Лабораторная работа № 1

Интегрированная среда разработки mikroC Pro для PIC-микроконтроллеров

1 Цель работы

Изучить и практически исследовать методы разработки и отладки программ на языке Си для PIC-микроконтроллеров с помощью интегрированной среды разработки mikroC PRO for PIC.

2 Основные теоретические сведения

Для создания программного обеспечения микроконтроллерных систем широко используются средства вычислительной техники, в том числе персональные компьютеры, позволяющие разработчику выполнить весь цикл проектирования, включая отладку целевой программы.

В настоящее время самым мощным средством разработки программного обеспечения для микроконтроллеров являются интегрированные среды разработки (IDE – Integrated Development Environment), имеющие в своем составе текстовый редактор, компилятор языков высокого уровня типа Паскаль или Си, отладчик-симулятор, а также библиотеки готовых функций.

Одним из таких программных инструментов является среда разработки mikroC PRO компании MikroElektronika. Данная среда разработки позволяет быстро создавать эффективные программы на распространенном языке высокого уровня Си. Среда имеет удобный интерфейс пользователя со встроенным редактором текста и мощным отладчиком программ. Встроенный мастер проектов позволяет в считанные минуты создать заготовку рабочей программы для любого микроконтроллера из целого семейства. Библиотека готовых функций обеспечивает программиста поддержкой для быстрого и безошибочного создания программы. Компания MikroElektronika создала среду разработки mikroC PRO для таких популярных микроконтроллеров, как семейство PIC компании Microchip, AVR компании Atmel и семейство MCS-51.

В лабораторных работах будет использоваться интегрированная среда разработки для PIC-микроконтроллеров – mikroC PRO for PIC.

С сайта www.mikroe.com компании MikroElektronika можно бесплатно скачать демонстрационную версию среды mikroC PRO for PIC, которая позволяет создавать программы с объемом исполняемого кода до 2 Кбайт.

Разработка программного обеспечения при использовании mikroC PRO состоит из следующих основных этапов:

1 Создание проекта.

2 Создание исходных файлов на языке mikroC.

3 Построение проекта.

4 Тестирование программы и ее отладка.

Среда mikroC PRO организует программное обеспечение в виде проектов, состоящих из одного файла проекта (файл с расширением *.mcppi) и одного или нескольких исходных файлов на языке mikroC (файлов с расширением *.c). Исходные файлы могут компилироваться только в том случае, если они включены в проект.

Файл проекта содержит:

• имя проекта;

• тип микроконтроллера;

• тактовую частоту его работы;

• слово конфигурации микроконтроллера;

• список исходных файлов для проекта;

• другие (вспомогательные) файлы.

3 Порядок выполнения работы

3.1 Создание папки для работы в mikroC PRO for PIC

3.1.1 Для выполнения лабораторных работ по дисциплине «Микропроцессорная техника» вам необходимо создать на компьютере свою рабочую папку. Сделать это можно следующим образом.

На панели TotCom выберите диск F. Затем выберите папку МРТ и раскройте ее. Внутри нее создайте новую папку (с помощью клавиши F7) с именем, соответствующим вашей фамилии (буквы обязательно латинские), например: Ivanov.

Примечание. При использовании в именах папок русских букв возможна неправильная работа исследуемых программ.

В дальнейшем вы будете записывать и хранить в вашей папке все файлы в процессе выполнения лабораторных работ. Полный путь к ней:

F:\MPT\Ivanov

3.1.2 При выполнении лабораторных работ вы будете создавать много различных файлов. Для того чтобы было легче ориентироваться в них, желательно для каждой лабораторной работы иметь свою отдельную папку. Допустим, что для выполнения лабораторной работы № 1 мы будем использовать папку с именем Lab1, которую необходимо предварительно создать.

С этой целью откройте вашу рабочую папку (с именем, соответствующим вашей фамилии) и создайте в ней (с помощью клавиши F7) новую папку с именем Lab1.

В дальнейшем вы будете записывать и хранить в этой папке все файлы в процессе выполнения лабораторной работы № 1. Полный путь к ней:

F:\MPT\Ivanov\Lab1

3.2 Запуск mikroC PRO for PIC

Интегрированная среда mikroC PRO for PIC запускается из стартового меню Windows, подобно остальным приложениям, либо, что удобнее, с помощью ярлычка с надписью “mikroC PRO for PIC” на рабочем столе компьютера.

После запуска программы на экране компьютера вы увидите рабочий стол среды mikroC PRO for PIC. В верхней части экрана находится строка заголовка, в которой выводится имя проекта. Под ней расположена строка главного меню. Главное меню содержит обширный набор команд для доступа к функциям mikroC PRO for PIC.

В центре экрана располагается окно Get Started с различными сервисными функциями для помощи начинающим пользователям. В левой части экрана расположены команды для создания нового проекта (**New Project…**), открытия существующего проекта (**Open Project…)** и открытия папки примеров (**Open Examples Folder…**). В нижней части экрана находится окно сообщений. Стартовую страницу можно открыть также с помощью команды меню **View > Start Page**.

Рассмотрим методику разработки программного обеспечения в интегрированной среде mikroC PRO for PIC на примерах простейших программ для микроконтроллера PIC16F84A.

3.3 Первая программа вывода данных в порт микроконтроллера

Вот текст первой программы:

```
/*****************************************************
 out.c – программа вывода данных в порт B
*****************************************************/
void  main( )
{
   TRISB = 0;              // настроить все линии порта B на вывод
   PORTB = 0xFF;           // вывод данных (все единицы) в порт B
   PORTB = 0x00;           // вывод данных (все нули) в порт B
}
```

3.3.1 *Создание нового проекта*. Процесс создания нового проекта в среде mikroC PRO очень прост. Подведите курсор мыши к строке команды для создания нового проекта **New Project…** и щелкните левой кнопкой.

На экране появится окно мастера создания нового проекта – New Project Wizard. Начальное окно содержит список действий, которые должны быть выполнены для создания нового проекта. Этот процесс разбит на несколько шагов (Steps), которые вы должны последовательно проделать.

Для продолжения щелкните по кнопке [Next].

Шаг 1 Установка настроек проекта

На этом шаге необходимо определить общую информацию для проекта: выбрать микроконтроллер, его тактовую частоту и, конечно, дать проекту название. Это очень важный этап, так как компилятор создает внутренние настройки на основе этой информации. По умолчанию мастер проекта предлагает определенные настройки, но в нашем случае их требуется изменить. С этой целью проделайте следующее:

1 Очистите строку Project Name и введите новое имя проекта. При выборе имени проекта желательно, чтобы оно имело какую-то смысловую нагрузку. В нашей первой программе будет выполняться вывод данных в порт микроконтроллера (МК), поэтому можно дать имя этому проекту как out (вывод). Напоминаем, что mikroC PRO использует только латинские буквы. Использование кириллицы **недопустимо**!

Итак, введите имя проекта – out (расширение *.mcppi набирать не нужно!). В строке Project Name должно быть:

Project Name: out

2 Для выбора папки хранения проекта щелкните по кнопке [Browse] – обзор. Откроется окно обзора папок. Последовательно выберите и раскройте нужные папки в соответствие с установленном в п. 3.1.2 пути f:\MPT\Ivanov\Lab1. Последней должна быть папка с именем Lab1. Щелкните по ее имени для открытия. В заключение щелкните по кнопке [OK] для подтверждения своего выбора и установке нового пути. В строке Project Folder должно появиться имя папки Lab1 и полный путь к ней:

Project Folder: f:\MPT\Ivanov\Lab1\

3 Введите новое имя для микроконтроллера – PIC16F84A. С этой целью щелкните по стрелке в правой части строки Device Name. В раскрывшемся списке PIC-микроконтроллеров выберите P16F84A и щелкните по этой строке. В строке Device Name должно появиться:

Device Name: P16F84A

4 И, наконец, необходимо установить в строке Device Clock тактовую частоту работы микроконтроллера. По умолчанию она равна 8 МГц. Для нашего первого проекта можно оставить это значение.

5 В заключение шага 1 щелкните по кнопке [Next] для продолжения.

Шаг 2 Добавление файлов

Этот шаг позволяет вам включить дополнительные файлы, которые будут необходимы для вашего проекта, например, некоторые заголовочные файлы. В нашем случае нет необходимости подключать какие-либо дополнительные файлы, поэтому просто щелкните по кнопке [Next] для продолжения.

Шаг 3 Подключение библиотек

Этот шаг позволяет вам определить, желаете ли вы включить все библиотеки в ваш проект или нет. Следует знать, что, даже если все библиотеки включены, они не требуют какой-либо дополнительной памяти, если они не используются в вашей программе. Главное достоинство опции подключения всех библиотек в том, что вы получаете возможность использования в вашей программе свыше 500 готовых

функций, поэтому следует оставить эту конфигурацию по умолчанию, однако убедитесь, что флажок на кнопке [Include All (Default)] установлен.

Щелкните по кнопке [Next] для продолжения.

Шаг 4 Окончание построения проекта

После того, как все необходимые настройки для проекта выполнены, заключительный шаг позволяет вам сделать дополнительные действия.

Имеется возможность разрешить автоматическое открытие окна редактирования проекта после завершения работы мастера проекта. С помощью этого окна можно изменять биты конфигурации микроконтроллера. Это окно также может быть открыто в любой момент разработки программы с помощью команды меню **Edit > Project**.

Следует отметить, что в результате предыдущих шагов мастером проекта были установлены для микроконтроллера PIC16F84A следующие биты конфигурации:

HS – высокочастотный генератор (тактовая частота от 4 до 20 МГц);

WDT off – сторожевой таймер WDT отключен;

Code Protection off – отключена защита кода от чтения, т. е. можно читать и записывать код в микроконтроллер.

Для выполнения лабораторных работ можно оставить эту конфигурацию МК по умолчанию.

Поэтому для окончания построения проекта просто щелкните по кнопке [Finish].

Результатом работы мастера проектов будет новый проект out.mcppi, имя которого вместе с указанием полного пути к нему появится в строке заголовка в верхней части рабочего стола среды mikroC PRO for PIC.

В центре рабочего стола откроется окно редактора кода с пустым исходным файлом для текста программы на языке mikroC:

```
void  main( ) {
}
```

Исходный файл по умолчанию имеет то же имя, что и проект, т. е. out.c. Следует отметить, что среда mikroC PRO не требует, чтобы исходный файл имел такое же название, как и у проекта, поэтому файл можно переименовать.

3.3.2 *Создание исходного файла*. Наберите в окне редактора кода текст программы вывода данных в порт B микроконтроллера. Текст комментария для экономии времени можно не набирать.

Замечание. При наборе текста программы обязательно делайте отступы (3–4 пробела), как это выполнено в примере. Отступы облегчают чтение программы:

```
/*****************************************************
 out.c – программа вывода данных в порт B
*****************************************************/
void  main( )
{
    TRISB = 0;              // настроить все линии порта B на вывод
    PORTB = 0xFF;           // вывод данных (все единицы) в порт B
    PORTB = 0x00;           // вывод данных (все нули) в порт B
}
```

После набора сохраните файл out.c, выполнив команду меню **File > Save**.

3.3.3 *Построение проекта*. Построение проекта в mikroC PRO – это процесс компиляции исходных файлов, их компоновки и создание hex-файла, предназначенного для загрузки в программную память микроконтроллера. Для построения проекта надо выполнить команду меню **Build > Build**. После процесса компиляции и компоновки в окне Messages (в нижней части экрана) выдается сообщение об итогах построения. Сообщение содержит информацию о количестве затраченных ресурсов памяти, наличии ошибок (Errors) и предупреждений (Warnings).

Возникающие ошибки (обычно синтаксические) не позволяют компилятору сгенерировать машинные коды программы на основе исходного текста, поэтому такие ошибки обязательно должны быть исправлены. При наличии ошибок выводится сообщение “Finished (with errors)”.

Предупреждения выдаются компилятором в том случае, когда он может сгенерировать машинные коды, но в процессе их формирования возникли обстоятельства, требующие внимания разработчика.

Если в результате построения проекта возникли сообщения об ошибках или предупреждения, то их перечень с указанием номера строки и типа ошибки отобразится (красным цветом) в окне Messages.

Место ошибки в исходном тексте можно легко найти, дважды щелкнув левой кнопкой мыши на строке соответствующего сообщения.

Если ошибок нет, то в окне Messages появится сообщение (голубого цвета) с текстом “Finished successfully”.

После успешной компиляции mikroC PRO генерирует выходные файлы, которые помещаются в папку с файлом проекта:

• ассемблерный файл (с расширением *.asm). Содержит команды Ассемблера, сгенерированные компилятором;

• файл листинга (с расширением *.lst). Это общая картина распределения памяти микроконтроллера: адреса команд, регистры, программы и метки;

• HEX-файл (с расширением *.hex). Это основной выходной файл, который используется для загрузки кодов программы в память микроконтроллера.

Выходные файлы по умолчанию имеют то же имя, что и файл проекта. Ассемблерный файл и листинг могут быть просмотрены с использованием команд меню **View**.

Задание. Выполните построение проекта out.mcppi с помощью команды меню **Build > Build**. По содержимому окна Messages ознакомьтесь с итогами построения проекта.

3.3.4 *Тестирование программы с помощью симулятора-отладчика*. Среда mikroC PRO имеет средства отладки, позволяющие проверить работоспособность программы и исправить при необходимости ошибки в исходном тексте. Отладку можно выполнять как с помощью внутреннего симулятора работы микроконтроллера, так и с помощью аппаратного отладчика, подключив к нему целевую микроконтроллерную систему. По умолчанию отладчик mikroC PRO настроен для работы в режиме симулятора.

Для запуска отладчика надо выполнить команду из меню **Run > Start Debugger**. В результате mikroC PRO перейдет в режим отладки (в нем текущая строка исходного кода обозначается зеленой стрелкой и по умолчанию выделяется синим цветом) и откроется окно наблюдения Watch Values, позволяющее отслеживать в ходе выполнения программы содержимое различных переменных и регистров. Значения обновляются в процессе симуляции работы программы в отладчике. Последние измененные элементы в окне наблюдения выделяются красным цветом.

Замечание. Если после запуска отладчика на экране будет отсутствовать окно наблюдения Watch Values, то его можно открыть с помощью команды меню **View > Debug Window > Watch Window**.

Для внесения в список окна наблюдения Watch Values той или иной переменной программы, значение которой необходимо отслеживать, ее нужно выбрать в раскрывающемся списке строки Select variable from list и нажать кнопку [Add].

После того как переменная добавлена в список окна Watch Values, можно настроить параметры отображения ее в столбце Value. Для этого можно воспользоваться одним из способов:

• дважды щелкнуть левой кнопкой мыши на соответствующей строке списка Watch Values в столбце Name или Address;

• щелкнуть левой кнопкой мыши в столбце Value и затем нажать расположенную справа кнопку […].

В любом случае на экране появится диалоговое окно Edit Value, позволяющее отредактировать значение переменной прямо в ходе выполнения программы или же изменить формат ее отображения.

Для выбора формата отображения необходимо установить флажок в соответствующем окошке:

• Dec – десятичный;
• Hex – шестнадцатеричный;
• Bin – двоичный;
• Float – с плавающей точкой;
• Char – символьный.

Установка флажка Signed означает активизацию отображения знака.

Для того чтобы подтвердить изменения, внесенные в значение или формат отображения переменной, в диалоговом окне Edit Value следует нажать кнопку [OK]. Нажатие кнопки [Cancel] отменяет все внесенные изменения.

Для удаления текущего элемента в списке Watch Values служит кнопка [Remove], а для полной очистки этого списка – кнопка [Remove All].

В режиме отладки открывается также окно хронометража Watch Clock. В нем отображается текущий счетчик Current Count командных (машинных) циклов и времени в микросекундах (us) от момента запуска отладчика. Секундомер (Stopwatch) измеряет время исполнения в количестве командных циклов и микросекундах от момента запуска отладчика и может быть обнулен в любой момент. Разность (Delta) представляет фактическое время выполнения участка программы от предыдущей точки останова до текущей, отображаемое в количестве командных циклов и микросекундах (при пошаговом исполнении отображается время выполнения одной строки кода программы на mikroC).

