Московский государственный технический университет
имени Н.Э. Баумана

Васильев И.А.

Лабораторная работа по моделированию

микроконтроллера PIC 16 F84A.

Москва 2012г.

1. Теоретическая часть работы.

Микроконтроллер это микропроцессорное устройство, предназначенное для управления внешними исполнительными механизмами и опрашивания их состояния. Для осуществления связи с внешними исполнительными механизмами (устройствами) контроллер имеет несколько портов ввода-вывода данных. Управляющая программа считывает данные о состоянии внешних устройств с порта ввода данных и посылает управляющие сигналы на внешние устройства в порт вывода (управление по прерыванию в данной работе не рассматривается).

Как правило, порт может быть запрограммирован на ввод или вывод данных по каждому из выводов порта по флагам в регистре управления портом. Таким образом, порт представляет собой двунаправленную шину данных с разрядностью равной разрядности соответствующего порта.

Модель порта в однобитном исполнении изображена на рисунке 1.

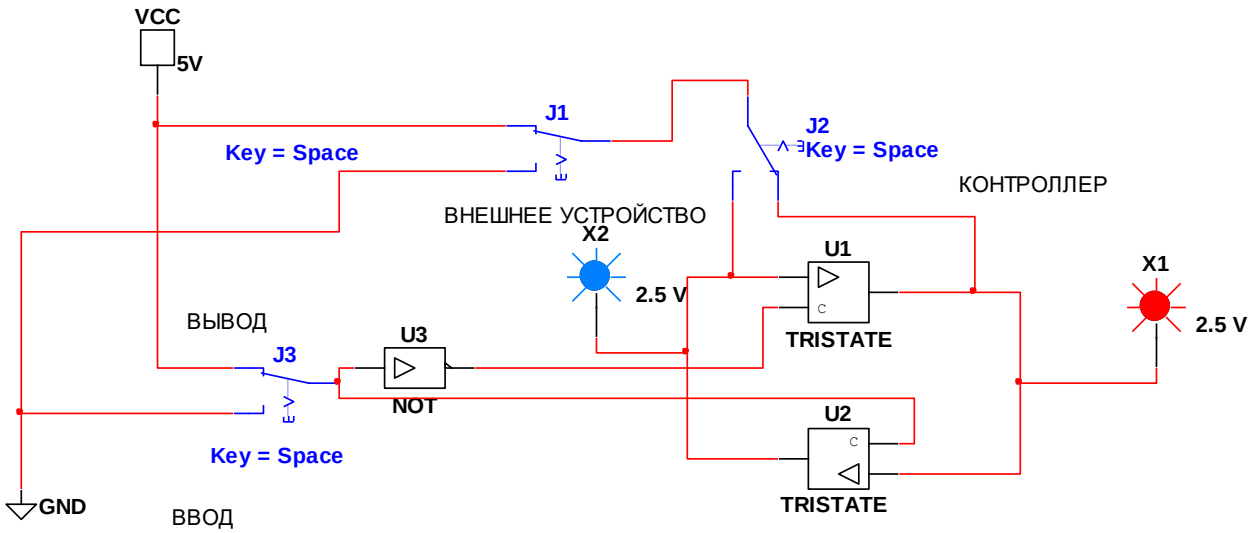

Рисунок 1- Модель однобитного порта ввода-вывода.

На рисунке 1 буферные (без инверсии) усилители U1 и U2 могут быть переведены в “Z” состояние по потенциалу управляющего входа “C”. При низком уровне управляющего входа усилитель переходит в состояние высокого импеданса по выходу и не передает сигнал от входа к выходу и не создает помехи в работе другого усилителя. При высоком уровне управляющего входа усилитель переходит в режим передачи сигнала от входа к выходу. Для устранения режима одновременного включения усилителей управляющий вход усилителя U1 подключен к переключателю выбора режим J3 через инвертор U3.

При выборе режима ВЫВОД (J3 в верхнем положении) усилитель U2 активен, а U1 пассивен (в “Z” состоянии). Источником сигнала является микроконтроллер (J2 переключаем в позицию МИКРОКОНТРОЛЛЕР). Внешнее устройство принимает сигнал микроконтроллера (индикаторы работают синхронно).

