

Міністерство освіти і науки України
Державний заклад
«Луганський національний університет імені Тараса Шевченка»

Факультет технологій та інформаційних систем

Кафедра інформаційних технологій та систем

Капустін Руслан Олексійович

**РОЗРОБКА ЛАБОРАТОРНОГО СТЕНДУ ІоТ "РОЗУМНИЙ
БУДИНОК" НА ESP32**

кваліфікаційна робота

**здобувача вищої освіти першого (бакалаврського) рівня
освітньої програми «Інженерія програмного забезпечення»
за спеціальністю 121 Інженерія програмного забезпечення**

Особистий підпис _____

Руслан КАПУСТІН

Науковий керівник _____

Світлана ДОНЧЕНКО,
асистент кафедри інформаційних
технологій та систем

Завідувач кафедри _____

Микола СЕМЕНОВ,
кандидат педагогічних наук, доцент
кафедри інформаційних технологій
та систем

Лубни – 2026

Міністерство освіти і науки України
Державний заклад „Луганський національний університет
імені Тараса Шевченка”

Факультет

Факультет технологій та інформаційних
систем

Кафедра

Інформаційних технологій та систем

Рівень освіти

Перший (бакалаврський)

Спеціальність

121 «Інженерія програмного забезпечення»

(код, назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри ІТС

М.А. Семенов

(підпис)

(ініціали, прізвище)

“ ”

2025 р.

**ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ**

Капустіну Руслану Олексійовичу

(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) **Розробка лабораторного стенду IoT "Розумний будинок" на ESP32**

Керівник кваліфікаційної роботи

Донченко С.М.

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджена наказом по університету

Від“ ” 2025 року №

2. Строк подання студентом проекту (роботи)

3. Вихідні дані до роботи (проекту)

у результаті виконання роботи

розроблено лабораторний стенд IoT "Розумний будинок" на ESP32

(визначаються кількісні або (та) якісні показники, яким повинен відповідати об'єкт розробки)

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) **АНАЛІТИЧНИЙ ОГЛЯД СУЧАСНИХ ТЕХНОЛОГІЙ**

«РОЗУМНИЙ БУДИНОК» ТА ФОРМУВАННЯ АРХІТЕКТУРИ IoT-

СИСТЕМИ. ОБГРУНТУВАННЯ ТА ВИБІР АПАРАТНО-ПРОГРАМНОГО

ЗАБЕЗПЕЧЕННЯ ЛАБОРАТОРНОГО СТЕНДУ IoT «РОЗУМНИЙ

БУДИНОК» НА БАЗІ МІКРОКОНТРОЛЕРА ESP32.

ПРОЄКТУВАННЯ, КОДУВАННЯ ТА РОЗГОРТАННЯ ПЕРИФЕРІЙНИХ

МОДУЛІВ IoT-ПЛАТФОРМИ «РОЗУМНИЙ БУДИНОК».

(визначаються назви розділів або (та) перелік питань, які повинні увійти до тексту ПЗ)

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада Консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання „_____” _____ 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
	Вибір теми роботи, вивчення наукової літератури, затвердження теми та керівника.	До 15 жовтня	
	Аналіз літературних джерел за темою роботи. Розробка та апробація методики дослідно-експериментальної роботи. Подання структури теоретичної частини роботи та плану експериментальних досліджень.	Другий тиждень листопада (10 листопада)	
	Робота над теоретичною частиною. Подання теоретичної частини роботи для першого читання науковим керівником.	До 15 грудня	
	Усунення зауважень, урахування рекомендацій наукового керівника. Подання теоретичної частини роботи на друге читання.	До 28 січня	
	Проведення експериментальної роботи. Поетапний аналіз та обговорення її результатів. Перевірка стану виконання роботи.	Перший тиждень березня	
	Урахування рекомендацій наукового керівника, усунення недоліків, підготовка варіанта роботи до передзахисту. Розробка презентації.	До 31 березня	
	Попередній захист роботи на кафедрі	квітень	
	Доопрацювання роботи з урахуванням рекомендацій після передзахисту. Подання роботи науковому керівникові та рецензентові на підготовку відгуку та рецензії	За 10 днів до державної атестації	
	Подання на кафедру остаточного варіанта роботи, переплетеного та підписаного автором, науковим керівником і рецензентом.	За 5 днів до державної атестації	

Здобувач освіти

підпис

Руслан КАПУСТІН

(ініціали, прізвище)

Керівник проекту (роботи)

підпис

Світлана ДОНЧЕНКО

АНОТАЦІЯ

Капустін Р. О.

Тема: Розробка лабораторного стенду IoT "Розумний будинок" на ESP32.

Спеціальність: 121 "Інженерія програмного забезпечення"

Установа: ЛНУ імені Тараса Шевченка, 2026 р.

Бакалаврська робота містить: 126 с., 92 рис., 12 табл., 2 додат., 32 джерел.

Об'єкт дослідження – процеси автоматизації, збору телеметричних даних та бездротового керування в кіберфізичних системах Інтернету речей (IoT).

Предмет дослідження – методи, алгоритми, програмне забезпечення мовою MicroPython та апаратні засоби інтеграції сенсорів і актуаторів у межах локальної IoT-платформи «Розумний будинок» на базі мікроконтролера ESP32.

Мета роботи – проєктування, програмно-апаратна реалізація, конфігурування та експериментальне тестування комплексного лабораторного стенду IoT «Розумний будинок» на базі мікроконтролера ESP32 під управлінням мови MicroPython для автоматизації кліматичних, безпекових та комунікаційних сценаріїв житлового простору.

Результати роботи. У роботі проведено аналіз сучасних технологій домашньої автоматизації, мережевих протоколів (X10, KNX, Z-Wave) та комерційних платформ (Ajax, Fibaro). Обґрунтовано доцільність розробки відкритого модульного стенду, сформовано технічні вимоги та спроектовано його трирівневу концептуальну архітектуру.

Виконано підбір та інженерне обґрунтування елементної бази комплексу на основі плати KEYESTUDIO ESP32 PLUS. Реалізовано схеми підключення сенсорного контуру (датчики PIR, DHT11, MQ-2, Steam Sensor, RFID RC522) та блоку виконавчих механізмів (сервопривід SG90, LCD 1602, RGB-діод SK6812, вентилятор, зумер). Програмний стек реалізовано мовою

MicroPython у середовищі Thonny IDE, що забезпечило високу швидкість розробки та налагодження (Rapid Prototyping).

Практична частина охоплює налаштування бездротового зв'язку Wi-Fi (STA, softAP, STA+AP) та створення алгоритмів для підсистем санкціонованого доступу, клімат-контролю та пожежної безпеки зі світлозвуковим оповіщенням. На базі вбудованого TCP/IP стеку розгорнуто автономний локальний вебсервер з адаптивним HTML-інтерфейсом для інтерактивного дистанційного моніторингу та керування актуаторами в реальному часі без залучення зовнішніх хмарних сервісів.

Практична цінність проєкту полягає у створенні автономного, безпечного та хмаронезалежного навчально-демонстраційного комплексу для впровадження в освітній процес ЗВО при вивченні дисциплін галузі IoT та вбудованих систем.

Ключові слова. ІНТЕРНЕТ РЕЧЕЙ (IoT), РОЗУМНИЙ БУДИНОК (SMART HOME), МІКРОКОНТРОЛЕР ESP32, MICROPYTHON, ЛАБОРАТОРНИЙ СТЕНД, СЕНСОРНА ПІДСИСТЕМА, АКТУАТОР, БЕЗДРОТОВИЙ ЗВ'ЯЗОК WI-FI, ЛОКАЛЬНИЙ ВЕБСЕРВЕР, КІБЕРФІЗИЧНА СИСТЕМА, ШВИДКЕ ПРОТОТИПУВАННЯ (RAPID PROTOTYPING).

ABSTRACT

Kapustin Ruslan

Theme: Development of IoT laboratory stand "Smart Home" on ESP32.

Speciality: 121 "Software Engineering"

Institution: Luhansk Taras Shevchenko National University (LTSNU), 2026.

Diploma work contains: 126 pages, 92 Fig., 12 Table, 2 adj., 32 source.

A research object is processes of automation processes, telemetry data collection, and wireless control in cyber-physical systems of the Internet of Things (IoT) automation processes, telemetry data collection, and wireless control in cyber-physical systems of the Internet of Things (IoT).

The article of research is methods, algorithms, software in MicroPython and hardware for integrating sensors and actuators within the local IoT platform "Smart Home" based on the ESP32 microcontroller.

An aim of work is design, hardware and software implementation, configuration and experimental testing of a comprehensive IoT laboratory stand "Smart Home" based on the ESP32 microcontroller and controlled by the MicroPython language for automating climate, security and communication scenarios of a living space.

Job performances. The paper analyzes modern home automation technologies, network protocols (X10, KNX, Z-Wave) and commercial platforms (Ajax, Fibaro). The feasibility of developing an open modular stand is substantiated, technical requirements are formed and its three-level conceptual architecture is designed.

The selection and engineering justification of the element base of the complex based on the KEYESTUDIO ESP32 PLUS board are performed. The connection diagrams of the sensor circuit (PIR sensors, DHT11, MQ-2, Steam Sensor, RFID RC522) and the block of actuators (SG90 servo drive, LCD 1602, SK6812 RGB diode, fan, buzzer) are implemented. The software stack is implemented in the MicroPython language in the Thonny IDE environment, which ensures high speed of development and debugging (Rapid Prototyping).

The practical part covers the configuration of Wi-Fi wireless communication (STA, softAP, STA+AP) and the creation of algorithms for authorized access subsystems, climate control and fire safety with light and sound notification. Based on the built-in TCP/IP stack, an autonomous local web server with an adaptive HTML interface has been deployed for interactive remote monitoring and control of actuators in real time without involving external cloud services.

The practical value of the project lies in the creation of an autonomous, secure and cloud-independent educational and demonstration complex for implementation in the educational process of higher education institutions when studying disciplines in the field of IoT and embedded systems.

Keywords. INTERNET OF THINGS (IoT), SMART HOME, ESP32 MICROCONTROLLER, MICROPYTHON, LABORATORY STAND, SENSOR SUBSYSTEM, ACTUATOR, Wi-Fi WIRELESS COMMUNICATION, LOCAL WEB SERVER, CYBER-PHYSICAL SYSTEM, RAPID PROTOTYPING.

ІТС. ІПЗ4.0726-ВІ
ВІДОМІСТЬ ПРОЄКТУ. РОЗРОБКА ЛАБОРАТОРНОГО СТЕНДУ ІoT
"РОЗУМНИЙ БУДИНОК" НА ESP32.

[illegible]

Міністерство освіти і науки України
Державний заклад «Луганський національний університет
імені Тараса Шевченка»

Факультет (інститут)

Факультет технологій та інформаційних
систем

(повна назва)

Кафедра

Кафедра інформаційних технологій та
систем

(повна назва)

(код, назва)

ТЕХНІЧНЕ ЗАВДАННЯ

на виконання програмної розробки (ПР):

**"РОЗРОБКА ЛАБОРАТОРНОГО СТЕНДУ ІoT "РОЗУМНИЙ
БУДИНОК" НА ESP32"**

ІТС. ІПЗ4.0726.02-ТЗ

ПОГОДЖЕНО

Керівник кваліфікаційної роботи

Донченко С.М.

“ _____ ” 2026р

ВИКОНАВЕЦЬ

Студент групи 4ІПЗ

Капустін Р. О.

“ _____ ” 2026р

Лубни – 2026

ЗМІСТ

1. ВСТУП	3
2. ПРИЗНАЧЕННЯ ПРОДУКЦІЇ	3
3. МЕТА СТВОРЕННЯ СИСТЕМИ.....	3
4. ХАРАКТЕРИСТИКА ОБ'ЄКТА.....	4
4.1. Основні задачі та функції об'єкта	4
5. ВИМОГИ ДО СИСТЕМИ	5
5.1. Вимоги до системи в цілому	5
5.1.1. Вимоги до функціональної структури.....	5
5.1.2. Вимоги до апаратного забезпечення.....	5
5.1.3. Вимоги до програмного та мережевого забезпечення.....	6
5.1.4. Вимоги до надійності та безпеки	7
6. ТЕХНІКО-ЕКОНОМІЧНІ ВИМОГИ ДО КІНЦЕВОГО ПРОДУКТУ	7
7. ВИМОГИ ДО МАТЕРІАЛІВ ТА КОМПЛЕКТУЮЧИХ.....	8
8. ЕТАПИ ВИКОНАННЯ ПР.....	8
9. ПРИЙМАННЯ	9
10. ПОРЯДОК ВНЕСЕННЯ ЗМІН ДО ТЕХНІЧНОГО ЗАВДАННЯ, ЩО	
ЗАТВЕРДЖЕНО.....	10

1. ВСТУП

1.1 Найменування: Розробка лабораторного стенду IoT "Розумний будинок" на ESP32.

1.2 Шифр ПР: АДР-12

1.3 Підстава до виконання ПР: Підставою для виконання даної розробки є завдання на дипломний проєкт.

1.4 Терміни розробки:

1.4.1 Початок 30 жовтня 2025р.

1.4.2 Закінчення 20 квітня 2026р.

1.5 Фінансується за рахунок коштів замовника.

2. ПРИЗНАЧЕННЯ ПРОДУКЦІЇ

2.1. Призначення:

Головним призначенням розробки є створення апаратно-програмного комплексу лабораторного стенду для дослідження технологій інтернет-речей (IoT) у розрізі автоматизації житлових приміщень. Система має на меті наочну демонстрацію принципів збору даних з різноманітних сенсорів, обробки сигналів мікроконтролером та управління виконавчими механізмами для забезпечення комфорту, безпеки та енергоефективності.

3. МЕТА СТВОРЕННЯ СИСТЕМИ

Головною метою проєкту є розробка, програмно-апаратна інтеграція та відлагодження діючого малогабаритного лабораторного стенду інтернет-речей (IoT) «Розумний будинок» на базі двоядерного 32-бітного мікроконтролера ESP32 під керуванням операційного середовища MicroPython. Стенд призначений для комплексної симуляції кіберфізичних процесів, алгоритмів автоматизації інженерних мереж та сценаріїв забезпечення безпеки і життєдіяльності в інтелектуальних житлових приміщеннях.

Для досягнення поставленої мети розроблювана система повинна забезпечити вирішення таких завдань:

- Автоматизація збору та первинної обробки телеметрії: Організація безперервного опитування гетерогенної мережі сенсорів (DHT11, MQ-2, PIR, датчика пари) в режимі реального часу без блокування основних обчислювальних потоків мікроконтролера.
- Реалізація підсистеми санкціонованого доступу та безпеки: Створення програмних алгоритмів ідентифікації користувачів за допомогою радіочастотної технології RFID (модуль RC522) з подальшим керуванням фізичним виконавчим механізмом (сервопривод SG90) та візуалізацією статусу доступу.
- Забезпечення превентивної безпеки та протиаварійного захисту: Розробка тригерних сценаріїв негайного реагування на надзвичайні ситуації (фіксація критичних концентрацій газу, задимленості або протікання води) через активацію локального світлозвукового сповіщення (п'єзозумер, адресні світлодіоди SK6812) та запуск аварійної вентиляції (мотор постійного струму через драйвер HR1124S).
- Кроссплатформне бездротове керування: Розгортання автономного локального TCP-веб-сервера безпосередньо у пам'яті ESP32 для забезпечення віддаленого моніторингу даних та асинхронного керування актуаторами через веб-інтерфейс з будь-якого мобільного чи стаціонарного пристрою.

4. ХАРАКТЕРИСТИКА ОБ'ЄКТА

Об'єктом автоматизації та моделювання є умовне житлове або офісне приміщення, що містить розподілені інженерні вузли, підсистеми життєзабезпечення та бар'єри контролю доступу.

4.1. Основні задачі та функції об'єкта

Об'єкт розробки повинен забезпечувати виконання таких ключових функцій:

1. Моніторинг параметрів середовища. Вимірювання температури, відносної вологості повітря, детектування появи водяної пари або конденсату.

2. Контроль безпеки життєдіяльності. Безперервний аналіз повітряного простору на наявність чадного газу/диму, фіксація переміщення сторонніх осіб у зоні видимості сенсорів.

3. Санкціонування доступу. Ідентифікація користувачів на фізичному периметрі за допомогою радіочастотних міток.

4. Виконавча реакція та індикація. Фізична зміна стану виконавчих пристроїв (серводвигунів, вентиляторів) та генерація світлозвукових сигналів тривоги у разі виникнення позаштатних чи аварійних ситуацій.

5. ВИМОГИ ДО СИСТЕМИ

5.1. Вимоги до системи в цілому

5.1.1. Вимоги до функціональної структури

Програмно-апаратний комплекс повинен мати чітку ієрархічну структуру, що складається з чотирьох взаємопов'язаних підсистем:

1. Підсистема збору даних (інформаційна). Здійснює періодичне безблокувальне опитування датчиків, первинну фільтрацію шумів та передачу числових значень телеметрії до центрального процесора.

2. Підсистема логічної обробки (керування). На основі отриманих даних аналізує поточні тригери (перевищення порогової концентрації диму, зчитування RFID-картки) та викликає відповідні програмні скрипти автоматизації.

3. Підсистема виконавчих механізмів (актуації). Реалізує фізичний відгук стенду (обертання валу сервоприводу, запуск реле/двигуна, активація сирени).

4. Підсистема користувацького інтерфейсу (HMI). Забезпечує локальне виведення метрик на LCD-екран та дистанційне відображення даних/приймання команд через Web-інтерфейс.

5.1.2. Вимоги до апаратного забезпечення

До складу апаратної частини лабораторного стенду висуваються вимоги щодо використання таких сертифікованих компонентів:

- **Обчислювальний модуль:** KEYESTUDIO ESP32 PLUS (SoC ESP32-WROOM-32, 2 ядра Xtensa LX6, 240 МГц, 520 КБ SRAM, 4 МБ Flash).
- **Сенсорна периферія:**
 - Цифровий термо-гігрометр DHT11 (діапазон 0-50°C, вологість 20-90%);
 - Напівпровідниковий сенсор диму та горючих газів MQ-2 (аналоговий та дискретний виходи);
 - Аналоговий датчик водяної пари (Steam Sensor);
 - Пасивний інфрачервоний датчик руху PIR (з регулюванням чутливості та часу затримки);
 - Модуль зчитування радіочастотних карток RFID RC522 (робоча частота 13,56 МГц, інтерфейс SPI/I2C).
- **Виконавчі та індикаційні пристрої:**
 - Символьний рідкокристалічний дисплей LCD 1602 з розширювачем портів PCF8574T (інтерфейс I2C, адреса 0x27);
 - Мікросерводвигун SG90 (кут повороту 0-180 градусів, керування за допомогою ШІМ);
 - Модуль двигуна постійного струму 130 з інтегованим Н-мостовим драйвером HR1124S;
 - Чотирьохпиксельний адресний RGB-модуль на чіпах SK6812;
 - Пасивний п'єзоелектричний випромінювач (Passive Buzzer Module).
- **Конструктивні вимоги.** З'єднання компонентів має виконуватися за допомогою гнучких 4-пінових шлейфів із роз'ємами EASY Plug RJ11, що унеможливорює неправильне підключення (захист від переполюсовки та короткого замикання).

5.1.3. Вимоги до програмного та мережевого забезпечення

Вбудоване ОС-середовище. На мікроконтролер має бути прошита офіційна збірка інтерпретатора MicroPython (сумісна зі стандартом Python 3), адаптована під архітектуру ESP32.

Інструментарій розробника. Написання, завантаження та відлагодження файлової структури (boot.py, main.py) повинно здійснюватися через Thonny IDE.

Мережеві протоколи. Вбудований сокет-сервер має функціонувати на базі стеку TCP/IP (порт 80).

Режими Wi-Fi модуля. Система повинна підтримувати одночасну роботу у режимах Клієнта (STA) для підключення до роутера та Точки доступу (softAP) для прямого сполучення зі смартфоном користувача без стороннього мережевого обладнання.

Інтерфейс користувача. Веб-сторінка має бути реалізована за допомогою мов HTML5/CSS3 та оптимізована для коректного відображення як на десктопних, так і на мобільних браузерях (Safari, Chrome, Firefox).

5.1.4. Вимоги до надійності та безпеки

Стійкість до збоїв живлення. У разі раптового вимкнення та подальшого відновлення живлення мікроконтролер повинен автоматично перезавантажуватися та відновлювати виконання основного алгоритму з файлу main.py без втрати працездатності.

Обробка виключних ситуацій. Програмний код має містити конструкції try-ехсерт для обробки помилок мережевого з'єднання (наприклад, втрата Wi-Fi сигналу), запобігаючи критичному зависанню системи.

Електрична безпека. Номінальна напруга живлення стенду не повинна перевищувати 5 В постійного струму (живлення від USB-порту ЕОМ або стандартного адаптера), що забезпечує повну безпеку для студентів під час виконання лабораторних робіт.

6. ТЕХНІКО-ЕКОНОМІЧНІ ВИМОГИ ДО КІНЦЕВОГО ПРОДУКТУ

Вартість робіт по розробці даної ПР визначається згідно договору на розробку. Вартість запропонованих аналогів повинна забезпечити економічну доцільність їх застосування.

7. ВИМОГИ ДО МАТЕРІАЛІВ ТА КОМПЛЕКТУЮЧИХ

7.1. Вимоги до екологічної безпечності під час експлуатації.

Не пред'являються.

7.2. Спеціальні вимоги до кінцевого продукту.

Не пред'являються.

7.3. Вимоги до безпеки для населення під час експлуатації продукції.

Не пред'являються.

8. ЕТАПИ ВИКОНАННЯ ПР.

Етапи виконання ПР можуть уточнювати згідно календарного плану робіт по узгодженню між замовником та виконавцем:

- **Аналітичний етап:** Огляд літератури, аналіз технологій IoT та формування архітектури системи.
- **Схемотехнічний етап:** Проектування схем підключення датчиків та виконавчих пристроїв до розпіновки (pinout) ESP32.
- **Етап програмної розробки:**
 - Написання драйверів та функцій для локального зчитування даних і логіки вбудованих підсистем.
 - Розробка коду HTTP-веб-сервера та інтеграція HTML/CSS інтерфейсу.
 - Налаштування сокет-з'єднання для комунікації з мобільним додатком.
- **Тестування та налагодження:** Комплексна перевірка роботи стенду в різних режимах, симуляція аварійних ситуацій (відсутність води, рух уночі), виправлення помилок (debugging).

№	Етапи виконання роботи	Термін виконання та обсяг робіт	звітні матеріали
1	Аналіз розробки програмного комплексу та розробка першої версії. Аналіз вимог. Розробка структури. Попереднє тестування		Частковий програмний комплекс на ЕОМ замовника, що

№	Етапи виконання роботи	Термін виконання та обсяг робіт	звітні матеріали
			виконує всі основні функції та звітна документація п.8.2
2	Коректування структури. Розробка допоміжних функцій. Розробка остаточної версії програмного комплексу та його опрацювання. Тестування		Готовий програмний комплекс на ЕОМ замовника та звітна документація п.8.2
3	Доопрацювання окремих модулів та навчання користувачів. Розробка звітних матеріалів згідно п.8 цього ТЗ		звітні матеріали згідно пункту 8

9. ПРИЙМАННЯ

7.1. Необхідні вимоги для впровадження ПР та завершення робіт.

Оцінка результатів розробки і доцільність її продовження здійснюється замовником по представленню наступних матеріалів:

- встановлений програмний комплекс на ЕОМ замовника;
- перелік файлів на резервному носії;
- стислий опис роботи ПР та опис всіх файлів, які необхідні для роботи ПР.
- перелік документів
 - Технічне завдання
 - Пояснювальна записка

7.2. Перелік звітих документів, необхідних для прийняття етапів роботи:

- стислий опис результатів етапу у вигляді анотованого звіту(для 1 та 2 етапів);
- частковий програмний комплекс на ЕОМ замовника згідно календарного плану робіт;
- акт приймання продукції.

7.3. Загальний перелік до приймання звітних документів, макетів, експериментальних зразків.

До приймання пред'являються: акт здачі-приймання продукції, акт впровадження ПР.

7.4. Тестування ПР

Тестування виконується до "Програми та методики тестування", яка розробляється виконавцем та затверджується замовником

10. ПОРЯДОК ВНЕСЕННЯ ЗМІН ДО ТЕХНІЧНОГО ЗАВДАННЯ, ЩО ЗАТВЕРДЖЕНО.

Дане технічне завдання може уточнюватися в процесі розробки ПР при узгодженні сторін з оформленням доповнень до ТЗ.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЗ «ЛУГАНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА»

Факультет технологій та інформаційних систем

(назва факультету, інституту)

Інформаційних технологій та систем

(назва кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

БАКАЛАВРА

(освітньо-кваліфікаційний рівень)

на тему: РОЗРОБКА ЛАБОРАТОРНОГО СТЕНДУ ІоТ "РОЗУМНИЙ
БУДИНОК" НА ESP32

Виконав: студент 4 курсу

Спеціальності 121 «Інженерія програмного
забезпечення»

(шифр і назва напрямку підготовки, спеціальності)

Капустін Р.О.

(прізвище та ініціали)

Керівник

Донченко С.М.

(прізвище та ініціали)

Рецензент

Козуб Ю.Г.

(прізвище та ініціали)

Лубни – 2026 року

ЗМІСТ

ВСТУП.....	5
РОЗДІЛ 1. АНАЛІТИЧНИЙ ОГЛЯД СУЧАСНИХ ТЕХНОЛОГІЙ «РОЗУМНИЙ БУДИНОК» ТА ФОРМУВАННЯ АРХІТЕКТУРИ ІОТ- СИСТЕМИ.....	9
1.1. Концептуальні засади побудови систем домашньої автоматизації Smart Home	9
1.1.1. Історія виникнення, еволюція та сучасний стан технологій «Розумний будинок»	9
1.1.2. Принципи об'єднання інженерних підсистем у єдиний комплекс керування	11
1.1.3. Функціональні можливості розумного житла: енергоефективність, мікроклімат, безпека та комфорт	13
1.1.4. Аналіз переваг та інженерних складнощів впровадження автоматизованих рішень.....	15
1.2. Порівняльний аналіз комерційних рішень та мережевих протоколів автоматизації будівель.....	17
1.3. Формування технічних вимог та концептуальної архітектури розроблюваної системи «Розумний будинок»	27
Висновки до розділу	33
РОЗДІЛ 2. ОБГРУНТУВАННЯ ТА ВИБІР АПАРАТНО-ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ЛАБОРАТОРНОГО СТЕНДУ ІОТ «РОЗУМНИЙ БУДИНОК» НА БАЗІ МІКРОКОНТРОЛЕРА ESP32	36
2.1. Огляд та аналіз компонентної бази апаратної складової системи	36
2.1.1. Центральний обчислювальний модуль системи.....	36
2.1.2. Підсистема сенсорів та збору даних	42
2.1.2.1. Датчик руху PIR Motion Sensor Module.....	42
2.1.2.2. Датчик температури та вологості DHT11	45
2.1.2.3. Модуль цифрової кнопки EASY Plug RJ11.....	49
2.1.2.4. Датчик водяної пари (Steam Sensor)	51

2.1.2.5. Модуль радіочастотної автентифікації RFID RC522	54
2.1.2.6. Напівпровідниковий датчик газу та диму MQ-2	59
2.1.3. Підсистема виконавчих механізмів (Актуатори)	63
2.1.3.1. Пасивний п'єзоелектричний модуль дзвінка (Passive Buzzer)	63
2.1.3.2. Модуль світлодіода Keyestudio LED Module	65
2.1.3.3. RGB-модуль SK6812	67
2.1.3.4. Мікросервопривід SG90	70
2.1.3.5. Рідкокристалічний дисплей LCD 1602 (з I2C-адаптером)	74
2.1.3.6. Модуль двигуна постійного струму (Малий вентилятор)	77
2.2. Обґрунтування вибору мови програмування та середовища розробки	79
Висновки до розділу	91
РОЗДІЛ 3. ПРОЄКТУВАННЯ, КОДУВАННЯ ТА РОЗГОРТАННЯ ПЕРИФЕРІЙНИХ МОДУЛІВ ІОТ-ПЛАТФОРМИ «РОЗУМНИЙ БУДИНОК»	93
3.1. Дослідження та налаштування режимів бездротового зв'язку Wi-Fi на базі мікроконтролера ESP32	93
3.1.1. Програмна реалізація мережевого підключення платформи у режимі клієнта (STA)	93
3.1.2. Організація автономної точки доступу (AP) на базі периферійного контролера	97
3.1.3. Конфігурування гібридного режиму взаємодії (STA+AP) для IoT-вузлів	99
3.2. Програмно-апаратна реалізація підсистеми санкціонованого доступу на основі RFID-технологій	102
3.3. Проєктування та тестування автоматизованої підсистеми мікроклімату та терморегулювання	106
3.4. Розробка та експериментальне дослідження підсистеми газового аналізу та аварійного оповіщення	110
3.5. Розгортання локального веб-сервера для комплексного дистанційного керування компонентами «Розумного будинку»	112

Висновки до розділу	117
ВИСНОВКИ	119
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	123
ДОДАТОК А	127
ДОДАТОК Б	131

ВСТУП

Сучасний етап розвитку інформаційних технологій характеризується стрімким проникненням концепції Інтернету речей (Internet of Things, IoT) у повсякденне життя людини, що зумовлює трансформацію класичного житлового простору в інтелектуальні кіберфізичні системи. Одним із найяскравіших і найбільш динамічних прикладів практичної реалізації IoT є системи типу «Розумний будинок» (Smart Home). Вони інтегрують диверсифікований комплекс сенсорів, обчислювальних модулів та виконавчих механізмів у єдину узгоджену інфраструктуру, яка забезпечує автоматизоване керування мікрокліматом, енергоефективністю, санкціонованим доступом та комплексним безпековим контуром об'єкта.

Специфіка проектування сучасних IoT-рішень вимагає від інженерів програмного забезпечення глибоких знань не лише у сфері високорівневого кодування, а й у галузі мікроконтролерної схемотехніки, бездротових комунікацій (Wi-Fi, Bluetooth, BLE) та розгортання локальних мережесервісів. У зв'язку з цим виникає гостра науково-методична та практична потреба у створенні спеціалізованого лабораторного обладнання. Розробка фізичних навчально-демонстраційних стендів дозволяє здійснювати наочне інженерне прототипування, відпрацьовувати складні алгоритми взаємодії периферійних вузлів та досліджувати режими бездротового зв'язку в реальних фізичних умовах.

Оптимальною апаратною основою для побудови таких систем є сучасні однокристальні системи, серед яких ключове місце посідає сімейство мікроконтролерів ESP32 (зокрема в інтеграції з модулями типу ESP32-WROOM-32 на платах KEYESTUDIO ESP32 PLUS). Завдяки наявності двоядерного 32-бітного обчислювального процесора Xtensa LX6, апаратній підтримці Wi-Fi/Bluetooth екосистем, розвиненій периферії (інтерфейси I2C, SPI, UART, АЦП, ШІМ) та високій енергоефективності, даний контролер дозволяє закрити переважну більшість потреб інтелектуальної автоматизації без залучення сторонніх шлюзів. Більше того, використання MicroPython як інтерпретованого

програмного стеку замість класичного C/C++ забезпечує високу швидкість розробки (Rapid Prototyping) та надає гнучкі інструменти інтерактивного відлагодження (консоль REPL, інструмент Plotter). Таким чином, розробка лабораторного стенду IoT "Розумний будинок" на базі мікроконтролера ESP32 є актуальним науково-практичним завданням, що безпосередньо відповідає вектору розвитку сучасної інженерії програмного забезпечення.

Об'єкт дослідження – процеси автоматизації, збору телеметричних даних та бездротового керування в кіберфізичних системах Інтернету речей (IoT).

Предмет дослідження – методи, алгоритми, програмне забезпечення мовою MicroPython та апаратні засоби інтеграції сенсорів і актуаторів у межах локальної IoT-платформи «Розумний будинок» на базі мікроконтролера ESP32

Мета роботи – проектування, програмно-апаратна реалізація, конфігурування та експериментальне тестування комплексного лабораторного стенду IoT «Розумний будинок» на базі мікроконтролера ESP32 під управлінням мови MicroPython для автоматизації кліматичних, безпекових та комунікаційних сценаріїв житлового простору.

Для досягнення поставленої мети необхідно вирішити такі завдання:

1. Здійснити порівняльний аналіз чинних комерційних рішень та мережевих протоколів на ринку систем «Розумний будинок» і обґрунтувати доцільність розробки модульного навчального стенду з відкритою архітектурою.
2. Обґрунтувати вибір елементної бази обчислювального ядра (KEYESTUDIO ESP32 PLUS) та виконати аналіз компонентів підсистем збору даних (сенсори PIR, DHT11, MQ-2, Steam Sensor, RFID RC522) та виконавчих механізмів (актуатори SG90, LCD 1602, SK6812, вентилятор, зумер).
3. Дослідити та програмно реалізувати режими бездротового зв'язку Wi-Fi (Station, softAP, AP+Station) на базі периферійного контролера для забезпечення мережевої взаємодії.

4. Спроекувати, закодувати та протестувати автономну підсистему санкціонованого доступу на основі RFID-технологій та сервоприводу.
5. Розробити та експериментально дослідити автоматизовані підсистеми терморегулювання, газового аналізу й аварійного світлозвукового оповіщення.
6. Розгорнути автономний локальний веб-сервер на базі мікроконтролера ESP32 для комплексного дистанційного керування компонентами «Розумного будинку» через кросплатформний користувацький інтерфейс.

Для вирішення поставлених завдань у роботі застосовано такі методи:

методи системного аналізу (для огляду існуючих систем автоматизації та вибору компонентної бази); методи об'єктно-орієнтованого програмування та структурування кодів MicroPython (для створення логіки керування периферією); методи комп'ютерного моделювання та алгоритмізації (для побудови схем взаємодії вузлів); експериментально-технічні методи тестування апаратних засобів у реальному часі за допомогою інструментів середовища Thonny IDE (для аналізу часових діаграм, калібрування сенсорів та оцінки відгуку актуаторів).

У першому розділі здійснено ретроспективний та системний аналіз розвитку концепції «Розумний будинок» від аналогових рішень до кіберфізичних IoT-систем. Проведено критичний порівняльний аналіз поширених мережевих протоколів (X10, KNX, Z-Wave) та закритих комерційних платформ (Ajax, BroadLink, Fibaro), визначено їхні архітектурні й економічні обмеження. На основі системного підходу сформульовано функціональні й технічні вимоги та спроектовано концептуальну трирівневу архітектуру відкритого лабораторного стенду домашньої автоматизації.

У другому розділі науково обґрунтовано вибір апаратно-програмного забезпечення системи. Як обчислювальне ядро обрано та детально проаналізовано плату KEYESTUDIO ESP32 PLUS на базі мікроконтролера

ESP32-WROOM-32. Виконано повний інженерний огляд і розрахунок параметрів сенсорної підсистеми (датчики PIR, DHT11, MQ-2, Steam Sensor, RFID RC522) та блоку виконавчих механізмів (SG90, LCD 1602 I2C, SK6812, вентилятор, зумер). Обґрунтовано доцільність розробки програмного стеку мовою MicroPython у кросплатформеному середовищі Thonny Python IDE для забезпечення швидкого інженерного прототипування (Rapid Prototyping).

У третьому розділі здійснено безпосередню програмно-апаратну реалізацію, кодування та експериментальне тестування периферійних модулів розробленої IoT-платформи. Досліджено та налаштовано три топологічні режими бездротового зв'язку Wi-Fi (STA, softAP, STA+AP). Спроектовано та верифіковано алгоритми роботи підсистем санкціонованого доступу (RFID), автоматичного мікроклімату й терморегулювання, а також протипожежної безпеки й аварійного світлозвукового оповіщення. На базі вбудованого TCP/IP стеку ESP32 успішно розгорнуто автономний локальний вебсервер для інтегрованого дистанційного керування всіма компонентами «Розумного будинку» через кросплатформовий графічний HTML-інтерфейс у реальному часі.

Практичне значення отриманих результатів полягає у створенні завершеного, працездатного та високотехнологічного апаратно-програмного комплексу інтернет-речей (IoT), який має чітко виражену прикладну інженерну та науково-методичну спрямованість. Спроектована система є універсальною відкритою платформою, результати розробки якої мають практичну цінність.

РОЗДІЛ 1. АНАЛІТИЧНИЙ ОГЛЯД СУЧАСНИХ ТЕХНОЛОГІЙ «РОЗУМНИЙ БУДИНОК» ТА ФОРМУВАННЯ АРХІТЕКТУРИ ІoT- СИСТЕМИ

1.1. Концептуальні засади побудови систем домашньої автоматизації Smart Home

1.1.1. Історія виникнення, еволюція та сучасний стан технологій «Розумний будинок»

Формування концепції автоматизації житлового простору розгорталося паралельно із глобальним прогресом у сферах мікроелектроніки, обчислювальної техніки та інфокомунікацій. Ретроспективний аналіз дозволяє виділити чотири ключові етапи еволюції цих систем, кожен з яких характеризувався зміною технологічного укладу та архітектурних парадигм:

- 1. Ера механізації та аналогової автоматики (перша половина ХХ ст.).** Поява перших побутових приладів, оснащених локальними реле, біметалевими термостатами та механічними таймерами. Взаємодія між пристроями була повністю відсутня, а автоматизація обмежувалася жорстко заданими внутрішніми алгоритмами окремого приладу.
- 2. Ера перших мережевих протоколів та дротової комунікації (1970–1990-ті роки).** Точкою відліку вважається 1975 рік, коли шотландська компанія *Pico Electronics* розробила технологію **X10**. Цей стандарт став революційним, оскільки запропонував використовувати наявну побутову електропроводку (Power Line Communication, PLC) як середовище передачі даних. Цифрові сигнали інжектувалися в мережу у вигляді високочастотних сплесків (120 кГц тривалістю 1 мс) у моменти переходу синусоїдальної напруги частотою 60 Гц (або 50 Гц) через нульове значення. Проте технологія X10 мала критичні інженерні недоліки: швидкість передачі не перевищувала 60 біт/с, система страждала від високого рівня гармонійних завад в електромережі, була схильна до взаємного

блокування пакетів, помилкових спрацьовувань та не мала механізмів апаратного шифрування. Наприкінці 1980-х років компанії на кшталт *Leviton* модернізували імпульсні фільтри та компоненти обв'язки PLC-інтерфейсів, але це не вирішило проблему низької пропускної здатності.

3. **Ера спеціалізованих промислових шин (кінець 1990-х – 2010-ті роки).** У відповідь на ненадійність PLC-мереж європейські виробники консорціуму EIBA (на базі рішень *Instabus* від *Siemens*) розробили стандарт **KNX** (децентралізована дротова шина типу «кручена пара»). Це дозволило гарантувати доставку пакетів та забезпечити високу завадостійкість, але вимагало прокладання окремого інформаційного кабелю до кожного вимикача, сенсора та реле.
4. **Сучасна ера Інтернету речей (IoT) та кіберфізичних систем (з 2015 року по теперішній час).** Характеризується стрімким проникненням бездротових технологій та однокристальних систем (System-on-Chip, SoC). Масова поява високоінтегрованих мікроконтролерів із підтримкою Wi-Fi та Bluetooth (зокрема сімейства ESP32 від *Espressif Systems*) докорінно змінила ринок. Сучасний «Розумний будинок» перестав бути просто набором реле; сьогодні це інтелектуальна екосистема, інтегрована з хмарними обчисленнями (Cloud Computing), технологіями штучного інтелекту (AIoT) та алгоритмами машинного навчання (Machine Learning), які аналізують поведінкові патерни користувача і здійснюють проактивне (випереджувальне) керування об'єктом.

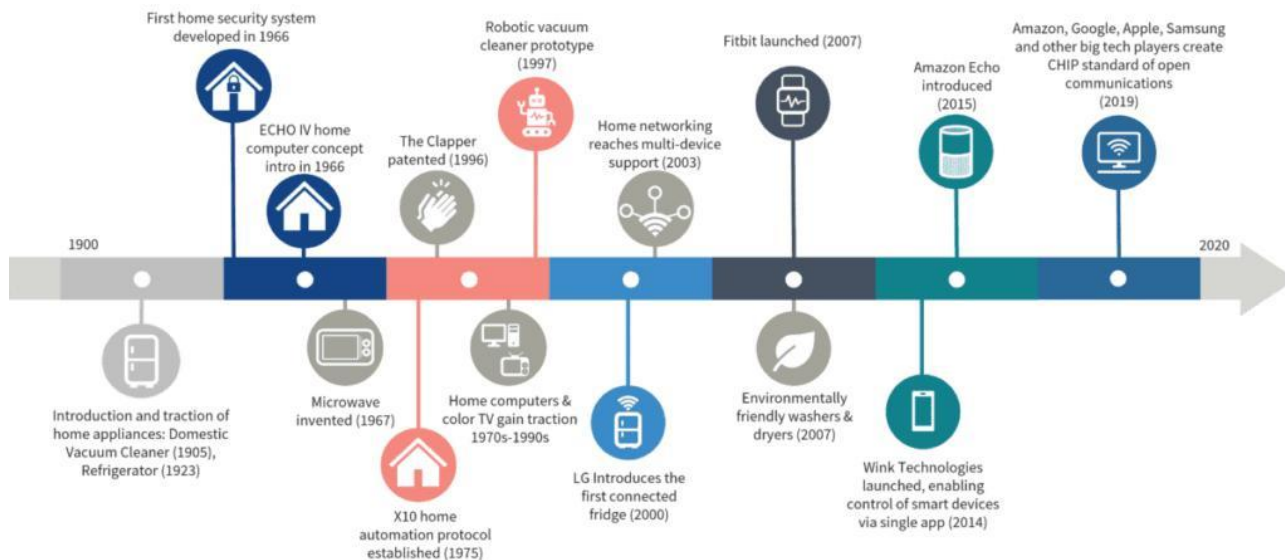


Рис. 1.1. Основні етапи розвитку технологій автоматизації житлового простору

1.1.2. Принципи об'єднання інженерних підсистем у єдиний комплекс керування

В основі проектування сучасних IoT-рішень для житлової автоматизації лежить принцип системного аналізу та декомпозиції. Функціонування комплексу базується на інтеграції програмних і апаратних компонентів у єдину трирівневу ієрархічну структуру, де координується робота різноманітних інженерних систем. Основним інженерним завданням є перехід від ізольованих контурів автоматики до наскрізних сценаріїв взаємодії. Об'єднання підсистем реалізується на таких концептуальних рівнях:

- **Рівень сприйняття (Сенсорний рівень).** Складається з гетерогенного масиву первинних перетворювачів (датчиків), які безперервно здійснюють збір телеметричних даних про фізичні параметри внутрішнього та зовнішнього середовища. Інформаційні сигнали від датчиків можуть надходити у дискретному (TTL-рівні), аналоговому (зміна напруги або струму, що потребує обробки АЦП) або цифровому послідовному вигляді (протоколи I2C, SPI, 1-Wire).
- **Комунікаційно-мережевий рівень.** Забезпечує транспортування інформаційних пакетів від сенсорних вузлів до обчислювального ядра та трансляцію команд на виконавчі механізми. Для мінімізації блукаючих потенціалів та захисту від помилок комутації у

навчально-лабораторних та побутових умовах доцільно використовувати надійні стандартизовані інтерфейси (наприклад, чотириконтактні роз'єми серії EASY Plug RJ11). Мережевий обмін здійснюється як через локальні бездротові мережі стандарту IEEE 802.11 (Wi-Fi) чи Bluetooth Low Energy (BLE), так і за допомогою легковагих протоколів прикладного рівня, оптимізованих для IoT, зокрема **MQTT** (Message Queuing Telemetry Transport) та HTTP REST API.

- **Прикладний рівень (Обчислювальний центр).** Центральний контролер (обчислювальне ядро) виконує роль координатора. Отримавши телеметрію, програмна логіка (написана, наприклад, на базі MicroPython або freeRTOS) здійснює поточний системний аналіз умов, перевіряє стан внутрішніх тригерів і таймерів, після чого ухвалює рішення щодо зміни параметрів внутрішнього середовища за допомогою однієї комплексної команди. У результаті значно спрощується управління житлом і зникає необхідність використання великої кількості окремих пультів, вимикачів та локальних контролерів.

