ТехноСфера

«ESP-12F WeMos D1 WiFi» + «Arduino UNO» + внешнее Приложение

Эксперименты, тесты, разработка соответствующих управляющих скриптов и т.д.

Важно!

1. Данный пример является продолжением цепочки примеров, приведенных «ранее».
2. Здесь (в этом документе) нет подробных пояснений по используемым скетчам (см. пункт 1, выше).
3. Крайне настоятельно рекомендуется предварительно ознакомиться со всей информацией, приведенной ранее, начиная с:

`https://roamer55.ru/main_programming/arduino/arduino_technosphere_000/`

Оглавление

Пример (26.11.2016)

«Смешались в кучу кони, люди…» (с) Лермонтов М.Ю. (Бородино).

Применительно к данному случаю: на «несчастный» Arduino UNO навешана куча «железа».
Оправдание этому – «приобретения опыта ради и экспериментов для…».

Рабочее название этого «гибрида»: «Метеостанция с охранной сигнализацией».

Функционал на уровне «местной автоматики»:
1. Данные о температуре и влажности от датчика DHT-11;
2. Данные об освещенности от датчика KY-018;
3. Управление «внешним» освещением (включение/отключение 4-х светодиодов на основе данных об освещенности);
4. Охранная сигнализация (PIR-датчик + зуммер + красный светодиод).

Функционал в рамках информационного обмена с внешним приложением (по запросу от внешнего приложения):
1. Отправка данных о температуре и влажности;
2. Отправка данных об освещенности (светло/темно);
3. Отправка данных о «чужих в охраняемой зоне» (есть/нет);
4. Изменение режима сигнализации («звуковая сирена + красный светодиод» или только «красный светодиод»).

Примечание – данный пример является продолжением цепочки примеров, приведенных здесь:
`https://roamer55.ru/main_programming/arduino/arduino_technosphere_000/arduino_technosphere_000_005/`

Для тестирования «всего этого» сформирована простая программка `My_TS_WS.exe` (см. ниже).

Схема соединений

На рисунке 1 приведена соответствующая схема соединений.

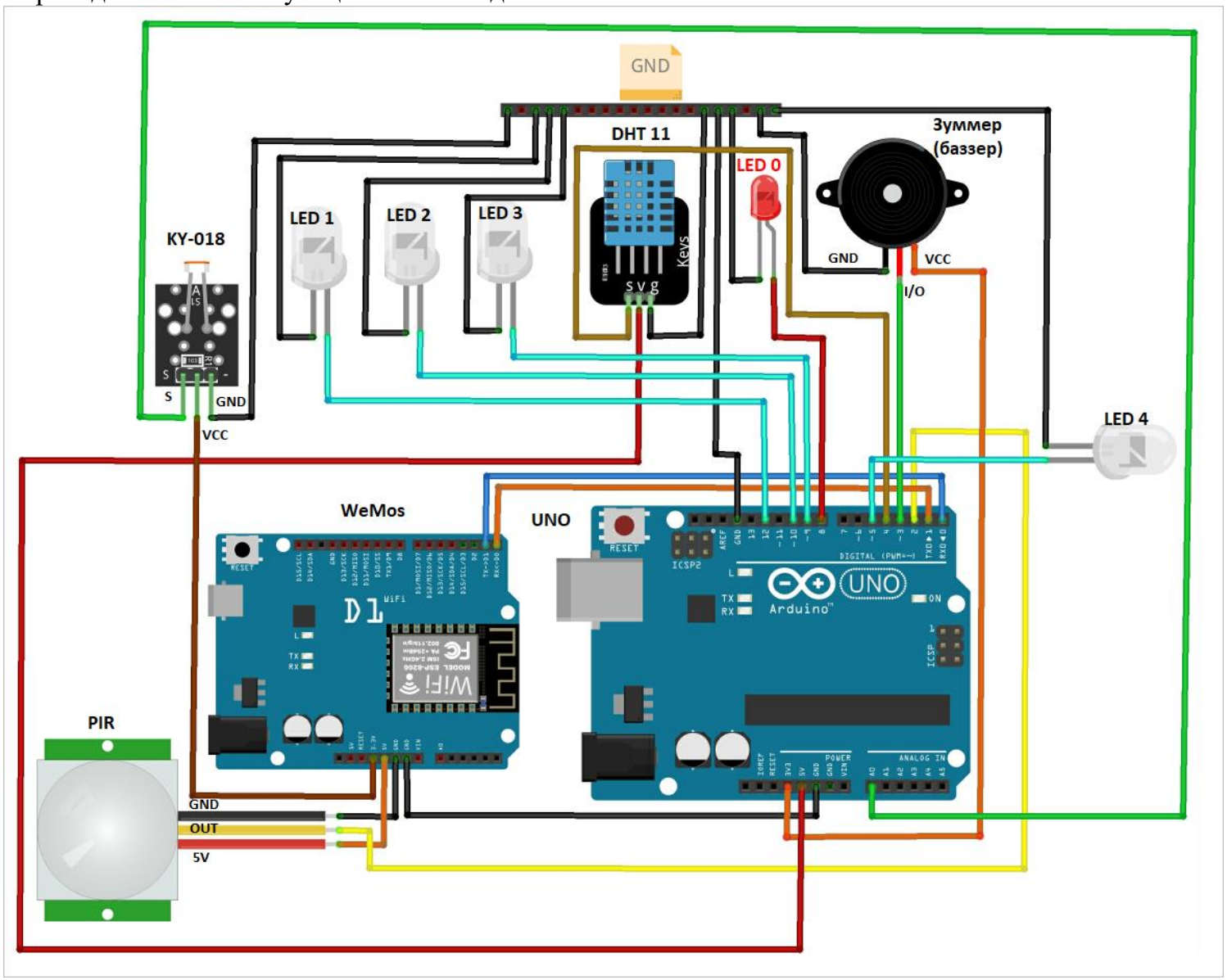


Рисунок 1 – Схема соединений для данного Примера (см., также, таблицы 1 и 2)

Таблица 1 – Обозначения элементов на рисунке 1

Обозначение элемента	Наименование
1	2
WeMOS	ESP-12F WeMos D1 WiFi
UNO	Arduino Uno
PIR	Датчик движения HC-SR501 (охранная сигнализация)
KY-018	Модуль фоторезистора KY-018 (внешнее освещение)
DHT 11	Датчик температуры и относительной влажности DHT 11
Зуммер (баззер)	Модуль зуммера (охранная сигнализация)
LED 0	Красный светодиод (охранная сигнализация)
LED 1 … LED 4	Белые светодиоды (внешнее освещение)
GND	Блок контактов GND («земля»)

Таблица 2 – Таблица соединений (см. рисунок 1)

Внешний элемент (контакт)	Arduino UNO (контакт)	ESP-12F WeMos D1 WiFi (контакт)	GND (блок контактов)
1	2	3	4
PIR-датчик (сигнализация)			
GND	–	GND	–
OUT	2	–	–
5V	–	5V	–
KY-018 (освещение)			
S	A0	–	–
VCC	–	3.3 V	–
GND	–	–	GND
DHT 11 (температура + влажность)			
S	4	–	–
V	5V	–	–
G	–	–	GND
Зуммер (сигнализация)			
GND	–	–	GND
I/O	3	–	–

Внешний элемент (контакт)	Arduino UNO (контакт)	ESP-12F WeMos D1 WiFi (контакт)	GND (блок контактов)
1	2	3	4
VCC	3.3 V	–	–
LED 0. Светодиод (сигнализация)			
“+”	8	–	–
“–”	–	–	GND
LED 1. Светодиод (освещение)			
“+”	12	–	–
“–”	–	–	GND
LED 2. Светодиод (освещение)			
“+”	10	–	–
“–”	–	–	GND
LED 3. Светодиод (освещение)			
“+”	9	–	–
“–”	–	–	GND
LED 4. Светодиод (освещение)			
“+”	5	–	–
“–”	–	–	GND
–	GND	GND	GND
–	0	1	–
–	1	0	–

На рисунке 2 приведен внешний вид гипотетической реализации этого «гибрида» (т.е., так это <u>могло бы быть…</u>), см., также, таблицу 3).