Также в окне Watch Clock отображается текущая тактовая частота микроконтроллера (Clock). Тактовую частоту в окне хронометража можно изменять, что приведет к пересчету времени в микросекундах. Однако это изменение не влияет на текущие установки проекта, где также задана тактовая частота микроконтроллера, а влияет только на расчет времени симуляции.

Управлять процессом отладки можно тремя способами:

1) с помощью пунктов меню **Run**;

2) при помощи «горячих клавиш» клавиатуры;

3) с использованием кнопок (значков) в окне Watch Values.

Практика показывает, что наиболее удобно для отладки использовать именно кнопки окна Watch Values.

Примечание. Чтобы узнать функцию кнопки (значка), нужно поместить курсор мыши на эту кнопку. На экране появится описание этой функции.

Название и функция основных кнопок управления отладкой (а также соответствующих клавиш клавиатуры) следующие:

Start Debugger (F9) – запуск отладчика;

Run/Pause Debugger (F6) – запуск/останов программы в непрерывном (автоматическом) режиме;

Stop Debugger (Ctrl + F2) – прекращение работы отладчика;

Step Into (F7) – шаг на одну строку программы;

Step Over (F8) – шаг через одну строку программы;

Step Out (Ctrl + F8) – шаг из функции;

Run To Cursor (F4) – выполнять до курсора;

Toggle Breakpoint (F5) – поставить/удалить точку останова.

Команда Step Into позволяет «шагать» по каждой строке исходного текста. При помощи ее можно войти в вызываемую функцию.

Команда Step Over позволяет «перешагнуть» через вызываемую функцию, не входя внутрь, а выполнив ее как единое целое.

Для завершения работы с отладчиком в любой момент следует выбрать из меню команду **Run > Stop Debugger**. Произойдет возврат в режим редактирования.

Задание. Выполните тестирование и отладку в mikroC PRO первой программы вывода в порт out.c. С этой целью перейдите в режим отладки с помощью команды меню **Run > Start Debugger**.

Чтобы проверить работу программы out.c, нужно следить за состоянием выводов порта B, которые отображаются переменной PORTB.

Для занесения в окно наблюдения переменной подведите курсор мыши к строке Select variable from list (Выбрать переменную из списка) и щелкните по стрелке в правой части. В раскрывшемся списке выберите строку PORTB и щелкните по ней. Для добавления выбранной переменной PORTB в список окна наблюдения щелкните по кнопке [Add]. Строка с переменной PORTB (желтого цвета) появится в окне наблюдения.

Выполните программу out.c в пошаговом режиме Step Over, используя соответствующую кнопку управления отладкой или клавишу F8. При каждом шаге синяя полоса будет перемещаться по тексту программы. Наблюдайте за значением переменной PORTB на каждом шаге.

Выполнение программы out.c прекращается при достижении правой фигурной скобки функции main( ). Для возможности нового выполнения программы нужно произвести сброс микроконтроллера. Это можно сделать путем повторного запуска отладчика командой **Run > Start Debugger** или щелчком по соответствующей кнопке управления отладкой в окне Watch Values.

Значения переменных в окне наблюдения по умолчанию отображаются в десятичном формате. В программе out.c значения PORTB записаны в шестнадцатеричном формате. Для изменения формата отображения щелкните два раза по строке PORTB. В раскрывшемся окне редактирования Edit Value PORTB поставьте флажок в окошке HEX. Затем щелкните по кнопке [OK] для подтверждения выбора и закрытия окна редактирования.

Выполните вновь программу out.c в пошаговом режиме Step Over. Убедитесь, что формат отображения PORTB стал шестнадцатеричным.

Для завершения работы с отладчиком нужно выполнить команду меню **Run > Stop Debugger**, при этом происходит возврат в режим редактирования.

В заключение закройте проект с помощью команды **Project > Close Project**.

3.4 Вторая программа вывода в порт

Во второй программе происходит бесконечный цикл вывода в порт B микроконтроллера, т. е. состояния выходов порта циклически переключаются.

Текст исходной программы:

```
/*****************************************************
 out2.c – вторая программа вывода данных в порт B
*****************************************************/
void main( )
{
     TRISB = 0;                  // настроить линии порта B на вывод
      while(1)                   // бесконечный цикл повторения
     {
         PORTB  = 0xFF;           // вывод единиц в порт B
         PORTB = 0x00;           // вывод нулей в порт B
      }
}
```

Задание. Для исследования программы создайте новый проект с именем out2 и поместите его в папку Lab1, микроконтроллер PIC16F84A, тактовая частота 8 МГц. Наберите текст программы в файле out2.c и сохраните его в папке. Выполните построение проекта, а затем перейдите в режим отладки.

Для тестирования программы занесите в окно наблюдения Watch Values имя порта PORTB из исходной программы. Установите HEX-формат отображения PORTB.

Выполните программу в пошаговом режиме Step Over. После нескольких шагов убедитесь, что состояния выходов порта B циклически переключаются.

Можно узнать время, затраченное на выполнение каждого оператора, в строке программы по секундомеру, который находится в окне хронометража Watch Clock. В строке Delta показывается количество командных циклов и время в микросекундах (us), затраченных на выполнение одной строки исходной программы.

Продолжайте выполнять программу в пошаговом режиме Step Over и наблюдайте за показаниями в окне Watch Clock. При каждом шаге показания секундомера Stopwatch будут увеличиваться. По значениям в строке Delta: следует, что оператор вывода в порт (строка программы) выполняется за время 1 мкс.

Выполните сброс МК с помощью команды **Run > Start Debugger** или щелчком по соответствующей кнопке управления в окне Watch Values.

Теперь запустите программу в автоматическом режиме с помощью кнопки управления Run/Pause Debugger или нажатием на клавишу F6. При работе программы в автоматическом режиме значения переменной PORTB не обновляются, а в окне хронометража идет отсчет времени от момента запуска.

Остановите выполнение программы повторным щелчком по кнопке управления Run/Pause Debugger или нажатием на клавишу F6. Зеленая стрелка слева от текста программы покажет номер строки, на которой произошел останов выполнения. Значение переменной PORTB (красного цвета) обновится. В окне Watch Clock будет отображаться общее время выполнения программы в автоматическом режиме от момента запуска.

Для завершения работы с отладчиком выполните команду меню **Run > Stop Debugger**, при этом произойдет возврат в режим редактирования.

В заключение закройте проект с помощью команды **Project > Close Project**.

3.5 Программа вывода в порт с временной задержкой

Из результатов исследования программы out2.c следует, что вывод в порт (одна строка программы) выполняется за 1 мкс. Очевидно, что если бы к порту были присоединены светодиоды, то их переключение было бы незаметно. Следовательно, необходимо ввести временную задержку между переключениями состояний выводов порта.

Рассмотрим программу, в которой между переключениями выходов порта B имеется временная задержка:

```
/*****************************************************
 out3.c – программа вывода данных в порт B с временной задержкой
*****************************************************/
void main( )
{
    TRISB = 0;
    while(1)
    {
        PORTB = 0xFF;
        Delay_ms(500);      // временная задержка на 500 мс
        PORTB = 0x00;
        Delay_ms(500);      // временная задержка на 500 мс
    }
}
```

В программе out3.c для реализации временной задержки используется встроенная функция компилятора mikroC PRO for PIC

Delay_ms(time_in_ms),

где параметр time_in_ms – требуемое время задержки в миллисекундах. Это целое число в пределах от 1 до 4294697295.

Задание. Для исследования программы создайте новый проект с именем out3 и поместите его в папку Lab1, микроконтроллер PIC16F84A, тактовая частота 8 МГц. Наберите текст программы в файле out3.c и сохраните его в папке. Выполните построение проекта, а затем перейдите в режим отладки.

Для тестирования программы занесите в окно наблюдения Watch Values имя порта PORTB из исходной программы.

Выполните программу в пошаговом режиме Step Over. По показаниям в строке Delta окна хронометража убедитесь, что время задержки равно заданному в программе и реализовано компилятором с высокой точностью.

Для завершения работы с отладчиком выполните команду меню **Run > Stop Debugger**, при этом произойдет возврат в режим редактирования.

В заключение закройте проект с помощью команды **Project > Close Project**.

4 Содержание отчета

Наименование и цель работы. Тексты исследуемых программ (комментарии в программах обязательны!).

Контрольные вопросы

1 Что такое IDE mikroC PRO for PIC?

2 Какие функции выполняет mikroC PRO for PIC?

3 Поясните процесс разработки программного обеспечения в mikroC PRO for PIC.

4 Что такое проект в mikroC PRO?

5 На каком этапе осуществляется выбор микроконтроллера?

6 Какие окна используются в симуляторе для наблюдения за ходом выполнения программы?

7 Как можно осуществить пошаговое выполнение программы?

Лабораторная работа № 2

Изучение и исследование среды разработки электронных устройств PROTEUS

1 Цель работы

Изучить основы работы со средой проектирования электронных устройств Proteus. Изучить и исследовать методику разработки микроконтроллерных устройств с помощью Proteus.

2 Основные теоретические сведения

Proteus (произносится *протеус*) – это пакет программ, объединяющий в себе две основные программы: ISIS – средство разработки и отладки в режиме реального времени электронных схем и ARES – средство разработки печатных плат. Разработчиком пакета Proteus является фирма Labcenter Electronics (Великобритания).

Отличие Proteus от аналогичных по назначению пакетов программ, например, Electronics Workbench Multisim, MicroCap заключается в развитой системе симуляции (интерактивной отладки в режимах реального времени и пошаговом) для различных семейств микроконтроллеров: MCS-51 (Intel), PIC (Microchip), AVR (Atmel) и др. Proteus имеет обширные библиотеки компонентов, в том числе периферийных устройств (светодиодные и ЖК-индикаторы, температурные датчики, часы реального времени), интерактивных элементов ввода-вывода (кнопки, переключатели, виртуальные порты) и виртуальных измерительных приборов, которые не всегда присутствуют в других подобных программах.

В данной лабораторной работе будет рассматриваться только программа ISIS.

3 Порядок выполнения работы

3.1 Запуск программы

Внимание! Перед запуском программы создайте на диске F в своей рабочей папке (с именем, соответствующим вашей фамилии) новую папку с именем Lab2, в которую вы будете помещать исследуемые программы. Полный путь к этой папке может быть, например, таким:

F:\MPT\Ivanov\Lab2

Запуск программы ISIS.exe выполняется с помощью ярлычка с надписью ISIS на рабочем столе компьютера. После запуска программы на экране появится основное окно.

Самое большое пространство отведено под окно редактирования Edit Window. Именно в нем происходят все основные процессы создания, редактирования и отладки схемы устройства.

Слева вверху расположено маленькое окно предварительного просмотра Overview Window. С его помощью можно перемещаться по окну редактирования.

Под окном предварительного просмотра находится Object Selector – список выбранных в данный момент компонентов, символов и других элементов. Выделенный в списке объект отображается в окне предварительного просмотра.

Все возможные функции и инструменты Proteus доступны через меню, расположенное в самом верху основного окна, через пиктограммы, находящиеся под меню и с левой стороны основного окна, а также через «горячие клавиши».

В самом низу основного окна расположены слева направо:

• кнопки вращения и разворота объекта вокруг своей оси;

• панель управления интерактивной симуляцией (выглядит как магнитофонная и функции такие же: Play – ПУСК, Step – ПОШАГОВЫЙ РЕЖИМ, Pause – ПАУЗА, Stop – СТОП);

• строка состояния (в ней отображаются ошибки, подсказки, текущее состояние процесса симуляции и т. д.;

• координаты курсора, отображаемые в дюймах.

3.2 Разработка проекта управления светодиодом от микроконтроллера

Первым таким проектом будет разработка простейшего микроконтроллерного устройства (МКУ), в котором МК управляет светодиодом, то есть включает или выключает его по определенной программе.

3.2.1 Создайте новый проект, используя команду меню **File > New Design**. На экране появится диалоговое окно Create New Design, предлагающее выбрать шаблон для создания нового проекта. Щелкните левой кнопкой мыши по варианту DEFAULT (проект с параметрами по умолчанию), а затем по кнопке [OK]. На экране появится основное окно Proteus с синим прямоугольником в окне редактирования, в котором вы будете создавать схему МКУ.

Принципиальная схема МКУ будет иметь вид, приведенный на рисунке 1.

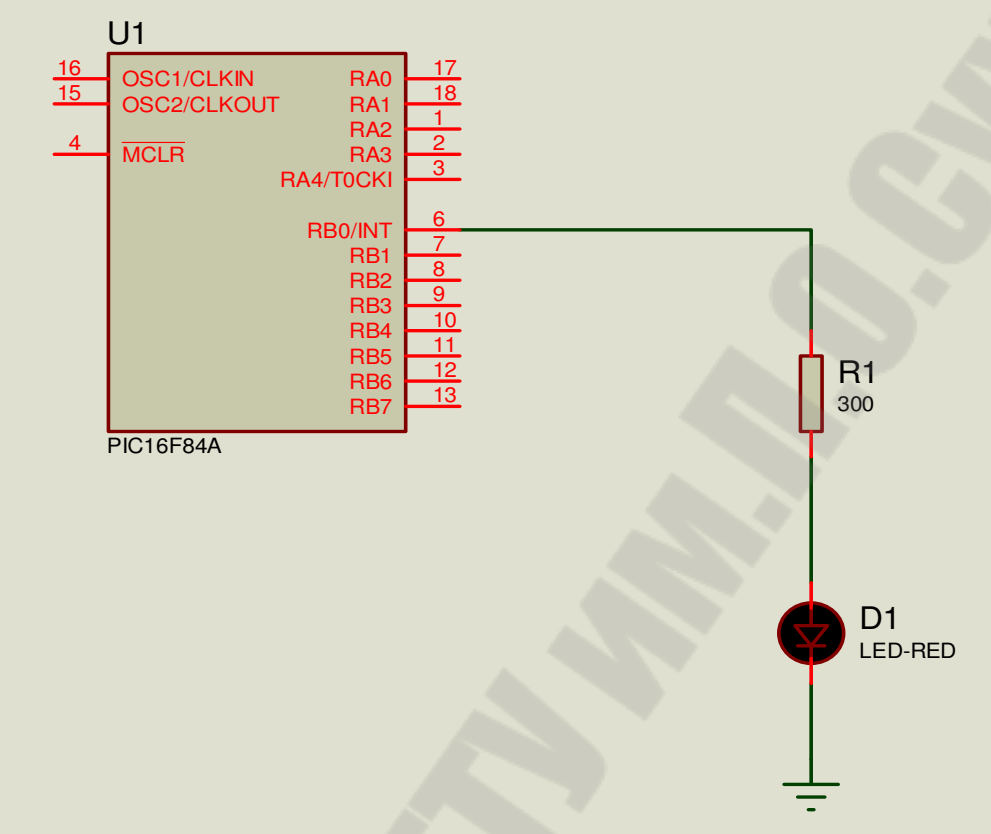

Рисунок 1 – Схема МКУ для управления светодиодом

В качестве микроконтроллера выбран PIC16F84A. На схеме не изображены цепи питания и сброса МК, а также задания частоты тактового генератора. Все это эмулируется программно, и нет необходимости добавлять их в схему. Однако, если вы будете разрабатывать проект до этапа изготовления печатной платы, то эти цепи и элементы придется добавить.

Итак, для построения схемы МКУ нам нужен микроконтроллер PIC16F84A, светодиод красного свечения и резистор с сопротивлением 300 Ом. Получить все эти элементы можно из библиотек компонентов, которые имеются в Proteus.

