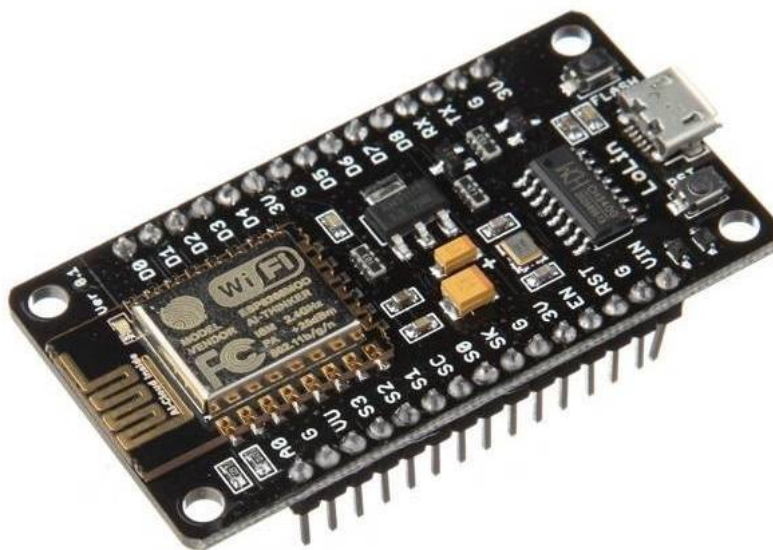


NodeMCU WiFi Data Logger

<http://pvmonitor.pl>



NodeMCU LoLin V3 ESP-12E

Co to jest data logger?

Data logger jest urządzeniem zbierającym dane i zapisującym je lokalnie (np. na karcie pamięci) lub wysyłającym je na zewnętrzny serwer (np. <http://pvmonitor.pl>)

Co umożliwia NodeMCU WiFi Data Logger?

- zliczanie impulsów pochodzących z zewnętrznych liczników energii elektrycznej (pomiar produkcji i zużycia energii w kWh),
- podłączenie do 6 liczników energii jednocześnie (3 liczniki produkcji i 3 liczniki zużycia),
- wysyłanie danych o produkcji/zużyciu energii na serwer <http://pvmonitor.pl>
- po zmianie w programie realizację innych funkcji (np.: pomiar napięcia, prądu, temperatury),

Dlaczego NodeMCU?

- wbudowany moduł WiFi *ESP8266* wraz z anteną PCB 2,4GHz (łączość bezprzewodowa bez konieczności prowadzenia sieci LAN),
- wbudowany zasilacz 3,3V dla układu WiFi (możliwość bezpośredniego zasilania z portu USB komputera lub np. za pomocą typowej taniej ładowarki do smartfona ze złączem micro USB),
- wbudowana przejściówka USB<->COM (możliwość programowania układu *ESP8266* i podmiany firmware),
- wyprowadzone piny GPIO układu *ESP8266* (możliwość podłączenia urządzeń peryferyjnych, np.: licznik energii z wyjściem impulsowym, przetwornik ADC na magistrali I2C do pomiaru napięcia i prądu DC, wyświetlacz LCD lub OLED na magistrali I2C, czujników temperatury 1wire, itp.),
- programowanie z poziomu bezpłatnego i łatwego w użyciu środowiska *Arduino IDE*
- bardzo dobry stosunek możliwości do ceny (koszt modułu nie przekracza kilkudziesięciu złotych), niewielkie wymiary modułu,

