Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
(повна назва)

Кафедра Автоматизації проектування обчислювальної техніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти перший(бакалаврський)

Дистанційно керована мобільна платформа з функцією відеомоніторингу

(тема)

Виконав: здобувач 4 року навчання,
групи КІУКІ-21-8
Машталяр С.В
(прізвище, ініціали)
Спеціальність 123 «Компьютерна інженерія»
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма
Компьютерна інженерія
(повна назва освітньої програми)
Керівник доц. Рожнова Т.Г.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри АПОТ _____ Чумаченко С.В
(підпис) (прізвище, ініціали)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління

Кафедра Автоматизації проектування обчислювальної техніки

Рівень вищої освіти перший(бакалаврський)

Спеціальність 123 «Комп’ютерна інженерія»
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Комп’ютерна інженерія
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)

“ 02 ” 05 2025р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Машталяру Сергію Вікторовичу
(прізвище, ім’я, по батькові)

1. Тема роботи Дистанційно керована мобільна платформа з функцією відеомоніторингу

затверджена наказом по університету від “ 21 ” 05 2025 р. № 403Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 12.06.2025

3. Вхідні дані до роботи

Характеристики плати ESP32–CAM
Характеристики драйверу L298N
Схема плати ESP32–CAM
Схема драйверу L298N
Мова програмування C++
Середовища розробки Visual Studio та Arduino IDE

4. Перелік питань, що потрібно опрацювати у роботі

Аналіз предметної області, засобів розробки і постановка задачі;
Аналіз та вибір компонентів, для реалізації поставленої задачі;
Розробка програмного забезпечення та дистанційно керованої платформи;
Тестування програмного забезпечення та платформи;
Аналіз недоліків розробленої платформи та програмного забезпечення до
Аналіз методів покращення працездатності платформи.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій ((п.5 включається до завдання за рішенням випускової кафедри)) 15 слайдів у форматі .ppt

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Розгляд тенденцій розвитку FPV-систем	05.05.25 - 15.05.25	+
2	Розгляд тенденцій наземних FPV-платформ	15.05.25 - 19.05.25	+
3	Вибір компонентів та комплектуючих	19.05.25– 21.05.25	+
4	Моделювання схеми наземної FPV-платформи	22.05.25– 24.05.25	+
5	Написання програмного забезпечення	25.05.25– 27.05.25	+
6	Тестування програмного забезпечення	28.05.25– 01.06.25	+
7	Збірка платформи	01.06.25– 05.06.25	+
8	Тестування роботи платформи	06.06.25– 09.06.25	+
9	Аналіз проблем та можливостей покращення роботи платформи	10.06.25– 11.06.25	+
10	Подання роботи для захисту в ЕК	12.06.25 -30.06.25	-

Дата видачі завдання 05.05 2025 р.

Здобувач _____ (підпис)

Керівник роботи _____ (підпис) доц. Рожнова Т.Г. (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 44 с., 19 рис., 3 дод., 10 джерел.

МІКРОКОНТРОЛЕРИ, ІНТЕРНЕТ, ДИСТАНЦІЙНЕ КЕРУВАННЯ, WI–FI, WLAN, WI–FI МОДУЛЬ.

Метою кваліфікаційної роботи є розробка, моделювання та проєктування дистанційно керованої мобільної платформи з можливостю відеомоніторингу.

У ході виконання кваліфікаційної роботи було змодельовано та сконструйовано платформу, керувати якою можна з будь-якого пристрою, підключеного до тієї ж мережі WI-FI, що і сама платформа. За діями платформи можливо спостерігати через відеокамеру з інтерфейсом, за допомогою якого і відбувається процес керування.

ABSTRACT

Master’s thesis 44 pages, 19 figures, 3 appendices, 10 sources.

INTERNET, ROUTER, WI-FI, WIRELESS NETWORK, WLAN, MICROCONTROLLER.

The major goal of this thesis is the development, modeling and design of a remote-controlled mobile platform with video monitoring capabilities.

In order to to achieve the desired result, the development and assembly of the platform was carried out, as well as testing its performance and the performance of all necessary components.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗН АЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

IP-адреса (Internet Protocol address) – це унікальний числовий ідентифікатор, присвоєний кожному пристрою, підключеному до комп'ютерної мережі, що використовує протокол Інтернет (IP) для зв'язку.

HTTP (Hypertext Transfer Protocol) – це протокол передачі гіпертексту, який лежить в основі обміну даними у World Wide Web.

PSRAM (Pseudo-Static Random-Access Memory) – це тип динамічної пам'яті (DRAM) з вбудованою схемою регенерації

UART (Universal Asynchronous Receiver/Transmitter) – це простий асинхронний протокол послідовної передачі даних між двома пристроями.

Wi-Fi – технологія бездротового зв’язку, що дозволяє передавати дані по радіоканалу в локальній мережі або підключатися до інтернету (англ. Wireless Fidelity).

ВСТУП

Розвиток сучасних технологій сприяє появі нових рішень у сфері дистанційного керування та відеомоніторингу. Особливої уваги заслуговують мобільні платформи, які можуть використовуватися для спостереження, охорони територій, дослідження важкодоступних місць, а також у розважальних і навчальних цілях. В реаліях сучасності, попит на такі пристрої зростає й у військовій сфері для оборони нашої держави. Дистанційно керовані мобільні платформи широко використовуються у рятувальних операціях, розмінуванні, доставці припасів, медикаментів та провіанту. Їх важливими характеристиками є надійність зв’язку, швидкість обробки команд та можливість автономної роботи.

Реалізація подібної системи вимагає ретельного підходу до інтеграції апаратних та програмних компонентів. Крім базових функцій відеомоніторин гу та керування, важливо також забезпечити захист даних та безпеку зв'язку, особливо при використанні платформи у військових або критично важливих застосуваннях. Додатковою перевагою буде розробка дружнього користувацького інтерфейсу для мобільного пристрою, що зробить керування платформою інтуїтивно зрозумілим та доступним для широкого кола користувачів.

У даній роботі буде запропоновано оптимальне та найпростіше рішення, що забезпечує стабільну передачу відео та точне керування платформою через будь-який пристрій, який має доступ до мережі WI-FI або WLAN.

Поєднання бездротового зв’язку, мікроконтролерних систем та FPV-технологій дає змогу створювати ефективні й зручні у використанні пристрої, що забезпечують передачу відео в реальному часі та дистанційне управління.

Одним із найбільш універсальних підходів до реалізації таких платформ є використання мікроконтролера ESP32-CAM, що забезпечує

передачу відео через Wi-Fi, а також драйвера L298N, який дозволяє керувати електродвигунами. Це дає змогу створити мобільну платформу з FPV-камерою, якою можна керувати за допомогою смартфона або іншого пристрою через бездротове з’єднання.

Метою даної роботи є розробка дистанційно керованої мобільної платформи з функцією відеомоніторингу, що передає зображення в реальному часі та керується з мобільного пристрою. Для цього необхідно виконати такі завдання:

– проаналізувати ринок доступних моделей;
– обрати необхідні компоненти;
– розробити схеми підключення компонентів;
– розробити апаратну частину пристрою;
– розробити програмне забезпечення для керування платформою та обробки відеопотоку;
– протестувати систему в різних режимах, що підтвердить її працездатність.

1 ПОСТАНОВКА ЗАВДАННЯ НА ПРОЄКТУВАННЯ

Сучасні технології відеоспостереження та дистанційного керування активно впроваджуються у різні сфери діяльності, зокрема в моніторинг об’єктів, рятувальні операції та дослідження важкодоступних територій, охорону та військову сферу [1]. Ефективне застосування таких систем вимагає інтеграції апаратних засобів із засобами бездротового зв’язку, що забезпечує можливість передачі відеопотоку та керування мобільними платформами у реальному часі [6]. В Україні розвивається розробка FPV-систем, що забезпечують дистанційне керування безпілотними літальними і наземними апаратами. Ці технології знаходять застосування як у цивільній сфері, так й у військовій.

Мета роботи – розробити просту FPV-платформу і показати на прикладі, що розробка та моделювання FPV-систем є простим і кропітливим процесом, але вкрай необхідним тому тема робота і тема є актуальними.

Вимогами для дистанційно керованої мобільної платформи з функцією відеомоніторингу, що забезпечують його ефективність, стабільну роботу та зручність експлуатації можуть стати такі [7]:

1) платформа повинна підтримувати дистанційне керування через бездротовий зв’язок (Wi-Fi або WLAN);

2) система має забезпечувати передачу відео в режимі реального часу з мінімальними затримками;

3) мобільна платформа повинна мати можливість маневрування;

4) керування платформою повинно здійснюватися через мобільний пристрій або комп’ютер із відповідним програмним забезпеченням;

5) система повинна мати простий та інтуїтивно зрозумілий інтерфейс керування;

6) простота розробки та доступність компонентів, що використано для реалізації.

2 РОЗРОБКА АПАРАТНОЇ ЧАСТИНИ

2.1 Модуль ESP32-CAM

ESP32-CAM – це компактний модуль на базі мікроконтролера ESP32, який поєднує в собі потужні обчислювальні можливості, бездротовий зв’язок та вбудовану камеру. Він широко використовується для реалізації систем відеоспостереження, дистанційного моніторингу, розпізнавання об’єктів та IoT-рішень [2].

Функціональні можливості ESP32-CAM включають передачу відео в реальному часі, обробку зображень, запис даних на карту пам’яті та інтеграцію з іншими електронними компонентами через інтерфейси UART, I2C, GPIO та PWM. Пристрій може працювати у складі IoT-систем завдяки сумісності з платформами Blynk, MQTT та Firebase.

Прошивка модуля здійснюється за допомогою зовнішнього USB-UART-програматора, оскільки ESP32-CAM не має вбудованого USB-інтерфейсу. Програмування виконується через середовища Arduino IDE, Espressif IDF або PlatformIO, використовуючи відповідні бібліотеки для керування камерою, передачею відео та роботи з бездротовими модулями.

