споживачів / А. Боровий, І. Майків, Р. Кочан, З. Домбровський, В. Кочан. № а 2008 06325; заявл. 13.05.2008; опубл. 10.06.2010, Бюл. № 11.

Надійшла до редакції
7.2.2013 р.

УДК 004.71

О.А. МЯСІЩЕВ
Хмельницький національний університет

ПРО МОЖЛИВІСТЬ ВИКОРИСТАННЯ СЕРЕДОВИЩА РОЗРОБКИ ARDUINO ДЛЯ СТВОРЕННЯ WEB - СЕРВЕРА НА БАЗІ ATMEGA32

Вивчено можливість використання програмного середовища Arduino для підключення різних модулів до сімейства мікроконтролерів AVR. На базі ATmega32 і мережевого модуля WIZ812MJ створений web-сервер для віддаленого управління виконавчими механізмами і зняття параметрів з датчиків для TCP/IP мережі. Показана можливість в мінімальні терміни створювати складні проекти для мікроконтролерів на основі використання об'єктно-орієнтованого програмування.

Ключові слова: проект Arduino, мікроконтролер, web-сервер, bootloader, порти введення / виводу мікроконтролера

The possibility of using Arduino programming environment to connect different modules to the family of microcontrollers AVR. Based on ATmega32 and network module WIZ812MJ built web-server for remote control of actuators and sensors to remove options for TCP / IP network. The possibility of a minimum time to create sophisticated designs for microcontrollers based on the use of object-oriented programming.

Key words: project Arduino, a microcontroller, web-server, bootloader, I / O ports of the microcontroller

Вступ

В даний час для управління різними пристроями (кондиціонерами, системами опалення, освітленням і т.д.) широко використовують мікроконтролери, наприклад ATMEGA. У свою чергу вони підключаються до відповідним платам розширення, які і керують виконавчими пристроями. Для програмування мікроконтролерів використовують відповідне програмне забезпечення. Зазвичай програми пишуться на Сі та Асемблері. Проте в кінці 2000р. з ініціативи італійських інженерів було створено проект Arduino, який зараз дуже інтенсивно розвивається для створення закінчених систем управління виконавчими пристроями. Проект включає в себе апаратну обчислювальної платформи, яка представляє собою плату вводу/виводу, а також засіб розробки програм [1,2].

Постановка завдання

Плата Arduino складається з обмеженого набору мікроконтролерів Atmel AVR (ATmega1280, ATmega328 в нових версіях і ATmega168, ATmega8 в старих) і елементної обв'язки для програмування та інтеграції з іншими схемами. На кожній платі обов'язково присутні лінійний стабілізатор напруги 5В і 16МГц кварцовий резонатор. У мікроконтролер попередньо прошитий завантажувач, тому зовнішній програматор не використовується. На концептуальному рівні всі плати програмуються через RS-232 інтерфейс. Однак на останні варіанти підключення для програмування виконується через USB завдяки мікросхемі конвертера USB-to-serial типу FTDI FT232. Плати Arduino дозволяють використовувати більшу частину I/O висновків мікроконтролера в зовнішніх схемах. Наприклад, в платі Arduino Uno (ATmega328) доступно 14 цифрових вводів / виводів, в платі Arduino Mega (ATmega1280) доступно вже 54 цифрових вводів / виводів.

Інтегроване середовище розробки Arduino - це багатоплатформовий додаток на Java, що включає в себе редактор коду, компілятор і модуль передачі прошивки в плату. Середа розробки заснована на мові програмування Processing. Мова програмування аналогічний мові, використовуваному в проекті Wiring. Програми обробляються за допомогою препроцесора (Wiring), а потім компілюється за допомогою AVR-GCC.

Популярність Arduino в даний час пов'язана його перевагами. Не потрібен програматор. Не потрібні особливо глибокі пізнання в програмування мікроконтролерів (програмування виконується на мові високого рівня). Проект Arduino повністю відкритий (можна виготовляти додаткові плати розширення і дописувати власні бібліотеки програм). Платформа набирає популярність - з'являється безліч сайтів в Інтернеті з бібліотеками, схемами і проектами. Стандартизація розташування виводів плати вводу/виводу робить привабливою її для виробників - з'являються все нові шілд (плати розширення). Проект використовує кроссплатформеное середовище розробки.