3.2.2 Выбор элементов из библиотек производится следующим образом. Необходимо перейти в режим компонентов Component Mode, щелкнув левой кнопкой мыши по соответствующей иконке на панели инструментов. Затем надо щелкнуть левой кнопкой мыши по кнопке [P] в верхнем левом углу переключателя объектов Object Selector, либо

дважды щелкнуть левой кнопкой мыши в поле Object Selector. На экране появится окно Pick Devices библиотеки компонентов.

Компоненты (элементы схем) можно выбирать по категориям Category, подкатегориям Subcategory, по производителю Manufacturer или же искать по ключевым словам Keywords. В случае, если вы знаете название элемента, то лучший способ – искать по нему. Итак, найдем микроконтроллер. Наберите в окне Keywords слово PIC16F84A. В окне результата Results появится строка с описанием выбранного компонента. Подведите курсор мыши к этой строке и щелкните левой кнопкой. Строка поменяет цвет на синий, что говорит о выделении данного компонента. Для перемещения выбранного компонента, в данном случае PIC16F84A, в список Object Selector нужно дважды щелкнуть левой кнопкой мыши по выделенной строке.

Для выбора следующего элемента схемы – светодиода – очистите строку Keywords и введите слово LED-RED (красный светодиод). В окне результата появится строка с описанием светодиода LED-RED из библиотеки ACTIVE. Щелкните левой кнопкой мыши по этой строке для выделения, а затем еще дважды для перемещения выбранного элемента в список Object Selector.

Для выбора следующего элемента схемы – резистора –очистите строку Keywords и введите слово RES (резистор). В окне результата появится список резисторов из библиотеки. Выберите строку RES DEVICE.

3.2.3 Теперь, когда элементы выбраны, необходимо перейти к следующему этапу разработки – их размещению в окне редактировании. Начнем с микроконтроллера. Щелкните левой кнопкой мыши по строке с текстом PIC16F84A. В окне предварительного просмотра появится изображение микроконтроллера. Если расположение выводов, то есть ориентация корпуса, соответствует исходной схеме, приведенной на рисунке 1, то переместите указатель мыши в середину окна редактирования и щелкните левой кнопкой. Контур микроконтроллера появится под указателем мыши и будет следовать за ним, когда мышь будет перемещаться по окну редактирования. Поместите контур в верхнюю часть окна и щелкните еще раз левой кнопкой мыши. После этого изображение микроконтроллера прорисуется полностью и будет помещено на схему.

Теперь щелкните левой кнопкой мыши по строке RES. Изображение резистора появится в окне предварительного просмотра. Если его ориентация не соответствует рисунку 1, то можно развернуть его

нажатием на иконки вращения панели инструментов. Затем надо поместить курсор мыши в окно редактирования и щелкнуть левой кнопкой. Перемещая мышь, надо разместить контур резистора в нужном месте схемы) и еще раз щелкнуть левой кнопкой.

Далее щелкните левой кнопкой мыши по строке LED-RED. Если ориентация светодиода будет соответствовать рисунку 1, то поместите его на схему в окне редактирования.

Не хватает еще одного важного элемента – «общего провода», или «корпуса». Элементы такого типа выбираются в режиме Terminals Mode, для чего требуется щелкнуть по иконке с соответствующей надписью на панели инструментов. После этого в окне Object Selector появится список доступных элементов. Точный русскоязычный эквивалент этих терминов в настоящее время не определился. Будем называть эти элементы клеммами. Выберите из списка элемент Ground («Земля», общий провод) и поместите его под светодиодом.

3.2.4 Теперь необходимо соединить элементы между собой согласно принципиальной схеме, приведенной на рисунке 1. Рассмотрим методику соединения или, по-другому, «разводки» на примере соединения вывода 6 (RB0/INT) микроконтроллера с выводом резистора R1.

Курсор мыши, при расположении его в произвольном участке схемы, имеет вид бесцветного карандаша. Подведите курсор к выводу 6 (RB0/INT) микроконтроллера. Цвет курсора изменится на зеленый, а на выводе появится красный квадратик, означающий, что соединение возможно. Щелкните левой кнопкой мыши. Курсор мыши изменится на бесцветный карандаш, чтобы показать, что происходит разводка, и предполагаемый путь будет следовать за движением мыши к концу проводника. При соприкосновении курсора с выводом резистора R1 цвет курсора вновь изменится на зеленый и появится красный квадратик. Это говорит о том, что соединение возможно. Щелкните левой кнопкой мыши. На схеме появится проводник, соединяющий вывод 6 микроконтроллера с верхним выводом резистора R1. По аналогии выполните соединения нижнего вывода резистора R1 с анодом светодиода, а затем катода светодиода с клеммой общего провода («Землей»).

Замечание. Соединение элементов схемы между собой должно быть выполнено по методике, описанной выше. Если же вы просто состыкуете выводы элементов путем их перемещения по экрану (например, нижний вывод резистора R1 и верхний вывод диода D1), то соединения не будет. Это будет разрыв цепи, хотя визуально выводы будут соединены вместе.

3.2.5 Теперь необходимо изменить значение сопротивления резистора на 300 Ом. С этой целью подведите курсор к изображению резистора R1 и дважды щелкните левой кнопкой мыши. Откроется окно редактирования свойств компонента. Введите в поле Resistance число 300 (сопротивление в омах), а затем нажмите кнопку [OK] для подтверждения выбора параметров. Итак, схема МКУ готова.

Теперь нужно сохранить проект в вашей папке. Выберите пункт меню **File > Save Design As…** Раскройте папку f:\...\Lab2 и сохраните в ней проект под именем led.dsn.

Сверните окно Proteus.

3.2.6 Следующим этапом проектирования МКУ является разработка программы работы микроконтроллера. Предположим, что после включения питания микроконтроллера (его сброса) светодиод будет мигать с частотой 1 Гц. Текст программы на языке mikroC может иметь следующий вид:

```
/*********************************************************
 led.c – программа управления светодиодом от микроконтроллера PIC16F84A
*********************************************************/
void main( )
{
    TRISB = 0;              // настроить выводы порта B на вывод
    PORTB = 0;              // вывести нули на выводы порта B
    while( 1 )              // бесконечный цикл
    {
        PORTB = 0xFF;       // зажечь светодиод
        Delay_ms(500);      // задержка на время 500 мс = 0,5 с
        PORTB = 0;          // погасить светодиод
        Delay_ms(500);
    }
}
```

Замечание. Несмотря на то что для переключения светодиода требуется периодически выводить логические 1 и 0 только на линию RB0 порта B, в программе led.c с целью ее упрощения выводятся нули и единицы на все линии порта. Методы управления отдельными линиями портов МК будут рассмотрены в дальнейших лабораторных работах.

Выполните разработку программы в следующей последовательности.

Запустите программу mikroC PRO for PIC. С помощью команды **New Project…** запустите New Project Wizard. Создайте новый проект с именем led в папке f:\ … \Lab2. Выберите микроконтроллер PIC16F84A и тактовую частоту 8 МГц. В окне редактора наберите текст исходной программы led.c и сохраните его в папке (с помощью команды меню **File > Save**). Выполните построение проекта, используя команду меню **Build > Build**. При отсутствии ошибок построения закройте проект командой **Project > Close Project**.

3.2.7 Теперь нужно проверить работу программы led.c в схеме МКУ с помощью Proteus. С этой целью разверните окно программы Proteus с проектом led.dsn.

3.2.8 Далее необходимо записать hex-код разработанной программы в память МК или, как часто говорят, запрограммировать МК. С этой целью подведите курсор мыши на изображение микроконтроллера и дважды щелкните левой кнопкой. Откроется окно редактирования свойств компонента Edit Component. В этом окне щелкните по кнопке в правой части строки Program File. Откроется окно Select File Name с hex-файлами из папки f:\ … \Lab2. Выберите файл с именем led.hex и нажмите кнопку [Открыть]. Затем в строке Processor Clock Frequency (тактовая частота процессора) выставьте 8 МГц. Нажмите кнопку [OK] для подтверждения выбора параметров. В заключение щелкните левой кнопкой мыши по пустому месту схемы для снятия выделения (красного цвета) с МК.

3.2.9 Для запуска программы на выполнение щелкните по кнопке [Play], расположенной слева в нижней части экрана. Светодиод должен попеременно загораться и гаснуть с интервалом 0,5 секунды.

Обратите внимание на маленькие квадратики около выводов МК, резистора и светодиода. Цвет этих квадратиков говорит об логических уровнях на выводах элементов в данный момент времени. Синий цвет – низкий, красный – высокий уровень.

Остановить выполнение программы можно с помощью кнопки [Stop].

В заключение сверните окно Proteus.

3.2.10 Задание 1 для самостоятельной работы. Необходимо разработать программу работы МКУ, изображенного на рисунке 1 и выполняющего следующий алгоритм. После запуска программы светодиод 4 раза загорается на 1 секунду и гаснет на 2 секунды. После четырех «миганий» светодиод окончательно гаснет. Тактовая частота МК равна

8 МГц. Цикл повторения организуйте с помощью оператора for( ). Программу назовите led2.c. В mikroC PRO for PIC создайте новый проект led2 и поместите его в папку Lab2. Получите файл led2.hex для загрузки в память микроконтроллера. Проверьте работу программы с помощью Proteus, открыв проект (схему МКУ) led.dsn из папки Lab2. Загрузите в микроконтроллер файл led2.hex из папки Lab2.

3.3 Разработка проекта управления от микроконтроллера семисегментным светодиодным индикатором

3.3.1 Теперь разработаем МКУ, в котором микроконтроллер выводит данные на семисегментный светодиодный индикатор. Принципиальная схема такого МКУ может иметь вид, приведенный на рисунке 2. В этой схеме используется индикатор с общими катодами. Сегменты индикатора подключены к выводам порта B, а общий вывод индикатора (катоды светодиодов) – к общему проводу схемы. Сегмент индикатора будет светиться при высоком уровне на соответствующем выводе порта B. На изображении индикатора выводы сегментов располагаются с левой стороны сверху вниз по порядку: *a*, *b*, *c*, *d*, *e*, *f*, *g*.

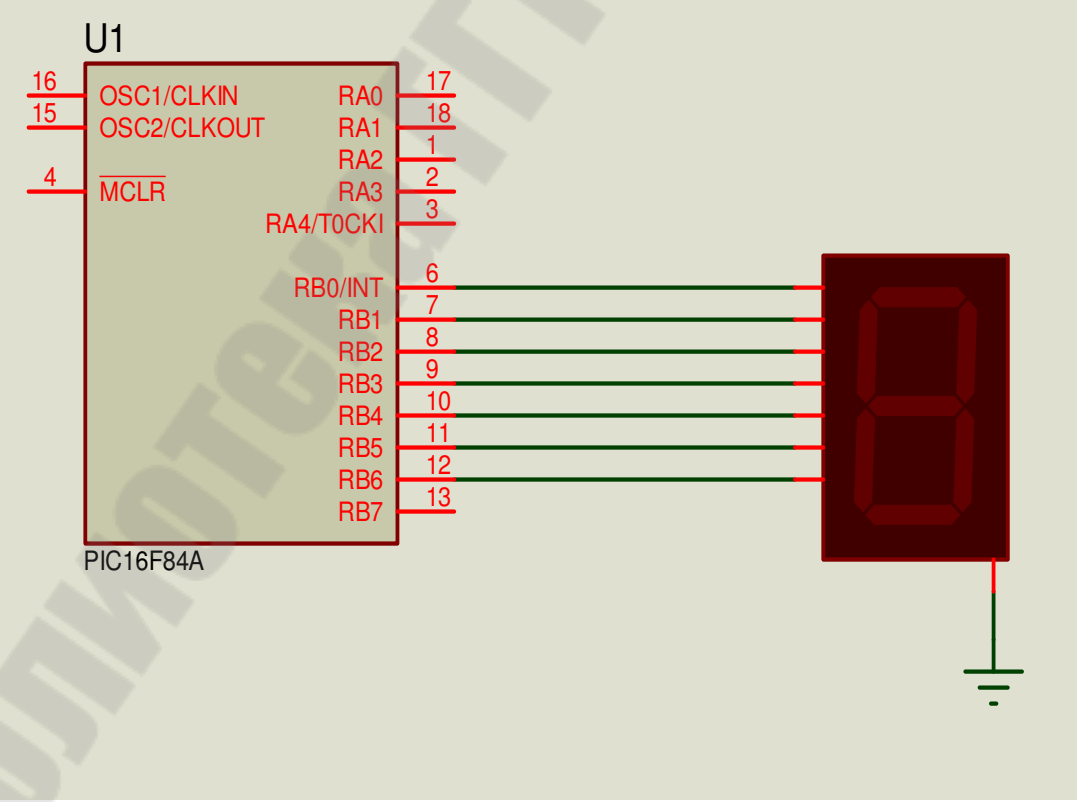

Рисунок 2 – Схема МКУ для управления индикатором

3.3.2 Создайте в Proteus новый проект, используя команду меню **File > New Design**.

3.3.3 Откройте библиотеку компонентов и выберите из нее микроконтроллер PIC16F84A.

Для поиска индикатора выберите в окне Category пункт Optoelectronics (оптоэлектронные приборы) и щелкните левой кнопкой мыши. Затем в окне Sub-category выберите строку 7-Segment Displays и также щелкните левой кнопкой. Выделите в окне Results строку

7SEG-COM-CATODE

Это 7-сегментный индикатор красного цвета свечения с общими катодами светодиодов. Щелкните два раза по строке, а затем закройте библиотеку.

3.3.4 Теперь разместите элементы МКУ в окне редактирования. Вначале поместите микроконтроллер. Для удобства размещения увеличьте масштаб отображения элементов. Затем перейдите в режим Terminals Mode и выберите из списка клемму Ground. Поместите ее ниже общего вывода индикатора.

3.3.5 Выполните соединение элементов между собой, согласно принципиальной схеме, приведенной на рисунке 2.

После завершения разводки необходимо сохранить проект. Для этого выполните команду меню **File > Save Design As…** Раскройте папку f:\...\Lab2 и сохраните в ней проект под именем ind.dsn.

3.3.6 Следующим этапом проектирования МКУ является разработка программы для МК. Пусть после включения питания на индикатор постоянно выводится определенный код символа, например, цифры 6.

Текст программы на языке mikroC, реализующий заданный алгоритм, может иметь следующий вид:

```
/*****************************************************
ind.c – программа вывода на индикатор цифры 6
*****************************************************/
char table[ ] =          // массив семисегментных кодов для индикатора
{
    0x3F,                 // семисегментный код цифры 0
    0x06,                 // семисегментный код цифры 1
    0x5B,                 // семисегментный код цифры 2
    0x4F,                 // семисегментный код цифры 3
```

```
    0x66,                   // семисегментный код цифры 4
    0x6D,                   //  семисегментный код цифры 5
    0x7D,                   //  семисегментный код цифры 6
    0x07,                   //  семисегментный код цифры 7
    0x7F,                   //  семисегментный код цифры 8
    0x6F                    //  семисегментный код цифры 9
};
void main( )
{
    char j;             // переменная-индекс элементов массива table[ ]
    TRISB = 0;              // настроить линии порта B на вывод
    PORTB = 0;              // погасить индикатор
    j = 6;
    PORTB = table[j];   // вывод на индикатор семисегментного кода
                            // цифры 6
}
```

3.3.7 Выполните разработку программы в следующей последовательности.

Запустите программу mikroC PRO for PIC. С помощью команды **New Project…** запустите New Project Wizard. Создайте новый проект с именем ind в папке f:\ … \Lab2. Выберите микроконтроллер PIC16F84A и тактовую частоту 8 МГц. В окне редактора наберите текст исходной программы ind.c и сохраните его в папке. Получите файл ind.hex для загрузки в память микроконтроллера.

3.3.8 Теперь нужно проверить работу программы ind.c в схеме МКУ с помощью Proteus. С этой целью разверните окно программы Proteus с проектом ind.dsn.