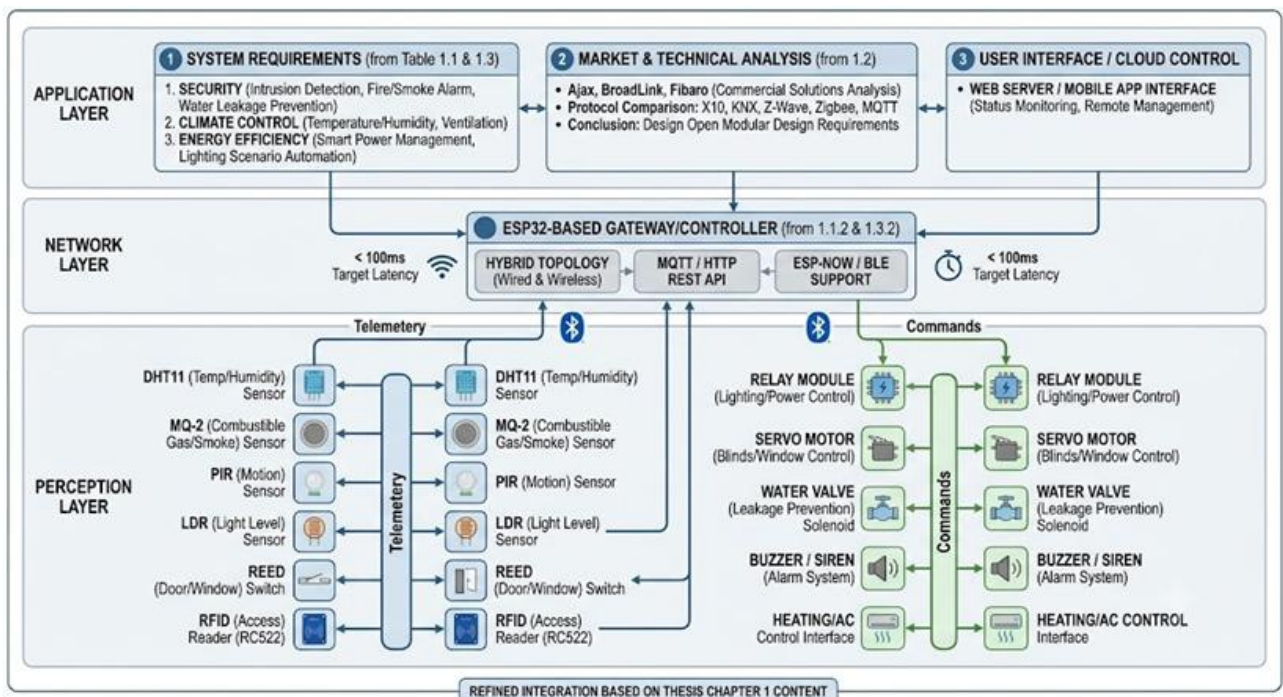


Рис. 1.2. Трирівнева структура інтегрованої системи керування інженерними підсистемами житла

1.1.3. Функціональні можливості розумного житла: енергоефективність, мікроклімат, безпека та комфорт

Сучасні архітектурні рішення Smart Home забезпечують покриття чотирьох життєво важливих доменів житлового простору:

1. Інтелектуальне регулювання температурного режиму та мікроклімату. Принцип роботи полягає у підтриманні оптимальних параметрів мікроклімату залежно від функціонального призначення приміщень та умов їх використання мешканцями. Контролер за допомогою датчиків (наприклад, типу DHT11) безперервно зчитує показники відносної вологості повітря та температури. Завдяки автоматичному керуванню системами опалення, вентиляції та кондиціонування (HVAC) можна суттєво скоротити витрати електричної енергії та водних ресурсів, не знижуючи при цьому рівень комфорту мешканців. У приміщеннях, які тимчасово не використовуються, або під час відсутності людей, зафіксованої датчиками руху, система автоматично знижує інтенсивність роботи опалювального або охолоджувального обладнання, переводячи його в режим «Еко». Користувач має змогу в будь-який момент отримати актуальну інформацію про стан мікроклімату через локальний або віддалений графічний інтерфейс вебсервера.

2. Енергоефективність та раціональне керування освітленням. Одним із найбільш ефективних інженерних способів енергозбереження є централізоване керування освітленням, яке здійснюється відповідно до заздалегідь сформованих графіків роботи або показань сенсорів. Значного ефекту досягають шляхом максимального використання природного інсоляційного ресурсу приміщень. Для цього віконні конструкції обладнуються автоматизованими жалюзі, ролетами або шторами з електроприводом (сервоприводами), які змінюють своє кутове положення залежно від рівня зовнішньої освітленості. Широке застосування датчиків руху та присутності (наприклад, на базі пасивних інфрачервоних PIR-елементів та лінз Френеля) дозволяє повністю ліквідувати неефективне споживання електрики порожніми кімнатами. Датчики освітленості аналізують рівень штучного і природного світла, забезпечуючи плавне

регулювання (димування) джерел світла, що не тільки економить енергію, а й продовжує термін експлуатації ламп.

3. Комплексна інженерно-технологічна та охоронна безпека. Концепція автоматизації дозволяє своєчасно виявляти потенційні загрози та оперативно реагувати на надзвичайні ситуації. Функціонал безпеки розділяється на два контури:

- ***Охоронний контур.*** Контроль периметра об'єкта, обмеження та санкціонування доступу до приміщень (наприклад, на основі безконтактних високочастотних RFID-технологій та зчитувачів типу RC522), запуск світлозвукових засобів оповіщення (п'єзоелектричних зумерів, сирен) та організація IP-відеоспостереження. Власник має можливість віддалено отримувати зображення з камер та лог доступу через мережу Інтернет незалежно від геопозиції. Важливою функцією є також імітація присутності мешканців під час їхньої тривалої відсутності: система автоматично вмикає та вимикає освітлення у різних кімнатах за певним псевдовипадковим алгоритмом, знижуючи ризик несанкціонованого проникнення.
- ***Технологічний контур (Запобігання аваріям).*** Виявлення ознак займання або задимлення за допомогою напівпровідникових газорезистивних датчиків (типу MQ-2), які аналізують концентрацію горючих і чадних газів (CO, CH, пропан-бутан) у повітрі. Не менш важливим є захист від аварій, пов'язаних із витокami води, що здатні завдати значних матеріальних збитків. Спеціалізовані системи контролю протікань на базі датчиків водяної пари (Steam Sensor) або кондуктометричних сенсорів наявності води встановлюються у зонах підвищеного ризику (біля пральних машин, змішувачів, трубопроводів). При фіксації вологи сигнал надходить до центрального контролера, який активує виконавчі механізми (електромагнітні клапани або сервоприводи) для автоматичного перекриття подачі води на вході в будівлю, локалізуючи аварію без антропогенного втручання.

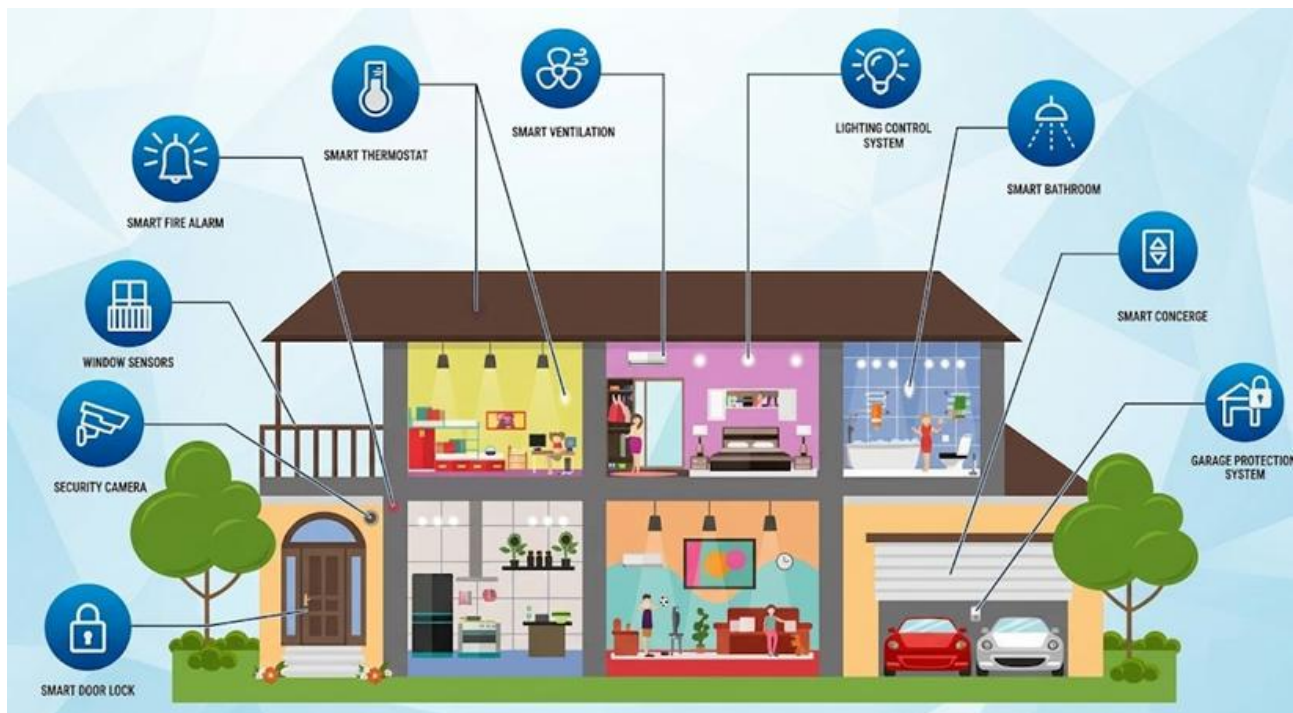


Рис. 1.3. Загальна схема впровадження технології «Розумний будинок»

1.1.4. Аналіз переваг та інженерних складнощів впровадження автоматизованих рішень

Незважаючи на високий рівень технологічної зрілості, масове інтегрування систем домашньої автоматизації Smart Home стримується низкою вагомих факторів, які потребують детального інженерного аналізу та пошуку компромісних рішень.

Для системного аналізу факторів, що впливають на впровадження технологій домашньої автоматизації, у таблиці 1.1 наведено порівняльну характеристику їхніх інженерних переваг та експлуатаційних викликів.

Таблиця 1.1

Порівняльна характеристика інженерних переваг та експлуатаційних викликів

Критерій аналізу	Інженерні переваги впровадження системи	Технічні складнощі та недоліки (Виклики)
Економічна ефективність	Скорочення витрат на комунальні послуги (електроенергія, газ, тепло) до 30% за	Висока початкова вартість обладнання (контролерів, сенсорів, актуаторів), тривалий

Критерій аналізу	Інженерні переваги впровадження системи	Технічні складнощі та недоліки (Виклики)
	рахунок алгоритмічного керування та переведення систем у режим «Еко».	термін окупності та значні витрати на сервісне обслуговування.
Технологічна безпека	Миттєва локалізація аварій (перекриття води при протіканні, відключення газу при витоку, знеструмлення мережі при задимленні) без участі людини.	Залежність надійності від якості резервного живлення та ризик хибних спрацьовувань датчиків через електромагнітні завади.
Інтеграція в інфраструктуру (Retrofitting)	Безперешкодне проектування дротових магістралей «з нуля» на етапі будівництва, що гарантує максимальну стабільність зв'язку.	Складність розгортання у готових приміщеннях. Дротові рішення вимагають руйнування оздоблення (шротування стін), а бездротові — обмежені радіусом дії.
Комунікаційна стабільність	Використання сучасних бездротових стандартів (Wi-Fi, Zigbee, Bluetooth BLE) дозволяє легко масштабувати систему та додавати нові вузли.	Висока завадозалежність бездротових каналів у багатоквартирних будинках (перевантаження частоти 2.4 ГГц), затухання сигналу залізобетонними стінами.

Критерій аналізу	Інженерні переваги впровадження системи	Технічні складнощі та недоліки (Виклики)
Апаратна сумісність	Сучасні відкриті платформи (на кшталт Home Assistant чи Node-RED) дозволяють об'єднувати пристрої різних брендів через програмні шлюзи.	Фрагментація ринку. Більшість комерційних лідерів (Аjax, Fibaro тощо) використовують пропрієтарні закриті протоколи, що блокує пряму сумісність.
Кібербезпека та автономність	Можливість глобального моніторингу об'єкта, отримання Push-повідомлень про тривоги та віддаленого керування через хмарні сервери.	Уразливість до хакерських атак, ризик витоку приватних даних. Критична залежність багатьох систем від наявності Інтернету та сторонніх хмар.

Зазначені інженерні складнощі та недоліки (зокрема висока вартість обладнання та апаратна закритість комерційних платформ) роблять розробку відкритих модульних систем автоматизації на базі доступних енергоефективних контролерів (таких як ESP32) та інтерпретованих мов програмування високого рівня (MicroPython) вкрай актуальним завданням для сучасної інженерії програмного забезпечення. Це дозволяє створювати гнучкі прототипи систем (Rapid Prototyping) із мінімальною собівартістю та повною автономністю від сторонніх хмарних провайдерів.

1.2. Порівняльний аналіз комерційних рішень та мережевих протоколів автоматизації будівель

Комерційні системи домашньої автоматизації використовуються приблизно з 1975 року в Сполучених Штатах. Новатором в цій області стала компанія Pico Electronics, яка створила протокол зв'язку X10. Надалі ринок

поповнився іншими компаніями власними протоколами зв'язку [4]. Розглянемо найбільш значущі з них. Протокол і стандарт зв'язку X10 є найстарішим і найпопулярнішим в області систем автоматизації. Він був розроблений в 1975 році компанією Pico Electronics для управління домашніми електроприладами. Технологія X10 заснована на передачі сигналів по електропроводці будинку. Для передачі сигналів використовуються коливання на частоті 120 кГц тривалістю 1 мс. Спочатку технологія працювала на частоті 60 Гц і мережевій напрузі 110 В і використовувалася тільки в США. X10 допомагає вирішувати найрізноманітніші завдання домашньої автоматизації, починаючи від управління освітлювальними приладами і закінчуючи системою безпеки. пристрою до виконавчого механізму немає необхідності прокладати нову проводку або використовувати дорогі датчики і контролери, оснащені бездротовим інтерфейсом. Управління електроприладом здійснюється безпосередньо, завдяки передавачам, які генерують і відправляють команди X10 в електромережу.

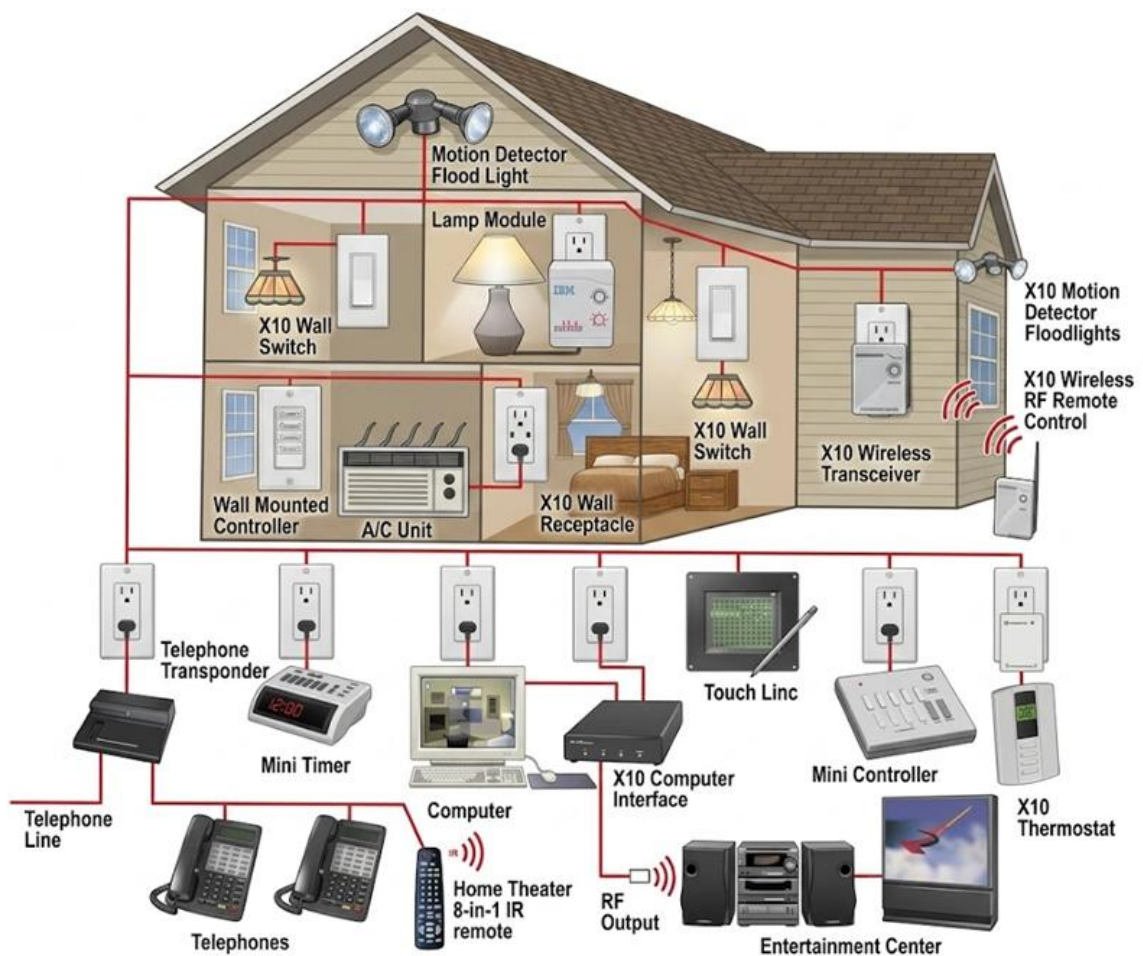


Рис. 1.4. Технологія X10

З іншого боку, такий підхід має багато недоліків, таких як:

- низька швидкість передачі даних;
- низька перешкодозахищеність;
- проблема помилкових спрацьовувань;
- між пристроями X10 різних виробників можуть виникати конфлікти;
- можливий несанкціонований доступ до пристроїв X10 по електромережі.

Z-Wave - бездротовий протокол передачі даних для домашньої автоматизації (рис. 1.5). Ця технологія покликана забезпечити простий і надійний спосіб бездротового управління освітленням, вентиляцією, системами безпеки, домашніми кінотеатрами, автоматичними воротами і органами управління.



Рис. 1.5. Технологія Z-Wave

У 2005 році компанія Zensys створила Z-Wave Alliance, в який увійшли різні компанії. На сьогоднішній день існують сотні сумісних продуктів Z-Wave, що продаються під різними брендами. Технологія Z-Wave призначена для забезпечення надійної і швидкої передачі простих команд управління з низькою

затримкою на швидкості до 100 кбіт / с. Z-Wave працює в діапазоні частот до 1 ГГц, це пов'язано з малою кількістю потенційних джерел перешкод. Z-Wave використовує архітектуру чарункової мережі. Пристрої можуть спілкуватися між собою за допомогою проміжних вузлів, це дозволяє обходити побутові перешкоди або «мертві зони». Найпростіша мережа являє собою один керований пристрій і первинний контролер. Додаткові пристрої можна додати в будь-який час. Мережа Z-Wave може включати до 232 приладів [5]. KNX - це стандартизований протокол зв'язку для автоматизації будівель. Продукція KNX поширювалася під кількома торговими марками. Найвідомішими є Instabus, ABB i-Bus, Tebis, Theben.

KNX Home Automation Solution

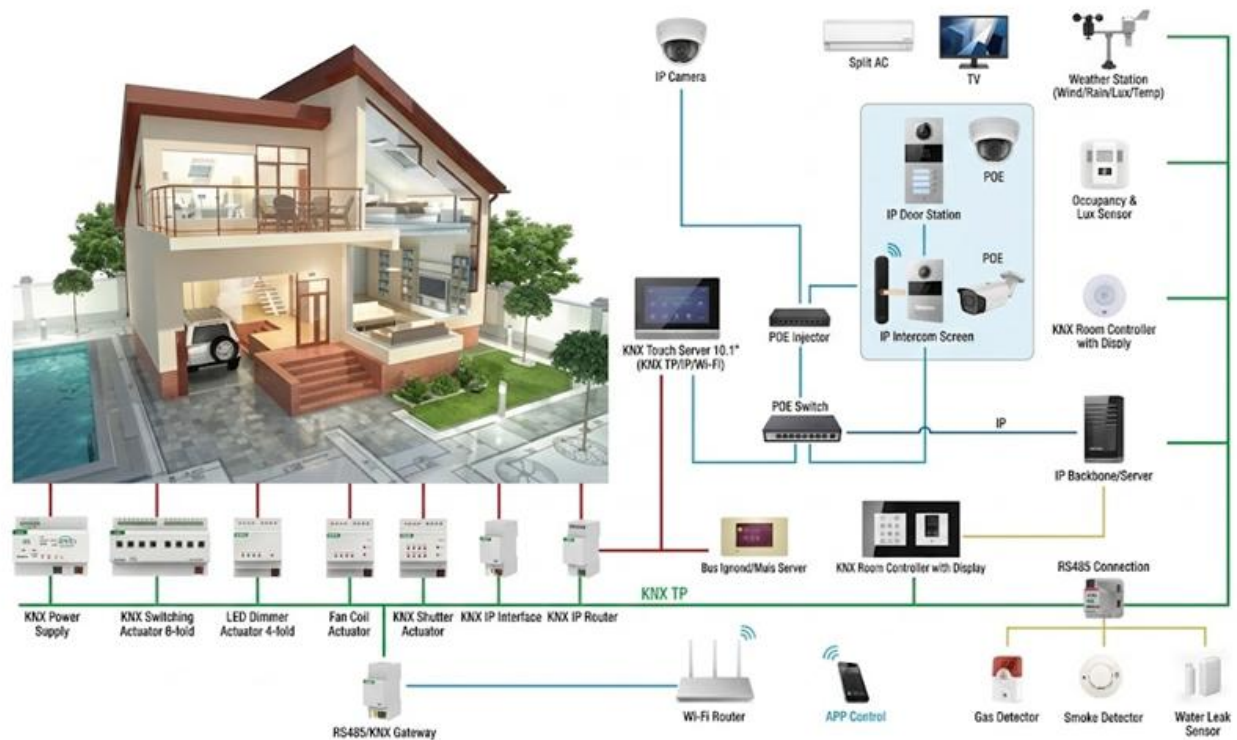


Рис. 1.6. Технологія KNX

Всі пристрої, що працюють за технологією KNX, з'єднані між собою двопровідною шиною, що дозволяє їм обмінюватися даними. Функції окремих пристроїв визначаються ще на етапі планування проекту, але можуть бути змінені і адаптовані в будь-який момент. Устаткування KNX включає в себе наступні типи пристроїв: датчики; диски; системні пристрої та компоненти [6]. Датчики є відправною точкою для кожної дії. Вони відповідають за запис певних зовнішніх подій. Виникнення тієї чи іншої події має викликати певний відгук з

боку системи. Після настання такої події датчик посилає команду управління по мережі на відповідний виконавчий механізм. Виконавчі механізми отримують пакети даних, які потім перетворюються в дії. Такими діями можуть бути управління жалюзі, вимкнення світла або управління системою опалення та кондиціонування. KNX - це повністю розподілена мережа, яка вміщує до 65536 пристроїв в 16 бітах індивідуального адресного простору. Логічна топологія дозволяє підключати до 256 пристроїв на одній лінії.

Однією з сильних сторін системи KNX є те, що будь-який продукт, розроблений різними компаніями, але при цьому має сертифікацію KNX, може працювати разом. Члени асоціації мають понад 7000 сертифікованих продуктів KNX. Такий широкий асортимент продукції дозволяє: - керувати освітленням; - керувати опалювальними установками і системами кондиціонування; - аналізувати сигнали тривоги; - керувати енергоспоживанням природних ресурсів. Основним недоліком комерційних рішень, що використовують власний протокол зв'язку, є їх секретність (закрита архітектура), тобто використовуються протоколи передачі даних з фірмовим кодуванням, які не завжди дозволяють інтегрувати чуже обладнання в єдину систему [7, 8].

Сучасні технології створюють нові можливості для оптимізації життєвого простору. Одним із яскравих прикладів є системи "розумний будинок", які об'єднують різноманітні пристрої та системи в єдину інтегровану платформу, забезпечуючи автоматизацію та управління різними функціями в житловому приміщенні.

Залежно від рівня інтеграції та набору функцій, системи "розумний будинок" можна поділити на такі категорії:

Базові системи: забезпечують управління окремими пристроями (освітлення, опалення) та моніторинг основних параметрів (температура, вологість).

Системи середнього рівня: об'єднують різні підсистеми (освітлення, безпека, мультимедіа) в єдину платформу, забезпечуючи більш складні сценарії автоматизації.

Інтелектуальні системи: здатні до самонавчання та адаптації до змін умов навколишнього середовища та поведінки користувачів.

Сучасні системи "розумний будинок" пропонують широкий спектр функцій, які значно підвищують комфорт та безпеку проживання:

1. Управління мікрокліматом: автоматичне регулювання температури, вологості, освітлення залежно від часу доби, погоди та особистих переваг користувачів.
2. Безпека: системи відеоспостереження, датчики руху, диму, витоку газу, контролю доступу забезпечують захист житла та його мешканців.
3. Енергоефективність: оптимізація споживання енергії за рахунок автоматичного вимкнення приладів у відсутність людей, регулювання освітлення та опалення.
4. Комфорт: створення комфортного житлового середовища за допомогою інтеграції музичних систем, систем "розумний дім" можуть керувати музикою, телебаченням та іншими розважальними системами.
5. Доступність: можливість віддаленого управління системою за допомогою смартфонів, планшетів та інших мобільних пристроїв.

Реалізація функціоналу систем "розумний будинок" стала можливою завдяки розвитку таких технологій:

Мікроконтролери: Arduino, Raspberry Pi та інші платформи, які слугують основою для створення розумних пристроїв.

Датчики: температури, вологості, руху, світла, якості повітря та інші, які збирають інформацію про стан навколишнього середовища.

Виконавчі механізми: електроприводи, реле, клапани, які керують роботою різних пристроїв у будинку.

Мережі: Wi-Fi, Bluetooth, Zigbee, Ethernet, які забезпечують обмін даними між пристроями системи.

Програмне забезпечення: спеціалізовані програмні платформи та мови програмування для розробки та конфігурування систем "розумний будинок".

Системи "розумний будинок" відкривають нові можливості для створення комфортного, безпечного та енергоефективного житла. Постійний розвиток технологій сприяє розширенню функціоналу та зниженню вартості таких систем, що робить їх доступними для широкого кола споживачів.

Система Ajax є одним із популярних рішень на ринку систем розумного будинку. Її основні переваги полягають у:

- високій надійності. Завдяки використанню двостороннього радіозв'язку Jeweller забезпечується стабільна та захищена передача даних між пристроями системи.
- простоті установки. Система легко встановлюється без необхідності проведення складних монтажних робіт.
- гнучкості. Можливість підключення великої кількості різних датчиків та виконавчих механізмів дозволяє адаптувати систему до індивідуальних потреб користувача.
- автономності. Наявність резервного живлення у хабі забезпечує безперебійну роботу системи навіть при відключенні електроенергії.
- мобільності. Керування системою здійснюється за допомогою мобільного додатка, що дозволяє контролювати свій будинок з будь-якої точки світу.

Однак, система Ajax, як і будь-яка інша, має свої особливості:

1. Працездатність датчиків безпосередньо залежить від роботи хаба, що може обмежувати можливості системи в разі виходу з ладу центрального вузла.
2. Для реалізації функцій відеоспостереження необхідно додатково підключати IP-камери.
3. Хоча управління системою через мобільний додаток є зручним, деякі користувачі можуть віддавати перевагу традиційним способам управління.



Рис. 1.7. Розумний будинок Ајах

Заснована на передових технологіях, система Ајах пропонує користувачам широкий спектр функцій для забезпечення безпеки та комфорту в домі. Її компактний дизайн та інтуїтивно зрозумілий інтерфейс роблять систему зручною у використанні. Завдяки цим якостям, Ајах є одним з найпопулярніших рішень на ринку систем "розумний будинок" [2].

Система BroadLink представляє собою універсальне рішення для автоматизації житлового простору. На відміну від спеціалізованих систем, таких як Ајах, орієнтованих на безпеку, BroadLink пропонує ширший спектр можливостей для управління різноманітними приладами та системами в будинку.

Ключові переваги системи BroadLink:

1. Гнучкість. Система підтримує широкий спектр протоколів зв'язку (IR, RF, Wi-Fi), що дозволяє інтегрувати в неї практично будь-який побутовий прилад.
2. Автономна робота датчиків. Кожен датчик працює незалежно, що забезпечує високу надійність системи.
3. Простота установки та налаштування. Інтуїтивно зрозумілий інтерфейс мобільного додатку дозволяє легко налаштувати систему без спеціальних знань.

4. Доступна вартість. Система BroadLink є одним з більш бюджетних варіантів на ринку систем "розумний будинок".

Однак, система має і певні обмеження. Сигнал системи може діяти на відстані до 50 метрів, що може бути недостатньо для великих приміщень. Відсутність хаба може ускладнювати управління системою в цілому. Система BroadLink в основному орієнтована на автоматизацію, а не на забезпечення високого рівня безпеки.

Займаючи впевнене друге місце в рейтингу систем "розумний будинок", BroadLink є одним з найбільш популярних рішень на ринку. Незважаючи на деякі недоліки, її висока функціональність, доступність та простота використання роблять її гідним конкурентом більш дорогим системам (рис. 1.8).



Рис. 1.8. Розумний будинок BroadLink

Система "Розумний будинок" Fibaro - це високотехнологічне рішення для автоматизації та безпеки житлових приміщень, яке вимагає професійної інсталяції. Завдяки широкому спектру функцій та можливостей налаштування, система Fibaro дозволяє створити індивідуальне та комфортабельне житлове середовище (рис. 1.9).



Рис. 1.9. Розумний будинок Fibaro

Система "Розумний дім" Fibaro пропонує широкий спектр можливостей для автоматизації житлового простору. Завдяки підтримці протоколу Z-Wave і різноманітним датчикам, вона дозволяє створити індивідуальне та комфортабельне середовище. Однак, висока вартість обладнання та необхідність професійного монтажу роблять її менш доступною для широкого кола користувачів [4].

Таблиця 1.2

Порівняльна таблиця систем "розумний будинок" Ajax, BroadLink, Fibaro

Характеристика	Ajax	BroadLink	Fibaro
Спеціалізація	Безпека, охорона, автоматизація	Універсальна автоматизація	Комплексна автоматизація будинку
Протоколи зв'язку	Jeweller (власний), Wi-Fi	IR, RF, Wi-Fi	Z-Wave
Центральний блок	Хаб Hub	Різноманітні контролери (RM Pro, Mini, etc.)	Хаб Home Center
Датчики та актуатори	Широкий асортимент датчиків (руху, відкриття дверей,	Різноманітні датчики та пульти дистанційного керування для	Широкий спектр датчиків та актуаторів для управління

Характеристика	Ajax	BroadLink	Fibaro
	витоку води тощо), сирени, реле	управління побутовою технікою	освітленням, опаленням, безпекою тощо
Мобільний додаток	Ajax Security System (iOS, Android)	BroadLink (iOS, Android)	Fibaro Home Center (iOS, Android)
Голосові помічники	Google Assistant, Amazon Alexa	Google Assistant, Amazon Alexa	Google Assistant, Amazon Alexa
Автономність роботи	Висока завдяки резервному живленню	Залежить від моделі пристрою	Висока завдяки вбудованому акумулятору в деяких пристроях
Вартість	Середня	Доступна	Вища
Складність встановлення	Відносно проста	Проста	Вимагає професійного монтажу
Функціональність	Орієнтована на безпеку та автоматизацію	Універсальна автоматизація, управління побутовою технікою	Комплексна автоматизація будинку, інтеграція з іншими системами
Масштабованість	Легко розширюється новими пристроями	Легко розширюється новими пристроями	Можливість створення складних сценаріїв автоматизації

1.3. Формування технічних вимог та концептуальної архітектури розроблюваної системи «Розумний будинок»

Проектування та програмно-апаратна реалізація сучасних кіберфізичних систем, що функціонують у межах парадигми Інтернету речей (IoT), вимагає глибокого попереднього аналізу, чіткого формулювання технічних вимог та деталізації концептуальної архітектури. Цей етап є фундаментальним у системній інженерії, оскільки він визначає межі функціональності системи, її топологію, вибір інтерфейсів взаємодії, алгоритмічну складність програмного забезпечення та підходи до забезпечення надійності й відмовостійкості.

Оскільки розроблювана система проєктована як комплексний лабораторний стенд на базі мікроконтролера ESP32 під управлінням мови MicroPython, формування архітектури має враховувати як прикладні побутові

сценарії автоматизації (кліматичні, безпекові, комунікаційні), так і дидактичну цінність стенду для дослідження різних режимів бездротового зв'язку та методів обробки сигналів.

Технічні вимоги до системи розподіляються на дві ключові категорії: функціональні (що саме повинна виконувати кожна підсистема) та системно-технічні й експлуатаційні (яким критеріям надійності, енергоефективності та масштабованості має відповідати апаратно-програмний комплекс).

1. Функціональні вимоги до підсистем стенду:

- Підсистема санкціонованого доступу та безпеки периметра. Система повинна забезпечувати безконтактне зчитування унікальних ідентифікаційних кодів (UID) з пасивних радіочастотних міток-транспондерів (карт чи брелоків). У разі успішної валідації коду у внутрішній базі даних мікроконтролер має ініціювати кутове переміщення сервоприводу для фізичного розблокування дверного макета приміщення. У випадку спроби доступу за допомогою незареєстрованого ключа система повинна блокувати привід та активувати тривожний контур.
- Автоматизована підсистема мікроклімату та терморегулювання. Необхідно забезпечити безперервний циклічний моніторинг показників температури та відносної вологості повітря всередині контрольованого простору. Програмна логіка повинна підтримувати задані температурні межі та автоматично замикати силові лінії драйвера двигуна для активації примусової вентиляції при перевищенні порогових значень, з паралельним виведенням поточної телеметрії на локальний рідкокристалічний або OLED-дисплей.
- Підсистема газового аналізу, протипожежної безпеки та аварійного оповіщення. Датчики повинні здійснювати постійний моніторинг повітряного середовища на предмет виявлення небезпечних концентрацій горючих газів (пропану, бутану, метану), чадного газу

та часток диму. При досягненні критичного рівня концентрації система зобов'язана миттєво переходити в режим тривоги, запускати світлозвукову сирену, виводити попередження на екран та транслювати екстрені пакети даних у мережу.

- Підсистема дистанційного мережевого керування. Мікроконтролер має підтримувати розгортання автономного мережевого сервісу, який надає користувачеві можливість кросплатформного віддаленого моніторингу датчиків та інтерактивного керування всіма наявними виконавчими механізмами через графічний інтерфейс звичайного веб-браузера без залучення сторонніх хмарних шлюзів.

2. Системно-технічні вимоги:

- Час відгуку (Latency). Час реакції системи на критичні події (виявлення руху в режимі охорони, фіксація задимлення чи витоку газу) має наближатися до режиму реального часу і не перевищувати частоту дискретизації й обробки переривань у кілька мікросекунд.
- Узгодження логічних рівнів. Усі компоненти системи повинні бути електрично сумісними з GPIO-інтерфейсами обчислювального ядра (3.3V TTL), а для модулів із 5-вольтовим живленням та виходом (наприклад, аналогові лінії деяких сенсорів) мають бути передбачені схеми захисту та дільники напруги.
- Модульність та ремонтпридатність. Конструкція стенду має передбачати швидку заміну або додавання нових периферійних вузлів без необхідності повної переробки друкованої плати чи архітектури програмного забезпечення.

Концептуальна архітектура системи.

Концептуальна архітектура розроблюваного лабораторного стенду базується на класичній тривірневій моделі побудови IoT-рішень: **Рівень сприйняття (сенсорний) → Мережевий рівень (обробка та комунікації) → Прикладний рівень (інтерфейс користувача).**

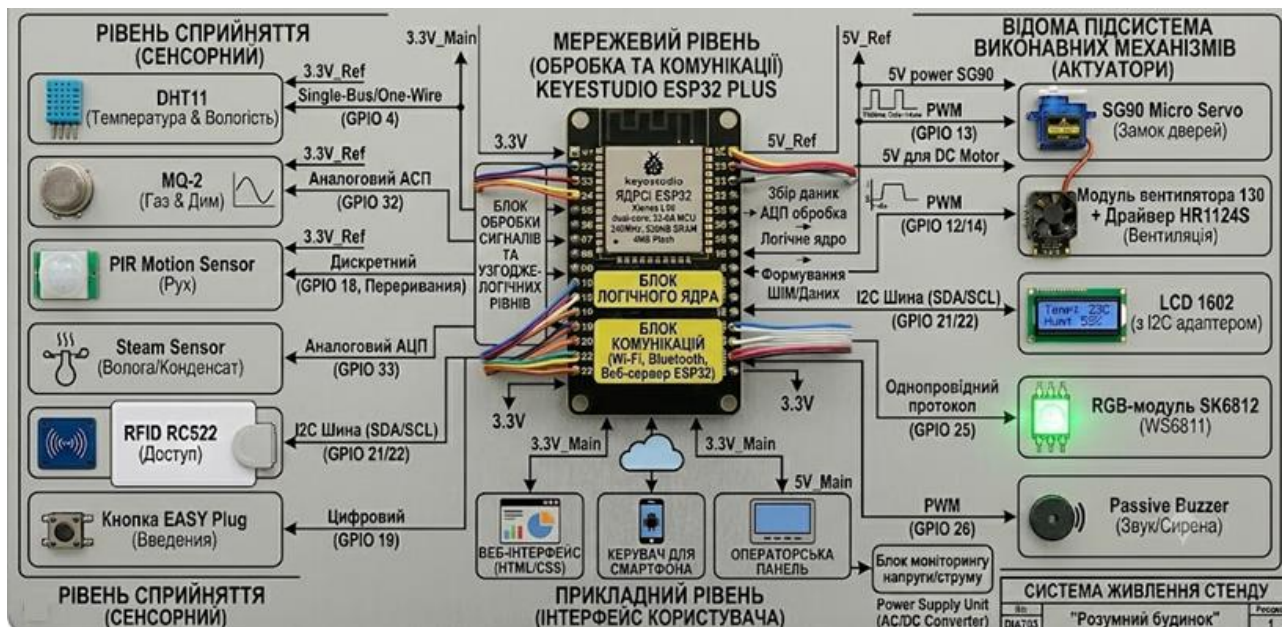


Рис. 1.10. Концептуальна архітектура системи

Така структура забезпечує чітке розділення обов'язків між апаратними модулями, спрощує написання скриптів мовою MicroPython та гарантує стабільний інформаційний обмін.

1. Центральний обчислювальний модуль системи (Рівень обробки даних).

Ядром концептуальної архітектури виступає розробка сумісної з Arduino плати KEYESTUDIO ESP32 PLUS, оснащеної високопродуктивним однокристальним двоядерним 32-бітним процесором *Xtensa LX6* (модуль *ESP32-WROOM-32*). Обчислювальне ядро працює на тактовій частоті до 240 МГц, має 520 КБ вбудованої оперативної пам'яті SRAM та 4 МБ Flash-пам'яті, що є достатнім для локального збереження прошивки, підключених бібліотек периферії та текстових файлів веб-сервера.

Використання операційного середовища MicroPython дозволяє реалізувати багатозадачність, паралельно виконуючи опитування сенсорів, підтримку мережевого сокету, обробку логічних скриптів автоматизації та виведення сигналів на актуатори без взаємного блокування потоків виконання.

2. Підсистема сенсорів та збору даних (Рівень сприйняття)

Цей блок відповідає за отримання первинної інформації про фізичний стан макета розумного будинку:

- **DHT11:** використовує резистивний датчик вологості та термістор для вимірювання мікроклімату, передаючи дані у вигляді 40-бітного послідовного пакету за власним 1-Wire часозалежним інтерфейсом.
- **MQ-2:** напівпровідниковий газорезистивний вузол, що виявляє присутність молекул палива або диму через зміну опору шару діоксиду олова (SnO) при нагріванні вбудованим ТЕНОм. Передає аналоговий рівень напруги на АЦП (ADC) мікроконтролера.
- **PIR Motion Sensor:** фіксує теплові інфрачервоні коливання від об'єктів за допомогою лінзи Френеля та піроелемента, формуючи на виході дискретний TTL-сигнал, який обробляється через механізм зовнішніх апаратних переривань ESP32.
- **Steam Sensor:** вимірює наявність пари чи крапель води на базі павутиноподібної геометрії відкритих провідників, переводячи зміну провідності середовища в аналоговий сигнал.
- **RFID RC522 (I2C/SPI варіант):** інтегрує високочастотний генератор та петльову антену на частоті 13.56 МГц, зчитуючи UID-коди пасивних міток за рахунок явища взаємної електромагнітної індукції та зворотного амплітудного моделювання.
- **Модуль цифрової кнопки EASY Plug RJ11:** тактильний елемент інтерфейсу користувача з апаратною Pull-up підтяжкою на 10 кОм, підключений через надійний фіксований роз'єм RJ11 для унеможливлення блукаючих потенціалів.

3. Підсистема виконавчих механізмів (Рівень актуаторів)

Блок призначений для реалізації фізичної відповіді системи на внутрішні тригери:

- **Мікросервопривід SG90:** отримує ШІМ-сигнал (PWM) частотою 50 Гц від мікроконтролера, де тривалість високого рівня імпульсу від 0.5 до 2.5 мс чітко задає кут повороту вихідного вала від 0° до 180°, забезпечуючи імітацію відчинення замка.

- **Модуль двигуна постійного струму (вентилятор):** використовує силовий мікрочіп-драйвер Н-моста *HR1124S* з PMOS/NMOS ключами низького опору, що дозволяє безпечно керувати швидкістю обертання лопатей вентилятора через ШІМ-модуляцію без ризику перевантаження GPIO портів ESP32.
- **Рідкокристалічний дисплей LCD 1602:** символічний екран 16x2 з інтегрованим I2C-адаптером (PCF8574), який перетворює послідовні дані шини (лінії SDA, SCL) у паралельний інтерфейс керування дисплеєм, заощаджуючи виводи контролера.
- **RGB-модуль SK6812:** розумна піксельна матриця, що використовує однопровідний протокол зв'язку, де кожні наступні 24 біти даних кольору (по 8 біт на Red, Green, Blue) фіксуються внутрішнім ШІМ-драйвером світлодіода, дозволяючи відображати до 16,7 млн відтінків.
- **Passive Buzzer Module:** п'єзоелектричний випромінювач, позбавлений вбудованого генератора, звук у якому виникає виключно за рахунок програмного формування хвиль квадратної форми (Square Wave) у заданому частотному діапазоні від 200 до 5000 Гц.

4. Комунікаційний та прикладний рівень (Рівень взаємодії)

Мережева складова системи повністю базується на інтегрованих бездротових можливостях KEYESTUDIO ESP32 PLUS. Залежно від мети експериментів, архітектура дозволяє конфігурувати три режими взаємодії:

- **Режим станції (STA):** плата реєструється в домашній Wi-Fi мережі як стандартний бездротовий клієнт і отримує внутрішню IP-адресу від DHCP-сервера роутера. Це дозволяє будь-якому іншому локальному пристрою (ноутбуку, смартфону) надсилати HTTP-запити на плату.
- **Режим точки доступу (softAP):** мікроконтролер самостійно генерує бездротову мережу (наприклад, з SSID «ESP32_Wifi») та шифруванням WPA2, очікуючи на пряме підключення гаджетів користувача, що гарантує автономність стенду навіть за відсутності зовнішнього інтернету чи маршрутизатора.

- **Гібридний режим (STA+AP):** забезпечує одночасну комунікацію з глобальною або локальною інфраструктурою через роутер та пряме сполучення з периферійними бездротовими вузлами.

На верхньому (прикладному) рівні за допомогою бібліотеки `socket` мовою `MicroPython` створюється TCP/IP сокет, який слухає стандартний 80-й порт. При отриманні вхідного запиту типу GET /A або GET /E сервер обробляє рядкові змінні, змінює стан відповідного GPIO-піна, а у відповідь надсилає сформований пакет даних, який містить заголовок HTTP/1.1 200 OK та тіло веб-сторінки, написаної на базі HTML/CSS. Браузер користувача графічно інтерпретує цей код, відображаючи інтерактивні кнопки та поточні показники датчиків.

Висновки до розділу

У першому розділі проведено теоретико-аналітичний огляд еволюційного поступу, сучасного стану та концептуальних засад побудови систем автоматизації житлового простору Smart Home, а також сформовано технічні вимоги та концептуальну архітектуру для проектування навчально-демонстраційного лабораторного стенду Internet of Things (IoT).

На основі виконаних досліджень та системного аналізу отримано такі результати:

1. Досліджено історичні етапи та еволюційні парадигми розвитку технологій домашньої автоматизації від первісної ери аналогової механізації до сучасної епохи Інтернету речей (IoT) та кіберфізичних систем (AIoT). Визначено, що інтеграція високопродуктивних однокристальних систем (SoC) та бездротових інфокомунікацій докорінно змінила архітектурні підходи, дозволивши перейти від ізольованих релейних контурів до проактивного інтелектуального керування об'єктами нерухомості.

2. Проаналізовано принципи об'єднання гетерогенних інженерних підсистем у єдиний комплекс на основі класичної трирівневої ієрархічної моделі (рівень сприйняття → комунікаційно-мережевий рівень → прикладний обчислювальний рівень). Обґрунтовано, що перехід від автономних контурів автоматики до наскрізних крос-системних сценаріїв дозволяє оптимізувати

процеси збору телеметрії та забезпечити гнучке централізоване керування через єдиний координаційний центр.

3. Класифіковано та деталізовано базові функціональні домени розумного житла, а саме: мікроклімат (HVAC), раціональне енергозбереження та освітлення, охоронна та інженерно-технологічна безпека (контури захисту від затоплень, витоків газу та задимлень). Визначено вагомі переваги автоматизації, серед яких зниження комунальних витрат до 30% та миттєва локалізація аварійних ситуацій без антропогенного втручання.

4. Проведено системний аналіз інженерних викликів та порівняльну характеристику комерційних лідерів ринку (Ajax, BroadLink, Fibaro) і базових мережевих протоколів (дротових PLC X10, промислових шин KNX та бездротових чарункових мереж Z-Wave). Виявлено їхні критичні експлуатаційні недоліки: фрагментацію ринку через закриту пропрієтарну архітектуру (фірмове кодування), високу капіталомісткість обладнання та жорстку залежність від сторонніх хмарних провайдерів. Це безпосередньо довело високу науково-методичну та інженерну доцільність розробки відкритої, модульної та автономної системи на базі доступного енергоефективного мікроконтролера ESP32 та мови MicroPython.

5. Сформовано перелік функціональних та системно-технічних вимог до розроблюваного лабораторного стенду, що охоплюють мінімізацію часу відгуку (Latency), електричне узгодження логічних рівнів периферії (3.3V TTL), модульність та високу дидактичну цінність комплексу.

