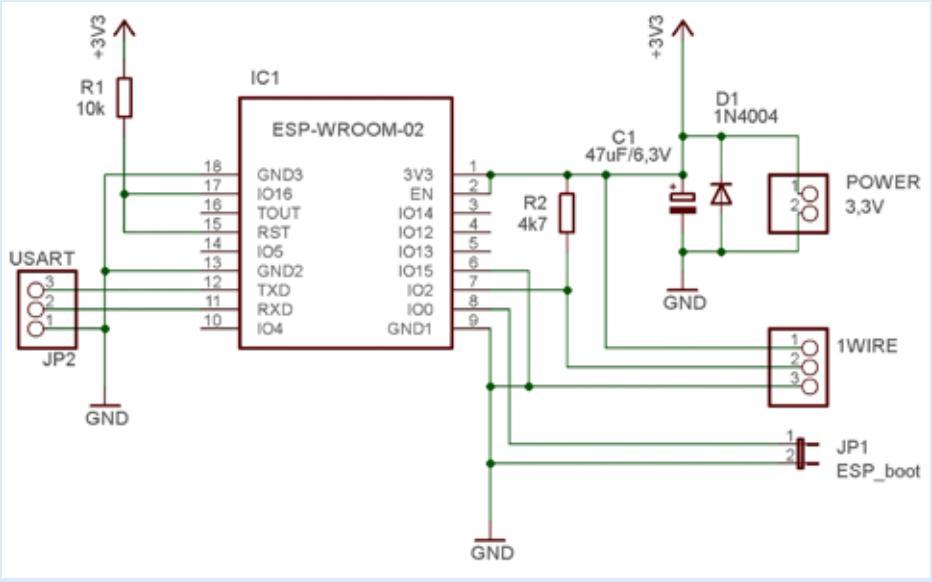
Rejestracja i wizualizacja temperatury

Tani system do zdalnego monitorowania temperatury pieca lub centralnego ogrzewania w domku jednorodzinnym, ogrzewanym za pomocą starszego typu pieców na węgiel. Problem dotyczy potencjalnej możliwości wygaśnięcia takiego pieca pod nieobecność właściciela z różnych powodów. Wiadomo, jakie skutki może to za sobą pociągać, gdy na zewnątrz panuje mróz.

Ponieważ system ma być przede wszystkim niskobudżetowy, to uznałem, że najlepszym rozwiązaniem będzie użycie tanich modułów Wi-Fi i wizualizacji danych za pomocą darmowych serwisu internetowego. Założyłem, że w domu musi być dostęp do Internetu.

Przeszukując oferty darmowej akwizycji danych trafiłem na stronę ***https://thingspeak.com***. Portal utworzono do przechowywania i wizualizacji danych. Darmowe korzystanie z usług jest okupione pewnymi ograniczeniami: maksymalna ilość wysłanych danych ograniczona do 3 mln rekordów/rok, minimalny odstęp pomiędzy wysyłanymi rekordami 15 sekund oraz ważność konta przez 1 rok. Biorąc jednak pod uwagę łatwość korzystania z serwisu ograniczenia nie są tak istotne.



Rysunek 1. Schemat modułu Wi-Fi Wroom-2 z czujnikiem temperatury DS18B20

Wśród modułów Wi-Fi dostępnych na rynku wyróżniają się te oparte o układ ESP8266, ze względu na dostępność i niską cenę. Dodatkowych atutem jest możliwość ich programowania z użyciem IDE Arduino. Zdecydowałem się na użycie modułu typu Wroom-02, który to wykorzystałem do budowy opisywanej już na łamach EP 1/2017 stacji THPS. Patrząc na schemat (**rysunek 1**), można powiedzieć, że do budowy urządzenia użyto „surowego" modułu Wroom-02, czujnika temperatury DS18B20, 2 rezystorów i zasilacza 3,3 V (w tym przypadku dwie baterie AAA). Dioda D1 zabezpiecza moduł na wypadek odwrotnego zasilania, a kondensator C1 podtrzymuje działanie modułu podczas wymiany baterii.

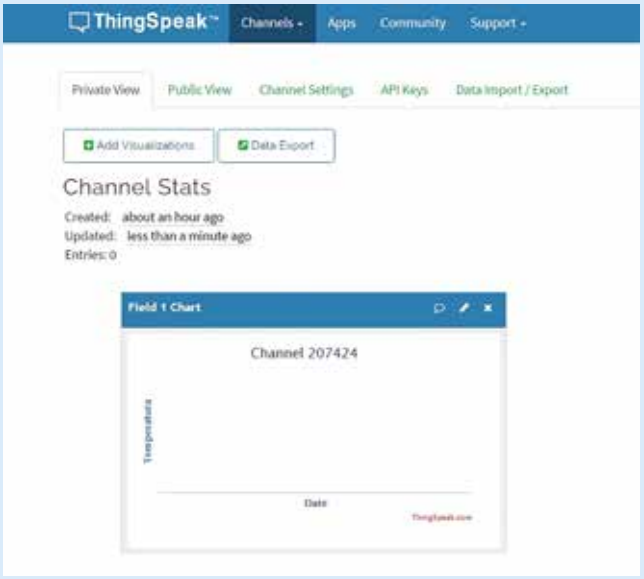
Aby można było pisać programy dla ESP8266, należy odpowiednio skonfigurować IDE Arduino. Odsyłam tutaj chociażby do tutoriala dostępnego na stronie Sparkfun: ***https://goo.gl/eWRBQq***. Następnie należy założyć konto na ***https://thingspeak.com***. Procedura jego założenia jest typowa i nie wymaga omawiania.

