ТехноСфера

Программа ESP_12F_WeMos_D1_WiFi_Test.exe

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

Пример

Включать и отключать «Сирену» на сопряженном (с Устройством) Arduino UNO (см., также, рисунки 1 и 2).

Под термином «Сирена» понимается (в данном случае) звук полицейской сирены и «мигание» светодиода.

Идея (и реализация в виде фрагментов соответствующего исходного кода с использованием библиотеки **`Thread.h`**) использования поточности взята отсюда:
`https://soltau.ru/index.php/arduino/item/373-kak-vypolnyat-parallelnye-zadachi-threads-v-programme-dlya-arduino`

Идея (и реализация в виде фрагментов соответствующего исходного кода) полицейской сирены взята отсюда:
`http://arduino-kid.ru/lesson/podklyuchenie-pischalki-buzzer-k-plate-arduino-urok-no-3#google_vignette`

Библиотека `Thread.h` взята отсюда: `https://github.com/ivanseidel/ArduinoThread/archive/master.zip`

«Базовый» скетч для сопряженного Arduino UNO доработан с учетом этого исходного кода (см. ниже).
То, что добавлено – отмечено синим цветом.

Управление «Сиреной» (ее включение и вЫключение) производится с использованием пина 13.
Если на пине 13 значение HIGH, то «Сирена» работает. В противном случае – нет.

А управлять состоянием пина 13 будем «удаленно».
С помощью скрипта, см. ниже, из программы `ESP_12F_WeMos_D1_WiFi_Test.exe`.

На рисунке 1 приведена соответствующая схема подключений.