Модуль застосовується у сфері відеоспостереження, розумного дому, дистанційного керування, FPV-систем, промислового моніторингу та автоматизації. Завдяки своїй функціональності та доступності ESP32-CAM є ефективним рішенням для створення бездротових систем відеоспостереження та інших проєктів, що потребують передачі зображень у реальному часі.

Модуль оснащений двоядерним процесором Tensilica LX6 з тактовою частотою до 240МГц, має 520КБ вбудованої оперативної пам’яті та підтримує карти microSD до 4ГБ. Завдяки вбудованим модулям Wi-Fi та Bluetooth ESP32-CAM забезпечує бездротовий обмін даними. Камера

OV2640, яка входить до комплекту, підтримує зйомку у роздільній здатності до 1600×1200 пікселів. Живлення здійснюється в діапазоні 3,3–5В, а середнє енергоспоживання становить приблизно 160мА. Пристрій підтримує енергозберігаючий режим Deep Sleep, що дозволяє значно зменшити споживання енергії. Компонування та розміщення виходів підключення, які позначені на схемі модуля (рисунок 1.1), робить процес роботи з ним легким та зручним.

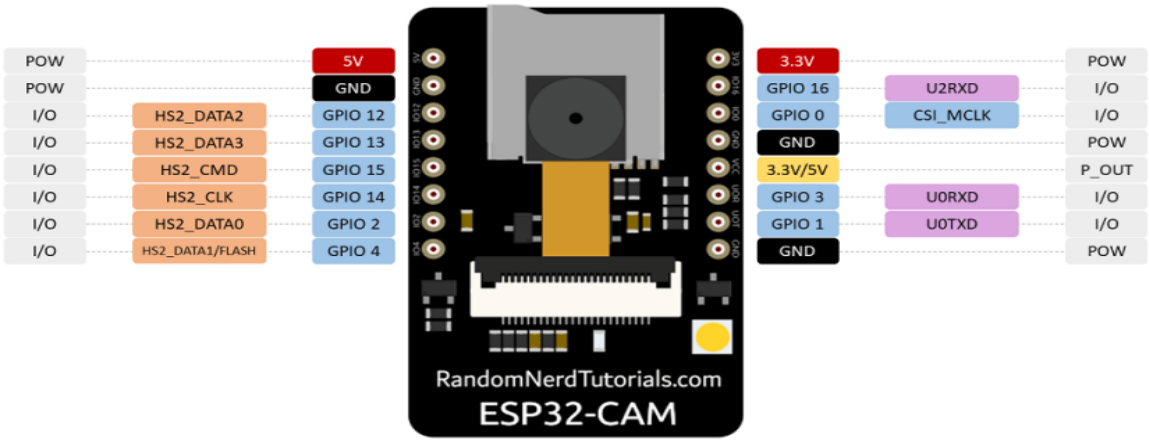

Рисунок 2.1 – Схема підключення мікроконтролера

Перевагами модуля ESP32-CAM є компактні розміри та низьке енергоспоживання, що робить його зручним для використання в автономних і мобільних пристроях. Завдяки вбудованій підтримці Wi-Fi та Bluetooth він може передавати відео та дані без необхідності додаткових модулів зв’язку. Підтримка карт пам’яті microSD дозволяє зберігати зображення та відео безпосередньо на пристрої. Вбудована камера OV2640 забезпечує достатню якість зображення для більшості задач відеоспостереження та автоматизації. Модуль має широкий набір інтерфейсів, що дозволяє інтегрувати його з датчиками, сервоприводами та іншими пристроями. Доступна ціна робить ESP32-CAM популярним вибором для проєктів у сфері IoT, відеомоніторингу та дистанційного керування [8].

А серед недоліків можна виділити відсутність вбудованого USB-інтерфейсу для програмування, що вимагає використання додаткового USB-

UART-програматора. Вбудована Wi-Fi-антена має обмежену ефективність, тому для покращення якості зв’язку може знадобитися зовнішня антена. Процесор ESP32, хоч і потужний для багатьох задач, має обмежені можливості для обробки відеопотоків високої роздільної здатності, що може призводити до затримок або зниження частоти кадрів. Модуль може перегріватися при тривалому використанні у режимі відеотрансляції, що впливає на стабільність роботи.

Так, відношення переваг і недоліків, а також доступність, якість, простота взаємодії та низка ціна робить цей модуль одним із кращих виборів для проєктів створення FPV-систем.

2.2 Драйвер L298N

Драйвер L298N – це двоканальний модуль керування двигунами, який використовується для управління напрямком і швидкістю обертання електродвигунів. Він побудований на основі мікросхеми L298, що є потужним H-мостовим драйвером, здатним працювати з постійними двигунами або кроковими двигунами.

Основне призначення L298N – забезпечення незалежного керування двома двигунами за допомогою сигналів керування від мікроконтролера, таких як Arduino, ESP32 або Raspberry Pi. Модуль підтримує вхідну напругу живлення до 35 В та максимальний струм навантаження до 2 А на канал.

Вбудований стабілізатор напруги дозволяє використовувати драйвер для живлення мікроконтролера, якщо напруга живлення двигунів перевищує 7 В. Драйвер L298N має виходи для регулювання швидкості двигунів через широтно-імпульсну модуляцію (ШІМ), що дозволяє плавно змінювати швидкість обертання моторів.

Завдяки надійності використання драйвер L298N широко застосовується у робототехніці, при створенні дистанційно керованих платформ, в автоматизованих механізмах і навчальних проєктах. А простота

конструкції драйвера забезпечує безперешкодний доступ до усіх частин модуля, які можна побачити на схемі драйвера (рисунок 2.2).

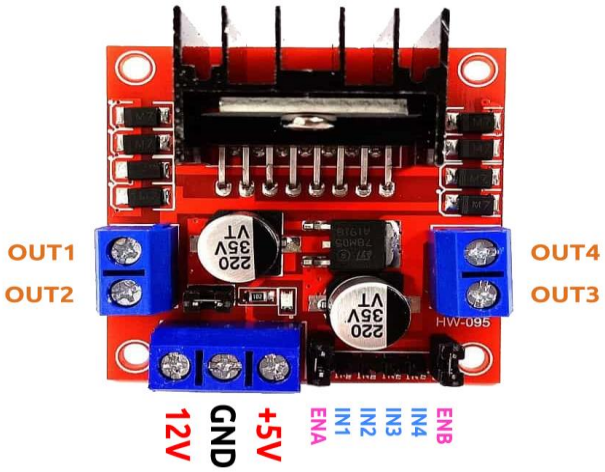

Рисунок 2.2 – Схема підключення драйвера L298N

Серед переваг драйвера L298N виділяють вбудований стабілізатор напруги, який може живити мікроконтролер; наявність підтримки широтно-імпульсної модуляції для регулювання швидкості; просте підключення до Arduino, ESP32 та інших мікроконтролерів; захист від зворотних струмів індуктивності; радіатор охоложення; наявністоь великої кількості навчальних матеріалів та інстукцій від виробника; низька ціна та доступність на ринку компонентів.

Недоліками ж є низький ККД через високі теплові втрати. Максимальний струм 2 А може бути недостатнім для потужних двигунів. Значне нагрівання при тривалому навантаженні потребує радіатора, який видно на схемі драйвера. Великі габарити у порівнянні із сучасними MOSFET-драйверами.

Недоліки драйвера є досить вагомими для його використання у великих системах. Але для даного проєкту його характеристик буде достатньо, тому його недоліки не будуть мати великого впливу на працездатність платформи.

2.3 Програматор FT232RL

Програматор FT232RL – це адаптер для перетворення інтерфейсу USB на послідовний інтерфейс UART, що використовується для програмування та налагодження мікроконтролерів, модулів ESP32-CAM, ESP8266, Arduino та інших пристроїв.

Мікросхема FT232RL забезпечує надійну передачу даних через USB-порт комп'ютера, підтримує напругу 3,3 В і 5 В, що робить її універсальною для роботи з різними мікроконтролерами. Завдяки автоматичному визначенню швидкості обміну даними та вбудованому захисту від перевантажень програматор стабільно працює в широкому діапазоні застосувань.

Пристрій компактний, енергоефективний і не потребує складного налаштування. Сучасні операційні системи зазвичай розпізнають його без необхідності встановлення додаткових драйверів. Програматор FT232RL широко використовується для оновлення прошивки, налагодження програмного коду та взаємодії мікроконтролерів із ПК.

Перевагою цього програматора є те, що він простий у використанні, забезпечує стабільне перетворення USB-UART [5]. Підтримує широкий діапазон напруги (3,3 В і 5 В), сумісний із більшістю мікроконтролерів, включаючи ESP32-CAM. Не потребує додаткових драйверів для сучасних операційних систем. У нього компактний розмір та компоновка, які можна побачити на схемі (рисунок 2.3), низьке енергоспоживання, вбудований захист від перевантажень. Також цей комопнент є дуже популярним, тому користується попитом і є доступним. Також для версій Windows 7 і вичше не потребує встановлення додаткових драйверів.

Недоліком можна вважати необхідність встановлення драйверів на старих версіях Windows. Обмежена швидкість передачі даних у порівнянні з більш сучасними програматорами. Відсутність апаратного кнопкового скидання для деяких мікроконтролерів, що потребує ручного

перезавантаження. Також із за популярності на ринку модулів пристні підроблені моделі або моделі низької якості.

Зважаючи на все вищесказане, програматор FT232RL виділяється як надійне та економічне рішення для більшості проєктів. Його перевірена часом архітектура та широка підтримка спільнотою розробників забезпечують впевненість у стабільній роботі, а постійний доступ на ринку робить його ідеальним вибором як для початківців, так і для досвідчених інженерів.

