

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**  
**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ**  
**імені ІГОРЯ СІКОРСЬКОГО»**

**ФАКУЛЬТЕТ ПРОГРАМНИХ СИСТЕМ ТА ПРИКЛАДНОЇ МАТЕМАТИКИ**

**Кафедра системного програмування і спеціалізованих комп'ютерних систем**

До захисту допущено  
Завідувач кафедри

\_\_\_\_\_ Віталій РОМАНКЕВИЧ

(підпис) (ініціали, прізвище)

“ \_\_\_\_ ” червня 2026 р.

**Дипломний проєкт**

**на здобуття ступеня бакалавра**

**за освітньо-професійною програмою**

**«Системне програмування та спеціалізовані комп'ютерні системи»**

**спеціальності 123 «Комп'ютерна інженерія»**

**на тему: «Програмна система віддаленого керування контролера безпілотних апаратів»**

Виконав (-ла):

студент (-ка) IV курсу, групи КВ-21

(шифр групи)

Кукса Кирило Віталійович

(прізвище, ім'я, по батькові)

(підпис)

Доцент каф. СПіСКС к.т.н., Морозов К.В.

(посада, науковий ступінь, вчене звання, прізвище та ініціали)

(підпис)

Консультант з нормоконтролю, доц. каф. СПіСКС, к.т.н., доц. Клятченко Я.М.

(назва розділу)

(посада, вчене звання, науковий ступінь, прізвище, ініціали)

(підпис)

Рецензент \_\_\_\_\_

(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали)

(підпис)

Засвідчую, що у цій дипломній роботі немає  
запозичень з праць інших авторів без  
відповідних посилань.

Студент \_\_\_\_\_

(підпис)

Київ – 2026 року

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ  
імені ІГОРЯ СІКОРСЬКОГО»**

**ФАКУЛЬТЕТ ПРОГРАМНИХ СИСТЕМ ТА ПРИКЛАДНОЇ МАТЕМАТИКИ**

**Кафедра системного програмування і спеціалізованих комп'ютерних систем**

Рівень вищої освіти – перший (бакалаврський)

Спеціальність 123 «Комп'ютерна інженерія»

Освітньо-професійна програма «Системне програмування та спеціалізовані комп'ютерні системи»

**ЗАТВЕРДЖУЮ**

Завідувач кафедри

\_\_\_\_\_ Віталій РОМАНКЕВИЧ

(підпис)

(ініціали, прізвище)

« \_\_\_\_ » \_\_\_\_\_ 2026р.

**ЗАВДАННЯ**

**на дипломний проєкт студента**

**Кукси Кирила Віталійовича**

(прізвище, ім'я, по батькові)

1. Тема проєкту «Вбудована програмна система дистанційного керування  
безпілотними апаратами» \_\_\_\_\_,

керівник проєкту к.т.н. Морозов К.В., Доцент Кафедри СПіСКС, к.т.н. \_\_\_\_\_,  
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету № 1984-с від «28» травня 2026 р.

2. Термін подання студентом проєкту \_\_\_\_\_

3. Вихідні дані до проєкту: вбудована програмна система дистанційного керування безпілотними апаратами

4. Зміст пояснювальної записки:

1) Аналіз існуючих рішень та актуальність теми дипломного проєкту;

2) Програмне та апаратне забезпечення;

3) Опис розробленої системи: принципи та компоненти.

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслеників, плакатів, презентацій тощо):

- 1) Міжмодульна взаємодія – схема структурна
- 2) Прийом налаштувань безпілотного апарату – схема алгоритму
- 3) Робота наземної станції – схема алгоритму
- 4) Робота безпілотного апарату в активному режимі – схема алгоритму
- 5) Обмін даними в додатку – схема алгоритму
- 6) Фрагменти коду
- 7) Слайди презентації.

6. Консультанти розділів проєкту

| Розділ        | Прізвище, ініціали та посада консультанта | Підпис, дата   |                  |
|---------------|-------------------------------------------|----------------|------------------|
|               |                                           | завдання видав | завдання прийняв |
| Нормоконтроль | Клятченко Я.М., доц. каф. СПіСКС, К.Т.Н   |                |                  |

7. Дата видачі завдання \_\_\_\_ 202\_ р.

#### Календарний план

| № з/п | Назва етапів виконання дипломного проєкту                             | Термін виконання етапів проєкту | Примітка |
|-------|-----------------------------------------------------------------------|---------------------------------|----------|
| 1.    | Видача завдання на дипломне проєктування                              |                                 |          |
| 2.    | Вивчення літератури за тематикою проєкту                              | 15.03.2026                      |          |
| 3.    | Розроблення та узгодження технічного завдання                         | 30.03.2026                      |          |
| 4.    | Аналіз існуючих рішень                                                | 05.04.2026                      |          |
| 5.    | Підготовка матеріалів першого розділу дипломного проєкту              | 15.04.2026                      |          |
| 6.    | Підготовка матеріалів другого та третього розділів дипломного проєкту | 10.05.2026                      |          |
| 7.    | Підготовка графічної частини дипломного проєкту                       | 20.05.2026                      |          |
| 8.    | Оформлення документації дипломного проєкту                            | 25.05.2026                      |          |
| 9.    | Попередній огляд матеріалів проєкту на кафедрі                        | 30.05.2026                      |          |

Студент

\_\_\_\_\_  
(підпис)

Кирило КУКСА

(ім'я, ПРІЗВИЩЕ)

Керівник проєкту

\_\_\_\_\_  
(підпис)

Костянтин МОРОЗОВ

(ім'я, ПРІЗВИЩЕ)

## АНОТАЦІЯ

Кваліфікаційна робота включає пояснювальну записку (52 с., 23 рис. 4 табл.).

Об'єкт розробки – створення програмної системи керування безпілотними апаратами, яка складається з прошивки для польотного контролера, програмного забезпечення на наземну станцію і додаток для налаштувань і керування.

Програмна система дозволяє: здійснювати передачу даних бездротовими каналами зв'язку: комунікацію між польотним контролером і наземною станцією обраним в налаштуваннях засобом, а також комунікацією між польотним контролером або наземною станцією і додатком за допомогою Wi-Fi. За допомогою додатку можливо змінювати налаштування і керувати пристроями в реальному часі. В процесі розробки були використані технології бездротового зв'язку Wi-Fi, TCP-сокети, операційна система FreeRTOS, а також інструмент розробки QT.

В ході розробки:

- проведено аналіз сучасного ПЗ керування безпілотними апаратами
- сформульовані вимоги до запропонованої системи керування безпілотними апаратами;
- розроблена операційна система польотного контролера;
- розроблено алгоритм можливості завантаження нових налаштувань польотного контролера;
- розроблено апаратну і програмну частину наземної станції;
- розроблено алгоритм паралельної роботи станції з польотним контролером і додатком;
- розроблено додаток для зміни налаштувань польотного контролера і керування ним через розширений режим наземної станції.

Упровадження цієї системи дозволить відкрити новий ринок, а також створити сильну базу для навчання власних розробників.

Ключові слова:

ПОЛЬОТНИЙ КОНТРОЛЕР, БЕЗПІЛОТНИЙ АПАРАТ, QT, FREERTOS, WI-FI, TCP, ESP32, ПРОГРАМНА СИСТЕМА.

## **ABSTRACT**

The qualification work includes an explanatory report (52 pages, 23 figures, 4 tables)

The object of development is the creation of a software control system for unmanned aerial vehicles, which consists of firmware for the aircraft controller, firmware for the ground station, and an application for settings and control.

The software system allows: to transfer data via wireless communication channels: communication between the flight controller and the ground station using the means selected in the settings, as well as communication between the flight controller or ground station and the application using Wi-Fi. Using the application, it is possible to change settings and control devices in real time. The development process used Wi-Fi wireless communication technologies, TCP sockets, the FreeRTOS operating system, and the QT development tool.

During the development process:

- analysis of modern UAV control software was conducted
- requirements for the proposed UAV control system were formulated;
- flight controller operating system was developed;
- algorithm for downloading new flight controller settings was developed;
- hardware and software of the ground station were developed;
- algorithm for parallel operation of the station with the flight controller and the application was developed;
- application was developed for changing flight controller settings and controlling it through the advanced mode of the ground station.

The implementation of this system will allow opening a new market, as well as creating a strong base for training new developers.

Keywords:

FLIGHT CONTROLLER, UNMANNED VEHICLE, QT, FREERTOS, WI-FI, TCP, ESP32, SOFTWARE SYSTEM.

| Поз. | Формат | ПОЗНАЧЕННЯ         | НАЙМЕНУВАННЯ                                                                                            | Кількість аркушів | № прим. | Примітки |
|------|--------|--------------------|---------------------------------------------------------------------------------------------------------|-------------------|---------|----------|
|      | A4     | ІАЛЦ.045430.002 ТЗ | Вбудована програмна Система дистанційного Керування безпілотними апаратами Технічне завдання            | 4                 |         |          |
|      | A4     | ІАЛЦ.045430.003 ТП | Вбудована програмна Система дистанційного Керування безпілотними апаратами Відомість технічного проєкту | 2                 |         |          |
|      | A4     | ІАЛЦ.045430.004 ПЗ | Вбудована програмна Система дистанційного Керування безпілотними апаратами Пояснювальна записка         | 52                |         |          |
|      | A1     | ІАЛЦ.045430.005 Д1 | Міжмодульна взаємодія. Схема структурна                                                                 | 1                 |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |
|      |        |                    |                                                                                                         |                   |         |          |

[illegible]

|                                                            |   |
|------------------------------------------------------------|---|
| 1. НАЙМЕНУВАННЯ ТА ГАЛУЗЬ РОЗРОБКИ .....                   | 2 |
| 2. ПІДСТАВА ДЛЯ РОЗРОБКИ.....                              | 2 |
| 3. МЕТА І ПРИЗНАЧЕННЯ РОБОТИ .....                         | 2 |
| 4. ДЖЕРЕЛА РОЗРОБКИ .....                                  | 2 |
| 5. ТЕХНІЧНІ ВИМОГИ.....                                    | 2 |
| 5.1 Вимоги до програмного продукту, що розробляється ..... | 3 |
| 5.2 Вимоги до апаратного забезпечення .....                | 3 |
| 5.3 Вимоги до програмного забезпечення .....               | 3 |
| 6. ЕТАПИ РОЗРОБКИ.....                                     | 4 |

|             |      |                 |        |      |                                                                                                                  |                                           |       |         |
|-------------|------|-----------------|--------|------|------------------------------------------------------------------------------------------------------------------|-------------------------------------------|-------|---------|
|             |      |                 |        |      | ІАЛЦ. 045430.002 ТЗ                                                                                              |                                           |       |         |
|             |      |                 |        |      |                                                                                                                  |                                           |       |         |
| Змін        | Арк. | № докум.        | Підпис | Дата |                                                                                                                  |                                           |       |         |
| Розробив    |      | Кукса К. В.     |        |      | Вбудована програмна система<br>дистанційного керування<br>безпілотними апаратами<br><br><b>Технічне завдання</b> | Літ.                                      | Аркуш | Аркушів |
| Перевірів   |      | Морозов К.В.    |        |      |                                                                                                                  |                                           | 1     | 4       |
|             |      |                 |        |      |                                                                                                                  | КПІ ім. Ігоря Сікорського,<br>ФПСІМ КВ-21 |       |         |
| Н. контроль |      | Клятченко Я.М.  |        |      |                                                                                                                  |                                           |       |         |
| Затвердив   |      | Романкевич В.О. |        |      |                                                                                                                  |                                           |       |         |

## 1. НАЙМЕНУВАННЯ ТА ГАЛУЗЬ РОЗРОБКИ

Назва розробки: «Програмна система віддаленого керування контролера безпілотних апаратів»

Галузь застосування: розробка та модифікація безпілотних апаратів.

## 2. ПІДСТАВА ДЛЯ РОЗРОБКИ

Підставою для розробки є завдання на дипломне проєктування на здобуття першого (бакалаврського) рівня вищої освіти, затверджене кафедрою системного програмування і спеціалізованих комп'ютерних систем Національного технічного університету України «Київський Політехнічний Інститут імені Ігоря Сікорського».

## 3. МЕТА І ПРИЗНАЧЕННЯ РОБОТИ

Метою даного проєкту є створення прошивки польотного контролера і додатку для керування.

## 4. ДЖЕРЕЛА РОЗРОБКИ

Джерелом інформації є технічна та науково-технічна література, технічна документація, публікації у періодичних виданнях та електронні статті у мережі Інтернет.

## 5. ТЕХНІЧНІ ВИМОГИ

### 5.1. Вимоги до програмного продукту, що розробляється

- Можливість підключитись до наземної станції і польотного контролера через Wi-Fi за допомогою додатку.
- Можливість редагувати налаштування польотного контролера через додаток;

|       |      |          |        |      |                    |      |
|-------|------|----------|--------|------|--------------------|------|
|       |      |          |        |      | ІАЛЦ.045430.002 ТЗ | Арк. |
|       |      |          |        |      |                    | 2    |
| Змін. | Арк. | № докум. | Підпис | Дата |                    |      |

- Можливість обирати тип підключеного мікроконтролера та кількість і тип підключеної периферії
- Можливість керувати польотним контролером за допомогою наземної станції;
- Можливість керувати польотним контролером за допомогою додатку, підключеного до наземної станції;
- Можливість польотного контролера працювати як в ручному так і в автономному режимі;
- Подальша підтримка і додавання нової периферії;
- Графічний інтерфейс з різною візуальною інформацією;

#### 5.2. Вимоги до програмного та апаратного забезпечення додатку

- Операційна система Windows 7 або вище
- Двохядерний процесор з частотою від 1.6 ГГц або вище
- Оперативна пам'ять: від 4 Гб;
- Наявність доступу до мережі Wi-Fi;
- Відеокарта з підтримкою DirectX 10 або вище

#### 5.3. Вимоги до апаратного забезпечення польотного контролера

- Мікроконтролер серії ESP32;

## 6. ЕТАПИ РОЗРОБКИ

| № з/п | Назва етапів виконання дипломного проєкту                | Термін виконання етапів |
|-------|----------------------------------------------------------|-------------------------|
| 1.    | Вивчення літератури за тематикою проєкту                 | 15.04.2026              |
| 2.    | Розроблення та узгодження технічного завдання            | 30.04.2026              |
| 3.    | Аналіз існуючих рішень                                   | 05.05.2026              |
| 4.    | Підготовка матеріалів першого розділу дипломного проєкту | 10.05.2026              |
| 5.    | Підготовка матеріалів другого розділу дипломного проєкту | 18.05.2026              |
| 6.    | Підготовка графічної частини дипломного проєкту          | 20.05.2026              |
| 7.    | Оформлення документації дипломного проєкту               | 25.05.2026              |
| 8.    | Попередній огляд матеріалів диплому на кафедрі           | 30.05.2026              |



[illegible]

**Пояснювальна записка**  
**до дипломного проєкту**

на тему: Вбудована програмна система дистанційного керування безпілотними  
апаратами

Київ – 2026

|                                                                         |    |
|-------------------------------------------------------------------------|----|
| ЗМІСТ                                                                   |    |
| ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ .....                    | 4  |
| ВСТУП .....                                                             | 7  |
| 1. АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ ТА АКТУАЛЬНІСТЬ ТЕМИ ДИПЛОМНОГО ПРОЄКТУ ..... | 9  |
| 1.1 Історія створення і розвитку безпілотних апаратів .....             | 9  |
| 1.2 Сучасне ПЗ для керування безпілотними апаратами .....               | 13 |
| 1.2.1 Відкриті системи .....                                            | 13 |
| 1.2.2 Закриті системи .....                                             | 16 |
| 1.3 Структура системи, що пропонується .....                            | 17 |
| 2 ПРОГРАМНЕ ТА АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ РОЗРОБКИ.....                  | 20 |
| 2.1 Мікроконтролерний модуль БА.....                                    | 20 |
| 2.2 Середовище програмування мікроконтролера.....                       | 21 |
| 2.2.1 Arduino IDE .....                                                 | 21 |
| 2.2.2 FreeRTOS.....                                                     | 23 |
| 2.2.2 Wi-Fi Stack .....                                                 | 23 |
| 2.3 Середовище створення додатку.....                                   | 24 |
| 2.3.1 QtFramework .....                                                 | 25 |
| 2.3.2 Qt Creator IDE .....                                              | 26 |
| 2.4 Протоколи комунікації .....                                         | 27 |
| 2.4.1 Wi-Fi .....                                                       | 27 |
| 2.4.2 TCP.....                                                          | 28 |
| 3 ОПИС РОЗРОБЛЕНОЇ СИСТЕМИ: ПРИНЦИПИ ТА КОМПОНЕНТИ .....                | 31 |
| 3.1 Архітектура системи.....                                            | 31 |
| 3.2 Безпілотний апарат .....                                            | 33 |
| 3.3 Наземна станція.....                                                | 37 |
| 3.4 Десктопний додаток.....                                             | 38 |

|                                               |    |
|-----------------------------------------------|----|
| 3.5 Протокол комунікації додатку .....        | 41 |
| 3.6 Модулі для відладки проєкту:.....         | 43 |
| ВИСНОВКИ.....                                 | 49 |
| СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ ТА ЛІТЕРАТУРИ..... | 51 |
| ДОДАТКИ.....                                  | 53 |

## ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

ADC (Analog-to-Digital Converter) — аналогово-цифровий перетворювач (АЦП), пристрій, що перетворює аналоговий сигнал у цифровий код для подальшої обробки мікроконтролером.

API (Application Programming Interface) — програмний інтерфейс додатка, набір готових класів, процедур, функцій, структур або констант, що надаються застосунком або операційною системою для використання у зовнішніх програмних продуктах.

ARM (Advanced RISC Machine) — архітектура 32-розрядних та 64-розрядних RISC-процесорів, яка широко використовується в енергоефективних вбудованих системах, смартфонах та мікроконтролерах.

AVR — родина 8-бітових мікроконтролерів (спочатку розроблена компанією Atmel, нині Microchip), що має RISC-архітектуру і є основою для багатьох популярних платформ, таких як Arduino.

CAN (Controller Area Network) — промислова мережева шина, розроблена для об'єднання різноманітних виконавчих пристроїв та датчиків в одну мережу (найчастіше використовується в автомобільній електроніці та автоматичності).

DAC (Digital-to-Analog Converter) — цифро-аналоговий перетворювач (ЦАП), пристрій для перетворення цифрового коду назад в аналоговий сигнал (наприклад, для генерації звуку чи керування напругою).

ESP32 — серія мікроконтролерів з підтримкою Wi-Fi та Bluetooth, розроблена компанією Espressif Systems, що використовується для створення вбудованих систем та проєктів Інтернету речей (IoT).

GMSK (Gaussian Minimum Shift Keying) — гауссова маніпуляція з мінімальним зсувом, тип цифрової модуляції з безперервною фазою, що використовується, наприклад, у стандартах зв'язку GSM для підвищення спектральної ефективності.

|       |      |          |        |      |                    |   |
|-------|------|----------|--------|------|--------------------|---|
|       |      |          |        |      | ІАЛЦ.045430.004 ПЗ |   |
| Змін. | Арк. | № докум. | Підпис | Дата |                    | 4 |

GPIO (General Purpose Input/Output) — універсальні входи/виходи мікроконтролера, які програмно налаштовуються для приймання або передавання цифрових сигналів.

GPS (Global Positioning System) — глобальна навігаційна супутникова система для визначення географічних координат, швидкості та точного часу.

GUI (Graphical User Interface) — графічний інтерфейс користувача, тип інтерфейсу, який дає змогу взаємодіяти з технікою за допомогою графічних елементів.

I2C (Inter-Integrated Circuit) — двопровідний послідовний інтерфейс для синхронного зв'язку між мікросхемами, використовує лінії SDA та SCL.

