ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
СРЕДНЕГО (ПОЛНОГО) ОБЩЕГО ОБРАЗОВАНИЯ
ЛИЦЕЙ ПРИ СПБГУТ

Вендор-ориентированный учебный курс в системе
«Старшая профильно-профессиональная школа-ВУЗ-Работодатель»:
«Программирование микроконтроллеров Microchip»

Богураев М.В.

«УПРАВЛЕНИЕ ПОРТОМ ВВОДА-ВЫВОДА»

Методические указания к выполнению
лабораторной работы

Санкт - Петербург
2013

Богураев М.В. «УПРАВЛЕНИЕ ПОРТОМ ВВОДА-ВЫВОДА». Методические указания к выполнению лабораторной работы № 2. СПб: ГОУ «Лицей при СПбГУТ», 2013.

ЛАБОРАТОРНАЯ РАБОТА №2
«УПРАВЛЕНИЕ ПОРТОМ ВВОДА-ВЫВОДА»

Цель работы

Научиться управлять портами ввода-вывода микроконтроллеров PIC на СИ с использованием компилятора HI-TECH C. Научиться симулировать работу программы на СИ для микроконтроллера в среде разработки MPLAB IDE.

Теоретические основы

Текст программы обязательно должен содержать описание функции main{}, подключение библиотечного файла htc.h, и слово конфигурации заданное директивой __CONFIG().

Внутри функции main нужно инициализировать микроконтроллер – настроить регистры (ячейки памяти данных) TRISx, разрешить или запретить прерывания, настроить маску прерываний, настроить периферийные модули. После этого можно писать программу.

Язык СИ не связан командами микроконтроллера. Перевод языковых конструкций в последовательность команд автоматизирован программой – компилятором. Программа – компилятор анализирует синтаксические ошибки текста программы; затем генерирует машинные коды в соответствии с текстом программы и размещает переменные в памяти данных.

Порт ввода-вывода микроконтроллера состоит из двух регистров: TRISx, и PORTx. Регистр TRISx определяет направление поступления информации. Информация может поступать в контроллер, или выводиться из контроллера. Если TRISx настроен на выход, то запись в PORTx приводит к появлению информации на ножках микросхемы. Большинство портов микроконтроллера восьмиразрядные. Разрядность порта ввода-вывода определяется количеством доступных бит в регистрах PORTx и TRISx. Один бит регистра TRISx управляет одним битом регистра PORTx. Так, например, для настройки всех разрядов порта PORTD на выход, в регистр TRISD должно быть занесено значение «все восемь бит – нули». Если планируется использовать на выход только один разряд (бит) порта, достаточно установить только один ноль в регистре TRISx. Например, если сбросить третий бит регистра TRISD (TRISD,3 = 0), то третий бит порта PORTD (PORTD,3) будет работать на вывод информации и передавать ноль или единицу из третьего бита PORTD на ножку корпуса микросхемы.

Чтобы на выходе PORTD появилась информация, порт должен быть настроен на выход, а в регистр PORTD нужно записать выводимую информацию. Функцию настройки порта на ввод или вывод выполняет регистр TRISD. Информация помещается в регистр PORTD. На языке СИ для этого достаточно двух конструкций: TRISD = 0x00; PORTD = 0x01; Программисту не надо заботиться об адресах в памяти, всё это автоматически сделает компилятор языка СИ. Через библиотечный файл htc.h подключается файл pic168xa.h, в котором содержатся описания всех символических имён, используемых программистом. Эти имена адресуют ячейки памяти данных и совпадают с документацией на микроконтроллер. Если компилятор был установлен на диск по умолчанию, тогда указанные файлы доступны по адресу C:\Program Files\HI-TECH Software\PICC\LITE\9.xx\include

Компилятор СИ проверяет только синтаксис программы. Логика программы должна быть продумана и проверена программистом. Компилятор не определяет логические ошибки программы, поэтому синтаксически верная программа может не работать или работать неправильно из-за нарушения логики. По этим причинам перед написанием любой программы нужно изобразить алгоритм работы микроконтроллера.

Задание

На диске C в папке Projects создайте папку Project2. В эту папку скопируйте файл Project2.c. Откомпилируйте проект.

Первый этап: PORTD настроен на выход. Промоделируйте в среде MPLAB IDE работу микроконтроллера. После моделирования соедините PORTD со светодиодами. Запрограммируйте лабораторный макет и предъявите результат.