Через те, що більшість персональних комп'ютерів та ноутбуків використовують 10-ту версію операційної системи Windows або вище, недолік у вигляді необхідності встановлення драйверу відпадає, що спрощує роботу з програматором. А його сумісність з модулем ESP32-CAM, простота підключення та компактність робить його чудовим вибором для використанні в цьому проєкті.

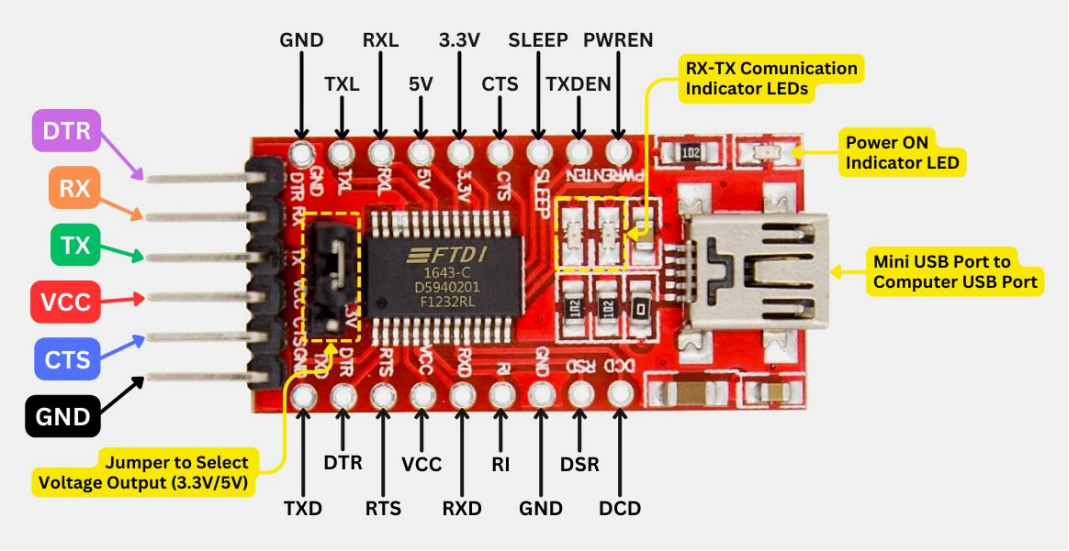

Рисунок 2.3 – Схема побудови та підключення FT232RL

3 НАПИСАННЯ ТА ЗАВАНТАЖЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ НА МОДУЛІ

3.1 Середовище розробки Arduino IDE

Arduino IDE (Integrated Development Environment) – це офіційне середовище розробки для програмування мікроконтролерів сімейства Arduino [3]. Воно дозволяє писати, компілювати та завантажувати код на мікроконтролери, а також виконувати відлагодження програм.

Редактор коду у Arduino IDE підтримує мову програмування C/C++ із вбудованими бібліотеками Arduino. Він має підсвічування синтаксису, базове автодоповнення та можливість роботи із вкладеними файлами у вигляді вкладок.

Компіляція та завантаження коду виконується автоматично після натискання кнопки "Завантажити". IDE спочатку перетворює вихідний код на машинний код, а потім передає його у пам’ять мікроконтролера через USB або інший інтерфейс зв’язку.

Монітор порту та графічний аналізатор (Serial Monitor & Serial Plotter) дозволяють взаємодіяти з платою через послідовний порт UART. Використовуючи монітор порту, можна переглядати вивід даних від мікроконтролера, а за допомогою графічного аналізатора – будувати графіки в реальному часі.

Менеджер бібліотек та плат дає змогу легко додавати нові бібліотеки для роботи з периферією. Наприклад, з Wi-Fi-модулями, датчиками чи дисплеями. Також можна встановлювати підтримку різних плат, зокрема ESP8266, ESP32, STM32 та інших.

Arduino IDE є кросплатформним середовищем, яке працює на Windows, macOS та Linux.

Існує дві основні версії Arduino IDE. Arduino IDE 1.x є класичною та

стабільною, має базовий функціонал. Arduino IDE 2.x є оновленою версією з покращеним інтерфейсом, розширеним автодоповненням, вбудованим відлагоджувачем і підтримкою багатопоточності.

Серед альтернатив Arduino IDE можна виділити PlatformIO, яка має розширену підтримку різних плат та розширений функціонал. Також популярним є Visual Studio Code з Arduino Plugin, який забезпечує зручне автодоповнення та розширені можливості налагодження. Для мікроконтролерів Microchip можна використовувати MPLAB X.

Arduino IDE є простим і потужним інструментом для роботи з мікроконтролерами, що найкраще підходить під вимоги даного проєкту. За допомогою цього середовища буде розроблено необхідне програмне забезпечення та завантажено на плату ESP32-CAM.

3.2 Розробка драйвера для модуля ESP32-CAM

Для керування FPV-платформою на базі ESP32-CAM необхідно розробити програмне забезпечення для самого модуля, яке виконує кілька основних завдань. По-перше, воно забезпечує роботу камери та потокову передачу відео через Wi-Fi. По-друге, воно реалізує керування рухом машини за допомогою GPIO-виходів, які підключені до моторного драйвера.

Програмне забеспечення написане мовою C++ з використанням середовища Arduino IDE та бібліотек ESP32 Camera і Wi-Fi. Код виконує ініціалізацію камери, налаштовує з’єднання з мережею Wi-Fi і запускає вебсервер, який дозволяє переглядати відео в режимі реального часу. Також передбачена можливість керування моторчиками через відповідні GPIO-піни, що забезпечує рух уперед, назад і повороти.

Бібліотека esp_camera.h надає функції для роботи з камерою, підключеною до мікроконтролера ESP32. Вона дозволяє налаштовувати параметри камери, отримувати зображення у різних форматах і передавати їх через мережу.

Бібліотека WiFi.h дозволяє мікроконтролеру ESP32 підключатися до мереж Wi-Fi, створювати точки доступу та взаємодіяти з іншими пристроями через IP-з'єднання.

Ці бібліотеки не є вбудованими в C++, тому їх треба підключати окремо – так, як показано на рисунку 3.1.

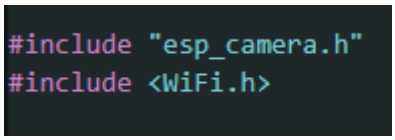

Рисунок 3.1 – Підключення бібліотек "esp_camera.h" та "WiFi.h"

Разом вони дозволяють створити відеострімінг через вебінтерфейс і реалізувати керування FPV-машинкою.

Для коректної взаємодії камери модулю ESP32-CAM та програмного забезпечення для нього в коді програми необхідно визначити модель камери. Найбільш поширеною моделлю є AI_THINKER, яка і використовується в проєкті. Модель камери потрібно самостійно визначити в коді програми прописавши наступну команду #define CAMERA_MODEL_AI_THINKER.

Після визначення моделі камери потрібно встановити коректні налаштування для правильної та стабільної передачі відеопотоку. Так, одним із налаштувань є встановлення GPIO-пінів, відповідних до моделі встановленої камери, які наведено в лістингу 3.1.

Лістинг 3.1 – Встановлення GPIO-пінів

```
#if defined(CAMERA_MODEL_WROVER_KIT)
#define PWDN_GPIO_NUM    -1
#define RESET_GPIO_NUM   -1
#define XCLK_GPIO_NUM    21
#define SIOD_GPIO_NUM    26
#define SIOC_GPIO_NUM    27

#define Y9_GPIO_NUM      35
#define Y8_GPIO_NUM      34
#define Y7_GPIO_NUM      39
#define Y6_GPIO_NUM      36
#define Y5_GPIO_NUM      19
```

```
#define Y4_GPIO_NUM      18
#define Y3_GPIO_NUM       5
#define Y2_GPIO_NUM       4
#define VSYNC_GPIO_NUM   25
#define HREF_GPIO_NUM    23
#define PCLK_GPIO_NUM    22

#elif defined(CAMERA_MODEL_AI_THINKER)
#define PWDN_GPIO_NUM     32
#define RESET_GPIO_NUM    -1
#define XCLK_GPIO_NUM      0
#define SIOD_GPIO_NUM     26
#define SIOC_GPIO_NUM     27

#define Y9_GPIO_NUM       35
#define Y8_GPIO_NUM       34
#define Y7_GPIO_NUM       39
#define Y6_GPIO_NUM       36
#define Y5_GPIO_NUM       21
#define Y4_GPIO_NUM       19
#define Y3_GPIO_NUM       18
#define Y2_GPIO_NUM        5
#define VSYNC_GPIO_NUM    25
#define HREF_GPIO_NUM     23
#define PCLK_GPIO_NUM     22
```

А для виводу відлагоджувальної інформації створена функція setup(), яка зображена на рисунку 3.2.

```
void setup() {
  Serial.begin(115200);
  Serial.setDebugOutput(true);
  Serial.println();
```

Рисунок 3.2 – Функція setup()

Для передачі зображення та отримання сигналів керування модуль повинен мати з’єднання з пристроєм керування. В нашому випадку сигнали керування та відеопотік повинні передаватися через Wi-Fi-мережу. Це зумовлено тим, що модуль ESP32-CAM на основі плати ESP32 має вбудований Wi-Fi-адаптер.

Для отримання сигналів та передачі відеозображення модуль ESP32-CAM та пристрій керування повинні бути під’єднані до однієї мережі Wi-Fi,

що дозволяє проводити обмін сигналами через вебсервер.

Для стабільності передачі сигналів та захисту передачі відеопотоку з'єднання буде з Wi-Fi-мережею, захищену паролем. Для підключення модулю ESP32-CAM до захищеної Wi-Fi-мережі треба прописати в коді програми налаштування мережі, а саме: назву мережі та пароль. Ці налаштування зберігаються у вигляді змінної з типом даних char, де в лапках потрібно вписати назву та пароль мережі, до якої потрібно під'єднатися.