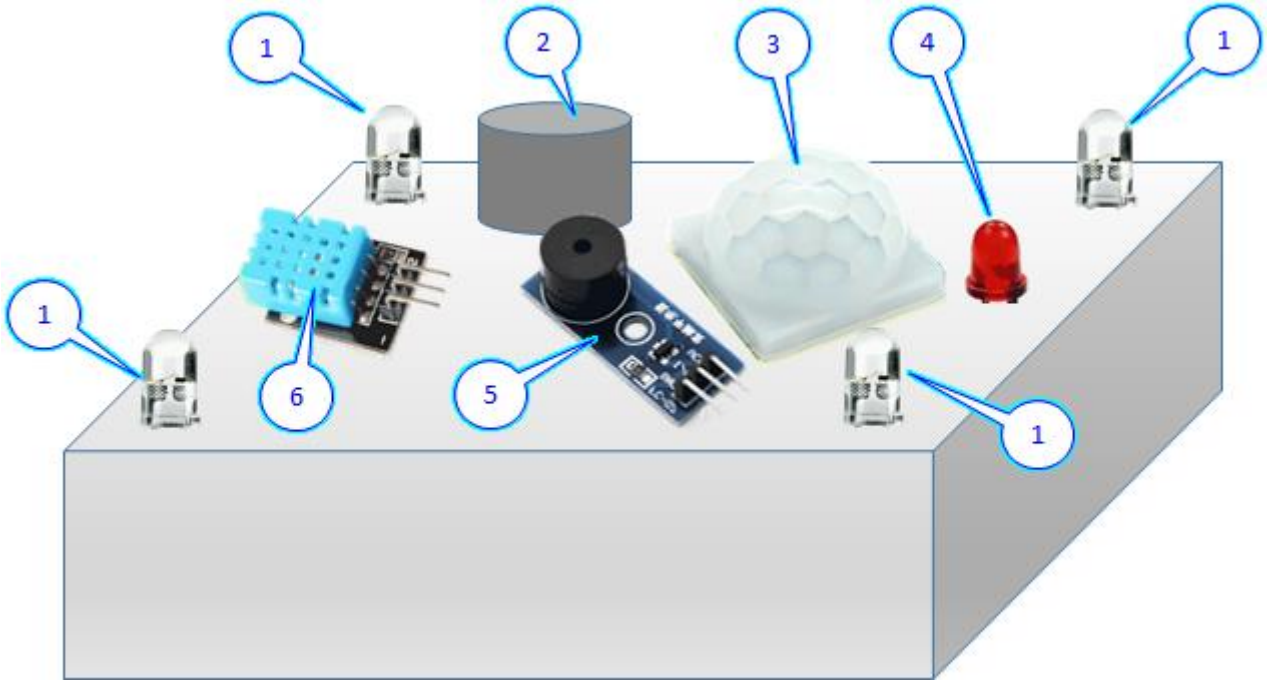


Рисунок 2 – «Хотели, как лучше…» ☺

Таблица 3 – Список выносок на рисунке 2

Выноски	Обозначение
1	2
1	Светодиоды внешнего освещения (LED 1 … LED 4)
2	Контейнер для размещения датчика KY-018
3	PIR-датчик (сигнализация)
4	Красный светодиод (сигнализация)
5	Зуммер (сигнализация)
6	Датчик температуры и относительной влажности DHT 11

Но… По причине того, что «все это» − лишь подготовительные действия (эксперименты), Автор решил ограничиться вот таким «энтропийным» вариантом (см. рисунок 3).

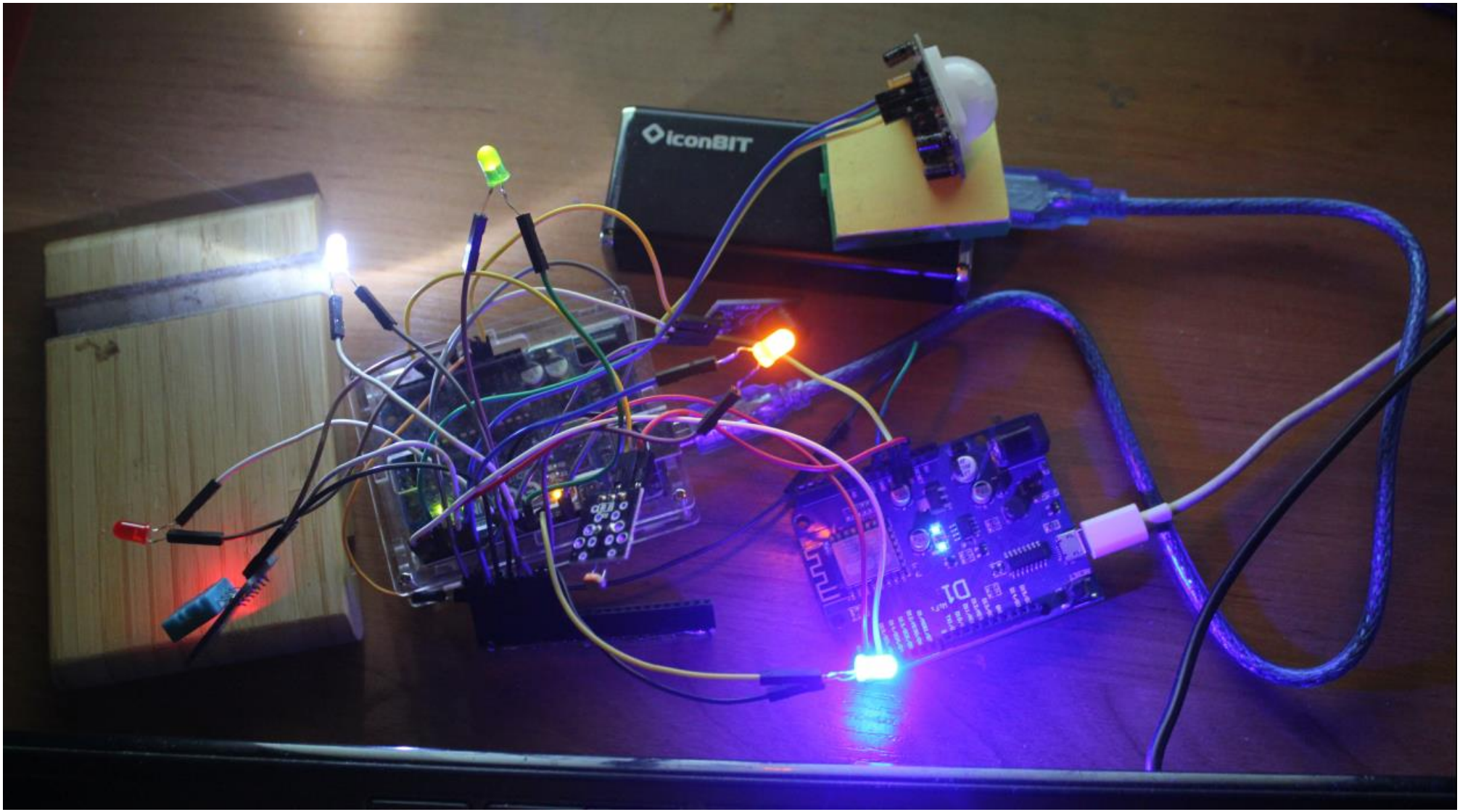


Рисунок 3 – «А получилось – как всегда…». Но «железо» работает… И вся система (в целом) – тоже… ☺