6. Розроблено концептуальну трирівневу архітектуру IoT-системи, яка базується на обчислювальному ядрі KEYESTUDIO ESP32 PLUS. Визначено оптимальний склад сенсорного контуру (датчики DHT11, MQ-2, PIR, Steam Sensor, RFID RC522) та блоку виконавчих механізмів (сервопривід SG90, вентилятор з драйвером HR1124S, LCD 1602 I2C, адресна матриця SK6812 та пасивний зумер). Обґрунтовано вибір мережевої взаємодії через TCP/IP сокети та розгортання автономного локального веб-сервера (порт 80) у гібридних

бездротових режимах (STA+AP) для забезпечення гнучкого кросплатформного дистанційного моніторингу та керування.

РОЗДІЛ 2. ОБГРУНТУВАННЯ ТА ВИБІР АПАРАТНО-ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ЛАБОРАТОРНОГО СТЕНДУ ІоТ «РОЗУМНИЙ БУДИНОК» НА БАЗІ МІКРОКОНТРОЛЕРА ESP32

2.1. Огляд та аналіз компонентної бази апаратної складової системи

2.1.1. Центральний обчислювальний модуль системи

При проєктуванні автоматизованої системи "Розумний будинок" обґрунтування вибору апаратного забезпечення є критично важливим етапом, що визначає її функціональність, відмовостійкість та загальну ефективність. Для забезпечення комплексного моніторингу параметрів середовища та біометричних показників користувача необхідна інтеграція платформи, здатної до стабільної архівації, обробки та трансляції даних у режимі реального часу.

Апаратна архітектура розроблюваної системи базується на комплексі взаємопов'язаних модулів, що гарантують точність вимірювань, надійність інфокомунікацій та високий рівень автономності. Основними критеріями вибору елементної бази визначено енергоефективність, завадостійкість, експлуатаційну стабільність та масштабованість. Структура апаратного комплексу включає:

- обчислювальне ядро (мікроконтролер або одноплатний комп'ютер) для первинної обробки даних;
- сенсорну підсистему для фіксації соматичних параметрів стану людини та показників мікроклімату;
- інтерфейси бездротового зв'язку для периферійного обміну даними;
- модулі живлення, оптимізовані для тривалого автономного функціонування.

Кожен компонент інтегрується з урахуванням мінімізації антропогенного втручання, забезпечення безперебійної роботи системи та максимізації інформативності даних для оперативного прийняття рішень.

Плата KEYESTUDIO ESP32 PLUS — універсальна розробка WIFI та Bluetooth плата на основі ESP32, інтегрована з модулем ESP32-WOROOM-32 та сумісний з Arduino.

Він оснащений датчиком Холла, високошвидкісним SDIO/SPI, UART, I2S та I2C. Крім того, оснащена операційною системою freeRTOS, яка досить підходить для Інтернету речей і розумного дому.

Завдяки поєднанню потужності, бездротових інтерфейсів та енергоефективності, вона закриває 95% потреб такого проєкту.

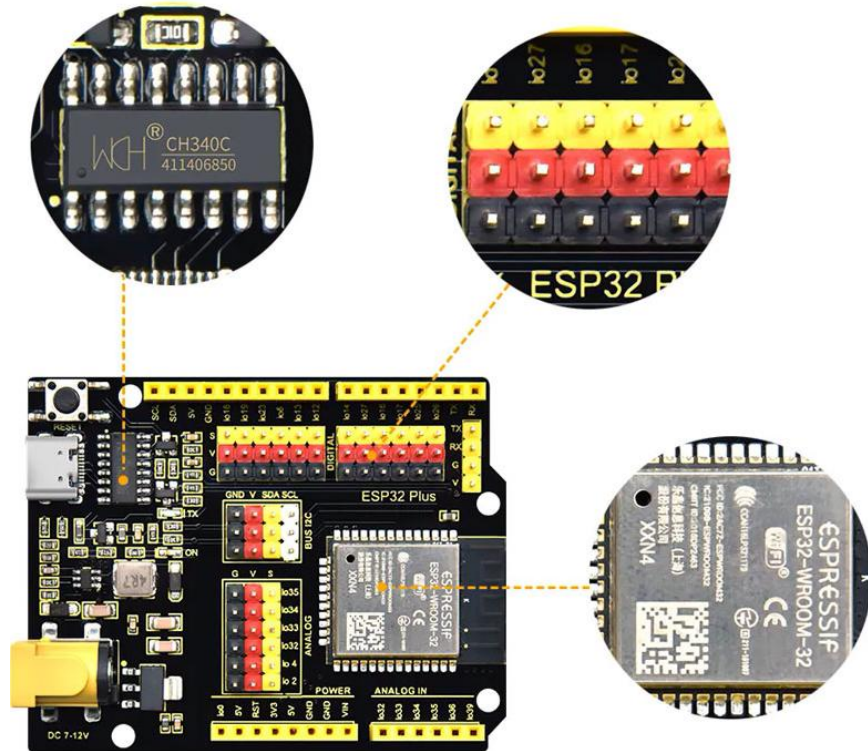


Рис. 2.1. Загальний вигляд плати KEYESTUDIO ESP32 PLUS

Наведемо опис технічних характеристик плати ESP32 PLUS на WROOM-32 на її практичних перевагах для розробки та функціонування системи «Розумний будинок»:

1. Обчислювальна потужність та архітектура

- **Процесор та мікроконтролер.** Використання двоядерного процесора Xtensa LX6 (на базі ESP32-WROOM-32) забезпечує повноцінну багатозадачність. Це дозволяє розумному будинку паралельно опитувати датчики, керувати реле та підтримувати мережеве з'єднання без затримок.
- **Тактова частота (до 240 МГц).** Гарантує миттєву обробку даних та швидку реакцію автоматики на тригери (наприклад, увімкнення світла за датчиком руху).

- **Пам'ять.** 520 КБ SRAM підтримує роботу складних локальних сценаріїв автоматизації, а 4 МБ Flash-пам'яті дозволяють зберігати прошивку разом із розгорнутим на ній автономним веб-інтерфейсом для керування будинком через браузер.
- **Стабільність.** Вбудовані генератори на 26 МГц та 32 кГц забезпечують точність таймерів та стабільну цілодобову роботу системи.

2. Мережеві можливості та бездротова екосистема

- **Wi-Fi зв'язок.** Працює у стандартному діапазоні 2.4 ~ 2.5 ГГц за протоколами 802.11 b/g/n/d/e/i/k/r, що гарантує сумісність із будь-яким домашнім роутером та надійне бездротове покриття. Швидкість до 150 Мбіт/с та оптимізація A-MPDU/A-MSDU забезпечують миттєву передачу команд.
- **Режими Wi-Fi.** Підтримка режимів *Station / softAP / AP+Station / P2P* дозволяє платі як підключатися до домашньої мережі, так і створювати власну точку доступу для прямого керування розумним будинком зі смартфона (навіть якщо інтернет тимчасово зник).
- **Bluetooth v4.2 + BLE.** Наявність Bluetooth Low Energy (BLE) є критично важливою для інтеграції в екосистему будинку бездротових безпаячних BLE-датчиків (температури, відкриття дверей, розумних ваг), які працюють на одній батарейці роками.
- **Радіотракт.** Приймач із високою чутливістю (−98 dBm) та підсилювач LNA гарантують стабільний зв'язок із віддаленими модулями через стіни та перекриття.

3. Підключення периферії та сенсорів домашнього моніторингу

- **Масштабованість (до 32 GPIO).** Дозволяє фізично підключити велику кількість дротових модулів — від фізичних вимикачів до виконавчих реле.

- **Сучасні інтерфейси.** Наявність SD, UART, SPI, I²C, I²S, IR спрощує інтеграцію кліматичних сенсорів, OLED-дисплеїв, зчитувачів карт доступу (RFID) та ІЧ-передавачів для керування старими кондиціонерами чи телевізорами.
- **Аналогові та силові модулі.** 12-бітний ADC слугує для точного зчитування аналогових датчиків (наприклад, витoku газу чи вологості ґрунту вазонів). Апаратний ШІМ (LED PWM, Motor PWM) дозволяє плавно регулювати яскравість світлодіодних стрічок або керувати обертами вентиляторів вентиляції.
- **Сенсорні входи та аудіо.** Підтримка Touch Sensor дозволяє створювати стильні сенсорні вимикачі на стіну. Аудіокодеки CVSD, SBC дають змогу реалізувати роботу з аудіомодулями (наприклад, для голосових сповіщень або функцій розумного дверного дзвінка).
- **Вбудовані датчики.** Вбудований датчик Холла можна використати для фіксації відкриття корпусу пристрою, а температурний датчик — для моніторингу перегріву самої плати в щитку.

4. Живлення та енергоефективність

- **Енергоспоживання.** При середньому струмі ~80 мА система споживає мінімум електрики. Піковий струм до 500 мА активується лише під час передачі даних по Wi-Fi, що запобігає збоям у мережі.
- **Зручність підключення.** Основне живлення 5В дозволяє жити платі від стандартних блоків живлення або телефонних зарядок. Вбудований стабілізатор струму на 800 мА дозволяє жити суміжні датчики безпосередньо від плати ESP32.
- **Автономність.** Підтримка режиму Deep Sleep (глибокого сну) є незамінною для бездротових автономних вузлів розумного будинку, які живляться від акумуляторів і мають прокидатися лише раз на кілька хвилин для відправки метрик.

5. Інтеграція, безпека та програмне забезпечення

- **Кібербезпека IoT-даних.** Захист Wi-Fi мережі (WPA/WPA2/WPA2-Enterprise) та апаратне шифрування за алгоритмами AES, RSA, ECC, SHA надійно захищають розумний будинок від зовнішнього зламу чи перехоплення контролю над замками та освітленням.
- **Розумні протоколи.** Підтримка MQTT, HTTP, SSL та стеків IPv4/IPv6 забезпечує швидку та нативну інтеграцію плати з локальними серверами розумного будинку (на кшталт Home Assistant, OpenHAB, Node-RED) або віддаленими хмарними сервісами для керування через інтернет.
- **Керування та розробка.** Програмування через Arduino IDE чи Thonny Python IDE значно спрощує написання та модифікацію логіки автоматизації, а конвертер CH340 дозволяє легко оновлювати прошивку через звичайний USB-кабель. Також підтримується налаштування через мобільні застосунки (Android/iOS) та AT-команди.
- **Віддалене оновлення (OTA).** Підтримка бездротового оновлення прошивки «по повітрю» є критично важливою — вам не доведеться демонтувати плату зі стіни чи підрозетника, щоб завантажити нову версію програми.

6. Конструктивні особливості

- **Компактність.** Габарити плати всього 27×55 мм дозволяють легко сховати її всередину настінного вимикача, у підрозетник або невеликий розподільчий щиток.
- **Експлуатаційна стійкість.** Робочий діапазон температур від -40°C до $+85^{\circ}\text{C}$ гарантує, що автоматика безперебійно працюватиме як у спекотному горіщному приміщенні, так і в зовнішніх блоках систем (наприклад, у модулі керування вуличними воротами чи системою поливу взимку).

- **Прототипування.** Відстань між пінами у 25.5 мм робить плату зручною для швидкого тестування схем автоматизації на макетних платах без паяння.

Принципова схема плати показана на рисунку 2.3.

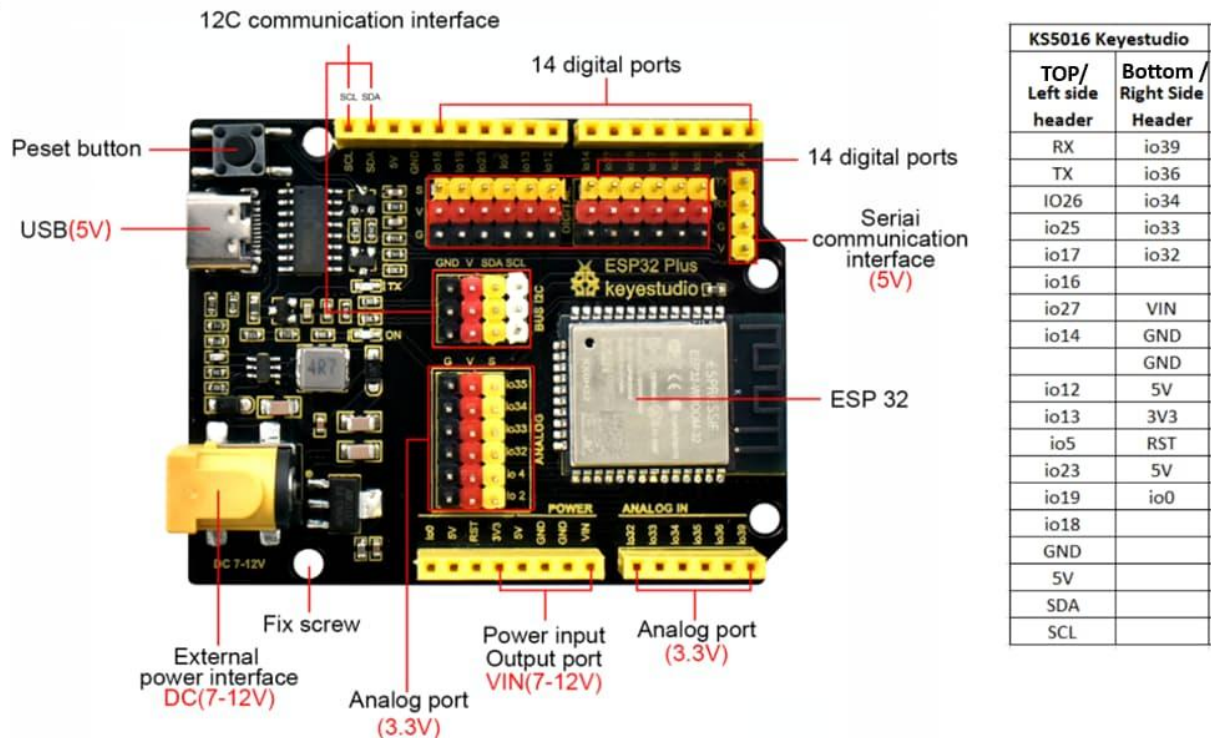
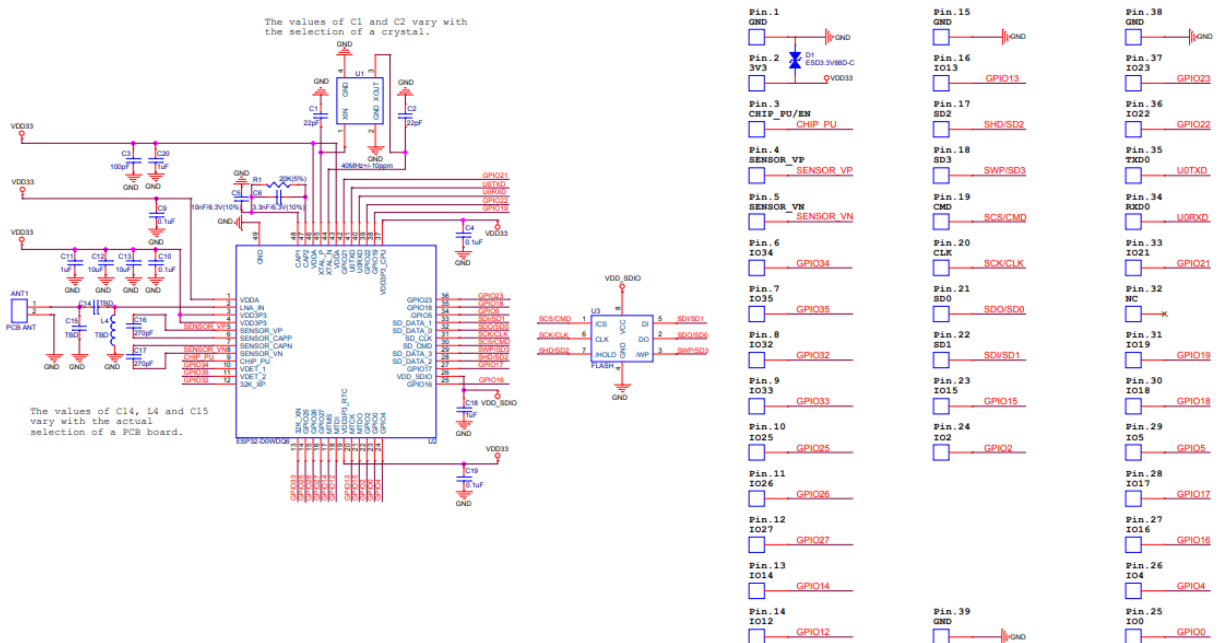
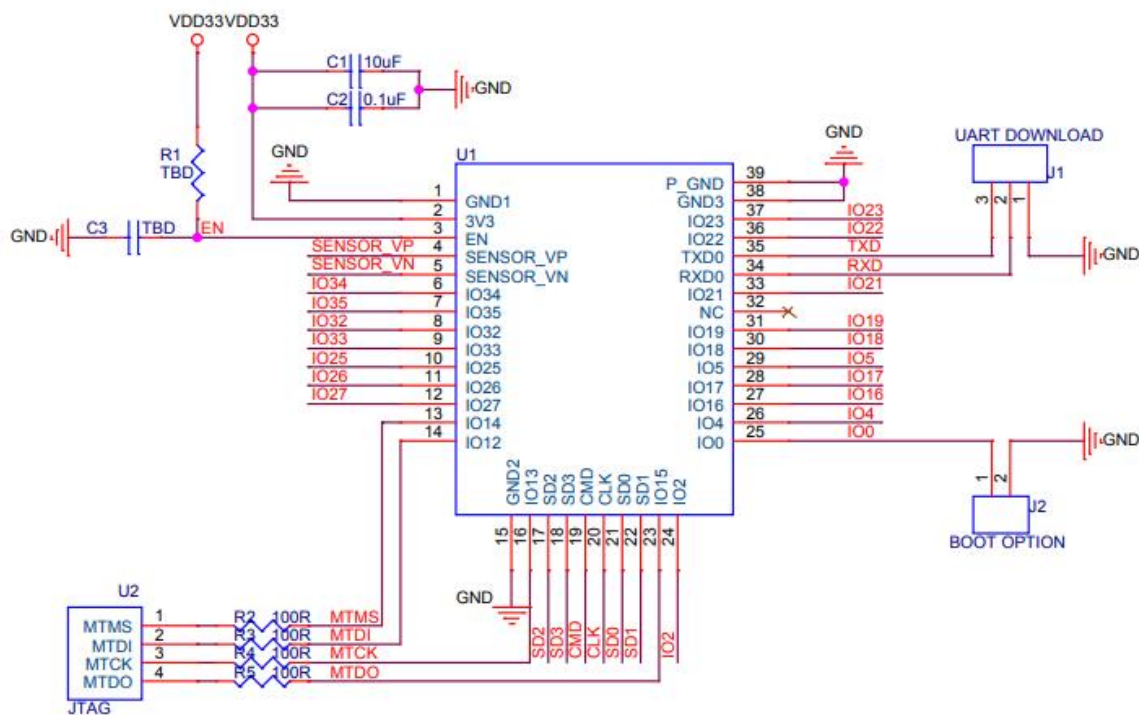


Рис. 2.2. Структура виводів плати KEYESTUDIO ESP32 PLUS





MTDI should be kept at a low electric level when powering up the module.

Рис. 2.3. Принципова схема плати KEYESTUDIO ESP32 PLUS

2.1.2. Підсистема сенсорів та збору даних

Цей блок відповідає за моніторинг внутрішнього середовища «розумного будинку». Для стенду відібрано датчики, що працюють за різними інтерфейсами передачі даних (цифрові One-Wire/I2C та аналогові), що має важливе дидактичне значення для лабораторної практики.

2.1.2.1. Датчик руху PIR Motion Sensor Module

Датчик руху PIR (Passive Infrared Sensor) є пасивним інфрачервоним сенсором, який реагує на зміну теплового випромінювання об'єктів. Коли людина потрапляє в зону дії датчика, він формує цифровий сигнал, який надходить на мікроконтролер ESP32.

Робота сенсора базується на фіксації змін рівня інфрачервоного (теплового) випромінювання, що надходить від навколишніх об'єктів. Основу модуля становить піроелектричний чутливий елемент, закритий лінзою Френеля.

Лінза Френеля розбиває зону огляду на сегменти. Коли людина перетинає ці сегменти, фокусоване теплове випромінювання по черзі потрапляє на два вікна піроелемента, викликаючи різницю потенціалів. Вбудована мікросхема-драйвер

(BISS0001) обробляє цей сигнал і формує на виході модуля дискретний сигнал високого логічного рівня (3.3 V).

У системі «Розумний будинок» датчик використовується для автоматичного виявлення присутності людей або сторонніх об'єктів на контрольованій території.

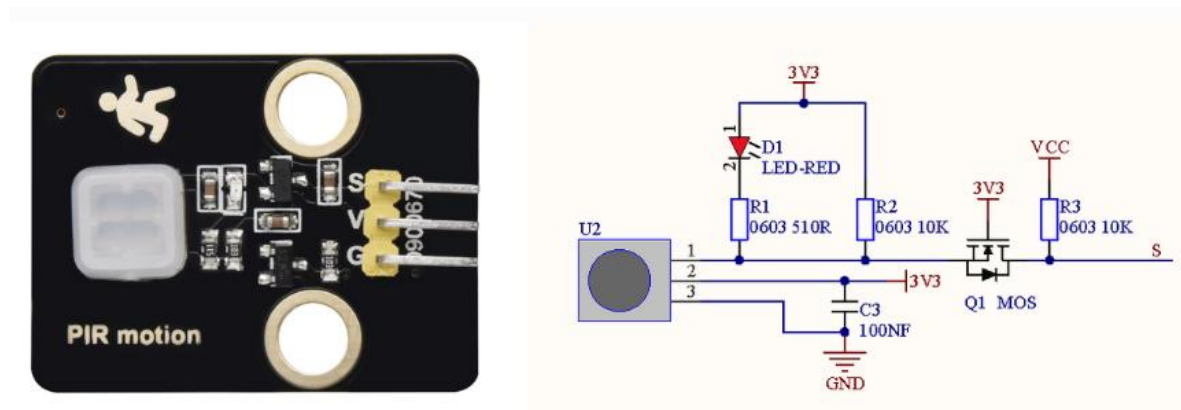


Рис. 2.4. Датчик руху PIR Motion Sensor Module

Технічні характеристики для проектної документації

Ці параметри демонструють інженерне обґрунтування вибору компонента для кімнатних умов:

- **Робоча напруга модуля:** +5V – +20V (ефективно живиться від шини 5V стенду).
- **Вихідний сигнал (логічний рівень):** TTL-сумісний (3.3V – високий рівень 0V – низький рівень). Це забезпечує **пряму сумісність із GPIO-пінами ESP32** без використання дільників напруги.
- **Дальність виявлення об'єктів:** до 7м (регулюється вбудованим потенціометром).
- **Кут огляду конуса:** $< 120^\circ$.
- **Час затримки вихідного сигналу:** від 0.3с до 18⁰ (регулюється другим потенціометром).

Сценарії використання в системі «Розумний будинок»

На лабораторному стенді за допомогою цього датчика відпрацьовуються два фундаментальні алгоритми автоматизації:

- 1) **Енергозбереження (Smart Lighting):** При фіксації руху ESP32 надсилає команду на увімкнення реле освітлення або активацію адресної стрічки

SK6812. Якщо рух відсутній протягом заданого часу (таймауту), світло автоматично вимикається.

2) **Охоронна сигналізація (Security System):** Якщо система переведена в режим «Охорона» (через хмару Blynk чи локальну кнопку), то при спрацьовуванні датчика руху ESP32 активує тривожний сценарій: вмикає червону індикацію на RGB-модулі, активує звуковий зумер та надсилає push-сповіщення (або Telegram/MQTT-повідомлення) користувачеві.

Особливості інтеграції з ESP32

Оскільки датчик HC-SR501 може працювати у двох режимах повторного тригерування (регулюється перемичкою/джампером на платі), для дипломної роботи важливо вказати обраний режим:

- **Режим "L" (поодинокий тригер):** після виявлення руху вихід переходить у високий рівень на час T, після чого падає в 0, навіть якщо рух продовжується.

- **Режим "H" (динамічний тригер - Рекомендовано):** вихід залишається високим увесь час, поки об'єкт рухається в зоні видимості датчика.

У програмному забезпеченні мікроконтролера опитування датчика руху доцільно реалізувати не через постійне сканування порту в циклі loop(), а за допомогою **зовнішніх переривань (Hardware Interrupts)** через функцію attachInterrupt(). Це дозволяє ESP32 миттєво реагувати на рух і оптимізує використання процесорного часу.

Таблиця 2.1

Підключення PIR Motion Sensor Module до KEYESTUDIO ESP32 PLUS

Модульний Pin	Колір дроту	ESP32 Pin плати
V	ЧЕРВОНИЙ	V
G	ЧОРНИЙ	G
S	ЖОВТИЙ	io5

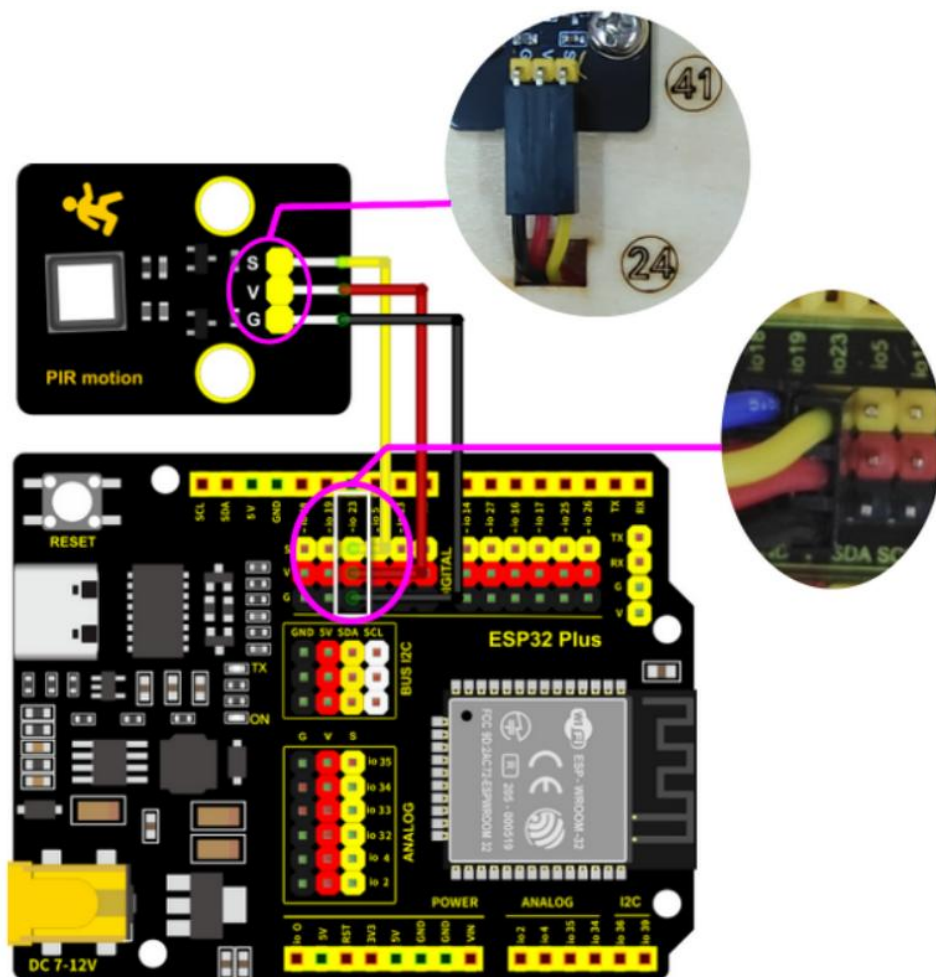


Рис. 2.4. Підключення PIR Motion Sensor Module до KEYESTUDIO ESP32 PLUS

2.1.2.2. Датчик температури та вологості DHT11

Датчик температури та вологості DHT11 видає цифрові сигнали. Він застосовує принципи аналогового прийому та перетворення сигналу, а також технологію вимірювання температури та вологості, що забезпечує довготривалу стабільність і високу надійність. Крім того, сенсор інтегрує високоточний резистивний датчик вологості та резистивний термочутливий датчик температури, а також підключений до 8-бітного високопродуктивного мікромікроконтролю.

Кожен датчик DHT11 є відкалібрований у лабораторії і є надзвичайно точним у калібруванні вологості. Коефіцієнти калібрування зберігаються як програми в пам'яті ОЗП. Однопровідний послідовний інтерфейс робить системну інтеграцію швидкою та легкою.

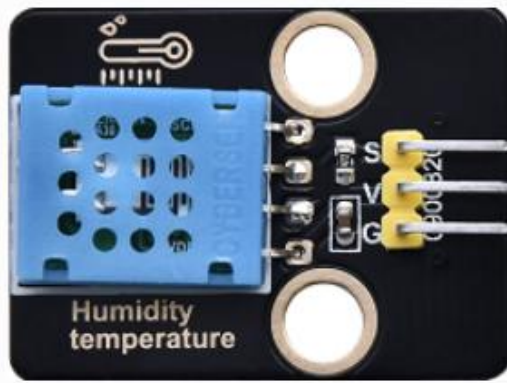


Рис. 2.5. Датчик температури та вологості DHT11

Принцип функціонування пристрою

Сенсор DHT11 є комбінованим приладом, що складається з двох основних частин:

1. **Резистивний датчик вологості (гігрометр):** вимірює електричний опір матеріалу (зазвичай вологочутливого полімеру), який змінюється залежно від концентрації водяної пари в повітрі.
2. **Термістор (NTC-датчик):** вимірює температуру шляхом фіксації змін власного електричного опору при нагріванні чи охолодженні.

Головною перевагою модуля є наявність вбудованого 8-бітного мікроконтролера, який зчитує аналогові значення з обох сенсорів, виконує їх калібрування та перетворює у цифровий сигнал. Дані передаються за власним часозалежним однопровідним протоколом (Single-Bus/1-Wire).

Дані температури та вологості передають по одному провіднику та мають послідовний інтерфейс, де одиниця та нуль кодуються різними тривалостями імпульсу. Один процес передачі даних триває близько 4 мс.

Дані складаються з десяткової та цілої частин. Повна передача даних становить 40 біт, і датчик першим надсилає вищий біт даних. Формат даних датчика такий:

Вологість: 8-біт ціла частина	Вологість: 8-біт десяткова частина	Температура: 8-біт ціла частина	Температура: 8-біт десяткова частина	Чек сума: 8-біт
----------------------------------	---------------------------------------	------------------------------------	---	--------------------

Рис. 2.6. Схема передачі даних

Часова діаграма передачі даних DHT11.

Коли контролер надсилає сигнал старту, датчик DHT11 переходить із режиму низького енергоспоживання на режим роботи, очікуючи, поки контролер завершить сигнал старту. Далі DHT11 надсилає сигнал відповіді з 40-бітними даними, які включають інформацію про відносну вологість і температуру. Після збору даних DHT11 перейде в режим низького енергоспоживання, доки знову не отримає сигнал старту від контролера.

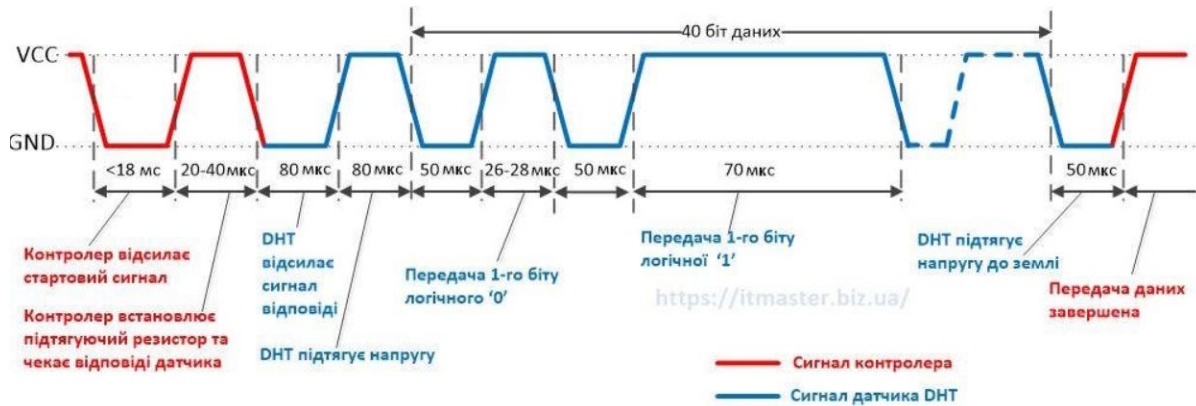


Рис. 2.7. Передача сигналу в датчику DHT11

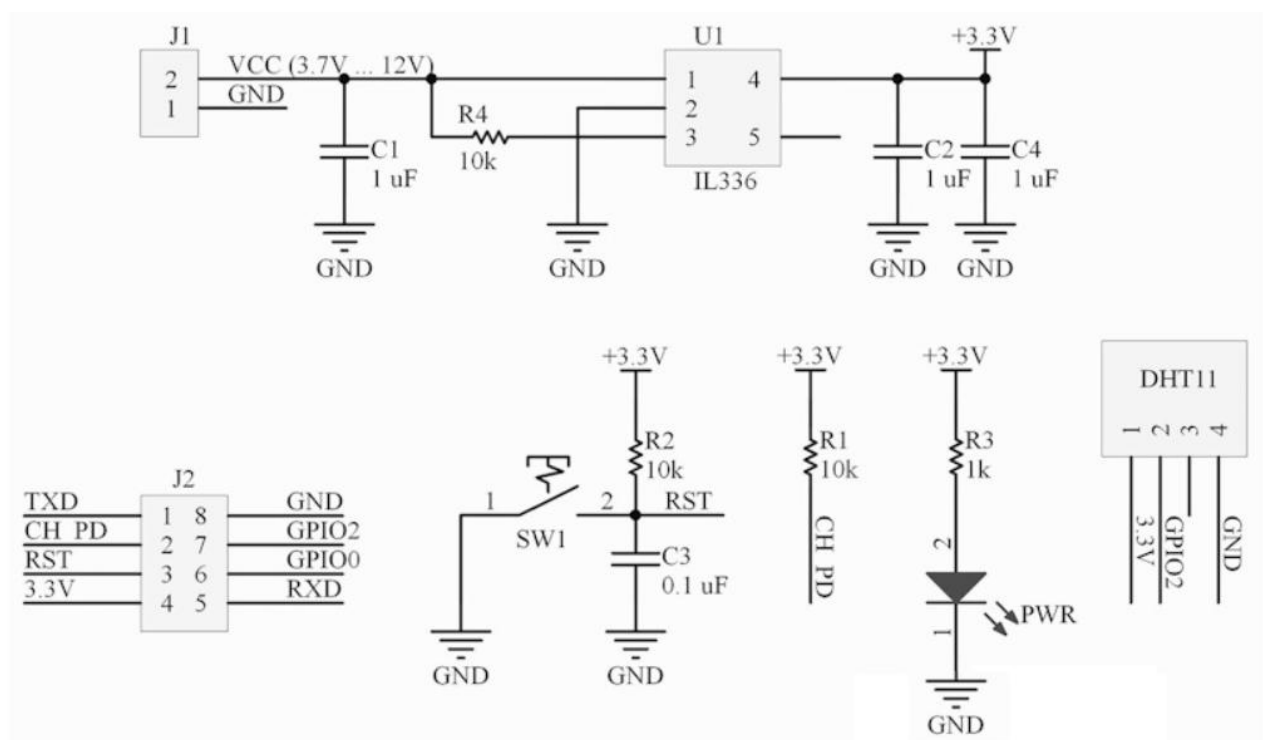


Рис. 2.8. Електрична схема модуля DHT11

Параметри датчика DHT11 обґрунтовують його застосування в навчальних цілях та для закритих приміщень:

- **Напруга живлення:** +3.3V – +5.5V (може житися як від лінії 3.3V, так і від 5V плати ESP32 Plus).

- **Діапазон вимірювання температури:** від 0° до $+50^{\circ}$ (похибка $\pm 2^{\circ}\text{C}$).
- **Діапазон вимірювання відносної вологості:** від 20% до 90% (похибка $\pm 5\%$).
- **Частота опитування (дискретизація):** не частіше ніж 1 раз на 1–2 секунди (0.5 – 1 Гц).

Сценарії використання в системі «Розумний будинок».

На базі отриманих від DHT11 даних мікроконтролер ESP32 реалізує такі алгоритми керування кліматом (імітація клімат-контролю):

1. **Автоматизація вентиляції (Керування вологістю).** Якщо показник вологості перевищує заданий поріг (наприклад, 65%), ESP32 замикає реле, що запускає витяжний вентилятор (або сервопривід відчиняє квартиру). При нормалізації показників вентиляція вимикається.
2. **Автоматизація опалення/охолодження.** При зниженні температури нижче комфортної (18°C) система подає сигнал на реле для увімкнення нагрівального елемента. При перевищенні ліміту (25°C) — активується імітація кондиціонера або надсилається попередження.
3. **Хмарний моніторинг:** Поточні значення $T^{\circ}\text{C}$ та $H\%$ циклічно відправляються через Wi-Fi за протоколом MQTT або HTTP на хмарну платформу (наприклад, Blynk або локальний сервер) для побудови графіків та віддаленого контролю користувачем.

Таблиця 2.2

Підключення датчика температури та вологості DHT11 до KEYESTUDIO ESP32 PLUS

Модульний Pin	Колір дроту	ESP32 Pin плати
V	ЧЕРВОНИЙ	V
G	ЧОРНИЙ	G
S	ЖОВТИЙ	Io17

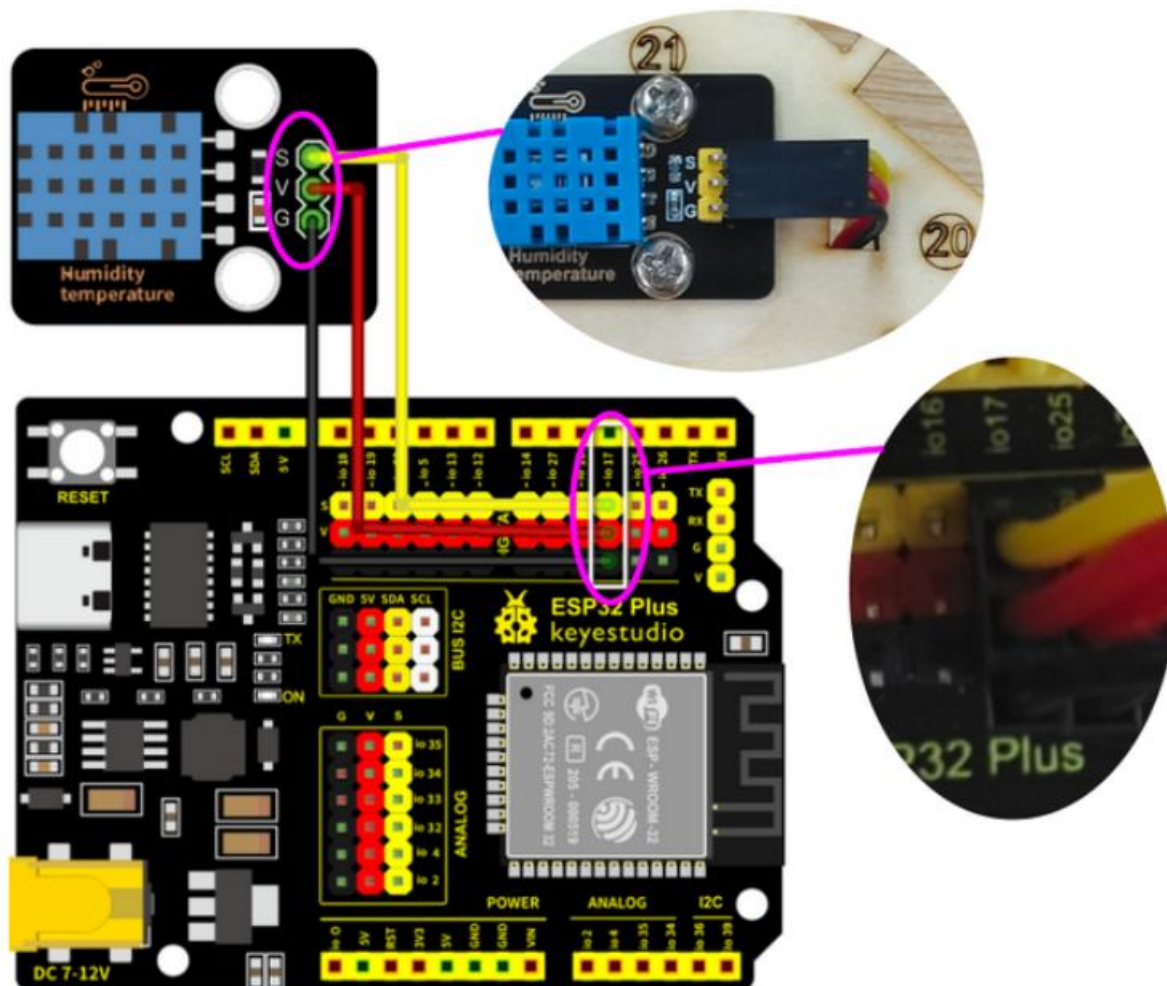


Рис. 2.9. Підключення датчика DHT11 Temperature and Humidity Sensor до KEYESTUDIO ESP32 PLUS

2.1.2.3. Модуль цифрової кнопки EASY Plug RJ11

Модуль тактильної кнопки EASY Plug Push-button являє собою компактне апаратне рішення, призначене для комутації сигналів постійного струму в системах автоматизації. Цей базовий елемент інтерфейсу користувача сумісний із популярними мікроконтролерними платформами, такими як Arduino та Raspberry Pi. Функціонування модуля базується на зміні логічних рівнів: при замиканні контактів (натисканні) на цифровому виході формується сигнал високого рівня (HIGH), а при розмиканні (відпусканні) — низького рівня (LOW). Завдяки простоті підключення до ліній введення-виведення (GPIO), модуль є оптимальним для первинного тестування портів мікроконтролера та освоєння базових алгоритмів обробки зовнішніх переривань.



Рис. 2.10. Загальний вигляд модуля цифрової кнопки EASY Plug RJ11

Обраний кнопковий модуль має такі технічні та конструктивні параметри. Пристрій належить до класу цифрових датчиків та формує на виході дискретний сигнал. Діапазон напруги живлення становить від 3.3 до 5 В, що забезпечує повну сумісність як з 5-вольтними логічними рівнями плат Arduino, так і з 3.3-вольтними рівнями мікроконтролера ESP32. Для швидкого та надійного підключення до макету в модулі передбачено інтерфейсний роз'єм RJ11. Конструктивно елемент оснащений ергономічною тактильною кнопкою збільшеної площі із захисним ковпачком підвищеної зносостійкості.

Для роботи треба з'єднати червоний з V, чорний з G, жовтий з S.

Головною інженерною особливістю компонентів серії EASY Plug є використання стандартизованих чотириконтактних роз'ємів **RJ11 (6P4C)** замість класичних штирових з'єднувачів (Dupont-кабелів). Це проєктне рішення має кілька суттєвих переваг для побудови лабораторного стенду:

1. **Захист від помилок комутації (Плутанини пінів).** Коннектор RJ11 має фіксатор-засувку, що унеможливорює неправильне підключення модуля (переполюсовку живлення VCC та GND або замикання сигнального виходу).
2. **Надійність контакту.** На відміну від макетних плат (Breadboard), де контакти з часом послаблюються, інтерфейс RJ11 гарантує стабільне з'єднання, що критично для демонстраційних стендів.
3. **Вбудована обв'язка.** Модуль кнопки вже містить на своїй друкованій платі необхідний підтягуючий резистор (Pull-up) номіналом 10 кОм. Це спрощує проєктування, оскільки розробнику не потрібно реалізовувати зовнішню резистивну обв'язку для запобігання "плаваючого" потенціалу на вході мікроконтролера.

При підключенні до базової плати розширення (наприклад, Keyestudio ESP32 Plus) модуль використовує лише один сигнальний провід у кабелі RJ11, який з'єднується з відповідним цифровим GPIO-піном мікроконтролера.

Таблиця 2.3

Підключення кнопкового модуля до KEYESTUDIO ESP32 PLUS

Модульний Pin	Колір дроту	ESP32 Pin плати
V	ЧЕРВОНИЙ	V
G	ЧОРНИЙ	G
S	ЖОВТИЙ	io5

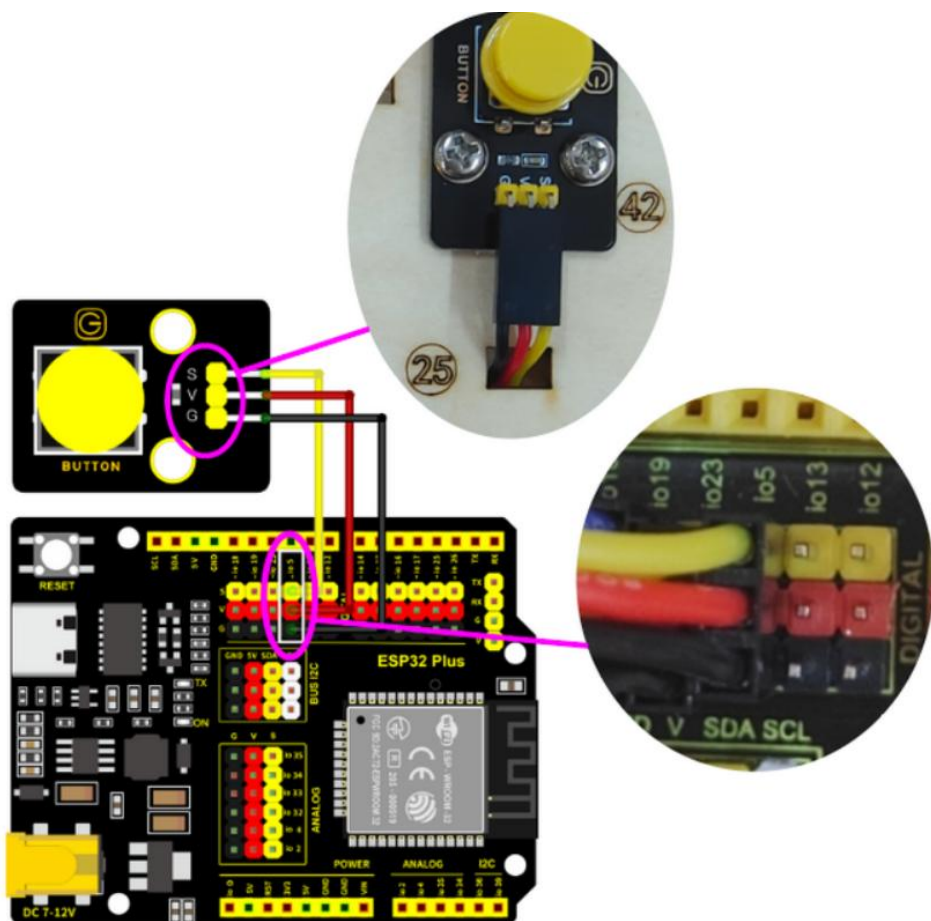


Рис. 2.11. Підключення кнопки EASY Plug RJ11 до плати KEYESTUDIO ESP32 PLUS

2.1.2.4. Датчик водяної пари (Steam Sensor)

Датчик пари виявляє наявність води, тому його зазвичай використовують для виявлення дощу. Якщо дощ потрапляє на провідні панелі датчика, він надішле сигнал на плату Arduino.



