Povežimo DHT11/DHT22 senzor na oblak z ESP8266

Naučite se, kako ustvariti preprosto spletno vremensko postajo s pomočjo senzorja DHT11, povezanega s ploščo z ESP8266 in Arduino okoljem.

Kaj potrebujemo v tem projektu?

Hardverske komponente:
- *NodeMCU ESP8266 razvojna plošča × 1*
- *DHT11 senzor temperature in vlage (3 priključki) × 1*

Softverske aplikacije in spletne storitve:
- *Cloud4RPi*
- *Microsoft VS Code*

Zgodba

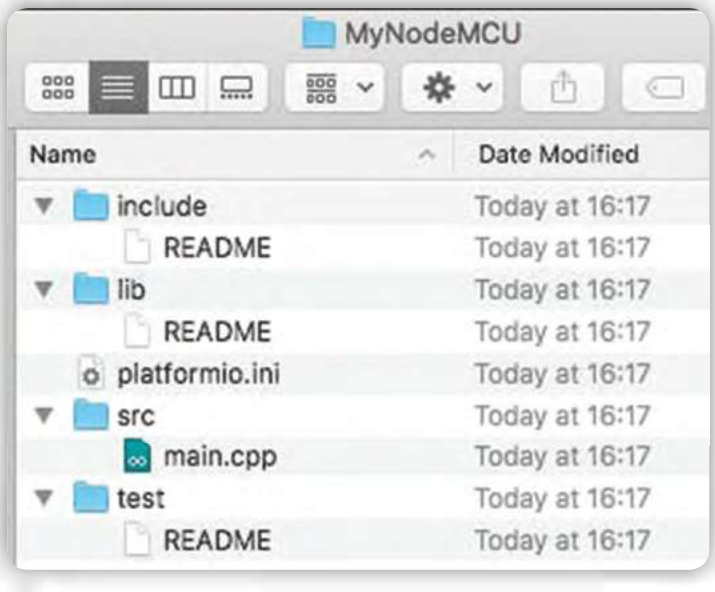
V enem od mojih člankov [1] sem objavil, kako povezati NodeMCU razvojno ploščo, ki temelji na ESP8266 in na Cloud4RPI storitvi. Zdaj je čas, da naredimo realen projekt.

Hardverske zahteve

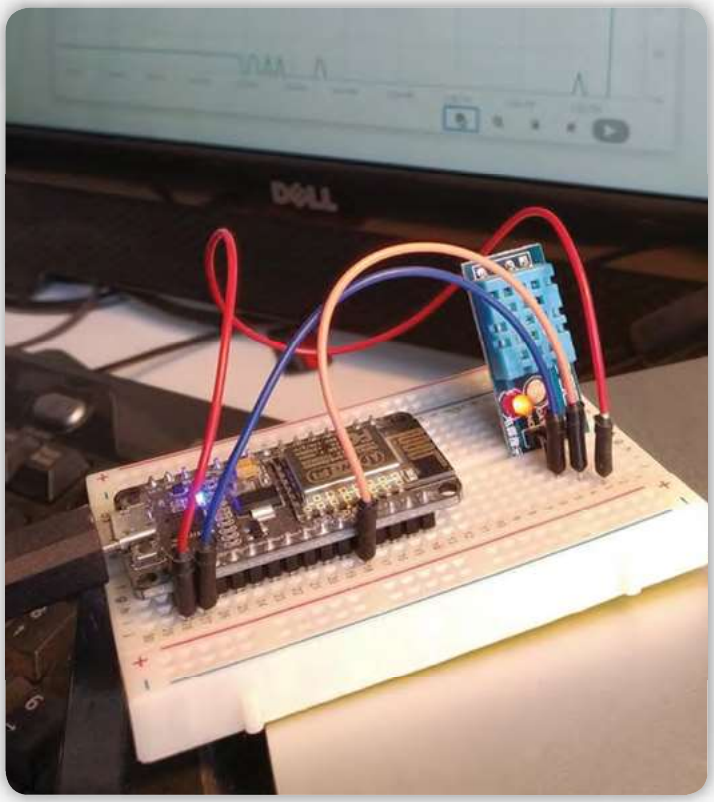
Potrebujemo katero koli razvojno ploščo, ki temelji na ESP8266 modulu. Kot senzor lahko uporabimo DHT11 ali DHT22.