Однак істотним недоліком проекту є орієнтація програмного середовища на обмежену кількість мікроконтролерів AVR. Ряд популярних і поширених мікроконтролерів, наприклад ATmega16, ATmega32, ATmega644 в корпусі DIP не описані в середовищі розробки. В даний час доступні налагоджувальні плати (наприклад, налагоджувальний комплекс AVR-EASY-KIT [3]), які мають вводи/виводи, що відрізняються від вводів/виводів, прийнятих в апаратній частині проекту Arduino. У цій роботі розглядається можливість використання інтегральної середовища розробки Arduino для створення web - сервера на базі

мікроконтролера ATmega32, налагоджувальний комплексу AVR-EASY-KIT і модуля WIZ812MJ [4]. Причому web - сервер повинен виконувати функції управління виконавчими пристроями (3 пристрої), відображати на браузер клієнта стан виконавчих пристроїв, температури (зчитаної з датчика DS18B20) і показання напруги, отриманого з одного з аналогових входів мікроконтролера. На рідкокристалічному індикаторі LCD повинен виводиться ір адреса віддаленого підключеного клієнта. Зазвичай такі завдання вирішуються в проекті "Розумний будинок".

Основна частина

Для вирішення завдання необхідно спочатку підправити файли середовища розробки Arduino. Робота з Arduino виконується на комп'ютері з операційною системою Windows XP. Кореневим каталогом є d: \ arduino-1.0.3, в якому встановлені всі файли проекту.

1. У файл опису конфігурації плати Arduino для мікроконтролера ATmega16 необхідно додати рядки:

```
atmega16.name=mega16
atmega16.upload.protocol=avr109
atmega16.upload.maximum_size=15336
atmega16.upload.speed=19200
atmega16.build.variant=m16
atmega16.build.mcu=atmega16
atmega16.build.f_cpu=7372800L
atmega16.build.core=arduino
А для ATmega32 - строки:
atmega32.name= mega32
atmega32.upload.protocol=stk500v2
atmega32.upload.maximum_size=28336
atmega32.upload.speed=38400
atmega32.build.variant=m16
atmega32.build.mcu=atmega32
atmega32.build.f_cpu=7372800L
atmega32.build.core=arduino
```

Тут avr109 і stk500v2 - протоколи для прошивки мікроконтролерів з використанням bootloader, 7372800L - частота в герцах тактирования мікроконтролера в герцах, m16 - каталог (d:\arduino-1.0.3\hardware\arduino\variants\m16) з распіновкой портів мікроконтролера.

2. Необхідно відкоригувати файл - d:\arduino-1.0.3\hardware\arduino\variants\standard\pins_arduino.h, в якому описуються порти ATmega8, ATmega168 так, щоб він описував порти ATmega16, ATmega32 для вирішення поставленого завдання. Після цього цей файл буде скопійований в каталог d:\arduino-1.0.3 hardware\arduino\variants\m16. На відміну від ATmega168 ця заміна призведе до наступної відповідності між портами і виводами:

цифровими: PortD - PD0 - вивід 0, PD1-1, PD2-2, PD3-3, PD4-4, PD5-5, PD6-6, PD7-7;

PortB - PB2 - вивід 8, PB3-9, PB4-10, PB5-11, PB6-12, PB7-13;

PortC - PC0 – вивід 14, PC1-15, PC2-16, PC3-17, PC4-18, PC5-19, PC6-20, PC7-21;

аналоговими: PortA - PA0 - вивід 22, PA1-23, PA2-24, PA3-25, PA4-26, PA5-27, PA6-28, PA7-29.

3. Додати в файл опису програматорів (d:\arduino-1.0.3\hardware\arduino\programmers.txt), за допомогою яких можна виконувати програмування мікроконтролера зовнішнім програматором наступні рядки:

```
avr109.name=avr109
avr109.communication=serial
avr109.speed=19200
avr109.protocol=avr109
```

Це необхідно для ATmega16, bootloader якого вимагає протокол avr109. Програмування виконується через послідовний порт зі швидкістю 19200Біт/с.