Скетч (для ESP-12F WeMos D1 WiFi)

```
/*
   Дата актуальности: 26.11.2021
   Скетч для ESP-12F WeMos D1 WiFi, сопряженного с Arduino UNO
   Назначение - обеспечение информационного обмена:
     1. "ESP-12F WeMos D1 WiFi" с "внешним" ПО через Wi-Fi;
     2. "ESP-12F WeMos D1 WiFi" с сопряженным Arduino UNO через "Tx/Rx";
     3. "Внешнего" ПО с сопряженным Arduino UNO через "ESP-12F WeMos D1 WiFi".
   Важно!
     Для того, чтобы не связываться с нюансами парсинга HTTP-запросов принято,
     что в строке HTTP-запроса вместо символа "точка с запятой" используется цепочка символов z9z
   См., также, https://roamer55.ru/main_programming/arduino/arduino_technosphere_000/
 */

#include <ESP8266WebServer.h>

//.................................
//Идентификация wi-fi сети и этого устройства
const char* ssid = "ИмяТвоейСети"; // Указываем имя существующей точки доступа (wi-fi сеть)
const char* password = "ПарольКСети"; // Указываем пароль существующей точки доступа (wi-fi сеть)
const String id_this = "ED0EBECDA1A74DDB8B666351F57734D2"; // Уникальный ID этого устройства
//.................................
//.................................
//"Настроечные" переменные
bool ItsMode_Debug = false;      //Режим отладки. Значение true актуально только, если это устройство подсоединено к ПК
через USB (COM-порт)
bool LED_BUILTIN_SYS = true;     //Флаг, определяющий режим использования встроенного светодиода
int SERIAL_READLN_DELAY = 5000;  //Время ожидания ответа от Arduino (мсек)
//.................................
//.................................
//Вспомогательные переменные (используются в оперативных целях)
String param0_name = "";
String param0_val = "";
String sInfo = "";
int pin_num = 0;
bool itsok=false;
int va = 0;
int v = LOW;
//................................

ESP8266WebServer server(80);

void setup(void)
{
```

```
  //Установка режима встроенного светодиода: OutPut
  pinMode(LED_BUILTIN, OUTPUT);
  //"Выключить" встроенный светодиод
  digitalWrite(LED_BUILTIN, HIGH);

  //Инициализация последовательного соединения со скоростью 9600
  Serial.begin(9600);

  //Инициализация Wi-Fi модуля
  WiFi.mode(WIFI_STA); // Установка Wi-Fi модуля в режим клиента (STA)
  WiFi.begin(ssid, password); // Подключение к сети

  // Ожидание подключения к сети
  while (WiFi.status() != WL_CONNECTED) { delay(500); }

  //Если включен режим отладки, то по последовательному соединению
  //отправляется соотв. информ. блок
  if (ItsMode_Debug==true)
  {
     Serial.println("");
     Serial.println("-----------------------");
     Serial.print("WI-FI network: ");
     Serial.println(ssid);
     Serial.print("IP address: ");
     Serial.println(WiFi.localIP());
     Serial.print("My ID: ");
     Serial.println(id_this);
     Serial.println("-----------------------");
  }

// ------------------------------------------
// Обработчики
  //Отработка запроса на получение общей информации
  server.on("/", handleRoot);
  //Отработка запроса на получение ID устройства
  server.on("/who_is_it", []() { who_is_it(); });
  //Отработка запроса на изменение значения переменной LED_BUILTIN_SYS (см. выше)
  server.on("/led_builtin_sys", []() { led_builtin_sys(); });
  //Отработка запроса на изменение состояния заданного Пина
  server.on("/pin_write", []() { pin_write(); });
  //Отработка запроса на получение состояния заданного Пина
  server.on("/pin_read", []() { pin_read(); });
  //Отработка запроса на установку режима заданного Пина (Input или Output)
  server.on("/pin_mode", []() { pin_mode(); });
```

```
  //Отработка запроса на получение состояния всех Пинов
  server.on("/pins_vals_get", []() { pins_vals_get(); });
  //Отработка запроса на получение или изменение значения переменной SERIAL_READLN_DELAY  (см. выше)
  server.on("/serial_readln_delay", []() { serial_readln_delay(); });
  //Отработка запроса на отсылку данных на сопряженный Arduino
  server.on("/serial_println", []() { serial_println(); });
  //Отработка запроса на получение данных от сопряженного Arduino
  server.on("/serial_readln", []() { serial_readln(); });
// -----------------------------------------

  // Вызывается, когда обработчик не назначен
  server.onNotFound(handleNotFound);

  // Запуск сервера
  server.begin();
}

void loop(void)
{
  server.handleClient();
}

void handleRoot()
{
  // Обработчик запроса клиента по корневому адресу (возвращается список команд)
  server.send(200,  "text/plain",  id_this+":\n"+"Commands:  \n  who_is_it  \n  pin_mode  \n  pin_write  \n  pin_read  \n
pins_vals_get \n led_builtin_sys \n serial_println \n serial_readln \n serial_readln_delay");
}

bool serial_wait(unsigned long dt)
{ //Ожидание ответа по последовательному соединению заданное время (dt) в миллисекундах
  bool res=false;
  if (dt < 0) { dt = 0; }
  unsigned long t0 = millis();
  bool yes_exit = false;
  while (yes_exit == false)
  {
    if (Serial.available() > 0)
    {
       res = true;
       yes_exit = true;
    }
    else
    {
```

```
      if ((millis() - t0) > dt)
      {
      yes_exit = true;
      }
    }
  }
  return res;
}

int StringToPin(String s)
{ //Конвертация строкового значения в номер Пина
  int res;
  res = -1;
  s.trim();
  s.toUpperCase();
  if (s == "D0") { res = D0; };
  if (s == "D1") { res = D1; };
  if (s == "D2") { res = D2; };
  if (s == "D3") { res = D3; };
  if (s == "D4") { res = D4; };
  if (s == "D5") { res = D5; };
  if (s == "D6") { res = D6; };
  if (s == "D7") { res = D7; };
  if (s == "D8") { res = D8; };
  if (s == "D9") { res = D9; };
  if (s == "D10") { res = D10; };
  if (s == "D11") { res = D11; };
  if (s == "D12") { res = D12; };
  if (s == "D13") { res = D13; };
  if (s == "D14") { res = D14; };
  if (s == "D15") { res = D15; };
  if (s == "LED_BUILTIN") { res = LED_BUILTIN; };
  if (s == "2") { res = LED_BUILTIN; };
  if (s == "A0") { res = A0; };
  return res;
}

void LED_BUILTIN_SYS_ON_OFF(bool itsON)
{
  /*
    В начале каждого обработчика "зажжем" встроенный светодиод (itsON = true).
    А в конйе каждого обработчика "погасим" его (itsON = false).
  */
  if (itsON == true)
```

```
  {
     if (LED_BUILTIN_SYS == true) {  digitalWrite(LED_BUILTIN, LOW);}
  }
  else
  {
     if (LED_BUILTIN_SYS == true)
     {
        delay(200);
        digitalWrite(LED_BUILTIN, HIGH);
     }
  }
}

//*****************************************************
// Обработчики...
void handleNotFound()
{ // Ошибка 404 (некорректный URL)
  String message = "File Not Found\n\n";
  message += "URI: ";
  message += server.uri();
  message += "\nMethod: ";
  message += (server.method() == HTTP_GET) ? "GET" : "POST";
  message += "\nArguments: ";
  message += server.args();
  message += "\n";
  for (uint8_t i = 0; i < server.args(); i++)
  {
    message += " " + server.argName(i) + ": " + server.arg(i) + "\n";
  }
  server.send(404, "text/plain", id_this+":\n"+message);
}

void who_is_it()
{ //Запрос на получение ID устройства
    //Пример:   http://192.168.0.102/who_is_it
    LED_BUILTIN_SYS_ON_OFF(true);
    server.send(200, "text/plain", id_this);
    LED_BUILTIN_SYS_ON_OFF(false);
}

void serial_readln()
{ //Запрос на получение данных от сопряженного Arduino
    /*
     Примеры:
```

```
        http://192.168.0.103/serial_readln
        http://192.168.0.103/serial_readln?098BB20A0CEE4FBBAB7BDC71EB2A8778 z9z pin_12 z9z pin_13 z9z pin_A2
        http://192.168.0.103/serial_readln?device_all z9z info_all
        http://192.168.0.103/serial_readln?device_all z9z pin_12 z9z pin_13 z9z pin_A2
    */
    LED_BUILTIN_SYS_ON_OFF(true);
    String scomm = "";
    for (uint8_t i = 0; i < server.args(); i++)
    {
      param0_name = (String) server.argName(i);
      param0_name.trim();
      if (param0_name != "") { param0_name += "="; } else { param0_name = " "; }
      param0_val = (String) server.arg(i);
      param0_val.trim();
      scomm += " " + param0_name + param0_val;
    }
    scomm.trim();
    if (scomm != "")
    {
       scomm.replace("=", " ");
       scomm.replace(" z9z ", "; ");
       scomm.trim();
    }
    if (scomm == "") {scomm = "device_all; info_all"; }
    Serial.println("GET: "+scomm); //передача команды на сопряженное устройство (Arduino)
    serial_wait(SERIAL_READLN_DELAY); //Ожидаем ответ Arduino заданное время
    String sInfo = "";
    while (Serial.available() > 0)
    {
      sInfo += (char)Serial.read();
      delay(2);
    }
    sInfo.trim();
    /*
    //Отладочный блок
    if (sInfo == "")
    {
        //sInfo = "A: 098BB20A0CEE4FBBAB7BDC71EB2A8778; mode=DEBUG; <[pin_1=1; pin_A0=222; pin_A3=18; pin_12=0;]>";
//отладка
        //sInfo = "A: 098BB20A0CEE4FBBAB7BDC71EB2A8778; mode=DEBUG; <[]>"; //отладка
    }
    */
    if (sInfo.length() < 3)
    {
```

```
        sInfo="";
    }
    else
    {
       String s = sInfo.substring(0, 2);
       if (s != "A:") { sInfo=""; }
    }
    if (sInfo.length() > 3)
