Politechnika Wrocławska

Sprawozdanie 6

Ćwiczenie 6.Akwizycja danych

DHT11 z wykorzystaniem łączności WiFi

Krzysztof Zalewa,Wiktor Wojnar

27.1.2025

Spis treści

1 Wstęp teoretyczny **2**
- 1.1 Charakterystyka i zasady działania systemu FreeRTOS . . . 2
- 1.2 Wątki w FreeRTOS . . . 2
- 1.3 DHT11 . . . 2
  - 1.3.1 Budowa . . . 3
  - 1.3.2 Zasady działania . . . 3

2 Zadanie laboratoryjne **4**
- 2.1 Treść zadania . . . 4
- 2.2 Opis działania programu . . . 4
- 2.3 Schemat połączenia . . . 4
- 2.4 Kod programu . . . 6

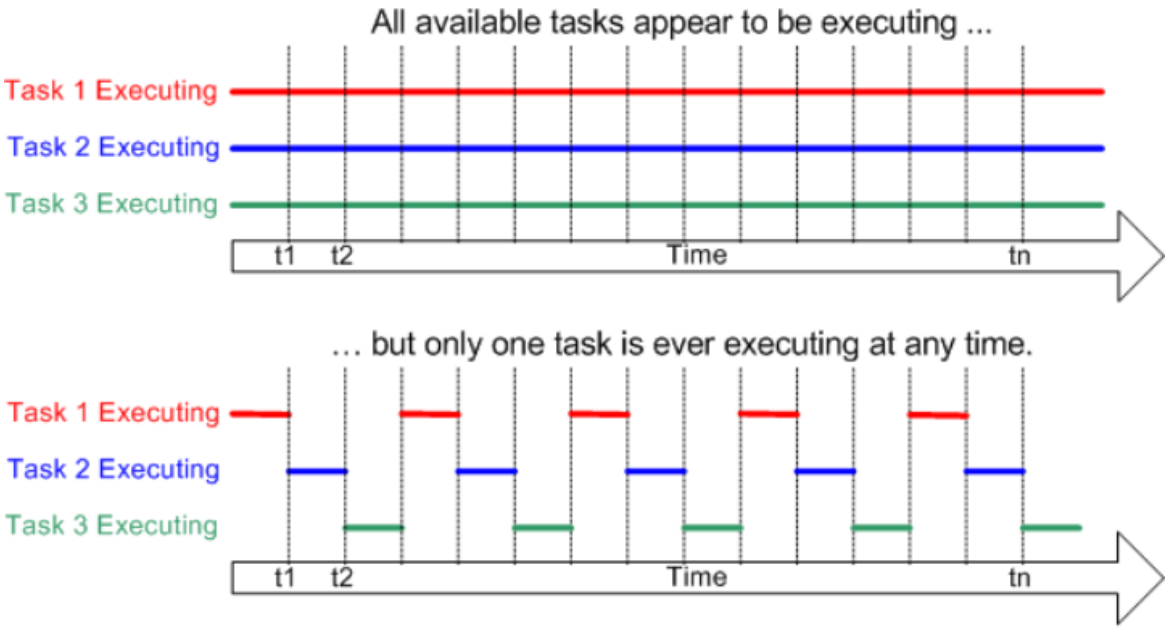
3 Źródła **8**

1 Wstęp teoretyczny

1.1 Charakterystyka i zasady działania systemu FreeRTOS

FreeRTOS jest systemem operacyjnym czasu rzeczywistego (ang. Real time operating system) dla systemów wbudowanych. FreeRTOS został zaprojektowany tak by kod źródłowy był prosty i krótki. Takie podejście pozwala na użycie go nawet na najmniejszych urządzeniach.

1.2 Wątki w FreeRTOS



Rysunek 1: Wykonywanie wielu zadań[2]

W systemach takich jak Linux programy wykonywalne implementowane są przez jeden lub więcej wątków. W systemach RTOS wątki zwykle nazywane są zadaniami. Jedno rdzeniowe processory mogą wykonywać tylko jedną operacje w danym momencie. Jednakże poprzez szybkie przełączanie między wykonywanym zadaniem można zbliżyć się do wykonywania wielu zadań jednocześnie. Za wybór które zadanie powinno być wykonywane odpowiada planista (ang. scheduler).

