

РАЗРАБОТКА СИСТЕМЫ ПОДДЕРЖАНИЯ ТЕМПЕРАТУРЫ В ЖИЛОМ ПОМЕЩЕНИИ*

А.И. НЕЗВАНОВ¹, Е.Д. ЖАРОВ²

¹ 630073, РФ, г. Новосибирск, пр. Карла Маркса, 20, Новосибирский государственный технический университет, студент кафедры автоматики. E-mail: nezvanovml@mail.ru

² 630073, РФ, г. Новосибирск, пр. Карла Маркса, 20, Новосибирский государственный технический университет, студент кафедры автоматики. E-mail: edar.vih.esrr@yandex.ru

Идеей данного проекта послужило желание получить практику построения и сборки системы, позволяющей повысить личный комфорт жизни. Современный человек все больше привыкает к автоматизации рутинных процессов, а если эта автоматизация позволяет еще и экономить средства, то вопрос о ее необходимости даже не стоит. Экономия, которую возможно достичь при регулировании температуры в помещении, состоит в том, что не происходит перегрева воздуха в помещении и поэтому не приходится открывать окна для снижения температуры. А открытое окно в зимний период в Сибири – выброшенные на улицу деньги, так как лишнее тепло просто выбрасывается, а энергоресурсы для его производства стоят недешево. А еще хуже то, что их стоимость с каждым годом растет. Исходя из этого создание системы поддержания температуры в помещении видится целесообразным. Созданную систему для отдельно взятого помещения легко масштабировать до большего размера. Возможна покупка готовой системы, но ее стоимость будет однозначно выше самостоятельно собранной. В статье изложены материалы, используя которые была реализована система поддержания температуры. Далее происходит процесс проектирования схемы работы системы и подготовка платы под дальнейшую физическую сборку. По итогам проделанной работы был получен уникальный опыт в разработке и внедрении практически полезной системы. Количество сэкономленных энергоресурсов подсчитать не представляется возможным, но уровень комфорта за счет использования системы получилось увеличить. А комфорт для современного человека, возможно, значит больше, чем экономия.

Ключевые слова: микроконтроллер, ESP8266, отопление, электромагнитный клапан, реле, программирование, C++, пайка

* Статья получена 09 октября 2019 г.

ВВЕДЕНИЕ

На сегодняшний день в России бурно развивается частное домостроение и перед собственниками встает вопрос организации комфортной среды. Первое, что приходит на ум, – поддержание комфортной температуры в помещении. И это не только вопрос комфорта, но еще и экономии: с каждым годом энергоресурсы становятся всё дороже, и автоматика позволяет экономить средства, используя минимально необходимое количество ресурсов. Решение данной проблемы рассматривается в настоящей статье. В ходе работы был разработан комплекс устройств для поддержания температуры в помещении частного дома.

1. МАТЕРИАЛЫ, ИСПОЛЬЗУЕМЫЕ ДЛЯ РАБОТЫ

ESP8266 – микроконтроллер китайского производителя Espressif Systems с интерфейсом Wi-Fi (рис. 1). Микроконтроллер отличается отсутствием флеш-памяти в SoC, программы пользователя исполняются из внешней флеш-памяти с интерфейсом SPI.

Характеристики микроконтроллера:

- 80 МГц 32-битный процессор;
- интегрированный Wi-Fi с поддержкой WEP & WPA/WPA2;
- 11 портов ввода-вывода;
- питание 3,3 В.

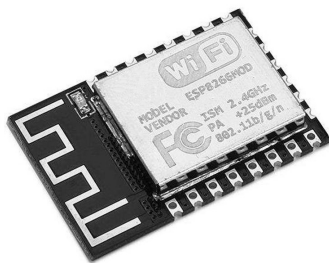


Рис. 1. Вид МК ESP8266

Программирование данного микроконтроллера возможно с использованием среды ARDUINO IDE. Для загрузки программы в МК необходимо использовать конвертер USB-UART, 21-й и 22-й выводы МК подключить к выводам RX/TX конвертера и подтянуть к земле 18-й вывод МК. Непосредственно

перед загрузкой программы необходимо кратковременно замкнуть первый вывод (RESET) на землю.