IMU (Inertial Measurement Unit) — інерціальний вимірювальний модуль, електронний пристрій (зазвичай складається з акселерометра та гіроскопа), який вимірює кутову швидкість і лінійне прискорення тіла.

LED (Light Emitting Diode) — світлодіод, напівпровідниковий прилад, що випромінює світло при проходженні через нього електричного струму.

MEMS (Micro-Electro-Mechanical Systems) — мікроелектромеханічні системи, мікроскопічні пристрої, що поєднують у собі механічні та електронні компоненти.

PID (Proportional-Integral-Derivative) — ПІД-регулятор (пропорційно-інтегрально-диференціальний), пристрій або алгоритм у контурі керування зі зворотним зв'язком, що підтримує задане значення параметра

PWM (Pulse Width Modulation) — широтно-імпульсна модуляція (ШІМ), метод управління аналоговими сигналами за допомогою зміни тривалості цифрових імпульсів.

RTOS (Real Time Operating System) — операційна система реального часу, яка оптимізована для забезпечення чітко визначеного часу відповіді на події в приладах з обмеженими процесорними ресурсами.

SMBus (System Management Bus) — двопровідний інтерфейс, розроблений на базі I2C, призначений для низькошвидкісного зв'язку

найчастіше для моніторингу стану батарей комп'ютерів, датчиків температури тощо.

SPI (Serial Peripheral Interface) — послідовний периферійний інтерфейс, високошвидкісний синхронний протокол для обміну даними між мікроконтролером та зовнішніми пристроями (пам'яттю, дисплеями, датчиками).

TCP (Transmission Control Protocol) — протокол керування передаванням, призначений для забезпечення надійної доставки даних у комп'ютерних мережах на транспортному рівні моделі OSI.

UART (Universal Asynchronous Receiver/Transmitter) — універсальний асинхронний приймач-передавач, популярний інтерфейс послідовної передачі даних без використання загальної лінії тактування, використовує лінії RX та TX.

USB (Universal Serial Bus) — універсальна послідовна шина, стандарт інтерфейсу для підключення периферійних пристроїв до комп'ютера або обміну даними між пристроями.

VTOL (Vertical Take-Off and Landing) — літальний апарат з вертикальним злетом і посадкою, тип повітряного судна, який здатний злітати, зависати та сідати вертикально, як гелікоптер, але летіти за рахунок крил, як літак.

|       |      |          |        |      |                    |   |
|-------|------|----------|--------|------|--------------------|---|
|       |      |          |        |      | ІАЛЦ.045430.004 ПЗ |   |
| Змін. | Арк. | № докум. | Підпис | Дата |                    | 6 |

## ВСТУП

Сучасний розвиток безпілотних апаратів є одним із найважливіших напрямків розвитку робототехнічних та інформаційно-комунікаційних систем. Безпілотні платформи активно застосовуються у військовій сфері, сільському господарстві, картографуванні місцевості, моніторингу інфраструктури, логістиці, пошуково-рятувальних операціях та багатьох інших галузях. Широке поширення таких систем стало можливим завдяки розвитку мікроелектроніки, бездротових технологій передачі даних та сучасних алгоритмів керування.

Одним із ключових елементів будь-якого безпілотного апарата є польотний контролер, який забезпечує збір інформації із сенсорів, оцінювання просторового положення апарата, виконання алгоритмів стабілізації та формування керуючих сигналів для силової установки. Від ефективності роботи польотного контролера безпосередньо залежать стійкість польоту, точність керування та загальна надійність системи.

На сьогоднішній день існує велика кількість готових рішень, таких як ArduPilot, PX4 та Betaflight, однак їхня архітектура часто є надлишковою для навчальних або дослідницьких проєктів. Розробка програми власного польотного контролера дозволяє детально вивчити принципи функціонування безпілотних систем, особливості роботи сенсорів, алгоритми стабілізації та організацію взаємодії між програмними й апаратними компонентами.

У даному дипломному проєкті розглядається розробка програмно-апаратного комплексу керування квадрокоптером або наземним транспортом. Як центральний обчислювальний вузол використовується мікроконтролер ESP32, який поєднує достатню обчислювальну продуктивність, розвинений набір периферійних інтерфейсів та вбудовані засоби бездротового зв'язку.

Метою дипломного проєкту є розробка програмно-апаратної системи керування квадрокоптером на базі мікроконтролера ESP32 із підтримкою бездротового зв'язку, збору телеметричної інформації та автоматичної стабілізації польоту.

|       |      |          |        |      |                    |   |
|-------|------|----------|--------|------|--------------------|---|
|       |      |          |        |      | ІАЛЦ.045430.004 ПЗ | 7 |
| Змін. | Арк. | № докум. | Підпис | Дата |                    |   |

Предметом дослідження є програмні та апаратні засоби реалізації польотного контролера на базі мікроконтролера ESP32, а також алгоритми оцінювання орієнтації та стабілізації польоту.

Практична цінність роботи полягає у створенні працездатного прототипу польотного контролера та наземної станції керування, які можуть бути використані як навчальна або дослідницька платформа для подальшого розвитку безпілотних систем.

|       |      |          |        |      |                     |   |
|-------|------|----------|--------|------|---------------------|---|
|       |      |          |        |      | ІАЛЦ. 045430.004 ПЗ |   |
|       |      |          |        |      |                     |   |
| Змін. | Арк. | № докум. | Підпис | Дата |                     | 8 |

# 1. АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ ТА АКТУАЛЬНІСТЬ ТЕМИ ДИПЛОМНОГО ПРОЄКТУ

## 1.1 Історія створення і розвитку безпілотних апаратів

Ідея створення безпілотних апаратів з'явилась задовго до розповсюдження електроніки, Першим прототипом безпілотного апарату вважається «Aerial Target». винахід британця Арчібальда Лоуда в 1916 році – це був радіокерований літак. Цей проєкт не увінчався успіхом.

Першим масовим проєктом вважається БПЛА «QQ-2» (рис. 1) американської компанії «Radioplane Corp». Перший політ було проведено в 1939 році. Основним експлуатантом були ВПС США, які використовували безпілотники для тренування зенітної артилерії та авіації. Його основними перевагами окрім безпеки пілотів була низька ціна і невеликі розміри.



Рисунок 1 – БПЛА «QQ2»

Після другої світової війни швидко зріз попит використання БПЛА в військових потребах, особливо армією США. За цей час окрім технічних характеристик покращувалась також функціональна частина: тепер безпілотні апарати використовувались для фото, радіорозвідки і нанесення авіаударів

Як приклад, можна навести модель AQM-34 (рис. 2)



Рисунок 2 – БПЛА «AQM-34»

У 1990-2000-х роках почалося активне використання безпілотних апаратів в цивільному житті: вони почали використовуватись у кінозйомці, аграрному секторі, логістиці, рятувальних операціях, науці й моніторингу навколишнього середовища.

Окрім доступності електронних компонентів, БПЛА отримали таке поширення завдяки створенню квадрокоптера, який має можливість зависати на місці і при цьому зберігає неймовірну маневреність. Ідея створення першого квадрокоптера належить французькому інженеру Етьєну Омнігену. Його пілотований апарат Oehmichen 2 (рис. 3) здійснив свій найдовший політ довжиною у 360 метрів в 1924 році

|       |      |          |        |      |
|-------|------|----------|--------|------|
|       |      |          |        |      |
| Змін. | Арк. | № докум. | Підпис | Дата |

ІАЛЦ.045430.004 ПЗ

Арк.

10

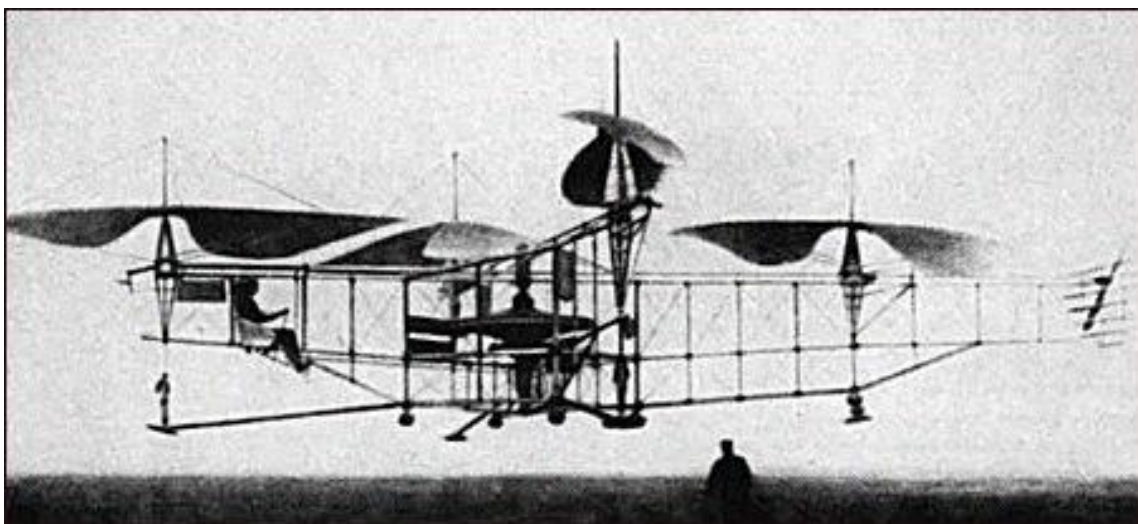


Рисунок 3 – Перший пілотований квадрокоптер

В сучасності квадрокоптери – це найпопулярніший тип безпілотних апаратів, їх випускають такі сучасні компанії як DJI (рис. 4), Autel Robotics, Potensic, Ryze Tech



Рисунок 4 – DJI Mavic 3 в польоті

Також необхідно окремо виділити Безпілотні наземні транспортні засоби (БНТЗ) або наземні роботизовані комплекси (НРК). Для переміщення вони використовують колеса, гусениці або роботизовані кінцівки. На відміну від безпілотних літальних апаратів, НРК можуть транспортувати значно більші корисні навантаження, працювати тривалий час без підзарядки та виконувати завдання у складних погодних умовах.

Перші наземні роботизовані системи розроблялися переважно для військових потреб, зокрема для дистанційного розмінування, розвідки та роботи

|       |      |          |        |      |
|-------|------|----------|--------|------|
|       |      |          |        |      |
| Змін. | Арк. | № докум. | Підпис | Дата |

ІАЛЦ.045430.004 ПЗ

Арк.

11

в небезпечних для людини умовах. З розвитком комп'ютерного зору, навігаційних систем та алгоритмів автономного керування сфера їх застосування значно розширилась.

Сьогодні наземні роботизовані комплекси використовуються у військовій справі, промисловості, логістиці, сільському господарстві та рятувальних операціях. Вони можуть виконувати патрулювання території, транспортування вантажів, евакуацію поранених, доставку боєприпасів, моніторинг технічного стану об'єктів та інші завдання. Особливо актуальними НРК стали під час сучасних військових конфліктів, де вони дозволяють зменшити ризик для особового складу та підвищити ефективність виконання бойових завдань.

Сучасні роботизовані комплекси часто оснащуються відеокамерами, тепловізорами, лазерними далекомірами, навігаційними системами, супутниковими приймачами та засобами бездротового зв'язку.

Одним з прикладів сучасних НРК можна назвати Комплекс роботизованого очищення (розмінування) «Змій» (рис. 5) - український дистанційно керований засіб, створений компанією Rovertech.



Рисунок 5 – КНРО «Змій»

Таким чином, розвиток безпілотних літальних апаратів та наземних роботизованих комплексів демонструє загальну тенденцію до автоматизації транспортних і спеціалізованих систем. Постійне вдосконалення електроніки, сенсорів та програмного забезпечення відкриває нові можливості.

## 1.2 Сучасне ПЗ для керування безпілотними апаратами

### 1.2.1 Відкриті системи

Відкритими системами називатимемо системи віддаленого або автономного контролю безпілотних апаратів, які не вимагають купівлю апаратної і програмної частини лише в одного виробника. Такі системи не користуються закритими протоколами передачі, можуть працювати з аналогами в інших виробників а доволі часто навіть мають відкритий для загального користування вихідний код

#### Ardupilot

Головним прикладом такої системи можна назвати Ardupilot (рис. 6)

Це набір програмного забезпечення, що складається з навігаційного програмного забезпечення, або на професійному сленгу прошивки і програмного забезпечення для керування наземною станцією Mission Planner.

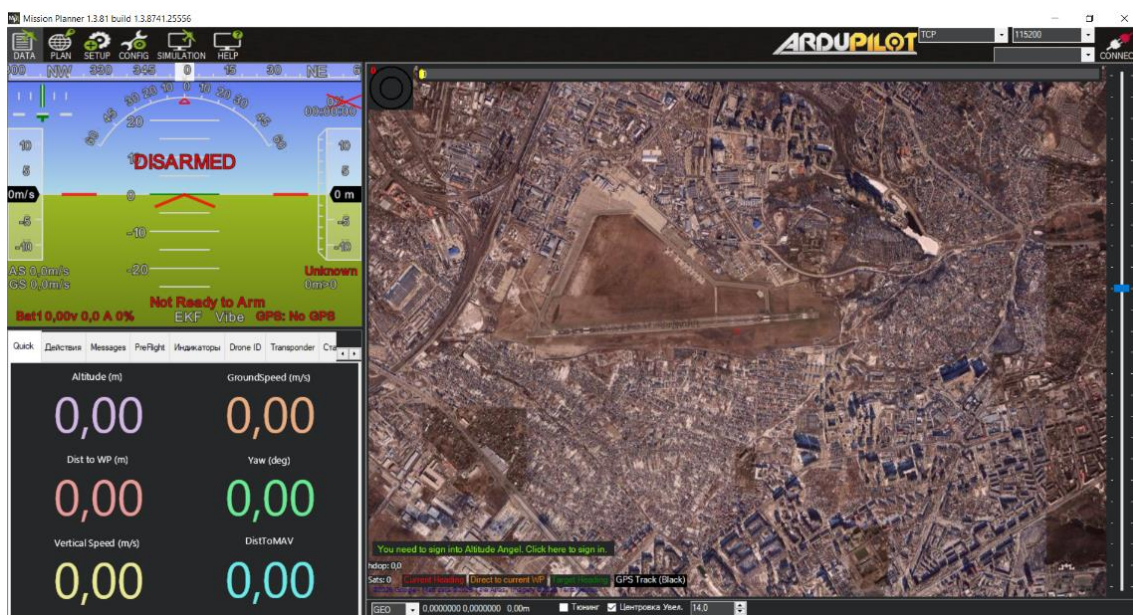


Рисунок 6 – Головна панель перегляду телеметрії в Ardupilot

ArduPilot є системою керування безпілотними апаратами з відкритим програмним кодом, яка розробляється спільнотою ArduPilot Development Team з 2009 року. Основна частина програмного забезпечення написана мовою програмування C++, що забезпечує високу продуктивність та можливість роботи в режимі реального часу. Завдяки відкритому коду система активно розвивається

спільнотою розробників і підтримується великою кількістю виробників апаратного забезпечення.

Однією з головних переваг ArduPilot є підтримка різних режимів роботи. Система дозволяє здійснювати ручне керування, напіваавтоматичне керування із використанням допоміжних функцій стабілізації, а також повністю автономне виконання польотного завдання. Перед початком місії оператор може сформувати маршрут із набору контрольних точок, після чого безпілотний апарат буде самостійно виконувати політ відповідно до заданого сценарію. У разі втрати зв'язку система здатна автоматично повернутися до точки старту або продовжити виконання місії.

Важливою особливістю ArduPilot є універсальність. Система підтримує безпілотні літальні апарати літакового типу, мультикоптери, конвертоплани (VTOL), вертольоти, наземні роботизовані комплекси, надводні та підводні апарати. Завдяки цьому розробникам не потрібно створювати окреме програмне забезпечення для кожного типу транспортного засобу.

Для взаємодії з периферійними пристроями та датчиками підтримується велика кількість апаратних інтерфейсів, серед яких SPI, I2C, UART, CAN та SMBus. Система може працювати з інерціальними вимірювальними модулями (IMU), GPS-приймачами, магнітометрами, барометрами, далекомірами, лідарами, оптичними сенсорами та іншими пристроями. Для керування виконавчими механізмами підтримуються різні типи електродвигунів, сервоприводів та електронних регуляторів швидкості.

Для обміну даними між автопілотом та наземною станцією використовується протокол MAVLink, який забезпечує передачу телеметрії, параметрів системи, команд керування та службової інформації. Завдяки цьому оператор може в режимі реального часу контролювати положення апарата, стан датчиків, заряд акумуляторів, якість GPS-сигналу та інші параметри.

iNav та Betaflight — це сучасні open-source прошивки для польотних контролерів безпілотних літальних апаратів. Як і ArduPilot, вони складаються з програмного забезпечення для польотного контролера та окремого

|       |      |          |        |      |                    |      |
|-------|------|----------|--------|------|--------------------|------|
|       |      |          |        |      | ІАЛЦ.045430.004 ПЗ | Арк. |
|       |      |          |        |      |                    | 14   |
| Змін. | Арк. | № докум. | Підпис | Дата |                    |      |

конфігуратора для налаштування параметрів системи через персональний комп'ютер.

Обидві системи підтримують різні типи літальних апаратів, роботу з датчиками, налаштування PID-регуляторів, режими стабілізації та телеметрію. Конфігурація виконується через графічний інтерфейс, що значно спрощує процес налаштування порівняно з більш складними платформами.

Головною відмінністю від ArduPilot є орієнтація на простоту використання та ручне керування. Розробники приділяють значну увагу зручності налаштування, швидкому запуску та мінімізації кількості параметрів, необхідних для початку роботи.

Betaflight найчастіше використовується в гоночних FPV-дронах та дронах для фристайлу. Прошивка оптимізована для роботи на високих частотах циклів керування та забезпечує дуже швидку реакцію на дії пілота. Значна увага приділяється алгоритмам фільтрації шумів від двигунів та високоточному налаштуванню PID-регуляторів. Завдяки цьому Betaflight забезпечує високу маневреність та точність керування навіть під час виконання складних акробатичних маневрів.

INAV можна розглядати як проміжне рішення між ArduPilot та Betaflight. На відміну від Betaflight, він підтримує GPS-навігацію, автоматичне повернення до точки старту, утримання позиції та виконання простих автономних місій. Водночас система залишається значно простішою у налаштуванні порівняно з ArduPilot. Саме тому INAV часто використовується для FPV-літаків, крил та мультикоптерів, де необхідне поєднання ручного керування та базових функцій автономної навігації.

Apache NuttX RTOS - це безкоштовна відкрита POSIX-сумісна операційна система реального часу. На високому рівні використовується польотна програма PX4 Autipilot.

За рівнем можливостей дану прошивку можна порівняти з Ardupilot. Обидві системи підтримують широкий спектр типів безпілотних апаратів, протокол MAVLink, автономні місії та інтеграцію з сучасними навігаційними

|       |      |          |        |      |                    |      |
|-------|------|----------|--------|------|--------------------|------|
|       |      |          |        |      | ІАЛЦ.045430.004 ПЗ | Арк. |
|       |      |          |        |      |                    | 15   |
| Змін. | Арк. | № докум. | Підпис | Дата |                    |      |

засобами. Головні переваги PX4 – це більш сучасний інтерфейс QGroundControl, менша кількість багів в прошивці завдяки явному розділу роботи ОС і польотного модуля, сучасні підходи до розробки програмного забезпечення, використання операційних систем реального часу та модульна архітектура, а також постійна підтримка від розробників. Наразі ця система програє в масштабах використання Ardupilot, але з кожним роком набирає все більше популярності.

### 1.2.2 Закриті системи

Закритими системами називатимемо системи віддаленого або автономного контролю безпілотних апаратів, які працюють лише у екосистемі виробника або надають мінімальний доступ до власних алгоритмів і протоколів. Такі системи зазвичай дорожчі і менш гнучкі, але знімають з користувача необхідність попереднього налаштування, а також надають гарантію роботи

Dajiang Innovation Technology (DJI) – китайська приватна компанія, яка випускає власні квадрокоптери, мікроконтролери, відеообладнання.