1.3 DHT11

Rysunek 2: Płytka DHT11[3]

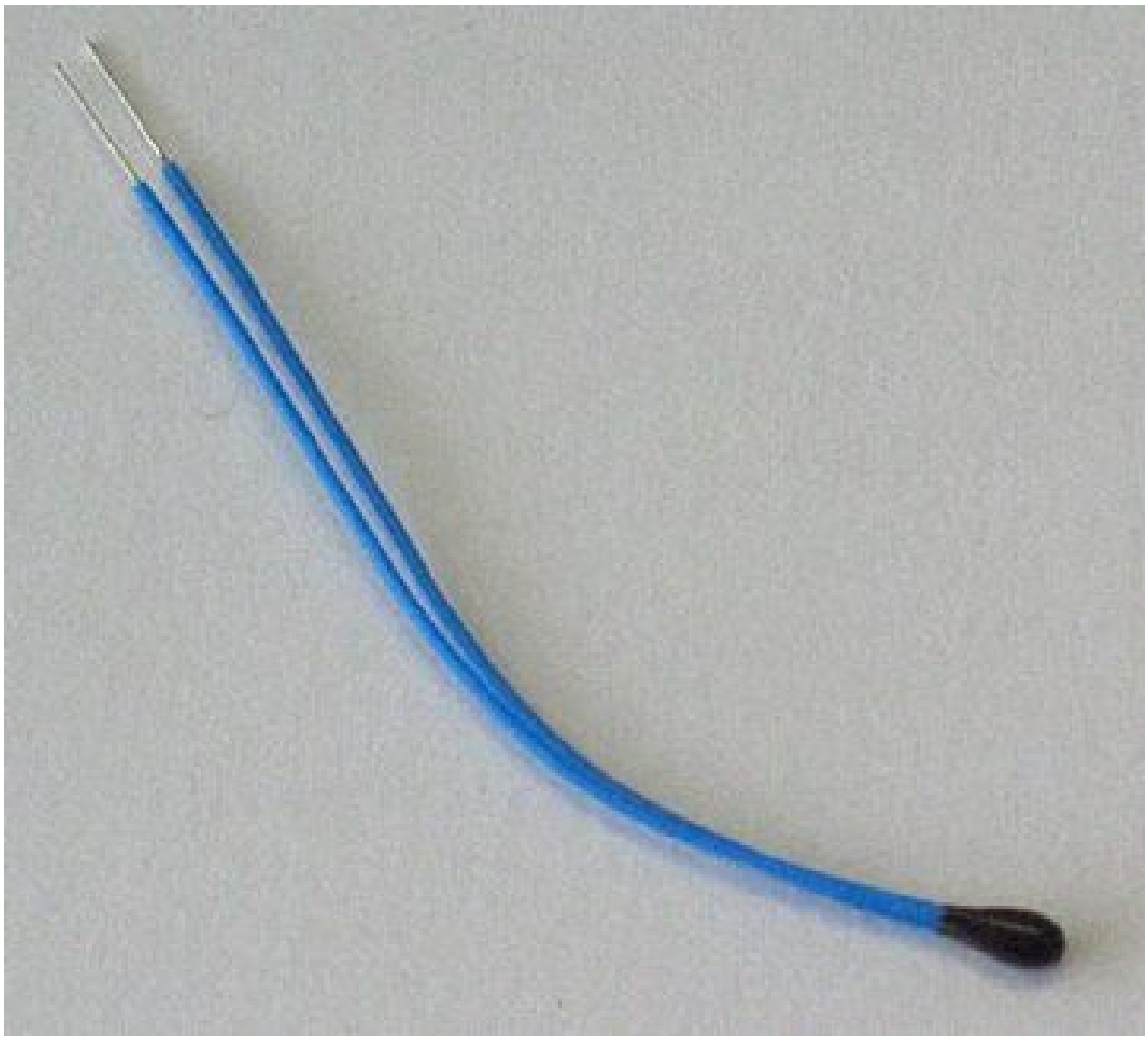
1.3.1 Budowa

DHT11 zawiera w sobie cyfrowy sensor temperatury i wilgoci. Według źródła 4. układ ten do pomiaru temperatury wykorzystuje układ NTC a do pomiaru wilgotności układ oporowy. Wykonywane pomiary są w zakresie:

