МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра вычислительной техники

УТВЕРЖДАЮ
Проректор по учебной работе
_____ О.Г. Локтионова
«14» 12 2021 г.

Программирование периферийных устройств информационных систем

Методические указания к практическим работам
для студентов направления подготовки 09.03.02

Курск 2021

УДК 004

Составители: А.В. Киселев, О.О. Яночкина

Рецензент
Кандидат технических наук, доцент *Ю.А. Халин*

Программирование периферийных устройств информационных систем: методические указания к практическим работам для студентов направлений подготовки 09.03.02/ Юго-Зап. гос. ун-т; сост.; А.В. Киселев, О.О. Яночкина. – Курск, 2021. - 18 с.: - ил. 5. – Библиогр.: с. 18

Содержат сведения по вопросам применения современных программных средств решения профессиональных задач.

Предназначены для студентов направления подготовки 09.03.02 очной и заочной форм обучения.

Методические указания соответствуют рабочей программе дисциплины «Программирование периферийных устройств информационных систем».

Текст печатается в авторской редакции

Подписано в печать 14.12.. Формат 60*84 1/16.
Усл. печ. л. 0,9. Уч.-изд. л. 0,8. Тираж 50 экз. Заказ 1708. Бесплатно.
Юго-Западный государственный университет.
305040 Курск, ул. 50 лет Октября, 94.

Практическая работа 1

Управление яркостью свечения светодиода

Цель работы – получение базовых навыков технологии программирования периферийного устройства на базе платформы Arduino.

1. Скачайте Arduino IDE с официального сайта (http://arduino.cc/en/Main/Software)

2. Если у вас Windows и Arduino IDE из zip-файла, установите драйверы из папки drivers

3. Подключите Arduino к компьютеру через USB

4. Запустите Arduino IDE

5. В «Tools → Board» выберите модель вашей платы (интерфейс Arduino IDE изображен на рисунке 1.1)

6. В «Tools → Serial Port» выберите порт, куда она подключена

7. Жмите «Upload» на панели инструментов для прошивки платы

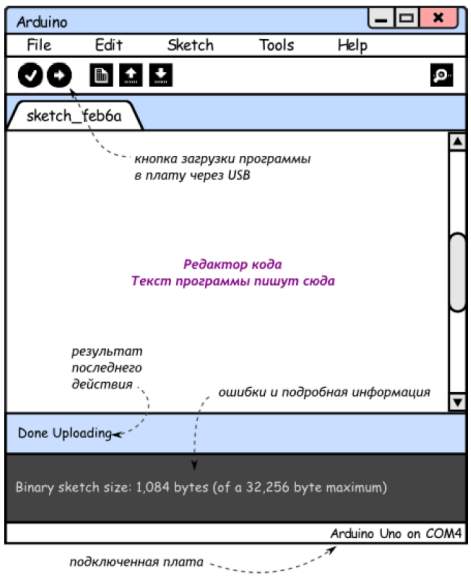

Рисунок 1.1 – Интерфейс Arduino IDE

Список компонентов для выполнения работы: плата Arduino Uno, беспаечная макетная плата, 2 светодиода, 2 резистора номиналом 220 Ом, 8 проводов «папа-папа», потенциометр.

Ход работы:

1. Подключите компоненты к плате согласно электрической принципиальной схеме (рисунок 1.2).