    {
       String s2 = sInfo.substring(sInfo.length()-2, sInfo.length());
       if (s2 != "]>") { sInfo=""; }
    }
    if (sInfo == "")
    {
       sInfo = "NO INFO";
    }
    else
    {
       sInfo.setCharAt(0, ' ');
       sInfo.setCharAt(1, ' ');
       sInfo.trim();
    }
    server.send(200, "text/plain", id_this + ": "+sInfo);
    LED_BUILTIN_SYS_ON_OFF(false);
}

void serial_println()
{ //Запрос на отсылку данных на сопряженный Arduino
    /*
     Примеры:
       http://192.168.0.103/serial_println?098BB20A0CEE4FBBAB7BDC71EB2A8778 z9z pin_12=HIGH z9z pin_13=1
       http://192.168.0.103/serial_println?098BB20A0CEE4FBBAB7BDC71EB2A8778 z9z pin_12=LOW z9z pin_13=0
       http://192.168.0.103/serial_println?098BB20A0CEE4FBBAB7BDC71EB2A8778 z9z pin_A0=117 z9z pin_A3=99
       http://192.168.0.103/serial_println?098BB20A0CEE4FBBAB7BDC71EB2A8778 z9z pin_12=1 z9z pin_A3=99
    */
    LED_BUILTIN_SYS_ON_OFF(true);
    String scomm = "";
    for (uint8_t i = 0; i < server.args(); i++)
    {
      param0_name = (String) server.argName(i);
      param0_name.trim();
      if (param0_name != "") { param0_name += "="; } else { param0_name = " "; }
      param0_val = (String) server.arg(i);
      param0_val.trim();
```

```
      scomm += " " + param0_name + param0_val;
    }
    scomm.trim();
    if (scomm != "")
    {
       scomm.replace(" z9z ", "; ");
       Serial.println("SET: "+scomm); //передача команды на сопряженное устройство (Arduino)
       server.send(200, "text/plain", id_this + ": "+"OK; "+"serial_println"+"; "+scomm);
    }
    else
    {
       server.send(200, "text/plain", id_this + ": "+"Error. serial_println. Command is bad");
    }
    LED_BUILTIN_SYS_ON_OFF(false);
}

void serial_readln_delay()
{ //Запрос на получение или изменение значения переменной SERIAL_READLN_DELAY (см. выше)
    /*
     Примеры:
       http://192.168.0.103/serial_readln_delay
       http://192.168.0.103/serial_readln_delay?val=2000
       http://192.168.0.103/serial_readln_delay?val=500 - минимально возможное значение
    */
    LED_BUILTIN_SYS_ON_OFF(true);
    param0_name = (String) server.argName(0);
    param0_val = (String) server.arg(0);
    param0_val.trim();
    if (param0_val == "")
    {
       server.send(200,       "text/plain",       id_this       +       ":       "+"OK;       "+"serial_readln_delay"+";
SERIAL_READLN_DELAY="+SERIAL_READLN_DELAY);
    }
    else
    {
       va = param0_val.toInt();
       if (va <500) { va = 500; }
       SERIAL_READLN_DELAY = va;
       server.send(200,       "text/plain",       id_this       +       ":       "+"OK;       "+"serial_readln_delay"+";
SERIAL_READLN_DELAY="+SERIAL_READLN_DELAY);
    }
    LED_BUILTIN_SYS_ON_OFF(false);
}
```

```
void pin_mode()
{ //Запрос на установку режима заданного Пина (Input или Output)
    /*
     Примеры:
       http://192.168.0.103/pin_mode?LED_BUILTIN=OUTPUT
       http://192.168.0.103/pin_mode?D8=OUTPUT
       http://192.168.0.103/pin_mode?D10=INPUT
    */
    LED_BUILTIN_SYS_ON_OFF(true);
    param0_name = (String) server.argName(0);
    param0_val = (String) server.arg(0);
    pin_num = StringToPin(param0_name);
    itsok=false;
    if (pin_num>=0)
    {
       param0_val.toUpperCase();
       //if (param0_val == "OUTPUT")
       if ((param0_val == "OUT") || (param0_val == "OUTPUT"))
       {
          pinMode(pin_num, OUTPUT);
          itsok=true;
       }
       //if (param0_val == "INPUT")
       if ((param0_val == "IN") ||  (param0_val == "INPUT"))
       {
          pinMode(pin_num, INPUT);
          itsok=true;
       }
       if (itsok == true)
       {
          server.send(200, "text/plain", id_this + ": "+"OK; "+"pin_mode"+"; "+pin_num+"; "+param0_val);
       }
       else
       {
          server.send(200, "text/plain", id_this + ": "+"Error. pin_mode. "+pin_num+". +"+"Pin mode is bad");
       }
    }
    else
    {
      server.send(200, "text/plain", id_this + ": "+"Error. pin_mode. "+param0_name+". "+"Pin number is bad");
    }
    LED_BUILTIN_SYS_ON_OFF(false);
}
```

```
void pin_read()
{ //Запрос на получение состояния заданного Пина
    /*
     Примеры:
       http://192.168.0.103/pin_read?LED_BUILTIN
       http://192.168.0.103/pin_read?D8
       http://192.168.0.103/pin_read?A0
    */
    LED_BUILTIN_SYS_ON_OFF(true);
    param0_name = (String) server.argName(0);
    pin_num = StringToPin(param0_name);
    if (pin_num>=0)
    {
       if (pin_num == A0)
       {
          va = analogRead(pin_num);
          server.send(200, "text/plain", id_this + ": "+"OK; "+"pin_read"+"; "+pin_num+"; "+va);
       }
       else
       {
          v = digitalRead(pin_num);
          server.send(200, "text/plain", id_this + ": "+"OK; "+"pin_read"+"; "+pin_num+"; "+v);
       }
    }
    else
    {
      server.send(200, "text/plain", id_this + ": "+"Error. pin_read. "+param0_name+". "+"Pin number is bad");
    }
    LED_BUILTIN_SYS_ON_OFF(false);
}

void pin_write()
{ //Запрос на изменение состояния заданного Пина
    /*
     Примеры:
       http://192.168.0.103/pin_write?LED_BUILTIN=LOW
       http://192.168.0.103/pin_write?LED_BUILTIN=HIGH
       http://192.168.0.103/pin_write?D8=HIGH
       http://192.168.0.103/pin_write?A0=183
    */
    LED_BUILTIN_SYS_ON_OFF(true);
    param0_name = (String) server.argName(0);
    param0_val = (String) server.arg(0);
    pin_num = StringToPin(param0_name);
```

```
    itsok=false;
    if (pin_num>=0)
    {
       param0_val.toUpperCase();
       if (pin_num == A0)
       {
           va = param0_val.toInt();
           analogWrite(pin_num, va);
           server.send(200, "text/plain", id_this + ": "+"OK; "+"pin_write"+"; "+pin_num+"; "+va);
       }
       else
       {
          if ((param0_val == "LOW") || (param0_val == "0"))
          {
             digitalWrite(pin_num, LOW);
             itsok=true;
             server.send(200, "text/plain", id_this + ": "+"OK; "+"pin_write"+"; "+pin_num+"; "+LOW);
          }
          if ((param0_val == "HIGH") || (param0_val == "1"))
          {
             digitalWrite(pin_num, HIGH);
             itsok=true;
             server.send(200, "text/plain", id_this + ": "+"OK; "+"pin_write"+"; "+pin_num+"; "+HIGH);
          }
          if (itsok == false)
          {
             server.send(200, "text/plain", id_this + ": "+"Error. pin_write. "+pin_num+". "+"Value is bad");
          }
       }
    }
    else
    {
      server.send(200, "text/plain", id_this + ": "+"Error. pin_write. "+param0_name+". "+"Pin number is bad");
    }
    LED_BUILTIN_SYS_ON_OFF(false);
}

void led_builtin_sys()
{ //Запрос на изменение значения переменной LED_BUILTIN_SYS (см. выше)
    /*
     Примеры:
       http://192.168.0.103/led_builtin_sys?1
       http://192.168.0.103/led_builtin_sys?TRUE
       http://192.168.0.103/led_builtin_sys?0
```

```
      http://192.168.0.103/led_builtin_sys?FALSE
   */
   digitalWrite(LED_BUILTIN, LOW);
   param0_name = (String) server.argName(0);
   itsok=false;
   param0_name.toUpperCase();
   if ((param0_name == "FALSE") || (param0_name == "0"))
   {
      LED_BUILTIN_SYS = false;
      itsok=true;
   }
   if ((param0_name == "TRUE") || (param0_name == "1"))
   {
      LED_BUILTIN_SYS = true;
      itsok=true;
   }
   if (itsok == true)
   {
      server.send(200, "text/plain", id_this + ": "+"OK; "+"led_builtin_sys"+"; "+LED_BUILTIN_SYS);
   }
   else
   {
      server.send(200, "text/plain", id_this + ": "+"Error. led_builtin_sys. "+"Value is bad");
   }
   delay(200);
   digitalWrite(LED_BUILTIN, HIGH);
}


int pin_val_get(int pn, bool its_digit)
{ //Получить значение заданного Пина
  int res=-1;
  if (pn >= 0)
  {
     if (its_digit == true)
     {
        res = digitalRead(pn);
     }
     else
     {
        res = analogRead(pn);
     }
     String s = "";
     s = "pin_" + String(pn) + "=" + String(res) + "; ";
```

```
      sInfo = sInfo + s;
  }
  return res;
}

void pins_vals_get()
{
  //Отработка запроса на получение состояния всех Пинов
  //Пример:   http://192.168.0.102/pins_vals_get
  LED_BUILTIN_SYS_ON_OFF(true);
  sInfo = "";
  pin_val_get(D0, true);
  pin_val_get(D1, true);
  pin_val_get(D2, true);
  pin_val_get(D3, true);
  pin_val_get(D4, true);
  pin_val_get(D5, true);
  pin_val_get(D6, true);
  pin_val_get(D7, true);
  pin_val_get(D8, true);
  pin_val_get(D9, true);
  pin_val_get(D10, true);
  pin_val_get(D11, true);
  pin_val_get(D12, true);
  pin_val_get(D13, true);
  pin_val_get(D14, true);
  pin_val_get(D15, true);
  //pin_val_get(LED_BUILTIN, true);
  pin_val_get(A0, false);
  sInfo.trim();
  String sEndData = "]>";
  String sStartData = "<[";
  sInfo = sStartData + id_this+"; " + sInfo  + sEndData;
  server.send(200, "text/plain", id_this + ": "+sInfo);
  LED_BUILTIN_SYS_ON_OFF(false);
}
//*****************************************************
```