Компанія має велику кількість ліній дронів, наприклад Mavic, Air Mini, Inspire. Одним із найважливіших досягнень компанії стала розробка технології передачі даних та відеосигналу OcuSync, яка забезпечує передачу цифрового відео високої якості на значні відстані з мінімальною затримкою. Також сучасні апарати DJI оснащуються системами уникнення перешкод, побудованими на основі камер, ультразвукових датчиків та алгоритмів комп'ютерного зору.

Skydio – американська компанія, що спеціалізується на розробці автономних безпілотних систем із використанням комп'ютерного зору і штучного інтелекту.

Завдяки побудові 3D-карти середовища в реальному часі, дрони компанії мають можливість автоматично будувати маршрут, розпізнавати перешкоди і слідкувати за необхідним об'єктом.

Ukrspecsystems – українське оборонне підприємство, що спеціалізується на розробці безпілотних авіаційних комплексів, систем зв'язку та спеціалізованої військової електроніки. Їх головним продуктом є розвідувальний дрон Shark

|       |      |          |        |      |                    |      |
|-------|------|----------|--------|------|--------------------|------|
|       |      |          |        |      | ІАЛЦ.045430.004 ПЗ | Арк. |
|       |      |          |        |      |                    | 16   |
| Змін. | Арк. | № докум. | Підпис | Дата |                    |      |

(рис. 7) з захистом каналів зв'язку, використанням криптографічних засобів та стійкості до засобів радіоелектронної боротьби



Рисунок 7 – Дрон Shark компанії Ukrspecsystems

### 1.3 Структура системи, що пропонується

Розглянутий у відкритих системах функціонал дозволяє легко і точно налаштувати безпілотний апарат під свої умови, але лише у випадку здатності фізично підключати контакти мікроконтролера до периферії.

Існує декілька рішень як вирішити проблему підключення:

1. Запаяти контролер з необхідними датчиками і периферією на власну розведену плату, що може бути рішенням для великих компаній-виробників польотних контролерів, або людей з серйозними навичками у hardware-частині, але точно не для любителів безпілотних апаратів чи програмістів-початківців

2. Купити готовий польотний контролер у провідних компаній, який вже матиме перевірену прошивку, але обмежує користувача у своїх діях, бо фізичні польотні контролери розраховані лише на один окремий тип безпілотного апарату, також зазвичай вже мають такі вбудовані сенсори як гіроскоп і акселерометр, не підлягають заміні одного типу двигунів на інші. Такий варіант є надлишковим і не дуже бюджетним для початківців

Обидва рішення можуть бути невідповідними для людей-початківців, які бажають створювати власні безпілотні апарати, або для невеликих підприємств,

|       |      |          |        |      |
|-------|------|----------|--------|------|
|       |      |          |        |      |
| Змін. | Арк. | № докум. | Підпис | Дата |

ІАЛЦ.045430.004 ПЗ

Арк.

17

які бажають створювати дешеві безпілотні апарати з подальшою можливістю їх підтримки і модифікації, а також для міні-дронів.

Запропоноване в дипломному проєкті рішення буде полягати у створенні системи керування безпілотним апаратом на основі базових мікроконтролерів, які використовуються у відладочних платах, або платах розробників (Arduino, STM32, ESP32). З прошивкою, яка матиме можливості налаштування як типу безпілотного апарату, так і типу будь-якої периферії, що під'єднується на інтерфейс GPIO з власним конфігуратором і наземною станцією

Таблиця 1 – порівняння різних систем керування безпілотними апаратами

|                                       | Закриті системи         | Відкриті системи       | Пропонована система |
|---------------------------------------|-------------------------|------------------------|---------------------|
| Приблизна ціна контролера             | Неможливо купити окремо | >50\$                  | <15\$               |
| Мінімальна ціна готової системи       | 500\$                   | 150\$                  | 30\$                |
| Можливість програмної модифікації     | Ні                      | Так                    | Так                 |
| Можливість апаратної модифікації      | Ні                      | Лише окремі компоненти | Так                 |
| Можливість зміни типу апарату         | Ні                      | Ні                     | Так                 |
| Вимоги до загальних знань з роботи БА | Низькі                  | Високі                 | Високі              |

У першому розділі було проведено аналіз історії розвитку безпілотних систем, сучасного стану галузі та існуючих програмно-апаратних рішень для

керування безпілотними апаратами. Розглянуто основні етапи еволюції безпілотних літальних апаратів від перших радіокерованих моделей до сучасних високотехнологічних комплексів, які активно застосовуються у військовій сфері, промисловості, сільському господарстві, логістиці та моніторингових системах.

Під час аналізу сучасних рішень було досліджено можливості поширених платформ керування безпілотними апаратами, зокрема ArduPilot, Betaflight та iNav. Встановлено, що зазначені системи забезпечують широкий функціонал, підтримують значну кількість типів літальних апаратів, датчиків та режимів польоту, а також мають розвинені засоби налаштування і телеметрії. Разом з тим їх використання часто потребує спеціалізованих польотних контролерів, глибоких знань у сфері налаштування систем або застосування попередньо визначених апаратних конфігурацій.

Окремо було розглянуто діяльність сучасних виробників безпілотних систем та роботизованих комплексів, серед яких DJI, Skydio та українська компанія Ukrspecsystems. Аналіз показав, що сучасні комерційні рішення характеризуються високою складністю, значною вартістю та орієнтацією на конкретні сценарії використання. При цьому можливості їх модифікації та адаптації для навчальних або експериментальних задач часто є обмеженими.

У результаті проведеного дослідження було виявлено актуальну проблему створення доступної, гнучкої та масштабованої системи керування безпілотними апаратами, яка дозволяла б використовувати поширені мікроконтролерні платформи та довільний набір периферійних модулів без необхідності розробки власної друкованої плати або придбання дорогих спеціалізованих контролерів.

На основі проведеного аналізу сформульовано концепцію системи, що розробляється в межах дипломного проєкту. Запропоноване рішення базується на використанні мікроконтролерних платформ загального призначення, підтримує модульне підключення периферійних пристроїв, бездротовий обмін даними та можливість гнучкого налаштування через спеціалізований програмний застосунок. Такий підхід дозволяє поєднати низьку вартість

|       |      |          |        |      |                    |      |
|-------|------|----------|--------|------|--------------------|------|
|       |      |          |        |      | ІАЛЦ.045430.004 ПЗ | Арк. |
|       |      |          |        |      |                    | 19   |
| Змін. | Арк. | № докум. | Підпис | Дата |                    |      |

реалізації, простоту модифікації та достатній функціонал для побудови експериментальних і навчальних безпілотних систем.

Таким чином, результати першого розділу підтвердили актуальність теми дипломного проєкту та обґрунтували доцільність розробки власної програмно-апаратної системи керування безпілотним апаратом на базі сучасних мікроконтролерів і відкритих технологій.

## **2 ПРОГРАМНЕ ТА АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ РОЗРОБКИ**

### **2.1 Мікроконтролерний модуль БА**

Для розробки програмного забезпечення для безпілотного апарату було обрано модуль ESP-32-WROOM (рис. 8) через її високі обчислювальні потужності і низьку ціну, а також вбудований WI-FI-модуль.



Рисунок 8 – ESP32-WROOM-32

Принцип роботи мікроконтролера – керування GPIO - 38 пінів, де кожен пін може або приймати або надсилати логічний сигнал. Для нашої плати – це 3.3В.

Окрім стандартного режиму цифрової логіки, деякі окремі піни мають додаткові функції, це може бути ADC, DAC, SPI, UART, I2C, PWM, а також інші режими для використання.

За можливість використання окремих протоколів відповідають конкретні піни. В загальній і обраній моделі мікроконтролера серії ESP32 використовуються такі можливості:

- 18 пінів з можливістю аналогово-цифрового перетворювача
- 3 SPI інтерфейси
- 3 UART інтерфейси
- 2 I2C інтерфейси
- 16 пінів виводу широко-імпульсної модуляції
- 2 піни з можливістю цифро-аналогового перетворювача

Цього цілком достатньо для можливості підключення різноманітних сенсорів і модулів для керування.

Окрім цього, мікроконтролер має двоядерний процесор Xtensa LX6, що дозволяє одночасно виконувати декілька задач у режимі реального часу, оперативну пам'ять на 520 кілобайт і flash-пам'ять для збереження інформації від 4мб.

Також мікроконтролер має на борту Wi-Fi-модуль з підтримкою 802.11 2.4 GHz, що дозволяє йому підключатись до будь-якої Wi-Fi мережі і обмінюватись даними з іншими підключеними до мережі пристроями.

Для спрощення розробки використовується налагоджувальна плата ESP32 DevKit, на якій додатково розміщено стабілізатор напруги 3,3 В, мікросхему перетворювача USB-UART для програмування та обміну даними з персональним комп'ютером

## **2.2 Середовище програмування мікроконтролера**

Для роботи з мікроконтролером дуже важливо розуміти весь процес від написання коду до його прошивки. Від обраних IDE, програматора і мови програмування залежить дуже багато.

### **2.2.1 Arduino IDE**

|       |      |          |        |      |                     |      |
|-------|------|----------|--------|------|---------------------|------|
|       |      |          |        |      | ІАЛЦ. 045430.004 ПЗ | Арк. |
|       |      |          |        |      |                     | 21   |
| Змін. | Арк. | № докум. | Підпис | Дата |                     |      |

Arduino IDE – безкоштовне офіційне середовище розробки проєктів для плат Arduino, а також інших мікроконтролерів. Розроблена компанією Arduino Software у 2005 році.

Arduino IDE 2.x (зокрема версія 2.2.1) (рис. 9) є сучасною реалізацією середовища, побудованою на базі Visual Studio Code та Electron. Вона має покращений інтерфейс, вбудований менеджер бібліотек і плат, систему автодоповнення коду, налагодження та інтеграцію з серійним монітором. Це значно спрощує розробку та тестування програм для мікроконтролерів у порівнянні з попередніми версіями.

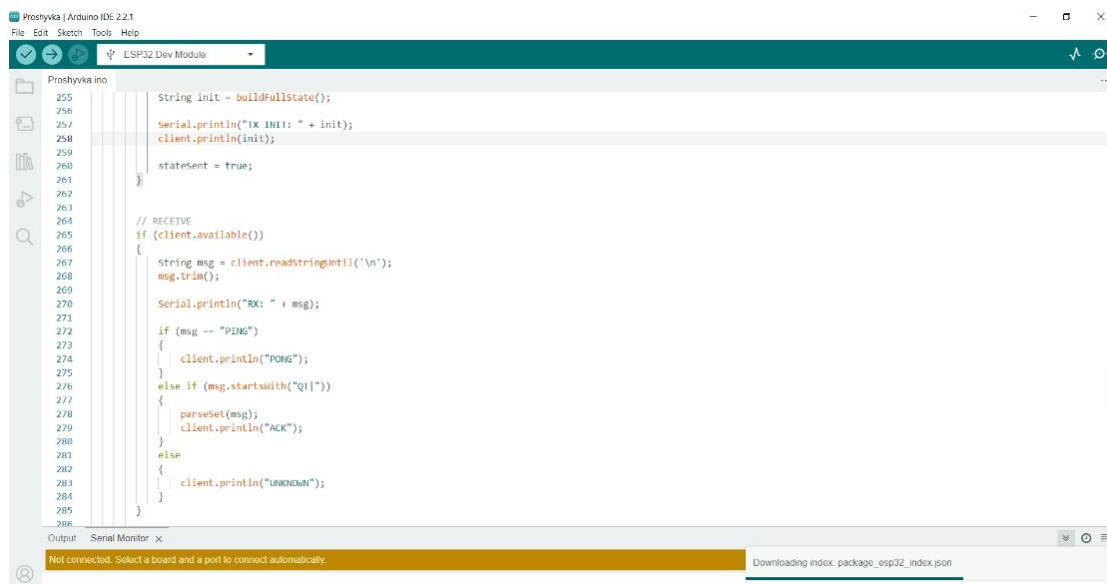


Рисунок 9 – Середовище розробки Arduino IDE 2.2.1

На самому початку IDE підтримувало тільки плати зі стандартного набору Arduino. Але завдяки відкритому вихідному коду з'явилась можливість додавати сторонні ядра, тому завдяки компілятору xtensa-esp32-elf-gcc та різним бібліотекам стало можливим використовувати серелрвище Arduino для написання програмного коду для ESP32.

Arduino IDE має велику кількість переваг у порівнянні з іншими IDE, а саме:

- Сумісність з стандартними функціями Arduino
- Можливість дебагу за допомогою Serial Monitor і Serial Plotter

|       |      |          |        |      |
|-------|------|----------|--------|------|
|       |      |          |        |      |
| Змін. | Арк. | № докум. | Підпис | Дата |

ІАЛЦ.045430.004 ПЗ

Арк.

22

- Зручний текстовий редактор, що допомагає знаходити помилки на етапі написання коду

- Велика кількість бібліотек, доступних для завантаження
- Простий інтерфейс компіляції і завантаження прошивки
- Вбудовані в прошивку можливості ESP32 за допомогою <Arduino.h>

### 2.2.2 FreeRTOS

При роботі зі складними проектами, як, наприклад, безпілотні апарати, дуже важливо жорстко керувати окремі задачі з точністю до кванту, оскільки невиконання певної задачі може порушити всю систему.

FreeRTOS – операційна система реального часу, розроблена Real Time Engineers Ltd. Вона є еталоном для сучасних операційних систем реального часу. Вона використовує м'ютекси, семафори і таймери для керування задачами, а також виділенням пам'яті.

FreeRTOS написана мовою C з невеликою кількістю коду, написаного на асемблері. Підтримується платформами ARM, Atmel, AVR та іншими.

Основною одиницею для роботи у FreeRTOS вважається задача – програма, яка містить частину ініціалізації та частину бескінечного виконання, або тіло. Задача не створюється і не видаляється сама по собі – цим займається програміст за допомогою API. При створенні задача може отримувати різні параметри а також пріоритет. Менеджментом часу для задач займається планувальник.

FreeRTOS має велику кількість переваг: готову, відладжену базу коду, безкоштовність, портативність, мінімальний розмір ядра.

Важливо сказати, що у нашому випадку використовується IDF FreeRTOS. Оскільки ми маємо на нашому мікроконтролері 2 ядра. То важливо займатись менеджментом між задачами на двох ядрах одночасно. Для цього використовується архітектура SMP.

### 2.2.2 Wi-Fi Stack

Разом з встановленим пакетом плат ESP32 в Arduino IDE також йде вбудована бібліотека WiFi.h, яка реалізує базовий стек бездротового зв'язку

|       |      |          |        |      |                    |      |
|-------|------|----------|--------|------|--------------------|------|
|       |      |          |        |      | ІАЛЦ.045430.004 ПЗ | Арк. |
|       |      |          |        |      |                    | 23   |
| Змін. | Арк. | № докум. | Підпис | Дата |                    |      |

стандарту IEEE 802.11. Ця бібліотека є високорівневою обгорткою над функціоналом Wi-Fi драйвера ESP32 і дозволяє швидко реалізувати мережеву взаємодію без необхідності роботи з низькорівневими API. Завдяки цьому девайс може працювати як сканер ефіру, точка доступу, або підключений до Wi-Fi мережі пристрій.

З технічної точки зору Wi-Fi стек ESP32 підтримує повноцінну реалізацію MAC та частково PHY-рівня, що дозволяє працювати з такими режимами як STA, AP та ESP-NOW (пропрієтарний протокол Espressif для прямого обміну пакетами між пристроями без повноцінного TCP/IP стеку). Це особливо важливо для систем реального часу, де необхідна мінімальна затримка передачі даних.

Одним із найпоширеніших способів використання Wi-Fi стеку в embedded-системах є реалізація TCP або UDP серверів для обміну даними між мікроконтролером та зовнішніми пристроями. У середовищі Arduino це реалізується за допомогою класу WiFiServer, який ініціалізується вказанням порту: WiFiServer server(N), де server – об'єкт сервера, а N – номер TCP-порту, на якому здійснюється прослуховування вхідних з'єднань.

Після запуску сервера (метод server.begin()) ESP32 починає очікувати підключення клієнтів у локальній мережі. Будь-який пристрій, що знаходиться в тій самій мережі, може встановити TCP-з'єднання з мікроконтролером та обмінюватися даними у вигляді текстових або бінарних пакетів.

## 2.3 Середовище створення додатку

При роботі з мікроконтролерами дуже важливо мати зручний додаток для їх налаштування і керування. Найкращим варіантом буде обрати саме десктопний, а не веб чи мобільний додаток, оскільки при роботі від нього вимагається:

- Витримувати навантаження середньої складності
- Бути зручним для налаштувань і аналізу польоту
- Мати можливість поширюватись на різні операційні системи
- Спілкуватись по TCP-порту з мікроконтролером
- Мати велику кількість візуальної інформації

|       |      |          |        |      |                     |      |
|-------|------|----------|--------|------|---------------------|------|
|       |      |          |        |      | ІАЛЦ. 045430.004 ПЗ | Арк. |
|       |      |          |        |      |                     | 24   |
| Змін. | Арк. | № докум. | Підпис | Дата |                     |      |

Серед основних інструментів для розробки десктопних додатків вважаю доречним QT (кьют).

### 2.3.1 QtFramework

Qt – це кросплатформний фреймворк для створення додатків, що надає можливості створення додатків майже в будь-якому напрямку. Розробляється двома компаніями: The Qt Company і Qt Project.

Фреймворк має як комерційну так і некомерційну ліцензії. Основна мова програмування для розробки – C++, але також є можливості розробки іншими мовами програмування, наприклад Python.

Qt надає такі можливості як: розробка графічних інтерфейсів, робота з мережею, мультимедіа, базами даних, 2D і 3D графікою, веб, тестуванням, навіть навігацією і різними інструментами вводу.

Кросплатформеність проєкту дозволяє створювати версії під різні операційні системи не змінюючи коду продукту - це вирішується на етапі компіляції. Окрім десктопних і мобільних пристроїв, є можливість портувати додатки на Embedded системи, такі як Linux і QNX.

Фреймворк ділиться на 3 модулі (рис. 10)

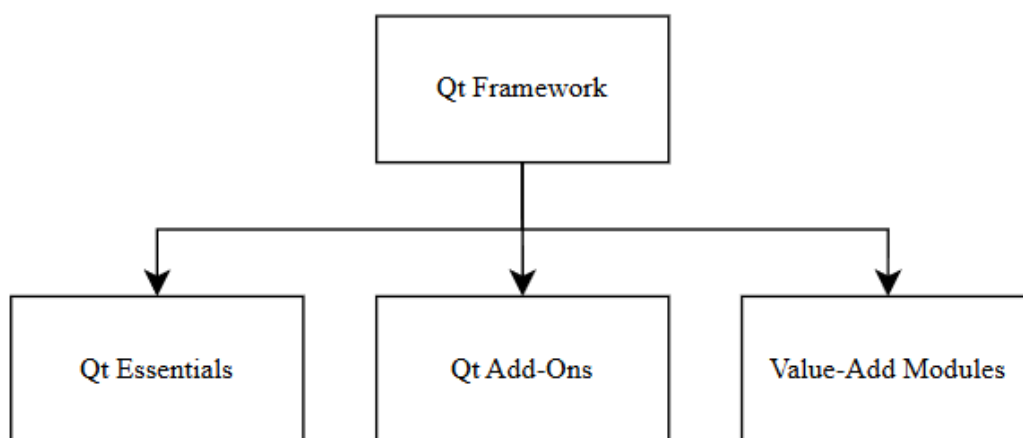


Рисунок 10 – Схема модулів фреймворку Qt

Qt Essentials – базові модулі, що доступні на всіх платформах і підтримують сумісність. Це, наприклад, Qt Core, Qt GUI, Qt Multimedia, Qt Network, Qt Widgets.