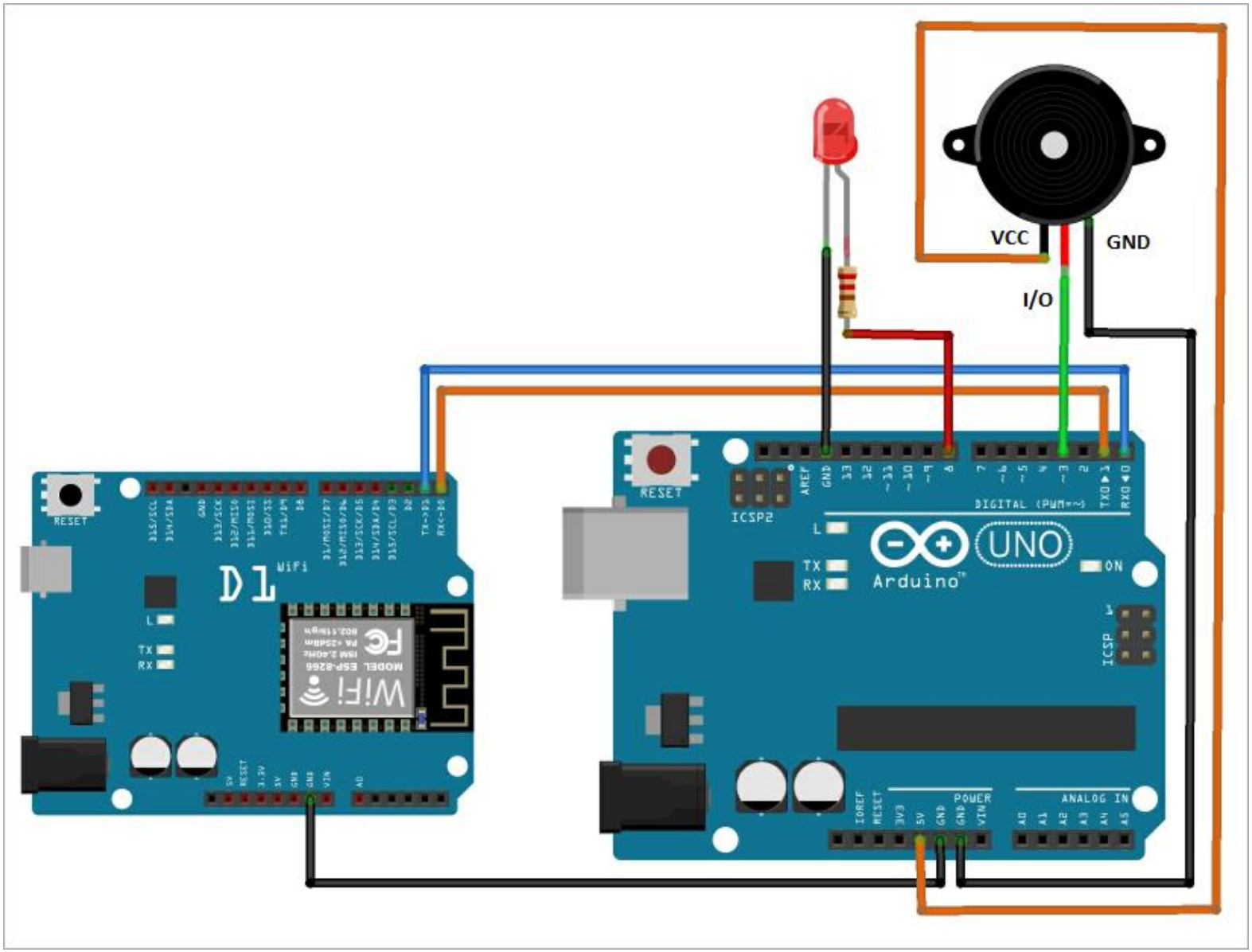

Рисунок 1 – Схема соединений для тестирования скрипта в Примере

Скетч:

```
/*
   Дата актуальности: 14.11.2021
   Назначение:
     1. Обеспечение информационного обмена между "ESP-12F WeMos D1 WiFi" и этим (сопряженным) Arduino UNO через "Tx/Rx".
     2. 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
     3. "Внешнее" управление "Сиреной" + светодиодом (с использованием пина 13)
*/
#include <Thread.h>  // подключение библиотеки ArduinoThread
                     // (https://github.com/ivanseidel/ArduinoThread/archive/master.zip)
//.................................
const String id_this = "098BB20A0CEE4FBBAB7BDC71EB2A8778"; // Уникальный ID этого устройства
//.................................
//.................................
//Постоянно действующие установки
String sSysVar = "";   // системная переменная (резерв)
//.................................
//.................................
//Вспомогательные переменные (используются в оперативных целях при обработке поступившей команды)
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
const int soundPin = 3;   //баззер
const int ledPin   = 8;   //светодиод
const int dt_sound = 500;       //интервал "срабатывания" сирены (баззера) и светодиода, мсек
Thread soundThread = Thread();  //создается поток управления сиреной (баззером) и светодиодом
//.................................

//*********************************
//*********************************
```

```
//*********************************
void setup()
{
  setup0();
  //----------
  //Специфика...
  pinMode(soundPin, OUTPUT);  //режим пина 3 (баззер)
  pinMode(ledPin, OUTPUT);    //режим пина 8 (светодиод)
  soundThread.onRun(sound);           // назначение потоку задачи: sound() - баззер
  soundThread.setInterval(dt_sound); // интервал срабатывания (для потока), мсек
  //digitalWrite(13, HIGH);  //отладка
  //----------
} //setup()

void loop()
{
    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
    int p13 = digitalRead(13); //Читать состояние пина 13
    if (p13 == HIGH)
    {
       // "Сирена"
       if (soundThread.shouldRun())
           soundThread.run(); // запускается поток
    }
    else
    {
       noTone(soundPin);  // вЫключение звука
       digitalWrite(13, LOW); // вЫключение светодиода
    }
    //----------------------------------------
} //loop()
```

```
//**********************************
//**********************************
//**********************************

//----------------------------------------
//Специфика
// Полицейская сирена + 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); // сделаем паузу
```

```
}
//----------------------------------------

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 != "")
  {
     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 = "";
  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()
{ //Отработка 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);
   sInfo = sInfo + "sys_var=" + sSysVar + "; ";
   res = true;
}
else
{
   String s;
   s = sParamName;
   s.toUpperCase();
   if (s == "SYS_VAR")
   {
       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;
}
```

