



Seria urządzeń NPE

Instrukcja użytkownika



Spis treści

Rozdział 1 Wprowadzenie	5
1.1. O dokumencie NPE	5
Organizacja dokumentu	5
1.2. O urządzeniu NPE	6
Najważniejsze cechy	6
Wersje urządzeń	7
Architektura urządzenia	8
Interfejsy i zasoby sprzętowe	8
Wbudowana funkcjonalność	9
1.3. O systemie NPE	9
Architektura systemu	9
Dostępne pakiety	9
Rozdział 2 Uruchomienie urządzenia	11
Opis wyprowadzeń	11
Opis złącz	12
Podłączenie zasilania	13
2.1. Sygnalizacja parametrów pracy	14
Stan połączenia GPRS	14
2.2. Podłączenie do komputera	16
Podłączenie za pomocą portu Ethernet	16
Kabel połączeniowy	16
Domyślna konfiguracja IP urządzenia	16
Konfiguracja IP komputera	17
Test poprawności połączenia sieciowego	18
Uzyskanie połączenia terminalowego	18
Połączenie FTP	19
Podłączenie za pomocą portu szeregowego	19
Kabel połączeniowy dla komunikacji szeregowej RS-232	19
Uruchomienie aplikacji terminalowej	20
2.3. Zarządzanie startem urządzenia i aplikacji	22
Skrypt autostartu	22
Jak dodać własną aplikację do autostartu	22
Konfiguracja startowa	22
Restart urządzenia	22
Przycisk Reset	23
Polecenie reboot	24
Przywracanie ustawień fabrycznych (przycisk SW)	24
Automatyczny restart - watchdog	25
2.4. Połączenie GPRS	25
Nawiązanie połączenia GPRS	25
Logowanie do sieci GSM	25
Nawiązanie połączenia	26
Rozłączenie połączenia	26
Zarządzanie modemem GPRS	26
2.5. Uruchomienie pierwszej aplikacji	27
Sprawdzenie dostępnego miejsca w pamięci Flash	27
Java	27
Tworzenie aplikacji	27
Uruchomienie aplikacji	30
ANSI C, C++	30
Kompilowanie pierwszej aplikacji	30
Upload i uruchomienie aplikacji	30
Rozdział 3 Konfiguracja i zarządzanie zasobami urządzenia	32

3.1. Jak konfigurujemy urządzenie	32
3.2. Konfiguracja Startowa	32
Format pliku	32
Położenie pliku	32
Przykładowy plik	33
Opis parametrów	33
Parametry określające start usług	33
Konfiguracja interfejsu Ethernet	33
Konfiguracja modemu GSM*	34
3.3. Zarządzanie zasobami hardware	35
Dostępne narzędzia	35
3.4. Zarządzanie systemem	35
Zarządzanie procesami, demonami	35
Zatrzymanie procesu	36
Ustawianie zegara systemowego	36
Manualne ustawianie czasu	36
Ustawienie strefy czasowej	37
Synchronizacja daty z serwerem czasu	37
Wykonywanie zaplanowanych zadań	38
Rozdział 4 Konfiguracja i zarządzanie interfejsami komunikacyjnymi	39
4.1. Telnet / FTP	39
4.2. DNS	39
4.3. Apache	41
4.4. PHP	42
4.5. IPTABLES	42
Definiowanie filtrów	42
Zapisanie filtrów w autostarcie	44
4.6. NAT	44
Przekierowanie wewnętrznych portów	45
Przekierowanie portu telnetowego	45
Przekierowanie portu FTP	46
Przekierowanie ruchu na inny adres IP	46
Udostępnienie połączenia internetowego poprzez GPRS	47
4.7. Połączenie GPRS	49
Zarządzanie połączeniem - skrypt gprs	49
Multipleksacja portu szeregowego	49
Konfiguracja rozkazami AT	50
Wysłanie wiadomości SMS	51
4.8. NFS (sieciowy system plików)	52
4.9. Mail	53
4.10. SNMP	54
Przeglądarka SNMP	54
Odczyt/Zapis zasobu	55
Rozdział 5 Przykładowe aplikacje	57
5.1. Język JAVA	57
Przykład 1: Serwer TCP/IP	57
Przykład 2: Klient TCP/IP	58
Przykład 3: Test portu szeregowego	58
ShowCOM	58
DualSerialTest	59
Programy Writer i Reader	59
Program TestHardware	60
5.2. Język ANSI C	60
Przykład 1: Serwer TCP/IP	60
Przykład 2: Klient TCP/IP	61
Przykład 3: Test portu szeregowego	61
RStest	61

Przykład 4: Komunikacja RS232/Ethernet	62
Com2Tcp	62
Rozdział 6 Programowanie NPE	63
6.1. Narzędzia deweloperskie	63
NPE Tool Chain	63
Instalacja	63
Kompilowanie aplikacji	64
6.2. ANSI C/ C++	65
Przygotowanie środowiska Eclipse C/C++	65
Debugging programem Insight	69
Opis API	70
Dostęp do portów wyjściowych	71
Dostęp do portów wejściowych	71
Sterowanie diodami LED	71
Sterowanie Buzzerem	71
Odczyt wejść przetworników ADC	72
6.3. JAVA	72
JamVM	72
Instalacja JamVM	72
Pierwszy projekt w Eclipse SDK	73
Opis API	75
Odczyt wartości wejść cyfrowych	75
Ustawianie stanu wyjść cyfrowych, diod LED i Buzzera	75
Odczyt wartości przetworników ADC	76
Rozdział 7 Bazy Danych	77
7.1. Systemy baz danych	77
Postgre SQL	77
Instalacja	77
Narzędzia	78
Dostęp z wiersza poleceń	78
Przykładowe skrypty	80
SQLite3	80
INSTALACJA	80
Narzędzia	80
Dostęp z wiersza poleceń	81
Przykładowe skrypty	81
7.2. Dostęp z poziomu języka PHP	81
SQLite3	81
PostgreSql	81
Przykładowe skrypty	82
7.3. Dostęp z poziomu języka Java	82
Postgre Sql	82
SQLite3	83
7.4. Wymagania systemowe:	83
Rozdział 8 Dodatki	84
8.1. Informacje podstawowe	84
Wersja systemu	84
8.2. Specyfikacje	84
Specyfikacja urządzenia	84

Rozdział 1 Wprowadzenie

1.1. O dokumencie NPE

Niniejsza dokumentacja dotyczy komputerów przemysłowych z serii NPE wyposażonych w wydajny procesor RISC ARM9 z preinstalowanym w pamięci Flash systemem operacyjnym Linux. Opcjonalnie urządzenie może być wyposażone w modem GPRS oraz dodatkowe platformy takie jak *iMod*, *TRM*.

Organizacja dokumentu

Rozdział 1. Wprowadzenie

Zawiera podstawowe informacje o niniejszym dokumencie

Rozdział 2: Uruchomienie urządzenia

Opisuje rozmieszczenie i sposób podłączenia do poszczególnych zasobów sprzętowych urządzenia.

Rozdział 3: Konfiguracja i zarządzanie zasobami urządzenia

Opis konfiguracji startowej urządzenia oraz podstawy konfiguracji aplikacji.

Rozdział 4: Konfiguracja i zarządzanie interfejsami komunikacyjnymi

Opis konfiguracji interfejsów komunikacyjnych urządzenia.

Rozdział 5: Przykładowe aplikacje NPE

Przykładowe aplikacje demonstrujące główne funkcjonalności urządzenia.

Rozdział 6: Programowanie NPE

Opis tworzenia aplikacji uruchamianych pod systemem NPE.

Rozdział 7: Bazy danych

Opis dostępu i zarządzania systemami baz danych

Rozdział 8: Dodatki

1.2. O urządzeniu NPE

NPE to uniwersalny kontroler zbudowany z myślą o potrzebach rynku automatyki, telekomunikacji i systemów wbudowanych. Obok przedstawiono, umożliwiające wymianę informacji, interfejsy wejścia/wyjścia, którymi dysponuje urządzenie.



- Urządzenia serii NPE to innowacyjna i wydajna platforma dla nowoczesnych systemów automatyki, telemetry oraz zdalnego nadzoru.
- Dzięki systemowi Linux cechuje je stabilność, funkcjonalność i bezpieczeństwo na najwyższym poziomie.
- Sterowniki NPE pracują już w instalacjach automatyki w wielu krajach świata.

Najważniejsze cechy

Kontroler systemów sterowania i monitorowania

- konwerter portów szeregowych
- MODBUS Master/Slave/Gateway
- datalogger, SQL serwer
- serwer SNMP/MAIL/FTP/WWW



Modbus



PSQL



SNMP



E-MAIL



WWW

Spełnia wymogi urządzeń systemów rozproszonych

- niski pobór mocy
- bezwentylatorowa obudowa
- instalacja na szynie DIN
- Watchdog, SecurityChip, RealTime Clock



DIN



Watchdog



Security



RTC



Fanless

Bogactwo funkcji i zasobów sieciowych

- Ethernet i modem GPRS/EDGE
- GPRS router, NAT
- obsługa protokołów VPN, SSH, PPP i wiele innych
- bezpieczeństwo - firewall, SSL



Ethernet



Firewall



NAT



GSM



Router

Rozbudowana Platforma Deweloperska

- wbudowany system Linux
- specjalnie przygotowany SDK
- bogaty zestaw narzędzi programistycznych
- obsługa języków C, C++, JAVA



Linux



SDK



Tools



Java



C/C++

Wersje urządzeń

Podstawowe zasoby NPE - tabela poniżej przedstawia bazową konfigurację hardware dla wszystkich urządzeń serii NPE

MODEL	SDRAM	FLASH MEMORY	SD CARD FLASH	RTC	Ethernet	RS-232	RS-485	Digital Inputs
NPE-9XXX	64/128 MB	up to 1 GB	up to 2 GB	•	•	2	1	8

Zestawienie modeli NPE - tabela poniżej prezentuje dostępne konfiguracje modeli serii NPE

MODEL	DO	DOP	AI DC	AI AC	ONE WIRE	Add. 1 GB Flash	MODEM	MODBUS	SNMP
NPE-9100	6						GPRS/EDGE	○	○
NPE-9100R	2	2					GPRS/EDGE	○	○
NPE-9200	6		4				GPRS/EDGE	○	○
NPE-9201	6		3	1			GPRS/EDGE	○	○
NPE-9200R	2	2	4				GPRS/EDGE	○	○
NPE-9300	6					•	GPRS/EDGE	○	○
NPE-9300W	6				•	•	GPRS/EDGE	○	○
NPE-9300R	2	2				•	GPRS/EDGE	○	○
NPE-9300RW	2	2			•	•	GPRS/EDGE	○	○
NPE-9400	6		4			•	GPRS/EDGE	○	○
NPE-9401	6		3	1		•	GPRS/EDGE	○	○
NPE-9400R	2	2	4			•	GPRS/EDGE	○	○
NPE-9400RW	2	2	4		•	•	GPRS/EDGE	○	○

○ Funkcjonalność opcjonalna

• Funkcjonalność wbudowana

RTC Zegar czasu rzeczywistego

DI Wejścia cyfrowe

DO Wyjścia cyfrowe

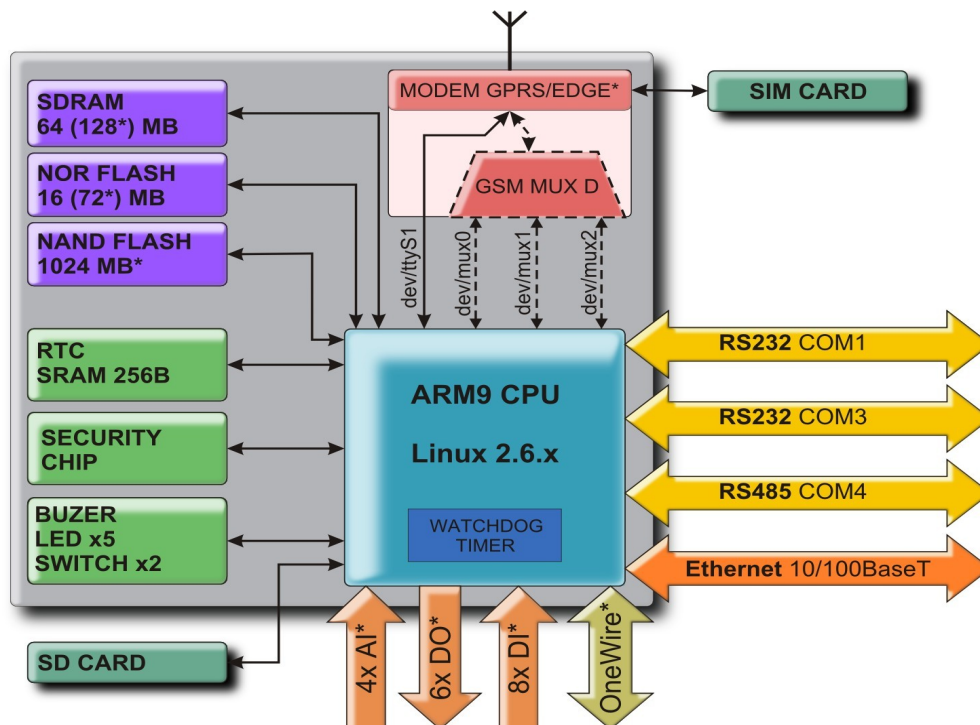
DOP Wyjścia przekaźnikowe

AI DC Wejścia analogowe na prąd stały 0...10 V

AI AC Wejścia analogowe na prąd zmienny 0...70 V

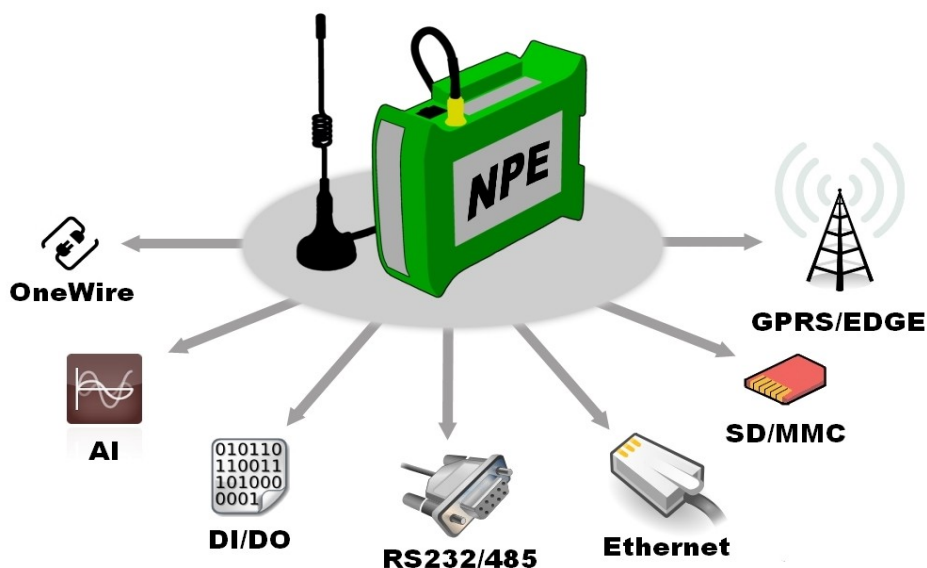
Architektura urządzenia

Sercem komputera jest wydajny i energooszczędny procesor ARM9 typu RISC, co razem z dużymi zasobami pamięci oraz systemem Linux daje nieograniczone możliwości zastosowań.



Interfejsy i zasoby sprzętowe

NPE to uniwersalny kontroler zbudowany z myślą o potrzebach rynku automatyki, telekomunikacji i systemów wbudowanych. Obok przedstawiono, umożliwiające wymianę informacji, interfejsy wejścia/wyjścia, którymi dysponuje urządzenie.



Wbudowana funkcjonalność

NPE może pracować w różnych trybach w zależności od konfiguracji. W ten sposób można uaktywniać jego wybrane funkcje w zależności od potrzeb.

Tryby pracy komputera NPE:

- Konwerter portów szeregowych
- Modbus Master/Slave/Gateway
- Serwer SNMP
- Network GPRS router
- Serwer internetowy: www/ftp/mail
- Rejestrator danych, serwer SQL



Modbus



SNMP



E-MAIL



WWW



PSQL

1.3. O systemie NPE

To, co oferuje system Linux dla serwerów wie każdy specjalista IT. Jednak coraz częściej, dzięki większym mocom energooszczędnych procesorów system ten staje się idealną platformą dla systemów embedded i wszelkiego rodzaju sterowników.

System Linux wyróżniają:

- światowy standard, stabilność oraz nieograniczone możliwości rozbudowy platformy
- olbrzymia baza programów i narzędzi dostępna bezpłatnie, wraz z kodem źródłowym
- dostęp do pełnej bazy wiedzy, dokumentacji, przewodników na różnym poziomie zaawansowania, setki tysięcy specjalistów na całym świecie

Architektura systemu

NPE działa z wykorzystaniem systemu Linux, więc ma dostęp do ogromnej bazy aplikacji. Niektóre z nich są już prekompilowane i dostarczane na życzenie klienta wraz z urządzeniem.

Dostępne pakiety

Podstawowe informacje	
Jądro	Linux Kernel 2.6.x
Stos protokołów	ARP, PPP, IPv4, ICMP, TCP, UDP, DHCP, FTP, SNMP, HTTP, NFS, SMTP, Telnet
System plików	Ext2, Ext3, JFFS2, FAT
Powłoka	Ash
Busybox	Podstawowa kolekcja pakietów systemu Linux
Narzędzia	
ncftp ncftpls	Program do komunikacji FTP
mc	Konsolowy menadżer plików
gzip	Program do kompresji plików

gunzip	Program do dekompresji plików
telnet	Klient telnetowy
Bazy danych	
PostgreSQL*	Relacyjna baza danych
sqlite3	Prosta plikowa baza danych
Demony	
pppd	Demon szeregowego połączenia point_to_point
proftpd	Demon serwera FTP
snmpd	Demon agenta SNMP
telnetd	Demon serwera telnet

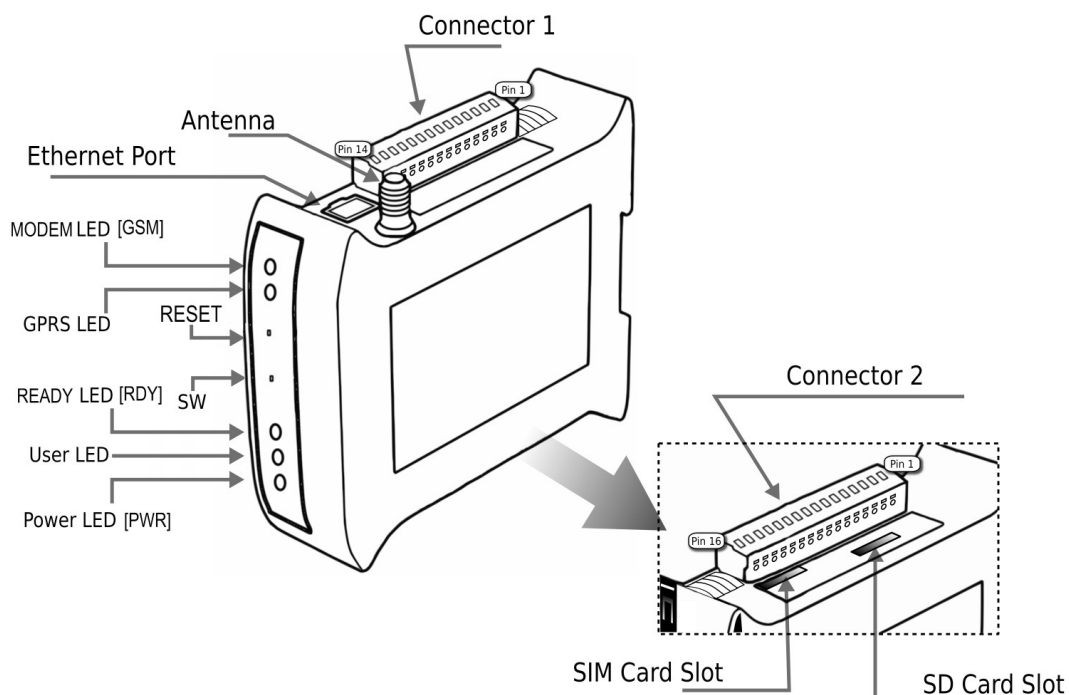
* opcjonalnie



Wszystkie dostępne pakiety można wyświetlić wpisując w konsoli polecenie *busybox*.

Rozdział 2 Uruchomienie urządzenia

Opis wyprowadzeń

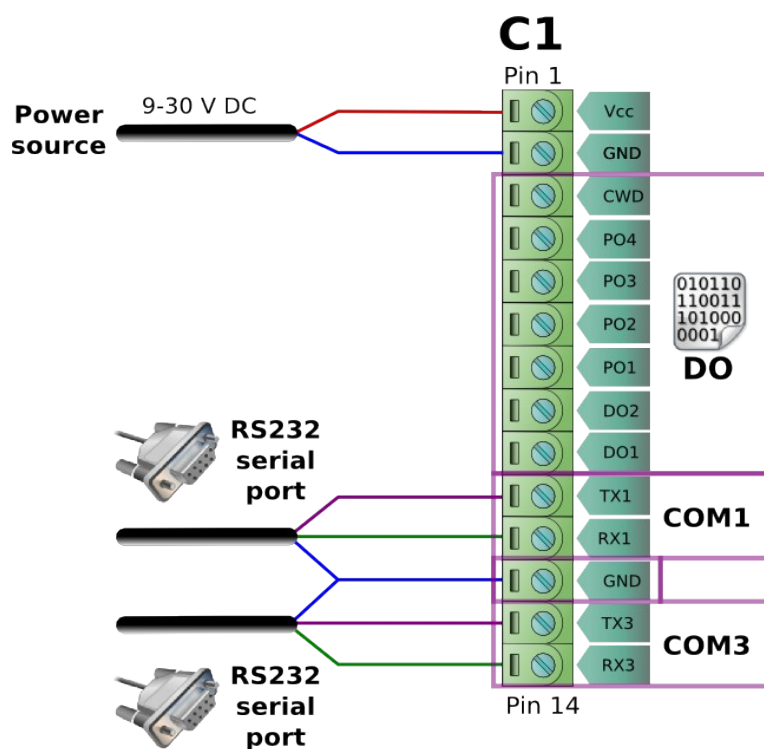


Rys. 1 Wygląd urządzenia

Nazwa elementu	Oznaczenie	Opis
Connector 1	C1	Złącze zasilania i portów szeregowych. Więcej »
Connector 2	C2	Złącze wejść analogowych i portu RS485. Więcej »
SIM Card Slot*	SIM	Slot karty SIM
SD Card Slot	SD	Slot kart SD/MMC
Ethernet Port	ETH	Gniazdo Ethernet
Antenna*	ANT	Wejście antenowe modemu
Power LED	PWR	Dioda zasilania
User LED	USER	Dioda do dyspozycji użytkownika
Status LED	RDY	Dioda statusu urządzenia. Więcej »
GPRS LED	GPRS	Dioda aktywnego połączenia GPRS. Więcej »
Modem LED	GSM	Dioda modemu GSM

* - zależnie od wersji

Opis złącz



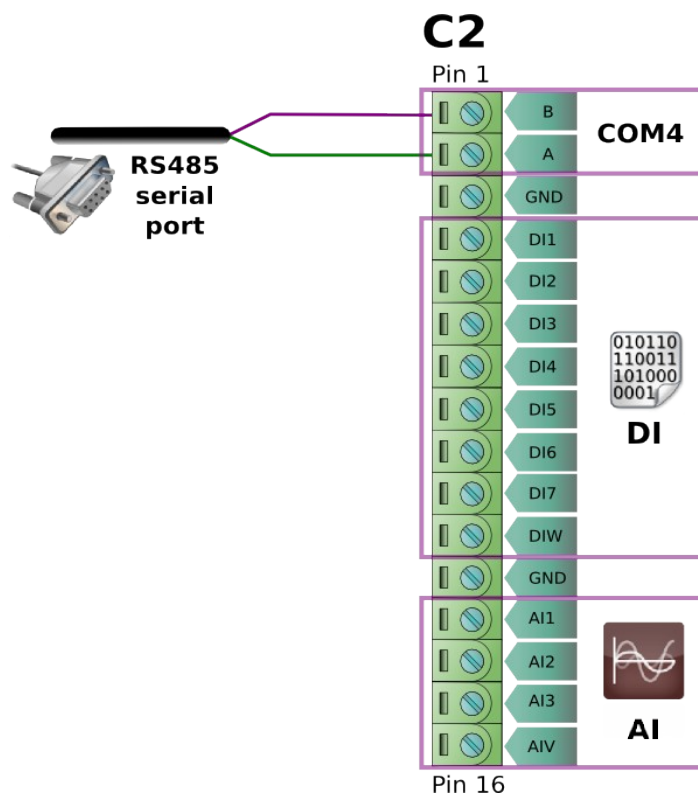
Rys. 2 Opis złącza C1

Numer pinu	Oznaczenie	Opis
1	Vcc	Plus zasilania
2	GND	Minus zasilania (masa)
3	CWD	Zacisk diod zabezpieczających wyjścia tranzystorowe
4	PO4	Wyjście cyfrowe Open Collector lub wyjście przekaźnikowe
5	PO3	Wyjście cyfrowe Open Collector lub wyjście przekaźnikowe
6	PO2	Wyjście cyfrowe Open Collector lub wyjście przekaźnikowe
7	PO1	Wyjście cyfrowe Open Collector lub wyjście przekaźnikowe
8	DO2	Wyjście cyfrowe Open Collector
9	DO1	Wyjście cyfrowe Open Collector
10	TX1	Port szeregowy RS232 - COM1 – Transmit Data
11	RX1	Port szeregowy RS232 - COM1 – ReceiveData
12	GND	Minus zasilania (masa)
13	TX3	Port szeregowy RS232 - COM3 – Transmit Data
14	RX3	Port szeregowy RS232 - COM3 – ReceiveData

Tab. 1 Opis pinów złącza C1



Port COM2 jest wewnętrznym portem szeregowym służącym do komunikacji między urządzeniem a modemem GSM/GPRS/EDGE.



