UMCS

UNIWERSYTET MARII CURIE-SKŁODOWSKIEJ
W LUBLINIE
Wydział Matematyki, Fizyki i Informatyki

Kierunek: **informatyka**

Jan Bylina
nr albumu: 303827

Projekt oraz implementacja systemu gromadzenia rozproszonych danych z wykorzystaniem technologii LoRa

Design and implementation of the distributed data collection system using LoRa technology

Praca licencjacka
napisana w Katedrze Oprogramowania Systemów Informatycznych
Instytutu Informatyki UMCS
pod kierunkiem **dr hab. Przemysława Stpiczyńskiego**

Lublin 2023

Spis treści

Wstęp

Tu treść wstępu WSTĘP WSTEP ——

Rozdział 1

Rozdział — tutorial

1.1 Sekcja A

W tabeli 1.1 widzimy przykład tabeli z nagłówkiem i odnośnikiem. Tabele tworzymy z nagłówkiem na górze oraz opcją `[t]`. Natomiast na rysunku 1.1 — widzimy przykład rysunku z nagłówkiem i odnośnikiem. Rysunki tworzymy z nagłówkiem pod spodem oraz opcją `[b]`. Rysunki powinny być w formacie PDF; jeśli to niemożliwe, to PNG (w wysokiej rozdzielczości); a ostatecznie JPG (jak tu). Jeśli chcemy sterować rozmiarem, to zwykle najwygodniej użyć `width=...` Ponadto możemy odwoływać się do bibliografii

Jeśli chodzi o wzory, możemy złożyć je na kilka sposobów, w zależności od potrzeb — w tekście: $e = \lim_{n\to\infty} \left(1 + \frac{1}{n}\right)^n$, wyniesiony do osbnej linii (warto zwrócić uwagę, że ten i kolejny są złożone nieco inaczej niż pierwszy):

$$e = \lim_{n\to\infty} \left(1 + \frac{1}{n}\right)^n,$$

a także wyniesiony z numerem:

$$e = \lim_{n\to\infty} \left(1 + \frac{1}{n}\right)^n. \tag{1.1}$$

Do tego oostatniego możemy się odwołać: (1.1). No i oczywiście listingi — listing 1.1 pokazuje, jak zrobić to w miarę poprawnie...

Rysunek 1.1: Przykładowy rysunek

Tabela 1.1: Przykładowa tabela

slkdjfslj	sdkskd	s;lkdsdk
slkjd	skljdsldj	skljdsjdsldj
sljkdslkjd	woieupowiepoweiwiewp	weoiw eppowie wpo

```
1 tab[0:n] = dem[nRows][nCols]; //?
2 #pragma acc data copy(tab [0:n], slope [0:n])
```

Listing 1.1: Jakieś dwie linijki w C++ (z OpenACC)

Rozdział 2

Wykorzystane narzędzia, technologie i protokoły

2.1 Urządzenia wykorzystywane w projekcie

2.1.1 ESP32

ESP32 to jednoukładowy mikrokontroler, zaprojektowany i produkowany przez firmę Espressif Systems. Jego najważniejsze cechy to:

- energooszczędny procesor RISC o częstotliwości do 240 MHz
- 520 kB pamięci SRAM
- WiFi 802.11 b/g/n
- Bluetooth
- liczne interfejsy cyfrowe i analogowe, w tym:
  - UART
  - I2C
  - SPI
  - I2S
  - CAN
  - ADC
  - DAC
  - PWM
  - Ethernet MAC

  - USB 2.0

- ...

[5]
Powstało wiele wersji tego układu, różniące się m.in. szybkością procesora, ilością pamięci flash, ilością pinów, ilością interfejsów cyfrowych i analogowych, a także możliwością pracy w trybie bezprzewodowym (WiFi) lub przewodowym (Ethernet)[6]. Najczęściej układ te wykorzystywane różnych projektach IoT, zarówno jako czujniki, jak i serwery.[5]

W projekcie ESP32 zostało wykorzystane w dwóch płytkach TTGO T3 V1.6.1. Jedna z płytek została wykorzystana jako przekaźnik danych pomiędzy siecią LoRa a siecią WiFi, a druga jako cześć systemu zbierania danych z wykorzystaniem LoRa

2.1.2 Rasberry Pi Pico

Rasbperry Pi Pico to płytka z mikrokontrolerem RP2040, zaprojektowana i produkowany przez firmę Raspberry Pi Foundation. Charakteryzuje się ona dwurdzeniowym procesorem ARM Cortex-M0+ o częstotliwości 133 MHz, 264 kB pamięci SRAM oraz 2 MB pamięci flash. Płytka posiada również wiele interfejsów cyfrowych i analogowych, w tym:

- UART
- I2C
- SPI
- I2S
- ADC
- DAC
- PWM
- USB 1.1

[3, 2] Płytka ta jest często wykorzystywana przez hobbystów do różnych projektów IoT, a także jako sterownik silników, czy kontroler robotów.[2]

W projekcie dwie płytki zostały wykorzystane jako część systemu zbierania danych.

