

Mariusz **Duka**

KOMPUTER IOT ONION OMEGA2

Podręcznik użytkownika



Helion 

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz Helion SA dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz Helion SA nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Redaktor prowadzący: Małgorzata Kulik

Projekt okładki: Studio Gravite / Olsztyn
Obarek, Pokoński, Pazdrijowski, Zaprucki
Grafika na okładce została wykorzystana za zgodą Shutterstock.com

Helion SA
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: helion@helion.pl
WWW: <http://helion.pl> (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
<http://helion.pl/user/opinie/koiopo>
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Kody źródłowe wybranych przykładów dostępne są pod adresem:
<ftp://ftp.helion.pl/przyklady/koiopo.zip>

ISBN: 978-83-283-6629-9

Copyright © Mariusz Duka 2020

Printed in Poland.

- Kup książkę
- Poleć książkę
- Oceń książkę

- Księgarnia internetowa
- Lubię to! » Nasza społeczność

Spis treści

Przedmowa	9
Rozdział 1. Parametry techniczne	14
Moduły rozszerzające typu Dock	16
Moduły rozszerzające typu Expansion	23
Ceny	31
Rozdział 2. Uruchomienie i konfiguracja	32
Przygotowanie sprzętu	32
Nawiązanie komunikacji	33
Konfiguracja w przeglądarce internetowej	35
Konfiguracja w konsoli systemowej	37
Terminal SSH	40
Edytor kodu	41
Reset i restart	41
Przywrócenie do ustawień fabrycznych	43
Rozdział 3. GPIO	44
Interfejsy komunikacyjne	46
Zarządzanie pinami GPIO	46
Grupy i funkcje pinów	50
Dioda Omega2 LED	51
GPIO Tool	52
GPIO w języku Python	55
Rozdział 4. Rozszerzenie pamięci masowej	57
Karta microSD	58
Pamięć USB (pendrive)	61
microSD/USB i partycja swap	63
Rozdział 5. Oprogramowanie	70
Firmware	70
OPKG	70
Git	73
Python	74

Node.js	76
PHP 7	78
C i C++	82
JamVM — wirtualna maszyna Javy	85
Perl	87
Erlang	89
Ruby	90
Rozdział 6. OLED, wyświetlanie tekstu i grafiki	92
OLED w konsoli graficznej	92
OLED w konsoli systemowej	93
OLED w języku C, Python i PHP	97
Rozdział 7. ADC, przetwornik analogowo-cyfrowy	100
ADC w konsoli graficznej	101
ADC w konsoli systemowej	101
ADC w praktyce	103
ADC w języku Python	105
Rozdział 8. RELAY, przełączanie napięcia	107
RELAY w konsoli graficznej	108
RELAY w konsoli systemowej	109
RELAY w praktyce	110
RELAY w językach C, Python i PHP	112
Rozdział 9. PWM, zasilanie i kontrola pracy urządzeń	116
PWM w konsoli graficznej	117
PWM w konsoli systemowej	118
PWM w praktyce	119
PWM w języku C, Python i PHP	120
Rozdział 10. GPS, system nawigacji satelitarnej	125
GPS w konsoli systemowej	125
GPS i OLED w języku PHP	127
Rozdział 11. Ethernet, sieć przewodowa, ruter i most	131
Funkcje sieciowe	132
Rozdział 12. RFID/NFC, identyfikacja bezprzewodowa	140
RFID/NFC w konsoli graficznej	140
RFID/NFC w konsoli systemowej	141
RFID/NFC, OLED i RELAY w języku Python	146

Rozdział 13. BLE, komunikacja Bluetooth	149
Parowanie urządzeń	149
Test połączenia	152
Problemy z połączeniem	152
Rozdział 14. Arduino, czyli AVR na pokładzie	153
Instalacja i konfiguracja oprogramowania	154
Pierwszy program dla Arduino	156
Wgrywanie programu do mikrokontrolera	157
Praca w konsoli systemowej	158
Problemy z komunikacją	159
Rozdział 15. SAMBA, serwer plików	160
Instalacja	160
Konfiguracja	160
Rozdział 16. SSL, szyfrowanie połączenia WWW	163
Instalacja OpenSSL	163
Generowanie certyfikatów SSL	163
Rozdział 17. Blynk, zdalne zarządzanie IoT	166
Smartfon lub tablet	166
Minikomputer Omega2	167
Rozdział 18. WEBCAM, zdalny monitoring	170
WEBCAM w konsoli graficznej	170
WEBCAM w konsoli systemowej	171
WEBCAM w praktyce	172
Rozdział 19. XMPP, serwer komunikacyjny Jabber	175
Instalacja Erlang	175
Instalacja serwera XMPP ejabberd	176
Instalacja klienta XMPP mcabber	176
Rozdział 20. Technika Cross-Compiling	178
Przygotowanie i kompilacja systemu	178
Kompilacja kodu dla minikomputera	179
Cross-Compiling w praktyce	181
Serwer Minecraft na minikomputerze Omega2	181
Zakończenie	184
Skorowidz	185

Rozdział 12. RFID/NFC, identyfikacja bezprzewodowa

RFID (ang. *Radio-frequency identification*) i NFC (ang. *Near Field Communication*) to technologie wykorzystujące fale radiowe do bezprzewodowego i bezdotykowego transferu danych cyfrowych na niewielkie odległości. Transfer danych, w odróżnieniu od technologii Bluetooth, nie wymaga dodatkowego parowania urządzeń, natomiast bezpieczeństwo transmisji opiera się na bliskiej odległości urządzeń i ich „otagowaniu”, czyli zapisaniu w układzie elektronicznym informacji identyfikacyjnych. Technologia NFC zaimplementowana jest w większości nowoczesnych smartfonów, za pomocą których możliwe jest realizowanie płatności zbliżeniowo. Ciekawym przykładem wykorzystania technologii NFC jest system kart *Mifare*, który w założeniu ma zastąpić tradycyjne bilety papierowe (*Mifare Ultralight*), karty magnetyczne oraz monety.

