МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ЛЬВІВСЬКА ПОЛІТЕХНІКА

Кафедра ЕОМ

АВТОМАТИЗОВАНЕ ПРОЕКТУВАННЯ КОМП'ЮТЕРНИХ СИСТЕМ

Task 3. Implement Server (HW) and Client (SW) parts of game (FEF)

Виконав:

Ст. гр. КІ-402

Середа Д.А.

Прийняв: Федак П.Р.

Львів 2024

Мета: Реалізувати серверну (HW) і клієнтську (SW) частину гри (FEF) згідно варіанту.

Завдання

Game requirements:

The simple games will be used as projects for development (see table#1). Configuration and state saving should be done using config format from table#1. Server part should execute on HW. SW as a client.

Game should have a menu that allows to configure parameters:

Play modes:

· Man vs AI

· Man vs Man

· AI vs AI:

  ○ Random move;

  ○ Win strategy;

Actions:

· New;

· Save;

· Load;

Варіант:

19	tik-tac-toe 3x3	XML

Теоретичні відомості

Комунікаційна схема SW (клієнт) <-> UART <-> HW (сервер): Комунікаційна схема між клієнтом і сервером через UART (Universal Asynchronous Receiver-Transmitter) дозволяє обмін повідомленнями між програмним забезпеченням (SW) та апаратним забезпеченням (HW). UART є

послідовним інтерфейсом передачі даних, що широко використовується в системах з низьким енергоспоживанням. У цій схемі клієнт (SW) надсилає повідомлення через UART до сервера (HW), який обробляє або модифікує його та відправляє назад на клієнт. UART працює за асинхронним протоколом, тобто передача даних не потребує синхронізації, що спрощує апаратну реалізацію.

Передача і модифікація повідомлень: Згідно з вимогами, клієнт (SW) має передати певне повідомлення серверу (HW), який змінює його (наприклад, додає префікс чи змінює текст) і надсилає змінене повідомлення назад клієнту. Це може бути реалізовано за допомогою протоколу обміну, де клієнт посилає запит, а сервер відповідає зі зміненими даними. Це допомагає не тільки перевірити двосторонню комунікацію, але й дає змогу серверу виконати певну обробку даних.

Файл конфігурації YML:

YML-файли (YAML Ain't Markup Language) часто використовуються для опису конфігурацій та робочих процесів автоматизації, особливо в контексті CI/CD (Continuous Integration/Continuous Deployment). У цьому завданні YML файл має містити наступні етапи:

- Збірка всіх бінарних файлів: Цей етап передбачає компіляцію всього коду, а для цього можна створити відповідні скрипти у папці `ci/`.
- Запуск тестів: Автоматизовані тести дозволяють перевірити роботу системи на різних етапах. Тести можна створити для перевірки успішності зв’язку, коректності обробки повідомлень та передачі даних через UART.
- Створення артефактів: Результати збірки (бінарні файли) та звіти тестів мають бути збережені як артефакти, що дозволяє їх використовувати в наступних етапах або аналізувати у разі виникнення проблем.

Клієнтська частина взаємодіє з сервером для відправлення ходів та отримання оновлень стану гри. Сервер відповідає за перевірку правильності ходів, виконання логіки гри, перевірку на виграш чи нічию, а також реалізацію стратегій AI. Клієнт лише візуалізує гру та дозволяє користувачеві взаємодіяти з нею через інтерфейс. Конфігураційний файл дозволяє зберігати і відновлювати стан гри, що робить її більш зручною для користувача.