Kilka słów o układzie WiFi ESP8266

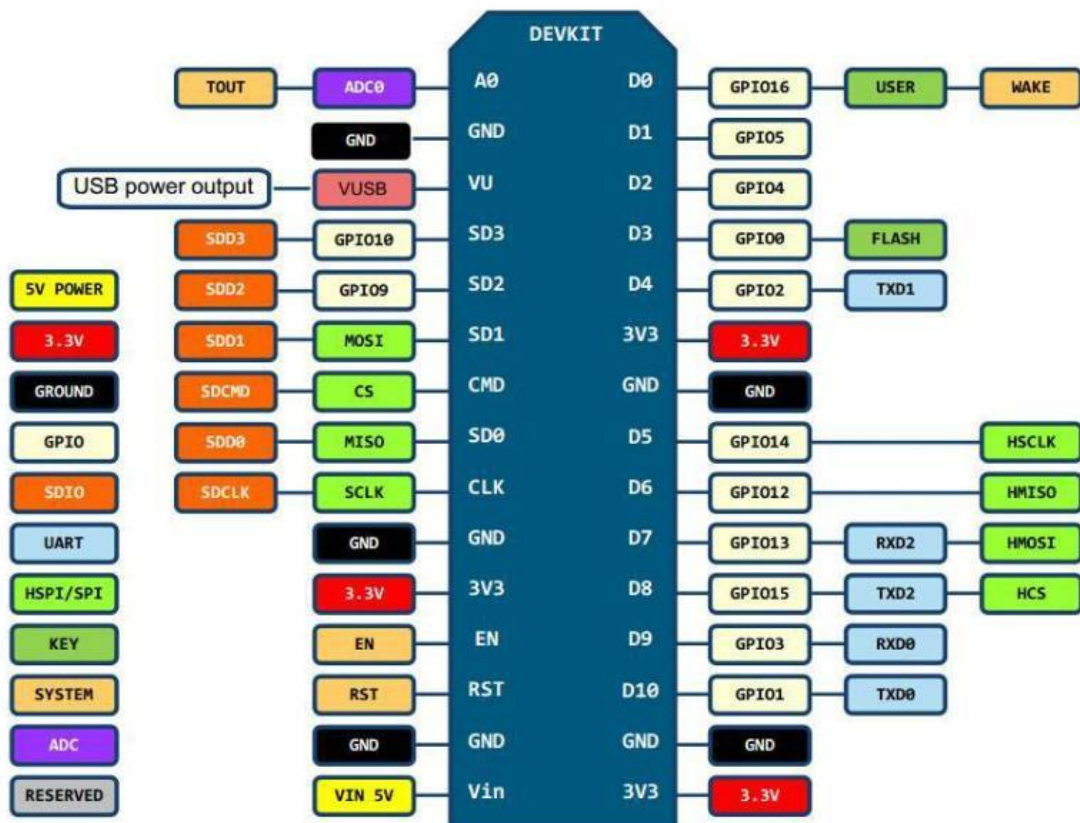


ESP8266 to bardzo tani układ WiFi 2,4GHz IEEE 802.11 b/g/n chińskiej firmy *Espressif*. Posiada na pokładzie m.in. wbudowany stos TCP/IP, procesor 32 bitowy z pamięcią RAM oraz 10 bitowy przetwornik ADC. Umożliwia obsługę innych układów poprzez popularne protokoły transmisji SPI, I²C, I²S.

Producent wprowadził na rynek wiele różnorodnych modułów opartych o [ESP8266](#), różniących się wielkością i wyposażeniem.

Schemat wszystkich wejść/wyjść modułu *NodeMCU*.

PIN DEFINITION



D0(GPIO16) can only be used as gpio read/write, no interrupt supported, no pwm/i2c/ow supported.

Wejścia modułu *NodeMCU* wykorzystywane do zliczania impulsów z liczników:

- produkcja energii (3 szt.): **D1, D2, D3**
- zużycie energii (3 szt.): **D5, D6, D7**