Moduł rozszerzający *RFID & NFC Expansion* wyposażony jest w popularny układ PN532, umożliwiający komunikację na częstotliwości 13,56 MHz w odległości do 6 cm oraz 30-pinowe złącze pozwalające na montaż dodatkowych modułów typu *Expansion*. Moduł obsługuje popularny standard kart *Mifare*, a także *Innovision Jewel* i *FeliCa*. Do komunikacji wykorzystywany jest interfejs szeregowy UART. W zestawie dołączone są dwie naklejki NFC, na których można zapisać do 100 bajtów danych.

RFID/NFC w konsoli graficznej

Odczytanie identyfikatora (*UID*) kart lub naklejek NFC umożliwia narzędzie RFID Reader (rysunek 12.1), które dostępne jest w aplikacji OnionOS. W prawym górnym rogu znajdziesz przycisk *Start Scanning*, którym uruchomisz procedurę skanowania, natomiast przyciskiem *Clear List* wyczyścisz dotychczasową listę. Wystarczy, że przyłożysz do czytnika oryginalną nalepkę NFC, która wchodzi w skład wyposażenia modułu, a na ekranie powinien pojawić się jej identyfikator.



Rysunek 12.1. Odczyt identyfikatora UID kart lub naklejek NFC w narzędziu RFID Reader

Narzędzie RFID Reader, pomimo tego, że nadaje się tylko do odczytywania identyfikatorów (*UID*), umożliwia sprawne i szybkie przetestowanie wielu kart i nalepek NFC. Zapis informacji na kartach wymaga wykorzystania narzędzi dostępnych w konsoli systemowej.

RFID/NFC w konsoli systemowej

Zapis danych w pamięci karty *Mifare Ultralight* (na nalepce NFC dołączonej do rozszerzenia *RFID & NFC Expansion*) lub informacji w formacie NDEF (ang. *NFS Data Exchange Format*) na kartach zblizeniowych typu *Mifare Classic* umożliwiają narzędzia takie jak `nfc-mfclassic` i `mifare-classic-write-ndef` dostępne w konsoli systemowej. Procedura zapisu danych na kartach NFC nie jest skomplikowana, jednak wymaga podstawowej umiejętności pracy w konsoli.

Jeśli wcześniej korzystałeś z narzędzia RFID Reader, część niezbędnego oprogramowania została już zainstalowana na minikomputerze Omega2. Sprawdź, czy w narzędziu RFID Reader aktywna jest procedura skanowania, jeśli tak, to musisz ją teraz wyłączyć, klikając przycisk *Stop Scanning*.

Zainstaluj najnowsze oprogramowanie do obsługi NFC:

```
opkg update
opkg install nfc-exp nfc-utils
```

Na początek sprawdź, czy moduł rozszerzający *RFID & NFC Expansion* jest widoczny w systemie:

```
nfc-scan-device
```

Lista aktywnych modułów NFC w systemie przedstawiona jest na rysunku 12.2.


```

root@Omega-0FB4: ~
root@Omega-0FB4:~# nfc-scan-device
nfc-scan-device uses libnfc v0.2.2
1 NFC device(s) found:
- Omega NFC Expansion:
  pn532_uart:/dev/ttyS1
root@Omega-0FB4:~#

```

Rysunek 12.2. Skanowanie urządzeń NFC podłączonych do minikomputera Omega2

Teraz możesz odczytać informacje zakodowane w układzie elektronicznym na nalepce NFC dostarczonej w komplecie z rozszerzeniem. Przyłóż nalepkę do czytnika RFID/NFC i wykonaj polecenie:

```
nfc-list
```

Na konsoli zostaną wydrukowane informacje identyfikacyjne układu NFC (rysunek 12.3).

```

root@Omega-0FB4: ~
root@Omega-0FB4:~# nfc-list
nfc-list uses libnfc v0.2.2
NFC device: Omega NFC Expansion opened
1 ISO14443A passive target(s) found:
ISO/IEC 14443A (106 kbps) target:
  ATQA (SENS_RES): 00 44
  UID (NFCID1): 04 16 7a 32 ed 4c 80
  SAK (SEL_RES): 00
root@Omega-0FB4:~#

```

Rysunek 12.3. Identyfikacja karty (nalepki) NFC za pomocą narzędzia nfc-list w konsoli systemowej

Każda karta NFC posiada swój unikatowy identyfikator (ang. *Unique Identification Number*) oraz zakodowaną informację o jej typie i producencie (ATQA — ang. *Answer To Request* i SAK — ang. *Select Acknowledge*). W tabeli 12.1 przedstawione są ATQA i SAK dla konkretnych typów kart NFC.

Tabela 12.1. Dane ATQA i SAK dla kart zbliżeniowych typu Mifare

Typ karty	ATQA	SAK	Długość UID
<i>Mifare Classic 1K</i>	00 04	08	4 bajty
<i>Mifare Classic 4K</i>	00 02	18	4 bajty
<i>Mifare Ultralight</i>	00 44	00	7 bajtów

Jak można zauważyć, dostarczone w komplecie nalepki NFC są w standardzie *Mifare Ultralight* i właśnie na nich spróbujemy w następnych krokach odczytać i zapisać dane.

Podstawowe narzędzia do obsługi kart zbliżeniowych NFC dostępne na minikomputerze Omega2 to *nfc-mfultralight* dla kart *Mifare Ultralight* i *nfc-mfclassic* dla kart *Mifare Classic*.

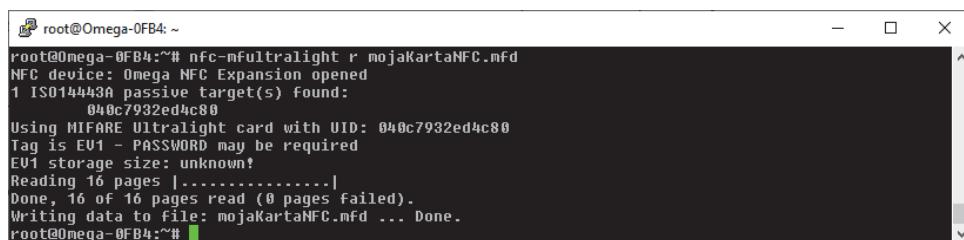
Odczyt danych z kart Mifare Ultralight

Karty *Mifare Ultralight* ze względu na swoją kompaktową budowę (forma cienkiej nalepki) bardzo często są wykorzystywane w systemach do identyfikacji przedmiotów. Karta ta posiada wbudowaną pamięć typu EEPROM, na której można zapisać 48 bajtów danych.