4. Ethernet модуль WIZ812MJ працює з мікроконтролером по шині SPI. Вибір пристрою виконується в проекті Arduino за замовчуванням 10-м вводом/виводом (лінія SS). Згідно зі зміненим файлу pins_arduino.h для ATmega32 він відповідає порту PB4 (_BV (4)), а не PB2 (_BV (2)) як для ATmega168. Тому у файлі d:\arduino-1.0.3\libraries\Ethernet\utility\w5100.h виконується заміна на:

```
#else
  inline static void initSS()    { DDRB  |=  _BV(4); };
  inline static void setSS()     { PORTB &= ~_BV(4); };
  inline static void resetSS()   { PORTB |=  _BV(4); };
#endif
```

Вибірка SD карти пам'яті, яка також працює по шині SPI, буде виконуватися 8-м вводом/виводом (портом PB2) на зібраному web-сервері. На карті знаходиться текстова інформація, яку сервер передає браузеру клієнта.

5.Для відображення на LCD індикаторі web-сервера ip-адреси клієнта, який підключився, необхідно дописати наступні команди в кінець файлу d:\arduino-1.0.3\libraries\Ethernet\Ethernet.cpp:

```
// Add IPremoute
uint8_t *EthernetClient::getRemoteIP(uint8_t remoteIP[])
```

```
{
W5100.readSnDIPR(_sock, remoteIP);
return remoteIP;
}
```

і в файл

d:\arduino-1.0.3\libraries\Ethernet\Ethernet.h под строкой `virtual void stop();` :

```
uint8_t *getRemoteIP(uint8_t RemoteIP[]);//adds remote ip address
```

Для полегшення процедури програмування мікроконтролера проект Arduino передбачає використання bootloader. Зазвичай bootloader Arduino компілюються на тактову частоту 16МГц і для обмеженого числа мікроконтролерів, які не включають ATmega32. У роботі компіляція bootloader виконується з вихідного коду, представленого в роботі [4]. У Makefile робляться коригування для ATmega32, частоти 7372800 Гц, адреси завантажувальної області 0x7800 і швидкості обміну даними 38400. У файлі d:\arduino-1.0.3\hardware\arduino\bootloaders\stk500v2\Makefile знаходяться рядки

```
penguino: MCU = atmega32
penguino: F_CPU = 7372800
penguino: BOOTLOADER_ADDRESS = 7800
penguino: CFLAGS += -D_PENGUINO_ -DBAUDRATE=38400
penguino: begin gccversion sizebefore build sizeafter end
                    mv $(TARGET).hex stk500boot_v2_penguino.hex
```

і переносяться вгору цього файлу. Компілятор (наприклад, AVR Studio з WinAVR [5]) при обробці Makefile вибере першу модель плати (мікроконтроллер) за списком. Після цього отриманий файл з розширенням *.hex можна прошивати в завантажувальну область за допомогою звичайного програматора.

Для ATmega16 bootloader компілюється згідно з описом, представленому в джерелі [8].

Як уже зазначалося, модуль WIZ812MJ (рис.1) підключається до мікроконтролера по шині SPI. Цей модуль виступає проміжним інтерфейсом між мережею Ethernet і деяким кінцевим пристроєм (мікроконтролером). Він заснований на мікросхемі W5100, що апаратно підтримує стек TCP/IP, має вбудований блок фізичного рівня PHY. Для полегшення підключення до мережі він містить розняття RJ-45.

Рис. 1. Модуль WIZ812MJ

На рисунку 2 показані два дворядних штирьових роз'ємну модуля WIZ812MJ з підключенням їх до портів налагоджувальний комплексу AVR-EASY-KIT, на якому встановлені мікроконтролер ATmega32, SD карта пам'яті, LCD індикатор WH1602B-YYH-CTK на базі HD44780, датчик температури DS18B20 і три світлодіода , що імітують роботу 3-х виконавчих пристроїв.