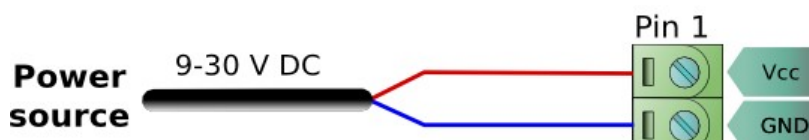
Rys. 3 Opis złącza C2

Numer pinu	Oznaczenie	Opis
1	B	Port szeregowy RS485 – COM4 - DATA+
2	A	Port szeregowy RS485 – COM4 - DATA-
3	GND	Minus zasilania (masa)
4-7	DI1 - DI4	Wejścia cyfrowe
8-10	DI5 - DI7	Wejścia cyfrowe z funkcją przerwaniową
11	DIW	Interfejs OneWire lub wejście cyfrowe z funkcją przerwaniową
12	GND	Minus zasilania (masa)
13-15	AI1 - AI3	Wejścia analogowe pomiaru napięcia stałego
16	AIV	Wejście analogowe pomiaru napięcia stałego lub przemiennego

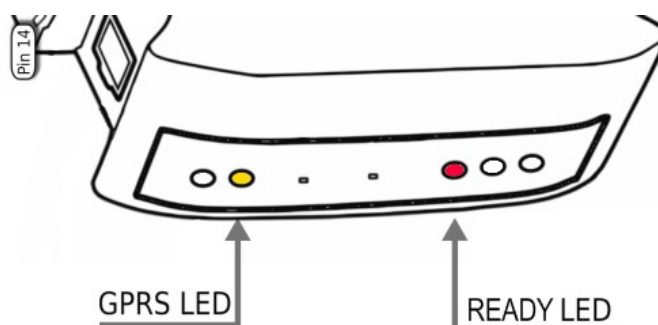
Tab. 2 Opis pinów złącza C2

Podłączenie zasilania

Urządzenie jest gotowe do pracy bezpośrednio po podłączeniu zasilania. Urządzenie potrzebuje źródła zasilania napięciem stałym od 9V do 30V o mocy 12W. Sposób podłączenia zasilania widoczny jest na rysunku 2 przedstawiającym złącze C1.



2.1. Sygnalizacja parametrów pracy



Na ścianie frontowej urządzenia znajdują się dwie diody sygnalizujące status:

1. Dioda statusu pracy urządzenia – READY LED
2. Dioda stanu połączenia GPRS – GPRS LED*

* Występuje tylko w wersji z wbudowanym modemem GPRS/EDGE

Poniżej opisany jest szczegółowo sposób ich sygnalizacji.

Położenie diod uwidocznione jest na rysunku 1.

Stan połączenia GPRS*

Aktualny stan połączenia GPRS sygnalizowany jest przed diodę LED oznaczoną na jako GPRS LED.

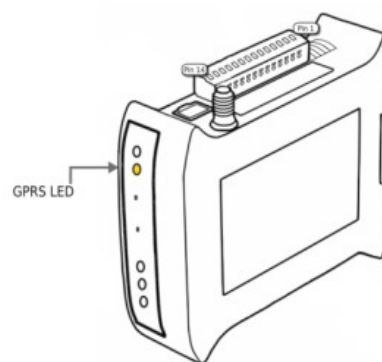







Tabela 3 przedstawia znaczenie każdej z sekwencji migania diody LED.

Lp.	Stan	Typ	Prezentacja	Opis
1	Brak połączenia	info		Dioda zgaszona
2	Inicjalizacja	info		Ciągłe świecenie

* Dotyczy tylko wersji z wbudowanym modemem GPRS/EDGE

3	Aktywne połączenie	info		Powolne miganie (około 1Hz)
4	Oczekiwanie na rejestrację	info		Szybkie miganie
5	Błąd odpowiedzi na ping	warning		Powtarzający się pojedynczy impuls
6	Niepoprawny PIN	error		Powtarzające się dwa krótkie impulsy
7	Wymagany PUK	error		Powtarzające się trzy krótkie impulsy

Tab. 3 Znaczenie stanu diody GPRS



Jeżeli w konfiguracji sprzętowej urządzenia występuje wbudowany modem GPRS nawiązywanie połączenia GPRS jest domyślnie włączone.

Błąd sygnalizowany jako niepoprawny PIN opisany w tabeli 3 wskazuje, że kod PIN zdefiniowany w pliku **/mnt/mtd/syscfg** nie jest poprawny dla użytej karty SIM. Należy go zweryfikować i uruchomić ponownie urządzenie.

**UWAGA**

Po trzykrotnym uruchomieniu konwertera z niewłaściwym kodem PIN karta zostanie zablokowana i będzie trzeba ją odblokować podając kod PUK. Urządzenie będzie to sygnalizować jako błąd – *Wymagany PUK* opisany w tabeli 3.

Należy wówczas, korzystając z połączenia terminalowego użyć polecenia **/mnt/mtd/setpuk** w następujący sposób:

```
/mnt/mtd/setpuk 12345678
```

Gdzie 12345678 jest przykładowym kodem PUK.



Przy każdym uruchomieniu **setpuk** wyświetla pozostałą liczbę prób zmiany numeru PUK. Standardowo wynosi ona 10. Należy uważać, aby nie wykorzystać wszystkich prób, gdyż wiąże się to z zablokowaniem karty SIM.

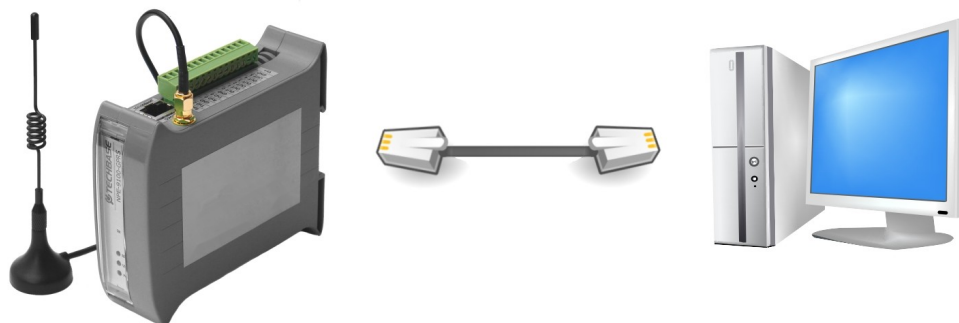
2.2. Podłączenie do komputera

Są dwa kanały bezpośredniego dostępu do konfiguracji i zarządzania urządzeniem za pomocą komputera:

1. poprzez port Ethernet
2. poprzez port szeregowy – COM1

Poniżej opisany jest krok po kroku sposób nawiązania połączenia z urządzeniem wykorzystując każdy z powyższych kanałów komunikacyjnych.

Podłączenie za pomocą portu Ethernet



Rys. 4 Połączenie bezpośrednie z Komputerem

Kabel połączeniowy

Urządzenie należy podłączyć bezpośrednio do komputera za pomocą kabla sieciowego Ethernet.



Urządzenie automatycznie wykrywa typ kabla połączeniowego, dzięki czemu można stosować zarówno kable połączeniowe crossowane oraz standardowe.

Domyślna konfiguracja IP urządzenia

iMod posiada następującą domyślną konfigurację:

Adres IP:	192.168.0.101
Maska podsieci:	255.255.255.0

Tą informację wykorzystamy w następnym kroku.

Konfiguracja IP komputera

Aby nawiązać połączenie należy znać adres IP komputera. Powinien znajdować się on w tej samej podsieci co urządzenie. Podsieć, do której podłączony jest komputer (oraz urządzenie), definiują dwa parametry: adres IP komputera oraz maska podsieci.

Domyślne wartości tych parametrów dla urządzenia podano powyżej i zakładając że nie zostały one zmienione, konfigurację IP na komputerze należy ustawić w następujący sposób:

Adres IP:	192.168.0.X
Maska podsieci:	255.255.255.0

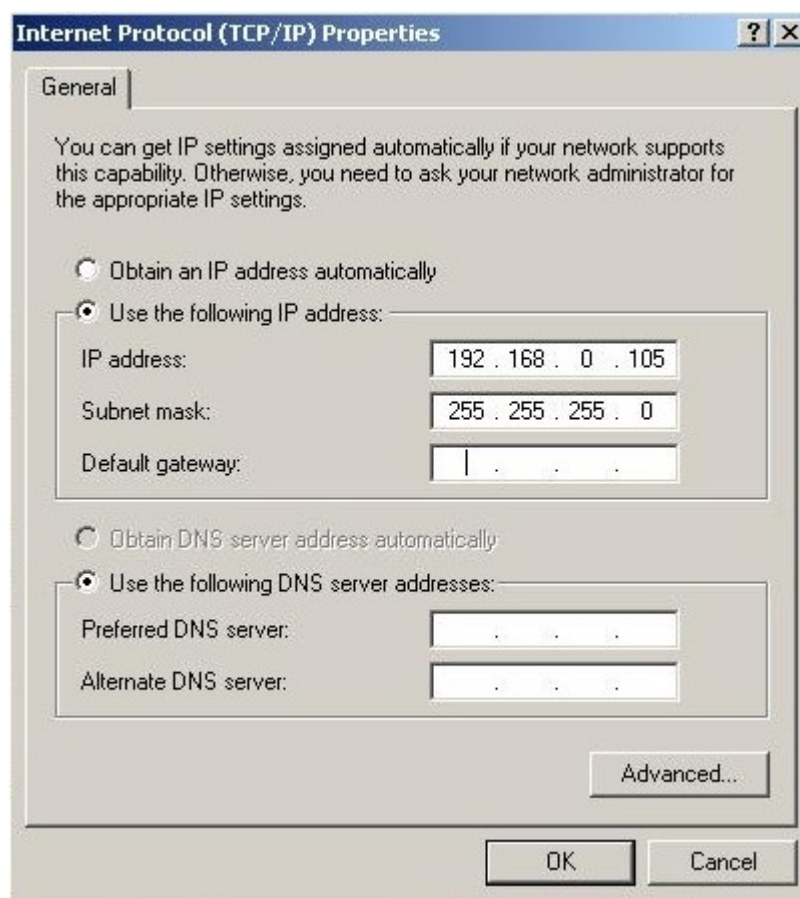
gdzie X oznacza dowolną wartość z przedziału 1-244 wyłączając 101

Przykładowe poprawne adresy IP komputera to: 192.168.0.1 lub 192.168.0.102

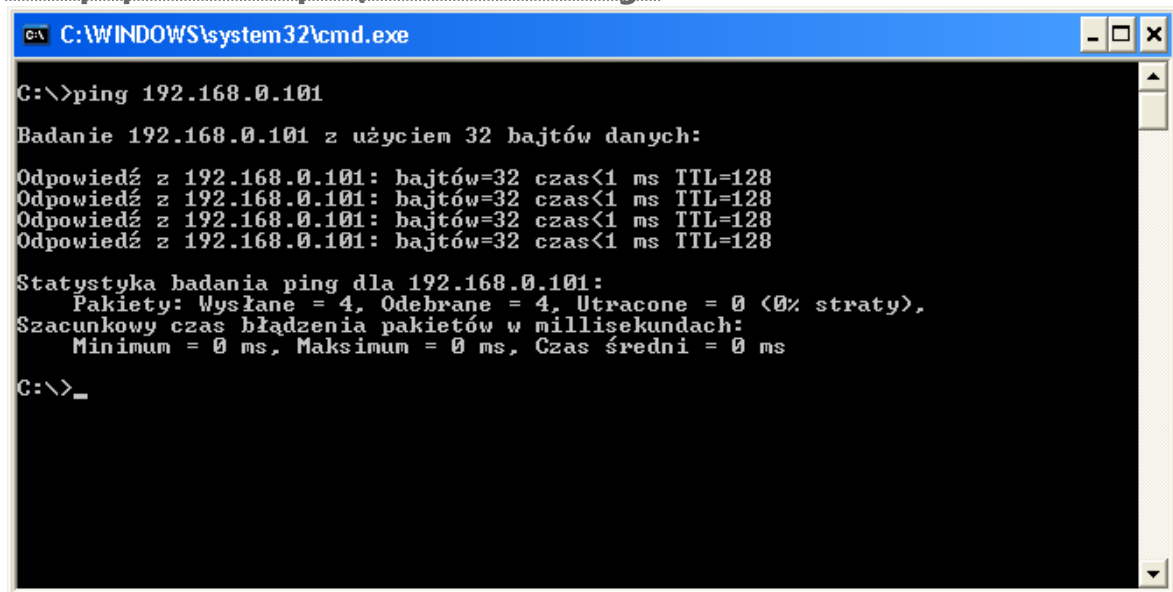


Bieżąca konfiguracja przedstawia ustawienia fabryczne. Jeśli zostały one zmienione należy dostosować parametry Ethernet na komputerze tak aby urządzenie było widoczne w tej samej podsieci. Jeśli nie wiemy jakie parametry zostały ustawione w urządzeniu, należy podłączyć się do urządzenia używając portu szeregowego opisanego w rozdziale: *Podłączenie za pomocą portu szeregowego*, a następnie wyświetlić bieżące parametry Ethernet urządzenia za pomocą polecenia *ifconfig*.

Poniżej przykładowa konfiguracja IP w systemie Windows.



Test poprawności połączenia sieciowego



```
C:\>ping 192.168.0.101

Badanie 192.168.0.101 z użyciem 32 bajtów danych:

Odpowiedź z 192.168.0.101: bajtów=32 czas<1 ms TTL=128
Odpowiedź z 192.168.0.101: bajtów=32 czas<1 ms TTL=128
Odpowiedź z 192.168.0.101: bajtów=32 czas<1 ms TTL=128
Odpowiedź z 192.168.0.101: bajtów=32 czas<1 ms TTL=128

Statystyka badania ping dla 192.168.0.101:
    Pakiety: Wysłane = 4, Odebrane = 4, Utracone = 0 (0% straty),
Szacunkowy czas błędzenia pakietów w millisekundach:
    Minimum = 0 ms, Maksimum = 0 ms, Czas średni = 0 ms

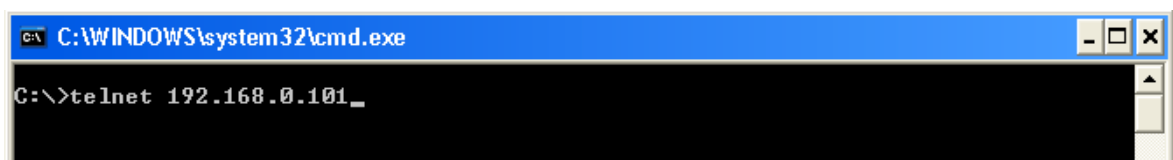
C:\>_
```

Rys. 5 Przykład pinga po prawidłowym podłączeniu

Poprawnie skonfigurowane urządzenie powinno odpowiadać na polecenie `ping <adres IP urządzenia>`

Uzyskanie połączenia terminalowego

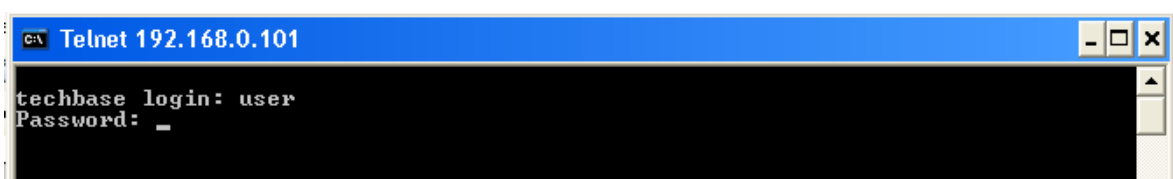
Aby uzyskać połączenie poprzez terminal należy uruchomić „wiersz poleceń”. Następnie wpisać polecenie `telnet` i IP urządzenia:



```
C:\>telnet 192.168.0.101_
```

Rys. 6 Przykład połączenia poprzez telnet

Powinien się pojawić następujący ekran:



```
Telnet 192.168.0.101

techbase login: user
Password: _
```

Rys. 7 Ekran logowania telnet

Aby się zalogować należy podać login i hasło. Domyślne wartości to:

Login: `user`

Hasło: `user`



Wpisywane hasło nie jest w żaden sposób widoczne na ekranie monitora.

Aby uzyskać uprawnienia administratora, należy użyć komendy `su`.

```
$ su
Password: _
```

Domyślne hasło to: `techbase`

Połączenie FTP

W celu połączenia się z urządzeniem poprzez klienta FTP należy mieć podłączone urządzenie do sieci LAN bądź bezpośrednio do komputera.



Sposób podłączenia urządzenia poprzez sieć Ethernet opisany jest w rozdziale *Podłączenie za pomocą portu Ethernet*.

W ustawieniach klienta FTP należy podać następujące dane:

Adres Serwera : **Aktualny adres IP urządzenia**

Port: **21**

Użytkownik: **root**

Hasło: **techbase**

Opcjonalnie można ustawić katalog zdalny na **/** tak aby zawsze rozpoczynać z katalogu głównego.

Podłączenie za pomocą portu szeregowego

Ten rozdział opisuje krok po kroku sposób nawiązania połączenia terminalowego z urządzeniem za pomocą portu szeregowego.

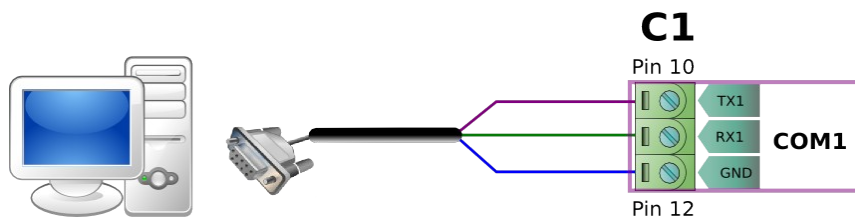


Aby podłączyć urządzenie za pomocą portu szeregowego, usługa dostępu terminalowego na tym porcie musi być włączona. Domyślnie jest ona wyłączona dla portu COM1. Jest to widoczne w pliku konfiguracyjnym SYSCFG w linii **START_CONSOLE=N**

O konfiguracji urządzenia dowiesz się w kolejnym rozdziale pt. *Konfiguracja Startowa*

Kabel połączeniowy dla komunikacji szeregowej RS-232

Rysunek 8 prezentuje sposób podłączenia kabla połączeniowego, łączącego interfejs szeregowy komputera z interfejsem szeregowym COM1 urządzenia. Od strony urządzenia kabel połączeniowy ma postać 3 żył podłączonych do odpowiednich zacisków złącza C1.



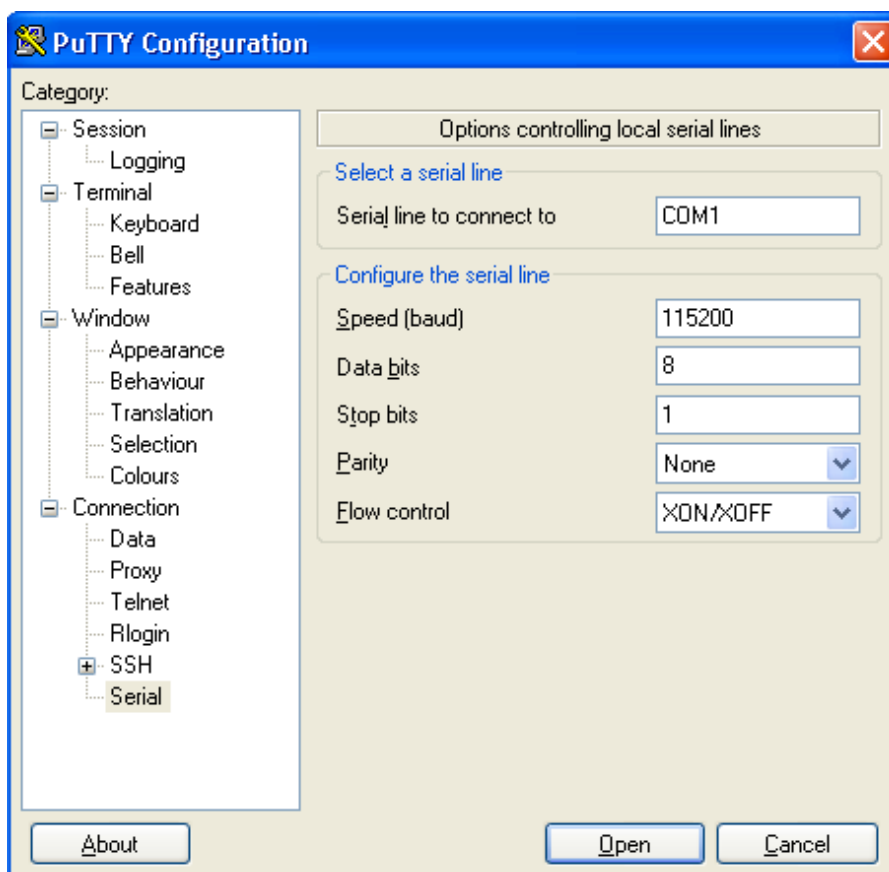
Rys. 8 Podłączenie urządzenia do komputera poprzez port szeregowy

Uruchomienie aplikacji terminalowej

Po podłączeniu urządzenia za pomocą kabla możemy przystąpić do uruchomienia programu terminalowego na komputerze. Może to być dowolny program terminalowy obsługujący porty szeregowo. Tu opisaliśmy konfigurację połączenia na przykładzie popularnego programu PuTTY.



Program PuTTY można ściągnąć z następującej lokalizacji:
<http://putty.very.rulez.org/download.html>



Rys. 9 Ustawienia portu szeregowego w aplikacji PuTTY

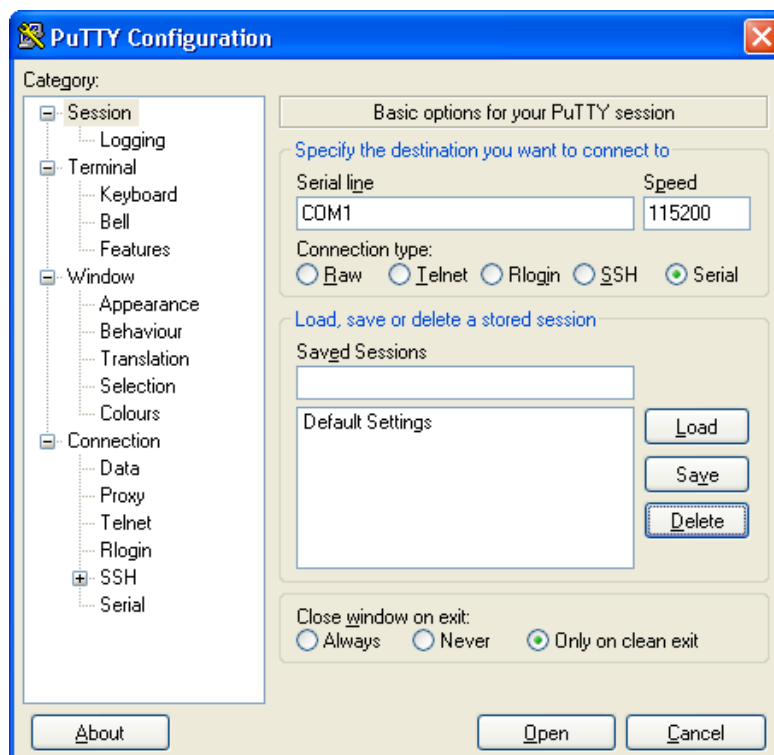
Numer portu na komputerze musi być odpowiednio ustawiony (np. COM1), w zależności do którego portu komputera podłączyliśmy urządzenie.



UWAGA

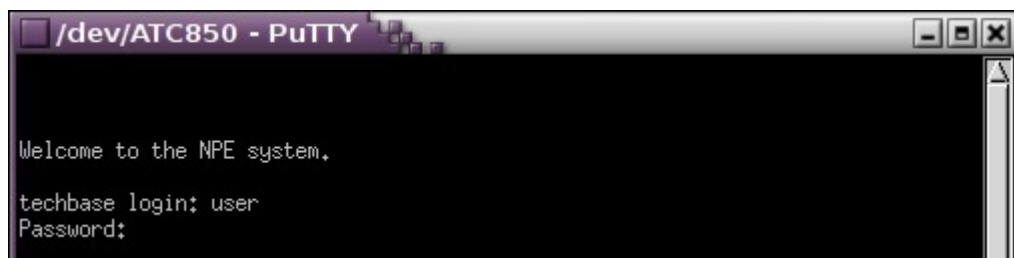
Identyfikator portu COM w programie terminalowym odnosi się do portu szeregowego komputera i nie ma nic wspólnego z identyfikatorem (COM1) w urządzeniu.

Kolejne okienko programu umożliwia otwarcie sesji terminalowej – przycisk *Open*.



Rys. 10 Konfiguracja sesji w programie PuTTY

Po naciśnięciu przycisku *Open* powinno pojawić się następujące okno:



Aby się zalogować należy podać login i hasło. Domyślne wartości to:

Login: *user*

Hasło: *user*

Aby uzyskać uprawnienia administratora, należy użyć komendy *su*.



Domyślnie hasło: *techbase*

2.3. Zarządzanie startem urządzenia i aplikacji

Skrypt autostartu

Jak dodać własną aplikację do autostartu

System NPE podczas inicjalizacji wykorzystuje plik rcs znajdujący się w katalogu /mnt/mtd. Umożliwia on uruchomienie programów wybranych przez użytkownika przy starcie systemu. Struktura pliku wygląda następująco:

```
#!/bin/sh
#
# Custom System Startup script
# Run once at boot time
#

# Start selected services
/etc/init.d/rcS0

# Put custom actions here

#[SET TIMEZONE]
echo "MET-1MET" > /etc/TZ

#[SET TIME via NTP]
/mnt/mtd/ntpdate ntp.task.gda.pl

#[GPRS]
/mnt/mtd/gprs/setup-ln.sh

#[JAVA]
/mnt/sd/JamVM/setup-ln.sh

#[GPRS_AUTOSTART]
/mnt/mtd/gprs/setup-autostart.sh
```

Jeżeli chcemy by nasza aplikacja uruchamiała się przy starcie systemu należy dodać jej wywołanie pod wierszem *echo Custom actions....* analogicznie jak byśmy wpisywali to w konsoli. Komendy dodane do pliku rcs są też uruchamiane przy starcie po zakończeniu wykonywania poleceń startowych z rcs0.

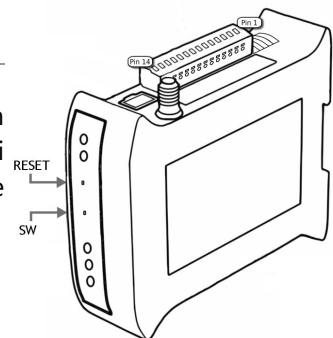
Konfiguracja startowa

Więcej informacji dotyczących konfiguracji startowej urządzenia NPE znajduje się w rozdziale 3.