3.3.9 Далее необходимо записать hex-код разработанной программы в память МК. С этой целью подведите курсор мыши на изображение микроконтроллера и дважды щелкните левой кнопкой. Откроется окно редактирования свойств компонента Edit Component. В этом окне щелкните по кнопке в правой части строки Program File. Откроется окно Select File Name с hex-файлами из папки f:\ … \Lab2. Выберите файл с именем ind.hex и нажмите кнопку [Открыть]. Затем в строке Processor Clock Frequency (тактовая частота процессора) выставьте 8 МГц. Нажмите кнопку [OK] для подтверждения выбора параметров. В заключение щелкните мышью по пустому месту схемы для снятия выделения (красного цвета) с МК.

3.3.10 Для запуска программы на выполнение щелкните по кнопке [Play], расположенной слева в нижней части экрана. Если на индикаторе появилась цифра 6, то все в порядке.

Остановить выполнение программы можно с помощью кнопки [Stop].

3.3.11 Задание 2 для самостоятельной работы. Используя схему МКУ, приведенную на рисунке 3, проверьте работу программы, выполняющую следующий алгоритм. После включения питания микроконтроллера на индикатор выводится цифра 5, а затем она начинает мигать с частотой 1 Гц (0,8 секунды светится и 0,2 секунды нет).

Программу назовите ind2.c. В mikroC PRO for PIC создайте новый проект ind2 и поместите его в папку Lab2. Получите файл ind2.hex для загрузки в память микроконтроллера. Проверьте работу программы с помощью Proteus, открыв проект (схему МКУ) ind.dsn из папки Lab2. Загрузите в микроконтроллер файл ind2.hex из папки Lab2.

4 Содержание отчета

Наименование и цель работы. Схемы МКУ и тексты всех программ к заданиям для самостоятельной работы (комментарии в программах обязательны!).

Контрольные вопросы

1 Какие функции выполняет программный пакет Proteus?

2 Из каких частей состоит пакет Proteus?

3 Как выбираются компоненты принципиальной схемы в Proteus?

4 Каким образом можно перемещать схему по экрану дисплея?

5 Как можно удалить из схемы какой-либо компонент?

6 Каким образом можно изменить величину сопротивления резистора на схеме?

7 Как можно загрузить программу в микроконтроллер в среде Proteus?

Лабораторная работа № 3

Разработка и исследование управляющих программ для PIC-микроконтроллеров

1 Цель работы

Изучить основы разработки и отладки управляющих программ на языке mikroC для микроконтроллеров семейства PIC16 с помощью интегрированной среды mikroC PRO for PIC и пакета схемотехнического моделирования Proteus.

2 Основные теоретические сведения

В программах для микроконтроллеров часто требуется управлять отдельными линиями портов МК, а также опрашивать их состояния.

Язык mikroC позволяет иметь доступ к отдельными битами регистров специальных функций. Это можно выполнить следующими способами:

1 Указать имя регистра, а затем после знака точки как разделителя идентификатор номера бита в регистре в виде: B0, B1, B2, B3, B4, B5, B6, B7, где B0 – младший бит.

Например:

```
// для порта C, настроенного на вывод
PORTC.B0 = 1;       // вывести лог. 1 на 0-ю линию порта C
PORTC.B7 = 0;       // вывести лог. 0 на 7-ю линию порта C

INTCON.B7 = 1;      // Установить в 1 бит GIE разрешения глобальных
                    // прерываний  (7-й бит в регистре INTCON)
```

Для управления отдельными битами регистров портов можно использовать поразрядные операции, например:

```
PORTC.B1 &= 0;     // вывести лог.0 на 1-ю линию порта C
PORTC.B2 |= 1;     // вывести лог.1 на 2-ю линию порта C
PORTC.B4 ^= 1;     // инвертировать 4-й разряд порта C
PORTC.B7 = ~PORTC.B7;  // инвертировать 7-й разряд порта C
```

2 Указать имя бита в регистре специальных функций, а затем идентификатор вида: _bit.

Например:

```
GIE_bit = 1;                // установить в 1 бит GIE в регистре INTCON
```

Биты регистров данных (линий портов) обозначаются как:

RA0, RA1, RA3 и т.п. для регистра PORTA;
RB0, … , RB7 для регистра PORTB;
RC0, … , RC7 для регистра PORTC.

```
// для порта С, настроенного на вывод
RC0_bit = 1;                // вывести лог. 1 на 0-ю линию порта С
RC7_bit = 0;                // вывести лог. 0 на 7-ю линию порта С
```

Для управления отдельными битами портов можно также использовать поразрядные операции, например:

```
RC1_bit &= 0;     // вывести лог.0 на 1-ю линию порта С
RC2_bit |= 1;     // вывести лог.1 на 2-ю линию порта С
RC4_bit ^= 1;     // инвертировать 4-й разряд порта С
RC7_bit = ~RC7_bit;  // инвертировать 7-й разряд порта С
```

Биты регистров направления портов обозначаются как:

TRISA0_bit, TRISA1_bit и т.п. для регистра TRISA;
TRISB0_bit, … , TRISB7_bit для регистра TRISB;
TRISC0_bit, … , TRISC7_bit для регистра TRISC.

В управляющих программах для микроконтроллеров часто возникает необходимость в формировании временных задержек. Компилятор mikroC имеет так называемые встроенные функции задержки, которые формируют временные задержки программным методом. Наибольшее применение находят три функции:

1 Delay_us(константа), где константа определяет величину задержки в микросекундах (константа – это целое число типа unsigned long в пределах от 1 до 4294967295);

2 Delay_ms(константа), где константа определяет величину задержки в миллисекундах (константа – это целое число типа unsigned long в пределах от 1 до 4294967295);

3 Vdelay_ms(переменная), где значение переменной определяет величину задержки в миллисекундах (переменная должна быть типа unsigned int, а величина ее в пределах от 1 до 65535).

Примеры:

```
Delay_us(100);                  // задержка на время 100 мкс
Delay_ms(2000);                 // задержка на время 2000 мс = 2 с
unsigned  int  time = 500;
Vdelay_ms(time);                // задержка на время 500 мс
```

3 Порядок выполнения работы

3.1 Создание папки для работы в mikroC PRO for PIC

При выполнении лабораторной работы № 3 мы будем использовать папку с именем Lab3, которую необходимо предварительно создать.

С этой целью откройте вашу рабочую папку и создайте в ней (с помощью клавиши F7) новую папку с именем Lab3.

В дальнейшем вы будете записывать и хранить в этой папке все файлы при выполнении лабораторной работы № 3. Полный путь к этой папке будет такой:

F:\MPT\Ivanov\Lab3

3.2 Исследование программ управления светодиодом

3.2.1 Рассмотрим простейшее микроконтроллерное устройство (МКУ), в котором МК типа PIC16F877 управляет светодиодом, присоединенным к линии (разряду) RC0 порта С. Принципиальная схема такого МКУ (при моделировании в среде Proteus) может иметь вид, приведенный на рисунке 1. При высоком уровне (логическая 1) на выводе порта RC0 светодиод горит, а при низком уровне (логический 0) на выводе порта RC0 светодиод не горит.

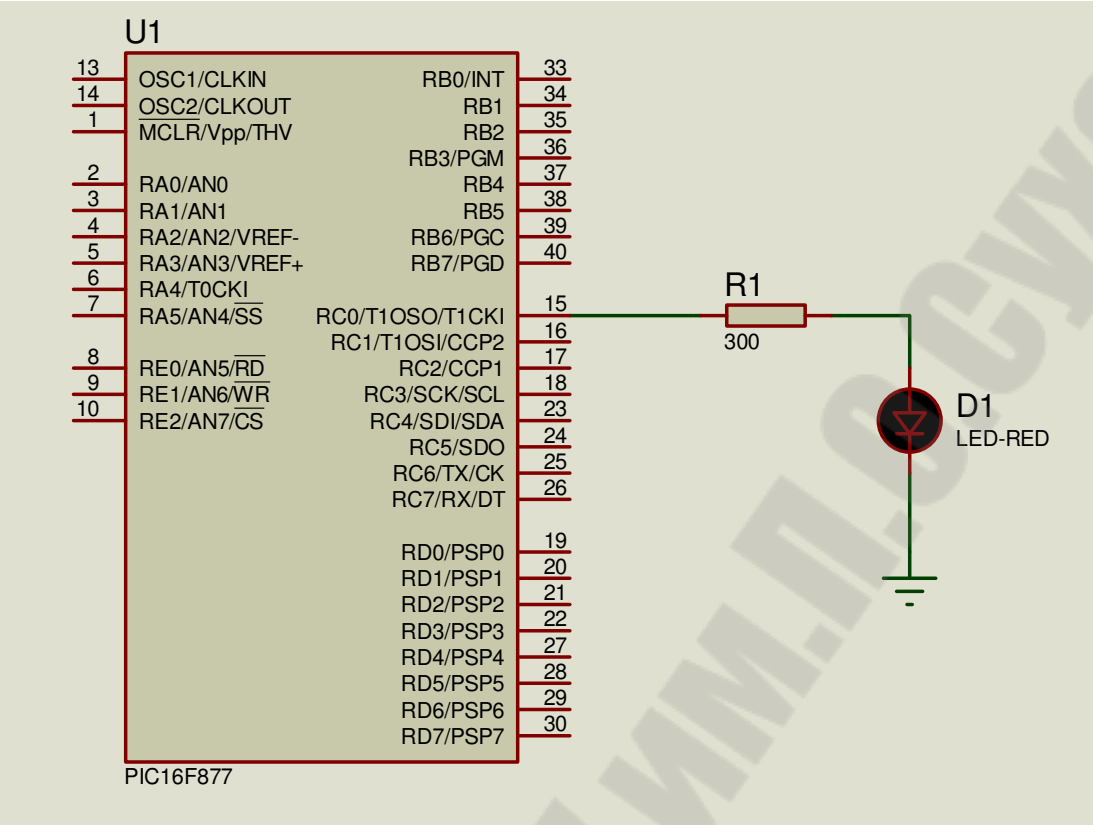

Рисунок 1 – Принципиальная схема МКУ
для управления светодиодом

Рассмотрим программу, которая производит переключение светодиода через интервал 0,5 с, т. е. с частотой 1 Гц:

```
/*****************************************************
led.c – программа  управления светодиодом
*****************************************************/
void  main( )
{
    TRISC.B0 = 0;     // настроить 0-ю линию (RC0) порта С на вывод
    PORTC.B0 = 0;                  // погасить светодиод
    while(1)                       // бесконечный цикл вывода
    {
        PORTC.B0 = 1;          // зажечь светодиод
        Delay_ms(500);         // задержка на 500 мс
        PORTC.B0 = 0;          // погасить светодиод
        Delay_ms(500);         // задержка на 500 мс
    }
}
```

3.2.2 Выполните разработку программы в следующей последовательности.

Запустите программу mikroC PRO for PIC. С помощью команды **New Project…** запустите New Project Wizard. Создайте новый проект с именем led в папке f:\ … \Lab3. Выберите микроконтроллер PIC16F877 и тактовую частоту 8 МГц. В окне редактора наберите текст исходной программы led.c и сохраните его в папке. Получите файл led.hex для загрузки в память микроконтроллера.

3.2.3 Теперь надо проверить работу программы в «реальных» условиях. С этой целью запустите программу ISIS.exe пакета Proteus с помощью ярлычка с надписью ISIS на рабочем столе компьютера. Создайте новый проект, используя пункт меню **File > New Design**.

На экране появится диалоговое окно Create New Design, предлагающее выбрать шаблон для создания нового проекта. Щелкните левой кнопкой мыши по варианту DEFAULT (проект с параметрами по умолчанию), а затем по кнопке [OK].

3.2.4 Для построения схемы МКУ нам нужен микроконтроллер PIC16F877, светодиод красного свечения и резистор сопротивлением 300 Ом. Получить все эти элементы можно из библиотек компонентов, которые имеются в Proteus.

Выбор элементов из библиотек Proteus производится следующим образом. Надо щелкнуть левой кнопкой мышьи по кнопке [P] в верхнем левом углу переключателя объектов Object Selector, либо дважды щелкнуть левой кнопкой мыши в поле Object Selector. На экране появится окно Pick Devices библиотеки компонентов.

Компоненты (элементы схем) лучше выбирать по ключевым словам Keywords (названиям элементов) следующим образом:

• сначала найдите микроконтроллер. Для этого наберите в окне Keywords слово PIC16F877;

• затем выберите светодиод. Для этого очистите строку Keywords и введите слово LED-RED (красный светодиод);

• затем выберите резистор. С этой целью очистите строку Keywords и введите слово RES (резистор). В окне результата выберите строку RES DEVICE;

Теперь можно закрыть библиотеку. Для этого нужно закрыть окно библиотеки или, что проще, нажать клавишу Enter.

3.2.5 Разместите элементы МКУ в окне редактирования. Вначале поместите микроконтроллер. Для удобства размещения увеличьте масштаб отображения элементов. Разместите на схеме резистор и светодиод согласно рисунку 1.

Затем перейдите в режим Terminals Mode, для чего требуется щелкнуть левой кнопкой мыши по иконке с соответствующей надписью на панели инструментов. После этого в окне Object Selector появится список доступных элементов. Выберите из списка клемму GROUND (Земля, общий) и поместите ее около светодиода.

Выполните соединение элементов между собой согласно принципиальной схеме, приведенной на рисунке 1.

3.2.6 Далее необходимо установить параметры компонентов принципиальной схемы МКУ.

Подведите курсор на изображение резистора R1 и дважды щелкните левой кнопкой мыши. Откроется окно редактирования свойств резистора. Введите в поле Resistance число 300 (сопротивление). Щелкните по кнопке [OK] для подтверждения выбора.

3.2.7 После завершения разводки необходимо сохранить проект. Для этого выберите пункт меню **File > Save Design As…** В диалоговом окне найдите папку f:\...\Lab3 и сохраните в ней проект под именем led.dsn.

3.2.8 Теперь можно проверить работу МКУ, однако перед этим необходимо записать hex-код разработанной программы в память МК, или, как часто говорят, запрограммировать МК. С этой целью подведите курсор мыши на изображение микроконтроллера и дважды щелкните левой кнопкой. Откроется окно редактирования свойств компонента Edit Component. В этом окне щелкните по кнопке в правой части строки Program File. Откроется окно Select File Name с hex-файлами из папки f:\...\Lab3. Выберите файл с именем led.hex и нажмите кнопку [Открыть]. Затем в строке Processor Clock Frequency (тактовая частота процессора) выставьте 8 МГц. Нажмите кнопку [OK] для подтверждения выбора параметров. Для запуска программы на выполнение щелкните по кнопке [Play], расположенной слева в нижней части экрана. Светодиод должен попеременно загораться и гаснуть с интервалом 0,5 секунды. Остановить выполнение программы можно с помощью кнопки [Stop].

3.3 Исследование программы формирования меандра

В различных управляющих и измерительных устройствах широко применяются периодические последовательности прямоугольных импульсов, у которых длительность импульса $t_и$ равна паузе $t_п$:

$$t_и = t_п = T / 2,$$

где T – период следования импульсов; $f = 1/T$ – частота следования.

Последовательность прямоугольных импульсов с такими параметрами часто называют меандром. Меандр легко сформировать программным методом с использованием функций временной задержки.

3.3.1 Допустим, что требуется разработать МКУ, которое выдает на линию RC1 порта С микроконтроллера меандр с частотой 10 Гц (период 100 мс). Схема МКУ (при моделировании ее в среде Proteus) приведена на рисунке 2.