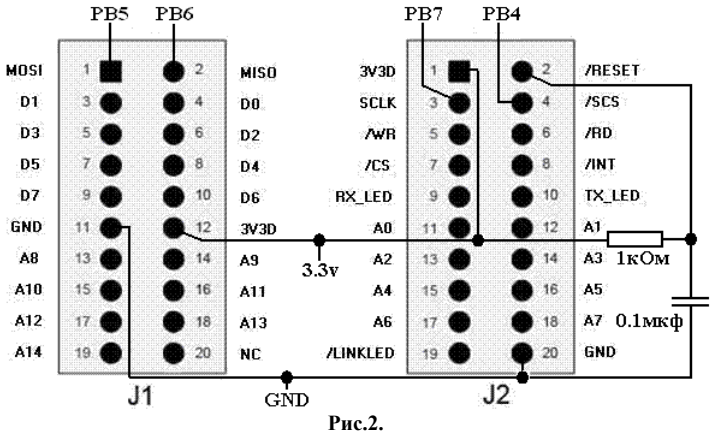

Рис.2.

Нижче наводиться програма на мові Wiring з докладними коментарями, по якій мікроконтролер спільно з модулем WIZ812MJ реалізує функції web-сервера, що керує виконавчими пристроями та знімає показання.

```
#include <OneWire.h> // Підключаємо опис бібліотеки шини OneWire
#include <SPI.h>     // Підключаємо опис бібліотеки шини SPI
```

```
#include <Ethernet.h>// Підключаємо опис бібліотеки Ethernet
#include <SD.h>  // Підключаємо опис бібліотеки для роботи з SD картою пам'яті
#include <LiquidCrystal.h>// Підключаємо опис бібліотеки для роботи з LCD
OneWire ds(6);//6-й вивід визначений як вивід роботи з шиною OneWire для DS18B20
File myFile;
//Здається будь-яку MAC адресу, що не збігається з іншими MAC адресами в мережі
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
//Здається будь-яку IP адресу з мережі 192.168.1.x, шлюз за замовчуванням, маска
// мережі
IPAddress ip(192,168,1,100);
IPAddress gateway (192,168,1,1);
IPAddress subnet(255,255,255,0);
LiquidCrystal lcd(14, 16, 18, 19, 20, 21);//Перерахування виводів контролера,
// які підключаться до LCD
byte rip[]={0,0,0,0}; // Поміщаємо сюди IP-адреса віддаленого клієнта
EthernetServer server(80);  // Створюємо сервер, який слухає 80-й порт
byte gotAMessage = 1;
const int analogInPin = A3;// На цей вивід знімаємо напругу
float sensorValue = 0;
float outputValue = 0;
void setup() {
pinMode(3, OUTPUT);   // 3-й вивід визначений як вихід для червоного світлодіода
pinMode(5, OUTPUT);   // 5-й вивід визначений як вихід для зеленого світлодіода
pinMode(7, OUTPUT);   // 7-й вивід визначений як вихід для синього світлодіода
lcd.begin(16, 2); // Ініціалізувати LCD, що має по 16 символів в 2-х рядках
lcd.print("      Wait");// Виводимо на 1-му рядку
lcd.setCursor(0, 1);// Переводимо курсор на 2-й рядок
lcd.print("   connection");// Виводимо на 2-й рядок
  Ethernet.begin(mac, ip, gateway, subnet);  // Ініціалізувати Ethernet модуль
  server.begin();  // Починаємо очікувати з'єднань на 80 порту
 // SD карта підключається до SPI шині. Лінія CS, відповідна вибору
 // пристрої, підключена до 8-у висновку.
SD.begin(8);  // Ініціалізувати SD модуль.
}
float temp()  // Підпрограма визначення температури з датчика DS18B20 (Приклад з
// бібліотеки OneWire).
{
  byte I, data[10], addr[8];
  float celsius;
  ds.search(addr);
  ds.reset();
  ds.select(addr);
  ds.write(0x44,1);  // start conversion, with parasite power on at the end
  delay(1000);
  ds.reset();
  ds.select(addr);
  ds.write(0xBE);             // Read Scratchpad
  for ( i = 0; i < 9; i++) { // we need 9 bytes
    data[i] = ds.read();}
  int raw = (data[1] << 8) | data[0];
    unsigned char t_mask[4] = {0x7, 0x3, 0x1, 0x0};
    byte cfg = (data[4] & 0x60) >> 5;
    raw &= ~t_mask[cfg];
  celsius = (float)raw / 16;
  return celsius;
}
void filewrite(EthernetClient client,char fname[])  //Підпрограма
// посимвольного читання з SD карти файлів з іменем,
// вказаному в fname[] і посимвольной передачі зчитаного клієнтові.
{