В качестве датчиков температуры используем DALLAS DS18B20 [2]. DS18B20 – цифровой термометр с программируемым разрешением, от 9 до 12 bit, которое может сохраняться в EEPROM памяти прибора. DS18B20 обменивается данными по 1-Wire шине и при этом может быть как единственным устройством на линии, так и работать в группе. Все процессы на шине управляются центральным микропроцессором. Диапазон измерений от -55 до $+125$ °C и точностью $0,5$ °C в диапазоне от -10 до $+85$ °C. В дополнение DS18B20 может питаться напряжением линии данных («parasite power») при отсутствии внешнего источника напряжения. Каждый DS18B20 имеет уникальный 64-битный последовательный код, который позволяет общаться с множеством датчиков DS18B20, установленных на одной шине. Такой принцип позволяет использовать один микропроцессор, чтобы контролировать множество датчиков DS18B20, распределенных по большому участку. Приложения, которые могут извлечь выгоду из этой особенности, включают системы контроля температуры в зданиях, в оборудовании или машинах, а также контроль и управление температурными процессами. Подключение датчиков весьма простое – необходим всего один вывод МК, работающий в режиме 1-Wire. Вдаваться в подробности протокола 1-Wire смысла нет, для работы с ним используем готовую библиотеку.

Для индикации состояния устройства используем доступный дисплей LCD 1602, работающий по протоколу I2C. Внешний вид дисплея представлен на рис. 2.



Рис. 2. Схема дисплея LCD 1602

Для подачи или перекрытия теплоносителя в батареях используется электромагнитный клапан, работающий от 230 В. Клапан является нормально открытым, т. е. при отсутствии электроэнергии или отключении разрабатываемой системы отопление работает так, будто клапана не существует. Внешний вид клапана представлен на рис. 3.



Рис. 3. Электромагнитный клапан 2W12-20

В качестве удаленного контроллера, расположенного у электромагнитного клапана, используем WeMos D1, так как он имеет на борту интегрированный Wi-Fi контроллер и представляет собой всё тот же ESP-8266 в готовом к использованию виде.

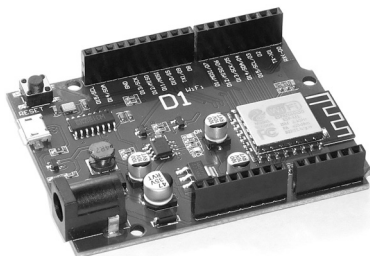


Рис. 4. Микроконтроллер WeMos D1

В качестве реле, управляющего электромагнитным клапаном посредством сигнала с микроконтроллера, используем стандартный модуль на двух реле для Arduino.



Рис. 5. Модуль на двух реле для Arduino

2. АЛГОРИТМ РАБОТЫ РАЗРАБАТЫВАЕМОГО УСТРОЙСТВА

Для контроля температуры необходимо в каждом помещении установить датчик температуры, при этом система ориентируется на минимальное показание термометра. Это необходимо для того, чтобы исключить влияние некачественного утепления или прочих особенностей отдельно взятого помещения на комфорт в нем. Такое решение позволяет гарантировать, что никто не замерзнет, но в то же время возможен перегрев другого помещения, что решается регулировкой балансировочного клапана радиаторов отдельного помещения.

Подача теплоносителя осуществляется при снижении температуры ниже заданной на $0,5\text{ }^{\circ}\text{C}$, а перекрытие – при превышении на $0,5\text{ }^{\circ}\text{C}$. Такой небольшой диапазон позволяет компенсировать большую инертность жидкостной системы отопления – на нагрев и остывание радиатора требуется время, соответственно помещение остывает во время начального прогрева радиаторов и догревается при их остывании. Таким образом, начальный диапазон $\pm 0,5\text{ }^{\circ}\text{C}$ расширяется до $\pm 0,75\text{ }^{\circ}\text{C}$ ($\pm 1\text{ }^{\circ}\text{C}$ в зависимости от внешних факторов).

Для исключения необходимости прокладки управляющих проводов до исполнительного устройства (клапана) используем беспроводной канал связи – Wi-Fi. Рядом с клапаном установим контроллер WeMos D1 с подключенным к нему модулем реле. Данный контроллер подключается к основному и запрашивает необходимое состояние клапана.

