

Linux w systemach wbudowanych

Laboratorium L5 G1

Sprawozdanie

Adam Grącikowski, 327350

Warszawa, 4 czerwca 2025

Spis treści

1	Cel ćwiczenia laboratoryjnego	2
2	System Kolejowania Zleceń Produkcyjnych	2
3	Przeniesienie rozwiązania na OpenWrt	2
3.1	Pobranie wersji prekompilowanego	2
3.2	Konfiguracja sieci w OpenWrt	2
3.3	Konfiguracja środowiska OpenWrt	2
3.4	Przeniesienie kodu źródłowego	3
3.5	Problemy z biblioteką do obsługi GPIO	4
4	Podsumowanie	5

1 Cel ćwiczenia laboratoryjnego

Celem ćwiczenia laboratoryjnego było uruchomienie i dostosowanie rozwiązania zrealizowanego w ramach czwartego zadania laboratoryjnego, tym razem w środowisku opartym na systemie OpenWrt. W odróżnieniu od poprzedniego ćwiczenia, w którym wykorzystano środowisko Buildroot do budowy systemu, w niniejszym zadaniu zastosowano prekompilowany obraz OpenWrt.

Ćwiczenie miało na celu zapoznanie się z konfiguracją i dostosowywaniem gotowego systemu OpenWrt do potrzeb konkretnej aplikacji.

2 System Kolejowania Zleceń Produkcyjnych

Aplikacja, którą należało uruchomić w ramach niniejszego zadania, to program zaimplementowany podczas czwartego ćwiczenia laboratoryjnego. W moim przypadku była to aplikacja realizująca System Kolejowania Zleceń Produkcyjnych.

Szczegółowe założenia projektowe oraz opis działania interfejsów — zarówno planisty, jak i pracownika linii produkcyjnej — zostały zawarte w raporcie do poprzedniego zadania laboratoryjnego.

3 Przeniesienie rozwiązania na OpenWrt

3.1 Pobranie wersji prekompilowanego

Pierwszym krokiem w kierunku przeniesienia rozwiązania na OpenWrt było pobranie i przeniesienie na płytkę Raspberry Pi 4 prekompilowanej wersji systemu.

Zgodnie z materiałami wykładowymi, jako źródło prekompilowanej wersji wybrana została odpowiednia dla platformy docelowej zakładka oficjalnej strony OpenWrt, czyli [brcm27xx/bcm2711](#).

Na tej stronie pobrany został plik skompresowany `rpi-4-ext4-factory.img.gz`. Plik ten został następnie rozpakowany, a obraz systemu został wgrany na kartę pamięci przy pomocy polecenia `dd`.

3.2 Konfiguracja sieci w OpenWrt

Zgodnie z materiałami wykładowymi, dokonana została modyfikacja obsługi sieci w OpenWrt przez podłączenie do rzeczywistej sieci.

Należało dokonać edycji zbioru `/etc/config/network` poprzez zamianę istniejącej konfiguracji interfejsu `lan` następującą konfiguracją:

```
config interface 'lan'
    option ifname 'eth0'
    option proto 'dhcp'
```

Po skończonej edycji wywołano `/etc/init.d/network reload` oraz usunięto zbiór uruchamiający serwer DNS/DHCP `dnsmasq` poleceniem `rm /etc/rc.d/S19dnsmasq`.

3.3 Konfiguracja środowiska OpenWrt

W celu uruchomienia aplikacji Systemu Kolejowania Zleceń Produkcyjnych w środowisku OpenWrt, konieczne było przygotowanie systemu poprzez instalację odpowiednich pakietów. Ponieważ apli-

kacja została zaimplementowana w języku Python, pierwszym krokiem było zaktualizowanie listy dostępnych pakietów oraz zainstalowanie interpretera Python 3:

```
opkg update
opkg install python3
python3 --version
```

Następnie zainstalowano biblioteki niezbędne do działania aplikacji, w szczególności związane z obsługą serwera HTTP oraz komunikacją WebSocket. Użyto do tego poniższego polecenia:

```
opkg install python3-tornado \
python3-websocket-client \
python3-websockets \
python3-pyopenssl
```

W dalszej kolejności zainstalowano menedżer pakietów `pip`, który umożliwia instalację dodatkowych modułów Python:

```
opkg install python3-pip
pip --version
```

Na zakończenie, za pomocą `pip`, zainstalowano bibliotekę `UniversalGPIO`, która była wymagana do poprawnego działania aplikacji w warstwie sprzętowej:

```
pip install UniversalGPIO
```

Tak przygotowane środowisko umożliwiło uruchomienie aplikacji w systemie OpenWrt bez konieczności jego rekompilacji.

Skrypt zawierający wszystkie polecenia wymienione w tym podrozdziale został dołączony wraz z raportem. Jego nazwa to `server_install`.