2.1.3 STM32

STM32 to rodzina 32 bitowych mikrokonolerów produkowanych przez firmę STMicroelectronics. Bazują one na architekturze ARM Cortex-M, oferują wysoką wydajność i energooszczędność. Cztery główne rodzaje mikrokontrolerów STM32 to:

- Rodzina płytek z częścią kodu F(4/5) i H — overują największą wydajność
- Rodzina płytek z częścią kodu L — oferują największą energooszczędność
- Rodzina płytek z częścią kodu G/C/F(1/3) — do zastosowań ogólnych
- Rodzina płytek z częścią kodu W(L/B/BA) — do zastosowań bezprzewodowych. [SRPAWDZIĆ]

[4] Głównym zastosowaniem STM32 są urządzenia wbudowane, w tym urządzenia medyczne, roboty, samochody, a także urządzenia IoT[NEED CITE]
W projekcie wykorzystano dwie płytki STM32WL55, które zostały wykorzystane jako część systemu zbierania danych.

2.2 Języki programowania i technologie

2.2.1 `C++` for Arduio

—

2.2.2 `C` for STM32

—

2.2.3 `MicroPython` for Rasberry Pi Pico

MicroPython jest językiem

2.3 Bazy danych i pozostałe technologie

2.3.1 InfluxDB 2

—

2.3.2 `Python` for MQTT

2.4 Protokoły komunikacyjne

—

2.4.1 `MQTT`

—

2.4.2 `LoRa`

—

2.4.3 `HTTP`

—

2.5 Bazy danych i pozostałe technologie

2.5.1 InfluxDB 2

—

2.5.2 Docker

—

2.5.3 PlatformIO

—

Rozdział 3

Istniejące rozwiązania

—

3.1 LoRaWAN

—

3.1.1 The Things Network ?

—

3.1.2 ChirpStack ?

—

3.1.3 Loriot ?

—

3.2 Artykuły

—

3.3 Wpisy w sieci i blogach

Rozdział 4

Założenie i Implementacja

W poniższym rozdziale zaprocentowano założenia, potrzeby i projekt projektu Sieci Measch na bazie LoRa

4.1 Podstawowe cele sieci

System ma kilka podstawowych założeń:

- Jeden centralny punkt gromadzenia danych
- Zapisywanie danych do bazy danych szeregów czasowych, w celu ich dalszego przetwarzania
- Zbieranie danych z dużego obszaru
- Niezależność od istniejących metod przesyłu danych (WiFI, Sieci komórkowe, Łączność satelitarna)
- Niezależność od platformy sprzętowej
- Zapewniać możliwie dużą dostarczalność pakietów
- Dane czasowe nie muszą być super dokładne (dopuszczalne są drobne opóźnienia)
- Węzły sieci tylko wysyłają dane, same nie konsumują przychodzących wiadomości

4.2 Części sieci

W celu uzyskania wyżej wspomnianych założeń, zaproponowano system złożonych z kilku części:

4.2.1 Węzły sieci

Węzły sieci to urządzenia wyposażone w moduł LoRa i odpowiednie oprogramowanie pozwalające na pełną obsługę sieci. Gdy węzeł odbierze wiadomość, sprawdza jej poprawność i rozsyła ja dalej w celu zapewnienia jak największego zasięgu i wystarczalności

Urządzenie to może być również wyposażone w różnego rodzaju czujniki, które dostarczają danych bazy danych.

4.2.2 Stacja przekaźnikowa

Stacja przekaźnikowa to urządzenie wyposażone zarówno w moduł LoRa jak i moduł umożliwiający komunikację z siecią Internet (np. moduł WiFI lub moduł bazy Ethernet).

