МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

АВТОМАТИЗОВАНЕ ПРОЕКТУВАННЯ КОМП'ЮТЕРНИХ СИСТЕМ

Лабораторна робота №2

Виконав: ст. гр. КІ-405
Легкобит Н.В

Прийняв:
Шпіцер А.С

Львів 2024

Тема: ознайомлення з комунікаційними інтерфейсами.

Порядок виконання лабораторної роботи:

1. Create a simple schema SW(client) <-> UART <-> HW(bridge) <-> HW i-fase <-> HW(server).

   NOTE: that SW(client) is NOT a terminal or other downloaded SW. It is SW developed by students.

2. The client should send a message through the bridge to the server. The server should modify the message and send it back to the client through the bridge.
3. Required steps.

Виконання роботи

Налаштування com0com

1. Встановлено com0com на Windows.
2. Створено пару віртуальних портів COM12 і COM11 за допомогою командного рядка com0com:

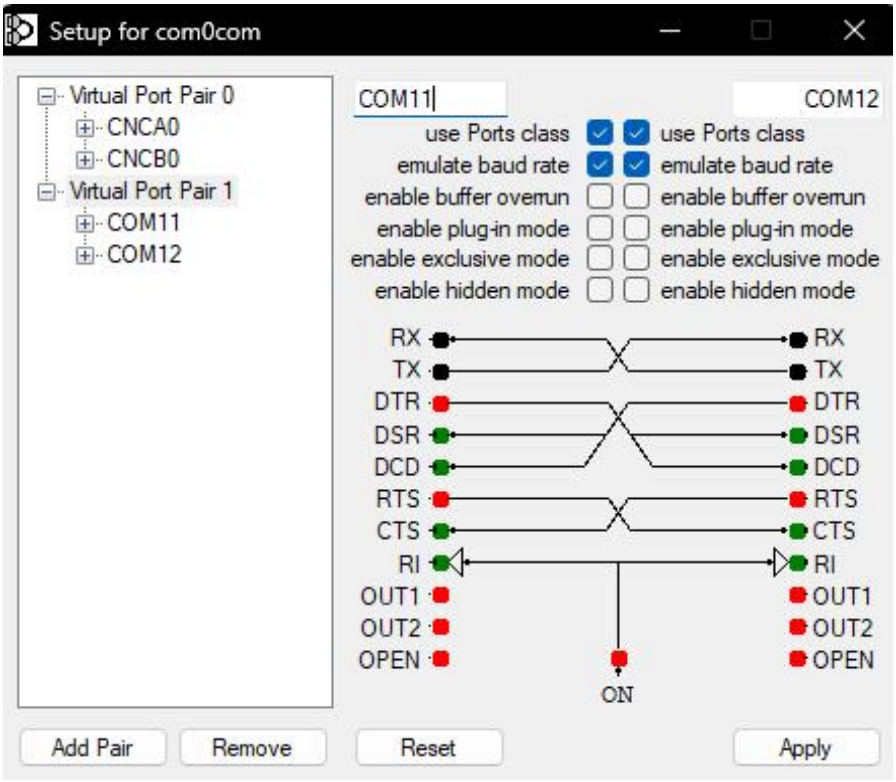


Рис. 1. Налаштування com0com та створення віртуальних портів.

2 Створення Python-сервера

1. Розробив скрипт server.py, який прослуховує порт COM11, приймає повідомлення від клієнта, змінює їх та відправляє назад.