Скетч (для Arduino UNO)

```
/*
   Дата актуальности: 26.11.2021
   Скетч для Arduino UNO, сопряженного с ESP-12F WeMos D1 WiFi
   Назначение:
     1. Обеспечение информационного обмена между "ESP-12F WeMos D1 WiFi" и этим (сопряженным) Arduino UNO через "Tx/Rx".
     2. "Внешнее" управление режимом сигнлализации (с использованием пина 13)
   Были использованы материалы с сайтов:
        https://soltau.ru/index.php/arduino/item/373-kak-vypolnyat-parallelnye-zadachi-threads-v-programme-dlya-
arduino;
        http://arduino-kid.ru/lesson/podklyuchenie-pischalki-buzzer-k-plate-arduino-urok-no-3#google_vignette
   См., также, https://roamer55.ru/main_programming/arduino/arduino_technosphere_000/
*/

#include <Thread.h>  // подключение библиотеки ArduinoThread
                     // (https://github.com/ivanseidel/ArduinoThread/archive/master.zip)
#include "DHT.h"     // Подключение библиотекии DHT (для датчиков DHT-11 и DHT-22)

//.................................
const String id_this = "098BB20A0CEE4FBBAB7BDC71EB2A8778"; // Уникальный ID этого устройства
//.................................
//.................................
String sSysVar = "";  // системная переменная (именно в ней содержится информация, передаваемая к внешнему Приложению)
bool ItsMode_Debug = false; //Режим отладки. Значение true актуально только, если этот Arduino подсоединен к ПК через
USB (COM-порт)
//.................................
//.................................
//Вспомогательные переменные (используются в оперативных целях при обработке поступившей команды)
bool  InCommandProcessing  =   false;  //Флаг.  Если  TRUE  -  то  идет  обработка  команды  (не  факт,  что  надо...  -
поэкспериментировать)
String sComm = "";      // текст команды
String sAct = "";       // действие:  GET: или SET:
String sInfo = "";      // возвращаемая информация (в ответ на GET:)
String sParam = "";     // Входной параметр
String sParamName = ""; // Имя входного параметра
String sParamVal = "";  // Значение входного параметра
int va = 0;
int nParam = 0;
int pin_num = 0;
//................................

//=====================================================
//Специфика
//------------------------------
```

```
//"Сигнализация"
const int soundPin = 3;   //зуммер (баззер)
const int ledPin   = 8;   //красный светодиод
const int dt_sound = 500;      //интервал "срабатывания" сигнализации, мсек
Thread soundThread = Thread(); // создается поток управления сиреной (зуммером) и светодиодом
//PIR
const int pirPin = 2;       //номер пина для подключения PIR-датчика
bool pir_alien_yes = false; //флаг "Чужой в охраняемой зоне" (есть/нет) - для отправки к внешнему Приложению
//-------------------------------
//-------------------------------
//DHT
Thread dhtThread = Thread(); //поток управления DHT
const int dt_DHT = 2000;     //интервал "измерения" DHT, мсек
const int DHTpin = 4;        //номер пина для подключения DHT
float DHT_h_now = 0.0;       //текущее значение влажности
float DHT_h_avg = 0.0;       //среднее значение влажности
float DHT_h_min = 9999.0;    //минимальное значение влажности
float DHT_h_max = -9999.0;   //максимальное значение влажности
float DHT_t_now = 0.0;       //текущее значение температуры
float DHT_t_avg = 0.0;       //среднее значение температуры
float DHT_t_min = 9999.0;    //минимальное значение температуры
float DHT_t_max = -9999.0;   //максимальное значение температуры
DHT dht(DHTpin, DHT11);
//.........................
//.........................
//"Освещение"
int light_limit  = 400;  //Граница освещенности, когда надо включать/вЫключать "свет" (светодиоды light_pin_out1 ...
light_pin_out4)
int light_pin_in = A0;   //вход от фоторезистора
int light_pin_out1 = 12; //выход (освещение: светодиод LED 1)
int light_pin_out2 = 10; //выход (освещение: светодиод LED 2)
int light_pin_out3 =  9; //освещение: выход (светодиод LED 3)
int light_pin_out4 =  5; //освещение: выход (светодиод LED 4)
String light_What = "";  //Состояние "освещенности" (светло/темно) для отправки к внешнему Приложению
//.........................

//=========================================================

//**********************************
//**********************************
//**********************************
void setup()
{
  setup0();
```

```
  //-------------------------------------------------
  //Специфика...
  //digitalWrite(13, HIGH);  //для отладки
  //..........................
  pinMode(pirPin, INPUT);     //режим пина 2 (PIR)
  //..........................
  //..........................
  //Освещение
  pinMode(light_pin_out4, OUTPUT);    //режим пина (освещение)
  pinMode(light_pin_out3, OUTPUT);    //режим пина (освещение)
  pinMode(light_pin_out2, OUTPUT);    //режим пина (освещение)
  pinMode(light_pin_out1, OUTPUT);    //режим пина (освещение)
  digitalWrite(light_pin_out4, LOW);  //начальное состояние пина (освещение)
  digitalWrite(light_pin_out3, LOW);  //начальное состояние пина (освещение)
  digitalWrite(light_pin_out2, LOW);  //начальное состояние пина (освещение)
  digitalWrite(light_pin_out1, LOW);  //начальное состояние пина (освещение)
  //..........................
  //..........................
  //"Сигнализация"
  pinMode(soundPin, OUTPUT);  //режим пина 3 (зуммер)
  pinMode(ledPin, OUTPUT);    //режим пина 8 (крачный светодиод)
  soundThread.onRun(sound);          // назначение потоку задачи: sound() - зуммер
  soundThread.setInterval(dt_sound); // интервал срабатывания (для зуммера), мсек
  //..........................
  //..........................
  //DHT
  dhtThread.onRun(DHT_Action);
  dhtThread.setInterval(dt_DHT);
  dht.begin();
  //..........................
  //-------------------------------------------------
} //setup()

void loop()
{
    InCommandProcessing = false;
    if (Serial_Readln_If_Available() == true)
    {  //Поступила внешняя команда и она обработана
       //В переменной sAct хранится идентификатор действия (GET или SET)
       //Здесь - соответствующий, специфичный код...
       //Serial.println("sAct=" + sAct); //для отладки
    }
    else
    {
```

```
   //Здесь - соответствующий, специфичный код...
}
//----------------------------------------
//Специфика
//.........................
//DHT
if (dhtThread.shouldRun())
      dhtThread.run(); // запускается поток
//.........................
//.........................
//"Сирена"
//digitalWrite(13, HIGH); //отладка
int pirVal = digitalRead(pirPin); //Читать состояние PIR-датчика
if (pirVal == HIGH)
{  //Датчик сработал
   pir_alien_yes = true;
   int p13 = digitalRead(13); //Читать состояние пина 13
   if (p13 == LOW)
   {  //Звук разрешен
      //"Сирена"
      if (soundThread.shouldRun())
          soundThread.run(); // запускается поток
   }
   else
   {  //Звук запрещен
      digitalWrite(ledPin, HIGH);
      delay(2000);
   }
}
else
{
   //Нет движения
   pir_alien_yes = false;
   noTone(soundPin);  // вЫключение звука
   digitalWrite(ledPin, LOW); // вЫключение светодиода
}
//.........................
//.........................
//Освещение
float va0 = analogRead(light_pin_in);
if (va0 > light_limit)
{  // Темно - включаем "освещение"
   digitalWrite(light_pin_out4, HIGH);
   digitalWrite(light_pin_out3, HIGH);
```

```
      digitalWrite(light_pin_out2, HIGH);
      digitalWrite(light_pin_out1, HIGH);
      light_What = "DARK";
      if (ItsMode_Debug == true)
      {
         Serial.println("<<< DARK >>>");
      }
   }
   else
   { // Светло - вЫключаем "освещение"
      digitalWrite(light_pin_out4, LOW);
      digitalWrite(light_pin_out3, LOW);
      digitalWrite(light_pin_out2, LOW);
      digitalWrite(light_pin_out1, LOW);
      light_What = "LIGHT";
      if (ItsMode_Debug == true)
      {
         Serial.println("< LIGHT >");
      }
   }
   //.........................
   if (ItsMode_Debug==true) { delay(500); } // отладка
   //----------------------------------------


} //loop()
//**********************************
//**********************************
//**********************************

//=================================================================
//Специфика
//----------------------------------------
void DHT_Action()
{ //температура + влажность
  float h = dht.readHumidity(); //Данные по влажности
  float t = dht.readTemperature(); //Данные по температуре
  if (isnan(h) || isnan(t)) {  // Проверка. Если не удается считать показания, выводится «Ошибка считывания», и программа
завершает работу
     //Serial.println("Ошибка считывания");
  }
  else
  {
```

```
    DHT_t_now  = t; //текущее значение температуры
    DHT_h_now  = h; //текущее значение влажности
    if (InCommandProcessing == false)
    { //Внешняя команда НЕ обрабатывается. Можно вычислять новые значения
       //.............................
       //МиниМаксные значения
       if (DHT_h_min > DHT_h_now) { DHT_h_min = DHT_h_now; }
       if (DHT_t_min > DHT_t_now) { DHT_t_min = DHT_t_now; }
       if (DHT_h_max < DHT_h_now) { DHT_h_max = DHT_h_now; }
       if (DHT_t_max < DHT_t_now) { DHT_t_max = DHT_t_now; }
       //.............................
       //.............................
       //Вычисление средних значений
       DHT_h_avg = (DHT_h_max + DHT_h_min) / 2;
       DHT_t_avg = (DHT_t_max + DHT_t_min) / 2;
       //.............................
       //.............................
       //Режим отладки
       if (ItsMode_Debug == true)
       {
          SysVar_Value_Calc();
          Serial.println("sSysVar: "+sSysVar);
       }
       //.............................
    }
  }
}
//-----------------------------------------
//-----------------------------------------
// Сигнализация: Полицейская сирена + красный светодиод
void sound()
{
   const int i_led_max = 20;
   int i_sound_up=0;
   int i_sound_down=0;
   int i_led = 0;
   bool stat_led = true;
    digitalWrite(ledPin, HIGH);  // включить светодиод
    for (i_sound_up = 200; i_sound_up < 1500; i_sound_up += 10)
    { // цикл по диапазону частот от 200Гц до 1500Гц с шагом 10
        tone(soundPin, i_sound_up);
        //delay(10);
        delay(20);
        i_led = i_led + 1;
```

```
        if (i_led > i_led_max)
        {
            stat_led = !stat_led;
            i_led = 0;
            digitalWrite(ledPin, stat_led);  // включить/вЫключить светодиод
        }
    }
    for (i_sound_down = i_sound_up; i_sound_down > 200; i_sound_down -= 10)
    { // цикл по диапазону частот от 1500Гц до 200Гц с шагом 10