Programska oprema in storitve

- *DHT sensor library by Adafruit — v1.3.7*
- *Adafruit Unified Sensor — v1.0.3*
- *cloud4rpi-esp-arduino — v0.1.0*
- *Cloud4RPI — Cloud control panel for IoT devices*
- *PlatformIO IDE for VSCode*



Slika 1: Konfiguracija projekta



Cilj: meriti temperaturo in vlago

Senzor DHT11 sem že imel v predalu, zato sem se odločil, da ga bom uporabil za merjenje temperature in vlažnosti. Izberimo Arduino knjižnico za branje podatkov s senzorja.

Arduino register vsebuje več knjižnic, iz katerih sem izbral najbolj priljubljeno.

Glede na njihov GitHub moramo dodati tudi Adafruit Unified Sensor paket.

1. korak: kreirajte in konfigurirajte projekti

Opisal sem že, kako ustvariti PlatformIO projekt in namestimo knjižnice v prvem delu [1]. Moj projekt se imenuje "MyNodeMCU". Struktura je prikazana na sliki 1.

Ta projekt je nekoliko spremenjen Cloud4RPi projekt. Odločil sem se, da namesto kode shranim žeton naprave in poverilnice za Wi-Fi v konfiguracijsko datoteko. Kako

izgleda platform.io datoteka si oglejte na spodaj v programu: Platform.io datoteka

```
[platformio]
default_envs = nodemcuv2
[env:nodemcuv2]
platform = espressif8266
framework = arduino
board = nodemcuv2
```

2. korak: namestite knjižnice

Namestitev knjižnic je precej preprosta. To lahko storite znotraj IDE grafičnega vmesnika ali z dodajanjem potrebnih imen knjižnic v razdelek lib_deps datoteke platform.io:

```
; Library options
lib_deps =
cloud4rpi-esp-arduino
Adafruit Unified Sensor
DHT sensor library
build_flags =
-D MQTT_MAX_PACKET_SIZE=1024
-D MQTT_MAX_TRANSFER_SIZE=128
-D CLOUD4RPI_DEBUG=0
-D SSID_NAME=\"__YOUR_WIFI__\"
-D SSID_PASSWORD=\"__YOUR_WIFI_PASS__\"
-D CLOUD4RPI_TOKEN=\"__YOUR_DEVICE_TOKEN__\"
```

Dodane knjižnice bodo samodejno nameščene v podmapo projekta, slika 2.

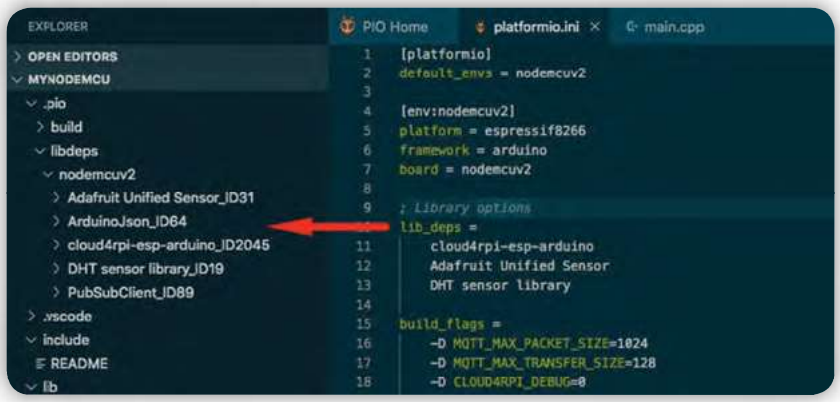
Glava main.cpp izgleda tako:

```
#include <Arduino.h>
#include <ESP8266WiFi.h>
#include <Cloud4RPi.h>
#include "DHT.h"
```