Przyłóż kartę (nalepkę) NFC do czytnika RFID/NFC i wykonaj polecenie:

```
nfc-mfultralight r mojaKartaNFC.mfd
```

Odczyt danych z karty *Mifare Ultralight* (nalepki NFC) w konsoli systemowej przedstawiony jest na rysunku 12.4.



```
root@Omega-0FB4: ~
root@Omega-0FB4:~# nfc-mfultralight r mojaKartaNFC.mfd
NFC device: Omega NFC Expansion opened
1 ISO14443A passive target(s) found:
    040c7932ed4c80
Using MIFARE Ultralight card with UID: 040c7932ed4c80
Tag is EV1 - PASSWORD may be required
EV1 storage size: unknown!
Reading 16 pages |.....|
Done, 16 of 16 pages read (0 pages failed).
Writing data to file: mojaKartaNFC.mfd ... Done.
root@Omega-0FB4:~#
```

Rysunek 12.4. Odczyt danych z karty Mifare Ultralight (nalepki NFC) w konsoli systemowej

W pliku *mojaKartaNFC.mfd* zostały zapisane dane w formacie MFD pochodzące z karty zbliżeniowej NFC. Dla wygody skorzystaj z narzędzia `xxd` do wyświetlenia zawartości pliku w przyjaznej formie.

Wykonaj polecenie:

```
xxd mojaKartaNFC.mfd
```

Wyświetlona zostanie zawartość pliku w systemie heksadecymalnym (szesnastkowym):

```
00000000: 0416 7ae0 32ed 4c80 1348 0000 e110 1200  ..z.2.L..H.....
00000010: 0000 0000 0000 0000 0000 0000 0000 0000  .....
00000020: 0000 0000 0000 0000 0000 0000 0000 0000  .....
00000030: 0000 0000 0000 0000 0000 0000 0000 0000  .....
```

Pierwsza linia (pierwsze 16 bajtów) zarezerwowana jest m.in. na identyfikator i informację o aktywnym trybie „tylko do odczytu”, natomiast trzy kolejne linie (łącznie 48 bajtów) mogą być zapisane dowolną informacją. Z powyższego zapisu wynika, że na karcie (nalepce) NFC zapisane są wartości zerowe, co oznaczają, że karta jest pusta.

Modyfikacja danych i zapis do formatu MFD

Informacje pobrane z karty zbliżeniowej NFC mogą zostać zmienione i ponownie na niej zapisane. Dla ułatwienia edycji danych wykorzystaj ponownie narzędzie `xxd`. Wykonaj polecenie:

```
xxd mojaKartaNFC.mfd > mojaKartaNFC.hex
```

Otwórz w edytorze tekstowym `vi` plik zapisany w systemie szesnastkowym:

```
vi mojaKartaNFC.hex
```

i wstaw nową zawartość w liniach `0000010` i `0000020`:

```
0000000: 0416 7ae0 32ed 4c80 1348 0000 e110 1200  ..z.2.L..H.....
0000010: 4e46 4320 6e61 206d 696e 696b 6f6d 7075
0000020: 7465 727a 6520 4f6d 6567 6132 2100 0000
0000030: 0000 0000 0000 0000 0000 0000 0000 0000  .....
```

Ostatnim krokiem jest przekonwertowanie danych z formatu szesnastkowego na format MFD:

```
xxd -r mojaKartaNFC.hex mojaKartaNFC.mfd
```

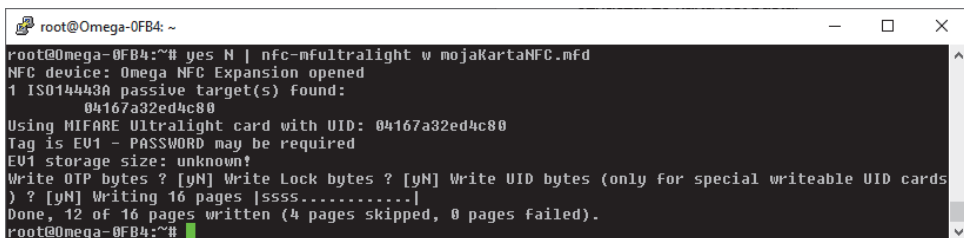
Zapis danych na karcie Mifare Ultralight

Wcześniej przygotowany plik w formacie MFD można zapisać w pamięci karty.

Przyłóż kartę (nalepkę) NFC do czytnika RFID/NFC i wykonaj polecenie:

```
yes N | nfc-mfultralight w mojaKartaNFC.mfd
```

Zapis danych na karcie zbliżeniowej (nalepce) NFC w konsoli systemowej przedstawiony jest na rysunku 12.5.



```
root@Omega-0FB4: ~
root@Omega-0FB4:~# yes N | nfc-mfultralight w mojaKartaNFC.mfd
NFC device: Omega NFC Expansion opened
1 IS014443A passive target(s) Found:
  04167a32ed4c80
Using MIFARE Ultralight card with UID: 04167a32ed4c80
Tag is EV1 - PASSWORD may be required
EV1 storage size: unknown!
Write OTP bytes ? [yN] Write Lock bytes ? [yN] Write UID bytes (only for special writeable UID cards
) ? [yN] Writing 16 pages [ssss.....]
Done, 12 of 16 pages written (4 pages skipped, 0 pages failed).
root@Omega-0FB4:~#
```

Rysunek 12.5. Zapis danych na karcie zbliżeniowej (nalepce) NFC w konsoli systemowej

Zapis na karcie (nalepce) NFC został pomyślnie wykonany. Zwróć uwagę, że podczas zapisu system udzielił automatycznych odpowiedzi (przez polecenie `yes N`), co znacznie przyspieszyło proces.

Dla pewności sprawdź, czy zapis na karcie został wykonany prawidłowo.