Restart urządzenia

Zmiany w plikach konfiguracyjnych są zatwierdzone przy ponownym uruchomieniu urządzenia. System NPE udostępnia użytkownikowi sprzętowe i programowe źródła resetu. Ponowne uruchomienie systemu NPE można wymusić poprzez:

- Odłączenie i ponowne podłączenie zasilania
- Przycisk Reset
- Polecenie reboot w konsoli




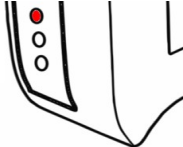
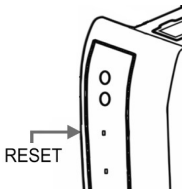






W przypadku gdy urządzenie nie odpowiada na skutek awarii, złych ustawień użytkownika istnieje możliwość przywrócenia ustawień fabrycznych z wykorzystaniem przycisku SW.



Jeżeli łączymy się przy pomocy *aplikacji telnetowej* należy zapamiętać adres IP urządzenia NPE, gdyż podczas resetu wszelkie połączenia zostaną zerwane i wznowione przy ponownym uruchomieniu systemu.

Przycisk Reset

Źródło sprzętowego resetu urządzenia zostało wprowadzone na panelu głównym. Po naciśnięciu przycisku w systemie wywoływane jest przerwanie o najwyższym priorytecie wymuszające ponowne uruchomienie systemu. Schemat restartu jest następujący:

Prezentacja	Akcja użytkownika	Reakcja systemu
<p>Sygnalizacja diodą READY</p>  	<p>Stabilna praca systemu.</p> <p>Jednorazowe naciśnięcie przycisku Reset</p> 	<p>Sygnalizacja diodą READY</p>  <ol style="list-style-type: none"> 1. Wymuszenie resetu systemu: / ----- zamykanie systemu ----- / 2. Odmontowanie partycji 3. Zamknięcie kanałów komunikacyjnych / ----- ponowne uruchomienie ----- / 4. Rozpakowanie obrazu systemu z pamięci Flash do RAM 5. Ładowanie systemu
<p>Sygnalizacja diodą READY</p>  <p>Sygnalizacja diodą USER LED</p>  	<p>W tym momencie istnieje możliwość przywrócenia ustawień fabrycznych przyciskiem SW.</p> <p>Szczegółowy opis znajduje się w kolejnym podrozdziale.</p>	<ol style="list-style-type: none"> 1. Oczekiwanie na wciśnięcie przycisku SW - 5s 2. Wczytanie ustawień z plików <i>rcs</i> i <i>syscfg</i> 3. Uruchomienie systemu 4. Inicjalizacja procesów sterujących systemem
<p>Sygnalizacja diodą READY</p>  <p>Sygnalizacja diodą USER LED</p> 	<p>Możliwość logowania</p>	<ol style="list-style-type: none"> 5. Otwarcie interfejsów komunikacyjnych

Polecenie reboot

Polecenie *reboot* z uprawnieniami *roota*(super użytkownika) wpisane w terminalu wywołuje programowy reset urządzenia. Przebiega on identycznie jak przy wymuszeniu przyciskiem *Reset*. Jednak po wywołaniu zostanie wykonany po zakończeniu działania aktualnie obsługiwanego procesu.


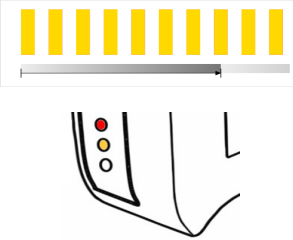
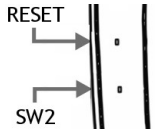
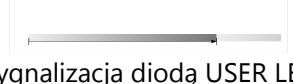



```
$ su
Password:

BusyBox v1.11.0 (2009-03-24 22:02:00 CET) built-in shell (ash)
Enter 'help' for a list of built-in commands.

$ reboot
```

Przywracanie ustawień fabrycznych (przycisk SW)

Jeżeli straciliśmy komunikację, bądź urządzenie nie działa poprawnie to istnieje możliwość przywrócenia ustawień fabrycznych. Dzięki temu zmiany w plikach *syscfg*, *rsc* zostaną anulowane i wczytana zostanie poprawna, fabryczna konfiguracja z plików *syscfg.bak* oraz *rsc.bak*

Prezentacja	Akcja użytkownika	Reakcja systemu
Początkowe akcje są identyczne jak przy Resetowaniu urządzenia.		
Sygnalizacja diodą READY 	W tym momencie istnieje możliwość przywrócenia ustawień fabrycznych przyciskiem SW.	Oczekiwanie na wciśnięcie przycisku SW - 5s
Sygnalizacja diodą USER LED 	Wciśnięcie przycisku SW 	Rozpoczęcie odliczania czasu
Sygnalizacja diodą READY  Sygnalizacja diodą USER LED 	Przytrzymanie przycisku SW	Odliczanie czasu. Po upływie 10s parametry konfiguracyjne przywracane są do wartości fabrycznych
Sygnalizacja diodą READY  Sygnalizacja diodą USER LED 	Puszczenie przycisku SW	Wczytanie poprawnych startowych ustawień fabrycznych.

<p>Sygnalizacja diodą RDY</p> 	<ol style="list-style-type: none"> 1. Wczytanie ustawień z plików <i>rcS</i> i <i>syscfg</i> 2. Uruchomienie systemu 3. Inicjalizacja procesów sterujących systemem 4. Otwarcie interfejsów komunikacyjnych 5. Możliwość logowania
---	---

Automatyczny restart - watchdog

Urządzenie NPE wyposażone jest w wewnętrzny timer Watchdog. Timer bardzo często odnajduje zastosowanie w nadzorowaniu i wykrywaniu niestabilnej pracy systemu. Umożliwia on wymuszenie resetu po określonym czasie w przypadku gdy urządzenie się zawiesi i nie można nawiązać połączenia telnetowego i poprzez port szeregowy. Dzięki temu możemy wymuszać automatycznie ponowne uruchomienie systemu bez konieczności aktywnego bezpośredniego połączenia.

Polecenie inicjalizujące timer Watchdog:

```
watchdog -t <czas w [s | ms]> /dev/watchdog
```

Pole *czas* określa interwał czasowy próby zapisu danych do urządzenia */dev/watchdog*. W przypadku braku dostępu – co sygnalizuje awarię – urządzenie zostanie zresetowane. Aby przetestować poprawność działania timera możemy wymusić zabicie procesu *watchdog*:

```
killall -9 watchdog
```

W tym momencie timer watchdoga zaczyna odliczać do zadanej wartości. Jeżeli osiągnie czas określony przez użytkownika - urządzenie zostanie zresetowane.

2.4. Połączenie GPRS

Model urządzenia z dołączonym modułem modemu GSM w wersjach GPRS lub EDGE, pozwala na wykorzystanie sieci GSM w celu transmisji danych, dźwięku lub wysyłania komunikatów SMS.



Do podstawowej obsługi modemu GSM wykorzystujemy skrypt *gprs* z określonymi atrybutami. Pakiet można pobrać z repozytorium. Dokładny opis wykorzystania skryptu znajduje się w rozdziale Konfiguracja połączenia GPRS.

Nawiązanie połączenia GPRS

Logowanie do sieci GSM

Skrypt *gprs* automatycznie wykonuje wszystkie operacje związane z logowaniem do sieci i pobraniem adresów sieciowych.



UWAGA

Skrypt *gprs* pobiera z pliku *syscfg* parametry potrzebne do nawiązania połączenia. Przykładowo parametr zawierający kod PIN ma fabrycznie przypisaną pustą wartość. Jeżeli wykorzystywana karta wymaga podania kodu PIN należy to uwzględnić w pliku konfiguracyjnym *syscfg*. Więcej

informacji znajduje się w rozdziale Konfiguracja startowa.

Nawiązanie połączenia

W celu nawiązania połączenia z siecią GSM:

1. Modyfikujemy parametry pliku syscfg tak aby odpowiadały naszej karcie SIM
2. Uruchamiamy skrypt *gprs connect* z uprawnieniami roota.

```
$ su
Password:
$ gprs connect
```

Poprawne wykonanie skryptu automatycznie loguje, pobiera adresy i instaluje w systemie nowe połączenie sieciowe ppp0. Po zalogowaniu do sieci GSM modem przechodzi w stan gotowości (opis sygnalizacji diodą GPRS znajduje się w rozdziale: Stan połączenia).

Aktywne połączenia można sprawdzić za pomocą komendy *ifconfig*. Fragment dotyczący nowo utworzonego połączenia powinien wyglądać tak:

```
ppp0      Link encap:Point-to-Point Protocol
          inet addr:87.251.253.95 P-t-P:10.0.0.1 Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST MTU:1500 Metric:1
          RX packets:101 errors:0 dropped:0 overruns:0 frame:0
          TX packets:94 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:6696 (6.5 KiB) TX bytes:7076 (6.9 KiB)
```

Korzystanie z połączenia ppp0 jest analogiczne do pozostałych połączeń sieciowych typu Ethernet.

Rozłączenie połączenia

Rozłączenie aktywnego połączenia GPRS i usunięcie połączenia sieciowego ppp0 można przeprowadzić w dwojaki sposób:

1. Poleceniem *gprs disconnect*, które powoduje tylko rozłączenie aktywnego połączenia. Modem jest nadal uruchomiony i w każdej chwili może ponownie nawiązać połączenie.
2. Poleceniem *gprs modemoff*, które rozłącza aktywne połączenie i odłącza zasilanie modemu GSM. Ponowne nawiązanie komunikacji GPRS poprzedza uruchomienie modemu co wydłuża całą operację.

Zarządzanie modemem GPRS

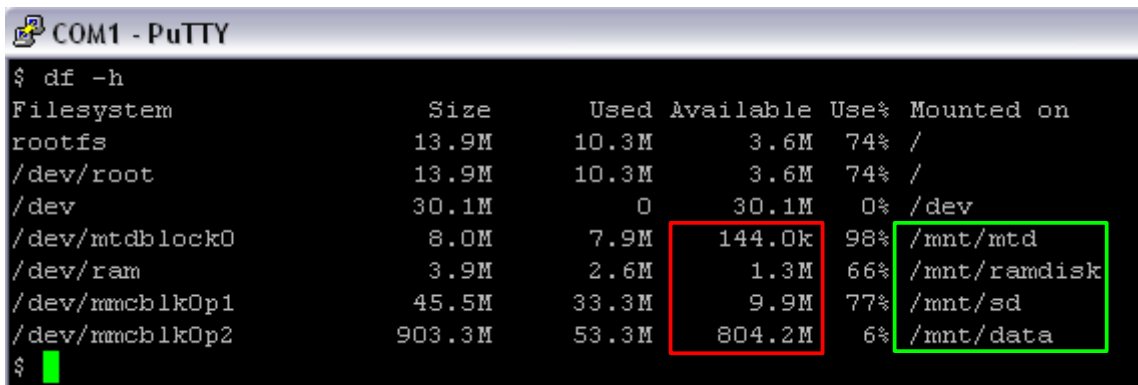
Szczegółowy opis konfiguracji modemu GSM znajduje się w rozdziale: Połączenie GPRS

2.5. Uruchomienie pierwszej aplikacji

Ten rozdział opisuje sposób stworzenia, kompilacji i uruchomienia podstawowej aplikacji napisanej przez użytkownika na podstawie programu HelloWorld stworzonej w języku JAVA i ANSI C/C++. Do programowania wykorzystamy środowisko Eclipse SDK oraz Notatnik.

Sprawdzenie dostępnego miejsca w pamięci Flash

Przed wgraniem i uruchomieniem skompilowanej aplikacji należy upewnić się czy mamy wystarczającą ilość wolnego miejsca na urządzeniu NPE. W tym celu po zalogowaniu w wierszu poleceń wpisujemy:



```
$ df -h
Filesystem      Size      Used Available Use% Mounted on
rootfs          13.9M     10.3M      3.6M   74% /
/dev/root       13.9M     10.3M      3.6M   74% /
/dev            30.1M        0     30.1M    0% /dev
/dev/mtdblock0   8.0M       7.9M     144.0k   98% /mnt/mtd
/dev/ram         3.9M       2.6M      1.3M   66% /mnt/ramdisk
/dev/mmcblkOp1  45.5M     33.3M      9.9M   77% /mnt/sd
/dev/mmcblkOp2 903.3M     53.3M    804.2M    6% /mnt/data
$
```

W efekcie uzyskaliśmy przykładowy opis ilości wolnego miejsca na dostępnych partycjach. Partycje dostępne dla użytkownika znajdują się w katalogu `/mnt/`.

Java

Urządzenie posiada możliwość zainstalowania wirtualnej maszyny Javy – *JamVM*, która cechuje się małym rozmiarem i jest dedykowana do systemów typu Embedded takich jak NPE. Opis sposobu instalacji pakietu *JamVM* znajduje się w rozdziale: Instalacja *JamVM*

Tworzenie aplikacji

Aplikacje tworzone w języku Java cechują się przenośnością, zatem można je tworzyć niezależnie od systemu docelowego. Wystarczy dowolny program wspierający programowanie w tym języku a nawet prosty edytor tekstowy zarówno w systemie Windows jak i Unix.

Do stworzenia pierwszej aplikacji można wykorzystać zwykły edytor tekstowy. W tym celu otwieramy *notatnik*(w systemie Windows) albo *gedit*(w systemie Linux).

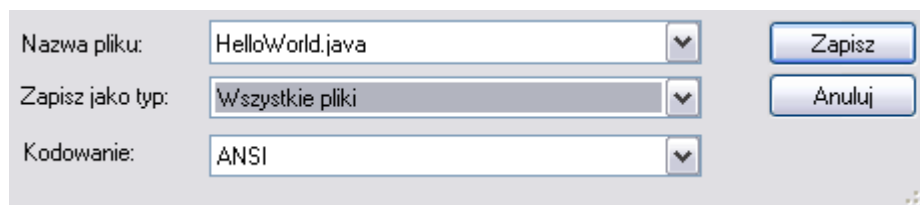
W pustym pliku w wpisujemy:

```
public class HelloWorld {

    public static void main(String[] args) {
        System.out.println("Hello, Java World!");
    }

}
```

Plik zapisujemy nadając mu nazwę HelloWorld.java.



Ważne aby przy zapisie zmienić pole *Zapisz jako typ* na *Wszystkie pliki*



Nazwa klasy głównej może być dowolna (inna niż HelloWorld). Jednak należy pamiętać aby nazwa pliku była identyczna z nazwą głównej klasy.

Następnym krokiem jest skompilowanie utworzonego pliku *HelloWorld.java*. W efekcie kompilacji stworzony zostanie plik *HelloWorld.class*, który można uruchomić na dowolnej maszynie Javy. Proces kompilacji zostanie wywołany bezpośrednio z wiersza poleceń za pomocą komendy:

```
javac NazwaPliku.java
```

```
C:\>javac HelloWorld.java
```

Jeżeli pojawi się komunikat :

```
C:\>javac HelloWorld.java
Nazwa 'javac' nie jest rozpoznawana jako polecenie wewnętrzne lub
program wykonywalny lub plik wsadowy.
```

Oznacza to, że w systemie nie ma zainstalowanego *Java JDK* lub co najbardziej prawdopodobne nie mamy dodanej ścieżki do katalogu *Java JDK* w zmiennej *PATH*.

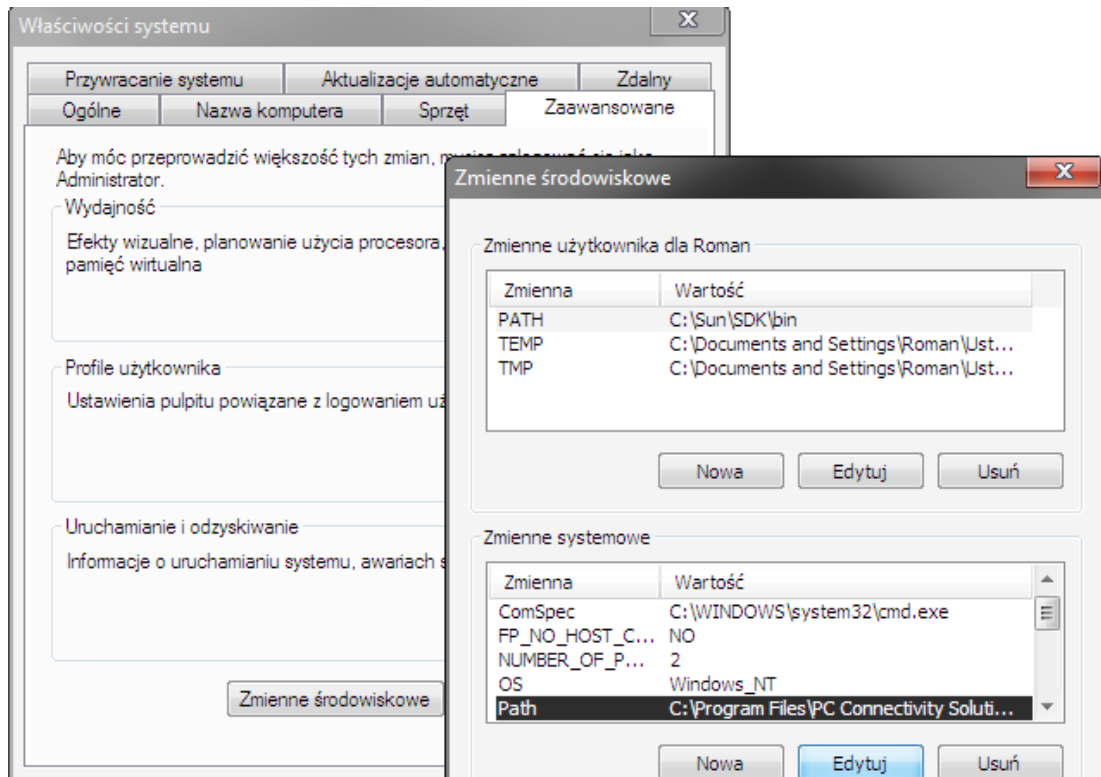


Aktualną wersję *Java JDK* możemy pobrać ze strony producenta:

<http://java.sun.com/javase/downloads/widget/jdk6.jsp>

Dodanie ścieżki do zmiennej PATH:

1. Klikamy prawym przyciskiem na Mój komputer
2. Wybieramy Właściwości >> zakładka Zaawansowane >> przycisk Zmienne środowiskowe
3. Wybieramy zmienna **PATH** z pola Zmienne systemowe >> przycisk Edytuj



4. Na końcu ścieżki w polu wartości zmiennej dopisujemy po średniku

C:\Program Files\Java\jdk1.6.0_17\bin



Ścieżka do katalogu bin może być różna w zależności od miejsca instalacji i wersji pakietu Java JDK. Należy się upewnić czy w katalogu znajdują się pliki wykonywalne *java* oraz *javac*

Po wykonaniu powyższych czynności ponawiamy proces kompilacji i jeżeli nie pojawiły się błędy w składni pliku *HelloWorld.java* utworzony zostanie plik *HelloWorld.class*.

Następnie możemy uruchomić i przetestować stworzony program lokalnie na komputerze PC. Uruchamiamy za pomocą polecenia:

```
java NazwaPliku
```

W efekcie powinniśmy uzyskać:

```
C:\>javac HelloWorld.java
C:\>java HelloWorld
Hello, Java World!
```


Uruchomienie aplikacji

Wirtualna maszyna Javy *JamVM* posiada możliwość uruchomienia aplikacji z rozszerzeniami **.class* i **.jar*. Chcąc uruchomić utworzoną w poprzednim rozdziale klasę *HelloWorld.class* musimy skopiować plik na urządzenie (przykładowo to katalogu */user*). W tym celu należy nawiązać połączenie FTP dowolnym programem.



Opis konfiguracji połączenia FTP znajduje się w rozdziale: *Połączenie FTP*

Następnie nawiązujemy połączenie *telnetowe* za pomocą konsoli systemowej i wyświetlamy zawartość katalogu., sprawdzając czy plik został skopiowany poprawnie:

```
$ cd /home/user/  
$ ls  
HelloWorld.class
```

Aby uruchomić stworzoną klasę za pomocą wirtualnej maszyny należy w terminalu uruchomić skrypt *java* z podaną po spacji ścieżką katalogu, w którym znajduje się nasz plik oraz jego nazwą. Jeżeli poprawnie wprowadzimy dane zobaczymy taki sam efekt jak podczas uruchomienia w programie Eclipse:

```
$ java -cp . HelloWorld  
Hello, Java World!
```

*Kropka w systemie Unix oznacza ścieżkę do katalogu w którym aktualnie przebywamy..

ANSI C, C++

Języki ANSI C, C++ są najpopularniejszymi językami umożliwiającymi tworzenie aplikacji, systemów operacyjnych i innych programów wykorzystujących bezpośredni – niskopoziomowy dostęp do zasobów danego urządzenia.

Kompilowanie pierwszej aplikacji

Programy napisane w języku C/C++ wymagają kompilacji pod dedykowaną architekturę systemu, na którym mają być uruchamiane.



Opis tworzenia i wszystkich narzędzi potrzebnych przy kompilacji znajduje się w rozdziale: *Narzędzia developerskie*

Upload i uruchomienie aplikacji

Jeżeli posiadamy wystarczającą ilość wolnego miejsca możemy wgrać aplikację na urządzenie NPE za pomocą dowolnego klienta FTP. Na potrzeby tej instrukcji wykorzystamy gotowy, skompilowany program *Hello*, który znajduje się na dołączonej do urządzenia płycie CD w katalogu *Apps/Hello*.

Dla ułatwienia warto skopiować cały katalog *Hello* zawierający również źródła.



Opis konfiguracji połączenia FTP znajduje się w rozdziale *Połączenie FTP*

Po poprawnym wgraniu katalogu *Hello* pliku na urządzenie NPE za pomocą konsoli przechodzimy

do katalogu *Hello* .

Aby uruchomić program musimy nadać uprawnienia dla pliku wykonywalnego. W tym celu posłużymy się poleceniem:

```
chmod + x hello
```

Następnie możemy uruchomić program za pomocą polecenia:

```
./hello
```

Przy poprawnym uruchomieniu powinniśmy zobaczyć:



Rozdział 3 Konfiguracja i zarządzanie zasobami urządzenia

3.1. Jak konfigurujemy urządzenie

Urządzenie NPE konfiguruje się za pomocą jednego pliku konfiguracyjnego.

1. Plik `syscfg` odpowiada za opcje startowe urządzenia systemu NPE. Ta konfiguracja zwana jest dalej Konfiguracją Startową.



Aby zmodyfikować plik startowy można użyć połączenia FTP z urządzeniem. Szczegółowe informacje na temat tego jak uzyskać połączenie między systemem NPE a komputerem, znajdują się w rozdziale: *Połączenie FTP*



Aby nowa konfiguracja została wczytana, wymagany jest restart urządzenia. Można to zrealizować stosując polecenie `reboot` z konsoli lub wyłączyć i włączyć zasilanie urządzenia.

3.2. Konfiguracja Startowa

Format pliku

Plik konfiguracji startowej ma postać pliku tekstowego o formacie typu:

Parametr=Wartość



Niedozwolone jest umieszczanie spacji pomiędzy znakiem '=' a tekstem. Nie należy również stosować cudzysłówów np. "Wartość".

Położenie pliku

Plik konfiguracyjny znajduje się w lokalizacji:

Ścieżka: `/mnt/mtd/`

Nazwa: `syscfg`

Przykładowy plik

```
# Custom System configuration file
#
HOST_NAME=techbase

RTC_RESTORE=Y
OUT_RESTORE=Y
DEFAULT_ROUTE=GPRS

ETH_IP=192.168.0.1
ETH_MASK=255.255.0.0
ETH_IP=192.168.0.101
ETH_MAC=18:83:13:09:85:e7

START_CONSOLE=N
START_BLINK=N
START_DHCP=Y
START_FTP=Y
START_TELNET=Y
START_SNMP=N
START_NPESRV=N

GSM_BAUD=230400
GPRS_PIN=
GPRS_MUX=Y
GPRS_AUTOSTART=Y
GPRS_DNS=AUTO
GPRS_APN=
GPRS_RECONNECT=N
GPRS_PING_IP_1=208.67.222.222
GPRS_PING_IP_2=208.67.220.220 #
```

Opis parametrów

Parametry określające start usług

Nazwa parametru	Opis	Format wartości	Wartość domyślna
START_CONSOLE	Uruchomienie konsoli systemowej na porcie szeregowym COM1 (/dev/ttyS0)	Y lub N	N
START_DHCP	Uruchomienie klienta DHCP (dynamiczne IP)	Y lub N	Y
START_FTP	Uruchomienie serwera FTP	Y lub N	Y
START_TELNET	Uruchomienie usługi telnet	Y lub N	Y
START_SNMP	Uruchomienie obsługi SNMP Opcja dostępna tylko dla wersji z SNMP	Y lub N	N

Konfiguracja interfejsu Ethernet

Nazwa parametru	Opis	Format wartości	Wartość domyślna
HOST_NAME	Nazwa hosta	String	techbase
GW_IP	Adres IP bramy sieciowej	XXX.XXX.XXX.XXX	192.168.0.1
NET_MASK	Maska podsieci	XXX.XXX.XXX.XXX	255.255.0.0
HOST_IP	Adres IP urządzenia	XXX.XXX.XXX.XXX	192.168.0.101
HOST_MAC	Adres MAC urządzenia	XX.XX.XX.XX.XX.XX	Widoczny na obudowie

Konfiguracja modemu GSM*

Nazwa parametru	Opis	Format wartości	Wartość domyślna
GPRS_AUTOSTART	Automatyczne nawiązywanie połączenia GPRS podczas startu modemu	Y lub N	Y
GPRS_RECONNECT	Uruchomienie procesu wznowiającego połączenie GPRS po jego przerwaniu	Y lub N	N
GPRS_APN_NAME	Nazwa APN dla połączenia GPRS	String	
GPRS_LOGIN	Login do punktu APN połączenia GPRS	String	puste pole
GPRS_PASSWORD	Hasło do punktu APN połączenia GPRS	String	puste pole
GPRS_PIN	Kod PIN dla karty SIM. W przypadku braku kodu PIN na karcie to ustawienie będzie ignorowane.	4 cyfry	
GPRS_DNS	Określenie serwera DNS dla połączenia GPRS	AUTO – serwer DNS pobierany automatycznie XXX.XXX.XXX.XXX- jawne wpisanie konkretnego adresu IP	AUTO
Parametry które w normalnych warunkach pracy nie wymagają modyfikacji			
DEFAULT_ROUTE	Ustawienie trasy domyślnej – Gateway - dla połączeń wychodzących poza podsieć własnego APN i Ethernet - typowo połączenia internetowe.	GPRS - połączenia zewnętrzne obsługiwane przez wbudowany modem GSM/GPRS ETHERNET -połączenia zewnętrzne obsługiwane przez interfejs Ethernet	GPRS
GPRS_PING_IP_1	Adres IP serwera służącego do sprawdzania połączenia GPRS powiązany z GPRS_RECONNECT.	XXX.XXX.XXX.XXX	208.67.222.222 (adres serwera OpenDNS)
GPRS_PING_IP_2	Adres IP serwera służącego do sprawdzania połączenia GPRS powiązany z GPRS_RECONNECT.	XXX.XXX.XXX.XXX	208.67.220.220 (zapasowy adres serwera OpenDNS)
GPRS_MUX	Tryb multipleksacji modemu GSM Umożliwia jednoczesne korzystanie z sesji GPRS oraz wysyłanie SMS	Y lub N	Y
GSM_BAUD	Prędkość komunikacji z modemem GSM.	Dozwolone prędkości w bit/s: 300, 600, 1200, 1800, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400	230400



Jeżeli w konfiguracji sprzętowej urządzenia występuje wbudowany modem GPRS nawiązywanie połączenia GPRS jest domyślnie włączone. Wskazane jest także włączenie ponawiania połączenia poprzez wpis GPRS_RECONNECT=Y, ale razem z tym parametrem należy ustawić odpowiednio parametry GPRS_PING_IP_1 i GPRS_PING_IP_2.