Po zalogowaniu się do naszego nowoutworzonego konta przechodzimy do zakładki „Channels/my channel„ i klikamy na zielony prostokąt z napisem „New Channel". Otworzy nam się nowa strona z konfiguracją pól pomiarowych. Możemy utworzyć

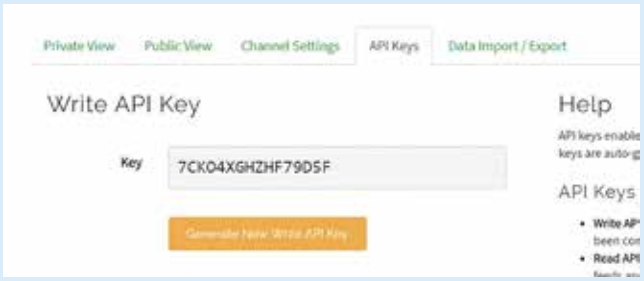
Listing 1. Przykładowy program

```
#include <ESP8266WiFi.h>
#include <DallasTemperature.h>
#define ONE_WIRE_BUS 2   // numer pinu z DS18B20

OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature DS18B20(&oneWire);
String apiKey = „7CK04XGHZHF79D5F";          // klucz API Key
const char* server = „api.thingspeak.com";

void setup() {
    int tries=0;
    WiFi.mode(WIFI_STA);
    WiFi.begin(„SSID", „HASLO");  // dane dostępu do sieci Wi-Fi

    while (WiFi.status() != WL_CONNECTED) {  //czy połączeno z Wi-Fi
        delay(10);
        tries++;    //zwiększaj zmienną pomocniczą co 10ms
                    //jeśli brak połączenia przez 30s to uśpienie na 5min
        if (tries > 3000) ESP.deepSleep(30000000);
    }
}

void loop() {
    float temp;  //zmienna przechowująca temperaturę
    DS18B20.requestTemperatures();  //pomiar temperatury
    delay(1000);                    //opóźnienie 1s
    temp = DS18B20.getTempCByIndex(0); //odczytanie temperatury
    WiFiClient client;
    if (client.connect(server,80)) {   //"184.106.153.149" or api.thingspeak.com
        String postStr = apiKey;    // składanie tekstu  do wysłania na serwer
        postStr +="&field1=";
        postStr += String(temp);
        postStr += „\r\n\r\n";
        client.print(„POST /update HTTP/1.1\n");
        client.print(„Host: api.thingspeak.com\n");
        client.print(„Connection: close\n");
        client.print(„X-THINGSPEAKAPIKEY: „+apiKey+"\n");
        client.print(„Content-Type: application/x-www-form-urlencoded\n");
        client.print(„Content-Length: „);
        client.print(postStr.length());
        client.print(„\n\n");
        client.print(postStr);
        client.stop();
        delay(200); //odczekaj 200ms
        ESP.deepSleep(90000000); //uśpienie na 15 minut
    }
}
```

Poprzednie części kursu i dodatkowe materiały dostępne są na FTP: ftp://ep.com.pl, user: 77322, pass: 8qxonzsb



Rysunek 2. Prawidłowo utworzone pole z wykresem dla temperatury



Rysunek 3. Klucz niezbędny do zapisywania danych

do 8 pól – „Field1-8”. Ponieważ chcemy mieć tylko jeden wykres z temperaturą zaznaczamy, prostokąt z prawej strony przy polu „Field1”. Jednocześnie w tabeli „Field Label 1” wpisujemy etykietę – „temperatura”. Poniżej możemy zaznaczyć pole „Make Public”. Zaznaczenie tej opcji spowoduje, że nasze dane/wykres będą widoczne dla innych użytkowników serwisu. Dla nas również jest to wygodne, ponieważ pozwala na podgląd danych bez konieczności logowania do serwisu. Na koniec klikamy „Save Channel”. W tym momencie otworzy nam się zakładka „Private Viev” z jednym wykresem. Dodatkowo, zostaną utworzone zakładki „Public Viev”, „Channel Settings”, „API Keys”, „Data Import/Export” (**rysunek 2**). W zakładce „Channel Settings” możemy dodawać kolejne pola, unikalny opis kanału, wprowadzić tagi, które będą identyfikować kanał itp. Kluczowa dla nas jest zakładka „API Keys”, na której jest podawany „Write API Key” – należy go zanotować.