Qt Add-Ons – модулі, розроблені специфічно для певної платформи. Наприклад, multitouch для мобільних пристроїв.

Value-Add Modules – модулі, для роботи з вбудованими системами, які доступні лише з комерційною ліцензією.

Qt використовується в таких популярних проєктах, як Google Earth, Wolfram Mathematica, Telegram Desktop та інших.

Також однією з особливостей фреймворку є власна система – присутні власні аналоги базових класів бібліотеки STL, такі як QVector, QString та інші. Їхнє використання гарантує зменшену кількість низькорівневих проблем.

### 2.3.2 Qt Creator IDE

Qt Creator – це крос-платформне середовище розробки саме для фреймворку Qt. Тут окрім редактору коду присутній також дизайнер графічного інтерфейсу, система збірки і засоби налагодження.

Збірка графічного інтерфейсу здійснюється методом перетягування віджетів на поле вікна, хоча існує можливість редагувати інтерфейс кодом. (рис.

11)

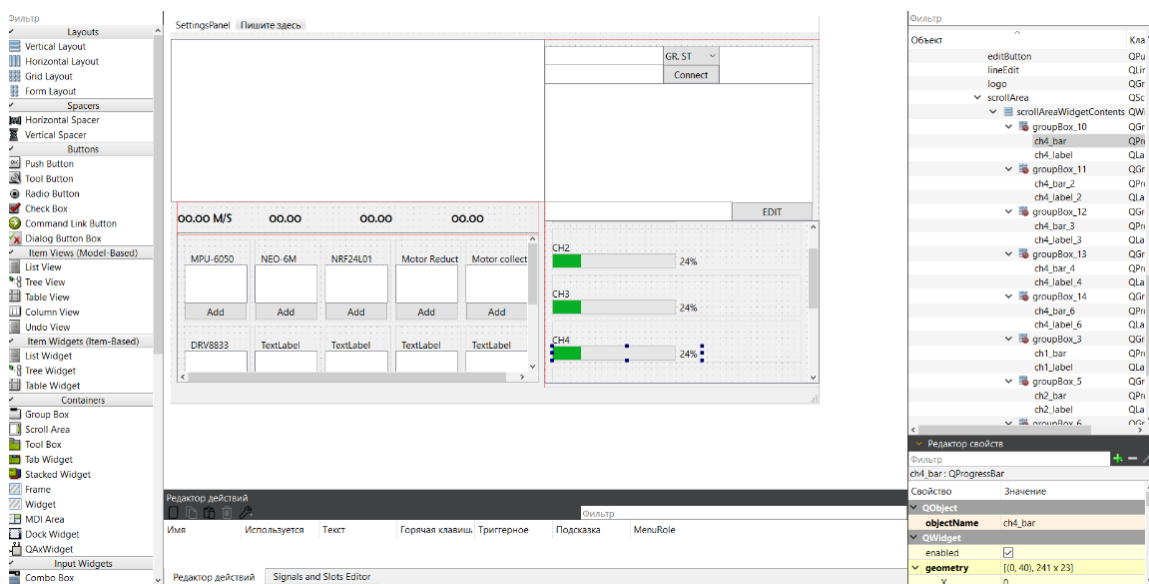


Рисунок 11 – Редактор графічного інтерфейсу в Qt Creator

Qt має власний збірник qmake, структура проєкту виглядає так:

- Project.pro – головний конфігураційний файл
- mainwindow.ui – графічний інтерфейс головного екрану

- Заголовочні файли
- Вихідні файли
- resources.qrc – механізм Qt для вбудованих файлів безпосередньо у виконуваний файл програми. Пишеться мовлю XML.

- Інші файли (медіа).

## 2.4 Протоколи комунікації

### 2.4.1 Wi-Fi

Wireless Friendly – це сімейство протоколів бездротової передачі даних, що засноване на стандартах IEEE 802.11 та забезпечує організацію локальних бездротових мереж (WLAN). Технологія Wi-Fi дозволяє здійснювати передачу даних між пристроями на основі радіоканалів у діапазонах 2,4 ГГц, 5 ГГц, а у сучасних версіях також 6 ГГц.

Перші стандарти сімейства IEEE 802.11 були розроблені у 1997 році, однак їх практичне впровадження почалося наприкінці 1990-х років. У 1999 році була створена організація Wi-Fi Alliance, яка займалась сертифікацією сумісності пристроїв та сприяла формуванню єдиного комерційного бренду "Wi-Fi", що значно прискорило масове поширення технології. Основні версії технологічних стандартів 802.11 включають в себе такі стандарти:

Таблиця 2 – стандарти сімейства IEEE 802.11

| Назва     | Робоча частота | Максимальна швидкість |
|-----------|----------------|-----------------------|
| 802.11b   | 2,4 ГГц        | 11 Мбіт/с             |
| 802.11a/g | 2,4 ГГц        | 54 Мбіт/с             |
| 802.11n   | 2,4 ГГц, 5 ГГц | 600 Мбіт/с            |
| 802.11ac  | 5 ГГц          | 3,5 Гбіт/с            |
| 802.11ax  | 5 ГГц          | 9,6 Гбіт/с            |

Зазвичай, в мікроконтролерах серії ESP32 ставлять модулі стандарту b/g/n (підтримка 802.11 до 150 Мбіт/с)

Оскільки зараз Wi-Fi роутери стоять майже в кожній квартирі, дуже важливо забезпечити надійність зв'язку від втрат та колізій, а також зберігати високу швидкість. Для цього використовують такі технології:

MIMO (Multiple Input Multiple Output) - використання декількох антен для одночасної передачі та прийому даних, що збільшує пропускну здатність та стійкість до перешкод;

Beamforming - спрямоване формування радіосигналу у бік клієнта, що підвищує якість зв'язку та дальність роботи;

OFDM (Orthogonal Frequency Division Multiplexing) - розділення каналу на піднесучі частоти для підвищення ефективності передачі даних;

CSMA/CA - механізм доступу до середовища, що зменшує колізії у багатокористувацьких мережах.,

Завдяки цим технологіям, мережа дуже стійка до постійних перевантажень і сусідні роутери майже не погіршують зв'язок.

#### 2.4.2 TCP

TCP (Transmission Control Protocol) — один із базових транспортних протоколів стеку TCP/IP, який забезпечує надійну, впорядковану та контрольовану передачу даних між мережевими пристроями. Його основною особливістю є гарантія доставки пакетів, контроль цілісності даних та автоматична повторна передача втрачених сегментів.

Протокол TCP був розроблений у рамках досліджень мережі ARPANET у 1970-х роках американськими інженерами Вінтоном Серфом та Робертом Каном. У 1981 році протокол був стандартизований у документі RFC 793, який став базовою специфікацією для реалізації сучасних інтернет-мереж.

TCP є орієнтованим на з'єднання протоколом, що означає необхідність встановлення логічного каналу між відправником і отримувачем перед початком передачі даних. Процес встановлення з'єднання реалізується через триетапне рукостискання (Three-way handshake): SYN - SYN-ACK - ACK. Після встановлення з'єднання дані передаються у вигляді послідовності сегментів.

|       |      |          |        |      |                    |      |
|-------|------|----------|--------|------|--------------------|------|
|       |      |          |        |      | ІАЛЦ.045430.004 ПЗ | Арк. |
|       |      |          |        |      |                    | 28   |
| Змін. | Арк. | № докум. | Підпис | Дата |                    |      |

Кожний TCP-сегмент складається з заголовка та корисного навантаження (payload). Мінімальний розмір заголовка становить 20 байт і може збільшуватись за рахунок додаткових опцій (Рис. 12)

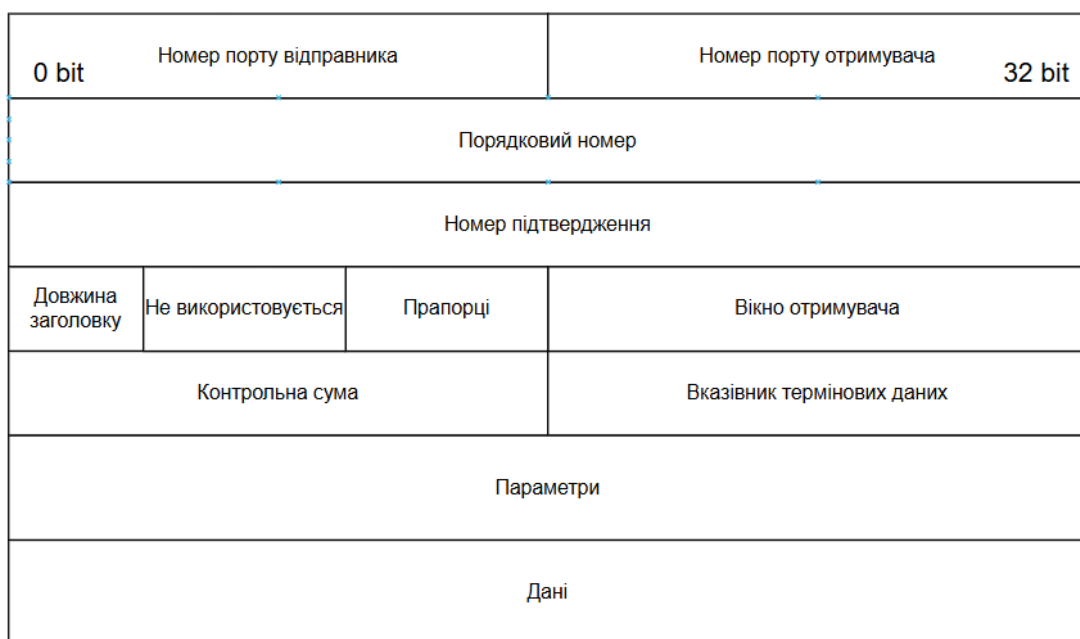


Рисунок 12 – структура заголовку протоколу TCP

У другому розділі було проведено аналіз та обґрунтування вибору апаратних і програмних засобів, що використовуються для розробки системи керування безпілотним літальним апаратом. Вибір кожного компонента здійснювався з урахуванням вимог до продуктивності, вартості, доступності, можливостей масштабування та сумісності між окремими елементами системи.

Як центральний обчислювальний модуль було обрано мікроконтролер ESP32-WROOM-32, який поєднує достатню обчислювальну потужність, розвинений набір периферійних інтерфейсів та вбудований модуль бездротового зв'язку Wi-Fi. Проведений аналіз показав, що наявність двоядерного процесора, значної кількості GPIO, підтримки інтерфейсів SPI, I2C, UART, PWM, ADC та DAC дозволяє використовувати дану платформу як основу для побудови польотного контролера без необхідності застосування додаткових спеціалізованих обчислювальних модулів. Для створення програмного забезпечення мікроконтролера було обрано середовище Arduino

IDE, яке забезпечує швидкий цикл розробки, простий процес компіляції та завантаження програмного коду, а також доступ до великої кількості готових бібліотек. Використання Arduino Core для ESP32 дозволило поєднати простоту розробки із можливістю використання апаратних ресурсів мікроконтролера на низькому рівні.

Особливу увагу було приділено використанню операційної системи реального часу FreeRTOS. Аналіз її архітектури показав, що застосування багатозадачності є необхідною умовою для побудови сучасного польотного контролера, оскільки система одночасно повинна виконувати обробку даних із сенсорів, реалізовувати алгоритми стабілізації, підтримувати мережевий обмін та забезпечувати роботу каналів зв'язку. Використання FreeRTOS дозволяє розділити функціональність системи між незалежними задачами, підвищити передбачуваність роботи програмного забезпечення та ефективно використовувати можливості двоядерної архітектури ESP32.

Окремо було розглянуто програмний стек бездротового зв'язку Wi-Fi, що входить до складу програмного забезпечення ESP32. Встановлено, що використання стандартного мережевого стека забезпечує можливість реалізації мережевої взаємодії між польотним контролером та наземною станцією без використання додаткового обладнання. Наявність підтримки режимів точки доступу, клієнта та TCP/IP-протоколів створює основу для побудови систем моніторингу, налаштування та віддаленого керування.

Для розробки програмного забезпечення наземної станції було обрано фреймворк Qt та середовище Qt Creator. Проведений аналіз підтвердив доцільність такого вибору завдяки кросплатформеності, підтримці сучасних графічних інтерфейсів, наявності вбудованих засобів мережевої взаємодії та високому рівню інтеграції компонентів. Використання Qt дозволяє створити повноцінний десктопний застосунок для конфігурування системи, відображення телеметрії та керування польотним контролером із мінімальними витратами на підтримку різних операційних систем.

|       |      |          |        |      |                     |      |
|-------|------|----------|--------|------|---------------------|------|
|       |      |          |        |      | ІАЛЦ. 045430.004 ПЗ | Арк. |
|       |      |          |        |      |                     | 30   |
| Змін. | Арк. | № докум. | Підпис | Дата |                     |      |

Також у розділі було розглянуто основні мережеві технології, використані в проєкті. Аналіз протоколу Wi-Fi показав його придатність для організації бездротового обміну даними у локальних мережах, а дослідження протоколу TCP підтвердило доцільність його використання для побудови надійного каналу зв'язку між наземною станцією та мікроконтролером. Використання TCP забезпечує гарантовану доставку повідомлень, контроль цілісності даних та підтримку постійного з'єднання між вузлами системи.

Таким чином, у результаті проведеного аналізу було сформовано програмно-апаратну основу майбутньої системи керування безпілотним літальним апаратом. Обрані засоби розробки та технології забезпечують необхідний рівень продуктивності, функціональності та надійності, а також створюють умови для реалізації багатозадачної системи керування польотом із підтримкою бездротового зв'язку, телеметрії та засобів конфігурування через наземну станцію.

## **3 ОПИС РОЗРОБЛЕНОЇ СИСТЕМИ: ПРИНЦИПИ ТА КОМПОНЕНТИ**

### **3.1 Архітектура системи**

Систему можна поділити на 3 великих модулі: безпілотний апарат, (мікроконтролер), наземна станція(мікроконтролер) і додаток.

Безпілотний апарат – мікроконтролер, на базі якого виконується пряме керування за допомогою GPIO. Він аналізує значення сенсорів, комунікує з наземною станцією, та керує двигунами.

Наземна станція – мікроконтролер, на базі якого виконується комунікація з апаратом, а також аналіз вводу користувача. Також в цей момент може паралельно спілкуватись з додатком за допомогою Wi-Fi.

Додаток виконує функцію як системи налаштувань, так і графічного аналізу роботи безпілотного апарату в залежності від того, до кого він підключений

Існує можливість працювати в різних режимах, де можуть бути не задіяні певні модулі

|       |      |          |        |      |                    |      |
|-------|------|----------|--------|------|--------------------|------|
|       |      |          |        |      | ІАЛЦ.045430.004 ПЗ | Арк. |
|       |      |          |        |      |                    | 31   |
| Змін. | Арк. | № докум. | Підпис | Дата |                    |      |

Таблиця 3 – режими роботи системи

| Безпілотний апарат | Наземна Станція | Додаток   | Режим                                        |
|--------------------|-----------------|-----------|----------------------------------------------|
| Ввімкнено          | Вимкнено        | Ввімкнено | Налаштування БА                              |
| Вимкнено           | Ввімкнено       | Ввімкнено | Налашодження НС                              |
| Ввімкнено          | Ввімкнено       | Вимкнено  | Робочий режим                                |
| Ввімкнено          | Ввімкнено       | Ввімкнено | Робочий режим з графічним аналізом (Рис. 13) |

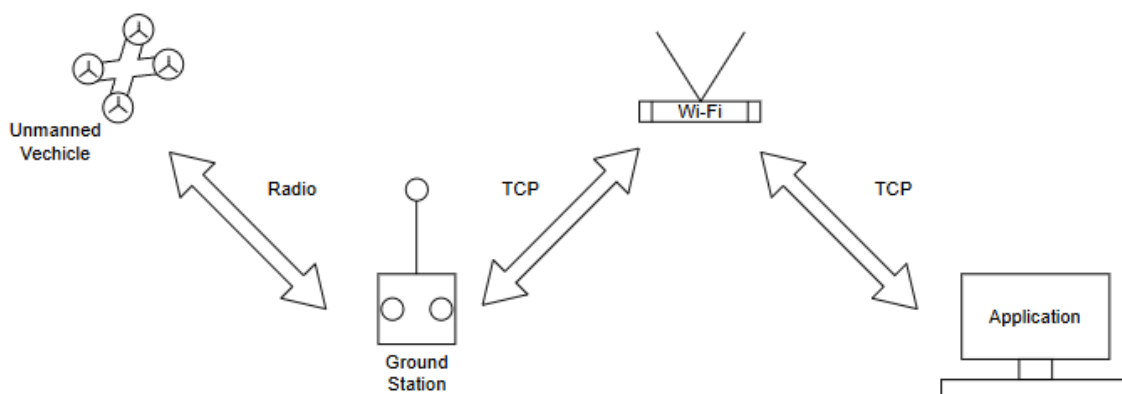


Рисунок 13 – графік взаємодії між окремими модулями

Налаштування БА – комунікація між безпілотним апаратом і додатком за допомогою мережі Wi-Fi з можливістю в додатку зчитувати налаштування БА, редагувати їх або надсилати нові налаштування .

Налашодження НС – комунікація між наземною станцією і додатком за допомогою мережі Wi-Fi, де додаток показує в графічному вигляді всі користувацькі дії на наземній станції.

Робочий режим – комунікація між безпілотним апаратом і наземною станцією згідно обраного в налаштуваннях протоколу.

Робочий режим з графічним аналізом – так само, що і робочий режим, але тепер наземна станція також надсилає додатку інформацію як від безпілотного апарату, так і від себе, додаток це аналізує і виводить графічну інформацію.

### 3.2 Безпілотний апарат

Програмне забезпечення безпілотного апарату реалізоване на базі мікроконтролера ESP32 з використанням операційної системи реального часу FreeRTOS. Основною особливістю архітектури є розподіл функціональності між різними задачами, які виконуються паралельно на двох ядрах процесора. Такий підхід дозволяє розділити процеси збору даних із датчиків, обробки команд керування, мережевої взаємодії та формування керуючих сигналів для двигунів.

Для роботи з додатком використовується структура Device, що зберігає основні важливі компоненти для роботи з девайсом (Рис. 14).

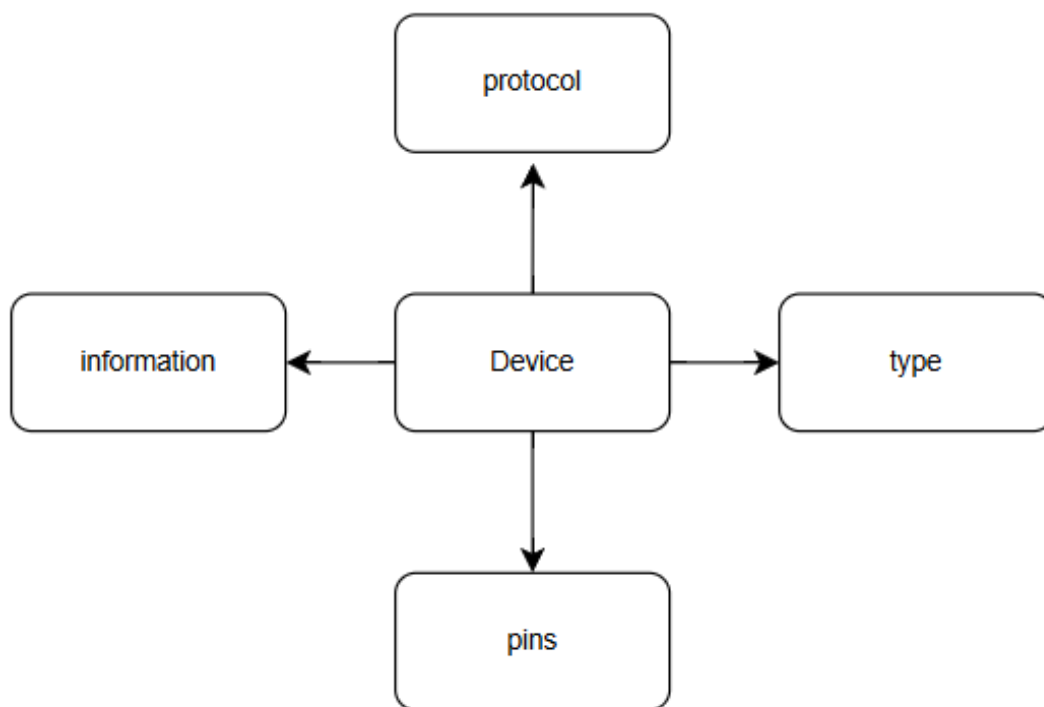


Рисунок 14 – структура Device

- Protocol – назва низькорівневого протоколу, за яким працює девайс. Це можуть бути SPI, I2C, UART і т. д.
- Type – тип модуля, необхідно для розуміння який це сенсор або вихід

для відповідної комунікації з ним.