3.3. Zarządzanie zasobami hardware

Dostępne narzędzia

System Linux modułu NPE zawiera zestaw dodatkowych programów i skryptów, które dają wygodny dostęp do sterowania dostępnymi elementami systemu.

Funkcja	Rodzaj	Opis
sdon	Skrypt	Podłącz kartę SD do katalogu /mnt/sd
sdooff	Skrypt	Odłącz kartę SD z katalogu /mnt/sd
mtdon	Skrypt	Podłącz pamięć Flash do katalogu /mnt/mtd
mtdooff	Skrypt	Odłącz kartę Flash z katalogu /mnt/mtd
nfson <zasób>	Skrypt	Podłącz zasób NFS do katalogu /mnt/nfs, przykładowo nfson 192.168.0.200:/nfs
nfsoff	Skrypt	Odłącz zasób NFS z katalogu /mnt/nfs
blink <Led> <Czas>	Skrypt	Start migania sygnalizatora LED. LED1 - 6 LED2 - 7, czas w mikro sekundach
saveusers <sd flash>	Skrypt	Zapisz bieżące pliki konfiguracyjne użytkowników na kartę SD lub do pamięci Flash
settime <MMDDhhmmYYYY>	Skrypt	Ustawienie czasu i daty w zegarze podtrzymywanym bateryjnie
gprs	Skrypt	gprs {connect [force[:%timeout%]] disconnect reconnect [on off] status [nocheck] modemoff abort
changepin	Skrypt	Zmiana numeru PIN karty SIM
setpuk	Skrypt	Podanie kodu PUK w celu odblokowania karty SIM
muxon	Skrypt	Włączenie multipleksacji portu szeregowego
muxoff	Skrypt	Wyłączenie multipleksacji portu szeregowego
modem	Program	Pozwala wysłać komendę AT do modemu GSM (gdy zainstalowany)
npe	Program	Daje dostęp do wszystkich elementów systemu NPE: porty wejścia/wyjścia, zegar i pamięć podtrzymywana bateryjnie
npecfg	Program	Narzędzie konfiguracji systemu NPE

3.4. Zarządzanie systemem

Zarządzanie procesami, demonami

Demony w systemie Linux są programami działającymi w tle, niezależnie od uruchomionych aplikacji. Zazwyczaj umożliwiają dostęp do kanałów komunikacyjnych takich jak http, FTP czy email.



Podczas startu urządzenia domyślnie uruchamiane są demony: *telnetd*, *proftpd*, *snmpd*.

Wyboru uruchamianych demonów dokonujemy poprzez plik *syscfg* i ustawienie konkretnego atrybutu(Y/N).

W celu sprawdzenia aktualnie działających procesów(w tym demonów) należy w konsoli wpisać polecenie *ps* lub uruchomić pakiet *top*.

Przykładowy fragment listy procesów:

```
$ ps
PID    USER      COMMAND
1      root      init
2      root      [ksoftirqd/0]
3      root      [watchdog/0]
4      root      [events/0]
5      root      [khelper]
6      root      [kthread]
55     root      [kblockd/0]
70     root      [kswapd0]
71     root      [aio/0]
72     root      [cifsdlockd]
73     root      [cifsdnotifyd]
891    root      /bin/sh /mnt/mtd/gprs/bin/pulse 150000 150000 150000
1017   root      udhcpd -b -s /usr/share/udhcpd/default.script
1020   nobody    proftpd: (accepting connections)
1023   root      telnetd
1027   root      snmpd
```

Każdy uruchomiony proces w systemie Unix posiada unikalny numer PID, który widoczny jest w pierwszej kolumnie.

Zatrzymanie procesu

Demony fabrycznie wkompirowane w jądro systemu zarządzane są poprzez plik `syscfg`. Każdorazowa zmiana tego pliku wymaga ponownego uruchomienia urządzenia.

Zatrzymanie i poprawne zamknięcie danego procesu/demona jest realizowane poleceniem:

```
kill <ID procesu>
```

Przykładowo na podstawie wcześniej wyświetlonej listy procesów wyłączenie demona odpowiedzialnego za połączenia telnetowe należało by wywołać poleceniem `kill 1023`.

W efekcie zamknięty został proces obsługujący komunikację na porcie 23(telnet). Dla sprawdzenia możemy spróbować się połączyć :

```
telnet <nr IP urządzenia NPE>
```

Połączenie powinno być odrzucone:

```
Łączenie z 192.168.0.107..
Nie można nawiązać połączenia z hostem na porcie 23
```

Ustawianie zegara systemowego

Manualne ustawiania czasu

Zmianę czasu i daty zegara systemowego umożliwia skrypt:

```
settime <MMDDhhmmYYYY>
```

Przykład ustawienia wybranej daty:

```
$ settime 021119281988
Thu Feb 11 19:28:00 MET 1988
Time saved: Thu Feb 11 19:28:00 1988
Time set & saved to NU RTC
```


Ustawienie strefy czasowej

W zależności od miejsca użytkowania urządzenia NPE może zająć potrzeba zmiany domyślnej strefy czasowej (Central Europe - MET-1METDST). Dzięki odpowiedniemu dopasowaniu uniwersalnego czasu wymiana danych pomiędzy stacjami w różnych strefach czasowych będzie przebiegać spójnie. Generowane pliki na każdym z urządzeń będą posiadać ten sam czas lokalny (miejscowy).

Aktualna strefa czasowa znajduje się w pliku `/etc/TZ` a zmiana strefy czasowej odbywa się poprzez nadpisanie tego pliku nową wartością.

Przykładowo polecenia zmiana strefy czasowej Centralnej Europy na strefę Centralną w USA i ustawienia czasu:

```
echo "CST6CDT " > /etc/TZ
settime <MMDDhhmmYYYY>
```

Przed:

```
$ cat /etc/TZ
MET-1METDST
$ date
Thu Jul 15 11:26:10 METDST 2010
```

Po zmianie:

```
$ cat /etc/TZ
MET-1METDST
$ echo "CST6CDT" > /etc/TZ
$ settime 071511262010
Thu Jul 15 11:26:00 CDT 2010
Time saved: DST=1 Thu Jul 15 11:26:00 2010
Time set & saved to NU RTC
```



Zalecane jest umieszczenie polecenia w pliku `/mnt/mtd/rcs` tak aby przy ponownym uruchomieniu automatycznie ustawiona została określona strefa czasowa.

Synchronizacja daty z serwerem czasu

Synchronizacja czasu z serwerem NTP w Linuksie nie wymaga uruchamiania demona. W celu pobrania czasu z sieci należy wyłącznie uruchomić pakiet `ntpdate` a jako argument podać URL serwera czasu.



Pakiet `ntpdate` znajduje się w repozytorium. Wszelkie szczegóły dotyczące repozytorium udzielane drogą telefoniczną bądź mailową.

Przykładowe zastosowanie:

```
/mnt/mtd/ntpdate ntp1.us.edu.pl
```

```
$ /mnt/mtd/ntpdate ntp1.us.edu.pl
22 Jun 13:14:06 ntpdate[7428]: adjust time server 155.158.253.13 offset -0.106662 sec
```

Najefektywniej dodać powyższy skrypt do `crontab`, aby codziennie o określonej godzinie aktualizować i synchronizować czas. Opis wykorzystania `crona` znajduje się w poniższym podrozdziale.

Wykonywanie zaplanowanych zadań

Urządzenie NPE umożliwia okresowe wykonywanie poleceń, programów o zadanej godzinie. System Linux wykorzystuje demona *Cron*d do okresowego wywoływania poleceń zawartych w pliku *crontab*. Jeżeli chcielibyśmy wykorzystać możliwości *Cron*a należy:

1. Utworzyć katalog *crontabs* poleceniem

```
mkdir -p /var/spool/cron/crontabs
```

2. Utworzyć plik *crontab* w katalogu */mnt/mtd/* zawierający polecenia:

```
mm hh DD MM YYYY <polecenie do wykonania>  
mm hh DD MM YYYY <polecenie do wykonania>
```

3. Dodać ścieżkę do pliku zawierającego polecenia do wykonywania:

```
crontab /mnt/mtd/crontab
```

4. Uruchomić demona *Cron*d poleceniem *crond*

Przykładowo wpisując do pliku *crontab*:

```
5 0 * * * /mnt/mtd/ntpdate ntp1.us.edu.pl
```

Codziennie o godzinie 00:05 uruchomione zostanie polecenie aktualizujące datę na podstawie zegaru sieciowego *ntp1.us.edu.pl*



Polskie publiczne serwery czasu

- tempus1.gum.gov.pl [212.244.36.227] (atomowy zegar, Główny Urząd Miar)
- tempus2.gum.gov.pl [212.244.36.228] (atomowy zegar Główny Urząd Miar)
- ntp.task.gda.pl

Światowe publiczne serwery czasu:

- tick.ucla.edu
- clock.isc.org
- utcnist.colorado.edu

Rozdział 4 Konfiguracja i zarządzanie interfejsami komunikacyjnymi

4.1. Telnet / FTP

Za komunikację *telnet* i *ftp* odpowiadają demony *telnetd* oraz *proftpd* działające w systemie. Zarządzanie demonami zostało opisane w rozdziale Zarządzanie procesami. Podstawowa konfiguracja została również zawarta w rozdziale Konfiguracja Startowa.

4.2. DNS

DNS (ang. Domain Name System, system nazw domenowych) jest usługą zapewniającą zamianę adresów logicznych (numery IP) na ich adresy mnemoniczne. Usługa DNS warstwy aplikacji modelu TCP/IP, jest związana z portem 53 TCP/UDP. Dzięki zastosowaniu tej usługi ograniczamy koszt utrzymania stałego adresu IP dla połączenia GPRS.

DynDNS jest darmowym, dynamicznym serwerem DNS oferującym użytkownikom darmowe domeny. Aby dodać nazwę urządzenia sieciowego do serwera DynDNS należy początkowo założyć konto na stronie www.dyndns.com.

Podczas logowania należy wybrać nazwę serwera oraz podać przydzielony adres IP urządzenia sieci GPRS (polecenie *gprs status* przy aktywnym połączeniu).



Ważne jest aby korzystać z karty dedykowanej połączeniom GPRS (wykupione usługi), której przydzielany jest dynamiczny, zewnętrzny adres IP. Szczegółów dotyczących opcji połączenia należy szukać u wybranego operatora sieci.

Hostname:	<input type="text" value="npe"/> - <input type="text" value="homelinux.org"/>
Wildcard Status:	Disabled [Want Wildcard support?]
Service Type:	<input checked="" type="radio"/> Host with IP address [?] <input type="radio"/> WebHop Redirect [?] <input type="radio"/> Offline Hostname [?]
IP Address:	<input type="text" value="77.113.187.7"/> Your current location's IP address is 89.79.77.168 TTL value is 60 seconds. Edit TTL .

Kolejnym krokiem jest pobranie z repozytorium skompilowanego pakietu *Inadyn* (dedykowanego pod architekturę ARM), który umożliwia aktualizację numeru IP nadanego dynamicznie przy kolejnym zalogowaniu do sieci GPRS.

Po pobraniu pakietu należy rozpakować go na dysku twardym i z wykorzystaniem dowolnego

klienta FTP wgrać plik *inadyn* do dowolnego katalogu w systemie NPE.



Zalecane jest skopiowanie pliku do katalogu `/mnt/mtd` lub na karcie SD.

Jednorazowe wywołanie pakietu wymaga podania atrybutów:

```
inadyn -u <Nazwa użytkownika> -p <hasło> --update_period_sec <interwał  
czasowy> --dyndns_system dyndns@dyndns.org --alias <nazwa domeny>
```



Niezalecane jest ustalanie interwału odświeżania adresu IP urządzenia na mniejszy niż 600s.



Istnieje możliwość wczytywania atrybutów konfiguracyjnych z pliku. Generator pliku *inadyn.conf* oraz opis użycia znajduje się pod adresem:

<https://www.dyndns.com/support/tools/clientconfig.html>

Po wygenerowaniu zawartości należy wpisać dane wybrane podczas rejestracji(**login i hasło**).

Dzięki wykorzystaniu pliku konfiguracyjnego *inadyn* może działać w tle(jako demon). Przykładowy plik konfiguracyjny może wyglądać następująco:

```
## inadyn configuration file  
update_period 60000 # Check for a new IP every 60 seconds  
# DynDNS username and password here  
--username username  
--password password  
--dyndns_system dyndns@dyndns.org  
## Dynamic DNS hosts  
--alias npe.homeunix.com  
## Like demon  
--background
```



Uruchomienie klienta *inadyn* bez podania atrybutów automatycznie poszukuje pliku *inadyn.conf* w katalogu `/etc/` którego zawartość nie jest zachowywana po restarcie systemu. W tym celu należy zapisać plik konfiguracyjny przykładowo na karcie SD a do pliku *rfs* dodać wiersz kopiujący plik do określonego katalogu i uruchamiający pakiet, np.

```
cp /mnt/sd/inadyn.conf /etc/  
./inadyn
```

Przykładowe wykorzystanie klienta *inadyn*:

```
$ gprs status  
NPE GPRS Connection Manager Utility [Version 1006171022]  
$ gprs status  
NPE GPRS Connection Manager Utility [Version 1006171022]  
Ping secondary address: Succeeded  
STATUS: CONNECTED  
PPP IP: 77.112.181.147  
$ ./inadyn -u username -p password --dyndns_system  
dyndns@dyndns.org --alias npe.homeunix.com
```

W efekcie:

```
Telnet 192.168.0.140
$ ./inadyn -u rjaloza -p romjal0717 --dyndns_system dyndns@dyndns
.org --alias npe.homeunix.com
INADYN: Started 'INADYN version 1.96.2' - dynamic DNS updater.
I:INADYN: IP address for alias 'npe.homeunix.com' needs update to
'87.251.240.150'
I:INADYN: Alias 'npe.homeunix.com' to IP '87.251.240.150' updated
successful.
```

Po zakończeniu działania dynamicznie przyznany IP: 87.251.240.150 adres jest widziany w sieci pod nazwą mnemoniczną: *npe.homeunix.com*.

Dla sprawdzenia można wysłać ramkę ICMP poleceniem ping:

```
C:\>ping npe.homeunix.com
Badanie npe.homeunix.com [87.251.240.150] z użyciem 32 bajtów danych:
Odpowiedź z 87.251.240.150: bajtów=32 czas=2180ms TTL=52
Odpowiedź z 87.251.240.150: bajtów=32 czas=478ms TTL=52
```

4.3. Apache

Fabryczna konfiguracja urządzenia NPE nie zawiera zainstalowanego serwera Apache z PHP. Jeżeli zależy nam na takiej funkcjonalności możemy je manualnie zainstalować.

Zainstalowanie serwera Apache wymaga:

1. Rozpakowania pakietu *apache.tar.gz* do katalogu */mnt/sd*. Po rozpakowaniu Apache zajmuje około 6,7MB i musimy sprawdzić czy posiadamy wystarczająco wolnego miejsca(np. poleceniem *df -h*).
2. Dodania do pliku *rcs* tworzenia odwołań symbolicznych do odpowiednich katalogów:

```
ln -s f /mnt/sd/apache/apache2/bin/* /bin
ln -s -f /mnt/sd/apache/apache2/lib/* /lib
```

3. Dodać uprawnienia do wykonywania:

```
chmod +x /mnt/sd/apache/apache2/bin
```

4. Uruchomić serwer Apache z PHP5

```
apachectl start
```

Jeżeli nie wystąpiły żadne błędy serwer został uruchomiony poprawnie.



Plik konfiguracyjny znajduje się w katalogu */mnt/sd/apache2/conf/httpd.conf*

Domyślna konfiguracja :

```
DocumentRoot "/mnt/sd/apache2/htdocs"
ServerName 192.168.0.101:80
```

W celu przetestowania serwera możemy uruchomić dowolną przeglądarkę internetową i w polu adresu wpisać adres IP podany w pliku konfiguracyjnym w polu *ServerName*.

Jeżeli chcemy wyłączyć serwer Apache należy użyć komendy:

```
apachectl stop
```



Jeżeli chcemy korzystać z serwera cały czas warto dodać wywołanie do pliku `/mnt/mtd/rcs` aby serwer był uruchamiany każdorazowo przy starcie systemu. Do pliku `rcs` należy dodać:

```
/mnt/mtd/nfs/apache  
/apachectl start
```

4.4. PHP

Korzystanie z języka PHP wymaga uruchomionego serwera Apache. W powyższym rozdziale znajduje się opis instalacji serwera wspierającego PHP5. Zatem uruchomienie serwera Apache umożliwia korzystanie ze skryptów napisanych w języku PHP.

4.5. IPTABLES

IPTABLES jest narzędziem zarządzającym filtrami pakietów IP w systemie Linux. Program może być używany jako filtr ramek, bądź tzw. stanowa zapora systemu, kontrolująca połączenia wchodzące i wychodzące do sieci komputerowej lub stacji roboczej.



Strona projektu Netfilter zawierającego dokumentację i przykłady użycia pakietu iptables znajduje się pod adresem:

<http://www.netfilter.org/documentation/index.html#documentation-howto>

Definiowanie filtrów

Inicjowanie filtrów dotyczących ruchu sieciowego(niezależnie od protokołu) umożliwia polecenie

```
# iptables [-t tables] [-P] [INPUT, OUTPUT, FORWARD, PREROUTING, OUTPUT,  
POSTROUTING][ACCEPT, DROP]
```



Główne atrybuty:

- INPUT - opisuje działania dla pakietów przychodzących,
- OUTPUT – pakietów wychodzących,
- FORWARD – dla pakietów przechodzących pomiędzy interfejsami.
- PREROUTING – dla pakietów przychodzących przed routowaniem
- POSTROUTING – dla pakietów wychodzących po routowaniu.
- ACCEPT (zaakceptowanie pakietu),
- DROP (usunięcie)
- REJECT (odrzućcie z powiadomieniem nadawcy).

Dodanie filtra dla konkretnego protokołu umożliwia polecenie:

```
# iptables [-t table] [-AI] [INPUT, OUTPUT, FORWARD] [-io interface]
```

```
[-p tcp, udp, icmp, all] [-s IP/network] [--sport ports] [-d IP/network]
[--dport ports] -j [ACCEPT. DROP]
```



Atrybuty:

- A dodanie nowej reguły na końcu obecnego łańcucha
- I dodanie więcej niż jednej reguły dla danego łańcucha
- i nazwa interfejsu wejściowego
- o nazwa interfejsu wyjściowego
- p protokół, który ma być filtrowany w danym łańcuchu
- s adres źródłowy
--sport numer portu źródłowego
- d adres docelowy
--dport port docelowy
- j definiuje akcje dla pakietów pasujących do filtra np. ACCEPT, DROP lub LOG.

Przykładowe zastosowanie:

- Akceptowanie wszystkich pakietów z interfejsu wewnętrznego lo

```
# iptables -A INPUT -i lo -j ACCEPT
```

- Akceptowanie pakietów TCP wysyłanych z adresu 192.168.0.1

```
# iptables -A INPUT -i eth0 -p tcp -s 192.168.0.1 -j ACCEPT
```

- Akceptowanie pakietów TCP wysyłanych adresów sieciowych klasy C 192.168.1.0/24

```
# iptables -A INPUT -i eth0 -p tcp -s 192.168.1.0/24 -j ACCEPT
```

- Odrzucanie pakietów TCP wysyłanych z adresu 192.168.1.25

```
# iptables -A INPUT -i eth0 -p tcp -s 192.168.1.25 -j DROP
```

- Odrzucanie pakietów TCP adresowanych na port 21

```
# iptables -A INPUT -i eth0 -p tcp --dport 21 -j DROP
```

- Akceptowanie pakietów TCP wysyłanych z adresu 192.168.0.20 na port 130,...,138,139

```
# iptables -A INPUT -i eth0 -p tcp -s 192.168.0.24 --dport 130:139
-j ACCEPT
```

- Odrzucanie wszystkich pakietów wysyłanych z urządzenia o adresie MAC 11:22:33:44:55:66

```
# iptables -A INPUT -i eth0 -p all -m mac --mac-source 11:22:33:44:55:66
-j DROP
```

- Usunięcie konkretnego istniejącego filtra w iptables

```
# iptables -D INPUT -i eth0 -p tcp --dport 21 -j DROP
```

- Usunięcie wszystkich filtrów w łańcuchu INPUT iptables

```
# iptables -F INPUT
```



Przy korzystaniu z wielu reguł należy pamiętać o kolejności ich dodawania: od najbardziej szczegółowych(filtry dla pojedynczych protokołów) do najbardziej ogólnych(filtry dla całych łańcuchów).

Zapisanie filtrów w autostarcie

Efektywne zarządzanie filtrami wymaga przechowywania ustalanych reguł. Zalecane jest zapisywanie reguł do pliku przy każdorazowej poprawnej i ostatecznej zmianie filtracji. W tym celu w systemie NPE wkompiłowane zostały pakiety *iptables-save* oraz *iptables-restore*.

Zapis konfiguracji do pliku odbywa się za pomocą:

```
iptables-save > <ścieżka do pliku>
```



Zalecane jest zapisywanie plików konfiguracyjnych w:

- pamięci Flash: */mnt/mtd*
- na karcie SD: */mnt/sd*.

Pliki w innych lokalizacjach nie są zapamiętywane po restarcie systemu.

Wczytanie konfiguracji iptables z pliku umożliwia polecenie:

```
iptables-restore < <ścieżka do pliku>
```

Przykładowo dodamy kilka reguł do iptables:

```
$ iptables -A INPUT -i eth0 -p icmp -s 0/0 -j REJECT
$ iptables -A INPUT -i ppp0 -p tcp -s 0/0 -j DROP
$ iptables -A OUTPUT -p icmp -s 0/0 -j REJECT
```

Cała tablica wygląda następująco:

```
$ iptables -nL
Chain INPUT (policy ACCEPT 1934 packets, 129K bytes)
 pkts bytes target    prot opt in     out     source destination
  3    297 REJECT    icmp -- eth0    *       0.0.0.0/0 0.0.0.0/0
  0      0 DROP     tcp  -- ppp0    *       0.0.0.0/0 0.0.0.0/0
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source destination
Chain OUTPUT (policy ACCEPT 1179 packets, 64806 bytes)
 pkts bytes target    prot opt in     out     source destination
  0      0 REJECT    icmp -- *       *       0.0.0.0/0 0.0.0.0/0
```

Następnie zapisujemy konfigurację do pliku:

```
$ iptables-save > /mnt/mtd/iptables.conf
```

W celu wczytania zapisanych ustawień wykorzystujemy polecenie:

```
$ iptables-restore < /mnt/mtd/iptables.conf
```



Warto dodać do pliku *rcs* komendę wczytującą konfigurację *iptables* lub stworzyć osobny skrypt. Dzięki temu przy ponownym uruchomieniu systemu nie utracimy ustawionych reguł.

4.6. NAT

NAT (Network Address Translation) jest protokołem, który umożliwia translację adresów IP podsieci przy pomocy adresu IP innej sieci. NAT bardzo często znajduje zastosowanie w rozbudowanej architekturze sieciowej. Główną potrzebą wykorzystania translacji jest chęć zdalnego dostępu poprzez globalną sieć do dowolnego hosta w podsieci.

Urządzenie NPE zazwyczaj monitoruje, zarządza zasobami wewnątrz podsieci. Dzięki mapowaniu lokalnego adresu sieci na zewnętrzny globalny adres IP oraz tłumaczeniu globalnego adresu do

adresu lokalnego można z dowolnego miejsca zdalnie zarządzać urządzeniem NPE. Translacja adresów w systemie NPE wykorzystuje omówione wcześniej narzędzie jakim jest pakiet *iptables*.

W poniższych przykładach omówione zostaną dwa główne zastosowania NATu:

- przekierowanie portów wewnątrz urządzenia
- przekierowanie na inny adres
- udostępnienie połączenia z internetem za pomocą GPRS



Oznaczenia wykorzystywane w tym rozdziale:

- < adres IP > adres IP urządzenia NPE, na którym konfigurujemy przekierowanie
- < adres IP docelowy > adres IP urządzenia, na które przekierowujemy ruch
- < port wejściowy > przekierowywany numer portu
- < port docelowy > numer portu, na który ustawiane jest przekierowanie

Przekierowanie wewnętrznych portów

Pakiet *iptables* umożliwia przekierowanie ruchu z danego portu na inny. Na potrzeby instrukcji zajmiemy się przypadkiem przekierowania pakietu wchodzącego na port 6060, umożliwiając na nim nawiązanie połączenia telnetowego oraz przekierowanie portu 8080 w celu udostępnienia połączenia FTP.

Przekierowanie portu telnetowego

Protokół Telnet łączy się domyślnie z portem 23 serwera. Naszym celem jest dodanie reguł w *iptables* tak aby pakiet wchodzący na zewnętrzny port 6060 był automatycznie przekierowany na wewnętrzny port 23.