Деталі реалізації

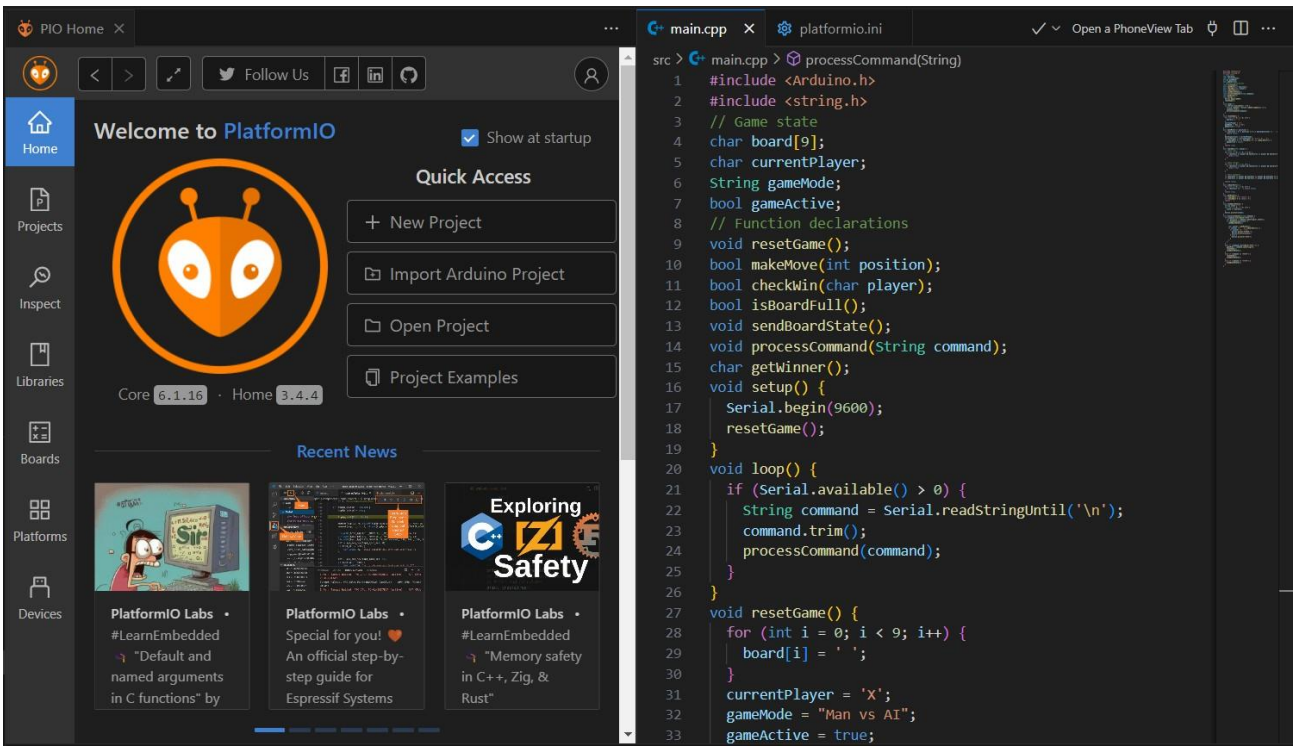

Рис.1. Реалізація сервера за допомогою PlatformIO у Visual Studio Code

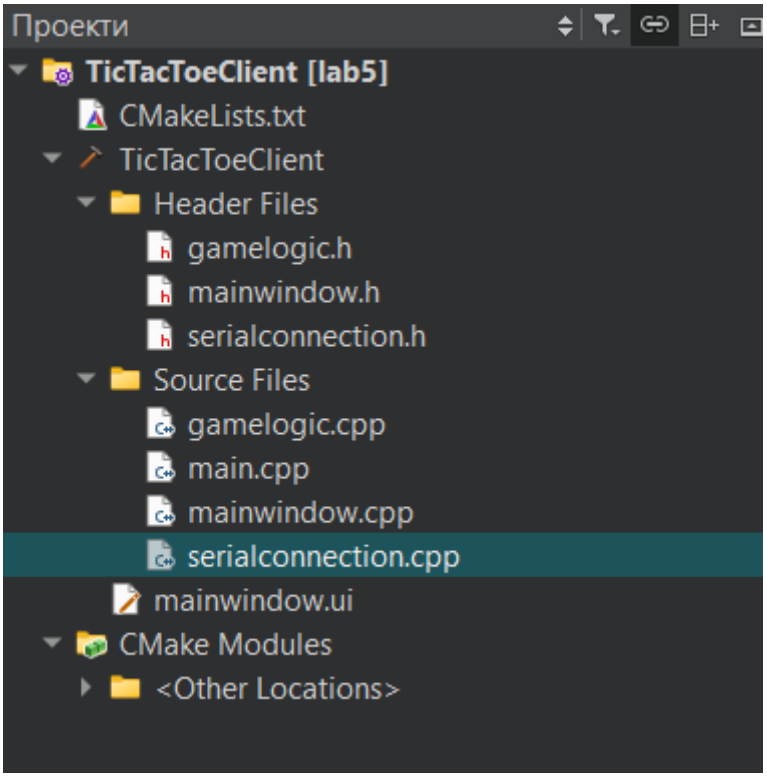

Рис.2. Реалізація клієнтської частини гри

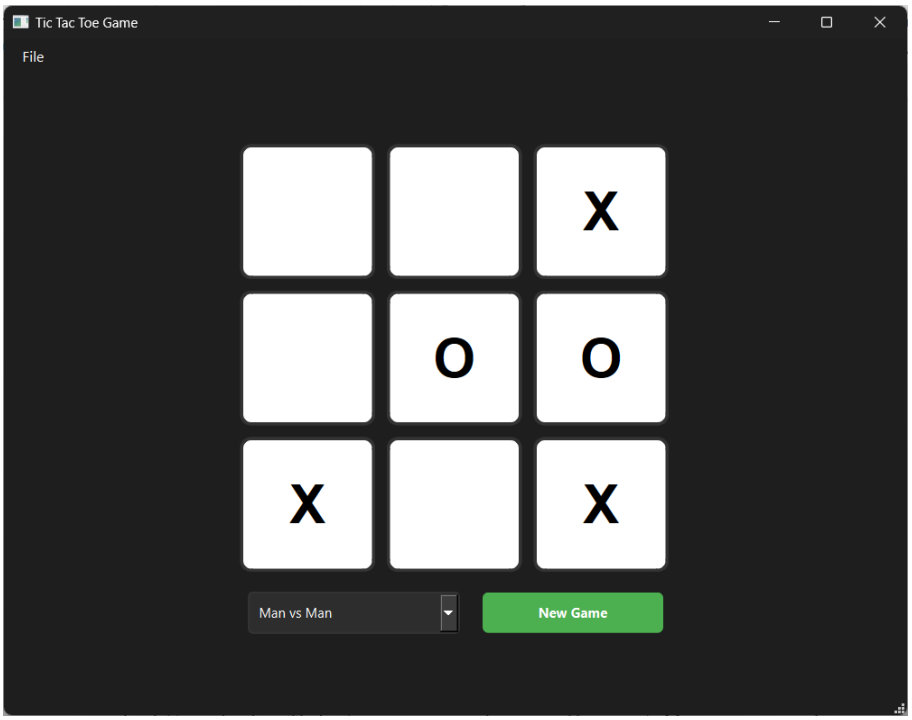

Рис.3. Головна сторінка з ігровим полем

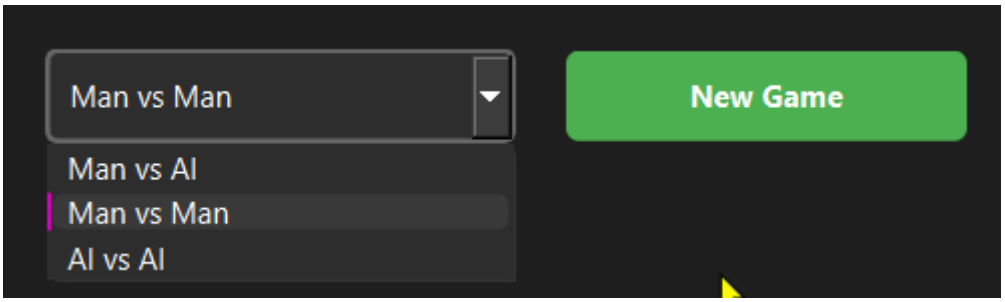

Рис.4. Випадаючий список з вибором режимів гри

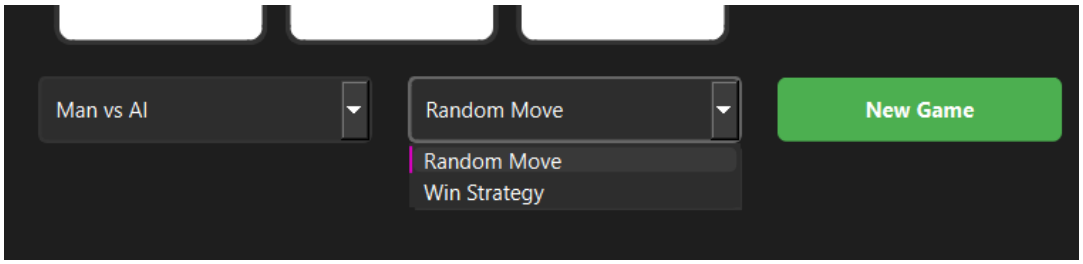