Також для коректної роботи камери потрібно зберегти певні налаштування, такі як: GPIO-піни, які використовуються для підключення камери, частота тактового генератора (20 МГц), формат зображення (JPEG). Ці налаштування зберігаються у структурі config. Після заповнення структури config також потрібно перевірити, чи має модуль ESP32-CAM додаткову пам'ять (PSRAM). Наявність додаткової пам'яті дозволить встановити вищу якість зображення. Заповнення структури config та перевірка наявності додаткової пам'яті відбувається в окремій частині коду (лістинг 3.2).

Лістинг 3.2 – Структура config

```
camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sscb_sda = SIOD_GPIO_NUM;
  config.pin_sscb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
  config.xclk_freq_hz = 20000000;
  config.pixel_format = PIXFORMAT_JPEG;
```

```
if(psramFound()){
  config.frame_size = FRAMESIZE_UXGA;
  config.jpeg_quality = 10;
  config.fb_count = 2;
} else {
  config.frame_size = FRAMESIZE_SVGA;
  config.jpeg_quality = 12;
  config.fb_count = 1;
}
```

Після збереження налаштувань камери можна до неї підключатися. Процес підключення, який зображено на лістингу 3.3, виконується шляхом ініціалізації підключення до камери та виклику необхідного методу. Для цього використовуємо один з методів підключеної бібліотеки “Wi-Fi.h”, в який передаємо збережені налаштування мережі. Після ініціалізації підключення до мережі програма входить в цикл, в якому перевіряється підключення. Якщо підключення пройшло успішно, програма виводить відповідне повідомлення та переходить до наступного кроку, а якщо ні – програма показує, що підключення ще триває. Наступним кроком програми є запуск вебсерверу. Сервер дозволяє отримувати доступ до відео з камери через браузер за локальною IP-адресою ESP32, отримання та вивід якої є наступним кроком програми.

Лістинг 3.3 - Ініціалізація підключення до камери, старт роботи вебсерверу та обробка IP-адреси

```
esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK) {
           Serial.printf("Camera init failed with error
  0x%x", err);
    return;
  }

  sensor_t * s = esp_camera_sensor_get();
  s->set_framesize(s, FRAMESIZE_CIF);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
```

```
  }
  Serial.println("");
  Serial.println("WiFi connected");

  startCameraServer();

  Serial.print("Camera Ready! Use 'http://");
  Serial.print(WiFi.localIP());
  WiFiAddr = WiFi.localIP().toString();
  Serial.println("' to connect");
}
```

Для коректного керування платформою потрібно встановити певні налаштування, оголосити відповідні змінні та встановлення початкових значень. Так, для початку, оголошуються глобальні змінні, які зберігають номери GPIO-пінів для керування моторами та світлодіодом (рис. 3.3).

```
extern int gpLb =  2; // Left 1
extern int gpLf = 14; // Left 2
extern int gpRb = 15; // Right 1
extern int gpRf = 13; // Right 2
extern int gpLed =  4; // Light
extern String WiFiAddr ="";
```

Рисунок 3.3 – Оголошення змінних GPIO-пінів для керування

Після оголошення змінних потрібно встановити режим роботи пінів, які оголошені в цих самих змінних. Це відбувається за допомогою метода pinMode(), передаючи в який назви пінів та режим роботи, як вхідні параметри. Параметр OUTPUT означає, що ці піни будуть використовуватись для виводу сигналу (рисунок 3.4). Таким чином, ESP32 буде керувати моторами та світлодіодом, подаючи або знімаючи напругу на цих пінах.

```
pinMode(gpLb, OUTPUT); //Left Backward
pinMode(gpLf, OUTPUT); //Left Forward
pinMode(gpRb, OUTPUT); //Right Forward
pinMode(gpRf, OUTPUT); //Right Backward
pinMode(gpLed, OUTPUT); //Light

//инициализация
digitalWrite(gpLb, LOW);
digitalWrite(gpLf, LOW);
digitalWrite(gpRb, LOW);
digitalWrite(gpRf, LOW);
digitalWrite(gpLed, LOW);
```

Рисунок 3.4 –Встановлення режиму роботи пінів

3.3 Реалізація вебсервера для керування платформою

Для керування платформою, окрім програмного забезпечення для модуля ESP32-CAM, необхідно створити вебсервер з інтерфейсом керування. Це необхідно для можливості керування платформою з різних девайсів, підключених до тієї ж Wi-Fi-мережі, що і платформа, наприклад, ноутбук або смартфон. Це не тільки зробить процес керування платформою більш зручним а й дозволить самому оператору стати більш мобільним.

Вебсервер дає змогу отримати контроль над платформою через веббраузер, а інтерфейс надасть змогу чітко керувати рухами платформи через елементи керування, включати та виключати підсвітку та отримувати відеозображення [4]. Окрім керування, інтерфейс дасть можливість налаштовувати якість зображення, керувати параметрами камери та передачі фото та відео.

Для функціонування вебсервера, розробленого на мові C++, необхідні додаткові бібліотеки, які розширюють функціонал мови програмування та надають зручні інструменти для розробки інтерфейсу та налаштування працездатності програми.

Так, необхідні бібліотеки для роботи із зобрадженнями та відео, перетворення зображення, роботи з таймером для вимірювання часу, керування модулем ESP32-CAM та створення вебсервера сумісного із цим модулем.

Всі необхідні бібліотеки, які задовольнять потреби проєкта, потрібно імплементувати на початку програми, як це зображено на рисунку 3.10.

Після підключення бібліотек треба оголосити змінні (лістинг 3.4), які будуть відповідати GPIO-пінам для керування платформою, а саме:

– піни для руху вперед, назад;

– поворотів вправо та вліво;

– включення світлодіода на модулі ESP32-CAM (лістинг 3.4);

Також оголошується змінна для збереження IP-адреси в мережі Wi-Fi.

Лістинг 3.4 - Підключення необхідних бібліотек та оголошення змінних

```
#include "esp_http_server.h"
#include "esp_timer.h"
#include "esp_camera.h"
#include "img_converters.h"
#include "camera_index.h"
#include "Arduino.h"

extern int gpLb;
extern int gpLf;
extern int gpRb;
extern int gpRf;
extern int gpLed;
extern String WiFiAddr;
```

Функції запуску вебсервера та інтерфейса керування. Функція index _handler призначена для створенняа HTML-сторінки, яка буде відправлена у відповідь на HTTP-запит. Вона встановлює тип відповіді як "text/html" і формує HTML-код, що включає в себе метатег для адаптивної верстки, JavaScript для асинхронних запитів через XMLHttpRequest, а також зображення потоку з камери. В HTML-сторінці є кнопки для управління рухом платформи: вперед, назад, вліво, вправо, зупинка. Ці кнопки мають події, що викликають функції для відправки відповідних команд через HTTP-запити (наприклад, "go", "stop", "left"). Також присутні кнопки для включення та вимикання світла на платформі. Після формування сторінки вона відправляється у відповідь на запит за допомогою функції httpd_resp_send.

Функція startCameraServer ініціалізує HTTP-сервер для керування камерою та платформою. Спочатку створюється конфігурація сервера за допомогою httpd_config_t, а потім реєструються обробники для різних URI-шляхів, таких як /go /back, /stop, /left, /right, /ledon, /ledoff та інші, що відповідають за рухи платформи та управління освітленням. Кожен URI обробляється відповідною функцією, наприклад, go_handler для руху вперед або ledon_handler для вмикання світла. Сервер камери запускається на певному порті, і після цього реєструються всі відповідні обробники. Крім

того, налаштовується другий сервер для потоку з камери, який запускається на іншому порті. У функції також ініціалізується фільтр ra_filter, що використовується для обробки потокових даних. Завершує ініціалізацію запуск вебсервера на відповідному порті та реєстрація всіх потрібних URI-обробників.

Функції керування платформою та налаштування камери. Для керування платформою та налаштування камери модуля керування ESP32-CAM потрібно передавати інструкції та команди, які реалізовані у вигляді методів. Основні методи виконують дві головні задачі, а саме – налаштування камери та зображення, керування платформою.

Налаштування камери виконується у функції cmd_handler(), яка приймає HTTP-запити та змінює параметри камери. Ця функція зчитує вхідні дані, відповідно до них змінює параметри камери за допомогою API esp_camera_sensor_get(), а потім повертає відповідь HTTP.

Знімок з камери отримується у функції capture_handler(), яка викликається при HTTP-запиті. Ця функція використовує esp_camera_fb_get() для отримання кадру, перевіряє, чи кадр отримано успішно, після чого відправляє його у відповідь у форматі JPEG. Після надсилання зображення функція звільняє пам’ять, використану для кадру, і повертає HTTP-відповідь.

Потокова передача відео реалізується у функції stream_handler(), яка обробляє HTTP-запити. Ця функція у циклі постійно захоплює кадри за допомогою esp_camera_fb_get(), формує HTTP-відповідь у форматі MJPEG (multipart/x-mixed-replace) та передає послідовні зображення клієнту. Після кожної передачі кадру використана пам’ять звільняється, а цикл продовжується до завершення з'єднання або помилки.

Основною функцією керування є WheelAct(), яка зображена на рисунку 3.11. Функція використовується для керування платформою, зокрема її колесами, які підключені до пінів ESP32-CAM. Вона отримує чотири параметри, які визначають, які колеса повинні бути активовані (HIGH) або вимкнені (LOW). Таким чином, функція WheelAct() дає можливість точно

контролювати кожне колесо платформи, що дозволяє йому рухатися в різних напрямках або зупинятись.

```
void WheelAct(int nLf, int nLb, int nRf, int nRb)
{
 digitalWrite(gpLf, nLf);
 digitalWrite(gpLb, nLb);
 digitalWrite(gpRf, nRf);
 digitalWrite(gpRb, nRb);
}
```

Рисунок 3.5 – Функція керування WheelAct()

Функція WheelAct() є основою керування платформою. Можливість точно контролювати кожне колесо надає змогу вибору напрямку руху. Вона визивається у фунціях вибору напрямку руху з різними параметрами, що дає змогу зробити платформу досить мобільною. Це реалізовано у функціях, наведених нижче, та зображено на рисунку 3.5.