Przyłóż kartę (nalepkę) NFC do czytnika RFID/NFC i wykonaj polecenie:

```
nfc-mfultralight r mojaKartaNFC.mfd
```

Zdekoduj informację poleceniem:

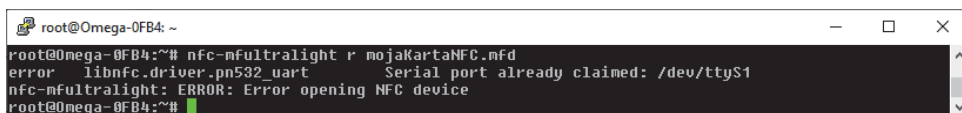
```
xxd mojaKartaNFC.mfd
```

Powinna zostać wyświetlona następująca informacja:

```
00000000: 0416 7ae0 32ed 4c80 1348 0000 e110 1200  ..z.2.L..H.....
00000010: 4e46 4320 6e61 206d 696e 696b 6f6d 7075  NFC na minikompu
00000020: 7465 727a 6520 4f6d 6567 6132 2100 0000  terze Omega2!...
00000030: 0000 0000 0000 0000 0000 0000 0000 0000  .....
```

Problem z dostępem do modułu RFID/NFC

Podczas korzystania z rozszerzenia *RFID & NFC Expansion* może się zdarzyć, że wykonanie polecenia odczytu lub zapisu nie zakończy się prawidłowo, co w konsekwencji uniemożliwi dalszą pracę. Ponowne wykonanie jakiegokolwiek polecenia spowoduje wyświetlenie komunikatu o wystąpieniu problemu (rysunek 12.6).



```
root@Omega-0FB4: ~
root@Omega-0FB4:~# nfc-mfultralight r mojaKartaNFC.mfd
error libnfc.driver.pn532_uart Serial port already claimed: /dev/ttyS1
nfc-mfultralight: ERROR: Error opening NFC device
root@Omega-0FB4:~#
```

Rysunek 12.6. Komunikat o braku dostępu do rozszerzenia RFID & NFC Expansion

Przyczyną wystąpienia problemu może być uruchomiony już proces skanowania w narzędziu RFID Reader, który zajmuje port szeregowy `/dev/ttyS1`.

W sytuacji, kiedy pomimo wyłączenia skanowania nadal nie jest możliwy dostęp do modułu *RFID & NFC Expansion*, wykonaj następujące polecenia:

```
opkg update
opkg install screen
```

Uruchom narzędzie screen poleceniem:

```
screen /dev/ttyS1
```

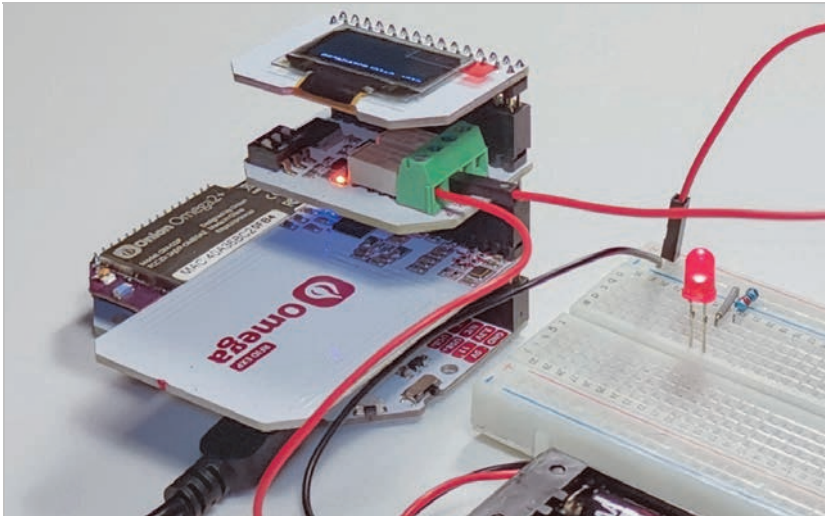
a następnie zamknij je kombinacją klawiszy `Ctrl+a+k`.

Aktywny proces, który blokuje dostęp do czytnika RFID/NFC, powinien zostać zatrzymany.

RFID/NFC, OLED i RELAY w języku Python

Ciekawym pomysłem wykorzystania rozszerzenia *RFID & NFC Expansion* może być układ elektryczny przełączający napięcie w obwodzie zaufaną kartą zbliżeniową.

Zmontuj układ przedstawiony w rozdziale 8. „RELAY, przełączanie napięcia”, w podrozdziale „RELAY w praktyce”, a także dodatkowo wepnij do 30-pinowego złącza kolejno moduły *RFID & NFC Expansion* i *OLED Expansion*. Zmontowany układ z *RFID & NFC Expansion*, *Relay Expansion*, *OLED Expansion* i diodą LED przedstawiony jest na rysunku 12.7.



Rysunek 12.7. Układ z *RFID & NFC Expansion*, *Relay Expansion*, *OLED Expansion* i diodą LED

Elementy układu:

- ◆ minikomputer Omega2 oraz moduł rozszerzający, np. *Expansion Dock*,
- ◆ moduł rozszerzający *RFID & NFC Expansion*,
- ◆ moduł rozszerzający *Relay Expansion*,
- ◆ moduł rozszerzający *OLED Expansion*,
- ◆ płytki stykowej,
- ◆ dioda LED,
- ◆ rezystor 220 Ω ,
- ◆ zasilanie 3 V (2 baterie 1,5 V AA lub AAA),
- ◆ przewody połączeniowe męsko-męskie.

Odczytaj unikatowy numer UID karty (nalepki) NFC, ponieważ wykorzystasz go w programie obsługującym zmontowany układ. Możesz użyć narzędzia RFID Reader, dostępnego w aplikacji OnionOS.

Zainstaluj oprogramowanie wraz z bibliotekami programistycznymi:

```
opkg update
opkg install python3-light
opkg install python3-onion-i2c python3-oled-exp python3-relay-exp
opkg install nfc-exp nfc-utils
```

Utwórz w edytorze tekstowym vi plik *nfc.py*:

```
vi /root/nfc.py
```

i przepisz poniższy kod w języku Python (listing 12.1).