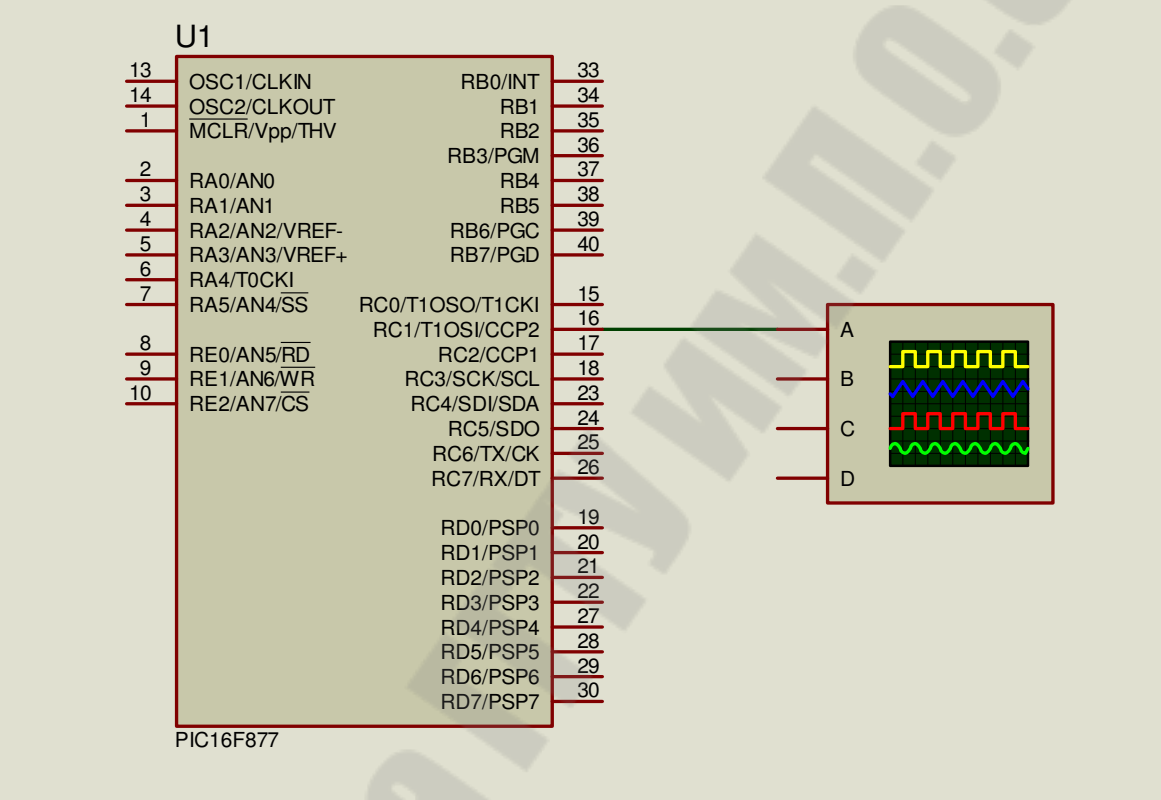

Рисунок 2 – Схема МКУ для генерации меандра

Текст программы генерации меандра на языке mikroC следующий:

```
/**************************************************
meander.c – программа генерации меандра частоты 10 Гц
**************************************************/
void  main( )
{
    TRISC.B1 = 0;      // настроить 1-ю линию (RC1) порта С на вывод
     while(1)
     {
          PORTC.B1 = 1;
```

```
        Delay_ms(50);
        PORTC.B1 = 0;
        Delay_ms(50);
    }
}
```

3.3.2 Выполните разработку программы в следующей последовательности.

Запустите программу mikroC PRO for PIC. С помощью команды **New Project…** запустите New Project Wizard. Создайте новый проект с именем meander в папке f:\ … \Lab3. Выберите микроконтроллер PIC16F877 и тактовую частоту 8 МГц. В окне редактора наберите текст исходной программы meander.c и сохраните его в папке. Получите файл meander.hex для загрузки в память микроконтроллера.

3.3.3 Для проверки работы МКУ запустите программу ISIS.exe пакета Proteus. Создайте новый проект, используя пункт меню **File > New Design**. Затем выберите из библиотек Proteus микроконтроллер PIC16F877.

Разместите микроконтроллер в окне редактирования. Затем перейдите в режим Virtual Instruments Mode (Режим виртуальных инструментов), для чего требуется щелкнуть левой кнопкой мыши по иконке с соответствующей надписью на панели инструментов. В окне Object Selector появится список доступных элементов (приборов). Выберите из списка Oscilloscope (осциллограф) и поместите его в окне редактирования согласно рисунку 2. Соедините канал А осциллографа с линией RC1 порта С микроконтроллера.

После завершения разводки необходимо сохранить проект. Для этого выберите пункт меню **File > Save Design As…** Раскройте папку f:\....\Lab3 и сохраните в ней проект под именем meander.dsn.

3.3.4 Теперь можно приступить к проверке работы МКУ. Сначала нужно записать hex-код разработанной программы meander.hex в память МК. Для проверки работы МКУ с помощью кнопки [Play] запустите программу. Настройте осциллограф, чтобы на экране было устойчивое изображение импульсов. Замерьте период и длительность импульсов. Сравните полученные значения с указанными в программе. Остановите работу МК с помощью кнопки [Stop].

3.4 Программирование и исследование процедуры опроса переключателя

3.4.1 Рассмотрим теперь, как на языке mikroC выполняется опрос состояния контактов переключателя (кнопки). На рисунке 3 приведена схема МКУ (при моделировании ее в Proteus), в которой управление светодиодом D1 производится от переключателя (кнопки) SB1 по следующему алгоритму. Когда контакт SB1 замкнут, светодиод D1 горит, когда контакт SB1 разомкнут, D1 не горит. Рассмотрим два варианта программы управления светодиодом D1 от кнопки SB1. Для упрощения текста программы будем считать, что у кнопки SB1 нет дребезга контактов.

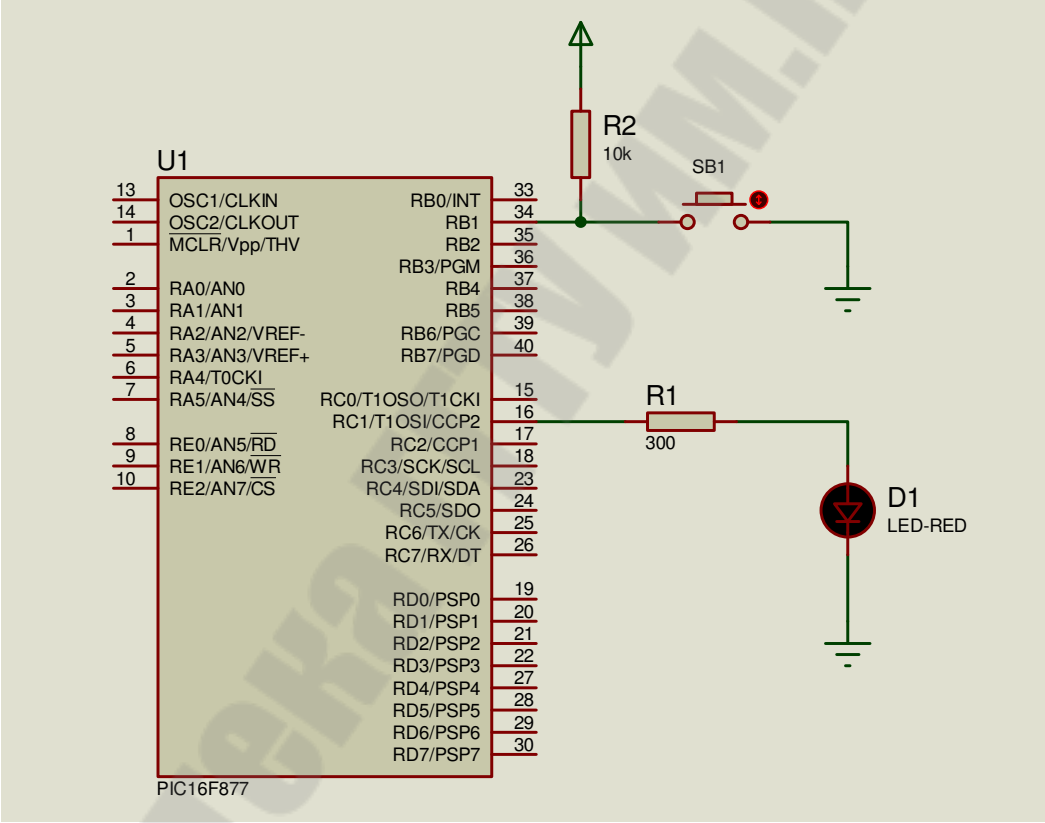

Рисунок 3 – Принципиальная схема МКУ с управлением светодиодом от кнопки

3.4.2 Сначала исследуем программу, в которой для опроса состояния контактов кнопки используется оператор выбора if … else:

```
/*************************************************************
led_but1.c – первая программа управления светодиодом от кнопки
*************************************************************/
```

```
void  main( )
{
   TRISB.B1 = 1;        // настроить 1-ю линию (RB1) порта B на ввод
   TRISC.B1 = 0;       // настроить 1-ю линию (RC1) порта C на вывод
   PORTC.B1 = 0;                // погасить светодиод
   while(1)                     // бесконечный цикл опроса
   {
      if(PORTB.B1 ==0)      // если контакт кнопки SB1 замкнут, то
         PORTC.B1 = 1;      // зажечь светодиод
      else                      // иначе
         PORTC.B1 = 0;       // погасить светодиод
    }
}
```

3.4.3 Выполните разработку программы в следующей последовательности.

Запустите программу mikroC PRO for PIC. С помощью команды **New Project…** запустите New Project Wizard. Создайте новый проект с именем led_but1 в папке f:\ … \Lab3. Выберите микроконтроллер PIC16F877 и тактовую частоту 8 МГц. В окне редактора наберите текст исходной программы led_but1.c и сохраните его в папке. Получите файл led_but1.hex для загрузки в память микроконтроллера.

3.4.4 Теперь нужно проверить работу программы led_but1.c в схеме МКУ с помощью Proteus. С этой целью запустите программу ISIS.exe пакета Proteus VSM с помощью ярлычка с надписью ISIS на рабочем столе компьютера. Создайте новый проект, используя пункт меню **File > New Design**. В открывшемся диалоговом окне щелкните по варианту DEFAULT, а затем по кнопке [OK].

Для выбора элементов схемы из библиотек Proteus щелкните левой кнопкой мыши по кнопке [P] в верхнем левом углу переключателя объектов Object Selector. На экране появится окно Pick Devices библиотеки компонентов.

Компоненты (элементы схемы) выбирайте по ключевым словам Keywords (названиям элементов) следующим образом:

• для выбора микроконтроллера в окне Keywords введите слово PIC16F877;

• для светодиода введите слово LED-RED (красный светодиод);

• для резистора введите слово RES (резистор). В окне результата выберите строку RES DEVICE;

• для выбора переключателя (кнопки) введите слово BUTTON (кнопка). В окне результатов выберите строку BUTTON ACTIVE.

3.4.5 Разместите элементы МКУ (резисторы, светодиод и кнопку) в окне редактирования согласно рисунку 3.

Затем перейдите в режим Terminals Mode, для чего требуется щелкнуть левой кнопкой мыши по иконке с соответствующей надписью на панели инструментов. После этого в окне Object Selector появится список доступных элементов. Выберите из списка клемму GROUND (земля, общий провод) и поместите ее около кнопки и светодиода. Затем выберите из списка клемму POWER (питание). Выполните соединение элементов между собой.

Далее необходимо установить параметры резистора R1. Затем сделайте надпись SB1 около изображения кнопки на схеме. С этой целью перейдите в режим Text Script Mode, щелкнув по иконке (значку) с одноименной всплывающей надписью на панели инструментов. Затем подведите курсор к изображению кнопки (примерно на 1 см выше) и щелкните левой кнопкой мыши. На экране появится окно редактирования текста Edit Script Block. Введите в поле Text слово SB1 и щелкните по кнопке [ОК]. Надпись SB1 должна появиться около изображения кнопки на схеме МКУ.

После завершения разводки необходимо сохранить проект. Для этого выберите пункт меню **File > Save Design As…** Раскройте папку f:\...\Lab3 и сохраните в ней проект под именем led_but1.dsn.

3.4.5 Запишите hex-код программы led_but1.hex из папки Lab3 в микроконтроллер. Для запуска программы на выполнение щелкните по кнопке [Play], расположенной слева в нижней части экрана. Согласно алгоритму программы led_but1.c после запуска МКУ светодиод должен быть погашен. Нажмите кнопку SB1 на схеме МКУ, щелкнув по кружку активатора (красного цвета) около кнопки. Контакт кнопки замкнется, а светодиод D1 должен загореться. После этого вновь щелкните левой кнопкой мыши по кружку активатора. Контакт кнопки должен разомкнуться, а светодиод погаснуть. Теперь щелкните по середине изображения кнопки. Контакт кнопки SB1 должен кратковременно замкнуться, а затем разомкнуться. В это время светодиод D1 должен загореться. Если все эти действия МКУ выполняет правильно, то можно сделать вывод, что программа работает согласно

заданному алгоритму. Щелчком по кнопке [Stop] остановите выполнение программы.

3.4.6 Теперь исследуем программу, в которой для опроса состояния контактов кнопки SB1 используются операторы цикла while. Схема МКУ для этой программы приведена на рисунке 4.

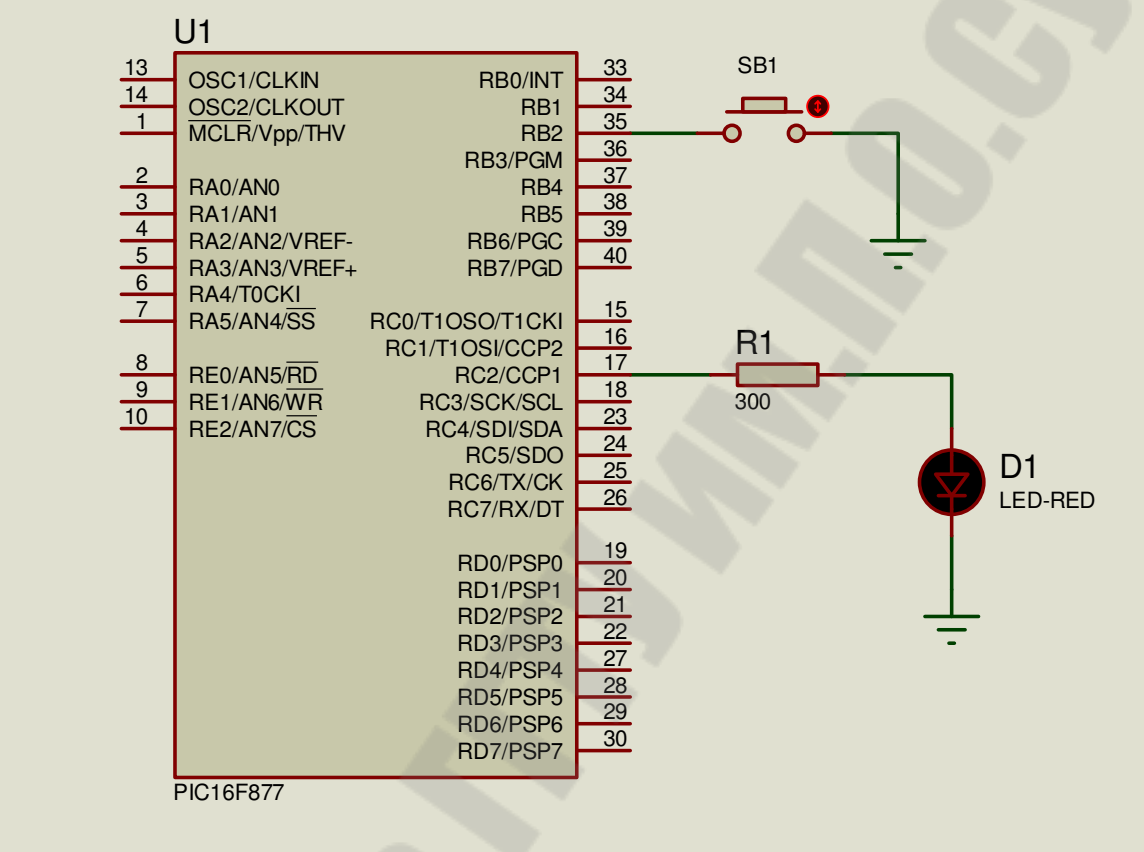

Рисунок 4 – Вторая схема МКУ с управлением светодиодом от кнопки

Особенностью схемы МКУ является отсутствие внешнего подтягивающего резистора на линии RB2 порта B, к которой подключены контакты кнопки. Дело в том, что в микроконтроллерах семейства PIC16 имеется возможность программного подключения к линиям порта B внутренних подтягивающих резисторов. Это выполняется сбросом в нуль седьмого разряда в регистре конфигурации (опций) OPTION_REG. Следует отметить, что программным методом можно подключить внутренние резисторы только к линиям порта B, настроенным на ввод:

```
/*****************************************************
 led_but2.c -  вторая программа управления светодиодом от кнопки
***************************************************** */
void  main( )
{
    TRISB.B2 = 1;             // настроить 2-ю линию  порта B на ввод
    TRISC.B2 = 0;            // настроить 2-ю линию  порта C на вывод
    PORTC.BC2 = 0;                    // погасить светодиод
    OPTION_REG.B7 = 0;     // подключить внутренние
                      // подтягивающие резисторы к линиям порта B
    while(1)                          // бесконечный цикл повторения
    {
        while(PORTB.B2 == 1);   // ожидание замыкания контакта SB1
            PORTC.B2 = 1;               // зажечь светодиод
        while(PORTB.B2 == 0);  // ожидание размыкания контакта SB1
            PORTC.B2 = 0;               // погасить светодиод
    }
}
```

3.4.7 Выполните разработку программы в следующей последовательности.