1. Temperatura 0-50°C błąd ±2°C
2. Wilgotność 20-90% RH ±5% RH

1.3.2 Zasady działania

Pomiar wilgotności polega na pomiarze zmiany rezystancji w materiale pomiarowym. Jako że rezystancja zależy też od temperatury materiale to do urządzenia musi być dołączony układ pomiaru temperatury.
Pomiar temperatury tak samo jak w przypadku wilgotności polega na pomiarze rezystancji. Do wykonania takiego pomiaru używa się termistora (rezystora o rezystancji silnie zależnej od temperatury). Rezystor NTC ma ujemny współczynnik temperturowy (wzrost temperatury powoduje zmniejszenie rezystancji)



Rysunek 3: Termistor NTC

2 Zadanie laboratoryjne

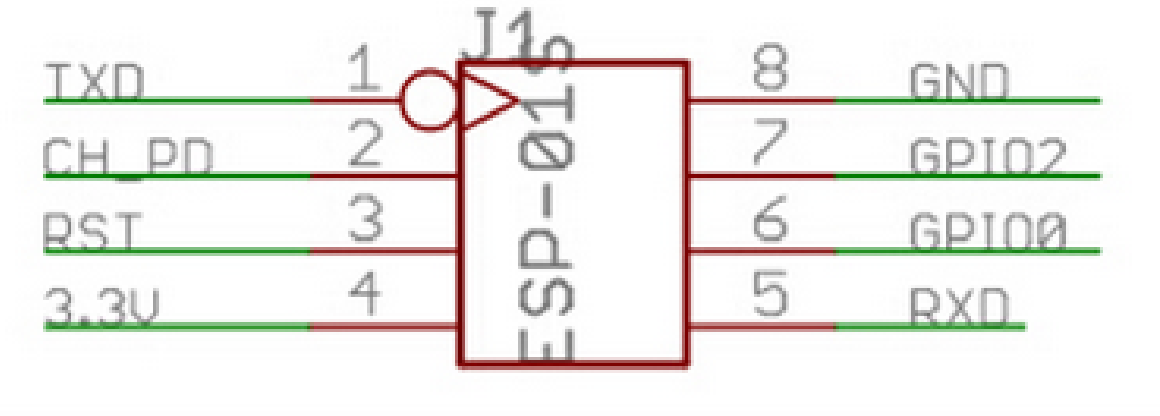
2.1 Treść zadania

W ramach zadania laboratoryjnego należało skonfigurować układ ESP32 i uruchomić przykładowy program. Następnie należało zainstalować system FreeRTOS oraz uruchomić wątki (1-akwizycji pomiarów, 2-przetwarzania danych, 3-transmisji wyników). Na koniec należało rozbudować wątki 1 i 2 w stopniu uzgodnionym z prowadzącym.