Listing 12.1

Plik *nfc.py*

```
# import modułów RELAY i OLED
from OmegaExpansion import oledExp
from OmegaExpansion import relayExp
import subprocess, time

addrRelay = 7      # adres I2C urządzenia od 0 (0x20) do 7 (0x27)
channelRelay = 0 # numer przekaźnika 0 lub 1
acceptedUids = ['04167a32ed4c80'] # zaufane UID identyfikatorów

# inicjacja modułów rozszerzających OLED i RELAY
def initial_setup():
    status_oled = oledExp.driverInit()
    status_relay = relayExp.driverInit(addrRelay)
    check = relayExp.readChannel(addrRelay, channelRelay)
    if check == 1:
        relay_off()
    return

# włącz przekaźnik
def relay_on():
    relayExp.setChannel(addrRelay, channelRelay, 1) # 1 - włączenie
    return

# wyłącz przekaźnik
def relay_off():
    relayExp.setChannel(addrRelay, channelRelay, 0) # 0 - wyłączenie
    return

# włącz przekaźnik, a następnie po 5 sekundach wyłącz
# wyświetl UID karty (nalepki) NFC na ekranie OLED
def access(uid):
    oledExp.write("UID: " + uid)
    relay_on()
    time.sleep(5)
    relay_off()
    oledExp.clear()
    return
```

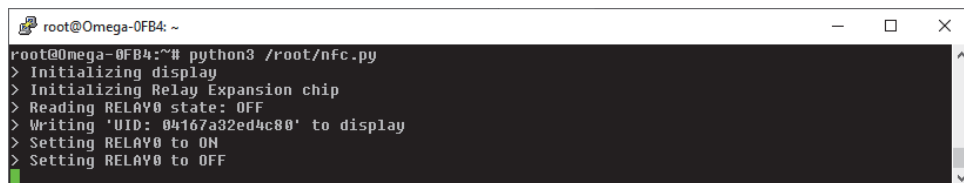
```
def __main__():
    initial_setup()
    while 1:
        # odczytaj UID karty i wstaw do zmiennej cmd
        cmd = "nfc-list | grep UID | sed -e 's/ //g' -e 's/^.*://'"
        uid = subprocess.check_output(cmd, shell=True)
        uid = uid.decode('utf-8').rstrip('\n')
        # sprawdź, czy UID jest zaufany
        for acceptedUid in acceptedUids:
            if(acceptedUid == uid):
                access(uid)

if __name__ == '__main__':
    __main__()
```

Uruchom kod poleceniem:

```
python3 /root/nfc.py
```

Po uruchomieniu się programu przyłóż zaufaną kartę (nalepkę) NFC do czytnika RFID/NFC. Dioda LED powinna się zaświecić i po 5 sekundach zgasnąć. Dodatkowo numer UID zostanie wyświetlony na ekranie OLED. W trakcie pracy programu na konsoli systemowej będą drukowane komunikaty o stanie przekaźników (rysunek 12.8). Zamiast diody możesz wykorzystać elektryczny zamek do drzwi i w ten sposób sterować dostępem do pomieszczeń w domu lub w biurze.



```
root@Omega-0FB4: ~
root@Omega-0FB4:~# python3 /root/nfc.py
> Initializing display
> Initializing Relay Expansion chip
> Reading RELAY0 state: OFF
> Writing 'UID: 04167a32ed4c00' to display
> Setting RELAY0 to ON
> Setting RELAY0 to OFF
```

Rysunek 12.8. Sterowanie obwodem elektrycznym za pomocą karty (nalepki) NFC

Skorowidz

A

ADC, 31, 37, 75
 Expansion, 100
 cena, 31
 czujnik natężenia światła, 103
 czujnik temperatury, 103
 parametry techniczne, 24
 schemat modułu, 100
 w języku Python, 105
 w konsoli graficznej, 101
 w konsoli systemowej, 101

adres
 I2C, 108
 IP, 34
 WWW, 34

aktualizacja
 firmware’u, 38, 71
 strefy czasowej, 40

akumulator LiPo, 20

aplikacja
 Blynk, 166
 Code Editor, 37
 Legacy Console, 37, 43, 52
 NFC-RFID Expansion, 37
 OnionOS, 19, 36
 Power Dock 2, 37
 PuTTY, 37
 Sensor Monitor, 37
 Sparkfun, 37

Terminal SSH, 40
 Timelapse Camera, 37

Arduino, 153
 pierwszy program, 156

Arduino Dock R2, 21–23, 32, 75, 153, 154
 cena, 31
 parametry techniczne, 21

Arduino IDE, 21
 instalacja, 154
 konfiguracja, 154
 połączenie z minikomputerem, 154
 w konsoli systemowej, 158
 wgrywanie programu, 157, 159

Arduino Uno, 154
 Arduino Uno R3, 21, 153
 ATmega328P, 21, 22
 AVR, 153
 avrdude, 159

B

biblioteka Onion Omega2, 78
 biblioteki PHP, 78
 BLE Expansion, 29, 149
Patrz także Bluetooth
 cena, 31
 konfiguracja, 149, 150

parametry techniczne, 30

Bluetooth, 15, 28–30
 parowanie urządzeń, 149
 problemy z połączeniem, 152
 test połączenia, 152

Blynk, 37, 166
 instalacja
 oprogramowania, 167
 status połączenia, 168
 tworzenie projektu, 167
 zarządzanie IoT, 166
 zmiana stanu pinu, 169