Запустите программу mikroC PRO for PIC. С помощью команды **New Project…** запустите New Project Wizard. Создайте новый проект с именем led_but2 в папке f:\ … \Lab3. Выберите микроконтроллер PIC16F877 и тактовую частоту 8 МГц. В окне редактора наберите текст исходной программы led_but2.c и сохраните его в папке. Получите файл led_but2.hex для загрузки в память микроконтроллера.

3.4.8 Теперь нужно проверить работу программы led_but2.c в схеме МКУ с помощью Proteus VSM.

Запустите программу ISIS.exe пакета Proteus. Создайте новый проект, используя пункт меню **File > New Design**. В открывшемся диалоговом окне щелкните по варианту DEFAULT, а затем по кнопке [OK].

Для выбора элементов схемы из библиотек Proteus щелкните мышью по кнопке [P] в верхнем левом углу переключателя объектов Object Selector. На экране появится окно Pick Devices библиотеки компонентов. Выберите необходимые элементы и разместите их на схеме согласно рисунку 4. После завершения разводки сохраните проект в папке f:\ … \Lab3 под именем led_but2.dsn. Запишите код программы из файла led_but2.hex в память микроконтроллера.

3.4.9 Выполните проверку работы МКУ. Для этого с помощью кнопки [Play] запустите выполнение программы. Согласно алгоритма программы led_but2.c после запуска МКУ светодиод должен быть погашен. Нажмите кнопку SB1 на схеме МКУ, щелкнув по кружку активатора около кнопки. Контакт кнопки замкнется, а светодиод D1 должен загореться. После этого вновь щелкните левой кнопкой мыши по кружку активатора. Контакт кнопки должен разомкнуться, а светодиод погаснуть. Теперь щелкните по середине изображения кнопки. Контакт кнопки SB1 должен кратковременно замкнуться, а затем разомкнуться. В это время светодиод D1 должен загореться. Если все эти действия МКУ выполняет правильно, то можно сделать вывод, что программа работает согласно заданному алгоритму. Щелчком по кнопке [Stop] остановите выполнение программы.

4 Содержание отчета

Наименование и цель работы. Принципиальные схемы МКУ и тексты исследованных программ.

Контрольные вопросы

1 Как в языке mikroC выполняется управление отдельными битами портов МК?

2 Как выполняется ввод состояния линии порта МК?

3 Как выполняется временная задержка в программах?

4 Какой сигнал называется в электронике меандром?

5 Как можно сформировать меандр с помощью PIC-микроконтроллера?

6 Как выполняется опрос состояния контактов кнопки?

7 Как можно программно подключить к порту PIC-микроконтроллера внутренние подтягивающие резисторы?

Лабораторная работа № 4

Исследование совместной работы ЖК-дисплея и PIC-микроконтроллера

1 Цель работы

Изучить принципы построения алфавитно-цифрового жидкокристаллического дисплея (ЖКД) с контроллером HD44780. Изучить и исследовать методы программирования его работы на языке mikroC для микроконтроллеров семейства PIC16. Исследовать методику проверки работы МКУ с ЖК-дисплеем с помощью среды моделирования Proteus.

2 Основные теоретические сведения

2.1 Структура жидкокристаллического дисплея

Алфавитно-цифровые ЖКД, или по-английски LCD (Liquid Crystal Display), представляют собой недорогие и удобные модули, позволяющие сэкономить время и ресурсы при разработке новых изделий, при этом обеспечивающие отображение большого объема информации при хорошей различимости и низком энергопотреблении.

Японская фирма Hitachi разработала специальную микросхему – контроллер HD44780 для управления подобными ЖК-дисплеями. Этот контроллер определил интерфейс, который стал стандартом «де-факто» для ЖК-дисплеев. Аналоги этого контроллера или совместимые с ним по интерфейсу микросхемы выпускают множество фирм, среди которых Epson, Toshiba, Sanyo, Samsung, Philips. Большее число форм производят ЖК-дисплеи на базе данных контроллеров. Эти дисплеи можно встретить в самых разнообразных устройствах: измерительных приборах, медицинском оборудовании, промышленном и технологическом оборудовании, офисной технике – принтерах, телефонах, факсимильных и копировальных аппаратах.

Контроллер HD44780 может управлять двумя строками по 40 символов в каждой (для дисплеев с четырьмя строками по 40 символов используются два однотипных контроллера), при матрице символов 5 × 7 точек. Существует несколько различных стандартных форматов ЖК-дисплеев (количество символов × число строк): 8 × 2, 16 × 1, 16 × 2, 16 × 4, 20×1, 20 × 2, 20 × 4, 40 × 2, 40 × 4. Примером

может служить ЖКД типа WM-C1602N-2YLY с 32 позиционными индикаторами (16 × 2 символов, размер 64 × 16,5 мм). Конструктивно этот дисплей представляет собой печатную плату с установленными на ней контроллером HD44780 и ЖК-индикатором. Плата содержит 14-контактное поле, расположенное в нижней части, а также 2 контакта (выводы питания подсветки) в правой части. Назначение выводов поясняет таблица 1.

Таблица 1 – Назначение выводов ЖК-дисплея

Номер вывода	**Обозначение**	**Функция**
1	V_{DD}	Напряжение питания (+5 В)
2	V_{SS}	Общий (земля)
3	V_{EE}	Управление контрастом
4	RS	Сигнал выбора регистра
5	R/W	Сигнал чтение/запись
6	E	Сигнал разрешение
7–14	D0-D7	Биты данных

Основными чертами интерфейса контроллера HD44780 являются такие характеристики.

Данные передаются по 4- или 8-разрядной шине данных, что определяется пользователем. Эти данные могут быть либо командами, либо символьной информацией. Использование 4-разрядного режима позволяет ограничить весь интерфейс семью линиями, однако процесс передачи данных будет немного более медленным, чем при 8-разрядном режиме.

Управление дисплеем выполняется с использованием трех линий:

• линия RS (выбор регистра), которая определяет, будет передаваться команда или символьные данные;

• линия R/W (чтение/запись), которая определяет направление перемещения данных (R/W = 1 – чтение, R/W = 0 – запись);

• линия E (разрешение), которая выполняет функцию тактирования с целью синхронизации процесса передачи данных.

Контроллер имеет простой набор команд, который позволяет управлять работой дисплея. В его состав входят команды инициализации и сброса дисплея, управления положением и характеристиками курсора и т. д.

2.2 Библиотека функций управления ЖК-дисплеем

Для облегчения разработки программ в интегрированной среде mikroC PRO for PIC имеются библиотечные функции для управления алфавитно-цифровым ЖК-дисплеем на основе контроллера HD44780 с 4-разрядной шиной данных. Рассмотрим эти функции.

Lcd_Init – эта функция инициализирует модуль ЖКД.

Прототип:

void Lcd_Init( ).

Перед тем, как использовать эту функцию, необходимо указать соединение ЖК-модуля с выводами портов МК.

Пример

// Рассмотрим процесс инициализации ЖКД, присоединенного к порту С микроконтроллера. В таблице 2 указаны выводы ЖКД и соответствующие им выводы порта С. Следует отметить, что в рассматриваемых функциях не используется режим чтения из ЖКД, поэтому на вывод R/W дисплея ннужно подавать постоянно логический 0, для чего его можно просто соединить с общим проводом.

Таблица 2 – Соединение выводов ЖКД и МК

Наименование выводов ЖКД	**Наименование выводов порта С**
RS	RC2
E	RC3
D4	RC4
D5	RC5
D5	RC6
D6	RC7

```
// Присоединение выводов ЖКД
sbit LCD_RS at RC2_bit;
sbit LCD_EN at RC3_bit;
sbit LCD_D4 at RC4_bit;
sbit LCD_D5 at RC5_bit;
sbit LCD_D6 at RC6_bit;
sbit LCD_D7 at RC7_bit;
// Направление выводов
sbit LCD_RS_Direction at TRISC2_bit;
sbit LCD_EN_Direction at TRISC3_bit;
sbit LCD_D4_Direction at TRISC4_bit;
sbit LCD_D5_Direction at TRISC5_bit;
```

```
sbit LCD_D6_Direction at TRISC6_bit;
sbit LCD_D7_Direction at TRISC7_bit;
      // Инициализация модуля ЖКД
Lcd_Init( );
```

Lcd_Out – эта функция выводит текст на ЖКД начиная с позиции, которая указана как параметр функции.

Прототип:

void Lcd_Out(char row, char column, char *text).

Параметры:

row – начальная позиция номера строки;

column – начальная позиция номера колонки;

text – текст для вывода, может быть строковая переменная или текст, помещенный в двойные кавычки.

На рисунке 1 приведено расположение позиций дисплея формата 16 × 2 с нумерацией колонок и строк.

Номера колонок (позиций)																
Строка 1	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Строка 2	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Рисунок 1 – Расположение позиций дисплея формата 16 × 2

Пример

```
// Вывести текст “Hello!” на ЖКД, начиная со строки 2, колонка 1
Lcd_Out(2, 1, “Hello!”);
```

Lcd_Out_Cp – эта функция выводит текст на ЖКД, начиная с текущей позиции курсора.

Прототип:

void Lcd_Out_Cp(char *text);

Параметр:

text – текст для вывода, может быть строковая переменная или текст, помещенный в двойные кавычки.

Пример

```
    // Вывести на ЖКД текст “Student” с текущей позиции
    // курсора
Lcd_Out_Cp(“Student”);
```

Lcd_Chr – эта функция выводит на ЖКД символ в позицию, которая указана как параметр.

Прототип:

void Lcd_Chr(char row, char column, char out_char).

Параметры:

row – начальная позиция номера строки;

column – начальная позиция номера колонки;

out_char – символ для вывода, может быть строковая переменная или литерал, заключенный в одиночные кавычки.

Пример

```
// Вывести символ d в строку 2, колонка 3
Lcd_Chr(2, 3, ‘d’);
```

Lcd_Chr_Cp – эта функция выводит символ на ЖКД в текущую позицию курсора.

Прототип:

void Lcd_Chr_Cp(char out_char);

Параметр:

out_char – символ для вывода, может быть строковая переменная или литерал, заключенный в одиночные кавычки.

Пример

```
/ Вывести символ m в текущую позицию курсора
Lcd_Chr_Cp(‘m’);
```

Lcd_Cmd – эта функция посылает команду в ЖКД.

Прототип:

void Lcd_Cmd(char out_char);

Параметр:

out_char – команда для вывода.

При выполнении данной лабораторной работы будут использованы только 4 команды:

_LCD_CLEAR – очистка дисплея;

_LCD_CURSOR_OFF – отключение отображения курсора;

_LCD_SHIFT_LEFT – сдвиг текста на одну позицию влево

_LCD_SHIFT_RIGHT – сдвиг текста на одну позицию вправо

Пример

```
// Очистить ЖКД
Lcd_Cmd(_LCD_CLEAR);
// Отключить отображение курсора
Lcd_Cmd(_LCD_CURSOR_OFF);
```

3 Порядок выполнения работы

3.1 Создание папки для работы в mikroC PRO for PIC

При выполнении лабораторной работы № 4 мы будем использовать папку с именем Lab4, которую необходимо предварительно создать.

С этой целью откройте вашу рабочую папку и создайте в ней (с помощью клавиши F7) новую папку с именем Lab4.

В дальнейшем Вы будете записывать и хранить в этой папке все файлы при выполнении лабораторной работы № 4. Полный путь к этой папке будет такой:

F:\MPT\Ivanov\Lab4

3.2 Вывод текстовой информации на ЖКД

3.2.1 На рисунке 2 приведена схема МКУ, в котором ЖКД присоединен к выводам порта С микроконтроллера.

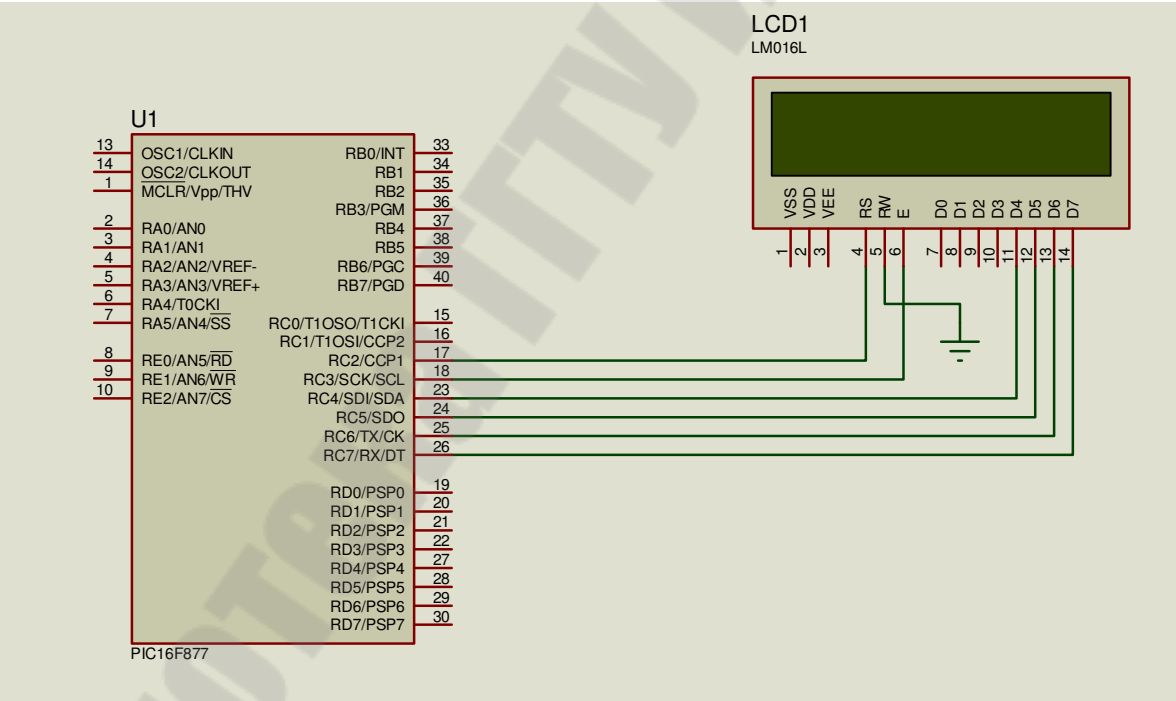

Рисунок 2 – Схема МКУ с ЖКД

Рассмотрим программу, которая после запуска выводит на ЖКД различные сообщения.