  myFile = SD.open(fname);        // Відкриваємо файл
    while (myFile.available())    // Якщо відкриття файлу пройшло успішно, то
        {
       char c=myFile.read();     // зберігаємо лічений символ у змінній c і
        client.write(c);          // і пересилаємо його клієнту.
         }
       myFile.close();            // Закриваємо файл.
 }
void loop()
{
 // Якщо хтось встановив з'єднання з нашим сервером, наступна строчка
 // створить об'єкт-клієнт.
```

```
  EthernetClient client = server.available();
  if (client) {
 // Визначаємо IP-адресу клієнта і роздруковуємо його на LCD.
       if (gotAMessage==1) {
        lcd.begin(16, 2);
      lcd.print("Client. RemoteIP");
          client.getRemoteIP(rip);
          lcd.setCursor(0, 1);
     for (int bcount= 0; bcount < 4; bcount++)
     {
        lcd.print(rip[bcount], DEC);
        if (bcount<3) lcd.print(".");
     }
       gotAMessage = 0;
       }
// Якщо client не нульовий (тобто з'єднання є)
// то хтось підключився. Згідно протоколу HTTP клієнт
// шле досить складний запит (див. приклад запитів), але він не розбирається -
// досить дочекатися закінчення запиту.
// Запит закінчується порожнім рядком, програма дочекається
// отримання символу '\n', перед яким теж були отримані '\n' і '\r'.
// Згідно HTTP після кожного рядка заголовка слідують символи '\n' і '\r'.
// currentLineIsBlank - це змінна-прапор. Вона дорівнює true, якщо
// знов отриманий від клієнта рядок порожній, тобто в отриманих даних
// не зустрілося символів відмінних від '\n' і '\r'.
// Будемо виставляти змінну currentLineIsBlank в false при
// отримання будь відмінного від '\n' і '\r' символу.
// Однак, спочатку currentLineIsBlank = true
 boolean currentLineIsBlank = true;
 byte k=0;//k=0 у разі, якщо читається тільки перший рядок заголовка запиту
     byte kk=0;
    String buffer = "";
    while (client.connected()) {  // Поки клієнт підключений
      if (client.available()) {   // і якщо від нього прийшов символ
        char c = client.read();   // то цей символ читається
// У рядковий змінної buffer зберігаємо тільки перший рядок запиту
      if(k==0) buffer+=c;
// Якщо отриманий перехід рядка ('\n') і currentLineIsBlank == true,
// значить отримана порожній рядок, тобто запит клієнта закінчено -
// можна слати відповідь (відповідь сервера)
        if (c == '\n' && currentLineIsBlank) {
// Посилаємо стандартний HTTP заголовок:
          client.println("HTTP/1.1 200 OK");
          client.println("Content-Type: text/html; charset=utf-8");
          client.println();  // Заголовок теж закінчується порожнім рядком!
// Зчитуємо з SD файл з ім'ям h1.txt з HTML текстом і посимвольно посилаємо його
// клієнту
             filewrite(client,"h1.txt");
// Якщо на 3-му виході логічна одиниця
         if (digitalRead(3)) {
//то клієнтові (браузеру) відправляємо текст, записаний на SD у файлі led1.txt
        filewrite(client,"led1.txt");
        } else {
// в іншому випадку - текст з файлу led2.txt.
        filewrite(client,"led2.txt");
        }
// Аналогічно для 5 і 7-го виходів