Функція go_handler() обробляє HTTP-запит на переміщення платформи вперед. Вона викликає функцію WheelAct() з аргументами, що відповідають за активування передніх коліс і задніх коліс платформи для руху вперед (HIGH для передніх лівих та правих коліс, LOW для задніх лівих і правих), як на рисунку 3.6.

Функція back_handler() обробляє HTTP-запит на переміщення платформи назад. Вона викликає функцію WheelAct() з аргументами, які активують задні колеса платформи для руху назад (LOW для передніх лівих та правих коліс, HIGH для задніх лівих і правих).

Функція left_handler() відповідає на HTTP-запит на поворот платформи вліво. Вона викликає функцію WheelAct() з аргументами, які активують переднє ліве колесо (HIGH) та заднє праве колесо (HIGH), при цьому всі інші колеса залишаються вимкненими (LOW). Це забезпечує рух платформи вліво.

Функція right_handler() обробляє HTTP-запит на поворот платформи вправо. Вона викликає функцію WheelAct() з аргументами, що активують

переднє праве колесо (HIGH) та заднє ліве колесо (HIGH), що спричиняє рух платформи вправо.

Функція stop_handler() відповідає за зупинку платформи. Вона викликає функцію WheelAct() з усіма параметрами, встановленими на LOW, що вимикає всі колеса платформи та припиняє її рух.

Функція ledon_handler() виконується, коли прийшов HTTP-запит на ввімкнення світлодіода (LED). Вона викликає функцію digitalWrite() для піну, підключеного до світлодіода (gpLed), і встановлює його в HIGH, вмикаючи LED.

Функція ledoff_handler() виконується при отриманні запиту на вимкнення світлодіода. Вона викликає функцію digitalWrite() для піну, підключеного до світлодіода (gpLed), і встановлює його в LOW, вимикаючи LED.

Ці функції виконують основну роль в керуванні платформою і світлодіодом через HTTP-запити, обробляючи кожен запит відповідно до заданих команд для руху або керування світлом.

```
static esp_err_t go_handler(httpd_req_t *req){
    WheelAct(HIGH, LOW, HIGH, LOW);
    Serial.println("Go");
    httpd_resp_set_type(req, "text/html");
    return httpd_resp_send(req, "OK", 2);
}
```

Рисунок 3.6 – Приклад функції вибору напрямку руху

3.4 Завантаження програмного забезпечення на модулі керування

Після розробки програмного забезпечення його потрібно завантажити на модуль керування ESP32-CAM. Через відсутність роз’єму підключення у модуля ESP32-CAM його необхідно підключити до програматора FT232RL, який дає змогу встановити зв’язок із комп’ютером та завантажити програмне забезпечення. З’єднання програматора та модуля ESP32-CAM має схему

підключення, зображену на рисунку 3.7.

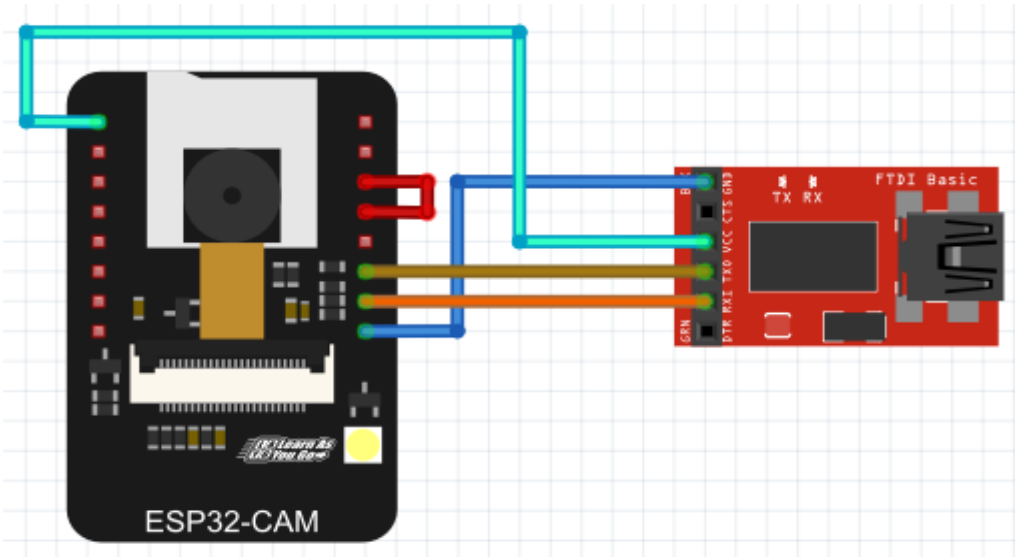

Рисунок 3.7 – Схема підключення модуля ESP32-CAM до програматора FT232RL.

Використовуючи макетну плату, під’єднуємо компоненти за допомогою проводів з типом під’єднання father-father. Підключати компоненти до макетної плати треба за схемою, зображеною на рисунку 3.8.

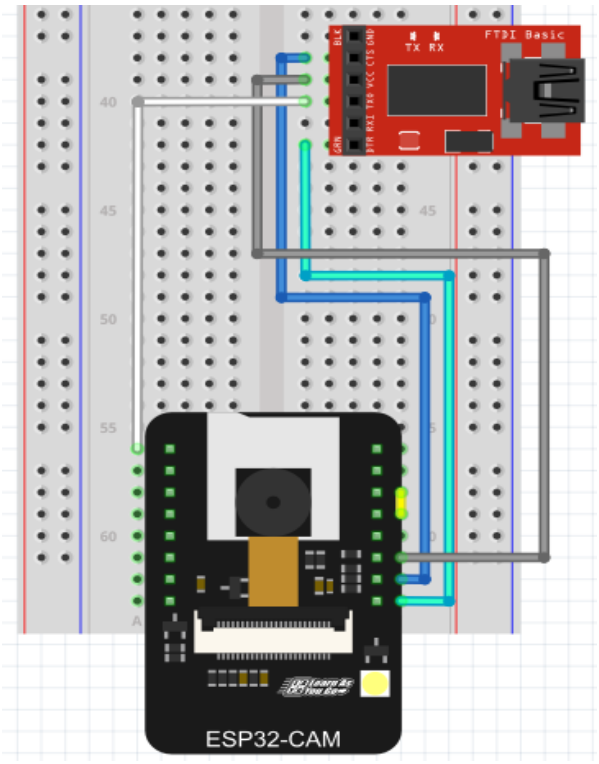

Рисунок 3.8 – Схема підключення ESP32-CAM до програматора FT232RL

Після з'єднання компонентів з макетною платою їх можна підключати до комп'ютера. Після підключення до ком'ютера на програматорі включиться світлодіод, що вказує на встановлення підключення [9]. Це видно на рисунку 3.9. Через особливості роботи прогроматора, можливо, потрібно буде встановити для нього драйвери, які є у вільному доступі на сайті виробника, але на нових версіях Windows це відбувається автоматично.

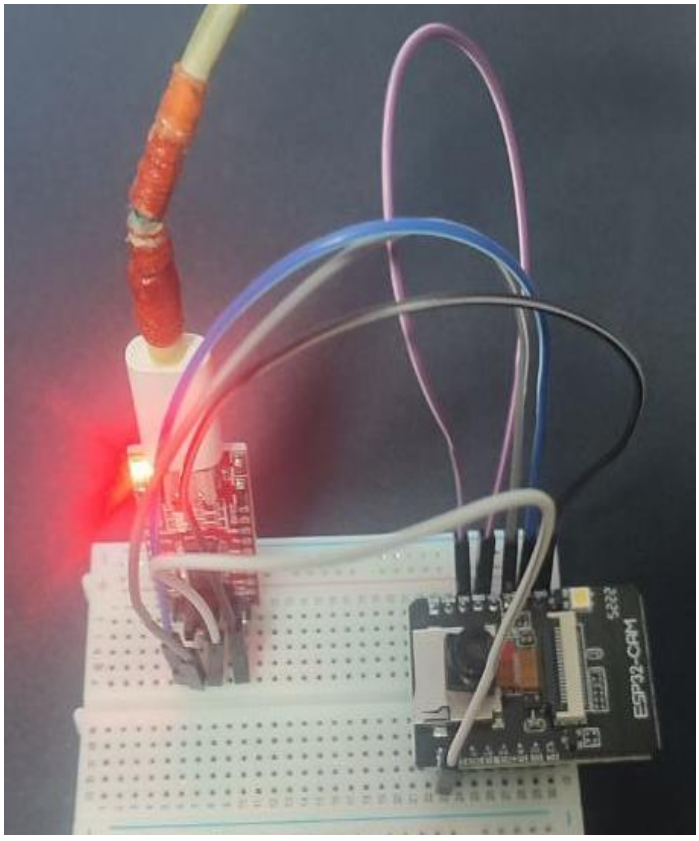

Рисунок 3.9 –Підключені до комп’ютера ESP32-CAM та FT232RL

Після підключення до компьютера можна заванажувати на ESP32-CAM програмне забезпечення. Це відбувається в середовищі розробки Arduino IDE. В налаштуваннях основного файлу програмного забезпечення потрібно вказати посилання на налуштування у вікні менеджера плат, взяте із сайта виробника мікроконтроллера ESP32-CAM, і встановити пакет плат esp. Це необхідно для правильного завантаження програмного забезпечення на плату та її коректної роботи.

Після процесу завантаження програмного забезпечення середовище розробки у вікні монітора порта виведе текст про успішне завантаження та напише IP-адресу, за якою можна буде підключитися до вебсервера з интерфейсом керування на вебсторінці браузера (рисунок 3.10).