3. ПРОЕКТИРОВАНИЕ СХЕМЫ И ПОДГОТОВКА ПЛАТЫ

Для разработки схемы была использована программа, позволяющая размещать элементы и соединяющие дорожки – Sprint Layout. В ходе разработки были добавлены клеммники, резисторы, конденсаторы, стабилизатор AMS-1117-3.3 для получения напряжения в 3,3 В. Получившаяся схема представлена на рис. 6.

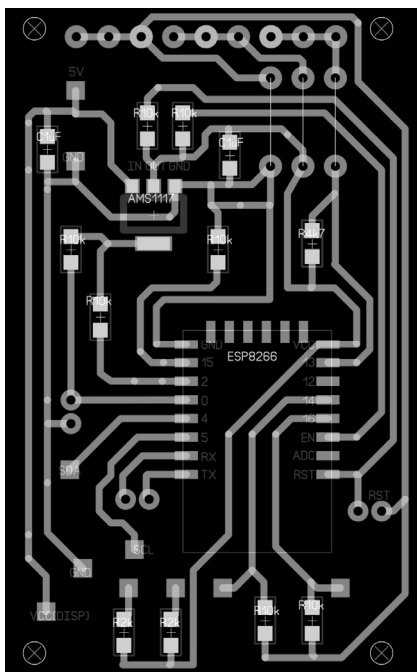


Рис. 6. Схема разработанной печатной платы основного контроллера

Для нанесения токопроводящих дорожек на текстолит был применен станок с ЧПУ – CNC3018. Он представляет широкие возможности в домашнем изготовлении печатных плат: гравировка дорожек, сверление отверстий, вырезание платы и срезание полигонов, естественно с использованием соответствующих инструментов – граверов, сверл и т. д. Управление производится с компьютера. Станок представлен на рис. 7.

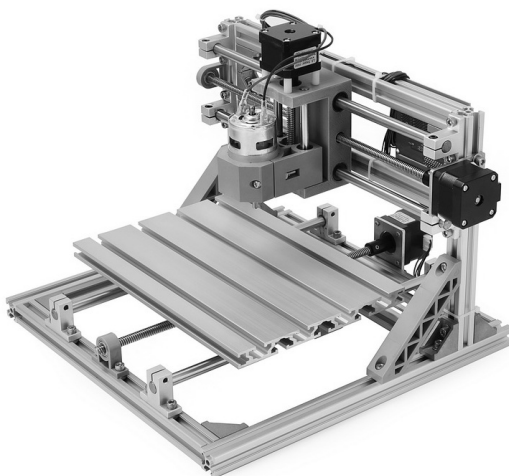


Рис. 7. ЧПУ станок CNC3018

На изготовление платы на станке потребовалось около 50 минут. Пайка элементов заняла еще около часа. Затем потребовалось изготовить корпус для устройства. Была создана 3d-модель, и корпус был напечатан на принтере. Внешний вид собранного устройства представлен на рис. 8.



Рис. 8. Собранный контроллер

Контроллер, расположенный непосредственно около клапана, представлен на рис. 10. Его внешний вид не имеет особого значения, поэтому в качестве корпуса был использован готовый бокс для автоматических выключателей.

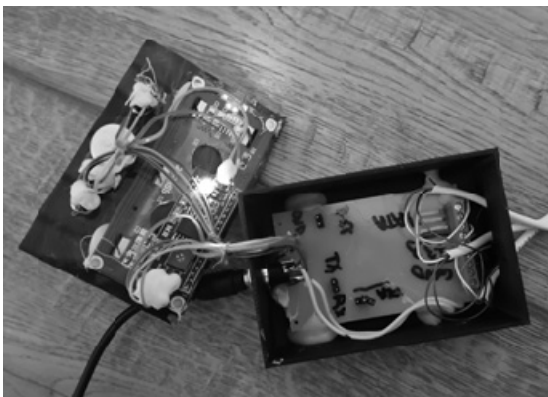


Рис. 9. Контроллер в разобранном виде



Рис. 10. Удаленный контроллер для управления клапаном

4. ИСХОДНЫЙ КОД ОСНОВНОГО КОНТРОЛЛЕРА

Библиотеки и части исходного кода (алгоритм работы с сетью) взяты из открытых источников, и их авторство ни в коей мере не оспаривается.