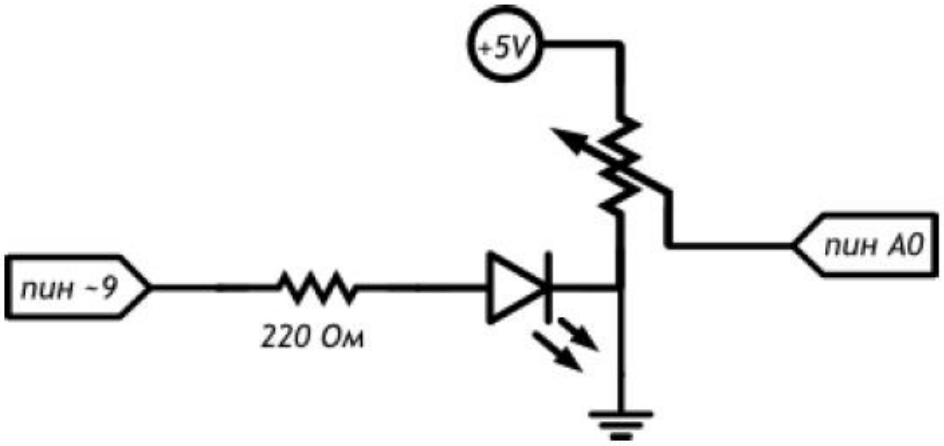

Рисунок 1.2 – Электрическая принципиальная схема

2. Откройте Arduino IDE, выберете вкладку File => New Sketch
3. Введите следующий скетч в окно редактора кода:

```
// даём имена для пинов со светодиодом
// и потенциометром (англ potentiometer или просто «pot»)
#define LED_PIN 9
#define POT_PIN A0
void setup()
{
// пин со светодиодом — выход, как и раньше...
pinMode(LED_PIN, OUTPUT);
// ...а вот пин с потенциометром должен быть входом
// (англ. «input»): мы хотим считывать напряжение,
// выдаваемое им
pinMode(POT_PIN, INPUT);
}
void loop()
{
// заявляем, что далее мы будем использовать 2 переменные с
```

```
// именами rotation и brightness, и что хранить в них будем
// целые числа (англ. «integer», сокращённо просто «int»)
int rotation, brightness;
// считываем в rotation напряжение с потенциометра:
// микроконтроллер выдаст число от 0 до 1023
// пропорциональное углу поворота ручки
rotation = analogRead(POT_PIN);
// в brightness записываем полученное ранее значение rotation
// делённое на 4. Поскольку в переменных мы пожелали хранить
// целые значения, дробная часть от деления будет отброшена.
// В итоге мы получим целое число от 0 до 255
brightness = rotation / 4;
// выдаём результат на светодиод
analogWrite(LED_PIN, brightness);
}
```

4. Загрузите его в плату, нажав «Upload» на панели инструментов для прошивки платы

Контрольные вопросы

1. Можем ли мы при сборке схемы подключить светодиод и потенциометр напрямую к разным входам GND микроконтроллера?

2. В какую сторону нужно крутить переменный резистор для увеличения яркости светодиода?

3. Что будет, если стереть из программы строчку pinMode(LED_PIN, OUTPUT)? строчку pinMode(POT_PIN, INPUT)?

4. Зачем мы делим значение, полученное с аналогового входа перед тем, как задать яркость светодиода? что будет, если этого не сделать?

Практическая работа 2

Измерения и индикация на базе Arduino Uno

Цель работы – получение навыков применения встроенного инструментария периферийного устройства на базе платформы Arduino.

Список компонентов для выполнения работы: плата Arduino Uno, беспаечная макетная плата, 2 резистора номиналом 10 кОм, выпрямительный диод, текстовый экран (LCD), 16 проводов «папа-папа», клеммник.

Ход работы:

1. Подключите компоненты к плате согласно электрической принципиальной схеме (рисунок 2.1).