Второй этап: PORTD настроен на выход, а PORTB,0 настроен на вход. Промоделируйте работу микроконтроллера. После моделирования соедините на макете PORTB0 с кнопкой, а PORTD присоедините к светодиодам. Запрограммируйте лабораторный макет и предъявите результат.

Порядок выполнения

Первый этап. На диске C:\ в папке Projects создайте папку Project2. В папку Project2 скопируйте файл Project2.c. Запустите программу MPLAB IDE и создайте проект на СИ в этой папке. Откомпилируйте проект, нажав на красную кнопку (рис. 1).

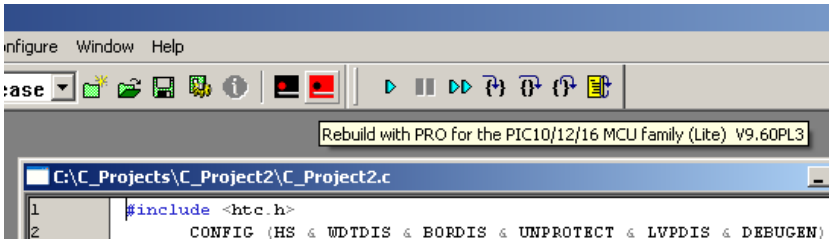

Рис. 1. Компиляция проекта.

Если компиляция прошла успешно, то в окне output не будет сообщений об ошибках (рис. 2).

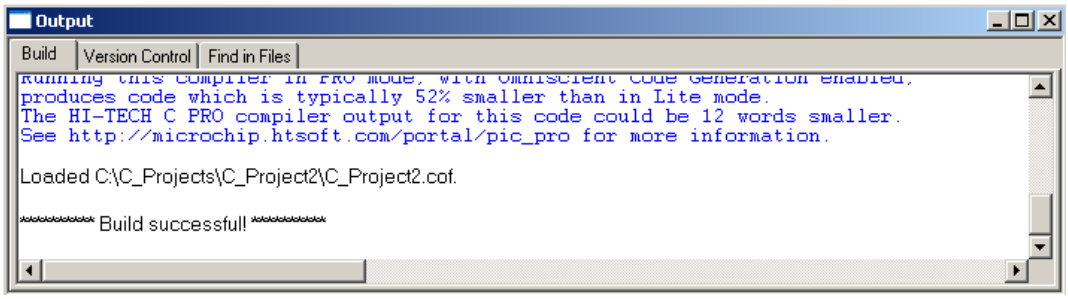

Рис. 2. Окно Output после успешной компиляции.

Откройте окно Watch (View/Watch). В окно Watch в выпадающем списке выберите регистр TRISD, затем нажмите Add SFR (рис. 3) в окне появится регистр TRISD. Аналогичным образом добавьте PORTD. Вид настроенного окна на рис. 4.

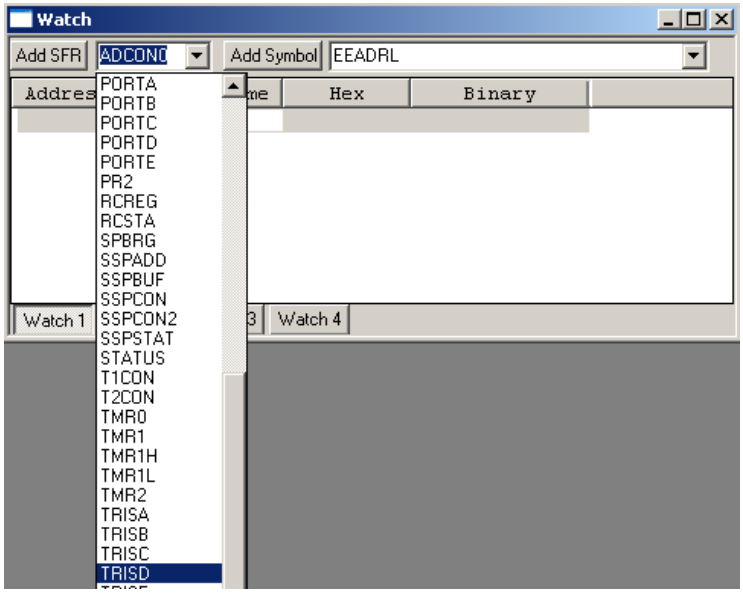

Рис. 3. Вывод регистра TRISD в окно Watch.