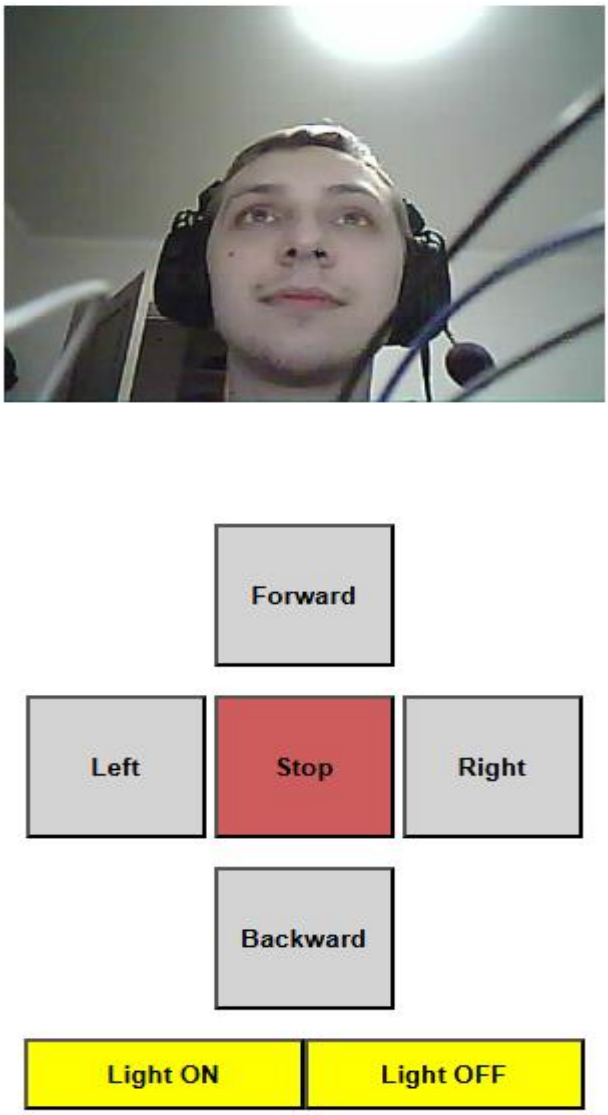

Рисунок 3.10- –Вебсторінка з інтерфейсом керування та відеопотоком

4 ЗБІРКА ПЛАТФОРМИ ТА ТЕСТУВАННЯ ЇЇ ПРАЦЕЗДАТНОСТІ

Після завантаження програмного забезпечення на ESP32-CAM та перевірки працездатності вебсервера можна підключати модуль до інших компонентів. Платформа буде складатися з модуля ESP32-CAM, драйвера керування двигунами L298N, чотирьох електродвигунів, блока живлення та вимикача. Побудувавши схему підключення (рис. 4.1), починаємо з'єднання компонентів.

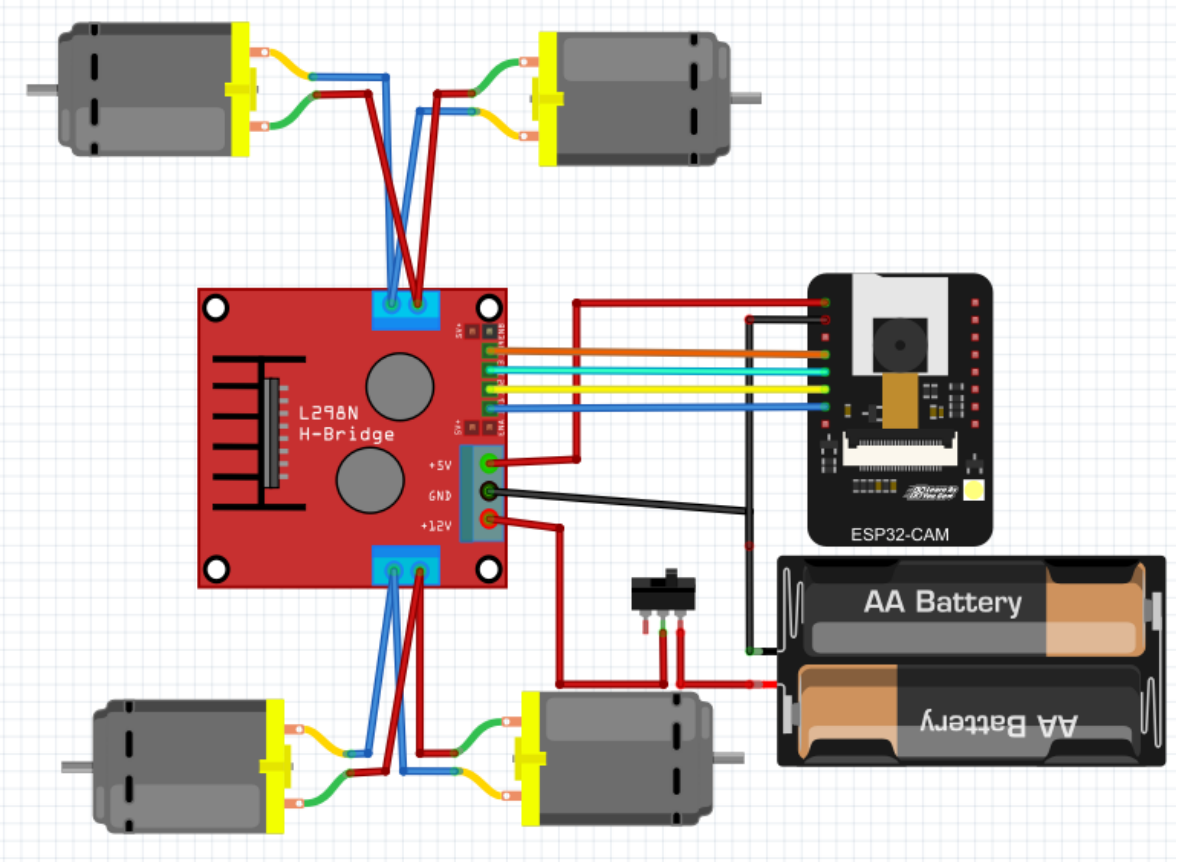

Рисунок 4.1 – Схема з'єднання всіх компонентів

Збирання платформи починаємо з підготовки двигунів. Для початку до них треба під'єднати провідники, після чого зафіксувати на основі платформи, що зображено на рисунках 4.2 та 4.3.

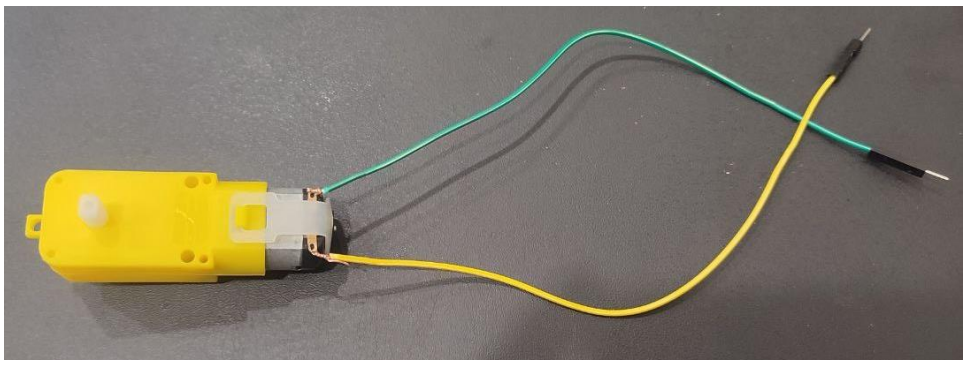

Рисунок 4.2 – Під’єднання провідника до двигунів

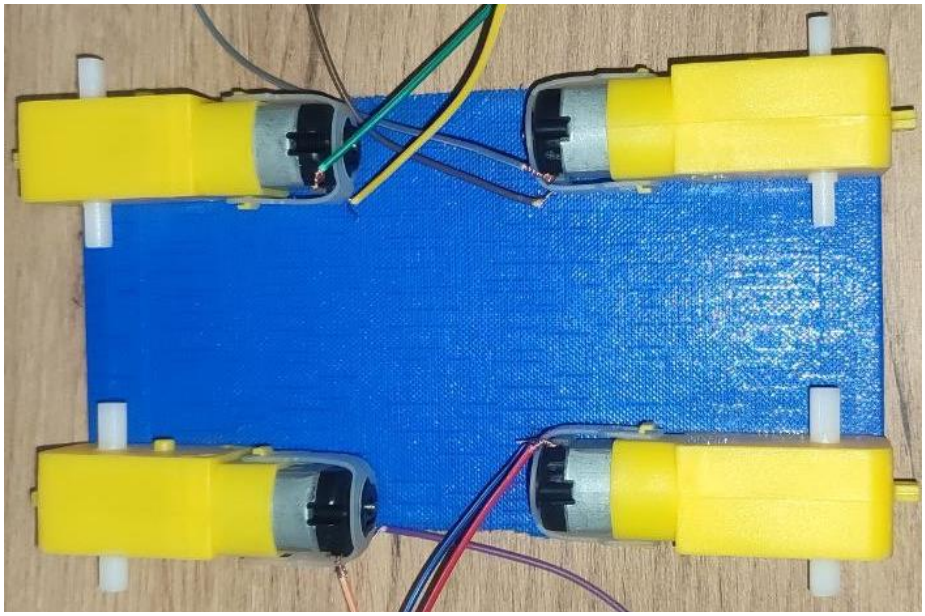

Рисунок 4.3 – Під’єднання двигунів до основи

Потім встановлюємо драйвер двигунів L298N, до якого під’єднуємо до відповідних контактів чотири двигуни, зберігаючи полярність, за допомогою провідників – так, як це зображено на рисунку 4.4. Плюс двигунів підключаємо до плюсу драйвера, мінус двигунів підключаємо до мінуса драйвера.

Після встановлення драйвера можна встановити блок живлення. Його встановлюємо на середину платформи – так, як зображено на рисунку 4.5, для зручного підключення інших компонентів.