   if(digitalRead(5)){filewrite(client,"le1.txt");}
   else{filewrite(client,"le2.txt");}
    if(digitalRead(7)){filewrite(client,"led3.txt");}
    else{filewrite(client,"led4.txt");}
// Далі клієнту передаються форма з радіо-кнопками для вибору режиму роботи
// світлодіодів, і текст «температура»
    filewrite(client,"ha2.txt");
// Роздруківка на клієнті температури
     client.println(temp());
     filewrite(client,"vol.txt");//Роздруківка тексту «Напряжение, подаваемое на
// стабилизатор»
     sensorValue = analogRead(analogInPin);// Зчитується аналоговий вхід A0.
     outputValue = 6.24*5.0*sensorValue/1023;// Розраховується напруга
// по дільнику напруги на вході стабілізатора напруги плати.
       client.print(outputValue); // Роздруковується напруга на клієнті
```

```
// Якщо прапор kk = 1, то клієнтові посимвольно передається текст цієї програми,
// який зчитується з SD карти (файл test.txt).
if(kk==1)   {filewrite(client,"test.txt");}
// Після передачі даних сервером можна виходити з циклу while і розривати
// з'єднання з клієнтом (по команді client.stop ())
     break;
        }
if (c == '\n') {// Якщо отриманий вдруге перехід рядка, значить почалася
  // новарядок. Виставляємо прапор currentLineIsBlank = true
      currentLineIsBlank = true;
      buffer="";// Після кожного переходу до нового рядка обнуляем строкову
                // змінну buffer.
        }
else if (c == '\r')//Якщо отриманий символ '\r' (відразу за ним буде йти символ
// '\n'), не чекаючи переходу до нового рядка, виконуємо перевірку умов:
{
if(buffer.indexOf("GET /")>=0) k=1;
// 1. Якщо це перший рядок заголовка запиту, встановлюємо k = 1 для того,
// щоб рядковий змінної buffer не привласнювати наступні рядки заголовка.
if(buffer.indexOf("GET / HTTP")>=0) kk=1;
// 2. Якщо в першому рядку запиту не передається параметрів
// перемикання світлодіодів, то kk = 1, що викличе роздруківку тексту програми.
char r=buffer[8]; // Вибірка символу, відповідного стану червоного
// світлодіода (1-включений, 0-викл.).
    char g=buffer[12]; // ... - зеленого світлодіода
    char b=buffer[16]; // ... - синього світлодіода
   if( r==0x31) digitalWrite(3,HIGH); // Якщо для червоного світлодіода символ
// к дорівнює 1 (0x31) - запалюємо світлодіод.
if(r==0x30) digitalWrite(3,LOW);//Якщо r дорівнює 0 (0x30)-вимикаємо світлодіод.
if( g==0x31) digitalWrite(5,HIGH); // Аналогічно для зеленого
if( g==0x30) digitalWrite(5,LOW);
if( b==0x31) digitalWrite(7,HIGH); // і синього світлодіодів
if( b==0x30) digitalWrite(7,LOW);
      }
   else {    // якщо отримано будь-який інший символ, який відрізняється
// від повернення каретки ('\r'), значить одержувана рядок не порожній
          currentLineIsBlank = false;
        }
      }
    }
// Робимо паузу, щоб дані могли піти з сервера,
    delay(10);
// друкуємо, що клієнт від'єднався
    gotAMessage = 1; lcd.begin(16, 2); lcd.print("     Client");
    lcd.setCursor(0, 1); lcd.print("   disconnect");
// і розриваємо з'єднання з клієнтом.
    client.stop();
  }
}
```