Исходный текст программы:

```
#include <Arduino.h>
#include <ESP8266WiFi.h>
#include <LiquidCrystal_I2C.h>
#include <Wire.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#include <EEPROM.h>
#define ONE_WIRE_BUS 13
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature DS18B20(&oneWire);
char temperatureCString[4];
float tempC1,tempC2,tempC3;
LiquidCrystal_I2C lcd(0x27, 16, 2);
const char* ssid = "MYWIFI";
const char* password = "12345678";
WiFiServer server(80);
unsigned long currentTime = 0;
int targetTemperature = 22; // target temperature
int state = 0; // valve status, 0 = OPEN, 1 = CLOSE
int lastMeasure = 0, lastConnection = 0;
int numOfMeasures = 0;
String s = "";
void eeWriteInt(int pos, int val) {
    byte* p = (byte*) &val;
    EEPROM.write(pos, *p);
    EEPROM.write(pos + 1, *(p + 1));
    EEPROM.write(pos + 2, *(p + 2));
    EEPROM.write(pos + 3, *(p + 3));
    EEPROM.commit();
}
int eeGetInt(int pos) {
    int val;
    byte* p = (byte*) &val;
    *p = EEPROM.read(pos);
    *(p + 1) = EEPROM.read(pos + 1);
    *(p + 2) = EEPROM.read(pos + 2);
    *(p + 3) = EEPROM.read(pos + 3);
    if(val >= 40 || val <= 0) val = 22;
    return val;
}
void setTargetTemperature(int direction){// 1 - up, 0 - down
    if(direction == 1){ targetTemperature++;eeWriteInt(1, targetTemperature);}
```

```
    else if(direction == 0){ targetTemperature--;eewriteInt(1, targetTempera-
ture);}
    lcd.setCursor(0,1);
    lcd.print("TT:");
    lcd.setCursor(4,1);
    lcd.print(targetTemperature);
}

void setup()
{
    DS18B20.begin();
    pinMode(14, INPUT);
    pinMode(16,INPUT);
    lcd.begin(4,5);    // In ESP8266-01, SDA=0, SCL=2
    lcd.backlight();
    lcd.setCursor(0,0);
    lcd.print("Connecting to:");
    lcd.setCursor(0,1);
    lcd.print(ssid);
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
    }
    server.begin();
    lcd.clear();
    EEPROM.begin(10);
    targetTemperature = eeGetInt(1);
    setTargetTemperature(2);
}

float getTemperature() {
    DS18B20.requestTemperatures();
    tempC1 = DS18B20.getTempCByIndex(0);
    tempC2 = DS18B20.getTempCByIndex(1);
    tempC3 = DS18B20.getTempCByIndex(2);
    float min = tempC1;
    if(tempC1 > -50 && tempC1 < 60){
        lcd.setCursor(0,0);    // Установить курсор на вторую строку
        dtostrf(tempC1, 2, 1, temperatureCString);
        lcd.print(temperatureCString);    // Вывести текст
    }
    if(tempC2 > -50 && tempC2 < 60){
        lcd.setCursor(6,0);    // Установить курсор на вторую строку
        dtostrf(tempC2, 2, 1, temperatureCString);
        lcd.print(temperatureCString);    // Вывести текст
        if(tempC2 < min ) min = tempC2;
    }
    if(tempC3 > -50 && tempC3 < 60){
        lcd.setCursor(12,0);    // Установить курсор на вторую строку
        dtostrf(tempC3, 2, 1, temperatureCString);
    }
}
```

```

        lcd.print(temperatureCString);    // Вывести текст
        if(tempC3 < min ) min = tempC3;
    }
    lastMeasure = millis();
    return min;
}
float actualTemperature = getTemperature();
int lastTemperature = (int)actualTemperature;
void loop()
{
    currentTime = millis();
    if(currentTime - lastMeasure >= 10000){
        actualTemperature = getTemperature();
        if(actualTemperature == lastTemperature) numOfMeasures++;
        else numOfMeasures = 0;
    }
    while(digitalRead(14) == 1){setTargetTemperature(1);delay(500);};
    while(digitalRead(16) == 1){setTargetTemperature(0);delay(500);};
    if(state == 0){ //valve open
        if(actualTemperature >= targetTemperature+0.5 ){
            numOfMeasures = 0;
            state = 1;
        }
    } else if(state == 1){ //valve closed
        if(actualTemperature <= targetTemperature-0.5){
            numOfMeasures = 0;
            state = 0;
        }
    }
    lcd.setCursor(7, 1);
    if(state == 1) lcd.print("CLS");
    else if(state == 0) lcd.print("OPN");
    lcd.setCursor(11, 1);
    if((currentTime-lastConnection)/1000 < 10) lcd.print("OK   ");
    else lcd.print("CHECK!");
    // Check if a client has connected
    WiFiClient client = server.available();
    if (!client) {
        return;
    }
    int num = 0;
    if(!client.available()){
        return;
    }
    // Read the first line of the request
    String req = client.readStringUntil('\r');
    client.flush();