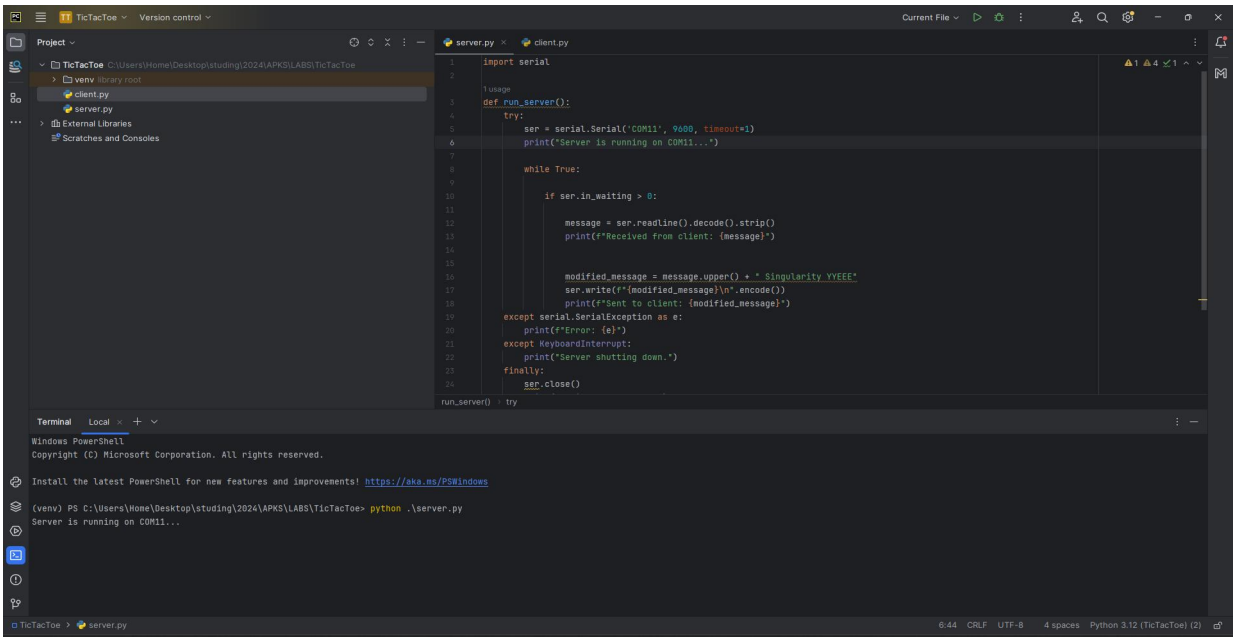


Рис. 2. Запуск сервера та вивід отриманих даних.

3 Створення Python-клієнта

1. Створив скрипт client.py, який підключається до порту COM12, надсилає повідомлення серверу та отримує відповідь.