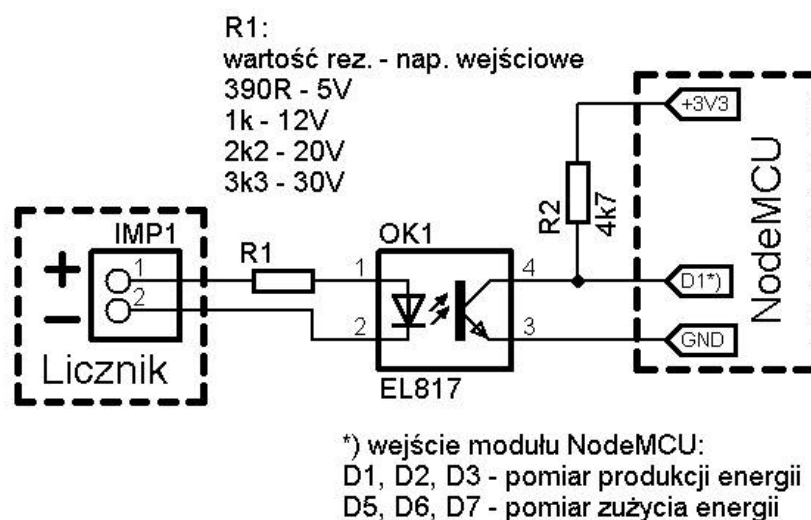
Przygotowany program data loggera pozwala na zliczanie impulsów jednocześnie nawet z sześciu liczników energii. Należy wykorzystać liczniki z wyjściem impulsowym 1000 imp./kWh. Zastosowanie innego typu licznika wymaga odpowiedniej zmiany w programie urządzenia.

Co będzie potrzebne do zbudowania podstawowej wersji *NodeMCU WiFi Data Loggera*?

- moduł *NodeMCU* lub jego klon np. *LoLin V3*,
- elementy elektroniczne, w zależności od potrzeb i zaawansowania użytkownika:
 - a) 2 transoptory (np. EL817), 2 rezystory 0,25W (wartość rezystancji wg schematu zamieszczonego poniżej), 2 rezystory 0,25W 4k7, złącze typu goldpin 2,54mm do połączenia z modułem, złącza ARK do podłączenia przewodów, kawałek uniwersalnej płytki drukowanej,
 - b) lub wykonana gotowa podstawa dla modułu *NodeMCU LoLin V3*, umożliwiająca montaż układu w obudowie na szynę DIN (szczegóły w dalszej części tekstu)
- kabel micro-USB typu B / USB typu A,
- komputer z portem USB typu A (na potrzeby programowania układu),
- 2 liczniki energii elektrycznej AC z wyjściem 1000 impulsów/1kWh (jeżeli chcemy monitorować tylko samą produkcję energii wystarczy jeden licznik), np. *ORNO OR-WE-501*

Opcja minimum - schemat układu wejść pomiarowych do zliczania impulsów

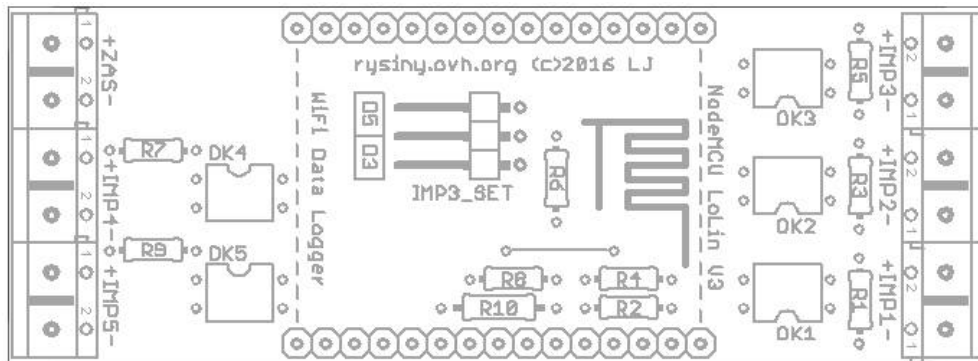
Aby bezpiecznie i precyzyjnie zliczać impulsy pochodzące z zewnętrznego licznika energii należy na fragmencie uniwersalnej płytki drukowanej wykonać prosty układ według poniższego schematu. Przedstawiony układ pozwala precyzyjnie zliczać impulsy o czasie trwania nawet poniżej 1ms. Zastosowanie transoptorów zabezpiecza wejścia modułu przed uszkodzeniem wskutek ewentualnego przekroczenia dopuszczalnej wartości napięcia wejściowego, oraz izoluje galwanicznie licznik od data loggera. **Układ ze schematu należy zastosować na każdym z wejść impulsowych, do których planujemy podłączenie zewnętrznego licznika energii.**