```

```

int val;
if (req.indexOf("status") != -1){
    client.flush();
    if(state) s = "ON\r";
    else s = "OFF\r";
    client.print("Minimal temperature:
");client.print(actualTemperature);client.print("C\r\n ");client.print("All
sensors: ");client.print(tempC1);client.print("C
");client.print(tempC2);client.print("C
");client.print(tempC3);client.print("C ");
    client.print("\r\n");
    if((currentTime-lastConnection)/1000 < 10){
        if(state == 1) client.print("Valve closed");
        else if(state == 0) client.print("Valve opened");
    }
    else client.print("Check valve!");
    client.print("\r\n");
    delay(1);
    client.stop();
    return;
}
else{
    client.flush();
    if(state) s = "ON\r";
    else s = "OFF\r";
    client.print(s);
    delay(1);
    lastConnection = millis();
    client.stop();
    return;
}
}

```

5. ИСХОДНЫЙ КОД УДАЛЕННОГО КОНТРОЛЛЕРА

```

#include <ESP8266WiFi.h>
const char* ssid = " MYWIFI ";
const char* password = "12345678";
int count = 0;
const char* host = "192.168.1.50";
#define RELAY1 12
#define RELAY2 4
void setup()
{
    pinMode(LED_BUILTIN, OUTPUT);
    digitalWrite(LED_BUILTIN, HIGH);//off LED
    pinMode(RELAY1,OUTPUT);
    digitalWrite(RELAY1, HIGH);
}

```

```
pinMode(RELAY2, OUTPUT);
digitalWrite(RELAY2, HIGH);
Serial.begin(9600);
delay(10);
//Serial.begin(115200);
//Serial.println();
Serial.printf("Connecting to %s ", ssid);
    // "Подключение к "
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED)
{
    delay(500);
    Serial.print(".");
}
Serial.println(" connected");
    // " подключено"
}
void loop()
{
    WiFiClient client;
    Serial.printf("\n[Connecting to %s ... ", host);
        // "Подключение к "
digitalWrite(LED_BUILTIN, LOW); //off LED
if (client.connect(host, 80))
{
    Serial.println("connected");
        // " подключено"
    Serial.println("[Sending a request]");
        // "Отправка запроса"
    client.print(String("/data"));

    Serial.println("[Response:]");
        // "Ответ:"
    while (client.connected())
    {
        if (client.available())
        {
            String line = client.readStringUntil('\r');
            Serial.println(line);
            if(line == "ON"){
                digitalWrite(RELAY1, LOW);digitalWrite(RELAY2, LOW);
            } else if(line == "OFF"){
                digitalWrite(RELAY1, HIGH);digitalWrite(RELAY2, HIGH);
            }
            count = 0;
        }
    }
}
client.stop();
Serial.println("\n[Disconnected]");
```

```
        // "Отключено"
    }
    else
    {
        count++;
        if(count > 4){
            digitalWrite(RELAY1, HIGH);
            digitalWrite(RELAY2, HIGH);
        }
        Serial.println("connection failed!]);
        // "подключиться не удалось!"
        client.stop();
    }
    digitalWrite(LED_BUILTIN, HIGH); //off LED
    delay(5000);
}
```

ЗАКЛЮЧЕНИЕ

В данной работе были достигнуты поставленные цели, собранная система полностью работоспособна. Также был получен навык в программировании МК ESP-8266, проектировании и сборке печатных плат. По итогам тестирования была получена обратная связь от собственника дома, система полностью оправдала ожидания.