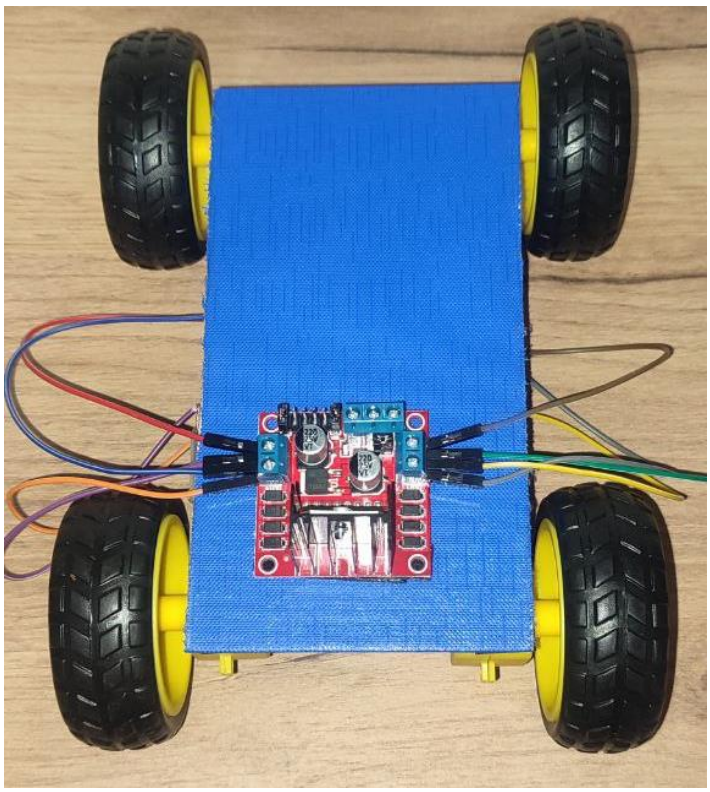

Рисунок 4.4 – Встановлення драйвера L298N та під’єднання до нього двигунів

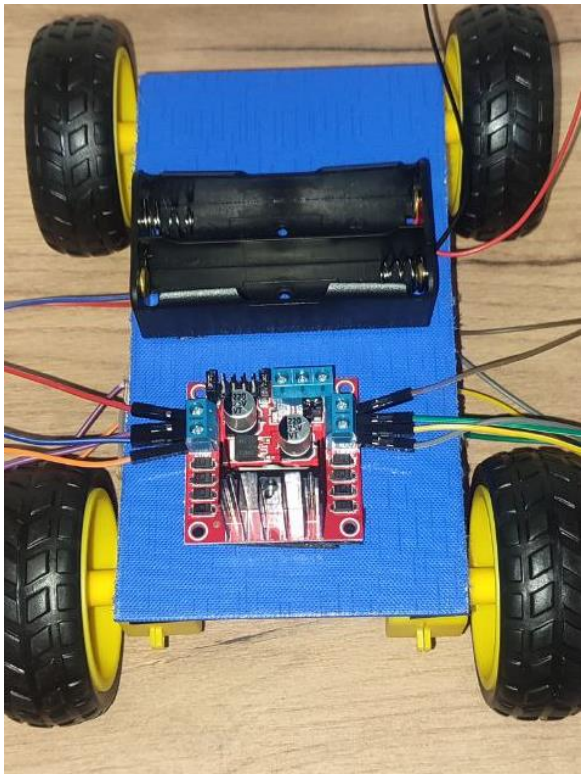

Рисунок 4.5 – Встановлення та під’єднання блока живлення

Блок живлення підключаємо плюсом через вимикач до драйвера через роз’єм +12V, а інший контакт з’єднуємо з контактом живлення модуля ESP32-CAM (рисунок 4.5) і підключаємо до роз’єму GND (ground, «земля») на драйвері. Це дозволить підключити до живлення і драйвер, і модуль керування ESP32-CAM, з можливістю контролювати подання напруги на модулі.

Після встановлення блока живлення можна встановити основний модуль ESP32-CAM, підключивши його до блока живлення та драйвера двигунів (рисунок 4.6). Підключивши блок живлення до роз’єму GND (ground, «земля») на ESP32-CAM та роз’єм драйвера +5V до такого ж роз’єму на модулі. ми підключили модуль до живлення. Роз’єми IN1, IN2, IN3, IN4 на драйвері підключаємо до відповідних роз’ємів GPIO 2, GPIO 14, GPIO 15, GPIO 13 на ESP32-CAM, що дасть змогу керувати двигунами.

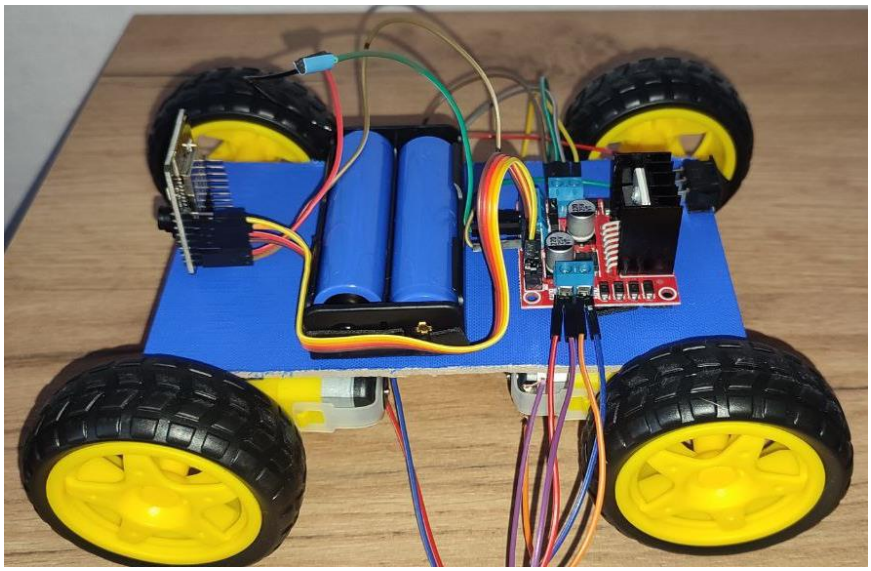

Рисунок 4.6 – Встановлення та під’єднання модуля ES32-CAM

Після підключення всіх модулів можна тестувати роботу платформи. Після включення платформи на драйвері включиться червоний світлодіод, який свідчить про подачу живлення на драйвер та його роботу. Після включення платформа також розвернеться на 180 градусів навколо осі, що покаже працездатність двигунів та правильне їх підключення до драйвера.

У випадку, якщо двигуни підключені неправильно, платформа не зможе зробити оберт. В такому випадку слід вимкнути платформу та змінити полярність двигунів.

Якщо все підключено правильно, після оберту платформа зупиниться на місці. Після чого можна підключатися до вебсервера з інтерфейсом керування. На інтерфейсі повинно бути вікно відеопотоку з камери платформи та кнопки керування, на натискання яких платформа повинна реагувати.

У разі відсутності кнопки, або декількох кнопок керування, або відеопотоку слід перевірити код програмного забезпечення для модуля ESP32-CAM, а саме файл ініціалізації та запуску вебсервера. Чеерз те, що завантаження програмного забезпечення на ESP32-CAM потребує використання макетної плати та програматора, слід зробити монтаж ESP32-CAM на платформу зручним з можливістю легко під’єднати або від’єднати цей модуль.

Також у випадку, коли під час першого включення платформа не реагує відразу, слід зачекати кілька секунд. Це необхідно тому, що процесор модуля не вмикався до цього разом з іншими модулями, і ESP32-CAM потребує певного часу на обробку завантаженого програмного забезпечення.

Після включення платформи та виправлення усіх помилок, якщо такі виникли, вона повинна працювати в стандартному режимі. Рух платформи повинен бути досить плавний та без значних затримок. Відеопотік також повинен бути без серйозних затримок та втрати кадрів.

5 ПРОБЛЕМИ, ЩО ВИНИКЛИ ПРИ РОЗРОБЦІ ПЛАТФОРМИ, ШЛЯХИ ЇХ УСУНЕННЯ

5.1 Проблеми платформи

Під час розробки будь-чого: чи то програмне забезпечення, чи то модулі та плати, чи то інженерні проєкти, виникають проблеми та складнощі [10], вирішеня яких є частиною робочого процесу програмістів, тестувальників та інженерів. Іноді проблеми та труднощі виникають із самого початку розробки – їх помітно відразу, а деякі потрібно навмисно шукати, щоб покращити роботу продукту та виключити ситуації, коли такі непомітні помилки можуть призвести до серйозних проблем у роботі продукту.

При розробці даної платформи також виникали помилки та проблеми, деякі – явні, а деякі були виявлені тільки після тестування платформи.

Скоріше незручністю, аніж проблемою можна назвати і відсутність можливості завантажувати програмне забезпечення напряму на модуль ESP32-CAM. Це збільшує час, витрачений на процес завантаження програмного забезпечення на модуль та його тестування, а також збільшує вартість розробки, адже потрібно додатково придбати програматор та макетну плату.

Основною проблемою платформи стала проблема поганого зв’язку із пристроєм керування. Під час завантаження програмного забезпечення на модуль ESP32-CAM за допомогою макетної плати, було протестовано її здатність до передачі відео. Під час процесу завантаження програмного забезпечення та його тестування модуль ESP32-CAM був під’єднаний до макетної схеми та прогроматора, який, в свою чергу, був під’єднаний до персонального комп’ютера, що дало змогу платі підключитися до інтернет-мережі напряму, тому під час тестування відеопотік був стабільний, з гарною якістю передачі відео та передавав до 70 кадрів на секунду. Але після

відключення ESP32-CAM від макетної плати та встановлення на платформу, вона підключалася до інтернет-мережі через Wi-Fi, використовуючи свій Wi-Fi-модуль. Компактність плати не дозволила зробити Wi-Fi-модуль досить великим і потужним, тому маленький розмір антени не дає змоги передавати великий об'єм даних швидко. Це вплинуло на якість передачі відео і швидкість реагування платформи на команди. При гарному підключенні платформа працює так само, як і в тестовому режимі, але якщо якість підключення падає, то падає і якість передачі відео та швидкість реагування на команди. При слабкому зв'язку з'являється затримка передачі відео та втрачається частина кадрів, виникає затримка в реакції на команди.