Рис. 2.12. Загальний вигляд датчику пари

Датчик пари виявляє наявність води, тому його зазвичай використовують для виявлення дощу. Якщо дощ потрапляє на провідні панелі датчика, він надішле сигнал на плату Arduino.

Робота датчика базується на принципі вимірювання електропровідності (опору) між відкритими друкованими провідниками на його платі, що утворюють характерний геометричний малюнок у вигляді павутини.

У сухому стані опір між доріжками є максимально високим, і струм через них не проходить. Коли на поверхню сенсора потрапляють краплі води, конденсат або водяна пара, вони замикають між собою позолочені лінії. Провідність різко зростає, а загальний опір пару вихідного каскаду падає. Вбудована схемотехніка модуля перетворює цю зміну опору в аналоговий сигнал (напругу), величина якого прямо пропорційна площі покриття плати вологою та її щільності.

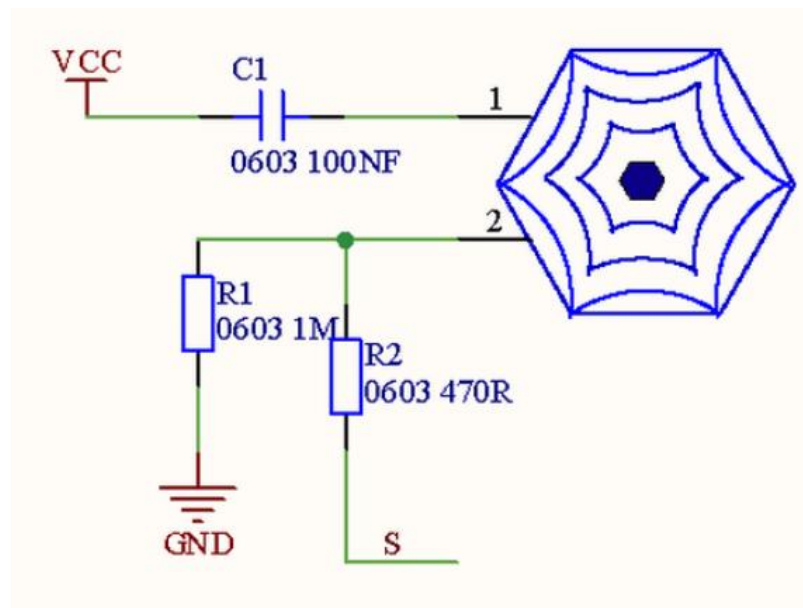


Рис. 2.13. Електрична схема датчику пари

Технічні характеристики

- **Тип сигналу.** Аналоговий (напруга на виході змінюється від 0 В до значення напруги живлення).
- **Робоча напруга.** +3.3V – +5V (рекомендується підключати до лінії +3.3 V для прямого узгодження з АЦП мікроконтролера ESP32 без ризику перевантаження пінів).
- **Споживаний струм:** < 20 мА (енергоефективне рішення).
- **Інтерфейс підключення (виводи на платі):**
 - G (Ground) — заземлення, підключається до загального мінусу системи.
 - V (VCC) — живлення датчика.
 - S (Signal) — аналоговий вихід, що підключається до одного з пінів АЦП (ADC) мікроконтролера ESP32.

Сценарії використання в системі «Розумний будинок».

На базі Steam Sensor реалізуються критично важливі алгоритми автоматизації та інженерного захисту будівлі:

1. **Система захисту від затоплення (Anti-Flood).** Датчик розміщується на підлозі в місцях потенційного витоку води (кухня, ванна кімната, підвал). При фіксації води ESP32 миттєво перекриває вхідні магістралі за допомогою сервоприводу (або соленоїдного клапана), активує пасивний дзвінок (Buzzer) та надсилає аварійне push-сповіщення власнику будинку.
2. **Контроль мікроклімату та вентиляції (Steam/Vapor Control).** Датчик встановлюється у зоні ванної кімнати чи домашньої пральні. При інтенсивному випаровуванні гарячої води (підвищенні концентрації пари) датчик фіксує утворення конденсату, і мікроконтролер автоматично запускає витяжну вентиляцію через блок реле для запобігання появі плісняви.
3. **Модуль метеостанції (Датчик дощу).** Якщо стенд імітує зовнішній моніторинг, сенсор виноситься на вулицю під кутом 45

градусів. При перших краплях дощу система автоматично закриває вікна (імітується за допомогою сервоприводу SG90) або згортає маркізи (сонцезахисні навіси).

Таблиця 2.4

Підключення датчика пари до KEYESTUDIO ESP32 PLUS

Модульний Pin	Колір дроту	ESP32 Pin плати
V	ЧЕРВОНИЙ	V
G	ЧОРНИЙ	G
S	ЖОВТИЙ	Io35

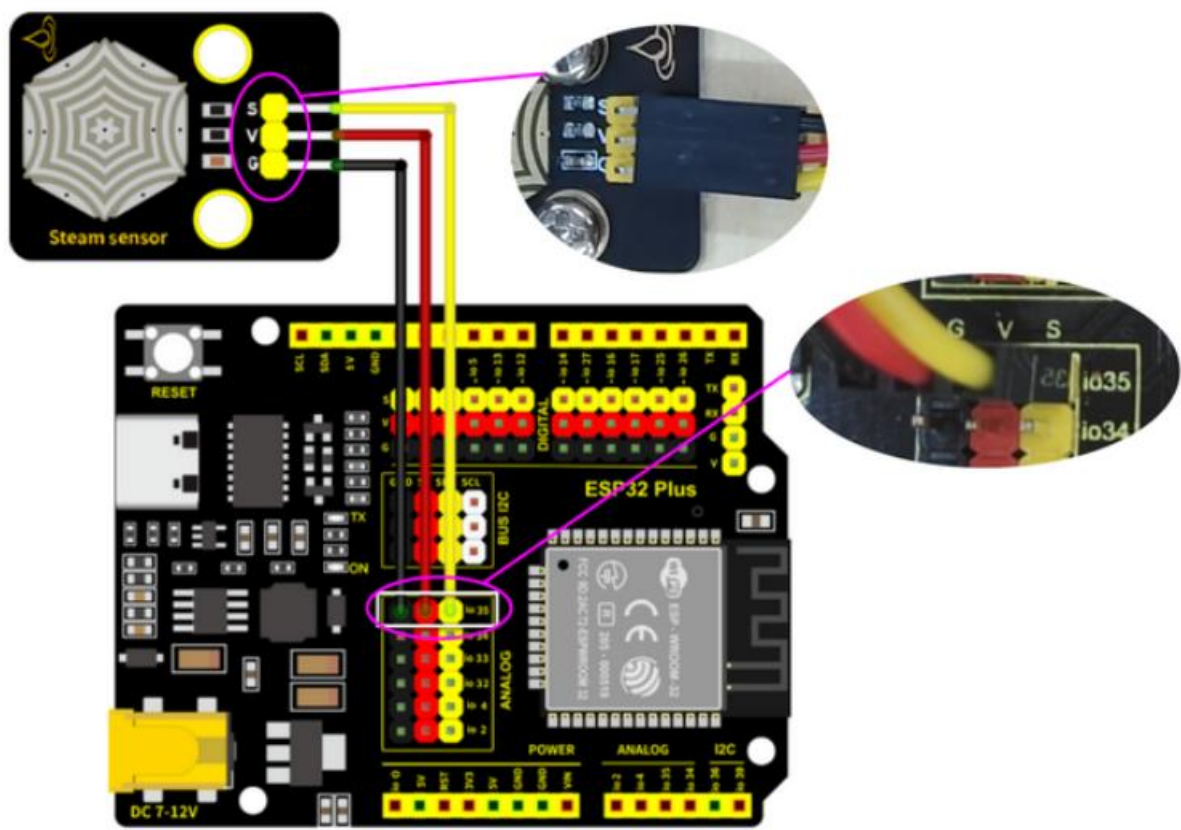


Рис. 2.14. Підключення датчика пари до плати KEYESTUDIO ESP32 PLUS

2.1.2.5. Модуль радіочастотної автентифікації RFID RC522

Для реалізації функцій санкціонованого доступу до об'єкта («Розумного будинку»), обліку користувачів та безпеки, у лабораторному стенді застосовано технологію RFID (Radio Frequency IDentification — радіочастотна ідентифікація).

Принцип побудови та класифікація RFID-систем.

Архітектура будь-якої RFID-системи є двокомпонентною і складається з двох основних пристроїв:

1. **Зчитувач (рідер / інтеррогатор):** апаратний пристрій, призначений для генерації радіочастотного поля, опитування міток, зчитування та запису інформації в їхню пам'ять.
2. **Відповідач (транспондер / мітка):** мініатюрний ідентифікатор (у вигляді карти, брелока чи наліпки), що містить унікальний ідентифікаційний код (UID) та закріплюється за об'єктом або користувачем.

За типом джерела живлення транспондерів RFID-продукти класифікують на три категорії:

- **Активні:** мають інтегровану батарею, транслюють сигнал на великі відстані, працюють у високочастотних діапазонах.
- **Напівактивні (внутрішньоживлені):** батарея живить лише чип мітки, а передача сигналу активується полем зчитувача.
- **Пасивні:** не мають власного джерела живлення, функціонують виключно за рахунок енергії електромагнітного поля рідера.

Пасивні RFID-системи є найбільш технологічно зрілими, економічно вигідними та масовими на ринку побутової електроніки. Вони використовуються повсюдно: від електронних проїзних квитків та обідніх карт до банківських платіжних систем і безконтактних ключів доступу (інтеркомів, готельних замків).

Основними робочими частотами для пасивних систем є 125 кГц (низька частота, LF), 13.56 МГц (висока частота, HF), а також діапазони 860 – 960 МГц (надвисока частота, UHF).

Технічні особливості інтегрованого RFID-модуля

У розроблюваному IoT-стенді використано пасивний RFID-модуль, що функціонує на високочастотній несучій 13.56 МГц (відповідно до стандартів ISO/IEC 14443 на базі мікросхеми RC522).

Апаратна архітектура зчитувача складається з високочастотного генератора, сигнального процесора та друкованої петльової антени. Транспондер (мітка) є повністю безбатарейним сенсорним пристроєм, конструкція якого

обмежена кремнієвим кристалом інтегральної схеми (IC), енергонезалежною пам'яттю (EEPROM) та котушкою індуктивності, яка виконує роль внутрішньої антени.

Фізичний принцип взаємодії (Електромагнітна індукція)

Взаємодія між компонентами системи базується на явищі магнітного спряження (індуктивного зв'язку) і підпорядковується фундаментальним законам електродинаміки:

1. Антена зчитувача безперервно випромінює синусоїдальне магнітне поле на частоті 13.56 МГц.
2. Коли пасивна мітка потрапляє в радіус дії цього поля (зону зчитування), магнітний потік пронизує витки її внутрішньої котушки.
3. Відповідно до закону електромагнітної індукції Фарадея, у котушці мітки індукується електрорушійна сила (ЕРС). Завдяки вбудованому діодному випрямлячу та згладжувальному конденсатору всередині мітки, ця енергія перетворюється на постійний струм, який живить та активує логічний чип транспондера.
4. Активованій чип починає модулювати відбите поле зчитувача (метод амплітудної маніпуляції Backscatter Modulation), передаючи свій унікальний UID-код назад на антену рідера.

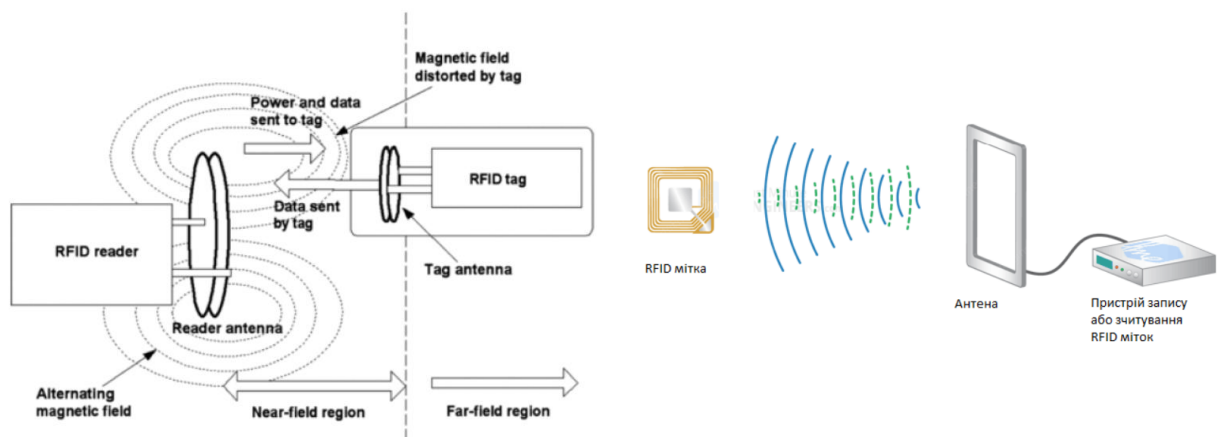


Рис. 2.15. Принцип роботи RFID

Модуль RFID-RC522. MFRC522 — це високоінтегрована мікросхема зчитування/запису для безконтактного зв'язку на частоті 13,56 МГц. Внутрішній передавач MFRC522 здатний керувати антеною зчитування/запису, призначеною для зв'язку з картками та транспондерами ISO/IEC 14443 A/MIFARE без додаткової активної схеми. Модуль приймача забезпечує надійну та ефективну реалізацію для демодуляції та декодування сигналів від карток та транспондерів, сумісних з ISO/IEC 14443 A/MIFARE. Цифровий модуль керує повною функціональністю кадрювання та виявлення помилок (парність та CRC) за стандартом ISO/IEC 14443A.

Цей RFID-модуль використовує MFRC522 як керуючий чіп та використовує інтерфейс I2C (міжінтегральна схема).



Рис. 2.16. Загальний вигляд MFRC522

Таблиця 2.5

Технічні характеристики модуля RFID- MFRC522

Параметр	Значення / Характеристика
Робоча напруга	3.3 – 5 В постійного струму (DC)
Робочий струм	13 – 100 мА (при DC 5 В)
Струм холостого ходу	10 – 13 мА (при DC 5 В)
Струм у режимі сну	<80 мкА
Піковий струм	<100 мА
Робоча частота	13.56 МГц
Максимальна потужність	0.5 Вт

Параметр	Значення / Характеристика
Підтримувані типи карток	MIFARE1 S50, MIFARE1 S70, MIFARE UltraLight, MIFARE Pro, MIFARE Desfire
Швидкість передачі даних	максимум 10 Мбіт/с
Робоча температура навколишнього середовища	від -20°C до $+80^{\circ}\text{C}$
Температура зберігання	від -40°C до $+85^{\circ}\text{C}$
Відносна вологість навколишнього середовища	від 5% до 95%

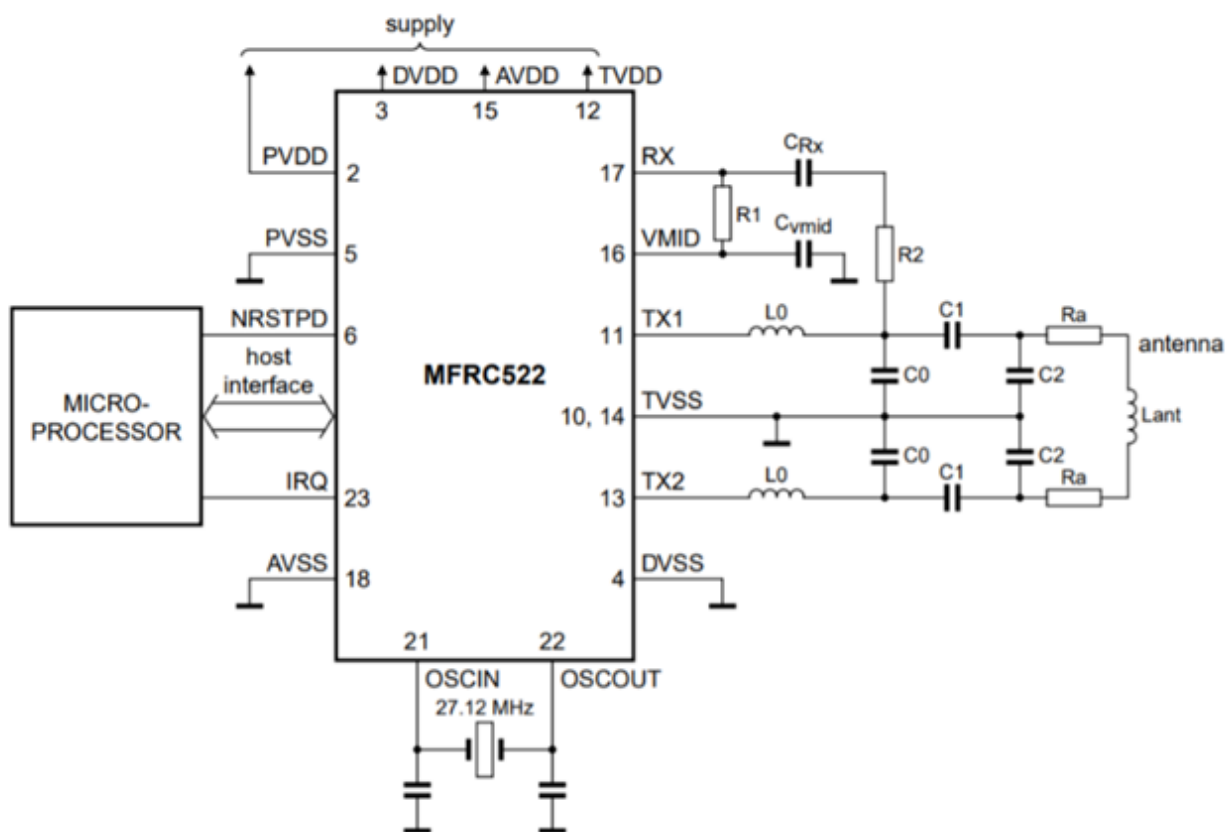


Рис. 2.17. Електрична схема MFRC522

Сценарій використання в системі «Розумний будинок»

На базі цього модуля на лабораторному стенді відпрацьовується сценарій «Розумний замок» (Smart Lock):

- При піднесенні карти до зчитувача, ESP32 отримує UID мітки через інтерфейс SPI.

- Мікроконтролер порівнює код із базою даних (дозволеними ключами).
- Якщо код валідний — сервопривід (актуатор) відкриває замок дверного макету, а зелений світлодіод підтверджує доступ.
- Якщо код невідомий — система блокує доступ, активує червону індикацію RGB-модуля SK6812 та вмикає пасивний дзвінок (Buzzer) у режимі попередження.

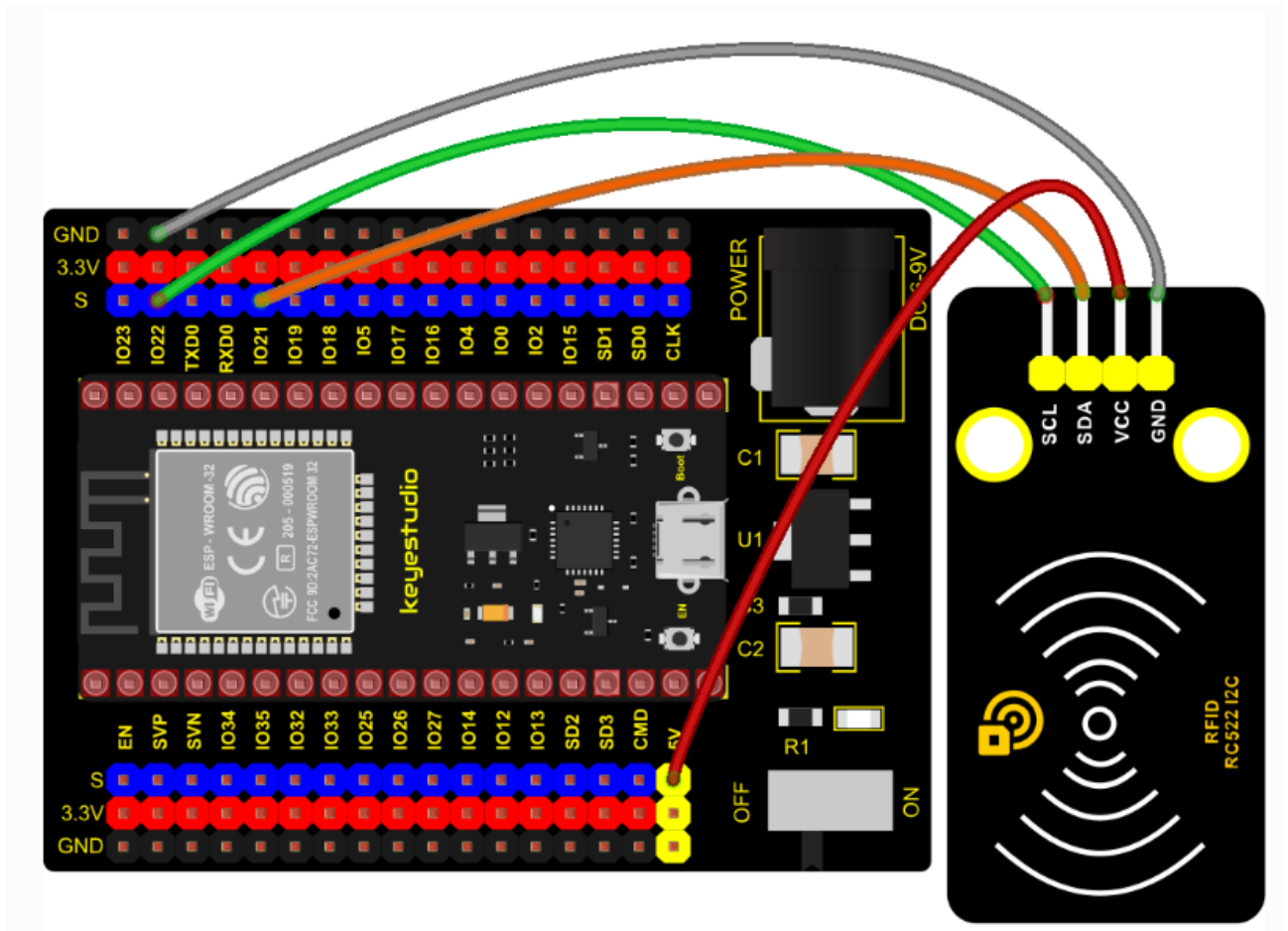


Рис. 2.18. Підключення MFRC522 до плати KEYESTUDIO ESP32 PLUS

2.1.2.6. Напівпровідниковий датчик газу та диму MQ-2

Для реалізації функцій протипожежної безпеки, виявлення витоків горючих газів та задимлення всередині приміщення «Розумного будинку», у лабораторному стенді застосовано датчик серії MQ-2.

Цей датчик підходить для виявлення зрідженого нафтового газу (LPG), йодованого бутану, пропану, метану, спирту, водню та диму. Він має високу чутливість та швидку реакцію.

Крім того, чутливість можна регулювати обертанням потенціометра. В експерименті ми зчитуємо аналогове значення на портах A0 та D0, щоб визначити вміст газу.



Рис. 2.19. Загальний вигляд диму MQ2

Принцип функціонування пристрою.

Сенсор MQ-2 належить до класу напівпровідникових (газорезистивних) пристроїв. Його внутрішня структура містить керамічну трубку з покриттям із діоксиду олова (SnO_2), всередині якої встановлено ніхромовий нагрівальний елемент (підігрівач).

Принцип роботи базується на зміні провідності шару діоксиду олова при нагріванні в присутності різних газів:

1. У чистому повітрі провідність сенсора залишається низькою.
2. При появі в приміщенні молекул горючих газів (пропан, бутан, метан, водень) або часток диму (продуктів горіння), вони вступають у хімічну реакцію з киснем на поверхні напівпровідника.
3. Це призводить до зниження потенційного бар'єру в шарі SnO_2 , внаслідок чого електричний опір датчика різко падає, а провідність зростає.

Модуль MQ-2, що використовується на лабораторних стендах, зазвичай постачається на платі-носії, яка має як аналоговий, так і цифровий виходи:

- **Робоча напруга:** +5В постійного струму (DC).

Нагрівальний елемент датчика потребує стабільної напруги +5В для досягнення робочої температури камери (близько 400°C). Тому живлення модуля здійснюється виключно від 5-вольтової шини плати ESP32 Plus.

- **Споживаний струм:** 150 – 180 мА (через постійну роботу внутрішнього ТЕНа). Цей факт необхідно врахувати при розрахунку загального балансу потужності джерела живлення стенду.
- **Типи інтерфейсів (виходів модуля):**
 - **Аналоговий вихід (A0 / AO):** Видає напругу від 0 В до 5 В, яка пропорційна концентрації газу в повітрі.
 - **Цифровий вихід (D0 / DO):** Видає логічний сигнал (TTL-рівень) за принципом «0/1» (норма/тривога). Пороговий рівень спрацьовування налаштовується вручну за допомогою компаратора (мікросхеми LM393) та вбудованого триммера (потенціометра) на платі модуля.
- **Діапазон виявлення речовин:**
 - Зріджений нафтовий газ (LPG), пропан, бутан: 200 – 5000 ppm} (частин на мільйон).
 - Природний газ (метан): 500 – 5000 ppm.
 - Дим, горючі пари: 300 – 10000 ppm.

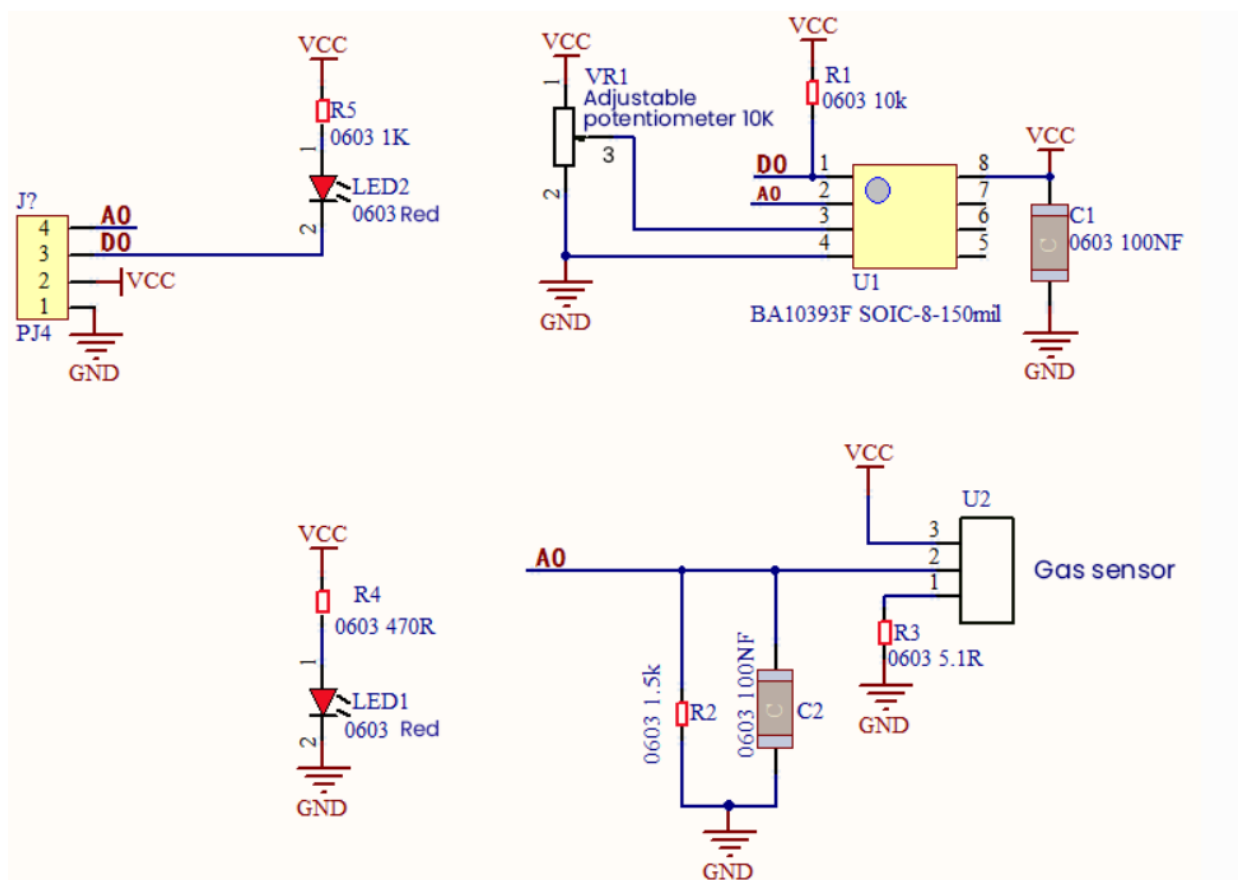


Рис. 2.20. Електрична схема MQ2

Сценарії використання в системі «Розумний будинок»

На базі отриманих від MQ-2 даних мікроконтролер ESP32 реалізує комплексні сценарії захисту життєдіяльності (Smart Security):

1. **Аварійна вентиляція (Локальне керування).** При перевищенні безпечного порогу концентрації газу/диму мікроконтролер замикає реле для активації витяжного вентилятора та відкриває сервоприводом віконні макети для провітрювання приміщення.
2. **Світлозвукова сигналізація.** Одночасно із запуском вентиляції ESP32 переводить адресний RGB-модуль SK6812 в режим пульсуючого червоного світіння та запускає пасивний дзвінок (Buzzer) у режимі переривчастої сирени.
3. **Хмарне сповіщення (IoT).** Оскільки витік газу є критичною загрозою, повідомлення негайно передається через Wi-Fi за протоколом MQTT/HTTP на сервер або мобільний додаток користувача у вигляді push-нотифікації з точним значенням концентрації в ppm.

При підключенні аналогового виходу MQ-2 до АЦП мікроконтролера ESP32 необхідно враховувати два інженерні аспекти:

1. **Узгодження рівнів.** Вихідний сигнал датчика може досягати +5 В, тоді як вхідні піни АЦП ESP32 розраховані на максимальну напругу +3.3 В. Для безпечного зчитування сигнальну лінію підключають через резистивний дільник напруги (наприклад, резистори 10 кОм та 20 кОм).
2. **Час прогріву (Preheat time).** Після увімкнення сенду датчику потрібно від 20 до 60 секунд для прогріву чутливого шару, протягом яких його покази є нестабільними. У коді програми реалізується програмна затримка або ігнорування показів АЦП у цей період.

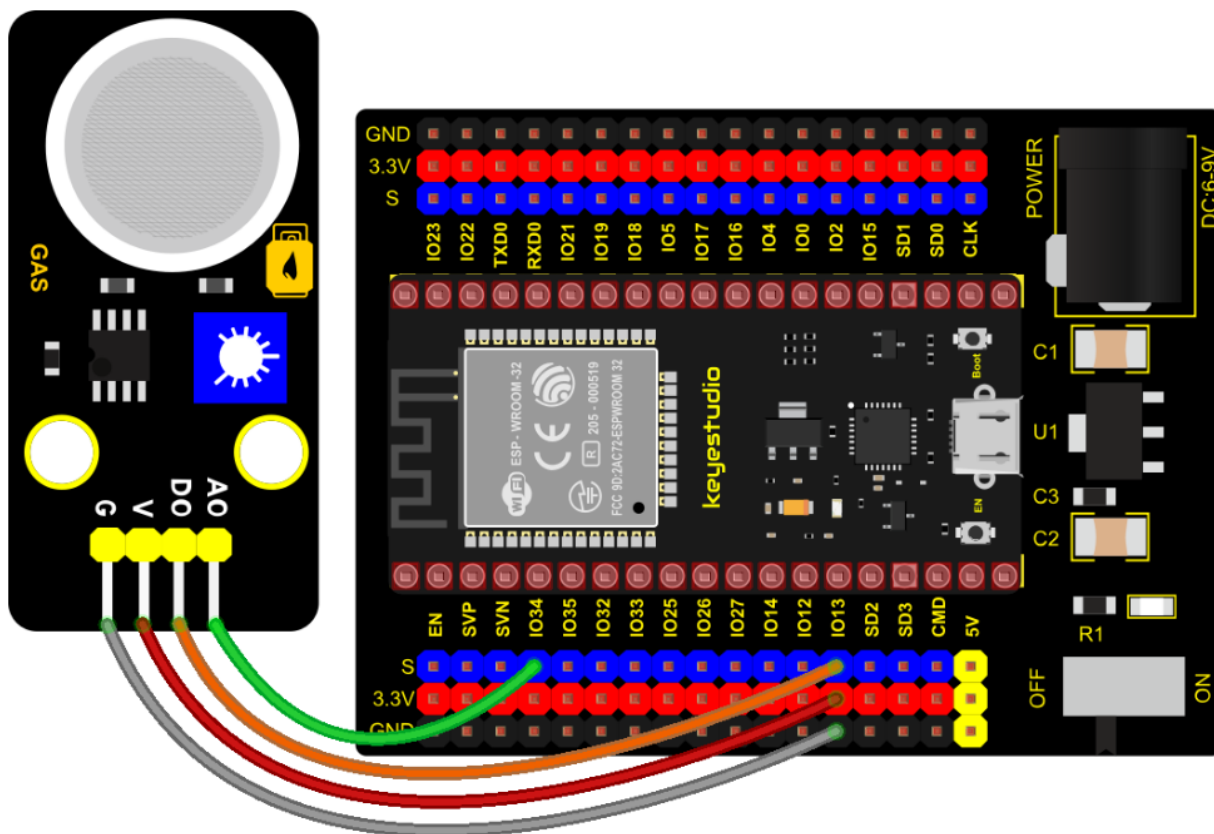


Рис. 2.21. Підключення MQ2 до плати KEYESTUDIO ESP32 PLUS

2.1.3. Підсистема виконавчих механізмів (Актуотори)

2.1.3.1. Пасивний п'єзоелектричний модуль дзвінка (Passive Buzzer)

Пасивний дзвінок (Passive Buzzer Module) — це звуковий модуль, який використовується для генерації звукових сигналів різної частоти. На відміну від активного дзвінка, пасивний не має вбудованого генератора, тому для створення звуку необхідно подавати на нього імпульсний сигнал певної частоти від мікроконтролера.

Основні характеристики

- Тип: пасивний п'єзоелектричний дзвінок.
- Робоча напруга: 3.3–5 В.
- Можливість відтворення різних тонів і мелодій.
- Низьке енергоспоживання.
- Сумісність з ESP32 та Arduino.

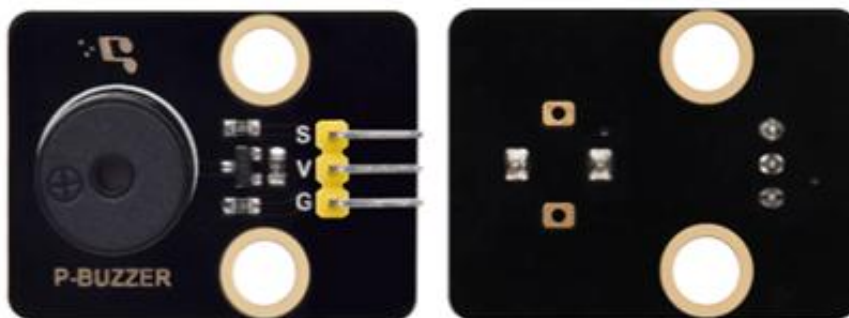


Рис. 2.22. Passive Buzzer Module

Пасивний дзвінок не може вібрувати, випромінюючи сам звук, якщо не подається сигнал квадратної хвилі з певною частотою. Крім того, випромінюючий звук змінюється залежно від частоти квадратної хвилі, тому пасивний дзвінок може імітувати мелодії.

Аналогову хвилю сквайра можна генерувати, змінюючи рівень потужності на контактах. Наприклад, після того, як високий рівень, що триває 500 мс, він змінюється на низький ще на 500 мс, а потім знову на високий.

Ми приводимо сигнал через хвилю сквайра в межах 200~5000 Гц і можемо обчислити частоту(f): $f=1/T$, T — це період (загальний час високого та низького рівнів).

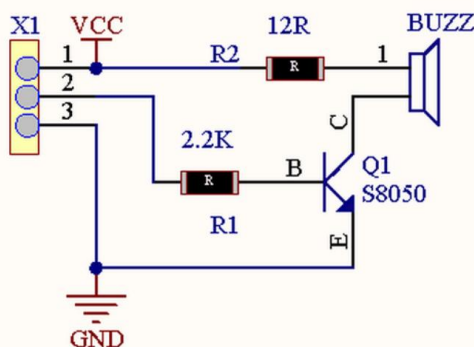


Рис. 2.23. Схема Passive Buzzer Module

Таблиця 2.6

Підключення Passive Buzzer Module до KEYESTUDIO ESP32 PLUS

Модульний Pin	Колір дроту	ESP32 Pin плати
V	ЧЕРВОНИЙ	V
G	ЧОРНИЙ	G
S	ЖОВТИЙ	io5

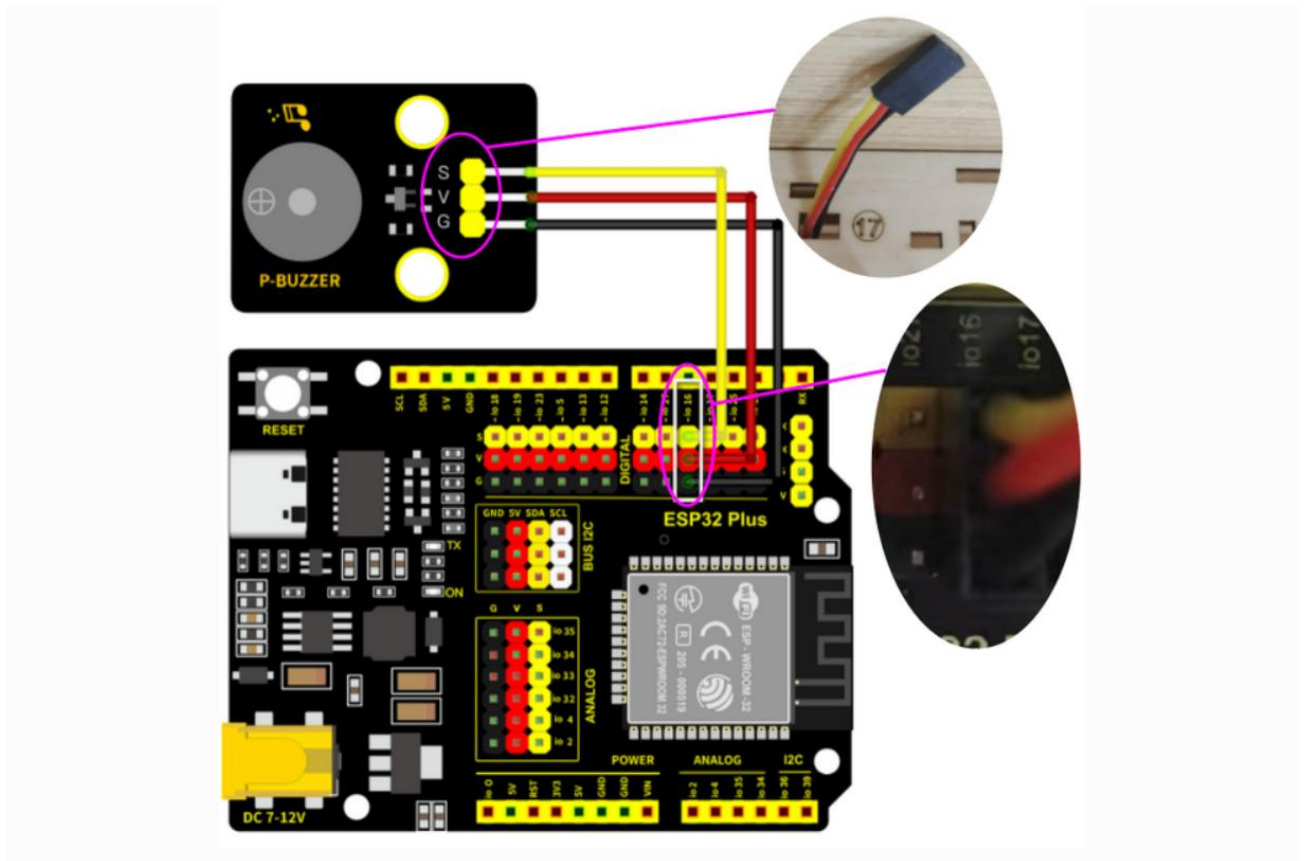


Рис. 2.24. Підключення Passive Buzzer Module до KEYESTUDIO ESP32 PLUS

2.1.3.2. Модуль світлодіода Keyestudio LED Module

Модуль світлодіода Keyestudio LED Module призначений для візуальної індикації стану роботи системи. Він містить світлодіод та необхідні елементи для безпечного підключення до мікроконтролера, зокрема плати KEYESTUDIO ESP32 PLUS.

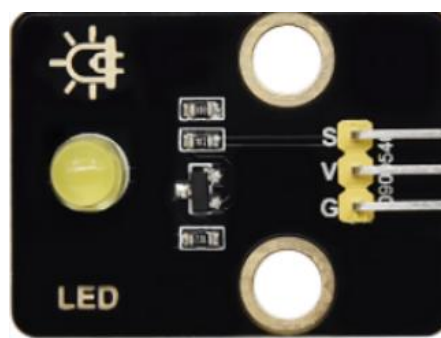


Рис. 2.25. Модуль світлодіода Keyestudio LED Module

Основні характеристики

- Робоча напруга: 3.3–5 В.
- Інтерфейс підключення: S (Signal), V (VCC), G (GND).
- Низьке енергоспоживання.

- Простота інтеграції з платформами ESP32 та Arduino.

Мікроконтролер ESP32 подає цифровий сигнал на контакт S. При подачі логічної одиниці (HIGH) світлодіод загоряється, а при логічному нулі (LOW) — вимикається. Таким чином користувач може швидко оцінити поточний стан системи без підключення до програмного інтерфейсу.

Таблиця 2.7

Підключення до KEYESTUDIO ESP32 PLUS

Модульний Pin	Колір дроту	ESP32 Pin плати
V	ЧЕРВОНИЙ	V
G	ЧОРНИЙ	G
S	ЖОВТИЙ	io5

Переваги використання

- наочна індикація роботи системи;
- швидке виявлення несправностей;
- простота монтажу та програмування;
- сумісність із навчальними та промисловими IoT-проектами.

У системі «Розумний будинок» модуль світлодіода виконує роль простого та надійного засобу оповіщення користувача про стан датчиків.

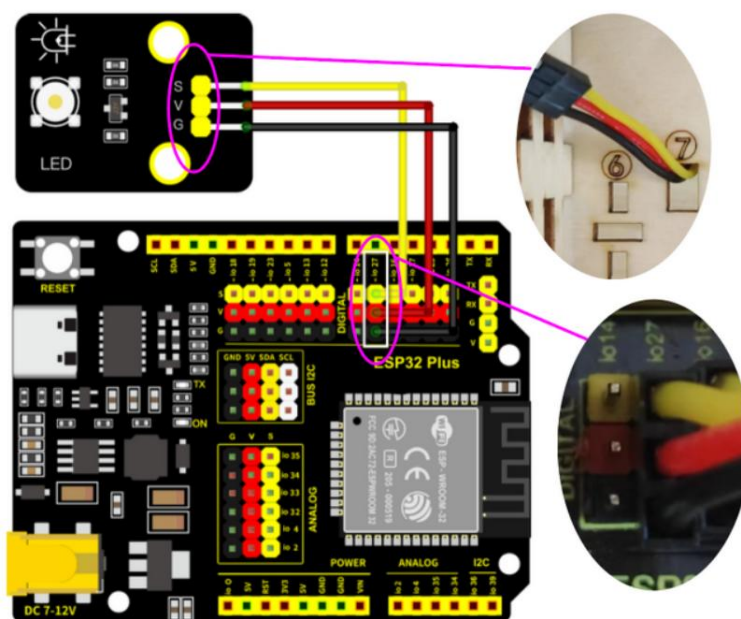


Рис. 2.26. Підключення світлодіода Keyestudio LED Module до KEYESTUDIO ESP32 PLUS

2.1.3.3. RGB-модуль SK6812

Вбудована мікросхема SK6812 може відображати $256 * 256 * 256$ кольорів, що дозволяє досягти різноманітних ефектів. Керування здійснюється лише через один сигнальний провід. Це інтелектуальне зовнішнє кероване світлодіодне джерело світла зі схемою керування та схемою випромінювання світла. Кожен світлодіодний елемент аналогічний світлодіодній лампі 5050, а кожен компонент є пікселем. На модулі є чотири лампові намистини, що відповідає чотирьом пікселям.