Текст программы на языке mikroC имеет следующий вид:

```
/*********************************************************
lcd.c – программа вывода текста на ЖКД
*********************************************************/
// Присоединение выводов ЖКД
sbit  LCD_RS  at  RC2_bit;
sbit  LCD_EN  at  RC3_bit;
sbit  LCD_D4  at  RC4_bit;
sbit  LCD_D5  at  RC5_bit;
sbit  LCD_D6  at  RC6_bit;
sbit  LCD_D7  at  RC7_bit;
        // Направление выводов
sbit  LCD_RS_Direction  at TRISC2_bit;
sbit  LCD_EN_Direction  at TRISC3_bit;
sbit  LCD_D4_Direction  at  TRISC4_bit;
sbit  LCD_D5_Direction  at  TRISC5_bit;
sbit  LCD_D6_Direction  at  TRISC6_bit;
sbit  LCD_D7_Direction  at  TRISC7_bit;

char  text1[ ] = “MikroElektronika”;      // текст для вывода на ЖКД
char  text2[ ] = “Example”;               // текст для вывода на ЖКД

char  j;                                  // переменная-счетчик
int  time = 500;                          // величина временной задержки

void  main( )
{
    Lcd_Init( );                          // инициализация ЖКД
    Lcd_Cmd(_LCD_CLEAR);                  // очистить ЖКД
    Lcd_Cmd(_LCD_CURSOR_OFF);             // запретить отображение
                                          // курсора
        // вывод на ЖКД текста приветствия
    Lcd_Out(1, 2, “Hello, world!”);       // вывод текста в 1-ю строку
    Lcd_Out(2, 2, “I am PIC16F877”);      // вывод текста во 2-ю строку
    Delay_ms(2000);                       // задержка на время 2 с
    Lcd_Cmd(_LCD_CLEAR);                  // очистить ЖКД
        // вывод на ЖКД текста
    Lcd_Out(1, 1, text1);                 // вывод текста в 1-ю строку
    Lcd_Out(2, 5, text2);                 // вывод текста во 2-ю строку
    Delay_ms(2000);
```

```
    // сдвиг текста вправо
  for(j = 0; j < 4; j++)
  {
     Lcd_Cmd(_LCD_SHIFT_RIGHT);
     Vdelay_ms(time);                    // задержка на время 0,5 с
  }
    // бесконечный сдвиг текста влево и вправо
  while(1)
  {
      for(j = 0; j < 10; j++)
      {
         Lcd_Cmd(_LCD_SHIFT_LEFT);
         Vdelay_ms(time);                // задержка на время 0,5 с
      }
      for(j = 0; j < 10; j++)
      {
         Lcd_Cmd(_LCD_SHIFT_RIGHT);
         Vdelay_ms(time);                // задержка на время 0,5 с
      }
  }
}
```

3.2.2 Выполните разработку программы в следующей последовательности.

Запустите программу mikroC PRO for PIC. С помощью команды **New Project…** запустите New Project Wizard. Создайте новый проект с именем lcd в папке f:\ … \Lab4. Выберите микроконтроллер PIC16F877 и тактовую частоту 8 МГц. В окне редактора наберите текст исходной программы lcd.c и сохраните его в папке. Получите файл lcdhex для загрузки в память микроконтроллера.

3.2.3 Далее нужно создать с помощью Proteus схему МКУ, приведенную на рисунке 2, для проверки работы программы lcd.c. С этой целью запустите программу ISIS пакета Proteus. Создайте новый проект, используя пункт меню **File > New Design.**

3.2.4 Для выбора элементов схемы из библиотек Proteus щелкните левой кнопкой мыши по кнопке [P] в верхнем левом углу переключателя объектов Object Selector. На экране появится окно Pick Device библиотеки компонентов.

Для выбора ЖК-дисплея поставьте курсор мыши в окне Category (Категория) на строку Optoelectronics (Оптоэлектронные приборы) и щелкните левой кнопкой. Затем в окне Sub-category выберите строку Alphanumeric LCDs и также щелкните левой кнопкой. В окне Results (Результаты) щелкните по строке

LM016L DISPLAY 16 × 2 Alphanumeric LCD

Это алфавитно-цифровой жидкокристаллический двухстрочный дисплей для отображения 16 символов в строке.

Для выбора микроконтроллера наберите в окне Keywords слово PIC16F877 и щелкните по строке в окне Results.

Если выбранные элементы появились в списке окна Object Selector, то закройте библиотеку.

3.2.5 Теперь разместите микроконтроллер и ЖК-дисплей в окне редактирования согласно схеме на рисунке 2. Выполните соединение элементов между собой согласно схеме, приведенной на рисунке 2.

После завершения разводки необходимо сохранить проект. Для этого выберите пункт меню **File > Save Design As…**. В диалоговом окне раскройте папку f:\ ... \Lab4 и сохраните в ней проект под именем lcd.dsn.

Далее загрузите в микроконтроллер файл lcd.hex из папки Lab4, установите тактовую частоту 8 МГц.

3.2.6 Для проверки работы МКУ с помощью кнопки [Play] запустите выполнение программы.

Убедитесь, что на дисплее появляются последовательно различные тексты, а затем производятся сдвиги текста влево и вправо.

В заключение остановите выполнение программы и сверните окно Proteus.

3.3 Задание для самостоятельной работы

Для МКУ на рисунке 2 нужно разработать программу, которая в бесконечном цикле выводит на ЖКД «бегущую строку» текста “IDE mikroC PRO for PIC”. Сдвиг должен выполняться влево.

Указание. В программе начните вывод на ЖКД в 1-ю строку с колонки 16. Сдвиг должен выполняться с задержкой 0,5 с.

Разработанную программу назовите lcd2.c.

В среде mikroC PRO for PIC создайте новый проект с именем lcd2 (микроконтроллер PIC16F877, частота 8 МГц) и поместите его в папку f:\ … \Lab4. Получите файл lcd2.hex для загрузки в память МК. Проверьте работу программы с помощью Proteus, открыв проект (схему

МКУ) lcd.dsn из папки Lab4. В микроконтроллер загрузите файл lcd2.hex.

С помощью кнопки [Play] запустите выполнение программы. Убедитесь в ее работе согласно заданного алгоритма.

В заключение остановите работу МК с помощью кнопки [Stop] и сверните окно Proteus.

4 Содержание отчета

Наименование и цель работы. Тексты программ, принципиальную схему МКУ к заданию для самостоятельной работы (комментарии в тексте программы обязательны!).

Контрольные вопросы

1 Каково назначение контроллера HD44780?

2 Поясните назначение выводов ЖК-дисплея.

3 Какие существуют способы соединения ЖК-дисплея с управляющим устройством?

4 Какие библиотечные функции используются в mikroC PRO for PIC для управления ЖК-дисплеем?

5 Объясните схему подключения ЖК-дисплея к PIC-микроконтроллеру.

6 Объясните порядок вывода текста на ЖК-дисплей.

Лабораторная работа № 5

Исследование аналого-цифрового преобразователя PIC-микроконтроллера

1 Цель работы

Изучить принципы построения встроенного аналого-цифрового преобразователя (АЦП) микроконтроллеров семейства PIC16. Изучить и исследовать методы программирования АЦП на языке mikroC. Исследовать методику проверки работы МКУ с АЦП и ЖК-дисплеем с помощью среды моделирования Proteus.

2 Основные теоретические сведения

2.1 Аналого-цифровые преобразователи в PIC-микроконтроллерах

Многие микроконтроллеры семейства PIC16 имеют встроенные многоканальные АЦП, используемые для преобразования аналогового входного напряжения в диапазоне от 0 до UDD (напряжение питания МК) в цифровой код. Аналого-цифровые преобразователи выдают 10-разрядный код, а число аналоговых каналов зависит от конкретного типа МК. Так, PIC16F877 имеет 8 аналоговых каналов. Для ввода аналоговых сигналов обычно используются линии порта А, которые требуется настроить на ввод. Реальная величина погрешности преобразования, характер использования выводов портов и скорость выполнения операций зависят от конкретного типа МК, а также тактовой частоты его работы.

Модуль АЦП имеет внутренний делитель частоты, обеспечивающий деление тактовой частоты МК в 2, 8 и 32 раза. Встроенный RC-генератор с частотой 250 кГц обычно используется для АЦ-преобразования в тех случаях, когда сам МК переводится в энергосберегающий режим SLEEP.

2.2 Функции управления аналогово-цифровым преобразователем компилятора mikroC PRO

Для облегчения разработки программ управления встроенным аналогово-цифровым преобразователем PIC-микроконтроллеров в библиотеке IDE mikroC PRO for PIC имеются две функции.

ADC_Init – эта функция производит инициализацию внутреннего модуля АЦП микроконтроллера. При этом предполагается,

что АЦП будет тактироваться от внутреннего RC-генератора модуля, а выводы портов МК, используемые как аналоговые входы, сконфигурированы на ввод.

Прототип:

```
void  ADC_Init( );
```

Пример

```
// Инициализировать модуль АЦП с использованием в качестве
// аналоговых входов линии порта A – RA0, RA2, RA3
// (аналоговые каналы AN0, AN1, AN2)
TRISA = 0b00000111;    // Настроить линии порта A - RA0, RA1, RA2
                       // на ввод, остальные на вывод
……..
ADC_Init( );
```

ADC_Read – эта функция вводит аналоговое напряжение из заданного канала и запускает АЦП. Функция возвращает 10-разрядное двоичное число без знака (выходной код АЦП). Перед использованием этой функции модуль АЦП должен быть инициализирован.

Прототип:

```
unsigned  int  ADC_Read(unsigned  char  channel);
```

Здесь параметр channel (канал) представляет собой номер аналогового канала, из которого извлекается напряжение.

Для микроконтроллера PIC16F877 номера аналоговых каналов соответствуют следующим выводам портов A и E:

Номер канала	Вывод порта A или E
0	RA0 (AN0)
1	RA1 (AN1)
2	RA2 (AN2)
3	RA3 (AN3)
4	RA5 (AN4)
5	RE0 (AN5)
6	RE1 (AN6)
7	RE2 (AN7)

Пример

```
// Прочитать код из канала 2 модуля АЦП
unsigned  int  adc_result;       // Объявление двухбайтной целой
                                 // переменной
……
adc_result = ADC_Read(2);
```

3 Порядок выполнения работы

3.1 Создание папки для работы в mikroC PRO for PIC

При выполнении лабораторной работы № 5 мы будем использовать папку с именем Lab5, которую необходимо предварительно создать.

С этой целью откройте вашу рабочую папку и создайте в ней (с помощью клавиши F7) новую папку с именем Lab5.

В дальнейшем Вы будете записывать и хранить в этой папке все файлы при выполнении лабораторной работы № 5. Полный путь к этой папке будет такой:

F:\MPT\Ivanov\Lab5

3.2 Разработка МКУ для измерения и индикации величины аналогового напряжения

3.2.1 Требуется разработать микроконтроллерное устройство (МКУ) на базе PIC16F877, которое измеряет аналоговое напряжение $U1$ в диапазоне от 0 до +5 В и выводит результат измерения на ЖК-дисплей. Тактовая частота работы МК равна 8 МГц. Принципиальная схема МКУ (при моделировании ее в среде Proteus) приведена на рисунке 1.

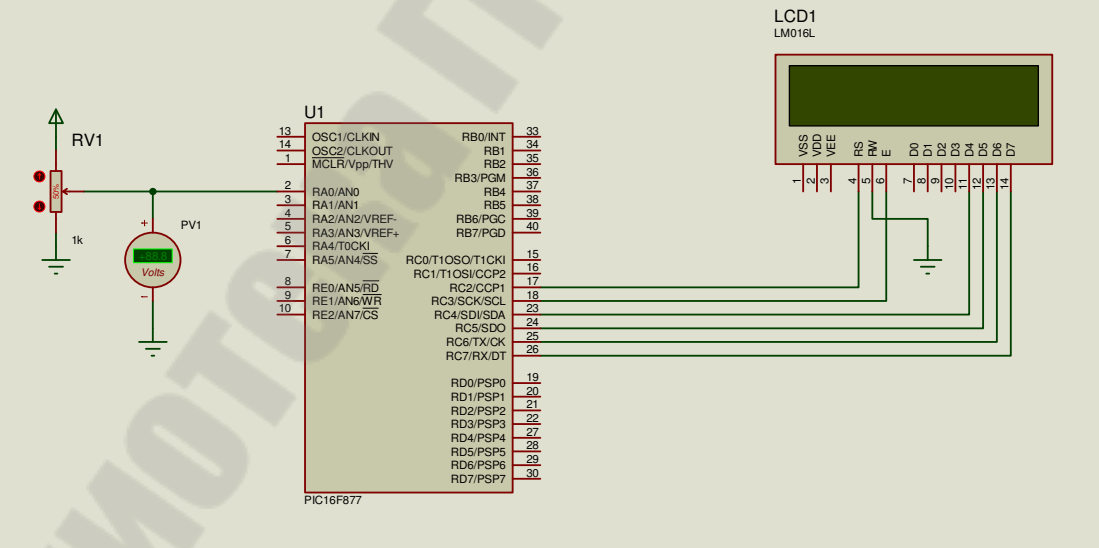

Рисунок 1 – Схема МКУ для измерения аналогового напряжения

На рисунке 1 потенциометр RV1 используется для изменения величины входного напряжения $U1$ в пределах от 0 до +5 В (напряжение UDD = +5 В). Вольтметр PV1 служит для измерения действи-

тельной величины входного напряжения $U1$, снимаемого с движка потенциометра RV1. На экран дисплея должен выводиться текст в виде строки: величина напряжения $U1$ в вольтах, например:

$$U1 = 2.534 \text{ V}$$

Текст программы работы МКУ на языке mikroC может быть следующий:

```
/*******************************************************
adc.c – программа измерения напряжения и вывода  на ЖКД
*******************************************************/
      // Присоединение выводов ЖКД
sbit  LCD_RS  at  RC2_bit;
sbit  LCD_EN  at  RC3_bit;
sbit  LCD_D4  at  RC4_bit;
sbit  LCD_D5  at  RC5_bit;
sbit  LCD_D6  at  RC6_bit;
sbit  LCD_D7  at  RC7_bit;
      // Направление выводов
sbit  LCD_RS_Direction  at TRISC2_bit;
sbit  LCD_EN_Direction  at TRISC3_bit;
sbit  LCD_D4_Direction  at  TRISC4_bit;
sbit  LCD_D5_Direction  at  TRISC5_bit;
sbit  LCD_D6_Direction  at  TRISC6_bit;
sbit  LCD_D7_Direction  at  TRISC7_bit;

void  main( )
{
   int  res_ADC;                  // переменная для хранения кода АЦП
   int  mvolts;             // переменная для хранения напряжения в
                            // милливольтах
   char  num;              // переменная для хранения цифр напряжения в
                           // вольтах
   TRISA = 0x01;           // настроить линию RA0 на ввод
   ADC_Init( );              // инициализация модуля АЦП
   Lcd_Init( );              // инициализация модуля ЖКД
   Lcd_Cmd(_LCD_CLEAR);
   Lcd_Cmd(_LCD_CURSOR_OFF);
   while(1)
```

```
    {
        res_ADC = ADC_Read( 0 );                    // чтение кода АЦП
        mvolts = ((long)res_ADC  * 5000) / 0x03FF;     // преобразование
                                                    // кода АЦП в милливольты
        Lcd_Cmd(_LCD_CLEAR);
        Lcd_Out(2, 4, “U = “);
        num = mvolts / 1000;              // извлечение единиц вольт
         Lcd_Chr_Cp(48 + num);          // отображение числа в коде ASCII
         Lcd_Chr_Cp(‘.’);                // отображение десятичной точки
         num = (mvolts / 100) % 10;    // извлечение десятых долей вольт
         Lcd_Chr_Cp(48 + num);        // отображение числа в коде ASCII
         num = (mvolts / 10) % 10;       // извлечение сотых долей вольт
         Lcd_Chr_Cp(48 + num);       //  отображение числа в коде ASCII
         num = mvolts % 10;            // извлечение тысячных долей вольт
         Lcd_Chr_Cp(48 + num);       // отображение числа в коде ASCII
         Lcd_Out_Cp(“ V”);
         Delay_ms(2000);
     }
}
```

3.2.2 Выполните разработку программы в следующей последовательности.