- Pins – пінни GPIO, які займає цей девайс
- Information – додаткова інформація, що може бути корисна користувачу або прошивці.

Безпілотний апарат містить вектор таких девайсів, Додаток містить аналогічний вектор і при підключенні додатку до безпілотного апарату, БА першим пакетом надсилає власні налаштування.

Також ця структура зберігається у флеш-пам'яті пристрою і після перезавантаження пристрій не забуває власні налаштування.

Девайс має можливість комунікувати з додатком лише до початку основної роботи. Це загальноживана практика для забезпечення відсутності преривань головної програми, що матиме критичні наслідки. У випадку необхідності надіслати нові налаштування, достатньо перезавантажити контролер кнопкою BOOT

Після ініціалізації, девайс протягом 10 секунд намагається підключитись до Wi-Fi мережі, після чого розгортає TCP-сервер і протягом 30 секунд чекає на підключення клієнта (додатку). У випадку невдачі на будь-якому з етапів, контролер вимикає Wi-Fi і починає стандартну роботу.

У випадку успіху підключення, надсилає власні налаштування і входить у режим очікування нових налаштувань. Він може продовжуватись дуже довго, оскільки між налаштуваннями додаток також надсилає ехо-запити. У випадку відсутності будь-яких команд протягом 30 секунд, девайс також повертається у звичайний режим.

Під час базової роботи безпілотний апарат одночасно запускає велику кількість паралельних процесів. Для загальної роботи вони використовують спільну структуру даних – стан девайсу або SharedState (Рис. 15).

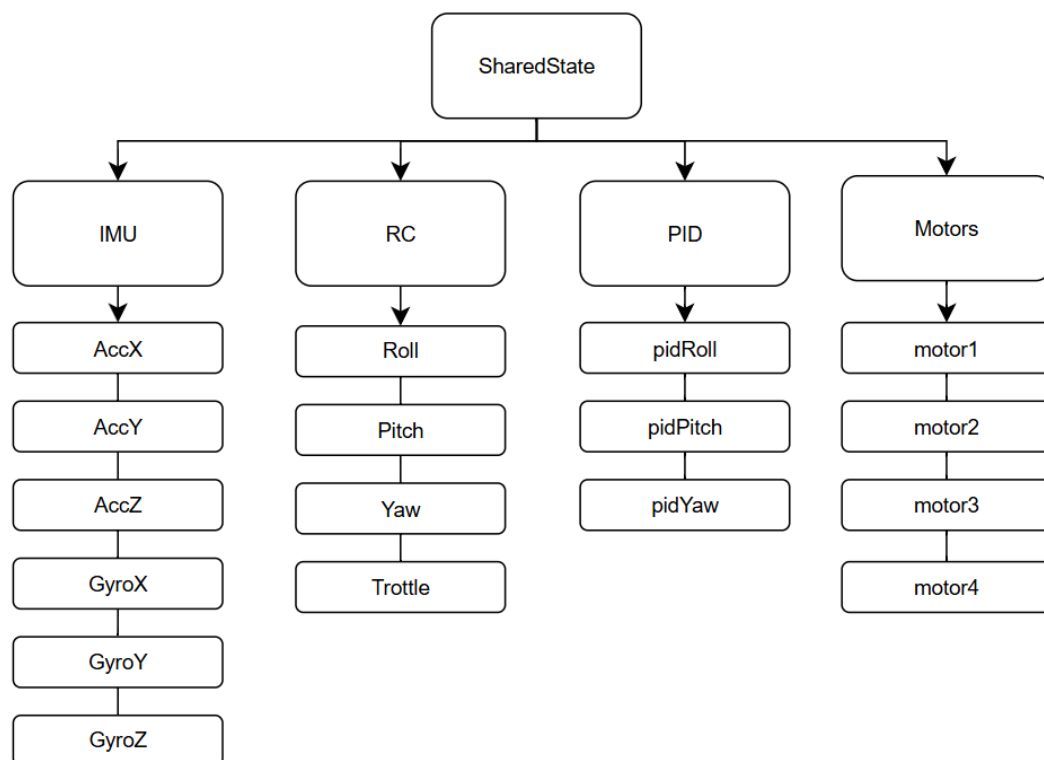


Рисунок 15 – структура SharedState

В програмному коді використовуються такі паралельні задачі:

- systemTask() – підключається до Wi-Fi і створює TCP-сервер. Вимикається у випадку відсутності клієнтів
- commTask() – задача мережевого обміну між безпілотним апаратом і додатком, вимикається размо з systemTask
- mpuTask() – задача аналізу девайсів: в першу чергу гіроскопа і акселерометра з використанням комплементарного фільтру та інтегрування кутової швидкості
- nrfTask() – задача радіообміну з наземною станцією, передає телеметрію і отримує сигнали керування
- controlTask() – задача регулювання польотом контролера (актуально саме для літаючих безпілотних апаратів) де використовуються PID-регулятори
- motorTask() – задача керування двигунами, подає необхідний сигнал на виходи мікроконтролера
- debugTask() – задача діагностики, виводу необхідних даних у Serial

|       |      |          |        |      |
|-------|------|----------|--------|------|
|       |      |          |        |      |
| Змін. | Арк. | № докум. | Підпис | Дата |

ІАЛЦ.045430.004 ПЗ

Арк.

35

ПІД-регулятор (Proportional–Integral–Derivative controller) є одним із базових алгоритмів автоматичного керування, який широко використовується в системах стабілізації динамічних об'єктів. У безпілотних літальних апаратах він застосовується для підтримання заданих кутів орієнтації: крену, тангажу та ролування.

Принцип роботи ПІД-регулятора полягає у постійному порівнянні поточного стану системи з еталонним (заданим) значенням та формуванні керуючого впливу на основі величини помилки та її динаміки.

ПІД-регулятор складається з трьох складових:

- Р (пропорційна складова) -реагує на поточну похибку;
- І (інтегральна складова) - враховує накопичену похибку;
- D (диференціальна складова) - реагує на швидкість зміни похибки.

У розробленій системі ПІД-регулювання реалізовано у задачі controlTask(). Алгоритм працює з даними, отриманими від інерціального датчика Гіроскопа, та заданими значеннями керування, які надходять із радіоканалу.

Спочатку обчислюється похибка між заданим та фактичним кутом:

$$eR = \text{targetRoll} - \text{roll}$$

$$eP = \text{targetPitch} - \text{pitch}$$

Далі формується інтегральна складова шляхом накопичення похибки у часі, а диференціальна складова визначається як різниця між поточним та попереднім значенням помилки.

У результаті керуючий сигнал обчислюється за формулами:

$$\text{outR} = K_p \cdot eR + K_i \cdot \sum eR + K_d \cdot \Delta eR$$

$$\text{outP} = K_p \cdot eP + K_i \cdot \sum eP + K_d \cdot \Delta eP$$

Отримані значення передаються до структури стану системи та використовуються для подальшого формування керуючих сигналів на двигуни.

Сигнали, що надходять з інерціального вимірювального модуля, містять значний рівень шумів, які виникають через механічні вібрації, похибки вимірювання та дрейф гіроскопа. Без попередньої обробки ці шуми призводять до нестабільної роботи системи керування.

Для усунення цих проблем у системі застосовано цифрову фільтрацію сигналів.

Цифровий фільтр — це алгоритм обробки дискретних сигналів, який дозволяє зменшити високочастотні шуми, згладити різкі стрибки вимірювань і підвищити стабільність керуючих алгоритмів.

В програмному забезпеченні безпілотного апарату було використано експоненційне згладжування:

$$\text{gyroFiltered} = \alpha \cdot \text{gyroFiltered} + (1 - \alpha) \cdot \text{gyroRaw}$$

### 3.3 Наземна станція

У нашому випадку Наземна станція - це прошивка ESP32 з архітектурою FreeRTOS та двома різними каналами для передачі інформації.

В програмному коді використовуються такі паралельні задачі:

- nrfTask() – прийом і збереження телеметрії з безпілотного апарату
- txTask() – зчитування осей джойстиків і відправка значень на девайс
- wifiTask() – підключення до мережі і розгортання власного TCP

серверу

Для комунікації між мікроконтролерами використовуються спеціальні структури для передачі: ControlPacket і TelemetryPacket. (рис. 16)

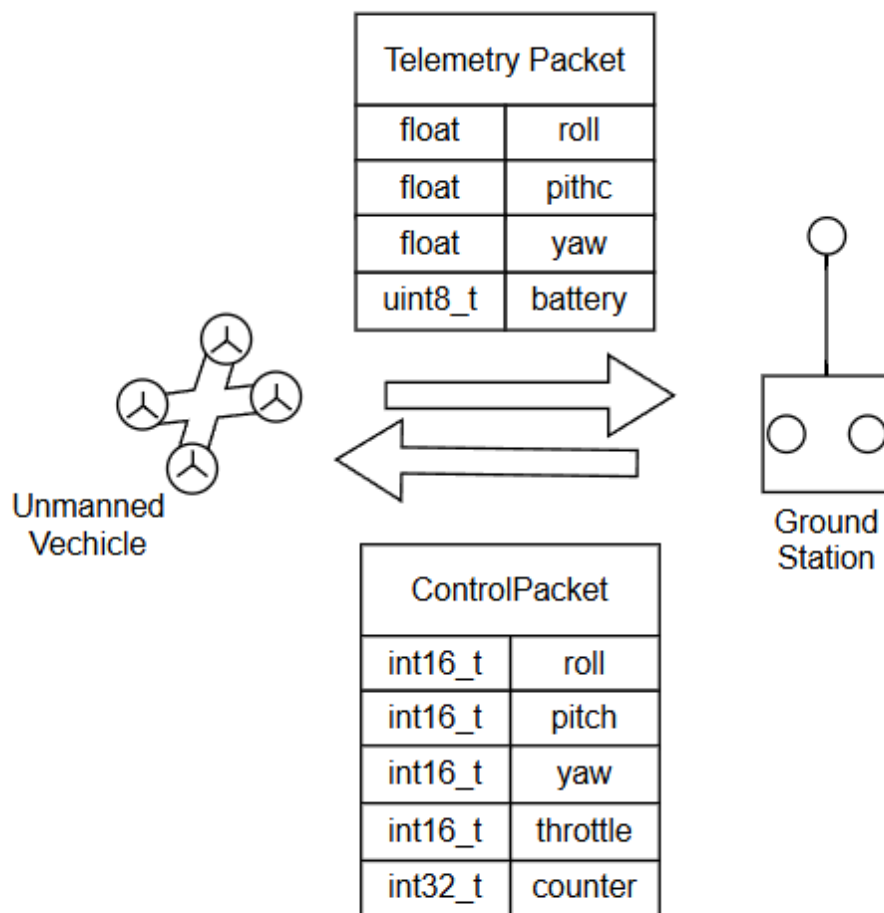


Рисунок 16 – пакети комунікації між польотним контролером і НС.

### 3.4 Десктопний додаток

Десктопний додаток – односторінковий додаток для функцій налаштування і керування безпілотним апаратом, розроблений у Qt Creator.

Інтерфейс додатку (рис. 17) містить як переважно текстову інформацію, так і графічну). Жовтими номерами позначені окремі функціональні елементи додатку.



Рисунок 17 – GUI десктопного додатку

- 1) Графічне поле 3д моделі абстрактного безпілотноного апарату, яка повторює положення реального безпілотноного апарату
- 2) Поле вводу IP для встановлення комунікації з безпілотноним апаратом або наземною станцією
- 3) Поле вводу порту для встановлення комунікації з безпілотноним апаратом або наземною станцією
- 4) Інформаційне поле вибору підключеного девайсу (UV – Unmanned Vechicle, GC – Ground Station)
- 5) Кнопка Connect для запиту комунікації до TCP сервера.
- 6) Таблиця налаштувань безпілотноного апарату, де рядок – це підключений девайс, а в стовбчику позазуються значення відповідні структурі Device (РІС11)
- 7) Поле для написання команд редагування налаштуваннями за допомогою devcom
- 8) Кнопка для відправки devcom команд, написаних в полі 7
- 9) Набір progress bar, які змінюють візуально повторюють значення каналів керування на наземній станції
- 10) Набір міток для текстового відображення шивдкості дрону і його кутів нахилу

11) Журнал для можливості додати новий елемент під час налаштувань безпілотного апарату. При додаванні нового елементу він автоматично додається в журнал з найкраще підходящими для цього пінами або не додає елемент у випадку відсутності вільних місць.

Проект(рис. 18) складається з вихідних файлів, а також файлів ui та sq

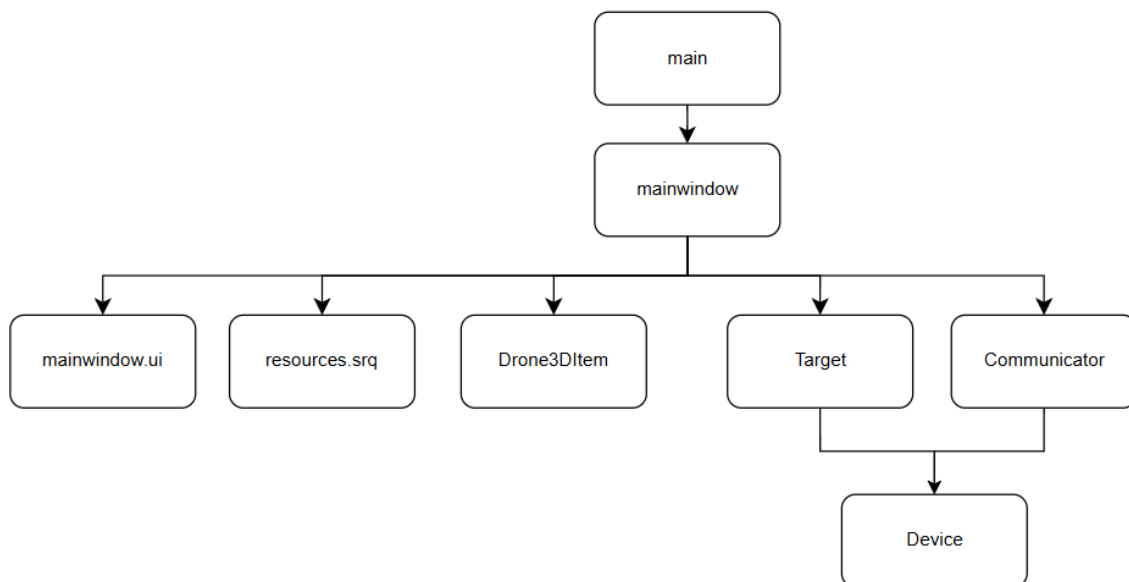


Рисунок 18 – міжмодульна взаємодія додатку

Клас Device використовується для зберігання інформації про підключений або доступний пристрій у системі конфігурації контролера. Його основні поля відповідають Device на безпілотному апараті окрім одного – Qstring image, рядка що зберігає шлях до фото.

Клас Drone3DItem – наслідується від QGraphicsItem і використовується для візуалізації орієнтації квадрокоптера в 3D-просторі всередині QGraphicsScene. Серед основних методів:

- boundingRect() – повертає область малювання об'єкта
- rotate3D(QVector3D v) – виконує обертання точки у тривимірному просторі
- paint() – відображення моделі

Клас Target описує моделює плату ESP32 Devboard та виконує роль менеджера ресурсів GPIO. На плату користувач додає різні модулі та периферію через графічний інтерфейс. Серед основних методів:

- AddDevice (Device new\_device, QVector <int> req\_pins) – додає новий пристрій за умови що вільні всі необхідні піни, інакше повертає помилку.

- DeleteDevice (int index) – видаляє пристрій мікроконтролера за індексом

Клас Communicator відповідає за мережевий обмін даними між Qt-застосунком та ESP32 через TCP-з'єднання. Він реалізує прийом телеметрії, налаштувань, парсинг TCP, передачу даних іншим компонентам. Серед основних методів:

- Connect() – ініціалізує TCP-підключення до ESP32
- Send(QString msg) – передає текстове повідомлення на контролер
- SendPing() – відправляє повідомлення без інформаційного змісту, очікує назад таке саме повідомлення

- Receve() – метод обробки вхідних даних і обробки телеметрії

Клас MainWindow –центральний клас, що містить в собі графічний інтерфейс, власний комунікатор і конфігурацію пристрою. Серед основних методів:

- onTelemetry() – оброблює вхідну телеметрію для дрона і оновлює індикатори безпілотного апарату і канали наземної станції.

- buildQtSStateString() – перетворює конфігурацію плати в рядок для зручного передавання в communicator.

- parseTargetState() – перетворює отриманий рядок в нові налаштування конфігурації

- refreshTable() – будує таблицю згідно налаштувань target

- addRow() – додає в таблицю доданий на таргет модуль.

### 3.5 Протокол комунікації додатку

Для комунікації додатку з пристроями було розроблено протокол поверх TCP. У Безпілотного апарату і наземної станції вони відрізняються

Комунікація між додатком і безпілотним апаратом (рис. 19) працює так: Після підключення по Wi-Fi безпілотний апарат одразу надсилає свої налаштування у форматі:

- QT – заголовок, далі йде інформація про N доданих девайсів, де protocolN – протокол модуля N

- TypeN – тип модуля N
- InfoN – інформація модуля N
- PinsN - піни до яких підключений модуль N

Після чого додаток приймає їх і починає щосекунди надсилати PING, девайс на кожен PING відповідає PONG. У випадку, коли юзер сконфігурував нові налаштування, то вже додаток надсилає пакет з налаштуваннями.