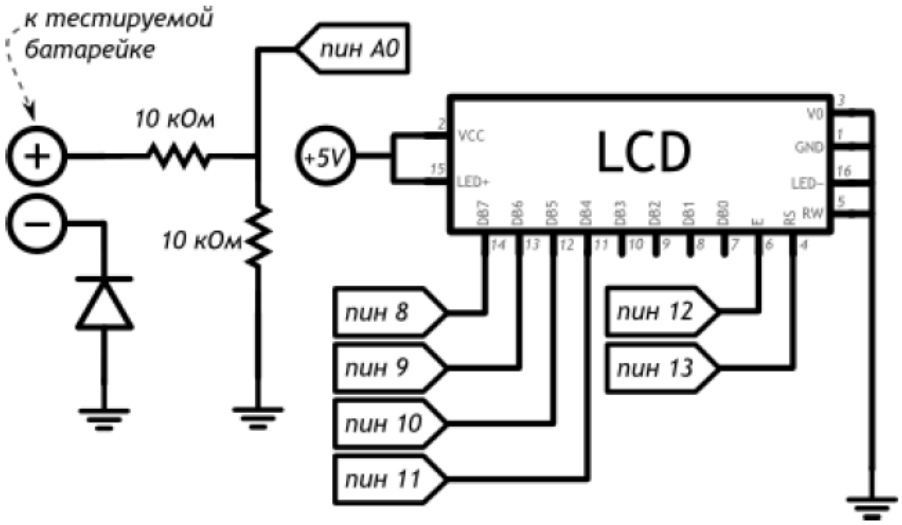

Рисунок 2.1 – Электрическая принципиальная схема

Мы подключаем «плюс» батарейки через делитель напряжения с равными плечами (R1 = R2 = 10 кОм), таким образом деля подаваемое напряжение пополам. Поскольку в аналоговый вход Arduino мы можем подавать до 5В, мы можем измерять напряжение до 10В. Не пробуйте измерять большее напряжение, вы можете повредить плату. На принципиальной схеме

внутри изображения дисплея подписаны названия его выводов согласно datasheet, а снаружи — номера его ножек. Ножки нашего ЖК-дисплея нумеруются не подряд: 15 и 16 ножки находятся перед 1. Диод необходим для того чтобы исключить возможность выхода из строя платы, если пользователь перепутает «+» и «-» батарейки.

2. Откройте Arduino IDE, выберете вкладку File => New Sketch
3. Введите следующий скетч в окно редактора кода:

```
// Подключаем библиотеку для работы с жидкокристаллическим
// экраном (англ. Liquid Crystal Display или просто LCD)
#include <LiquidCrystal.h>
// на диоде, защищающем от неверной полярности, падает доля
// напряжения (англ. voltage drop). Необходимо это учитывать
#define DIODE_DROP 0.7
// Объявляем объект, для управления дисплеем. Для его создания
// необходимо указать номера пинов, к которым он подключен в
// порядке: RS E DB5 DB6 DB7 DB8
LiquidCrystal lcd(13, 12, 11, 10, 9, 8);
void setup()
{
// начинаем работу с экраном. Сообщаем объекту количество
// строк и столбцов. Опять же, вызывать pinMode не требуется:
// функция begin сделает всё за нас
lcd.begin(16, 2);
// печатаем сообщение на первой строке
lcd.print("Battery voltage:");
}
void loop()
{
// высчитываем напряжение подключенной батарейки
```

```
float voltage = analogRead(A0) / 1024.0 * 10.0;
// если напряжение на делителе напряжения было зафиксировано,
// нужно прибавить напряжение на диоде, т.к. оно было съедено
if (voltage > 0.1)
voltage += DIODE_DROP;
// устанавливаем курсор, колонку 0, строку 1. На деле — это
// левый квадрат 2-й строки, т.к. нумерация начинается с нуля
lcd.setCursor(0, 1);
// печатаем напряжение в батарейке с точностью до сотых долей
lcd.print(voltage, 2);
// следом печатаем единицы измерения
lcd.print(" Volts");
}
```

Контрольные вопросы

1. Из-за чего измерения напряжения в этом эксперименте могут быть неточными (на что мы можем повлиять)?
2. Какая библиотека облегчает работу с нашим текстовым экраном? Какие шаги нужно предпринять до начала вывода текста на него?
3. Каким образом мы задаем позицию, с которой на экран выводится текст?
4. Можем ли мы писать на экране кириллицей?

Практическая работа 3

Счетчик на базе выходного сдвигового регистра

Цель работы – получение навыков рационального использования возможностей периферийного устройства на базе платформы Arduino при его взаимодействии с другими микросхемами.