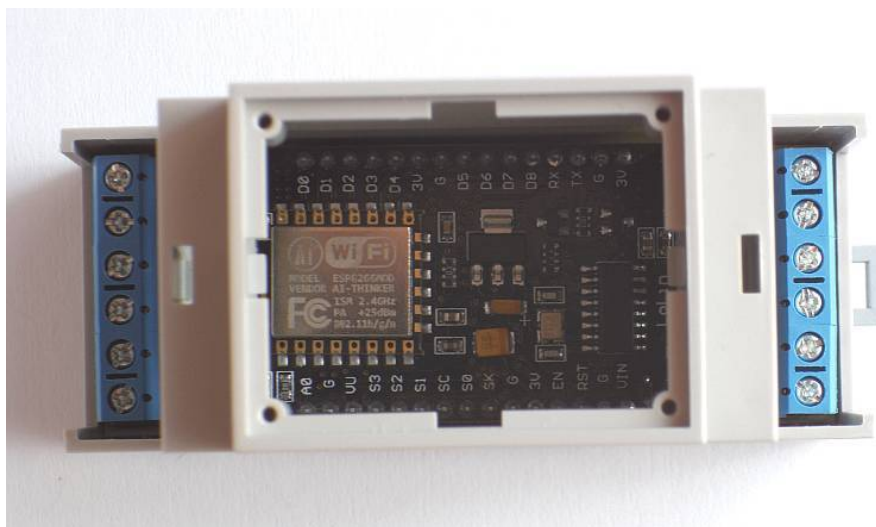
Wartość rezystora $R1$ ograniczającego prąd diody LED transoptora należy dobrać na podstawie wartości napięcia wyjścia impulsowego danego licznika energii. Dla napięcia impulsu o wartości 5V wartość rezystora powinna wynosić około 390R (12V -> 1k, 20V -> 2k2, 30V -> 3k3).

UWAGA: Powyższe typowe wartości rezystora $R1$ obliczono przy założeniu, że podczas trwania impulsu przez diodę LED transoptora płynie prąd około 10mA. Można zastosować nieco inne wartości rezystora dla danego napięcia, jednak różnica nie powinna przekraczać $\pm 20\%$. Zastosowanie innych wartości rezystora może powodować błędne zliczanie impulsów lub uszkodzić diodę transoptora.

Gotowa podstawka dla modułu NodeMCU LoLin V3



Przedstawiony [projekt PCB](#) pozwala umieścić NodeMCU WiFi Data Logger w typowej obudowie na szynę DIN.



Z powodu konieczności zasilania modułu NodeMCU bezpośrednio z zacisków na płytce PCB podstawka posiada tylko 5 niezależnych wejść impulsowych do zliczania impulsów pochodzących z liczników energii:

$IMP1 - D1$ (produkcja energii),

$IMP2 - D2$ (produkcja energii),

$IMP4 - D6$ (zużycie energii),

$IMP5 - D7$ (zużycie energii),

$IMP3$ – wejście konfigurowalne za pomocą zworki $IMP3_SET$ – można ustawić jako $D3$ (produkcja energii) lub $D5$ (zużycie energii).

UWAGA: Zalecane napięcie zasilania układu, przyłączone do zacisków ZAS: 5V-6V/500mA DC. Zbyt wysokie napięcie zasilania może spowodować uszkodzenie stabilizatora 3V3 wbudowanego w moduł NodeMCU.