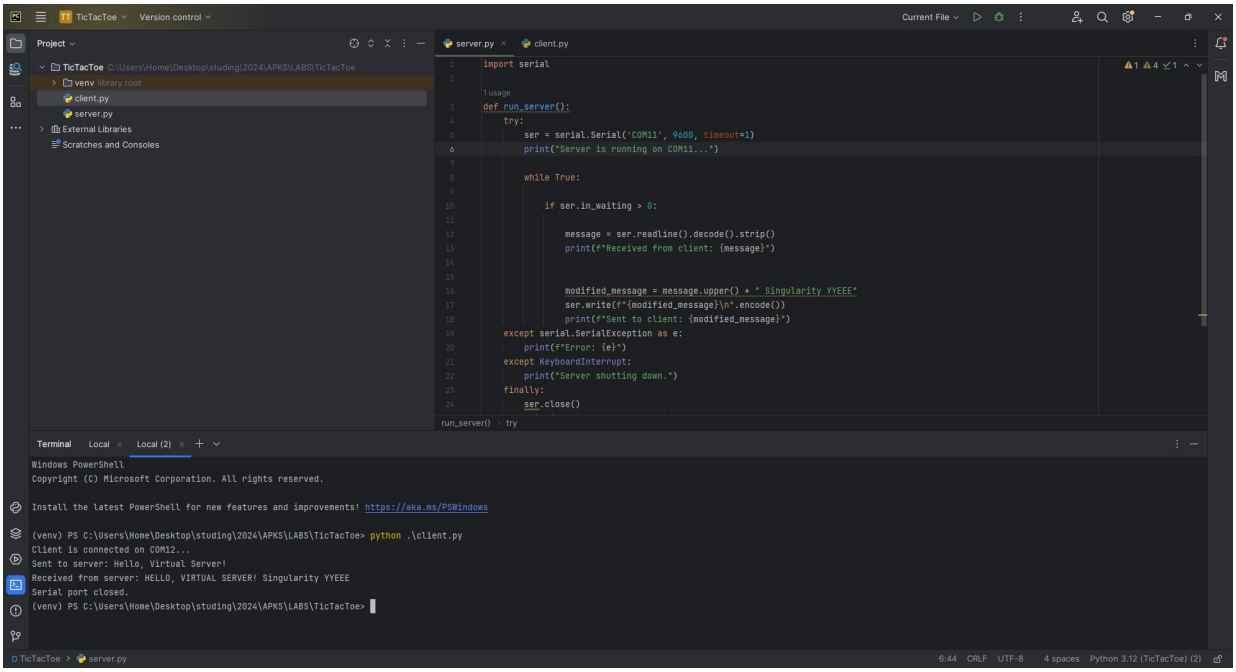


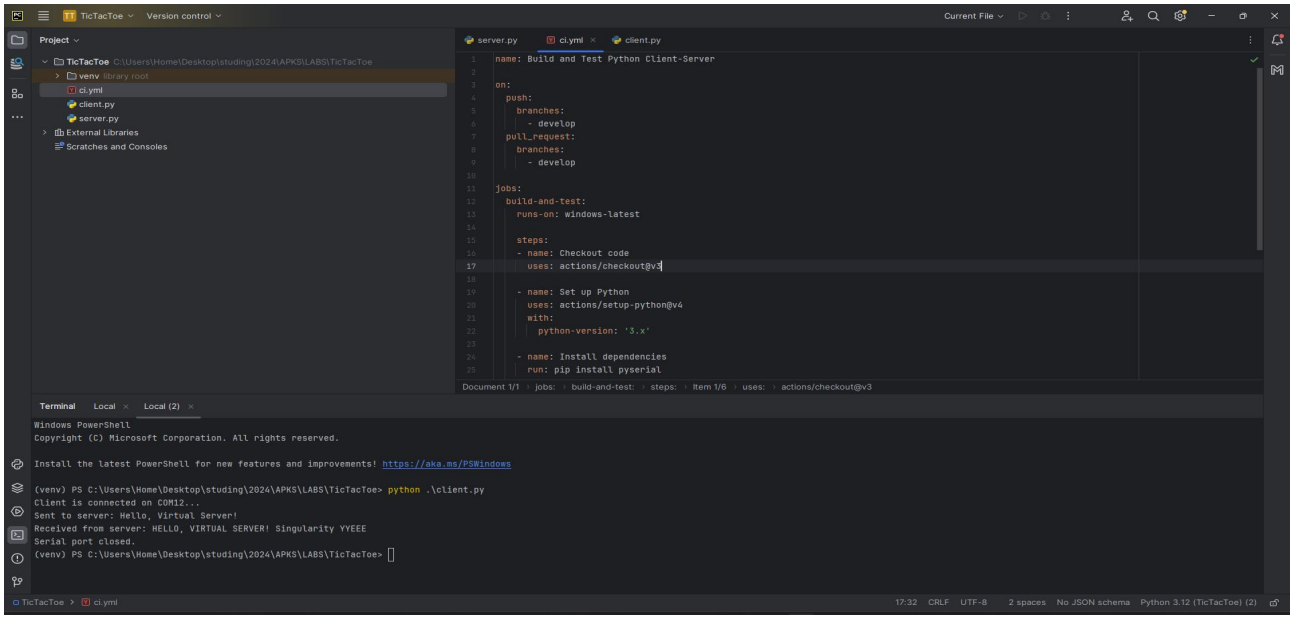
Рис. 3. Запуск клієнта та вивід отриманих даних.

4. Результати

- Реалізував емуляцію зв'язку між програмним клієнтом та сервером за допомогою віртуальних портів com0com.
- Клієнт успішно надіслав повідомлення на COM12, сервер обробив його на COM11 та повернув змінене повідомлення.
- Забезпечив коректну роботу системи

5. Налаштування YML

Після цього налаштував YML-файл для автоматизації тестування та побудови проекту. У файлі .github/workflows/ci.yml було описано кроки, які включають налаштування середовища, запуск серверного скрипта у фоновому режимі, запуск клієнта та автоматичне тестування зв'язку.



Висновок: на цій лабораторній роботі я ознайомилася з комунікаційними інтерфейсами.

Додатки

client.py

```
import serial
import time


def run_client():
    try:
        ser = serial.Serial('COM12', 9600, timeout=1)
        print("Client is connected on COM12...")

        if ser.is_open:
            message = "Hello, Virtual Server!"
            ser.write(f"{message}\n".encode())
            print(f"Sent to server: {message}")

            time.sleep(1)
            response = ser.readline().decode().strip()
            print(f"Received from server: {response}")
        else:
            print("Failed to open serial port.")
    except serial.SerialException as e:
        print(f"Error: {e}")
    except KeyboardInterrupt:
        print("Client shutting down.")
    finally:
        ser.close()
        print("Serial port closed.")


if __name__ == "__main__":
    run_client()
```

server.py

```
import serial

def run_server():
    try:
        ser = serial.Serial('COM11', 9600, timeout=1)
        print("Server is running on COM11...")

        while True:

            if ser.in_waiting > 0:
```

```
            message = ser.readline().decode().strip()
            print(f"Received from client: {message}")


            modified_message = message.upper() + " Singularity YYEEE"
            ser.write(f"{modified_message}\n".encode())
            print(f"Sent to client: {modified_message}")
    except serial.SerialException as e:
        print(f"Error: {e}")
    except KeyboardInterrupt:
        print("Server shutting down.")
    finally:
        ser.close()
        print("Serial port closed.")

if __name__ == "__main__":
    run_server()
```

ci.yml

```yaml
name: Build and Test Python Client-Server

on:
  push:
    branches:
      - develop
  pull_request:
    branches:
      - develop

jobs:
  build-and-test:
    runs-on: windows-latest

    steps:
    - name: Checkout code
      uses: actions/checkout@v3

    - name: Set up Python
      uses: actions/setup-python@v4
      with:
        python-version: '3.x'

    - name: Install dependencies
      run: pip install pyserial

    - name: Run Server in Background
      run: start /B python server.py
      shell: cmd

    - name: Run Client Test
      run: python client.py

    - name: Cleanup
      run: taskkill /F /IM python.exe
      shell: cmd
```