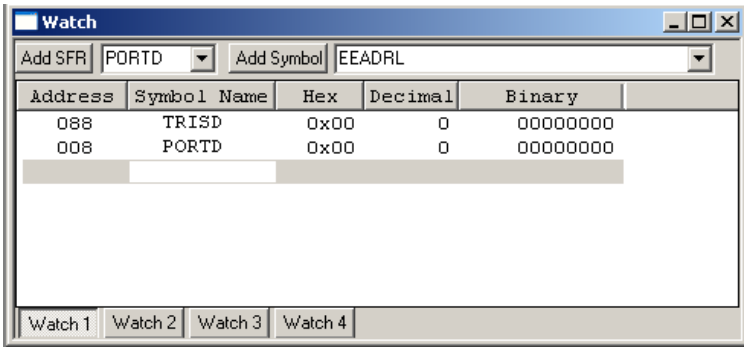

Рис. 4. Настроенное окно Watch.

Выберите отладчик MPLAB SIM (Debugger/Select Tool/MPLAB SIM – рис. 5). Настройте параметры симуляции. Для этого откроем окно настройки параметров (выберем Debugger/Settings – рис. 6)

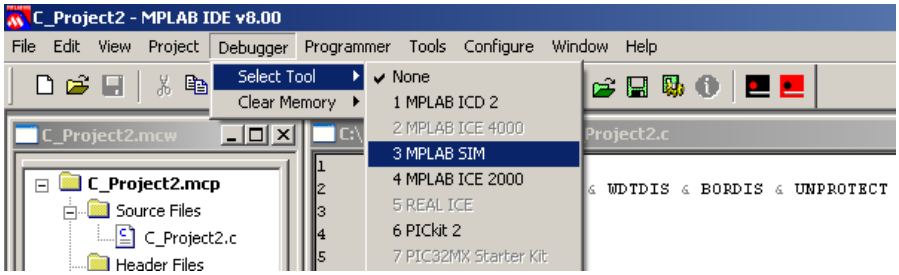

Рис. 5. Выбор симулятора MPLAB SIM.

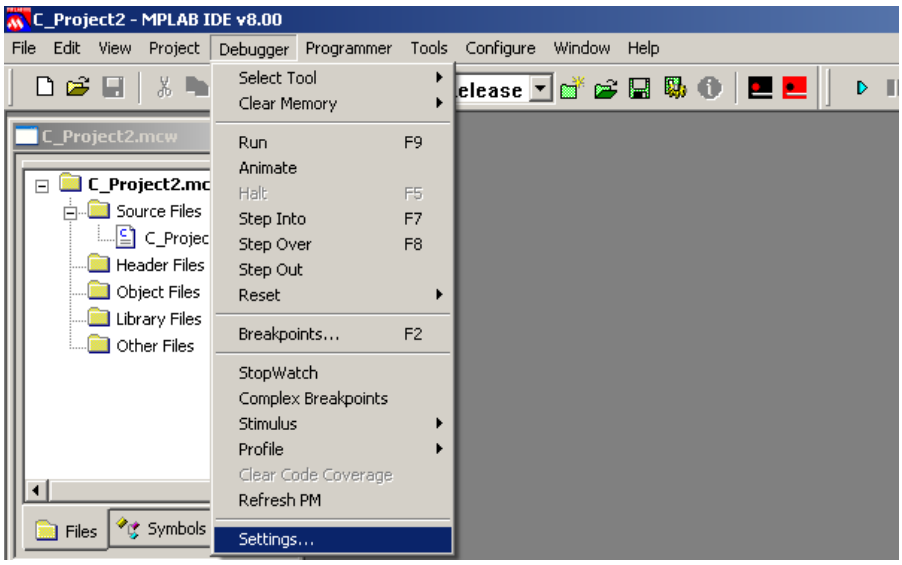

Рис. 6. Настройка параметров симулятора.

