Блок управления вентиляторами компьютера

Обучающийся Богачев-Воевудский А.А

Преподаватель Максимов А.В.

Цели и задачи

- Цель – собрать устройство, позволяющее управлять оборотами вентилятора компьютера через WIFI
- Задачи:

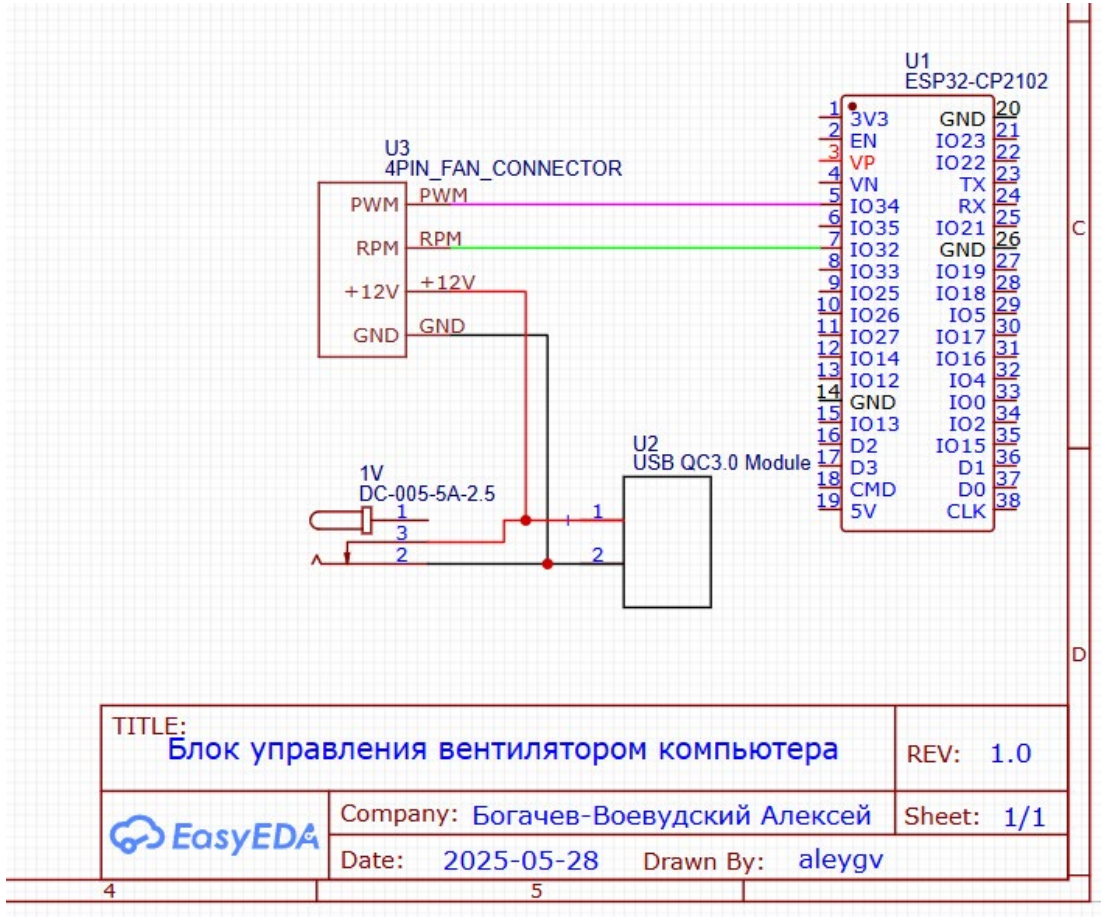
1. Купить и собрать устройство из комплектующих
2. Написать код

Актуальность и аналоги

- Данная тема актуальна для людей, у которых нет встроенного управления вентиляторами в компьютере
- Аналоги:

