ЗВІТ

з лабораторної роботи **№2**

з дисципліни “Автоматизоване проектування комп’ютерних систем”

Виконала:
ст.гр. КІ-401
Кріль Катерина
Перевірив:
Федак П. Р.

Львів - 2024

Ініціалізація Git репозиторію

Хід роботи:

1. Створіть просту комунікаційну схему SW (клієнт)
<-> UART <-> HW (сервер).
2. Клієнт повинен надіслати повідомлення на сервер. The
сервер повинен змінити повідомлення та надіслати його назад до клієнт.
3. Створіть файл YML із такими функціями:
    a. зібрати всі двійкові файли (створити сценарії в папці ci/if потреба);
    b. запустити тести;
    в. створювати артефакти з бінарними файлами та звітами про тестування;
4. Необхідні кроки.

Student number	Game	config format
1	tik-tac-toe 3x3	XML
2	rock paper scissors	JSON
3	tik-tac-toe 3x3	INI
4	rock paper scissors	XML
5	tik-tac-toe 3x3	JSON
6	rock paper scissors	INI
7	tik-tac-toe 3x3	XML
8	rock paper scissors	JSON
9	tik-tac-toe 3x3	INI
10	rock paper scissors	XML
11	tik-tac-toe 3x3	JSON
12	rock paper scissors	INI
13	tik-tac-toe 3x3	XML
14	rock paper scissors	JSON
15	tik-tac-toe 3x3	INI
16	rock paper scissors	XML
17	tik-tac-toe 3x3	JSON
18	rock paper scissors	INI
19	tik-tac-toe 3x3	XML
20	rock paper scissors	JSON
21	tik-tac-toe 3x3	INI
22	rock paper scissors	XML
23	tik-tac-toe 3x3	JSON
24	rock paper scissors	INI
25	tik-tac-toe 3x3	XML
26	rock paper scissors	JSON
27	tik-tac-toe 3x3	INI
28	rock paper scissors	XML
29	tik-tac-toe 3x3	JSON
30	rock paper scissors	INI
31	tik-tac-toe 3x3	XML
32	rock paper scissors	JSON
33	tik-tac-toe 3x3	INI
34	rock paper scissors	XML
35	tik-tac-toe 3x3	JSON

Табл.1 Завдання

Виконання роботи:

Створення схеми комунікації:

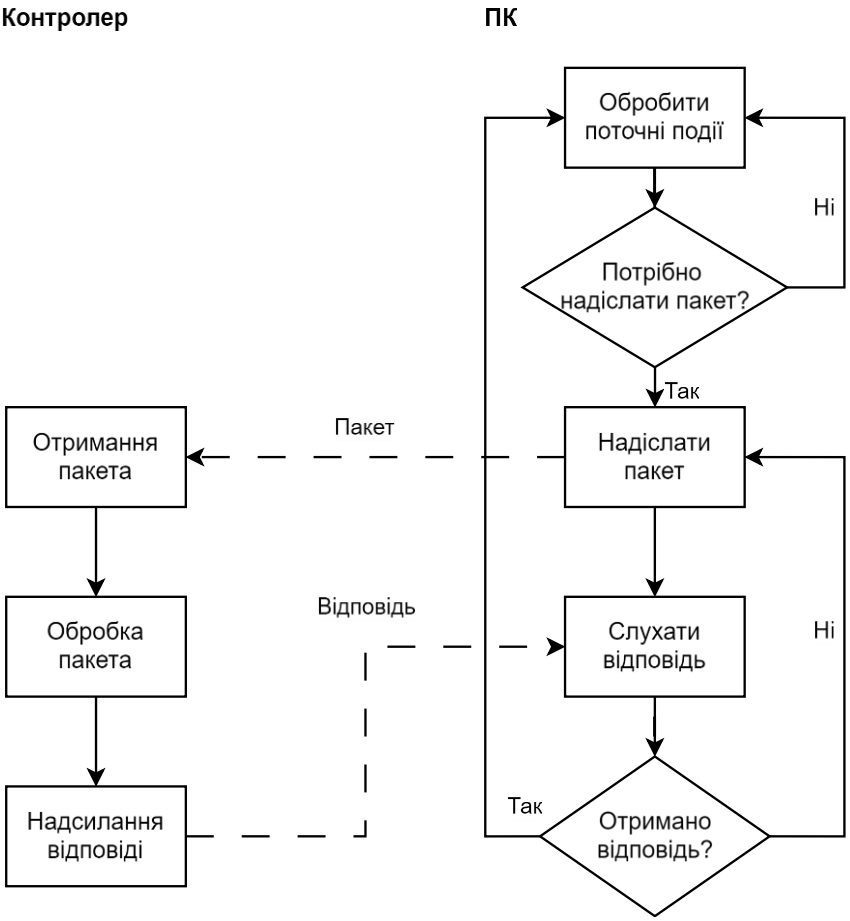

Рис. 1 Схема комунікації

Схема комунікації на базі UART між Python-кодом і PSoC6 BLE включає взаємодію між компонентами шляхом передачі та отриманням даних через UART-інтерфейс. У даній схемі Python-код виступає як джерело даних, що надсилає контрольні пакети до PSoC6 BLE, який в свою чергу слухає ці пакети, обробляє їх та відправляє відповіді назад до Python-коду через UART. Плата слухає контрольні пакети, обробляє їх та надсилає відповіді назад Python-коду через UART.

У цій схемі UART використовується для послідовного обміну даними між Python-кодом та PSoC6 BLE. Python-код надсилає контрольні пакети через UART до PSoC6 BLE, який приймає ці пакети, обробляє їх та генерує відповіді, які також відправляє назад до Python-коду через UART. У свою чергу, плата слухає контрольні пакети, обробляє їх та надсилає відповіді назад через UART.

Ця схема комунікації UART може бути використана для взаємодії між Python-кодом та PSoC6 BLE для виконання різних завдань, таких як керування пристроями, збір та обробка даних тощо.

Створення YAML файла:

Цей конфігураційний YAML файл (ci-cd.yaml) описує конвеєр для неперервної інтеграції та постійної доставки (CI/CD). Він використовує синтаксис YAML для визначення кроків (stages) та відповідних завдань (jobs), які потрібно виконати під час кожного етапу конвеєру.

Цей файл містить опис двох етапів: build та test. Кожен етап містить список команд, які потрібно виконати. Наприклад, етап build містить команди cd ci/ та ./setup_python.bat, а етап test містить команду ./run_tests.bat.

Такий конфігураційний файл може бути використаний в інструментах для автоматизації розгортання та управління інфраструктурою, таких як Jenkins, GitLab CI/CD, CircleCI та інших. Його використання дозволяє автоматизувати процеси збирання, тестування та розгортання програмного забезпечення, що сприяє покращенню процесів розробки та розгортання програмного забезпечення.

```
# Filename: ci-cd.yaml

stages:
  - build
  - test

build:
  stage: build
  script:
    - cd ci/
    - ./setup_python.bat

test:
  stage: test
  script:
    - ./run_tests.bat
```

ВИСНОВОК:

В ході роботи №2 було створено схему комунікації на основі інтерфейса UART та створено конвеєр на основі YAML формату.