В появившемся окне выберем вкладку Animation/Realtime Updates и поставим галочку напротив Enable Realtime watch updates (рис. 7)

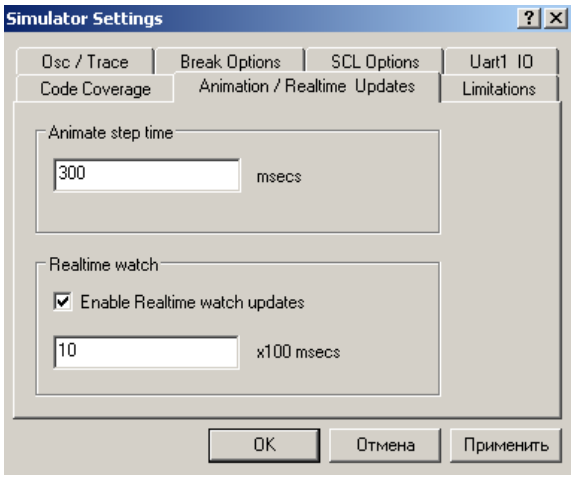

Рис. 7. Окно настройки симулятора.

Выполним программу по шагам (рис. 8), наблюдая за изменением содержимого регистров TRISD и PORTD. Проанализируйте изменения в окне Watch.

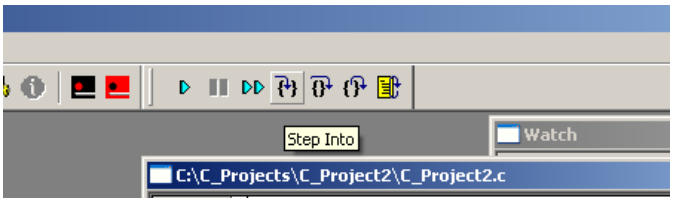

Рис. 8. Пошаговая отладка.

После моделирования соедините PORTD со светодиодами. Запрограммируйте лабораторный макет и предъявите результат.

Второй этап. Промоделируем работу микроконтроллера, когда PORTD настроен на выход, а PORTB,0 настроен на вход. Для этого отредактируем текст исходной программы. В начале функции main запишем конструкцию TRISB = 0x01 – это настроит PORTB,0 на вход. Исключим все настройки PORTD. После этого добавим бесконечный цикл while (1). Внутри бесконечного цикла добавим конструкцию if … else, которая в зависимости от наличия единицы на входе RB0, будет изменять содержимое PORTD. Получается такая программа:

```
#include <htc.h>
__CONFIG (HS & WDTDIS & BORDIS & UNPROTECT & LVPDIS & DEBUGEN);
void main (void)
{
        TRISD = 0x00;
        TRISB = 0x01;
        while (1)
        {
                if (0 == RB0)
                PORTD = 0x0F;
                else
                PORTD = 0xF0;
        }
}
```

Обратите внимание на условие if (0 == RB0). В языке СИ двойное равно – это оператор сравнения, тогда как одиночное равно – это оператор присваивания, не перепутайте!

Откомпилируем новую программу (рис. 1). После компиляции создадим файл стимулов. Для этого нужно выбрать Debugger/Stimulus/New Workbook (рис. 9).

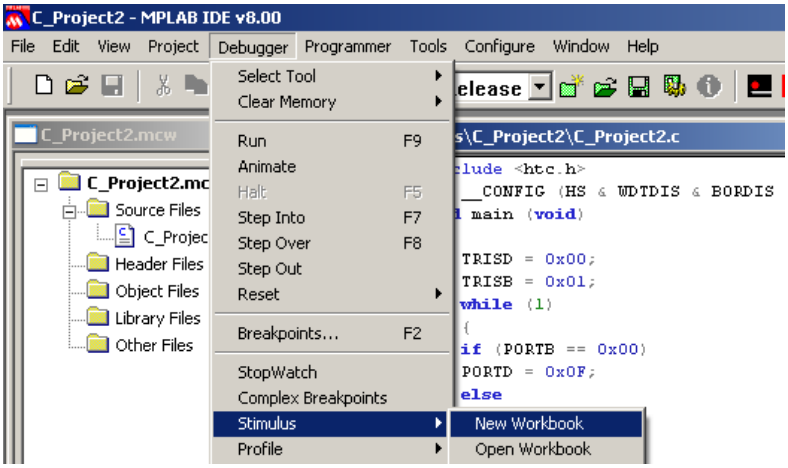

Рис. 9. Создание файла стимулов.

В появившемся окне откроем вкладку Asynch. Щёлкнем левой кнопкой мыши на столбце Pin/SFR и выпадающем списке выберем RB0 (рис. 10).