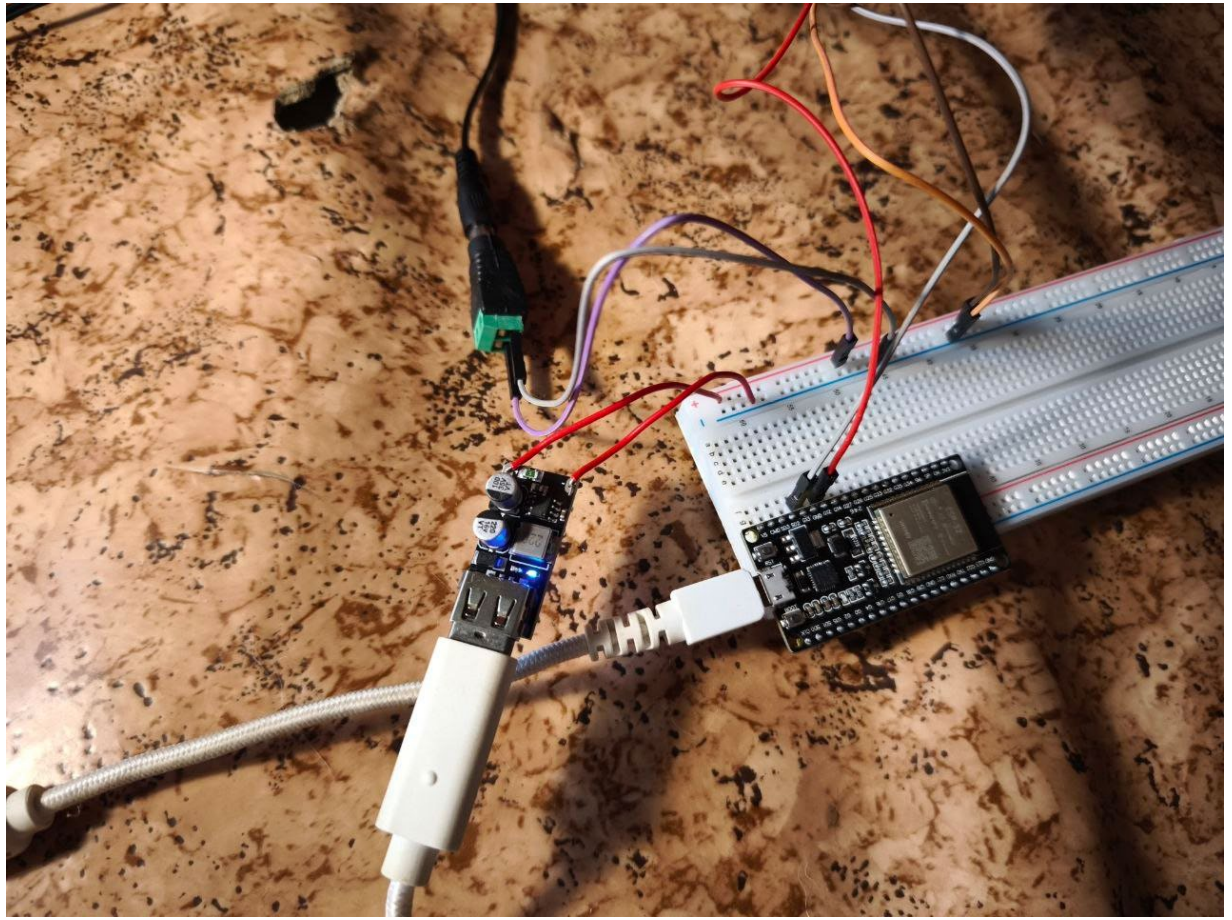
1. "Testing Mosquitto MQTT Broker With ESP32" — В этой статье описывается, как использовать MQTT-брокер Mosquitto с ESP32 для удаленного управления устройствами, такими как лампы, вентиляторы и датчики.
2. "ESP32 Fan Control System for PC, Server, and Rack Temperature" — Эта работа посвящена системе контроля температуры на основе ESP32 с поддержкой MQTT по Wi-Fi.

Схема устройства и себестоимость



- Разъем питания DC 2.5 x 5.5
- ESP32 NODEMCU 38pin DEVKIT V1 WIFI + Bluetooth CP2102
- Беспаечная макетная плата
- DC-DC step-down понижающий преобразователь с поддержкой QC3.0
- Модуль реле – 1 канал
- Переходник USB OTG micro USB Defender, адаптер для передачи данных с телефона
- Блок питания (адаптер) T120100-2C1 12V 1A 5.5 x 2.1