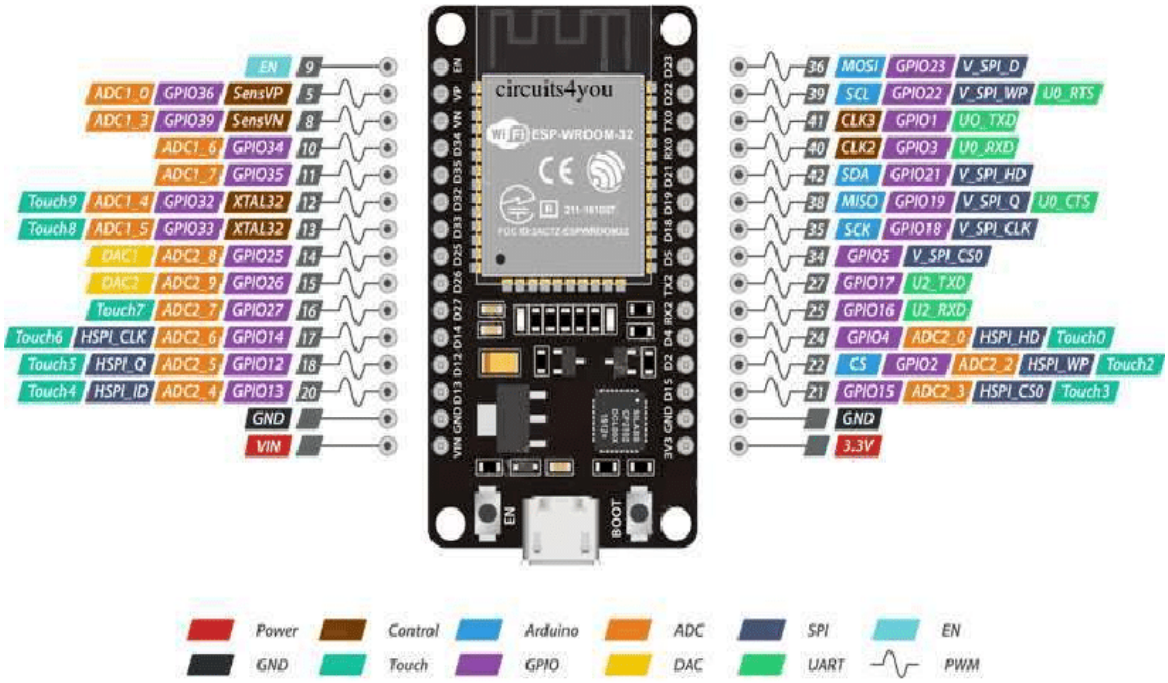
2.2 Opis działania programu

Zgodnie z zadaniem zaimplementowany program ma trzy główne wątki. Jeden do obsługi łączności WiFi. Drugi do pobierania danych podłączonego z DHT11. Trzeci do wysyłania danych do użądzeń końcowych. W wyniku działania programu otrzymano prostą stronę HTML która zawiera napis : Temperature today is: X °C, and the humidity in the air is: Y. Gdzie X to temperatura pobrana z DHT11 a Y to wilgotność pobrana z DHT11.

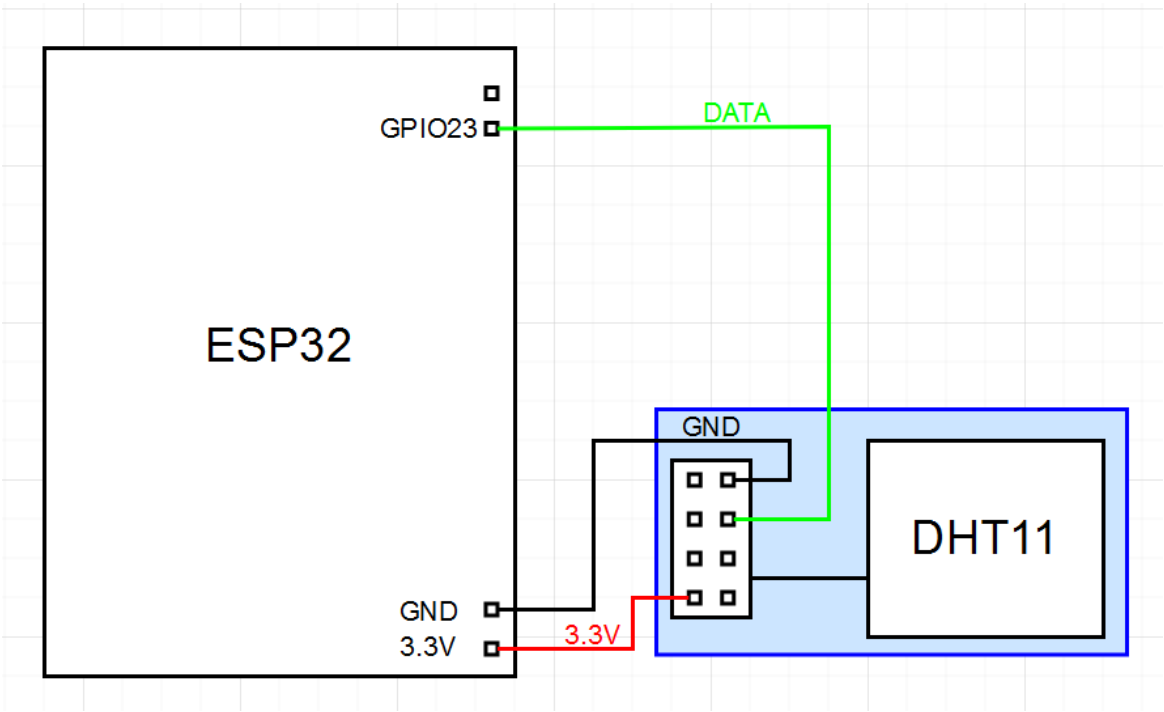
2.3 Schemat połączenia



Rysunek 4: Schemat połączeń w płytce DHT11



Rysunek 5: Schemat połączeń w płytce ESP32



Rysunek 6: Schemat połączenia

Płytka ESP32 była podłączona do płytki lutowanej w taki sposób że jedynie połowa portów była dostępna do użytku.Wybrane wejście GPIO23(General purpose input output) nie powinno wpływać na funkcjonowanie programu.