3. korak: priključite DHT11 senzor

Adafruit ponuja DHTtester.ino program kot primer vzorčne povezave. Ta program inicializira senzor in definira strukturo za shranjevanje rezultata meritev (v primeru, da je bila uspešna):

```
#define DHTPIN 2      // Digital pin connected to
the DHT sensor#define DHTTYPE DHT11 // DHT 11
// ...
DHT dht(DHTPIN, DHTTYPE);
dht.begin();
```



Slika 2: Odvisnosti IO knjižnice

```
// ...
structDHT_Result {
float h;
float t;
};
DHT_Result dhtResult;
```

Naslednja funkcija prikazuje, kako prebrati podatke senzorjev in jih shraniti v zgoraj opisano strukturo podatkov.

```
void readSensors() {
float h = dht.readHumidity();
// Read temperature as Celsius (the default)
float t = dht.readTemperature();
// Check if any reads failed and exit
if (isnan(h) || isnan(t)) {
Serial.println(F("Failed   to   read   from   DHT
sensor!"));
return;
}
dhtResult.h = h;
dhtResult.t = t;
}
```

4. korak: pošiljanje podatkov v oblak

Ko imamo te podatke, je naslednji korak, da jih pošljemo v Cloud4RPi storitev. Cloud4RPi stran za Arduino opisuje API knjižnice, ki je niz metod, ki se uporabljajo za:

- *ustvarite, preberite in posodobite spremenljivke,*
- *pošljite vrednosti spremenljivk v oblak z MQTT protokolom*

Knjižnica podpira tri spremenljive vrste: Bool, Numeric in String. Delovni potek knjižnice se začne z ustvarjanjem primerka API-ja s pomočjo žetona naprave s spletnega mesta cloud4rpi.io (podrobnosti najdete v članku [1]).

```
#if defined(CLOUD4RPI_TOKEN)
Cloud4RPi c4r(CLOUD4RPI_TOKEN);
```

```
#else
Cloud4RPi c4r("!!!_NO_DEVICE_TOKEN_!!!");
#endif
```

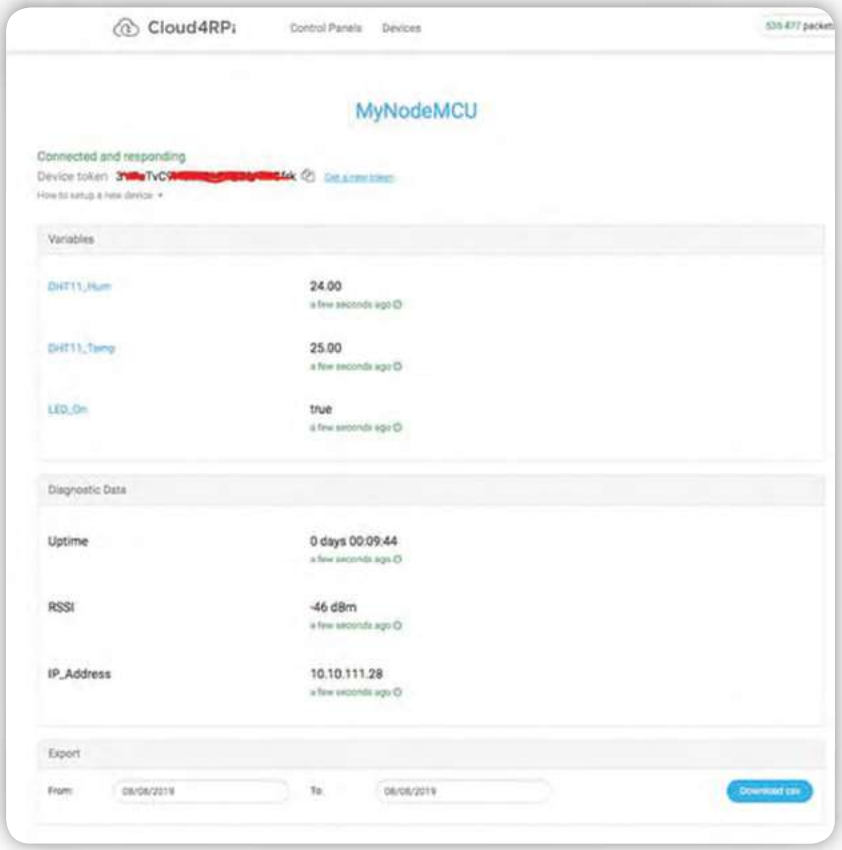
Nato označite spremenljivke za odčitke DHT11:

```
c4r.declareNumericVariable("DHT11_Temp");
c4r.declareNumericVariable("DHT11_Hum");
```