Рис.5. Випадаючий список з для режиму, де присутнє AI

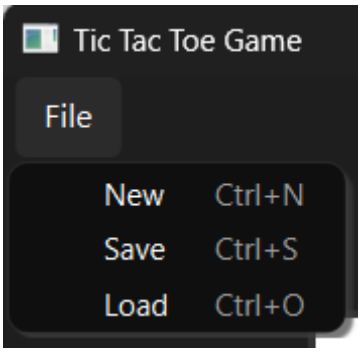

Рис. 6. Список дій, які можна зробити з ігровим полем

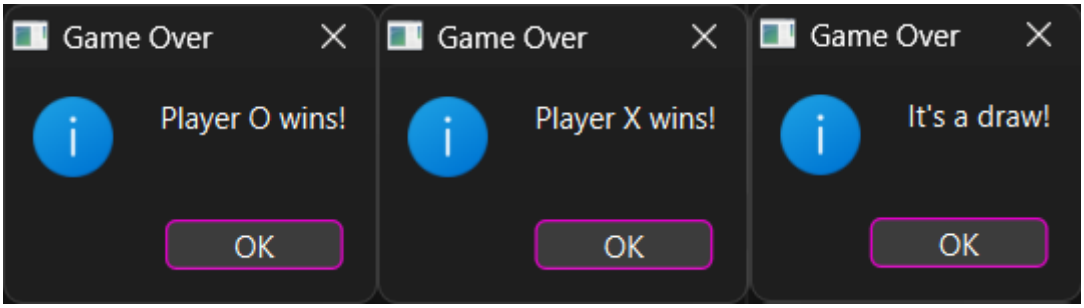

Рис. 7. Діалогові вікна, які відображаються при відповідних результатах гри

Висновок:

Виконуючи лабораторну роботу я створив просту гру Tic-Tac-Toe, де програмний клієнт (SW) взаємодіє з апаратним сервером (HW) через інтерфейс UART. Така архітектура дозволяє здійснювати асинхронний обмін даними між пристроями без необхідності синхронізації тактового сигналу, що значно спрощує організацію комунікації та забезпечує ефективний і надійний зв'язок між клієнтом і сервером. Реалізував весь необхідний функціонал з режимами гри та її діями (збереження, завантаження з XML), які повинні виконуватись.

Список використаної літератури:

1. GitHub Docs, https://docs.github.com/
2. CSAD instructions for practical tasks and coursework