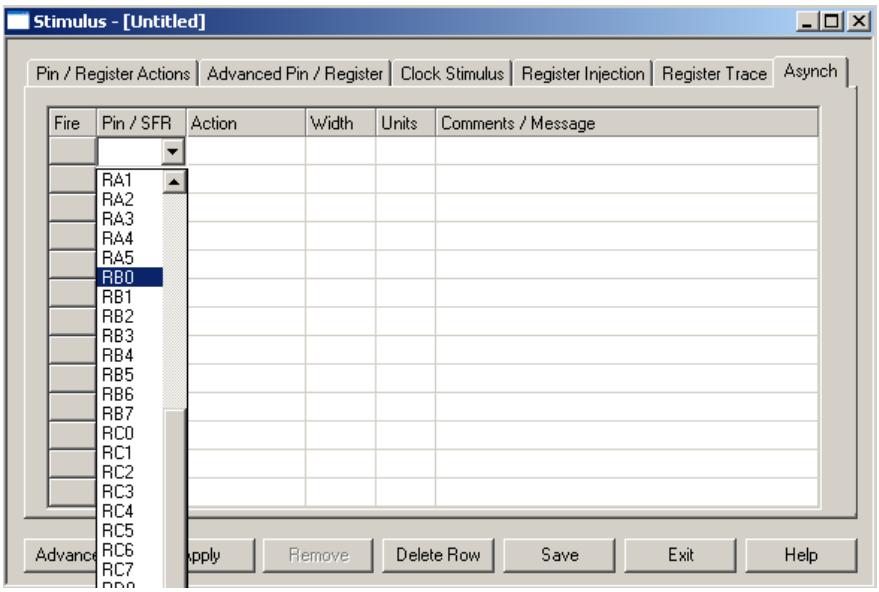

Рис. 10. Выбор вывода RB0 для моделирования.

В столбце Action (тип воздействия) выберем Toggle – переключатель (рис. 11).

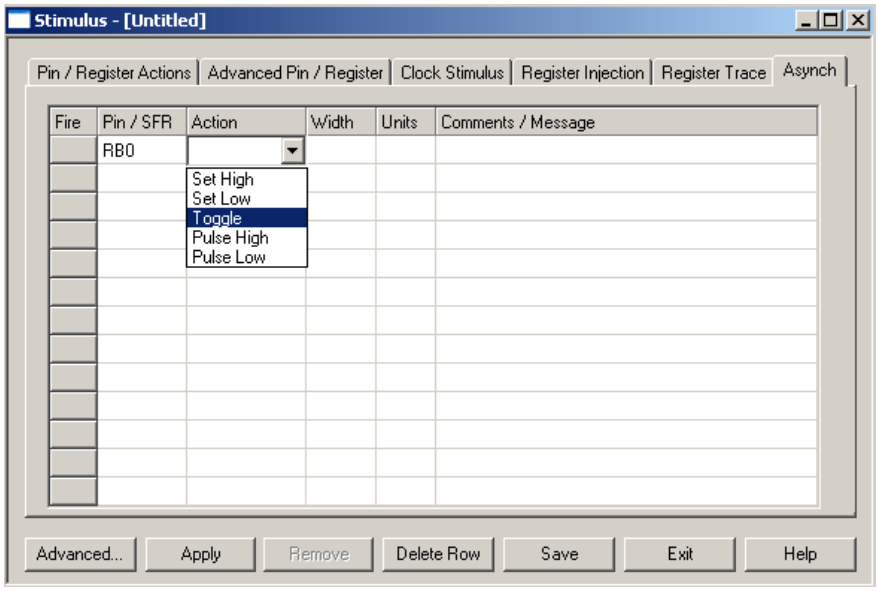

Рис. 11. Выбор типа воздействия Toggle – переключатель.

Убедитесь, что в симуляторе включена опция Enable Realtime watch updates (поставлена галочка рис. 7). Запустите программу, нажав кнопку RUN (рис. 12).

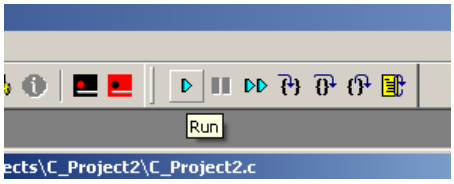

Рис. 12. Кнопка RUN, которая запускает симуляцию.