Список компонентов для выполнения работы: плата Arduino Uno, беспаечная макетная плата, тактовая кнопка, выходной сдвиговый регистр 74HC595, семисегментный индикатор, 7 резисторов номиналом 220 Ом, 24 провода «папа-папа»

Ход работы:

1. Подключите компоненты к плате согласно электрической принципиальной схеме (рисунок 3.1).

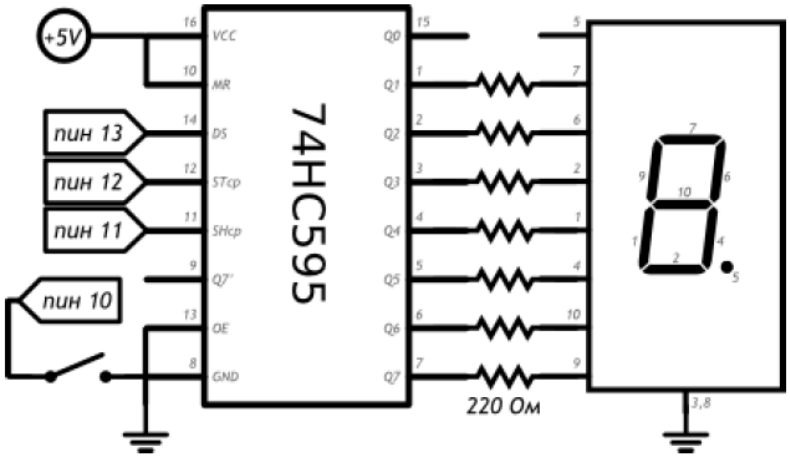

Рисунок 3.1 – Электрическая принципиальная схема

Выходной сдвиговый регистр дает нам возможность «сэкономить» цифровые выходы, использовав всего 3 вместо 8. Каскад регистров позволил

бы давать 16 и т.д. сигналов через те же три пина. Перед использованием микросхемы нужно внимательно изучить схему ее подключения в datasheet’е. Для того, чтобы понять, откуда считать ножки микросхемы, на них с одной стороны есть полукруглая выемка. Если мы расположим нашу 74HC595 выемкой влево, то в нижнем ряду будут ножки 1-8, а в верхнем 16-9. На принципиальной схеме ножки расположены в другом порядке, чтобы не вышло путаницы в соединениях. Назначения выводов согласно datasheet’у подписаны внутри изображения микросхемы, номера ножек — снаружи. На изображении семисегментного индикатора подписаны номера его ножек и их соответствие сегментам.

2. Откройте Arduino IDE, выберете вкладку File => New Sketch
3. Введите следующий скетч в окно редактора кода:

```
#define DATA_PIN 13 // пин данных (англ. data)
#define LATCH_PIN 12 // пин такта (англ. clock)
#define CLOCK_PIN 11 // пин строба (англ. latch)
#define BUTTON_PIN 10
int clicks = 0;
boolean buttonWasUp = true;
byte segments[10] = {
0b01111101, 0b00100100, 0b01111010, 0b01110110, 0b00100111,
0b01010111, 0b01011111, 0b01100100, 0b01111111, 0b01110111
};
void setup()
{
pinMode(DATA_PIN, OUTPUT);
pinMode(CLOCK_PIN, OUTPUT);
pinMode(LATCH_PIN, OUTPUT);
pinMode(BUTTON_PIN, INPUT_PULLUP);
```

```
}
void loop()
{
// считаем клики кнопки, как уже делали это раньше
if (buttonWasUp && !digitalRead(BUTTON_PIN)) {
delay(10);
if (!digitalRead(BUTTON_PIN))
clicks = (clicks + 1) % 10;
}
buttonWasUp = digitalRead(BUTTON_PIN);
// для записи в 74HC595 нужно притянуть пин строба к земле
digitalWrite(LATCH_PIN, LOW);
// задвигаем (англ. shift out) байт-маску бит за битом,
// начиная с младшего (англ. Least Significant Bit first)
shiftOut(DATA_PIN, CLOCK_PIN, LSBFIRST, segments[clicks]);
// чтобы переданный байт отразился на выходах Qx, нужно
// подать на пин строба высокий сигнал
digitalWrite(LATCH_PIN, HIGH);
}
```