Jakie możliwości oferuje udostępniony program?

- 1) Konfiguracja ustawień urządzenia z poziomu środowiska *Arduino IDE* przed kompilacją programu.
- 2) Komunikacja z użytkownikiem poprzez port RS232 z poziomu *Arduino IDE* lub z użyciem zewnętrznego programu typu terminal (np. *PuTTY*). Data logger podczas pracy przesyła do terminala najważniejsze informacje.
- 3) Wyszukiwanie dostępnych sieci WiFi (domyślnie wyłączone z kompilacji). Podczas instalacji urządzenia w układzie docelowym funkcja ta pozwala podejrzeć w terminalu, jakie sieci WiFi są widziane przez układ *ESP8266* i jaka w danym miejscu jest siła ich sygnału (RSSI). Podczas normalnej pracy data loggera zalecam wyłączyć tę funkcję, gdyż jej wykonanie zajmuje sporo czasu mikroprocesora i o kilka sekund wydłuża interwał pomiędzy wysyłaniem danych na serwer.
- 4) Automatyczne podłączanie do ustawionej sieci WiFi. Bieżąca kontrola połączenia z siecią i automatyczne wznawianie utraconego połączenia. Wbudowana dioda LED sygnalizuje status podłączenia do sieci WiFi (LED wł. = podłączony, LED wył. = niepodłączony).
- 5) Cykliczne wysyłanie wybranych danych na serwer <http://pvmonitor.pl> (co około 60s). Funkcja weryfikuje skuteczność przesłania danych poprzez analizę odpowiedzi serwera. W przypadku braku potwierdzenia odbioru przez serwer, transmisja danych jest automatycznie powtarzana (dodatkowe 2 próby). Wysyłanie danych jest sygnalizowane błyskiem wbudowanej diody LED.

Jak wykonać własny data logger?

- 1) **a)** W zależności od ilości wykorzystywanych liczników energii, wykonujemy odpowiednią ilość układów wejść pomiarowych na podstawie zamieszczonego wcześniej schematu. Układy można zlutować na kawałku uniwersalnej płytki drukowanej, lub na tzw. pająka, ale nie polecam tej drugiej opcji (ze względu na ryzyko wystąpienia przypadkowych zwarcí).
b) lub wykonujemy gotową podstawkę dla modułu *NodeMCU* według [projektu PCB](#).

- 2) Pobieramy i instalujemy środowisko *Arduino IDE* ze strony <https://www.arduino.cc/en/Main/Software>
- 3) Otwieramy *Arduino IDE*, pobieramy i instalujemy pakiet dla płytek *NodeMCU*:

- otwieramy *Plik* -> *Preferencje*
- w zakładce *Ustawienia* w polu *Dodatkowe adresy URL do menedżera płytek* wklejamy następujący link: http://arduino.esp8266.com/staging/package_esp8266com_index.json i potwierdzamy klawiszem *OK*
- otwieramy *Narzędzia* -> *Płytki: " "* -> *Menedżer płytek...*
- w pole wyszukiwania wpisujemy *esp8266* i instalujemy pakiet ***esp8266 by ESP8266 Community***
- restartujemy *Arduino IDE*

- 4) Pobieramy i instalujemy sterowniki do wbudowanej przejściówki USB<->COM. W przypadku, gdy naszym modułem jest *NodeMCU LoLin V3* instalujemy sterowniki do układu [CH340/CH341](#).
- 5) Podłączamy do komputera moduł *NodeMCU* i sprawdzamy w systemie (*Menedżer urządzeń*) nr portu COM pod jakim jest dostępny nasz moduł, np. *USB-SERIAL CH340 (COM12)*
- 6) Pobieramy spakowany folder projektu *NodeMCU WiFi Data Logger* ze strony http://elektronika.5v.pl/e/datalogger/nodemcu_wifi_datalogger.zip, zapisujemy lokalnie na komputerze i rozpakowujemy, np. darmowym programem *7-Zip*.

