МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ЛЬВІВСЬКА ПОЛІТЕХНІКА



**АВТОМАТИЗОВАНЕ ПРОЕКТУВАННЯ**

**КОМП’ЮТЕРНИХ СИСТЕМ**

Завдання 2: «Розробка клієнт серверної архітектури»

***Виконав:***

ст. гр. КІ-404

Зелений Т.Р.

***Прийняв****:*

Федак П.Р.

Львів – 2024

**ЗАВДАННЯ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Варіант** | **Ім’я студента** | **Група** | **Гра** | **Config формат** |
| **7** | Зелений Тарас Романович | КІ-404 | tik-tac-toe 3x3 | XML |

1. Створити просту схему зв'язку SW(клієнт) <-> UART <-> HW(сервер).
2. Клієнт повинен послати повідомлення на сервер. Сервер повинен модифікувати повідомлення і відправити його назад клієнту.
3. Створіть YML файл з наступними функціями:
   1. Зберіть всі двійкові файли (створіть скрипти в папці ci/, якщо потрібно);
   2. запуск тестів;
   3. створити артефакти з двійковими файлами та тестовими звітами;

**ТЕОРЕТИЧНИЙ МАТЕРІАЛ**

**1. Створення схеми зв'язку SW(client) <-> UART <-> HW(server)**

**UART** (Universal Asynchronous Receiver-Transmitter) – це апаратний інтерфейс для обміну даними між пристроями. Основні моменти:

* *Асинхронний режим:* UART не використовує тактовий сигнал, що означає, що клієнт і сервер повинні мати однакову швидкість передачі (baud rate).
* *Протокол:* Клієнт (SW) і сервер (HW) спілкуються через UART, де дані передаються як послідовність біт. UART перетворює байти даних у послідовність сигналів (битів) і навпаки на обох кінцях комунікації.

**Схема**

* **SW (client)** – це програмна частина (наприклад, програма на ПК чи мікроконтролері), яка надсилає повідомлення до HW.
* **HW (server)** – це апаратна частина (наприклад, інший мікроконтролер або плата), яка отримує повідомлення, модифікує його і відправляє назад клієнту через той же UART інтерфейс.

**2. Надсилання повідомлення від клієнта до сервера і його модифікація**

Завдання передбачає, що клієнт надсилає певне повідомлення до сервера через UART. Сервер повинен обробити це повідомлення (наприклад, змінити символи, додати дані, перетворити формат) і надіслати його назад клієнту.

**Приклад процесу:**

* Клієнт: надсилає текст «Hello, server!»
* Сервер: отримує текст, додає до нього «Response: » і відправляє назад повідомлення " Response: Hello, server!"

**3. Створення YML файлу з функціоналом**

**YML** (YAML Ain't Markup Language) – це формат даних для зручного опису конфігурацій. Він широко використовується в автоматизації завдань, зокрема в системах CI/CD для опису кроків збірки, тестування і створення артефактів.

**Основні елементи YML файлу:**

* Збірка всіх бінарних файлів: У YML файлі може бути описано, як викликати скрипти збірки (наприклад, make, cmake тощо), які знаходяться в папці ci/.
* Запуск тестів: Ви повинні прописати команду, яка запускає тестові сценарії, можливо з використанням таких інструментів, як pytest, unittest або інших, залежно від платформи.
* Створення артефактів: Після успішної збірки потрібно створити артефакти – це можуть бути бінарні файли, звіти про тести чи інші важливі файли. У YML це може виглядати як крок з виведенням результатів у певну директорію.

**ВИКОНАННЯ РОБОТИ**

**ЗАПУСК СЕРВЕРНОЇ СТОРОНИ(HW)**

1. Клонуйте репо за допомогою наступної команди нижче:

***Команда:*** *https://github.com/Taras-Zelenyy/csad2425ki404zelenyytr07.git*

1. Відкрийте git bash
2. Перейдіть до feature/develop/task2. Використовуйте наступну команду:

***Команда:*** git checkout feature/develop/task2

1. Знайдіть наступний файл за наступним шляхом: your\_path\server\server.ino
2. Відкрийте Arduino IDE, виберіть порт (у мене це COM3), плату та завантажте код.

**ЗАПУСК КЛІЄНТСЬКОЇ СТОРОНИ(SW)**

1. Відкрийте pull feature/develop/task2
2. Перейдіть на вкладку Action
3. Виберіть останню збірку проекту
4. Завантажте артефакти
5. Розархівуйте завантажену папку
6. Перейдіть до «your\_path\build-artifacts\Debug\client.exe»
7. Двічі клацніть на client.exe

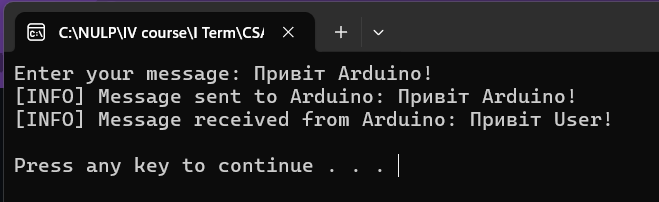
**ЯК ЦЕ ПРАЦЮЄ**

Користувач пише повідомлення у клієнтській консолі. Сервер обробляє введене користувачем повідомлення і повертає відповідь.

***Приклад 1:***

***User input:*** Привіт Arduino!

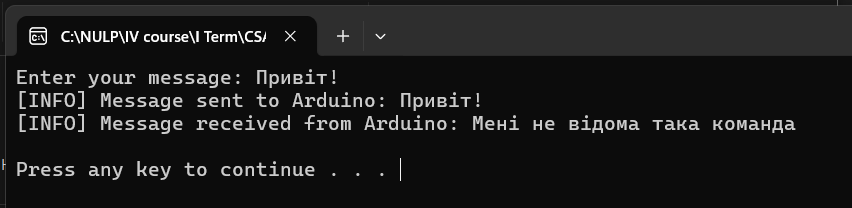
***Server answer:*** Привіт User!



**Приклад 2:** (Користувач вводить будь яке інше повідомлення)

***User input:*** Привіт!

***Server answer:*** Мені не відома така команда



**ВИСНОВОК**

Реалізовано просту комунікаційну схему, де програмний клієнт (SW) взаємодіє з апаратним сервером (HW) через інтерфейс UART. Така архітектура дозволяє асинхронний обмін даними між пристроями без потреби у тактовому сигналі, забезпечуючи ефективний зв'язок між ними.

Реалізовано передачу повідомлення від клієнта до сервера через UART. Сервер обробляє отримане повідомлення, модифікує його, а потім надсилає його назад клієнту.

Створено YML файлу з функціоналом для автоматизації:

* Збірка всіх бінарних файлів: Налаштовано автоматичну збірку всіх необхідних бінарних файлів для забезпечення безперервної інтеграції.
* Створення артефактів: Налаштовано створення артефактів, включаючи бінарні файли та звіти тестування, що забезпечує збереження та доступність результатів роботи для подальшого аналізу.

**СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ**

1. Шпіцер А.С. Instructions for practical tasks and coursework from «Computer systems automated design» – методичка. НУ «Львівська Політехніка», 12 с.