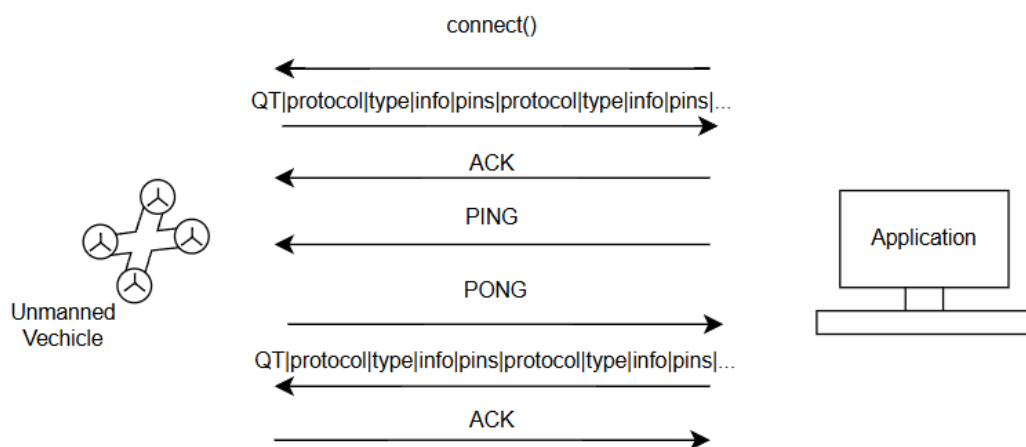


Рисунок 19 - Комунікація між безпілотним апаратом і додатком

Комунікація між додатком і наземною станцією (рис. 20) працює так: Після підключення по WiFi наземна станція починає з доволі швидко надсилати пакети з телеметрією, де

- QT – заголовок
- ch1-ch4 – канали керування базової станції.
- Battery – рівень заряду безпілотного апарату.
- Roll, pitch, yaw – осі нахилу аксилерометра безпілотного апарату

У випадку, якщо безпілотний апарат не підключено до наземної станції, то вона передаватиме нулі усюди, окрім каналів керування

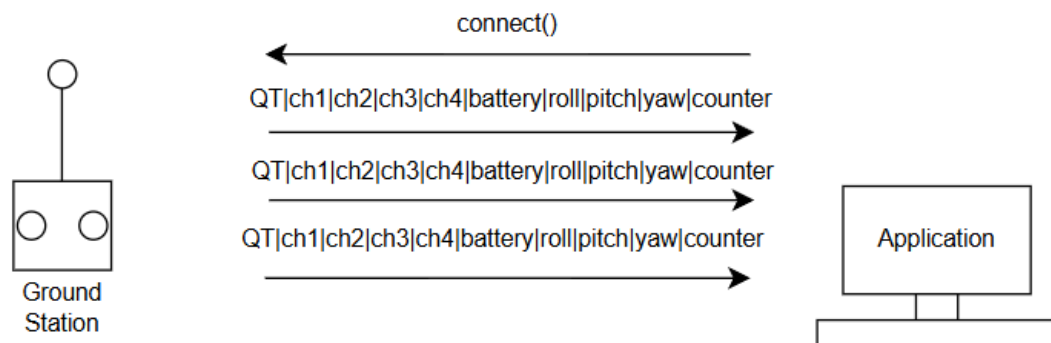


Рисунок 20 - Комунікація між Наземною станцією і додатком

### 3.6 Модулі для відладки проєкту:

Джойстик KY-023 (рис. 21) – популярне рішення для вводу інформації. Він має 2 аналогові пini, що характеризують нахили по осям VRx та VRy, а також 1 цифровий pin SW, що відповідає за кнопку. В даташиті \* джойстика показано живлення 5В, хоча це необов'язково. У випадку підключення 3.3В він буде працювати, але вже за максимального значення вихідних значень 3.3В, тобто осі працюють як потенціометри.



Рисунок 21 - джойстик KY-023

Два таких джойстики можна під'єднати до наземної станції і забезпечити мінімальне керування безпілотним апаратом

MPU 6050 GY-521 (рис. 22) – компактний модуль трьохосьового акселерометра і трьохосьового гіроскопу, що комунікує за допомогою протоколу I2C.

Всередині модуля використовуються мікромеханічні структури (MEMS), розроблені прямо на кристалі. У випадку акселерометра всередині знаходиться інерційна маса підвищена на мікропружинах. У випадку зміни положення відбувається зміщення маси, а отже і ємність між електродами. Гіроскоп в свою чергу працює за допомогою сили Коріоліса

Модуль має такі виходи:

- VCC – Живлення модуля, розраховано на 3.3V
- GND – Земля
- SCL – тактування I2C
- SDA – дані I2C
- XDA – SDA для допоміжної шини
- XCL – SCL для допоміжної шини
- AD0 – вибір I2C адреси
- INT – вивід апаратного преривання



Рисунок 22 - модуль MPU 6050 GY-521

Такий варіант ідеально підійде для аналізу положення безпілотного апарату. Для цього буде достатньо підключити живлення, GND, SCL і SDA.

|       |      |          |        |      |                     |      |
|-------|------|----------|--------|------|---------------------|------|
|       |      |          |        |      | ІАЛЦ. 045430.004 ПЗ | Арк. |
|       |      |          |        |      |                     | 44   |
| Змін. | Арк. | № докум. | Підпис | Дата |                     |      |

NRF24L01 (рис. 23) – модуль безпроводного зв'язку, що працює на частоті 2.4ГГц. Кожен модуль може працювати як приймач а бо передатчик. Керується за допомогою протоколу SPI.

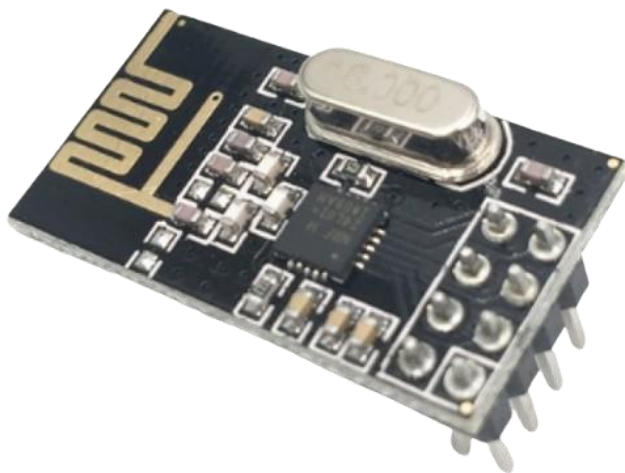


Рисунок 23 - модуль безпроводного зв'язку NRF24L01

Його основні характеристики це: робота від напруги 2.7-3.6В, робоча частота 2400-2524 МГц, модуляція GMSK, споживання середнє: 45мА, максимальне 115 мА. Швидкість передачі до 2 Мбіт/сек.

Як було помічено, максимальне споживання цього модуля набагато більше, ніж живлення ESP32, що часто призводить до ребути плати. Але оскільки 115 мА - це максимальне значення, що досягається лише в піках, то найпростішим вирішенням буде припаяти паралельно до ліній живлення конденсатор на 100 мікрофард. В такому випадку при пікових споживаннях NRF буде просто розряджати конденсатор після чого ESP почне знову його заряджати.

Модуль має такі виходи:

Таблиця 4 – піни роботи модуля NRF24L01

| Пін  | Назва | Опис               | Направлення |
|------|-------|--------------------|-------------|
| Пін1 | GND   | Земля              | -           |
| Пін2 | VCC   | Живлення 2.7-3.6В  | -           |
| Пін3 | CE    | Режим роботи RX/TX | Вхід        |

|      |       |                   |       |
|------|-------|-------------------|-------|
| Пін4 | CSN   | SPI chip select   | Вхід  |
| Пін5 | SCK   | SPI clock         | Вхід  |
| Пін6 | MOSI  | SPI Input         | Вхід  |
| Пін7 | MISO  | SPI Output        | Вихід |
| Пін8 | I/OAM | Вихід переривання | Вихід |

Такі модулі можна використовувати як інструмент комунікації між безпілотним апаратом і наземною станцією.

У даному розділі було розглянуто структуру та принципи функціонування розробленої системи керування безпілотним літальним апаратом, а також описано основні програмні та апаратні компоненти, що забезпечують її роботу. У результаті проектування була сформована модульна архітектура, яка складається з трьох взаємопов'язаних підсистем: безпілотного апарата, наземної станції та десктопного застосунку для конфігурування й моніторингу.

Розроблена архітектура забезпечує можливість роботи системи в декількох режимах залежно від поточних потреб користувача. Передбачено окремі режими налаштування безпілотного апарата, налагодження наземної станції, автономної роботи радіоканалу та повноцінного режиму з графічним відображенням телеметричної інформації. Такий підхід дозволяє ефективно використовувати систему як під час розробки та тестування, так і в процесі експлуатації.

У ході розробки програмного забезпечення польотного контролера було реалізовано багатозадачну архітектуру на основі FreeRTOS. Використання окремих задач для збору даних із сенсорів, мережевої взаємодії, радіообміну, стабілізації польоту, керування двигунами та діагностики дозволило розділити функціональність системи на незалежні логічні модулі. Застосування спільної структури стану SharedState забезпечило централізоване зберігання інформації та взаємодію між окремими процесами без дублювання даних.

Особливу увагу було приділено реалізації алгоритмів стабілізації польоту. Для визначення орієнтації літального апарата використано

|       |      |          |        |      |                    |      |
|-------|------|----------|--------|------|--------------------|------|
|       |      |          |        |      | ІАЛЦ.045430.004 ПЗ | Арк. |
|       |      |          |        |      |                    | 46   |
| Змін. | Арк. | № докум. | Підпис | Дата |                    |      |

інерціальний вимірювальний модуль MPU6050, дані якого обробляються за допомогою цифрової фільтрації та комплементарного алгоритму об'єднання показників акселерометра і гіроскопа. Це дозволило отримати стабільні оцінки кутів нахилу та зменшити вплив шумів і похибок вимірювання. Для автоматичної стабілізації апарата реалізовано PID-регулятор, який формує керуючі сигнали на основі відхилення поточного положення від заданого оператором значення.

Для організації зв'язку між безпілотним апаратом та наземною станцією використано радіомодулі NRF24L01, які забезпечують передачу команд керування та телеметричних даних у режимі реального часу. Для підвищення надійності передачі були розроблені власні структури пакетів ControlPacket та TelemetryPacket, що дозволяють компактно передавати необхідну інформацію між мікроконтролерами.

Окремим компонентом системи став десктопний застосунок, розроблений із використанням фреймворку Qt. Програмне забезпечення забезпечує віддалене налаштування конфігурації контролера, моніторинг телеметричних параметрів, візуалізацію положення безпілотного апарата у тривимірному просторі та контроль стану підключених модулів. Для цього було розроблено набір спеціалізованих класів, що відповідають за мережевий обмін, управління конфігурацією, графічне відображення інформації та керування ресурсами мікроконтролера.

Також у розділі було описано розроблений прикладний протокол обміну даними поверх TCP, який забезпечує взаємодію між десктопним застосунком та мікроконтролерами. Реалізований протокол підтримує передачу конфігураційних параметрів, телеметричних даних та службових повідомлень, що дозволяє організувати єдиний механізм комунікації між усіма компонентами системи.

Для практичної перевірки працездатності програмного забезпечення було обрано набір апаратних модулів, до складу якого увійшли джойстики KY-023, інерціальний модуль MPU6050 та радіомодулі NRF24L01. Проведений аналіз

|       |      |          |        |      |                    |      |
|-------|------|----------|--------|------|--------------------|------|
|       |      |          |        |      | ІАЛЦ.045430.004 ПЗ | Арк. |
|       |      |          |        |      |                    | 47   |
| Змін. | Арк. | № докум. | Підпис | Дата |                    |      |

показав, що обрані компоненти повністю задовольняють вимоги проєкту та дозволяють реалізувати всі необхідні функції керування, стабілізації та передачі даних.

Таким чином, у результаті виконання третього розділу було розроблено повну програмно-апаратну архітектуру системи керування безпілотним літальним апаратом, реалізовано механізми взаємодії між її складовими, алгоритми стабілізації та засоби моніторингу. Отримані результати створюють завершену основу для побудови працездатної системи керування безпілотним апаратом та підтверджують можливість використання мікроконтролерів ESP32 як універсальної платформи для подібних рішень.

|       |      |          |        |      |                     |      |
|-------|------|----------|--------|------|---------------------|------|
|       |      |          |        |      | ІАЛЦ. 045430.004 ПЗ | Арк. |
|       |      |          |        |      |                     | 48   |
| Змін. | Арк. | № докум. | Підпис | Дата |                     |      |

## ВИСНОВКИ

У ході виконання дипломного проєкту на тему «Вбудована програмна система дистанційного керування безпілотними апаратами» було розроблено, реалізовано та протестовано багатофункціональну вбудовану систему, що поєднує програмну обробку даних у реальному часі та інтерфейс для віддаленої взаємодії.

Метою проєкту було створення дешевого, доступного та простого рішення для стабільного управління безпілотним апаратом, здатним виконувати поставлені завдання та надавати користувачу актуальну телеметричну інформацію. Під час відладки проєкту було досягнуто повної інтеграції між апаратною частиною та програмною логікою, реалізованою з використанням FreeRTOS. Особливу увагу приділено впровадженню алгоритмів стабілізації польоту на базі ПІД-регуляторів, цифрової фільтрації даних сенсорів, а також протоколів передачі інформації.

Система забезпечує безпечний та надійний обмін даними з наземною станцією через радіоканал, що дозволяє дистанційно керувати дроном, отримувати телеметрію та здійснювати оновлення параметрів польоту. Програмний інтерфейс дозволяє користувачу швидко інтегрувати нові датчики та пристрої без потреби зміни основної архітектури.

Частину системи побудовано на основі ESP32, що забезпечує високу обчислювальну потужність при низькій вартості. Хоча проєкт не замінює комерційні системи повного автопілотування, закладена архітектура демонструє практичність, функціональність і перспективу подальшого розвитку.

У подальшій роботі систему можна значно розширити, зокрема інтегрувати прошивку на інші мікроконтролери, додати можливість роботи за місією, розширювати кількість підтримуваних модулів.

Виконана робота досягла поставлених цілей і заклала основу для подальших досліджень у сфері вбудованих систем керування безпілотними апаратами.

|       |      |          |        |      |                    |      |
|-------|------|----------|--------|------|--------------------|------|
|       |      |          |        |      | ІАЛЦ.045430.004 ПЗ | Арк. |
|       |      |          |        |      |                    | 49   |
| Змін. | Арк. | № докум. | Підпис | Дата |                    |      |

Розроблений комплекс є прикладом ефективної інтеграції доступної апаратної платформи, сучасних алгоритмів стабілізації та гнучкої програмної архітектури, що робить його перспективною основою для подальших прикладних застосувань.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ ТА ЛІТЕРАТУРИ

1. Espressif Systems. ESP32 Technical Reference Manual [Електронний ресурс]. Shanghai: Espressif Systems, 2026. Режим доступу: <https://www.espressif.com/en/support/documents/technical-documents/esp32->. Дата звернення: 03.04.2026.
2. Espressif Systems. ESP32 Datasheet [Електронний ресурс]. Shanghai: Espressif Systems, 2026. Режим доступу: <https://docs.espressif.com/projects/esp-idf/en/v4.2.2/esp32/hw-reference/index.html>. Дата звернення: 03.04.2026.
3. Espressif Systems. ESP-IDF Programming Guide [Електронний ресурс]. Shanghai: Espressif Systems, 2026. Режим доступу: <https://www.espressif.com/en/support/documents/technical-documents/esp32->. Дата звернення: 03.04.2026.
4. ArduPilot Development Team. ArduPilot Documentation [Електронний ресурс]. 2026. Режим доступу: <https://ardupilot.org/ardupilot/>. Дата звернення: 15.04.2026.
5. PX4 Development Team. PX4 User Guide and Developer Documentation [Електронний ресурс]. 2026. Режим доступу: [https://docs.px4.io/main/en/hardware/porting\\_guide\\_nuttx.html](https://docs.px4.io/main/en/hardware/porting_guide_nuttx.html). Дата звернення: 15.04.2026.
6. Apache Software Foundation. Apache NuttX RTOS Documentation [Електронний ресурс]. 2026. Режим доступу: <https://nuttx.apache.org/docs/latest/>. Дата звернення: 15.04.2026.
7. Valvano J. W. Embedded Systems: Real-Time Interfacing to ARM Cortex-M Microcontrollers. Sebastopol: CreateSpace Independent Publishing Platform, 2014. 488 с.
8. Valvano J. W. Embedded Systems: Real-Time Operating Systems for ARM Cortex-M Microcontrollers. Sebastopol: CreateSpace Independent Publishing Platform, 2015. 522 с.
9. Barr M., Massa A. Programming Embedded Systems with C and GNU Development Tools. 2nd ed. Sebastopol: O'Reilly Media, 2006. 336 с.

|       |      |          |        |      |                     |      |
|-------|------|----------|--------|------|---------------------|------|
|       |      |          |        |      | ІАЛЦ. 045430.004 ПЗ | Арк. |
|       |      |          |        |      |                     | 51   |
| Змін. | Арк. | № докум. | Підпис | Дата |                     |      |

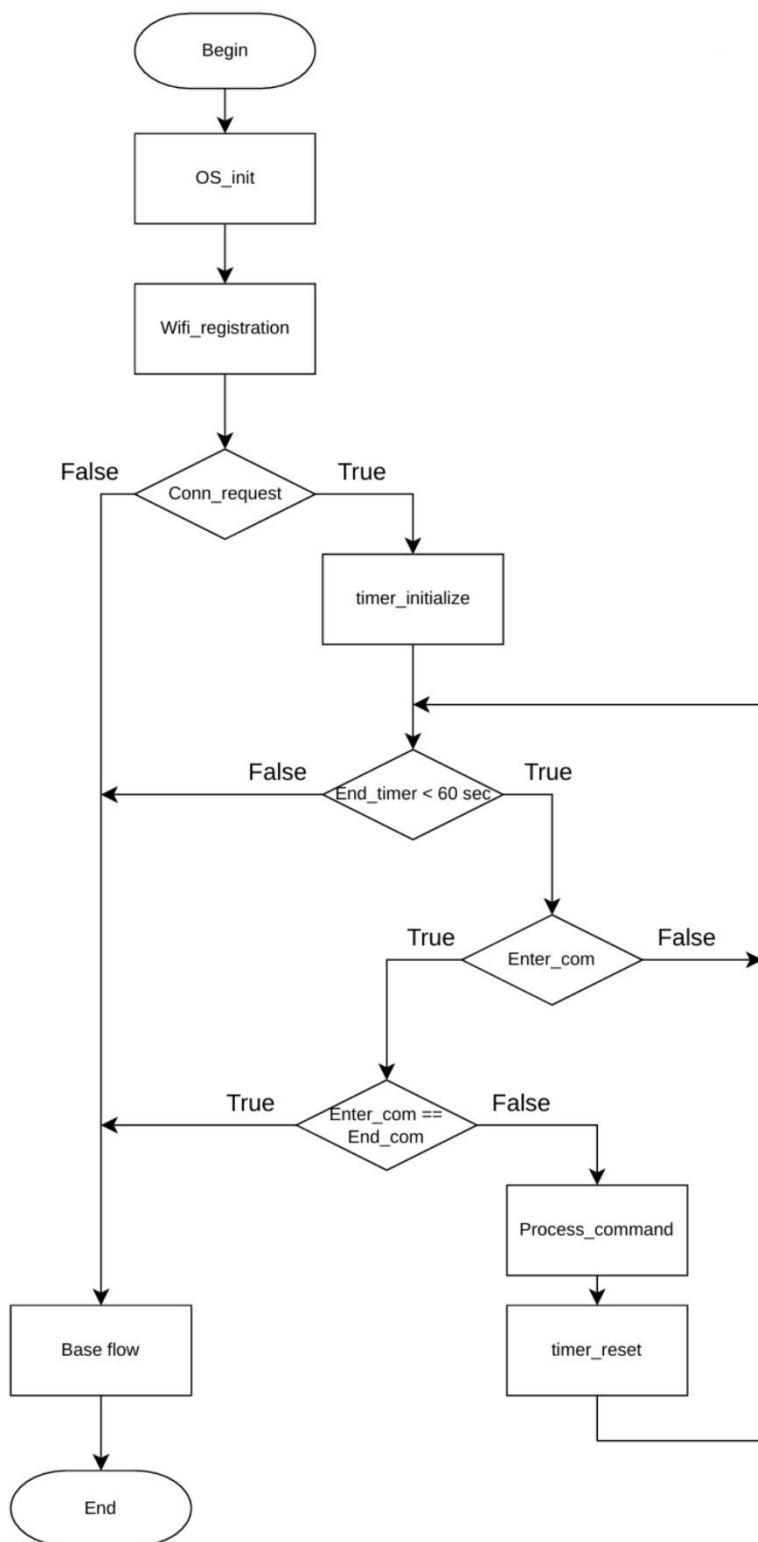
10. Samek M. Practical UML Statecharts in C/C++, Event-Driven Programming for Embedded Systems. 2nd ed. Newnes, 2008. 728 с.
11. Labrosse J. J. MicroC/OS-II: The Real-Time Kernel. 2nd ed. CMP Books, 2002. 664 с.
12. FreeRTOS Community. FreeRTOS Kernel Developer Documentation [Електронний ресурс]. 2026. Режим доступу: <https://www.freertos.org/>. Дата звернення: 09.05.2026.
13. Koubaa A., et al. Micro Air Vehicle Link (MAVLink) in a Nutshell: A Survey // IEEE Access. 2019. Режим доступу: <https://arxiv.org/abs/1906.10641>. Дата звернення: 09.05.2026.
14. Allouch A., et al. MAVSec: Securing the MAVLink Protocol for ArduPilot/PX4 Unmanned Aerial Systems // International Conference on Unmanned Aircraft Systems, 2019. Режим доступу: <https://arxiv.org/abs/1905.00265>. Дата звернення: 10.05.2026.
15. Sardinha H. R. M., Dragone M., Vargas P. A. Closing the Gap in Swarm Robotics Simulations: An Extended ArduPilot/Gazebo Plugin [Електронний ресурс] // arXiv, 2018. Режим доступу: <https://arxiv.org/abs/1811.06948>. Дата звернення: 25.05.2026.