W tym celu należy dodać reguły:

- Odblokowania opcji przekierowania w jądrze systemu

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

- Ustawienia globalnej polityki bezpieczeństwa

```
iptables -P INPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -P OUTPUT ACCEPT
```

- Otworzenie portu wejściowego na urządzeniu NPE

```
iptables -A FORWARD -p tcp -d < adres IP > --dport <port wejściowy>
-j ACCEPT
```

- Otworzenie portu docelowego na urządzeniu NPE

```
iptables -t nat -A PREROUTING -p tcp -d < adres IP > --dport <port
wejściowy> -j DNAT --to <adres IP>:<port docelowy>
```

Przykładowo:

```
$ echo 1 > /proc/sys/net/ipv4/ip_forward
$ iptables -P INPUT ACCEPT
$ iptables -P FORWARD ACCEPT
$ iptables -P OUTPUT ACCEPT
$ iptables -A FORWARD -p tcp -d 192.168.0.113 --dport 6060 -j ACCEPT
$ iptables -t nat -A PREROUTING -p tcp -d 192.168.0.113 --dport 6060
-j DNAT --to 192.168.0.113:23
```

W efekcie możemy się zalogować na urządzeniu NPE poprzez telnet na porcie 6060:

```
>telnet 192.168.0.113 6060
```



Przekierowanie portu FTP

Protokół FTP domyślnie używa portów 20 i 21 do połączenia z serwerem. Jeżeli FTP pracuje w trybie aktywnym, korzysta z portów: 21 do wysyłania poleceń (połączenie to jest zestawiane przez klienta) oraz 20 do przesyłu danych. Jeżeli FTP pracuje w trybie pasywnym wykorzystuje port 21 do poleceń i port o numerze > 1024 do transmisji danych.



Do komunikacji z urządzeniem NPE zalecane jest użycie pasywnego połączenia, wykorzystującego do sterowania port 21.

Umożliwienie przekierowanie połączenia FTP z portu 8080 na port 21 realizujemy analogicznie jak dla połączenia telnetowego, zmieniając jedynie numery portów w regułach:

- otwarcia portu wejściowego na urządzeniu NPE

```
iptables -A FORWARD -p tcp -d < adres IP > --dport 8080 -j ACCEPT
```

- Otworzenie portu docelowego na urządzeniu NPE

```
iptables -t nat -A PREROUTING -p tcp -d < adres IP > --dport 8080  
-j DNAT --to <adres IP>:21
```

Przekierowanie ruchu na inny adres IP

Wykorzystując pakiet *iptables* możemy umożliwić dostęp do innego urządzenia sieciowego poprzez przekierowanie na inny adres IP. Przykładowo dostosujemy reguły tak, by połączenie z urządzeniem na porcie 2020 zostało przekierowane na port 23 drugiego urządzenia NPE.



Oznaczenia wykorzystywane w tym rozdziale:

- < adres IP > adres IP urządzenia NPE, na którym konfigurujemy przekierowanie
- < adres IP docelowy > - adres IP urządzenia, na które przekierowujemy ruch

W tym celu:

- Odblokowujemy możliwość przekierowań w jądrze systemu

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

- Ustawiamy globalną politykę bezpieczeństwa (o ile nie została już zdefiniowana)

```
iptables -P INPUT ACCEPT  
iptables -P FORWARD ACCEPT  
iptables -P OUTPUT ACCEPT
```

- Otwieramy wejściowy port na urządzeniu NPE

```
iptables -A FORWARD -p tcp -d < adres IP > --dport 2020 -j ACCEPT
```

- Ustawiamy docelowy adres i port na urządzeniu NPE

```
iptables -t nat -A PREROUTING -p tcp -d < adres IP > --dport 2020 -j DNAT  
--to <adres IP docelowy>:23
```

- Aktywujemy translację adresów(maskaradę) dla pakietów routowanych.

```
iptables -t nat -A POSTROUTING -p tcp -j MASQUERADE
```

Po wykonaniu tych czynności połączenie z urządzeniem NPE na porcie 2020 jest przekierowywane na port 23 innego urządzenia.

Udostępnienie połączenia internetowego poprzez GPRS

Urządzenie NPE opcjonalnie wyposażone jest w moduł GSM. Istnieje zatem możliwość przystosowania systemu NPE do udostępniania połączeń GPRS innym urządzeniom i pełnienia roli bramy sieciowej oddzielającej lokalną sieć od internetu.

Na potrzeby tego rozdziału zakładamy, że urządzenie posiada skonfigurowane i aktywne połączenie *ppp0* (poprzez modem GPRS). Szczegóły jak nawiązać połączenie znajdują się w rozdziale: Połączenie GPRS.

Parametry połączenia *ppp0* wykorzystanego w tym rozdziale:

```
ppp0      Link encap:Point-to-Point Protocol  
          inet addr:87.251.253.95  P-t-P:10.0.0.1  Mask:255.255.255.255  
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1  
          RX packets:112 errors:0 dropped:0 overruns:0 frame:0  
          TX packets:105 errors:0 dropped:0 overruns:0 carrier:0  
          collisions:0 txqueuelen:3  
          RX bytes:7344 (7.1 KiB)  TX bytes:7892 (7.7 KiB)
```

Konfiguracja pakietu *iptables* jest analogiczna jak dla przekierowania numeru IP. Dodatkowo należy zdefiniować interfejsy sieciowe wejścia(atrybut -i)-wyjścia(atrybut -o).



Standardowo interfejsem łączącym urządzenie z internetem jest interfejs *ppp0* a wewnętrznym w sieci LAN *eth0*.

Reguły konfiguracji komunikacji i przekierowania pomiędzy interfejsami:

```
echo 1 > /proc/sys/net/ipv4/ip_forward  
iptables -t nat -D POSTROUTING -o ppp0 -j MASQUERADE  
iptables -A FORWARD -i eth0 -o ppp0 -j ACCEPT  
iptables -A FORWARD -i ppp0 -o eth0 -m state --state RELATED,ESTABLISHED  
-j ACCEPT
```

Tak skonfigurowane przekierowanie przekazuje cały ruch pomiędzy sieciami na danym interfejsie.

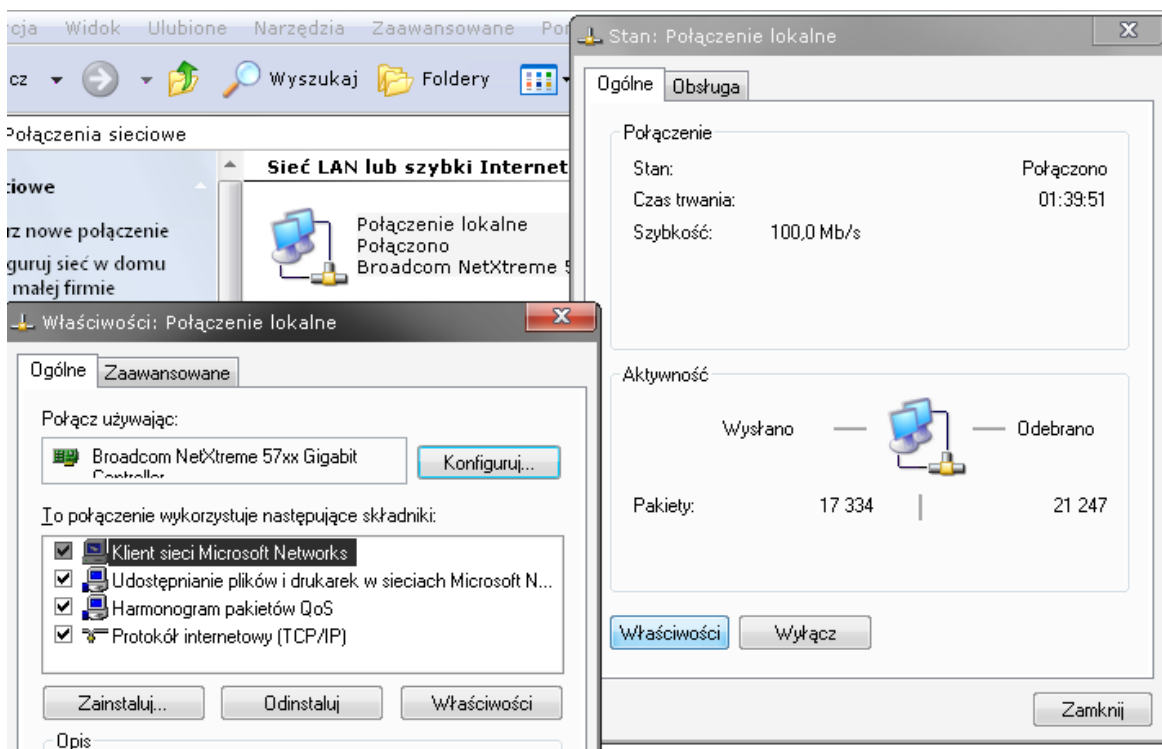


Jeżeli zależałoby nam na zezwoleniu połączenia tylko dla jednego konkretnego hosta należało by uzupełnić łańcuch o atrybuty:

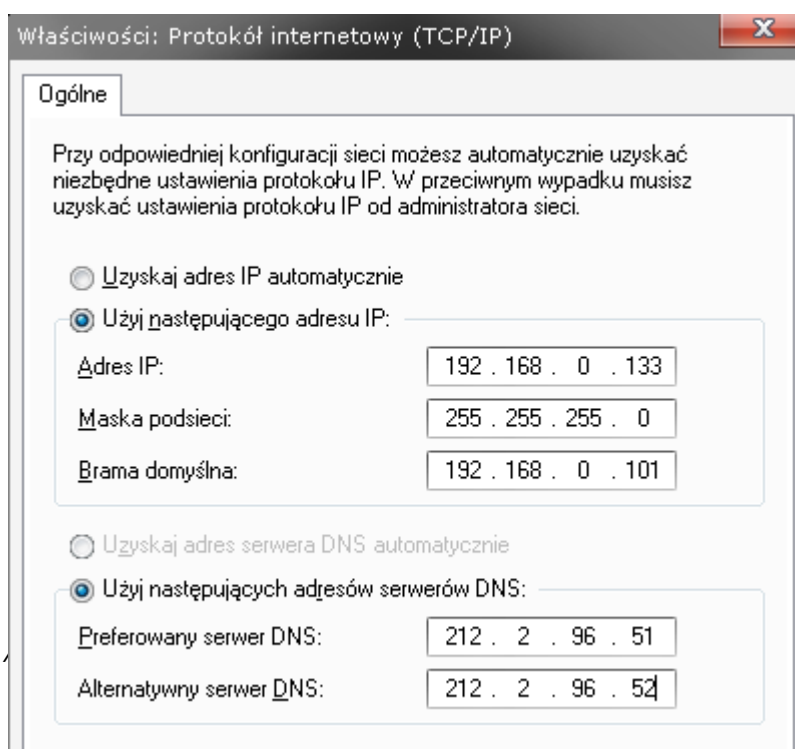
```
iptables -A FORWARD -i eth0 -s <IP komputera> -o ppp0 -j ACCEPT  
iptables -A FORWARD -i ppp0 -o eth0 -d <IP komputera> -m state --state  
RELATED,ESTABLISHED -j ACCEPT
```

Kolejnym krokiem jest skonfigurowanie ustawień połączenia sieciowego komputera. Zmiany wymagają:

- Zmiana domyślnej bramy sieciowej (gateway) na adres IP urządzenia NPE. W tym celu wybieramy:
 - Start > Panel Sterowania > Połączenia Sieciowe > Połączenie Lokalne > Właściwości



- Protokół internetowy (TCP/IP) > Właściwości



Adres IP komputera

(adres musi być w tej samej podsieci co urządzenie NPE)
192.168.0.XXX(domyślnie)

Maska podsieci:

255.255.255.0(domyślnie)

Brama domyślna

Adres IP urządzenia NPE
192.168.0.101(domyślnie)

DNS

Ustawienia DNS nie są niezbędne. Jednak zalecane jest dopisanie adresów zawartych w pliku `/etc/resolv.conf`.



Jeżeli nie zdefiniujemy serwerów DNS w przeglądarce internetowej będziemy musieli posługiwać się adresami logicznymi(adresy IP serwerów) zamiast nazw np `www.a2s.pl`

Po zatwierdzeniu zmian i wyjściu z panelu sterowania połączenie zostanie skonfigurowane i użytkownik na komputerze lokalnym ma dostęp do sieci Internet poprzez modem GSM urządzenia NPE.

4.7. Połączenie GPRS

Zarządzanie połączeniem GSM odbywać się może przy pomocy skryptu konfiguracyjnego *gprs* bądź jednostkowych poleceń i komend AT. Dzięki wykorzystaniu skryptu konfiguracja modemu, nawiązanie połączenia oraz utworzenie systemowego interfejsu komunikacyjnego odbywa się automatycznie – użytkownik nie musi wpływać na poszczególne etapy konfiguracji. Dodatkowo skrypt domyślnie wykorzystuje multipleksację portu(szczegółowy opis w podrozdziale Multipleksacja portu szeregowego)

Zarządzanie połączeniem – skrypt *gprs*

Skrypt *gprs* został stworzony w celu ułatwienia konfiguracji i obsługi modemu GSM i połączeń GPRS. Konfigurację przeprowadza się z wykorzystaniem atrybutów:

Atrybut	Opis
<code>gprs connect [force[:%timeout%]]</code>	Nawiązanie połączenia z siecią GPRS. Opcja <i>force</i> <czas s> wymusza wielokrotnie próbę połączenia przez określony czas.
<code>gprs disconnect</code>	Zamykanie aktywnego połączenia GPRS
<code>gprs reconnect [on off]</code>	Umożliwienie automatycznego restartowania modemu w przypadku braku połączenia (pingi nie docierają do serwera)
<code>gprs status [nocheck]</code>	Odczytanie aktualnego statusu urządzenia. W przypadku aktywnego połączenia wyświetlany jest numer IP nadany w sieci GPRS.
<code>gprs modemoff</code>	Zamykanie aktywnego połączenia i wyłączenie modemu.
<code>gprs abort</code>	Wymuszenie zakończenia działania procesu <i>reconnect</i> .



Wyłączenie domyślnej ultipleksacji portu szeregowego w przypadku nawiązanego połączenia GPRS wymaga ponownego połączenia z siecią(*gprs reconnect*). Opis wyłączenia multipleksacji znajduje się w poniższym rozdziale.

Multipleksacja portu szeregowego

Dla pełnego wykorzystania funkcjonalności modułu GSM przydatna jest możliwość jednoczesnego

używania połączenia PPP i np. wysyłania komunikatów SMS. Nie jest to możliwe przy użyciu tylko jednego portu szeregowego. Rozwiązaniem tego problemu jest użycie multipleksacji portu szeregowego.



W systemie NPE włączenie/wyłączenie multipleksacji dla portu ttyS1 umożliwiają skrypty muxon, muxoff

Po włączeniu multipleksacji port ttyS1 staje się niedostępny a w jego miejsce powstają trzy wirtualne porty: mux1, mux2, mux3

```
$ ls -l /dev | grep mux
lrwxrwxrwx 1 root root 10 Jan 13 22:48 mux0 -> /dev/pts/0
lrwxrwxrwx 1 root root 10 Jan 13 22:48 mux1 -> /dev/pts/1
lrwxrwxrwx 1 root root 10 Jan 13 22:48 mux2 -> /dev/pts/2
lrwxrwxrwx 1 root root 9 Jan 13 22:48 ttyS5 -> /dev/mux0
lrwxrwxrwx 1 root root 9 Jan 13 22:48 ttyS6 -> /dev/mux1
lrwxrwxrwx 1 root root 9 Jan 13 22:48 ttyS7 -> /dev/mux2
```



Przy korzystaniu z multipleksacji portu szeregowego należy unikać wysyłania komend AT na urządzenie ttyS1, gdyż jest ono nieaktywne i grozi zablokowaniem modemu GSM.

Ewentualne komendy AT wysyłamy na jeden z wirtualnych portów:

ttyS5(mux1), ttyS6(mux2), ttyS7(mux3)

Przykładowo:

```
modem /dev/mux0 115200 "at+cpin?"
```

```
$ modem /dev/mux0 115200 "at+cpin?"
at+cpin?
+CPIN: SIM PIN
OK
```

Konfiguracja rozkazami AT

Modemy GSM sterowanie są za pomocą tekstowych komend o określonej składni. Polecenia w zależności od modemu mogą się nieznacznie różnić, dlatego zalecane jest przeglądanie dokumentacji technicznej wybranego modelu.



Urządzenia NPE opcjonalnie wyposażone są w modemy SIM300 oraz SIM600. Dokumentacja techniczna modułów producenta SIMCOM dostępna jest po zarejestrowaniu się na stronie: <http://wm.sim.com/>

Moduły GSM można w prosty sposób sterować komendami AT.

Wygodny dostęp do sterowania modemem umożliwia specjalny program modem, który umożliwia wysłanie pojedynczej komendy AT.

```
modem <device> <baud rate> "<AT command>"
```



Domyślnie modem GSM widoczny jest w systemie jako /dev/ttyS1

```
$ ls -l /dev | grep ttyS
crw-rw---- 1 root root 4, 64 Jan 1 1998 ttyS0
crw-rw---- 1 root root 4, 65 Jan 13 23:07 ttyS1
crw-rw---- 1 root root 4, 66 Jan 1 1998 ttyS2
crw-rw---- 1 root root 4, 67 Jan 1 1998 ttyS3
crw-rw---- 1 user users 4, 68 Jan 13 22:58 ttyS4
lrwxrwxrwx 1 root root 10 Jan 13 23:07 ttyS5 -> /dev/ttyS1
```



Przy wysyłaniu komend AT do modemu pole <device> uzupełniamy nazwą urządzenia /dev/ttyS1.

Przykładowo w celu sprawdzenia stanu karty SIM użyjemy polecenia:

```
modem /dev/ttyS1 115200 "at+cpin?"
```

Typowa odpowiedź przy poprawnym skonfigurowaniu połączenia to:

```
$ modem /dev/ttyS1 115200 "at+cpin?"
at+cpin?
+CPIN: SIM PIN
OK
```

Odpowiedź ta oznacza, że modem został poprawnie załadowany do sieci.

Wysyłanie wiadomości SMS

Modem GPRS znajduje wykorzystanie przy wysyłaniu wiadomości SMS na skutek określonego zdarzenia. W poniższym rozdziale przedstawiony zostanie sposób wysyłania wiadomości SMS przy użyciu aplikacji *modem* wraz z komendami AT. Do obsługi modemu można wykorzystać dowolny program do komunikacji sieciowej/szeregowej np. HyperTerminal, telnet.

Aby wysłać wiadomość należy:

- Sprawdzić czy modem reaguje na komendy AT

```
modem /dev/mux1 115200 <komenda AT>
```

```
$ modem /dev/mux1 115200 AT
AT
OK
```

- Ustawienie opcji wpisywania treści SMS w konsoli

```
modem /dev/mux1 115200 <komenda AT+CMGF= 0/1>
```

```
$ modem /dev/mux1 115200 AT+CMGF=1
AT+CMGF=1
OK
```

- Wywołanie tworzenia wiadomości SMS pod określony numer.

```
modem /dev/mux1 115200 <komenda AT+CMGS= "numer telefonu">
```

```
$ modem /dev/mux1 115200 AT+CMGS=\ "48600123123\"
AT+CMGS=\ "48600123123"
>
```

- Wysyłanie treści wiadomości poprzez:

```
modem /dev/mux1 115200 "treść wiadomości z CTRL+Z"
```



Używając komendy `AT+CMGS` należy pamiętać by po zakończeniu wpisywania treści wiadomości nacisnąć `CTRL+Z`. Zależnie od systemu i programu poprzez, który komunikujemy się (telnet, PuTTY) kombinacja klawiszy może dopisać znak na końcu polecenia (nie będzie widoczny w treści wiadomości).

- Sprawdzenie poprawności wysłania:

```
modem /dev/mux1 115200 AT
```

Przy poprawnym utworzeniu i wysłaniu efekt powinien być następujący:

```
$ modem /dev/mux1 115200 AT+CMGS=\ "48600123123\"
AT+CMGS="48600123123"
>
$ modem /dev/mux1 115200 "Hello GSM World"
Hello GSM World
$ modem /dev/mux1 115200 AT
+CMGS: 98
OK
```

4.8. NFS (sieciowy system plików)

Przydatne staje się udostępnienie katalogu `/home/nfs/` poprzez NFS przykładowo dla przyspieszenia wysyłania skompilowanych aplikacji do urządzenia NPE. W związku z tym wymagane jest zainstalowanie i poprawne skonfigurowanie pakietu `nfs-kernel-server` na komputerze PC.



Zalecane jest pobranie pakietu `nfs-kernel-server` dla wykorzystywanej dystrybucji systemu Linux. W przypadku systemu Ubuntu możliwe jest wyszukanie i zainstalowanie pakietu programem *Synaptic Package Manager*.



Należy się upewnić czy katalog `/home/nfs/` znajduje się w systemie oraz czy posiadamy uprawnienia do odczytu/zapisu.

W celu konfiguracji NFS należy:

1. Otworzyć plik `/etc/exports` w edytorze tekstowym:

```
$sudo gedit /etc/exports
```

2. Dodać nową linię na końcu pliku:

```
/home/nfs 192.168.0.101(rw, no_root_squash)
```

Następnie zapisać zmiany.



Domyślny numer IP urządzenia NPE to: 192.168.0.101. Jeżeli urządzenie posiada inny numer IP należy to uwzględnić i wpisać aktualny.

3. Zmiany w konfiguracji zostaną zatwierdzone po wykonaniu poleceń:

```
$sudo exportfs -a
$sudo /etc/init.d/portmap restart
$sudo /etc/init.d/nfs-common restart
$sudo /etc/init.d/nfs-kernel-server restart
```

4. W celu przetestowania wprowadzonych zmian należy wpisać:


```
$sudo showmount -e
```

W efekcie powinniśmy uzyskać:

```
$ sudo showmount -e
Export list for desktop:
/home/nfs 192.168.0.140
romanus@michal-desktop:/$
```



Czasami wymagane jest ustawienie statycznego numeru portu dla *mountd* i *status*. Należy to wykonywać tylko jeżeli pojawiają się problemy z zamontowaniem katalogu */home/nfs* na urządzeniu NPE.

Należy otworzyć plik */etc/default/nfs-common*, odnaleźć linię *STATDOPTS* i zastąpić *STATDOPTS="-port 4000"*.

Następnie otworzyć plik */etc/default/nfs-kernel-server*, wyszukać linię *RPCMOUNT-DOPTS* i zamienić na *RPCMOUNTDOPTS="-p 4002"*

4.9. Mail

Jedną z głównych funkcjonalności systemu NPE jest wysyłanie informacji o monitorowanym systemie. Przy okresowym lub zdarzeniowym raportowaniu danych bardzo często wykorzystywane są protokoły SMTP. System NPE zawiera pakiet *email* dzięki, któremu możemy wysyłać wiadomości mailowe.

Dostępne opcje pakietu *email* uzyskamy wpisując w konsoli polecenie *email* bez atrybutów.

```
Options information is as follows
email [options] recipient1,recipient2,...

-h, -help module          Print this message or specify one of the below options
-V, -verbose              Display mailing progress.
-f, -from-addr             Senders mail address
-n, -from-name             Senders name
-b, -blank-mail            Allows you to send a blank email
-e, -encrypt               Encrypt the e-mail for first recipient before sending
-s, -subject subject      Subject of message
-r, -smtp-server server    Specify a temporary SMTP server for sending
-p, -smtp-port port        Specify the SMTP port to connect to
-a, -attach file           Attach file and base64 encode
-c, -conf-file file        Path to non-default configuration file
-t, -check-config          Simply parse the email.conf file for errors
-x, -timeout               Set socket timeout.
    -cc email,email,...    Copy recipients
    -bcc email,email,...    Blind Copy recipients
    -sign                  Sign the email with GPG
    -html                  Send message in HTML format ( Make your own HTML! )
    -tls                   Use TLS/SSL
-m, -smtp-auth type        Set the SMTP AUTH type (plain or login)
-u, -smtp-user username    Specify your username for SMTP AUTH
-i, -smtp-pass password    Specify your password for SMTP AUTH
-g, -gpg-pass              Specify your password for GPG
-H, -header string         Add header (can be used multiple times)
    -high-priority          Send the email with high priority
    -no-encoding            Don't use UTF-8 encoding
```

Atrybuty wymagane do wysyłania wiadomości email:

```
email -f <mail nadawcy> -n <nazwa nadawcy> -r <serwer smtp nadawcy>  
-u <nazwa użytkownika> -i <hasło mail> -m <sposób autoryzacji>  
-s <tytuł maila> <mail adresata>
```

Po wpisaniu poprawnej komendy zostanie uruchomiony konsolowy edytor *vi*. Po naciśnięciu litery „i” możemy rozpocząć wpisywanie treści wiadomości. Gdy skończymy naciskamy klawisz *Esc* i wpisujemy ciąg „: wq”. Po zatwierdzeniu klawiszem *Enter* wiadomość email zostanie wysłana.

Wysyłanie wiadomości z systemu NPE najczęściej wiąże się z przekazywaniem określonych wartości, zawartości plików, czy informacji obłędach. Jeżeli chcielibyśmy w treści wiadomości zawrzeć zawartość określonego pliku należy komendę email poprzedzić poleceniem „cat <nazwa pliku> |”

```
cat <nazwa pliku> | email -f <mail nadawcy> -n <nazwa nadawcy>  
-r <serwer smtp nadawcy> -u <nazwa użytkownika> -i <hasło mail>  
-m <sposób autoryzacji> -s <tytuł maila> <mail adresata>
```

Przykładowe zastosowanie:

- ręczne wpisanie treści wiadomości

```
$ email -f npe@a2s.pl -n npe -r mail.a2s.pl -u npe@a2s.pl -i npe223322  
-m login -s NPE info@a2s.pl
```

z wykorzystaniem edytora *vi*

```
To jest testowa wiadomosc  
wpisana recznie  
~  
:wq
```

- wpisanie zawartości pliku zewnętrznego

```
$ cat mail_test | email -f npe@a2s.pl -n npe -r mail.a2s.pl -u npe@a2s.pl  
-i npe223322 -m login -s NPE info@a2s.pl
```

Wykonanie obu komend w efekcie wyśle wiadomość mailową o zadanej treści.



Jeżeli chcielibyśmy przy starcie systemu wysyłać wiadomości należy powyższe polecenie zapisać jako skrypt i dodać do autostartu (szczegółowy opis w rozdziale parametry startowe).
Jeżeli zależy nam na okresowo wysyłanych mailach warto dodać skrypt do crona (szczegółowy opis w rozdziale wykonywanie zaplanowanych zadań)

4.10. SNMP

Fabrycznie w systemie NPE uruchomiony jest demon usługi SNMP. Protokół ten umożliwia zarządzanie urządzeniami sieciowymi, takimi jak routery, przełączniki, komputery. Wykorzystując SNMP możemy monitorować urządzenia NPE, tj. porty wejścia, wyjścia, przetworniki A/C.