Urządzenie to odbiera przychodzące wiadomości LoRa i przesyła je do brokera wiadomości

4.2.3 Broker wiadomości

Broker wiadomości to program, działający na komputerze mającym dostęp do sieci, umożliwia on wydajną komunikację pomiędzy Stacją Przekaźnikową a Bazą Danych

4.2.4 Baza danych

Baza Danych umożliwiająca zapisywanie sporej ilości danych, uwzględniając również ich czas (baza danych szeregów czasowych). O zapisy danych z brokera wiadomości do bazy danych dba osobny program, który powinien sprawdzać również poprawność tych wiadomości, jak i dbać o to by nie zapisywać powtórzonych wiadomości

4.3 Wiadomości

Każda z wiadomości przesyłanych za pomocą tych sieci powinna mieć formę jak zaprezentowano na listingu 4.1

Zawiera ona pola:

- `ttl` — [Ang. time to live — czas życia] Wartość określająca maksymalną liczbę skoków pomiędzy węzłami sieci. Domyślnie wynosi 10, może zostać wydłużona w zależności od wielkości planowanej sieci
- `m_id` — UUID [1] wiadomości, gwarantujący niepowtarzalności tej wiadomości. Ułatwia również jej dalsze przetwarzanie

- `d_id` — numer identyfikacyjny urządzenia z którego pochodzi wiadomość
- `values` — słownik zawierający dane z urządzenia, do zapisania w bazie

4.4 Obsługa protokołu

4.4.1 Działanie węzłów

Wiadomości generowane są przez węzły sieci, zawierając wszystkie niezbędne pola (wymienione wyżej) i odczyty z czujników zamieszczonych na węźle. Następnie zostaje ona rozesłana do wszystkich węzłów w zasięgu (broadcasting).

Węzeł odbierając wiadomość, sprawdza jej poprawność (czy jest odpowiednio sformatowana, czy zawiera wszystkie potrzebne pola), i jeżeli wiadomość jest poprawna,a pole `ttl` jest większe od 0 rozsyła wiadomość dalej. Sprawdzanie wiadomości odbywa się by wyeliminować wiadomości niepoprawne z sieci.

4.4.2 Działanie przekaźnika

Przekaźnik odbiera wiadomości, i przesyła je do brokera wiadomości. Nie sprawdza poprawności wiadomości by zapewnić maksymalną wydajność i niezawodność.

4.4.3 Działanie bazy danych i programu zapisującego dane

Program pobiera kolejne wiadomości od brokera i przetwarza je w kolejności:

1. Sprawdzenie poprawności wiadomości
2. Sprawdzenie czy wiadomość nie została już sprawdzona (na podstawie pola `m_id`)
3. Zapisanie danych ze słownika `values` do bazy danych i przyporządkowanie ich do `d_id`
4. Zapisanie w pamięci operacyjnej `m_id`, potem do wykorzystania w kroku 2

4.5 Implementacja

4.5.1 Implementacja przykładowych węzłów sieci

W wyniku pracy nad systemem zbierania danych przygotowano implementację węzłów sieci w wykonrzystaniem 3 platform sprzętowych:

- Rasberry Pi Pico +
- ESP32 TTYGO
- STM32 WL...

```
{
    "d_id": "id_233",
    "values": {
        "temp": "21",
        "hum": "50",
        "press": "1000",
        "light": "100",
        "co2": "1000",
        "pm25": "10",
        "pm10": "20"
    },
    "ttl": 10,
    "m_id": "eaa17a7b-9388-43b6-9310-731c942fc6b9"
}
```

Listing 4.1: Przykładowa wiadomość przesyłana przez system

Rozdział 5

Wdrożenie i testy

Rozdział 6

Wnioski i perspektywy rozwoju

Spis listingów

Spis tabel

Spis rysunków

Bibliography

[1] ietf. *A Universally Unique IDentifier (UUID) URN Namespace*. 2005. (Visited on 04/23/2023).

[2] Raspberry Pi Foundation. *Raspberry Pi Documentation*. 2023. URL: https://www.raspberrypi.com/documentation/microcontrollers/raspberry-pi-pico.html (visited on 04/16/2023).

[3] Raspberry Pi Foundation. *Raspberry Pi Pico Datasheet*. 2023. URL: https://datasheets.raspberrypi.com/pico/pico-datasheet.pdf (visited on 04/16/2023).

[4] STMicroelectronics. *STM32 Overwiev*. 2023. (Visited on 04/23/2023).

[5] Espressif Systems. *ESP32 Datasheet*. 2023. URL: https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf (visited on 04/16/2023).

[6] Espressif Systems. *ESP32 SoCs*. 2023. URL: https://www.espressif.com/en/products/socs (visited on 04/16/2023).