Рис. 2.27. RGB-модуль SK6812

Принцип трьох основних кольорів

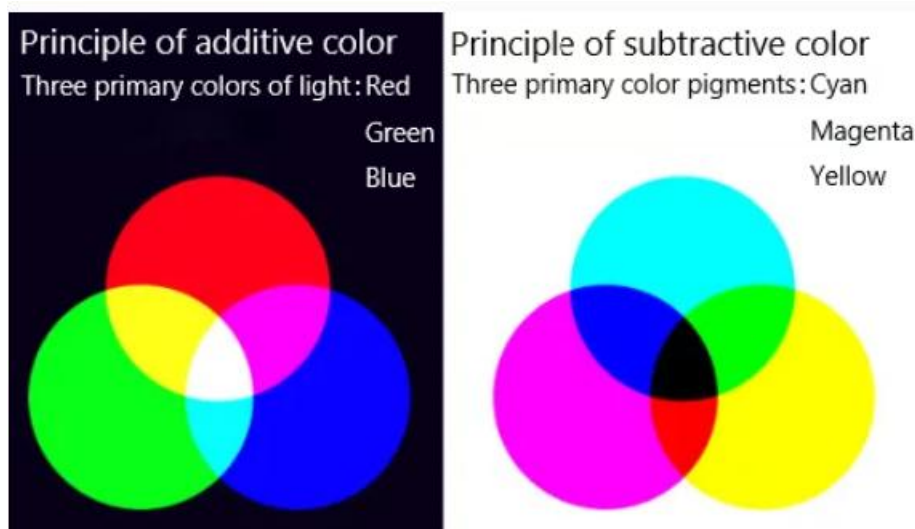


Рис. 2.28. Принцип трьох основних кольорів

Три основні кольори пігментів – червоний, жовтий і синій, а три основні кольори кольорового світла – червоний, зелений і синій, що називається RGB.

Людське око найбільш чутливе до кольорів RGB, і більшість кольорів можна отримати, комбінуючи кольори RGB у різних пропорціях. Аналогічно,

більшу частину монохроматичного світла також можна розкласти на три кольори RGB. Це найпростіший принцип колориметрії (три основні кольори).

Три основні кольори RGB додаються в різних пропорціях для утворення змішаних кольорів, що називається адитивним змішуванням кольорів. Існує також метод субтрактивного змішування кольорів. Кольори можна додавати та віднімати за потреби. Три основні кольори пігментів не можна перетворити на білий, тоді як три основні кольори кольорового світла можна реалізувати за допомогою оптичних елементів, які отримують шляхом змішування трьох рівних частин червоного, середньо-зеленого та темно-синього.

Параметри.

Робоча напруга: 5 В постійного струму

Максимальна потужність: 1 Вт

Джерело світла: SMD 5050 RGB

Модель IC: 4/WS2811

Рівень сірого: 256

Кут освітлення: 180°

Колір світла: можна налаштувати на білий, червоний, жовтий, синій та зелений за допомогою контролера

Робоча температура: -10°C ~ +50°C

Розміри: 32 x 23,8 x 7,4 мм

Розмір отвору для позиціонування: 4,8 мм у діаметрі

Порт: 3-контактний зігнутий штифт з відстанню 2,54 мм

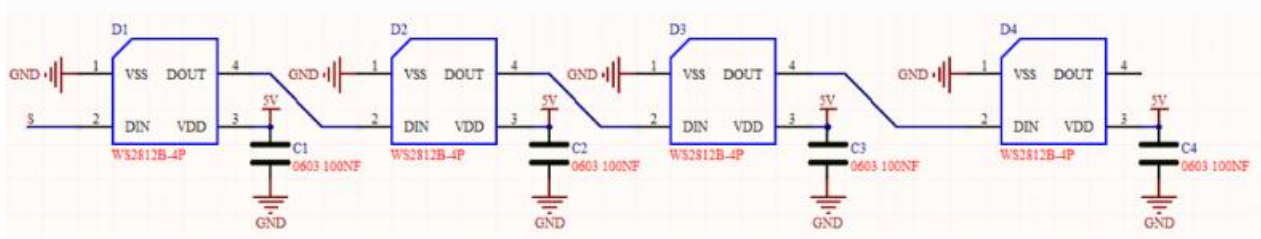


Рис. 2.29. Принципова схема RGB-модуля SK6812

Зі схеми видно, що ці чотири піксельні світлодіодні намістини з'єднані послідовно. Фактично, незалежно від їх кількості, ми можемо використовувати контакт для керування світлом і дозволити йому відображати будь-який колір.

Піксельна точка містить схему керування підсилювачем формування сигналу фіксації даних, високоточний внутрішній генератор і програмований блок керування постійним струмом 12 В високої напруги, що ефективно забезпечує високу стабільність кольору світла піксельної точки.

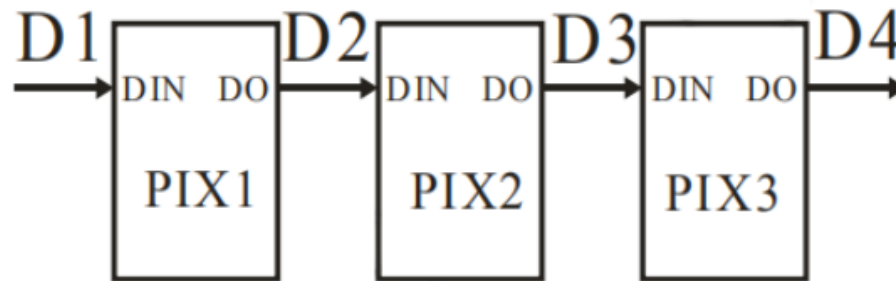


Рис. 2.30. Протокол передачі даних

Протокол передачі даних використовує метод зв'язку на основі однополярного коду нульового відображення. Після скидання точки пікселя DIN-термінал приймає дані, передані від контролера. Перші 24-бітні дані витягуються першою точкою пікселя та надсилаються на фіксатор даних всередині точки пікселя. Решта даних формується та посилюється внутрішньою схемою обробки формування, а потім починає пересилатися та виводитися на наступний каскад точок пікселя через порт DO. Після передачі однієї точки пікселя сигнал зменшується на 24 біти.

Піксельна точка використовує технологію автоматичного формування та переадресації, завдяки чому кількість каскадів піксельної точки не залежить від обмеження передачі сигналу, а залежить лише від обмеження швидкості передачі сигналу.

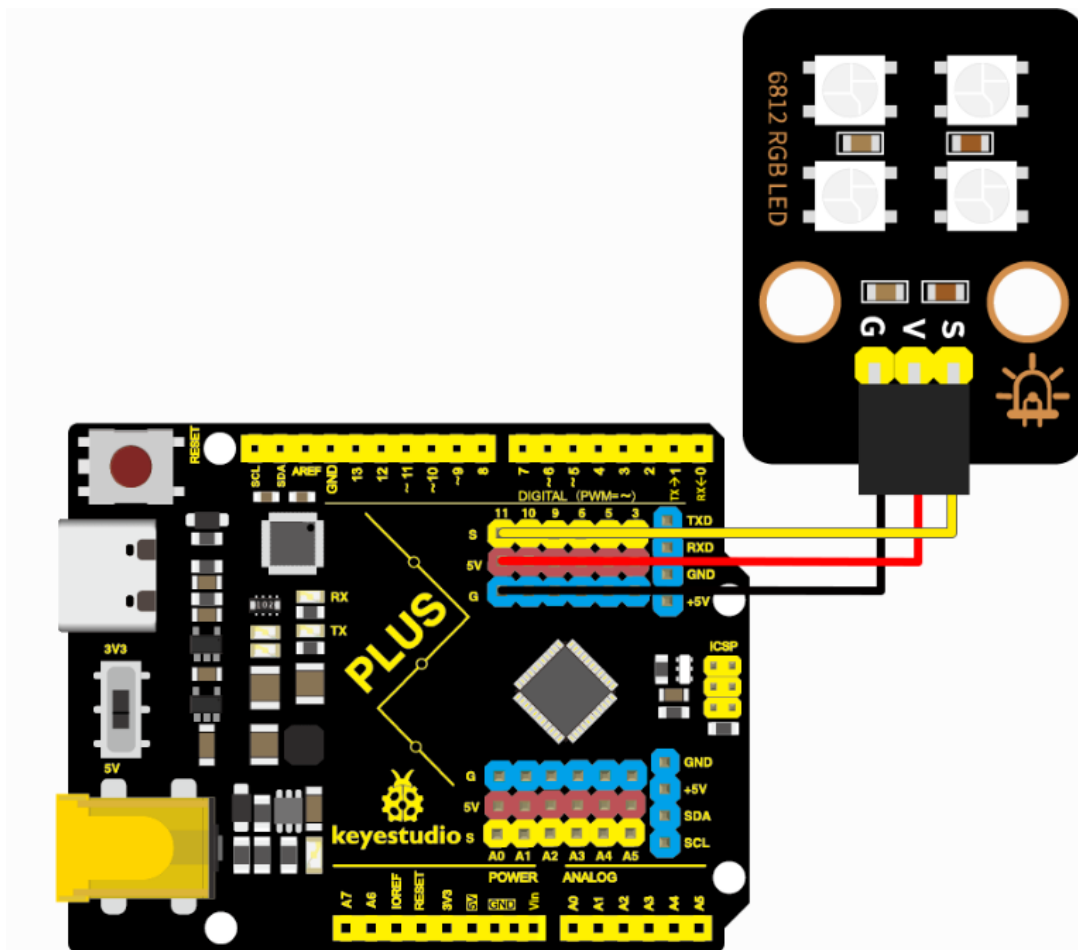


Рис. 2.31. Підключення RGB-модуля SK6812 до KEYESTUDIO ESP32 PLUS

2.1.3.4. Мікросервопривід SG90

Для забезпечення рухомості дверей було обрано сервопривід SG-90 (рис. 2.32), який відповідає вимогам за габаритами та вантажопідйомністю, а також відрізняється доступною вартістю.



Рис. 2.32. Мікро сервопривід Tower Pro SG90

Сервопривід - це різновид сервоприводу положення, який в основному складається з корпусу, друкованої плати, безконтактного двигуна, шестерні та датчика положення.

Принцип його роботи полягає в тому, що приймач або мікроконтролер надсилає сигнал до сервоприводу, який має внутрішню схему опорного сигналу, що генерує опорний сигнал з періодом 20 мс та шириною 1,5 мс, і порівнює напругу зміщення постійного струму з напругою потенціометра для отримання різниці вихідних напруг. ІС на друкованій платі визначає напрямок обертання, а потім керує безсердечним двигуном для початку обертання та передає потужність на поворотний важіль через редуктор, тоді як датчик положення надсилає сигнал назад, щоб визначити, чи досягнуто позиціонування.

Він підходить для тих систем керування, які потребують постійної зміни кута та можуть бути обслуговуваними.

Коли двигун обертається з певною швидкістю, потенціометр обертається каскадним редуктором так, що різниця напруг дорівнює 0, і двигун зупиняється. Діапазон кута загального обертання сервоприводу становить від 0 до 180 градусів.

Період імпульсу для керування сервоприводом становить 20 мс, ширина імпульсу — від 0,5 мс до 2,5 мс, а відповідне положення — від -90 градусів до +90 градусів. Нижче наведено приклад сервоприводу з кутом повороту 180 градусів:

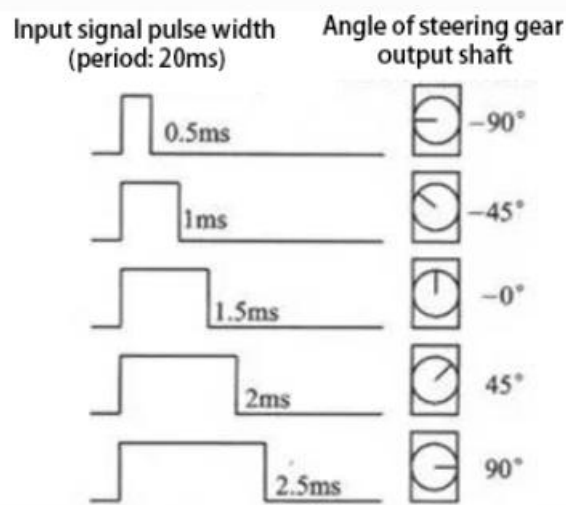


Рис. 2.33. Кути обертання

Серводвигуни мають багато специфікацій, але всі вони мають три з'єднувальні дроти: коричневий, червоний та помаранчевий (різні марки можуть мати різні кольори). Коричневий – це заземлення, червоний – позитивний полюс живлення, а помаранчевий – сигнальна лінія.

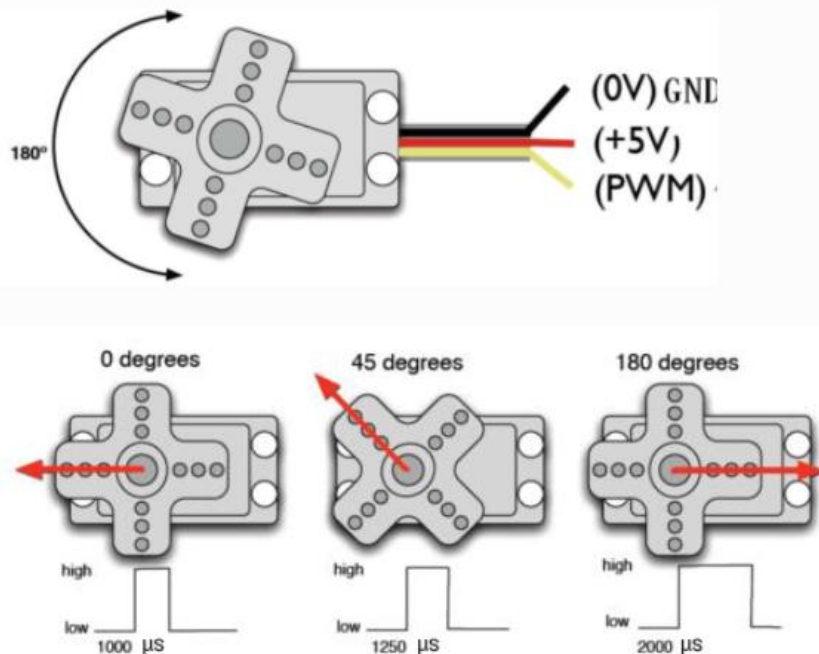


Рис. 2.34. Принцип роботи

Кут повороту серводвигуна контролюється шляхом регулювання шпаруватості сигналу ШІМ (широтно-імпульсна модуляція). Стандартний цикл сигналу ШІМ становить 20 мс (50 Гц). Теоретично, ширина розподілена між 1 мс-2 мс, але насправді вона становить від 0,5 мс до 2,5 мс. Ширина відповідає куту повороту від 0° до 180°. Але зверніть увагу, що для двигунів різних марок один і той самий сигнал може мати різні кути повороту.

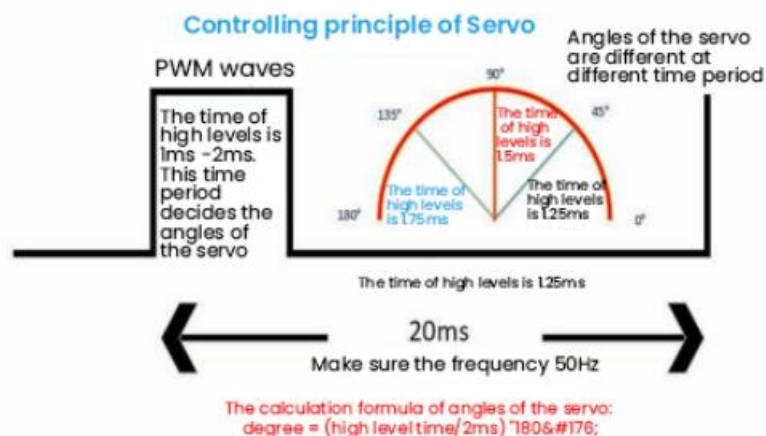


Рис. 2.35. Кут повороту

Технічні характеристики мікро сервоприводу Tower Pro SG90

Характеристика	Показник
Вага, г	14,7
Частота, мкс	1520
Тип двигуна	Колекторний
Габаритні розміри, мм	23 x 12 x 28,5
Матеріал корпусу та редуктора	Пластик
Швидкість, сек/60°	0,1 – 0,12
Живлення, В	4,8 – 6
Зусилля, кг/см	1,2 – 2,5

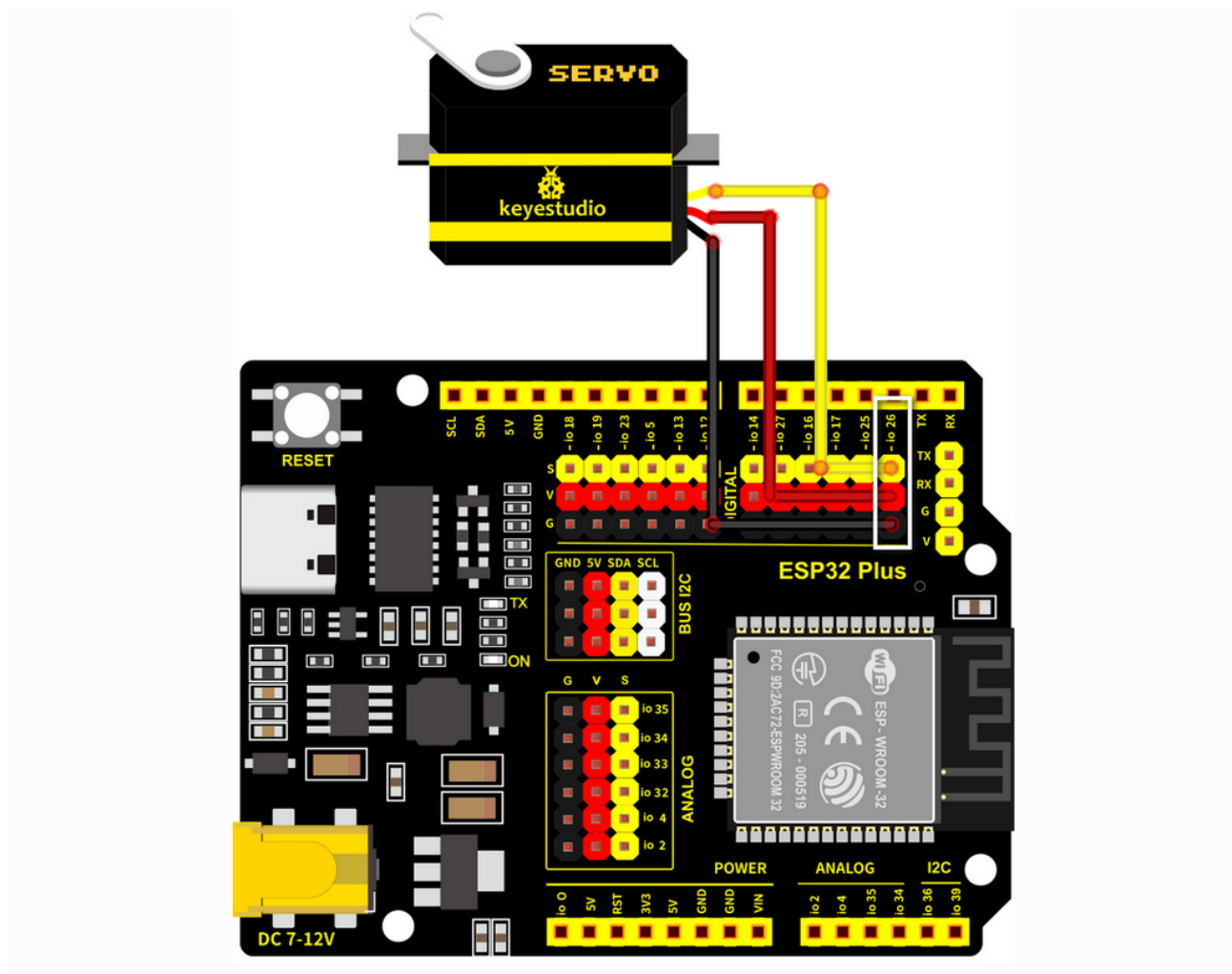


Рис. 2.36. Схема підключення серво до KEYESTUDIO ESP32 PLUS

2.1.3.5. Рідкокристалічний дисплей LCD 1602 (з I2C-адаптером)

Рідкокристалічні дисплеї LCD 1602 є поширеним вибором для виведення текстової інформації у багатьох електронних проєктах завдяки своїй доступності та широкому спектру модифікацій [27]. Класична схема підключення LCD 1602 вимагає використання мінімум 6 цифрових виводів мікроконтролера, що може бути значним обмеженням для проєктів з обмеженою кількістю доступних портів. Це пов'язано з тим, що для керування кожним сегментом символів на дисплеї потрібен окремий сигнал керування.

Для подолання цього обмеження часто використовують LCD модулі з інтерфейсом I2C. Цей інтерфейс дозволяє підключити дисплей до мікроконтролера лише за допомогою двох проводів (SCL і SDA), суттєво зменшуючи кількість зайнятих портів. Це робить LCD дисплеї з інтерфейсом I2C більш універсальними та зручними у використанні з платформами ESP 32.

Характеристики:

- Роз'єм: RJ11
- Сумісність із бібліотекою Arduino LiquidCrystal
- Білий текст на блакитному тлі
- Ширина 16 символів, 2 рядки
- Адреса I2C: 0x27
- Підсвічування: синє
- Колір тексту: Білий
- Напруга живлення: 5 В
- Регулювання контрастності: потенціометр
- Розміри: 99мм * 37мм * 21мм
- Вага: 37.4 г

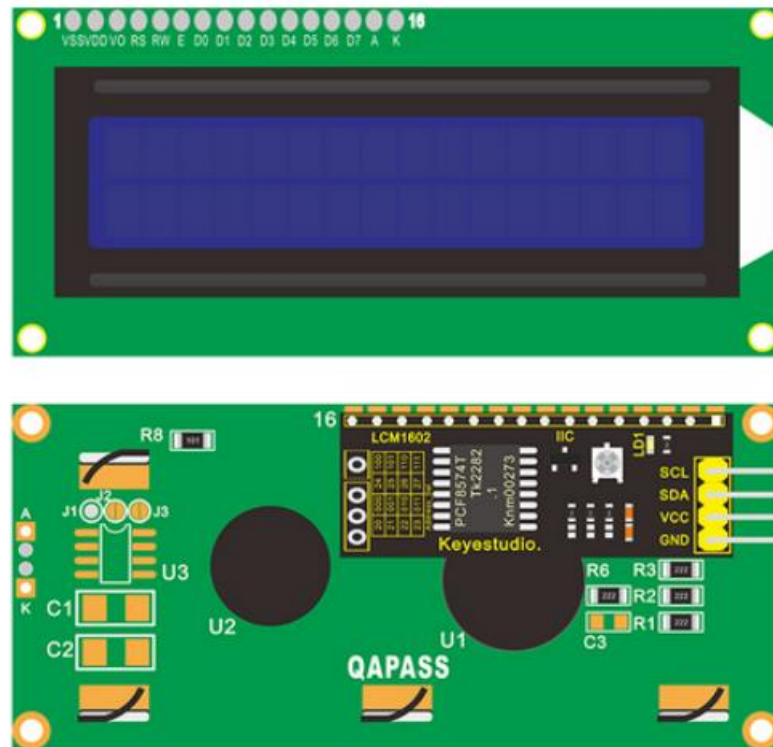


Рис. 2.37. Дисплей LCD 1602

Кожен з виводів рідкокристалічного дисплея відіграє критичну роль у його функціонуванні:

- **GND (земля):** слугує точкою відліку для електричних потенціалів у схемі.
- **5 В (живлення):** забезпечує необхідну напругу для роботи електроніки дисплея.
- **Контраст:** дозволяє регулювати яскравість зображення шляхом зміни напруги на цьому виводі.
- **RS (Register Select):** визначає, чи буде наступна передана інформація інтерпретована як команда (наприклад, встановити курсор) чи як дані для відображення на екрані.
- **RW (Read/Write):** використовувався в старих моделях дисплеїв для читання даних з дисплея. В сучасних моделях зазвичай заземлений.
- **Enable (E):** Сигнал, який інформує дисплей про те, що дані на лініях даних готові для прийому.
- **DB0-DB7 (Data Bits):** група з 8 ліній, за якими передаються біти даних, що визначають, які сегменти рідких кристалів мають бути увімкнені для формування зображення.

- **А (анод) та К (катод):** виводи для підключення зовнішнього джерела живлення для підсвічування дисплея.

Рідкокристалічний дисплей (LCD) має наступні технічні характеристики:

- Тип відображення: символів, з можливістю завантаження користувацьких символів.
- Підсвічування: світлодіодне, що забезпечує яскраве та рівномірне підсвічування символів.
- Контролер: HD44780, що відповідає за керування роботою дисплея.
- Живлення: 5 вольт постійного струму.
- Розмір дисплея: 16 символів по ширині та 2 рядки по висоті.
- Робоча температура: від -20°C до +70°C.
- Температура зберігання: від -30°C до +80°C.
- Кут огляду: 180 градусів, що забезпечує гарну видимість з різних кутів.

Стандартна схема приєднання монітора безпосередньо до KEYESTUDIO ESP32 PLUS зображена на рисунку 2.38.

Підключіть червоний до V, чорний до G, синій з SDA, зелений до SCL.

Таблиця 2.9

Підключення LCD 1602 до KEYESTUDIO ESP32 PLUS

Модульний Pin	Колір дроту	ESP32 Pin плати
V	ЧЕРВОНИЙ	V
G	ЧОРНИЙ	G
SCL	ЗЕЛЕНИЙ	SCL
SDA	СИНІЙ	SDA

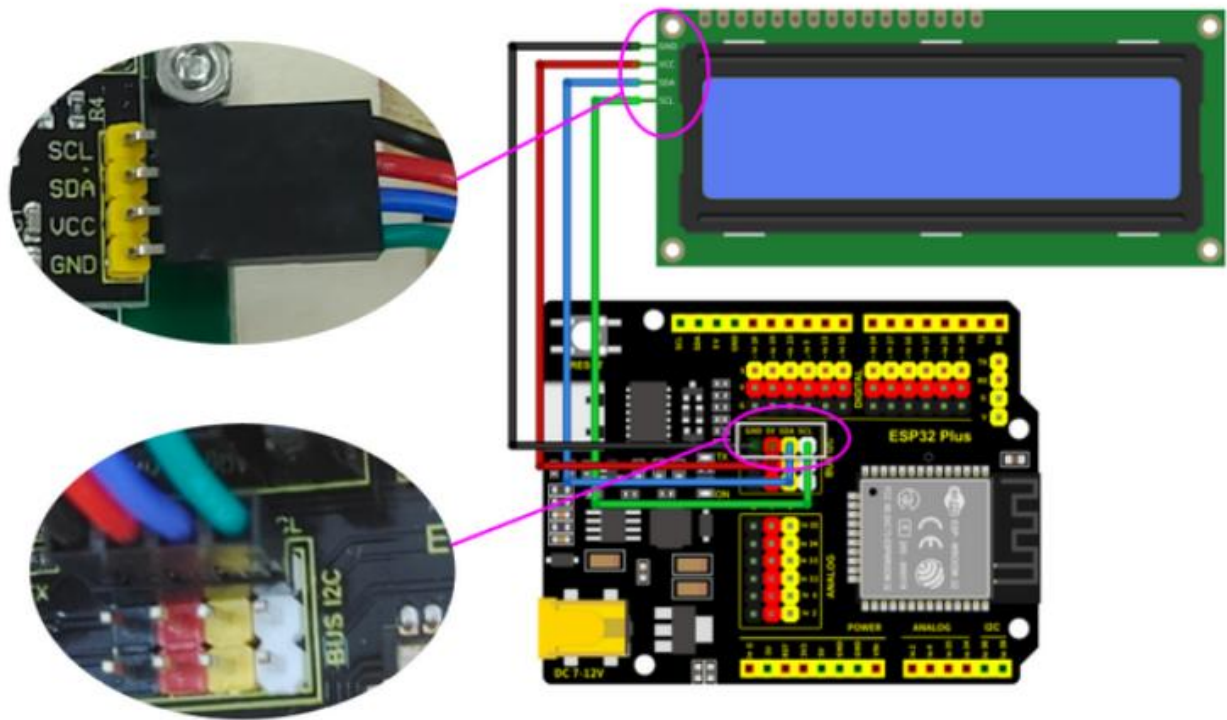


Рис. 2.38. Схема підключення LCD 1602 до KEYESTUDIO ESP32 PLUS

2.1.3.6. Модуль двигуна постійного струму (Малий вентилятор)

Маленький вентилятор використовує мотор на 130 постійного струму та безпечні лопаті вентилятора. Можна використовувати PWM Вихід для керування швидкістю вентилятора.

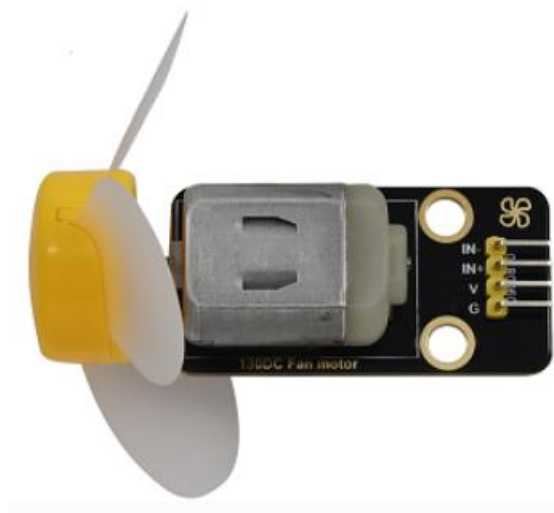


Рис. 2.39. Малий вентилятор

Мотор на 130 постійного струму використовує мікросхему керування двигуном HR1124S, яка є одноканальною мікросхемою драйвера Н-моста, що використовується в рішеннях для двигунів постійного струму. У керуючій частині Н-моста використовуються силові лампи PMOS та NMOS з низьким

опором увімкнення, що забезпечує низькі втрати потужності мікросхеми та безпечну роботу мікросхеми протягом тривалого часу. Крім того, HR1124S підтримує низький струм очікування та низький статичний робочий струм, що робить модуль двигуна 130 простим у використанні.

130 параметрів двигуна:

- Робоча напруга: 5 В
- Робочий струм: ≤ 200 мА
- Робоча потужність: 2 Вт
- Робоча температура: $-10^{\circ}\text{C} \sim +50^{\circ}\text{C}$

Принцип роботи двигуна 130.

Мікросхема HR1124S допомагає керувати двигуном, який не може керуватися тріодом або безпосередньо портом вводу-виводу через великий струм, необхідний двигуну.

Двигун можна обертати, додаючи напругу до обох кінців. Якщо напрямок напруги різний, напрямок обертання двигуна не однаковий. У межах граничної напруги, чим вища напруга, тим швидше обертатиметься двигун; З іншого боку, чим нижча напруга, тим повільніше обертатиметься двигун або зупиниться.

Існує два методи керування: один — високий та низький рівень (керування вмикається та вимикається), а інший — ШІМ (керування швидкістю).

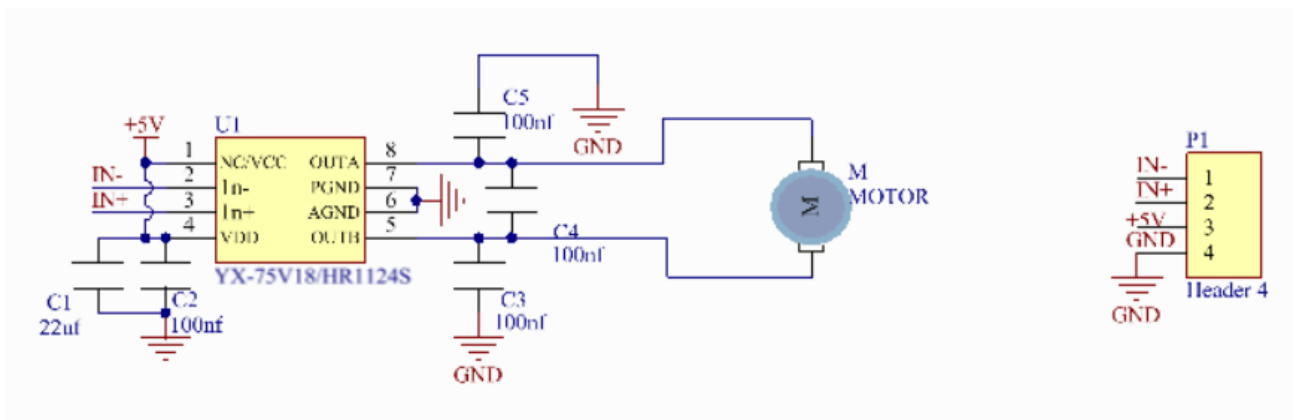


Рис. 2.40. Принципова схема малого вентилятора

Для керування мотором вентилятора потрібні два контакти: один для INA та дві для INB. Діапазон значень PWM становить 0~255. Коли вихід PWM Два контакти — це різниця, вентилятор може обертатися.

Підключення малого вентилятора до KEYESTUDIO ESP32 PLUS

Модульний Pin	Колір дроту	ESP32 Pin плати
Y-	ЗЕЛЕНИЙ	Io18
IN+	СИНИЙ	Io19
V	ЧЕРВОНИЙ	V
G	ЧОРНИЙ	G

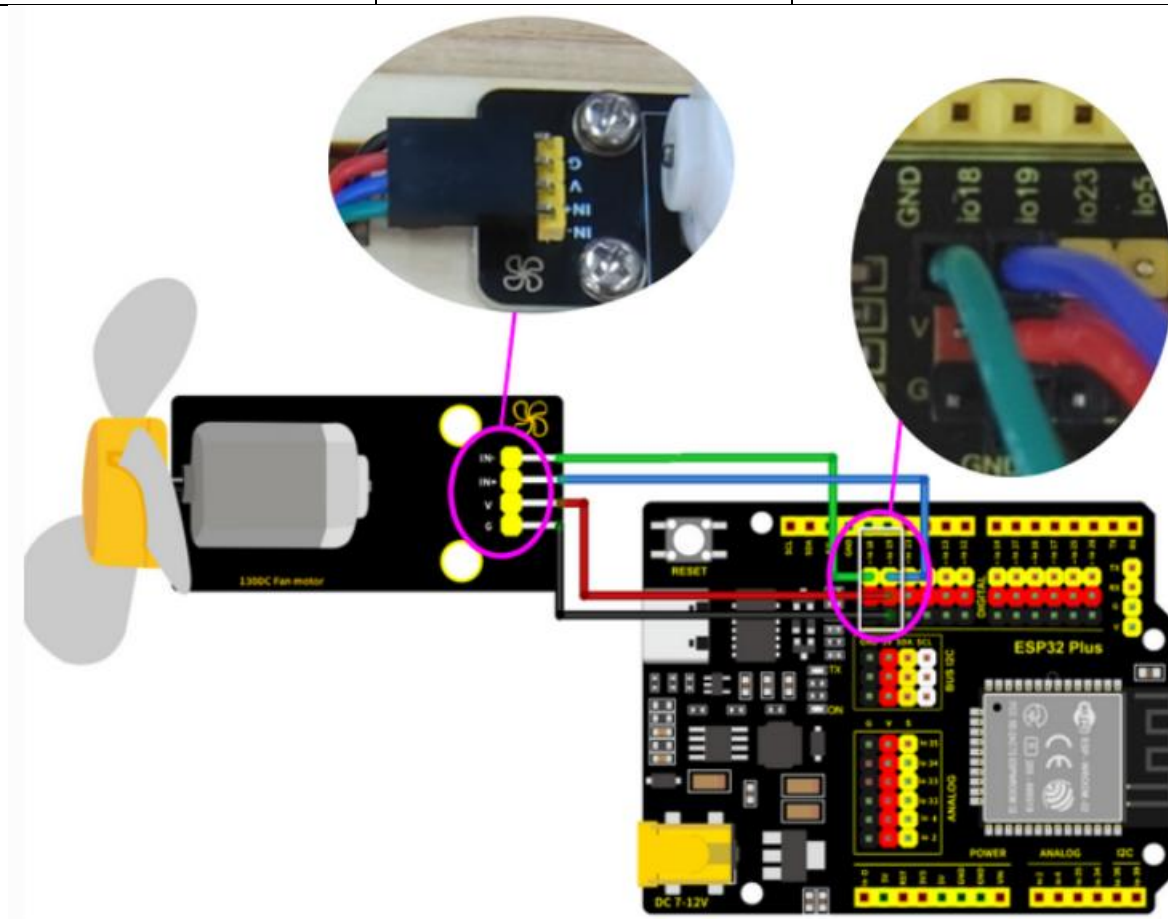


Рис. 2.41. Схема підключення малого вентилятора до KEYESTUDIO ESP32 PLUS

2.2. Обґрунтування вибору мови програмування та середовища розробки

Для розроблення програмного забезпечення проектованої системи було обрано мову програмування Python. Вона є високорівневою об'єктно-орієнтованою мовою програмування, яка поєднує простоту використання, гнучкість та широкі можливості для створення сучасних програмних рішень. Завдяки підтримці об'єктно-орієнтованої парадигми Python забезпечує

ефективне структурування коду, його масштабованість та повторне використання програмних компонентів. Програми, написані цією мовою, часто називають скриптами, оскільки Python широко використовується для створення сценаріїв автоматизації та швидкого розроблення програмних застосунків.

Вибір Python для реалізації програмного забезпечення даного проєкту обумовлений низкою переваг:

- мова забезпечує високу читабельність та зрозумілість коду завдяки чітким правилам синтаксису та єдиному стилю оформлення програм;
- підтримує принципи об'єктно-орієнтованого програмування, що сприяє модульності, повторному використанню та спрощенню супроводу програмного забезпечення;
- дозволяє значно скоротити обсяг програмного коду порівняно з такими мовами, як Java, C++ та іншими, що прискорює процес розроблення;
- є інтерпретованою мовою, тому не потребує тривалого етапу компіляції, забезпечуючи швидке тестування та налагодження програм;
- підтримує кросплатформність, завдяки чому створені програми можуть працювати на різних операційних системах без необхідності внесення суттєвих змін до коду;
- містить потужну стандартну бібліотеку з великою кількістю вбудованих функцій, модулів та інструментів для розроблення програм різного призначення;
- має розвинену екосистему сторонніх бібліотек, що дозволяє легко розширювати функціональні можливості програмного забезпечення та інтегрувати сучасні технології.

Для реалізації алгоритмів керування, опитування сенсорів та координації виконавчих механізмів лабораторного стенду необхідне спеціалізоване інтегроване середовище розробки (IDE — Integrated Development Environment). У цьому проєкті як базове програмне забезпечення обрано кросплатформену

систему Thonny Python IDE, яка орієнтована на розробку додатків мовою Python та її адаптованою для мікроконтролерів версією — MicroPython.

Традиційне програмування мікроконтролерів сімейства ESP32 у середовищі Arduino IDE здійснюється мовою C/C++. Проте використання інтерпретованої мови MicroPython (оптимізованої версії Python 3 для вбудованих систем) має низку суттєвих переваг для побудови навчально-демонстраційного стенду:

- Швидкість розробки (Rapid Prototyping): синтаксис мови Python є лаконічним, що дозволяє значно скоротити обсяг вихідного коду порівняно з C/C++.
- Програмна інтерпретація в реальному часі: MicroPython виконує код безпосередньо на контролері без попередньої тривалої компіляції на ПК. Це дозволяє тестувати окремі команди чи модулі через інтерактивну консоль (REPL — *Read-Eval-Print Loop*).
- Дидактична цінність: MicroPython є сучасним стандартом у сфері навчання IoT-технологій, оскільки абстрагує розробника від низькорівневого керування пам'яттю, фокусуючи увагу на логіці побудови інтернет-речей.

Thonny являє собою інтегроване середовище розробки (IDE), спеціально адаптоване для роботи з мовою Python та її мікроконтролерною версією MicroPython. Середовище підтримує операційні системи Windows, Linux та macOS, що робить його універсальним рішенням для розробників вбудованих систем.

Основними перевагами використання Thonny Python IDE є:

- простий та інтуїтивно зрозумілий інтерфейс користувача;
- підтримка мов Python та MicroPython;
- можливість прямого підключення до мікроконтролерів ESP32 через USB;
- вбудований редактор програмного коду;
- консоль для інтерактивного виконання команд;

- інструменти налагодження та пошуку помилок;
- робота з файловою системою мікроконтролера;
- автоматичне встановлення та оновлення прошивки MicroPython.

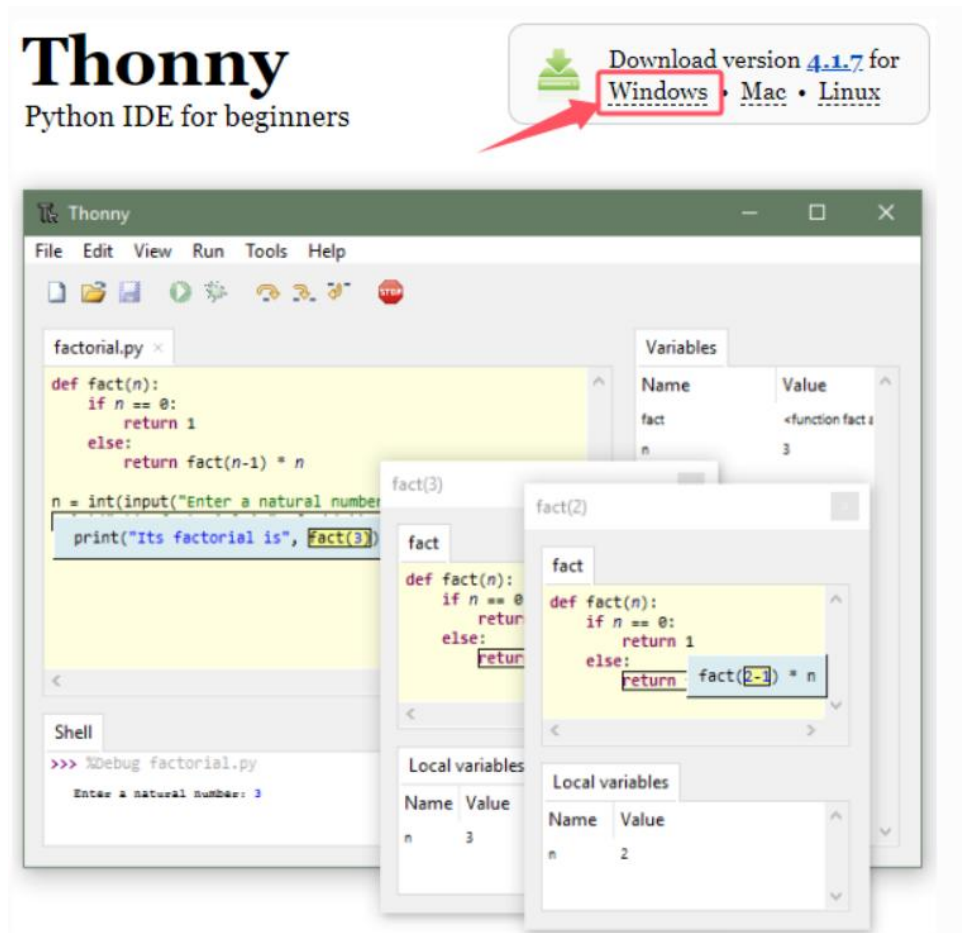


Рис. 2.42. Середовище програмування Thonny Python IDE для написання скриптів на мові Python

Інтерфейс середовища розробки складається з редактора програмного коду, області повідомлень та інтерактивної консолі Shell. У редакторі створюється програмний код, який після перевірки передається до мікроконтролера ESP32 для виконання. Консоль Shell забезпечує обмін даними між користувачем і платою в режимі реального часу, що значно спрощує процес тестування окремих програмних модулів.