Gdy mamy klucz oraz skonfigurowane środowisko Arduino przystępujemy do pisania oprogramowania na moduł Wi-Fi. Przykładowy program pokazano na **listingu 1**.

Program ogranicza się do obsługi czujnika DS18B20 oraz do wysłania temperatury na serwer. Kluczowe jest tutaj wpisanie poprawnego klucza **API Key** oraz danych dostępowych do naszej sieci Wi-Fi, za pomocą której będziemy łączyć się z Internetem. Odczytana temperatura składana jest wraz z kluczem API w string poleceniami String postStr. W naszym przypadku wysyłamy tylko jeden parametr – temperaturę i trafia ona do pola o nazwie „Field1”. Gdybyśmy mierzyli dodatkowo inne parametry np. temperaturę i ciśnienie z drugiego czujnika, temperaturę i wilgotność z trzeciego czujnika i mieli w sumie 5 parametrów do wysłania to składanie stringa miałoby postać:

```
String postStr = apiKey;
postStr +="&field1=";
postStr += String(temp1);
postStr +="&field2=";
postStr += String(temp2);
postStr +="&field3=";
postStr += String(pressure);
postStr +="&field4=";
postStr += String(temp3);
postStr +="&field5=";
postStr += String(humidity);
postStr += „\r\n\r\n”;
```

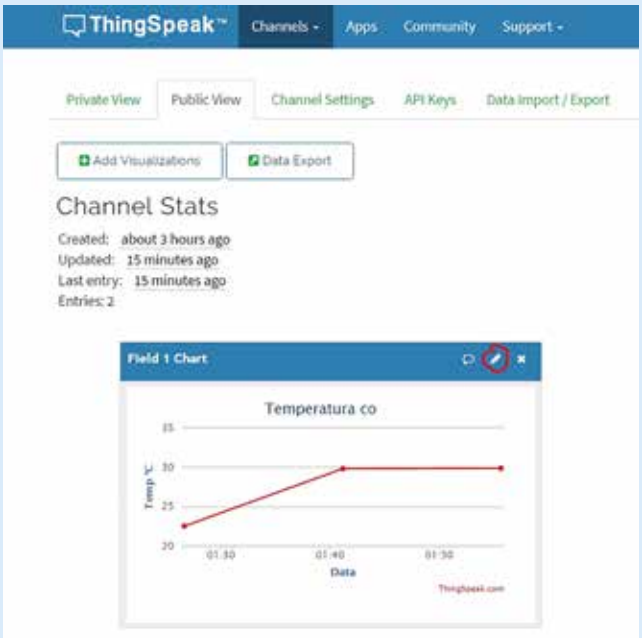
Oczywiście należałoby zaznaczyć w zakładce „Channel Settings” pola „Field1-5” w naszym kanale pomiarowym.

Po wysłaniu danych na serwer moduł przechodzi w stan uśpienia z obniżonym poborem prądu. W tym czasie pracuje tylko RTC, pozostałe bloki funkcjonalne są wyłączone. Instrukcja **ESP.deepSleep(xxx);** powoduje przejście w stan głębokiego uśpienia – jej argumentem jest czas podawany w mikrosekundach. Po tym czasie na pin IO16 jest zerowany, a ponieważ jest połączony z pinem RST, następuje restart modułu i program rozpoczyna pracę od nowa. Na listingu widać również, że moduł zostanie wprowadzony w uśpienie, jeśli nie uda mu się zalogować do sieci Wi-Fi. Ma to na celu ochronę akumulatorków/baterii, z których jest zasilany. Z doświadczenia wiem, że przy dobrym zasięgu sieci moduł loguje się w ciągu 2…3 sekund, przy słabym – do 10 sekund. Dlatego, jeśli w 30 sekund nie uda mu się zalogować, to lepiej jest spróbować za kilka minut, niż rozładować baterie.

Osie wykresu można edytować klikając symbol ołówka – zaznaczony w czerwonym kółku. Zachęcam również do poznania możliwości ustawień różnych opcji w tej samej zakładce – jak chociażby liczba wyświetlanych rekordów, uśrednianie wyników itp.

Na rynku mamy ofertę różnych modułów WiFi opartych o układ ESP8266. W podobnej cenie jak Wroom-02 można kupić moduł ESP-12E z płytką i wyprowadzonymi wszystkimi pinami. Mając do dyspozycji bardzo bogatą listę czujników obsługiwanych przez Arduino możemy zbudować praktycznie nieograniczony system czujników. Ograniczeniem jest tylko nasza pomysłowość i chęć eksperymentowania, do czego gorąco zachęcam.

Grzegorz Burzyński,
Sp5ein@gmail.com



Rysunek 4. Wykres z pomiarami temperatury w odstępach 15-minutowych

Poprzednie części kursu i dodatkowe materiały dostępne są na FTP: **ftp://ep.com.pl**, user: **77322**, pass: **8qxonzsb**