Для переключения уровня на RB0 нажимайте кнопку «>» в графе Fire. После каждого нажатия в окне Output появляется сообщение (рис. 13).

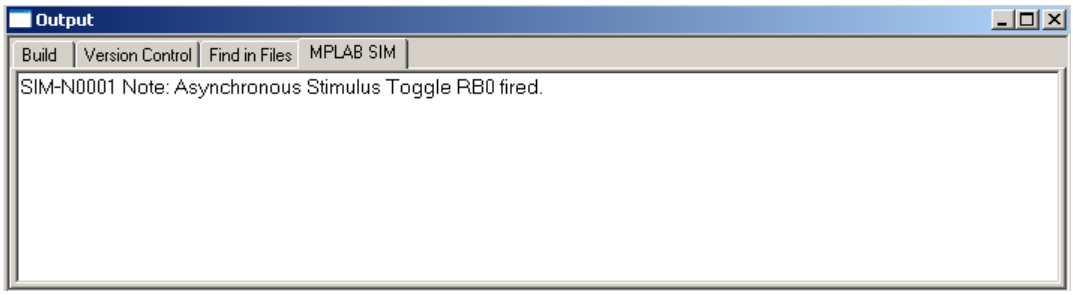

Рис. 13. Сообщение о применении стимулов.

В окне Watch наблюдайте содержимое PORTD в зависимости от уровня на PORTB,0. Дайте оценку происходящим изменениям.

Соберите аппаратное обеспечение – соедините восемь диодов с PORTD, а кнопку с PORTB0 (рис. 18). Затем в среде разработки выберите имеющийся программатор, например, PICkit 2 (рис 14).

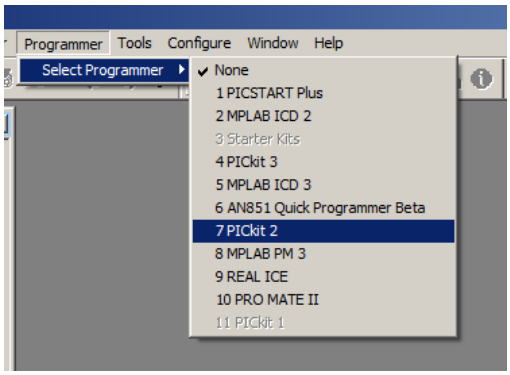

Рис. 14. Выбор программатора PICkit2

Запрограммируйте лабораторный макет, для чего нажмите кнопку программирования микроконтроллера (рис. 15)

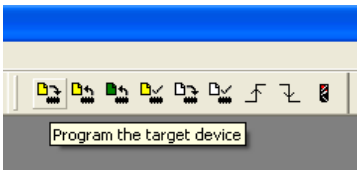

Рис. 15. Кнопка программирования микроконтроллера.

Рис. 16. Кнопка вывода микроконтроллера из состояния Reset.

Нажмите кнопку вывода микроконтроллера из состояния Reset (Рис. 16). Предъявите результат.

Аппаратное обеспечение

Эта работа выполняется и на компьютере в среде разработки MPLAB IDE, и на макете. Принципиальная схема изображена на рис. 17. Схема, собранная на макете, изображена на рис. 18.