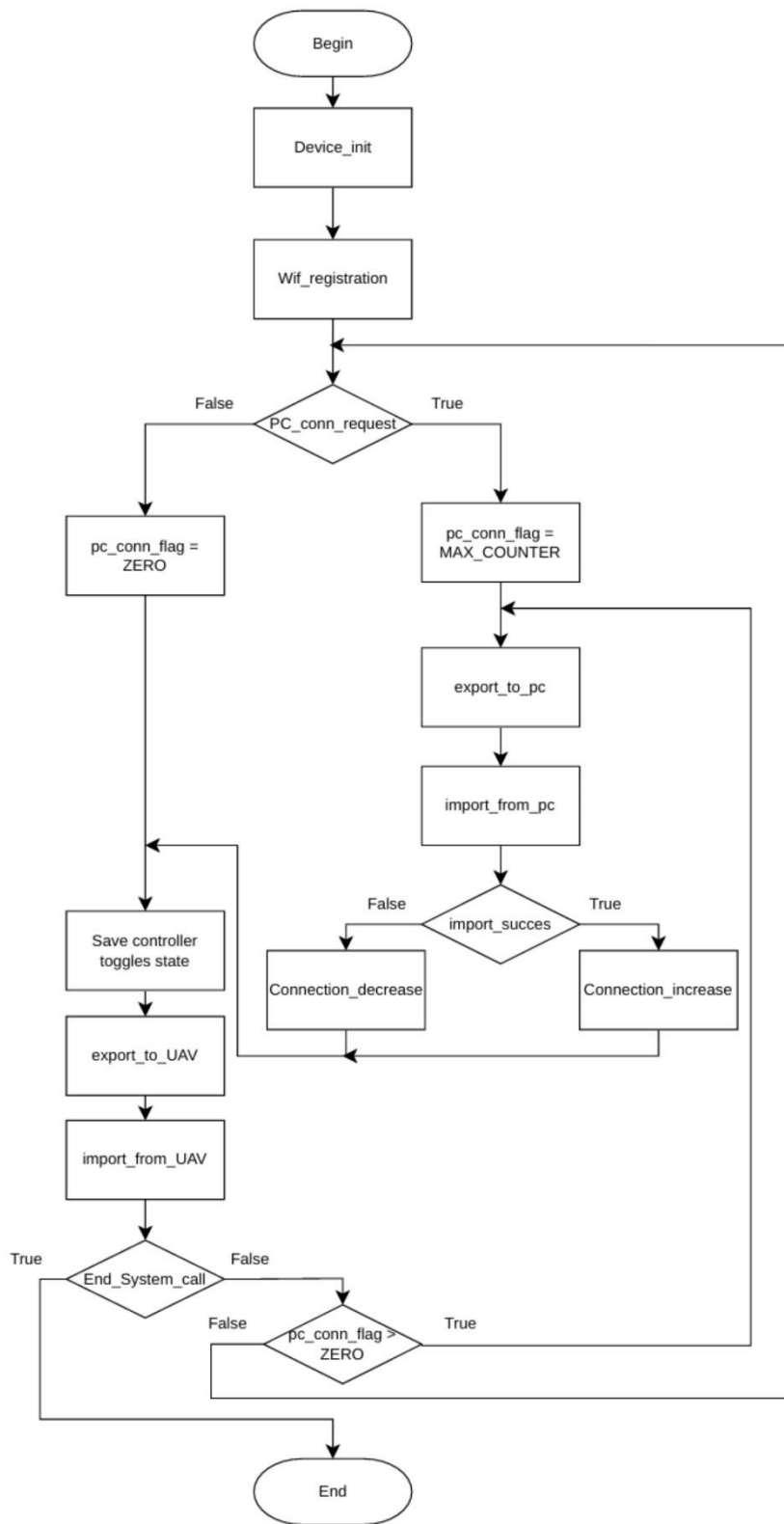
## **ДОДАТКИ**

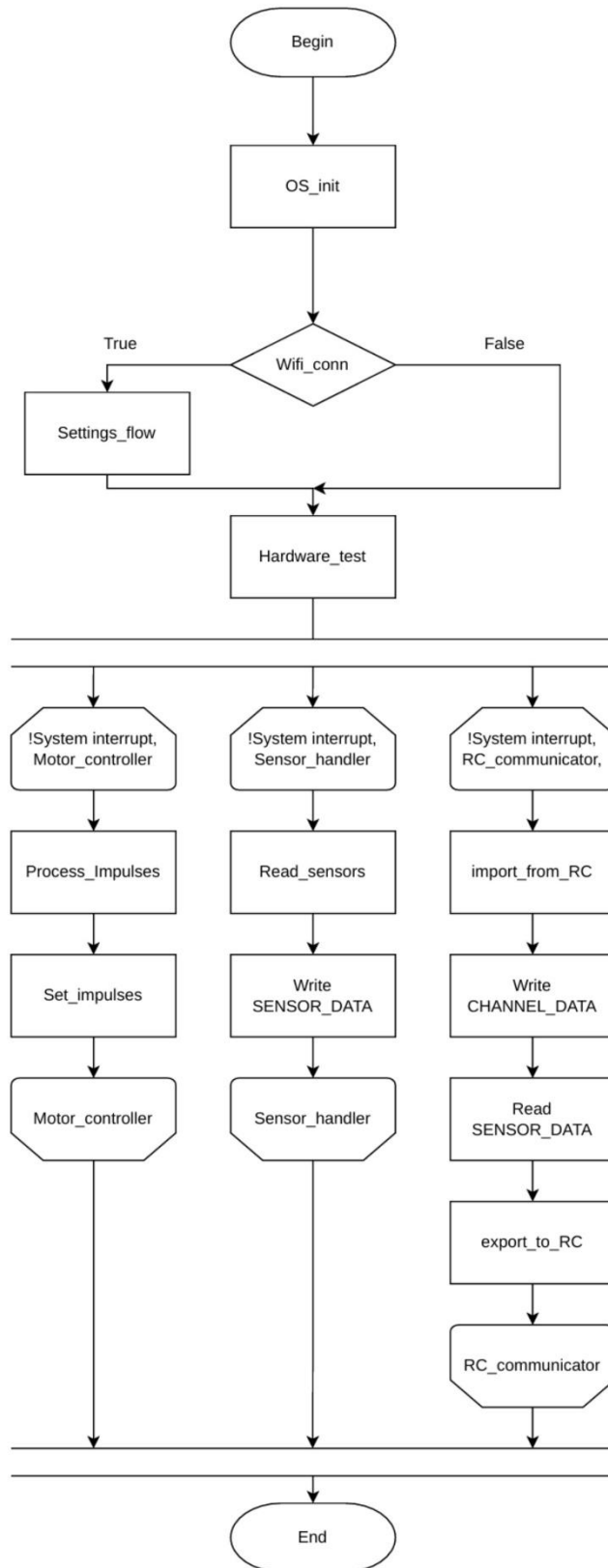
Додаток 1.

Копії графічного матеріалу

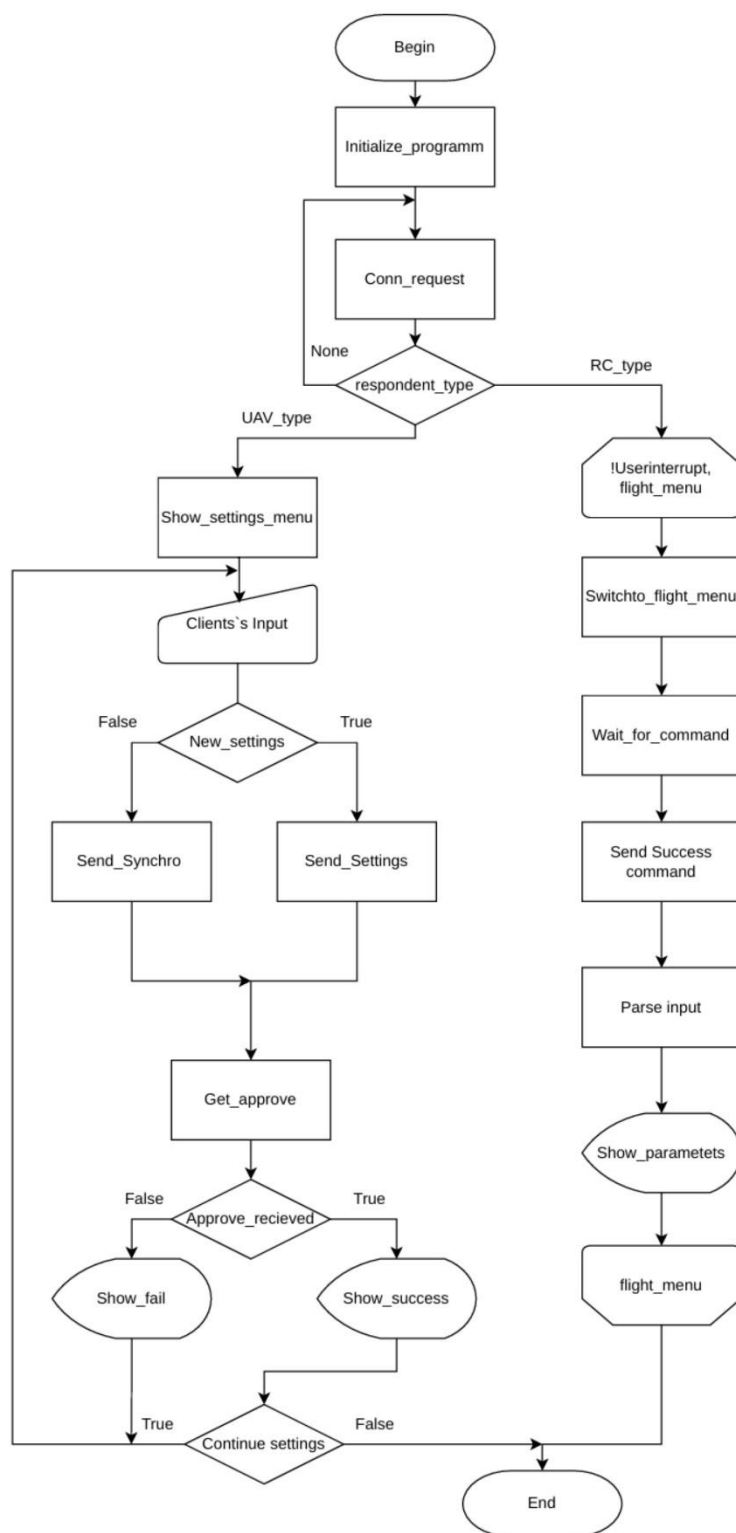








|                     |         |            |        |                                                                  |       |       |       |
|---------------------|---------|------------|--------|------------------------------------------------------------------|-------|-------|-------|
| ІА/ПЦ.045430.008 Д4 |         |            |        | Робота системного менеджера автономного робота - схема алгоритму |       |       |       |
| Дата                | Відп.   | Н. розроб. | Ізмен. | Дата                                                             | Відр. | Відр. | Відр. |
| Розроб.             | Стор. 1 |            |        |                                                                  |       |       |       |
| Стор. 1             | Стор. 1 |            |        |                                                                  |       |       |       |
| Відр.               | Відр.   |            |        |                                                                  |       |       |       |
| Відр.               | Відр.   |            |        |                                                                  |       |       |       |
| Відр.               | Відр.   |            |        |                                                                  |       |       |       |

[illegible]

Додаток 2.

Фрагменти коду

## mainwindow.cpp

```
#include "mainwindow.h"

#include "ui_mainwindow.h"

QString MainWindow::buildQtStateString()
{
    QString msg = "QT";

    for (int i = 0; i < connected_target.connected_devices.size(); i++)
    {
        const Device &d = connected_target.connected_devices[i];

        msg += "|" + d.protocol;
        msg += "|" + d.type;
        msg += "|" + d.information;

        QString pins;

        for (int p : d.pins)
            pins += QString::number(p) + ",";

        if (!pins.isEmpty())
            pins.chop(1);

        msg += "|" + (pins.isEmpty() ? "-" : pins);
    }

    return msg;
}

void MainWindow::parseTargetState(QString msg)
{
    QStringList parts = msg.split("|");

    if (parts.size() < 5)
        return;

    if (parts[0] != "QT")
        return;

    connected_target.connected_devices.clear();

    for (int i = 1; i + 3 < parts.size(); i += 4)
    {
        Device d;

        d.protocol    = parts[i];
        d.type        = parts[i + 1];
        d.information = parts[i + 2];

        QString pinStr = parts[i + 3];

        if (pinStr != "-")
        {
            QStringList pinList = pinStr.split(",", Qt::SkipEmptyParts);

            for (const QString &p : pinList)
                d.pins.push_back(p.toInt());
        }

        connected_target.connected_devices.push_back(d);
    }

    refreshTable();
}

void MainWindow::refreshTable()
{
    ui->targetSettings->setRowCount(0);

    for (int i = 0; i < connected_target.connected_devices.size(); i++)
    {
        const Device &d = connected_target.connected_devices[i];
```

```

        int row = ui->targetSettings->rowCount();
        ui->targetSettings->insertRow(row);

        ui->targetSettings->setItem(row, 0, new QTableWidgetItem(QString::number(row)));
        ui->targetSettings->setItem(row, 1, new QTableWidgetItem(d.type));
        ui->targetSettings->setItem(row, 2, new QTableWidgetItem(d.protocol));
        ui->targetSettings->setItem(row, 3, new QTableWidgetItem(d.information));

        QString pins;
        for (int p : d.pins)
            pins += QString::number(p) + " ";
        ui->targetSettings->setItem(row, 4, new QTableWidgetItem(pins));

    }
}

void MainWindow::setGraphicsImage(QGraphicsView *view, const QString &imagePath)
{
    QPixmap pix(":/images/" + imagePath);

    if (pix.isNull())
    {
        qDebug() << "Image not found:" << imagePath;
        return;
    }

    QGraphicsScene *scene = new QGraphicsScene(this);

    QGraphicsPixmapItem *item = scene->addPixmap(pix);
    item->setPos(0, 0);

    scene->setSceneRect(pix.rect());

    view->setScene(scene);
    view->setRenderHint(QPainter::SmoothPixmapTransform);

    view->fitInView(scene->sceneRect(), Qt::KeepAspectRatio);
}

void Drone3DItem::setRotationAngles(float r, float p, float y)
{
    roll = r;
    pitch = p;
    yaw = y;
}

void MainWindow::updateDrone(float roll, float pitch, float yaw)
{
    droneItem->setRotationAngles(roll, pitch, yaw);
    droneItem->update();

    ui->label_3->setText(QString("ROLL: %1°").arg(roll, 0, 'f', 1));
    ui->label_2->setText(QString("PITCH: %1°").arg(pitch, 0, 'f', 1));
    ui->label->setText(QString("YAW: %1°").arg(yaw, 0, 'f', 1));
}

void MainWindow::onTelemetryReceived(float r, float p, float y)
{
    updateDrone(r, p, y);
}

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    connection_mode = 0;
    ui->setUpUi(this);

    scene = new QGraphicsScene(this);

```

```

scene = new QGraphicsScene(this);

droneItem = new Drone3DItem();
scene->addItem(droneItem);

ui->graphicsfield->setScene(scene);

ui->ch1_bar->setRange(-1000, 1000);
ui->ch2_bar->setRange(-1000, 1000);
ui->ch3_bar->setRange(-1000, 1000);
ui->ch4_bar->setRange(-1000, 1000);

QTimer *t = new QTimer(this);

connect(&communicator, &Communicator::TargetStateReceived,
        this, &MainWindow::parseTargetState);

setStyleSheet(R"(
MainWindow
{
    background-color: #151515;
}

QWidget
{
    color: #E0E0E0;
    font-size: 10pt;
}

QGroupBox
{
    border: 1px solid #00AA00;
    border-radius: 4px;
    margin-top: 8px;
    background-color: #202020;
}

QGroupBox::title
{
    subcontrol-origin: margin;
    left: 8px;
    padding: 0 4px;
    color: #00FF00;
}

QPushButton
{
    background-color: #2A2A2A;
    border: 1px solid #00AA00;
    border-radius: 3px;
    min-height: 24px;
}

QPushButton:hover
{
    background-color: #00AA00;
    color: black;
}

QPushButton:pressed
{
    background-color: #008800;
}

QLineEdit
{
    background-color: #101010;
    border: 1px solid #00AA00;
    padding: 3px;
}
)");

```

```

QComboBox
{
    background-color: #101010;
    border: 1px solid #00AA00;
    padding: 3px;
}

QComboBox::drop-down
{
    border: none;
}

QTableWidget
{
    background-color: #101010;
    gridline-color: #00AA00;
    border: 1px solid #00AA00;
}

QHeaderView::section
{
    background-color: #202020;
    color: #00FF00;
    border: 1px solid #00AA00;
}

QScrollArea
{
    border: 1px solid #00AA00;
}

QScrollBar:vertical
{
    background: #202020;
    width: 12px;
}

QScrollBar::handle:vertical
{
    background: #00AA00;
}

QProgressBar
{
    border: 1px solid #00AA00;
    background-color: #101010;
    text-align: center;
}

QProgressBar::chunk
{
    background-color: #00AA00;
}

QMenuBar
{
    background-color: #202020;
}

QMenuBar::item:selected
{
    background-color: #00AA00;
    color: black;
}

QMenu
{
    background-color: #202020;
}

QMenu::item:selected
{
    background-color: #00AA00;
    color: black;
}
)";

ui->centralwidget->setStyleSheet(

```

```

        "background: transparent;"
    );

    setGraphicsImage(ui->graphicsView_21, "MPU6050.png");
    setGraphicsImage(ui->graphicsView_17, "gyneo6mv2.png");
    setGraphicsImage(ui->graphicsView_22, "NRF24L01.png");
    setGraphicsImage(ui->graphicsView_27, "drv8833.png");
    setGraphicsImage(ui->graphicsView_24, "motorDrone.png");
    setGraphicsImage(ui->graphicsView_23, "motor.png");

    magazine.append(Device("", "SPI", {1, 2, 3, 4}, "test_SPI_ACCEL", "SENSOR"));
    magazine.append(Device("", "I2C", {1, 2}, "test_I2C_ACCEL", "SENSOR"));
    magazine.append(Device("", "UART", {1, 2}, "test_UART_SIGNAL", "SIGNAL"));
    magazine.append(Device("", "PWM", {1}, "test_PWM_MOTOR", "MOTOR"));

    ui->targetSettings->setColumnCount(5);

    ui->targetSettings->setHorizontalHeaderLabels({
        "Index",
        "Type",
        "Protocol",
        "Information",
        "Pins"
    });
}

void MainWindow::onTelemetry(float ch1, float ch2, float ch3, float ch4,
                             float roll, float pitch, float yaw,
                             int battery, uint32_t counter)
{
    float rollDeg = roll * 180.0f;
    float pitchDeg = pitch * 180.0f;
    float yawDeg = yaw * 180.0f;

    updateDrone(rollDeg, pitchDeg, yawDeg);

    ui->label_3->setText(QString("ROLL: %1").arg(rollDeg));
    ui->label_2->setText(QString("PITCH: %1").arg(pitchDeg));
    ui->label->setText(QString("YAW: %1").arg(yawDeg));

    ui->ch1_bar->setValue((int)ch1);
    ui->ch2_bar->setValue((int)ch2);
    ui->ch3_bar->setValue((int)ch3);
    ui->ch4_bar->setValue((int)ch4);
}

MainWindow::~MainWindow()
{
    delete ui;
}

void MainWindow::on_connectButton_clicked()
{
    connection_mode++;

    if (connection_mode >= 3) connection_mode = 0;
    communicator.mode = connection_mode;
    qDebug() << "status changed to " << connection_mode;
}

void MainWindow::on_addSPISensor_clicked()
{
    int result = connected_target.AddDevice(
        magazine[0],
        connected_target.pinsSPI.mid(0, 3)
    );

    if (result == 0)
        addRow();
}