Breadboard Dock, 21, 32, 44
 cena, 31

C

C/C++, 71, 82
 kompilacja kodu, 82, 83

certyfi­kat SSL, 163, 165

CGI, Common Gateway Interface, 88

Code Editor, 37, 41, 42

cron, 173

Cross-Compiling, 178, 181

CSR, Certificate Signing Request, 164

częstotliwość, frequency, 116

czujnik

- natężenia światła, 103
- temperatury, 103

D

dane

- ATQA, 142
- SAK, 142

DHCP, Dynamic Host

- Configuration Protocol, 134

dioda

- LED RGB, 17
- Omega2 LED, 51, 52

Dock, 149, 170

Docker, 178

dostęp

- do minikomputera, 33
- do modułu RFID/NFC, 145
- do plików, 161

E

edytor kodu, 41

- vi, 62

ejabberd, 175

ekran OLED, 93

Erlang, 10, 89, 90

- instalacja, 175

Ethernet, 131

Ethernet Expansion, 27, 131

- cena, 31
- parametry techniczne, 28

Expansion Dock, 16, 23, 31, 54

- cena, 31
- parametry techniczne, 17

F

fast-gpio, 49

- opcje narzędzia, 48

firmware, 38, 70

folder sieciowy, 162

format

- HEX, 158
- MFD, 143, 144
- NDEF, 141

formatowanie partycji, 67

funkcje

- pinów, 51
- sieciowe, 132

G

generowanie

- certyfikatu SSL, 165
- pliku CSR, 164

Git, 73, 181

gniazdo JST-PH, 33

GPIO, General-Purpose

- Input/Output, 14–17, 30, 44–48, 55, 75, 81, 153
- piny, 46

- Tool, 52–54

gpioctl, 47

- opcje narzędzia, 47

GPS, Global Positioning

- System, 15, 125

- antena, 130

- Expansion, 11, 27, 125

- cena, 31
- parametry techniczne, 27

- rozszerzenie Arduino

- Dock 2, 128

- w języku PHP, 127

- w konsoli systemowej, 125

grupy pinów, 50

H

hasło, 35

II²C, Inter-Integrated

- Circuit,, 15, 21–26, 45, 75, 108

I²S, Inter-IC Sound, 15, 45

ICSP, 22

identyfikacja

- bezprowodowa, 140
- karty NFC, 142

identyfikator

- UID, 141
- UUID, 60, 63

informacje

- o karcie microSD, 64–67
- o pamięci masowej, 61, 64
- o pamięci USB, 62
- o stanie pinów GPIO, 56

instalacja

- Arduino IDE, 154
- biblioteki, 79
- Erlang, 175
- klienta mcabber, 176
- konsoli, 40
- Minecraft, 181
- modułu, 77
- Node.js, 76
- OpenSSL, 163
- oprogramowania Blynk, 167
- pakietu, 75, 77
- PHP, 78
- Samby, 160

instalator, 35

interfejs

- eth0, 137
- lan, 134

- sieciowy eth0, 131
- SPI, 75
- szeregowy UART, 29
- wan, 134
- interfejsy komunikacyjne, 46
- IoT, internet rzeczy, 9
 - zdalne zarządzanie, 166

J

- JamVM, Java Virtual Machine, 86
- język
 - C, 82
 - C++, 82
 - Erlang, 89
 - Perl, 87
 - Python, 74
 - Ruby, 90
- JST-PH, 18, 19, 33

K

- karta
 - microSD, 57, 58, 64
 - Mifare, 140
 - Classic, 141
 - dane ATQA, 142
 - dane SAK, 142
 - Ultralight, 141–144
- klient, 132
 - konfiguracja
 - interfejsów, 132
 - mcabber, 176, 177
 - udostępnianie sieci, 133
- klucz prywatny, 164
- kompilacja kodu
 - w C, 82
 - w C++, 83
 - w Javie, 86

- konfiguracja
 - Arduino IDE, 154
 - DHCP, 135
 - interfejsu sieciowego, 137
 - karty sieciowej, 134
 - narzędzia mjpg-streamer, 172
 - obsługi PHP 7, 80
 - partycji, 59
 - połączenia
 - mostkowego, 138
 - rutingu, 137
 - Samby, 160
 - serwera Minecraft, 182
 - sieci bezprzewodowej, 38, 136, 138
 - systemu, 68
 - szyfrowania, 136
 - w konsoli systemowej, 37
 - w przeglądarce, 35
- konsola graficzna
 - ADC, 101
 - OLED, 92
 - PWM, 117
 - RELAY, 108
 - RFID/NFC, 140
 - WEBCAM, 170
- konsola systemowa, 43
 - ADC, 101
 - Arduino IDE, 158
 - GPS, 125
 - OLED, 93
 - PWM, 118
 - RELAY, 109
 - RFID/NFC, 141
 - WEBCAM, 171
- kontrola pracy urządzeń, 116

L

- LAN, Local Area Network, 131
- LED RGB, 17, 32
- LEDE, Linux Embedded Development Environment, 70–73, 85–90
 - dołączenie
 - repozytoriów, 72
- Legacy Console, 37, 40–43, 52
- login, 35

M

- magistrala I2C, 26, 75
- mcabber
 - instalacja klienta, 176
- menedżer pakietów pip, 74
- microSD, 58
 - informacje o karcie, 64–67
 - montowanie, 59
 - partycja swap, 66
 - pliki systemowe, 68
- microSD/USB, 63
- Minecraft
 - instalacja serwera, 181
 - konfiguracja serwera, 182
 - uruchomienie serwera, 183
- Mini Dock, 20, 21, 32
 - cena, 31
 - parametry techniczne, 20, 21
- MIPS, Microprocessor without Interlocked Piped Stages, 14, 15, 73
- mjpg-streamer, 171

moduł

- ADC Expansion, 100
- Arduino Dock R2, 153
- BLE Expansion, 149
- Breadboard Dock, 44
- Ethernet Expansion, 131
- GPS Expansion, 125
- node-oled-exp, 77
- node-pwm-exp, 77
- node-relay-exp, 77
- Relay Expansion, 107
- RFID & NFC Expansion, 140

moduły rozszerzające, 23

monitoring, 170

montaż minikomputera, 32

most, bridge, 131, 138

konfiguracja

połączenia

mostkowego, 138

sieci bezprzewodowej, 138

N

narzędzie

- adc-exp, 101, 102
- Arduino IDE, 155, 156
- avrdude, 159
- bluetoothctl, 150
- Docker, 178
- Code Editor, 41, 42
- fast-gpio, 48, 49
- fswebcam, 170–174
- GPIO Tool, 52–54
- gpioctl, 47
- hcitool, 152
- mjpg-streamer, 170–173
- OLED Control, 92
- oled-exp, 93–97
- OpenSSL, 163