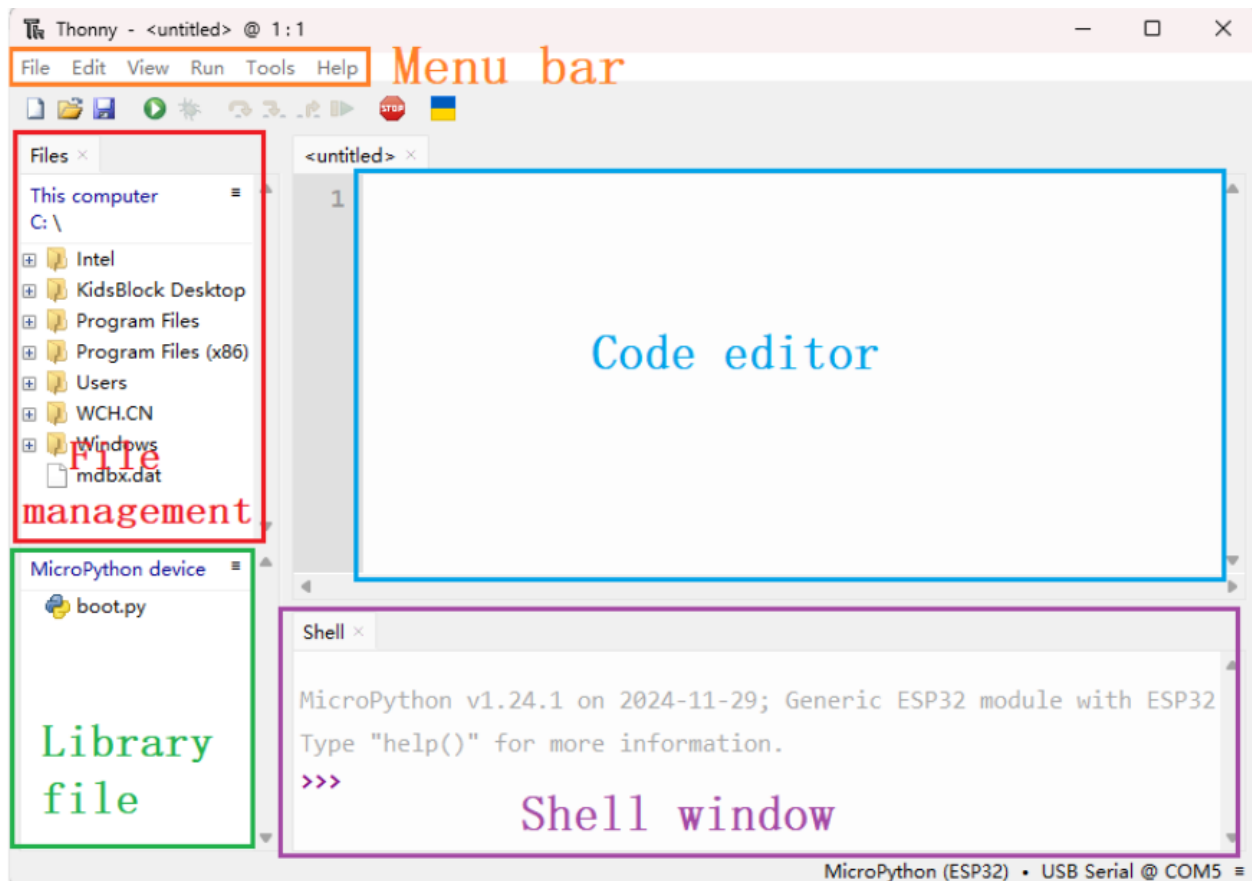


Рис. 2.43. Інтерфейс Thonny

Ключовою особливістю Thonny IDE при роботі з IoT-стендом є вікно REPL (інтерактивна консоль). Через REPL розробник може надсилати поодинокі команди MicroPython безпосередньо в процесор ESP32 через USB-кабель і миттєво отримувати відповідь. Це дозволяє, наприклад, увімкнути пасивний дзвінок (Buzzer) або зчитати поточне аналогове значення з датчика диму MQ-2 однією командою, що суттєво спрощує первинне тестування апаратної частини стенду під час проведення лабораторних занять.

Візуальний інструмент «Plotter» (Графопобудовник), інтегрований у Thonny, здатний у реальному часі будувати графіки на основі числових даних, що надходять з консолі. Це використовується для наочної демонстрації динаміки зміни температури з датчика DHT11 або концентрації газів із сенсора MQ-2.

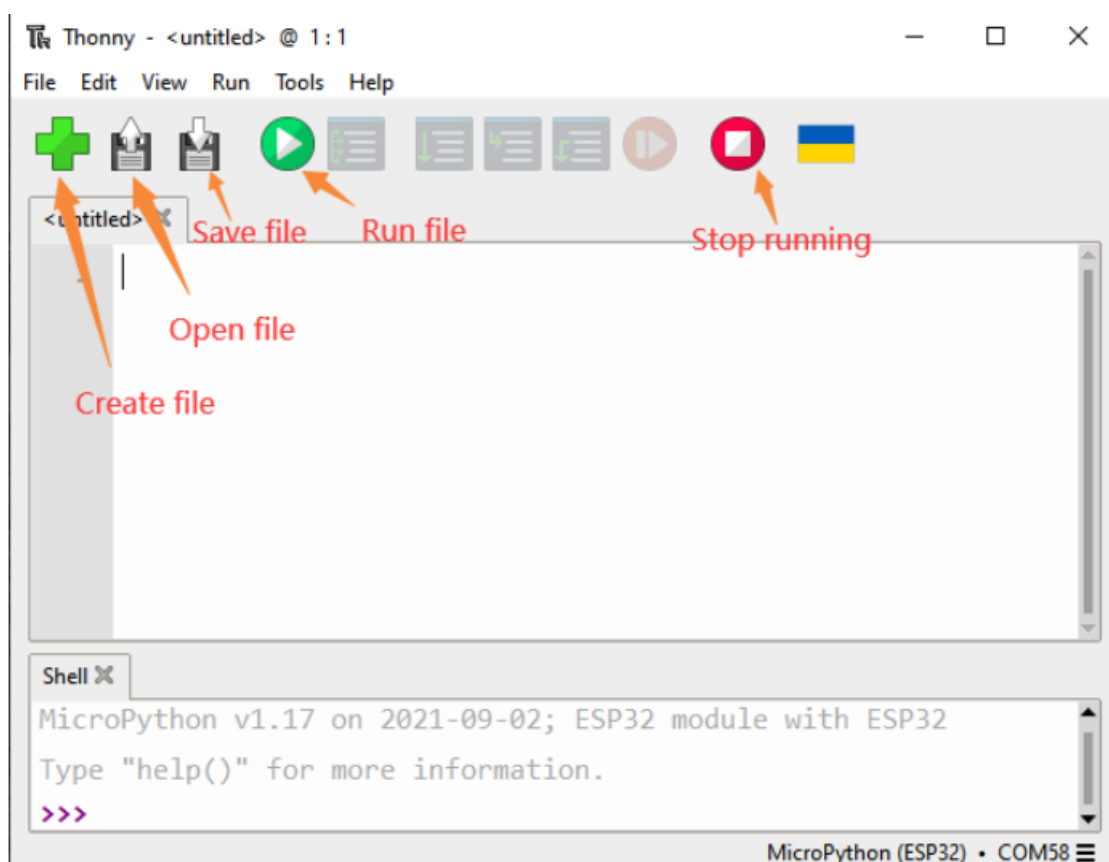
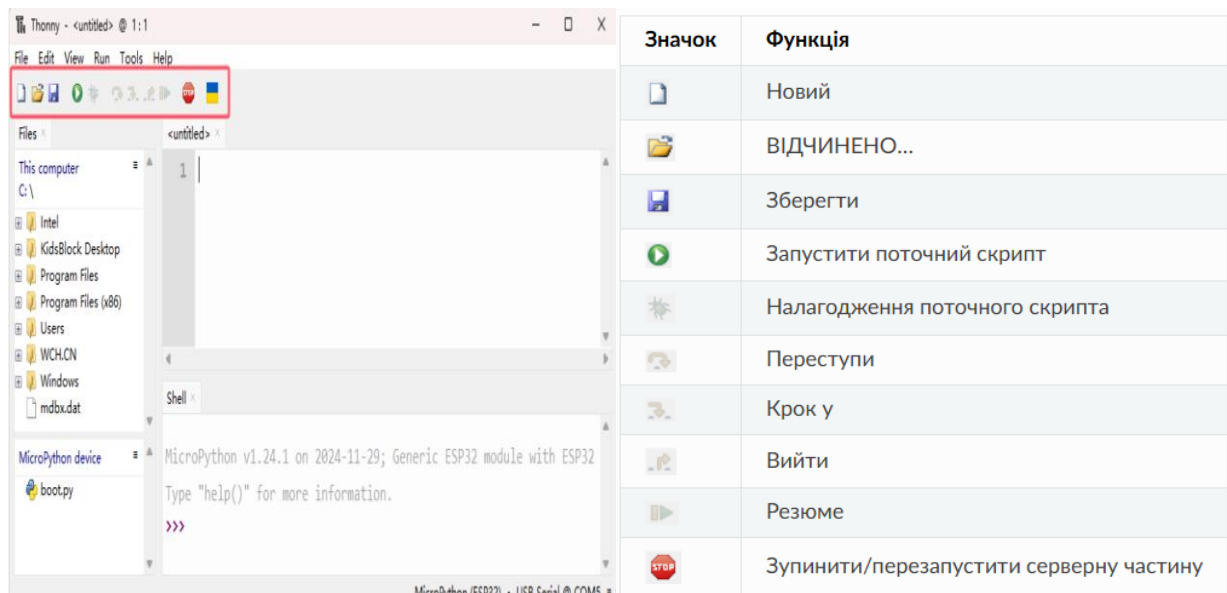


Рис. 2.44. Панель інструментів Thonny

Взаємодія між Thonny IDE та мікроконтролером ESP32 Plus здійснюється через вбудований на платі USB-UART міст (чип-конвертер CP2102 або CH340) за послідовним інтерфейсом. Процес підготовки та розробки складається з двох основних етапів:

1. **Прошивка інтерпретатора (Flashing Firmware).** Перед початком роботи енергонезалежна пам'ять (Flash) мікроконтролера

очищується, і в неї завантажується бінарний файл прошивки MicroPython (інтерпретатор європейського стандарту). У середовищі Thonny це реалізується через інтегрований інструмент діалогового вікна «*MicroPython installer*», який використовує утиліту esptool для запису образу прошивки за адресою 0x1000.

2. **Організація файлової системи та завантаження скриптів.** Після успішної прошивки всередині Flash-пам'яті ESP32 створюється віртуальна файлова система. Thonny IDE надає зручний менеджер файлів, який дозволяє безпосередньо редагувати, створювати та видаляти файли на самому контролері.

Перед початком програмування необхідно встановити інтерпретатор MicroPython у пам'ять мікроконтролера. Для цього плата ESP32 підключається до персонального комп'ютера через USB-кабель. Після запуску середовища Thonny у меню «Інструменти» (Tools) обирається пункт налаштування інтерпретатора. У вікні налаштувань користувач задає тип пристрою ESP32 та відповідний COM-порт, після чого запускає процедуру встановлення прошивки.

Прошивка MicroPython являє собою спеціальне програмне забезпечення, яке забезпечує виконання Python-коду безпосередньо на мікроконтролері. Після запису прошивки ESP32 отримує можливість інтерпретувати команди мови Python та керувати периферійними пристроями без використання компілятора.

Під час завантаження прошивки середовище автоматично стирає попередній вміст флеш-пам'яті мікроконтролера та записує нову версію MicroPython. Після завершення процесу відбувається автоматичне перезавантаження плати та встановлення з'єднання між IDE і мікроконтролером.

Опишемо прошивку KEYESTUDIO ESP32 PLUS.

1. Підключаємо плату ESP32 до комп'ютера через Micro USB.

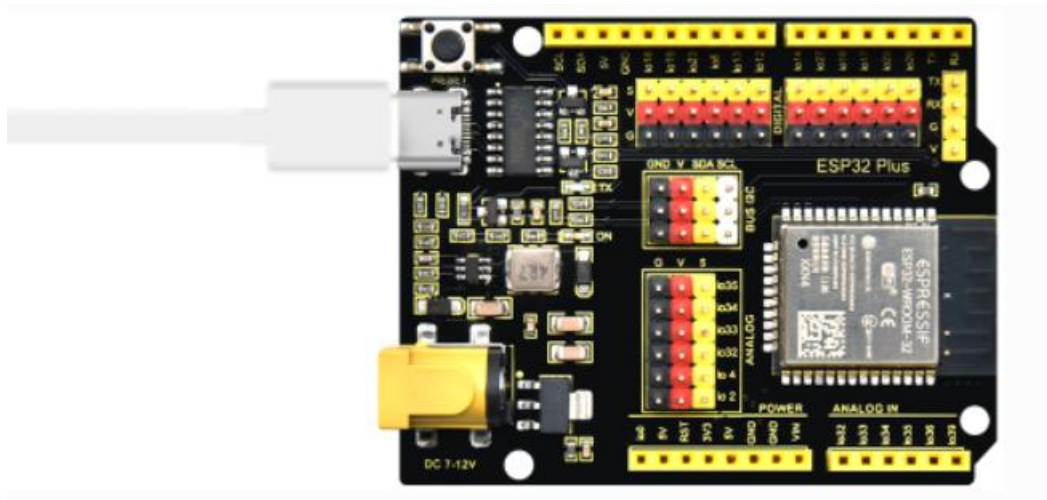


Рис. 2.45. Підключаємо плату ESP32 до комп'ютера

2. Натискаємо кнопку «Виконати», щоб вибрати «Інтерпретатор конфігурації».

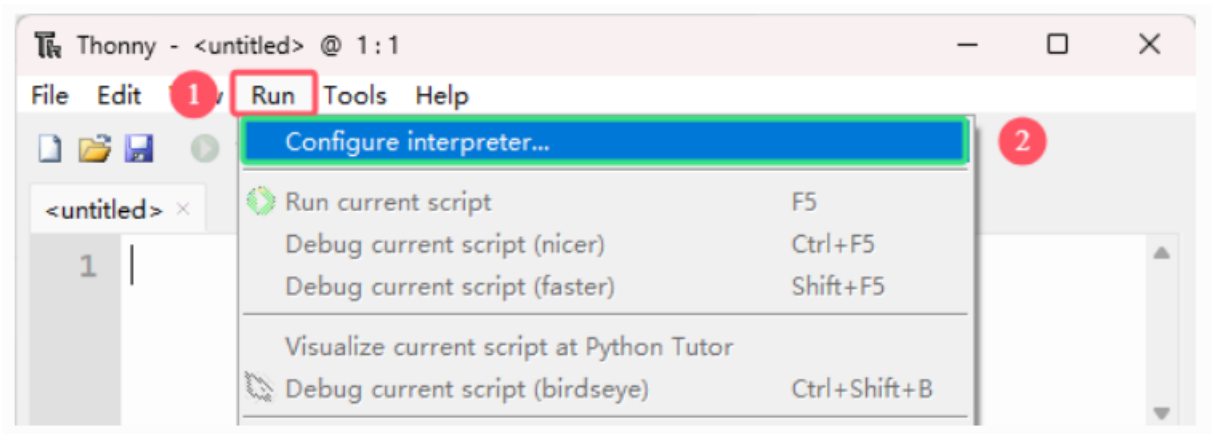


Рис. 2.46. Інтерпретатор конфігурації

3. Обираємо «Інтерпретатор» та «MicroPython (ESP32)», а потім «Встановити або оновити MicroPython (esptool) (UF2)».

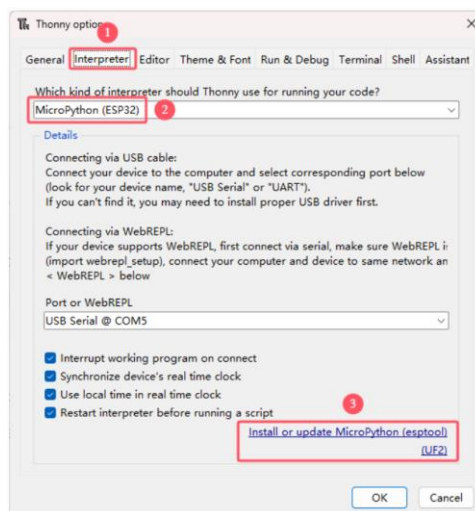


Рис. 2.47. Налаштування інтерпретатора

4. Натискаємо значок «☰», щоб «Вибрати локальний образ MicroPython...». Знаходимо прошивку в папці Firmware esp32 та обираємо «ESP32_GENERIC-20241129-v1.24.1.bin», щоб «Відкрити».

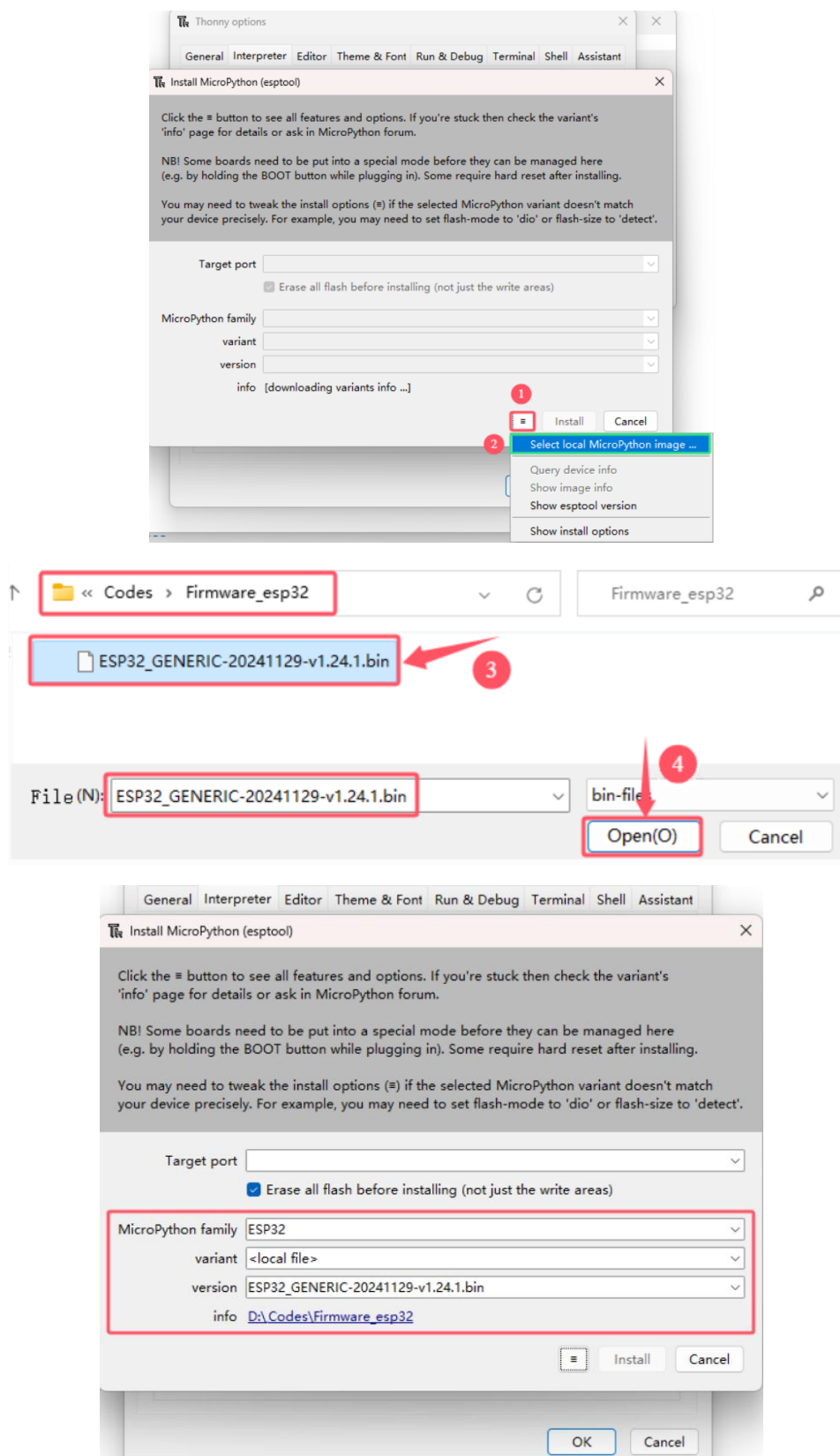


Рис. 2.48. Вибір локального образу MicroPython

5. Обираємо «Послідовний порт USB (COMXX)» (номери послідовних портів (COMXX) відрізняються залежно від плат), щоб «Встановити» його.

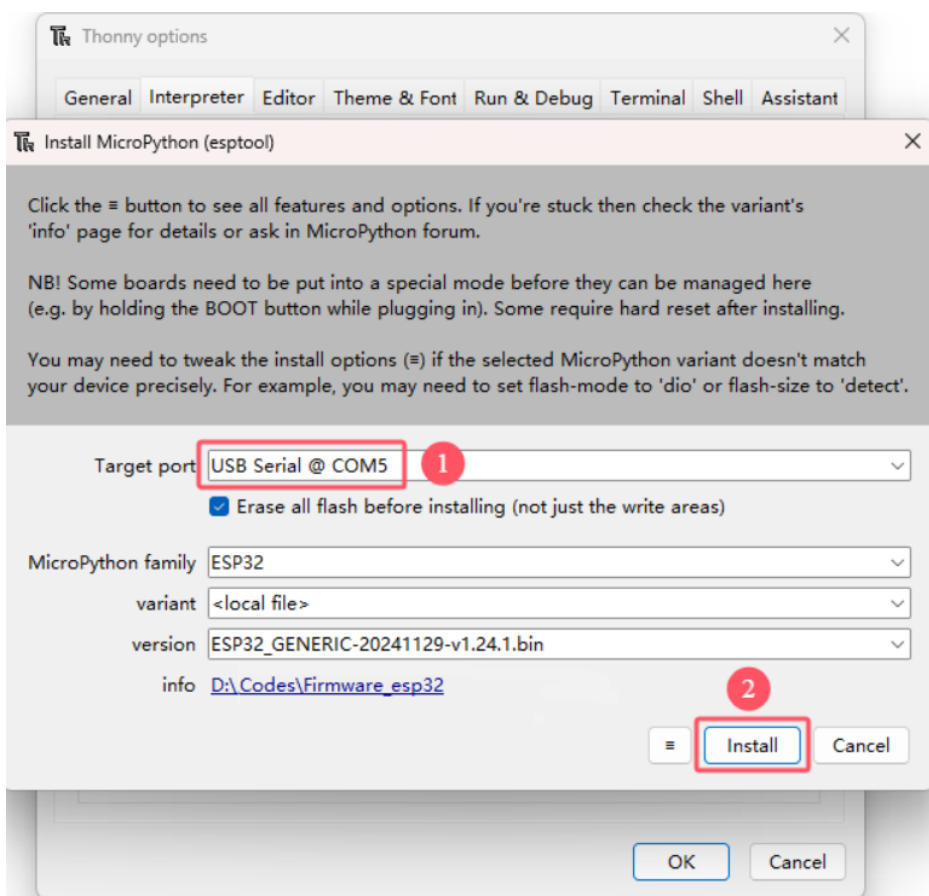


Рис. 2.49. Послідовний порт USB (COMXX)

6. Після встановлення натисніть «Закрити» та «ОК».

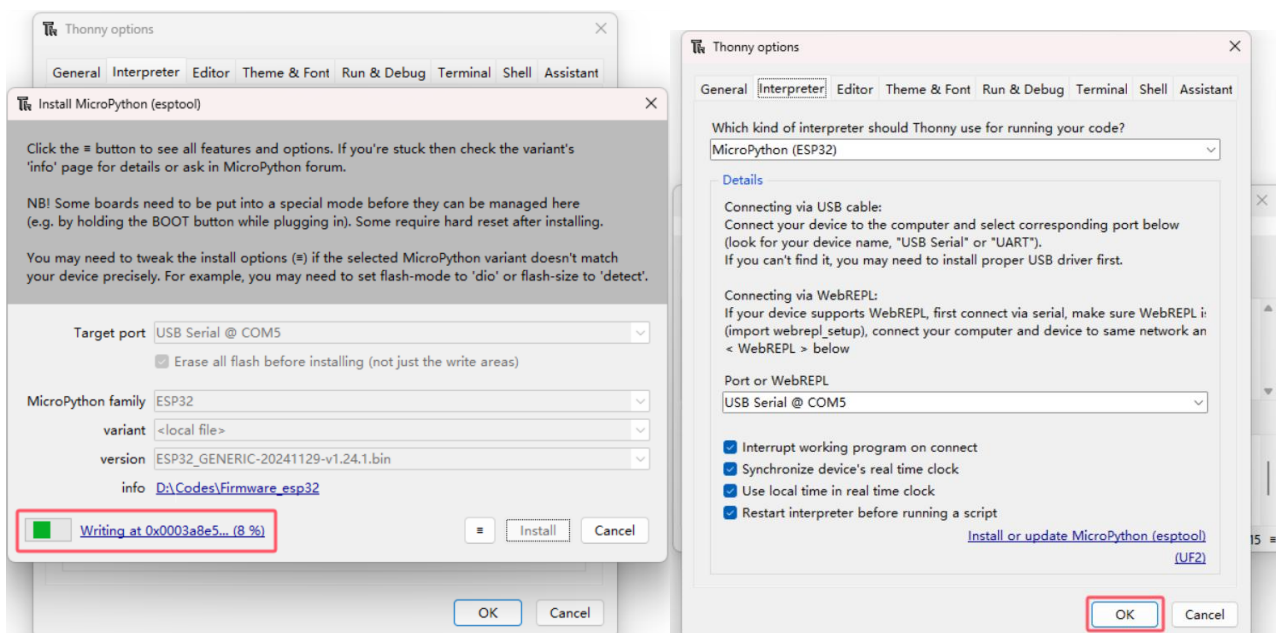


Рис. 2.50. Завершення прошивки

7. Тепер прошивку ESP32 встановлено.

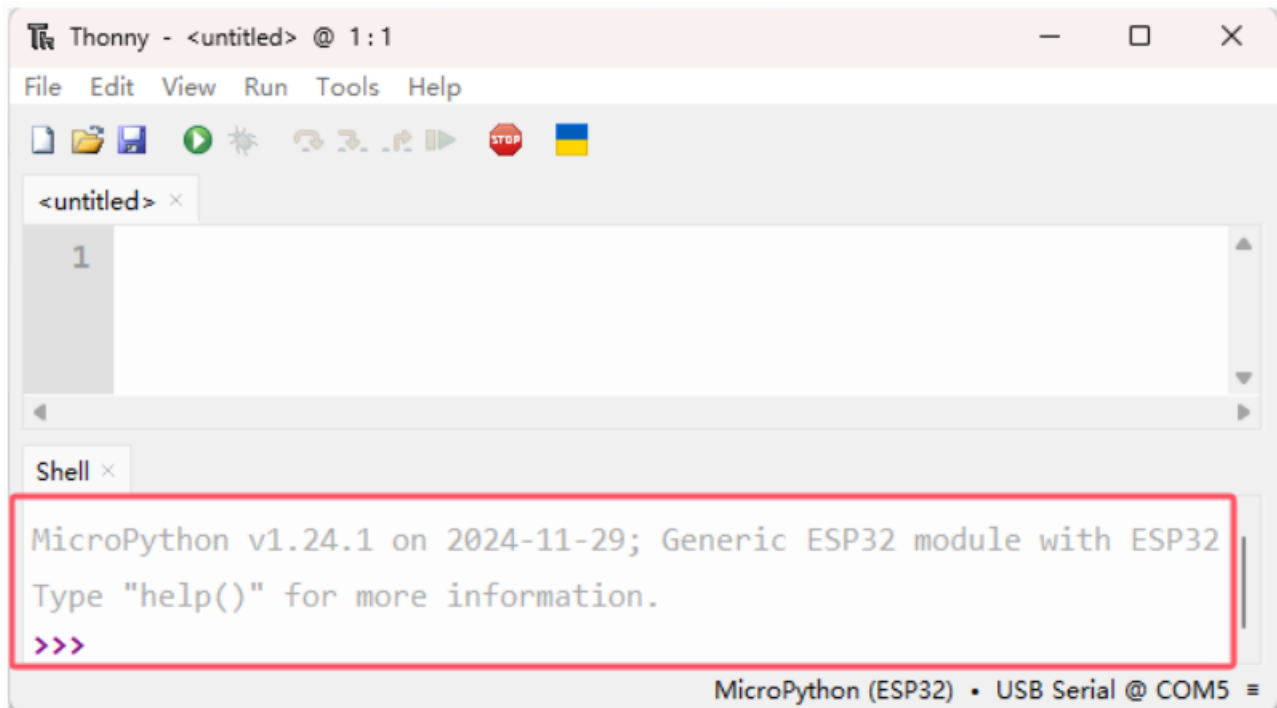


Рис. 2.51. Результат прошивки

Для забезпечення повністю автономної роботи лабораторного стенду «Розумний будинок» (без постійного підключення до ПК) структура файлів у пам'яті ESP32 проектується за чіткою ієрархічною схемою:

- **boot.py** — перший сервісний скрипт, який автоматично викликається ядром MicroPython при кожній подачі живлення або апаратному скиданні (Reset). У ньому реалізуються первинні налаштування системи: ініціалізація підсистеми введення-виведення, конфігурація режимів енергозбереження та запуск алгоритму автоматичного підключення до локальної Wi-Fi мережі.
 - **main.py** — основний файл програми, виконання якого стартує відразу після успішного завершення boot.py. У ньому розгортається безкінечний операційний цикл (while True:), всередині якого відбувається циклічне опитування датчиків (DHT11, MQ-2, Steam Sensor), обробка логічних умов та надсилання команд на актуатори.
 - **Каталог lib/** — системна директорія, призначена для зберігання сторонніх модулів та драйверів периферії (наприклад, файли lcd1602.py або neopixel.py), які імпортуються в головний скрипт за допомогою стандартної директиви import.

Під час розроблення програмного забезпечення для системи керування використовувались стандартні бібліотеки MicroPython для роботи з цифровими та аналоговими входами і виходами, таймерами, інтерфейсами UART, SPI та I²C, а також мережевими можливостями Wi-Fi.

Завдяки підтримці бездротового зв'язку ESP32 може передавати інформацію на вебсервер або мобільний додаток, отримувати команди від користувача та забезпечувати дистанційний моніторинг параметрів системи. Реалізація даних функцій значно спрощується завдяки наявності великої кількості готових бібліотек для MicroPython.

Після написання програми її можна запускати безпосередньо з середовища Thonny або зберігати у файл main.py для автоматичного виконання після перезавантаження мікроконтролера. У разі виникнення помилок середовище відображає повідомлення про місце їх виникнення, що значно прискорює процес налагодження програмного забезпечення.

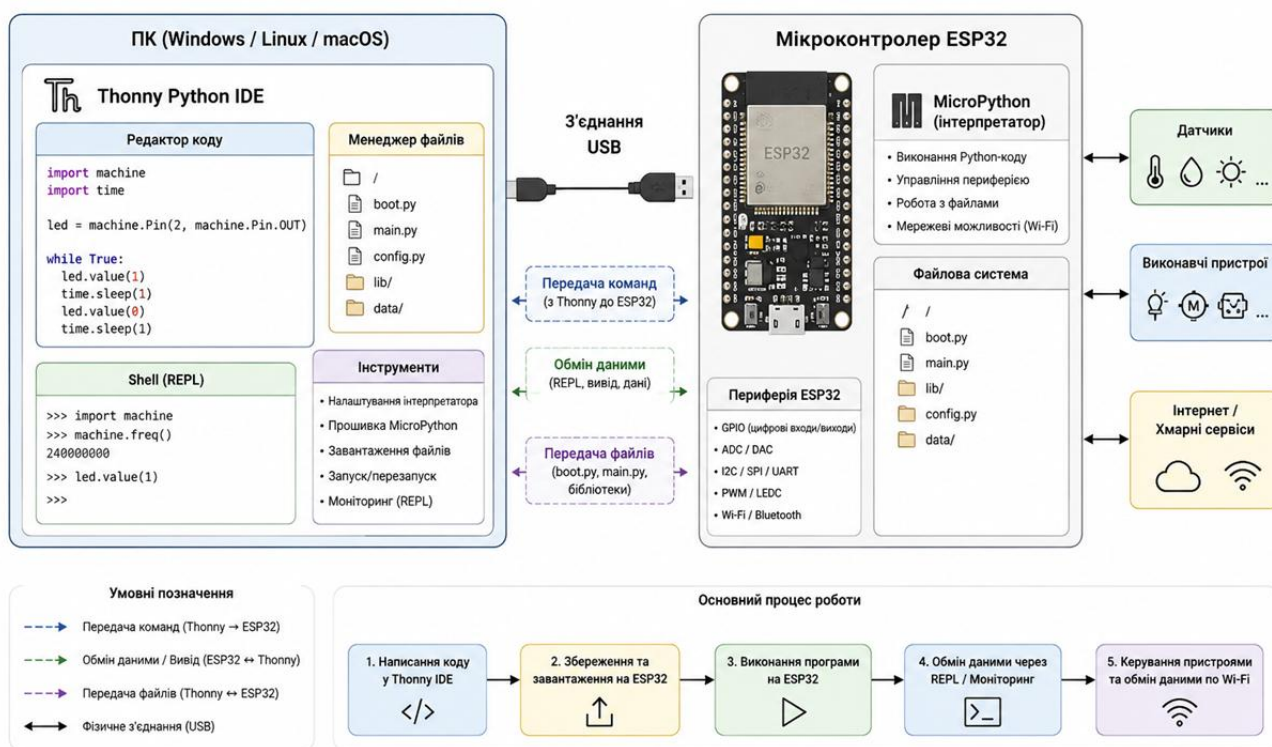


Рис. 2.52. Схему взаємодії середовища Thonny Python IDE з мікроконтролером ESP32

Висновки до розділу

У другому розділі кваліфікаційної роботи виконано комплексний науково-технічний аналіз, обґрунтування та вибір компонентної бази апаратної складової, архітектури бездротового зв'язку, а також програмного забезпечення для розробки лабораторного стенду Internet of Things (IoT) «Розумний будинок».

На основі проведеного дослідження сформовано такі основні висновки:

1. Обґрунтовано вибір центрального обчислювального модуля системи. Як головну керуючу платформу обрано плату розширення Keystudio ESP32 Plus Core Board. Продемонстровано, що інтегрований двоядерний 32-бітний мікроконтролер Xtensa LX6 з тактовою частотою до 240 МГц, вбудованою Flash-пам'яттю об'ємом 4 МБ та апаратними модулями Wi-Fi й Bluetooth є оптимальним інженерним рішенням. Платформа забезпечує необхідну обчислювальну потужність для обробки даних у реальному часі та безпосередній зв'язок із хмарною інфраструктурою без використання додаткових периферійних шлюзів.

2. Спроектовано та структуровано підсистему сенсорів та збору даних. Для комплексного моніторингу параметрів макета приміщення обґрунтовано використання диверсифікованого набору датчиків із різними інтерфейсами передачі сигналів. Інтеграція цифрового датчика мікроклімату DHT11, напівпровідникового датчика газу та диму MQ-2, аналогового датчика водяної пари (Steam Sensor), пасивного інфрачервоного датчика руху PIR HC-SR501, модуля безконтактної автентифікації RFID-RC522 та цифрової кнопки EASY Plug RJ11 дозволяє повноцінно реалізувати кліматичні та безпекові сценарії, а також забезпечує високу дидактичну цінність стенду для вивчення методів цифрової й аналогової обробки сигналів (АЦП, SPI, Single-Bus).

3. Визначено склад підсистеми виконавчих механізмів (актуаторів). Для забезпечення зворотного зв'язку з користувачем та фізичної імітації роботи інженерних систем житла обрано виконавчі пристрої з високим рівнем енергоефективності та модульності. Рідкокристалічний дисплей LCD 1602 (з адаптером I2C) забезпечує локальний вивід інформації; інтелектуальний

адресний RGBW-модуль SK6812 та світлодіод Keyestudio реалізують адаптивне освітлення; пасивний п'єзоелектричний дзвінок (Passive Buzzer) виступає в ролі звукової сирени; мікросервопривід SG90 та модуль малого вентилятора імітують роботу автоматичних дверних замків і систем примусової вентиляції відповідно.

4. Обґрунтовано вибір програмного стеку розробки. Доведено доцільність використання кросплатформеного інтегрованого середовища Thonny Python IDE та операційного середовища MicroPython (оптимізованого стандарту мови Python 3 для вбудованих систем). Використання MicroPython дозволяє реалізувати концепцію швидкого інженерного прототипування (Rapid Prototyping), абстрагуватися від низькорівневого керування пам'яттю за допомогою високого рівня синтаксису, а також використовувати інтерактивну консоль REPL та інструмент Plotter для відлагодження алгоритмів у реальному часі.

РОЗДІЛ 3. ПРОЄКТУВАННЯ, КОДУВАННЯ ТА РОЗГОРТАННЯ ПЕРИФЕРІЙНИХ МОДУЛІВ ІОТ-ПЛАТФОРМИ «РОЗУМНИЙ БУДИНОК»

3.1. Дослідження та налаштування режимів бездротового зв'язку Wi-Fi на базі мікроконтролера ESP32

Найпростіший спосіб отримати доступ до Інтернету — це використати Wi-Fi для підключення. Головна плата керування ESP32 оснащена Wi-Fi-модулем, що робить наш розумний дім легко доступний до Інтернету.

Однією з найкорисніших функцій ESP32 є те, що він може не лише виконувати функції веб-сервера, а й створювати власну мережу для підключення інших пристроїв та доступу до веб-сторінок. ESP32 може працювати у трьох режимах: режим станції (STA), режиму м'якої точки доступу (AP) та режиму станція+точка доступу.

- Режим станції: Активне підключення до роутера як Wi-Fi-пристрою, також відомого як Wi-Fi Client
- Режим точки доступу: Як точка доступу для підключення інших Wi-Fi-пристроїв, тобто Wi-Fi-точок доступу
- Режим станції+точка доступу: Хоча ESP32 підключається до роутера, він також є точкою доступу для інших Wi-Fi-пристроїв.

Усі проєкти програмування Wi-Fi мають бути налаштовані на режим роботи Wi-Fi перед використанням, інакше Wi-Fi не може бути використаний.

3.1.1. Програмна реалізація мережевого підключення платформи у режимі клієнта (STA)

У режимі станції ESP32 підключається до існуючої Wi-Fi-мережі (мережі, створеної бездротовим роутером).

При встановленні режиму станції ESP32 приймається як Wi-Fi-клієнт. Він може підключатися до мережі роутера та спілкуватися з іншими пристроями роутера через Wi-Fi. Як показано на рисунку нижче, ПК і мобільний пристрій підключені до роутера. Якщо ESP32 хоче спілкуватися з ПК і мобільним пристроєм, він має бути підключений до роутера.

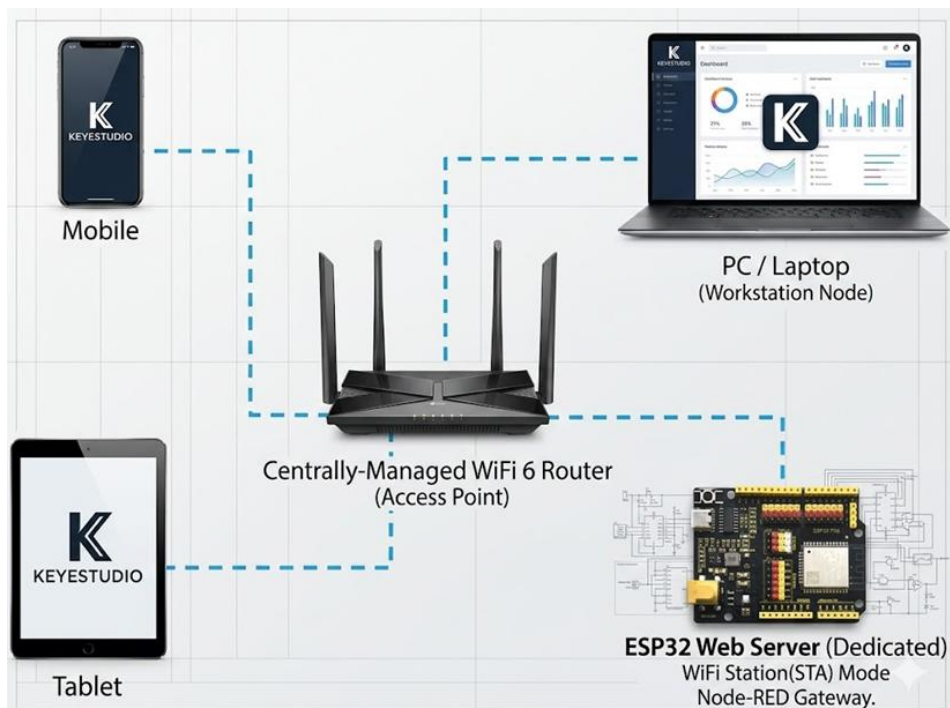


Рис 3.1. Режим станції

У режимі станції ESP32 отримує свою IP-адресу від бездротового роутера, до якого він підключений. З цією IP-адресою він може налаштувати веб-сервер і надавати веб-сторінки всім підключеним пристроям у існуючій WiFi-мережі.

Компоненти.



Рис 3.2. Компоненти для режиму станції

Схема підключення.

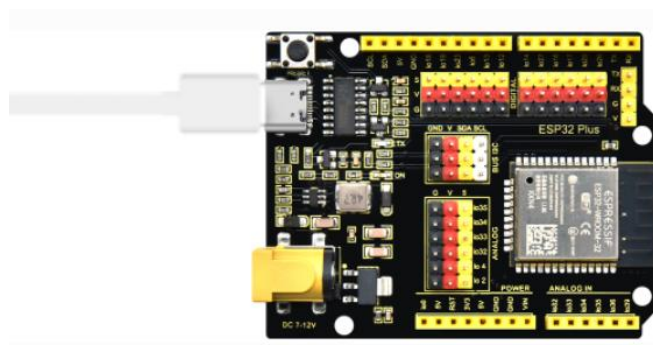


Рис 3.3. Схема підключення

Тестовий код.

Перед завантаженням коду, треба замінити ім'я WiFi (REPLACE_WITH_YOUR_SSID) у коді та паролі (REPLACE_WITH_YOUR_PASSWORD) у своїй.

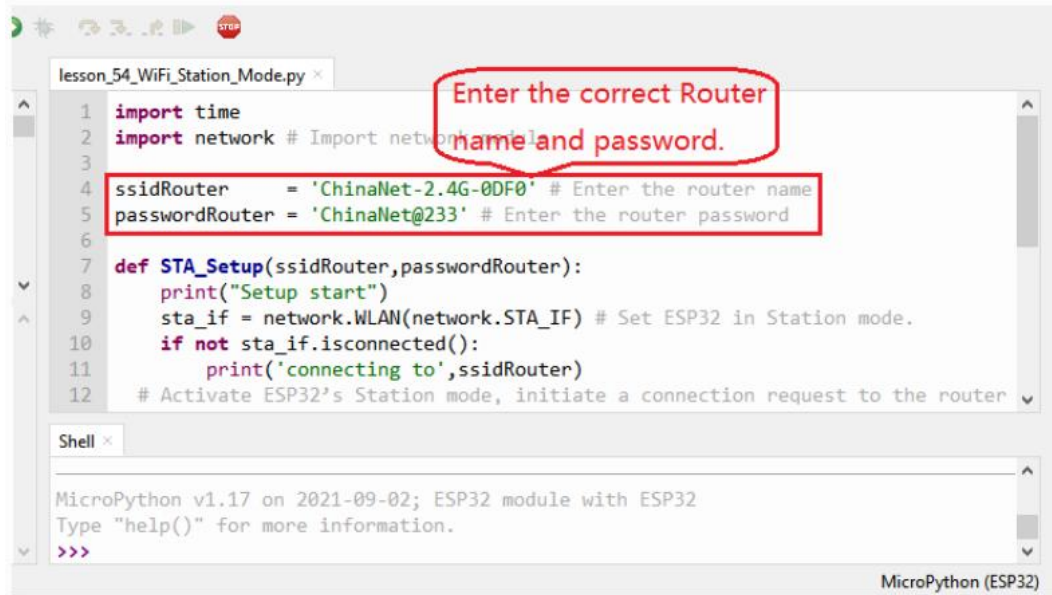


Рис 3.4. Змінення налаштувань

```
...
* Filename      : IoT_ESP32_WiFi_Station_Mode
* Thonny        : Thonny 4.1.7
* Author        : http://www.keyestudio.com
...

import time
import network # Importing network library.

# REPLACE WITH YOUR NETWORK CREDENTIALS(Put your SSID & Password)
ssidRouter = 'REPLACE_WITH_YOUR_SSID' # Enter router name
passwordRouter = 'REPLACE_WITH_YOUR_PASSWORD' # Enter router password

def STA_Setup(ssidRouter,passwordRouter):
    print("Setup start")
    sta_if = network.WLAN(network.STA_IF) # Set the ESP32 to Station mode.
    if not sta_if.isconnected():
        print('connecting to',ssidRouter)
    # Activate the station mode of the ESP32, initiate a connection request to the router and enter the
    sta_if.active(True)
    sta_if.connect(ssidRouter,passwordRouter)
    # Wait for the ESP32 to connect to the router successfully.
    while not sta_if.isconnected():
        pass
    # Print the IP address assigned to the ESP32 MPU in the Shell.
    print('Connected, IP address:', sta_if.ifconfig())
    print("Setup End")

try:
    STA_Setup(ssidRouter,passwordRouter)
except:
    sta_if.disconnect()
```

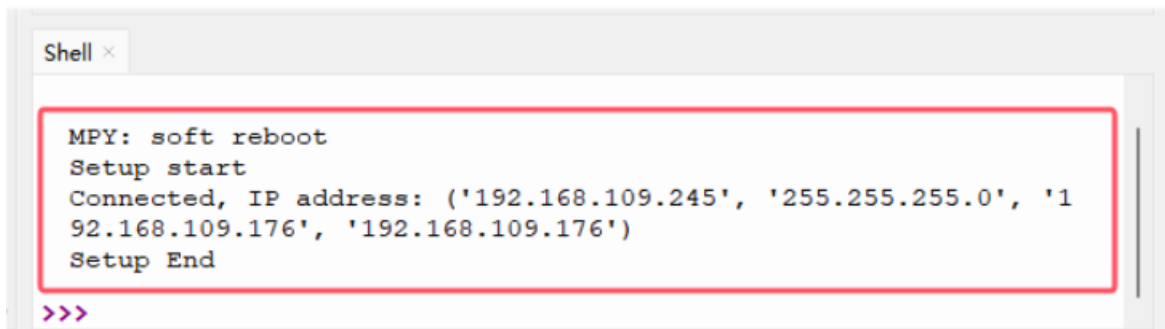
Результат тесту.

1. Відкриваємо "Thonny IDE" і виберіть "This Computer" → "D:" → "Codes" → "MicroPython_Code". Відкрийте файл "2-2-04_IoT_WiFi_Station_Mode.py" або скопіюйте та вставте вищезазначений код у "Thonny IDE".

2. Перед завантаженням коду, будь ласка, замініть ім'я WiFi (REPLACE_WITH_YOUR_SSID) у коді та паролі (REPLACE_WITH_YOUR_PASSWORD) у своїй.

Переконайтеся, що ім'я WiFi та паролі в коді збігаються з мережею, підключеною до вашого комп'ютера, телефону/планшета, плати ESP32 та роутера. Вони мають бути в одній локальній мережі (WiFi).