Przeglądarka SNMP

Nawiązanie komunikacji z urządzeniem NPE przy pomocy komunikatów SNMP wymaga zainstalowanej odpowiedniej przeglądarki na komputerze PC. Na potrzeby tego rozdziału posłużymy się programem MIB browser firmy iReasoning. W procesie zarządzania używane są bazy MIB (ang. Management Information Base - baza informacji zarządzania), czyli zbiory zmiennych,

które manager SNMP w zależności od uprawnień może odczytać lub zmienić w danym urządzeniu.



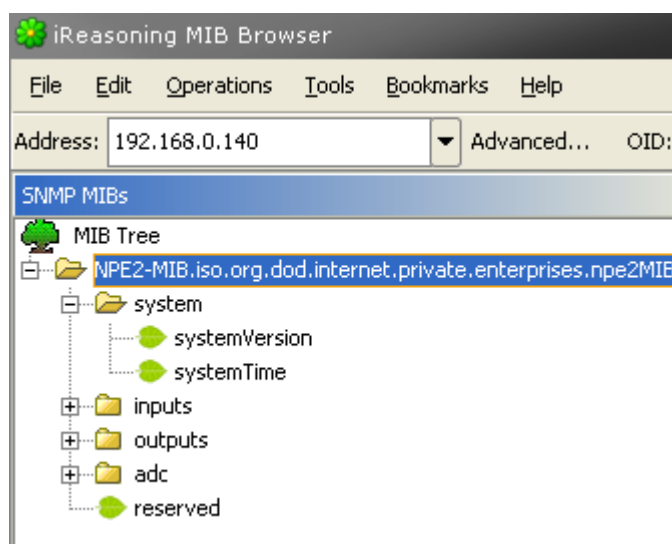
Aktualną wersję MIB Browser możemy pobrać ze strony:
<http://www.ireasoning.com/download.shtml>



W celu dodania bazy MIB dedykowanej dla urządzenia NPE należy w Menu wybrać *File -> Load MIBs* i w menadżerze wybrać lokalizację i plik bazy. Baza znajduje się w systemie NPE w katalogu */etc/snmp/NPE2-MIB.txt* i można ją pobrać za pomocą dowolnego klienta FTP.

Odczyt/Zapis zasobu

Z menu SNMP MIBs rozwijamy interesującą nas bazę. Pojawiły się grupy zasobów (informacje o systemie, wejścia, wyjścia, przetworniki).

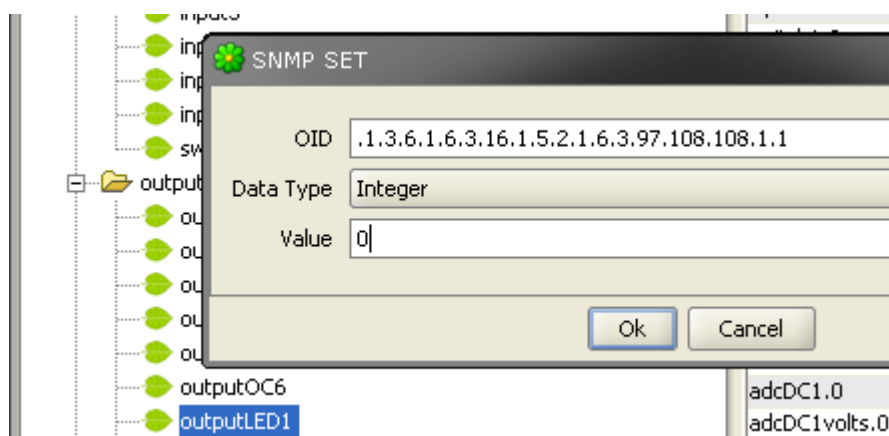


Jeżeli chcemy odczytać wartość danego zasobu klikamy prawym przyciskiem myszy, wybieramy GET (lub skrót klawiszowy *CTRL+G*) i w oknie *Result Table* pojawi się odczytana wartość. Istnieje możliwość wybrania opcji WALK (skrót klawiszowy *CTRL+W*), która wyświetla wszystkie dostępne zasoby.

Result Table	
Name/OID	Value
systemVersion.0	NPE version 4.0
systemTime.0	Wed Jul 14 09:17:00 2010
input1.0	1
input2.0	1
input3.0	0
input4.0	0
input5.0	1
input6.0	1
input7.0	1
input8.0	1
switch1.0	1
outputOC1.0	1
outputOC2.0	1
outputOC3REL1.0	1
outputOC4REL2.0	0
outputOC5.0	1
outputOC6.0	0
outputLED1.0	0
outputLED2.0	0
adcStart.0	1
adcFilter.0	1
adcDC1.0	4
adcDC1volts.0	0.0 V
adcDC2.0	3
adcDC2volts.0	0.0 V
adcDC3.0	1
adcDC3volts.0	0.0 V
adcDC4.0	3
adcDC4volts.0	0.0 V
adcAC1.0	3
adcAC1volts.0	0.1 V

Jeżeli interesuje nas ustawienie danej wartości, np. Logicznej „1” na wyjściu lub zapalenie diody należy klikając prawym przyciskiem wybrać SET (skrót klawiszowy **CTRL+S**) a w kolejnym oknie wybrać format i docelową wartość.

Przykładowo zapalenie diody *UserLED* odbywa się poprzez wybranie *outputLED1* i ustawienie wartości całkowitej 0.



Rozdział 5 Przykładowe aplikacje

5.1. Język JAVA

Pliki źródłowe przykładowych aplikacji można pobrać po zalogowaniu na stronie www.a2s.pl.

Przykład 1: Serwer TCP/IP

Pierwsza przykładowa aplikacja *TCPserver* ukazuje klasyczny program typu TCP echo server. Aplikacja nasłuchuje na określonym przy wywołaniu porcie i odpowiada na każdą nadesłaną odpowiedź odsyłając odebrane wiadomości do nadawcy. Aby przetestować tę aplikację należy połączyć komputer z urządzeniem kablem Ethernet lub podłączyć oba urządzenia do jednej sieci LAN.

Aby w pełni przetestować aplikację należy pobrać również program *TCPclient*, który umożliwia wysyłanie wiadomości na określony adres IP.

Po pobraniu obu aplikacji za pomocą dowolnego klienta FTP wgrywamy plik do wybranego katalogu w urządzeniu NPE.

Uruchomienie programu wymaga podania portu na którym serwer będzie nasłuchiwał nadchodzących wiadomości, tj.:

```
java -cp . TCPserver < numer portu >
```

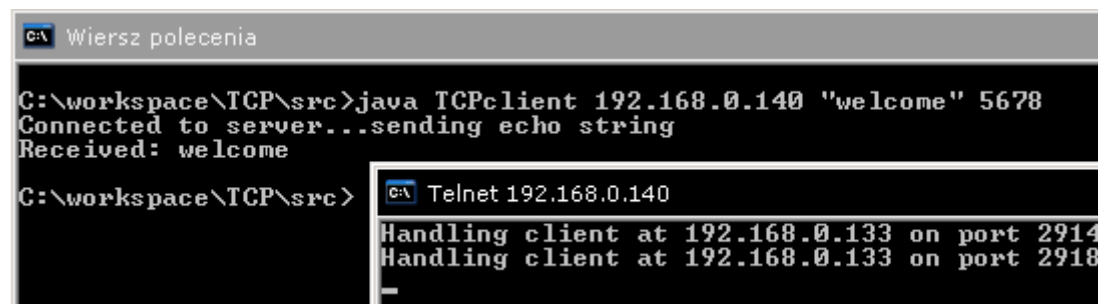
Serwer działa w tle i reaguje tylko w sytuacji, gdy nadejdzie wiadomość na port 5678.

Kolejnym krokiem jest uruchomienie programu *TCPclient* na lokalnym komputerze. Poprawne wywołanie musi zawierać adres IP serwera TCP, treść wiadomości oraz port w postaci:

```
java TCPclient <nr IP serwera> „treść wiadomości” <numer portu>
```

W momencie wywołania:

```
C:\workspace\TCP\src>java TCPserver 5678
Listening on port 5678
```



The screenshot shows a Windows command prompt window titled "Wiersz polecenia". The command prompt shows the following sequence of commands and outputs:

```
C:\workspace\TCP\src>java TCPclient 192.168.0.140 "welcome" 5678
Connected to server...sending echo string
Received: welcome

C:\workspace\TCP\src>
```

In the background, another window titled "Telnet 192.168.0.140" is visible, showing the server's response:

```
Handling client at 192.168.0.133 on port 2914
Handling client at 192.168.0.133 on port 2918
```

Klient uzyskuje odpowiedź od serwera a serwer odnotowuje z jakiego numeru IP oraz portu nadawał klient.

Przykład 2: Klient TCP/IP

Kolejna aplikacja *TCPclient* jest programem typu TCP client, który wysyła wiadomość na określony przy wywołaniu port adresu IP serwera i odczytuje odebrane wiadomości typu echo z serwera. Testowanie aplikacji odbywa się identycznie jak w poprzednim przykładzie.

Aby w pełni przetestować aplikację należy pobrać również program *TCPserver*, który będzie nasłuchiwał wiadomości wysyłanych od klienta.

Po pobraniu obu aplikacji za pomocą dowolnego klienta FTP wgrywamy plik do wybranego katalogu w urządzeniu NPE.



Ważne aby aplikację *TCPserver* uruchomić jako pierwszą a dopiero później testować program *TCPclient*. W innym przypadku komunikacja nie zostanie poprawnie nawiązana. Opis uruchomienia serwera znajduje się w rozdziale: Przykład 1

Uruchomienie programu *TCPclient* wymaga podania adresu IP, wiadomości oraz portu, na którym nasłuchuje serwer. Wywołanie powinno wyglądać następująco:

```
java -cp . TCPclient <nr IP serwera> „treść wiadomości” <numer portu>
```

Uzyskana odpowiedź powinna być analogiczna jak w poprzednim przykładzie.

Przykład 3: Test portu szeregowego

Obecnie komunikacja szeregową jest jedną z najpowszechniejszych metod transmisji danych. Konfiguracja i obsługa portów szeregowych odbywa się poprzez bibliotekę RXTX, która dostarczana jest domyślnie w pakiecie JavVM wraz z urządzeniem NPE. Aplikacje *ShowCOM*, *DualSerialTest*, *Reader*, *Writer* przedstawiają sposób nawiązywania połączeń poprzez RS232 z urządzeniem NPE.

ShowCOM

Program *ShowCOM* wykorzystuje bibliotekę RXTX do odszukania dostępnych portów COM. Po znalezieniu wylistowuje nazwy dostępnych portów szeregowych.

Przykładowe wywołanie:

```
java -cp . ShowCOM
```

Efekt działania w systemach:

Windows:

```
C:\workspace\Serial\bin>java ShowCOM
COM1
LPT1
```

Linux:

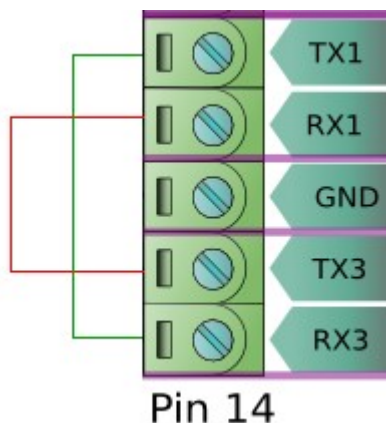
```
C:\ Telnet 192.168.0.140
$ java -cp . ShowCOM
/dev/ttyS7
/dev/ttyS6
/dev/ttyS0
/dev/ttyS1
/dev/ttyS2
/dev/ttyS3
/dev/ttyS4
```



W systemie Linux widoczne są wszystkie porty szeregowo – również te do komunikacji wewnętrznej. Porty COM wyprowadzone i udostępnione dla użytkownika są widziane w systemie jako *ttyS0* i *ttyS2*.

DualSerialTest

Aplikacja testuje porty szeregowo urządzenia NPE wysyłając dane wybranym portem i odbierając kolejnym. Na potrzeby tego programu wymagane jest połączenie portów dowolnymi przewodami w sposób taki jak na rysunku:



Szczegółowy opis złącz znajduje się w rozdziale: Opis złącz.

Kolejnym krokiem jest skopiowanie skompilowanego programu *DualSerialTest.class* do dowolnego katalogu w urządzeniu NPE przy pomocy wybranego klienta FTP.

Uruchomienie aplikacji odbywa się poprzez komendę:

```
java -cp . DualSerialTest <port wysyłający> <port nasłuchujący>
```

Przykładowo uruchamiając aplikację i wpisując *Hello Serial World* efekt powinien być następujący:

```
C:\ Telnet 192.168.0.140
$ java -cp . DualSerialTest /dev/ttyS0 /dev/ttyS2
Sending data via /dev/ttyS0 port
Hello Serial World

Incoming data via /dev/ttyS2 port
Hello Serial World
```

Programy Writer i Reader

Aplikacje Writer i Reader służą do jednostronnej komunikacji wybranymi interfejsami szeregowymi. Dzięki temu możliwe jest uruchomienie programu Reader, nasłuchującego na porcie COM1 lokalnego komputera PC oraz wysyłanie danych z urządzenia NPE poprzez program Writer oraz w odwrotną stronę.



Opis sposobu podłączenia i konfiguracji połączenia szeregowego znajduje się w rozdziale: Połączenie za pomocą portu szeregowego.

Sposób przetestowania:

1. Połączenie komputera z urządzeniem NPE za pomocą portów RS232
2. Przykładowe wywołanie programu Reader na komputerze lokalnym: *java Reader COM1*
3. Przykładowe wywołanie programu Writer na urządzeniu NPE: *java Writer /dev/ttyS0*

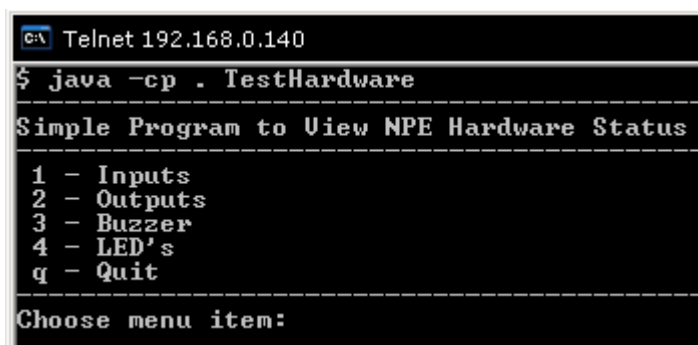
Treść wpisana w konsoli urządzenia NPE jest wysyłana i wyświetlana na komputerze lokalnym. Analogicznie możemy uruchomić program Writer na komputerze lokalnym i wysyłać dane do urządzenia NPE.

Program TestHardware

Konsolowa aplikacja TestHardware demonstruje sposób dostępu do zasobów sprzętowych urządzenia NPE. Dzięki klasie *HWNPE* programista Javy może łatwo odczytać stan wejść, ustawić stany wyjściowe jak i sterować diodami.

Aplikację uruchamiamy poleceniem:

```
java -cp . TestHardware
```



```
C:\> Telnet 192.168.0.140
$ java -cp . TestHardware
Simple Program to View NPE Hardware Status
-----
1 - Inputs
2 - Outputs
3 - Buzzer
4 - LED's
q - Quit
-----
Choose menu item:
```

Wybierając konkretne kategorie możemy odczytywać stany poszczególnych wejść, wyjść, diod.

5.2. Język ANSI C

Pliki źródłowe przykładowych aplikacji można pobrać po zalogowaniu na stronie www.a2s.pl. W poniższym rozdziale znajduje się opis i sposób przetestowania każdej z nich.



Zalecane jest uprzednie zapoznanie się z rozdziałem: *Narzędzia deweloperskie*, gdyż przykładowe programy wymagają kompilacji.



Uruchomienie i kompilacja programów wymagają posiadania systemu Linux na komputerze PC

Przykład 1: Serwer TCP/IP

Przykład *tcp_server* ukazuje program typu TCP echo serwer. Aplikacja nasłuchuje na wybranym porcie TCP i po nawiązaniu połączenia wysyła dane do nadawcy. Aby przetestować tą aplikację należy połączyć komputer z urządzeniem kablem Ethernet lub podłączyć oba urządzenia do sieci LAN. Następnie:

1. Skopiować plik za pomocą dowolnego klienta FTP do wybranego katalogu
2. Następnie należy zmienić: uprawnienia dla pliku wykonywalnego i uruchomić program

```
$ ./tcp_server
```

3. Teraz należy na komputerze PC uruchomić program *tcp_client* symulujący klienta TCP. Przy wywołaniu podajemy adres IP oraz port urządzenia NPE:

```
$ ./tcp_client 192.168.0.101 4000
```

Teraz możemy przetestować aplikację wpisując ciąg znaków, który zostanie wysłany do serwera na urządzeniu, aby zakończyć naciśnij 'q'

Przykład 2: Klient TCP/IP

Przykładowy program *tcp_client* ukazuje jak stworzyć klienta komunikującego się z serwerem TCP. Program odbiera dane z określonego adresu IP serwera. Aby przetestować ten program należy połączyć komputer z urządzeniem kablem Ethernet lub podłączyć oba urządzenia do sieci LAN. Następnie:

1. Na komputerze PC uruchomić program *tcp_server* podając port

```
$ ./tcp_test 4005
```

2. Skopiować program *tcp_klient* do dowolnego katalogu urządzenia NPE
3. Wywołać aplikację z określonym numerem portu

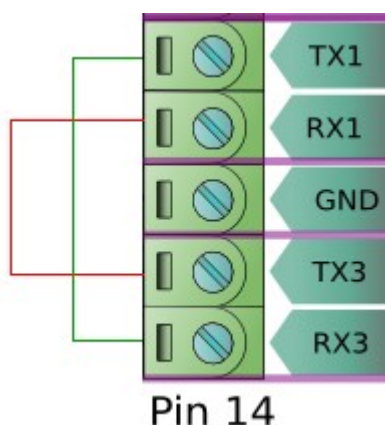
```
$ ./tcp_client 4005
```

4. Po nawiązaniu połączenia można wysyłać dane. Serwer po otrzymaniu wiadomości odsyła jej echo do nadawcy. Zakończenie komunikacji odbywa się po wpisaniu q

Przykład 3: Test portu szeregowego

RStest

Przykład programu RStest sprawdza komunikację pomiędzy dwoma dostępnymi portami RS232 w urządzeniu NPE. Przed uruchomieniem aplikacji należy połączyć porty wejścia-wyjścia interfejsów szeregowych w następujący sposób:



Szczegółowy opis złącz znajduje się w rozdziale: Opis złącz.

Aby uruchomić program należy skopiować pakiet do dowolnego katalogu urządzenia i uruchomić poleceniem:

```
$. /RStest
```

Program wysyła 100 losowych bajtów z *ttyS0* do *ttyS2* i weryfikuje odebrane dane. Następnie test zostaje przeprowadzony w odwrotnym kierunku a po zakończeniu wyświetlany jest rezultat.

Przykład 4: Komunikacja RS232/Ethernet

Com2Tcp

Kolejna przykładowa aplikacja konwertuje dane przesyłane pakietami TCP i przekazuje je dalej portem szeregowym. Wszystkie wymagane opcje konfiguracyjne znajdują się w pliku *gateway.conf*.



Zalecane jest dokładne zapoznanie się z plikiem konfiguracyjnym przed uruchomieniem aplikacji

Program może być uruchomiony w tle jako demon. Aby wymusić taką opcję pracy należy przy uruchomieniu wywołać program z atrybutem:

```
$com2tcp -d
```

Warto sprawdzić czy program działa w tle:

```
$ps -e | grep com2tcp
```

Zatrzymanie wykonywania programu jako demona polega na zabiciu procesu za pomocą polecenia:

```
$kill -s SIGINT PID
```

Gdzie PID jest pierwszym numerem procesu wylistowanego powyżej

Zatrzymanie aplikacji działającej w normalnym trybie odbywa się poprzez naciśnięcie *Ctrl+C*.

Rozdział 6 Programowanie NPE

6.1. Narzędzia deweloperskie

Tworzenie autorskich aplikacji dedykowanych dla urządzenia NPE wymaga środowiska programistycznego wspierającego różne języki, tj. Java, C, C++. Na potrzeby poniższego rozdziału posłużymy się programem Eclipse SDK.



Program Eclipse SDK można pobrać ze strony producenta:

<http://www.eclipse.org/downloads/>



Dostępne są wersje na różne systemy operacyjne (Windows, Linux, Mac).

Wersja dedykowana dla języka C - Eclipse IDE for C/C++ Developers zajmuje 79 MB

Wersja podstawowa dla języka Java - Eclipse Classic zajmuje 162 MB

Program jest gotowy do uruchomienia zaraz po rozpakowaniu.

NPE Tool Chain

W tym rozdziale znajdują się informacje, jak krok po kroku zainstalować niezbędne narzędzia programistyczne potrzebne do tworzenia przez użytkowników własnych aplikacji w języku ANSI C/C++ pod systemem Linux.

Instalacja

Tool Chain składa się z kompilatora wraz z powiązаныmi narzędziami, takimi jak niezbędne biblioteki i nagłówki. Wymagane komponenty muszą zostać zainstalowane na komputerze PC z systemem Linux. Na potrzeby instrukcji skorzystaliśmy z dystrybucji Ubuntu 7.04 (feisty).

Tool Chain wymaga ok. 134MB wolnego miejsca na dysku.



Przed rozpoczęciem instalacji należy upewnić się czy w naszym systemie zostały zainstalowane pakiety: *tar*, *gedit*. Do wykonania poniższych czynności można wykorzystać również inne pakiety dostępne w systemie Linux.

Aby zainstalować Tool Chain musimy:

1. Uruchomić terminal i przejść do katalogu gdzie znajduje się plik

NPE_toolchain_010308.tar.gz

```
$cd /<ścieżka_do_pliku>/
```

2. Wpisać i wykonać następujące polecenie w terminalu, które rozpakuje spakowany plik:

```
$sudo tar xzf NPE_toolchain_010308.tar.gz -C
```

3. Począkać aż wszystkie pliki zostaną rozpakowane
4. Dodać katalog `/home/user/ARMLinux/buildroot/build_arm/staging_dir/bin/` do ścieżki za pomocą polecenia w terminalu:

```
$export PATH=
PATH:/home/user/ARMLinux/buildroot/build_arm/staging_dir/bin/
```

5. W celu dodania tej ścieżki każdorazowo przy uruchomieniu systemu, należy dodać to polecenie na końcu pliku `/etc/bash.bashrc`. Aby to wykonać należy wpisać w terminalu:

```
$sudo gedit /etc/bash.bashrc
export PATH=
$PATH:/home/user/ARMLinux/buildroot/build_arm/staging_dir/bin/
```

Następnie zachować zmianę. Przy uruchomieniu ścieżka zostanie dodana automatycznie.



Plik `/etc/bash.bashrc` znajduje się tylko w dystrybucji Ubuntu.

Każda dystrybucja Linuksa posiada analogiczny plik konfiguracyjny, jednak może on posiadać inną nazwę i znajdować się w odmiennych katalogach – co należy uwzględnić podczas wpisywania komend.

Kompilowanie aplikacji

Pierwszym krokiem jest zamontowanie katalogu `/home/NFS` z komputera PC w systemie docelowym używającym systemu NFS. W tym celu z poziomu urządzenia NPE należy użyć polecenia:

```
$nfson <adres IP urządzenia NPE>:/home/nfs
```

W efekcie powinniśmy uzyskać:

```
$ nfson 192.168.0.133:/home/nfs
NFS ready to use.
$ df -h
Filesystem      Size      Used Available Use% Mounted on
rootfs          13.9M    11.2M      2.7M   81% /
/dev/root       13.9M    11.2M      2.7M   81% /
/dev            30.1M       0      30.1M    0% /dev
/dev/mtdblock0    8.0M    880.0k     7.1M   11% /mnt/mtd
/dev/mmcblk0p1  949.6M    53.4M   847.9M    6% /mnt/sd
192.168.0.133:/home/nfs
100.4G       7.9G   87.4G    8% /mnt/nfs
```

Dzięki temu katalog `/home/nfs` powinien zostać zamontowany jako `/mnt/nfs` w systemie NPE.

Dla upewnienia się sprawdzono zamontowane w systemie partycje poleceniem `df -h`.



Odmontowanie katalogu obsługuje polecenie `nfsoff`

Kompilowanie programów zawierających plik *Makefile* z wiersza poleceń w urządzeniu NPE polega na:

1. Przejściu w terminalu do katalogu zawierającego daną aplikację
2. Następnie wykonaniu poleceń

```
$make clean  
$make
```

3. Wykonanie powyższych czynności wymusza skompilowanie źródeł oraz skopiowanie pliku wykonywalnego do katalogu */mnt/nfs* w systemie docelowym
4. Kolejnym krokiem jest nadanie uprawnień do wykonywania

```
$chmod +x /mnt/nfs/<nazwa skompilowanego pliku>
```

5. Uruchomienie skompilowanej aplikacji

```
$/mnt/nfs/<nazwa skompilowanego pliku>
```



Przykładowe kody źródłowe aplikacji stworzonych w języku C znajdują się na stronie www.a2s.pl w dziale NPE. Programy te wymagają osobnej kompilacji pod system Linux.