Запустите программу mikroC PRO for PIC. С помощью команды **New Project…** запустите New Project Wizard. Создайте новый проект с именем adc в папке f:\ … \Lab5. Выберите микроконтроллер PIC16F877 и тактовую частоту 8 МГц. В окне редактора наберите текст исходной программы adc.c и сохраните его в папке. Получите файл adc.hex для загрузки в память микроконтроллера.

3.2.3 С помощью Proteus нужно создать схему МКУ, приведенную на рисунке 1, для проверки работы программы adc.c. С этой целью запустите программу ISIS пакета Proteus. Создайте новый проект, используя пункт меню **File > New Design**.

3.2.4 Для выбора элементов схемы из библиотек Proteus щелкните левой кнопкой мыши по кнопке [P] в верхнем левом углу переключателя объектов Object Selector. На экране появится окно Pick Device библиотеки компонентов.

Для выбора ЖК-дисплея поставьте курсор мыши в окне Category (Категория) на строку Optoelectronics (Оптоэлектронные приборы) и щелкните левой кнопкой. Затем в окне Sub-category выберите

строку Alphanumeric LCDs и также щелкните левой кнопкой. В окне Results (Результаты) щелкните по строке

LM016L DISPLAY 16 × 2 Alphanumeric LCD

Это алфавитно-цифровой жидкокристаллический двухстрочный дисплей для отображения 16 символов в строке.

Для выбора микроконтроллера наберите в окне Keywords слово PIC16F877 и щелкните по строке в окне Results.

Для выбора потенциометра наберите в окне Keywords слово POT-HG. Это потенциометр с активаторами, т. е. возможностью интерактивного изменения положения движка. В окне результата дважды щелкните по строке POT-HG ACTIVE для занесения компонента в список окна Object Selector. После этого закройте библиотеку.

3.2.5 Теперь разместите микроконтроллер и ЖК-дисплей в окне редактирования согласно схеме на рисунке 1. Для удобства размещения компонентов увеличьте масштаб их отображения. Затем разместите потенциометр в окне редактирования согласно рисунку 1. Для выбора вольтметра подведите курсор мыши к значку на панели инструментов с всплывающей надписью Virtual Instruments Mode и щелкните левой кнопкой мыши. В окне Object Selector появится список виртуальных приборов (Instruments). Выберите строку DC VOLTMETER (вольтметр постоянного напряжения) и щелкните левой кнопкой мыши. Затем разместите вольтметр в окне редактирования согласно рисунку 1. После этого перейдите в режим Terminals Mode, для чего требуется щелкнуть левой кнопкой мыши по иконке с соответствующей надписью на панели инструментов. Выберите из списка клемму GROUND (земля, общий провод), а затем клемму POWER (питание). Выполните соединение элементов между собой согласно принципиальной схеме, приведенной на рисунке 1.

После завершения разводки необходимо сохранить новый проект. Для этого выберите пункт меню **File > Save Design As…** Раскройте папку f:\...\Lab5, и сохраните в ней проект под именем adc.dsn.

3.2.6 Загрузите в микроконтроллер файл adc.hex из папки f:\ … \Lab5, установите тактовую частоту 8 МГц.

Для проверки работы МКУ с помощью кнопки [Play] запустите программу. На экране ЖК-дисплея должен появиться текст с сообщением о величине входного напряжения *U*1 в вольтах. Действительное значение этого напряжения на входе МК показывает вольтметр PV1. Если эти значения совпадают, все сделано правильно.

Далее можно проверить работу МКУ при различных значениях напряжения *U*1, которые можно задавать изменением положения движка потенциометра RV1. С этой целью подведите курсор мыши к активатору потенциометра RV1 со значком «+», и несколько раз щелкните левой кнопкой мыши. Наблюдайте за изменением положения движка потенциометра RV1, величиной напряжения на вольтметре PV1 и данными на экране ЖКД.

Затем подведите курсор мыши к активатору потенциометра RV1 со значком «–» и также несколько раз щелкните левой кнопкой мыши. Наблюдайте за изменением положения движка потенциометра RV1, величиной напряжения на вольтметре PV1 и данными на экране ЖКД.

В заключение остановите выполнение программы МКУ и сверните окно Proteus.

3.3 Задание для самостоятельной работы

3.3.1 Требуется разработать МКУ, состоящее из микроконтроллера PIC16F877, ЖК-дисплея и двух кнопок. МКУ должно измерять два аналоговых напряжения и выводить результаты на ЖК-дисплей. Схема МКУ (при моделировании ее в Proteus) приведена на рисунке 2.

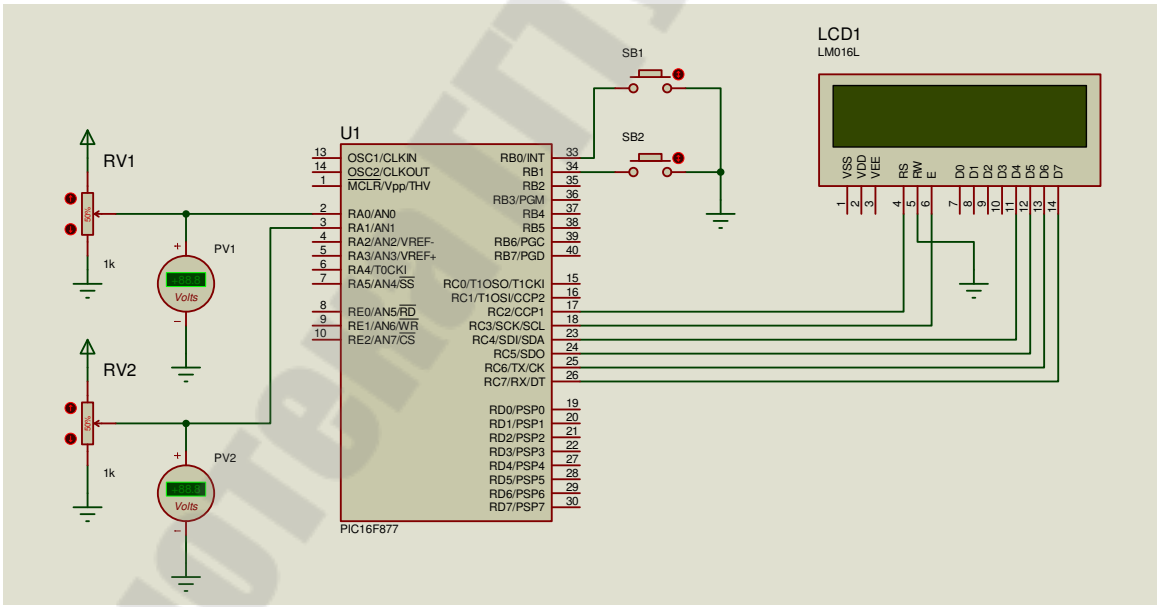

Рисунок 2 – Схема двухканального вольтметра

МКУ должно выполнять следующий алгоритм. После запуска программы и разомкнутых контактах кнопок SB1, SB2 на дисплей выводится текст приглашения: «Нажмите только одну клавишу: SB1 или SB2» в виде:

Press only one

key: SB1 or SB2

При нажатии кнопки SB1 (замыкании ее контакта) МК производит измерение напряжения $U1$, и на дисплей выводится результат в виде:

$U1 = 2.524$ V

При нажатии кнопки SB2 (замыкании ее контакта) МК производит измерение напряжения $U2$, и на дисплей выводится результат в виде:

$U2 = 4.105$ V

В случае одновременного нажатия двух кнопок SB1 и SB2 на экран дисплея выводится текст сообщения об ошибке (одновременного нажатия кнопок) в виде:

Error! Two keys

are pressed

Программа должна производить опрос кнопок и измерение входных напряжений в бесконечном цикле. После каждого вывода на ЖКД должна быть временная задержка на 2 с.

Разработанную программу назовите voltmeter.c.

Текст программы может быть следующий:

```
/*****************************************************
voltmeter.c – программа измерения двух напряжений с выводом
 на ЖКД
*****************************************************/
// Присоединение выводов ЖКД
sbit  LCD_RS  at  RC2_bit;
sbit  LCD_EN  at  RC3_bit;
sbit  LCD_D4  at  RC4_bit;
sbit  LCD_D5  at  RC5_bit;
sbit  LCD_D6  at  RC6_bit;
sbit  LCD_D7  at  RC7_bit;
       // Направление выводов
sbit  LCD_RS_Direction  at TRISC2_bit;
sbit  LCD_EN_Direction  at TRISC3_bit;
sbit  LCD_D4_Direction  at  TRISC4_bit;
sbit  LCD_D5_Direction  at  TRISC5_bit;
```

```
sbit  LCD_D6_Direction  at  TRISC6_bit;
sbit  LCD_D7_Direction  at  TRISC7_bit;

int  res_ADC;      // переменная для хранения кода АЦП
int  mvolts;       // переменная для хранения напряжения в
                   // милливольтах
char  num;         // переменная для хранения цифр напряжения в
                   // вольтах
char  get_sb;      // переменная для хранения состояния контактов
                         // кнопок SB1, SB2
void  init( );             // прототип функции инициализации
void  out_U1( );   // прототип функции вывода на ЖКД напряжения U1
void  out_U2( );  // прототип функции вывода на ЖКД напряжения U2
void  out_invite( );       // прототип функции вывода на ЖКД
                           // текста приглашения
void  out_error( );        // прототип функции вывода на ЖКД
                           // текста сообщения об ошибке
void  main( )
{
    init( );                                  // вызов функции инициализации
    while(1)
    {
        get_sb = PORTB & 0b00000011;    // чтение линий порта И и
                                        // и выделение разрядов RB1, RB0
        switch( get_sb )
        {
            case  0b0000 0011 :         // обе кнопки SB1, SB2 не нажаты
                out_invite( );          // вывод на ЖКД текста приглашения
                break;
            case  0b0000 0010 :               // нажата кнопка SB1
                out_U1( );      // вывод на ЖКД результата измерения U1
                break;
            case  0b0000 0001 :               // нажата кнопка SB2
                out_U2( );      // вывод на ЖКД результата измерения U2
                break;
            default :
                out_error( );      // вывод на ЖКД сообщения об ошибке
        }
    }
```

```
}
void  init( )                  // функция инициализации
{
    ……                         // требуется написать самостоятельно для
    ……                    // инициализации линий портов МК, АЦП и ЖКД
}
void  invite( )              // функция вывода на ЖКД текста приглашения
{                            // требуется написать самостоятельно
    ……..                    // для вывода на ЖКД текста:
    ……..                    //       Press only one
}                            //       key:  SB1 or SB2
void  out_U1( )              // функция измерения напряжения U1 и вывода
{                            // результата на ЖКД
    ……..                    // требуется написать самостоятельно
    ……..
}
void  out_U2( )              // функция измерения напряжения U2 и вывода
{                            // результата на ЖКД
     ……                      // требуется написать самостоятельно
     ……
}
void  out_error( )           // функция вывода на ЖКД сообщения
                             // об ошибке
{                            // требуется написать самостоятельно
     ……                      // для вывода на ЖКД текста:
     ……                      //      Error! Two keys
}                            //      are pressed
```

3.3.2 Выполните разработку программы в следующей последовательности.

Запустите программу mikroC PRO for PIC. С помощью команды **New Project…** запустите New Project Wizard. Создайте новый проект с именем voltmeter в папке f:\ … \Lab5. Выберите микроконтроллер PIC16F877 и тактовую частоту 8 МГц. В окне редактора наберите текст исходной программы voltmeter.c и сохраните его в папке. Получите файл voltmeter.hex для загрузки в память микроконтроллера.

3.3.3 Для проверки работы МКУ запустите программу ISIS.exe пакета Proteus. Для упрощения разработки схемы можно использовать файл проекта с именем adc.dsn. С этой целью выберите пункт меню

File > Open Design. В раскрывшемся диалоговом окне найдите папку f:\ … \Lab5 и выберите из нее файл adc.dsn.

3.3.4 В схему из файла adc.dsn необходимо добавить кнопки. Их надо выбрать из библиотек Proteus по слову BUTTON. После выбора всех необходимых компонентов выполните соединение элементов между собой согласно принципиальной схеме, приведенной на рисунке 2. Затем сделайте надписи элементов схемы.

И в заключение необходимо сохранить разработанный проект. Для этого выберите пункт меню **File > Save Design As…** Раскройте папку f:\ ... \Lab5 и сохраните в ней проект под именем voltmeter.dsn.

3.3.5 Загрузите в микроконтроллер файл voltmeter.hex из папки f:\ … \Lab5. установите тактовую частоту 8 МГц.

Для проверки работы МКУ запустите проект на выполнение с помощью кнопки [Play]. На экране дисплея должен появиться текст:

Press only one

key: SB1 or SB2

Если это произошло, то нажмите и зафиксируйте кнопку SB1. На экране дисплея должны появиться значения напряжения $U1$ в вольтах.

Далее разомкните кнопку SB1, нажмите и зафиксируйте кнопку SB2. На экране дисплея должны появиться значения напряжения $U2$ в вольтах.

И, наконец, замкните и зафиксируйте обе кнопки: SB1 и SB2. На экране дисплея должно появиться сообщение об ошибке в виде:

Error! Two keys

are pressed

В заключение проверьте правильность измерения напряжений $U1$ и $U2$ при различных положениях движков потенциометров RV1 и RV2.

Остановите выполнение программы и закройте окно Proteus.

4 Содержание отчета

Наименование и цель работы. Текст программы и принципиальная схема МКУ к заданию для самостоятельной работы (комментарии в тексте программы обязательны!).

Контрольные вопросы

1 Сколько разрядов имеет выходной код АЦП микроконтроллеров PIC16?

2 Какие библиотечные функции используются для управления АЦП?

3 Как должны быть настроены линии порта МК, которые используются для ввода аналоговых сигналов?

4 Какие действия должна производить программа для выполнения АЦ-преобразования?

Литература

1 Шпак, Ю. А. Программирование на языке С для AVR и PIC микроконтроллеров / Ю. А. Шпак. – К. : МК-Пресс ; СПб. : КОРОНА-ВЕК, 2011. – 532 с.

2 Уилмсхерст, Т. Разработка встроенных систем с помощью микроконтроллеров PIC. Принципы и практические примеры / Т. Уилмсхерст ; пер. с англ. – К. : МК-Пресс ; СПб. : КОРОНА-ВЕК, 2008. – 544 с.

3 MikroC PRO for PIC. User's manual. – 2017. – Режим доступа: http://www.mikroe.com.

Содержание

Учебное электронное издание комбинированного распространения

Учебное издание

МИКРОПРОЦЕССОРНАЯ ТЕХНИКА

Практикум
по выполнению лабораторных работ
для студентов специальности
1-36 04 02 «Промышленная электроника»
заочной формы обучения

Составитель **Виноградов** Эдуард Михайлович

Электронный аналог печатного издания

Редактор *О. С. Ковалёва*
Компьютерная верстка *Н. Б. Козловская*

Подписано в печать 16.03.22.
Формат 60x84/16. Бумага офсетная. Гарнитура «Таймс».
Цифровая печать. Усл. печ. л. 3,95. Уч.-изд. л. 3.
Изд. № 1.
http://www.gstu.by

Издатель и полиграфическое исполнение
Гомельский государственный
технический университет имени П. О. Сухого.
Свидетельство о гос. регистрации в качестве издателя
печатных изданий за № 1/273 от 04.04.2014 г.
пр. Октября, 48, 246746, г. Гомель