Для того, чтобы передать порцию данных, которые будут отправлены через сдвиговый регистр далее, нам нужно подать LOW на latch pin (вход STcp микросхемы), затем передать данные, а затем отправить HIGH на latch pin, после чего на соответствующих выходах 74HC595 появится переданная комбинация высоких и низких уровней сигнала. Для передачи данных мы использовали функцию shiftOut(dataPin, clockPin, bitOrder, value). Функция ничего не возвращает, а в качестве параметров ей нужно сообщить пин Arduino, который подключен ко входу DS микросхемы (data pin), пин Arduino, соединенный со входом SHcp (clock pin), порядок записи битов: LSBFIRST

(least significant bit first) - начиная с младшего, или MSBFIRST(most significant bit first) - начиная со старшего, байт данных, который нужно передать. Функция работает с порциями данных в один байт, если нужно передать больше, придется вызывать ее несколько раз.

Контрольные вопросы

1. Для чего нужен выходной сдвиговый регистр?
2. Как найти ножку микросхемы, на которую отправляются данные?
3. Что нужно сделать до и после отправки собственно данных на 74HC595?
4. Сколько данных можно передать с помощью shiftOut() и как управлять порядком их передачи?

Практическая работа 4

Управление устройством на базе платформы Arduino через USB-интерфейс

Цель работы – получение базовых навыков управления периферийным устройством на базе платформы Arduino через USB-интерфейс.

Список компонентов для выполнения работы: плата Arduino Uno, беспаечная макетная плата, светодиод, резистор номиналом 220 Ом, 2 провода «папа-папа».

Ход работы:

1. Подключите компоненты к плате согласно электрической принципиальной схеме (рисунок 4.1).

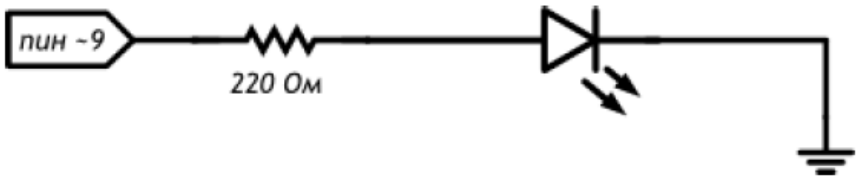

Рисунок 4.1 – Электрическая принципиальная схема

2. Откройте Arduino IDE, выберете вкладку File => New Sketch
3. Введите следующий скетч в окно редактора кода:

```
#define LED_PIN 9
// для работы с текстом существуют объекты-строки (англ. string)
String message;
void setup()
{
pinMode(LED_PIN, OUTPUT);
Serial.begin(9600);
}
```

```
void loop()
{
// передаваемые с компьютера данные поставляются байт за
// байтом, в виде отдельных символов (англ. character). Нам
// нужно последовательно их обрабатывать пока (англ. while)
// в порту доступны (англ. available) новые данные
while (Serial.available()) {
// считываем (англ. read) пришедший символ в переменную
char incomingChar = Serial.read();
// не стоит путать целые числа и символы. Они соотносятся
// друг с другом по таблице, называемой кодировкой. Например
// '0' — это 48, '9' — 57, 'A' — 65, 'B' — 66 и т.п. Символы
// в программе записываются в одинарных кавычках
if (incomingChar >= '0' && incomingChar <= '9') {
// если пришёл символ-цифра, добавляем его к сообщению
message += incomingChar;
} else if (incomingChar == '\n') {
// если пришёл символ новой строки, т.е. enter, переводим
// накопленное сообщение в целое число (англ. to integer).
// Так последовательность символов '1', '2', '3' станет
// числом 123. Результат выводим на светодиод
analogWrite(LED_PIN, message.toInt());
// обнуляем накопленное сообщение, чтобы начать всё заново
message = "";
}
}
// посылайте сообщения-числа с компьютера через Serial Monitor
}
```