При выборе режима ВВОД (J3 в нижнем положении) усилитель U2 пассивен (в “Z” состоянии), а U1 активен. Источником сигнала является внешнее устройство (J2 переключаем в позицию ВНЕШНЕЕ УСТРОЙСТВО). Внешнее устройство передает сигнал микроконтроллеру (индикаторы работают синхронно). При неправильном выборе источника сигнала индикаторы не будут работать синхронно**.**

В микроконтроллере переключение режимов ввода и вывода выполняется установкой битов в регистре управления портом.

На рисунке 2 изображена схема включения микроконтроллера для моделирования выполнения домашнего задания №2.

Микроконтроллер имеет два порта ввода-вывода. Это порт A (выводы микросхемы 17, 18, 1,2,3) и порт B (выводы микросхемы с 6 по 13 и 15). Порты имеют имена (идентификаторы) PORTA и PORTB. Для вывода числа k в порт используется оператор

PORTA = k;

Для чтения данных dat оператор

dat = PORTA;

Порты – программно доступные регистры по их именам, через них управляющая программа может прочитать данные с внешнего устройства или выставить управляющие сигналы на внешнее устройство.

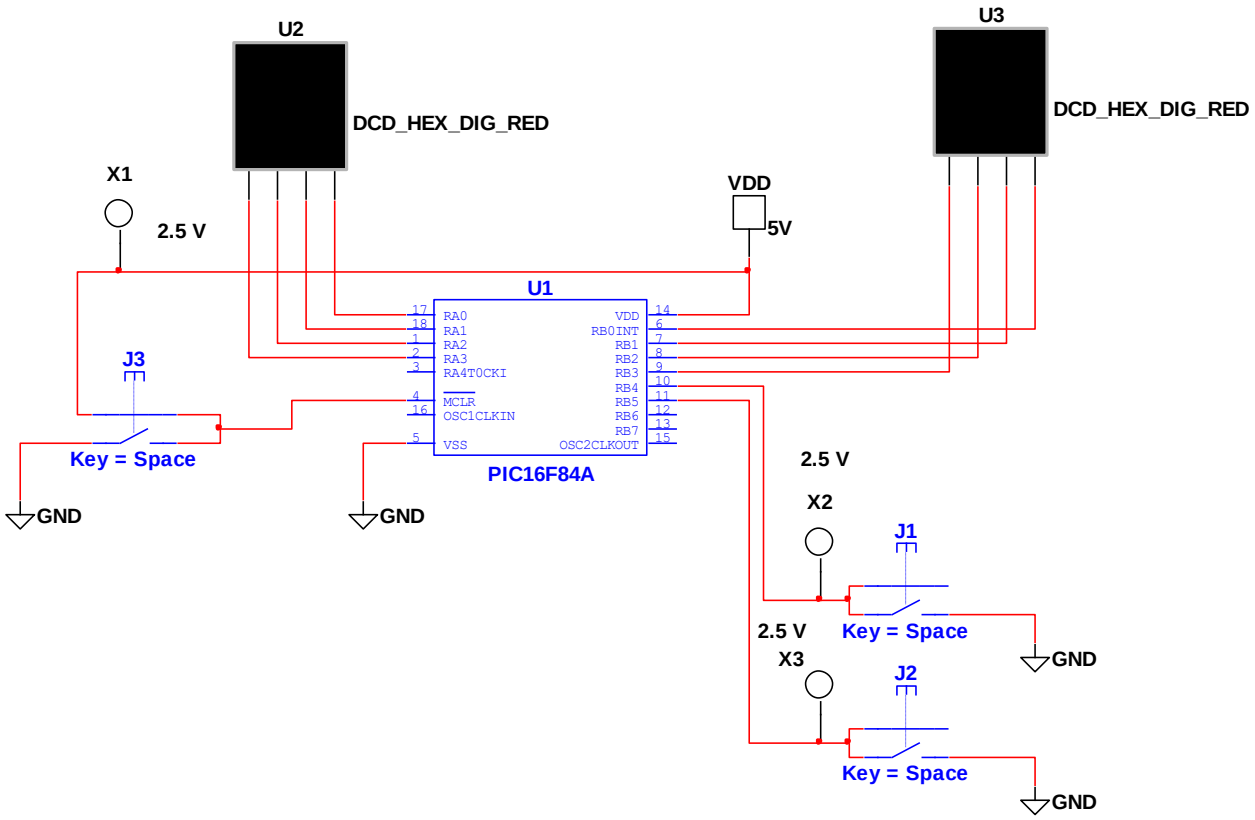

Рисунок 2 – Схема микроконтроллера для выполнения задания № 2.