Вид устройства



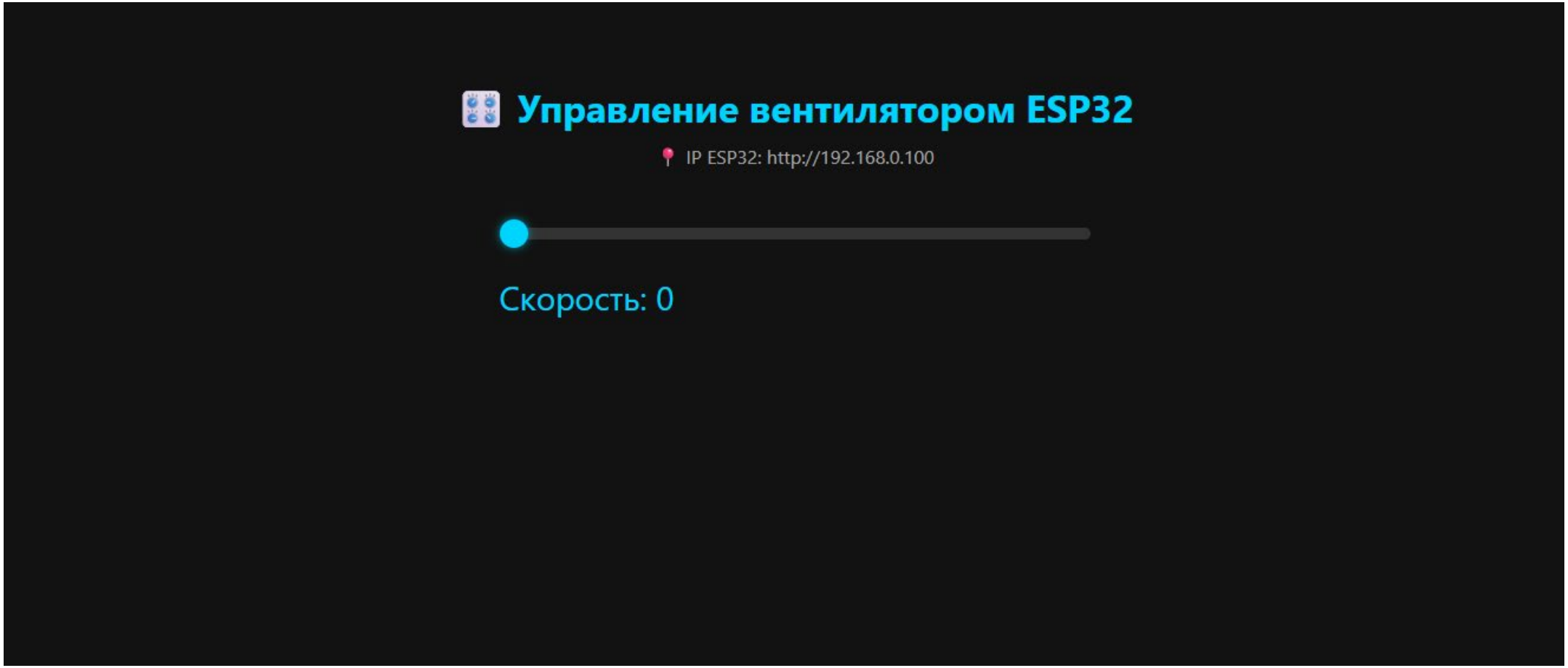
Код

```ino
#include <WiFi.h>
#include <ESPAsyncWebServer.h>
#include <HTTPClient.h>

// ==== Настройки Wi-Fi ====
const char* ssid = "Bogachevich";          // Укажи свой SSID
const char* password = "DNEAT78R";         // Укажи свой пароль

// ==== Пины ====
#define FAN_PWM_PIN 12     // PWM пин (управление вентилятором)
#define FAN_RPM_PIN 13     // RPM пин (считывание импульсов)

// ==== Переменные ====
volatile byte fanRPMCount = 0;
volatile unsigned long last_interrupt_time = 0;
unsigned long lastRPMSample = 0;
int fanSpeed = 128;  // Стартовая скорость (0–255)
const int pulsesPerRevolution = 2;

AsyncWebServer server(80);

// ==== Обработка прерывания для RPM ====
void IRAM_ATTR rpmISR() {
  unsigned long interrupt_time = micros();
  if (interrupt_time - last_interrupt_time > 1000) {
    fanRPMCount++;
    last_interrupt_time = interrupt_time;
  }
}
```

```ino
// ==== Настройка скорости вентилятора ====
void setFanSpeed(int speed) {
  ledcWrite(0, speed);
}

// ==== Отправка IP на Flask-бэкенд ====
void sendIPToBackend() {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;
    String url = "http://192.168.0.105:5000/receive_ip";  // ← Укажи IP
    http.begin(url);
    http.addHeader("Content-Type", "application/json");

    String jsonData = "{\"ip\":\"";
    jsonData += WiFi.localIP().toString();
    jsonData += "\"}";

    int httpResponseCode = http.POST(jsonData);

    if (httpResponseCode > 0) {
      Serial.printf("HTTP Response code: %d\n", httpResponseCode);
      String response = http.getString();
      Serial.println("Response: " + response);
    } else {
      Serial.print("Error sending POST: ");
      Serial.println(http.errorToString(httpResponseCode).c_str());
    }

    http.end();
  } else {
    Serial.println("WiFi not connected");
  }
```

```ino
void setup() {
  Serial.begin(115200);

  // ==== Настройка вентилятора ====
  pinMode(FAN_PWM_PIN, OUTPUT);
  ledcSetup(0, 25000, 8);           // канал 0, 25 кГц, 8 бит
  ledcAttachPin(FAN_PWM_PIN, 0);
  setFanSpeed(fanSpeed);

  // ==== Настройка RPM ====
  pinMode(FAN_RPM_PIN, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(FAN_RPM_PIN), rpmISR, FALLING);

  // ==== Подключение к Wi-Fi ====
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.print(".");
  }

  Serial.println("\nConnected to WiFi");
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());

  // ==== Отправка IP на Flask ====
  sendIPToBackend();
```

Вид страницы веб-сервера



Заключение

- Собран устройство, позволяющее управлять оборотами вентилятора компьютера через WIFI