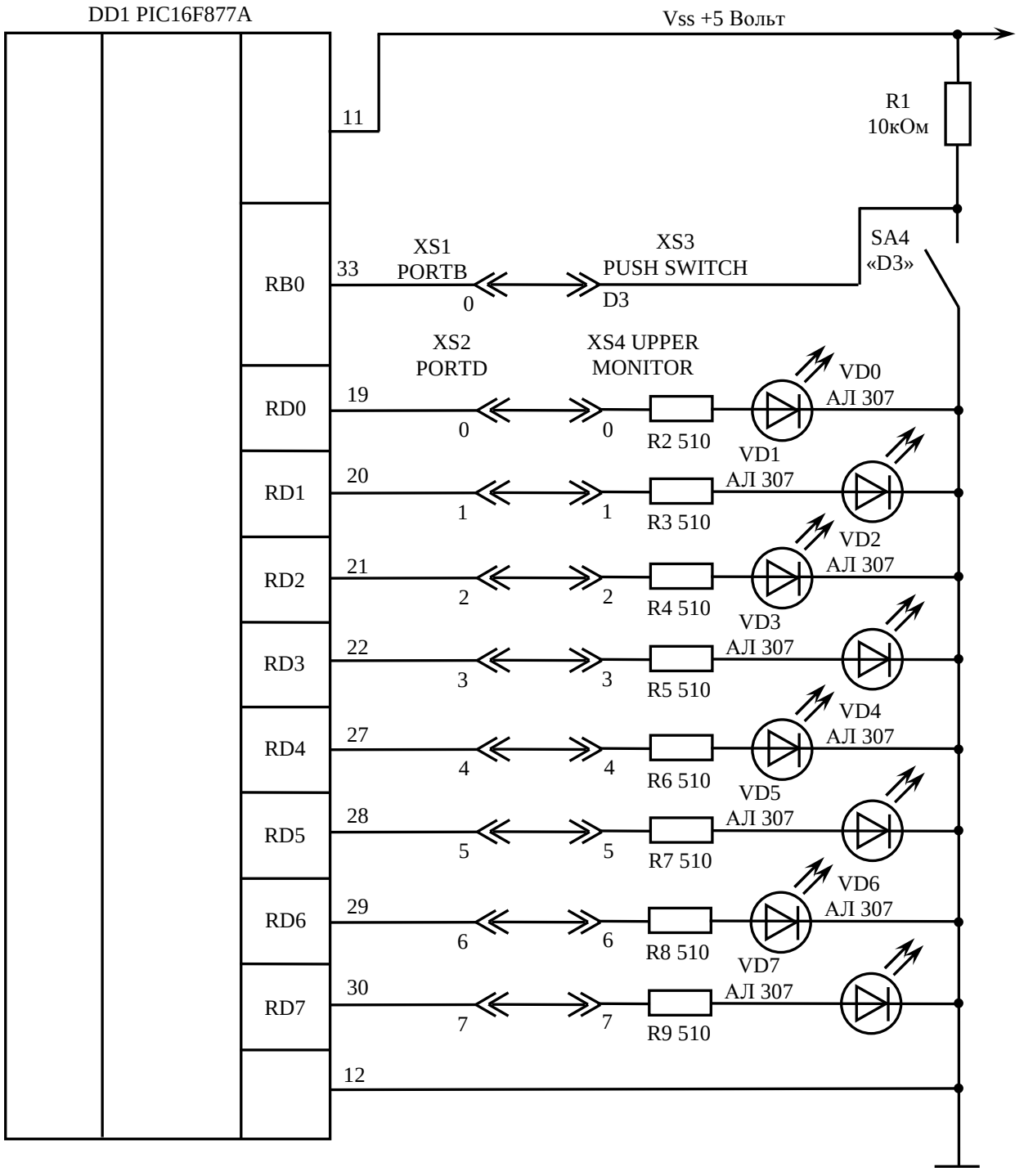

Рис. 17. Схема электрическая принципиальная.