К микросхеме котроллера подключено питание (здесь нет режима питания по умолчанию). Кнопка J3 осуществляет общий сброс процессора. Дисплей U2 предназначен для вывода чисел домашнего задания. Дисплей U3 предназначен для вывода номера импульса. Кнопка J1 предназначена для старта работы автомата, а кнопка J2

предназначена для его остановки. Листинг программы управления автоматом приведен ниже

```
#include<htc.h> //команды препроцессора - подключение макросов и библиотек

//prototype functions

void Pause (int);//прототип функции паузы

void Inc_Port (void); // прототип функции инициализации порта

//end prototype functions


void main()
	{
		int n,i,j; //описание целых переменных

		int dat[8] = {1,4,3,7,6,5,0,2};//домашнее задание

		j=8; //число импульсов дом. зад.

		n = 200; //параметр для функции паузы

		Inc_Port ();//инициализация порта
```

```
lb1:    while (RB4 == 1); //ожидание флага по кнопке J1

            while (1)     //бесконечный цикл вывода домашнего задания №2

                {
                    for (i=0; i < j; i++)//конечный цикл по числу импульсов д.з.№2
                    {

                        PORTA = dat[i]; //вывод на дисплей U2 чисел д.з.

                        PORTB = i; // вывод на дисплей U3 номера импульса

                        Pause (n); //пауза

        while (RB5 == 0) // флаг по кнопке J2 пока кнопка не нажата игнорируется
                        {
                            goto lb1; //
                        }

                    }

                }

                }
```

```
void Pause (int n) //функция паузы

{
        int k; // внутренний параметр паузы

        for( k = 0; k < n ; k++) //цикл по внешнему параметру паузы

        {
                k++;//инкремент
                k--;//декремент
        }

}

void Inc_Port (void) //функция инициализации портов
{

TRISA = 0x00;//port A output установка флагов на вывод

TRISB = 0xF0;//portB B7-B4 input B4-B0 output

RBPU =          1; // knopka “1” input pri razomknutom kluche

}
```

Оператор

int dat[8] = {1,4,3,7,6,5,0,2};//домашнее задание

предназначен для инициализации значений массива данных домашнего задания. Число чисел домашнего задания №2 указывается в индексных скобках в имени массива. Значению параметра j нужно так же присвоить значение числа цифр задания. Программа управляется двумя внешними устройствами (кнопками J1 и J2 рисунок 2). Микроконтроллер должен считывать данные с кнопок и функционировать в соответствии с внешними командами, посылаемыми оператором с пульта управления. Такой тип управления называется управлением по ожиданию значения флага, а соответствующие биты называются флагами.

Для того, чтобы биты порта могли быть включены на ввод или вывод данных используется регистр управления портом. **Имена регистров не являются элементами языка СИ и зависят от применяемого микроконтроллера.**

Для данного микроконтроллера для управления портом B используется регистр с именем TRISB, а портом A с именем TRISA.

```
void Inc_Port (void) //функция инициализации портов
        {
        TRISA = 0x00;//port A output установка флагов на вывод
        TRISB = 0xF0;//portB B7-B4 input B4-B0 output
        RBPU =      1; // knopka “1” input pri razomknutom kluche
        }
```

В микроконтроллере каждый бит порта может быть настроен (сконфигурирован) на ввод или вывод данных по значению соответствующих битов в регистре управления. Если в

бите с номером N регистра управления установлен 0 то соответствующий бит порта будет открыт на вывод данных, а при 1 на ввод данных (чтение).