Nato pridobite podatke s senzorja, jih shranite v spremenljivke in jih objavite v Cloud4RPi:

```
c4r.setVariable("DHT11_Temp", dhtResult.t);
c4r.setVariable("DHT11_Hum", dhtResult.h);
c4r.publishData();
```

Temperatura in vlaga se ne spreminjata hitro, zato pošiljanje več kot ene vrednosti na 5 minut ni potrebno.



Slika 3: Clud4RPI stran z napravami

5. korak: diagnostika

Cloud4RPi podpira diagnostične podatke skupaj s spremenljivimi vrednostmi. Kot diagnostične podatke sem uporabil uptime, moč signala Wi-Fi in IP naslov:

```
c4r.declareDiagVariable("IP_Address");
c4r.declareDiagVariable("RSSI");  //  WiFi  signal
strength
c4r.declareDiagVariable("Uptime");
```

Opomba: funkcija milis, ki jo uporabljam za pridobitev ponastavitve, se ponastavi na nič vsakih približno 50 dni. Kar je za moj projekt več kot dovolj. Naslednja koda nastavi diagnostične spremenljivke:

```
c4r.setDiagVariable("RSSI", (String)WiFi.RSSI() +
" dBm");
c4r.setDiagVariable("IP_Address",  WiFi.localIP().
toString());
c4r.setDiagVariable("Uptime", uptimeHumanReadable(
currentMillis));
c4r.publishDiag();
```

Funkcija uptimeHumanReadable pretvori milisekunde v priročno obliko:

```
StringuptimeHumanReadable(unsignedlong milliseconds)
{
staticchar uptimeStr[32];
unsignedlong secs = milliseconds / 1000;
unsignedlong mins = secs / 60;
unsignedint hours = mins / 60;
unsignedint days = hours / 24;
secs -= mins * 60;
mins -= hours * 60;
hours -= days * 24;
sprintf(uptimeStr,"%d   days   %2.2d:%2.2d:%2.2d",
(byte)days, (byte)hours, (byte)mins, (byte)secs);
returnString(uptimeStr);
}
```

Funkcija odda tako niz kot 5 dni 10:23:14 namesto čudno velikega števila.

6. korak: projekt zaženite in ga razhroščite

Ko zbere ustvarjeno kodo in jo preklopi v NodeMCU, se naprava poveže v storitev v oblaku in začne pošiljati podatke. Določitev pogostost beleženja lahko povečate tako, da spremenljivko predprocesorja CLOUD4RPI_DEBUG nastavite na 1 (dodajte -D CLOUD4RPI_DEBUG = 1 za odsek build_flags v datoteki platform.io).

Nato odprite spletno mesto cloud4rpi.io in opazite novo napravo v spletu. Odprite jo in si oglejte vse spremenljive vrednosti, prejete od naprave: senzor in diagnostiko.