        tone(soundPin, i_sound_down);
        //delay(10);
        delay(20);
        i_led = i_led + 1;
        if (i_led > i_led_max)
        {
            stat_led = !stat_led;
            i_led = 0;
            digitalWrite(ledPin, stat_led);  // включить/вЫключить светодиод
        }
    }
    noTone(soundPin); // пауза
    digitalWrite(ledPin, LOW); // вЫключение светодиода
}
//-----------------------------------------
//=================================================================
//=================================================================
//Специфика
void SysVar_Value_Calc()
{ //Сформировать значение переменной sSysVar (sys_var) - для отправки во внешнее Приложение
  //Вычисляется специфично для конкретной задачи
  sSysVar = "";
  String s ="";
  s = s + "["+light_What+"]";
  s = s + "[";
  if (pir_alien_yes == true)
  {
     s = s + "ALIEN_YES";
  }
  else
  {
     s = s + "ALIEN_NO";
  }
  s = s + "]";
  s = s + "[T="+String(DHT_t_now)+"]";
```

```
  s = s + "[T_min="+String(DHT_t_min)+"]";
  s = s + "[T_avg="+String(DHT_t_avg)+"]";
  s = s + "[T_max="+String(DHT_t_max)+"]";
  s = s + "[H="+String(DHT_h_now)+"]";
  s = s + "[H_min="+String(DHT_h_min)+"]";
  s = s + "[H_avg="+String(DHT_h_avg)+"]";
  s = s + "[H_max="+String(DHT_h_max)+"]";
  s.trim();
  if (s == "") { s = "NO DATA"; }
  sSysVar = s;
}
//=================================================================

void setup0()
{
  pinMode(13, OUTPUT);
  digitalWrite(13, LOW);
  Serial.begin(9600);
}

bool Serial_Readln_If_Available()
{  //Обработка внешней команды (поступившей через последовательное соединение Rx/Tx)
  bool res=false;
  sComm = "";
  sAct = "";
  sInfo = "";
  sParam = "";
  sParamName = "";
  sParamVal = "";
  while (Serial.available() > 0)
  {
    sComm += (char)Serial.read();
    delay(2);
  }
  sComm.trim();
  if (sComm.length() < 4)
  {
      sComm = "";
  }
  else
  {
     sAct = sComm.substring(0, 4);
     sAct.trim();
     if ((sAct != "GET:") && (sAct != "SET:"))
```

```
   {
      sComm = "";
      sAct = "";
   }
}
if (sComm != "")
{
   InCommandProcessing = true;
   sComm.setCharAt(0, ' ');
   sComm.setCharAt(1, ' ');
   sComm.setCharAt(2, ' ');
   sComm.setCharAt(3, ' ');
   sComm.trim();
   if (sComm == "") { sComm = "device_all; info_all"; }
}
if (sComm != "")
{
   sComm.trim();
   nParam = 0;
   while (sComm != "")
   {
      sParam = comm_parse();
      sComm.trim();
      if (sParam != "")
      {
         nParam += 1;
        if (nParam == 1)
        {
           bool itsok = false;
           if (sParam == "") { itsok = true; }
           if (sParam == "device_all") { itsok = true; }
           if (sParam == id_this) { itsok = true; }
           if (itsok == false)
           {
              sComm = "";
           }
        }
        else
        {
           if (sParam == "info_all")
           {
              sParam = "info_all=info_all";
           }
           sParamName = param_parse();
```

```
            if (sParamName != "")
            {
                sParamVal = sParam;
                sParamVal.trim();
                if (sAct == "GET:")
                {
                    res = action_get();
                }
                if (sAct == "SET:")
                {
                    res = action_set();
                }
            }
          }
        }
      }
      if (sAct == "GET:")
      {
         sInfo.trim();
         if (sInfo != "")
         {
            String sEndData = "]>";
            String sStartData = "<[";
            Serial.println("A: " + sStartData + id_this+"; " + sInfo + sEndData);
            //Serial.flush();
            res = true;
         }
      }
  }
  sComm = "";
  if (res = false) { sAct = ""; }
  sInfo = "";
  sParam = "";
  sParamName = "";
  sParamVal = "";
  InCommandProcessing = false;
  return res;
}

int StringToPin(String s)
{ //Конвертация строкового значения в номер Пина
  int res;
  res = -1;
  s.trim();
```

```
  s.toUpperCase();
  //-------------------------------------
  //Цифровые пины
  if (s == "PIN_0") { res = 0; };
  if (s == "PIN_1") { res = 1; };
  if (s == "PIN_2") { res = 2; };
  if (s == "PIN_3") { res = 3; };
  if (s == "PIN_4") { res = 4; };
  if (s == "PIN_5") { res = 5; };
  if (s == "PIN_6") { res = 6; };
  if (s == "PIN_7") { res = 7; };
  if (s == "PIN_8") { res = 8; };
  if (s == "PIN_9") { res = 9; };
  if (s == "PIN_10") { res = 10; };
  if (s == "PIN_11") { res = 11; };
  if (s == "PIN_12") { res = 12; };
  if (s == "PIN_13") { res = 13; };
  if (s == "PIN_14") { res = 14; };
  if (s == "PIN_15") { res = 15; };
  if (s == "LED_BUILTIN") { res = 13; };
  //-------------------------------------
  //-------------------------------------
  //ШИМ (PWM)
  if (s == "PIN_W3") { res = 3; };
  if (s == "PIN_W5") { res = 5; };
  if (s == "PIN_W6") { res = 6; };
  if (s == "PIN_W9") { res = 9; };
  if (s == "PIN_W10") { res = 10; };
  if (s == "PIN_W11") { res = 11; };
  //-------------------------------------
  //-------------------------------------
  //Аналоговые пины
  if (s == "PIN_A0") { res = A0; };
  if (s == "PIN_A1") { res = A1; };
  if (s == "PIN_A2") { res = A2; };
  if (s == "PIN_A3") { res = A3; };
  if (s == "PIN_A4") { res = A4; };
  if (s == "PIN_A5") { res = A5; };
  //if (s == "PIN_A6") { res = A6; };
  //-------------------------------------
  return res;
}

String comm_parse()
```

```
{  //Парсинг поступившей команды
  String res = "";
  sComm.trim();
  if (sComm != "")
  {
     int n = sComm.indexOf(';');
     if (n > 0)
     {
        res = sComm.substring(0, n);
        res.trim();
        String s = sComm.substring(n+1, sComm.length());
        sComm = s;
        sComm.trim();
     }
     else
     {
        res = sComm;
        sComm = "";
     }
  }
  return res;
}

String param_parse()
{  //Парсинг параметра
  String res = "";
  sParam.trim();
  if (sParam != "")
  {
     int n = sParam.indexOf('=');
     if (n > 0)
     {
        res = sParam.substring(0, n);
        res.trim();
        String s = sParam.substring(n+1, sParam.length());
        sParam = s;
        sParam.trim();
     }
     else
     {
        res = sParam;
        sParam = "";
     }
  }
```

```
  return res;
}

int pin_val_get(int pn, bool its_digit)
{ //Получить значение заданного Пина
  int res=-1;
  if (pn >= 0)
  {
     if (its_digit == true)
     {
        res = digitalRead(pn);
     }
     else
     {
        res = analogRead(pn);
     }
     String s = "";
     s = "pin_" + String(pn) + "=" + String(res) + "; ";
     sInfo = sInfo + s;
  }
  return res;
}

bool action_get()
{ //Отрабогка команды GET (serial_readln)
  bool res = false;
  sParamName.trim();
  if (sParamName != "")
  {
     if (sParamVal == "info_all")
     {
        pin_val_get(0, true);
        pin_val_get(1, true);
        pin_val_get(2, true);
        pin_val_get(3, true);
        pin_val_get(4, true);
        pin_val_get(5, true);
        pin_val_get(6, true);
        pin_val_get(7, true);
        pin_val_get(8, true);
        pin_val_get(9, true);
        pin_val_get(10, true);
        pin_val_get(11, true);
        pin_val_get(12, true);
```

```
   pin_val_get(13, true);
   pin_val_get(14, true);
   pin_val_get(15, true);
   pin_val_get(A0, false);
   pin_val_get(A1, false);
   pin_val_get(A2, false);
   pin_val_get(A3, false);
   pin_val_get(A4, false);
   pin_val_get(A5, false);
   //pin_val_get(A6, false);
   SysVar_Value_Calc();
   sInfo = sInfo + "sys_var=" + sSysVar + "; ";
   res = true;
}
else
{
   String s;
   s = sParamName;
   s.toUpperCase();
   if (s == "SYS_VAR")
   {
       SysVar_Value_Calc();
       sInfo = sInfo + "sys_var=" + sSysVar + "; ";
       res = true;
   }
   else
   {
      if (s == "ID")
      {
          sInfo = sInfo + "ID=" + id_this + "; ";
          res = true;
      }
      else
      {
         pin_num = StringToPin(sParamName);
         if (pin_num>=0)
         {
            va = 0;
            if (sParamName.indexOf('A') < 0)
            {
               if (sParamName.indexOf('W') < 0) // Проверка: интерпретировать, как ШИМ (PWM)?
               {
                  va = digitalRead(pin_num);
               }
```

```
                    else
                    {
                       va = analogRead(pin_num);
                    }
                 }
                 else
                 {
                    va = analogRead(pin_num);
                 }
                 sInfo +=  sParamName + "=" + va + "; ";
                 res = true;
              }
           }
        }
     }
  }
  return res;
}

bool action_set()
{ //Отработка команды  GET (serial_println)
  bool res = false;
  sParamName.trim();
  if (sParamName != "")
  {
     sParamVal.trim();
     if (sParamVal != "")
     {
        if (sParamName == "sys_var")
        {
           sSysVar = sParamVal;
           res = true;
        }
        else
        {
           pin_num = StringToPin(sParamName);
           Serial.println(sParamName + "  = "+pin_num);
           if (pin_num>=0)
           {
              bool yes_next = true;
              sParamVal.toUpperCase();
              if ((sParamVal == "IN") || (sParamVal == "INPUT"))
              {
                  pinMode(pin_num, INPUT);
```

```
                res = true;
                yes_next = false;
            }
            if ((sParamVal == "OUT") || (sParamVal == "OUTPUT"))
            {
                pinMode(pin_num, OUTPUT);
                res = true;
                yes_next = false;
            }
            if (yes_next == true)
            {
               va = sParamVal.toInt();
               if (sParamName.indexOf('A') < 0)
               { //Цифровой
                  if (sParamName.indexOf('W') < 0) // Проверка: интерпретировать, как ШИМ (PWM)?
                  {  //Нет
                     if (va < 0) {va = 0; }
                     if (va > 1) {va = 1; }
                     digitalWrite(pin_num, va);
                  }
                  else
                  {  //Да
                     if (va < 0) {va = 0; }
                     if (va > 255) {va = 255; }
                     analogWrite(pin_num, va);
                  }
               }
               else
               {  //Аналоговый
                  analogWrite(pin_num, va);
               }
               res = true;
            }
         }
      }
   }
  }
  return res;
}
```