При чтении данных допускается побитный ввод данных. Идентификаторы RB4 и RB5 в листинге программы означают побитное чтение четвертого и пятого битов порта B. Таким образом, RB4 или RB5 каждый равен нулю или единице в зависимости от того нажата или отпущена соответствующая кнопка управления (J1 или J2).

Оператор

```
 RBPU =        1; // knopka “1” input pri razomknutom kluche
```

выполняет подключения резистора порта (подтягивающего резистора) к источнику питания. Это позволяет иметь “уверенную” единицу на входе бита порта при разомкнутой кнопке.

Первый исполняемый оператор

```
lb1:     while (RB4 == 1); //ожидание флага по кнопке J1
```

имеет метку lb1:.

Оператор цикла while (RB4 == 1); будет выполняться до смены истинности выражения в скобках. При разомкнутой кнопке J1 считывается единица с четвертого бита порта B и RB4 равно единице при замыкании кнопки RB4 равно нулю выражение RB4==1 становиться ложным и программа переходит к исполнению следующего оператора. Микроконтроллер приступает к циклическому выводу данных и номера импульса.

Программа

```
void Pause (int n) //функция паузы
```

необходима для устранения быстрого мелькания дисплея и принципиального значения не имеет.

Второй оператор цикла предназначен для остановки процесса вывода данных. При отпущенной кнопке значение RB5 равно единице. Исполняемый оператор игнорируется и цикл вывода данных продолжается.

```
while (RB5 == 0) // флаг по кнопке J2 пока кнопка не нажата оператор игнорируется
{
        goto lb1; //
}
```

После нажатия кнопки J2 (её нужно удерживать некоторое время, так как в цикле есть функция паузы) программа переходит на метку lb1 и контроллер ожидает нажатие кнопки J1.

2.Прядок выполнения лабораторной работы

2.1. Загрузите программу моделирования порта ввода-вывода “Модель_порта”. Положением переключателя J1 (рисунок 1) выбираем направление передачи информации и определяем передатчик и приемник сигнала управления. Поясните полученный результат. **Выключите исполнение программы “Модель_порта”.**

2.2. Загрузите программу “MCU_DZ_2”.

В главном меню войдите MCU далее MCU PIC 16… далее MCU Code Manager нажать левую кнопку мыши. Появиться окно. В окне находим закладку “main.c”. На изображении закладки дважды нажимаем левую кнопку мыши. Появиться окно ввода кода управляющей программы. В окне MCU Code Manager нажмите кнопку ОК. Произведите редактирование кода в соответствии с Вашим домашним заданием №2. Введите данные домашнего задания в операторе присваивания значения массива. Например, если задание имеет вид (1,4,7,6,3) то число состояний (тактовых импульсов) равно 5 и строчка ввода данных будет иметь вид

```
int dat[5] = {1,4,7,6,3};//домашнее задание

j=5; //число импульсов дом. зад.
```

Таким образом, редактируется две строки программы. После этого производим компиляцию нового кода для создания исполняемого кода микропроцессора. В левом

верхнем углу нажимаем стрелочку в виде зеленого треугольника. Появиться запрос на изменения кода. Подтверждаем наши действия. Убеждаемся в нижнем окне в отсутствии предупреждений и ошибок. После этого, в левом нижнем углу переходим к закладке MCU_DZ_2 и выходим на схему рисунок 2. Программа находиться в режиме ожидания нажатия кнопки J1.

Нажимаем кнопку J1 и убеждаемся в правильности чередования выводимых цифр и соответствии их номерам импульса. Останавливаем процесс вывода нажатием кнопки J2. Кнопку удерживаем до остановки процесса. Вновь запускаем программу и останавливаем процесс вывода кнопкой J3. **J3 нажимаем кратковременно.** Убеждаемся в быстром срабатывании кнопки J3.

Вновь запускаем программу и пытаемся остановить процесс вывода кнопкой J2 кратковременно её нажимая сразу после смены данных на дисплее. Убеждаемся в отсутствии управляющего действия.

Выключаем исполнение программы MCU_DZ_2.

2.3.Переходим на “белый лист” и создаем новый проект.

В строке инструментов находим Place MCU (изображена микросхема с многими выводами). Выбираем контроллер PIC и устанавливаем его на листе. После установки появиться окно присоединения кода к данной микросхеме. Вводим имя (произвольно несколько букв) и далее дважды Next.