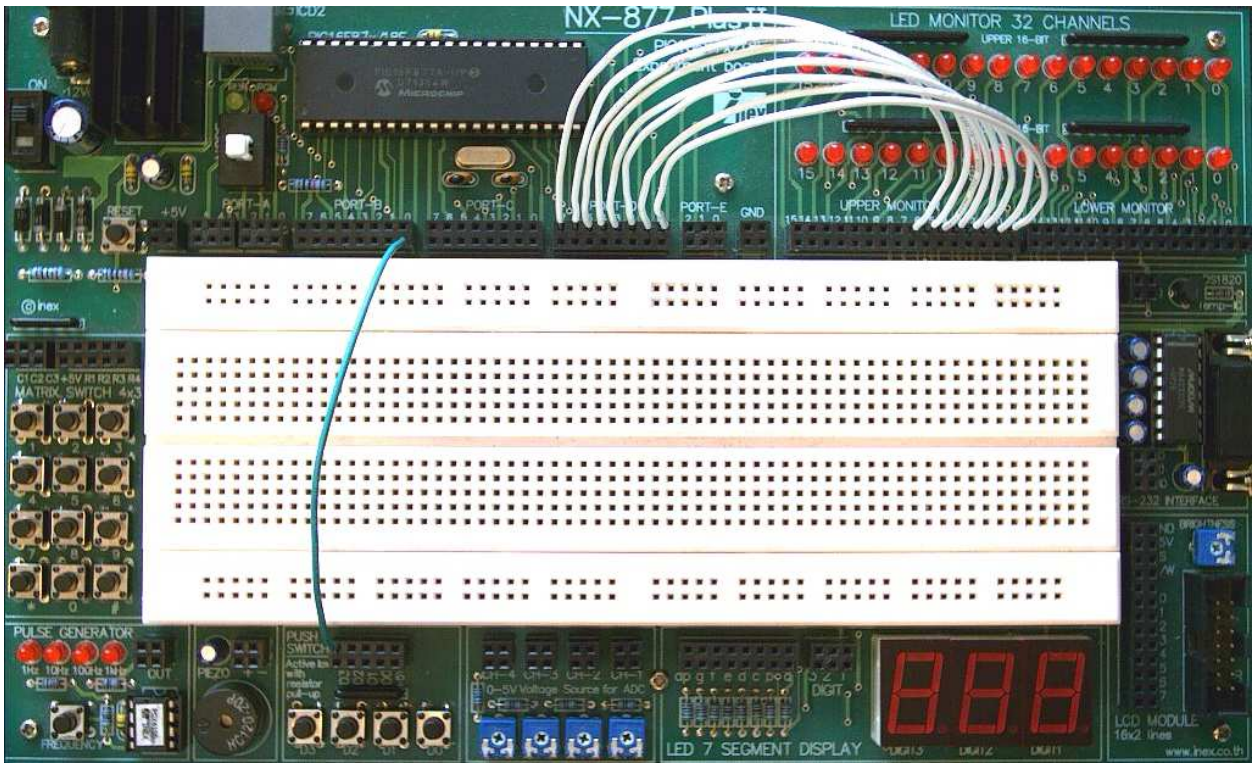

Рис. 18. Схема, собранная на макете.

Программное обеспечение

Текст файла Project2.c

```
#include <htc.h>
__CONFIG (HS & WDTDIS & BORDIS & UNPROTECT & LVPDIS & DEBUGEN);
void main (void)
{
                    TRISD = 0x00;
                    PORTD = 0x0F;
                    PORTD = 0xF0;
}
```

Индивидуальные задания

В главном меню выберите View и откройте окно Program Memory. Поясните содержимое памяти команд.

Контрольные вопросы

1. Что должно обязательно содержаться в тексте программы?
2. Связан ли язык СИ командами микроконтроллера?
3. Из каких регистров состоит порт ввода – вывода?
4. Какова функция регистров TRISx?
5. Какова функция регистров PORTx?
6. В каком файле описываются символические имена, используемые программистом?
7. Определяет ли компилятор логические ошибки программы?
8. Почему перед написанием программы нужно изобразить алгоритм?
9. В какой регистр нужно записывать выводимую информацию, и как настроить порт на вывод?

Оглавление:

ЛАБОРАТОРНАЯ РАБОТА №2 «УПРАВЛЕНИЕ ПОРТОМ ВВОДА-ВЫВОДА» 3
Цель работы 3
Теоретические основы 3
Задание 4
Порядок выполнения 4
Аппаратное обеспечение 10
Программное обеспечение 12
Индивидуальные задания 12
Контрольные вопросы 12

Список рисунков:

Рис. 1. Компиляция проекта. 4
Рис. 2. Окно Output после успешной компиляции. 4
Рис. 3. Вывод регистра TRISD в окно Watch. 5
Рис. 4. Настроенное окно Watch. 5
Рис. 5. Выбор симулятора MPLAB SIM. 5
Рис. 6. Настройка параметров симулятора. 6
Рис. 7. Окно настройки симулятора. 6
Рис. 8. Пошаговая отладка. 6
Рис. 9. Создание файла стимулов. 7
Рис. 10. Выбор вывода RB0 для моделирования. 8
Рис. 11. Выбор типа воздействия Toggle – переключатель. 8
Рис. 12. Кнопка RUN, которая запускает симуляцию. 8
Рис. 13. Сообщение о применении стимулов. 9
Рис. 14. Выбор программатора PICkit2 9
Рис. 15. Кнопка программирования микроконтроллера. 9
Рис. 16. Кнопка вывода микроконтроллера из состояния Reset. 9
Рис. 17. Схема электрическая принципиальная. 10
Рис. 18. Схема, собранная на макете. 11