Скрипт:

```
#language PascalScript

(*
  Пример скрипта, который включает и вЫключает "сирену" + светодиод
  на сопряженном (с Устройством) Arduino UNO (с использованием пина 13).
  Баззер - на 3-м пине. Светодиод - на 8-м пине.
  См., также, здесь:
  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
*)
Const
  Устройство_Id    = 'ED0EBECDA1A74DDB8B666351F57734D2';
  Устройство_Адрес_ПоУмолчанию = 'http://192.168.0.100';
  Arduino_Id       = '098BB20A0CEE4FBBAB7BDC71EB2A8778';

Var
  Устройство_Ответ : TStrings;
  Устройство_СписокДопустимыхАдресов:TStrings;
  Устройство_Адрес: string;
  Команда:string;
  ЧтоСделать:integer;
  ВыйтиИзЦикла:boolean;

procedure Выполнить_Команду;
begin
  Команда:=trim(Команда);
  if Команда<>'' then begin
     Устройство_Адрес:=trim(Устройство_Адрес);
     if Устройство_Адрес='' then Устройство_Адрес:=Устройство_Адрес_ПоУмолчанию;
     Устройство_Адрес:=Устройство_Команду_Выполнить(Устройство_Id,
                                Устройство_Адрес,
                                Устройство_СписокДопустимыхАдресов,
                                Команда,
                                Устройство_Ответ);
  end;
end;

BEGIN
  Сообщ_Очистить;  //очистить окно сообщений
  Устройство_СписокДопустимыхАдресов := TStringList.Create;
  Устройство_Ответ := TStringList.Create;
```

```
  TRY
    //---------------------------
    //Сформировать список допустимых http-адресов (IP) для данного Устройства
    Устройство_СписокДопустимыхАдресов.Add('http://192.168.0.100');
    Устройство_СписокДопустимыхАдресов.Add('http://192.168.0.101');
    Устройство_СписокДопустимыхАдресов.Add('http://192.168.0.102');
    Устройство_СписокДопустимыхАдресов.Add('http://192.168.0.103');
    Устройство_СписокДопустимыхАдресов.Add('http://192.168.0.104');
    Устройство_СписокДопустимыхАдресов.Add('http://192.168.0.105');
    Устройство_СписокДопустимыхАдресов.Add('http://192.168.0.106');
    //---------------------------
    Устройство_Адрес := Устройство_Адрес_ПоУмолчанию;
    Сообщ_Добавить('Начало. Адрес: '+Устройство_Адрес+';  '+DateTimeToStr(NOW));
    ВыйтиИзЦикла := false;
    while not ВыйтиИзЦикла do
     begin
       ЧтоСделать := Вопрос('Yes-включить, No-вЫключить, Cancel-прервать', mtConfirmation, mbYesNoCancel);
       if ЧтоСделать=mrYes then begin
          //Включить "сирену"
          Команда := '/serial_println?'+Arduino_Id+'; pin_13=1';
          Выполнить_Команду;
          Сообщ_Добавить('+++ Включили: '+DateTimeToStr(NOW));
       end;
       if ЧтоСделать=mrNo then begin
          //ВЫключить "сирену"
          Команда := '/serial_println?'+Arduino_Id+'; pin_13=0';
          Выполнить_Команду;
          Сообщ_Добавить('--- ВЫключили: '+DateTimeToStr(NOW));
       end;
       if ЧтоСделать=mrCancel then ВыйтиИзЦикла := true;
       Application_ProcessMessages;
    end;
    Сообщ_Добавить('Конец.  '+DateTimeToStr(NOW));
    ShowMessage('Конец');
  FINALLY
    FreeAndNil(Устройство_Ответ);
    FreeAndNil(Устройство_СписокДопустимыхАдресов);
  END;
END.
```

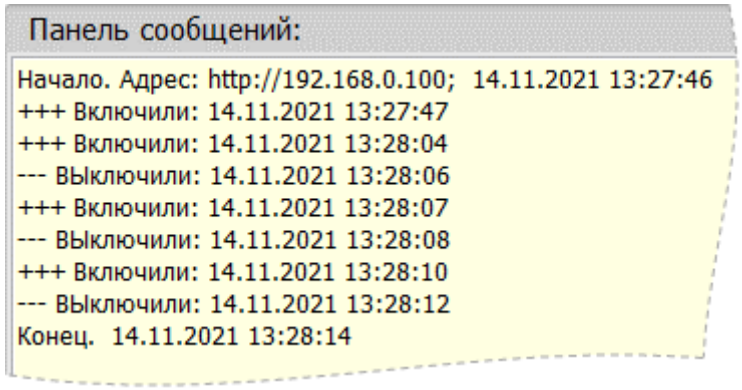

Рисунок 2 − Тексты сообщений в панели сообщений