- 7) Otwieramy plik projektu *nodemcu_wifi_datalogger.ino* w programie *Arduino IDE*.
- 8) Z menu górnego programu *Arduino IDE* otwieramy *Narzędzia* i ustawiamy parametry pracy naszego modułu *NodeMCU* oraz numer portu COM, pod którym nasz moduł jest dostępny:

```
Płytką: "NodeMCU 1.0 (ESP-12E Module)"
CPU Frequency: "80MHz"
Flash Size: "4M (3M SPIFFS)"
Upload Speed: "115200"
Port: "COM12"
```

- 9) Konfigurujemy odpowiednio ustawienia data loggera w nagłówku programu **:
 - pogrubioną czcionką oznaczono pozycje, które można zmieniać
 - objaśnienia do poszczególnych pozycji są podane w komentarzach po znakach //




```
//parametry ustawiane przez użytkownika
//#####
#define mySSID "NASZEWIFI" //nazwa domowej sieci WiFi
#define myPASS "naszehaslo" //hasło do domowej sieci WiFi (jeżeli sieć jest niezabezpieczona - "")

#define pvSERV "pvmonitor.pl" //adres serwera na który wysyłamy dane, np. pvmonitor.pl lub 37.187.66.191
#define pvID "XXX" //nazwa użytkownika - konto w serwisie http://pvmonitor.pl
#define pvPASS "xxxxyyzzz" //hasło do konta w serwisie http://pvmonitor.pl

#define WYSYLAJ_PRODUKCJE 1 //0 = pomiń, 1 = wysyłaj wartość produkcji energii na serwer
#define WYSYLAJ_ZUZYCIE 1 //0 = pomiń, 1 = wysyłaj wartość zużycia energii na serwer
#define WYSWIETL_SIECI 0 //0 = wyłącz, 1 = wyszukaj i wyświetl dostępne sieci WiFi (akcja wywoływana co 1 minutę)

#define wifiSTATIC 0 //0 = pozyskaj adres IP automatycznie z DHCP routera, 1 = ustaw stałe IP
IPAddress wifiIP(192, 168, 0, 44); //stały adres IP
IPAddress wifiMASK(255, 255, 255, 0); //maska podsieci
IPAddress wifiGATE(192, 168, 0, 1); //brama domyślna
IPAddress wifiDNS(192, 168, 0, 1); //adres serwera DNS (routera)
//#####
```

**UWAGA: Jeżeli licznik(i) produkcji energii znajdują się w innym miejscu niż licznik(i) zużycia energii, można zastosować dwa moduły *NodeMCU* i po odpowiednim skonfigurowaniu parametrów *WYSYLAJ_PRODUKCJE* oraz *WYSYLAJ_ZUZYCIE*, z jednego modułu wysyłać samą produkcję, a z drugiego samo zużycie.

- 10) Weryfikujemy poprawność programu klikając na przycisk .
- 11) Jeżeli weryfikacja przebiegła pomyślnie, wgrywamy program do modułu *NodeMCU* z poziomu *Arduino IDE* przyciskiem .
- 12) Otwieramy *Monitor szeregowy*  i obserwujemy wyświetlane komunikaty:

```
NodeMCU WiFi Data Logger - http://pvmonitor.pl
Created by Lukasz Janikowski http://elektronika.5v.pl
Connecting to NASZEWIFI.....
Connected - IP: 192.168.0.44 RSSI: -56 dBm

Sending data... OK P: 0,008 kWh / U: 0,012 kWh Date: Mon, 09 May 2016 17:29:08 GMT
Sending data... OK P: 0,017 kWh / U: 0,023 kWh Date: Mon, 09 May 2016 17:30:08 GMT
Sending data... OK P: 0,025 kWh / U: 0,034 kWh Date: Mon, 09 May 2016 17:31:08 GMT
```

- 13) Jeżeli wszystko wykonaliśmy poprawnie i komunikaty wyglądają podobnie, jak w powyższej ramce, możemy już się cieszyć własnym data loggerem.
- 14) Jeżeli coś nie działa – szukamy błędu sprawdzając wszystko jeszcze raz.

