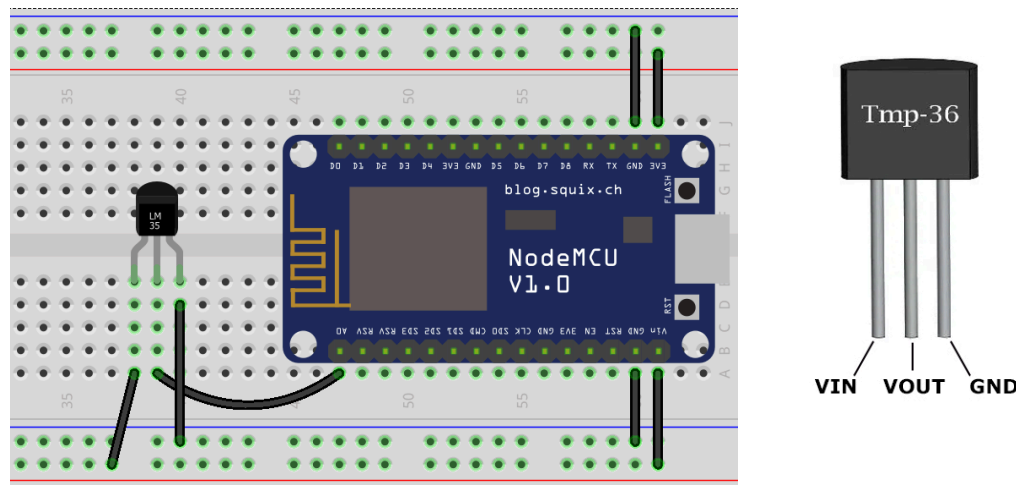


Zad.1. W środowisku Arduino IDE należy dodać następujące biblioteki:

- ESPAsyncTCP
- ESPAsyncWebServer
- ESPDash

Zad.2. Połączyć układ jak na poniższym rysunku.



Po wykonaniu wszystkich połączeń należy zaprogramować moduł *ESP8266* za pomocą poniższego programu.

```
// DASH 3.0.2
#define BAUDRATE 115200
#define TMP36pin A0

#include <ESP8266WiFi.h>
#include <ESPAsyncTCP.h>
#include <ESPAsyncWebServer.h>
#include <ESPDash.h>

AsyncWebServer server(80);

ESPDash dashboard(&server);
Card title_card(&dashboard, GENERIC_CARD, "Title");
Card temperature_card(&dashboard, TEMPERATURE_CARD,
                    "Temperature", "°C");
```

```
Card my_button(&dashboard, BUTTON_CARD, "Fire");

double temperature;
long p_millis=0;

const char* ssid = "Laboratorium-IoT";
const char* pass = "";

void setup() {
  WiFi.begin(ssid,pass);
  Serial.begin(BAUDRATE);
  Serial.print("Connecting network ");
  while(WiFi.status()!=WL_CONNECTED) {
    Serial.print("."); delay(500); }
  Serial.println(" done");
  Serial.print("IP adres: ");
  Serial.print(WiFi.localIP());

  server.begin();
  delay(500);
  String s = "Web Thermometer";
  title_card.update(s);
  dashboard.sendUpdates();

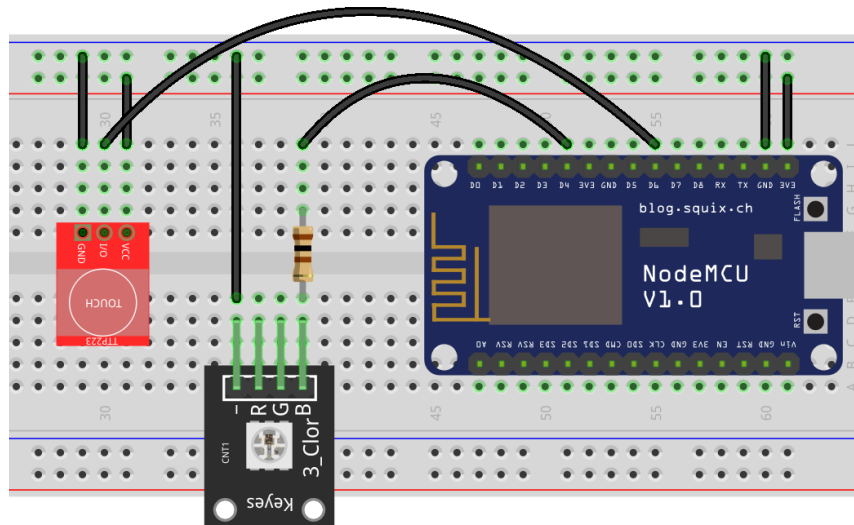
  my_button.attachCallback([&](bool value){
    Serial.println("Button: " + String((value)?"true":"false"));
    digitalWrite(LED_BUILTIN,!value);
    my_button.update(value);
    dashboard.sendUpdates(); });

  pinMode(LED_BUILTIN,OUTPUT);
}

#define MAIN_LOOP_DELAY 2000

void loop() {
  if(millis()-p_millis>MAIN_LOOP_DELAY) {
    temperature = ((analogRead(TMP36pin)/1024.0)*3300-500)/10;
    Serial.print("Temperature[degC] = ");
    Serial.println(temperature);
    temperature_card.update(String(temperature));
    dashboard.sendUpdates();
    p_millis = millis();}
}
```