Далее собираем схему изображенную на рисунке 3.

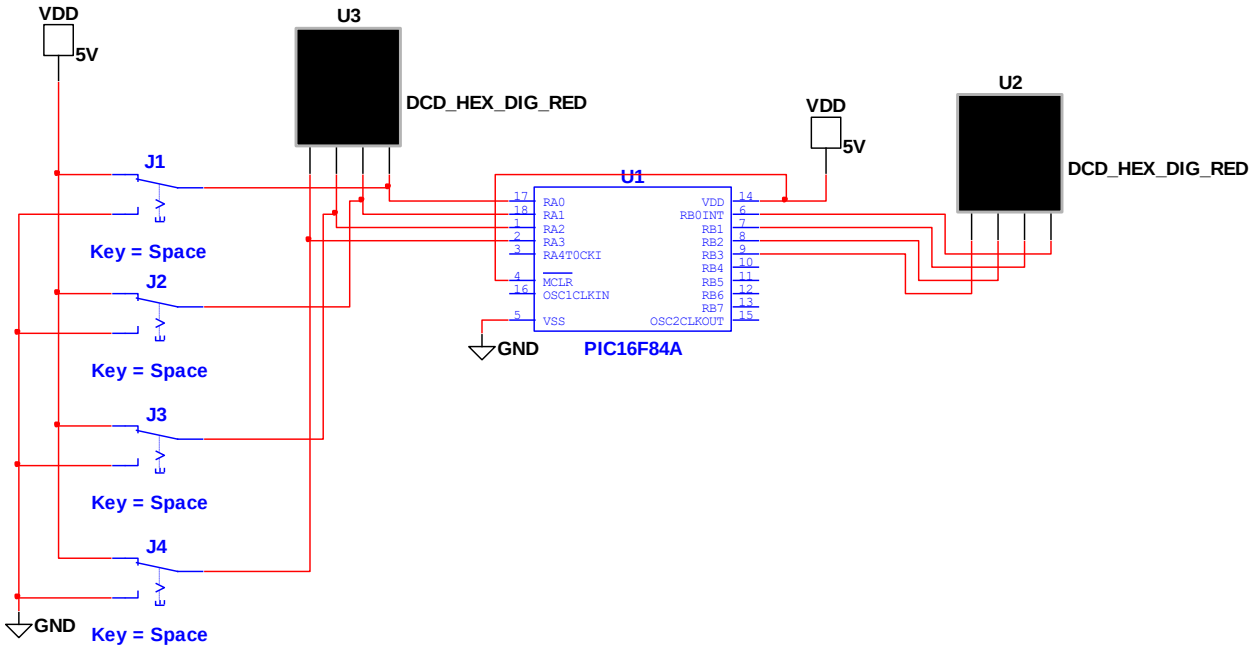

Рисунок 3 - Новый проект.

В главном меню войдите MCU далее MCU PIC 16… далее MCU Code Manager нажать левую кнопку мыши. Появиться окно. В окне находим закладку “main.c”. На изображении закладки дважды нажимаем левую кнопку мыши. Появиться окно ввода кода управляющей программы. В окне MCU Code Manager нажмите кнопку ОК.

После сборки схемы преступаем к созданию нового проекта управляющей программы.

```
#include<htc.h>

void main()
{
        int i;

        TRISA = 0xFF;//port A input

        TRISB = 0x00;//port B output
```

```
while(1)
{
   i = PORTA;
         PORTB = i;
}
}
```

Данная программа читает биты кнопок ввода данных в порт А и выводит данные в пот В.

3.Вопросы по теме лабораторной работы

3.1. Для чего предназначены порты ввода-вывода.

3.2. Как переключить направление передачи данных в порте ввода-вывода. Для чего служит регистр управления портом.

3.3. Как осуществляется управление программой по ожиданию флага.

3.4. Приведите примеры управления программой по ожиданию флага из листинга программы “MCU_DZ_2”. Дайте пояснения.

3.5. Поясните действие программы ввода и вывода (последний листинг). Выпишите и дайте комментарии к каждой строке листинга программы.