СПИСОК ЛИТЕРАТУРЫ

1. Espressif – ESP8266EX Datasheet. – URL: https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf (accessed: 18.12.2019).
2. Dallas Semiconductor – DS18B20 Datasheet. – URL: https://cxem.net/ckfinder/userfiles/comments/43118_ds18b20-rus.pdf (accessed: 18.12.2019).
3. Петин В. Проекты с использованием контроллера Arduino. – СПб.: БХВ-Петербург, 2014. – 400 с.
4. Margolis M. Arduino cookbook. – Sebastopol: O'Reilly, 2012. – 724 p.
5. Blum J. Exploring Arduino: tools and techniques for engineering wizardry. – Indianapolis, IN: Wiley, 2013. – 357 p.
6. Гребнев В.В. Микроконтроллеры семейства AVR фирмы Atmel. – М.: Радиософт, 2002. – 176 с.
7. Евстифеев А.В. Микроконтроллеры AVR семейства Tiny. – М.: Додэка-XXI, 2007. – 290 с. – (Программируемые системы).
8. Евстифеев А.В. Микроконтроллеры AVR семейств Tiny Mega Atmel. – М.: Додэка-XXI, 2005. – 546 с.

9. *Мортон Дж.* Микроконтроллеры AVR. Вводный курс. – М.: Додэка-XXI, 2005. – 345 с.
10. *Белов А.В.* Микроконтроллеры AVR в радиолюбительской практике. – СПб.: Наука и техника, 2007. – 337 с.
11. *Редькин П.П.* Микроконтроллеры Atmel архитектуры AVR32 семейства AT32UC3. – М.: Техносфера, 2010. – 783 с.
12. *Голубцов М.С.* Микроконтроллеры AVR: от простого к сложному. – М.: Солон-Пресс, 2003. – 287 с.
13. *Гадре Д., Мэлхотра Н.* Занимательные проекты на базе микроконтроллеров tinyAVR. – СПб.: БХВ-Петербург, 2012. – 340 с.
14. *Шпак Ю.А.* Программирование на языке С для AVR и PIC микроконтроллеров. – Изд. 2-е, перераб. и доп. – Киев: МК-Пресс; СПб.: Корона-Век, 2011. – 544 с.
15. *Кривченко И.В.* AVR-микроконтроллеры: очередной этап на пути развития // Компоненты и технологии. – 2002. – № 3 (20). – С. 98–103.
16. *Баранов В.Н.* Применение микроконтроллеров AVR: схемы, алгоритмы, программы. – 2-е изд. – М.: Додэка-XXI, 2006. – 288 с.

Незванов Александр Игоревич, студент кафедры автоматике Новосибирского государственного технического университета. E-mail: nezvanovml@mail.ru

Жаров Евгений Дмитриевич, студент кафедры автоматике Новосибирского государственного технического университета. E-mail: edar.vih.esrr@yandex.ru

DOI: 10.17212/2307-6879-2019-3-4-122-138

Development of residential temperature maintenance system*

A.I. Nezvanov¹, E.D. Zharov²

¹ *Novosibirsk State Technical University, 20 Karl Marx Prospect, Novosibirsk, 630073, Russian Federation, student department of automation. E-mail: nezvanovml@mail.ru*

² *Novosibirsk State Technical University, 20 Karl Marx Prospect, Novosibirsk, 630073, Russian Federation, student department of automation. E-mail: edar.vih.esrr@yandex.ru*

The idea of this project was the desire to get the practice of building and assembling a system that allows to increase personal comfort of life. Modern man is becoming increasingly used to automation of routine processes, and if this automation also allows to save funds – the question of its necessity is not even worth it. The savings that can be achieved by controlling the room tempera-

* Received 09 October 2019.

ture are that there is no overheating of the room air and therefore no opening of the windows to reduce the temperature. And the open window in winter in Siberia - the money thrown on the street, as excess heat is simply thrown away, and energy resources for its production are not cheap. Worse still, their cost rises every year. On this basis, the creation of a room temperature maintaining system seems appropriate. The created system for a single room is easily scaled to a larger size. It is possible to buy a finished system, but its cost will be definitely higher than the self-collected system. The article describes the materials used by the temperature maintenance system. Next, the process of designing the system operation diagram and preparing the board for further physical assembly takes place. As a result of the work done, unique experience was gained in the development and implementation of a practically useful system. The amount of energy saved cannot be calculated, but the level of comfort due to the use of the system has been increased. And comfort for a modern man may mean more than saving.