Dodatkowe opcje...

Osoby mające odwagę i nieco doświadczenia w programowaniu mikrokontrolerów w *Arduino* lub języku *C/C++* mogą pokusić się o modyfikację programu urządzenia według własnych potrzeb. Przy odrobinie chęci można m.in.:

- skorzystać z wbudowanego przetwornika ADC i mierzyć np. napięcie ogniw PV (wymagany zewnętrzny dzielnik napięcia), napięcie z czujnika temperatury *LM35* (napięcie przeliczamy na temperaturę),
- skorzystać z dodatkowych zewnętrznych przetworników ADC na magistralę I^2C , np. gotowy moduł [ADS1115](#) (możliwy 16-bitowy pomiar napięcia ogniw PV, pomiar prądu z wykorzystaniem zewnętrznego bocznika pomiarowego, pomiar temperatury za pomocą czujników *LM35*),
- mierzyć temperaturę wykorzystując cyfrowe czujniki *DS18B20*,
- wyświetlać parametry na wyświetlaczach LCD/OLED podłączonych na magistrali I^2C ,
- stworzyć własny mini [serwer HTTP](#), umożliwiający np. zmianę ustawień lub podgląd parametrów poprzez przeglądarkę internetową,
- programować układ przez WiFi bez fizycznego podłączenia z komputerem ([OTA](#)),
- wykorzystać inne gotowe moduły czujników lub układy stworzone dla *Arduino*.

Jak widać możliwości rozbudowy jest bardzo dużo.

Podsumowanie

Przedstawiony data logger był testowany w różnych konfiguracjach, przy różnych czasach trwania impulsów pomiarowych, różnych konfiguracjach sieci WiFi (m.in. różne zabezpieczenia, adres IP z DHCP, stały adres IP, połączenie internetowe przez łącze szerokopasmowe, łącze GSM), różne odległości modułu od routerów, praca kilkudniowa bez przerwy... i działał prawidłowo. Nie mogę jednak zagwarantować 100% poprawności działania w każdym możliwym przypadku.

Ewentualne resetowanie układu wykonujemy przyciskiem *RST* obok złącza USB modułu *NodeMCU*.

Moduł po zaprogramowaniu nie wymaga dodatkowych regulacji i powinien od razu działać.

UWAGA: Należy zwrócić uwagę na dostępność i siłę sygnału sieci WiFi w miejscu, w którym ma pracować data logger. Siłę sygnału sieci (RSSI) widzianą przez moduł można sprawdzić korzystając z programu typu terminal (wyświetlana jest po resecie lub po ponownym podłączeniu do sieci WiFi). Im wyższa wartość RSSI, tym lepszy sygnał WiFi (uwaga: wartości RSSI są liczbami ujemnymi). Odpowiednie ustawienie wbudowanej anteny modułu *ESP8266*, względem anteny routera, może wpływać na jakość połączenia. Nie należy umieszczać modułu w metalowej obudowie oraz zastaniać wbudowanej anteny PCB.

Niniejszy projekt powstał w celu rozwoju amatorskich instalacji OZE.

Nie wyrażam zgody na wykorzystanie projektu lub zastosowanie opisanych w tym opracowaniu rozwiązań w celach komercyjnych (zarobkowych).

Nie ponoszę odpowiedzialności za skutki nieprawidłowej pracy urządzenia. Próby wykonywania i uruchamiania opisanych elementów i urządzeń podejmujesz na własną odpowiedzialność.

Zezwalam na kopiowanie, publikowanie i wytwarzanie na użytek własny, jednakże w przypadku przedruku czy publikacji wymagana jest moja pisemna zgoda.

Lukasz Janikowski