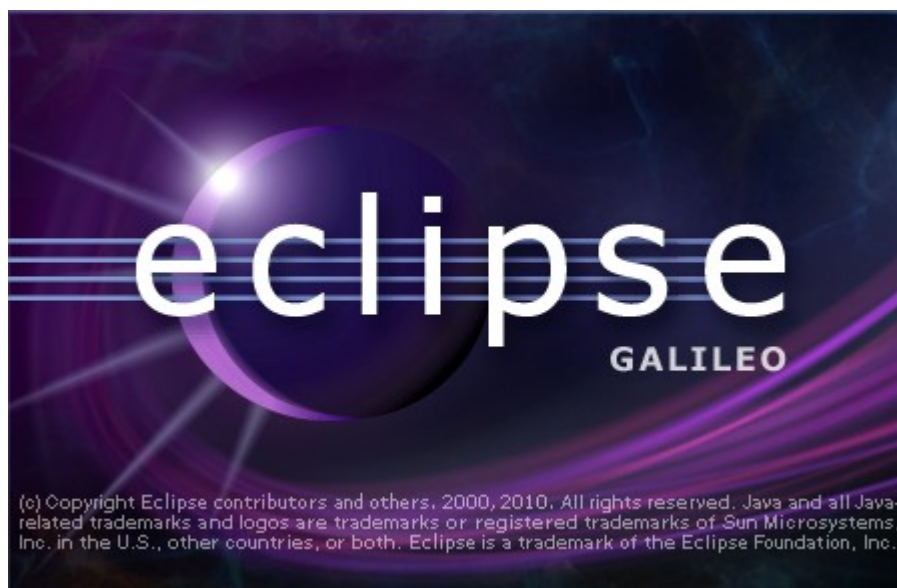
6.2. ANSI C/ C++

Przygotowanie środowiska Eclipse C/C++

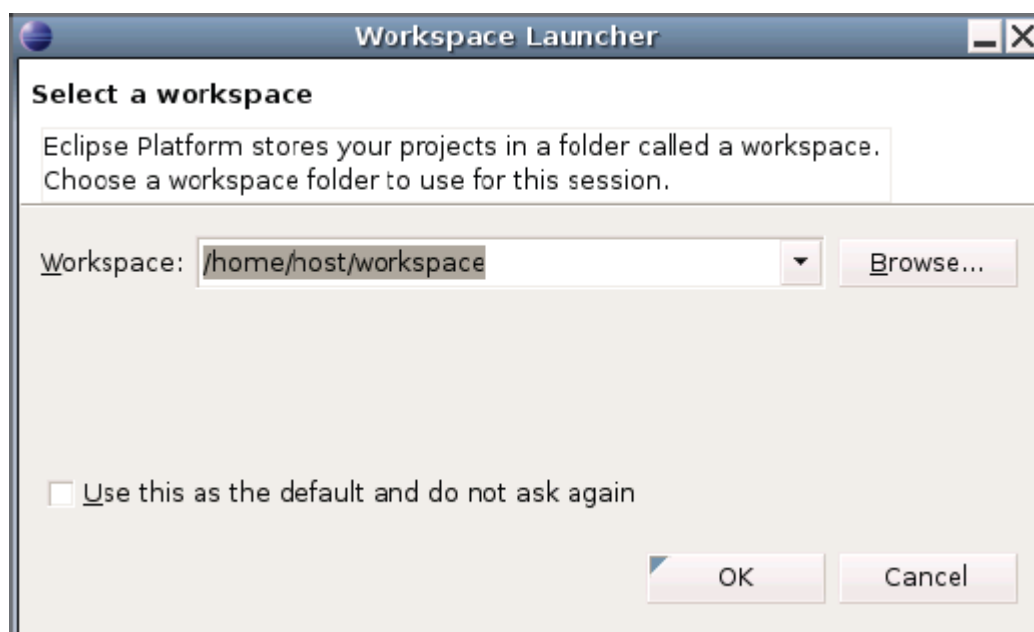
Projekt CDT (C/C++ Development Tools) zapewnia pełną funkcjonalność zintegrowanego środowiska C i C++ w programie Eclipse. Zalecane jest używanie Eclipse CDT IDE w procesie tworzenia aplikacji dedykowanych dla urządzenia NPE. W rozdziale został przedstawiony proces instalacji i konfiguracji środowiska IDE pod systemem Ubuntu Linux.

Pierwszym krokiem jest pobranie i rozpakowanie programu Eclipse IDE for C/C++ Developers.

Zaraz po rozpakowaniu aplikacja jest gotowa do uruchomienia. Po wywołaniu programu powinna się pojawić ikona ładowania:

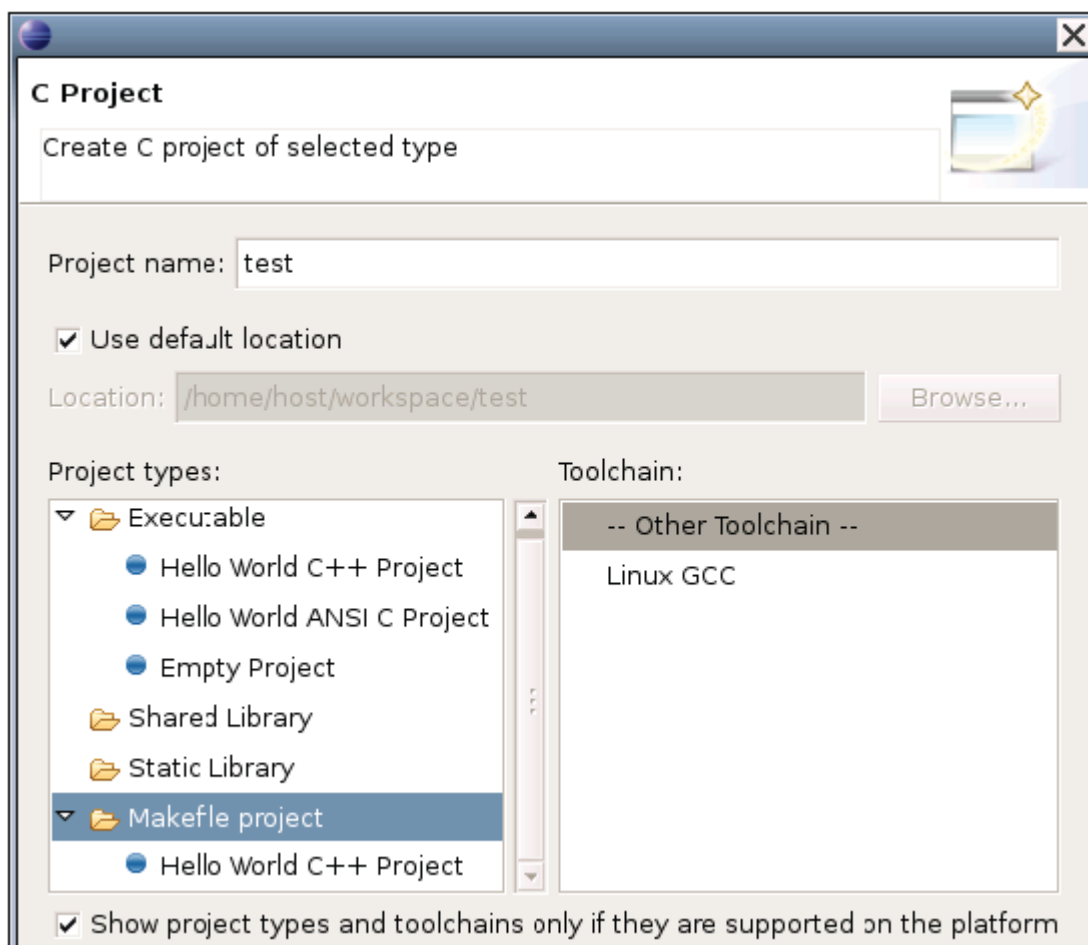


Po uruchomieniu aplikacji powinno pojawić się okno wyboru katalogu, gdzie będą tworzone nowe projekty:



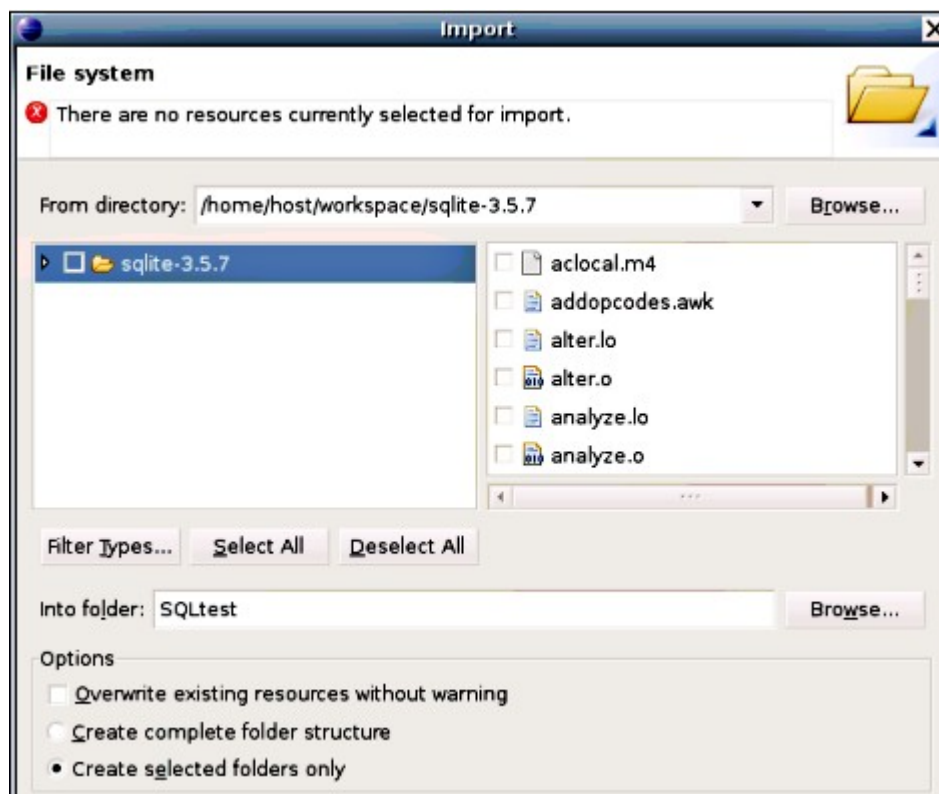
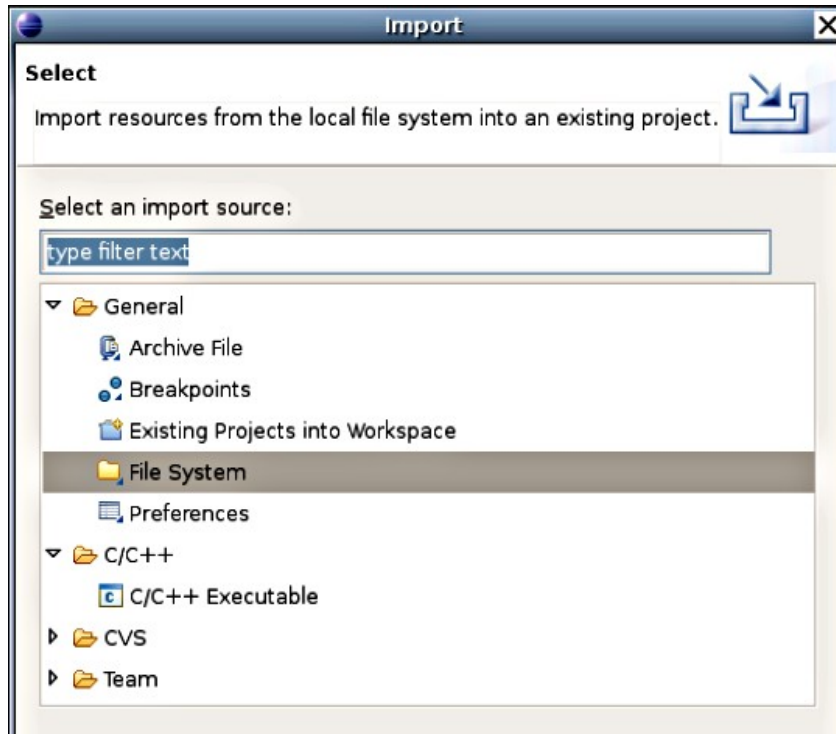
Po zatwierdzeniu powinno się pojawić okno programu IDE.

Następnie wybieramy *File->New->C Project* i wpisujemy wybraną nazwę nowego projektu, zaznaczając przy tym z listy wersję projektu typu *Makefile* i wybieramy *Finish*.



Folder nowo utworzonego projektu powinien być widoczny w oknie Project Explorer. Teraz możliwe jest utworzenie (*File -> New*) lub importowanie (*File -> Import*) istniejących plików do projektu. Następnie należy wybrać pliki do importowania (źródła, biblioteki...).

Przykładowo załączamy biblioteki obsługujące bazę SQLite3.



Projekt typu *makefile* musi posiadać plik Makefile, w którym zawarte są informacje dotyczące kompilacji. Środowisko IDE Eclipse może generować domyślną zawartość pliku Makefile, ale zalecane jest tworzenie własnego tak by mieć pełną kontrolę nad procesem budowania i kompilowania projektu.

Przykładowa zawartość pliku Makefile w projekcie aplikacji dedykowanej do systemu NPE powinna wyglądać następująco:

```
LDLFLAGS = -s -Wl,-warn-common -lc
SRC := $(APP).c file.c
OBJ := $(APP).o file.o

#Compile section
.c.o:
    @echo Compile... $<
    @$ (CC) $(CFLAGS) -I. -c -o $@ $<

#make all section
all: $(APP)
$(APP): $(OBJ)
    @echo Link... $@
    @$ (CC) $(CFLAGS) -o $@ $^ $(LDFLAGS)
    @echo Strip... $@
    @$ (CROSS)strip $@
    @echo Copy to nfs...
    @cp $@ /mnt/nfs/
    @$ (CROSS)size $@
    @echo Done.

#make clean section
clean:
    @rm -f *.o *.gdb $(APP)

#make install section
install:
    @echo Copy to initrd...
    @cp $(APP) ../../buildroot/target/generic/target_skeleton/bin/
```

Debugging programem Insight

Insight jest graficznym interfejsem konsolowej aplikacji GDB, GNU debuggera. Poniższy rozdział ukazuje jak kompilować i skonfigurować program Insight do zdalnego debugowania.



Program Insight można pobrać ze strony projektu:
<ftp://sourceware.org/pub/insight/releases>

Kompilowanie programu Insight polega na:

1. Pobraniu najnowszej wersji źródła programu Insight
2. Rozpakowaniu archiwów do wybranego katalogu
3. Przejściu w terminalu do rozpakowanego katalogu i uruchomieniu skryptu konfiguracyjnego z opcjami:

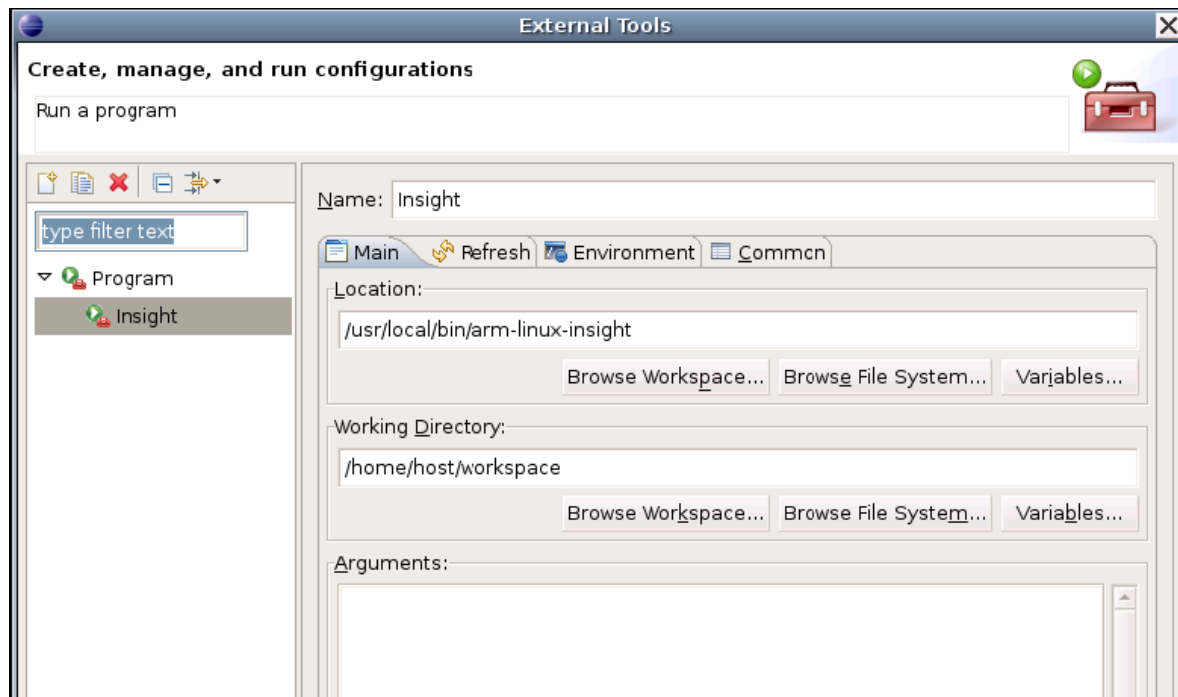
```
./configure -build=i686-linux -host=i686-linux -target=arm-linux
```

4. Uruchomieniu skryptu *Makefile* i instalacji:

```
make && make install
```

Po poprawnym zainstalowaniu zalecane jest dodanie *Insight* jako zewnętrznego dodatku do Eclipse. W tym celu:

1. Po uruchomieniu Eclipse wybieramy w menu
Run -> External Tools -> External Tools...
2. Powinno pojawić się okno:



Program, który chcemy debugować musi być skompilowany bez żadnych optymalizujących flag kompilatora gcc i z flagą `-g` (generuje informacje debugujące niezbędne dla debuggera).

Aby przetestować program Insight na prostym programie Hello World należy:

1. Uruchomić program, który chcemy debugować w urządzeniu NPE z wykorzystaniem *gdbservera* za pomocą polecenia

```
arm-linux-gdbserver <adres IP>:Port <ścieżka do pliku. np./mnt/mtd/hello>
```

Opis API

Hwnpe jest biblioteką umożliwiającą dostęp do hardwareowych zasobów urządzenia NPE. Jeżeli chcemy uzyskać dostęp do zasobów spod języka ANSI C/C++ musimy dołączyć bibliotekę: *libhwnpe.h* do pliku Makefile naszego projektu w miejscach:

```
LIBS = ../libnpe/lib/libhwnpe
INCLUDE = . ../libhwnpe

...

$(CC) $(CFLAGS) -I. -I../libhwnpe -c -o $@ $<
...
all: $(APP)
...
$(CC) $(CFLAGS) -o $@ $^ $(LDFLAGS) $(LIBS)
```

Dostęp do portów wyjściowych

Ustawianie niebuforowanego stanu portu wyjściowego urządzenia NPE odbywa się poprzez funkcje:

- `set_do()`:

```
int set_do(int port, int val);
```

Funkcja ta nie umożliwia późniejszego odczytu ustawionego stanu. Jako parametry przyjmują nazwę portu (zgodnie z oznaczeniami w tabeli 1 i 2) oraz wartość 0 lub 1. W przypadku poprawnego zapisu funkcja zwraca 1 a w innym przypadku -1.

- `set_do_buf()`:

```
int set_do_buf(int port, int val);
```

Funkcja umożliwia odczyt ustawionego stanu za pomocą funkcji `get_do_buff()`. Jako parametry przyjmują nazwę portu (zgodnie z oznaczeniami w tabeli 1 i 2) oraz wartość 0 lub 1. W przypadku poprawnego zapisu funkcja zwraca 1 a w innym przypadku -1.

- `get_do_buff()`:

```
int get_do_buf(int port);
```

Funkcja implementuje odczyt aktualnego stanu określonego portu wyjściowego. Jako parametr przyjmują nazwę portu (zgodnie z oznaczeniami w tabeli 1 i 2).

Dostęp do portów wejściowych

Odczyt aktualnego stanu portu wejściowego odbywa się poprzez wykorzystanie funkcji `get_di()`:

```
int get_di(int port);
```

Funkcja implementuje odczyt aktualnego stanu określonego portu wejściowego. Jako parametr przyjmują nazwę portu (zgodnie z oznaczeniami w tabeli 1 i 2).

Sterowanie diodami LED

Diody w systemie widoczne są jako określone porty wyjściowe. Ustawianie stanu diod *User* i *Status* odbywa się poprzez funkcje dostępu do portów wyjściowych z oznaczeniami pinów `USER_LED` lub `STATUS_LED` analogicznie jak w podrozdziale Dostęp do portów wyjściowych.

Sterowanie diodą GPRS wymaga wykorzystania osobnej funkcji:

```
int set_gprs_led(int st);  
int get_gprs_led();
```

Jako parametr funkcja `set_gprs_led()` przyjmuje wartość 0 lub 1. W przypadku poprawnego zapisu funkcja zwraca 1 a w innym przypadku -1.

Sterowanie Buzzerem

Obsługa systemowego buzzera została zaimplementowana w funkcjach:

```
int set_buzzer(int st);  
int get_buzzer();
```

Jako parametr funkcja `set_buzzer()` przyjmuje wartość 0 lub 1.

Odczyt wejść przetworników ADC

Odczyt sygnału wejściowego przetworników ADC umożliwiających pomiar napięć wykorzystuje funkcję:

```
int get_ai(int chan);
```

Jako parametr przyjmuję nazwę portu(zgodnie z oznaczeniami w tabeli 1 i 2).

6.3. JAVA

JamVM

JamVM jest bardzo małą wirtualną maszyną Javy typu Open Source. JamVM została zaprojektowana do wykorzystania bibliotek Javy klasy GNU Classpath. System NPE fabrycznie zawiera główne klasy Java takie jak: classpath.zip i glibj.zip wraz z bibliotekami.

W pakiecie znajdują się również biblioteki RX/TX – obsługujące operacje wejścia/wyjścia, implementacje jamod modbus, biblioteki API dla hardware urządzenia NPE(JNI – SWIG).

Instalacja JamVM

Pierwszym krokiem instalacji wirtualnej maszyny JamVM jest skopiowanie skryptu instalacyjnego `java_install_<wersja>` pobranego z serwera na kartę SD a następnie nadanie mu atrybutów uruchomienia:

```
chmod +x java_install_<numer wersji>
```

Kolejnym krokiem jest uruchomienie skryptu:

```
./java_install_<numer wersji>
```

Instalacja nie wymaga ingerencji użytkownika, skrypt automatycznie sprawdza dostępne miejsce w katalogach `/mnt/sd`, `/mnt/nand` i instaluje tam gdzie ilość dostępnego miejsca jest wystarczająca. W przypadku braku miejsca instalacja jest przerywana i wyświetlany jest określony komunikat.

Po zakończeniu instalacji można sprawdzić dostępność JamVM za pomocą polecenia `java`

```
C:\ Telnet 192.168.0.140
$ java
Usage: /mnt/sd/JamVM/bin/jamvm [-options] class [arg1 arg2 ...]
      <to run a class file>
      or /mnt/sd/JamVM/bin/jamvm [-options] -jar jarfile [arg1 arg2 ...]
      <to run a standalone jar file>

where options include:
  -client          compatibility <ignored>
  -server          compatibility <ignored>

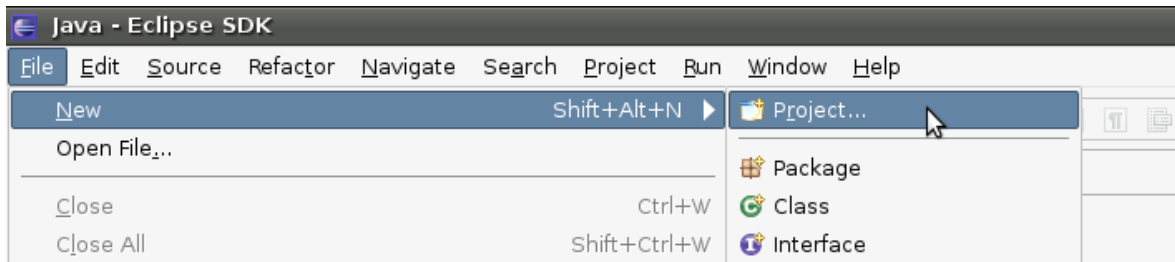
  -cp              <jar/zip files and directories separated by :>
  -classpath       <jar/zip files and directories separated by :>
                  locations where to find application classes
  -D<name>=<value> set a system property
  -verbose[:class!gc!jni]
                  :class print out information about class loading, etc.
                  :gc print out results of garbage collection
                  :jni print out native method dynamic resolution
  -version         print out version number and copyright information
  -showversion     show version number and copyright and continue
  -fullversion     show jpackage-compatible version number and exit
  -? -help        print out this message
  -X              show help on non-standard options
```

Pierwszy projekt w Eclipse SDK



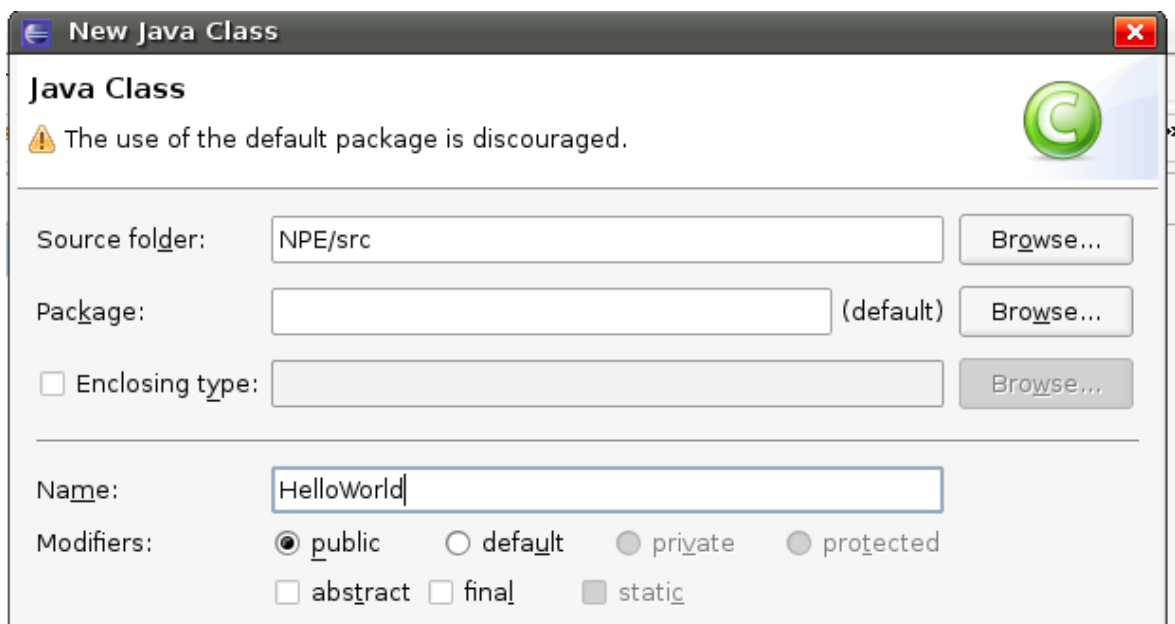
W tym rozdziale wykorzystujemy wersję programu - Eclipse Classic.