3.4 Przeniesienie kodu źródłowego

Skompresowany plik z całym kodem źródłowym został przeniesiony na urządzenie docelowe przy pomocy polecenia `wget`.

```

/server
├── templates/
│   ├── index.html
│   ├── login.html
│   └── login_failed.html
├── handlers/
│   ├── base.py
│   ├── auth.py
│   └── __init__.py
├── /logs
│   └── events.log
├── config.py
├── models.py
├── enums.py
├── dtos.py
├── notifications.py
├── messages.py
├── hardware.py
├── storage.py
├── orders.json
├── routes.py
├── server.py
└── client.py

```

Rysunek 1: Struktura plików w projekcie `server`

Folder `server` z całym kodem źródłowym został umieszczony w folderze `/usr`. Skryptom `server.py` oraz `client.py` nadane zostały uprawnienia do wykonywania:

```
chmod +x server.py client.py
```

Następnie, w folderze `/etc/init.d` umieszczony został zmodyfikowany skrypt `S99server`, uruchamiający serwer podczas startu systemu. Został on przystosowany do środowiska OpenWrt. Skrypt ten został przesłany wraz z raportem.

Ważnymi poleceniami dla poprawnego działania skryptu były:

```
chmod +x /etc/init.d/S99server
/etc/init.d/S99server enable
```

Poprawność działania skryptu można było przetestować manualnie w następujący sposób:

```
/etc/init.d/S99server start
/etc/init.d/S99server stop
```

3.5 Problemy z biblioteką do obsługi GPIO

W podrozdziale 3.3 poleceniem `pip install UniversalGPIO` pobrana została biblioteka `UniversalGPIO`. Nie był to jednak oczywisty wybór z racji na fakt, że w poprzedniej implementacji projektu, wykorzystywana była biblioteka `pigpio` oraz związany z nią daemon `pigpiod`.

Z powodu problemów z konfiguracją daemona na systemie docelowym, zdecydowałem się na zmianę implementacji klasy `GPIOModule` tak, aby wykorzystywała inną bibliotekę do obsługi przycisków oraz diód. Dzięki zastosowaniu interfejsu `HardwareModule`, który stanowił abstrakcję na operacje wykonywane na GPIO, zmiany w projekcie ograniczyły się tylko do tej pojedynczej klasy.

4 Podsumowanie

Ćwiczenie laboratoryjne miało na celu praktyczne zapoznanie się z procesem uruchamiania aplikacji w systemie OpenWrt z wykorzystaniem gotowego, prekompilowanego obrazu. Przeniesienie wcześniej stworzonego rozwiązania — Systemu Kolejowania Zleceń Produkcyjnych — do nowego środowiska pozwoliło na pogłębienie wiedzy z zakresu konfiguracji systemu, instalacji zależności oraz integracji aplikacji z mechanizmem autostartu w OpenWrt.

Ćwiczenie miało przede wszystkim charakter edukacyjny i pozwoliło na zdobycie cennych doświadczeń związanych z praktycznym zastosowaniem systemu OpenWrt w kontekście systemów wbudowanych.

Podczas realizacji zadania napotkano jednak pewne trudności techniczne, w szczególności:

- Problemy z uruchomieniem i konfiguracją demona `pigpiod`, niezbędnego do działania wcześniej wykorzystywanej biblioteki `pigpio`, co uniemożliwiło jej zastosowanie w środowisku OpenWrt.
- Konieczność dostosowania aplikacji do nowej biblioteki obsługującej GPIO — `UniversalGPIO`. Dzięki zastosowanej architekturze z warstwą abstrakcji sprzętowej (`HardwareModule`) modyfikacje ograniczyły się jedynie do jednej klasy.
- Ręczna konfiguracja sieci oraz systemowych usług startowych (takich jak skrypt `S99server`), wymagająca dobrej znajomości struktury plików i mechanizmów OpenWrt.

Pomimo tych trudności, udało się uruchomić aplikację w docelowym środowisku i zapewnić jej pełną funkcjonalność. Ćwiczenie było wartościowe zarówno z perspektywy technicznej, jak i dydaktycznej.

Literatura

- [1] dr hab. inż. Wojciech Zabołotny, prezentacja do *Wykładu 8*.
- [2] Oficjalna strona środowiska OpenWrt - <https://downloads.openwrt.org/releases/24.10.1/targets/bcm27xx/bcm2711/>, dostęp 4 czerwca 2025
- [3] Biblioteka Tornado - <https://www.tornadoweb.org/en/stable/>, dostęp 4 czerwca 2025
- [4] Biblioteka UniversalGPIO - <https://pypi.org/project/UniversalGPIO/>, dostęp 4 czerwca 2025