Нижче наведені приклади заголовків запитів від клієнта.

1. Запит при зверненні до сервера за адресою: http://192.168.1.100 GET / HTTP/1.1

```
Host: 192.168.1.100
User-Agent: Mozilla/5.0 (Windows NT 5.1; rv:18.0) Gecko/20100101 Firefox/18.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ru-RU,ru;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Connection: keep-alive
```

2. Запит при зверненні до сервера за адресою: http://192.168.1.100/?r=1&g=0&b=1
для включення червоного і синього світлодіодів. Зелений вимкнений.

```
GET /?r=1&g=0&b=1 HTTP/1.1
Host: 192.168.1.100
User-Agent: Mozilla/5.0 (Windows NT 5.1; rv:18.0) Gecko/20100101 Firefox/18.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ru-RU,ru;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.100/?r=1&g=0&b=0
Connection: keep-alive
```

На рисунку 3 представлений приклад web - сторінки після подачі команди включення виконавчих механізмів, відповідних червоному і синьому світлодіодам.

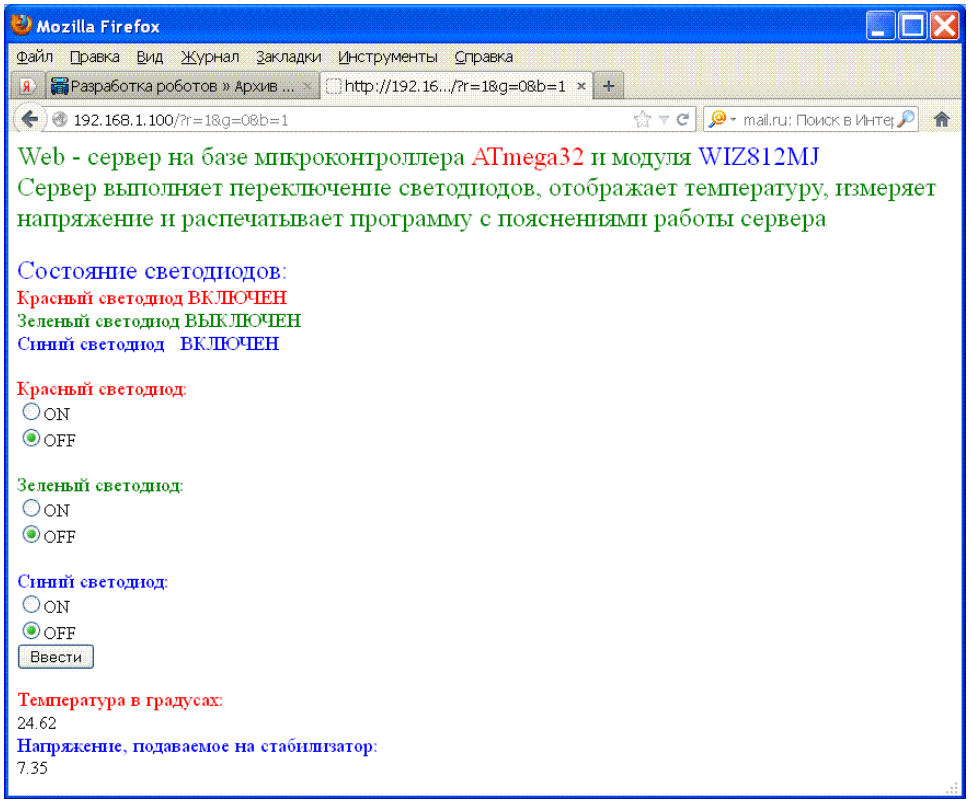

Рис.3.

Висновки

1. Показано можливість використання програмної частини проекту Arduino для підключення різних модулів до сімейства мікроконтролерів AVR, які не описані в середовищі Arduino.

2. Виконана практична стикування мікроконтролера ATmega32 з модулем WIZ812MJ на основі бібліотек середовища Arduino з метою віддаленого управління виконавчими пристроями і зняття параметрів через TCP/IP мережу.

3. Побудований на базі ATmega32 і модуля WIZ812MJ реально працюючий web - сервер, що виконує функцію управління трьома виконавчими пристроями. Встановлено, що для вирішення поставленого завдання мікроконтролер повинен мати не менше 32кБайт програмної пам'яті і 2кБайт оперативної.

4. Показано можливість використання об'єктно-орієнтованого програмування для швидкого створення досить складних проектів на базі мікроконтролерів.

Література

1. Arduino. [Electronic resource]. – Mode of access: http://www.arduino.cc/, 2013.

2. Проект CraftDuino. [Electronic resource]. – Mode of access: http://www.robocraft.ru/page/summary/, 2013.

3. Отладочный комплекс AVR-EASY-KIT. [Electronic resource]. – Mode of access: http://kosmodrom.ua/razrabotka/avreasy5.php

4. Arduino bootLoader STK500V2. [Electronic resource]. – Mode of access: http://robot-develop.org/archives/2290, 2011.

5. Белов А.В. Создаем устройства на микроконтроллерах. – СПб.: Наука и Техника, 2007. – 304 с.: ил.

6. Модуль WIZ812MJ. [Electronic resource]. – Mode of access: http://www.mt-system.ru/sites/default/files/docs/documents/wiz812mj_datasheet_v_1.0.pdf, 2008.

7. Долгушин С. Работаем с Vinculum II и W5100. USB от FTDI + Ethernet от Wiznet. // Компоненты и технологии. – 2011г. – № 125. – с. 108-111.

8. AVR. Учебный Курс. [Electronic resource]. – Mode of access: Использование Bootloader'a. http://easyelectronics.ru/avr-uchebnyj-kurs-ispolzovanie-bootloadera.html, 2008.

Надійшла до редакції
9.2.2013 р.