2.4 Kod programu

.

```
#include <WiFi.h>
#include <SPI.h>
#include <DHT.h>


#define DHTPIN 23
#define DHTTYPE DHT11


TaskHandle_t ReadData;
TaskHandle_t FormatData;


const int ledPin = 5;
const char ssid[] = "maszt_sygnalowy";


float temperature, humidity;
WiFiServer server(21);
char httpMSG[256];
char header[] = "HTTP/1.1 200 OK\nContent-type:text/html\nConnection: close\n";


```

```
DHT dht(DHTPIN, DHTTYPE);
;
void fUpdateData(void* params);
void fFormatData(void* params);


void setup() {
  Serial.begin(115200);
  Serial.printf("\nConnecting to WiFi...\n");
  dht.begin();
  // Connect to WiFi
  IPAddress ip(192, 168, 1, 22);
  WiFi.config(ip);
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid);
  Serial.printf("\nConnecting to WiFi...\n");


  while (WiFi.status() != WL_CONNECTED) {
    Serial.printf(".");
    delay(100);
  }


  Serial.printf("Connection successful\n");
  Serial.print("Connected to WiFi. My address: ");
  Serial.println(WiFi.localIP());


  server.begin();
  Serial.print(httpMSG);
  // Create tasks
  xTaskCreatePinnedToCore(fUpdateData, "Data Update", 2048, NULL, 1, &ReadData, 0);
  xTaskCreatePinnedToCore(fFormatData, "String Update", 2048, NULL, 1,
  ↪  &FormatData, 1);
}


void loop() {  // Listen for incoming clients WiFiClient client =
↪  server.available();
  WiFiClient client = server.available();
  if (client) {
    Serial.println("New client");
      // An HTTP request ends with a blank line
      bool currentLineIsBlank = true;


    while (client.connected()) {
      if (client.available()) {
        char c = client.read();
        Serial.println(httpMSG);


        client.print(httpMSG);
      }
    }
```

```
78      delay(15);
79      client.stop();
80      Serial.println("Client disconnected");
81    }
82  }
83
84
85  void fUpdateData(void* params) {
86    while (true) {
87      float prevHumidity = humidity;
88      float prevTemperature = temperature;
89      humidity = dht.readHumidity();
90      temperature = dht.readTemperature();
91      if (isnan(humidity) || isnan(temperature)) {
92        Serial.println("Failed to read from DHT sensor!");
93        temperature = prevTemperature;
94        humidity = prevHumidity;
95      }                                       // Random humidity between 0.0 and
        ↪  100.0
96      vTaskDelay(2000 / portTICK_PERIOD_MS);  // Delay for 1 second
97    }
98  }
99
100
101 void fFormatData(void* params) {
102   while (true) {
103     // Format the data into the HTTP message
104     sprintf(httpMSG, "%s<!DOCTYPE html>\n<html>\n<head>\n<title>ESP32 Web
        ↪  Server</title>\n</head>\n<body>\nTemperature today is: %.1f °C, and the
        ↪  humidity in the air is: %.1f%%</body>\n</html>\n", header, temperature,
        ↪  humidity);
105     //Serial.print(httpMSG);
106     vTaskDelay(1000 / portTICK_PERIOD_MS);  // Delay for 1 second
107   }
108 }
109
110
111
```

Fragment kodu 1: Fragment kodu z programu

3 Źródła

1. http://www.embeddeddev.pl/kurs-freertos-wprowadzenie/
2. https://www.freertos.org/Documentation/01-FreeRTOS-quick-start/01-Beginners-guide/01-RTOS-fundamentals
3. https://www.waveshare.com/wiki/DHT11_Temperature-Humidity_Sensor
4. http://wiki.sunfounder.cc/images/c/c7/DHT11_datasheet.pdf
5. https://en.wikipedia.org/wiki/Hygrometer
6. https://pl.wikipedia.org/wiki/Termistor