Программа `My_TS_WS.exe`

Программа предельно проста и не требует серьезного описАния…

Вкладка «Информация/Управление»

На рисунке 4 приведен внешний вид программы `My_TS_WS.exe`, открытой на вкладке «Информация/Управление».

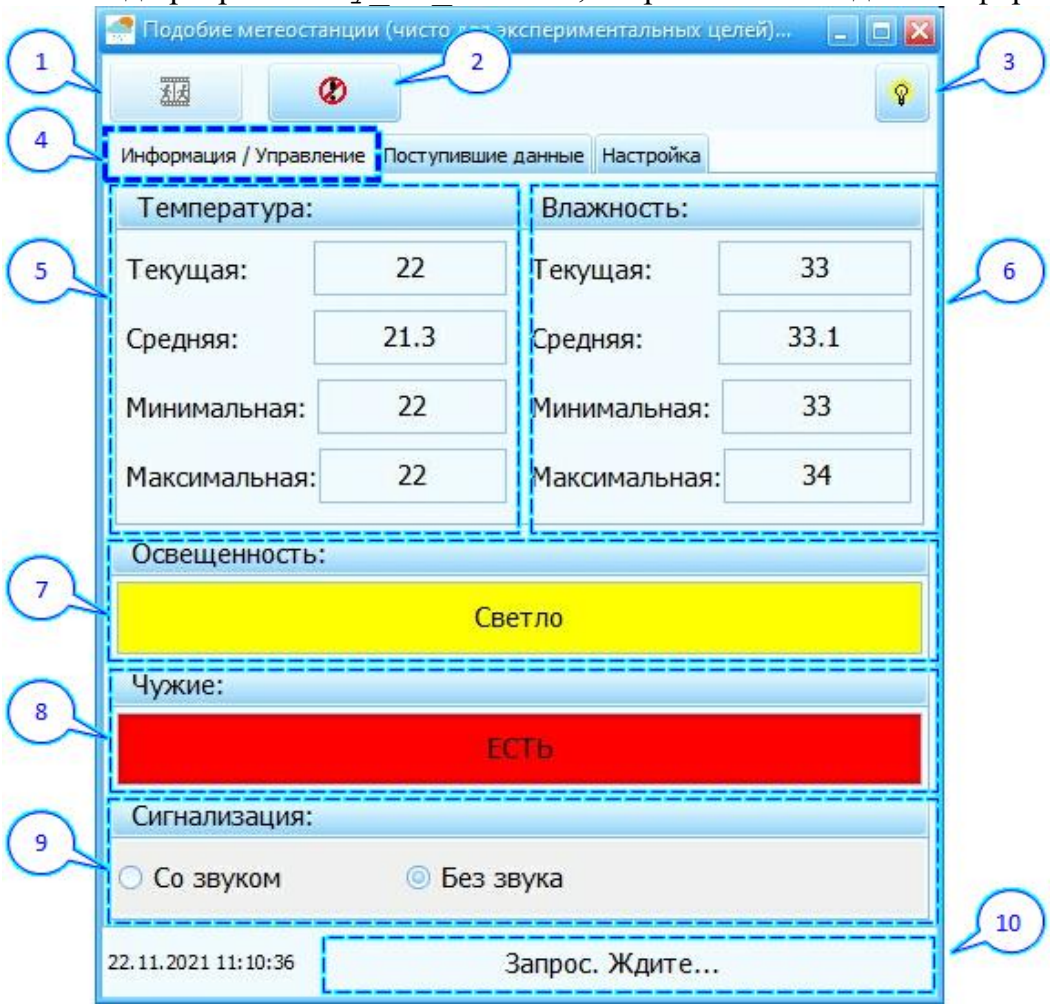


Рисунок 4 – Внешний вид программы `My_TS_WS.exe`,
открытой на вкладке «Информация/Управление» (см., также, таблицу 4)

Таблица 4 – Список выносок на рисунке 4

Выноски	Обозначение
1	2
1	Кнопка «Старт для информационного обмена в цикле»
2	Кнопка «Стоп для информационного обмена в цикле»
3	Индикация фаз информационного обмена в цикле (запрос/ожидание)
4	Вкладка «Информация/Управление»
5	Данные о температуре
6	Данные о влажности
7	Данные об освещенности (светло/темно)
8	Данные от сигнализации (наличие «чужих»: есть/нет)
9	Панель для отображения текущего режима сигнализации (данные поступают от «железа») и интерактивного управления им
10	Информационная панель

Вкладка «Поступившие данные»

На рисунке 5 приведен внешний вид программы `My_TS_WS.exe`, открытой на вкладке «Поступившие данные».

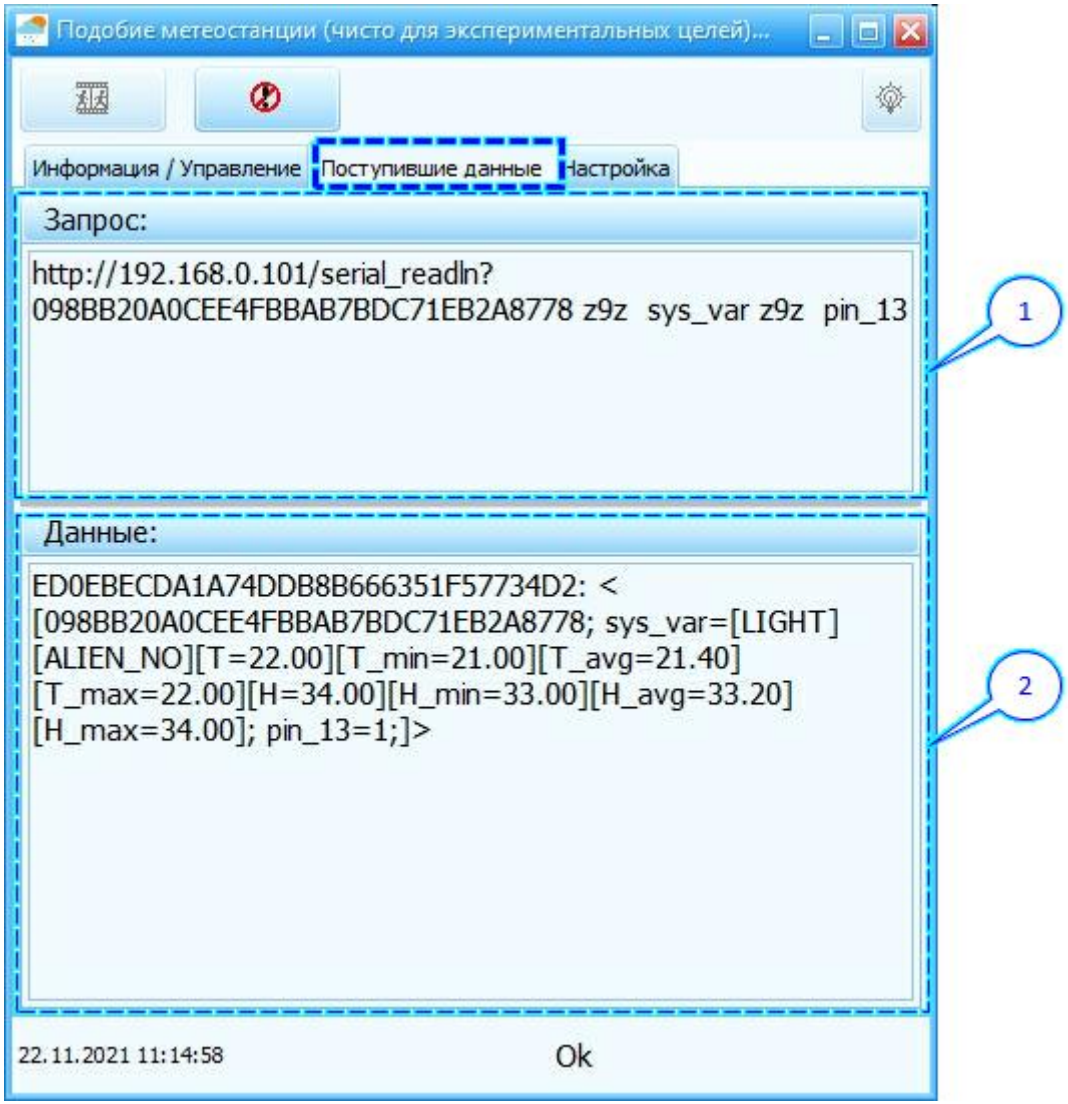


Рисунок 5 – Внешний вид программы `My_TS_WS.exe`, открытой на вкладке «Поступившие данные»

Список выносок на рисунке 5:
выноской-1 обозначена область, где размещается текст http-запроса, отправляемого программой на ESP-12F WeMos D1 WiFi;
выноской-2 обозначена область, где размещается текст ответа ESP-12F WeMos D1 WiFi (в ответ на запрос программы).