3. Підключіть плату ESP32 до комп'ютера через USB-кабель і натисніть  для запуску коду. Після завантаження коду та очікування підключення відповідна **IP-адреса** відображається на «Shell».



```
Shell x
MPY: soft reboot
Setup start
Connected, IP address: ('192.168.109.245', '255.255.255.0', '192.168.109.176', '192.168.109.176')
Setup End
>>>
```

Рис 3.5. Результати тесту

Якщо на «Shell» немає IP-адреси, натисніть кнопку RESET на платі ESP32 і натисніть  ще раз.



Рис 3.6. Кнопка RESET на платі ESP32

3.1.2. Організація автономної точки доступу (AP) на базі периферійного контролера

Під час налаштування режиму точки доступу буде створено мережу точок доступу, яка очікує підключення інших пристроїв Wi-Fi. Як показано нижче:

Візьміть ESP32 як точку доступу. Якщо телефону або ПК потрібно зв'язатися з ESP32, його потрібно підключити до точки доступу ESP32. Зв'язок можливий лише після встановлення з'єднання через ESP32.

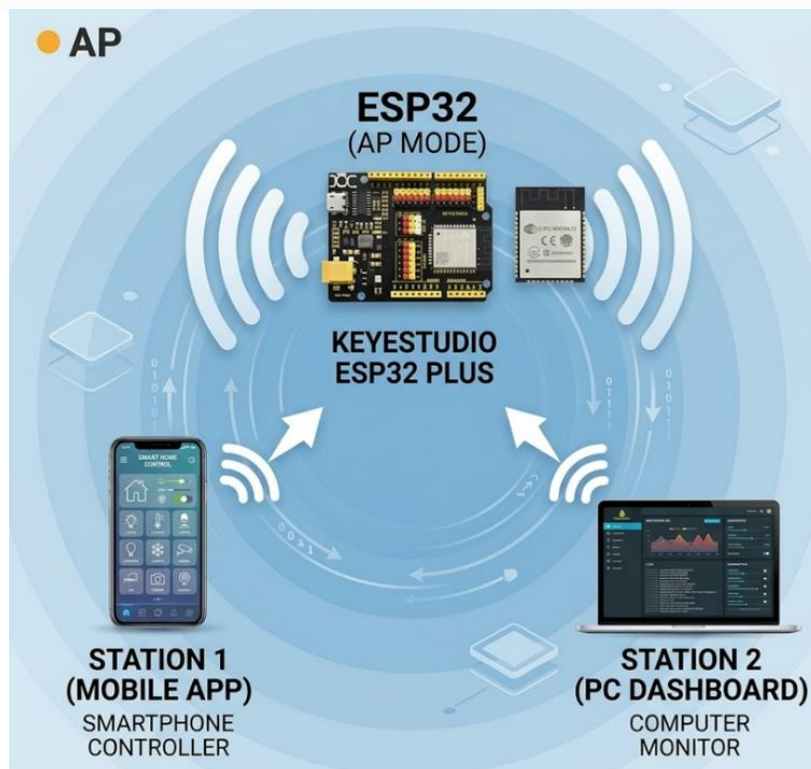


Рис 3.7. Режим точки доступу Wi-Fi

Тестовий код.

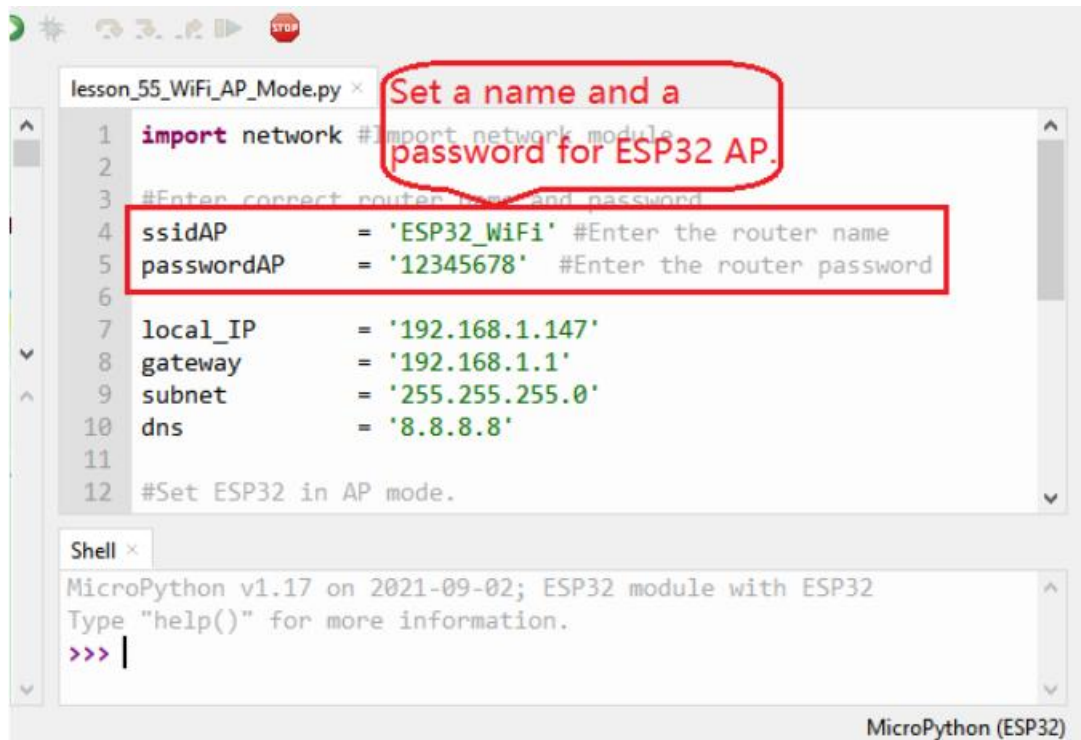


Рис 3.8. Змінення налаштувань

```

import network #Import network module.

#Enter correct router name and password.
ssidAP      = 'ESP32_WiFi' #Enter the router name
passwordAP  = '12345678' #Enter the router password

local_IP    = '192.168.1.147'
gateway     = '192.168.1.1'
subnet      = '255.255.255.0'
dns         = '8.8.8.8'

#Set ESP32 in AP mode.
ap_if = network.WLAN(network.AP_IF)

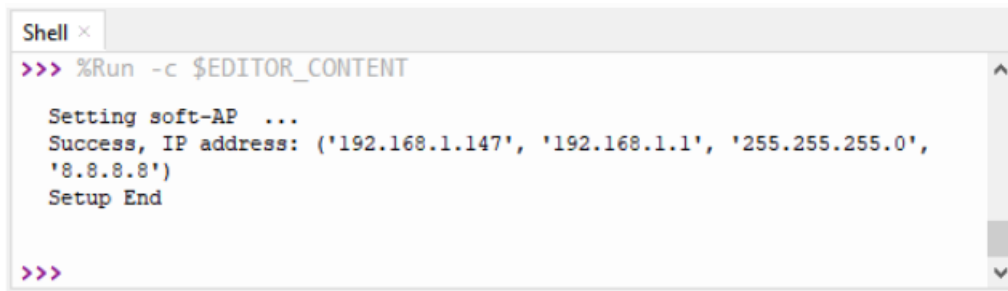
def AP_Setup(ssidAP,passwordAP):
    ap_if.ifconfig([local_IP,gateway,subnet,dns])
    print("Setting soft-AP ... ")
    ap_if.config(essid=ssidAP,authmode=network.AUTH_WPA_WPA2_PSK, password=passwordAP)
    ap_if.active(True)
    print('Success, IP address:', ap_if.ifconfig())
    print("Setup End\n")

try:
    AP_Setup(ssidAP,passwordAP)
except:
    print("Failed, please disconnect the power and restart the operation.")
    ap_if.disconnect()
  
```

Результат тесту.

Ви можете змінити ім'я та пароль точки доступу або залишити їх незмінними. Натисніть кнопку  «Запустити поточний скрипт», код почне

виконуватися. Відкрийте функцію точки доступу ESP32, монітор оболонки виведе інформацію.



```
Shell x
>>> %Run -c $EDITOR_CONTENT

Setting soft-AP ...
Success, IP address: ('192.168.1.147', '192.168.1.1', '255.255.255.0',
'8.8.8.8')
Setup End

>>>
```

Рис 3.9. Результати тесту

Увімкніть функцію пошуку Wi-Fi на вашому телефоні, після чого ви побачите ssid_AP, яка називається «ESP32_Wifi» у цьому коді. Ви можете ввести пароль «12345678» для підключення або змінити назву точки доступу та пароль за допомогою коду.

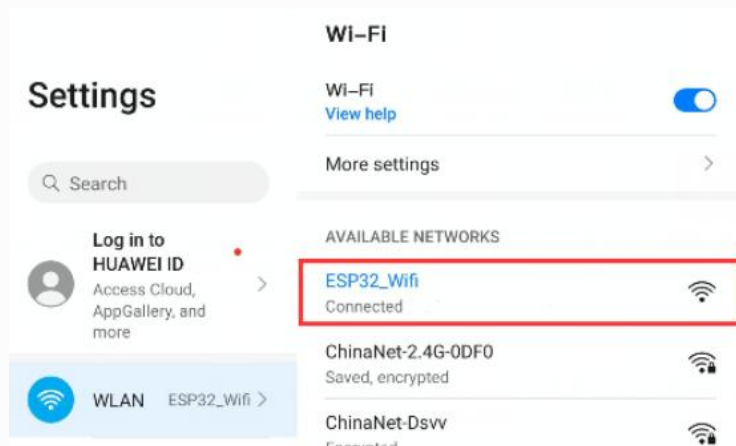


Рис 3.10. Підключення на телефоні

3.1.3. Конфігурування гібридного режиму взаємодії (STA+AP) для IoT-вузлів

Окрім режиму точки доступу та режиму станції, одночасно можна використовувати режим точки доступу + станції. Увімкніть режим станції на ESP32, підключіть його до мережі маршрутизатора, і він зможе взаємодіяти з Інтернетом через маршрутизатор. Потім увімкніть режим точки доступу, щоб створити мережу точки доступу. Інші пристрої Wi-Fi можна підключити до мережі маршрутизатора або мережі точки доступу для зв'язку з ESP32.

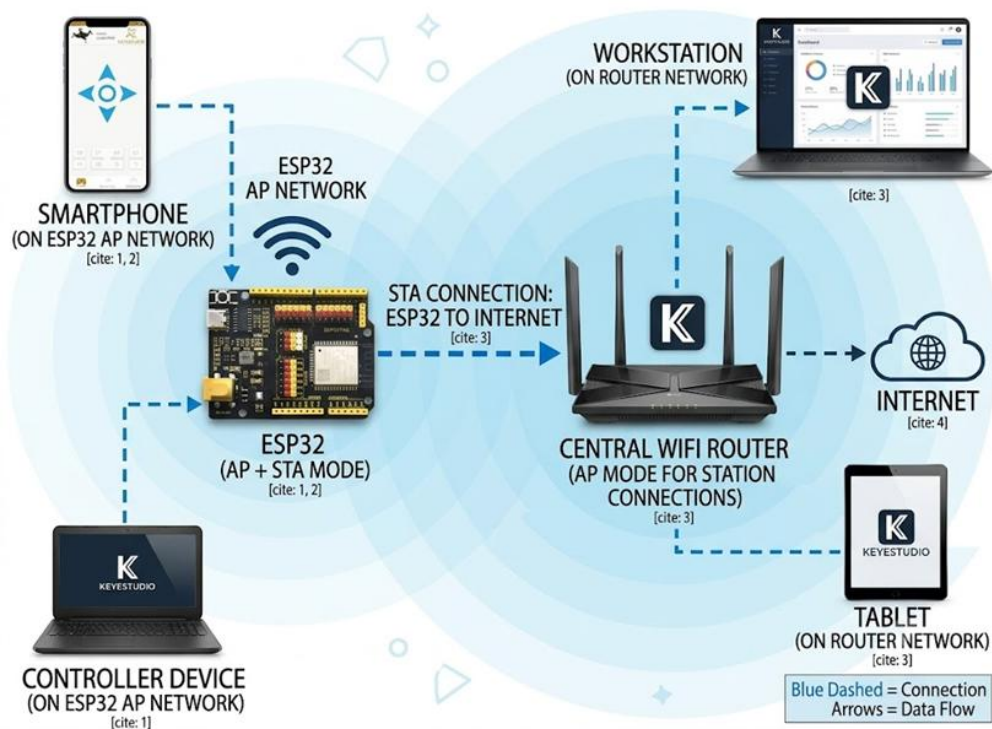


Рис 3.11. Режим точки доступа Wi-Fi + станції

Тестовий код.

```

lesson_56_WiFi_Station+AP_Mode.py
1 import network #Import network module.
2
3 ssidRouter = 'ChinaNet-2.4G-0DF0' #Enter the router name
4 passwordRouter = 'ChinaNet@233' #Enter the router password
5
6 ssidAP = 'ESP32_WiFi' #Enter the AP name
7 passwordAP = '12345678' #Enter the AP password
8
9 local_IP = '192.168.4.147'
10 gateway = '192.168.1.1'
11 subnet = '255.255.255.0'
12 dns = '8.8.8.8'
13
14 sta_if = network.WLAN(network.STA_IF)
15 ap_if = network.WLAN(network.AP_IF)

```

Please enter the correct names and passwords of Router and AP.

Shell

MicroPython v1.17 on 2021-09-02; ESP32 module with ESP32
Type "help()" for more information.
>>>

MicroPython (ESP32)

```

import network #Import network module.

ssidRouter      = 'ChinaNet-2.4G-0DF0' #Enter the router name
passwordRouter  = 'ChinaNet@233' #Enter the router password

ssidAP          = 'ESP32_WiFi' #Enter the AP name
passwordAP      = '12345678' #Enter the AP password

local_IP        = '192.168.4.147'
gateway         = '192.168.1.1'
subnet          = '255.255.255.0'
dns             = '8.8.8.8'

sta_if = network.WLAN(network.STA_IF)
ap_if = network.WLAN(network.AP_IF)

def STA_Setup(ssidRouter,passwordRouter):
    print("Setting soft-STA ... ")
    if not sta_if.isconnected():
        print('connecting to',ssidRouter)
        sta_if.active(True)
        sta_if.connect(ssidRouter,passwordRouter)
        while not sta_if.isconnected():
            pass
        print('Connected, IP address:', sta_if.ifconfig())
        print("Setup End")

def AP_Setup(ssidAP,passwordAP):
    ap_if.ifconfig([local_IP,gateway,subnet,dns])
    print("Setting soft-AP ... ")
    ap_if.config(essid=ssidAP,authmode=network.AUTH_WPA_WPA2_PSK, password=passwordAP)
    ap_if.active(True)
    print('Success, IP address:', ap_if.ifconfig())
    print("Setup End\n")

try:
    AP_Setup(ssidAP,passwordAP)
    STA_Setup(ssidRouter,passwordRouter)
except:
    sta_if.disconnect()
    ap_if.disconnect()

```

Результат тесту.

Перед запуском коду потрібно змінити ssidRouter, passwordRouter, ssidAP та passwordAP. Переконавшись, що код змінено правильно, натисніть кнопку  «Запустити поточний скрипт», і у вікні «Оболонка» з'явиться наступне:

```
Shell x
MicroPython v1.17 on 2021-09-02; ESP32 module with ESP32
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT

Setting soft-AP ...
Success, IP address: ('192.168.4.147', '192.168.1.1', '255.255.255.0',
'8.8.8.8')
Setup End

Setting soft-STA ...
connecting to ChinaNet-2.4G-0DF0
Connected, IP address: ('192.168.1.147', '255.255.255.0', '192.168.1.1'
, '114.114.114.114')
Setup End

>>>
```

Рис 3.12. Результат тесту

Тоді ви можете побачити ssid_A на ESP32.

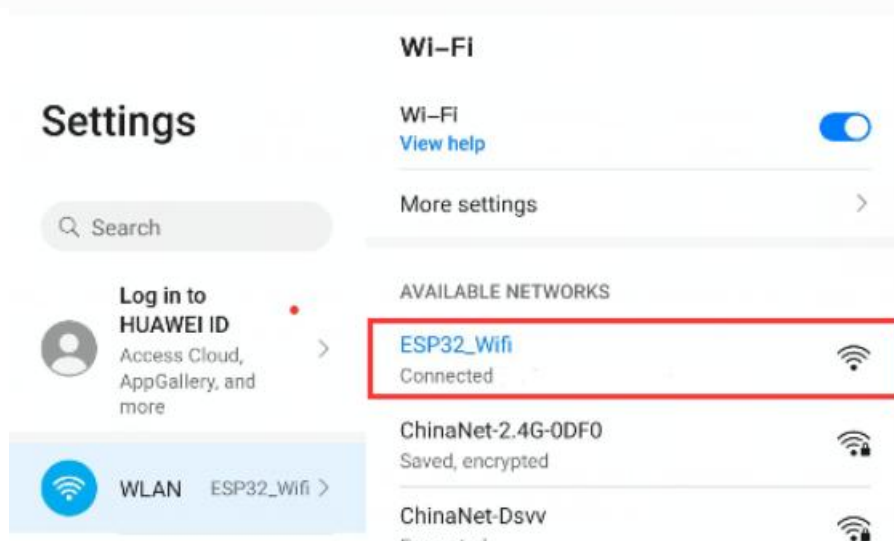


Рис 3.13. Підключення на телефоні

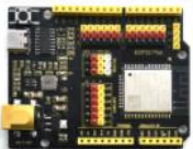
3.2. Програмно-апаратна реалізація підсистеми санкціонованого доступу на основі RFID-технологій

У сучасних будинках автоматизація починається з вхідних дверей. Звичайна автономна кодова панель чи замок працюють ізольовано. Проєкт **"Smart Access Control"** демонструє інтеграцію периферійних пристроїв у загальну інфраструктуру Розумного будинку.

У цьому проєкті зчитувач RFID RC522 виступає як елемент підсистеми сенсорів та збору даних, сервопривід — як елемент підсистеми виконавчих механізмів (актуаторів), а мікроконтролер об'єднує їх та забезпечує зв'язок із користувачем через бездротову мережу (наприклад, у режимі локального вебсервера чи через Wi-Fi маршрутизатор). Тут ми застосовуємо UID-код для

керування сервоприводом, щоб він повертався на певний кут, імітуючи відкриття дверей.

Компоненти.

		
Головна плата ESP32 x1	RFID-модуль x1	Дроти F-F DuPont
		
IC-карта/брелок x1	Кабель Micro USB x1	Servo x1

Принципова схема.

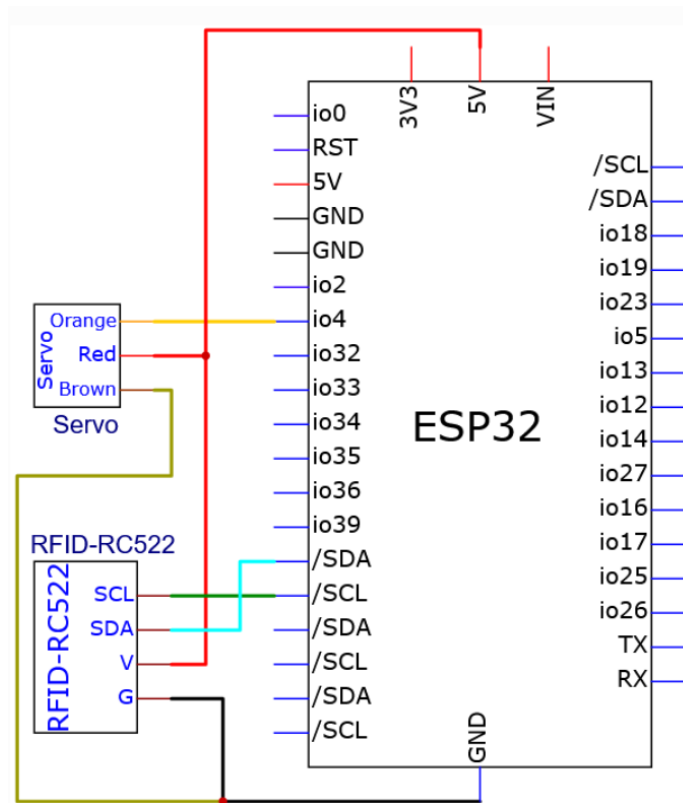


Рис 3.14. Принципова схема

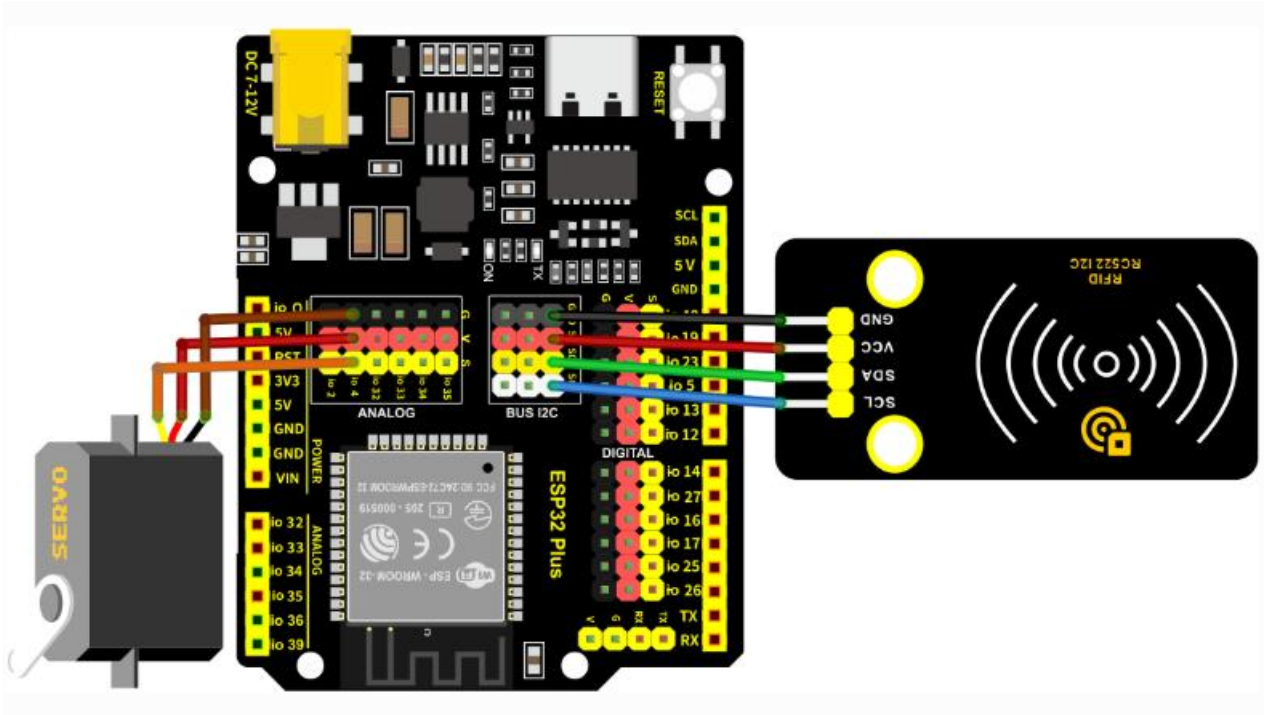


Рис 3.15. Схема підключення

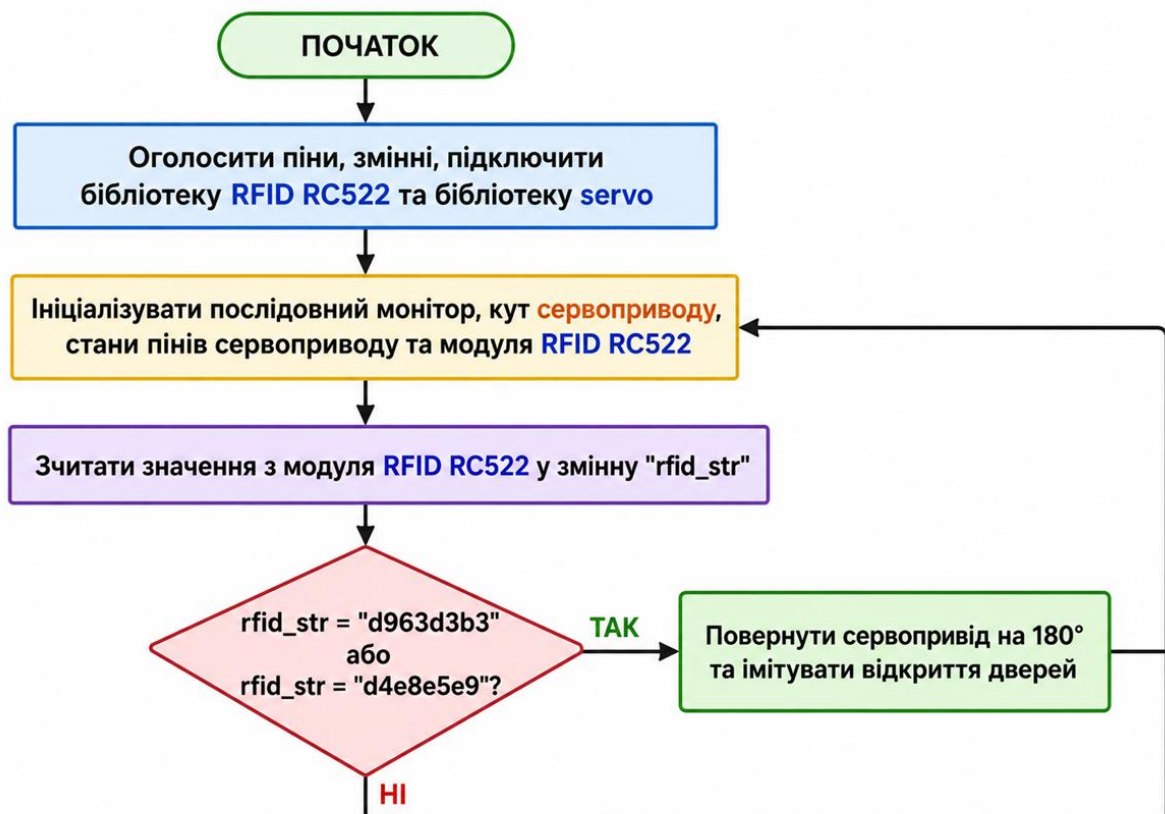


Рис 3.16. Алгоритм роботи

Тестовий код.

```

from machine import Pin, PWM
import time
from mfrc522_i2c import mfrc522

# define the output frequency of I04 as 50Hz and assign it to PWM.
servoPin = PWM(Pin(4))
servoPin.freq(50)
servoPin.duty(256)

# i2c config.
addr = 0x28
scl = 22
sda = 21

# initialize the pins and addresses of mfrc522
rc522 = mfrc522(scl, sda, addr)
rc522.PCD_Init()
rc522.ShowReaderDetails() # display details of the PCD-MFRC522 card reader

# UID code values of the white card and green key chain
uid1 = [180, 180, 170, 219]
uid2 = [231, 185, 101, 101]

# initial Angle of the servo
servoPin.duty_u16(1638)
time.sleep(1)

# Read the UID code values for the white card and the green key chain. If the value is correct, the serv
while True:
    if rc522.PICC_IsNewCardPresent():
        #print("Is new card present!")
        if rc522.PICC_ReadCardSerial() == True:
            print("Card UID:", end=' ')
            print(rc522.uid.uidByte[0 : rc522.uid.size])
            if rc522.uid.uidByte[0 : rc522.uid.size] == uid1 or rc522.uid.uidByte[0 : rc522.uid.size] ==
                servoPin.duty_u16(8100)
                time.sleep(1)
            else :
                servoPin.duty_u16(1638)
                time.sleep(1)

```

Результат тесту.

Підключаємо основну плату ESP32 до комп'ютера за допомогою кабелю Micro USB. Натиснемо , щоб запустити код. Якщо код не вдається запустити, у розділі «Shell» з'явиться повідомлення про помилку.

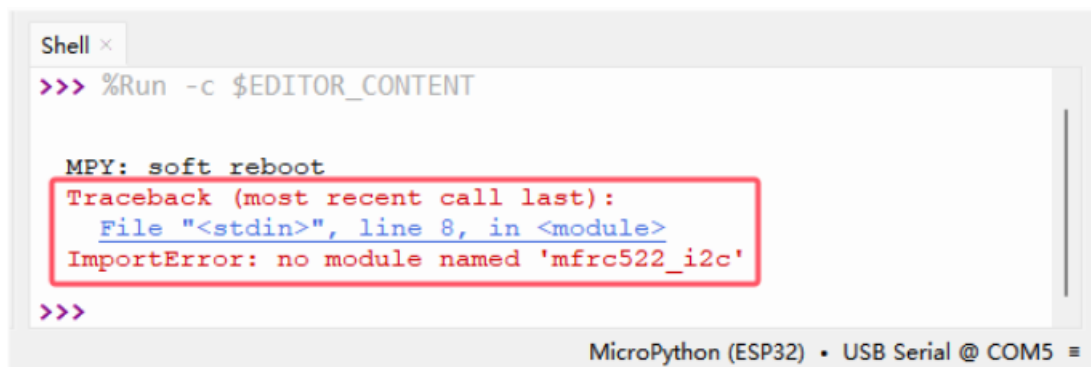
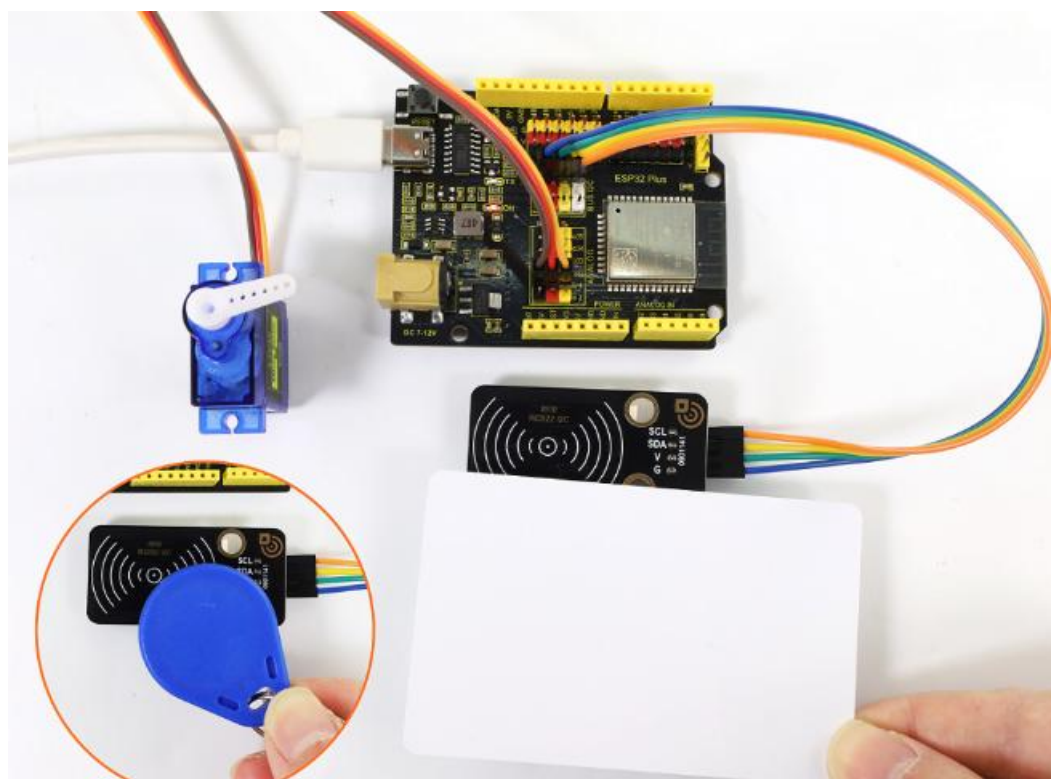


Рис 3.17. Повідомлення про помилку

Натиснемо , щоб запустити код. При використанні правильної ІС-карти або брелока сервопривід обертається (щоб відкрити двері). В іншому випадку сервопривід не обертатиметься. Водночас, «Shell» відображає дані, зчитані RFID-модулем.



```
Shell x
>>> %Run -c $EDITOR_CONTENT

MPY: soft reboot
MFRC522 Software Version:146 = v2.0
Card UID:
[180, 180, 170, 219]
Card UID:
[180, 180, 170, 219]
Card UID:
[180, 180, 170, 219]
Card UID:
[231, 185, 101, 101]
Card UID:
[231, 185, 101, 101]
Card UID:
[231, 185, 101, 101]
```

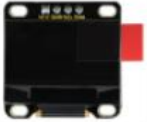
Рис 3.18. Результат роботи тесту

3.3. Проектування та тестування автоматизованої підсистеми мікроклімату та терморегулювання

У цьому експерименті ми використовуватимемо датчик температури та деякі модулі для створення інтелектуального охолоджувального пристрою.

Коли датчик температури виявляє, що температура навколишнього середовища перевищує певне значення, двигун 130 вмикається для охолодження, а потім значення температури відображається на OLED-дисплеї.

Компоненти

		
Материнська плата ESP32 x1	OLED x1	Дроти MF DuPont
		
Кабель Micro USB x1	130 двигун постійного струму x1	вентилятор x1
		
Дроти FF DuPont	макетна плата x1	дроти для перемикання !

Принципова схема.

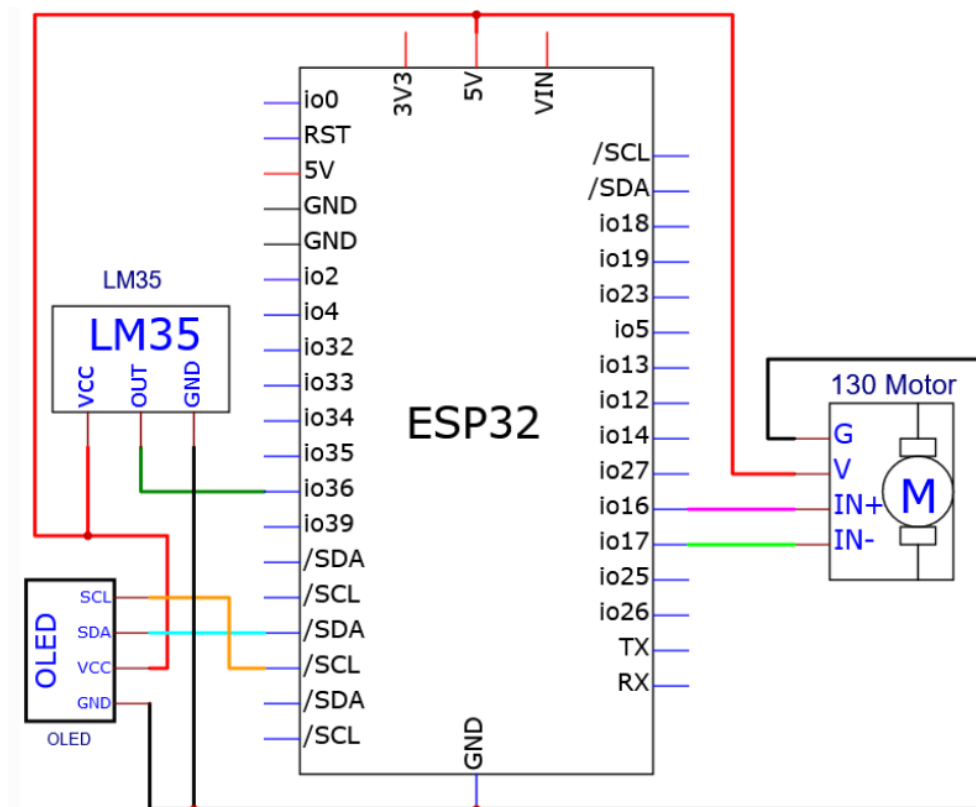


Рис 3.19. Принципова схема

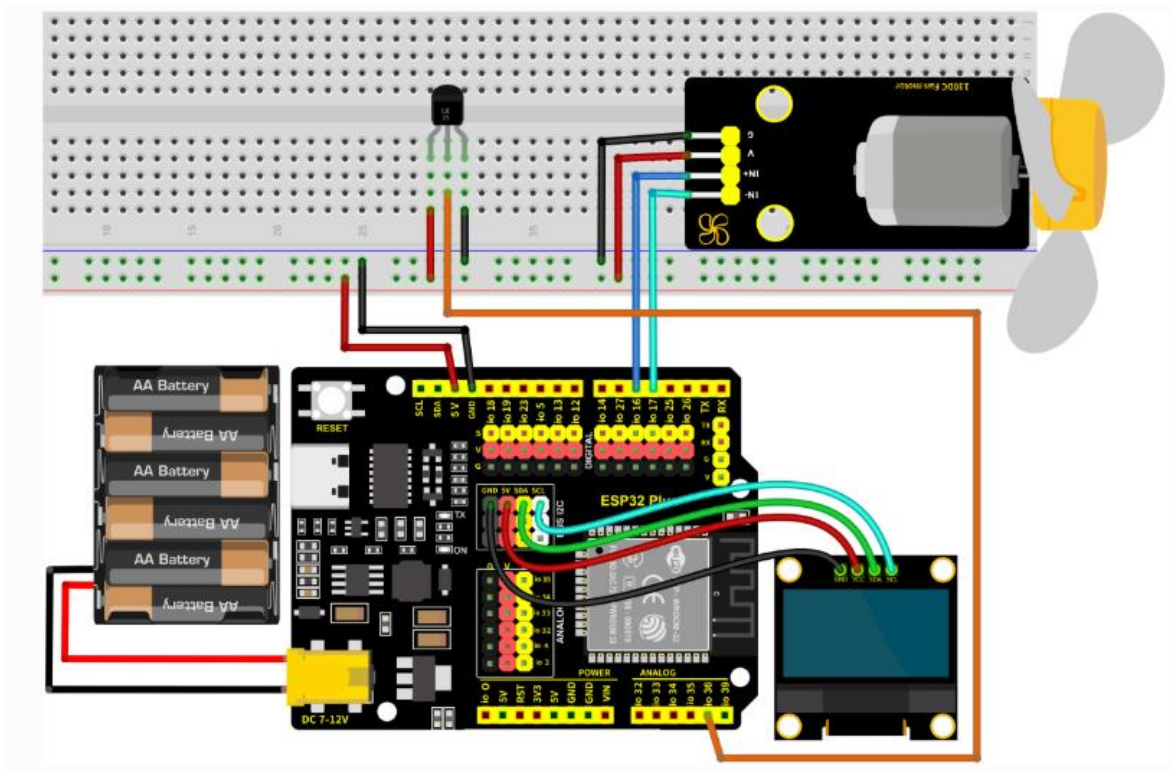


Рис 3.20. Схема підключення

Ми встановлюємо температурний поріг (його можна змінювати відповідно до фактичної ситуації). Двигун обертається, коли виявлене значення перевищує поріг, і OLED-дисплей відображає значення температури. Коли датчик температури виявляє, що температура навколишнього середовища вища за певне значення, двигун вмикається для охолодження, а потім значення температури відображається на OLED-дисплеї.

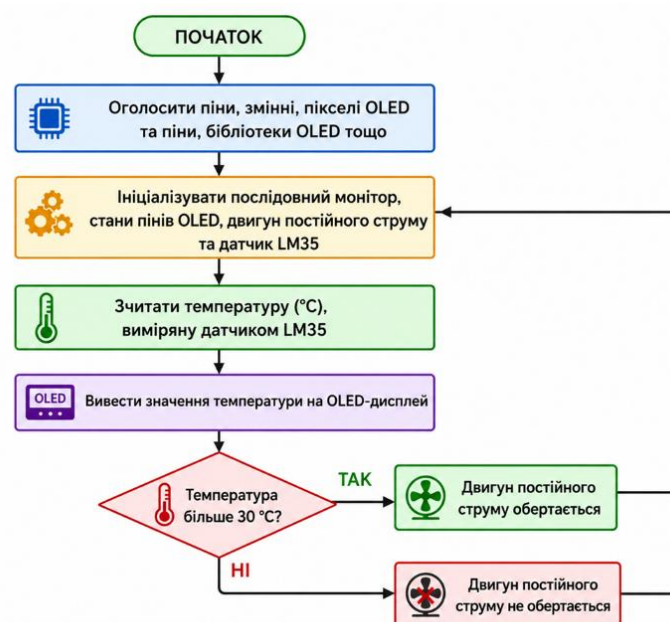


Рис 3.21. Алгоритм роботи

Тестовий код.

```
from machine import Pin, ADC, PWM, SoftI2C
import machine
import time
import ssd1306
from time import sleep

# ESP32 Pin assignment
i2c = SoftI2C(scl=Pin(22), sda=Pin(21))

# Set the width, height and i2c of the OLED
oled_width = 128
oled_height = 64
oled = ssd1306.SSD1306_I2C(oled_width, oled_height, i2c)

# Enable and configure ADC with a range of 0-3.3V
adc=ADC(Pin(36))
adc.atten(ADC.ATTN_11DB)
adc.width(ADC.WIDTH_12BIT)
conversion_factor = 5.0 / (4095)

# motor pins
INA = Pin(16, Pin.OUT) # INA to IN+
INB = Pin(17, Pin.OUT) # INB to IN-

# OLED displays the temperature detected by the LM35. When it is higher than 30°C, turn on the motor.
while True:
    adcVal=adc.read()
    reading = adcVal * conversion_factor
    temperature = reading * 102.4
    temperature_string = str(temperature)
    oled.text('Cooling Device', 0, 0)
    oled.text('Temper(C):', 0, 20)
    oled.text(temperature_string, 0, 40)
    oled.show()
    time.sleep(0.2)
    oled.fill(0)
    if temperature > 30: # When the temperature is higher than 30°C
        # turn on the motor
        INA.value(0)
        INB.value(1)
    else: # When the temperature is not higher than 30°C
        # turn off the motor
        INA.value(0)
        INB.value(0)
```

Результат тесту.

Підключаємо основну плату ESP32 до комп'ютера за допомогою кабелю Micro USB. Тиснемо , щоб запустити код.

OLED-дисплей відображає значення температури навколишнього середовища, виявлене датчиком LM35. Коли воно перевищує 30°C, двигун 130 вмикається для охолодження.

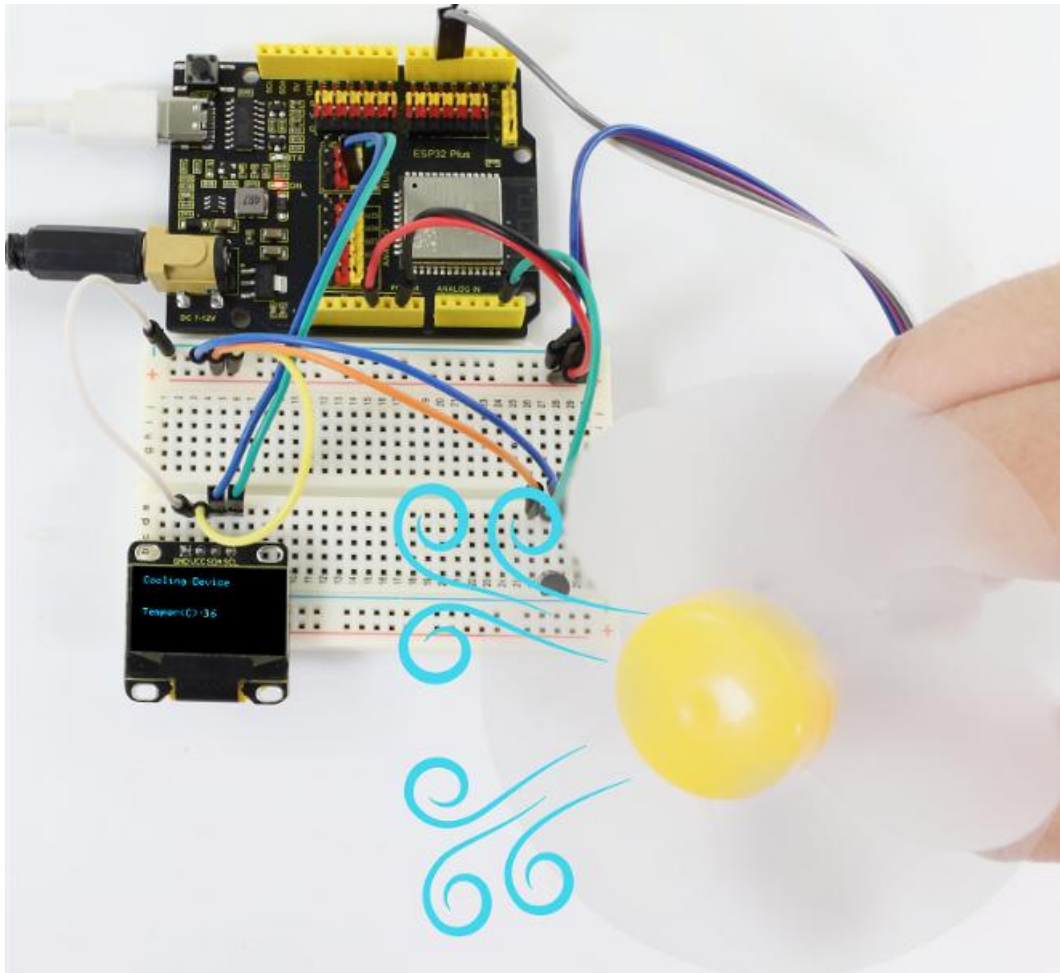


Рис 3.22. Результат роботи тесту

3.4. Розробка та експериментальне дослідження підсистеми газового аналізу та аварійного оповіщення

У цьому експерименті ми створимо димову сигналізацію за допомогою сегментного модуля TM16504-Digit, газового датчика та зумера.

Коли газовий датчик виявляє високу концентрацію небезпечного газу, Дзвінок подасть сигнал тривоги, і на екрані з'явиться небезпека (відобразить значення концентрації).

Схема підключення.