```

```

}

void MainWindow::addRow()
{
    int row = ui->targetSettings->rowCount();
    ui->targetSettings->insertRow(row);

    Device dev = connected_target.connected_devices.last();

    ui->targetSettings->setItem(row, 0,
                               new QTableWidgetItem(QString::number(row)));

    ui->targetSettings->setItem(row, 3,
                               new QTableWidgetItem(dev.information));

    ui->targetSettings->setItem(row, 1,
                               new QTableWidgetItem(dev.type));

    ui->targetSettings->setItem(row, 2,
                               new QTableWidgetItem(dev.protocol));

    QString pinsString;
    for (int pin : dev.pins)
        pinsString += QString::number(pin) + " ";

    ui->targetSettings->setItem(row, 4,
                               new QTableWidgetItem(pinsString.trimmed()));
}

void MainWindow::on_addI2Csensor_clicked()
{
    int result = connected_target.AddDevice(
        magazine[1],
        connected_target.pinsI2C.mid(0, 2)
    );

    if (result == 0)
        addRow();
}

void MainWindow::on_addUARTsignal_clicked()
{
    int result = connected_target.AddDevice(
        magazine[2],
        connected_target.pinsUART.mid(0, 2));

    if (result != 0)
        result = connected_target.AddDevice(
            magazine[2],
            connected_target.pinsUART.mid(2, 2));

    if (result == 0)
        addRow();
}

void MainWindow::on_addPWMmotor_clicked()
{
    int result = 1;
    int index = 0;
    while (result != 0 && index < connected_target.freepins.size())
    {
        result = connected_target.AddDevice(
            magazine[3],
            connected_target.pinsPWM.mid(index, 1));
        index++;
    }

    if (result == 0)
        addRow();
}

void MainWindow::on_editButton_clicked()
{

```

```

QString cmd = ui->lineEdit->text();

if (cmd == "devcom send")
{
    QString msg = buildQtStateString();
    communicator.Send(msg);

    qDebug() << "Sent config:" << msg;
    return;
}

if (cmd.startsWith("devcom delete "))
{
    QStringList parts = cmd.split(' ');

    if (parts.size() != 3)
    {
        qDebug() << "Wrong command";
        return;
    }

    bool ok;
    int index = parts[2].toInt(&ok);

    if (!ok)
    {
        qDebug() << "Wrong index";
        return;
    }

    int result = connected_target.DeleteDevice(index);

    if (result == 0)
    {
        ui->targetSettings->removeRow(index);

        for (int i = 0; i < ui->targetSettings->rowCount(); i++)
        {
            ui->targetSettings->setItem(
                i, 0,
                new QTableWidgetItem(QString::number(i))
            );
        }

        qDebug() << "Device deleted";
    }
    else
    {
        qDebug() << "Delete error";
    }

    return;
}

if (cmd.startsWith("devcom set pins "))
{
    QStringList parts = cmd.split(' ');

    if (parts.size() < 4)
    {
        qDebug() << "Wrong command format";
        return;
    }

    bool ok;
    int index = parts[3].toInt(&ok);

    if (!ok)
    {
        qDebug() << "Wrong index";
        return;
    }

    if (index < 0 || index >= connected_target.connected_devices.size())
    {
        qDebug() << "Index out of range";
    }
}

```

```

        return;
    }

    Device &dev = connected_target.connected_devices[index];

    int begin = cmd.indexOf('{');
    int end = cmd.indexOf('}');

    if (begin == -1 || end == -1 || end <= begin)
    {
        qDebug() << "Wrong pins format";
        return;
    }

    QString pinsText = cmd.mid(begin + 1, end - begin - 1);
    QStringList pinList = pinsText.split(' ', Qt::SkipEmptyParts);

    if (pinList.size() != dev.pins.size())
    {
        qDebug() << "Wrong pins count";
        return;
    }

    QVector<int> newPins;

    for (QString s : pinList)
    {
        bool pinOk;
        int pin = s.toInt(&pinOk);

        if (!pinOk)
        {
            qDebug() << "Wrong pin";
            return;
        }

        if (pin < 0 || pin > 39)
        {
            qDebug() << "Pin out of range";
            return;
        }

        newPins.push_back(pin);
    }

    for (int i = 0; i < connected_target.connected_devices.size(); i++)
    {
        if (i == index) continue;

        for (int used : connected_target.connected_devices[i].pins)
        {
            if (newPins.contains(used))
            {
                qDebug() << "Pin occupied";
                return;
            }
        }
    }

    dev.pins = newPins;

    QString pinsString;
    for (int p : newPins)
        pinsString += QString::number(p) + " ";

    ui->targetSettings->setItem(
        index, 4,
        new QTableWidgetItem(pinsString.trimmed())
    );

    qDebug() << "Pins updated";
    return;
}

qDebug() << "Unknown command";
}

```



## Додаток 3 Презентація

# Вбудована програмна система дистанційного керування безпілотними апаратами

Виконав: Студент 4 курсу  
КВ-21 Кукса К. В.

Керівник: Доцент каф.  
СПІСКС Морозов К. В.

## Актуальність теми

Актуальність роботи зумовлена зростаючою потребою у доступних та ефективних системах керування безпілотними літальними апаратами, що дозволяють збір телеметричних даних та моніторинг стану системи в реальному часі у різних сферах діяльності.

# Мета роботи

Мета роботи - розробка програмного комплексу для керування безпілотним літальним апаратом та збору телеметричних даних у реальному часі. Вона передбачає створення комунікації між пультом керування та безпілотним апаратом, обробку сигналів з джойстиків, налаштування і отримання даних з сенсорів та візуалізацію стану апарата в інтерфейсі.

3/20

## Аналіз існуючих рішень

Закриті системи - системи віддаленого або автономного контролю безпілотних апаратів, які працюють лише у екосистемі виробника, який надає мінімальний доступ до власних алгоритмів і протоколів. (Приклад – квадрокоптер DJI Mavic)



Відкриті системи - системи віддаленого або автономного контролю безпілотних апаратів, які не вимагають купівлі апаратної і програмної частини лише в одного виробника. Такі системи не користуються закритими протоколами передачі, можуть працювати з аналогами в інших виробників а доволі часто навіть мають відкритий для загального користування вихідний код. (Приклад: мікроконтролер MATEK+ прошивка Ardupilot)



4/20

## Порівняння запропонованої системи та існуючих рішень

|                                       | Закриті системи         | Відкриті системи       | Пропонована система |
|---------------------------------------|-------------------------|------------------------|---------------------|
| Приблизна ціна контролера             | Неможливо купити окремо | >50\$                  | <15\$               |
| Мінімальна ціна готової системи       | 500\$                   | 150\$                  | 30\$                |
| Можливість програмної модифікації     | Ні                      | Так                    | Так                 |
| Можливість апаратної модифікації      | Ні                      | Лише окремі компоненти | Так                 |
| Можливість зміни типу апарату         | Ні                      | Ні                     | Так                 |
| Вимоги до загальних знань з роботи БА | Низькі                  | Високі                 | Високі              |

Основна перевага запропонованої системи – відкритість і робота з мікроконтролерами загального значення

5/20

## Технічні вимоги до програмного забезпечення

- Наявність в системі трьох елементів: польотного контролера, наземної станції (пульту керування) і графічного додатку;
- Можливість підключитись до наземної станції і польотного контролера через Wi-Fi за допомогою додатку;
- Можливість редагувати налаштування польотного контролера через додаток;
- Можливість роботи польотного контролера відповідно до налаштувань: аналізувати сенсори і керувати двигунами
- Можливість керувати польотним контролером за допомогою наземної станції;
- Можливість підключити додаток до наземної станції з метою отримання телеметрії;

6/20

## Вибір мікроконтролера

Для розробки програмного забезпечення для безпілотного апарату було обрано модуль ESP-32-WROOM через його високі обчислювальні потужності, низьку ціну, вбудований WI-FI-модуль, а також 38 цифрових пінів, серед яких:

- 18 пінів з можливістю аналогово-цифрового перетворювача
- 3 SPI інтерфейси
- 3 UART інтерфейси
- 2 I2C інтерфейси
- 16 пінів виводу широко-імпульсної модуляції
- 2 піни з можливістю цифро-аналогового перетворювача



7/20

## Вибір програмного середовища для створення прошивки

Для розробки польотного контролера і наземної станції будемо використовувати Arduino IDE. Дане середовище має велику кількість переваг у порівнянні з іншими IDE, а саме:

- Сумісність з стандартними функціями Arduino
- Можливість відлагодження за допомогою Serial Monitor і Serial Plotter
- Зручний текстовий редактор, що допомагає знаходити помилки на етапі написання коду
- Велика кількість бібліотек, доступних для завантаження
- Простий інтерфейс компіляції і завантаження прошивки
- Вбудовані в прошивку можливості ESP32 за допомогою <Arduino.h>

8/20

# Вибір програмного середовища для створення додатку

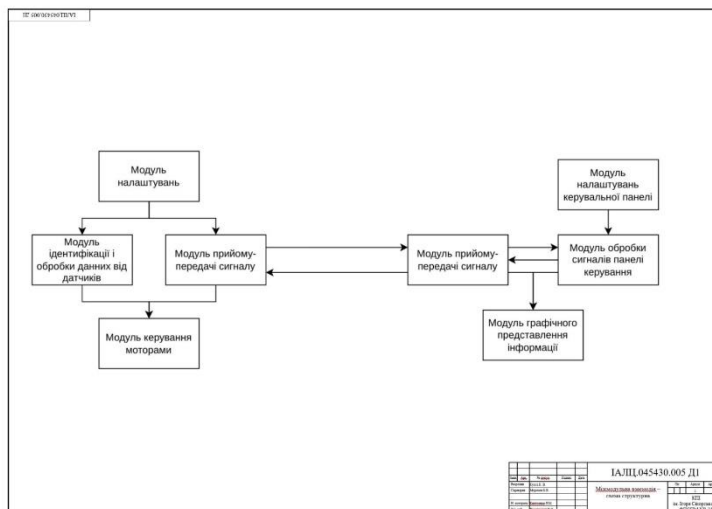
Qt – це кросплатформний фреймворк для створення додатків, що надає можливості створення додатків майже в будь-якому напрямку

Qt надає такі можливості як: розробка графічних інтерфейсів, робота з мережею, мультимедіа, базами даних, 2D і 3D графікою, веб, тестуванням, навіть навігацією і різними інструментами вводу.

Кросплатформеність проєкту дозволяє створювати версії під різні операційні системи не змінюючи коду продукту - це вирішується на етапі компіляції.

9/20

## Міжмодульна взаємодія



Під час виконання дипломного проєкту була розроблена схема міжмодульної взаємодії. Надалі вона використовувалася як основа для подальшої розробки системи.

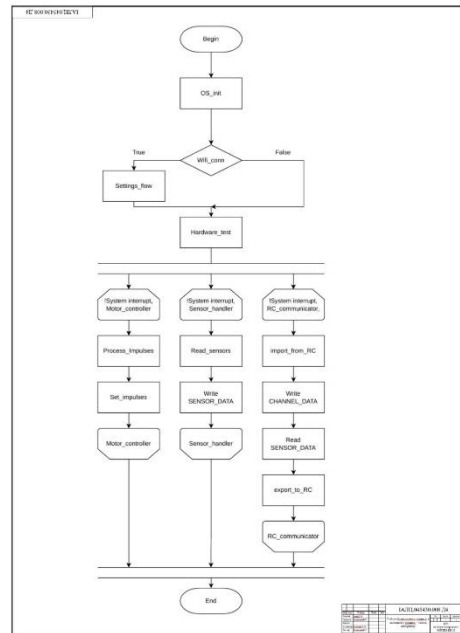
10/20

Під час виконання дипломного проекту була розроблена схема алгоритму прийому налаштувань безпілотного апарату, цей алгоритм впроваджено зі сторони польотного контролера при початку роботи

Під час виконання дипломного проєкту була розроблена схема алгоритму роботи наземної станції. Тут описаний процес менеджменту одночасного підключення як до безпілотного апарату, так і до додатку.

## Робота безпілотного апарату в активному режимі

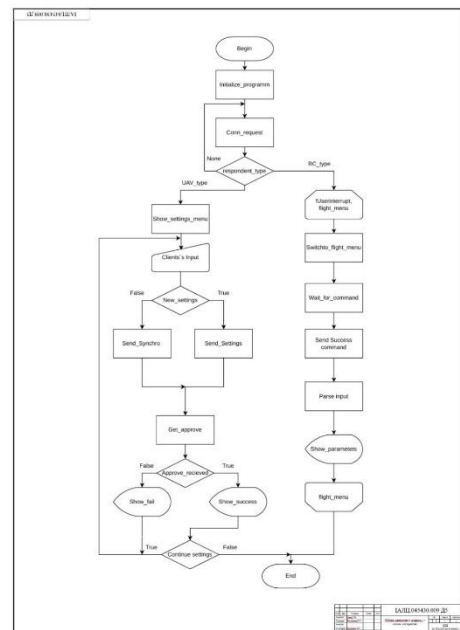
Під час виконання дипломного проєкту була розроблена схема алгоритму роботи безпілотного апарату в активному режимі. Він запускається зі сторони польотного контролера безпілотного апарату після завершення алгоритму прийому налаштувань.



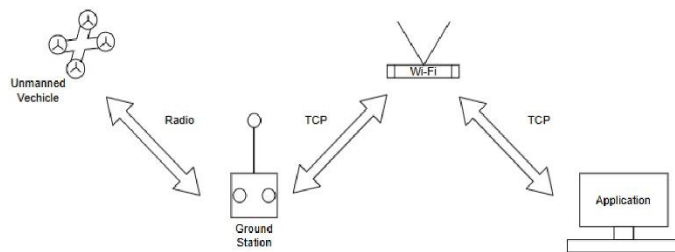
13/20

## Обмін даними в додатку

Під час виконання дипломного проєкту була розроблена схема алгоритму обміну даними в додатку, яка, окрім алгоритму комунікації з іншими модулями, показує також етапи оновлення інтерфейсу.



14/20



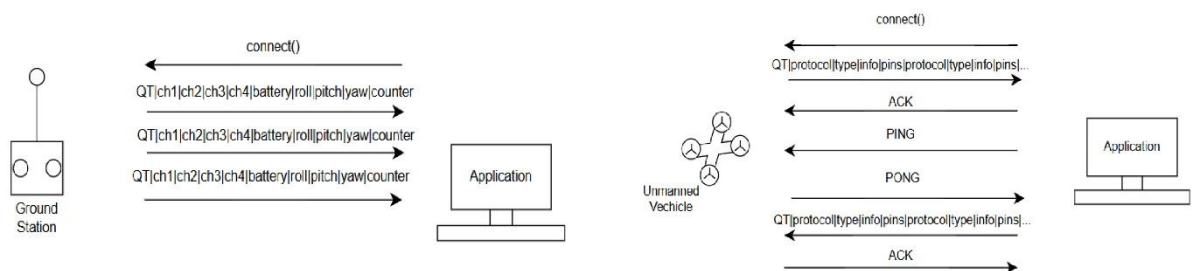
| Безпілотний апарат | Наземна Станція | Додаток   | Режим                              |
|--------------------|-----------------|-----------|------------------------------------|
| Ввімкнено          | Вимкнено        | Ввімкнено | Налаштування БА                    |
| Вимкнено           | Ввімкнено       | Ввімкнено | Налагодження НС                    |
| Ввімкнено          | Ввімкнено       | Вимкнено  | Робочий режим                      |
| Ввімкнено          | Ввімкнено       | Ввімкнено | Робочий режим з графічним аналізом |

Існує можливість працювати в різних режимах, де можуть бути не задіяні певні модулі

- Налаштування БА – комунікація між безпілотним апаратом і додатком за допомогою мережі Wi-Fi з можливістю в додатку зчитувати налаштування БА, редагувати їх або надсилати нові налаштування .
- Налагодження НС – комунікація між наземною станцією і додатком за допомогою мережі Wi-Fi, де додаток показує в графічному вигляді всі користувацькі дії на наземній станції.
- Робочий режим – комунікація між безпілотним апаратом і наземною станцією згідно обраного в налаштуваннях протоколу.
- Робочий режим з графічним аналізом – так само, що і робочий режим, але тепер наземна станція також надсилає додатку інформацію як від безпілотного апарату, так і від себе, додаток це аналізує і виводить графічну інформацію.

15/20

## Протоколи комунікації додатку



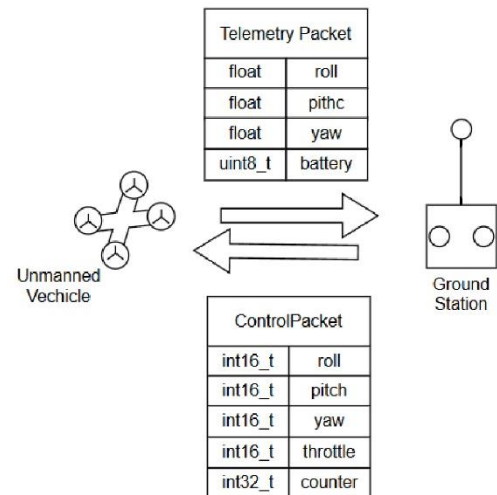
Для комунікації додатку з пристроями було розроблено протокол поверх TCP. У Безпілотного апарату і наземної станції вони відрізняються.

16/20

# Протокол комунікації між БА і НС

Для комунікації між мікроконтролером використовуються спеціальні структури для передачі: ControlPacket, TelemetryPacket

Telemetry Packet також може містити додаткові байти як значення підключені модулів



17/20

## Інтерфейс додатку



Було розроблено односторінковий простий додаток, який містить як переважно текстову, так і графічну інформацію. Жовтими номерами позначені окремі функціональні елементи додатку.

18/20

## Область застосування і подальші можливості

Цей проект завдяки низькій ціні може стати новим рішенням для виробників дронів, які намагаються зменшити витрати або намагаються розробляти міні-дрони, а також цей проект підходить для початківців як у сфері електроніки, так і у сфері безпілотних апаратів через легкість використання.

У подальшій роботі систему можна значно розширити, зокрема інтегрувати прошивку на інші мікроконтролери, додати різні режими роботи, розширити можливість графічного інтерфейсу, збільшувати кількість підтримуваних модулів, додати підтримку нових типів безпілотних апаратів

19/20

## Висновки

Розроблено і протестовано вбудовану систему дистанційного керування безпілотним апаратом, що забезпечує стабільне управління та передачу телеметричних даних у реальному часі. Впроваджено протоколи обміну даними між дроном і наземною станцією.

Система демонструє ефективну інтеграцію доступної апаратної платформи та програмної логіки, дозволяє легко додавати нові сенсори і модулі, і закладає основу для подальшого розвитку та розширення функціоналу вбудованих систем керування безпілотними апаратами.

20/20