Лаборатория электроники и программирования

Электронный журнал с приложениями

№ 1

Автор Засыпкин С.В.

1. Учебные занятия.

1.1. Программирование на языке С на примере микроконтроллера ATmega16A. Часть 1: Установка логических сигналов.

Этим занятием начинается серия уроков для начинающих программировать микроконтроллеры.

Самая простая программа для микроконтроллера ATmega16A [1], которую требуется написать и понять позволяет установить на его выходах логические сигналы «0» и «1». Установка таких сигналов может понадобиться для многих задач — управление отдельными светодиодами, матричными светодиодными индикаторами, LCD модулями, реле и т.д.

Соберите схему, показанную на рисунке 1. В данной схеме показан необходимый минимум элементов, чтобы микроконтроллер заработал.

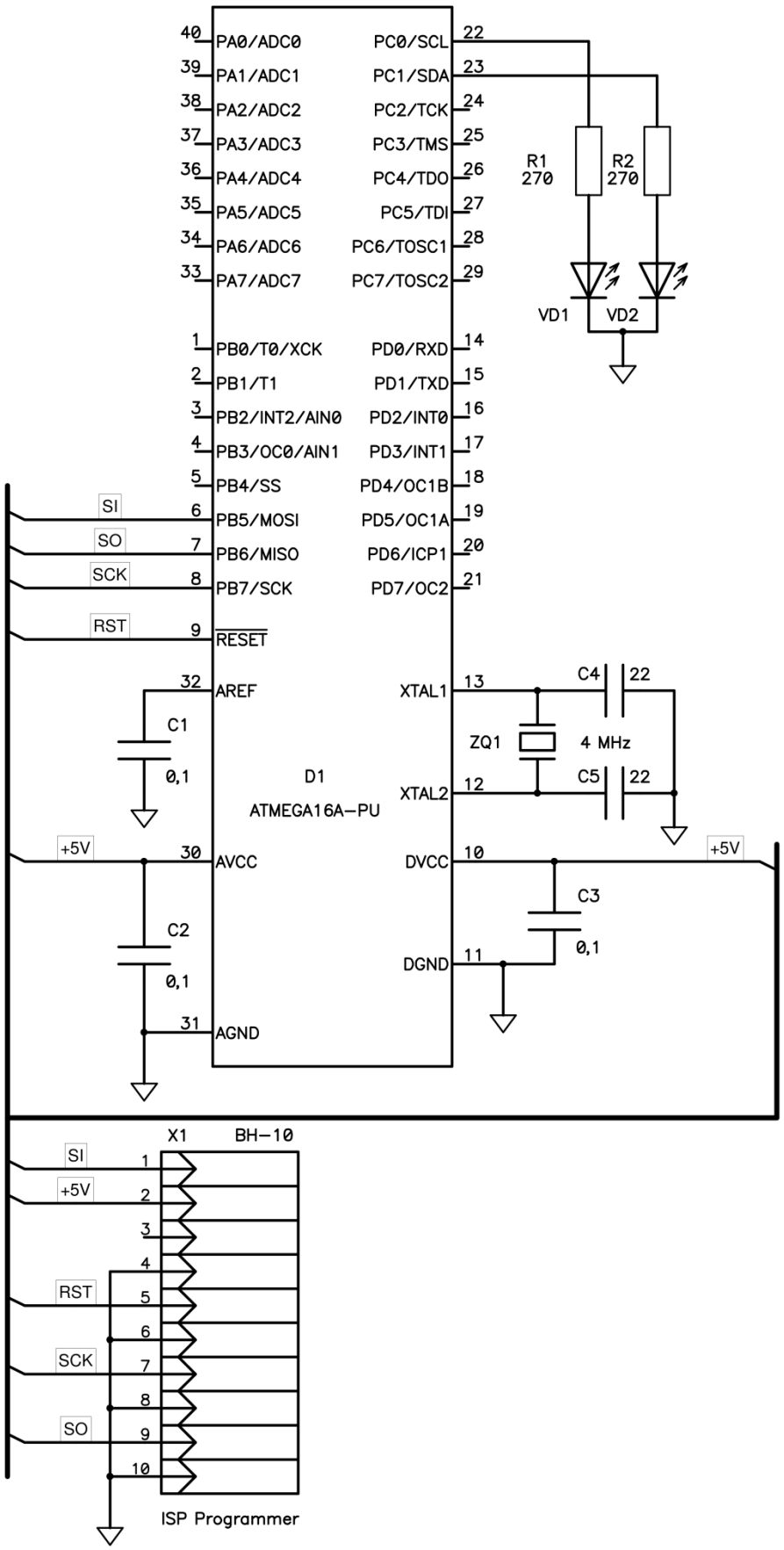
PA0/ADC0
PA1/ADC1
PA2/ADC2
PA3/ADC3
PA4/ADC4
PA5/ADC5
PA6/ADC6
PA7/ADC7
PB0/T0/XCK
PB1/T1
PB2/INT2/AIN0
PB3/OC0/AIN1
PB4/SS
PB5/MOSI
PB6/MISO
PB7/SCK
RESET
AREF
AVCC
AGND
PC0/SCL
PC1/SDA
PC2/TCK
PC3/TMS
PC4/TDO
PC5/TDI
PC6/TOSC1
PC7/TOSC2
PD0/RXD
PD1/TXD
PD2/INT0
PD3/INT1
PD4/OC1B
PD5/OC1A
PD6/ICP1
PD7/OC2
XTAL1
XTAL2
DVCC
DGND
D1
ATMEGA16A-PU
R1
270
R2
270
VD1
VD2
C1
0,1
C2
0,1
C3
0,1
C4
22
C5
22
ZQ1
4 MHz
SI
SO
SCK
RST
+5V
X1
BH-10
ISP Programmer

