

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ЛЬВІВСЬКА ПОЛІТЕХНІКА



Автоматизоване проектування комп'ютерних систем

Task 2. Implementing UART Communication

Виконав:

ст. гр КІ - 401

Сторожков О. В.

Прийняв:

Федак П. Р.

Опис теми

Для виконання завдання №2 потрібно виконати наступні задачі:

- 1) Створити просту схему комунікації між клієнтом та сервером використовуючи UART.
- 2) Клієнт повинен відправляти повідомлення серверу, після чого сервер повинен його модифікувати та відправляти клієнту.
- 3) Створити JSON файл який буде включати наступний функціонал:
 - Створення бінарних файлів проєкту
 - Запуск тестування проєкту
 - Створення артефактів з бінарними файлами та результатами тестування.

Теоретичні відомості

UART (Universal Asynchronous Receiver-Transmitter) — пристрій для асинхронної передачі даних по послідовній лінії без тактового сигналу, з використанням стартового і стопового бітів. Застосовується у мікроконтролерах та комп'ютерах.

Arduino — відкрита платформа для створення електронних проєктів, яка поєднує мікроконтролерні плати та просте середовище програмування. Ідеальна для початківців і хобістів, підтримує багато бібліотек та має активну спільноту.

Виконання завдання

1. Написав просту схему комунікації між клієнтом та сервером:

main.py

```
import serial
import time

def setup_serial_port():
    try:
```

```

        port = input("Enter the serial port (e.g., /dev/ttyUSB0 or COM3):
")
        return serial.Serial(port, 9600, timeout=1)
    except serial.SerialException as e:
        print(f"Error: {e}")
        exit(1)

def send_message(message, ser):
    try:
        ser.write((message + '\n').encode())
        print(f"Sent: {message}")
    except serial.SerialException as e:
        print(f"Error sending message: {e}")

def receive_message(ser):
    try:
        received = ser.readline().decode('utf-8', errors='ignore').strip()
        if received:
            print(f"Received: {received}")
        return received
    except serial.SerialException as e:
        print(f"Error receiving message: {e}")
        return None

if __name__ == "__main__":
    ser = setup_serial_port()
    try:
        while True:
            user_message = input("Message to server: ")
            if user_message.lower() == 'exit':
                print("Exiting...")
                break
            send_message(user_message, ser)
            receive_message(ser)
    except KeyboardInterrupt:
        print("Exit!")
    finally:
        if ser.is_open:
            print("Closing serial port...")
            ser.close()

```

test_serial_communication.py

```

import pytest
from unittest.mock import patch, MagicMock
import serial
import sys
import os
sys.path.insert(0,
os.path.abspath(os.path.join(os.path.dirname(__file__), '..')))
from main import send_message, receive_message

def test_send_message():
    mock_serial = MagicMock(spec=serial.Serial)
    send_message("Hello", mock_serial)
    mock_serial.write.assert_called_with(b"Hello\n")

def test_receive_message():
    mock_serial = MagicMock(spec=serial.Serial)
    mock_serial.readline.return_value = b"Test Message\n"
    result = receive_message(mock_serial)
    assert result == "Test Message"

def test_receive_empty_message():
    mock_serial = MagicMock(spec=serial.Serial)
    mock_serial.readline.return_value = b"\n"
    result = receive_message(mock_serial)
    assert result == ""

@patch('builtins.input', return_value='COM3')
def test_serial_port(mock_input):
    mock_serial = MagicMock(spec=serial.Serial)
    mock_serial.portstr = 'COM3'
    port = 'COM3'
    ser = mock_serial
    assert ser.portstr == port

```

task2.ino

```

void setup() {
    Serial.begin(9600);
}

void loop() {
    if (Serial.available() > 0) {
        String receivedMessage = Serial.readStringUntil('\n');
        Serial.println("modified: " + receivedMessage);
    }
}

```

2. Створив YML файл:

build.yml

```
name: task2

on:
  push:
    branches:
      - feature/develop/task2
  pull_request:

jobs:
  build:
    runs-on: windows-latest

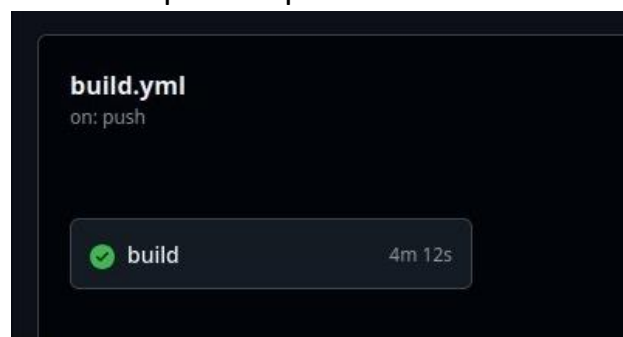
    steps:
      - name: Checkout code
        uses: actions/checkout@v3

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

      - name: Run CI Script
        env:
          PORT: COM3
        run: ./ci-script.bat

      - name: Upload binaries and test reports
        uses: actions/upload-artifact@v3
        with:
          name: build-artifacts
          path: |
            .pio/build/esp32dev/firmware.bin
            python_project/tests/reports/results.xml
```

3. Створив артефакти з бінарними файлами та звітом тестів:



Висновок

Під час виконання завдання №2 було розроблено просту схему комунікації між клієнтом та сервером, а також скрипти для перевірки цілісності проєкту.

Список використаних джерел

1. Wikipedia. "UART".
https://en.wikipedia.org/wiki/Universal_asynchronous_receivertransmitter.
2. Arduino. "What is Arduino?". <https://www.arduino.cc/en/Guide>