opkg, 70

PWM Control, 117

pwm-exp, 49, 118

Relay Control, 108

relay-exp, 109, 110

RFID Reader, 140, 141

Sensor Monitor, 102

ubus, 126

uvcdynctrl, 171

Webcam, 170

xxd, 144

nawiązanie połączenia, 34

nawigacja satelitarna,

Patrz GPS, 125

NDEF, NFS Data Exchange

Format, 141

NFC, Near Field

Communication, 15, 28,

140

format MFD, 143, 144

identyfikacja karty, 142

modyfikacja danych,

144

narzędzie xxd, 144

odczyt danych, 143

odczyt identyfikatora

UID, 141

skanowanie urządzeń,

142

sterowanie obwodem

elektrycznym, 148

zapis danych, 144

Node.js, 76, 77

instalacja, 76

moduły, 77

O

obsługa

Arduino Dock R2, 154,

155

GPIO, 81

interfejsu eth0, 137

języka Perl, 88

kamery internetowej, 170

NFC, 141

OLED Expansion, 182

OLED, Organic Light-

Emitting Diode, 10, 23,

72–79, 85, 92

Control, 92

Expansion, 23, 72, 76,

79, 85, 92

cena, 31

instalacja biblioteki,

72

obsługa, 182

oprogramowanie, 72

parametry techniczne,

23

wtyczka, 182

w języku

C, 97

PHP, 97, 127

Python, 97, 146

w konsoli graficznej, 92

w konsoli systemowej,

93

zarządzanie ekranem,

92

oled-exp, 84

Omega2, 14

LED, 51, 52

LTE, 14, 31, 184

Pro, 14, 31, 76, 184

Onion

Cloud, 37

Corporation, 14

OnionOS, 19, 36, 40

opcje narzędzia

adc-exp, 102

oled-exp, 95

pwm-exp, 118

relay-exp, 109

- OpenSSL
 - instalacja, 163
 - OpenWRT/LEDE, 70
 - oprogramowanie
 - sprzętowe, firmware, 38, 70
- P**
- pakiet
 - oled-qr-code-generator, 76
 - onion-omega-oled, 77
 - pyOledExp, 74
 - pyOmegaArduinoDock, 75
 - pyOnionGpio, 75
 - pyOnionI2C, 75
 - pyOnionSpi, 75
 - pyPwmExp, 74
 - pyRelayExp, 74
 - python3-adc-exp, 75
 - python3-oled-exp, 75
 - python3-onion-i2c, 75
 - python3-pwm-exp, 75
 - python3-relay-exp, 75
 - python-adc-exp, 75
 - pakiety
 - instalacja, 75, 76
 - LEDE, 72
 - w języku Python 2.7, 74
 - w języku Python 3, 75
 - pamięć
 - flash, 58, 61
 - masowa, 57
 - montowanie, 61
 - parametry techniczne
 - ADC Expansion, 24
 - Arduino Dock R2, 21
 - Breadboard, 21
 - Ethernet Expansion, 28
 - Expansion Dock, 16
 - GPS Expansion, 27
 - Mini Dock, 20
 - OLED Expansion, 23
 - Power Dock 2, 18
 - Relay Expansion, 25
 - RFID & NFC Expansion, 29
 - Servo Expansion, 26
 - partycje, 60
 - formatowanie, 67
 - tworzenie, 65
 - typu swap, 58, 63, 66
 - pendrive, 61
 - Perl, 87-89
 - konfiguracja obsługi języka, 88
 - PHP, 14, 78—81
 - PHP 7, 78
 - biblioteka Onion Omega2, 78
 - instalacja, 78
 - obsługa GPIO, 81
 - obsługa języka, 80
 - piny, 44
 - GPIO, 45
 - grupowanie, 50
 - zmienianie funkcji, 51
 - informacje o stanie, 56
 - narzędzie
 - fast-gpio, 48
 - gpioctl, 47
 - plik
 - adc.py, 105
 - blynk.js, 167
 - fstab, 59, 62
 - gps.php, 128
 - nfc.py, 147
 - oled.c, 84
 - oled.sh, 95
 - pwm.c, 120
 - pwm.php, 123
 - relay.c, 112
 - relay.php, 114
 - relay.py, 113
 - pliki CSR, 164
 - podłączenie
 - czujnika temperatury, 103
 - diody LED, 54
 - obwodu elektrycznego, 111
 - serwomechanizmu, 120
 - polecenie
 - bluetoothctl, 150
 - d, 65
 - df -h, 58, 61, 63
 - discoverable on, 150
 - F, 65
 - fdisk, 67
 - firstboot -y, 43
 - free, 65
 - hciconfig -a, 149
 - ipconfig, 134
 - make, 179
 - n, 65
 - opkg update, 40
 - OPKG, 70
 - ouprgrade, 38
 - ouprgrade, 70
 - pairable on, 150
 - reboot, 43
 - scan off, 151
 - scan on, 150
 - sync, 43
 - t, 66
 - w, 67
 - wifisetup, 38
 - połączenie, 34
 - mostkowe, 139
 - port JST-PH, 19

Power Dock 2, 18, 23, 32, 37
 cena, 31
 parametry techniczne,
 18
 stan akumulatora, 19

Proto Expansion, 30
 cena, 31

protokół
 DHCP, 134
 UART, 44

przełączanie napięcia, 107

przetwornik analogowo-
 cyfrowy, 100

przycisk Reset, 43

punkt dostępowy Wi-Fi,
 137

PuTTY, 37

PWM, Pulse-Width
 Modulation, 11, 15, 23,
 45–49, 74–77, 116, 119
 Control, 117
 podłączenie
 serwomechanizmu,
 120