Також виникла проблема підключення до вебсервера платформи зі смартфона. При спробі підключитися браузер повідомляє про помилку підключення, бо сайт недоступний. В той же час при спробі підключитися з ПК все працює, як і очікується.

Однією з неявних проблем платформи стало перегрівання модулів під час роботи. Так, недивлячись на наявність радіатора, драйвер L298N під час роботи нагрівається до високих температур, що може впливати на якість та швидкість передачі сигналів від модуля керування ESP32-CAM до двигунів, що може викликати затримки під час керування платформою. Також під час роботи нагрівається і процесор модуля ESP32-CAM, надмірне нагрівання якого може впливати на його швидкодію та якість передачі сигналу.

Відсутність зовнішнього корпусу платформи, хоч і вирішує частково проблему нагрівання модулів, але спричиняє виникнення проблеми із захистом платформи від потрапляння пилу, бруду та вологи, що робить неможливою роботу платформи в поганих кліматичних умовах.

5.2 Вирішення проблем платформи

Незручність завантаження програмного забезпечення на модуль ESP32-CAM вирішується придбанням спеціального програматора саме для цієї

плати. Він під'єднується напряму до плати та дає можливість працювати із програмним забезпеченням без використання макетної плати, що спрощує та пришвидшує процес.

Проблему втрати якості зв’язку можна вирішити кількома способами. Так, одним із шляхів вирішення даної проблеми є зниження якості відео, що передається з камери модуля ESP32-CAM на інтерфейс керування. Це знизить навантаження як на процесор модуля ESP32-CAM, так і на інтернет-мережу, що дозволить платформі працювати.

Також одним із шляхів вирішення проблеми поганого зв’язку може стати зовнішня антена, встановлена на платформі. Але наявність зовнішньої антени потребує зміни конструкції платформи та схеми підключення, та призводить до підвищення загальної вартості платформи, що в межах даного проєкту недоцільне.

Проблему підключення до вебсервера платформи потрібно аналізувати, адже причин її винекнення безліч. Сучасні смартфони можуть блокувати локальні HTTP-запити через безпеку, що виправити, не маючи відповідного доступу до налаштувань смартфона та знань, як це зробити, не є можливим. Також якщо на телефоні встановлений VPN, антивірус або брандмауер, вони можуть блокувати доступ до локальної IP-адреси, тому слід спробувати вимкнути ці програми.

Налаштування сучасних роутерів також можуть викликати цю проблему, адже деякі з них мають ізольовані підмережі для кабельних і Wi-Fi-підключень. В цьому випадку саме налаштування роутера викликало проблему підключення з інших пристроїв, тому для її вирішення потрібно в адмінпанелі змінити налаштування, пов'язані з "AP Isolation" або "Client Isolation". Після вимкнення цих функцій все починає працювати коректно.

Віришити проблему нагрівання модулів можна, встановивши додаткове охолодження на платформу у вигляді одного або декількох кулерів, в залежность від їх розміру. Це дасть змогу охолоджувати всі модулі. Але встановлення такого охолодження створить ще дві проблеми, а саме –

ускладнення схеми підключення модулів та недостатня потужність блоку живлення. Тобто для встановлення кулерної системи охолодження на платформу потрібно оновити загальну схему підключення та встановити більш потужний блок живлення, що збільшить кінцеву вартість платформи. Але такі дії є не досить доцільними, адже нагрів модулів є поступовим, що дає змогу охолоджувати їх просто залишивши платформу на кілька хвилин бездіяльною. Також на охолодження модулів позитивно впливає відсутність зовнішнього корпусу платфоми, що дає модливість модулям охолоджуватися від оточуючого повітря.

5.3 Вдосконалення платформи

Усі проблеми платформи є взаємопов’язаними. Тому підхід до вирішення проблем та вдосконалення платформи повинен бути комплексним.

Для того, щоб вирішити проблеми платформи, потрібно переробити схему підключення модулів, що дасть змогу під’єднати систему охолодження та роз’єм для заряджання акумуляторів. Сам блок живлення можна замінити або використовувати більш потужні акумулятори. Це дає можливість не тільки живити систему охолодження, а й подовжити термін роботи платформи без підзарядки.

Встановлення системи охолодження дає змогу зменшити вплив температури на продуктивність на стабільність роботи модулів. А встановлення корпусу, у свою чергу, підвищить захист від вологи та пилу і дасть можливість встановити на свою верхню частину зовнішню антену, що дає змогу платформі працювати в гірших погодних умовах, на більших відстанях.

Таким чином, впровадження таких вдосконалень, хоча й передбачає ускладнення процесу виготовлення та збільшення вартості платформи, але є необхідними кроками.

Модернізація схеми підключення, впровадження ефективної системи

охолодження, використання більш потужного живлення та встановлення захисного корпусу та зовнішньої антени, у своїй сукупності, забезпечать стабільнішу роботу в ширшому діапазоні умов, збільшать дальність дії та якість зв'язку.

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було розроблено дистанційно керовану платформу з можливістю відеомоніторингу. Основною метою проєкту було створення мобільної платформи, яка може керуватися з будь-якого пристрою, підключеного до Wi-Fi-мережі, та передавати відеопотік у реальному часі.

Проєктування платформи включало вибір та інтеграцію таких ключових компонентів, як мікроконтролер ESP32-CAM, драйвер двигунів L298N, електродвигуни та блок живлення. Програмне забезпечення для керування платформою було розроблено в середовищі Arduino IDE з використанням бібліотек для роботи з камерою та Wi-Fi. Вебінтерфейс, створений для керування платформою, дозволяє користувачу спостерігати за відеопотоком та керувати рухом платформи через браузер.

Під час тестування платформи було виявлено, що вона забезпечує стабільну передачу відео та точне виконання команд. Однак деякі проблеми, такі як обмежена дальність Wi-Fi-зв'язку та нагрівання компонентів, потребують подальшого вдосконалення. Для покращення продуктивності платформи можна розглянути використання зовнішньої антени для покращення якості зв'язку, а також встановлення додаткового охолодження для запобігання перегріву компонентів.

Загалом розроблена платформа демонструє високу ефективність у виконанні поставлених завдань – таких, як дистанційне керування та відеоспостереження, задовільняє усім поставленим вимогам. Вона може бути використана в різних сферах, включаючи охорону, дослідження важкодоступних місць, а також у навчальних цілях. Є можливість розширити функціональність платформи шляхом додаваня нових модулів, таких як датчики відстані, GPS-навігація, та ін.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Цирульник С. М., Азаров О. Д., Крупельницький Л. В., Трояновська Т.І. Мікропроцесорна техніка : Вінниця : ВНТУ, 2017. 122с. URL: https://irbis-nbuv.gov.ua/cgi-bin/irbis_nbuv/cgiirbis_64.exe.pdf (date of access: 20.02.2025).

2. Євчук О. В. Мікроконтроллери : Івано-Франківськ, 2019. 147с. URL: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://dn.nung.edu.-ua/pluginfile.php/141570/mod_resource/content/.pdf (date of access 21.02.2025).

3. Глухов О.В., Кравчук О.О., Левченко Є.В. Вивчення властивостей мікроконтролерів і електронних систем на базі платформи Ардуіно: навч. посібник для студентів ВНЗ. Харків: ХНУРЕ, 2019. 192 с. URL: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://openarchive.nure.ua/server/api/core/bitstreams/7349b613-9f68-4edf-9f1d-c35baac25c76/content (date of access 25.03.2025).

4. Засорнов О. С., Засорнова І. О. Програмування мікроконтролерних та робототехнічних систем. Київ : Кондор, 2023. 280с. URL: https://balka-book.com/ua/tehnicheskaya-1547/programuvannyamkrokontrolernihtaroboto-tehnchnihsistemnavchalniyposbnik-289743 (date of access 24.04.2025).

5. Сокол Є. І., Домнін І. Ф., Рисований О. М. Спеціалізовані мікроконт ролерні системи. Теорія і практика : НТУ "ХПІ", 2007. 252 с. URL: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://web.kpi.kharkov.ua/pbme/wp-content/uploads/sites/161/2016/03/Specializovani-mikrokontrolerni-sistemi.-Teoriya-i-praktika.pdf (date of access 15.05.2025).

6. Blue Bird. What is an FPV drone? URL: https://www.blue-bird.tech/en/news-en/what-is-an-fpv-drone/ (date of access: 21.02.2025).

7. CAL BRYANT. A Comprehensive guide to FPV drone technology. URL: https://calbryant.uk/blog/a-comprehensive-guide-to-fpv-drone-technology/# (date of access: 23.02.2025).

8. Espressif. ESP32-CAM and Other Cool Projects on RNT. URL: https://www.espressif.com/en/news/ESP32_CAM (date of access: 23.02.2025).

9. Espressif.ESP32 moduls. URL: https://www.espressif.com/en/products-/modules (date of access: 23.02.2025).

10. Машталяр С.В., FIRST PERSON VIEW Технології та їх розвиток / Т.Г. Рожнова, С.В. Машталяр С.В // XV МНТК «Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління». – Тези доповідей п’ятнадцятої міжнародної науково-технічної конференції (24 – 25 квітня 2025 року). – Том 2: секція 2. – 2025. – 142 с. (С. 23–24).URL: https://nure.ua/wpcontent/uploads/2025/tom_2_ict_2025_compressed.pdf (date of access: 09.06.2025).