4. Загрузите его в плату, нажав «Upload» на панели инструментов для прошивки платы

В этой программе мы создаем объект класса String. Это встроенный класс, предназначенный для работы со строками, т.е. с текстом. Не путайте его с типом данных string, который является просто массивом символов. String же позволяет использовать ряд методов для удобной работы со строками. Мы знакомимся с новым видом циклов: цикл с условием while. В отличие от цикла со счетчиком for, цикл while(expression) выполняется до тех пор, пока логическое выражение expression истинно. Метод available() объекта Serial возвращает количество байт, полученных через последовательный порт.

В данной работе цикл while работает до тех пор, пока available() возвращает ненулевое значение, любое из которых приводится к true. Переменные типа char могут хранить один символ. В этом примере символ мы получаем методом Serial.read(), который возвращает первый байт, пришедший на последовательный порт, или -1, если ничего не пришло. Обратите внимание, что в if мы сравниваем не пришедший символ с 0 и 9, но их коды. Если пришел какой-то символ, который не является цифрой, мы не будем его добавлять к нашей строке message. Объекты типа String позволяют производить конкатенацию, т.е. объединение строк. Это можно сделать так: message = message + incomingChar, но можно записать в сокращенной форме: message += incomingChar. В этой программе мы дополняем if конструкцией else if. Это еще один условный оператор, который проверяется только в случае ложности выражения, данного первому оператору. Несколько else if могут следовать друг за другом, при этом каждое следующее условие будет проверяться только в случае невыполнения всех предыдущих. Если в конце разместить else, он выполнится только если ни одно из условий не выполнено. Последовательностью \n кодируется символ переноса строки. Если он был передан устройству, мы передаем полученные ранее символы как параметр для analogWrite(), которая включает светодиод. Мы используем один из методов

String, toInt(), который заставляет считать строку не набором цифр, но числом. Он возвращает значение типа long, при этом, если строка начинается с символа, не являющегося цифрой, будет возвращен 0. Если после цифр, идущих в начале строки, будут символы не-цифры, на них конверсия остановится. Обратите внимание на выпадающее меню внизу монитора порта: чтобы наше устройство получало символ перевода строки, там должно быть выбрано «Новая строка (NL)» Пустая строка обозначается так: "". Опустошив ее, мы готовы собирать новую последовательность символов.

Контрольные вопросы

1. Какие объекты позволяют легко манипулировать текстовыми данными?
2. Что возвращают методы Serial.available() и Serial.read()?
3. Чем отличаются конструкции for и while?
4. Каким образом можно организовать более сложное ветвление, чем if … else?
5. Как можно объединить текстовые строки?
6. Как можно привести текстовую строку, содержащую цифры, к числовому типу?

Список литературы

1. Соммер, У. Программирование микроконтроллерных плат Arduino/Freeduino [Текст] / У. Соммер. – СПб.: БВХ – Петербург, 2012. – 256 с.
2. Болл Стюарт, Р. Аналоговые интерфейсы микроконтроллеров [Текст] / Р. Болл Стюарт – М.: Издательский дом «Додэка-XXI», 2007. – 360с.
3. Петин, В. А. Проекты с использованием контроллера Arduino [Текст] / В. А. Петин. – СПб.: БВХ-Петербург, 2014. – 400 с.
4. Ramos Enrique, M., Casto Ciriaco, D. Arduino and Kinect Projects [Текст] / M. Ramos Enrique, D. Casto Ciriaco. - Technology in Action, RM, ITA, 2010. – 411c.
5. Patrick di, J., Gertz, E. Atmospheric Monitoring with Arduino [Текст] / J. Patrick di, E. Gertz. – Sebastopol: O'Reilly Media, CA, USA, 2012. – 89c.