Примечание − тексты команд (http-запросов) и их формат, а также, формат ответа, подробно рассмотрены в разделе «Arduino. ТехноСфера. Шаг-2. Скетч для ESP-12F WeMos D1 WiFi» и далее..

Вкладка «Настройка

На рисунке 6 приведен внешний вид программы `My_TS_WS.exe`, открытой на вкладке «Настройка».

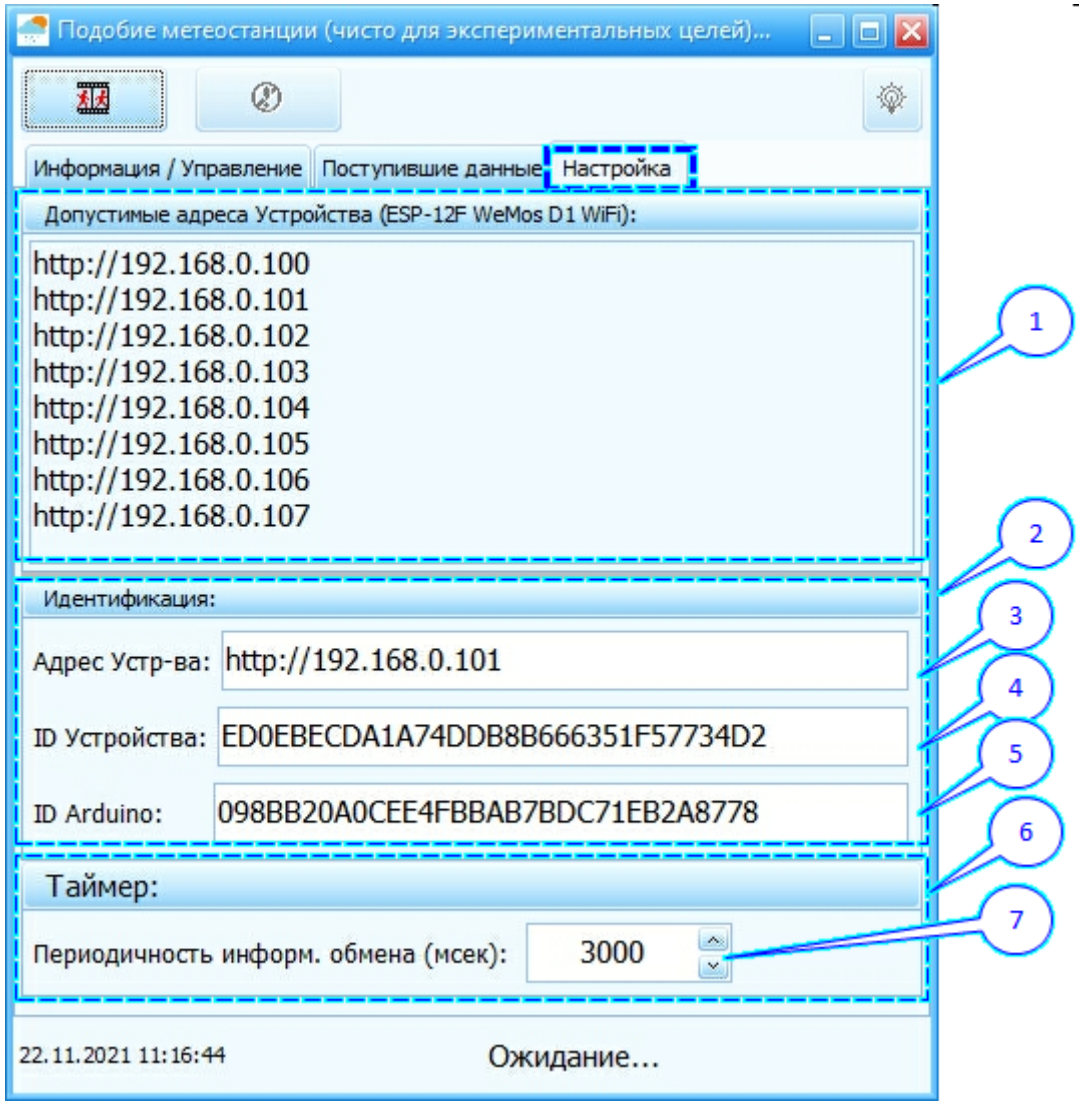


Рисунок 6 – Внешний вид программы `My_TS_WS.exe`, открытой на вкладке «Настройка»

Список выносок на рисунке 6:

выноской-1 обозначен список допустимых http-адресов, к которым может быть «привязан» ESP-12F WeMos D1 WiFi;

выноской-2 обозначена группа элементов управления, где отображается идентификационная информация;

выноской-3 обозначен элемент управления, где отображается актуальный (в текущую сессию информационного обмена) http-адрес ESP-12F WeMos D1 WiFi (это значение вычисляется автоматически в процессе информационного обмена);

выноской-4 обозначен элемент управления, где отображается уникальный ID для ESP-12F WeMos D1 WiFi (это определено в соответствующем скетче, см выше − в разделе «Скетч (для ESP-12F WeMos D1 WiFi)»);

выноской-5 обозначен элемент управления, где отображается уникальный ID для сопряженного Arduino UNO (это определено в соответствующем скетче, см выше − в разделе «Скетч (для Arduino UNO)»);

выноской-6 обозначена группа элементов управления, где определен регламент информационного обмена (в данном случае, только один параметр);

выноской-7 обозначен элемент управления, где отображается периодичность информационного обмена в миллисекундах.

Основные операции

Важно!

1. Все команды отрабатываются с некоторой задержкой, которая определяется (в основном) соответствующими задержками в скетче для Arduino UNO (см. выше).
2. Отображаемая в программе информация также «запаздывает» (относительно «железа»), в зависимости от периодичности информационного обмена.
3. При каждой операции, связанной с информационным обменом, программа проверяет актуальность http-адреса для заданного ESP-12F WeMos D1 WiFi и, если заданный http-адрес «не отвечает» или на нем Устройство с иным ID, то программа производит поиск Устройства по списку допустимых http-адресов. Это занимает определенное время (до нескольких минут). Т.е., надо набраться терпения…

1. Сформировать список допустимых http-адресов, к которым <u>может быть</u> «привязан» ESP-12F WeMos D1 WiFi

Выполнить следующие действия:

1. Открыть вкладку «Настройка»;
2. Интерактивно (с клавиатуры) внести соответствующие http-адреса в текстовое поле ввода, обозначенное выноской 1 на рисунке 6.

2. Указать начальный http-адрес для ESP-12F WeMos D1 WiFi

1. Открыть вкладку «Настройка»;
2. Интерактивно (с клавиатуры) внести соответствующий http-адрес в текстовое поле ввода, обозначенное выноской 3 на рисунке 6.

Примечание – в процессе информационного обмена этот адрес будет заменен на актуальный.

3. Указать уникальный ID для ESP-12F WeMos D1 WiFi

1. Открыть вкладку «Настройка»;
2. Интерактивно (с клавиатуры) внести соответствующий ID в текстовое поле ввода, обозначенное выноской 4 на рисунке 6.

Примечание – это значение должно соответствовать значению константы `id_this` в разделе «Скетч (для ESP-12F WeMos D1 WiFi)».

4. Указать уникальный ID для сопряженного Arduino UNO

1. Открыть вкладку «Настройка»;
2. Интерактивно (с клавиатуры) внести соответствующий ID в текстовое поле ввода, обозначенное выноской 5 на рисунке 6.

Примечание – это значение должно соответствовать значению константы `id_this` в разделе «Скетч (для Arduino UNO)».

5. Задать периодичность информационного обмена

1. Открыть вкладку «Настройка»;
2. Интерактивно (с клавиатуры) внести соответствующее значение в поле ввода, обозначенное выноской 7 на рисунке 6.

Примечание – минимальное значение = 500 мсек.

6. Запустить процесс информационного обмена в цикле

1. Открыть вкладку «Информация/Управление»;
2. Нажать на кнопку  (см. выноска-1 на рисунке 4).

7. Прервать (остановить) процесс информационного обмена

1. Открыть вкладку «Информация/Управление»;
2. Нажать на кнопку  (см. выноска-2 на рисунке 4).

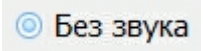
8. Установить режим сигнализации: «Звук» + «Свет»

1. Открыть вкладку «Информация/Управление»;
2. Нажать на радио-кнопку 

(см. выноска-9 на рисунке 4);
3. Дождаться отработки соответствующего http-запроса.

Примечание – для управления режимом сигнализации используется соответствующее состояние пина 13 на Arduino UNO.

9. Установить режим сигнализации: «только Свет»

1. Открыть вкладку «Информация/Управление»;
2. Нажать на радио-кнопку 

(см. выноска-9 на рисунке 4);
3. Дождаться отработки соответствующего http-запроса.

Примечание – для управления режимом сигнализации используется соответствующее состояние пина 13 на Arduino UNO.

Выводы

Примененные: инструментарий, способы, алгоритмы и методика информационного обмена, в целом оказались вполне удовлетворительными для дальнейшего их использования (при соответствующей доработке), в рамках игрового мини-проекта

https://roamer55.ru/main_programming/arduino/arduino_technosphere_000/