Keywords: Microcontroller, ESP8266, water heating, solenoid valve, relay, programming, C, soldering

REFERENCES

1. *Espressif – ESP8266EX Datasheet*. Available at: https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf (accessed 18.12.2019).
2. *Dallas Semiconductor – DS18B20 Datasheet*. Available at: https://cxem.net/ckfinder/userfiles/comments/43118_ds18b20-rus.pdf (accessed 18.12.2019).
3. Petin V. *Proekty s ispol'zovaniem kontrollera Arduino* [Projects using Arduino]. St. Petersburg, BHV-Peterburg Publ., 2014. 400 p.
4. Margolis M. *Arduino cookbook*. Sebastopol, O'Reilly, 2012. 724 p.
5. Blum J. *Exploring Arduino: tools and techniques for engineering wizardry*. Indianapolis, IN, Wiley, 2013. 357 p.
6. Grebnev V.V. *Mikrokontrollery semeistva AVR firmy Atmel* [Microcontrollers of AVR type from Atmel]. Moscow, Radiosoft Publ., 2002. 176 p.
7. Evstifeev A.V. *Mikrokontrollery AVR semeistva Tiny* [Microcontrollers Tiny of AVR type]. Moscow, Dodeka-XXI Publ., 2007. 290 p.
8. Evstifeev A.V. *Mikrokontrollery AVR semeistv Tiny Mega Atmel* [Microcontrollers Tiny, Mega of AVR type]. Moscow, Dodeka-XXI Publ., 2005. 546 p.
9. Morton J. *AVR: an introductory course*. Oxford, Newnes, 2002 (Morton Dzh. *Mikrokontrollery AVR. Vvodnyi kurs*. Moscow, Dodeka-XXI Publ., 2005. 345 p).
10. Belov A.V. *Mikrokontrollery AVR v radiolyubitel'skoi praktike* [Microcontrollers AVR if radiotechnics practice]. St. Petersburg, Nauka i tekhnika Publ., 2007. 337 p.
11. Red'kin P.P. *Mikrokontrollery Atmel arhitektury AVR32 semejstva AT32UC3* [Microcontrollers Atmelwith AVR32 architecture]. Moscow, Tekhnosfera Publ., 2010. 783 p.

12. Golubtsov M.S. *Mikrokontrollery AVR: ot prostogo k slozhnomu* [Micro-controllers AVR: from easy to hard]. Moscow, Solon-Press Publ., 2003. 287 p.
13. Gadre D., Malhotra N. *TinyAVR microcontroller projects for the evil genius*. New York, McGraw-Hill, 2011 Russ. ed.: Gadre D., Melkhotra N. *Zanimatel'nye proekty na baze mikrokontrollerov tinyAVR*. St. Petersburg, BHV-Peterburg Publ., 2012. 340 p.).
14. Shpak Yu.A. *Programmirovaniye na yazyke C dlya AVR i PIC mikrokontrollerov* [Programming on C for AVR & PIC microcontrollers]. Kiev, MK-Press Publ., St. Petersburg, Korona-Vek Publ., 2011. 544 p.
15. Krivchenko I.V. AVR-mikrokontrollery: ocherednoi etap na puti razvitiya [AVR-microcontrollers: stage of evolution]. *Komponenty i tekhnologii – Components & Technologies*, 2002, no. 3 (20), pp. 98–103.
16. Baranov V.N. *Primeneniye mikrokontrollerov AVR: skhemy, algoritmy, programmy* [Using microcontrollers AVR: schemes, algorithmes, programmes]. 2nd ed. Moscow, Dodeka-XXI, 2006. 288 p.

Для цитирования:

Незванов А.И., Жаров Е.Д. Разработка системы поддержания температуры в жилом помещении // Сборник научных трудов НГТУ. – 2019. – № 3–4 (96). – С. 122–138. – DOI: 10.17212/2307-6879-2019-3-4-122-138.

For citation:

Nezvanov A.I., Zharov E.D. Razrabotka sistemy podderzhaniya temperatury v zhilom pomeshchenii [Development of residential temperature maintenance system]. *Sbornik nauchnykh trudov Novosibirskogo gosudarstvennogo tekhnicheskogo universiteta – Transaction of scientific papers of the Novosibirsk state technical university*, 2019, no. 3–4 (96), pp. 122–138. DOI: 10.17212/2307-6879-2019-3-4-122-138.