PWM26, 46, 116
 w języku
 C, 120
 PHP, 123
 Python, 122
 w konsoli systemowej,
 118

pwm.py, 122

pwm-exp, 49, 77

Python, 14, 55, 74–76, 84

Python 2.7, 74

Python 3, 75

Q

QR Code, 76

Qwiic Expansion, 184

R

RELAY, 74–77, 107, 110
 adres I2C, 108
 cena, 31
 Control, 108
 Expansion, 11, 25, 75,
 107
 obwód elektryczny, 111
 parametry techniczne,
 25
 schemat modułu, 107
 w języku
 C, 112
 PHP, 114
 Python, 113, 146
 w konsoli
 graficznej, 108
 systemowej, 109

repozytoria LEDE, 73

resetowanie
 minikomputera, 41, 58
 ustawień, 43

RFID, Radio-frequency
 identification, 28, 140

RFID/NFC, 37
 Expansion, 28
 cena, 31
 parametry techniczne,
 29
 dostęp do modułu, 145
 Reader, 140, 141
 w języku Python, 146
 w konsoli graficznej,
 140
 w konsoli systemowej,
 141

rozmiar partycji, 60

Ruby, 90, 91

ruter, router, 131, 136
 konfiguracja
 interfejsu sieciowego,
 137

rutingu, 137
 sieci bezprzewodowej,
 136
 szyfrowanie, 136

ruting, 137

S

Samba, 160
 dostęp do plików, 161
 instalacja, 160
 konfiguracja, 160

Sensor Monitor, 37, 102

Servo (PWM) Expansion,
 26, 77, 116, *Patrz także*

PWM
 cena, 31
 parametry techniczne,
 26

serwer, 134
 ejabberd, 175
 konfiguracja
 DHCP, 135
 interfejsów, 134
 komunikacyjny Jabber,
 175

Minecraft, 181

plików, 160

WWW, 80

sieć
 bezprzewodowa, 38
 LAN, 131
 przewodowa, 131
 Wi-Fi, 133

skanowanie
 sieci Wi-Fi, 38
 urządzeń
 Bluetooth, 151
 NFC, 142

slot na kartę, 57

Sparkfun, 37

SPI, Serial Peripheral Interface, 15, 23, 44–46, 75

SSH, Secure Shell, 37, 109

SSL, Secure Socket Layer, 163

- generowanie
 - certyfikatów, 163, 165

sterowanie diodą, 51

system

- kontroli wersji, Git, 73, 181
- operacyjny LEDE, 70
- plików ext4, 59
- wbudowany,
 - embedded system, 70, 178

szfrowanie, 136

- połączenia WWW, 163
- WPA2, 137

Ś

środowisko

- programistyczne Node.js, 76

T

Terminal SSH, 40

Timelapse Camera, 37

toolchain, 178

tworzenie partycji, 65

U

UART, Universal Asynchronous Receiver and Transmitter, 15, 21, 23, 29, 44–46

ubus, 126

udostępnianie sieci, 133

USB, 61

USB-to-Serial, 16, 20, 32, 36, 39

ustawienia fabryczne, 43, 58

UUID, Universally Unique Identifier, 60, 63

uvcdynctrl, 171

W

WEBCAM, 170–172

Wi-Fi, 133

wirtualna maszyna Javy, 85

WPA2, Wi-Fi Protected Access II, 136

wypełnienie, duty cycle, 116

wyświetlanie

- plików graficznych, 97
- usług, 126

X

XMPP, Extensible Messaging and Presence Protocol, 10, 90, 175

- klient mcabber, 176
- panel administracyjny, 177

Z

zapis migawek, 173

zarządzanie

- ekranem OLED, 92
- IoT, 166
- kamerą, 171

zasilanie, 116

zdalne zarządzanie IoT, 166

zdalny monitoring, 170

złącze

- 30-pinowe
 - interfejsy, 45
 - zakresy napięć, 45
- JST-PH, 18

PROGRAM PARTNERSKI

— GRUPY HELION —

1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

<http://program-partnerski.helion.pl>

GRUPA
Helion 

Internet rzeczy z minikomputerem Omega2

- Poznaj budowę i zasadę działania popularnego sterownika IoT
- Odkryj niesamowite możliwości oferowane przez Omegę
- Naucz się realizować praktyczne projekty urządzeń IoT

Internet rzeczy to hasło, które codziennie przewija się nie tylko w specjalistycznej prasie, ale też w mediach głównego nurtu. Nie mamy tu do czynienia z techniczną ciekawostką czy fantastyką naukową. IoT otacza nas już właściwie z każdej strony, i to w dosłownym sensie – „sprytnych” sprzętów jest wokół nas coraz więcej. Lodówki zamawiające żywność, inteligentne domy dbające o nasz komfort i bezpieczeństwo, samochody, które w razie wypadku drogowego alarmują służby ratunkowe, aby te mogły szybko udzielić pomocy – działanie tych urządzeń zawdzięczamy rozwojowi technologii związanych z internetem rzeczy.

Wszystkie też mają pewną wspólną cechę: aby spełniać swoje funkcje, muszą być odpowiednio sterowane i skomunikowane ze światem zewnętrznym. Realizuje się to za pomocą zintegrowanych minikomputerów, których przedstawicielem jest Omega2 – główny bohater tej książki. Jej lektura pozwoli poznać podstawowe informacje na temat tego minikomputera, odkryć możliwości, które oferuje, oraz nauczyć się praktycznie wykorzystywać je do swoich celów. Nie musisz być inżynierem, żeby zacząć przygodę z Omegą – wystarczy podstawowa znajomość zagadnień informatycznych, ciekawość i chęć do nauki. Dzięki tej książce fascynujący świat IoT stanie przed Tobą otworem!

- Parametry techniczne minikomputera Omega2
- Sposób uruchomienia i konfiguracji urządzenia
- Rozszerzanie pamięci i aktualizacja oprogramowania
- Wyświetlanie danych tekstowych i graficznych
- Odczyt analogowych sygnałów wejściowych
- Sterowanie obwodami i urządzeniami zewnętrznymi
- Odczyt danych GPS, komunikacja sieciowa i Bluetooth
- Praktyczne zastosowania w komunikacji i monitoringu
- Tworzenie projektów IoT i generowanie kodu dla Omegi

IoT od podstaw – TYLKO z tą książką!

 helion.pl	<i>Sprawdź nasze szkolenia!</i> SZKOLENIA  AKADEMIA IT & BUSINESS HELIONSZKOLENIA.PL	KOD KORZYŚCI Sięgnij po więcej! ▶  ISBN 978-83-283-6629-9  9 788328 366299
 HELION SA ul. Kościuszki 1c 44-100 Gliwice tel.: 32 230 98 63 helion@helion.pl		interface Poland Sp. z o.o.
INFORMATYKA W NAJLEPSZYM WYDANIU		Cena: 39,90 zł