Рис. 1. Принципиальная схема для управления светодиодами.

D1 – микроконтроллер.

C4, C5, ZQ1 – часть схемы тактового генератора микроконтроллера (оставшаяся часть находится внутри микроконтроллера), который задает тактовую частоту микроконтроллера и соответственно его быстродействие. В данном примере тактовая частота равна 4 МГц.

C1 … C3 — блокировочные конденсаторы.

X1 — разъем для программирования микросхемы (прошивки программы).

VD1, VD2, R1, R2 – светодиоды с токоограничительными резисторами, которыми будем управлять.

Чтобы в программе обращаться к узлам микроконтроллера по определенным именам, которые обозначают регистры микроконтроллера, необходимо сначала подключить специальные файлы, в которых эти описания присутствуют.

Для каждого микроконтроллера (группы микроконтроллеров) существует типовой файл. В данных файлах находятся описания регистров микроконтроллера, битов и т.п. Эти файлы обычно находятся в каталоге (папке) «include» каталога (папки), в котором установлен компилятор С. Добавляются они с помощью директивы **#include** [2]:

```
#include <avr/io.h> // описания регистров микроконтроллера
```

Такая директива нужна в каждой программе.

То, что находится после символов //, называется комментарием. Здесь можно писать свои примечания на русском или английском языках, допускаются сокращения и грамматические ошибки. Операторы программы должны быть написаны на английском языке и без ошибок.

Тип микроконтроллера вы задаете при создании проекта.

Другая часто используемая директива - это директива **#define**, которая позволяет определить текстовое представление числа. Например, вместо номеров выводов порта 0 и 1, можно сделать смысловые имена:

```
#define VD1  0 // объявим имена для светодиодов
#define VD2  1
```

Далее в программе вместо «0» и «1» можно писать VD1 и VD2, что существенно облегчит читаемость программы. Читаемость программы важна в программировании, так быстрее можно понять смысл программы, исправить ошибки и внести изменения.

Прежде, чем использовать какое-либо периферийное устройство микроконтроллера, его необходимо настроить на нужное функционирование. Такая настройка обычно называется инициализацией.