Zad.3. Połączyć układ jak na poniższym rysunku.



Po wykonaniu wszystkich połączeń należy zaprogramować moduł *ESP8266* za pomocą poniższego programu.

```
#define BAUDRATE 115200
#define BUTTON 12          // D6
#define LIGHT_OUTPUT 2     // D4

#include <ESP8266WiFi.h>
#include <ESPAsyncTCP.h>
#include <ESPAsyncWebServer.h>
#include <ESPDash.h>

AsyncWebServer server(80);

ESPDash dashboard(&server);

Card my_button(&dashboard, BUTTON_CARD, "Light!");
Card my_dimmer(&dashboard, SLIDER_CARD, "Dimmer_blue", "", 0, 1023);
Card my_dimmerR(&dashboard, SLIDER_CARD, "Dimmer_red", "", 0, 1023);

long p_millis=0;

const char* ssid = "Light_switch";    // SSID
```

Ćwiczenie nr 2: Moduł ESP8266 - biblioteka DASH

```
const char* password = ""; //WiFi Password/Empty Password = Open AP

bool switch_state = false;
int light_intensity = 0;
int light_intensityR = 0;

void setup() {
  Serial.begin(BAUDRATE);
  WiFi.mode(WIFI_AP);
  WiFi.softAP(ssid, password);
  delay(1000);
  Serial.print("\nIP Address: ");
  Serial.println(WiFi.softAPIP());

  server.begin();
  delay(500);

  my_button.attachCallback([&](bool value){
    Serial.println("Button: " + String((value)?"true":"false"));
    switch_state = value;
    my_button.update(value);
    dashboard.sendUpdates(); });

  my_dimmer.attachCallback([&](int value){
    Serial.println("Light intensity: "+String(value));
    light_intensity = value;
    my_dimmer.update(value);
    dashboard.sendUpdates(); });

  my_dimmerR.attachCallback([&](int value){
    Serial.println("Light intensity: "+String(value));
    light_intensityR = value;
    my_dimmer.update(value);
    dashboard.sendUpdates(); });

  attachInterrupt(digitalPinToInterrupt(BUTTON), changeLight, RISING);
}

#define MAIN_LOOP_DELAY 100

void loop() {
  if(millis() - p_millis > MAIN_LOOP_DELAY)
    if(switch_state) { analogWrite(LIGHT_OUTPUT, light_intensity);
      analogWrite(LIGHT_OUTPUT, light_intensityR); }
```

```
    else { analogWrite(LIGHT_OUTPUT,0);  
    analogWrite(LIGHT_OUTPUT,0);}  
}  
  
ICACHE_RAM_ATTR void changeLight() {  
    switch_state = !switch_state;  
    Serial.println(switch_state);  
    my_button.update(switch_state);  
    dashboard.sendUpdates();  
}
```

Zad.4. Zmodyfikować program z zadania 2 pod kątem wykorzystanie czujnika BMP-280. Wyświetlić na wyświetlaczu wartość temperatury, ciśnienia. Te same wartości opublikować w sieci bezprzewodowej.

Dla zainteresowanych:

1. ESP-DASH:

github.com/ayushsharma82/ESP-DASH

2. Dokumentacja biblioteki DASH:

docs.espdash.pro