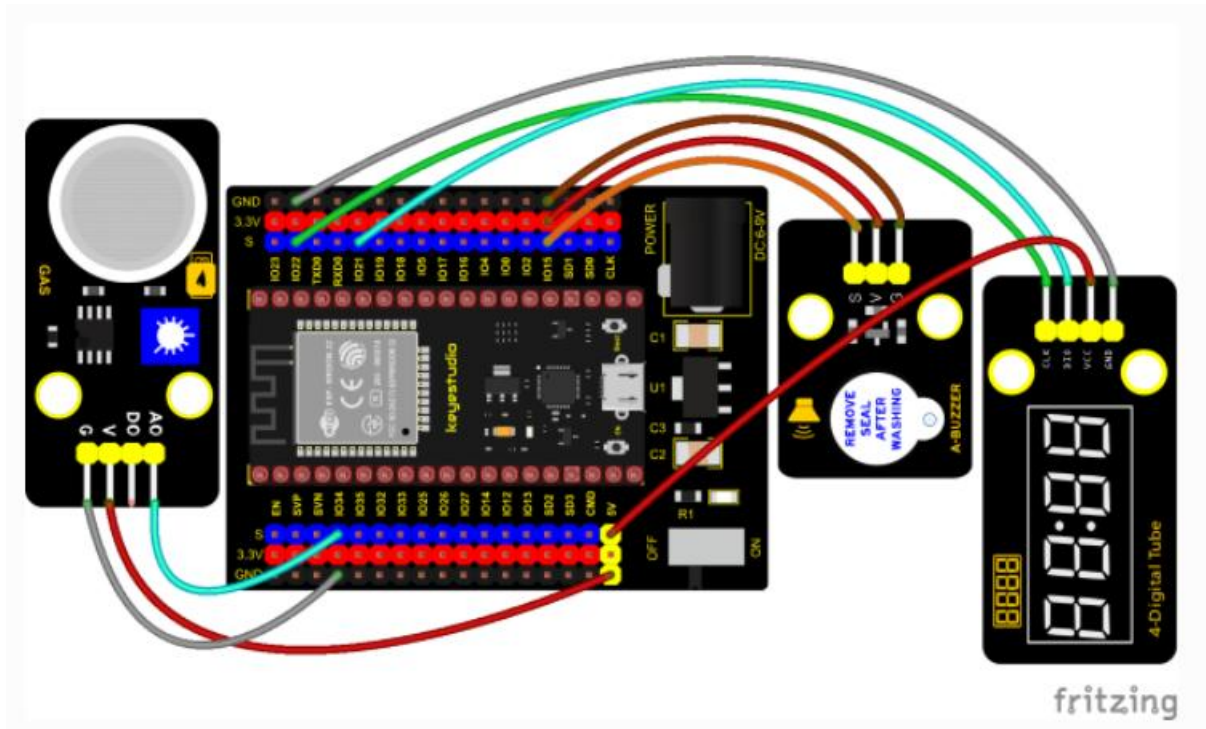


Рис 3.23. Схема підключення

Тестовий код.

```
from time import sleep_ms, ticks_ms
from machine import SoftI2C, Pin
from i2c_lcd import I2cLcd

DEFAULT_I2C_ADDR = 0x27

scl_pin = Pin(22, Pin.OUT, pull=Pin.PULL_UP) # GPIO22 with internal pull-up enabled
sda_pin = Pin(21, Pin.OUT, pull=Pin.PULL_UP) # GPIO21 with internal pull-up enabled

i2c = SoftI2C(scl=Pin(22), sda=Pin(21), freq=100000)
lcd = I2cLcd(i2c, DEFAULT_I2C_ADDR, 2, 16)

from machine import Pin
import time
gas = Pin(23, Pin.IN, Pin.PULL_UP)

while True:
    gasVal = gas.value() # Reads the value of button 1
    print("gas =", gasVal) # Print it out in the shell
    lcd.move_to(1, 1)
    lcd.putstr('val: {}'.format(gasVal))
    if(gasVal == 1):
        #lcd.clear()
        lcd.move_to(1, 0)
        lcd.putstr('Safety      ')
    else:
        lcd.move_to(1, 0)
        lcd.putstr('dangerous')
    time.sleep(0.1) #delay 0.1s
```

Результати тесту.

Екран відображає «безпеку» у нормальному стані. Однак, коли газ Сенсор виявляє деякі небезпечні гази, такі як чадний газ, на рівні При певній концентрації дзвінок подасть сигнал тривоги і екран показує «небезпечний».

3.5. Розгортання локального веб-сервера для комплексного дистанційного керування компонентами «Розумного будинку»

Тут ми керуємо кількома модулями на веб-сторінках через ESP32 WiFi.

Компоненти.

		
Материнська плата ESP32 x1	Транзистор S8050 x1	сервопривід x1
		
Двигун постійного струму x1	макетна плата x1	перемикальні дроти
		
вентилятор x1	тримач батареї x1	Батарейки типу AA (надаються окремо) x1
		
мобільний пристрій x1	1 RGB-світлодіод	реле x1
		
активний зумер x1	Резистор 220 Ом x4	Дроти FF DuPont

Схема підключення.

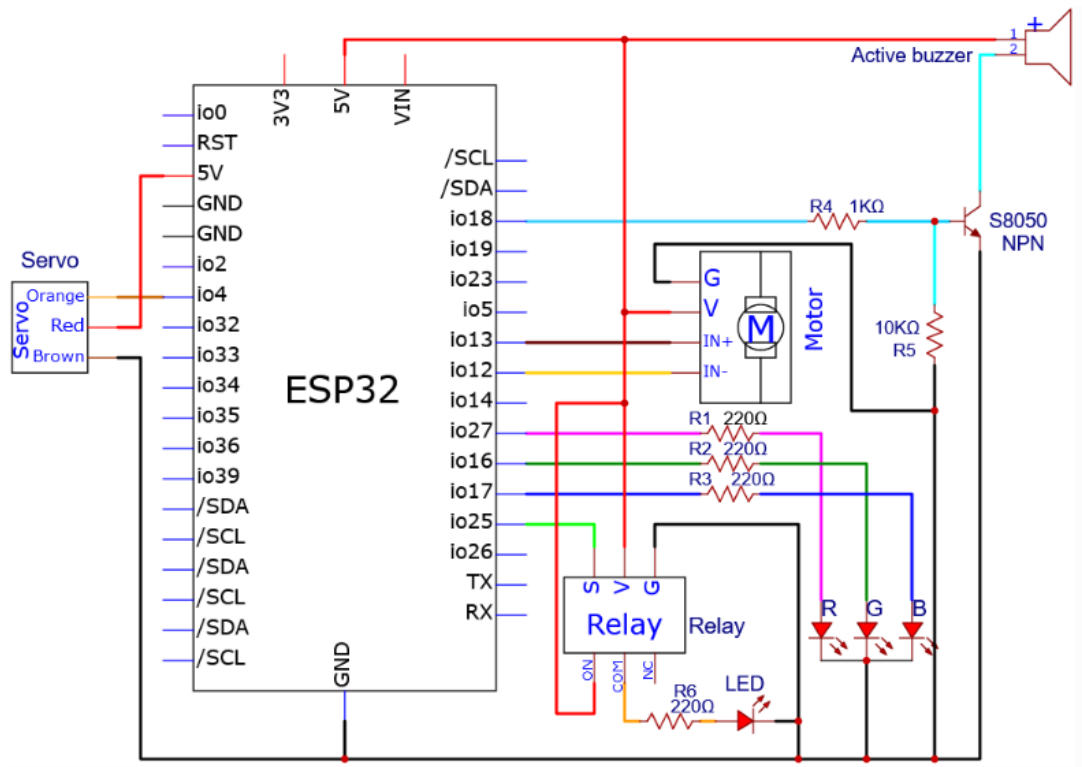


Рис 3.24. Принципова схема

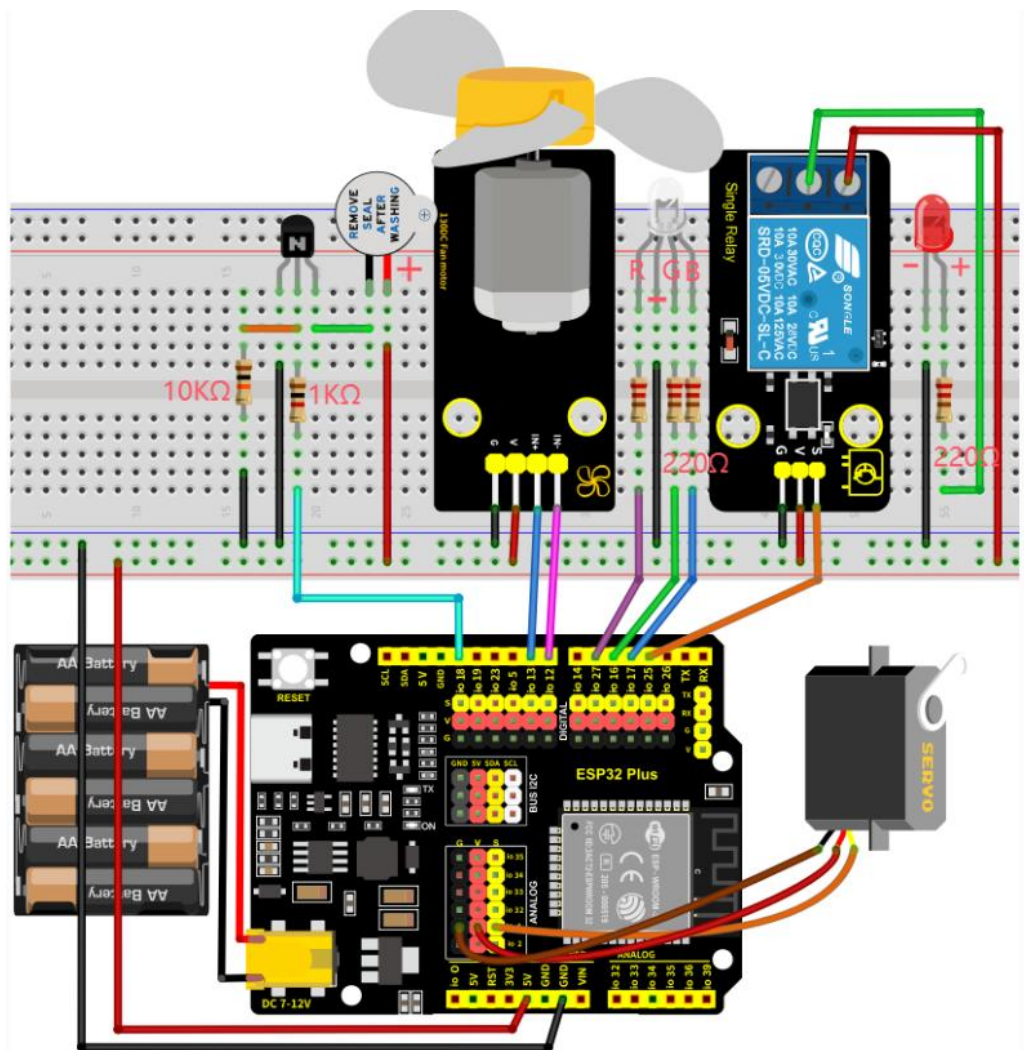


Рис 3.25. Схема підключення

Тестовий код.

Перед завантаженням коду, будь ласка, замініть назву WiFi (REPLACE_WITH_YOUR_SSID) у коді та паролі (REPLACE_WITH_YOUR_PASSWORD) на ваші.

```
12 # REPLACE WITH YOUR NETWORK CREDENTIALS(Put your SSID & Password)
13 ssid = 'REPLACE_WITH_YOUR_SSID' # Enter router name
14 password = 'REPLACE_WITH_YOUR_PASSWORD' # Enter router password
15
```

Рис 3.26. Зміна налаштувань WiFi

Наведемо частину коду. Повна частина в додатку Б.

```
import machine
import network
import socket
import time
from machine import Pin, PWM

# REPLACE WITH YOUR NETWORK CREDENTIALS(Put your SSID & Password)
ssid = 'REPLACE_WITH_YOUR_SSID' # Enter router name
password = 'REPLACE_WITH_YOUR_PASSWORD' # Enter router password

buzzer = machine.Pin(18, machine.Pin.OUT) # Set the GPIO pin to output (assuming the active buzzer is c
motora = machine.Pin(13, machine.Pin.OUT) # Set GPIO pin to output (assuming DC motor IN+ is connected
motorb = machine.Pin(12, machine.Pin.OUT) # Set GPIO pin to output (assuming DC motor IN-connected IN G
relay = machine.Pin(25, machine.Pin.OUT) # Set the GPIO pin to output (assuming the relay is connected

# Define the GPIO pins for the RGB LED
RED_PIN = 27
GREEN_PIN = 16
BLUE_PIN = 17
# Set up the PWM channels
red = PWM(Pin(RED_PIN))
green = PWM(Pin(GREEN_PIN))
blue = PWM(Pin(BLUE_PIN))

# Set the PWM frequency
red.freq(1000)
green.freq(1000)
blue.freq(1000)

# Map input values from one range to another
def interval_mapping(x, in_min, in_max, out_min, out_max):
    return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min

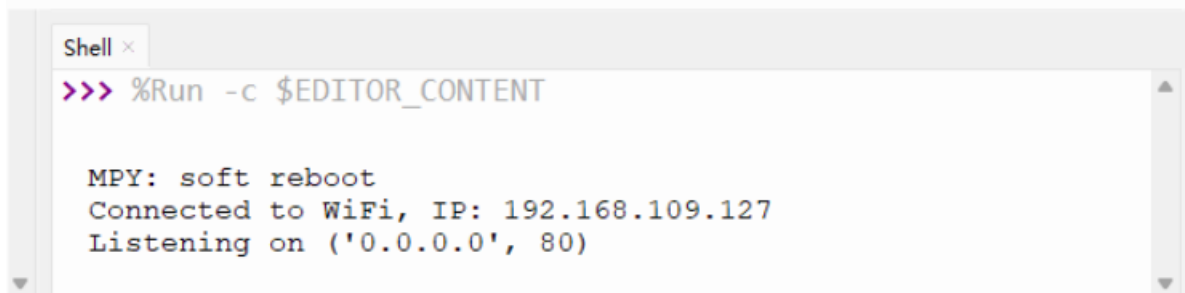
# Convert color values (0-255) to duty cycle values (0-1023)
def color_to_duty(rgb_value):
    rgb_value = int(interval_mapping(rgb_value, 0, 255, 0, 1023))
    return rgb_value

def set_color(red_value, green_value, blue_value):
    red.duty(color_to_duty(red_value))
    green.duty(color_to_duty(green_value))
    blue.duty(color_to_duty(blue_value))
```

Результат тесту.

Перед завантаженням коду, будь ласка, замініть назву WiFi (REPLACE_WITH_YOUR_SSID) у коді та паролі (REPLACE_WITH_YOUR_PASSWORD) на ваші.

Підключіть плату ESP32 до комп'ютера за допомогою USB-кабелю та натисніть кнопку  запуску коду. Після завантаження коду та очікування з'єднання відповідна IP-адреса відобразиться в «Shell».

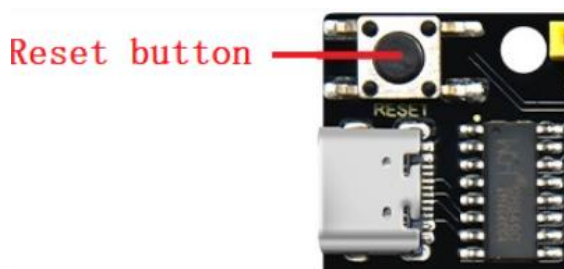


```
Shell x
>>> %Run -c $EDITOR_CONTENT

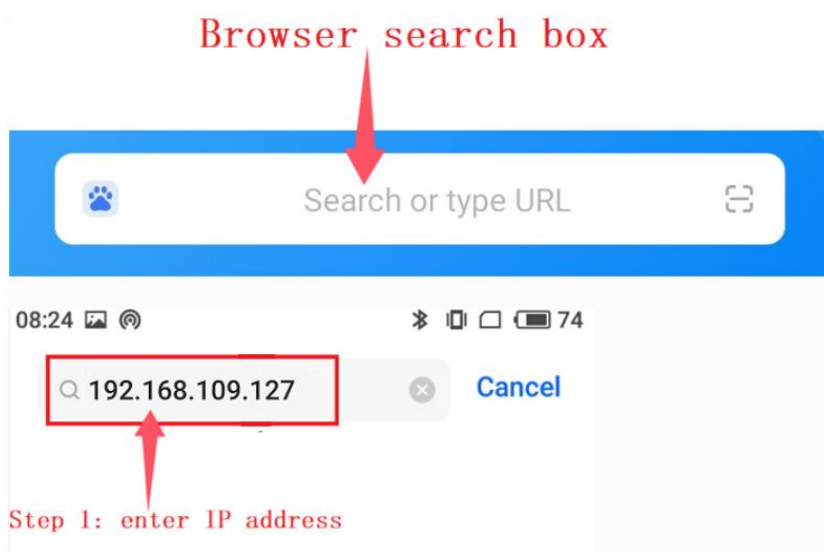
MPY: soft reboot
Connected to WiFi, IP: 192.168.109.127
Listening on ('0.0.0.0', 80)
```

Рис 3.27. Відображення IP-адреси в «Shell»

Якщо в «Shell» немає IP-адреси, натисніть кнопку RESET на платі ESP32 і клацніть  ще раз.



Відкрийте браузер на телефоні, введіть IP-адресу Wi-Fi та натисніть «Пошук».



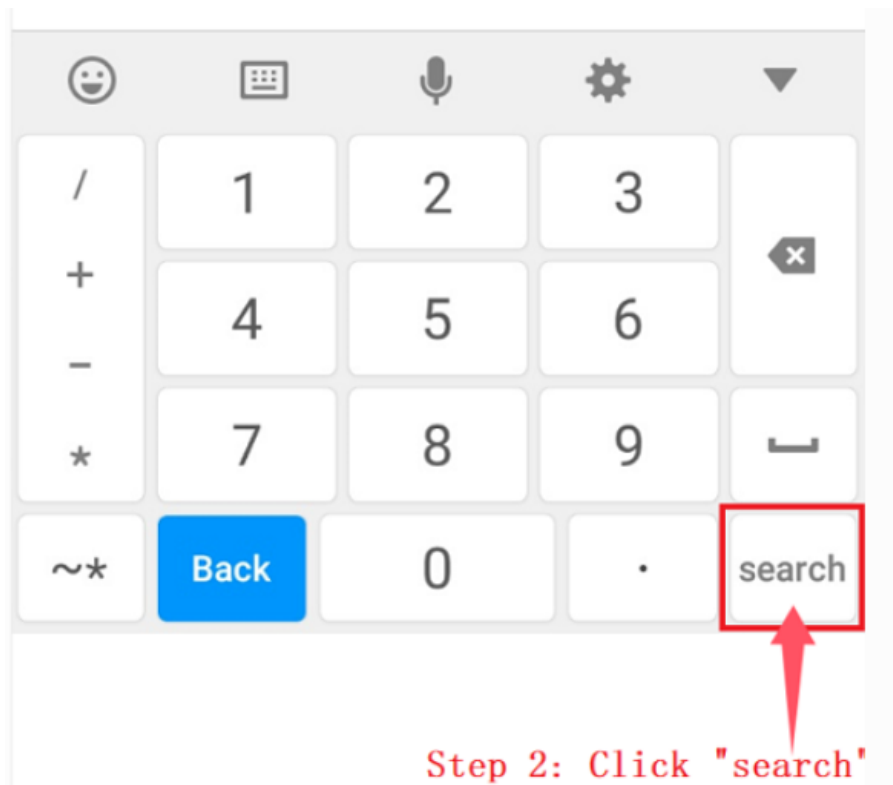


Рис 3.28. Налаштування телефону

Через кілька секунд відкриється веб-сторінка WiFi, що свідчить про успішне підключення модуля ESP32 до WiFi. На веб-сторінці WiFi відображається наступний вміст.



Рис 3.29. Зміст веб-сторінки WiFi



Рис 3.30. Результат роботи системи

Висновки до розділу

У третьому розділі виконано повний цикл програмно-апаратної реалізації, конфігурування та експериментального тестування ключових компонентів інтелектуального житла на базі платформи KEYESTUDIO ESP32 PLUS мовою MicroPython у середовищі Thonny IDE.

На основі отриманих результатів сформульовано такі основні висновки:

1. В ході дослідження та налаштування режимів бездротового зв'язку Wi-Fi на базі мікроконтролера ESP32 практично підтверджено гнучкість архітектури пристрою під час роботи у режимах клієнта (STA), автономної точки доступу (AP) та їхнього гібридного поєднання (STA+AP). Це дозволило забезпечити надійну основу для мережевої інтеграції та безперебійного обміну даними всередині розроблюваної IoT-інфраструктури.
2. Завдяки програмно-апаратній реалізації підсистеми санкціонованого доступу на основі RFID-технологій створено діючий вузол безпеки, що поєднує зчитувач RC522 та сервопривід виконавчого механізму. Написаний алгоритм

продемонстрував чітку ідентифікацію унікальних UID-кодів ідентифікаторів та стабільне керування кутом повороту приводу для імітації відмикання дверей.

3. На етапі проєктування та тестування автоматизованої підсистеми мікроклімату та терморегулювання було успішно об'єднано аналоговий термодатчик LM35, OLED-екран та двигун постійного струму. Експерименти довели високу точність реагування системи на динаміку температурних змін навколишнього середовища та безвідмовне ввімкнення охолоджувального вентилятора у разі перевищення межі у 30°C.

4. У межах розробки та експериментального дослідження підсистеми газового аналізу та аварійного оповіщення реалізовано надійний захисний контур будинку на базі газового сенсора, індикатора TM1650 та активного зумера. Тестування підтвердило здатність підсистеми миттєво ідентифікувати небезпечні концентрації газів із паралельним виведенням повідомлення «Danger» на дисплей та активацією звукової тривоги.

5. Розгортання локального веб-сервера для комплексного дистанційного керування компонентами «Розумного будинку» дозволило консолідувати всі периферійні виконавчі пристрої (реле, RGB-модуль, вентилятор, зумер, сервопривід) в межах єдиної системи. Створений веб-інтерфейс продемонстрував високу швидкість відгуку та кросплатформну стабільність під час керування будинком через браузер мобільного пристрою за локальною IP-адресою.

Практичні результати тестування всіх спроектованих підсистем засвідчили повну працездатність, високу масштабованість та технічну зрілість розроблених рішень автоматизації.

ВИСНОВКИ

У кваліфікаційній роботі вирішено актуальне науково-практичне та методичне завдання з проєктування, модульно-схемотехнічної інтеграції, програмного кодування мовою MicroPython та експериментального тестування діючого навчально-демонстраційного лабораторного стенду інтернет-речей (IoT) «Розумний будинок». Створений комплекс дозволяє ефективно симулювати, моніторити та оптимізувати процеси збору телеметричних даних і бездротового керування кліматичними, безпековими та комунікаційними сценаріями сучасного інтелектуального житла.

За результатами теоретичних досліджень, етапів інженерного проєктування та серії практичних експериментів сформульовано висновки:

1. Проведено системно-порівняльний аналіз теоретичних засад розвитку концепції домашньої автоматизації від перших закритих proprietary-протоколів (зокрема технологій Leviton та X10, що функціонували у часозалежних мережах США) до сучасних відкритих децентралізованих архітектур. Досліджено комерційні екосистеми (Ajax, BroadLink, Fibaro), визначено їхні архітектурні обмеження (високу капіталомісткість, апаратну закритість, складність масштабування сторонніми розробниками) та науково обґрунтовано доцільність розробки модульного відкритого лабораторного стенду для задоволення потреб освітнього процесу та rapid prototyping.
2. Обґрунтовано вибір апаратної платформи обчислювального ядра. Доведено, що плата розширення KEYESTUDIO ESP32 PLUS на базі системи на кристалі (SoC) ESP32-WROOM-32 є оптимальним базисом для побудови IoT-платформ. Наявність двоядерного 32-бітного процесора Xtensa LX6 із тактовою частотою 240 МГц та 520 КБ SRAM забезпечує повноцінну паралельну обробку потоків (multithreading), що дозволяє одночасно здійснювати низькорівневе зчитування периферії, виконувати локальні алгоритми автоматизації

та підтримувати мережеві стеки зв'язку без затримок у реальному часі.

3. Спроектовано сенсорну підсистему збору телеметрії. У межах розробки інтегровано гетерогенний контур первинних перетворювачів із різними інтерфейсами передачі інформації:

- цифровий датчик DHT11 (протокол Single-Bus) для прецизійного покімнатного моніторингу температури та вологості повітря;
- напівпровідниковий газорезистивний сенсор MQ-2 (аналогово-цифровий інтерфейс) для ідентифікації небезпечних ppm-концентрацій диму й чадного газу;
- аналоговий датчик водяної пари (Steam Sensor) для детектування аварійного розгерметизування інженерних мереж;
- пасивний інфрачервоний датчик руху PIR (дискретний TTL-рівень з динамічним тригеруванням) для контролю простору;
- модуль безконтактної радіочастотної автентифікації RFID RC522 (інтерфейс I2C) для підсистеми санкціонованого доступу. Усі сенсори пройшли етап калібрування, а їхні сигнальні лінії були узгоджені за амплітудою з вхідними каскадами АЦП та GPIO мікроконтролера.

4. Розроблено та інтегровано комплекс виконавчих механізмів (актуаторів), що імітують фізичну відповідь будівлі на керівні впливи контролера. Локальний вивід інформації покладено на рідкокристалічний дисплей LCD 1602 із I2C-адаптером (адреса 0x27), що вивільнило значний ресурс GPIO-портів. Роботу запірних конструкцій симулює мікросервопривід SG90 через регулювання шпаруватості ШІМ-сигналу частотою 50 Гц. Для превентивної вентиляції інтегровано модуль двигуна постійного струму 130 під

керуванням спеціалізованого Н-мостового драйвера HR1124S. Світлозвукове сповіщення реалізовано за допомогою пасивного п'єзоелектричного зумера (Passive Buzzer) та чотирьохпиксельного адресного RGB-модуля SK6812 (протокол однополярного коду нульового відображення), що дозволяє динамічно змінювати колористику та яскравість середовища за допомогою одного сигнального провідника.

5. Обґрунтовано та практично розгорнуто високорівневий програмний стек розробки на базі інтерпретованого операційного середовища MicroPython. Доведено, що використання MicroPython у поєднанні з кросплатформеною IDE Thonny забезпечує швидкість написання коду, спрощує архівацію файлової системи безпосередньо у Flash-пам'яті контролера через структуру boot.py та main.py, а інтерактивна консоль REPL та інструмент візуалізації Plotter дозволяють проводити миттєве налагодження та відстеження часових діаграм перетворення сигналів у реальному часі.
6. Експериментально досліджено мережеві режими бездротового зв'язку стандарту IEEE 802.11 b/g/n на базі модуля ESP32. Програмно зреалізовано та протестовано три топологічні конфігурації:
 - режим клієнта (STA) для інтеграції стенду в існуючу інфраструктуру локальної Wi-Fi мережі та комунікації з іншими вузлами;
 - режим автономної точки доступу (softAP) для прямого сполучення зі смартфоном користувача у разі відсутності зовнішнього маршрутизатора;
 - гібридний режим (STA+AP) для побудови каскадованих розподілених IoT-мереж.
7. Здійснено повний цикл відлагодження кросплатформеного керування. На базі вбудованого TCP/IP стеку мікроконтролера

розгорнуто автономний локальний веб-сервер (порт 80). Розроблений адаптивний HTML-інтерфейс дозволяє віддалено зчитувати поточний стан датчиків та надсилати асинхронні HTTP-запити для прямого керування актуаторами (реле, RGB-підсвіткою, вентилятором, сиреною, замком дверей) через веб-браузер будь-якого мобільного чи стаціонарного девайсу.

Створений прототип лабораторного стенду демонструє високу дидактичну та практичну цінність. Модульна архітектура, надійність роз'ємів комутації серії EASY Plug RJ11 та захист від помилок переполюсовки роблять комплекс повністю готовим до інтеграції в освітній процес вищих навчальних закладів. Стенд забезпечує наочну базу для виконання лабораторних робіт із дисциплін «Інтернет речей», «Архітектура мікроконтролерних систем», «Вбудовані системи (Embedded Systems)» та «Проектування інженерії програмного забезпечення» за спеціальністю 121. Комплекс стимулює розвиток у студентів навичок як низькорівневої схемотехнічної інтеграції, так і високорівневого алгоритмічного кодування кіберфізичних систем.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Василенко О. М. Використання хмарних платформ для візуалізації телеметричних даних у реальному часі. *Вісник Житомирського державного технологічного університету. Серія: Технічні науки*. 2023. № 1 (91). С. 141–148.
2. Джевага Г., Яковлев К. Створення віддаленої лабораторії для виконання практичних робіт з робототехніки, автоматизації виробництва та IoT. *Вісник Чернігівського національного педагогічного університету. Серія: Педагогічні науки*. 2025. Вип. 33 (189). С. 33–40. DOI: 10.58407/visnik.2533052.
3. Довженко Н., Мазур Н., Костюк Ю., Рзаєва С. Інтеграція IoT та штучного інтелекту в інтелектуальні транспортні системи. *Кібербезпека: освіта, наука, техніка*. 2024. № 2 (26). С. 430–444. DOI: 10.28925/2663-4023.2024.26.708.
4. Дурняк Б. В., Шевчук В. Б. *Основи алгоритмізації та програмування мікроконтролерних систем автоматизації* : навч. посіб. Львів : Українська академія друкарства, 2022. 188 с.
5. *Інтернет речей (IoT) : підручник* / В. М. Оліферчук, О. І. Оліферчук, В. І. Гуменюк та ін. ; за ред. В. М. Оліферчука. Київ : Сталь, 2022. 340 с.
6. Ковалюк О. П., Петренко І. С. *Проектування систем Інтернету речей на базі мікроконтролерів* : навч. посіб. Львів : Новий Світ—2000, 2023. 215 с.
7. Козак Р. М. Проектування інтелектуальних систем керування мікрокліматом приміщень. *Автоматика. Автоматизація. Електротехнічні комплекси та системи*. 2023. № 2. С. 63–70.
8. Кравченко Ю. В., Бойко О. В. Бездротові сенсорні мережі на основі протоколів Wi-Fi та BLE: архітектура та безпека. *Радіoeлектроніка, інформатика, управління*. 2024. № 3. С. 89–97.

9. Микитин Г. В. Порівняльний аналіз енергоефективності мікроконтролерів ESP8266 та ESP32 в автономних системах IoT. *Сучасні інформаційні технології та системи керування*. 2025. Т. 12, № 1. С. 28–35.
10. Посашков О., Цимбал О. Розроблення архітектури системи віддаленого доступу до навчального лабораторного обладнання з використанням автоматизованих рішень. *Сучасний стан наукових досліджень та технологій в промисловості*. 2024. № 3 (29). С. 64–75. DOI: 10.30837/2522-9818.2024.3.064.
11. Al-Fuqaha A., Guizani M., Mohammadi M., Aledhari M., Ayyash M. Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications. *IEEE Communications Surveys & Tutorials*. 2015. Vol. 17, No. 4. P. 2347–2376.
12. Arduino IDE: інтегроване середовище розробки : вебсайт. URL: <https://www.arduino.cc/en/software> (дата звернення: 01.06.2026).
13. Bahatskyi O., Bahatskyi V., Sokolov V. Smart Home Subsystem for Calculating the Quality of Public Utilities. *Workshop on Cybersecurity Providing in Information and Telecommunication Systems*. 2023. Vol. 3421. P. 168–173.
14. Blynk IoT Platform Documentation : вебсайт. URL: <https://docs.blynk.io/> (дата звернення: 20.04.2026).
15. DHT11/DHT22 Humidity & Temperature Sensors Guide : вебсайт / Adafruit Learning System. URL: <https://learn.adafruit.com/dht> (дата звернення: 05.05.2026).
16. ESP32 Technical Reference Manual : електрон. дані / Espressif Systems. URL: <https://www.espressif.com/en/support/documents/technical-documents> (дата звернення: 01.06.2026).
17. ESP32-WROOM-32 Datasheet : електрон. дані / Espressif Systems. Version 3.6. URL:

https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf (дата звернення: 01.05.2026).

18. Espressif IoT Development Framework (ESP-IDF): Official Documentation : вебсайт. URL: <https://docs.espressif.com/projects/esp-idf/> (дата звернення: 01.05.2026).
19. FreeRTOS Kernel Developer Documentation : вебсайт / Amazon Web Services. URL: https://www.freertos.org/Documentation/RTOS_book.html (дата звернення: 01.05.2026).
20. HC-SR04 Ultrasonic Sensor User Guide : вебсайт / Components101. URL: <https://components101.com/sensors/ultrasonic-sensor-working-pinout-datasheet> (дата звернення: 01.05.2026).
21. IEEE Standard for Information Technology – Telecommunications and Information Exchange between Systems Local and Metropolitan Area Networks – Specific Requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. IEEE Std 802.11-2020. New York : IEEE, 2020. URL: https://standards.ieee.org/standard/802_11-2020.html (дата звернення: 01.05.2026).
22. Introduction to ESP32 Platform : вебсайт / Espressif Systems. URL: <https://www.espressif.com/en/products/socs/esp32> (дата звернення: 12.03.2026).
23. Jacko P., Brna M., Kocian V., Hargas L., Durovsky F. Remote IoT Education Laboratory for Microcontrollers Based on the STM32 Chips. *Sensors*. 2022. Vol. 22, No. 4. Art. 1440. DOI: 10.3390/s22041440.
24. Kalashnikov A., Shtefan V., Pakhomova A. Remote Laboratory: Using Internet-of-Things (IoT) for E-learning. *Information and Communication Technologies in Education, Research, and Industrial Applications*. Cham : Springer, 2017. P. 43–46.

25. MQTT Essentials: A Guide to IoT Communication Protocols / HiveMQ Team. URL: <https://www.hivemq.com/mqtt-essentials/> (дата звернення: 18.04.2026).
26. MQTT Version 5.0. OASIS Standard. 2019. URL: <https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html> (дата звернення: 01.06.2026).
27. Rejón C., Martin S., Robles-Gómez A. Easy Development of Industry 4.0 Remote Labs. *Electronics*. 2024. Vol. 13, No. 8. Art. 1508. DOI: 10.3390/electronics13081508.
28. Sánchez Sánchez P. M., Martínez J. F., Skarmeta A. F. LwHBench: A Low-Level Hardware Component Benchmark and Dataset for Single Board Computers. *Internet of Things*. 2023. DOI: 10.1016/j.iot.2023.100764.
29. Santos R., Santos S. *Developing IoT Projects with ESP32: Cloud Integration and Data Visualization*. 2nd ed. New York : O'Reilly Media, 2023. 310 p.
30. Wokwi IoT Simulator : інтернет-ресурс. URL: <https://wokwi.com/> (дата звернення: 01.06.2026).
31. Zhebka V., Toliupa S., Trush A., Shved A. Methodology for Predicting Failures in a Smart Home Based on Machine Learning Methods. *Workshop on Cybersecurity Providing in Information and Telecommunication Systems*. 2024. Vol. 3654. P. 322–332.
32. Zhebka V., Toliupa S., Trush A., Shved A. Methods for Predicting Failures in a Smart Home. *Digital Economy Concepts and Technologies Workshop*. 2024. Vol. 3665. P. 70–78.

ДОДАТОК А

Лістинг коду проєкту «Димова сигналізація»

```
from machine import ADC, Pin
import time

# Turn on and configure the ADC with the range of 0-3.3V
adc=ADC(Pin(34))
adc.atten(ADC.ATTN_11DB)
adc.width(ADC.WIDTH_12BIT)

buzzer = Pin(15, Pin.OUT)
# definitions for TM1650
ADDR_DIS = 0x48 #mode command
ADDR_KEY = 0x49 #read key value command

# definitions for brightness
BRIGHT_DARKEST = 0
BRIGHT_TYPICAL = 2
BRIGHTTEST      = 7

on  = 1
off = 0

# number:0~9
NUM = [0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f]
# DIG = [0x68,0x6a,0x6c,0x6e]
DIG = [0x6e,0x6c,0x6a,0x68]
DOT = [0,0,0,0]

clkPin = 22
dioPin = 21
clk = Pin(clkPin, Pin.OUT)
dio = Pin(dioPin, Pin.OUT)

DisplayCommand = 0

def writeByte(wr_data):
    global clk,dio
    for i in range(8):
        if(wr_data & 0x80 == 0x80):
            dio.value(1)
        else:
            dio.value(0)
        clk.value(0)
        time.sleep(0.0001)
        clk.value(1)
        time.sleep(0.0001)
        clk.value(0)
        wr_data <<= 1
    return
```

```

def start():
    global clk,dio
    dio.value(1)
    clk.value(1)
    time.sleep(0.0001)
    dio.value(0)
    return

def ack():
    global clk,dio
    dy = 0
    clk.value(0)
    time.sleep(0.0001)
    dio = Pin(dioPin, Pin.IN)
    while(dio.value() == 1):
        time.sleep(0.0001)
        dy += 1
        if(dy>5000):
            break
    clk.value(1)
    time.sleep(0.0001)
    clk.value(0)
    dio = Pin(dioPin, Pin.OUT)
    return

def stop():
    global clk,dio
    dio.value(0)
    clk.value(1)
    time.sleep(0.0001)
    dio.value(1)
    return

def displayBit(bit, num):
    global ADDR_DIS
    if(num > 9 and bit > 4):
        return
    start()
    writeByte(ADDR_DIS)
    ack()
    writeByte(DisplayCommand)
    ack()
    stop()
    start()
    writeByte(DIG[bit-1])
    ack()
    if(DOT[bit-1] == 1):
        writeByte(NUM[num] | 0x80)
    else:
        writeByte(NUM[num])
    ack()
    stop()

```

```

    return

def clearBit(bit):
    if(bit > 4):
        return
    start()
    writeByte(ADDR_DIS)
    ack()
    writeByte(DisplayCommand)
    ack()
    stop()
    start()
    writeByte(DIG[bit-1])
    ack()
    writeByte(0x00)
    ack()
    stop()
    return

def setBrightness(b = BRIGHT_TYPICAL):
    global DisplayCommand,brightness
    DisplayCommand = (DisplayCommand & 0x0f)+(b<<4)
    return

def setMode(segment = 0):
    global DisplayCommand
    DisplayCommand = (DisplayCommand & 0xf7)+(segment<<3)
    return

def displayOnOFF(OnOff = 1):
    global DisplayCommand
    DisplayCommand = (DisplayCommand & 0xfe)+OnOff
    return

def displayDot(bit, OnOff):
    if(bit > 4):
        return
    if(OnOff == 1):
        DOT[bit-1] = 1;
    else:
        DOT[bit-1] = 0;
    return

def InitDigitalTube():
    setBrightness(2)
    setMode(0)
    displayOnOFF(1)
    for _ in range(4):
        clearBit(_)
    return

def ShowNum(num): #0~9999
    displayBit(1,num%10)

```

```

    if(num < 10):
        clearBit(2)
        clearBit(3)
        clearBit(4)
    if(num > 9 and num < 100):
        displayBit(2,num//10%10)
        clearBit(3)
        clearBit(4)
    if(num > 99 and num < 1000):
        displayBit(2,num//10%10)
        displayBit(3,num//100%10)
        clearBit(4)
    if(num > 999 and num < 10000):
        displayBit(2,num//10%10)
        displayBit(3,num//100%10)
        displayBit(4,num//1000)

InitDigitalTube()

while True:
    adcVal=adc.read()
    print(adcVal)
    ShowNum(adcVal)
    if adcVal > 1000:
        buzzer.value(1)
    else:
        buzzer.value(0)
    time.sleep(0.1)

```

ДОДАТОК Б

Лістинг коду для проєкту «IoT WiFi Веб-керування «Розумне життя»»

```
import machine
import network
import socket
import time

from machine import Pin, PWM

# REPLACE WITH YOUR NETWORK CREDENTIALS(Put your SSID & Password)
ssid = 'REPLACE_WITH_YOUR_SSID'          # Enter router name
password = 'REPLACE_WITH_YOUR_PASSWORD' # Enter router password
buzzer = machine.Pin(18, machine.Pin.OUT) # Set the GPIO pin to
output (assuming the active buzzer is connected to GPIO18)
motora = machine.Pin(13, machine.Pin.OUT) # Set GPIO pin to output
(assuming DC motor IN+ is connected to GPIO13)
motorb = machine.Pin(12, machine.Pin.OUT) # Set GPIO pin to output
(assuming DC motor IN-connected IN GPIO12)
relay = machine.Pin(25, machine.Pin.OUT) # Set the GPIO pin to
output (assuming the relay is connected to GPIO25)


# Define the GPIO pins for the RGB LED
RED_PIN = 27
GREEN_PIN = 16
BLUE_PIN = 17
# Set up the PWM channels
red = PWM(Pin(RED_PIN))
green = PWM(Pin(GREEN_PIN))
blue = PWM(Pin(BLUE_PIN))


# Set the PWM frequency
red.freq(1000)
green.freq(1000)
blue.freq(1000)


# Map input values from one range to another
def interval_mapping(x, in_min, in_max, out_min, out_max):
```

```

        return (x - in_min) * (out_max - out_min) / (in_max - in_min)
+ out_min

# Convert color values (0-255) to duty cycle values (0-1023)
def color_to_duty(rgb_value):
    rgb_value = int(interval_mapping(rgb_value,0,255,0,1023))
    return rgb_value

def set_color(red_value,green_value,blue_value):
    red.duty(color_to_duty(red_value))
    green.duty(color_to_duty(green_value))
    blue.duty(color_to_duty(blue_value))

# Create a PWM (Pulse Width Modulation) object on Pin 4
servo = machine.PWM(machine.Pin(4))

# Set the frequency of the PWM signal to 50 Hz, common for servos
servo.freq(50)

# Define a function for interval mapping
def interval_mapping(x, in_min, in_max, out_min, out_max):
    return (x - in_min) * (out_max - out_min) / (in_max - in_min)
+ out_min

# Define a function to write an angle to the servo
def servo_write(pin, angle):

    pulse_width = interval_mapping(angle, 0, 180, 0.5, 2.5) #
Calculate the pulse width
    duty = int(interval_mapping(pulse_width, 0, 20, 0, 1023))
# Calculate the duty cycle
    pin.duty(duty) # Set the duty cycle of the PWM signal

def connect_wifi():
    wlan = network.WLAN(network.STA_IF)

```

```

wlan.active(True)
wlan.connect(ssid, password)
while not wlan.isconnected():
    time.sleep(1)
print('Connected to WiFi, IP:', wlan.ifconfig()[0])

connect_wifi()

# Create a simple Web server
def web_page():
    html = """
    <html>
    <body>
    <h1>ESP32 Web Server</h1>
    <p style="font-size:7vw;">Click <a href="/A">here</a> turn
on Relay<br></p>
    <p style="font-size:7vw;">Click <a href="/B">here</a> turn
off Relay<br></p>
    <p style="font-size:7vw;">Click <a href="/C">here</a> turn
on RGB<br></p>
    <p style="font-size:7vw;">Click <a href="/D">here</a> turn
off RGB<br></p>
    <p style="font-size:7vw;">Click <a href="/E">here</a> turn
on fan<br></p>
    <p style="font-size:7vw;">Click <a href="/F">here</a> turn
off fan<br></p>
    <p style="font-size:7vw;">Click <a href="/G">here</a> turn
on buzzer<br></p>
    <p style="font-size:7vw;">Click <a href="/H">here</a> turn
off buzzer<br></p>
    <p style="font-size:7vw;">Click <a href="/I">here</a>
<br>servo turn to 180</p>
    <p style="font-size:7vw;">Click <a href="/J">here</a>
<br>servo turn to 0</p>
    </body>

```

```

    </html>
    """
    return html

# Start a TCP server
addr = socket.getaddrinfo('0.0.0.0', 80)[0][-1]
s = socket.socket()
s.bind(addr)
s.listen(1)
print('Listening on', addr)

set_color(0, 0, 0)
servo_write(servo, 0)

while True:
    cl, addr = s.accept()
    print('Client connected from', addr)
    request = cl.recv(1024)
    request = str(request)

    if 'GET /A' in request:
        relay.value(1) # Relay pull
    if 'GET /B' in request:
        relay.value(0) # Relay off
    if 'GET /C' in request:
        # basic colors
        set_color(255, 0, 0) # Red
        time.sleep(1) # Wait for 1 second
        set_color(0, 255, 0) # Green
        time.sleep(1) # Wait for 1 second
        set_color(0, 0, 255) # Blue
        time.sleep(1) # Wait for 1 second
        # blended colors
        set_color(255, 0, 252) # Magenta
        time.sleep(1) # Wait for 1 second

```

```

set_color(237, 109, 0) # Orange
time.sleep(1)          # Wait for 1 second
set_color(255, 215, 0) # Yellow
time.sleep(1)          # Wait for 1 second
set_color(34, 139, 34) # Forest Green
time.sleep(1)          # Wait for 1 second
set_color(0, 112, 255) # Light Blue
time.sleep(1)          # Wait for 1 second
set_color(0, 46, 90)   # Indigo
time.sleep(1)          # Wait for 1 second
set_color(128, 0, 128) # Purple
time.sleep(1)          # Wait for 1 second
if 'GET /D' in request:
    set_color(0, 0, 0) # Black
if 'GET /E' in request:
    motora.value(1) # Turn on the motor
    motorb.value(0)
if 'GET /F' in request:
    motora.value(0) # Turn off the motor
    motorb.value(0)
if 'GET /G' in request:
    buzzer.value(1) # The buzzer buzzes
if 'GET /H' in request:
    buzzer.value(0) # The buzzer does not sound
if 'GET /I' in request:
    servo_write(servo, 180) # the Servo turns to 180°
if 'GET /J' in request:
    servo_write(servo, 0) # the Servo turns to 0°
cl.send('HTTP/1.1 200 OK\r\n')
cl.send('Content-Type: text/html\r\n\r\n')
cl.send(web_page())
cl.close()

```