Proces tworzenia rozpoczynamy od uruchomienia programu Eclipse i podania ścieżki do *Workspace*, gdzie będą przechowywane utworzone projekty. Na początku należy stworzyć projekt. Na górze w menu naciskamy *File>New>Project*

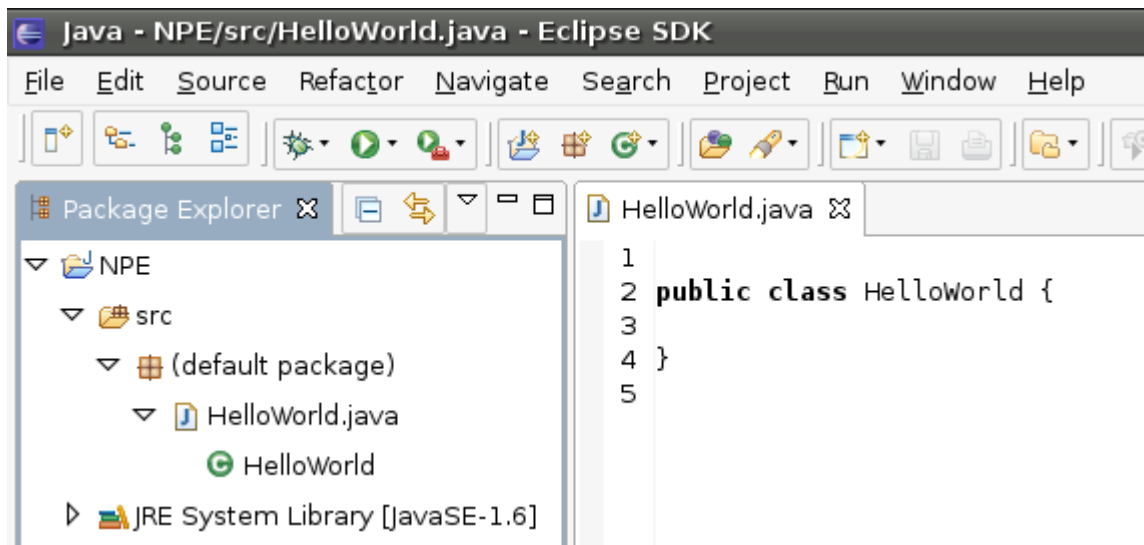


W kolejnym oknie wybieramy typ projektu: *Java Project* i klikamy *Next*. Następnie zostaniemy poproszeni o podanie nazwy projektu – przykładowo wybieramy *NPE*. W polu *Directory* znajduje się ścieżka do katalogu z naszym projektem. Potwierdzając klikamy *Finish*.

Stworzyliśmy pusty projekt, z lewej strony ekranu widoczna jest zakładka *Package Explorer*, gdzie znajdują się katalogi z plikami. Aby dodać klasę klikamy prawym przyciskiem myszy na klasie *NPE*, z menu wybieramy *New>Class*. Pojawi się okno tworzenia klasy w naszym projekcie. W polu *Name* wpisujemy *HelloWorld* i wybieramy poniżej opcję *Public*



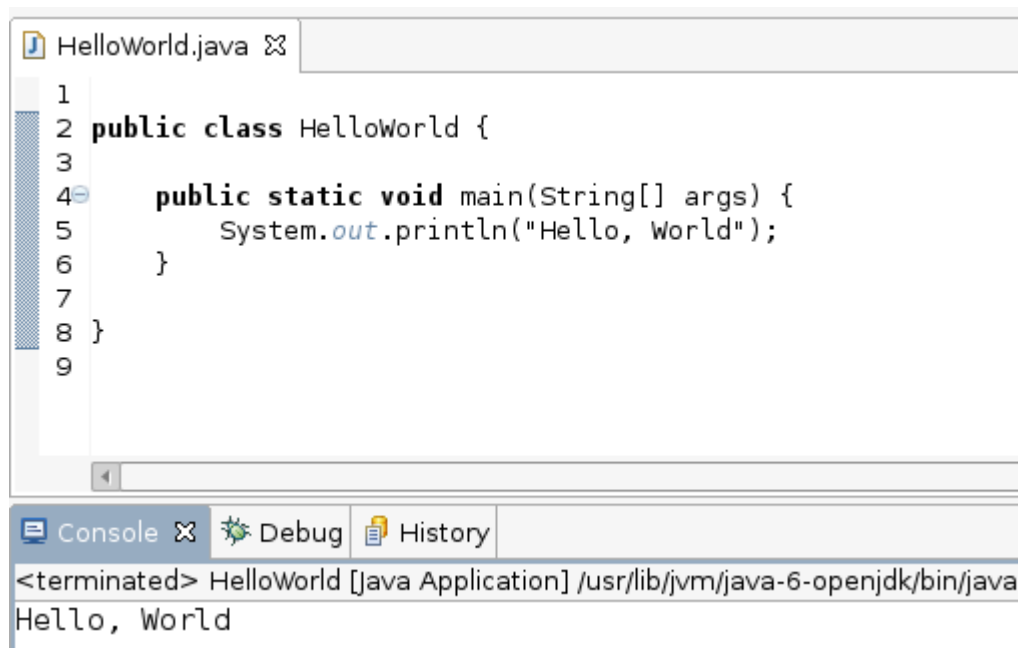
Po zatwierdzeniu zostanie dodany plik *HelloWorld.java* do naszego projektu.



W otwartym pliku wpisujemy:

```
#public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello, World");  
    }  
}
```

Zapisujemy zmiany: *File>Save* a następnie uruchamiamy projekt wybierając z menu: *Run>Run*. Jeżeli wszystko wpisaliśmy poprawnie na dole w oknie *Console* powinno się pojawić:



Automatycznie po uruchomieniu stworzony został plik *HelloWorld.class*, który możemy skopiować na urządzenie NPE i uruchomić spod lokalnej maszyny JamVM.

Opis API

Urządzenie NPE posiada zaimplementowane biblioteki umożliwiające wysokopoziomowy dostęp do zasobów urządzenia. W oparciu o klasę *HardwareNPE* użytkownik ma możliwość tworzenia aplikacji w języku Java, które korzystają z dostępnych zasobów i funkcjonalności. Funkcje odczytujące i ustawiające stany portów IO zostały zaimplementowane tak, aby w wypadku niepowodzenia generowały wyjątek, który użytkownik może przechwycić i obsłużyć.

Klasę testową sprawdzającą dostęp do diod LED i buzzera dostępu można uruchomić komendą:

```
java -jar <katalog zainstalowania JamVM>/hwnpe.jar
```

Odczyt wartości wejść cyfrowych

Wysokopoziomowy dostęp do wejść cyfrowych zaimplementowany jest w klasach *outputRead* i *DigitalInputs*.

Sposób przykładowego odczytu stanu na porcie DI1:

```
HardwareNPE.getReference().outputRead(DigitalInput.DI1)
```

*Nazwy poszczególnych portów wejściowych są analogiczne jak w tabeliach 1 i 2.

Ustawianie stanu wyjść cyfrowych, diod LED i Buzzera

Obsługa wyjść cyfrowych urządzenia NPE zaimplementowana została w funkcji *outputSet* i klasie *DigitalOutput*. Dzięki temu użytkownik może ustawiać i odczytywać stany poszczególnych portów.

Przykładowy odczyt aktualnego stanu na porcie DO1:

```
HardwareNPE.getReference().outputSet(DigitalOutput.DO1
```

*Nazwy poszczególnych portów wejściowych są analogiczne jak w tabeliach 1 i 2.

Dodatkowo porty wyjściowe umożliwiają buforowanie stanów dzięki czemu użytkownik, może sprawdzić aktualnie ustawione wartości. Odczytanie zbuforowanego stanu odbywa się poprzez funkcję *outputRead* analogicznie jak przy odczycie wejść cyfrowych.

```
HardwareNPE.getReference().outputRead(DigitalOutput.USER_LED)
```

Buzzer i diody dostępne w urządzeniu NPE są widoczne w systemie jako wyjścia cyfrowe i dostęp do nich zaimplementowany jest w klasie *DigitalOutput*. Do dyspozycji użytkownika przeznaczone są diody *Gprs*, *User* i *Status*. Wykorzystując odpowiednie funkcje klasy *HardwareNPE* można odczytywać jak i zadawać określone stany.

Przykładowo zapalenie wszystkich diod i aktywowanie buzzera wygląda następująco:

```
HardwareNPE.getReference().outputSet(DigitalOutput.USER_LED, 0)
HardwareNPE.getReference().outputSet(DigitalOutput.GPRS_LED, 0)
HardwareNPE.getReference().outputSet(DigitalOutput.STATUS_LED, 0)
HardwareNPE.getReference().outputSet(DigitalOutput.BUZZER, 1)
```

Odczyt wartości przetworników ADC

Klasa *HardwareNpe* umożliwia odczyt napięcia wejściowego z przetworników dostępnych na portach *AI1*, *AI2*, *AI3* i *AI4*. Odczytanie napięcia umożliwia funkcja *analogRead()*, której przykładowe wywołanie wygląda następująco:

```
HardwareNPE.getReference().analogRead(AnalogInput.AI3)
```

*Nazwy poszczególnych portów wejściowych są analogiczne jak w tabeliach 1 i 2.

W efekcie uzyskaliśmy wartość napięcia odczytaną przez przetwornik na porcie *AI3*.

Rozdział 7 Bazy Danych

7.1. Systemy baz danych

Urządzenie NPE znajduje szerokie zastosowanie w systemach monitorujących i zbierających dane. Dzięki obsłudze systemów bazodanowych takich jak SQLite3 oraz PostgreSQL istnieje możliwość łatwego zarządzania dużą liczbą przechowywanych danych pomiarowych.

Rozdział opisuje sposób instalacji i dostępu do wyżej wymienionych systemów baz danych z poziomu języka Java.

Postgre SQL

PostgreSQL jest jednym z najbardziej zaawansowanych systemów typu OpenSource zarządzającym relacyjnymi bazami danych. System implementuje język SQL i jest z nim w pełni kompatybilny.

Instalacja

1. Rozpakuj spakowany plik *pgsql.tar.gz* do */mnt/sd*.

2. Wejdź do powstałego katalogu:

```
cd /mnt/mtd/pgsql
```

3. Nadaj uprawnienia dla katalogu *pgsql*

```
chmod -R 750 /mnt/mtd/pgsql
```

4. Uruchom skrypt (po wykonaniu skryptu jesteśmy użytkownikiem postgres)

```
./postinst
```

5. Utwórz cluster dla bazy danych(zestaw baz danych zarządzanych przez jeden serwer)

```
initdb -D /mnt/sd/pgsql/data
```

6. Uruchom skrypt

```
./post_run
```

7. Sprawdź logfile, aby sprawdzić czy baza jest gotowa na przyjęcie połączenia

```
cat /mnt/sd/pgsql/logfile
```

```
$ cat /mnt/sd/pgsql/logfile
LOG:  could not create IPv6 socket: Address family not supported by protocol
LOG:  could not create socket for statistics collector: Address family not supported
by protocol
LOG:  trying another address for the statistics collector
LOG:  database system was shut down at 1988-02-29 14:15:25 MET
LOG:  autovacuum launcher started
LOG:  database system is ready to accept connections
```

Narzędzia

Podstawowe informacje	
initdb	tworzenie clustera bazy danych
createdb	tworzenie bazy danych
pg_ctl	obsługa serwera bazy danych
dropdb	usuwa bazę danych
postmaster	uruchamia serwer
Linux	
pgAdmin III	program do pełnej obsługi bazy danych
Razorsql	program do pełnej obsługi bazy danych
Windows	
PostgreSQL	Program do pełnej obsługi bazy danych
PostgreSQL Data Wizard	Program do pełnej obsługi bazy danych
EMS PostgreSQL Manager	Program do pełnej obsługi bazy danych

Dostęp z wiersza poleceń

Stworzenie bazy danych w domyślnej lokalizacji wywołamy za pomocą polecenia *createdb*

```
createdb <nazwa_naszej_bazy>
```

Jeśli chcemy przechowywać naszą bazę danych np. w Pamięci SD, Flash lub RAM możemy użyć komendy *tablespace* podczas tworzenia bazy.

Na początku musimy stworzyć pusty katalog oraz nadać mu uprawnienia:

```
mkdir /mnt/sd/baza  
chmod -R 750 /mnt/sd/baza  
chown -R postgres.web /mnt/sd/baza
```

W postgresql stwórzmy nowy wpis tablespace(miejsce na bazę):

```
CREATE TABLESPACE sdcard LOCATION '/mnt/sd/baza';
```

```
postgres=# create tablespace sdcard location '/mnt/sd/baza';  
CREATE TABLESPACE
```

Podgląd wpisanych lokacji w tablespace uzyskamy za pomocą:

```
SELECT * FROM pg_tablespace;
```

Następnie możemy przejść do stworzenia bazy danych w zadanej lokalizacji:

```
createdb --tablespace=sdcard nasza_baza
```

Aby uruchomić terminal do obsługi bazy danych należy użyć komendy

```
psql <nazwa_naszej_bazy>
```

```
postgres=# select * from pg_tablespace;
 spcname | spcowner | spclocation | spcacl
-----+-----+-----+-----
 pg_default |      10 |              |
 pg_global  |      10 |              |
 sdcard     |      10 | /mnt/sd/baza |
(3 rows)
```

Stwórzmy tabelę oraz wprowadzamy przykładowe dane:

```
create table testowa(id serial primary key, dane integer, text varchar);
insert into testowa values ('1','15','Techbase');
insert into testowa values ('2','47','NPE');
```

```
postgres=# create table testowa(id serial primary key, dane integer, text varchar);
insert into testowa values ('1','15','Techbase');
insert into testowa values ('2','47','NPE');
NOTICE: CREATE TABLE will create implicit sequence "testowa_id_seq" for serial column "testowa.id"
NOTICE: CREATE TABLE / PRIMARY KEY will create implicit index "testowa_pkey" for table "testowa"
CREATE TABLE
postgres=# INSERT 0 1
postgres=# INSERT 0 1
```



Przydatne komendy :

```
\dt    wyświetla istniejące tabele w bazie
\l+    wyświetla bazy danych wraz z lokalizacją
\q     wyjście z terminalu
```

Zdalny dostęp

Jeżeli zależy nam na zdalnym dostępie do bazy danych musimy zmodyfikować plik `/data/pg_hba.conf` po przez dodanie wiersza:

```
Host      all      all      192.168.0.0/24      md5
```

oraz plik `/data/postgresql.conf`:

```
listen_addresses = '*'
```



Należy pamiętać o odkomentowaniu wiersza. Inaczej zmiany nie zostaną uwzględnione.

Następnie zrestartujemy serwer:

```
pg_ctl restart -D <ścieżka do bazy>
```

Efekt powinien być następujący:

- klient

```
$ pg_ctl restart -D /mnt/sd/pgsql/data
waiting for server to shut down.... done
server stopped
server starting
```

- serwer

```
LOG:  received smart shutdown request
LOG:  autovacuum launcher shutting down
LOG:  shutting down
LOG:  database system is shut down
```

Przykładowe skrypty



Na płycie CD dostarczanej z urządzeniem NPE znajdują się programy demonstrujące interakcję z bazą danych PostgreSQL za pomocą języka Java:

Postgres_create.java – generowanie testowej bazy danych

Postgres_example.java - połączenie z bazą oraz wyświetlenie przykładowej tabeli

Postgres_speed.java – testowanie szybkości wyłuskiwania oraz modyfikacji danych

SQLite3

SQLite3 to system zarządzania bazą danych oraz biblioteka C implementująca taki system, obsługująca język SQL. Zawartość bazy danych przetrzymywana jest w jednym pliku (do 2 TB). Baza SQLite jest utrzymywana na dysku przy użyciu B-drzew. Bazy danych zapisywane są jako pliki binarne a ich bezpieczeństwo jest oparte na zabezpieczeniach oferowanych przez używany system plików.



System zarządzania bazą danych SQLite3 instalowany jest domyślnie w każdym urządzeniu NPE. W ogólnym przypadku możemy pominąć podrozdział Instalacji.

INSTALACJA

1. Skopiowanie pliku *sqlite3* do katalogu */bin/* za pomocą dowolnego klienta FTP
2. Skopiowanie biblioteki *libsqlite3.so.0.8.6* do katalogu */lib/*
3. Stworzenie symbolicznego dowiązania poleceniem:

```
ln /lib/libsqlite3.so.0.8.6 /lib/libsqlite3.so.0
```

4. Uruchomienie bazy *sqlite3* za pomocą

```
sqlite3
```

Na ekranie powinniśmy uzyskać:

```
$ sqlite3
SQLite version 3.5.7
Enter ".help" for instructions
sqlite>
```

5. Wyjście z bezy danych za pomocą komendy *exit*

Narzędzia

Podstawowe informacje	
Sqlite3	Konsolowy program obsługi bazy danych.
Linux	
Sqlite database browser	Środowisko graficzne obsługi bazy danych.
Windows	
SQLite3 manager	Program do pełnej obsługi bazy danych
SQLite Maestro	Program do pełnej obsługi bazy danych
SQLite3 database manager Lite	Program do pełnej obsługi bazy danych

Dostęp z wiersza poleceń

Dostęp do bazy danych uzyskujemy przez program *Sqlite3*

```
sqlite3 test.db "create table t1 (t1key INTEGER PRIMARY KEY, data TEXT,
num double,timeEnter DATE);"

sqlite3 test.db "insert into t1 (data,num) values ('This is sample
data',3);"

sqlite3 test.db "insert into t1 (data,num) values ('More sample
data',6);"

sqlite3 test.db "insert into t1 (data,num) values ('And a little
more',9);"
```

Przykładowe skrypty



Na płycie CD dostarczanej z urządzeniem NPE znajdują się programy demonstrujące interakcję z bazą danych Sqlite3 za pomocą języka Java:

Sqlite3_create.java – generowanie testowej bazy danych

Sqlite3_example.java - połączenie się do bazy oraz wyświetlenie przykładowej tabeli

Sqlite3_speed.java – test szybkości wyłuskiwania i modyfikowania danych w bazie

7.2. Dostęp z poziomu języka PHP

SQLite3

Uruchomić w przeglądarce internetowej example.html (/mnt/sd/apache/htdocs)

```
http://<adres IP urządzenia NPE>/example.html
```

Fragment kodu odpowiedzialnego za połączenie do bazy w PHP przy wykorzystaniu PHP Data Object (PDO).

Test szybkości bazy danych w zależności od miejsca umieszczenia bazy danych przekopiować przykładowe pliki bazy danych(baza10.db oraz baza10000) do:

/mnt (RAM)

/mnt/mtd (FLASH)

/mnt/sd (SD CARD)

uruchomić http://192.168.0.107/sqlite3_speed.php

PostgreSql

Uruchomić w przeglądarce internetowej plik: postgresql_example.php(/mnt/sd/apache/htdocs)

```
http://<adres IP urządzenia NPE>/postgres_example.php
```

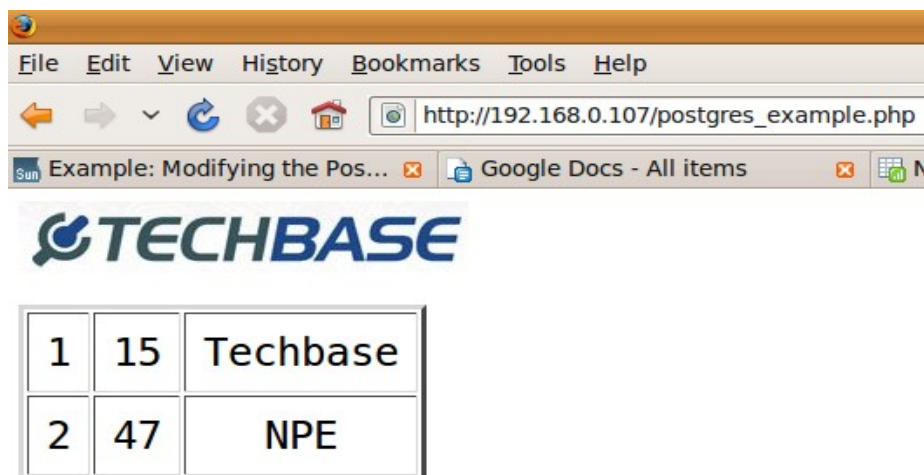
Przykładowe skrypty

Otwórz przed uruchomieniem podane skrypty, aby zapoznać się z kodem oraz wprowadzić własne modyfikacje:

postgres_example.php

postgres_speed.php

sqlite3_speed.php



7.3. Dostęp z poziomu języka Java

Postgre Sql

Uruchom przykładowy program wyświetlający zawartość tabeli podając kolejno: adres serwera, bazę danych, login oraz password:

```
java Postgres_example localhost baza_testowa postgres haslo
```

1. Na ekranie powinniśmy uzyskać:

```
$ java Postgres_example localhost baza_testowa postgres hg;  
Liczba rekordow w tablicy=2  
1 15 Techbase  
2 47 NPE  
$
```

Fragment kodu odpowiedzialny za połączenie do bazy PostgreSQL:

```
Class.forName("org.postgresql.Driver");  
String url = "jdbc:postgresql://" + args[0] + "/" + args[1] +  
    "user=" + args[2] + "&password=" + args[3];  
Connection conn = DriverManager.getConnection(url);  
Statement stat = conn.createStatement();
```



Kolejno podawane parametry oznaczają:

args[0] = adres serwera
args[1] = nazwa bazy danych
args[2] = nazwa użytkownika
args[3] = hasło

SQLite3

Umieśćmy klasę przykładowego programu w classpath:

```
Sqlite_example.classs do /mnt/sd/JamVM
```

Włączmy przykładowy program do obsługi bazy:

```
java Sqlite3_example example.db
```

Fragment kodu odpowiedzialny za połączenie do bazy SQLite3:

```
Class.forName("org.sqlite.JDBC");  
Connection conn = DriverManager.getConnection("jdbc:sqlite:"+args[0]);  
Statement stat = conn.createStatement();
```



Podawany parametr oznacza:

args[0] = adres do pliku bazy danych (*.db) np. /mnt/sd/baza.db

7.4. Wymagania systemowe:

Obsługa systemów baz danych wymaga zainstalowanej wirtualnej maszyny Java wraz z bibliotekami i klasami odpowiedzialnymi za komunikację.



Archiwa .jar zawierające klasy do obsługi bazy danych za pomocą Javy są automatycznie instalowane wraz z pakietem JamVM.



Klasy również można pobrać ze stron projektów:

JDBC3 Postgresql Driver	http://jdbc.postgresql.org/download.html
SQLiteJDBC	http://www.zentus.com/sqlitejdbc/



Zmodyfikuj ścieżkę dostępu do bazy w Xbootclasspath (w pliku /mnt/sd/JamVM/java).

```
$javapath/bin/jamvm -Xbootclasspath:$javapath/classes.zip$javapath/glibj.zip:  
$javapath/sqlitejdbc-v056.jar:$javapath/postgresql-8.4-701.jdbc3.jar:  
$javapath/RXTXcomm.jar $@
```

Rozdział 8 Dodatki

8.1. Informacje podstawowe

Wersja systemu

W celu uzyskania informacji o aktualnej wersji oprogramowania należy skorzystać z aplikacji *npecfg*, która nadzoruje konfigurację całego systemu.

Wywołanie aplikacji z atrybutem

```
npecfg system
```

Wyświetla wszelkie informacje o systemie i urządzeniu. Przykładowa odpowiedź systemu wygląda następująco:

```
C:\ Telnet 192.168.0.140
$ npecfg
#[SYSTEM]
KERNEL_VERSION="2.6.20.4"
COMPILATION_DATE="Wed May 19 20:10:14 2010"
PROCESSOR_TYPE="ARM920T"
DEVICE_ID="NPE single board computer"
UP_TIME="1:46"
SYSTEM_TIME="Fri Jul 9 12:21:55 MET 2010"

#[RAM]
RAM_TOTAL="61560 kB"
RAM_AVAILABLE="33236 kB"

#[FLASH]
FLASH_LOCATION="/mnt/mtd"
FLASH_TOTAL="8192 kB"
FLASH_AVAILABLE="7270 kB"

#[SD_CARD]
SD_1_LOCATION="/mnt/sd"
SD_1_TOTAL="972390 kB"
SD_1_AVAILABLE="868659 kB"
```

8.2. Specyfikacje

Specyfikacja urządzenia

System	
CPU	ARM9 32-bit RISC CPU
CPU Frequency	180 MHz
CPU MIPS	200 MIPS
RAM MEMORY	SDRAM 64 MB / 128 MB*
FLASH MEMORY	16 MB / 72 MB*
NAND FLASH MEMORY	1 GB*

Flash SD	1x socket SD card
System	Linux v 2.6.x
Zegar RTC	RTC, SRAM 240B, Watch Dog Timer
Interfejs Ethernet	
Interfejs Ethernet	Ethernet 10/100 Mbps (1 x RJ45)
Porty szeregowo	
Porty RS-232	2x RS-232, wbudowane zabezpieczenie 15 KV ESD
Porty RS-485	1x RS-485, wbudowane zabezpieczenie 15 KV ESD
	Bity danych: 5, 6, 7, 8
	Bity stopu: 1, 1.5, 2
	Parzystość: None, Even, Odd, Space, Mark
	Prędkość: 50 b/s do 921,6 Kb/s
Kontrolki LED/klawiatura/wejścia-wyjścia	
Sygnalizacja LED	Gotowość systemu x 1, użytkownika x 1
Sygnalizacja Ethernet	LED link, LED 100 Mbit (zintegrowana z gniazdem RJ45)
Przełącznik	1 x Switch monostabilny (dostęp od czoła obudowy)
Gniazdo programowe	1 x złącze 6 pinowe (dostęp od czoła obudowy)
GPIO	8 Wejść cyfrowych, 2-6 Wyjść cyfrowych, Opcjonalnie: 4 Wejścia analogowe, 2 Wyjścia przekaźnikowe
Zasilanie	
Napięcie zasilania	12 ~ 36 Vdc(opcjonalnie 12 ~ 48 Vdc – zakres telekomunikacyjny)
Pobór mocy	7 W maks.
Parametry Mechaniczne	
Wymiary	(Szerokość x Głębokość x Wysokość) 35 x 120 x 101 mm
Waga	300 g
Obudowa	ABS, montaż na szynie DIN
Warunki pracy i przechowywania	
Parametry pracy	Temperatura pracy: -10 ~ 60°C Wilgotność: 5 ~ 95% RH (bez kondensacji) Opcjonalnie: -40 to 75°C (dla NPE-9100-E)
Parametry przechowywania	Temperatura przechowywania: -20 ~ 80°C Wilgotność: 5 ~ 95% RH (bez kondensacji) Opcjonalnie: -40 to 85° C (dla NPE-9100-E)

* - opcje dostępne w modelach 9300/9400 za dodatkową opłatą