Настройка порта ввода-вывода проста. Есть специальный регистр, который отвечает за установку вывода на вход или выход. Для порта С, который используется, он называется DDRC, при

записи логической «1» вывод порта настраивается на вывод, при «0» - на вход [1]. Таким образом, запишем часть программы:

```
// инициализация порта C

DDRC=0x03; // 0 и 1 разряды на вывод
```

С помощью данного оператора устанавливаются разряды 0 и 1 порта С, к которым подключены светодиоды, на выход.

Чтобы требуемые логические «0» и «1» появились уже на выводах микроконтроллера PC0 и PC1, необходимо их записать в специальный выходной регистр, который связан с выводами. В данном случае это PORTC [1]:

```
PORTC=0x00; // светодиоды выключены
```

С помощью данного оператора записаны 0 во все разряды порта С.

Заметим, что в данных примерах данные записывались сразу во все 8 разрядов порта. Для того чтобы установить значение одного требуемого разряда, существуют специальные команды [2], применительно для примера урока:

```
PORTC |= _BV(VD1); // устанавливаем бит VD1 в порту PORTC

PORTC &= ~(_BV(VD2)); // сбрасываем бит VD2 в порту PORTC
```

В программе на С, кроме уже рассмотренных директив и операторов существуют **функции**. Обычно это набор операторов для выполнения определенной задачи. В программе всегда должна быть обязательно так называемая главная функция — **main**. Остальные функции добавляются по необходимости.

Записывается она следующим образом:

```
int main(void)

{



}
```

Внутри скобок размещаются операторы и директивы. Директивы могут располагаться и вне функции.

В принципе все готово для написания программы по управлению светодиодами. Она будет выглядеть следующим образом:

```
#include <avr/io.h> // описания регистров микроконтроллера

#define VD1 0 // объявим имена для светодиодов

#define VD2 1
```

```
int main(void)
{
	// инициализация порта C
	PORTC=0x00; // светодиоды выключены
	DDRC=0x03; // 0 и 1 разряды на вывод

	PORTC |= _BV(VD1); // устанавливает бит VD1 в порту PORTC
	PORTC &= ~(_BV(VD2)); // сбрасывает бит VD2 в порту PORTC

	// зациклим программу
	while (1) {};
}
```

Данная программа включает светодиод VD1 и выключает VD2. Поскольку при инициализации оба светодиода были выключены, формально выключать VD2 не требуется и строку сброса бита VD2 можно не писать.

Если вы заметили, то светодиоды выключаются до того, как порт настраивается на вывод (установка нулевых значений в PORTC производится до настройки направления порта DDRC). Это обычно делается для того, чтобы избежать нежелательных переключений сигналов на выводах при переключении порта с работы «как вход» на работу «как выход».

Пояснения требует также оператор:

```
// зациклим программу
while (1) {};
```

Это оператор цикла. В данном его применении мы выполняем бесконечный пустой цикл. Т.е. микроконтроллер работает, но никаких конкретных действий не производит. Добавление этого оператора необходимо для того, чтобы вы увидели, как загорится светодиод.

Для проверки работы программы создайте новый проект в среде разработки [3]. Введите в него приведенную выше программу. Файл примера есть в приложении. Откомпилируйте проект с построением hex-файла (команда Build). Полученный в результате .hex файл представляет собой программу в машинных кодах, которую можно «зашить» в микроконтроллер. Запрограммируйте микроконтроллер с помощью имеющегося программатора и проверьте работу программы на макете. Также проверку работоспособности можно сделать в отладчике.

При программировании FUSES микроконтроллера установите тип генератора – кварцевый.

Для самоконтроля усвоения материала ответьте на вопросы и решите задачи:

1. Из каких объектов состоит программа на С?
2. Какие директивы Вы знаете?
3. Какая функция должна обязательно присутствовать в программе?
4. Напишите и испытайте на макете работоспособность программы, которая зажигает светодиод VD2, а VD1 - нет.
5. Напишите и испытайте на макете работоспособность программы, которая зажигает оба светодиода (VD1 и VD2).

Продолжение следует.

Список литературы и ссылки

1. Datasheet ATmega16A. © Atmel Corporation. http://www.atmel.com/
2. WinAVR™ . http://winavr.sourceforge.net/
3. AVR Studio®. © Atmel Corporation. http://www.atmel.com/